Estrategias de Persistencia

Clase Nro 3.

En esta clase Nro. 3, vamos a ejecutar códigos en javascript con la tecnología Nodejs, para esta ejecución de códigos e instalación de dependencias sugiero hacer grupos de no más de 3 personas de manera virtual, a modo que entre todos puedan solucionar los inconvenientes que les puedan aparecer en el período de instalación de dependencias.

Esta clase durará 2 semanas ya que hay muchas dependencias e instalaciones que hacer para poder empezar a realizar los ejercicios en Nodejs.
En esta clase vamos a ver los siguientes puntos.

1) ¿Qué es un ORM
2) ¿Instalación de paquetes y dependencias?
3) Ejecución de códigos simples con Nodejs y Sequelize.

---

1) ¿Qué es un ORM?
En el siguiente link vamos a encontrar un video interesante sobre qué es y cómo se utiliza un ORM. https://www.youtube.com/watch?v=pS1nrxDj3yM

2) Para instalar Nodejs y sus dependencias vamos a realizar las acciones que figuran a continuación:

Para Linux:
https://www.digitalocean.com/community/tutorials/how-to-install-node-js-on-ubuntu-20-04-es
Para windows : https://phoenixnap.com/kb/install-node-js-npm-on-windows
Para Mac: https://www.webucator.com/article/how-to-install-nodejs-on-a-mac/

Ejemplo de instalación en entorno Linux

IMPORTANTE: La siguiente instalación, se realizará en un entorno LINUX Ubuntu 20.04, en el caso que haya alumnos que no posean este sistema operativo arriba encontrarán tutoriales alternativos.

a) Instalación de node y npm

```
sudo apt update
sudo apt install nodejs
sudo apt install npm
```

Para verificar la correcta instalación, podemos ejecutar la siguiente instrucción y verificar que nos indique el número de versión:

```
node --version
npm --version
```

b) Descargamos los ficheros del campus a un directorio de nuestra computadora.
Ejemplo:
`/Users/pmarcelli/UNAHUR/ep/`

c) Ingresamos a la terminal, y nos dirigimos por línea de comando al PATH donde se encuentran los ficheros descargados.
Ejemplo:
`cd /Users/pmarcelli/UNAHUR/ep/`

d) Una vez dentro del PATH ejecutamos la instalación del ORM sequelize:
(Sequelize es un ORM propio de Nodejs.)

```
npm install --save sequelize
```

e) Luego instalamos la librería que nos permitirá conectarnos a la BD que vamos a utilizar,en este caso es “mariadb” dentro de nuestro proyecto.
(IMPORTANTE: si no tienen instalado mariadb en sus computadoras deberán instalarlo)

```
npm install --save mariadb
```

En esta sección vamos a ver los ficheros que se encuentran compartidos en el campus con el nombre “Ejemplos”.
Lo primero que debemos hacer es conocer la sintaxis de lo que queremos ejecutar, así que a continuación voy a explicar lo que significa cada una de las líneas de nuestros programas.

La forma de ejecutar este programa es desde la terminal:
`node /PATH_EN_DONDE_ESTAN_LOS_FUENTES/nombre_del_programa.js`

IMPORTANTE:
En estos programas, se utilizará una base de datos llamada prueba. La misma debe ser creada de manera manual.

Pueden ejecutar el siguiente script en el cliente de base de datos que estén utilizando:

```
create database prueba;
```

```
// Esta línea hace referencia  a que el programa va a requerir el ORM llamado sequelize.
const Sequelize = require('sequelize');


// Cadena de conexión a la base de datos.
// base de datos llamada "prueba".
// usuario y clave, llamado "root".
const sequelize = new Sequelize('prueba', 'root', 'root', {
  host: 'localhost',
  dialect: 'mariadb' /* one of 'mysql' | 'mariadb' | 'postgres' | 'mssql' */
});


// se intenta autenticar a la base de datos, con el método authenticate().
// si la autenticación es correcta, arroja un log de conexión exitosa, caso contrario arroja un error.
sequelize
  .authenticate()
  .then(() => {
    console.log('Connection has been established successfully.');
  })
  .catch(err => {
    console.error('Unable to connect to the database:', err);
  });


// Creo una clase llamada "User" que la extiendo del ORM.
// Indicándole los atributos que quiero que espeje en la base de datos.
class User extends Sequelize.Model {}
User.init({
  firstName: Sequelize.STRING,
  lastName:Sequelize.STRING
}, { sequelize, modelName: 'users' });


// Creo un nuevo usuario y guardo el registro en la tabla creada anteriormente.
sequelize.sync()
  .then(() => User.create({
    firstName: 'Pedro',
    lastName: 'Rodriguez'
  }))
  .then(jane => {
    console.log(jane.toJSON());
  });
```

Actividad práctica.

Ejecutar los ejercicios que figuran en la carpeta del campus llamada **Ejemplos**
Se debería poder insertar, actualizar y eliminar registros de la base de datos.