**BASES DE DATOS 2**

**PROYECTO DE CURSO: Sistema Notas Poli**

**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

**Características técnicas Back**

* **Lenguaje Usado: NodeJS**

Las siguientes son algunas de las características importantes que hacen de Node.js la primera opción para los arquitectos de software.

* + 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.
* Métodos de conexión a base datos mongo
* Métodos de consulta a Mongo
* Métodos de Inserción a Mongo
* Métodos de Edición a Mongo
* Métodos de Eliminación a Mongo
* **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**

**Características técnicas Front**

* Lenguaje Usado: **Angular**
* Características

**Bases de datos**

* Mongo

El proyecto que se diseñó para la práctica tiene su backen 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 Angula 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 estrés al que se verá sometido según el proyecto, 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).

En MongoDB tenemos múltiples operadores que nos permiten crear consultas poderosas con poco código, además que contamos con las Agregaciones que son un mecanismo que nos permite realizar operaciones entre múltiples colecciones.

Aunque las otras bases de datos también tienen herramientas de código abierto MongoDB permite un uso “comercial” del proyecto, además de que existen muchos proveedores que ofrecen MongoDB.

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.

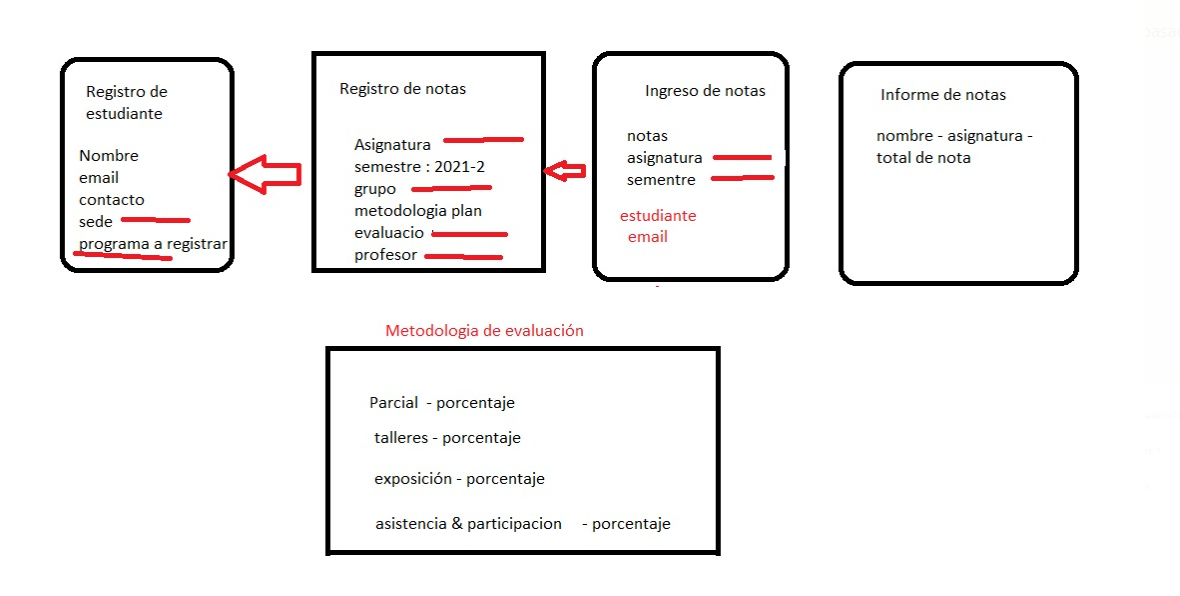
**Ejecutar y Montar entorno del Proyecto General**

**Angular**

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
11. Repositorio Git <https://github.com/deisyossa/bdFront>

**Nodejs+Postgres**

1. Video tutoria <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 src\conections\conectionPostGres.js

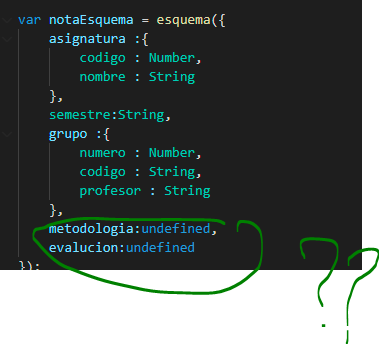


**Dudas**

No entendí como tiene planeado el registro de notas y el ingreso de notas

Que entidades se manejaban ahí

Que datos guardaría cada entidad



Cada entidad se debe definir de la anterior manera para que lo tengan en cuenta.

Se cambia nombre del proyecto principal

Se realizan instalaciones y configuraciones para trabajar con NodeJs

Se crean la carpeta Modelo estructurado

Se crean algunas entidades de para mongo

Y se crean algunas fuentes de prueba para la conexión Postgress