**BASES DE DATOS 2**

**PROYECTO DE CURSO:**

**SISTEMA DE NOTAS POLI**

Deisy Carolina Ossa

Andres Felipe Ospina Henao

Emanuel Muñoz Lopera

Manuel Alejandro Correa

Politécnico colombiano Jaime Isaza Cadavid

Facultad de ingeniería

Medellín

2022

**Introducción**

Como propuesta al desarrollo del proyecto “Sistema de Notas Poli”, como equipo de trabajo hemos optado por las siguientes herramientas, ya que las consideramos las más óptimas por sus características y funcionamiento conjunto.

Para los sistemas de bases de datos NoSQL: Postgres y MongoDB.

Para el Back-End: NodeJS.

Para el Front-End: Angular.

A continuación mencionaremos las principales características de cada uno para dar más claridad y soporte a las elecciones.

**Características técnicas Back**

* **NodeJS**
  + Asíncrono y controlado por eventos: todas las API de la biblioteca Node.js son asíncronas, es decir, sin bloqueo. Esencialmente significa que un servidor basado en Node.js nunca espera que una API devuelva datos. El servidor pasa a la siguiente API después de llamarlo y un mecanismo de notificación de Events of Node.js ayuda al servidor a obtener una respuesta de la llamada API anterior.
  + Muy rápido: al estar construido en el motor JavaScript V8 de Google Chrome, la biblioteca Node.js es muy rápida en la ejecución de código.
  + Sin almacenamiento en búfer: las aplicaciones Node.js nunca almacenan en búfer ningún dato. Estas aplicaciones simplemente generan los datos en fragmentos
  + Procesos en un solo hilo, pero altamente escalable: Node.js utiliza un modelo de un solo hilo con bucle de eventos. El mecanismo de eventos ayuda al servidor a responder sin bloqueos y hace que el servidor sea altamente escalable en comparación con los servidores tradicionales que crean hilos limitados para manejar las solicitudes. Node.js utiliza un solo programa de subprocesos y el mismo programa puede proporcionar servicio a un número mucho mayor de solicitudes que los servidores tradicionales como el Servidor Apache HTTP.
  + Licencia basada en código abierto − Node.js se distribuye gracias a una licencia MIT.
* **Postgresql**
  + Gracias a sus estándares y estructura SQL, Postgresql resulta sencillo para realizar consultas y scripts
  + Tiene soporte a diferentes tipos de datos
  + Es un sistema gratuito y multiplataforma
  + Posee gran escalabilidad, estabilidad y confiabilidad

**Características técnicas Front**

* Lenguaje Usado: **Angular, Angular Material**

Características:

* + Las aplicaciones de Angular se cargan rápidamente gracias al nuevo enrutador de componentes. Este ofrece una división automática de códigos para que los usuarios solo carguen el código necesario para procesar la vista que solicitan.
  + Fácil de implementar diseños basados en Material Design (desarrollo creado por Google)
  + Cada Feature puede tener su diseño y código aparte generando estructura y control en los cambios realizados.
  + Permite crear rápidamente vistas de interfaz de usuario con una sintaxis de plantilla simple y potente.

**Bases de datos**

* **Mongo**
  + El proyecto que se diseñó para la práctica tiene su backend en NodeJS por lo cual se busca empelar una arquitectura básica MEAN (Mongo Express Angular Node.js) Donde Mongo se presenta como la opción de Base de datos a Usar.
  + Mongo almacena los datos en estructuras parecidas a un JSON, De igual forma NodeJs y Angular tiene la capacidad natural de trabajar con esta estructura, lo que hace ideal integrar las tres herramientas y mantener una estructura común de los datos en todo el proceso.
  + MongoDB vienen ya con serializadores (mecanismo ampliamente usado para transportar objetos a través de una red, para hacer persistente un objeto en un archivo o base de datos, o para distribuir objetos idénticos a varias aplicaciones o localizaciones) que convierten las estructuras de datos de tu lenguaje de programación en documentos que MongoDB puede entender.
  + MongoDB es una buena alternativa para el volumen de datos, y de ser necesario es flexible y puede ser escalada según demandan tanto de forma vertical (CPU y RAM) sino que también de forma horizontal (creando más nodos).
  + Dentro de las bases propuestas como alternativas, es con la que más contacto académico, diseño y programación ha tenido en general el equipo de trabajo. No es un contacto especializado y con mucha experiencia, pero a comparación de las otras bases de datos si es mucho.

**Ejecutar y Montar entorno del Proyecto General**

**Repositorio Git:**

<https://github.com/manuelcorrea16/SistemaNotasPoli>

**Postgresql**

**Parámetros instancia de conexión**

Útiles para conectarse a la BD desde cualquier herramienta que prefieran.

