

Manejo de errores desde el Frontend con SweetAlert

Contenido

Manejo de errores desde el Frontend	1
Manejo de error en el Frontend en obtener por id	2
Manejo de error en el Frontend en create,update y delete	3

Manejo de error en el Frontend en obtener por id

Tened en cuenta que esto se hace desde Angular, supongo que desde React también será parecido.

Tenemos que irnos a cliente.service, que es donde estamos interactuando con el backend.

Primero importamos:

```
import { catchError } from 'rxjs/operators';
import swal from 'sweetalert2';

import { Router } from '@angular/router';
```

Luego en el constructor añadimos el router para redirigir si hay error al listado de clientes:

```
constructor(private http: HttpClient, private router: Router) { }
```

Y por último modificamos el getCiente(id)

```
1 getCiente(id): Observable<Cliente>{
2   return this.http.get<Cliente>(`${this.urlEndPoint}/${id}`).pipe(
3     catchError(e=>{
4       this.router.navigate(['/clientes'])
5       console.error(e.error.mensaje);
6       swal('Error al editar',e.error.mensaje,'error');
7       return throwError(e);
8     })
9   );
10 }
```

Si levantamos el servicio y vamos a editar, y cambiamos el id:

The screenshot shows the Angular application interface with a table of clients. A modal dialog is displayed over the table with the title "Error al editar" and the message "El cliente con el ID: 88888 no existe en la base de datos." The modal has an "OK" button. On the right side of the browser window, the Chrome DevTools Console is open, showing several error messages. The most relevant ones are: "HTTP404: NO ENCONTRADO: el servidor no encontró nada que coincidiera con el URI (identificador uniforme de recursos) solicitado. (XHR)GET - http://localhost:8080/api/clientes/88888", "El cliente con el ID: 88888 no existe en la base de datos.", and "ERROR [object Object]". The console also shows messages about "Live Reloading enabled" and "Token inesperado".

Manejo de error en el Frontend en create,update y delete

En el método create no vamos a reenviar a otro formulario, así que obviamos la parte del reenvío:

```
create(cliente: Cliente) : Observable <Cliente>{  
  return this.http.post<Cliente>(this.urlEndPoint,cliente,{headers: this.httpHeaders}).pipe(  
    catchError(e=>{  
      console.error(e.error.mensaje);  
      swal('Error al crear el cliente',e.error.mensaje,'error');  
      return throwError(e);  
    })  
  )  
  ;  
}
```

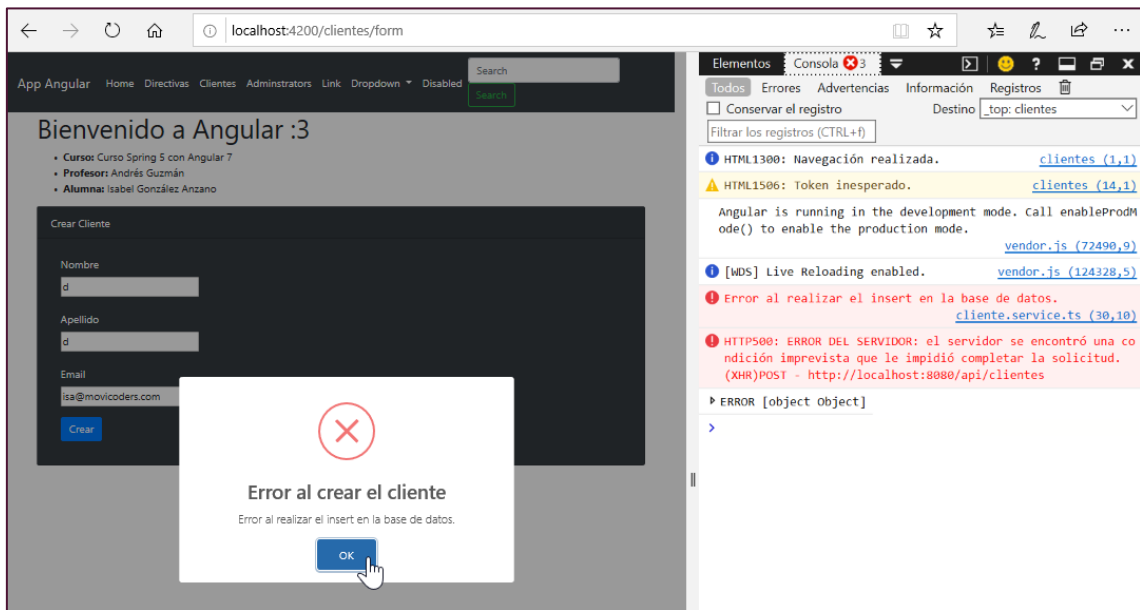
Y lo mismo para el update y delete:

```
update(cliente: Cliente):Observable <Cliente>{  
  return this.http.put<Cliente>(`${this.urlEndPoint}/${cliente.id}`,cliente,{headers: this.httpHeaders}).pipe(  
    catchError(e=>{  
      console.error(e.error.mensaje);  
      swal('Error al editar el cliente',e.error.mensaje,'error');  
      return throwError(e);  
    })  
  )  
  ;  
}  
  
delete(id: number) :Observable <Cliente> {  
  return this.http.delete<Cliente>(`${this.urlEndPoint}/${id}`,headers: this.httpHeaders).pipe(  
    catchError(e=>{  
      console.error(e.error.mensaje);  
      swal('Error al eliminar el cliente',e.error.mensaje,'error');  
      return throwError(e);  
    })  
  )  
  ;  
}
```

Error al crear un cliente con un email ya registrado:

Isabel González Anzano

MANEJO DE ERRORES DESDE EL FRONTEND CON SWEETALERT



```
0-exec-4] org.hibernate.SQL : insert into clientes (apellido, create_at, email, nombre) values (?, ?, ?, ?)
0-exec-4] o.h.engine.jdbc.spi.SqlExceptionHelper : SQL Error: 1062, SQLState: 23000
0-exec-4] o.h.engine.jdbc.spi.SqlExceptionHelper : Duplicate entry 'isa@movicoders.com' for key 'UK_1c96wv36rk2hwui7qhjks3mvg'
```

Si intentamos crear un cliente sin datos:

