Parcial 1 - Bases de datos I

Johan A. Ruiz B. Cod 201942434

Universidad del Valle
Cali
11 de enero del 2021

Índice

Introducción

Attendance es una herramienta de software que permite el registro de asistencia de estudiantes a clase, e igualmente se registran las asistencias del personal. El administrador puede crear un curso y agregar detalles del personal, tanto estudiantes como profesores.

Análisis de requerimientos

El desarrollo de la aplicación web inicia con el diseño conceptual del modelo de datos. Luego el desarrollo de una micro aplicación encargada de la Creación, lectura, actualización y eliminación de los datos denominada backend y finalmente la construcción de una interfaz gráfica de la aplicación.

Para este desarrollo, se ha decidido utilizar PostgreSQL, nodeJS + Express, Bent y ReactJS.

- PostgreSQL es una base de datos avanzada de código abierto, totalmente gratuita, que cumple bien con estándares SQL y su rendimiento es comparable con otros SGBD de uso comercial, este potente sistema de base de datos relacional con más de 30 años de desarrollo activo ha ganado una sólida reputación.

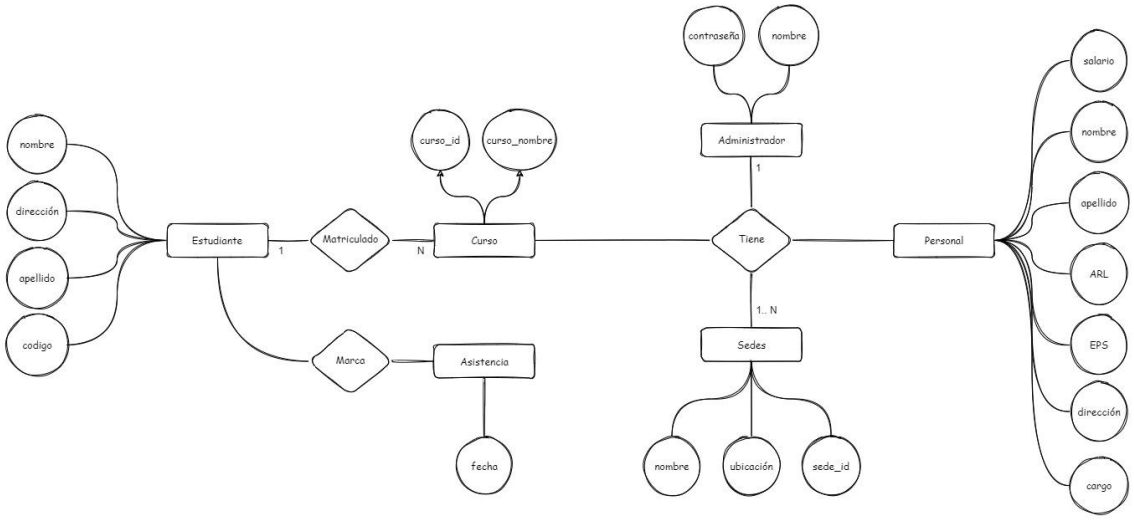
- Node + Express es un marco de aplicación web mínimo y flexible que proporciona un conjunto sólido de funciones para aplicaciones web y móviles. Escrito en JavaScript se aloja en el entorno de ejecución de Node.js.

- ReactJS es el marco de JavaScript front-end más popular. Los desarrolladores utilizan JSX, una combinación de HTML y JavaScript, para crear vistas de forma natural. Los desarrolladores también pueden crear componentes para bloques que se pueden reutilizar en sus aplicaciones.

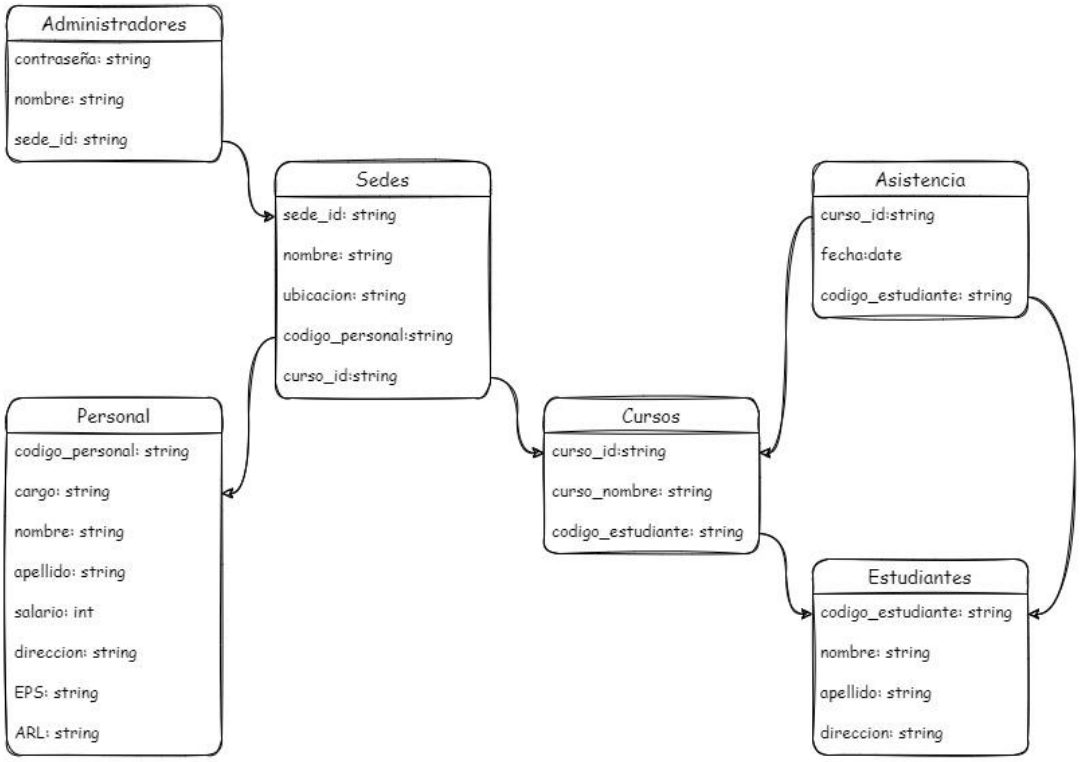
- Bent es un cliente HTTP funcional para Node.js y navegadores con async/await. Versión de navegador increíblemente pequeña basada en fetch sin dependencias externas ni polyfills.

Diseño

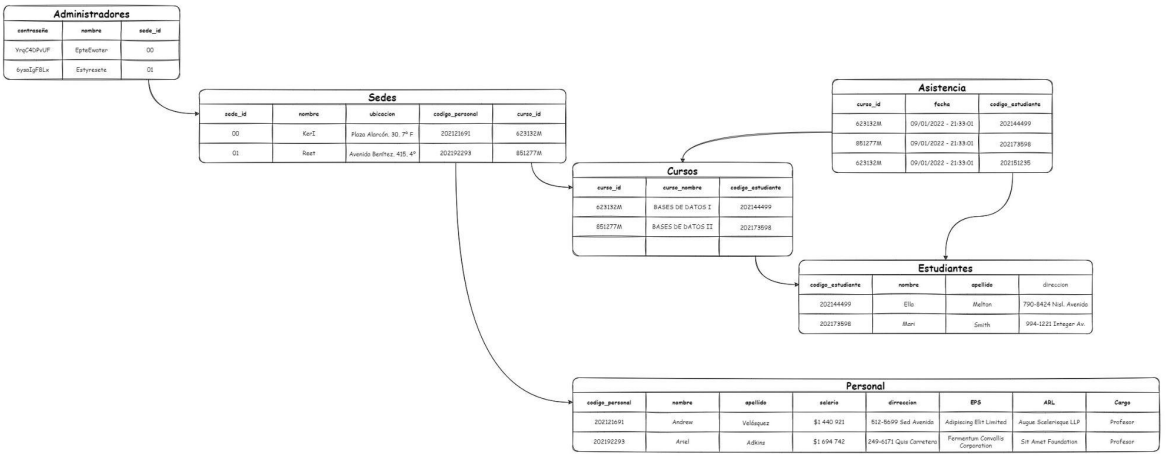
Para el diseño de la base de datos se intentó usar una estructura definida en el siguiente diagrama entidad relación a traves de Draw.io:



Aquí también se presenta el diagrama relacional:



Y adicionalmente, para el diseño de la base de datos se hicieron algunos ejemplos:



Para el diseño del servidor se usó una app generada por Exprees y posteriormente editada para cumplir con las especificaciones requeridas. El código fuente se puede encontrar en GitHub.

