

1. Crear nuevo Proyecto de Angular
 - a. En Consola: ng new "nombreProyecto"
2. Levantar proyecto (dentro de carpeta del Proyecto)
 - a. En Consola: ng serve
3. Colocar en src/environments/environment.ts las urls de APIs

```
// The list of file replacements can be found
// here: https://angular.io/guide/environment-variables

export const environment = {
  production: false,
  urlAPI: 'http://localhost:3000/api/'
};
```

- a.
4. En src/index.html importar Bootstrap y otras librerías

```
FrontendAngular > src > index.html > @ html > @ head > @ script
1 <!doctype html>
2 <html lang="en">
3 <head>
4 <meta charset="utf-8">
5 <title>FrontendAngular</title>
6 <meta name="viewport" content="width=device-width, initial-scale=1">
7 <link rel="icon" type="image/x-icon" href="favicon.ico">
8 <!-- CSS only -->
9 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-B0vHeVXK4R7YK1ZDnR9wvO0w0BvFBL6001tPr14tjfhskwup1fMph4vWor" crossorigin="anonymous">
10 <!-- JavaScript Bundle with Popper -->
11 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/js/bootstrap.bundle.min.js" integrity="sha384-p0n0373K10j520fA3065/S10k1dk1baU1F531v3A3YSDngEcC6M52n02" crossorigin="anonymous"></script>
12 </head>
13 <body>
14 <app-root></app-root>
15 </body>
16 </html>
```

- a.
5. En src/app/app.component.html dejar lo necesario
 - a. El {{ title }} toma el valor de la variable del app.component.ts mediante **interpolación de Strings**

```
FrontendAngular > src > app > app.component.html > @ router-outlet
1 <span>{{ title }} app is running!</span>
2 <router-outlet></router-outlet>
3
```

- b.
 - c. Router-outlet para mostrar los componentes hijos de un componente y para cuando redireccione a otras paginas el html de ese componente sea embebido dentro del app.component.html
6. En src/app/app.component.ts **programamos la funcionalidad del componente**
 - a. Para los otros componentes de acá se toma el nombre del selector para insertarlo en el app.component.html y mostrarlo en pantalla

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'FrontendAngular';
}
```

- b.
7. Definir las **redirecciones** en src/app/app.routing.module.ts, de modo que según la rota específico que componente me va a tener que abrir

```
TS app-routing.module.ts X
FrontendAngular > src > app > TS app-routing.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3
4 const routes: Routes = [];
5
6 @NgModule({
7   imports: [RouterModule.forRoot(routes)],
8   exports: [RouterModule]
9 })
10 export class AppRoutingModule { }
```

- a.

8. En src/app/app.module.ts **declaramos todos los servicios** que van a ser inyectados en nuestro código

```
FrontendAngular > src > app > TS app.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6
7 @NgModule({
8   declarations: [
9     AppComponent
10  ],
11   imports: [
12     BrowserModule,
13     AppRoutingModule
14  ],
15   providers: [],
16   bootstrap: [AppComponent]
17 })
18 export class AppModule { }
19
```

a.

9. **Crear nuevo Componente:**

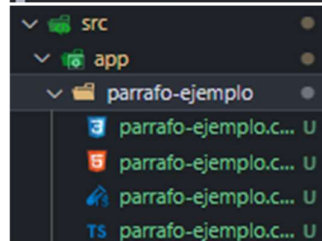
a. En terminal:

i. cd carpetaproyecto

ii. ng generate component nombreComponente

```
PS C:\Users\fabry\Desktop\Programacion III\ModeloAngular> cd .\FrontendAngular\
PS C:\Users\fabry\Desktop\Programacion III\ModeloAngular\FrontendAngular> ng generate component parrafoEjemplo
```

b.



c.

- d. **Automaticamente se importo en el app.module.ts**

```
FrontendAngular > src > app > TS app.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { ParrafoEjemploComponent } from './parrafo-ejemplo/parrafo-ejemplo.component';
7 import { ListasComponent } from './listas/listas.component';
8
9 @NgModule({
10   declarations: [
11     AppComponent,
12     ParrafoEjemploComponent,
13     ListasComponent
14   ],
15   imports: [
16     BrowserModule,
17     AppRoutingModule
18   ],
19   providers: [],
20   bootstrap: [AppComponent]
21 })
22 export class AppModule { }
```

i.

- e. Importar en `src/app/app.component.html` el selector del Componente de `parrafoEjemplo` para que aparezca visible en la pantalla

