UNIVERSIDAD DE OVIEDO

ESCUELA DE INGENIERÍA INFORMÁTICA

TRABAJO FIN DE GRADO

"Herramienta generadora de imágenes a partir de texto"

AUTOR: Diego Santos Neila

DIRECTOR: Cristian González García

Declaración de originalidad

Yo Diego Santos Neila, estudiante del Grado en Ingeniería Informática del Software, declaro que el presente Trabajo de Fin de Grado es una obra original realizada íntegramente por mi persona.

Asimismo, aseguro que todas las fuentes y datos de otros autores que he utilizado han sido debidamente citadas en el presente documento, respetando las normativas éticas y académicas.

Agradecimientos

Querría agradecer a todas las personas que han formado parte de mi vida durante los años de realización del grado, en especial a mi familia, mi círculo de amigos y otras personas cercanas que me han apoyado y acompañado durante el proceso, ya que, en los momentos de menor motivación o estrés por el estudio, son los que realmente me han ayudado y motivado para poder hacer el esfuerzo necesario para completar el grado.

También una especial mención a los compañeros que he conocido durante estos años.

Resumen

Hoy en día, durante la realización del proyecto, nos encontramos en la época que mayor auge, adopción del público general, inversión y avances continuos ha experimentado el sector de la inteligencia artificial, por lo que la realización de este proyecto se encuentra en el mejor momento posible hasta el momento para ser llevado a cabo. Dentro de este sector, el tipo de inteligencia artificial que se encuentra como número uno en todas estas dimensiones y además está en la boca de todos, es la inteligencia artificial generativa, de la cual trata este proyecto, más concretamente, la inteligencia artificial generativa de imágenes.

El objetivo de este trabajo es la implementación de una herramienta, que, a través del uso de modelos de inteligencia artificial generativa, sea capaz de generar un número prácticamente ilimitado de imágenes, relacionadas con una descripción textual de estas imágenes proporcionada por el usuario, sin la necesidad de tener una infraestructura de recursos *hardware* extremadamente alta y por lo tanto que pueda ser utilizada como una aplicación de escritorio en un equipo con unas especificaciones *hardware* relativamente asequibles dentro del utilizado para la inferencia con grandes modelos de inteligencia artificial. Además, esta herramienta permitirá utilizar diferentes modelos y poder configurar algunos de sus parámetros de forma que se añade cierta flexibilidad a los resultados buscados, teniendo en cuenta que las imágenes generadas serán guardadas en el equipo en el que se ejecute esta aplicación, quedando a total disponibilidad para el usuario.

Para ello se implementará una interfaz de usuario y la funcionalidad de la generación de imágenes a través del uso de una *API*, a través de la cuál, se abre la posibilidad de interaccionar con modelos de difusión estable, los más avanzados para llevar a cabo la tarea buscada en el proyecto a día de hoy.

Por lo que esta herramienta ofrece multitud de posibilidades al público general y a los usuarios con experiencia en el tema, entre estas posibilidades se encuentra la posibilidad de generar *datasets* de imágenes personalizados, tareas de creatividad o tareas de experimentación con modelos de inteligencia artificial de manera completamente local, entre otras muchas posibilidades solamente limitadas por la creatividad de los usuarios.

Palabras Clave

Inteligencia Artificial, Generación De imágenes, Difusión Estable, Procesamiento De Lenguaje Natural, Aprendizaje Profundo, Aprendizaje automático.

Abstract

Nowadays, during the development of this project, we are witnessing the greatest period of growth, public adoption, investment, and continuous advancements in the field of artificial intelligence. As such, the execution of this project is happening at the most opportune time to date. Within this field, the type of artificial intelligence that stands out as number one in all these dimensions and is currently a topic of global discussion is generative artificial intelligence. This project specifically focuses on generative image-based artificial intelligence.

The objective of this work is to develop a tool that, through the use of generative AI models, is capable of generating an almost unlimited number of images based on textual descriptions provided by the user. This tool is designed to function without requiring an extremely high-end hardware infrastructure, making it accessible as a desktop application that can run on hardware with relatively affordable specifications, compared to what is typically needed for large-scale AI model inference. Additionally, the tool will allow users to work with different models and configure various parameters, adding flexibility to the desired results. The generated images will be saved directly on the user's device, making them fully available for personal use.

To achieve this, a user interface will be implemented, along with the functionality for image generation through the use of an API. This API will enable interaction with state-of-the-art stable diffusion models, which are currently the most advanced models for this type of task.

This tool offers numerous possibilities for both the general public and experienced users. These include the creation of customized image datasets, creative tasks, or experimentation with AI models entirely locally, so the potential applications are vast and limited only by the creativity of the users.

Keywords

Artificial Intelligence, Image Generation, Stable Diffusion, Natural Language Processing, Deep Learning, Machine Learning.

Índice General

Índice de Figuras

Capítulo 1. Memoria del Proyecto

A lo largo de este capítulo se presenta el proyecto, de forma que se describe la motivación de la realización de este, además de los objetivos y el alcance del proyecto.

1.1 Resumen de la Motivación, Objetivos y Alcance del Proyecto

Hoy en día nos encontramos en el momento donde mayor auge ha experimentado la inteligencia artificial y más en concreto la inteligencia artificial generativa donde han surgido multitud de nuevos avances y una adopción por parte del público general de aplicaciones desarrolladas con este tipo de modelos de inteligencia artificial, como por ejemplo el más conocido *ChatGPT*, capaz de sintetizar ideas y razonamientos de una forma muy similar a un ser humano, además de ofrecer la funcionalidad de generar imágenes a partir de descripciones de texto proporcionadas por el usuario, lo cual se relaciona más con el presente proyecto. Por lo tanto, este proyecto se encuadra en el mejor momento para llevarse a cabo gracias a este auge y avances comentados y adopción de estas herramientas por parte del público general. Aunque este tipo de herramientas conocidas presentan ciertas desventajas, como por ejemplo que los resultados obtenidos están limitados por las políticas de empresa adoptadas por estas empresas desarrolladoras de los sistemas, de forma que los resultados obtenidos se pueden ver limitados y no obtener los resultados buscados o verse afectados por la necesidad de tener que pagar versiones *premium*. Además, este tipo de aplicaciones se caracterizan por la necesidad de tener una conexión a internet, solamente poder generar un número de imágenes limitado, tener cupos de uso, ofrecer un número de modelos limitado y ofrecer rigidez en la capacidad de configuración de los parámetros o configuraciones de los modelos.

Por lo que este proyecto busca solucionar la serie de problemas que nos presentan este tipo de herramientas online más conocidas, de forma que este proyecto busca ofrecer una herramienta gratuita capaz de generar un número de imágenes ilimitado, sin la necesidad de una conexión a internet, centralizando en la misma herramienta diferentes modelos con diferentes características y ofreciendo la flexibilidad al usuario de poder modificar los parámetros y configuraciones de cada uno de estos modelos ofrecidos por la herramienta.

Por lo tanto, el objetivo de este proyecto es desarrollar una aplicación de escritorio con interfaz gráfica de usuario, de forma que pueda ser utilizada sin la necesidad de tener grandes conocimientos técnicos ni conexión a internet, la cual permita introducir una descripción textual de las imágenes que el usuario quiera generar y la herramienta sea capaz de generar un número de imágenes ilimitado proporcionado por el usuario, mediante una serie de modelos de inteligencia artificial generativa, además de ofrecer la posibilidad a los usuarios con mayor conocimiento técnico la posibilidad de modificar los parámetros utilizados por el modelo para llevar a cabo la generación de imágenes, de forma que se puedan obtener resultados más precisos según los objetivos del usuario y guardar estas imágenes en el equipo utilizado de forma local por los usuarios.

Por lo que el proyecto ofrece una solución con grandes posibilidades y aplicaciones, pudiendo impulsar tareas de creatividad, realizar labores de experimentación o generar grandes conjuntos de datos de imágenes de forma que se puedan obtener *datasets* personalizados de forma totalmente gratuita y solamente limitado por el rendimiento del hardware utilizado o la creatividad ofrecida por el usuario a la hora de generar imágenes.

1.2 Resumen de Todos los Aspectos

- **Capítulo 1. Memoria del proyecto**: Resumen de los aspectos tratados durante el proyecto.
- **Capítulo 2. Introducción**: Descripción de la justificación y objetivos del proyecto, estudio de la situación actual y evaluación de las alternativas estudiadas para llevar a cabo el proyecto.
- **Capítulo 3. Aspectos teóricos**: Descripción de los conceptos clave relacionados con el proyecto, necesarios para entender lo expuesto a lo largo del proyecto.
- **Capítulo 4. Planificación y presupuesto iniciales del proyecto**: Se expone la estimación previa realizada para llevar a cabo el proyecto en cuanto a la planificación de tareas a llevar a cabo para completar el proyecto junto al presupuesto estimado del proyecto.
- **Capítulo 5. Análisis**: Se expone la especificación de requisitos del proyecto junto a la documentación relativa al análisis del sistema a desarrollar, relacionada con estos requisitos.
- **Capítulo 6. Diseño del sistema**: Se expone la documentación obtenida tras la fase de análisis de forma que se representará la estructura y arquitectura del sistema y código que servirá como guía para llevar a cabo la implementación del sistema.
- **Capítulo 7. Implementación**: Se describe el proceso de desarrollo de la herramienta llevado a cabo, junto a los elementos y herramientas necesarios, utilizados para llevar a cabo el desarrollo de la herramienta.
- **Capítulo 8. Desarrollo de las pruebas**: Se detalla el conjunto de pruebas realizado en el sistema, además de describir la implementación y proceso realizado para llevar a cabo las pruebas del sistema.
- **Capítulo 9. Manual del sistema**: Incluye los diferentes manuales desarrollados con diferentes enfoques y detalles en función del rol de la persona que utilizará cada manual, de forma que se contemplen todos los conocimientos necesarios a saber sobre el sistema.
- **Capítulo 10. Conclusiones y ampliaciones**: Se expone una recapitulación sobre lo desarrollado en el proyecto contemplando los resultados esperados con los obtenidos contemplando las elecciones tomadas durante el proceso, además de incluir posibles futuras mejoras como ampliación del proyecto desarrollado detallando como llevar a cabo estas mejoras y su justificación.
- **Capítulo 11. Planificación y presupuesto iniciales del proyecto**: Se exponen las tareas realizadas para llevar a cabo el proyecto junto a su presupuesto.
- **Capítulo 12. Referencias Bibliográficas.**
- **Capítulo 13. Apéndices.**

Capítulo 2. Introducción

A lo largo de este capítulo se describirá la justificación de la realización del proyecto, además de los objetivos de este. Para continuar se contemplará la situación actual relativa a sistemas similares existentes a este proyecto y para acabar se describirán las alternativas estudiadas previamente a la ejecución del proyecto para poder llevarlo a cabo de forma posterior.

2.1 Justificación del Proyecto

Hoy en día, durante el tiempo de desarrollo de este proyecto y unos cuántos meses previos al comienzo de este, ha surgido un gran auge en el desarrollo de nuevas aplicaciones de inteligencia artificial generativa y más concretamente en aplicaciones de generación de imágenes mediante modelos generativos, por lo que el desarrollo de este proyecto se encuentra en el momento idóneo para llevarse a cabo debido a este periodo de tiempo en el que se encuentra el mercado de alta demanda de este tipo de aplicaciones.

El lado menos positivo de este auge en el sector es que existe una gran competencia entre desarrolladores y entre empresas con dar con la tecla acertada en cuanto a quien desarrolla la aplicación de generación de imágenes más avanzada o llamativa, por poner en contexto existen ya ciertas aplicaciones similares en el mercado que llevan a cabo una tarea similar a la buscada por este proyecto. Entre ellas podemos destacar *Microsoft Designer* [1], *chatgpt* [2] o *MidJourney* [3] la cuales se tratan aplicaciones web similares con un campo de texto a través del cual se puede introducir una descripción textual de las imágenes a generar y esta aplicación nos generará una serie de imágenes correspondientes a esta descripción proporcionada. Entre los problemas que podemos encontrar en estas aplicaciones cabe destacar que tienen una limitación del número de imágenes que podemos generar, o para poder des limitar este número debemos obtener una versión de pago, además de tener que estar conectados a internet y vernos limitados por las políticas de empresa desarrolladoras de estas compañías, por lo que no podremos obtener en ciertos casos los resultados que podríamos esperar.

Por lo tanto, sabiendo esto, este proyecto ofrece una solución a un gran conjunto de limitaciones que presentan este tipo de aplicaciones más usadas hoy en día por un gran número de usuarios. Entre las soluciones que ofrece este proyecto se encuentra la posibilidad de generar imágenes de forma solamente limitada por los recursos *hardware* del equipo utilizado, la posibilidad de generar imágenes utilizando diferentes modelos centralizados en una misma aplicación, la flexibilidad de poder modificar los parámetros de los diferentes modelos y todo esto sin la necesidad de tener activa una conexión a internet ni depender de terceros.

Conociendo esta solución ofrecida, esta herramienta a desarrollar en el proyecto ofrece un gran abanico de posibilidades para diferentes ámbitos, como la generación de grandes *datasets* de imágenes personalizados, experimentación con diferentes modelos y configuraciones de estos o con fines de creatividad aplicados a muchos sectores diferentes que al usuario se le pueda ocurrir.

2.2 Objetivos del Proyecto

1. Crear una herramienta capaz de generar imágenes basadas en una entrada de texto descriptiva de ellas.
2. Permitir generar imágenes mediante diferentes modelos de inteligencia artificial generativa.
3. Permitir generar un número ilimitado de imágenes (solo limitado por el *hardware* utilizado)
4. Permitir modificar los parámetros de los modelos generativos a usar en la generación.
5. Proporcionar una interfaz gráfica de usuario para llevar a cabo las funcionalidades de la herramienta.
6. Permitir el seguimiento al usuario del estado en el que se encuentra el proceso de generación de las imágenes.
    a. Mostrando el número de imágenes actualmente generadas.
    b. Mostrando al usuario cada imagen generada en tiempo real en la interfaz de usuario.
7. Permitir al usuario guardar las imágenes generadas en un directorio de su equipo.

2.3 Estudio de la Situación Actual

Debido a la existencia de diferentes posibilidades para poder llevar a cabo el proyecto y las continuas salidas al mercado de diferentes soluciones como se menciona en 2.1, previamente a la realización del proyecto se ha hecho un estudio previo de evaluación de alternativas disponibles en el momento para poder cumplir con los objetivos mencionados en 2.2. Por lo tanto, a lo largo de esta sección se describirán las diferentes alternativas.

2.3.1 Evaluación de Alternativas

A continuación, a lo largo de la presente sección se describirán el conjunto de alternativas estudiadas previamente a la realización del proyecto para poder llevar a cabo la solución buscada en el proyecto. Para cada una de las alternativas estudiadas se incluirá una breve descripción, así como las ventajas y desventajas identificadas para cada una de ellas.

En concreto se tratarán las diferentes alternativas relacionadas con las librerías y *APIs* de generación de imágenes y las relacionadas con librerías de interfaz gráfica de usuario, de forma que a partir de este estudio se pueda decidir que se utilizará para llevar a cabo el desarrollo del proyecto.

2.3.1.1 Librerías y APIs de generación de imágenes

A continuación, se tratarán las diferentes alternativas relacionadas con las librerías y *APIs* de generación de imágenes.

2.3.1.1.1 OpenAI API

Esta *API* [4] ofrecida por la hoy en día conocida empresa *OpenAI* ofrece la funcionalidad de generación de imágenes a partir de descripciones de texto utilizando los modelos *DALL-E 2* [5] y *DALL-E 3* [6] de forma que esta API nos expone un *endpoint* y también una implementación en *Python* en forma de librería, a través de los cuáles podremos enviar como parámetro el texto descriptivo de la imagen o imágenes a generar y también podremos especificar otros parámetros como el modelo a utilizar, la calidad de imagen, el tamaño de cada imagen o el número de imágenes, entre otros. De forma que esta *API* nos devolverá la imagen o imágenes acorde a los parámetros especificados en la petición a la *API*.

2.3.1.1.1.1 Ventajas

- Integración sencilla en nuestra aplicación, ya que podremos acceder a los *endpoints* expuestos por la *API*, diseñados y documentados de forma sencilla en la página web de *OpenAI* sin la necesidad de configurar entornos ni instalar librerías complejas.
- Calidad de los resultados obtenidos, gracias a poder utilizar algunos de los modelos que hoy en día compiten con modelos de otras empresas por ofrecer los mejores resultados del mercado, así como tener acceso a las actualizaciones continuas de estos modelos ofrecidos por la *API*.

- Ahorro de recursos en el equipo local y eficiencia de la aplicación, gracias a que todo el cómputo se realiza en la nube de *OpenAI* diseñada para esta tarea, la cuál es conocida por necesitar una gran carga de recursos de cómputo y memoria.

2.3.1.1.1.2 Desventajas

- Coste de la generación, debido a que esta *API* es de pago por lo que, si se necesita generar una gran cantidad de imágenes, una alta calidad o descripciones complejas, puede resultar en obtener unos altos costes recurrentes por el uso de la aplicación [7].
- Restricciones de uso, dado que este tipo de *APIs* están sujetas a ciertas políticas de la empresa que la ofrece, por lo que es posible que si queremos generar ciertos contenidos la *API* no nos lo permita o nos genere resultados que se alineen con las políticas de la empresa y no con los resultados buscados, así como limitaciones de uso de recursos que pueda establecer la empresa en ciertos momentos de carga en los servidores.
- Exposición de datos sensibles, ya que esto puede ocurrir en caso de que busquemos llevar a cabo la generación de imágenes con ciertos datos que puedan tener un carácter sensible y estaríamos exponiendo datos sensibles a un servicio externo, lo cual casi siempre es un riesgo.

2.3.1.1.2 Google Gemini API

A continuación, se describirá la *API de Google Gemini* [8], desarrollada por *Google* como su propio nombre indica, es ofrecida de forma similar a la mencionada previamente de *OpenAI*, a través de *endpoints* y una librería de *Python*. A través de esta *API* podremos enviar una descripción textual y obtener una o varias imágenes que reflejen esta descripción utilizando el modelo de generación de imágenes de *Google Imagen 3* [9].

2.3.1.1.2.1 Ventajas

- Calidad de los resultados, gracias a que *Google* ofrece uno de los modelos de generación de imágenes que compiten por ser el mejor del mercado, por lo que podemos obtener resultados de gran calidad, fidelidad y realismo.
- Integración con *Google Cloud*, de forma que podremos integrar esta *API* con todo el ecosistema y servicios de la nube de *Google*, permitiendo llevar un mejor seguimiento de la facturación y centralizar multitud de servicios ofrecidos por *Google* en un mismo entorno.
- Ahorro de recursos locales, ya que esta *API* se ejecutará en los servidores en la nube de *Google*.
- Mejora continua en los modelos, de forma transparente gracias a las continuas actualizaciones y mejoras ofrecidas por *Google*.

2.3.1.1.2.2 Desventajas

- Acceso limitado, debido a que en estos momentos esta *API* se encuentra en fase *beta* por lo que el acceso o funcionalidades estará restringido a un número limitado de personas.

- Costes de uso, debido a que la *API* funciona a modo de servicio *Cloud* y por lo tanto al igual que la *API* mencionada previamente, si necesitamos generar una gran cantidad podemos incurrir en altos costes por el uso de la *API*.
- Dependencia de infraestructura de *Google*, de forma que para llevar a cabo la generación de las imágenes debemos tener conexión a internet y aunque sea poco usual, si ocurren problemas en la infraestructura de *Google* nos veremos afectados por ello.
- Restricciones de uso por políticas de la empresa, de manera que el contenido generado se pueda ver influenciado por políticas de empresa o verse restringido en caso de que se necesiten generar imágenes con algún tipo de contenido sensible. Además de poder vernos afectados por algún cupo de generación impuesto por la empresa de forma que no sea posible generar imágenes de manera ilimitada.

2.3.1.1.3 Python Diffusers

La librería *diffusers* [10] se trata de una librería de código abierto ofrecida por *Hugging Face*, a través de la cuál podremos utilizar un gran conjunto de modelos de difusión de generación de imágenes y diferentes funcionalidades a realizar con estos modelos, incluida la generación de imágenes a partir de descripciones textuales. Esta librería de *Python* nos facilitará el uso y descarga de diferentes modelos de difusión de generación de imágenes pre-entrenados ofrecidos a través de los repositorios de *Hugging Face*. Gracias a esto, esta librería permite integrar estos modelos en aplicaciones de forma local permitiendo tener un control total sobre el código de los modelos, así como de los parámetros, pipelines y procesamiento de estos modelos.

2.3.1.1.3.1 Ventajas

- Proyecto *Open Source,* lo que permite obtener actualizaciones continuas, documentación y soporte por un gran número de colaboradores pertenecientes a la comunidad de *Hugging Face*.
- Gran flexibilidad, ya que es posible tener un control total sobre la configuración, ajustes, parámetros y otras personalizaciones sin depender de terceros, además de poder utilizar una gran cantidad de modelos ofrecidos por la comunidad.
- Uso local, gracias a que es posible descargar los modelos en nuestro equipo desde el repositorio de *Hugging Face* y ejecutarlos en nuestro equipo sin la necesidad de depender de la nube de otro proveedor o de una conexión a internet.
- Uso gratuito, lo que nos permitirá poder generar el número de imágenes o con la mayor calidad posible que nos puedan ofrecer los recursos *hardware* que tengamos a nuestra disposición.

2.3.1.1.3.2 Desventajas

- Requiere de recursos de cómputo, lo cual puede provocar que el rendimiento en cuanto a tiempos de ejecución de los modelos y en cuanto al uso de memoria sean poco favorables si no se dispone del *hardware* adecuado.
- Mantenimiento de entorno, lo que puede provocar dificultades para desarrolladores con poca experiencia en entornos relacionados con la IA o con librerías de bajo nivel, ya

que para el uso de esta librería es necesario configurar un entorno para *Python* con una serie de librerías requeridas para su funcionamiento.

- Menor rendimiento que *APIs* avanzadas en la nube, debido a que empresas punteras en el mundo están desarrollando sus *APIs* y modelos con una gran competencia por la cuota de mercado, lo que hace que integren mejoras continuas tanto en su *hardware* como en el rendimiento de los resultados obtenidos por sus modelos, lo que provoca que sea difícil competir con estas *APIs* de pago ofrecidas por grandes empresas.

2.3.1.1.4 Keras CV Stable Diffussion

Esta librería [11] pertenece al proyecto *KerasCV*, el cuál proporciona implementaciones de visión artificial y entre uno de sus módulos se encuentra el conocido por *Stable Diffusion*, el cual permite realizar generaciones de imágenes a partir de descripciones textuales, para ello utiliza la biblioteca de *TensorFlow/Keras*, lo que permite a los desarrolladores ejecutar modelos de difusión dentro del entorno de *Keras*.

2.3.1.1.4.1 Ventajas

- Integración con entorno *TensorFlow/Keras*, lo cual es beneficioso ya que este entorno es el más utilizado en el ecosistema relacionado con la IA, por lo que los desarrolladores familiarizados con este entorno tendrán una gran ventaja respecto a la velocidad de aprendizaje y sin tener que aprender una nueva librería.
- *API* sencilla, ya que se hace uso de la *API* de *Keras* la cual tiene como objetivo ser intuitiva y de rápido aprendizaje para desarrolladores menos experimentados en este ámbito.
- Flexibilidad de ajustes, ya que al pertenecer al entorno *Keras* ofrece las capacidades de modificar capas, reentrenar el modelo y ajustar los híper parámetros en *TensorFlow*.

2.3.1.1.4.2 Desventajas

- Altos requisitos de *hardware*, debido a que se hace uso del modelo *Stable Diffussion*, que como otros modelos de difusión se requiere de un alto consumo de capacidad de cómputo y memoria, por lo que, si no se dispone de una *GPU* o una *TPU*, la ejecución del modelo de esta librería puede ser altamente lento o directamente inviable.
- Complejidad de instalación, debido a que es necesario configurar *TensorFlow* con *GPU* o *TPU*, por lo que los desarrolladores que no tengan experiencia con este tipo de tareas pueden encontrar altas dificultades.
- Mantenimiento, debido a que este módulo se encuentra en fases iniciales de desarrollo, por lo que los desarrolladores pueden encontrarse con la necesidad de realizar actualizaciones continuas debido a correcciones continuas de *bugs* o inclusión de nuevas mejoras, además de correr el riesgo de que el único modelo utilizado por esta librería de *Keras* quede obsoleto en un periodo corto de tiempo debido a la alta competencia en este ámbito.

2.3.1.2 Librerías de interfaz de usuario

A continuación, se tratarán las diferentes alternativas relacionadas con las librerías de interfaz gráfica de usuario.

2.3.1.2.1 tkinter

La librería *tkinter* [12] se trata de la librería de interfaz gráfica de usuario integrada en la biblioteca estándar de *Python*, la cual nos proporciona un conjunto de *widgets* y componentes para desarrollar aplicaciones de escritorio con interfaz gráfica de usuario de forma rápida y sencilla. Esta librería nos permitirá crear ventanas, botones, menús y la mayoría de los elementos necesarios para desarrollar interfaces gráficas.

2.3.1.2.1.1 Ventajas

- Está incluida en la biblioteca estándar de *Python*, por lo que no será necesario realizar instalaciones adicionales en nuestro entorno, lo que facilita la configuración del entorno y la distribución de la aplicación.
- Es multiplataforma, lo que nos permitirá crear aplicaciones para *Windows*, *MacOs* y diferentes distribuciones de *Linux*.
- Es muy simple, lo cual facilita la velocidad en el desarrollo y en el aprendizaje de las funcionalidades de la librería, gracias a esto con poco código podremos empezar a obtener resultados interesantes.
- Extensa documentación, debido a que lleva mucho tiempo en funcionamiento, por lo que existen multitud de ejemplos, foros de discusión para resolver problemas y tutoriales en línea para poder aprender a utilizar la librería o profundizar en más conocimientos.

2.3.1.2.1.2 Desventajas

- Diseño y temáticas anticuadas, debido a la longevidad de la librería y los rápidos avances y modas cambiantes en el diseño de interfaces gráficas el aspecto de las interfaces desarrolladas con esta librería puede lucir en la mayoría de los casos desactualizado o anticuado en comparación con las aplicaciones utilizadas hoy en día.
- Bajo rendimiento en aplicaciones complejas, ya que en aplicaciones con usos intensivos o con una gran carga de *widgets* o elementos gráficos la fluidez de la aplicación puede verse afectada considerablemente y repercutir en la usabilidad de esta.
- Funcionalidades limitadas, para el caso de que se quiera desarrollar una aplicación con animaciones o con *widgets* personalizados o más modernos esta librería se quedaría corta en funcionalidades ofrecidas.

2.3.1.2.2 ttkbootstrap

La librería *ttkbootstrap* [13] es una extensión del módulo *ttk* perteneciente a *Tkinter*, esta librería proporciona un aspecto y temáticas modernas a las aplicaciones desarrolladas con ella,

gracias que a través de una *API* similar a la ofrecida por *Tkinter*, proporciona *widgets* y componentes más modernos y componentes extra más avanzados en comparación a los estándar proporcionados por *tkinter*, de forma que permite crear aplicaciones mediante un *framework* similar al de *tkinter*, pero adaptando el estilo de las interfaces gráficas desarrolladas a los estándares modernos utilizados en las aplicaciones de hoy en día.

2.3.1.2.2.1 Ventajas

- Adaptación sencilla, gracias a que es una extensión de *Tkinter*, lo que permitirá desarrollar aplicaciones de forma que no sea necesario aprender un nuevo *framework* completamente diferente a *Tkinter*, el estándar proporcionado por *Python*.
- Aspecto y temática moderna, lo que permite lucir una interfaz con aspecto adaptado a las interfaces gráficas utilizadas hoy en día, de forma que las aplicaciones no lucirán desactualizadas de cara al usuario y proporcionarán una mayor satisfacción de uso a los usuarios finales.
- Es multiplataforma, por lo que permitirá desarrollar aplicaciones de escritorio para los principales sistemas operativos utilizados hoy en día.

2.3.1.2.2.2 Desventajas

- Poca documentación, debido a que no es una librería estándar y no tiene mucho recorrido, por lo que existe poca documentación y tutoriales en línea y una comunidad que no es muy extensa.
- Dependencia de *Tkinter*, ya que es una extensión de esta librería, por lo tanto, mantiene ciertas desventajas que nos encontramos con la librería estándar, como de rendimiento o difícil escalabilidad del código en aplicaciones complejas.

2.3.1.2.3 customtkinter

La librería *customtkinter* [14] se trata de una extensión de *Tkinter* pero que introduce sus *widgets* personalizados, ofreciendo a estos un aspecto moderno. Por lo que esta librería permite crear aplicaciones de interfaz gráfica de usuario modernas aportando temáticas oscuras y claras, esquinas redondeadas y estilos que mantienen la coherencia entre sí, pero sin perder los elementos base proporcionados por *Tkinter*.

2.3.1.2.3.1 Ventajas

- Profesionalidad de la interfaz, ya que los *widgets* mantienen un estilo coherente entre sí, además de un aspecto moderno con una fácil configuración, lo que mejorará la satisfacción del usuario al manejar la interfaz gráfica de la aplicación.
- Sencillez de la *API*, ya que está basada en la implementación de *Python* y *Tkinter*, lo que facilitará el uso y aprendizaje de los desarrolladores familiarizados con esta librería estándar de *Python*.
- Facilidad en la selección de la temática, debido a que con pocas líneas de código esta librería permite cambiar entre diferentes temáticas de entre las que se encuentran disponibles en ella.

2.3.1.2.3.2 Desventajas

- Capacidades limitadas, a causa de que no todos los *widgets* de la librería se encuentran estilizados por esta, por lo que existirán ciertos componentes de la librería con el aspecto estándar que proporciona *Tkinter* de base.
- Pequeña comunidad, debido a que esta librería es la que menor recorrido tiene de entre las mencionadas en esta sección, por lo que la documentación y soporte de la comunidad está poco extendida.
- Dependencia de *Tkinter*, por lo que mantendrá algunas de las desventajas estructurales que presenta *Tkinter*.
- Integración del *framework* complicada, lo cual puede ocurrir en caso de querer migrar nuestra aplicación desarrollada previamente con *Tkinter* debido a que será necesario adaptar la mayoría de *los widgets* pertenecientes a esta librería que no pertenecen al mismo *framework* que la librería estándar de *Python*.

Capítulo 3. Aspectos Teóricos

A lo largo de esta sección se detallan una serie de conceptos que son de importancia clave para lograr entender lo explicado a lo largo del presente documento y las características del proyecto a desarrollar.

Para ello, se exponen estos conceptos en las siguientes subsecciones en forma de una breve descripción que permitirá al lector comprender el trabajo realizado a lo largo del proyecto

3.1 Inteligencia Artificial

La inteligencia artificial también conocido por sus siglas *IA* o *AI* en inglés *Artificial Intelligence [15]*, se trata de una rama de la computación cuyo objetivo es el desarrollo de sistemas que permitan realizar tareas cuyo conocimiento para realizarlas se relaciona con el conocimiento humano, es decir, la capacidad de aprendizaje, de razonamiento o la capacidad de resolución de problemas.

Estos sistemas basan su implementación en una serie de modelos estadísticos y algoritmos que dan la posibilidad a los sistemas de conseguir analizar información, extraer una serie de patrones y conseguir tomar decisiones de forma prácticamente autónoma. Por lo que esta rama de la computación ofrece una amplia gama de posibilidades en cuanto a la capacidad de poder desarrollar sistemas capaces de resolver una serie de problemas de forma autónoma en una gran variedad de campos y sectores, desde sistemas de conducción autónoma, sistemas de visión artificial capaces de reconocer enfermedades o como es el caso de este proyecto sistemas capaces de generar nuevo contenido como imágenes.

3.2 Inteligencia Artificial Generativa

La inteligencia artificial generativa [16] surge recientemente como una nueva rama de la inteligencia artificial, esta rama emplea diferentes modelos entrenados con datos existentes previamente para poder reconocer patrones y a raíz de esto ser capaz de generar nuevo contenido no existente previamente. Esto permite que los programas de inteligencia artificial generativa puedan realizar una serie de tareas con creatividad propia como redactar textos, producir vídeos y música o como es el caso de este proyecto crear imágenes, entre otras muchas posibilidades. Conociendo esto, esta rama proporciona enormes posibilidades en los campos de la creatividad, desarrollo de software o automatización de tareas.

Esta rama de la IA, ha cogido relevancia recientemente, aunque su origen proviene de ciertas investigaciones realizadas a mediados del siglo XX, aunque esta relevancia empezó a coger peso a partir de la década del 2010 con el surgimiento de los modelos de *deep learning* conocidos como *GANs* [17] y los modelos conocidos como *Transformers* [18], que gracias a los avances obtenidos con este tipo de modelos se ha llegado a conseguir desarrollar sistemas que son capaces de reconocer patrones en datos con estructuras complejas como imágenes o vídeos y a

partir de estos poder llegar a generar nuevo contenido. Gracias a la existencia de este tipo de modelos, en este proyecto se puede lograr una solución cuyo objetivo es llegar a generar una serie de imágenes correspondientes a una descripción textual proporcionada por el usuario.

3.3 Redes neuronales

Las redes neuronales [19] son modelos de inteligencia artificial, los cuales están basados en el funcionamiento del cerebro humano, más en concreto inspirados en la manera en la que las neuronas del cerebro humano procesan la información. Por lo que este tipo de modelos implementan una representación de neuronas artificiales conectadas mediante diferentes capas interconectadas entre ellas, gracias a esto son capaces de aprender y reconocer patrones en datos con el objetivo de realizar tareas de predicción, clasificación o como en el caso de este proyecto de generar contenido.

Este tipo de modelos surgen a finales del siglo XX durante investigaciones realizadas sobre modelos matemáticos capaces de representar la forma en la que las neuronas humanas aprenden, no obstante, estos modelos no obtuvieron una gran relevancia hasta la década de 2010, donde los avances de hardware y la capacidad de procesar grandes cantidades de datos permitieron poder realizar grandes avances en la implementación de las redes neuronales.

Las redes neuronales tienen una gran importancia en el campo de la inteligencia artificial, debido a que son capaces de aprender representaciones de datos complejas y reconociendo relaciones en los datos de forma no lineal. Además, este tipo de modelos son la base de los modelos *Deep Learning* que se aplican en este proyecto, ya que este tipo de modelos se basan en redes neuronales con una gran cantidad de capas y gracias a esto son modelos con una gran versatilidad en cuanto a diferentes aplicaciones en multitud de campos, gracias a que son capaces de adaptarse a diferentes contextos siempre y cuando se disponga de la cantidad de datos suficientes para lograr entrenar los modelos.

3.4 Deep Learning

El campo de la Inteligencia artificial conocido como *Deep Learning o* aprendizaje profundo [20] en español*,* trata sobre el empleo de modelos de redes neuronales con un gran número de capas, de ahí su nombre aprendizaje profundo, cuyo objetivo es reconocer patrones en estructuras de datos complejas como imágenes o textos y aprender de estos patrones para conseguir resolver una serie de problemas de forma autónoma.

Su origen se remonta a la década de 1950 relacionado con investigaciones realizadas en los orígenes de la inteligencia artificial, con las investigaciones realizadas sobre los perceptrones [21] y comienza a conseguir una gran relevancia a partir de la década de 2010 debido a otra serie de avances que permiten darle un impulso a la implementación de este tipo de modelos, como por ejemplo los avances relacionados con la mejora del *hardware* existente o la disponibilidad de grandes volúmenes de datos necesarios para realizar el entrenamiento de este tipo de modelos.

Este campo tiene gran relevancia en cuanto a aplicaciones posibles, como por ejemplo en el área de visión artificial, procesamiento del lenguaje natural, reconocimiento de voz o sistemas de recomendación, por lo que para la realización de este proyecto los avances en este campo han sido esenciales para conseguir los objetivos buscados, gracias a los avances en el procesamiento del lenguaje natural o los avances relacionados con la visión artificial.

Por lo tanto, este campo tiene una gran importancia ya que este tipo de modelos, logran obtener resultados *state-of-the-art*, en múltiples problemas y sirve de base para la inteligencia artificial moderna, como, por ejemplo, aplicado a este proyecto, sirve de base para los sistemas de inteligencia artificial generativa y más en concreto para los modelos de inteligencia artificial de generación de imágenes.

3.5 Modelos de difusión

Este tipo de modelos pertenecen al campo de la inteligencia artificial generativa y su funcionamiento se basa en la simulación de procesos de difusión [22], de forma que son capaces de convertir ruido aleatorio en datos estructurados, como imágenes o audio. Este funcionamiento se resume en la existencia de una fase de difusión, que podría simplificarse como la fase en la que se añade el ruido y otra fase conocida como *denoising*, en la que el modelo va eliminando el ruido de forma gradual en una serie de pasos. Gracias a esto, este tipo de modelos son capaces de aprender a generar contenidos de una gran coherencia y calidad, como imágenes, vídeos o audios con un realismo tan elevado que puede ser difícil discernir contenido real del contenido generado por este tipo de modelos, por lo que este tipo de modelos hoy en día se tratan de los más avanzados y novedosos en el campo de la inteligencia artificial.

El origen de este tipo de modelos surge de la investigación en el campo de la física estadística, más concretamente en los campos del modelado estadístico y los procesos de difusión estocásticos, aplicados a campos concretos de la física, aunque a partir de la década de 2010 empiezan a cobrar relevancia en el campo de la inteligencia artificial, tras descubrir que estas técnicas aplicadas a este campo eran capaces de generar contenido de alta calidad, de forma que han acabado comiéndole el terreno a las *GANs*, dentro de la aplicación de la generación de contenido como imágenes.

Conociendo todo esto explicado previamente, este tipo de modelos, además de la aplicación relacionada con este proyecto que es la generación de imágenes a partir de una descripción textual, son capaces de ofrecer grandes soluciones en otros campos, como la restauración o edición de imágenes o la generación o la mejora de señales de audio y de vídeo. Por lo tanto, este tipo de modelos tienen una gran importancia dentro del área de la inteligencia artificial y hoy en día este tipo de modelos son los más sonados gracias a la gran calidad y realismo de los resultados obtenidos, son modelos menos inestables que sus alternativas, tienen una gran versatilidad dentro de su área de aplicación y la gran mayoría de los modelos de generación de imágenes de última generación más avanzados se basan en este enfoque, por lo que en este proyecto se ha optado en utilizar este tipo de modelos para conseguir resultados de alta calidad.

Capítulo 4. Planificación del Proyecto y Presupuesto Iniciales

En esta sección se muestra y describe la planificación inicial del proyecto a elaborar, así como el presupuesto inicial elaborado a partir de esta planificación, junto a la descripción de los elementos pertenecientes a este.

4.1 Planificación Inicial

A continuación, en esta sección se plantea la planificación inicial propuesta para la realización del proyecto, de forma que se puedan fijar las expectativas para la ejecución de este proyecto.

En esta planificación, se ha propuesto como fecha de inicio del proyecto el 03/06/2024 y como fecha de fin del proyecto se ha estimado el 25/10/2024, llegando a este resultado una vez estimada la duración de todas las actividades y tareas a realizar para completar el proyecto. En esta planificación se ha establecido una jornada laboral de 7 días, debido a que se han incluido en la jornada los días del fin de semana, además se ha establecido una duración de la jornada de 3 horas diarias, de forma que cada semana laboral contará con 21 horas de trabajo. También se debe tener en cuenta que se ha establecido aproximadamente todo el mes de agosto como no laborable, debido a las vacaciones de verano, concretamente se ha establecido como días no laborables desde el día 29/07/2024 hasta el día 01/09/2024, por lo tanto, durante ese periodo no se realizará ninguna hora de trabajo del proyecto.

Conociendo esta información, la duración del proyecto teniendo en cuenta la fecha de fin y de inicio y las vacaciones establecidas será de aproximadamente 4 meses, aunque realmente de trabajo serían 3 meses de forma también aproximada. Por lo tanto, una vez establecido todo lo expuesto previamente, se han estimado unas 335 horas de trabajo a dedicar para llevar a cabo la ejecución del proyecto.

A continuación, se expone un resumen de las actividades principales de la estructura de desglose de trabajo elaborada en la planificación inicial, junto a las horas de dedicación requeridas para realizar cada una de estas, que como se puede comprobar la suma de las horas de trabajo de cada tarea principal coincide con las horas totales de trabajo a dedicar para completar todo el proyecto.

EDT	Nombre de la tarea	Trabajo
1.1	Estudio inicial	87 horas
1.2	Documentación del proyecto	96,5 horas
1.3	Desarrollo del software	123 horas
1.4	Realización de las pruebas	16,5 horas

1.5	Cierre de proyecto	12 horas

Figura 4.1 Resumen de tareas principales del proyecto

4.1.1 Estudio Inicial

Las tareas a realizar durante el estudio inicial se corresponden con las relacionadas con la investigación previa a realizar sobre los diferentes elementos que debemos desarrollar durante el proyecto, de forma que se puedan adquirir los conocimientos necesarios para la posterior ejecución del proyecto.

En la siguiente tabla se expone un resumen de las tareas pertenecientes al estudio inicial, junto a la planificación temporal y las horas de dedicación de trabajo.

EDT	Nombre de tarea	Fecha comienzo	Fecha fin	Trabajo
1.1	Estudio inicial	Mon 03/06/24	Mon 01/07/24	87 horas
1.1.1	Planificación inicial	Mon 03/06/24	Tue 04/06/24	6 horas
1.1.2	Investigación inicial	Wed 05/06/24	Mon 01/07/24	81 horas
1.1.2.1	Investigación sistemas equivalentes	Wed 05/06/24	Fri 07/06/24	9 horas
1.1.2.2	Investigación apis de generación	Sat 08/06/24	Fri 21/06/24	42 horas
1.1.2.3	Investigación librerías IU	Sat 22/06/24	Mon 24/06/24	9 horas
1.1.2.4	Investigación arquitectura del sistema	Tue 25/06/24	Sun 30/06/24	16 horas
1.1.2.5	Lectura de trabajos de otros alumnos	Sun 30/06/24	Mon 01/07/24	5 horas

Figura 4.2 Resumen de tarea del estudio inicial

4.1.2 Documentación del proyecto

Este conjunto de tareas se centra en la realización de la actual documentación, en la cual se recopila toda la información necesaria para llevar a cabo la ejecución del proyecto.

