

Taller FrontEnd

PRESENTADO POR:

Jorge Luis Rojas Muñoz

PRESENTADO A:

Vicente Aux Revelo

UNIVERSIDAD DE NARIÑO

INGENIERIA DE SISTEMAS

DIPLOMADO DE ACTUALIZACIÓN EN NUEVAS TECNOLOGÍAS PARA EL
DESARROLLO DE SOFTWARE

2024

1. Creación de componentes, uso de ngModel, RouterLink, Servicios

Primero nos ubicamos en la carpeta Diplomado2024B luego en la terminal ejecutamos el comando `ng new TallerfrontEndPawpal --no-standalone`, este comando crea una nueva aplicación Angular llamada "**TallerfrontEndPawpal**". La opción `--no-standalone` indica que no se generará un módulo independiente, por lo que se utilizará el enfoque clásico de Angular con módulos y componentes.

```
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B> ng new TallerfrontEndPawpal --no-standalone
? Which stylesheet format would you like to use? CSS [ https://developer.mozilla.org/docs/Web/CSS ]
]
? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? yes
CREATE TallerfrontEndPawpal/angular.json (3158 bytes)
CREATE TallerfrontEndPawpal/package.json (1310 bytes)
CREATE TallerfrontEndPawpal/README.md (1108 bytes)
CREATE TallerfrontEndPawpal/tsconfig.json (1045 bytes)
CREATE TallerfrontEndPawpal/.editorconfig (290 bytes)
CREATE TallerfrontEndPawpal/.gitignore (629 bytes)
CREATE TallerfrontEndPawpal/tsconfig.app.json (504 bytes)
CREATE TallerfrontEndPawpal/tsconfig.spec.json (449 bytes)
CREATE TallerfrontEndPawpal/server.ts (1800 bytes)
CREATE TallerfrontEndPawpal/.vscode/extensions.json (134 bytes)
CREATE TallerfrontEndPawpal/.vscode/launch.json (400 bytes)
CREATE TallerfrontEndPawpal/.vscode/tasks.json (980 bytes)
CREATE TallerfrontEndPawpal/src/main.ts (256 bytes)
CREATE TallerfrontEndPawpal/src/index.html (319 bytes)
CREATE TallerfrontEndPawpal/src/styles.css (81 bytes)
CREATE TallerfrontEndPawpal/src/main.server.ts (71 bytes)
CREATE TallerfrontEndPawpal/src/app/app-routing.module.ts (255 bytes)
CREATE TallerfrontEndPawpal/src/app/app.module.ts (467 bytes)
CREATE TallerfrontEndPawpal/src/app/app.component.html (20239 bytes)
CREATE TallerfrontEndPawpal/src/app/app.component.spec.ts (1120 bytes)
CREATE TallerfrontEndPawpal/src/app/app.component.ts (231 bytes)
CREATE TallerfrontEndPawpal/src/app/app.component.css (0 bytes)
CREATE TallerfrontEndPawpal/src/app/app.module.server.ts (332 bytes)
CREATE TallerfrontEndPawpal/public/favicon.ico (15086 bytes)
✓ Packages installed successfully.
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/app/app.module.ts', LF will be replaced by CRLF the next time Git touches it
  Successfully initialized git.
○ PS C:\Users\7Jrmo7\Documents\Diplomado2024B>
```

También se ejecutó el comando `ng add @ng-bootstrap/ng-bootstrap`, que agrega el paquete de `ng-bootstrap` al proyecto Angular.

```
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng add @ng-bootstrap/ng-bootstrap
Would you like to share pseudonymous usage data about this project with the Angular Team
at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more
details and how to change this setting, see https://angular.dev/cli/analytics.
yes
Thank you for sharing pseudonymous usage data. Should you change your mind, the following
command will disable this feature entirely:
  ng analytics disable
Global setting: enabled
Local setting: enabled
Effective status: enabled
✓ Determining Package Manager
  > Using package manager: npm
✓ Searching for compatible package version
  > Found compatible package version: @ng-bootstrap/ng-bootstrap@17.0.1.
✓ Loading package information from registry
✓ Confirming installation
✓ Installing package
UPDATE package.json (1451 bytes)
✓ Packages installed successfully.
UPDATE src/app/app.module.ts (540 bytes)
UPDATE angular.json (3383 bytes)
UPDATE src/main.ts (301 bytes)
UPDATE tsconfig.app.json (532 bytes)
UPDATE tsconfig.spec.json (477 bytes)
○ PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal>
```

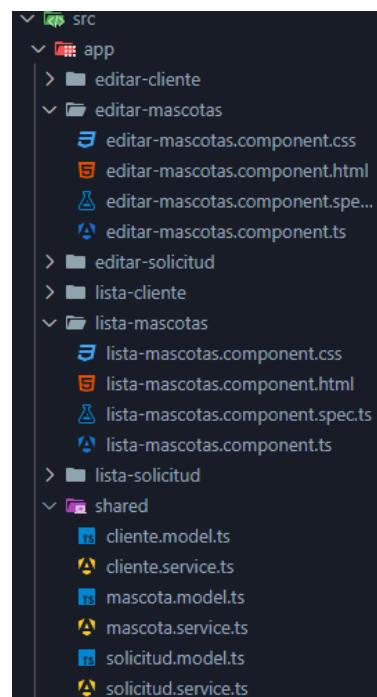
Luego también en la carpeta **TallerfrontEndPawpal**, se generaron un servicio y dos componentes para manejar la lógica y las interfaces relacionadas con las mascotas, el servicio **shared/mascota** se creó para encapsular la lógica de negocio compartida entre diferentes componentes, adicionalmente, se generaron los componentes **listaMascotas** y **editarMascotas**, que permiten visualizar una lista de mascotas, crear, editar y eliminar la información de cada una, respectivamente, estos componentes incluyen sus propios archivos HTML, CSS y TypeScript para manejar tanto la estructura visual como la lógica de cada vista, por último, el archivo **app.module.ts** fue actualizado automáticamente para declarar estos nuevos componentes en el módulo raíz de la aplicación.

```
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng generate service shared/mascota
CREATE src/app/shared/mascota.service.spec.ts (378 bytes)
CREATE src/app/shared/mascota.service.ts (145 bytes)
○ PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> █
```

```
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng generate component listaMascotas
CREATE src/app/lista-mascotas/lista-mascotas.component.html (30 bytes)
CREATE src/app/lista-mascotas/lista-mascotas.component.spec.ts (670 bytes)
CREATE src/app/lista-mascotas/lista-mascotas.component.ts (240 bytes)
CREATE src/app/lista-mascotas/lista-mascotas.component.css (0 bytes)
○ PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> █
```

```
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng generate component editarMascotas
CREATE src/app/editar-mascotas/editar-mascotas.component.html (31 bytes)
CREATE src/app/editar-mascotas/editar-mascotas.component.spec.ts (677 bytes)
CREATE src/app/editar-mascotas/editar-mascotas.component.ts (244 bytes)
CREATE src/app/editar-mascotas/editar-mascotas.component.css (0 bytes)
UPDATE src/app/app.module.ts (898 bytes)
```

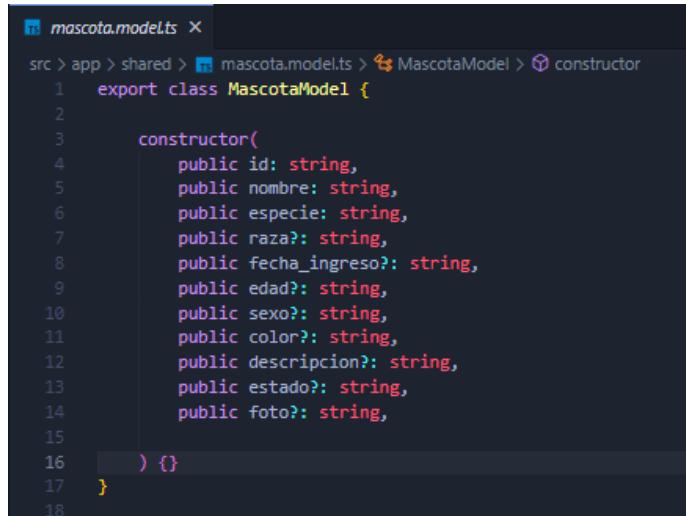
Podemos ver que si se agregaron en la siguiente imagen



El archivo **mascota.service** quedo así:

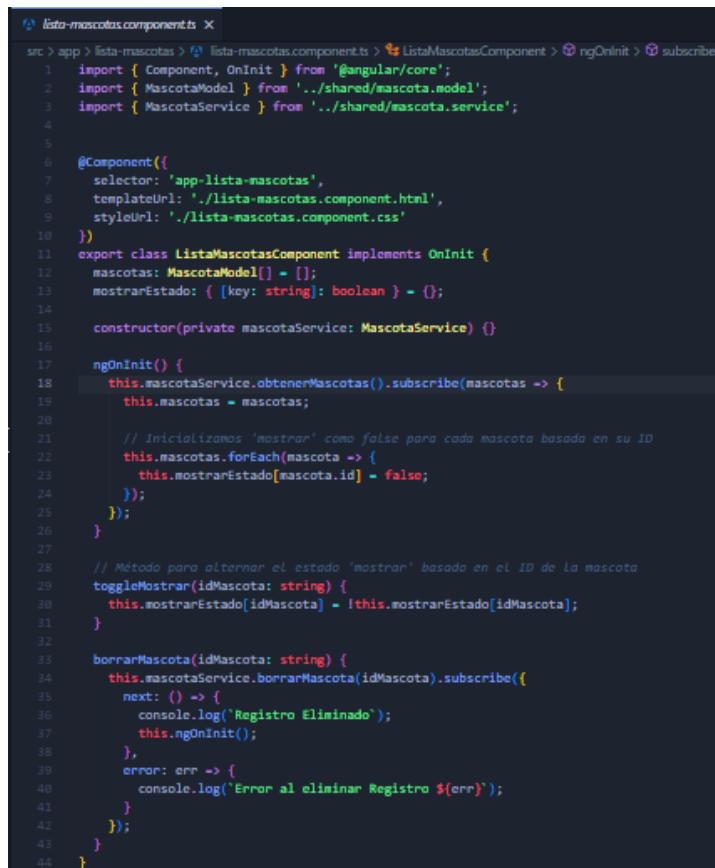
```
mascota.service.ts
src > app > shared > □ mascota.service.ts > MascotaService
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { MascotaModel } from './mascota.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class MascotaService {
9
10   BASE_URL='http://localhost:4000';
11
12   constructor(private http: HttpClient) {
13
14     //Lista completa mascotas
15   }
16   obtenerMascotas(){
17     return this.http.get<MascotaModel[]>(`${this.BASE_URL}/mascotas/buscarM`);
18   }
19
20   //buscar mascota por ID
21   obtenerMascota(idMascota:string){
22     return this.http.get<MascotaModel>(`${this.BASE_URL}/mascotas/buscarIdM/${idMascota}`);
23   }
24
25   agregarMascota(mascota:MascotaModel){
26     return this.http.post<string>(`${this.BASE_URL}/mascotas/crearM`,mascota);
27   }
28   actualizarMascota(mascota:MascotaModel){
29     return this.http.put<string>(`${this.BASE_URL}/mascotas/actualizarM/${mascota.id}`,mascota);
30   }
31
32   borrarMascota(idMascota:string){
33     return this.http.delete<string>(`${this.BASE_URL}/mascotas/eliminarM/${idMascota}`);
34   }
35
36
37 }
```

y también se le creo un modelo el cual se llama **mascota.model.ts** el cual quedo así para su funcionamiento:



```
src > app > shared > mascota.model.ts > MascotaModel > constructor
1  export class MascotaModel {
2
3      constructor(
4          public id: string,
5          public nombre: string,
6          public especie: string,
7          public raza?: string,
8          public fecha_ingreso?: string,
9          public edad?: string,
10         public sexo?: string,
11         public color?: string,
12         public descripcion?: string,
13         public estado?: string,
14         public foto?: string,
15     ) {}
16 }
17
18
```

En el caso de **lista-mascotas.component.ts** su código respectivo es el siguiente:



```
src > app > lista-mascotas > lista-mascotas.component.ts > ListaMascotasComponent > ngOnInit > subscribe()
1  import { Component, OnInit } from '@angular/core';
2  import { MascotaModel } from '../shared/mascota.model';
3  import { MascotaService } from '../shared/mascota.service';
4
5
6  @Component({
7      selector: 'app-lista-mascotas',
8      templateUrl: './lista-mascotas.component.html',
9      styleUrls: ['./lista-mascotas.component.css'
10 })
11 export class ListaMascotasComponent implements OnInit {
12     mascotas: MascotaModel[] = [];
13     mostrarEstado: { [key: string]: boolean } = {};
14
15     constructor(private mascotaService: MascotaService) {}
16
17     ngOnInit() {
18         this.mascotaService.obtenerMascotas().subscribe(mascotas => {
19             this.mascotas = mascotas;
20
21             // Inicializamos 'mostrar' como false para cada mascota basada en su ID
22             this.mascotas.forEach(mascota => {
23                 this.mostrarEstado[mascota.id] = false;
24             });
25         });
26     }
27
28     // Método para alternar el estado 'mostrar' basado en el ID de la mascota
29     toggleMostrar(idMascota: string) {
30         this.mostrarEstado[idMascota] = !this.mostrarEstado[idMascota];
31     }
32
33     borrarMascota(idMascota: string) {
34         this.mascotaService.borrarMascota(idMascota).subscribe({
35             next: () => {
36                 console.log('Registro Eliminado');
37                 this.ngOnInit();
38             },
39             error: err => {
40                 console.log(`Error al eliminar Registro ${err}`);
41             }
42         });
43     }
44 }
```