```
TS parrafo-ejemplo.component.ts U X
FrontendAngular > src > app > parrafo-ejemplo > TS parrafo-ejemplo.component.ts > ...
1 import { Component, OnInit } from '@angular/core';
2
3 @Component({
4   selector: 'app-parrafo-ejemplo',
5   templateUrl: './parrafo-ejemplo.component.html',
6   styleUrls: ['./parrafo-ejemplo.component.css']
7 })
8 export class ParrafoEjemploComponent implements OnInit {
9
10  constructor() {}
11
12  ngOnInit(): void {
13  }
14
15 }
16
```

i.

```
app.component.html M X
FrontendAngular > src > app > app.component.html > app-parrafo
1 <span>{{ title }} app is running!</span>
2 <app-parrafo-ejemplo></app-parrafo-ejemplo>
3 <router-outlet></router-outlet>
```

ii.

10. Creacion de un LOGIN

- a. Generar componente de Login

```
PS C:\Users\fabry\Desktop\Programacion III\ModeloAngular> cd .\FrontendAngular\
PS C:\Users\fabry\Desktop\Programacion III\ModeloAngular\FrontendAngular> ng generate component login
```

i.

```
src
├── app
│   ├── listas
│   └── login
│       ├── login.component... U
│       ├── login.component... U
│       ├── login.component... U
│       └── login.component.ts U
```

ii.

- b. HTML del Componente Login

```
login.component.html U X
FrontendAngular > src > app > login > login.component.html > div.container
1 <div class="container-fluid bg-primary py-4">
2   <div class="container text-white">
3     <h1 class="display-3">Bienvenidos</h1>
4     <p class="lead">Ingresa al sistema por favor</p>
5   </div>
6   <div class="container">
7     <div class="row">
8       <div class="col-6">
9         <form action="POST">
10           <div class="form-group">
11             <label for="nombreUsuario">Nombre de usuario:</label>
12             <input type="text" class="form-control" id="nombreUsuario" aria-describedby="UserHelp" placeholder="Enter Username">
13             <small id="UserHelp" class="form-text text-muted">We'll never share your credentials with anyone else.</small>
14           </div>
15           <div class="form-group">
16             <label for="password">Contraseña:</label>
17             <input type="password" class="form-control" id="password" placeholder="Password">
18           </div>
19           <div class="form-group">
20             <button type="submit" class="btn btn-primary">Ingresar</button>
21           </div>
22         </form>
23       </div>
24     </div>
25   </div>
26 </div>
```

i.

- c. Agregar el Componente al `app.component.html`

```
app.component.html M X
FrontendAngular > src > app > app.component.html > ...
1 <span>{{ title }} app is running!</span>
2 <app-parrafo-ejemplo></app-parrafo-ejemplo>
3 <app-listas></app-listas>
4 <app-login></app-login>
5 <router-outlet></router-outlet>
```

i.

- d. Agregar Rutas de Redireccion en app-routing.module.ts

```
TS app-routing.module.ts M X
FrontendAngular > src > app > TS app-routing.module.ts > routes > component
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ListasComponent } from '../listas/listas.component';
4 import { LoginComponent } from '../login/login.component';
5
6 const routes: Routes = [
7   {
8     path: 'login',
9     component: LoginComponent
10  },
11
12   {
13     path: 'listas',
14     component: ListasComponent
15  }
16 ];
17
18 @NgModule({
19   imports: [RouterModule.forRoot(routes)],
20   exports: [RouterModule]
21 })
22 export class AppRoutingModule { }
```

i.

- e. Crear navbar de redirecciones en app.component.html

```
TS app.component.html M X
FrontendAngular > src > app > TS app.component.html > ...
1 <!-- <span>{{ title }} app is running!</span>
2 <app-parrafo-ejemplo></app-parrafo-ejemplo>
3 <app-listas></app-listas>
4 <app-login></app-login> -->
5 <nav class="navbar navbar-expand-lg navbar-light bg-light">
6   <div class="container-fluid">
7     <a class="navbar-brand" href="#">Navbar</a>
8     <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavAltMarkup">
9       <span class="navbar-toggler-icon"></span>
10    </button>
11    <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
12      <div class="navbar-nav">
13        <a class="nav-link active" aria-current="page" routerLink="/login">Login</a>
14        <a class="nav-link" routerLink="/listas">Lista Personas</a>
15      </div>
16    </div>
17  </div>
18 </nav>
19 <router-outlet></router-outlet>
```

i.

- f. Crear modelos en carpeta Interfaces dentro de la carpeta app

- i. Crear Usuario.ts

- ii. Ng generate interface interfaces/Usuario

