

**GA8-220501096-AA1-EV01 DESARROLLAR SOFTWARE A PARTIR
DE LA INTEGRACIÓN DE SUS MÓDULOS COMPONENTES**

PRESENTADO POR

EDGAR ALSIBAR MUNOZ ISAZA
JONIER ESTEBAN RODRIGUEZ
JOSE EDUARDO LONDOÑO

INSTRUCTOR

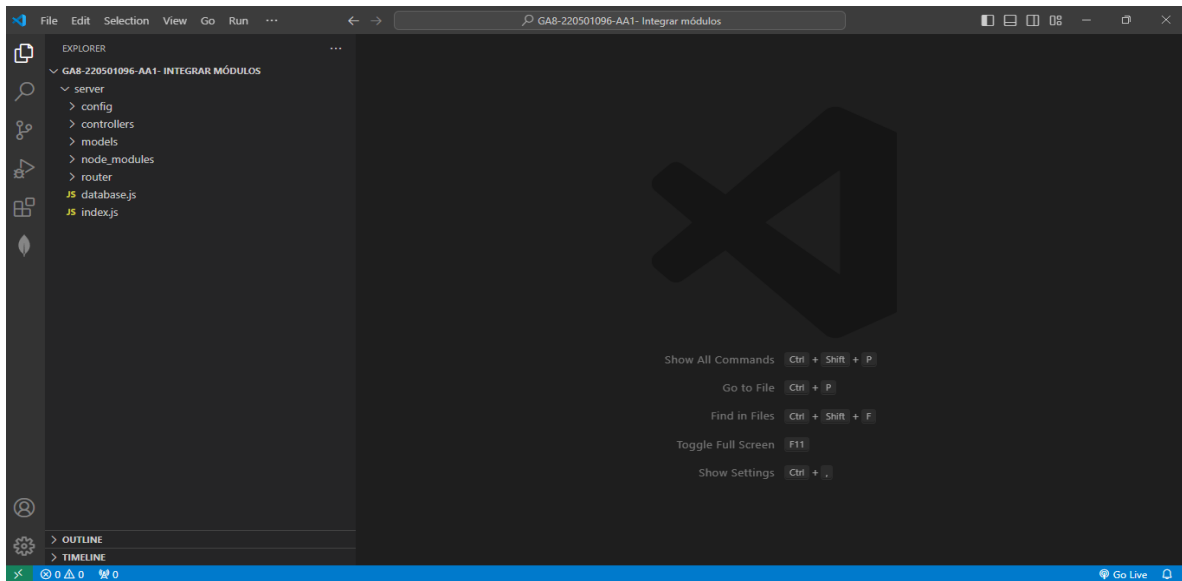
JUAN MANUEL ALDANA
INGENIERO DE SISTEMAS

