

Continuación FrontEnd.

Crear servicio de Autenticación

Primeramente ,agregar módulo [MatSnackBarModule](#) al archivo **material.module.ts** para usar el servicio de mensajes; modificar el archivo **auth.service.ts**

Abrir terminal y ejecutar el siguiente comando para crear el servicio de autenticación: **ng g s shared/services/auth**

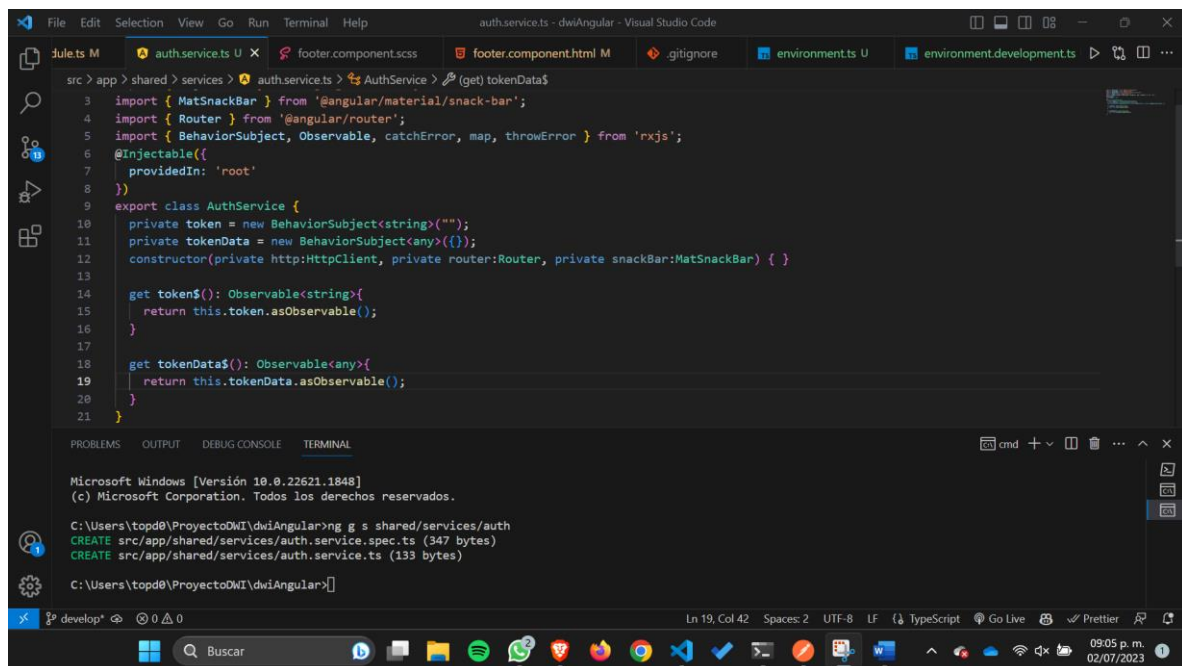
Agregamos la siguiente instrucción para ocupar elementos a utilizar para el manejo del Token y datos de usuario

Agregar las propiedades a la clase AuthService:

token Es el token encriptado

tokenData Datos de usuario encriptado qué es el PayLoad

Agregar las siguientes instrucciones al archivo **auth.service.ts** con el objetivo de definir las propiedades **token** y **tokenData** como Observables y permitir suscripciones y saber de sus valores



The screenshot shows the Visual Studio Code interface with the **auth.service.ts** file open. The code defines the **AuthService** class with **token** and **tokenData** properties and methods to return them as Observables. The terminal at the bottom shows the command **ng g s shared/services/auth** being executed, resulting in the creation of **src/app/shared/services/auth.service.spec.ts** and **src/app/shared/services/auth.service.ts**.

```
src > app > shared > services > auth.service.ts > AuthService > (get) tokenData$
3 import { MatSnackBar } from '@angular/material/snack-bar';
4 import { Router } from '@angular/router';
5 import { BehaviorSubject, Observable, catchError, map, throwError } from 'rxjs';
6 @Injectable({
7   providedIn: 'root'
8 })
9 export class AuthService {
10   private token = new BehaviorSubject<string>('');
11   private tokenData = new BehaviorSubject<any>({});
12   constructor(private http:HttpClient, private router:Router, private snackBar:MatSnackBar) { }
13
14   get token$(): Observable<string>{
15     return this.token.asObservable();
16   }
17
18   get tokenData$(): Observable<any>{
19     return this.tokenData.asObservable();
20   }
21 }
```

Microsoft Windows [Versión 10.0.22621.1848]
(c) Microsoft Corporation. Todos los derechos reservados.

