

Ingeniería en Software 3

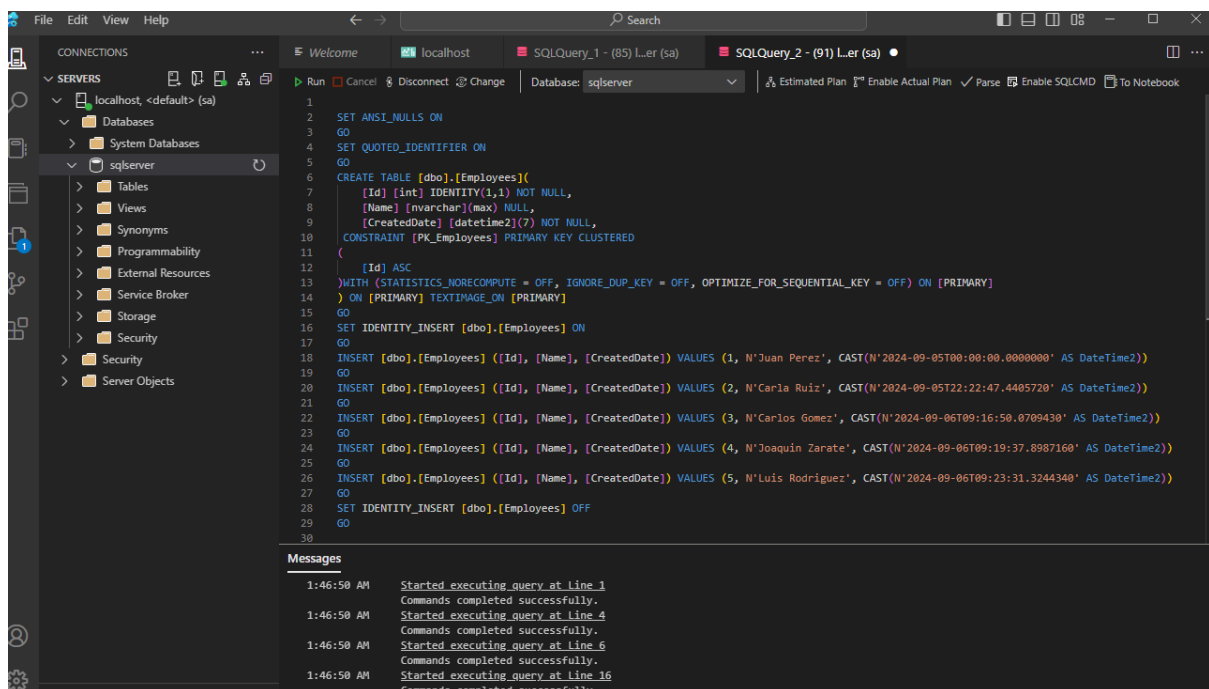
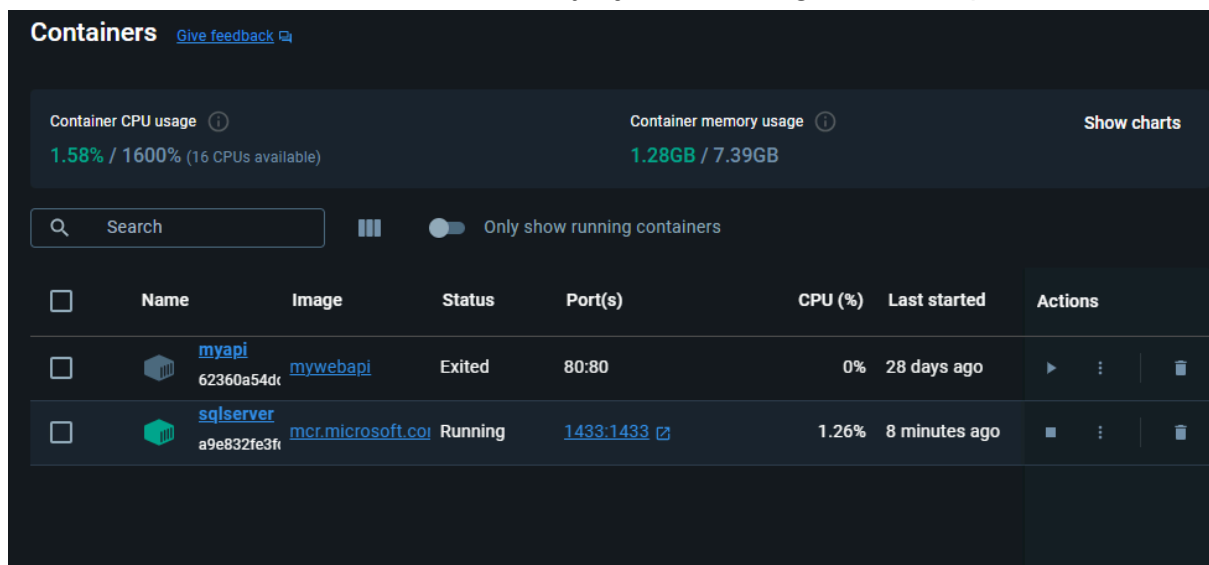
TP6 - Pruebas Unitarias

Nicolas Bergami

4- Desarrollo:

4.1 Creación de una BD SQL Server para nuestra App

- A. Crear una BD Azure SQL Database (Ver Instructivo 5.1) o montar una imagen Docker de SQL Server como se solicitó en el punto 12 del [TP02].
- B. En caso de optar por la opción de montar la imagen de docker, una vez levantada el contenedor, conectarse y ejecutar el siguiente script:



```

2> GO
Msg 263, Level 16, State 1, Server 17e6cc708f21, Line 1
Must specify table to select from.
1> SELECT * FROM [dbo].[Employees];
2> GO
Id          Name
-----
CreatedDate
-----
1 Juan Perez
2024-09-05 00:00:00.0000000
2 Carla Ruiz
2024-09-05 22:22:47.4405720
3 Carlos Gomez
2024-09-06 09:16:50.0709430
4 Joaquin Zarate
2024-09-06 09:19:37.8987160
5 Luis Rodriguez
2024-09-06 09:23:31.3244340

(5 rows affected)
1> s|

