udp UNIVERSIDAD DIEGO PORTALES

Documentación Técnica

Proyectos interdisciplinarios en TEC.

Integrantes:

- Jose Undurraga
- Alejandro Rivera
- Boris Contreras

Índice

1. Introducción

a. Contexto y Antecedentes del Proyecto

El proyecto "UDP Apuntes" tiene como objetivo crear un espacio colaborativo para estudiantes de la Universidad Diego Portales (UDP), facilitando el intercambio y acceso a recursos de estudio. Este sitio web, accesible solo para estudiantes mediante autenticación con Google OAuth, contará con funciones avanzadas de carga, búsqueda y visualización de documentos, respaldadas por un sistema de gestión de likes y dislikes para garantizar la relevancia y calidad del contenido. El diseño y funcionalidad de la plataforma se basan en las sugerencias y preocupaciones de alumnos y coordinadores de carrera, asegurando que se adapte a las necesidades reales de los usuarios.

b. Objetivo

El nuevo sistema busca adaptarse a las necesidades del equipo de diseño de la UDP para crear una plataforma colaborativa dirigida a los estudiantes de la Facultad de Humanidades e Historia. Permite compartir y acceder a materiales de estudio de manera sencilla y eficiente, utilizando una arquitectura cliente-servidor. La plataforma utiliza un sistema de autenticación con Google para garantizar que solo los estudiantes matriculados puedan acceder y contribuir. En lugar de un sistema de moderación centralizado, se confiará en los usuarios para gestionar el contenido mediante un sistema de likes y dislikes, promoviendo así la autorregulación y la colaboración entre estudiantes. Donde el coordinador de carrera, posteriormente podrá eliminar los archivos que tengan un contador de dislike mayor a sus likes.

c. Consideraciones de Tecnología

En la fase de desarrollo de "UDP Apuntes" inicialmente se planificó el uso de Angular con Node.js para construir la aplicación. Sin embargo, debido a una serie de cambios de último momento que surgieron tras la evaluación de la entrega dos, se decidió migrar a Flask como el framework principal. Esta decisión no solo buscaba mejorar aspectos críticos del proyecto como la integración y el rendimiento, sino que también se vio influenciada por la falla del equipo de diseño en entregar un prototipo funcional o utilizable del frontend a tiempo. La migración a Flask permitió una mayor flexibilidad y eficiencia en el desarrollo, facilitando la adaptación rápida a las necesidades del proyecto y garantizando una experiencia de usuario coherente y efectiva. Esta transición nos permitió cumplir con los plazos establecidos y mantener la calidad del producto final, a pesar de los desafíos imprevistos.

2. Arquitectura y patrón

a. Arquitectura de la Aplicación

La aplicación sigue una arquitectura cliente-servidor y se adhiere al patrón de diseño Facade para simplificar las interacciones complejas entre los componentes del sistema.