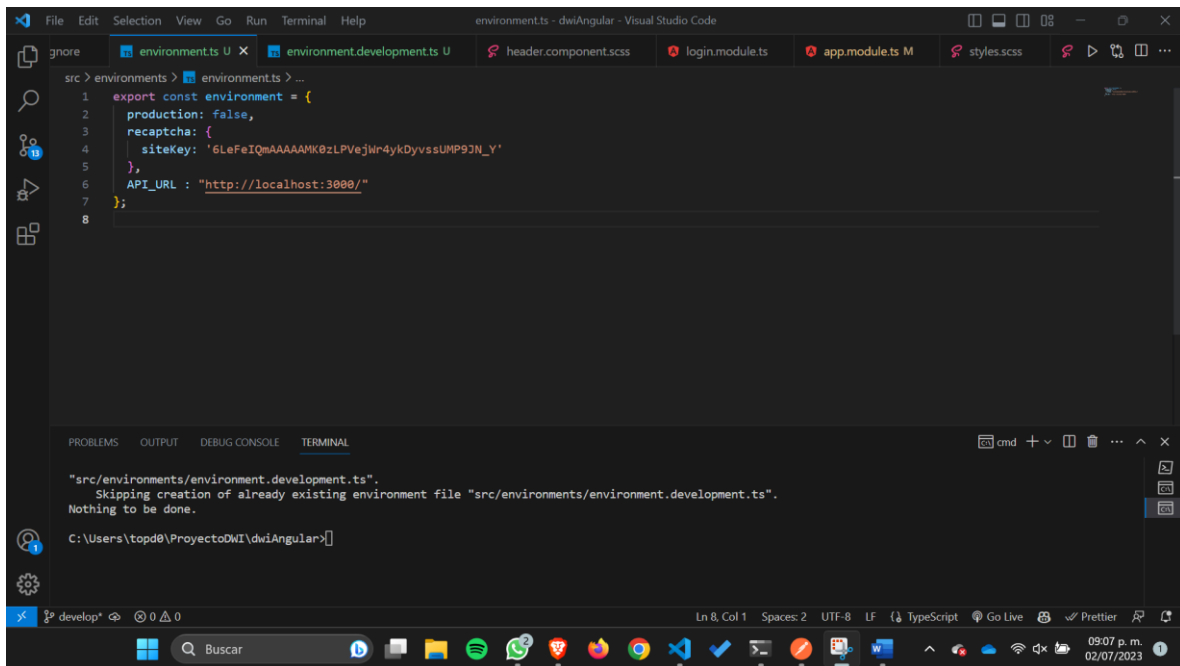
C:\Users\topd8\ProyectoDWI\dwAngular>ng g s shared/services/auth
CREATE src/app/shared/services/auth.service.spec.ts (347 bytes)
CREATE src/app/shared/services/auth.service.ts (133 bytes)

C:\Users\topd8\ProyectoDWI\dwAngular>

el utilizar environments en tu aplicación es con el objetivo de utilizar API KEY para entornos de pruebas y de producción durante el desarrollo de la aplicación.

En la terminal ejecutar el siguiente comando: **ng g environments**

Modificar los archivos environments de la siguiente forma



The screenshot shows the Visual Studio Code editor with the file `environment.ts` open. The code defines an `environment` object with `production` set to `false`, a `recaptcha` object containing a `siteKey`, and an `API_URL` pointing to `http://localhost:3000/`. The terminal at the bottom shows a message indicating that the environment file `src/environments/environment.development.ts` was skipped because it already exists. The status bar at the bottom indicates the file is in `UTF-8` encoding with `LF` line endings.

```
src > environments > environment.ts > ...
1 export const environment = {
2   production: false,
3   recaptcha: {
4     siteKey: '6LeFeIQmAAAAAMK0zLPVeJW4ykDyvssUMP9JN_Y'
5   },
6   API_URL : "http://localhost:3000/"
7 };
8
```

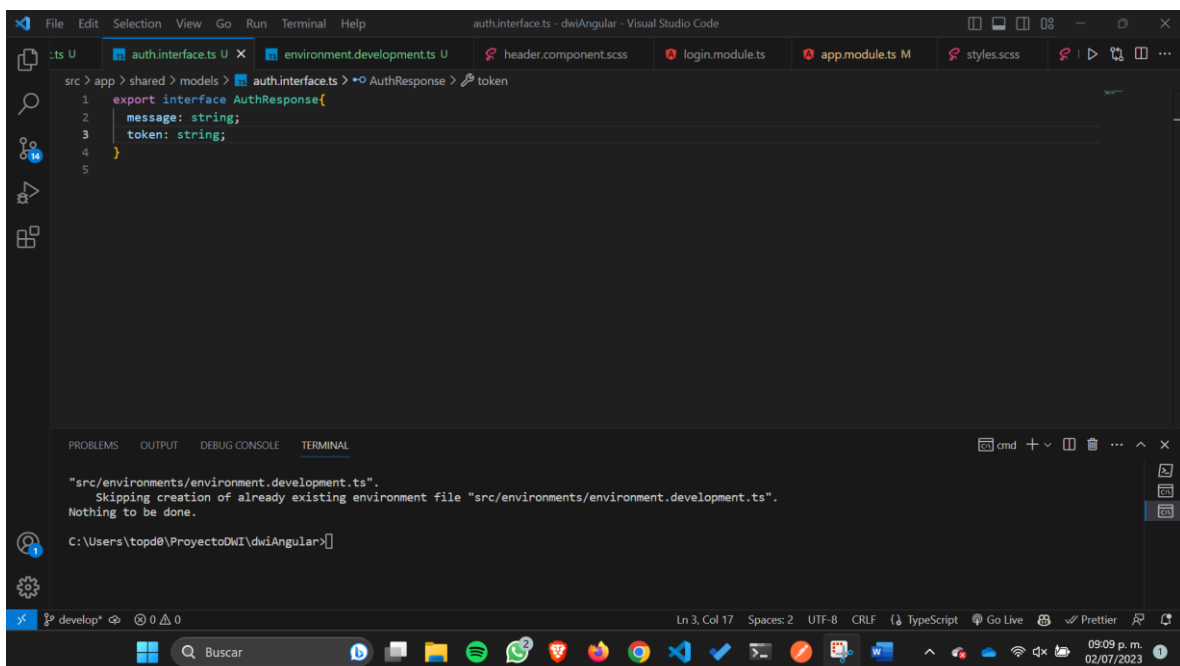
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

"src/environments/environment.development.ts".
Skipping creation of already existing environment file "src/environments/environment.development.ts".
Nothing to be done.

