Descripción del Proyecto

El proyecto Aula Virtual es una aplicación web desarrollada con Python (Flask), MySQL, HTML, CSS y JavaScript, que simula un entorno educativo digital.
Permite la gestión de cursos, la autenticación de usuarios (alumnos y profesores), la publicación de contenidos y la interacción entre usuarios.

Objetivos principales:

- Proveer un espacio donde profesores puedan crear cursos y publicar materiales.
- Permitir que alumnos se inscriban mediante un código y participen en la clase.
- Garantizar seguridad y control de accesos mediante roles (Alumno, Profe).

Funcionalidades implementadas

Registro y login (signup y login) con roles

- Un usuario puede registrarse como Alumno o Profe desde el formulario signup.html.
- Validación de email con código de verificación enviado por correo.
- Sistema de login seguro con flask_login y contraseñas encriptadas.

Gestión de cursos

- El profesor puede crear cursos desde el panel principal (Index.html), cada curso tiene un código único.
- Los cursos se guardan en la tabla cursos y se asocian a su creador en cursos_usuarios.

Inscripción mediante código

- Los alumnos pueden unirse a un curso ingresando el código.
- Se registra la relación en la tabla cursos_usuarios.

Publicación de contenidos (posts)

- El profesor puede publicar posts (texto y archivo) en la página del curso.
- Los alumnos también pueden publicar, pero el profesor puede eliminarlos.

Control y moderación

- El profesor puede borrar un post de un alumno.
- El profesor puede expulsar a un alumno del curso (eliminando su registro en cursos_usuarios).

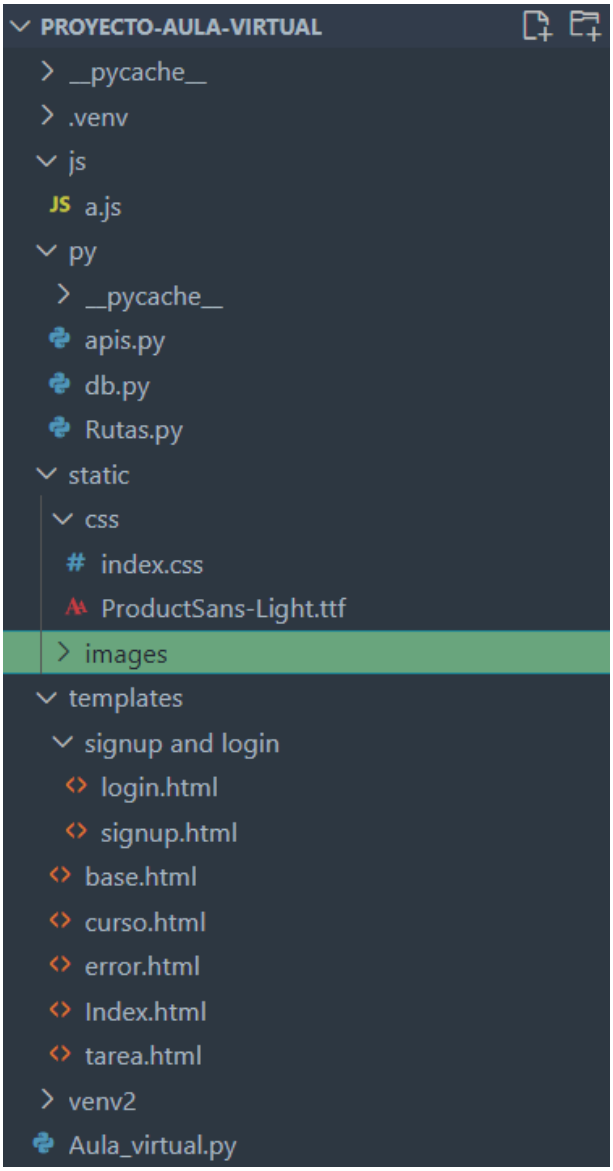
Gestión de tareas

- Los **profesores** pueden crear tareas dentro de un curso, con título, descripción y archivos adjuntos.
- Cada tarea queda vinculada al curso en la tabla Tarea.
- Los **alumnos** pueden realizar una única entrega por tarea:
    - Pueden adjuntar archivos (imágenes, documentos, videos).
    - La entrega se guarda en la tabla entrega y sus archivos en archivos.
- Un alumno puede cancelar su entrega para volver a subir otra (similar a Google Classroom).
- Los **profesores** pueden ver el listado completo de todas las entregas de una tarea:
    - Se muestra el nombre del alumno, fecha de entrega y archivos enviados.
    - Esto permite la revisión individual de cada entrega.
- Los archivos de cada entrega pueden visualizarse (si son imágenes/videos) o descargarse.

Cierre de sesión seguro

- Implementado mediante flask_login.logout_user().

Estructura de carpetas



4. Base de datos

Tablas principales:

- usuario → almacena nombre, email, contraseña encriptada, rango (Alumno o Profe).
- cursos → id, nombre, código único.
- cursos_usuarios → relación entre curso y usuario.
- posts → publicaciones de texto y archivos en cursos.
- Tarea → tareas creadas por profesores.
- entrega → entregas de alumnos a tareas.
- archivos → contenido de archivos subidos en posts o entregas.
- comentario → comentarios en posts.

Relaciones clave:

- usuario (1:N) cursos_usuarios (N:1) cursos
- cursos (1:N) posts
- cursos (1:N) Tarea
- Tarea (1:N) entrega (1:N) archivos.

Instrucciones para correr el proyecto

Clonar el proyecto y entrar al directorio:
git clone <URL_REPOSITORIO>
cd Proyecto-Aula-Virtual

Instalar dependencias:
pip install -r requirements.txt

Configurar la base de datos:

Crear la BD y tablas con:
mysql -u root -p < db_aula.sql

Configurar la app Flask:

En Aula_virtual.py, asegurarse de la URI:
app.config['SQLALCHEMY_DATABASE_URI'] =
'mysql+pymysql://root:<password>@localhost/aula'
Levantar el servidor:
python Aula_virtual.py

Acceder en: http://127.0.0.1:5000/.

Flujo de uso probado

- **Alumno:**
  Signup → login → unirse a curso con código → enviar post con imagen → realizar entrega de una tarea → cerrar sesión.

- **Profesor:**
  Signup → login → crear curso → publicar post → crear tarea → revisar entregas de alumnos → borrar post del alumno → expulsar alumno desde inscripciones.

Puntos pendientes / Mejoras futuras

- Implementar evaluaciones (tests/exámenes) con corrección automática.
- Añadir notificaciones.
- Mejorar diseño responsive.
- Añadir roles con más permisos (admin).
- Permitir calificación de entregas y feedback al alumno.