b. Arquitectura Cliente-Servidor

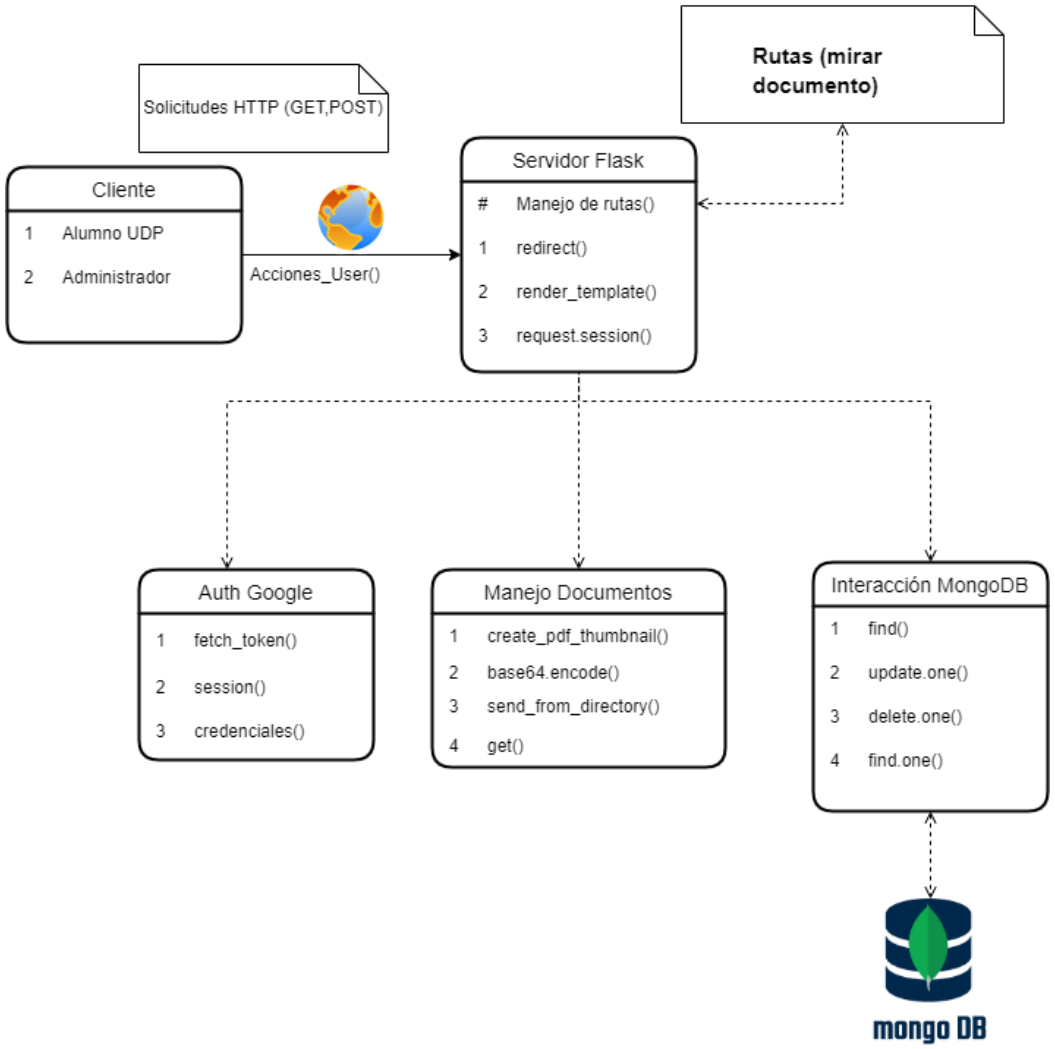
Cliente:

- Interfaz de usuario (HTML, CSS, JavaScript) accesible desde un navegador web.
- Envío de solicitudes HTTP (GET, POST) al servidor Flask.
- Recepción de respuestas HTTP para actualizar la interfaz de usuario.

Servidor:

- Aplicación Flask que maneja las solicitudes del cliente.
- Interacción con MongoDB para almacenar y recuperar datos.
- Autenticación con Google OAuth para manejar el inicio de sesión de usuarios.
- Generación de miniaturas de PDF y manejo de archivos en el sistema de archivos.

c. Patrón Fachada



El patrón Facade se implementa en la aplicación para proporcionar una interfaz simplificada para las interacciones del usuario y la administración de documentos. Este patrón ayuda a ocultar la complejidad del sistema subyacente y proporciona puntos de entrada claros para las operaciones principales.

Facade para Autenticación: El uso de Google OAuth 2.0 abstrae la complejidad de la autenticación y la gestión de sesiones.

Facade para Manejo de Documentos: Las rutas de Flask actúan como puntos de entrada simplificados para operaciones CRUD (Create, Read, Update, Delete) en los documentos.

Facade para Interacción con MongoDB: Las funciones y rutas de Flask proporcionan una interfaz clara para interactuar con la base de datos sin exponer detalles internos.

3. Componentes

a. Componentes de la Aplicación

Flask Application (app.py):

- Maneja las rutas y lógica de negocio.
- Configuración de OAuth y conexión a MongoDB.
- Implementación de decoradores para verificación de inicio de sesión y privilegios de administrador.

MongoDB:

- Almacenamiento de información de documentos y usuarios.
- Colección documents para almacenar metadatos de los documentos.
- Colección users para almacenar información de usuarios registrados.

Google OAuth:

- Proporciona autenticación segura para los usuarios.
- Verificación de usuarios administradores y manejo de sesiones.

