TRABAJO PRACTICO FINAL

Desarrollo de APP WEB – APP ANDROID

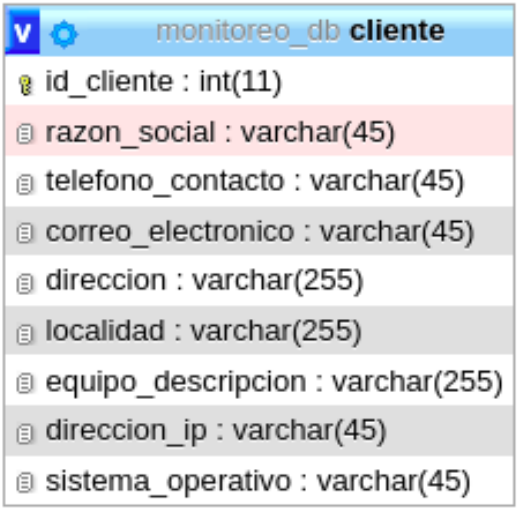
PARTE N°1

CONSIGNAS: Inicio de construcción de APP WEB (cliente y servidor)

1-BASE DE DATOS:

Crear una nueva Base de Datos denominada **"monitoreo_db"**

- Crear una tabla denominada **"cliente"** que deberá contar con los siguientes campos:



2-SERVER

Crear un único servidor denominado **"SERVER-MONITOREO"** que integre tanto la **API** y **SocketIO**. Para ello se deberá llevar a cabo la inicialización e instalación de dependencias:

- Inicializar un nuevo proyecto de **NODEJS**
- Instalar las dependencias necesarias para el desarrollo de la API y el SOCKET
    - Express
    - Morgan
    - Nodemon
    - Cors
    - Mysql
    - SockelO

Desarrollo de API:

Implementar un **CRUD** de la tabla **"cliente"** realizando las consultas correspondientes a la base:

- Insert
- Delete
- Update
- Select

Luego, implementar las rutas destinadas a cada una de las operaciones CRUD. Se aconseja usar clientes para testear (ej. INSOMNIA).

Desarrollo Socket:

Dentro del mismo servidor donde se llevo adelante la implementación de la API, configurar lo necesario para también emitir datos a través del protocolo WEBSOCKET, segundo a segundo a través de **"setInterval()".**

A continuación mostrar los siguientes indicadores, que se especifican en detalle(respetar la forma en que se deben acceder a los datos):

CPU

- cpu.usage()
- cpu.free()
- cpu.count()
- cpu.model()

DRIVE: Mostrar las propiedades del objeto que entrega **info()**

- drive.info()
  - usedGb
  - freeGb
  - freePercentage
  - totalGb

MEMORIA

- mem.free()
  - totalMemMb
  - freeMemMb
- mem.used()
  - usedMemMb

NETSTAT

- netstat.inOut()
  - eth0 → inputMb
  - eth0 → outputMb

- netstat.stats()
  - interface: 'lo' → inputBytes
  - interface: 'lo' → outputBytes

OS (Sistema operativo)

- os.oos()
- os.hostname()
- os.arch()

3-CLIENTE

Inicializar un nuevo proyecto de VUE, que llevará la denominación "CLIENTE-MONITOREO"

Dicho cliente deberá contar con las siguientes dependencias instaladas:

- axios V0.21.4
- bootswatch 5
- vue-axios V3.3.6
- socket.io V2.3.0
- socket.io-client 2.3.0

En dicho proyecto generar las siguientes vistas y componentes:

- Vista **Cliente.vue** → componente **ClienteCrud.vue**
- Vista **Cpu.vue** → componente **CpuIndicador.vue**
- Vista **Drive.vue** → componente **DriveIndicador.vue**
- Vista **Memoria.vue** → componente **MemoriaIndicador.vue**
- Vista **Netstat.vue** → componente **NetstatIndicador.vue**
- Vista **Os.vue** → componente **OsIndicador.vue**

En cada componente, se diseñarán los indicadores antes solicitados. Aquellos datos emitidos por el servidor, que sean constantes, como el **os.oos()**, no requerirá emitirlo segundo a segundo (es decir, no necesita de un **setInterval()** en el servidor). Pues el mismo no varía en el tiempo.

IMPORTANTE

El diseño (HTML Y CSS en conjunción con Bootswatch) de la aplicación es libre. Recuerden que cada VISTA representa una página. Por lo tanto en cada una de ellas, se deberá mostrar el componente correspondiente. A la hora de evaluar el diseño, se tendrá en cuenta el uso correcto de filas y columnas, así como también márgenes y espaciados de los diferentes elementos HTML.
Cumplir además con cada uno de los nombramientos, tanto de archivos, base de datos, tabla, etc.
La forma de entrega se especificará en la siguiente parte de TP FINAL.
Si alguno de los indicadores no logra mostrarse debido a incompatibilidad de la librería NODE-OS-UTILS con su sistema operativo, debe notificarlo, para tenerlo presente a la hora de evaluar el desarrollo.