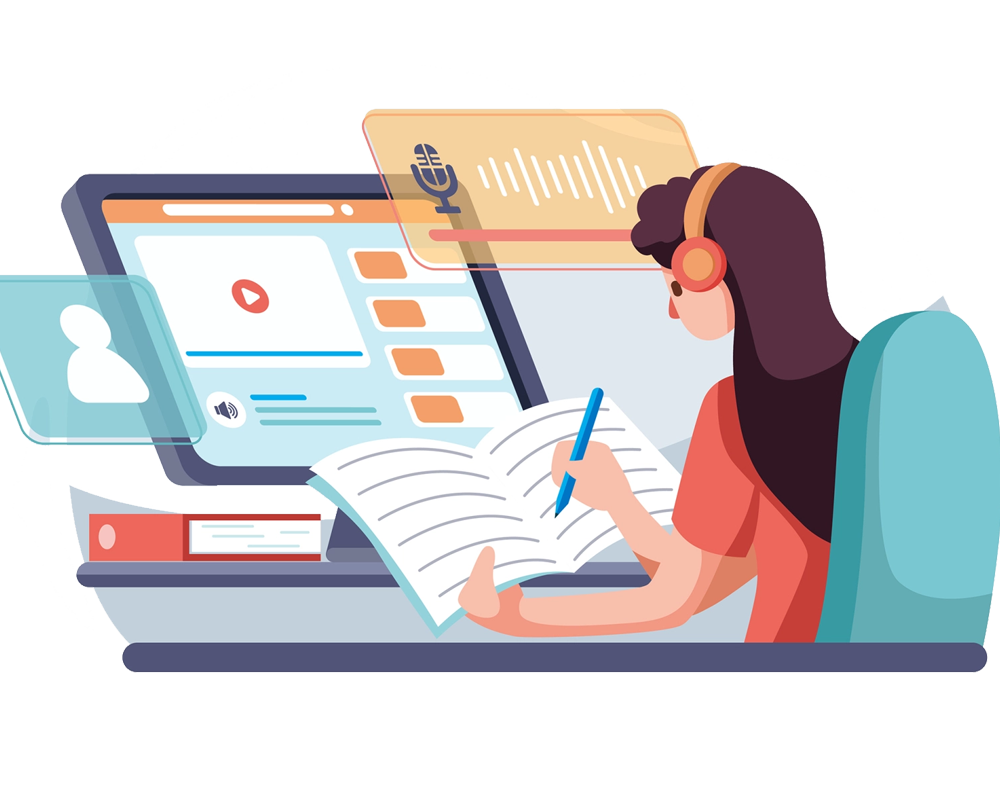
**TÍTULO DEL PROYECTO**

**SISTEMA DE GESTIÓN DE ALUMNOS Y NOTAS**

**INTEGRANTES:**

* **XXXXXXXXXX**
* **XXXXXXXXXX**
* **XXXXXXXXXX**

**1. Objetivos Principales**

El principal objetivo de este proyecto es desarrollar un sistema web que permita la gestión eficiente de los alumnos y sus calificaciones en una institución educativa.

Los objetivos específicos son:

* Permitir el **registro de alumnos**, almacenando sus datos personales de manera organizada.
* Facilitar la **carga y gestión de notas**, asegurando que cada estudiante tenga un historial académico accesible.
* Brindar una **interfaz intuitiva** que permita a los administradores consultar, editar y eliminar alumnos y sus notas.
* Automatizar el **cálculo de promedios** de cada alumno en función de sus calificaciones en distintas materias.
* Desarrollar un sistema escalable que pueda adaptarse a futuras necesidades.
* Implementar herramientas que optimicen el manejo de información y reduzcan la carga administrativa.

**2. Tecnologías Utilizadas**

El desarrollo de este proyecto se ha llevado a cabo utilizando diversas tecnologías modernas que garantizan un sistema robusto, eficiente y escalable. A continuación, se detallan las tecnologías utilizadas junto con sus definiciones y su aplicación en el proyecto:

**Lenguaje de Programación: Python (Framework Flask)**

* **Python** es un lenguaje de programación interpretado, de alto nivel y multipropósito. Su simplicidad y legibilidad lo hacen ideal para el desarrollo rápido de aplicaciones web.
* **Flask** es un framework ligero de Python diseñado para crear aplicaciones web. Permite manejar rutas, formularios, bases de datos y autenticación con facilidad.
* **Aplicación en el proyecto:** Se utilizó Flask para construir el backend del sistema, gestionando las solicitudes HTTP, las interacciones con la base de datos y la lógica de negocio.