C:\Users\topd0\ProyectoDWI\dwAngular>

En carpeta **shared/models** agregar un archivo con el nombre **auth.interface.ts**, en caso que no exista se deberá de crear

Modificar el archivo **auth.interface.ts**



The screenshot shows the Visual Studio Code editor with the file `auth.interface.ts` open. The code defines an `AuthResponse` interface with `message` and `token` properties. The terminal at the bottom shows the same message as the previous screenshot, indicating that the environment file was skipped. The status bar at the bottom indicates the file is in `UTF-8` encoding with `CRLF` line endings.

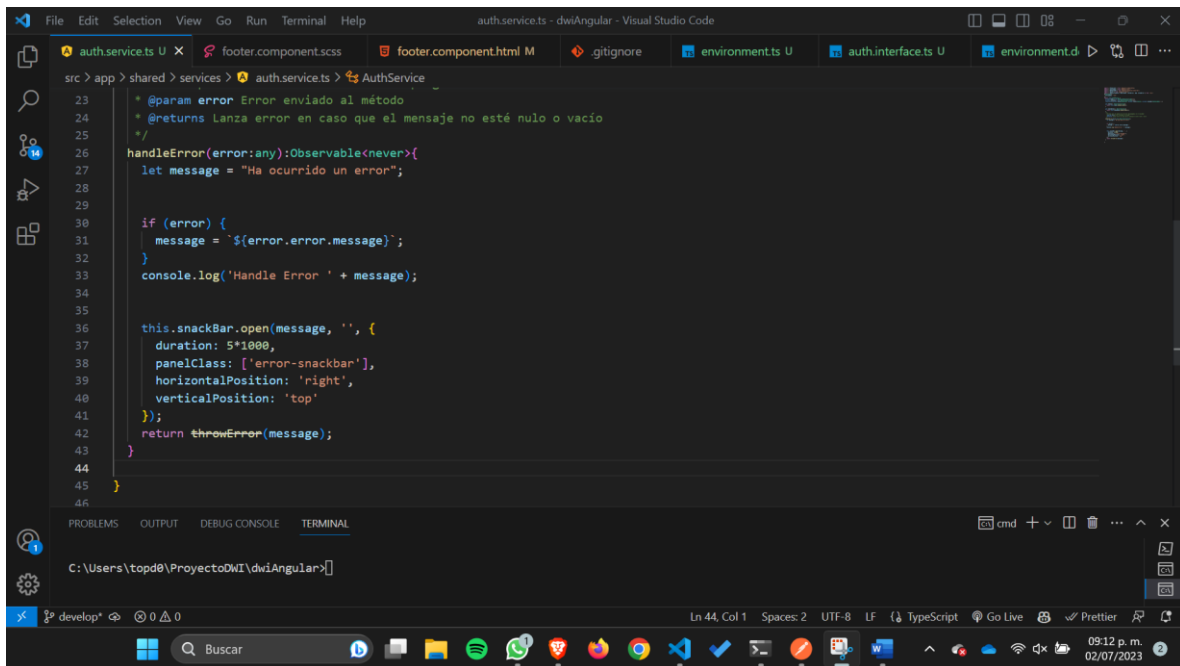
```
src > app > shared > models > auth.interface.ts > AuthResponse > token
1 export interface AuthResponse{
2   message: string;
3   token: string;
4 }
5
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

"src/environments/environment.development.ts".
Skipping creation of already existing environment file "src/environments/environment.development.ts".
Nothing to be done.

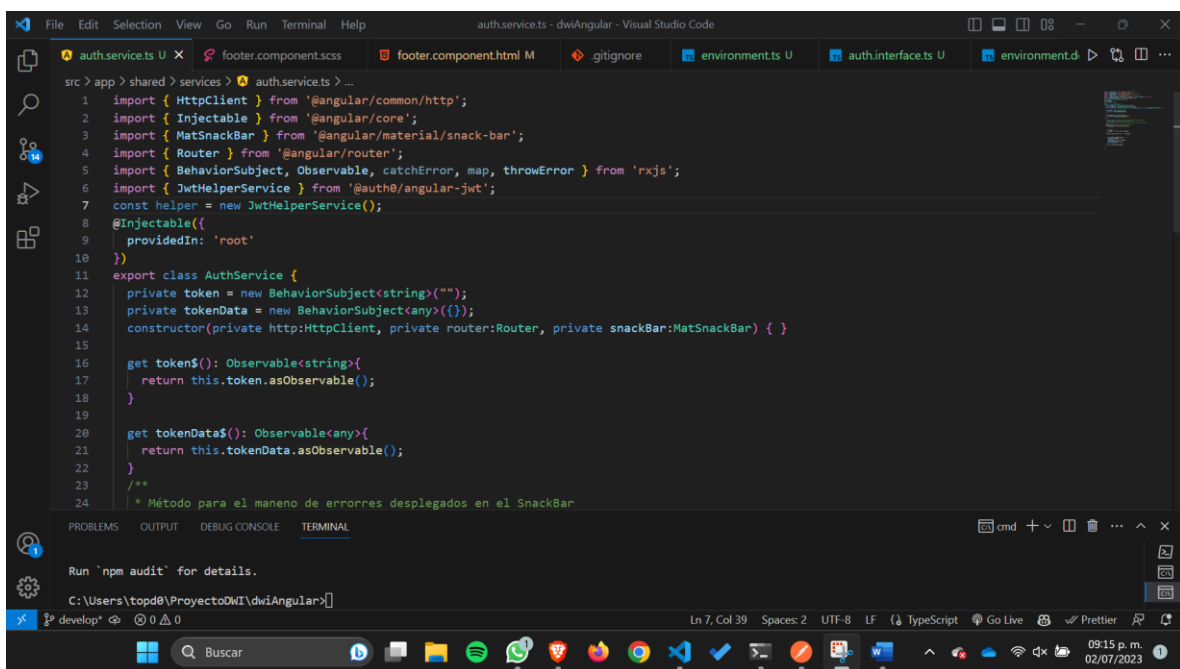
C:\Users\topd0\ProyectoDWI\dwAngular>

Agregar el siguiente método `handleError` al archivo **auth.service.ts**



