Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Base de Datos 1, Sección N

Estudiantes:

William Alejandro Borrayo Alarcón – 201909103

José Daniel Alvarado Fajardo – 201904061

Manual técnico de Proyecto Final

Descripción de la solución:

Para la parte del cliente se utilizó React como Framework, se trabajaron los módulos de maestro, alumno y administrador con sus respectivas funciones, para el servidor se utilizó NodeJS, por medio de peticiones http se realizan consultas inserciones y cambios en la base de datos trabajada en MySQL, para poder subir archivos desde el cliente y que sean almacenados en el servidor se utilizó el middleware Multer en NodeJS.

Software y Hardware utilizado

Sistema: Windows 11 Home Single Language v21H2

Tipo de sistema: Sistema operativo de 64 bits, procesador basado en x64

Procesador: 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz

Ram: 16.0 GB (15.7 GB usable)

Espacio disponible en disco: 80GB

React: v18.1.0

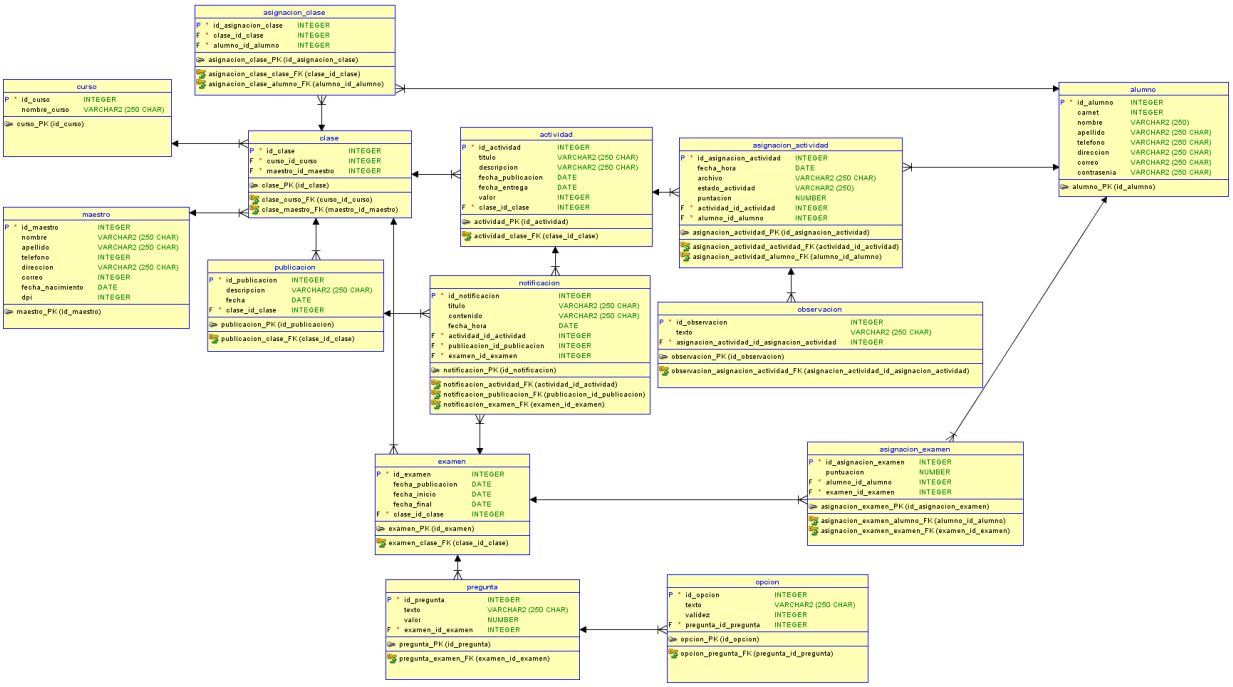
Node: v14.17.5

MySQL: Ver 8.0.27 for Win64 on x86_64 (MySQL Community Server - GPL)

DBeaver: v22.0.3.2

Microsoft Edge: Versión 101.0.1210.32

Molelo Entidad Relación:



Enpoints:

Tanto en la parte del frontend como en el backend, se utilizaron archivos distintos para separa cada uno de los módulos (Administrador, Maestro, Alumno, Login).

Administrador:

- cargaMasiva: Este endpoint sirve para realizar la carga masiva de un archivo csv con algún tipo de usuarios. El archivo se almacena en la carpeta temp del servidor, y desde ahí se convierte a formato JSON para que pueda ser iterado e insertar cada registro.

- crearUsuario: Este endpoint de tipo POST, recibe como parámetro todos los datos para la creación de un nuevo usuario (maestro, alumno), según el tipo.

- eliminarUsuario: Este endpoint es de tipo POST, recibe como parámetro el id del usuario que se desea eliminar y su tipo para indicar si es maestro o alumno.

- getAlumnos: Este endpoint es de tipo GET y se encarga de recibir a todos los alumnos disponibles en la base de datos.

- getCursos: Este endpoint de tipo GET, se encargar de recibir todos los cursos creados para la asignación de maestros.

- getMaestros: Este endpoint de tipo GET, se encargar de recibir todos los maestros de la base de datos para su asignación a cursos.

- verificarUsuario: Este endpoint de tipo POST, se encarga de verificar que el carnet y el correo de un usuario no esté ya en uso, porque este debe ser único.

Maestro:

- calificarEntrega: Este endpoint de tipo POST, recibe como parámetro los datos que corresponden a la entrega realizada por un estudiante y los nuevos datos como la puntuación y las observaciones ingresadas por el maestro.

- crearActividad: Este endpoint de tipo POST, recibe como parámetro todos los datos necesarios para que un maestro pueda crear una actividad para una clase en específico.

- createPublicacion: Este endpoint es de tipo POST, recibe como parámetro todos los datos necesarios para que un maestro pueda crear una publicación para una clase en específico.

- getEntregas: Este endpoint sirve para obtener todas las entregas que han hecho los alumnos a una actividad publicada por el Maestro.

- updatePublicacion: Este endpoint es de tipo POST, recibe como parámetro todos los datos necesarios para que un maestro pueda actualizar una publicación creada anteriormente.

Alumno:

- entregarActividad: Este endpoint de tipo POST sirve para realizar la entrega de una actividad, recibe como parámetro el id de la asignación a la actividad y el archivo subido. Actualiza en la base de datos el estado de la actividad de “Pendiente” a “Entregado” para que el maestro pueda hacer la calificación.

- getPublicacionesAlumno: Este endpoint de tipo POST, sirve para obtener todas las publicaciones que el alumno debe visualizar.

- getPublicacionesAlumno: Este endpoint de tipo POST, sirve para obtener todas las publicaciones que el alumno debe visualizar.

- getTotalAlumno: Este endpoint sirve para obtener el total de notas del alumno, calcula el total en base a las actividades calificadas como también de los exámenes realizados.

Login

- iniciarSesion: Este endpoint de tipo POST sirve para validar que las credenciales ingresadas por el usuario de Carnet-contraseña o Registro-contraseña sean correctas, verificando que existe al menos un registro con esos datos.

Procedimientos Almacenados:

- get_maestro: Este procedimiento sirve para obtener todos los datos de un maestro en específico, recibe como parámetro el id del maestro que se busca.

- get_publicaciones_maestro: Este procedimiento sirve para obtener todas las publicaciones que ha hecho un maestro.

- delete_publicacion: Este procedimiento sirve para eliminar una publicación realizada previamente por el maestro, recibe de parámetro el id de la publicación.

- insert_actividad: Este procedimiento sirve para insertar en la tabla actividad, la nueva actividad creada por el maestro.

- get_alumnos_curso: Este procedimiento sirve para obtener los alumnos que pertenecen a un curso impartido por el maestro.

- asignar_punteo_actividad: Este procedimiento sirve para colocar la nota a una actividad que ha sido calificada por el maestro.

- entregar_actividad: Este procedimiento sirve para actualizar el estado de una actividad asignada al alumno, de estado “Pendiente” a “Entregado”.

- get_notas_alumno: Este procedimiento sirve para obtener las notas de las actividades de un alumno. Para que pueden verse en el apartado de notas, recibe de parámetro el id del alumno y el id de la clase.

Triggers:

- notificar_publicacion: Este trigger sirve para crear una notificación cada vez que un catedrático crea una publicación.

- notificar_actividad: Este trigger sirve para crear una notificación cada vez que un catedrático crea una actividad.

- notificar_examen_actividad: Este trigger sirve para crear una notificación cada vez que un catedrático crea un examen.