**Base de Datos: SQLite**

* **SQLite** es un sistema de gestión de bases de datos relacional (RDBMS) ligero y sin servidor. Se almacena en un solo archivo, lo que lo hace ideal para aplicaciones pequeñas y medianas.
* **Aplicación en el proyecto:** Se utilizó SQLite para almacenar los datos de los alumnos y sus calificaciones de manera estructurada. La base de datos se gestiona a través de SQLAlchemy.

**Frontend: HTML, CSS, Bootstrap y JavaScript (jQuery para AJAX)**

* **HTML (HyperText Markup Language):** Es el lenguaje estándar para la estructura de las páginas web.
* **CSS (Cascading Style Sheets):** Se usa para el diseño y la presentación de las páginas web.
* **Bootstrap:** Es un framework CSS que facilita la creación de interfaces responsivas y modernas.
* **JavaScript:** Es un lenguaje de programación utilizado en el frontend para la interactividad de la aplicación.
* **jQuery:** Es una biblioteca de JavaScript que simplifica la manipulación del DOM y las solicitudes AJAX.
* **Aplicación en el proyecto:** Se usó HTML y CSS para la estructura y el diseño de las páginas, Bootstrap para mejorar la experiencia visual y la compatibilidad con dispositivos móviles, y JavaScript con jQuery para realizar solicitudes asíncronas y mejorar la interactividad.

**Backend: Flask con SQLAlchemy**

* **SQLAlchemy** es un ORM (Object-Relational Mapper) que permite interactuar con bases de datos en Python de manera sencilla y eficiente.
* **Aplicación en el proyecto:** Se utilizó SQLAlchemy para la creación, gestión y manipulación de la base de datos, permitiendo que las interacciones con SQLite sean más intuitivas y menos propensas a errores.

**Herramientas de Desarrollo**

* **Visual Studio Code:** Un editor de código ligero pero potente, utilizado para la escritura y depuración del código fuente del proyecto.
* **Draw.io:** Herramienta en línea para la creación de diagramas UML, como diagramas de clases y de flujo.
* **dbdiagram.io:** Plataforma para modelado de bases de datos, utilizada para diseñar la estructura de la base de datos antes de su implementación.
* **Aplicación en el proyecto:** Se usó Visual Studio Code para escribir el código, Draw.io para diseñar los diagramas del sistema y dbdiagram.io para estructurar la base de datos antes de codificarla.

**Entorno de Ejecución: Servidor local con Flask, WSGI para despliegue futuro**

* **WSGI (Web Server Gateway Interface):** Es el estándar de interfaz entre servidores web y aplicaciones web en Python.
* **Aplicación en el proyecto:** Actualmente, el sistema se ejecuta en un servidor local con Flask. Para el futuro despliegue, se podrá utilizar WSGI con servidores como Gunicorn o uWSGI para manejar múltiples solicitudes de usuarios simultáneamente.

**3. Fundamentación del Proyecto**

En el contexto educativo actual, la **digitalización de la información académica** se ha vuelto una necesidad imperante. El uso de registros físicos y planillas de Excel ha demostrado ser **ineficiente**, propenso a errores y de difícil mantenimiento. Este sistema surge como una **solución tecnológica** que permite la gestión centralizada de alumnos y sus notas, mejorando la organización y optimizando el tiempo de los docentes y administrativos.

**4. Necesidades que Originaron el Proyecto**

El proyecto se origina como respuesta a diversas problemáticas observadas en la gestión de información académica:

* **Dificultad en la organización** de registros de alumnos y sus calificaciones.
* **Errores humanos** en la manipulación de datos cuando se usan métodos manuales.
* **Falta de un sistema centralizado**, lo que dificulta el acceso a la información por parte de los administradores.
* **Pérdida de tiempo** en la búsqueda de información cuando no se dispone de una base de datos eficiente.
* **Imposibilidad de obtener estadísticas** y reportes automatizados sobre el rendimiento académico de los alumnos.

Este sistema busca resolver estos problemas mediante una **plataforma web accesible y eficiente**.