```
src > app > shared > services > auth.service.ts > AuthService
23
24 * @param error Error enviado al método
25 * @returns Lanza error en caso que el mensaje no esté nulo o vacío
26 */
27 handleError(error:any):Observable<never>{
28   let message = "Ha ocurrido un error";
29
30   if (error) {
31     message = `${error.error.message}`;
32   }
33   console.log('Handle Error ' + message);
34
35   this.snackBar.open(message, '', {
36     duration: 5*1000,
37     panelClass: ['error-snackbar'],
38     horizontalPosition: 'right',
39     verticalPosition: 'top'
40   });
41   return throwError(message);
42 }
43
44
45
46
```

Agregar una variable llamada helper en auth.service.ts



```
src > app > shared > services > auth.service.ts > ...
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { MatSnackBar } from '@angular/material/snack-bar';
4 import { Router } from '@angular/router';
5 import { BehaviorSubject, Observable, catchError, map, throwError } from 'rxjs';
6 import { JwtHelperService } from '@auth0/angular-jwt';
7 const helper = new JwtHelperService();
8
9 @Injectable({
10   providedIn: 'root'
11 })
12 export class AuthService {
13   private token = new BehaviorSubject<string>("");
14   private tokenData = new BehaviorSubject<any>({});
15   constructor(private http:HttpClient, private router:Router, private snackBar:MatSnackBar) {}
16
17   get token$(): Observable<string>{
18     return this.token.asObservable();
19   }
20
21   get tokenData$(): Observable<any>{
22     return this.tokenData.asObservable();
23   }
24   /**
25    * Método para el manejo de errores desplegados en el SnackBar
26    */
27 }
```

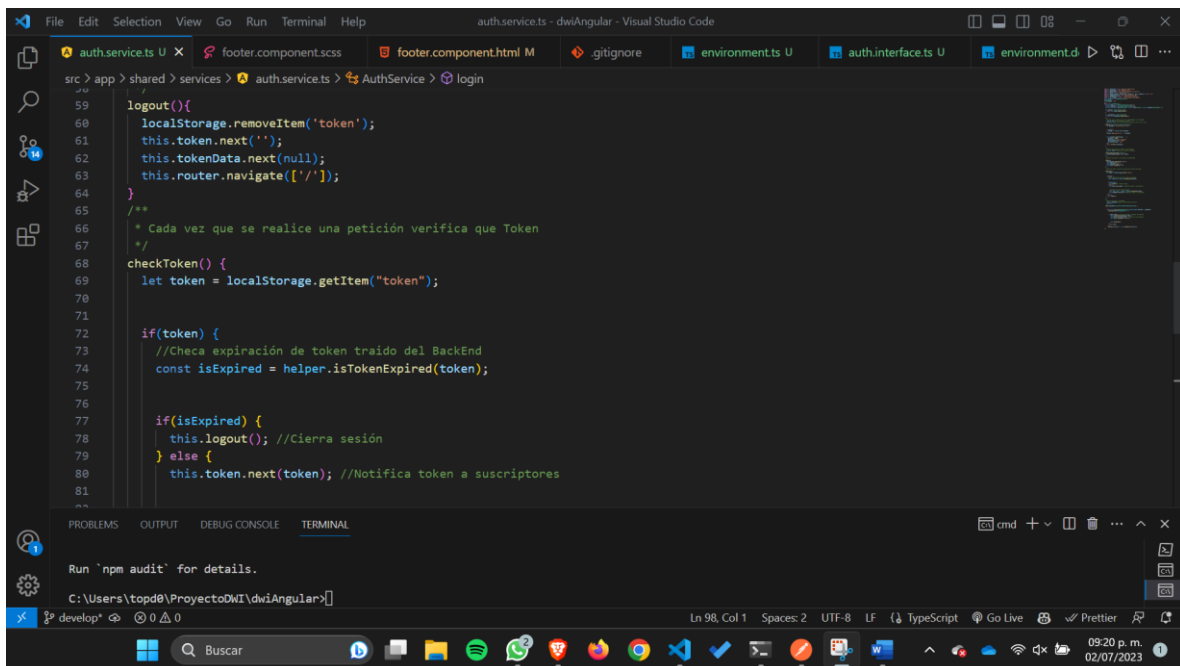
LocalStorage

El LocalStorage permite almacenar datos en el navegador web. Y que estos persistan y estén disponibles durante la navegación en la aplicación web, hasta que esta información sea borrada del navegador.

Agregar el método **saveLocalStorage** en archivo **auth.service.ts**

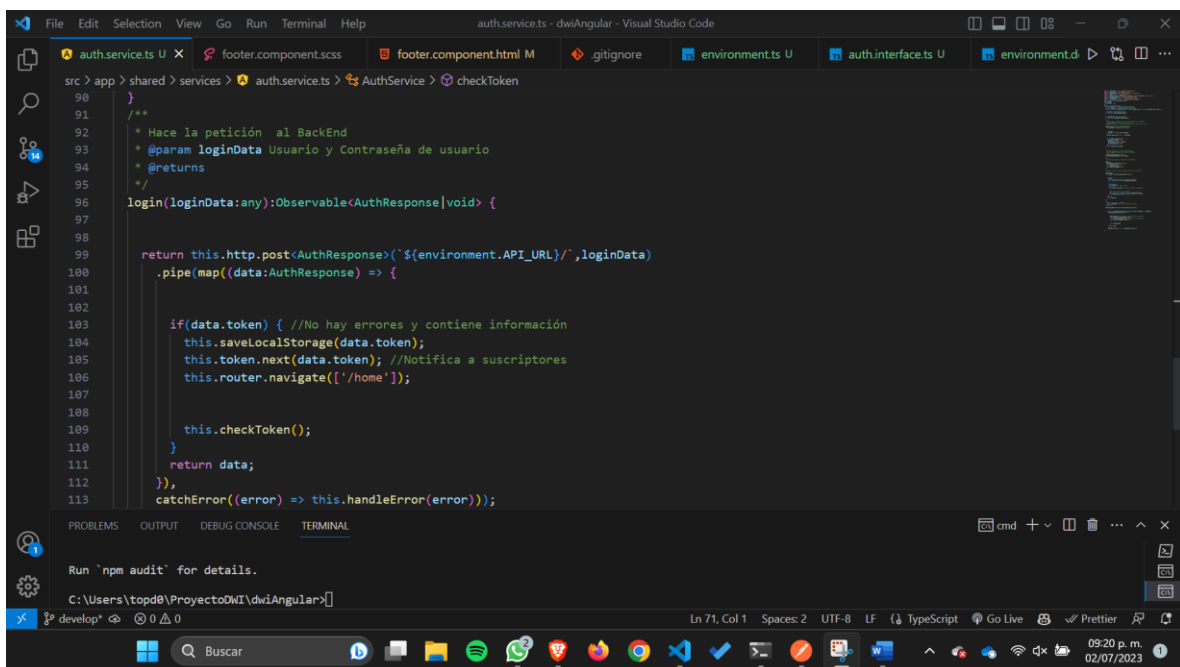
Agregar el método **logout** en archivo **auth.service.ts**

Agregar el método **checklogin** en archivo **auth.service.ts**



```
59  logout(){
60      localStorage.removeItem('token');
61      this.token.next('');
62      this.tokenData.next(null);
63      this.router.navigate(['/']);
64  }
65  /**
66   * Cada vez que se realice una petición verifica que Token
67   */
68  checkToken() {
69      let token = localStorage.getItem("token");
70
71
72      if(token) {
73          //Checa expiración de token traído del BackEnd
74          const isExpired = helper.isTokenExpired(token);
75
76
77          if(isExpired) {
78              this.logout(); //Cierra sesión
79          } else {
80              this.token.next(token); //Notifica token a suscriptores
81          }
82      }
```

Modificar método login de la siguiente forma:



```
90  }
91  /**
92   * Hace la petición al BackEnd
93   * @param loginData Usuario y Contraseña de usuario
94   * @returns
95   */
96  login(loginData:any):Observable<AuthResponse|void> {
97
98
99
100      return this.http.post<AuthResponse>(`${environment.API_URL}/`,loginData)
101          .pipe(map((data:AuthResponse) => {
102
103              if(data.token) { //No hay errores y contiene información
104                  this.saveLocalStorage(data.token);
105                  this.token.next(data.token); //Notifica a suscriptores
106                  this.router.navigate(['/home']);
107
108
109                  this.checkToken();
110              }
111              return data;
112          })),
113      catchError((error) => this.handleError(error));
```

Modificar el archivo **header.component.scss**

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the project structure. The file `header.component.scss` is selected. The main editor area shows the following SCSS code:

```
src > app > shared > components > header > header.component.scss > .spacer
1  .spacer{
2    flex:1 1 auto;
3  }
4
```

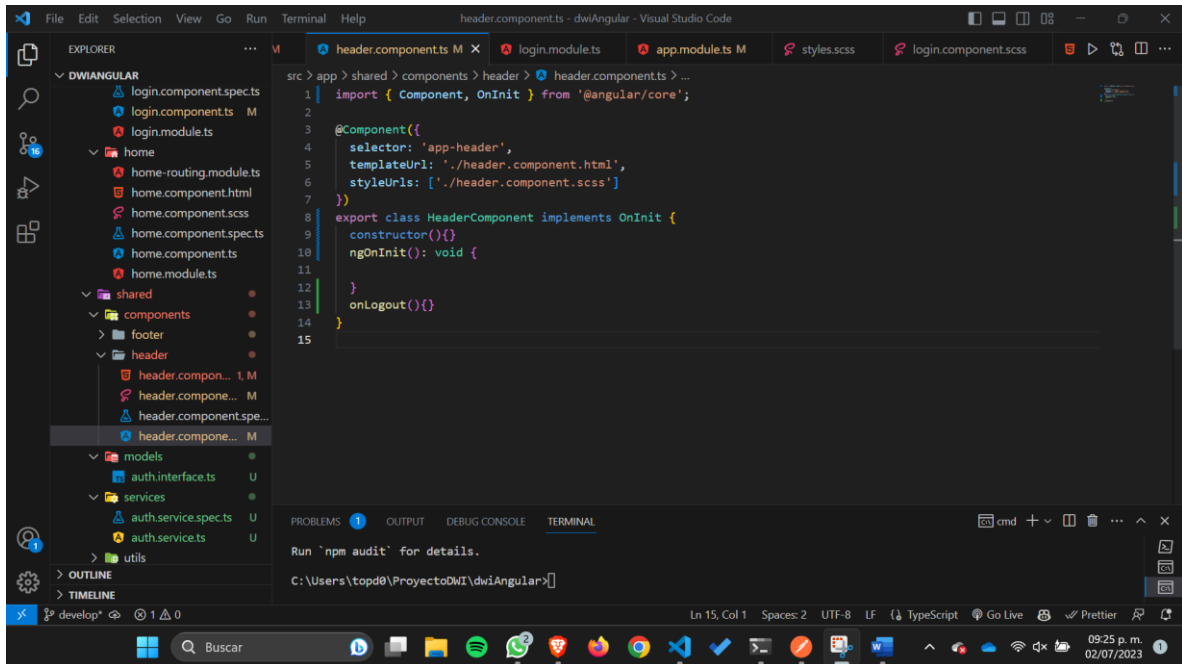
The bottom panel shows the terminal with the command `Run 'npm audit' for details.` and the path `C:\Users\topd0\ProyectoDWI\dwAngular>`. The status bar at the bottom indicates the file is at line 2, column 17, with 2 spaces, in UTF-8 encoding, using CRLF line endings.

The screenshot shows the Visual Studio Code editor with the file explorer on the left. The file `header.component.html` is selected. The main editor area shows the following HTML code:

```
src > app > shared > components > header > header.component.html > ...
Go to component
1  <mat-toolbar>
2    <span>Desarrollo Web Integral</span>
3    <span class="spacer"></span>
4
5    <button mat-button>{{data?.username}}</button>
6    <button mat-icon-button (click)="onLogout()">
7      <mat-icon>login</mat-icon>
8    </button>
9  </mat-toolbar>
10
```

The bottom panel shows the terminal with the command `Run 'npm audit' for details.` and the path `C:\Users\topd0\ProyectoDWI\dwAngular>`. The status bar at the bottom indicates the file is at line 10, column 1, with 2 spaces, in UTF-8 encoding, using CRLF line endings.

Modificar el archivo **header.component.ts**



Modificar el archivo **auth.service.ts** y agregar al constructor la siguiente línea:

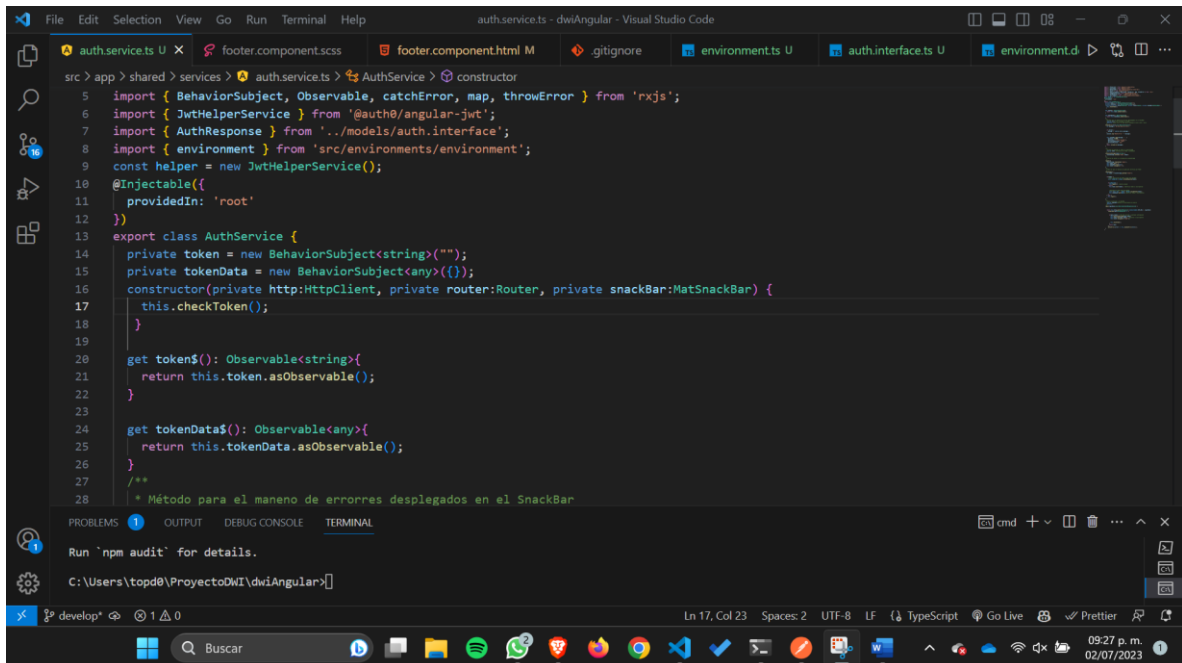
```

