UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Diseño de aplicación web para el control de gastos personales

Trabajo Fin de Grado

Grado en Tecnología Digital y Multimedia

AUTOR/A: Ocheda Gimenez, Oscar

Tutor/a: Guerola Navarro, Vicente

CURSO ACADÉMICO: 2023/2024

RESUMEN

Este proyecto surge como resultado de la búsqueda de un trabajo final de carrera vistoso y útil que integre tecnologías aprendidas durante mi formación académica. En este caso, la atención se centra en el uso de herramientas como Python, HTML y posiblemente JavaScript para desarrollar aplicaciones web que permitan a los usuarios administrar eficazmente sus finanzas personales.

La motivación detrás de este proyecto es crear una herramienta práctica y funcional que aborde una necesidad común: el control de gastos. Al proporcionar una plataforma intuitiva y accesible, se busca empoderar a los usuarios para que tomen decisiones financieras informadas y promuevan una mejor salud financiera.

Este proyecto no sólo logra objetivos académicos aplicando conocimientos de programación y desarrollo web, sino que también brinda la oportunidad de destacar creando una aplicación colorida y atractiva para los usuarios. Además, el proyecto se puede implementar de manera eficiente y sólida utilizando tecnologías familiares como Python y HTML.

En resumen, este proyecto representa un trabajo final de carrera que combina la aplicación práctica de las técnicas aprendidas con la creación de herramientas útiles y atractivas de gestión financiera personal

Palabras clave: Control de gastos, Finanzas personales, Aplicación web, Python, HTML

RESUM

Aquest projecte sorgeix com a resultat de la recerca d'un treball final de carrera vistós i útil que integre tecnologies apreses durant la meua formació acadèmica. En aquest cas, l'atenció es centra en l'ús d'eines com Python, HTML i possiblement JavaScript per desenvolupar aplicacions web que permeten als usuaris administrar eficaçment les seues finances personals.

La motivació darrere d'aquest projecte és crear una eina pràctica i funcional que aborde una necessitat comuna: el control de despeses. En proporcionar una plataforma intuïtiva i accessible, es busca capacitar els usuaris perquè prenguen decisions financeres informades i promoguen una millor salut financera.

Aquest projecte no només aconsegueix objectius acadèmics aplicant coneixements de programació i desenvolupament web, sinó que també brinda l'oportunitat de destacar creant una aplicació colorida i atractiva per als usuaris. A més, el projecte es pot implementar de manera eficient i sòlida utilitzant tecnologies familiars com Python i HTML.

En resum, aquest projecte representa un treball final de carrera que combina l'aplicació pràctica de les tècniques apreses amb la creació d'eines útils i atractives de gestió financera personal.

Paraules clau: Control de despeses, Finances personals, Aplicació web, Python, HTML

ABSTRACT

This project arises as a result of the search for a visually appealing and useful final career work that integrates technologies learned during my academic training. In this case, the focus is on the use of tools such as Python, HTML, and possibly JavaScript to develop web applications that allow users to effectively manage their personal finances.

The motivation behind this project is to create a practical and functional tool that addresses a common need: expense control. By providing an intuitive and accessible platform, the aim is to empower users to make informed financial decisions and promote better financial health.

This project not only achieves academic objectives by applying programming and web development knowledge but also provides the opportunity to stand out by creating a colorful and attractive application for users. Additionally, the project can be efficiently and robustly implemented using familiar technologies such as Python and HTML.

In summary, this project represents a final career work that combines the practical application of learned techniques with the creation of useful and attractive tools for personal financial management.

Keywords: Expense Management, Personal Finance, Web Application, Python, HTML

RESUMEN EJECUTIVO

La memoria del TFG del Grado en Tecnología Digital y Multimedia debe desarrollar en el texto los siguientes conceptos, debidamente justificados y discutidos, centrados en el ámbito de las tecnologías digitales y multimedia.

CONCEPT (ABET)	CONCEPTO (traducción)	¿Cumple? (S/N)	¿Dónde? (páginas)
1. IDENTIFY:	1. IDENTIFICAR:		
1.1. Problem statement and opportunity	1.1. Planteamiento del problema y oportunidad	S	9
1.2. Constraints (standards, codes, needs, requirements & specifications)	1.2. Toma en consideración de los condicionantes (normas técnicas y regulación, necesidades, requisitos y especificaciones)	S	11, 20-21
1.3. Setting of goals	1.3. Establecimiento de objetivos	S	10-11
2. FORMULATE:	2. FORMULAR:		
2.1. Creative solution generation (analysis)	2.1. Generación de soluciones creativas (análisis)	S	9-11
2.2. Evaluation of multiple solutions and decisionmaking (synthesis)	2.2. Evaluación de múltiples soluciones y toma de decisiones (síntesis)	S	9-11, 22-29
3. SOLVE:	3. RESOLVER:		
3.1. Fulfilment of goals	3.1. Evaluación del cumplimiento de objetivos	S	36-40
3.2. Overall impact and significance (contributions and practical recommendations)	3.2. Evaluación del impacto global y alcance (contribuciones y recomendaciones prácticas)	S	49

Índice

Índice de Figuras

1. Introducción

En la era digital en la que nos encontramos, la capacidad de poder gestionar las finanzas personales se ha convertido en una habilidad esencial para mantener la estabilidad económica y alcanzar objetivos financieros a largo plazo. Sin embargo, muchas personas tienen dificultades para controlar sus gastos de forma eficiente. Este problema se hace más evidente por la falta de herramientas accesibles y fáciles de usar que permitan a los usuarios llevar un seguimiento detallado de sus ahorros y proponerse diferentes metas financieras. Muchas de las soluciones disponibles en el mercado son complejas, muy costosas o piden facilitar información sensible como número de tarjeta o cuenta bancaria.

Esta carencia de herramientas intuitivas y accesibles que faciliten la gestión de ahorros personales representa una oportunidad para desarrollar una aplicación web que sea fácil de usar y pueda ser aprovechada por cualquier persona, independientemente de su conocimiento técnico o financiero.

1.1. Descripción

Este proyecto se centra en la creación de una aplicación web de gestión de ahorros personales. La idea principal es proporcionar a los usuarios una herramienta sencilla, intuitiva y accesible para llevar un control de sus finanzas. Esta aplicación permite a los usuarios registrar sus gastos y ahorros, establecer objetivos financieros, y visualizar sus datos mediante gráficos interactivos.

Para llevar a cabo este proyecto, se han utilizado diversas tecnologías que permiten tanto la creación de una interfaz de usuario atractiva, como la implementación de una funcionalidad robusta en el servidor, hecho en Python con el framework Flask, junto con una base de datos SQLite, que almacena la información de los usuarios. Por otro lado, en el frontend, se ha utilizado HTML, para construir la página visible, CSS para hacerla más vistosa y agradable y, finalmente, JavaScript, para añadir interactividad y crear los diferentes gráficos. En resumen, este proyecto busca que las personas aprendan a controlar mejor sus finanzas a través de una aplicación sencilla y fácil de usar.

1.2.Relación con las asignaturas del grado

Este proyecto se relaciona directamente con varias asignaturas del grado, integrando los conocimientos adquiridos en diversas áreas a lo largo del tiempo. El uso de Python en el backend se conecta con las asignaturas de Programación y Tecnologías Web, donde se abordaron tanto los principios de la programación como las técnicas para el desarrollo de aplicaciones web. Algunos ejemplos de la aplicación de estos conocimientos podrían ser la

estructuración de la base de datos, la lógica de la aplicación o la manipulación de datos en tiempo real. Además, la seguridad del proyecto, que incluye encriptación de contraseñas, está basada en conceptos estudiados en la asignatura Seguridad y Gestión de Derechos Digitales, donde se exploraron prácticas para la protección de contenidos digitales.

En cuanto al frontend, el uso de HTML, CSS y JavaScript se relaciona con las asignaturas Tecnologías Web y Frameworks para el Desarrollo Completo de Aplicaciones Web, que proporcionaron una base sólida para el diseño e implementación de interfaces de usuario interactivas y accesibles. Conceptos vistos también en las asignaturas Ideación, Diseño y Programación de Proyectos Interactivos y Aplicaciones y Usabilidad, donde se abordaron los principios de la interactividad y las mejores prácticas para crear experiencias de usuario atractivas, y aportaron herramientas y metodologías para garantizar que la interfaz sea intuitiva y fácil de usar, mejorando así la experiencia general del usuario.

En resumen, este proyecto demuestra cómo los conocimientos adquiridos en diversas asignaturas del grado se integran y unen entre sí para desarrollar una aplicación web completa y funcional. La combinación de principios de Programación, Tecnologías Web y Frameworks para el Desarrollo Completo de Aplicaciones Web ha permitido construir una estructura robusta tanto en el backend como en el frontend. Además, al aplicar algunas de las medidas de seguridad vistas en Seguridad y Gestión de Derechos Digitales se asegura la protección de los datos, y finalmente, los conceptos de Ideación, Diseño y Programación de Proyectos Interactivos y Aplicaciones y Usabilidad han sido clave para diseñar una interfaz de usuario intuitiva y atractiva. Cabe destacar que el desarrollo de esta aplicación se ha visto enormemente facilitado debido al enfoque multidisciplinar que ofrece el grado.

1.3. Objetivos

El desarrollo de esta aplicación de gestión de ahorros personales se ha planteado con varios objetivos generales y específicos en mente.

1.3.1. Objetivos Generales

- **Fomentar la Educación financiera personal:** Uno de los principales objetivos del proyecto es ayudar a los usuarios a adquirir una mejor comprensión y control de sus finanzas personales. La aplicación está diseñada para que puedan visualizar de forma clara y comprensible sus hábitos de gasto y ahorro, lo que puede ayudar a fomentar una mejor toma de decisiones económicas.
- **Proporcionar una herramienta accesible y fácil de usar:** El objetivo es desarrollar una aplicación que sea accesible para cualquier usuario, independientemente de su

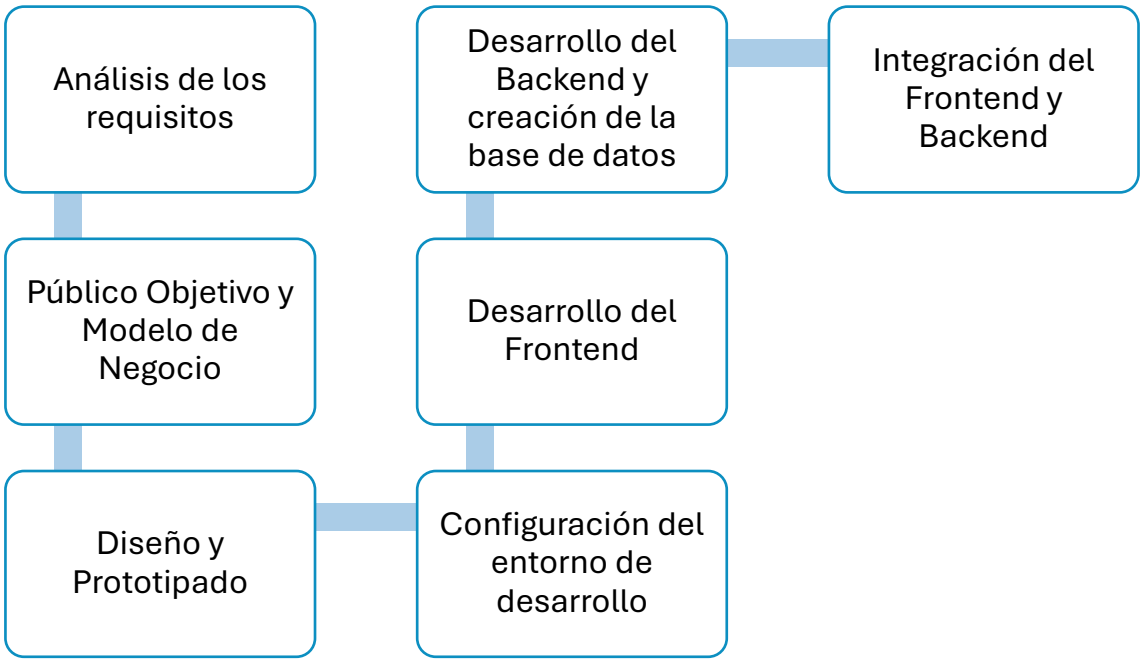
nivel de habilidad tecnológica. La interfaz y las funcionalidades deben ser claras e intuitivas.

1.3.2. Objetivos Específicos

- **Registro y seguimiento de ahorros y gastos:** Permitir a los usuarios registrar sus ingresos y gastos.
- **Establecimiento de objetivos**: Ofrecer a los visitantes la capacidad de establecer objetivos de ahorro específicos. Dichas metas pueden ser a corto, medio o largo plazo, pudiendo definir la cantidad de ahorro deseada y el mes de inicio y fin del objetivo.
- **Visualización de datos:** Implementar gráficos y estadísticas que permitan a los usuarios visualizar sus datos económicos de forma clara y comprensible.
- **Seguridad y privacidad:** Garantizar que la aplicación pueda proteger los datos de los clientes, implementando medidas de seguridad como la autenticación de usuarios y cifrado de datos.
- **Adaptabilidad y escalabilidad:** Diseñar la aplicación de manera que pueda adaptarse fácilmente a las nuevas funcionalidades y escalar con el tiempo, sin comprometer el rendimiento o la usabilidad.

2. Fases del proyecto

Para el diseño y desarrollo de la aplicación web de control de gastos personales se ha decidido seguir una metodología estructurada que contiene diferentes fases y puede observarse gráficamente en el siguiente esquema.



2.1. Análisis de los requisitos

Fase inicial del proyecto, se identifican y documentan las funcionalidades esenciales que la aplicación debe ofrecer, así como los requisitos no funcionales, como la seguridad y escalabilidad.

