Base de Datos | Proyecto

Guillermo Acquistapace, Camilo Danielli, Diego Moreno

Decisiones de Implementación

En nuestro proyecto, tomamos varias decisiones importantes para que funcionara de la mejor manera posible.

- **Uso de Docker para la Contenerización:**

Usamos Docker para que sea fácil de ejecutar y replicar, sin importar desde qué computadora se ejecute. Docker nos deja aislar los componentes y manejar las distintas partes de forma efectiva. Esto fue hecho así porque queríamos evitar problemas de configuración en diferentes entornos y asegurar que todos los miembros del equipo trabajáramos en un mismo entorno.

- **Elección de MySQL como Base de Datos:**

Optamos por MySQL porque es robusto y tiene un amplio soporte en la comunidad. Necesitábamos un sistema confiable para manejar nuestras transacciones y relaciones entre tablas. Además, MySQL nos ofrece herramientas y funcionalidades que se ajustan bien a los requerimientos de nuestro proyecto.

- **Implementación del Backend con Flask:**

Decidimos utilizar Flask para construir nuestra API RESTful porque es un framework ligero y flexible que se adapta bien a proyectos como el nuestro. Lo hicimos de esta manera para facilitar la integración con la base de datos y el manejo de solicitudes HTTP. Flask nos permitió desarrollar rápidamente y mantener un código limpio y entendible.

- **Manejo de Codificación y Caracteres Especiales:**

Prestamos especial atención a la codificación de archivos y al manejo de caracteres como tildes y la letra "ñ". Esto fue crucial para evitar errores durante la ejecución de los scripts SQL y asegurar que los datos en español se representen correctamente. Nos encontramos con algunos problemas que resolvimos ajustando la codificación a UTF-8 y evitando el uso de caracteres especiales en los nombres de tablas y columnas.

- **Establecimiento de Restricciones en la Base de Datos:**

Implementamos restricciones y claves foráneas para mantener la integridad de los datos. Por ejemplo, establecimos que un instructor no puede dar dos clases en el mismo turno, asegurando así la consistencia de la información. También definimos restricciones para evitar que un alumno esté en dos clases al mismo tiempo.

Mejoras Implementadas o Consideradas en el Modelo de Datos

- **Normalización de Tablas:**

Nos aseguramos de que las tablas estuvieran normalizadas para eliminar redundancias y mejorar la eficiencia en las consultas. Esto incluye separar la información en tablas distintas y establecer relaciones claras entre ellas.

- **Implementación de Índices:**

Consideramos crear índices en campos clave para acelerar las búsquedas y optimizar el rendimiento de la base de datos. Esto es especialmente útil en campos que se utilizan frecuentemente en las consultas.

- **Validaciones Adicionales:**

Añadimos validaciones en el backend para controlar casos como evitar que un alumno esté inscrito en dos clases en el mismo turno, o que no pueda inscribirse en actividades para las que no cumple con la edad mínima requerida.

- **Optimización de Consultas SQL:**

Refinamos nuestras consultas para reducir la carga en la base de datos y mejorar los tiempos de respuesta de la aplicación. Por ejemplo, utilizamos joins eficientes y evitamos consultas innecesarias.

- **Manejo de Errores y Excepciones:**

Implementamos un manejo adecuado de errores y excepciones tanto en el backend como en la base de datos, para ofrecer mensajes claros al usuario y facilitar el proceso de depuración.

Bitácora del Trabajo Realizado

Durante el desarrollo del proyecto, seguimos varios pasos que nos guiaron en su construcción:

1. **Definición del Alcance y Requerimientos:**
    - Establecimos las funcionalidades principales que debía tener nuestra aplicación, como gestión de instructores, alumnos, clases y actividades.
    - Diseñamos el modelo entidad-relación para representar las relaciones entre las diferentes entidades y asegurar que cumpliera con los requisitos.
2. **Configuración del Entorno de Desarrollo:**
    - Configuramos Docker para contenerizar nuestra aplicación, permitiendo una fácil implementación y portabilidad.
    - Instalamos y configuramos MySQL como nuestra base de datos y creamos los scripts SQL necesarios.
3. **Desarrollo del Backend:**
    - Comenzamos a desarrollar el backend utilizando Flask, creando las rutas y la lógica necesaria para manejar las operaciones CRUD.
    - Integramos la base de datos con la aplicación, utilizando la librería pymysql para conectarnos a MySQL.

4. **Manejo de Errores y Depuración:**
    - Nos encontramos con problemas relacionados con la inicialización de la base de datos en Docker. Tuvimos errores donde las tablas no se creaban correctamente.
    - Trabajamos en resolver errores de codificación y caracteres especiales en los scripts SQL, especialmente con el uso de la letra "ñ" y caracteres acentuados.
    - Utilizamos herramientas como logs y mensajes de error para identificar y solucionar los problemas.
5. **Integración y Validación:**
    - Realizamos pruebas para asegurar que todas las funcionalidades trabajaran correctamente, probando cada una de las rutas y operaciones.
    - Validamos que las restricciones y validaciones estuvieran funcionando como se esperaba, por ejemplo, que no se pudiera asignar un instructor a dos clases en el mismo turno.
6. **Implementación de Funcionalidades Adicionales:**
    - Añadimos reportes y estadísticas para enriquecer la aplicación, como actividades que más ingresos generan o turnos con más clases dictadas.
    - Mejoramos la documentación y añadimos comentarios al código para facilitar su mantenimiento y comprensión.
7. **Preparación para la Entrega:**
    - Realizamos pruebas finales para asegurar la calidad del proyecto y corregimos los últimos errores.
    - Preparamos este informe y la documentación necesaria para la presentación.

Bibliografía

- **ChatGPT:**

Utilizamos ChatGPT especialmente en el proceso de Dockerización, y nos brindó comandos que nos sirvieron para resolver problemas con la inicialización de la base de datos y la configuración de los contenedores. Fue de gran ayuda para entender errores y encontrar soluciones rápidas.

- **Documentación de MySQL:**

Consultamos la documentación oficial para entender mejor la sintaxis y funcionalidades avanzadas.

- **Guía de Flask:**

Nos basamos en la guía oficial para estructurar nuestro backend y manejar las solicitudes HTTP.

- **Docker Documentation:**

Utilizamos la documentación de Docker para configurar nuestros contenedores y resolver problemas de despliegue.

- **Foros y Comunidades en Línea:**

Stack Overflow y otros foros fueron de gran ayuda para solucionar errores y comprender mejores prácticas.

- **Material de Clase:**

Referimos a las notas y recursos proporcionados por el profesor durante el curso.