EDT	Nombre tarea	Fecha inicio	Fecha fin	Trabajo
1.2	Documentación del proyecto	Tue 02/07/24	Mon 21/10/24	96,5 horas
1.2.1	Memoria del proyecto	Tue 02/07/24	Tue 02/07/24	1,5 horas
1.2.1.1	Resumen de la motivación, objetivos y alcance del proyecto	Tue 02/07/24	Tue 02/07/24	1 horas

1.2.1.2	Resumen de todos los aspectos	Tue 02/07/24	Tue 02/07/24	0,5 horas
1.2.2	Introducción	Tue 02/07/24	Thu 04/07/24	7,5 horas
1.2.2.1	Justificación del proyecto	Tue 02/07/24	Tue 02/07/24	0,5 horas
1.2.2.2	Objetivos del proyecto	Tue 02/07/24	Tue 02/07/24	1 horas
1.2.2.3	Estudio de la situación actual	Wed 03/07/24	Thu 04/07/24	6 horas
1.2.3	Aspectos teóricos	Fri 05/07/24	Fri 05/07/24	2 horas
1.2.4	Planificación inicial del proyecto	Fri 05/07/24	Mon 08/07/24	10 horas
1.2.5	Análisis	Mon 08/07/24	Wed 17/07/24	21 horas
1.2.5.1	Definición del sistema	Mon 08/07/24	Mon 08/07/24	1 horas
1.2.5.2	Requisitos del sistema	Mon 08/07/24	Tue 09/07/24	3 horas
1.2.5.3	Identificación de los subsistemas	Tue 09/07/24	Tue 09/07/24	0,5 horas
1.2.5.4	Diagrama de clases preliminar	Tue 09/07/24	Thu 11/07/24	6 horas
1.2.5.5	Análisis de casos de uso y escenarios	Thu 11/07/24	Sat 13/07/24	6 horas
1.2.5.6	Relación casos de uso y escenarios	Sat 13/07/24	Sat 13/07/24	0,5 horas
1.2.5.7	Análisis de interfaces de usuario	Sun 14/07/24	Mon 15/07/24	4 horas
1.2.5.8	Especificación del plan de pruebas	Mon 15/07/24	Wed 17/07/24	6 horas
1.2.6	Diseño	Wed 17/07/24	Fri 26/07/24	29 horas

1.2.6.1	Arquitectura del sistema	Wed 17/07/24	Fri 19/07/24	6 horas
1.2.6.2	Diseño de clases	Fri 19/07/24	Sat 20/07/24	4 horas
1.2.6.3	Diagramas de interacción y estados	Sat 20/07/24	Mon 22/07/24	5 horas
1.2.6.4	Diseño de la interfaz	Mon 22/07/24	Wed 24/07/24	6 horas
1.2.6.5	Especificación técnica del plan de pruebas	Wed 24/07/24	Fri 26/07/24	8 horas
1.2.7	Implementación del sistema	Fri 11/10/24	Sun 13/10/24	6,5 horas
1.2.7.1	Estándares y normas seguidos	Fri 11/10/24	Fri 11/10/24	0,5 horas
1.2.7.2	Lenguajes de programación	Fri 11/10/24	Fri 11/10/24	2 horas
1.2.7.3	Herramientas y programas usados	Fri 11/10/24	Sat 12/10/24	1 horas
1.2.7.4	Creación del sistema	Sat 12/10/24	Sun 13/10/24	3 horas
1.2.8	Desarrollo de las pruebas	Mon 14/10/24	Wed 16/10/24	4 horas
1.2.8.1	Pruebas unitarias	Mon 14/10/24	Mon 14/10/24	2 horas
1.2.8.2	Pruebas de integración y del sistema	Mon 14/10/24	Mon 14/10/24	0,5 horas
1.2.8.3	Pruebas de usabilidad y accesibilidad	Tue 15/10/24	Tue 15/10/24	1 horas
1.2.8.4	Pruebas de rendimiento	Wed 16/10/24	Wed 16/10/24	0,5 horas
1.2.9	Manuales del sistema	Fri 18/10/24	Sat 19/10/24	3 horas
1.2.9.1	Manual de instalación	Fri 18/10/24	Fri 18/10/24	0,5 horas

1.2.9.2	Manual de ejecución	Fri 18/10/24	Fri 18/10/24	0,5 horas
1.2.9.3	Manual de usuario	Sat 19/10/24	Sat 19/10/24	1 horas
1.2.9.4	Manual del programador	Sat 19/10/24	Sat 19/10/24	1 horas
1.2.10	Conclusiones y ampliaciones	Sat 19/10/24	Sat 19/10/24	1 horas
1.2.11	Planificación y presupuestos finales	Sun 20/10/24	Sun 20/10/24	3 horas
1.2.12	Referencias bibliográficas	Mon 21/10/24	Mon 21/10/24	0,5 horas
1.2.13	Apéndices	Mon 21/10/24	Mon 21/10/24	1,5 horas

Figura 4.3 Resumen de tareas de documentación del proyecto

4.1.3 Desarrollo del software

A continuación, se muestra el resumen de forma similar a las secciones previas de las tareas relacionadas con el desarrollo del software del proyecto, para llevar a cabo la implementación del sistema.

EDT	Nombre tarea	Fecha inicio	Fecha Fin	Trabajo
1.3	Desarrollo del software	Sat 27/07/24	Thu 10/10/24	123 horas
1.3.1	Interfaz de usuario	Sat 27/07/24	Fri 13/09/24	42 horas
1.3.2	Cargar modelo en memoria	Sat 14/09/24	Fri 20/09/24	21 horas
1.3.3	Generar imágenes	Sat 21/09/24	Fri 04/10/24	42 horas
1.3.4	Modificar opciones avanzadas	Sat 05/10/24	Mon 07/10/24	9 horas
1.3.5	Refinamientos finales	Tue 08/10/24	Thu 10/10/24	9 horas

Figura 4.4 Resumen de tareas de desarrollo del software

4.1.4 Realización de las pruebas

En esta sección se recopilan las tareas relacionadas con la realización e implementación de las diferentes pruebas a realizar en el sistema.

EDT	Nombre tarea	Fecha inicio	Fecha fin	Trabajo
1.4	Realización de las pruebas	Sun 13/10/24	Fri 1810/24	16,5 hrs
1.4.1	Pruebas unitarias	Sun 13/10/24	Mon 14/10/24	3 hrs
1.4.2	Pruebas de integración	Mon 14/10/24	Mon 14/10/24	0,5 hrs
1.4.3	Pruebas de usabilidad y accesibilidad	Mon 14/10/24	Tue 15/10/24	4 hrs
1.4.4	Pruebas de rendimiento	Tue 15/10/24	Wed 16/10/24	3 hrs
1.4.5	Corrección defectos encontrados	Wed 16/10/24	Fri 18/10/24	6 hrs

Figura 4.5 Resumen tareas de realización de las pruebas

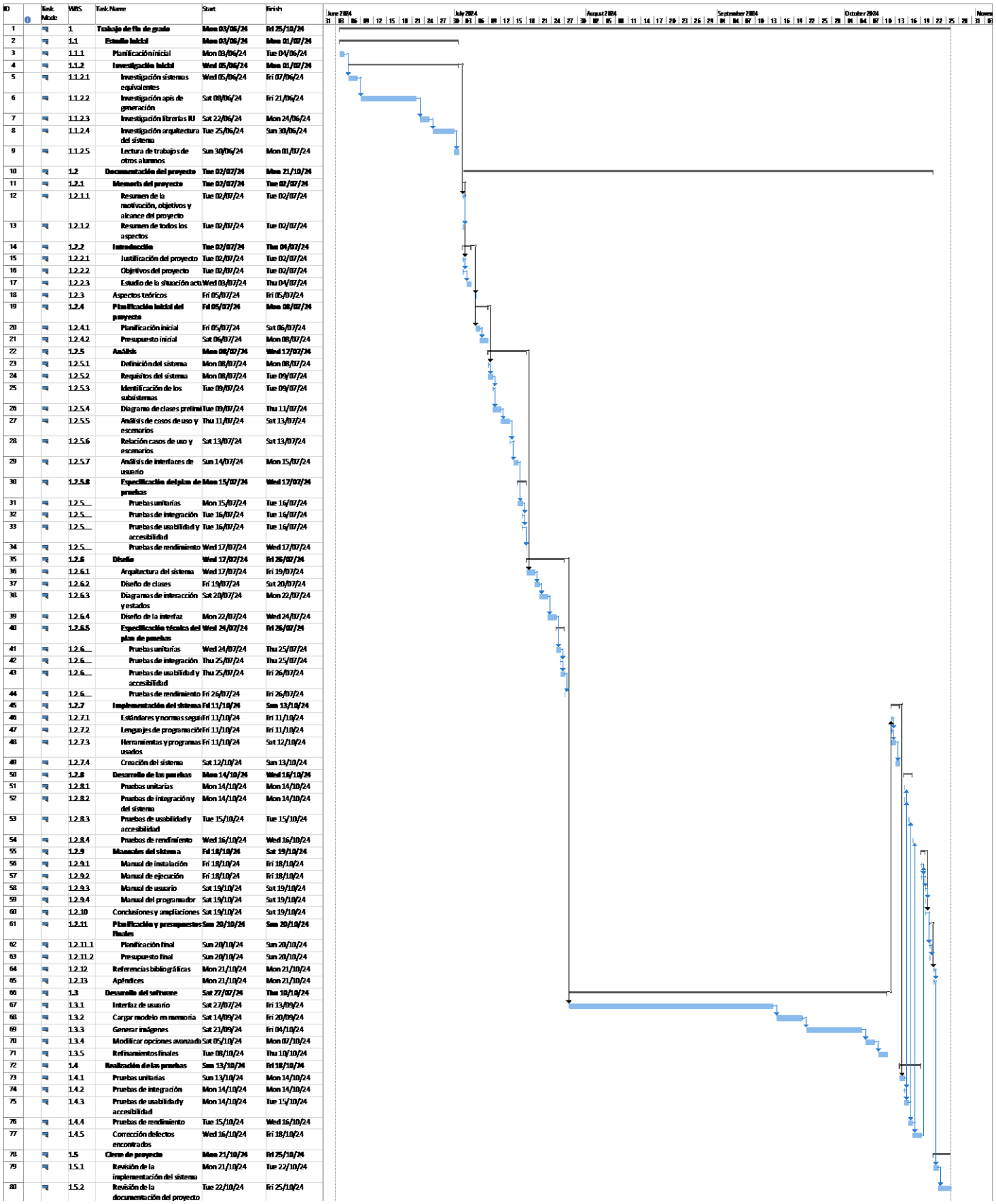
4.1.5 Cierre del proyecto

Para acabar, antes de realizar la entrega del trabajo actual, se deberá llevar a cabo una revisión en profundidad de los diferentes aspectos realizados durante el proyecto, entre ellos se encuentran las tareas expuestas a continuación.

EDT	Nombre tarea	Fecha inicio	Fecha fin	Trabajo
1.5	Cierre de proyecto	Mon 21/10/24	Fri 25/10/24	12 hrs
1.5.1	Revisión de la implementación del sistema	Mon 21/10/24	Tue 22/10/24	3 hrs
1.5.2	Revisión de la documentación del proyecto	Tue 22/10/24	Fri 25/10/24	9 hrs

Figura 4.6 Resumen tareas de cierre de proyecto

A continuación, se expone en la siguiente página el diagrama de *Gantt* completo del proyecto junto a todo el conjunto de tareas recogidas en este.



ID	Task Mode	WBS	Task Name	Start	Finish
1		1	**Trabajo de fin de grado**	**Mon 03/06/24**	**Fri 25/10/24**
2		1.1	**Estudio Inicial**	**Mon 03/06/24**	**Mon 01/07/24**
3		1.1.1	Planificación inicial	Mon 03/06/24	Tue 04/06/24
4		1.1.2	**Investigación Inicial**	**Wed 05/06/24**	**Mon 01/07/24**
5		1.1.2.1	Investigación sistemas equivalentes	Wed 05/06/24	Fri 07/06/24
6		1.1.2.2	Investigación apis de generación	Sat 08/06/24	Fri 21/06/24
7		1.1.2.3	Investigación librerías IU	Sat 22/06/24	Mon 24/06/24
8		1.1.2.4	Investigación arquitectura del sistema	Tue 25/06/24	Sun 30/06/24
9		1.1.2.5	Lectura de trabajos de otros alumnos	Sun 30/06/24	Mon 01/07/24
10		1.2	**Documentación del proyecto**	**Tue 02/07/24**	**Mon 21/10/24**
11		1.2.1	**Memoria del proyecto**	**Tue 02/07/24**	**Tue 02/07/24**
12		1.2.1.1	Resumen de la motivación, objetivos y alcance del proyecto	Tue 02/07/24	Tue 02/07/24
13		1.2.1.2	Resumen de todos los aspectos	Tue 02/07/24	Tue 02/07/24
14		1.2.2	**Introducción**	**Tue 02/07/24**	**Thu 04/07/24**
15		1.2.2.1	Justificación del proyecto	Tue 02/07/24	Tue 02/07/24
16		1.2.2.2	Objetivos del proyecto	Tue 02/07/24	Tue 02/07/24
17		1.2.2.3	Estudio de la situación actu	Wed 03/07/24	Thu 04/07/24
18		1.2.3	Aspectos teóricos	Fri 05/07/24	Fri 05/07/24
19		1.2.4	**Planificación inicial del proyecto**	**Fri 05/07/24**	**Mon 08/07/24**
20		1.2.4.1	Planificación inicial	Fri 05/07/24	Sat 06/07/24
21		1.2.4.2	Presupuesto inicial	Sat 06/07/24	Mon 08/07/24
22		1.2.5	**Análisis**	**Mon 08/07/24**	**Wed 17/07/24**
23		1.2.5.1	Definición del sistema	Mon 08/07/24	Mon 08/07/24
24		1.2.5.2	Requisitos del sistema	Mon 08/07/24	Tue 09/07/24
25		1.2.5.3	Identificación de los subsistemas	Tue 09/07/24	Tue 09/07/24
26		1.2.5.4	Diagrama de clases prelimi	Tue 09/07/24	Thu 11/07/24
27		1.2.5.5	Análisis de casos de uso y escenarios	Thu 11/07/24	Sat 13/07/24
28		1.2.5.6	Relación casos de uso y escenarios	Sat 13/07/24	Sat 13/07/24
29		1.2.5.7	Análisis de interfaces de usuario	Sun 14/07/24	Mon 15/07/24
30		1.2.5.8	**Especificación del plan de pruebas**	**Mon 15/07/24**	**Wed 17/07/24**
31		1.2.5_	Pruebas unitarias	Mon 15/07/24	Tue 16/07/24
32		1.2.5_	Pruebas de integración	Tue 16/07/24	Tue 16/07/24
33		1.2.5_	Pruebas de usabilidad y accesibilidad	Tue 16/07/24	Tue 16/07/24
34		1.2.5_	Pruebas de rendimiento	Wed 17/07/24	Wed 17/07/24
35		1.2.6	**Diseño**	**Wed 17/07/24**	**Fri 26/07/24**
36		1.2.6.1	Arquitectura del sistema	Wed 17/07/24	Fri 19/07/24
37		1.2.6.2	Diseño de clases	Fri 19/07/24	Sat 20/07/24
38		1.2.6.3	Diagramas de interacción y estados	Sat 20/07/24	Mon 22/07/24
39		1.2.6.4	Diseño de la interfaz	Mon 22/07/24	Wed 24/07/24
40		1.2.6.5	**Especificación técnica del plan de pruebas**	**Wed 24/07/24**	**Fri 26/07/24**
41		1.2.6_	Pruebas unitarias	Wed 24/07/24	Thu 25/07/24
42		1.2.6_	Pruebas de integración	Thu 25/07/24	Thu 25/07/24
43		1.2.6_	Pruebas de usabilidad y accesibilidad	Thu 25/07/24	Fri 26/07/24
44		1.2.6_	Pruebas de rendimiento	Fri 26/07/24	Fri 26/07/24
45		1.2.7	**Implementación del sistema**	**Fri 11/10/24**	**Sun 13/10/24**
46		1.2.7.1	Estándares y normas seguid	Fri 11/10/24	Fri 11/10/24
47		1.2.7.2	Lenguajes de programación	Fri 11/10/24	Fri 11/10/24
48		1.2.7.3	Herramientas y programas usados	Fri 11/10/24	Sat 12/10/24
49		1.2.7.4	Creación del sistema	Sat 12/10/24	Sun 13/10/24
50		1.2.8	**Desarrollo de las pruebas**	**Mon 14/10/24**	**Wed 16/10/24**
51		1.2.8.1	Pruebas unitarias	Mon 14/10/24	Mon 14/10/24
52		1.2.8.2	Pruebas de integración y del sistema	Mon 14/10/24	Mon 14/10/24
53		1.2.8.3	Pruebas de usabilidad y accesibilidad	Tue 15/10/24	Tue 15/10/24
54		1.2.8.4	Pruebas de rendimiento	Wed 16/10/24	Wed 16/10/24
55		1.2.9	**Manuales del sistema**	**Fri 18/10/24**	**Sat 19/10/24**
56		1.2.9.1	Manual de instalación	Fri 18/10/24	Fri 18/10/24
57		1.2.9.2	Manual de ejecución	Fri 18/10/24	Fri 18/10/24
58		1.2.9.3	Manual de usuario	Sat 19/10/24	Sat 19/10/24
59		1.2.9.4	Manual del programador	Sat 19/10/24	Sat 19/10/24
60		1.2.10	Conclusiones y ampliaciones	Sat 19/10/24	Sat 19/10/24
61		1.2.11	**Planificación y presupuestos finales**	**Sun 20/10/24**	**Sun 20/10/24**
62		1.2.11.1	Planificación final	Sun 20/10/24	Sun 20/10/24
63		1.2.11.2	Presupuesto final	Sun 20/10/24	Sun 20/10/24
64		1.2.12	Referencias bibliográficas	Mon 21/10/24	Mon 21/10/24
65		1.2.13	Apéndices	Mon 21/10/24	Mon 21/10/24
66		1.3	**Desarrollo del software**	**Sat 27/07/24**	**Thu 10/10/24**
67		1.3.1	Interfaz de usuario	Sat 27/07/24	Fri 13/09/24
68		1.3.2	Cargar modelo en memoria	Sat 14/09/24	Fri 20/09/24
69		1.3.3	Generar imágenes	Sat 21/09/24	Fri 04/10/24
70		1.3.4	Modificar opciones avanzada	Sat 05/10/24	Mon 07/10/24
71		1.3.5	Refinamientos finales	Tue 08/10/24	Thu 10/10/24
72		1.4	**Realización de las pruebas**	**Sun 13/10/24**	**Fri 18/10/24**
73		1.4.1	Pruebas unitarias	Sun 13/10/24	Mon 14/10/24
74		1.4.2	Pruebas de integración	Mon 14/10/24	Mon 14/10/24
75		1.4.3	Pruebas de usabilidad y accesibilidad	Mon 14/10/24	Tue 15/10/24
76		1.4.4	Pruebas de rendimiento	Tue 15/10/24	Wed 16/10/24
77		1.4.5	Corrección defectos encontrados	Wed 16/10/24	Fri 18/10/24
78		1.5	**Cierre de proyecto**	**Mon 21/10/24**	**Fri 25/10/24**
79		1.5.1	Revisión de la implementación del sistema	Mon 21/10/24	Tue 22/10/24
80		1.5.2	Revisión de la documentación del proyecto	Tue 22/10/24	Fri 25/10/24

Figura 4.7 Diagrama de Gantt de la planificación inicial

4.2 Presupuesto Inicial

A continuación, una vez obtenida la planificación inicial, se expone de forma detallada el presupuesto inicial elaborado para la ejecución del proyecto. Para ello se ha simulado una definición de empresa en la cual se han establecido diferentes roles para representar los diferentes recursos de trabajo que ejecutarán las diferentes tareas, realizadas por el autor del presente trabajo y del director de este.

En las siguientes subsecciones se describe esta definición de empresa, así como las diferentes partidas y elementos elaborados para obtener el presupuesto inicial del proyecto.

4.2.1 Definición de empresa

A continuación, se describirá la definición de empresa encargada de la ejecución del proyecto, para ello se han propuesto seis roles: Jefe de proyecto, Jefe de equipo, Ingeniero de software, Desarrollador Python, QA Tester y Analista de software. Estos roles serán simulados por el mismo autor del proyecto presente en función de la tarea del proyecto a realizar, salvo el rol de Jefe de proyecto que se corresponderá con el director de este mismo proyecto.

En las figuras expuestas a continuación, se describirá la definición de empresa elaborada para la ejecución del proyecto, en las que se recogerán los elementos necesarios para obtener esta definición y necesarios también para elaborar los elementos requeridos posteriormente para realizar el presupuesto inicial. Entre ellos se encuentran los salarios establecidos para cada recurso de la empresa, así como su productividad y costes directos e indirectos asociado a cada uno de ellos.

Personal	Sueldo Bruto	Coste Salarial Año
Jefe de proyecto	50.000,00 €	65.000,00 €
Jefe de equipo	40.000,00 €	52.000,00 €
Ingeniero de software	35.000,00 €	45.500,00 €
Desarrollador Python	25.000,00 €	32.500,00 €
QA tester	27.000,00 €	35.100,00 €
Analista de software	30.000,00 €	39.000,00 €
Total	207.000,00 €	269.100,00 €

Figura 4.8 Personal y coste anual de la empresa

Personal	TOTAL	Productividad	Coste directo	CI %	Coste indirecto
Jefe de proyecto	65.000,00 €	0%	- €	100%	65.000,00 €
Jefe de equipo	52.000,00 €	50%	26.000,00 €	50%	26.000,00 €
Ingeniero de software	45.500,00 €	85%	38.675,00 €	15%	6.825,00 €
Desarrollador Python	32.500,00 €	95%	30.875,00 €	5%	1.625,00 €
QA tester	35.100,00 €	90%	31.590,00 €	10%	3.510,00 €
Analista de software	39.000,00 €	80%	31.200,00 €	20%	7.800,00 €
Total	269.100,00 €		158.340,00 €		110.760,00 €

Figura 4.9 Costes directos e indirectos del personal

Personal	Productividad	Horas/año	Horas productivas/año	Horas productivas
Jefe de proyecto	0%			
Jefe de equipo	50%	1787	893,5	893,5
Ingeniero de software	85%	1787	1518,95	1518,95
Desarrollador Python	95%	1787	1697,65	1697,65
QA tester	90%	1787	1608,3	1608,3
Analista de software	80%	1787	1429,6	1429,6
			Total	5718,4

Figura 4.10 Productividad del personal

A continuación, se exponen los costes indirectos asociados a la empresa, relacionados con los costes en los que debe incurrir la empresa por el simple hecho de mantener su operación activa. Además, se muestran los medios de producción establecidos para llevar a cabo la ejecución del proyecto.

Servicio	Coste mes	Coste año
Limpieza	500,00 €	6.000,00 €
Asesoría	200,00 €	2.400,00 €
Tributos y tasas diversas	100,00 €	1.200,00 €
Alquileres o arrendamientos de inmuebles	800,00 €	9.600,00 €
Consumo eléctrico	80,00 €	960,00 €
Consumo calefacción	40,00 €	480,00 €
Consumo agua	20,00 €	240,00 €
Gastos en comunicaciones	120,00 €	1.440,00 €
Gastos de viaje, desplazamiento, manutención y estancias de personal no productivo	100,00 €	1.200,00 €
Gastos en material de oficina	45,00 €	540,00 €
Gastos financieros	60,00 €	720,00 €
Total	2.065,00 €	24.780,00 €

Figura 4.11 Costes indirectos de la empresa

A continuación, en la siguiente tabla podemos observar los medios de producción definidos para la empresa que hemos elaborado para la realización del proyecto, como se ha mencionado previamente se han propuesto diferentes roles que serán simulados por el autor del proyecto, por lo que, partiendo de esta simulación, en la siguiente tabla se reflejarán los medios de producción utilizados por cada rol simulado en la empresa, es decir, simularemos que la empresa tiene varios trabajadores y por lo tanto necesitarán cada uno su material como licencias y equipos, lo cual podemos verlo reflejado en la columna unidades, por ejemplo necesitaremos 5 *PC Lenovo ThinkPad* (1 unidad por cada trabajador que requiera el material).

Equipo/Licencia	Unidades	Precio	Coste total	Coste año	Tipo	Plazo
Licencia Windows 10 Professional	5	30,00 €	150,00 €	150,00 €	Alquiler	-
Licencia Office 365	5	28,90 €	144,50 €	144,50 €	Alquiler	-
PC Lenovo ThinkPad	5	1.500,00 €	7.500,00 €	1.875,00 €	Amortización	4

Monitor Lenovo ThinkVision	**5**	**165,00 €**	**825,00 €**	**206,25 €**	**Amortización**	**4**
Licencia MS Project	**2**	**20,00 €**	**40,00 €**	**40,00 €**	**Alquiler**	**-**
Visual Studio Code	**4**	**0,00 €**	**0,00 €**	**0,00 €**	**Alquiler**	
Total				**2.415,75 €**		

Figura 4.12 Medios de producción

A continuación, en la figura 4.13, se describen los precios por hora establecidos para cada recurso de la empresa, tanto como a precios de coste como costes con beneficios incluidos, estos últimos serán los utilizados para realizar la facturación por horas al cliente, mientras que los primeros serán utilizados para calcular las partidas de costes de la empresa.

Para obtener los precios de coste por hora, se han realizado calculando los costes directos por hora del recurso, los cuales se calculan a partir de las horas facturables del recurso y de los costes directos de este. Una vez obtenidos estos costes directos por hora, se promediarán con los costes indirectos totales de la empresa, de forma que todos estos costes indirectos podrán ser cubiertos por los precios/hora (Sin beneficios) establecidos para cada recurso.

En cuanto a los precios/hora con beneficios, utilizados para ser facturados por el trabajo realizado para el cliente, se calcularán tomando como base los precios explicados previamente y aplicándoles un incremento del 25%, correspondiente con los beneficios deseados y además se le aplicará un incremento adicional del 4%, utilizado para poder tener cierto margen ante imprevistos que puedan surgir durante la ejecución del proyecto.

Personal	Precio / Hora	Horas productivas	Facturación	Precio/hora (Sin beneficios)
Jefe de proyecto		**0,00**		
Jefe de equipo	**70,24 €**	**893,50**	**62.762,15 €**	**54,45 €**
Ingeniero de software	**61,46 €**	**1.518,95**	**93.358,70 €**	**47,65 €**
Desarrollador Python	**43,90 €**	**1.697,65**	**74.530,06 €**	**34,03 €**
QA tester	**47,41 €**	**1.608,30**	**76.256,02 €**	**36,76 €**
Analista de software	**52,68 €**	**1.429,60**	**75.314,58 €**	**40,84 €**
Total		**7.148,00**	**382.221,52 €**	

Figura 4.13 Precios/hora del personal

Para acabar, en la siguiente figura se muestra un resumen de la definición de empresa elaborada para el proyecto.

Nº	Concepto	Importe
1	**Total de costes directos**	**158.340,00 €**
2	**Total de costes indirectos**	**137.955,75 €**
3	**Suma de costes directos e indirectos**	**296.295,75 €**
4	**Beneficio deseado (25%)**	**74.073,94 €**

5	**Coste total (Suma de costes directos, indirectos y beneficios)**	**370.369,69 €**
6	**Facturación posible en función de las horas de producción y precios/hora calculados**	**382.221,52 €**
7	**Margen entre el coste total y la facturación (relación entre 5 y 6)**	**3,10%**

Figura 4.14 Resumen de la empresa definida

4.2.2 Presupuesto de costes inicial

En esta sección se describe el presupuesto de costes inicial obtenido para la ejecución del proyecto, donde primeramente se mostrará el presupuesto de costes desarrollado con las diferentes partidas desglosadas y finalmente se mostrará al final de la subsección el resumen del presupuesto de costes inicial obtenido.

Para realizar el presupuesto de costes inicial, se ha dividido primeramente en varias partidas:

- Estudio inicial.
    - Se tratará de un conjunto de tareas relacionadas con labores de investigación previa a la ejecución del proyecto, por lo que se considerarán horas no facturables que deberemos promediar en el presupuesto de cliente más adelante, por lo que esta partida no será incluida en el presupuesto del cliente.
- Documentación del proyecto.
    - Todo el conjunto de tareas relacionadas con la elaboración de la presente documentación del proyecto.
- Desarrollo del software.
    - Las tareas que se realizarán para llevar a cabo la implementación del sistema.
- Realización de las pruebas.
    - Se tratará del conjunto de tareas relacionadas con la ejecución de las diferentes pruebas a realizar en el sistema.
- Cierre del proyecto.
    - Conjunto de tareas relacionadas con la revisión de los trabajos realizados durante la ejecución del proyecto, las cuáles serán realizadas por el Jefe de proyecto, que en este caso no tiene un coste asociado por hora debido a su productividad nula, por lo que esta partida tampoco será incluida en el presupuesto del cliente.

A continuación, se muestran las diferentes partidas desarrolladas para el presupuesto de costes inicial.

Estudio inicial													
l1	l2	l3	l4	l5	Descripción	Cantidad	Unidades	Precio	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1					**Estudio inicial**								**4.737,15 €**
	1				**Planificación inicial**							**326,70 €**	
		1			Jefe de equipo	6	Horas	54,45 €					
	2				**Investigación inicial**							**4.410,45 €**	
		1			**Investigación sistemas equivalentes**						490,05 €		
			1		Jefe de equipo	9	Horas	54,45 €					
		2			**Investigación apis de generación**						2.286,90 €		
			1		Jefe de equipo	42	Horas	54,45 €					
		3			**Investigación librerías IU**						490,05 €		
			1		Jefe de equipo	9	Horas	54,45 €					
		4			**Investigación arquitectura del sistema**						871,20 €		
			1		Jefe de equipo	16	Horas	54,45 €					
		5			**Lectura de trabajos de otros alumnos**						272,25 €		
			1		Jefe de equipo	5	Horas	54,45 €					

Figura 4.15 Partida de estudio inicial

Documentación del proyecto															
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Documentación del proyecto**									**4.422,37 €**
	1					**Memoria del proyecto**								81,68 €	
		1				**Resumen de la motivación, objetivos y alcance del proyecto**							54,45 €		
			1			Jefe de equipo	1	Horas	54,45 €						
		2				**Resumen de todos los aspectos**									
			1			Jefe de equipo	0,5	Horas	54,45 €				27,23 €		
	2					**Introducción**								408,38 €	
		1				**Justificación del proyecto**							27,23 €		
			1			Jefe de equipo	0,5	Horas	54,45 €						
		2				**Objetivos del proyecto**							54,45 €		
			1			Jefe de equipo	1	Horas	54,45 €						
		3				**Estudio de la situación actual**							326,70 €		
			1			Jefe de equipo	6	Horas	54,45 €						
	3					**Aspectos teóricos**								108,90 €	
			1			Jefe de equipo	2	Horas	54,45 €						
	4					**Planificación inicial del proyecto**								544,50 €	
		1				**Planificación inicial**							217,80 €		
			1			Jefe de equipo	4	Horas	54,45 €						
		2				**Presupuesto inicial**							326,70 €		
			1			Jefe de equipo	6	Horas	54,45 €						
	5					**Análisis**								1.078,20 €	
		1				**Definición del sistema**							40,84 €		
			1			Analista Software	1	Horas	40,84 €						
		2				**Requisitos del sistema**							122,52 €		
			1			Analista Software	3	Horas	40,84 €						
		3				**Identificación de los subsistemas**							20,42 €		
			1			Analista Software	0,5	Horas	40,84 €						
		4				**Diagrama de clases preliminar**							245,04 €		
			1			Analista Software	6	Horas	40,84 €						
		5				**Análisis de casos de uso y escenarios**							245,04 €		
			1			Analista Software	6	Horas	40,84 €						
		6				**Relación casos de uso y escenarios**							20,42 €		
			1			Analista Software	0,5	Horas	40,84 €						
		7				**Análisis de interfaces de usuario**							163,36 €		
			1			Analista Software	4	Horas	40,84 €						
		8				**Especificación del plan de pruebas**							220,56 €		
			1			**Pruebas unitarias**						110,28 €			
				1		QA Tester	3	Horas	36,76 €						
			2			**Pruebas de integración**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
			3			**Pruebas de usabilidad y accesibilidad**						55,14 €			
				1		**QA Tester**	1,5	Horas	36,76 €						
			4			**Pruebas de rendimiento**						36,76 €			
				1		QA Tester	1	Horas	36,76 €						
	6					**Diseño**								1.294,73 €	
		1				**Arquitectura del sistema**							285,90 €		
			1			Ingeniero de software	6	Horas	47,65 €						
		2				**Diseño de clases**							190,60 €		
			1			Ingeniero de software	4	Horas	47,65 €						
		3				**Diagramas de interacción y estados**							238,25 €		
			1			Ingeniero de software	5	Horas	47,65 €						
		4				**Diseño de la interfaz**							285,90 €		
			1			Ingeniero de software	6	Horas	47,65 €						
		5				**Especificación técnica del plan de pruebas**							294,08 €		
			1			**Pruebas unitarias**						147,04 €			
				1		QA Tester	4	Horas	36,76 €						
			2			**Pruebas de integración**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
			3			**Pruebas de usabilidad y accesibilidad**						73,52 €			
				1		QA Tester	2	Horas	36,76 €						
			4			**Pruebas de rendimiento**						55,14 €			
				1		QA Tester	1,5	Horas	36,76 €						
	7					**Implementación del sistema**								309,73 €	
		1				**Estándares y normas seguidos**							23,83 €		
			1			Ingeniero de software	0,5	Horas	47,65 €						
		2				**Lenguajes de programación**							95,30 €		
			1			Ingeniero de software	2	Horas	47,65 €						
		3				**Herramientas y programas usados**							47,65 €		
			1			Ingeniero de software	1	Horas	47,65 €						
		4				**Creación del sistema**							142,95 €		
			1			Ingeniero de software	3	Horas	47,65 €						
	8					**Desarrollo de las pruebas**								147,04 €	
		1				**Pruebas unitarias**							73,52 €		
			1			**QA Tester**	2	Horas	36,76 €						
		2				**Pruebas de integración y del sistema**							18,38 €		
			1			QA Tester	0,5	Horas	36,76 €						
		3				**Pruebas de usabilidad y accesibilidad**							36,76 €		
			1			QA Tester	1	Horas	36,76 €						
		4				**Pruebas de rendimiento**							18,38 €		
			1			QA Tester	0,5	Horas	36,76 €						
	9					**Manuales del sistema**								122,52 €	
		1				**Manual de instalación**							23,83 €		
			1			Ingeniero de software	0,5	Horas	47,65 €						
		2				**Manual de ejecución**							23,83 €		
			1			Ingeniero de software	0,5	Horas	47,65 €						
		3				**Manual de usuario**							40,84 €		
			1			Analista Software	1	Horas	40,84 €						
		4				**Manual del programador**							34,03 €		
			1			Desarrollador python	1	Horas	34,03 €						
	10					**Conclusiones y ampliaciones**								54,45 €	
			1			Jefe de equipo	1	Horas	54,45 €						
	11					**Planificación y presupuestos finales**								163,35 €	
		1				**Planificación final**							54,45 €		
			1			Jefe de equipo	1	Horas	54,45 €						
		2				**Presupuesto final**							108,90 €		
			1			Jefe de equipo	2	Horas	54,45 €						
	12					**Referencias bibliográficas**								27,23 €	
		1				Jefe de equipo	0,5	Horas	54,45 €						
	13					**Apéndices**								81,68 €	
		1				Jefe de equipo	1,5	Horas	54,45 €						

Figura 4.16 Partida de documentación

Desarrollo del software														
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Desarrollo del software**									**4.185,69 €**
	1					**Interfaz de usuario**								1.429,26 €	
						Desarrollador Python	42	Horas	34,03 €						
	2					**Cargar modelo en memoria**								714,63 €	
						Desarrollador Python	21	Horas	34,03 €						
	3					**Generar imágenes**								1.429,26 €	
						Desarrollador Python	42	Horas	34,03 €						
	4					**Modificar opciones avanzadas**								306,27 €	
						Desarrollador Python	9	Horas	34,03 €						
	5					**Refinamientos finales**								306,27 €	
						Desarrollador Python	9	Horas	34,03 €						

Figura 4.17 Partida de desarrollo del software

Realización de las pruebas															
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Realización de las pruebas**									**590,16 €**
	1					**Pruebas unitarias**								110,28 €	
		1				QA Tester	3	Horas	36,76 €						
	2					**Pruebas de integración**								18,38 €	
		1				QA Tester	0,5	Horas	36,76 €						
	3					**Pruebas de usabilidad y accesibilidad**								147,04 €	
		1				QA Tester	4	Horas	36,76 €						
	4					**Pruebas de rendimiento**								110,28 €	
		1				QA Tester	3	Horas	36,76 €						
	5					**Corrección defectos encontrados**								204,18 €	
		1				Desarrollador Python	6	Horas	34,03 €						

Figura 4.18 Partida de realización de las pruebas

Cierre del proyecto															
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Cierre del proyecto**									**0,00 €**
	1					**Revisión de la implementación del sistema**								0,00 €	
		1				Jefe de proyecto	3	Horas	0,00 €						
	2					**Revisión de la documentación del proyecto**								0,00 €	
		1				Jefe de proyecto	9	Horas	0,00 €						

Figura 4.19 Partida de cierre de proyecto

Para acabar podremos ver el presupuesto de costes inicial con todas las partidas agregadas, así como el resumen del presupuesto de costes inicial de la empresa.

Presupuesto de costes				
Código	Item	Partida	Importe	Total
1		Estudio inicial		4.737,15 €
	1	Planificación inicial	326,70 €	
	2	Investigación inicial	4.410,45 €	
2		Documentación del proyecto		4.422,37 €
	1	Memoria del proyecto	81,68 €	
	2	Introducción	408,38 €	
	3	Aspectos teóricos	108,90 €	
	4	Planificación inicial del proyecto	544,50 €	
	5	Análisis	1.078,20 €	
	6	Diseño	1.294,73 €	
	7	Implementación del sistema	309,73 €	
	8	Desarrollo de las pruebas	147,04 €	
	9	Manuales del sistema	122,52 €	
	10	Conclusiones y ampliaciones	54,45 €	
	11	Planificación y presupuestos finales	163,35 €	
	12	Referencias bibliográficas	27,23 €	
	13	Apéndices	81,68 €	
3		Desarrollo del software		4.185,69 €
	1	Interfaz de usuario	1.429,26 €	
	2	Cargar modelo en memoria	714,63 €	
	3	Generar imágenes	1.429,26 €	
	4	Modificar opciones avanzadas	306,27 €	
	5	Refinamientos finales	306,27 €	
4		Realización de las pruebas		590,16 €
	1	Pruebas unitarias	110,28 €	
	2	Pruebas de integración	18,38 €	
	3	Pruebas de usabilidad y accesibilidad	147,04 €	
	4	Pruebas de rendimiento	110,28 €	
	5	Corrección defectos encontrados	204,18 €	
5		Cierre del proyecto		0,00 €
	1	Revisión de la implementación del sistema	0,00 €	
	2	Revisión de la documentación del proyecto	0,00 €	
Total costes				13.935,37 €

Figura 4.20 Presupuesto de costes inicial agregado

Resumen presupuesto de costes		
Cod	Partida	Total
1	Estudio inicial	4.737,15 €
2	Documentación del proyecto	4.422,37 €
3	Desarrollo del software	4.185,69 €
4	Realización de las pruebas	590,16 €
5	Cierre del proyecto	€ -
Total Coste		13.935,37 €

Figura 4.21 Resumen del presupuesto de costes

4.2.3 Presupuesto de cliente inicial

A continuación, se detalla el presupuesto inicial elaborado para el cliente con las partidas correspondientes para este. Como se puede observar las partidas del presupuesto del cliente difieren de las mostradas en el presupuesto de costes, más concretamente no se ha incluido la partida de estudio inicial ni la de cierre de proyecto. Esto es debido a que estas partidas se han llegado a la conclusión de que no pertenecen al conjunto de partidas cuyo trabajo es facturable al cliente de forma directa, en concreto la de cierre de proyecto porque el jefe de proyecto tiene una productividad del 0% y la de estudios iniciales porque se ha concebido de forma que contiene tareas previas a la ejecución del proyecto que no se han contemplado para ser facturables de forma directa al cliente.

Una vez asumida esta información, el valor económico de las partidas mencionadas como no facturables se debe promediar entre las partidas que se facturarán al cliente, es decir las que se muestran en la siguiente figura. Teniendo en cuenta esto y que debemos añadir un 25% de beneficios sobre el total de las partidas de costes, se obtiene un valor a promediar entre las partidas facturables de 8220,99 €. Por lo tanto, se ha obtenido el valor porcentual que se debe incrementar cada partida de costes, para obtener el valor de cada partida del cliente. Este valor se ha obtenido calculando la relación entre el valor de las partidas que se deben promediar entre el valor de las partidas a las que aplicaremos este incremento, en este caso con un valor de 9198,22€. Tras obtener este valor se han incrementado las partidas a presentar al cliente, obtenidas en el presupuesto de costes un 89,38%, obteniendo tras aplicar este incremento, el valor final de las partidas que se presentarán en el presupuesto inicial del cliente, detalladas en la siguiente figura de forma desglosada y en la siguiente a modo de resumen.

Presupuesto de cliente				
Código	Item	Partida	Importe	Total
1		Documentación del proyecto		8.374,90 €
	1	Memoria del proyecto	154,67 €	
	2	Introducción	773,36 €	
	3	Aspectos teóricos	206,23 €	
	4	Planificación inicial del proyecto	1.031,15 €	
	5	Análisis	2.041,85 €	
	6	Diseño	2.451,91 €	
	7	Implementación del sistema	586,54 €	
	8	Desarrollo de las pruebas	278,46 €	
	9	Manuales del sistema	232,02 €	
	10	Conclusiones y ampliaciones	103,12 €	
	11	Planificación y presupuestos finales	309,35 €	
	12	Referencias bibliográficas	51,56 €	
	13	Apéndices	154,67 €	
2		Desarrollo del software		7.926,69 €
	1	Interfaz de usuario	2.706,67 €	
	2	Cargar modelo en memoria	1.353,34 €	
	3	Generar imágenes	2.706,67 €	
	4	Modificar opciones avanzadas	580,00 €	
	5	Refinamientos finales	580,00 €	
3		Realización de las pruebas		1.117,62 €
	1	Pruebas unitarias	208,84 €	
	2	Pruebas de integración	34,81 €	
	3	Pruebas de usabilidad y accesibilidad	278,46 €	
	4	Pruebas de rendimiento	208,84 €	
	5	Corrección defectos encontrados	386,67 €	
Total cliente antes de impuestos				17.419,21 €
IVA				21,00%
Total cliente				21.077,24 €

Figura 4.22 Presupuesto de cliente inicial

Resumen presupuesto de cliente		
Cod	Partida	Total
1	Documentación del proyecto	8.374,90 €
2	Desarrollo del software	7.926,69 €
3	Realización de las pruebas	1.117,62 €
Total cliente antes de impuestos		17.419,21 €
Total cliente después de impuestos (21%)		21.077,24 €

Figura 4.23 Presupuesto de cliente resumido

Capítulo 5. Análisis

Este apartado contiene toda la especificación de requisitos y toda la documentación del análisis de la aplicación, a partir de la cual se elaborará posteriormente el diseño.

5.1 Definición del Sistema

5.1.1 Determinación del Alcance del Sistema

El sistema a desarrollar se trata de una herramienta que permitirá al usuario generar un número de imágenes determinado, mediante diferentes modelos de inteligencia artificial generativa y guardarlas en un directorio local de su equipo, solicitando al usuario, a través de una interfaz de usuario una descripción textual de las imágenes a generar, además de unas opciones básicas obligatorias y unas opciones avanzadas modificables de forma opcional.

Entre las opciones básicas obligatorias se encuentran el número de imágenes a generar, el dispositivo en el que se ejecutará el modelo, que podrá ser la CPU o una GPU CUDA [23] disponible en el equipo del usuario, un directorio existente en el equipo del usuario en el que se guardarán las imágenes y la selección de uno de los modelos generativos disponibles en la herramienta. Los modelos generativos disponibles en la herramienta serán: *runwayml/stable-diffusion-v1-5* [24], *stabilityai/stable-diffusion-2 [25]*, *stabilityai/stable-diffusion-xl-base-1.0 [26]* y *stabilityai/sdxl-turbo [27]*. Además, previamente a usar un modelo determinado para realizar la generación de imágenes, este deberá ser cargado en memoria por el usuario a través de la interfaz de usuario.

En cuanto a las opciones avanzadas opcionales, serán un conjunto de parámetros que permitirán ajustar la generación de imágenes de los modelos, estos parámetros podrán ser modificados por el usuario una vez se haya seleccionado un modelo y se haya cargado en memoria. Inicialmente los parámetros tendrán los valores por defecto del modelo seleccionado y concretamente los parámetros que se podrán modificar para cada modelo serán los siguientes: *strength*, *inference steps*, *guidance scale*, *predicted noise* y *negative prompt* [28].

5.2 Requisitos del Sistema

En esta sección se incluirán tanto los requisitos funcionales como los requisitos no funcionales que definirán el sistema a desarrollar, en este caso la herramienta generativa de imágenes.

5.2.1 Obtención de los Requisitos del Sistema

5.2.1.1 Requisitos funcionales

RF1. El sistema permitirá al usuario generar imágenes basadas en una descripción textual utilizando modelos de inteligencia artificial generativa.

RF1.1. El sistema mostrará al usuario una opción para introducir un texto descriptivo de las imágenes a generar.

RF1.1.1. El sistema mostrará un área en la que el usuario podrá insertar el texto descriptivo.

RF1.1.2. El texto no podrá ser vacío.

RF1.1.3 Esta opción es de carácter obligatorio.

RF1.2. El sistema mostrará al usuario una opción para introducir el directorio donde se guardarán las imágenes.

RF1.2.1. Esta opción es de carácter obligatorio.

RF.1.2.2. El sistema comprobará que la ruta introducida existe en el equipo del usuario.

RF1.2.2.3. En caso de que no exista, el sistema mostrará un mensaje de error indicando la causa de este.

RF1.3 El sistema mostrará al usuario una opción al usuario para introducir el número de imágenes a generar.

RF1.3.1. Esta opción es de carácter obligatorio.

RF1.3.1. El número de imágenes será un número entero mayor o igual a 1 y menor o igual a 1.000.000.

RF1.3.1.1. Este número podrá ser ajustable a petición del cliente.

RF1.3.1.2. En caso de que el usuario introduzca un valor inválido el sistema mostrará un mensaje de error indicando la causa de este.

RF1.4. El sistema mostrará al usuario una opción para seleccionar en que *hardware* ejecutar el modelo de generación de imágenes.

RF1.4.1. Esta opción permitirá solamente seleccionar un ítem, de forma exclusiva en cada generación, es decir no se podrá seleccionar más de uno a la vez.

RF1.4.1.1. El sistema permitirá seleccionar el ítem *CPU*.

RF1.4.1.2. El sistema permitirá seleccionar el ítem *GPU CUDA*.