Actividades que lo componen:

a. Requisitos funcionales
b. Requisitos no funcionales

2.2. Público Objetivo y Modelo de Negocio

Se define el público destinatario de la aplicación como cualquier persona que desee controlar sus gastos e ingresos, siendo la única condición para utilizarla, el contar con un ordenador con acceso a internet, lo que la hace ampliamente accesible.

Al ser una aplicación con un público objetivo tan grande se ha optado por un modelo de negocio de tipo “Freemium”, que consiste en que la aplicación pueda ser usada por todo el mundo de forma gratuita, pero si se quiere utilizar alguna opción más avanzada se deberá pagar una suscripción mensual o anual.

2.3. Diseño y prototipado

Durante esta fase se elaboran prototipos de las distintas páginas y funcionalidades de la aplicación, utilizando la herramienta Balsamiq. Se define la estructura visual y de navegación del sitio para asegurar una experiencia de usuario intuitiva.

Actividades que lo componen:

a. Arquitectura de la aplicación
b. Prototipado
c. Diseño del logotipo de la aplicación

2.4. Configuración del entorno de desarrollo

En esta etapa se configura el entorno de desarrollo necesario, incluyendo la instalación de todas las dependencias y herramientas necesarias como Python, Flask, SQLite y el editor de código Visual Studio Code, facilitando un desarrollo organizado y eficiente.

Actividades que lo componen:

a. Instalación de Python

b. Creación del entorno virtual
c. Instalación de dependencias
d. Instalación de herramientas de desarrollo

2.5. Desarrollo del Frontend

Durante esta fase de desarrolla la interfaz de usuario utilizando HTML para crear la página, CSS para añadir estilos y JavaScript para añadir interactividad, centrándose en que la interacción del usuario con la aplicación fuese sea intuitiva.

2.6. Desarrollo del Backend y creación de la base de datos

En esta etapa se desarrollan las funcionalidades del servidor utilizando Flask y se establece la base de datos con SQLite, asegurando la seguridad de los datos y la correcta implementación de la lógica de la aplicación.

Actividades que lo componen:

a. Desarrollo del Backend
b. Creación de la base de datos

2.7. Integración del Frontend y Backend

Finalmente, se unen los componentes de Frontend y Backend, asegurando que trabajan de forma fluida en conjunto. Para comprobar el correcto flujo de datos y la interacción entre cliente y servidor, se realizaron diferentes pruebas de integración, garantizando así el funcionamiento óptimo de la aplicación completa.

3. Marco teórico

En este apartado se detallan las herramientas, conocimientos y técnicas utilizadas para desarrollar la aplicación web de control de gastos personales.

3.1. Tecnologías utilizadas

Para la creación de este proyecto, se han utilizado varias tecnologías que, combinadas, permiten desarrollar una aplicación web robusta, interactiva y fácil de usar. Las principales tecnologías empleadas son Python, con el framework Flask, HTML, CSS, JavaScript y SQLite.

3.1.1. Python

Python es un lenguaje de programación de alto nivel, interpretado y fácil de aprender, conocido por su simplicidad, eficiencia y legibilidad, además de que puede funcionar en muchos sistemas diferentes. Python aumenta la velocidad de desarrollo ya que cuenta con una sintaxis clara similar a la del inglés y a su gran biblioteca estándar de códigos reutilizables. Es ampliamente utilizado en el desarrollo web del lado del servidor, ciencia de datos, inteligencia artificial, automatización de scripts y desarrollo de software, entre otros. (Amazon Web Services, 2024; Python Docs, 2024)

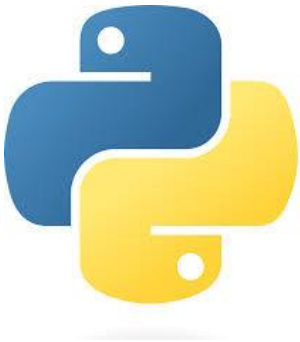


Figura 1: Logo de Python.

3.1.2. Flask

Flask es un microFramework para Python, utilizado para la creación de aplicaciones web. Es conocido por ser simple y flexible, permitiendo a los desarrolladores crear desde proyectos sencillos hasta proyectos complejos. Se dice que es un "microFramework" porque al instalarlo solo tiene unas cuantas herramientas para la creación de una aplicación web, pero si se necesita alguna otra funcionalidad se le pueden añadir extensiones. (Equipo editorial de IONOS, 2023; José Domingo Muñoz, Qué es Flask, 2017)

Algunas de las características por las que he elegido este Framework son:

- **Ligero y extensible:** Como ya se ha mencionado anteriormente, Flask ofrece unas herramientas básicas, pero se puede ampliar con muchas otras extensiones.
- **Motor de plantillas Jinja2:** Utiliza Jinja2 para generar HTML dinámico a través de plantillas, lo que facilita la creación de páginas dinámicas.
- **Soporte para testing:** Flask incorpora un servidor en el que poder realizar pruebas y ver los resultados obtenidos.

Figura 2: Logo de Flask.

3.1.3. HTML

HTML (HyperText Markup Language) es el lenguaje estándar para la creación de páginas web. Define la estructura y el contenido de una página web mediante el uso de etiquetas, que el dicen al navegador como mostrar cada elemento. (W3Schools, HTML Introduction)

Figura 3: Logo de HTML.

3.1.4. CSS

CSS (Cascading Style Sheets) es el lenguaje que se utiliza para describir como se va a presentar un documento HTML. CSS controla el diseño y el formato de las páginas web, permitiendo aplicar estilos, colores, fuentes y cambiar la disposición de los elementos HTML. Por lo tanto, CSS es muy importante para crear interfaces de usuario atractivas y coherentes. (W3School, CSS Introduction)

Figura 4: Logo de CSS.

3.1.5. JavaScript

JavaScript es un lenguaje de programación que permite la creación de contenido web dinámico e interactivo. Se ejecuta en el navegador del cliente y permite manipular el DOM (Document Object Model), además es capaz de realizar cálculos, modificar datos, validar inputs… mejorando la experiencia de uso en las páginas web. (W3Schools, JS Home; Maria Coppola, 2023)

Figura 5: Logo de JavaScript.

3.1.6. SQLite

SQLite es una librería de software que proporciona un sistema de gestión de bases de datos relacional. Es un motor de base de datos ligero, autónomo y que no requiere ningún software de servidor adicional, por ello se conoce a SQLite como una herramienta simple y eficiente, siendo una opción popular para muchas aplicaciones que requieren almacenamiento de datos local. Algunas de sus características más importantes son: (Equipo Editorial de IONOS, 2023)

- **Autocontenido:** No requiere de una configuración por separado en el servidor, ya que toda la base de datos se encuentra en un mismo archivo.
- **Rápido y ligero:** SQLite es muy eficiente en términos de recursos y rendimiento, siendo adecuado para proyectos pequeños y medianos.

- **Portabilidad:** Al ser una biblioteca autónoma, se puede portar de una forma muy fácil entre diferentes plataformas y sistemas operativos.

Figura 6: Logo de SQLite.

3.1.7. Visual Studio Code

Visual Studio Code, también conocido como VS Code, es un editor de código desarrollado por Microsoft, muy utilizado por los desarrolladores debido a su versatilidad, facilidad de uso y amplia gama de características. VS Code es un editor ligero pero potente, que admite gran cantidad de lenguajes de programación y herramientas de desarrollo, lo que lo convierte en una opción muy popular. A continuación, se detallan algunas características y beneficios de usar VisualStudio Code: (Microsoft, Visual Studio; Frankier Flores, 2022)

- **Editor de código inteligente:** VS Code proporciona un editor con características como el autocompletado, resaltado de sintaxis u ocultar fragmentos de código, características que facilitan la escritura de código limpio y eficiente, reduciendo los errores y mejorando la productividad.
- **Extensiones y personalización:** Una de las mayores fortalezas del editor es la enorme biblioteca de extensiones disponibles, que permiten personalizar y ampliar aún más sus capacidades, añadiendo soporte para nuevos lenguajes, depuradores.... Esto hace que Visual Studio Code se pueda adaptar fácilmente a las necesidades de casi cualquier proyecto.
- **Integración con control de versiones:** VS Code integra de forma nativa herramientas de control de versiones como Git, permitiendo a los desarrolladores gestionar sus repositorios directamente desde el editor, esto incluye características como commits, ramas, fusiones....

Estas son solo unas pocas de las muchas características que hacen a Visual Studio Code el entorno de desarrollo más usado. (Frankier Flores, 2022)

Figura 7: Logo de Visual Studio Code.

3.1.8. Herramientas de desarrolladores de Google

Las herramientas de desarrollador de Google, o también conocidas como Google Developer Tools, son un conjunto de herramientas de depuración y desarrollo web integradas directamente en Google Chrome, que brindan a los desarrolladores con una amplia de funciones, como inspeccionar, editar y depurar el código en tiempo real, permitiendo optimizar de forma más eficiente y eficaz las aplicaciones web. (Chrome, Chrome for developers)

Figura 8: Logo de Google Developer Tools.

3.1.9. Balsamiq

Balsamiq es una herramienta para el diseño y creación de prototipos de interfaces de usuario, principalmente se utiliza para el desarrollo de representaciones simplificadas y esquemáticas de la interfaz de usuario, también conocidas como wireframes, que son esenciales en las primeras etapas del diseño de una página o aplicación web.

Lo que hace especial a Balsamiq es su interfaz intuitiva que permite a los usuarios arrastrar y soltar los elementos en el lienzo. Además, cuenta con una gran biblioteca de componentes pre-diseñados, como botones, menús…, lo que facilita la creación rápida

de prototipos. Otra característica de esta herramienta es que los wireframes generados con ella tienen un estilo de dibujo a mano alzada, lo que ayuda a enfatizar el concepto de prototipo y evitar distracciones con otros detalles estéticos.

Balsamiq también fomenta la colaboración, permitiendo a los usuarios compartir y revisar prototipos fácilmente, así como añadir comentarios y anotaciones directamente en el diseño. (Balsamiq, What is Balsamiq Wireframes?; smartbrand, ¿Qué es Balsamiq?)

Figura 9: Logo de Balsamiq.

3.2. Principios del desarrollo web

El desarrollo web moderno se sustenta de una serie de principios y arquitecturas que facilitan la creación de aplicaciones web, dos de estos principios son el modelo de desarrollo Model-View-Controller (MVC) y las API RESTful. Ambos principios son fundamentales para organizar el código y manejar la comunicación entre los diferentes componentes de una web.

3.2.1. Model-View-Controller (MVC)

Este patrón de diseño divide la aplicación en tres componentes principales: Modelo, Vista y Controlador, lo que permite gestionar de manera eficiente la lógica de la página, la interfaz de usuario y el flujo de datos. (Uriel Hernández, 2015)

- **Modelo:** Representa los datos y la lógica de la aplicación, siendo responsable de gestionar el acceso y la manipulación de los datos, generalmente interactuando con una base de datos.
- **Vista**: Es la parte de la aplicación que interactúa directamente con el usuario, le presenta los datos del modelo y recoge la información introducida ara enviársela al controlador.
- **Controlador:** Actúa como intermediario entre el modelo y la vista, recibe los datos de entrada de la vista, los procesa (posiblemente interactuando con el modelo) y le devuelve una respuesta.

El uso de este patrón de diseño trae algunos beneficios como podrían ser la separación de responsabilidades, que hace que el código sea más fácil de gestionar y mantener, agiliza el desarrollo, ya que los desarrolladores pueden trabajar en diferentes componentes de la aplicación, y finalmente, mejora la reutilización de código.

3.2.2. RESTful APIs

Las APIs RESTful son un estándar para diseñar servicios web que permiten la comunicación entre diferentes sistemas a través de internet, utilizan los métodos estándar de HTTP (Get, Post, Put y Delete) para realizar operaciones CRUD (Create, Read, Update, Delete) en recursos representados en formato JSON o XML. (Amazon Web Services, ¿Qué es una API RESTful?)

Algunos de los beneficios que ofrece su uso en el desarrollo de aplicaciones web son:

- **Interoperabilidad:** Facilita la comunicación entre diferentes sistemas, independientemente de las tecnologías utilizadas.
- **Mantenibilidad:** Al usar métodos estándar HTTP y estructuras de URI, las APIs RESTful son fáciles de entender y mantener.
- **Flexibilidad:** Permite que la API evolucione sin romper integraciones existentes.

4. Desarrollo del proyecto

4.1. Análisis de los requisitos

En esta sección se definen los requerimientos mínimos que la aplicación debe tener para considerarse un éxito, en este caso, está dividida en dos partes, los requisitos funcionales y los no funcionales.

4.1.1. Requisitos funcionales

Los requisitos funcionales describen las funciones del sistema, centrándose en lo que el sistema debe hacer y cómo debe interactuar con los usuarios y otros sistemas.

- **Registro y autenticación de usuarios:** El sistema debe permitir que los usuarios se registren mediante el formulario correspondiente, que recogerá su nombre, dirección de correo electrónico y contraseña. Así mismo, el sistema debe permitir autenticarse utilizando su correo electrónico y contraseña en el formulario de inicio de sesión.

- **Gestión de ahorros:** La aplicación debe calcular y mostrar la media de ahorro de los usuarios, así como representarlos en gráficos interactivos.
- **Gestión de objetivos de ahorro:** El sistema ha de permitir a los usuarios establecer objetivos de ahorro, especificando la cantidad deseada y el período de tiempo. En estos objetivos se calculará la cantidad mensual requerida para alcanzar el objetivo y mostrar la cantidad total restante para conseguirlo.

4.1.2. Requisitos no funcionales

Los requisitos no funcionales describen los atributos del sistema que no están relacionados directamente con las funcionalidades de la aplicación, pero que son importantes para su operación y calidad.

- **Rendimiento:** El sistema debe ser capaz de manejar múltiples solicitudes simultáneas sin degradas significativamente el rendimiento.
- **Seguridad:** El sistema debe encriptar todas las contraseñas de los usuarios, así como manejar sus sesiones de forma segura, usando cookies seguras y tokens de inicio de sesión.
- **Usabilidad:** La interfaz de usuario debe ser intuitiva y fácil de usar.

4.2. Público objetivo y Modelo de Negocio

El público objetivo de esta aplicación web es muy diverso, cubriendo diferentes grupos de personas, sin importar la edad, ya que el único requisito es poder tener acceso a un ordenador con conexión a internet. La página está dirigida a personas interesadas en gestionar y controlar sus gastos y ahorros de manera eficiente, de manera que podría ser tanto una persona joven que quiere proponerse un pequeño objetivo de ahorro a una persona adulta que quiere controlar como utiliza su dinero.

Se ha optado por un modelo “Freemium” para la explotación económica de la aplicación, esto significa que el uso de la aplicación base es gratuito, pero si se quiero utilizar o mejorar alguna funcionalidad más avanzada se deberá pagar una suscripción mensual o anual. Algunas de las funcionalidades que se podrían implementar con la suscripción serían poder mostrar más movimientos en la tabla de la página Perfil o la opción de crear una cuenta “familiar”, similar al modelo de cuentas compartidas de plataformas como Spotify, en la que se podría invitar a más personas para que estas pusiesen en común los ingresos y gastos, fomentando así la colaboración financiera dentro del hogar.

Otra opción, podría ser implementar una suscripción para negocios, costando un poco más que la suscripción estándar, pero incluyéndola también a esta, es decir, con la suscripción de negocio se tendría acceso a funcionalidades exclusivas de este plan, como por ejemplo la generación de informes financieros, avisos de pagos a proveedores…, así como a las de la suscripción premium personal.

Para conseguir suficientes usuarios se optaría por poner anuncios patrocinados en diferentes plataformas, como YouTube, Spotify o el propio buscador de Google para posicionar más arriba la página. Para incentivar que los nuevos usuarios se conviertan a premium se podrían lanzar ciertas ofertas como que para los primeros X clientes que se suscriban tendrán un precio reducido durante un tiempo.

4.3. Diseño y prototipado

4.3.1. Arquitectura de la aplicación

4.3.1.1. Patrón Model-View-Controller

Como se ha mencionado anteriormente, la aplicación web sigue este patrón, que es ampliamente utilizado para el desarrollo web por su capacidad de separar la lógica de la aplicación, la interfaz de usuario y el control de los inputs del usuario. Consta de tres componentes:

- **Model:** Que representa la estructura de datos de la aplicación y la lógica de negocio. En el caso de este proyecto, el modelo está representado por las clases de Python que definen las entidades de la base de datos y sus relaciones. Se utiliza SQLAlchemy como ORM (Obejct-Relational Mapping), para interactuar con la base de datos SQLite, permitiendo usar las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) de una forma más sencilla y eficiente.
- **Vista:** Gestiona la presentación de la información al usuario, en este caso son las plantillas HTML renderizadas mediante el motor de plantillas Jinja2.
- **Controlador:** Hace de intermediario entre la Vista y el Modelo, manejando las interacciones del usuario, en esta web está implementado como funciones en el archivo de rutas que hace uso de Flask. Estas funciones procesan las solicitudes HTTP e interactúan con el modelo para obtener o modificar datos.

4.3.1.2. Interacción cliente servidor

La página sigue un modelo de interacción cliente-servidor, donde el cliente realiza solicitudes al servidor, que las procesa y le envía una respuesta de vuelta. Dicha interacción está gestionada principalmente mediante HTTP.

- **Solicitudes HTTP:** Las solicitudes del cliente son generalmente de tipo GET o POST, para obtener o enviar datos respectivamente. Por ejemplo, al iniciar sesión, el navegador del cliente envía una solicitud POST al servidor con las credenciales.
- **Respuestas HTTP:** El servidor procesa las solicitudes, interactúa con el modelo y responde con la información necesaria. La respuesta podría ser una página HTML renderizada o un JSON en el caso de una API.

4.3.1.3. Diagrama de componentes

El siguiente diagrama representa visualmente los distintos componentes que forman parte de la arquitectura de la aplicación web, esto nos ayuda a entender cómo interactúan entre sí los componentes principales y qué tecnologías utilizan.

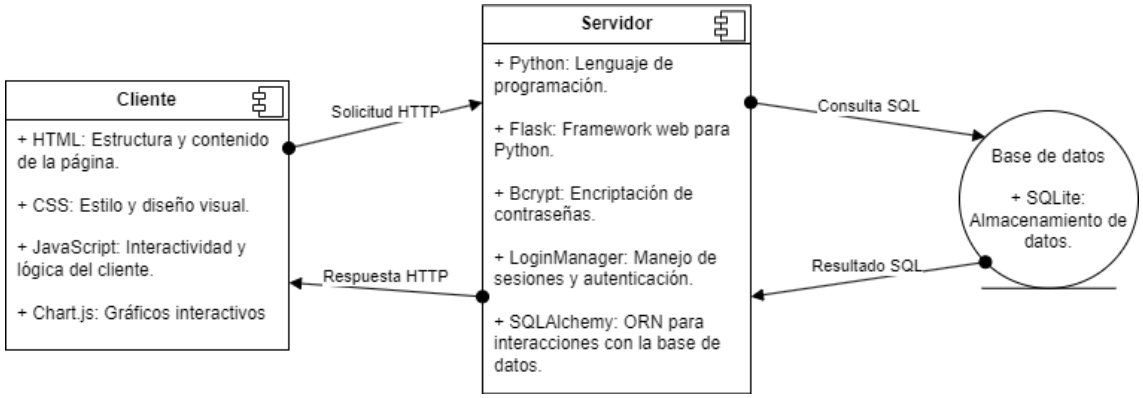


Figura 10: Diagrama de componentes de la aplicación web. Fuente: Elaboración propia

El diagrama está formado por tres componentes principales, que a su vez están formado por más subcomponentes:

- Cliente (Frontend): Que a su vez está formado por más componentes.
  - HTML: Lenguaje de marcado utilizado para la estructura y el contenido de las páginas web.
  - CSS: Lenguaje de estilos utilizado para el diseño visual y la presentación de las páginas web.
  - JavaScript: Lenguaje de programación utilizado para añadir interactividad y lógica al lado del cliente.

- Chart.js: Biblioteca de JavaScript utilizada para la creación de gráficos interactivos.

- Servidor (Backend):
  - Python: Lenguaje de programación.
  - Flask: Framework web para Python.
  - Bcrypt: Encriptación de contraseñas
  - LoginManager: Manejo de sesiones y autenticación.
  - SQAlchemy: ORM para interacciones con la base de datos.

4.3.2. Prototipado

Después de definir los objetivos, el público objetivo y los requerimientos mínimos que debe tener la aplicación, empezamos con el diseño de un prototipo.

4.3.2.1. Prototipo Menú

Empezamos diseñando el prototipo del menú, que será prácticamente igual en las 3 páginas principales que formarán la aplicación web, para asegurar que los usuarios puedan navegar por las diferentes páginas de una forma fácil e intuitiva. En este prototipo hay un texto a la izquierda que normalmente será el nombre de la página en la que se encuentra, y en la parte derecha hay tres barras horizontales que representan otro menú desplegable, que contendrá los enlaces para navegar por la aplicación y la opción de cerrar sesión.

Figura 11: Prototipo del menú de la aplicación. Fuente: Elaboración propia

4.3.2.2. Prototipo página Dashboard

Seguimos con la que será la primera página que los usuarios vean al entrar en la aplicación después de iniciar sesión, se trata del Dashboard, en el que se mostrarán diferentes datos como los ingresos, gastos y ahorros totales, varios gráficos para visualizar los datos anteriores, un semáforo que informará al usuario de si los gastos representan un gran porcentaje de los ingresos o si por otra parte son moderados o bajos, también se mostrará un ranking con los tres mayores gastos, el coste total que tiene cada uno y el porcentaje de los ingresos totales que representan.

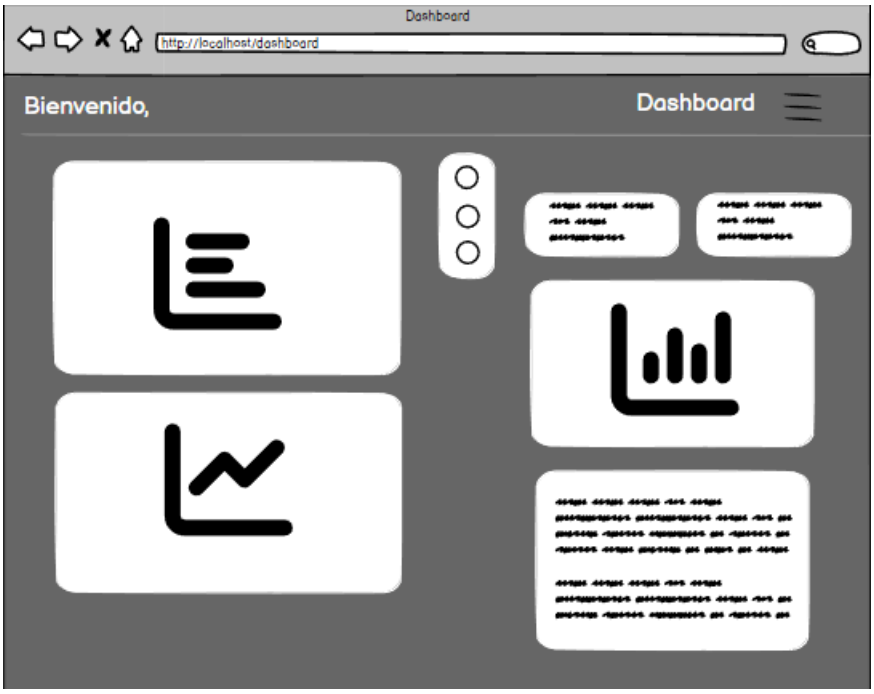


Figura 12: Prototipo de la página "Dashboard". Fuente: Elaboración propia

4.3.2.3. Prototipo de la página de Gastos

A continuación, procedemos con el prototipo de la página de control de gastos, en la que habrá diferentes formularios para añadir el presupuesto y los gastos, así como su categoría y color, del mes seleccionado en la parte superior. También contará con un gráfico que mostrará los gastos y la cantidad del presupuesto restante y una tabla en la que se añadirán también la categoría y el coste de los diferentes desembolsos.

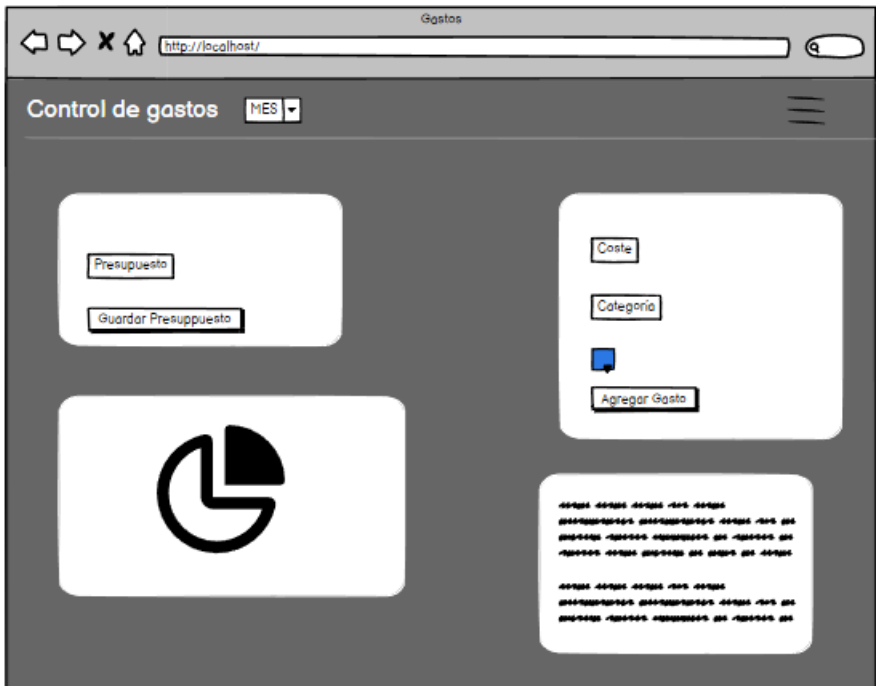


Figura 13: Prototipo de la página "Control de gastos". Fuente: Elaboración propia

4.3.2.4. Prototipo de la página de Ahorros

En esta página, como en la anterior, se podrá seleccionar el mes en la parte superior, habrá también un apartado en el que se mostrarán algunos datos, como el gasto total en el mes seleccionado, la cantidad de ahorro o la media de ahorro de los meses anteriores y en el gráfico de podrá visualizar el ahorro que se ha conseguido cada mes hasta el actual. Por otro lado, en la parte derecha de la página habrá un formulario para añadir objetivos de ahorro, con opciones para agregar la cantidad que se desea ahorra, así como el intervalo de tiempo para conseguirlo, debajo de esta sección habrá otra que mostrará los objetivos creados con el formulario anterior.

