**Project1 Apply Web API with Angular 4**

**Notes:-**

**1-we will create table Employee**

CREATE TABLE [dbo].[Employee](

[EmployeeID] [int] IDENTITY(1,1) NOT NULL,

[FirstName] [varchar](50) NULL,

[LastName] [varchar](50) NULL,

[EmpCode] [varchar](50) NULL,

[Position] [varchar](50) NULL,

[Office] [varchar](50) NULL,

CONSTRAINT [PK\_Employee] PRIMARY KEY CLUSTERED

**2- In the web api click to the file tab 🡪 New 🡪 Web API Project 🡪 and set the Entity Model to the database called Employee**

**3- Set the following code in the EmployeeController**

**using System;**

**using System.Collections.Generic;**

**using System.Data;**

**using System.Data.Entity;**

**using System.Data.Entity.Infrastructure;**

**using System.Linq;**

**using System.Net;**

**using System.Net.Http;**

**using System.Web.Http;**

**using System.Web.Http.Description;**

**using WebAPI.Models;**

**using System.Web.Http.Cors;**

**namespace WebAPI.Controllers{**

**//[EnableCors(origins: "http://localhost:4200", headers: "\*", methods: "\*")]**

**public class EmployeeController : ApiController{**

**private CompanyDBEntities db = new CompanyDBEntities();**

**// Get All Employees**

**public HttpResponseMessage GetEmployees(){**

**try{**

**using (CompanyDBEntities db = new CompanyDBEntities()){**

**List<Employee> lst = db.Employees.ToList();**

**return Request.CreateResponse(HttpStatusCode.OK, lst);**

**}}**

**catch (Exception ex){**

**return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex.Message);**

**}}**

**// Get Specific Employee**

**public HttpResponseMessage GetEmployee(int id){**

**try{**

**using (CompanyDBEntities db = new CompanyDBEntities()){**

**Employee emp = db.Employees.Where(a => a.EmployeeID == id).FirstOrDefault();**

**if (emp == null){**

**return Request.CreateErrorResponse(HttpStatusCode.NotFound, "The Employee with id " + id.ToString() + " not found");}**

**else{**

**return Request.CreateResponse(HttpStatusCode.OK, emp);**

**}}}**

**catch (Exception ex){**

**return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex.Message);**

**}}**

**public HttpResponseMessage PutEmployee(int id, Employee employee){**

**try{**

**using (CompanyDBEntities db = new CompanyDBEntities()){**

**Employee emp = db.Employees.Where(e => e.EmployeeID == id).FirstOrDefault();**

**if (emp == null){**

**return Request.CreateResponse(HttpStatusCode.NotFound, "Employee With Id " + id.ToString() + " not found");}**

**else{**

**emp.FirstName = employee.FirstName;**

**emp.LastName = employee.LastName;**

**emp.EmpCode = employee.EmpCode;**

**emp.Position = employee.Position;**

**emp.Office = employee.Office;**

**db.SaveChanges();**

**return Request.CreateResponse(HttpStatusCode.OK, "The Employee with New Value " + emp.FirstName + " " + emp.LastName + " " + emp.Position + " " + emp.Office + " is updated");**

**}}}**

**catch (Exception ex){**

**return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex.Message);}}**

**// POST api/Employee**

**[ResponseType(typeof(Employee))]**

**public HttpResponseMessage PostEmployee(Employee employee){**

**try{**

**using (CompanyDBEntities db = new CompanyDBEntities()){**

**db.Employees.Add(employee);**

**db.SaveChanges();**

**return Request.CreateResponse(HttpStatusCode.OK, "The Employee with Full Name "**

**+employee.FirstName+" "+employee.LastName+"is Created");**

**}}**

**catch (Exception ex){**

**return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex.Message);**

**}}**

**// DELETE api/Employee/5**

**[ResponseType(typeof(Employee))]**

**public HttpResponseMessage DeleteEmployee(int id){**

**try{**

**using (CompanyDBEntities db = new CompanyDBEntities()){**

**Employee employee = db.Employees.Find(id);**

**if (employee == null){**

**return Request.CreateErrorResponse(HttpStatusCode.NotFound, "The Employee with id " + id.ToString() + " not found");}**