Código lista-mascotas.component.html los códigos se podrán visualizar mejor en github:

```
listo-mascotas.component.html
src > app > lista-mascotas > listo-mascotas.component.html > div.container.mt-4 > nav.navbar.navbar-expand-lg.navbar-light.bg-light.mb-4
1  <div class="container mt-4" style="background-color: #00BCBA; min-height: 100vh; padding: 20px;">
2    <div class="logo" style="text-align: center; margin-bottom: 20px;">
3      
7        <a class="navbar-brand" href="#"></a>
8        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
9          <span class="navbar-toggler-icon"></span>
10         </button>
11        <div class="collapse navbar-collapse" id="navbarNav">
12          <ul class="navbar-nav">
13            <li class="nav-item">
14              <a class="nav-link" [routerLink]="/clientes">Clientes</a>
15            </li>
16            <li class="nav-item">
17              <a class="nav-link" [routerLink]="/solicitudes">Solicitudes</a>
18            </li>
19          </ul>
20        </div>
21      </nav>
22
23      <div class="row mb-3">
24        <div class="col-sm-12">
25          <form class="form-inline">
26            <fieldset class="form-group col-sm-11"></fieldset>
27            <fieldset class="form-group col-sm-1">
28              <a class="btn btn-primary" [routerLink]="/mascotas/agregar">Nueva Mascota</a>
29            </fieldset>
30          </form>
31        </div>
32      </div>
33      <div class="row">
34        <div class="col-sm-6 col-md-4 mb-4" *ngFor="let mascota of mascotas">
35          <div class="card" style="position: relative;"><!-- Añadir posición relativa -->
36            <!-- ID de la mascota en la esquina superior derecha -->
37            <div class="mascota-id" style="position: absolute; top: 10px; right: 10px; background-color: rgba(0,0,0,0.7); color: white; padding: 5px; border-radius: 5px;">
38              ID: {{ mascota.id }}</div>
39
40            <!-- Centrar la imagen y ajustar tamaño -->
41            <div class="card-img-top" style="height: 250px; overflow: hidden; display: flex; justify-content: center; align-items: center;">
42              <img [src]="mascota.foto" alt="{{ mascota.nombre }}" style="height: 100%; width: auto; max-width: 100%;">
43            </div>
44
45            <div class="card-body">
46              <h5 class="card-title">{{ mascota.nombre }}</h5>
47
48              <button class="btn btn-primary" (click)="toggleMostrar(mascota.id)">
49                {{ mostrarEstado[mascota.id] ? 'Click para ocultar información' : 'Click para ver más información' }}</button>
50
51            <!-- Mostrar toda la información -->
52            <div *ngIf="mostrarEstado[mascota.id]" class="mt-3">
53              <p class="card-text">
54                <strong>Especie:</strong> {{ mascota.especie }}<br>
55                <strong>Raza:</strong> {{ mascota.raza }}<br>
56                <strong>Edad:</strong> {{ mascota.edad }} años<br>
57                <strong>Sexo:</strong> {{ mascota.sexo }}<br>
58                <strong>Color:</strong> {{ mascota.color }}<br>
59                <strong>Descripción:</strong> {{ mascota.descripcion }}<br>
60                <strong>Estado:</strong> {{ mascota.estado }}<br>
61                <strong>Fecha de ingreso:</strong> {{ mascota.fecha_ingreso | date:'dd/MM/yyyy' }}<br>
62              </p>
63              <a class="btn btn-info" [routerLink]="/mascotas/editar/", {{ mascota.id }}>Editar</a>
64              <a class="btn btn-danger" (click)="borrarMascota(mascota.id)">Borrar</a>
65            </div>
66          </div>
67        </div>
68      </div>
69    </div>
70  </div>
71</div>
```

Para el caso de **editar-mascotas.component.ts** el código quedo así:

```
editor-mascotas.component.ts X
src > app > editar-mascotas > editor-mascotas.component.ts > EditorMascotasComponent > ngOnInit
1 import { Component, OnInit } from '@angular/core';
2 import { MascotaModel } from './shared/mascota.model';
3 import { MascotaService } from '../shared/mascota.service';
4 import { ActivatedRoute, Router } from '@angular/router';
5
6 @Component({
7   selector: 'app-editar-mascotas',
8   templateUrl: './editar-mascotas.component.html',
9   styleUrls: ['./editar-mascotas.component.css']
10 })
11 export class EditorMascotasComponent implements OnInit {
12   idMascota = '';
13   mascota = new MascotaModel('', '', '');
14
15   constructor(private mascotaService: MascotaService, private route: ActivatedRoute, private router: Router) {}
16
17 ngOnInit() {
18   this.idMascota = this.route.snapshot.params['idMascota'];
19   console.log(`El idMascota es ${this.idMascota}`);
20
21   if (this.idMascota) {
22     console.log('La solicitud viene de Edita');
23     this.mascotaService.obtenerMascota(this.idMascota).subscribe({
24       next: data => {
25         this.mascota = data;
26       },
27       error: err => {
28         console.log(`Error ${err}`);
29       }
30     });
31   } else {
32     console.log('La solicitud viene de Nueva Mascota');
33   }
34 }
35
36 onSubmit() {
37   console.log(`On Submit`, this.mascota);
38   if (this.mascota.id) {
39     this.mascotaService.actualizarMascota(this.mascota).subscribe({
40       next: data => {
41         console.log(data);
42         this.router.navigate(['/mascotas']);
43       },
44       error: err => {
45         console.log(`Error al actualizar ${err}`);
46       }
47     });
48   } else {
49     this.mascotaService.agregarMascota(this.mascota).subscribe({
50       next: data => {
51         console.log(data);
52         this.router.navigate(['/mascotas']);
53       },
54       error: err => {
55         console.log(`Error al Agregar ${err}`);
56       }
57     });
58   }
59 }
60 }
61
62
```

Código **editar-mascotas.component.html** :

```
editor-mascotas.component.html X
src > app > editar-mascotas > editar-mascotas.component.html > ...
1 <div class="container mt-5">
2   <h2 class="mb-4 text-center">Formulario de Mascotas</h2>
3   <form (ngSubmit)="onSubmit()" #mascotaForm="ngForm" class="bg-light p-4 rounded shadow">
4     <fieldset class="form-group">
5       <div class="mb-3">
6         <label for="nombre" class="form-label">Nombre</label>
7         <input type="text" class="form-control" required [(ngModel)]="mascota.nombre" name="nombre" placeholder="Ingresa el nombre">
8       </div>
9     </fieldset>
10
11    <fieldset class="form-group">
12      <div class="mb-3">
13        <label for="especie" class="form-label">Especie</label>
14        <input type="text" class="form-control" required [(ngModel)]="mascota.especie" name="especie" placeholder="Ingresa la especie">
15      </div>
16    </fieldset>
17
18    <fieldset class="form-group">
19      <div class="mb-3">
20        <label for="raza" class="form-label">Raza</label>
21        <input type="text" class="form-control" [(ngModel)]="mascota.raza" name="raza" placeholder="Ingresa la raza">
22      </div>
23    </fieldset>
24
25    <fieldset class="form-group">
26      <div class="mb-3">
27        <label for="edad" class="form-label">Edad</label>
28        <input type="number" class="form-control" required [(ngModel)]="mascota.edad" name="edad" placeholder="Ingresa la edad">
29      </div>
30    </fieldset>
```

```

<fieldset class="Form-group">
  <div class="mb-3">
    <label for="sexo" class="form-label">Sexo</label>
    <select class="form-select" required [(ngModel)]="mascota.sexo" name="sexo">
      <option value="" disabled selected>Selecciona el sexo</option>
      <option value="Macho">Macho</option>
      <option value="Hembra">Hembra</option>
    </select>
  </div>
</fieldset>

<fieldset class="Form-group">
  <div class="mb-3">
    <label for="color" class="form-label">Color</label>
    <input type="text" class="form-control" [(ngModel)]="mascota.color" name="color" placeholder="Ingresa el color">
  </div>
</fieldset>

<fieldset class="Form-group">
  <div class="mb-3">
    <label for="descripcion" class="form-label">Descripción</label>
    <textarea class="form-control" [(ngModel)]="mascota.descripcion" name="descripcion" rows="3" placeholder="Ingresa una descripción"></textarea>
  </div>
</fieldset>

<fieldset class="Form-group">
  <div class="mb-3">
    <label for="estado" class="form-label">Estado</label>
    <select class="form-select" [(ngModel)]="mascota.estado" name="estado">
      <option value="" disabled selected>Selecciona el estado</option>
      <option value="Disponible">Disponible</option>
      <option value="Adoptado">Adoptado</option>
    </select>
  </div>
</fieldset>

<fieldset class="Form-group">
  <div class="mb-3">
    <label for="fecha_ingreso" class="form-label">Fecha de Ingreso</label>
    <input type="date" class="form-control" [(ngModel)]="mascota.fecha_ingreso" name="fecha_ingreso">
  </div>
</fieldset>

<fieldset class="Form-group">
  <div class="mb-3">
    <label for="foto" class="form-label">URL de la Foto</label>
    <input type="text" class="form-control" [(ngModel)]="mascota.foto" name="Foto" placeholder="Ingresa la URL de la foto">
  </div>
</fieldset>

<div class="mb-3">
  <button type="submit" class="btn btn-primary" [disabled]="!mascotaForm.valid">Enviar</button>
</div>
</form>
</div>

```

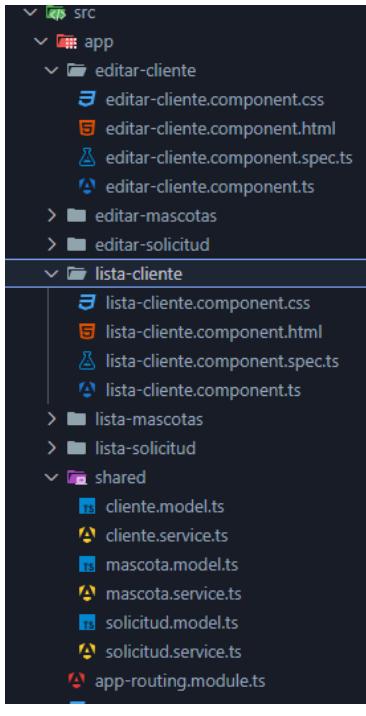
En los anteriores códigos se puede ver que se cumple con el requerimiento de **uso de ngModel**, **RouterLink** y **Servicios** dichos puntos fueron esenciales para que el sistema pueda ejecutarse de manera correcta de acuerdo lo visto en clases.

Después, se hizo el mismo proceso con cliente a continuación los comandos utilizados:

- PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> **ng generate service shared/cliente**
CREATE src/app/shared/cliente.service.spec.ts (378 bytes)
CREATE src/app/shared/cliente.service.ts (145 bytes)
- PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> **ng generate component listaCliente**
CREATE src/app/lista-cliente/lista-cliente.component.html (29 bytes)
CREATE src/app/lista-cliente/lista-cliente.component.spec.ts (663 bytes)
CREATE src/app/lista-cliente/lista-cliente.component.ts (236 bytes)
CREATE src/app/lista-cliente/lista-cliente.component.css (0 bytes)
UPDATE src/app/app.module.ts (1167 bytes)
- PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> **ng generate component editarCliente**
CREATE src/app/editar-cliente/editar-cliente.component.html (30 bytes)
CREATE src/app/editar-cliente/editar-cliente.component.spec.ts (670 bytes)
CREATE src/app/editar-cliente/editar-cliente.component.ts (240 bytes)
CREATE src/app/editar-cliente/editar-cliente.component.css (0 bytes)
UPDATE src/app/app.module.ts (1283 bytes)
- PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> █

De igual manera, en **TallerfrontEndPawpal**, se generaron un servicio y dos componentes, el servicio **shared/cliente**, además, se generaron los componentes **listaCliente** y **editarCliente**, los cuales permiten visualizar una lista de clientes, crear, editar y eliminar su información, respectivamente, y nuevamente el archivo **app.module.ts** fue actualizado automáticamente para declarar estos nuevos componentes en el módulo raíz de la aplicación.

En la carpeta podemos ver que se agregaron correctamente:



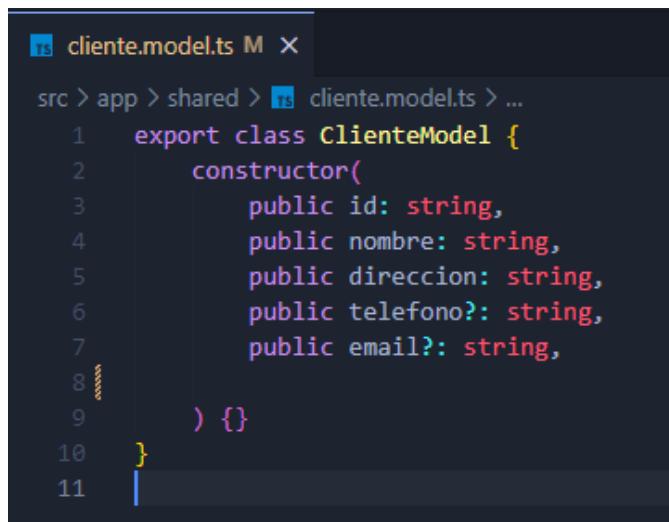
el archivo **cliente.service.ts** quedo así:

```

cliente.service.ts ×
src > app > shared > cliente.service.ts > ClienteService
1 import { HttpClient } from '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { ClienteModel } from './cliente.model';
4
5 @Injectable({
6   providedIn: 'root'
7 })
8 export class ClienteService {
9   BASE_URL = 'http://localhost:4000'; // URL base de La API
10
11   constructor(private http: HttpClient) {}
12
13   // Obtener todos Los clientes
14   obtenerClientes() {
15     return this.http.get<ClienteModel[]>(`${this.BASE_URL}/cliente/buscarClientes`);
16   }
17
18   // Buscar cliente por ID
19   obtenerCliente(idCliente: string) {
20     return this.http.get<ClienteModel>(`${this.BASE_URL}/cliente/buscarClienteId/${idCliente}`);
21   }
22
23   // Agregar nuevo cliente
24   agregarCliente(cliente: ClienteModel) {
25     return this.http.post<ClienteModel>(`${this.BASE_URL}/cliente/crearCliente`, cliente);
26   }
27
28   // Actualizar cliente existente
29   actualizarCliente(cliente: ClienteModel) {
30     return this.http.put<ClienteModel>(`${this.BASE_URL}/cliente/actualizarCliente/${cliente.id}`, cliente);
31   }
32
33   // Eliminar cliente
34   borrarCliente(idCliente: string) {
35     return this.http.delete<void>(`${this.BASE_URL}/cliente/eliminarCliente/${idCliente}`);
36   }
37 }
38

