



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**título del TFG
Documentación Técnica**



Presentado por nombre alumno
en Universidad de Burgos — 2 de enero
de 2024

Tutor: nombre tutor

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	iv
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Estudio de viabilidad	1
Apéndice B Especificación de Requisitos	5
B.1. Introducción	5
B.2. Objetivos generales	5
B.3. Catálogo de requisitos	5
B.4. Casos de uso	7
Apéndice C Especificación de diseño	15
C.1. Introducción	15
C.2. Diseño de datos	15
C.3. Diseño procedimental	16
C.4. Diseño arquitectónico	17
Apéndice D Documentación técnica de programación	23
D.1. Introducción	23
D.2. Estructura de directorios	23
D.3. Manual del programador	24

D.4. Compilación, instalación y ejecución del proyecto	25
D.5. Pruebas del sistema	25
Apéndice E Documentación de usuario	27
E.1. Introducción	27
E.2. Requisitos de usuario	27
E.3. Instalación	27
E.4. Manual del usuario	27
Apéndice F Anexo de sostenibilización curricular	29
F.1. Introducción	29
Bibliografía	31

Índice de figuras

B.1. Diagrama de casos de uso	7
C.1. Arquitectura de la base de datos	15
C.2. Login	16
C.3. Administración	17
C.4. Gestor	17
C.5. Arquitectura de la aplicación	18
C.6. Pantalla de login.	19
C.7. Pantalla usuario gestor.	20
C.8. Pantalla de administración.	21

Índice de tablas

B.1. CU-1 Registro de usuarios.	8
B.2. CU-2 Login de usuarios.	9
B.3. CU-3 Información de la aplicación.	10
B.4. CU-04 Cargar reseñas de un archivo.	11
B.5. CU-5 Puntuar los textos.	12
B.6. CU-6 Almacenar resultados.	13
B.7. CU-07 Información estadística.	13
B.8. CU-08 Configurar el idioma de la aplicación.	14

Apéndice A

Plan de Proyecto Software

A.1. Introducción

A.2. Planificación temporal

A.3. Estudio de viabilidad

La viabilidad de un proyecto de software construido con Python, Flask, Vue, SQLAlchemy, flask_resx, OpenAI y Langchain Tecnologías Utilizadas: Python con Flask, Vue, SQLAlchemy, flask_resx: Estas son tecnologías ampliamente utilizadas y tienen una sólida base de usuarios y soporte comunitario. Son conocidas por su eficiencia y flexibilidad en el desarrollo web. OpenAI: El uso de OpenAI, en particular GPT-3, puede agregar un componente de inteligencia artificial avanzada a tu proyecto. Sin embargo, debes tener en cuenta los costos asociados y las políticas de uso de OpenAI. Viabilidad Técnica: Las tecnologías mencionadas son conocidas por ser sólidas y eficientes, pero la viabilidad técnica también dependerá de la complejidad y los requisitos específicos de tu proyecto. Asegúrate de que estas herramientas sean apropiadas para tus necesidades. Complejidad del Proyecto: Evalúa la complejidad del proyecto en términos de características, funcionalidades y requisitos técnicos. ¿Es realista y alcanzable con las herramientas y habilidades disponibles? Costos Asociados: Considera los costos asociados con el desarrollo, implementación y mantenimiento del proyecto. Esto incluye posibles tarifas de servicios externos como OpenAI, así como cualquier costo adicional asociado con la utilización de herramientas específicas. Demanda del Mercado: Investiga la demanda del mercado para tu tipo de proyecto.

¿Hay una necesidad real que tu proyecto pueda abordar? ¿Cómo se compara con las soluciones existentes? Competencia: Evalúa la competencia en el mercado. ¿Existen proyectos similares? ¿Cómo puedes diferenciarte y ofrecer un valor único? Modelo de Negocio: Define claramente tu modelo de negocio y la estrategia de monetización. ¿Planeas ofrecer el software de forma gratuita, freemium, mediante suscripciones u otro modelo? Legalidad y Cumplimiento: Asegúrate de cumplir con todas las leyes y regulaciones relevantes, especialmente en términos de protección de datos, privacidad y propiedad intelectual. Equipo y Habilidades: Evalúa si tienes el equipo con las habilidades necesarias para implementar y mantener el proyecto. Si es necesario, considera la posibilidad de contratar o asociarte con expertos en áreas específicas. Escalabilidad: Diseña tu proyecto pensando en la escalabilidad. ¿Puede crecer de manera eficiente y mantener un rendimiento sólido a medida que aumenta la base de usuarios? En resumen, la viabilidad del proyecto dependerá de cómo abordes estos aspectos y de la alineación de tu visión con las necesidades del mercado. Realizar un análisis detallado y planificar cuidadosamente cada etapa del proyecto contribuirá significativamente a su éxito.