Para el diseño de la GUI se usó React, react-bootstrap, y bootstrap. El diseño que se tenía planeado era parecido al de la página inicial del campus virtual, es decir, así:

Desarrollo

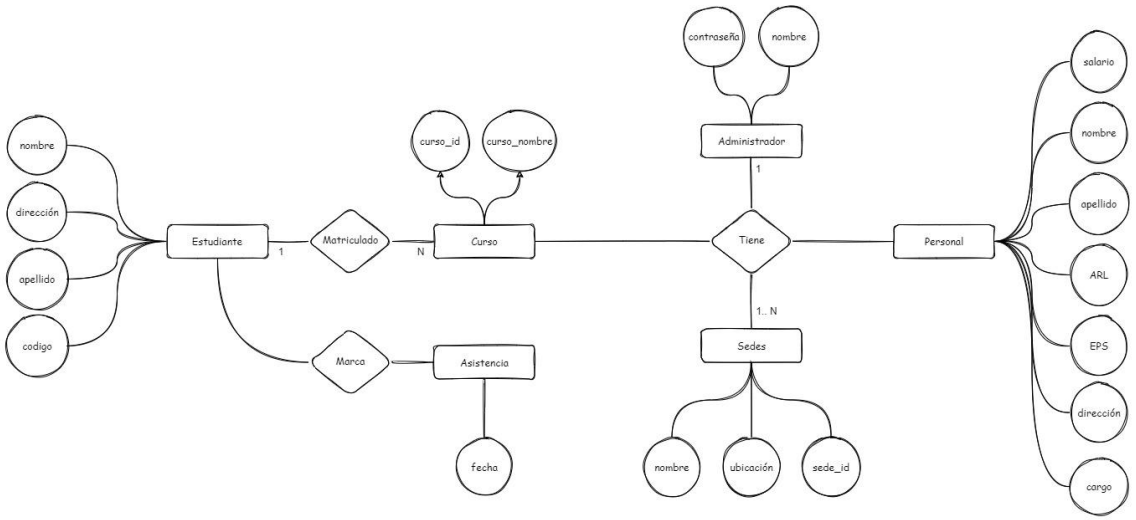
Capa de datos

Para el desarrollo e implementación de la base de datos, se usó postgres, más específicamente un docker de postgres con Docker Desktop. El archivo SQL contiene la información de las tablas que se extrajo del diseño hecho con Draw.io:

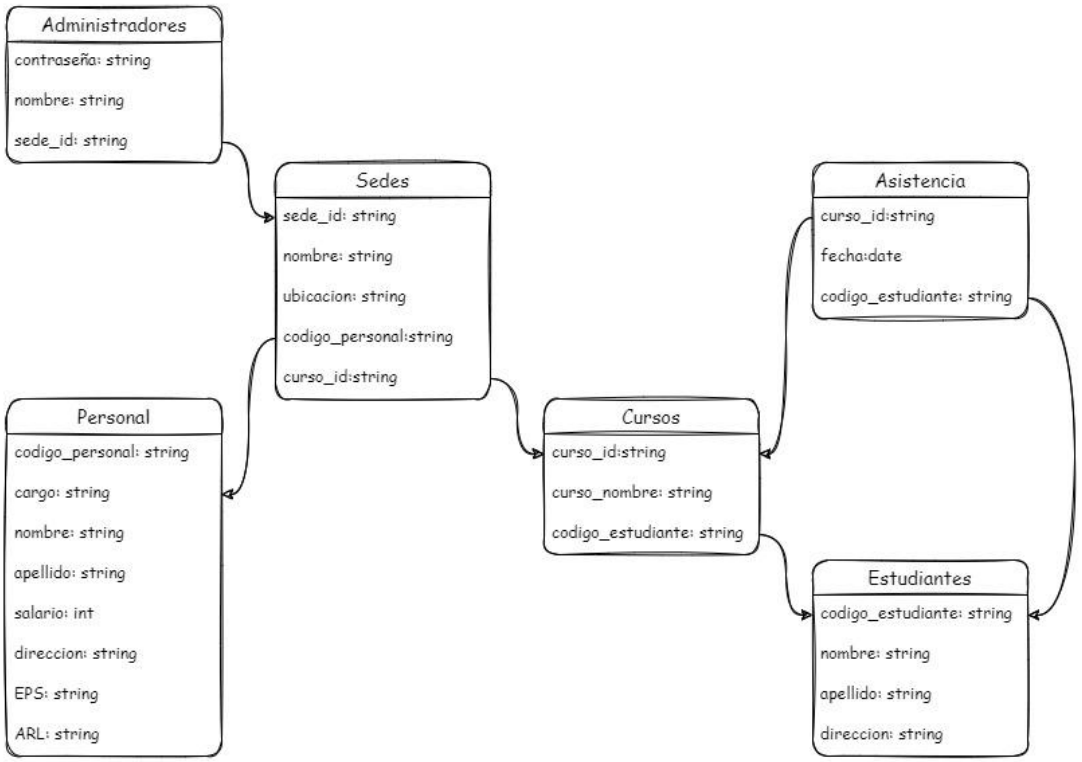
attendace.sql

Requerimientos

1. [15] Crear un diseño utilizando el modelo Entidad-Relación (ER) con notación Chen o Chen-Extendida:



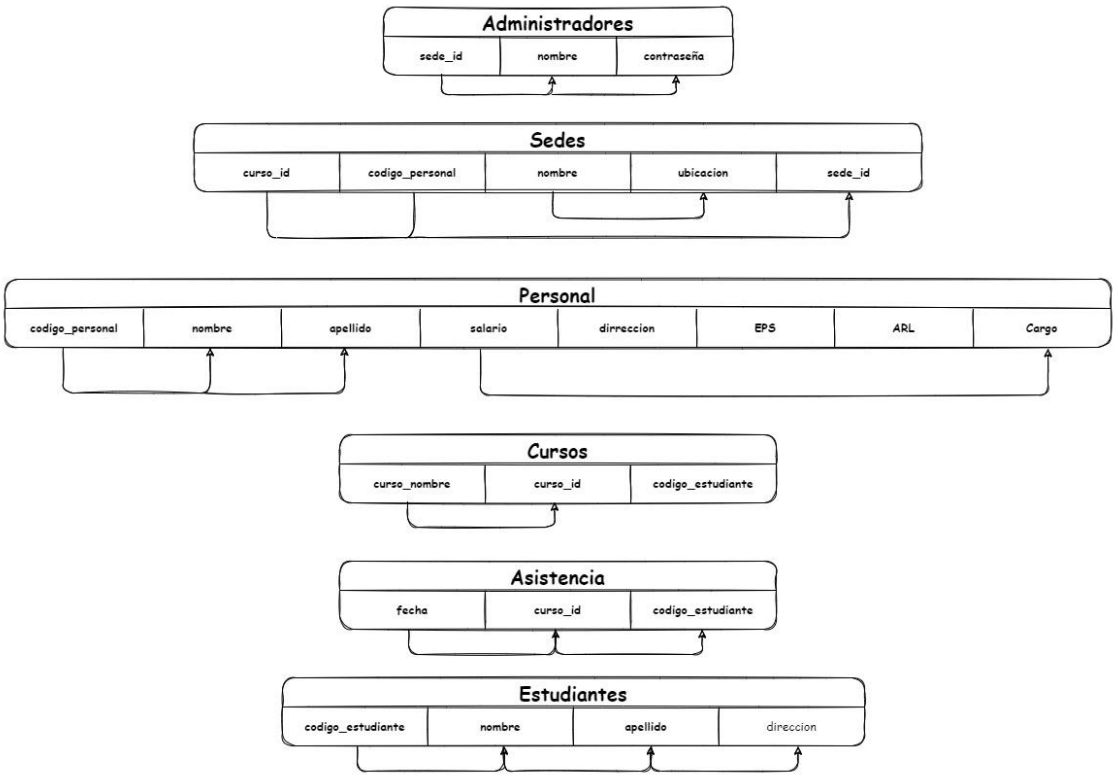
2. [10] Crear un diseño utilizando el modelo Relacional utilizando como base el diseño ER.