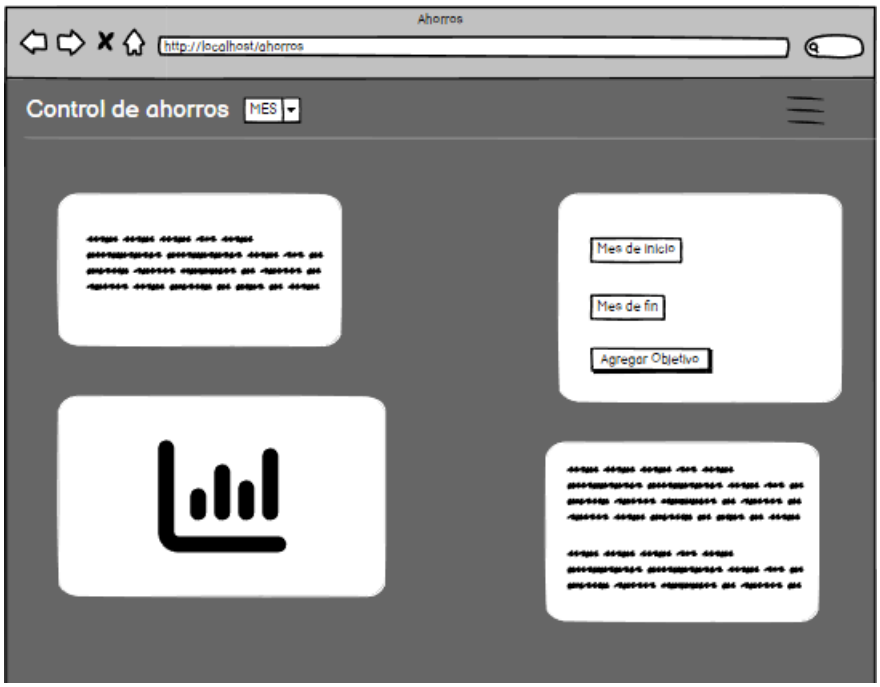


Figura 14: Prototipo de la página "Control de ahorros". Fuente: Elaboración propia

4.3.2.5. Prototipo de la página de Perfil

Finalmente, empezamos con el prototipado de la página de perfil, la parte más personal para el usuario de la aplicación. Al acceder a esta página el usuario verá tres secciones diferenciadas, dos en la parte izquierda y una en la derecha. Empezando por la parte izquierda, se mostrará la foto de perfil que el usuario haya elegido, en caso de no haberlo hecho se mostrará una imagen por defecto, si quiere cambiarla solo tendrá que hacer pulsar sobre ella y elegir la que más le guste de entre sus archivos. Debajo de esta sección encontrará otra que muestran sus credenciales, como el nombre de usuario o correo electrónico registrado, y le permitirá cambiar la contraseña de una manera rápida y segura.

Continuando con la parte derecha, se mostrará una tabla con los últimos movimientos realizados, detallando su tipo, concepto, mes en el que se ha realizado y la cantidad, facilitando así el seguimiento de sus actividades financieras.

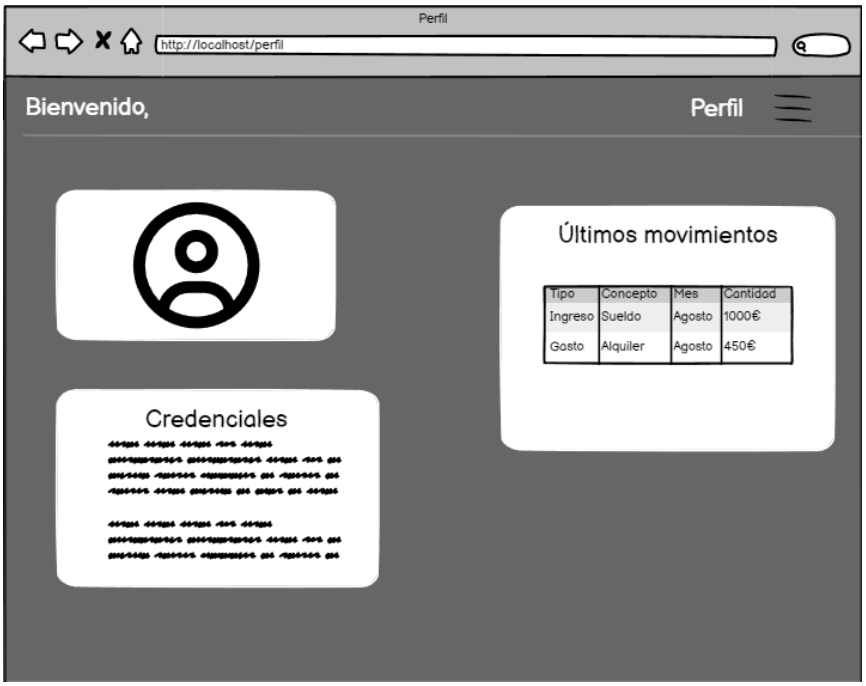


Figura 15: Prototipo de la página "Perfil". Fuente: Elaboración propia

4.3.2.6. Mapa de navegación

Cuando un usuario intente entrar en la aplicación se comprobará si este tiene una sesión activa o no, en caso afirmativo, se le llevará a la página de Dashboard, desde donde podrá navegar hasta las páginas de Control de gastos y Control de ahorros sin ningún impedimento, pudiendo cerrar sesión en cualquier momento y siendo redirigido a la página para iniciar sesión. En caso de no tener una sesión activa, el sistema dirigirá al usuario a la página correspondiente para activarla, en este punto, el visitante podrá iniciar sesión con sus credenciales o, si no dispone de una cuenta, podrá entrar al formulario de registro para crear una y poder acceder a la aplicación, una vez tenga la cuenta creada solo debe iniciar sesión y navegar por las diferentes páginas.

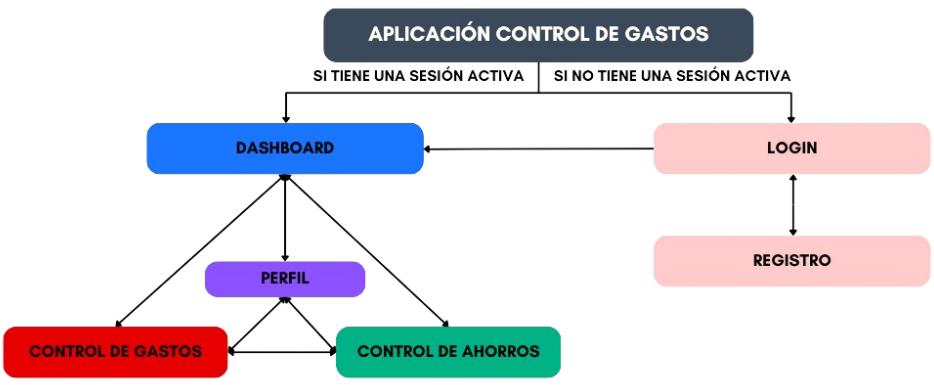


Figura 16: Mapa de navegación de la aplicación. Fuente: Elaboración propia

4.3.3. Diseño del logotipo de la aplicación

En esta sección se ha diseñado el logotipo de la aplicación, con la ayuda de la inteligencia artificial integrada en el navegador web Microsoft Edge. Para que la IA pudiese crear un logotipo que representase la aplicación se le especificó alguna idea clave, como que el foco principal fuese un cerdo parecido a una hucha, con monedas y con un estilo de dibujo simple y minimalista. Se eligieron estas palabras porque normalmente se asocia la palabra de ahorro a una hucha y esta suele tener forma de cerdo, también quería tuviese una estética moderna y fuese fácil de recordar, por eso debía ser relativamente simple y minimalista.

Figura 17: Logotipo de la aplicación. Fuente: Elaboración propia

4.4. Configuración del entorno de desarrollo

En esta sección configuraremos nuestro entorno de desarrollo, para facilitar la escritura, prueba y depuración del código, y garantizar unas condiciones homogéneas en todo momento.

4.4.1. Instalación de Python

Ya que la página está construida sobre Python y el microframework Flask, el primero paso es instalar Python y Pip. Para ello, entraremos en la página oficial de Python y descargaremos la versión 3.12.4, por ser la más reciente en el momento de creación del proyecto. Una vez descargado el archivo Python-3.12.4.exe, solo hay que hacer ejecutarlo y se abrirá el instalador, en esta ventana solo hay que aceptar que Python se añada al PATH, para que podamos invocarlo desde cualquier carpeta de nuestro ordenador, y seguir los pasos de instalación. Cuando finalice la instalación podemos comprobar que todo ha ido bien ejecutando el comando "*python –-version*" en una consola de comandos.

Figura 18: Captura del instalador de Python. Fuente: Elaboración propia

4.4.2. Creación del entorno virtual

Una vez finalizada la instalación de Python, podemos crear el entorno virtual en el que desarrollaremos el proyecto, este paso nos permite aislar las dependencias del proyecto, evitando así conflictos con otras aplicaciones que puedan estar usando versiones diferentes de las mismas bibliotecas.

Para la creación del entorno usaremos el siguiente comando: "*python -m <nombre del entorno>*", una vez creado el entorno podremos activarlo con el comando *".\<nombre del entorno>\Scripts\activate*". (Stephen Sanwo, How Set Up a Virtual Environment in Python)

```
Microsoft Windows [Versión 10.0.22631.3880]
(c) Microsoft Corporation. Todos los derechos reservados.

D:\GTDM>python -m venv entorno

D:\GTDM>.\entorno\Scripts\activate

(entorno) D:\GTDM>
```

Figura 19: Comandos para la creación y activación del entorno virtual. Fuente: Elaboración propia

4.4.3. Instalación de dependencias

Ahora vamos a instalar las bibliotecas necesarias, estas se instalan mediante el Pip, el sistema de gestión de paquetes de Python, siguiendo la siguiente estructura, "*pip install <nombre de la dependencia>",* en este caso el comando para instalar todas las dependencias utilizadas sería: "*pip install flask flask-bcrypt flask-login Flask-SQLAlchemy flask-wtf WTForms pillow*".

```
(entorno) D:\GTDM>pip install flask flask-bcrypt flask-login Flask-SQLAlchemy flask-wtf WTForms
Requirement already satisfied: flask in d:\gtdm\entorno\lib\site-packages (3.0.3)
Collecting flask-bcrypt
  Using cached Flask_Bcrypt-1.0.1-py3-none-any.whl (6.0 kB)
Collecting flask-login
  Using cached Flask_Login-0.6.3-py3-none-any.whl (17 kB)
Requirement already satisfied: Flask-SQLAlchemy in d:\gtdm\entorno\lib\site-packages (3.1.1)
Collecting flask-wtf
  Using cached flask_wtf-1.2.1-py3-none-any.whl (12 kB)
Collecting WTForms
  Using cached wtforms-3.1.2-py3-none-any.whl (145 kB)
```

Figura 20: Comando para la instalación de las dependencias. Fuente: Elaboración propia

Si queremos comprobar que se han instalado correctamente solo debemos introducir el comando "*pip list*" para que se nos muestre una lista con todas las bibliotecas instaladas en este entorno.

```
(entorno) D:\GTDM>pip list
Package           Version
----------------- -------
bcrypt            4.2.0
blinker           1.8.2
click             8.1.7
colorama          0.4.6
Flask             3.0.3
Flask-Bcrypt      1.0.1
Flask-Login       0.6.3
Flask-SQLAlchemy  3.1.1
Flask-WTF         1.2.1
greenlet          3.0.3
itsdangerous      2.2.0
Jinja2            3.1.4
MarkupSafe        2.1.5
pip               21.2.4
setuptools        58.1.0
SQLAlchemy        2.0.31
typing_extensions 4.12.2
Werkzeug          3.0.3
WTForms           3.1.2
```

Figura 21: Comando para mostrar todas las bibliotecas. Fuente: Elaboración propia

4.4.4. Instalación de las herramientas de desarrollo

En esta sección vamos a descargar e instalar la última herramienta de desarrollo que se ha utilizado durante este proyecto, se trata del editor de código fuente Visual Studio Code, se ha optado por este editor por la alta popularidad y gran número de extensiones disponibles, que ayudan a que el progreso de la aplicación sea lo más rápido y fácil posible. Para proceder con la descarga solo debemos entrar en la página oficial de VS Code y descargar el instalador correspondiente a nuestro sistema operativo, después lo iniciamos y seguimos los pasos que se nos indiquen. Con este paso finalizado, ya podemos empezar con la programación de la página.

4.5. Desarrollo del Frontend

Haciendo uso de Visual Studio Code, desarrollamos la parte frontend de la aplicación, es decir, la que el usuario verá y con la que interactuará, que ha sido construida con HTML,

para definir la estructura y contenido de las diferentes páginas, cada componente visual y de contenido de la aplicación, desde los formularios de entrada de datos hasta los gráficos y tablas, está representado por este lenguaje. Seguimos con CSS, que complementa al HTML proporcionando el diseño y el estilo visual de las páginas, por ejemplo, cambiando colores, fuentes, márgenes, alineaciones y disposiciones de los elementos en la interfaz. Finalmente, usamos el lenguaje de programación JavaScript para añadir interactividad y dinamismo a la aplicación, en este caso, se ha integrado con Chart.js, una biblioteca que facilita la creación de gráficos interactivos y visualmente atractivos, para representar algunos datos como los gastos o los ahorros totales.

4.6. Desarrollo del Backend y Creación de la Base de Datos

4.6.1. Desarrollo Backend

Se ha elegido Python como lenguaje de programación por su claridad y eficiencia, permitiendo desarrollar un código limpio y mantenible, además en el lenguaje con el que me encuentro más cómodo, ya que es el que se ha usado para la mayoría de trabajos durante el grado. Por otro lado, se ha elegido el microframework Flask, por su enfoque minimalista, que proporciona las herramientas básicas necesarias para construir la aplicación web, pero sin imponer una estructura rígida.

Flask se encarga de manejar las rutas y solicitudes HTTP, cada ruta está asocidad a una función que define la lógica de negocio correspondiente. Son estas funciones las que se encargan de procesar las solicitudes del cliente, interactuar con la base de datos, y devolver las respuestas adecuadas. Gracias a la estructura modular de Flask se ha podido organizar el código en diferentes módulos y paquetes, mejorando la legibilidad y facilitando el mantenimiento.

Algo que no se puede dejar de lado en este tipo de proyectos es la seguridad, por eso se ha implementado un sistema de autenticación y autorización utilizando la extensión Flask-Login, herramienta que simplifica la gestión de sesiones de usuario, asegurando que solo los usuarios autenticados pueden acceder a la aplicación. Además, para proteger sus contraseñas se ha utilizado la biblioteca Bcrypt, que proporciona métodos robustos para el hashing, que consiste en transformar cadenas de texto de cualquier longitud en otra cadena de longitud fija llamada hash, y la verificación de contraseñas.