RF1.4.2. Esta opción es de carácter obligatorio.

RF1.4.3. El sistema establecerá el ítem *CPU* de forma predeterminada.

RF1.5. El sistema mostrará al usuario una opción para seleccionar el modelo en el que se ejecutará la generación de imágenes.

RF1.5.1. Esta opción será de carácter obligatorio.

RF1.5.2. Esta opción permitirá seleccionar un modelo, de forma exclusiva, es decir, no se podrá seleccionar más de uno a la vez.

RF1.5.2.1. El sistema permitirá seleccionar el modelo *runwayml/stable-diffusion-v1-5***.**

RF1.5.2.2. El sistema permitirá seleccionar el modelo *stabilityai/stable-diffusion-2***.**

RF1.5.2.3. El sistema permitirá seleccionar el modelo *stabilityai/stable-diffusion-xl-base-1.0***.**

RF1.5.2.4. El sistema permitirá seleccionar el modelo *stabilityai/sdxl-turbo.*

RF1.5.3. El sistema mostrará una opción al usuario para cargar en memoria el modelo seleccionado.

RF1.5.3.1. El sistema permitirá cargar en memoria el modelo seleccionado en caso de que no haya sido cargado previamente.

RF1.5.3.1.1. El sistema mostrará al usuario durante la ejecución del proceso de carga del modelo que este proceso se encuentra en progreso.

RF1.5.3.1.2. En caso de que el proceso de carga se haya completado, el sistema mostrará al usuario un mensaje informando que el modelo se ha cargado correctamente.

RF1.5.3.1.3. En caso de que el proceso de carga no se haya podido completar, el sistema mostrará un mensaje de error al usuario informando la causa de este.

RF1.6. El sistema permitirá al usuario modificar parámetros de ajuste de generación del modelo seleccionado **[RF1.5]**.

RF1.6.1. El sistema mostrará al usuario un área en la que se mostrarán las opciones que permiten modificar los parámetros del modelo seleccionado.

RF1.6.1.1. El sistema solamente mostrará el área en caso de que el modelo seleccionado se encuentre cargado en memoria **[RF1.5.3]**

RF1.6.1.3. El sistema establecerá un valor inicial para cada opción comprendida en el área.

RF1.6.1.3.1. El valor inicial se corresponde con el valor por defecto del parámetro del modelo seleccionado, asociado a esa opción.

RF1.6.1.4. El usuario modificará el valor de las opciones comprendidas en el área de forma opcional.

RF1.6.1.5. El sistema mostrará una opción al usuario para modificar el valor del parámetro *strength* del modelo seleccionado.

RF1.6.1.5.1. El valor debe ser un número decimal mayor o igual a 0.0 y menor o igual a 1.0.

RF1.6.1.6. El sistema mostrará una opción al usuario para modificar el valor del parámetro *inference steps* del modelo seleccionado.

RF1.6.1.6.1. El valor debe ser un número entero mayor o igual a 1 y menor o igual a 100.

RF1.6.1.7. El sistema mostrará una opción al usuario para modificar el valor del parámetro *guidance scale* del modelo seleccionado.

RF1.6.1.7.1. El valor debe ser un número decimal mayor o igual a 0.0 y menor o igual a 10.0.

RF1.6.1.8. El sistema mostrará una opción al usuario para modificar el valor del parámetro *predicted noise* del modelo seleccionado.

RF1.6.1.8.1. El valor será *predicted noise* activado o desactivado.

RF1.6.1.9. El sistema mostrará un área al usuario para modificar el valor del parámetro *negative prompt* del modelo seleccionado.

RF1.6.1.9.1. El valor debe ser de tipo texto.

RF1.6.1.10. El sistema solamente dará la opción al usuario de seleccionar un valor válido en cada una de las opciones del área.

RF1.7. El sistema mostrará al usuario una opción que permite arrancar el proceso de generación de imágenes.

RF1.7.1. El sistema permitirá arrancar el proceso de generación de imágenes al usuario tras validar que el usuario ha introducido datos válidos en las opciones.

RF1.7.1.1. El sistema comprobará que se ha introducido un directorio válido **[RF1.2.].**

RF1.7.1.2. El sistema comprobará que se ha introducido una descripción textual de las imágenes válida **[RF1.1.].**

RF1.7.1.3. El sistema comprobará que se ha introducido un número de imágenes a generar válido **[RF1.3.].**

RF1.7.1.4. En caso de que se introduzca algún dato inválido en las opciones, el sistema mostrará al usuario un mensaje de error, indicando la causa de este.

RF1.7.2. El sistema permitirá arrancar el proceso de generación de imágenes al usuario en caso de que el modelo seleccionado esté cargado en memoria **[RF1.5.3.]**.

RF1.7.2.1. En caso de que el modelo no esté cargado en memoria, el sistema mostrará al usuario un mensaje de error, indicando la causa de este.

RF1.7.3. El sistema gestionará el proceso de generación de imágenes guiado por las opciones introducidas por el usuario.

RF1.7.3.1. El sistema generará las imágenes ejecutando el modelo seleccionado por el usuario **[RF1.5.]**.

RF1.7.3.1.1. El modelo seleccionado utilizará como parámetro *prompt* el texto descriptivo introducido por el usuario **[RF1.1.]**.

RF1.7.3.1.2. Los valores de opciones de parámetros de ajuste del modelo **[RF1.6]** se utilizarán como parámetros en la ejecución del modelo seleccionado.

RF1.7.3.2. El sistema guardará cada imagen en el directorio indicado por el usuario **[RF1.2.]**.

RF1.7.3.2.1. El sistema guardará cada imagen de forma individual, es decir en el momento que el sistema genere la imagen.

RF1.7.3.2.2. El sistema guardará las imágenes en formato *png*.

RF1.7.3.2. El sistema generará el número de imágenes indicado por el usuario **[RF1.3.]**.

RF1.7.3.4. El sistema ejecutará la generación de imágenes en el hardware indicado por el usuario **[RF1.4.]**.

RF1.7.3.4.1. En caso de que el usuario introduzca la opción *GPU CUDA* el sistema comprobará que el equipo del usuario tenga disponible una *GPU* compatible con *CUDA*.

RF1.7.3.4.1.1. Si el usuario no tiene disponible una *GPU CUDA*, el sistema ejecutará el modelo en la *CPU*.

RF1.7.3.4.1.2. Si el usuario no tiene disponible una *GPU CUDA*, el sistema mostrará un mensaje indicando que el modelo se ejecutará en la *CPU* debido a la causa mencionada.

RF1.7.3.4.2. En caso de que el usuario introduzca la opción *CPU*, el sistema ejecutará el modelo en la *CPU* del usuario.

RF1.7.4. Una vez comenzado el proceso de generación de imágenes, el sistema mostrará al usuario un área que indique el estado en que se encuentra el proceso.

RF1.7.4.1. El sistema informará al usuario durante la ejecución del proceso que la generación se encuentra en progreso.

RF1.7.4.2. El sistema informará al usuario durante la ejecución del proceso, el número de imágenes ya generadas por este.

RF1.7.4.3. El sistema mostrará al usuario cada imagen en el momento en que se haya generado.

RF1.7.5. Una vez finalizado el proceso de generación de imágenes, el sistema informará al usuario el estado en el que este ha terminado.

RF1.7.5.1. En caso de que el proceso se ejecute correctamente, finalizará en estado de éxito.

RF1.7.5.2. En caso de que se produzca un error durante el proceso, finalizará en un estado de error.

RF1.7.6. Una vez finalizado el proceso de generación de imágenes el sistema informará al usuario en que ruta se han guardado las imágenes.

5.2.1.2 Requisitos no funcionales

RNF1. El sistema será diseñado de forma que sea posible añadir nuevos modelos generativos en el futuro.

RNF2. El sistema será diseñado de forma que sea posible configurar nuevos parámetros del modelo en el futuro.

RNF3. El sistema se implementará utilizando la versión 3.12. de *Python*.

RNF3.1. El código fuente deberá seguir la guía de estilo *PEP8* de *Python*.

RNF4. Las funciones principales del código fuente deberán ser documentadas de forma clara.

RNF5. La interfaz de usuario no se bloqueará durante la ejecución de procesos de generación de imágenes.

RNF6. El sistema indicará al usuario la utilidad de cada opción de la interfaz de usuario.

RNF7. En caso de que se ejecuten procesos con largos tiempos de espera para el usuario, el sistema indicará al usuario que estos procesos se encuentran en curso.

5.2.2 Identificación de Actores del Sistema

El único actor que tendrá el sistema será el denominado **Usuario,** será aquel usuario que utilice la herramienta generativa de imágenes. Su actividad consistirá en configurar los parámetros necesarios para llevar a cabo la generación de imágenes y proceder a lanzar este proceso de generación de imágenes en la herramienta.

5.2.3 Especificación de Casos de Uso

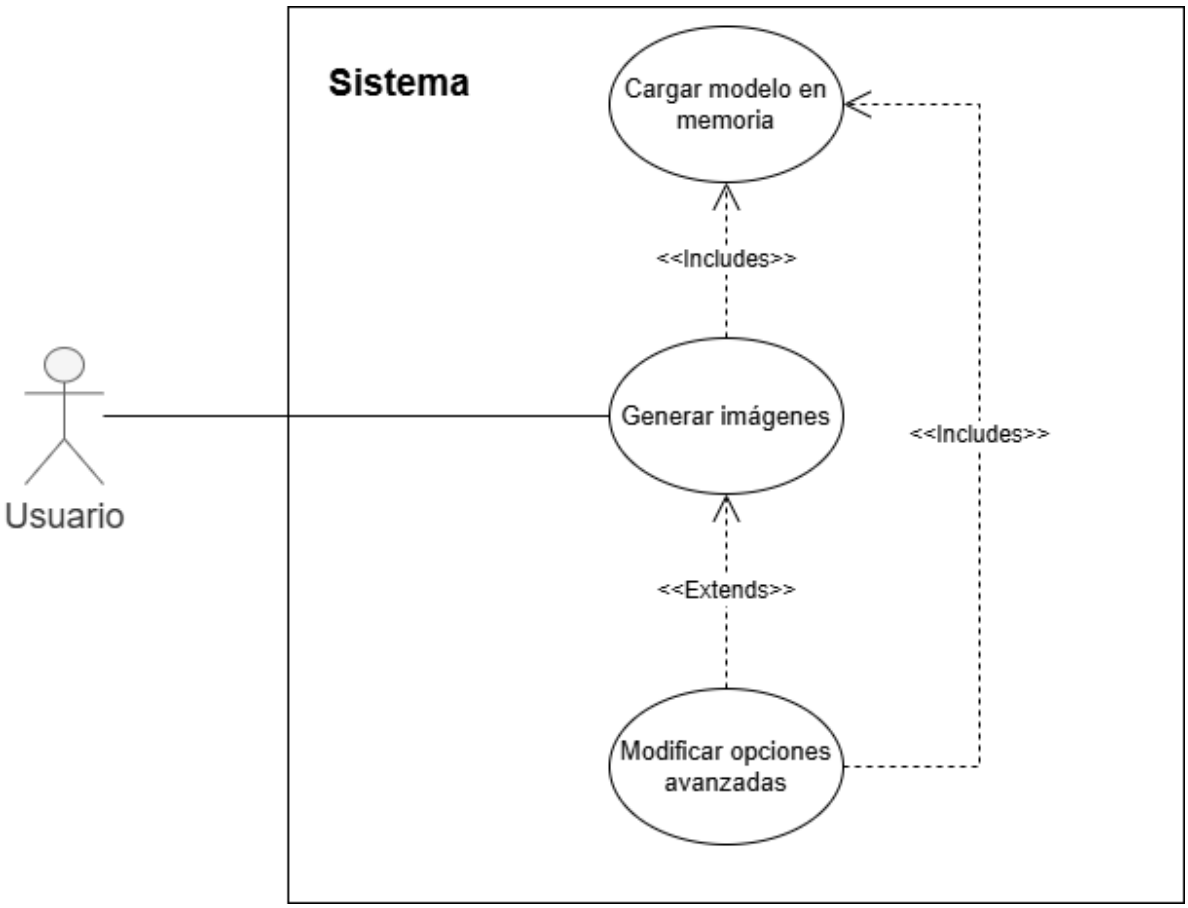


Figura 5.1 Diagrama de casos de uso del sistema

Nombre del Caso de Uso
Generar imágenes
Descripción
El usuario podrá generar imágenes introduciendo previamente en el sistema un texto descriptivo además de una serie de opciones, entre ellas se encuentran: • Modelo de generación de imágenes. • *Hardware* de ejecución del modelo. • Número de imágenes a generar. • Directorio local en el que guardar las imágenes. • Parámetros del modelo de generación. Una vez introducidas las opciones, el usuario indicará al sistema que inicie el proceso de generación y el sistema comenzará a generar el número de imágenes seleccionado, ejecutando el modelo seleccionado con los parámetros y descripción textual de las imágenes establecidos como entrada, en el *hardware* seleccionado. Durante esta ejecución el modelo generará cada imagen y el sistema las mostrará al usuario y las guardará en el directorio indicado por este, hasta que el número de imágenes total haya sido generado.

Nombre del Caso de Uso
Cargar modelo en memoria
Descripción
El usuario seleccionará un modelo de entre los ofrecidos por el sistema, una vez seleccionado, el sistema ofrecerá la opción al usuario de cargar este modelo en memoria, en caso de que no haya sido cargado previamente. En caso de que el usuario indique al sistema de cargar el modelo seleccionado, el sistema iniciará el proceso de cargar este modelo en memoria y notificará al usuario una vez este proceso haya terminado. Este caso de uso es de obligatoria previa realización para realizar el caso de uso Generar imágenes y Modificar opciones avanzadas.

Nombre del Caso de Uso
Modificar opciones avanzadas
Descripción
El usuario modificará los parámetros de generación del modelo seleccionado, a través de una serie de opciones que mostrará el sistema. Inicialmente el sistema establecerá los valores por defecto del modelo seleccionado de estos parámetros como valores iniciales en estas opciones, por lo que el sistema utilizará los parámetros por defecto para la generación de imágenes en caso de que el usuario decida no modificarlos, ya que este caso de uso es de carácter opcional para realizar el caso de uso Generar imágenes.

5.3 Identificación de los Subsistemas en la Fase de Análisis

En esta sección se descompondrá el sistema en diferentes subsistemas, es decir sistemas más pequeños, además, posteriormente se describirán las interfaces que permitirán realizar la comunicación entre estos subsistemas.

5.3.1 Descripción de los Subsistemas

El sistema se descompondrá en dos subsistemas principales, los cuales se describirán a continuación:

- **Subsistema UI**: Este subsistema proporciona una interfaz gráfica al usuario, lo que permite al usuario ejecutar y configurar el proceso de generación de imágenes a través de esta interfaz, para poder realizar ambas operaciones este subsistema se comunica con el Subsistema Generador de imágenes. Este subsistema se ha diseñado para ejecutarse en el equipo del usuario como una aplicación de escritorio.
- **Subsistema Generador de imágenes**: Este subsistema ofrece una interfaz a través de la cual se permite realizar las operaciones de cargar un modelo en memoria, configurar los parámetros y ejecutar un modelo generativo de imágenes. Este subsistema permite realizar estas operaciones en una serie de modelos generativos diferentes, obteniendo las imágenes deseadas a partir de la configuración y ejecución de alguno de estos modelos.

5.3.2 Descripción de los Interfaces entre Subsistemas

El sistema está diseñado de forma que los subsistemas están integrados dentro del mismo programa desarrollado mediante el lenguaje de programación *Python*. Teniendo esto en cuenta los subsistemas se comunicarán internamente a través de mensajes dentro del mismo sistema siguiendo la arquitectura *MVC*.

5.4 Diagrama de Clases Preliminar del Análisis

En esta sección se mostrará el diagrama de clases preliminar identificado durante la fase de análisis del sistema. Para ello primeramente se mostrará el diagrama *UML* de las clases identificadas y posteriormente, se proporcionará la descripción con mayor detalle de cada una de estas clases.

5.4.1 Diagrama de Clases

A continuación, se muestra el diagrama *UML* de las clases identificadas durante la fase de análisis.

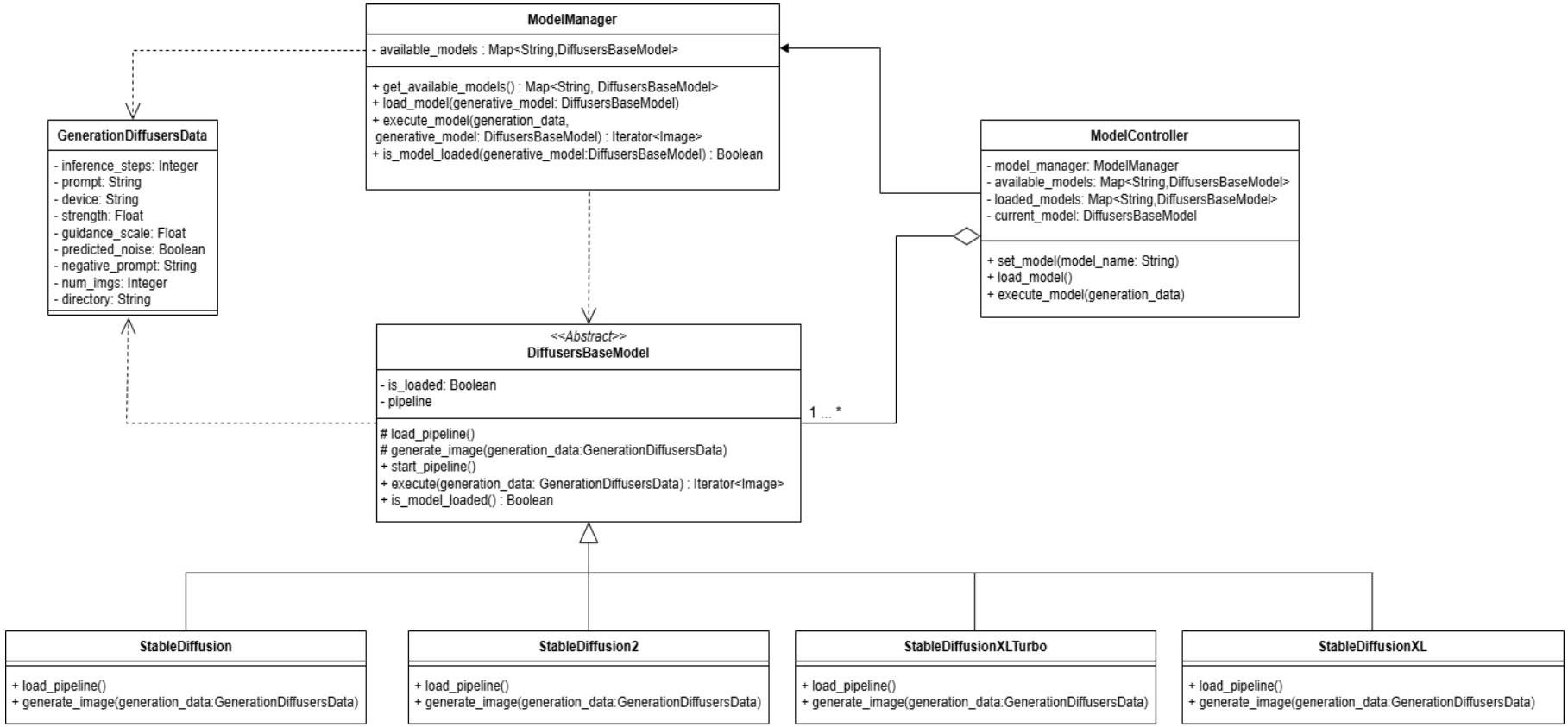


Figura 5.2 Diagrama de clases de fase de análisis

5.4.2 Descripción de las Clases

A continuación, en los subapartados siguientes se describirán las clases identificadas en el diagrama *UML* elaborado anteriormente, donde en cada uno de estos subapartados se incluirán las clases de cada subsistema identificado durante esta fase.

5.4.2.1 Subsistema UI

Nombre de la Clase
ModelController
Descripción
Se trata del controlador principal que gestionará la interacción entre el subsistema UI y el subsistema generador de imágenes. Esta clase permite de manejar la lógica de negocio relacionada con la selección y carga de modelos, así como la iniciar el proceso de generación de imágenes
Responsabilidades
Coordinará las operaciones necesarias para llevar a cabo la generación de imágenes según lo especificado por el usuario. Entre estas operaciones se encuentran: Selección de modelo a utilizar, carga en memoria de los modelos indicados por el usuario, inicio del proceso de generación de imágenes utilizando el modelo y parámetros seleccionados y delegación de las tareas relacionadas con la ejecución de generación de imágenes y configuración de modelos mediante la comunicación con la clase *ModelManager*
Atributos Propuestos
model_manager: Instancia de la clase *ModelManager* utilizada para interactuar con los modelos generativos. **available_models**: Diccionario que contiene los modelos disponibles para ser utilizados por el usuario. Mapea los nombres de los modelos a su clase correspondiente. **loaded_models**: Diccionario que almacena los modelos que ya han sido cargados en memoria para evitar que sean cargados en memoria más de una vez. **current_model:** Mantiene una referencia a la instancia del modelo que se encuentra seleccionado por el usuario en un momento concreto.
Métodos Propuestos
set_model: Establece como modelo seleccionado en un momento concreto a partir del nombre proporcionado, creando una instancia de este en caso de que no haya sido creada previamente. **load_model**: Inicia el proceso de carga en memoria del modelo que se encuentra seleccionado actualmente. **execute_model:** Inicia el proceso de generación de imágenes utilizando para ello el modelo que se encuentra seleccionado actualmente.

5.4.2.2 Subsistema Generador de Imágenes

Nombre de la Clase
ModelManager
Descripción
Clase encargada de gestionar las operaciones de los modelos generativos, la cual funcionará como intermediaria entre el controlador y las instancias de los modelos generativos. Esta clase facilitará la carga en memoria, ejecución y obtención de información relacionada con los modelos disponibles en el sistema.
Responsabilidades
Tiene como objetivo centralizar la gestión de los modelos generativos de forma que se pueda simplificar el flujo de trabajo. Entre sus responsabilidades principales se encuentran: proporcionar los modelos disponibles en el sistema, la carga en memoria de modelos, la ejecución de los modelos, verificar el estado de los modelos y mapear los datos de generación recibidos para ser enviados a los modelos de forma correcta para realizar las operaciones de generación de imágenes.
Atributos Propuestos
available_models: Diccionario que mapea los modelos disponibles a sus clases correspondientes, permite al sistema conocer que modelos están disponibles y asociarlos a sus clases.
Métodos Propuestos
get_available_models: Retorna un diccionario con los modelos generativos disponibles en el sistema. **load_model**: Carga el modelo generativo proporcionado en memoria, interactuando con la operación correspondiente del modelo proporcionado. **execute_model:** Ejecuta el modelo generativo proporcionado especificando a este los parámetros y opciones también proporcionados. Devolverá un iterador relacionado con las imágenes generadas por el modelo proporcionado, de forma secuencial. **is_model_loaded:** Verifica si el modelo proporcionado se encuentra cargado en memoria, lo que permitirá evitar cargas en memoria de los modelos redundantes.

Nombre de la Clase
DiffusersBaseModel
Descripción
Se trata de una clase abstracta de que define la interfaz común y funcionalidad compartida para todos los modelos generativos incluidos en el sistema. Esta clase incluye los métodos abstractos que las clases hijas deben implementar para manejar la implementación concreta que difiere en cada modelo.
Responsabilidades
Proporciona la estructura fundamental que deben heredar las clases de los modelos concretos, lo que facilita la extensión del sistema con nuevos modelos, así como asegurar la consistencia en la implementación. Entre las responsabilidades concretas de la clase, además de la definición de la interfaz principal, se encuentran: gestionar la carga del modelo en memoria, control de la ejecución de la generación de imágenes, configuración del dispositivo de ejecución, así como proporcionar los parámetros por defecto de los modelos.
Atributos Propuestos
is_loaded: Indica si el modelo está cargado en memoria.

pipeline: Se trata del pipeline del modelo, utilizado para llevar a cabo la generación de imágenes en los modelos. El tipo específico de este atributo podrá variar en función del modelo.
Métodos Propuestos
load_pipeline: Se trata de un método abstracto a ser implementado por las subclases, se encarga de cargar el pipeline especifico de cada subclase en memoria. **generate_image**: Método abstracto a ser implementado por las subclases, se encarga de generar una imagen utilizando el pipeline cargado en memoria previamente. De forma que en cada subclase se implementa en función de cómo genera imágenes cada modelo específico. **start_pipeline**: Comprueba si el pipeline del modelo no ha sido cargado en memoria, en ese caso se encarga de cargar el pipeline en memoria a través del método **load_pipeline** a ser implementado en cada subclase. **execute:** Se encarga de configurar el dispositivo de ejecución y de gestionar la generación de imágenes. Este método iterará sobre el número de imágenes a generar y llamando al método **generate_image**, que implementará cada subclase devolverá cada imagen generada por el modelo concreto a través de un generador. **is_model_loaded:** Devolverá el atributo **is_loaded**, indicando si el pipeline del modelo ha sido cargado en memoria.

Nombre de la Clase
StableDiffusion
Descripción
Clase que implementará la funcionalidad relacionada con el modelo *Stable Diffussion 1.5*. Implementará la clase *DiffusionBaseModel* y proporcionará las implementaciones de las operaciones necesarias para utilizar este modelo, definidas en la clase base.
Responsabilidades
Proporcionar la implementación de la carga en modelo y generación de imágenes del modelo *Stable Diffussion 1.5*, así como de proporcionar los parámetros por defecto para este modelo.
Atributos Propuestos
Métodos Propuestos
load_pipeline: Implementa la funcionalidad para cargar el pipeline en memoria del modelo *Stable Diffusion 1.5.* **generate_image**: Implementa la funcionalidad para generar una imagen utilizando el modelo *Stable Diffusion 1.5.*

Nombre de la Clase
StableDiffusion2
Descripción
Clase que implementará la funcionalidad relacionada con el modelo *Stable Diffusion* versión 2. Implementará la clase *DiffusionBaseModel* y proporcionará las implementaciones de las operaciones necesarias para utilizar este modelo, definidas en la clase base.
Responsabilidades
Proporcionar la implementación de la carga en modelo y generación de imágenes del modelo *Stable Diffusion* versión 2, así como de proporcionar los parámetros por defecto para este modelo.
Atributos Propuestos

Métodos Propuestos
load_pipeline: Implementa la funcionalidad para cargar el pipeline en memoria del modelo Stable Diffusion versión 2. **generate_image**: Implementa la funcionalidad para generar una imagen utilizando el modelo Stable Diffusion versión 2.

Nombre de la Clase
StableDiffusionXLTurbo
Descripción
Clase que implementará la funcionalidad relacionada con el modelo *Stable Diffusion XL Turbo*. Implementará la clase *DiffusionBaseModel* y proporcionará las implementaciones de las operaciones necesarias para utilizar este modelo, definidas en la clase base.
Responsabilidades
Proporcionar la implementación de la carga en modelo y generación de imágenes del modelo *Stable Diffusion XL Turbo*, así como de proporcionar los parámetros por defecto para este modelo.
Atributos Propuestos
Métodos Propuestos
load_pipeline: Implementa la funcionalidad para cargar el pipeline en memoria del modelo *Stable Diffusion XL Turbo*. **generate_image**: Implementa la funcionalidad para generar una imagen utilizando el modelo *Stable Diffusion XL Turbo.*

Nombre de la Clase
StableDiffusionXL
Descripción
Clase que implementará la funcionalidad relacionada con el modelo *Stable Diffusion XL*. Implementará la clase *DiffusionBaseModel* y proporcionará las implementaciones de las operaciones necesarias para utilizar este modelo, definidas en la clase base.
Responsabilidades
Proporcionar la implementación de la carga en modelo y generación de imágenes del modelo *Stable Diffusion XL*, así como de proporcionar los parámetros por defecto para este modelo.
Atributos Propuestos
Métodos Propuestos
load_pipeline: Implementa la funcionalidad para cargar el pipeline en memoria del modelo *Stable Diffusion XL*. **generate_image**: Implementa la funcionalidad para generar una imagen utilizando el modelo *Stable Diffusion XL*.

Nombre de la Clase
GenerationDiffusersData
Descripción
Se trata de una clase que encapsula todos los atributos necesarios para ejecutar los modelos de generación de imágenes. Actúa como contenedor de datos, de forma que servirá para proporcionar una forma estructurada de enviar datos a los modelos generativos y llevar a cabo la generación

Responsabilidades
Permitir la gestión de forma sencilla de los parámetros requeridos por los modelos de generación de imágenes, de forma que se asegura que los modelos recibirán la información de forma correcta para llevar a cabo el proceso de generación.
Atributos Propuestos
inference_steps: Número de pasos de inferencia que el modelo utilizará durante la generación de imágenes. **prompt**: Texto descriptivo que guía al modelo en la creación de imágenes. Es la entrada principal que define el contenido de la imagen generada. **device:** Dispositivo en el que se ejecutará el modelo, como "*CPU*" o "*GPU*". Permite especificar el *hardware* a utilizar. **strength:** Controla la intensidad o fuerza de ciertos efectos en la generación, dependiendo del modelo. Puede influir en la fidelidad al *prompt* o en la creatividad de la generación. **guidance_scale:** Determina qué tan fuertemente el modelo debe seguir el *prompt* proporcionado. Valores más altos hacen que el modelo siga más de cerca el *prompt*. **predicted_noise:** Indica si se debe añadir ruido predicho durante la generación. Puede afectar la diversidad y el estilo de las imágenes generadas. **negative_prompt:** Texto que el modelo debe evitar en la generación de imágenes. Ayuda a refinar y controlar los resultados eliminando elementos no deseados. **num_imgs:** Número de imágenes que se desean generar. Permite generar múltiples imágenes en una sola ejecución**.** **directory:** Ruta o directorio donde se guardarán las imágenes generadas. Facilita la organización y el acceso posterior a los resultados.
Métodos Propuestos

5.5 Análisis de Casos de Uso y Escenarios

A continuación, en esta sección se describirán los casos de uso identificados en las secciones previas de este capítulo y se detallarán junto a sus escenarios, además de sus precondiciones, postcondiciones, actores involucrados y toda la información requerida para comprender los casos de uso de forma detallada.

5.5.1 Generar imágenes

1. Generar imágenes	
Precondiciones	Existe al menos un modelo cargado en memoria.
Postcondiciones	El sistema habrá guardado las imágenes en la ruta especificada por el usuario y el sistema volverá a estar en disposición de generar imágenes.
Actores	Usuario
Descripción	**Escenario1.1.** 1. El usuario selecciona un modelo cargado en memoria. 2. El sistema muestra las opciones avanzadas del modelo. 3. El usuario ingresa un texto descriptivo de las imágenes a generar en el sistema.

	4. El usuario selecciona el hardware en el que se ejecutara el modelo. 5. El usuario introduce el número de imágenes a generar. 6. El usuario introduce un directorio existente en su equipo. 7. El usuario puede optar por modificar las opciones avanzadas del modelo seleccionado mediante el caso de uso *"Modificar opciones avanzadas"*. 8. El usuario inicia el proceso de generación de imágenes haciendo "clic" en el botón "*Generar imágenes"*. 9. El sistema desactiva el botón *"Generar imágenes"* y el botón *"Cargar Modelo"*. 10. El sistema indica el estado en el que se encuentra el progreso de la generación de imágenes. 11. El sistema mostrará cada imagen generada al usuario de forma secuencial. 12. El sistema guardará cada imagen generada en el directorio indicado por el usuario. 13. El sistema reactiva los botones *"Generar imágenes" y "Cargar modelo"*. 14. El sistema estará listo para volver a iniciar el caso de uso *"Generar imágenes"*.
Variaciones (escenarios secundarios)	• **Escenario 1.2**: El usuario selecciona un modelo no cargado en memoria previamente a hacer "clic" en el botón *"Generar imágenes*" (**Paso 8, Escenario1.1**). o Tras hacer "clic" en el botón "G*enerar imágenes"* el sistema mostrará un mensaje de error al usuario indicando que debe seleccionar un modelo cargado en memoria y no iniciará el proceso de generación. • **Escenario1.3**: El usuario introduce un texto descriptivo vacío en el **Paso 3, Escenario1.1,** posteriormente hace clic en el botón *"Generar imágenes"*. o El sistema mostrará un mensaje de error al usuario indicando que ha introducido un *prompt* vacío y no iniciará el proceso de generación. • **Escenario1.4**: El usuario selecciona la opción de *"GPU CUDA"* en el hardware de ejecución en **Paso 4, Escenario1.1,** posteriormente el usuario llega al paso 8 y ejecuta el proceso de generación, teniendo en cuenta que el equipo del usuario no cuenta con un dispositivo *GPU CUDA*. o El sistema mostrará un mensaje al usuario indicando que no ha encontrado un dispositivo *GPU CUDA* e iniciará el proceso de generación ejecutando el modelo en la *CPU*. • **Escenario1.5:** El usuario introduce un número de imágenes a generar que no se corresponde con un entero entre 1 y 1.000.000. o El sistema mostrará un mensaje de error al usuario indicando que ha introducido un numero inválido, junto al rango de números que son aceptados y establecerá en esta opción el valor 1.

	• **Escenario1.6:** El usuario introduce un directorio no existente en su equipo en el **Paso 6. Escenario1,1**. Tras esto hace "clic" en el botón *"Generar imágenes".* ○ El sistema validará si la ruta introducida se encuentra en el equipo del usuario y mostrará un mensaje de error al usuario indicando que ha introducido una ruta no existente en su equipo y no se iniciará el proceso de generación.
Excepciones	**-**
Notas	-

Figura 5.3 Generar imágenes caso de uso y escenarios

5.5.2 Cargar modelo en memoria

2. Cargar modelo en memoria	
Precondiciones	Existe al menos un modelo ofrecido por el sistema que no ha sido cargado en memoria por el usuario.
Postcondiciones	El modelo pasará a estar cargado en la memoria del equipo del usuario y podrá utilizarse para generar imágenes.
Actores	Usuario.
Descripción	**Escenario2.1.** 1. El sistema muestra los modelos disponibles. 2. El usuario selecciona un modelo no cargado en memoria. 3. El sistema comprueba si el modelo está cargado en memoria. 4. El sistema muestra el botón *"Cargar modelo en memoria".* 5. El usuario hace clic en el botón del paso anterior. 6. El sistema oculta el botón *"Cargar modelo en memoria"* e inicia el proceso de carga del modelo. 7. El sistema muestra un mensaje al usuario indicando que el proceso de carga se ha iniciado e informa del progreso de este. 8. Una vez terminado el proceso de carga del modelo el sistema informa al usuario de que el modelo se ha cargado correctamente y cambia el estado del modelo a "*cargado en memoria".*
Variaciones (escenarios secundarios)	• **Escenario2.2:** El usuario selecciona un modelo que se encuentra cargado en memoria.

	○ El sistema comprueba que el modelo ya ha sido cargado en memoria y no mostrará el botón “Cargar modelo en memoria”.
Excepciones	-
Notas	-

Figura 5.4 Cargar modelo en memoria caso de uso y escenarios

5.5.3 Modificar opciones avanzadas

3. Modificar opciones avanzadas	
Precondiciones	Existe al menos un modelo cargado en memoria
Postcondiciones	Los parámetros utilizados por el modelo seleccionado para generar imágenes se corresponderán con los indicados por el usuario.
Actores	Usuario.
Descripción	**Escenario3.1.** 1. El usuario selecciona un modelo que ha sido cargado en memoria. 2. El sistema comprueba si el modelo seleccionado ha sido cargado en memoria y muestra la opción de mostrar el panel de opciones avanzadas. 3. El usuario selecciona la opción de mostrar opciones avanzadas. 4. El sistema muestra el panel de opciones y en cada opción el valor de cada parámetro por defecto del modelo seleccionado. 5. El usuario modifica el valor de una o varias opciones. 6. El sistema actualizará el valor de los parámetros de generación del modelo seleccionado y los utilizará en caso de que el usuario inicie el proceso de generación tras el paso anterior.
Variaciones (escenarios secundarios)	• **Escenario3.2.** Tras modificar los parámetros del modelo seleccionado actualmente, el usuario selecciona otro modelo y vuelve a seleccionar el modelo del cual había modificado sus parámetros previamente. ○ El sistema muestra en cada opción correspondiente los valores por defecto de los parámetros del modelo seleccionado. • **Escenario3.3.** El usuario selecciona un modelo que no ha sido cargado en memoria. ○ El sistema no muestra la opción de mostrar el panel de opciones avanzadas.
Excepciones	-
Notas	-

Figura 5.5 Modificar opciones avanzadas casos de uso y escenarios

5.6 Relación Escenarios – Casos de Uso – Requisitos

En la siguiente tabla se muestra de una manera visual y resumida la relación que existe entre los casos de uso y los diferentes escenarios definidos durante la fase de análisis.

Casos de Uso:	1	2	3
Escenario 1.1	X		
Escenario 1.2	X		
Escenario 1.3	X		
Escenario 1.4	X		
Escenario 1.5	X		
Escenario 1.6	X		
Escenario 2.1		X	
Escenario 2.2		X	
Escenario 3.1			X
Escenario 3.2			X
Escenario 3.3			X

Figura 5.6 Relación entre casos de uso y escenarios

5.7 Análisis de Interfaces de Usuario

En la siguiente sección se describirá la interfaz gráfica de usuario mediante el diseño de un *mockup* realizado durante la fase de análisis y se describirá el comportamiento de sus elementos a través de este *mockup*. Para el diseño de este se ha tenido en cuenta mantener esta interfaz bajo un diseño lo más simple posible, pero teniendo en cuenta el lograr un buen aspecto y usabilidad, de forma que sea de uso sencillo, intuitivo y agradable para el usuario que interactuará con esta interfaz.

5.7.1 Descripción de la Interfaz

La interfaz diseñada solamente consta de una ventana principal, la cual está dividida en diferentes paneles que agrupan elementos con un objetivo común, estos elementos consisten en una serie de opciones tanto básicas de carácter obligatorio para poder generar las imágenes deseadas por el usuario, como de unas opciones avanzadas de carácter opcional que permitirán modificar los parámetros por defecto del proceso de generación de imágenes de los modelos disponibles. Además, existirá un panel cuyo cometido será informar al usuario del estado en el que se encuentra la aplicación y los procesos que el sistema se encuentre ejecutando.

5.7.1.1 Ventana principal

Se trata de la ventana principal que contendrá todos los elementos disponibles en la aplicación, como se ha comentado previamente será la única ventana de la aplicación y permitirá al usuario

configurar todas las opciones disponibles para llevar a cabo el proceso de generación de imágenes.

A continuación, se muestra el mockup elaborado en la fase de análisis para representar la interfaz de usuario de la aplicación.

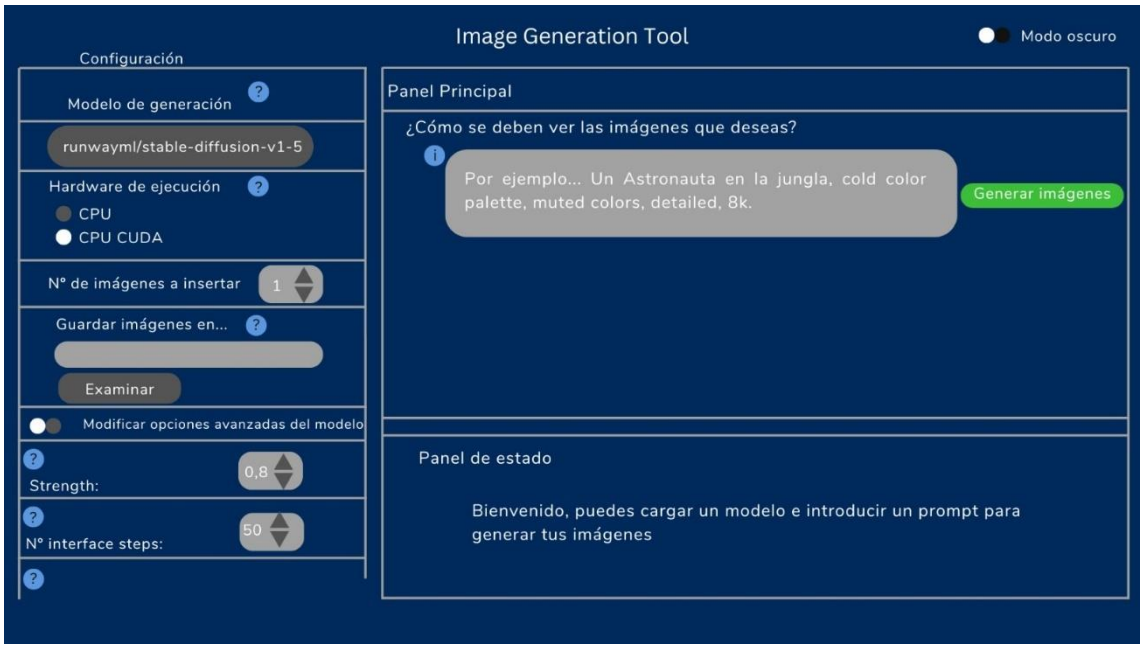


Figura 5.7 Mockup de la interfaz de usuario

A continuación, se enumerarán los elementos propuestos en el *mockup* de la interfaz diseñada junto a una descripción de estos:

- Selector de modelo de generación: Permitirá al usuario seleccionar uno de los modelos disponibles ofrecidos por el sistema.
- *Hardware* de ejecución: Permitirá al usuario seleccionar en que dispositivo se ejecutará el modelo seleccionado, solamente tendremos la opción de ejecutar el modelo en la *CPU* o en una *GPU Cuda*. El usuario solamente podrá seleccionar una de las opciones en cada generación.
- Número de imágenes a generar: Se trata de un *spinbox* que permitirá al usuario introducir el número de imágenes que desea generar.
- Selector de directorio de guardado de imágenes: Permite al usuario introducir la ruta de un directorio local en el que guardar las imágenes, además incluye un botón que permitirá examinar un directorio existente en el equipo del usuario a través del explorador de archivos.
- *Toggle* mostrar opciones avanzadas: Funcionará a modo de interruptor, de forma que al activarlo se mostrará el panel de opciones avanzadas y al desactivarlo ocultará este panel.
- Panel opciones avanzadas: Se trata de un panel con *scroll* que contendrá todas las opciones que el usuario podrá modificar, donde en cada opción se mostrará de forma

inicial el valor de cada parámetro por defecto del modelo seleccionado en el momento, en este caso en el prototipo solo se muestran dos de ellas a modo de ejemplo.

- Área de texto para introducir *prompt*: Se trata de un área donde el usuario podrá introducir el texto descriptivo de las imágenes que desea generar de modo que guiará el proceso de generación de imágenes.
- Panel de estado actual de la aplicación: Se trata de un panel que informará al usuario del estado del progreso y el estado del proceso de generación de imágenes o de la carga del modelo en memoria.
- *Tooltip* junto a cada opción: Permiten al usuario obtener una descripción detallada del cometido de cada opción junto a la que se encuentra cada *tooltip*, de forma que una vez que el usuario haga *hover* sobre uno de ellos, se mostrará una descripción de la opción.
- *Toggle* modo oscuro: Se trata de un interruptor que permitirá al usuario cambiar la temática de aspecto de la interfaz de usuario, de forma que cuando se encuentre activado la interfaz mostrará una temática oscura y cuando se desactive se mostrará una temática clara.
- Botón generar imágenes: Se trata de un botón que iniciará el proceso de generación de imágenes de acuerdo con las opciones introducidas por el usuario, teniendo en cuenta que lanzará las validaciones de todos los campos previamente a poder ejecutar este proceso, de forma que si existe algún campo incorrecto este proceso no se lanzará y se mostrará un error.
- Botón cargar modelo: Se trata de un botón encargado de iniciar el proceso de carga del modelo seleccionado en memoria. En el caso del *mockup* no se encuentra visible, ya que en este *mockup* se representa el estado de la aplicación en el momento de que el modelo seleccionado está cargado en memoria, de forma que este botón se ocultará si el modelo seleccionado si el modelo se encuentra cargado, de modo que, si el modelo seleccionado no está cargado en memoria, este botón se mostrará justo debajo del selector de modelos disponibles.

5.7.2 Descripción del Comportamiento de la Interfaz

Dado que el sistema esta directa y fuertemente influenciado por las entradas de datos y acciones que realice el usuario, el sistema deberá realizar una verificación y validación de estos datos introducidos y acciones realizadas por el usuario para evitar un mal funcionamiento de la aplicación. Además, como ayuda al usuario se han incorporado *tooltips*, cuyo objetivo es informar al usuario de que dato debe introducir en cada opción, de forma que será de ayuda para que el usuario pueda manejar la aplicación de una forma más amigable y estar informado del objetivo de cada opción de la interfaz y no introducir datos incorrectos.

