
# Plataforma de coleccionismo de cartas digitales

**GRADO EN INGENIERÍA INFORMÁTICA DEL SOFTWARE**

TRABAJO FIN DE GRADO

**AUTOR**

Paula Suárez Prieto

**TUTOR**

Hugo Lebredo Buján

Julio 2024

Teaching Innovation Project: PINN-19-A-029 (University of Oviedo)
This work has been published in [1] [2]

# Declaración Responsable

La alumna Paula Suárez Prieto

Con DNI:

Y UO:

Declara que esta obra es completamente original y se han citado debidamente las fuentes utilizadas durante la realización de la misma.

Y para que así conste, firma la presente declaración en Oviedo, a 08 de julio de 2024.

# Índice general

**1. INTRODUCCIÓN** **17**

1.1. RESUMEN . . . 17
1.2. PALABRAS CLAVE . . . 17
1.3. *ABSTRACT* . . . 17
1.4. *KEYWORDS* . . . 17

**2. PLANIFICACIÓN DEL SISTEMA DE INFORMACIÓN** **18**

2.1. INICIO DEL PLAN DE SISTEMAS DE INFORMACIÓN . . . 19
2.1.1. Análisis de la Necesidad del Plan de Sistemas de Información . . . 19
2.1.2. Identificación del Alcance del Plan de Sistemas de Información . . . 19
2.1.3. Determinación de Responsables . . . 20
2.2. DEFINICIÓN Y ORGANIZACIÓN DEL PLAN DE SISTEMAS DE INFORMACIÓN . . . 21
2.2.1. Especificación del Ámbito y Alcance . . . 21

**3. ESTUDIO DE VIABILIDAD DEL SISTEMA** **22**

3.1. ANÁLISIS DE SISTEMAS SIMILARES . . . 23
3.1.1. Análisis de sistemas de intercambio de cartas digitales . . . 23
3.1.2. Análisis de sistemas de subastas en línea . . . 32
3.1.3. El mercado de coleccionistas de Pokémon . . . 37
3.2. VALORACIÓN DE ALTERNATIVAS DE SOLUCIÓN . . . 41
3.2.1. Valoración de alternativas para la arquitectura . . . 41
3.2.2. Valoración de alternativas para el Backend . . . 44
3.2.3. Valoración de alternativas para el Frontend . . . 46
3.2.4. Valoración de alternativas para la Base de Datos . . . 48
3.2.5. Valoración de alternativas de proveedor de Servicios Cloud . . . 50

**4. DEFINICIÓN DE LA ARQUITECTURA TECNOLÓGICA 53**

4.1. SELECCIÓN DE LA ARQUITECTURA TECNOLÓGICA . . . 54

**5. PLANIFICACIÓN Y GESTIÓN DEL TFG 56**

5.1. PLANIFICACIÓN DEL PROYECTO . . . 57

5.1.1. Identificación de Interesados . . . 57

5.1.2. OBS y PBS . . . 57

5.1.3. Planificación Inicial. WBS . . . 68

5.1.4. Riesgos . . . 88

5.1.5. Presupuesto . . . 89

5.2. EJECUCIÓN DEL PROYECTO . . . 114

5.2.1. Plan Seguimiento de Planificación . . . 114

5.2.2. Bitácora de Incidencias del Proyecto . . . 114

5.2.3. Riesgos . . . 115

5.3. CIERRE DEL PROYECTO . . . 116

5.3.1. Planificación Final . . . 116

5.3.2. Informe Final de Riesgos . . . 122

5.3.3. Presupuesto Final de Costes . . . 122

5.3.4. Presupuesto Final de Cliente . . . 122

5.3.5. Análisis desviaciones de presupuesto . . . 124

5.3.6. Informe de Lecciones Aprendidas . . . 125

**6. ANÁLISIS y DISEÑO DEL SISTEMA DE INFORMACIÓN 127**

6.1. DEFINICIÓN DEL SISTEMA . . . 128

6.1.1. Determinación del Alcance del Sistema . . . 128

6.1.2. Identificación de Actores del Sistema . . . 128

6.2. ESTABLECIMIENTO DE REQUISITOS . . . 130

6.2.1. Obtención de los Requisitos del Sistema . . . 130

6.3. ESPECIFICACIÓN DE CASOS DE USO . . . 140

6.3.1. Casos de uso. Registrarse e iniciar sesión . . . 140

6.3.2. Casos de uso. Gestión de perfil . . . 145

6.3.3. Casos de uso. Gestión de colección de cartas . . . 151

6.3.4. Casos de uso. Gestión de subastas y pujas . . . 155

6.3.5. Casos de uso. Gestión de transacciones . . . 167

6.3.6. Casos de uso. Gestión de notificaciones . . . 169
6.4. IDENTIFICACIÓN DE SUBSISTEMAS DE ANÁLISIS . . . 171
6.4.1. Descripción de los Subsistemas . . . 171
6.4.2. Descripción de los Interfaces entre Subsistemas . . . 172
6.5. ARQUITECTURA DE LOS SUBSISTEMAS DE ANÁLISIS . . . 174
6.5.1. Diagrama de paquetes . . . 174
6.5.2. Descripción de los Paquetes . . . 175
6.5.3. Diagramas de Componentes . . . 176
6.5.4. Diagrama de Despliegue . . . 204
6.6. DEFINICIÓN DE INTERFACES DE USUARIO . . . 207
6.6.1. Descripción de la Interfaz . . . 207
6.6.2. Definición del aspecto de la interfaz . . . 219
6.6.3. Descripción del Comportamiento de la Interfaz . . . 230
6.6.4. Diagrama de Navegabilidad . . . 233
6.7. ESPECIFICACIÓN DEL PLAN DE PRUEBAS . . . 237
6.7.1. Especificación del Plan de Pruebas . . . 237
6.7.2. Resultados de las Pruebas . . . 238
6.8. DISEÑO FÍSICO DE DATOS . . . 255

**7. CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN 258**
7.1. PREPARACIÓN DEL ENTORNO DE GENERACIÓN Y CONSTRUCCIÓN . . . 259
7.1.1. Herramientas y programas usados para el desarrollo . . . 259
7.2. ELABORACIÓN DE LOS MANUALES DE USUARIO . . . 266
7.2.1. Manual de Instalación y Ejecución . . . 266
7.2.2. Manual de Usuario . . . 267
7.3. CARGA INICIAL DE DATOS . . . 281

**8. CONCLUSIONES Y AMPLIACIONES 282**
8.1. AMPLIACIONES . . . 283
8.2. CONCLUSIONES . . . 285

**ANEXOS** **286**

PLAN DE GESTIÓN DE RIESGOS . . . . . 287

REGISTRO DE RIESGOS . . . . . 291

PRUEBAS DE ACCESIBILIDAD . . . . . 302

PRUEBAS DE ADAPTABILIDAD . . . . . 311

GLOSARIO DE VARIABLES . . . . . 321

CONTENIDO ENTREGADO EN LOS ANEXOS . . . . . 322

# Índice de figuras

3.1. Página de traspasos de jugadores de EA SPORTS FC™ 24 Companion . . . . . 24
3.2. Página de compra de sobres de EA SPORTS FC™ 24 Companion . . . . . 25
3.3. Página de mercado de jugadores de EA SPORTS FC™ Mobile Fútbol . . . . . 26
3.4. Página de intercambio de cartas de EA SPORTS FC™ Mobile Fútbol . . . . . 27
3.5. Página de mercado de jugadores de MyNBA 2K Companion App . . . . . 29
3.6. Página de colección de jugadores de NBA 2K24 MyTEAM . . . . . 29
3.7. Página de mercado de jugadores de LaLiga Fantasy . . . . . 31
3.8. Página de subastas de la sección de coleccionismo de cartas de eBay . . . . . 33
3.9. Página de subastas de la sección de coleccionismo de cartas Pokémon de Catawiki . . . . . 35
3.10. Página de subasta de una carta de Pokémon en Catawiki . . . . . 36
3.11. Página de cartas de Pokémon de PokemonPrice . . . . . 38
3.12. Página de una carta de Pokémon de PokemonPrice . . . . . 38
3.13. Página de cartas de Pokémon de PriceCharting . . . . . 39
3.14. Página de una carta de Pokémon de PriceCharting . . . . . 39
3.15. Arquitectura Monolítica . . . . . 42
3.16. Arquitectura de Microservicios . . . . . 43
3.17. Arquitectura de REST API y WebApp . . . . . 43

4.1. Arquitectura REST API y Web App . . . . . 54
4.2. MERN Stack: MongoDB, Express, React, Node.js . . . . . 55

5.1. OBS del proyecto . . . . . 58
5.2. PBS. Visión general . . . . . 63
5.3. PBS. Análisis del sistema . . . . . 64
5.4. PBS. Seguimiento del sistema . . . . . 64
5.5. PBS. Diseño del sistema . . . . . 65

5.6. PBS. Implementación del sistema . . . 66
5.7. PBS. Pruebas del sistema . . . 66
5.8. PBS. Despliegue del sistema . . . 67
5.9. PBS. Documentación del sistema . . . 67
5.10. WBS. Visión general . . . 68
5.11. WBS. Análisis del proyecto . . . 69
5.12. WBS. Seguimiento del sistema . . . 69
5.13. WBS. Diseño del sistema . . . 70
5.14. WBS. Implementación del sistema . . . 71
5.15. WBS. Fase de pruebas . . . 72
5.16. WBS. Despliegue del sistema . . . 72
5.17. WBS. Documentación . . . 73
5.18. Planificación inicial. Línea temporal . . . 74
5.19. Planificación inicial. Diagrama de Gantt, visión general . . . 75
5.20. Planificación inicial. Diagrama de Gantt, análisis del proyecto . . . 76
5.21. Planificación inicial. Diagrama de Gantt, seguimiento del proyecto . . . 77
5.22. Planificación inicial. Diagrama de Gantt, diseño del sistema *backend* . . . 79
5.23. Planificación inicial. Diagrama de Gantt, diseño del sistema *frontend* . . . 80
5.24. Planificación inicial. Diagrama de Gantt, diseño de pruebas . . . 80
5.25. Planificación inicial. Diagrama de Gantt, implementación del sistema *backend* . . . 82
5.26. Planificación inicial. Diagrama de Gantt, implementación del sistema *frontend* . . . 83
5.27. Planificación inicial. Diagrama de Gantt, implementación de pruebas . . . 84
5.28. Planificación inicial. Diagrama de Gantt, fase de pruebas . . . 85
5.29. Planificación inicial. Diagrama de Gantt, despliegue del sistema . . . 86
5.30. Planificación inicial. Diagrama de Gantt, documentación del proyecto . . . 88

6.1. Casos de uso. Registrarse e iniciar sesión . . . 140
6.2. Casos de uso. Gestión de perfil . . . 145
6.3. Casos de uso. Gestión de colección de cartas . . . 151
6.4. Casos de uso. Gestión de subastas y pujas . . . 155
6.5. Casos de uso. Gestión de transacciones . . . 167
6.6. Casos de uso. Gestión de notificaciones . . . 169
6.7. Diagrama de Paquetes del Sistema . . . 175
6.8. Diagrama de componentes del subsistema restapi . . . 177

6.9. Diagrama de componentes del subsistema webapp . . . 201
6.10. Diagrama de Despliegue de la Aplicación . . . 206
6.11. Boceto de la página de inicio . . . 208
6.12. Boceto de la página de inicio de sesión . . . 209
6.13. Boceto de la página de registro . . . 210
6.14. Boceto de la página inicial de usuario . . . 211
6.15. Boceto de la página de colección . . . 212
6.16. Boceto de la página de detalles de carta . . . 213
6.17. Boceto de la página de creación de subasta . . . 214
6.18. Boceto de la página de subastas . . . 215
6.19. Boceto de la página de detalles de subasta . . . 216
6.20. Boceto de la página de detalles de subasta propia . . . 217
6.21. Boceto de la página de detalles de subasta en la que el usuario ha pujado . . . 218
6.22. Boceto de la página de histórico de transacciones . . . 219
6.23. Página Home, página de inicio de la aplicación. . . . 220
6.24. Página de información sobre la aplicación. . . . 221
6.25. Página de inicio de sesión. . . . 221
6.26. Página de registro de usuario. . . . 222
6.27. Página principal de la aplicación, una vez que el usuario ha iniciado sesión. . . . 222
6.28. Página de la colección de cartas del usuario. . . . 223
6.29. Página de detalle de una carta de la colección del usuario. . . . 223
6.30. Página de las subastas activas de cartas. . . . 224
6.31. Página de detalle de una subasta activa. . . . 225
6.32. Página de detalle de una subasta activa creada por el usuario. . . . 225
6.33. Página de pujas activas del usuario en subastas. . . . 226
6.34. Página de la tienda de sobres de cartas. . . . 226
6.35. Página de recarga de saldo de la aplicación. . . . 227
6.36. Página de transacciones realizadas por el usuario. . . . 227
6.37. Página de perfil del usuario. . . . 228
6.38. Página de perfil del usuario, con la opción de cambiar la imagen de perfil desplegada. . . . 228
6.39. Página de notificaciones del usuario. . . . 229
6.40. Página Home en tema oscuro. . . . 229
6.41. Página de pujas activas del usuario en tema oscuro. . . . 230

6.42. Modal de creación de subasta. . . . 231
6.43. Modal de confirmación de subasta. . . . 231
6.44. Mensaje de éxito de creación de subasta. . . . 232
6.45. Alerta de que la carta está en subasta. . . . 232
6.46. Modal de confirmación de compra de sobre. . . . 233
6.47. Modal de cartas obtenidas en el sobre. . . . 233
6.48. Diagrama de navegabilidad del usuario no autenticado. . . . 234
6.49. Diagrama de navegabilidad del usuario autenticado. . . . 235
6.50. Diagrama de navegabilidad del administrador. . . . 236
6.51. Cobertura de Código del Subsistema restapi . . . . 239
6.52. Resultados de las pruebas unitarias automáticas de la webapp . . . . 246
6.53. Resultados de las pruebas de integración automáticas de la webapp . . . . 246
6.54. Modelo de datos de la base de datos MongoDB . . . . 257

7.1. Logo de TypeScript . . . . 259
7.2. Logo de Node.js . . . . 259
7.3. Logo de MongoDB . . . . 260
7.4. Logo de Microsoft Azure . . . . 260
7.5. Logo de React . . . . 260
7.6. Logo de Material-UI . . . . 260
7.7. Logo de Express . . . . 261
7.8. Logo de Git . . . . 262
7.9. Logo de GitHub . . . . 262
7.10. Logo de GitKraken . . . . 262
7.11. Logo de Visual Studio Code . . . . 262
7.12. Logo de Notepad++ . . . . 263
7.13. Logo de MobaXterm . . . . 263
7.14. Logo de Postman . . . . 263
7.15. Logo de LaTeX . . . . 264
7.16. Logo de Overleaf . . . . 264
7.17. Logo de Lucidchart . . . . 264
7.18. Logo de Draw.io . . . . 264
7.19. Logo de PlantUML . . . . 264
7.20. Logo de Excalidraw . . . . 264

7.21. Logo de Microsoft Excel . . . 265
7.22. Logo de Microsoft Project . . . 265
7.23. Página de inicio de sesión. . . . 267
7.24. Página de registro de usuario. . . . 268
7.25. Página principal de la aplicación, una vez que el usuario ha iniciado sesión. . . . 268
7.26. Página de la tienda de sobres de cartas. . . . 269
7.27. Modal de cartas obtenidas en el sobre. . . . 269
7.28. Página de la colección de cartas del usuario. . . . 270
7.29. Página de detalle de una carta de la colección del usuario. . . . 271
7.30. Modal de creación de subasta. . . . 271
7.31. Página de las subastas activas de cartas. . . . 272
7.32. Página de detalle de una subasta activa. . . . 273
7.33. Modal de creación de puja. . . . 274
7.34. Página de detalle de una subasta activa creada por el usuario. . . . 274
7.35. Página de las pujas activas del usuario. . . . 275
7.36. Página de detalle de una puja activa. . . . 275
7.37. Página de transacciones realizadas por el usuario. . . . 276
7.38. Página de recarga de saldo de la aplicación. . . . 276
7.39. Indicador de notificaciones pendientes. . . . 277
7.40. Página de notificaciones del usuario. . . . 277
7.41. Página de perfil del usuario. . . . 278
7.42. Página de perfil del usuario, con la opción de cambiar la imagen de perfil desplegada. . . . 278
7.43. Página de subastas activas para el administrador. . . . 279
7.44. Diálogo informativo de cierre de subastas para el administrador. . . . 280

8.1. Metodología de gestión de riesgos aplicada . . . 287
8.2. RBS, desglose de categorías de riesgos del proyecto . . . 288
8.3. Matriz de probabilidad vs impacto . . . 289
8.4. Accesibilidad Página Home . . . 302
8.5. Accesibilidad Página Acerca de . . . 303
8.6. Accesibilidad Página Login . . . 303
8.7. Accesibilidad Página Registro . . . 304
8.8. Accesibilidad Página Perfil . . . 305
8.9. Accesibilidad Página principal de usuario autenticado . . . 305

8.10. Accesibilidad Página Colección de cartas del usuario . . . 306
8.11. Accesibilidad Página Tienda . . . 307
8.12. Accesibilidad Página Recarga de saldo . . . 307
8.13. Accesibilidad Página Subastas activas . . . 308
8.14. Accesibilidad Página Detalle de una carta . . . 309
8.15. Accesibilidad Página Pujas activas . . . 309
8.16. Accesibilidad Página Historial de transacciones . . . 310
8.17. Accesibilidad Página Historial de transacciones - Error en etiqueta de formulario . . . 310
8.18. Adaptabilidad Página Home . . . 311
8.19. Adaptabilidad Página Login . . . 312
8.20. Adaptabilidad Página Registro . . . 312
8.21. Adaptabilidad Página Acerca de . . . 313
8.22. Adaptabilidad Página Perfil . . . 313
8.23. Adaptabilidad Página principal de usuario autenticado . . . 314
8.24. Adaptabilidad Página Colección de cartas del usuario . . . 314
8.25. Adaptabilidad Página Tienda . . . 315
8.26. Adaptabilidad Página Detalle de una carta . . . 315
8.27. Adaptabilidad Página Subastas activas . . . 316
8.28. Adaptabilidad Página Detalle de una subasta . . . 316
8.29. Adaptabilidad Página Subastas propias . . . 317
8.30. Adaptabilidad Página Mis pujas activas . . . 317
8.31. Adaptabilidad Página Recarga de saldo . . . 318
8.32. Adaptabilidad Página Historial de transacciones . . . 318
8.33. Adaptabilidad Detalle de una transacción . . . 319
8.34. Adaptabilidad Página Actualizar perfil . . . 319
8.35. Adaptabilidad Menú general de la aplicación . . . 320

# Índice de tablas

3.1. Comparación de Arquitecturas de Software . . . . . 44
3.2. Comparación de Tecnologías para el Backend . . . . . 46
3.3. Análisis Comparativo de Frameworks y Bibliotecas de Frontend . . . . . 48
3.4. Comparativa entre MySQL, MongoDB y PostgreSQL . . . . . 50

5.1. Roles y abreviaturas de los recursos . . . . . 58
5.2. Tabla RACI de la fase Análisis del proyecto . . . . . 59
5.3. Tabla RACI de la fase Seguimiento del proyecto . . . . . 59
5.4. Tabla RACI de la fase Diseño del sistema . . . . . 60
5.5. Tabla RACI de la fase Implementación del sistema . . . . . 61
5.6. Tabla RACI de la Fase de pruebas . . . . . 62
5.7. Tabla RACI de la fase Despliegue del sistema . . . . . 62
5.8. Tabla RACI de la fase Documentación del proyecto . . . . . 62
5.9. Planificación inicial. Visión general . . . . . 74
5.10. Planificación inicial. Análisis del proyecto . . . . . 75
5.11. Planificación inicial. Seguimiento del proyecto . . . . . 76
5.12. Planificación inicial. Diseño del sistema . . . . . 77
5.13. Planificación inicial. Implementación del sistema . . . . . 81
5.14. Planificación inicial. Fase de pruebas . . . . . 84
5.15. Planificación inicial. Despliegue del sistema . . . . . 85
5.16. Planificación inicial. Documentación del proyecto . . . . . 86
5.17. Costes salariales del proyecto . . . . . 90
5.18. Asignación de los costes salariales a costes directos e indirectos . . . . . 90
5.19. Número de horas productivas por perfil y en total . . . . . 91
5.20. Costes de servicios . . . . . 91
5.21. Costes de los medios de producción . . . . . 92

5.22. Precios por hora de trabajo y facturación total de la empresa . . . 94
5.23. Precio por hora a usar para el cálculo de los proyectos de costes . . . 94
5.24. Resumen del modelo de empresa . . . 95
5.25. Resumen presupuesto de costes del proyecto . . . 95
5.26. Presupuesto Inicial. Partida 1: Análisis del proyecto . . . 96
5.27. Presupuesto Inicial. Partida 2: Seguimiento del sistema . . . 96
5.28. Planificación final. Diseño del sistema . . . 98
5.29. Planificación final. Diseño del sistema . . . 102
5.30. Presupuesto Inicial. Partida 5: Fase de pruebas . . . 105
5.31. Presupuesto Inicial. Partida 6: Fase de despliegue . . . 105
5.32. Planificación final. Documentación del sistema . . . 107
5.33. Única reserva global de 2 semanas . . . 111
5.34. Reservas de gestión . . . 111
5.35. Cálculo de porcentaje a incrementar . . . 112
5.36. Presupuesto inicial del cliente . . . 112
5.37. Bitácora de incidencias del proyecto . . . 114
5.38. Planificación final. Visión general . . . 116
5.39. Planificación final. Análisis del proyecto . . . 116
5.40. Planificación final. Seguimiento del proyecto . . . 117
5.41. Planificación final. Diseño del sistema . . . 117
5.42. Planificación final. Implementación del sistema . . . 119
5.43. Planificación final. Fase de pruebas . . . 120
5.44. Planificación final. Despliegue del sistema . . . 121
5.45. Planificación final. Documentación del proyecto . . . 121
5.46. Presupuesto final de costes del proyecto. . . . 122
5.47. Presupuesto del cliente . . . 123
5.48. Comparación de desviaciones del presupuesto de costes del proyecto . . . 124
5.49. Comparación de desviaciones del presupuesto del cliente . . . 125

6.1. Caso de uso. Registrarse . . . 141
6.2. Caso de uso. Iniciar sesión . . . 142
6.3. Caso de uso. Cerrar sesión . . . 143
6.4. Caso de uso. Modificar foto de perfil . . . 145
6.5. Caso de uso. Actualizar contraseña . . . 146

6.6. Caso de uso. Recargar saldo . . . 147
6.7. Caso de uso. Histórico de transacciones realizadas . . . 149
6.8. Caso de uso. Ver colección de cartas . . . 151
6.9. Caso de uso. Ver detalle de una carta . . . 152
6.10. Caso de uso. Comprar sobre . . . 153
6.11. Caso de uso. Crear una subasta . . . 155
6.12. Caso de uso. Retirar una subasta . . . 157
6.13. Caso de uso. Consultar subastas activas . . . 158
6.14. Caso de uso. Consultar el detalle de una subasta activa . . . 159
6.15. Caso de uso. Realizar puja . . . 161
6.16. Caso de uso. Retirar puja . . . 162
6.17. Caso de uso. Consultar pujas activas . . . 163
6.18. Caso de uso. Consultar el detalle de una puja activa . . . 164
6.19. Caso de uso. Cerrar subastas finalizadas . . . 165
6.20. Caso de uso. Histórico de transacciones del sistema . . . 167
6.21. Caso de uso. Consultar bandeja de notificaciones . . . 169
6.22. Caso de uso. Marcar notificación como leída . . . 170
6.23. Descripción del componente: Server . . . 178
6.24. Descripción del componente: App . . . 178
6.25. Descripción del componente: AuthMiddleware . . . 179
6.26. Descripción del componente: AuthSocket . . . 180
6.27. Descripción del componente: AuctionRouter . . . 181
6.28. Descripción del componente: BidRouter . . . 181
6.29. Descripción del componente: CardPackRouter . . . 182
6.30. Descripción del componente: CardRouter . . . 183
6.31. Descripción del componente: NotificationRouter . . . 183
6.32. Descripción del componente: PaypalRouter . . . 184
6.33. Descripción del componente: AuctionController . . . 185
6.34. Descripción del componente: BidController . . . 187
6.35. Descripción del componente: CardController . . . 187
6.36. Descripción del componente: CardPackController . . . 188
6.37. Descripción del componente: DeckController . . . 188
6.38. Descripción del componente: NotificationController . . . 189

6.39. Descripción del componente: PaypalController . . . 190
6.40. Descripción del componente: PurchaseController . . . 190
6.41. Descripción del componente: TransactionController . . . 191
6.42. Descripción del componente: UserCardController . . . 192
6.43. Descripción del componente: UserController . . . 192
6.44. Descripción del componente: Auction . . . 193
6.45. Descripción del componente: Bid . . . 193
6.46. Descripción del componente: Card . . . 194
6.47. Descripción del componente: CardPack . . . 194
6.48. Descripción del componente: Deck . . . 195
6.49. Descripción del componente: Notification . . . 195
6.50. Descripción del componente: Transaction . . . 195
6.51. Descripción del componente: User . . . 196
6.52. Descripción del componente: UserCard . . . 196
6.53. Descripción del componente: Modelos . . . 197
6.54. Descripción del componente: FetchDeckData . . . 197
6.55. Descripción del componente: loadDecks . . . 198
6.56. Descripción del componente: FetchPokemonData . . . 198
6.57. Descripción del componente: LoadPokemonData . . . 199
6.58. Descripción del componente: LoadCardPacksData . . . 200
6.59. Tests de auctionRouter . . . 239
6.60. Tests de bidRouter . . . 240
6.61. Tests de cardPackRouter . . . 240
6.62. Tests de cardRouter . . . 240
6.63. Tests de deckRouter . . . 241
6.64. Tests de notificationRouter . . . 242
6.65. Tests de purchasesRouter . . . 242
6.66. Tests de transactionRouter . . . 243
6.67. Tests de userCardRouter . . . 243
6.68. Tests de userRouter . . . 244
6.69. Cuestionario de usabilidad de la aplicación . . . 247
6.70. Cuestionario para el responsable de pruebas . . . 250
6.71. Resultados de las pruebas de usabilidad . . . 251

8.1. Riesgo 1. Falta de comunicación con el tutor del TFG . . . 291
8.2. Riesgo 1. Seguridad de la información . . . 292
8.3. Riesgo 3. Intento de fraude por parte de los usuarios finales . . . 293
8.4. Riesgo 4. Errores en las estimaciones de tareas . . . 294
8.5. Riesgo 5. Conciliación entre responsabilidades académicas y laborales . . . 295
8.6. Riesgo 6. Aceptación del cliente . . . 296
8.7. Riesgo 7. Probelmas de cumplimiento normativo y legal . . . 297
8.8. Riesgo 8. Cambios en la licencia de software . . . 298
8.9. Riesgo 9. Problemas de rendimiento . . . 299
8.10. Riesgo 10. Fallos en el hardware o software . . . 299
8.11. Riesgo 11. Problemas de usabilidad . . . 300
8.12. Riesgo 12. Problemas de accesibilidad . . . 301
8.13. Parámetros de configuración . . . 321

# Capítulo 1

# INTRODUCCIÓN

## 1.1. RESUMEN

Las plataformas digitales han popularizado un modo de juego basado en la colección de cartas, permitiendo a los usuarios acceder a un mercado de intercambio de estas. Este modelo de juego genera miles de millones de euros al año, convirtiéndose en una de las formas de entretenimiento más destacadas en la actualidad.

Este trabajo presenta el desarrollo de una plataforma online de coleccionismo e intercambio de activos digitales, concretamente de cartas de Pokémon. Esta plataforma permitirá a los usuarios adquirir sobres de cartas con distinta rareza, lo que añade un componente de exclusividad y emoción a la colección. Además, las cartas podrán ser subastadas en tiempo real, permitiendo a los usuarios adquirir las cartas más deseadas mediante pujas, mejorando así la experiencia de coleccionismo digital.

## 1.2. PALABRAS CLAVE

Coleccionismo, subastas, cartas, Pokémon

## 1.3. *ABSTRACT*

Digital platforms have popularized a game mode based on card collecting, allowing users to access a marketplace for exchanging these cards. This gaming model generates billions of euros annually, becoming one of the most prominent forms of entertainment today.

This project presents the development of an online platform for the collection and exchange of digital assets, specifically Pokemon cards. This platform will allow users to acquire packs of cards with varying rarities, adding a component of exclusivity and excitement to the collection. Additionally, the cards can be auctioned in real-time, allowing users to acquire the most desired cards through bids, thus enhancing the digital collecting experience.

## 1.4. *KEYWORDS*

Collecting, auctions, cards, Pokemon

# Capítulo 2

# PLANIFICACIÓN DEL SISTEMA DE INFORMACIÓN

## 2.1. INICIO DEL PLAN DE SISTEMAS DE INFORMACIÓN

### 2.1.1. Análisis de la Necesidad del Plan de Sistemas de Información

Con la tecnología, la forma en la que nos relacionamos con nuestros pasatiempos ha cambiado. La popularidad de plataformas como EA Sports FC y su modo Ultimate Team[3], deja en evidencia que el mercado de coleccionismo digital está en pleno crecimiento [4]. Sin embargo, aún existe un nicho de mercado específico por explorar: una plataforma dedicada exclusivamente al coleccionismo de cartas digitales.

BidMon Universe trata de dar respuesta a esta necesidad, ofreciendo a los coleccionistas digitales una forma segura e innovadora de comprar y vender activos digitales mediante un sistema de pujas ciego.

El Plan de Sistemas de Información para este proyecto se aborda desde una perspectiva que considera tanto la arquitectura técnica como la experiencia del usuario final. De esta manera se establece un marco de trabajo que garantiza que, además de cumplir con las expectativas del mercado actual, el proyecto proporciona características y experiencias novedosas que enriquecen la dinámica del coleccionismo digital.

### 2.1.2. Identificación del Alcance del Plan de Sistemas de Información

La aplicación web a desarrollar deberá de proporcionar al usuario final las siguientes funcionalidades de alto nivel:

- Todo usuario final puede acceder a la aplicación web y consultar información genérica sobre la misma.
  - Información sobre el servicio que se ofrece.
  - Información sobre el equipo de desarrollo.
- Cualquier usuario, mayor de 18 años, podrá crear una cuenta de usuario en la aplicación.
- Cualquier usuario registrado en la aplicación puede iniciar sesión en esta.
- Los usuarios autenticados tendrán una colección de cartas digitales que podrán consultar.
- Los usuarios autenticados pueden recargar el saldo de su cuenta.
- Los usuarios autenticados tendrán acceso a un catálogo de sobres de cartas que podrán adquirir si cuentan con saldo suficiente.
- Los usuarios autenticados podrán realizar subastas y pujas ciegas sobre las cartas.
- Los usuarios autenticados podrán consultar el histórico de transacciones realizadas.
- Los usuarios autenticados podrán modificar su perfil.
- Un usuario con rol de administrador podrá acceder al histórico de subastas y cancelar las que considere fraudulentas, el sistema notificará al usuario que la ha iniciado.

En base a estas funcionalidades se extraen los siguientes **objetivos estratégicos** a lograr para alcanzar el éxito del proyecto:

- Implementar un sistema de autenticación de usuarios.

- Implementar un sistema de venta de cartas.
- Implementar un sistema de subastas ciegas que permita a los usuarios comprar y vender cartas.
- Implememntar un sistema de gestión de subastas por parte de un usuario autenticado.
- Implememntar un sistema de gestión de subastas por parte de un usuario administrador.
- Implementar un sistema de notificaciones que informe a los usuarios sobre el estado de las subastas en las que participan.
- Interfaz intuitiva que permita al usuario realizar consultas sobre el estado de las pujas y subastas, además de visualizar la colección de cartas adquirida.

### 2.1.3. Determinación de Responsables

A continuación, se detallan las personas que participan en el proyecto junto con sus funciones.

- El tutor del trabajo se encargará de la supervisión del proyecto.
- El alumno se encargará del desarrollo del proyecto y documentación de este.

## 2.2. DEFINICIÓN Y ORGANIZACIÓN DEL PLAN DE SISTEMAS DE INFORMACIÓN

### 2.2.1. Especificación del Ámbito y Alcance

Este proyecto se enfoca en el desarrollo de una plataforma, llamada BidMon Universe, diseñada para permitir a los usuarios coleccionar activos digitales, en particular, cartas coleccionables de Pokémon. Para la adquisición de estas cartas, los usuarios podrán optar por compras directas dentro de la aplicación o mediante la participación en subastas.

La plataforma incorporará una moneda virtual para facilitar las transacciones de compra y subasta de cartas. Los usuarios podrán aumentar su saldo a través de una pasarela de pago integrada. Cada usuario autenticado podrá comprar cartas y, si así lo desea, ofrecerlas en subastas ciegas, donde las ofertas de otros participantes permanecerán ocultas hasta el cierre de la misma. Este sistema de subasta a ciegas garantiza imparcialidad en el proceso de compra-venta y añade un elemento emocionante a la experiencia del usuario.

Respecto a la mecánica de subastas, al participante que realice la oferta más alta se le adjudicará la carta al concluir el periodo establecido, y el importe correspondiente se deducirá automáticamente de su saldo. El vendedor podrá establecer condiciones específicas para la subasta, como la duración y el precio inicial, y tendrá la opción de cancelar la subasta en cualquier momento.

Un sistema integral de notificaciones mantendrá a los usuarios constantemente informados sobre las actualizaciones de las subastas en las que están involucrados, ya sea como vendedores o como postores. Esto se complementa con un registro exhaustivo de todas las transacciones efectuadas en la plataforma, promoviendo una total transparencia y trazabilidad de las operaciones. Este registro no solo incluirá compras y ventas, sino también el historial completo de cada carta dentro de la plataforma.

Los usuarios tendrán acceso a un panel de control donde podrán revisar su historial de transacciones y modificar aspectos de su perfil. Por su parte, los administradores contarán con herramientas específicas para supervisar y gestionar las subastas, incluida la capacidad de cancelar aquellas que consideren fraudulentas.

Dada la complejidad del proyecto y las restricciones de tiempo para su desarrollo, se han establecido ciertas limitaciones en la versión inicial de la plataforma. No se incluirá un sistema de mensajería interna ni se permitirá la formación de conexiones sociales directas entre usuarios. Además, no se incluirán funcionalidades de jugabilidad con las cartas, ni se permitirá la exportación de las colecciones de cartas a formatos externos ni la conversión del saldo virtual en dinero real.

Finalmente, los datos iniciales de las cartas se cargarán una única vez, y no se permitirá la adición de nuevas cartas a través de la interfaz de usuario. De igual forma, no se incluirá la opción de añadir nuevos sobres de cartas a la tienda virtual desde la interfaz de usuario. Estas modificaciones se realizarán directamente en la base de datos.

# Capítulo 3

# ESTUDIO DE VIABILIDAD DEL SISTEMA

## 3.1. ANÁLISIS DE SISTEMAS SIMILARES

En este apartado, se aborda un análisis detallado de las plataformas y sistemas que actualmente exhiben un comportamiento similar al del sistema que se pretende desarrollar para 'BidMon Universe'.

La realización de este análisis es muy importante, ya que proporciona una base sólida para la toma de decisiones estratégicas. Al examinar detenidamente tanto las ventajas como las desventajas de los sistemas existentes, se identifican estrategias exitosas y se reconocen posibles fallos o áreas susceptibles de mejora. Este enfoque no solo informa las decisiones de diseño y tecnología, sino que también las fundamenta en un conocimiento práctico y bien contextualizado. Así, se garantiza que el desarrollo de 'BidMon Universe' esté armonizado con las tendencias actuales y adopte las soluciones más efectivas y adecuadas para su propósito.

El objetivo del proyecto es permitir a los usuarios coleccionar cartas e intercambiarlas mediante subastas, específicamente cartas de Pokémon. Por esta razón, se ha llevado a cabo un análisis de sistemas similares enfocados en los siguientes puntos:

- Sistemas de intercambio de cartas digitales: Se analizarán diversos sistemas de intercambio de cartas digitales, destacando su relevancia en el mercado y las funcionalidades que ofrecen.
- Sistemas de subastas en línea: Se evaluarán plataformas de subastas en línea, analizando sus ventajas y desventajas, con el objetivo de identificar las mejores prácticas aplicables a nuestro proyecto.
- El mercado de coleccionistas de Pokémon: Se estudiará la extensa base de seguidores y coleccionistas de cartas Pokémon, comprendiendo su comportamiento, preferencias y el impacto que tienen en el mercado de subastas.

### 3.1.1. Análisis de sistemas de intercambio de cartas digitales

En esta sección se analizarán los sistemas de intercambio de cartas digitales, como EA Sports FC, NBA 2K y LaLiga Fantasy, que permiten a los usuarios adquirir y vender cartas de jugadores. Aunque el sistema de coleccionismo de cartas puede variar según la plataforma del juego, todos comparten una serie de características comunes. Estas incluyen la posibilidad de pujar por un jugador o comprarlo directamente, la existencia de distintas rarezas de cartas y la inclusión de eventos especiales.

#### 3.1.1.1. EA Sports FC

EA Sports FC es una destacada franquicia de videojuegos de fútbol disponible en diversas plataformas. Este análisis se centrará en dos modos de juego: FIFA Ultimate Team y EA SPORTS FC™ Mobile Fútbol,

FIFA Ultimate Team busca transformar la interacción de los jugadores, permitiéndoles construir y gestionar su propio equipo de fútbol. Los usuarios pueden buscar jugadores en el mercado de transferibles, pujar por ellos o comprarlos directamente. Además, pueden incrementar el valor de sus jugadores completando desafíos y cambiar la apariencia de las cartas. Este modo incluye tarjetas de jugadores con distintas rarezas, haciendo que algunas sean más valiosas que otras.

Se estima que EA Sports FC ha generado más de 6.000 millones de dólares en ingresos netos entre 2020 y 2025[4]. En 2020, los ingresos casi se triplicaron en comparación con 2015, demostrando un crecimiento exponencial. El informe anual de Electronic Arts para el año fiscal 2023[5] especifica que se generaron 7.426 millones de dólares en ingresos netos y revela que el modo Ultimate Team de FIFA 23 es el más popular de la

franquicia, con más de 10 millones de jugadores activos mensuales, siendo una de las mayores fuentes de ingresos de la empresa. El informe también destaca la intención de continuar desarrollando este mercado mediante la incorporación de nuevas características para el modo Ultimate Team.

La popularidad de este modo de juego ha llevado a Electronic Arts a lanzar una aplicación móvil llamada EA SPORTS FC™ 24 Companion, que permite a los usuarios gestionar su club de FIFA Ultimate Team desde dispositivos móviles. Los usuarios necesitan tener el juego FIFA 24 para utilizar la aplicación, la cual se conecta a la cuenta del usuario en el juego.

Desde la aplicación, los usuarios pueden comprar sobres de cartas, gestionar su plantilla, comprar jugadores en el mercado de transferibles y pujar por jugadores en subastas para utilizarlos posteriormente en el juego. Esta aplicación permite a los usuarios estar al tanto de las últimas novedades y ofertas del mercado, así como recibir notificaciones en tiempo real sobre el estado de sus pujas.

En las siguientes figuras, se muestra la interfaz de usuario de EA SPORTS FC™ 24 Companion.

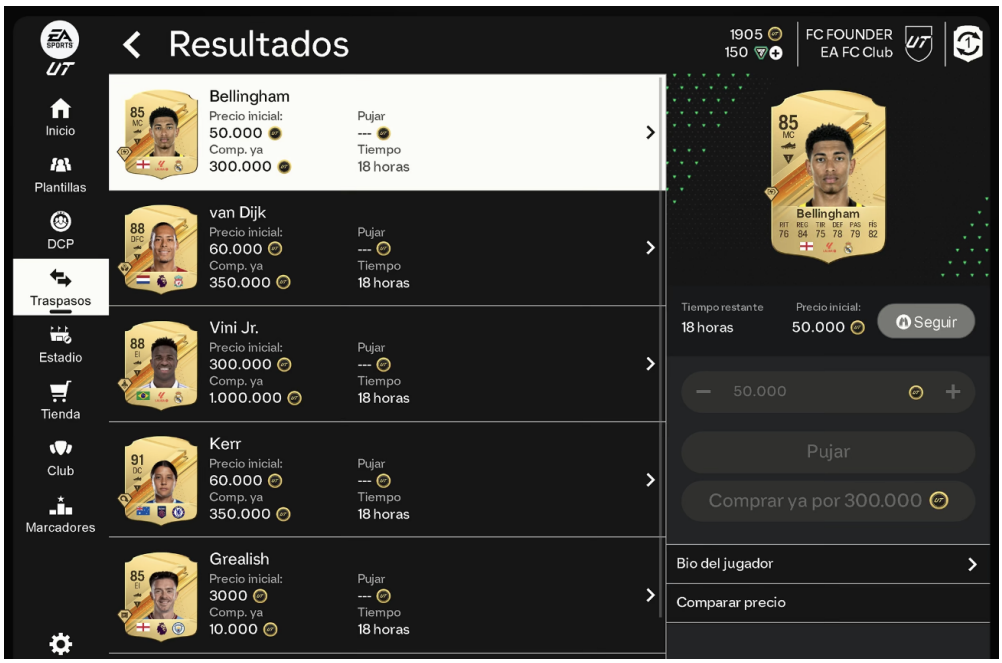


Figura 3.1: Página de traspasos de jugadores de EA SPORTS FC™ 24 Companion

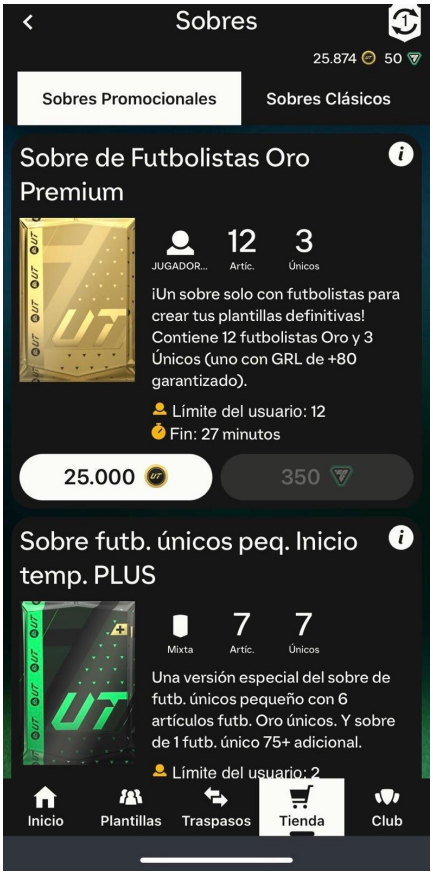


Figura 3.2: Página de compra de sobres de EA SPORTS FC™ 24 Companion

Por otra parte, EA SPORTS FC™ Mobile Fútbol es una versión adaptada de FIFA Ultimate Team para dispositivos móviles. Los usuarios pueden construir su equipo, competir en eventos y partidos en tiempo real, sin necesidad de haber comprado el juego físico. Las compras están integradas en la aplicación, permitiendo a los usuarios adquirir monedas y puntos FIFA para mejorar su equipo, pero se pueden obtener de forma gratuita a través de eventos y desafíos. Existe una tienda en la que los usuarios pueden comprar sobres de cartas por medio de dinero real o monedas del juego, así como un mercado de transferibles en el que pueden adquirir jugadores mediante subastas o compras directas. También incorpora la opción de intercambiar cartas por cartas sorpresa.

En las siguientes capturas de pantalla, se muestra la interfaz de usuario de EA SPORTS FC™ Mobile Fútbol.

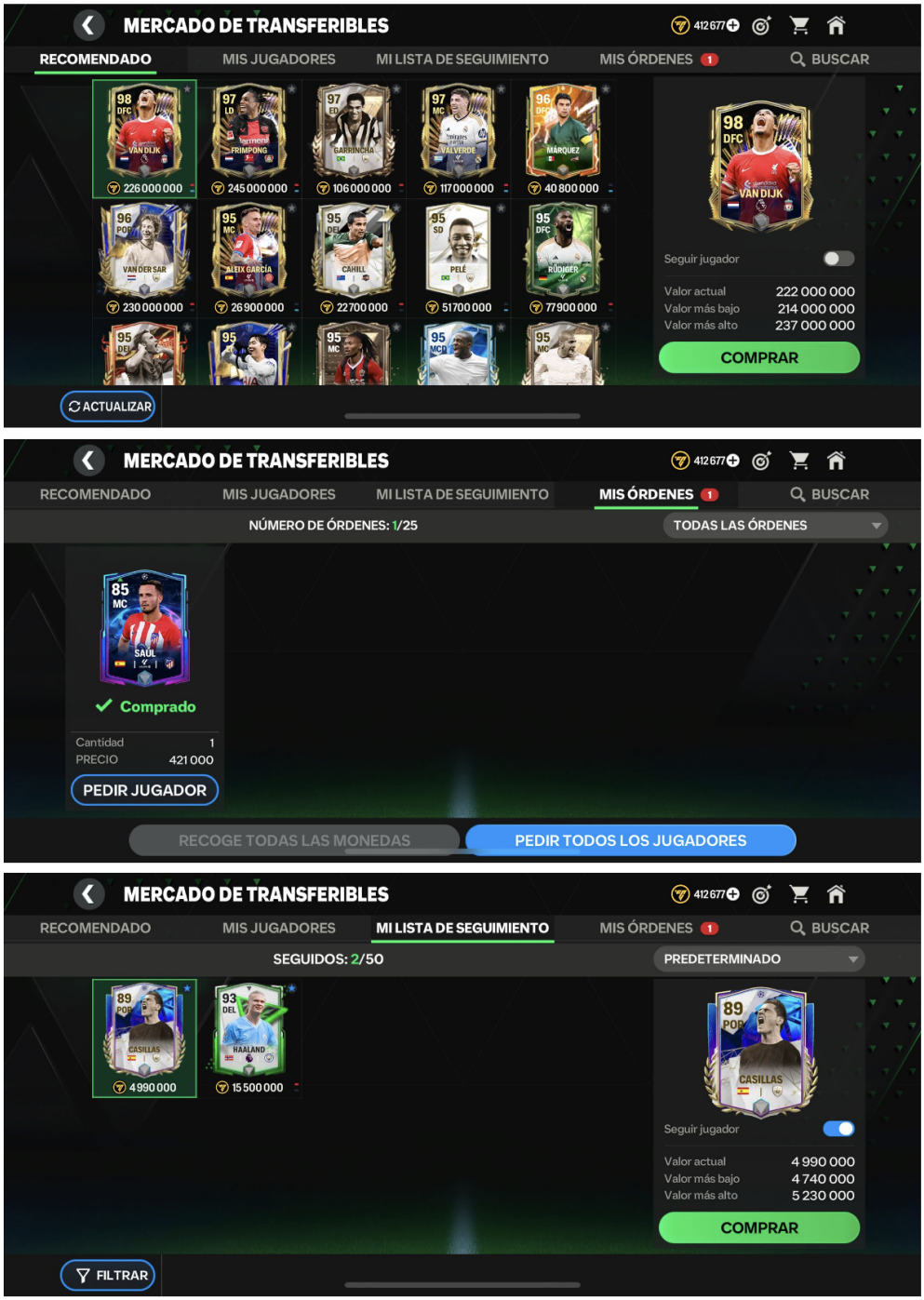


Figura 3.3: Página de mercado de jugadores de EA SPORTS FC™ Mobile Fútbol


Figura 3.4: Página de intercambio de cartas de EA SPORTS FC™ Mobile Fútbol

A continuación, se presentan las ventajas y desventajas de EA SPORTS FC™ Mobile Fútbol, así como las ventajas del nuevo sistema respecto a este. El motivo de centrarse en EA SPORTS FC™ Mobile Fútbol en lugar de la aplicación EA SPORTS FC™ 24 Companion es que la primera es una versión gratuita y accesible para todos los usuarios, además tienen características similares.

#### *3.1.1.1.1 Ventajas de EA SPORTS FC™ Mobile Fútbol*

En el contexto de EA SPORTS FC™ Mobile Fútbol, destacan las siguientes ventajas:

- Los usuarios pueden marcar una carta para efectuar un seguimiento y recibir notificaciones en caso de una disminución en su precio.
- Se proporcionan estadísticas detalladas de cada carta, incluyendo su evolución de precio en las últimas semanas, así como los valores más bajos y más altos a los que se está vendiendo actualmente.
- Existe la opción de vender una carta directamente, evitando la necesidad de utilizar el sistema de subastas.
- Cuenta con un buscador de ofertas que incorpora varios filtros.
- Los usuarios tienen la posibilidad de editar o cancelar sus pujas en curso.
- El sistema de subastas implementado opera bajo un mecanismo de puja ciega.
- La aplicación lanza ofertas de intercambio de cartas, permitiendo a los usuarios cambiar una o varias cartas que cumplan con ciertos requisitos por una carta sorpresa.

#### *3.1.1.1.2 Desventajas de EA SPORTS FC™ Mobile Fútbol*

Sin embargo, es importante mencionar algunas desventajas de EA SPORTS FC™ Mobile Fútbol, que incluyen:

- Un número limitado de órdenes de canje simultáneas como, por ejemplo, el máximo de 25 permitido en EA Sports FC Mobile.

- Las pujas solo se pueden realizar por un valor igual o superior al establecido por el juego.
- Existen varios tipos de monedas, lo que hace que no puedas conseguir ciertas cartas si no pagas por esas monedas ya que son muy díficiles de conseguir de forma gratuita.

#### *3.1.1.1.3 Ventajas del nuevo sistema respecto EA Sports FC*

El sistema en desarrollo presenta varias características que mejoran la experiencia del usuario en comparación con EA Sports FC.

- El acceso a la aplicación es completamente gratuito.
- Ofrece una mayor personalización en el proceso de puja, brindando a los usuarios una experiencia más flexible y adaptada a sus preferencias.
- El usuario tiene la posibilidad de acceder a un histórico exhaustivo en el que se detallan todas las transacciones realizadas.

### 3.1.1.2. NBA 2K

NBA 2K es una destacada franquicia de videojuegos de baloncesto disponible en diversas plataformas. En esta sección se analizarán las aplicaciones MyNBA 2K Companion App y NBA 2K24 MyTEAM.

MyTEAM busca transformar la interacción de los jugadores, permitiéndoles construir y gestionar su propio equipo de baloncesto. Los usuarios pueden buscar jugadores en la tienda de la aplicación, para mejorar su equipo y competir en eventos y partidos en tiempo real. Existen distintos niveles de cartas para añadir un elemento de emoción al juego y cartas de recompensa que se pueden obtener completando desafíos y eventos especiales.

Se estima que NBA 2K ha generado ingresos significativos en los últimos años. En el año fiscal 2023, los ingresos de Take-Two Interactive alcanzaron niveles récord gracias a sus franquicias principales, incluyendo NBA 2K. El informe anual de Take-Two Interactive para el año fiscal 2023[6] destaca que NBA 2K es una de las mayores fuentes de ingresos de la empresa, con una base de usuarios activa y comprometida, especialmente en el modo MyTEAM. En el informe se menciona la intención de seguir desarrollando este mercado ya que se espera que continúe creciendo en los próximos años.

La popularidad de este modo de juego ha llevado a Take-Two Interactive a lanzar una aplicación móvil llamada MyNBA 2K Companion App. Es una aplicación gratuita, pero los usuarios necesitan tener el juego NBA 2K para utilizarla. La aplicación permite canjear códigos a los usuarios para obtener recompensas en el juego, así como consultar estadísticas de su saldo sin necesidad de iniciar sesión en la consola. Tiene otras funciones interesantes como la posibilidad de personalizar su propio jugador por medio de la función de escaneo facial, la cual permite a los usuarios escanear su rostro y añadirlo al juego.

En la siguiente figura, se muestra la interfaz de usuario de MyNBA 2K Companion App.


Figura 3.5: Página de mercado de jugadores de MyNBA 2K Companion App

Por otra parte, NBA 2K24 MyTEAM es una versión adaptada de MyTEAM que perimite registarse y jugar sin necesidad de tener el juego NBA 2K. Los usuarios pueden construir su equipo, competir en eventos y partidos, y mejorar su equipo mediante la adquisición de jugadores en la tienda de la aplicación. No pueden realizar subastas ni vender cartas, pero se permite el intercambio de cartas repetidas por cartas sorpresa.

En las siguientes capturas de pantalla, se muestra la interfaz de usuario de NBA 2K24 MyTEAM.


Figura 3.6: Página de colección de jugadores de NBA 2K24 MyTEAM

A continuación, se presentan las ventajas y desventajas de NBA 2K24 MyTEAM, así como las ventajas del nuevo sistema respecto a este. Se ha decidido centrarse en NBA 2K24 MyTEAM en lugar de MyNBA 2K Companion App, ya que la primera es una versión gratuita y accesible para todos los usuarios.

#### *3.1.1.2.1 Ventajas de NBA 2K24 MyTEAM*

En el contexto de NBA 2K, destacan las siguientes ventajas:

- Se pueden buscar jugadores en la tienda de la aplicación y adquirirlos para mejorar el equipo.
- Existe la opción de intercambiar cartas repetidas por cartas sorpresa.

#### *3.1.1.2.2 Desventajas de NBA 2K24 MyTEAM*

Sin embargo, es importante mencionar algunas desventajas de NBA 2K, que incluyen:

- No hay un mercado de subastas en el que los usuarios puedan adquirir jugadores mediante pujas ni vender cartas.
- Existen dos tipos de monedas en el juego, una de pago y otra gratuita, lo que puede hacer que no puedas comprar cierta carta o acceder a ciertas características del juego si no pagas.

#### *3.1.1.2.3 Ventajas del nuevo sistema respecto NBA 2K*

El sistema en desarrollo presenta varias características que mejoran la experiencia del usuario en comparación con NBA 2K.

- Ofrece la opción de subasta de cartas, brindando a los usuarios una experiencia más flexible y adaptada a sus preferencias.
- Solo se utilizará una moneda en el juego, lo que facilita la adquisición de cartas y sobres.

### 3.1.1.3. LaLiga Fantasy

LaLiga Fantasy un juego basado en la liga de fútbol española, conocida como LaLiga. En este juego, los usuarios tienen la capacidad de crear equipos que se componen de jugadores cuyo rendimiento se correlaciona con el desempeño real en los partidos de LaLiga. Además, existe la posibilidad de competir por premios reales en determinadas instancias del juego. El juego dispone de un mercado en el que los usuarios pueden vender o adquirir jugadores a través de un sistema de subastas, por lo que estamos ante un escenario similar al anterior.

A continuación, se muestra la interfaz de usuario de LaLiga Fantasy.

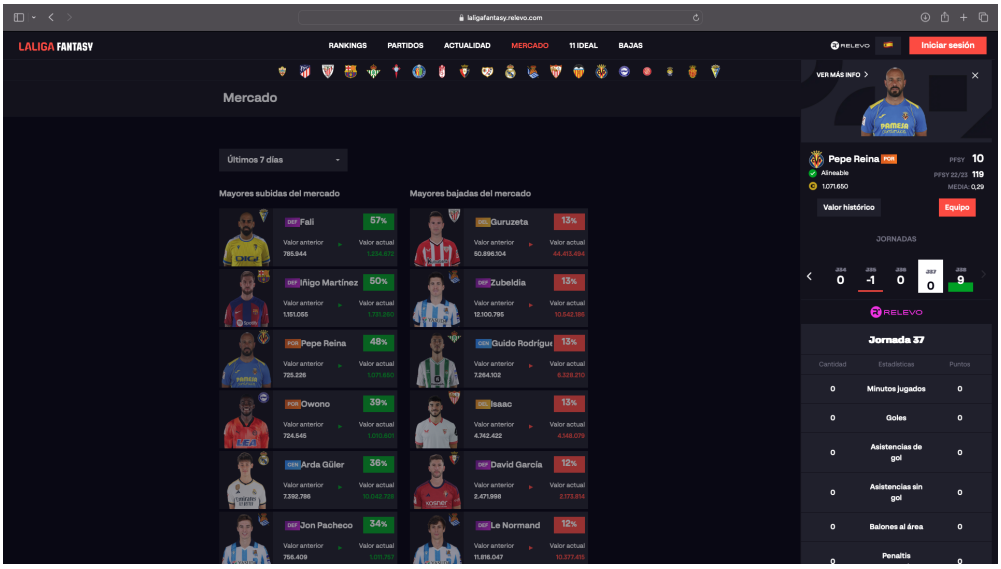


Figura 3.7: Página de mercado de jugadores de LaLiga Fantasy

#### *3.1.1.3.1 Ventajas de LaLiga Fantasy*

Dentro del marco de LaLiga Fantasy, se destacan las siguientes características:

- El juego renueva constantemente el mercado de transferibles.
- El juego cuenta con la capacidad de generar ofertas automáticas por los futbolistas que se encuentran en venta. Estas ofertas se establecen a partir de un valor aleatorio que oscila entre el valor de mercado del jugador, con un margen del 10 % tanto por encima como por debajo de dicho valor.
- Recientemente se ha incorporado el mercado de "clausulazos", donde los usuarios pueden adquirir un jugador pagando su cláusula, que será más elevada que el valor de mercado, sin tener que depender del propietario del jugador.
- Se pueden realizar ofertas por jugadores a otros usuarios.

#### *3.1.1.3.2 Desventajas de LaLiga Fantasy*

Se pueden identificar las siguientes desventajas:

- La limitación a un máximo de 24 jugadores en la plantilla, lo que puede ocasionar la pérdida de oportunidades en subastas.
- La plataforma brinda escasa flexibilidad en lo que respecta a la personalización al momento de poner un jugador en venta.

#### *3.1.1.3.3 Ventajas del nuevo sistema respecto LaLiga Fantasy*

- LaLiga Fantasy ofrece una suscripción de 0,99€/mes para poder jugar sin anuncios mientras que el sistema que se desarrollará carece de anuncios.

- Se proporciona un historial de transacciones completo para que los usuarios puedan rastrear las compras y ventas.
- Se implementa un sistema de subastas en tiempo real que, además, brinda una mayor personalización al usuario en aspectos como la duración de la subasta y los valores de venta, entre otros.
- Además de acceder al mercado, los usuarios tienen la opción de adquirir sobres de cartas, lo que añade un elemento de emoción a la aplicación.

### 3.1.2. Análisis de sistemas de subastas en línea

#### 3.1.2.1. eBay

eBay es una plataforma de comercio online que permite a los usuarios comprar y vender productos a través de subastas o ventas directas. Es una de las plataformas de venta *online* más grandes y conocidas del mundo, con una amplia variedad de productos disponibles, desde artículos de colección hasta productos de consumo diario.

En el primer trimestre de 2024, eBay ha reportado varias métricas financieras clave [7] que reflejan su desempeño sólido y su amplia adopción global. La plataforma cuenta con 132 millones de compradores activos a nivel mundial, lo que subraya su extensa base de usuarios y el interés que genera en el mercado. Los ingresos totales para el primer trimestre de 2024 fueron de 2.6 billones de dólares, demostrando la robustez del modelo de negocio de eBay. Desde esta plataforma los vendedores tienen la opción de establecer un precio fijo por su producto o permitir que los compradores pujen por él.

La plataforma ofrece dos opciones para comprar un producto. La primera es la compra directa, en la que el comprador adquiere el producto al precio establecido por el vendedor, y la segunda es la subasta, en la que los compradores pujan por el producto y el comprador con la puja más alta al final del tiempo de la subasta se lleva el producto.

En el caso de las subastas, eBay permite a los vendedores establecer el precio base, que es el precio mínimo al que están dispuestos a vender el producto y también tienen la opción de establecer un precio de compra directa, que es el precio al que un comprador puede adquirir el producto sin necesidad de pujar.

La aplicación permite a los compradores hacer seguimiento de los productos que les interesan, recibir notificaciones sobre el estado de las subastas y pujar por productos de forma automática, estableciendo un precio máximo.

Ofrece información en tiempo real sobre las subastas, como el número de pujas, el precio actual, el tiempo restante y el número de pujadores. Además, proporciona un historial de las pujas realizadas en un producto, lo que permite a los compradores ver cómo ha evolucionado el precio y quiénes han pujado. También permite a los usuarios consultar las valoraciones de los vendedores y ver los productos que tienen en venta.

En la siguiente figura, se muestra la interfaz de usuario de la sección de coleccionismo de cartas de eBay.

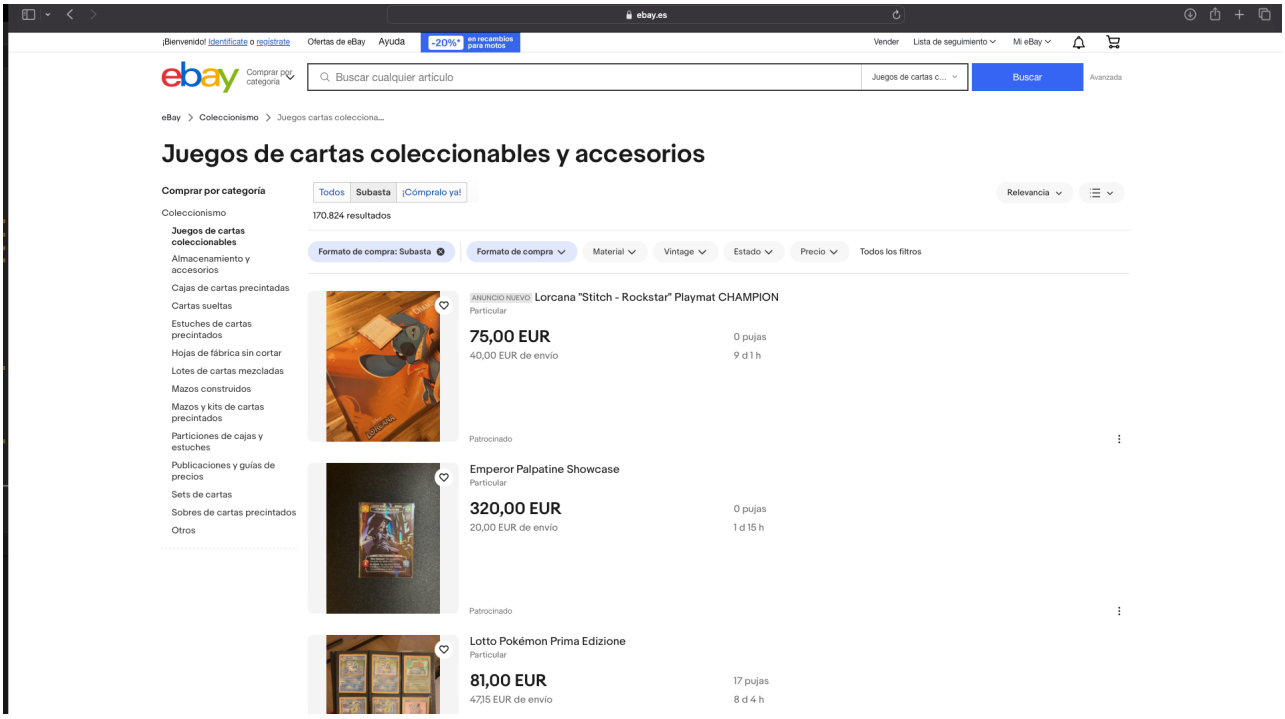


Figura 3.8: Página de subastas de la sección de coleccionismo de cartas de eBay

#### 3.1.2.1.1 *Ventajas de eBay*

Dentro del marco de LaLiga Fantasy, se destacan las siguientes características:

- Para cada producto subastado, la plataforma ofrece la posibilidad de visualizar información detallada que incluye el número de pujas, la cantidad de pujadores, las retractaciones, el tiempo restante en la subasta, y proporciona un historial completo de las pujas realizadas en ese producto. Este historial incluye datos relevantes sobre las pujas, como su valor y la fecha en la que se efectuaron. Además, se brinda acceso a información pertinente sobre los pujadores involucrados.
- eBay proporciona a los usuarios la capacidad de configurar pujas automáticas. En este proceso, el comprador define el precio máximo que está dispuesto a pagar por el producto, y la plataforma aumenta automáticamente la oferta en su nombre, siempre que sea necesario, para mantener al comprador como el principal postor hasta alcanzar el límite previamente establecido.
- Los usuarios tienen la posibilidad de examinar el perfil del vendedor, explorar otros productos que este tenga en venta y establecer contacto directo con él.
- Como vendedor, la plataforma te brinda la capacidad de definir el valor inicial de la puja, decidir si deseas recibir ofertas, establecer la fecha de inicio de la subasta, determinar su duración y habilitar la opción de compra directa a un precio específico de tu elección.

#### 3.1.2.1.2 *Desventajas de eBay*

A continuación, es posible señalar las siguientes desventajas:

- La interfaz de usuario muestra una gran cantidad de información, lo que puede resultar confuso para alguien que no está acostumbrado a ella.
- Cuando se va a realizar una puja, no aparece una ventana emergente de confirmación de puja o similar por lo que es fácil introducir una puja errónea y, posteriormente, puede resultar complicado retirarla.
- Las subastas tienen incrementos de puja predefinidos, lo que significa que no puedes especificar un valor concreto por el que desees pujar.

##### *3.1.2.1.3 Ventajas del nuevo sistema respecto eBay*

- El sistema en desarrollo presentará una interfaz simple e intuitiva, que proporcionará todos los datos esenciales para efectuar una puja con confianza, sin abrumar al usuario con una sobrecarga de información.
- El sistema que se desarrollará proporcionará al usuario información sobre las condiciones para retirar una puja antes de su confirmación.

#### 3.1.2.2. Catawiki

Catawiki es una plataforma de subastas online que permite a los usuarios comprar y vender productos de colección. La plataforma cuenta con varias secciones dedicadas a diferentes categorías de coleccionismo, tales como arte, relojes y cómics. Existe una sección específica dedicada al coleccionismo de cartas, donde los usuarios pueden pujar por cartas de interés, como las cartas de Pokémon.

La plataforma presenta una interfaz intuitiva y sencilla que permite a los usuarios pujar por cartas de manera rápida y eficaz. Según la propia empresa, posee más de 10 millones de compradores en todo el mundo, cuenta con aproximadamente 240 expertos encargados de la preparación de los objetos subastados, opera en 60 países y ofrece atención al cliente en 17 idiomas [8]. En base a esto, se puede afirmar que Catawiki es una plataforma consolidada y de confianza en el mercado de subastas *online*.

El uso de la página web es gratuito, aunque se cobra una comisión por la venta de los productos subastados. Los usuarios pueden establecer un precio mínimo de puja, denominado 'Precio de Reserva', que permanece oculto para los demás usuarios, permitiendo a los vendedores asegurarse de que no se venderá por debajo de un precio determinado. Si se realiza una puja en un lote con un precio de reserva, la plataforma informará al usuario que no se ha alcanzado dicho precio y ofrecerá la opción de aumentar su puja.

Los vendedores pueden consultar de forma transparente las comisiones que se aplicarán en caso de que se venda el producto. En general, la página web recibe buenas opiniones por parte de los usuarios, quienes destacan la seguridad que ofrece al realizar transacciones.

La aplicación cuenta con aproximadamente 240 expertos en las diferentes categorías de objetos subastados. Estos expertos revisan virtualmente los objetos enviados, ayudan a los vendedores con la presentación de los objetos y responden a las preguntas de los compradores.

Al acceder a un objeto subastado, se puede consultar quién lo vende y qué experto lo ha supervisado, además se pueden ver las pujas realizadas y el tiempo restante para que finalice la subasta. En algunos casos, en la información relativa al objeto aparece el valor que el experto considera que tiene el objeto, lo que puede ayudar a los compradores a decidir si pujar por el objeto.

Es posible ver comentarios acerca del vendedor y la reputación que tiene en la plataforma, lo que puede influir en la decisión de los compradores al pujar por un objeto.

La plataforma también ofrece la opción de pujar de forma automática, estableciendo un precio máximo, y se encargará de pujar por el usuario hasta alcanzar dicho precio. La interfaz informa sobre la reputación del vendedor, el número de pujas realizadas y la fecha de finalización de la subasta.

Las pujas son vinculantes, la plataforma informa de que si se gana una subasta, el usuario está obligado a comprar el objeto. Esto puede ser un aspecto a tener en cuenta, ya que si se gana una subasta, se deberá abonar el importe correspondiente al objeto subastado.

En resumen, ofrece bastante información sobre los objetos subastados y los vendedores, lo que puede ayudar a los compradores a tomar una decisión informada. Además, posee una interfaz sencilla y fácil de usar, lo que facilita la navegación por la plataforma.

En la Figura 3.9:Página de subastas de la sección de coleccionismo de cartas Pokémon de Catawiki se muestra la interfaz de usuario de la sección de coleccionismo de cartas de Pokémon Catawiki.

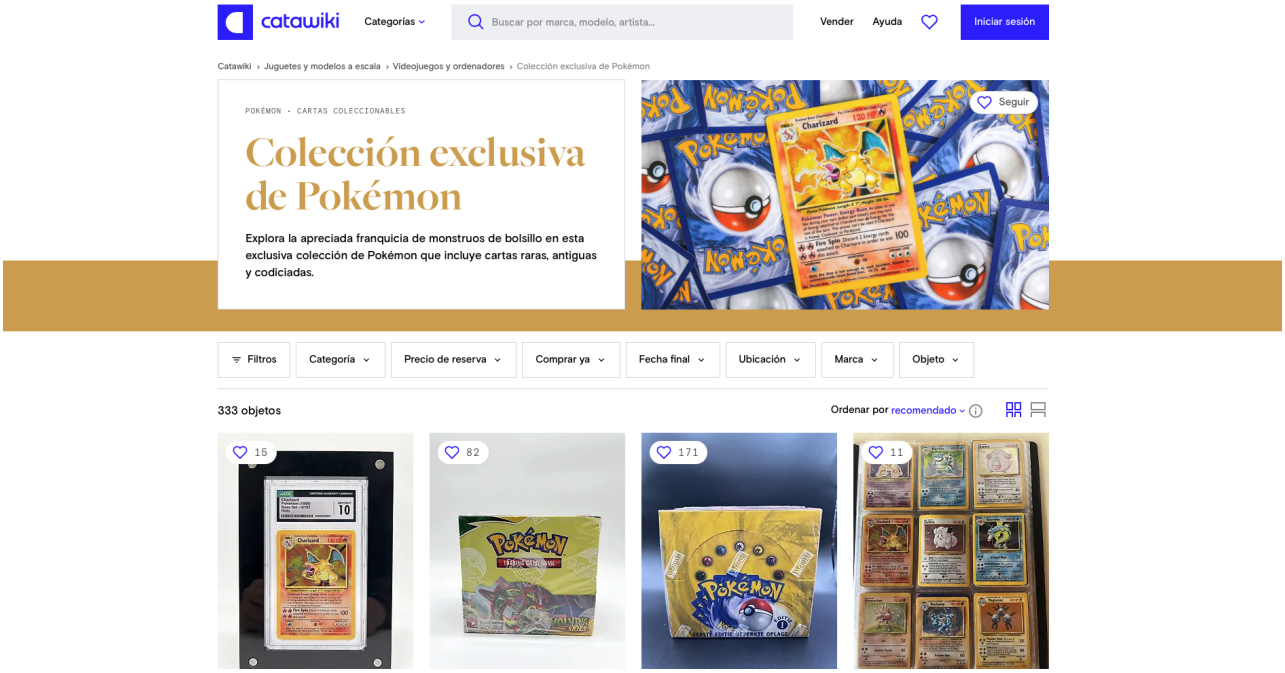


Figura 3.9: Página de subastas de la sección de coleccionismo de cartas Pokémon de Catawiki

En la Figura 3.10:Página de subasta de una carta de Pokémon en Catawiki se muestra la interfaz de usuario de la subasta de una carta de Pokémon en Catawiki.

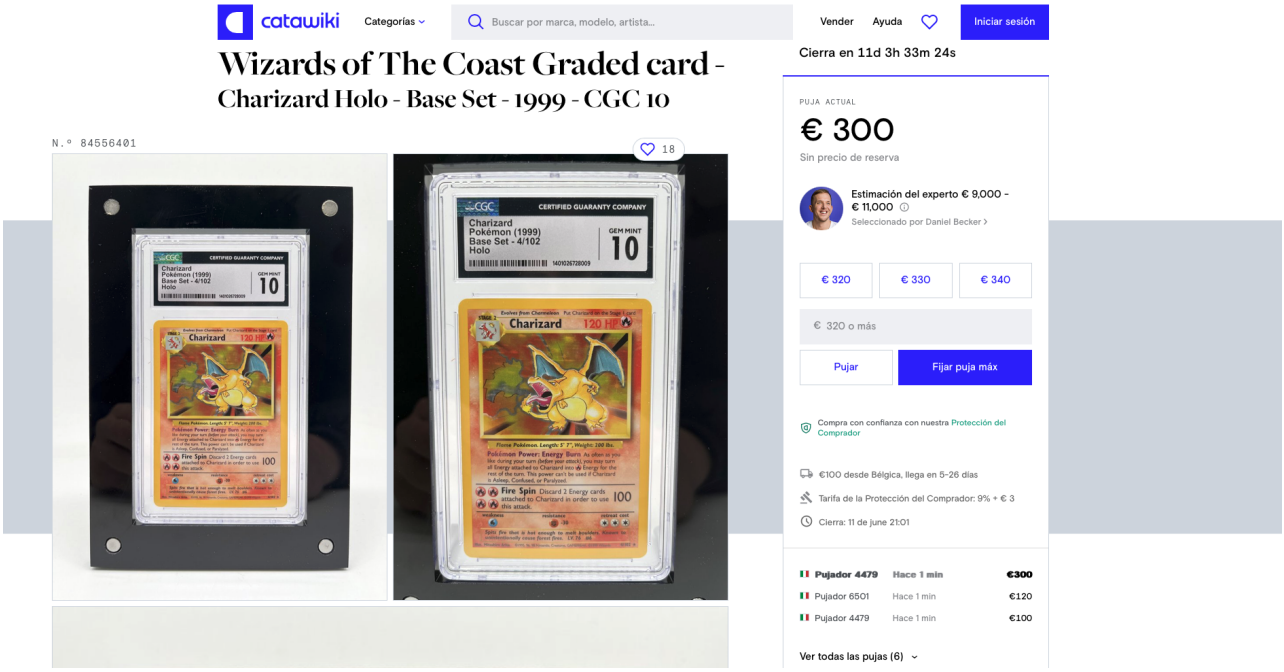


Figura 3.10: Página de subasta de una carta de Pokémon en Catawiki

#### 3.1.2.2.1 *Ventajas de Catawiki*

Dentro del marco de Catawiki, se destacan las siguientes características:

- La plataforma ofrece una amplia variedad de objetos de colección, lo que permite a los usuarios encontrar productos de su interés.
- Los usuarios pueden consultar la reputación del vendedor y la valoración junto con comentarios que ha recibido de otros compradores.
- La plataforma ofrece la posibilidad de pujar de forma automática, estableciendo un precio máximo, lo que facilita la participación en las subastas.
- Los expertos de la plataforma revisan los objetos subastados, lo que puede ayudar a los compradores a tomar una decisión informada y sentirse seguros al realizar una puja.

#### 3.1.2.2.2 *Desventajas de Catawiki*

A continuación, es posible señalar las siguientes desventajas:

- Las pujas son vinculantes, lo que significa que si se gana una subasta, el usuario está obligado a comprar el objeto. Una vez realizada una puja no se puede retirar ni modificar, lo que puede ser un problema si se ha pujado por error o si se ha cambiado de opinión.
- Algunos clientes se quejan de que los precios de envío, comisiones y otros gastos adicionales pueden ser elevados, lo que puede aumentar en exceso el precio final del objeto.

#### *3.1.2.2.3 Ventajas del nuevo sistema respecto Catawiki*

Algunos de los elementos que diferencian al sistema en desarrollo de Catawiki benefician a los usuarios y mejoran su experiencia en la plataforma son los siguientes:

- El sistema en desarrollo permitirá a los usuarios retirar una puja antes de que finalice la subasta, lo que puede ser útil si se ha pujado por error o si se ha cambiado de opinión.
- La plataforma que se desarrollará no cobrará comisiones por la venta de los productos subastados, lo que puede resultar atractivo para los usuarios.

### 3.1.3. El mercado de coleccionistas de Pokémon

El mercado de coleccionistas de Pokémon es un sector en constante crecimiento y expansión. La popularidad de las cartas de Pokémon ha aumentado significativamente en los últimos años, lo que ha llevado a un incremento en la demanda de cartas de colección y a un aumento en los precios de las cartas más raras y valiosas. Las plataformas de subastas en línea han contribuido notablemente a esta expansión, permitiendo a los coleccionistas comprar y vender cartas de Pokémon de forma segura y eficiente.

Estas plataformas, a menudo, emplean intermediarios encargados de verificar la autenticidad de las cartas y de garantizar la seguridad de las transacciones, brindando a los coleccionistas mayor confianza al realizar compras y ventas.

Existen varias plataformas web que ofrecen análisis detallados de los precios de las cartas de Pokémon, como PokemonPrice y PriceCharting. Estas plataformas permiten a los coleccionistas consultar el valor de mercado de las cartas de Pokémon, así como el precio de venta de las cartas más populares y raras. Además, proporcionan historiales de precios y gráficos que muestran la evolución del precio de las cartas a lo largo del tiempo. Los datos se actualizan regularmente, recopilando información de plataformas de ventas como eBay.

En la Figura 3.11:Página de cartas de Pokémon de PokemonPrice se muestra el análisis de precios de las cartas de Pokémon de PokemonPrice.

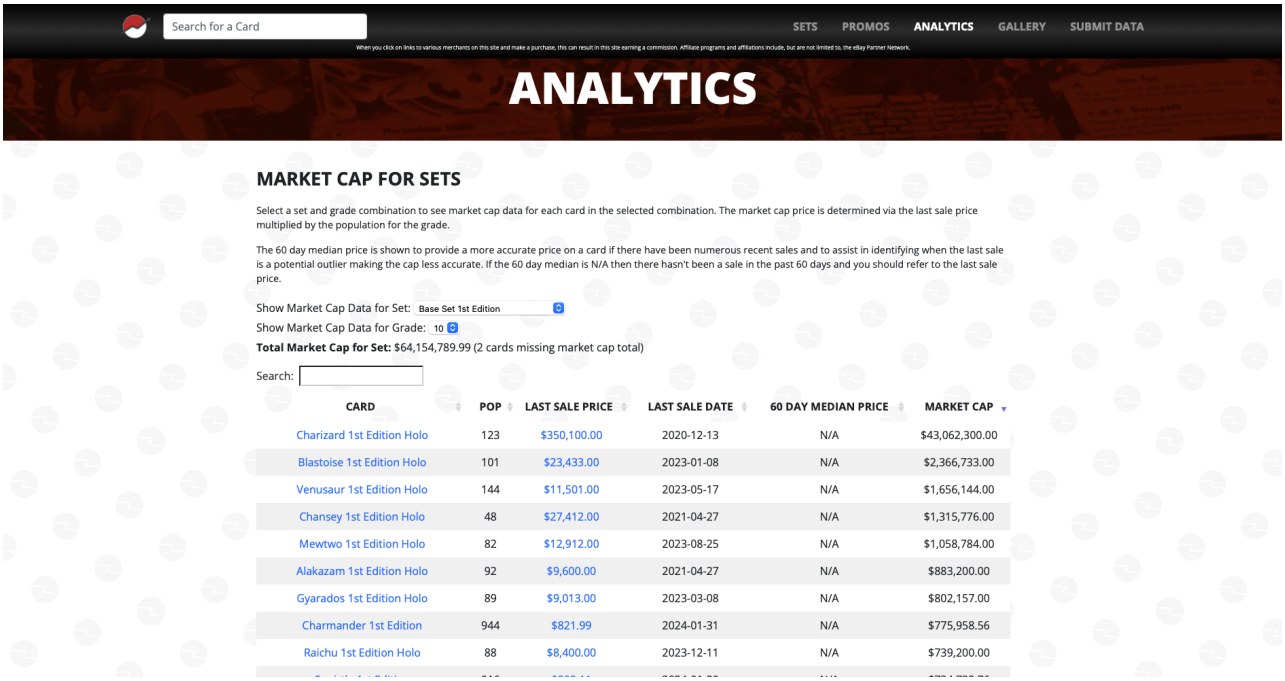


Figura 3.11: Página de cartas de Pokémon de PokemonPrice

En esta plataforma, los usuarios pueden ver el precio de mercado de una carta concreta y las transacciones realizadas, como se muestra en la Figura 3.12:Página de una carta de Pokémon de PokemonPrice.

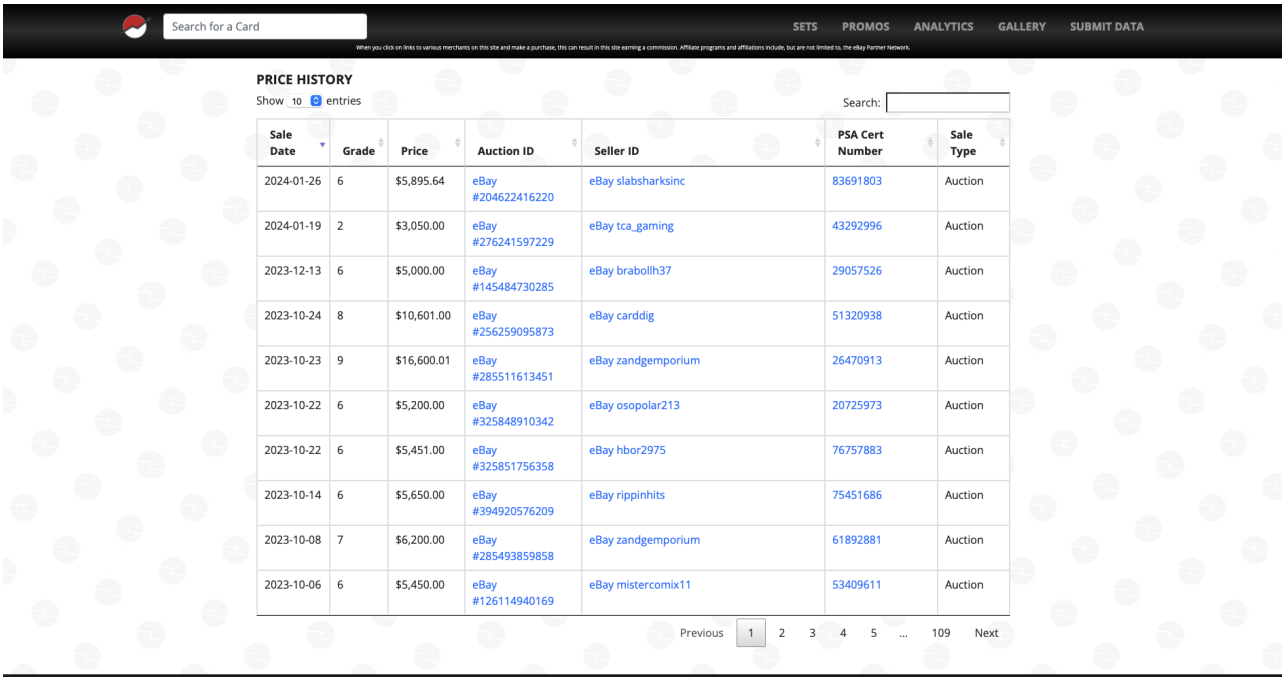


Figura 3.12: Página de una carta de Pokémon de PokemonPrice

En la Figura 3.13:Página de cartas de Pokémon de PriceCharting se muestra la interfaz de usuario de consulta

de datos para un set de cartas de Pokémon de la plataforma PriceCharting.

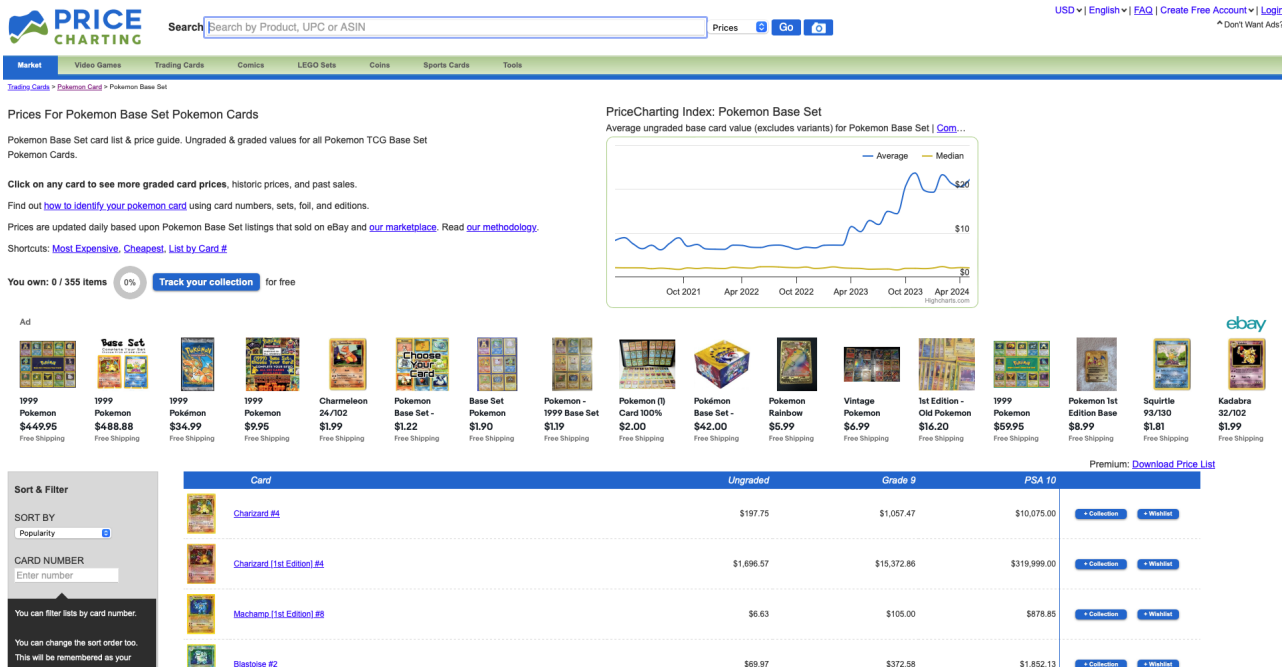


Figura 3.13: Página de cartas de Pokémon de PriceCharting

PriceCharting también proporciona información detallada sobre cada transacción y su precio, como se muestra en la Figura 3.14:Página de una carta de Pokémon de PriceCharting.

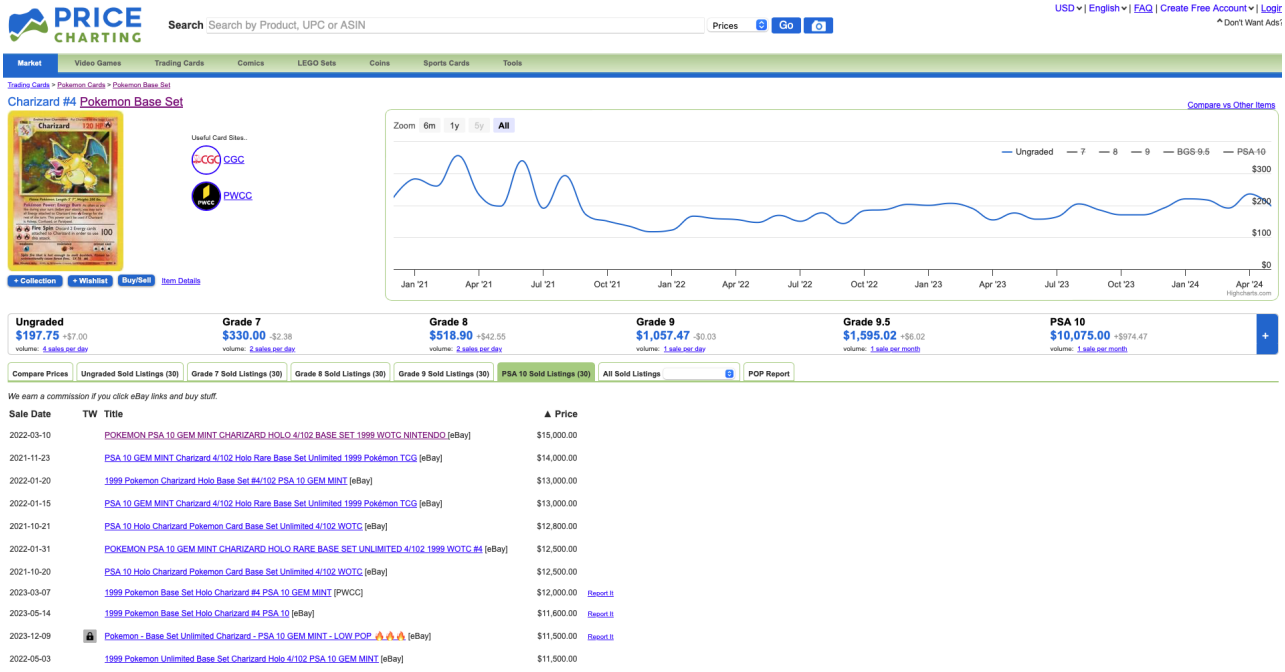


Figura 3.14: Página de una carta de Pokémon de PriceCharting

El análisis de los datos recopilados en estas plataformas revela que el precio de las cartas de Pokémon varía en función de su rareza, estado de conservación y popularidad. Es evidente que el mercado de coleccionistas de Pokémon es un sector en constante crecimiento y expansión, generando una gran cantidad de ingresos, lo que lo convierte en un mercado altamente atractivo para su explotación.

## 3.2. VALORACIÓN DE ALTERNATIVAS DE SOLUCIÓN

En esta sección, se realiza una evaluación detallada de las alternativas tecnológicas y arquitectónicas disponibles que satisfacen los requisitos establecidos para el proyecto. Se seleccionarán la arquitectura tecnológica, las tecnologías utilizadas para el desarrollo del backend y el frontend, la base de datos y el proveedor de servicios en la nube más adecuados para facilitar el desarrollo exitoso de la aplicación.

El análisis involucra una evaluación rigurosa de las ventajas y desventajas asociadas a cada opción. El objetivo principal es identificar la solución más apropiada que no solo cumpla con los requisitos funcionales y no funcionales del proyecto, sino que también se alinee de manera óptima con las restricciones y objetivos estratégicos.

Este proceso es muy importante para asegurar que la tecnología seleccionada respalde efectivamente las necesidades del proyecto, maximizando la eficiencia operativa y permitiendo una escalabilidad futura.

### 3.2.1. Valoración de alternativas para la arquitectura

A continuación, se presenta una comparación de varias alternativas arquitectónicas para el desarrollo del sistema.

#### 3.2.1.1. Arquitectura Monolítica

La arquitectura monolítica es un enfoque de desarrollo de software en el que una aplicación se construye como una sola unidad. En este caso, todos los componentes del sistema se diseñan y se implementan como un único bloque, que se ejecuta como un único proceso. Este enfoque en un proyecto pequeño puede ser beneficioso, ya que simplifica el proceso de desarrollo y sobretodo de despliegue. Sin embargo, a medida que el proyecto crece, la arquitectura monolítica se vuelve cada vez más compleja y difícil de mantener. Además, la escalabilidad de la aplicación se ve limitada por la necesidad de escalar el sistema en su conjunto, en lugar de poder escalar componentes individuales de forma independiente.