```
TS Usuario.ts U X
FrontendAngular > src > app > interfaces > TS U
1 export interface Usuario {
2   nombre: string;
3   password: string;
4 }
```

iii.

- g. En login.component.ts vamos a programar la lógica del login

```
TS login.component.ts U X
FrontendAngular > src > app > login > TS login.component.ts > ...
1 import { Component, OnInit } from '@angular/core';
2 import { Usuario } from '../interfaces/Usuario';
3
4 @Component({
5   selector: 'app-login',
6   templateUrl: './login.component.html',
7   styleUrls: ['./login.component.css']
8 })
9 export class LoginComponent implements OnInit {
10
11   usuario = {} as Usuario;
12   constructor() {
13
14   }
15
16   ngOnInit(): void {
17   }
18
19 }
```

- i.
 - ii. **Interpolacion:** es la forma de pasar información del Component.ts a la vista (unidireccional)
 - iii. **Binding con ngmodel:** para pasar información de la vista al Componente.ts y viceversa (bidireccional)
 - iv. **Data Propertie Binding:** para Imagenes
- h. **Enviar información de la Vista al Componente mediante ngModel**
- i. Agregar ngModel al app.module.ts

- 1. Importar el FormsModule

```
login.component.html U TS app.module.ts M X TS Usuario.ts U TS login.component.ts U
FrontendAngular > src > app > TS app.module.ts > ? AppModule
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { FormsModule } from '@angular/forms';
5 import { AppRoutingModule } from './app-routing.module';
6 import { AppComponent } from './app.component';
7 import { ParrafoEjemploComponent } from './parrafo-ejemplo/parrafo-ejemplo.component';
8 import { ListasComponent } from './listas/listas.component';
9 import { LoginComponent } from './login/login.component';
10
11 @NgModule({
12   declarations: [
13     AppComponent,
14     ParrafoEjemploComponent,
15     ListasComponent,
16     LoginComponent
17   ],
18   imports: [
19     BrowserModule,
20     AppRoutingModule,
21     FormsModule
22   ],
23   providers: [],
24   bootstrap: [AppComponent]
25 })
26 export class AppModule { }
```

- 2.

ii. Agregar funcionalidades en login.component.html

1. Agregar los name para que funcione

```
login.component.html X  TS Usuario.ts U  TS login.components.ts U
FrontendAngular > src > app > login > login.component.html > div.container > div.row > div.col-6 > img
1 <div class="container-fluid bg-primary py-4">
2   <div class="container text-white">
3     <h1 class="display-3">Bienvenidos</h1>
4     <p class="lead">Ingresa al sistema por favor</p>
5   </div>
6   <div class="container">
7     <div class="row">
8       <div class="col-6">
9         <p class="lead">Contador: {{contador}}</p>
10        <img [src]="urlImagen" alt="">
11        <form action="POST">
12          <div class="form-group">
13            <label for="nombreUsuario">Nombre de usuario:</label>
14            <input type="text" [(ngModel)]="usuario.nombre" name="nombre" class="form-control" id="nombreUsuario" placeholder="Enter Username">
15          </div>
16          <div class="form-group">
17            <label for="password">Contraseña:</label>
18            <input type="password" [(ngModel)]="usuario.password" name="password" class="form-control" id="password" placeholder="Password">
19          </div>
20          <div class="form-group">
21            <button type="submit" class="btn btn-primary" (click)="login()">Ingresar</button>
22          </div>
23        </form>
24      </div>
25    </div>
26  </div>
27</div>
```

2.

iii. Crear función de Login en el login.component.ts