Viabilidad económica

La viabilidad económica de un proyecto de software construido con las tecnologías mencionadas dependerá de varios factores, incluyendo el alcance y los objetivos del proyecto, el mercado al que se dirige y la estrategia de monetización. Aquí hay algunas consideraciones generales: Costos de Desarrollo: Las tecnologías mencionadas, como Python con Flask, Vue, SQLAlchemy, flask_resx, OpenAI, y Langchain, son en su mayoría de código abierto, lo que significa que son accesibles sin costos iniciales significativos. Sin embargo, ten en cuenta los costos asociados con el desarrollo y la implementación, así como posibles tarifas por el uso de servicios externos (por ejemplo, OpenAI). Monetización: Define cómo planeas monetizar tu proyecto. ¿Es un software de pago, freemium, publicidad, suscripciones u otro modelo de negocio? La elección dependerá de la propuesta de valor de tu producto y las expectativas del mercado. Demanda del Mercado: Investiga la demanda del mercado para tu solución. ¿Hay una necesidad real para tu producto? ¿Cómo se compara con otras soluciones existentes? Evalúa la competencia y busca áreas donde tu proyecto pueda destacar. Modelo de Precios para OpenAI: Si planeas utilizar OpenAI, ten en cuenta su modelo de precios. Evalúa cómo impactará en los costos a medida que tu proyecto crezca y obtenga más usuarios. Desarrollo Incremental: Considera la posibilidad de implementar el proyecto de manera incremental, lanzando versiones tempranas.

nas y recopilando comentarios para realizar mejoras. Esto puede ayudar a reducir riesgos y costos iniciales. Soporte y Mantenimiento: Asegúrate de considerar los costos asociados con el soporte y el mantenimiento continuo del software. Esto incluye actualizaciones, corrección de errores y atención al cliente. Escalabilidad: Diseña tu proyecto con la escalabilidad en mente. Evalúa cómo se comportará en términos de rendimiento y costos a medida que crezca la base de usuarios. Licencias y Legalidad: Asegúrate de cumplir con las licencias de todas las tecnologías utilizadas y verifica la legalidad de su uso en tu proyecto. En resumen, la viabilidad económica dependerá de la capacidad del proyecto para abordar una necesidad del mercado, la efectividad de la estrategia de monetización, y la gestión eficiente de costos y riesgos. Realizar un análisis de mercado, entender las expectativas de los usuarios y planificar cuidadosamente la implementación serán pasos esenciales para evaluar y mejorar la viabilidad económica de tu proyecto.

Viabilidad legal

Analizar la viabilidad legal de un proyecto de software implica tener en cuenta varios aspectos, como la licencia de las tecnologías utilizadas, posibles problemas de propiedad intelectual, cumplimiento de leyes de privacidad y protección de datos, entre otros. Aquí hay algunas consideraciones generales: Licencias de Software: Python con Flask, Vue, SQLAlchemy, flask_resx: Estas tecnologías son conocidas por tener licencias de software de código abierto. Sin embargo, es esencial revisar las condiciones específicas de cada una para asegurarse de que son compatibles con el uso que se planea para el proyecto y para evitar posibles conflictos. OpenAI: OpenAI proporciona API y modelos como GPT-3. Es importante revisar y cumplir con los términos de servicio y las licencias asociadas. OpenAI ha actualizado sus políticas a lo largo del tiempo, por lo que es crucial estar al tanto de las condiciones actuales. Protección de Datos y Privacidad: Asegurarse de cumplir con las leyes de privacidad y protección de datos aplicables, como el Reglamento General de Protección de Datos (GDPR) en la Unión Europea o leyes similares en otras jurisdicciones. Asegúrate de manejar adecuadamente la información del usuario y obtener el consentimiento cuando sea necesario. Propiedad Intelectual: Si estás utilizando tecnologías de código abierto, asegúrate de cumplir con las licencias y de entender las implicaciones de la propiedad intelectual. También, verifica que el proyecto que estás construyendo no infrinja patentes o derechos de autor de terceros. Contratos y Acuerdos: Si colaboras con otras personas en el proyecto, considera la importancia de acuerdos claros y contratos que definan la propiedad intelectual, responsabilidades y cualquier aspecto legal relevante. Marcas Registradas: Si planeas

comercializar el software bajo un nombre específico, verifica la disponibilidad de la marca y considera registrarla para evitar posibles problemas legales en el futuro.

Apéndice B

Especificación de Requisitos

B.1. Introducción

B.2. Objetivos generales

Crear una aplicación que permita realizar análisis de sentimiento a textos extraídos de reseñas de Amazon. Almacenar dicha información en una base de datos para que sea fácilmente accesible. Mostrar dicha información con gráficos y tablas para poder analizarlos.

Crear un manual de usuario.

B.3. Catálogo de requisitos

Requisitos funcionales

- **RF1 - Registro de usuarios:** La aplicación debe permitir dar de alta usuarios y asignarles un rol.
- **RF2 - Login de usuarios:** La aplicación debe ser capaz de dadas las credenciales de un usuario permitirle acceder a las funcionalidades.
- **RF3 - Información de la aplicación:** El usuario debe poder obtener información sobre el funcionamiento de la aplicación.
- **RF4 - Analizar textos:** La aplicación debe ser capaz de analizar textos.
 - **RF4.1 - Cargar reseñas de un archivo:** La aplicación debe poder cargar las reseñas de un archivo de texto

- **RF4.2 - Puntuar los textos:** La aplicación debe realizar el análisis del texto obtenido y devolver un valor entero entre 1 y 5 dependiendo del sentimiento del mismo, siendo 1 un sentimiento negativo y 5 un sentimiento positivo.
- **RF4.3 - Almacenar resultados:** La aplicación debe almacenar las puntuaciones junto con el texto analizado, fecha en la que se escribió y un identificador.
- **RF5 - Buscar resultados por identificador:** El usuario debe poder buscar los resultados de un identificador con unas fechas seleccionadas.
 - **RF5.1 - Buscar por producto:** El usuario debe poder introducir el nombre de un producto y mostrar las reseñas del mismo.
 - **RF5.2 - Buscar por usuario:** El usuario debe poder buscar un usuario y mostrar la información de las reseñas que ha creado.
 - **RF5.3 - Buscar por texto:** El usuario debe poder buscar una palabra para ver los resultados del análisis de los textos en los que está esa palabra.
- **RF6 - Matriz de confusión:** Debe mostrar la matriz de confusión para los datos seleccionados, comparando los resultados obtenidos y la calificación dada por el usuario.
- **RF7 - Configurar idioma:** El usuario debe poder cambiar el idioma de la aplicación.