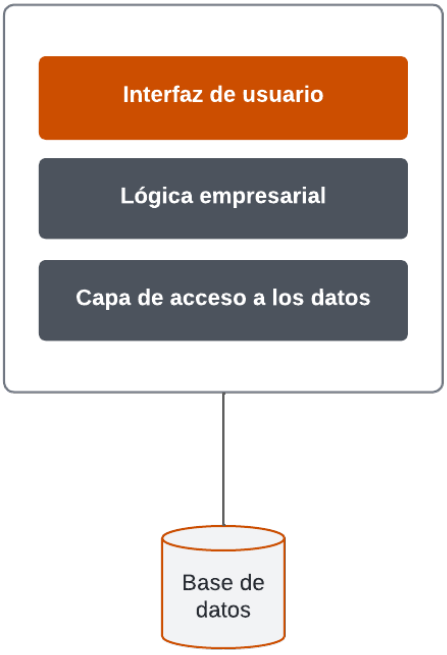


Figura 3.15: Arquitectura Monolítica

#### 3.2.1.2. Arquitectura de Microservicios

La arquitectura de microservicios es un enfoque de desarrollo de software en el que una aplicación se construye como un conjunto de servicios pequeños, independientes y altamente escalables. Cada servicio se ejecuta como un proceso separado y se comunica con otros servicios mediante mecanismos ligeros, como una API REST. Este enfoque permite que los servicios se desarrollen, despliegen y escalen de forma independiente, lo que facilita la gestión de proyectos complejos. Sin embargo, la arquitectura de microservicios también introduce una mayor complejidad en el desarrollo y la gestión de la aplicación, ya que requiere la implementación de mecanismos de comunicación entre los servicios, así como la gestión de la escalabilidad de cada uno de ellos.

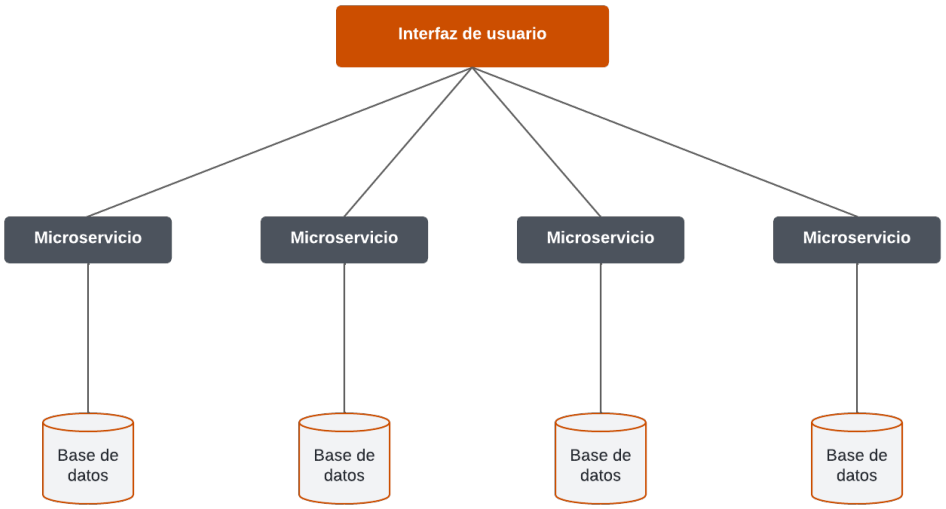


Figura 3.16: Arquitectura de Microservicios

#### 3.2.1.3. Arquitectura de REST API y WepApp

Se caracteriza por una clara división entre cliente y servidor, encapsulados respectivamente en WepApp (frontend) y REST API (backend). Esta separación promueve una organización modular, facilitando el mantenimiento del proyecto al separar de forma clara las distintas responsabilidades. Este enfoque es un punto intermedio entre las dos arquitecturas anteriores, ya que permite una mayor flexibilidad en el desarrollo y la gestión de la aplicación, sin introducir una complejidad excesiva.

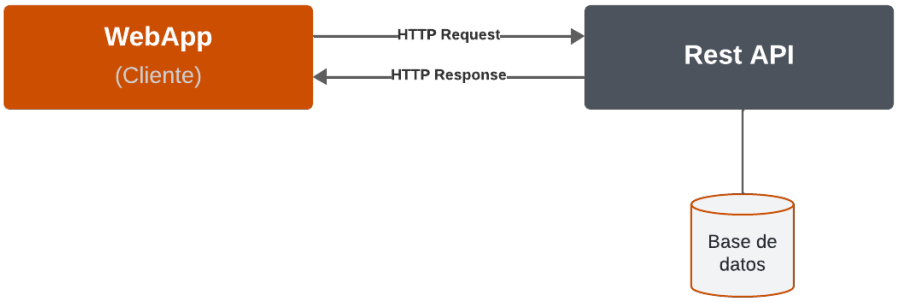


Figura 3.17: Arquitectura de REST API y WebApp

#### 3.2.1.4. Comparativa de alternativas arquitectónicas

Mediante la siguiente tabla, se presenta una comparación de las alternativas arquitectónicas previamente mencionadas.

Tabla 3.1: Comparación de Arquitecturas de Software

| Criterio | Arquitectura Monolítica | Arquitectura de Microservicios | API REST y WebApp |
|---|---|---|---|
| Complejidad Inicial | Baja | Alta | Moderada |
| Escalabilidad | Limitada | Alta | Moderada |
| Facilidad de Mantenimiento | Alta en proyectos pequeños | Moderada a baja | Moderada si cada módulo tiene una estructura interna clara |
| Despliegue | Sencillo | Complejo | Moderado |
| Gestión de Proyectos | Simple en proyectos pequeños | Requiere coordinación compleja | Equilibrada |
| Independencia de Componentes | No | Sí | Parcial |
| Adaptabilidad a Cambios | Baja | Alta | Moderada |
| Recomendado para | Proyectos pequeños y simples | Proyectos grandes y escalables | Proyectos con necesidad de separación clara entre frontend y backend |

#### 3.2.1.5. Decisión final de la arquitectura

Tras un análisis exhaustivo de las alternativas disponibles, se ha optado por implementar una arquitectura de API REST y WebApp.

Esta decisión permite alcanzar un equilibrio entre la simplicidad inherente a la arquitectura monolítica y la escalabilidad ofrecida por la arquitectura de microservicios. Además, esta elección facilita la gestión del proyecto mediante una separación clara entre el frontend y el backend. Tal distinción posibilita el desarrollo y despliegue independiente de cada componente, contribuyendo a la minimización de riesgos asociados a fallos en cadena. Además, cada módulo, operando de manera aislada, mejora significativamente la disponibilidad y confiabilidad del sistema.

### 3.2.2. Valoración de alternativas para el Backend

Es crucial seleccionar una tecnología de backend que no solo soporte eficientemente las operaciones en tiempo real sino que también se integre de manera óptima con el frontend. A continuación, se presenta un análisis de varias alternativas para el desarrollo del backend teniendo en cuenta estos requisitos.

#### 3.2.2.1. Java con Spring Boot

Java es un lenguaje de programación orientado a objetos que destaca por su portabilidad y robustez, siendo ampliamente utilizado en el desarrollo de aplicaciones empresariales. La combinación con Spring Boot permite un desarrollo ágil con una amplia gama de herramientas como Spring Data y Spring Security entre otras. Entre sus ventajas, además de las herramientas mencionadas, destacan su escalabilidad, su robustez en el manejo de transacciones y su amplia comunidad. Sin embargo, Java con Spring Boot presenta una curva de aprendizaje significativa y, aunque Spring Boot puede manejar aplicaciones en tiempo real mediante Spring WebFlux, su rendimiento en este ámbito puede ser inferior comparado con otras tecnologías. Además, la integración con el frontend, podría no ser la más rápida en términos de desarrollo.

##### *3.2.2.1.1 Node.js con Express*

Node.js es un entorno de ejecución para JavaScript en el lado del servidor, conocido por su modelo de I/O no bloqueante y su eficiencia en el manejo de múltiples conexiones simultáneas La integración de Node.js con Express, un framework ligero y flexible, ofrece una solución óptima para desarrollar aplicaciones web dinámicas. Esta combinación destaca por su capacidad para gestionar operaciones en tiempo real de manera eficiente y su sinergia natural con tecnologías frontend basadas en JavaScript, facilitando un desarrollo cohesivo y ágil entre el backend y el frontend. Como desventajas, cabe mencionar que Node.js puede no ser la opción más adecuada para tareas que requieren un alto uso de CPU debido a su naturaleza de ejecución de un solo hilo, ya que su rendimiento en este ámbito puede ser inferior al de otras tecnologías.

#### 3.2.2.2. Python con Django

Python, un lenguaje de programación de alto nivel, se caracteriza por su versatilidad y amplia gama de bibliotecas. Utilizado ampliamente en aplicaciones empresariales combinado con Django, un framework de alto nivel que se enfoca en el desarrollo rápido y eficiente. Como ventajas, destacan su desarrollo rápido, sus excelentes capacidades de seguridad y su buena documentación. Aunque Django puede ser configurado para soportar aplicaciones en tiempo real mediante Django Channels, su rendimiento en estos escenarios puede no ser tan óptimo como el de Node.js. Además, aunque Django asegura una buena integración con el frontend, puede no ser la opción más ágil para aplicaciones que requieren una interacción constante.

#### 3.2.2.3. Comparativa de alternativas para el Backend

En la siguiente tabla, se presenta una comparación de las alternativas para el desarrollo del Backend.

| Criterio | Java con Spring Boot | Node.js con Express | Python con Django |
|---|---|---|---|
| **Modelo de Programación** | Orientado a objetos, con énfasis en inyección de dependencias. | Basado en eventos y callbacks. | Orientado a componentes/-modelos con énfasis en la reutilización de código. |
| **Funcionalidad en tiempo real** | Buen manejo de transacciones pero menos óptimo para tiempo real. | Excelente para operaciones en tiempo real gracias a su eficiencia en I/O. | Posible pero requiere más configuración para tiempo real. |
| **Integración con Frontend** | Buena, puede requerir esfuerzos adicionales. | Natural y fluida. | Buena, pero puede necesitar configuraciones extra. |
| **Escalabilidad** | Alta, pero con escalabilidad vertical. | Alta, con facilidad para escalar horizontalmente. | Moderada, con algunas limitaciones en escalabilidad. |
| **Seguridad** | Fuertes capacidades de seguridad. | Requiere implementaciones adicionales para seguridad. | Seguridad integrada y robusta. |

Tabla 3.2: Comparación de Tecnologías para el Backend

#### 3.2.2.4. Decisión final del Backend

Considerando las necesidades del sistema a desarrollar, para soportar subastas en tiempo real y una integración fluida con el frontend la opción más adecuada es Node.js con Express.

### 3.2.3. Valoración de alternativas para el Frontend

En esta sección, se realiza una evaluación detallada de las tecnologías *frontend* que se integran de manera óptima con el entorno de ejecución Node.js y el framework Express. Dado que ambos componentes del backend utilizan JavaScript como su lenguaje de programación principal, se seleccionarán aquellas tecnologías *frontend* que no solo mantengan coherencia con este lenguaje, sino que también optimicen la interactividad, la respuesta y la eficiencia de la aplicación.

#### 3.2.3.1. React

React es una biblioteca de JavaScript desarrollada y mantenida por Facebook, centrada en la construcción de interfaces de usuario a través de componentes reutilizables. Se caracteriza por su virtual DOM y su enfoque declarativo.

Entre sus ventajas, destacan su amplia comunidad, su rendimiento optimizado con Virtual DOM y su flexibilidad en la elección de estilos y componentes. Como desventajas cabe mencionar las actualizaciones frecuentes que implican mantenerse al día con los cambios y que necesita integración con otras herramientas para ser una solución completa.

#### 3.2.3.2. Angular

Angular es un framework de desarrollo web mantenido por Google, conocido por su enfoque en aplicaciones de página única (SPA). Utiliza TypeScript como lenguaje principal y proporciona un entorno robusto y completo para el desarrollo.

Entre sus ventajas, destaca su ecosistema completo, promueve un estilo de desarrollo coherente y mantenible y, además, cuenta con una comunidad activa y un amplio soporte de Google, lo que garantiza actualizaciones y soporte continuo. Sin embargo, Angular puede ser excesivo para proyectos pequeños, ya que su curva de aprendizaje es pronunciada y su configuración inicial puede ser compleja.

#### 3.2.3.3. Vue.js

Vue.js es un framework progresivo para la construcción de interfaces de usuario, creado por Evan You. Se destaca por su facilidad de adopción, su sistema reactividad y su enfoque en la simplicidad.

Entre sus ventajas, destacan su sintaxis sencilla, su buena documentación y su facilidad de integración en proyectos existentes. Aunque su uso va en aumento, su comunidad es más pequeña en comparación con React o Angular, lo que se traduce en menos librerías y recursos disponibles.

#### 3.2.3.4. Comparativa de alternativas para el Frontend

A continuación, se presenta una comparación de las alternativas para el desarrollo del Frontend.

| Criterio | React | Angular | Vue.js |
|---|---|---|---|
| **Velocidad y Rendimiento** | Alto con Virtual DOM. Optimizado para cambios dinámicos de UI. | Buen rendimiento, pero puede ser más lento en proyectos grandes debido a su complejidad. | Rendimiento similar a React, con optimizaciones en la actualización de la UI. |
| **Mantener el Estado** | Requiere bibliotecas adicionales como Redux para manejo complejo del estado. | Gestión de estado integrada, adecuada para aplicaciones complejas. | Sistema reactividad sencillo, para manejo de estado complejo requiere la biblioteca Vuex. |
| **Comunidad** | Muy grande y activa, con un ecosistema extenso. | Amplia y soportada por Google, con muchas empresas adoptándolo. | Creciente y entusiasta, con un enfoque en la facilidad de uso. |
| **Curva de Aprendizaje** | Moderada; JSX y el ecosistema pueden requerir tiempo de aprendizaje. | Elevada; TypeScript y su arquitectura completa requieren más tiempo para aprender. | Baja; Vue es considerado fácil de aprender, especialmente para principiantes. |
| **Escalabilidad** | Muy escalable con un enfoque modular y reutilizable. | Diseñado para aplicaciones empresariales escalables y complejas. | Escalable, pero más adecuado para proyectos de tamaño mediano. |
| **Ecosistemas y Módulos** | Rico ecosistema con una gran cantidad de módulos y herramientas. | Ecosistema completo con soluciones integradas para muchas necesidades. | Ecosistema más pequeño aunque en crecimiento, con módulos y librerías en aumento. |

Tabla 3.3: Análisis Comparativo de Frameworks y Bibliotecas de Frontend

#### 3.2.3.5. Decisión final del Frontend

Tras una evaluación detallada de las distintas opciones disponibles y en función de los requisitos específicos del sistema a desarrollar, se ha determinado que React es la tecnología más adecuada para el frontend.

Esta decisión se basa, principalmente, en la gran comunidad de React y en la rica colección de recursos disponles.

### 3.2.4. Valoración de alternativas para la Base de Datos

Por último, se presenta un análisis de alternativas para la base de datos del sistema.

#### 3.2.4.1. MySQL

MySQL es un sistema de gestión de bases de datos relacional de código abierto, ampliamente utilizado en aplicaciones web.

Entre sus ventajas, destacan su amplia adopción lo que conlleva una gran cantidad de recursos y documentación disponibles, su fiabilidad y robustez y su facilidad de uso.

Por otro lado, MySQL puede no ser la opción más adecuada para aplicaciones que requieren de operaciones avanzadas de análisis y procesamiento de datos, ya que su rendimiento en este ámbito puede ser inferior al de otras bases de datos.

#### 3.2.4.2. MongoDB

MongoDB es una base de datos NoSQL orientada a documentos, diseñada para la escalabilidad y la flexibilidad. Se caracteriza por su capacidad para manejar grandes volúmenes de datos y su esquema flexible.

Entre sus ventajas, destacan su flexibilidad, permite que la base de datos crezca con la aplicación añadiendo nuevos campos a los documentos. Además, MongoDB es altamente escalable y puede manejar grandes volúmenes de datos de manera eficiente.

Sin embargo, MongoDB puede no ser la opción más adecuada para aplicaciones que requieren transacciones ACID complejas, ya que su modelo de consistencia eventual puede no ser adecuado para todas las aplicaciones.

#### 3.2.4.3. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos relacional de código abierto y gratuito, conocido por su robustez, escalabilidad y soporte para transacciones ACID.

Entre sus ventajas, destacan su cumplimiento con los estándares SQL, su soporte para transacciones ACID y su capacidad para manejar grandes volúmenes de datos asegurando la integridad y seguridad de los mismos.

Como inconvenientes, cabe mencionar que PostgreSQL puede ser más lento que otras bases de datos en operaciones de lectura y escritura para bases de datos pequeñas ya que está enfocada en manejar un gran volumen de datos, y su curva de aprendizaje puede ser pronunciada para usuarios no familiarizados con SQL.

#### 3.2.4.4. Comparativa de alternativas para la Base de Datos

En la siguiente tabla, se presenta una comparación de las alternativas para la base de datos del sistema.

| Criterio | MySQL | MongoDB | PostgreSQL |
|---|---|---|---|
| **Tipo de Base de Datos** | Relacional | NoSQL | Relacional |
| **Modelo de Datos** | Tablas y Filas | Documentos JSON/BSON | Tablas y Filas |
| **Lenguaje de Consulta** | SQL | MongoDB Query Language (MQL) | SQL |
| **Escalabilidad** | Escalabilidad vertical y soporte para horizontal | Escalabilidad horizontal mediante sharding | Escalabilidad vertical y horizontal |
| **Transacciones** | Soporte completo de ACID | Soporte parcial de ACID en ciertas operaciones | Soporte completo de ACID |
| **JSON** | Soporte limitado mediante campos JSON | JSON nativo | Soporte avanzado de JSON y JSONB |
| **Flexibilidad de Esquema** | Esquemas rígidos | Esquemas flexibles | Esquemas rígidos |
| **Rendimiento** | Alto rendimiento en lecturas | Alto rendimiento en escrituras | Alto rendimiento en lecturas y escrituras |

Tabla 3.4: Comparativa entre MySQL, MongoDB y PostgreSQL

#### 3.2.4.5. Decisión final de la Base de Datos

Tras un análisis detallado de las distintas opciones disponibles y en función de los requisitos específicos del sistema a desarrollar, se ha determinado que MongoDB es la tecnología más adecuada para la base de datos.

Esta elección se basa en un *trade-off*, es decir, se acepta la falta de integridad referencial y la consistencia eventual a cambio de una mayor flexibilidad y escalabilidad en el manejo de datos que es lo que se necesita para el sistema a desarrollar.

### 3.2.5. Valoración de alternativas de proveedor de Servicios Cloud

Para el despliegue de la aplicación, es necesario seleccionar un proveedor de servicios en la nube que ofrezca una infraestructura escalable, segura y fiable. A continuación, se presentan las alternativas más populares y se realiza una comparación entre ellas.

#### 3.2.5.1. Amazon Web Services (AWS)

AWS, proveedor de servicios en la nube de Amazon, es líder en el mercado de servicios en la nube, ofreciendo una amplia gama de servicios que incluyen computación, almacenamiento y bases de datos.

AWS es conocido por su capacidad de escalabilidad y tiene una gran red de centros de datos alrededor del mundo, lo que asegura una alta disponibilidad y baja latencia. Además, tiene características avanzadas de seguridad. Todo esto hace que sea una de las opciones preferidas para empresas y organizaciones de gran tamaño.

#### 3.2.5.2. Google Cloud Platform (GCP)

Google Cloud Platform es el proveedor de servicios en la nube de Google, que ofrece una amplia gama de servicios de infraestructura y plataformas de desarrollo.

Proporciona inteligencia artificial y aprendizaje automático que pueden agilizar el desarrollo de aplicaciones y mejorar la experiencia del usuario. Se integra fácilmente con otros servicios de Google, como Google Workspace y Google Analytics, proporcionando un ecosistema cohesionado para empresas que utilizan servicios de Google.

#### 3.2.5.3. Microsoft Azure

Microsoft Azure proveedor de servicios en la nube de Microsoft, esel segundo proveedor de servicios en la nube más grande del mundo y ofrece una amplia gama de servicios de infraestructura y plataformas de desarrollo.

Azure es conocido por su integración con herramientas y servicios de Microsoft, como Office 365 y Dynamics 365, lo que facilita la adopción por parte de empresas que ya utilizan productos de Microsoft.

#### 3.2.5.4. Comparativa de Proveedores de Servicios Cloud

Los principales proveedores de servicios en la nube han sido evaluados según su capacidad para cumplir con los requisitos esenciales de escalabilidad, seguridad y fiabilidad. Aunque cada uno de estos proveedores satisface adecuadamente estas expectativas fundamentales, la elección entre ellos debe tener en cuenta factores diferenciadores adicionales tales como el costo, la facilidad de uso y la integración con herramientas y servicios complementarios.

Todos operan bajo un modelo de pago por uso y ofrecen una gama de servicios similares. Cada uno requiere un determinado nivel de competencia técnica para su configuración y gestión eficaz. En este contexto, la decisión final se basará principalmente en las condiciones de licencia accesibles para el equipo de desarrollo, adaptándose a las posibilidades económicas del proyecto.

#### 3.2.5.5. Decisión final del Proveedor de Servicios Cloud

Tras un análisis de las distintas licencias para estudiantes, se ha seleccionado Microsoft Azure como el proveedor de servicios en la nube para el despliegue de nuestra aplicación.

Esta decisión está respaldada por la disponibilidad de una licencia académica proporcionada por la universidad, que ofrece significativas ventajas económicas. Además, la familiaridad del equipo con el ecosistema de

Microsoft asegura una integración fluida y una curva de aprendizaje reducida, facilitando así el despliegue y la gestión de la aplicación en la nube.

## Capítulo 4

# DEFINICIÓN DE LA ARQUITECTURA TECNOLÓGICA

## 4.1. SELECCIÓN DE LA ARQUITECTURA TECNOLÓGICA

En esta sección se presentan las decisiones tomadas en cuanto a la arquitectura tecnológica del proyecto. En la sección 3.2: VALORACIÓN DE ALTERNATIVAS DE SOLUCIÓN se puede consultar el análisis de las alternativas valoradas.

El proyecto está diseñado para tener una clara distinción entre el *frontend* y el *backend*, lo cual facilita la escalabilidad y el mantenimiento del sistema. Se ha optado por implementar la arquitectura de tipo REST API y Web App. El módulo REST API, es decir, el *backend* será responsable de la lógica de negocio, mientras que el módulo Web App, *frontend*, gestionará la presentación de la información.

Esta estructura permite que el desarrollo del *frontend* se realice de manera independiente al *backend*, permitiendo modificaciones en uno sin afectar al otro. La comunicación entre el *frontend* y el *backend* se llevará a cabo mediante peticiones HTTP, como se ilustra en la Figura 4.1: Arquitectura REST API y Web App.

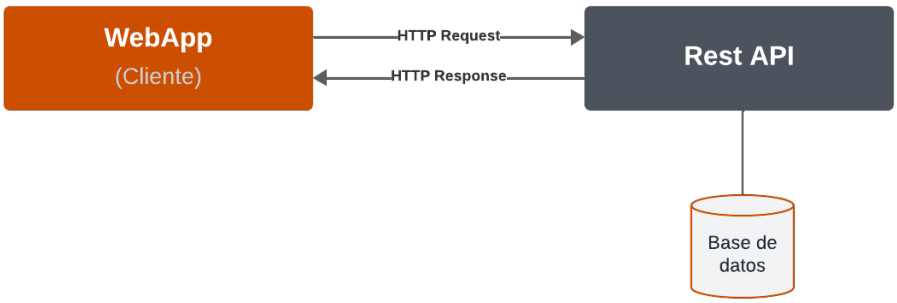


Figura 4.1: Arquitectura REST API y Web App

Se han seleccionado las siguientes tecnologías como alternativas de solución: Node.js con Express para el backend, React.js para el frontend, y MongoDB Atlas como base de datos.

Estas decisiones conforman una arquitectura **MERN** (MongoDB, Express, React, Node.js), ejemplificada en la Figura 4.2: MERN Stack: MongoDB, Express, React, Node.js.

Esta arquitectura presenta diversas ventajas, tales como la familiaridad de los desarrolladores con las tecnologías involucradas, el amplio soporte y documentación disponibles debido a su uso común en el desarrollo de aplicaciones web actuales, y la facilidad de integración entre las tecnologías.

Si bien esta arquitectura se fundamenta en el lenguaje de programación JavaScript, se ha decidido adoptar TypeScript. Este lenguaje es una extensión de JavaScript que introduce tipado estático al lenguaje, lo que no solo facilita la detección de errores durante la compilación, sino que también contribuye significativamente a la mejora de la calidad del código y a la comprensión de este.

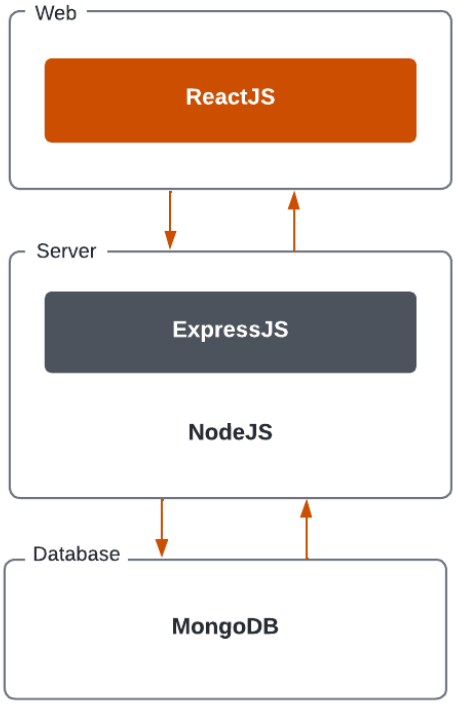


Figura 4.2: MERN Stack: MongoDB, Express, React, Node.js

Para el despliegue de la aplicación se ha optado por utilizar la plataforma de servicios en la nube de Microsoft, Azure. Esta elección se complementa con el uso de GitHub Actions para la implementación de procesos de integración continua, permitiendo así un desarrollo ágil y sistemático. Como se ha comentado anteriormente, se incorporará MongoDB Atlas para gestionar la base de datos en la nube, aprovechando su capacidad de escalabilidad y flexibilidad. Por último, se utilizará Docker para la contenerización de la aplicación, facilitando la portabilidad y la consistencia del entorno de ejecución a través de diferentes infraestructuras de despliegue. Este enfoque integrado asegura una arquitectura robusta y eficiente para el despliegue de la aplicación en un entorno de nube.

## Capítulo 5

# PLANIFICACIÓN Y GESTIÓN DEL TFG

## 5.1. PLANIFICACIÓN DEL PROYECTO

En este apartado se abordará la planificación inicial del proyecto, detallando los aspectos más relevantes del mismo. Se incluirá la identificación de interesados, las tareas a realizar, la estructura de desglose de trabajo, la planificación temporal, los riesgos identificados y el presupuesto inicial.

### 5.1.1. Identificación de Interesados

Los interesados en el proyecto son las personas o grupos que pueden afectar o verse afectados por el proyecto. En el caso de este proyecto, los interesados son los siguientes:

- **Usuarios:** Compradores de sobres, vendedores en subastas, pujadores y coleccionistas. Su satisfacción y experiencia de usuario son críticas para el éxito de la plataforma.
- **Desarrolladores:** Equipo encargado del desarrollo, mantenimiento y actualización de la plataforma.
- **Propietarios/Creadores:** Fundadores y administradores del proyecto, responsables de la visión, dirección y toma de decisiones estratégicas.
- **Inversores:** Personas o entidades que financian el proyecto, con expectativas de retorno sobre su inversión. Su interés reside en la rentabilidad y sostenibilidad del negocio.
- **Proveedores:** Proveedores de servicios tecnológicos y de infraestructura, como *hosting*, sistemas de pago y seguridad.
- **Comunidades:** Grupos y foros de fans y coleccionistas de Pokémon que pueden influir en la popularidad y adopción de la plataforma. Su participación y *feedback* pueden guiar mejoras y nuevas características.
- **Reguladores:** Entidades gubernamentales y de la industria que aseguran el cumplimiento de leyes y regulaciones aplicables al comercio digital y la protección de datos.

### 5.1.2. OBS y PBS

El OBS, *Organizational Breakdown Structure* es una estructura que representa las responsabilidades sobre la realización de las tareas del proyecto. Por otro lado, el PBS *(Product Breakdown Structure)* es una estructura jerárquica que descompone el proyecto en los productos que se deben entregar para cumplir con los objetivos del proyecto.

#### 5.1.2.1. OBS

En el contexto del desarrollo del proyecto, se ha procedido a la identificación y análisis de los perfiles profesionales implicados, así como de las respectivas responsabilidades y tareas asignadas a cada uno. Aunque la ejecución del proyecto se realizará de forma individual, esta estructuración permite una asignación y gestión presupuestaria más precisa, alineando los recursos financieros con las funciones específicas requeridas para cada etapa del proyecto. Los roles se han representado por medio de la estructura jerárquica OBS (*Organizational Breakdown Structure*), mostrada en la Figura: 5.1 OBS del proyecto, faiclitando la visualización de la estructura organizativa del proyecto.

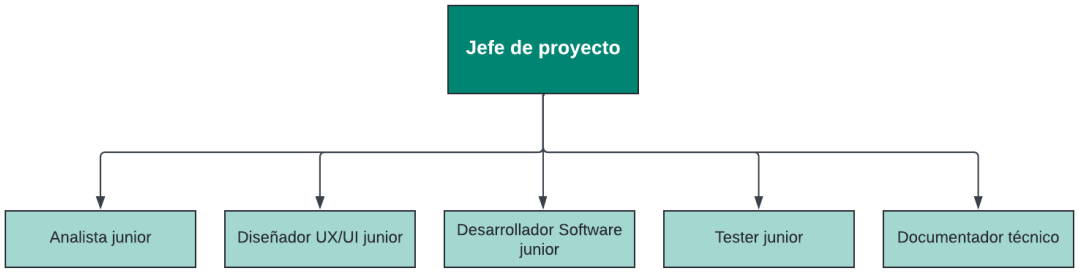


Figura 5.1: OBS del proyecto

Se ha decidido realizar una matriz de asignación de responsabilidades RACI para relacionar las tareas identifcadas en 5.1.3.1 WBS con cada rol. En esta matriz, a cada tarea se le asigna un rol con una de las siguientes responsabilidades:

- **R:** Responsable. Persona que realiza la tarea.
- **A:** Aprobador. Persona que aprueba la tarea.
- **C:** Consultado. Persona a la que se consulta sobre la tarea.
- **I:** Informado. Persona a la que se informa sobre el avance y los resultados de la tarea.

Un mismo recurso puede tener varias responsabilidades en una misma tarea, en tal caso se anotarán separadas por el caracter "/". Los recursos que no tienen ninguna responsabilidad en una tarea se dejan en blanco. En la tabla Tabla 5.1: Roles y abreviaturas de los recursos, se muestran los roles y las abreviaturas utilizadas en la matriz RACI.

Tabla 5.1: Roles y abreviaturas de los recursos

| Abreviatura | Rol |
|---|---|
| JP | Jefe de Proyecto |
| AN | Analista junior |
| DI | Diseñador junior |
| DS | Desarrollador de Software junior |
| TE | Tester junior |
| DOC | Documentador técnico |

#### *5.1.2.1.1 Análisis del proyecto*

En la Tabla: 5.2 Tabla RACI de la fase Análisis del proyecto se muestra la matriz de responsabilidades para la fase de análisis.

Tabla 5.2: Tabla RACI de la fase Análisis del proyecto

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| Análisis del sistema | **A** | **R** | | **C** | | |
| Análisis de la arquitectura | **A** | **R** | | **C** | | |
| Análisis de la infraestructura | **A** | **R** | | **C** | | |
| Determinación del alcance de desarrollo | **A** | **R** | **I** | **C** | **I** | **I** |

#### *5.1.2.1.2 Seguimiento del proyecto*

En la Tabla: 5.3 Tabla RACI de la fase Seguimiento del proyecto se muestra la matriz de responsabilidades para la fase de seguimiento.

Tabla 5.3: Tabla RACI de la fase Seguimiento del proyecto

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| Reunión de arranque | **A** | **C** | **I** | **I** | **I** | **R** |
| Reuniones periódicas | **A** | **I** | **I** | **C** | **I** | **R** |
| Reunión de revisión | **A** | **I** | **I** | **C** | **I** | **R** |
| Reunión final | **A** | **I** | **I** | **C** | **I** | **R** |

#### *5.1.2.1.3 Diseño del sistema*

En la Tabla: 5.4 Tabla RACI de la fase Diseño del sistema se presenta la matriz de responsabilidades correspondiente a la fase de diseño. Las tareas listadas en esta matriz son las tareas "hoja"de dicha fase, es decir, aquellas que no se descomponen en subtareas más pequeñas y, por lo tanto, representan las tareas finales de la fase de diseño.

Tabla 5.4: Tabla RACI de la fase Diseño del sistema

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| *Backend*. Diseño del módulo de usuarios | A | | I | R | | I |
| *Backend*. Diseño del módulo de cartas | A | | I | R | | I |
| *Backend*. Diseño del módulo de sobres de cartas | A | | I | R | | I |
| *Backend*. Diseño del módulo de subastas | A | | I | R | | I |
| *Backend*. Diseño del módulo de transacciones | A | | I | R | | I |
| *Frontend*. Diseño de logo de la aplicación | A | | R | I | | I |
| *Frontend*. Diseño de la moneda de la aplicación | A | | R | I | | I |
| *Frontend*. Diseño de la temática | A | | R | C/I | | I |
| *Frontend*. Diseño del árbol de navegación | A | | R | C/I | | I |
| *Frontend*. Diseño de las páginas de información | A | | R | C/I | | I |
| *Frontend*. Diseño de la página *Home* | A | | R | C/I | | I |
| *Frontend*. Diseño de la página de error | A | | R | C/I | | I |
| *Frontend*. Diseño del módulo de usuarios | A | | R | C/I | | I |
| *Frontend*. Diseño del módulo de cartas | A | | R | C/I | | I |
| *Frontend*. Diseño del módulo de sobres de cartas | A | | R | C/I | | I |
| *Frontend*. Diseño del módulo de subastas | A | | R | C/I | | I |
| *Frontend*. Diseño del módulo de transacciones | A | | R | C/I | | I |
| Diseño de las pruebas unitarias | A | | | C/I | R | I |
| Diseño de las pruebas de carga/estrés | A | | | C/I | R | I |
| Diseño de las pruebas *end-to-end* | A | | | C/I | R | I |

#### *5.1.2.1.4 Implementación del sistema*

En la Tabla: 5.5 Implementación del sistema se muestra la matriz de responsabilidades para la fase de implementación. Igual que en el caso anterior, las tareas listadas en esta matriz son las tareas finales de dicha fase.

Tabla 5.5: Tabla RACI de la fase Implementación del sistema

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| *Backend*. Implementación del módulo de usuarios | **A** | | | **R** | **I** | **I** |
| *Backend*. Implementación del módulo de cartas | **A** | | | **R** | **I** | **I** |
| *Backend*. Implementación del módulo de sobres de cartas | **A** | | | **R** | **I** | **I** |
| *Backend*. Implementación del módulo de subastas | **A** | | | **R** | **I** | **I** |
| *Backend*. Implementación del módulo de transacciones | **A** | | | **R** | **I** | **I** |
| *Frontend*. Implementación de la temática | **A** | | **C** | **R** | | **I** |
| *Frontend*. Implementación de las rutas de navegación | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación de las páginas de información | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación de la página *Home* | **A** | | C | **R** | **I** | **I** |
| *Frontend*. Implementación de la página de error | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación del módulo de usuarios | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación del módulo de cartas | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación del módulo de sobres de cartas | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación del módulo de subastas | **A** | | **C** | **R** | **I** | **I** |
| *Frontend*. Implementación del módulo de transacciones | **A** | | **C** | **R** | **I** | **I** |
| Implementación de las pruebas unitarias | **A** | | | **C/I** | **R** | **I** |
| Implementación de las pruebas de carga/estrés | **A** | | | **C/I** | **R** | **I** |
| Implementación de las pruebas *end-to-end* | **A** | | | **C/I** | **R** | **I** |

#### 5.1.2.1.5 *Fase de pruebas*

A continuación, se detallan las responsabilidades de cada rol en la fase de pruebas del sistema. En la Tabla: 5.6 Tabla RACI de la Fase de pruebas, se muestra la matriz de responsabilidades para la fase de pruebas.

Tabla 5.6: Tabla RACI de la Fase de pruebas

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| Ejecución de pruebas unitarias | **A** | | | **C/I** | **R** | **I** |
| Ejecución de pruebas de carga/estrés | **A** | | | **C/I** | **R** | **I** |
| Ejecución de pruebas *end-to-end* | **A** | | | **C/I** | **R** | **I** |

#### 5.1.2.1.6 *Despliegue del sistema*

En la Tabla: 5.7 Tabla RACI de la fase Despliegue del sistema, se muestra la matriz de responsabilidades para la fase de despliegue.

Tabla 5.7: Tabla RACI de la fase Despliegue del sistema

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| Configuración del servidor Azure | **A** | **C** | | **R** | | **I** |
| Despliegue del sistema | **A** | | | **R** | | **I** |

#### 5.1.2.1.7 *Documentación del proyecto*

Por último, en la Tabla: 5.8 Tabla RACI de la fase Documentación del proyecto, se muestra la matriz de responsabilidades para la fase de documentación.

Tabla 5.8: Tabla RACI de la fase Documentación del proyecto

| | Roles | | | | | |
|---|---|---|---|---|---|---|
| **Tarea** | **JP** | **AN** | **DI** | **DS** | **TE** | **DOC** |
| Creación de la plantilla LaTeX | **A** | | | | | **R** |
| Redacción del documento final | **A** | | | | | **R** |
| Redacción de anexos | **A** | | | | | **R** |
| Presentación del proyecto | **A** | | | | | **R** |

#### 5.1.2.2. PBS

El PBS, *Product Breakdown Structure*, es una estructura jerárquica que descompone el proyecto en los productos que se deben entregar para cumplir con los objetivos del proyecto. En las siguientes secciones se detallan los productos que se deben entregar en el proyecto BidMon Universe.

#### 5.1.2.3. PBS. Visión general

En la Figura 5.2: PBS. Visión general se muestran los productos de alto nivel que se deben entregar en el proyecto BidMon Universe. En las siguientes secciones, se entrará en detalle en cada uno de los productos.

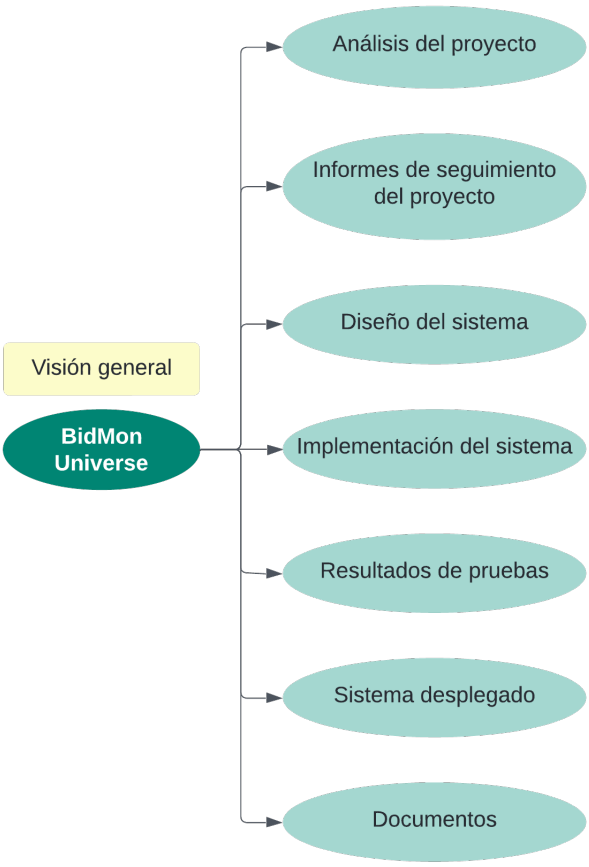


Figura 5.2: PBS. Visión general

#### 5.1.2.4. PBS. Análisis del sistema

En la Figura 5.3: PBS. Análisis del sistema, se detallan los productos que se deben entregar en la fase de análisis del sistema.

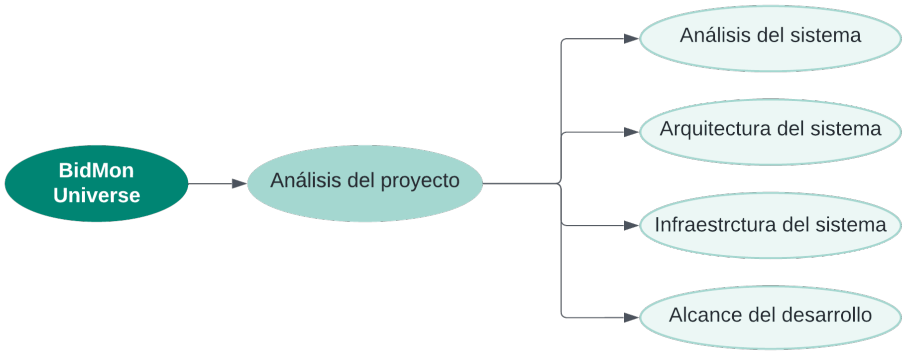


Figura 5.3: PBS. Análisis del sistema

#### 5.1.2.5. PBS. Seguimiento del sistema

En esta fase se detallan los productos que se obtienen en la fase de segumiento del proyecto, principalmente documentación e informes como se muestra en la Figura 5.4: PBS. Seguimiento del sistema.

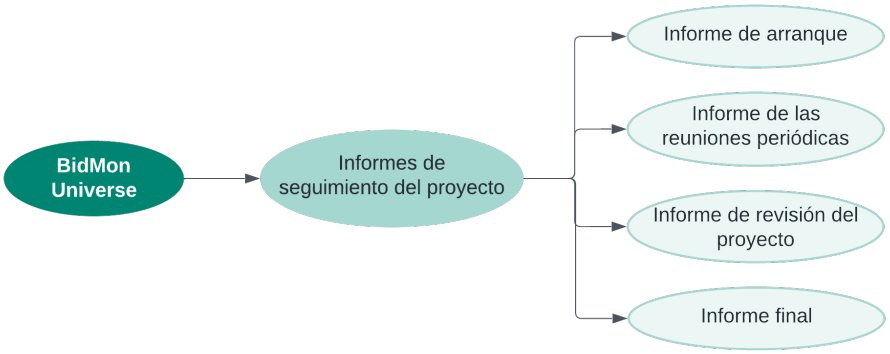


Figura 5.4: PBS. Seguimiento del sistema

#### 5.1.2.6. PBS. Diseño del sistema

En la Figura 5.5: PBS. Diseño del sistema, se detallan los productos que se deben entregar en la fase de diseño del sistema.

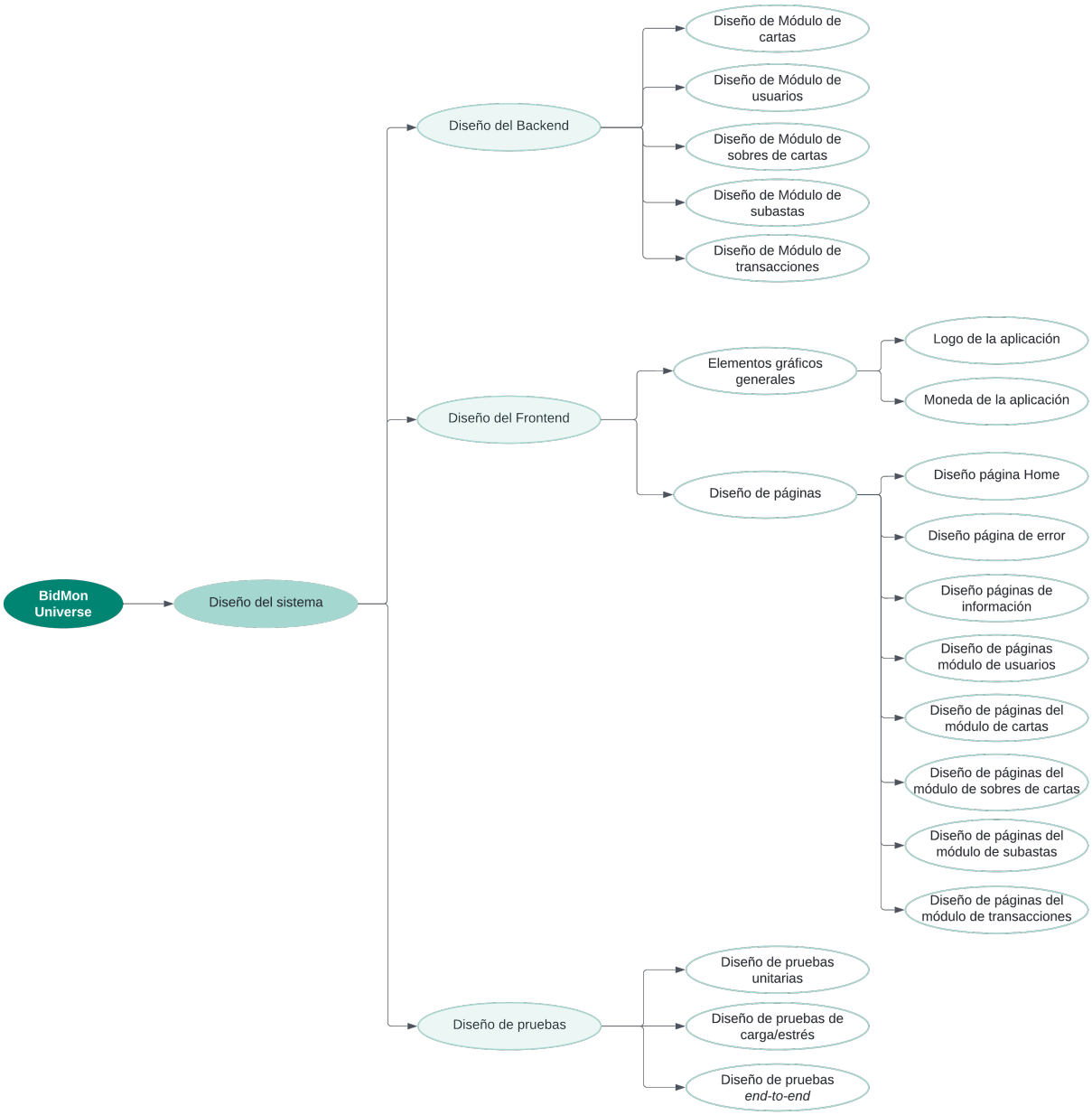


Figura 5.5: PBS. Diseño del sistema

#### 5.1.2.7. PBS. Implementación del sistema

En la Figura 5.6: PBS. Implementación del sistema, se detallan los productos a realizar en la fase de implementación del sistema.

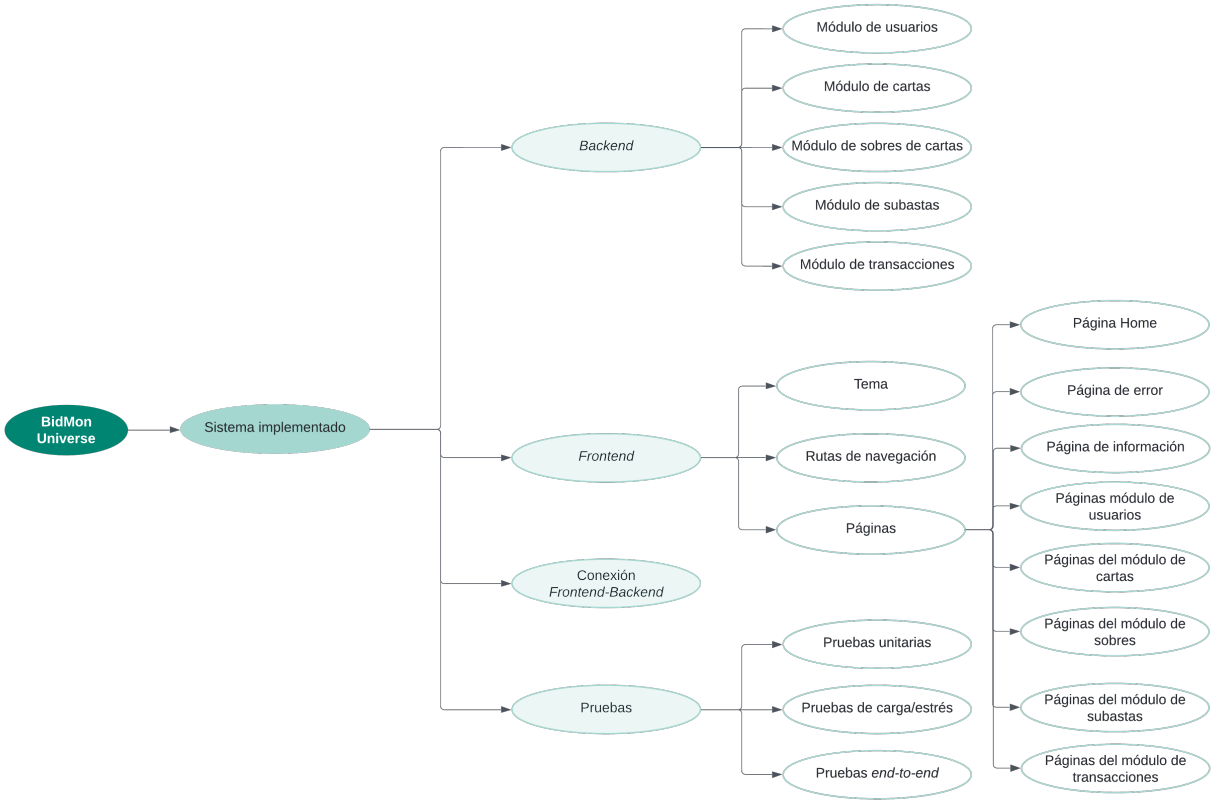


Figura 5.6: PBS. Implementación del sistema

#### 5.1.2.8. PBS. Pruebas del sistema

En la fase de pruebas del sistema se obtienen como productos los resultados de la ejecución de dichas pruebas.

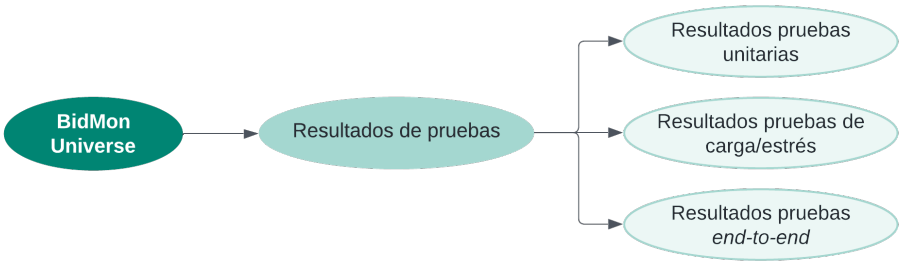


Figura 5.7: PBS. Pruebas del sistema

#### 5.1.2.9. PBS. Despliegue del sistema

En la fase de despliegue del sistema se obtiene como producto el sistema desplegado y en funcionamiento.

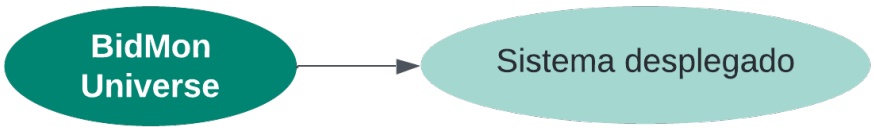


Figura 5.8: PBS. Despliegue del sistema

#### 5.1.2.10. PBS. Documentación del sistema

En la fase de documentación del sistema se obtienen como productos los documentos técnicos que describen el proyecto junto con los anexos.

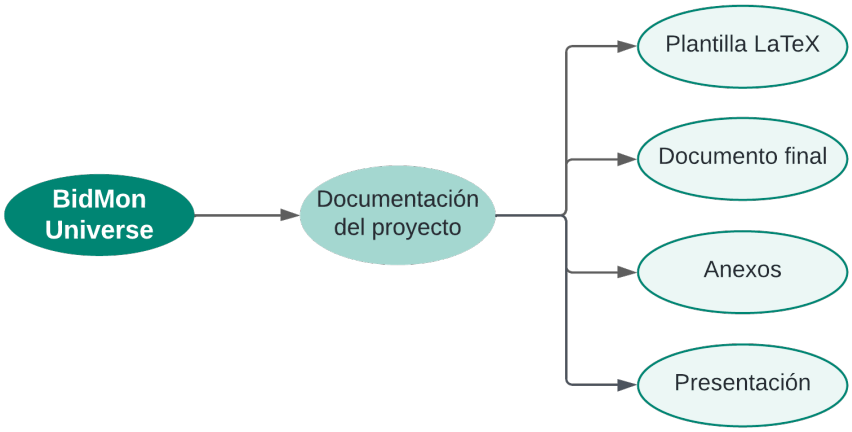


Figura 5.9: PBS. Documentación del sistema

### 5.1.3. Planificación Inicial. WBS

#### 5.1.3.1. WBS

En esta sección se detalla la estructura de desglose del trabajo del proyecto también conocida como WBS, *Work Breakdown Structure*. En ella se especifican las tareas necesarias para obtener los productos detallados en 5.1.2.2 PBS.

Estas tareas se representan en forma de árbol jerárquico, donde cada rama representa una tarea y sus subramas las tareas que la componen. El diagrama se ha dividido en las fases en las que se divide el proyecto para mejorar la legibilidad, facilitando la comprensión de las tareas y sub-tareas que se deben realizar en cada una de ellas.

##### *5.1.3.1.1 WBS. Visión general*

En la Figura 5.10: WBS. Visión general se muestra la estructura de desglose del trabajo del proyecto de alto nivel, es decir, las tareas generales o fases que se deben realizar para cumplir con los objetivos del proyecto. En las siguientes secciones, se entrará en detalle en cada una de las tareas.

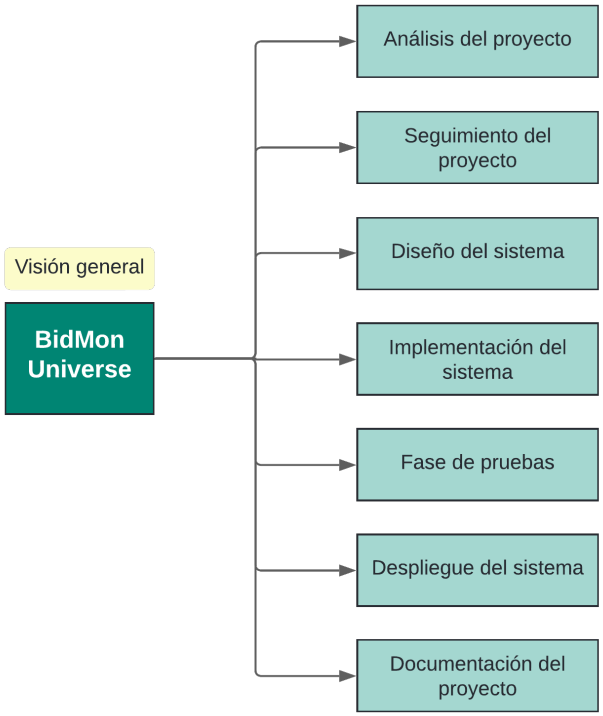


Figura 5.10: WBS. Visión general

##### *5.1.3.1.2 WBS. Análisis del proyecto*

En la Figura 5.11: WBS. Análisis del proyecto, se detallan las tareas que se deben realizar en la fase de análisis del sistema para cumplir con los objetivos del proyecto.

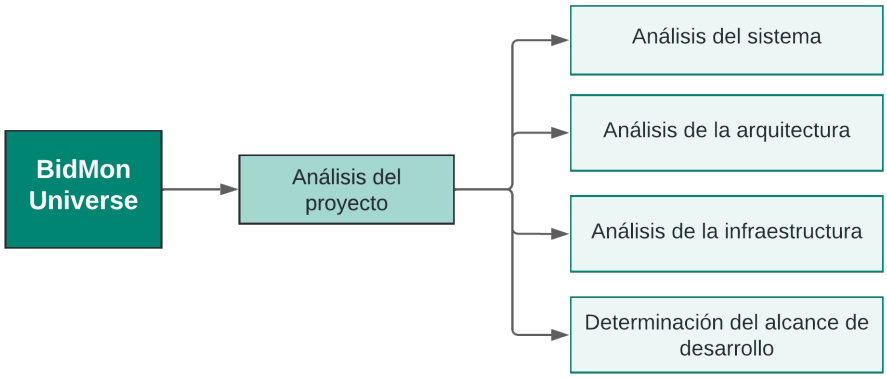


Figura 5.11: WBS. Análisis del proyecto

#### *5.1.3.1.3 WBS. Seguimiento del sistema*

En esta fase se realizan las tareas de seguimiento del proyecto, a través de distintas reuniones en las que se recopilará infomarción sobre el avance del proyecto.

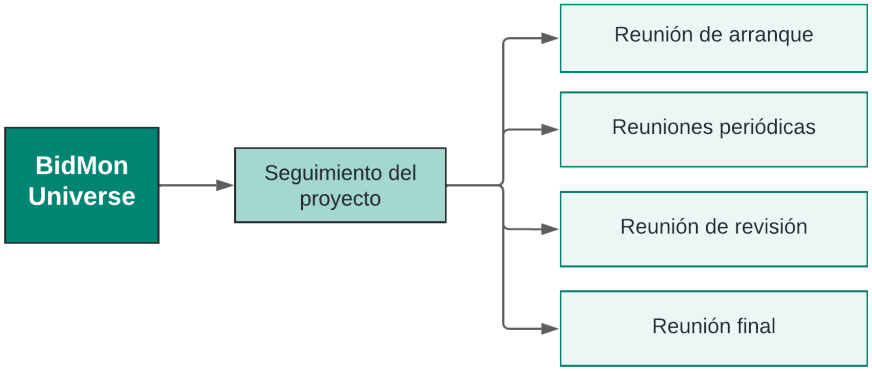


Figura 5.12: WBS. Seguimiento del sistema

#### *5.1.3.1.4 WBS. Diseño del sistema*

En la Figura 5.13: WBS. Diseño del sistema, se detallan las tareas que se deben realizar en la fase de diseño del sistema.

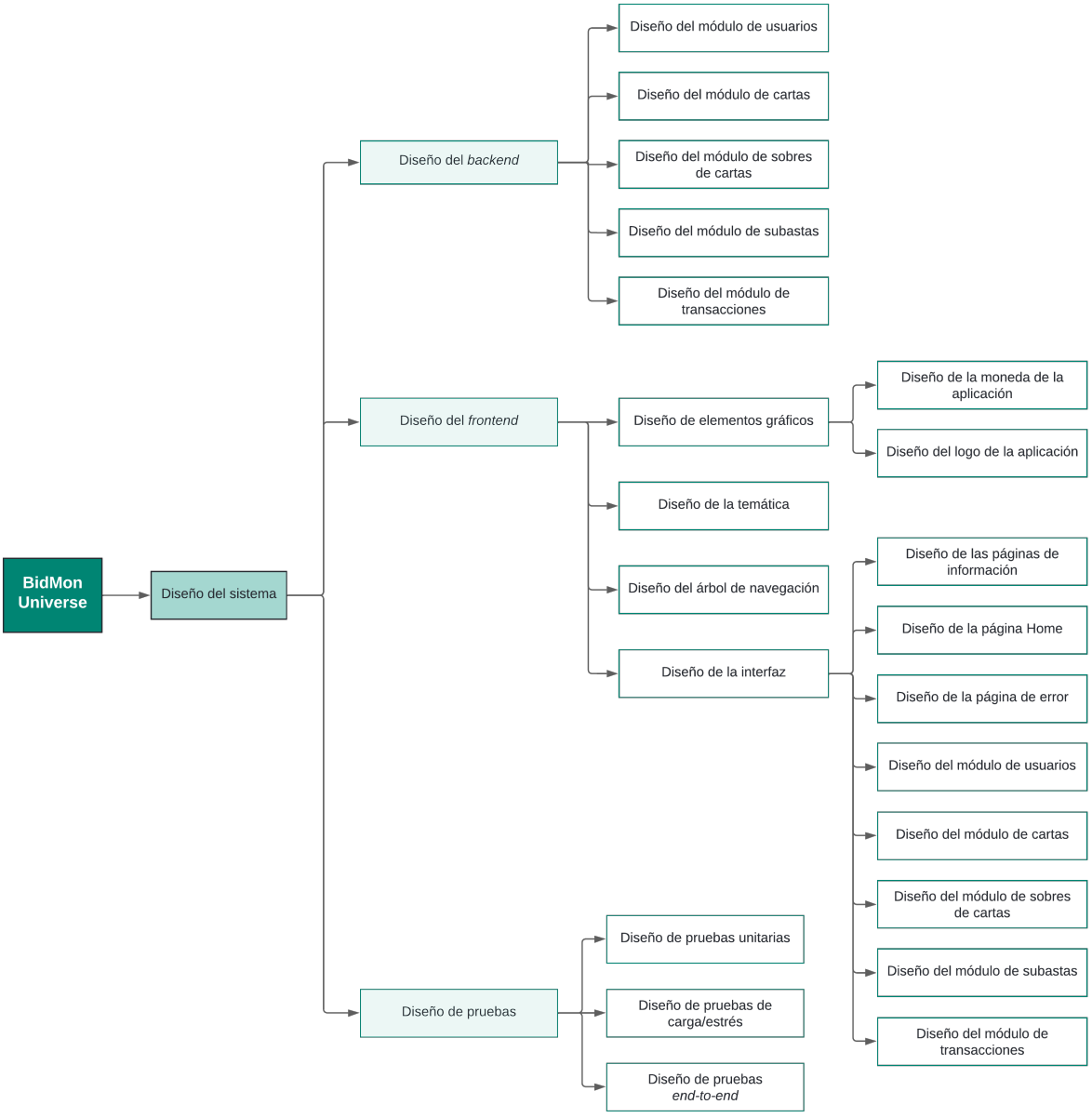


Figura 5.13: WBS. Diseño del sistema

#### *5.1.3.1.5 WBS. Implementación del sistema*

En la Figura 5.14: WBS. Implementación del sistema, se detallan las tareas que se deben realizar en la fase de implementación del sistema.

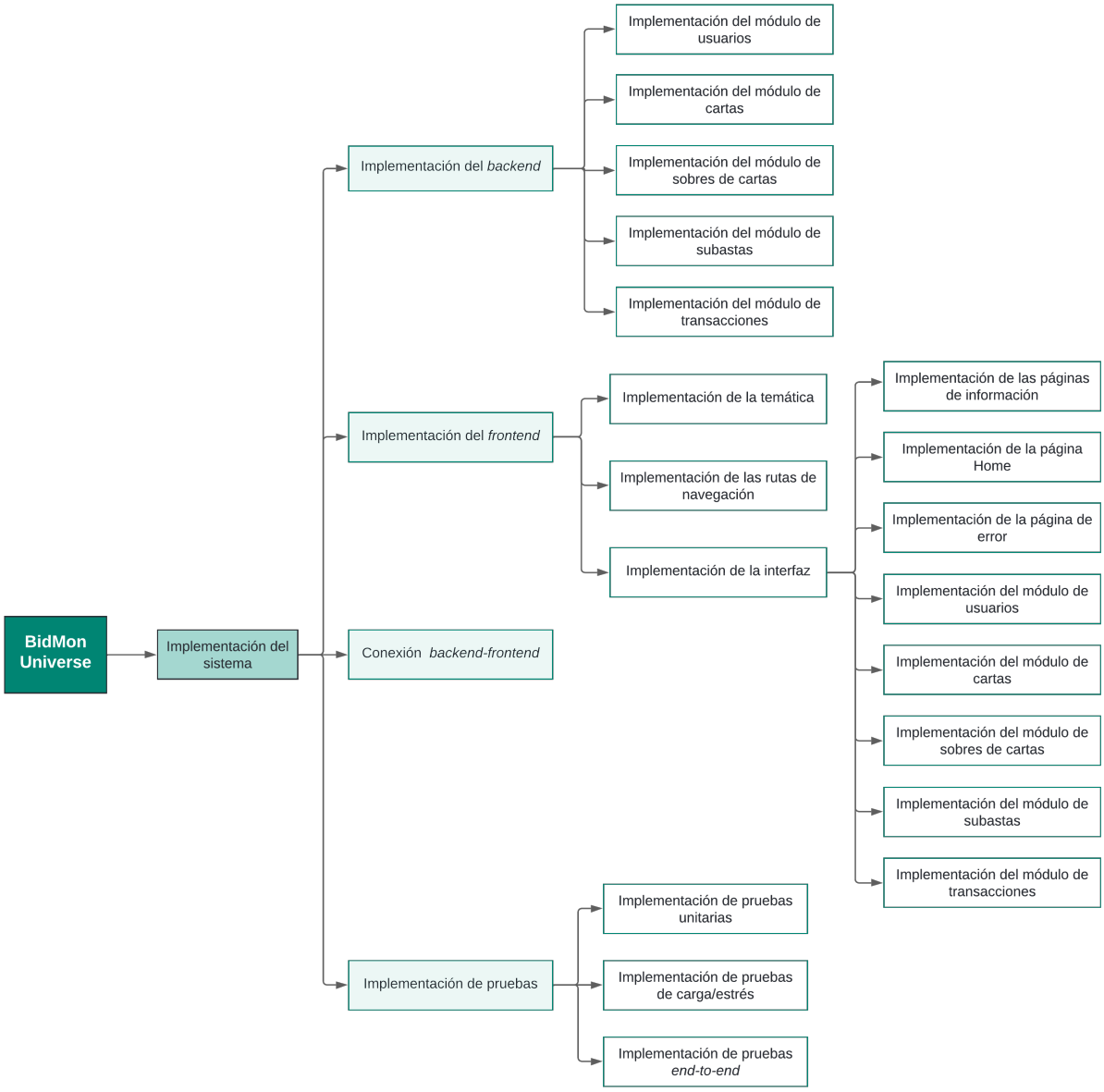


Figura 5.14: WBS. Implementación del sistema

#### *5.1.3.1.6 WBS. Fase de pruebas*

En la fase de pruebas del sistema se realizan las tareas necesarias para comprobar que el sistema cumple con los requisitos establecidos, recogiendo los informes especificados en Figura 5.7: PBS. Pruebas del sistema.

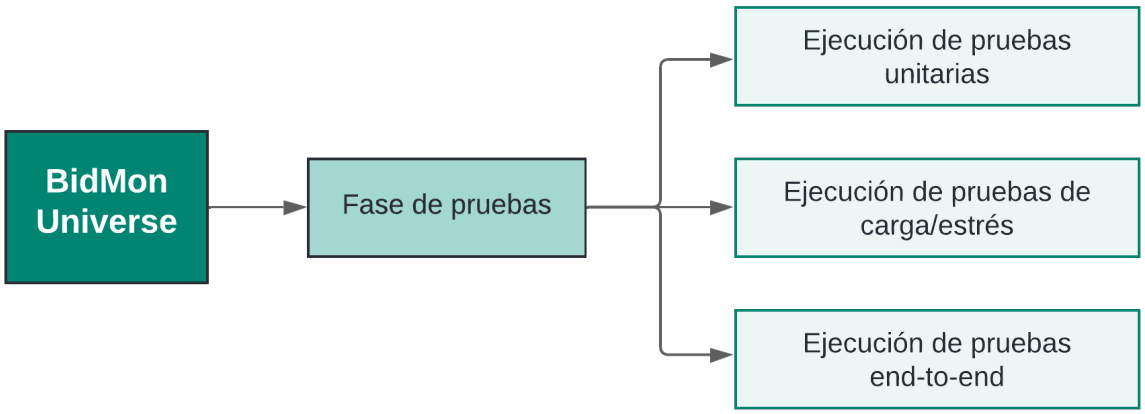


Figura 5.15: WBS. Fase de pruebas

#### *5.1.3.1.7 WBS. Despliegue del sistema*

En la fase de despliegue del sistema se realizan las tareas necesarias para poner en producción el sistema, como se detalla en la Figura 5.16: WBS. Despliegue del sistema.

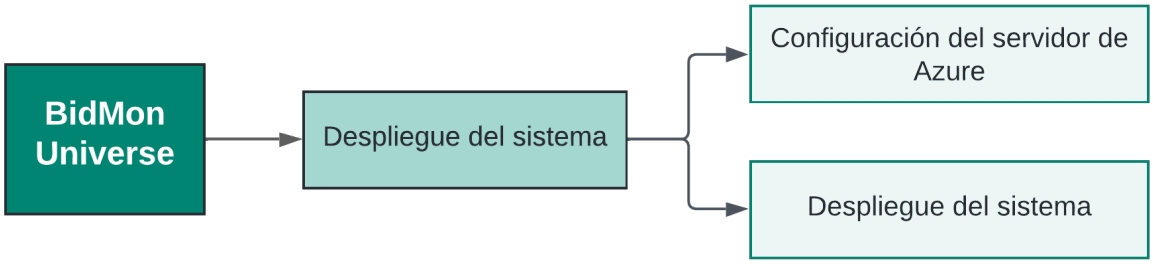


Figura 5.16: WBS. Despliegue del sistema

#### *5.1.3.1.8 WBS. Documentación*

En la fase de documentación se realizan las tareas necesarias para la redacción de la memoria del proyecto, así como la preparación de la presentación del mismo.

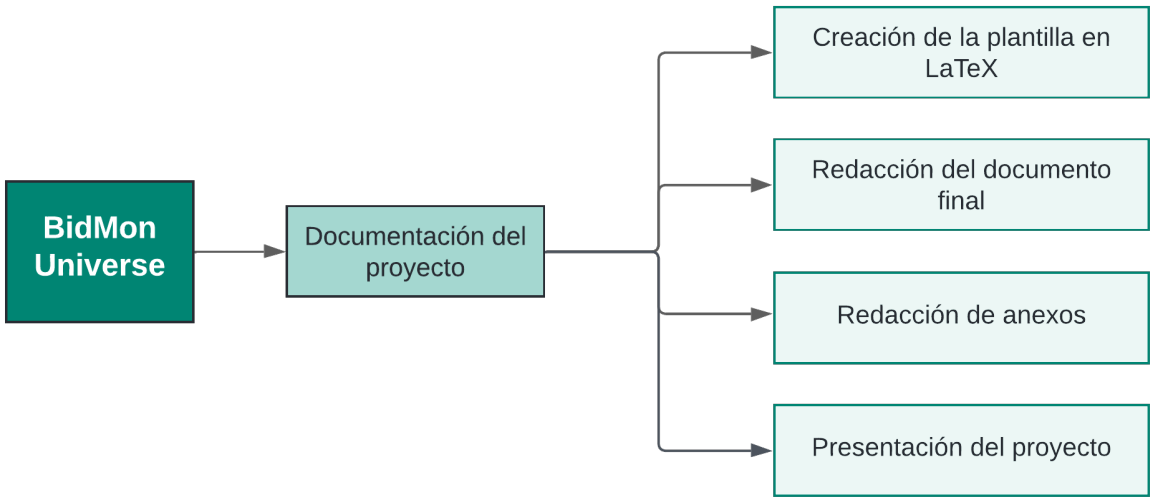


Figura 5.17: WBS. Documentación

#### 5.1.3.2. Planificación inicial

A partir de las tareas definidas en la sección 5.1.3.1 WBS se ha realizado una planificación inicial del proyecto, en la que se han establecido las fechas de inicio y fin de cada tarea, así como su duración y las dependencias entre ellas. La planificación inicial se ha realizado en la herramienta Microsoft Project. Al igual que en la sección anterior, se ha dividido la planificación en las fases en las que se divide el proyecto para mejorar la legibilidad mostrando primero una visión general y después entrando en detalle en cada una de las fases. Se incluye una tabla con las siguientes columnas:

- **EDT:** Estructura de Desglose del Trabajo, también conocida como WBS. Detalla el identificador de la tarea.
- **Nombre tarea:** Nombre de la tarea.
- **Duración:** Duración de la tarea en horas.
- **Fecha inicio:** Fecha de inicio de la tarea.
- **Fecha fin:** Fecha de fin de la tarea.

Además, se incluye una línea temporal del proyecto para visualizar de forma más clara la planificación del proyecto así como el diagrama de Gantt de cada una de las fases del proyecto.

##### *5.1.3.2.1 Planificación inicial. Visión general*

En la 5.9: Planificación inicial. Visión general se muestra la planificación inicial del proyecto de alto nivel, es decir, las tareas generales o fases que se deben realizar para cumplir con los objetivos del proyecto. El total de días estimados para la realización del proyecto es de 71,13 días (462 horas), desde el 01/04/2024 hasta el 18/06/2024.

Tabla 5.9: Planificación inicial. Visión general

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1 | Proyecto BidMon Universe | 462 horas | 01/04/2024 | 18/06/2024 |
| 1.1 | Análisis del proyecto | 12 horas | 01/04/2024 | 12/04/2024 |
| 1.2 | Seguimiento del proyecto | 30,5 horas | 01/04/2024 | 18/06/2024 |
| 1.3 | Diseño del sistema | 76 horas | 03/04/2024 | 15/05/2024 |
| 1.4 | Implementación del sistema | 156,5 horas | 03/04/2024 | 22/05/2024 |
| 1.5 | Fase de pruebas | 9 horas | 28/05/2024 | 01/06/2024 |
| 1.6 | Despliegue del sistema | 8 horas | 12/06/2024 | 13/06/2024 |
| 1.7 | Documentación del proyecto | 170 horas | 02/04/2024 | 18/06/2024 |

En la Figura 5.18: Planificación inicial. Línea temporal se muestra la línea temporal del proyecto. Mediante esta planificación se pretende priorizar las tareas de análisis y diseño del sistema, para posteriormente realizar la implementación y las pruebas del sistema, y finalmente desplegar el sistema. La fase de documentación del proyecto se realizará de forma paralela a las demás tareas al igual que el seguimiento del proyecto.

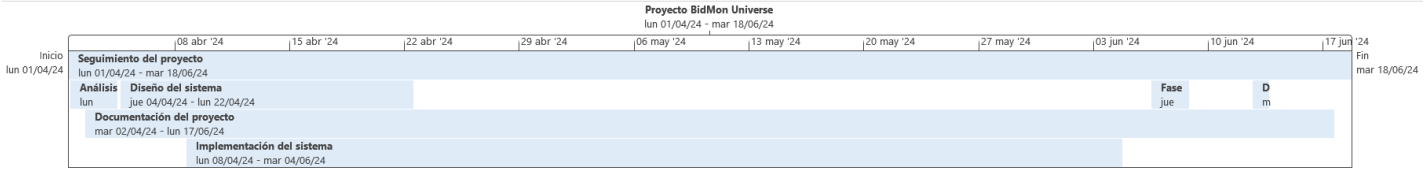


Figura 5.18: Planificación inicial. Línea temporal

En la Figura 5.19: Planificación inicial. Diagrama de Gantt, visión general se muestra el diagrama de Gantt de la planificación inicial del proyecto.


Figura 5.19: Planificación inicial. Diagrama de Gantt, visión general

#### *5.1.3.2.2 Planificación inicial. Análisis del proyecto*

En la 5.10: Planificación inicial. Análisis del proyecto, se detallan la planificación de las tareas que se deben realizar en la fase de análisis del sistema. El total de horas estimadas para la realización de esta fase es de 12 horas, desde el 01/04/2024 hasta el 12/04/2024.

Tabla 5.10: Planificación inicial. Análisis del proyecto

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.1 | Análisis del proyecto | 12 horas | 01/04/2024 | 12/04/2024 |
| 1.1.1 | Análisis del sistema | 5 horas | 01/04/2024 | 02/04/2024 |
| 1.1.2 | Análisis de la arquitectura | 3 horas | 11/04/2024 | 11/04/2024 |
| 1.1.3 | Análisis de la infraestructura | 4 horas | 12/04/2024 | 12/04/2024 |
| 1.1.4 | Determinación del análisis | 2 horas | 12/04/2024 | 12/04/2024 |

En la Figura 5.20: Planificación inicial. Diagrama de Gantt, análisis del proyecto, se muestra el diagrama de Gantt de la planificación inicial de la fase de análisis del proyecto.

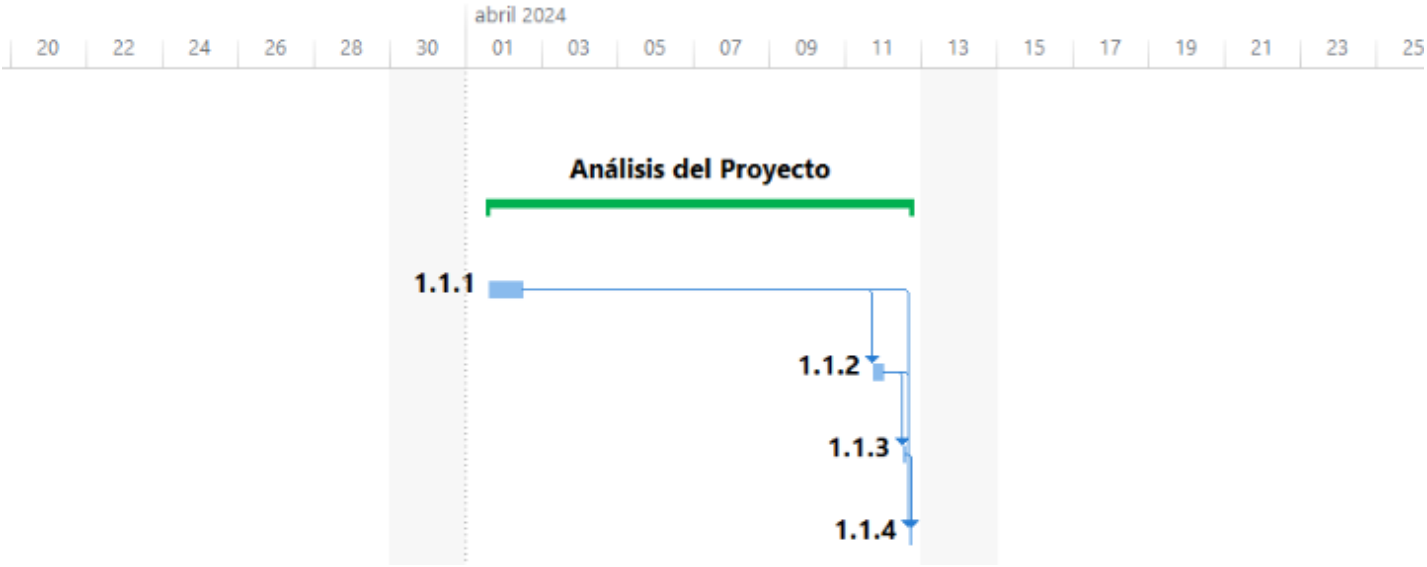


Figura 5.20: Planificación inicial. Diagrama de Gantt, análisis del proyecto

#### *5.1.3.2.3 Planificación inicial. Seguimiento del proyecto*

En la 5.11: Planificación inicial. Seguimiento del proyecto, se detalla la planificación de las tareas que se deben realizar en la fase de seguimiento del proyecto. El total de horas estimadas para la realización de esta fase es de 30,5 horas, desde el 01/04/2024 hasta el 18/06/2024, abarca todo el proyecto debido a que se realiza una reunión inicial, reuniones periódicas y una reunión final para la revisión del proyecto.

Tabla 5.11: Planificación inicial. Seguimiento del proyecto

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.2 | Seguimiento del proyecto | 30,5 horas | 01/04/2024 | 12/04/2024 |
| 1.2.1 | Reunión de arranque | 2 horas | 01/04/2024 | 01/04/2024 |
| 1.2.2 | Reuniones periódicas | 20,5 horas | 27/04/2024 | 30/04/2024 |
| 1.2.3 | Reunión de revisión | 4 horas | 15/06/2024 | 15/06/2024 |
| 1.2.4 | Reunión final | 4 horas | 17/04/2024 | 18/04/2024 |

En la Figura 5.21: Planificación inicial. Diagrama de Gantt, seguimiento del proyecto, se muestra el diagrama de Gantt de la planificación inicial de la fase de seguimiento del proyecto.


Figura 5.21: Planificación inicial. Diagrama de Gantt, seguimiento del proyecto

#### *5.1.3.2.4 Planificación inicial. Diseño del sistema*

En la 5.12: Planificación inicial. Diseño del sistema, se detalla la planificación de las tareas que se deben realizar en la fase de diseño del sistema. El total de horas estimadas para la realización de esta fase es de 76 horas, desde el 03/04/2024 hasta el 15/05/2024.

Tabla 5.12: Planificación inicial. Diseño del sistema

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.3 | Diseño del sistema | 76 horas | 03/04/2024 | 15/05/2024 |
| 1.3.1 | Diseño del *backend* | 22 horas | 03/04/2024 | 16/04/2024 |
| 1.3.1.1 | Diseño del módulo de usuarios | 2 horas | 08/04/2024 | 08/04/2024 |
| 1.3.1.2 | Diseño del módulo de cartas | 6 horas | 03/04/2024 | 03/04/2024 |
| 1.3.1.3 | Diseño del módulo de sobres de cartas | 3 horas | 06/04/2024 | 08/04/2024 |
| 1.3.1.4 | Diseño del módulo de subastas | 6.5 horas | 09/04/2024 | 10/04/2024 |
| 1.3.1.5 | Diseño del módulo de transacciones | 4.5 horas | 16/04/2024 | 16/04/2024 |
| 1.3.2 | Diseño del *frontend* | 11.81 días | 13/04/2024 | 26/04/2024 |
| 1.3.2.1 | Diseño de elementos gráficos | 1.13 días | 13/04/2024 | 13/04/2024 |

Planificación inicial. Diseño del sistema – Continúa en la siguiente página...

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.3.2.1.1 | Diseño de la moneda de la aplicación | 1 hora | 13/04/2024 | 13/04/2024 |
| 1.3.2.1.2 | Diseño del logo de la aplicación | 2 horas | 13/04/2024 | 13/04/2024 |
| 1.3.2.2 | Diseño de la temática | 2 horas | 13/04/2024 | 13/04/2024 |
| 1.3.2.3 | Diseño del árbol de navegación | 2 horas | 13/04/2024 | 13/04/2024 |
| 1.3.2.4 | Diseño de la interfaz de usuario | 9.44 días | 15/04/2024 | 26/04/2024 |
| 1.3.2.4.1 | Diseño de las páginas de información | 3 horas | 24/04/2024 | 25/04/2024 |
| 1.3.2.4.2 | Diseño de la página Home | 4 horas | 23/04/2024 | 24/04/2024 |
| 1.3.2.4.3 | Diseño de la página de error | 1 hora | 25/04/2024 | 25/04/2024 |
| 1.3.2.4.4 | Diseño del módulo de usuarios | 5.5 horas | 22/04/2024 | 23/04/2024 |
| 1.3.2.4.5 | Diseño del módulo de cartas | 4 horas | 19/04/2024 | 19/04/2024 |
| 1.3.2.4.6 | Diseño del módulo de sobres de cartas | 2 horas | 24/04/2024 | 24/04/2024 |
| 1.3.2.4.7 | Diseño del módulo de subastas | 5.5 horas | 15/04/2024 | 15/04/2024 |
| 1.3.2.4.8 | Diseño del módulo de transacciones | 4 horas | 25/04/2024 | 26/04/2024 |
| 1.3.3 | Diseño de pruebas | 4.19 días | 11/05/2024 | 15/05/2024 |
| 1.3.3.1 | Diseño de pruebas unitarias | 6.5 horas | 11/05/2024 | 11/05/2024 |
| 1.3.3.2 | Diseño de pruebas de carga/estrés | 5 horas | 14/05/2024 | 15/05/2024 |
| 1.3.3.3 | Diseño de pruebas *end-to-end* | 6.5 horas | 11/05/2024 | 13/05/2024 |

A continuación se muestra el diagrama de Gantt de la planificación inicial de la fase de diseño del sistema. Se ha dividido en tres partes para mejorar la legibilidad estas partes son: diseño del *backend*, diseño del *frontend* y diseño de pruebas. En la Figura 5.22: Planificación inicial. Diagrama de Gantt, diseño del sistema *backend*, se muestra parte de la fase de diseño del sistema correspondiente con el *backend*.

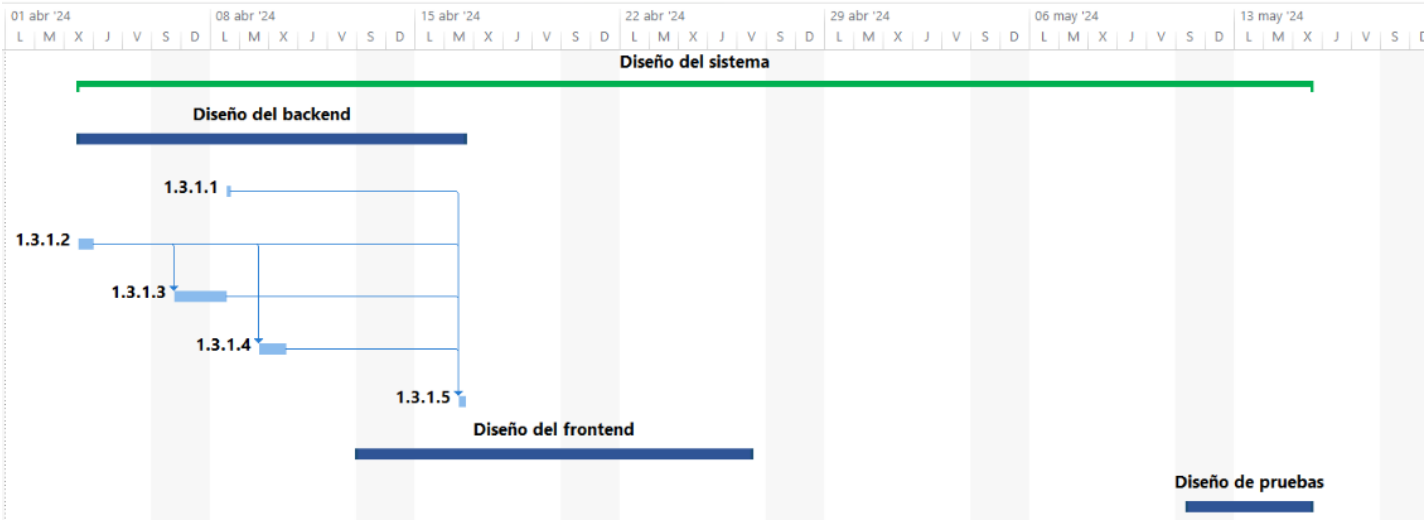


Figura 5.22: Planificación inicial. Diagrama de Gantt, diseño del sistema *backend*

En la Figura 5.23: Planificación inicial. Diagrama de Gantt, diseño del sistema *frontend*, se muestra parte de la fase de diseño del sistema correspondiente con el *frontend*.

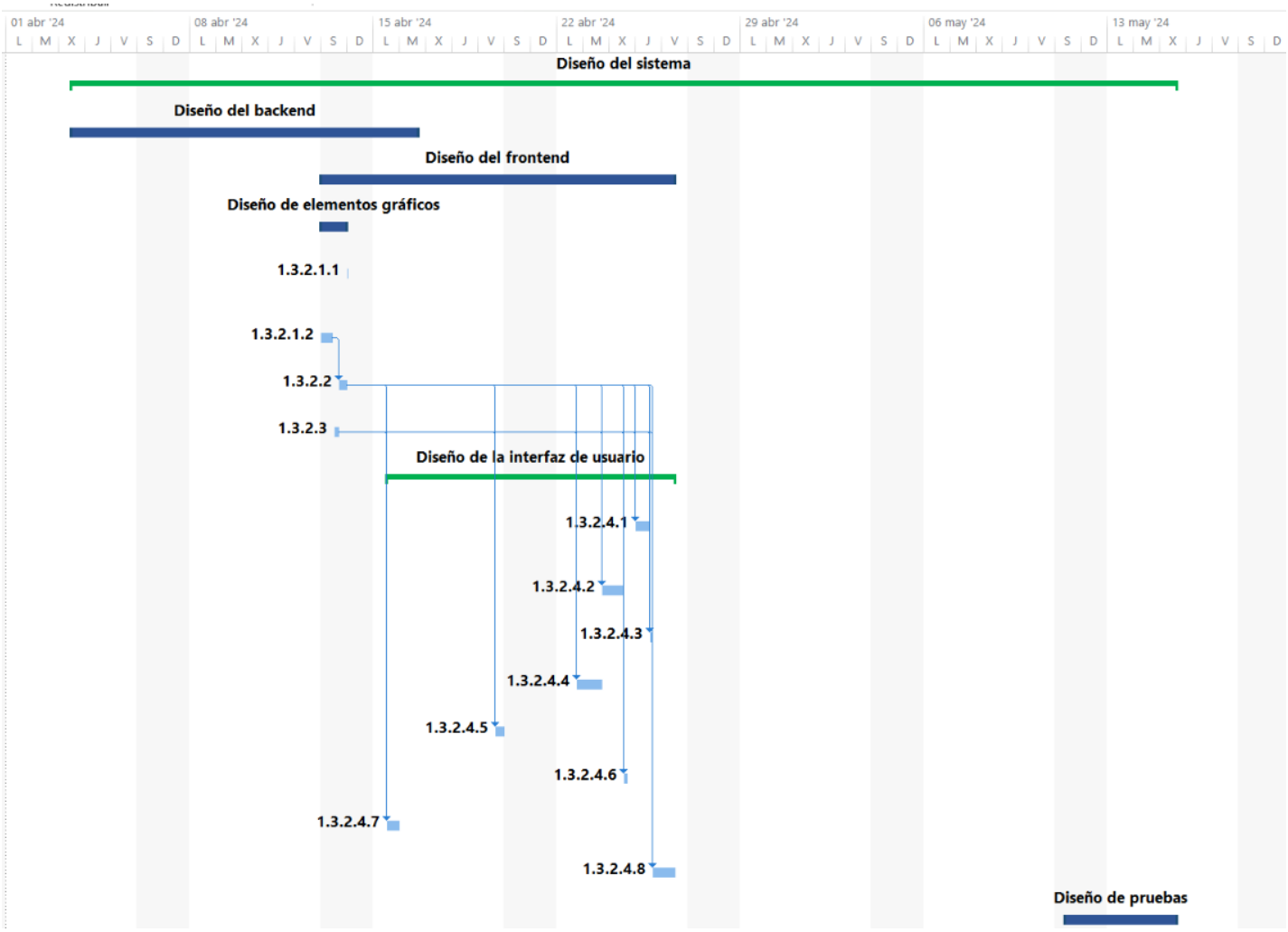


Figura 5.23: Planificación inicial. Diagrama de Gantt, diseño del sistema *frontend*

En la Figura 5.24: Planificación inicial. Diagrama de Gantt, diseño de pruebas, se muestra parte de la fase de diseño del sistema correspondiente con las pruebas.