```
login.component.html U  TS Usuario.ts U  TS login.components.ts X
FrontendAngular > src > app > login > TS login.components.ts > LoginComponent > urlImagen
1 import { Component, OnInit } from '@angular/core';
2 import { Usuario } from '../interfaces/Usuario';
3 import { Router } from '@angular/router';
4
5 @Component({
6   selector: 'app-login',
7   templateUrl: './login.component.html',
8   styleUrls: ['./login.component.css']
9 })
10 export class LoginComponent implements OnInit {
11
12   usuario = {} as Usuario;
13   contador: any;
14   urlImagen: string = '';
15   constructor(private router: Router) {
16     this.contador = 0;
17     this.urlImagen = 'https://cdn.cloudflare.com/steam/apps/221100/capsule_616x353.jpg?t=1643209285';
18   }
19
20   ngOnInit(): void {
21   }
22
23   login() {
24     this.contador++;
25     if(this.usuario.nombre == 'admin' && this.usuario.password == 'admin') {
26       alert('Login correcto');
27       this.router.navigateByUrl('/listas');
28     } else {
29       alert('Login incorrecto');
30     }
31   }
32 }
```

1.

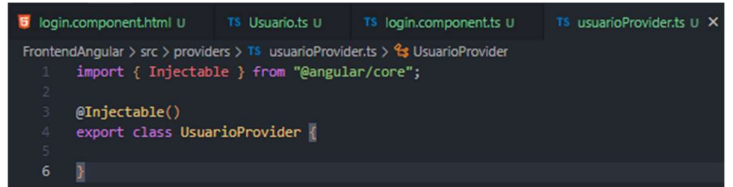
iv. Llamada a la API para el Login

1. Definir la URL de la API en environments/environments.ts

```
login.component.html U  TS Usuario.ts U  TS login.components.ts U  TS environments.ts X
FrontendAngular > src > environments > TS environments.ts > environment > urlAPI
1 // This file can be replaced during build by using the `fileReplacements` array.
2 // `ng build` replaces `environment.ts` with `environment.prod.ts`.
3 // The list of file replacements can be found in `angular.json`.
4
5 export const environment = {
6   production: false,
7   urlAPI: 'https://localhost:7132/'
8 };
a.
```


2. Crear providers para gestionar las acciones en las APIs

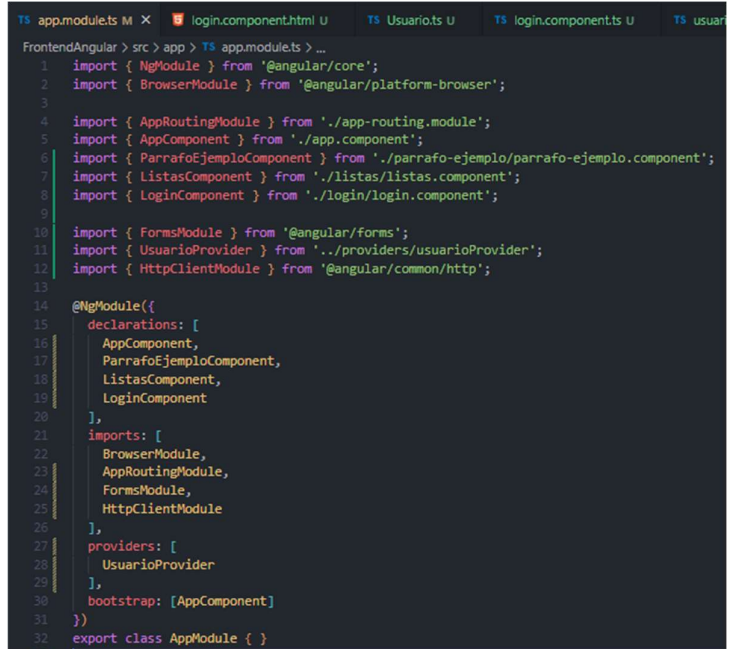
- Crear carpeta de providers en src
- Creamos usuarioProvider.ts



The screenshot shows a VS Code editor with the file explorer on the left displaying the project structure. The main editor shows the content of `usuarioProvider.ts` in the `providers` folder. The code defines an `UsuarioProvider` class decorated with `@Injectable()`.