Requisitos no funcionales

- **RNF1 - Usabilidad:** La aplicación debe ser amigable para que la experiencia del usuario sea lo más positiva posible. Además debe poder adaptarse a diferentes formatos de pantalla.
- **RNF2 - Eficiencia:** El almacenamiento de resultados en la base de datos y la respuesta de la aplicación al navegar entre ventanas, sobre todo, al mostrar los gráficos debe ser lo más rápido posible.
- **RNF3 - Disponibilidad:** La aplicación debe estar disponible durante el mayor tiempo posible y en la mayoría de lugares.
- **RNF4 - Escalabilidad:** La aplicación debe estar preparada para soportar nuevos desarrollos que generen mayor cantidad de trabajo.

- **RNF5 - Confiabilidad:** La aplicación cumplirá la función para la que se ha creado.
- **RNF6 - Mantenibilidad:** La aplicación debe permitir cambios y correcciones de forma eficaz.
- **RNF7 - Internacionalización:** La aplicación debe soportar varios idiomas.
- **RNF8 - Seguridad:** La aplicación debe soportar usuarios con diferentes roles que tendrán acceso a diferentes funcionalidades.

B.4. Casos de uso

Diagrama de casos de uso

Como resumen de los casos de uso anteriores obtenemos el siguiente diagrama de casos de uso:

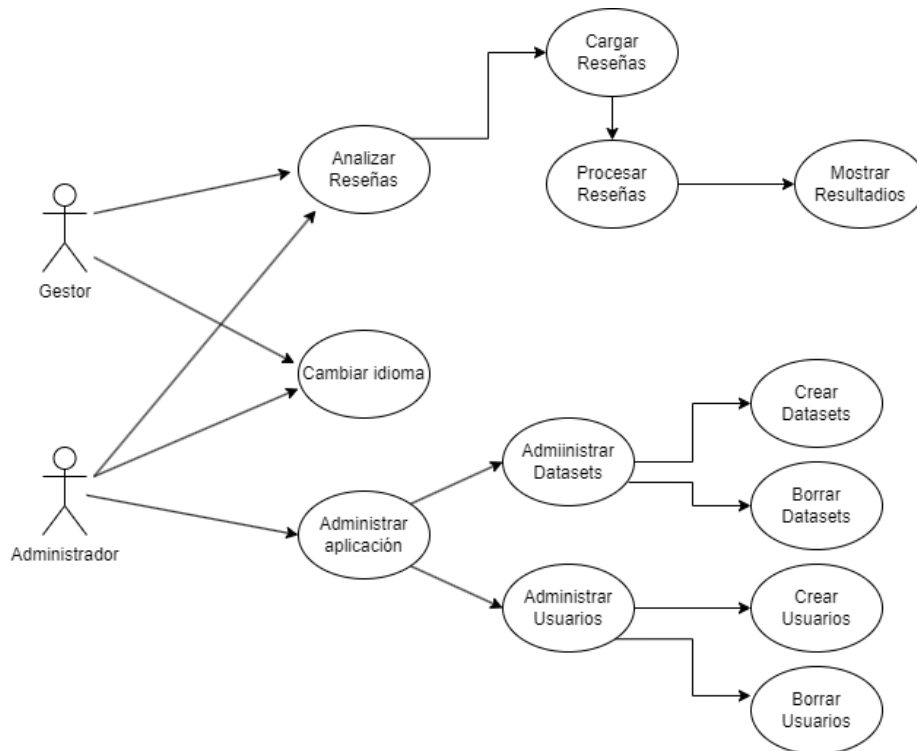


Figura B.1: Diagrama de casos de uso

CU-1	Registro de usuarios
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF-1
Descripción	Un usuario con el rol de administrador podrá crear cuentas de usuario y asignarles un rol.
Precondición	El usuario tiene que tener el rol de administrador
Acciones	<ol style="list-style-type: none"> 1. El usuario administrador accede a la interfaz de la aplicación. 2. Accede a la opción Crear usuarios 3. Introduce los datos del usuarios 4. Confirma los cambios.
Postcondición	La cuenta de usuario no debe existir con antelación
Excepciones	<ol style="list-style-type: none"> 1. El usuario ya existe 2. Complejidad de la contraseña no válida 3. Datos incorrectos
Importancia	Alta

Tabla B.1: CU-1 Registro de usuarios.

CU-2	Login de usuarios
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF-2
Descripción	El usuario accede a la aplicación tendiendo acceso a la parte que tenga acceso según su rol.
Precondición	Usuario registrado
Acciones	<ol style="list-style-type: none">1. Accede a la pantalla de login2. Introduce nombre y contraseña3. Pulsa en Aceptar
Postcondición	La contraseña debe coincidir
Excepciones	<ol style="list-style-type: none">1. El usuario no existe2. La contraseña no es válida
Importancia	Alta

Tabla B.2: CU-2 Login de usuarios.