ANALISIS Y DESARROLLO DE SOFTWARE

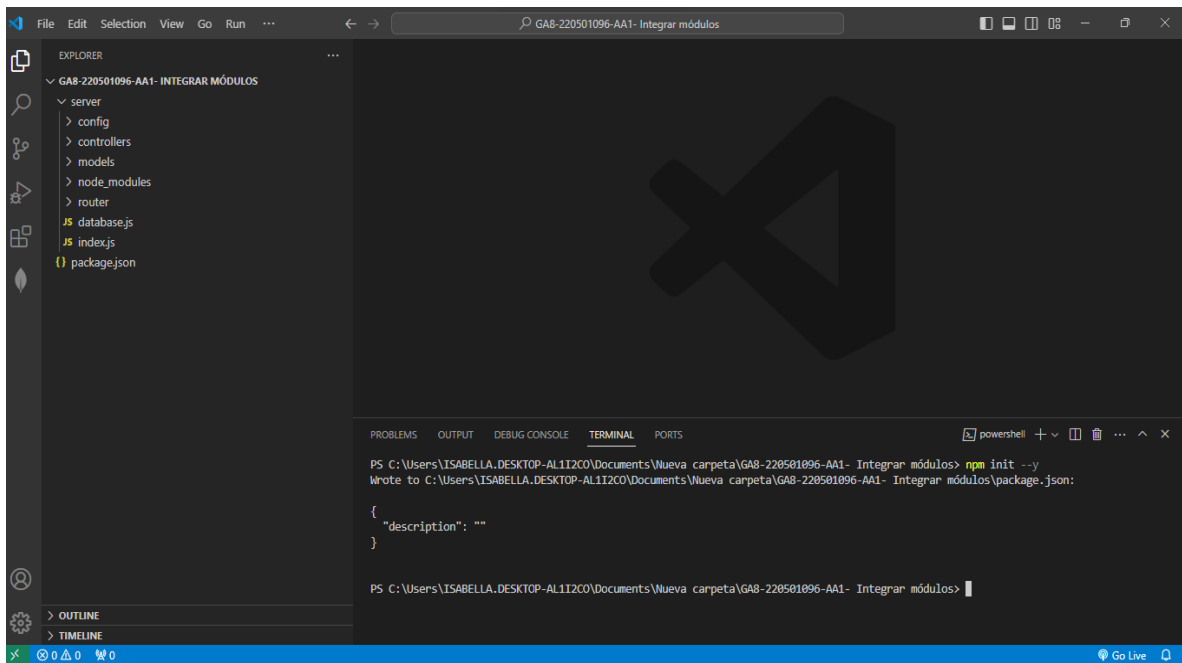
SERVICIO NACIONAL DE APRENDIZAJE SENA
ABRIL DE 2024

GITHUB <https://github.com/Edgarmuoz2003/Sifact-completo.git>

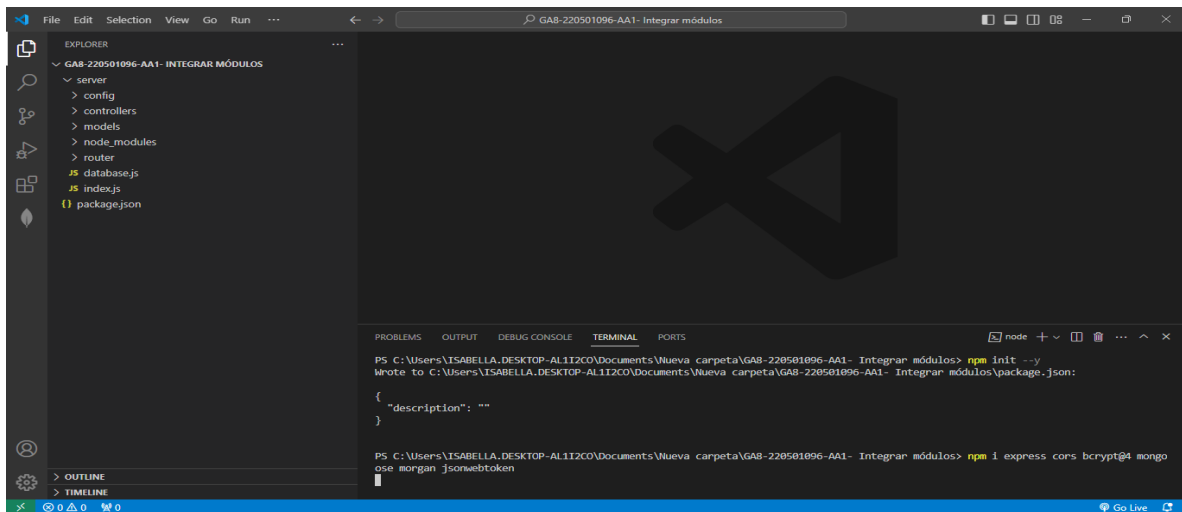
Creamos las carpetas para las diferentes instancia que son los controladores los Reuters módulos y la configuración



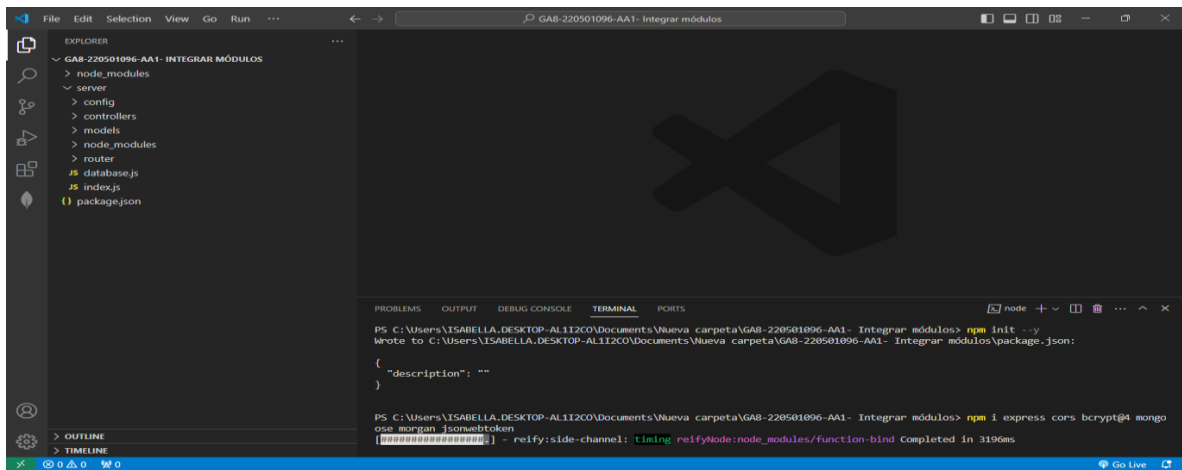
Inicializamos el proyecto con nodejs



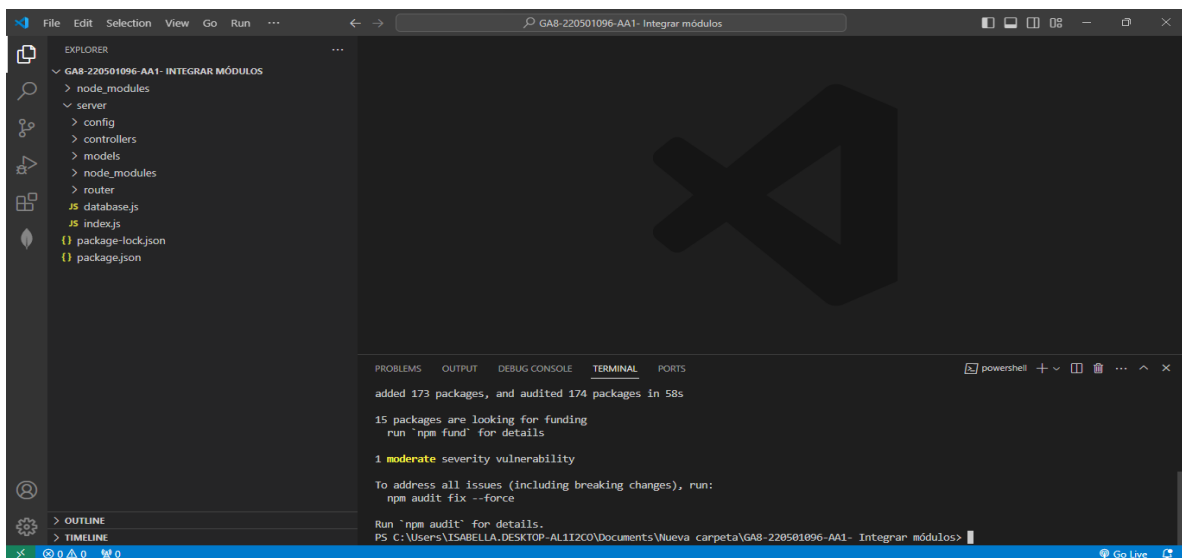
Identificamos e instalamos la dependencia necesaria para que nuestro servidor corra



```
File Edit Selection View Go Run ...  
GAB-220501096-AA1- Integrar módulos  
EXPLORER  
GAB-220501096-AA1- INTEGRAR MÓDULOS  
server  
config  
controllers  
models  
node_modules  
router  
database.js  
index.js  
package.json  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
node  
PS C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos> npm init -y  
Wrote to C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos\package.json:  
  
{  
  "description": ""  
}  
  
PS C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos> npm i express cors bcrypt@4 morgan  
ose morgan jsonwebtoken
```