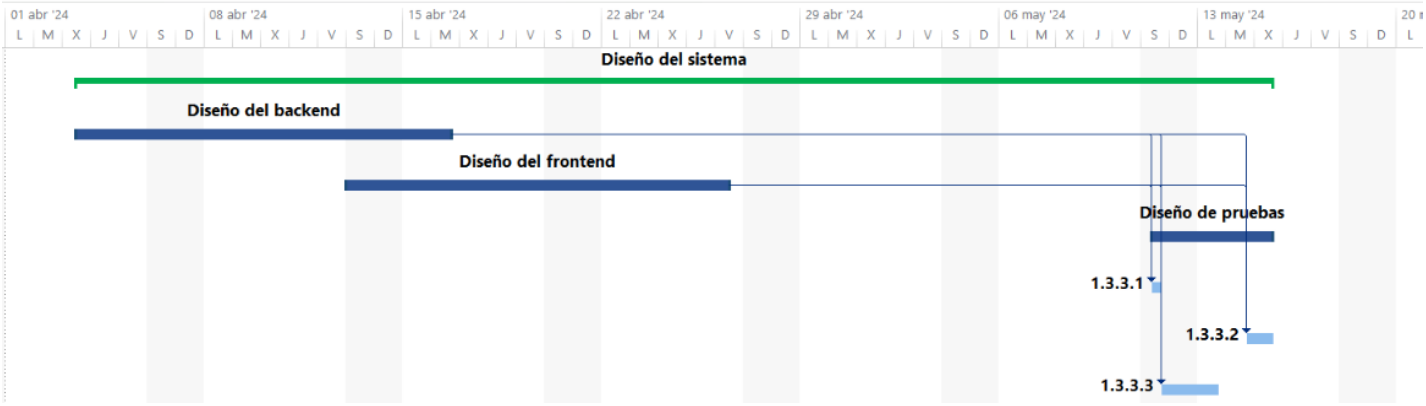


Figura 5.24: Planificación inicial. Diagrama de Gantt, diseño de pruebas

#### *5.1.3.2.5 Planificación inicial. Implementación del sistema*

En la 5.13: Planificación inicial. Implementación del sistema, se detalla la planificación de las tareas que se deben realizar en la fase de implementación del sistema. El total de horas estimadas para la realización de esta fase es de 156,5 horas, desde el 03/04/2024 hasta el 22/05/2024.

Tabla 5.13: Planificación inicial. Implementación del sistema

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.4 | Implementación del sistema | 43.44 días | 03/04/2024 | 22/05/2024 |
| 1.4.1 | Implementación del *backend* | 39.13 días | 03/04/2024 | 17/05/2024 |
| 1.4.1.1 | Implementación del módulo de usuarios | 5.5 horas | 08/04/2024 | 09/04/2024 |
| 1.4.1.2 | Implementación del módulo de cartas | 20.5 horas | 03/04/2024 | 06/04/2024 |
| 1.4.1.3 | Implementación del módulo de sobres de cartas | 7.5 horas | 10/04/2024 | 11/04/2024 |
| 1.4.1.4 | Implementación del módulo de subastas | 17 horas | 16/04/2024 | 19/04/2024 |
| 1.4.1.5 | Implementación del módulo de transacciones | 16 horas | 15/05/2024 | 17/05/2024 |
| 1.4.2 | Implementación del *frontend* | 26.38 días | 15/04/2024 | 14/05/2024 |
| 1.4.2.1 | Implementación de la temática | 1 hora | 15/04/2024 | 15/04/2024 |
| 1.4.2.2 | Implementación de las rutas de navegación | 1 hora | 01/05/2024 | 01/05/2024 |
| 1.4.2.3 | Implementación de la interfaz | 11.69 días | 01/05/2024 | 14/05/2024 |
| 1.4.2.3.1 | Implementación de las páginas de información | 3 horas | 01/05/2024 | 02/05/2024 |
| 1.4.2.3.2 | Implementación de la página Home | 8.5 horas | 02/05/2024 | 03/05/2024 |
| 1.4.2.3.3 | Implementación de la página de error | 1 hora | 01/05/2024 | 01/05/2024 |
| 1.4.2.3.4 | Implementación del módulo de usuarios | 8.5 horas | 03/05/2024 | 04/05/2024 |

Planificación inicial. Implementación del sistema – Continúa en la siguiente página...

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.4.2.3.5 | Implementación del módulo de cartas | 10.5 horas | 04/05/2024 | 07/05/2024 |
| 1.4.2.3.6 | Implementación del módulo de sobres de cartas | 7.5 horas | 07/05/2024 | 08/05/2024 |
| 1.4.2.3.7 | Implementación del módulo de subastas | 17 horas | 08/05/2024 | 11/05/2024 |
| 1.4.2.3.8 | Implementación del módulo de transacciones | 6.5 horas | 13/05/2024 | 14/05/2024 |
| 1.4.3 | Implementación de pruebas | 4.31 días | 18/05/2024 | 22/05/2024 |
| 1.4.3.1 | Implementación de pruebas unitarias | 13.5 horas | 18/05/2024 | 20/05/2024 |
| 1.4.3.2 | Implementación de pruebas de carga/estrés | 5.5 horas | 21/05/2024 | 22/05/2024 |
| 1.4.3.3 | Implementación de pruebas *end-to-end* | 6.5 horas | 20/05/2024 | 21/05/2024 |

Al igual que en la fase de diseño del sistema, se ha dividido la fase de implementación en tres partes para mejorar la legibilidad estas partes son: implementación del *backend*, implementación del *frontend* e implementación de pruebas. En la Figura 5.25: Planificación inicial. Diagrama de Gantt, implementación del sistema *backend*, se muestra parte de la fase de implementación del sistema correspondiente con el *backend*.


Figura 5.25: Planificación inicial. Diagrama de Gantt, implementación del sistema *backend*

En la Figura 5.26: Planificación inicial. Diagrama de Gantt, implementación del sistema *frontend*, se muestra parte de la fase de implementación del sistema correspondiente con el *frontend*.

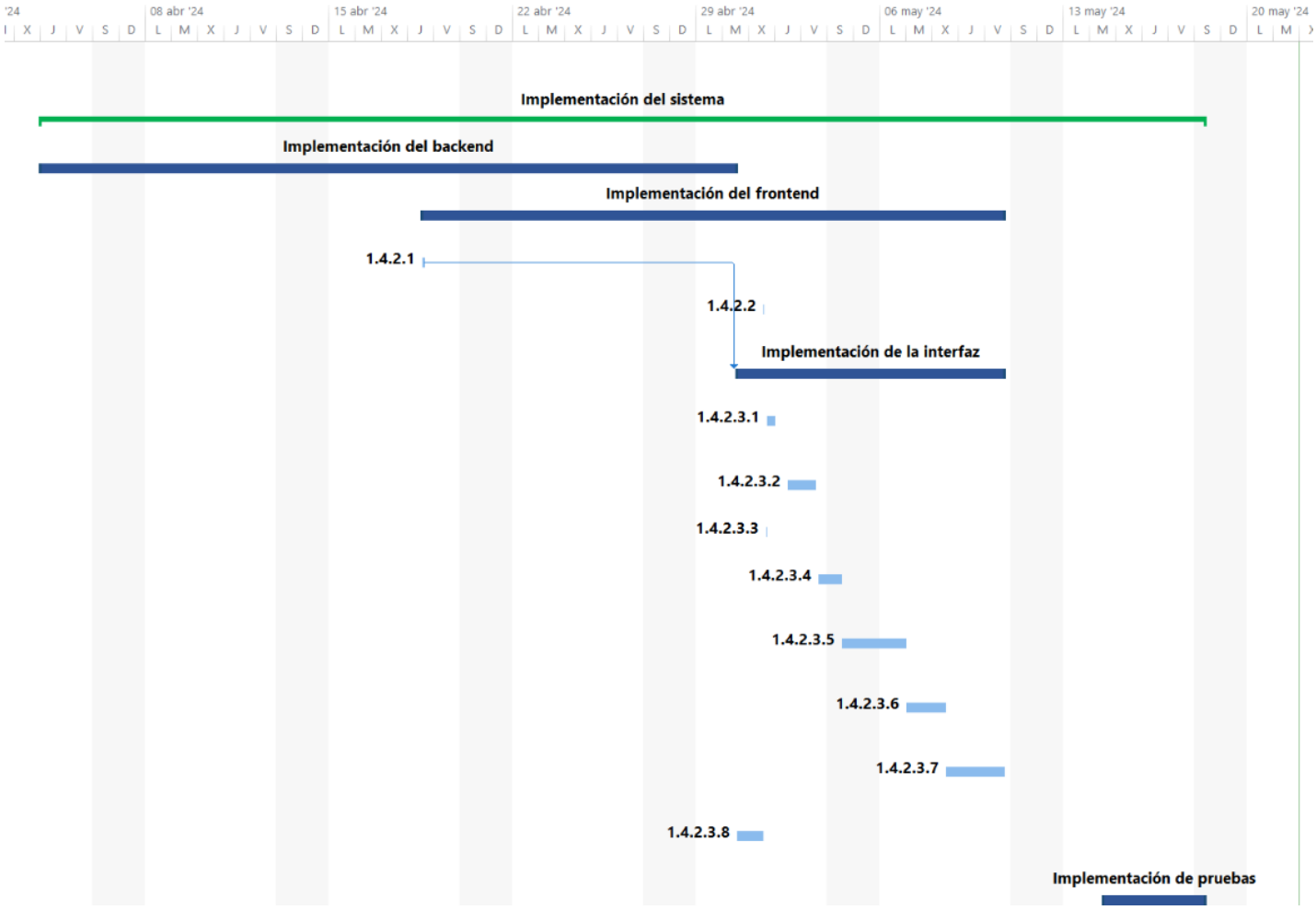


Figura 5.26: Planificación inicial. Diagrama de Gantt, implementación del sistema *frontend*

Por último, en la Figura 5.27: Planificación inicial. Diagrama de Gantt, implementación de pruebas, se muestra parte de la fase de implementación del sistema correspondiente con las pruebas.

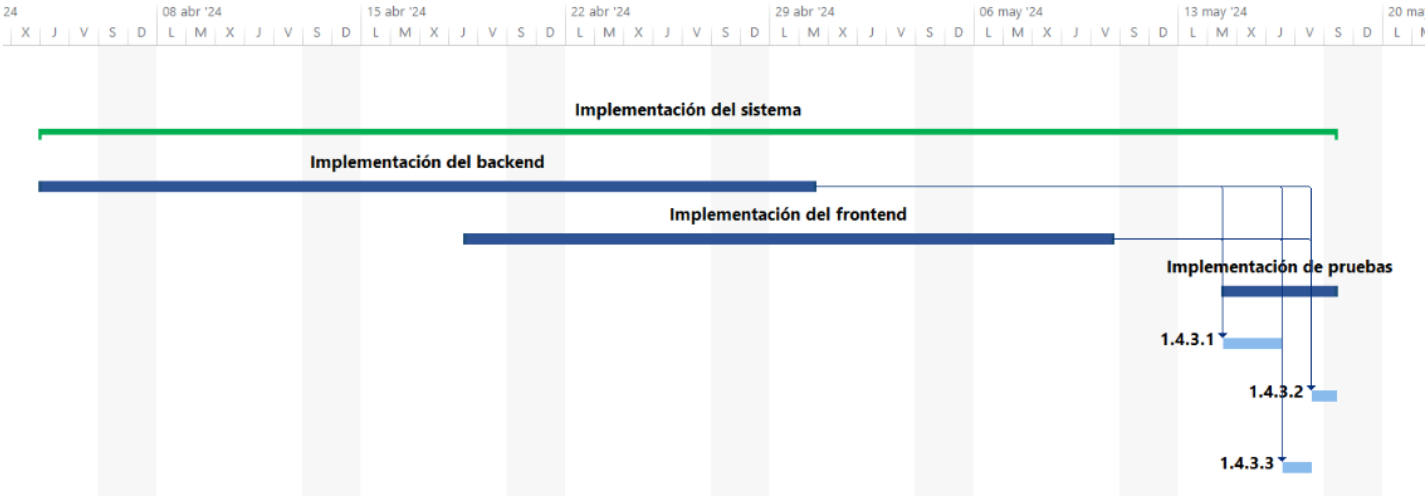


Figura 5.27: Planificación inicial. Diagrama de Gantt, implementación de pruebas

#### *5.1.3.2.6 Planificación inicial. Fase de pruebas*

En la 5.14: Planificación inicial. Fase de pruebas, se detalla la planificación de las tareas que se deben realizar en la fase de pruebas del sistema. El total de horas estimadas para la realización de esta fase es de 9 horas, desde el 28/05/2024 hasta el 01/06/2024.

Tabla 5.14: Planificación inicial. Fase de pruebas

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.5 | Fase de pruebas | 9 horas | 28/05/2024 | 01/06/2024 |
| 1.5.1 | Pruebas unitarias | 3 horas | 28/05/2024 | 29/05/2024 |
| 1.5.2 | Pruebas de carga/estrés | 3 horas | 31/05/2024 | 01/06/2024 |
| 1.5.3 | Pruebas *end-to-end* | 3 horas | 29/05/2024 | 29/05/2024 |

En la Figura 5.28: Planificación inicial. Diagrama de Gantt, fase de pruebas, se muestra el diagrama de Gantt de la planificación inicial de la fase de pruebas del sistema.

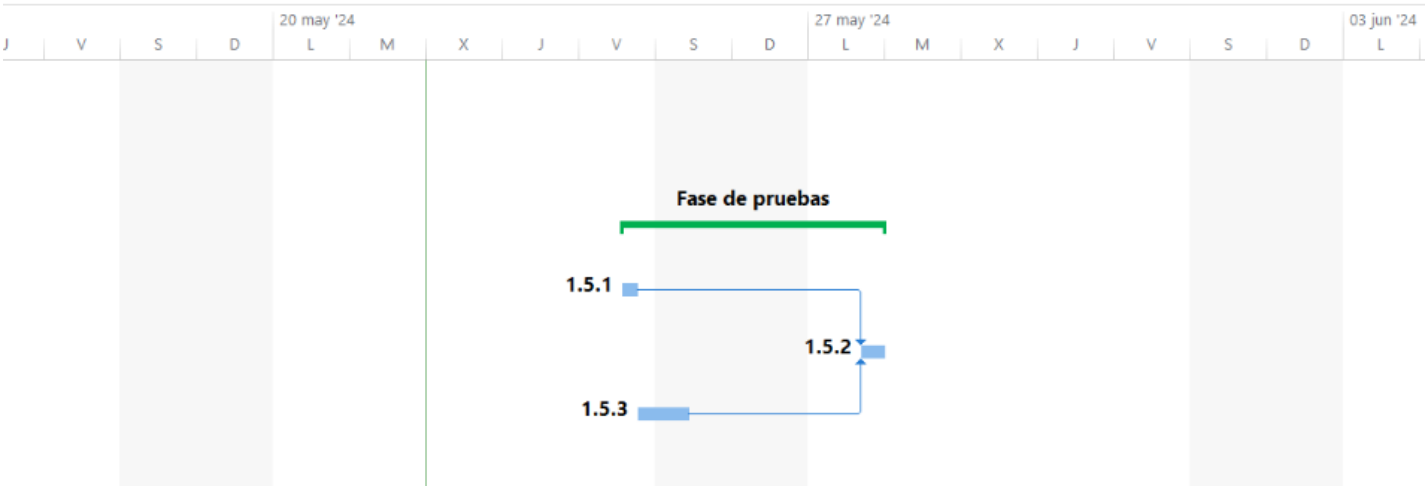


Figura 5.28: Planificación inicial. Diagrama de Gantt, fase de pruebas

#### *5.1.3.2.7 Planificación inicial. Despliegue del sistema*

En la 5.15: Planificación inicial. Despliegue del sistema, se detalla la planificación de las tareas que se deben realizar en la fase de despliegue del sistema. El total de días estimados para la realización de esta fase es de 8 horas, desde el 12/06/2024 hasta el 13/06/2024.

Tabla 5.15: Planificación inicial. Despliegue del sistema

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.6 | Despliegue del sistema | 8 horas | 12/06/2024 | 13/06/2024 |
| 1.6.1 | Configuración del servidor de Azure | 4 horas | 12/06/2024 | 12/06/2024 |
| 1.6.2 | Despliegue del *frontend* | 4 horas | 12/06/2024 | 13/06/2024 |

En la Figura 5.29: Planificación inicial. Diagrama de Gantt, despliegue del sistema, se muestra el diagrama de Gantt de la planificación inicial de la fase de despliegue del sistema.


Figura 5.29: Planificación inicial. Diagrama de Gantt, despliegue del sistema

#### *5.1.3.2.8 Planificación inicial. Documentación del proyecto*

En la 5.16: Planificación inicial. Documentación del proyecto, se detalla la planificación de las tareas que se deben realizar en la fase de documentación del proyecto. Esta fase se ha simplificado en el presente documento, de tal manera que para la tarea de la documentación de la memoria solo se especifica las tareas principales, sin detallar las subtareas.

Tabla 5.16: Planificación inicial. Documentación del proyecto

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.7 | Documentación del proyecto | 69.25 días | 02/04/2024 | 18/06/2024 |
| 1.7.1 | Creación de la plantilla en LaTeX | 4 horas | 01/06/2024 | 01/06/2024 |
| 1.7.2 | Redacción del documento final | 68.25 días | 02/04/2024 | 17/06/2024 |
| 1.7.2.1 | Introducción | 1 hora | 17/06/2024 | 17/06/2024 |
| 1.7.2.3 | Planificación del sistema de información | 9.38 días | 12/04/2024 | 22/04/2024 |
| 1.7.2.4 | Definición de la arquitectura tecnológica | 16.75 días | 20/05/2024 | 08/06/2024 |
| 1.7.2.5 | Estudio de viabilidad del sistema | 2.13 días | 18/05/2024 | 20/05/2024 |
| 1.7.2.6 | Planificación y gestión del TFG | 44.69 días | 26/04/2024 | 15/06/2024 |
| 1.7.2.7 | Análisis del sistema de información | 58.69 días | 02/04/2024 | 07/06/2024 |

Planificación inicial. Documentación del proyecto – Continúa en la siguiente página...

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.7.2.8 | Diseño del sistema de información | 11.13 días | 31/05/2024 | 12/06/2024 |
| 1.7.2.9 | Construcción del sistema de información | 12.88 días | 18/05/2024 | 01/06/2024 |
| 1.7.2.10 | Conclusiones y ampliaciones | 2 horas | 15/06/2024 | 15/06/2024 |
| 1.7.3 | Anexos | 20.81 días | 25/05/2024 | 17/06/2024 |
| 1.7.4 | Presentación proyecto | 4 horas | 18/06/2024 | 18/06/2024 |

En la Figura 5.30: Planificación inicial. Diagrama de Gantt, documentación del proyecto, se muestra el diagrama de Gantt de la planificación inicial de la fase de documentación del proyecto.

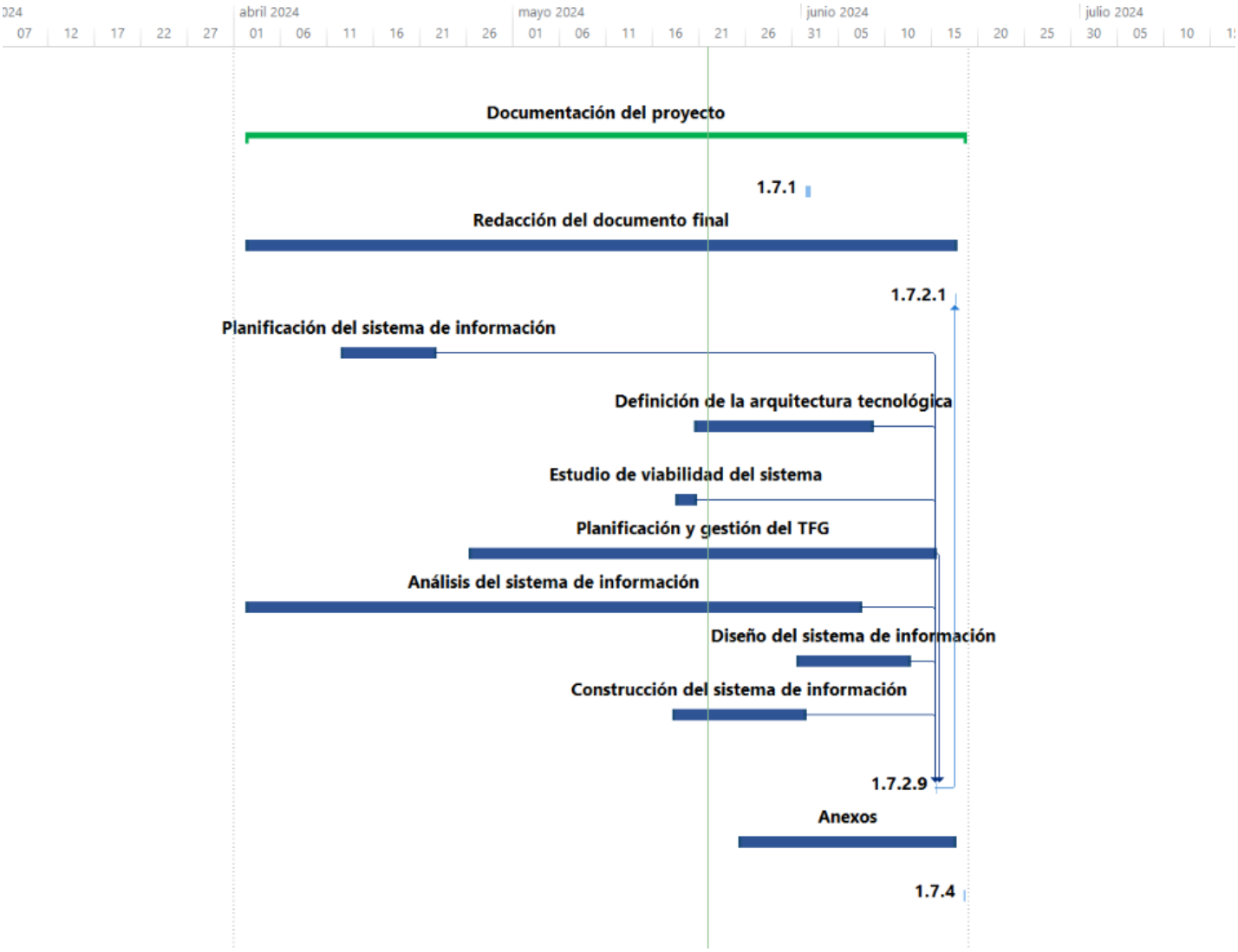


Figura 5.30: Planificación inicial. Diagrama de Gantt, documentación del proyecto

### 5.1.4. Riesgos

En esta sección, se detallarán los riesgos identificados para el proyecto y cómo serán abordados. Según el PMBOK:

> Un riesgo es un evento o condición incierta que, si sucede, tiene un efecto positivo o negativo en por lo menos uno de los objetivos del proyecto, tales como el alcance, el cronograma, el coste y la calidad. [9]

#### 5.1.4.1. Plan de Gestión de Riesgos

El proyecto sigue una estrategia proactiva frente al riesgo, es decir, se han evaluado los riesgos inherentes al proyecto antes de que estos se produzcan con el objetivo de prevenirlos. El procedimiento realizado para identificar y evaluar los riesgos se detalla en el Plan de gestión de riesgos.

#### 5.1.4.2. Identificación de Riesgos

Se han identificado un total de 12 riesgos que afectan al proyecto en distintos ámbitos. Estos riesgos, ordenados de mayor a menor prioridad, son los siguientes:

1. Falta de comunicación con el tutor del TFG.
2. Seguridad de la información.
3. Intento de fraude por parte de los usuarios finales.
4. Errores en las estimaciones de tareas.
5. Conciliación entre las responsabilidades académicas y laborales.
6. Problemas con la Aceptación del Cliente.
7. Problemas de cumplimiento normativo y legal.
8. Cambios de licencia en el software.
9. Problemas de rendimiento.
10. Fallos en el hardware o software.
11. Problemas de usabilidad.
12. Problemas de accesibilidad.

#### 5.1.4.3. Registro de Riesgos

Se ha llevado a cabo un análisis de los riesgos identificados en el proyecto, detallando la probabilidad de ocurrencia, el impacto en el proyecto, la prioridad y la estrategia de respuesta a cada riesgo. Este análisis se puede consultar en el anexo Registro de riesgos.

### 5.1.5. Presupuesto

En esta sección se detallan los costes asociados al proyecto, incluyendo los costes de personal, materiales, licencias, infraestructura y otros gastos necesarios para la realización del proyecto. Además, se incluye un presupuesto de cliente, que detalla los costes que el cliente deberá asumir para la realización del proyecto.

#### 5.1.5.1. Definición de la empresa

En esta sección se presenta la definición del modelo empresarial que ejecutará el proyecto. Se especificarán los costos asociados al personal que integra la empresa, así como los costos de los materiales, licencias, infraestructura y otros gastos necesarios para llevar a cabo las diversas actividades empresariales. Adicionalmente, se proporcionará un análisis de la facturación anual y los beneficios obtenidos por la empresa.

#### 5.1.5.1.1 *Personal de la empresa y sueldos*

La empresa cuenta con un total de 6 personas, en el apartado 5.1.2.1 OBS se detalla la estructura organizativa de la empresa. A continuación, se especifica el sueldo bruto anual de cada uno de los perfiles de la empresa, así como el coste total anual que supone el personal de la empresa.

Tabla 5.17: Costes salariales del proyecto

| Personal | Número | Sueldo Bruto Anual | Coste Salarial Anual | TOTAL |
|---|---|---|---|---|
| Jefe de proyecto | 1 | 30.000,00€ | 38.400,00€ | 38.400,00€ |
| Analista junior | 1 | 23.000,00€ | 29.440,00€ | 29.440,00€ |
| Diseñador UX/UI junior | 1 | 21.000,00€ | 26.880,00€ | 26.880,00€ |
| Desarrollador Software junior | 1 | 22.000,00€ | 28.160,00€ | 28.160,00€ |
| Tester junior | 1 | 21.000,00€ | 26.880,00€ | 26.880,00€ |
| Documentador técnico | 1 | 24.500,00€ | 31.360,00€ | 31.360,00€ |
| **TOTAL** | **6** | | | **181.120,00€** |

#### 5.1.5.1.2 *Costes debidos a trabajos no productivos*

En la tabla 5.18 Costes debidos a trabajos no productivos se detalla la asignación de los costes salariales a costes directos e indirectos. Esta asignación se debe a que no todo el tiempo de trabajo de los empleados se dedica a tareas productivas, también se dedica tiempo a tareas no productivas, como reuniones, formación, etc.

La productividad se define como el porcentaje de tiempo que se dedica a tareas productivas y se calcula mediante la fórmula:

$$\text{Productividad (R)} = \frac{\text{Cantidad de producto obtenido}}{\text{Cantidad de recurso utilizado (R)}}$$

De esta manera, se pueden distinguir claramente los costes directos, asociados al tiempo efectivamente empleado en la producción, y los costes indirectos, correspondientes al tiempo dedicado a actividades no productivas. En la tabla identificamos coste directo como CD y coste indirecto como CI.

Tabla 5.18: Asignación de los costes salariales a costes directos e indirectos

| Personal | TOTAL | Prod. (%) | CD | CI (%) | CI |
|---|---|---|---|---|---|
| Jefe de proyecto | 38.400,00€ | 0 % | 0,00€ | 100 % | 38.400,00€ |
| Analista junior | 29.440,00€ | 85 % | 25.024,00€ | 15 % | 4.416,00€ |

Continúa en la siguiente página...

| Personal | TOTAL | Prod. ( %) | CD | CI ( %) | CI |
|---|---|---|---|---|---|
| Diseñador UX/UI junior | 26.880,00€ | 85 % | 22.848,00€ | 15 % | 4.032,00€ |
| Desarrollador Software junior | 28.160,00€ | 85 % | 23.936,00€ | 15 % | 4.224,00€ |
| Tester junior | 26.880,00€ | 85 % | 22.848,00€ | 15 % | 4.032,00€ |
| Documentador técnico | 31.360,00€ | 90 % | 28.224,00€ | 10 % | 3.136,00€ |
| **TOTAL** | **181.120,00€** | | **122.880,00€** | | **58.240,00€** |

Calculando las horas productivas por perfil se obtiene la tabla 5.19 Costes debidos a trabajos no productivos, donde se obtiene el número de horas productivas por perfil y el total de horas productivas de la empresa.

Tabla 5.19: Número de horas productivas por perfil y en total

| Personal | Nº | Prod. ( %) | Horas / año | Horas productivas / año (Por persona) | Horas productivas (Total empresa) |
|---|---|---|---|---|---|
| Jefe de proyecto | 1 | 0 % | 2032 | 0 | 0 |
| Analista junior | 1 | 80 % | 2032 | 1625,6 | 1625,6 |
| Diseñador UX/UI junior | 1 | 85 % | 2032 | 1727,2 | 1727,2 |
| Desarrollador Software junior | 1 | 85 % | 2032 | 1727,2 | 1727,2 |
| Tester junior | 1 | 80 % | 2032 | 1625,6 | 1625,6 |
| Documentador técnico | 1 | 90 % | 2032 | 1828,8 | 1828,8 |
| **TOTAL** | **6** | | | | **8534,4** |

#### *5.1.5.1.3 Costes de servicios*

Es necesario calcular los costes de los servicios inherentes a la empresa, estos costes se detallan en la tabla 5.20 Costes de servicios. Estos costes son aquellos que no están directamente relacionados con la producción, pero que son necesarios para el funcionamiento de la empresa.

Tabla 5.20: Costes de servicios

| Servicio | Coste mes | Coste anual |
|---|---|---|
| Alquiler de inmueble | 800,00€ | 9.600,00€ |

Continúa en la siguiente página...

| Servicio | Coste mes | Coste anual |
|---|---|---|
| Tributos y tasas diversas | 650,00€ | 7.800,00€ |
| Limpieza | 500,00€ | 6.000,00€ |
| Consumos de agua | 80,00€ | 960,00€ |
| Consumos de electricidad (excepto consumos para producción) | 130,00€ | 1.560,00€ |
| Honorarios de asesorías, auditorías y otros profesionales | 150,00€ | 1.800,00€ |
| Primas de seguros | 600,00€ | 7.200,00€ |
| Gastos en material de oficina | 50,00€ | 600,00€ |
| Gastos financieros | 200,00€ | 2.400,00€ |
| **TOTAL** | **3.160,00€** | **37.920,00€** |

#### *5.1.5.1.4 Costes de los medios de producción*

Por otro lado, es necesario calcular los costes de los medios de producción, estos costes se detallan en la tabla 5.21 Costes de los medios de producción. Estos costes son aquellos que están directamente relacionados con la producción, como los costes de los equipos informáticos, licencias de software, etc.

Estos costes se han clasificado en dos tipos: costes de amortización y costes de alquiler. Los costes de amortización son aquellos que se pagan de una sola vez y se amortizan a lo largo de varios años, mientras que los costes de alquiler son aquellos que se pagan de forma periódica. Para los costes de amortización se ha calculado el coste anual de amortización y el plazo de amortización en años.

En la tabla identificamos coste anual como CA y coste total como CT

Tabla 5.21: Costes de los medios de producción

| Equipo / Licencia | Características | ud. | Precio | CT | CA | Tipo | Plazo |
|---|---|---|---|---|---|---|---|
| Portátiles | MacBook Pro 16 pulgadas Chip M3 Max de Apple con CPU de 14 núcleos, GPU de 30 núcleos y Neural Engine de 16 núcleos, 36GB memoria unificada, 1TB de almacenamiento SSD | 1 | 4.299,00€ | 4.299,00€ | 1.074,75€ | Amortizac. | 4 |

Continúa en la siguiente página...

| Equipo / Licencia | Características | ud. | Precio | CT | CA | Tipo | Plazo |
|---|---|---|---|---|---|---|---|
| Portátiles | Portátil HP Pavilion Plus 16-ab0004ns, Windows 11 Home, Intel® Core™ i7 13700H (13.ª generación), 16 GB RAM, 1 TB SSD, 16 pulgadas, WQXGA (2560 x 1600), 120 Hz, NVIDIA® GeForce RTX™ 3050 (6 GB) | 5 | 1.499,00€ | 7.495,00€ | 1.873,75€ | Amortizac. | 4 |
| Licencia Overleaf | Licencia *Group Standard* para 10 usuarios | 1 | 1.160,00€ | 1.160,00€ | 1.160,00€ | Alquiler | - |
| Licencia Excalidraw | Licencia Anual | 2 | 72,00€ | 144,00€ | 144,00€ | Alquiler | - |
| Licencia Office 365 | Microsoft 365 Plan Empresa Premium | 6 | 247,20€ | 1.483,20€ | 1.483,20€ | Alquiler | - |
| Licencia MS Project | Compra de pago único Project Profesional 2021 | 1 | 1.659,00€ | 1.659,00€ | 331,80€ | Amortizac. | 5 |
| Conexión a internet | Fibra 600 MB con centralita virtual | 1 | 1.013,76€ | 1.013,76€ | 1.013,76€ | Alquiler | - |
| **TOTAL** | | | | **7.081,26€** | | | |

#### *5.1.5.1.5 Precio por hora de trabajo y facturación total de la empresa*

En la tabla 5.22 Precio por hora de trabajo y facturación total de la empresa se detalla el precio por hora de trabajo de cada uno de los perfiles de la empresa, así como el total de horas productivas de la empresa y la facturación total de la empresa.

Este precio por hora permite cubrir los costes salariales, los costes de servicios y los costes de los medios de producción, así como obtener un beneficio para la empresa.

La facturación total de la empresa se calcula multiplicando el precio por hora de trabajo de cada perfil por el número de horas productivas de cada perfil y sumando el total de horas productivas de la empresa, con lo que se obtiene que la empresa facturará un total de 230.530,40€ al año.

Tabla 5.22: Precios por hora de trabajo y facturación total de la empresa

| Personal | Precio/hora | Horas productivas (Total empresa) | Facturación |
|---|---|---|---|
| Jefe de proyecto | 40,00€ | 0,00 | 0,00€ |
| Analista junior | 30,00€ | 1625,60 | 48.768,00€ |
| Diseñador UX/UI junior | 26,00€ | 1727,20 | 44.907,20€ |
| Desarrollador Software junior | 25,00€ | 1727,20 | 43.180,00€ |
| Tester junior | 25,00€ | 1625,60 | 40.640,00€ |
| Documentador técnico | 29,00€ | 1828,80 | 53.035,20€ |
| **TOTAL** | | **8534,40** | **230.530,40€** |

#### *5.1.5.1.6 Precio por hora a usar para el cálculo de los proyectos de costes*

En la tabla 5.23 Precio por hora a usar para el cálculo de los proyectos de costes se detalla el precio por hora a usar para el cálculo de los proyectos de costes. Este es el precio real que la empresa tiene que pagar por hora de trabajo de cada uno de los perfiles de la empresa.

Tabla 5.23: Precio por hora a usar para el cálculo de los proyectos de costes

| Personal | Precio/hora |
|---|---|
| Jefe de proyecto | 18,90€ |
| Analista junior | 14,49€ |
| Diseñador UX/UI junior | 13,23€ |
| Desarrollador Software junior | 13,86€ |
| Tester junior | 13,23€ |
| Documentador técnico | 15,43€ |

#### *5.1.5.1.7 Resumen del modelo de empresa*

Tras los cálculos realizados, se obtiene el resumen del modelo de empresa, que se detalla en la tabla 5.24 Resumen del modelo de empresa. En esta tabla se detalla el total de los costes directos, el total de los costes indirectos, la suma de los costes directos e indirectos, el beneficio deseado, el coste total, la facturación posible en función de las horas de producción y de los precios por hora calculados, y el margen entre el coste total y la facturación.

La empresa obtiene un margen del 2,034 % entre el coste total y la facturación, lo que indica que la empresa obtiene un beneficio del 2,034 % sobre el coste total.

Tabla 5.24: Resumen del modelo de empresa

| Nº | Concepto | Importe |
|---|---|---|
| 1 | Total de los costes directos | 122.880,00€ |
| 2 | Total de los costes indirectos | 65.321,26€ |
| 3 | Suma de los costes directos e indirectos | 188.201,26€ |
| 4 | Beneficio deseado (20 %) | 37.640,25€ |
| 5 | Coste total (suma de los costes directos, indirectos y beneficios) | 225.841,51€ |
| 6 | Facturación posible en función de las horas de producción y de los precios por hora calculados | 230.530,40€ |
| 7 | Margen entre el coste total y la facturación (relación entre 5 y 6) | 2,034 % |

#### 5.1.5.2. Presupuesto de Costes

En esta sección se presentará el presupuesto de costes. Es necesario tener en cuenta que el presupuesto de costes es una estimación de los recursos necesarios para llevar a cabo el proyecto. El presupuesto de costes se divide en diferentes partidas, una por cada fase identificada en 5.1.3.2: Planificación inicial. Cada partida se desglosa en diferentes ítems, que representan las tareas y recursos necesarios para llevar a cabo la fase correspondiente. Los datos de los ítems se han obtenido a partir de la planificación inicial del proyecto y de la definición del modelo de empresa del apartado 5.1.5.1: Definición de la empresa.

En primer lugar, se presentará un resumen de la cuantía total necesaria para llevar a cabo el proyecto, ver Tabla 5.25: Presupuesto de Costes , y a continuación se detallarán los presupuestos de cada una de las partidas que lo componen.

La cuantía total necesaria para llevar a cabo el proyecto asciende a **6.738,36€**.

Tabla 5.25: Resumen presupuesto de costes del proyecto

| Nº | Nombre de la partida | Total |
|---|---|---|
| 1 | Análisis del sistema | 173,54€ |
| 2 | Seguimiento del proyecto | 478,27€ |
| 3 | Diseño del sistema | 1.023,75€ |
| 4 | Implementación del sistema | 2.158,17€ |
| 5 | Fase de pruebas | 136,85€ |
| 6 | Despliegue del sistema | 144,16€ |
| 7 | Documentación del proyecto | 2.623,62€ |

Continúa en la siguiente página...

| Nº | Nombre de la partida | Total |
|---|---|---|
| | **TOTAL** | **6.738,36€** |

#### *5.1.5.2.1 Presupuesto Inicial. Partida 1: Análisis del proyecto*

En esta partida se detallan los costes asociados a la fase de análisis del proyecto. Esta partida alcanza un total de **173,54€**.

Tabla 5.26: Presupuesto Inicial. Partida 1: Análisis del proyecto

| I1 | I2 | Descripción | Cant. | ud. | Precio | Subtotal (2) | Total |
|---|---|---|---|---|---|---|---|
| 1 | | Análisis del sistema | | | | | 72,44€ |
| | 1 | Analista junior | 5 | horas | 14,49€ | 72,44€ | |
| 2 | | Análisis de la Arquitectura | | | | | 43,46€ |
| | 1 | Analista junior | 3 | horas | 14,49€ | 43,46€ | |
| 3 | | Análisis de la Infraestructura | | | | | 28,98€ |
| | 1 | Analista junior | 2 | horas | 14,49€ | 28,98€ | |
| 4 | | Determinación del alcance de desarrollo | | | | | 28,66€ |
| | 1 | Desarrollador Software junior | 0,5 | horas | 13,86€ | 6,93€ | |
| | 2 | Analista junior | 1,5 | horas | 14,49€ | 21,73€ | |
| | | | | | | **TOTAL:** | **173,54€** |

#### *5.1.5.2.2 Presupuesto Inicial. Partida 2: Seguimiento del sistema*

En esta partida se detallan los costes asociados al seguimiento del sistema. Se estima para esta partida un total de **478,27€**.

Tabla 5.27: Presupuesto Inicial. Partida 2: Seguimiento del sistema

| I1 | I2 | Descripción | Cant. | ud. | Precio | Subtotal (2) | Total |
|---|---|---|---|---|---|---|---|
| 1 | | Reunión de arranque | | | | | 30,39€ |
| | 1 | Documentador técnico | 1,5 | horas | 15,43€ | 23,15€ | |
| | 2 | Analista junior | 0,5 | horas | 14,49€ | 7,24€ | |
| 2 | | Reuniones periódicas | | | | | 299,84€ |
| | 1 | Documentador técnico | 10 | horas | 15,43€ | 154,33€ | |

Presupuesto Inicial. Partida 2: Seguimiento del sistema – Continúa en la siguiente página...

| I1 | I2 | Descripción | Cant. | ud. | Precio | Subtotal | Total |
|---|---|---|---|---|---|---|---|
| | 2 | Desarrollador Software junior | 10,5 | horas | 13,86€ | 145,51€ | |
| 4 | | Reunión de revisión | | | | | 89,45€ |
| | 1 | Documentador técnico | 2 | horas | 15,43€ | 30,87€ | |
| | 2 | Desarrollador Software junior | 2 | horas | 13,86€ | 27,72€ | |
| 4 | | Reunión final | | | | | 58,58€ |
| | 1 | Documentador técnico | 2 | horas | 15,43€ | 30,87€ | |
| | 2 | Desarrollador Software junior | 2 | horas | 13,86€ | 27,72€ | |
| | | | | | | **TOTAL:** | **478,27€** |

#### *5.1.5.2.3 Presupuesto Inicial. Partida 3: Diseño del sistema*

Esta partida detalla los costes asociados al diseño del sistema, alcanzando un total de **1.023,75€**.

Tabla 5.28: Planificación final. Diseño del sistema

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | Diseño del backend | | | | | | | 304,88€ |
| | 1 | | | Diseño del módulo de usuarios | | | | | | 304,88€ | |
| | | 1 | | Desarrollador Software junior | 2 | horas | 13,86€ | | | 27,72€ | |
| | 2 | | | Diseño del módulo de cartas | | | | | | | |
| | | 1 | | Desarrollador Software junior | 6 | horas | 13,86€ | | | 83,15€ | |
| | 3 | | | Diseño del módulo de sobres de cartas | | | | | | | |
| | | 1 | | Desarrollador Software junior | 3 | horas | 13,86€ | | | 41,57€ | |
| | 4 | | | Diseño del módulo de subastas | | | | | | | |
| | | 1 | | Desarrollador Software junior | 6.5 | horas | 13,86€ | | | 90,08€ | |
| | 5 | | | Diseño del módulo de transacciones | | | | | | | |
| | | 1 | | Desarrollador Software junior | 4.5 | horas | 13,86€ | | | 62,36€ | |
| 2 | | | | Diseño del frontend | | | | | | | 479,81€ |
| | 1 | | | Diseño de elementos gráficos | | | | | | | 39,69€ |
| | | 1 | | Diseño de la moneda de la aplicación | 1 | horas | 13,23€ | 13,23€ | | | |
| | | 2 | | Diseño del logo de la aplicación | 2 | horas | 13,23€ | 26,46€ | | | |
| | 2 | | | Diseño de la temática | | | | | 26,46€ | | |
| | | 1 | | Diseñador UX/UI junior | 2 | horas | 13,23€ | | | 26,46€ | |
| | 3 | | | Diseño del árbol de navegación | | | | | 26,46€ | | |
| | | 1 | | Diseñador UX/UI junior | 2 | horas | 13,23€ | | | 26,46€ | |

Planificación final. Diseño del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 4 | | | Diseño de la interfaz de usuario | | | | | | 387,21€ | |
| | | 1 | | Diseño de las páginas de información | | | | | 40,00€ | | |
| | | | 1 | Diseñador UX/UI junior | 2.5 | horas | 13,23€ | 33,07€ | | | |
| | | | 2 | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | | 2 | | Diseño de la página Home | | | | | 53,23€ | | |
| | | | 1 | Diseñador UX/UI junior | 3.5 | horas | 13,23€ | 46,30€ | | | |
| | | | 2 | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | | 3 | | Diseño de la página de error | | | | | 13,35€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.8 | horas | 13,23€ | 10,58€ | | | |
| | | | 2 | Desarrollador Software junior | 0.2 | horas | 13,86€ | 2,77€ | | | |
| | | 4 | | Diseño del módulo de usuarios | | | | | 73,39€ | | |
| | | | 1 | Diseñador UX/UI junior | 4.5 | horas | 13,23€ | 59,53€ | | | |
| | | | 2 | Desarrollador Software junior | 1 | horas | 13,86€ | 13,86€ | | | |
| | | 5 | | Diseño del módulo de cartas | | | | | 53,54€ | | |
| | | | 1 | Diseñador UX/UI junior | 3 | horas | 13,23€ | 39,69€ | | | |
| | | | 2 | Desarrollador Software junior | 1 | horas | 13,86€ | 13,86€ | | | |
| | | 6 | | Diseño del módulo de sobres de cartas | | | | | 26,77€ | | |
| | | | 1 | Diseñador UX/UI junior | 1.5 | horas | 13,23€ | 19,84€ | | | |
| | | | 2 | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | | 7 | | Diseño del módulo de subastas | | | | | 73,39€ | | |
| | | | 1 | Diseñador UX/UI junior | 4.5 | horas | 13,23€ | 59,53€ | | | |
| | | | 2 | Desarrollador Software junior | 1 | horas | 13,86€ | 13,86€ | | | |

Planificación final. Diseño del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 8 | | Diseño del módulo de transacciones | | | | | 53,54€ | | |
| | | | 1 | Diseñador UX/UI junior | 3 | horas | 13,23€ | 39,69€ | | | |
| | | | 2 | Desarrollador Software junior | 1 | horas | 13,86€ | 13,86€ | | | |
| 3 | | | | Diseño de pruebas | | | | | | | 239,06€ |
| | 1 | | | Diseño de pruebas unitarias | | | | | | 86,30€ | |
| | | 1 | | Tester junior | 6 | horas | 13,23€ | 79,37€ | | | |
| | | 2 | | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | 2 | | | Diseño de las pruebas de carga/estrés | | | | | | 66,46€ | |
| | | 1 | | Tester junior | 4.5 | horas | 13,23€ | 59,53€ | | | |
| | | 2 | | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | 3 | | | Diseño de las pruebas end-to-end | | | | | | 86,30€ | |
| | | 1 | | Tester junior | 6 | horas | 13,23€ | 79,37€ | | | |
| | | 2 | | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | | | | | | | | | | **TOTAL:** | **1.023,75€** |

#### *5.1.5.2.4 Presupuesto Inicial. Partida 4: Implementación del sistema*

En esta partida se detallan los costes asociados a la implementación del sistema, se estima un total de **2.158,17€**.

Tabla 5.29: Planificación final. Diseño del sistema

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | Implementación del backend | | | | | | | 921,57€ |
| | 1 | | | Implementación del módulo de usuarios | | | | | | 921,57€ | |
| | | 1 | | Desarrollador Software junior | 5.5 | horas | 13,86€ | | | 76,22€ | |
| | 2 | | | Implementación del módulo de cartas | | | | | | | |
| | | 1 | | Desarrollador Software junior | 20.5 | horas | 13,86€ | | | 284,09€ | |
| | 3 | | | Implementación del módulo de sobres de cartas | | | | | | | |
| | | 1 | | Desarrollador Software junior | 7.5 | horas | 13,86€ | | | 103,94€ | |
| | 4 | | | Implementación del módulo de subastas | | | | | | | |
| | | 1 | | Desarrollador Software junior | 17 | horas | 13,86€ | | | 235,59€ | |
| | 5 | | | Implementación del módulo de transacciones | | | | | | | |
| | | 1 | | Desarrollador Software junior | 16 | horas | 13,86€ | | | 221,73€ | |
| 2 | | | | Implementación del frontend | | | | | | | 898,33€ |
| | 1 | | | Implementación de la temática | | | | | | | 13,80€ |
| | | 1 | | Diseñador UX/UI junior | 0.1 | horas | 13,23€ | 1,32€ | | | |
| | | | | Desarrollador Software junior | 0.9 | horas | 13,86€ | 12,47€ | | | |
| | 2 | | | Implementación de las rutas de navegación | | | | | | | 13,80€ |
| | | 1 | | Diseñador UX/UI junior | 0.1 | horas | 13,23€ | 1,32€ | | | |
| | | | | Desarrollador Software junior | 0.9 | horas | 13,86€ | 12,47€ | | | |

Planificación final. Diseño del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 3 | | | Implementación de la interfaz de usuario | | | | | | 870,74€ | |
| | | 1 | | Implementación de las páginas de información | | | | | 41,26€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 2.5 | horas | 13,86€ | 34,65€ | | | |
| | | 2 | | Implementación de la página Home | | | | | 124,41€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 8.5 | horas | 13,86€ | 117,80€ | | | |
| | | 3 | | Implementación de la página de error | | | | | 13,73€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.2 | horas | 13,23€ | 2,65€ | | | |
| | | | 2 | Desarrollador Software junior | 0.8 | horas | 13,86€ | 11,09€ | | | |
| | | 4 | | Implementación del módulo de usuarios | | | | | 117,48€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 8 | horas | 13,86€ | 110,87€ | | | |
| | | 5 | | Implementación del módulo de cartas | | | | | 145,20€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 10 | horas | 13,86€ | 138,58€ | | | |
| | | 7 | | Implementación del módulo de sobres de cartas | | | | | 103,62€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 7 | horas | 13,86€ | 97,01€ | | | |
| | | 8 | | Implementación del módulo de subastas | | | | | 235,28€ | | |

Planificación final. Diseño del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 16.5 | horas | 13,86€ | 228,66€ | | | |
| | | 9 | | Implementación del módulo de transacciones | | | | | 89,76€ | | |
| | | | 1 | Diseñador UX/UI junior | 0.5 | horas | 13,23€ | 6,61€ | | | |
| | | | 2 | Desarrollador Software junior | 6 | horas | 13,86€ | 83,15€ | | | |
| 3 | | | | Implementación de pruebas | | | | | | | 338,27€ |
| | 1 | | | Implementación de las pruebas unitarias | | | | | | 178,90€ | |
| | | 1 | | Tester junior | 13 | horas | 13,23€ | 171,97€ | | | |
| | | 2 | | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | 2 | | | Implementación de las pruebas de carga/estrés | | | | | | 73,07€ | |
| | | 1 | | Tester junior | 5 | horas | 13,23€ | 66,14€ | | | |
| | | 2 | | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | 3 | | | Implementación de las pruebas end-to-end | | | | | | 86,30€ | |
| | | 1 | | Tester junior | 6 | horas | 13,23€ | 79,37€ | | | |
| | | 2 | | Desarrollador Software junior | 0.5 | horas | 13,86€ | 6,93€ | | | |
| | | | | | | | | | | **TOTAL:** | **2.158,17€** |

#### 5.1.5.2.5 *Presupuesto Inicial. Partida 5: Fase de pruebas*

En esta partida se detallan los costes asociados a la fase de pruebas, alcanzando un total de **136,85€**.

Tabla 5.30: Presupuesto Inicial. Partida 5: Fase de pruebas

| I1 | I2 | Descripción | Cant. | ud. | Precio | Subtotal (2) | Total |
|---|---|---|---|---|---|---|---|
| 1 | | Ejecución de pruebas unitarias | | | | | 45,83€ |
| | 1 | Tester junior | 2,5 | horas | 15,43€ | 38,58€ | |
| | 2 | Desarrollador Software junior | 0,5 | horas | 14,49€ | 7,24€ | |
| 2 | | Ejecución de pruebas carga/estrés | | | | | 45,51€ |
| | 1 | Tester junior | 2,5 | horas | 15,43€ | 38,58€ | |
| | 2 | Desarrollador Software junior | 0,5 | horas | 13,86€ | 6,93€ | |
| 3 | | Ejecución de pruebas end-to-end | | | | | 45,51€ |
| | 1 | Tester junior | 2,5 | horas | 15,43€ | 38,58€ | |
| | 2 | Desarrollador Software junior | 0,5 | horas | 13,86€ | 6,93€ | |
| | | | | | | **TOTAL:** | **136,85€** |

Tabla 5.31: Presupuesto Inicial. Partida 6: Fase de despliegue

| I1 | I2 | Descripción | Cant. | ud. | Precio | Subtotal (2) | Total |
|---|---|---|---|---|---|---|---|
| 1 | | Configuración del servidor de Azure | | | | | 88,73€ |
| | 1 | Analista junior | 0,5 | horas | 15,43€ | 7,72€ | |
| | 2 | Desarrollador Software junior | 3,5 | horas | 14,49€ | 50,71€ | |
| | 3 | Máquina virtual de Azure. 1 B2s (2 Cores, 4 GB RAM) x 1 Month (Pay as you go), Linux, (Pay as you go); 1 managed disk – E3, LRS - 6 GB; Inter Region transfer type, 5 GB outbound data transfer from Este de EE. UU. to Este de Asia | 1 | mes | 30,30€ | 30,30€ | |

Presupuesto Inicial. Partida 6: Fase de despliegue – Continúa en la siguiente página...

| I1 | I2 | Descripción | Cant. | ud. | Precio | Subtotal | Total |
|---|---|---|---|---|---|---|---|
| 2 | | Despliegue del sistema | | | | | 55,43€ |
| | 1 | Desarrollador Software junior | 4 | horas | 13,86€ | 55,43€ | |
| | | | | | | **TOTAL:** | **144,16€** |

#### *5.1.5.2.6 Presupuesto Inicial. Partida 7: Documentación del sistema*

En último lugar, se detallan los costes asociados a la documentación del sistema, alcanzando un total de **2.438,42€**. Este coste es el más elevado de todos los presupuestos iniciales debido a la gran cantidad de horas de trabajo dedicadas a la redacción del documento final. Esta labor no solo implica la escritura, sino también el análisis exhaustivo y detallado necesario para la elaboración de la documentación técnica, que es crucial para asegurar la precisión y la integridad del sistema descrito. La minuciosidad en este proceso garantiza que todos los aspectos técnicos sean comprendidos y documentados adecuadamente, facilitando futuras referencias y mantenimientos del sistema.

Tabla 5.32: Planificación final. Documentación del sistema

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | | Creación de la plantilla en LaTeX | | | | | | | 61,73€ |
| | 1 | | | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| 2 | | | | Redacción del documento final | | | | | | | 2.376,69€ |
| | 1 | | | Introducción | | | | | | 15,43€ | |
| | | 1 | | Documentador técnico | 1 | horas | 15,43€ | | | | 15,43€ |
| | 2 | | | Planificación del sistema de información | | | | | | 277,80€ | |
| | | 1 | | Inicio del plan del sistema de información | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| | | 2 | | Definición y organización | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| | | 3 | | Estudio de información relevante | | | | | 216,06€ | | |
| | | | 1 | Documentador técnico | 14 | horas | 15,43€ | | | | 216,06€ |
| 3 | | | | Definición de la arquitectura tecnológica | | | | | | | 77,17€ |
| | 1 | | | Identificación de necesidades | | | | | 46,30€ | | |
| | | | 1 | Documentador técnico | 3 | horas | 15,43€ | | | | 46,30€ |
| | 2 | | | Selección de la arquitectura | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| 4 | | | | Estudio de viabilidad del sistema | | | | | | | 123,46€ |
| | 1 | | | Análisis de sistemas similares | | | | | 61,73€ | | |

Planificación final. Documentación del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| | 2 | | | Análisis de tecnologías | | | | | 61,73€ | | |
| | | | 1 | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| 5 | | | | Planificación y gestión del TFG | | | | | | | 802,52€ |
| | 1 | | | Planificación del proyecto | | | | | 524,72€ | | |
| | | | 1 | Documentador técnico | 34 | horas | 15,43€ | | | | 524,72€ |
| | 2 | | | Ejecución del proyecto | | | | | 92,60€ | | |
| | | | 1 | Documentador técnico | 6 | horas | 15,43€ | | | | 92,60€ |
| | 3 | | | Cierre del proyecto | | | | | 185,20€ | | |
| | | | 1 | Documentador técnico | 12 | horas | 15,43€ | | | | 185,20€ |
| 6 | | | | Análisis del sistema de información | | | | | | | 447,56€ |
| | 1 | | | Definición del sistema | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| | 2 | | | Establecimiento de requisitos | | | | | 61,73€ | | |
| | | | 1 | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| | 3 | | | Identificación de subsistemas de análisis | | | | | 61,73€ | | |
| | | | 1 | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| | 4 | | | Análisis de casos de uso | | | | | 100,31€ | | |
| | | | 1 | Documentador técnico | 6.5 | horas | 15,43€ | | | | 100,31€ |
| | 5 | | | Análisis de clases | | | | | 61,73€ | | |
| | | | 1 | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| | 6 | | | Definición de interfaces de usuario | | | | | 100,31€ | | |
| | | | 1 | Documentador técnico | 6.5 | horas | 15,43€ | | | | 100,31€ |

Planificación final. Documentación del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 7 | | | Especificación de pruebas | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| 7 | | | | Diseño del sistema de información | | | | | | | 378,11€ |
| | 1 | | | Diseño de casos de uso reales | | | | | 100,31€ | | |
| | | | 1 | Documentador técnico | 6.5 | horas | 15,43€ | | | | 100,31€ |
| | 2 | | | Diseño de clases | | | | | 84,88€ | | |
| | | | 1 | Documentador técnico | 5.5 | horas | 15,43€ | | | | 84,88€ |
| | 3 | | | Diseño de la arquitectura de módulos del sistema | | | | | 84,88€ | | |
| | | | 1 | Documentador técnico | 5.5 | horas | 15,43€ | | | | 84,88€ |
| | 4 | | | Diseño físico de datos | | | | | 46,30€ | | |
| | | | | Documentador técnico | 3 | horas | 15,43€ | | | | 46,30€ |
| | 5 | | | Diseño de la migración y carga inicial de datos | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| | 6 | | | Especificación técnica del plan de pruebas | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| 8 | | | | Construcción del sistema de información | | | | | | | 223,78€ |
| | 1 | | | Preparación del entorno de generación y construcción | | | | | 61,73€ | | |
| | | | 1 | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| | 2 | | | Generación del código de los componentes | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |

Planificación final. Documentación del sistema – Continúa en la siguiente página...

| I1 | I2 | I3 | I4 | Descripción | Cant. | ud. | Precio | Subt. 4 | Subt. 3 | Subt. 2 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 3 | | | Ejecución de las pruebas unitarias | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| | 4 | | | Ejecución de las pruebas de integración | | | | | 38,58€ | | |
| | | | 1 | Documentador técnico | 2.5 | horas | 15,43€ | | | | 38,58€ |
| | 5 | | | Ejecución de las pruebas del sistema | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| | 6 | | | Elaboración de manuales de usuario | | | | | 30,87€ | | |
| | | | 1 | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| 9 | | | | Conclusiones y ampliaciones | | | | | | | 30,87€ |
| | 1 | | | Documentador técnico | 2 | horas | 15,43€ | | | | 30,87€ |
| 3 | | | | Anexos | | | | | | | 123,46€ |
| | 1 | | | Documentador técnico | 8 | horas | 15,43€ | | | | 123,46€ |
| 4 | | | | Presentación | | | | | | | 61,73€ |
| | 1 | | | Documentador técnico | 4 | horas | 15,43€ | | | | 61,73€ |
| | | | | | | | | | | **TOTAL:** | **2.623,62€** |

#### *5.1.5.2.7 Análisis de reservas*

Se ha llevado a cabo un análisis de reservas para mitigar los riesgos asociados a la planificación inicial. Se ha establecido una única reserva global de dos semanas para cubrir posibles retrasos en la ejecución del proyecto. Por otro lado, se ha establecido una reserva de gestión del proyecto en la que se almacena un 1 % del presupuesto total para cubrir posibles sobrecostes y se asigna una reserva de hasta un 10 % del coste del proyecto.

También se ha establecido una reserva de control de riesgos para cubrir posibles sobrecostes asociados a los riesgos identificados en la planificación inicial, que asciende a 1.000€.

El coste semanal del proyecto es de 673,84€, por lo que la reserva global asciende a 774,91€ calculándose en función de la probabilidad de que ocurra un retraso en la ejecución del proyecto. Se puede observar en la tabla table:reserva-globalTabla 5.33: Análisis de reservas.

Tabla 5.33: Única reserva global de 2 semanas

| Retraso | Probabilidad | Coste |
|---|---|---|
| Retraso de 1 semana | 65 % | 437,99 € |
| Retraso de 2 semanas | 50 % | 336,92 € |
| **TOTAL** | | **774,91 €** |

Por otro lado, la reserva de un 1 % del presupuesto total asciende a 67,38€, como se puede observar en la tabla table:reserva-sobrecostesTabla 5.34: Análisis de reservas.

Tabla 5.34: Reservas de gestión

| Reserva | % | TOTAL |
|---|---|---|
| Aportación: fondo de reservas de gestión | 1 % | 67,38 € |
| Reserva de gestión asignada al proyecto | 10 % | 673,84 € |

Esto hace que las reservas asignadas al proyecto asciendan a **1.842,30€**.

#### *5.1.5.2.8 Porcentaje a aplicar al cliente*

El porcentaje a aplicar al cliente se determina en función de los costes totales del proyecto, las reservas asignadas y el porcentaje de beneficio deseado. En este caso, el porcentaje a aplicar es del **47,25 %**, como se puede observar en la Tabla 5.35: Porcentaje a aplicar al cliente.

Es importante señalar que este porcentaje se aplicará únicamente a los precios por hora de los recursos humanos. Los costes de adquisición de hardware y software no se verán afectados por este incremento. Un ejemplo de ello es el coste de la máquina virtual de Azure, que asciende a 30,30 € y se mantendrá igual tanto para la propia empresa como para el cliente en el presupuesto.

Tabla 5.35: Cálculo de porcentaje a incrementar

| Concepto | TOTAL |
|---|---|
| Reserva | 1.842,30 € |
| Beneficio (20 %) | 1.341,61 € |
| Total a compensar | 3.183,91 € |
| **Porcentaje a aplicar** | **47,250 %** |

#### 5.1.5.3. Presupuesto de Cliente

El presupuesto a presentar al cliente se detalla en la tabla Tabla 5.36: Presupuesto de Cliente , donde se desglosan los costes asociados a cada una de las tareas de primer nivel del proyecto. El total del presupuesto asciende a **11.937,30€**, incluyendo un IVA del 21 %.

Tabla 5.36: Presupuesto inicial del cliente

| I1 | I2 | Descripción | Subtotal | Total |
|---|---|---|---|---|
| 1 | | Análisis del sistema | **Total** | 255,54€ |
| | 1 | Análisis del sistema | 106,67€ | |
| | 2 | Análisis de la Arquitectura | 64,00€ | |
| | 3 | Análisis de la infraestructura | 42,67€ | |
| | 4 | Determinación del alcance de desarrollo | 42,20€ | |
| 2 | | Seguimiento del proyecto | | 704,25€ |
| | 1 | Reunión de arranque | 44,75€ | |
| | 2 | Reuniones periódicas | 441,52€ | |
| | 3 | Reunión de revisión | 131,71€ | |
| | 4 | Reunión final | 86,26€ | |
| 3 | | Diseño del sistema | | 1.507,47€ |
| | 1 | Diseño del backend | 448,94€ | |
| | 2 | Diseño del frontend | 706,52€ | |
| | 3 | Diseño de pruebas | 352,01€ | |
| 4 | | Implementación del sistema | | 3.177,92€ |
| | 1 | Implementación del backend | 1.357,02€ | |
| | 2 | Implementación del frontend | 1.322,80€ | |

Continúa en la siguiente página...

| I1 | I2 | Descripción | Subtotal | Total |
|---|---|---|---|---|
| | 3 | Implementación de pruebas | 498,10€ | |
| 5 | | Fase de pruebas | | 201,51€ |
| | 1 | Ejecución de pruebas unitarias | 67,48€ | |
| | 2 | Ejecución de pruebas carga/estrés | 67,02€ | |
| | 3 | Ejecución de pruebas end-to-end | 67,02€ | |
| 6 | | Despliegue del sistema | | 156,03€ |
| | 1 | Configuración del servidor Azure | 44,11€ | |
| | 2 | Despliegue del sistema | 81,63€ | |
| | 3 | Máquina virtual de Azure. 1 B2s (2 Cores, 4 GB RAM) x 1 Month (Pay as you go), Linux, (Pay as you go); 1 managed disk – E3, LRS - 6 GB; Inter Region transfer type, 5 GB outbound data transfer from Este de EE. UU. to Este de Asia | 30,30€ | |
| 7 | | Documentación del proyecto | | 3.863,30€ |
| | 1 | Creación de la plantilla en LaTeX | 90,90€ | |
| | 2 | Redacción del documento final | 3.499,69€ | |
| | 3 | Anexos | 181,80€ | |
| | 4 | Presentación del proyecto | 90,90€ | |
| | | | Total sin IVA | 9.866,03€ |
| | | | IVA (21 %) | 2.071,27€ |
| | | | **TOTAL CON IVA** | **11.937,30€** |

## 5.2. EJECUCIÓN DEL PROYECTO

### 5.2.1. Plan Seguimiento de Planificación

El criterio de seguimiento de la planificación se basará en la comparación de las tareas planificadas con las tareas realizadas, para ello se usará la planificación inicial como referencia. Se llevará a cabo un seguimiento semanal de las tareas realizadas, ayudándose de herramientas como Trello Trello y GitHub.

Si se detecta que una tarea no se ha completado en el tiempo estimado, se analizarán las causas y se reevaluará la planificación para ajustar los plazos y recursos necesarios. Estas desviaciones se registrarán en la bitácora de incidencias del proyecto.

En el caso de que se detecten desviaciones significativas en la planificación, se informará al jefe del proyecto y se tomarán las medidas necesarias para corregir la situación.

### 5.2.2. Bitácora de Incidencias del Proyecto

A lo largo del proyecto se han registrado las siguientes incidencias:

Tabla 5.37: Bitácora de incidencias del proyecto

| Descripción | Solución | Efecto en la planificación |
|---|---|---|
| **Falta de análisis del sistema antes de realizar la planificación** . | Reunión de equipo para acordar el alcance del sistema y revisión de la planificación | Aumento del tiempo dedicado a la realización de la planificación |
| **Desviación en la estimación de tiempo de diseño del modelo de datos** | Reunión de equipo para reevaluar la estrategia de diseño | Aumento del tiempo dedicado al diseño del sistema |
| **Problemas de compatibilidad horaria del equipo de desarrollo con su trabajo actual** | Reducción del calendario del recurso | Aumento de los plazos de entrega |
| **Nuevos requisitos no contemplados en la planificación inicial** | Corrección de la planificación y actualización de los requisitos y alcance del sistema. | Aumento de los plazos de entrega |
| **Desviación en la estimación de tiempo de despliegue de la plataforma** | Configurar el sistema para que pueda comunicarse mediante el protocolo HTTPS | Aumento del tiempo dedicado al despliegue de la plataforma |

Continúa en la siguiente página...

Tabla 5.37 Bitácora de incidencias del proyecto – continuación de la página anterior

| Descripción | Solución | Efecto en la planificación |
|---|---|---|
| **Rehacer documento final de la memoria** | Se han adaptado los apartados del documento acorde a la plataforma que se está desarrollando | Aumento del tiempo dedicado a la redacción de la memoria |

### 5.2.3. Riesgos

De los riesgos identificados en la planificación inicial, Registro de riesgos, se han producido los siguientes:

- **Falta de comunicación con el tutor del TFG:** Se ha producido una falta de comunicación con el tutor del TFG, lo que ha llevado a una desviación en la planificación. Esta falta de comunicación ha implicado rehacer el modelado de datos y la planificación inicial. Se ha solucionado mediante más reuniones con el tutor del TFG y reevaluando la estrategia de diseño.
- **Conciliación entre responsabilidades académicas y laborales:** Se ha producido una desviación en la planificación debido a problemas de compatibilidad horaria del equipo de desarrollo con su trabajo actual. Se ha solucionado reduciendo el calendario del recurso y aumentando los plazos de entrega.
- **Errores en las estimaciones de tareas:** Se ha producido una desviación en la planificación debido a una desviación en la estimación de tiempo de despliegue de la plataforma y al diseño del modelo de datos. Se ha solucionado reajustando el cronograma.

## 5.3. CIERRE DEL PROYECTO

### 5.3.1. Planificación Final

Debido a las incidencias y riesgos que se han producido a lo largo del proyecto, se ha tenido que reajustar la planificación inicial. Se han agregado tareas adicionales, eliminado tareas que no se correspondían con la realidad y reorganizado las tareas existentes. Esto ha provocado cambios en las fechas de entrega de las tareas y en la duración de las mismas.

A continucación, se muestra la planificación final del proyecto, con las tareas reajustadas y las fechas de entrega actualizadas.

Tabla 5.38: Planificación final. Visión general

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1 | Proyecto BidMon Universe | 434.5 horas | 01/04/2024 | 04/07/2024 |
| 1.1 | Análisis del proyecto | 15 horas | 01/04/2024 | 04/04/2024 |
| 1.2 | Seguimiento del proyecto | 23 horas | 01/04/2024 | 01/07/2024 |
| 1.3 | Diseño del sistema | 81 horas | 04/04/2024 | 22/04/2024 |
| 1.4 | Implementación del sistema | 153,5 horas | 22/04/2024 | 15/06/2024 |
| 1.5 | Fase de pruebas | 11 horas | 17/06/2024 | 22/06/2024 |
| 1.6 | Despliegue del sistema | 11 horas | 25/06/2024 | 04/07/2024 |
| 1.7 | Documentación del proyecto | 138 horas | 21/05/2024 | 04/07/2024 |

#### *5.3.1.0.1 Planificación final. Análisis del proyecto*

En la 5.39: Planificación final. Análisis del proyecto, se detallan la planificación de las tareas que se han realizado en la fase de análisis del proyecto.

Tabla 5.39: Planificación final. Análisis del proyecto

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.1 | Análisis del proyecto | 15 horas | 01/04/2024 | 04/04/2024 |
| 1.1.1 | Análisis del sistema | 8 horas | 01/04/2024 | 03/04/2024 |
| 1.1.2 | Análisis de la arquitectura | 3 horas | 03/04/2024 | 04/04/2024 |
| 1.1.3 | Análisis de la infraestructura | 2 horas | 03/04/2024 | 04/04/2024 |
| 1.1.4 | Determinación del análisis | 2 horas | 03/04/2024 | 04/04/2024 |

#### *5.3.1.0.2 Planificación final. Seguimiento del proyecto*

En la 5.40: Planificación final. Seguimiento del proyecto, se detalla la planificación de las tareas llevadas a cabo en la fase de seguimiento del proyecto.

Tabla 5.40: Planificación final. Seguimiento del proyecto

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.2 | Seguimiento del proyecto | 23 horas | 01/04/2024 | 01/07/2024 |
| 1.2.1 | Reunión de arranque | 2 horas | 01/04/2024 | 01/04/2024 |
| 1.2.2 | Reuniones periódicas | 17 horas | 23/05/2024 | 27/07/2024 |
| 1.2.3 | Reunión final | 4 horas | 01/07/2024 | 01/07/2024 |

#### *5.3.1.0.3 Planificación final. Diseño del sistema*

En la 5.41: Planificación final. Diseño del sistema, se muestran las tareas que se han realizado en la fase de diseño del sistema. La principal diferencia con la planificación inicial es que se ha añadido el módulo de notificaciones.

Tabla 5.41: Planificación final. Diseño del sistema

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.3 | Diseño del sistema | 81 horas | 04/04/2024 | 22/04/2024 |
| 1.3.1 | Diseño del *backend* | 38 horas | 04/04/2024 | 22/04/2024 |
| 1.3.1.1 | Diseño del módulo de usuarios | 2 horas | 04/04/2024 | 05/04/2024 |
| 1.3.1.2 | Diseño del módulo de cartas | 10 horas | 05/04/2024 | 06/04/2024 |
| 1.3.1.3 | Diseño del módulo de sobres de cartas | 8 horas | 06/04/2024 | 09/04/2024 |
| 1.3.1.4 | Diseño del módulo de subastas | 9 horas | 09/04/2024 | 11/04/2024 |
| 1.3.1.5 | Diseño del módulo de transacciones | 6 horas | 12/04/2024 | 13/04/2024 |
| 1.3.1.6 | Diseño del módulo de notificaciones | 3 horas | 12/04/2024 | 13/04/2024 |
| 1.3.2 | Diseño del *frontend* | 39 horas | 13/04/2024 | 20/04/2024 |
| 1.3.2.1 | Diseño de elementos gráficos | 3 horas | 13/04/2024 | 13/04/2024 |

Planificación final. Diseño del sistema – Continúa en la siguiente página...

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.3.2.1.1 | Diseño de la moneda de la aplicación | 1 hora | 13/04/2024 | 13/04/2024 |
| 1.3.2.1.2 | Diseño del logo de la aplicación | 2 horas | 13/04/2024 | 13/04/2024 |
| 1.3.2.2 | Diseño de la temática | 4 horas | 13/04/2024 | 13/04/2024 |
| 1.3.2.3 | Diseño del árbol de navegación | 2 horas | 13/04/2024 | 13/04/2024 |
| 1.3.2.4 | Diseño de la interfaz de usuario | 30 horas | 15/04/2024 | 20/04/2024 |
| 1.3.2.4.1 | Diseño de las páginas de información | 3 horas | 15/04/2024 | 16/04/2024 |
| 1.3.2.4.2 | Diseño de la página Home | 4 horas | 15/04/2024 | 16/04/2024 |
| 1.3.2.4.3 | Diseño de la página de error | 1 hora | 16/04/2024 | 16/04/2024 |
| 1.3.2.4.4 | Diseño del módulo de usuarios | 5 horas | 16/04/2024 | 17/04/2024 |
| 1.3.2.4.5 | Diseño del módulo de cartas | 4 horas | 17/04/2024 | 18/04/2024 |
| 1.3.2.4.6 | Diseño del módulo de sobres de cartas | 2 horas | 18/04/2024 | 19/04/2024 |
| 1.3.2.4.7 | Diseño del módulo de subastas | 5 horas | 19/04/2024 | 20/04/2024 |
| 1.3.2.4.8 | Diseño del módulo de transacciones | 4 horas | 20/04/2024 | 20/04/2024 |
| 1.3.2.4.9 | Diseño del módulo de notificaciones | 2 horas | 20/04/2024 | 20/04/2024 |
| 1.3.3 | Diseño de pruebas | 4 g¡horas | 21/04/2024 | 22/04/2024 |
| 1.3.3.1 | Diseño de pruebas unitarias | 2 horas | 21/04/2024 | 21/04/2024 |
| 1.3.3.2 | Diseño de pruebas *end-to-end* | 2 horas | 22/04/2024 | 22/04/2024 |

#### *5.3.1.0.4 Planificación inicial. Implementación del sistema*

En la 5.42: Planificación inicial. Implementación del sistema, se detalla la planificación de las tareas realizadas en la fase de implementación del sistema.

Tabla 5.42: Planificación final. Implementación del sistema

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.4 | Implementación del sistema | 153,5 horas | 22/04/2024 | 15/06/2024 |
| 1.4.1 | Implementación del *backend* | 73.5 horas | 22/04/2024 | 08/05/2024 |
| 1.4.1.1 | Implementación del módulo de usuarios | 5.5 horas | 22/04/2024 | 23/04/2024 |
| 1.4.1.2 | Implementación del módulo de cartas | 20.5 horas | 23/04/2024 | 26/04/2024 |
| 1.4.1.3 | Implementación del módulo de sobres de cartas | 7.5 horas | 27/04/2024 | 28/04/2024 |
| 1.4.1.4 | Implementación del módulo de subastas | 17 horas | 29/04/2024 | 02/05/2024 |
| 1.4.1.5 | Implementación del módulo de transacciones | 16 horas | 03/05/2024 | 05/05/2024 |
| 1.4.1.6 | Implementación del módulo de notificaciones | 3 horas | 07/05/2024 | 07/05/2024 |
| 1.4.1.7 | Implementación del módulo de PayPal | 4 horas | 07/05/2024 | 08/05/2024 |
| 1.4.2 | Implementación del *frontend* | 66 horas | 07/05/2024 | 13/06/2024 |
| 1.4.2.1 | Implementación de la temática | 1 hora | 07/05/2024 | 07/05/2024 |
| 1.4.2.2 | Implementación de las rutas de navegación | 1 hora | 12/06/2024 | 12/06/2024 |
| 1.4.2.3 | Implementación de la interfaz | 64 horas | 08/05/2024 | 12/06/2024 |
| 1.4.2.3.1 | Implementación de las páginas de información | 3 horas | 08/05/2024 | 09/05/2024 |
| 1.4.2.3.2 | Implementación de la página Home | 3 horas | 09/05/2024 | 10/05/2024 |
| 1.4.2.3.3 | Implementación de la página de error | 1 hora | 10/05/2024 | 10/05/2024 |
| 1.4.2.3.4 | Implementación del módulo de usuarios | 7 horas | 10/05/2024 | 11/05/2024 |

Planificación final. Implementación del sistema – Continúa en la siguiente página...

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.4.2.3.5 | Implementación del módulo de cartas | 12 horas | 11/05/2024 | 14/05/2024 |
| 1.4.2.3.6 | Implementación del módulo de sobres de cartas | 7 horas | 14/05/2024 | 15/05/2024 |
| 1.4.2.3.7 | Implementación del módulo de subastas | 15 horas | 16/05/2024 | 18/05/2024 |
| 1.4.2.3.8 | Implementación del módulo de transacciones | 7 horas | 18/05/2024 | 20/05/2024 |
| 1.4.2.3.9 | Implementación del módulo de notificaciones | 3 horas | 20/05/2024 | 21/05/2024 |
| 1.4.2.3.10 | Implementación del módulo de PayPal | 6 horas | 11/06/2024 | 12/06/2024 |
| 1.4.3 | Implementación de pruebas | 14 horas | 13/06/2024 | 15/06/2024 |
| 1.4.3.1 | Implementación de pruebas unitarias | 8 horas | 13/06/2024 | 14/06/2024 |
| 1.4.3.2 | Implementación de pruebas *end-to-end* | 6 horas | 15/06/2024 | 15/06/2024 |

#### *5.3.1.0.5 Planificación final. Fase de pruebas*

En la 5.43: Planificación final. Fase de pruebas, se detalla la planificación de las tareas que se han realizado en la fase de pruebas del sistema. Respecto a la planificación inicial, se han añadido tareas de pruebas de accesibilidad y adaptabilidad.

Tabla 5.43: Planificación final. Fase de pruebas

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.5 | Fase de pruebas | 13 horas | 17/06/2024 | 22/06/2024 |
| 1.5.1 | Pruebas unitarias | 3 horas | 17/06/2024 | 17/06/2024 |
| 1.5.2 | Pruebas de accesibilidad | 5 horas | 20/06/2024 | 22/06/2024 |
| 1.5.3 | Pruebas de adaptabilidad | 2 horas | 22/06/2024 | 22/06/2024 |
| 1.5.4 | Pruebas *end-to-end* | 3 horas | 19/06/2024 | 19/06/2024 |

#### *5.3.1.0.6 Planificación final. Despliegue del sistema*

En la 5.44: Planificación final. Despliegue del sistema, se especifican las tareas que se han realizado en la fase de despliegue del sistema.

Tabla 5.44: Planificación final. Despliegue del sistema

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.6 | Despliegue del sistema | 11 horas | 25/06/2024 | 04/07/2024 |
| 1.6.1 | Configuración del servidor de Azure | 6 horas | 25/06/2024 | 26/06/2024 |
| 1.6.2 | Despliegue del *frontend* | 5 horas | 03/07/2024 | 04/07/2024 |

#### *5.3.1.0.7 Planificación final. Documentación del proyecto*

En la 5.45: Planificación final. Documentación del proyecto, se detalla la planificación de las tareas que se han llevado a cabo en la fase de documentación del proyecto. Se han añadido algunas tareas y eliminado otras debido a cambios en la estructura final del documento de la memoria.

Tabla 5.45: Planificación final. Documentación del proyecto

| EDT | Nombre tarea | Duración | Fecha inicio | Fecha fin |
|---|---|---|---|---|
| 1.7 | Documentación del proyecto | 138 horas | 21/05/2024 | 04/07/2024 |
| 1.7.1 | Creación de la plantilla en LaTeX | 8 horas | 21/05/2024 | 23/05/2024 |
| 1.7.2 | Redacción del documento final | 120 horas | 21/05/2024 | 04/07/2024 |
| 1.7.2.1 | Introducción | 1 hora | 30/06/2024 | 04/07/2024 |
| 1.7.2.2 | Planificación del sistema de información | 4 horas | 23/05/2024 | 27/05/2024 |
| 1.7.2.3 | Definición de la arquitectura tecnológica | 2 horas | 28/05/2024 | 29/05/2024 |
| 1.7.2.4 | Estudio de viabilidad del sistema | 18 horas | 04/06/2024 | 08/06/2024 |
| 1.7.2.5 | Planificación y gestión del TFG | 51 horas | 31/05/2024 | 29/06/2024 |
| 1.7.2.6 | Análisis del sistema de información | 32 horas | 28/05/2024 | 20/06/2024 |
| 1.7.2.7 | Construcción del sistema de información | 10 horas | 22/06/2024 | 27/06/2024 |
| 1.7.2.8 | Conclusiones y ampliaciones | 2 horas | 15/06/2024 | 15/06/2024 |
| 1.7.3 | Anexos | 10 horas | 27/05/2024 | 04/07/2024 |

### 5.3.2. Informe Final de Riesgos

Finalmente, se ha realizado un análisis exhaustivo de los riesgos identificados en el cierre del proyecto. Se han reevaluado los riesgos iniciales, identificando aquellos que se han materializado y aquellos que no.

Los riesgos que se han materializado son los siguientes:

- **Conciliación entre responsabilidades académicas y laborales:** Este riesgo ha persistido hasta el fin del proyecto, afectando tanto a la planificación como a los plazos de entrega.
- **Errores en las estimaciones de tareas:** Este riesgo ha seguido impactando la planificación del proyecto, provocando desviaciones en los plazos de entrega.
- **Problemas de rendimiento:** Este riesgo se ha materializado durante la fase de pruebas, afectando al rendimiento de la plataforma. Este riesgo podría haberse mitigado con una planificación más detallada y una mayor dedicación a la optimización del rendimiento, o eliminado optando por una versión de pago de la base de datos.

En cuanto a los riesgos identificados en la fase de seguimiento, se ha subsanado el riesgo **Falta de comunicación con el tutor del TFG**, manteniendo una comunicación más fluida y constante con el tutor del proyecto en el tramo final del mismo, lo que ha facilitado el cierre exitoso del proyecto.

### 5.3.3. Presupuesto Final de Costes

En base a la planificación final del proyecto, se ha realizado un análisis de los costes asociados al proyecto. El modelo de empresa se corresponde con el definido en la sección 5.1.5.1 Definición de la empresa.

El presupuesto final de costes del proyecto se detalla en la tabla 5.46: Presupuesto Final de Costes.

Tabla 5.46: Presupuesto final de costes del proyecto.

| Nª | Nombre de la partida | Total |
|---|---|---|
| 1 | Análisis del sistema | 217,01 € |
| 2 | Seguimiento del proyecto | 337,95 € |
| 3 | Diseño del sistema | 1.073,51 € |
| 4 | Implementación del sistema | 2.154,08 € |
| 5 | Fase de pruebas | 197,01 € |
| 6 | Despliegue del sistema | 186,99 € |
| 7 | Documentación del proyecto | 2.129,76 € |
| | **TOTAL** | **6.296,24 €** |

### 5.3.4. Presupuesto Final de Cliente

Debido a los cambios sufridos en la planificación del proyecto, ha sido necesario ajustar el presupuesto inicial del cliente para poder analizar las desviaciones producidas en el presupuesto final.

En la tabla 5.47 se detallan los costes asociados a cada una de las tareas de primer nivel del proyecto. El total del presupuesto asciende a **11.279,53€**, incluyendo un IVA del 21 %.

Tabla 5.47: Presupuesto del cliente

| I1 | I2 | Descripción | Subtotal | Total |
|---|---|---|---|---|
| 1 | | Análisis del sistema | **Total** | 321,96€ |
| | 1 | Análisis del sistema | 171,96€ | |
| | 2 | Análisis de la Arquitectura | 64,48€ | |
| | 3 | Análisis de la infraestructura | 42,99€ | |
| | 4 | Determinación del alcance de desarrollo | 42,52€ | |
| 2 | | Seguimiento del proyecto | | 501,14€ |
| | 1 | Reunión de arranque | 45,07€ | |
| | 2 | Reuniones periódicas | 369,20€ | |
| | 4 | Reunión final | 86,87€ | |
| 3 | | Diseño del sistema | | 1.591,87€ |
| | 1 | Diseño del backend | 780,90€ | |
| | 2 | Diseño del frontend | 731,58€ | |
| | 3 | Diseño de pruebas | 79,40€ | |
| 4 | | Implementación del sistema | | 3.194,20€ |
| | 1 | Implementación del backend | 1.510,42€ | |
| | 2 | Implementación del frontend | 1.290,53€ | |
| | 3 | Implementación de pruebas | 393,25€ | |
| 5 | | Fase de pruebas | | 292,14€ |
| | 1 | Ejecución de pruebas unitarias | 67,95€ | |
| | 2 | Ejecución de pruebas de accesibilidad | 112,09€ | |
| | 3 | Ejecución de pruebas end-to-end | 67,49€ | |
| | 4 | Ejecución de pruebas de adaptabilidad | 44,60€ | |
| 6 | | Despliegue del sistema | | 262,65€ |
| | 1 | Configuración del servidor Azure | 129,60€ | |

Continúa en la siguiente página...

| I1 | I2 | Descripción | Subtotal | Total |
|---|---|---|---|---|
| | 2 | Despliegue del sistema | 102,75€ | |
| | 3 | Máquina virtual de Azure. 1 B2s (2 Cores, 4 GB RAM) x 1 Month (Pay as you go), Linux, (Pay as you go); 1 managed disk – E3, LRS - 6 GB; Inter Region transfer type, 5 GB outbound data transfer from Este de EE. UU. to Este de Asia | 30,30€ | |
| 7 | | Documentación del proyecto | | 3.158,14€ |
| | 1 | Creación de la plantilla en LaTeX | 183,08€ | |
| | 2 | Redacción del documento final | 2.747,21€ | |
| | 3 | Anexos | 228,85€ | |
| | | | TOTAL SIN IVA | 9.321,93€ |
| | | | IVA (21 %) | 1.957,60€ |
| | | | **TOTAL CON IVA** | **11.279,53€** |

### 5.3.5. Análisis desviaciones de presupuesto

En este apartado se analizan las desviaciones producidas en el presupuesto del proyecto y del cliente. Una cifra negativa indica que el coste ha sido menor al presupuestado, mientras que una cifra positiva indica que el coste ha sido mayor al presupuestado.

Empezaremos analizando las desviaciones del presupuesto de costes del proyecto, detalladas en la tabla Tabla 5.48: Análisis desviaciones de presupuesto.

Tabla 5.48: Comparación de desviaciones del presupuesto de costes del proyecto

| Nº | Nombre de la partida | P. Costes Inicial | P. Costes Final | Desviación |
|---|---|---|---|---|
| 1 | Análisis del sistema | 173,54€ | 217,01€ | 43,47€ |
| 2 | Seguimiento del proyecto | 478,27€ | 337,95€ | -140,32€ |
| 3 | Diseño del sistema | 1.023,75€ | 1.073,51€ | 49,76€ |
| 4 | Implementación del sistema | 2.158,17€ | 2.154,08€ | -4,09€ |

Continúa en la siguiente página...

| Nº | Nombre de la partida | P. Costes Inicial | P. Costes Final | Desviación |
|---|---|---|---|---|
| 5 | Fase de pruebas | 136,85€ | 197,01 | 60,16€ |
| 6 | Despliegue del sistema | 144,16€ | 186,99€ | 42,83€ |
| 7 | Documentación del proyecto | 2.623,62€ | 2.129,76€ | -493,86€ |
| | **TOTAL** | **6.738,36€** | **6.296,32€** | **-442,04€** |

El proyecto ha resultado en una desviación de 442,04€ menos de lo presupuestado inicialmente. La mayor desviación se ha producido en la documentación del proyecto, con una desviación de 493,86€ menos de lo presupuestado. Esta desviación es positiva, ya que se ha conseguido realizar el proyecto sin sobrepasar el presupuesto inicial.

Por otro lado, en el presupuesto del cliente se pueden observar las desviaciones detalladas en la tabla table:comparacion-desviaciones-clienteTabla 5.49: Análisis desviaciones de presupuesto.

Tabla 5.49: Comparación de desviaciones del presupuesto del cliente

| I1 | Descripción | P. Inicial Cliente | P. Final Cliente | Desviación |
|---|---|---|---|---|
| 1 | Análisis del sistema | 255,54€ | 321,79€ | 66,25€ |
| 2 | Seguimiento del proyecto | 704,25€ | 501,14€ | -203,11€ |
| 3 | Diseño del sistema | 1.507,47€ | 1.591,87€ | 84,40€ |
| 4 | Implementación del sistema | 3.177,92€ | 3.194,20€ | 16,28€ |
| 5 | Fase de pruebas | 201,51€ | 292,14€ | 90,63€ |
| 6 | Despliegue del sistema | 156,03€ | 262,65€ | 106,62€ |
| 7 | Documentación del proyecto | 3.863,30€ | 3.158,14€ | -705,16€ |
| | **TOTAL** | **9.866,03€** | **9.321,93€** | **-544,10€** |

En el presupuesto del cliente se ha producido una desviación de 544,10€ menos de lo inicialmente presupuestado. La mayor desviación se ha observado en la documentación del proyecto, con una diferencia de 705,16€ por debajo de lo previsto.

Finalmente, la empresa ha obtenido un beneficio de **3.025,61€**, inferior al calculado en el presupuesto inicial (**3.127,67€**), pero positivo.

### 5.3.6. Informe de Lecciones Aprendidas

Este apartado me ha permitido comprender la importancia de dedicar el tiempo suficiente a la planificación de las tareas. He aprendido que una planificación adecuada es fundamental para asegurar un buen progreso del proyecto.

Unas tareas bien planificadas no solo facilitan el seguimiento y control del proyecto, sino que también contribuyen significativamente a cumplir con los plazos y el presupuesto establecido. Lo que influye directamente en la satisfacción del cliente y en la calidad del producto final.

Por otro lado, la falta de una planificación adecuada desde el inicio me llevó a tener que rehacer la planificación inicial para ajustarla a la realidad del proyecto, lo que implicó emplear un tiempo que podría haberse evitado.

Esta experiencia subraya la necesidad de dedicar el esfuerzo necesario en la fase de planificación y análisis del proyecto, para asegurar que se establecen unos objetivos claros y alcanzables, y que se identifican y gestionan adecuadamente los riesgos y las expectativas de los interesados.

# Capítulo 6

# ANÁLISIS y DISEÑO DEL SISTEMA DE INFORMACIÓN

## 6.1. DEFINICIÓN DEL SISTEMA

### 6.1.1. Determinación del Alcance del Sistema

BidMon Universe tiene como objetivo principal permitir al usuario coleccionar activos digitales, en este caso, cartas. Estas se pueden obtener mediante una compra directa a la aplicación o por medio de subastas ciegas, donde las ofertas de otros participantes permanecen ocultas hasta el cierre de la subasta, añadiendo un elemento emocionante y asegurando la imparcialidad en el proceso de compra y venta.

Es esencial establecer un sistema de trazabilidad robusto que registre cada transacción de compra y venta dentro de la aplicación de manera clara e inequívoca, asegurando la integridad y la transparencia de todas las operaciones. Un sistema integral de notificaciones mantendrá a los usuarios constantemente informados sobre las actualizaciones de las subastas en las que están involucrados, ya sea como vendedores o como postores. En esta versión del sistema, la carga inicial de datos se llevará a cabo una única vez. Los administradores del sistema no podrán añadir nuevas cartas a través de la interfaz de usuario, cualquier adición de nuevas cartas se realizará directamente en la base de datos.