CU-3	Información de la aplicación.
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF-3
Descripción	Permite al usuario obtener información sobre la configuración de la aplicación.
Precondición	Usuario logado.
Acciones	<ol style="list-style-type: none">1. El usuario accede a la opción información.
Postcondición	
Excepciones	
Importancia	Baja

Tabla B.3: CU-3 Información de la aplicación.

CU-4	Cargar reseñas de un archivo
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF4.1
Descripción	Un usuario administrador puede cargar reseñas desde un archivo
Precondición	Usuario logado
Acciones	<ol style="list-style-type: none">1. Acceder a la aplicación.2. Seleccionar archivo.3. Seleccionar opciones.4. Pulsar en cargar.
Postcondición	El fichero tiene el formato correcto.
Excepciones	<p>El fichero no tiene el formato correcto.</p> <p>El fichero no se puede abrir.</p> <p>El fichero no se ha podido cargar.</p>
Importancia	Alta

Tabla B.4: CU-04 Cargar reseñas de un archivo.

CU-5	Puntuar los textos
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF4.2
Descripción	La aplicación debe realizar el análisis del texto obtenido y devolver un valor entero entre 1 y 5 dependiendo del sentimiento del mismo, siendo 1 un sentimiento negativo y 5 un sentimiento positivo.
Precondición	Los textos están cargados
Acciones	<ol style="list-style-type: none"> 1. Se accede a la aplicación. 2. Fichero de reseñas se ha cargado. 3. Se selecciona la opción procesar.
Postcondición	Se recibe una respuesta del servicio de análisis.
Excepciones	No hay respuesta
Importancia	Alta

Tabla B.5: CU-5 Puntuar los textos.

CU-6	Almacenar resultados
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF4.3
Descripción	La aplicación debe almacenar las puntuaciones junto con el texto analizado, fecha en la que se escribió y un identificador.
Precondición	Se han procesado las reseñas
Acciones	<ol style="list-style-type: none"> 1. Se procesan los elementos 2. El resultado se guarda
Postcondición	La base de datos tiene que estar disponible.
Excepciones	La base de datos no está disponible
Importancia	Alta

Tabla B.6: CU-6 Almacenar resultados.

CU-07	Información estadística.
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF6
Descripción	Debe mostrar la información estadística para los datos analizados.
Precondición	Análisis de sentimiento ejecutado.
Acciones	<ol style="list-style-type: none"> 1. Seleccionar la opción de visualizar resultados.
Postcondición	
Excepciones	Datos no generados
Importancia	Media

Tabla B.7: CU-07 Información estadística.

CU-08	Configurar idioma de la aplicación.
Versión	1.0
Autor	Teodoro Ricardo García Sánchez
Requisitos asociados	RF7
Descripción	El usuario debe poder cambiar el idioma de la aplicación.
Precondición	El usuario debe de estar logado
Acciones	<ol style="list-style-type: none"> 1. Seleccionar la opción configuración 2. Elegir el idioma deseado, según los idiomas disponibles
Postcondición	
Excepciones	
Importancia	Media

Tabla B.8: CU-08 Configurar el idioma de la aplicación.

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de datos

Para almacenar la información necesaria para la aplicación he diseñado una base de datos que sirva para facilitar el procesamiento de las reseñas y su posterior análisis por parte de los usuarios de la aplicación:

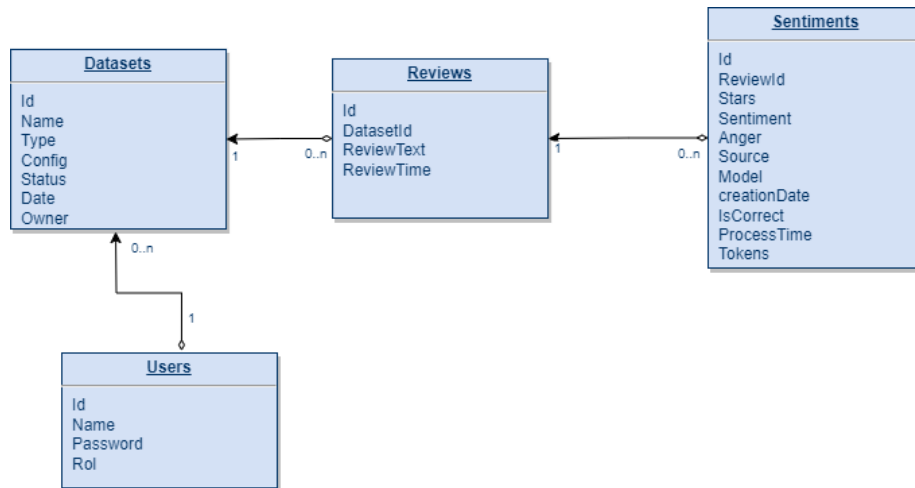


Figura C.1: Arquitectura de la base de datos

Como vemos almacenamos la información relativa a los conjuntos de datos en la tabla "datasets". En esta tabla almacenamos la información

necesaria para cargarlo como es el tipo (type) y el payload. Según el tipo de conjunto de datos el método para cargarlo será diferente. Una vez iniciado el proceso de carga, la información relativa a las reseñas se almacenan en la tabla Review", "Productsz ReviewUsers".

Por último, se almacena en la tabla "sentiments" la información relativa al análisis de sentimiento. Se incluye la información obtenida del modelo así como el tiempo que ha tardado y el número de tokens procesados. Este último campo (tokens) se puede utilizar para calcular el coste en el caso de OpenAI. También se indica si la respuesta del modelo se considera errónea por tener un formato incorrecto. En este caso la respuesta del modelo se almacena en el campo "source".

C.3. Diseño procedimental

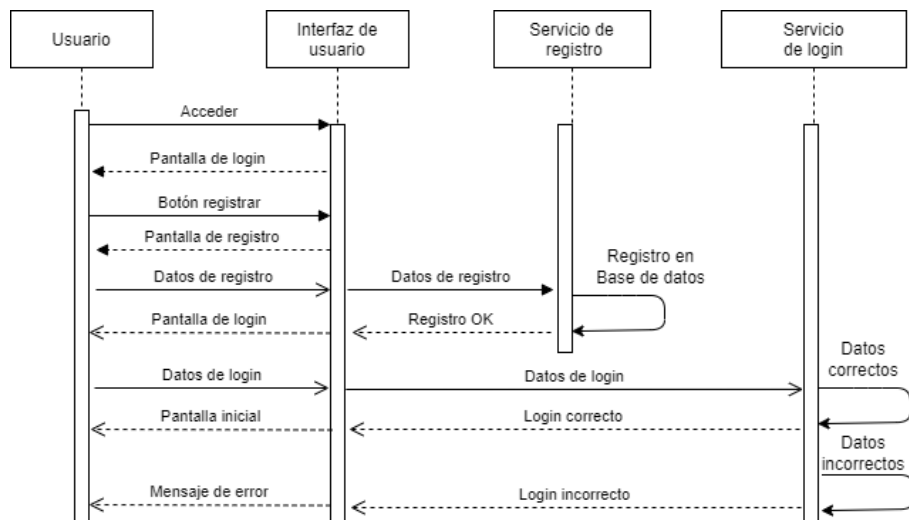


Figura C.2: Login

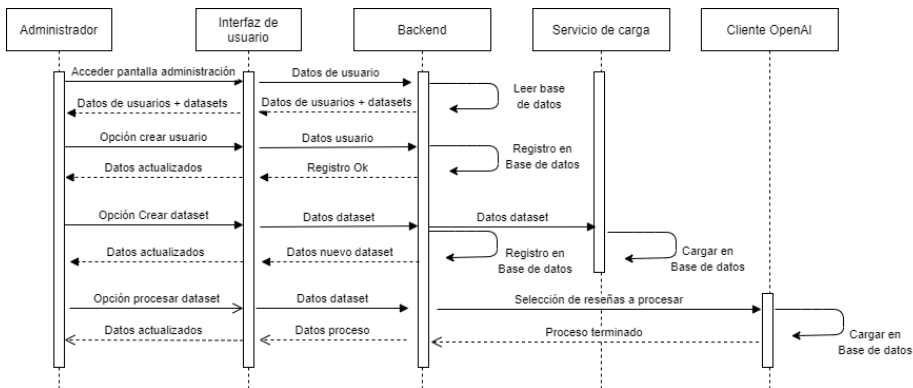


Figura C.3: Administración

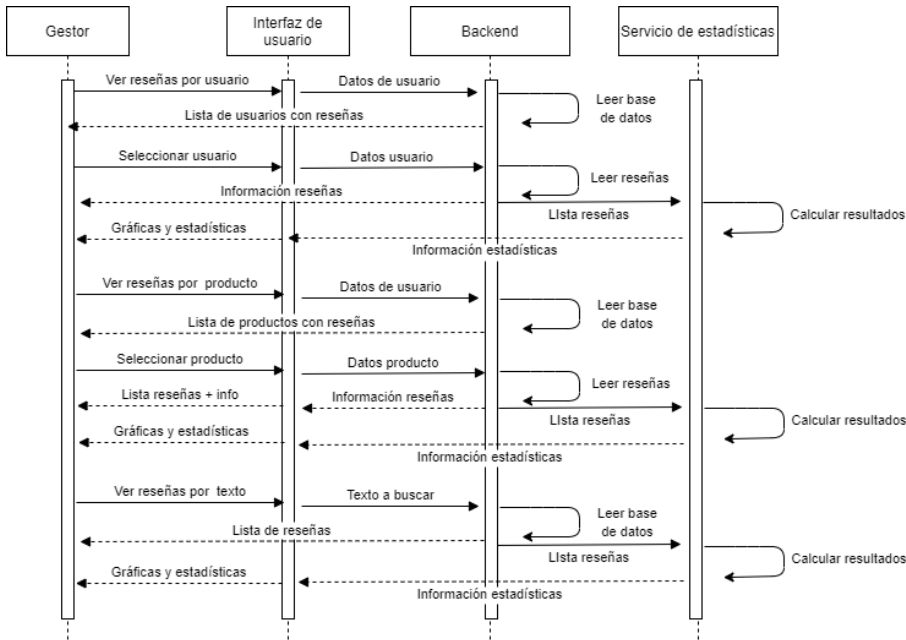


Figura C.4: Gestor

C.4. Diseño arquitectónico

Arquitectura de la aplicación

En el caso de la arquitectura de la aplicación, vemos que está centrada en la base de datos. Desde el módulo de administración se pueden gestionar los usuarios y las reseñas. Desde el módulo de servicios se cargan las reseñas

y se gestionan los diferentes procesos de análisis. A través de un servicio web se expone al usuario los diferentes informes de resultados.