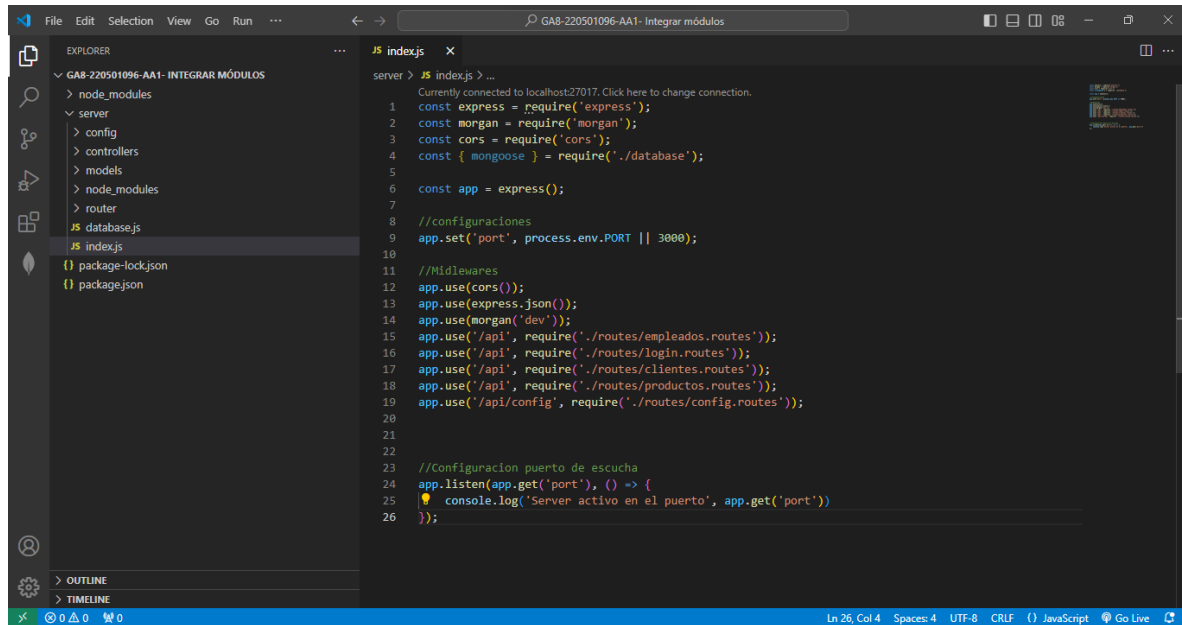


```
File Edit Selection View Go Run ...  
GAB-220501096-AA1- Integrar módulos  
EXPLORER  
GAB-220501096-AA1- INTEGRAR MÓDULOS  
node_modules  
server  
config  
controllers  
models  
node_modules  
router  
database.js  
index.js  
package.json  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
node  
PS C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos> npm i express cors bcrypt@4 morgan  
jsonwebtoken  
Wrote to C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos\package.json:  
  
{  
  "description": ""  
}  
  
PS C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos> npm i express cors bcrypt@4 morgan  
jsonwebtoken  
[reify:side-channel: timing reifyNode:node_modules/function-bind Completed in 3196ms]
```



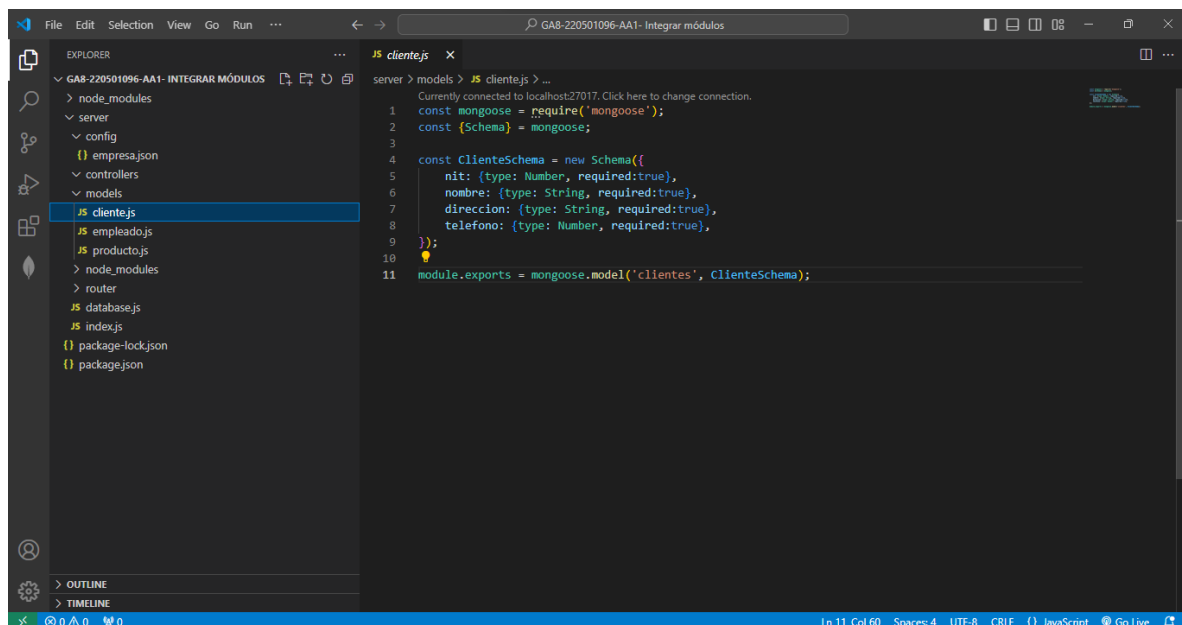
```
File Edit Selection View Go Run ...  
GAB-220501096-AA1- Integrar módulos  
EXPLORER  
GAB-220501096-AA1- INTEGRAR MÓDULOS  
node_modules  
server  
config  
controllers  
models  
node_modules  
router  
database.js  
index.js  
package-lock.json  
package.json  
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
powershell  
added 173 packages, and audited 174 packages in 58s  
  
15 packages are looking for funding  
run 'npm fund' for details  
  
1 moderate severity vulnerability  
  
To address all issues (including breaking changes), run:  
  npm audit fix --force  
  
Run 'npm audit' for details.  
PS C:\Users\ISABELLA.DESKTOP-AL112CO\Documents\Nueva carpeta\GAB-220501096-AA1- Integrar módulos>
```

