Universidad Mariano Gálvez de Guatemala
Mazatenango Suchitepéquez
Programación Web
Ing. Axel Aguilar

PROYECTO PROGRAMACION WEB

Fase Final

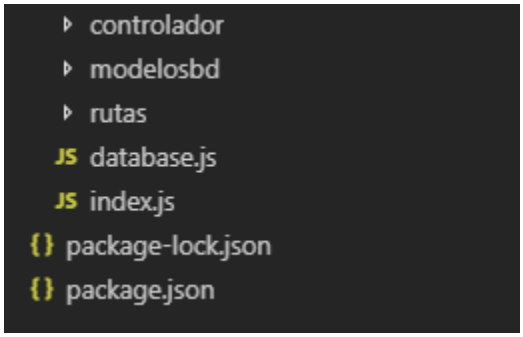
Nombre: Yonatan Alexander Ixcoy Chuc
Carrera: Ingeniera en Sistemas
Carné: 3090-15-12540
Sección: “A”

Semestre: VIII

FASE FINAL PROYECTO DE PROGRAMACIÓN WEB

Documentación Codificación

- Creamos tres carpetas las cuales contendrán los archivos de nuestro Backend



La primer carpeta contendrá todos los controladores, la segunda contendrá los modelos de las base de datos y en la carpeta colocaremos las URL que van a estar disponibles dentro de nuestro servidor.

- Creamos el documento con el nombre de index.js que es el encargado de arrancar el servidor.

- Creamos variables

```
const express = require('express');
```

Esta variable será la encargada de tener toda la funcionalidad de nuestro servidor.

- Configuramos el puerto del servidor

```
app.set('port', process.env.PORT || 3000)
```

Set crea una variable que va ser accedida a cualquier parte de nuestra aplicación.

- Creamos funciones que nos van servir para procesar los datos

```
app.use(morgan('dev'));
app.use(express.json());
```

- Creamos otro archivo el cual nombraremos database.js

```
const mongoose = require('mongoose');

const URI = 'mongodb://localhost/crud-final-cliente';

mongoose.connect(URI)

    .then(db => console.log('DB is connected'))
    .catch(err => console.error(err));

module.exports = mongoose;
```

En este modulo, creamos la conexión a nuestra base de datos creada en Mongo.

- Dentro de index.js requerimos la conexión de mongoose

```
const { mongoose } = require('./database');
```

- Creamos un nuevo archivo al cual llamaremos cliente.ruta.js, el cual contendrá las rutas de los clientes.

```
1   const express = require('express');
2   const router = express.Router();
3   const cliente = require('../controlador/cliente.controlador');
4
5   router.get('/', cliente.getClientes);
6   router.post('/',cliente.createCliente);
7   router.get('/:id',cliente.getCliente);
8   router.put('/id:',cliente.editCliente);
9   router.delete('/id:',cliente.deleteCliente);
10
11  module.exports = router;
```

A través de este archivo podemos agregar clientes, eliminarlos, enlistarlos.
Requerimos express nuevamente, ahora servirá para crear rutas del servidor.
El módulo router nos devuelve un objeto que tenemos que almacenar.

- En nuestro archivo index.js mandamos a requerir la ruta que hemos creado

```
app.use('/api/cliente',require('./rutas/cliente.ruta'));
```

- Creamos un archivo el cual llamaremos cliente.js, en el definiremos los esquemas de nuestra base de datos.

```
const mongoose = require('mongoose');
const {Schema} = mongoose;

const ClienteSchema = new Schema({

    id: {type: String, required: true},
    nombre: {type: String, required: true},
    telefono: {type: String, required: true},
    direccion: {type: String, required: true},
    nit: {type: String, required: true},

});

module.exports = mongoose.model('Cliente',ClienteSchema);
```

- Creamos un nuevo archivo con el nombre cliente.controlador.js

```
const Cliente = require('../modelosbd/cliente')

const clienteCtrl = {};

clienteCtrl.getClientes = async (req, res) => {

    const clientes = await Cliente.find();
   res.json(clientes);
}


clienteCtrl.createClientes = async (req, res) => {

    console.log(req.body);
    res.json('recibido');

}
```

El controlador vamos a colocarlo como un objeto, luego exportamos el objeto, a este objeto podemos agregarle múltiples métodos, como por ejemplo, obtener clientes, agregar clientes y podemos agregar cualquier función.