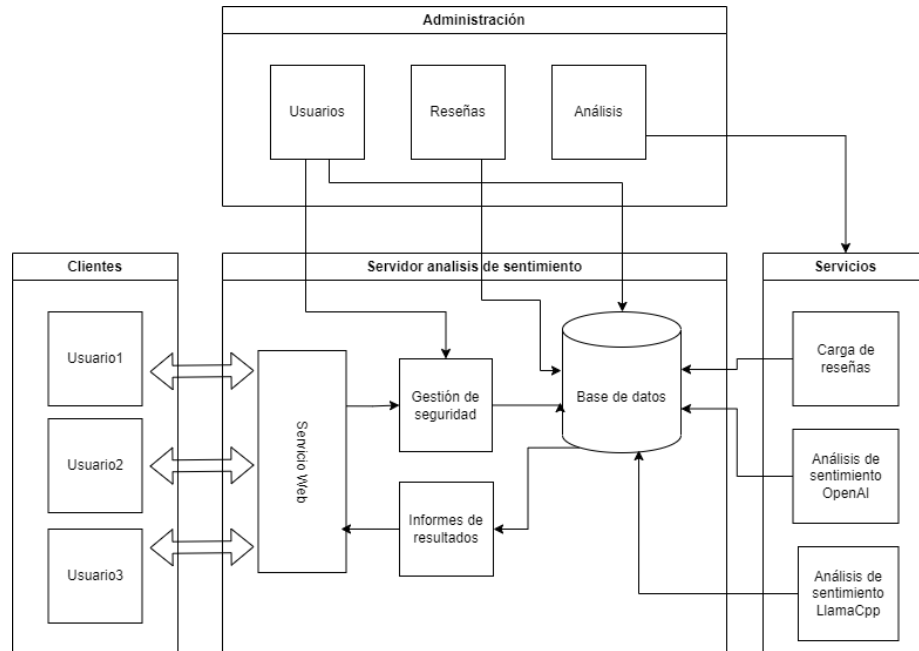
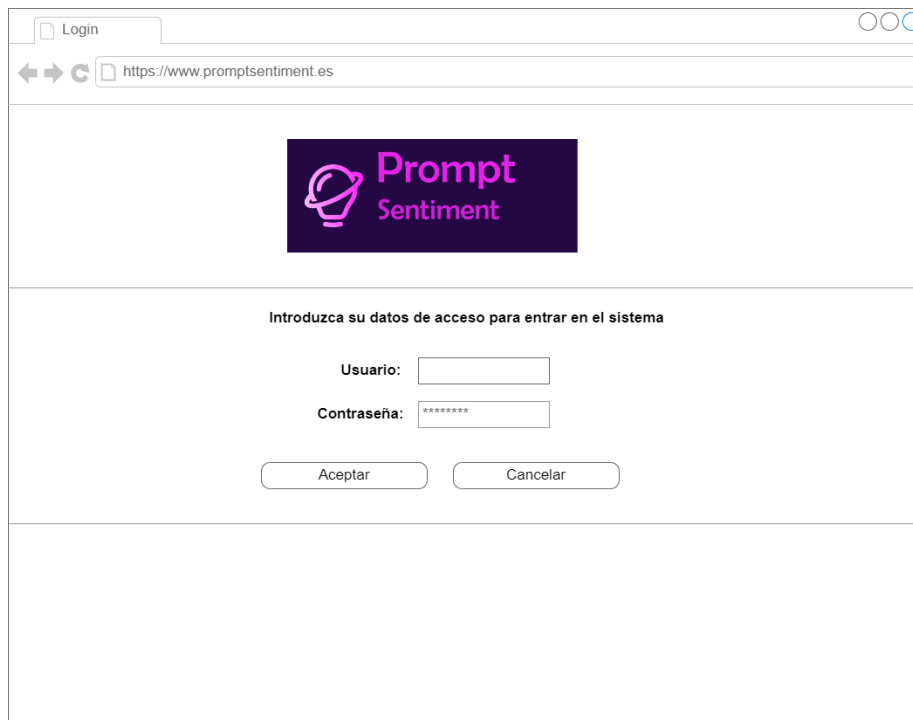


Figura C.5: Arquitectura de la aplicación

Interfaz de usuario

En la pantalla de login debe aparecer el logo de la aplicación junto con la información necesaria para poder acceder a la aplicación. Se puede mostrar también una pequeña introducción sobre el proposito de la misma y un pequeño vínculo para acceder al registro como usuario.



The image shows a web browser window with a single tab titled "Login". The address bar displays "https://www.promptsentiment.es". The main content area features the "Prompt Sentiment" logo, which consists of a purple square containing a white icon of a speech bubble with a signal wave and the text "Prompt Sentiment" in white. Below the logo, the instruction "Introduzca su datos de acceso para entrar en el sistema" is centered. This is followed by two input fields: "Usuario:" and "Contraseña:". The password field is masked with asterisks. At the bottom of the form are two buttons: "Aceptar" and "Cancelar".

Figura C.6: Pantalla de login.

En esta pantalla se mostrará la información relativa a las reseñas y al análisis de sentimiento realizado hasta el momento.