En la primera parte requerimos las dependencias instaladas luego configuramos el puerto donde se escuchara nuestra aplicación y seguidamente configuramos middlewares y le decimos a la app en donde están nuestras respectivas rutas para cada una de las módulos por ultimo configuramos el puerto de escucha



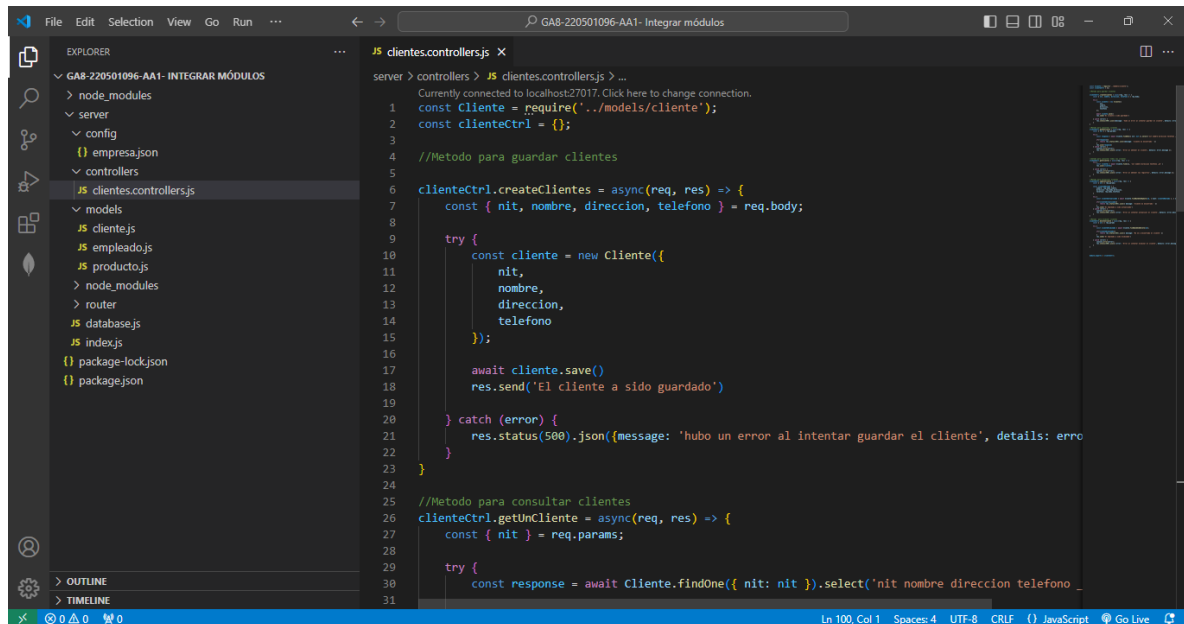
```
server > JS index.js > ...
Currently connected to localhost27017. Click here to change connection.
1  const express = require('express');
2  const morgan = require('morgan');
3  const cors = require('cors');
4  const { mongoose } = require('./database');
5
6  const app = express();
7
8  //configuraciones
9  app.set('port', process.env.PORT || 3000);
10
11 //Middleware
12 app.use(cors());
13 app.use(express.json());
14 app.use(morgan('dev'));
15 app.use('/api', require('./routes/empleados.routes'));
16 app.use('/api', require('./routes/login.routes'));
17 app.use('/api', require('./routes/clientes.routes'));
18 app.use('/api', require('./routes/productos.routes'));
19 app.use('/api/config', require('./routes/config.routes'));
20
21
22
23 //Configuracion puerto de escucha
24 app.listen(app.get('port'), () => {
25   console.log('Server activo en el puerto', app.get('port'))
26 });
```

Creamos los modelos que usara nuestro gestor de bases de datos “mongodb” para determinar los respectivos campo que llevara cada tabla según las diferentes entidades requeridas en el proyecto