**else{**

**db.Employees.Remove(employee);**

**db.SaveChanges();**

**return Request.CreateResponse(HttpStatusCode.OK, "The Employee with id " + id.ToString() + " is Deleted");**

**}}}**

**catch (Exception ex){**

**return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ex.Message);**

**}}**

**protected override void Dispose(bool disposing){**

**if (disposing){**

**db.Dispose();}**

**base.Dispose(disposing);**

**}}}**

**4-on CMD go to the node\_module under the folder project**

**Cd AngPro/my-app/node\_modules**

**5-install the rxjs under the path**

**npm install rxjs-compat --save**

**npm install ngx-toastr --save**

**6-on the index.html we will update the code**

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<title>AngularCRUD</title>

<base href="/">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="icon" type="image/x-icon" href="favicon.ico">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta.2/css/bootstrap.min.css" integrity="sha384-PsH8R72JQ3SOdhVi3uxftmaW6Vc51MKb0q5P2rRUpPvrszuE4W1povHYgTpBfshb" crossorigin="anonymous">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">

<script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.css" ></script>

</head>

<body>

<app-root></app-root>

<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"

crossorigin="anonymous"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-beta.2/js/bootstrap.min.js" integrity="sha384-alpBpkh1PFOepccYVYDB4do5UnbKysX5WZXm3XxPqe5iKTfUKjNkCk9SaVuEZflJ"

crossorigin="anonymous"></script>

<script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/toastr.min.js"></script>

</body>

</html>

**7-on the app\_module we will put the following code**

import { BrowserModule } from '@angular/platform-browser';

//we will put the NgModule on the app\_module

import { NgModule } from '@angular/core';

//we will put the FormsModule on the app\_module

import { FormsModule} from '@angular/forms';

import { AppComponent } from './app.component';

//we see that we added the HttpClientModule in the app.module.ts

import {HttpClientModule} from '@angular/common/http';

import { EmployeeListComponent } from './Components/employee-list/employee-list.component';

import {EmployeeServiceService} from './Services/employee-service.service';

import { EmployeesComponent } from './Components/employees/employees.component';

import { EmployeeComponent } from './Components/employee/employee.component';

import { ToastrModule } from 'ngx-toastr';

@NgModule({

declarations: [

AppComponent,

EmployeeListComponent,

EmployeesComponent,

EmployeeComponent

],

imports: [

BrowserModule,

HttpClientModule,

FormsModule,

ToastrModule.forRoot()

],

providers: [EmployeeServiceService],

bootstrap: [AppComponent]

})

export class AppModule { }

**8-on the app-component.html we write**

<!--The content below is only a placeholder and can be replaced.-->

<div style="text-align:center">

<h1>

Welcome to {{ title }}!

</h1>

</div>

<app-employees></app-employees>

**9-we will create the following components**

**A-app-employee**

**B-app-employee-list**

**C-app-employees**

**And we will create service and class**

**10-on the class called EmployeeModule**

export class EmployeeModule {

EmployeeID:number;

FirstName:string;

LastName:string;

EmpCode:string;

Position:string;

Office:string;}

**11-we will create service called employee-service.ts**

import { HttpHeaders } from '@angular/common/http';

import { headersToString } from 'selenium-webdriver/http';

import { Injectable } from '@angular/core';

import { Http, Response, Headers, RequestOptions, RequestMethod } from '@angular/http';

import {EmployeeModule} from '../Services/employee-module';

//we put th e Http in the emp-service.service.ts file as the below code

import {HttpClient,HttpErrorResponse} from '@angular/common/http';

import 'rxjs/add/operator/catch';

import 'rxjs/add/observable/throw';

//we put observable in the service.ts

import { Observable } from 'rxjs/Observable';

declare var toastr:any;

@Injectable({

providedIn: 'root'

})

export class EmployeeServiceService {

constructor(private \_http:HttpClient) { }

Employees:EmployeeModule[];

selectedEmployee:EmployeeModule;

weburl:string = 'http://localhost/WebAPI/api/Employee/';

GetPersons() : Observable<EmployeeModule[]>

{

debugger

const httpOptions =

{

headers: new HttpHeaders({

'accept': 'application/json'

})

};

return this.\_http.get<EmployeeModule[]>(this.weburl,httpOptions).catch(this.errorHandler);

}

GetPerson(id:number) : Observable<EmployeeModule>

{

const httpOptions =

{

headers: new HttpHeaders({

'accept': 'application/json'

})

};

return this.\_http.get<EmployeeModule>(this.weburl+"/"+id,httpOptions).catch(this.errorHandler);

}

postEmployee(emp : EmployeeModule)

{

debugger;

var body = JSON.stringify(emp);

const httpOptions =

{

headers: new HttpHeaders({

'Content-Type': 'application/json',

'Authorization': 'my-auth-token'

})

};

return this.\_http.post<EmployeeModule>(this.weburl,body,httpOptions).catch(this.errorHandler);

}

putEmployee(id, emp) {

debugger;

var body = JSON.stringify(emp);

const httpOptions =

{

headers: new HttpHeaders({

'Content-Type': 'application/json',

'Authorization': 'my-auth-token'

})

};

return this.\_http.put<EmployeeModule>(this.weburl+"/"+id,body,httpOptions).catch(this.errorHandler);

}

DeletePerson(id) {

const httpOptions =

{

headers: new HttpHeaders({

'Content-Type': 'application/json',

'Authorization': 'my-auth-token'

})

};

return this.\_http.delete<EmployeeModule>(this.weburl+"/"+id,httpOptions).catch(this.errorHandler);

}

errorHandler(error:HttpErrorResponse)

{

return Observable.throw(error.message || "Server Error");

}

}

**12-on the employees.html we will write the following code**

<div style="text-align:center">

<h2 class="jumbotron bg-secondary text-white">Employee Register</h2>

</div>

<div class="row">

<div class="col-md-6">

<app-employee></app-employee>

</div>

<div class="col-md-6">

<app-employee-list></app-employee-list>

</div>

</div>

**And in the code behind**

import { Component, OnInit } from '@angular/core';

import {EmployeeServiceService} from '../../Services/employee-service.service';

@Component({

selector: 'app-employees',

templateUrl: './employees.component.html',

styleUrls: ['./employees.component.css'],

providers:[EmployeeServiceService]

})

export class EmployeesComponent implements OnInit {

constructor(private employeeService : EmployeeServiceService) { }

ngOnInit() {

}

}

**13-on the employee-list we will update the code**

<p style="color: red;">{{errorMsg}}</p>

<table class="table table-sm table-hover">

<tr \*ngFor="let employee of empService.Employees">

<td>{{employee.FirstName}} - {{employee.LastName}}</td>

<td>{{employee.EmpCode}}</td>

<td>

<a class="btn" (click)="showForEdit(employee.EmployeeID)">

<i class="fa fa-pencil-square-o"></i>

</a>

<a class="btn text-danger" (click)="onDelete(employee.EmployeeID)">

<i class="fa fa-trash-o"></i>

</a>

</td>

</tr>

</table>

**And in the code behind**

import { Component, OnInit } from '@angular/core';

import {EmployeeServiceService} from '../../Services/employee-service.service';

import {EmployeeModule} from '../../Services/employee-module';

import { ToastrService } from 'ngx-toastr';

@Component({

selector: 'app-employee-list',

templateUrl: './employee-list.component.html',

styleUrls: ['./employee-list.component.css']

})

export class EmployeeListComponent implements OnInit {

errorMsg:string;

constructor(private empService:EmployeeServiceService,private toaster:ToastrService)

{

debugger;

this.ShowAllPersons();

}

ngOnInit() {}

ShowAllPersons()

{

debugger;

this.empService.GetPersons().subscribe(

Data => this.empService.Employees = Data,error => this.errorMsg = error

);

debugger;

this.toaster.success('Record Updated Successfully!', 'Employee Register');}

showForEdit(Empid:number) {

this.empService.GetPerson(Empid).subscribe(

Data => this.empService.selectedEmployee = Data,error => this.errorMsg = error);}

onDelete(id: number) {

if (confirm('Are you sure to delete this record ?') == true) {

this.empService.DeletePerson(id)