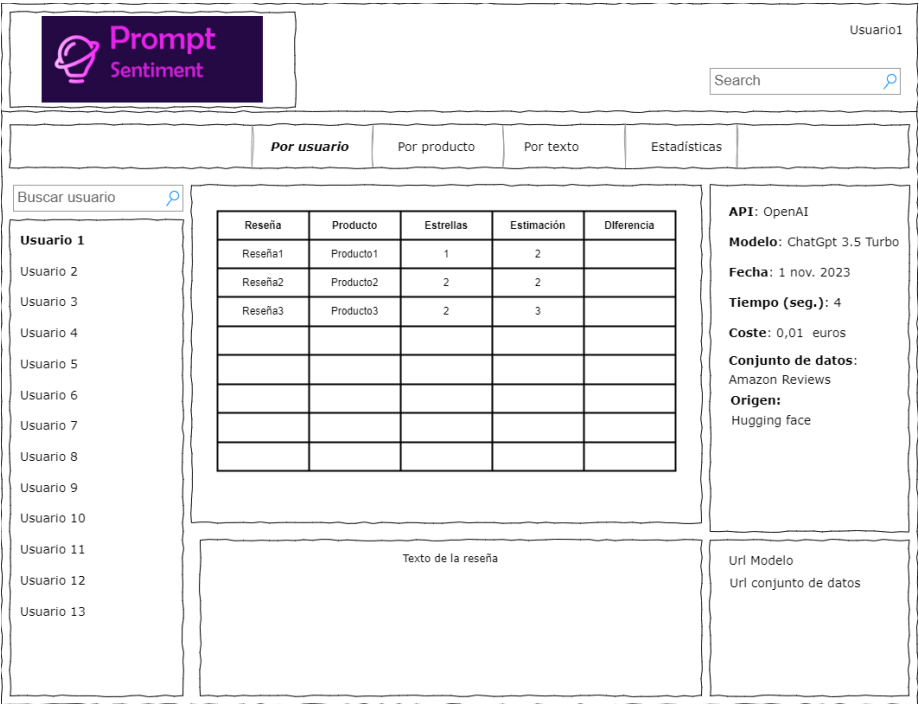


Figura C.7: Pantalla usuario gestor.

En la pantalla de administración se podrá gestionar los usuarios y las reseñas.

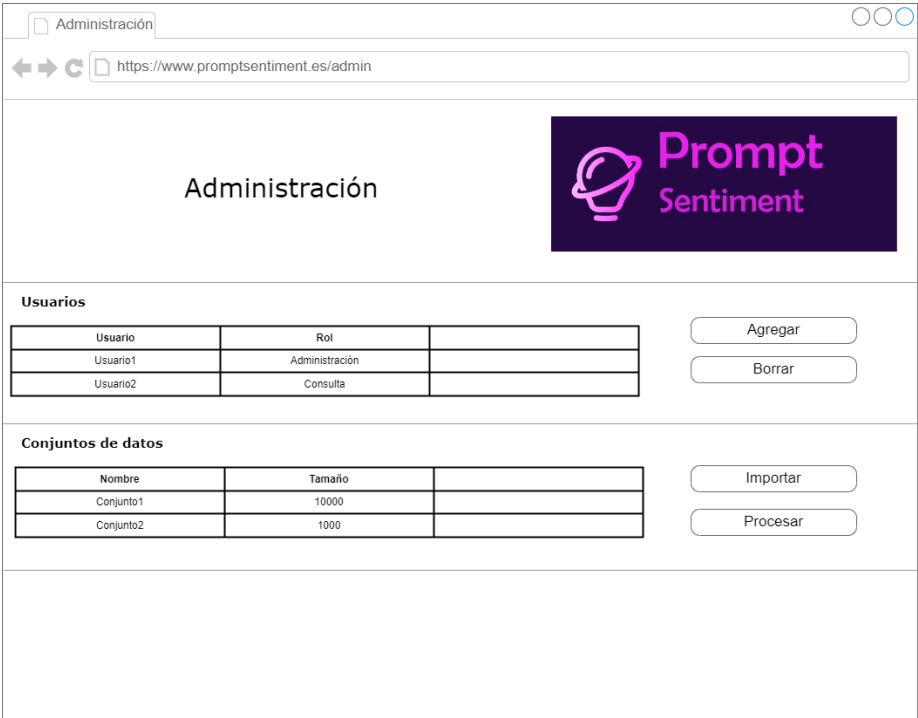


Figura C.8: Pantalla de administración.

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este apéndice se va a explicar cómo está organizado el proyecto, el manual del programador y los requisitos necesarios para poder ejecutar el proyecto.

D.2. Estructura de directorios

En el proyecto hay una estructura de directorios que intenta organizar el trabajo de manera lógica: Por un lado tenemos el código y por otra la documentación. Dentro del código podemos apreciar una carpeta llamada colab con un cuaderno de Jupyter para poder ejecutarlo en Colab. Hay otra carpeta con el código del proyecto es sí. La estructura de directorios quedaría así:

- Código
 - colab: Contiene un cuaderno de Jupyter preparado para ejecutar un ejemplo en Colab.
 - prompt_sentiment: Contiene el código del trabajo.
 - frontend: Contiene el código de la interfaz de usuario. Está hecha con Vue, usando el plugin Vuetify.

- backend: Contiene el código del servicio web de la aplicación. Está hecha en Flask (python) usando Flask-restx y SQLAlchemy para el acceso a las base de datos.
- nginx: En este directorio está la configuración del servidor nginx que se usa en la aplicación como proxy inverso y para gestionar la configuración https.