3. [10] Crear los archivos con las instrucciones SQL para llevar sus diseños a PostgreSQL

R:// Archivo que contiene las instrucciones SQL: attendace.sql

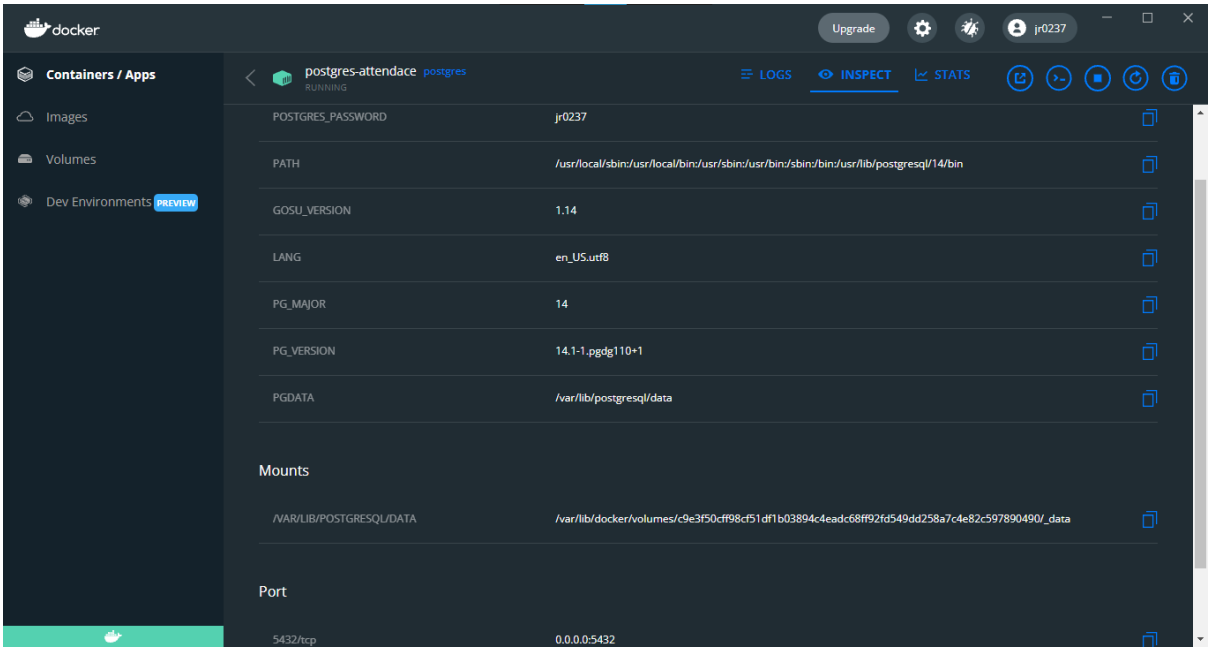
4. [10] Encuentre las dependencias funcionales de cada una de sus tablas.



5. [10] Describa por qué su diseño se encuentra normalizado.