```

4.2 Obtener nuestra App

A. Clonar el repo

https://github.com/ingsoft3ucc/Angular_WebAPINetCore8_CRUD_Sample.git

```

Microsoft Windows [Versión 10.0.22631.4169]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3>git clone https://github.com/ingsoft3ucc/Angular_WebAPINetCore8_CRUD_Sample.git
Cloning into 'Angular_WebAPINetCore8_CRUD_Sample'...
remote: Enumerating objects: 236, done.
remote: Counting objects: 100% (236/236), done.
remote: Compressing objects: 100% (124/124), done.
remote: Total 236 (delta 110), reused 227 (delta 103), pack-reused 0 (from 0)
Receiving objects: 100% (236/236), 486.02 KiB | 1.77 MiB/s, done.
Resolving deltas: 100% (110/110), done.

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3>

```

B. Seguir las instrucciones del README.md del repo clonado prestando atención a la modificación de la cadena de conexión en el appSettings.json para que apunte a la BD creada en 4.1

```

1 appsettings.json
2 {
3   "ConnectionStrings": {
4     "DefaultConnection": "Server=localhost,1433;Initial Catalog=sqlserver;Persist Security Info=False;User ID=sa;Password=Nicolasbergami123_;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;"
5   },
6   "Logging": {
7     "LogLevel": {
8       "Default": "Information",
9       "Microsoft.AspNetCore": "Warning"
10    }
11  },
12  "AllowedHosts": "*"

```

C. Navegar a <http://localhost:7150/swagger/index.html> y probar uno de los controladores para verificar el correcto funcionamiento de la API.

```
lea más sobre la telemetría de las herramientas de la CLI de .NET: https://aka.ms/dotnet-cli-telemetry

-----
Instalar un certificado de desarrollo HTTPS de ASP.NET Core.
Para confiar en el certificado, ejecute "dotnet dev-certs https --trust"
Obtenga información sobre HTTPS: https://aka.ms/dotnet-https

-----
Escribir su primera aplicación: https://aka.ms/dotnet-hello-world
Descubra las novedades: https://aka.ms/dotnet-whats-new
Explore la documentación: https://aka.ms/dotnet-docs
Notificar problemas y encontrar el código fuente en GitHub: https://github.com/dotnet/core
Use "dotnet --help" para ver los comandos disponibles o visite: https://aka.ms/dotnet-cli

-----
Compilando...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPI\NetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi

-----
```

Swagger
Select a definition EmployeeCrudApi v1

EmployeeCrudApi ^{1.0} OAS3

<http://localhost:5000/swagger/v1/swagger.json>

Employee

GET	/api/Employee/GetAll	▼
GET	/api/Employee/GetById	▼
POST	/api/Employee/Create	▼
PUT	/api/Employee/Update	▼
DELETE	/api/Employee/Delete	▼

Schemas

Employee >

Responses

Curl

```
curl -X 'GET' \
'http://localhost:7150/api/Employee/GetAll' \
-H 'accept: text/plain'
```

Request URL

<http://localhost:7150/api/Employee/GetAll>

Server response

Code Details

200

Response body


```
{
  "id": 1,
  "name": "Juan Perez",
  "createdDate": "2024-09-05T00:00:00"
},
{
  "id": 2,
  "name": "Carla Ruiz",
  "createdDate": "2024-09-05T22:22:47.440572"
},
{
  "id": 3,
  "name": "Carlos Gomez",
  "createdDate": "2024-09-06T09:16:50.070943"
},
{
  "id": 4,
  "name": "Joaquin Zarate",
  "createdDate": "2024-09-06T09:19:37.898716"
},
{
  "id": 5,
  "name": "Luis Rodriguez",
  "createdDate": "2024-09-06T09:23:31.324434"
}
}
```











Response headers

```
content-type: application/json; charset=utf-8
date: Wed, 25 Sep 2024 04:52:06 GMT
server: Kestrel
transfer-encoding: chunked
```

D. Navegar a <http://localhost:4200> y verificar el correcto funcionamiento de nuestro front-end Angular

EmployeeCrudAngular

Employees:  [Add New Employee](#)

Id	Name	Created Date		
1	Juan Perez	05/09/2024 00:00:00		
2	Carla Ruiz	05/09/2024 22:22:47		
3	Carlos Gomez	06/09/2024 09:16:50		
4	Joaquin Zarate	06/09/2024 09:19:37		
5	Luis Rodriguez	06/09/2024 09:23:31		

E. Una vez verificado el correcto funcionamiento de la Aplicación procederemos a crear un proyecto de pruebas unitarias para nuestra API.

4.3 Crear Pruebas Unitarias para nuestra API

A. En el directorio raiz de nuestro repo crear un nuevo proyecto de pruebas unitarias para nuestra API

B. Instalar dependencias necesarias

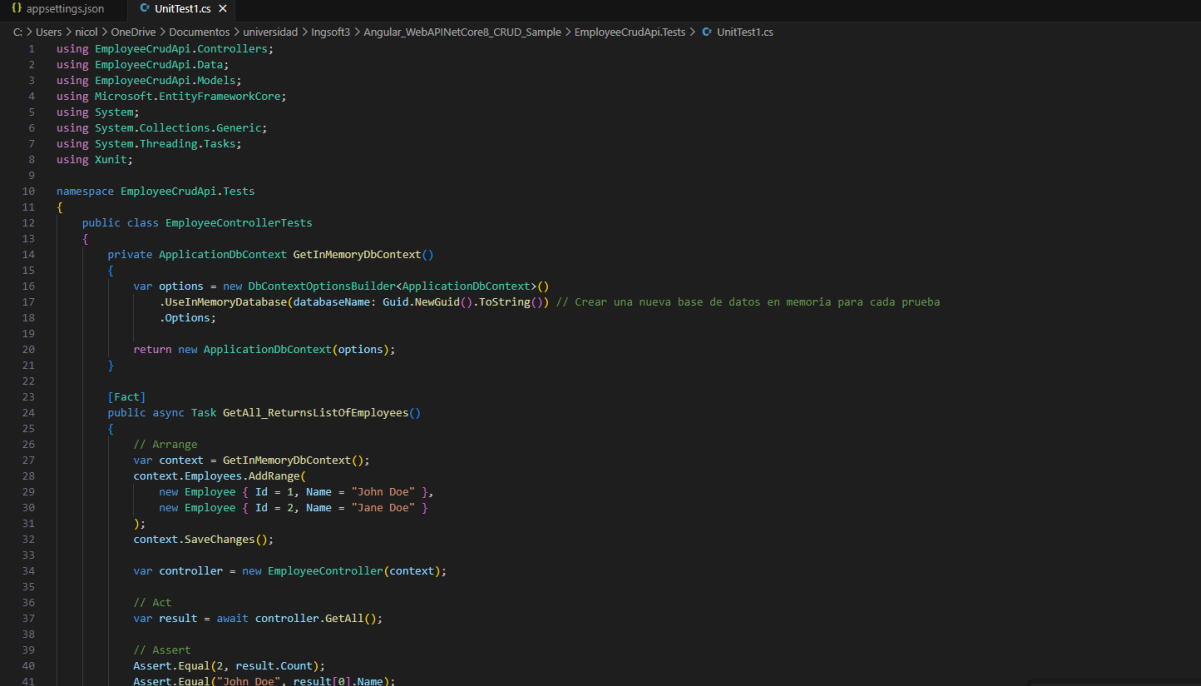
```

info : Generación de archivo MSBuild C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\obj\EmployeeCrudApi.Tests.csproj.nuget.g.props.
info : Generación de archivo MSBuild C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\obj\EmployeeCrudApi.Tests.csproj.nuget.g.targets.
info : Escribiendo el archivo de recursos en el disco. Ruta de acceso: C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\obj\project.assets.json
log : Se ha restaurado C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\EmployeeCrudApi.Tests.csproj (en 655 ms).

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>dotnet
add package Microsoft.EntityFrameworkCore.InMemory
Determinando los proyectos que se van a restaurar...
Writing C:\Users\nicol\AppData\Local\Temp\tmpikgvon.tmp
info : La validación de la cadena de certificados X.509 utilizará el almacén de confianza predeterminado seleccionado por .NET para la firma de código.
info : La validación de la cadena de certificados X.509 utilizará el almacén de confianza predeterminado seleccionado por .NET para la marca de tiempo.
info : Agregando PackageReference para el paquete "Microsoft.EntityFrameworkCore.InMemory" al proyecto "C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\EmployeeCrudApi.Tests.csproj".
info : GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.inmemory/index.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.inmemory/index.json 850 ms
info : GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.inmemory/page/0.0.1-alpha/3.1.3.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.inmemory/page/0.0.1-alpha/3.1.3.json 695 ms
info : GET https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.inmemory/page/3.1.4/6.0.0-preview.7.21378.4.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.entityframeworkcore.inmemory/page/3.1.4/6.0.0-preview.7.21378.4.json 705 ms

```

C. Editar archivo UnitTest1.cs



```

1  using EmployeeCrudApi.Controllers;
2  using EmployeeCrudApi.Data;
3  using EmployeeCrudApi.Models;
4  using Microsoft.EntityFrameworkCore;
5  using System;
6  using System.Collections.Generic;
7  using System.Threading.Tasks;
8  using Xunit;
9
10 namespace EmployeeCrudApi.Tests
11 {
12     public class EmployeeControllerTests
13     {
14         private ApplicationDbContext GetInMemoryDbContext()
15         {
16             var options = new DbContextOptionsBuilder<ApplicationDbContext>()
17                 .UseInMemoryDatabase(databaseName: Guid.NewGuid().ToString()) // Crear una nueva base de datos en memoria para cada prueba
18                 .Options;
19             return new ApplicationDbContext(options);
20         }
21
22         [Fact]
23         public async Task GetAll_ReturnsListOfEmployees()
24         {
25             // Arrange
26             var context = GetInMemoryDbContext();
27             context.Employees.AddRange(
28                 new Employee { Id = 1, Name = "John Doe" },
29                 new Employee { Id = 2, Name = "Jane Doe" }
30             );
31             context.SaveChanges();
32
33             var controller = new EmployeeController(context);
34
35             // Act
36             var result = await controller.GetAll();
37
38             // Assert
39             Assert.Equal(2, result.Count);
40             Assert.Equal("John Doe", result[0].Name);
41         }
42     }
43 }

```

D. Renombrar archivo UnitTest1.cs por EmployeeControllerUnitTests.cs

E. Editar el archivo EmployeeCrudApi.Tests/EmployeeCrudApi.Tests.csproj para agregar una referencia a nuestro proyecto de EmployeeCrudApi reemplazando su contenido por

```

<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <TargetFramework>net8.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>

    <IsPackable>false</IsPackable>
    <IsTestProject>true</IsTestProject>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="coverlet.collector" Version="6.0.0" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.InMemory" Version="8.0.8" />
    <PackageReference Include="Microsoft.NET.Test.Sdk" Version="17.8.0" />
    <PackageReference Include="Moq" Version="4.20.71" />
    <PackageReference Include="xunit" Version="2.9.0" />
    <PackageReference Include="xunit.runner.visualstudio" Version="2.5.3" />
  </ItemGroup>

  <ItemGroup>
    <ProjectReference Include="..\EmployeeCrudApi\EmployeeCrudApi\EmployeeCrudApi.csproj" />
  </ItemGroup>

  <ItemGroup>
    <Using Include="Xunit" />
  </ItemGroup>

</Project>

```

F. Ejecutar los siguientes comandos para ejecutar nuestras pruebas

G. Verificar que se hayan ejecutado correctamente las pruebas

```

Microsoft Windows [Versión 10.0.22631.4169]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>dotnet
build
Determinando los proyectos que se van a restaurar...
Se ha restaurado C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeC
rudApi.Tests\EmployeeCrudApi.Tests.csproj (en 965 ms).
1 de 2 proyectos están actualizados para la restauración.
EmployeeCrudApi -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\Empleo
eCrudApi\EmployeeCrudApi\bin\Debug\net8.0\EmployeeCrudApi.dll
EmployeeCrudApi.Tests -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\E
mployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll

Compilación correcta.
    0 Advertencia(s)
    0 Errores

Tiempo transcurrido 00:00:03.78

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>dotnet
test

```

```
C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>dotnet test
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
EmployeeCrudApi -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi
EmployeeCrudApi\bin\Debug\net8.0\EmployeeCrudApi.dll
EmployeeCrudApi.Tests -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\Employee
EmployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll
Serie de pruebas para C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi
i.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll (.NETCoreApp,Version=v8.0)
Versión 17.11.0 (x64) de VSTest

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error: 0, Superado: 5, Omitido: 0, Total: 5, Duración: 79 ms - EmployeeCrudApi.Tests.dll (net8
.0)

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>
```

H. Verificar que no estamos usando una dependencia externa como la base de datos.

I. Modificar la cadena de conexión en el archivo appsettings.json para que use un usuario o password incorrecto y recompilar el proyecto

EmployeeCrudApi

J. Verificar que nuestro proyecto ya no tiene acceso a la BD navegando a <http://localhost:7150/swagger/index.html> y probando uno de los controladores:

The screenshot shows the Swagger UI for 'EmployeeCrudApi' (1.0 OAS3) at <http://localhost:7150/swagger/v1/swagger.json>. The 'Employee' section is expanded, showing the 'GET /api/Employee/GetAll' endpoint. The 'Parameters' tab is selected, showing 'No parameters'. The 'Execute' button is highlighted. Below, the 'Responses' tab shows a '500 Internal Server Error' response. The 'Response body' is displayed as a long stack trace from Microsoft.Data.SqlClient, indicating a 'Login failed for user 'sqladmin'' error.

K. En la carpeta de nuestro proyecto EmployeeCrudApi.Tests volver a correr las pruebas

L. Verificar que se hayan ejecutado correctamente las pruebas inclusive sin tener acceso a la BD, lo que confirma que es efectivamente un conjunto de

pruebas unitarias que no requieren de una dependencia externa para funcionar.

```
EmployeeCrudApi -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi\bin\Debug\net8.0\EmployeeCrudApi.dll
EmployeeCrudApi.Tests -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll

Compilación correcta.
    0 Advertencia(s)
    0 Errores

Tiempo transcurrido 00:00:03.78

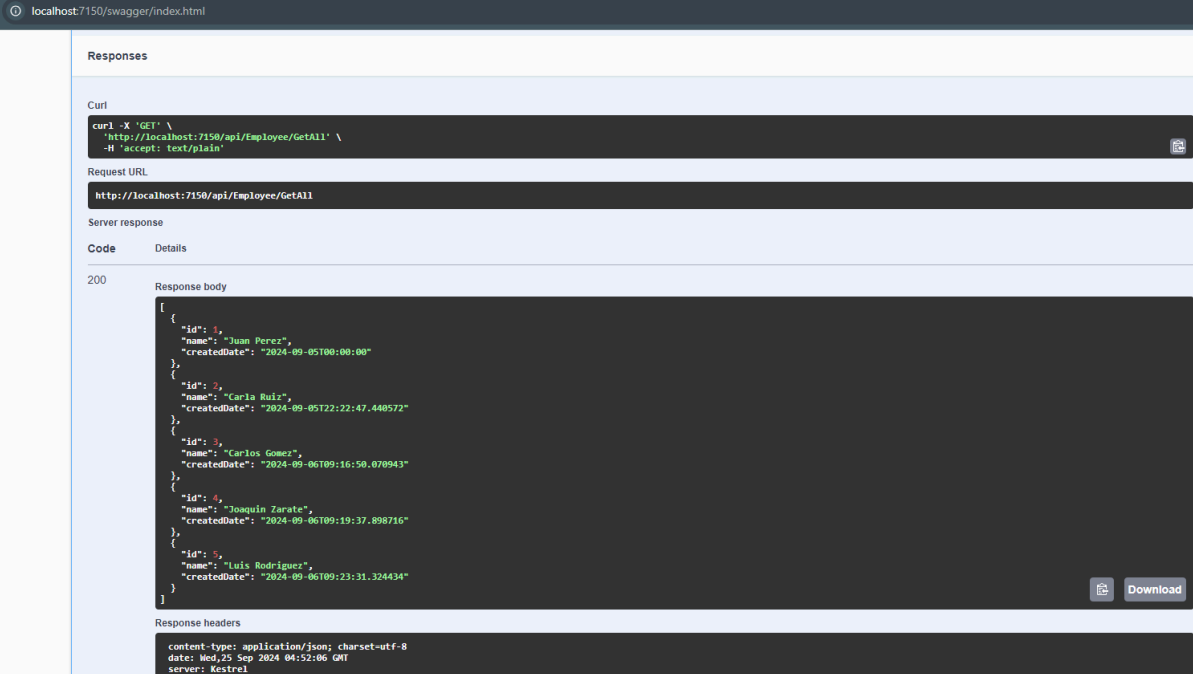
C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>dotnet test
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
EmployeeCrudApi -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi\bin\Debug\net8.0\EmployeeCrudApi.dll
EmployeeCrudApi.Tests -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll
Serie de pruebas para C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll (.NETCoreApp,Version=v8.0)
Versión 17.11.0 (x64) de VSTest

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error:    0, Superado:    5, Omitido:    0, Total:    5, Duración: 79 ms - EmployeeCrudApi.Tests.dll (net8.0)

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi.Tests>
```

M. Modificar la cadena de conexión en el archivo appsettings.json para que use el usuario y password correcto y recompilar el proyecto EmployeeCrudApi
N. Verificar que nuestro proyecto vuelve a tener acceso a la BD navegando a <http://localhost:7150/swagger/index.html> y probando uno de los controladores:



The screenshot shows the Swagger UI interface for a web API. The address bar displays `localhost:7150/swagger/index.html`. The 'Responses' tab is active, showing the details of a GET request to `http://localhost:7150/api/Employee/GetAll`. The request was made using curl with the command: `curl -X 'GET' \ 'http://localhost:7150/api/Employee/GetAll' \ -H 'accept: text/plain'`. The server response is a 200 status code. The response body is a JSON array containing 5 employee records, each with an id, name, and createdAt timestamp. The response headers are also visible at the bottom.

```
Responses

Curl
curl -X 'GET' \
  'http://localhost:7150/api/Employee/GetAll' \
  -H 'accept: text/plain'

Request URL
http://localhost:7150/api/Employee/GetAll

Server response
Code    Details
200
Response body
[
  {
    "id": 1,
    "name": "Juan Perez",
    "createdAt": "2024-09-05T00:00:00"
  },
  {
    "id": 2,
    "name": "Carla Ruiz",
    "createdAt": "2024-09-05T22:22:47.440572"
  },
  {
    "id": 3,
    "name": "Carlos Gomez",
    "createdAt": "2024-09-06T09:16:50.070943"
  },
  {
    "id": 4,
    "name": "Joaquin Zarate",
    "createdAt": "2024-09-06T09:19:37.898716"
  },
  {
    "id": 5,
    "name": "Luis Rodriguez",
    "createdAt": "2024-09-06T09:23:31.324434"
  }
]

Response headers
content-type: application/json; charset=utf-8
date: Wed, 25 Sep 2024 04:52:06 GMT
server: Kestrel
transfer-encoding: chunked
```

4.4 Creamos pruebas unitarias para nuestro front de Angular:

- A. Nos posicionamos en nuestro proyecto de front, en el directorio EmployeeCrudAngular/src/app
- B. Editamos el archivo app.component.spec.ts reemplazando su contenido por:


```
appsettings.json  UnitTests  EmployeeCrudApi.Tests.csproj  TS app.component.spec.ts
C:\Users\nicol> OneDrive\Documents\universidad\Ingsoft3\Angular_WebAPI\NetCore8_CRUD_Sample\EmployeeCrudAngular\src\app> TS app.component.spec.ts > ...
1  import { TestBed } from '@angular/core/testing';
2  import { AppComponent } from './app.component'; // Ajusta la ruta si es necesario
3
4  describe('AppComponent', () => {
5    beforeEach(async () => {
6      await TestBed.configureTestingModule({
7        imports: [AppComponent], // Usa imports en lugar de declarations
8      }).compileComponents();
9    });
10
11    it('should render title', () => {
12      const fixture = TestBed.createComponent(AppComponent);
13      fixture.detectChanges();
14      const compiled = fixture.nativeElement as HTMLElement;
15      expect(compiled.querySelector('h1')?.textContent).toContain('EmployeeCrudAngular');
16    });
17
18  });
```

C. Creamos el archivo `employee.service.spec.ts` reemplazando su contenido por:

```
TS employee.service.spec.ts X
TS employee.service.spec.ts > ...
1  import { TestBed } from '@angular/core/testing';
2  import { HttpClientTestingModule, HttpTestingController } from '@angular/common/http/testing';
3  import { EmployeeService } from './employee.service';
4  import { Employee } from './employee.model';
5  import { DatePipe } from '@angular/common';
6
7  describe('EmployeeService', () => {
8    let datePipe: DatePipe;
9    let httpMock: HttpTestingController;
10    let service: EmployeeService;
11
12    beforeEach(() => {
13      TestBed.configureTestingModule({
14        imports: [HttpClientTestingModule],
15        providers: [
16          EmployeeService,
17          DatePipe
18        ]
19      });
20
21      service = TestBed.inject(EmployeeService);
22      httpMock = TestBed.inject(HttpTestingController);
23      datePipe = TestBed.inject(DatePipe);
24    });
25
26    afterEach(() => {
27      httpMock.verify();
28    });
29
30
31
32    it('should retrieve all employees', () => {
33      const today = new Date();
34      const expectedDateTime = datePipe.transform(today, 'dd/MM/yyyy HH:mm:ss', undefined) ?? ''; // Consistente con el servicio
35
36      const dummyEmployees: Employee[] = [
37        new Employee(1, 'John Doe', expectedDateTime),
38        new Employee(2, 'Jane Smith', expectedDateTime)
39      ];
40
41      service.getAllEmployee().subscribe(employees => {
42        expect(employees.length).toBe(2);
43        employees.forEach((employee, index) => {
44          // Agrega depuración aquí
45          console.log('Employee createdDate:', datePipe.transform(employee.createdDate, 'dd/MM/yyyy HH:mm:ss', undefined) ?? ''); // Imprimir el valor generado por el servicio
46          console.log('Dummy employee createdDate:', datePipe.transform(dummyEmployees[index].createdDate, 'MM/dd/yyyy HH:mm:ss', undefined) ?? ''); // Imprimir el valor esperado
47          expect(datePipe.transform(employee.createdDate, 'dd/MM/yyyy HH:mm:ss', undefined) ?? '').toEqual(datePipe.transform(dummyEmployees[index].createdDate, 'MM/dd/yyyy HH:mm:ss', undefined) ?? '');
48        });
49      });
50    });
51  });
```

D. Editamos el archivo `employee.component.spec.ts` ubicado en la carpeta `employee` reemplazando su contenido por:

```

C: > Users > nicol > OneDrive > Documentos > universidad > Ingsoft3 > Angular_WebAPINetCore8_CRUD_Sample > EmployeeCrudAngular > src > app > employee > TS employee.component.spec.ts > ...
1  import { TestBed } from '@angular/core/testing';
2  import { EmployeeComponent } from './employee.component';
3  import { HttpClientTestingModule } from '@angular/common/http/testing';
4  import { DatePipe } from '@angular/common';
5
6  describe('EmployeeComponent', () => {
7    beforeEach(() => {
8      TestBed.configureTestingModule({
9        imports: [EmployeeComponent, HttpClientTestingModule],
10       providers: [DatePipe] // Añade DatePipe a los proveedores
11     });
12   });
13
14   it('should create', () => {
15     const fixture = TestBed.createComponent(EmployeeComponent);
16     const component = fixture.componentInstance;
17     expect(component).toBeTruthy();
18   });
19 });
20

```

E. Editamos el archivo addemployee.component.spec.ts ubicado en la carpeta addemployee reemplazando su contenido por:

```

C: > Users > nicol > OneDrive > Documentos > universidad > Ingsoft3 > Angular_WebAPINetCore8_CRUD_Sample > EmployeeCrudAngular > src > app > addemployee
1  import { TestBed } from '@angular/core/testing';
2  import { AddemployeeComponent } from './addemployee.component';
3  import { HttpClientTestingModule } from '@angular/common/http/testing';
4  import { ActivatedRoute } from '@angular/router';
5  import { of } from 'rxjs'; // para simular observables
6  import { DatePipe } from '@angular/common';
7
8  describe('AddemployeeComponent', () => {
9    beforeEach(() => {
10      TestBed.configureTestingModule({
11        imports: [AddemployeeComponent, HttpClientTestingModule],
12        providers: [
13          DatePipe,
14          {
15            provide: ActivatedRoute, // Simula ActivatedRoute
16            useValue: {
17              params: of({ id: 1 }) // simula el parametro id en la URL
18            }
19          }
20        ]
21      });
22    });
23
24    it('should create', () => {
25      const fixture = TestBed.createComponent(AddemployeeComponent);
26      const component = fixture.componentInstance;
27      expect(component).toBeTruthy();
28    });
29  });

```

F. En el directorio raiz de nuestro proyecto EmployeeCrudAngular ejecutamos el comando

G. Vemos que se abre una ventana de Karma con Jasmine en la que nos indica que los tests se ejecutaron correctamente

Karma v 6.4.3 - connected; test: complete;



4 specs, 1 failure, randomized with seed 35182

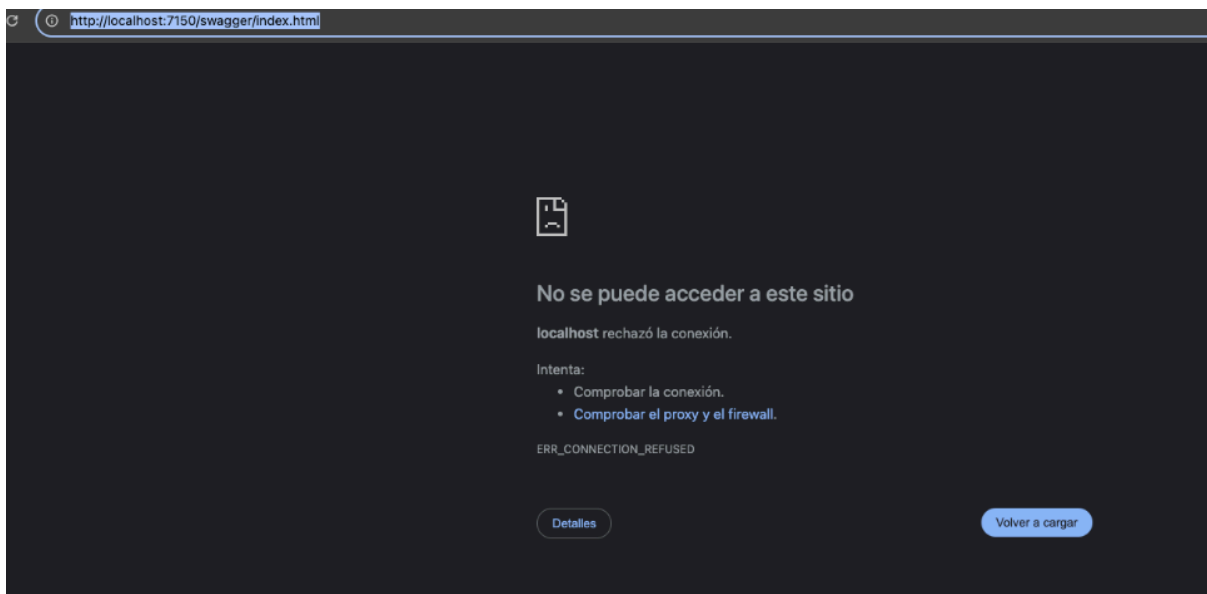
Spec List | [Failures](#)

```
AppComponent
  • should render title
EmployeeComponent
  • should create
AddEmployeeComponent
  • should create
EmployeeService
  • should retrieve all employees
```

H. Vemos que los tests se ejecutaron correctamente:

```
C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPI\NetCore8_CRUD_Sample\EmployeeCrudAngular>ng test
✓ Browser application bundle generation complete.
25 09 2024 02:22:20.541:WARN [karma]: No captured browser, open http://localhost:9876/
25 09 2024 02:22:20.569:INFO [karma-server]: Karma v6.4.3 server started at http://localhost:9876/
25 09 2024 02:22:20.570:INFO [launcher]: Launching browsers Chrome with concurrency unlimited
25 09 2024 02:22:20.577:INFO [launcher]: Starting browser Chrome
25 09 2024 02:22:21.230:INFO [Chrome 128.0.0.0 (Windows 10)]: Connected on socket AAPgoOmYMH5EIPmkAAAB with id 34993467
```

I. Verificamos que no esté corriendo nuestra API navegando a <http://localhost:7150/swagger/index.html> y recibiendo esta salida:



J. Los puntos G y H nos indican que se han ejecutado correctamente las pruebas inclusive sin tener acceso a la API, lo que confirma que es efectivamente un conjunto de pruebas unitarias que no requieren de una dependencia externa para funcionar.

4.5 Agregamos generación de reporte XML de nuestras pruebas de front.

A. Instalamos dependencia karma-junit-reporter

```

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudAngular>npm install karma
a-junit-reporter --save-dev

added 2 packages, and audited 937 packages in 4s

120 packages are looking for funding
  run `npm fund` for details

11 vulnerabilities (7 moderate, 4 high)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudAngular>

```

B. En el directorio raíz de nuestro proyecto (al mismo nivel que el archivo angular.json) creamos un archivo karma.conf.js con el siguiente contenido

```

K karma.conf.js > ...
1  module.exports = function (config) {
2    config.set({
3      frameworks: ['jasmine', '@angular-devkit/build-angular'],
4      plugins: [
5        require('karma-jasmine'),
6        require('karma-chrome-launcher'),
7        require('karma-junit-reporter'),
8        require('@angular-devkit/build-angular/plugins/karma')
9      ],
10     reporters: ['progress', 'junit'],
11     junitReporter: {
12       outputDir: 'test-results',
13       outputFile: 'test-results.xml',
14       useBrowserName: false
15     },
16     port: 9876,
17     colors: true,
18     logLevel: config.LOG_INFO,
19     autoWatch: true,
20     browsers: ['ChromeHeadless'],
21     singleRun: true,
22     restartOnFileChange: true
23   });
24 };

```


C. Ejecutamos nuestros test de la siguiente manera:

```

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudAngular>ng test -
-karma-config=karma.conf.js --watch=false --browsers ChromeHeadless
/ Browser application bundle generation complete.
25 09 2024 03:19:01.842:INFO [karma-server]: Karma v6.4.3 server started at http://localhost:9877/
25 09 2024 03:19:01.845:INFO [launcher]: Launching browsers ChromeHeadless with concurrency unlimited
25 09 2024 03:19:01.852:INFO [launcher]: Starting browser ChromeHeadless
25 09 2024 03:19:02.607:INFO [Chrome Headless 128.0.0.0 (Windows 10)]: Connected on socket y6G718fD34XVoxYtAAAB with id
97286126

```

D. Verificamos que se creo un archivo test-result.xml en el directorio test-results que está al mismo nivel que el directorio src

src		24/9/2024 18:04
test-results		25/9/2024 03:19

4.6 Modificamos el código de nuestra API y creamos nuevas pruebas unitarias:

A. Realizar al menos 5 de las siguientes modificaciones sugeridas al código de la API:

- Al agregar y al editar un empleado, controlar que el nombre del empleado no esté repetido.

Intentamos agregar otro empleado llamado Juan Perez y devuelve bad request porque ese nombre ya existe

POST /api/Employee/Create

Parameters

No parameters

Request body

```
{
  "id": 0,
  "name": "Juan Perez",
  "createdDate": "2024-09-25T07:07:02.924Z"
}
```

Code

Details

400

Undocumented

Error: Bad Request

Response body

```
Employee with this name already exists.
```

Response headers

```
access-control-allow-origin: *  
content-type: text/plain; charset=utf-8  
date: Wed, 25 Sep 2024 07:04:04 GMT  
server: Kestrel  
transfer-encoding: chunked
```

```
Employee.cs EmployeeController.cs • UnitTest1.cs  
Controllers > EmployeeController.cs  
11 {  
12 {  
13 }  
14 }  
34  
35 [HttpPost]  
36 public async Task<IActionResult> Create([FromBody] Employee employee)  
37 {  
38     // Check if the employee name already exists  
39     if (await _context.Employees.AnyAsync(e => e.Name == employee.Name))  
40     {  
41         return BadRequest("Employee with this name already exists.");  
42     }  
43  
44     employee.CreatedDate = DateTime.Now;  
45     await _context.Employees.AddAsync(employee);  
46     await _context.SaveChangesAsync();  
47     return Ok();  
48 }  
49  
50 [HttpPut]  
51 public async Task<IActionResult> Update([FromBody] Employee employee)  
52 {  
53     Employee employeeToUpdate = await _context.Employees.FindAsync(employee.Id);  
54     if (employeeToUpdate == null)  
55     {  
56         return NotFound();  
57     }  
58  
59     // Check if the new name already exists for another employee  
60     if (await _context.Employees.AnyAsync(e => e.Name == employee.Name && e.Id != employee.Id))  
61     {  
62         return BadRequest("Ya existe empleado con es nombre");  
63     }  
64  
65     employeeToUpdate.Name = employee.Name;  
66     await _context.SaveChangesAsync();  
67     return Ok();  
68 }  
69  
70 [HttpDelete]  
71 public async Task Delete(int id)  
72 {  
73     var employeeToDelete = await _context.Employees.FindAsync(id);  
74     _context.Remove(employeeToDelete);  
75     await _context.SaveChangesAsync();  
76 }  
77 }  
78 }
```

- La longitud máxima del nombre y apellido del empleado debe ser de 100 caracteres.

Hacemos un create de un empleado con mas de 100 caracteres en el nombre y devuelve otro bad request

[illegible]

Code	Details
400	
<i>Undocumented</i>	Error: Bad Request
	Response body
	<pre>Empleado no debe tener mas de 100 caracteres en el nombre.</pre>

```
Employee.cs EmployeeController.cs X UnitTest1.cs
Controllers > EmployeeController.cs
11 {
15 {
37 {
40 {
41     return BadRequest("Employee with this name already exists.");
42 }
43
44 // Validate name length
45 if (employee.Name.Length > 100)
46 {
47     return BadRequest("Empleado no debe tener mas de 100 caracteres en el nombre.");
48 }
49
50 employee.CreatedDate = DateTime.Now;
51 await _context.Employees.AddAsync(employee);
52 await _context.SaveChangesAsync();
53 return Ok();
54 }
55
56 [HttpPut]
57 public async Task<ActionResult> Update([FromBody] Employee employee)
58 {
59     Employee employeeToUpdate = await _context.Employees.FindAsync(employee.Id);
60     if (employeeToUpdate == null)
61     {
62         return NotFound();
63     }
64
65     // Check if the new name already exists for another employee
66     if (await _context.Employees.AnyAsync(e => e.Name == employee.Name && e.Id != employee.Id))
67     {
68         return BadRequest("Employee with this name already exists.");
69     }
70
71     // Validate name length
72     if (employee.Name.Length > 100)
73     {
74         return BadRequest("Empleado no debe tener mas de 100 caracteres en el nombre.");
75     }
76
77     employeeToUpdate.Name = employee.Name;
78     await _context.SaveChangesAsync();
79     return Ok();
80 }
```

- Asegurar que el nombre del empleado no contenga caracteres especiales o números, a menos que sea necesario (por ejemplo, caracteres especiales en apellidos como "O'Connor" o "García").
- Validar que el nombre tenga un número mínimo de caracteres, por ejemplo, al menos dos caracteres para evitar entradas inválidas como "A".