```

y también se le creo un modelo el cual se llama **cliente.model.ts** el cual quedo así para su funcionamiento:

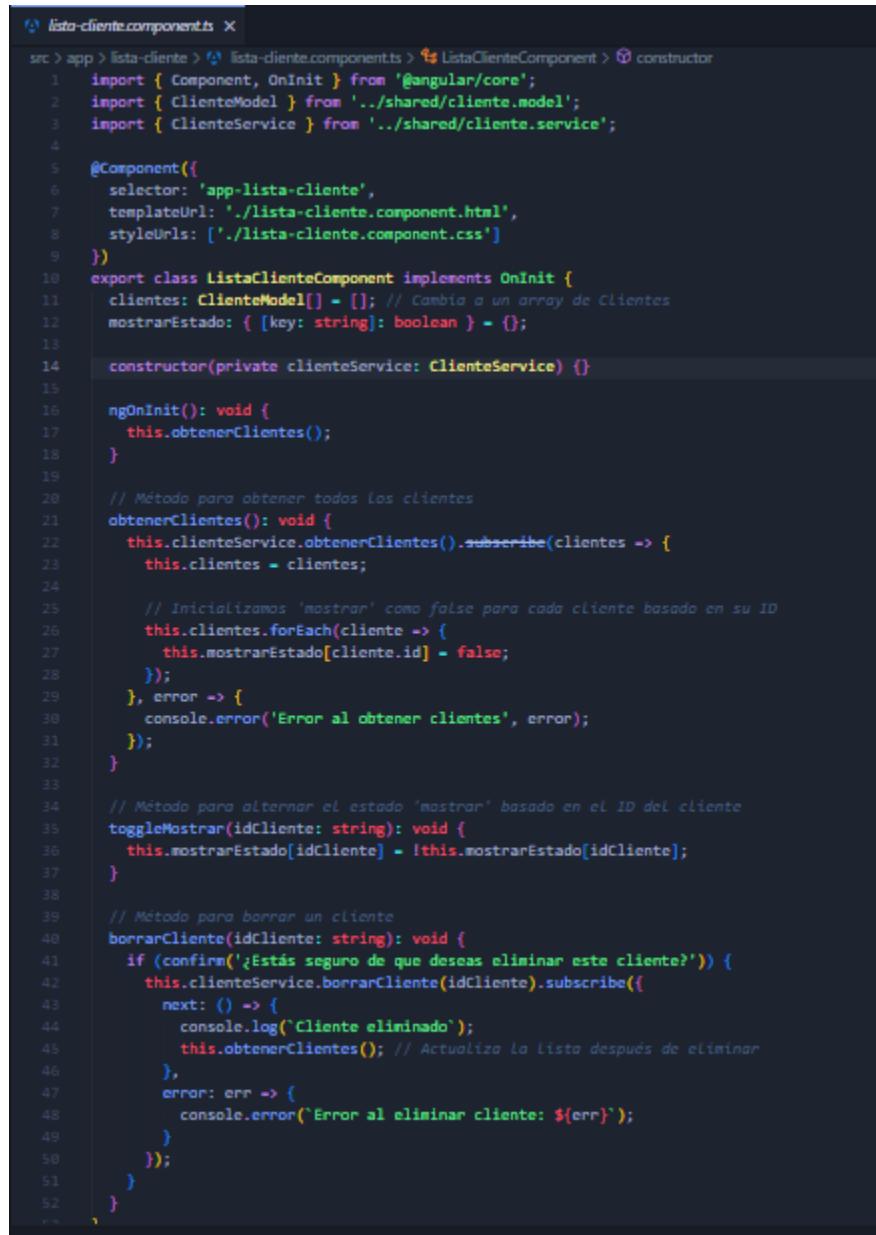


```

cliente.model.ts M X
src > app > shared > cliente.model.ts > ...
1  export class ClienteModel {
2    constructor(
3      public id: string,
4      public nombre: string,
5      public direccion: string,
6      public telefono?: string,
7      public email?: string,
8    ) {}
9
10 }
11

```

En el caso de **lista-cliente.component.ts** su código respectivo es el siguiente:



```
src > app > Lista-cliente > lista-cliente.component.ts > ListaClienteComponent > constructor
1 import { Component, OnInit } from '@angular/core';
2 import { ClienteModel } from '../shared/cliente.model';
3 import { ClienteService } from '../shared/cliente.service';
4
5 @Component({
6   selector: 'app-lista-cliente',
7   templateUrl: './lista-cliente.component.html',
8   styleUrls: ['./lista-cliente.component.css']
9 })
10 export class ListaClienteComponent implements OnInit {
11   clientes: ClienteModel[] = [];
12   mostrarEstado: { [key: string]: boolean } = {};
13
14   constructor(private clienteService: ClienteService) {}
15
16   ngOnInit(): void {
17     this.obtenerClientes();
18   }
19
20   // Método para obtener todos los clientes
21   obtenerClientes(): void {
22     this.clienteService.obtenerClientes().subscribe(clientes => {
23       this.clientes = clientes;
24
25       // Inicializamos 'mostrar' como false para cada cliente basado en su ID
26       this.clientes.forEach(cliente => {
27         this.mostrarEstado[cliente.id] = false;
28       });
29     }, error => {
30       console.error('Error al obtener clientes', error);
31     });
32   }
33
34   // Método para alternar el estado 'mostrar' basado en el ID del cliente
35   toggleMostrar(idCliente: string): void {
36     this.mostrarEstado[idCliente] = !this.mostrarEstado[idCliente];
37   }
38
39   // Método para borrar un cliente
40   borrarCliente(idCliente: string): void {
41     if (confirm(`Estás seguro de que deseas eliminar este cliente?`)) {
42       this.clienteService.borrarCliente(idCliente).subscribe({
43         next: () => {
44           console.log('Cliente eliminado');
45           this.obtenerClientes(); // Actualiza la lista después de eliminar
46         },
47         error: err => {
48           console.error(`Error al eliminar cliente: ${err}`);
49         }
50       });
51     }
52   }
53 }
```

Código **lista-cliente.component.html** los códigos se podrán visualizar mejor en [github](#):

```

  □ Liso-cliente.component.html ×
src > app > lista-cliente > □ lista-cliente.component.html > □ div.container.mt-4 > □ div.logo
1  <div class="container mt-4" style="background-color: #00BCD4; min-height: 100vh; padding: 20px;">
2    <!-- Logo en La parte Superior y centrado -->
3    <div class="logo" style="text-align: center; margin-bottom: 20px;">
4      
8      <a class="navbar-brand" href="#"></a>
9      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
10        <span class="navbar-toggler-icon"></span>
11      </button>
12      <div class="collapse navbar-collapse" id="navbarNav">
13        <ul class="navbar-nav">
14          <li class="nav-item">
15            <a class="nav-link" [routerLink]="/mascota">Mascotas</a>
16          </li>
17          <li class="nav-item">
18            <a class="nav-link" [routerLink]="/solicitudes">Solicitudes</a>
19          </li>
20        </ul>
21      </div>
22    </nav>
23
24    <div class="row mb-3">
25      <div class="col-sm-12">
26        <form class="form-inline">
27          <fieldset class="form-group col-sm-11"></fieldset>
28          <fieldset class="form-group col-sm-1">
29            <a class="btn btn-primary" [routerLink]="/clientes/agregar">Nuevo Cliente</a>
30          </fieldset>
31        </form>
32      </div>
33    </div>
34
35    <div class="card">
36      <div class="card-header">
37        <h5 class="mb-0">Lista de Clientes</h5>
38      </div>
39      <div class="card-body">
40        <table class="table table-striped table-hover">
41          <thead>
42            <tr>
43              <th>ID</th>
44              <th>Nombre</th>
45              <th>Dirección</th>
46              <th>Teléfono</th>
47              <th>Email</th>
48              <th>Acciones</th>
49            </tr>
50          </thead>
51          <tbody>
52            <tr *ngFor="let cliente of clientes">
53              <td>{{ cliente.id }}</td>
54              <td>{{ cliente.nombre }}</td>
55              <td>{{ cliente.direccion }}</td>
56              <td>{{ cliente.telefono }}</td>
57              <td>{{ cliente.email }}</td>
58              <td>
59                <a class="btn btn-info btn-sm" [routerLink]="/clientes/editar", cliente.id>Editar</a>
60                <a class="btn btn-danger btn-sm" (click)="borrarCliente(cliente.id)">Borrar</a>
61              </td>
62            </tr>
63          </tbody>
64        </table>
65      </div>
66    </div>
67  </div>

```

En el caso de **editar-cliente.component.ts** su código respectivo es el siguiente:

```
src > app > editar-cliente > editor-cliente.component.ts > EditorClienteComponent > onsubmit > next
1  import { Component, OnInit } from '@angular/core';
2  import { ActivatedRoute, Router } from '@angular/router';
3  import { ClienteService } from './shared/cliente.service';
4  import { ClienteModel } from './shared/cliente.model';
5
6  @Component({
7    selector: 'app-editar-cliente',
8    templateUrl: './editar-cliente.component.html',
9    styleUrls: ['./editar-cliente.component.css']
10   })
11 export class EditorClienteComponent implements OnInit {
12   idCliente = '';
13   cliente: ClienteModel = new ClienteModel('', '', '', '');
14
15   constructor(
16     private clienteService: ClienteService,
17     private route: ActivatedRoute,
18     private router: Router
19   ) {}
20
21   ngOnInit() {
22     this.idCliente = this.route.snapshot.params['idCliente'];
23     if (this.idCliente) {
24       this.clienteService.obtenerCliente(this.idCliente).subscribe({
25         next: (data) => {
26           this.cliente = data;
27         },
28         error: (err) => {
29           console.error('Error al obtener el cliente:', err);
30         }
31       });
32     }
33   }
34
35   onsubmit() {
36     if (this.cliente.id) {
37       this.clienteService.actualizarCliente(this.cliente).subscribe({
38         next: () => {
39           this.router.navigate(['/clientes']);
40         },
41         error: (err) => {
42           console.error('Error al actualizar el cliente:', err);
43         }
44       });
45     } else {
46       // Add a new client
47       this.clienteService.agregarCliente(this.cliente).subscribe({
48         next: () => {
49           this.router.navigate(['/clientes']);
50         },
51         error: (err) => {
52           console.error('Error al agregar el cliente:', err);
53         }
54       });
55     }
56   }
57 }
58 }
```

En el caso de **editar-cliente.component.html** su código respectivo es el siguiente:

```
src > app > editar-cliente > editor-cliente.component.html > div.container.mt-4 > form > div.form-group
1  <div class="container mt-4">
2  <h2>{{ cliente.id ? 'Editar Cliente' : 'Nuevo Cliente' }}</h2>
3  <form (ngSubmit)="onSubmit()" #clienteForm="ngForm">
4    <div class="form-group">
5      <label for="nombre">Nombre</label>
6      <input type="text" class="form-control" id="nombre" [(ngModel)]="cliente.nombre" name="nombre" required>
7    </div>
8    <div class="form-group">
9      <label for="direccion">Dirección</label>
10     <input type="text" class="form-control" id="direccion" [(ngModel)]="cliente.direccion" name="direccion" required>
11   </div>
12   <div class="form-group">
13     <label for="telefono">Teléfono</label>
14     <input type="text" class="form-control" id="telefono" [(ngModel)]="cliente.telefono" name="telefono" required>
15   </div>
16   <div class="form-group">
17     <label for="email">Email</label>
18     <input type="email" class="form-control" id="email" [(ngModel)]="cliente.email" name="email" required>
19   </div>
20   <div class="d-flex justify-content-between mt-4">
21     <button type="submit" class="btn btn-primary" [disabled]="!clienteForm.valid">Guardar Cliente</button>
22     <a class="btn btn-secondary" [routerLink]="/clientes">Cancelar</a>
23   </div>
24 </form>
25 </div>
```

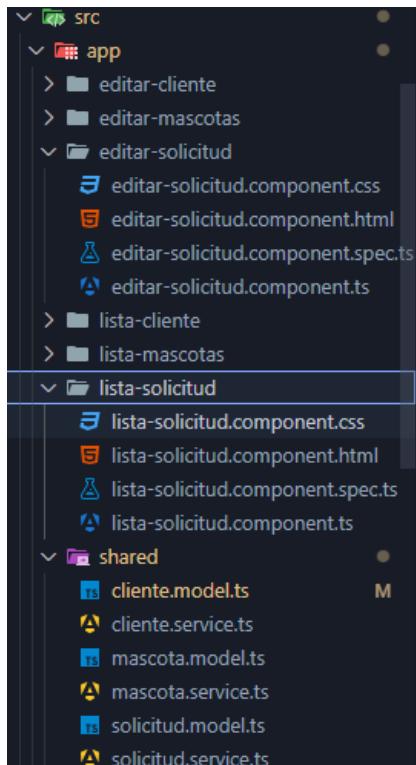
En los anteriores códigos de la implementación de cliente se puede ver que se cumple con el requerimiento de **uso de ngModel, RouterLink y Servicios** dichos puntos fueron esenciales para que el sistema pueda ejecutarse de manera correcta de acuerdo lo visto en clases.

Finalmente se hizo lo mismo para solicitudes lo comandos utilizados fueron:

```
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng generate service shared/solicitud
CREATE src/app/shared/solicitud.service.spec.ts (388 bytes)
CREATE src/app/shared/solicitud.service.ts (147 bytes)
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng generate component listaSolicitud
CREATE src/app/lista-solicitud/lista-solicitud.component.html (31 bytes)
CREATE src/app/lista-solicitud/lista-solicitud.component.spec.ts (677 bytes)
CREATE src/app/lista-solicitud/lista-solicitud.component.ts (244 bytes)
CREATE src/app/lista-solicitud/lista-solicitud.component.css (0 bytes)
UPDATE src/app/app.module.ts (1403 bytes)
● PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal> ng generate component editarSolicitud
CREATE src/app/editar-solicitud/editar-solicitud.component.html (32 bytes)
CREATE src/app/editar-solicitud/editar-solicitud.component.spec.ts (684 bytes)
CREATE src/app/editar-solicitud/editar-solicitud.component.ts (248 bytes)
CREATE src/app/editar-solicitud/editar-solicitud.component.css (0 bytes)
UPDATE src/app/app.module.ts (1527 bytes)
○ PS C:\Users\7Jrmo7\Documents\Diplomado2024B\TallerfrontEndPawpal>
```

Nuevamente, en **TallerfrontEndPawpal**, se generaron un servicio y dos componentes para manejar las solicitudes, el servicio **shared/solicitud** encapsula la lógica de negocio, mientras que los componentes **listaSolicitudes** y **editarSolicitudes** permiten visualizar, crear, editar y eliminar solicitudes y nuevamente el archivo **app.module.ts** fue actualizado automáticamente para declarar estos componentes en el módulo.

En la carpeta podemos ver que se agregaron correctamente:



el archivo **solicitud.service.ts** quedo así:

```
solicitud.service.ts ×
src > app > shared > solicitud.service.ts > SolicitudService > constructor
1 // src/app/shared/solicitud.service.ts
2 import { Injectable } from '@angular/core';
3 import { HttpClient } from '@angular/common/http';
4 import { Observable } from 'rxjs';
5 import { SolicitudModel } from './solicitud.model';
6
7 @Injectable({
8   providedIn: 'root'
9 })
10 export class SolicitudService {
11   BASE_URL='http://localhost:4000/solicitud';
12
13   constructor(private http: HttpClient) {
14     //Lista completa mascotas
15   }
16
17   // Crear nueva solicitud
18   crearSolicitud(solicitud: SolicitudModel): Observable<any> {
19     return this.http.post(`${this.BASE_URL}/crearSolicitud`, solicitud); // Cambia La ruta según tu API
20   }
21
22   // Obtener todas las solicitudes
23   buscarSolicitudes(): Observable<SolicitudModel[]> {
24     return this.http.get<SolicitudModel[]>(`${this.BASE_URL}/buscarSolicitudes`); // Cambia La ruta según tu API
25   }
26
27   // Obtener solicitud por ID
28   buscarSolicitudId(id: number): Observable<SolicitudModel> {
29     return this.http.get<SolicitudModel>(`${this.BASE_URL}/buscarSolicitudId/${id}`); // Cambia La ruta según tu API
30   }
31
32   // Actualizar una solicitud existente
33   actualizarSolicitud(id: number, solicitud: SolicitudModel): Observable<any> {
34     return this.http.put(`${this.BASE_URL}/actualizarSolicitud/${id}`, solicitud); // Cambia La ruta según tu API
35   }
36
37   // Borrar una solicitud
38   eliminarSolicitud(id: number): Observable<any> {
39     return this.http.delete(`${this.BASE_URL}/eliminarSolicitud/${id}`); // Cambia La ruta según tu API
40   }
41 }
```

y también se le creo un modelo el cual se llama **solicitud.model.ts** el cual quedo así para su funcionamiento:

```

src > app > shared > solicitud.model.ts > ...
1  export class SolicitudModel {
2      constructor(
3          public id: number,
4          public fecha_solicitud: string,
5          public estado: string,
6          public clienteId: string,
7          public mascotaId: string,
8          public clienteNombre?: string, // Propiedad opcional
9          public mascotaNombre?: string // Propiedad opcional
10     ) {}
11 }
12

```

En el caso de **lista-solicitud.component.ts** su código respectivo es el siguiente:

```

src > app > lista-solicitud > lista-solicitud.component.ts > ListaSolicitudComponent
1  import { Component, OnInit } from '@angular/core';
2  import { SolicitudModel } from '../shared/solicitud.model';
3  import { SolicitudService } from '../shared/solicitud.service';
4  import { ClienteService } from '../shared/cliente.service';
5  import { MascotaService } from '../shared/mascota.service';
6  import { MascotaModel } from '../shared/mascota.model';
7  import { ClienteModel } from '../shared/cliente.model';
8
9  @Component({
10   selector: 'app-lista-solicitud',
11   templateUrl: './lista-solicitud.component.html',
12   styleUrls: ['./lista-solicitud.component.css']
13 })
14 export class ListaSolicitudComponent implements OnInit {
15   solicitudes: SolicitudModel[] = [];
16   clientes: ClienteModel[] = [];
17   mascotas: MascotaModel[] = [];
18
19   constructor(
20     private solicitudService: SolicitudService,
21     private clienteService: ClienteService,
22     private mascotaService: MascotaService
23   ) {}
24
25   ngOnInit(): void {
26     this.obtenerSolicitudes();
27     this.obtenerClientes();
28     this.obtenerMascotas();
29   }
30
31   obtenerSolicitudes(): void {
32     this.solicitudService.buscarSolicitudes().subscribe(solicitudes => {
33       this.solicitudes = solicitudes;
34       this.asignarNombres(); // Asigna nombres después de obtener solicitudes
35     }, error => {
36       console.error('Error al obtener solicitudes', error);
37     });
38   }
39
40   obtenerClientes(): void {
41     this.clienteService.obtenerClientes().subscribe(clientes => {
42       console.log('Clientes obtenidos:', clientes);
43       this.clientes = clientes;
44       this.asignarNombres(); // Asigna nombres si Los clientes están disponibles
45     }, error => {
46       console.error('Error al obtener clientes', error);
47     });
48   }
49
50   obtenerMascotas(): void {
51     this.mascotaService.obtenerMascotas().subscribe(mascotas => {
52       console.log('Mascotas obtenidas:', mascotas);
53       this.mascotas = mascotas;
54       this.asignarNombres(); // Asigna nombres si Las mascotas están disponibles
55     }, error => {
56       console.error('Error al obtener mascotas', error);
57     });
58   }
59
60   asignarNombres() {
61     this.solicitudes.forEach(solicitud => {
62       const cliente = this.clientes.find(c => c.id === solicitud.clienteId);
63       const mascota = this.mascotas.find(m => m.id === solicitud.mascotaId);
64       solicitud.clienteNombre = cliente ? cliente.nombre : 'Sin cliente';
65       solicitud.mascotaNombre = mascota ? mascota.nombre : 'Sin mascota';
66     });
67   }
68
69   borrarSolicitud(idSolicitud: number): void {
70     if (confirm('¿Estás seguro de que deseas eliminar esta solicitud?')) {
71       this.solicitudService.eliminarSolicitud(idSolicitud).subscribe({
72         next: () => {
73           console.log('Solicitud eliminada');
74           this.obtenerSolicitudes(); // Actualiza La lista después de eliminar
75         },
76         error: err => {
77           console.error(`Error al eliminar solicitud: ${err}`);
78         }
79       });
80     }
81   }
82 }
83

```

En el caso de **lista-solicitud.component.html** su código respectivo es el siguiente:

```
1  lista-solicitud.component.html <
2  src > app > lista-solicitud > lista-solicitud.component.html > div.container.mt-4 > nav.navbar.navbar-expand-lg.navbar-light.bg-light.mb-4 > div.navbarNav.collapse.navbar-collapse
3      <div class="container mt-4" style="background-color: #00BCD4; min-height: 100vh; padding: 20px;">
4          <div class="logo" style="text-align: center; margin-bottom: 20px;">
5              
6          </div>
7          <!-- Menú de navegación -->
8          <nav class="navbar navbar-expand-lg navbar-light bg-light mb-4">
9              <div class="nav-bar-brand" href="#"></a>
10             <button class="nav-bar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
11                 <span class="nav-bar-toggler-icon"></span>
12             </button>
13             <div class="collapse navbar-collapse" id="navbarNav">
14                 <ul class="nav-bar-nav">
15                     <li class="nav-item">
16                         <a class="nav-link" [routerLink]="/mascota">Mascotas</a>
17                     </li>
18                     <li class="nav-item">
19                         <a class="nav-link" [routerLink]="/clientes">Clientes</a>
20                     </li>
21                 </ul>
22             </div>
23         </nav>
24 
25         <div class="row mb-3">
26             <div class="col-sm-12">
27                 <form class="form-in-line">
28                     <fieldset class="form-group col-sm-11"></fieldset>
29                     <fieldset class="form-group col-sm-1">
30                         <a class="btn btn-primary" [routerLink]="/solicitudes/agregar">Nueva Solicitud</a>
31                     </fieldset>
32                 </form>
33             </div>
34         </div>
35 
36     <div class="card">
37         <div class="card-header">
38             <h5 class="mb-0">Lista de Solicituds</h5>
39         </div>
40         <div class="card-body">
41             <table class="table table-striped table-hover">
42                 <thead>
43                     <tr>
44                         <th>ID</th>
45                         <th>Fecha de Solicitud</th>
46                         <th>Estado</th>
47                         <th>Cliente</th>
48                         <th>Mascota</th>
49                         <th>Acciones</th>
50                     </tr>
51                 </thead>
52                 <tbody>
53                     <tr *ngFor="let solicitud of solicitudes">
54                         <td>{{ solicitud.id }}</td>
55                         <td>{{ solicitud.fecha_solicitud | date: 'dd/MM/yyyy' }}</td>
56                         <td>{{ solicitud.estado }}</td>
57                         <td>{{ solicitud.clienteNombre }}</td> <!-- Mostrar nombre del cliente -->
58                         <td>{{ solicitud.mascotaNombre }}</td> <!-- Mostrar nombre de la mascota -->
59                         <td>
60                             <a class="btn btn-info btn-sm" [routerLink]="/solicitudes/editar, solicitud.id">Editar</a>
61                             <a class="btn btn-danger btn-sm" (click)="borrarSolicitud(solicitud.id)">Borrar</a>
62                         </td>
63                     </tr>
64                 </tbody>
65             </table>
66         </div>
67     </div>
68 
```

En el caso de **editar-solicitud.component.ts** su código respectivo es el siguiente:

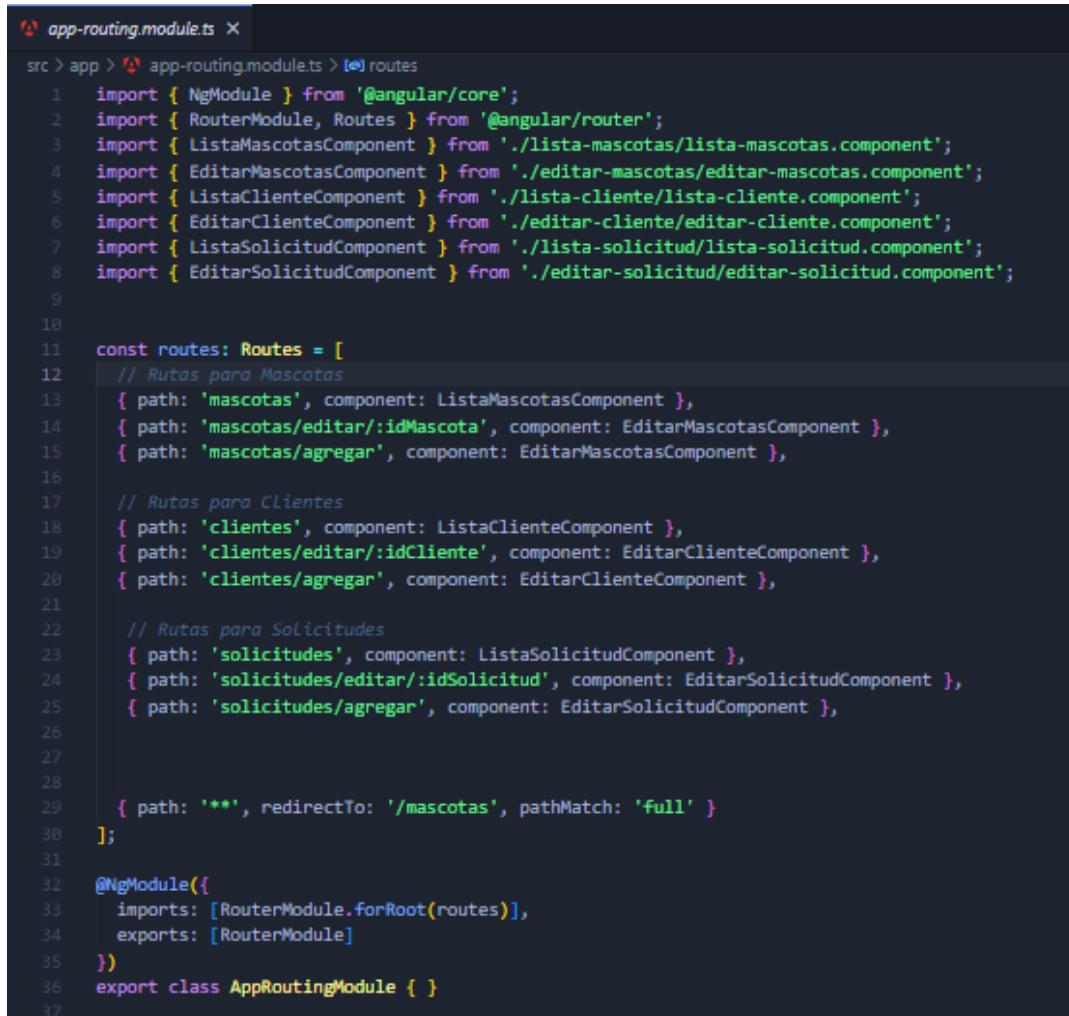
```
editor-solicitud.component.ts X
src > app > editar-solicitud > editor-solicitud.component.ts > EditorSolicitudComponent
  1 import { Component, OnInit } from '@angular/core';
  2 import { ActivatedRoute, Router } from '@angular/router';
  3 import { SolicitudModel } from '../shared/solicitud.model';
  4 import { SolicitudService } from '../shared/solicitud.service';
  5
  6 @Component({
  7   selector: 'app-editar-solicitud',
  8   templateUrl: './editar-solicitud.component.html',
  9   styleUrls: ['./editar-solicitud.component.css']
 10 })
 11 export class EditorSolicitudComponent implements OnInit {
 12   idSolicitud: string = ''; // Cambiar a string para obtener el id de la ruta
 13   solicitud: SolicitudModel = new SolicitudModel(0, '', 'Pendiente', '', '');
 14
 15   constructor(
 16     private solicitudService: SolicitudService,
 17     private route: ActivatedRoute,
 18     private router: Router
 19   ) {}
 20
 21   ngOnInit() {
 22     this.idSolicitud = this.route.snapshot.params['idSolicitud'];
 23     if (this.idSolicitud) {
 24       const id = Number(this.idSolicitud);
 25       this.solicitudService.buscarSolicitudId(id).subscribe({
 26         next: (data) => {
 27           this.solicitud = data;
 28         },
 29         error: (err) => {
 30           console.error('Error al obtener solicitud:', err);
 31         }
 32       });
 33     }
 34   }
 35
 36   onSubmit() {
 37     if (this.solicitud.id) {
 38       this.solicitudService.actualizarSolicitud(this.solicitud).subscribe({
 39         next: () => {
 40           this.router.navigate(['/solicitudes']);
 41         },
 42         error: (err) => {
 43           console.error('Error updating request:', err);
 44         }
 45       });
 46     } else {
 47       this.solicitudService.crearSolicitud(this.solicitud).subscribe({
 48         next: () => {
 49           this.router.navigate(['/solicitudes']);
 50         },
 51         error: (err) => {
 52           console.error('Error al actualizar la solicitud:', err);
 53         }
 54       });
 55     }
 56   }
 57 }
```