```
FrontendAngular > src > providers > Ts usuarioProvider.ts > UsuarioProvider
1 import { Injectable } from "@angular/core";
2
3 @Injectable()
4 export class UsuarioProvider {
5
6 }
```

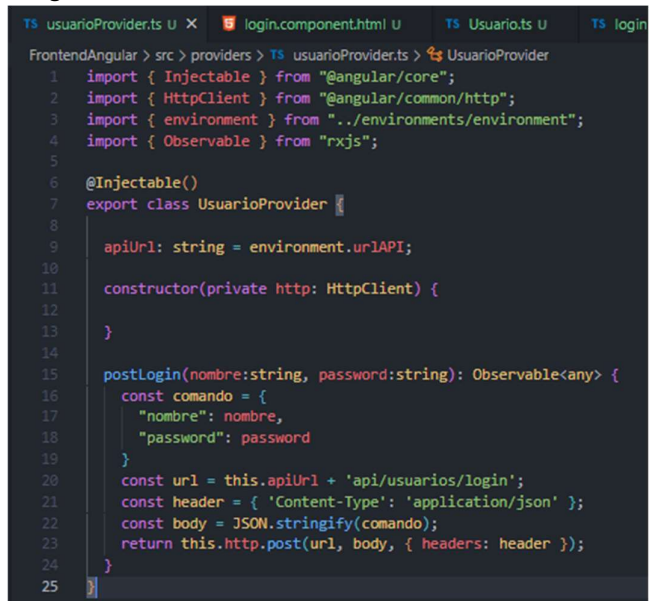
- Lo importamos el `UsuarioProvider` y el `Cliente HTTP` en el `app.module.ts`



The screenshot shows the `app.module.ts` file in the `src/app` directory. It imports various Angular modules and components, including `UsuarioProvider` and `HttpClientModule`. The `@NgModule` decorator is configured with declarations, imports, and providers.

```
FrontendAngular > src > app > Ts app.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { BrowserModule } from '@angular/platform-browser';
3
4 import { AppRoutingModule } from './app-routing.module';
5 import { AppComponent } from './app.component';
6 import { ParrafoEjemploComponent } from './parrafo-ejemplo/parrafo-ejemplo.component';
7 import { ListasComponent } from './listas/listas.component';
8 import { LoginComponent } from './login/login.component';
9
10 import { FormsModule } from '@angular/forms';
11 import { UsuarioProvider } from '../providers/usuarioProvider';
12 import { HttpClientModule } from '@angular/common/http';
13
14 @NgModule({
15   declarations: [
16     AppComponent,
17     ParrafoEjemploComponent,
18     ListasComponent,
19     LoginComponent
20   ],
21   imports: [
22     BrowserModule,
23     AppRoutingModule,
24     FormsModule,
25     HttpClientModule
26   ],
27   providers: [
28     UsuarioProvider
29   ],
30   bootstrap: [AppComponent]
31 })
32 export class AppModule { }
```

- Construir lógica del `UsuarioProvider`



The screenshot shows the implementation of the `UsuarioProvider` class. It imports `Injectable`, `HttpClient`, `environment`, and `Observable`. The class defines an `apiUrl` property, a constructor that takes an `HttpClient`, and a `postLogin` method that sends a POST request to the login endpoint.

```
FrontendAngular > src > providers > Ts usuarioProvider.ts > UsuarioProvider
1 import { Injectable } from "@angular/core";
2 import { HttpClient } from "@angular/common/http";
3 import { environment } from "../../environments/environment";
4 import { Observable } from "rxjs";
5
6 @Injectable()
7 export class UsuarioProvider {
8
9   apiUrl: string = environment.apiUrl;
10
11   constructor(private http: HttpClient) {
12
13   }
14
15   postLogin(nombre:string, password:string): Observable<any> {
16     const comando = {
17       "nombre": nombre,
18       "password": password
19     };
20     const url = this.apiUrl + 'api/usuarios/login';
21     const header = { 'Content-Type': 'application/json' };
22     const body = JSON.stringify(comando);
23     return this.http.post(url, body, { headers: header });
24   }
25 }
```

- e. Dentro del login.component.ts hacer uso del Provider para consumir la API

```
login.component.ts x usuarioProviders.ts login.component.html u Usuarios.ts
FrontendAngular > src > app > login > TS login.component.ts > LoginComponent > login > subscribe() callback
9 styleUrls: ['./login.component.css']
10 })
11 export class LoginComponent implements OnInit {
12
13     usuario = {} as Usuario;
14     contador:any;
15     urlImagen:string = '';
16
17     constructor(private router: Router, private usuarioProvider: UsuarioProvider) {
18         this.contador = 0;
19         this.urlImagen = 'https://cdn.cloudflare.com/steam/apps/221100/capsule_616x353.jpg?t=1643209285';
20     }
21
22     ngOnInit(): void {
23     }
24
25     login() {
26         // this.contador++;
27         // if(this.usuario.nombre == 'admin' && this.usuario.password == 'admin') {
28         //     alert('Login correcto');
29         //     this.router.navigateByUrl('/listas');
30         // } else {
31         //     alert('Login incorrecto');
32         // }
33         this.usuarioProvider.postLogin(this.usuario.nombre, this.usuario.password).subscribe(
34             (data) => {
35                 if(data.ok) {
36                     this.router.navigateByUrl('/listas');
37                 } else {
38                     alert('Login incorrecto');
39                 }
40             }
41         );
42     }
43 }
```

i.

11. Llenado de Lista por medio del GET

- a. Crear el método Get en el Provider

```
usuarioProviders.ts x listas.component.html u Usuarios.ts
FrontendAngular > src > providers > TS usuarioProviders.ts > UsuarioProvider > getTodos
1 import { Injectable } from '@angular/core';
2 import { HttpClient } from '@angular/common/http';
3 import { environment } from '../environments/environment';
4 import { Observable } from 'rxjs';
5
6 @Injectable()
7 export class UsuarioProvider {
8
9     apiUrl: string = environment.apiUrl;
10
11     constructor(private http: HttpClient) {
12     }
13
14     postLogin(nombre:string, password:string): Observable<any> {
15         const comando = {
16             'nombre': nombre,
17             'password': password
18         };
19         const url = this.apiUrl + 'api/usuarios/login';
20         const header = { 'Content-Type': 'application/json' };
21         const body = JSON.stringify(comando);
22         return this.http.post(url, body, { headers: header });
23     }
24
25     getTodos(): Observable<any> {
26         const url = this.apiUrl + 'api/usuarios';
27         const header = { 'Content-Type': 'application/json' };
28         return this.http.get(url, { headers: header });
29     }
30 }
31 }
```

i.

- b. Importar Provider y Router para redireccionar al Editar en listas.component.ts

```
listas.component.ts x usuarioProviders.ts listas.component.html u
FrontendAngular > src > app > listas > TS listas.component.ts > ListasComponent
1 import { Component, OnInit } from '@angular/core';
2 import { UsuarioProvider } from '../providers/usuarioProvider';
3 import { Router } from '@angular/router';
4
5 @Component({
6     selector: 'app-listas',
7     templateUrl: './listas.component.html',
8     styleUrls: ['./listas.component.css']
9 })
10 export class ListasComponent implements OnInit {
11
12     constructor() { }
13
14     ngOnInit(): void {
15     }
16
17 }
```

i.

c. Inyectar Router y Provider en el Constructor

```
TS listas.component.ts U • TS usuarioProvider.ts U listas.component.html U TS Usuario.ts U
FrontendAngular > src > app > listas > TS listas.component.ts > ListasComponent > obtenerUsuarios
1 import { Component, OnInit } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { UsuarioProvider } from '../../providers/usuarioProvider';
4
5 @Component({
6   selector: 'app-listas',
7   templateUrl: './listas.component.html',
8   styleUrls: ['./listas.component.css']
9 })
10 export class ListasComponent implements OnInit {
11
12   constructor(private router: Router, private usuarioProvider: UsuarioProvider) {
13
14   }
15
16   ngOnInit(): void {
17   }
18
19   async obtenerUsuarios() {
20
21   }
22 }
```

i.

d. Crear función para cargar Lista con los datos consumidos de la API (Verificar que el Provider se encuentre en el app.modules.ts)

```
TS listas.component.ts U X TS usuarioProvider.ts U listas.component.html U TS Usuario.ts U
FrontendAngular > src > app > listas > TS listas.component.ts > ListasComponent > obtenerUsuarios > subscribe()
1 import { Component, OnInit } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { UsuarioProvider } from '../../providers/usuarioProvider';
4
5 @Component({
6   selector: 'app-listas',
7   templateUrl: './listas.component.html',
8   styleUrls: ['./listas.component.css']
9 })
10 export class ListasComponent implements OnInit {
11
12   listaUsuarios:any = [];
13   constructor(private router: Router, private usuarioProvider: UsuarioProvider) {
14
15   }
16
17   ngOnInit(): void {
18     this.obtenerUsuarios();
19   }
20
21   async obtenerUsuarios() {
22     this.listaUsuarios = await this.usuarioProvider.getUsuarios().subscribe((data) => {
23       if(data.ok) {
24         this.listaUsuarios = data.listaUsuarios;
25       } else {
26         alert(data.error);
27       }
28     })
29   }
30 }
```

i.

- e. Creamos y Mostramos la Tabla desde listas.component.html mediante *ngFor con interpolación con los datos recibidos de la API

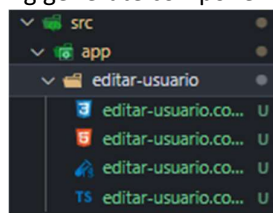
```
listas.component.html U X TS listas.component.ts U TS usuarioProvider.ts U TS Usuario.ts U
FrontendAngular > src > app > listas > listas.component.html > div.container > table.table.table-dark.table-striped > tbody
1 <div class="container">
2   <table class="table table-dark table-striped">
3     <thead>
4       <tr>
5         <th scope="col">Nombre</th>
6         <th scope="col">Apellido</th>
7         <th scope="col">Rol</th>
8         <th scope="col">Acciones</th>
9       </tr>
10    </thead>
11    <tbody>
12      <tr *ngFor="let user of listaUsuarios">
13        <td>{{user.nombre}}</td>
14        <td>{{user.apellido}}</td>
15        <td>{{user.rol}}</td>
16        <td>
17          <button class="btn btn-primary" (click)="editUser(user.id)">Editar</button>
18          <button class="btn btn-danger" (click)="eliminarUsuario(user.id)">Eliminar</button>
19        </td>
20      </tr>
21    </tbody>
22  </table>
23 </div>
```

- i.
- f. Creamos funciones para editar y eliminar usuario según el id enviado desde la table llamando a las APIs correspondientes

- i. Creamos función de editar, primero nos redirecciona a una nueva ruta enviando por url el Id del usuario a modificar dentro de listas.component.ts

```
TS listas.component.ts U X TS editar-usuario.component.ts U TS lista
FrontendAngular > src > app > listas > TS listas.component.ts > ListasComp
30
31   async editUser(userId: string) {
32     this.router.navigateByUrl('/editar/' + userId);
33   }
34
35   async eliminarUsuario(userId: any) {
36     console.log(userId);
37   }
38 }
```

- 1.
- ii. Generamos un nuevo componente para la edición
1. ng generate component editarUsuario



- 2.

iii. Agregamos la ruta y su componente en app-routing.module.ts

```
FrontendAngular > src > app > TS app-routing.module.ts > ...
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { EditarUsuarioComponent } from '../editar-usuario/editar-usuario.component';
4 import { ListasComponent } from '../listas/listas.component';
5 import { LoginComponent } from '../login/login.component';
6
7 const routes: Routes = [
8   {
9     path: 'login',
10    component: LoginComponent
11  },
12  {
13    path: 'listas',
14    component: ListasComponent
15  },
16  {
17    path: 'editar/:id',
18    component: EditarUsuarioComponent
19  }
20 ];
21
22 @NgModule({
23   imports: [RouterModule.forRoot(routes)],
24   exports: [RouterModule]
25 })
26 export class AppRoutingModule { }
```

1.

iv. Construcción del Componente de editar

1. Función para consultar API y obtener datos del usuario por ID

```
FrontendAngular > src > app > editar-usuario > TS editar-usuario.component.ts > EditarUsuarioComponent > constructo
1 import { Component, OnInit } from '@angular/core';
2 import { UsuarioProvider } from '../../providers/usuarioProvider';
3 import { ActivatedRoute } from '@angular/router';
4
5 @Component({
6   selector: 'app-editar-usuario',
7   templateUrl: '../editar-usuario.component.html',
8   styleUrls: ['../editar-usuario.component.css']
9 })
10 export class EditarUsuarioComponent implements OnInit {
11
12   idPersona:string = "";
13   constructor(private usuarioProvider: UsuarioProvider, private route: ActivatedRoute) {
14     this.idPersona = this.route.snapshot.params['id'];
15     alert(this.idPersona);
16   }
17
18   ngOnInit(): void {
19   }
20
21 }
22 }
```

2.