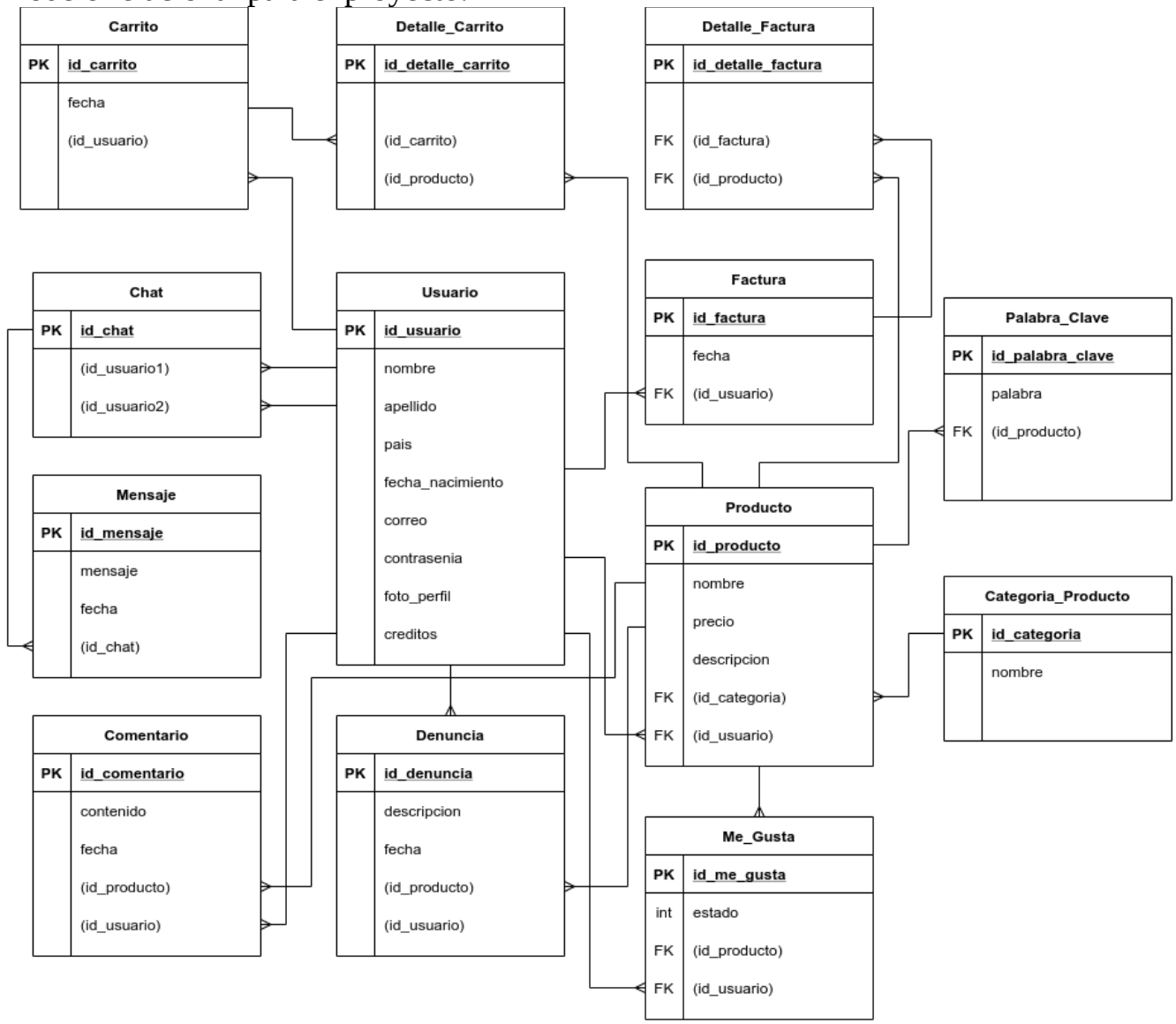
Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Manejo e implementacion de archivos
Sección A
Ing. Alvaro Diaz
Aux. Luis Hernandez