En el caso de **editar-solicitud.component.html** su código respectivo es el siguiente:

```
editor-solicitud.component.html X
src > app > editar-solicitud > editar-solicitud.component.html > div.container.mt-4 > form > div.form-group > select#estado.form-control > option
  1 <div class="container mt-4">
  2   <h2>{{ solicitud.id ? 'Editar Solicitud' : 'Nueva Solicitud' }}</h2>
  3   <form (ngSubmit)="onSubmit()" #solicitudForm="ngForm">
  4     <div class="form-group">
  5       <label for="estado">Estado</label>
  6       <select class="form-control" id="estado" [(ngModel)]="solicitud.estado" name="estado" required>
  7         <option value="Pendiente">Pendiente</option>
  8         <option value="Aprobada">Aprobada</option>
  9         <option value="Rechazada">Rechazada</option>
 10       </select>
 11     </div>
 12     <div class="form-group">
 13       <label for="clienteId">ID del Cliente</label>
 14       <input type="text" class="form-control" id="clienteId" [(ngModel)]="solicitud.clienteId" name="clienteId" required>
 15     </div>
 16     <div class="form-group">
 17       <label for="mascotaId">ID de la Mascota</label>
 18       <input type="text" class="form-control" id="mascotaId" [(ngModel)]="solicitud.mascotaId" name="mascotaId" required>
 19     </div>
 20     <div class="form-group">
 21       <label for="fecha_solicitud">Fecha de Solicitud</label>
 22       <input type="date" class="form-control" id="fecha_solicitud" [(ngModel)]="solicitud.fecha_solicitud" name="fecha_solicitud" required>
 23     </div>
 24     <div class="d-flex justify-content-between mt-4">
 25       <button type="submit" class="btn btn-primary" [disabled]="!solicitudForm.valid">Guardar Solicitud</button>
 26       <a class="btn btn-secondary" [routerLink]="/solicitudes">Cancelar</a>
 27     </div>
 28   </form>
 29 </div>
```

En todos los anteriores códigos tanto como los de mascota, cliente y el ultimo solicitud se puede ver que se cumple con el requerimiento de **uso de ngModel, RouterLink y Servicios** dichos puntos fueron esenciales para que el sistema pueda ejecutarse de manera correcta de acuerdo lo visto en clases, por otra parte, en github están los códigos para que se observen de mejor manera.

Como punto final y para tenerlo en cuenta también se hizo las rutas para que el sistema pueda navegar de forma óptima entre las interfaces:



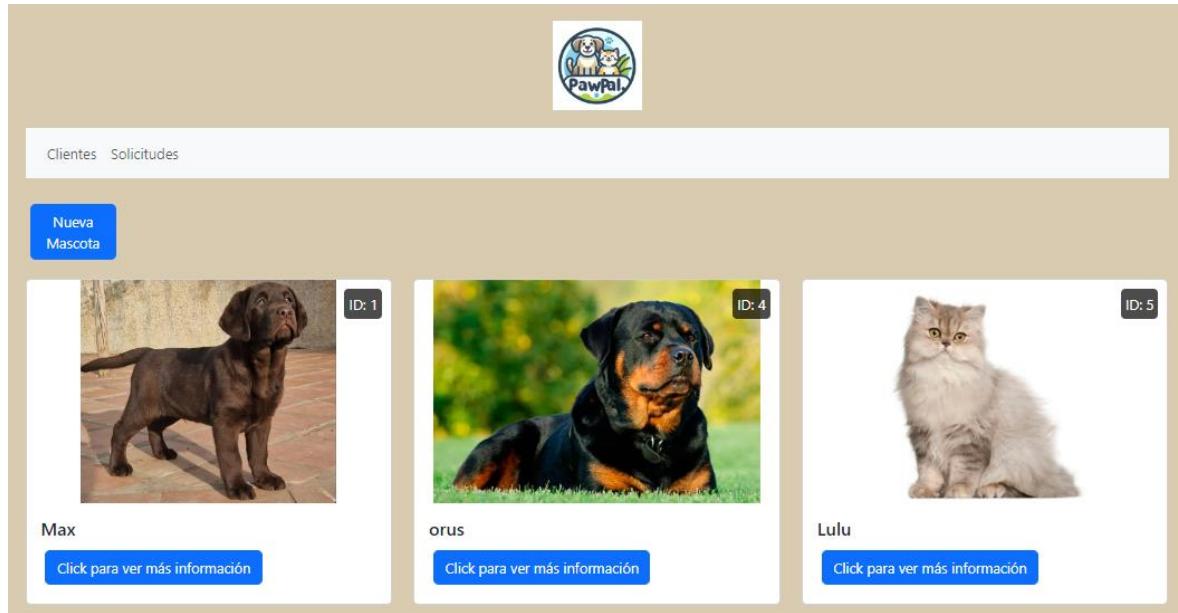
```
app-routing.module.ts
src > app > app-routing.module.ts > routes
1 import { NgModule } from '@angular/core';
2 import { RouterModule, Routes } from '@angular/router';
3 import { ListaMascotasComponent } from './lista-mascotas/lista-mascotas.component';
4 import { EditarMascotasComponent } from './editar-mascotas/editar-mascotas.component';
5 import { ListaClienteComponent } from './lista-cliente/lista-cliente.component';
6 import { EditarClienteComponent } from './editar-cliente/editar-cliente.component';
7 import { ListaSolicitudComponent } from './lista-solicitud/lista-solicitud.component';
8 import { EditarSolicitudComponent } from './editar-solicitud/editar-solicitud.component';
9
10
11 const routes: Routes = [
12   // Rutas para Mascotas
13   { path: 'mascotas', component: ListaMascotasComponent },
14   { path: 'mascotas/editar/:idMascota', component: EditarMascotasComponent },
15   { path: 'mascotas/agregar', component: EditarMascotasComponent },
16
17   // Rutas para Clientes
18   { path: 'clientes', component: ListaClienteComponent },
19   { path: 'clientes/editar/:idCliente', component: EditarClienteComponent },
20   { path: 'clientes/agregar', component: EditarClienteComponent },
21
22   // Rutas para Solicitudes
23   { path: 'solicitudes', component: ListaSolicitudComponent },
24   { path: 'solicitudes/editar/:idSolicitud', component: EditarSolicitudComponent },
25   { path: 'solicitudes/agregar', component: EditarSolicitudComponent },
26
27
28   { path: '**', redirectTo: '/mascotas', pathMatch: 'full' }
29 ];
30
31
32 @NgModule({
33   imports: [RouterModule.forRoot(routes)],
34   exports: [RouterModule]
35 })
36 export class AppRoutingModule { }
```

2. Uso de HTML 5 y JavaScript (Se debe desarrollar una estructura ordenada, con código legible y documentado).

En el proyecto TallerfrontEndPawpal, se puede evidenciar en las imágenes del primer punto el uso de HTML5 y JavaScript (TypeScript), además para mayor visibilidad todo el código fuente esta subido al repositorio de GitHub, la estructura del proyecto está organizada de manera clara y ordenada, siguiendo las mejores prácticas de desarrollo, se emplean etiquetas semánticas de HTML5 para crear interfaces web responsivas y accesibles, mientras que la lógica de la aplicación está desarrollada en TypeScript, además, el código está documentado, lo que facilita su mantenimiento y evolución.

3. Estilos CSS (Uso de Bootstrap), se debe generar una interface ordenada estructurada y agradable para el usuario final.

Al iniciar como principal se encuentra la interfaz de mascotas donde primeramente está el logo de la empresa **Pawpal** y luego un menú de navegación para las otras interfaces el menú es un estilo de **Bootstrap**, con el fin de cumplir el requerimiento del punto se organizan en un estilo tambien **Bootstrap** de tarjeta y conforme se van agregando se van organizando a continuación una imagen ejemplo:



Cada tarjeta contiene la imagen y nombre de la mascota, además un botón que al darle click muestra la información de la mascota y los botones para editar y eliminar el botón agregar esta debajo del menú, cabe resaltar que a los botones también se les aplico un estilo de **Bootstrap**, a continuación, las imágenes para evidenciar lo dicho:

This screenshot shows a detailed view of the pet management application, focusing on the three cards from the previous screenshot with their details expanded:

- Max (ID: 1)**:
 - Especie:** Perro
 - Raza:** Labrador
 - Edad:** 4 años
 - Sexo:** Macho
 - Color:** Marrón
 - Descripción:** Un perro muy energético y amigable.
 - Estado:** Disponible
 - Fecha de ingreso:** 09/09/2024

Buttons: 'Editar' (blue) and 'Borrar' (red).
- Orus (ID: 4)**:
 - Especie:** Perro
 - Raza:** Rotwailer
 - Edad:** 5 años
 - Sexo:** Macho
 - Color:** Negro
 - Descripción:** Un perro muy fiel y amigable.
 - Estado:** Disponible
 - Fecha de ingreso:** 11/09/2024

Buttons: 'Editar' (blue) and 'Borrar' (red).
- Lulu (ID: 5)**:
 - Especie:** Gato
 - Raza:** Persa
 - Edad:** 3 años
 - Sexo:** Hembra
 - Color:** Blanca
 - Descripción:** Perezosa pero muy amigable.
 - Estado:** Disponible
 - Fecha de ingreso:** 20/09/2024

Buttons: 'Editar' (blue) and 'Borrar' (red).

Below these cards are two additional cards partially visible:

- ID: 6**: A yellow Labrador Retriever.
- ID: 7**: A Malinois dog.

Formulario nueva mascota

A continuación, se mostrará como se crea una nueva mascota y se hará la verificación tanto en la página como en la base de datos a través de dBeaver.

Nombre
Pancho

Especie
Gato

Raza
Maine Coon

Edad
4

Sexo
Hembra

Color
Marrón y dorado

Descripción
Encantador Maine Coon con un pelaje atigrado que brilla con tonos dorados y marrones.

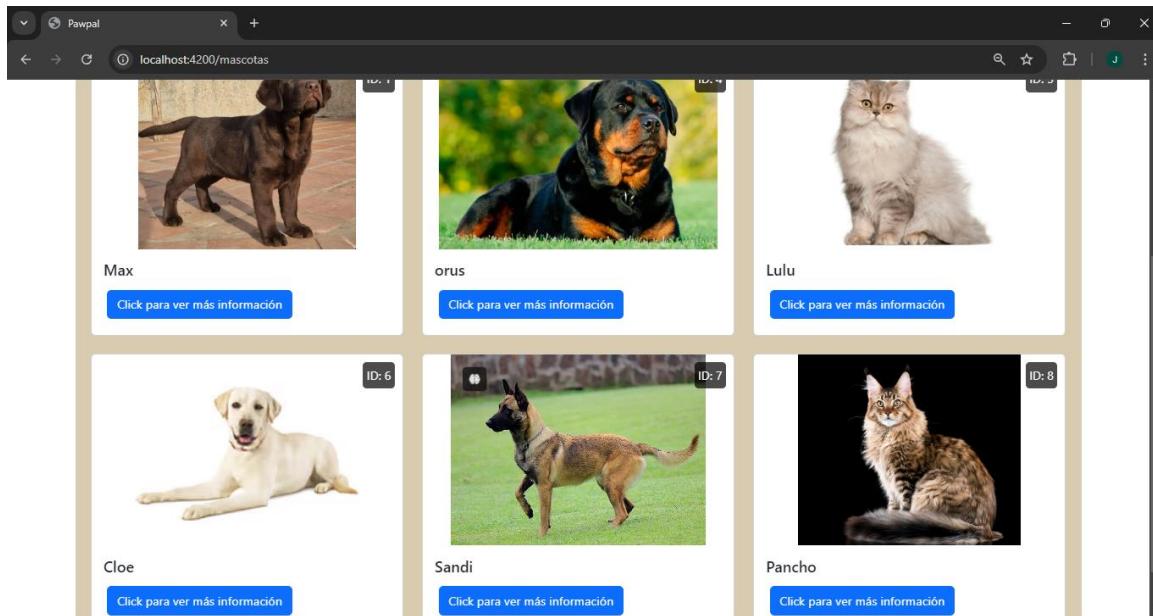
Estado
Disponible

Fecha de Ingreso
01/10/2024

URL de la Foto
https://www.zooplus.es/magazine/wp-content/uploads/2018/08/main-coon-3-768x658.jpg

Enviar

Al darle enviar se crea en la pagina y en la base de datos:



Base de datos:

	A-Z nombre	A-Z especie	A-Z raza	I23 edad	A-Z sexo	A-Z color	A-Z descripción	A-Z estado
1	Max	Perro	Labrador	4	Macho	Marrón	Un perro muy enérgico y amigable.	Disponible
4	orus	Perro	Rotwailer	5	Macho	Negro	Un perro muy fiel y amigable.	Disponible
5	Lulu	Gato	Persa	3	Hembra	Blanca	Perezosa pero muy amigable.	Disponible
6	Cloe	Perro	Labrador	5	Hembra	Blanca	Muy juguetona y fiel.	Disponible
7	Sandi	Perro	Belga malinois	5	Macho	Marrón	Un perro muy protector, amigable y	Disponible
8	Pancho	Gato	Maine Coon	4	Hembra	Marrón y dorado	Encantador Maine Coon con un pelaje atigrado que brilla con tonos dorados y marrones.	Disponible

Su información completa:


ID: 8

Pancho

Click para ocultar información

Especie: Gato
Raza: Maine Coon
Edad: 4 años
Sexo: Hembra
Color: Marrón y dorado
Descripción: Encantador Maine Coon con un pelaje atigrado que brilla con tonos dorados y marrones.
Estado: Disponible
Fecha de ingreso: 30/09/2024

[Editar](#)
[Borrar](#)

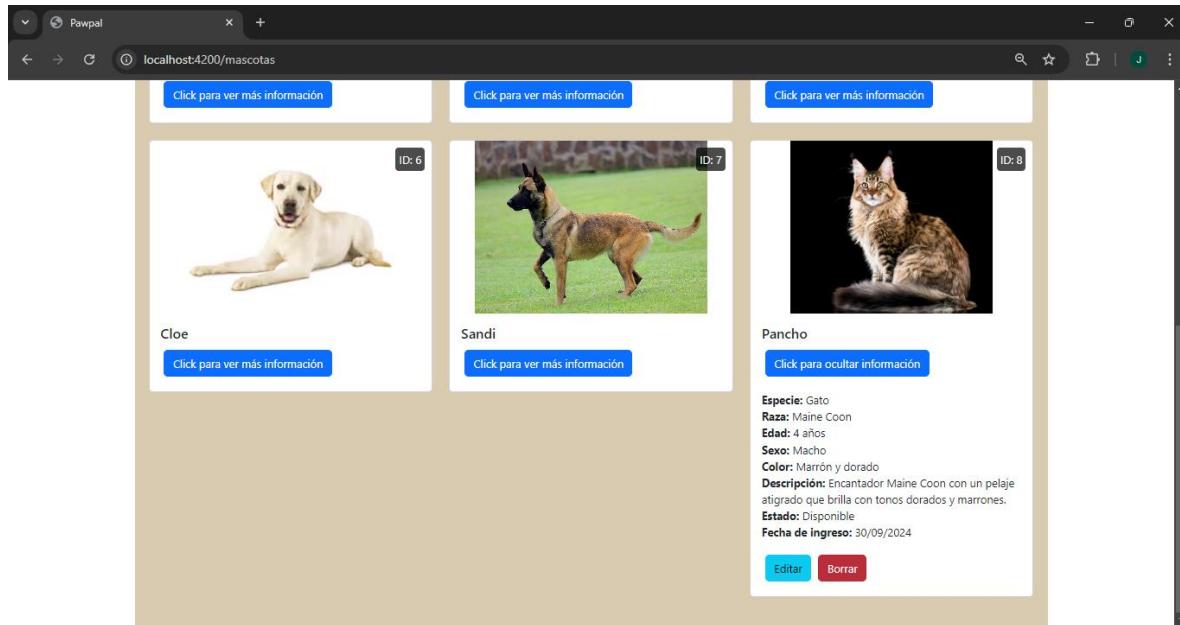
Para efectos de prueba del botón editar se puso sexo **hembra** de ese modo el formulario de edición quedaría o se visualizaría así:

Formulario de Mascotas

Nombre	<input type="text" value="Pancho"/>
Especie	<input type="text" value="Gato"/>
Raza	<input type="text" value="Maine Coon"/>
Edad	<input type="text" value="4"/>
Sexo	<div style="border: 1px solid #ccc; padding: 5px; width: fit-content; margin: auto;"> <input checked="" type="button" value="Hembra"/> <input type="button" value="Selección el sexo"/> <input type="button" value="Macho"/> <input type="button" value="Hembra"/> </div>
Descripción	<input type="text" value="Encantador Maine Coon con un pelaje atigrado que brilla con tonos dorados y marrones."/>

Como podemos evidenciar en la barra de búsqueda la ruta para cada formulario cambia, en este caso también los datos de la mascota fueron traídos directamente al formulario para poderlos modificar, se selecciona el dato correcto y se da enviar para que se haga efectivo el cambio:

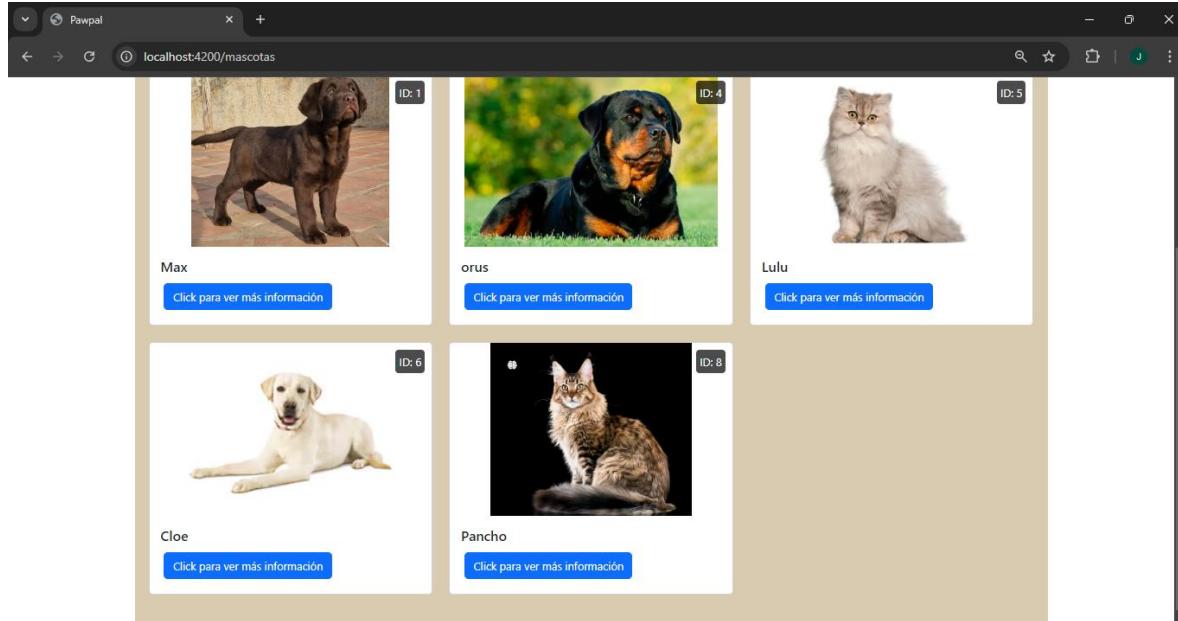
Una vez le damos enviar nos redirecciona a la pagina principal [/mascotas](#) y podemos verificar la edición:



En la base de datos:

	#	nombre	especie	raza	edad	sexo	color	descripción	estado
Grilla	1	Max	Perro	Labrador	4	Macho	Marrón	Un perro muy energético y amigable.	Disponible
Textos	2	orus	Perro	Rotwailer	5	Macho	Negro	Un perro muy fiel y amigable.	Disponible
Grilla	3	Lulu	Gato	Persa	3	Hembra	Blanca	Perezosa pero muy amigable.	Disponible
Textos	4	Cloe	Perro	Labrador	5	Hembra	Blanca	Muy juguetona y fiel.	Disponible
Grilla	5	Sandi	Perro	Belga malinois	5	Macho	Marrón	Un perro muy protector, amigable y	Disponible
Textos	6	Pancho	Gato	Maine Coon	4	Macho	Marrón y dorado	Encantador Maine Coon con un pelaje atigrado que brilla con tonos dorados y marrones.	Disponible

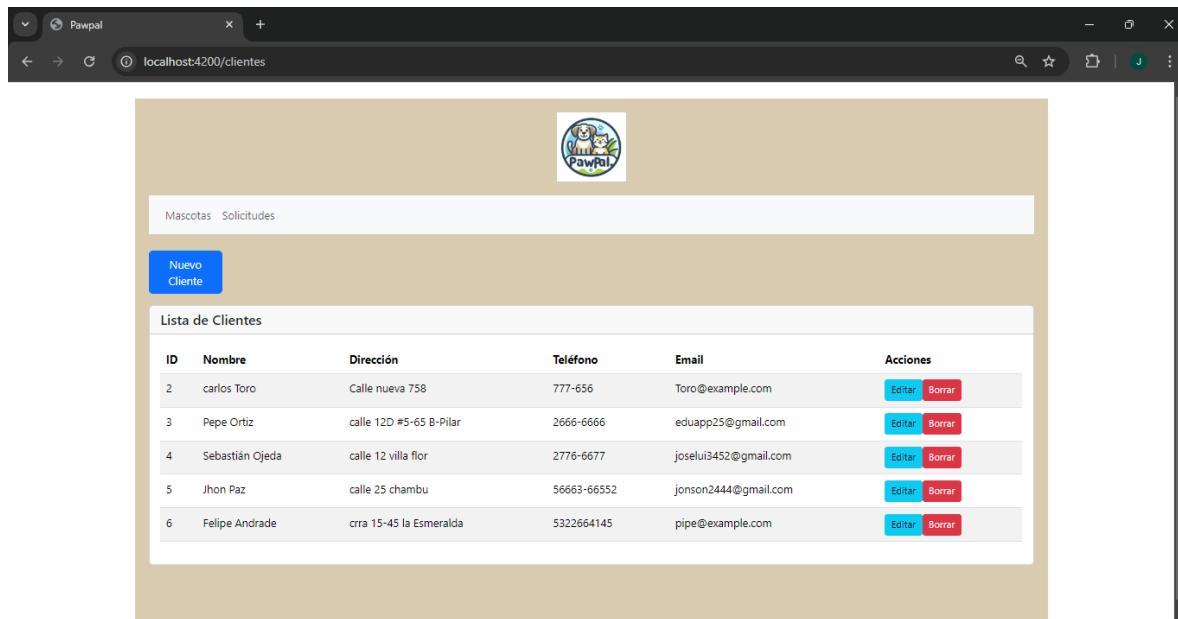
Para el botón **Borrar** vamos a eliminar la misma **sandi** con **ID:7**, al darle borrar se quita automáticamente de la lista y de la base de datos.



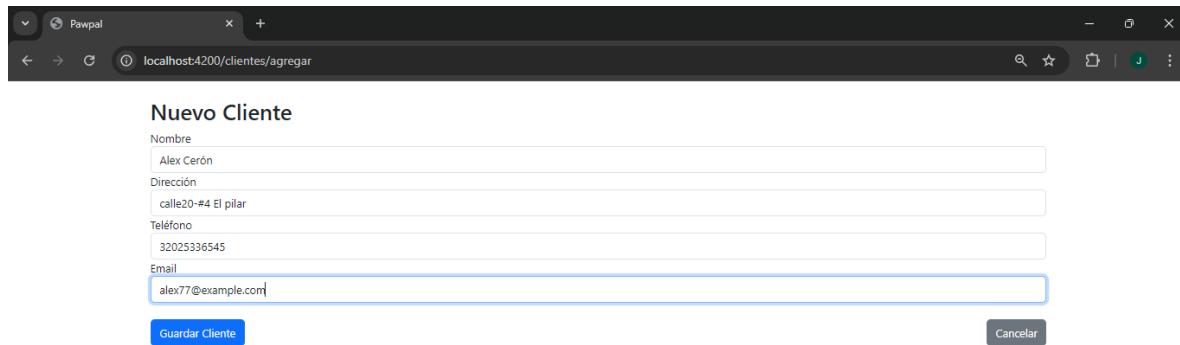
	A-Z nombre	A-Z especie	A-Z raza	123 edad	A-Z sexo	A-Z color	A-Z descripción	A-Z estado	Panorama
1	Max	Perro	Labrador	4	Macho	Marrón	Un perro muy enérgico y amigable.	Disponible	
4	orus	Perro	Rotwailer	5	Macho	Negro	Un perro muy fiel y amigable.	Disponible	
5	Lulu	Gato	Persa	3	Hembra	Blanca	Perezosa pero muy amigable.	Disponible	
6	Cloe	Perro	Labrador	5	Hembra	Blanca	Muy juguetona y fiel.	Disponible	
8	Pancho	Gato	Maine Coon	4	Macho	Marrón y dorado	Encantador Maine Coon con un pelaje	Disponible	

Cabe resaltar que a los formularios también se les agrego estilos de **Bootstrap**.

Ahora para la parte de clientes tenemos, el mismo menú y tabla con el uso de **Bootstrap**, lo que cambia del menú es que están las rutas para navegar a mascotas y solicitudes. Además, también tiene el botón de nuevo cliente, editar y borrar con uso de **Bootstrap** también, en la siguiente imagen se puede evidenciar lo dicho:



Prueba crear nuevo cliente:



Nuevo Cliente

Nombre
Alex Cerón

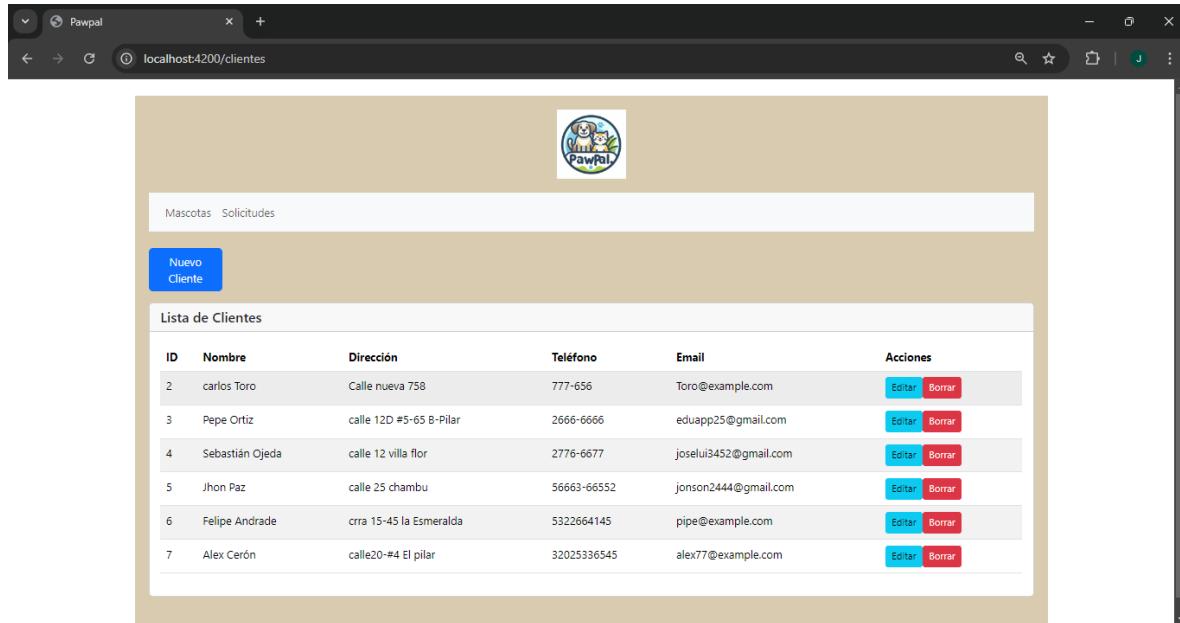
Dirección
calle20-#4 El pilar

Teléfono
32025336545

Email
alex77@example.com

Guardar Cliente Cancelar

El formulario también aplica el uso de **Bootstrap** y comprobamos la ruta que es **/agregar** para evidenciar el funcionamiento de rutas, una vez lleno los datos le damos **guardar cliente** en caso contrario **cancelar** y una vez nos redirecciona a /clientes verificamos en la página y en la base de datos que se haya creado a continuación las imágenes de evidencia.

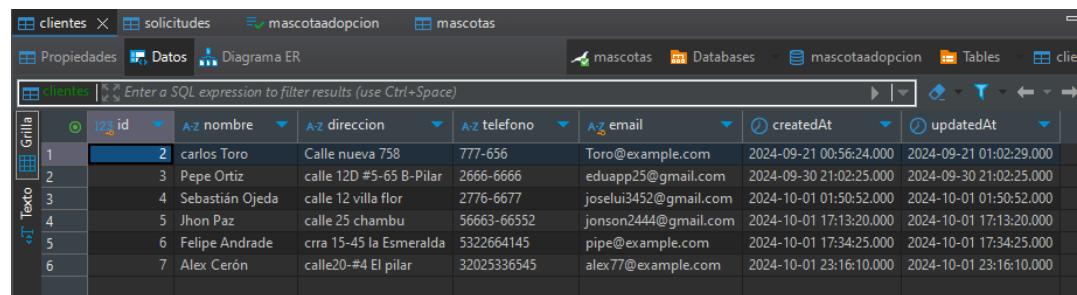


Mascotas Solicituds

Nuevo Cliente

Listado de Clientes

ID	Nombre	Dirección	Teléfono	Email	Acciones
2	carlos Toro	Calle nueva 758	777-656	Toro@example.com	<button>Editar</button> <button>Borrar</button>
3	Pepe Ortiz	calle 12D #5-65 B-Pilar	2666-6666	eduapp25@gmail.com	<button>Editar</button> <button>Borrar</button>
4	Sebastián Ojeda	calle 12 villa flor	2776-6677	joselu3452@gmail.com	<button>Editar</button> <button>Borrar</button>
5	Jhon Paz	calle 25 chambu	56663-66552	jonson2444@gmail.com	<button>Editar</button> <button>Borrar</button>
6	Felipe Andrade	crra 15-45 la Esmeralda	5322664145	pipe@example.com	<button>Editar</button> <button>Borrar</button>
7	Alex Cerón	calle20-#4 El pilar	32025336545	alex77@example.com	<button>Editar</button> <button>Borrar</button>



clientes x solicitudes x mascotaadopcion x mascotas

Propiedades Datos Diagrama ER

mascotas Databases Tables cliente

clientes Enter a SQL expression to filter results (use Ctrl+Space)

Grilla	1	2	3	4	5	6	7	id	nombre	dirección	teléfono	email	createdAt	updatedAt
Texto	1	2	3	4	5	6	7	2	carlos Toro	Calle nueva 758	777-656	Toro@example.com	2024-09-21 00:56:24.000	2024-09-21 01:02:29.000
		3	Pepe Ortiz	calle 12D #5-65 B-Pilar	2666-6666	eduapp25@gmail.com	2024-09-30 21:02:25.000	2024-09-30 21:02:25.000						
		4	Sebastián Ojeda	calle 12 villa flor	2776-6677	joselu3452@gmail.com	2024-10-01 01:50:52.000	2024-10-01 01:50:52.000						
		5	Jhon Paz	calle 25 chambu	56663-66552	jonson2444@gmail.com	2024-10-01 17:13:20.000	2024-10-01 17:13:20.000						
		6	Felipe Andrade	crra 15-45 la Esmeralda	5322664145	pipe@example.com	2024-10-01 17:34:25.000	2024-10-01 17:34:25.000						
		7	Alex Cerón	calle20-#4 El pilar	32025336545	alex77@example.com	2024-10-01 23:16:10.000	2024-10-01 23:16:10.000						

En las imágenes podemos evidenciar el registro del nuevo cliente el cual sería el del **ID:7** tanto en la pagina como en la BD.

Para comprobar el funcionamiento del botón editar en clientes al hacer click nos manda a la ruta **/clientes/editar/7** la ruta corresponde al formulario donde de va a editar los datos que automáticamente los trae para editarlos en el formulario a continuación vamos a editar el correo y el apellido del cliente.

Imagen sin cambios para evidenciar que trae los datos:

A screenshot of a web browser window titled "Pawpal". The address bar shows "localhost:4200/clientes/editar/7". The main content is a form titled "Editar Cliente" with the following fields:

- Nombre: Alex Ceron
- Dirección: calle20-#4 El pilar
- Teléfono: 32025336545
- Email: alex77@example.com

At the bottom are two buttons: "Guardar Cliente" (blue) and "Cancelar" (gray).

Imagen con cambios para evidenciar que se editan los datos tanto en la página como en la base de datos:

A screenshot of a web browser window titled "Pawpal". The address bar shows "localhost:4200/clientes/editar/7". The main content is a form titled "Editar Cliente" with the following fields:

- Nombre: Alex Córdova
- Dirección: calle20-#4 El pilar
- Teléfono: 32025336545
- Email: alexcordova@example.com

The "Email" field is highlighted with a blue border. At the bottom are two buttons: "Guardar Cliente" (blue) and "Cancelar" (gray).

Al darle guardar cliente nos redirecciona a la ruta principal de clientes **/clientes** y verificamos en la página y base de datos:

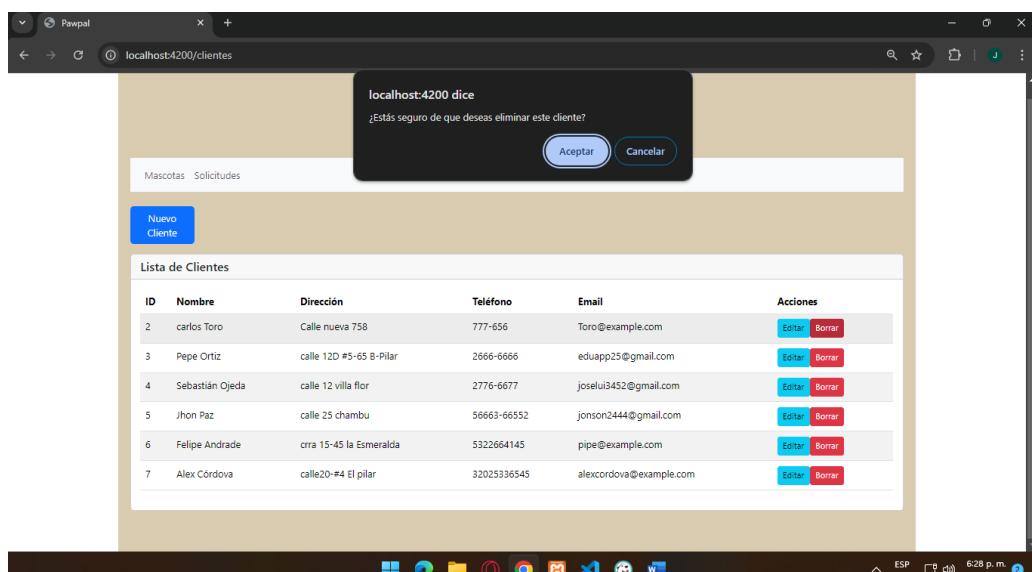
A screenshot of a web browser window titled "Pawpal". The address bar shows "localhost:4200/clientes". The main content is a table titled "Lista de Clientes" with the following data:

ID	Nombre	Dirección	Teléfono	Email	Acciones
2	carlos Toro	Calle nueva 758	777-656	Toro@example.com	Editar Borrar
3	Pepe Ortiz	calle 12D #5-65 B-Pilar	2666-6666	eduapp25@gmail.com	Editar Borrar
4	Sebastián Ojeda	calle 12 villa flor	2776-6677	joseui3452@gmail.com	Editar Borrar
5	Jhon Paz	calle 25 chambu	56663-66552	jonson2444@gmail.com	Editar Borrar
6	Felipe Andrade	crra 15-45 la Esmeralda	5322664145	pipe@example.com	Editar Borrar
7	Alex Córdova	calle20-#4 El pilar	32025336545	alexcordova@example.com	Editar Borrar

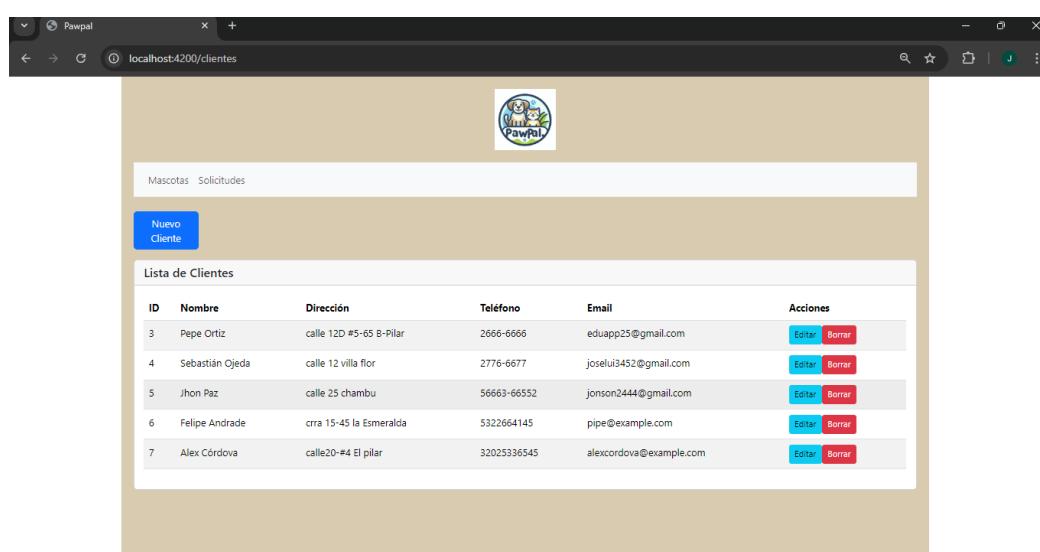
The screenshot shows the MySQL Workbench interface with the 'clientes' table selected. The table has columns: id, nombre, dirección, teléfono, email, createdAt, and updatedAt. The data includes entries for Carlos Toro, Pepe Ortiz, Sebastián Ojeda, Jhon Paz, Felipe Andrade, and Alex Córdova.

	id	nombre	dirección	teléfono	email	createdAt	updatedAt
1	2	carlos Toro	Calle nueva 758	777-656	Toro@example.com	2024-09-21 00:56:24.000	2024-09-21 01:02:29.000
2	3	Pepe Ortiz	calle 12D #5-65 B-Pilar	2666-6666	eduapp25@gmail.com	2024-09-30 21:02:25.000	2024-09-30 21:02:25.000
3	4	Sebastián Ojeda	calle 12 villa flor	2776-6677	joselui3452@gmail.com	2024-10-01 01:50:52.000	2024-10-01 01:50:52.000
4	5	Jhon Paz	calle 25 chambu	56663-66552	jonson2444@gmail.com	2024-10-01 17:13:20.000	2024-10-01 17:13:20.000
5	6	Felipe Andrade	crra 15-45 la Esmeralda	5322664145	pipe@example.com	2024-10-01 17:34:25.000	2024-10-01 17:34:25.000
6	7	Alex Córdova	calle20-#4 El pilar	32025336545	alexcordova@example.cc	2024-10-01 23:16:10.000	2024-10-01 23:25:08.000

Así entonces con lo anterior verificamos el funcionamiento tanto de las rutas como del formulario y botón aplicado el estilo, como punto final de esta interfaz el botón eliminar la prueba se hará con el cliente **carlos Toro con ID:2** a continuación las imágenes de evidencia:



Al darle aceptar se elimina de la pagina y de la BD:



The screenshot shows the MySQL Workbench interface again, this time with the "clientes" table having only 6 rows. The data is identical to the previous screenshot, except for the absence of the first row (ID 2).

	id	nombre	dirección	teléfono	email	createdAt	updatedAt
1	3	Pepe Ortiz	calle 12D #5-65 B-Pilar	2666-6666	eduapp25@gmail.com	2024-09-30 21:02:25.000	2024-09-30 21:02:25.000
2	4	Sebastián Ojeda	calle 12 villa flor	2776-6677	joselui3452@gmail.com	2024-10-01 01:50:52.000	2024-10-01 01:50:52.000
3	5	Jhon Paz	calle 25 chambu	56663-66552	jonson2444@gmail.com	2024-10-01 17:13:20.000	2024-10-01 17:13:20.000
4	6	Felipe Andrade	crra 15-45 la Esmeralda	5322664145	pipe@example.com	2024-10-01 17:34:25.000	2024-10-01 17:34:25.000
5	7	Alex Córdova	calle20-#4 El pilar	32025336545	alexcordova@example.cc	2024-10-01 23:16:10.000	2024-10-01 23:25:08.000

Por último, para la parte de solicitud tenemos, el mismo menú y tabla con el uso de **Bootstrap**, lo que cambia del menú es que están las rutas para navegar a mascotas y clientes. Además, también tiene el botón de **nueva solicitud**, **editar** y **borrar** con uso de **Bootstrap** también, en la siguiente imagen se puede evidenciar lo dicho:

ID	Fecha de Solicitud	Estado	Cliente	Mascota	Acciones
5	09/08/2024	Rechazada	Sin cliente	Max	<button>Editar</button> <button>Borrar</button>
44	09/10/2024	Pendiente	Jhon Paz	Lulu	<button>Editar</button> <button>Borrar</button>
45	30/10/2024	Pendiente	Sin cliente	Cloe	<button>Editar</button> <button>Borrar</button>
48	31/10/2024	Pendiente	Felipe Andrade	Sin mascota	<button>Editar</button> <button>Borrar</button>

Podemos mirar que tiene la ruta correspondiente **/solicitudes** para términos de aclaración en la tabla muestra que la solicitud **ID:5** y la **ID:45** dice **Sin cliente**, además la solicitud **ID:48** dice **Sin mascota**, eso se debe a que como en las pruebas anteriores hemos eliminado al cliente y a la mascota en el código le especificamos que cuando no encuentre **ID** registrado asigne por defecto esos datos por default.