Sistema de Archivos:

- Almacenamiento de archivos PDF subidos por los usuarios en la carpeta uploads.
- Almacenamiento de miniaturas generadas en la carpeta thumbnails.
- Estructura Cliente-Servidor

Rutas y Funcionalidades

- **/login:** Inicia el flujo de autenticación con Google OAuth.
- **/authorize:** Maneja la autorización y recuperación del token OAuth.
- **/logout:** Cierra la sesión del usuario.
- **/home:** Muestra los últimos 4 documentos subidos.
- **/upload:** Permite a los usuarios subir documentos PDF.
- **/thumbnails/<filename>:** Sirve las miniaturas de los documentos.
- **/buscar:** Permite a los usuarios buscar documentos por título.
- **/materias, /apuntes, /lecturas:** Muestran documentos filtrados por categoría.
- **/visualizador/<titulo>:** Muestra detalles y permite visualizar un documento específico.
- **/visualizador_A/<titulo>:** Vista administrativa de documentos.
- **/like/<titulo>, /dislike/<titulo>:** Manejan las acciones de like y dislike.
- **/review:** Muestra todos los documentos para la revisión del administrador.
- **/delete/<titulo>:** Permite al administrador eliminar documentos.

b. Interacciones y Flujo de Datos

i. Usuario Inicia Sesión:

- El usuario, a través de **/index.** accede a **`/login`**, que redirige a Google OAuth.
- Google OAuth autentica al usuario y redirige de vuelta a **`/authorize`**.
- La aplicación verifica el token y crea una sesión para el usuario.
- **`/authorize`**, redirige al usuario a **/home**

ii. Subida de Documento:

- El usuario sube un documento a través de **`/upload`**.
- La aplicación guarda el archivo en **`uploads`**, genera una miniatura y guarda la información en MongoDB.

iii. Visualización y Búsqueda:

- El usuario puede buscar documentos en **`/buscar`** o navegar por categorías (**`/materias`**, **`/apuntes`**, **`/lecturas`**).
- Al seleccionar un documento, se muestra en **`/visualizador/<titulo>`**.

iv. Acciones de Like y Dislike:

- El usuario puede dar like o dislike a un documento a través de **`/like/<titulo>`** o **`/dislike/<titulo>`**.
- La aplicación actualiza la base de datos y la sesión del usuario en consecuencia.

v. Revisión Administrativa:

- El administrador accede a **`/review`** para ver todos los documentos.
- Puede eliminar documentos a través de **`/delete/<titulo>`**.

4. Detalles de Implementación

a. Configuración del Entorno

i. Requisitos Previos

- Sistema Operativo: Ubuntu 20.04 (o superior)
- Python: Python 3.8 (o superior)
- Servidor Web: Nginx
- Servidor de Aplicaciones: Gunicorn
- Control de Versiones: Git
- Proveedor de Nube: Azure (recurso B4ms)
- Certificado SSL: Cloudflare

ii. Configuración Inicial del Servidor

1. **Actualizar el Sistema:**

   sudo apt update

   sudo apt upgrade

2. **Instalar Git:**

   sudo apt install git

3. **Clonar el Repositorio:**

   *git clone <URL-del-repositorio> **********

   cd <UDPapuntesBR>

4. **Instalar Python y pip:**

   sudo apt install python3 python3-pip python3-venv

5. **Configurar un Entorno Virtual:**

   python3 -m venv venv

   source venv/bin/activate

iii. Configuración del Proyecto

1. **Instalar Dependencias**

   pip install -r requirements.txt

2. **Configurar Variables de Entorno**

   FLASK_APP=app.py

   FLASK_ENV=production

   SECRET_KEY=<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>

   MONGO_URI=<mongodb-uri>

   GOOGLE_CLIENT_ID=4413767145-2temsr0uv0c2l5hqt68voi8cm8nergd2.apps.googleusercontent.com>