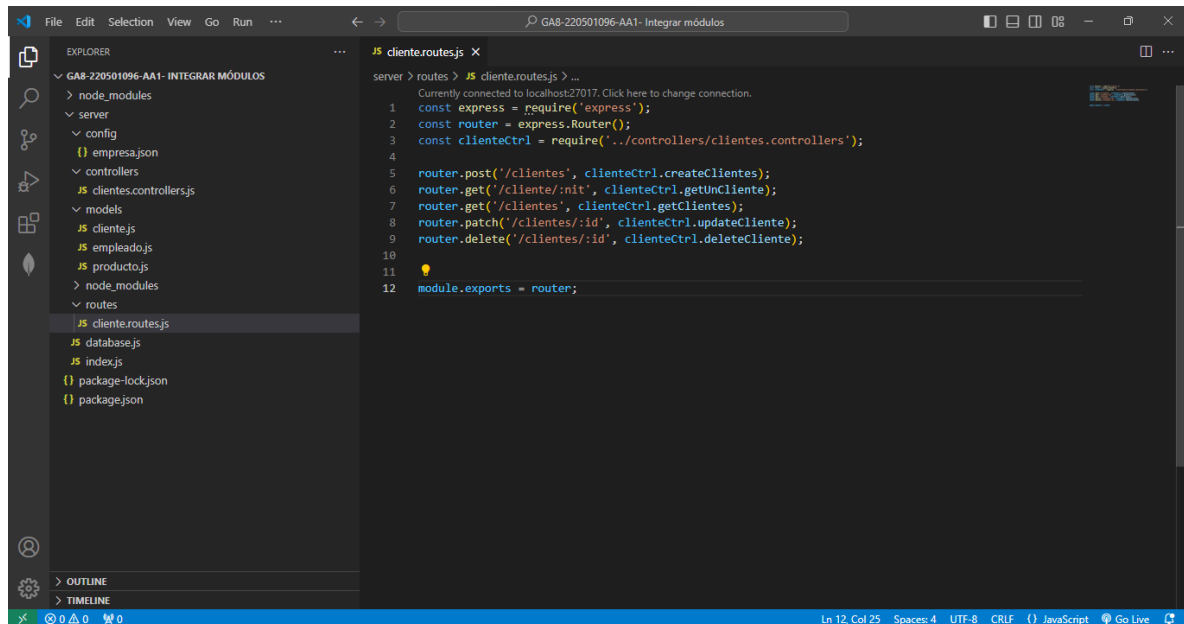
```
server > models > JS cliente.js > ...
Currently connected to localhost27017. Click here to change connection.
1  const mongoose = require('mongoose');
2  const { Schema } = mongoose;
3
4  const ClienteSchema = new Schema({
5    nit: {type: Number, required:true},
6    nombre: {type: String, required:true},
7    direccion: {type: String, required:true},
8    telefono: {type: Number, required:true},
9  });
10
11 module.exports = mongoose.model('clientes', ClienteSchema);
```

Dentro de la carpeta controladores creamos un archivo para manejar cada uno de los métodos que controlara cada una de las funciones de nuestra app y de igual manera creamos dichos métodos



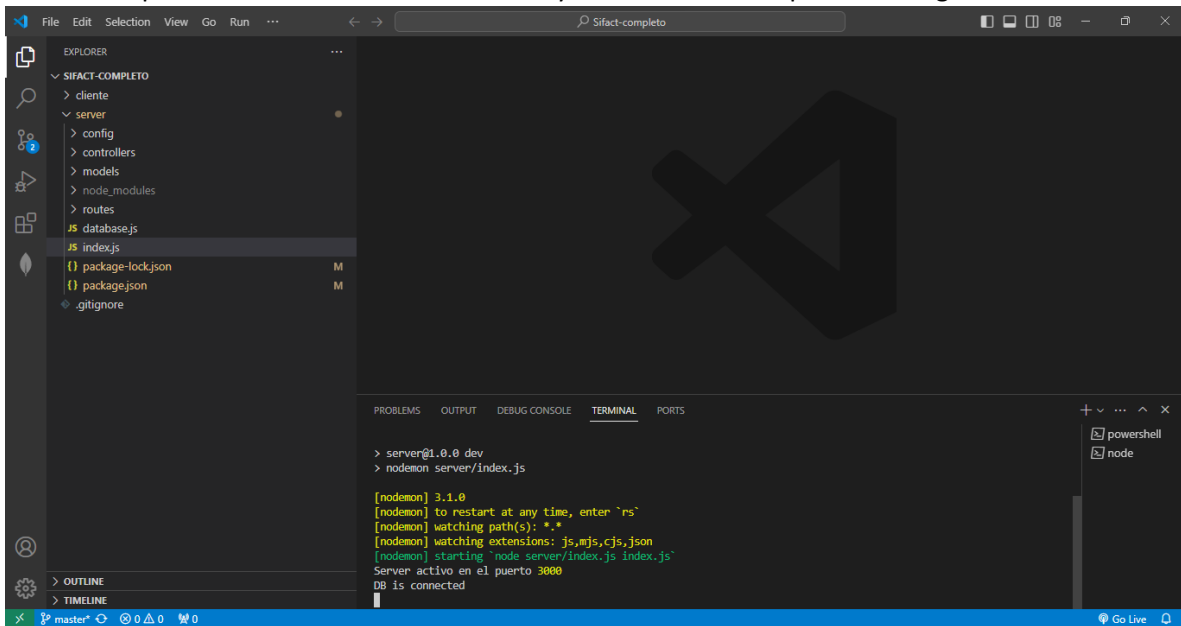
```
server > controllers > JS clientes.controllers.js > ...
Currently connected to localhost:27017. Click here to change connection.
1 const Cliente = require('../models/cliente');
2 const clienteCtrl = {};
3
4 //Metodo para guardar clientes
5
6 clienteCtrl.createClientes = async(req, res) => {
7   const { nit, nombre, direccion, telefono } = req.body;
8
9   try {
10     const cliente = new Cliente({
11       nit,
12       nombre,
13       direccion,
14       telefono
15     });
16     await cliente.save()
17     res.send("El cliente a sido guardado")
18   } catch (error) {
19     res.status(500).json({message: 'hubo un error al intentar guardar el cliente', details: error
20   })
21 }
22
23 //Metodo para consultar clientes
24
25 clienteCtrl.getUnCliente = async(req, res) => {
26   const { nit } = req.params;
27
28   try {
29     const response = await Cliente.findOne({ nit: nit }).select('nit nombre direccion telefono
30   )
31 }
```

Luego desde la carpeta de rutas creamos las rutas con los diferentes tipos de consulta que se realizaran a la base de datos (GET, POST, PATCH, DELETE)



```
server > routes > JS cliente.routes.js > ...
Currently connected to localhost:27017. Click here to change connection.
1 const express = require('express');
2 const router = express.Router();
3 const clienteCtrl = require('../controllers/clientes.controllers');
4
5 router.post('/clientes', clienteCtrl.createClientes);
6 router.get('/cliente/:nit', clienteCtrl.getUnCliente);
7 router.get('/clientes', clienteCtrl.getClientes);
8 router.patch('/clientes/:id', clienteCtrl.updateCliente);
9 router.delete('/clientes/:id', clienteCtrl.deleteCliente);
10
11
12 module.exports = router;
```

Probamos que nuestro servidor este corriendo y escuchando en el puerto configurado



```
File Edit Selection View Go Run ... Sifact-completo
```

EXPLORER

- SIFACT-COMPLETO
 - cliente
 - server
 - config
 - controllers
 - models
 - node_modules
 - routes
 - database.js
 - index.js
 - package-lock.json
 - package.json
 - .gitignore

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

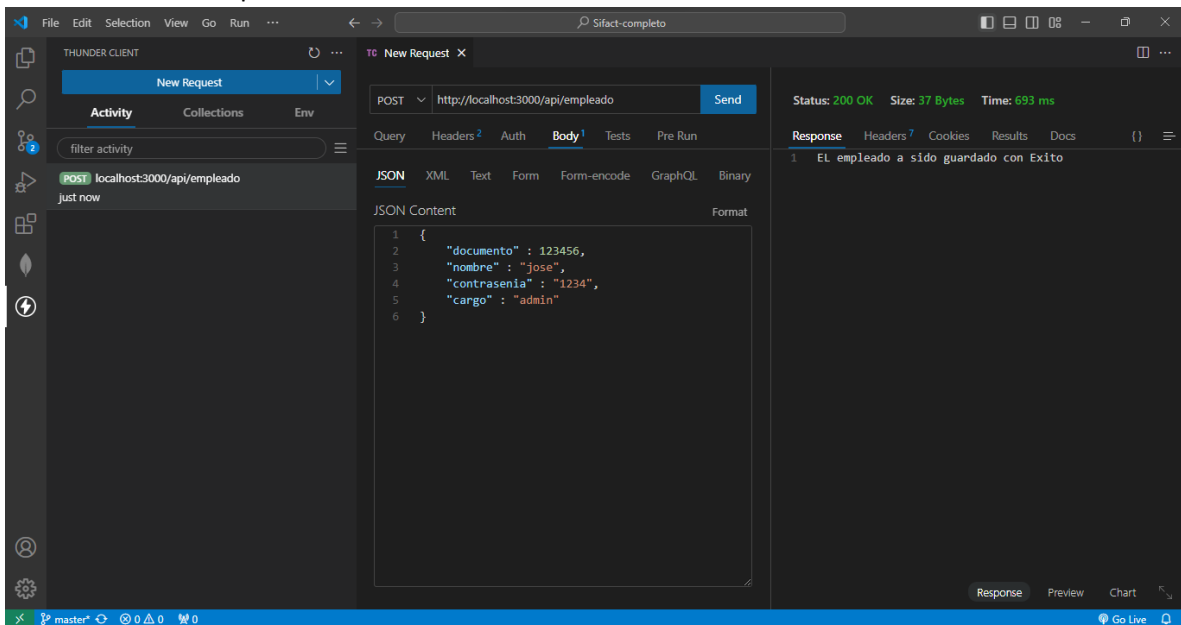
```
> server@1.0.0 dev
> nodemon server/index.js

[nodemon] 3.1.0
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server/index.js index.js`
Server activo en el puerto 3000
DB is connected
```

Go Live

Una vez nuestro servidor este corriendo pasamos a realizar las pruebas con a cada uno de los métodos creados previamente existen varias herramientas para ello nosotros usamos thunder

Prueba de crear empleado



```
File Edit Selection View Go Run ... Sifact-completo
```