Para comprobar el botón Nueva Solicitud hacemos click y nos manda a la ruta **/solicitudes/agregar** donde nos muestra el formulario para crearla, como prueba vamos a realizar con la nueva mascota y el nuevo cliente agregados como prueba en los anteriores casos, la mascota de nombre **Pancho** con **ID:8** y el cliente **Alex Córdova** con **ID:7** en la imagen siguiente se evidencia el formulario en la ruta especificada y los datos para hacer la prueba:

Una vez lleno los datos correspondientes para una solicitud le damos en **guardar solicitud**, y comprobamos el registro en la página **/solicitudes** y en la base de datos:

	id	fecha_solicitud	estado	createdAt	updatedAt	clienteId	mascotaId
1	5	2024-08-09 05:00:00.000	Rechazada	2024-09-30 21:31:40.000	2024-10-01 15:13:39.000	[NULL]	1
2	44	2024-10-10 00:00:00.000	Pendiente	2024-10-01 17:17:21.000	2024-10-01 17:17:21.000	5	5
3	45	2024-10-31 00:00:00.000	Pendiente	2024-10-01 17:17:41.000	2024-10-01 17:17:41.000	[NULL]	6
4	48	2024-11-01 00:00:00.000	Pendiente	2024-10-01 17:35:21.000	2024-10-01 17:35:21.000	6	[NULL]
5	51	2024-10-01 00:00:00.000	Pendiente	2024-10-01 23:48:34.000	2024-10-01 23:48:34.000	7	8

Podemos evidenciar en las imágenes anteriores que el cliente **Alex Córdova** realizó la petición de adopción para adoptar la mascota **Pancho** con registro de la fecha de la solicitud realizada y **estado pendiente** el registro se hizo tanto en la página como en la BD.

Para la prueba de el botón editar vamos a reasignar clientes y mascotas que ya existan para que no salga los datos **Sin cliente** y **Sin mascota** vamos a asignar en la solicitud **ID:5** al cliente **Pepe Ortiz** con **ID:3**, en la solicitud **ID:45** vamos a asignar al cliente **Jhon Paz** con **ID:5**, y en la solicitud **ID:48** vamos a asignar a la mascota **orus** con **ID:4**, además dichas solicitudes les vamos a cambiar su estado de aprobación a **aprobadas** a continuación al hacer click en el botón editar nos manda a la ruta con el respectivo id de la solicitud para editar en las imágenes a continuación se podrá evidenciar el formulario de edición que trae automáticamente los datos en los campos para poder modificarlos:

/solicitudes/editar/5 → sin asignar cliente ni estado

Editar Solicitud

Estado
Rechazada

ID del Cliente

ID de la Mascota
1

Fecha de Solicitud
09/08/2024

solicitudes/editar/5 → asignado cliente y su estado

Editar Solicitud

Estado
Aprobada

ID del Cliente
3

ID de la Mascota
1

Fecha de Solicitud
09/08/2024

/solicitudes/editar/45 → sin asignar cliente ni estado

Editar Solicitud

Estado
Pendiente

ID del Cliente

ID de la Mascota
6

Fecha de Solicitud
30/10/2024

/solicitudes/editar/45 → asignado cliente y cambiado estado

Editar Solicitud

Estado
Aprobada

ID del Cliente
5

ID de la Mascota
6

Fecha de Solicitud
30/10/2024

/solicitudes/editar/48 → sin asignar cliente ni estado

Editar Solicitud

Estado
Pending

ID del Cliente
6

ID de la Mascota

Fecha de Solicitud
31/10/2024

Guardar Solicitud Cancelar

/solicitudes/editar/48 → asignado cliente y su estado

Editar Solicitud

Estado
Aprobada

ID del Cliente
6

ID de la Mascota
4

Fecha de Solicitud
31/10/2024

Guardar Solicitud Cancelar

Y para comprobar que los cambios anteriores se hayan realizado verificamos la página y la base de datos:

Mascotas Clientes

Nueva Solicitud

Lista de Solicitudes

ID	Fecha de Solicitud	Estado	Cliente	Mascota	Acciones
5	08/08/2024	Aprobada	Pepe Ortiz	Max	<button>Editar</button> <button>Borrar</button>
44	09/10/2024	Pendiente	Jhon Paz	Lulu	<button>Editar</button> <button>Borrar</button>
45	29/10/2024	Aprobada	Jhon Paz	Cloe	<button>Editar</button> <button>Borrar</button>
48	30/10/2024	Aprobada	Felipe Andrade	orus	<button>Editar</button> <button>Borrar</button>
51	30/09/2024	Pendiente	Alex Córdova	Pancho	<button>Editar</button> <button>Borrar</button>

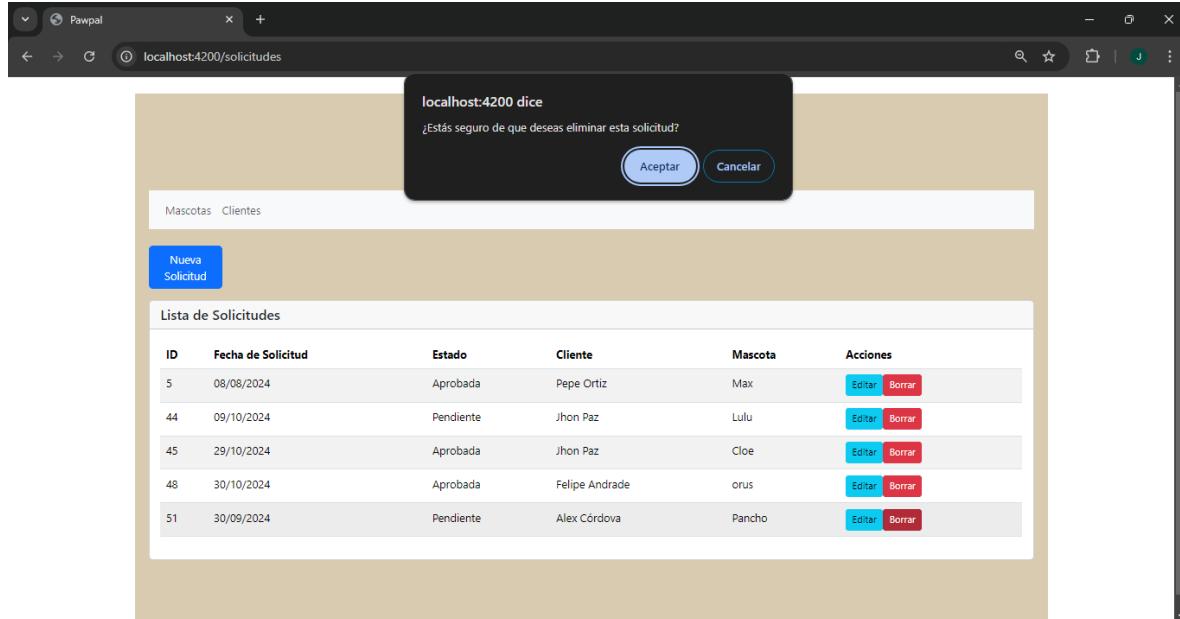
Grilla Texto

solicitudes | Enter a SQL expression to filter results (use Ctrl+Space)

id	fecha_solicitud	estado	createdAt	updatedAt	clientelid	mascotald
1	5	Aprobada	2024-09-30 21:31:40.000	2024-10-02 00:06:39.000	3	1
2	44	Pendiente	2024-10-01 17:17:21.000	2024-10-01 17:17:21.000	5	5
3	45	Aprobada	2024-10-01 17:17:41.000	2024-10-02 00:09:58.000	5	6
4	48	Aprobada	2024-10-01 17:35:21.000	2024-10-02 00:16:07.000	6	4
5	51	Pendiente	2024-10-01 23:48:34.000	2024-10-01 23:48:34.000	7	8

Como podemos ver los cambios se han realizado efectivamente además comprobamos que las rutas están funcionando perfectamente en solicitudes.

Por último, para comprobar el botón de **borrar** vamos a eliminar la solicitud con **ID:51** las imágenes a continuación muestran la ejecución y cambios tanto en la BD como en la página:



Al darle aceptar se borra y se hacen los cambios en ambas partes:

A screenshot of a web browser window titled "localhost:4200/solicitudes". The page displays the same table as before, but the fifth row (ID 51) is missing, showing only four rows of data. Below the browser window is a screenshot of a database management tool showing the "solicitudes" table in a MySQL-like interface. The table has four visible rows, corresponding to the data shown in the browser.

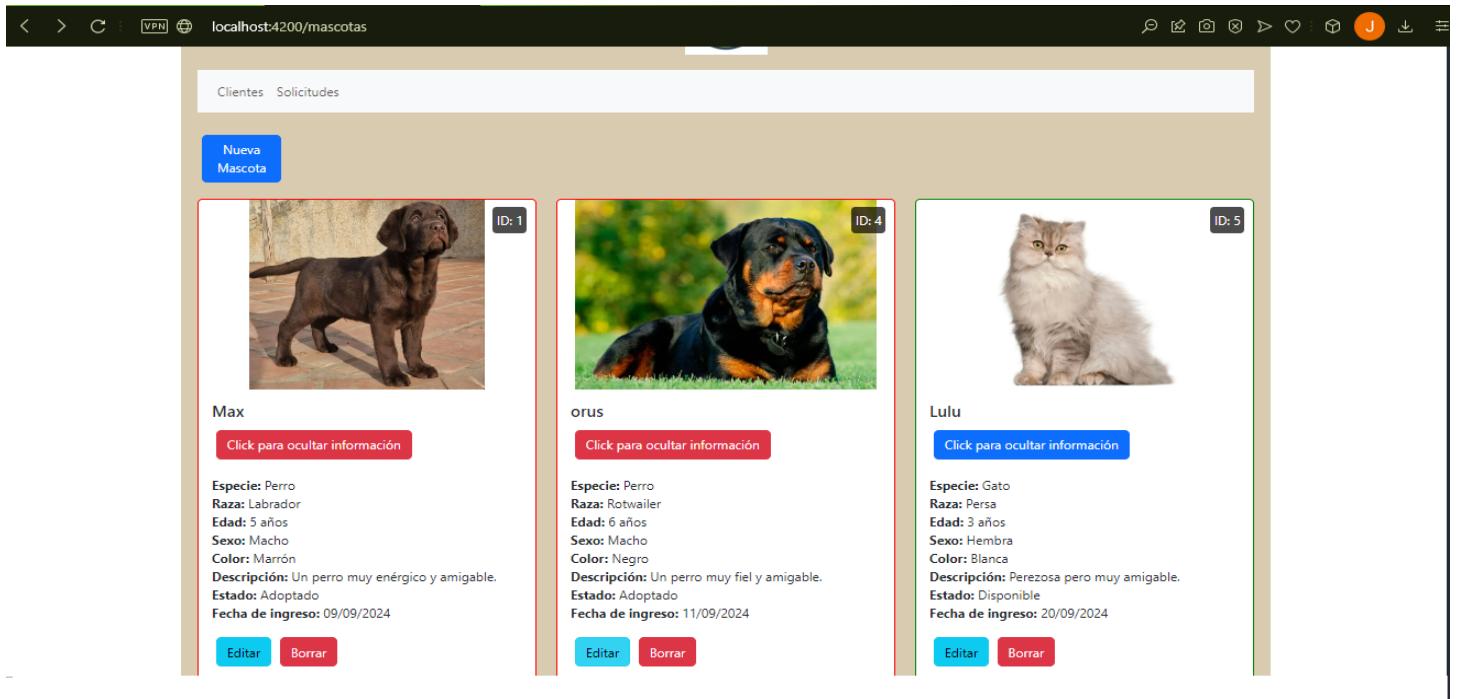
ID	Fecha de Solicitud	Estado	Cliente	Mascota	Acciones
5	08/08/2024	Aprobada	Pepe Ortiz	Max	<button>Editar</button> <button>Borrar</button>
44	09/10/2024	Pendiente	Jhon Paz	Lulu	<button>Editar</button> <button>Borrar</button>
45	29/10/2024	Aprobada	Jhon Paz	Cloe	<button>Editar</button> <button>Borrar</button>
48	30/10/2024	Aprobada	Felipe Andrade	orus	<button>Editar</button> <button>Borrar</button>

En conclusión, este proyecto de desarrollo frontend ha permitido no solo aplicar los conocimientos adquiridos en las clases profundizando en la experiencia del usuario y la accesibilidad, a través del uso de tecnologías como HTML, CSS y JavaScript, se logró crear una aplicación web intuitiva y atractiva que cumple con los requisitos establecidos al inicio del proyecto.

Actualización: como se amplió la fecha de entrega del taller y mi taller ya estaba enviada y para no dañar mi informe a continuación haremos un resumen de las modificaciones que hice.

Solo hice un pequeño cambio que fue la interfaz de mascotas donde la tarjeta se pondrá roja si la mascota se le pone el estado en adoptado y verde si está disponible.

A continuación, la imagen para evidenciar mejor la interfaz:



cómo podemos observar al estar adoptado se pone en rojo el botón y el contorno de la tarjeta, y cuando no está adoptado se pone su contorno de la tarjeta verde.