4.6.2. Creación de la Base de Datos

La base de datos es el componente en el que se almacena toda la información necesaria para el funcionamiento de la aplicación. Se ha optado por utilizar SQLite debido a su

simplicidad y eficiencia para aplicaciones de pequeño y mediano tamaño, además de que se trata de un base de datos ligera que no requiere una configuración de servidor especial, lo que facilita su integración y despliegue.

Para interactuar con la base de datos se utiliza SQLAlchemy, uni biblioteca de ORM (Object-Relational Mapping) para Python, que permite trabajar con la base de datos utilizando objetos de Python, en lugar de escribir directamente consultas SQL, cosa que simplifica el código y mejora la seguridad al prevenir inyecciones de código SQL.

Se ha diseñado la base de datos para que toda la información necesaria esté correctamente estructurada y accesible en tablas, que almacenan información de los usuarios como presupuestos, gastos y demás datos relevantes para el funcionamiento de la aplicación. Cabe añadir que cada tabla ha sido definida con las relaciones apropiadas para mantener la integridad relacional y permitir consultas eficientes.

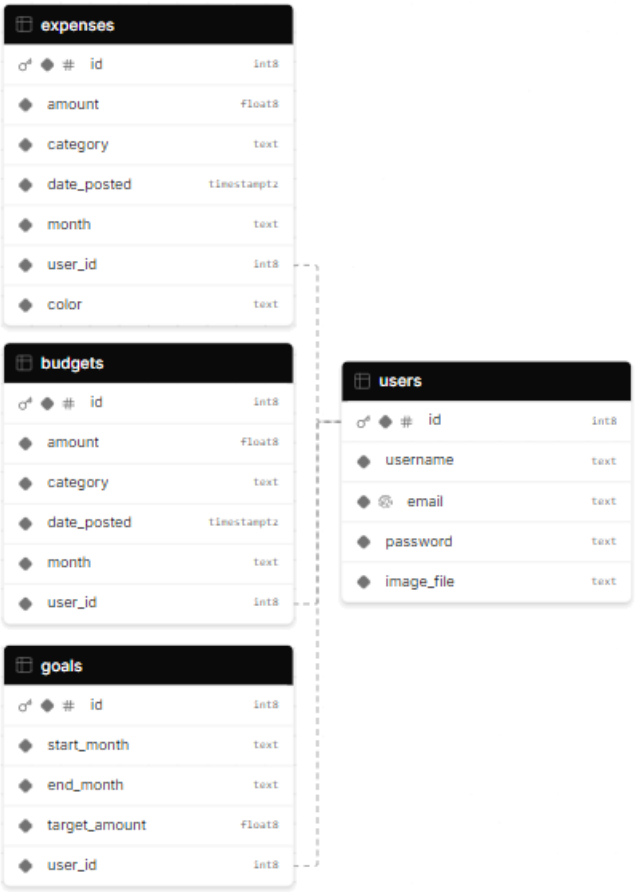


Figura 22: Diagrama de Entidad-Relación de la base de datos. Fuente: Elaboración propia

En resumen, el desarrollo backend de la aplicación ha implicado la implementación de la lógica de negocio y la gestión de usuarios utilizaqndo Python y Flask, junto con la creación y gestión de una base de datos SQLite a través de SQLAlchemy. Gracias a este enfoque se

ha construido una aplicación web robusta, segura y eficiente, preparada para satisfacer las necesidades actuales y futuras de los usuarios.

4.7.Integración del Frontend y Backend

Después de desarrollar y probar por separado el frontend y el backend de la página para su correcto funcionamiento, sigue integrarlos para que funcionen juntos y formen la aplicación final.

Una vez unidos los dos componentes y después de confirmar el correcto funcionamiento de la lógica de la aplicación, así como la visualización y la interacción con los datos de la base de datos, procedimos a realizar varias pruebas para asegurar un funcionamiento fluido tanto en el frontend como en el backend. Las pruebas son cruciales para validar el flujo adecuado de datos entre la interfaz de usuario y el servidor, y para garantizar el comportamiento esperado de la aplicación cuando interactúan sus diferentes componentes.

Durante las pruebas, se puedo especial énfasis en validar el correcto funcionamiento del inicio de sesión y registro de los usuarios, y de la gestión que estos pueden hacer de su información dentro de la aplicación, como añadir gastos, objetivos…. Además de asegurar que solo los usuarios autorizados pueden acceder a la aplicación y hacer uso de ella.

Algunas de las pruebas que se realizaron incluyeron:

- **Validación de Inicio de sesión y Registro:** Con esta prueba se puedo confirmar que los usuarios pueden registrarse y acceder a sus cuentas de manera segura, además se verificó que las contraseñas se encriptan y se almacenan correctamente en la base de datos.
- **Sincronización de datos:** Para asegurar que los datos ingresados por el usuario en el frontend se almacenan correctamente en la base de datos, además de que estos mismo se recuperen y muestren de la forma deseada.

Con estas pruebas, entre otras, se garantizó que la aplicación funciona de forma coherente y segura, proporcionando una experiencia de usuario fluida y confiable.

4.8.Estructura de directorios del proyecto

Después de finalizar el desarrollo encontramos una estructura similar a la mostrada en la figura 22, en la que se puede observar que hay 3 archivos y carpetas principales, que son "*instance*", carpeta creada automáticamente por la aplicación, en caso de no existir, y donde se guarda la base de datos, "*static*", que centraliza todos los archivos estáticos que son

necesarios para la correcta visualización y funcionamiento de la aplicación en el lado del cliente y que tiene a su vez 4 sub carpetas con nombres autodescriptivos:

- **css:** Que contiene los archivos que definen el estilo y la presentación visual de las diferentes páginas.
- **js:** Contiene los archivos JavaScript que dotan a la interfaz de usuario de dinamismo e interactividad.
- **logo:** Directorio en el que se guarda el lodo de la aplicación en diferentes tamaños.
- **profile_pics:** Directorio en el que se almacenan las fotografías que usan los usuarios como imagen de perfil.

Por último, encontramos la carpeta "*templates*", donde se encuentran las plantillas que el servidor renderizará y enviará a los usuarios con sus respectivos datos.

Pasando a los archivos, encontramos "*app.py*", el núcleo de la aplicación, donde se centraliza la configuración y ejecución del servidor web, así como las rutas que determinan cómo se manejarán las distintas solicitudes HTTP. Además, establece la conexión con la base de datos, gestiona la autenticación de usuarios y coordina la interacción entre los diferentes componentes de la aplicación.

Seguidamente, observamos "*forms.py*", que contiene las definiciones de los formularios utilizados en la página, dichos formularios son esenciales para interactuar con los usuarios, ya que permiten la recolección de datos que luego son enviados al servidor. Hace uso de una extensión de Flask llamada Flask-WTF para definir la estructura de los formularios, validar la entrada de datos y gestionar los errores, asegurando así que los datos recibidos en el backend cumplen los requisitos especificados antes de ser procesados o almacenados en la base de datos.

Finalmente, llegamos a "*models.py*", encargado de definir los modelos de datos de la aplicación, representando las tablas de la base de datos y sus relaciones, estos modelos son fundamentales para la interacción con la base de datos, ya que permiten realizar operaciones CRUD, es decir, de creación, lectura, actualización y eliminación de registros. Como ya se ha mencionado anteriormente, hace uso de SQLAlchemy, una herramienta que facilita la gestión de la base de datos de una manera estructura y eficiente, vinculando directamente objetos Python con las tablas de la base de datos.

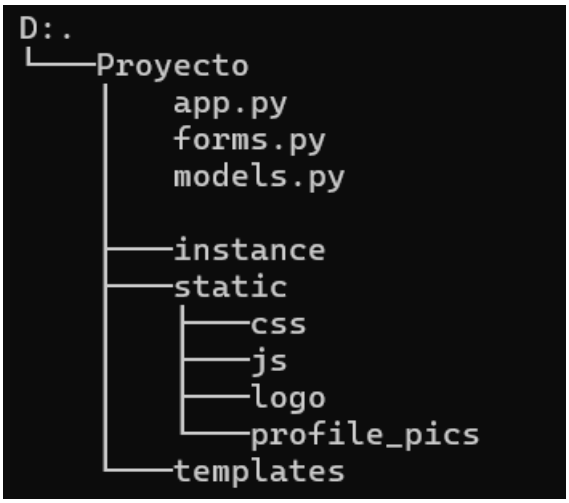


Figura 23: Estrutura de directorios del proyecto simplificado. Fuente: Elaboración propia

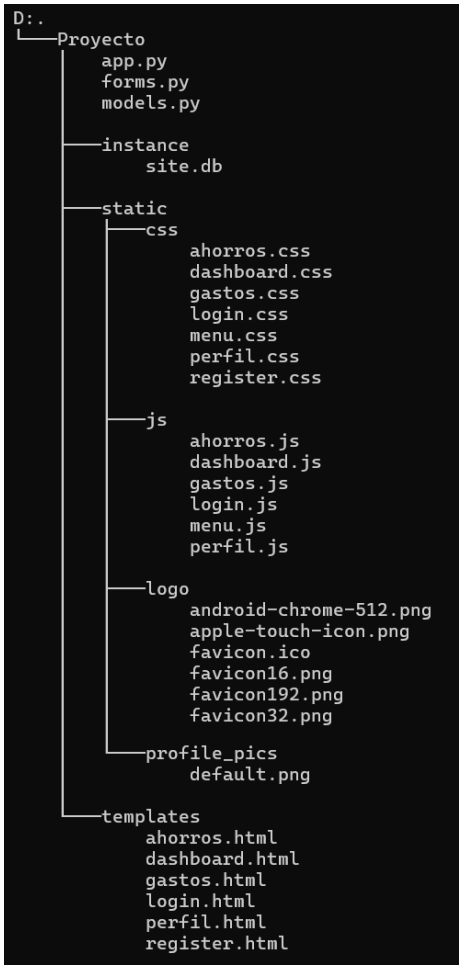


Figura 24: Estructura de directorios del proyecto completa. Fuente: Elaboración propia

4.9.Evaluación de los resultados

Después de realizar las pruebas anteriormente mencionadas, se puede decir que los objetivos establecidos al inicio del proyecto han sido alcanzados satisfactoriamente, se ha logrado

desarrollar una aplicación web que permite a los usuarios registrar sus ingresos y gastos, visualizar gráficos interactivos sobre sus finanzas y administrar sus ahorros, además se ha realizado de forma que se garantiza la seguridad de sus datos mediante la implementación de técnicas de encriptación y manejo seguro de sesiones.

En la página de inicio o Dashboard, los usuarios encuentran un resumen visual de su situación financiera y pueden ver gráficos interactivos que muestran la distribución de sus gastos y ahorros, gracias a esto pueden tener una perspectiva clara de su estado financiero rápidamente.

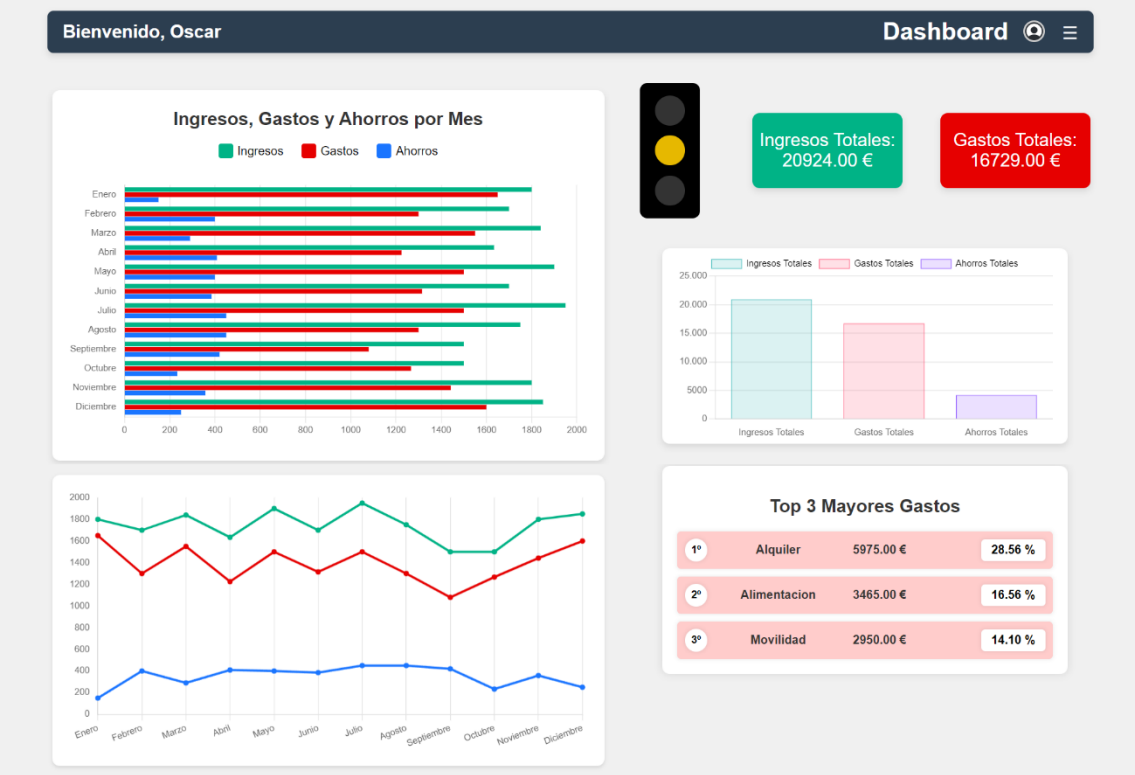


Figura 25: Captura de la página Dashboard. Fuente: Elaboración propia

En la sección de control de gastos, los usuarios pueden agregar sus despensas, o eliminarlas en caso de haberse equivocado al introducirlas, así como ver en un gráfico qué porcentaje de su presupuesto mensual representa ese gasto en concreto o el resto de ellos que hayan añadido con anterioridad. Esta página está diseñada para ser intuitiva, permitiendo a los usuarios seleccionar el mes al que se desea añadir un ingreso o gasto, y la cantidad y categoría de este, funcionalidad esencial para mantener un registro preciso de todos los desembolsos.