   GOOGLE_CLIENT_SECRET=<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>

iv. Configuración Nginx

1. **Crear un archivo de configuración para el sitio en `/etc/nginx/sites-available/`:**

```
server {
    listen 80;
    server_name <tu-dominio>;
    location / {
        proxy_pass http://127.0.0.1:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

2. **Habilitar el sitio y reiniciar Nginx:**
- sudo ln -s /etc/nginx/sites-available/<nombre-del-sitio> /etc/nginx/sites-enabled/
- sudo systemctl restart nginx

3. **Configurar Gunicorn:**
- sudo nano /etc/systemd/system/gunicorn.service
    - [Unit]
- Description=gunicorn daemon
- After=network.target
    - [Service]
- User=www-data
- Group=www-data
- WorkingDirectory=/home/azureuser/ptec/Proyecto-TEC
- ExecStart=/home/azureuser/ptec/Proyecto-TEC/venv/bin/gunicorn --workers 3 --bind unix:/home/azureuser/ptec/Proyecto-TEC/gunicorn.sock wsgi:app
    - [Install]
- WantedBy=multi-user.target

4. **Iniciar y habilitar Gunicorn:**
- sudo systemctl start gunicorn
- sudo systemctl enable gunicorn

b. Dependencias

A continuación, se presentan las dependencias clave y cómo instalarlas:

- **Flask:** Framework web ligero.
- **Flask-PyMongo:** Integración con MongoDB.
- **OAuthlib:** Manejo de autenticación OAuth.
- **gunicorn:** Servidor de aplicaciones WSGI.
- **cloudflare:** Gestión de DNS y SSL.

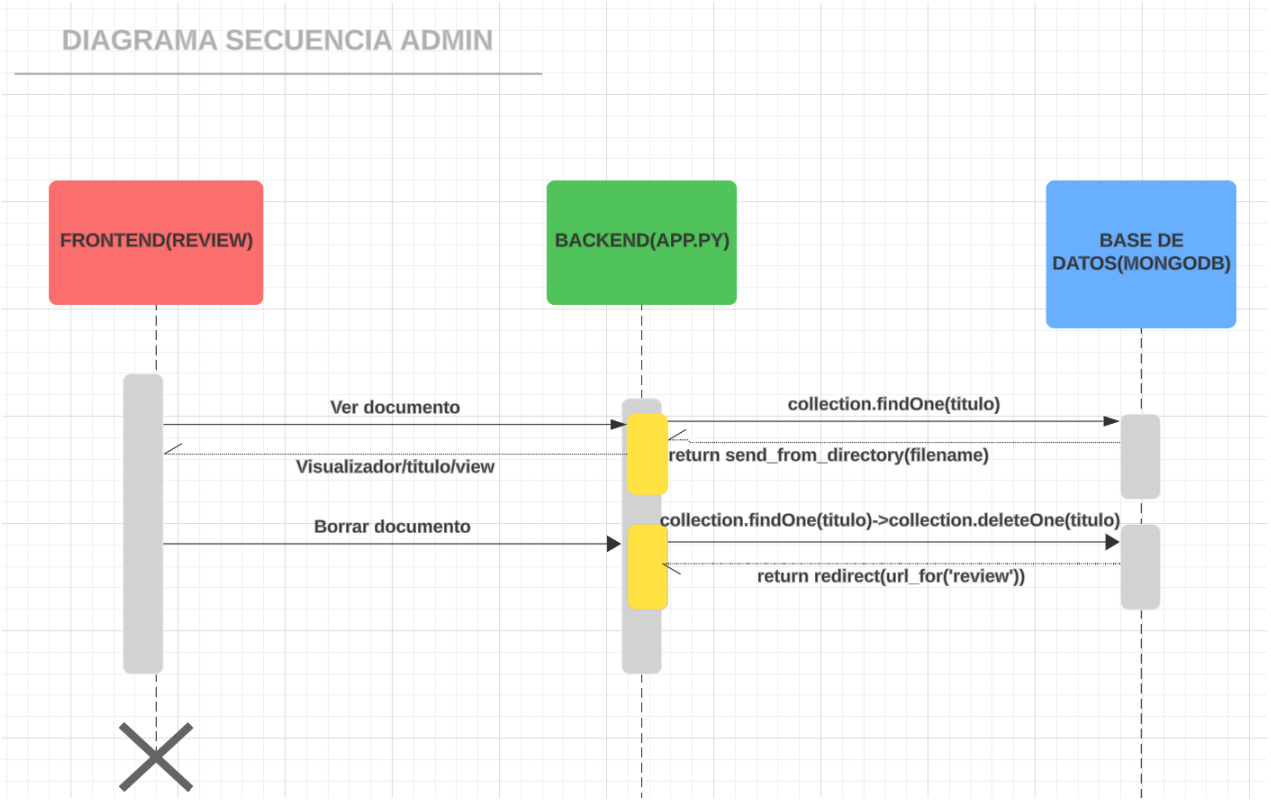
El archivo **`requirements.txt`** debe incluir:

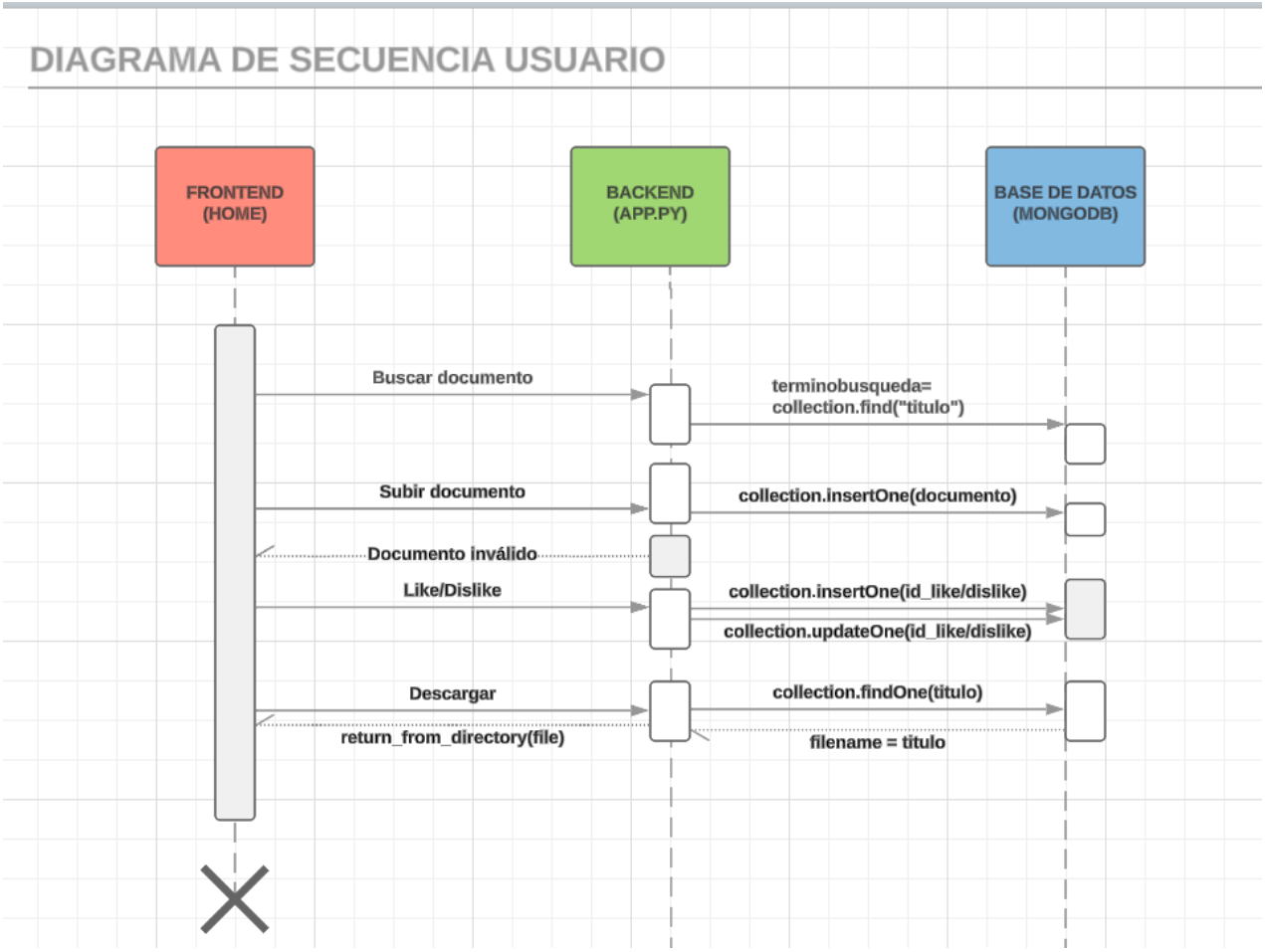
- Flask==2.0.2
- Flask-PyMongo==2.3.0
- oauthlib==3.1.0
- gunicorn==20.1.0
- cloudflare==2.8.15

c. Despliegue

- Para levantar el sitio, se utiliza:
    - gunicorn -b 0.0.0.0:8000 app:app

5. Documentación para el Usuario





a. Guía del Usuario

i. Acceso a la Plataforma

- Registro y Autenticación:
  - Accede a la plataforma mediante tu correo institucional de la Universidad Diego Portales.
  - En la página de inicio de sesión, haz clic en "Iniciar sesión con Google".
  - Ingresa tus credenciales de Google y otorga los permisos necesarios para acceder a tu información básica.

b. Navegación por la Plataforma

i. Página Principal:

- Últimos Documentos: En la página principal, encontrarás los últimos documentos subidos, organizados en una sección de fácil acceso.
- Menú de Navegación: Utiliza el menú de navegación para acceder a diferentes secciones del sitio, como subir documentos, ver todos los documentos, y tu perfil.

ii. Subir Documentos:

- Haz clic en el botón "Subir" en el menú de navegación.
- Rellena el formulario con los detalles del documento, incluyendo el título, descripción y categoría.
- Selecciona el archivo que deseas subir desde tu dispositivo.
- Haz clic en "Subir" para completar el proceso.

iii. Buscar Documentos:

- Usa la barra de búsqueda en la parte superior de la página para encontrar documentos específicos.
- Introduce palabras clave relacionadas con el título o el contenido del documento.

iv. Interacción con Documentos:

- **Ver Documentos:** Haz clic en el título de cualquier documento para ver más detalles y descargarlo.
- **Likes & Dislikes:** Puedes dar "Like" o "Dislike" a los documentos para indicar tu preferencia.
- **Comentarios:** En futuras versiones, se incluirá la opción de comentar sobre los documentos.

c. Preguntas Frecuentes (FAQ)

1. ¿Cómo puedo recuperar mi contraseña?
   1.1. La plataforma utiliza Google para la autenticación. Si tienes problemas para iniciar sesión, utiliza las opciones de recuperación de cuenta de Google.
2. ¿Qué tipos de archivos puedo subir?
   2.1. Actualmente, se aceptan archivos en formato PDF. En futuras versiones, se podrán subir otros tipos de archivos.