.subscribe(x => {

this.ShowAllPersons();

this.toaster.warning("Deleted Successfully","Employee Register");

},error =>{

this.errorMsg = error

})}}}

**14-on the employee we will update the following code**

<p style="color: red;">{{errorMsg}}</p>

<form class="emp-form" #employeeForm="ngForm" (ngSubmit)="onSubmit(employeeForm)">

<input type="hidden" name="EmployeeID" #EmployeeID="ngModel" [(ngModel)]="employeeService.selectedEmployee.EmployeeID">

<div class="form-row">

<div class="form-group col-md-6">

<input class="form-control" name="FirstName" #FirstName="ngModel" [(ngModel)]="employeeService.selectedEmployee.FirstName"

placeholder="First Name" required>

<div class="validation-error" \*ngIf="FirstName.invalid && FirstName.touched">This Field is Required.</div>

</div>

<div class="form-group col-md-6">

<input class="form-control" name="LastName" #LastName="ngModel" [(ngModel)]="employeeService.selectedEmployee.LastName" placeholder="Last Name"

required>

<div class="validation-error" \*ngIf="LastName.invalid && LastName.touched">This Field is Required.</div>

</div>

</div>

<div class="form-group">

<input class="form-control" name="Position" #Position="ngModel" [(ngModel)]="employeeService.selectedEmployee.Position" placeholder="Position">

</div>

<div class="form-row">

<div class="form-group col-md-6">

<input class="form-control" name="EmpCode" #EmpCode="ngModel" [(ngModel)]="employeeService.selectedEmployee.EmpCode" placeholder="Emp Code">

</div>

<div class="form-group col-md-6">

<input class="form-control" name="Office" #Office="ngModel" [(ngModel)]="employeeService.selectedEmployee.Office" placeholder="Office">

</div>

</div>

<div class="form-row">

<div class="form-group col-md-8">

<button [disabled]="!employeeForm.valid" type="submit" class="btn btn-lg btn-block btn-info">

<i class="fa fa-floppy-o"></i> Submit</button>

</div>

<div class="form-group col-md-4">

<button type="button" class="btn btn-lg btn-block btn-secondary" (click)="resetForm(employeeForm)">

<i class="fa fa-repeat"></i> Reset</button>

</div>

</div>

</form>

**And in the code behind**

import { Component, OnInit } from '@angular/core';

import { NgForm } from '@angular/forms';

import {EmployeeServiceService} from '../../Services/employee-service.service';

import {EmployeeModule} from '../../Services/employee-module';

import {EmployeeListComponent} from '../employee-list/employee-list.component';

import { ToastrService } from 'ngx-toastr';

@Component({

selector: 'app-employee',

templateUrl: './employee.component.html',

styleUrls: ['./employee.component.css']

})

export class EmployeeComponent implements OnInit {

constructor(private employeeService: EmployeeServiceService,private toaster:ToastrService) {}

private selectedEmployee:EmployeeModule;

errorMsg:string;

ngOnInit() {

this.resetForm();

}

resetForm(form?: NgForm) {

if (form != null)

form.reset();

this.employeeService.selectedEmployee = {

EmployeeID: null,

FirstName: '',

LastName: '',

EmpCode: '',

Position: '',

Office: ''

}

}

ShowAllPersons()

{

this.employeeService.GetPersons().subscribe(Data => {

this.employeeService.Employees = Data;});

}

onSubmit(form: NgForm) {

debugger;

if (form.value.EmployeeID == null) {

this.employeeService.postEmployee(form.value)

.subscribe(data => {

this.resetForm(form);

this.ShowAllPersons();

this.toaster.success('New Record Added Succcessfully', 'Employee Register');

},error =>

{

this.errorMsg = error

})

}

else {

debugger;

this.employeeService.putEmployee(form.value.EmployeeID, form.value)

.subscribe(data => {

this.resetForm(form);

this.ShowAllPersons();

this.toaster.info('Record Updated Successfully!');

},error =>{

this.errorMsg = error

});

}

}

}