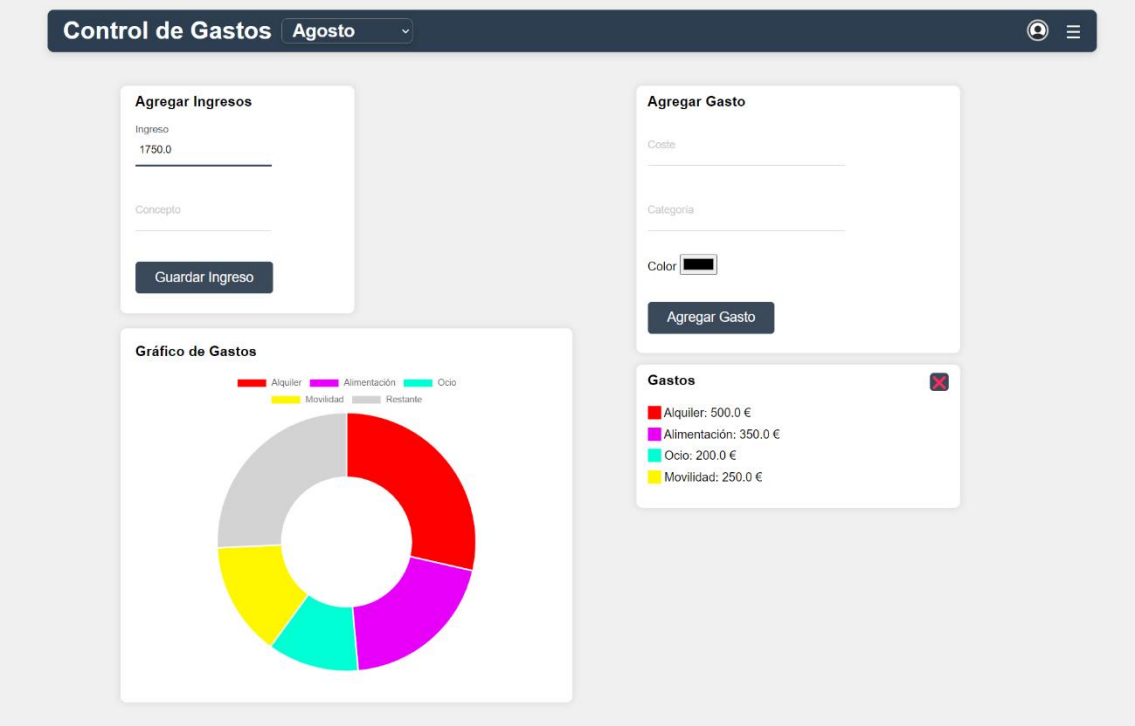


Figura 26: Captura de la página Control de Gastos. Fuente: Elaboración propia

La página de control de ahorros permite a los usuarios gestionar sus metas financieras, agregando objetivos de ahorro, que se actualizarán automáticamente con el paso del tiempo para mostrar dinámicamente la cantidad mensual necesaria para poder cumplir el objetivo y la restante para finalizarlo. También cuenta con una sección de resumen, con datos como el gasto y ahorro total en el mes seleccionado, así como una media de ahorro de los meses anteriores y un gráfico que muestra los excedentes de meses previos.

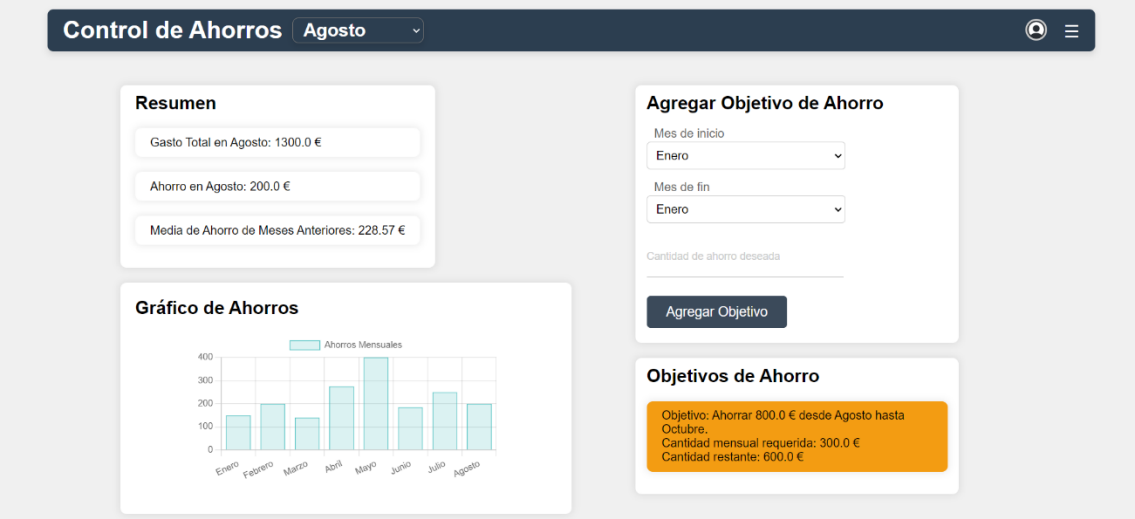


Figura 27: Captura de la página Control de Ahorros. Fuente: Elaboración propia

Por último, encontramos la página de perfil del usuario, en la que se puede observar la imagen de perfil que se haya seleccionado, pudiendo cambiarla por otra pulsando sobre ella, una

tabla que muestra los últimos movimientos, con su respectivo tipo, concepto, mes y cantidad, dependiendo de si el tipo es una ingreso o un gasto, el color de fondo de las celda será de color verde o rojo respectivamente y, finalmente, una sección en al que se muestran las credenciales, como el nombre o correo, y que permite cambiar la contraseña de una forma rápida y sencilla.

Figura 28: Captura de la página Perfil. Fuente: Elaboración propia

Cabe destacar que, para mejorar la experiencia del usuario y que sienta que está interactuando con la aplicación, se han implementado algunas animaciones a diferentes elementos, como los campos de los formularios para introducir datos, los botones de confirmación de dichos formularios y el selector del menú.

En el caso de los inputs, se muestra un texto, su label, en color gris dentro de ellos al estar inactivo, pero cuando el usuario pulsa sobre ellos pasan a estar activos y este texto se levanta y cambia de color a negro, dejando espacio para la información que se debe introducir, en este estado, el subrayado del campo se rellena de un color más intenso y grueso.

Figura 29: Diferencia entre input inactivo y activo. Fuente: Elaboración propia

En el caso de los botones de envío de los formularios, la animación es más simple, al pasar el ratón sobre ellos, estos se elevarán muy sutilmente, para dar una sensación de reactividad al usuario.

Figura 30: Diferencia entre botón inactivo y activo. Fuente: Elaboración propia

Finalmente, la animación del menú es la más simple de todas, ya que solo supone rotar 90º el símbolo que representa el menú.

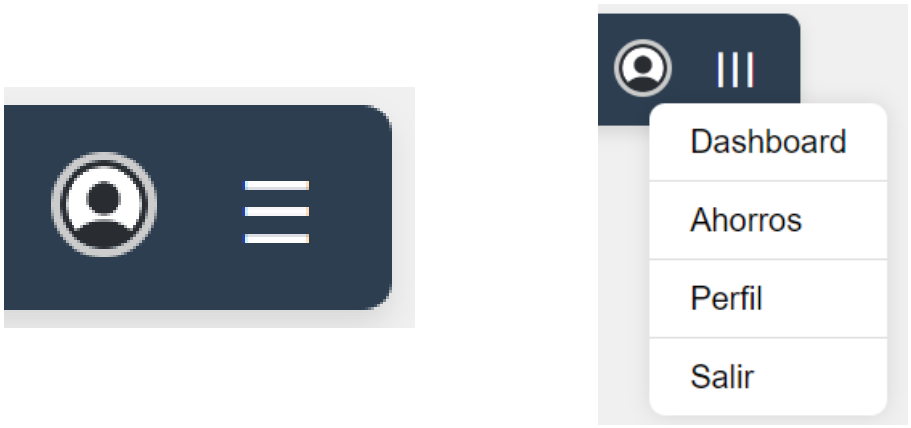


Figura 31: Diferencia menú inactivo y activo. Fuente: Elaboración propia

En resumen, la evaluación de los resultados muestra que la aplicación cumple con los objetivos propuestos, proporcionando una herramienta útil y eficiente para la gestión de gastos y ahorros personales. Combinando un diseño intuitivo, funcionalidades útiles y las medidas de seguridad adecuadas se asegura que la aplicación no solo es funcional sino también confiable y fácil de usar.

5. Manual de Usuario

Como ya se ha mencionado con anterioridad, lo primero que verá el usuario al intentar acceder a la aplicación será una ventana de inicio de sesión, en la que tendrá que ingresar sus credenciales para poder usar la aplicación.

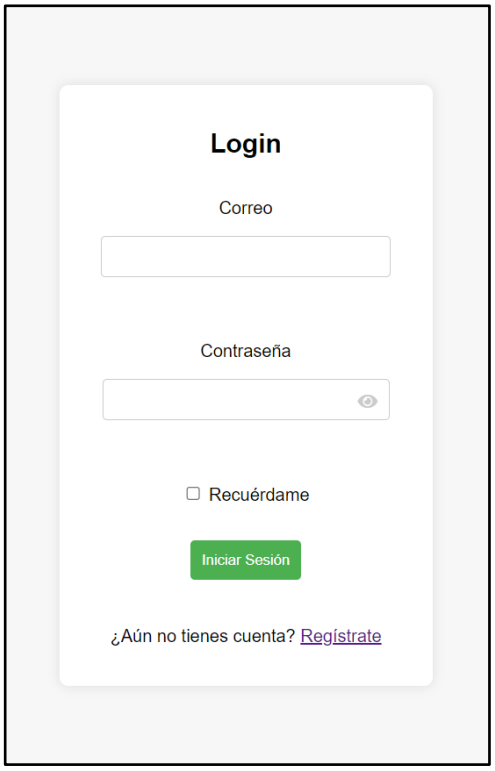


Figura 32: Formulario de inicio de sesión. Fuente: Elaboración propia

En caso de no disponer de dichas credenciales, el usuario deberá pulsar sobre el texto en el que pone “Regístrate”, al realizar dicha acción será redirigido a una página de registro en la que se encuentra un formulario que deberá ser cumplimentado con los datos correspondientes.

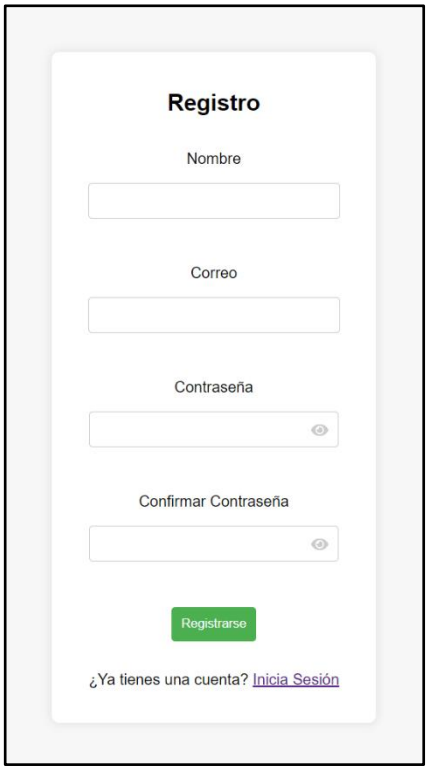


Figura 33: Formulario de registro. Fuente: Elaboración propia

Si no se introduce algún dato o este no cumple con los requisitos del sistema, se mostrará un aviso para que el usuario pueda hacerlo correctamente.

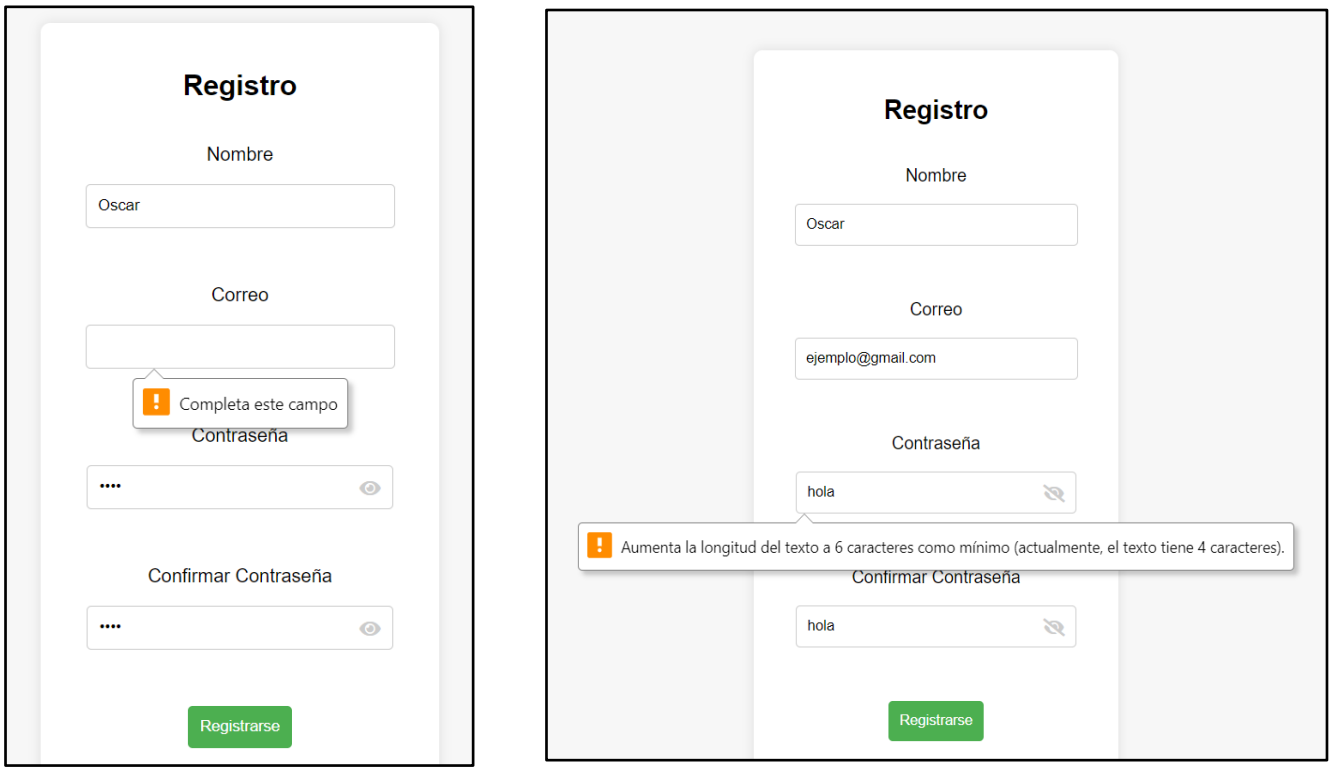


Figura 34: Ejemplos de errores en el registro de usuario. Fuente: Elaboración propia

Después de solucionar todos los errores y registrarse con éxito, será redirigido otra vez a la página de inicio de sesión, en la que se mostrará un mensaje indicándole que su cuenta se ha creado correctamente y que ya puede acceder a la aplicación.