3. ¿Cómo puedo eliminar un documento que subí?
   3.1. Comunícate con tu coordinador de carrera para que este elimine el archivo.
4. ¿Puedo editar un documento después de subirlo?
   4.1. Actualmente, no es posible editar un documento. Si necesitas hacer cambios, Comunícate con tu coordinador de carrera para que este elimine el archivo, y sube una nueva versión.
5. ¿Qué sucede si encuentro contenido inapropiado?
   5.1. Utiliza la funcionalidad de dislike, si esto es contenido que no debería estar en el sitio, comunícate con tu coordinador de carrera para que este elimine el archivo.

6. Anexos

6.1. Glosario

- **API (Interfaz de Programación de Aplicaciones):** Conjunto de definiciones y protocolos que permiten que dos aplicaciones se comuniquen entre sí.
- **Azure:** Plataforma de servicios en la nube creada por Microsoft para construir, probar, desplegar y gestionar aplicaciones y servicios.

- **Cloudflare:** Servicio que proporciona CDN (Red de Distribución de Contenidos) y protección DDoS, mejorando la seguridad y el rendimiento de sitios web.
- **Gunicorn:** Servidor WSGI para aplicaciones Python, que se utiliza para desplegar aplicaciones web basadas en Flask.
- **HTTPS (Protocolo Seguro de Transferencia de Hipertexto):** Protocolo de comunicación en internet que protege la integridad y confidencialidad de los datos entre el usuario y el sitio web.
- **Nginx:** Servidor web de código abierto que también se utiliza como proxy inverso, balanceador de carga y caché HTTP.
- **OAuth:** Estándar abierto para autorización, que permite el acceso seguro a los recursos de un usuario sin compartir las credenciales.
- **pytest:** Marco de pruebas para aplicaciones Python que permite escribir pruebas simples pero escalables.