* + - **Server** castor.db.elephantsql.com (castor-01)
    - **Region** amazon-web-services::us-east-1
    - **User & Default database** deprsegw
    - **Password :** PNne0NY0oipuuxzlHbJKvQiJM09-dU06
    - **StringConection(URL) :** postgres://deprsegw:PNne0NY0oipuuxzlHbJKvQiJM09-dU06@castor.db.elephantsql.com/deprsegw
    - **Api Key Acces:** bcf6ce92-805d-4037-9d41-c9d6ae143375

**Métodos de conexión a base datos Postgresql:** se encuentra en ..\bdFront\index.js

* + - **getSedes**
    - **getProgramas**
    - **getAsignaturas**
    - **terminarConexion**

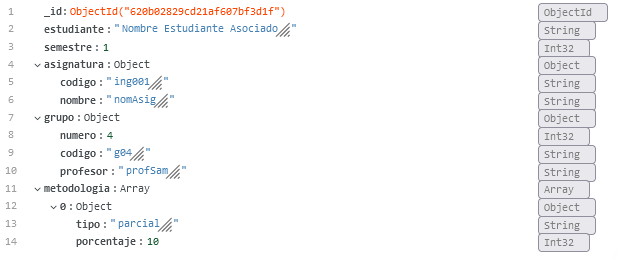
**MongoDB**

**Parámetros conexión**

* **URL:** mongodb+srv://db-user: db-password @cluster0.bjtml.mongodb.net/regEstudiantes?retryWrites=true&w=majority
* **Usuario:** db-user
* **Contraseña:** db-password
* **Nombre DB:** regEstudiantes
* **Estructura datos mongoDB**
  + Registros estudiantes



* + Notas



**Angular y Nodejs**

1. Video tutorial: <https://www.youtube.com/watch?v=kqYuyACFVkE>
2. Instalar Node <https://nodejs.org/es/>
3. Actualizar Node en línea de comandos: npm install –g npm@latest
4. Se recomienda Borrar cache Npm : npm cache clean –force (después npm cache verify)
5. Se recomienda desinstalar las Auditorias npm para que no impida inhalaciones y actualizaciones: npm set audit false
6. Verificamos que no tengamos una vesion de angular obsoleta:
   1. npm uninstall – g angular-cli
   2. npm uninstal –g @angular/cli
7. Instalar: npm install -g @angular/cli@latest
8. Abro la terminal dentro de la carpeta del proyecto
9. Creo proyecto Angular: ng new
10. Ejecutar:
    1. Dentro de la carpeta que se creo
    2. ng serve

**Postgres**

1. Video tutorial <https://www.youtube.com/watch?v=ap4C4384Cu8>
2. Página de apoyo: <https://node-postgres.com/>
3. Inicializar Proyecto (Sin o ha sido inicializado Node): npm init –y
4. Instalar librería para trabajar con PostGress : npm install pg
5. Intalar: npm i --save-de @types/node
6. Intalar Express : npm install express --save
7. Encriptador de contraseñas: npm install bcrypt-nodejs –save
8. Para trabajar y convertir Json a objetos y viceversa: npm install body-parse –save
9. Para usar ficheros atreves de los formularios: npm install connect-multiparty –save
10. Para trabajar con Bases de datos Mongo: npm install mongoose --save
11. npm install nodemon --save-dec
12. Ejecutar Fuentes Nodo: node ../index.js

**API MongoDB**

1. Video tutorial <https://www.youtube.com/watch?v=-7aK7zn7NaY>
2. Instalar Node y dependencias:
   1. Npm install mongojs cors
   2. Nodemon, express y angular ya instalados.
3. Configurar Nodemon en index.js
   1. Package.json ->

"scripts": {

…

        "startapi": "node Back/indexapi.js",

        "dev": "nodemon Back/index.js"

    },

Comandos para inicializar entorno:

Npm run startapi

1. Configurar express y entorno app en puerto 3000 dedicado para la API
2. Configurar de los middlewares
3. Conectar a la Base de datos Mongo y configuración CRUD
   1. Conexión a Mongo

Mongojs = Require(‘mongojs’)

Mongojs(‘mongodb+srv://db-user: db-password @cluster0.bjtml.mongodb.net/regEstudiantes?retryWrites=true&w=majority)

* 1. Métodos CRUD:
     + Registros.js -> Get, Post
     + Notas.js -> Get, Post
     + Informe.js -> Get

1. Integrar rutas API en Angular

**Plantilla de diseño**

Diagram

Description automatically generated

**Modelo Relacional BD NoSQL seleccionada**