R://

6. [5] Despliegue utilizando la versión contenerizada de PostgreSQL



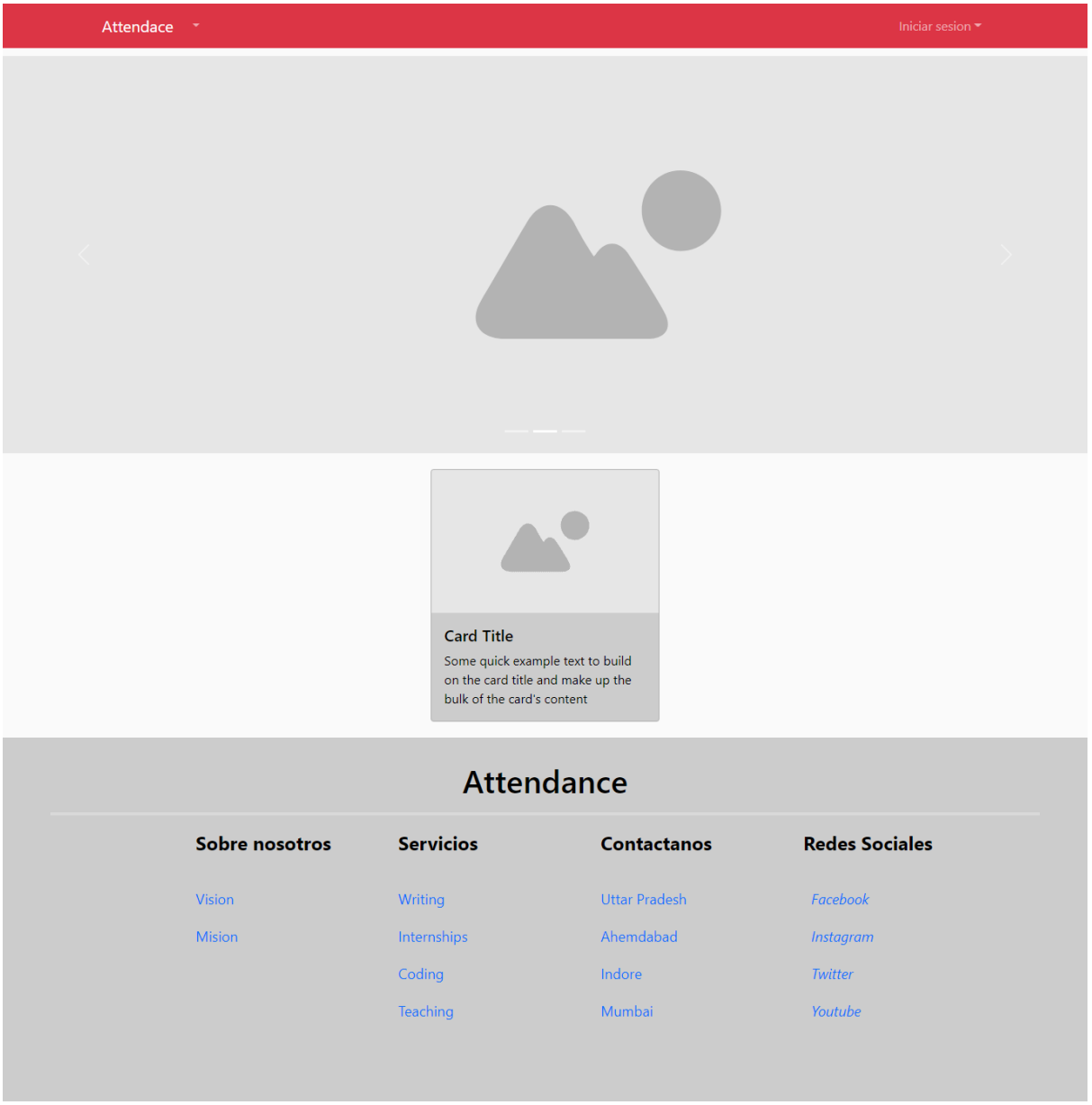
Capa de servidor

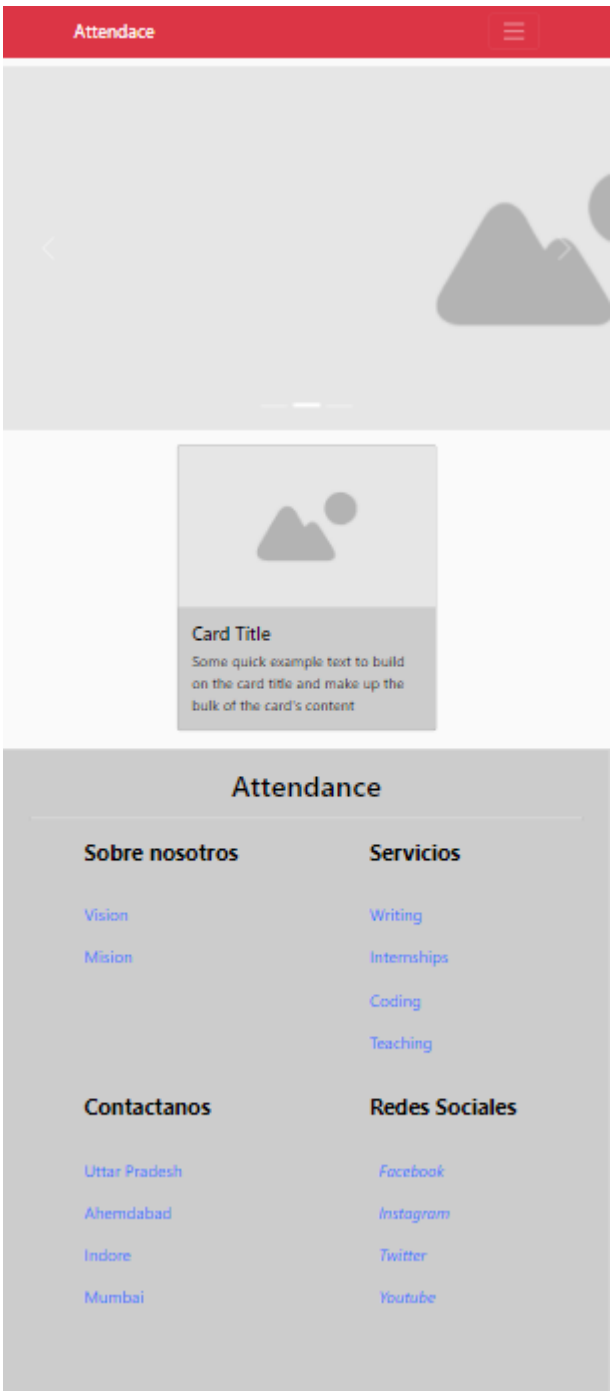
Para el desarrollo del servidor se usó la app generada por Express-Generator en Visual Studio Code, la app de servidor contiene las rutas y respuestas a las peticiones. Se puede ver con más claridad en GitHub:

Servidor

Capa de cliente

Para el desarrollo de la GUI se usó React, react-bootstrap y bootstrap. Se hizo el desarrollo con base a lo que había pensado en el diseño, dando un resultado como este:





[Código fuente GUI]

Arquitectura de la aplicación

El aplicativo en si es solo una pequeña parte de lo planteó inicialmente, ya que el tiempo dado para terminarlo fue insuficiente ya que se tenían que aprender muchas cosas nuevas como lo es usar React, Node.js, Express, peticiones HTTP y muchas que al parecer no fueron tenidas en cuenta al momento de plantearse el tiempo de entrega.

El aplicativo está compuesto por una capa de datos en postgres, una capa de servidor que procesa las solicitudes ingresando o recuperando información de la capa de datos apoyándose en Node.js + Express, y la capa gráfica muestra dichos datos que se realizó con React.

Mejoras futuras

Hay muchas posibilidades de mejora, ya que el aplicativo no está terminado por completo. Como posibles mejoras futuras sería la mejora en la estabilidad y velocidad de las peticiones, la mejora visual con colores acordes de la parte gráfica, como ser más elaborada pero sin perder de un tiempo de respuesta relativamente rápido.

Conclusión

En conclusión el desarrollo de la app fue bastante complejo por el poco tiempo que se le asignó a la entrega porque adicionalmente se tenían que aprender nuevos conceptos, realmente fue todo un reto desarrollar una app solo con una persona y básicamente desde cero en cuanto al tema del servidor y la parte gráfica

Anexos

Repositorio en GitHub: parcial-bd-1
Diagrama: Google Drive - Entidad_Relacion