MANUAL TÉCNICO

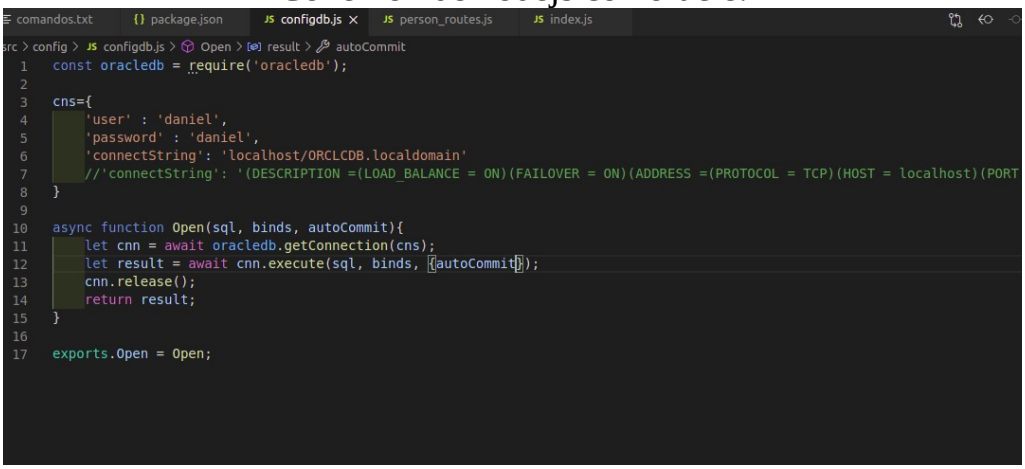
Carlos Daniel Monterroso Pacay 201602560

Guatemala 10 de Noviembre de 2020

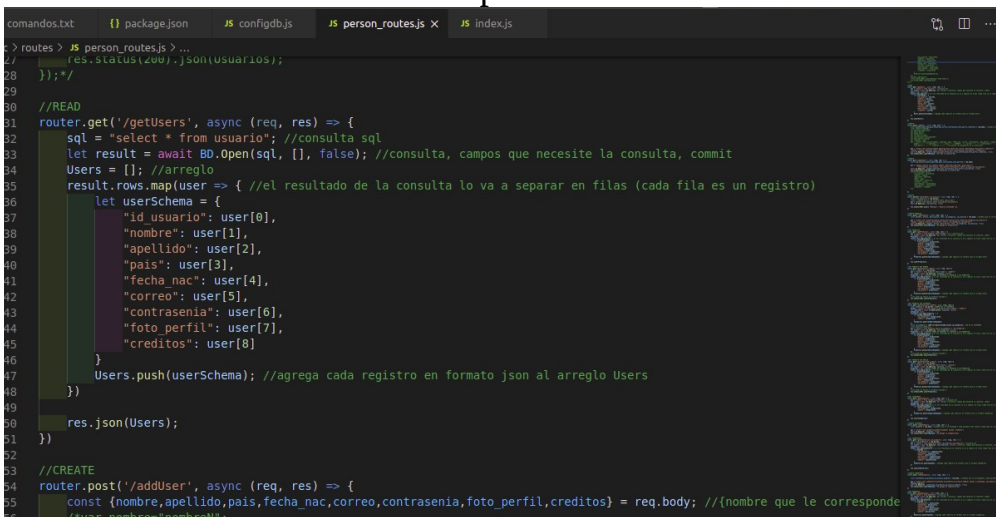
Modelo relacional para el proyecto:

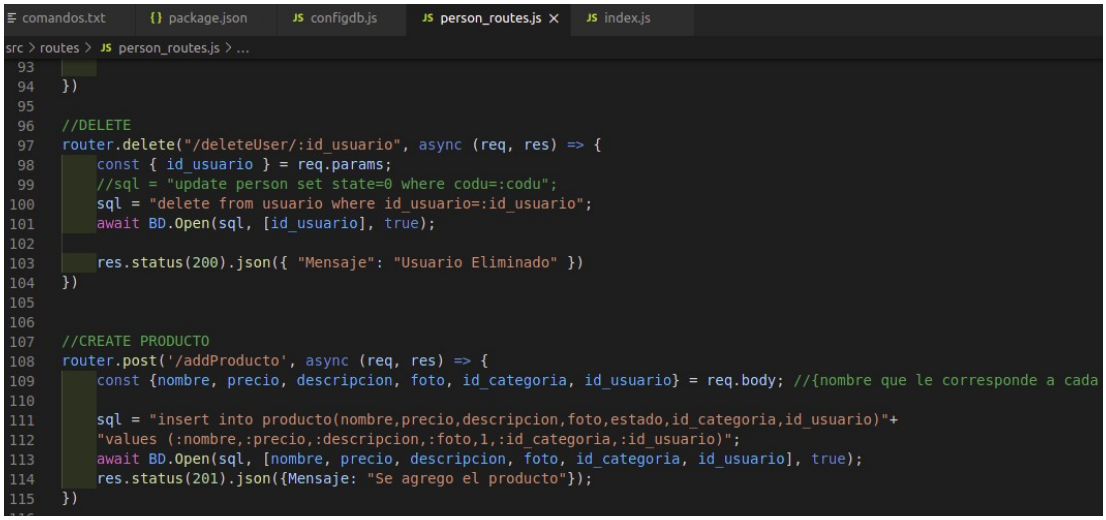


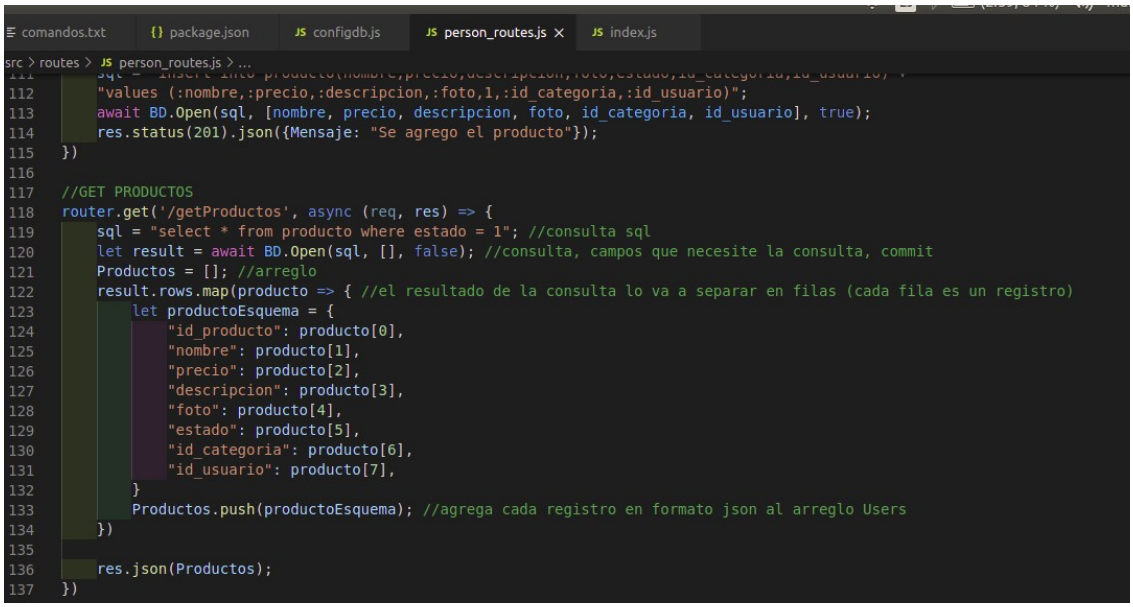
Conexion de nodejs con oracle:



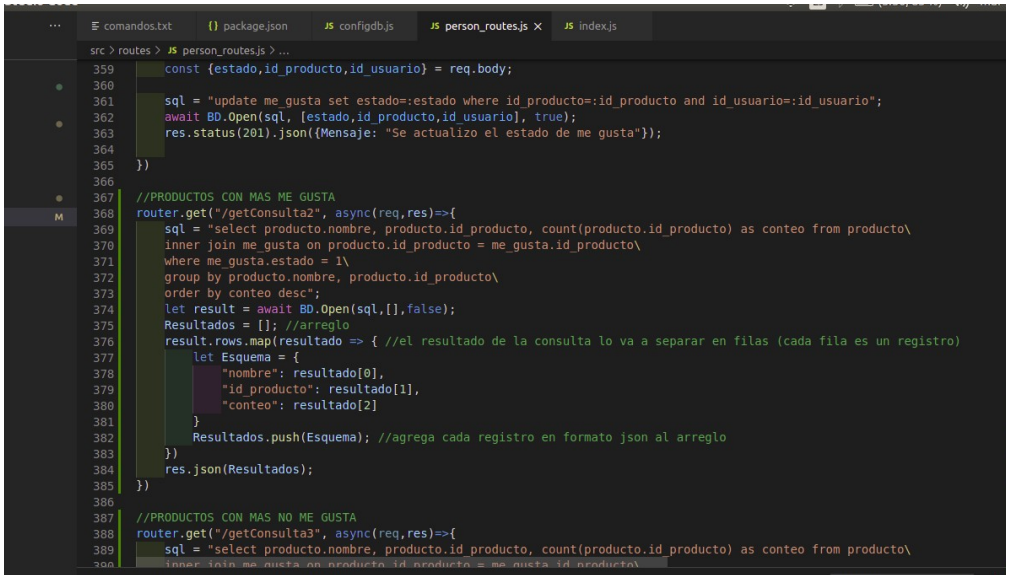
Endpoints:




comandos.txt
package.json
configdb.js
person_routes.js
index.js
src > routes > person_routes.js > ...
93
94 })
95
96 //DELETE
97 router.delete("/deleteUser/:id_usuario", async (req, res) => {
98 const { id_usuario } = req.params;
99 //sql = "update person set state=0 where codu=:codu";
100 sql = "delete from usuario where id_usuario=:id_usuario";
101 await BD.Open(sql, [id_usuario], true);
102
103 res.status(200).json({ "Mensaje": "Usuario Eliminado" })
104 })
105
106
107 //CREATE PRODUCTO
108 router.post('/addProducto', async (req, res) => {
109 const {nombre, precio, descripcion, foto, id_categoria, id_usuario} = req.body; //{nombre que le corresponde a cada
110
111 sql = "insert into producto(nombre,precio,descripcion,foto,estado,id_categoria,id_usuario)"+
112 "values (:nombre,:precio,:descripcion,:foto,1,:id_categoria,:id_usuario)";
113 await BD.Open(sql, [nombre, precio, descripcion, foto, id_categoria, id_usuario], true);
114 res.status(201).json({Mensaje: "Se agrego el producto"});
115 })

