



TECNOLÓGICO
NACIONAL DE MÉXICO



TECNOLOGICO NACIONAL DE MEXICO
INSTITUTO TECNOLÓGICO DE CUAUTLA

- BELAUNZARÁN FIORENTINI JOSEL
- ESPÍNDOLA TORRES ÁLVARO ISAIAS
- GUZMÁN GÓMEZ ÁNGEL HIRAMG
- JUÁREZ MENDOZA EFRAÍN
- PÉREZ SÁNCHEZ ANGIE HARUMI
- VIVAR TIEMPOS RIGOBERTO

PROGRAMACIÓN LÓGICA Y FUNCIONAL

ING. GISELA VEGA TORRES

EJERCICIO HORARIO.

7mo SEMESTRE

GRUPO 1

Codificación.

Aquí tenemos el BackEnd, el cual es el encargado de realizar las consultas en la Base de datos y retornar una respuesta.

```
index.controller.js | src/index.js | routes/index.js
17 //datos del docente
18 const datosDocente = async (req,res)=>{
19   let idDocente = req.params.id;
20   const respDatosDocente = await pool.query('SELECT nombredocente,apapatdocente,apematdocente FROM docente WHERE iddocente = $1',[idDocente]);
21   res.status(200).json({
22     ok:true,
23     respDatosDocente
24   });
25 }
26
27 //Horario del docente
28 const horarioDocente = async (req,res)=> {
29   let idDocente = req.params.id;
30   const respHorarioDocente = await pool.query('SELECT idedificiohorario,numeroaulahorario,horainicio,horafin,clavemateriahorario FROM horariogrupo WHERE iddocentehorario = $1',[idDocente]);
31   res.status(200).json({
32     ok:true,
33     respHorarioDocente
34   });
35 }
36
37 //datos del alumno
38 const datosAlumno = async (req,res)=>{
39   let numControl = req.params.id;
40   const respDatosAlumno = await pool.query('SELECT nombrealumno,apapatlumno,apematlumno FROM alumno WHERE nocontrol = $1',[numControl]);
41   res.status(200).json({
42     ok:true,
43     respDatosAlumno
44   });
45 }
46
47 //Horario del alumno
48 const horarioAlumno = async (req,res)=> {
49   let numControl = req.params.id;
50   const respHorarioAlumno = await pool.query('SELECT idedificiohorario,numeroaulahorario,horainicio,horafin,clavemateriahorario FROM horariogrupo WHERE nocontrolhorario = $1',[numControl]);
51   res.status(200).json({
52     ok:true,
53     respHorarioAlumno
54   });
55 }
```

En cuestión del FrontEnd, contamos con dos funciones principales, las cuales son las encargadas de solicitar las peticiones al BackEnd y posteriormente hacer las respectivas validaciones.

```
api.service.js | principal-component.component.ts | principal-component.component.html
34
35 buscarAlumno = () =>{
36   console.clear();
37   this._apiService.consultarAlumno(this.numControl).subscribe((resp:any) =>{
38
39     //Validamos si es que hubo respuesta de la BD
40     if (resp.respDatosAlumno.rowCount == 0) {
41       console.error("NoAlumno no registrado","font-size:20px");
42     }else{
43       console.log("NoAlumno Encontrado","color:green;font-size:20px");
44       console.log(resp.respDatosAlumno.rows);
45       this._apiService.localizarAlumno(this.numControl).subscribe((resp:any) =>{
46         console.log(resp.respHorarioAlumno.rows);
47         //Guardamos las filas que retorna la bd
48         let valores = resp.respHorarioAlumno.rows;
49
50         //Recorremos las rows para con las horas determinar la ubicación
51         for (const valor in valores) {
52           let horainicio = valores[valor]["horainicio"];
53           let horafin = valores[valor]["horafin"];
54
55           //Validamos que la hora de búsqueda esté en el rango
56           if ((this.hora >= horainicio) && (this.hora <= horafin)) {
57             console.log("NoAl alumno se encuentra en el edificio ${valores[valor]["idedificiohorario"]} aula ${valores[valor]["numeroaulahorario"]}", "color:yellow;font-size:20px");
58           }
59         }
60       });
61     }
62   });
63 }
64
65 buscarProfesor = () =>{
66   console.clear();
67   this._apiService.consultarDocente(this.idDocente).subscribe((resp:any) =>{
68     //Validamos si es que hubo respuesta de la BD
69     if (resp.respDatosDocente.rowCount == 0) {
70       console.error("NoProfesor no registrado","font-size:20px");
71     }else{
72       console.log("NoProfesor Encontrado","color:green;font-size:20px");
73       console.log(resp.respDatosDocente.rows);
74       this._apiService.localizarDocente(this.idDocente).subscribe((resp:any) =>{
75         console.log(resp.respHorarioDocente.rows);
76         //Guardamos las filas que retorna la bd
77         let valores = resp.respHorarioDocente.rows;
78
79         //Recorremos las rows para con las horas determinar la ubicación
80         for (const valor in valores) {
81           let horainicio = valores[valor]["horainicio"];
82           let horafin = valores[valor]["horafin"];
83
84           //Validamos que la hora de búsqueda esté en el rango
85           if ((this.hora >= horainicio) && (this.hora <= horafin)) {
86             console.log("NoAl docente se encuentra en el edificio ${valores[valor]["idedificiohorario"]} aula ${valores[valor]["numeroaulahorario"]}", "color:yellow;font-size:20px");
87           }
88         }
89       });
90     }
91   });
92 }
93
94 }
```

Esta será el API, la cual es la encargada de conectar ambos servicios, tanto el BackEnd como el FronEnd.

```
api.service.ts U • principal-component.component.ts U principal-component.component.html U
src > app > api.service.ts > ApiService
1 import { Injectable } from '@angular/core';
2 import { HttpClient } from '@angular/common/http';
3
4 @Injectable({
5   providedIn: 'root'
6 })
7 export class ApiService {
8   public url;
9
10  constructor(private _http:HttpClient) {
11    this.url = "http://localhost:3000";
12  }
13
14  consultarAlumno = (id:any) =>{
15    let url = `${this.url}/datos/alumno/${id}`;
16    return this._http.get(url);
17  }
18
19  consultarDocente = (id:any) =>{
20    let url = `${this.url}/datos/docente/${id}`;
21    return this._http.get(url);
22  }
23
24  localizarAlumno = (id:any) =>{
25    let url = `${this.url}/buscar/alumno/${id}`;
26    return this._http.get(url);
27  }
28
29  localizarDocente = (id:any,) =>{
30    let url = `${this.url}/buscar/docente/${id}`;
31    return this._http.get(url);
32  }
33 }
```

Registros.

Aquí encontramos las tablas con las que cuenta la Base de datos.

SQL Shell (psql)

```
itc=# \d
```

Esquema	Nombre	Tipo	Dueño
public	alumno	tabla	postgres
public	aula	tabla	postgres
public	carrera	tabla	postgres
public	docente	tabla	postgres
public	edificio	tabla	postgres
public	estatusmateria	tabla	postgres
public	horarioalumno	tabla	postgres
public	horariogrupo	tabla	postgres
public	materia	tabla	postgres
public	periodo	tabla	postgres

(10 filas)

```
itc=#
```

La tabla horariogrupo, contendrá información sobre los horarios, la cual está relacionada mediante llaves foráneas con las tablas pertinentes.

SQL Shell (psql)

```
itc=# \d horariogrupo;
```

Columna	Tipo	Ordenamiento	Nulable	Por omisión
clavehorariogrupo	character varying(20)		not null	
nocontrolhorario	integer		not null	
grupo	numeric(1,0)		not null	
iddocentehorario	integer		not null	
clavemateriahorario	character varying(8)		not null	
dia	character varying(1)		not null	
horainicio	time without time zone		not null	
horafin	time without time zone		not null	
idedificiohorario	character varying(1)		not null	
numeroaulahorario	numeric(2,0)		not null	
idperiodohorario	character varying(12)		not null	

Restricciones de llave foránea:

- "horariogrupo_clavemateriahorario_fkey" FOREIGN KEY (clavemateriahorario) REFERENCES materia(clavemateria)
- "horariogrupo_iddocentehorario_fkey" FOREIGN KEY (iddocentehorario) REFERENCES docente(iddocente)
- "horariogrupo_idedificiohorario_fkey" FOREIGN KEY (idedificiohorario) REFERENCES edificio(idedificio)
- "horariogrupo_idperiodohorario_fkey" FOREIGN KEY (idperiodohorario) REFERENCES periodo(idperiodo)
- "horariogrupo_nocontrolhorario_fkey" FOREIGN KEY (nocontrolhorario) REFERENCES alumno(nocontrol)

```
itc=#
```

Registros de la tabla horariogrupo.

SQL Shell (psql)

```
itc=# SELECT * FROM horariogrupo;
```

clavehorariogrupo	nocontrolhorario	grupo	iddocentehorario	clavemateriahorario	dia	horainicio	horafin	idedificiohorario	numeroaulahorario	idperiodohorario
7-ISC-1	19680096	1	1	1R8	1	07:00:00	08:00:00	A		1 AGO-DIC/2022
7-ISC-1	19680096	1	4	6R7	1	08:00:00	09:00:00	A		2 AGO-DIC/2022
7-ISC-1	19680096	1	1	3R7	1	09:00:00	10:00:00	B		3 AGO-DIC/2022
7-ISC-2	19680096	2	19	25W	1	10:00:00	11:00:00	C		4 AGO-DIC/2022
7-ISC-2	19680096	2	6	4R8	1	11:00:00	12:00:00	E		4 AGO-DIC/2022
7-ISC-2	19680096	2	8	1U6	5	12:00:00	13:00:00	F		3 AGO-DIC/2022

(6 filas)

```
itc=#
```

Función de las salidas.

Prueba exitosa de la búsqueda Alumno.

Horarios

Buscar Alumno Buscar Profesor

Número de control: 19680096 Hora de búsqueda: 09:30:18 a.m.

Buscar Alumno

Alumno Encontrado

El alumno se encuentra en el edificio B aula 3

Prueba fallida de la búsqueda Alumno.

Horarios

Buscar Alumno Buscar Profesor

Número de control: 10680147 Hora de búsqueda: 11:37:49 a.m.

Buscar Alumno

Alumno no registrado

Prueba exitosa de la búsqueda Docente.

Horarios

Buscar Alumno Buscar Profesor

Número de control: 1 Hora de búsqueda: 07:52:18 a.m.

Buscar Profesor

Profesor Encontrado

El docente se encuentra en el edificio A aula 1

```
SQL Shell (psql)
port: [5432]
Username: [postgres]
Contraseña para usuario postgres:
psql (13.15)
ADVERTENCIA: El código de página de la consola (650) difiere del código
de página de Windows (1252).
Los caracteres de 0 a 255 pueden funcionar incorrectamente.
Vea la página de referencia de psql <https://www.postgresql.org/docs/13/psql-features.html>
para obtener más detalles.
Digite «\help» para obtener ayuda.

11c=# SELECT * FROM horarios_grupo;
 clavehorariogrupo | nocontrolhorario | grupo | iddocentehorario | clavemateriahorario | dia | horainicio | horafin | idedificiohorario | numeroaulahorario | idperiodohorario
-----
 7-TIC-1           | 19680096         | 1      | 1                | 4                    | 1   | 07:00:00  | 08:00:00 | A                 | 1                  | AGO-DIC/2022
 7-TIC-1           | 19680096         | 1      | 4                | 687                  | 1   | 08:00:00  | 09:00:00 | A                 | 2                  | AGO-DIC/2022
 7-TIC-1           | 19680096         | 1      | 1                | 182                  | 1   | 09:00:00  | 10:00:00 | B                 | 3                  | AGO-DIC/2022
 7-TIC-2           | 19680096         | 2      | 10               | 254                  | 1   | 10:00:00  | 11:00:00 | C                 | 4                  | AGO-DIC/2022
 7-TIC-2           | 19680096         | 2      | 6                | 488                  | 1   | 11:00:00  | 12:00:00 | E                 | 6                  | AGO-DIC/2022
 7-TIC-2           | 19680096         | 2      | 8                | 516                  | 5   | 12:00:00  | 13:00:00 | F                 | 3                  | AGO-DIC/2022
(6 filas)
```

Prueba fallida de la búsqueda Docente.

Horarios

Buscar Alumno Buscar Profesor

Número de control: 90 Hora de búsqueda: 11:37:49 a.m.

Buscar Profesor

Profesor no registrado