THUNDER CLIENT

New Request

Activity Collections Env

filter activity

POST localhost:3000/api/empleado just now

POST http://localhost:3000/api/empleado Send

Status: 200 OK Size: 37 Bytes Time: 693 ms

Response Headers Cookies Results Docs {}

1 EL empleado a sido guardado con Exito

JSON XML Text Form Form-encode GraphQL Binary

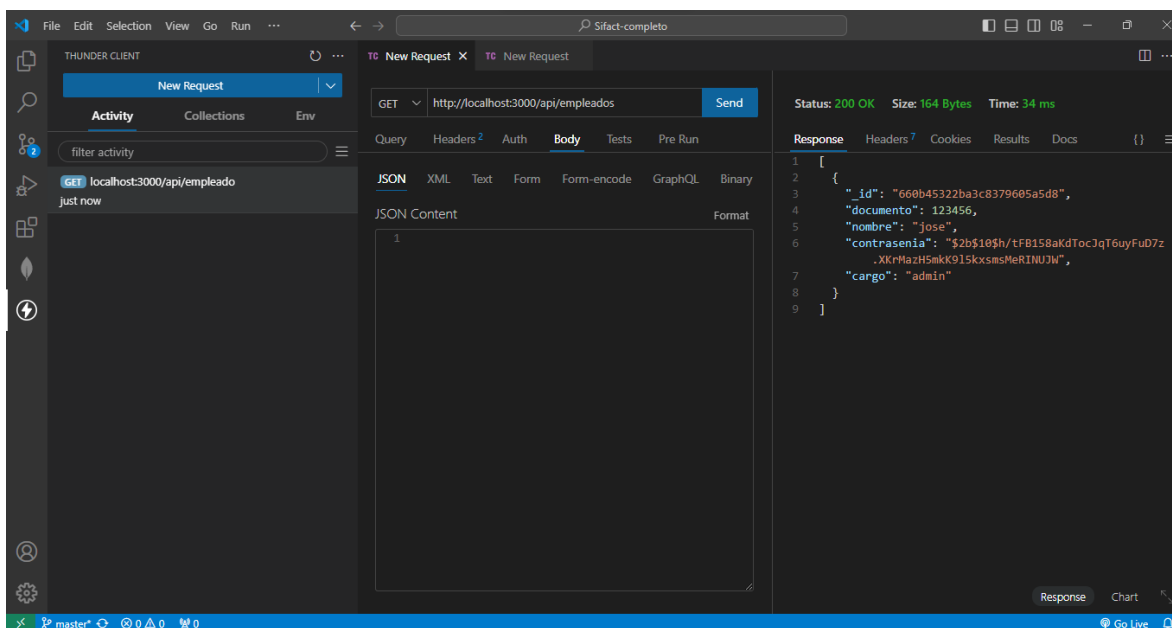
JSON Content Format

```
1 {
2   "documento" : 123456,
3   "nombre" : "jose",
4   "contrasenia" : "1234",
5   "cargo" : "admin"
6 }
```