5.7.2.1 Ventana principal

- El campo número de imágenes a generar solamente aceptará un valor entero comprendido entre 1 y 1.000.000. En el caso de que el usuario introduzca un dato inválido la ventana mostrará el mensaje "Entrada de usuario inválida, el número de imágenes debe ser un número comprendido entre 1 y 1.000.000".
- El campo para introducir un directorio solamente aceptará la ruta de un directorio existente en el equipo del usuario, en caso de introducir una ruta inválida la ventana mostrará el mensaje "Ruta de destino de imágenes inválida, debes introducir un directorio existente en tu equipo para guardar las imágenes generadas".
- El campo para introducir un *prompt* solamente aceptará texto no vacío, en caso de que se introduzca una entrada inválida en el área de texto del *prompt* y el usuario inicie la generación de imágenes a través del botón de generar imágenes la ventana mostrará el mensaje "*Prompt* introducido inválido, debes introducir un *prompt* no vacío, no ha sido posible generar las imágenes".
- En caso de que el usuario haya seleccionado un modelo y no lo haya cargado en memoria o ningún modelo haya sido seleccionado, al intentar iniciar el proceso de generación de imágenes, la ventana mostrará el mensaje "Error al intentar generar imágenes, no se ha cargado el modelo seleccionado, para generar imágenes se debe cargar previamente el modelo seleccionado".
- En caso de que el usuario seleccione "*GPU Cuda*" y este no tenga una *GPU Cuda* disponible en su equipo la ventana mostrará el mensaje "Error intentando detectar dispositivo *GPU Cuda*, se procederá a ejecutar el modelo en la *CPU* de su equipo".
- En caso de que el usuario intente iniciar el proceso de generación de imágenes y la opción introducida en un campo de los mencionados previamente en este listado se encuentre en un estado inválido, la ventana mostrará el mensaje correspondiente al campo cuyo dato introducido sea inválido.

5.7.3 Diagrama de Navegabilidad

En el caso de esta aplicación, la interfaz solamente tendrá una ventana principal, de modo que no habrá navegabilidad entre diferentes ventanas y por lo tanto no corresponde para este caso la inclusión de un diagrama de navegabilidad.

5.8 Especificación del Plan de Pruebas

5.8.1 Pruebas unitarias

Las pruebas unitarias se utilizarán para realizar la verificación y validación del correcto funcionamiento de los diferentes módulos del sistema. Para ello se tomarán como referencia los casos de uso y escenarios identificados durante esta fase de análisis, de forma que, las funcionalidades principales que realizará el usuario funcionan de acuerdo con lo definido y esperado.

5.8.2 Pruebas de integración y sistema

Mediante las pruebas de integración, se comprobará que los subsistemas de la aplicación se relacionan y comunican de forma adecuada, en este caso los subsistemas de la interfaz de usuario y el subsistema generador de imágenes.

5.8.3 Pruebas de usabilidad y accesibilidad

Para llevar a cabo las pruebas de usabilidad y accesibilidad se ha optado a realizar una serie de pruebas con diferentes perfiles de usuarios, de esta forma podremos comprobar que se cumplen los objetivos buscados de facilidad de uso y manejo de la aplicación, así como de la fluidez de uso de esta, para el perfil de usuarios que utilizará la aplicación.

5.8.4 Pruebas de rendimiento

Debido a que la ejecución de modelos generativos es un proceso pesado en términos de recursos *hardware*, dependerá en gran medida del equipo del usuario y del uso de una *GPU CUDA* con una buena potencia de procesamiento. Por ello las pruebas de rendimiento se harán teniendo en cuenta estas restricciones y estarán centradas en pruebas sobre los tiempos de ejecución de los procesos de generación de imágenes y de la carga de los modelos en memoria.

5.8.5 Pruebas de casos de uso

Caso de Uso 1: Generar imágenes	
Prueba1.1	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria, introduce un texto descriptivo válido, indica 1 como número de imágenes a generar, *CPU* como dispositivo, un directorio existente e inicia el proceso de generación	El sistema comienza el proceso de generación, una vez terminado este proceso, la imagen generada se encontrará guardada en el directorio indicado por el usuario.

Prueba1.2	**Resultado Esperado**
El usuario intenta generar imágenes con un modelo no cargado en memoria seleccionado.	El sistema valida que no existe un modelo cargado en memoria, por lo que no inicia el proceso de generación y muestra al usuario un mensaje de error.
Prueba1.3	**Resultado Esperado**
El usuario intenta generar imágenes con un texto descriptivo vacío.	El sistema valida que el texto introducido es vacío, por lo que no inicia el proceso de generación y muestra al usuario un mensaje de error.
Prueba1.4	**Resultado Esperado**
El usuario intenta generar imágenes con una *GPU*, pero el equipo del usuario no tiene una *GPU* instalada.	El sistema valida que el equipo del usuario no tiene disponible una *GPU CUDA*, por lo tanto, mostrará un mensaje al usuario indicando que no se ha encontrado una *GPU CUDA* e iniciará el proceso de generación en la *CPU* e informará al usuario de que se iniciará el proceso en la *CPU*.
Prueba1.5	**Resultado Esperado**
El usuario introduce un número de imágenes fuera del rango válido establecido por el sistema.	El sistema valida que el número de imágenes está fuera de rango, por lo que no inicia el proceso de generación y muestra al usuario un mensaje de error. Además, establecerá el valor 1 como número de imágenes.
Prueba1.6	**Resultado Esperado**
El usuario introduce un directorio no existente en su equipo e intenta generar imágenes.	El sistema valida que el directorio introducido es inválido, por lo que no inicia el proceso de generación y muestra al usuario un mensaje de error.

Figura 5.8 Prueba de caso de uso Generar imágenes

Caso de Uso 2: Cargar modelo en memoria	
Prueba2.1	**Resultado Esperado**
El usuario selecciona un modelo no cargado en memoria y lo carga en memoria.	El sistema inicia el proceso de carga en memoria del modelo seleccionado y una vez terminado el proceso el modelo se encontrará cargado en memoria.
Prueba2.2	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria e intenta cargarlo en memoria.	El sistema valida que el modelo seleccionado ya se encuentra cargado en memoria y no permite al usuario cargar el modelo en memoria.

Figura 5.9 Prueba de caso de uso Cargar modelo en memoria

Caso de Uso 3: Modificar opciones avanzadas	
Prueba3.1	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria y modifica el valor de las opciones avanzadas, tras esto inicia el proceso de generación.	El sistema inicia el proceso de generación indicando al modelo las opciones avanzadas establecidas por el usuario.
Prueba3.2	**Resultado Esperado**

El usuario selecciona un modelo cargado en memoria y modifica las opciones avanzadas, tras esto selecciona otro modelo y vuelve a seleccionar el modelo del que había modificado sus opciones.	El sistema establece los valores por defecto en las opciones avanzadas del modelo seleccionado primeramente por el usuario, del cual había modificado sus opciones avanzadas.
Prueba3.3	**Resultado Esperado**
El usuario selecciona un modelo que no ha sido cargado en memoria.	El sistema valida que el modelo seleccionado no ha sido cargado en memoria y no permite al usuario modificar sus opciones avanzadas.

Figura 5.10 Prueba de caso de uso Modificar opciones avanzadas

Capítulo 6. Diseño del Sistema

En este capítulo, una vez terminada la fase de análisis, nos centraremos en los aspectos relacionados con el diseño del sistema. Para empezar, se tratarán aspectos relacionados con la arquitectura del sistema, para ello se mostrarán en las siguientes secciones un conjunto de diagramas que representarán la estructura del sistema, para seguir se incluirán una serie de diagramas que expondrán el comportamiento del sistema. También se expondrán los aspectos relacionados con el diseño realizado para la interfaz de usuario y la especificación técnica diseñada para el plan de pruebas del sistema.

6.1 Arquitectura del Sistema

En esta sección se cubrirán los aspectos relacionados con la arquitectura mediante los diagramas de paquetes, componentes y despliegue. De esta forma se podrá exponer en detalle el diseño de la arquitectura del sistema.

6.1.1 Diagramas de Paquetes

A continuación, se detallan los paquetes identificados en el sistema, en el diagrama que se muestra al final de este capítulo se pueden identificar cada uno de ellos junto a las dependencias que existen entre estos. Además, en las siguientes subsecciones se detallará la descripción de cada paquete, explicando que elementos contiene cada uno y para qué sirven los mismos.

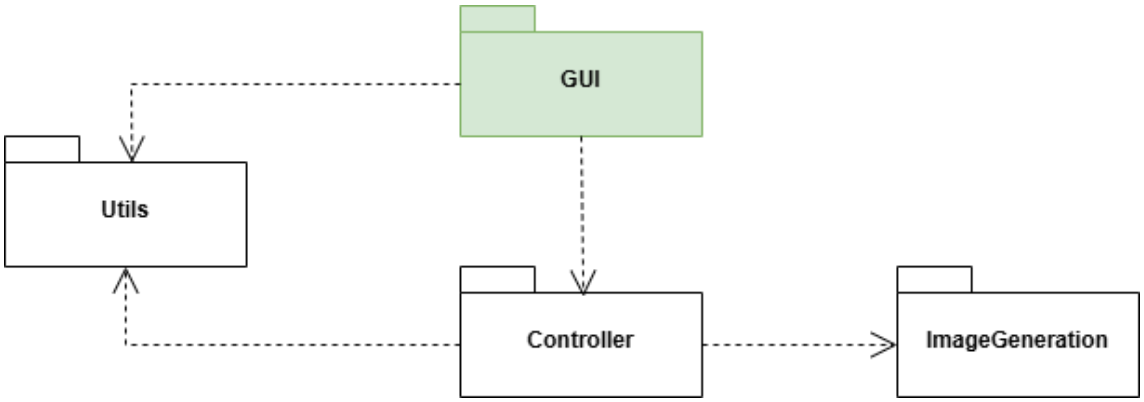


Figura 6.1 Diagrama de paquetes

6.1.1.1 Paquete GUI

Este paquete contiene todos los elementos relacionados con la interfaz de usuario, será responsable de gestionar la presentación visual de la aplicación al usuario, además de las

interacciones del usuario y las entradas introducidas por este. Por lo tanto, contendrá todas las clases relacionadas con los elementos visuales de la aplicación, como ventanas, *widgets*, paneles, etc.

6.1.1.2 Paquete Controller

Este paquete contiene todas las clases relacionadas con la lógica de negocio de la aplicación, es decir contendrá las clases que gestionarán las funcionalidades de la aplicación. Además, este paquete, actúa como intermediario entre la interfaz de usuario y los modelos de generación de imágenes facilitando la comunicación entre los paquetes que contienen estos elementos, de forma que la interfaz de usuario se mantenga de forma independiente a los modelos de generación, esto se consigue mediante varias clases de controladores que facilitan el desacoplamiento entre los paquetes mencionados.

6.1.1.3 Paquete ImageGeneration

Este paquete contiene los elementos encargados de las operaciones relacionadas con los modelos de generación de imágenes, entre estas operaciones podemos encontrar la carga en memoria de los modelos o la generación de imágenes. En este paquete se encontrarán las clases con la gestión de los modelos y los propios modelos de generación de imágenes incluidos en el sistema.

6.1.1.4 Paquete Utils

Este paquete proporciona operaciones de carácter utilitario que podrán ser de uso requerido en diferentes paquetes, entre estas operaciones se encontrarán las relacionadas con la gestión de archivos y con el manejo y procesamiento de imágenes, así como de operaciones auxiliares comunes a usar por los diferentes paquetes de la aplicación.

6.1.2 Diagramas de Componentes

A continuación, se muestra el diagrama de componentes del sistema, en el cual se identifican los componentes existentes en el sistema, así como las interfaces ofrecidas y utilizadas por cada componente, de modo que se pueda identificar de forma clara las dependencias entre cada componente del sistema y cómo los componentes interactúan entre sí.

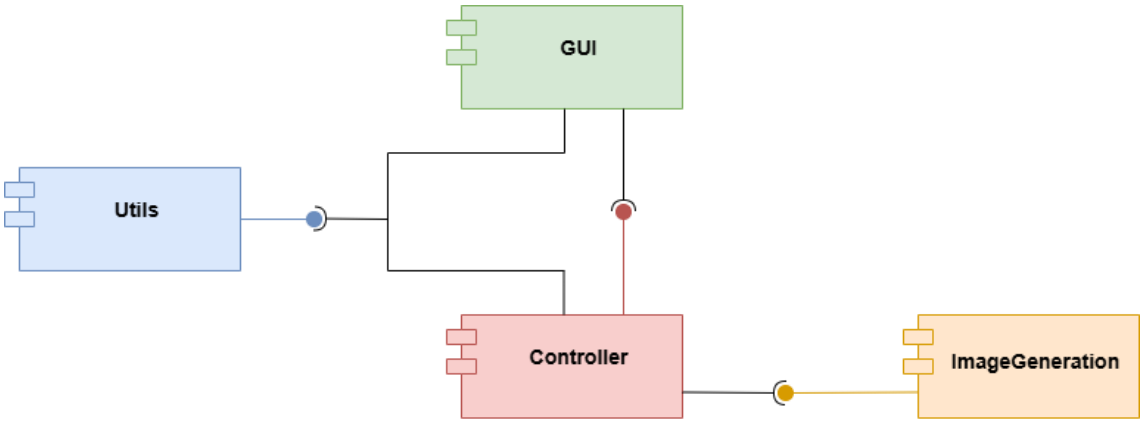


Figura 6.2 Diagrama de componentes

6.1.3 Diagramas de Despliegue

A continuación, en esta sección se detallará el diagrama de despliegue del sistema, que en este caso será muy sencillo, esto es debido a que nuestro sistema será empaquetado en un solo archivo ejecutable, el cual se ejecutará en el equipo del usuario y no requerirá de procesos a ser ejecutados en diferentes máquinas, sino que todo el sistema se ejecutará en conjunto en el equipo del usuario como una aplicación de escritorio.

Figura 6.3 Diagrama de despliegue

6.2 Diseño de Clases

En esta sección se mostrará el diagrama de clases elaborado durante la fase de diseño, para ello se ha tomado como base el diagrama elaborado durante la fase de análisis. En este diagrama se han incorporado respecto al previo las clases relacionadas con la interfaz de usuario y su correspondiente controlador, además de ciertas operaciones en alguna de las clases ya presentes en el diagrama elaborado durante la fase de análisis. Además, para este diagrama se ha elaborado de forma simplificada respecto a la forma que obtendrá realmente el sistema una vez se implemente, aunque manteniendo el nivel de detalle necesario para entender el diseño de clases del sistema, esto se ha hecho con el objetivo de mantener la legibilidad y claridad del diagrama, ya que algunos de los elementos omitidos servirán como un simple apoyo a las operaciones, atributos y clases principales que se verán detallados en el diagrama mostrado a continuación.

Por lo tanto, el diagrama de clases elaborado durante esta fase permitirá entender la estructura de clases y operaciones principales del sistema, cumpliendo con los objetivos de la fase de diseño del sistema.

6.2.1 Diagrama de Clases

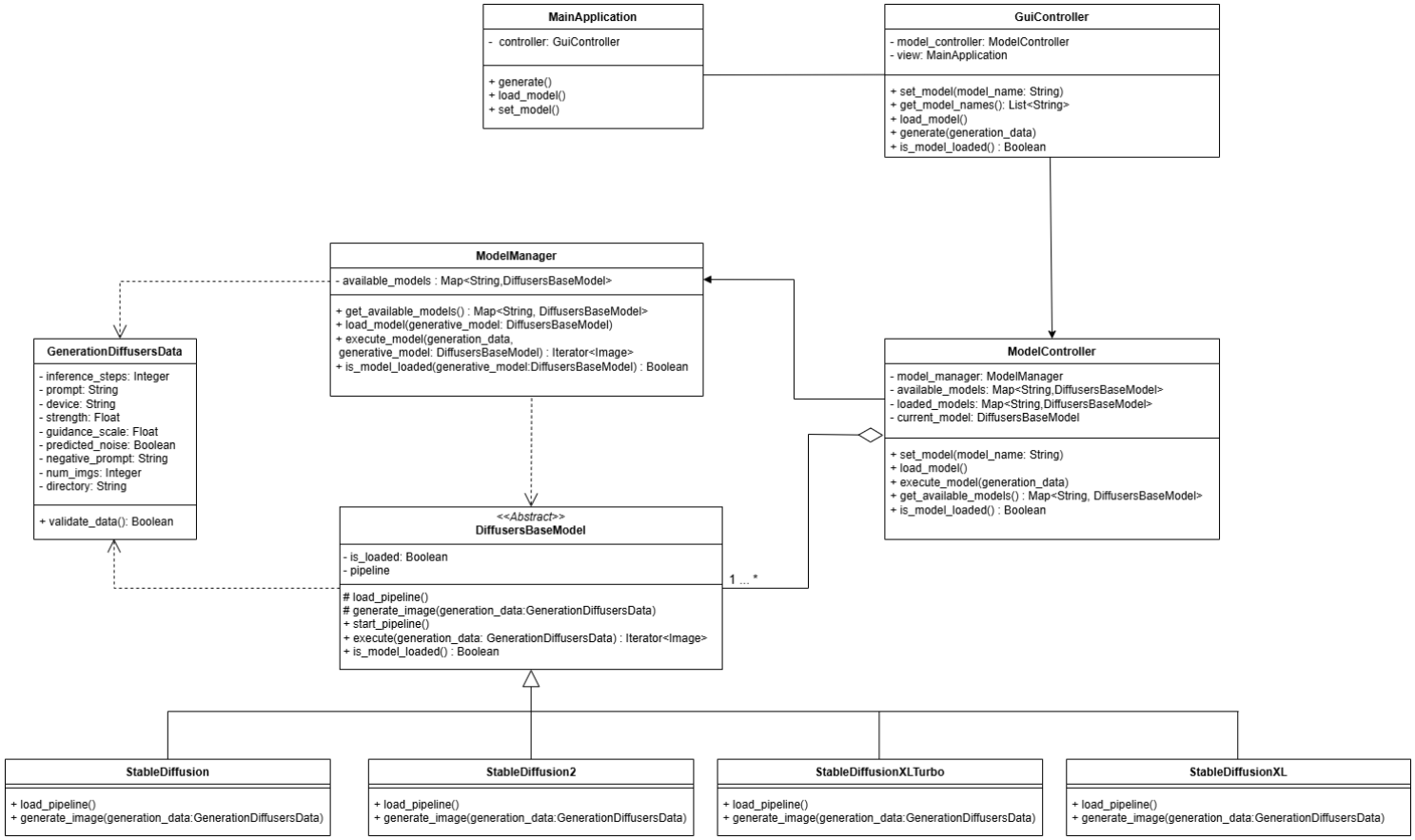


Figura 6.4 Diagrama global de clases del sistema

6.3 Diagramas de Interacción

En esta sección se definirán los diagramas de interacción correspondientes con los casos de uso del sistema, para ello se integrarán los 3 casos de uso dentro del mismo diagrama, debido a que existen casos de uso muy simples que forman parte del caso de uso principal, por lo que se realizará un único diagrama de interacción que permitirá entender de forma más clara el funcionamiento del sistema, ya que realizar estos casos de uso simples de forma separada lo único que haría sería entorpecer la creación del diagrama del caso de uso principal relacionado con la generación de imágenes y por lo tanto entorpecer también el entendimiento de este.

6.3.1 Casos de Uso 1, 2 y 3

El diagrama de interacción mostrado a continuación se corresponde con los casos de uso de generar imágenes, cargar modelo en memoria y modificar parámetros avanzados integrados dentro del mismo diagrama.

En cuanto al diagrama, podemos ver la interacción del *usuario* ***introducir datos de generación (1)***, la cual se ha simplificado, de modo que se correspondería con las acciones del usuario de introducir el *prompt*, número de imágenes a generar, dispositivo y ruta existente en su equipo. También debemos tener en cuenta que se han incluido dos recuadros para indicar el mismo bucle, debido a las restricciones de las herramientas de diseño, aunque estos se corresponden con el mismo bucle utilizado en el sistema, formando parte de este las interacciones incluidas en su interior.

Además, debemos tener en cuenta que en el diagrama se muestran las interacciones principales relacionadas con los casos de uso expuestos previamente, es decir, se han simplificado estas interacciones mostrando solamente las importantes para poder entender las interacciones de forma clara, sabiendo esto, existirán más interacciones que las mostradas en el sistema real, pero tendrán que ver con operaciones de apoyo integradas en las interacciones mostradas en el diagrama o interacciones relacionadas con la gestión de elementos de la interfaz de usuario, como mostrar u ocultar paneles, deshabilitar botones, etc.

En resumen, el diagrama se ha simplificado para poder comunicar correctamente las interacciones relacionadas con los casos de uso mencionado, pero siendo lo suficientemente completo para el objetivo buscado en este caso.

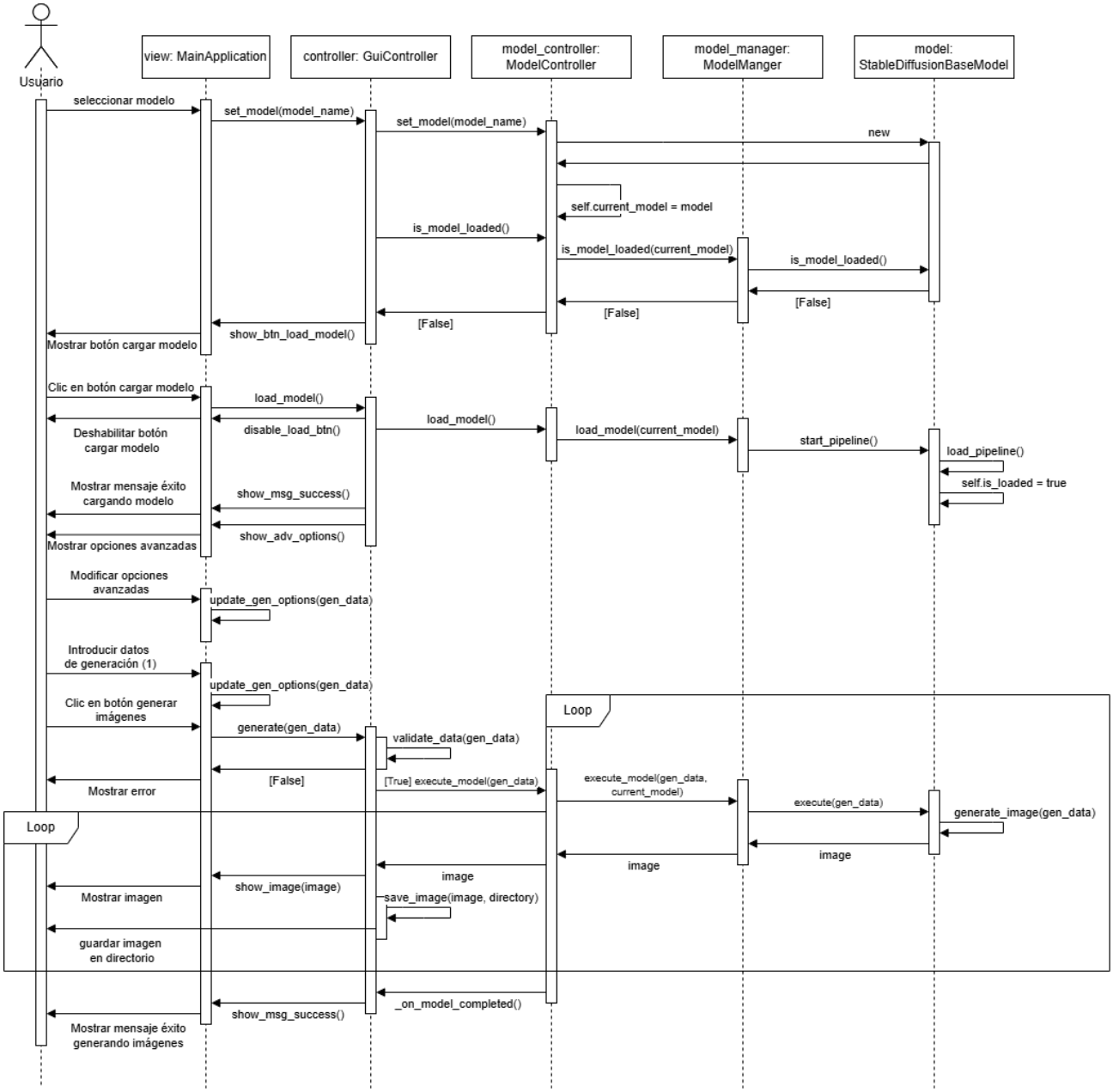


Figura 6.5 Diagrama de interacción

6.4 Diseño de la Interfaz

En esta sección se mostrará el diseño final de la interfaz de usuario, el cual se ha basado en los *mockups* realizados durante la fase de análisis. En este caso no existirá gran diferencia entre ambos, a continuación, se detallará de forma descriptiva el diseño final que se ha obtenido durante la fase actual de diseño del sistema.

A continuación, en la siguiente imagen, podemos ver la ventana principal del sistema, en este caso una vez el sistema acaba de iniciarse. En este momento podemos comprobar que no se muestra la opción de mostrar opciones avanzadas, esto es debido a que no hay ningún modelo seleccionado que se encuentre cargado en memoria, además podemos ver en la imagen que junto a cada opción tenemos una serie de *tooltips*, indicados con un signo de interrogación, de forma que el usuario al mover el cursor encima de estos, se mostrará una descripción de la opción asociada al *tooltip*. También podemos comprobar en la esquina superior derecha un widget a modo de interruptor que nos permitirá cambiar la temática de la interfaz, variando entre un tono claro u oscuro.

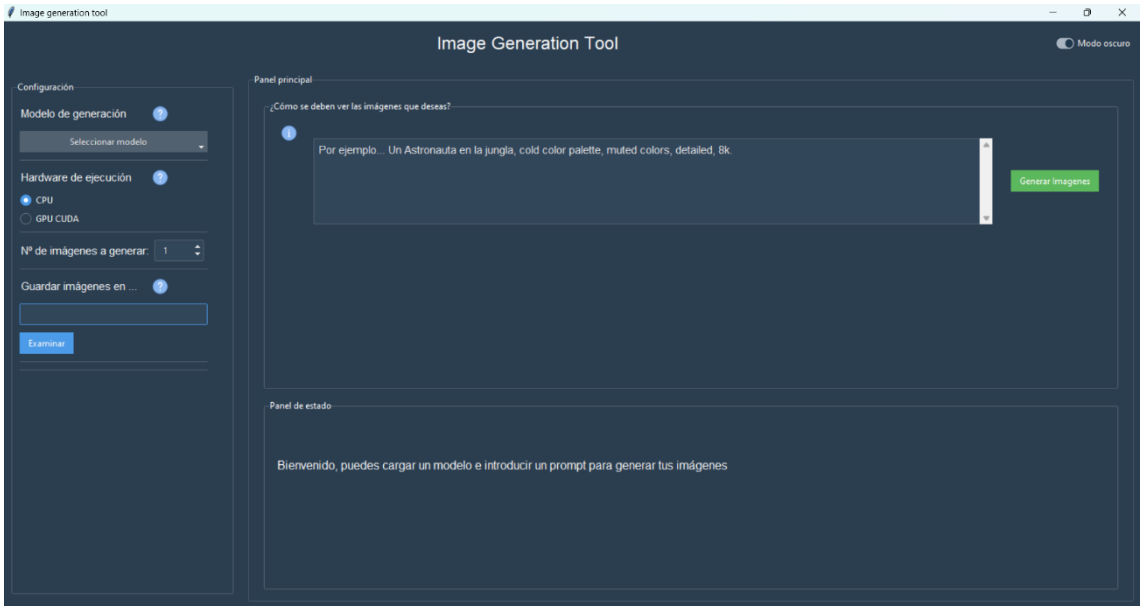


Figura 6.6 Interfaz gráfica al iniciar el sistema

En la siguiente imagen mostrada a continuación, podemos ver la interfaz de usuario con el tema claro tras haber desactivado el interruptor de la esquina superior derecha.

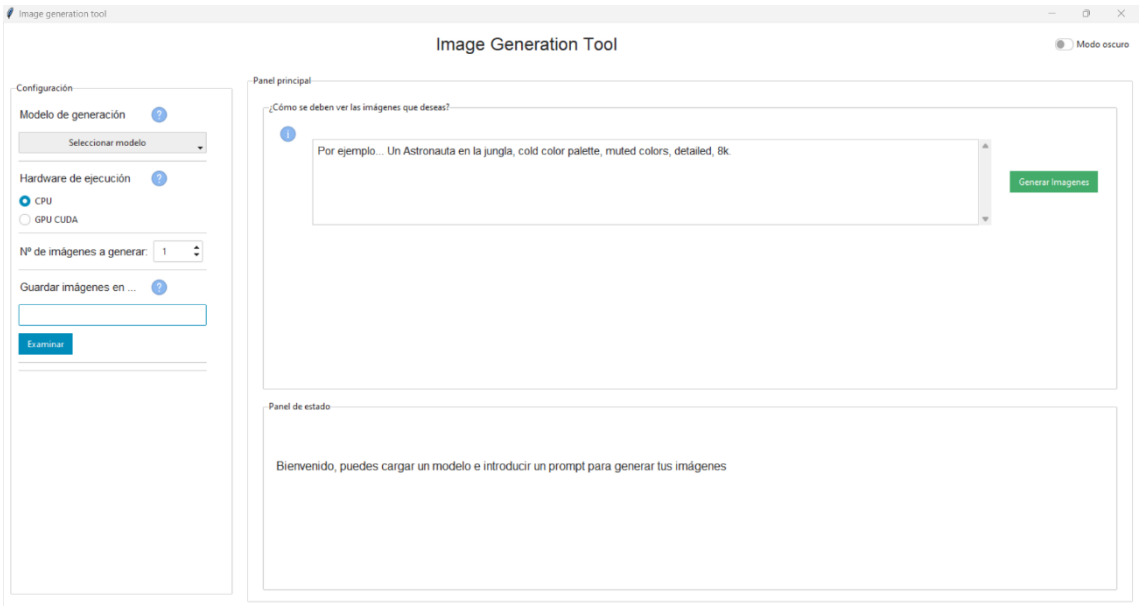


Figura 6.7 Interfaz gráfica del sistema con tema claro

A continuación, en la *figura 6.8* podemos ver un ejemplo de como se muestra un *tooltip* de una opción cuando el usuario mueve el cursor encima de este.

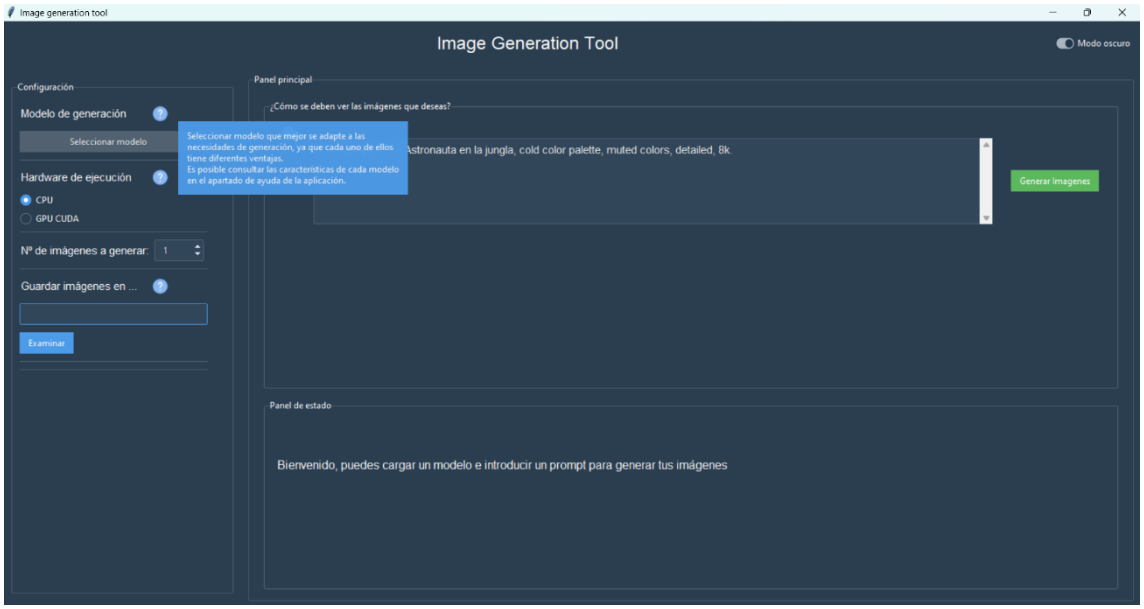


Figura 6.8 Interfaz gráfica del sistema con tooltip

A continuación, podemos ver en la siguiente figura el aspecto que tiene la interfaz una vez un usuario selecciona un modelo que no se ha cargado en memoria, como se puede apreciar en la imagen, el sistema mostrará el botón al usuario para cargar el modelo en memoria, este botón se encuentra justamente debajo del selector de modelo en la parte superior del panel de configuración, la única diferencia que existe al seleccionar un modelo que ya está cargado en memoria radica en que el botón de cargar modelo en memoria no será visible.

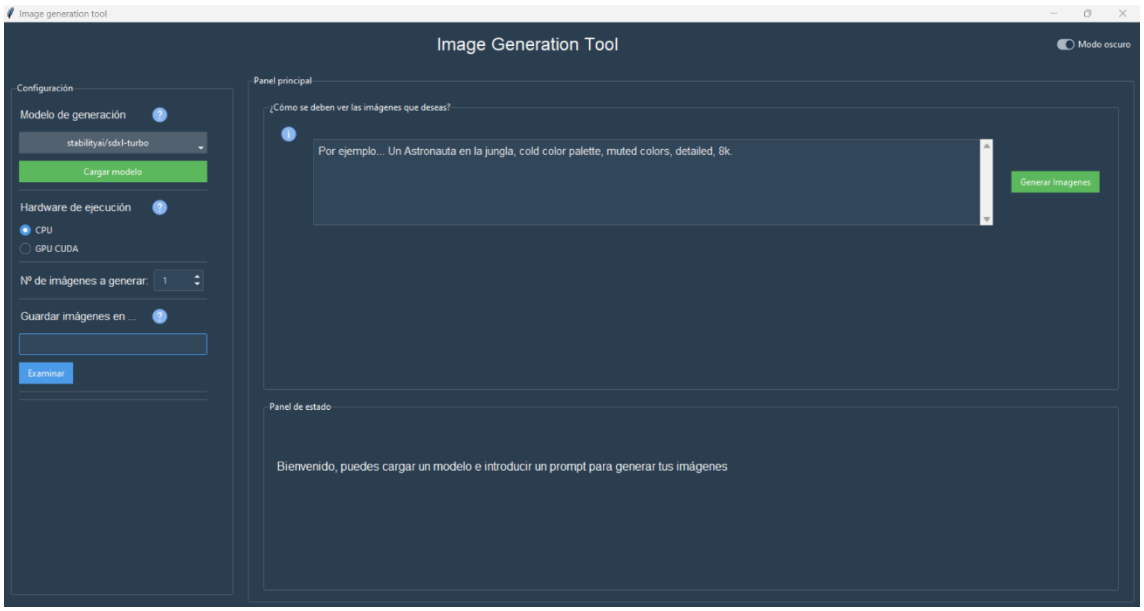


Figura 6.9 Interfaz gráfica del sistema con botón cargar modelo

Siguiendo con el flujo de la aplicación, en la siguiente imagen podemos ver, una vez el usuario ha hecho clic en el botón de cargar modelo, que este botón se verá desactivado, en el panel de estado se muestra una barra de progreso y además el sistema muestra un mensaje en la parte inferior derecha informando del inicio del proceso de carga del modelo.

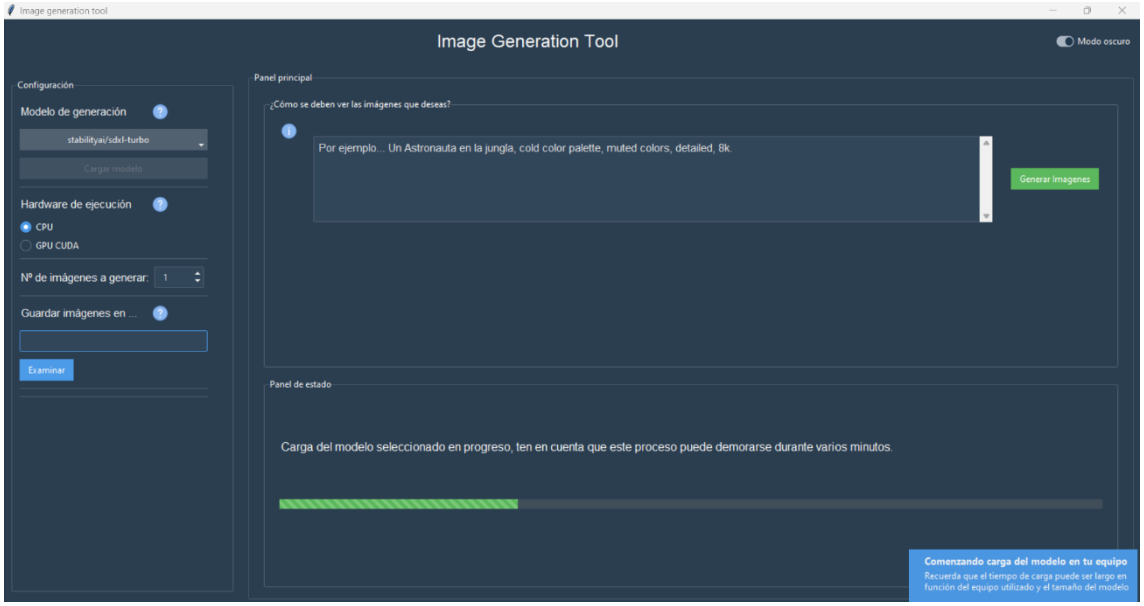


Figura 6.10 Interfaz de usuario con carga de modelo en proceso

Tras haberse completado el proceso de carga del modelo, podremos ver en la siguiente imagen que se ha mostrado el interruptor de mostrar opciones avanzadas (por defecto estará desactivado y el panel de opciones avanzadas no se mostrará por cuestiones de no sobrecargar la interfaz de usuario), en este caso activado para poder mostrar el panel de opciones avanzadas, también se puede ver que contará con un *scroll* que permitirá navegar por las opciones, además también podemos comprobar que a estas opciones se les ha añadido *tooltips* y que los *spinbox*

no permiten introducir directamente valores como en el caso del de número de imágenes, sino que el usuario deberá seleccionar el valor utilizando las flechas de este o hacer *scroll* con su ratón con el cursor apuntando a estos, de forma que solo podrá introducir valores que permita el sistema, se ha hecho de esta forma debido a que los rangos de valores son muy pequeños, por lo que mantiene una buena usabilidad y permite evitar errores al usuario al introducir valores inválidos. También podremos percibir que el botón de cargar modelo se encuentra oculto.

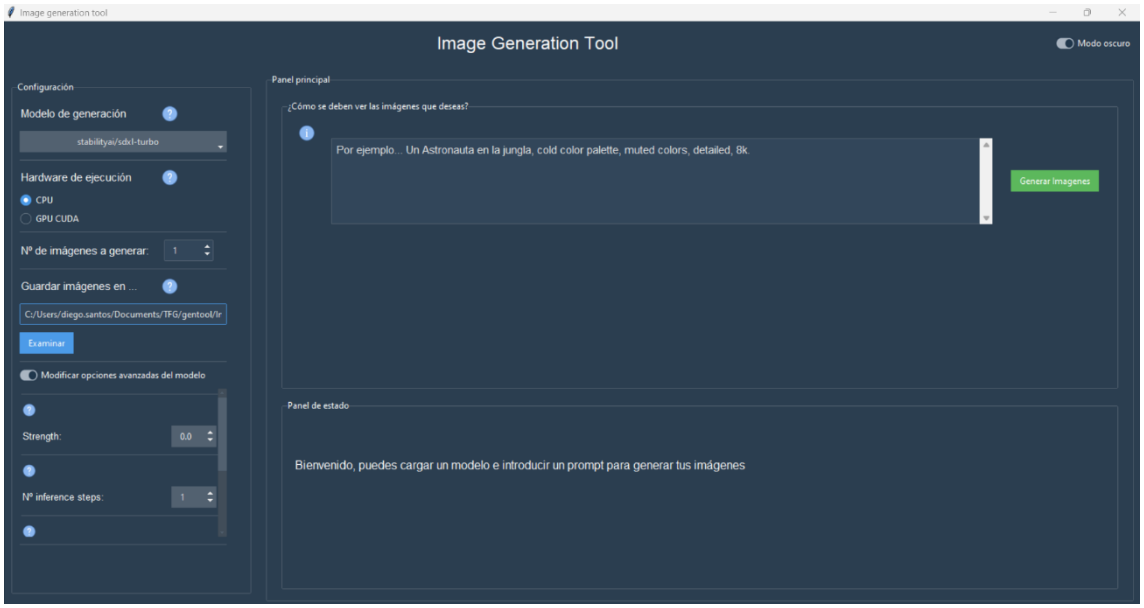


Figura 6.11 Interfaz de usuario con modelo cargado en memoria

A continuación, en la *figura 6.12* podemos ver un ejemplo de la realización de una validación del sistema, de forma que el sistema mostrará un mensaje de error, (Estos mensajes tanto de error como de información desaparecen al cabo de unos segundos) en la parte inferior derecha, debido a que como podemos ver en el selector de directorio, el usuario no ha introducido ninguna ruta a ningún directorio y además también podemos ver que los bordes de esta opción se encuentran marcados en color rojo.

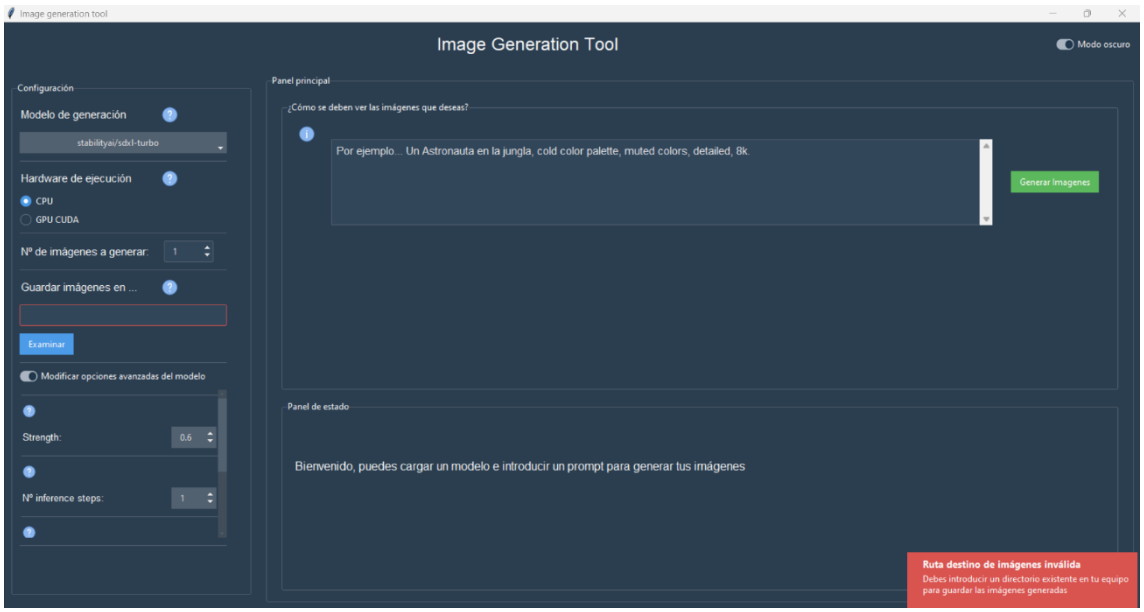


Figura 6.12 Interfaz de usuario con mensaje de error

Para continuar, se mostrará la interfaz una vez el usuario ha iniciado el proceso de generación, con todas las opciones introducidas de forma correcta, entre ellas podemos ver que se ha modificado el valor por defecto de la opción avanzada *strength* a 0.6, además de haber introducido un directorio del equipo del usuario, un modelo cargado en memoria, seleccionado un número de 2 imágenes a generar, la ejecución del modelo a realizarse en la *CPU* del equipo del usuario y una descripción textual de las imágenes a generar en el área de texto del panel principal.

También podemos ver en la siguiente imagen, en concreto en la *Figura 6.13* que el sistema muestra un mensaje en la parte inferior derecha en el que informa que se ha iniciado el proceso de generación y en el panel de estado podemos ver en la parte izquierda un *widget* que indica el número de imágenes a generar y cuántas ya se han generado, en este caso aún ninguna y en la parte derecha podemos ver que se muestra una barra de progreso indicando que el modelo se encuentra en ejecución.

En la siguiente imagen, en concreto en la *Figura 6.14*, podemos ver que se ha actualizado el contador de imágenes de la parte izquierda del panel de estado una vez se ha generado la primera imagen y además en la parte derecha del panel de estado se puede ver como la interfaz muestra al usuario esta primera imagen generada, de forma que se irán mostrando las imágenes de forma secuencial según se vayan generando.

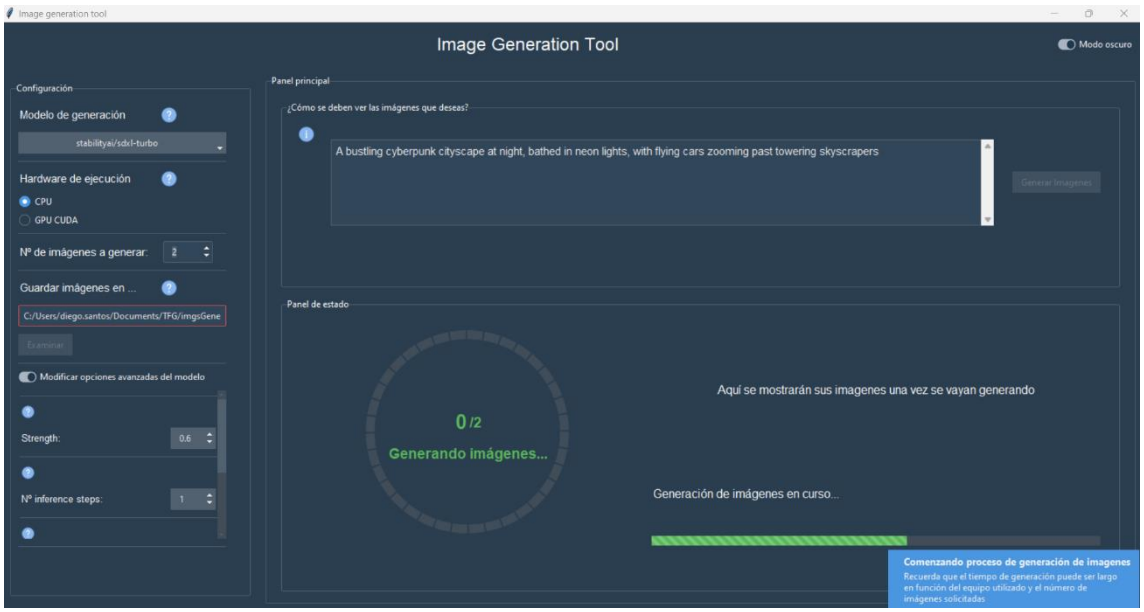


Figura 6.13 Interfaz de usuario con proceso de generación iniciado

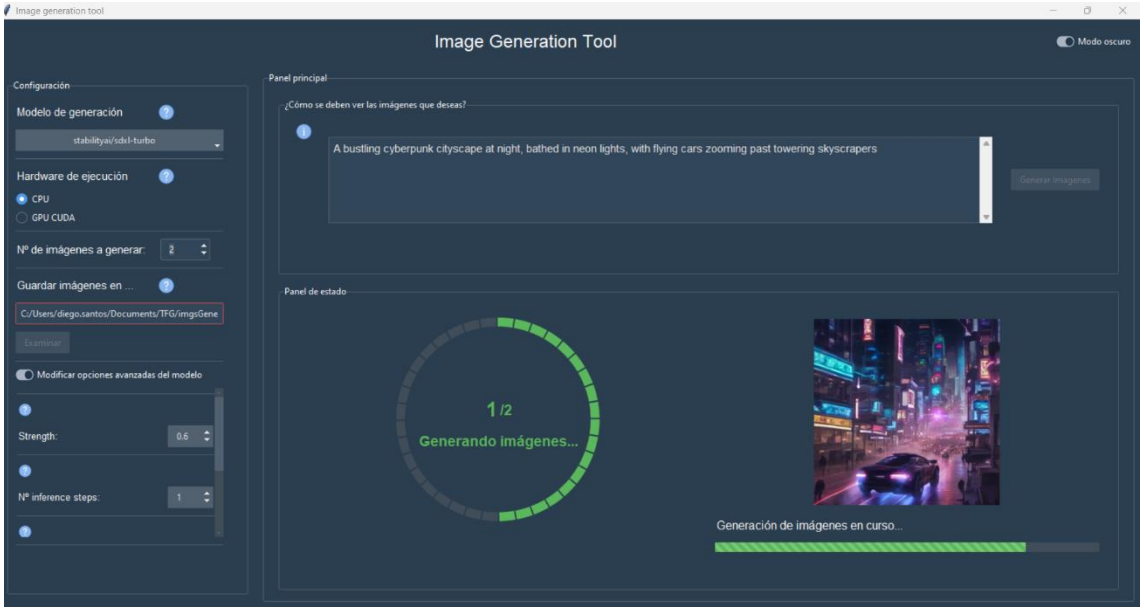


Figura 6.14 Interfaz de usuario con una imagen generada

Una vez se haya finalizado el proceso, el sistema mostrará un mensaje de éxito de forma similar a los mensajes mostrados en imágenes salvo por que tendrá un color de fondo verde e indicará que las imágenes se han generado con éxito. Tras esto la interfaz volverá al estado mostrado en la imagen representada en la *Figura 6.11.*

6.5 Especificación Técnica del Plan de Pruebas

En esta sección, se detallarán las pruebas explicadas previamente en la fase de análisis, de forma que se desarrollará la especificación técnica del plan de pruebas en las siguientes subsecciones que serán implementadas posteriormente en la fase de implementación de las pruebas. Por lo que a continuación, en estas subsecciones de explicará cada tipo de prueba a realizar y que objetivo tendrán y a que aspectos del sistema afectarán.

Estas pruebas se realizarán en un equipo con las siguientes características:

- Sistema operativo *Windows 11 Enterprise*.
- Procesador *13th Gen Intel(R) Core(TM) i7-1360P   2.20 GHz*.
- *RAM 32,0 GB* (*31,6 GB* usable).
- Unidad de almacenamiento *SSD 1TB*.

6.5.1 Pruebas Unitarias

En esta sección se expondrán las pruebas unitarias a realizar, para ello se han diseñado utilizando las pruebas de casos de uso expuestas en la fase de análisis en el apartado 5.8.5. Además de las establecidas en las siguientes tablas, durante la implementación de las pruebas unitarias es posible que se añadan más pruebas para complementar los escenarios expuestos en los casos de uso.

6.5.1.1 Caso de uso Generar imágenes.

Caso de Uso 1: Generar imágenes	
Prueba1.1	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria, introduce un texto descriptivo válido, indica 1 como número de imágenes a generar, *CPU* como dispositivo, un directorio existente e inicia el proceso de generación	La interfaz de usuario deshabilita el botón de cargar modelo en memoria y de generar imágenes. El controlador guarda en el directorio especificado por el usuario la imagen generada por el modelo con el nombre *"image1.png".*
Prueba1.2	**Resultado Esperado**
El usuario intenta generar imágenes con un modelo no cargado en memoria seleccionado.	El controlador valida que no existe un modelo cargado en memoria, por lo que no inicia el proceso de generación. La interfaz de usuario muestra al usuario un mensaje de error.
Prueba1.3	**Resultado Esperado**
El usuario intenta generar imágenes con un texto descriptivo vacío.	El controlador valida que el texto introducido es vacío, por lo que no inicia el proceso de generación. La interfaz de usuario muestra al usuario un mensaje de error.
Prueba1.4	**Resultado Esperado**

El usuario intenta generar imágenes con una *GPU*, pero el equipo del usuario no tiene una *GPU* instalada.	El controlador valida que el equipo del usuario no tiene disponible una *GPU CUDA*. La interfaz de usuario mostrará un mensaje al usuario indicando que no se ha encontrado una *GPU CUDA*. El controlador iniciará el proceso de generación en la *CPU* y la interfaz de usuario informará al usuario de que se iniciará el proceso en la *CPU*.
Prueba1.5	**Resultado Esperado**
El usuario introduce un número de imágenes fuera del rango válido establecido por el sistema.	La interfaz de usuario valida que el número de imágenes está fuera de rango, por lo que no inicia el proceso de generación y muestra al usuario un mensaje de error. Además, establecerá el valor 1 como número de imágenes.
Prueba1.6	**Resultado Esperado**
El usuario introduce un directorio no existente en su equipo e intenta generar imágenes.	El controlador valida que el directorio introducido es inválido, por lo que no inicia el proceso de generación. La interfaz de usuario muestra al usuario un mensaje de error.

Figura 6.15 Pruebas unitarias caso de uso generar imágenes

6.5.1.2 Caso de uso Cargar modelo en memoria.

Caso de Uso 2: Cargar modelo en memoria	
Prueba2.1	**Resultado Esperado**
El usuario selecciona un modelo no cargado en memoria y lo carga en memoria.	El controlador valida que el modelo no se encuentra cargado en memoria. La interfaz de usuario muestra el botón de cargar modelo. El controlador inicia el proceso de carga en memoria del modelo seleccionado. La interfaz de usuario deshabilita el botón de cargar modelo. Una vez terminado el proceso el modelo se encontrará cargado en memoria La interfaz ocultará el botón de cargar modelo en memoria.
Prueba2.2	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria e intenta cargarlo en memoria.	El controlador valida que el modelo seleccionado ya se encuentra cargado en memoria. La interfaz de usuario oculta el botón de cargar modelo en memoria. La interfaz de usuario muestra el *toggle* de mostrar opciones avanzadas.

Figura 6.16 Pruebas unitarias caso de uso cargar modelo en memoria

6.5.1.3 Caso de uso Modificar opciones avanzadas.

Caso de Uso 3: Modificar opciones avanzadas	
Prueba3.1	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria y modifica el valor de las opciones avanzadas, tras esto	El controlador valida que el modelo seleccionado ya se encuentra cargado en memoria. La interfaz de usuario muestra el *toggle* de mostrar opciones avanzadas.

inicia el proceso de generación.	El controlador inicia el proceso de generación indicando al modelo las opciones avanzadas establecidas por el usuario.
Prueba3.2	**Resultado Esperado**
El usuario selecciona un modelo cargado en memoria y modifica las opciones avanzadas, tras esto selecciona otro modelo y vuelve a seleccionar el modelo del que había modificado sus opciones.	El controlador valida que el modelo seleccionado ya se encuentra cargado en memoria. La interfaz de usuario muestra el *toggle* de mostrar opciones avanzadas. La interfaz establece los valores por defecto en las opciones avanzadas del modelo seleccionado primeramente por el usuario, del cual había modificado sus opciones avanzadas.
Prueba3.3	**Resultado Esperado**
El usuario selecciona un modelo que no ha sido cargado en memoria.	El controlador valida que el modelo seleccionado no se encuentra cargado en memoria. La interfaz de usuario no muestra/oculta el *toggle* de mostrar opciones avanzadas y el panel de opciones avanzadas.

Figura 6.17 Pruebas unitarias caso de uso modificar opciones avanzadas

6.5.2 Pruebas de Integración y del Sistema

Para realizar las pruebas de integración y del sistema se utilizarán las pruebas unitarias especificadas previamente, mediante estos se podrá verificar que los subsistemas se están comunicando y mantienen las relaciones esperadas entre estos.

6.5.3 Pruebas de Usabilidad y Accesibilidad

Este tipo de pruebas serán diseñadas de forma que nos permitan medir el nivel de satisfacción de los usuarios finales del sistema, para ello se evaluará el funcionamiento real del sistema con el objetivo de mejorar la experiencia del usuario utilizando el sistema, mejorando la calidad de interacción y reduciendo el tiempo necesario para poder realizar las diferentes tareas ofrecidas por el sistema.

A continuación, en las siguientes subsecciones se detallarán que pruebas se realizarán para poder conseguir los objetivos mencionados previamente.

6.5.3.1 Diseño de Cuestionarios

A continuación, se detallarán los cuestionarios elaborados con objetivo de medir el grado de satisfacción en cuanto a la experiencia utilizando el sistema de los diferentes usuarios. En las siguientes subsecciones se describirán también los diferentes tipos de cuestionarios elaborados y los objetivos buscados con cada uno de estos tipos de cuestionario.

6.5.3.1.1 Preguntas de carácter general

Las preguntas del siguiente cuestionario están dirigidas a conocer el nivel de conocimientos informáticos, así como las competencias del usuario final.

¿Usa un ordenador frecuentemente?
1. Todos los días 2. Varias veces a la semana 3. Ocasionalmente 4. Nunca o casi nunca
¿Qué tipo de actividades realiza con el ordenador?
1. Es parte de mi trabajo o profesión 2. Lo uso básicamente para ocio 3. Solo empleo aplicaciones estilo Office 4. Únicamente leo el correo y navego ocasionalmente
¿Ha usado alguna vez software como el de esta prueba?
1. Sí, he empleado software similar 2. No, aunque si empleo otros programas que me ayudan a realizar tareas similares 3. No, nunca
¿Qué busca Vd. Principalmente en un programa?
1. Que sea fácil de usar 2. Que sea intuitivo 3. Que sea rápido 4. Que tenga todas las funciones necesarias

Figura 6.18 Cuestionario de preguntas de carácter general

6.5.3.1.2 Actividades guiadas

El contenido del siguiente cuestionario tiene como objetivo conocer las ventajas y desventajas de nuestra aplicación, a través de las funcionalidades principales del sistema.

Cargar un modelo en memoria
Cosas que me gustaron: Cosas que me gustaría que se mejoraran:
Configurar las opciones básicas y generar imágenes
Cosas que me gustaron: Cosas que me gustaría que se mejoraran:
Modificar opciones avanzadas y generar imágenes

Cosas que me gustaron: Cosas que me gustaría que se mejoraran:

Figura 6.19 Cuestionario de actividades guiadas

6.5.3.1.3 Preguntas cortas sobre la aplicación

Facilidad de Uso	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Sabe dónde está dentro de la aplicación?				
¿Existe ayuda para las funciones en caso de que tenga dudas?				
¿Le resulta sencillo el uso de la aplicación?				
Funcionalidad	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Funciona cada tarea como Vd. Espera?				
¿El tiempo de respuesta de la aplicación es muy grande?				

Calidad del Interfaz				
Aspectos gráficos	**Muy Adecuado**	**Adecuado**	**Poco Adecuado**	**Nada Adecuado**
El tipo y tamaño de letra es				
Los iconos e imágenes usados son				
Los colores empleados son				
Diseño de la Interfaz		**Si**	**No**	**A veces**
¿Le resulta fácil de usar?				
¿El diseño de las pantallas es claro y atractivo?				
¿Cree que el programa está bien estructurado?				
Observaciones				
Cualquier comentario del usuario				

Figura 6.20 Cuestionario de preguntas cortas

6.5.3.1.4 Cuestionario para el responsable de las pruebas

Aspecto Observado	**Notas**
El usuario comienza a trabajar de forma rápida por las tareas.	
Tiempo en realizar cada tarea	
Errores leves cometidos.	
Errores graves cometidos.	
El proceso para realizar la generación de imágenes es entendido de forma intuitiva por el usuario.	
El usuario entiende de forma clara las descripciones incluidas en los tooltips de las opciones.	

El usuario entiende como corregir los errores cometidos de forma clara una vez lee los mensajes de error mostrados por el sistema.	

Figura 6.21 Cuestionario para el responsable de las pruebas

6.5.4 Pruebas de Rendimiento

Para realizar las pruebas de rendimiento, como se ha explicado en la fase de análisis se realizarán una serie de pruebas de las funcionalidades principales del sistema de forma manual, en las que se medirán los tiempos y consumo de recursos de estas, pero en cuanto a todo esto, debemos tener en cuenta que los procesos de generación de imágenes son muy dependientes del *hardware* en el que se ejecuten, por lo que estas medidas podrán variar mucho en función del equipo que se utilice para ejecutar los modelos.

Por lo tanto, se realizarán estas pruebas de forma que se tenga como referencia el rendimiento del sistema para un equipo con las características mencionadas al inicio de este capítulo.

Como resumen de lo comentado en esta subsección se realizarán principalmente pruebas para comprobar el rendimiento del sistema en cuanto a la carga en memoria y ejecución de los diferentes modelos del sistema, todo esto se verá detallado en tablas en la documentación de la fase de implementación de las pruebas, estas tablas tendrán la forma mostrada a continuación.

Modelo	**Prueba**	**Tiempo ejecución (s)**	**Uso de memoria (MB)**	**% Uso de CPU.**
Stable-diffusion-2	Cargar modelo			
Stable-diffusion-2	Generar imagen			
Stable-diffusion-1.5	Cargar modelo			
Stable-diffusion-1.5	Generar imagen			
Stable-diffusion-XL	Cargar modelo			
Stable-diffusion-XL	Generar imagen			
Stable-diffusion-XL-Turbo	Cargar modelo			
Stable-diffusion-XL-Turbo	Generar imagen			

Figura 6.22 Pruebas de rendimiento del sistema

Capítulo 7. Implementación del Sistema

A lo largo de este apartado se explicarán los diferentes aspectos relacionados con la implementación del sistema, entre estos aspectos se describirán los estándares y normas seguidos durante el desarrollo, los lenguajes de programación y librerías utilizadas para llevar a cabo la implementación del sistema, así como las herramientas y programas que se han utilizado durante esta etapa y finalmente se detallarán otros aspectos relacionados con la creación del sistema, como los problemas encontrados durante el desarrollo y la descripción detallada de las clases implementadas.

7.1 Estándares y Normas Seguidos

Durante la etapa de desarrollo de la herramienta se ha seguido la guía de estilo para escribir código *Python* conocida como *PEP8* [29], ampliamente reconocida y aplicada por la comunidad de este lenguaje de programación.

Esta guía de estilo define una serie de convenciones y normas de estilo para la escritura de código *Python* como la nomenclatura de funciones y variables u organización de importaciones, entre otros. Lo cual permitirá que adhiriéndonos a esta guía nuestro código será más legible y fácil de mantener tanto para el autor como para otros colaboradores que puedan trabajar en el desarrollo del código.

Para la validación de que el código desarrollado en el proyecto cumple con esta guía de estilo, se ha configurado la extensión *Pylance [30]* para aplicar un formateador que cumpla con esta guía de estilo, conocido como *autopep8* [31]. Lo cual permitirá aplicar este formato de código establecido por la guía, de forma automatizada durante el desarrollo de la herramienta.

7.2 Lenguajes de Programación

En este apartado se enumeran los lenguajes de programación utilizados para llevar a cabo la implementación del **sistema,** así como las librerías utilizadas, además se incluirá una descripción para cada lenguaje y librerías enumeradas en el apartado.

7.2.1 Python

El lenguaje utilizado para la implementación del proyecto ha sido *Python*, concretamente se ha utilizado la versión 3.12 *Python [32]*. *Python* es un lenguaje de alto nivel ampliamente utilizado en una gran variedad de campos, especialmente en el campo de la inteligencia artificial y aprendizaje automático. Es conocido por tener una sintaxis clara y legible, por lo que es sencillo de aprender y de usar. Además, es un lenguaje multiparadigma y multipropósito por lo que será

útil tanto para crear lógica de negocio como para crear interfaces de usuario entre otras múltiples posibilidades. También es conocido por tener una gran comunidad de desarrolladores y un gran número de bibliotecas y *frameworks*, incluidas las específicamente diseñadas para el procesamiento de imágenes y la inteligencia artificial, lo que brindará al proyecto acceso a poderosas herramientas probadas que se podrán integrar en el proyecto sin la necesidad de desarrollarlas desde cero. Por lo tanto, por estas razones este lenguaje es el idóneo para la implementación del proyecto y cualquier proyecto relacionado con la Inteligencia artificial.

7.2.1.1 Diffusers

Diffusers se trata de una librería de *Python* desarrollada por *Hugging Face [10]*, indicada para facilitar el trabajo con modelos generativos de difusión utilizados principalmente en la generación de imágenes, aunque también aplicables a otros tipos de datos como audio, vídeo o texto. Esta librería es muy interesante para investigadores y desarrolladores que tengan por objetivo trabajar en la generación de contenido de forma creativa a través de la inteligencia artificial, pudiendo acceder a modelos generativos avanzados sin la necesidad de tener unos conocimientos profundos sobre este tipo de modelos. Entre las principales características podemos destacar el soporte para una gran variedad de modelos de difusión, entrenamiento de modelos de difusión y generación de contenido a partir de modelos pre-entrenados, acceso a una amplia colección de modelos pre-entrenados, compatibilidad con aceleración por *GPU* y *TPU*, integración con la infraestructura de *Hugging Face* e integración con interfaces de alto nivel para usuarios sin experiencia avanzada.

7.2.1.2 PyTorch

Pytorch [34] es una librería de código abierto desarrollada por *Facebook AI Research Lab* [14], para la realización de tareas de aprendizaje profundo, es ampliamente utilizada tanto en el ámbito de la investigación como en producción industrial para entrenar redes neuronales y realizar tareas complejas de aprendizaje automático. Esta librería es conocida por su flexibilidad, además nos permite definir grafos de forma dinámica, soporte para *GPU*, un sistema automático de diferenciación y una gran comunidad con alta actividad con capacidad de proveer modelos pre-entrenados y diferentes recursos.

7.2.1.3 Tkinter

Una de las librerías utilizadas para la creación de la interfaz de usuario de la aplicación ha sido *Tkinter* [35], la cual se trata de la librería estándar de *Python* para la creación de interfaces gráficas de usuario. Esta librería nos permitirá desarrollar una aplicación con ventanas, menús, botones y demás widgets que se utilizan en la mayoría de las aplicaciones de forma simple, pero con amplias posibilidades, por lo que nos será ideal para el desarrollo de aplicaciones de escritorio con *Python* de una forma rápida y eficaz. Entre sus características a destacar podemos encontrar su simplicidad en cuanto a su forma de uso y accesibilidad, una gran variedad de *widgets*, compatible con los principales sistemas operativos como *MacOS*, *Windows* o *Linux*, no

requiere instalaciones adicionales ya que está integrada con *Python* y manejo de los diferentes eventos producidos por el usuario.

7.2.1.4 Venv

Durante el desarrollo de la herramienta se ha utilizado el módulo *venv* de *Python* para crear un entorno virtual de desarrollo[12]. Este módulo permite la creación de este entorno virtual cuyo cometido es crear un espacio aislado con la instalación de *Python* seleccionada y sus diferentes dependencias, de forma que podremos realizar el desarrollo en un entorno independiente y aislado de nuestro entorno global o de otros proyectos en los que se estén trabajando en la misma máquina. Por lo que este módulo resulta esencial para evitar problemas de compatibilidad y mantener los diferentes entornos de desarrollo bajo control, especialmente en los casos en los que se manejen varios proyectos. De este módulo se pueden destacar varias características, para empezar, como se ha comentado previamente, la capacidad de aislamiento de dependencias, su facilidad de uso gracias a la posibilidad de la creación de entornos virtuales mediante comandos nativos de *Python*, la facilidad de portabilidad de proyectos entre sistemas asegurando la consistencia de versiones y dependencias, la gestión de versiones y su ligereza, teniendo en cuenta que no es necesario instalar *software* adicional ya que este módulo viene incluido con la biblioteca estándar de *Python*.

7.2.1.5 Ttkbootstrap

Esta librería denominada *ttkbootstrap* [13], ha sido utilizada para el desarrollo de la interfaz de usuario de la herramienta, se trata de una librería que extiende de *Tkinter* y *ttk* [36], cuyo objetivo es proporcionar una interfaz gráfica de usuario moderna inspirada en Bootstrap, lo que nos proporcionará herramientas para desarrollar una interfaz con apariencias actuales sin la necesidad de desarrollar estilos y componentes complejos desde cero, por lo que resulta de gran utilidad cuando lo que se busca es mejorar la apariencia de las interfaces de usuario *Tkinter* sin renunciar a la facilidad de uso del desarrollo de interfaces de usuario con *Python*. Las principales características a destacar de esta librería son la capacidad de utilizar un conjunto de temas prediseñados con estilos atractivos, la completa compatibilidad con *Tkinter*, la capacidad de utilizar colores y tipografías personalizadas en diferentes componentes gráficos, proporcionar una interfaz de usuario consistente y profesional en aplicaciones con interfaz usuario y la facilidad de uso, ya que mantiene la simplicidad de la librería *Tkinter* con unos pequeños cambios para aplicar estilos a los diferentes componentes.

7.3 Herramientas y Programas Usados para el Desarrollo

A continuación, se enumeran las diferentes herramientas y sistemas adicionales que se han utilizado para llevar a cabo el desarrollo del proyecto.

7.3.1 Visual Studio Code

El *IDE* utilizado para el desarrollo del código de la aplicación ha sido *Visual Studio Code* [37], se trata de una herramienta desarrollada por *Microsoft* y es uno de los *IDE* más populares en el mercado gracias a su ligereza y versatilidad, además es compatible con los principales sistemas operativos y la mayoría de los lenguajes de programación existentes. Las características más relevantes de este *IDE* son, por ejemplo, el soporte a una cantidad muy extendida de extensiones, depuración integrada, integración directa con el sistema de control de versiones *Git*, terminal integrada en el editor o autocompletado inteligente.

7.3.2 Git

Para llevar a cabo el control de versiones del código durante la etapa de desarrollo del *software* se ha utilizado el sistema *Git* [38], el cual se trata de un *software open-source* de control de versiones desarrollado por Linus Torvalds en 2005, actualmente es el software más extendido para este cometido. Entre sus principales características se encuentran, el control de versiones distribuido, seguimiento detallado de modificaciones mediante el historial de cambios, trabajo en paralelo mediante el manejo de ramas o la colaboración mediante fusión de cambios y resolución de conflictos.

7.3.3 Google Collab

Para realizar la evaluación e investigación de los modelos de generación de imágenes a incluir en la aplicación, se ha optado por utilizar la plataforma *Google Collab* [39], la cual se trata de una herramienta proporcionada por Google con versión una gratuita, que nos permitirá escribir y ejecutar código *Python* en el navegador, donde además nos proporcionará recursos de procesamiento en la nube incluyendo *TPUs* o *GPUs*. Por lo tanto, se trata de una herramienta ideal para realizar tareas de desarrollo en el ámbito de la inteligencia artificial, análisis de datos o la realización de pruebas y prototipos sin la necesidad de ser propietario de *hardware* propio.

7.3.4 Microsoft Project

Para llevar a cabo la planificación de las actividades del proyecto, así como el presupuesto de este, se ha optado por elegir la herramienta *Microsoft Project* [40], se trata de un *software* desarrollado por *Microsoft* cuyo cometido es la planificación, ejecución y seguimiento de proyectos de manera eficiente y ordenada, es ampliamente utilizada en la planificación y ejecución de proyectos complejos, en especial en sectores como *IT*, la construcción o la manufactura. Entre las principales características se encuentran la planificación de tareas, la asignación de recursos, colaboración entre equipos, análisis financiero e integración con otras herramientas de *Microsoft*.

7.3.5 Microsoft Excel

Para la realización de los presupuestos del proyecto, se ha utilizado la herramienta *Microsoft Excel* [41], se trata de un software de hoja de cálculo desarrollado por *Microsoft* perteneciente a la *suite Microsoft Office*, esta herramienta es utilizada ampliamente para el análisis y manipulación de datos, establecida y extendida en la mayoría de las industrias para la gestión de datos, análisis financiero, planificación y presentación de informes. Entre las principales características que se encuentran en esta herramienta podemos destacar la manipulación de hojas de cálculo, fórmulas y funciones avanzadas, creación de gráficos y visualizaciones, tablas dinámicas o automatización con *macros*.

7.4 Creación del Sistema

En esta sección se explicarán los principales aspectos sobre el desarrollo del sistema.

7.4.1 Problemas Encontrados

A continuación, se enumeran los diferentes problemas que se han encontrado durante el desarrollo del sistema y la solución aplicada a estos.

7.4.1.1 Bloqueos en la interfaz de usuario

Uno de los problemas a destacar durante el desarrollo de la herramienta se corresponde con la congelación de la interfaz de usuario durante el proceso de cargar un modelo en memoria y durante el proceso de la generación de imágenes, debido a que no se había tenido en cuenta que estos procesos se ejecutarían en el hilo principal, donde además se ejecuta el proceso correspondiente con el bucle principal en el que se corre la interfaz de usuario y por lo tanto esta se quedaba en un estado bloqueado y no era posible mostrar ningún mensaje ni actualización de estado de la herramienta durante estos procesos, lo cual es un problema para la usabilidad de la aplicación y para la necesidad de mostrar al usuario el estado de carga de los modelos o el estado en el que se encuentra el proceso de generación de imágenes. La causa de este bloqueo de la interfaz fue debido a que ambos procesos que causaban esta problemática consumen una gran cantidad de tiempo desde el comienzo de su ejecución hasta la finalización de esta.

Para dar solución a esta problemática, se ha optado por lanzar estos procesos en otro hilo diferente al hilo principal, donde se ejecuta la interfaz de usuario de la herramienta, de forma que la interfaz ya no se bloquearía y sería posible que el usuario interaccione con esta durante ambos procesos, además de ser posible que los procesos notifiquen en qué estado se encuentra para poder ir mostrando este estado al usuario durante la ejecución de estos a través de la interfaz de usuario. Para implementar esta solución, se ha hecho uso de la librería *threading* [41], la cual nos permite gestionar la ejecución de los hilos en nuestra herramienta.

7.4.1.2 Diseño orientado a objetos de modelos generativos

Otro de los problemas que han surgido durante el desarrollo de la aplicación fue debido a las diferencias de implementación de cada modelo generativo a ser incluido en la aplicación, en cuanto a la forma de ser cargado en memoria y la operación de generación de imágenes, además de la probable necesidad futura de incluir nuevos modelos que pudieran tener diferentes implementaciones, teniendo en cuenta que para conseguir las funcionalidades de cargar en memoria un modelo y generar imágenes mediante diferentes modelos es necesario que todos los modelos realizasen estas funcionalidades de forma uniforme para poder integrarse con nuestra interfaz de usuario y para que no resulte un gran esfuerzo incluir nuevos modelos. También durante el desarrollo de estas funcionalidades se encontró que la implementación de estas era común en su mayoría salvo ciertas discrepancias en función de que modelo se tratase,

entre ellas diferentes parámetros en la forma de cargar en memoria el modelo o de ejecutar la generación de imágenes.

Para resolver esta problemática, cada modelo incluido en la herramienta, cuya implementación es obtenida a través de librerías externas ha sido encapsulado dentro de una clase y para obtener esta uniformidad en las operaciones a realizar por cada modelo se ha utilizado una clase abstracta que será implementada por cada clase en la que se encapsula cada modelo, de forma que podemos obtener esta uniformidad de operaciones entre todos los modelos, gracias a este enfoque orientado a objetos.

Además, para resolver la problemática de que se comparte una gran cantidad de código en estas operaciones salvo pequeñas discrepancias dependientes de cada modelo, se ha utilizado este enfoque orientado a objetos para establecer la funcionalidad común en la clase abstracta y estas pequeñas discrepancias en cada implementación de esta en las clases concretas de cada modelo, en concreto aplicando un enfoque basado en el patrón de diseño conocido como *template method* [42], este enfoque podemos verlo reflejado en el diseño de clases incluido en el apartado Diagrama de Clases.

7.4.1.3 Generación de archivo ejecutable

Esta sección está relacionada con el problema que se ha encontrado tras acabar el desarrollo del sistema, más concretamente, una vez desarrollado el sistema el objetivo era generar un archivo ejecutable utilizando herramientas que nos permitan realizar esto de forma automatizada, de forma que la herramienta desarrollada en el proyecto pueda ser distribuida mediante este ejecutable.

El problema que ha ocurrido se relaciona con problemas en las librerías utilizadas en el sistema, en concreto a librería *diffusers*, debido a que esta librería tiene una gran complejidad y es dependiente de otras librerías con alta complejidad como la librería *torch*, al crear el ejecutable con diferentes herramientas e intentar arrancar el sistema utilizando este ejecutable, el sistema no es capaz de arrancar y se muestra un error en nuestro equipo. Este error es debido a que las herramientas no son capaces de importar las dependencias de la librería correctamente, ya que esta librería contiene otras dependencias de otras librerías complejas y este tipo de herramientas no son capaces de manejarlas correctamente de forma automática.

Para intentar generar el ejecutable se han utilizado las librerías *PyInstaller[43]* , *cx-Freeze* [44] y *auto-py-to-exe [45]*. Llegando al mismo error con todas estas librerías, para intentar remediarlo se ha intentado probar con diferentes configuraciones en cada una de estas herramientas, pero no ha sido posible resolver el error, además se han consultado multitud de foros de la comunidad *Python* para buscar la forma de resolver el error, aunque ninguna de las soluciones propuestas ha servido como solución para el problema. Para acabar también se ha intentado probar la generación con un entorno limpio de dependencias e incluso actualizando todas las dependencias y librerías del proyecto a su última versión y el error ha persistido.

No se ha llegado a una solución durante la ejecución del presente proyecto, como solución propuesta se recomienda contactar con los desarrolladores de la librería, abrir un tema de debate en la comunidad de *Hugging Face* o abrir un *issue* en el propio repositorio de la librería.

7.4.2 Descripción Detallada de las Clases

Se podrá consultar la descripción detallada de las clases del proyecto en el archivo denominado *index.html*, el cual se encuentra contenido dentro de la carpeta "*docuPy*" del archivo adjunto del proyecto. Abriendo este archivo en el navegador, podremos navegar por la documentación de las diferentes clases principales del proyecto, la cuál ha sido generada con la herramienta *PyDoc* [25].

Capítulo 8. Desarrollo de las Pruebas

En este capítulo se presenta la implementación de las diferentes pruebas especificadas en el apartado 6.5.

8.1 Pruebas Unitarias

A continuación, se describen los resultados obtenidos tras la realización de las pruebas unitarias definidas en 6.5.1.

Caso de Uso 1: Generar imágenes		
Prueba1.1	**Resultado Esperado**	**Resultado obtenido**
El usuario selecciona un modelo cargado en memoria, introduce un texto descriptivo válido, indica 1 como número de imágenes a generar, *CPU* como dispositivo, un directorio existente e inicia el proceso de generación	La interfaz de usuario deshabilita el botón de cargar modelo en memoria y de generar imágenes. El controlador guarda en el directorio especificado por el usuario la imagen generada por el modelo con el nombre *"image1.png".*	Test OK. Se verifica que la interfaz deshabilita los botones, se verifica que la imagen es generada y guardada en el directorio especificado.
Prueba1.2	**Resultado Esperado**	**Resultado obtenido**
El usuario intenta generar imágenes con un modelo no cargado en memoria seleccionado.	El controlador valida que no existe un modelo cargado en memoria, por lo que no inicia el proceso de generación. La interfaz de usuario muestra al usuario un mensaje de error.	Test OK. Se verifica que no se inicia el proceso de generación de imágenes y la interfaz muestra un mensaje de error.
Prueba1.3	**Resultado Esperado**	**Resultado obtenido**
El usuario intenta generar imágenes con un texto descriptivo vacío.	El controlador valida que el texto introducido es vacío, por lo que no inicia el proceso de generación. La interfaz de usuario muestra al usuario un mensaje de error.	Test OK, ser verifica que no se inicia el proceso de generación de imágenes y la interfaz muestra un mensaje de error.
Prueba1.4	**Resultado Esperado**	**Resultado obtenido**
El usuario intenta generar imágenes con una *GPU*, pero el equipo del usuario no tiene una *GPU* instalada.	El controlador valida que el equipo del usuario no tiene disponible una *GPU CUDA*. La interfaz de usuario mostrará un mensaje al usuario indicando que no se ha encontrado una *GPU CUDA*. El controlador iniciará el proceso de generación en la *CPU* y la interfaz de	Test OK. Se verifica que se inicia el proceso de generación en la *CPU,* se verifica que la interfaz de usuario muestra el mensaje correspondiente.

	usuario informará al usuario de que se iniciará el proceso en la *CPU*.	
Prueba1.5	**Resultado Esperado**	**Resultado obtenido**
El usuario introduce un número de imágenes fuera del rango válido establecido por el sistema.	La interfaz de usuario valida que el número de imágenes está fuera de rango, por lo que no inicia el proceso de generación y muestra al usuario un mensaje de error. Además, establecerá el valor 1 como número de imágenes.	Test OK. Se verifica que no se inicia el proceso de generación de imágenes, muestra un mensaje de error y establece el valor 1 como numero de imágenes.
Prueba1.6	**Resultado Esperado**	**Resultado obtenido**
El usuario introduce un directorio no existente en su equipo e intenta generar imágenes.	El controlador valida que el directorio introducido es inválido, por lo que no inicia el proceso de generación. La interfaz de usuario muestra al usuario un mensaje de error.	Test OK. Se verifica que el proceso de generación de imágenes no se inicia y la interfaz de usuario muestra el mensaje de error.

Figura 8.1 Pruebas unitarias caso de uso generar imágenes

Caso de Uso 2: Cargar modelo en memoria		
Prueba2.1	**Resultado Esperado**	**Resultado obtenido**
El usuario selecciona un modelo no cargado en memoria y lo carga en memoria.	El controlador valida que el modelo no se encuentra cargado en memoria. La interfaz de usuario muestra el botón de cargar modelo. El controlador inicia el proceso de carga en memoria del modelo seleccionado. La interfaz de usuario deshabilita el botón de cargar modelo. Una vez terminado el proceso el modelo se encontrará cargado en memoria La interfaz ocultará el botón de cargar modelo en memoria.	Test OK. Se verifica que se muestra el botón de cargar modelo, se inicia el proceso de carga en memoria, se deshabilita el botón de cargar modelo, se carga en memoria el modelo y se oculta el botón al cargarse el modelo.
Prueba2.2	**Resultado Esperado**	**Resultado obtenido**
El usuario selecciona un modelo cargado en memoria e intenta cargarlo en memoria.	El controlador valida que el modelo seleccionado ya se encuentra cargado en memoria. La interfaz de usuario oculta el botón de cargar modelo en memoria. La interfaz de usuario muestra el *toggle* de mostrar opciones avanzadas.	Test OK. Se verifica que no se carga el modelo en memoria y la interfaz actúa como debería.

Figura 8.2 Pruebas unitarias caso de uso cargar modelo en memoria

Caso de Uso 3: Modificar opciones avanzadas		
Prueba3.1	**Resultado Esperado**	**Resultado obtenido**

El usuario selecciona un modelo cargado en memoria y modifica el valor de las opciones avanzadas, tras esto inicia el proceso de generación.	El controlador valida que el modelo seleccionado ya se encuentra cargado en memoria. La interfaz de usuario muestra el *toggle* de mostrar opciones avanzadas. El controlador inicia el proceso de generación indicando al modelo las opciones avanzadas establecidas por el usuario.	Test OK. Se verifica que se muestra el toggle, se verifica que se inicia el proceso de generación con las opciones indicadas
Prueba3.2	**Resultado Esperado**	**Resultado obtenido**
El usuario selecciona un modelo cargado en memoria y modifica las opciones avanzadas, tras esto selecciona otro modelo y vuelve a seleccionar el modelo del que había modificado sus opciones.	El controlador valida que el modelo seleccionado ya se encuentra cargado en memoria. La interfaz de usuario muestra el *toggle* de mostrar opciones avanzadas. La interfaz establece los valores por defecto en las opciones avanzadas del modelo seleccionado primeramente por el usuario, del cual había modificado sus opciones avanzadas.	Test OK. Se verifica que se muestra el toggle, se verifica que se muestran las opciones por defecto.
Prueba3.3	**Resultado Esperado**	**Resultado obtenido**
El usuario selecciona un modelo que no ha sido cargado en memoria.	El controlador valida que el modelo seleccionado no se encuentra cargado en memoria. La interfaz de usuario no muestra/oculta el *toggle* de mostrar opciones avanzadas y el panel de opciones avanzadas.	Test OK. Se verifica que no se muestra el panel de opciones avanzadas ni el toggle.

Figura 8.3 Pruebas unitarias caso de uso modificar opciones avanzadas

8.2 Pruebas de Usabilidad y Accesibilidad

A lo largo de este capítulo se detallarán los resultados obtenidos tras llevar a cabo las pruebas de usabilidad y accesibilidad descritas en 6.5.3.

8.2.1 Pruebas de Usabilidad

En esta sección se presentan los resultados obtenidos tras llevar a cabo las pruebas de usabilidad especificadas en 6.5.3. Para ello, se mostrarán los cuestionarios cumplimentados por 3 usuarios elegidos específicamente, de forma que cada uno de ellos tiene diferente perfil y experiencia con el uso de programas informáticos.

8.2.1.1 Resultados de preguntas de carácter general

En el siguiente cuestionario se marcará la opción elegida por cada usuario con el texto en negrita.

Usuario: Juan
¿Usa un ordenador frecuentemente?
5. Todos los días 6. Varias veces a la semana 7. Ocasionalmente 8. Nunca o casi nunca
¿Qué tipo de actividades realiza con el ordenador?
5. Es parte de mi trabajo o profesión 6. Lo uso básicamente para ocio 7. Solo empleo aplicaciones estilo Office 8. Únicamente leo el correo y navego ocasionalmente
¿Ha usado alguna vez software como el de esta prueba?
4. Sí, he empleado software similar 5. No, aunque si empleo otros programas que me ayudan a realizar tareas similares **6. No, nunca**
¿Qué busca Vd. Principalmente en un programa?
5. Que sea fácil de usar 6. Que sea intuitivo 7. Que sea rápido 8. Que tenga todas las funciones necesarias

Figura 8.4 Resultado cuestionario carácter general Juan

Usuario: Guillermo

¿Usa un ordenador frecuentemente?
9. Todos los días 10. Varias veces a la semana 11. Ocasionalmente 12. Nunca o casi nunca
¿Qué tipo de actividades realiza con el ordenador?
9. Es parte de mi trabajo o profesión 10. Lo uso básicamente para ocio 11. Solo empleo aplicaciones estilo Office 12. Únicamente leo el correo y navego ocasionalmente
¿Ha usado alguna vez software como el de esta prueba?
7. Sí, he empleado software similar 8. No, aunque si empleo otros programas que me ayudan a realizar tareas similares 9. No, nunca
¿Qué busca Vd. Principalmente en un programa?
9. Que sea fácil de usar 10. Que sea intuitivo 11. Que sea rápido **12. Que tenga todas las funciones necesarias**

Figura 8.5 Resultado cuestionario carácter general Guillermo

Usuario: Lucía
¿Usa un ordenador frecuentemente?
13. Todos los días 14. Varias veces a la semana **15. Ocasionalmente** 16. Nunca o casi nunca
¿Qué tipo de actividades realiza con el ordenador?
13. Es parte de mi trabajo o profesión 14. Lo uso básicamente para ocio **15. Solo empleo aplicaciones estilo Office** 16. Únicamente leo el correo y navego ocasionalmente
¿Ha usado alguna vez software como el de esta prueba?
10. Sí, he empleado software similar 11. No, aunque si empleo otros programas que me ayudan a realizar tareas similares **12. No, nunca**
¿Qué busca Vd. Principalmente en un programa?

13. Que sea fácil de usar
14. **Que sea intuitivo**
15. Que sea rápido
16. Que tenga todas las funciones necesarias

Figura 8.6 Resultado cuestionario carácter general Lucía

8.2.1.2 Resultados de actividades guiadas

A continuación, se muestran los resultados de los cuestionarios realizados por cada usuario que ha realizado las diferentes actividades guiadas.

Usuario: Juan
Cargar un modelo en memoria
Cosas que me gustaron: La aplicación es muy intuitiva por lo general. Cosas que me gustaría que se mejoraran: El diseño no es tan bueno como el de las aplicaciones que suelo usar.
Configurar las opciones básicas y generar imágenes
Cosas que me gustaron: Es muy clara la forma de ir configurando las opciones. Cosas que me gustaría que se mejoraran: No queda muy claro que modelo elegir si no tienes los suficientes conocimientos.
Modificar opciones avanzadas y generar imágenes
Cosas que me gustaron: Es de buena ayuda para mejorar el resultado final, sobre todo la opción de *negative prompt*, para la gente menos entendida. Cosas que me gustaría que se mejoraran: No se entiende fácilmente para que sirven todas las opciones avanzadas si no tienes un nivel de conocimientos suficiente.

Figura 8.7 Resultado actividades guiadas Juan

Usuario: Guillermo
Cargar un modelo en memoria
Cosas que me gustaron: El proceso es intuitivo gracias al diseño del botón y las instrucciones del panel principal. Cosas que me gustaría que se mejoraran: Los modelos podrían cargarse de forma automática la primera vez que se seleccionan, en el momento de darle al botón de generar imágenes.
Configurar las opciones básicas y generar imágenes
Cosas que me gustaron: El proceso es sencillo y claro.

