

## Informe #1

**Integrantes:** Carlos Quinaluisa, Gibrán Guzmán, Darwin Mosquera, Paola Franco, Mateo Beltrán

### Conexión de laravel(api) con angular(front-end)

#### Introducción

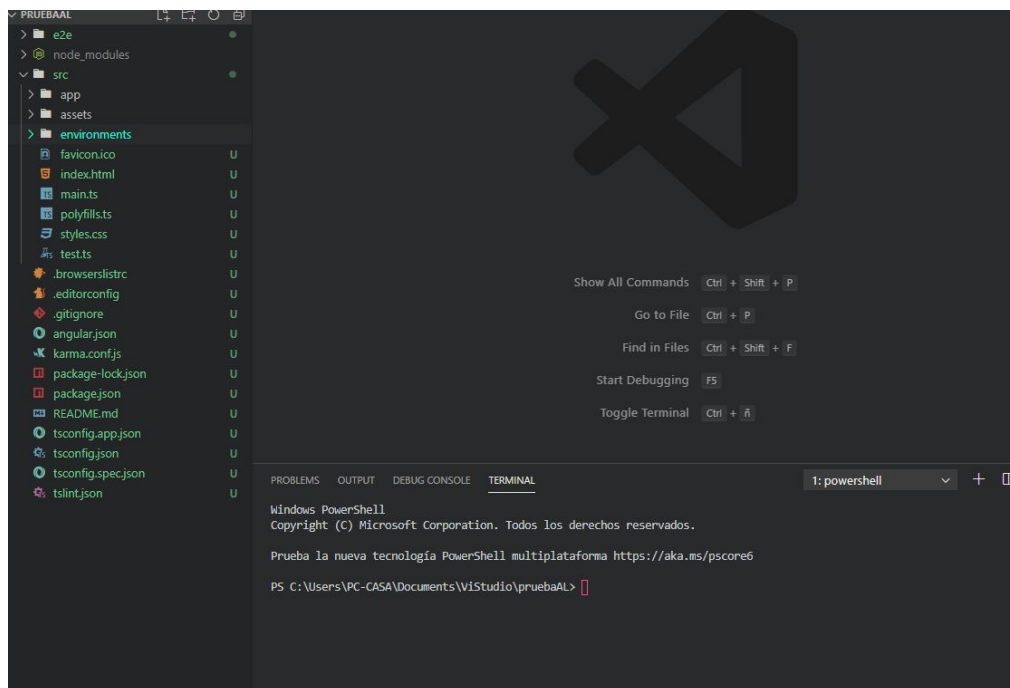
En este informe vamos a detallar como es la conexión del back-end hecho en laravel conectarla al front-end con Angular.

Recordar que primeramente ya debemos tener nuestro back-end ya realizado en laravel

Pasos:

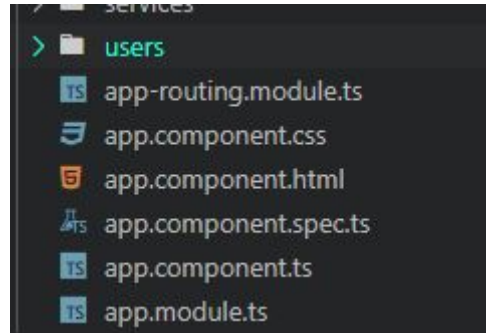
1. Empezar un nuevo proyecto en laravel con el siguiente comando ***ng new my-project-angular***.

Una vez ya terminado de crear el proyecto comienza la conexión.



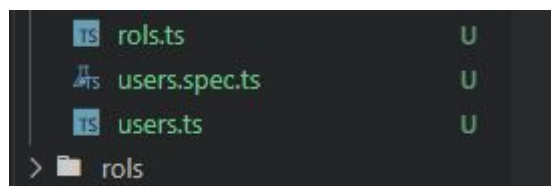
2. Comenzamos a generar los componentes de nuestro proyecto con el siguiente comando(recordando cuales son los modelos, migraciones de nuestro back-end en laravel) para esto se tomará de ejemplo el de user) ***ng g c nombrecomponente***

***g: generate , c: component***

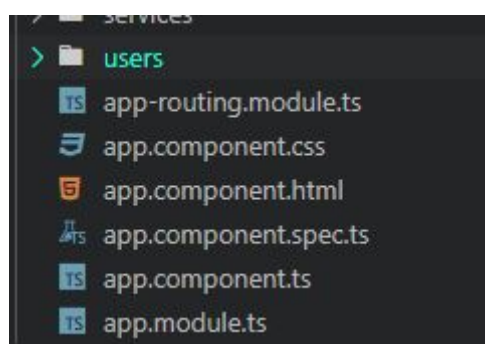


3. Ahora vamos a crear los models con el siguiente comando que está a continuación; para no tener problemas de unión usar el mismo nombre que el componente ***ng g class models/nombre del modelo***

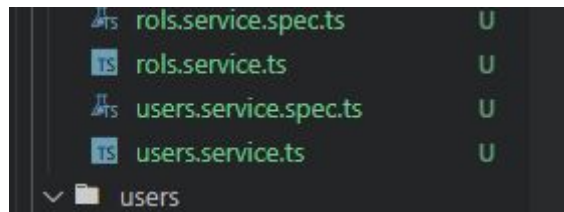
***class: una clase para realizar los modelos en angular***



4. Vamos a ejecutar el comando de route ***ng g module/nombre del archivo --routing***  
***--routing: enrutamiento***



5. Para finalizar con los componentes vamos a usar el comando de service **ng g s services/nombre del archivo**  
**s: services**



Haremos unas configuraciones para no tener inconvenientes con la conexión

6. Nos dirigimos al archivo que viene por defecto el cual es **app.module.ts**, este archivo ya viene creado al momento de hacer el proyecto. Lo que vamos a importar es dos elementos:
- **HttpClient Module:** La mayoría de las aplicaciones front-end necesitan comunicarse con un servidor a través del protocolo HTTP para descargar o cargar datos y acceder a otros servicios back-end.
  - **Forms Module:** Esta importación es para poder tener control sobre el formulario ya que vamos a usar bootstrap para el diseño.

```
src > app > app.module.ts > ...
7  import { UsersComponent } from '../users/
   users.component';
8  import { HttpClientModule } from '@angular/
   common/http';
9  import { FormsModule } from '@angular/
   forms';
10 import { RolsComponent } from '../rols/rols.
   component';
11
12
13 @NgModule({
14   declarations: [
15     AppComponent,
16     UsersComponent,
17     RolsComponent
18   ],
19   imports: [
20     BrowserModule,
21     AppRoutingModule,
22     HttpClientModule,
23     FormsModule
24   ],
25   providers: [],
26   bootstrap: [AppComponent]
27 })
28 export class AppModule { }
```

7. Una vez ya echo esta configuración nos dirigimos al elemento que creamos con el código de services el cual es **user.services.ts**

- En este archivo importamos el Http Client y también importamos el modelo de users.
- Para la conexión con el api en laravel llamamos a la ruta que tenemos en laravel y le ponemos en una variable privada: `private url = 'http://localhost:8000/api/'`
- También creamos el método el cual vamos a usar, esta vez usaremos el método post para enviar datos
- Verificar cual es la ruta que tenemos en laravel

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
import { Users } from '../models/users';
@Injectable({
  providedIn: 'root'
})
export class UsersService {
  private url = 'http://localhost:8000/api/'

  constructor( private http: HttpClient ) {
  }

  crearUser(data: any){
    return this.http.post(this.url +
      'user2', data);
  }

  getUsers(): Observable<Object>{
    return this.http.get<Users[]>(`${this.
      url}/user2`);
  }
}
```

8. Ahora nos dirigimos al **users.component.ts** en este le vamos a dar la lógica para saber que campos vamos a llenar para mandar a la base; también debemos importar el **User Services** el cual está hecho el anteriormente.

```
import { Component, OnInit } from '@angular/core';
import { UsersService } from '../services/users.service';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { error } from '@angular/compiler/src/util';

@Component({
  selector: 'app-users',
  templateUrl: './users.component.html',
  styleUrls: ['./users.component.css']
})
export class UsersComponent implements OnInit {

  constructor(
    private user : UsersService
  ) { }

  ngOnInit(): void {
  }

}
```

```
private user : UsersService

) { }

nombreUsuario: any;
nombreCompleto: any;
email: any;
telefono: any;

ngOnInit() {
}

crear() {
  let data = {
    "nombreUsuario": this.nombreUsuario,
    "nombreCompleto": this.nombreCompleto,
    "email": this.email,
    "telefono": this.telefono
  }
  this.user.createUser(data).subscribe(
    response => {
      console.log('Creado')
      alert("Usuario Creado")
    }
  )
}
```

```

crear(){
  let data = {
    "nombreUsuario": this.nombreUsuario,
    "nombreCompleto": this.nombreCompleto,
    "email": this.email,
    "telefono": this.telefono
  }
  this.user.createUser(data).subscribe(
    response => {
      console.log('Creado')
      alert("Usuario Creado")
    }, error => console.log(error)
  )
}
}

```

En este paso declaramos variables con any ya que aun no saben qué es lo que van a mandar, el data es un dato el cual nos sirve para que envíe al services.

9. Ahora vamos a crear la vista en el componente users.components.html

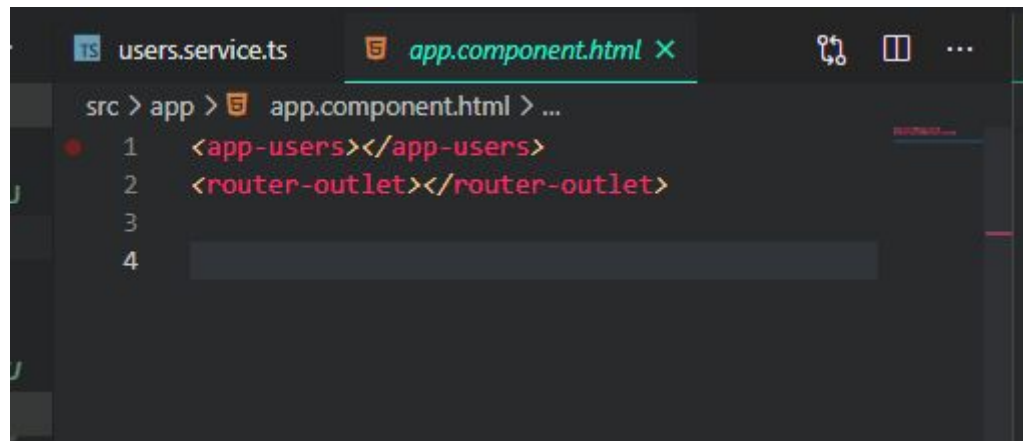
En este vamos a usar un ngModel: Crea una FormControl instancia a partir de un modelo de dominio y la vincula a un elemento de control de formulario.



rc > app > users > users.component.html > div.card

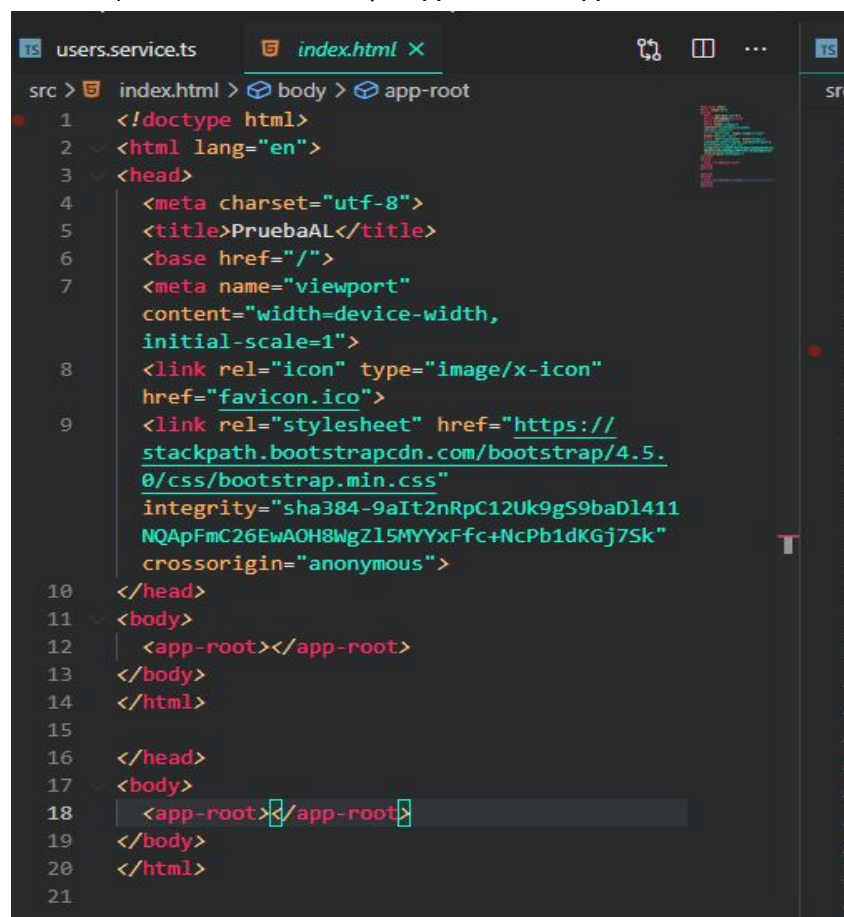
```
1 <h1>Crear Usuario</h1>
2
3
4
5 <div class="card">
6   <div class="card-body">
7     <label for=""> Su Usuario </label>
8     <input type="text"
9       [(ngModel)]="nombreUsuario"
10      placeholder="Ingrese su Usuario"
11      class="form-control"
12    >
13     <label for=""> Su nombre </label>
14     <input type="text"
15       [(ngModel)]="nombreCompleto"
16       placeholder="Ingrese su Nombre"
17       class="form-control"
18     >
19     <label for=""> Su Email </label>
20     <input type="text"
21       [(ngModel)]="email"
22       placeholder="Ingrese su Email"
23       class="form-control"
24     >
25     <label for=""> Su telefono </label>
26     <input type="text"
27       [(ngModel)]="telefono"
28       placeholder="Ingrese su telefono"
29       class="form-control"
30     >
31     <br>
32     <button type="button" class="btn
33       btn-primary btn-sm btn-block "
34       (click)="crear()" > Crear </button>
35   </div>
36 </div>
37
```

10. Una vez ya creado el formulario ahora nos vamos al elemento al `app.component.html` y ahora dentro del ello llamamos a la vista con esta etiqueta `<app-users></app-users>`



```
src > app > app.component.html > ...
1 <app-users></app-users>
2 <router-outlet></router-outlet>
3
4
```

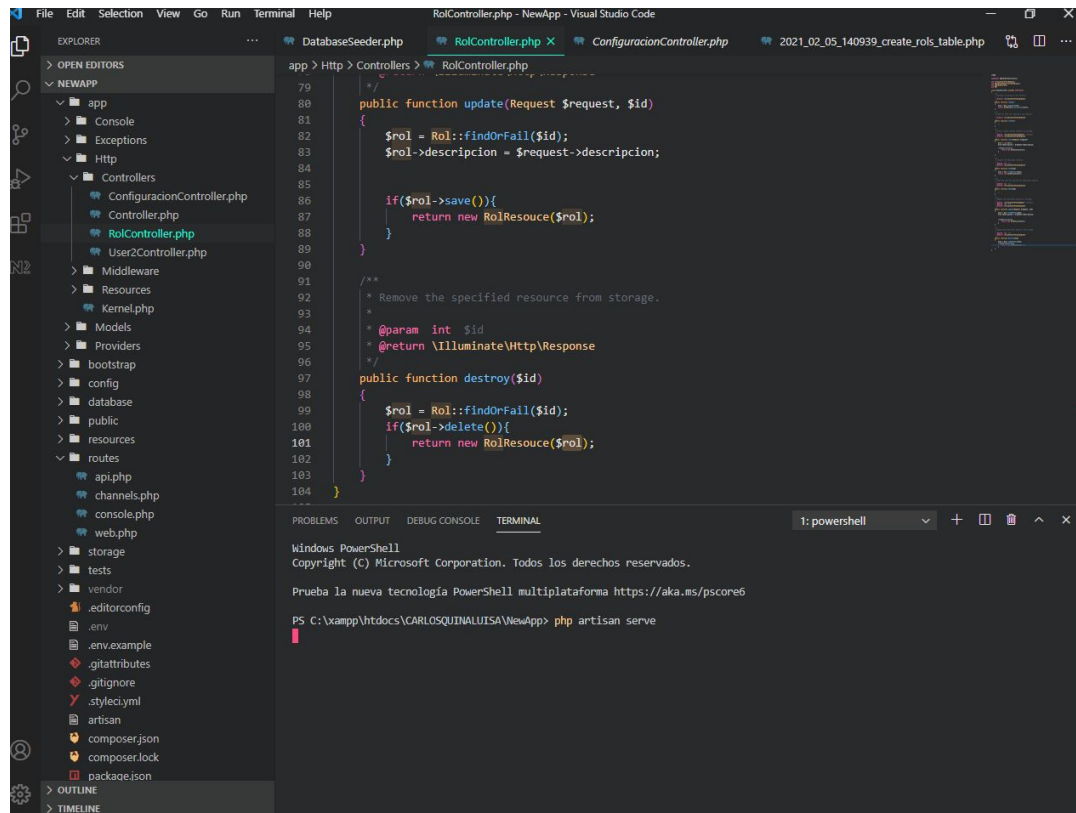
11. Ahora nos vamos al index que ya está creado por defecto y para ya poder visualizar llamamos con la etiqueta dentro del body `<app-root></app-root>`



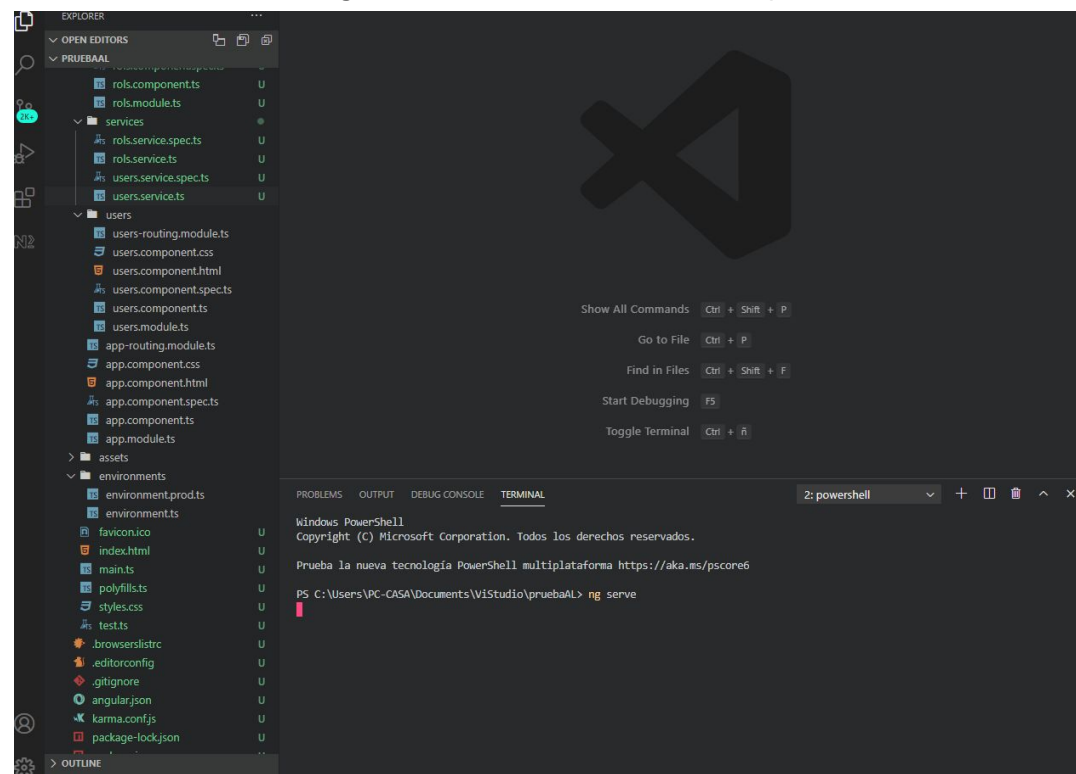
```
src > index.html > body > app-root
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>PruebaAL</title>
6   <base href="/">
7   <meta name="viewport"
8     content="width=device-width,
9     initial-scale=1">
10  <link rel="icon" type="image/x-icon"
11    href="favicon.ico">
12  <link rel="stylesheet" href="https://
13    stackpath.bootstrapcdn.com/bootstrap/4.5.
14    0/css/bootstrap.min.css"
15    integrity="sha384-9aIt2nRpC12Uk9gS9baD1411
16    NQApFmC26EwAOH8WgZl5MYxxFc+NcPb1dKGj7Sk"
17    crossorigin="anonymous">
18 </head>
19 <body>
20   <app-root></app-root>
21 </body>
22 </html>
```



12. Ahora para hacer pruebas prendemos el servidor en laravel con el comando **php artisan serve**



13. Prendemos el servidor en angular (front-end) con el comando **ng serve**



## Crear

localhost:4200 dice  
Usuario Creado

Aceptar

Su Usuario

Maritzalng

Su nombre

Maritza Tituaña

Su Email

marititu@hotmail.com

Su telefono

03446465465

Crear

- > Event Triggers
- > Extensions
- > Foreign Data Wrappers
- > Languages
- > Schemas (1)
  - > public
    - > Collations
    - > Domains
    - > FTS Configurations
    - > FTS Dictionaries
    - > FTS Parsers
    - > FTS Templates
    - > Foreign Tables
    - > Functions
    - > Materialized Views
    - > Sequences
    - > Tables (8)
      - > configuracions
      - > failed\_jobs
      - > migrations
      - > password\_resets
      - > rols
      - > user2\_rols
      - > user2s
      - > users
    - > Trigger Functions
    - > Types
    - > Views
- > postgres
- > prueba

public.user2s/newapp/postgres@PostgreSQL 10

Query Editor Query History


```
1 SELECT * FROM public.user2s
2 ORDER BY id ASC
```


Data Output Explain Messages Notifications

	id [PK] bigint	nombreUsuario character varying (255)	nombreCompleto character varying (255)	email character varying (255)	telefono character varying (255)
1	28	carlos_mijin	Carlos Quinaluisa	carlostedu@hotmail.com	09123655454
2	29	prueba	prueba2	pruba@hotmail.com	21321313
3	30	Maritzalng	Maritza Tituaña	marititu@hotmail.com	03446465465

Query Params

KEY	VALUE	DESCRIP
Key	Value	Descripti

Body Cookies Headers (10) Test Results  Status: 200

Pretty Raw Preview Visualize JSON 

```

23 ..... "configuracions_id": null,
24 ..... "created_at": "2021-02-18T15:04:15.000000Z",
25 ..... "updated_at": "2021-02-18T15:04:15.000000Z"
26 ..... },
27 ..... {
28 .....   "id": 30,
29 .....   "nombreUsuario": "MaritzaIng",
30 .....   "nombreCompleto": "Maritza Tituaña",
31 .....   "email": "marititu@hotmail.com",
32 .....   "telefono": "03446465465",
33 .....   "edad": null,
34 .....   "FechaNaci": null,
35 .....   "configuracions_id": null,
36 .....   "created_at": "2021-02-20T02:37:44.000000Z",
37 .....   "updated_at": "2021-02-20T02:37:44.000000Z"
38 ..... }
39 ..... ],
40 ..... "tituaña": f
  
```

Link de GitHub :: <https://github.com/KHORE/ConexionAPiAngu>