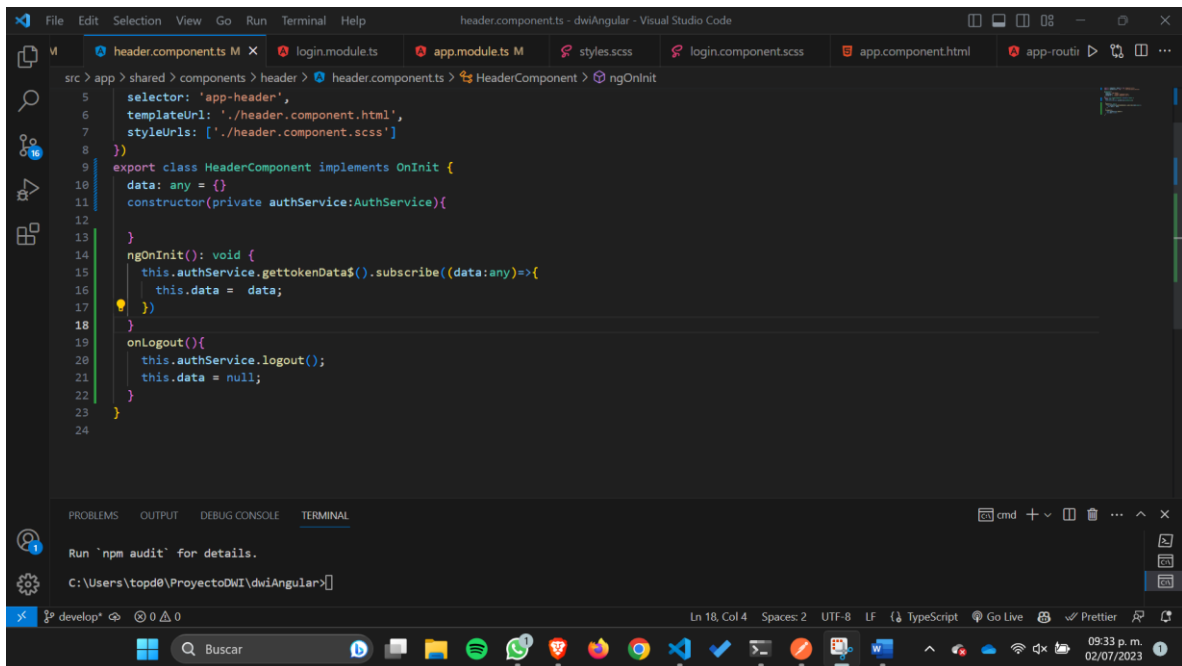
this.checkToken();

```

Modificar el archivo **header.component.ts**

Modificar el archivo **header.component.html**





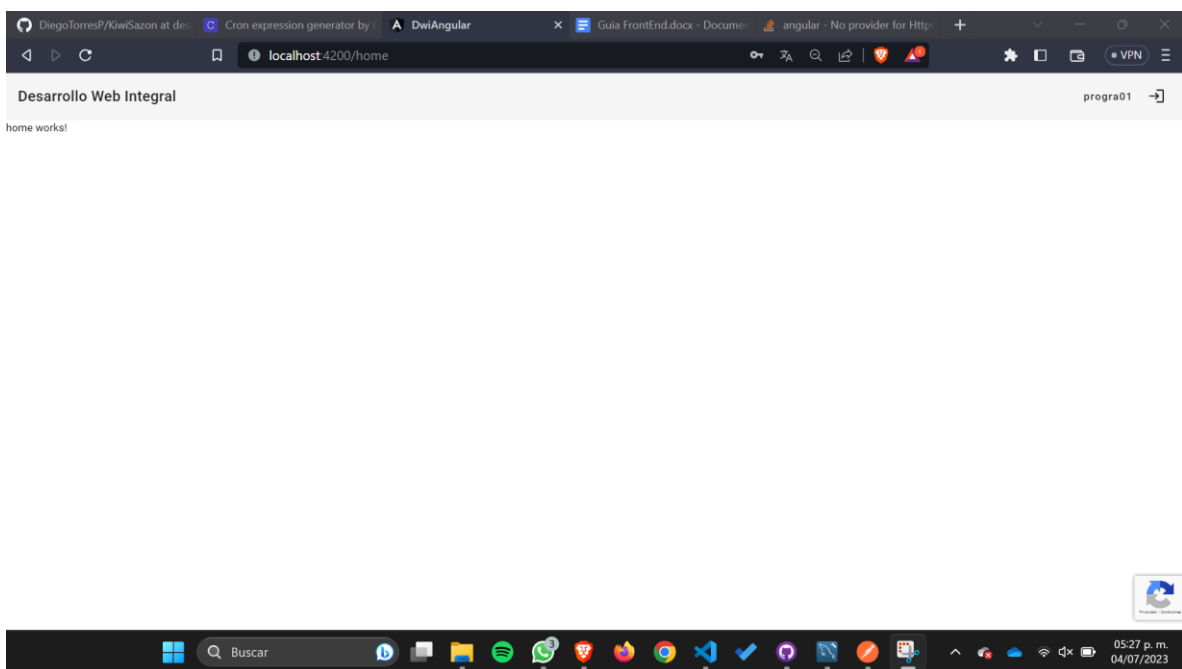
```
src > app > shared > components > header > HeaderComponent > ngOnInit
5 selector: 'app-header',
6 templateUrl: './header.component.html',
7 styleUrls: ['./header.component.scss']
8 })
9 export class HeaderComponent implements OnInit {
10   data: any = {}
11   constructor(private authService:AuthService){
12   }
13 }
14 ngOnInit(): void {
15   this.authService.getTokenData$.subscribe((data:any)=>{
16     this.data = data;
17   })
18 }
19 onLogout(){
20   this.authService.logout();
21   this.data = null;
22 }
23 }
24 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Run 'npm audit' for details.

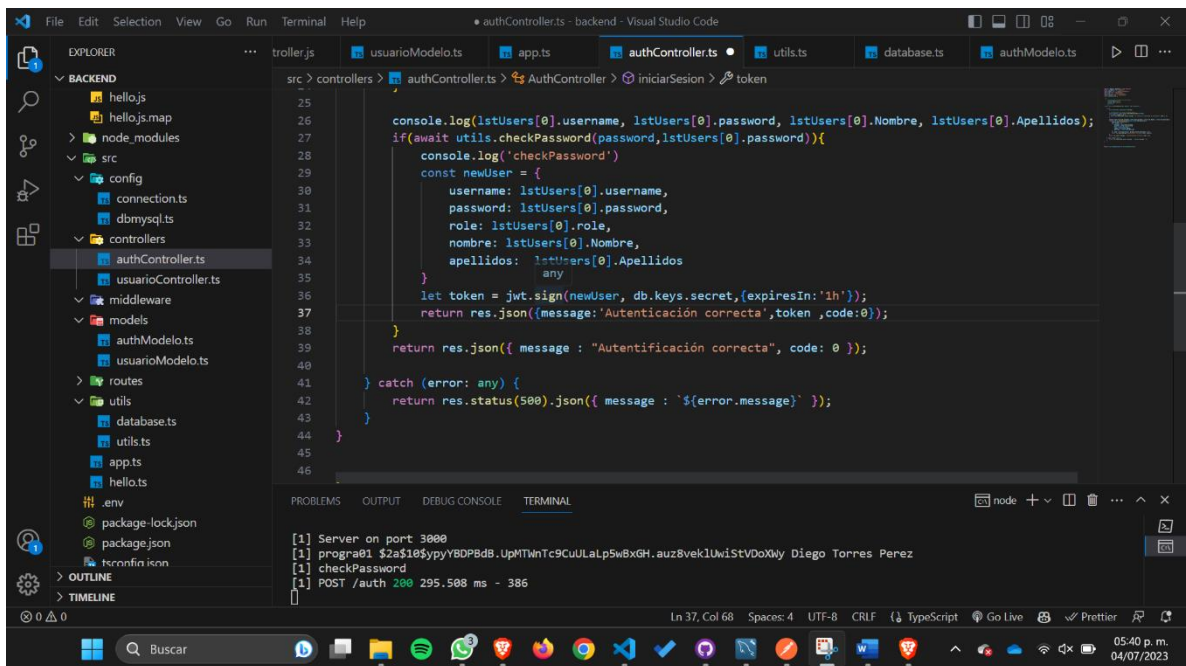
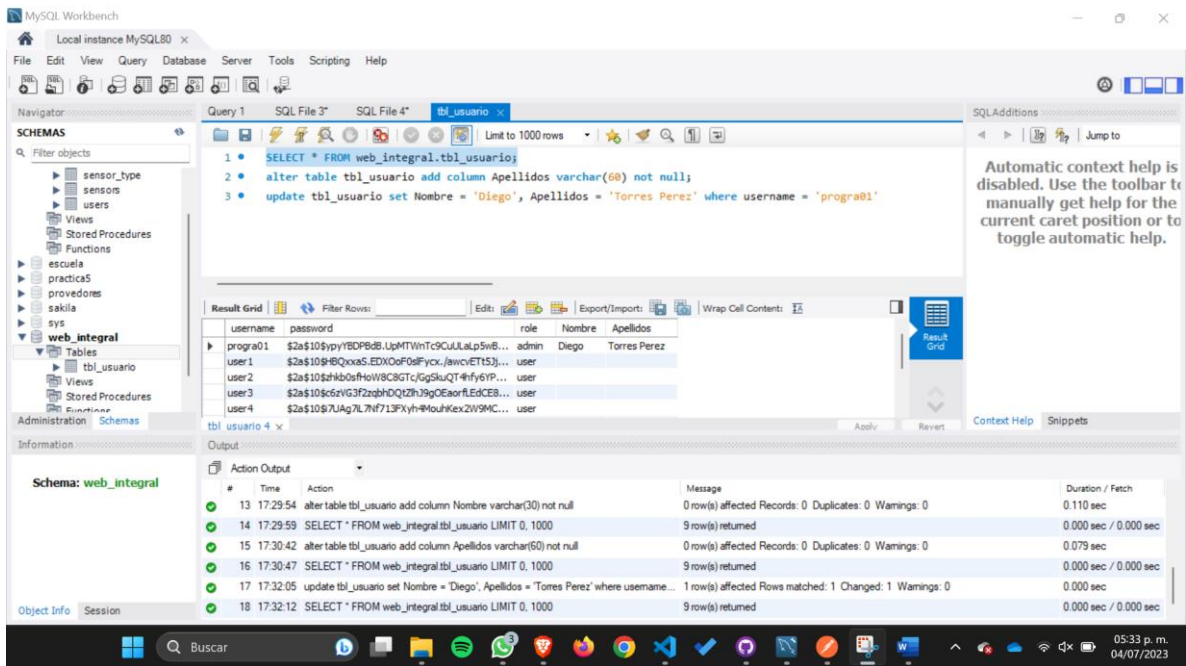
C:\Users\topdb\ProyectoDWI\dwiaAngular>

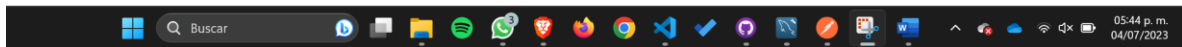
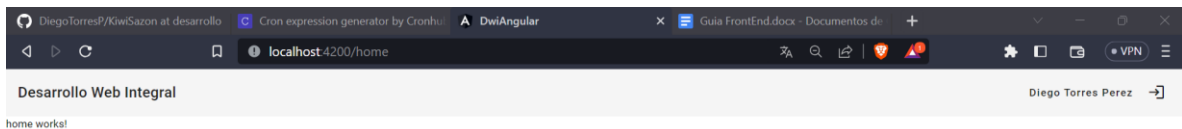
Al finalizar se mostrará de la siguiente manera



Desafío

- Agregar más campos a la tabla tbl_usuario: nombre, apellidos.
- Una vez que está logueado imprimir su nombre completo





Protección de rutas