Además, la plataforma no incluirá una opción que permita a los usuarios exportar la colección de cartas que posean. La aplicación no incluye un sistema de mensajería interna ni permite la formación de relaciones sociales directas entre los usuarios. La plataforma se centra exclusivamente en las transacciones y la colección de cartas.

En 2.2.1 Especificación del Ámbito y Alcance se detalla en profundidad el alcance del sistema y sus limitaciones. En el apartado 2.1.2 Identificación del Alcance del Plan de Sistemas de Información se listan las funcionalidades de alto nivel que se espera que tenga la aplicación, y para una descripción más exhaustiva, ver el apartado 6.2.1 Obtención de los Requisitos del Sistema.

### 6.1.2. Identificación de Actores del Sistema

El sistema está diseñado para soportar tres categorías principales de usuarios, cada uno con niveles de acceso y funcionalidades específicas adecuadas a su rol dentro de la plataforma:

1. **Usuarios No Autenticados**. Los usuarios no autenticados son aquellos que aún no han completado el proceso de registro ni han iniciado sesión en la plataforma.
    - Tienen acceso exclusivamente a la página de bienvenida y a otras páginas informativas de acceso público dentro de la plataforma.
    - El objetivo es proporcionar a los usuarios no registrados información general sobre la plataforma y sus características, sin comprometer la seguridad o la privacidad de las transacciones y actividades de los usuarios registrados.
2. **Usuarios Autenticados**. Un usuario autenticado es aquel que ha completado el proceso de registro y ha iniciado sesión en la plataforma.
    - Tienen acceso a todas las funcionalidades operativas de la plataforma, incluyendo la colección de cartas, compra y subasta de estas.
3. **Usuarios Administradores**. Un usuario administrador es un usuario que ha iniciado sesión en la plataforma y tiene permisos especiales para administrar y supervisar las operaciones de la plataforma.

- Tienen acceso a todas las funcionalidades operativas de la plataforma, sin poder realizar compras o subastas.
- Tiene la capacidad de intervenir en una subasta que considere fraudulenta.
- Tiene acceso a todo el historial de transacciones y subastas realizadas en la plataforma.

## 6.2. ESTABLECIMIENTO DE REQUISITOS

### 6.2.1. Obtención de los Requisitos del Sistema

#### 6.2.1.1. Requisitos funcionales

##### *6.2.1.1.1 Registro e inicio de sesión*

**RGU-1.** Un usuario deberá poder registrarse en el sistema.

- **RGU-1.1.** El usuario deberá proporcionar el nombre de usuario que desea utilizar.
  - **RGU-1.1.1.** El sistema verificará que el nombre de usuario está disponible.
  - **RGU-1.1.2.** El sistema verificará que el nombre de usuario cumpla con las restricciones de formato especificadas en GU_NOMBRE_USUARIO.
- **RGU-1.2.** El usuario deberá proporcionar una contraseña.
  - **RGU-1.2.1.** El sistema asegurará que la contraseña cumpla con las políticas de seguridad especificadas en GU_CONTRASEÑA.
- **RGU-1.3.** El usuario deberá de confirmar la contraseña
  - **RGU-1.3.1.** El sistema verificará que la contraseña y su confirmación coincidan.
- **RGU-1.4.** El usuario deberá de especificar la fecha de nacimiento.
  - **RGU-1.4.1.** El sistema verificará que la fecha de nacimiento está en el formato GU_FECHA_NACIMIENTO.
  - **RGU-1.4.2.** El sistema verificará que el usuario cumple con GU_MIN_EDAD.
- **RGU-1.5.** Si alguno de los datos introducidos no cumple con las restricciones especificadas, el sistema mostrará un mensaje de error.
- **RGU-1.6.** Una vez finalizado el registro, el sistema registrará los datos en la base de datos:
  - **RGU-1.6.1.** Nombre de usuario
  - **RGU-1.6.2.** Nombre de usuario en minúsculas
  - **RGU-1.6.3.** Contraseña cifrada
  - **RGU-1.6.4.** Fecha de nacimiento
  - **RGU-1.6.5.** Fecha de creación
  - **RGU-1.6.6.** GU_ROL_DEFECTO
  - **RGU-1.6.7.** GU_IMG_PERFIL_DEFECTO
  - **RGU-1.6.8.** GU_SALDO_DEFECTO
- **RGU-1.7.** El sistema notificará al usuario que el registro ha sido completado con éxito.
- **RGU-1.8.** El usuario será redirigido a la pantalla de inicio de sesión.

**RGU-2.** Un usuario podrá iniciar sesión en el sistema.

- **RGU-2.1.** Un usuario deberá proporcionar las siguientes credenciales para iniciar sesión:
  - **RGU-2.1.1.** Nombre de usuario.
  - **RGU-2.1.2.** Contraseña.
- **RGU-2.2.** El sistema validará las credenciales introducidas.
  - **RGU-2.2.1.** El sistema comprobará que el nombre de usuario existe en la base de datos.

**RGU-2.2.2.** El sistema comprobará que la contraseña se corresponde con la registrada para dicho nombre de usuario.

**RGU-2.3.** Si las credenciales son correctas:

**RGU-2.3.1.** El sistema redirigirá al usuario a la pantalla principal.

**RGU-2.3.2.** Se generará un token de sesión con las restricciones GU_TOKEN_SESION.

**RGU-2.3.3.** El sistema establecerá una conexión Socket con el cliente para notificarle de eventos en tiempo real.

**RGU-2.4.** Si las credenciales son incorrectas:

**RGU-2.4.1.** El sistema mostrará un mensaje de error.

**RGU-2.4.2.** El usuario podrá intentar iniciar sesión de nuevo.

**RGU-3.** Un usuario autenticado podrá cerrar sesión en el sistema.

**RGU-3.1.** El sistema invalidará el token de sesión.

**RGU-3.2.** El sistema cerrará la conexión Socket con el cliente.

**RGU-3.3.** El sistema redirigirá al usuario a la pantalla de inicio.

#### *6.2.1.1.2 Gestión de usuarios autenticados*

**RUA-1.** El sistema deberá de permitir a los usuarios autenticados modificar su información personal.

**RUA-1.1.** Un usuario deberá de poder modificar su imagen de perfil.

**RUA-1.1.1.** El sistema mostrará varias imágenes predefinidas para que el usuario pueda seleccionar una.

**RUA-1.2.** Un usuario deberá de poder modificar su contraseña.

**RUA-2.** El sistema deberá de permitir a los usuarios autenticados consultar su colección de cartas. (Ver Colección de cartas)

**RUA-3.** El sistema deberá de permitir a los usuarios autenticados consultar sus transacciones. (Ver Transacciones)

**RUA-4.** El sistema deberá de permitir a los usuarios autenticados consultar sus subastas activas. (Ver Subastas y pujas)

**RUA-5.** El sistema deberá de permitir a los usuarios autenticados consultar sus pujas activas. (Ver Subastas y pujas)

**RUA-6.** El sistema deberá de permitir a los usuarios autenticados consultar su saldo de monedas.

**RUA-7.** El sistema deberá de permitir a los usuarios autenticados recargar su saldo.

**RUA-8.** El usuario podrá consultar las notificaciones recibidas.

**RUA-8.1.** Podrá marcar todas las notificaciones como leídas.

**RUA-8.2.** Podrá marcar individualmente las notificaciones como leídas.

**RUA-9.** Si el usuario está conectado en el momento en el que se le envía una notificación catalogada como importante, el sistema deberá de mostrar una notificación emergente.

**RUA-9.1.** El usuario indicará la cantidad de monedas que desea recargar.

**RUA-9.2.** El sistema verificará que la cantidad de monedas a recargar cumpla con GU_RESTRICCIONES_RECARGA.

**RUA-9.3.** La equivalencia de monedas a dinero real será GU_EQUIVALENCIA.

**RUA-9.4.** El sistema informará al usuario del monto a pagar.

**RUA-9.5.** El usuario podrá recargar su saldo mediante GU_MÉTODOS_BANCARIOS.

**RUA-9.6.** Si el usuario confirma la recarga, el sistema incrementará el saldo del usuario.

#### *6.2.1.1.3 Colección de cartas*

**RCC-1.** El sistema tendrá una colección de cartas disponibles para los usuarios.

**RCC-1.1.** Las cartas serán representaciones de pokémon.

**RCC-1.2.** Cada carta cuenta con los siguientes campos:

**RCC-1.2.1.** Un identificador único (UID), generado por la base de datos.

**RCC-1.2.2.** Un identificador de colección (IDC) en formato CC_FORMATO_IDC_CARTA, generado por el sistema.

**RCC-1.2.3.** Nombre del pokémon.

**RCC-1.2.4.** Rareza de la carta.

**RCC-1.2.5.** Fecha de lanzamiento.

**RCC-1.2.6.** Cantidad disponible.

**RCC-1.2.7.** Identificadores de las cartas vendidas.

**RCC-1.2.8.** Tipo del pokémon.

**RCC-1.2.9.** Descripción del pokémon.

**RCC-1.2.10.** Imagen del pokémon.

**RCC-1.2.11.** HP del pokémon.

**RCC-1.2.12.** Ataque del pokémon.

**RCC-1.2.13.** Defensa del pokémon.

**RCC-1.2.14.** Velocidad del pokémon.

**RCC-1.2.15.** Peso del pokémon.

**RCC-1.2.16.** Altura del pokémon.

**RCC-1.2.17.** Valor que indica si el pokémon es legendario.

**RCC-1.2.18.** Valor que indica si el pokémon es mítico.

**RCC-1.2.19.** Gimnasio al que pertenece el pokémon, si es que pertenece a alguno.

**RCC-1.2.20.** Número de áreas en las que se puede encontrar el pokémon.

**RCC-1.2.21.** Número de encuentros.

**RCC-1.2.22.** Media de posibilidad de captura.

**RCC-1.3.** La rareza de las cartas se calcula en función de:

**RCC-1.3.1.** El pokémon que representan.

**RCC-1.3.2.** Rareza del pokémon.

**RCC-1.3.3.** Media de posibilidad de captura.

**RCC-1.3.4.** Valor aleatorio.

**RCC-2.** Los usuarios autenticados podrán coleccionar cartas.

**RCC-2.1.** El usuario podrá visualizar las cartas que posee en su colección.

**RCC-2.2.** El usuario podrá consultar la información de cada carta.

**RCC-2.3.** El usuario podrá añadir cartas a su colección mediante:

**RCC-2.3.1.** La compra de sobres (ver Gestión de sobres).

**RCC-2.3.2.** La compra de cartas individuales mediante subastas (ver Gestión de subastas y pujas).

**RCC-2.4.** El usuario podrá consultar los movimientos de cartas, es decir, las transacciones de las cartas de su colección.

**RCC-2.5.** El usuario podrá subastar cartas de su colección (ver Gestión de subastas y pujas)

#### *6.2.1.1.4 Gestión de sobres*

**RS-1.** El sistema tendrá una colección de sobres de cartas disponibles para los usuarios.

**RS-1.1.** Cada sobre de cartas tendrá los siguientes campos:

**RS-1.1.1.** Un identificador único (UID), generado por la base de datos.

**RS-1.1.2.** Un identificador de colección (IDC) en formato CC_FORMATO_IDC_SOBRE, generado por el sistema.

**RS-1.1.3.** Nombre del sobre.

**RS-1.1.4.** Cantidad disponible.

**RS-1.1.5.** Precio del sobre.

**RS-1.1.6.** Número de cartas que contiene.

**RS-1.1.7.** Cantidad de cartas de cada mazo.

**RS-1.1.8.** Fecha de lanzamiento.

**RS-1.1.9.** Campo que indica si el sobre está disponible.

**RS-1.2.** El valor de cada sobre de cartas será determinado por:

**RS-1.2.1.** La cantidad de cartas que contiene.

**RS-1.2.2.** Los mazos a los que pertenecen las cartas.

**RS-2.** Los usuarios autenticados podrán visualizar los sobres disponibles para su compra.

**RS-3.** Los usuarios autenticados tendrán la posibilidad de comprar sobres de cartas.

**RS-3.1.** El sistema verificará que el usuario tenga saldo suficiente antes de permitir la compra de un sobre.

**RS-3.2.** Al confirmar la compra, el sistema deducirá el precio del sobre del saldo del usuario.

**RS-3.3.** Se decrementará la cantidad disponible de ese tipo de sobre en el inventario.

**RS-3.4.** El sistema deberá de seleccionar un mazo del que extraer las cartas en base al tipo de sobre comprado.

**RS-3.5.** El sistema generará N cartas aleatorias.

**RS-3.6.** Las cartas generadas serán añadidas a la colección del usuario, identificadas como *USERCARD*.

**RS-3.7.** Cada *USERCARD* se vinculará al tipo de carta correspondiente.

**RS-3.8.** El sistema registrará en la base de datos una nueva transacción por cada carta obtenida, con los siguientes campos:

**RS-3.8.1.** Identificador único (UID), generado por la base de datos.

**RS-3.8.2.** Identificador único (UID) del usuario que realizó la compra.

**RS-3.8.3.** Nombre de usuario.

**RS-3.8.4.** Identificador único (UID) de la carta obtenida.
**RS-3.8.5.** Identificador de colección (IDC) de la carta obtenida.
**RS-3.8.6.** Concepto de compra, en este caso, la compra de un sobre.
**RS-3.8.7.** Identificador del sobre.
**RS-3.8.8.** Fecha de compra.
**RS-3.8.9.** Precio del sobre.

**RS-3.9.** Finalmente, el sistema mostrará las cartas adquiridas *USERCARD* al usuario.

#### *6.2.1.1.5 Gestión de subastas y pujas*

**RSP-1.** El sistema permitirá a los usuarios autenticados subastar cartas de su colección.

**RSP-1.1.** Un usuario podrá seleccionar una carta de su colección para subastar.
**RSP-1.2.** El sistema verificará que la carta seleccionada no esté en una subasta activa.
**RSP-1.3.** Un usuario podrá especificar el precio mínimo al que acepta vender la carta.
**RSP-1.4.** Un usuario podrá especificar la duración de la subasta en horas.
**RSP-1.5.** El sistema registrará en la base de datos una nueva subasta con los siguientes datos:
**RSP-1.5.1.** Identificador único (UID).
**RSP-1.5.2.** Identificador único (UID) del usuario vendedor.
**RSP-1.5.3.** Identificador único (UID) de la carta subastada.
**RSP-1.5.4.** Identificador de colección (IDC) de la carta subastada.
**RSP-1.5.5.** Nombre de usuario vendedor.
**RSP-1.5.6.** Estado de la subasta 'activa'.
**RSP-1.5.7.** Precio de salida.
**RSP-1.5.8.** Fecha de inicio.
**RSP-1.5.9.** Duración de la subasta.
**RSP-1.5.10.** Fecha estimada de finalización.
**RSP-1.6.** El sistema actualizará el estado de la carta a 'en subasta'.
**RSP-1.7.** Se registrará una transacción en la base de datos con los siguientes datos:
**RSP-1.7.1.** Identificador único (UID), generado por la base de datos.
**RSP-1.7.2.** Identificador único (UID) del usuario que pone en subasta la carta.
**RSP-1.7.3.** Nombre de usuario.
**RSP-1.7.4.** Identificador único (UID) de la carta subastada.
**RSP-1.7.5.** Identificador de colección (IDC) de la carta subastada.
**RSP-1.7.6.** Identificador único (UID) de la subasta.
**RSP-1.7.7.** Concepto de la transacción, en este caso, la puesta en subasta de una carta.
**RSP-1.7.8.** Precio de salida.
**RSP-1.7.9.** Fecha.
**RSP-1.8.** El sistema mostrará la subasta activa en la lista de subastas activas.

**RSP-2.** El sistema permitirá a los usuarios autenticados visualizar las subastas activas.

**RSP-2.1.** Los usuarios podrán consultar la duración restante de la subasta.

**RSP-2.2.** Si el usuario es el propietario de la subasta activa se mostrará la opción de retirar la subasta.

- **RSP-2.2.1.** Un usuario podrá seleccionar una subasta activa para retirar.
- **RSP-2.2.2.** El sistema verificará que la subasta seleccionada pertenezca al usuario.
- **RSP-2.2.3.** El sistema actualizará el estado de la subasta a 'cancelada'.
- **RSP-2.2.4.** El sistema devolverá la carta subastada a la colección del usuario.
- **RSP-2.2.5.** El sistema finalizará las pujas activas:
  - **RSP-2.2.5.1.** El sistema actualizará el estado de las pujas a 'cancelada'.
  - **RSP-2.2.5.2.** El sistema notificará a los usuarios que hayan pujado en la subasta de la cancelación.
- **RSP-2.2.6.** Se registrará una transacción en la base de datos por la cancelación de la subasta con los siguientes datos:
  - **RSP-2.2.6.1.** Identificador único (UID), generado por la base de datos.
  - **RSP-2.2.6.2.** Identificador único (UID) del usuario que cancela la subasta.
  - **RSP-2.2.6.3.** Nombre de usuario.
  - **RSP-2.2.6.4.** Identificador único (UID) de la carta subastada.
  - **RSP-2.2.6.5.** Identificador de colección (IDC) de la carta subastada.
  - **RSP-2.2.6.6.** Identificador único (UID) de la subasta.
  - **RSP-2.2.6.7.** Concepto de la transacción, en este caso, la cancelación de una subasta.
  - **RSP-2.2.6.8.** Precio base de la subasta.
  - **RSP-2.2.6.9.** Fecha.
- **RSP-2.2.7.** Se registrar una transacción por cada puja cancelada con los siguientes datos:
  - **RSP-2.2.7.1.** Identificador único (UID), generado por la base de datos.
  - **RSP-2.2.7.2.** Identificador único (UID) del usuario que realizó la puja.
  - **RSP-2.2.7.3.** Nombre de usuario.
  - **RSP-2.2.7.4.** Identificador único (UID) de la carta subastada.
  - **RSP-2.2.7.5.** Identificador de colección (IDC) de la carta subastada.
  - **RSP-2.2.7.6.** Identificador único (UID) de la subasta.
  - **RSP-2.2.7.7.** Identificador único (UID) de la puja.
  - **RSP-2.2.7.8.** Concepto de la transacción, en este caso, la cancelación de una subasta.
  - **RSP-2.2.7.9.** Precio de la puja.
  - **RSP-2.2.7.10.** Fecha.

**RSP-2.3.** El sistema permitirá a los usuarios autenticados pujar en subastas activas.

- **RSP-2.3.1.** El sistema verificará que no se trata de una subasta propia del usuario.
- **RSP-2.3.2.** Un usuario deberá especificar el precio de la puja.
- **RSP-2.3.3.** El sistema verificará que el usuario no tenga una puja activa en la subasta.
- **RSP-2.3.4.** El sistema registrará en la base de datos una nueva puja con los siguientes datos:
  - **RSP-2.3.4.1.** Identificador único (UID).
  - **RSP-2.3.4.2.** Identificador único (UID) del usuario que realizó la puja.
  - **RSP-2.3.4.3.** Nombre de usuario.
  - **RSP-2.3.4.4.** Identificador único (UID) de la carta subastada.
  - **RSP-2.3.4.5.** Identificador de colección (IDC) de la carta subastada.
  - **RSP-2.3.4.6.** Identificador único (UID) de la subasta.
  - **RSP-2.3.4.7.** Estado de la puja 'activa'.

**RSP-2.3.4.8.** Precio de la puja.
**RSP-2.3.4.9.** Fecha de la puja.
**RSP-2.3.4.10.** Fecha estimada de finalización.
**RSP-2.3.5.** El sistema mostrará la puja activa en la lista de pujas activas.
**RSP-2.3.6.** Se registrará una transacción en la base de datos por la puja realizada con los siguientes datos:
**RSP-2.3.6.1.** Identificador único (UID), generado por la base de datos.
**RSP-2.3.6.2.** Identificador único (UID) del usuario que realizó la puja.
**RSP-2.3.6.3.** Nombre de usuario.
**RSP-2.3.6.4.** Identificador único (UID) de la carta subastada.
**RSP-2.3.6.5.** Identificador de colección (IDC) de la carta subastada.
**RSP-2.3.6.6.** Identificador único (UID) de la subasta.
**RSP-2.3.6.7.** Identificador único (UID) de la puja.
**RSP-2.3.6.8.** Concepto de la transacción, en este caso, la realización de una puja.
**RSP-2.3.6.9.** Precio de la puja.
**RSP-2.3.6.10.** Fecha.

**RSP-3.** El sistema permitirá a los usuarios autenticados visualizar las pujas realizadas y que están pendientes de resolución.

**RSP-3.1.** Se mostrará el precio de la puja realizada.
**RSP-3.2.** El sistema permitirá al usuario retirar la puja.
**RSP-3.2.1.** El sistema verificará que la puja seleccionada pertenezca al usuario.
**RSP-3.2.2.** El sistema actualizará el estado de la puja a 'cancelada'.
**RSP-3.2.3.** La puja no se tendrá en cuenta en el proceso de selección de la puja ganadora.
**RSP-3.2.4.** El sistema no mostrará la puja en la lista de pujas activas.
**RSP-3.2.5.** Se registrará una transacción en la base de datos por la cancelación de la puja con los siguientes datos:
**RSP-3.2.5.1.** Identificador único (UID), generado por la base de datos.
**RSP-3.2.5.2.** Identificador único (UID) del usuario que realizó la puja.
**RSP-3.2.5.3.** Nombre de usuario.
**RSP-3.2.5.4.** Identificador único (UID) de la carta subastada.
**RSP-3.2.5.5.** Identificador de colección (IDC) de la carta subastada.
**RSP-3.2.5.6.** Identificador único (UID) de la subasta.
**RSP-3.2.5.7.** Identificador único (UID) de la puja.
**RSP-3.2.5.8.** Concepto de la transacción, en este caso, la cancelación de una puja.
**RSP-3.2.5.9.** Precio de la puja.
**RSP-3.2.5.10.** Fecha.

**RSP-4.** El administrador del sistema podrá visualizar todas las subastas activas.

**RSP-4.1.** El administrador puede consultar el precio base que el vendedor ha establecido.
**RSP-4.2.** Si considera que la subasta no cumple con las normas del sistema, podrá cancelarla.
**RSP-4.2.1.** Se notificará a los usuarios implicados en la subasta de la cancelación.
**RSP-4.2.2.** Se le enviará una notificación en tiempo real al usuario vendedor.

**RSP-4.2.3.** Se registrará una transacción en la base de datos por la cancelación de la subasta como se describe en Retirar subasta.

**RSP-5.** El administrador del sistema podrá finalizar las subastas activas cuya duración haya expirado.

**RSP-5.1.** El administrador confirmará la finalización de la subastas.

**RSP-5.2.** Si hay pujas activas, el sistema seleccionará la puja ganadora.

**RSP-5.2.1.** El sistema verificará que el usuario tenga saldo suficiente para realizar la compra.

**RSP-5.2.2.** Si el usuario no tiene saldo suficiente se seleccionará la siguiente puja más alta, repitiendo el proceso hasta encontrar un usuario con saldo suficiente.

**RSP-5.2.3.** Si el usuario tiene saldo suficiente se procederá a la compra de la carta.

**RSP-5.2.4.** En el caso de que no haya ninguna puja válida, la carta se devolverá a la colección del usuario vendedor.

**RSP-5.2.5.** En el caso de que haya una puja válida:

**RSP-5.2.5.1.** Se registrará una transacción en la base de datos con los siguientes datos:

**RSP-5.2.5.1.1.** Identificador único (UID), generado por la base de datos.

**RSP-5.2.5.1.2.** Identificador único (UID) del usuario que realizó la puja.

**RSP-5.2.5.1.3.** Nombre de usuario comprador.

**RSP-5.2.5.1.4.** Identificador único (UID) de la carta subastada.

**RSP-5.2.5.1.5.** Identificador de colección (IDC) de la carta subastada.

**RSP-5.2.5.1.6.** Identificador único (UID) de la subasta.

**RSP-5.2.5.1.7.** Identificador único (UID) de la puja.

**RSP-5.2.5.1.8.** Concepto de la transacción, en este caso, la compra de una carta al ganar una subasta.

**RSP-5.2.5.1.9.** Precio de la puja.

**RSP-5.2.5.1.10.** Fecha.

**RSP-5.2.5.2.** El sistema actualizará el saldo del usuario vendedor.

**RSP-5.2.5.3.** El sistema actualizará el saldo del usuario comprador.

**RSP-5.2.5.4.** El sistema transferirá la carta al usuario comprador.

**RSP-5.2.5.5.** El sistema actualizará el estado de la puja a 'ganadora'.

**RSP-5.2.5.6.** El sistema actualizará la fecha de finalización de la puja a la fecha en la que el administrador confirmó la finalización.

**RSP-5.2.5.7.** El sistema enviará una notificación al usuario vendedor en tiempo real.

**RSP-5.2.5.8.** El sistema enviará una notificación al usuario comprador en tiempo real.

**RSP-5.2.6.** Si no hay ninguna puja válida, el sistema devolverá la carta subastada a la colección del usuario.

**RSP-5.3.** El sistema notificará a los usuarios implicados en la subasta del resultado.

**RSP-5.4.** Se registrará la transacción de finalización de la subasta con los siguientes datos:

**RSP-5.4.1.** Identificador único (UID), generado por la base de datos.

**RSP-5.4.2.** Identificador único (UID) del usuario que realizó la puja.

**RSP-5.4.3.** Nombre de usuario.

**RSP-5.4.4.** Identificador único (UID) de la carta subastada.

**RSP-5.4.5.** Identificador de colección (IDC) de la carta subastada.

**RSP-5.4.6.** Identificador único (UID) de la subasta.

**RSP-5.4.7.** Identificador único (UID) de la puja.

**RSP-5.4.8.** Concepto de la transacción.

**RSP-5.4.8.1.** Si hay una puja válida, el concepto será 'venta de carta'.

**RSP-5.4.8.2.** Si no hay ninguna puja válida, el concepto será 'carta no vendida'.

**RSP-5.4.9.** Precio de la puja, si hay una puja válida.

**RSP-5.4.10.** Fecha.

**RSP-5.5.** El sistema actualizará el estado de la subasta a 'finalizada'.

**RSP-5.6.** El sistema actualizará la fecha de finalización de la subasta a la fecha en la que el administrador confirmó la finalización.

#### *6.2.1.1.6 Gestión de transacciones*

**RT-1.** El sistema deberá de registrar todos los movimientos de cartas realizados por los usuarios.

**RT-1.1.** Una transacción por cada carta aquirida mediante compra de sobres.

**RT-1.2.** Una transacción por cada carta puesta en subasta.

**RT-1.3.** Una transacción por cada carta retirada de subasta.

**RT-1.4.** Una transacción por cada carta adquirida mediante subasta.

**RT-1.5.** Una transacción por cada puja realizada en una subasta.

**RT-1.6.** Una transacción por cada puja retirada de una subasta.

**RT-1.7.** Una transacción por cada puja ganadora en una subasta.

**RT-2.** Cada transacción en el sistema deberá de poder identicarse de forma inequívoca.

**RT-2.1.** Deberá tener un identificador único.

**RT-2.2.** Deberá tener un identificador de usuario.

**RT-2.3.** Deberá tener un concepto explicativo.

**RT-2.4.** Deberá tener una fecha y hora de realización.

**RT-2.5.** Deberá tener un importe.

**RT-2.6.** Deberá de tener una referencia al activo afectado.

**RT-3.** El sistema deberá de permitir a los usuarios autenticados consultar sus transacciones.

**RT-4.** En el detalle de cada carta, el usuario podrá consultar las transacciones realizadas con esa carta.

**RT-4.1.** Solo se mostrarán las transacciones correspondientes a una transferencia de carta.

**RT-4.1.1.** Carta adquirida mediante compra de sobres.

**RT-4.1.2.** Carta adquirida mediante subasta.

**RT-4.2.** No se mostrarán transacciones relacionadas con pujas o puestas en subasta.

**RT-4.3.** No se mostrará el usuario que realizó la transacción.

**RT-5.** Un usuario con rol de administrador deberá de poder consultar las transacciones de todos los usuarios.

#### 6.2.1.2. Requisitos no funcionales

**RNF-1.** La aplicación deberá de ser accesible desde los navegadores:

- **RNF-1.1.** Google Chrome
- **RNF-1.2.** Mozilla Firefox
- **RNF-1.3.** Microsoft Edge
- **RNF-1.4.** Safari

**RNF-2.** El sistema debe cumplir con las normativas de protección de datos:

- **RNF-2.1.** Reglamento General de Protección de Datos (RGPD)
- **RNF-2.2.** Ley Orgánica de Protección de Datos y Garantía de los Derechos Digitales (LOPDGDD)

**RNF-3.** El sistema cumplirá con las normas de accesibilidad Web Content Accessibility Guidelines (WCAG) 2.1:

- **RNF-3.1.** La aplicación debe alcanzar un nivel de accesibilidad AAA.
  - **RNF-3.1.1.** Para verificar el cumplimiento de la accesibilidad, se utilizarán las siguientes herramientas:
    - **RNF-3.1.1.1.** WAVE
    - **RNF-3.1.1.2.** Google Chrome Lighthouse

**RNF-4.** La aplicación deberá de ser *responsive* a diferentes resoluciones de pantalla.

- **RNF-4.1.** Se deberá de adaptar a las siguientes resoluciones:
  - **RNF-4.1.1.** Monitor Full HD: 1920x1080
  - **RNF-4.1.2.** Monitor HD: 1366x768
  - **RNF-4.1.3.** Tablet: 1024x768
  - **RNF-4.1.4.** Móvil: 360x640
- **RNF-4.2.** Se comprobará la adaptabilidad con:
  - **RNF-4.2.1.** Google Chrome DevTools
  - **RNF-4.2.2.** Extensión de Chrome Mobile FIRST

**RNF-5.** Los usuarios de la aplicación deberán tener conocimientos tecnológicos medios.

- **RNF-5.1.** El usuario deberá de tener conocimientos básicos de navegación web y haber realizado compras online anteriormente.

**RNF-6.** El sistema se conectará con la base de datos que albergará todos los datos asociados a los usuarios registrados y todas las transacciones realizadas dentro de la aplicación.

- **RNF-6.1.** Las base de datos estará alojada en la nube.
- **RNF-6.2.** Los tiempos de carga de datos no deberán sobrepasar los 10 segundos.

**RNF-7.** El sistema se comunicará con los siguientes servicios externos:

- **RNF-7.1.** PokéAPI
- **RNF-7.2.** MongoDB Atlas
- **RNF-7.3.** PayPal

## 6.3. ESPECIFICACIÓN DE CASOS DE USO

En esta sección se detallan las especificaciones de los casos de uso identificados para el proyecto.

Los casos de uso son descripciones de las interacciones entre los actores y el sistema y son fundamentales para la definición de los requisitos funcionales del mismo.

La especificación de cada caso de uso se ha basado en las recomendaciones de Cockburn [10]. Cada caso de uso incluye los siguientes elementos:

- **Nombre del caso de uso**: un nombre corto y descriptivo.
- **Descripción**: una descripción general del caso de uso.
- **Actores principales**: los actores que inician el caso de uso.
- **Actores secundarios**: los actores que participan en el caso de uso, pero no lo inician.
- **Precondiciones**: las condiciones que deben cumplirse antes de que el caso de uso pueda comenzar.
- **Postcondiciones**: las condiciones que deben cumplirse al finalizar el caso de uso.
- **Disparadores**: los eventos que inician el caso de uso.
- **Escenario principal**: la secuencia de pasos que describe la interacción entre los actores y el sistema.
- **Escenarios alternativos**: descripciones de las ramificaciones del escenario principal.
- **Situaciones de error**: descripciones de las situaciones en las que el caso de uso puede fallar.

Los casos de uso están organizados en secciones según la funcionalidad que describen. Cada sección contiene una tabla con la información detallada de cada caso de uso.

### 6.3.1. Casos de uso. Registrarse e iniciar sesión

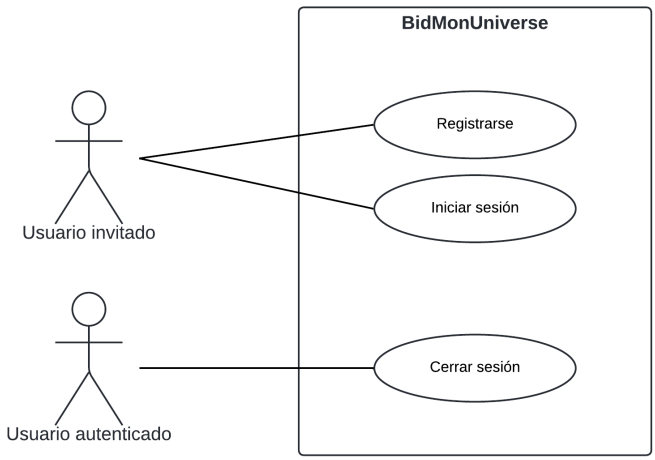


Figura 6.1: Casos de uso. Registrarse e iniciar sesión

#### 6.3.1.1. Caso de uso. Registrarse

Tabla 6.1: Caso de uso. Registrarse

| **Caso de uso** | **REGISTRO** |
|---|---|
| Descripción | Un usuario no autenticado se puede registrar en el sistema para poder acceder a las funcionalidades del mismo. |
| Actores principales | Usuario no autenticado |
| Actores secundarios | |
| Precondiciones | El usuario no debe estar registrado en el sistema. |
| Postcondiciones | ▪ Se crea un nuevo registro en la base de datos con los datos del usuario.<br>▪ Se notifica al usuario que su registro ha sido exitoso.<br>▪ Se redirige al usuario a la página de inicio de sesión. |
| Disparadores | El usuario hace clic en el botón de registro. |
| Escenario principal | 1. El sistema muestra el formulario de registro.<br>2. El usuario completa el formulario con sus datos personales.<br>3. El usuario hace clic en el botón de registro.<br>4. El sistema valida los datos del formulario.<br>5. El sistema crea un nuevo registro en la base de datos con los datos del usuario.<br>6. El sistema notifica al usuario que su registro ha sido exitoso.<br>7. El sistema redirige al usuario a la página de inicio de sesión. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela el registro.**<br>1. El usuario hace clic en otro enlace.<br>2. El sistema no crea el registro y redirige al usuario a la página correspondiente.<br>▪ **Escenario alternativo 2. El usuario ya está registrado en el sistema.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema no crea el registro y redirige al usuario de nuevo al formulario de registro.<br>▪ **Escenario alternativo 3. El usuario no completa el formulario correctamente.**<br>1. El sistema muestra un mensaje de error con los campos que no se han completado correctamente.<br>2. El sistema no crea el registro y redirige al usuario de nuevo al formulario de registro. |

Continúa en la siguiente página...

Tabla 6.1 Caso de uso. Registrarse – continuación de la página anterior

| **Caso de uso** | **REGISTRO** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema redirige al usuario de nuevo al formulario de registro. |

#### 6.3.1.2. Caso de uso. Iniciar sesión

Tabla 6.2: Caso de uso. Iniciar sesión

| **Caso de uso** | **INICIAR SESIÓN** |
|---|---|
| Descripción | Un usuario no autenticado puede iniciar sesión en el sistema para acceder a las funcionalidades del mismo. |
| Actores principales | Usuario no autenticado |
| Actores secundarios | |
| Precondiciones | El usuario debe de haberse registrado previamente en el sistema. |
| Postcondiciones | ▪ Se crea una nueva sesión para el usuario.<br>▪ Se redirige al usuario a la página de inicio. |
| Disparadores | El usuario hace clic en el botón de inicio de sesión. |
| Escenario principal | 1. El sistema muestra el formulario de inicio de sesión.<br>2. El usuario completa el formulario con sus credenciales.<br>3. El usuario hace clic en el botón de inicio de sesión.<br>4. El sistema valida las credenciales del usuario.<br>5. El sistema crea una nueva sesión para el usuario.<br>6. El sistema redirige al usuario a la página de inicio. |

Continúa en la siguiente página...

Tabla 6.2 Caso de uso. Iniciar sesión – continuación de la página anterior

| Caso de uso | INICIAR SESIÓN |
|---|---|
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela el inicio de sesión.**<br>1. El usuario hace clic en otro enlace.<br>2. El sistema no crea la sesión para el usuario y redirige al usuario a la página correspondiente.<br>▪ **Escenario alternativo 2. El usuario no está registrado en el sistema o la contraseña es incorrecta.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema no crea la sesión para el usuario y redirige al usuario de nuevo al formulario de inicio de sesión.<br>▪ **Escenario alternativo 3. El usuario no completa el formulario correctamente.**<br>1. El sistema muestra un mensaje de error con los campos que no se han completado correctamente.<br>2. El sistema no crea la sesión para el usuario y redirige al usuario de nuevo al formulario de inicio de sesión. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema redirige al usuario de nuevo al formulario de inicio de sesión. |

#### 6.3.1.3. Caso de uso. Cerrar sesión

Tabla 6.3: Caso de uso. Cerrar sesión

| Caso de uso | CERRAR SESIÓN |
|---|---|
| Descripción | Un usuario autenticado puede cerrar sesión en el sistema. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | El usuario debe de haber iniciado sesión previamente en el sistema. |
| Postcondiciones | ▪ Se elimina la sesión del usuario. |
| Disparadores | El usuario hace clic en el botón de cerrar sesión. |

Continúa en la siguiente página...

Tabla 6.3 Caso de uso. Cerrar sesión – continuación de la página anterior

| **Caso de uso** | **CERRAR SESIÓN** |
|---|---|
| Escenario principal | 1. El sistema muestra la opción de cerrar sesión.<br>2. El usuario hace clic en el botón de cerrar sesión.<br>3. El sistema elimina la sesión del usuario.<br>4. El sistema redirige al usuario a la página de inicio. |
| Escenarios alternativos | |
| Situaciones de error | |

### 6.3.2. Casos de uso. Gestión de perfil

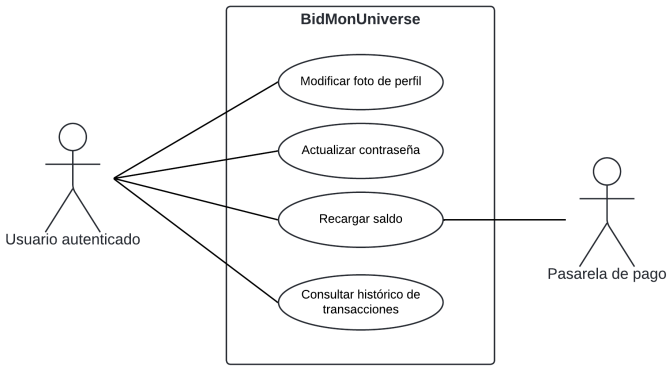


Figura 6.2: Casos de uso. Gestión de perfil

#### 6.3.2.1. Caso de uso. Modificar foto de perfil

Tabla 6.4: Caso de uso. Modificar foto de perfil

| **Caso de uso** | **MODIFICAR PERFIL** |
|---|---|
| Descripción | Un usuario autenticado puede modificar su foto de perfil de usuario. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | ▪ Se modifica el campo de la foto de perfil del usuario en la base de datos. |
| Disparadores | El usuario accede a la sección de modificar perfil. |

Continúa en la siguiente página...

Tabla 6.4 Caso de uso. Modificar foto de perfil – continuación de la página anterior

| **Caso de uso** | **MODIFICAR PERFIL** |
|---|---|
| Escenario principal | 1. El usuario accede a la sección de modificar perfil.<br>2. El sistema muestra la opción de modificar la foto de perfil.<br>3. El usuario hace clic en la opción de modificar la foto de perfil.<br>4. El sistema muestra las imágenes predefinidas para que el usuario pueda seleccionar una.<br>5. El usuario selecciona una imagen.<br>6. El usuario hace clic en el botón de guardar.<br>7. El sistema modifica el campo de la foto de perfil del usuario en la base de datos.<br>8. El sistema muestra un mensaje de éxito.<br>9. El sistema redirige al usuario a la página principal. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela la modificación de perfil.**<br>1. El usuario no hace clic en el botón de guardar.<br>2. El sistema no modifica los datos del perfil del usuario. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no modificará los datos del perfil del usuario. |

#### 6.3.2.2. Caso de uso. Actualizar contraseña

Tabla 6.5: Caso de uso. Actualizar contraseña

| **Caso de uso** | **ACTUALIZAR CONTRASEÑA** |
|---|---|
| Descripción | Un usuario autenticado puede actualizar su contraseña de usuario. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | ▪ Se modifica la contraseña del usuario en la base de datos. |

Continúa en la siguiente página...

Tabla 6.5 Caso de uso. Actualizar contraseña – continuación de la página anterior

| **Caso de uso** | **ACTUALIZAR CONTRASEÑA** |
|---|---|
| Disparadores | El usuario accede a la sección de modificar perfil. |
| Escenario principal | 1. El usuario accede a la sección de modificar perfil.<br>2. El sistema muestra la opción de modificar la contraseña.<br>3. El usuario hace clic en la opción de modificar la contraseña.<br>4. El sistema muestra un formulario con los campos de la nueva contraseña y la confirmación de la nueva contraseña.<br>5. El usuario completa los campos del formulario.<br>6. El usuario hace clic en el botón de guardar.<br>7. El sistema modifica el campo de la contraseña del usuario en la base de datos.<br>8. El sistema muestra un mensaje de éxito.<br>9. El sistema redirige al usuario a la página principal. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela la modificación de perfil.**<br>1. El usuario no hace clic en el botón de guardar.<br>2. El sistema no modifica los datos del perfil del usuario.<br>▪ **Escenario alternativo 2. El usuario no completa correctamente los campos del formulario.**<br>1. El usuario no completa correctamente los campos del formulario.<br>2. El sistema muestra un mensaje de error.<br>3. El sistema no modifica los datos del perfil del usuario. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no modificará los datos del perfil del usuario. |

#### 6.3.2.3. Caso de uso. Recargar saldo

Tabla 6.6: Caso de uso. Recargar saldo

| **Caso de uso** | **RECRAGAR SALDO** |
|---|---|
| Descripción | Un usuario autenticado puede recargar saldo en su cuenta. |
| Actores principales | Usuario autenticado |
| Actores secundarios | Pasarela de pago |

Continúa en la siguiente página...

Tabla 6.6 Caso de uso. Recargar saldo – continuación de la página anterior

| **Caso de uso** | **RECARGAR SALDO** |
|---|---|
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | ▪ Se incrementa el saldo del usuario en la base de datos. |
| Disparadores | El usuario accede a la sección de recarga de saldo. |
| Escenario principal | 1. El sistema muestra el formulario de recarga de saldo.<br>2. El usuario introduce la cantidad de saldo que desea recargar.<br>3. El usuario hace clic en el botón de recargar saldo.<br>4. El sistema valida la cantidad de saldo introducida.<br>5. El sistema redirige al usuario a la pasarela de pago.<br>6. El usuario completa el pago.<br>7. El sistema incrementa el saldo del usuario en la base de datos.<br>8. El sistema muestra un mensaje de éxito. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela la recarga de saldo en la pasarela de pago.**<br>1. El usuario cancela el pago.<br>2. El sistema no incrementa el saldo del usuario en la base de datos.<br>3. El sistema redirige al usuario a la página que muestra su saldo actual.<br>4. El sistema muestra un mensaje de cancelación.<br>▪ **Escenario alternativo 2. El usuario introduce una cantidad de saldo inválida.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema muestra los campos con errores.<br>3. El sistema permite al usuario corregir los errores. |

Continúa en la siguiente página...

Tabla 6.6 Caso de uso. Recargar saldo – continuación de la página anterior

| Caso de uso | RECARGAR SALDO |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no incrementará el saldo del usuario en la base de datos.<br>3. El sistema le dará al usuario la opción de intentar recargar de nuevo el saldo o volver a la página de inicio.<br>▪ **Error en la pasarela de pago.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no incrementará el saldo del usuario en la base de datos.<br>3. El sistema redirigirá al usuario a la página de recarga de saldo. |

#### 6.3.2.4. Caso de uso. Histórico de transacciones realizadas

Tabla 6.7: Caso de uso. Histórico de transacciones realizadas

| Caso de uso | HISTÓRICO DE TRANSACCIONES REALIZADAS |
|---|---|
| Descripción | Un usuario autenticado puede consultar el histórico de transacciones realizadas en el sistema. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de historial de transacciones. |
| Escenario principal | 1. El sistema muestra el historial de transacciones del usuario. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario no tiene transacciones realizadas.**<br>1. El sistema mostrará un mensaje indicando que el usuario no tiene transacciones realizadas. |

Continúa en la siguiente página...

Tabla 6.7 Caso de uso. Histórico de transacciones realizadas – continuación de la página anterior

| **Caso de uso** | **HISTÓRICO DE TRANSACCIONES REALIZADAS** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema le dará al usuario la opción de intentar cargar de nuevo el historial de transacciones o volver a la página principal. |

### 6.3.3. Casos de uso. Gestión de colección de cartas

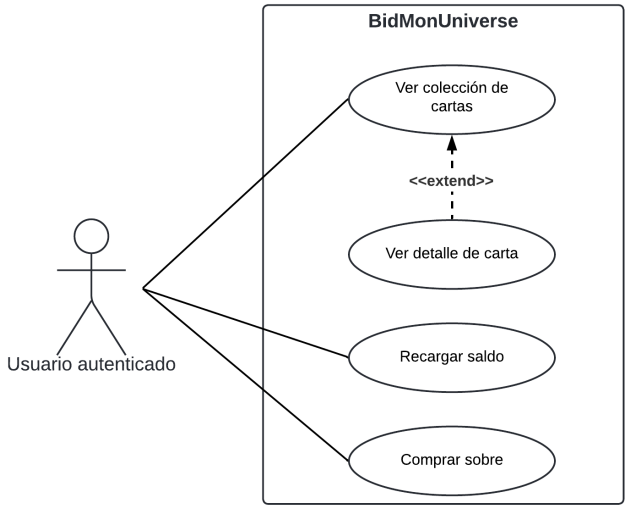


Figura 6.3: Casos de uso. Gestión de colección de cartas

#### 6.3.3.1. Caso de uso. Ver colección de cartas

Tabla 6.8: Caso de uso. Ver colección de cartas

| **Caso de uso** | **VER COLECCIÓN DE CARTAS** |
|---|---|
| Descripción | Un usuario autenticado puede ver la colección de cartas que posee. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de colección de cartas. |
| Escenario principal | 1. El sistema muestra la colección de cartas del usuario. |

Continúa en la siguiente página...

Tabla 6.8 Caso de uso. Ver colección de cartas – continuación de la página anterior

| **Caso de uso** | **VER COLECCIÓN DE CARTAS** |
|---|---|
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario no tiene cartas en su colección.**<br>1. Se mostrará un mensaje indicando que el usuario no tiene cartas en su colección.<br>2. Se mostrará la opción de comprar un sobre de cartas. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema le dará al usuario la opción de volver a la página principal. |

#### 6.3.3.2. Caso de uso. Ver detalle de una carta

Tabla 6.9: Caso de uso. Ver detalle de una carta

| **Caso de uso** | **VER DETALLE DE UNA CARTA** |
|---|---|
| Descripción | Un usuario autenticado puede ver el detalle de una carta de su colección. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de colección de cartas y selecciona una carta. |
| Escenario principal | 1. El sistema muestra la colección de cartas del usuario.<br>2. El usuario puede seleccionar una carta para ver su información detallada.<br>3. El sistema redirige al usuario a la página de detalle de la carta seleccionada.<br>4. El usuario puede consultar la información detallada de la carta.<br>5. El usuario puede consultar el histórico de transacciones de la carta.<br>6. Si la carta no está en subasta, el usuario puede ponerla en subasta. |
| Escenarios alternativos | |

Continúa en la siguiente página...

Tabla 6.9 Caso de uso. Ver detalle de una carta – continuación de la página anterior

| **Caso de uso** | **VER DETALLE DE UNA CARTA** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema le dará al usuario la opción de volver a la página principal. |

#### 6.3.3.3. Caso de uso. Comprar sobre

Tabla 6.10: Caso de uso. Comprar sobre

| **Caso de uso** | **COMPRAR SOBRE** |
|---|---|
| Descripción | Un usuario autenticado puede comprar un sobre de cartas. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario ha iniciado sesión en el sistema.<br>▪ El usuario dispone de saldo suficiente para comprar el sobre. |
| Postcondiciones | ▪ Se descuenta el precio del sobre del saldo del usuario.<br>▪ Se añaden las cartas del sobre a la colección del usuario.<br>▪ Se decrementa en una unidad la cantidad de sobres disponibles en el inventario.<br>▪ Se registra la transacción en el historial de compras del usuario. |
| Disparadores | El usuario hace clic en el botón de comprar sobre. |
| Escenario principal | 1. El sistema muestra el inventario de sobres disponibles.<br>2. El usuario selecciona el sobre que desea comprar.<br>3. El usuario hace clic en el botón de comprar sobre.<br>4. El sistema valida que el usuario dispone de saldo suficiente.<br>5. El sistema descuenta el precio del sobre del saldo del usuario.<br>6. El sistema genera las cartas del sobre.<br>7. El sistema añade las cartas del sobre a la colección del usuario. |

Continúa en la siguiente página...

Tabla 6.10 Caso de uso. Comprar sobre – continuación de la página anterior

| **Caso de uso** | **COMPRAR SOBRE** |
|---|---|
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario intenta comprar un sobre sin saldo suficiente.**<br>1. El usuario intenta comprar un sobre sin saldo suficiente.<br>2. El sistema muestra un mensaje de error.<br>3. El sistema le ofrece al usuario la posibilidad de recargar saldo. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema no descuenta el precio del sobre del saldo del usuario.<br>3. El sistema no añade las cartas del sobre a la colección del usuario.<br>4. El sistema no decrementa en una unidad la cantidad de sobres disponibles en el inventario. |

### 6.3.4. Casos de uso. Gestión de subastas y pujas

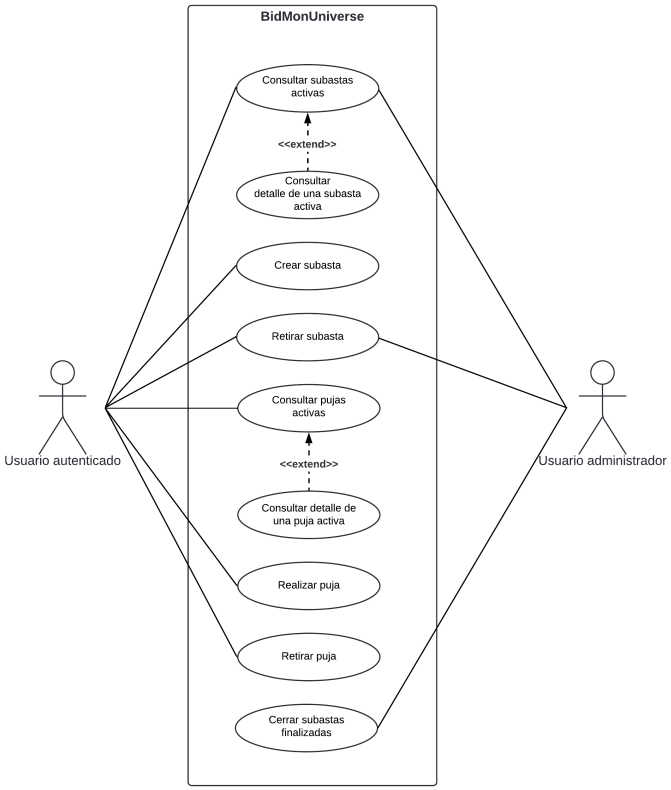


Figura 6.4: Casos de uso. Gestión de subastas y pujas

#### 6.3.4.1. Caso de uso. Crear una subasta

Tabla 6.11: Caso de uso. Crear una subasta

| **Caso de uso** | **CREAR UNA SUBASTA** |
|---|---|
| Descripción | Un usuario autenticado puede crear una subasta de una carta de su colección. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |

Continúa en la siguiente página...

Tabla 6.11 Caso de uso. Crear una subasta – continuación de la página anterior

| **Caso de uso** | **CREAR UNA SUBASTA** |
|---|---|
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema.<br>▪ El usuario debe tener al menos una carta en su colección.<br>▪ El usuario no debe tener ya una subasta activa para la carta que desea subastar. |
| Postcondiciones | ▪ Se crea una nueva subasta en la base de datos.<br>▪ Se marca la carta como 'en subasta' en la base de datos. |
| Disparadores | El usuario accede a la sección de subastas y hace clic en el botón de crear subasta. |
| Escenario principal | 1. El sistema muestra el formulario de creación de subasta.<br>2. El usuario selecciona la carta que desea subastar.<br>3. El usuario introduce el precio de salida y la duración de la subasta.<br>4. El usuario hace clic en el botón de crear subasta.<br>5. El sistema valida los campos del formulario.<br>6. El sistema crea una nueva subasta en la base de datos.<br>7. El sistema marca la carta como 'en subasta' en la base de datos.<br>8. El sistema muestra un mensaje de éxito.<br>9. El sistema redirige al usuario a la página de subastas.<br>10. El sistema muestra la subasta creada en la lista de subastas. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario ya tiene una subasta activa para la carta que desea subastar.**<br>1. El usuario intenta crear una subasta para una carta que ya está en subasta.<br>2. El sistema muestra un mensaje de error.<br>3. El sistema redirige al usuario a la página de subastas.<br>▪ **Escenario alternativo 2. El usuario introduce datos inválidos.**<br>1. El sistema muestra un mensaje de error.<br>2. El sistema no crea la subasta en la base de datos.<br>3. El sistema no marcará la carta como 'en subasta' en la base de datos.<br>4. El sistema le mostrará al usuario los campos con errores.<br>5. El sistema permitirá al usuario corregir los errores. |

Continúa en la siguiente página...

Tabla 6.11 Caso de uso. Crear una subasta – continuación de la página anterior

| Caso de uso | CREAR UNA SUBASTA |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no creará la subasta en la base de datos.<br>3. El sistema no marcará la carta como 'en subasta' en la base de datos.<br>4. El sistema redirigirá al usuario a la página de subastas. |

#### 6.3.4.2. Caso de uso. Retirar una subasta

Tabla 6.12: Caso de uso. Retirar una subasta

| Caso de uso | RETIRAR UNA SUBASTA |
|---|---|
| Descripción | Un usuario autenticado puede retirar una subasta activa en el sistema si es el vendedor de la subasta. Un usuario administrador puede retirar cualquier subasta activa en el sistema. |
| Actores principales | Usuario autenticado o usuario administrador |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema.<br>▪ El usuario debe tener al menos una subasta activa. |
| Postcondiciones | ▪ Se marca la subasta como 'cancelada' en la base de datos.<br>▪ La carta subastada vuelve a estar disponible en la colección del usuario.<br>▪ Se actualizan las pujas de la subasta en la base de datos, marcando las pujas como 'canceladas'.<br>▪ Se informa a los usuarios que hayan pujado en la subasta de que ha sido cancelada.<br>▪ Si la subasta ha sido retirada por un administrador, se notifica al usuario vendedor de la subasta en tiempo real. |
| Disparadores | El usuario accede a la sección de subastas y hace clic en el botón de retirar subasta. |

Continúa en la siguiente página...

Tabla 6.12 Caso de uso. Retirar una subasta – continuación de la página anterior

| **Caso de uso** | **RETIRAR UNA SUBASTA** |
|---|---|
| Escenario principal | 1. El sistema muestra la lista de subastas activas del usuario.<br>2. El usuario selecciona la subasta que desea retirar.<br>3. El usuario hace clic en el botón de retirar subasta.<br>4. El sistema muestra un mensaje de confirmación.<br>5. El usuario confirma la retirada de la subasta.<br>6. El sistema marca la subasta como 'cancelada' en la base de datos.<br>7. La carta subastada vuelve a estar disponible en la colección del usuario.<br>8. El sistema muestra un mensaje de éxito.<br>9. El sistema redirige al usuario a la página de subastas. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela la retirada de la subasta.**<br>1. El usuario hace clic en el botón de cancelar en el mensaje de confirmación.<br>2. El sistema no marcará la subasta como 'cancelada' en la base de datos.<br>3. El sistema redirigirá al usuario a la página de subastas. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no marcará la subasta como 'cancelada' en la base de datos.<br>3. El sistema no devolverá la carta subastada a la colección del usuario.<br>4. El sistema redirigirá al usuario a la página de subastas. |

#### 6.3.4.3. Caso de uso. Consultar subastas activas

Tabla 6.13: Caso de uso. Consultar subastas activas

| **Caso de uso** | **CONSULTAR SUBASTAS ACTIVAS** |
|---|---|
| Descripción | Un usuario autenticado o administrador puede consultar las subastas activas en el sistema. |
| Actores principales | Usuario autenticado o usuario administrador |
| Actores secundarios | |

Continúa en la siguiente página...

Tabla 6.13 Caso de uso. Consultar subastas activas – continuación de la página anterior

| **Caso de uso** | **CONSULTAR SUBASTAS ACTIVAS** |
|---|---|
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de subastas. |
| Escenario principal | 1. El sistema muestra la lista de subastas activas.<br>2. El usuario puede seleccionar una subasta para ver su información detallada. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. No hay subastas activas.**<br>1. El sistema mostrará un mensaje indicando que no hay subastas activas.<br>▪ **Escenario alternativo 2. El usuario consulta sus propias subastas.**<br>1. El usuario accede a la sección de subastas.<br>2. El sistema muestra la opción de ver las subastas activas del usuario.<br>3. El usuario selecciona la opción de ver sus propias subastas.<br>4. El sistema muestra la lista de subastas activas del usuario.<br>5. El usuario puede seleccionar una subasta para ver su información detallada.<br>▪ **Escenario alternativo 3. Un usuario administrador consulta las subastas activas.**<br>1. El usuario administrador accede a la sección de subastas.<br>2. El sistema muestra la lista de subastas activas de todos los usuarios.<br>3. El usuario administrador puede seleccionar una subasta para ver su información detallada. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error. |

#### 6.3.4.4. Caso de uso. Consultar el detalle de una subasta activa

Tabla 6.14: Caso de uso. Consultar el detalle de una subasta activa

| **Caso de uso** | **CONSULTAR DETALLE DE UNA SUBASTA ACTIVA** |
|---|---|

Continúa en la siguiente página...

Tabla 6.14 Caso de uso. Consultar el detalle de una subasta activa – continuación de la página anterior

| Caso de uso | CONSULTAR DETALLE DE UNA SUBASTA ACTIVA |
|---|---|
| Descripción | Un usuario autenticado o administrador puede consultar el detalle de una subasta activa en el sistema. |
| Actores principales | Usuario autenticado o usuario administrador |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de subastas y selecciona una subasta activa. |
| Escenario principal | 1. El usuario selecciona una subasta activa.<br>2. El sistema muestra la información detallada de la carta subastada, incluido el histórico de transacciones de esta.<br>3. El sistema muestra la duración restante de la subasta. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario que consulta la subasta es el propietario de la subasta.**<br>1. El usuario selecciona una subasta activa.<br>2. El sistema muestra la información detallada de la carta subastada, incluido el histórico de transacciones de esta.<br>3. El sistema muestra la duración restante de la subasta.<br>4. El sistema muestra el precio de salida establecido por el usuario.<br>5. El sistema muestra la opción de retirar la subasta.<br>▪ **Escenario alternativo 3. Un usuario administrador consulta la subasta.**<br>1. El usuario administrador selecciona una subasta activa.<br>2. El sistema muestra la información detallada de la carta subastada, incluido el histórico de transacciones de esta.<br>3. El sistema muestra la duración restante de la subasta.<br>4. El sistema muestra el precio de salida establecido por el vendedor.<br>5. El sistema muestra la opción de retirar la subasta. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error. |

#### 6.3.4.5. Caso de uso. Realizar puja

Tabla 6.15: Caso de uso. Realizar puja

| **Caso de uso** | **REALIZAR PUJA** |
|---|---|
| Descripción | Un usuario autenticado puede realizar una puja en una subasta activa. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema.<br>▪ El usuario debe tener saldo suficiente para realizar la puja. |
| Postcondiciones | ▪ Se registra la puja en la base de datos. |
| Disparadores | El usuario selecciona una subasta activa y hace clic en el botón de realizar puja. |
| Escenario principal | 1. El sistema muestra la información detallada de la subasta.<br>2. El usuario introduce la cantidad de la puja.<br>3. El usuario hace clic en el botón de realizar puja.<br>4. El sistema valida la cantidad de la puja.<br>5. El sistema registra la puja en la base de datos.<br>6. El sistema muestra un mensaje de éxito.<br>7. El sistema muestra un mensaje informativo, indicando que si en el momento de finalización del tiempo de la subasta no cuenta con el saldo suficiente, la puja no será válida.<br>8. El sistema redirige al usuario a la página de subastas. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario no tiene saldo suficiente para realizar la puja.**<br>1. El usuario intenta realizar una puja sin tener saldo suficiente.<br>2. El sistema mostrará un mensaje de error.<br>3. El sistema no registrará la puja en la base de datos.<br>4. El sistema redirigirá al usuario a la página de subastas.<br>▪ **Escenario alternativo 2. El usuario introduce una cantidad de puja inválida.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no registrará la puja en la base de datos.<br>3. El sistema le mostrará al usuario los campos con errores.<br>4. El sistema permitirá al usuario corregir los errores. |

Continúa en la siguiente página...

Tabla 6.15 Caso de uso. Realizar puja – continuación de la página anterior

| **Caso de uso** | **REALIZAR PUJA** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema no registrará la puja en la base de datos.<br>3. El sistema redirigirá al usuario a la página de subastas. |

#### 6.3.4.6. Caso de uso. Retirar puja

Tabla 6.16: Caso de uso. Retirar puja

| **Caso de uso** | **RETIRAR PUJA** |
|---|---|
| Descripción | Un usuario podrá retirar una puja realizada en una subasta activa. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema.<br>▪ El usuario debe haber realizado una puja en una subasta activa.<br>▪ La subasta no debe haber finalizado. |
| Postcondiciones | ▪ Se actualiza la puja en la base de datos, marcándola como 'retirada'.<br>▪ La puja no se muestra en la lista de pujas activas del usuario.<br>▪ La puja no se tendr´a en cuenta en el cálculo de la puja más alta. |
| Disparadores | El usuario accede a la sección de pujas activas, selecciona una puja y hace clic en el botón de retirar puja. |
| Escenario principal | 1. El sistema muestra la lista de pujas activas del usuario.<br>2. El usuario selecciona la puja que desea retirar.<br>3. El usuario hace clic en el botón de retirar puja.<br>4. El sistema muestra un mensaje de confirmación.<br>5. El usuario confirma la retirada de la puja.<br>6. El sistema actualiza la puja en la base de datos, marcándola como 'retirada'.<br>7. El sistema muestra un mensaje de éxito.<br>8. El sistema redirige al usuario a la página de subastas. |

Continúa en la siguiente página...

Tabla 6.16 Caso de uso. Retirar puja – continuación de la página anterior

| Caso de uso | RETIRAR PUJA |
|---|---|
| Escenarios alternativos | ▪ **Escenario alternativo 1. El usuario cancela la retirada de la puja.**<br>1. El usuario hace clic en el botón de cancelar en el mensaje de confirmación.<br>2. El sistema no marcará la puja como 'retirada' en la base de datos.<br>3. El sistema redirigirá al usuario a la página de subastas. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error.<br>2. El sistema redirigirá al usuario a la página de subastas. |

#### 6.3.4.7. Caso de uso. Consultar pujas activas

Tabla 6.17: Caso de uso. Consultar pujas activas

| Caso de uso | CONSULTAR PUJAS ACTIVAS |
|---|---|
| Descripción | Un usuario autenticado puede consultar las pujas activas en las subastas en las que ha participado. |
| Actores principales | Usuario autenticado o usuario administrador |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de subastas y hace clic en el botón de consultar pujas activas. |
| Escenario principal | 1. El sistema muestra la lista de pujas activas.<br>2. El usuario puede seleccionar una puja para ver su información detallada. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. No hay pujas activas.**<br>1. El sistema mostrará un mensaje indicando que no hay pujas activas. |

Continúa en la siguiente página...

Tabla 6.17 Caso de uso. Consultar pujas activas – continuación de la página anterior

| **Caso de uso** | **CONSULTAR PUJAS ACTIVAS** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error. |

#### 6.3.4.8. Caso de uso. Consultar el detalle de una puja activa

Tabla 6.18: Caso de uso. Consultar el detalle de una puja activa

| **Caso de uso** | **CONSULTAR DETALLE DE UNA PUJA ACTIVA** |
|---|---|
| Descripción | Un usuario autenticado puede consultar el detalle de una puja activa en el sistema. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de pujas activas y selecciona una puja activa. |
| Escenario principal | 1. El usuario selecciona una puja activa.<br>2. El sistema muestra la información detallada de la carta subastada, incluido el histórico de transacciones de esta.<br>3. El sistema muestra la duración restante de la subasta.<br>4. El sistema muestra la cantidad de la puja.<br>5. El sistema muestra la opción de retirar la puja. |
| Escenarios alternativos | |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error. |

#### 6.3.4.9. Caso de uso. Cerrar subastas finalizadas

Tabla 6.19: Caso de uso. Cerrar subastas finalizadas

| **Caso de uso** | **CERRAR SUBASTAS FINALIZADAS** |
|---|---|
| Descripción | Un usuario administrador podrá cerrar las subastas que hayan finalizado. El sistema se encargará de procesar las pujas, notificar a los usuarios ganadores y transferir la carta. |
| Actores principales | Usuario administrador |
| Actores secundarios | |
| Precondiciones | ▪ El usuario debe haber iniciado sesión en el sistema. |
| Postcondiciones | ▪ Se marcan las subastas como 'cerradas' en la base de datos.<br>▪ Se actualiza la información de las cartas subastadas en la base de datos.<br>▪ Se notifica a los dueños de las cartas subastadas del resultado de la subasta.<br>▪ Se notifica a los usuarios ganadores de las subastas si los hubiera.<br>▪ Se transfiere el monto de la puja ganadora a la cuenta del vendedor.<br>▪ Se descuenta el monto de la puja ganadora de la cuenta del usuario ganador.<br>▪ Se transfiere la carta subastada a los usuarios ganadores.<br>▪ Se registran las transacciones en la base de datos. |
| Disparadores | El usuario accede a la sección de subastas activas y hace clic en el botón de cerrar subastas. |
| Escenario principal | 1. El usuario selecciona la opción de cerrar subastas.<br>2. El sistema muestra la lista de subastas finalizadas que han sido cerradas con sus resultados.<br>3. El sistema informa a los dueños de las cartas subastadas del resultado de la subasta.<br>4. El sistema informa a los usuarios ganadores de las subastas si los hubiera. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. No hay subastas finalizadas.**<br>1. El sistema mostrará un mensaje indicando que no hay subastas que cerrar. |

Continúa en la siguiente página...

Tabla 6.19 Caso de uso. Cerrar subastas finalizadas – continuación de la página anterior

| **Caso de uso** | **CERRAR SUBASTAS FINALIZADAS** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema mostrará un mensaje de error. |

### 6.3.5. Casos de uso. Gestión de transacciones

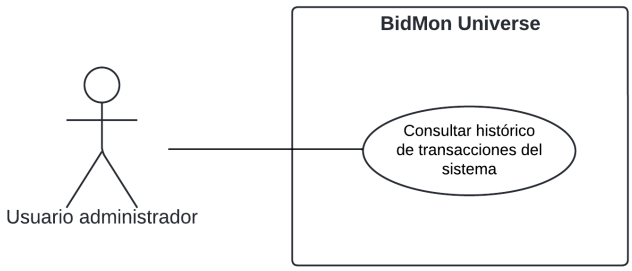


Figura 6.5: Casos de uso. Gestión de transacciones

#### 6.3.5.1. Caso de uso. Consultar histórico de transacciones del sistema

Tabla 6.20: Caso de uso. Histórico de transacciones del sistema

| **Caso de uso** | **HISTÓRICO DE TRANSACCIONES DEL SISTEMA** |
|---|---|
| Descripción | Un usuario administrador puede consultar el histórico de transacciones del sistema. |
| Actores principales | Usuario administrador |
| Actores secundarios | |
| Precondiciones | ▪ El usuario ha iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario selecciona la opción de consultar el histórico de transacciones. |
| Escenario principal | 1. El sistema muestra el listado de transacciones realizadas en el sistema por los diferentes usuarios.<br>2. El usuario puede filtrar las transacciones por diferentes criterios. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. No hay transacciones registradas.**<br>1. El sistema muestra un mensaje indicando que no hay transacciones registradas. |

Continúa en la siguiente página...

Tabla 6.20 – continuación de la página anterior

| **Caso de uso** | **HISTÓRICO DE TRANSACCIONES DEL SISTEMA** |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema muestra un mensaje de error. |

### 6.3.6. Casos de uso. Gestión de notificaciones

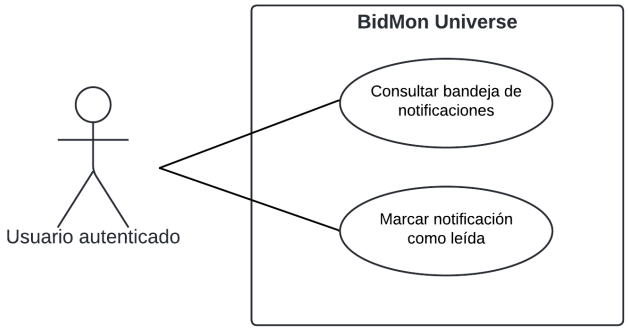


Figura 6.6: Casos de uso. Gestión de notificaciones

#### 6.3.6.1. Caso de uso. Consultar bandeja de notificaciones

Tabla 6.21: Caso de uso. Consultar bandeja de notificaciones

| **Caso de uso** | **CONSULTAR BANDEJA DE NOTIFICACIONES** |
|---|---|
| Descripción | Un usuario autenticado puede consultar las notificaciones recibidas en el sistema. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario ha iniciado sesión en el sistema. |
| Postcondiciones | |
| Disparadores | El usuario accede a la sección de notificaciones. |
| Escenario principal | 1. El sistema muestra el listado de notificaciones recibidas por el usuario. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. No hay notificaciones.**<br>1. El sistema muestra un mensaje indicando que no hay notificaciones. |

Continúa en la siguiente página...

Tabla 6.21 – continuación de la página anterior

| Caso de uso | CONSULTAR BANDEJA DE NOTIFICACIONES |
|---|---|
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema muestra un mensaje de error. |

#### 6.3.6.2. Caso de uso. Marcar notificación como leída

Tabla 6.22: Caso de uso. Marcar notificación como leída

| Caso de uso | MARCAR NOTIFICACIÓN COMO LEÍDA |
|---|---|
| Descripción | Un usuario autenticado puede marcar una notificación como leída. |
| Actores principales | Usuario autenticado |
| Actores secundarios | |
| Precondiciones | ▪ El usuario ha iniciado sesión en el sistema. |
| Postcondiciones | El sistema marca la notificación como leída en la base de datos y registra la fecha y hora de lectura. |
| Disparadores | El usuario accede a la sección de notificaciones y selecciona una notificación. |
| Escenario principal | 1. El sistema muestra el listado de notificaciones recibidas por el usuario.<br>2. El usuario selecciona una notificación.<br>3. El sistema marca la notificación como leída. |
| Escenarios alternativos | ▪ **Escenario alternativo 1. Marcar todas las notificaciones como leídas.**<br>1. El sistema muestra la opción de marcar todas las notificaciones como leídas.<br>2. El usuario selecciona la opción de marcar todas las notificaciones como leídas.<br>3. El sistema marca todas las notificaciones como leídas. |
| Situaciones de error | ▪ **Error de conexión a la base de datos.**<br>1. El sistema muestra un mensaje de error. |

## 6.4. IDENTIFICACIÓN DE SUBSISTEMAS DE ANÁLISIS

En esta sección se detallan los subsistemas por los que se compone el sistema de información, así como los interfaces entre ellos.

### 6.4.1. Descripción de los Subsistemas

En esta sección se enumeran los subsistemas que componen BidMon Universe. Un subsistema es un conjunto de elementos de software que interactúan entre sí para llevar a cabo una serie de tareas relacionadas. La aplicación se divide en dos subsistemas principales: el backend y el frontend.

#### 6.4.1.1. Backend

El backend es el subsistema que se encarga de gestionar la lógica de negocio y la persistencia de los datos. Este subsistema se comunica con el frontend para recibir y enviar información. El backend se divide en los siguientes módulos:

- **Módulo de usuarios:** se encarga de gestionar la autenticación de los usuarios y la información de los mismos.
- **Módulo de cartas:** se encarga de gestionar la colección de cartas disponibles en el sistema.
- **Módulo de sobres:** se encarga de gestionar los sobres de cartas disponibles en el sistema.
- **Módulo de mazos de cartas:** se encarga de gestionar los mazos de cartas disponibles en el sistema.
- **Módulo de cartas de usuario:** se encarga de gestionar la colección de cartas de los usuarios.
- **Módulo de transacciones:** se encarga de gestionar las transacciones realizadas por los usuarios.
- **Módulo de compras:** se encarga de gestionar las compras de sobres de cartas por parte de los usuarios.
- **Módulo de subastas:** se encarga de gestionar las subastas de cartas por parte de los usuarios.
- **Módulo de pujas:** se encarga de gestionar las pujas realizadas por los usuarios en las subastas.
- **Módulo de notificaciones:** se encarga de gestionar las notificaciones enviadas a los usuarios.
- **Módulo de PayPal:** se encarga de gestionar las transacciones realizadas a través de PayPal.

#### 6.4.1.2. Frontend

El frontend es el subsistema que se encarga de gestionar la interfaz de usuario. Este subsistema se comunica con el backend para obtener y enviar información. El frontend se divide en los siguientes módulos:

- **API** se encarga de gestionar las peticiones y respuestas entre el frontend y el backend.
- **Contenido estático** se encarga de gestionar el contenido estático de la aplicación.

- **Lógica** se encarga de gestionar la lógica de la interfaz de usuario, como los estados de Redux y la conexión con Socket.io.
- **Vistas** se encarga de gestionar las vistas de la aplicación.
  - **Componentes** se encarga de gestionar los componentes de la aplicación.
  - **Páginas** se encarga de gestionar las páginas de la aplicación.

### 6.4.2. Descripción de los Interfaces entre Subsistemas

Los subsistemas de BidMon Universe, específicamente el backend y el frontend, interactúan a través de una API REST y Socket.io. Los módulos internos de cada subsistema también interactúan entre sí localmente.

#### 6.4.2.1. API REST

El frontend se comunica con el backend principalmente a través de una API REST. Esta API es responsable de manejar todas las solicitudes HTTP enviadas desde el frontend, incluyendo la autenticación de usuarios, la gestión de cartas y las transacciones.

- **Endpoints**: Los endpoints de la API REST están estructurados para soportar operaciones CRUD sobre los recursos del sistema, como usuarios, cartas, y transacciones.
- **Seguridad**: Se utiliza HTTPS para asegurar la comunicación y JWT (JSON Web Tokens) para la autenticación y autorización de usuarios.
- **Datos Intercambiados**: Los datos se intercambian en formato JSON, con estructuras específicas para cada tipo de operación.

#### 6.4.2.2. Socket.io

Para operaciones en tiempo real, como las notificaciones se hace uso de Socket.io. Este subsistema permite una comunicación bidireccional basada en eventos entre el cliente y el servidor.

- **Conexiones de Sockets**: Los usuarios se autentican en la conexión y se suscriben al canal por defecto, permitiendo la recepción de notificaciones en tiempo real.
- **Manejo de Eventos**: Los eventos como `'notification'` son gestionados a través de sockets para actualizar a los usuarios en tiempo real.

#### 6.4.2.3. Persistencia de Datos

La comunicación entre el backend y MongoDB Atlas se gestiona a través de Mongoose, una biblioteca de modelado de objetos que facilita la estructuración de datos y las operaciones de base de datos.

- **Modelos de Datos**: Cada módulo del backend define modelos Mongoose que corresponden a las colecciones de la base de datos, asegurando que los datos se manejen consistentemente.

- **Transacciones de Datos**: Las operaciones que requieren integridad transaccional, como las compras de sobres de cartas, utilizan sesiones y transacciones de Mongoose para garantizar que se procesen completamente o se reviertan en caso de error.

#### 6.4.2.4. Módulo de PayPal

Este módulo del backend es crucial para la gestión de todas las transacciones financieras realizadas a través de Paypal. Implementa las siguientes funcionalidades clave:

- **Generación de Token de Acceso**: Autenticación con la API de PayPal para obtener permisos de transacción.
- **Creación de Órdenes de Pago**: Envío de detalles de transacciones financieras a PayPal y manejo de las respuestas.
- **Manejo de Respuestas de PayPal**: Procesamiento de las respuestas de la API para confirmar transacciones o manejar errores.

El frontend, mientras tanto, facilita la interacción del usuario con estas funciones, proporcionando interfaces claras y seguras para iniciar pagos y visualizar los resultados de las transacciones.

## 6.5. ARQUITECTURA DE LOS SUBSISTEMAS DE ANÁLISIS

En este apartado se detalla la arquitectura de los subsistemas de análisis. Como se ha visto en el apartado anterior, el sistema se divide en dos subsistemas principales: **restapi** y **frontend**. El subsistema restapi contiene las clases que implementan la API REST y la lógica de negocio, mientras que el subsistema frontend abarca la interfaz de usuario.

Primero, se presentará el diagrama de paquetes para ofrecer una visión general de la estructura del sistema. Este diagrama ilustrará la organización y agrupación de los distintos elementos del sistema en paquetes, permitiendo una comprensión clara de su disposición y jerarquía.

A continuación, se analizará cada subsistema en detalle mediante un diagrama de componentes por cada subsistema. Se ha optado por un diagrama de componentes debido a que el sistema sigue una arquitectura MERN (MongoDB, Express, React, Node.js) y este tipo de diagrama aporta más valor que un diagrama de clases al reflejar más fielmente la estructura y relaciones entre los distintos módulos del sistema.

Finalmente, se describirán los componentes identificados en el diagrama, explicando su función específica dentro del sistema y las relaciones que mantienen con otros componentes. Esta descripción detallada permitirá entender el papel de cada componente en el funcionamiento global del sistema y cómo colaboran entre sí para cumplir con los objetivos del sistema.

Por último, una vez descritos en detalle cada subsistema, se explicará el diagrama de despliegue. Este diagrama mostrará cómo se distribuyen físicamente los componentes del sistema en el entorno de ejecución, especificando las configuraciones de hardware y software necesarias, así como las interconexiones entre los distintos nodos del sistema.

### 6.5.1. Diagrama de paquetes

El sistema se divide en dos paquetes principales:

- **restapi**: se corresponde con el *backend* del sistema, contiene las clases que implementan la API REST del sistema y la lógica de negocio.
- **frontend**: se corresponde con el *frontend* del sistema, contiene las clases que implementan la interfaz de usuario.

En la Figura 6.7: Diagrama de Paquetes del Sistema se muestra el diagrama de paquetes del sistema.

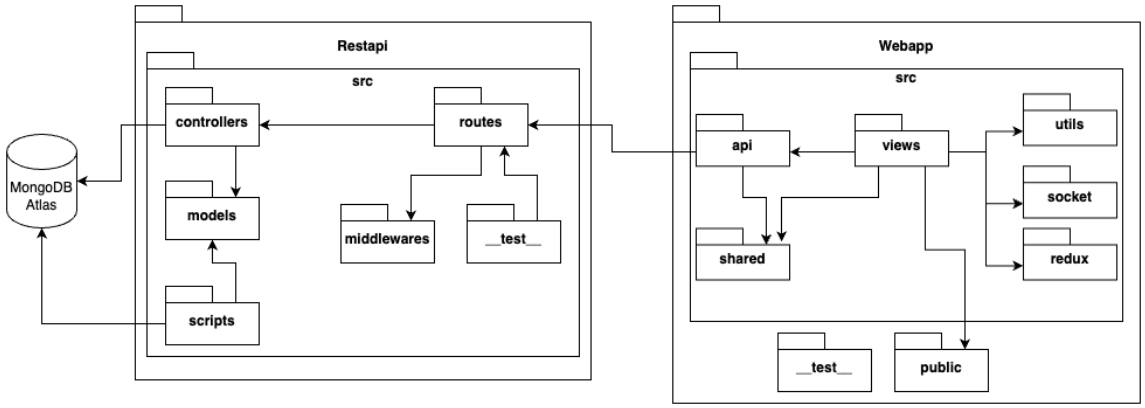


Figura 6.7: Diagrama de Paquetes del Sistema

### 6.5.2. Descripción de los Paquetes

#### 6.5.2.1. restapi

El paquete **restapi** contiene las clases que implementan la API REST del sistema y la lógica de negocio. Este paquete se divide en los siguientes subpaquetes:

- **controllers**: contiene las clases que implementan los controladores de la API REST, se encargan de gestionar las peticiones HTTP y las respuestas. Se comunica con la base de datos.
- **models**: contiene las clases que implementan los modelos de datos del sistema.
- **routes**: contiene las clases que implementan las rutas de la API REST, se encargan de definir las rutas y los métodos HTTP asociados.
- **middlewares**: contiene las clases que implementan los middlewares de la API REST, se encargan de gestionar la autenticación y la autorización de los usuarios.
- **scripts**: contiene las clases que implementan los scripts de inicialización de la base de datos.
- **tests**: contiene las clases que implementan las pruebas unitarias de las clases de los otros subpaquetes.

#### 6.5.2.2. frontend

El paquete **frontend** contiene los archivos que implementan la interfaz de usuario. Este paquete se divide en los siguientes subpaquetes:

- **src**: contiene los archivos que implementan la lógica de la interfaz de usuario.
  - **api**: contiene los archivos que implementan la API del frontend, se encargan de gestionar las peticiones HTTP/HTTPS al backend.
  - **views**: contiene los componentes que implementan las vistas de la interfaz de usuario. A su vez, se divide en los siguientes subpaquetes:
    - **components**: contiene los archivos que implementan los componentes de la interfaz de usuario.

    - **pages**: contiene las archivos que implementan las páginas de la interfaz de usuario.
  - **redux**: contiene las archivos que implementan los estados de Redux.
  - **socket**: contiene las archivos que implementan la conexión con Socket.io.
  - **shared**: contiene las archivos que implementan los tipos de datos compartidos entre las distintas partes de la interfaz de usuario.
  - **utils**: contiene las archivos que implementan utilidades de la interfaz de usuario.
- **public**: contiene los archivos estáticos de la interfaz de usuario.
- **tests**: contiene las clases que implementan las pruebas de la interfaz de usuario.

### 6.5.3. Diagramas de Componentes

En el estándar UML se define como componente:

> Ün Componente representa una parte modular de un sistema que encapsula su contenido y cuya manifestación es reemplazable dentro de su entorno. [...] Un Componente especifica un contrato formal de los servicios que proporciona a sus clientes y aquellos que requiere de otros Componentes o servicios en el sistema en términos de sus Interfaces proporcionadas y requeridas. Un Componente es una unidad sustituible que puede ser reemplazada en tiempo de diseño o en tiempo de ejecución por un Componente que ofrece una funcionalidad equivalente basada en la compatibilidad de sus Interfaces. Siempre que el entorno sea totalmente compatible con las Interfaces proporcionadas y requeridas de un Componente, este podrá interactuar con dicho entorno."
>
> [11, p. 209]

En el contexto de este proyecto, un componente es una parte modular del sistema que encapsula su contenido y, que en un futuro, su funcionalidad podría ser ampliada o reemplazada por otra siempre que cumpla con las interfaces proporcionadas y requeridas.

#### 6.5.3.1. Diagrama de componentes del subsistema restapi

A continuación, se presenta el diagrama de componentes del subsistema **restapi**. En este diagrama se muestran los componentes que forman parte del subsistema y las relaciones entre ellos.

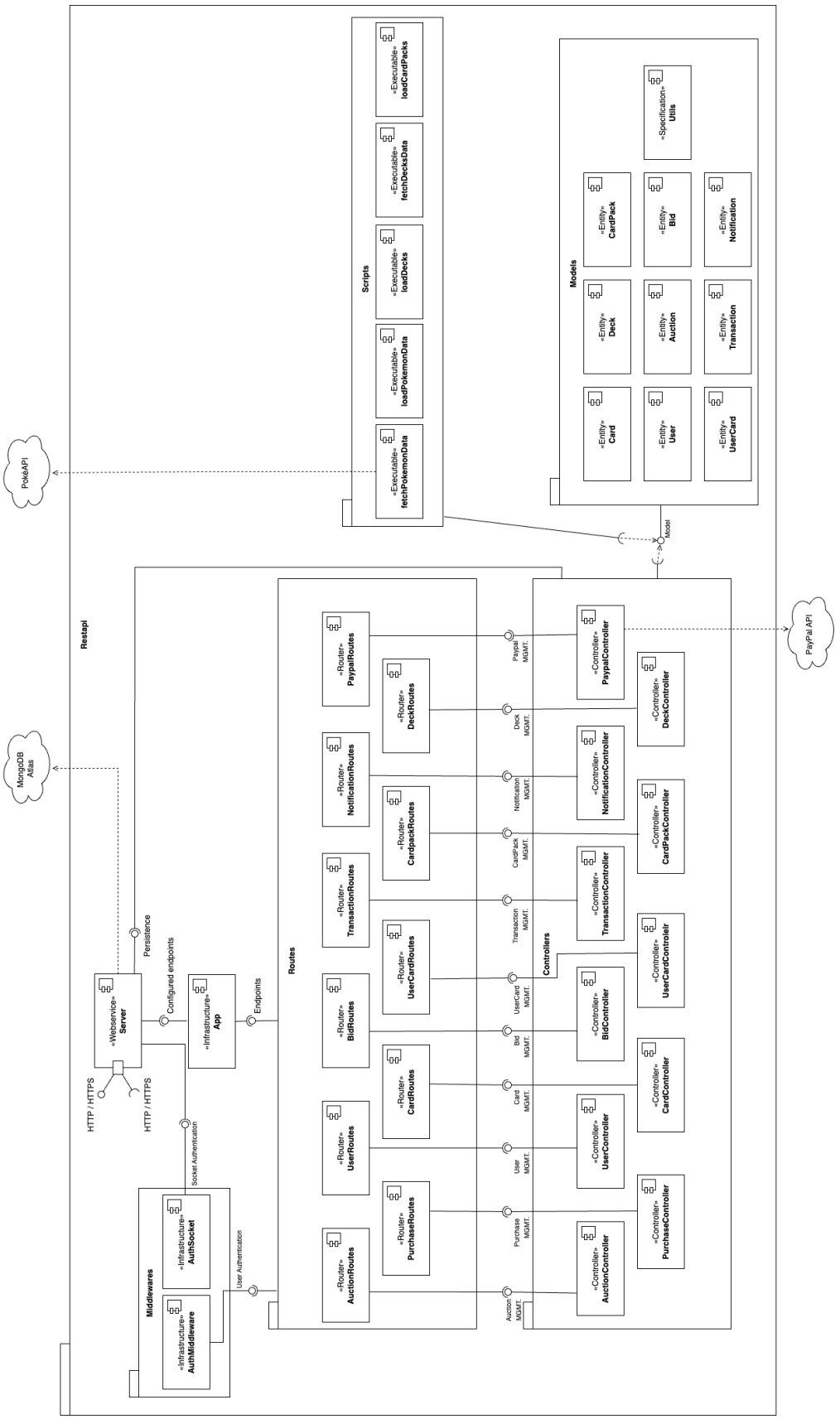


Figura 6.8: Diagrama de componentes del subsistema restapi

Por cada componente identificado en la Figura 6.8: Diagrama de componentes del subsistema restapi, se ha creado una tabla con su descripción detallada, explicando su función específica dentro del sistema y las relaciones que mantiene con otros componentes.

#### *6.5.3.1.1 Descripción de componentes del subsistema restapi. Server y App*

Tabla 6.23: Descripción del componente: Server

| **Componente** | **SERVER** |
|---|---|
| Descripción | Este componente configura y gestiona los servidores HTTP y HTTPS, la conexión a MongoDB, y la gestión de conexiones de sockets mediante Socket.IO. También incluye la configuración de variables de entorno y el manejo de errores. |
| Métodos | ▪ **config()**: void, configura las variables de entorno.<br>▪ **createServers()**: void, crea y configura los servidores HTTP y HTTPS.<br>▪ **connectToDatabase()**: void, conecta a la base de datos MongoDB.<br>▪ **startServers()**: void, inicia los servidores HTTP y HTTPS.<br>▪ **setupSocketIO()**: void, configura el middleware de autenticación de sockets y maneja eventos de conexión y desconexión.<br>▪ **closeServer()**: Promise<void>, cierra los servidores HTTP, HTTPS y la conexión a la base de datos. |
| Interfaces requeridas | ▪ **App**: Usa el componente App, que configura las rutas de la API REST.<br>▪ **Socket Authentication**: Usa el middleware de autenticación de sockets.<br>▪ **HTTP/HTTPS**: Para la conexión a los servidores HTTP y HTTPS. |
| Interfaces proporcionadas | ▪ **HTTP/HTTPS**: Proporciona la conexión a los servidores HTTP y HTTPS. |

Tabla 6.24: Descripción del componente: App

| **Componente** | **APP** |
|---|---|
| Descripción | Este componente configura y gestiona las políticas CORS, las rutas de la API y el middleware para el manejo de errores. |

Continúa en la siguiente página...

Tabla 6.24 Descripción del componente: App – continuación de la página anterior

| Componente | APP |
|---|---|
| Métodos | ▪ **use()**: void, permite configurar las rutas y los middlewares de la aplicación.<br>▪ **listen()**: void, inicia el servidor en el puerto especificado.<br>▪ **errorHandler()**: void, middleware para manejar errores en la aplicación. |
| Interfaces requeridas | ▪ **Endpoints**: Utiliza todas las rutas de la API REST, definidas en el paquete *routes*. Estas son:<br>• **AuctionRouter**: Rutas que gestionan las subastas.<br>• **BidRouter**: Rutas que gestionan las pujas.<br>• **CardPackRouter**: Rutas que gestionan los sobres de cartas.<br>• **CardRouter**: Rutas que gestionan las cartas.<br>• **DeckRouter**: Rutas que gestionan los mazos de cartas.<br>• **NotificationRouter**: Rutas que gestionan las notificaciones.<br>• **PaypalRouter**: Rutas que gestionan las transacciones de PayPal.<br>• **PurchasesRouter**: Rutas que gestionan las compras.<br>• **TransactionRouter**: Rutas que gestionan las transacciones propias de la aplicación.<br>• **UserCardRouter**: Rutas que gestionan las cartas de usuario.<br>• **UserRouter**: Rutas que gestionan los usuarios. |
| Interfaces proporcionadas | ▪ **Configured endpoints**: Proporciona la aplicación de Express con las rutas y middlewares configurados. |

#### *6.5.3.1.2 Descripción de componentes del subsistema restapi. Paquete middlewares*

Tabla 6.25: Descripción del componente: AuthMiddleware

| Componente | AUTHMIDDLEWARE |
|---|---|
| Descripción | Este componente proporciona middleware para la autenticación y autorización de usuarios mediante tokens JWT. Incluye la verificación de tokens y la verificación de roles de administrador. |

Continúa en la siguiente página...

Tabla 6.25 Descripción del componente: AuthMiddleware – continuación de la página anterior

| Componente | AUTHMIDDLEWARE |
|---|---|
| Métodos | ▪ **auth(req: Request, res: Response, next: any)**: void, middleware para verificar la autenticidad del token JWT en las peticiones.<br>▪ **verifyAdmin(req: Request, res: Response, next: any)**: void, middleware para verificar que el usuario tiene rol de administrador. |
| Interfaces requeridas | |
| Interfaces proporcionadas | ▪ **User Authentication**: Proporciona middleware para la autenticación de usuarios, verificando la validez del token JWT y, ofreciendo la posibilidad de verificar roles de administrador. |

Tabla 6.26: Descripción del componente: AuthSocket

| Componente | AUTHSOCKET |
|---|---|
| Descripción | Este componente proporciona el middleware para la autenticación de conexiones de sockets mediante tokens JWT. Verifica la validez y el formato del token proporcionado en el handshake de la conexión del socket. |
| Métodos | ▪ **authSocket(socket: Socket, next: (err?: Error) =>void)**: void, middleware para verificar la autenticidad del token JWT en las conexiones de sockets. |
| Interfaces requeridas | |
| Interfaces proporcionadas | ▪ **Socket Authentication**: Proporciona middleware para la autenticación de conexiones de sockets, verificando la validez del token JWT. |

#### 6.5.3.2. Descripción de componentes del subsistema restapi. Paquete *routes*

En el diagrama se muestra una relación de dependencia entre *App* y los componentes del paquete *routes*. En la práctica, cada componente del paquete *routes* proporciona sus rutas a *App* a través su propia instancia de *Router* de Express. Se ha decidido simplificar la representación para facilitar la comprensión del diagrama, dado que todos los componentes del paquete *routes* mantienen la misma relación con *App*.

Tabla 6.27: Descripción del componente: AuctionRouter

| **Componente** | **AUCTIONROUTER** |
|---|---|
| Descripción | Este componente configura y gestiona las rutas relacionadas con las subastas en la aplicación Express. Incluye la autenticación mediante middleware y validaciones para las peticiones. |
| Atributos | ▪ **auctionRouter**: Router, instancia del enrutador de Express para las subastas. |
| Métodos | ▪ **getAuctions(req: Request, res: Response)**: void, maneja la obtención de todas las subastas.<br>▪ **getAuction(req: Request, res: Response)**: void, maneja la obtención de una subasta por su ID.<br>▪ **getActiveAuctions(req: Request, res: Response)**: void, maneja la obtención de todas las subastas activas.<br>▪ **getActiveAuctionsByUser(req: Request, res: Response)**: void, maneja la obtención de todas las subastas activas de un usuario.<br>▪ **putUserCardUpForAuction(req: Request, res: Response)**: void, maneja la puesta en subasta de una carta de usuario.<br>▪ **withdrawnUserCardFromAuction(req: Request, res: Response)**: void, maneja la retirada de una carta de usuario de una subasta.<br>▪ **checkAllActiveAuctions(req: Request, res: Response)**: void, verifica todas las subastas activas y actualiza su estado si es necesario. |
| Interfaces requeridas | ▪ **User Authentication**: Middleware para la autenticación de usuarios.<br>▪ **Auction MGMT.**: Utiliza los métodos definidos en el controlador de subastas, *AuctionController*. |
| Interfaces proporcionadas | ▪ **Auction Router**: Proporciona las rutas para la gestión de subastas. |

Tabla 6.28: Descripción del componente: BidRouter

| **Componente** | **BIDROUTER** |
|---|---|
| Descripción | Este componente configura y gestiona las rutas relacionadas con las pujas en la aplicación Express. Incluye la autenticación mediante middleware y validaciones para las peticiones. |

Continúa en la siguiente página...

Tabla 6.28 Descripción del componente: BidRouter – continuación de la página anterior

| Componente | BIDROUTER |
|---|---|
| Atributos | ▪ **bidRouter**: Router, instancia del enrutador de Express para las pujas. |
| Métodos | ▪ **getBidById(req: Request, res: Response)**: void, maneja la obtención de una puja por su ID.<br>▪ **createBid(req: Request, res: Response)**: void, maneja la creación de una nueva puja.<br>▪ **getActiveBidsByUser(req: Request, res: Response)**: void, maneja la obtención de todas las pujas activas de un usuario.<br>▪ **withdrawBid(req: Request, res: Response)**: void, maneja la retirada de una puja. |
| Relaciones | ▪ **AuthMiddleware**: Middleware para la autenticación de usuarios.<br>▪ **BidController**: Importa y utiliza métodos del controlador de pujas. |
| Interfaces requeridas | ▪ **User Authentication**: Middleware para la autenticación de usuarios.<br>▪ **Bid MGMT.**: Utiliza los métodos definidos en el controlador de pujas, *BidController*. |
| Interfaces proporcionadas | ▪ **Bid Router**: Proporciona las rutas para la gestión de pujas. |

#### *6.5.3.2.1 Descripción del componente: CardPackRouter*

Tabla 6.29: Descripción del componente: CardPackRouter

| Componente | CARDPACKROUTER |
|---|---|
| Descripción | Este componente configura y gestiona las rutas relacionadas con los sobres de cartas en la aplicación Express. Incluye la autenticación mediante middleware. |
| Métodos | ▪ **getCardPacks(req: Request, res: Response)**: void, maneja la obtención de todos los sobres de cartas. |

Continúa en la siguiente página...

Tabla 6.29 Descripción del componente: CardPackRouter – continuación de la página anterior

| Componente | CARDPACKROUTER |
|---|---|
| Interfaces requeridas | ▪ **User Authentication**: Middleware para la autenticación de usuarios.<br>▪ **CardPack MGMT.**: Utiliza los métodos definidos en el controlador de sobres de cartas, *CardPackController*. |
| Interfaces proporcionadas | ▪ **CardPack Router**: Proporciona las rutas para la gestión de sobres de cartas. |

Tabla 6.30: Descripción del componente: CardRouter

| Componente | CARDROUTER |
|---|---|
| Descripción | Esta clase configura y gestiona las rutas relacionadas con las cartas en la aplicación Express. Incluye la autenticación mediante middleware y validaciones para las peticiones. |
| Métodos | ▪ **getCard(req: Request, res: Response)**: void, maneja la obtención de una carta por su ID. |
| Interfaces requeridas | ▪ **User Authentication**: Middleware para la autenticación de usuarios.<br>▪ **Card MGMT.**: Utiliza los métodos definidos en el controlador de cartas, *CardController*. |
| Interfaces proporcionadas | ▪ **Card Router**: Proporciona las rutas para la gestión de cartas. |

Tabla 6.31: Descripción del componente: NotificationRouter

| Componente | NOTIFICATIONROUTER |
|---|---|
| Descripción | Este componente configura y gestiona las rutas relacionadas con las notificaciones en la aplicación Express. Incluye la autenticación mediante middleware y validaciones para las peticiones. |

Continúa en la siguiente página...

Tabla 6.31 Descripción del componente: NotificationRouter – continuación de la página anterior

| Componente | NOTIFICATIONROUTER |
|---|---|
| Atributos | ▪ **notificationRouter**: Router, instancia del enrutador de Express para las notificaciones. |
| Métodos | ▪ **getNotifications(req: Request, res: Response)**: void, maneja la obtención de todas las notificaciones de un usuario.<br>▪ **markAsRead(req: Request, res: Response)**: void, maneja el marcado de una notificación como leída.<br>▪ **markAllAsRead(req: Request, res: Response)**: void, maneja el marcado de todas las notificaciones de un usuario como leídas.<br>▪ **hasUnreadNotifications(req: Request, res: Response)**: void, verifica si un usuario tiene notificaciones no leídas. |
| Interfaces requeridas | ▪ **User Authentication**: Middleware para la autenticación de usuarios.<br>▪ **Notification MGMT.**: Utiliza los métodos definidos en el controlador de notificaciones, *NotificationController*. |
| Interfaces proporcionadas | ▪ **Notification Router**: Proporciona las rutas para la gestión de notificaciones. |

Tabla 6.32: Descripción del componente: PaypalRouter

| Componente | PAYPALROUTER |
|---|---|
| Descripción | Este componente configura y gestiona las rutas relacionadas con las órdenes de PayPal en la aplicación Express. Incluye validaciones para las peticiones y manejo de errores. |
| Métodos | ▪ **createOrder(req: Request, res: Response)**: void, maneja la creación de una nueva orden de PayPal.<br>▪ **updateOrder(req: Request, res: Response)**: void, maneja la actualización del saldo de un usuario después de completar un pago. |

Continúa en la siguiente página...

Tabla 6.32 Descripción del componente: PaypalRouter – continuación de la página anterior

| Componente | PAYPALROUTER |
|---|---|
| Interfaces requeridas | ▪ **Paypal MGMT.**: Utiliza los métodos definidos en el controlador de PayPal, *PaypalController*. |
| Interfaces proporcionadas | ▪ **Paypal Router**: Proporciona las rutas para la gestión de órdenes de PayPal. |

#### *6.5.3.2.2 Descripción de componentes del subsistema restapi. Paquete controllers*

En el diagrama se muestra una relación de dependencia entre los componentes del paquete *controllers* y los componentes del paquete *models*. En la práctica, cada componente del paquete *controllers* se puede comunicar con cualquier componente del paquete *models* para realizar operaciones de lectura y escritura en la base de datos.

Tabla 6.33: Descripción del componente: AuctionController

| Componente | AUCTIONCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con las subastas en la aplicación Express, incluyendo la recuperación, creación y actualización de subastas, así como la gestión de pujas y la transferencia de cartas. |

Continúa en la siguiente página...

Tabla 6.33 Descripción del componente: AuctionController – continuación de la página anterior

| Componente | AUCTIONCONTROLLER |
|---|---|
| Métodos | ▪ **getAuctions(req: Request, res: Response)**: void, recupera todas las subastas activas disponibles en la base de datos.<br>▪ **getAuction(req: Request, res: Response)**: void, recupera una subasta específica por su ID.<br>▪ **getActiveAuctions(req: Request, res: Response)**: void, recupera todas las subastas activas, excluyendo las del usuario actual.<br>▪ **getActiveAuctionsByUser(req: Request, res: Response)**: void, recupera todas las subastas activas asociadas a un usuario específico.<br>▪ **putUserCardUpForAuction(req: Request, res: Response)**: void, pone una carta en subasta, realizando una serie de validaciones y creando registros asociados.<br>▪ **withdrawnUserCardFromAuction(req: Request, res: Response)**: void, retira una carta de la subasta, actualizando su estado y cancelando la subasta.<br>▪ **checkAllActiveAuctions(req: Request, res: Response)**: void, verifica todas las subastas activas y cierra aquellas que han finalizado, determinando los ganadores.<br>▪ **checkWinnerBid(auction: IAuction, session: any)**: Promise, verifica el ganador de una subasta y procesa la transferencia de la carta al mismo.<br>▪ **transferCard(auction: IAuction, bid: IBid, session: any)**: Promise, transfiere una carta de un usuario vendedor a un usuario comprador, registrando las transacciones de venta y compra. |
| Interfaces requeridas | ▪ **Auction**: Utiliza el modelo de datos de subasta.<br>▪ **Bid**: Utiliza el modelo de datos de puja.<br>▪ **Notification**: Utiliza el modelo de datos de notificación.<br>▪ **Transaction**: Utiliza el modelo de datos de transacción.<br>▪ **User**: Utiliza el modelo de datos de usuario.<br>▪ **UserCard**: Utiliza el modelo de datos de cartas de usuario. |
| Interfaces proporcionadas | ▪ **Auction Controller**: Proporciona las funcionalidades de gestión de subastas. |

Tabla 6.34: Descripción del componente: BidController

| Componente | BIDCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con las pujas en subastas de la aplicación Express, incluyendo la creación, recuperación y retirada de pujas. |
| Métodos | ▪ **createBid(req: Request, res: Response)**: void, realiza una puja por una carta en una subasta.<br>▪ **getActiveBidsByUser(req: Request, res: Response)**: void, obtiene todas las pujas activas realizadas por un usuario específico.<br>▪ **withdrawBid(req: Request, res: Response)**: void, retira una puja realizada por un usuario en una subasta específica.<br>▪ **getBidById(req: Request, res: Response)**: void, recupera una puja específica por su identificador. |
| Interfaces requeridas | ▪ **Auction**: Utiliza el modelo de datos de subasta.<br>▪ **Bid**: Utiliza el modelo de datos de puja.<br>▪ **Transaction**: Utiliza el modelo de datos de transacción.<br>▪ **User**: Utiliza el modelo de datos de usuario. |
| Interfaces proporcionadas | ▪ **Bid Controller**: Proporciona las funcionalidades de gestión de pujas. |

Tabla 6.35: Descripción del componente: CardController

| Componente | CARDCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con las cartas en la aplicación Express, incluyendo la obtención de cartas por ID. |
| Métodos | ▪ **getCard(req: Request, res: Response)**: void, obtiene una carta por su ID.<br>▪ **getCardById(id: any, session: ClientSession)**: Promise<ICard \| null>, obtiene una carta por su ID de tipo *mongoose.Types.ObjectId* en una sesión de base de datos. |
| Interfaces requeridas | ▪ **Card**: Utiliza el modelo de datos de carta. |

Continúa en la siguiente página...

Tabla 6.35 Descripción del componente: CardController – continuación de la página anterior

| Componente | CARDCONTROLLER |
|---|---|
| Interfaces proporcionadas | ▪ **Card Controller**: Proporciona las funcionalidades de gestión de cartas. |

Tabla 6.36: Descripción del componente: CardPackController

| Componente | CARDPACKCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con los sobres de cartas en la aplicación Express, incluyendo la obtención de todos los sobres de cartas disponibles. |
| Métodos | ▪ **getCardPacks(req: Request, res: Response)**: void, obtiene todos los sobres de cartas registrados en la base de datos. |
| Interfaces requeridas | ▪ **CardPack**: Utiliza el modelo de datos de sobres de cartas. |
| Interfaces proporcionadas | ▪ **CardPackController**: Proporciona las funcionalidades de gestión de sobres de cartas. |

Tabla 6.37: Descripción del componente: DeckController

| Componente | DECKCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con los mazos de cartas en la aplicación Express, incluyendo la obtención de todos los mazos disponibles y la recuperación de un mazo específico por su ID. |
| Métodos | ▪ **getDecks(req: Request, res: Response)**: void, obtiene todos los mazos de cartas registrados en la base de datos.<br>▪ **getDeck(req: Request, res: Response)**: void, obtiene un mazo de cartas por su ID.<br>▪ **getDeckByDeckId(deckId: string, session: ClientSession)**: Promise<IDeck \| null>, obtiene las cartas de un mazo por su ID en una sesión de base de datos. |

Continúa en la siguiente página...

Tabla 6.37 Descripción del componente: DeckController – continuación de la página anterior

| Componente | DECKCONTROLLER |
|---|---|
| Interfaces requeridas | ▪ **Deck**: Utiliza el modelo de datos de mazo de cartas. |
| Interfaces proporcionadas | ▪ **Deck Controller**: Proporciona las funcionalidades de gestión de mazos de cartas. |

Tabla 6.38: Descripción del componente: NotificationController

| Componente | NOTIFICATIONCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con las notificaciones en la aplicación Express, incluyendo la recuperación de notificaciones, el marcado como leídas y el envío de notificaciones en tiempo real. |
| Métodos | ▪ **getNotifications(req: Request, res: Response)**: void, recupera el histórico de notificaciones de un usuario específico.<br>▪ **markAsRead(req: Request, res: Response)**: void, marca una notificación específica como leída.<br>▪ **markAllAsRead(req: Request, res: Response)**: void, marca todas las notificaciones de un usuario como leídas.<br>▪ **hasUnreadNotifications(req: Request, res: Response)**: void, comprueba si el usuario tiene notificaciones sin leer.<br>▪ **sendNotification(notification: INotification)**: void, envía una notificación a un usuario específico.<br>▪ **sendRealTimeNotification(notification: INotification)**: void, envía una notificación en tiempo real. |
| Interfaces requeridas | ▪ **Notification**: Utiliza el modelo de datos de notificación.<br>▪ **User**: Utiliza el modelo de datos de usuario. |
| Interfaces proporcionadas | ▪ **Notification Controller**: Proporciona las funcionalidades de gestión de notificaciones. |

Tabla 6.39: Descripción del componente: PaypalController

| Componente | PAYPALCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con la integración de la API de PayPal en la aplicación Express, incluyendo la generación de tokens de acceso y la creación de órdenes de pago. |
| Métodos | ▪ **generateAccessToken()**: Promisestring , genera un token de acceso a la API de PayPal.<br>▪ **handleResponse(response: any)**: Promise{jsonResponse: any, httpStatusCode: number} , maneja la respuesta de la API de PayPal.<br>▪ **createOrder(username: string, balance: number, total: number)**: Promise{jsonResponse: any, httpStatusCode: number} , crea una orden de PayPal para recargar el saldo del usuario. |
| Interfaces requeridas | ▪ **PayPal API**: Utiliza la API de PayPal para la integración de pagos.<br>▪ **User**: Utiliza el modelo de datos de usuario para la actualización del saldo. |
| Interfaces proporcionadas | ▪ **Paypal Controller**: Proporciona las funcionalidades de integración con PayPal. |

Tabla 6.40: Descripción del componente: PurchaseController

| Componente | PURCHASECONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con la compra de paquetes de cartas en la aplicación Express, incluyendo la verificación de disponibilidad, generación de cartas y actualización del saldo del usuario. |
| Métodos | ▪ **purchaseCardPack(req: Request, res: Response)**: Promisevoid , permite comprar un paquete de cartas.<br>▪ **generateCards(deckId: string, quantity: number, session: ClientSession)**: PromiseICard[] , genera cartas aleatorias de un mazo. |

Continúa en la siguiente página...

Tabla 6.40 Descripción del componente: PurchaseController – continuación de la página anterior

| Componente | PURCHASECONTROLLER |
|---|---|
| Interfaces requeridas | ▪ **Card**: Utiliza el modelo de datos de carta.<br>▪ **CardPack**: Utiliza el modelo de datos de paquete de cartas.<br>▪ **Deck**: Utiliza el modelo de datos de mazo de cartas.<br>▪ **Transaction**: Utiliza el modelo de datos de transacción.<br>▪ **User**: Utiliza el modelo de datos de usuario.<br>▪ **UserCard**: Utiliza el modelo de datos de carta de usuario. |
| Interfaces proporcionadas | ▪ **Purchase Controller**: Proporciona las funcionalidades de compra de paquetes de cartas. |

Tabla 6.41: Descripción del componente: TransactionController

| Componente | TRANSACTIONCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con las transacciones en la aplicación Express, incluyendo la obtención de todas las transacciones, transacciones por nombre de usuario y transacciones por ID de carta de usuario. |
| Métodos | ▪ **getTransactions(req: Request, res: Response)**: Promisevoid , obtiene todas las transacciones registradas en la base de datos.<br>▪ **getTransactionsByUsername(req: Request, res: Response)**: Promisevoid , obtiene todas las transacciones de un usuario por su nombre de usuario.<br>▪ **getTransactionsByUserCardId(req: Request, res: Response)**: Promisevoid , obtiene todas las transacciones de cartas para el ID (ObjectId) de la carta de usuario. |
| Interfaces requeridas | ▪ **Transaction**: Utiliza el modelo de datos de transacción. |
| Interfaces proporcionadas | ▪ **Transaction Controller**: Proporciona las funcionalidades de gestión de transacciones. |

Tabla 6.42: Descripción del componente: UserCardController

| Componente | USERCARDCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con las cartas de usuario en la aplicación Express, incluyendo la obtención de todas las cartas de un usuario por su nombre de usuario y la obtención de una carta específica por su ID. |
| Métodos | ▪ **getUserCards(req: Request, res: Response)**: Promisevoid , obtiene todas las cartas de un usuario por su nombre de usuario.<br>▪ **getUserCard(req: Request, res: Response)**: Promisevoid , obtiene una carta de un usuario dado el ID de la carta. |
| Interfaces requeridas | ▪ **UserCard**: Utiliza el modelo de datos de carta de usuario. |
| Interfaces proporcionadas | ▪ **UserCardController**: Proporciona las funcionalidades de gestión de cartas de usuario. |

Tabla 6.43: Descripción del componente: UserController

| Componente | USERCONTROLLER |
|---|---|
| Descripción | Este componente gestiona las operaciones relacionadas con los usuarios en la aplicación Express, incluyendo la creación, autenticación, y actualización de datos del usuario. |
| Métodos | ▪ **createUser(req: Request, res: Response)**: Promisevoid , permite crear un nuevo usuario.<br>▪ **loginUser(req: Request, res: Response)**: Promisevoid , permite iniciar sesión a un usuario.<br>▪ **getUser(req: Request, res: Response)**: Promisevoid , permite obtener un usuario por su nombre de usuario.<br>▪ **updateUserAvatar(req: Request, res: Response)**: Promisevoid , permite actualizar la imagen de perfil de un usuario.<br>▪ **updateUserPassword(req: Request, res: Response)**: Promisevoid , permite actualizar la contraseña de un usuario. |
| Interfaces requeridas | ▪ **User**: Utiliza el modelo de datos de usuario. |

Continúa en la siguiente página...

Tabla 6.43 Descripción del componente: UserController – continuación de la página anterior

| Componente | USERCONTROLLER |
|---|---|
| Interfaces proporcionadas | ▪ **UserController**: Proporciona las funcionalidades de gestión de usuarios. |

#### *6.5.3.2.3 Descripción de componentes del subsistema restapi. Paquete models*

Los componentes del paquete *models* definen los esquemas de datos utilizados para almacenar la información en la base de datos MongoDB. Cada componente del paquete *models* se corresponde con una colección de la base de datos, salvo el componente *utils*, que encapsula funciones auxiliares para la definición de esquemas.

Tabla 6.44: Descripción del componente: Auction

| Componente | AUCTION |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para las subastas en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de las subastas, incluyendo la tarjeta en subasta, el vendedor, el precio inicial, el estado de la subasta y las pujas asociadas. |
| Métodos | |
| Interfaces requeridas | ▪ **Utils**: Utiliza el enum *AuctionStatus* para definir los posibles estados de una subasta. |
| Interfaces proporcionadas | ▪ **Auction**: Proporciona el modelo de datos para gestionar las subastas. |

Tabla 6.45: Descripción del componente: Bid

| Componente | BID |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para las pujas en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de las pujas, incluyendo la subasta, el usuario que realiza la puja, la carta en puja y el estado de la puja. |
| Métodos | |

Continúa en la siguiente página...

Tabla 6.45 Descripción del componente: Bid – continuación de la página anterior

| Componente | BID |
|---|---|
| Interfaces requeridas | ▪ **Utils**: Utiliza el enum *BidStatus* para definir los posibles estados de una puja. |
| Interfaces proporcionadas | ▪ **Bid**: Proporciona el modelo de datos para gestionar las pujas. |

Tabla 6.46: Descripción del componente: Card

| Componente | CARD |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para las cartas en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de las cartas de Pokémon, incluyendo sus atributos, tipo, rareza y estadísticas. |
| Métodos | |
| Interfaces requeridas | ▪ **Utils**: Utiliza los enums *CardRarity*, *PokemonGym* y *PokemonType* para definir las características de las cartas. |
| Interfaces proporcionadas | ▪ **Card**: Proporciona el modelo de datos para gestionar las cartas. |

Tabla 6.47: Descripción del componente: CardPack

| Componente | CARDPACK |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para los sobres cartas en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de los sobres de cartas, incluyendo el nombre, precio y el tipo de cartas que contiene. |
| Métodos | |
| Interfaces requeridas | |
| Interfaces proporcionadas | ▪ **CardPack**: Proporciona el modelo de datos para gestionar los sobres de cartas. |

Tabla 6.48: Descripción del componente: Deck

| **Componente** | **DECK** |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para los mazos en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de los mazos de cartas, incluyendo su ID, nombre, tipo, fecha de publicación y las cartas que contiene. |
| Métodos | |
| Interfaces requeridas | ▪ **Utils**: Utiliza el enum *CardRarity* para definir el tipo de mazo. |
| Interfaces proporcionadas | ▪ **Deck**: Proporciona el modelo de datos para gestionar los mazos. |

Tabla 6.49: Descripción del componente: Notification

| **Componente** | **NOTIFICATION** |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para las notificaciones en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar las notificaciones, incluyendo el usuario asociado, el tipo de notificación, el mensaje, la importancia y si es en tiempo real. |
| Métodos | |
| Interfaces requeridas | ▪ **Utils**: Utiliza los enums *NotificationType* y *NotificationImportance* para definir las características de las notificaciones. |
| Interfaces proporcionadas | ▪ **Notification**: Proporciona el modelo de datos para gestionar las notificaciones. |

Tabla 6.50: Descripción del componente: Transaction

| **Componente** | **TRANSACTION** |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para las transacciones en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar las transacciones realizadas por los usuarios, incluyendo la compra y venta de cartas, las subastas y las pujas. |

Continúa en la siguiente página...

Tabla 6.50 Descripción del componente: Transaction – continuación de la página anterior

| **Componente** | **TRANSACTION** |
|---|---|
| Métodos | |
| Interfaces requeridas | ▪ **Utils**: Utiliza el enum *TransactionConcept* para definir los conceptos de las transacciones. |
| Interfaces proporcionadas | ▪ **Transaction**: Proporciona el modelo de datos para gestionar las transacciones. |

Tabla 6.51: Descripción del componente: User

| **Componente** | **USER** |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para los usuarios en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de los usuarios, incluyendo nombre de usuario, contraseña, imagen de perfil, fecha de nacimiento, saldo y rol de usuario. |
| Métodos | |
| Interfaces requeridas | |
| Interfaces proporcionadas | ▪ **User**: Proporciona el modelo de datos para gestionar los usuarios. |

Tabla 6.52: Descripción del componente: UserCard

| **Componente** | **USERCARD** |
|---|---|
| Descripción | Este componente define el esquema y el modelo de datos para las cartas de usuario en la aplicación utilizando Mongoose. Proporciona una estructura para almacenar y gestionar la información de las cartas que pertenecen a los usuarios, incluyendo la referencia a la carta, el usuario propietario, el estado de la carta y el historial de transacciones. |
| Métodos | |
| Interfaces requeridas | ▪ **Utils**: Utiliza el enum *CardStatus* para definir los posibles estados de una carta de usuario. |

Continúa en la siguiente página...

Tabla 6.52 Descripción del componente: UserCard – continuación de la página anterior

| Componente | USERCARD |
|---|---|
| Interfaces proporcionadas | ▪ **UserCard**: Proporciona el modelo de datos para gestionar las cartas de usuario. |

Tabla 6.53: Descripción del componente: Modelos

| Componente | MODELOS |
|---|---|
| Descripción | Este componente contiene las enumeraciones, validaciones y líderes de gimnasio utilizados en la aplicación. Está compuesto por tres archivos: *Enums.ts*, *validations.ts* y *gymLeaders.ts*.<br>▪ **Enums.ts**: Contiene las enumeraciones para los diferentes tipos de Pokémon, gimnasios, rarezas de cartas y estados de las cartas.<br>▪ **validations.ts**: Define las validaciones para los datos de usuario utilizando la biblioteca Yup, incluyendo restricciones para nombres de usuario, contraseñas y fechas de nacimiento.<br>▪ **gymLeaders.ts**: Proporciona un mapa que asocia nombres de gimnasios con los líderes de Pokémon y una función para encontrar el gimnasio de un Pokémon si es un líder de gimnasio. |
| Métodos | |
| Interfaces requeridas | |
| Interfaces proporcionadas | ▪ **Utils**: Proporciona las enumeraciones, validaciones y la lógica para obtener los líderes de gimnasio utilizados en las cartas de la aplicación. |

#### *6.5.3.2.4 Descripción de componentes del subsistema restapi. Paquete scripts*

Los componentes del paquete *scripts* son archivos ejecutables que contienen funciones auxiliares para la inicialización de la base de datos y la carga de datos de prueba.

Tabla 6.54: Descripción del componente: FetchDeckData

| Componente | FETCHDECKDATA |
|---|---|
| Descripción | Este componente se encarga de recuperar y almacenar datos de mazos en un archivo CSV. Utiliza la biblioteca *csv-writer* para crear y escribir registros en un archivo llamado *decks_data.csv*. |

Continúa en la siguiente página...

Tabla 6.54 Descripción del componente: FetchDeckData – continuación de la página anterior

| Componente | FETCHDECKDATA |
|---|---|
| Métodos | ▪ **fetchAndStoreDecks()**: Promisevoid , genera un mazo de cartas por cada rareza y lo almacena en un archivo CSV. |
| Interfaces requeridas | ▪ **Utils**: Utiliza el enum *CardRarity*, que define la rareza de las cartas. |
| Interfaces proporcionadas | |

Tabla 6.55: Descripción del componente: loadDecks

| Componente | LOADDECKS |
|---|---|
| Descripción | Este componente se encarga de cargar datos de mazos desde un archivo CSV en la base de datos MongoDB. Utiliza las bibliotecas *csv-parser* y *mongoose* para leer el archivo y realizar operaciones en la base de datos. |
| Métodos | ▪ **connectToMongoDB()**: Promisevoid , conecta con la base de datos MongoDB utilizando la URI especificada en las variables de entorno.<br>▪ **loadCSVData(filePath: string)**: Promisevoid , lee los datos del archivo CSV especificado, los procesa y los almacena en la base de datos. |
| Interfaces requeridas | ▪ **Card**: Modelo utilizado para gestionar las cartas en la base de datos.<br>▪ **Deck**: Modelo utilizado para gestionar los mazos en la base de datos. |
| Interfaces proporcionadas | |

Tabla 6.56: Descripción del componente: FetchPokemonData

| Componente | FETCHPOKEMONDATA |
|---|---|
| Descripción | Este componente se encarga de recuperar y almacenar datos de Pokémon en un archivo CSV. Utiliza la biblioteca *axios* para realizar solicitudes a la API de PokeAPI y *csv-writer* para crear y escribir registros en un archivo llamado *cards_data.csv*. |

Continúa en la siguiente página...

Tabla 6.56 Descripción del componente: FetchPokemonData – continuación de la página anterior

| Componente | FETCHPOKEMONDATA |
|---|---|
| Métodos | ▪ **fetchAndStorePokemon()**: Promisevoid , recupera datos de Pokémon desde la PokeAPI y los almacena en un archivo CSV.<br>▪ **getPokemonRarity(n_encounters: number, n_location_area: number, averageMaxChance: number, is_legendary: boolean, is_mythical: boolean)**: string, determina la rareza de un Pokémon basado en diversos factores.<br>▪ **calculatePokemonRarity(n_encounters: number, n_location_area: number, averageMaxChance: number)**: string, calcula la rareza de un Pokémon utilizando criterios arbitrarios.<br>▪ **getCardRarity()**: CardRarity, genera aleatoriamente la rareza de una carta. |
| Interfaces requeridas | ▪ **PokéAPI**: Utiliza la API de PokéAPI para obtener datos de Pokémon.<br>▪ **Utils**: Proporciona la función *findGymByPokemon* para encontrar el gimnasio asociado a un Pokémon y las enumeraciones *PokemonGym*, *PokemonType*, *PokemonRarity* y *CardRarity*. |
| Interfaces proporcionadas | |

Tabla 6.57: Descripción del componente: LoadPokemonData

| Componente | LOADPOKEMONDATA |
|---|---|
| Descripción | Este componente se encarga de cargar datos de Pokémon desde un archivo CSV en la base de datos MongoDB. Utiliza las bibliotecas *csv-parser* para leer el archivo CSV y *mongoose* para interactuar con MongoDB. |
| Métodos | ▪ **connectToMongoDB()**: Promiseboolean , establece una conexión con la base de datos MongoDB.<br>▪ **loadCSVData(filePath: string)**: Promisevoid , carga datos desde un archivo CSV y los inserta en la base de datos MongoDB. |
| Interfaces requeridas | ▪ **Card**: Modelo de datos para las cartas de Pokémon. |
| Interfaces proporcionadas | |

Tabla 6.58: Descripción del componente: LoadCardPacksData

| Componente | LOADCARDPACKSDATA |
|---|---|
| Descripción | Este componente se encarga de cargar datos de paquetes de cartas desde un archivo CSV en la base de datos MongoDB. Utiliza las bibliotecas *csv-parser* para leer el archivo CSV y *mongoose* para interactuar con MongoDB. |
| Métodos | ▪ **connectToMongoDB()**: Promiseboolean , establece una conexión con la base de datos MongoDB.<br>▪ **loadCSVData(filePath: string)**: Promisevoid , carga datos desde un archivo CSV y los inserta en la base de datos MongoDB. |
| Interfaces requeridas | ▪ **CardPack**: Modelo de datos para los paquetes de cartas. |
| Interfaces proporcionadas | |

#### 6.5.3.3. Diagrama de componentes del subsistema webapp

A continuación, se presenta el diagrama de componentes del subsistema **webapp**. En este diagrama se muestran los componentes que forman parte del subsistema. Se han omitido las relaciones entre los componentes para simplificar el diagrama. Las relaciones entre los componentes se explicarán en la descripción de los componentes.

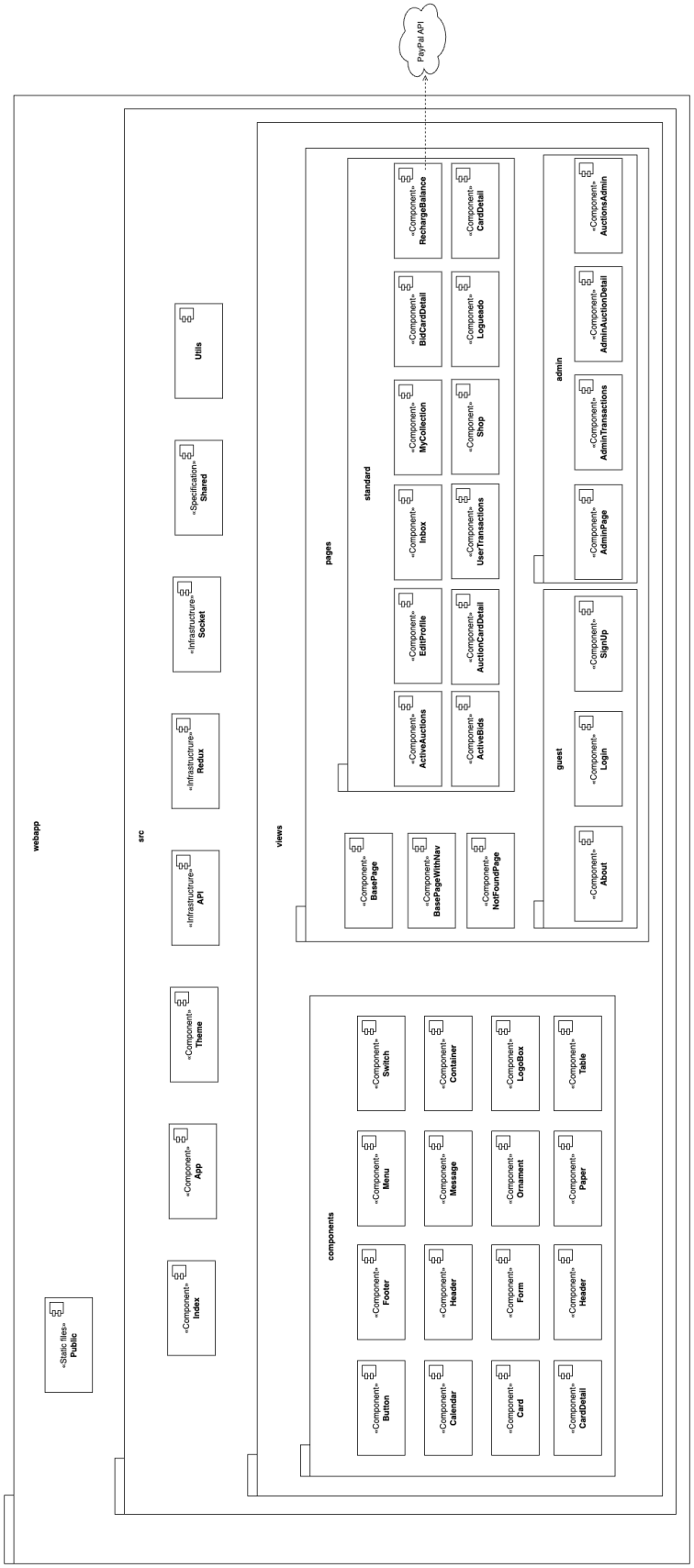


Figura 6.9: Diagrama de componentes del subsistema webapp

#### 6.5.3.4. Descripción de los Componentes del Subsistema webapp

Este apartado expone detalladamente los componentes del subsistema webapp, detallando su funcionalidad y su interrelación dentro de la arquitectura de la aplicación.

##### *6.5.3.4.0 Componente public*

Este componente almacena los archivos estáticos esenciales para la interfaz de usuario, incluyendo recursos gráficos y el archivo *index.html*. Este último establece la estructura fundamental de la aplicación, configurando el entorno inicial en el que se cargarán los demás componentes dinámicos.

##### *6.5.3.4.0 Paquete src*

El núcleo de la lógica de la interfaz de usuario se gestiona dentro de este paquete, donde cada componente tiene una función específica:

- **Index**: Actúa como el punto de entrada de la aplicación, encargado de inicializar el componente principal *App* y de configurar el entorno de ejecución.
- **App**: Es el componente central que gestiona la navegabilidad y el diagrama de flujo de la aplicación. Coordina el renderizado de diferentes componentes basado en la ruta actual y establece conexiones con los paquetes *pages* y *Theme*.
- **Theme**: Define y aplica la configuración visual global de la aplicación, incluyendo esquemas de color y tipografías, para mantener una coherencia estética en toda la interfaz de usuario.
- **API**: Administra todas las interacciones con las APIs, permitiendo la comunicación y el intercambio de datos entre el frontend y el backend.
- **Redux**: Es el gestor de estado de la aplicación, manteniendo y actualizando los estados globales como:
  - **user**: Almacena y gestiona la información del usuario autenticado.
  - **notification**: Controla la presentación y gestión de notificaciones en tiempo real.
  - **update**: Se encarga de las actualizaciones periódicas de los datos de la aplicación.
- **Socket**: Gestiona la comunicación en tiempo real con el servidor a través de Socket.io. Define la funcionalidad para gestionar las notificaciones en tiempo real.
- **Shared**: Define los tipos de datos y estructuras compartidas utilizadas a lo largo de la aplicación para asegurar la consistencia y la eficiencia en el manejo de la información.
- **Utils**: Provee una serie de utilidades y herramientas que facilitan operaciones comunes dentro de la aplicación, tales como validación de formularios, gestión de rutas privadas y funciones de ayuda general.
- **Views**: Incluye la implementación de las interfaces de usuario. Se divide en *components* y *pages* para organizar los elementos visuales de manera coherente y modular.
  - **Components**: Este subpaquete agrupa los componentes reutilizables que conforman la base de las interfaces de usuario en la aplicación:
    - **Button**: Implementa diferentes tipos de botones que se utilizan a través de la aplicación, adaptándose a diversas funciones y estilos según las necesidades de la interfaz.

- **Calendar**: Proporciona una representación visual de un calendario para la selección de fechas.
- **Card**: Define una estructura base para las cartas coleccionables, utilizadas en diferentes contextos de la aplicación.
- **CardDetail**: Muestra información detallada de una carta.
- **Cardpack**: Representa sobres de cartas, es el componente principal para la compra de cartas en la tienda.
- **Container**: Generaliza contenedores que encapsulan diferentes tipos de contenido.
- **Form**: Define los distintos tipos de formularios utilizados en la aplicación, con validación y gestión de eventos integrados.
- **Footer**: Define el pie de página de la aplicación, conteniendo enlaces e información relevante de contacto.
- **Header**: Encabeza las páginas de la aplicación, incluyendo elementos de navegación y acceso al perfil del usuario.
- **LogoBox**: Presenta el logotipo de la aplicación, utilizado generalmente en la cabecera de los formularios de inicio de sesión y registro.
- **Menu**: Organiza las opciones de navegación disponibles para los usuarios, adaptándose a los diferentes roles y estados de autenticación.
- **Messages**: Administra la presentación de mensajes informativos o de error, mejorando la interactividad y experiencia del usuario.
- **Ornament**: Añade elementos decorativos que mejoran la estética sin cargar la funcionalidad principal de los componentes.
- **Paper**: Ofrece un componente estilizado para mostrar contenido en forma de tarjeta elevada, utilizada para destacar información relevante.
- **Switch**: Permite a los usuarios alternar entre el modo claro y oscuro de la aplicación, adaptándose a sus preferencias visuales.
- **Table**: Proporciona estructuras de tabla para la visualización de datos complejos, con funciones de ordenación, paginación y filtrado integradas.

- **Pages**: Detalla las implementaciones específicas de las vistas de la aplicación, que se construyen utilizando los componentes anteriores. Cada página está diseñada para satisfacer las necesidades de diferentes tipos de usuarios y situaciones:
  - **BasePage**: Ofrece una plantilla base para las páginas de la aplicación, garantizando la uniformidad y coherencia en el diseño.
  - **BasePageWithNav**: Similar a BasePage, pero incluye componentes de navegación específicos para facilitar el acceso a otras secciones.
  - **NotFoundPage**: Proporciona una respuesta visual para rutas no encontradas, mejorando la navegación del usuario en casos de error.
  - El subpaquete **guest** contiene páginas accesibles sin necesidad de autenticación:
    - **Home**: La página principal de la aplicación, presentando información general y acceso al inicio de sesión o registro.
    - **Login**: Facilita el inicio de sesión para los usuarios.
    - **SignUp**: Permite a nuevos usuarios registrarse en la aplicación.
    - **About**: Ofrece información sobre la aplicación y sus creadores.
  - El subpaquete **admin** agrupa las páginas destinadas a usuarios con roles de administrador:
    - **AdminPage**: Página principal del panel de administración.

- ◇ **AdminTransactions**: Permite consultar todas las transacciones realizadas en la aplicación por los usuarios.
- ◇ **AdminAuctionDetail**: Detalla las subastas activas y permite su gestión directa.
- ◇ **AuctionsAdmin**: Permite a los administradores consultar y gestionar las subastas activas.

○ El subpaquete **standard**: Para usuarios autenticados sin privilegios administrativos, proporciona acceso a funcionalidades estándar de la aplicación:

- ◇ **Logueado**: Muestra la interfaz principal para usuarios autenticados, ofreciendo un resumen de su actividad y acceso directo a funcionalidades comunes.
- ◇ **EditProfile**: Permite a los usuarios modificar su perfil.
- ◇ **CardDetail**: Muestra la información de una carta de la colección del usuario.
- ◇ **AuctionCardDetail**: Presenta información detallada de una carta en subasta.
- ◇ **BidCardDetail**: Muestra el detalle de una puja realizada por el usuario.
- ◇ **Shop**: Tienda virtual donde los usuarios pueden adquirir sobres de cartas para ampliar su colección.
- ◇ **ActiveAuctions**: Lista las subastas en curso, permitiendo a los usuarios participar en ellas.
- ◇ **UserTransactions**: Resumen de las transacciones realizadas por el usuario, ofreciendo un historial detallado.
- ◇ **InBox**: Gestiona las notificaciones recibidas, asegurando que el usuario esté informado de eventos importantes.
- ◇ **MyCollection**: Permite al usuario visualizar y gestionar su colección de cartas.
- ◇ **RechargeBalance**: Integra métodos para recargar el saldo del usuario, enlazando con sistemas de pago externos como PayPal.

**Nota:** Cada componente en el diagrama puede representar uno o más elementos funcionales dentro de la aplicación. Por ejemplo, en el caso del componente *Button*, en la práctica se implementan varios componentes que se adaptan para cubrir distintas funcionalidades en el sistema.

### 6.5.4. Diagrama de Despliegue

En este apartado se describirá el proceso de despliegue de la aplicación, detallando los componentes y servicios necesarios para su correcto funcionamiento. Se presentará un diagrama de despliegue que ilustrará la arquitectura de la aplicación y la distribución de los componentes en el entorno de ejecución.

Como se ha visto en la sección 6.4. IDENTIFICACIÓN DE SUBSISTEMAS DE ANÁLISIS, la aplicación se divide en dos subsistemas principales: **restapi** y **webapp**. El subsistema **restapi** contiene las clases que implementan la API REST y la lógica de negocio, mientras que el subsistema **webapp** abarca la interfaz de usuario de la aplicación. Estos dos subsistemas serán desplegados en una máquina virtual de Azure, utilizando contenedores Docker para facilitar su despliegue y escalabilidad.

Para desplegar la aplicación, se han seguido los siguientes pasos:

1. **Crear la máquina virtual en Azure**: Se ha creado una máquina virtual en Azure con las características mencionadas anteriormente. Además, es necesario crear un grupo de recursos y una red virtual para la máquina virtual. Las características de la máquina virtual son las siguientes:
   - **Sistema Operativo**: Ubuntu 20.04 LTS.

- **Procesador**: 2 núcleos.
- **Memoria RAM**: 4 GB.
- **Discos de datos**: 4.
- **E/S máxima**: 1280 MB/s.
- **Almacenamiento local**: 8 GB.
- **Red**: IP pública y DNS asociado.

2. **Establecer un nombre de dominio**: Se ha asociado un nombre de dominio a la dirección IP pública de la máquina virtual para facilitar el acceso a la aplicación.

3. **Contenedores Docker**: Se ha utilizado Docker para contenerizar la aplicación y facilitar su despliegue. Para facilitar la tarea se ha configurado integración continua con GitHub Actions, de forma que cada vez que se realiza un *pull-request* a la rama principal del repositorio, se construyen y despliegan los contenedores en la máquina virtual de Azure. Se han creado dos contenedores, uno para el subsistema **restapi** y otro para el subsistema **webapp**. Cada uno de estos expone uno o varios puertos diferentes para la comunicación con el exterior.

4. **Configuración de Nginx como Proxy Inverso**: Se ha configurado Nginx como servidor web inverso para redirigir las peticiones a los contenedores correspondientes. Este paso es necesario para poder garantizar la seguridad de las comunicaciones con HTTPS. La configuración de Nginx incluye:

   - **Obtención de certificado SSL**: Se ha utilizado Certbot para obtener un certificado SSL gratuito de Let's Encrypt y habilitar el protocolo HTTPS en la aplicación.
   - **Redirección de Peticiones**: Configuración de Nginx para dirigir el tráfico HTTP y HTTPS a los contenedores adecuados.

5. **Configuración de la base de datos**: En la aplicación se usa como base de datos Mongo DB Atlas, que es una base de datos en la nube por lo que no es necesario configurarla en la máquina virtual, pero sí es necesario configurar la conexión a la base de datos en la aplicación y permitir el acceso desde la dirección IP de la máquina virtual.

6. **Docker Compose**: Se ha utilizado Docker Compose para orquestar los contenedores de la aplicación y facilitar su despliegue en la máquina virtual de Azure. Se ha creado un archivo *docker-compose.yml* en la máquina virtual que define los servicios de la aplicación y sus configuraciones.

Una vez realizados estos pasos, la aplicación estará desplegada y accesible a través del nombre de dominio asociado a la dirección IP pública de la máquina virtual. Debido a los costos asociados al uso de la máquina virtual en Azure, se ha optado por mantener la máquina virtual apagada cuando no se está utilizando la aplicación.

En la Figura 6.10: Diagrama de Despliegue de la Aplicación se muestra el diagrama de despliegue de la aplicación.

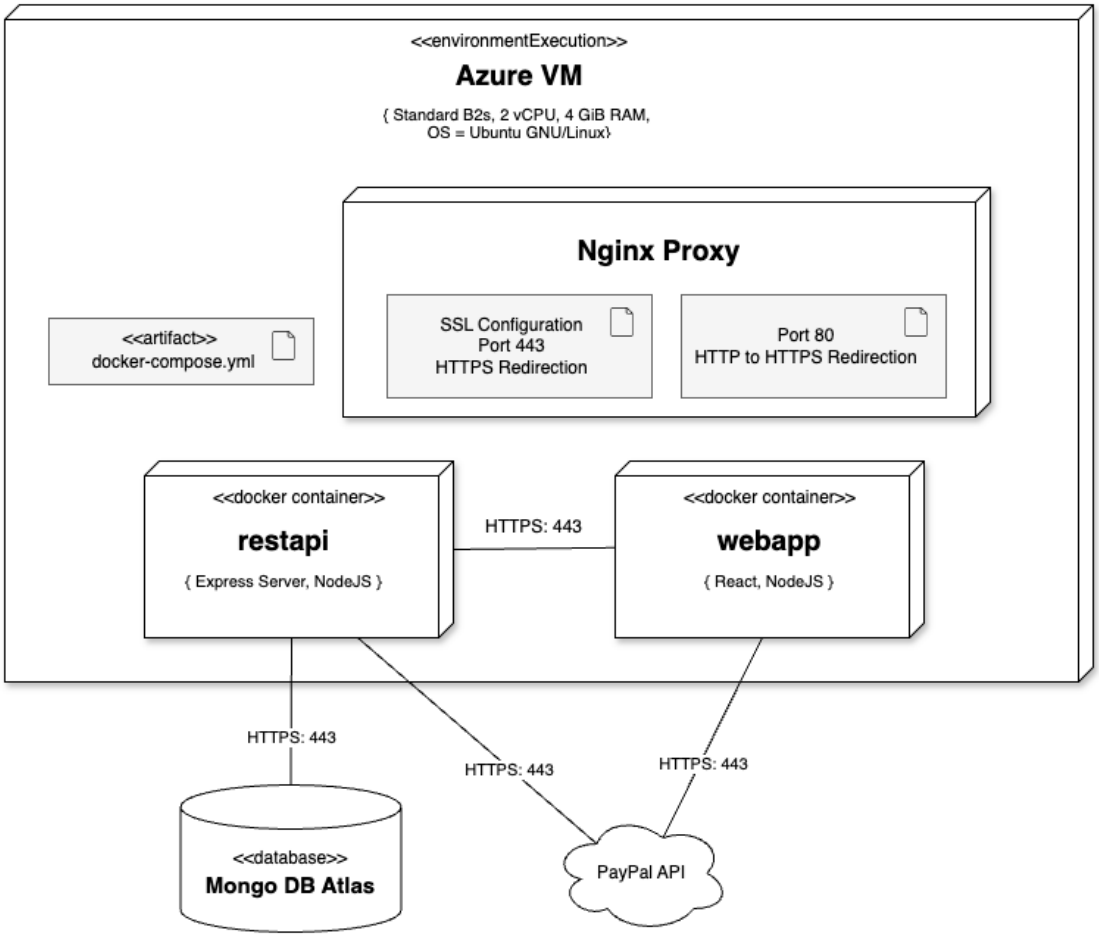


Figura 6.10: Diagrama de Despliegue de la Aplicación

## 6.6. DEFINICIÓN DE INTERFACES DE USUARIO

### 6.6.1. Descripción de la Interfaz

Se ha desarrollado un conjunto de bocetos representativos para cada una de las interfaces principales, con el objetivo de facilitar la comprensión visual y estructural del sistema. Se ha decidido omitir la creación de bocetos para ciertas páginas del sistema, considerando que su simplicidad inherente no justifica una representación gráfica detallada, o debido a que su diseño es análogo al de otras interfaces previamente especificadas.

#### 6.6.1.1. Página de Inicio

La página de inicio es la primera página que se muestra al usuario al acceder al sistema. Esta página se muestra cuando el usuario no ha iniciado sesión, y le permite acceder a las siguientes funcionalidades:

- Iniciar sesión.
- Registrarse.
- Consultar información pública sobre el sistema (Enlaces del pie de página)

En la Figura 6.11: Boceto de la página de inicio se muestra el boceto de la página de inicio.

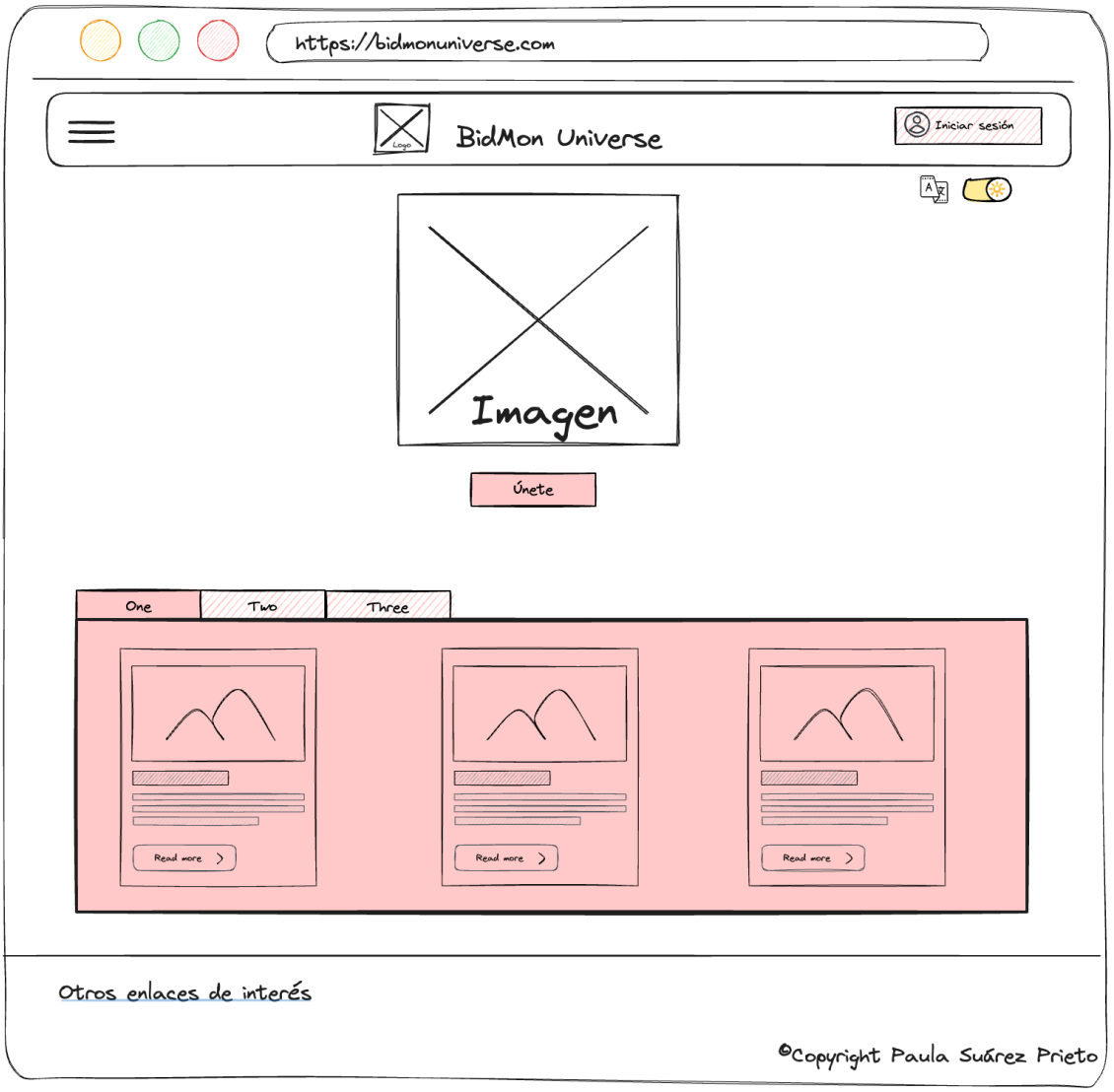


Figura 6.11: Boceto de la página de inicio

#### 6.6.1.2. Página de Inicio de Sesión

La página de inicio de sesión muestra un formulario que permite al usuario ingresar sus credenciales para acceder al sistema o recuperar su contraseña.

En la Figura 6.12: Boceto de la página de inicio de sesión se muestra el boceto de la página de inicio de sesión. En caso de que el usuario introduzca credenciales incorrectas, se mostrará un mensaje de error y se resaltarán los campos correspondientes.

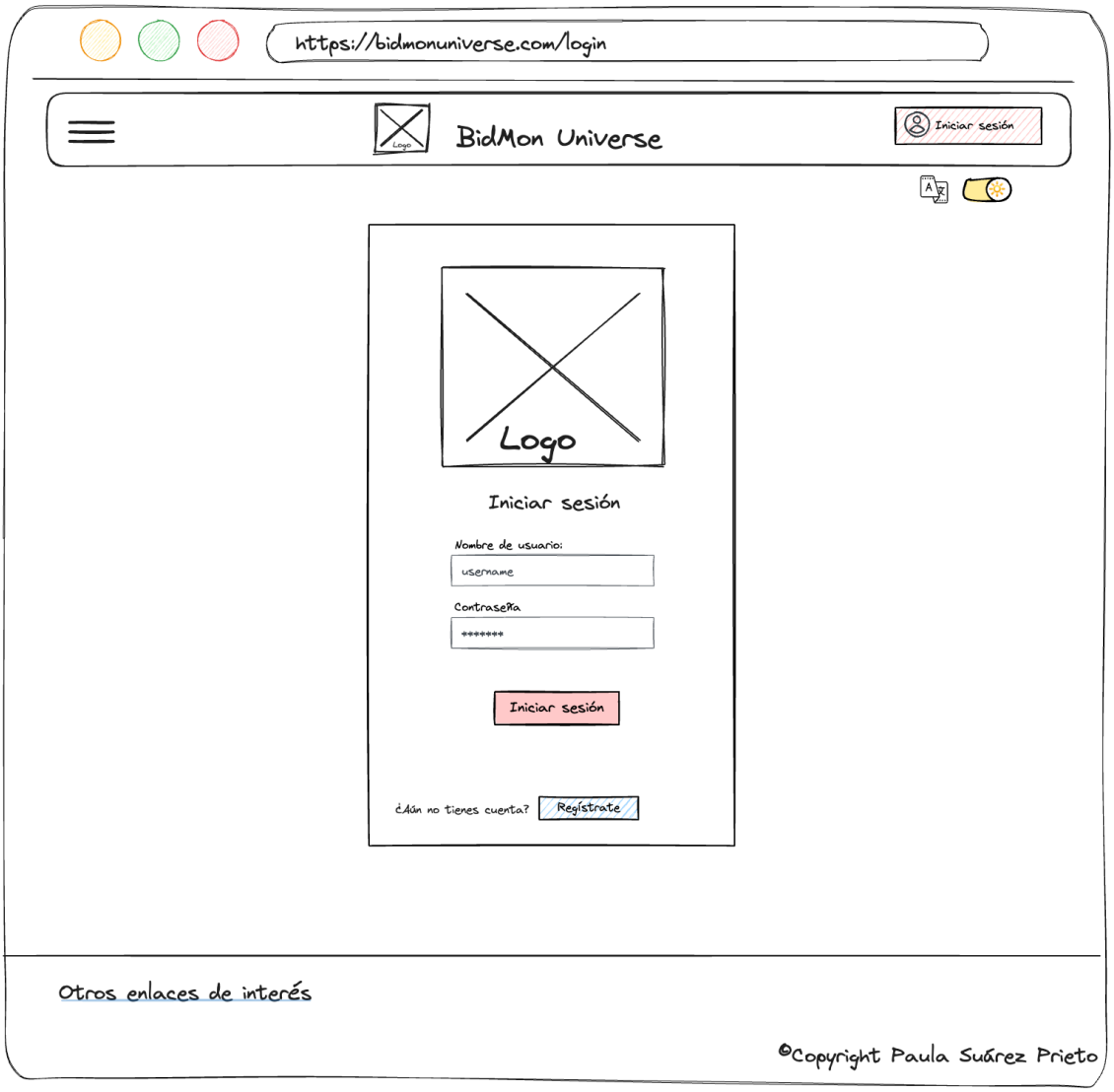


Figura 6.12: Boceto de la página de inicio de sesión

#### 6.6.1.3. Página de Registro

La página de registro muestra un formulario que permite al usuario crear una cuenta en el sistema o iniciar sesión si ya tiene una cuenta. En la Figura 6.13: Boceto de la página de registro se muestra el boceto de la página de registro.

En caso de que los campos del formulario no cumplan con las restricciones especificadas, se mostrará un mensaje de error y se resaltarán los campos correspondientes.

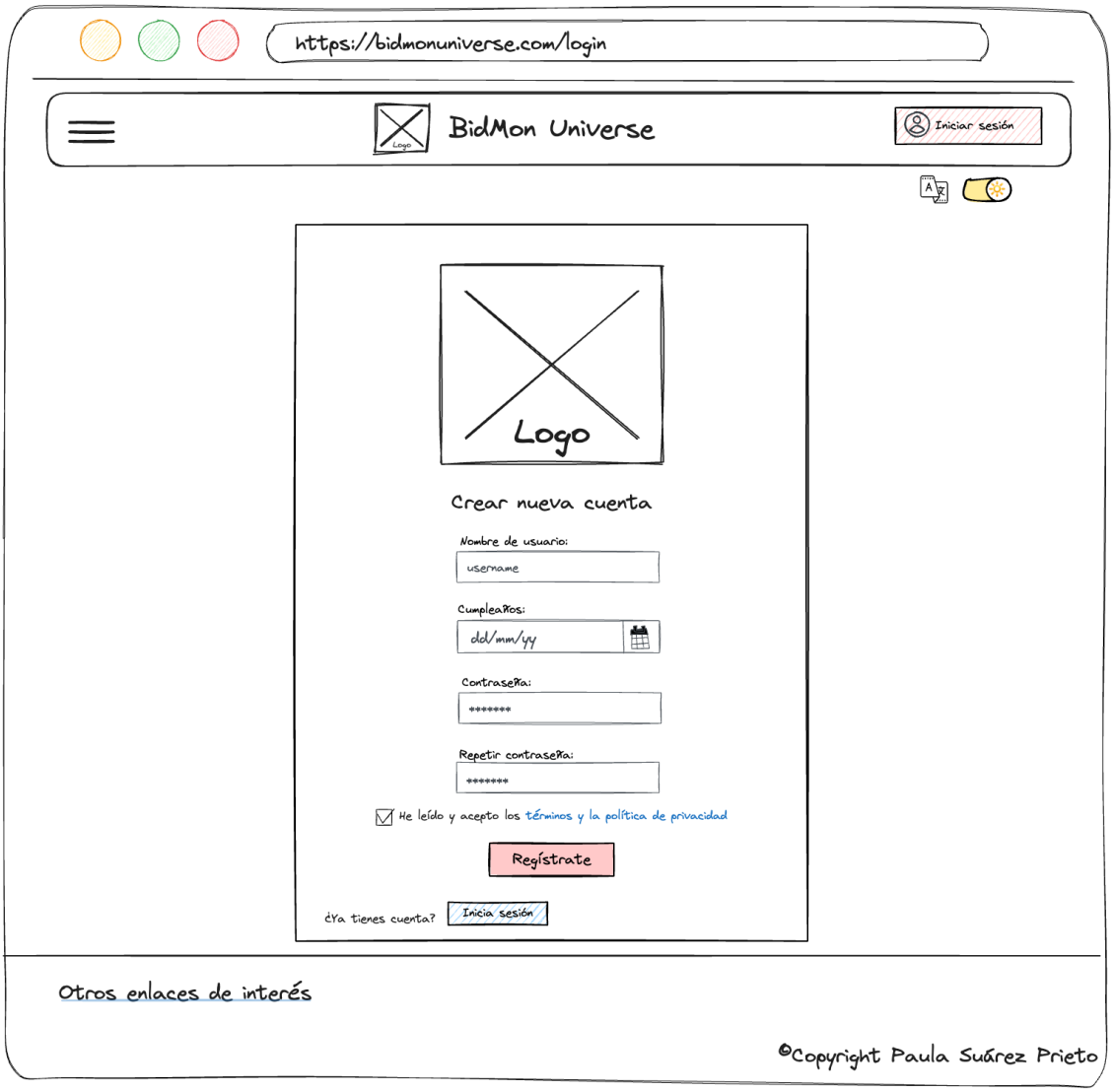


Figura 6.13: Boceto de la página de registro

#### 6.6.1.4. Página inicial de usuario

Una vez que el usuario ha iniciado sesión en la plataforma, se le redirige a la página inicial de usuario. Las páginas a las que tiene acceso un usuario autenticado muestran siempre como cabecera de la página un menú general de navegación que le permite acceder a las diferentes secciones del sistema, el saldo de su cuenta y un menú de usuario que le permite acceder a su perfil y cerrar sesión.

La página inicial de usuario muestra un resumen de la información más relevante para el usuario:

- Menú de acceso rápido a las secciones principales del sistema.
- Muestra las cartas más recientes de su colección con la posibilidad de acceder a la colección completa.
- Muestra las subastas activas iniciadas por el usuario más recientes con la posibilidad de acceder a las subastas completas.

- Muestra las pujas activas más recientes realizadas por el usuario con la posibilidad de acceder a las pujas completas.

En la Figura 6.14: Boceto de la página inicial de usuario se muestra el boceto de la página inicial de usuario.

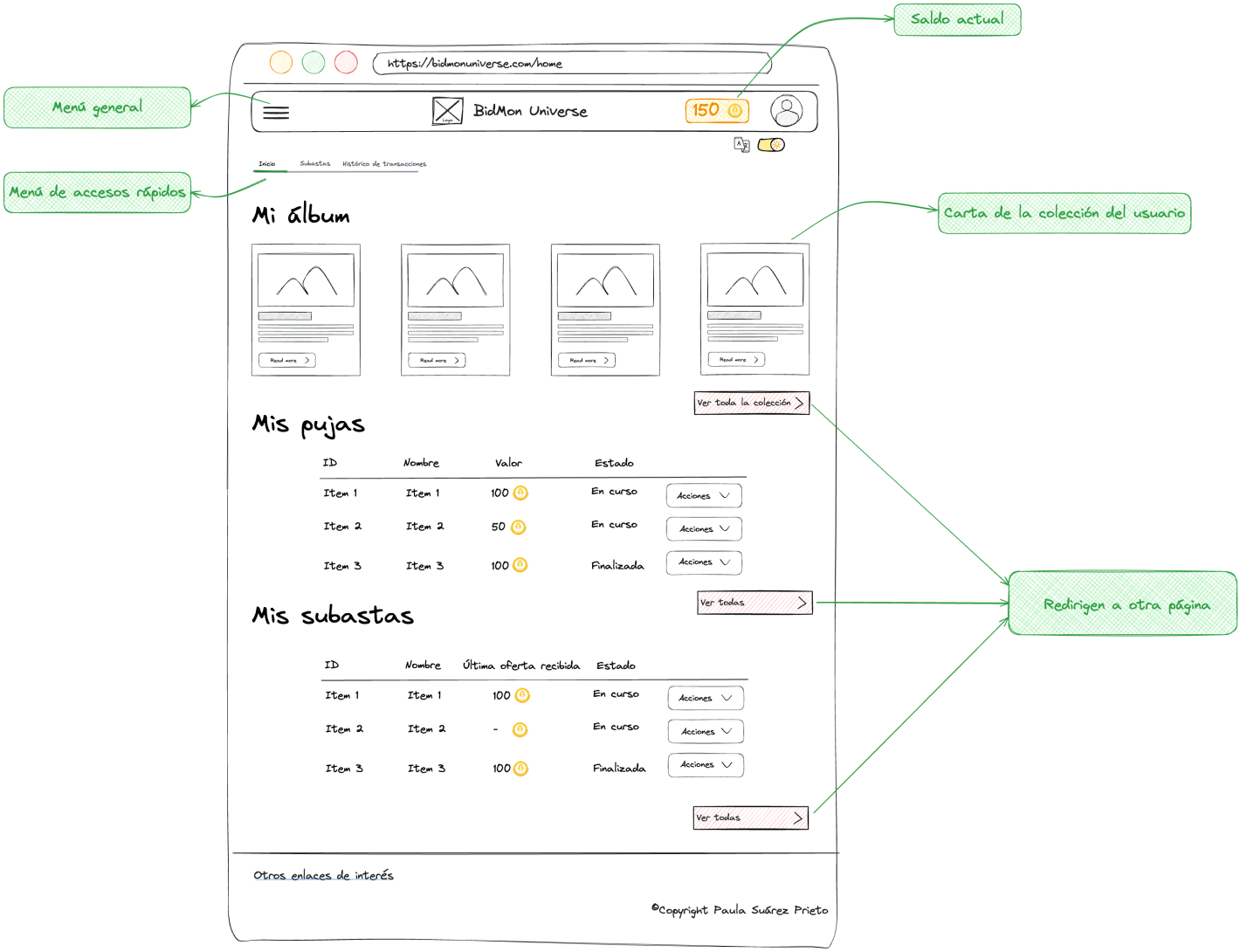


Figura 6.14: Boceto de la página inicial de usuario

#### 6.6.1.5. Página de Colección

La página de colección muestra una lista de las cartas que el usuario ha añadido a su colección. El usuario puede hacer clic en una carta para ver más detalles sobre ella.

En la Figura 6.15: Boceto de la página de colección se muestra el boceto de la página de colección.

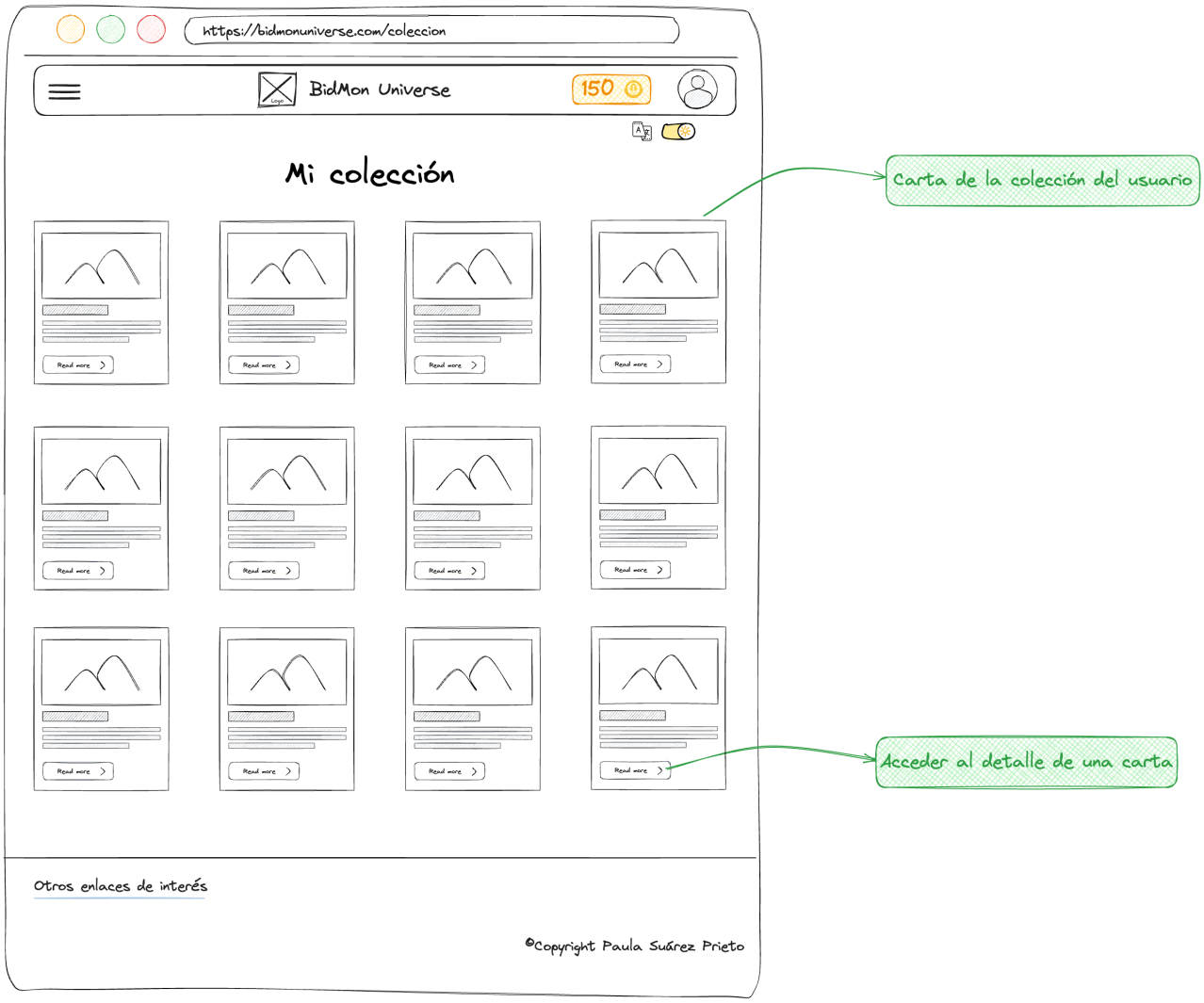


Figura 6.15: Boceto de la página de colección

#### 6.6.1.6. Página de Detalles de Carta

La página de detalles de carta muestra información detallada sobre una carta específica. Permite al usuario ver la imagen de la carta, los datos de la carta, las transacciones de esta y la posibilidad de marcar la carta como destacada o subastarla.

En la Figura 6.16: Boceto de la página de detalles de carta se muestra el boceto de la página de detalles de carta.


Figura 6.16: Boceto de la página de detalles de carta

#### 6.6.1.7. Página de Creación de Subasta

La página de creación de subasta muestra el detalle de la carta que se va a subastar y permite al usuario establecer el precio de salida y la duración de la subasta. Estos valores deben cumplir con las restricciones especificadas y son opcionales, si el usuario no los establece se usarán los valores por defecto y se le informará en la ventana de confirmación.

En la Figura 6.17: Boceto de la página de creación de subasta, se muestra el boceto de la página de creación de subasta.

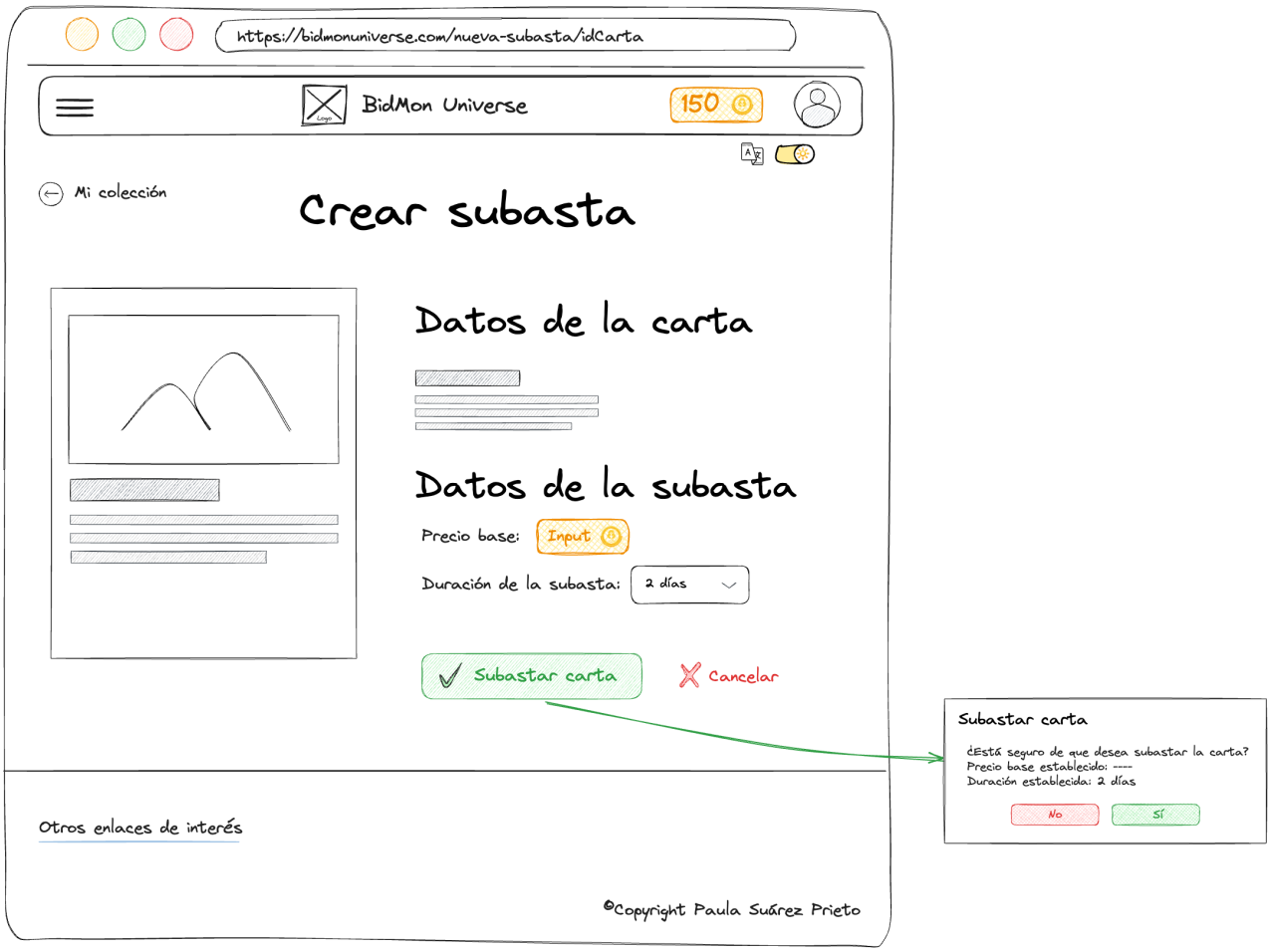


Figura 6.17: Boceto de la página de creación de subasta

#### 6.6.1.8. Página de Subastas

La página de subastas muestra una lista de las subastas activas en las que el usuario puede participar. En esta página hay un menú de navegación que permite al usuario filtrar las subastas por diferentes criterios:

- Todas: Todas las subastas activas.
- Mis Subastas: Subastas en las que el usuario ha creado.
- Pujas: Subastas en las que el usuario ha pujado.

La página de subastas muestra las cartas subastadas, al hacer clic en una carta se muestra la información detallada de la subasta.

En la Figura 6.18: Boceto de la página de subastas, se muestra el boceto de la página de subastas.

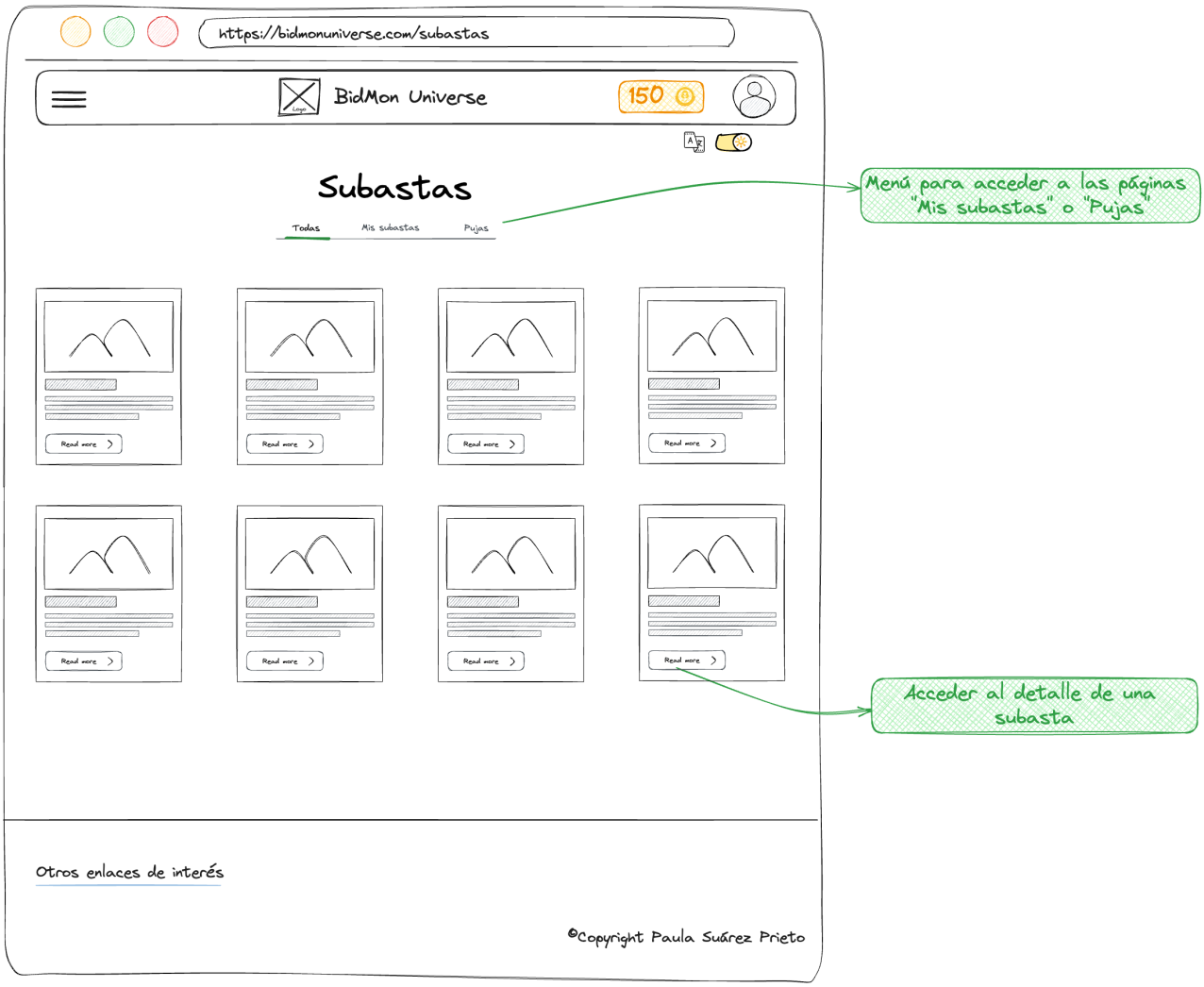


Figura 6.18: Boceto de la página de subastas

#### 6.6.1.9. Página de Detalles de Subasta

La página de detalles de subasta muestra información detallada sobre una subasta específica. Dependiendo de si la subasta es propia o no, el usuario puede realizar diferentes acciones.

En la Figura 6.19: Boceto de la página de detalles de subasta, se muestra el boceto de la página de detalles de una subasta activa creada por otro usuario.

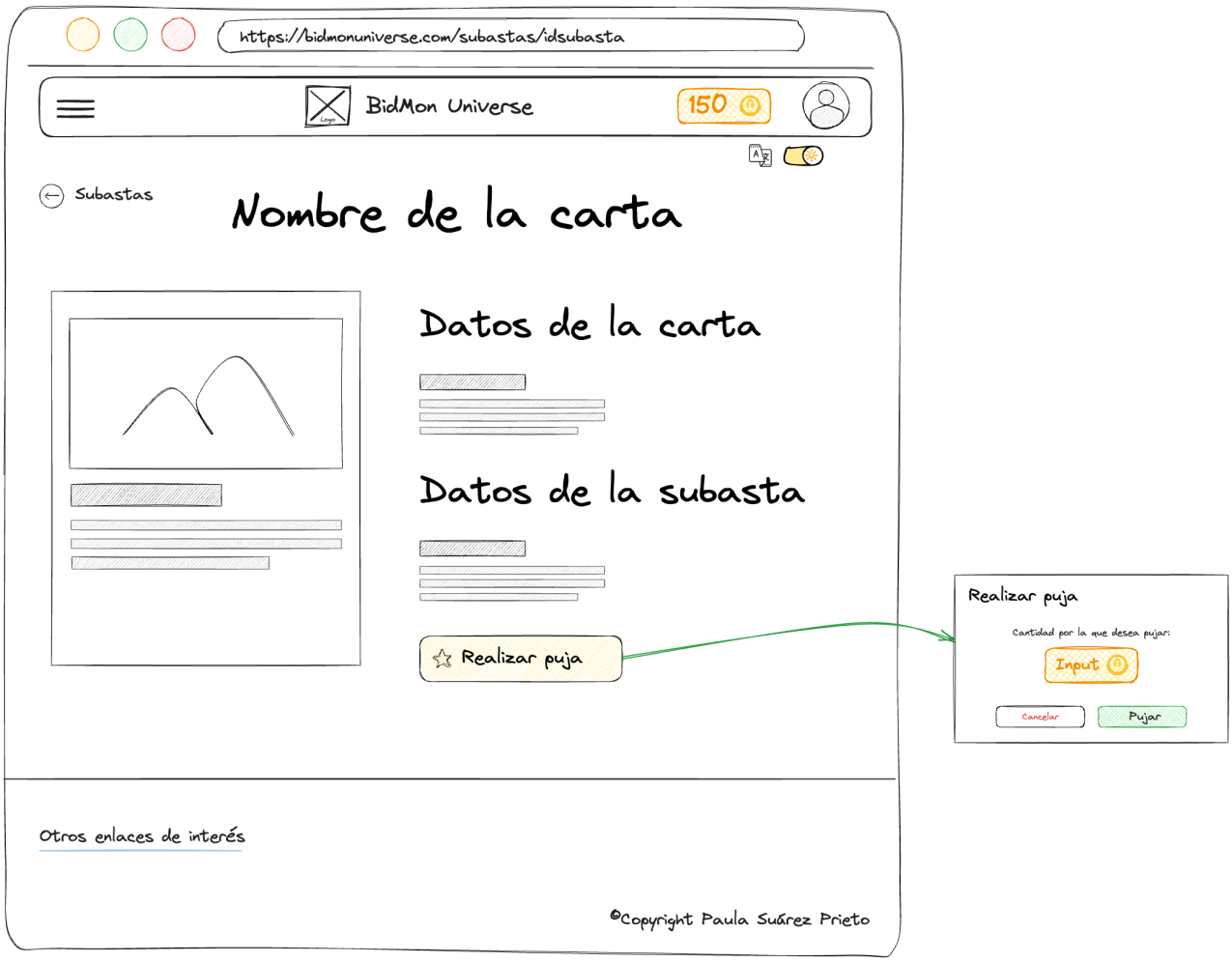


Figura 6.19: Boceto de la página de detalles de subasta

En el caso de que la subasta sea propia, el usuario puede cancelar la subasta como se muestra en la Figura 6.20: Boceto de la página de detalles de subasta propia.


Figura 6.20: Boceto de la página de detalles de subasta propia

En el caso de que sea una subasta en la que el usuario ha pujado, el usuario puede retirar la puja como se muestra en la Figura 6.21: Boceto de la página de detalles de subasta en la que el usuario ha pujado.


Figura 6.21: Boceto de la página de detalles de subasta en la que el usuario ha pujado

#### 6.6.1.10. Página de Histórico de Transacciones

La página de histórico de transacciones muestra una lista de las transacciones realizadas. Si el usuario que ha iniciado sesión es un administrador, se mostrarán todas las transacciones realizadas en el sistema. Si el usuario que ha iniciado sesión es un usuario normal, se mostrarán solo las transacciones realizadas por él.

En la Figura 6.22: Boceto de la página de histórico de transacciones, se muestra el boceto de la página de histórico de transacciones.

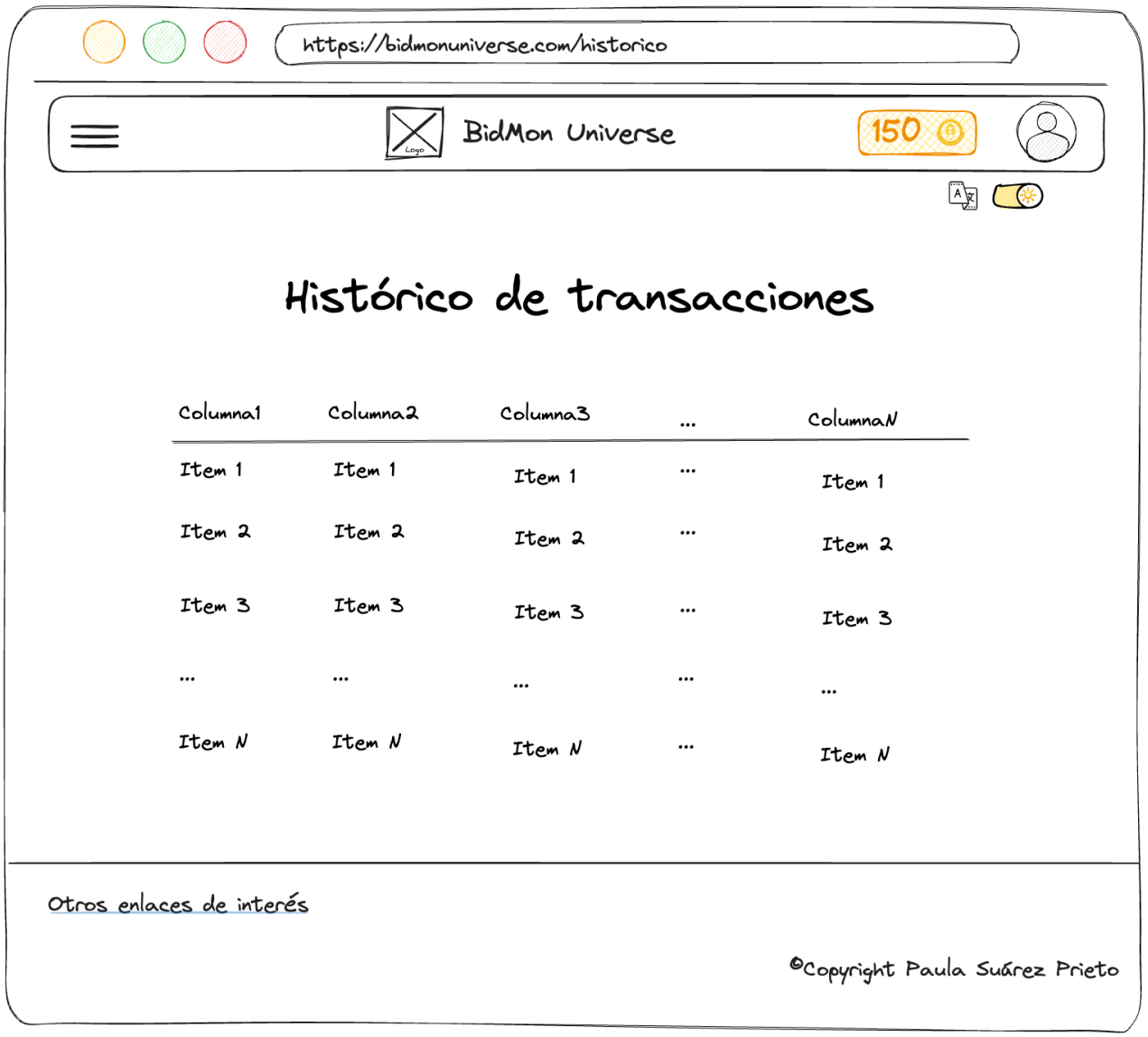


Figura 6.22: Boceto de la página de histórico de transacciones

### 6.6.2. Definición del aspecto de la interfaz

Se ha creado la interfaz en base a los bocetos de la sección anterior. La interfaz se ha desarrollado en React.

#### 6.6.2.1. Descripción de los recursos empleados en la interfaz

La primera tarea que se ha realizado ha sido la de definir los colores y tipografías que se van a utilizar en la interfaz. Se ha elegido la tipografía *Pokemon Hollow* para el título de la aplicación y para la bienvenida al usuario.

El logo de la aplicación se ha generado mediante inteligencia artificial, utilizando la herramienta ChatGPT , al igual que el *favicon* de la aplicación.

Los avatares que se pueden seleccionar como imagen de perfil son unas imágenes predefinidas. Estas se han obtenido de la página web Behance, Pokémon Avatars de Mikeel Araña.

Las imágenes de las medallas de los gimnasios se han obtenido de la página web Wikidex.

Los iconos de los tipos de Pokémon se han obtenido del repositorio pokemon-type-svg-icons, de duiker101.

Las imágenes de los Pokémon se han obtenido de la página web PokéAPI.

Para implementar la interfaz se ha utilizado principalmente la biblioteca Material-UI.

#### 6.6.2.2. Interfaz de la aplicación

A continuación se muestran las pantallas de las páginas principales de la aplicación. Pueden diferir ligeramente de los bocetos iniciales, ya que se han realizado ajustes para mejorar la usabilidad y la estética de la aplicación.

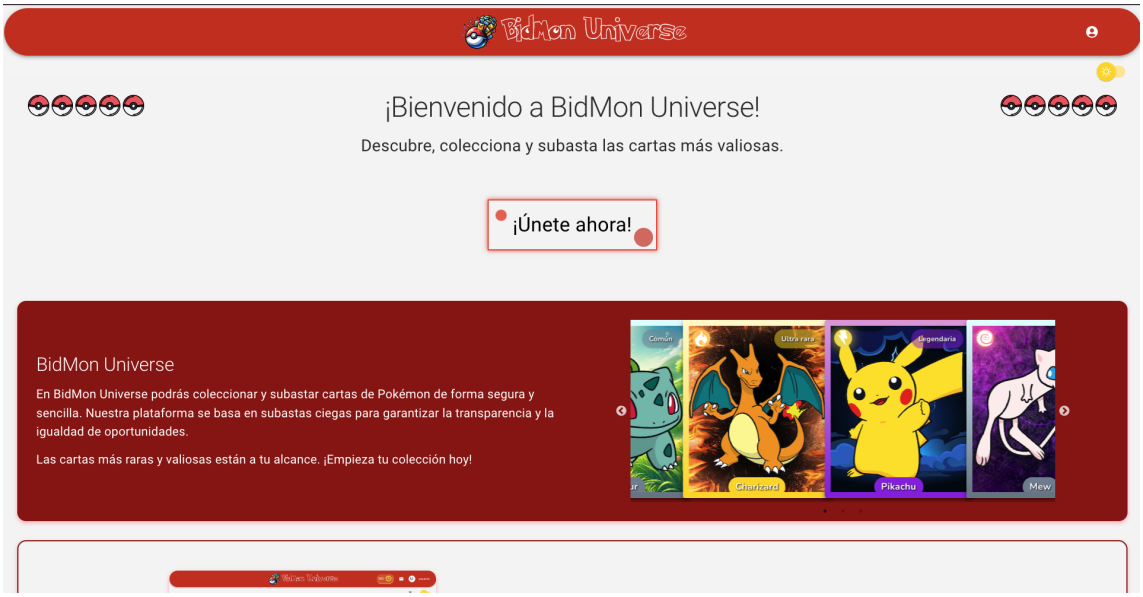


Figura 6.23: Página Home, página de inicio de la aplicación.

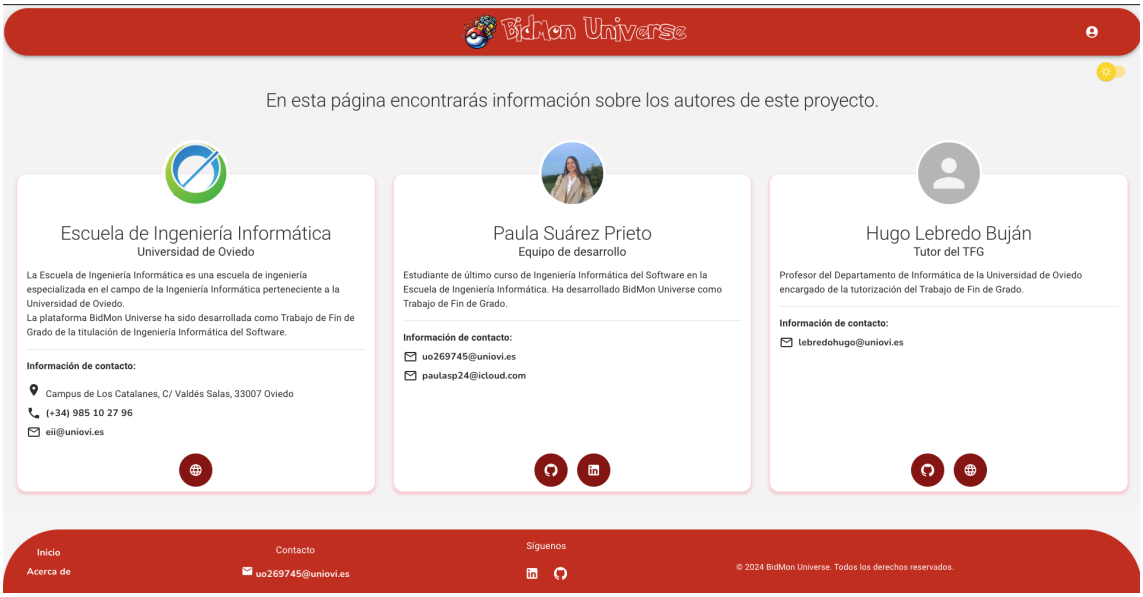


Figura 6.24: Página de información sobre la aplicación.

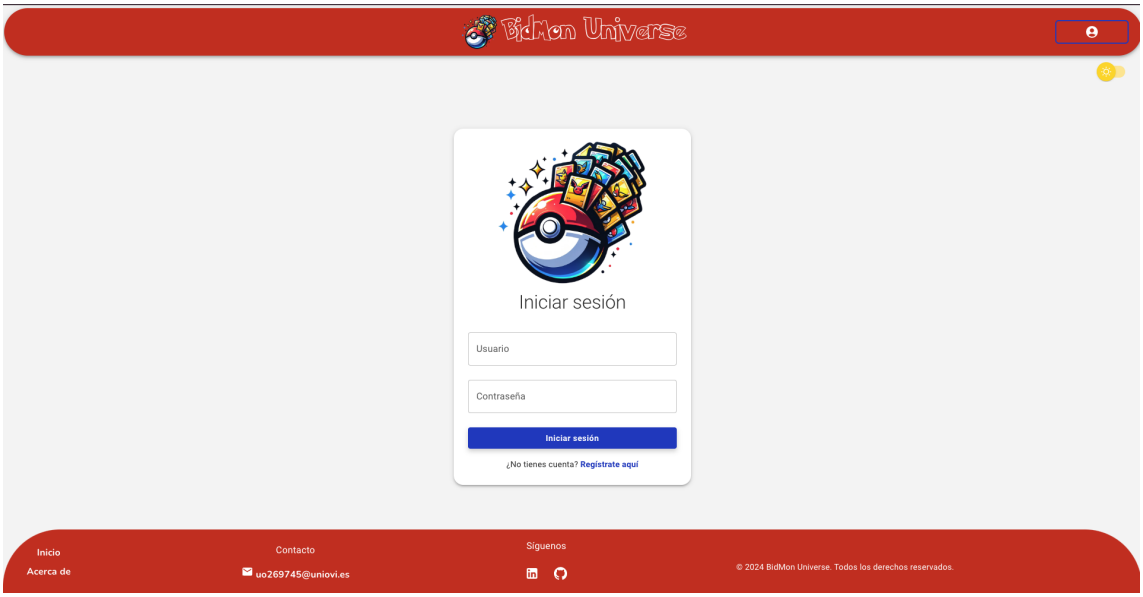


Figura 6.25: Página de inicio de sesión.

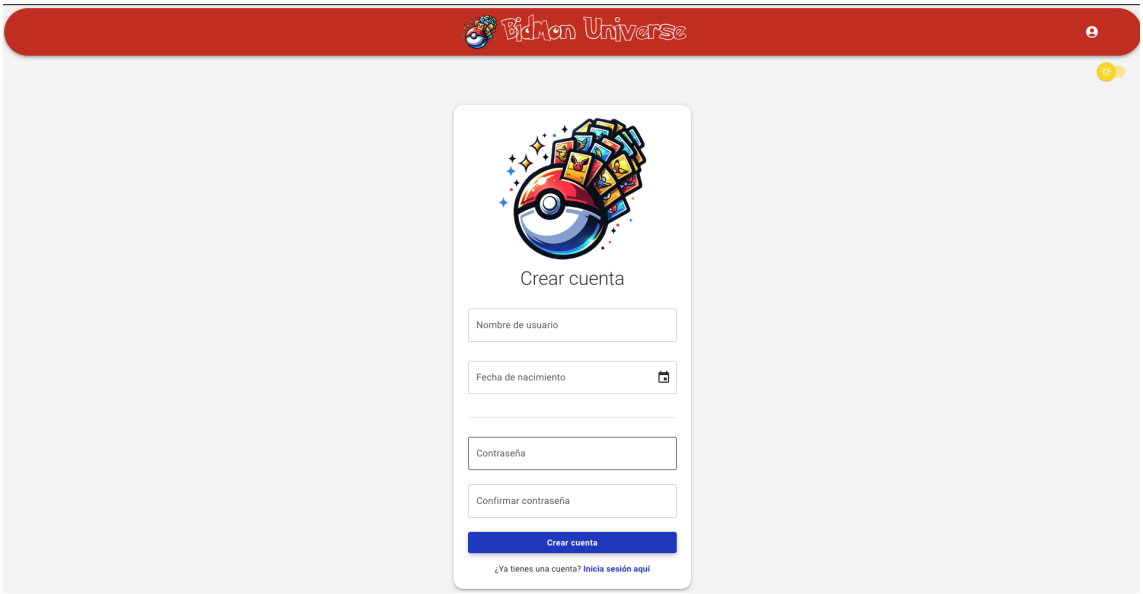


Figura 6.26: Página de registro de usuario.

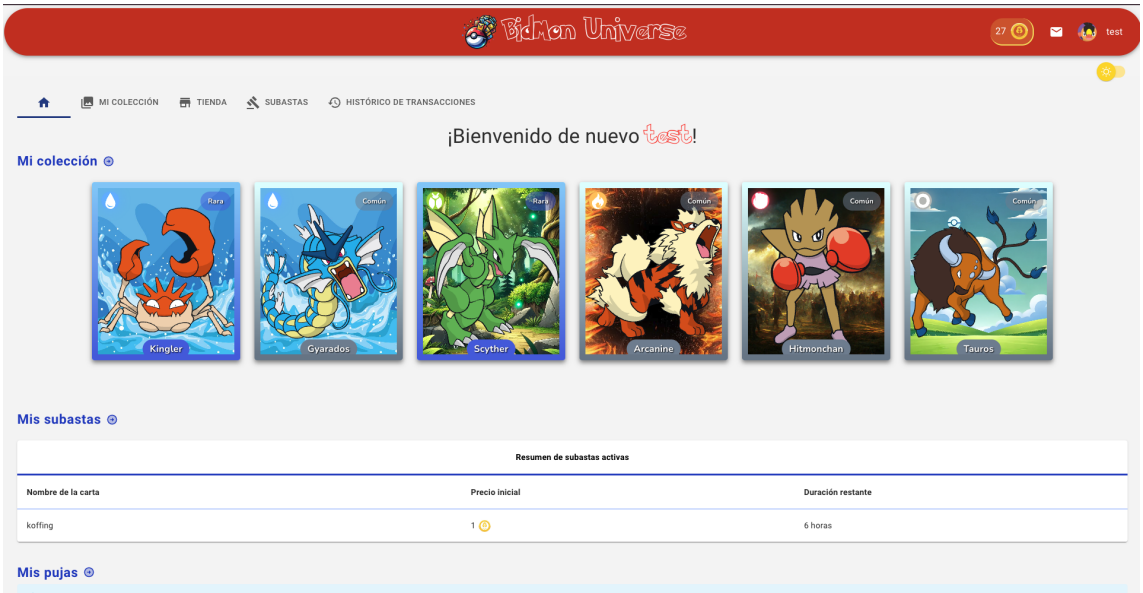


Figura 6.27: Página principal de la aplicación, una vez que el usuario ha iniciado sesión.

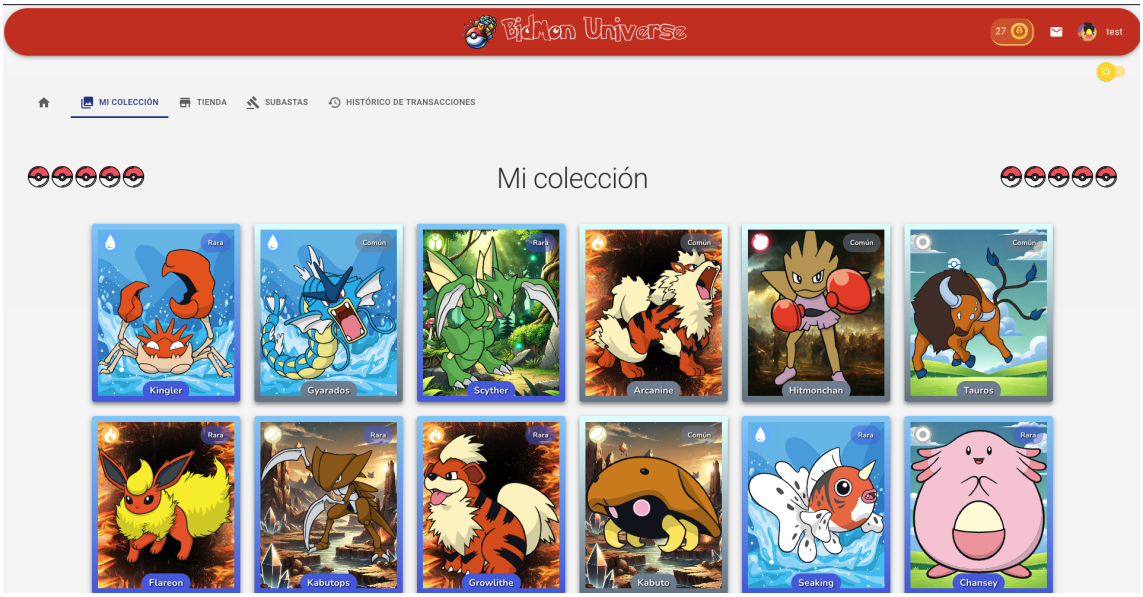


Figura 6.28: Página de la colección de cartas del usuario.

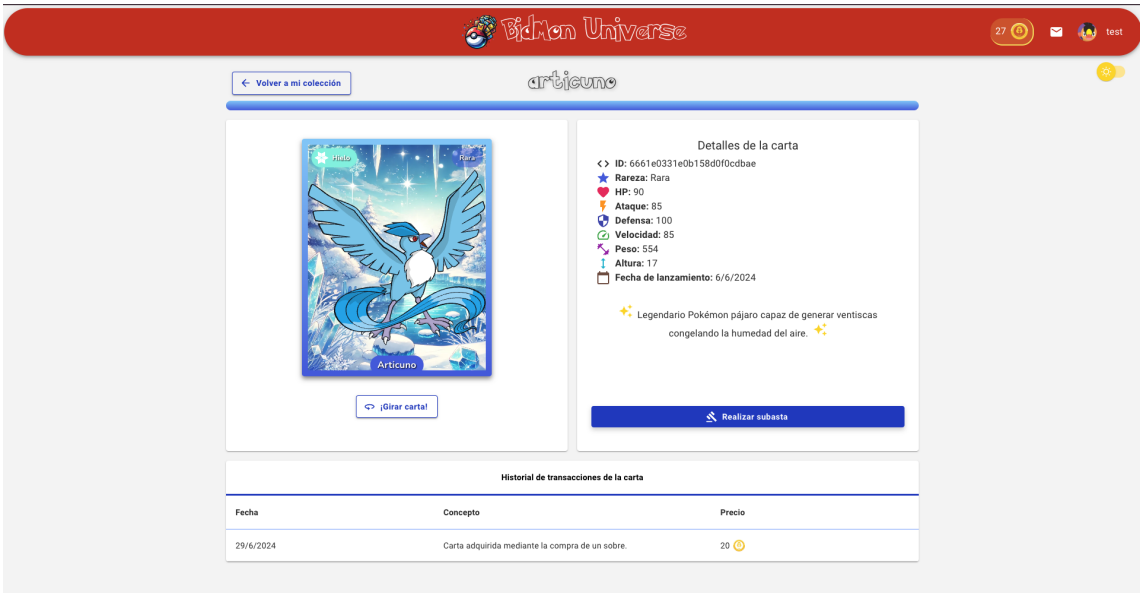


Figura 6.29: Página de detalle de una carta de la colección del usuario.

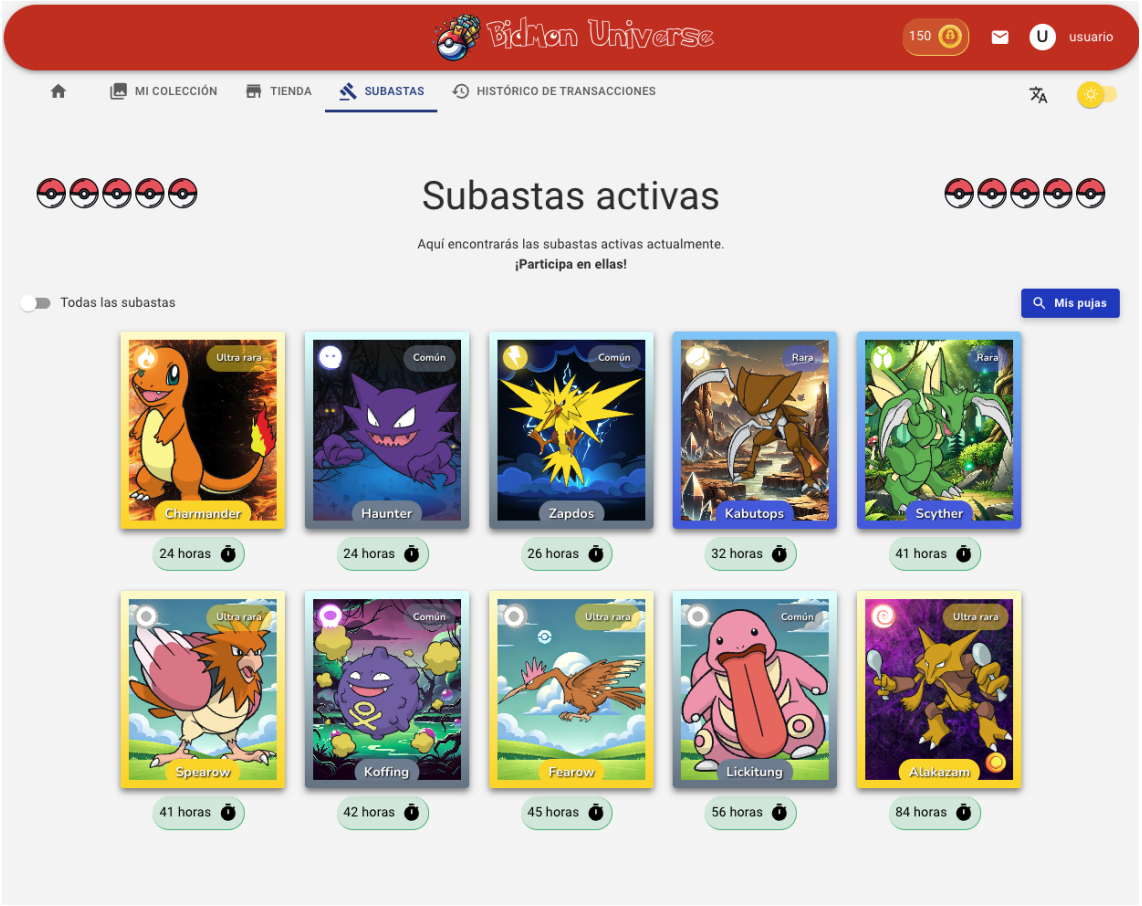


Figura 6.30: Página de las subastas activas de cartas.

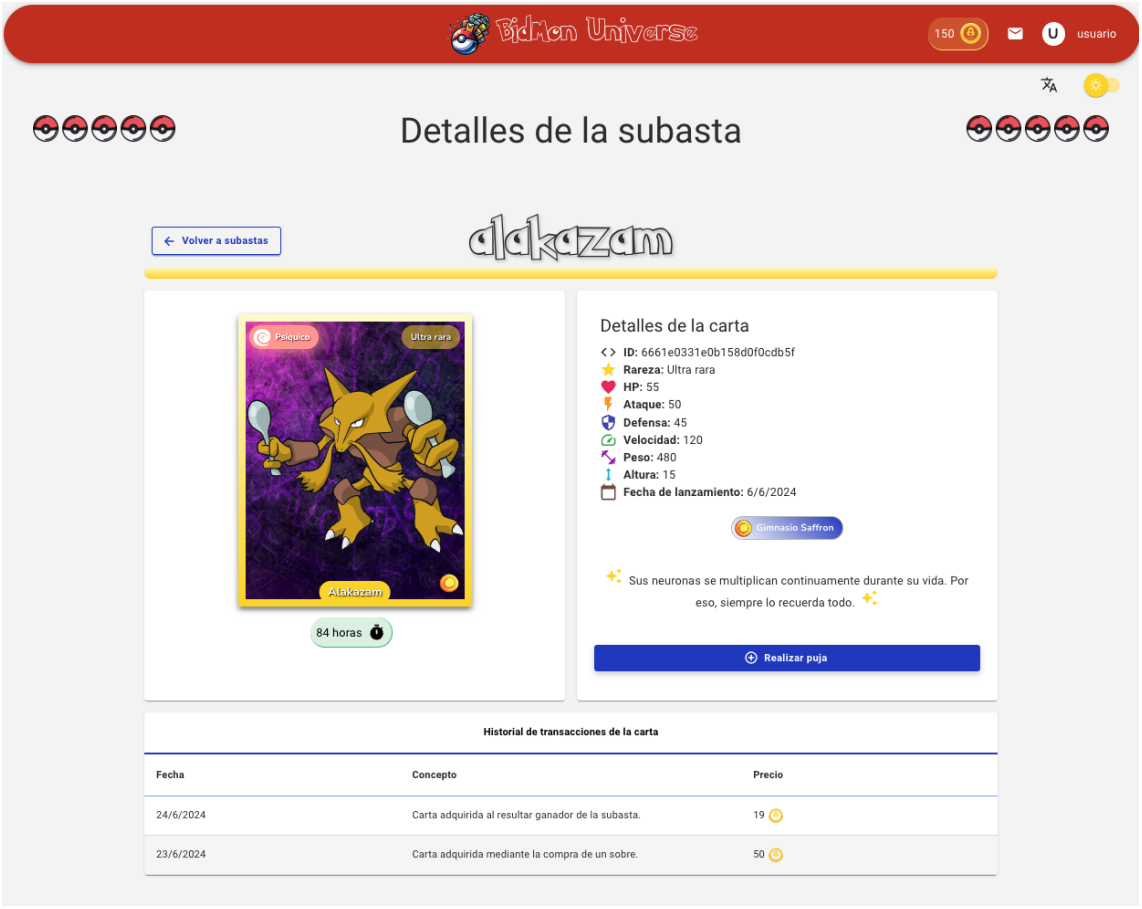


Figura 6.31: Página de detalle de una subasta activa.

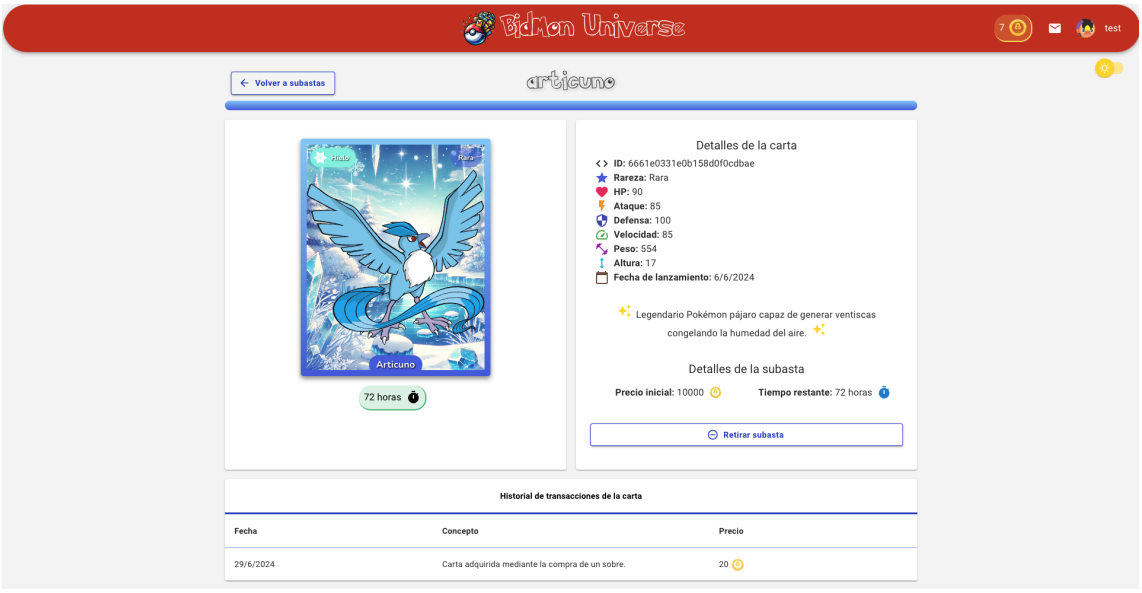


Figura 6.32: Página de detalle de una subasta activa creada por el usuario.

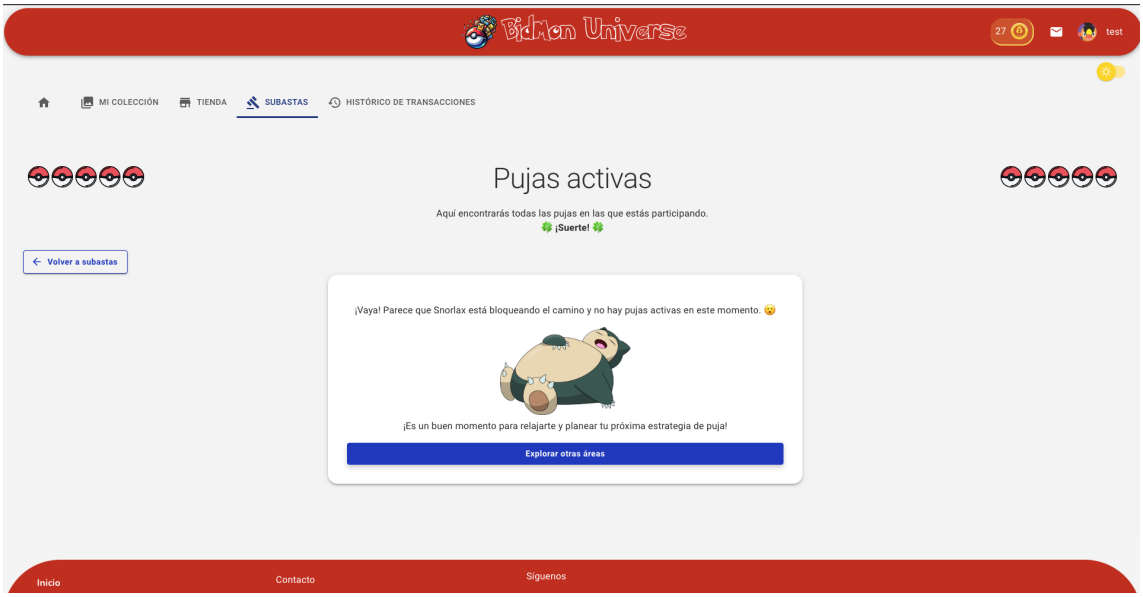


Figura 6.33: Página de pujas activas del usuario en subastas.

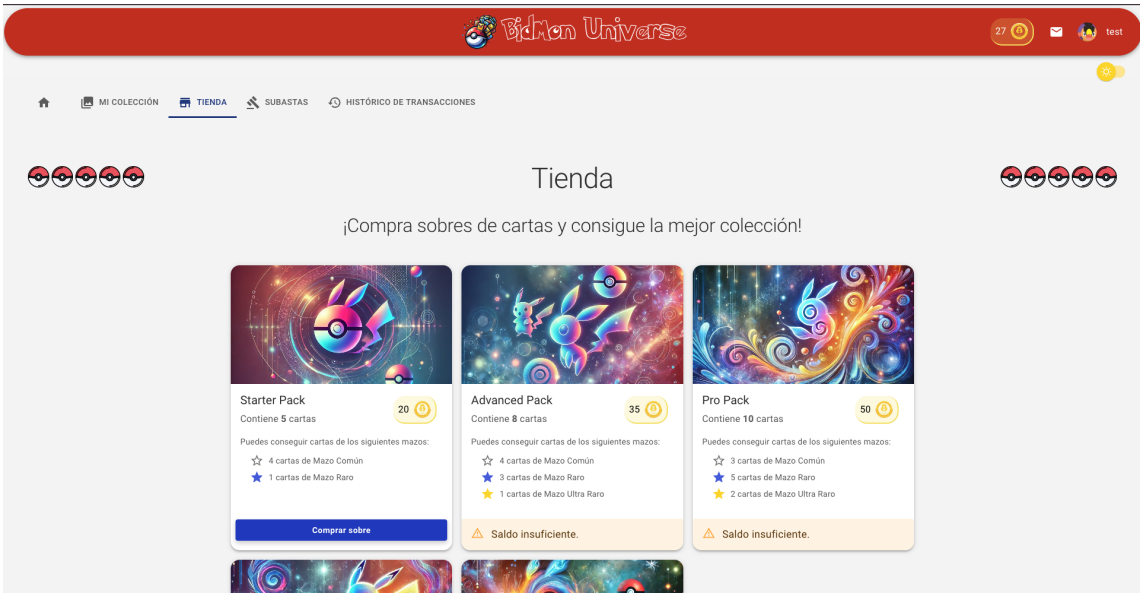


Figura 6.34: Página de la tienda de sobres de cartas.

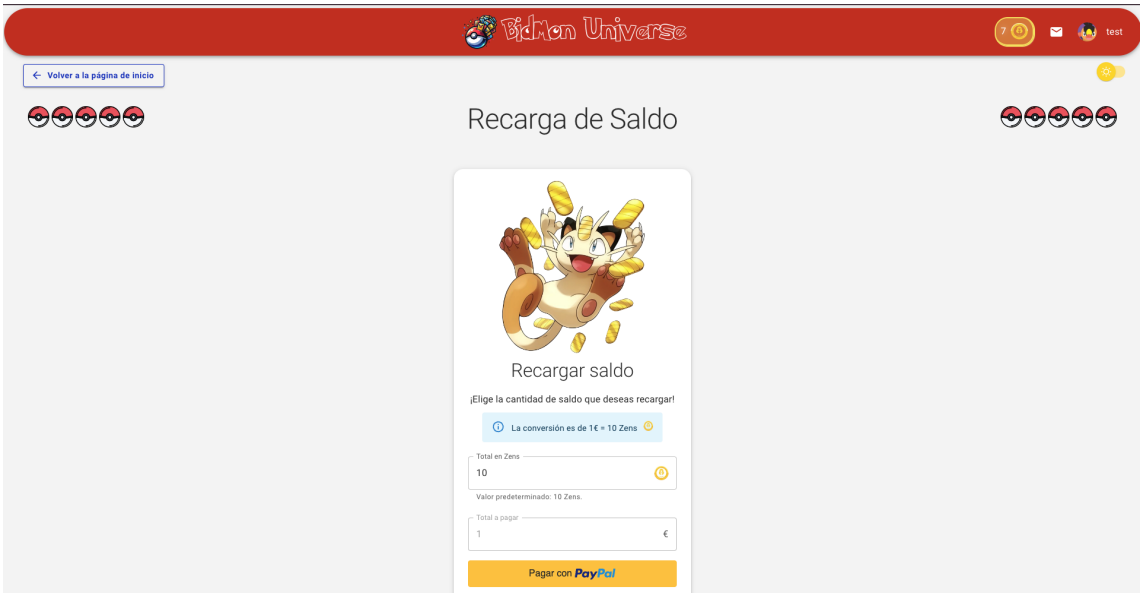


Figura 6.35: Página de recarga de saldo de la aplicación.

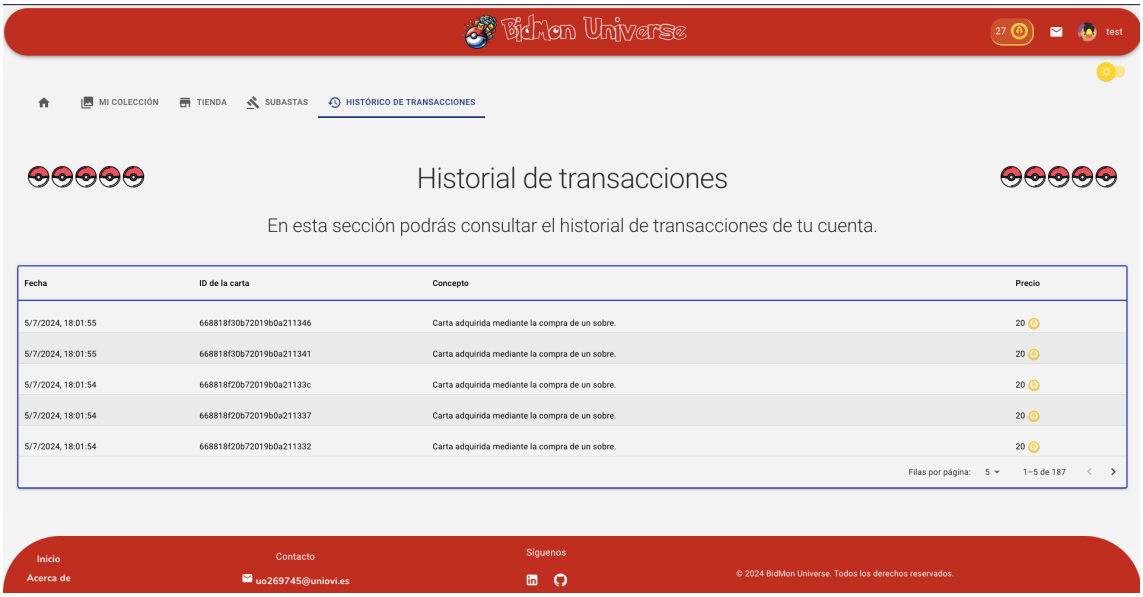


Figura 6.36: Página de transacciones realizadas por el usuario.

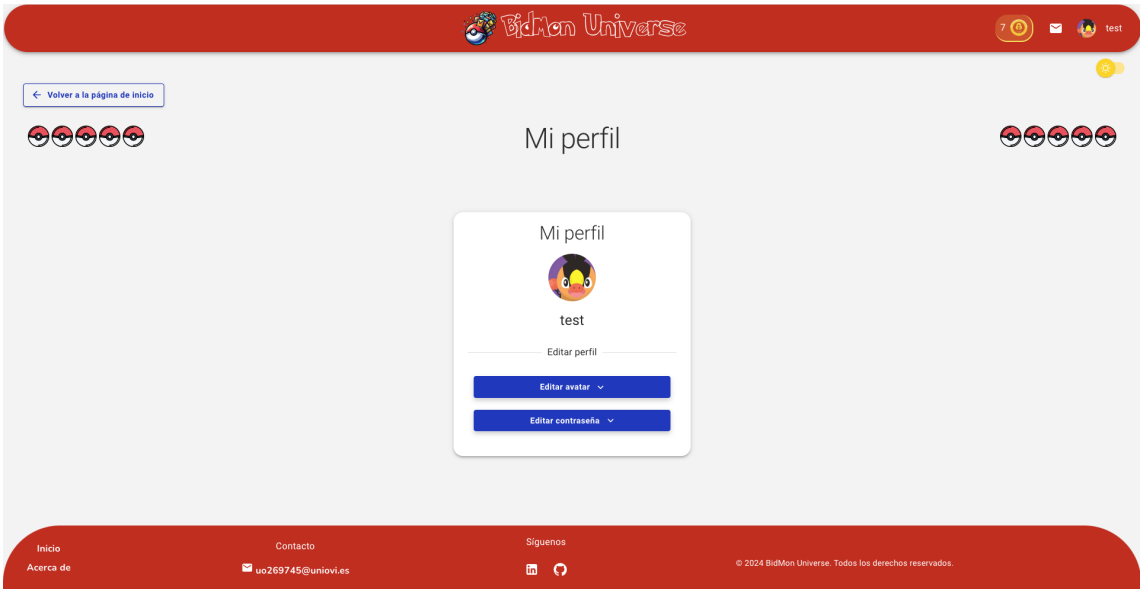


Figura 6.37: Página de perfil del usuario.

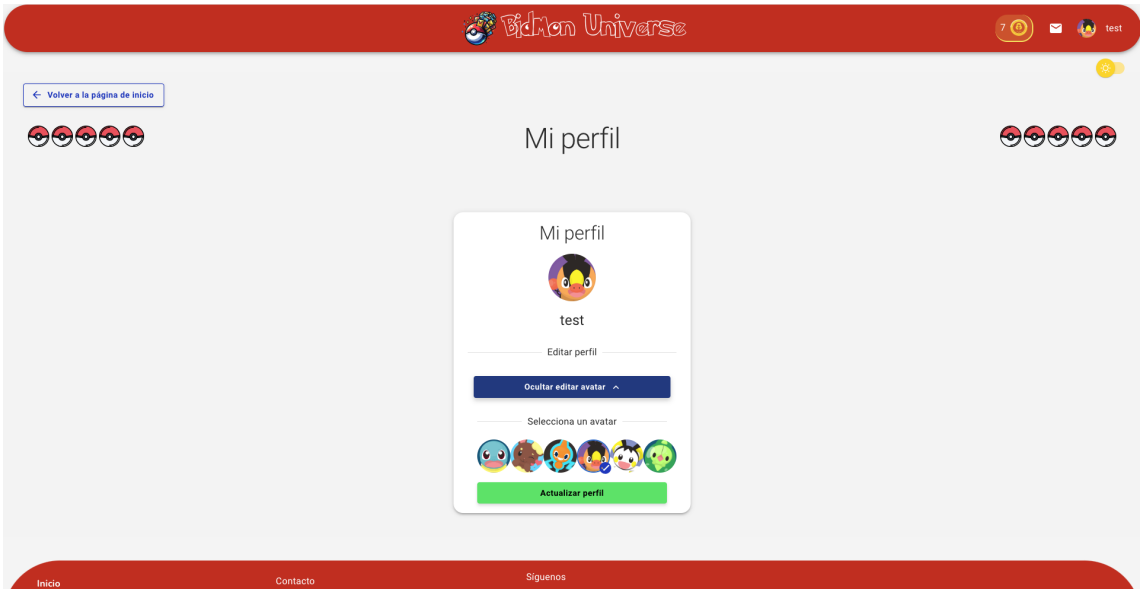


Figura 6.38: Página de perfil del usuario, con la opción de cambiar la imagen de perfil desplegada.

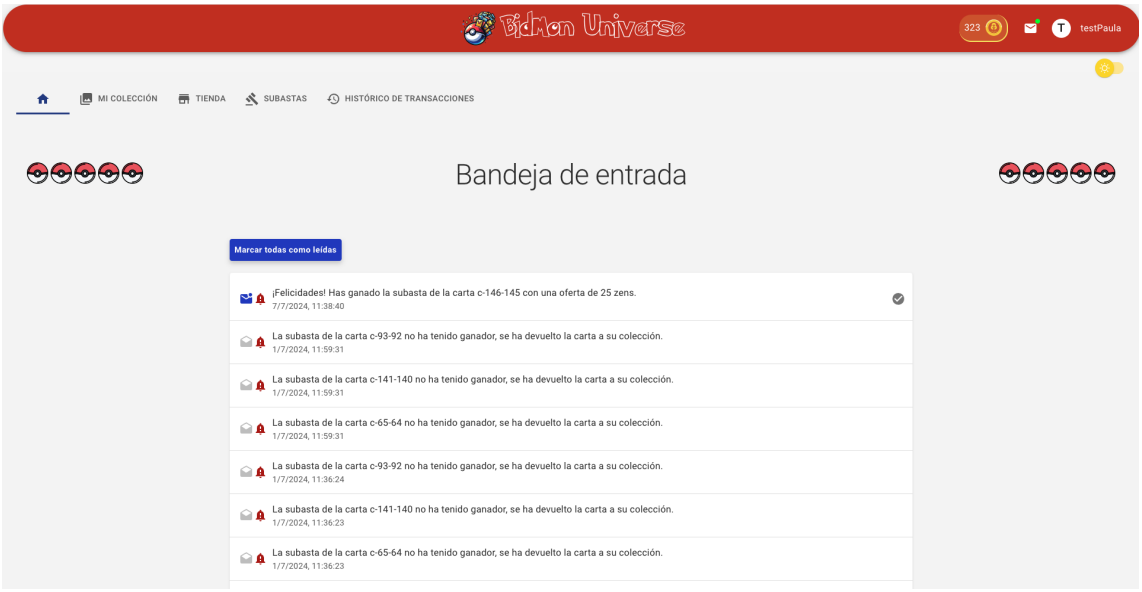


Figura 6.39: Página de notificaciones del usuario.

El usuario puede elegir entre dos temas de la aplicación: el tema claro y el tema oscuro. El tema claro es el que se muestra en las imágenes anteriores. A continuación se adjuntan un par de imágenes de la aplicación en el tema oscuro, la página de inicio y la página de pujas activas del usuario.

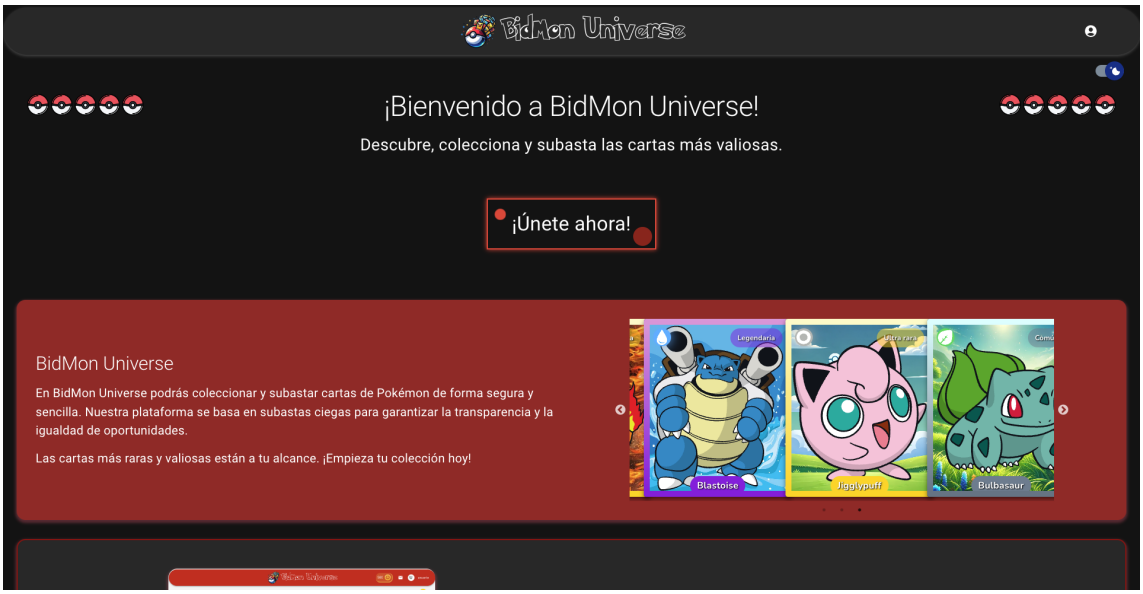


Figura 6.40: Página Home en tema oscuro.

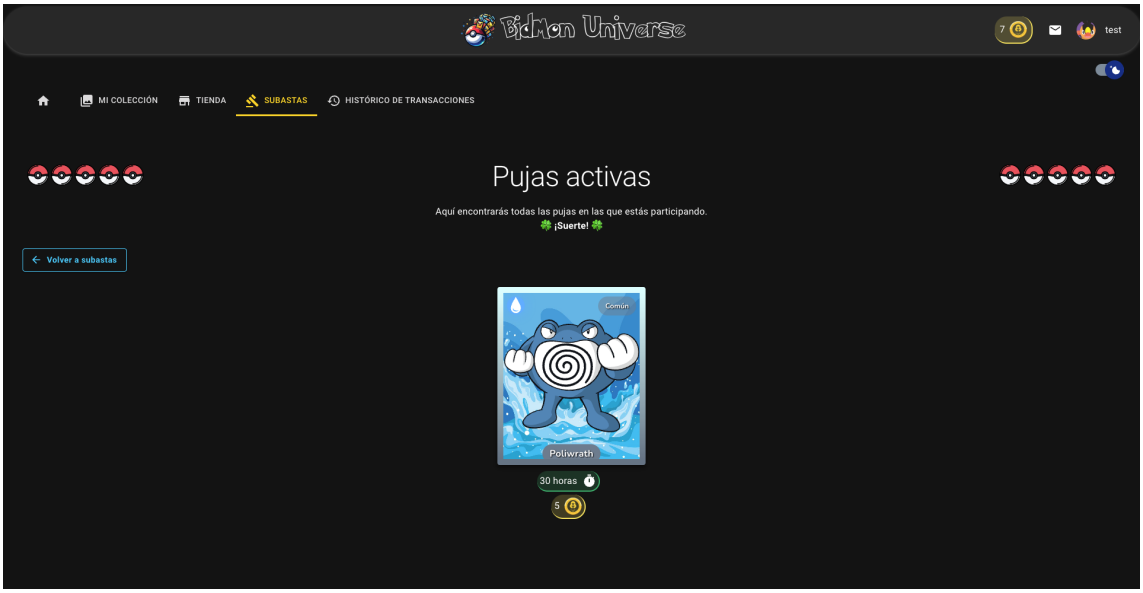


Figura 6.41: Página de pujas activas del usuario en tema oscuro.

### 6.6.3. Descripción del Comportamiento de la Interfaz

La interacción con la interfaz es sencilla, los botones tienen dos funcionalidades principales que son la de redirigir a otra página o la de realizar una acción.

Esta acción puede ser la de abrir un modal, como en el caso de la subasta de una carta o un menú desplegable, como en el caso del menú de usuario.

Los comportamientos más complejos de la interfaz son los de la subasta de una carta y la compra de un sobre de cartas. Estos comportamientos se han diseñado de forma que sean intuitivos y fáciles de entender para el usuario.

A continuación, se mostrará un ejemplo de cada uno de estos comportamientos.

#### 6.6.3.1. Subasta de una carta

En la Figura 6.42 se muestra la vista del modal de crear una subasta. En esta vista, el usuario puede seleccionar el precio de salida de la carta y el tiempo que durará la subasta.

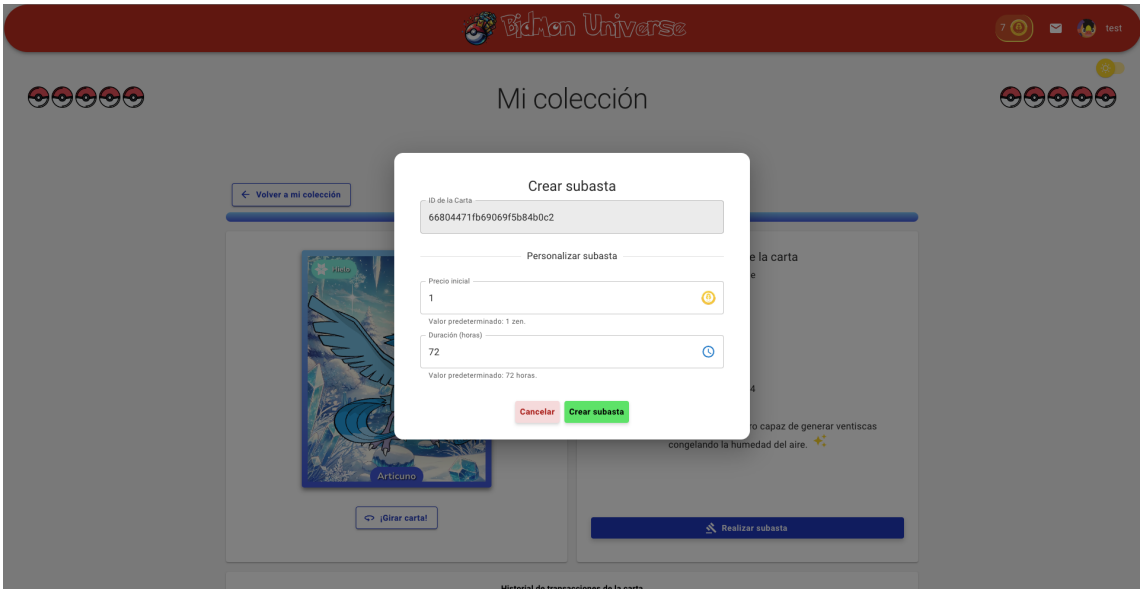


Figura 6.42: Modal de creación de subasta.

Una vez que el usuario ha rellenado los campos, puede pulsar el botón de *Crear subasta* para confirmar la subasta. Se le abrirá un modal de confirmación, en el que se le mostrará un resumen de la subasta y podrá confirmarla o cancelarla.

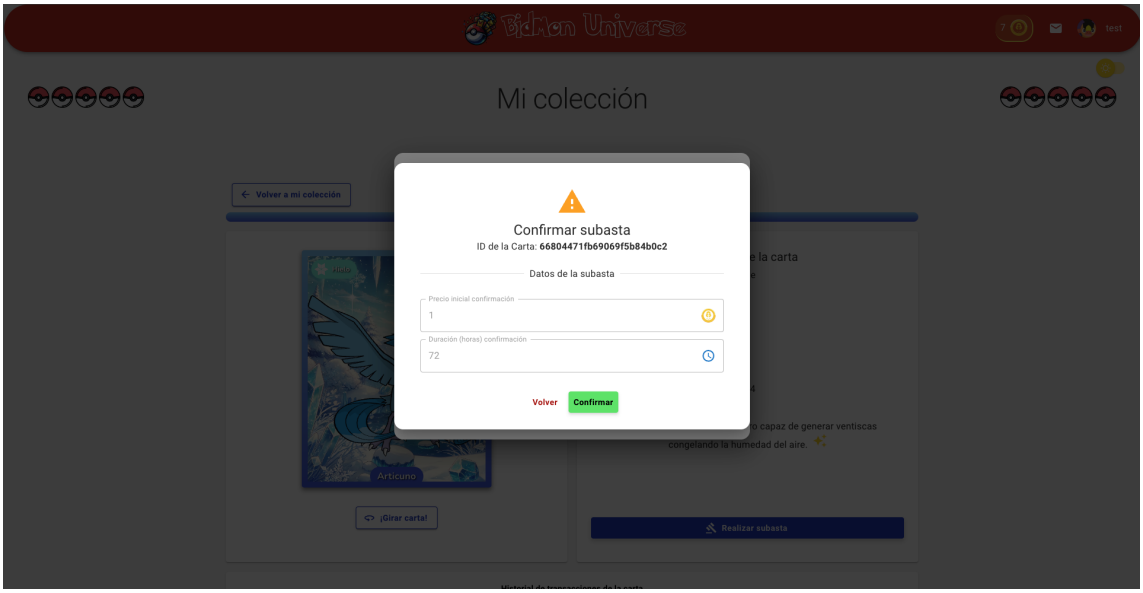


Figura 6.43: Modal de confirmación de subasta.

Si confirma la subasta, se mostrará un mensaje de éxito, se cerrará el modal y desparecerá el botón de subastar reemplazándolo por una alerta informativa de que la carta está en subasta.

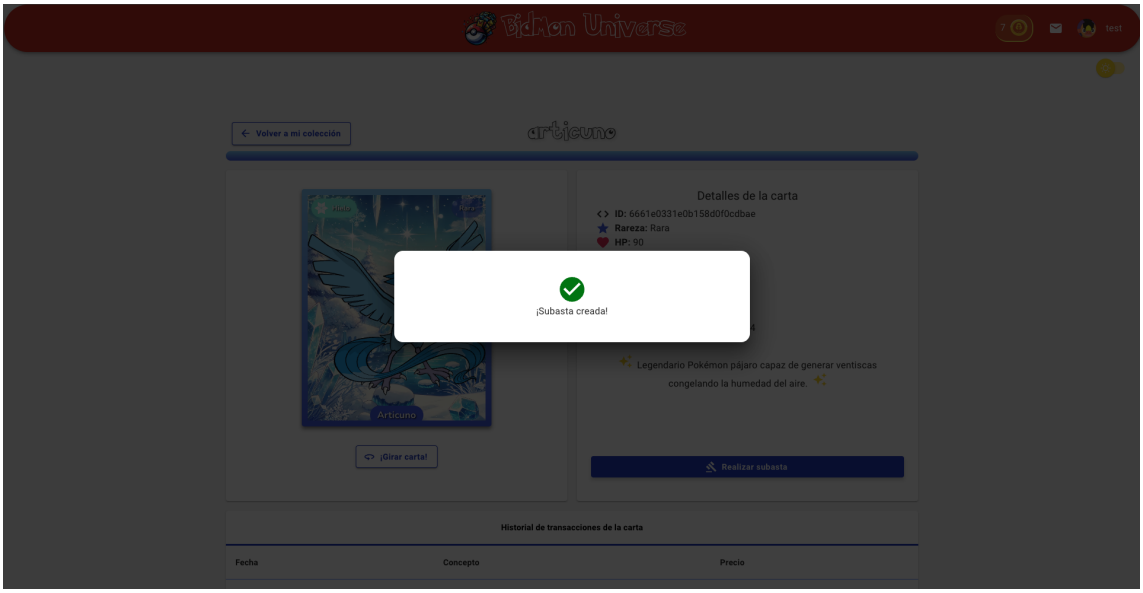


Figura 6.44: Mensaje de éxito de creación de subasta.

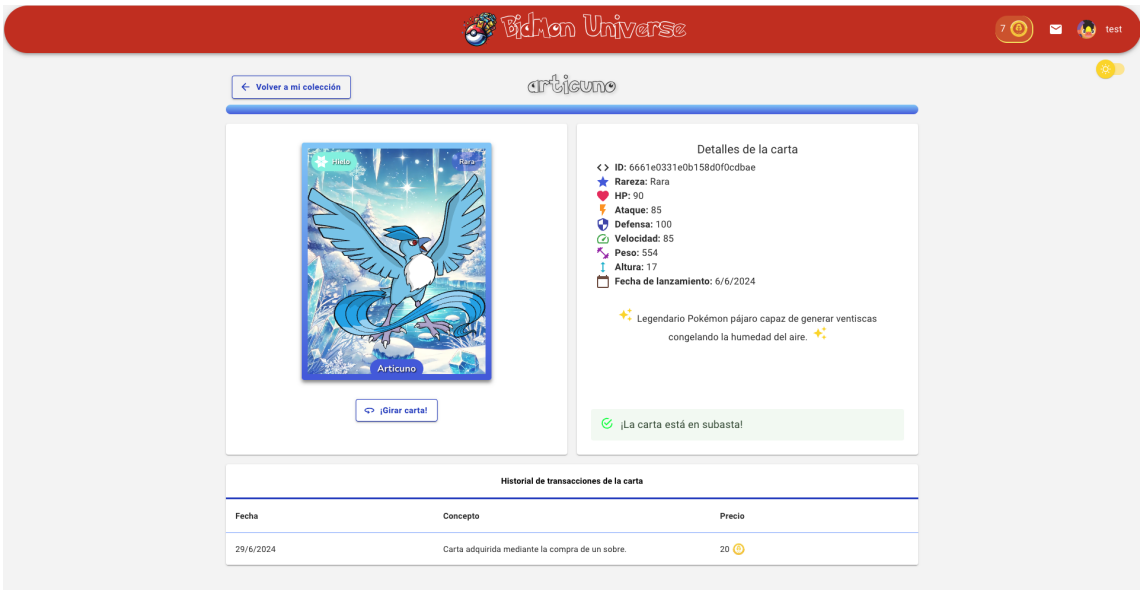


Figura 6.45: Alerta de que la carta está en subasta.

Si el usuario cancelase en algún momento el proceso de subasta, se cerraría el modal.

#### 6.6.3.2. Compra de un sobre de cartas

El proceso de compra de un sobre de cartas es más sencillo que el de la subasta de una carta. El usuario pulsa el botón de *Comprar sobre* del sobre que desea comprar y se le abrirá un modal de confirmación.

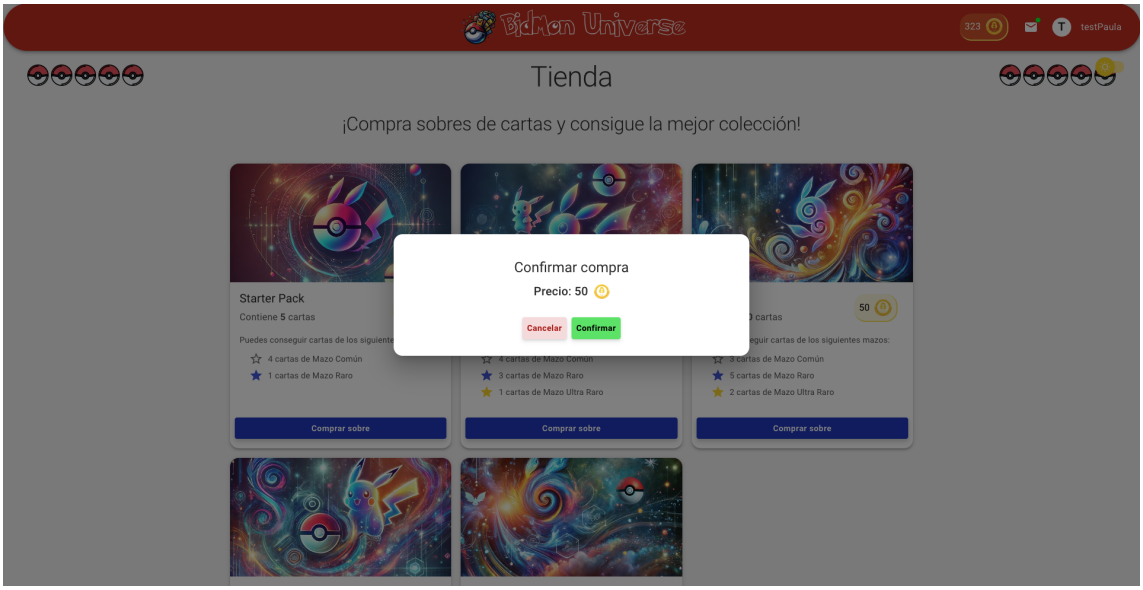


Figura 6.46: Modal de confirmación de compra de sobre.

Una vez que el usuario confirma la compra, se le mostrarán las cartas que ha obtenido en el sobre y un mensaje de éxito. Estas cartas aparecen volteadas por lo que el usuario debe pulsar sobre ellas para verlas. Tiene la opción de cerrar el modal o de ir a la colección de cartas para ver las cartas que ha obtenido.

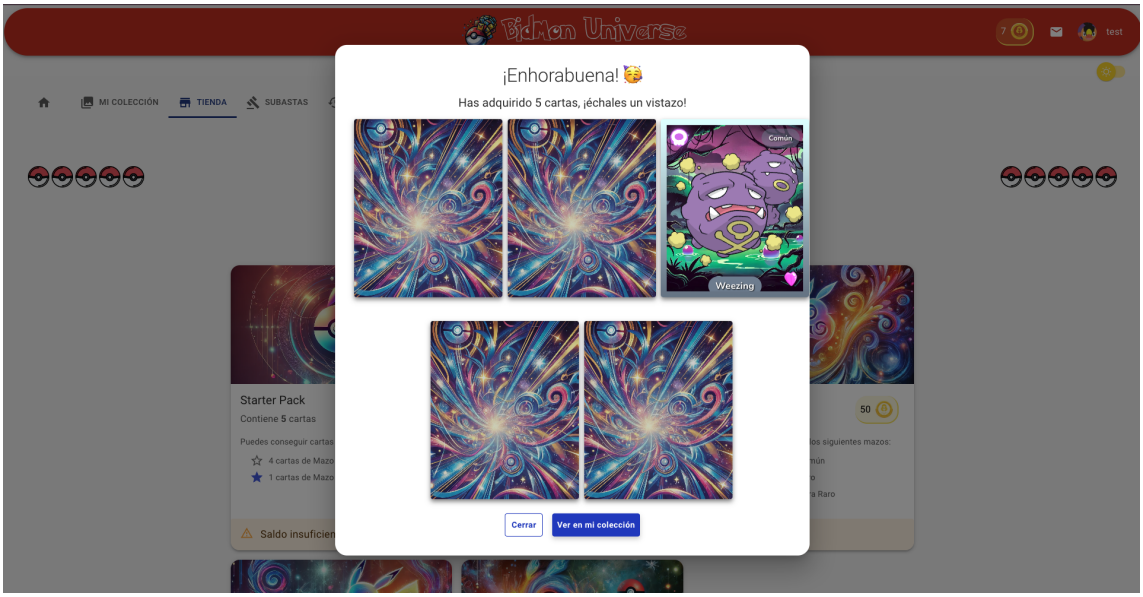


Figura 6.47: Modal de cartas obtenidas en el sobre.

### 6.6.4. Diagrama de Navegabilidad

En la sección 6.1.2 Identificación de Actores del Sistema se identificaron los actores del sistema, cada uno con niveles de acceso y funcionalidades específicas adecuadas a su rol dentro de la plataforma.

En este apartado se detalla el diagrama de navegabilidad de cada uno de los actores identificados en la sección mencionada anteriormente. Para cada actor se detallan las vistas a las que puede acceder.

Las cajas que aparecen sombreadas en el diagrama representan las vistas que son accesibles desde cualquier otra vista, debido a que se encuentran bien en el menú de navegación o en el pie de página de la aplicación.

La vista *Home* es la vista principal de la aplicación, en función de si el usuario está autenticado o no, tendrá una funcionalidad u otra, pero siempre es el punto de partida de la navegación y accesible desde cualquier vista.

#### 6.6.4.1. Usuario no autenticado

En la Figura 6.48 se muestra el diagrama de navegabilidad del usuario no autenticado.

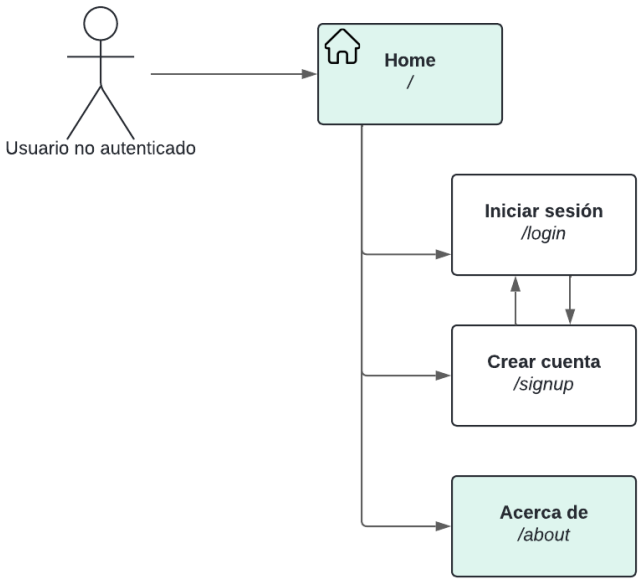


Figura 6.48: Diagrama de navegabilidad del usuario no autenticado.

#### 6.6.4.2. Usuario autenticado

En la Figura 6.49 se muestra el diagrama de navegabilidad del usuario autenticado.

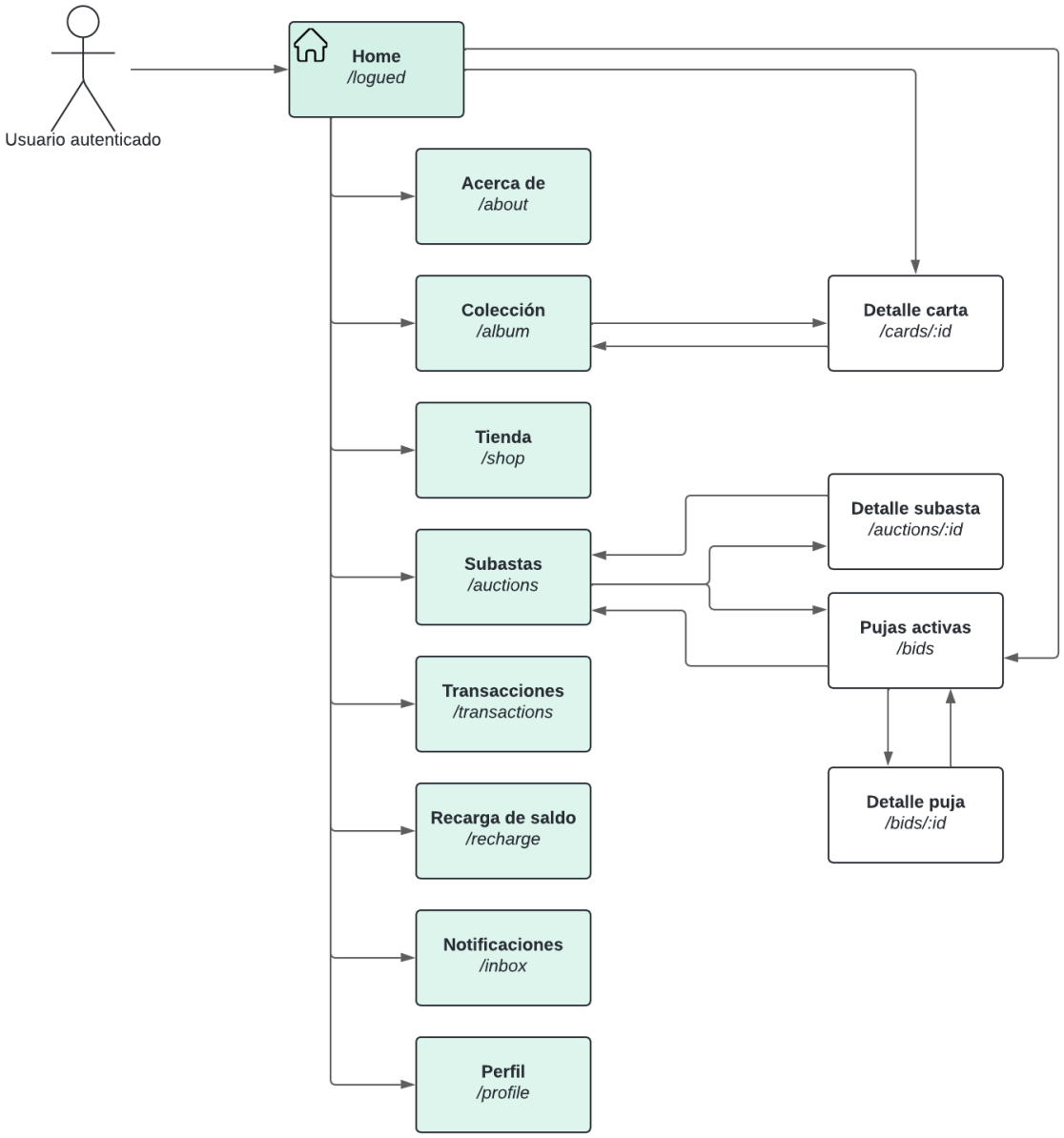


Figura 6.49: Diagrama de navegabilidad del usuario autenticado.

#### 6.6.4.3. Administrador

En la Figura 6.50 se muestra el diagrama de navegabilidad del administrador.

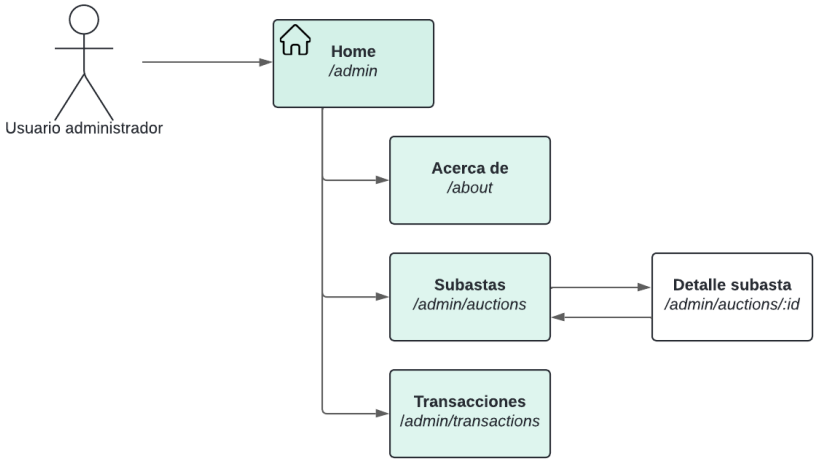


Figura 6.50: Diagrama de navegabilidad del administrador.

## 6.7. ESPECIFICACIÓN DEL PLAN DE PRUEBAS

### 6.7.1. Especificación del Plan de Pruebas

Se realizarán cuatro tipos de pruebas para garantizar la calidad del sistema: pruebas unitarias, pruebas de integración, pruebas de usabilidad y pruebas de accesibilidad. A continuación, se detallará cómo se llevarán a cabo las pruebas de cada tipo y se especificarán los criterios de aceptación para cada una de ellas.

#### 6.7.1.1. Pruebas Unitarias

Las pruebas unitarias se realizarán para comprobar que cada componente del sistema funciona correctamente de forma aislada. Se realizarán pruebas unitarias en el subsistema **restapi**, para comprobar que las rutas de la API REST funcionan correctamente, y en el subsistema **webapp**, para comprobar que los componentes de la interfaz de usuario se renderizan correctamente.

Para ello, se utilizará el framework de pruebas Jest para ambos subsistemas y se ejecutarán las pruebas en un entorno de test local. Se espera que todas las pruebas unitarias pasen con éxito y que se alcance un porcentaje de cobertura de código mínima del 60 % para el subsistema **restapi** y para el subsistema **webapp** se deberán de cubrir los componentes que más se reutilizan en la aplicación.

Estas pruebas se realizarán en un entorno de test local, se utilizarán datos de prueba que simulan la información que se almacenará en la base de datos junto con una base de datos de pruebas.

#### 6.7.1.2. Pruebas de Integración

Las pruebas de integración, también conocidas como pruebas de extremo a extremo o *end-to-end*, se realizarán para comprobar que los distintos componentes del sistema funcionan correctamente en conjunto. Se realizarán en el subsistema **webapp** con el framework de pruebas jest-cucumber y Puppeteer, que permite simular la interacción de un usuario con la aplicación y son muy descriptivas al iincluir escenarios de prueba escritos en lenguaje natural. Estas pruebas se ejecutarán en un entorno de local de desarrollo y se utilizarán datos reales de la base de datos de desarrollo.

Los criterios de aceptación para estas pruebas son que se compruebe que un usuario puede iniciar sesión en la aplicación y que se renderice correctamente la página de inicio.

#### 6.7.1.3. Pruebas de Usabilidad

Las pruebas de usabilidad se realizarán para comprobar que la interfaz de usuario es intuitiva y fácil de usar. Se realizarán pruebas de usabilidad en el subsistema **webapp** con usuarios reales, que evaluarán la interfaz de usuario y proporcionarán retroalimentación sobre su usabilidad. Estas pruebas se ejecutarán en un entorno de producción y se utilizarán datos reales de la base de datos de producción. Los criterios de aceptación para estas pruebas son que no se reporten errores graves de usabilidad y que la interfaz de usuario sea fácil de usar para los usuarios reales.

#### 6.7.1.4. Pruebas de Accesibilidad

Las pruebas de accesibilidad se realizarán para comprobar que la aplicación es accesible para personas con discapacidades. Se utilizará el plugin de Google Chrome WAVE Evaluation Tool para comprobar la accesibilidad de la aplicación, también se utilizará Google Lighthouse para comprobar la accesibilidad de la aplicación.

En un primer lugar se realizarán las pruebas de accesibilidad en el entorno de desarrollo y posteriormente, se repetirán en el entorno de producción con los errores corregidos.

Los criterios de aceptación para estas pruebas es que no haya errores de accesibilidad en el plugin WAVE Evaluation Tool, permitiendo solo los errores de accesibilidad relacionados con el diseño específico de la aplicación.

#### 6.7.1.5. Pruebas de adaptabilidad

Las pruebas de adaptabilidad se realizarán para comprobar que la aplicación se renderiza correctamente en distintos dispositivos y navegadores. Las pruebas se realizarán en el entorno de producción y se utilizarán distintos dispositivos y navegadores para comprobar la adaptabilidad de la aplicación. Los criterios de aceptación para estas pruebas son que la aplicación se renderice correctamente en dispositivos móviles, tabletas y ordenadores, y que funcione correctamente en los navegadores Google Chrome, Mozilla Firefox y Safari.

### 6.7.2. Resultados de las Pruebas

En esta sección se especificará en primer lugar las características del dispositivo y de los navegadores utilizados, y a continuación se detallarán los resultados de las pruebas realizadas.

#### 6.7.2.1. Dispositivo y navegadores utilizados

Las pruebas se han llevado a cabo en un ordenador portátil con las siguientes características:

- **Sistema Operativo**: macOS Sonoma 14.5.
- **Procesador**: Apple M3 Pro Max.
- **CPU**: 14 núcleos.
- **GPU**: 30 núcleos.
- **Memoria RAM**: 36 GB.

Como navegador se ha utilizado Google Chrome principalmente, las pruebas de adaptabilidad se han realizado además en Mozilla Firefox y Safari. Las versiones de los navegadores son las siguientes:

- **Google Chrome**: Versión 126.0.6478.127 (64 bits).
- **Mozilla Firefox**: Versión 127.0.2 (64 bits).
- **Safari**: Versión 17.5 (19618.2.12.11.6)

#### 6.7.2.2. Pruebas Unitarias

Las pruebas unitarias se han realizado con éxito, se han comprobado que los componentes del sistema funcionan correctamente de forma aislada. Se han realizado pruebas unitarias para cada *router* del subsistema **restapi** y para los componentes más importantes del subsistema **webapp**.

##### *6.7.2.2.1 Pruebas Unitarias. Restapi*

En el subsistema **restapi** se han realizado 51 pruebas unitarias, todas ellas han pasado con éxito obteniendo un 76.78 % de cobertura de código para *routers*. Se puede ver detallado el porcentaje de cobertura de código en la Figura 6.51: Cobertura de Código del Subsistema restapi.

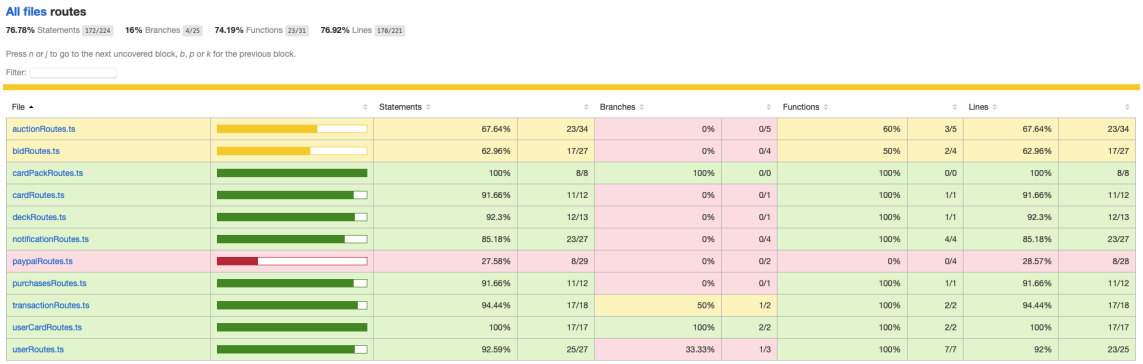


Figura 6.51: Cobertura de Código del Subsistema restapi

A continuación, se detallan las pruebas unitarias realizadas en el subsistema **restapi**, con su descripición y resultado esperado. Cabe destacar que para la ejecución de todas las pruebas, salvo para *POST /api/users/login* y *POST /api/users/signup*, se ha utilizado un token de autenticación válido, es decir, se ha iniciado sesión previamente con un usuario existente.

Tabla 6.59: Tests de auctionRouter

| **Tests de auctionRouter** | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/auctions** | Devuelve todas las subastas. | 200, la respuesta contiene una lista de subastas. |
| **GET /api/auctions/a/:id** | Devuelve una subasta específica. | 200, la respuesta contiene los datos de la subasta. |
| **GET /api/auctions/a/:id (no existe)** | Devuelve 404 si el ID de la subasta no se encuentra. | 404, la respuesta indica que la subasta no se encuentra. |

Continúa en la siguiente página...

Tabla 6.59 Tests de auctionRouter – continuación de la página anterior

| Tests de auctionRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/auctions/active/:username** | Devuelve todas las subastas activas para un nombre de usuario válido. | 200, la respuesta contiene una lista de subastas activas. |
| **GET /api/auctions/active/u/:username** | Devuelve todas las subastas activas para un usuario. | 200, la respuesta contiene una lista de subastas activas. |

Tabla 6.60: Tests de bidRouter

| Tests de bidRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/bids/b/:id** | Devuelve una puja específica. | 200, la respuesta contiene los datos de la puja. |
| **GET /api/bids/b/:id (no existe)** | Devuelve 404 si el ID de la puja no se encuentra. | 404, la respuesta indica que la puja no se encuentra. |
| **GET /api/bids/active/u/:username** | Devuelve todas las pujas activas para un nombre de usuario válido. | 200, la respuesta contiene una lista de pujas activas. |

Tabla 6.61: Tests de cardPackRouter

| Tests de cardPackRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/cardpacks** | Devuelve todos los paquetes de cartas disponibles. | 200, la respuesta contiene una lista de paquetes de cartas filtrados por disponibilidad. |

Tabla 6.62: Tests de cardRouter

| Tests de cardRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |

Continúa en la siguiente página...

Tabla 6.62 Tests de cardRouter – continuación de la página anterior

| Tests de cardRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/cards/:cardId** | Devuelve una carta por su ID. | 200, la respuesta contiene los datos de la carta, incluyendo el nombre 'bulbasaur'. |
| **GET /api/cards/:cardId (no existe)** | Devuelve 404 si la carta no se encuentra. | 404, la respuesta contiene el mensaje 'Carta no encontrada.'. |
| **GET /api/cards/:cardId (error)** | Maneja errores de forma adecuada. | 500, la respuesta contiene el mensaje 'Se ha producido un error al obtener la carta.'. |

Tabla 6.63: Tests de deckRouter

| Tests de deckRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/decks** | Devuelve todos los mazos de cartas. | 200, la respuesta contiene una lista de mazos de cartas. |
| **GET /api/decks/:deckid** | Devuelve un mazo de cartas por su ID. | 200, la respuesta contiene los datos del mazo de cartas, incluyendo 'deckId', 'name', 'type', y 'publicationDate'. |
| **GET /api/decks (error)** | Maneja errores de forma adecuada al obtener todos los mazos de cartas. | 500, la respuesta contiene el mensaje 'Se ha producido un error al obtener los mazos de cartas.'. |
| **GET /api/decks/:deckid (no existe)** | Devuelve 404 si el mazo de cartas no se encuentra. | 404, la respuesta contiene el mensaje 'Mazo de cartas no encontrado.'. |
| **GET /api/decks/:deckid (error)** | Maneja errores de forma adecuada al obtener un mazo de cartas por su ID. | 500, la respuesta contiene el mensaje 'Se ha producido un error al obtener el mazo de cartas.'. |

Tabla 6.64: Tests de notificationRouter

| Tests de notificationRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/notifications/:username** | Devuelve todas las notificaciones para un nombre de usuario válido. | 200, la respuesta contiene una lista de notificaciones. |
| **GET /api/notifications/unread/:username** | Devuelve todas las notificaciones no leídas para un nombre de usuario válido. | 200, la respuesta contiene una lista de notificaciones no leídas. |
| **PATCH /api/notifications/notification/:notificationId/read** | Marca una notificación como leída. | 200, la respuesta indica éxito. |
| **PATCH /api/notifications/read/:username** | Marca todas las notificaciones de un usuario como leídas. | 200, la respuesta indica éxito. |

Tabla 6.65: Tests de purchasesRouter

| Tests de purchasesRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **POST /api/purchases/cardpack** | Compra un paquete de cartas exitosamente, disminuyendo la cantidad disponible del paquete y el saldo del usuario, creando cartas de usuario y transacciones. | 200, la respuesta indica éxito y las verificaciones post-compra son correctas. |
| **POST /api/purchases/cardpack (usuario no existe)** | Maneja errores cuando el usuario no existe. | 500, la respuesta contiene el mensaje 'El usuario no existe.'. |
| **POST /api/purchases/cardpack (saldo insuficiente)** | Maneja errores cuando el usuario no tiene suficiente saldo. | 500, la respuesta indica un error de saldo insuficiente. |
| **POST /api/purchases/cardpack (paquete no existe)** | Maneja errores cuando el paquete de cartas no existe. | 500, la respuesta indica que el paquete de cartas no se encuentra. |
| **POST /api/purchases/cardpack (paquete no disponible)** | Maneja errores cuando el paquete de cartas no está disponible. | 500, la respuesta indica que el paquete de cartas no está disponible. |

Tabla 6.66: Tests de transactionRouter

| Tests de transactionRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/transactions** | Devuelve todas las transacciones. | 200, la respuesta contiene una lista de transacciones. |
| **GET /api/transactions/u/:username** | Devuelve las transacciones para un nombre de usuario válido. | 200, la respuesta contiene una lista de transacciones para el usuario. |
| **GET /api/transactions/c/:userCardId** | Devuelve las transacciones para un ID de carta de usuario válido. | 200, la respuesta contiene una lista de transacciones para la carta de usuario. |
| **GET /api/transactions (no admin)** | Devuelve 403 si el usuario no es administrador. | 403, la respuesta contiene el mensaje 'Acceso denegado. Se requiere rol de administrador.'. |
| **GET /api/transactions/u/:username (username inválido)** | Devuelve 400 para un nombre de usuario inválido. | 400, la respuesta contiene errores de validación para el nombre de usuario. |

Tabla 6.67: Tests de userCardRouter

| Tests de userCardRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/usercards/u/:username** | Devuelve las tarjetas de usuario para un nombre de usuario válido. | 200, la respuesta contiene un arreglo de tarjetas de usuario. |
| **GET /api/usercards/:id** | Devuelve una tarjeta de usuario específica. | 200, la respuesta contiene la tarjeta de usuario con el campo 'legibleCardId'. |
| **GET /api/usercards/:id (no existe)** | Devuelve 404 si la tarjeta de usuario no se encuentra. | 404, la respuesta indica que la tarjeta de usuario no se encuentra. |
| **GET /api/usercards/:id (ID inválido)** | Devuelve 400 para un ID de tarjeta de usuario inválido. | 400, la respuesta indica un error. |

Continúa en la siguiente página...

Tabla 6.67 Tests de userCardRouter – continuación de la página anterior

| Tests de userCardRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **GET /api/usercards/u/:username (nombre de usuario muy largo)** | Devuelve 400 si el nombre de usuario es demasiado largo. | 400, la respuesta indica un error con un mensaje sobre la longitud del nombre de usuario. |

Tabla 6.68: Tests de userRouter

| Tests de userRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **POST /api/users/login** | Inicia sesión un usuario existente con el nombre de usuario 'test' y la contraseña 'Password123-'. | 200, la respuesta contiene un token y datos del usuario. |
| **POST /api/users/signup** | Crea un nuevo usuario con el nombre de usuario 'newuser' y la contraseña 'Password123-'. | 201, la respuesta contiene un mensaje de éxito y datos del nuevo usuario. |
| **GET /api/users/:username** | Obtiene los detalles del usuario 'test' con un token válido. | 200, la respuesta contiene los datos del usuario. |
| **PATCH /api/users/update/avatar** | Actualiza la imagen de perfil del usuario 'test' a 'avatar1.png'. | 200, la respuesta indica éxito. |
| **PATCH /api/users/update/pass** | Actualiza la contraseña del usuario 'test' a 'NewPass1234-'. | 200, la respuesta indica éxito. |
| **GET /api/users/:username (error handling)** | Devuelve 400 si el nombre de usuario es demasiado largo. | 400, la respuesta indica error. |
| **GET /api/users/:username (sin token)** | Devuelve 401 si no se proporciona un token. | 401, la respuesta indica error. |
| **GET /api/users/:username (usuario no encontrado)** | Devuelve 404 si el usuario no se encuentra. | 404, la respuesta contiene mensaje de usuario no encontrado. |
| **GET /api/users/:username (error)** | Maneja errores de forma adecuada. | 500, la respuesta indica error interno. |
| **POST /api/users/login (usuario no existe)** | Devuelve 401 si el usuario no existe. | 401, la respuesta indica error. |

Continúa en la siguiente página...

Tabla 6.68 Tests de userRouter – continuación de la página anterior

| Tests de userRouter | | |
|---|---|---|
| **Ruta a probar** | **Descripción** | **Resultado esperado** |
| **POST /api/users/login (contraseña incorrecta)** | Devuelve 401 si la contraseña es incorrecta. | 401, la respuesta indica error. |
| **POST /api/users/login (error)** | Maneja errores de forma adecuada. | 500, la respuesta indica error interno. |
| **POST /api/users/signup (usuario ya existe)** | Devuelve 400 si el nombre de usuario ya existe. | 400, la respuesta indica error. |
| **POST /api/users/signup (datos incompletos)** | Devuelve 400 si falta el nombre de usuario, contraseña o fecha de nacimiento. | 400, la respuesta indica error. |
| **POST /api/users/signup (error)** | Maneja errores de forma adecuada. | 500, la respuesta contiene mensaje de error y autenticación fallida. |

#### *6.7.2.2.2 Pruebas Unitarias. Webapp*

En el subsistema **webapp** se han realizado 9 pruebas unitarias automáticas, se han probado los componentes más importantes de la aplicación y todas han pasado con éxito. El resto de componentes se han probado de forma manual y se ha comprobado que su comportamiento es el esperado. Las pruebas unitarias automáticas se han realizado sobre los componentes que más se reutilizan en la aplicación, estos son:

- **Componente *Button***: Se ha comprobado que el componente *Button* renderiza correctamente, que se muestra el texto esperado y que se ejecuta la función *onClick* cuando se hace clic en el botón.
- **Componente *Calendar***: Se ha comprobado que el componente *Calendar* renderiza correctamente, que se ejecuta la función *onChange* cuando se selecciona una fecha y que si hay un error en la fecha se muestre un mensaje de error.
- **Componente *ErrorMessageBox***: Se ha comprobado que el componente *ErrorMessageBox* renderiza correctamente y que redirige a la página de inicio cuando se hace clic en el botón que contiene.
- **Componente *PokemonCard***: Se ha comprobado que la carta de Pokémon se renderiza correctamente y que se ejecutan los distintos *onClick* dependiendo de su configuración inicial.

En la figura Figura 6.52. Resultados de las pruebas unitarias automáticas de la webapp se muestran los resultados de las pruebas unitarias automáticas realizadas.

```
> webapp@0.1.0 test:unit
> jest --config jest.unit.config.ts

 PASS  src/__test__/unitTest/Calendar.test.tsx
 PASS  src/__test__/unitTest/Button.test.tsx
 PASS  src/__test__/unitTest/ErrorMessageBox.test.tsx
 PASS  src/__test__/unitTest/PokemonCard.test.tsx
 PASS  src/__test__/unitTest/UserMenu.test.tsx

Test Suites: 5 passed, 5 total
Tests:       9 passed, 9 total
Snapshots:   0 total
Time:        3.851 s, estimated 4 s
Ran all test suites.
```

Figura 6.52: Resultados de las pruebas unitarias automáticas de la webapp

#### 6.7.2.3. Pruebas de Integración

Las pruebas de integración se han realizado con éxito, se han comprobado que los distintos componentes del sistema funcionan correctamente en conjunto. Se han realizado pruebas de integración en el subsistema **webapp** con el framework jest-cucumber y Puppeteer. Se han realizado 3 pruebas de integración automáticas, todas ellas han pasado con éxito. Las pruebas que se han realizado son las siguientes:

- **Prueba de Inicio de Sesión Exitoso**: Se ha comprobado que el inicio de sesión funciona correctamente. De esta forma, se ha comprobado que el usuario puede iniciar sesión con sus credenciales y acceder a la aplicación, lo que significa que la conexión con el backend funciona correctamente y la integración de los componentes de la aplicación es correcta.
- **Prueba de Inicio de Sesión Fallido**: Se ha comprobado que el inicio de sesión falla cuando las credenciales son incorrectas. De esta forma, se ha comprobado que la aplicación responde correctamente a errores en el inicio de sesión.
- **Prueba de Registro de Usuario Fallido**: Se ha comprobado que el registro de usuario falla cuando los datos introducidos no son válidos. Se verifica que la aplicación responde correctamente a errores en el registro de usuario y se resaltan los campos con errores.

En la figura Figura 6.52. Resultados de las pruebas unitarias automáticas de la webapp se muestran los resultados de las pruebas de integración automáticas realizadas.

```
> webapp@0.1.0 test:e2e
> jest --config jest.e2e.config.ts

 PASS  src/__test__/e2e/steps/home.steps.ts (6.848 s)
 PASS  src/__test__/e2e/steps/signup.steps.ts (9.056 s)
 PASS  src/__test__/e2e/steps/login.steps.ts (18.151 s)

Test Suites: 3 passed, 3 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        18.408 s
Ran all test suites.
```

Figura 6.53: Resultados de las pruebas de integración automáticas de la webapp

El resto de pruebas de integración se han realizado de forma manual y se ha comprobado que el comportamiento de la aplicación es el esperado.

#### 6.7.2.4. Pruebas de Usabilidad

Se ha creado un formulario de evaluación de usabilidad que se les ha proporcionado a los usuarios para que evalúen la interfaz de usuario. Se han realizado pruebas de usabilidad con 3 usuarios reales, que han evaluado la interfaz de usuario y han proporcionado retroalimentación sobre su usabilidad. Estos usuarios, que no han participado en el desarrollo de la aplicación, han tenido que completar una serie de tareas y responder a preguntas sobre la usabilidad de la aplicación. Los usuarios tienen una experiencia variada en el uso de aplicaciones web y representan a los diferentes tipos de usuarios que utilizarán la aplicación. Los datos de los usuarios son los siguientes:

- **Usuario 1**: Hombre de 51 años, con poca experiencia en el uso de aplicaciones web. No ha utilizado aplicaciones similares anteriormente ni realizado compras en línea.
- **Usuario 2**: Mujer de 22 años, con mucha experiencia en el uso de aplicaciones web. Ha utilizado aplicaciones similares anteriormente y ha realizado compras en línea.
- **Usuario 3**: Mujer de 18 años, con experiencia en el uso de aplicaciones web. Ha utilizado aplicaciones similares anteriormente, pero no ha llegado a realizar compras en ellas.

##### *6.7.2.4.1 Cuestionario de usabilidad*

Se ha realizado un cuestionario de usabilidad con distintas preguntas para evaluar la calidad de la interfaz de usuario de la aplicación. Estas preguntas pretenden evaluar la calidad del diseño de la interfaz, la facilidad de uso y la satisfacción del usuario. Este cuestionario se ha proporcionado a los usuarios para que lo completen después de realizar las tareas asignadas. En la tabla Tabla 6.69. Cuestionario de usabilidad se muestra el cuestionario de usabilidad que se ha proporcionado a los usuarios.

Tabla 6.69: Cuestionario de usabilidad de la aplicación

| **Cuestionario de usabilidad de la aplicación** | | | | | |
|---|---|---|---|---|---|
| Navegabilidad de la Aplicación | | | | | |
| **Pregunta** | **1** | **2** | **3** | **4** | **5** |
| Es fácil de navegar por la aplicación | | | | | |
| Sabe cómo volver a la página principal | | | | | |
| Encuentra fácilmente la información que busca | | | | | |
| Sabe dónde está en la aplicación en todo momento | | | | | |
| Facilidad de Uso | | | | | |

Continúa en la siguiente página...

Tabla 6.69 Cuestionario de usabilidad de la aplicación – continuación de la página anterior

| **Cuestionario de usabilidad de la aplicación** | | | | | |
|---|---|---|---|---|---|
| **Pregunta** | **1** | **2** | **3** | **4** | **5** |
| ¿Le resulta sencillo utilizar la aplicación? | | | | | |
| ¿Le resulta fácil realizar las tareas asignadas? | | | | | |
| ¿Le resulta fácil poner en subasta una carta? | | | | | |
| ¿Le resulta fácil comprar un paquete de cartas? | | | | | |
| ¿Le resulta fácil consultar sus notificaciones? | | | | | |
| ¿Le resulta fácil consultar sus susbastas activas? | | | | | |
| ¿Le resulta fácil consultar sus pujas activas? | | | | | |
| Funcionalidad | | | | | |
| **Pregunta** | **Sí** | **No** | **Comentarios** | | |
| ¿El comportamiento de los botones es el esperado? | | | | | |
| ¿La aplicación responde de forma rápida? | | | | | |
| ¿Ha encontrado algún error en la aplicación? | | | | | |
| Calidad del Interfaz | | | | | |
| ¿La aplicación muestra la información de forma clara y concisa? | | | | | |
| ¿La aplicación muestra mensajes de error claros y útiles? | | | | | |
| ¿Le resulta útil la información proporcionada? | | | | | |
| ¿Los colores empleados son agradables y fáciles de leer? | | | | | |
| ¿Los iconos utilizados son comprensibles y descriptivos? | | | | | |
| ¿La aplicación es visualmente atractiva? | | | | | |
| ¿La estructura de la aplicación es clara y fácil de entender? | | | | | |

Continúa en la siguiente página...

Tabla 6.69 Cuestionario de usabilidad de la aplicación – continuación de la página anterior

| **Cuestionario de usabilidad de la aplicación** | | | | | |
|---|---|---|---|---|---|
| Satisfacción del Usuario | | | | | |
| **Pregunta** | **1** | **2** | **3** | **4** | **5** |
| ¿Está satisfecho con la aplicación? | | | | | |
| ¿Recomendaría la aplicación a otras personas? | | | | | |
| ¿Volvería a utilizar la aplicación en el futuro? | | | | | |
| Comentarios Adicionales | | | | | |
| | | | | | |

#### *6.7.2.4.2 Tareas asignadas a los usuarios*

Se les ha asignado una serie de tareas a los usuarios para que realicen durante las pruebas de usabilidad. Estas tareas han sido diseñadas para evaluar la facilidad de uso y la eficacia de la aplicación. Están ordenadas de forma que los usuarios puedan completarlas de forma lógica y secuencial.

- **Tarea 1**: Crear una cuenta de usuario.
- **Tarea 2**: Iniciar sesión en la aplicación.
- **Tarea 3**: Cambiar la imagen de perfil.
- **Tarea 4**: Adquirir un sobre de cartas.
- **Tarea 5**: Consultar colección de cartas.
- **Tarea 6**: Poner en subasta una carta.
- **Tarea 7**: Consultar subastas activas de todos los usuarios.
- **Tarea 8**: Consultar sus propias subastas activas.
- **Tarea 9**: Consultar sus propias pujas activas.
- **Tarea 10**: Retirar una carta de una subasta.
- **Tarea 11**: Pujar por una carta en subasta.
- **Tarea 12**: Consultar notificaciones.
- **Tarea 13**: Marcar una notificación como leída.
- **Tarea 14**: Consultar transacciones.
- **Tarea 15**: Cerrar sesión.

#### 6.7.2.4.3 *Cuestionario para el responsable de pruebas*

Se ha creado un cuestionario para el responsable de pruebas con distintas preguntas para evaluar el resultado de las pruebas de usabilidad realizadas. Estas preguntas pretenden evaluar los comportamientos observados en los distintos usuarios a la hora de realizar las tareas asignadas.

En la tabla Tabla 6.70. Cuestionario para el responsable de pruebas se muestra el cuestionario para el responsable de pruebas. Deberá de marcar con una 'X' en la columna correspondiente si la tarea ha sido completada o no por el usuario, y añadir cualquier comentario adicional que considere relevante.

Tabla 6.70: Cuestionario para el responsable de pruebas

| Tarea | Sí | No | Comentarios |
|---|---|---|---|
| **Tarea 1**: Crear una cuenta de usuario | | | |
| **Tarea 2**: Iniciar sesión en la aplicación | | | |
| **Tarea 3**: Cambiar la imagen de perfil | | | |
| **Tarea 4**: Adquirir un sobre de cartas | | | |
| **Tarea 5**: Consultar colección de cartas | | | |
| **Tarea 6**: Poner en subasta una carta | | | |
| **Tarea 7**: Consultar subastas activas de todos los usuarios | | | |
| **Tarea 8**: Consultar sus propias subastas activas | | | |
| **Tarea 9**: Consultar sus propias pujas activas | | | |
| **Tarea 10**: Retirar una carta de una subasta | | | |
| **Tarea 11**: Pujar por una carta en subasta | | | |
| **Tarea 12**: Consultar notificaciones | | | |
| **Tarea 13**: Marcar una notificación como leída | | | |
| **Tarea 14**: Consultar transacciones | | | |
| **Tarea 15**: Cerrar sesión | | | |

#### 6.7.2.4.4 *Resultados de las pruebas de usabilidad*

Se han recopilado los resultados de las pruebas de usabilidad realizadas con los usuarios y el responsable de pruebas. Se han analizado los resultados y se han identificado los problemas de usabilidad y las áreas de mejora de la aplicación. En la tabla Tabla 6.71. Resultados de las pruebas de usabilidad, se muestran la recopilación de los resultados de las pruebas de usabilidad realizadas.

Tabla 6.71: Resultados de las pruebas de usabilidad

| Usuario | Sí | No | Comentarios |
|---|---|---|---|
| **Tarea 1. Crear una cuenta de usuario** | | | |
| **Usuario 1** | X | | Tras varios intentos ha conseguido crear una cuenta de usuario. Problema entendiendo las restricciones del nombre de usuario y contraseña. |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | Tras un intento fallido ha conseguido crear una cuenta de usuario. |
| **Tarea 2. Iniciar sesión en la aplicación** | | | |
| **Usuario 1** | X | | |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | |
| **Tarea 3. Cambiar la imagen de perfil** | | | |
| **Usuario 1** | | X | Como la imagen de perfil cambia automáticamente en la información del perfil, no confirmaba el cambio. |
| **Usuario 2** | X | | Ha sugerido una opción para eliminar la imagen de perfil. |
| **Usuario 3** | X | | |
| **Tarea 4. Adquirir un sobre de cartas** | | | |
| **Usuario 1** | X | | Sin querer ha cerrado el *modal* en el que se muestran las cartas adquiridas antes de verlas. |
| **Usuario 2** | X | | Ha encontrado un error al adquirir un sobre de cartas y seguidamente volver a intentar adquirir el mismo sobre. Este error ya ha sido corregido en la versión actual de la aplicación. |
| **Usuario 3** | X | | |
| **Tarea 5. Consultar colección de cartas** | | | |
| **Usuario 1** | X | | |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | |
| **Tarea 6. Poner en subasta una carta** | | | |
| **Usuario 1** | | X | No ha encontrado la opción para poner en subasta una carta. |
| **Usuario 2** | X | | |
| **Usuario 3** | | X | Finalmente, ha encontrado la opción para poner en subasta una carta, pero se considera como una tarea poco intuitiva ya que ha buscado en la sección de subastas en lugar de en la sección de colección de cartas. |
| **Tarea 7. Consultar subastas activas de todos los usuarios** | | | |
| **Usuario 1** | X | | |
| **Usuario 2** | X | | |

Continúa en la siguiente página...

Tabla 6.71 Resultados de las tareas de las pruebas de usabilidad – continuación de la página anterior

| Usuario | Sí | No | Comentarios |
|---|---|---|---|
| **Usuario 3** | X | | |
| **Tarea 8. Consultar sus propias subastas activas** | | | |
| **Usuario 1** | | X | No entendía el funcionamiento del *slider* para ver sus propias subastas activas. Se ha añadido un mensaje de ayuda para explicar el funcionamiento del *slider*. |
| **Usuario 2** | X | | Ha indicado como mejora que la interfaz indicase de forma más clara cuáles son sus propias subastas activas. En la versión actual se cambia el título y añade un mensaje de ayuda para indicar que se están viendo las subastas activas del usuario. |
| **Usuario 3** | X | | |
| **Tarea 9. Consultar sus propias pujas activas** | | | |
| **Usuario 1** | X | | Al principio, intentaba buscar la opción de consultar sus propias pujas activas en la sección de subastas activas. |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | |
| **Tarea 10. Retirar una carta de una subasta** | | | |
| **Usuario 1** | X | | |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | |
| **Tarea 11. Pujar por una carta en subasta** | | | |
| **Usuario 1** | X | | |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | |
| **Tarea 12. Consultar notificaciones** | | | |
| **Usuario 1** | X | | |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | Intentaba acceder al detalle de la notificación haciendo clic en la notificación. La aplicación no permite acceder al detalle de la notificación. |
| **Tarea 13. Marcar una notificación como leída** | | | |
| **Usuario 1** | X | | Al principio, intentaba marcar una notificación como leída haciendo clic en la notificación. |
| **Usuario 2** | X | | Al principio, intentaba marcar una notificación como leída haciendo clic en la notificación. |
| **Usuario 3** | X | | Al principio, intentaba marcar una notificación como leída haciendo clic en la notificación. |

Continúa en la siguiente página...

Tabla 6.71 Resultados de las tareas de las pruebas de usabilidad – continuación de la página anterior

| Usuario | Sí | No | Comentarios |
|---|---|---|---|
| **Tarea 14. Consultar transacciones** | | | |
| **Usuario 1** | X | | Ha indicado como mejora que la interfaz indicase de forma más clara cuáles son las transacciones de compra y venta. |
| **Usuario 2** | X | | Ha encontrado las notificaciones de adquisición de cartas mediante sobre poco informativas. |
| **Usuario 3** | X | | No entendía para que servía ver el identificador de la carta involucrada en la transacción. |
| **Tarea 15. Cerrar sesión** | | | |
| **Usuario 1** | | X | Ha cerrado la pestaña del navegador en lugar de cerrar sesión. |
| **Usuario 2** | X | | |
| **Usuario 3** | X | | |

En base a los resultados de las pruebas de usabilidad, se han identificado los siguientes problemas de usabilidad y áreas de mejora de la aplicación:

- **Problema 1**: Falta de mensajes de ayuda en algunas secciones de la aplicación. En la sección de subastas activas, se ha añadido un mensaje de ayuda para explicar el funcionamiento del *slider*. De igual forma, en la sección de subastas activas propias se ha cambiado el título y añadido un mensaje de ayuda para indicar que se están viendo las subastas activas del usuario.
- **Problema 2**: Falta de claridad en la navegación de la aplicación.
- **Problema 4**: Falta de claridad en la información proporcionada en las transacciones. Para una próxima versión se añadirá una Descripción más completa de la descripción y de los activos involucrados en la transacción. Además, se actualizará el diseño de tal manera que se diferencie claramente entre las transacciones de compra y venta.
- **Problema 5**: Falta de claridad en el botón de marcar la notificación como leída. Se ha añadido un *tooltip* para indicar que el botón sirve para marcar la notificación como leída.

#### 6.7.2.5. Pruebas de Accesibilidad

Una vez desplegada la aplicación en un entorno de producción, se ha realizado una auditoría de accesibilidad para comprobar que la aplicación cumple con los estándares de accesibilidad web. El proceso de auditoría de accesibilidad se ha realizado con el plugin de Google Chrome WAVE, que proporciona una serie de recomendaciones para mejorar la accesibilidad de la aplicación. Se han identificado una serie de problemas de accesibilidad en la aplicación y se han corregido para mejorar la accesibilidad de la aplicación. Una vez corregidos los problemas de accesibilidad, se ha vuelto a realizar la auditoría de accesibilidad para comprobar que la aplicación cumple con los estándares de accesibilidad web.

Los problemas de accesibilidad identificados y corregidos en la aplicación son los siguientes:

- **Problema 1**: Falta de etiquetas en los formularios.

- **Problema 2**: Falta de etiquetas en los botones.
- **Problema 3**: Falta de etiquetas en las imágenes.
- **Problema 4**: Falta de contraste en los colores.
- **Problema 5**: Falta de descripción en los enlaces.
- **Problema 6**: Falta de etiquetas en las tablas.
- **Problema 7**: Falta de etiquetas en los elementos de formulario.
- **Problema 8**: Falta de etiquetas en los elementos de navegación.

Se asumen algunos de los problemas de contraste de colores, ya que la aplicación tiene un diseño específico y se ha optado por mantener el diseño original. Concretamente, los errores de contraste se dan en el diseño de las cartas. Estos son elegidos de forma dinámica por el tipo de carta que representa y se ha optado por mantener el diseño original, asumiendo que no afecta a la usabilidad de la aplicación debido a que la información de la carta es accesible de otras formas.

En el Anexo. Accesibilidad de la aplicación muestra el resultado de la auditoría de accesibilidad realizada con el plugin WAVE de Google Chrome para las páginas principales de la aplicación.

#### 6.7.2.6. Pruebas de Adaptabilidad

Se han realizado pruebas de adaptabilidad de la aplicación en distintos dispositivos y tamaños de pantalla para comprobar que la aplicación se adapta correctamente a diferentes resoluciones y tamaños de pantalla. Se han probado la aplicación en dispositivos móviles, tabletas y ordenadores de escritorio para comprobar que la aplicación se ve correctamente en todos los dispositivos.

Concretamente, se han probado los siguientes dispositivos reales:

- **Dispositivos móviles**: Se ha probado la aplicación en un dispositivo móvil iPhone 12.
- **Tabletas**: Se ha probado la aplicación en una tableta iPad 8ª generación.
- **Ordenadores**: Se ha probado la aplicación en un ordenador de 16 pulgadas.
- **Monitor externo**: Se ha probado la aplicación en un monitor externo de 24 pulgadas.

Además, se han probado la aplicación en distintos navegadores y sistemas operativos para comprobar que la aplicación se ve correctamente en todos los navegadores y sistemas operativos. También se ha comprobado con las herramientas:

- Google Chrome DevTools
- Screenfly de BlueTree
- Extensión de Chrome Mobile FIRST

Como resultado de las pruebas de adaptabilidad, se ha comprobado que la aplicación se adapta correctamente a diferentes resoluciones y tamaños de pantalla. Se pueden ver las capturas de pantalla de la aplicación en los distintos dispositivos en el Anexo. Adaptabilidad de la aplicación.

## 6.8. DISEÑO FÍSICO DE DATOS

En este apartado se abordará el diseño físico de datos del sistema, que se encarga de definir cómo se almacenarán los datos en la base de datos.

#### 6.8.0.1. Descripción del SGBD utilizado

Para el almacenamiento de los datos se ha utilizado el sistema de gestión de bases de datos (SGBD) MongoDB, que es una base de datos NoSQL orientada a documentos. Concretamente se ha utilizado su versión en la nube, MongoDB Atlas.

MongoDB es una base de datos NoSQL que almacena los datos en documentos BSON (Binary JSON), que son una representación binaria de JSON. Estos documentos se almacenan en colecciones, que son agrupaciones de documentos.

MongoDB es una base de datos muy flexible, ya que no requiere que los documentos de una misma colección tengan la misma estructura. Aún así, en este proyecto se ha definido una estructura fija para los documentos de cada colección, para facilitar la gestión de los datos.

La integración de MongoDB Atlas en el proyecto se ha realizado mediante la librería Mongoose.

#### 6.8.0.2. Documentos definidos

A continuación se detallan los documentos definidos para cada colección de la base de datos junto con una breve descripción de cada uno.

- **Auction (Subasta)**: Representa una subasta donde se subasta una carta de usuario (UserCard).
- **UserCard (Carta de Usuario)**: Representa una carta de Pokémon que pertenece a un usuario.
- **User (Usuario)**: Representa un usuario del sistema.
- **Bid (Puja)**: Representa una puja realizada por un usuario en una subasta.
- **Card (Carta)**: Representa una carta de Pokémon con detalles específicos sobre el Pokémon, la carta y sus transacciones.
- **CardPack (Sobre de Cartas)**: Representa un sobre de cartas de Pokémon que contiene varias cartas.
- **Deck (Mazo de cartas)**: Representa un mazo de cartas que agrupa varias cartas de Pokémon.
- **Notification (Notificación)**: Representa una notificación enviada a un usuario.
- **Transaction (Transacción)**: Representa una transacción que involucra la compra o venta de una carta de usuario (UserCard).

También se han definido enumeraciones para los campos que requieren un valor de una lista predefinida, como el estado de una subasta o el tipo de una carta. Estos campos se almacenan como cadenas de texto en la base de datos, pero se han definido enumeraciones en el código para facilitar su uso. Estos son:

- **AuctionStatus (Estado de Subasta)**: Enumera los posibles estados de una subasta.
- **BidStatus (Estado de Puja)**: Enumera los posibles estados de una puja.
- **CardRarity (Rareza de Carta)**: Enumera los diferentes niveles de rareza de una carta de Pokémon.
- **PokemonType (Tipo de Pokémon)**: Enumera los diferentes tipos de Pokémon.
- **PokemonGym (Gimnasio Pokémon)**: Enumera los diferentes gimnasios Pokémon donde pueden encontrarse los Pokémon.
- **NotificationType (Tipo de Notificación)**: Enumera los diferentes tipos de notificaciones.
- **NotificationImportance (Importancia de la Notificación)**: Enumera los diferentes niveles de importancia de una notificación.
- **TransactionConcept (Concepto de Transacción)**: Enumera los diferentes conceptos para las transacciones realizadas.

#### 6.8.0.3. Modelo de datos

A continuación se muestra el modelo de datos de la base de datos, que representa las colecciones y los campos de cada documento.

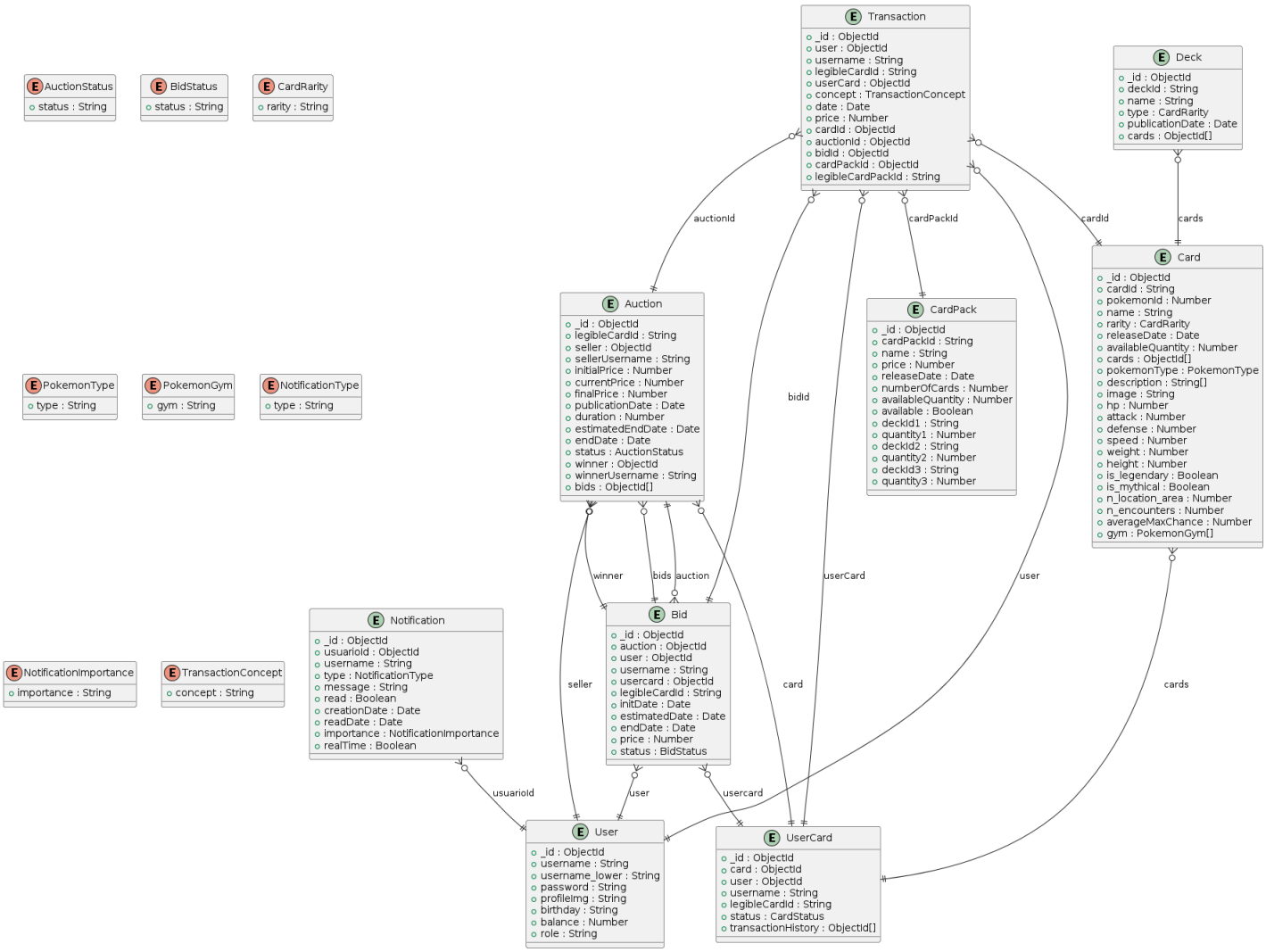


Figura 6.54: Modelo de datos de la base de datos MongoDB

Capítulo 7

# CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN

## 7.1. PREPARACIÓN DEL ENTORNO DE GENERACIÓN Y CONSTRUCCIÓN

### 7.1.1. Herramientas y programas usados para el desarrollo

Para el desarrollo de la aplicación se han utilizado diversas herramientas y programas que han facilitado la creación del sistema.

#### 7.1.1.1. Lenguaje de programación

El sistema ha sido desarrollado en su totalidad en TypeScript, tanto el *frontend* como el *backend*.

TypeScript es un superconjunto de JavaScript que añade tipado estático al lenguaje. Esto permite detectar errores en tiempo de compilación y facilita el mantenimiento del código ya que es mucho más descriptivo.


Figura 7.1: Logo de TypeScript

#### 7.1.1.2. Entorno de ejecución

Se ha utilizado Node.js como entorno de ejecución para JavaScript. Node.js permite ejecutar código JavaScript en el servidor y es muy popular en el desarrollo de aplicaciones web. A través de npm, el gestor de paquetes de Node.js, se han instalado las dependencias necesarias para el desarrollo de la aplicación. Este gestor de paquetes permite instalar y gestionar las dependencias de un proyecto de forma sencilla y cuenta con un amplio repositorio de paquetes.


Figura 7.2: Logo de Node.js

#### 7.1.1.3. Base de datos

La base de datos utilizada en el sistema es MongoDB, una base de datos NoSQL orientada a documentos. Se ha utilizado MongoDB Atlas como servicio de base de datos en la nube, lo que ha permitido desplegar la base de datos de forma sencilla y segura.


Figura 7.3: Logo de MongoDB

#### 7.1.1.4. Proveedor de servicios en la nube

Para el despliegue de la aplicación se ha utilizado Azure, la plataforma en la nube de Microsoft. Azure ofrece una amplia gama de servicios en la nube, como servidores virtuales, bases de datos, almacenamiento y servicios de inteligencia artificial.


Figura 7.4: Logo de Microsoft Azure

#### 7.1.1.5. Librerías y frameworks

Se han utilizado diversas librerías y frameworks para el desarrollo de la aplicación, tanto en el *frontend* como en el *backend*. Este tipo de herramientas permiten acelerar el desarrollo y facilitan la creación de interfaces de usuario y la gestión de la lógica de negocio.

#### 7.1.1.6. Frontend

El *frontend* de la aplicación ha sido desarrollado en React, una librería de JavaScript para la creación de interfaces de usuario, en combinación con Material-UI, una librería de componentes de React, se ha creado una interfaz de usuario moderna y atractiva.

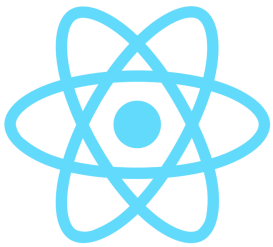

Figura 7.5: Logo de React

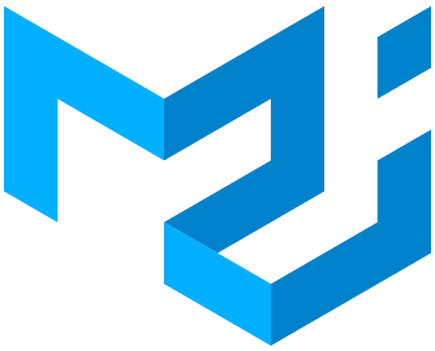

Figura 7.6: Logo de Material-UI

El resto de dependencias que cabe destacar utilizadas en el *frontend* son:

- `redux`: Biblioteca para el manejo del estado de la aplicación.

- `framer-motion`: Librería para animaciones en React.
- `socket.io-client`: Cliente para la comunicación en tiempo real con WebSockets.
- `styled-components`: Librería para escribir CSS en JavaScript.
- `yup`: Librería para la validación de esquemas de datos.
- `sweetalert2`: Librería para mostrar alertas personalizadas.
- `react-router-dom`: Librería para la navegación en React.
- `react-hook-form`: Librería para la gestión de formularios en React.
- `react-paypal-js`: Librería para la integración de pagos con PayPal.
- `react-slick`: Librería para la creación de carruseles en React.
- `Jest`: Framework de pruebas para JavaScript.
- `Puppeteer`: Librería para realizar pruebas de extremo a extremo.
- `Cucumber`: Herramienta para pruebas.

#### 7.1.1.7. Backend

El *backend* de la aplicación ha sido desarrollado con Express, un framework de Node.js para la creación de aplicaciones web y APIs. Express es muy popular en el desarrollo de aplicaciones web y permite crear servidores de forma sencilla y rápida.


Figura 7.7: Logo de Express

Las dependencias más destacadas utilizadas en el *backend* son:

- `axios`: Cliente HTTP basado en promesas para Node.js y el navegador.
- `bcrypt`: Librería para el hashing de contraseñas.
- `body-parser`: Middleware para analizar cuerpos de solicitudes HTTP en Node.js.
- `cors`: Middleware para habilitar CORS (Cross-Origin Resource Sharing).
- `jsonwebtoken`: Implementación de JSON Web Tokens.
- `mongoose`: Librería de modelado de datos de MongoDB para Node.js.
- `socket.io`: Biblioteca para aplicaciones web en tiempo real.

- `yup`: Librería para la validación de esquemas de datos.
- `csv-parser`: Librería para analizar archivos CSV.
- `csv-writer`: Librería para escribir archivos CSV.
- `paypal-rest-sdk`: SDK para la integración con PayPal.
- `Jest`: Framework de pruebas para JavaScript.
- `supertest`: Biblioteca para pruebas HTTP.

#### 7.1.1.8. Entorno de desarrollo

##### *7.1.1.8.1 Gestión de versiones*

Para la gestión de versiones se ha utilizado Git, un sistema de control de versiones distribuido que permite llevar un control de los cambios en el código fuente. Como plataforma de alojamiento de repositorios se ha utilizado GitHub, que permite alojar proyectos de software y facilita la colaboración entre desarrolladores. Además, se ha utilizado GitHub Actions para la integración continua y GitKraken como cliente de escritorio para la gestión de ramas y *pull requests*.

Figura 7.8: Logo de Git

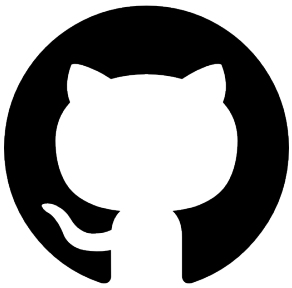

Figura 7.9: Logo de GitHub

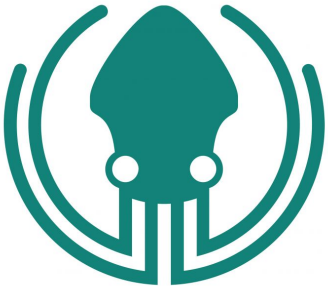

Figura 7.10: Logo de GitKraken

##### *7.1.1.8.2 Editor de código*

Para el desarrollo del código fuente se ha utilizado Visual Studio Code, un editor de código fuente desarrollado por Microsoft que incluye soporte para depuración, control de versiones y resaltado de sintaxis. Admite extensiones que permiten ampliar sus funcionalidades y es muy popular entre los desarrolladores.

Figura 7.11: Logo de Visual Studio Code

Además, se ha utilizado Notepad++ como editor de texto para la edición de archivos de configuración y otros archivos de texto necesarios para desplegar la aplicación en un servidor.


Figura 7.12: Logo de Notepad++

#### 7.1.1.9. Cliente SSH

Para la conexión remota con el servidor se ha utilizado MobaXterm, un cliente SSH que permite conectarse a servidores remotos de forma segura. Tiene una interfaz gráfica intuitiva y permite la transferencia de archivos a través de SCP y SFTP. Se ha utilizado principalmente para la conexión con el servidor de producción y para la transferencia de archivos.


Figura 7.13: Logo de MobaXterm

#### 7.1.1.10. Postman

Para probar las rutas de la API se ha utilizado Postman, una herramienta que permite realizar peticiones HTTP a servidores y analizar las respuestas. De esta manera se ha comprobado el funcionamiento de la API sin necesidad de utilizar el *frontend*, lo que ha facilitado el desarrollo y la depuración de la aplicación.


Figura 7.14: Logo de Postman

#### 7.1.1.11. Documentación

##### *7.1.1.11.1 Memoria del proyecto*

La documentación del proyecto se ha realizado con LaTeX, un sistema de composición de textos que permite crear documentos de alta calidad tipográfica. Se ha utilizado la herramienta Overleaf para la visualización y edición de los documentos, ya que permite la colaboración en tiempo real y la exportación a PDF.

Figura 7.15: Logo de LaTeX

Figura 7.16: Logo de Overleaf

#### *7.1.1.11.2 Diagramas*

Para la creación de diagramas se han utilizado diversas herramientas, como Lucidchart y Draw.io, que permiten crear diagramas de forma sencilla y exportarlos en varios formatos. Se ha utilizado PlantUML para la creación de diagramas de forma textual, lo que facilita la creación y modificación de los diagramas. Por último, se ha utilizado Excalidraw para la creación de las interfaces de usuario de la aplicación.

Figura 7.17: Logo de Lucidchart

Figura 7.18: Logo de Draw.io

Figura 7.19: Logo de PlantUML

Figura 7.20: Logo de Excalidraw

#### *7.1.1.11.3 Hojas de cálculo*

Para la creación de hojas de cálculo se ha utilizado Microsoft Excel, una herramienta de Microsoft Office que permite crear y editar hojas de cálculo. Se ha utilizado principalemente para la elaboración de presupuestos y cálculos de costes.

Figura 7.21: Logo de Microsoft Excel

#### *7.1.1.11.4 Planificación*

Para la planificación del proyecto se ha utilizado Microsoft Project, una herramienta de Microsoft Office que permite crear diagramas de Gantt y planificar tareas y recursos. Se ha utilizado para la planificación del proyecto y el seguimiento de las tareas.

Figura 7.22: Logo de Microsoft Project

## 7.2. ELABORACIÓN DE LOS MANUALES DE USUARIO

### 7.2.1. Manual de Instalación y Ejecución

El código fuente se adjunta en un archivo comprimido junto con la memoria del proyecto. Por lo que para instalar y ejecutar la aplicación se debe descomprimir el archivo y seguir las instrucciones que se detallan a continuación.

1. Clonar el repositorio de GitHub en un entorno local.
2. Instalar Node.js y npm en el sistema.
   - Para instalar Node.js se puede descargar el instalador desde la página oficial de Node.js nodejs.org.
   - npm se instala automáticamente con Node.js.
   - Para comprobar que Node.js y npm se han instalado correctamente, se puede ejecutar el siguiente comando en una terminal:
     - `node -v`
     - `npm -v`
3. Instalar las dependencias de *webapp*. Para ello hay que ejecutar los siguientes comandos:
   *a*) `cd webapp`
   *b*) `npm install`
4. Instalar las dependencias de *restapi*. Para ello hay que ejecutar los siguientes comandos:
   *a*) `cd restapi`
   *b*) `npm install`
5. Configurar las variables de entorno.
   *a*) Crear un archivo `.env` en la carpeta `restapi` con las siguientes variables de entorno:
   - `MONGO_URI`: URI de la base de datos de MongoDB.
   - `TOKEN_SECRET`: Clave secreta para la generación de tokens JWT.
   - `TEST_MONGO_URI`: URI de la base de datos de pruebas de MongoDB.
   - `PAYPAL_CLIENT_ID`: ID de cliente de PayPal.
   - `PAYPAL_CLIENT_SECRET`: Clave secreta de PayPal.
   - `NODE_ENV=development`: Entorno de desarrollo.
6. Iniciar la aplicación. Para ello hay que ejecutar los siguientes comandos:
   *a*) `cd backend`, si no se está en la carpeta del *backend*.
   *b*) `npm start`
   *c*) `cd frontend`, si no se está en la carpeta del *frontend*.
   *d*) `npm start`
7. Acceder a la aplicación en un navegador web a través de la dirección `http://localhost:3000`.

### 7.2.2. Manual de Usuario

El manual de usuario describe las funcionalidades de la aplicación y cómo utilizarlas. Se divide en varias secciones, cada una de las cuales describe una parte de la aplicación y cómo interactuar con ella.

#### 7.2.2.1. Inicio de sesión

Para acceder a la aplicación es necesario iniciar sesión con un usuario y una contraseña. Si no se dispone de una cuenta, se puede crear una nueva cuenta pulsando en el enlace de registro. Una vez iniciada la sesión, se accede a la página principal de la aplicación. El usuario recibe como bienvenida 100 Zens, la moneda virtual de la aplicación, que se pueden utilizar para comprar sobres de cartas y participar en subastas.

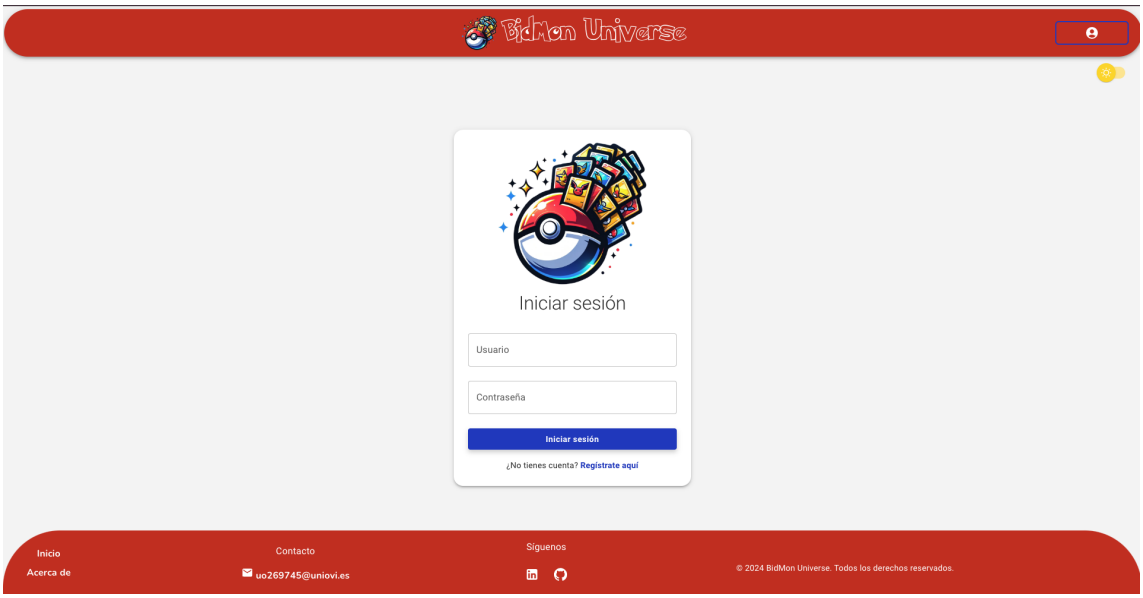


Figura 7.23: Página de inicio de sesión.

#### 7.2.2.2. Registro

Para registrarse en la aplicación es necesario introducir un nombre de usuario, la fecha de nacimiento y una contraseña. Los usuarios menores de 18 años no pueden registrarse en la aplicación. Una vez completado el registro, se puede iniciar sesión con el nuevo usuario y acceder a la página principal de la aplicación.

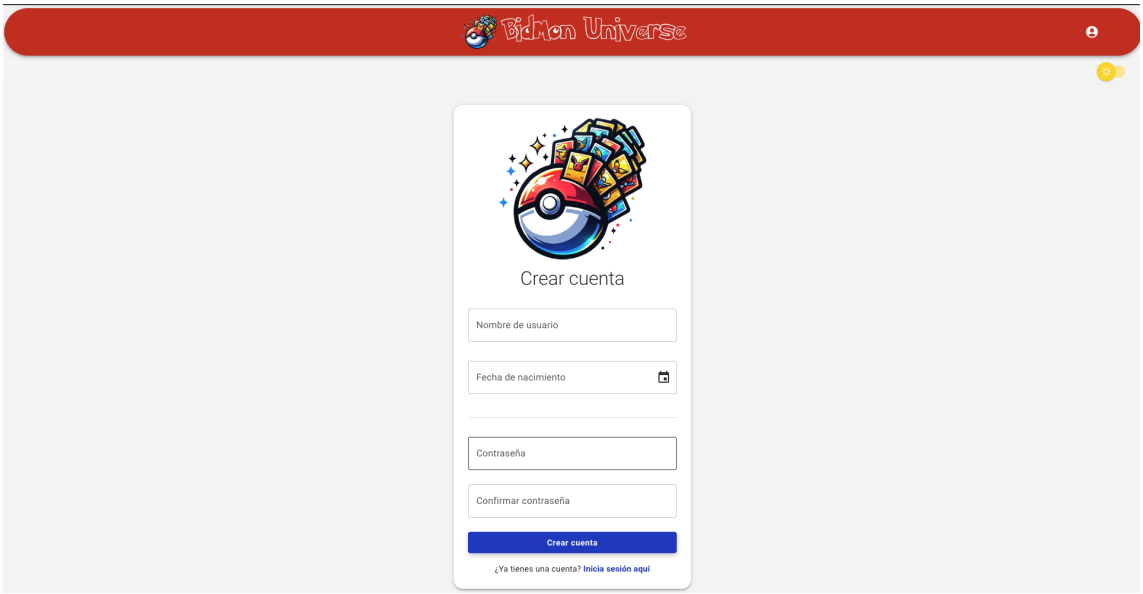


Figura 7.24: Página de registro de usuario.

#### 7.2.2.3. Página principal

En la página principal podemos consultar un resumen de la colección de cartas del usuario, el saldo de Zens y un resumen de las subastas que el usuario ha creado así como un resumen de las pujas activas. Desde esta página se puede acceder a las diferentes secciones de la aplicación, como la colección de cartas, las subastas, la tienda y el histórico de transacciones.

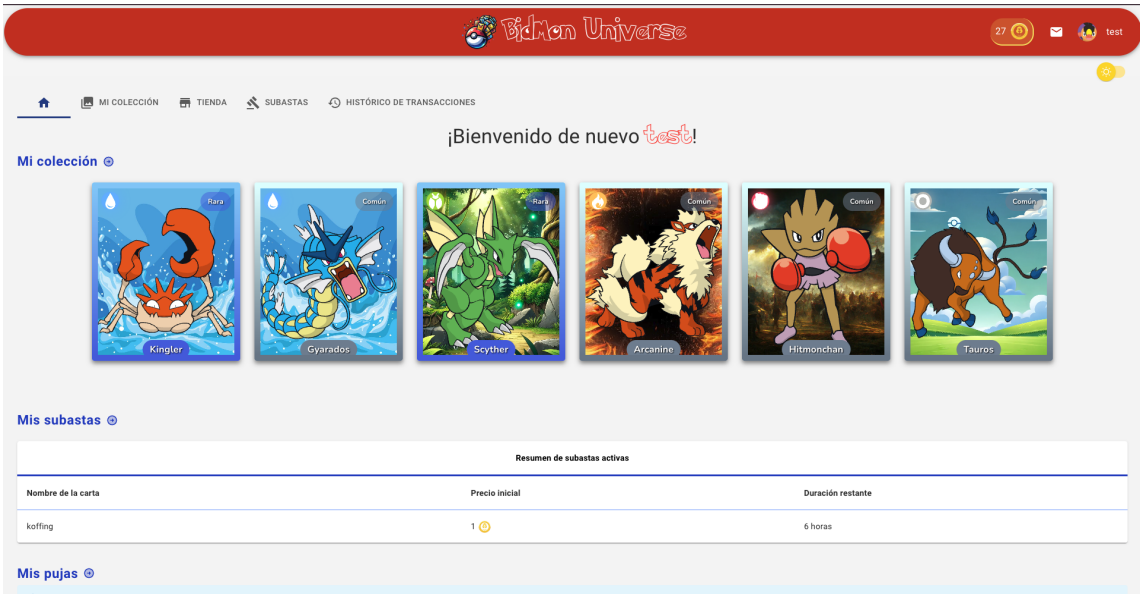


Figura 7.25: Página principal de la aplicación, una vez que el usuario ha iniciado sesión.

#### 7.2.2.4. Compra de cartas

En la tienda se pueden comprar sobres de cartas con Zens. Cada sobre contiene un número determinado de cartas, que se añaden a la colección del usuario. Estas cartas se pueden vender en subastas.

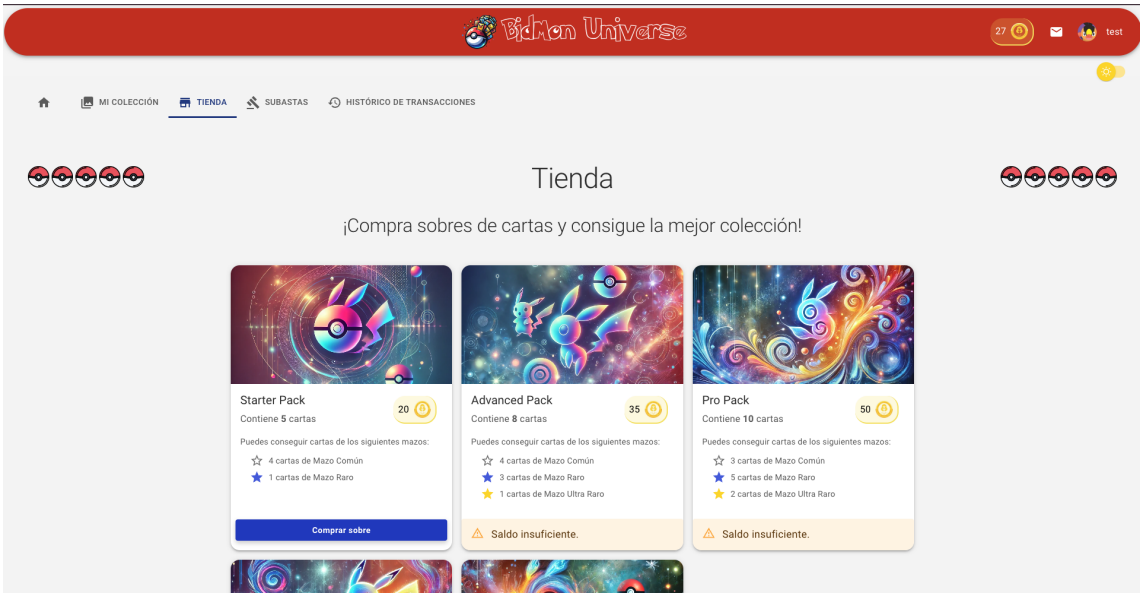


Figura 7.26: Página de la tienda de sobres de cartas.

Cuando el usuario compra un sobre, se le muestran las cartas que ha adquirido. Para simular la compra de un sobre real, las cartas aparecen inicialmente boca abajo y es el usuario quien tiene que hacer clic en cada carta para que esta se dé la vuelta y se muestre.

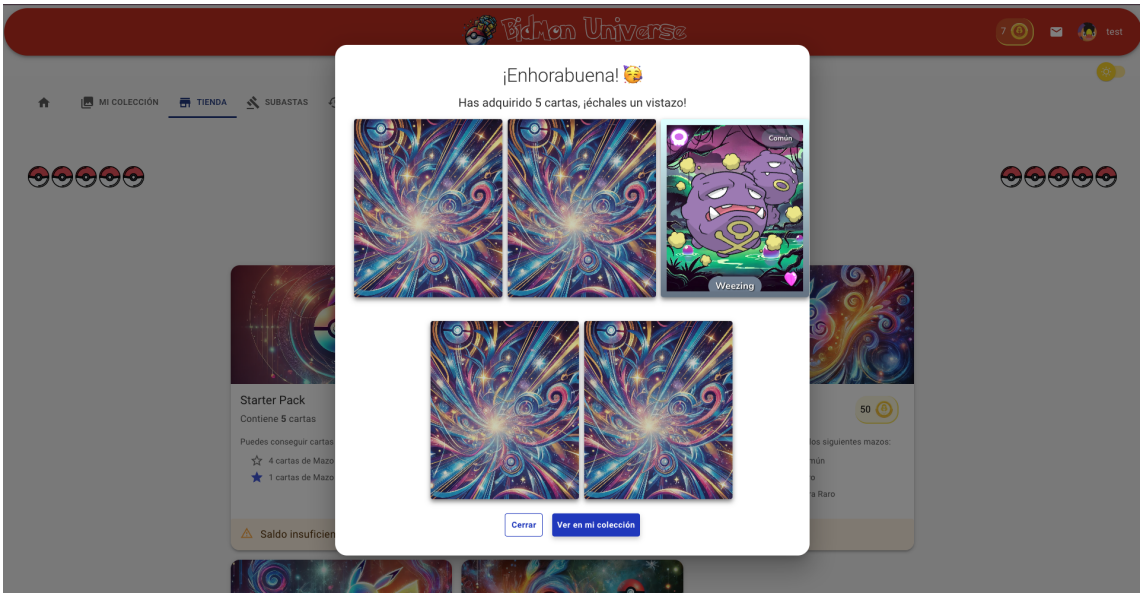


Figura 7.27: Modal de cartas obtenidas en el sobre.

Una vez que el usuario ha adquirido las cartas, puede verlas en la colección de cartas.

#### 7.2.2.5. Colección de cartas

En la colección de cartas se pueden ver todas las cartas que el usuario ha adquirido. Si se hace clic en una carta, se muestra un detalle de la carta con información adicional, como el nombre, el tipo, la rareza y el historial de transacciones de la carta.

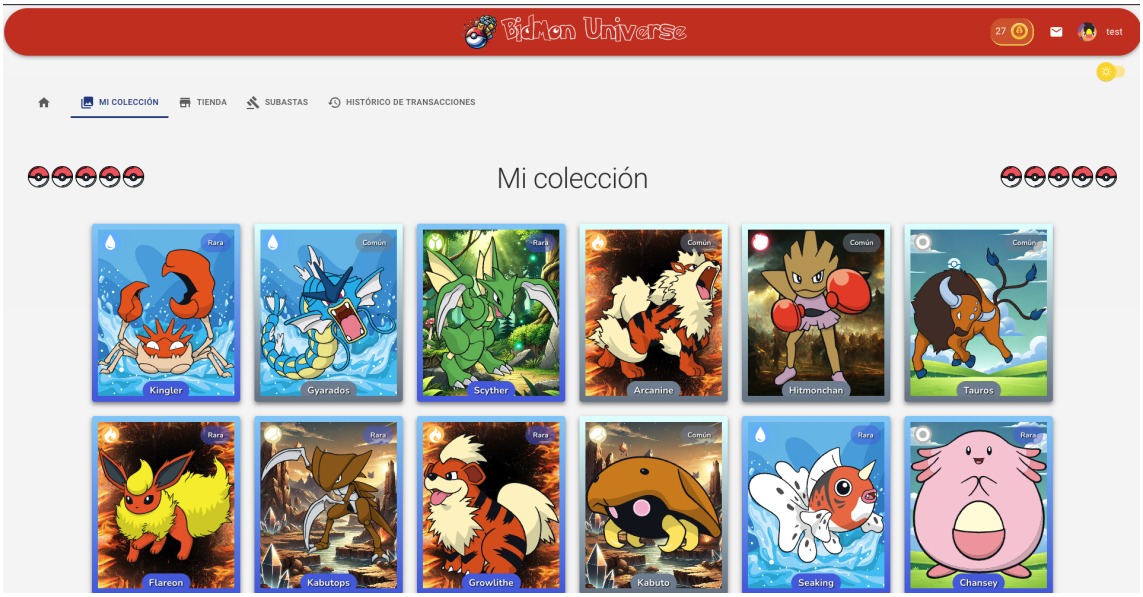


Figura 7.28: Página de la colección de cartas del usuario.

#### 7.2.2.6. Crear subasta

Para crear una subasta de una carta, el usuario debe acceder a su colección de cartas y seleccionar la carta que desea subastar. Una vez en la página de detalle de la carta, se puede pulsar el botón de *Realizar subasta* para iniciar el proceso de subasta.

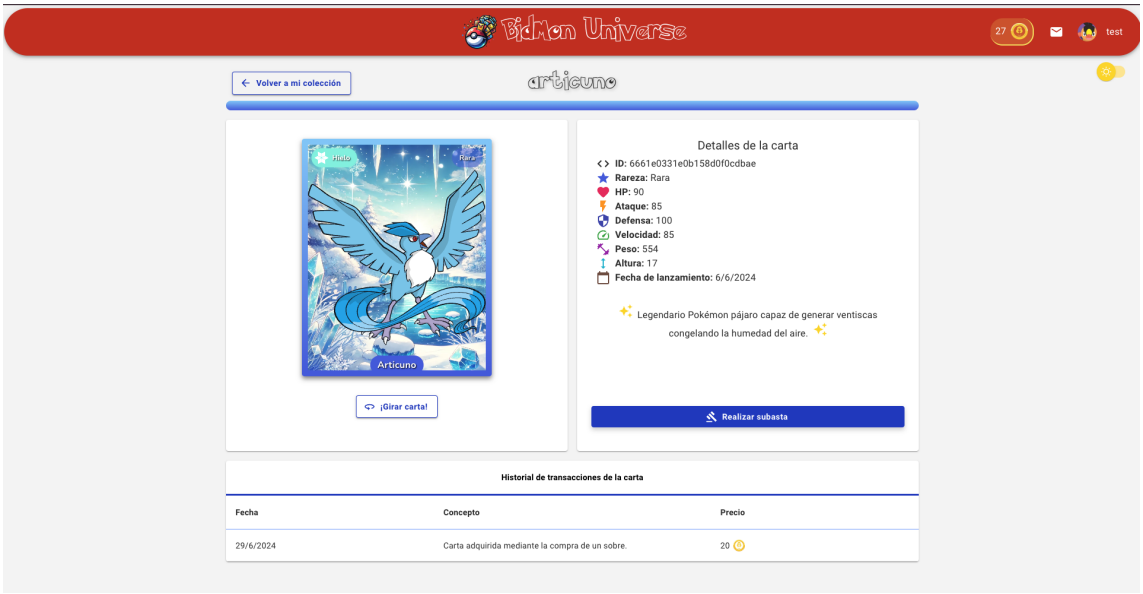


Figura 7.29: Página de detalle de una carta de la colección del usuario.

El usuario debe introducir el precio de salida y la duración de la subasta. Estos son opcionales, si no se especifican, se utilizarán valores por defecto.

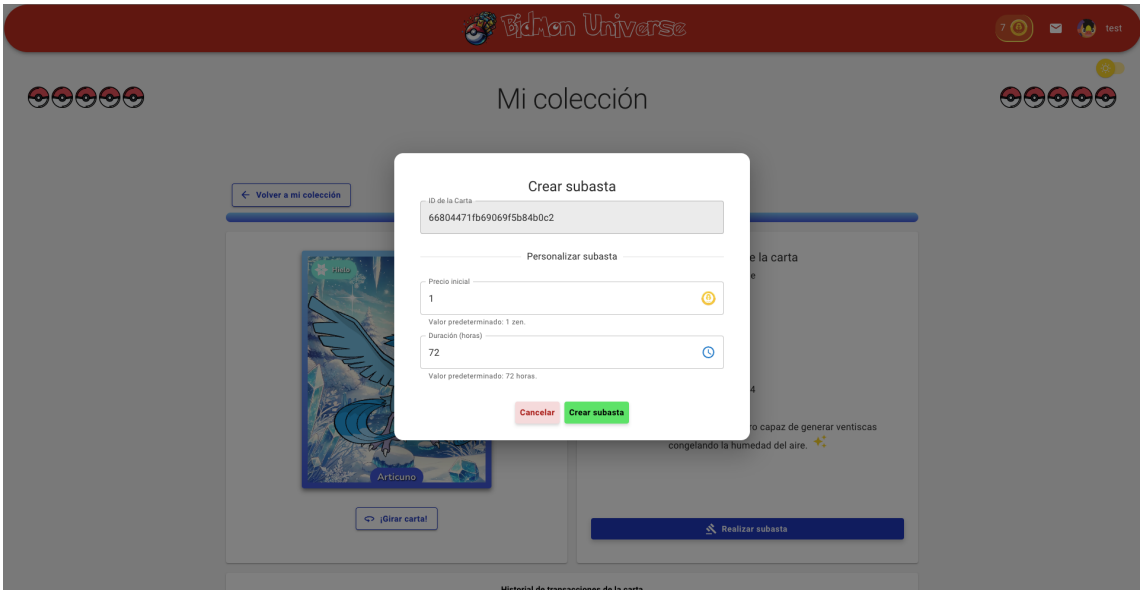


Figura 7.30: Modal de creación de subasta.

Una vez que el usuario ha rellenado los campos, puede pulsar el botón de *Crear subasta* para confirmar la subasta. Se le abrirá un modal de confirmación, en el que se le mostrará un resumen de la subasta y podrá confirmarla o cancelarla.

#### 7.2.2.7. Consultar subastas

En la página de subastas se pueden ver todas las subastas activas de cartas.

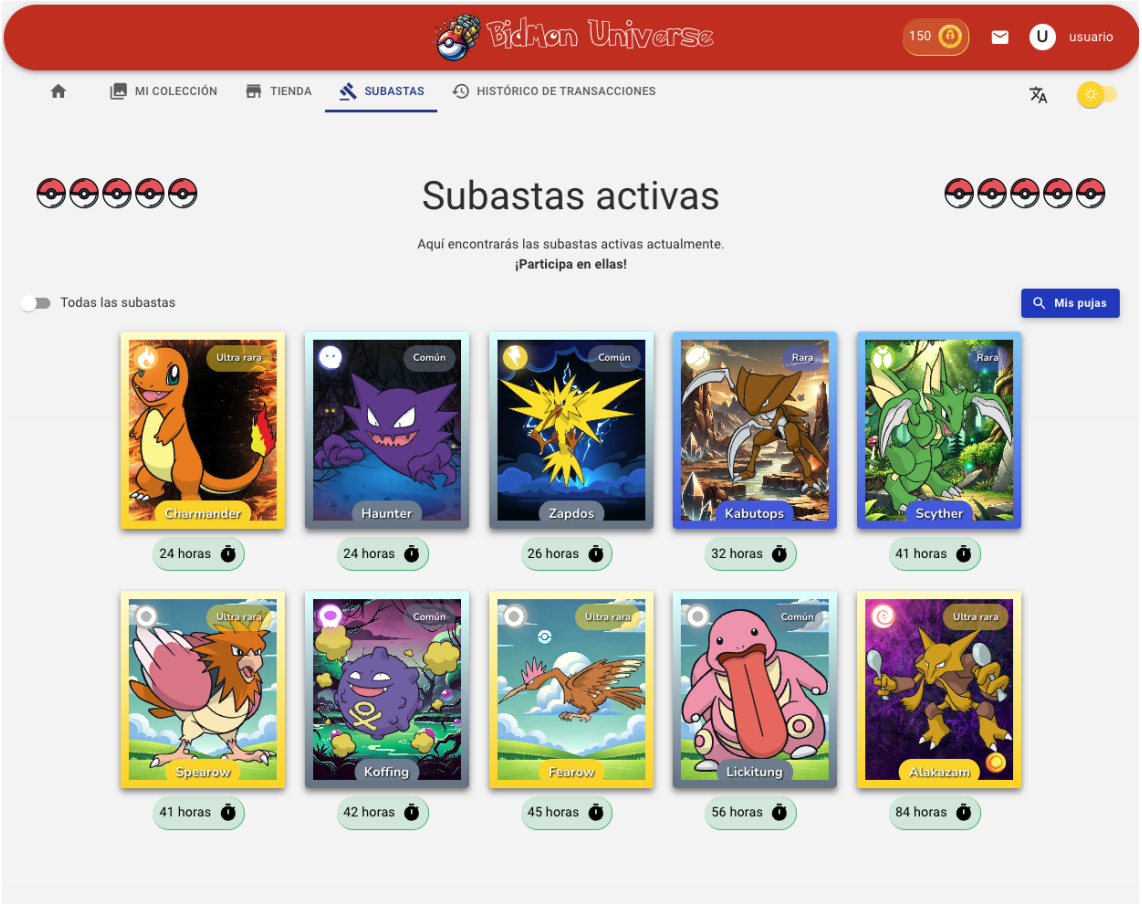


Figura 7.31: Página de las subastas activas de cartas.

Si se hace clic en una subasta, se muestra un detalle de la subasta con información de la carta subastada y la duración restante.

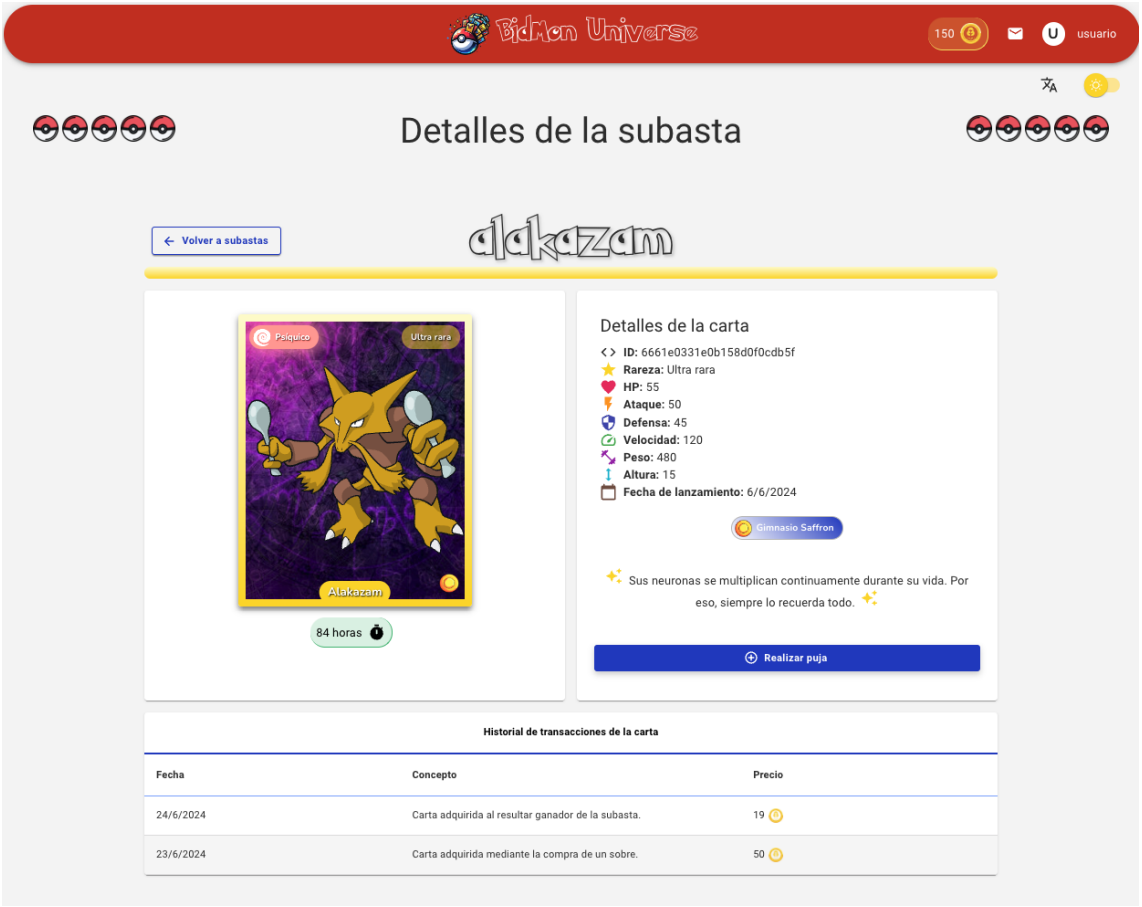


Figura 7.32: Página de detalle de una subasta activa.

El usuario tiene la opción de pujar por la carta haciendo clic en el botón de *Realizar puja* y especificando el precio de la puja.

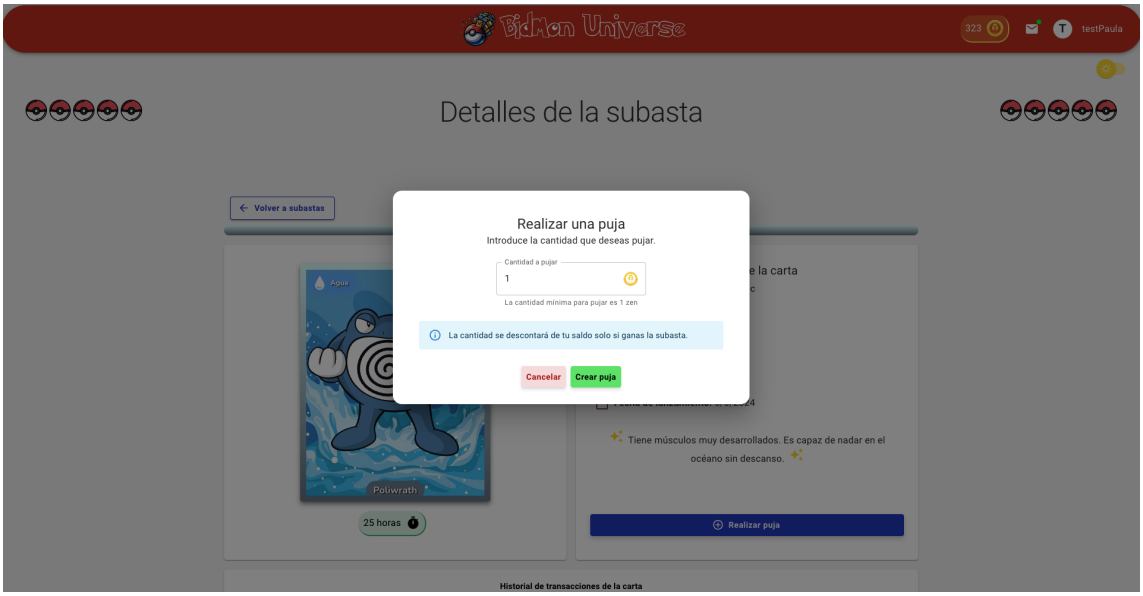


Figura 7.33: Modal de creación de puja.

Si el usuario ha creado una subasta, puede verla en la página de mis subastas activas.

Podría acceder al detalle de igual manera que en la página de subastas activas y podría retirar la subasta si lo desea.

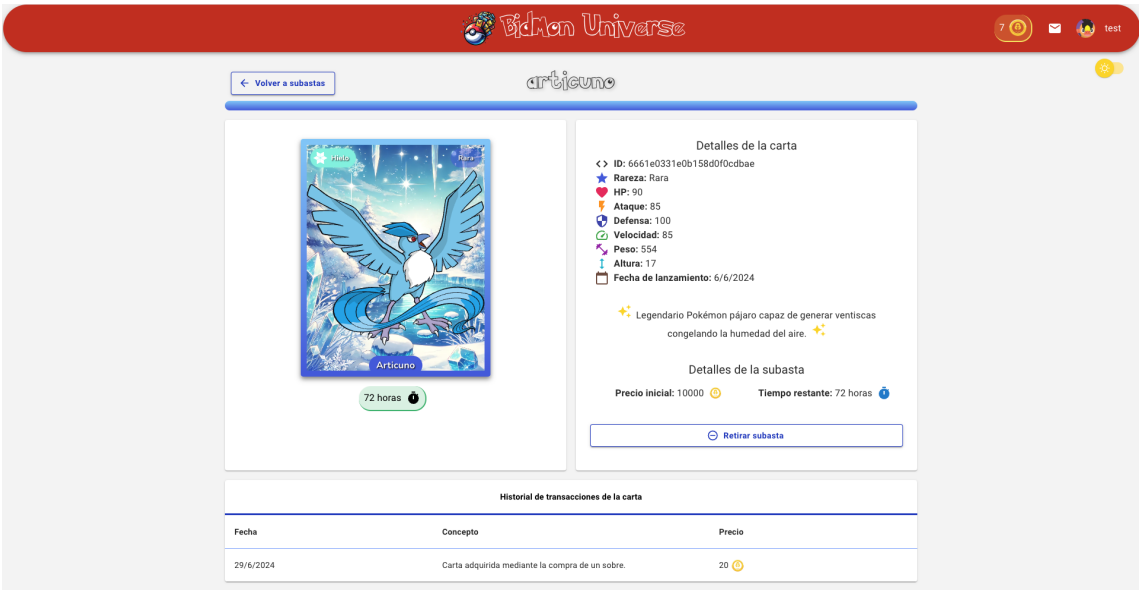


Figura 7.34: Página de detalle de una subasta activa creada por el usuario.

#### 7.2.2.8. Consultar pujas

En la página de subastas hay un botón *Mis pujas* que lleva a una página donde se pueden ver todas las pujas activas del usuario.

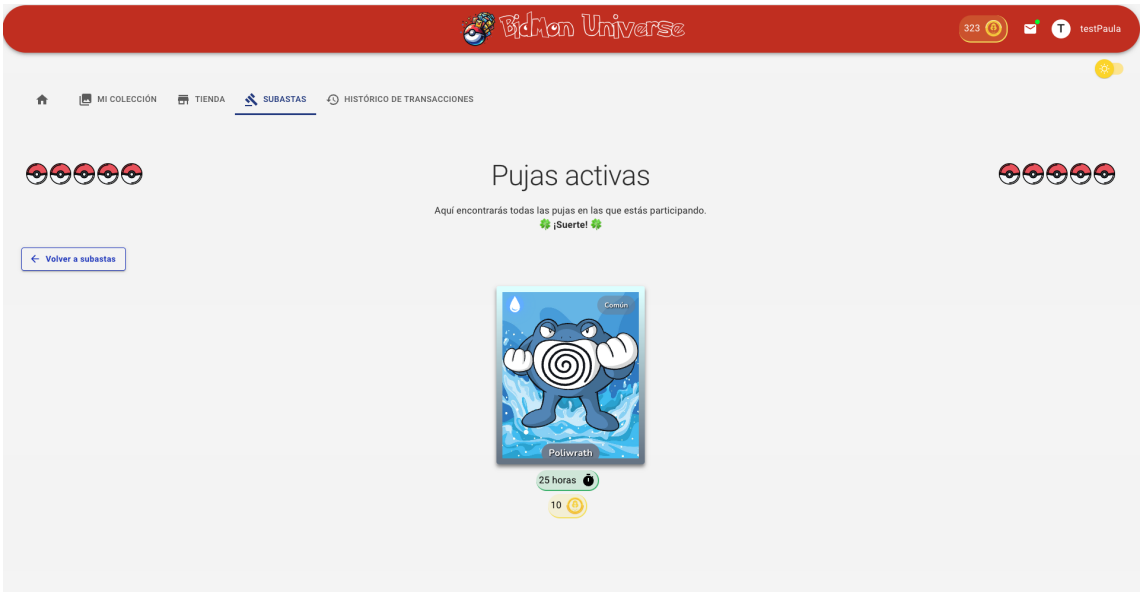


Figura 7.35: Página de las pujas activas del usuario.

Si se hace clic en una puja, se muestra un detalle de la puja con información de la subasta y la carta pujada. El usuario tiene la opción de retirar la puja si lo desea.

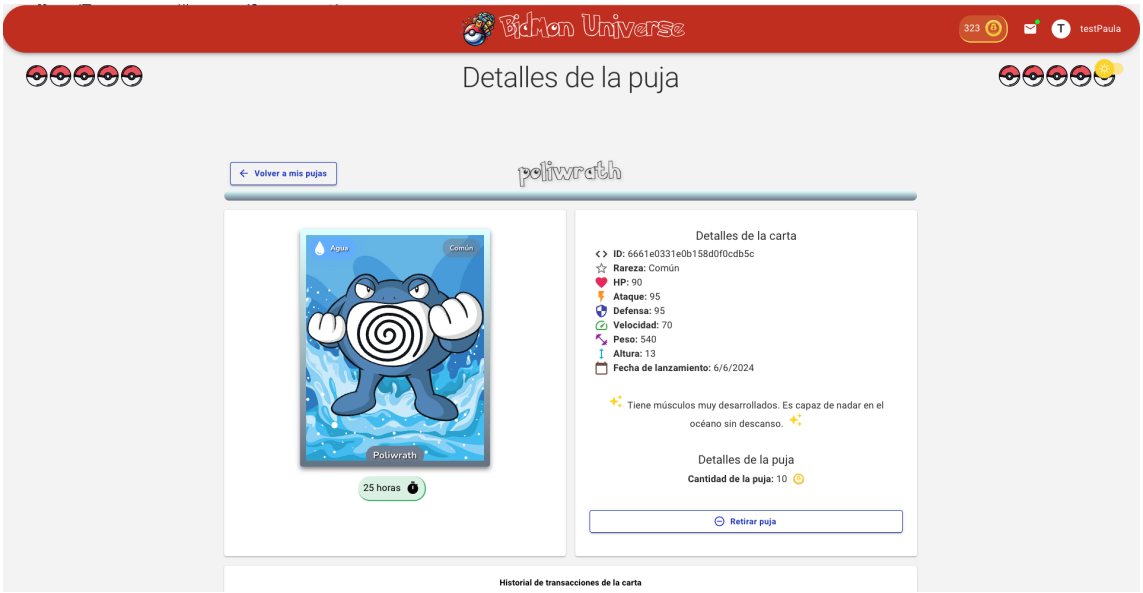


Figura 7.36: Página de detalle de una puja activa.

#### 7.2.2.9. Historial de transacciones

En la página de historial de transacciones se pueden ver todas las transacciones realizadas por el usuario. Estas incluyen la compra de sobres y todos los movimientos de cartas realizados en subastas, es decir, las pujas y subastas realizadas por el usuario.

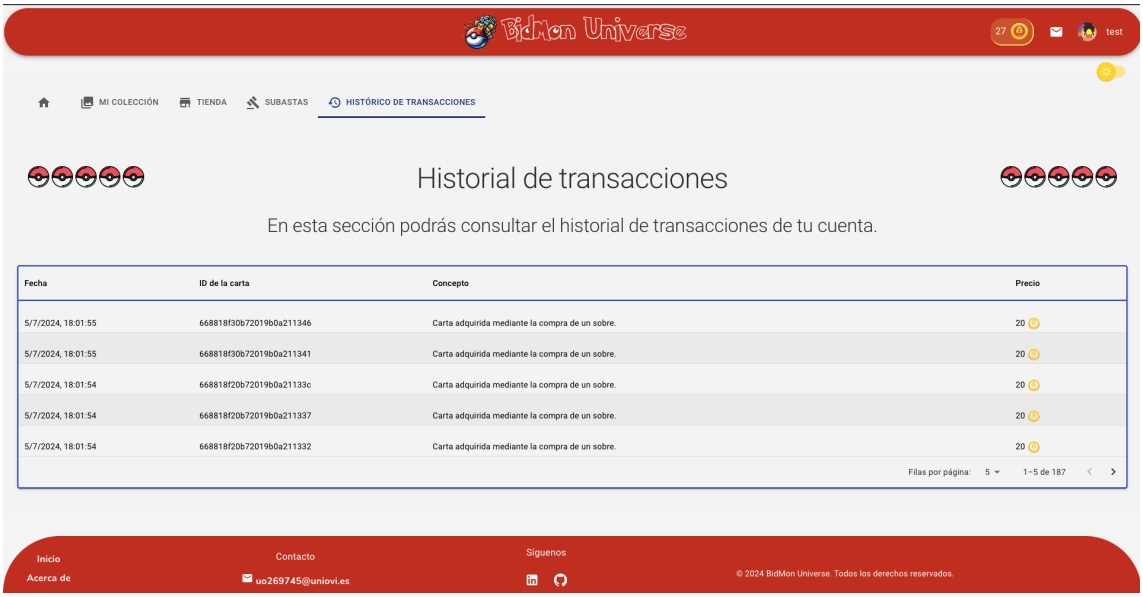


Figura 7.37: Página de transacciones realizadas por el usuario.

### 7.2.2.10. Recarga de saldo

En la página de recarga de saldo se puede recargar el saldo de la aplicación con dinero real. A esta página se accede desde el botón que muestra el saldo en la esquina superior derecha de la aplicación. Para ello se utiliza la pasarela de pago de PayPal, que permite realizar pagos de forma segura y sencilla. La equivalencia entre Zens y dinero real es de 1 Zens = 0,01 €. El mínimo de recarga es de 10 Zens y todas las recargas deben de ser múltiplos de 10.

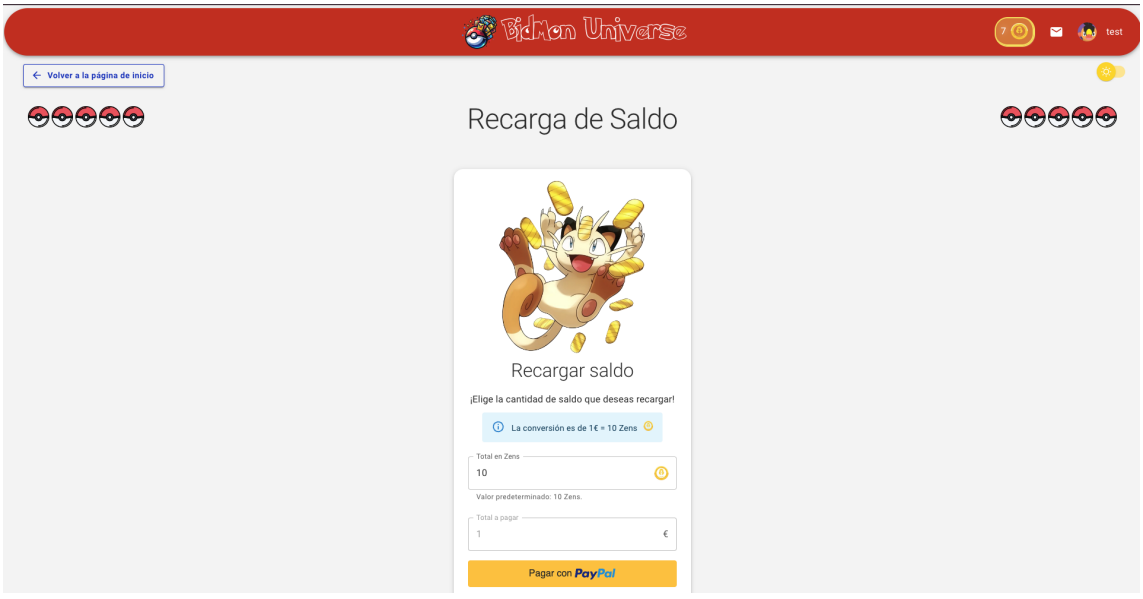


Figura 7.38: Página de recarga de saldo de la aplicación.

#### 7.2.2.11. Consultar notificaciones

En la esquina superior derecha de la aplicación hay un icono de sobre a través del cual se pueden consultar las notificaciones. Si el usuario tiene notificaciones pendientes, se mostrará un indicador en el icono.


Figura 7.39: Indicador de notificaciones pendientes.

Las notificaciones pueden ser de diferentes tipos, si están catalogadas como importantes, se mostrará un icono de una campana en color rojo. Si son notificaciones normales, no se mostrará ningún icono.

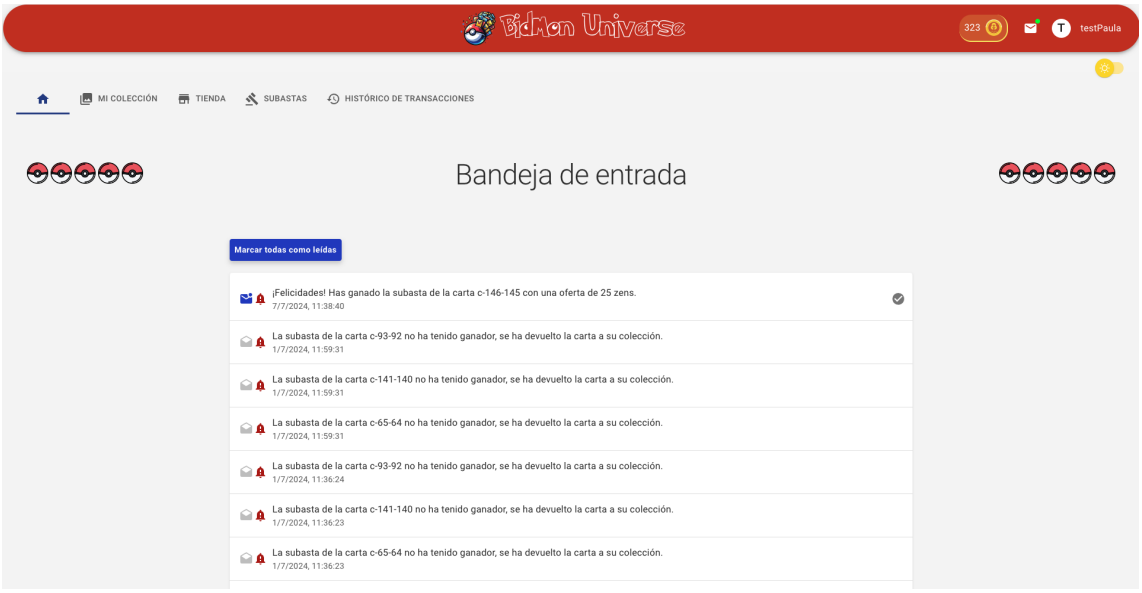


Figura 7.40: Página de notificaciones del usuario.

#### 7.2.2.12. Perfil de usuario

En la página de perfil de usuario se pueden ver los datos del usuario, como el nombre de usuario y la imagen de perfil. El usuario puede cambiar su imagen de perfil y su contraseña desde esta página. A esta página se accede haciendo clic en el avatar del usuario en la esquina superior derecha de la aplicación.

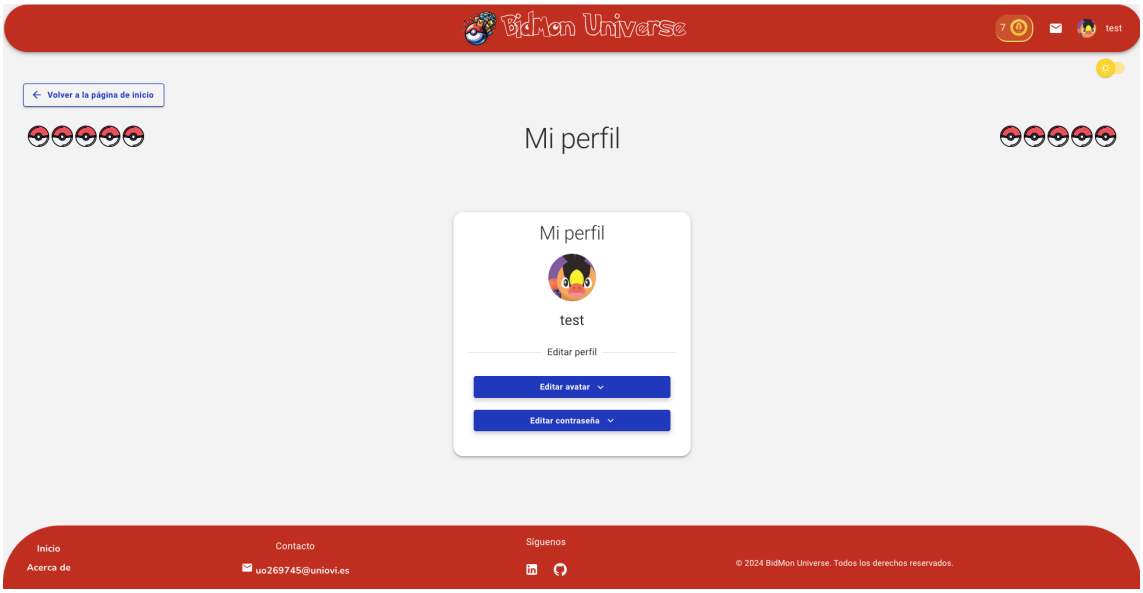


Figura 7.41: Página de perfil del usuario.

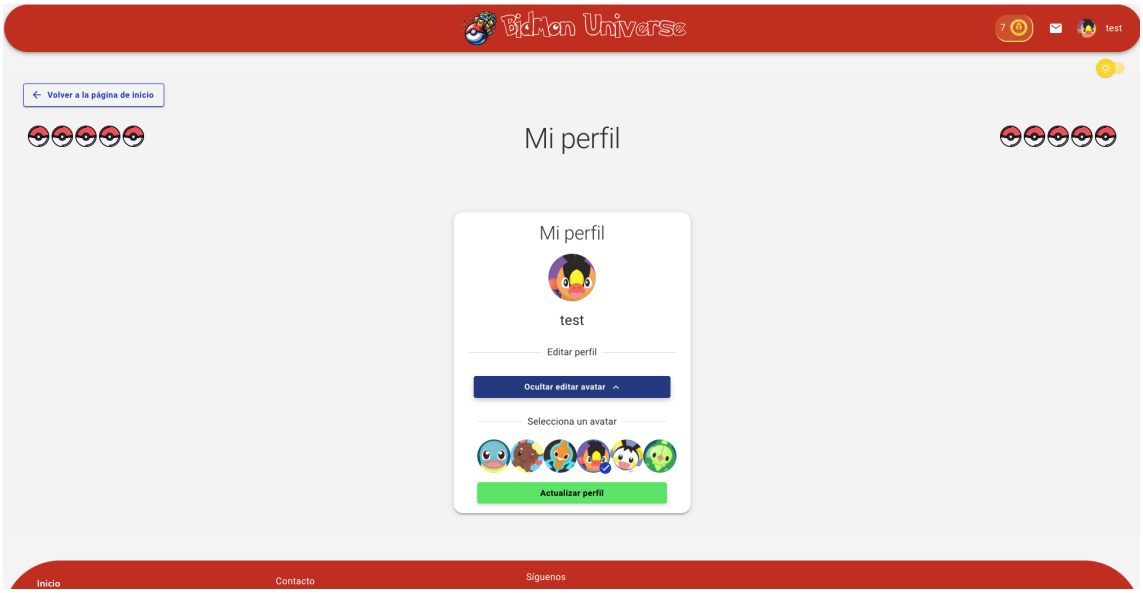


Figura 7.42: Página de perfil del usuario, con la opción de cambiar la imagen de perfil desplegada.

Es importante que el usuario recuerde su contraseña, ya que no se puede recuperar en caso de olvido. Además, debe asegurarse de confirmar los cambios antes de salir de la página de perfil, ya que los cambios no se guardarán si no se confirman.

#### 7.2.2.13. Cierre de sesión

Para cerrar la sesión, el usuario puede hacer clic sobre su avatar en la esquina superior derecha y seleccionar la opción de *Cerrar sesión*.

#### 7.2.2.14. Rol de administrador

El rol de administrador tiene acceso a una serie de funcionalidades distintas a las de un usuario normal. El administrador puede ver una lista de todas las transacciones realizadas en la aplicación, incluyendo las compras de sobres, las subastas y las pujas. Además, tiene la posibilidad de cerrar las subastas que han finalizado. Este proceso consiste en que el sistema procesa todas las subastas cuya duración ha expirado y asigna la carta subastada al usuario que ha realizado la puja más alta. Si no hay pujas, la carta se devuelve al usuario que la subastó. A continuación, se muestra la vista de subastas activas para el administrador.

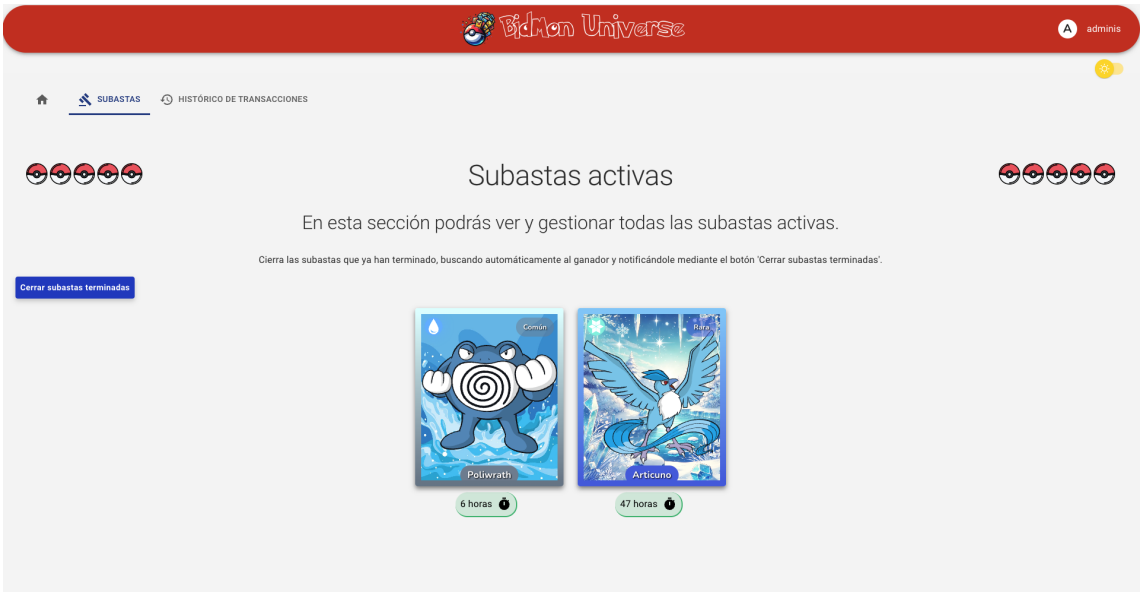


Figura 7.43: Página de subastas activas para el administrador.

Si pulsa en el botón de *Cerrar subastas terminadas*, se abrirá un modal de confirmación en el que se le informa que la acción no se puede deshacer y se le pide que confirme la acción. Una vez confirmada, el sistema procesará las subastas finalizadas y asignará las cartas a los usuarios correspondientes. Mostrará un diálogo con el resultado de la operación, indicando el precio final de la carta y el usuario al que se le ha asignado o si la carta ha sido devuelta al usuario que la subastó.

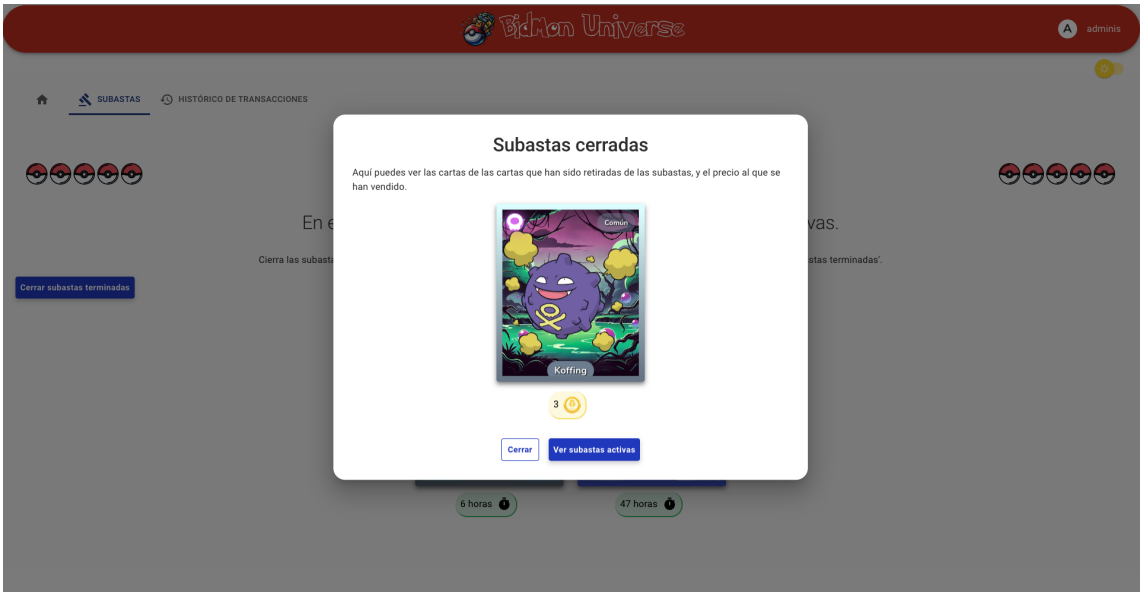


Figura 7.44: Diálogo informativo de cierre de subastas para el administrador.

El inicio de sesión se realiza de igual manera que para un usuario normal, pero no se puede registrar un nuevo administrador desde la aplicación.

## 7.3. CARGA INICIAL DE DATOS

Si se requiere una carga inicial de datos en la base de datos, se puede realizar mediante un script de importación de datos. Este script puede leer los datos de un archivo CSV y cargarlos en la base de datos del sistema. También se puede utilizar un script de exportación de datos para generar un archivo CSV con datos para inicializar la base de datos.

Estos scripts se ejecutarán localmente dependiendo de los datos que se quieran importar o exportar. Los archivos generados deben de estar en el directorio `data` de la carpeta `restapi`. Es necesario tener instalado Node.js y npm en el sistema para ejecutar los scripts y comprobar que la base de datos está en funcionamiento. Además, si se realiza una importanción con datos recopilados manualmente, es necesario que los datos estén en el formato correcto. Para ejecutar los scripts, se debe situar en la carpeta `restapi` y ejecutar los siguientes comandos:

- Exportar datos de cartas:
  1. `npm run fetch-data`
  2. Este comando generará un archivo `cards_data.csv` en el directorio `data`. Recopila datos de PokéAPI PokéAPI, aplica la lógica de negocio y los exporta a un archivo CSV.
- Importar datos de cartas:
  1. `npm run load-data`
  2. Este comando leerá el archivo `cards_data.csv` y cargará los datos en la base de datos.
- Exportar datos de mazos:
  1. `npm run fetchDecksData`
  2. Este comando generará un archivo `decks_data.csv` en el directorio `data`.
- Importar datos de mazos:
  1. `npm run loadDecksData`
  2. Este comando leerá el archivo `decks_data.csv` y cargará los datos en la base de datos.
- Importar datos de sobres de cartas:
  1. `npm run loadCardPacksData`
  2. Este comando leerá el archivo `cardPacks_data.csv` y cargará los datos en la base de datos.