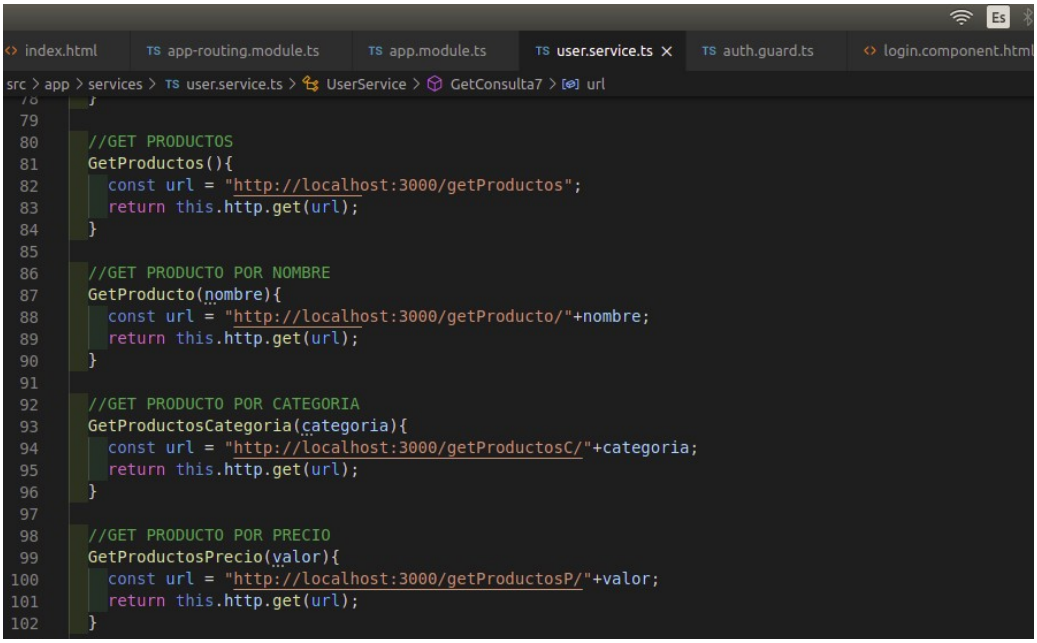

comandos.txt
package.json
configdb.js
person_routes.js
index.js
src > routes > person_routes.js > ...
112 "values (:nombre,:precio,:descripcion,:foto,1,:id_categoria,:id_usuario)";
113 await BD.Open(sql, [nombre, precio, descripcion, foto, id_categoria, id_usuario], true);
114 res.status(201).json({Mensaje: "Se agrego el producto"});
115 })
116
117 //GET PRODUCTOS
118 router.get('/getProductos', async (req, res) => {
119 sql = "select * from producto where estado = 1"; //consulta sql
120 let result = await BD.Open(sql, [], false); //consulta, campos que necesite la consulta, commit
121 Productos = []; //arreglo
122 result.rows.map(producto => { //el resultado de la consulta lo va a separar en filas (cada fila es un registro)
123 let productoEsquema = {
124 "id_producto": producto[0],
125 "nombre": producto[1],
126 "precio": producto[2],
127 "descripcion": producto[3],
128 "foto": producto[4],
129 "estado": producto[5],
130 "id_categoria": producto[6],
131 "id_usuario": producto[7],
132 }
133 Productos.push(productoEsquema); //agrega cada registro en formato json al arreglo Users
134 })
135
136 res.json(Productos);
137 })

Endpoints para los reportes:

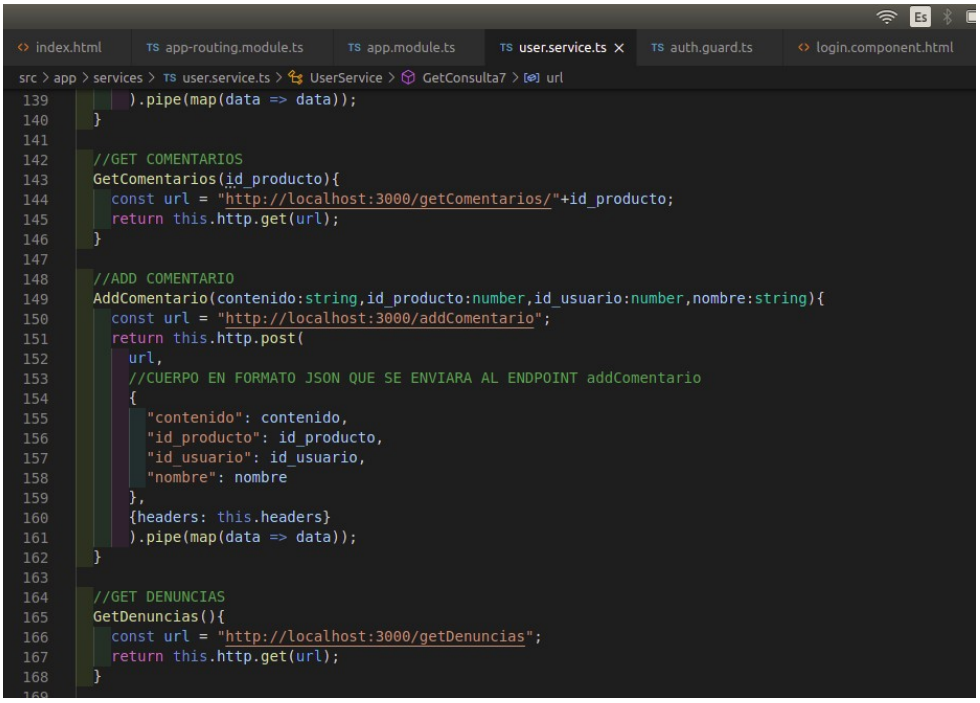


```
comandos.txt | package.json | configdb.js | person_routes.js | index.js
src > routes > person_routes.js > ...
359     const {estado,id_producto,id_usuario} = req.body;
360
361     sql = "update me_gusta set estado=:estado where id_producto=:id_producto and id_usuario=:id_usuario";
362     await BD.Open(sql, [estado,id_producto,id_usuario], true);
363     res.status(201).json({Mensaje: "Se actualizo el estado de me gusta"});
364
365 })
366
367 //PRODUCTOS CON MAS ME GUSTA
368 router.get("/getConsulta2", async(req,res)=>{
369     sql = "select producto.nombre, producto.id_producto, count(producto.id_producto) as conteo from producto\
370     inner join me_gusta on producto.id_producto = me_gusta.id_producto\
371     where me_gusta.estado = 1\
372     group by producto.nombre, producto.id_producto\
373     order by conteo desc";
374     let result = await BD.Open(sql,[],false);
375     Resultados = []; //arreglo
376     result.rows.map(resultado => { //el resultado de la consulta lo va a separar en filas (cada fila es un registro)
377         let Esquema = {
378             "nombre": resultado[0],
379             "id_producto": resultado[1],
380             "conteo": resultado[2]
381         }
382         Resultados.push(Esquema); //agrega cada registro en formato json al arreglo
383     })
384     res.json(Resultados);
385 })
386
387 //PRODUCTOS CON MAS NO ME GUSTA
388 router.get("/getConsulta3", async(req,res)=>{
389     sql = "select producto.nombre, producto.id_producto, count(producto.id_producto) as conteo from producto\
390     inner join me_gusta on producto.id_producto = me_gusta.id_producto\
391     where me_gusta.estado = 2\
392     group by producto.nombre, producto.id_producto\
393     order by conteo desc";
394     let result = await BD.Open(sql,[],false);
395     Resultados = []; //arreglo
396     result.rows.map(resultado => { //el resultado de la consulta lo va a separar en filas (cada fila es un registro)
397         let Esquema = {
398             "nombre": resultado[0],
399             "id_producto": resultado[1],
400             "conteo": resultado[2]
401         }
402         Resultados.push(Esquema); //agrega cada registro en formato json al arreglo
403     })
404     res.json(Resultados);
405 })
406
407 //CLIENTES CON MAS Y MENOS CREDITOS
408 router.get("/getConsulta4", async(req,res)=>{
409     sql = "select nombre, creditos as conteo from usuario where id_usuario<>5 order by creditos desc";
410     let result = await BD.Open(sql,[],false);
411     Resultados = []; //arreglo
412     result.rows.map(resultado => { //el resultado de la consulta lo va a separar en filas (cada fila es un registro)
413         let Esquema = {
414             "nombre": resultado[0],
415             "id_producto": 0,
```

Codigo del frontend:



```
index.html | app-routing.module.ts | app.module.ts | user.service.ts | auth.guard.ts | login.component.html
src > app > services > user.service.ts > UserService > GetConsulta7 > url
78  }
79
80  //GET PRODUCTOS
81  GetProductos(){
82    const url = "http://localhost:3000/getProductos";
83    return this.http.get(url);
84  }
85
86  //GET PRODUCTO POR NOMBRE
87  GetProducto(nombre){
88    const url = "http://localhost:3000/getProducto/"+nombre;
89    return this.http.get(url);
90  }
91
92  //GET PRODUCTO POR CATEGORIA
93  GetProductosCategoria(categoria){
94    const url = "http://localhost:3000/getProductosC/"+categoria;
95    return this.http.get(url);
96  }
97
98  //GET PRODUCTO POR PRECIO
99  GetProductosPrecio(valor){
100   const url = "http://localhost:3000/getProductosP/"+valor;
101   return this.http.get(url);
102 }
```