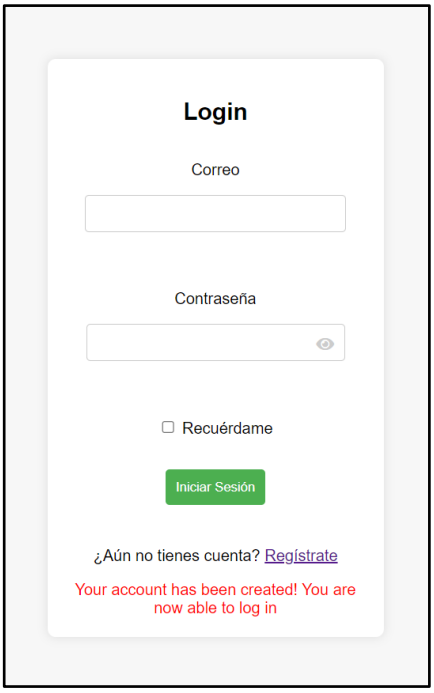


Figura 35: Mensaje de confirmación de la creación de la cuenta. Fuente: Elaboración propia

Tras iniciar sesión será llevado a la página de dashboard, en el que se muestra un resumen de sus ingresos y gastos totales, junto a un semáforo que sirve como medidor de situación económica, si este está de color verde no hay nada de lo que preocuparse (los gastos representan un 50% o menos de los ingresos), si está en ámbar hay que tener precaución (los gastos representan entre un 50% y un 80% de los ingresos) y si esta en rojo hay que tener cuidado (los gastos representan más de un 80% de los ingresos). También cuenta con dos gráficos que representan los ingresos, gastos y ahorros por meses, pudiendo interactuar con ellos, para ocultar o mostrar una categoría concreta con los botones superiores. Por último, cuenta con un ranking con los 3 mayores gastos y el porcentaje que estos representan sobre los ingresos totales.

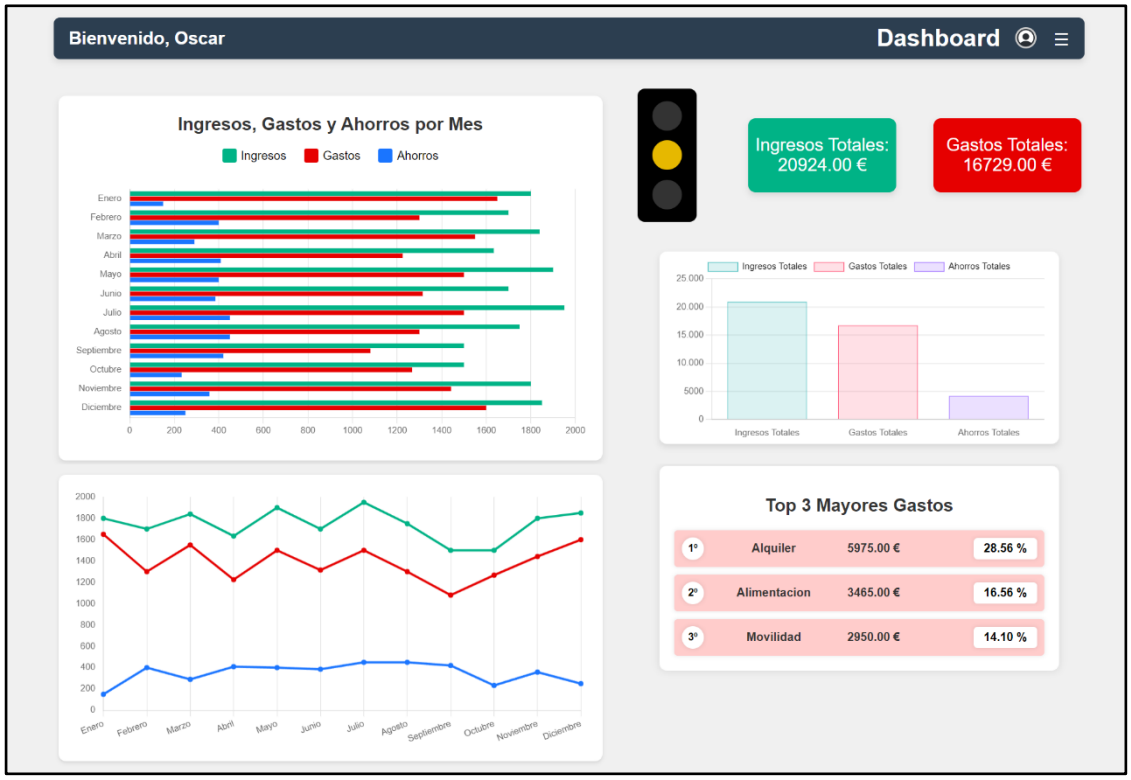


Figura 36: Captura de la página Dashboard. Fuente: Elaboración propia

Desde esta sección se puede entrar a las demás o cerrar sesión, pero si se trata de un nuevo usuario lo más lógico sería entrar a la página control de gastos para añadir algún ingreso, así como alguna expensa.

Al entrar a dicha sección, el usuario se encuentra ante 2 formularios, uno para añadir el importe del ingreso, así como el concepto asociado a él y otro para añadir el importe del gasto, la categoría a la que pertenece y el color que lo representa. Debajo de estos se encuentra un gráfico de anillo que muestra el porcentaje que representan sobre el ingreso y la cantidad restante, y a su lado se ubica una lista que contiene todos los gastos con su color correspondiente y un botón con una "X" que sirve para poder eliminar alguno en caso de haberse cometido un error, al pulsarlo se entrará en el modo de "eliminación", mientras de esté en este modo el botón cambiará su color de fondo para informar al usuario del estado en el que se encuentra y al pasar el ratón sobre un gasto, este se tachará y, si de pulsa sobre él, se eliminará, como se muestra en la figura 37.

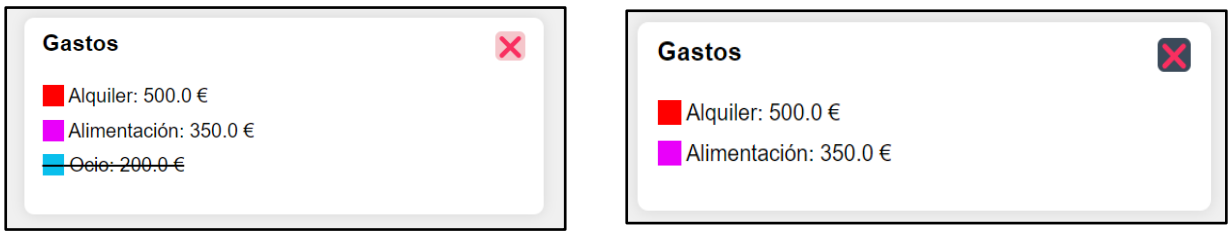


Figura 37: Diferencia entre el modo "eliminación" activo e inactivo. Fuente: Elaboración propia

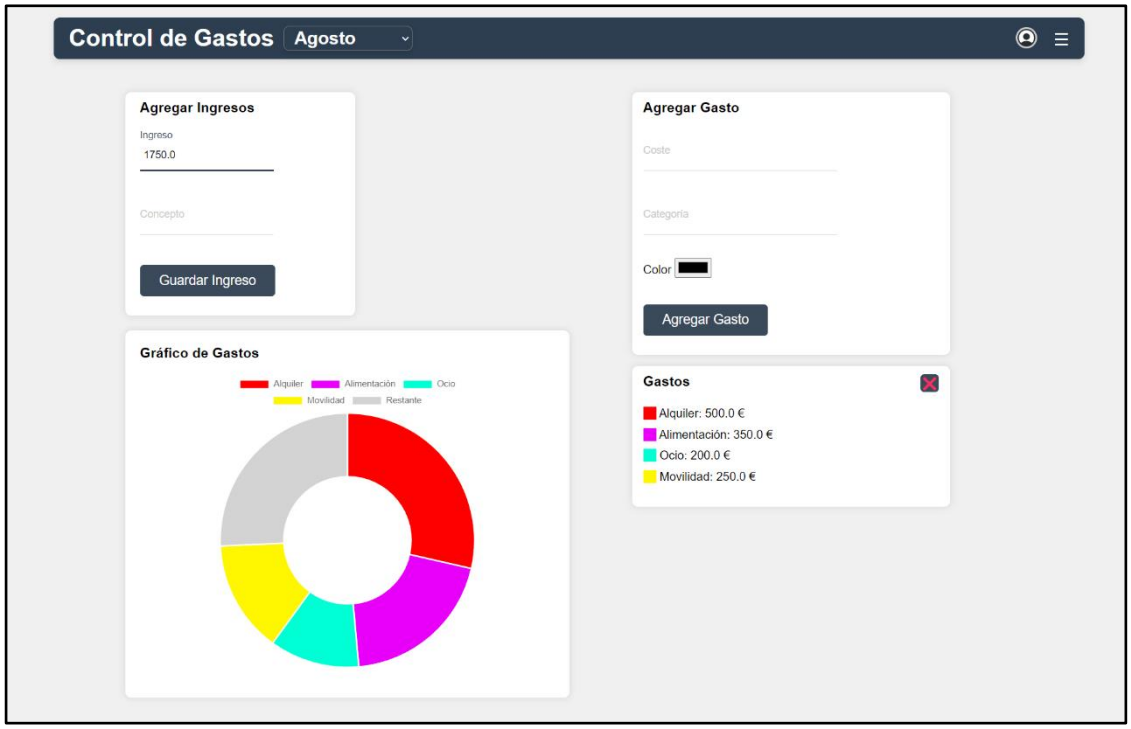


Figura 38: Captura de la página Control de Gastos. Fuente: Elaboración propia

Para cambiar el mes en el que se desean introducir datos solo hay que pulsar en el selector que se encuentra arriba a la izquierda, al lado del nombre de la página, y seleccionar uno de entre las opciones disponibles.

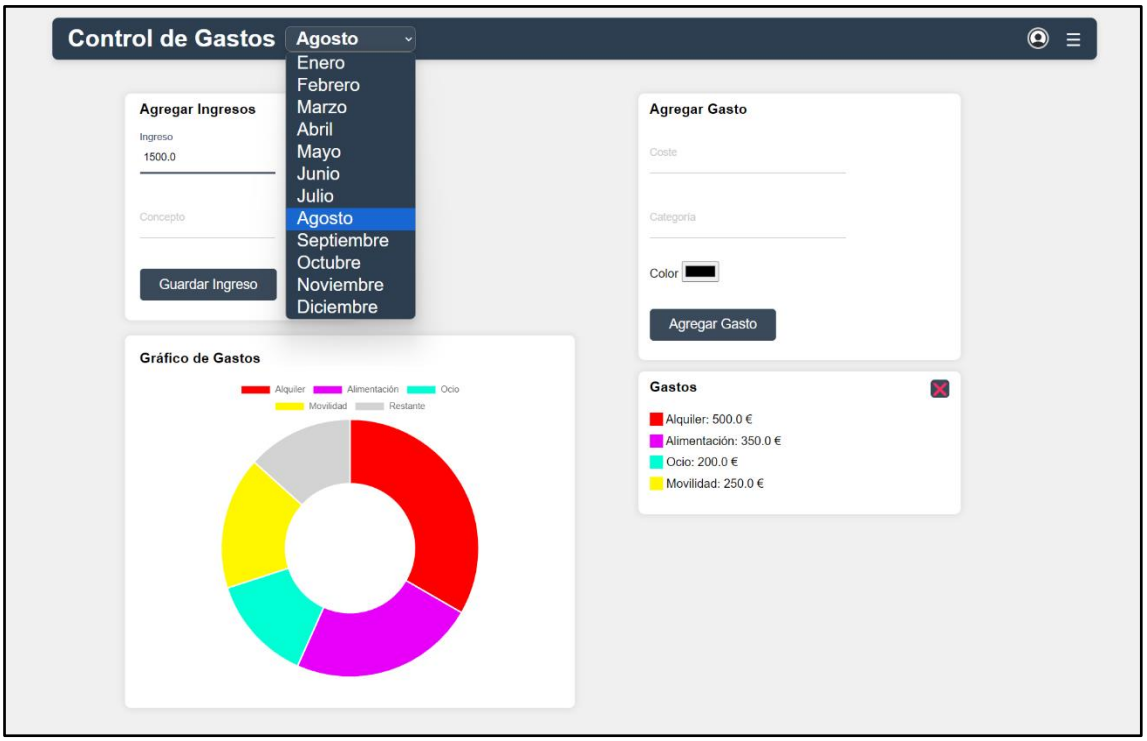


Figura 39: Captura del selector de mes. Fuente: Elaboración propia

Siguiendo con la página de ahorros, aquí se encuentran varias secciones diferenciadas, una consiste en un resumen del mes seleccionado, con el total gastado y ahorrado, y la media de ahorro de meses anteriores, debajo de este resumen se encuentra un gráfico de barras que muestra el ahorro durante los meses anteriores al actual, mientras que en la parte derecha es donde se ubica el formulario para añadir objetivos, teniendo que seleccionar el mes de inicio y fin, así como la cantidad deseada para cumplir la meta, debajo de dicho formulario se encuentra el listado de objetivos, que muestran la cantidad total de ahorro deseada, la cantidad restante para alcanzarla y la cantidad mensual necesaria para cumplir con el objetivo, dichos datos se actualizan dinámicamente dependiendo del mes seleccionado y la cantidad de ahorro conseguida.