# Capítulo 8

# CONCLUSIONES Y AMPLIACIONES

## 8.1. AMPLIACIONES

El proyecto se puede ampliar de diversas formas para mejorar la experiencia del usuario y añadir nuevas funcionalidades. A continuación, se detallan algunas posibles ampliaciones:

#### 8.1.0.1. Jugabilidad

Implementar un sistema de torneos en el que los usuarios compitan entre ellos utilizando las cartas de su colección. Estas cartas, que ya tienen un nivel de ataque y defensa, podrían mejorar con el uso, incrementando su valor. Los torneos ofrecerían premios a los ganadores, fomentando así la participación y el compromiso de los usuarios.

#### 8.1.0.2. Mejoras de rendimiento

Implementar la paginación en la obtención de resultados de la base de datos para optimizar el rendimiento de la aplicación. Esta mejora permitiría una gestión más eficiente de los datos y una experiencia de usuario más fluida.

#### 8.1.0.3. Notificaciones

Sería interesante mejorar el sistema de notificaciones permitiendo a los usuarios recibir alertas en tiempo real sobre eventos importantes, como el cierre de una subasta sin la necesidad de estar conectado a la aplicación. Para ello, se podría implementar un sistema que combine la implementación actual con un sistema de notificaciones por correo electrónico o SMS.

#### 8.1.0.4. Internacionalización

Desarrollar un sistema de internacionalización para que la aplicación esté disponible en varios idiomas.

Dado que la aplicación ya dispone de un componente de internacionalización adaptable a diferentes dispositivos y tamaños de pantalla, esta funcionalidad sería relativamente sencilla de implementar. Consistiría en añadir los textos en diferentes idiomas y hacer visible dicho componente en la interfaz de usuario.

#### 8.1.0.5. Administración del sistema

Se podría añadir nuevas funcionalidades al panel de administración para gestionar el sistema de forma más eficiente.

La página de bienvenida mostraría un resumen de las estadísticas de la plataforma, como el número de usuarios, cartas en circulación, subastas activas, etc.

Esto incluiría la capacidad de añadir nuevos sobres a la tienda, gestionar usuarios, descatalogar cartas y añadir nuevas cartas a la base de datos desde la interfaz de administración, entre otras funcionalidades.

Estas funcionalidades son sencillas de implementar, ya que la arquitectura de la aplicación permite la adición de nuevas características con facilidad. Por ejemplo, descatalogar una carta solo requeriría modificar un campo en la base de datos.

#### 8.1.0.6. Cierre de subastas automático

Implementar un *cron job* para cerrar las subastas automáticamente cuando llegue la fecha de cierre, o un *trigger* en la base de datos para cerrarlas al alcanzar el tiempo límite. Esto automatizaría el proceso de cierre de subastas, eliminando la necesidad de intervención manual por parte de los administradores.

El método de cierre de subastas ya está implementado. Este método busca las subastas que han alcanzado la fecha de cierre, procesa las pujas vinculadas a ellas, se selecciona la puja más alta válida y se realiza la transferencia de la carta y de saldo, registrando las transacciones correspondientes y notificando a los usuarios implicados.

El motivo de no implementar el cierre automático de subastas pese a tener la funcionalidad es la limitación presupuestaria, ya que un proceso automatizado requiere de un servidor que esté constantemente activo lo que implica un coste adicional.

#### 8.1.0.7. Estadísticas de mercado

Actualmente la aplicación cuenta con un registro de todas las transacciones realizadas por cada carta, por lo que un usuario manualmente podría analizar estos datos para obtener estadísticas del mercado de una carta.

Sería interesante implementar un sistema de estadísticas que permita a los usuarios visualizar la evolución del precio de una carta a lo largo del tiempo, así como comparar el precio de una carta en diferentes momentos. Esto proporcionaría a los usuarios información valiosa sobre el valor de sus cartas y les ayudaría a tomar decisiones informadas sobre su colección y sus inversiones.

#### 8.1.0.8. Retirada de saldo

Actualmente, los usuarios no pueden retirar el saldo de su cuenta. Implementar un sistema de retirada de saldo permitiría a los usuarios transferir el saldo acumulado en su cuenta a su cuenta bancaria o a una cartera de criptomonedas. Lo que haría más atractiva la inversión en la plataforma, ya que los usuarios podrían recuperar su inversión en cualquier momento.

## 8.2. CONCLUSIONES

Este proyecto ha sido una experiencia muy enriquecedora, ya que me ha permitido aplicar los numerosos conocimientos adquiridos durante mi carrera en un entorno que ha supuesto un reto tanto a nivel técnico como organizativo.

En el aspecto técnico, he profundizado en el desarrollo de aplicaciones web con React, lo que me ha permitido adquirir nuevos conocimientos sobre esta tecnología y mejorar mis habilidades en el desarrollo de interfaces de usuario. Además, he trabajado por primera vez con Sockets para la comunicación en tiempo real entre el cliente y el servidor, lo cual ha representado un desafío adicional, pero me ha permitido entender mejor el funcionamiento de las comunicaciones en tiempo real.

También he profundizado en el uso de MongoDB y Mongoose, mejorando así mis habilidades en el diseño de bases de datos NoSQL y en la interacción con las mismas. He tenido que rehacer varias veces el diseño de la base de datos para adaptarlo a las necesidades del proyecto, lo que me ha permitido aprender de mis errores y mejorar en el proceso de diseño.

Asimismo, aunque ya había desplegado otra aplicación en Azure, este proyecto me ha permitido profundizar en el uso de esta plataforma y sentirme más cómoda configurando los servicios necesarios para el despliegue de la aplicación.

A nivel organizativo, llevar a cabo un proyecto de estas características de forma individual ha sido un reto significativo. Al principio, tuve varios problemas debido a una mala planificación y a querer abarcar demasiado, pero con el tiempo aprendí a priorizar y a centrarme en las tareas más importantes. Me enfoqué en desarrollar un modelo base y luego añadir funcionalidades, lo que me permitió avanzar de forma más eficiente y cumplir con los plazos establecidos.

Una de las primeras decisiones que tomé fue realizar la documentación en LaTeX y, aunque inicialmente fue un desafío, con el tiempo adquirí soltura y logré completar la documentación con éxito. Al final, esta elección resultó ser acertada, ya que pude llevar un control de versiones de la documentación con Git, lo que facilitó la gestión de los cambios.

Es difícil destacar una única parte del proyecto, ya que todas las fases han sido importantes y han aportado un valor añadido. No obstante, si tuviera que destacar una, sería el diseño de la interfaz de usuario, ya que ha sido una de las partes más creativas y en la que pude plasmar la idea inicial del proyecto.

Debo agradecer a mi tutor por el tema propuesto, ya que me ha permitido explorar un campo que incentivó mi lado creativo, haciendo que el proceso de desarrollo haya sido más ameno y satisfactorio. Además, su apoyo y orientación han sido fundamentales para la realización del proyecto.

# ANEXOS

# PLAN DE GESTIÓN DE RIESGOS

El objetivo de la plan de gestión de riesgos es identificar, analizar y planificar la respuesta a los distintos riesgos a los que está expuesto el proyecto, de esta manera se busca minimizar la probabilidad de ocurrencia de los riesgos y reducir el impacto de los mismos si se llegan a materializar.

## Metodología aplicada

Este plan se ha realizado siguiendo principalmente la metodología Boehm [12] que divide la gestión de riesgos en dos fases: valoración de riesgos y control de riesgos. Cabe destacar que esta metodología ha sido adaptada a las necesidades del proyecto, añadiendo el paso previo de planificación según la metodología PMBOK [9]. De esta manera, finalmente tenemos tres fases:

- **Planificar la gestión de riesgos:** Definir los objetivos, alcance, metodología y responsabilidades.
- **Valoración de riesgos:** Identificar, analizar y priorizar los riesgos.
- **Control de riesgos:** Planificar las respuestas, resolver y monitorear los riesgos.

Se puede ver el esquema de la metodología aplicada en Figura 8.1: Metodología de gestión de riesgos aplicada.

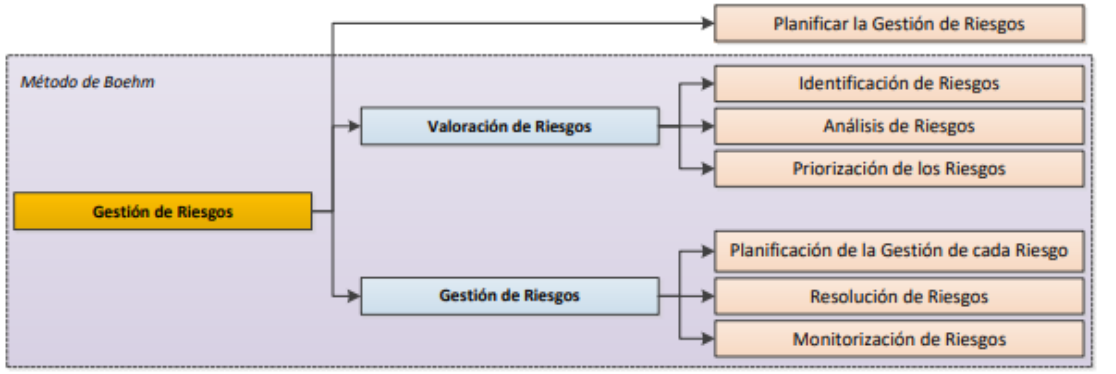


Figura 8.1: Metodología de gestión de riesgos aplicada

## Planificación de la gestión de riesgos

### Categorías de riesgos

Basándose en la estructura de desglose de riesgo representadas en el PMBOK, se han identificado cuatro categorías principales de riesgos para el proyecto:

- **Riesgos técnicos:** Relacionados con la tecnología y el desarrollo del software.
- **Riesgos externos:** Relacionados con riesgos que no se pueden controlar directamente, por ejemplo, cambios en la legislación.

- **Riesgos organizacionales:** Relacionados con el entorno empresarial y económico.
- **Riesgos de gestión del proyecto:** Relacionados con la planificación, organización y control del proyecto.

En la Figura 8.2: RBS, desglose de categorías de riesgos del proyecto se puede ver el desglose dettallado de categorías de riesgos del proyecto.

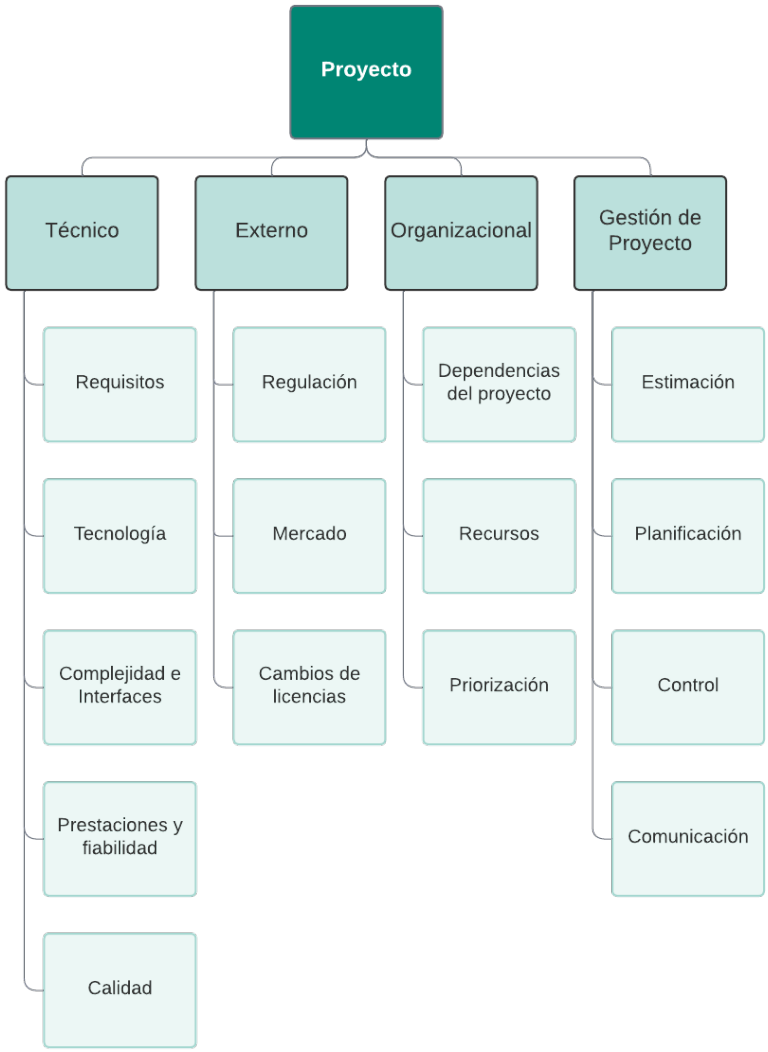


Figura 8.2: RBS, desglose de categorías de riesgos del proyecto

### Probabilidad e impacto

Para la valoración de riesgos se ha utilizado la Figura 8.3: Matriz de probabilidad vs impacto definida anteriormente. Esta matriz establece los valores utilizados en la priorización de riesgos. Cada riesgo tiene un valor asociado de acuerdo a su probabilidad de ocurrencia y el impacto en el proyecto.

| Probabilidad | | | | | | | |
|---|---|---|---|---|---|---|---|
| | Muy Alta | *0,90* | 0,05 | 0,14 | 0,27 | 0,50 | 0,81 |
| | Alta | *0,70* | 0,04 | 0,11 | 0,21 | 0,39 | 0,63 |
| | Media | *0,50* | 0,03 | 0,08 | 0,15 | 0,28 | 0,45 |
| | Baja | *0,30* | 0,02 | 0,05 | 0,09 | 0,17 | 0,27 |
| | Muy Baja | *0,10* | 0,01 | 0,02 | 0,03 | 0,06 | 0,09 |
| | | | *0,05* | *0,15* | *0,30* | *0,55* | *0,90* |
| | | | Inapreciable | Bajo | Medio | Alto | Crítico |
| | | | *0,05* | *0,15* | *0,30* | *0,55* | *0,90* |
| | | | Impacto | | | | |

Figura 8.3: Matriz de probabilidad vs impacto

### Identificación de riesgos

Para la identificación de riesgos se han utilizado dos técnicas principales: la recopilación de información mediante *brainstorming* y, de forma complementaria, el análisis de listas de control. La combinación de estas técnicas ha permitido una identificación de riesgos realista, abarcando tanto los riesgos inherentes al tipo de proyecto, es decir, desarrollo de software, como los riesgos específicos del proyecto en particular.

### Análisis de riesgos

Una vez identificados los riesgos, se ha procedido a analizarlos. Se ha llevado a cabo un análisis cualitativo de los riesgos que consta de los siguientes procesos:

- **Evaluación de la probabilidad e impacto:** Asignación de valores de probabilidad e impacto a cada riesgo, obteniendo un valor de prioridad basado en la Figura 8.3: Matriz de probabilidad vs impacto.
- **Categorización de riesgos:** Clasificación de los riesgos identificados según las categorías de riesgos establecidas en Figura 8.2: RBS, desglose de categorías de riesgos del proyecto.
- **Evaluación de la urgencia:** Priorización preliminar de los riesgos basada en la urgencia con la que deben ser tratados.

### Priorización

Una vez analizados los riesgos, se procede a su priorización. A cada riesgo se le asigna un valor de prioridad basado en los resultados del análisis de probabilidad e impacto, así como en la evaluación de la urgencia.

De esta manera, se obtiene una lista de riesgos ordenada de mayor a menor importancia, lo que permite centrar los esfuerzos en los riesgos más críticos.

### Planificación de la gestión de cada riesgo

Se lleva a cabo un análisis más detallado de los riesgos definiendo las estrategias de respuesta a los mismos. En el proyecto se ha optado por las siguientes estrategias de respuesta:

- **Evitar el riesgo:** Se trata de eliminar la amenaza o la oportunidad que representa el riesgo.

- **Transferir el riesgo:** Se trata de trasladar la responsabilidad del riesgo a un tercero.
- **Mitigar el riesgo:** Se trata de reducir la probabilidad de ocurrencia o en caso de que ocurra, reducir el impacto.
- **Aceptar el riesgo:** Se trata de asumir las consecuencias del riesgo y convivir con él.

#### Resolución de riesgos

Una vez planificadas las respuestas a los riesgos, se procede a la resolución de los mismos determinando si los riesgos son eliminados o solucionados. Para ello se ha llevado a cabo un desarrollo incremental con el fin de minimizar los riesgos y permitir una mayor flexibilidad en la gestión de los mismos.

#### Monitorización y control de riesgos

Por último, se establece un plan de monitorización y control de riesgos. La principal técnica utilizada para la monitorización y control de riesgos son las reuniones periódicas de seguimiento del proyecto, en las que se controla el progreso del proyecto, se revisan los riesgos y se actualiza la información de los mismos tomando las medidas correctoras necesarias en cada momento.

# REGISTRO DE RIESGOS

En esta sección se presentan los resultados obtenidos tras el análisis de los riesgos identificados en el proyecto (ver la sección 5.1.4.2 Identificación de Riesgos).

Los riesgos están ordenados en función de el valor obtenido al realizar el análisis según la probabilidad e impacto de cada uno de ellos. Para cada riesgo se detalla su descripción, categoría, probabilidad e impacto, así como la respuesta y estrategia a seguir para tratar con él.

Se puede consultrar la descripción del proceso de análisis de riesgos en el Plan de Gestión de Riesgos.

Tabla 8.1: Riesgo 1. Falta de comunicación con el tutor del TFG

| | | |
|---|---|---|
| **Identificador** | 1 | |
| **Nombre** | Falta de comunicación con el tutor del TFG | |
| **Descripción** | En proyectos académicos como un TFG, una comunicación inadecuada con el tutor puede llevar a desviaciones del objetivo del proyecto, retrasos y resultados que no cumplen con las expectativas académicas. | |
| **Categoría** | Riesgo de gestión del proyecto | |
| **Probabilidad** | Media | |
| **Impacto** | Presupuesto | Bajo |
| | Planificación | Alto |
| | Alcance | Crítico |
| | Calidad | Alto |
| | Total | 0.45 |
| **Respuesta** | Establecer un calendario de reuniones periódicas con el tutor para revisar el progreso del proyecto y asegurar que se cumplen los objetivos establecidos. Utilizar herramientas de comunicación como correo electrónico y Microsoft Teams con el fin de mantener una comunicación constante y efectiva. Documentar todas las decisiones y avances en un informe de progreso compartido con el tutor. | |
| **Estrategia** | Mitigar el riesgo | |

Tabla 8.2: Riesgo 1. Seguridad de la información

| | | |
|---|---|---|
| **Identificador** | 2 | |
| **Nombre** | Seguridad de la información | |
| **Descripción** | La seguridad de la información se refiere a la protección de datos contra accesos no autorizados, alteraciones, robos o eliminaciones, tanto accidentalmente como de manera intencional. La falta de seguridad en el sistema puede resultar en la pérdida de datos, daños a la reputación de la empresa y sanciones legales. | |
| **Categoría** | Riesgo técnico | |
| **Probabilidad** | Alta | |
| **Impacto** | Presupuesto | Medio |
| | Planificación | Bajo |
| | Alcance | Bajo |
| | Calidad | Alto |
| | Total | 0.39 |
| **Respuesta** | Durante la fase de puesta en producción, se contratarán servicios de seguridad gestionados a un proveedor externo especializado. Este proveedor será responsable de implementar y mantener las medidas de seguridad necesarias para proteger los datos y asegurar el cumplimiento con las normativas vigentes. | |
| **Estrategia** | Transferir el riesgo | |

Tabla 8.3: Riesgo 3. Intento de fraude por parte de los usuarios finales

| | | |
|---|---|---|
| **Identificador** | 3 | |
| **Nombre** | Intento de fraude por parte de los usuarios finales | |
| **Descripción** | Los intentos de fraude, como la creación de cuentas falsas o transacciones fraudulentas, pueden tener un impacto financiero y de reputación significativo. | |
| **Categoría** | Riesgo técnico | |
| **Probabilidad** | Alta | |
| **Impacto** | Presupuesto | Medio |
| | Planificación | Bajo |
| | Alcance | Bajo |
| | Calidad | Alto |
| | Total | 0.39 |
| **Respuesta** | Implementar sistemas de verificación de la identidad y medidas para reducir la posibilidad de que un usuario realice transacciones fraudulentas. | |
| **Estrategia** | Mitigar el riesgo | |

Tabla 8.4: Riesgo 4. Errores en las estimaciones de tareas

| | | |
|---|---|---|
| **Identificador** | 4 | |
| **Nombre** | Errores en las estimaciones de tareas | |
| **Descripción** | Las estimaciones incorrectas de la duración de las tareas pueden resultar en retrasos en el proyecto y en la necesidad de reajustar el cronograma. | |
| **Categoría** | Riesgo de gestión del proyecto | |
| **Probabilidad** | Media | |
| **Impacto** | Presupuesto | Medio |
| | Planificación | Alto |
| | Alcance | Alto |
| | Calidad | Medio |
| | Total | 0.28 |
| **Respuesta** | Consultar con el tutor las estimaciones realizadas y realizar varias iteraciones para afinarlas. Realizar revisiones periódicas del cronograma del proyecto para identificar desviaciones tempranas y ajustar las estimaciones según sea necesario. | |
| **Estrategia** | Mitigar el riesgo | |

Tabla 8.5: Riesgo 5. Conciliación entre responsabilidades académicas y laborales

| | | |
|---|---|---|
| **Identificador** | 5 | |
| **Nombre** | Conciliación entre responsabilidades académicas y laborales | |
| **Descripción** | La conciliación entre las responsabilidades académicas y laborales puede reducir significativamente la disponibilidad de tiempo para dedicar al proyecto. Esto, a su vez, puede disminuir la productividad y eventualmente provocar retrasos en la entrega del proyecto. | |
| **Categoría** | Riesgo de gestión del proyecto | |
| **Probabilidad** | Alta | |
| **Impacto** | Presupuesto | Bajo |
| | Planificación | Alto |
| | Alcance | Medio |
| | Calidad | Medio |
| | Total | 0.28 |
| **Respuesta** | Establecer un horario de trabajo fijo para dedicar tiempo al proyecto. Priorizar las tareas del proyecto y planificar con anticipación para evitar retrasos, apoyándose además de en la planificación temporal en tableros de trabajo como Trello. | |
| **Estrategia** | Mitigar el riesgo | |

Tabla 8.6: Riesgo 6. Aceptación del cliente

| | | |
|---|---|---|
| **Identificador** | 6 | |
| **Nombre** | Aceptación del cliente | |
| **Descripción** | Si el producto final no cumple con las expectativas o requisitos del cliente, en este caso el tribunal del TFG y el propio tutor, puede resultar en el rechazo del proyecto. | |
| **Categoría** | Riesgo organizacional | |
| **Probabilidad** | Media | |
| **Impacto** | Presupuesto | Bajo |
| | Planificación | Bajo |
| | Alcance | Bajo |
| | Calidad | Alto |
| | Total | 0.28 |
| **Respuesta** | Para mitigar este riesgo, se implementarán las mejores prácticas en el desarrollo del proyecto y se elaborará una documentación detallada que cubra todos los aspectos del proyecto, desde la planificación y el diseño hasta el cierre del mismo. Además, se mantendrá un seguimiento continuo con el tutor del TFG para verificar que se cumpla con el alcance del proyecto. | |
| **Estrategia** | Mitigar el riesgo | |

Tabla 8.7: Riesgo 7. Probelmas de cumplimiento normativo y legal

| | | |
|---|---|---|
| **Identificador** | 7 | |
| **Nombre** | Problemas de cumplimiento normativo y legal | |
| **Descripción** | La falta de cumplimiento de las normativas y leyes aplicables puede resultar en sanciones legales, multas y daños a la reputación de la empresa. | |
| **Categoría** | Riesgo organizacional | |
| **Probabilidad** | Alta | |
| **Impacto** | Presupuesto | Medio |
| | Planificación | Medio |
| | Alcance | Medio |
| | Calidad | Alto |
| | Total | 0.28 |
| **Respuesta** | Antes de la puesta en producción del proyecto, se llevará a cabo una auditoría exhaustiva con un experto en cumplimiento normativo y legal. Este experto revisará todas las fases del proyecto para asegurar que se cumplan todas las normativas y leyes aplicables y se llevarán a cabo las medidas correctoras necesarias. | |
| **Estrategia** | Transferir el riesgo | |

Tabla 8.8: Riesgo 8. Cambios en la licencia de software

| | | |
|---|---|---|
| **Identificador** | 8 | |
| **Nombre** | Cambios en la licencia de software | |
| **Descripción** | Cambios en la licencia de software pueden afectar la disponibilidad, problemas legales, costos incrementados o la necesidad de buscar alternativas de software. | |
| **Categoría** | Riesgo externo | |
| **Probabilidad** | Baja | |
| **Impacto** | Presupuesto | Crítico |
| | Planificación | Bajo |
| | Alcance | Alto |
| | Calidad | Bajo |
| | Total | 0.27 |
| **Respuesta** | Se asumirá el riesgo, y en caso de que se produzcan cambios en la licencia de software, se realizarán las siguientes acciones:<br>▪ Evaluar las nuevas condiciones de la licencia para entender el impacto.<br>▪ Buscar y analizar alternativas de software que cumplan con los requisitos del proyecto.<br>▪ Asegurar el cumplimiento normativo.<br>▪ Actualizar el presupuesto y el cronograma del proyecto si es necesario.<br>▪ Comunicar los cambios a todas las partes interesadas.<br>▪ Llevar a cabo la integración del nuevo software, si fuese necesario. | |
| **Estrategia** | Asumir el riesgo | |

Tabla 8.9: Riesgo 9. Problemas de rendimiento

| | | |
|---|---|---|
| **Identificador** | 9 | |
| **Nombre** | Problemas de rendimiento | |
| **Descripción** | Un sistema que no puede manejar la carga de trabajo esperada puede resultar en una experiencia de usuario deficiente y, en última instancia, en la pérdida de clientes. | |
| **Categoría** | Riesgo técnico | |
| **Probabilidad** | Alta | |
| **Impacto** | Presupuesto | Medio |
| | Planificación | Bajo |
| | Alcance | Bajo |
| | Calidad | Alto |
| | Total | 0.21 |
| **Respuesta** | Para mitigar este riesgo, se implementarán las mejores prácticas de optimización de rendimiento dentro del presupuesto y recursos disponibles. Esto incluye realizar pruebas de carga y estrés y optimizar el código. | |
| **Estrategia** | Mitigar el riesgo | |

Tabla 8.10: Riesgo 10. Fallos en el hardware o software

| | | |
|---|---|---|
| **Identificador** | 10 | |
| **Nombre** | Fallos en el hardware o software | |
| **Descripción** | Problemas con el hardware o software crítico para el proyecto pueden resultar en pérdida de datos importantes, retrasos en el proyecto y aumento en los costos. | |
| **Categoría** | Riesgo organizacional | |
| **Probabilidad** | Baja | |
| **Impacto** | Presupuesto | Alto |
| | Planificación | Alto |
| | Alcance | Alto |
| | Calidad | Medio |
| | Total | 0.17 |
| **Respuesta** | Se asumirá el riesgo, lo que implica asumir los costos de sustitución de hardware o software si fuesen necesarios. | |
| **Estrategia** | Asumir el riesgo | |

Tabla 8.11: Riesgo 11. Problemas de usabilidad

| | | |
|---|---|---|
| **Identificador** | 11 | |
| **Nombre** | Problemas de usabilidad | |
| **Descripción** | Una interfaz de usuario no intuitiva puede resultar en una curva de aprendizaje empinada, frustración por parte de los usuarios y, en consecuencia, una menor adopción del producto. | |
| **Categoría** | Riesgo técnico | |
| **Probabilidad** | Baja | |
| **Impacto** | Presupuesto | Inapreciable |
| | Planificación | Bajo |
| | Alcance | Bajo |
| | Calidad | Alto |
| | Total | 0.17 |
| **Respuesta** | Se realizará un diseño de la aplicación teniendo en cuenta las buenas prácticas en usabilidad. Además, se utilizarán herramientas para identificar y solucionar problemas de usabilidad durante el desarrollo. | |
| **Estrategia** | Eliminar el riesgo | |

Tabla 8.12: Riesgo 12. Problemas de accesibilidad

| | | |
|---|---|---|
| **Identificador** | 12 | |
| **Nombre** | Problemas de accesibilidad | |
| **Descripción** | Si un sitio web no está diseñado para ser accesible a personas con discapacidades, se podría excluir a un segmento significativo de usuarios potenciales. | |
| **Categoría** | Riesgo técnico | |
| **Probabilidad** | Baja | |
| **Impacto** | Presupuesto | Inapreciable |
| | Planificación | Bajo |
| | Alcance | Bajo |
| | Calidad | Alto |
| | Total | 0.06 |
| **Respuesta** | Se utilizarán herramientas específicas para evaluar y corregir problemas de accesibilidad, garantizando que el sitio web cumpla con las pautas de accesibilidad WCAG. | |
| **Estrategia** | Eliminar el riesgo | |

# PRUEBAS DE ACCESIBILIDAD

En este apartado se detallan los resultados obtenidos en la realización de las pruebas de accesibilidad del sistema. Se han realizado pruebas de accesibilidad en las distintas vistas de la aplicación, comprobando que se cumplen los requisitos de accesibilidad definidos en la sección 6.2.1.2 Requisitos no funcionales.

Se muestran a continuación los resultados obtenidos en las pruebas de accesibilidad realizadas en las vistas principales de la aplicación. Estos resultados han sido obtenidos de la herramienta de evaluación de accesibilidad *WAVE*.

## Home

- **Resultado:** La vista principal de la aplicación cumple con los requisitos de accesibilidad.
- **Observaciones:** Los errores de contraste se deben a los colores de las cartas de ejemplo y son aceptados.

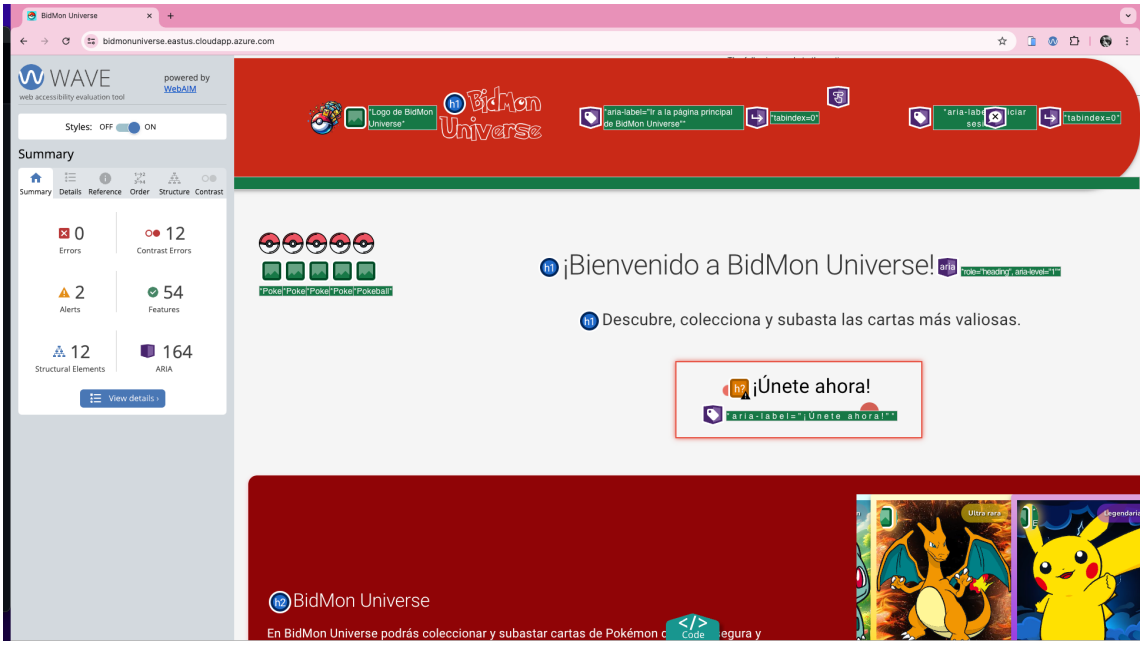


Figura 8.4: Accesibilidad Página Home

## Acerca de

- **Resultado:** La vista de información sobre la aplicación cumple con los requisitos de accesibilidad.

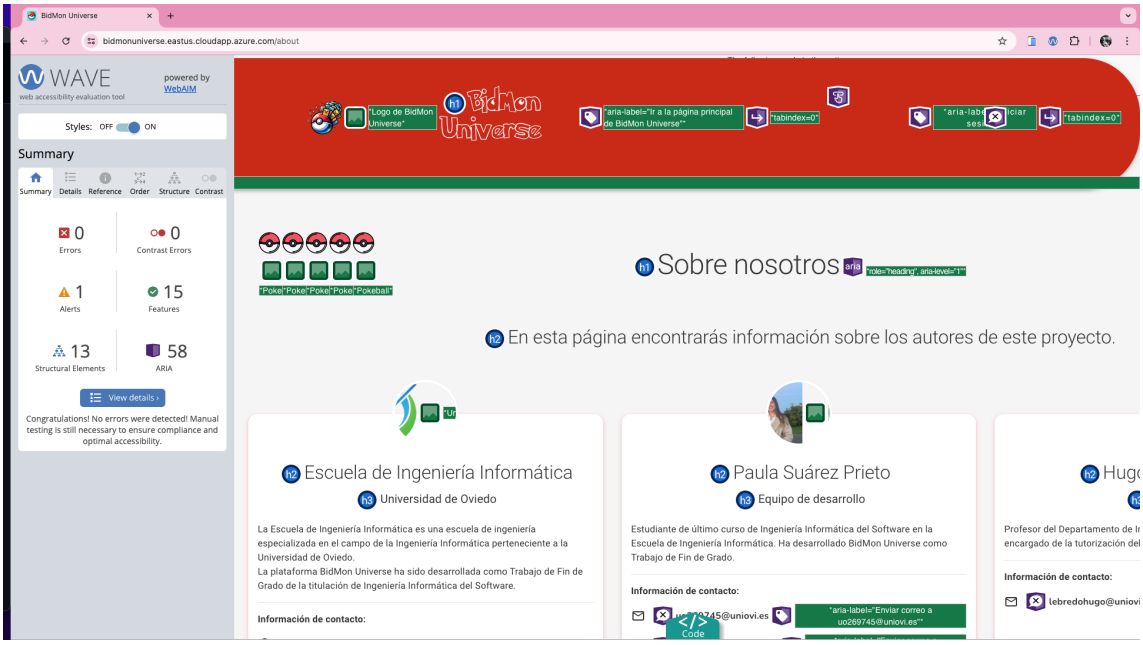


Figura 8.5: Accesibilidad Página Acerca de

## Login

- **Resultado:** La vista de inicio de sesión cumple con los requisitos de accesibilidad.

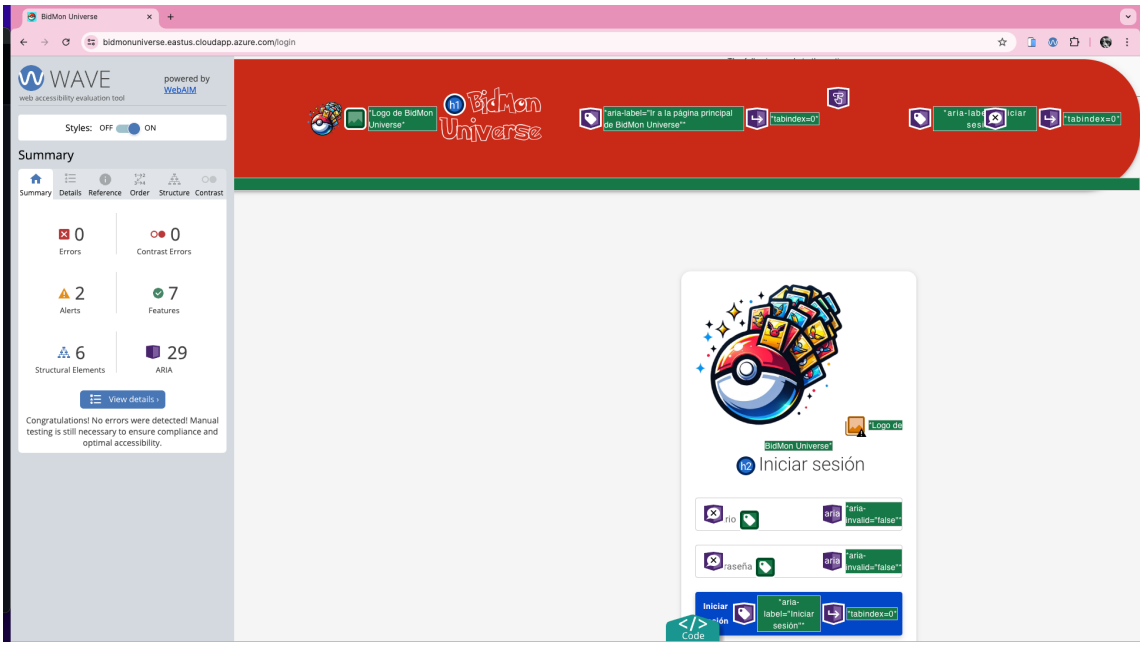


Figura 8.6: Accesibilidad Página Login

## Registro

- **Resultado:** La vista de registro cumple con los requisitos de accesibilidad.

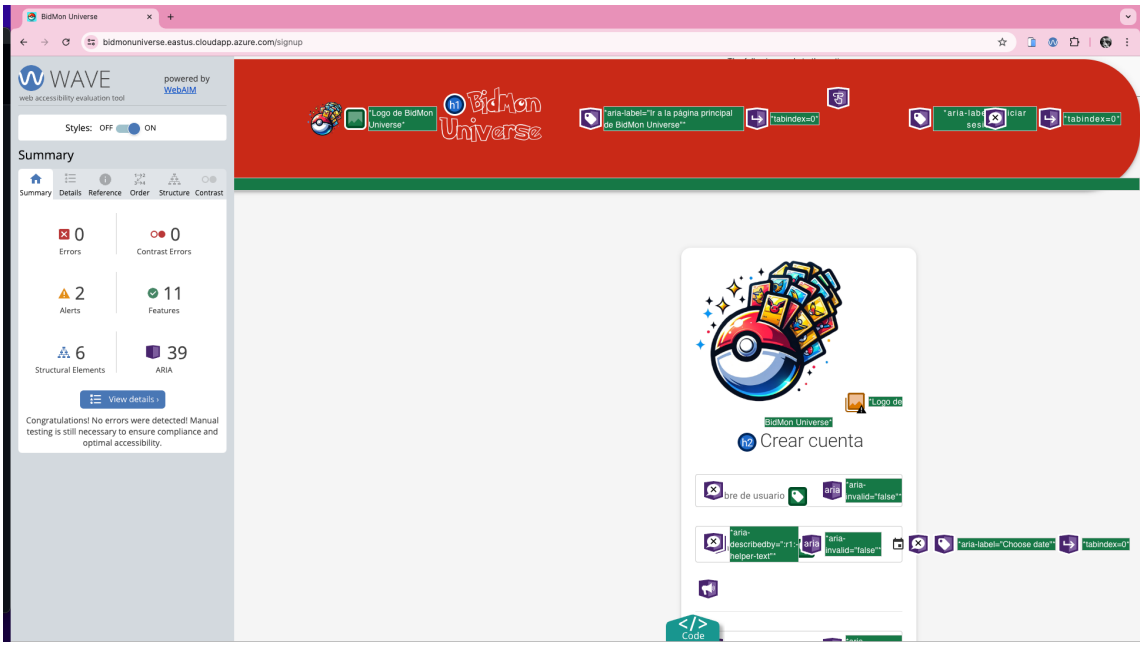


Figura 8.7: Accesibilidad Página Registro

## Perfil

- **Resultado:** La vista de perfil cumple con los requisitos de accesibilidad.

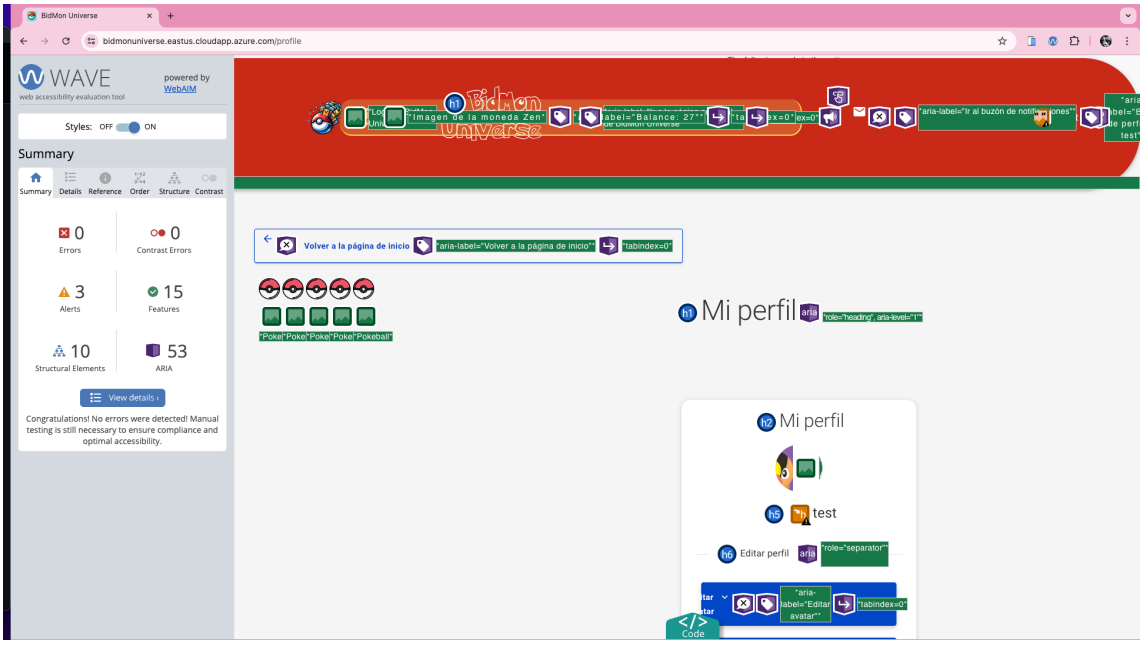


Figura 8.8: Accesibilidad Página Perfil

## Página principal de usuario autenticado

- **Resultado:** La vista principal de usuario autenticado cumple con los requisitos de accesibilidad.
- **Observaciones:** Los errores de contraste se deben a los colores de las cartas y son aceptados.

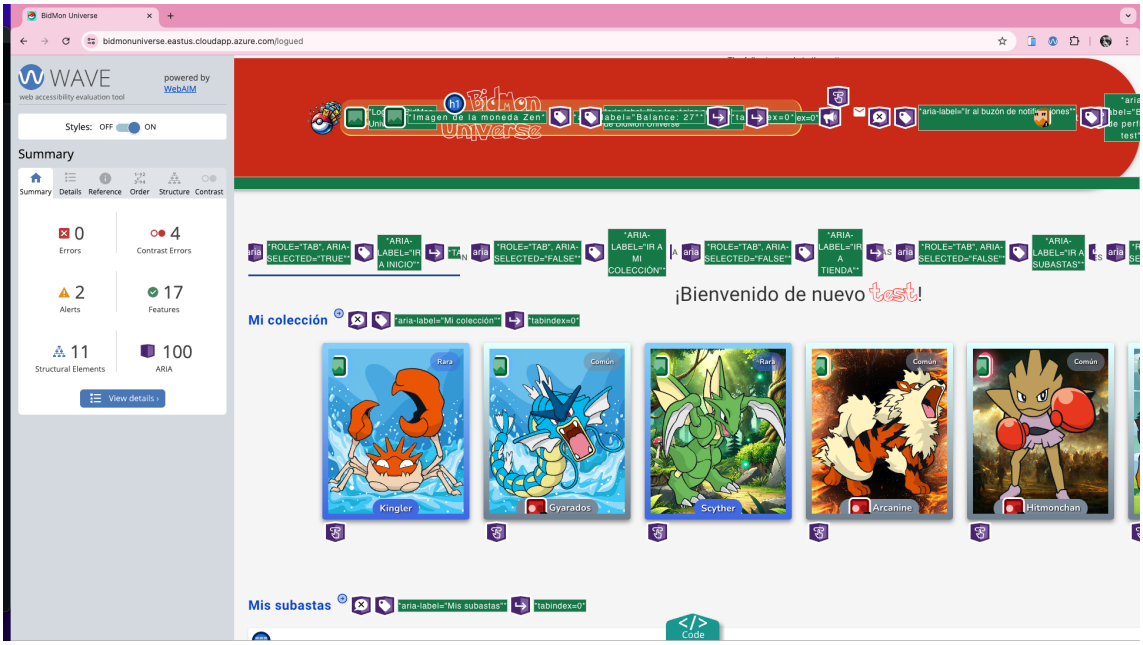


Figura 8.9: Accesibilidad Página principal de usuario autenticado

## Colección de cartas del usuario

- **Resultado:** La vista de colección de cartas del usuario cumple con los requisitos de accesibilidad.
- **Observaciones:** Los errores de contraste se deben a los colores de las cartas y son aceptados.

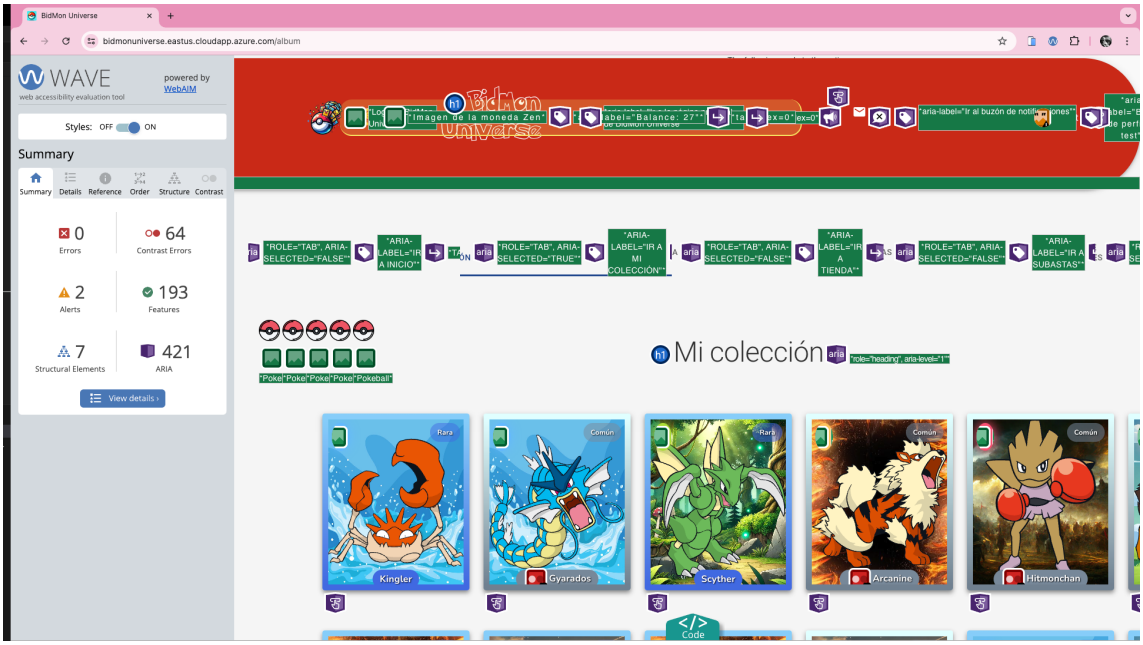


Figura 8.10: Accesibilidad Página Colección de cartas del usuario

## Tienda

- **Resultado:** La vista de la tienda cumple con los requisitos de accesibilidad.

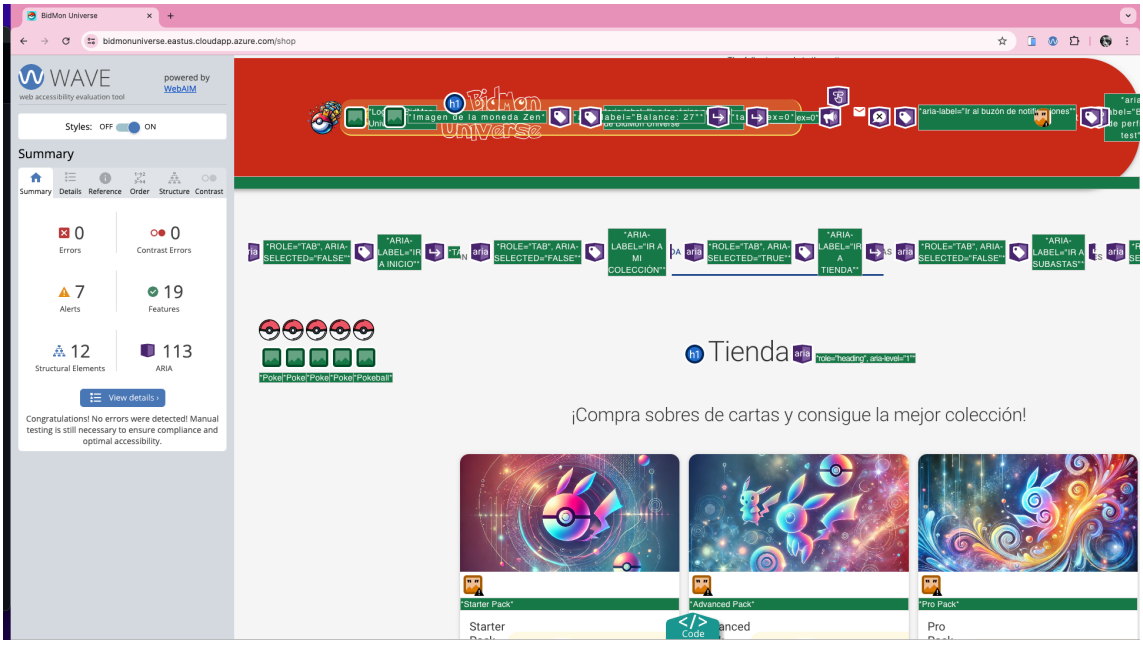


Figura 8.11: Accesibilidad Página Tienda

## Recarga de saldo

- **Resultado:** La vista de recarga de saldo cumple con los requisitos de accesibilidad.

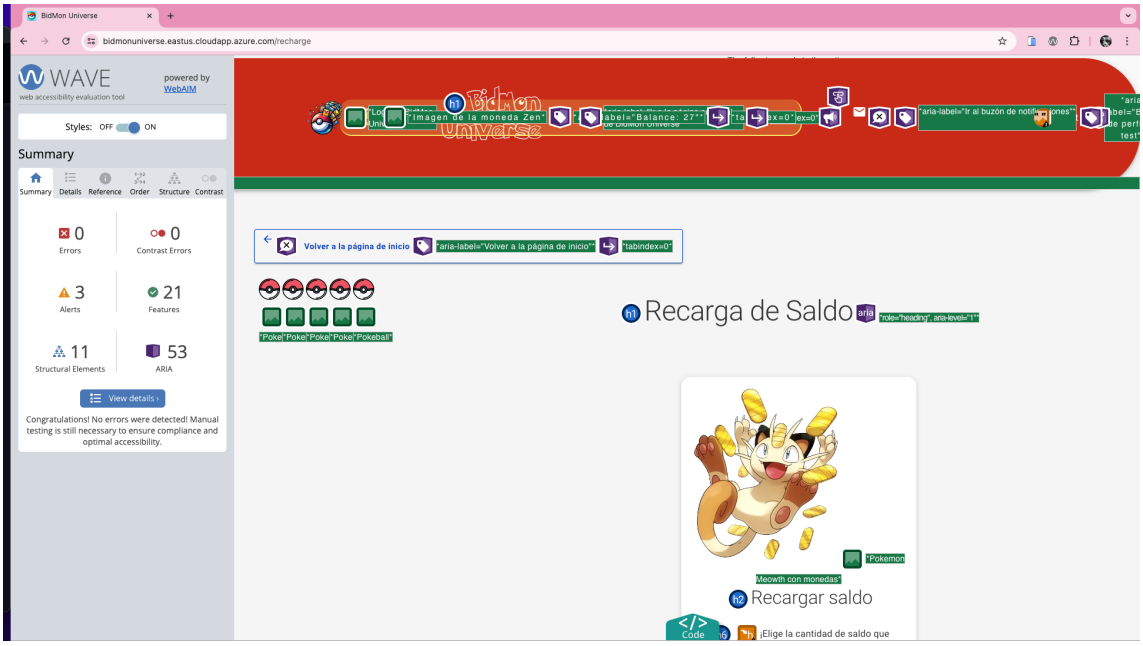


Figura 8.12: Accesibilidad Página Recarga de saldo

### Subastas activas

- **Resultado:** La vista de subastas activas cumple con los requisitos de accesibilidad.
- **Observaciones:** Los errores de contraste se deben a los colores de las cartas y son aceptados.

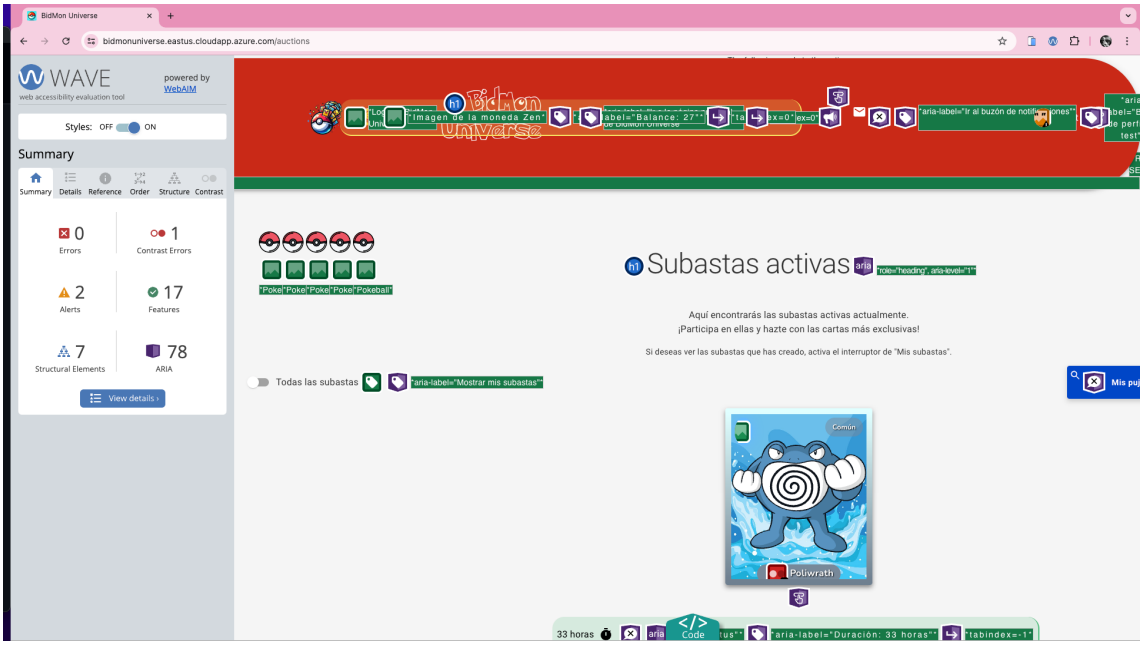


Figura 8.13: Accesibilidad Página Subastas activas

### Detalle de una carta

- **Resultado:** La vista de detalle de una carta cumple con los requisitos de accesibilidad.
- **Observaciones:** Dependiendo de la carta que se consulte se puede obtener un error de contraste se debe al color de la carta y es aceptado.

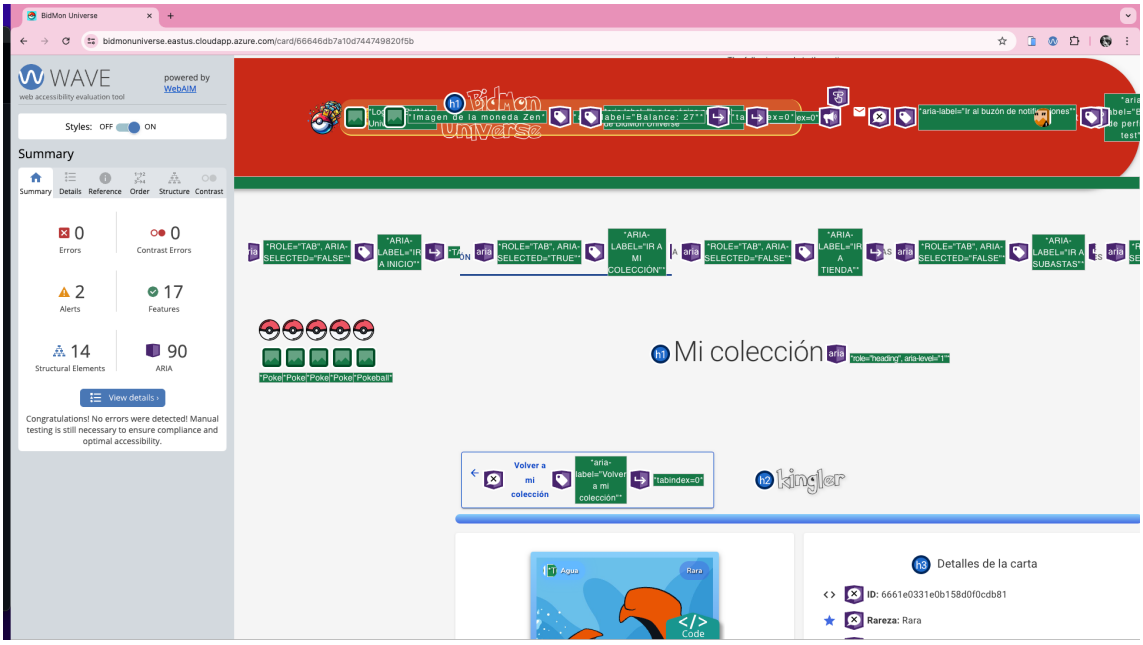


Figura 8.14: Accesibilidad Página Detalle de una carta

## Pujas activas

- **Resultado:** La vista de pujas activas cumple con los requisitos de accesibilidad.
- **Observaciones:** Si hay pujas activas, se pueden obtener errores de contraste que se deben a los colores de las cartas y son aceptados.

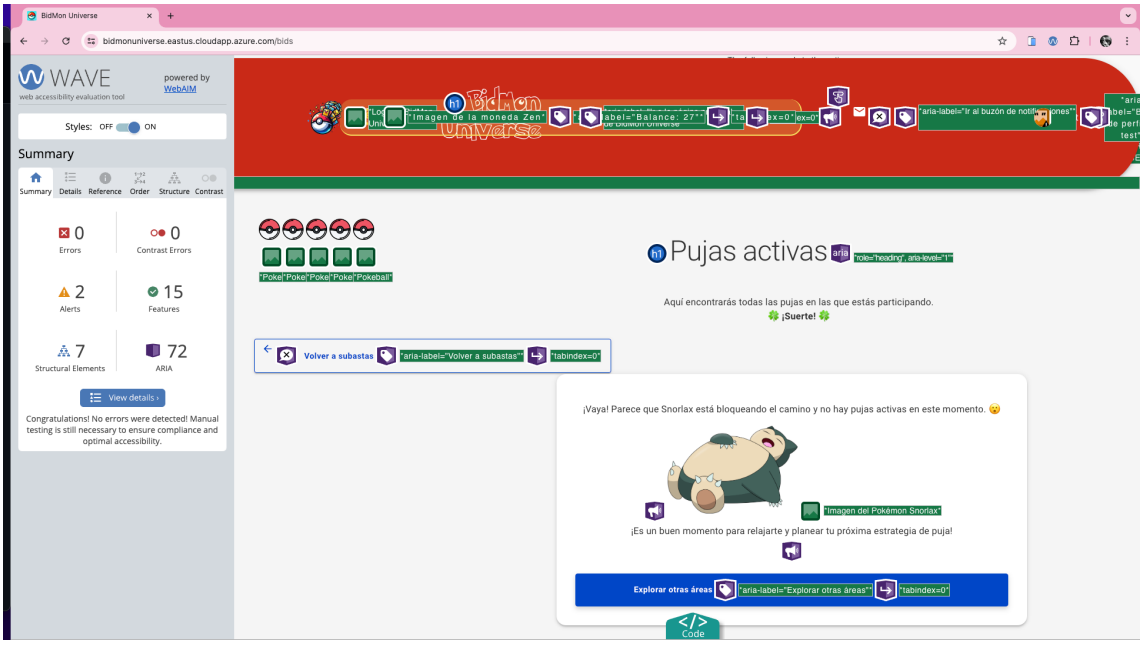


Figura 8.15: Accesibilidad Página Pujas activas

### Historial de transacciones

- **Resultado:** La vista de historial de transacciones cumple con los requisitos de accesibilidad salvo por un error en una etiqueta de formulario.
- **Observaciones:** El error en la etiqueta de formulario se debe a que no se ha asociado correctamente con el campo de entrada, en concreto, es en el campo de paginación que crea automáticamente la librería de componentes utilizada. Este error no afecta a la accesibilidad de la página, ya que es en la etiqueta de la páginación y no en los propios controles de la tabla. Se puede ver detallado en la Figura 8.17: Accesibilidad Página Historial de transacciones - Error en etiqueta de formulario.

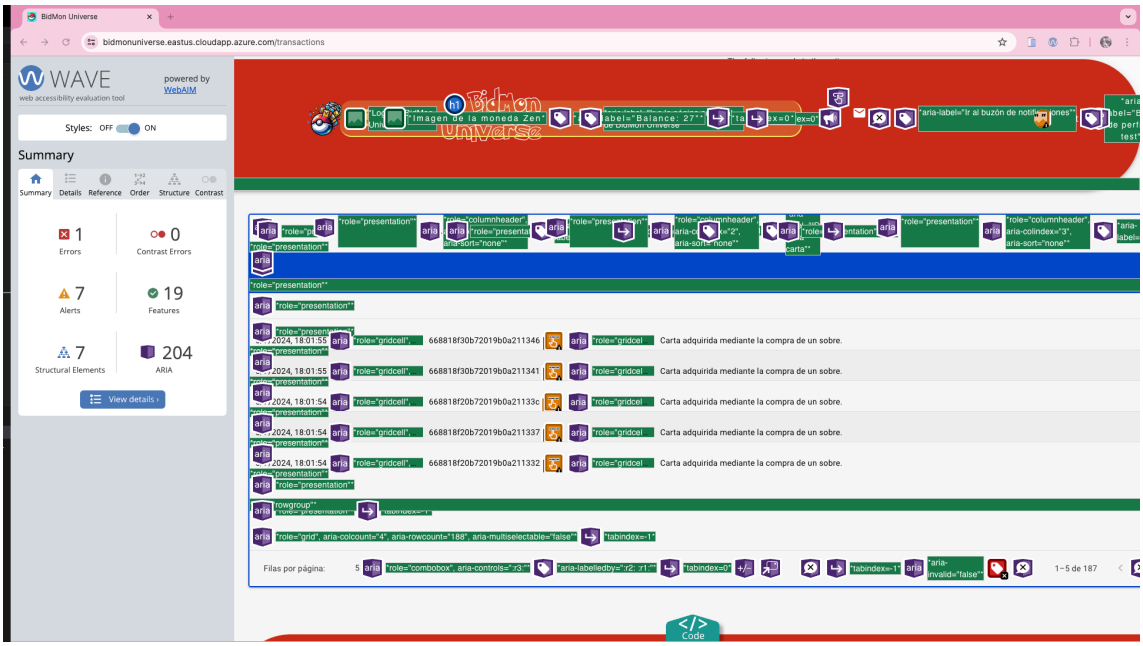


Figura 8.16: Accesibilidad Página Historial de transacciones

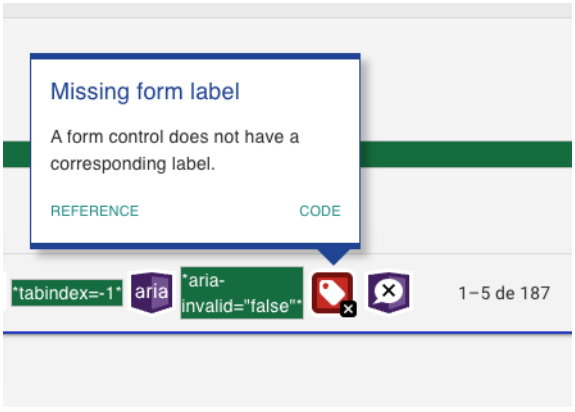


Figura 8.17: Accesibilidad Página Historial de transacciones - Error en etiqueta de formulario

# PRUEBAS DE ADAPTABILIDAD

En este anexo se presentan los resultados de las pruebas de accesibilidad realizadas en las vistas principales de la aplicación. Las capturas se han realizado con un iPhone 12 utilizando el navegador Safari en modo claro. Las especificaciones de la pantalla del dispositivo pueden consultarse en Apple - iPhone 12 - Especificaciones.

Se ha decido realizar las capturas de pantalla de las vistas principales de la aplicación en modo claro, ya que es el modo por defecto en el que se visualiza la aplicación. Además, solo se detallan las capturas para el iPhone 12, ya que es el dispositivo con la pantalla más pequeña y el que, por tanto, puede presentar más problemas de adaptabilidad.

## Home

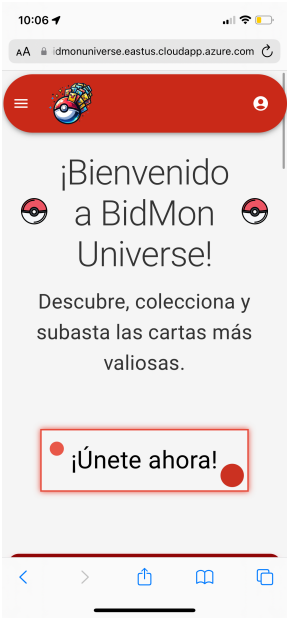


Figura 8.18: Adaptabilidad Página Home

## Login

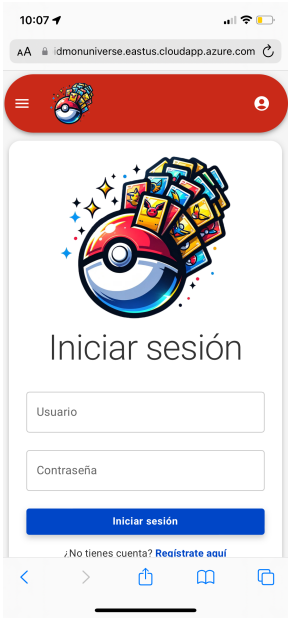


Figura 8.19: Adaptabilidad Página Login

## Registro

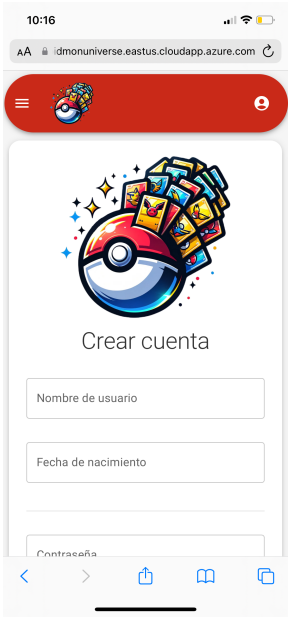


Figura 8.20: Adaptabilidad Página Registro

## Acerca de

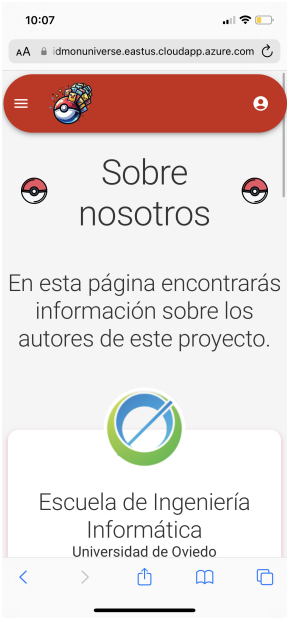


Figura 8.21: Adaptabilidad Página Acerca de

## Perfil

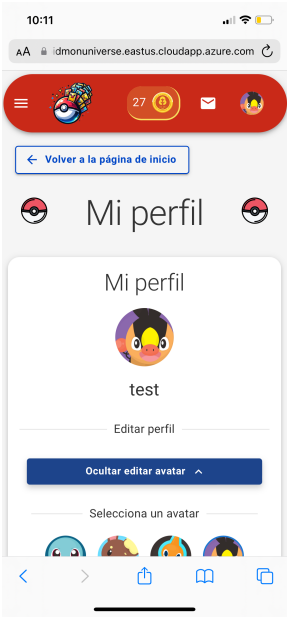


Figura 8.22: Adaptabilidad Página Perfil

## Página principal de usuario autenticado

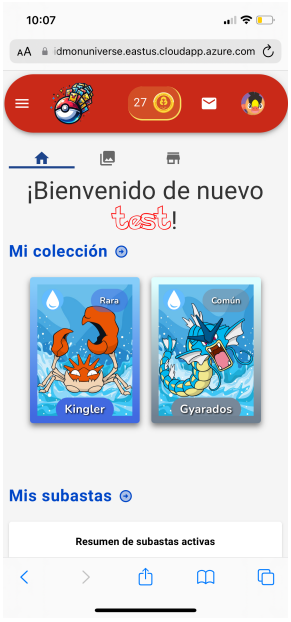


Figura 8.23: Adaptabilidad Página principal de usuario autenticado

## Colección de cartas del usuario


Figura 8.24: Adaptabilidad Página Colección de cartas del usuario

## Tienda

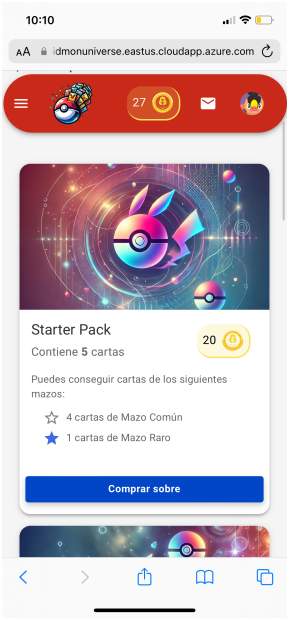


Figura 8.25: Adaptabilidad Página Tienda

## Detalle de una carta

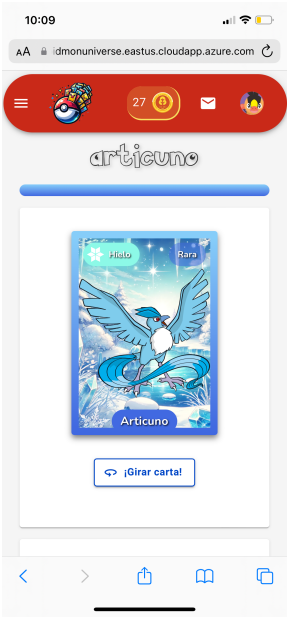


Figura 8.26: Adaptabilidad Página Detalle de una carta

## Subastas activas

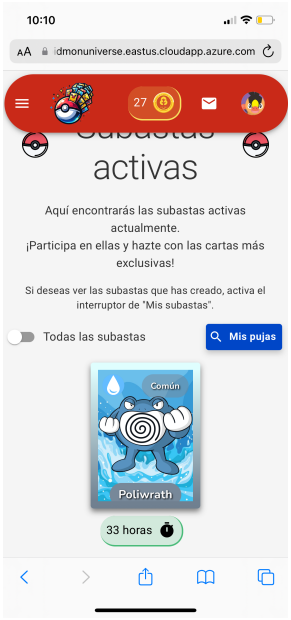


Figura 8.27: Adaptabilidad Página Subastas activas

## Detalle de una subasta

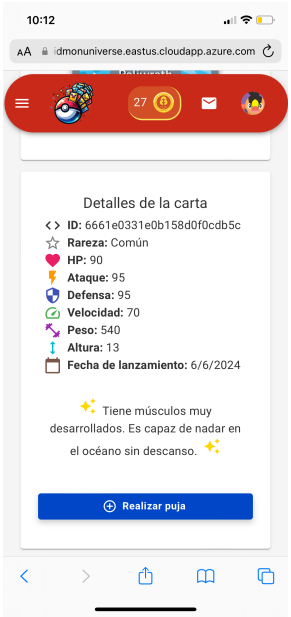


Figura 8.28: Adaptabilidad Página Detalle de una subasta

## Subastas propias

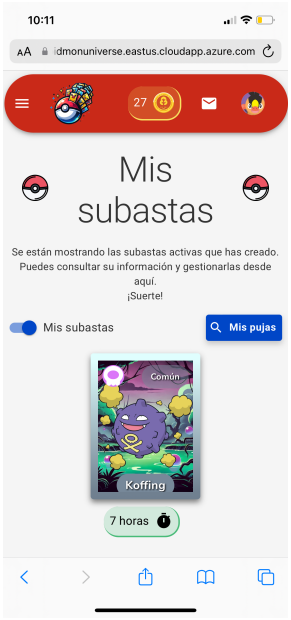


Figura 8.29: Adaptabilidad Página Subastas propias

## Mis pujas activas

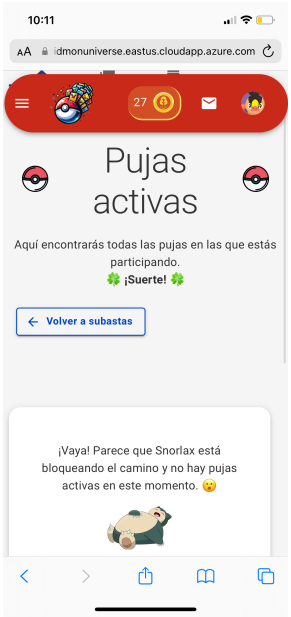


Figura 8.30: Adaptabilidad Página Mis pujas activas

### Recarga de saldo

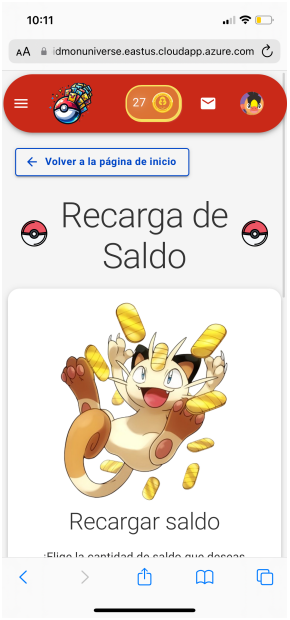


Figura 8.31: Adaptabilidad Página Recarga de saldo

### Historial de transacciones

En la versión móvil se puede hacer clic en una celda de la tabla para ver los detalles de la transacción.


Figura 8.32: Adaptabilidad Página Historial de transacciones

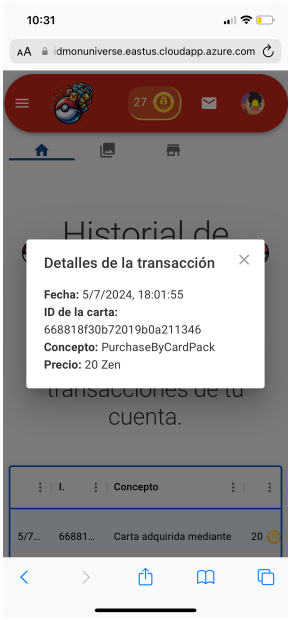


Figura 8.33: Adaptabilidad Detalle de una transacción

## Actualizar perfil

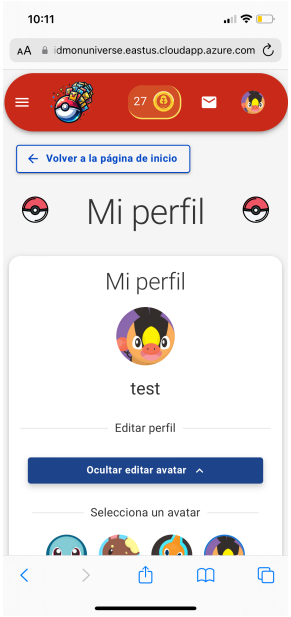


Figura 8.34: Adaptabilidad Página Actualizar perfil

### Menú general de la aplicación (usuario autenticado)

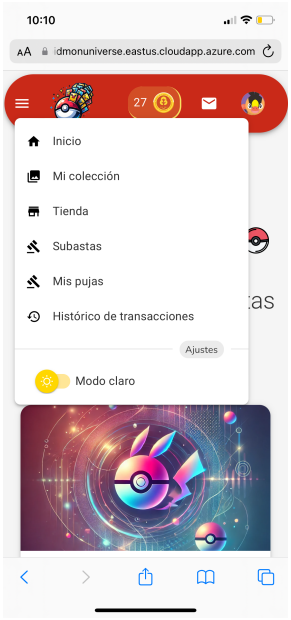


Figura 8.35: Adaptabilidad Menú general de la aplicación

### Resumen de los resultados

La aplicación cumple con los requisitos de adaptabilidad a diferentes resoluciones de pantalla. Se ha comprobado la adaptabilidad de la aplicación en las resoluciones de pantalla más comunes, como las de los dispositivos móviles, tablets y portátiles.

# GLOSARIO DE VARIABLES

En esta sección se detallan los distintos parámetros de configuración definidos en la aplicación. Estos parámetros se encuentran en la tabla 8.13.

Tabla 8.13: Parámetros de configuración

| Parámetro | Descripción |
|---|---|
| **GU_NOMBRE_USUARIO** | El nombre de usuario debe tener una longitud mínima de 6 caracteres y máxima de 20. Además, solo puede contener letras y números. |
| **GU_CONTRASEÑA** | La contraseña debe tener una longitud mínima de 8 caracteres y máxima de 20. Además, debe contener al menos una letra mayúscula, una letra minúscula, un número y un carácter especial. |
| **GU_MIN_EDAD** | La edad mínima para usar el sistema es 18 años. |
| **GU_FECHA_NACIMIENTO** | El formato de la fecha es DD/MM/AAAA. |
| **GU_ROL_DEFECTO** | El rol por defecto será *"STANDARD"*. |
| **GU_IMG_PERFIL_DEFECTO** | La imagen de perfil por defecto será la de un avatar. |
| **GU_SALDO_DEFECTO** | El saldo por defecto será de 100 monedas. |
| **GU_TOKEN_SESION** | El token de sesión no tendrá una duración máxima. |
| **GU_EQUIVALENCIA** | La equivalencia de monedas a dinero real será de 1 moneda = 0.01 euros. |
| **GU_RESTRICCIONES_RECARGA** | La cantidad de monedas a recargar debe ser un número entero positivo. La cantidad mínima a recargar es de 10 monedas. Se recargarán monedas en múltiplos de 10. |
| **GU_MÉTODOS_BANCARIOS** | El método bancario aceptado será PayPal. |
| **CC_FORMATO_IDC_CARTA** | El formato del identificador de colección de carta será C-XX-YY, donde XX es un número entero de dos dígitos y YY es un número entero de dos dígitos. |
| **CC_FORMATO_IDC_SOBRE** | El formato del identificador de colección de carta será CP-XX, donde XX es un número entero de dos dígitos. |

# CONTENIDO ENTREGADO EN LOS ANEXOS

## Contenidos

Además de este documento, se hace entrega de una carpeta comprimida que contiene todo el código fuente de la aplicación y otros archivos necesarios para la correcta comprensión del proyecto. A continuación se detallan los archivos que contiene dicha carpeta:

- **documentación** ->Archivo PDF con la memoria final del proyecto.
- **anexos** ->Contiene los archivos necesarios para la correcta comprensión del proyecto.
- **codigo_UO269745.zip** ->Carpeta que contiene todo el código fuente de la aplicación.

## Código fuente

El código fuente se divide en dos carpetas principales: **restapi** y **webapp**. Además, se incluyen los archivos necesarios para el despliegue de la aplicación en un servidor de producción, así como el archivo de configuración de GitHub Actions para la integración continua.

- **restapi** ->Contiene el código fuente de la API REST desarrollada con Node.js y Express.
- **webapp** ->Contiene el código fuente de la aplicación web desarrollada con React.
- **.github/workflows/TFG.yml** ->Archivo de configuración de GitHub Actions para la integración continua.
- **Dockerfile** ->Archivo de configuración de Docker para el despliegue de la aplicación en un contenedor.
- **docker-compose-deploy.yml** ->Archivo de configuración de Docker Compose para el despliegue de la aplicación en un servidor de producción.
- **docker-compose.yml** ->Archivo de configuración de Docker Compose para el despliegue de la aplicación en un entorno de desarrollo.
- **README.md** ->Archivo con la descripción del proyecto e información sobre el autor.
- **.gitignore** ->Archivo de configuración de Git para ignorar archivos y directorios.

En la sección 6.5 ARQUITECTURA DE LOS SUBSISTEMAS DE ANÁLISIS se puede encontrar la información detallada de la arquitectura del sistema.

## Anexos

En la carpeta de anexos se incluyen los siguientes directorios y archivos:

- **presupuestos** ->Contiene los presupuestos de la aplicación en formato Excel.
  - **presupuesto_inicial_UO269745.xlsx** ->Presupuesto estimado antes del desarrollo de la aplicación.

- **presupuesto_final_UO269745.xlsx** ->Presupuesto tras el desarrollo de la aplicación.

- **planificaciones** ->Contiene las planificaciones del proyecto en formato MS Project.
  - **planificacion_inicial_UO269745.mpp** ->Planificación estimada antes del desarrollo de la aplicación.
  - **planificacion_final_UO269745.mpp** ->Planificación final tras el desarrollo de la aplicación.

# Bibliografía

[1] Jose Manuel Redondo, "Documentos-modelo para Trabajos de Fin de Grado/Master de la Escuela de Informática de Oviedo." https://www.researchgate.net/publication/327882831_Plantilla_de_Proyectos_de_Fin_de_Carrera_de_la_Escuela_de_Informatica_de_Oviedo, 2019. Accessed: 13-Jul-2020.

[2] Jose Manuel Redondo, "Creación y evaluación de plantillas para trabajos de fin de grado como buena práctica docente," *Revista de Innovación y Buenas Prácticas Docentes*, vol. pp, no. pp, p. pp, 2020.

[3] Electronic Arts Inc., "EA SPORTS FC™ ultimate team web app - sitio oficial de EA SPORTS." https://www.ea.com/es-es/games/ea-sports-fc/ultimate-team/web-app, 2024. Accessed: 30-Mar-2024.

[4] J. Sanmartín, "6.000 millones de dólares: esa es la cifra que ha ingresado EA en los últimos seis años con FIFA ultimate team." https://www.vidaextra.com/deportes/6-000-millones-dolares-esa-cifra-que-ha-ingresado-ea-ultimos-seis-anos-fifa-ultimate-team, 2021. Accessed: 30-Mar-2024.

[5] E. A. Inc., "Annual report 2023." https://s22.q4cdn.com/894350492/files/doc_financials/2023/ar/418941-1-_8_Electronic-Arts-Inc-Proxy10-K-Combo_clean.pdf, 2023. Accessed: 29-May-2024.

[6] Take-Two Interactive, "Take-two interactive annual report 2023." https://ir.take2games.com/static-files/9aa44006-1d93-43cf-99f8-3763578e29f6, 2023. Accessed: 30-May-2024.

[7] eBay Inc., "Financial summary." https://investors.ebayinc.com/financial-information/financial-summary/default.aspx, 2024. Accessed: 01-Jun-2024.

[8] Catawiki, "Información relativa a catawiki." https://www.catawiki.com/es/v/register, 2024. Accessed: 31-May-2024.

[9] Project Management Institute, *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*. Newtown Square, Pennsylvania: Project Management Institute, 5th ed., 2013.

[10] A. Cockburn, *Writing Effective Use Cases*. Addison-Wesley Professional, 2000.

[11] Object Management Group, "Unified modeling language: A specification defining a graphical language for visualizing, specifying, constructing, and documenting the artifacts of distributed object systems." https://www.omg.org/spec/keyword/component-diagram, 2017. Version 2.5.1, formal, diciembre 2017, Accessed: 27-Jun-2024.

[12] B. W. Boehm, "Software risk management: Principles and practices," *IEEE Software*, vol. 8, pp. 32–42, Jan. 1991.

[13] J. M. Requena, "El consejero de Universidad pide apostar por la innovación y generar conocimiento." https://www.lne.es/asturias/2019/08/13/consejero-universidad-pide-apostar-innovacion/2514937.html, 2019. Accessed: 13-Jul-2020.

[14] S. McConnell, *Desarrollo y Gestión de Proyectos Informáticos*. McGraw-Hill / Interamericana de España, 1 ed., 1997.

[15] LaLiga, "Cuentas anuales 2022-2023." https://assets.laliga.com/assets/2023/12/18/originals/b7cb46afb70d0e373fd5136aa219d110.pdf, 2023. Accessed: 30-May-2024.

[16] H. Lebredo, "Diagrama de casos de uso." https://emctwo.home.blog/2022/10/22/diagrama-de-casos-de-uso/, October 2022. Accessed: 01-Jun-2024.

[17] H. Lebredo, "Diagrama de clases." https://emctwo.home.blog/2022/11/27/diagrama-de-clases/, November 2022. Accessed: 01-Jun-2024.

[18] U. T. en Español, "Diagrama de componentes i - tutorial uml en español." https://www.youtube.com/watch?v=oOycG_n1ARs, 2024. Accessed: 03-Jul-2024.

[19] Audicity, "Uml structural diagrams: Component diagram - georgia tech - software development process." https://www.youtube.com/watch?v=ipKJwnPsst8, 2024. Accessed: 03-Jul-2024.

[20] Audicity, "Uml structural diagrams: Deployment - georgia tech - software development process." https://www.youtube.com/watch?v=IzEzX5HW_CU, 2024. Accessed: 03-Jul-2024.

[21] N. N. Group, "Usability testing 101." https://www.nngroup.com/articles/usability-testing-101/, 2024. Accessed: 01-Jul-2024.

[22] Hotjar, "El proceso de test de usabilidad." https://www.hotjar.com/es/test-de-usabilidad/proceso/, 2024. Accessed: 01-Jul-2024.

[23] A. Osmani, *Learning JavaScript Design Patterns: A JavaScript and React Developer's Guide*. Sebastopol, CA: O'Reilly Media, 2nd ed., 2023.

[24] TypeScript, "Typescript: Typed javascript at any scale." https://www.typescriptlang.org, 2024. Accessed: 04-Jul-2024.

[25] GitHub, "Github: Where the world builds software." https://github.com/, 2024. Accessed: 04-Jul-2024.

[26] G. SCM, "Git - distributed version control system." https://git-scm.com/, 2024. Accessed: 04-Jul-2024.

[27] GitKraken, "Gitkraken: Git gui and issue boards." https://www.gitkraken.com/, 2024. Accessed: 04-Jul-2024.

[28] Node.js, "Node.js." https://nodejs.org/en, 2024. Accessed: 04-Jul-2024.

[29] R. Slick, "React slick: Carousel component built with react." https://react-slick.neostack.com, 2024. Accessed: 04-Jul-2024.

[30] MUI, "Mui: React components for faster and easier web development." https://mui.com, 2024. Accessed: 04-Jul-2024.

[31] F. Motion, "Framer motion: A production-ready motion library for react." https://www.framer.com/motion/, 2024. Accessed: 04-Jul-2024.

[32] Socket.IO, "Socket.io: Documentation for socket.io." https://socket.io/docs/v4/client-api/, 2024. Accessed: 04-Jul-2024.

[33] Yup, "Yup: Javascript schema builder for value parsing and validation." https://www.npmjs.com/package/yup, 2024. Accessed: 04-Jul-2024.

[34] SweetAlert2, "Sweetalert2: A beautiful, responsive, customizable, accessible (wai-aria) replacement for javascript's popup boxes." https://sweetalert2.github.io, 2024. Accessed: 04-Jul-2024.

[35] R. H. Form, "React hook form: Performant, flexible and extensible forms with easy-to-use validation." https://react-hook-form.com, 2024. Accessed: 04-Jul-2024.

[36] R. P. JS, "React paypal js: Official paypal react library." https://www.npmjs.com/package/@paypal/react-paypal-js, 2024. Accessed: 04-Jul-2024.

[37] R. R. DOM, "React router dom: Declarative routing for react." https://www.npmjs.com/package/react-router-dom, 2024. Accessed: 04-Jul-2024.

[38] Jest, "Jest: Delightful javascript testing." https://jestjs.io/es-ES/, 2024. Accessed: 04-Jul-2024.

[39] Puppeteer, "Puppeteer: Headless chrome node.js api." https://pptr.dev, 2024. Accessed: 04-Jul-2024.

[40] Cucumber, "Cucumber: Tools for executable specifications." https://cucumber.io, 2024. Accessed: 04-Jul-2024.

[41] Express.js, "Express.js: Fast, unopinionated, minimalist web framework for node.js." https://expressjs.com/es/, 2024. Accessed: 04-Jul-2024.

[42] JWT, "Jwt.io: Json web tokens introduction." https://jwt.io, 2024. Accessed: 04-Jul-2024.

[43] bcrypt, "bcrypt: A library to help you hash passwords." https://www.npmjs.com/package/bcrypt, 2024. Accessed: 04-Jul-2024.

[44] Mongoose, "Mongoose: Elegant mongodb object modeling for node.js." https://mongoosejs.com, 2024. Accessed: 04-Jul-2024.

[45] CORS, "Cors: Node.js package for providing a connect/express middleware that can be used to enable cors." https://www.npmjs.com/package/cors, 2024. Accessed: 04-Jul-2024.

[46] csv parser, "csv-parser: Streaming csv parser that aims to follow the node.js stream api." https://www.npmjs.com/package/csv-parser, 2024. Accessed: 04-Jul-2024.

[47] csv writer, "csv-writer: A library to write csv files in node.js easily." https://www.npmjs.com/package/csv-writer, 2024. Accessed: 04-Jul-2024.

[48] Axios, "Axios: Promise based http client for the browser and node.js." https://axios-http.com/es/docs/intro, 2024. Accessed: 04-Jul-2024.

[49] SuperTest, "Supertest: Super-agent driven library for testing node.js http servers." https://www.npmjs.com/package/supertest, 2024. Accessed: 04-Jul-2024.

[50] Notepad++, "Notepad++: A free (as in "free speech.ªnd also as in "free beer") source code editor and notepad replacement." https://notepad-plus-plus.org, 2024. Accessed: 04-Jul-2024.

[51] MobaXterm, "Mobaxterm: Enhanced terminal for windows with x11 server, tabbed ssh client, network tools, and much more." https://mobaxterm.mobatek.net, 2024. Accessed: 04-Jul-2024.

[52] LaTeX, "Latex: A document preparation system." https://www.latex-project.org, 2024. Accessed: 04-Jul-2024.

[53] Overleaf, "Overleaf: Online latex editor." https://www.overleaf.com, 2024. Accessed: 04-Jul-2024.

[54] M. Azure, "Microsoft azure: Cloud computing services." https://azure.microsoft.com, 2024. Accessed: 04-Jul-2024.

[55] MongoDB, "Mongodb: The database for modern applications." https://www.mongodb.com, 2024. Accessed: 04-Jul-2024.

[56] React, "React: A javascript library for building user interfaces." https://reactjs.org, 2024. Accessed: 04-Jul-2024.

[57] A. A. J. Fuente and B. L. Pérez, *Guía de Aprendizaje de la asignatura Dirección y Planificación de Proyectos Informáticos*. Escuela de Ingeniería Informática, Universidad de Oviedo, 0.093 ed., Feb. 2022.