**5. Implementación del Proyecto**

El sistema se ha desarrollado en varias etapas:

1. **Análisis de Requisitos:** Se definieron los objetivos y necesidades del sistema.
2. **Diseño de la Base de Datos:** Se crearon las tablas alumnos y notas con su respectiva relación 1:N.
3. **Desarrollo del Backend:** Se implementaron rutas en Flask para manejar la lógica de negocio (CRUD de alumnos y notas).
4. **Desarrollo del Frontend:** Se diseñó una interfaz responsiva con Bootstrap.
5. **Integración y Pruebas:** Se verificó la funcionalidad completa del sistema, asegurando su correcto desempeño.
6. **Documentación:** Se elaboraron diagramas de clases, diagramas de flujo y documentación técnica.

DIAGRAMA DE CLASES

Diagrama

AI-generated content may be incorrect.

**Explicación del Diagrama de Clases**

**📌 Actor Principal: Administrador/Usuario del Sistema**

El **actor principal** en este sistema es el **administrador o usuario del sistema**, que puede:  
✔ Registrar alumnos con sus datos personales.  
✔ Asignar notas a los alumnos en diferentes materias.  
✔ Consultar y gestionar las notas de los alumnos.  
✔ Editar y eliminar tanto alumnos como notas.

**📊 Diagrama de Clases**

El sistema tiene **dos clases principales**:

1️ **Alumno** (Alumno):

* Representa a los estudiantes registrados en el sistema.
* Contiene atributos básicos como nombre, dni, email, domicilio y foto.

2️ **Nota** (Nota):

* Representa las calificaciones que recibe cada alumno en distintas materias.
* Cada nota está vinculada a un alumno a través de alumno\_id.

**📌 Relaciones del Sistema**

✔ **Relación Uno a Muchos (1:N)** → Un alumno puede tener **muchas notas**, pero cada nota pertenece a un solo alumno.

**🔍 Descripción del Sistema**

📌 **Objetivo**: Administrar alumnos y sus calificaciones en diferentes materias.

📌 **Funciones Principales**:  
✅ **Registrar Alumno** → Se crea un nuevo alumno con su nombre, DNI, email y foto.  
✅ **Cargar Notas** → Se asignan calificaciones a los alumnos en materias predefinidas.  
✅ **Mostrar Notas** → Se listan los alumnos junto con sus calificaciones y promedio.  
✅ **Editar Datos** → Se pueden modificar los datos del alumno y las notas.  
✅ **Eliminar Notas** → Se pueden borrar notas individuales o todas las notas de un alumno.  
✅ **Eliminar Alumno** → Se elimina un alumno completamente del sistema.

**📌 Resumen General**

* **El actor principal** es el **administrador/usuario** del sistema.
* **El sistema** permite gestionar alumnos y sus notas.
* **Cada alumno puede tener múltiples notas** en diferentes materias.
* **Se pueden realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar)** sobre alumnos y notas.

DIAGRAMA DE SECUENCIA

Diagrama

AI-generated content may be incorrect.

**Explicación del Diagrama de Secuencia**

El **diagrama de secuencia** representa la interacción entre los actores (usuarios) y los objetos del sistema a lo largo del tiempo. En este caso, muestra cómo el **usuario** interactúa con el **sistema**, y cómo el **sistema** consulta y manipula la **base de datos**.

**📊 Elementos del Diagrama**

1️ **Actor Principal:** Usuario (Administrador/Docente que usa el sistema).  
2️ **Objetos:**

* **Sistema** (interfaz web y lógica del backend).
* **Base de Datos** (almacena alumnos y sus notas).  
  3️ **Mensajes/Flujos:**
* Flechas representan **peticiones y respuestas** entre el usuario, el sistema y la base de datos.

**📌 Explicación del Flujo**

**1. Registro de Alumno**

📌 El **usuario** ingresa los datos del alumno en el formulario de registro.  
📌 El **sistema** procesa la solicitud y envía los datos a la **base de datos**.  
📌 La **base de datos** guarda la información y responde al sistema.  
📌 El **sistema** confirma que el alumno ha sido registrado con éxito.

**2. Carga de Notas**

📌 El **usuario** introduce el **DNI** del alumno para buscarlo.  
📌 El **sistema** consulta la **base de datos** y devuelve los datos del alumno.  
📌 El **usuario** ingresa las notas en distintas materias y las envía.  
📌 El **sistema** almacena las notas en la **base de datos** y muestra el **promedio** calculado.