En este caso intentamos en crear un Empleado con solo 1 caracter en su nombre y vuelve a devolver bad request porque deben ser 2 caracteres minimo

POST

/api/Employee/Create

Parameters

No parameters

Request body

```
{  "id": 0,  "name": "A",  "createdDate": "2024-09-25T07:18:38.687Z"}}
```

Code	Details
400	Error: Bad Request
<i>Undocumented</i>	<div><div>Response body</div><div>El nombre del empleado debe tener al menos 2 caracteres.</div></div>

```
// Validate minimum name length
if (employee.Name.Length < 2)
{
    return BadRequest("El nombre del empleado debe tener al menos 2 caracteres.");
}
```

- Verificar que el nombre no contenga números, ya que no es común en los nombres de empleados.
- Evitar que se ingresen caracteres repetidos de forma excesiva, como "Juuuaannnnn".

Primero probamos agregando un nombre que contenga numeros, como puede ser Juan10 y devuelve bad request

POST `/api/Employee/Create`

Parameters

No parameters

Request body

```
{
  "id": 0,
  "name": "Juan10",
  "createdDate": "2024-09-25T07:24:57.557Z"
}
```

Request URL

`http://localhost:7150/api/Employee/Create`

Server response

Code	Details
400 <i>Undocumented</i>	Error: Bad Request

Response body

```
El nombre del empleado no debe contener números.
```

Luego probamos agregando un nombre con muchos caracteres repetidos y devuelve bad request

POST /api/Employee/Create

Parameters

No parameters

Request body

```
{
  "id": 0,
  "name": "Juuuuuuuuuuuuuuuan",
  "createdDate": "2024-09-25T07:24:57.557Z"
}
```

Code	Details
400 <i>Undocumented</i>	Error: Bad Request
	Response body
	El nombre del empleado no debe contener caracteres repetidos de forma excesiva.
	Response headers

```
// Validar que no contenga numeros los nombres
if (Regex.IsMatch(employee.Name, @"\d"))
{
    return BadRequest("El nombre del empleado no debe contener números.");
}

// que no contengan caracteres excesivamente repetidos
if (Regex.IsMatch(employee.Name, @"(\.){1,2}"))
{
    return BadRequest("El nombre del empleado no debe contener caracteres repetidos de forma excesiva.");
}
```

B. Crear las pruebas unitarias necesarias para validar las modificaciones realizadas en el código

```
namespace EmployeeCrudApi.Tests
{
    using EmployeeCrudApi.Controllers;
    using EmployeeCrudApi.Data;
    using EmployeeCrudApi.Models;
    using Microsoft.AspNetCore.Mvc;
    using Microsoft.EntityFrameworkCore;
    using System.Threading.Tasks;
```

```

using Xunit;

public class UnitTest1
{
    private readonly EmployeeController _controller;
    private readonly ApplicationDbContext _context;

    public UnitTest1()
    {
        var options = new
DbContextOptionsBuilder<ApplicationDbContext>()
        .UseInMemoryDatabase(databaseName: "TestDatabase")
        .Options;
        _context = new ApplicationDbContext(options);
        _controller = new EmployeeController(_context);
    }

    [Fact]
    public async Task
Create_ShouldReturnBadRequest_WhenNameIsDuplicate()
    {
        // Arrange
        var employee = new Employee { Name = "John Doe" };
        await _context.Employees.AddAsync(employee);
        await _context.SaveChangesAsync();

        // Act
        var result = await _controller.Create(new Employee { Name =
"John Doe" });

        // Assert
        Assert.IsType<BadRequestObjectResult>(result);
    }

    [Fact]
    public async Task
Create_ShouldReturnBadRequest_WhenNameIsTooLong()
    {
        // Act
        var result = await _controller.Create(new Employee { Name =
new string('A', 101) });

        // Assert

```

```

        Assert.IsType<BadRequestObjectResult>(result);
    }

    [Fact]
    public async Task
Create_ShouldReturnBadRequest_WhenNameIsTooShort()
    {
        // Act
        var result = await _controller.Create(new Employee { Name =
"A" });

        // Assert
        Assert.IsType<BadRequestObjectResult>(result);
    }

    [Fact]
    public async Task
Create_ShouldReturnBadRequest_WhenNameContainsNumbers()
    {
        // Act
        var result = await _controller.Create(new Employee { Name =
"John123" });

        // Assert
        Assert.IsType<BadRequestObjectResult>(result);
    }

    [Fact]
    public async Task
Create_ShouldReturnBadRequest_WhenNameContainsExcessiveRepeatedCharacters()
    {
        // Act
        var result = await _controller.Create(new Employee { Name =
"Juuuuaannnn" });

        // Assert
        Assert.IsType<BadRequestObjectResult>(result);
    }
}

```

```

Microsoft Windows [Versión 10.0.22631.4169]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi\EmployeeCrudApi.Tests>dotnet test
Determinando los proyectos que se van a restaurar...
Todos los proyectos están actualizados para la restauración.
EmployeeCrudApi.Tests -> C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi\EmployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll
Serie de pruebas para C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi\EmployeeCrudApi.Tests\bin\Debug\net8.0\EmployeeCrudApi.Tests.dll (.NETCoreApp,Version=v8.0)
Versión 17.11.0 (x64) de VSTest

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error:      0, Superado:      1, Omitido:      0, Total:      1, Duración: < 1 ms - EmployeeCrudApi.Tests.dll (net 8.0)

C:\Users\nicol\OneDrive\Documentos\universidad\Ingsoft3\Angular_WebAPINetCore8_CRUD_Sample\EmployeeCrudApi\EmployeeCrudApi\EmployeeCrudApi.Tests>

```

4.7 Modificamos el código de nuestro Front y creamos nuevas pruebas unitarias:

A. Realizar en el código del front las mismas modificaciones hechas a la API.

B. Las validaciones deben ser realizadas en el front sin llegar a la API, y deben ser mostradas en un toast como por ejemplo

<https://stackblitz.com/edit/angular12-toastr?file=src%2Fapp%2Fapp.component.ts> o

<https://stackblitz.com/edit/angular-error-toast?file=src%2Fapp%2Fcore%2Frxjs.sops.ts>

C. Crear las pruebas unitarias necesarias en el front para validar las modificaciones realizadas en el código del front.