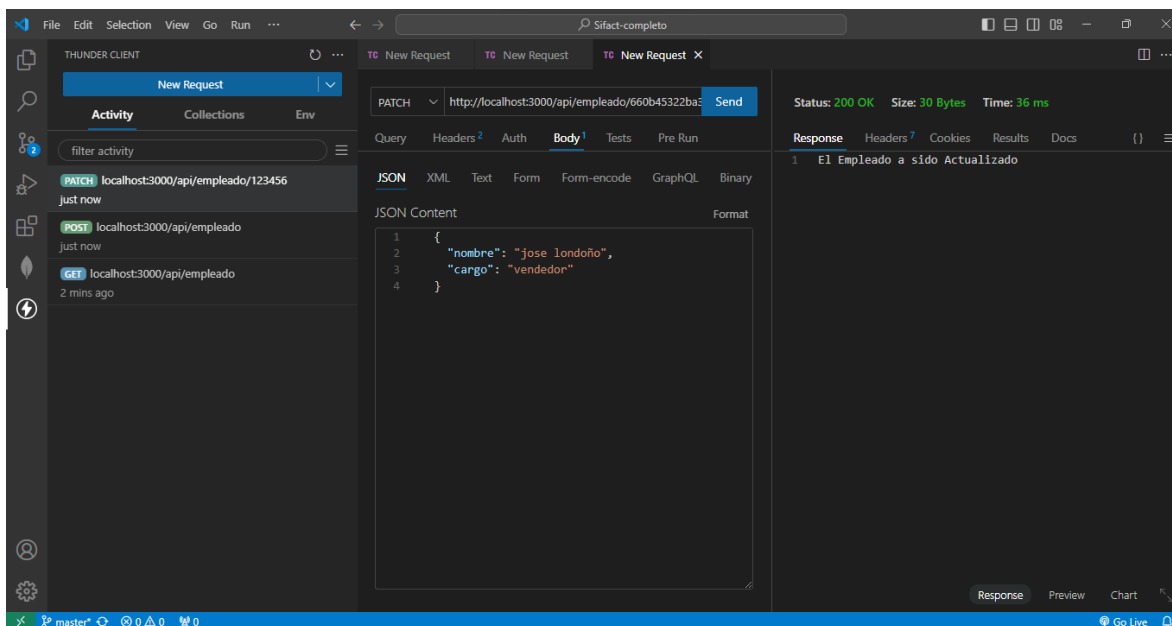
Response Preview Chart

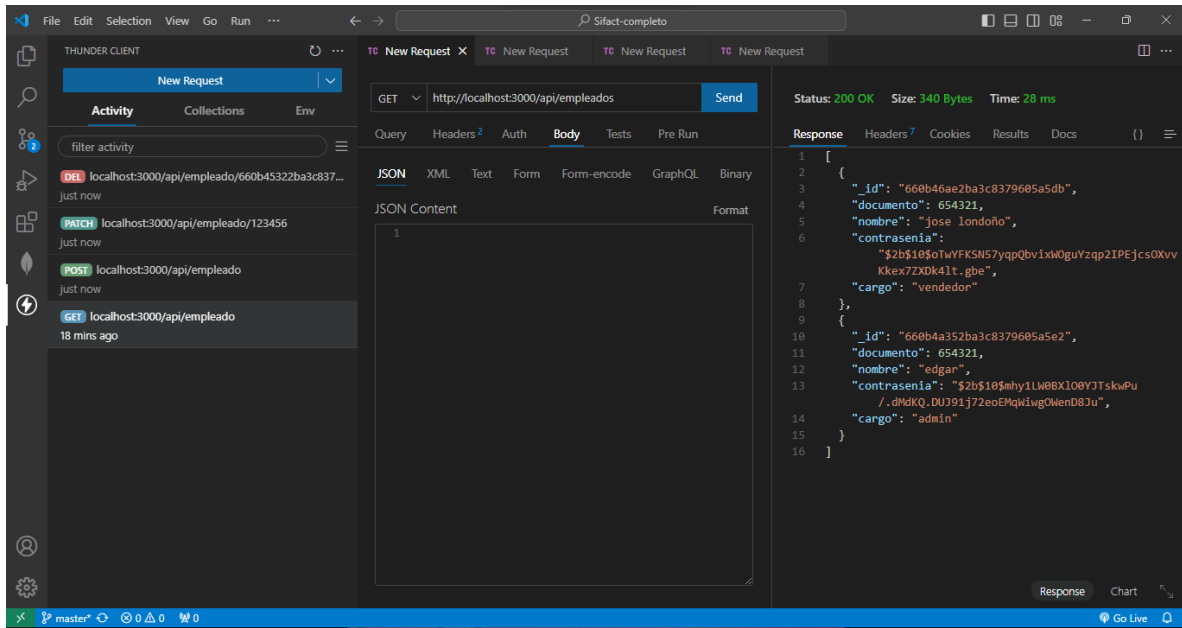
Go Live

Prueba de consultar empleados

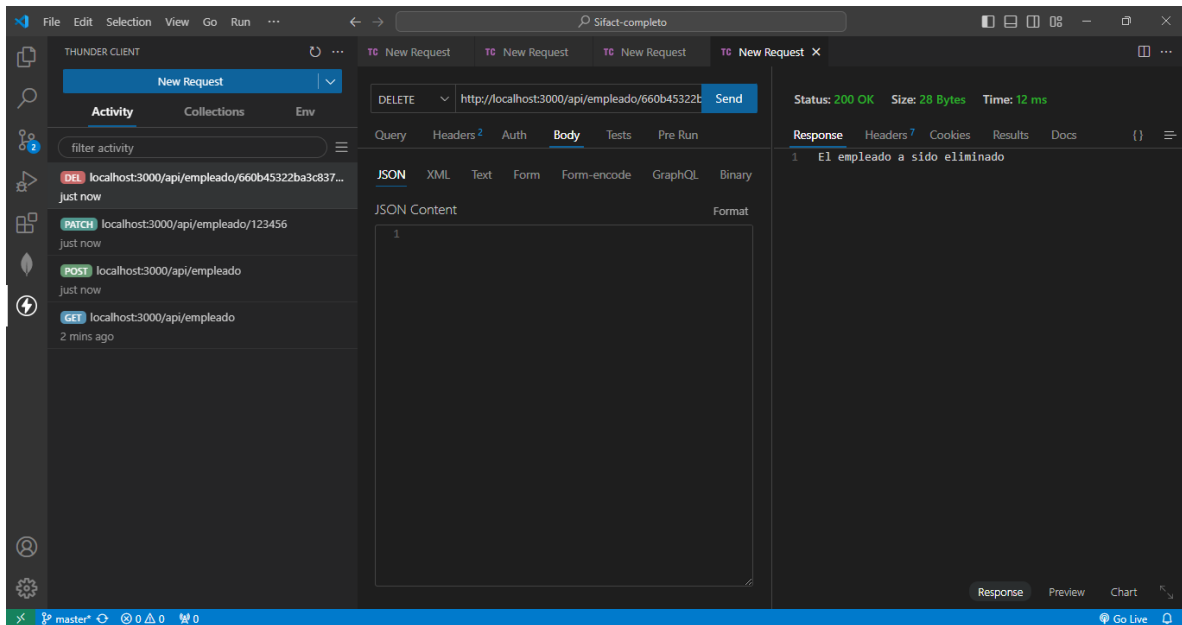


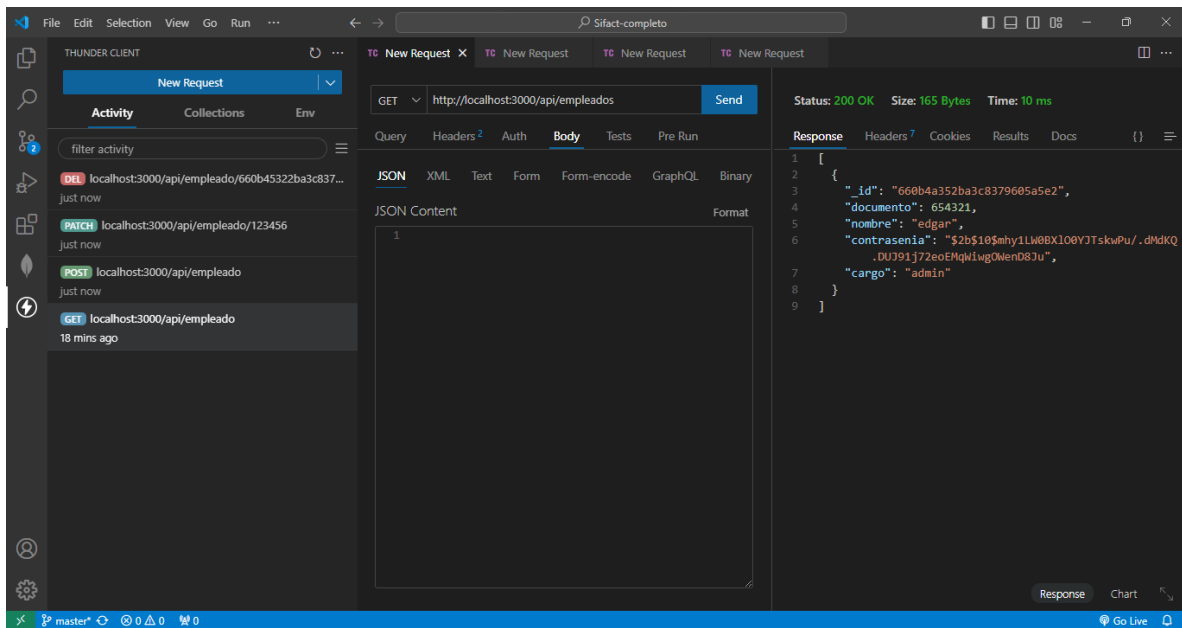
Prueba de actualizar datos de empleados





Prueba de eliminar un empleado





una vez realizadas todas las pruebas nuestro API esta lista para ser consumida por el frond end usando cada una de las rutas correspondiente según la función que se necesite