**3. Consulta de Notas**

📌 El **usuario** selecciona un alumno para ver sus notas.  
📌 El **sistema** obtiene las calificaciones desde la **base de datos**.  
📌 El **sistema** muestra la lista de notas junto con el **promedio**.

**4. Edición/Eliminación de Notas**

📌 El **usuario** selecciona una nota y elige **editar o eliminar**.  
📌 El **sistema** actualiza o elimina la nota en la **base de datos**.  
📌 Se actualiza la lista de notas y se recalcula el **promedio**.

**5. Eliminación de Todas las Notas de un Alumno**

📌 El **usuario** solicita borrar todas las notas de un alumno.  
📌 El **sistema** ejecuta la acción en la **base de datos**.  
📌 Se confirma la eliminación y se muestra un mensaje de éxito.

**📌 Resumen**

El diagrama de secuencia muestra la **interacción dinámica** del usuario con el sistema para **gestionar alumnos y notas**, detallando los procesos clave de **registro, consulta, edición y eliminación**.

BASE DE DATOS

**Explicación de la Base de Datos (DB)**

El sistema usa una base de datos **SQLite** con **dos tablas principales**:

1️ **alumnos** → Almacena los datos personales de los alumnos.  
2️ **notas** → Registra las calificaciones de cada alumno en distintas materias.

🔹 **Relación entre las tablas**:

* alumnos y notas tienen una **relación 1:N (uno a muchos)**.
* Cada alumno puede tener **varias notas**, pero cada nota pertenece a **un solo alumno**.
* La conexión se realiza mediante la clave foránea alumno\_id en la tabla notas.

**📊 Estructura de la Base de Datos**

**📌 Tabla alumnos**

| **Campo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único del alumno |
| nombre | TEXT | Nombre completo del alumno |
| dni | TEXT (UNIQUE) | DNI del alumno (único) |
| email | TEXT (UNIQUE) | Correo electrónico del alumno |
| domicilio | TEXT | Dirección del alumno |
| foto | TEXT (Opcional) | Ruta de la foto del alumno |

**📌 Tabla notas**

| **Campo** | **Tipo de Dato** | **Descripción** |
| --- | --- | --- |
| id | INTEGER (PK) | Identificador único de la nota |
| alumno\_id | INTEGER (FK) | Referencia al ID del alumno |
| materia | TEXT | Nombre de la materia |
| nota | FLOAT | Calificación del alumno en la materia |

**📌 Relación entre Tablas**

* **alumnos.id** → **notas.alumno\_id** (Clave foránea).
* Esto significa que **un alumno puede tener muchas notas, pero cada nota pertenece a un solo alumno**.

Diagrama

AI-generated content may be incorrect.

**6. Resultados Obtenidos**

El sistema ha logrado:

* **Registrar alumnos de manera eficiente** con sus datos personales y fotografía.
* **Cargar y visualizar notas** de forma rápida y organizada.
* **Automatizar el cálculo de promedios**.
* **Facilitar la búsqueda de alumnos** mediante su DNI.
* **Permitir la edición y eliminación de registros** tanto de alumnos como de notas.
* **Brindar una interfaz amigable** que mejora la experiencia del usuario.

**7. Proyección a Futuro**

Este sistema tiene un gran potencial de mejora y expansión. Algunas futuras implementaciones incluyen:

* **Autenticación de usuarios** con diferentes roles (Administradores, Profesores, Alumnos).
* **Exportación de reportes en PDF y Excel**.
* **Notificaciones automáticas** para alumnos sobre sus calificaciones.
* **Integración con plataformas educativas** como Moodle o Google Classroom.
* **Despliegue en un servidor en la nube** para su acceso desde cualquier lugar.

**8. Conclusión**

Este proyecto representa un avance significativo en la digitalización de la gestión académica. Permite un manejo eficiente de los registros estudiantiles, **reduciendo errores y optimizando el tiempo** de los administradores. La escalabilidad del sistema lo hace ideal para ser implementado en **instituciones educativas de distintos niveles**.

En conclusión, esta plataforma no solo mejora la **organización y seguridad** de la información académica, sino que también abre la puerta a futuras mejoras tecnológicas en el ámbito educativo.