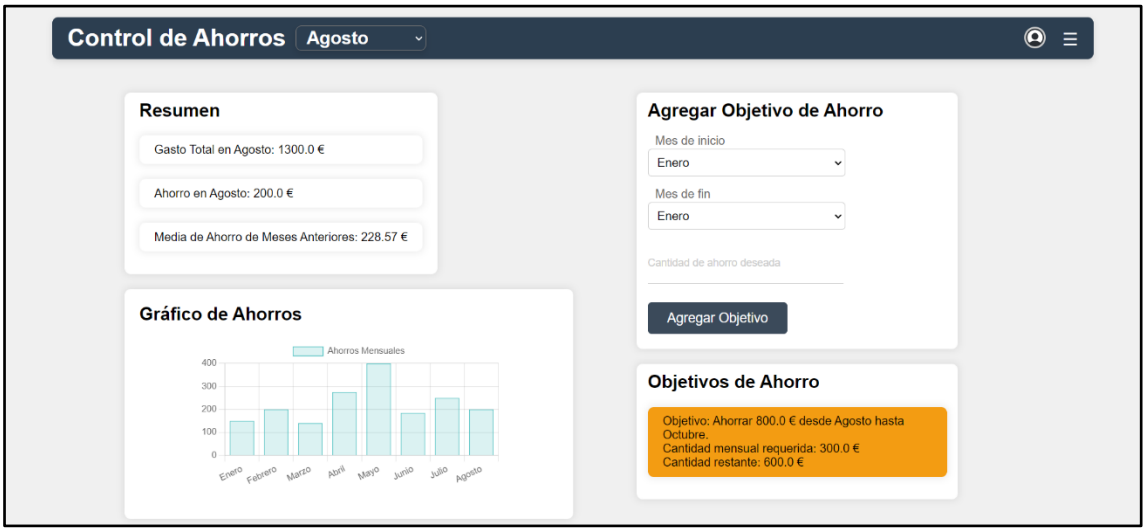


Figura 40: Captura de la página Control de Ahorros. Fuente: Elaboración propia

Finalmente, se encuentra la página de perfil, accesible desde todas las pestañas mencionadas anteriormente, desde el menú o directamente pulsando sobre la imagen de perfil que se encuentra a la izquierda del propio menú.

Figura 41: Imagen de perfil que lleva a la página Perfil. Fuente: Elaboración propia

Una vez dentro se observan varias secciones diferenciadas, la primera es la imagen de perfil, que puede ser editada por el usuario y utilizar una que se encuentre entre sus archivos (siempre que esta cuente con la extensión jpg o png), al cambiarla, esta no se mostrará solamente en el perfil, si no que también en las páginas anteriores, sustituyendo a la imagen por defecto anterior. La segunda sección, y puede que la que más destaque, es una tabla con los últimos 10 movimientos, con su tipo, concepto, mes y cantidad, dependiendo de si el tipo es un ingreso o un gasto las celdas tendrán un fondo de color verde o rojo respectivamente. La tercera, y última, sección es

la de credenciales, en las que el usuario ver información personal como su nombre o correo electrónico registrado, este último se encuentra censurado y hay que pulsar sobre el ojo situado al lado del campo para mostrarlo, otra opción es cambiar la contraseña haciendo click sobre el botón “Editar” que se encuentra al lado del campo correspondiente, tras pulsarlo se expandirá un formulario con varios campos a cumplimentar, siendo estos la contraseña actual, la nueva contraseña y la repetición de esta, tras guardar los cambios los valores se actualizarán en la base de datos y se harán efectivos.

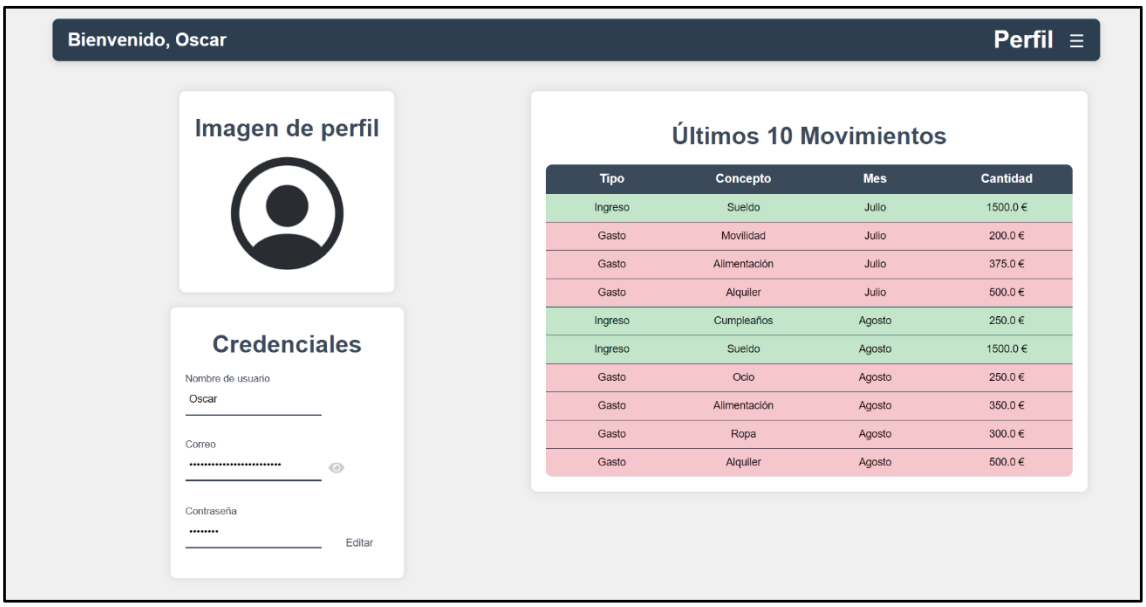


Figura 42: Captura de la página Perfil. Fuente: Elaboración propia

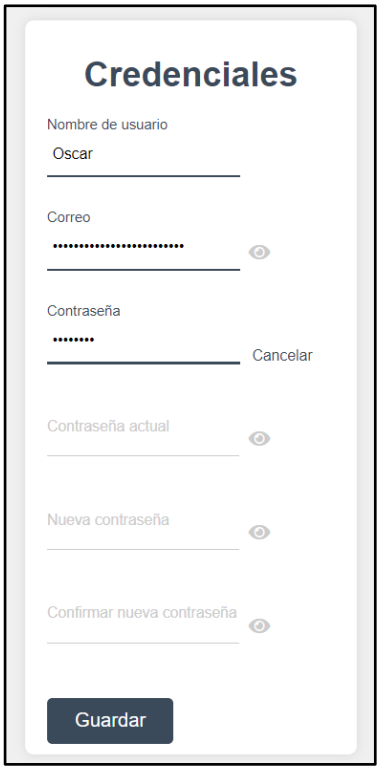


Figura 43: Formulario para el cambio de contraseña. Fuente: Elaboración propia

6. Conclusión

Este proyecto ha consistido en el diseño y desarrollo de una aplicación web para el control de gastos personales, que sea accesible y fácil de utilizar y promueva un uso eficiente de las finanzas de los usuarios. El desarrollo de la aplicación ha sido enriquecedor y desafiante y ha permitido aplicar y consolidar una amplia gama de conocimientos y habilidades en desarrollo web aprendidas durante el grado. Como se ha mencionado anteriormente, se han completado los objetivos propuestos, creando una herramienta funcional y de fácil acceso, esto ha sido posible gracias al seguimiento de una metodología estructurada y al análisis de los resultados.

Si bien la aplicación ha alcanzado los objetivos propuestos y un nivel de funcionalidad satisfactorio, siempre hay espacio para mejora y expansiones futuras. Algunas áreas en las que se podrías desarrollar nuevas o mejores funcionalidades serían:

- **Notificaciones y alertas:** Implementar un sistema de notificaciones para alertar a los usuarios sobre eventos importantes, como gastos inusuales o metas de ahorro alcanzadas.
- **Integración con otros servicios:** Permitir que los usuarios sincronicen sus cuentas bancarias u otros sistemas financieros para automatizar algunas funciones y ofrecer una gestión más completa.
- **Ampliación de funcionalidades:** Incluir funcionalidades adicionales como el análisis de tendencias de gasto o recomendaciones personalizadas.
- **Accesibilidad:** Actualmente la aplicación es accesible y funcional desde prácticamente cualquier dispositivo con acceso a internet, pero está diseñada para verse en un dispositivo con una pantalla de mediano-gran tamaño como la de un ordenador o una Tablet, sería positivo hacer que se ajustase mejor a pantallas de tamaño más pequeño y formato vertical como las de los teléfonos móviles.

En conclusión, este proyecto ha supuesto una experiencia sumamente gratificante y educativa. Se ha desarrollado una herramienta con el objetivo de ser de gran utilidad para cualquier persona que desee llevar un control más riguroso y eficiente de sus finanzas, con un cuidadoso y robusto diseño que hace posible cumplir con las demandas actuales y futuras de los usuarios.

7. Bibliografía

1000 Logos. (29 de Marzo de 2024). *CSS Logo*. Obtenido de https://1000logos.net/css-logo/

Amazon Web Services. (s.f.). *¿Qué es Python?* Obtenido de https://aws.amazon.com/es/what-is/python/

Amazon Web Services. (s.f.). *¿Qué es una API RESTful?* Obtenido de https://aws.amazon.com/es/what-is/restful-api/

Balsamiq. (s.f.). *Balsamiq Brand Assets*. Obtenido de https://balsamiq.com/company/brandassets/

Balsamiq. (s.f.). *Introduction to Balsamiq Wireframes for Desktop*. Obtenido de https://balsamiq.com/wireframes/desktop/docs/intro/

Branditechture. (20 de Junio de 2022). *Google Developer Logo*. Obtenido de https://branditechture.agency/brand-logos/download/google-developers/

Chrome for Developers. (s.f.). *Chrome DevTools*. Obtenido de https://developer.chrome.com/docs/devtools?hl=es-419

Coppola, M. (12 de Julio de 2023). *Qué es JavaScript, para qué sirve y como funciona*. Obtenido de https://blog.hubspot.es/website/que-es-javascript

Flores, F. (22 de Julio de 2022). *Qué es Visual Studio Code y qué ventajas ofrece*. Obtenido de https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/

Hernández, U. (22 de Febrero de 2015). *MVC (Model, View, Controller) explicado*. Obtenido de https://codigofacilito.com/articulos/mvc-model-view-controller-explicado

IONOS, E. E. (27 de Junio de 2023). *SQLite: la famosa biblioteca en detalle*. Obtenido de https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/sqlite/

Microsoft. (s.f.). *Microsoft Visual Studio*. Obtenido de https://visualstudio.microsoft.com/es/

Nikotaf. (4 de Mayo de 2016). *Javascript badge.svg*. Obtenido de https://commons.wikimedia.org/wiki/File:Javascript_badge.svg

Python Software Foundations. (s.f.). *El tutorial de Python*. Obtenido de https://docs.python.org/es/3/tutorial/

Sanwo, S. (s.f.). *How to Set Up a Virtual Environment in Python - And Why It's Useful*. Obtenido de https://www.freecodecamp.org/news/how-to-setup-virtual-environments-in-python/

seeklogo. (s.f.). *Flask Logo PNG Vector*. Obtenido de https://seeklogo.com/vector-logo/273085/flask

smartbrand. (s.f.). *Balsamiq*. Obtenido de https://www.sb.digital/diccionario-ux/balsamiq

Toews, M. (3 de Octubre de 2010). *SQLite370.svg*. Obtenido de https://commons.wikimedia.org/wiki/File:SQLite370.svg

uxwing. (s.f.). *Visual Studio Code icon*. Obtenido de https://uxwing.com/visual-studio-code-icon/

W3Schools Online Web Tutorials. (s.f.). *CSS Introduction*. Obtenido de https://www.w3schools.com/css/css_intro.asp

W3Schools Online Web Tutorials. (s.f.). *HTML Introduction*. Obtenido de https://www.w3schools.com/html/html_intro.asp

W3Schools Online Web Tutorials. (s.f.). *JavaScript Tutorial*. Obtenido de https://blog.hubspot.es/website/que-es-javascript

Wikipedia, la enciclopedia libre. (17 de Mayo de 2017). *HTML5_logo_and_wordmark.svg*. Obtenido de https://es.m.wikipedia.org/wiki/Archivo:HTML5_logo_and_wordmark.svg#filelinks

Wikipedia, la enciclopedia libre. (21 de Agosto de 2022). *Python-logo-notext.svg*. Obtenido de https://es.m.wikipedia.org/wiki/Archivo:Python-logo-notext.svg