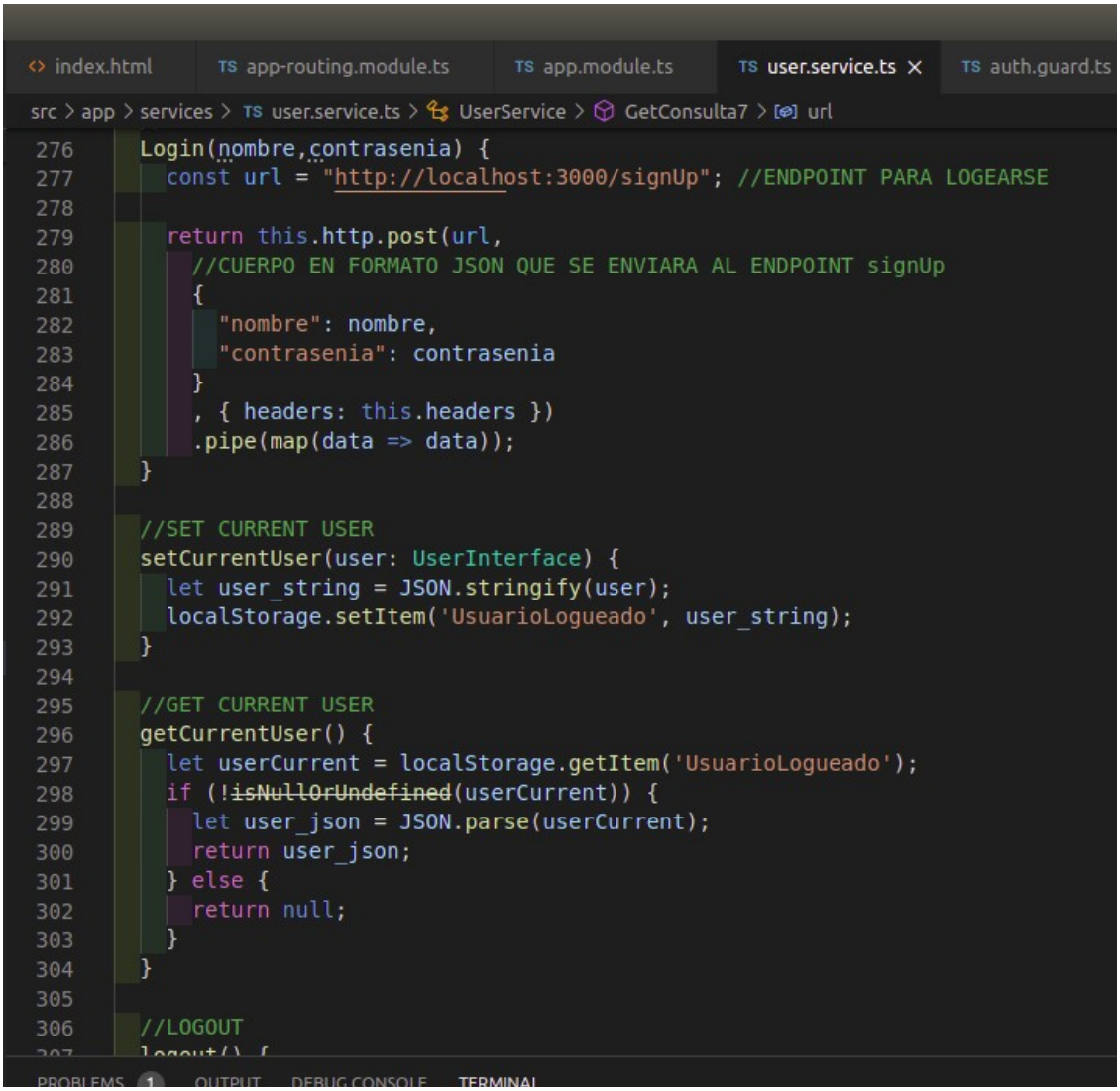
```
index.html | app-routing.module.ts | app.module.ts | user.service.ts | auth.guard.ts | login.component.html
src > app > services > user.service.ts > UserService > GetConsulta7 > url
139     ).pipe(map(data => data));
140   }
141
142   //GET COMENTARIOS
143   GetComentarios(id_producto){
144     const url = "http://localhost:3000/getComentarios/"+id_producto;
145     return this.http.get(url);
146   }
147
148   //ADD COMENTARIO
149   AddComentario(contenido:string,id_producto:number,id_usuario:number,nombre:string){
150     const url = "http://localhost:3000/addComentario";
151     return this.http.post(
152       url,
153       //CUERPO EN FORMATO JSON QUE SE ENVIARA AL ENDPOINT addComentario
154       {
155         "contenido": contenido,
156         "id_producto": id_producto,
157         "id_usuario": id_usuario,
158         "nombre": nombre
159       },
160       {headers: this.headers}
161       ).pipe(map(data => data));
162   }
163
164   //GET DENUNCIAS
165   GetDenuncias(){
166     const url = "http://localhost:3000/getDenuncias";
167     return this.http.get(url);
168   }
```

```
198
199   //ADD ME GUSTA
200   AddMeGusta(estado:number,id_producto:number,id_usuario:number){
201     const url = "http://localhost:3000/addMeGusta";
202     return this.http.post(
203       url,
204       //CUERPO EN FORMATO JSON QUE SE ENVIARA AL ENDPOINT addMeGusta
205       {
206         "estado": estado,
207         "id_producto": id_producto,
208         "id_usuario": id_usuario
209       },
210       {headers: this.headers}
211       ).pipe(map(data => data));
212   }
213   //GET ALL ME GUSTA
214   GetAllMeGusta(id_producto){
215     const url = "http://localhost:3000/getAllMeGusta/"+id_producto;
216     return this.http.get(url);
217   }
218   //GET ALL NO ME GUSTA
219   GetAllNoMeGusta(id_producto){
220     const url = "http://localhost:3000/getAllNoMeGusta/"+id_producto;
221     return this.http.get(url);
222   }
223
```