- Documentación

- Manual de usuario: El manual de uso de la aplicación
- Memoria: Es la memoria del proyecto

D.3. Manual del programador

Instalación local

Para la instalación local del proyecto tenemos realizar las siguientes instalaciones:

Instalación de Python 3 [1]

Para ejecutar nuestro proyecto es necesario instalar Python, para ello podemos descargar la versión aquí: <https://www.python.org/downloads/> Cuando la descarga haya finalizado ejecutamos e instalamos. También se recomienda usar conda para crear un entorno virtual para python. <https://docs.conda.io/projects/conda/en/latest/index.html>

Instalación de node.js

Para la parte de angular es necesario instalar Nodejs, por tanto accedemos a la página oficial, <https://nodejs.org/es/download/>, y descargaremos la versión que necesitemos y llevaremos a cabo su instalación

Instalación de PostgreSQL

Para la instalación de PostgreSQL tenemos que bajarnos la versión adecuada a nuestro sistema operativo e instalarla: <https://www.postgresql.org/download/>

Docker

Para la instalación de docker sólo hay que instalar Docker-Desktop.

Instalación de Docker Desktop

<https://docs.docker.com/engine/install/>

D.4. Compilación, instalación y ejecución del proyecto

Ejecución local

El proyecto tiene 2 partes claramente diferenciadas: frontend y backend. Para la ejecución del backend tenemos que abrir una consola de comandos, movernos al directorio backend y ejecutar: //Instalamos las referencias pip install -r requirements.txt

Creamos la base de datos (sólo en el caso de que no exista)

```
python manage.py crear
python manage.py rellenar (en caso de que esté vacía)
python manage.py run
```

Para el frontend, nos cambiamos la carpeta Frontend y ejecutamos:

```
npm install
npm run serve
```

Ejecución en docker

Para la ejecución en docker sólo hay que ejecutar en el directorio raíz del código:

```
docker-compose up
```

D.5. Pruebas del sistema

Para la ejecución de las pruebas unitarias del backend hay que ejecutar:

```
pytest
```


Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuario
- E.3. Instalación
- E.4. Manual del usuario

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

Este anexo incluirá una reflexión personal del alumnado sobre los aspectos de la sostenibilidad que se abordan en el trabajo. Se pueden incluir tantas subsecciones como sean necesarias con la intención de explicar las competencias de sostenibilidad adquiridas durante el alumnado y aplicadas al Trabajo de Fin de Grado.

Más información en el documento de la CRUE https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sostenibilidad_Crue2012.pdf. Este anexo tendrá una extensión comprendida entre 600 y 800 palabras.

La sostenibilidad de un proyecto de software como el mencionado dependerá de varios factores, y es crucial considerar tanto los aspectos económicos como los sociales y ambientales. Aquí hay algunas reflexiones sobre la sostenibilidad del proyecto:

Económica: Modelo de Negocio Sostenible: Un modelo de negocio sólido y sostenible es esencial. Evalúa la capacidad del proyecto para generar ingresos de manera constante y asegúrate de que el modelo de monetización elegido sea viable a largo plazo. Gestión de Costos: Controlar eficientemente los costos de desarrollo, mantenimiento y operación es clave para la sostenibilidad económica. Considera la posibilidad de utilizar servicios en la nube de manera eficiente y evalúa constantemente la rentabilidad de las herramientas y tecnologías utilizadas. Social: Impacto Positivo: Evalúa cómo el proyecto puede tener un impacto positivo en la sociedad. Por ejemplo,

si el software ayuda a las empresas a comprender y mejorar la satisfacción del cliente, esto podría contribuir a un impacto social positivo. Accesibilidad: Asegúrate de que el software sea accesible para la mayor cantidad de personas posible. Considera aspectos como la usabilidad, la inclusión de personas con discapacidades y la disponibilidad en diferentes idiomas. Ética en la Inteligencia Artificial: Si el proyecto utiliza inteligencia artificial (como OpenAI), es crucial considerar aspectos éticos. Asegúrate de que el uso de IA sea transparente, justo y ético, y evita sesgos o discriminación. Ambiental: Eficiencia Energética: Evalúa la eficiencia energética del software y las tecnologías utilizadas. La sostenibilidad ambiental implica minimizar el consumo de recursos, incluida la energía. Impacto del Desarrollo: Considera el impacto ambiental durante el desarrollo del software. Esto podría incluir prácticas de desarrollo sostenibles, como la reducción de residuos de código o la eficiencia en el uso de recursos de desarrollo. Comunidad y Colaboración: Involucramiento de la Comunidad: Fomenta la participación y retroalimentación de la comunidad. Una comunidad activa puede contribuir al éxito continuo del proyecto y proporcionar ideas valiosas para mejoras sostenibles. Colaboración Abierta: Considera la posibilidad de colaborar con otros proyectos de código abierto y contribuir al ecosistema más amplio. La colaboración puede generar sinergias y mejorar la sostenibilidad a largo plazo. En resumen, para lograr la sostenibilidad de un proyecto de software, es necesario equilibrar aspectos económicos, sociales y ambientales. La transparencia, la ética y la consideración de los impactos a largo plazo son esenciales para construir y mantener un proyecto que beneficie tanto a los usuarios como al entorno en el que se desarrolla.

Bibliografía

- [1] Mark Lutz. *Programming python*. "O'Reilly Media, Inc.", 2001.