Cosas que me gustaría que se mejoraran: Podrían actualizarse los mensajes del panel principal indicando las opciones que faltan por rellenar según se realiza el proceso de configurar las opciones.
Modificar opciones avanzadas y generar imágenes
Cosas que me gustaron: La herramienta muestra *tooltips* explicando la utilidad de cada opción y no da opción a seleccionar un valor erróneo. Cosas que me gustaría que se mejoraran: Los *spinbox* con rangos altos de números pueden tardar en configurarse si se quieren mover entre valores lejanos.

Figura 8.8 Resultado actividades guiadas Guillermo

Usuario: Lucía.
Cargar un modelo en memoria
Cosas que me gustaron: Se ve claramente cuando se empieza a cargar el modelo y cuando acaba por los mensajes y barra de carga de la aplicación. Cosas que me gustaría que se mejoraran: No queda muy claro que sea necesario darle al botón de cargar modelo para generar las imágenes.
Configurar las opciones básicas y generar imágenes
Cosas que me gustaron: Es intuitiva la forma de rellenar las opciones. Cosas que me gustaría que se mejoraran: Podría indicarse más claramente de alguna manera que es obligatorio rellenar las opciones antes de darle al botón de generar.
Modificar opciones avanzadas y generar imágenes
Cosas que me gustaron: Es intuitiva la forma de modificar las opciones y fácil de usar. Cosas que me gustaría que se mejoraran: Si no tienes conocimientos sobre IA no es muy útil sin investigar primero este tipo de opciones para una persona sin experiencia.

Figura 8.9 Resultado actividades guiadas Lucía

8.2.1.3 Resultados de preguntas cortas sobre la aplicación

En esta sección se presentan los resultados de las respuestas obtenidas de los usuarios en cuanto al cuestionario relacionado con las preguntas cortas sobre la aplicación.

Usuario: Juan				
Facilidad de Uso	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Sabe dónde está dentro de la aplicación?		X		

¿Existe ayuda para las funciones en caso de que tenga dudas?	X			
¿Le resulta sencillo el uso de la aplicación?		X		
Funcionalidad	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Funciona cada tarea como Vd. Espera?	X			
¿El tiempo de respuesta de la aplicación es muy grande?			X	
Calidad del Interfaz				
Aspectos gráficos	**Muy Adecuado**	**Adecuado**	**Poco Adecuado**	**Nada Adecuado**
El tipo y tamaño de letra es		X		
Los iconos e imágenes usados son		X		
Los colores empleados son			X	
Diseño de la Interfaz		**Si**	**No**	**A veces**
¿Le resulta fácil de usar?		X		
¿El diseño de las pantallas es claro y atractivo?				X
¿Cree que el programa está bien estructurado?		X		
Observaciones				
Cualquier comentario del usuario: N/A				

Figura 8.10 Resultados preguntas cortas Juan

Usuario: Guillermo				
Facilidad de Uso	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Sabe dónde está dentro de la aplicación?	X			
¿Existe ayuda para las funciones en caso de que tenga dudas?	X			
¿Le resulta sencillo el uso de la aplicación?	X			
Funcionalidad	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Funciona cada tarea como Vd. Espera?		X		
¿El tiempo de respuesta de la aplicación es muy grande?			X	
Calidad del Interfaz				
Aspectos gráficos	**Muy Adecuado**	**Adecuado**	**Poco Adecuado**	**Nada Adecuado**
El tipo y tamaño de letra es		X		
Los iconos e imágenes usados son		X		
Los colores empleados son		X		
Diseño de la Interfaz		**Si**	**No**	**A veces**
¿Le resulta fácil de usar?		X		
¿El diseño de las pantallas es claro y atractivo?				X
¿Cree que el programa está bien estructurado?		X		
Observaciones				
Cualquier comentario del usuario: La usabilidad de la aplicación mejoraría si los modelos se pudieran cargar de forma automática.				

Figura 8.11 Resultados preguntas cortas Guillermo

Usuario: Lucía.				
Facilidad de Uso	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Sabe dónde está dentro de la aplicación?	X			
¿Existe ayuda para las funciones en caso de que tenga dudas?	X			
¿Le resulta sencillo el uso de la aplicación?		X		
Funcionalidad	**Siempre**	**Frecuentemente**	**Ocasionalmente**	**Nunca**
¿Funciona cada tarea como Vd. Espera?		X		
¿El tiempo de respuesta de la aplicación es muy grande?			X	
Calidad del Interfaz				
Aspectos gráficos	**Muy Adecuado**	**Adecuado**	**Poco Adecuado**	**Nada Adecuado**
El tipo y tamaño de letra es	X			
Los iconos e imágenes usados son		X		
Los colores empleados son			X	
Diseño de la Interfaz		**Si**	**No**	**A veces**
¿Le resulta fácil de usar?				X
¿El diseño de las pantallas es claro y atractivo?		X		
¿Cree que el programa está bien estructurado?		X		
Observaciones				
Cualquier comentario del usuario: N/A				

Figura 8.12 Resultados preguntas cortas Lucía

8.2.1.4 Resultados de cuestionario para el responsable de las pruebas

A continuación, se muestra el cuestionario realizado por la persona encargada de llevar a cabo las diferentes pruebas con los usuarios.

Aspecto Observado	**Notas**
El usuario comienza a trabajar de forma rápida por las tareas.	Todos los usuarios coinciden en que la manera de trabajar y opciones a seleccionar son intuitivas.
Tiempo en realizar cada tarea	Todos los usuarios realizan las tareas en un tiempo considerablemente rápido.
Errores leves cometidos.	No ha habido errores leves, pero algunos usuarios se saltan el paso de cargar el modelo en memoria por prestar poca atención a las instrucciones mostradas en el panel principal al iniciar la herramienta.
Errores graves cometidos.	No ha habido errores graves.
El proceso para realizar la generación de imágenes es entendido de forma intuitiva por el usuario.	El proceso de configurar las opciones se ha realizado de forma muy directa por los usuarios, aunque algunos no han cargado el modelo en memoria previamente, por lo que hasta que no han leído el mensaje de error no se dan cuenta de que es necesario cargarlo en memoria utilizando el botón de cargar modelo en memoria.

El usuario entiende de forma clara las descripciones incluidas en los tooltips de las opciones.	Los usuarios entienden las descripciones de los *tooltips* por lo general, aunque los *tooltips* de las opciones avanzadas no son entendidos por los usuarios sin conocimientos sobre informática, ya que requiere de entender sobre aspectos relacionados con modelos de IA.
El usuario entiende como corregir los errores cometidos de forma clara una vez lee los mensajes de error mostrados por el sistema.	Los usuarios han sabido corregir las opciones mal configuradas una vez que se han equivocado y han leído los mensajes de error.

Figura 8.13 Resultados cuestionario responsable de las pruebas

8.2.2 Pruebas de Accesibilidad

A continuación, se muestra la tabla con las diferentes pruebas de accesibilidad que se han propuesto llevar a cabo con sus correspondientes resultados y notas obtenidas tras llevar a cabo estas pruebas.

Prueba	Resultado	Notas
¿El manual de usuario incluye ejemplos prácticos y accesibles?	Correcto.	
¿Se pueden ajustar el brillo y el contraste de la aplicación?	Incorrecto.	Es posible cambiar la temática a oscura o clara en función de las necesidades del usuario.
¿La interfaz responde adecuadamente al cambio de tamaño de ventana?	Correcto.	Responde correctamente, a no ser que se minimice a tamaños muy pequeños que los usuarios no suelen utilizar, donde la interfaz no responde correctamente, aunque con tamaños estándar utilizados en portátiles responde correctamente aunque se minimice la ventana.
¿Los errores y advertencias son claros y específicos?	Correcto.	
¿Se ofrecen modos de alto contraste o temas oscuros para personas con sensibilidad a la luz?	Correcto.	No se muestran modos de alto contraste pero se permite utilizar un modo oscuro.
¿La aplicación permite navegar completamente utilizando el teclado?	Incorrecto.	
Soporta todos los sistemas operativos?	Correcto.	Soporta todos los sistemas operativos soportados por tkinter.
Las imágenes e iconos, ¿Son fáciles de entender?	Correcto.	Solamente se incluyen iconos para los tooltips.

¿Se puede cambiar la fuente y su tamaño?	Incorrecto.	
No se usa el color como único medio para transmitir información.	Correcto.	Además del color se muestran mensajes de error con texto.
¿Son apropiados para todos los usuarios los colores de la aplicación?	Correcto.	

Figura 8.14 Resultados pruebas accesibilidad

8.3 Pruebas de Rendimiento

A continuación, en la siguiente tabla se muestran los resultados obtenidos tras llevar a cabo las pruebas de rendimiento, realizadas en un equipo con las especificaciones descritas en 6.5, por lo tanto, se ejecutarán los modelos en *CPU* y todos utilizarán el mismo *prompt*.

Para llevar a cabo la medida del tiempo de ejecución de cada modelo, para el caso de prueba de generar imagen, se ha ejecutado cada modelo configurado con 10 pasos de inferencia y midiendo el tiempo que tarda en ejecutarse cada paso, de forma que la media de tiempo de ejecución los pasos de inferencia será el resultado del tiempo de ejecución para generar una imagen, es decir, el tiempo de ejecución se corresponderá con el tiempo medio de ejecución de un paso de inferencia, esto se hace debido a que cada modelo por defecto utiliza un número de

pasos de inferencia diferente por defecto, por lo que de esta forma podremos saber el tiempo que tarda en ejecutarse un paso de inferencia en cada modelo, así será posible tener una imagen más acertada del rendimiento de cada modelo.

Para medir el tiempo de carga del modelo en memoria, se ha medido de forma manual, desde que se inicia el proceso en la interfaz hasta que el sistema indica que se ha cargado el modelo, de forma que se repite el proceso 5 veces para obtener la media de todos los tiempos de las cargas en memoria y obtener un valor más realista.

Para seguir, el resultado de uso de memoria y uso de *CPU* se ha obtenido también de forma manual mediante el administrador de tareas, de forma que se escogerá como resultado en la tabla el valor máximo mostrado en el administrador de tareas durante la ejecución de dichos procesos.

También debemos tener en cuenta que, las pruebas se han realizado una vez que los modelos se encuentran ya cargados en el directorio temporal del equipo, por lo que en caso de que sea la primera vez que se ejecuta el sistema los tiempos serán mayores, debido a la necesidad de descargar localmente los modelos, como se explica en 9.3.

Modelo	**Prueba**	**Tiempo ejecución (s)**	**Uso de memoria (MB)**	**% Uso de CPU.**
Stable-diffusion-2	Cargar modelo	1,61s	300,6MB	3,1%
Stable-diffusion-2	Generar imagen	32,06s	680,1MB	89,5%
Stable-diffusion-1.5	Cargar modelo	1,46s	324,8MB	1,8%
Stable-diffusion-1.5	Generar imagen	12,79s	700,1MB	48,5%
Stable-diffusion-XL	Cargar modelo	1,35s	373,9	4,1%
Stable-diffusion-XL	Generar imagen	914,93s	743,8MB	87,9%
Stable-diffusion-XL-Turbo	Cargar modelo	1,62s	423MB	2,1%
Stable-diffusion-XL-Turbo	Generar imagen	4,66s	857MB	40,9%

Figura 8.15 Resultados pruebas de rendimiento

Capítulo 9. Manuales del Sistema

En este capítulo se presentan los diferentes manuales del sistema, entre ellos se encuentran el manual de instalación, el manual de ejecución, el manual de usuario y el manual del programador, donde cada uno de ellos estará diseñado para presentar información con diferentes enfoques según el rol de la persona a la que está dirigido cada manual.

9.1 Manual de Instalación

El sistema se distribuye como un archivo comprimido junto al contenido adjunto. Para empezar, se debe descomprimir este archivo en un directorio del equipo y una vez descomprimido se creará la carpeta *ImageGenTool* dentro de la cual se encontrarán todos los archivos necesarios para más adelante poder ejecutar el sistema.

Antes de poder seguir con los siguientes pasos es necesario instalar *Python 3.12.0,* el cual se puede encontrar en [47], además debemos tener en cuenta durante la instalación, que es necesario seleccionar la opción *Add Python 3.12.0 to PATH* en caso de no haber seleccionado la opción durante la instalación o simplemente para comprobar que se ha añadido correctamente al *PATH* es posible seguir el siguiente tutorial en *[48].*

Una vez se ha instalado correctamente *Python*, el siguiente paso será abrir el símbolo del sistema escribiendo "*cmd*" en la barra de búsqueda de Windows y ejecutando el programa mostrado denominado "Símbolo del sistema".

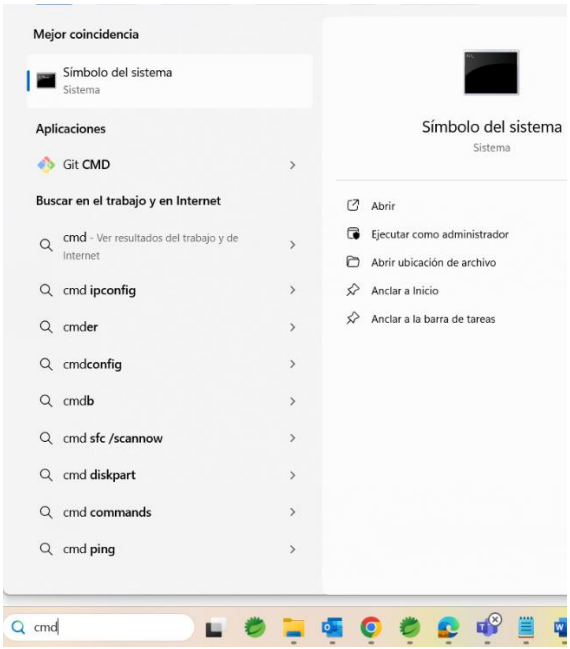


Figura 9.1 Manual de instalación abrir cmd

Una vez abierto el símbolo del sistema debemos movernos al directorio de la carpeta que se ha creado en el primer paso al extraer el archivo comprimido escribiendo el siguiente comando en el *CMD:*

```
cd <directorioExtracciónComprimido>\ImageGenTool\ImageGenTool
```

Una vez dentro del directorio de la carpeta obtenida tras descomprimir, debemos crear el entorno virtual donde instalaremos nuestras dependencias necesarias para arrancar el proyecto ejecutando el siguiente comando en el *CMD.*

```
python -m venv venvgentool
```

Tras esto, debemos activar el entorno virtual donde se encontrarán todos los módulos necesarios para posteriormente poder ejecutar el sistema, para ello se debe ejecutar el siguiente comando en el *CMD.*

```
venvgentool\Scripts\activate
```

Tras ejecutar este comando se habrá activado el entorno virtual, podremos comprobar que se ha activado correctamente si no se ha mostrado ningún error tras ejecutar el comando y podremos ver en la línea de comandos que se muestra *(venvgentool)* de la siguiente forma:

```
C:\Users\diego.santos\Desktop\gentoolclean\ImageGenTool\ImageGenTool>venvgentool\Scripts\activate

(venvgentool) C:\Users\diego.santos\Desktop\gentoolclean\ImageGenTool\ImageGenTool>
```

Figura 9.2 cmd con venv activo

Tras activarse el entorno virtual deberemos descargar las dependencias del proyecto ejecutando el siguiente comando en el *CMD.*

```
pip install -r requirements.txt
```

Una vez que se hayan descargado todas las dependencias, debemos modificar el nombre de una constante utilizada dentro de la librería de *ttkbootstrap,* debido a un bug en la librería no corregido por los autores, que causará problemas al generar imágenes en caso de no ser resuelto, para ello, debemos abrir con un editor el archivo denominado widgets.py, el cual se encuentra dentro del entorno virtual que acabamos de crear, en el directorio *venvgentool**Lib**site-packages**ttkbootstrap**widgets.py*.

Por lo que, una vez encontrado el archivo, debemos abrirlo y buscar la constante llamada *CUBIC,* encontrada en la siguiente parte del código.

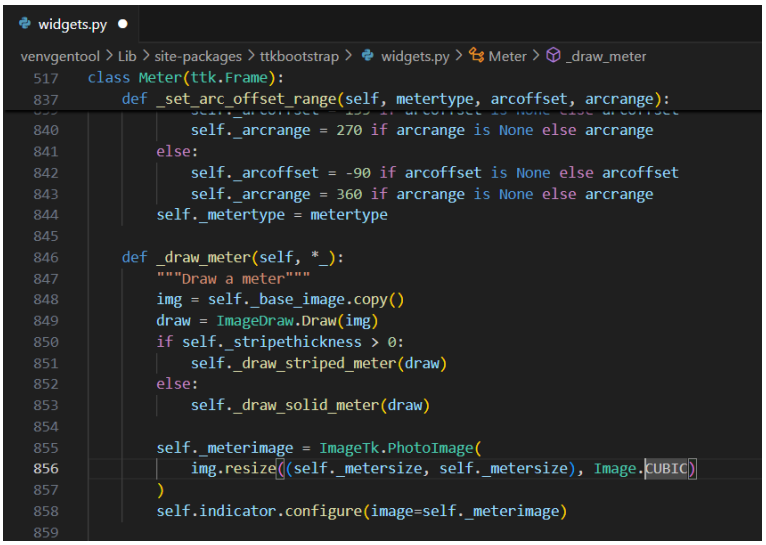


Figura 9.3 Constante ttkbootsrap

Una vez localizada la constante la sustituiremos por la constante *BICUBIC* y guardaremos los cambios, por lo que el código nos quedaría arreglado de la siguiente forma:

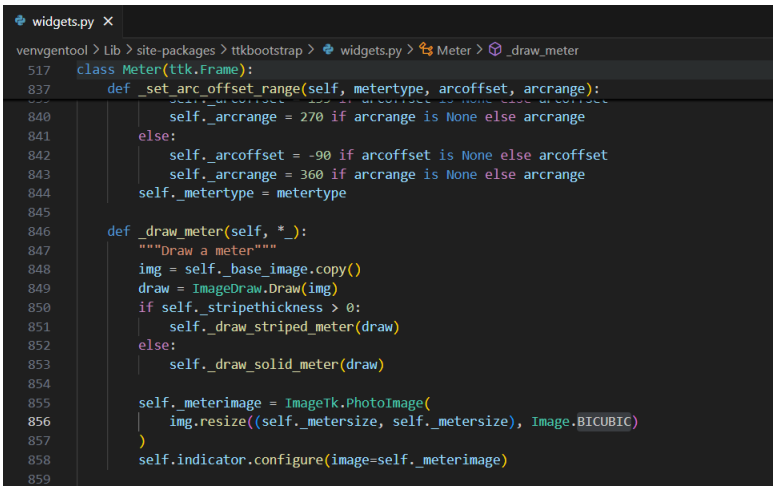


Figura 9.4 Constante ttkbootstrap arreglada

Tras modificar y guardar este archivo, el sistema estará preparado para poder ser ejecutado, ya que tendremos el entorno ya configurado con todas las dependencias y módulos configurados dentro de este entorno virtual.

9.2 Manual de Ejecución

Una vez completados con éxito todos los pasos especificados en 9.1 el sistema estará listo para poder ser ejecutado. Para llevar a cabo la ejecución del sistema, primeramente, debemos abrir un *CMD* y activar el entorno virtual en caso de que no esté activado, para ello, una vez abierto el *CMD* es necesario movernos a la ruta donde se encuentra la carpeta del entorno virtual ejecutando el siguiente comando:

```
<directorioExtracciónComprimido>\ImageGenTool\ImageGenTool
```

Una vez en este directorio se debe activar el entorno virtual con el siguiente comando:

```
venvgentool\Scripts\activate
```

Tras activar el entorno virtual, es necesario movernos al directorio en el que se encuentra el archivo *main.py* encargado de arrancar el sistema, el cual se encuentra en el directorio raíz del código fuente del sistema, para movernos a esta ruta debemos ejecutar en el *CMD* el siguiente comando:

```
cd tool
```

Una vez dentro del directorio raíz, podremos arrancar el sistema ejecutando el siguiente comando:

```
python main.py
```

Tras unos breves segundos se ejecutará el sistema y se mostrará en pantalla la interfaz gráfica de usuario mediante la cual será posible comenzar a operar con la herramienta.

9.3 Manual de Usuario

En esta sección se explican las funcionalidades principales que el usuario podrá llevar a cabo con el sistema una vez se haya instalado siguiendo los pasos de 9.1 y una vez se haya ejecutado el sistema siguiendo los pasos establecidos en 9.2.

En las siguientes subsecciones se explica como generar una o varias imágenes con las opciones básicas obligatorias y también se explica como generar imágenes modificando las opciones avanzadas de los modelos generativos disponibles.

9.3.1 Generar imágenes con opciones básicas

En esta sección se explica como generar imágenes con la configuración mínima que se debe realizar para que el sistema permita generar una o varias imágenes.

Una vez se arranque el sistema aparecerá la ventana principal de la interfaz gráfica de usuario configurada de la forma mostrada en la siguiente captura.

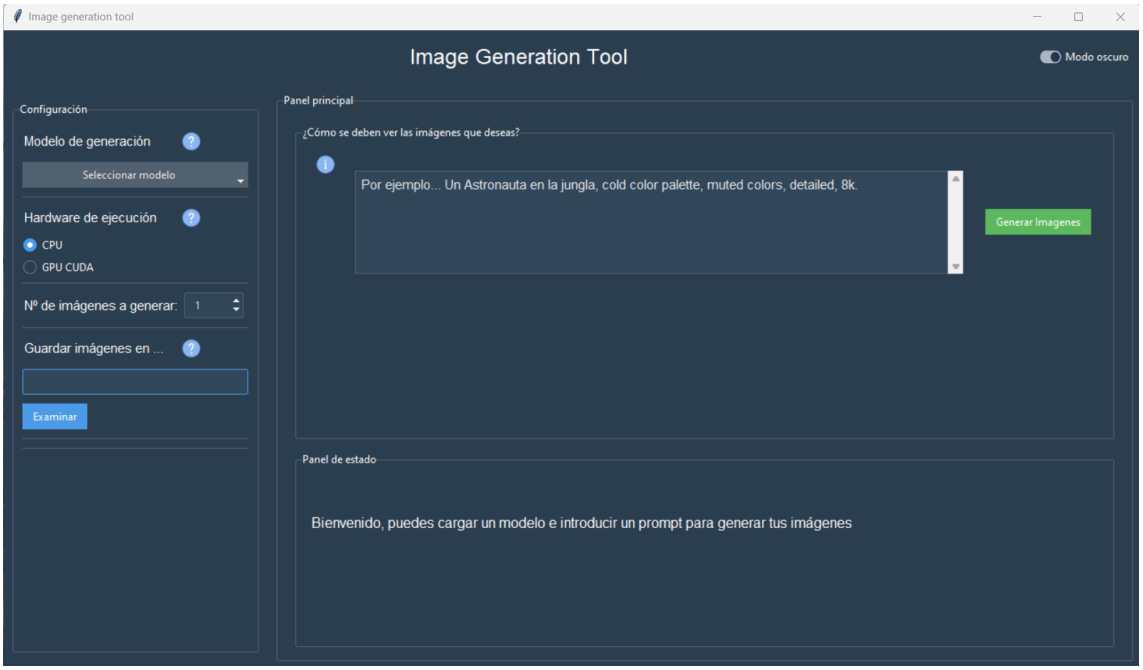


Figura 9.5 Manual de usuario ventana inicio

Para poder llevar a cabo la generación de imágenes con las opciones básicas, primeramente es necesario seleccionar un modelo y cargarlo en memoria, para ello, en el panel "Configuración" situado en la parte lateral izquierda de la interfaz, se debe desplegar el selector de modelo de generación situado en la primera fila del panel de configuración, una vez abierto, el sistema mostrará los diferentes modelos disponibles y se debe seleccionar uno para continuar, como se muestra a continuación en la captura, una vez seleccionado un modelo el sistema mostrará el botón "Cargar Modelo" justo debajo del selector de modelos.

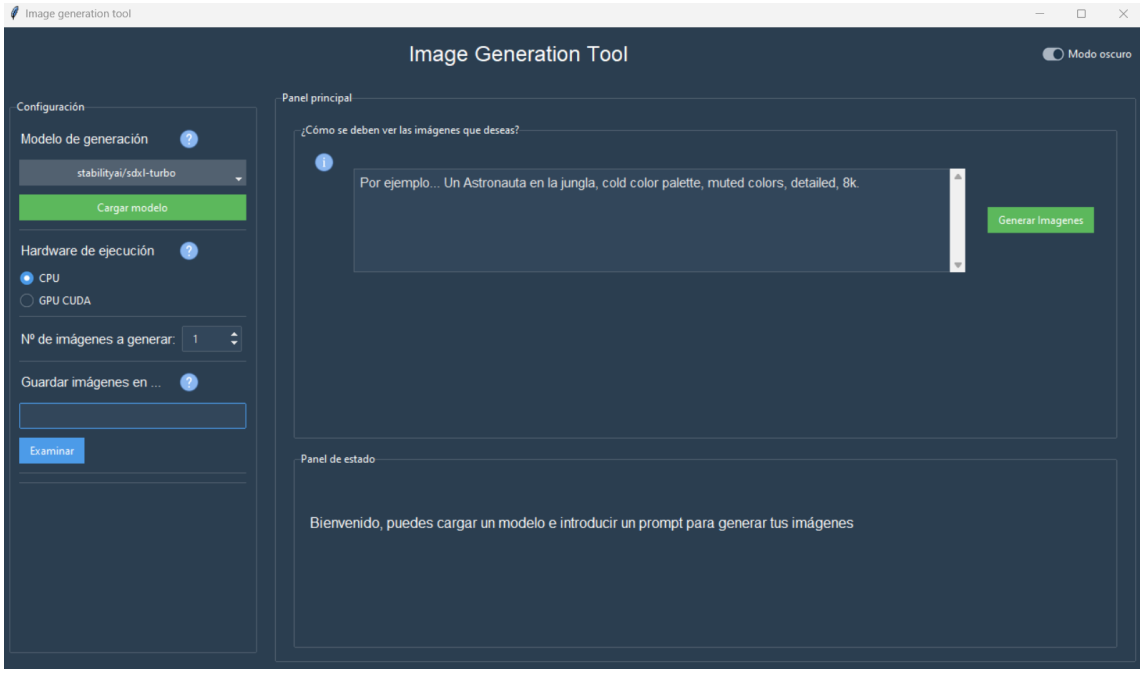


Figura 9.6 Manual de usuario botón cargar modelo

Esto se debe a que, durante esta ejecución del sistema, el modelo seleccionado no se ha cargado aún en memoria lo cual es necesario para llevar a cabo la generación de imágenes con cualquiera de los modelos que el usuario seleccione y no hayan sido cargados previamente durante la ejecución.

Por lo tanto, se debe hacer *click* en el botón para cargar un modelo y una vez hecho esto comenzará el proceso de carga en memoria del modelo seleccionado y el sistema indicará en el panel de estado situado en la parte inferior, que el proceso se encuentra en progreso como se puede ver a continuación en la captura, mediante una barra de progreso, además de desactivar el botón de carga en memoria y mostrará un mensaje en la parte inferior derecha indicando que el proceso ha comenzado.

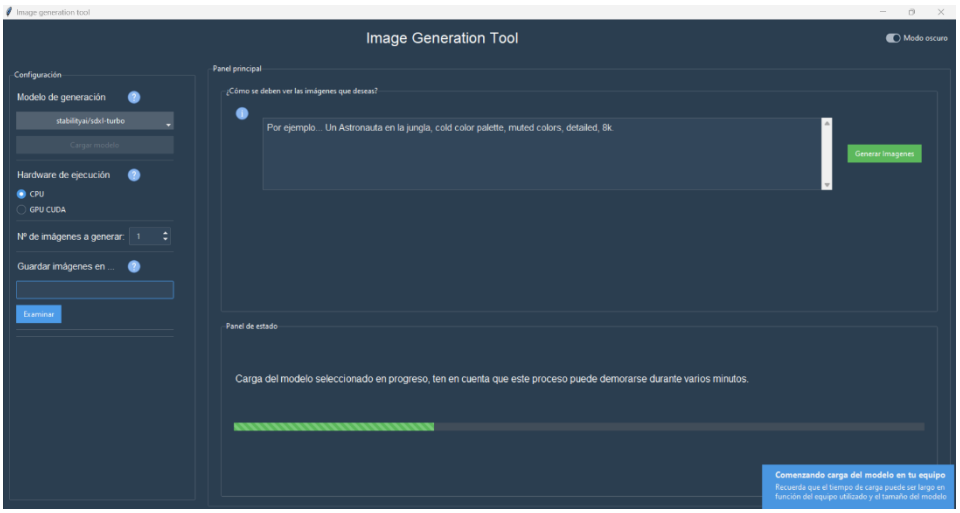


Figura 9.7 Manual de usuario proceso carga en memoria

Además, se debe tener en cuenta que, si se trata de la primera vez que se ejecuta el sistema en el equipo del usuario, el proceso de carga en memoria de los modelos puede alargarse durante aproximadamente entre 3 y 10 minutos, en función de los recursos de los que disponga el usuario, ya que en la primera carga en memoria el sistema deberá descargar los modelos en un directorio temporal.

Una vez se carguen en memoria los modelos por primera vez, en las siguientes ejecuciones del sistema los modelos se cargarán en memoria de forma prácticamente instantánea, ya que se encontrarán almacenados en un directorio temporal del equipo del usuario.

Una vez que el modelo seleccionado se haya cargado en memoria el sistema ocultará el botón de cargar en memoria cada vez que se encuentre seleccionado este modelo, ya que se guardará en memoria a lo largo de toda la ejecución del sistema, hasta que se cierre el programa, que será necesario volver a cargar los modelos ya que se borrarán de la memoria de nuestro ordenador.

Además, una vez se haya cargado el modelo correctamente el sistema mostrará un mensaje de éxito, por lo que el sistema lucirá de la siguiente forma mostrada en la captura de a continuación.

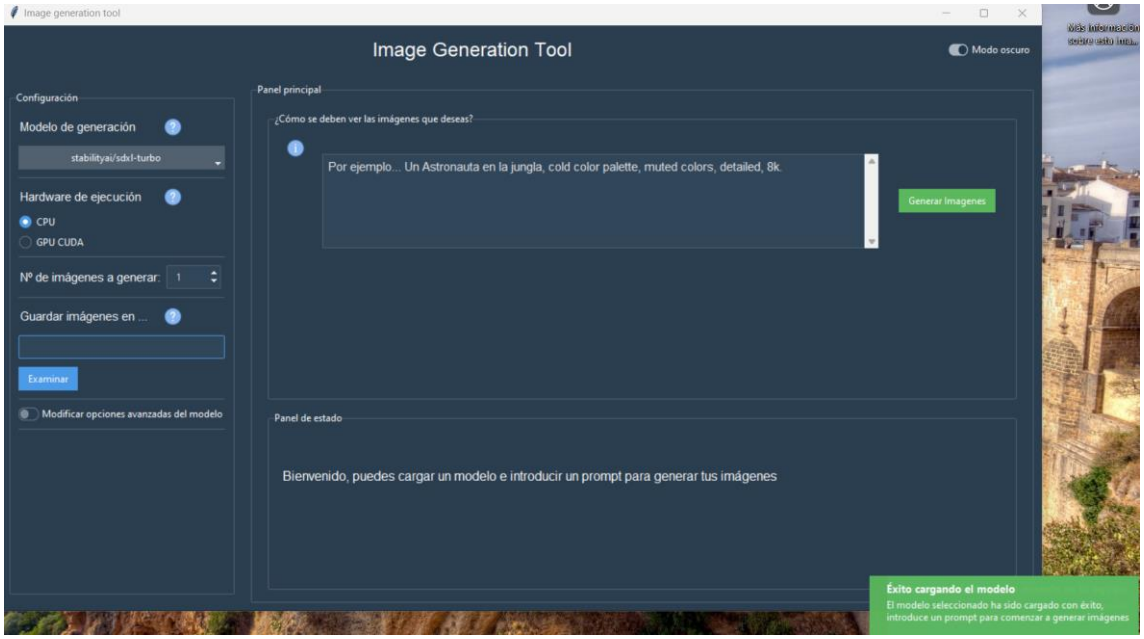


Figura 9.8 Manual de usuario modelo cargado en memoria

El siguiente paso será configurar el resto de las opciones visibles en el panel lateral izquierdo de configuración, comenzando por la opción "*Hardware* de ejecución", mostrada debajo del selector de modelo, en este caso solamente será posible seleccionar una de las opciones "*CPU*" o "*GPU*" (por defecto estará seleccionada la opción *CPU*), en función de la opción que se elija, el sistema generará las imágenes utilizando la *CPU* o en el otro caso la *GPU* del equipo.

Debemos tener en cuenta que la generación de imágenes se ejecutará con un rendimiento mucho mayor si se selecciona la opción "*GPU*", pero para ello, el usuario deberá tener instalada una *GPU* compatible con *CUDA [49]* instalada en el equipo donde se ejecute el sistema, en caso

de que no se tenga y se seleccione esta opción, al generar las imágenes se mostrará un mensaje de error y se ejecutará la generación en la *CPU* del equipo del usuario de forma automática.

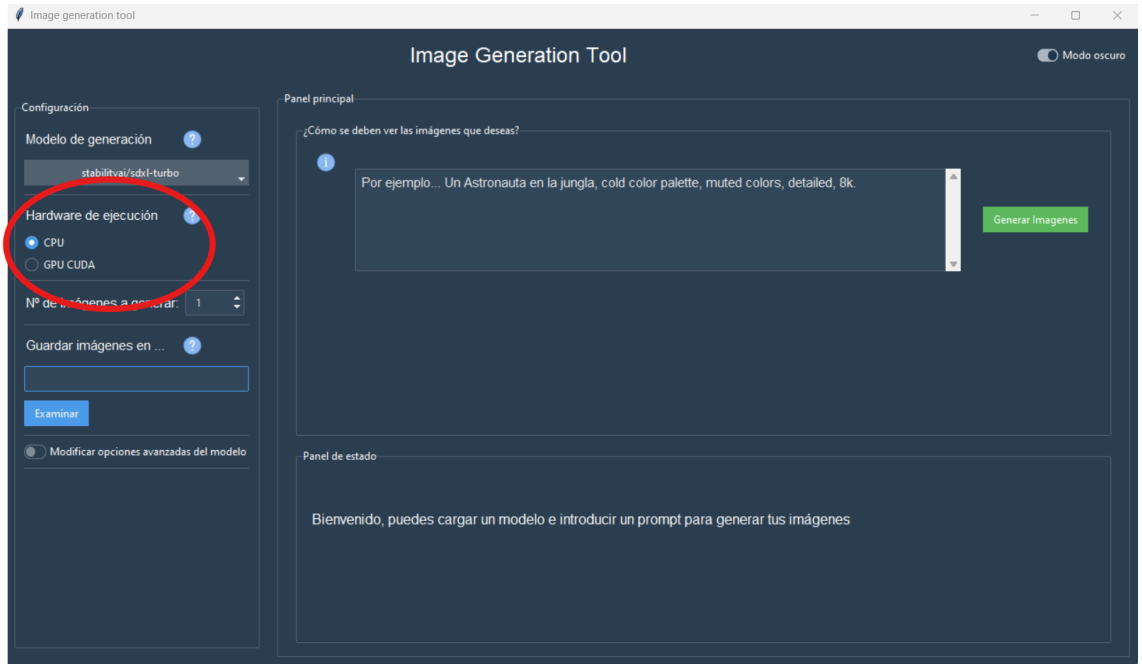


Figura 9.9 Manual de usuario Hardware de ejecución

Tras seleccionar una de las opciones de *hardware* de ejecución, será necesario introducir el número de imágenes que se quiere generar, por defecto el número de imágenes será 1 y aunque se ha mencionado que se pueden generar imágenes de forma ilimitada, se han limitado a 10 millones por razones de seguridad, aunque si algún usuario lo solicita es posible modificar este límite de forma sencilla.

Para introducir un número de imágenes será posible utilizar las flechas del *widget* selector de número de imágenes o introducir directamente el número en el widget, señalado en la interfaz a continuación en la siguiente captura.

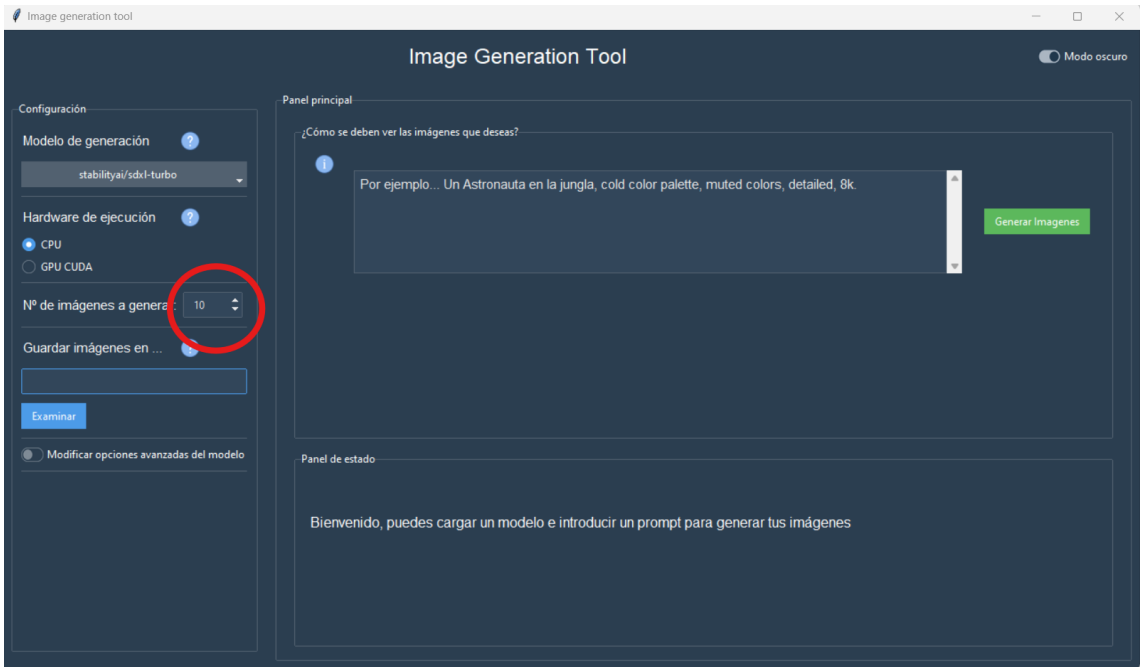


Figura 9.10 Manual de usuario nº imágenes a generar

El siguiente paso será introducir el directorio en el que se irán guardando las imágenes durante el proceso de generación, se podrá introducir el directorio directamente en el cuadro de texto o hacer *click* en el botón azul "Examinar" para abrir el explorador de archivos del equipo y seleccionar un directorio como se muestra a continuación.

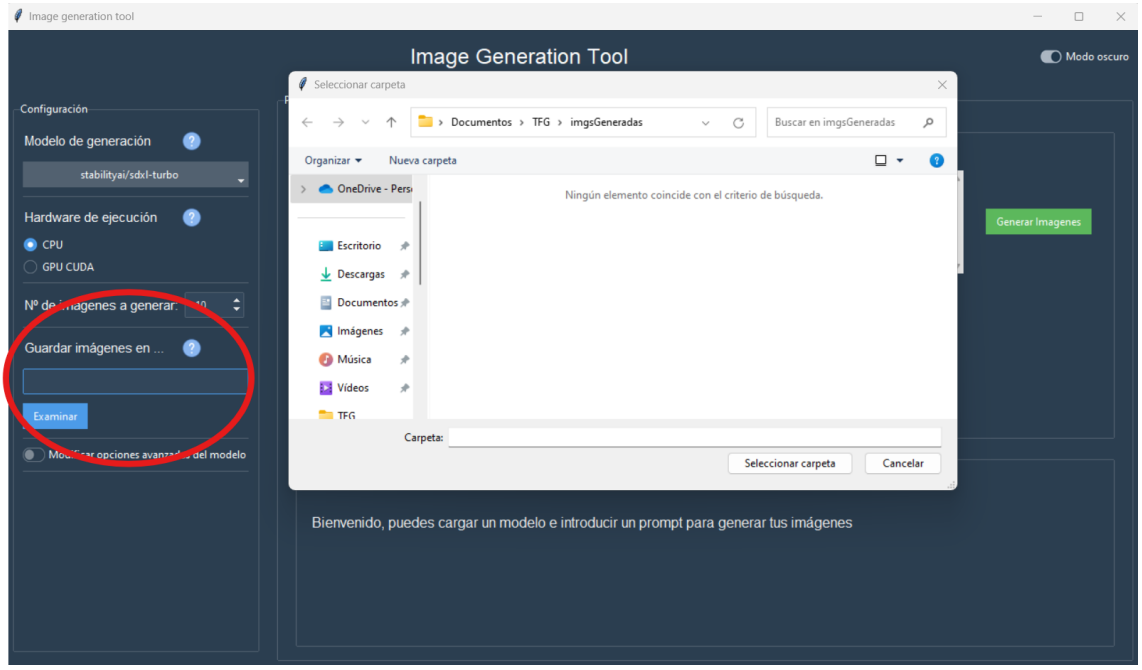


Figura 9.11 Manual de usuario seleccionar directorio

Una vez seleccionemos la carpeta en el explorador de archivos podremos ver la ruta seleccionada en el campo de texto de la opción "Guardar imágenes en…" como se puede ver en la siguiente captura.

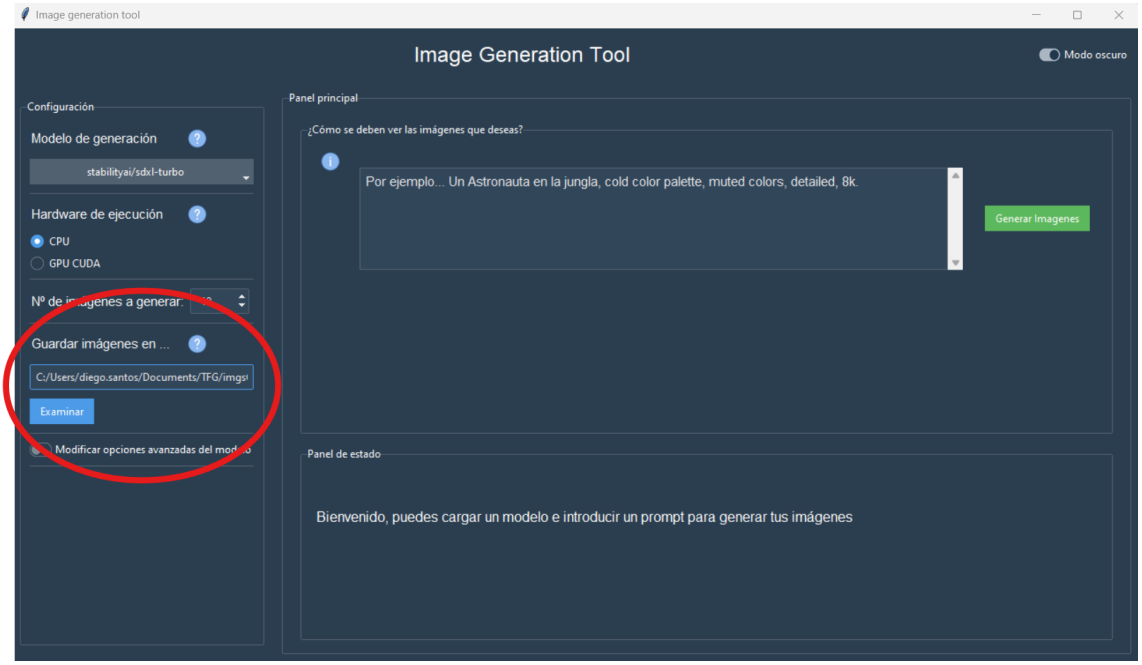


Figura 9.12 Manual de usuario directorio seleccionado

Tras esto podremos proseguir al siguiente y último paso, que será introducir una descripción textual de las imágenes que se quieren generar, para ello deberemos introducir en el área de texto del panel principal esta descripción, aunque vendrá una por defecto podemos introducir por ejemplo la mostrada en la captura siguiente y tras esto podremos hacer *click* en el botón verde a la derecha del área de texto para comenzar la generación de las imágenes.

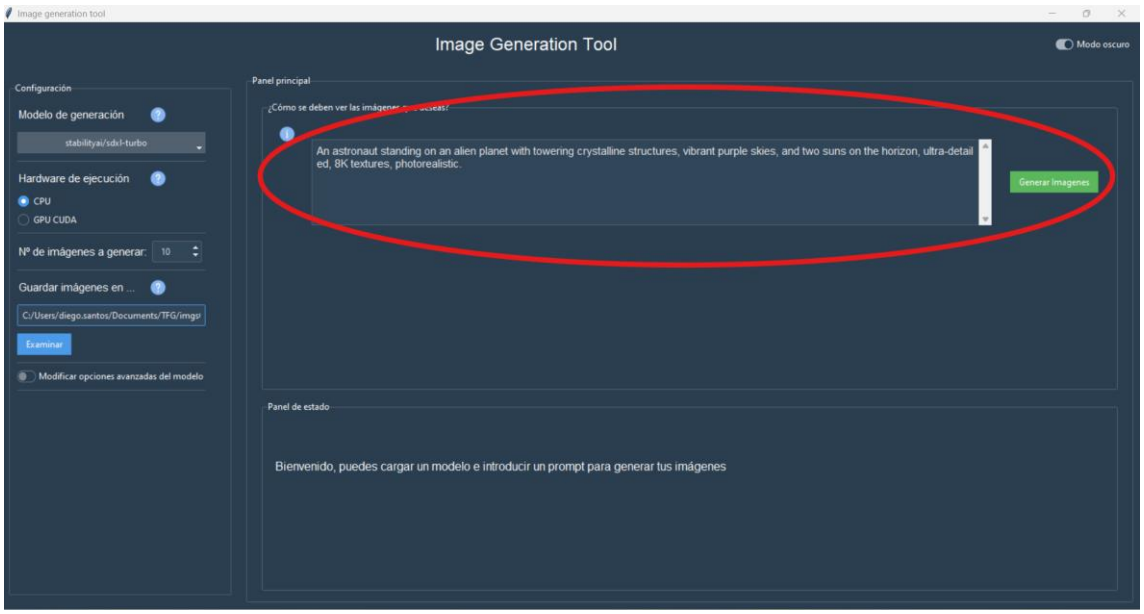


Figura 9.13 Manual de usuario texto descriptivo

Como último paso, tras hacer *click* en el botón generar imágenes comenzará el proceso de generación de imágenes y se mostrará el progreso del número de imágenes que ya se han

generado y además la interfaz también mostrará las imágenes que se van generando durante el proceso, como se puede ver en la siguiente captura.

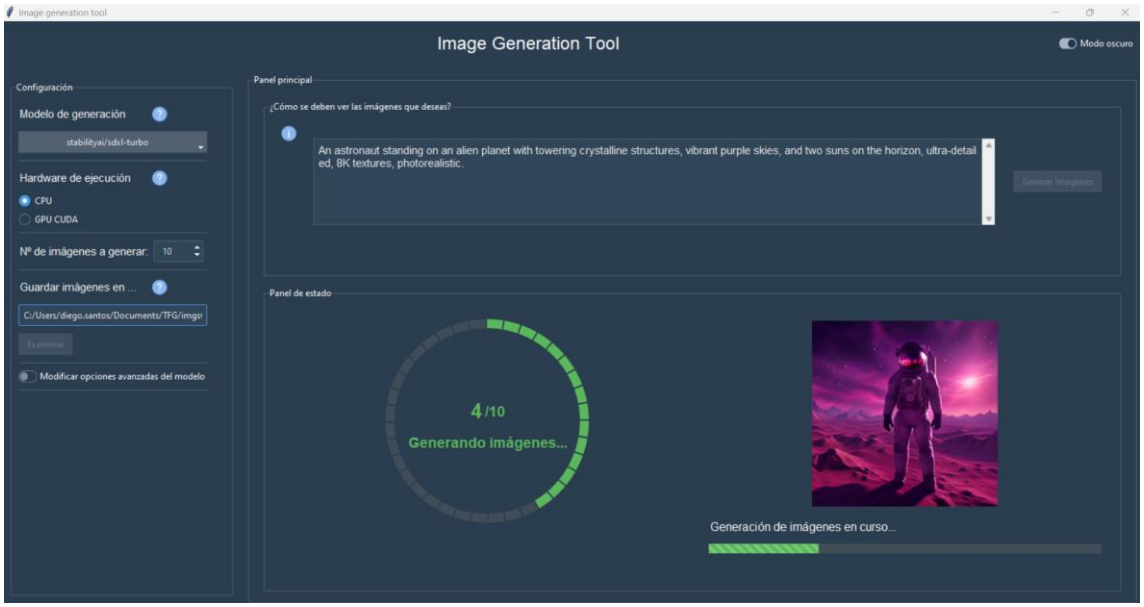


Figura 9.14 Manual de usuario proceso de generación

Una vez que se hayan generado todas las imágenes, se mostrará un mensaje de éxito verde en la parte inferior derecha y las imágenes estarán disponibles en el directorio que se ha introducido previamente en la opción “Guardar imágenes en…”.

Para el caso del ejemplo podremos ver las imágenes guardadas de la siguiente forma mostrada en la siguiente captura en el directorio establecido.

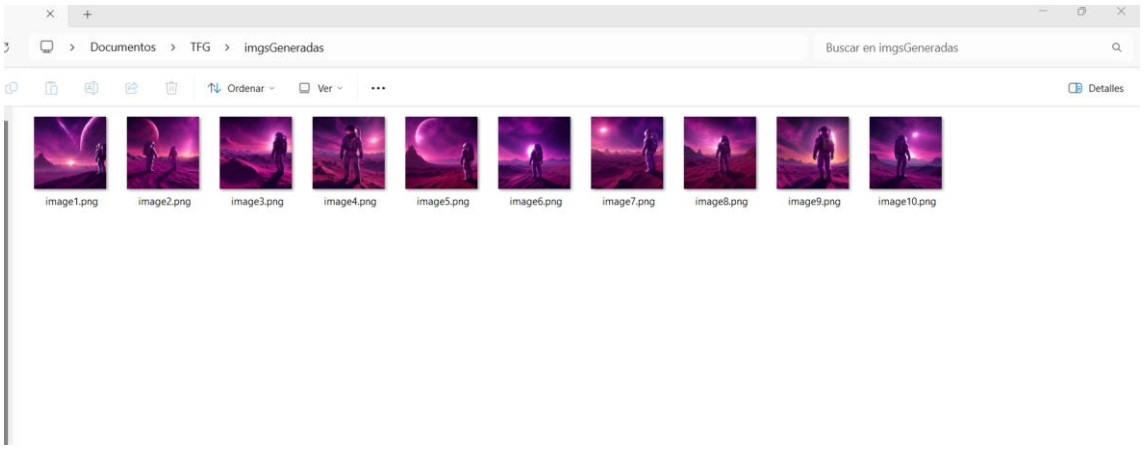


Figura 9.15 Manual de usuario imágenes guardadas

Tras todos estos pasos se habrá completado el proceso de generación y se podrá comenzar otra vez a generar imágenes.

9.3.2 Generar imágenes con opciones avanzadas

En esta subsección se explica como generar imágenes con opciones avanzadas del modelo seleccionado, de forma que es posible modificar ciertos parámetros del modelo que se encuentra seleccionado y cargado en memoria.

Para facilitar el manual, primeramente, debemos seleccionar un modelo y cargarlo en memoria y establecer todas las opciones básicas como se explica en 9.3.1.

Tras esto será posible generar imágenes, aunque en este caso se explicará como generarlas modificando ciertos parámetros del modelo seleccionado, teniendo en cuenta que estos parámetros pueden variar en función del modelo que se seleccione.

Para empezar, se puede comprobar, que una vez realizada la carga en memoria de un modelo o se seleccione un modelo que se encuentra cargado en memoria se mostrará un *widget* a modo de interruptor en la parte baja del panel lateral de configuración, se puede ver en indicado en la segunda captura.

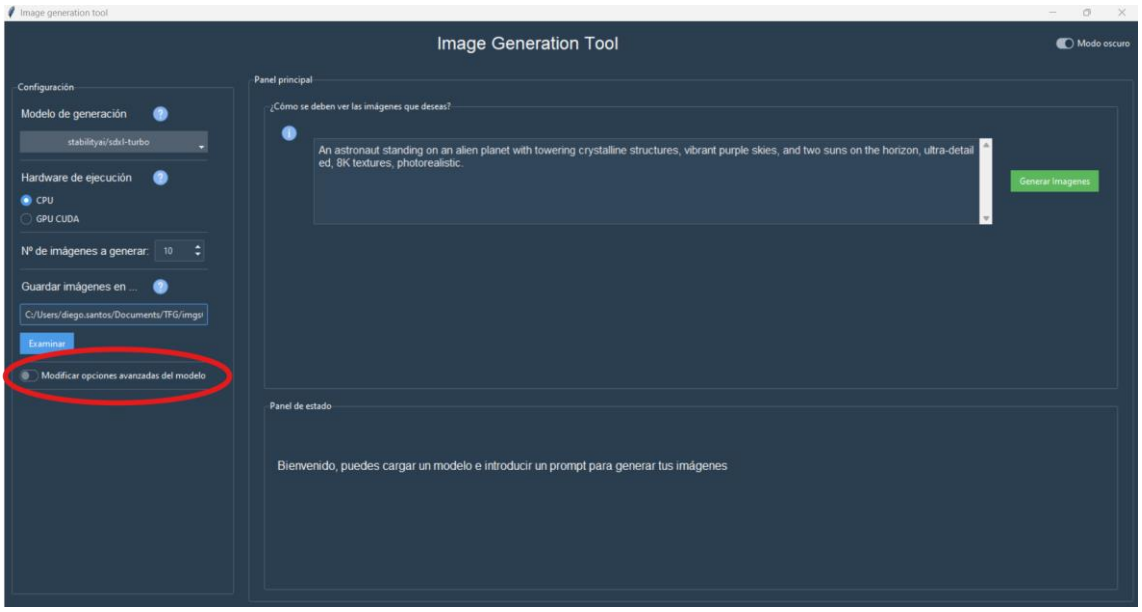


Figura 9.16 Manual de usuario widget opciones avanzadas

Una vez localizado este *widget*, al activar el interruptor haciendo *click* en el *widget*, se mostrará un panel con diferentes opciones referentes a los parámetros que se pueden modificar del modelo seleccionado, donde podremos visualizar todas estas opciones haciendo *scroll* por el panel con la barra lateral de *scroll* del panel.

Figura 9.17 Manual de usuario panel opciones avanzadas

Una vez conocido esto, se puede modificar cada valor por defecto utilizando las flechas de cada selector dentro de un rango limitado por el sistema dentro de los valores válidos para el modelo seleccionado, salvo la última opción llamada "*negative prompt*" la cual es un área de texto en la que se deben introducir un texto descriptivo de lo que no queremos que el modelo nos genere en nuestras imágenes.

Además, en el caso de que no se conozca para que sirve cada parámetro se puede colocar el cursor sobre los iconos azules con una interrogación situados en la parte superior izquierda de cada opción del panel, conocidos como *tooltips.* Al hacer esto se mostrará un cuadro de texto en el que se explicará para que sirve el parámetro asociado al *tooltip,* ver siguiente captura.

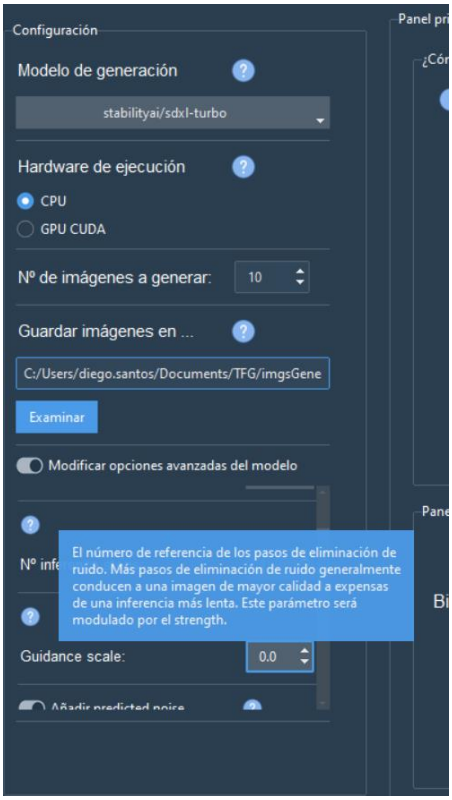


Figura 9.18 Manual de usuario tooltip inference steps

Se debe tener en cuenta, que la generación puede perder calidad o ganar calidad en función de los parámetros del modelo y en función del modelo seleccionado unos parámetros pueden funcionar mejor que otros, además de que estos parámetros también serán más efectivos o menos en función de los objetivos del usuario, aunque esto queda fuera del alcance de este manual.

A continuación, se muestra un ejemplo de los parámetros modificados para el modelo seleccionado previamente en el manual.

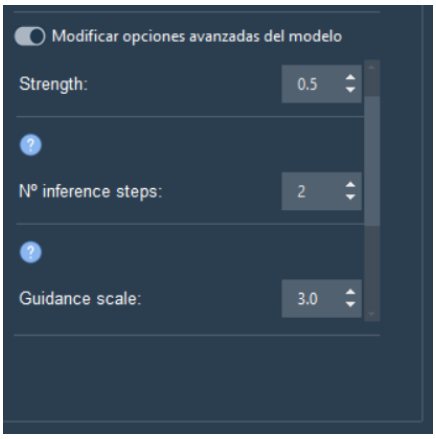


Figura 9.19 Manual de usuario opciones avanzadas1

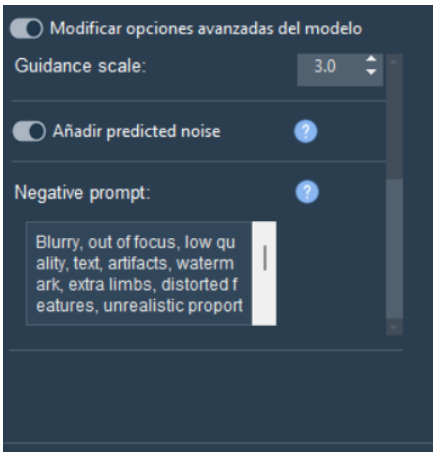


Figura 9.20 Manual de usuario opciones avanzadas2

Una vez seleccionadas estas opciones y las básicas como se ha explicado previamente en esta sección, podremos proceder a generar imágenes haciendo *click* en el botón generar imágenes, tras esto comenzará el proceso de generación y una vez terminado se guardarán estas imágenes en el directorio introducido en las opciones básicas como se explica en 9.3.1, por lo tanto una vez seguidos todos estos pasos, habrá sido posible generar imágenes con los parámetros avanzados del modelo modificados al gusto del usuario, de forma que se puedan obtener diferentes resultados en función de estos valores introducidos respecto a los parámetros avanzados por defecto que utilizan los diferentes modelos.

9.4 Manual del Programador

En este capítulo se explicarán diferentes aspectos del sistema con el objetivo de servir como guía para otros programadores, de modo que, en el futuro sea posible de forma fácil ampliar el sistema con nuevas funcionalidades o extender las ya existentes además de permitir entender cómo llevar una mantenibilidad del sistema, para ello se explicarán los aspectos clave que es necesario entender para lograr estos objetivos.

9.4.1 Añadir nuevo modelo al sistema

En esta subsección se explica como añadir nuevos modelos al sistema, en concreto, para empezar, como añadir un nuevo modelo perteneciente a la librería *diffusers*, ya que el sistema está diseñado para trabajar con esta librería y por tanto diseñado de forma que sea sencillo añadir nuevos modelos con cierta agilidad gracias al diseño desarrollado.

Para ello el nuevo modelo debe introducirse creando una clase que debe heredar de la clase abstracta *DiffusersBaseModel*, ya que esta clase define los métodos que deben implementar las clases hijas correspondientes con modelos concretos, además de contener funcionalidad común, que comparten las implementaciones de los modelos basados en la librería *diffusers* de forma que no haya que repetir este código cada vez que se añada un nuevo y por tanto se facilite la mantenibilidad del código y la agilidad para introducir nuevos modelos pertenecientes a esta librería.

Por lo tanto, para añadir un nuevo modelo se creará una nueva clase que extienda a *DiffusersBaseModel*, donde esta nueva clase debe implementar los métodos abstractos definidos en la clase base, en concreto los siguientes:

- *load_pipeline:* Este método debe implementar la funcionalidad de establecer que pipeline de la librería *diffusers* específico con el modelo que se quiere añadir, en concreto asocia a la variable *pipeline* de la clase padre el pipeline del modelo concreto.
- *generate_image:* Este método es el encargado de implementar la funcionalidad de generar una imagen, por lo general solamente llamará al *generate_image* de la clase padre, pero en caso de ser necesario los nuevos modelos que se añadan deberán implementar este método de forma que requieran para poder generar una imagen.
- *get_default_inference_steps:* Este método está relacionado con las opciones avanzadas de cada modelo, debe devolver el valor por defecto del parámetro *inference_steps* del modelo que se añade.
- *get_default_guidance_scale:* Este método también está relacionado con las opciones avanzadas del modelo y debe devolver el valor por defecto del parámetro *guidance_scale* por defecto del modelo añadido.

Una vez implementados estos métodos, es necesario añadir este modelo a las opciones de la interfaz para que pueda ser utilizado por los usuarios, para ello, en la clase *ModelManager* debemos modificar el método *get_available_models*, el cual devuelve un diccionario, cuyas claves son el nombre del modelo que se muestra en el selector de la interfaz y los valores son

las clases de los modelos concretos, por lo tanto para añadir el nuevo modelo, debemos añadir una nueva entrada al diccionario cuya clave sea el nombre que se quiera mostrar en el selector de modelos de la interfaz y cuyo valor sea la nueva clase que se ha creado para este modelo.

Tras esto ya será posible utilizar este nuevo modelo añadido en el sistema para generar imágenes.

9.4.2 Aspectos generales para la mantenibilidad y extensión del sistema

A lo largo de esta sección se explican los conceptos generales sobre cómo está organizada la implementación del sistema de forma que, en el futuro, otros programadores puedan guiarse fácilmente en torno a los diferentes elementos que componen el sistema, para realizar labores de mantenibilidad, extensión, modificación u otros objetivos a realizar sobre el sistema.

Dentro de la carpeta raíz del código fuente, en concreto la carpeta */tool* encontrada dentro de la carpeta que almacena el código fuente del proyecto se encuentran diferentes carpetas con diferentes archivos *Python* organizados según funcionalidades u objetivos, en concreto se encuentran las carpetas */gui*, */imageGeneration*, */tests* y */utils*. Además, dentro de la carpeta raíz se encuentra el archivo *main.py* que contiene el código encargado de arrancar el sistema. A continuación, se resumirá el contenido de las carpetas principales necesarias para entender la implementación del proyecto y cada archivo de código contenido dentro de estas de forma ordenada.

Para empezar, se comenzará por la carpeta */gui*, que contiene los siguientes archivos de código:

- *gui.py:* Este archivo contiene todo el código relacionado con la interfaz gráfica de usuario del sistema, está construido sobre *tkinter* y utiliza la librería *ttkbootstrap* para mejorar la apariencia y estilo de la interfaz. Las funcionalidades clave que contiene este archivo son los widgets de selección de modelo, configuración de opciones avanzadas, gestión del inicio del proceso de generación y gestión de muestra de mensajes al usuario. Contiene los siguientes componentes principales:
    - *MainApplication:* Se trata de la ventana principal de la aplicación y controla la estructura principal de la interfaz además de gestionar los eventos.
    - *HeaderFrame:* Se trata de la cabecera de la aplicación y contiene el título y un *toggle* para cambiar la temática oscura o clara de la aplicación.
    - *LateralFrame:* Contiene el panel de configuración de la aplicación con diferentes widgets para configurar las opciones básicas del modelo, hardware, etc.
    - *MainFrame:* Panel principal en el que se encuentra el área de texto del prompt y el panel de estado de la aplicación.
- *controller.py*: Gestiona la lógica del sistema intermediando entre la interfaz gráfica y los modelos generativos, entre sus funcionalidades clave se encuentran la selección y carga en memoria de los modelos, gestión de hilos de los procesos de carga y generación. Por lo que su propósito general es la administración de la lógica de negocio de la aplicación y añadir una capa de abstracción al funcionamiento interno de los modelos de la

aplicación. Este archivo está formado por dos clases principales, *GUIControler* y *ModelController,* las cuales se relacionan de forma que GUIController utiliza instancias de la otra para interactuar con los modelos generativos y gestionar la lógica de la interfaz, accionada por la interacción del usuario de la interfaz, por lo que:

- *GUIController*: Gestiona la interacción entre la interfaz gráfica y el modelo generativo que se encuentre seleccionado, por lo que administra la selección del modelo, generación de imágenes y actualizaciones visuales de la interfaz.
- *ModelController*: Se encarga de la gestión de los modelos generativos disponibles en el sistema y la ejecución de estos.

A continuación, se seguirá por la siguiente carpeta denominada */imageGeneration* la cual contiene los siguientes archivos de código:

- *generative_model.py*: Se trata de una parte fundamental del sistema, ya que se trata del lugar donde se gestionan e implementan los modelos generativos utilizados en el sistema para llevar a cabo la generación de imágenes. Además, es importante porque es el lugar necesario de modificar para escalar el sistema en cuanto a añadir nuevos modelos o parámetros de los modelos, en el que se establece una estructura diseñada para llevar a cabo estas tareas de forma ágil. Por lo que, en resumen, proporciona la lógica para manejar diferentes modelos de generación y ejecutar los modelos de generación del sistema. Contiene las siguientes clases principales, entre otras:
  - *ModelManager:* Encargada de gestionar los modelos generativos disponibles, permite listar los modelos disponibles, cargarlos en memoria y ejecutar los modelos, además de encapsular estas funcionalidades y mapear los datos a los necesarios para ejecutar los modelos.
  - *DiffusersBaseModel:* Clase abstracta base para los modelos generativos concretos, define lógica común que comparten los modelos concretos y define los métodos abstractos que deben implementar estos modelos concretos.
  - Clases de modelos concretos: Implementan la lógica que define la clase base para poder llevar a cabo las operaciones de carga en memoria y ejecución de los modelos.

  Para acabar se resumirá el contenido de la carpeta */utils,* la cual contiene los siguientes archivos de utilidades, que sirven como apoyo para gestionar tareas concretas de la herramienta:
- *file_util.py*: Contiene funciones relacionadas con operaciones necesarias para la gestión de archivos y directorios del proyecto.
- *image_util.py:* Contiene funciones relacionadas con operaciones necesarias para la gestión de manejo de imágenes.

Capítulo 10. Conclusiones y Ampliaciones

En este capítulo se resumen las conclusiones obtenidas tras verse llevado a cabo el proyecto, además de mencionarse posibles ampliaciones que se han pensado para poder mejorar el sistema en el futuro.

10.1 Conclusiones

Tras haber llevado a cabo todas las tareas necesarias para llevar a cabo el proyecto, como producto se ha obtenido una herramienta la cual permite generar un número de imágenes determinado por el usuario, correspondiéndose con una descripción textual proporcionada por el usuario, además permitiendo la configuración de diferentes opciones, como el hardware donde se ejecutarán los modelos y diferentes parámetros de los modelos, en cuanto a estos modelos, el sistema ofrece varios modelos diferentes con los que generar las imágenes, los cuales se han implementado utilizando la librería *diffusers* de Hugging Face, la cual nos ofrece el uso de forma local de modelos de generación de imágenes basados en difusión estable, por lo que gracias a esto la aplicación es capaz de generar las imágenes con muy buenos resultados ya que los modelos basados en difusión estable se tratan de los modelos de inteligencia artificial más avanzados hoy en día, en cuanto a los enfocados a la generación de imágenes.

Por lo que, en cuanto a los resultados obtenidos hablando de calidad de las imágenes generadas se han obtenido unos resultados mejor de los esperados, ya que como idea inicial se había planteado el uso de *APIs* en línea muy conocidas mencionadas en 2.3.1.1, las cuales nos aseguraban la calidad y rendimiento de los modelos ejecutados, aunque fueron descartadas debido al alto coste en el que recaería el proyecto, ya que esté enfocado a generar un volumen muy alto de imágenes y debido al coste por cada imagen que este tipo de *APIs* establecen.

En cuanto a aspectos negativos de la librería utilizada se encuentra el rendimiento limitado en la ejecución de los modelos en equipos que no dispongan de una *GPU CUDA*, ya que la ejecución de estos modelos en la *CPU* resulta notablemente más lenta.

Por lo que, tras este estudio inicial, e integración de *APIs* de generación de imágenes se han aprendido multitud de conceptos relacionados con la inteligencia artificial, sobre todo los relacionados con modelos generativos de imágenes y en concreto con los modelos más avanzados que existen hoy en día. Entre los conceptos aprendidos sobre la inteligencia artificial generativa de imágenes, se han aprendido los diferentes enfoques de modelos para generar imágenes como la difusión estable o las redes generativas antagónicas. También se ha aprendido la arquitectura por la que están construidos estos modelos de difusión y tras llevarse a cabo la implementación del sistema se ha llegado a conocer que componentes forman estos modelos de difusión, las diferentes implementaciones existentes de este tipo de modelos, así como los diferentes parámetros que necesitan estos modelos y para acabar el proceso que llevan a cabo

estos modelos durante su ejecución desde que reciben los parámetros hasta que ofrecen un resultado en forma de imagen.

En cuanto a lo aprendido durante el desarrollo de la herramienta también hay que tener en cuenta que se ha desarrollado una interfaz gráfica de usuario utilizando *Python*, al igual que se ha utilizado *Python* para la implementación de los modelos. Por lo que durante el proceso de análisis, diseño y desarrollo de la interfaz gráfica de usuario se ha aprendido a como este tipo de interfaces de aplicaciones de escritorio son desarrolladas con la librería estándar de *Python* para este cometido, durante este proceso se ha detectado que esta librería estándar se encuentra un tanto anticuada y se han estudiado soluciones para mejorar la apariencia de la herramienta hasta llegar a la solución obtenida con muy buenos resultados y aspecto moderno y renovado. De forma que se ha profundizado y conocido las diferentes soluciones existentes para desarrollar aplicaciones de escritorio con *Python* con aspecto moderno y renovado, obteniendo mejores resultados que los esperados al inicio del proyecto donde se planteaba el uso de la librería estándar de *Python*, se pueden ver las alternativas estudiadas en 2.3.1.2.

A parte de los conocimientos aplicados y aprendidos en la parte de implementación del sistema explicada previamente, se ha realizado un extenso trabajo en el desarrollo de la presente memoria, que, para ello, se ha tenido que aplicar conocimiento obtenido a lo largo de todo el grado, relacionado con todo el proceso de la ingeniería del *software* y la dirección y gestión de proyectos, para ello se han tenido que aplicar conocimientos relacionados con las fases de análisis, diseño, implementación y pruebas de un proyecto de desarrollo *software*, además de los conceptos más relacionados con la dirección y gestión de proyectos informáticos, como llevar a cabo la planificación y presupuestos de un proyecto *software* teniendo en cuenta todas las partes implicadas para llevar a cabo un proyecto de estas características.

Teniendo en cuenta lo explicado a lo largo de esta sección, como experiencia personal, el desarrollo del proyecto me ha permitido refrescar todos los conocimientos aprendidos durante el grado relacionados con todo el ciclo de desarrollo de software y gestión de proyectos de forma práctica, aplicados a un proyecto muy interesante hoy en día donde se encuentra en auge todo lo relacionado con la inteligencia artificial y en un periodo donde se han publicado modelos generativos que proporcionan resultados de muy alta calidad.

Por lo que, para acabar, personalmente, pese a que la redacción de la memoria ha sido tediosa y durante la implementación del sistema y pruebas se han encontrado multitud de problemas y contratiempos, los conocimientos obtenidos durante este proceso han sido muy valorables y el resultado final ha resultado más satisfactorio de lo esperado, salvo por no haber incluido ciertas mejoras por falta de tiempo para ello, explicadas en 10.2.

10.2 Ampliaciones

En esta sección se presentan las ampliaciones que se ha pensado que aportarían una gran mejora al sistema desarrollado. Estas funcionalidades o ampliaciones no habían sido contempladas de forma inicial, sino que han sido detectadas durante la ejecución del proyecto por lo que podrían haberse introducido como una ampliación del alcance del proyecto, lo cual no ha sido posible debido a la falta de recursos o tiempo fijados para la realización del proyecto.

10.2.1 Botón de cancelación de generación

Actualmente la única manera que existe para cancelar un proceso de generación de imágenes es cerrar la aplicación, lo que puede empeorar la usabilidad de la herramienta y la satisfacción de los usuarios con ella. Por lo que por esta razón se propone ampliar el sistema implementando esta solución, de forma que se añada un botón de cancelación debajo del botón de generación y el cuál solamente sea visible cuando el proceso de generación se encuentre en ejecución.

Para implementar esta opción existen varias opciones, ya que para ello se debe interactuar con un hilo en ejecución en segundo plano lo puede resultar complicado, por lo que entre las soluciones planteadas la que mayor sencillez respecto a su efectividad tiene sería añadir un *flag* de cancelación el cuál será compartido entre el hilo principal de sistema y el hilo lanzado durante la ejecución de la generación de imágenes. Para ello sería necesario añadir un nuevo método en el controlador que lance el evento de cancelación iniciado desde el botón de la interfaz de usuario, tras esto se pasará el evento al controlador de modelos que gestiona el hilo de la generación y durante el bucle en el que se ejecuta la generación dentro de la implementación de los modelos se comprobará si este *flag* que transporta el evento está activo, de forma que si se encuentra activo se romperá el bucle y tras esto se terminará la generación. Para más información sobre cada uno de los elementos consultar 9.4.2.

10.2.2 Importación de modelos

Esta ampliación se corresponde con la posibilidad de ofrecer al usuario la opción de importar uno o varios modelos compatibles con la librería *diffusers*, de forma que los modelos disponibles en la aplicación puedan ser completamente extensibles para los usuarios sin la necesidad de que sea necesario ampliar el código de la aplicación por estos, lo que es posible gracias a que existe un repositorio de la comunidad desarrolladora de la librería *diffusers* en la que se pueden publicar los modelos desarrollados por los miembros de esta comunidad [50] gracias a esto existen publicados diferentes modelos en este repositorio implementados por otros desarrolladores, aunque también sería posible publicar los modelos desarrollados por uno mismo.

Por lo que sabiendo esto, la idea sería añadir un botón de importación de modelos en la interfaz, de forma que se pueda introducir el *endpoint* del modelo almacenado en el repositorio o seleccionar este modelo de un directorio local del equipo del usuario, una vez hecho esto, la propia documentación de la librería proporciona ejemplos sobre cómo es posible importar

modelos del repositorio o de un directorio local, por lo que para implementar esta ampliación sería posible seguir la siguiente documentación [51]

10.2.3 Ventana comparativa sobre modelos

Otra de las ampliaciones propuestas, es añadir una nueva ventana en la herramienta, esta ventana contendría información sobre los modelos disponibles en el sistema, entre esta información se encontrarían las ventajas y desventajas de cada modelo en cuanto a los resultados buscados por el usuario, de forma que los usuarios menos conocedores de este tipo de modelos puedan elegir qué modelo utilizar en función de sus necesidades.

Para llevar a cabo esto, sería posible añadir un pequeño botón junto al selector de modelos que abra esta ventana, además de la información explicada en el párrafo anterior también se puede añadir una tabla comparativa entre todos los modelos disponibles de forma que sea aún más fácil poder decidir qué modelo seleccionar.

Además, también se propone introducir información sobre los valores recomendados de los parámetros avanzados de cada modelo, de forma que el usuario pueda elegir los óptimos recomendados para cada modelo en función de los objetivos buscados por el usuario.

10.2.4 Selección de scheduler de modelo

Una modificación interesante en el sistema en cuanto al aumento de la flexibilidad de configuración de los modelos sería añadir una opción en el sistema que permita al usuario seleccionar desde la interfaz que *scheduler* [52] utilizará el modelo seleccionado para realizar el proceso de generación de imágenes.

Para ello se propone añadir en la interfaz un selector con los *schedulers* disponibles para utilizar con el modelo seleccionado, esta opción se mostraría una vez se seleccione un modelo y no se encuentre cargado en memoria, ya que el *scheduler* del modelo se establece una vez se carga el modelo en memoria, por lo que una vez se seleccione un modelo se cargará en memoria con el *scheduler* seleccionado, en caso de que no se seleccione ninguno el modelo se cargará en memoria con su *scheduler* predeterminado, de la misma forma que se hace actualmente en el sistema. Entonces, una vez se inicie el proceso de carga en memoria, la interfaz enviará al modelo el *scheduler* seleccionado como parámetro, para más información sobre como añadir estas modificaciones se puede consultar el apartado 9.4.2 y para consultar los diferentes *schedulers* ofrecidos por la librería y la forma de configurar los modelos con ellos se puede consultar la documentación de la librería relativa a esto en [53].

10.2.5 Versión web de la aplicación

Debido al uso extendido y conocido de internet por la gran mayoría de usuarios que utilizarán este tipo de sistema, se propone crear aplicación web del sistema, de forma que sea mucho más distribuible entre los usuarios, además de poder ejecutar los modelos en un servidor con

recursos mucho mayores que un equipo personal de un usuario. Las desventajas de esta ampliación es que existirían costes derivados del uso de servidores, en este caso se propone que, en la nube, por la facilidad de despliegue, mantenimiento del entorno y los ahorros en compra y mantenimiento del *hardware*.

Para llevar a cabo esto sería necesario alquilar un servidor en la nube, como *AWS* [54] o *Microsoft Azure* [55], además de esto sería necesario realizar el ciclo completo del desarrollo de software necesario para implementar una interfaz web, crear un entorno de servidor en la nube e integrar el código de la implementación de los modelos del sistema en esta nueva interfaz web, así como el código *backend* necesario para comunicar la web con la implementación de los modelos de generación.

La desventaja de esta ampliación se resumen en el aumento de la complejidad del sistema, así como de uso de recursos, tanto de trabajo como económicos, ya que se trataría de una modificación del alcance notablemente alta, aunque gracias al diseño del sistema la implementación de los modelos se ha desacoplado de la interfaz de usuario con la intención de poder realizar este tipo de modificaciones en el sistema, para más información sobre la modificación del sistema consultar 9.4.2.

Capítulo 11. Planificación del Proyecto y Presupuesto finales

En este capítulo se explicará la planificación y presupuesto final obtenidos, donde, además, se explicarán las diferencias existentes entre estos presupuestos y planificación finales en comparación con los estimados en el Capítulo 4.

11.1 Planificación Final

En esta sección se explicará la planificación final que se ha obtenido tras llevar a cabo las tareas correspondientes del proyecto, además se explicarán las diferencias entre la planificación inicial estimada en 4.1, donde las principales diferencias se tratarán del tiempo estimado para realizar diferentes tareas, las cuales han variado en cuanto al tiempo estimado en esta planificación inicial y lo que realmente ha llevado hacerlas, lo cual se reflejará en esta planificación final, además se ha incluido una tarea no incluida en la planificación inicial y se ha eliminado otra, relativa a el ámbito de las pruebas, que se verán reflejadas a continuación a lo largo de la presente sección.

En cuanto a la fecha de inicio del proyecto se mantendrá la misma establecida en la planificación inicial, pero en cuanto la fecha de finalización del proyecto ha variado respecto a la establecida en la final debido al cambio de horas llevadas a cabo para realizar las tareas del proyecto, en concreto, el proyecto tendrá como fecha de inicio el 03/06/2024 y tendrá como fecha de fin el 03/11/2024, por lo que la duración del proyecto será de 5 meses, aunque de trabajo serían aproximadamente 4 meses debido al parón de agosto explicado en la planificación inicial, ya que el calendario establecido en cuanto a días laborables, horas de trabajo semanales, etc., será el mismo que se ha establecido en la planificación inicial.

La fecha de finalización ha variado debido a la variación de horas de trabajo de diferentes tareas como se ha explicado previamente, por lo que la realización del proyecto ha llevado más tiempo del esperado, en concreto ha llevado 27 horas más de las esperadas.

Para empezar, en las tareas de documentación se han realizado 22,5 horas de trabajo más que las estimadas en la planificación inicial, en cuanto a las tareas relacionadas con el desarrollo del software han llevado 8 horas de trabajo más que las estimadas, luego en las tareas de realización de las pruebas se han trabajado 2,5 horas más de las estimadas. El resto de las tareas, empezando por el estudio inicial se han trabajado las mismas horas que las estimadas inicialmente y en cuanto a las tareas relacionadas con el cierre de proyecto, se han trabajado finalmente la mitad de las horas de las estimadas, es decir, 6 horas de trabajo, de forma que finalmente la estimación inicial ha sido más optimista de lo que finalmente se ha trabajado y por tanto ha resultado en horas extra de trabajo en comparación de lo inicialmente estimado.

En cuanto a las causas de estas diferencias respecto a la planificación inicial y la final, comenzando por las tareas de documentación, la causa del incremento de horas de trabajo

realizadas se debe de forma general en todas las tareas que han llevado más tiempo del esperado a una estimación demasiado optimista en la planificación inicial, ya que no se han encontrado problemas durante la realización de estas sino que solamente pequeñas correcciones que no afectaron de forma considerable en la realización de cada tarea, aunque hubo alguna tarea que llevó menos tiempo del esperado no ha resultado relevante en el saldo final de horas ya que han sido pocas en las que ha sucedido, además de que ha sido poco el tiempo menor de trabajo realizado respecto al estimado.

En cuanto a las tareas relacionadas con el desarrollo del *software*, se ha trabajado más horas de las estimadas debido a que han surgido problemas durante el proceso de desarrollo no contemplados en las fases previas, los más relevantes que han producido un mayor desfase en las horas trabajadas respecto a las estimadas se encuentran explicados en 7.4.1. En concreto las tareas de desarrollo que han llevado más tiempo del estimado son la de cargar el modelo en memoria que ha llevado 1 hora de trabajo más, la de generar imágenes que ha llevado 8 horas más que las estimadas, modificar opciones avanzadas que ha llevado el doble de lo estimado es decir 9 horas más y los refinamientos finales que han llevado 1 hora más de las estimadas, por lo que debido a estos problemas no esperados y refactorizaciones de código resultantes de estos problemas han causado este incremento de horas respecto de las esperadas.

En cuanto a la realización de las pruebas, estas tareas han llevado 2,5 horas más de las estimadas debido a que las pruebas unitarias han llevado 9 horas más de las estimadas, a causa de la complejidad en la implementación de las pruebas unitarias automatizadas relacionadas con procesos asíncronos, lo cual no había sido tenido en cuenta, por otro lado las demás tareas han durado algo menos de lo estimado aunque debido a las pruebas unitarias la realización de las pruebas ha llevado más tiempo del esperado.

Para acabar, las tareas de cierre de proyecto han llevado 6 horas de trabajo menos de lo estimado, aunque debido a las horas del aumento de las horas de trabajo en las demás tareas no ha sido relevante para reducir las horas de trabajo finales respecto a las estimadas inicialmente.

Por lo que teniendo en cuenta toda esta información las horas de trabajo totales para llevar a cabo el proyecto son de 362 horas, por lo que se ha obtenido un número de horas de trabajo mayor que el estimado en la planificación inicial de 335 horas.

A continuación, se expone el resumen de las actividades principales de la estructura de desglose de trabajo, junto a las horas requeridas para llevar a cabo cada actividad, de la misma forma que se ha realizado en 4.1, de forma que podremos utilizar ambos capítulos a modo de comparativa entre lo estimado inicialmente y lo que realmente se ha realizado.

EDT	Nombre de la tarea	Trabajo
1.1	Estudio inicial	87 horas
1.2	Documentación del proyecto	119 horas

1.3	Desarrollo del software	131 horas
1.4	Realización de las pruebas	19 horas
1.5	Cierre de proyecto	6 horas

Figura 11.1 Planificación final resumen de tareas

11.1.1 Estudio Inicial

A continuación, se muestran las tareas relacionadas con el estudio inicial junto a las horas de trabajo realizadas durante el proyecto, se puede usar a modo de comparativa con las estimadas en 4.1.1. Como se puede ver estas tareas coinciden con las estimadas en la planificación inicial.

EDT	Nombre de tarea	Fecha comienzo	Fecha fin	Trabajo
1.1	Estudio inicial	Mon 03/06/24	Mon 01/07/24	87 horas
1.1.1	Planificación inicial	Mon 03/06/24	Tue 04/06/24	6 horas
1.1.2	Investigación inicial	Wed 05/06/24	Mon 01/07/24	81 horas
1.1.2.1	Investigación sistemas equivalentes	Wed 05/06/24	Fri 07/06/24	9 horas
1.1.2.2	Investigación apis de generación	Sat 08/06/24	Fri 21/06/24	42 horas
1.1.2.3	Investigación librerías IU	Sat 22/06/24	Mon 24/06/24	9 horas
1.1.2.4	Investigación arquitectura del sistema	Tue 25/06/24	Sun 30/06/24	16 horas
1.1.2.5	Lectura de trabajos de otros alumnos	Sun 30/06/24	Mon 01/07/24	5 horas

Figura 11.2 Planificación final tareas estudio inicial

11.1.2 Documentación del proyecto

A continuación, se describen las tareas junto a las horas de trabajo realizadas para llevar a cabo cada una de ellas, en concreto las tareas relativas a la realización de la documentación del proyecto, a comparar con las estimadas en 4.1.2.

EDT	Nombre tarea	Fecha inicio	Fecha fin	Trabajo
1.2	Documentación del proyecto	Tue 02/07/24	Fri 01/11/24	119 horas
1.2.1	Memoria del proyecto	Tue 02/07/24	Tue 02/07/24	1,5 horas
1.2.1.1	Resumen de la motivación, objetivos y alcance del proyecto	Tue 02/07/24	Tue 02/07/24	1 horas

1.2.1.2	Resumen de todos los aspectos	Tue 02/07/24	Tue 02/07/24	0,5 horas
1.2.2	Introducción	Tue 02/07/24	Sat 06/07/24	11,5 horas
1.2.2.1	Justificación del proyecto	Tue 02/07/24	Tue 02/07/24	1 horas
1.2.2.2	Objetivos del proyecto	Tue 02/07/24	Tue 02/07/24	0,5 horas
1.2.2.3	Estudio de la situación actual	Wed 03/07/24	Sat 06/07/24	10 horas
1.2.3	Aspectos teóricos	Sat 06/07/24	Sun 07/07/24	3 horas
1.2.4	Planificación inicial del proyecto	Sat 06/07/24	Tue 09/07/24	10 horas
1.2.5	Análisis	Tue 09/07/24	Sun 21/07/24	36 horas
1.2.5.1	Definición del sistema	Tue 09/07/24	Tue 09/07/24	1 horas
1.2.5.2	Requisitos del sistema	Wed 10/07/24	Fri 12/07/24	7 horas
1.2.5.3	Identificación de los subsistemas	Fri 12/07/24	Fri 12/07/24	0,5 horas
1.2.5.4	Diagrama de clases preliminar	Fri 12/07/24	Mon 15/07/24	10 horas
1.2.5.5	Análisis de casos de uso y escenarios	Mon 15/07/24	Thu 18/07/24	8 horas
1.2.5.6	Relación casos de uso y escenarios	Thu 18/07/24	Thu 18/07/24	0,5 horas
1.2.5.7	Análisis de interfaces de usuario	Thu 18/07/24	Sat 20/07/24	5 horas
1.2.5.8	Especificación del plan de pruebas	Sat 20/07/24	Sun 21/07/24	4 horas
1.2.6	Diseño	Sun 21/07/24	Thu 05/09/24	34 horas

1.2.6.1	Arquitectura del sistema	Sun 21/07/24	Wed 24/07/24	8 horas
1.2.6.2	Diseño de clases	Wed 24/07/24	Fri 26/07/24	6 horas
1.2.6.3	Diagramas de interacción y estados	Fri 26/07/24	Sat 27/07/24	5 horas
1.2.6.4	Diseño de la interfaz	Sun 28/07/24	Tue 03/09/24	8 horas
1.2.6.5	Especificación técnica del plan de pruebas	Tue 03/09/24	Thu 05/09/24	7 horas
1.2.7	Implementación del sistema	Sat 19/10/24	Mon 21/10/24	5,5 horas
1.2.7.1	Estándares y normas seguidos	Sat 19/10/24	Sat 19/10/24	0,5 horas
1.2.7.2	Lenguajes de programación	Sat 19/10/24	Sun 20/10/24	2 horas
1.2.7.3	Herramientas y programas usados	Sun 20/10/24	Sun 20/10/24	1 horas
1.2.7.4	Creación del sistema	Sun 20/10/24	Mon 2/10/24	2 horas
1.2.8	Desarrollo de las pruebas	Fri 25/10/24	Sun 27/10/24	3,5 horas
1.2.8.1	Pruebas unitarias	Fri 25/10/24	Sat 26/10/24	2 horas
1.2.8.2	Pruebas de usabilidad y accesibilidad	Sat 26/10/24	Sat 26/10/24	1 horas
1.2.8.3	Pruebas de rendimiento	Sun 27/10/24	Sun 27/10/24	0,5 horas
1.2.9	Manuales del sistema	Sun 27/10/24	Tue 29/10/24	5,5 horas
1.2.9.1	Manual de instalación	Sun 27/10/24	Mon 28/10/24	1 horas
1.2.9.2	Manual de ejecución	Mon 28/10/24	Mon 28/10/24	0,5 horas

1.2.9.3	Manual de usuario	Mon 28/10/24	Mon 28/10/24	2 horas
1.2.9.4	Manual del programador	Tue 29/10/24	Tue 29/10/24	2 horas
1.2.10	Conclusiones y ampliaciones	Tue 29/10/24	Wed 30/10/24	2 horas
1.2.11	Planificación y presupuestos finales	Wed 30/10/24	Thu 31/10/24	4 horas
1.2.12	Referencias bibliográficas	Thu 31/10/24	Thu 31/10/24	0,5 horas
1.2.13	Apéndices	Thu 31/10/24	Fri 01/11/24	2 horas

Figura 11.3 Planificación final tareas documentación del proyecto

11.1.3 Desarrollo del software

A continuación, se muestra el resumen de forma similar a las tareas estimadas en 4.1.3, relativas al desarrollo de *software* y el trabajo realizado en cada una durante el proyecto.

EDT	Nombre tarea	Fecha inicio	Fecha Fin	Trabajo
1.3	Desarrollo del software	Fri 06/09/24	Sar 19/10/24	131 horas
1.3.1	Interfaz de usuario	Fri 06/09/24	Tue 17/09/24	35 horas
1.3.2	Cargar modelo en memoria	Tue 17/09/24	Tue 24/09/24	20 horas
1.3.3	Generar imágenes	Tue 24/09/24	Thu 10/10/24	50 horas
1.3.4	Modificar opciones avanzadas	Fri 11/10/24	Wed 16/10/24	18 horas
1.3.5	Refinamientos finales	Thu 17/10/24	Sat 19/10/24	8 horas

Figura 11.4 Planificación final tareas desarrollo de software

11.1.4 Realización de las pruebas

En esta sección se recopilan las tareas relacionadas con la realización e implementación de las diferentes pruebas a realizar en el sistema junto a las horas requeridas para llevarlas a cabo a comparar con las estimadas en 4.1.4.

EDT	Nombre tarea	Fecha inicio	Fecha fin	Trabajo
1.4	Realización de las pruebas	Mon 21/10/24	Sun 27/10/24	19 hrs
1.4.1	Pruebas unitarias	Mon 21/10/24	Fri 25/10/24	12 hrs
1.4.2	Pruebas de usabilidad y accesibilidad	Fri 25/10/24	Sat 26/10/24	3 hrs
1.4.3	Pruebas de rendimiento	Sat 26/10/24	Sun 27/10/24	2 hrs
1.4.4	Corrección defectos encontrados	Sun 27/10/24	Sun 27/10/24	2 hrs

Figura 11.5 Planificación final tareas realización de las pruebas

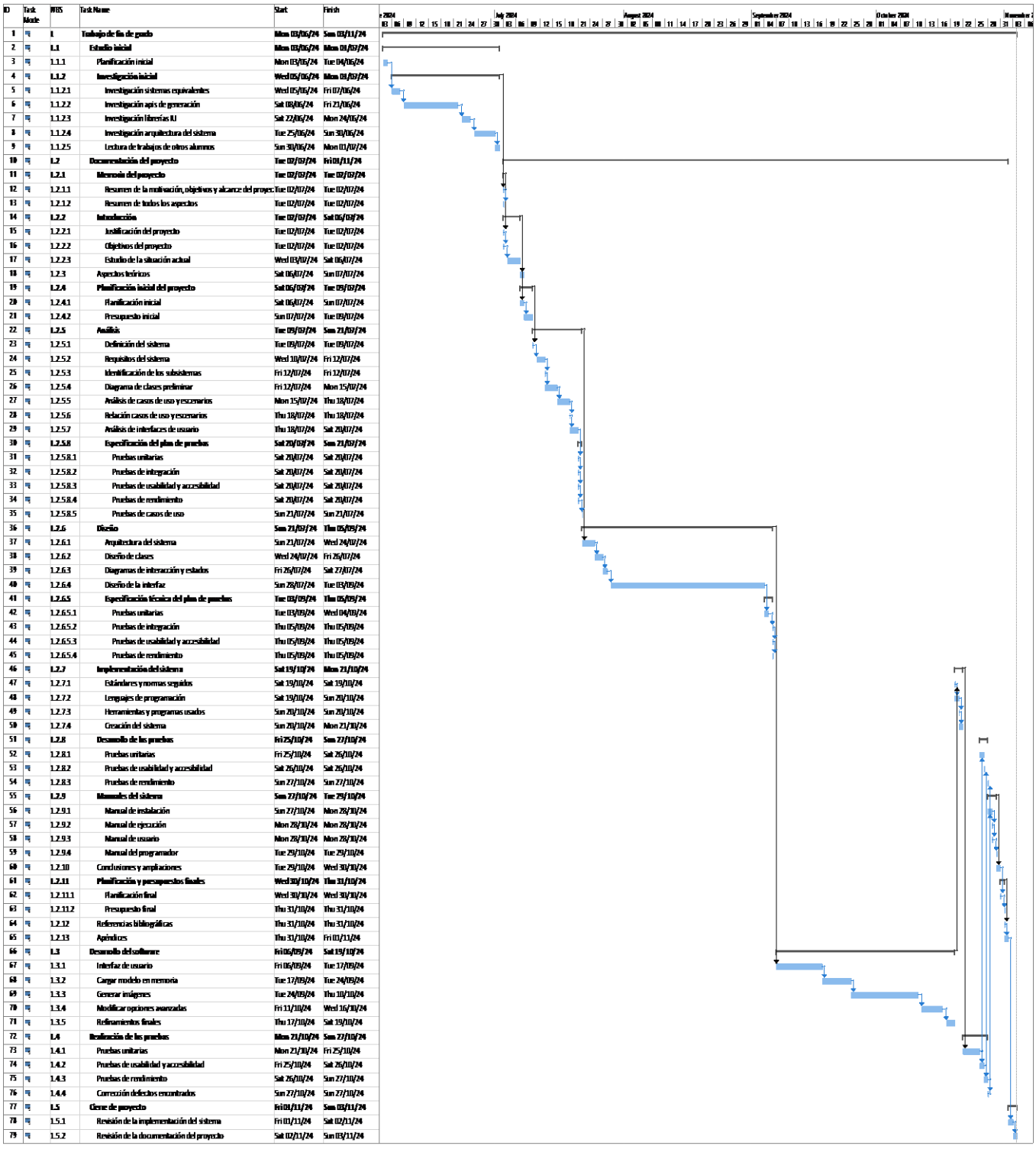
11.1.5 Cierre del proyecto

Para acabar, se exponen las tareas relativas al cierre del proyecto, de la misma forma que las estimadas en 4.1.5.

EDT	Nombre tarea	Fecha inicio	Fecha fin	Trabajo
1.5	Cierre de proyecto	Fri 01/11/24	Sun 03/11/24	6 hrs
1.5.1	Revisión de la implementación del sistema	Fri 01/11/24	Sat 02/11/24	2 hrs
1.5.2	Revisión de la documentación del proyecto	Sat 02/11/24	Sun 03/11/24	4 hrs

Figura 11.6 Planificación final tareas de cierre de proyecto

Para acabar se expone el diagrama de *Gantt* del proyecto, junto a todas las tareas de este, exponiendo así un resumen de la planificación final realizada en el proyecto mostrando las fechas de inicio y fin, así como su representación en el diagrama de *Gantt*.



ID	Task Mode	WBS	Task Name	Start	Finish
1		1	Trabajo de fin de grado	Mon 03/06/24	Sun 03/11/24
2		1.1	Estudio inicial	Mon 03/06/24	Mon 08/07/24
3		1.1.1	Planificación inicial	Mon 03/06/24	Tue 04/06/24
4		1.1.2	Investigación inicial	Wed 05/06/24	Mon 08/07/24
5		1.1.2.1	Investigación sistemas equivalentes	Wed 05/06/24	Fri 07/06/24
6		1.1.2.2	Investigación apis de generación	Sat 08/06/24	Fri 21/06/24
7		1.1.2.3	Investigación librerías IU	Sat 22/06/24	Mon 24/06/24
8		1.1.2.4	Investigación arquitectura del sistema	Tue 25/06/24	Sun 30/06/24
9		1.1.2.5	Lectura de trabajos de otros alumnos	Sun 30/06/24	Mon 01/07/24
10		1.2	Documentación del proyecto	Tue 02/07/24	Fri 01/11/24
11		1.2.1	Memoria del proyecto	Tue 02/07/24	Tue 02/07/24
12		1.2.1.1	Resumen de la motivación, objetivos y alcance del proyec	Tue 02/07/24	Tue 02/07/24
13		1.2.1.2	Resumen de todos los aspectos	Tue 02/07/24	Tue 02/07/24
14		1.2.2	Introducción	Tue 02/07/24	Sat 06/07/24
15		1.2.2.1	Justificación del proyecto	Tue 02/07/24	Tue 02/07/24
16		1.2.2.2	Objetivos del proyecto	Tue 02/07/24	Tue 02/07/24
17		1.2.2.3	Estudio de la situación actual	Wed 03/07/24	Sat 06/07/24
18		1.2.3	Aspectos teóricos	Sat 06/07/24	Sun 07/07/24
19		1.2.4	Planificación inicial del proyecto	Sat 06/08/24	Tue 09/07/24
20		1.2.4.1	Planificación inicial	Sat 06/07/24	Sun 07/07/24
21		1.2.4.2	Presupuesto inicial	Sun 07/07/24	Tue 09/07/24
22		1.2.5	Análisis	Tue 09/08/24	Sun 21/08/24
23		1.2.5.1	Definición del sistema	Tue 09/07/24	Tue 09/07/24
24		1.2.5.2	Requisitos del sistema	Wed 10/07/24	Fri 12/07/24
25		1.2.5.3	Identificación de los subsistemas	Fri 12/07/24	Fri 12/07/24
26		1.2.5.4	Diagrama de clases preliminar	Fri 12/07/24	Mon 15/07/24
27		1.2.5.5	Análisis de casos de uso y escenarios	Mon 15/07/24	Thu 18/07/24
28		1.2.5.6	Relación casos de uso y escenarios	Thu 18/07/24	Thu 18/07/24
29		1.2.5.7	Análisis de interfaces de usuario	Thu 18/07/24	Sat 20/07/24
30		1.2.5.8	Especificación del plan de pruebas	Sat 20/08/24	Sun 21/08/24
31		1.2.5.8.1	Pruebas unitarias	Sat 20/07/24	Sat 20/07/24
32		1.2.5.8.2	Pruebas de integración	Sat 20/07/24	Sat 20/07/24
33		1.2.5.8.3	Pruebas de usabilidad y accesibilidad	Sat 20/07/24	Sat 20/07/24
34		1.2.5.8.4	Pruebas de rendimiento	Sat 20/07/24	Sat 20/07/24
35		1.2.5.8.5	Pruebas de casos de uso	Sun 21/07/24	Sun 21/07/24
36		1.2.6	Diseño	Sun 21/08/24	Thu 05/09/24
37		1.2.6.1	Arquitectura del sistema	Sun 21/07/24	Wed 24/07/24
38		1.2.6.2	Diseño de clases	Wed 24/07/24	Fri 26/07/24
39		1.2.6.3	Diagramas de interacción y estados	Fri 26/07/24	Sat 27/07/24
40		1.2.6.4	Diseño de la interfaz	Sun 28/07/24	Tue 03/09/24
41		1.2.6.5	Especificación técnica del plan de pruebas	Tue 03/09/24	Thu 05/09/24
42		1.2.6.5.1	Pruebas unitarias	Tue 03/09/24	Wed 04/09/24
43		1.2.6.5.2	Pruebas de integración	Thu 05/09/24	Thu 05/09/24
44		1.2.6.5.3	Pruebas de usabilidad y accesibilidad	Thu 05/09/24	Thu 05/09/24
45		1.2.6.5.4	Pruebas de rendimiento	Thu 05/09/24	Thu 05/09/24
46		1.2.7	Implementación del sistema	Sat 19/10/24	Mon 21/10/24
47		1.2.7.1	Estándares y normas seguidos	Sat 19/10/24	Sat 19/10/24
48		1.2.7.2	Lenguajes de programación	Sat 19/10/24	Sun 20/10/24
49		1.2.7.3	Herramientas y programas usados	Sun 20/10/24	Sun 20/10/24
50		1.2.7.4	Creación del sistema	Sun 20/10/24	Mon 21/10/24
51		1.2.8	Desarrollo de las pruebas	Fri 25/10/24	Sun 27/10/24
52		1.2.8.1	Pruebas unitarias	Fri 25/10/24	Sat 26/10/24
53		1.2.8.2	Pruebas de usabilidad y accesibilidad	Sat 26/10/24	Sat 26/10/24
54		1.2.8.3	Pruebas de rendimiento	Sun 27/10/24	Sun 27/10/24
55		1.2.9	Manuales del sistema	Sun 27/10/24	Tue 29/10/24
56		1.2.9.1	Manual de instalación	Sun 27/10/24	Mon 28/10/24
57		1.2.9.2	Manual de ejecución	Mon 28/10/24	Mon 28/10/24
58		1.2.9.3	Manual de usuario	Mon 28/10/24	Mon 28/10/24
59		1.2.9.4	Manual del programador	Tue 29/10/24	Tue 29/10/24
60		1.2.10	Conclusiones y ampliaciones	Tue 29/10/24	Wed 30/10/24
61		1.2.11	Planificación y presupuestos finales	Wed 30/10/24	Thu 31/10/24
62		1.2.11.1	Planificación final	Wed 30/10/24	Wed 30/10/24
63		1.2.11.2	Presupuesto final	Thu 31/10/24	Thu 31/10/24
64		1.2.12	Referencias bibliográficas	Thu 31/10/24	Thu 31/10/24
65		1.2.13	Apéndices	Thu 31/10/24	Fri 01/11/24
66		1.3	Desarrollo del software	Fri 05/09/24	Sat 19/10/24
67		1.3.1	Interfaz de usuario	Fri 06/09/24	Tue 17/09/24
68		1.3.2	Cargar modelo en memoria	Tue 17/09/24	Tue 24/09/24
69		1.3.3	Generar imágenes	Tue 24/09/24	Thu 10/10/24
70		1.3.4	Modificar opciones avanzadas	Fri 11/10/24	Wed 16/10/24
71		1.3.5	Refinamientos finales	Thu 17/10/24	Sat 19/10/24
72		1.4	Realización de las pruebas	Mon 21/10/24	Sun 27/10/24
73		1.4.1	Pruebas unitarias	Mon 21/10/24	Fri 25/10/24
74		1.4.2	Pruebas de usabilidad y accesibilidad	Fri 25/10/24	Sat 26/10/24
75		1.4.3	Pruebas de rendimiento	Sat 26/10/24	Sun 27/10/24
76		1.4.4	Corrección defectos encontrados	Sun 27/10/24	Sun 27/10/24
77		1.5	Cierre de proyecto	Fri 08/11/24	Sun 03/11/24
78		1.5.1	Revisión de la implementación del sistema	Fri 01/11/24	Sat 02/11/24
79		1.5.2	Revisión de la documentación del proyecto	Sat 02/11/24	Sun 03/11/24

Figura 11.7 Planificación final Diagrama de Gantt

.

11.2 Presupuesto Final

En esta sección se detalla el presupuesto final obtenido, para ello se tendrá en cuenta la planificación final descrita en 11.1 y la definición de empresa definida en la sección del presupuesto inicial en 4.2.1, ya que, en este caso, para el presupuesto final no habrá ninguna variación en la definición de empresa final respecto a la inicial por lo que tomaremos como referencia la detallada en la sección referenciada previamente.

Las diferencias entre el presupuesto final y el inicial se resumen solamente en el incremento de horas de trabajo en las diferentes tareas explicadas en la planificación final, por lo tanto, estos aumentos han afectado a los costes finales establecidos para llevar a cabo cada tarea. Por lo que el coste final del proyecto ha sido de 15.376,33€, en comparación con el coste inicial estimado de 13.935,37€, como resultado ha habido un incremento final en el presupuesto de costes de 1440,96€.

Pero si tenemos en cuenta el presupuesto para el cliente calculado en 4.2.3, tras aplicar impuestos, el valor facturado al cliente es de 21.077,24€ por lo que aún podríamos obtener unos beneficios respecto al coste final y el valor facturado al cliente de 5700,91€, aunque debido a las estimaciones demasiado optimistas en la planificación inicial los beneficios serán menores que los inicialmente estimados.

Teniendo en cuenta el aumento de horas de realización del proyecto debido a lo explicado en 11.1, se traslada en sobrecostes en el presupuesto en las diferentes partidas respecto a las partidas estimadas en la planificación inicial, concretamente ente en la partida de documentación del proyecto ha habido unos sobrecostes de 1065,9€, por otro lado en la partida de desarrollo de *software* ha habido unos sobrecostes de 272,24€, en cuanto a la partida de realización de las pruebas nos encontramos con un sobrecoste de 102,82€ y en cuanto al cierre del proyecto, ya que la persona asignada no proporciona horas productivas según lo definido en 4.2.1, aunque se han reducido las horas de trabajo no ha surtido efecto en el presupuesto y para acabar, la partida de estudio inicial no ha experimentado ningún tipo de cambio en el presupuesto ni en la planificación inicial, por lo que no ha aplicado cambios en el presupuesto final respecto al inicial, entonces, si sumamos estos sobrecostes obtenemos el valor del sobrecoste total obtenido en el presupuesto final respecto al estimado de forma inicial.

Entonces, teniendo en cuenta lo mencionado, a continuación, se muestran las diferentes partidas y elementos del presupuesto final actualizados teniendo en cuenta las variaciones de la planificación final respecto a la inicial y la definición de empresa mencionada.

11.2.1 Presupuesto de costes final

A continuación, se describe el presupuesto de costes final elaborado para llevar a cabo la realización completa del proyecto, primeramente, se desglosarán las partidas incluidas en este presupuesto y finalmente se mostrará la partida de resumen de este presupuesto de costes final con todas las partidas incluidas en este.

En este presupuesto se incluirán las mismas partidas descritas en 4.2.2, pero con los datos actualizados para este caso, teniendo en cuenta lo descrito en la sección padre anterior 11.2,

por lo que a continuación, se muestran las diferentes partidas desarrolladas de este presupuesto de costes final.

Estudio inicial													
l1	l2	l3	l4	l5	Descripción	Cantidad	Unidades	Precio	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1					**Estudio inicial**								**4.737,15 €**
	1				**Planificación inicial**							**326,70 €**	
		1			Jefe de equipo	6	Horas	54,45 €					
	2				**Investigación inicial**							**4.410,45 €**	
		1			**Investigación sistemas equivalentes**						490,05 €		
			1		Jefe de equipo	9	Horas	54,45 €					
		2			**Investigación apis de generación**						2.286,90 €		
			1		Jefe de equipo	42	Horas	54,45 €					
		3			**Investigación librerías IU**						490,05 €		
			1		Jefe de equipo	9	Horas	54,45 €					
		4			**Investigación arquitectura del sistema**						871,20 €		
			1		Jefe de equipo	16	Horas	54,45 €					
		5			**Lectura de trabajos de otros alumnos**						272,25 €		
			1		Jefe de equipo	5	Horas	54,45 €					

Figura 11.8 Presupuesto costes final partida estudio inicial

Documentación del proyecto															
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Documentación del proyecto**									**5.488,27 €**
	1					**Memoria del proyecto**								81,68 €	
		1				**Resumen de la motivación, objetivos y alcance del proyecto**							54,45 €		
			1			Jefe de equipo	1	Horas	54,45 €						
		2				**Resumen de todos los aspectos**									
			1			Jefe de equipo	0,5	Horas	54,45 €				27,23 €		
	2					**Introducción**								626,18 €	
		1				**Justificación del proyecto**							54,45 €		
			1			Jefe de equipo	1	Horas	54,45 €						
		2				**Objetivos del proyecto**							27,23 €		
			1			Jefe de equipo	0,5	Horas	54,45 €						
		3				**Estudio de la situación actual**							544,50 €		
			1			Jefe de equipo	10	Horas	54,45 €						
	3					**Aspectos teóricos**								163,35 €	
			1			Jefe de equipo	3	Horas	54,45 €						
	4					**Planificación inicial del proyecto**								544,50 €	
		1				**Planificación inicial**							217,80 €		
			1			Jefe de equipo	4	Horas	54,45 €						
		2				**Presupuesto inicial**							326,70 €		
			1			Jefe de equipo	6	Horas	54,45 €						
	5					**Análisis**								1.453,92 €	
		1				**Definición del sistema**							40,84 €		
			1			Analista Software	1	Horas	40,84 €						
		2				**Requisitos del sistema**							285,88 €		
			1			Analista Software	7	Horas	40,84 €						
		3				**Identificación de los subsistemas**							20,42 €		
			1			Analista Software	0,5	Horas	40,84 €						
		4				**Diagrama de clases preliminar**							408,40 €		
			1			Analista Software	10	Horas	40,84 €						
		5				**Análisis de casos de uso y escenarios**							326,72 €		
			1			Analista Software	8	Horas	40,84 €						
		6				**Relación casos de uso y escenarios**							20,42 €		
			1			Analista Software	0,5	Horas	40,84 €						
		7				**Análisis de interfaces de usuario**							204,20 €		
			1			Analista Software	5	Horas	40,84 €						
		8				**Especificación del plan de pruebas**							147,04 €		
			1			**Pruebas unitarias**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
			2			**Pruebas de integración**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
			3			**Pruebas de usabilidad y accesibilidad**						18,38 €			
				1		**QA Tester**	0,5	Horas	36,76 €						
			4			**Pruebas de rendimiento**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
			5			**Pruebas de casos de uso**						73,52 €			
						QA Tester	2	Horas	36,76 €						
	6					**Diseño**								1.543,87 €	
		1				**Arquitectura del sistema**							381,20 €		
			1			Ingeniero de software	8	Horas	47,65 €						
		2				**Diseño de clases**							285,90 €		
			1			Ingeniero de software	6	Horas	47,65 €						
		3				**Diagramas de interacción y estados**							238,25 €		
			1			Ingeniero de software	5	Horas	47,65 €						
		4				**Diseño de la interfaz**							381,20 €		
			1			Ingeniero de software	8	Horas	47,65 €						
		5				**Especificación técnica del plan de pruebas**							257,32 €		
			1			**Pruebas unitarias**						147,04 €			
				1		QA Tester	4	Horas	36,76 €						
			2			**Pruebas de integración**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
			3			**Pruebas de usabilidad y accesibilidad**						73,52 €			
				1		QA Tester	2	Horas	36,76 €						
			4			**Pruebas de rendimiento**						18,38 €			
				1		QA Tester	0,5	Horas	36,76 €						
	7					**Implementación del sistema**								262,08 €	
		1				**Estándares y normas seguidos**							23,83 €		
			1			Ingeniero de software	0,5	Horas	47,65 €						
		2				**Lenguajes de programación**							95,30 €		
			1			Ingeniero de software	2	Horas	47,65 €						
		3				**Herramientas y programas usados**							47,65 €		
			1			Ingeniero de software	1	Horas	47,65 €						
		4				**Creación del sistema**							95,30 €		
			1			Ingeniero de software	2	Horas	47,65 €						
	8					**Desarrollo de las pruebas**								128,66 €	
		1				**Pruebas unitarias**							73,52 €		
			1			**QA Tester**	2	Horas	36,76 €						
		2				**Pruebas de usabilidad y accesibilidad**							36,76 €		
			1			QA Tester	1	Horas	36,76 €						
		3				**Pruebas de rendimiento**							18,38 €		
			1			QA Tester	0,5	Horas	36,76 €						
	9					**Manuales del sistema**								221,22 €	
		1				**Manual de instalación**							47,65 €		
			1			Ingeniero de software	1	Horas	47,65 €						
		2				**Manual de ejecución**							23,83 €		
			1			Ingeniero de software	0,5	Horas	47,65 €						
		3				**Manual de usuario**							81,68 €		
			1			Analista Software	2	Horas	40,84 €						
		4				**Manual del programador**							68,06 €		
			1			Desarrollador python	2	Horas	34,03 €						
	10					**Conclusiones y ampliaciones**								108,90 €	
			1			Jefe de equipo	2	Horas	54,45 €						
	11					**Planificación y presupuestos finales**								217,80 €	
		1				**Planificación final**							108,90 €		
			1			Jefe de equipo	2	Horas	54,45 €						
		2				**Presupuesto final**							108,90 €		
			1			Jefe de equipo	2	Horas	54,45 €						
	12					**Referencias bibliográficas**								27,23 €	
		1				Jefe de equipo	0,5	Horas	54,45 €						
	13					**Apéndices**								108,90 €	
		1				Jefe de equipo	2	Horas	54,45 €						

Figura 11.9 Presupuesto costes final partida documentación

Desarrollo del software														
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Desarrollo del software**									**4.457,93 €**
	1					**Interfaz de usuario**								1.191,05 €	
						Desarrollador Python	35	Horas	34,03 €						
	2					**Cargar modelo en memoria**								680,60 €	
						Desarrollador Python	20	Horas	34,03 €						
	3					**Generar imágenes**								1.701,50 €	
						Desarrollador Python	50	Horas	34,03 €						
	4					**Modificar opciones avanzadas**								612,54 €	
						Desarrollador Python	18	Horas	34,03 €						
	5					**Refinamientos finales**								272,24 €	
						Desarrollador Python	8	Horas	34,03 €						

Figura 11.10 Presupuesto costes final partida desarrollo de software

Realización de las pruebas														
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Realización de las pruebas**									**692,98 €**
	1					**Pruebas unitarias**								441,12 €	
		1				QA Tester	12	Horas	36,76 €						
	2					**Pruebas de usabilidad y accesibilidad**								110,28 €	
		1				QA Tester	3	Horas	36,76 €						
	3					**Pruebas de rendimiento**								73,52 €	
		1				QA Tester	2	Horas	36,76 €						
	4					**Corrección defectos encontrados**								68,06 €	
		1				Desarrollador Python	2	Horas	34,03 €						

Figura 11.11 Presupuesto costes final partida realización pruebas

Cierre del proyecto															
l1	l2	l3	l4	l5	l6	Descripción	Cantidad	Unidades	Precio	Subtotal (6)	Subtotal (5)	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
1						**Cierre del proyecto**									**0,00 €**
	1					**Revisión de la implementación del sistema**								0,00 €	
		1				Jefe de proyecto	2	Horas	0,00 €						
	2					**Revisión de la documentación del proyecto**								0,00 €	
		1				Jefe de proyecto	4	Horas	0,00 €						

Figura 11.12 Presupuesto costes final partida cierre del proyecto

Finalmente se muestra a continuación el presupuesto de costes final agregado y el resumen del presupuesto final de costes.

Presupuesto de costes				
Código	**Item**	**Partida**	**Importe**	**Total**
1		**Estudio inicial**		**4.737,15 €**
	1	**Planificación inicial**	326,70 €	
	2	**Investigación inicial**	4.410,45 €	
2		**Documentación del proyecto**		**5.488,27 €**
	1	**Memoria del proyecto**	81,68 €	
	2	**Introducción**	626,18 €	
	3	**Aspectos teóricos**	163,35 €	
	4	**Planificación inicial del proyecto**	544,50 €	
	5	**Análisis**	1.453,92 €	
	6	**Diseño**	1.543,87 €	
	7	**Implementación del sistema**	262,08 €	
	8	**Desarrollo de las pruebas**	128,66 €	
	9	**Manuales del sistema**	221,22 €	
	10	**Conclusiones y ampliaciones**	108,90 €	
	11	**Planificación y presupuestos finales**	217,80 €	
	12	**Referencias bibliográficas**	27,23 €	
	13	**Apéndices**	108,90 €	
3		**Desarrollo del software**		**4.457,93 €**
	1	**Interfaz de usuario**	1.191,05 €	
	2	**Cargar modelo en memoria**	680,60 €	
	3	**Generar imágenes**	1.701,50 €	
	4	**Modificar opciones avanzadas**	612,54 €	
	5	**Refinamientos finales**	272,24 €	
4		**Realización de las pruebas**		**692,98 €**
	1	**Pruebas unitarias**	441,12 €	
	3	**Pruebas de usabilidad y accesibilidad**	110,28 €	
	4	**Pruebas de rendimiento**	73,52 €	
	5	**Corrección defectos encontrados**	68,06 €	
5		**Cierre del proyecto**		**0,00 €**
	1	**Revisión de la implementación del sistema**	0,00 €	
	2	**Revisión de la documentación del proyecto**	0,00 €	
Total costes				**15.376,33 €**

Figura 11.13 Presupuesto de costes final agregado

Resumen presupuesto de costes		
Cod	**Partida**	**Total**
1	**Estudio inicial**	**4.737,15 €**
2	**Documentación del proyecto**	**5.488,27 €**
3	**Desarrollo del software**	**4.457,93 €**
4	**Realización de las pruebas**	**692,98 €**
5	**Cierre del proyecto**	**€ -**
Total Coste		**15.376,33 €**

Figura 11.14 Presupuesto de costes final resumen

Capítulo 12. Referencias Bibliográficas

12.1 Referencias en Internet

Páginas Web consultadas para cualquier aspecto relacionado con el desarrollo del sistema o su documentación.

[1] "Microsoft Designer - Stunning designs in a flash." Accessed: Jan. 16, 2025. [Online]. Available: https://designer.microsoft.com/

[2] "ChatGPT." Accessed: Jan. 16, 2025. [Online]. Available: https://chatgpt.com/

[3] "Midjourney." Accessed: Jan. 16, 2025. [Online]. Available: https://www.midjourney.com/home

[4] "Image generation - OpenAI API." Accessed: Jan. 14, 2025. [Online]. Available: https://platform.openai.com/docs/guides/images

[5] "DALL·E 2 | OpenAI." Accessed: Jan. 14, 2025. [Online]. Available: https://openai.com/index/dall-e-2/

[6] "DALL·E 3 | OpenAI." Accessed: Jan. 14, 2025. [Online]. Available: https://openai.com/index/dall-e-3/

[7] "Pricing | OpenAI." Accessed: Jan. 14, 2025. [Online]. Available: https://openai.com/api/pricing/

[8] "Genera imágenes con Imagen 3 | Gemini API | Google AI for Developers." Accessed: Jan. 15, 2025. [Online]. Available: https://ai.google.dev/gemini-api/docs/imagen?hl=es-419

[9] "Imagen 3 - Google DeepMind." Accessed: Jan. 15, 2025. [Online]. Available: https://deepmind.google/technologies/imagen-3/?hl=es-419

[10] "Diffusers." Accessed: Oct. 29, 2024. [Online]. Available: https://huggingface.co/docs/diffusers/index

[11] "Stable Diffusion 3 in KerasHub!" Accessed: Jan. 15, 2025. [Online]. Available: https://keras.io/keras_hub/guides/stable_diffusion_3_in_keras_hub/

[12] "venv — Creation of virtual environments — Python 3.13.0 documentation." Accessed: Oct. 29, 2024. [Online]. Available: https://docs.python.org/3/library/venv.html

[13] "ttkbootstrap - ttkbootstrap." Accessed: Oct. 29, 2024. [Online]. Available: https://ttkbootstrap.readthedocs.io/en/latest/

[14] "Official Documentation And Tutorial | CustomTkinter." Accessed: Jan. 16, 2025. [Online]. Available: https://customtkinter.tomschimansky.com/

[15] "What is Artificial Intelligence? | AI in Business | SAP." Accessed: Jan. 20, 2025. [Online]. Available: https://www.sap.com/products/artificial-intelligence/what-is-artificial-intelligence.html

[16] "¿Qué es la IA generativa? | IBM." Accessed: Jan. 20, 2025. [Online]. Available: https://www.ibm.com/es-es/topics/generative-ai

[17] "¿Qué es una GAN? - Explicación sobre las redes generativas antagónicas - AWS." Accessed: Jan. 19, 2025. [Online]. Available: https://aws.amazon.com/es/what-is/gan/

[18] "¿Qué son los transformadores?: explicación de los transformadores en inteligencia artificial: AWS." Accessed: Jan. 19, 2025. [Online]. Available: https://aws.amazon.com/es/what-is/transformers-in-artificial-intelligence/

[19] "¿Qué son las redes neuronales? | IBM." Accessed: Jan. 20, 2025. [Online]. Available: https://www.ibm.com/es-es/topics/neural-networks

[20] "¿Qué es el deep learning? | IBM." Accessed: Jan. 20, 2025. [Online]. Available: https://www.ibm.com/es-es/topics/deep-learning

[21] "Perceptrón-Comunidad Huawei Enterprise." Accessed: Jan. 19, 2025. [Online]. Available: https://forum.huawei.com/enterprise/intl/es/thread/Perceptr%C3%B3n/667234585961971712?blogId=667234585961971712

[22] "What is the process of diffusion? - BBC Bitesize." Accessed: Jan. 19, 2025. [Online]. Available: https://www.bbc.co.uk/bitesize/articles/znqbcj6#zq6896f

[23] "CUDA - Wikipedia." Accessed: Oct. 29, 2024. [Online]. Available: https://en.wikipedia.org/wiki/CUDA

[24] "stable-diffusion-v1-5/stable-diffusion-v1-5 · Hugging Face." Accessed: Oct. 29, 2024. [Online]. Available: https://huggingface.co/stable-diffusion-v1-5/stable-diffusion-v1-5

[25] "stabilityai/stable-diffusion-2 · Hugging Face." Accessed: Oct. 29, 2024. [Online]. Available: https://huggingface.co/stabilityai/stable-diffusion-2

[26] "stabilityai/stable-diffusion-xl-base-1.0 · Hugging Face." Accessed: Oct. 29, 2024. [Online]. Available: https://huggingface.co/stabilityai/stable-diffusion-xl-base-1.0

[27] "stabilityai/sdxl-turbo · Hugging Face." Accessed: Oct. 29, 2024. [Online]. Available: https://huggingface.co/stabilityai/sdxl-turbo

[28] "Text-to-image." Accessed: Oct. 29, 2024. [Online]. Available: https://huggingface.co/docs/diffusers/main/en/using-diffusers/conditional_image_generation#configure-pipeline-parameters

[29] "PEP 8 – Style Guide for Python Code | peps.python.org." Accessed: Oct. 29, 2024. [Online]. Available: https://peps.python.org/pep-0008/

[30] "Pylance - Visual Studio Marketplace." Accessed: Oct. 29, 2024. [Online]. Available: https://marketplace.visualstudio.com/items?itemName=ms-python.vscode-pylance

[31] "autopep8 · PyPI." Accessed: Oct. 29, 2024. [Online]. Available: https://pypi.org/project/autopep8/

[32] "Python Release Python 3.12.0 | Python.org." Accessed: Oct. 28, 2024. [Online]. Available: https://www.python.org/downloads/release/python-3120/

[33] "PyTorch." Accessed: Oct. 29, 2024. [Online]. Available: https://pytorch.org/

[34] "tkinter — Python interface to Tcl/Tk — documentación de Python - 3.13.0." Accessed: Oct. 29, 2024. [Online]. Available: https://docs.python.org/es/3/library/tkinter.html

[35] "tkinter.ttk — Tk themed widgets — documentación de Python - 3.13.0." Accessed: Oct. 29, 2024. [Online]. Available: https://docs.python.org/es/3/library/tkinter.ttk.html

[36] "Visual Studio Code - Code Editing. Redefined." Accessed: Oct. 29, 2024. [Online]. Available: https://code.visualstudio.com/

[37] "Git." Accessed: Oct. 29, 2024. [Online]. Available: https://git-scm.com/

[38] "Google Colab." Accessed: Oct. 29, 2024. [Online]. Available: https://research.google.com/colaboratory/intl/es/faq.html

[39] "Microsoft Project para la web | Administrar proyectos online." Accessed: Oct. 29, 2024. [Online]. Available: https://www.microsoft.com/es-es/microsoft-365/planner/microsoft-project

[40] "Software de hojas de cálculo Microsoft Excel | Microsoft 365." Accessed: Oct. 29, 2024. [Online]. Available: https://www.microsoft.com/es-es/microsoft-365/excel?msockid=3827f8fde3806ed93d1beddbe2d06f15

[41] "threading — Thread-based parallelism — Python 3.13.0 documentation." Accessed: Oct. 29, 2024. [Online]. Available: https://docs.python.org/3/library/threading.html

[42] "Template method pattern - Wikipedia." Accessed: Oct. 29, 2024. [Online]. Available: https://en.wikipedia.org/wiki/Template_method_pattern

[43] "PyInstaller Manual — PyInstaller 6.11.1 documentation." Accessed: Nov. 14, 2024. [Online]. Available: https://pyinstaller.org/en/stable/

[44] "cx-Freeze · PyPI." Accessed: Nov. 14, 2024. [Online]. Available: https://pypi.org/project/cx-Freeze/

[45] "auto-py-to-exe · PyPI." Accessed: Nov. 14, 2024. [Online]. Available: https://pypi.org/project/auto-py-to-exe/

[46] "pydoc — Documentation generator and online help system — Python 3.13.0 documentation." Accessed: Nov. 10, 2024. [Online]. Available: https://docs.python.org/3/library/pydoc.html

[47] "Python Release Python 3.12.0 | Python.org." Accessed: Jan. 20, 2025. [Online]. Available: https://www.python.org/downloads/release/python-3120/

[48] "How to add Python to Windows PATH? - GeeksforGeeks." Accessed: Jan. 20, 2025. [Online]. Available: https://www.geeksforgeeks.org/how-to-add-python-to-windows-path/

[49] "¿Qué GPUs soportan CUDA? – NVIDIA." Accessed: Jan. 21, 2025. [Online]. Available: https://support.nvidia.eu/hc/es/articles/5123694633314--Qu%C3%A9-GPUs-soportan-CUDA

[50] "The Model Hub." Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/docs/hub/models-the-hub

[51] "Load community pipelines and components." Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/docs/diffusers/using-diffusers/custom_pipeline_overview

[52] "Schedulers in AI Image Generation | by Millun Atluri | Invoke | Medium." Accessed: Jan. 22, 2025. [Online]. Available: https://medium.com/invokeai/schedulers-in-ai-image-generation-2ca6d7458f17

[53] “Schedulers.” Accessed: Jan. 22, 2025. [Online]. Available: https://huggingface.co/docs/diffusers/api/schedulers/overview

[54] “AWS | Cloud Computing - Servicios de informática en la nube.” Accessed: Jan. 22, 2025. [Online]. Available: https://aws.amazon.com/es/

[55] “Inicio - Microsoft Azure.” Accessed: Jan. 22, 2025. [Online]. Available: https://portal.azure.com/#home

Capítulo 13. Apéndices

13.1 Glosario y Diccionario de Datos

- AI: Acrónimo en ingles referido a "*Artificial Intelligence"* traducido al español como "Inteligencia Artificial"
- API: Acrónimo en inglés referido a "Application Programming Interface" traducido al español como Interfaz de programación de aplicaciones, lo cual se trata de un mecanismo que permite a dos componentes *software* comunicarse mediante una serie de definiciones y protocolos.
- Backend: *Software* alojado y ejecutado en el servidor encargado de gestionar la lógica y funcionamiento de una aplicación web.
- Cloud: Traducido al español como "Nube", se refiere al conjunto de servidores pertenecientes a una red distribuida por todo el mundo, de forma que funcionan como un único ecosistema a través del cual, es posible almacenar y acceder a diferentes conjuntos de datos.
- Controlador: Componente *software* que dentro del contexto del patrón Modelo Vista Controlador se refiere al encargado actuar como intermediario de comunicación entre otros componentes *software*.
- Dataset: Conjunto de datos estructurados almacenados en un sistema de información.
- Difusión estable: Modelo de inteligencia artificial generativa que tiene como entrada una descripción de texto y como salida una o varias imágenes correspondientes con la descripción de entrada.
- Endpoint: Ubicación en la que se encuentra un recurso en un servidor, la cual suele seguir el formato de una *URL*.
- Entorno virtual: Área de trabajo que actúa de forma independiente y permite mantener configuraciones y dependencias de un sistema *software* de forma aislada a la máquina física en la que se encuentra.
- IA: Acrónimo para "Inteligencia Artificial".
- Inferencia: Es la capacidad de un modelo de inteligencia Artificial para obtener predicciones o resultados a partir de datos de entrada no utilizados previamente en el proceso de entrenamiento.
- Interfaz gráfica: Componente *software* encargado de representar información y elementos de forma visual a los usuarios además de permitir a estos realizar acciones e interactuar con el *software* a través de ella.
- Modelo de generación: *Software* que utiliza una serie de algoritmos y métodos estadísticos con la capacidad de crear nuevos contenidos no existentes previamente.
- Pipeline: Conjunto de procesos *software* que se encuentran conectados en serie.
- Prompt: Conjunto de instrucciones en formato de texto utilizados para obtener un resultado buscado en el proceso de interacción con un modelo de inteligencia artificial generativa.
- Python: Lenguaje de programación de alto nivel y multipropósito ampliamente utilizado.

- QA: Acrónimo para "*Quality Assurance*" traducido al español como Aseguramiento de la calidad, referido en este contexto a asegurar la calidad del *software*.
- Scheduler: Componente clave de los modelos de difusión estable, se trata de un algoritmo encargado del proceso de la eliminación del ruido en las imágenes en este tipo de modelos.
- Scroll: Acción de desplazamiento normalmente de forma vertical en una aplicación con interfaz gráfica de usuario.
- Widget: Componente de una interfaz gráfica de usuario que se considera como una pequeña aplicación que facilita la realización de alguna funcionalidad determinada.

13.2 Contenido Entregado en el Archivo adjunto

En este capítulo se presenta el contenido presente en el archivo adjunto entregado junto a la presente documentación del proyecto.

13.2.1 Contenidos

En esta sección se describe el contenido del archivo adjunto entregado junto a la documentación del proyecto, se explicará la estructura concreta de directorios dentro de este archivo y el contenido de cada uno de estos directorios.

13.2.1.1 Estructura general de directorios del archivo adjunto

La siguiente figura explica los diferentes directorios y contenidos dentro del archivo adjunto.

Directorio / Archivo	Contenido
./ImageGenTool.7z	El archivo comprimido que contiene el código fuente de la aplicación de escritorio, además de, los archivos de configuración de *git*, el archivo *requirements.txt* con las dependencias del proyecto, y la carpeta *venvgentool* con el contenido necesario para arrancar el entorno virtual.
./doc	Contiene toda la documentación relativa al proyecto.
./doc/planificacionInicial	Contiene los archivos relacionados con la planificación inicial del proyecto y el presupuesto inicial, por lo que contiene el archivo de *MS project planificacionInicial.mpp* y el archivo *Excel presupuestoInicial.xlsx*
./doc/planificacionFinal	Contiene los archivos relacionados con la planificación final del proyecto y el presupuesto final, por lo que contiene el archivo de *MS project* planificacionFinal.mpp y el archivo *Excel* presupuestoFinal.xlsx
./doc/docuPy	Directorio que contiene la documentación generada del código fuente.
./README.txt	Contiene los pasos necesarios para instalar y arrancar la aplicación

Figura 13.1 Contenidos del archivo adjunto

13.2.1.2 Estructura de directorios de "Desarrollo"

La siguiente figura describe la estructura de directorios del proyecto *software* desarrollado.

Directorio / Archivo	Contenido
./ Directorio raíz de "desarrollo"	
./.gitignore	Contiene la configuración de *git* que indica los archivos que *git* debe ignorar en el control de versiones.
./requirements.txt	Contiene las dependencias requeridas por la aplicación.
./tool	Contiene todo el código fuente de la aplicación. Contiene el archivo *main.py* encargado de arrancar la aplicación.
./tool/gui	Contiene los archivos de código fuente *gui.py* y *controller.py*, relacionados con la interfaz de usuario y la lógica de negocio de la herramienta.
./tool/imageGeneration	Contiene el archivo generative_*model.py*, que contiene el código fuente relacionado con la implementación de los modelos generativos y código relacionado con la gestión y ejecución de estos modelos.
./tool/utils	Contiene los archivos de código fuente *file_util.py* e *image_util.py* encargados de realizar operaciones auxiliares y de utilidades de apoyo a la lógica principal.
./tool/tests	Todos los ficheros *Python* relacionados con las pruebas unitarias automatizadas.

Figura 13.2 Estructura de directorios de "desarrollo"

13.2.2 Código Ejecutable e Instalación

Para instalar y ejecutar la aplicación, empezando por la instalación, primeramente, es necesario crear el entorno virtual con las dependencias necesarias, para ello es necesario seguir los pasos descritos en 9.1.

Una vez activo el entorno es posible ejecutar la aplicación ejecutando con *Python* el archivo *main.py* que se encuentra en el directorio *./tool*, si es necesaria más información para ejecutar el proyecto es posible seguir los pasos descritos en 9.2.

13.3 Código Fuente

El código fuente de la herramienta se encuentra en el directorio raíz de los contenidos adjuntos del proyecto en un archivo comprimido, más concretamente en *./ImageGenTool.7z.*