index.html | app-routing.module.ts | app.module.ts | user.service.ts | auth.guard.ts | login.component.html

rc > app > services > user.service.ts > UserService > GetConsulta7 > url

```
244   //CONSULTA 3
245   GetConsulta3(){
246     const url = "http://localhost:3000/getConsulta3";
247     return this.http.get(url);
248   }
249
250   //CONSULTA 4
251   GetConsulta4(){
252     const url = "http://localhost:3000/getConsulta4";
253     return this.http.get(url);
254   }
255
256   //CONSULTA 5
257   GetConsulta5(){
258     const url = "http://localhost:3000/getConsulta5";
259     return this.http.get(url);
260   }
261
262   //CONSULTA 6
263   GetConsulta6(){
264     const url = "http://localhost:3000/getConsulta6";
265     return this.http.get(url);
266   }
267
268   //CONSULTA 7
269   GetConsulta7(){
270     const url = "http://localhost:3000/getConsulta7";
271     return this.http.get(url);
272   }
273
274
```


index.html
app-routing.module.ts
app.module.ts
user.service.ts
auth.guard.ts
src > app > services > user.service.ts > UserService > GetConsulta7 > url
Login(nombre,contrasenia) {
const url = "http://localhost:3000/signUp"; //ENDPOINT PARA LOGEARSE
return this.http.post(url,
//CUERPO EN FORMATO JSON QUE SE ENVIARA AL ENDPOINT signUp
{
"nombre": nombre,
"contrasenia": contrasenia
}
, { headers: this.headers })
.pipe(map(data => data));
}
//SET CURRENT USER
setCurrentUser(user: UserInterface) {
let user_string = JSON.stringify(user);
localStorage.setItem('UsuarioLogueado', user_string);
}
//GET CURRENT USER
getCurrentUser() {
let userCurrent = localStorage.getItem('UsuarioLogueado');
if (!isNullOrUndefined(userCurrent)) {
let user_json = JSON.parse(userCurrent);
return user_json;
} else {
return null;
}
}
//LOGOUT
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL