Step : 1 Create database using Ms SQL Server

create database Angular\_CRUD\_DB

use Angular\_CRUD\_DB

drop table EmployeeDetails

CREATE TABLE [dbo].[EmployeeDetails](

[EmpId] [int] identity primary key,

[EmpName] [varchar](50) NULL,

[DateOfBirth] [date] NULL,

[EmailId] [varchar](50) NULL,

[Gender] [varchar](10) NULL,

[Address] [varchar](100) NULL,

[PinCode] bigint NULL

)

insert into EmployeeDetails (Empname,DateOfBirth,EmailId,Gender,Address,PinCode)

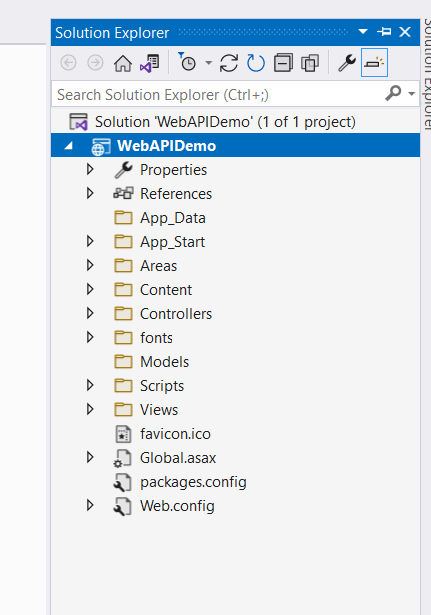
values ('gem','10-08-1985','gem@yahoo.com','female','Tup',2233)

select \* from EmployeeDetails

Step 2:

Open visual studio 2022 (or) 2019

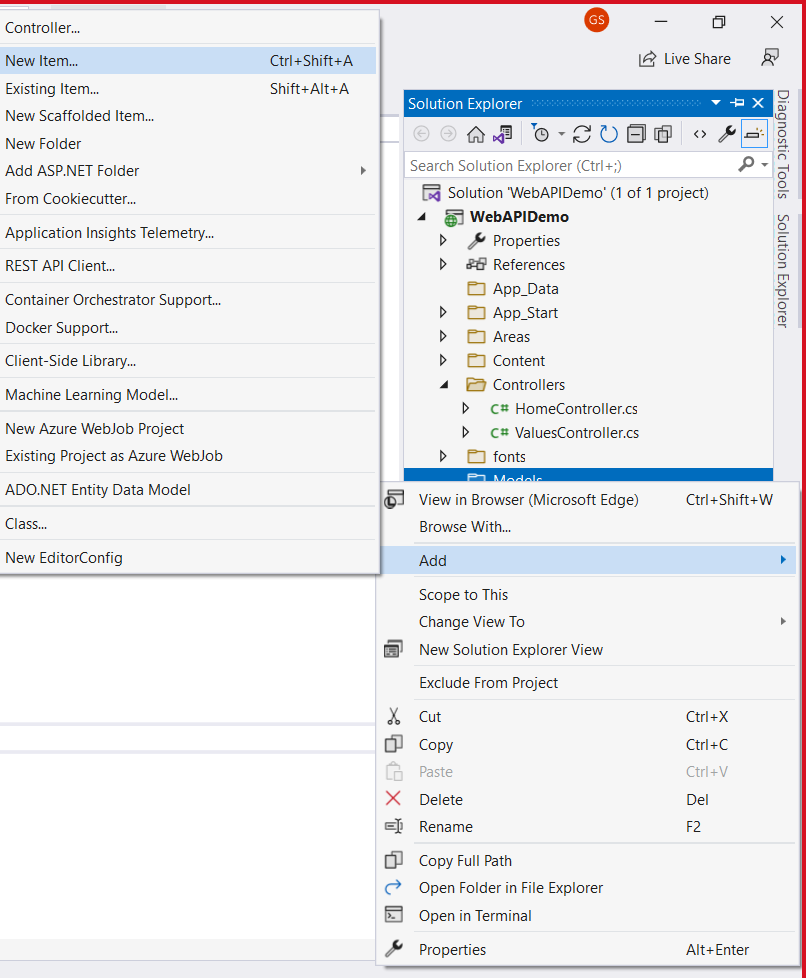
Create new Project🡺 Asp.Net web api (.net framework) 🡺 web API🡺 ok

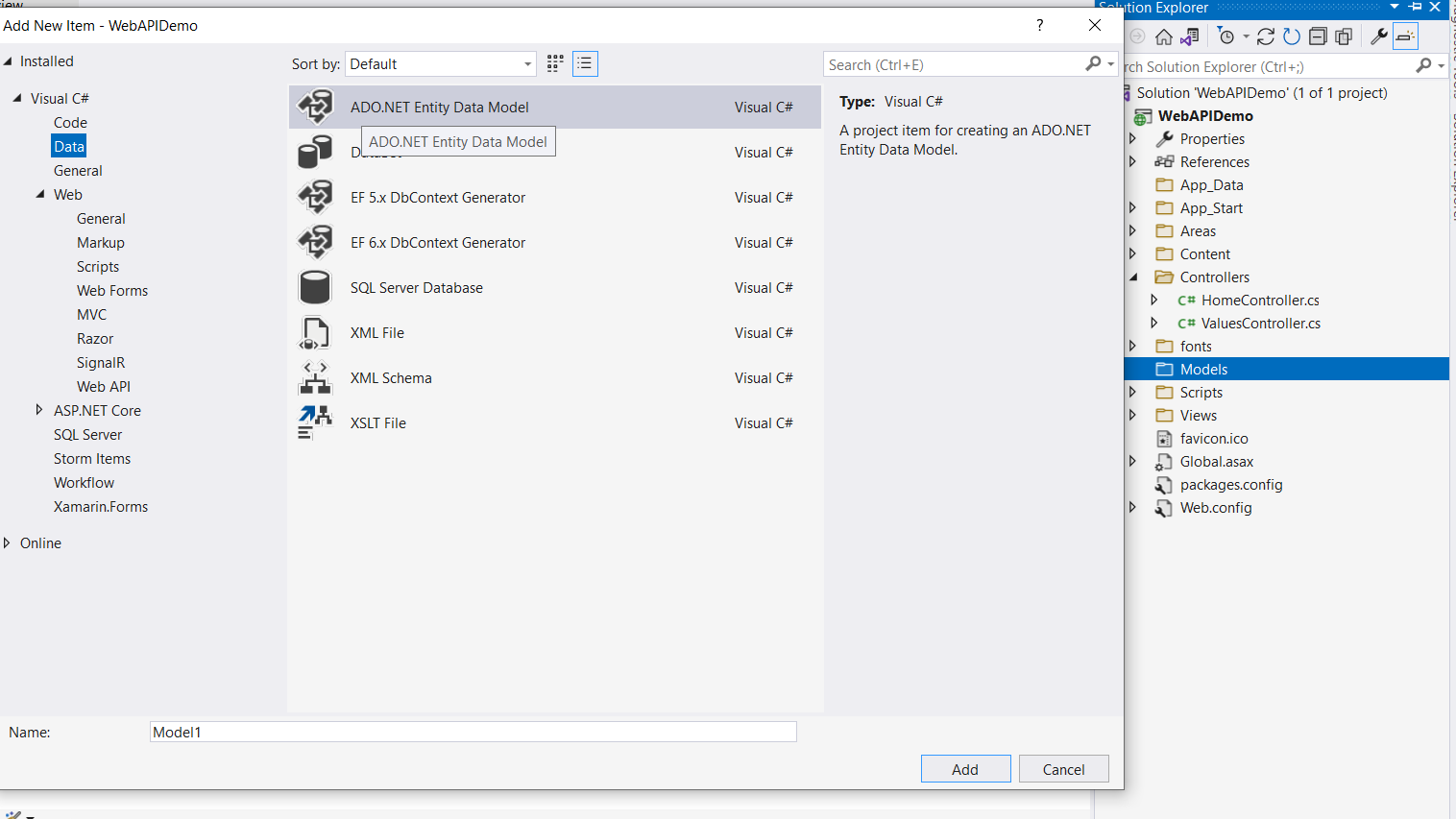


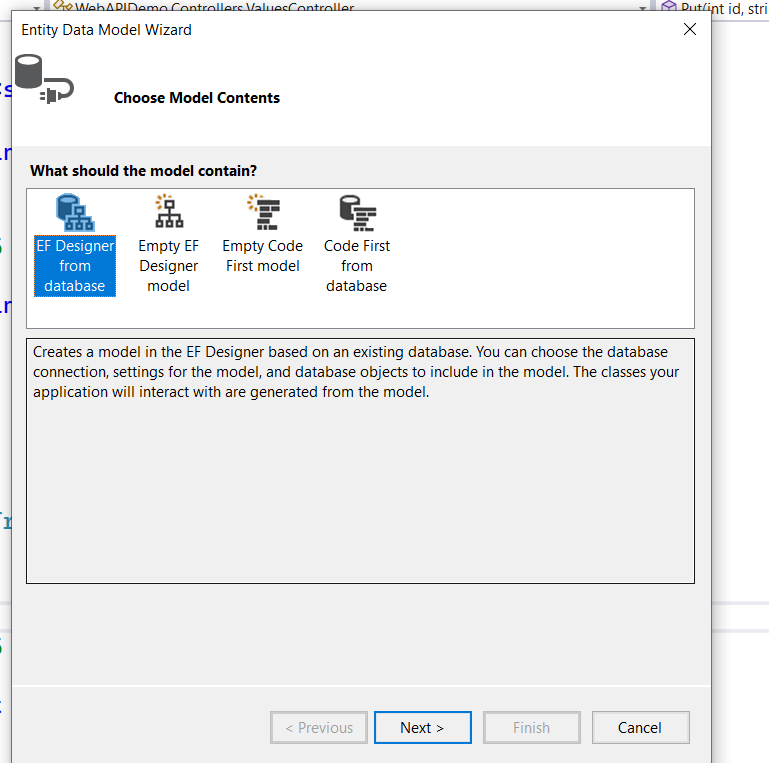
Step 3: web API should communicate with database

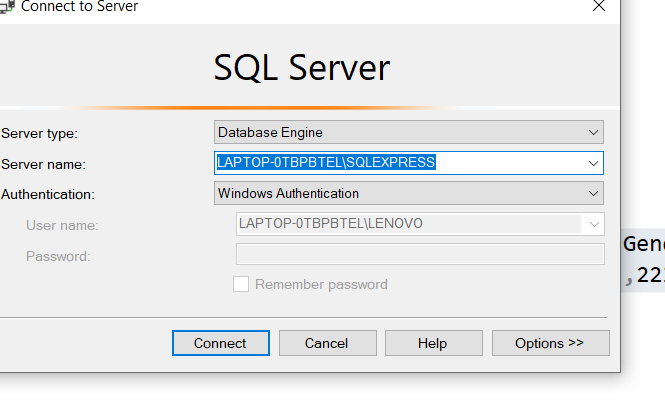
Entity framework database first approach

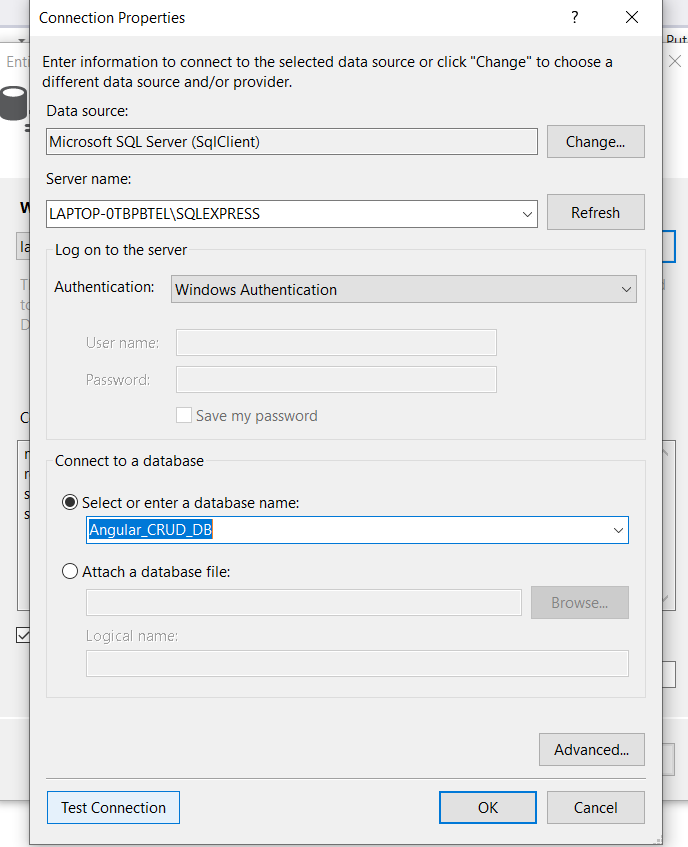
In solution Explorer 🡺 Models 🡺 right click🡺 Add🡺 New Item

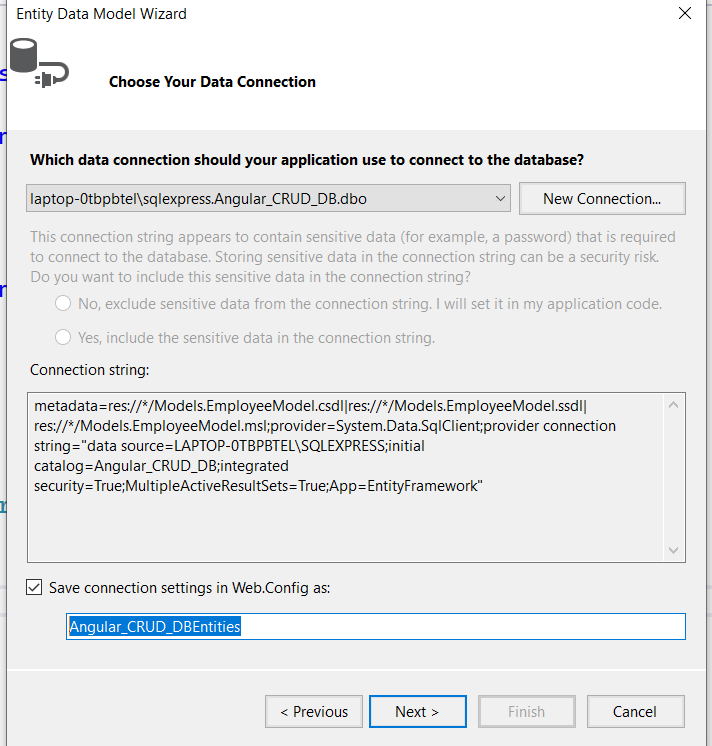


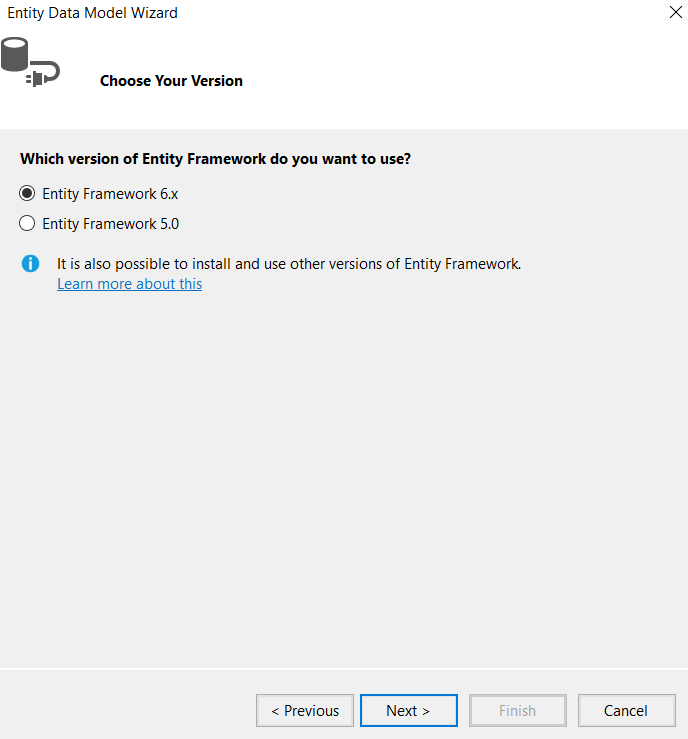


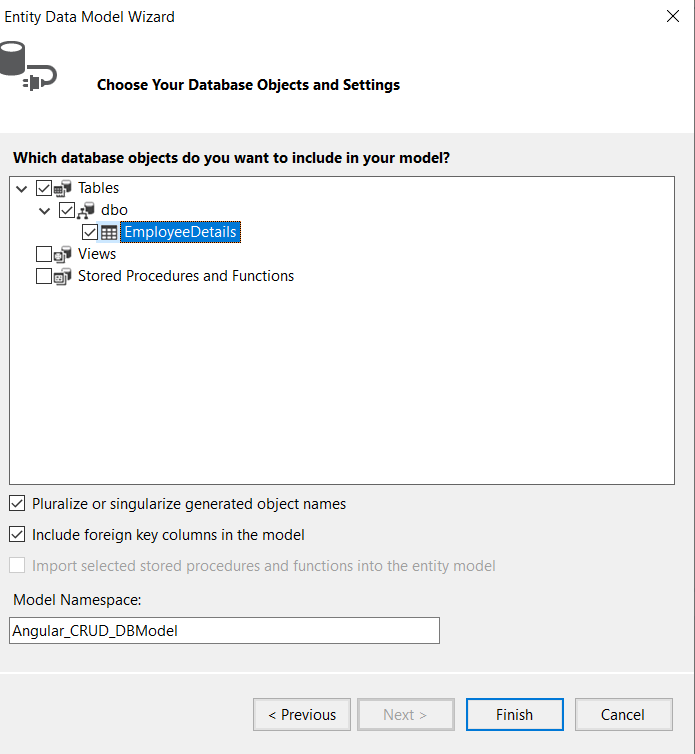


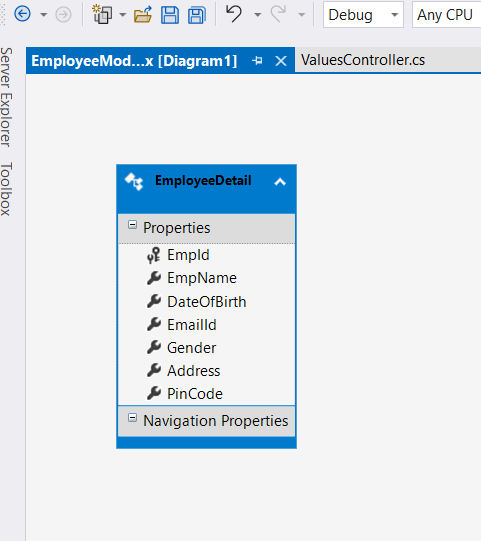






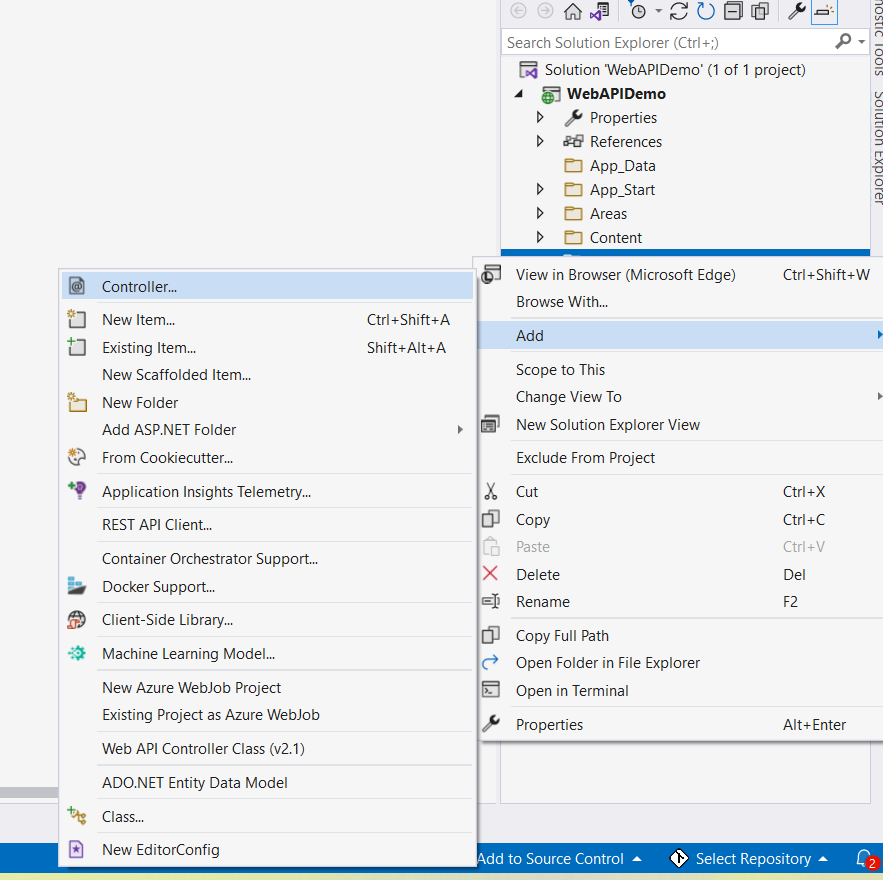


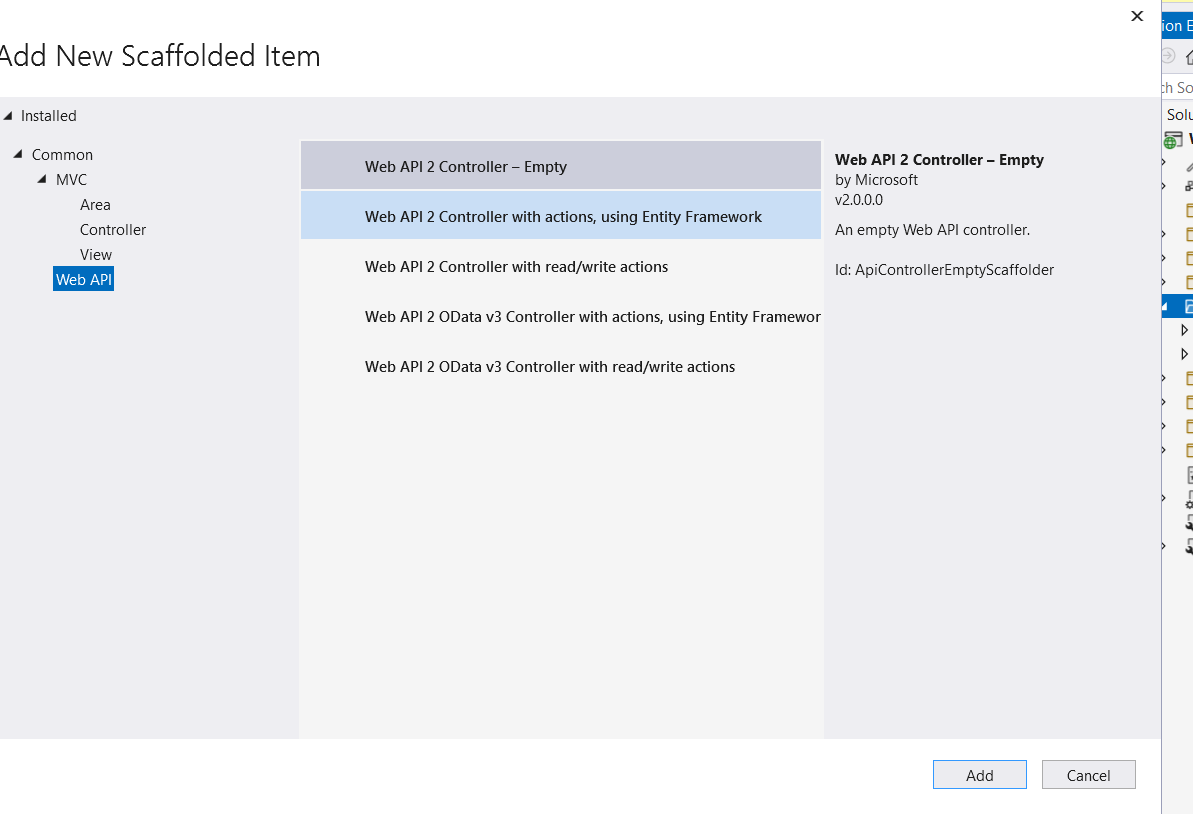


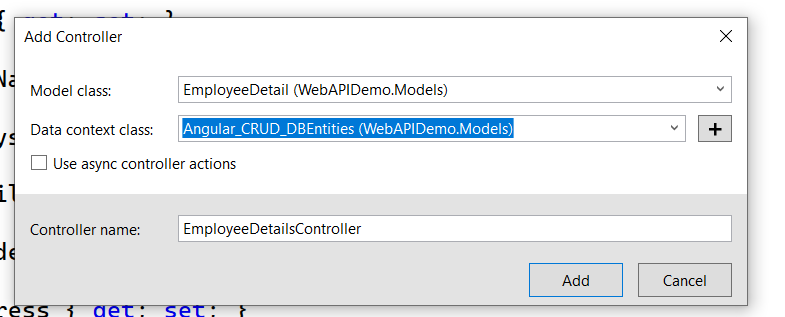


Build your project.

Goto solution Explorer 🡺 right click on Controllers foldes🡺 Add 🡺 Controller







Run your application



Create an Angular application

Write all code to consume web api.

*ng g class employee*

1. **export** **class** Employee {
2. EmpId: string;
3. EmpName: string;
4. DateOfBirth: Date;
5. EmailId: string;
6. Gender: string;
7. Address: string;
8. PinCode: string;

}

To consuming api we need to create employeeservice

ng g s employee

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

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

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

import { Observable } from 'rxjs';

import { Employee } from './employee';

@Injectable({

  providedIn: 'root'

})

export class EmployeeService {

  url = 'https://localhost:44387/api/Employeedetails';

  constructor(private http: HttpClient) { }

  getAllEmployee(): Observable<Employee[]> {

    return this.http.get<Employee[]>(this.url );

  }

  getEmployeeById(employeeId: string): Observable<Employee> {

    return this.http.get<Employee>(this.url +"/" + employeeId);

  }

  createEmployee(employee: Employee): Observable<Employee> {

    const httpOptions = { headers: new HttpHeaders({ 'Content-Type': 'application/json' }) };

    return this.http.post<Employee>(this.url + '/PostEmployeeDetail/', employee, httpOptions);

  }

  updateEmployee(employee: Employee): Observable<Employee> {

    const httpOptions = { headers: new HttpHeaders({ 'Content-Type': 'application/json' }) };

    return this.http.put<Employee>(this.url + '/'+employee.EmpId, employee, httpOptions);

  }

  deleteEmployeeById(employeeid: string): Observable<number> {

    const httpOptions = { headers: new HttpHeaders({ 'Content-Type': 'application/json' }) };

    return this.http.delete<number>(this.url + '/DeleteEmployeeDetail?id=' + employeeid, httpOptions);

  }

}

We are using Material theme for designing (css) purpose

*npm install --save @angular/material @angular/cdk @angular/animations*

ng g c employee

employee.component.ts

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

import { FormBuilder, Validators } from '@angular/forms';

import { Observable } from 'rxjs';

import { EmployeeService } from '../employee.service';

import { Employee } from '../employee';

@Component({

  selector: 'app-employee',

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

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

})

export class EmployeeComponent implements OnInit {

  dataSaved = false;

  employeeForm: any;

  allEmployees: Observable<Employee[]>;

  employeeIdUpdate = null;

  massage = null;

  constructor(private formbulider: FormBuilder, private employeeService: EmployeeService) { }

  ngOnInit() {

    this.employeeForm = this.formbulider.group({

      EmpName: ['', [Validators.required]],

      DateOfBirth: ['', [Validators.required]],

      EmailId: ['', [Validators.required]],

      Gender: ['', [Validators.required]],

      Address: ['', [Validators.required]],

      PinCode: ['', [Validators.required]],

    });

    this.loadAllEmployees();

  }

  loadAllEmployees() {

    this.allEmployees = this.employeeService.getAllEmployee();

  }

  onFormSubmit() {

    this.dataSaved = false;

    const employee = this.employeeForm.value;

    this.CreateEmployee(employee);

    this.employeeForm.reset();

  }

  loadEmployeeToEdit(employeeId: string) {

    this.employeeService.getEmployeeById(employeeId).subscribe(employee => {

      this.massage = null;

      this.dataSaved = false;

      this.employeeIdUpdate = employee.EmpId;

      this.employeeForm.controls['EmpName'].setValue(employee.EmpName);

      this.employeeForm.controls['DateOfBirth'].setValue(employee.DateOfBirth);

      this.employeeForm.controls['EmailId'].setValue(employee.EmailId);

      this.employeeForm.controls['Gender'].setValue(employee.Gender);

      this.employeeForm.controls['Address'].setValue(employee.Address);

      this.employeeForm.controls['PinCode'].setValue(employee.PinCode);

    });

  }

  CreateEmployee(employee: Employee) {

    if (this.employeeIdUpdate == null) {

      this.employeeService.createEmployee(employee).subscribe(

        () => {

          this.dataSaved = true;

          this.massage = 'Record saved Successfully';

          this.loadAllEmployees();

          this.employeeIdUpdate = null;

          this.employeeForm.reset();

        }

      );

    } else {

      employee.EmpId = this.employeeIdUpdate;

      this.employeeService.updateEmployee(employee).subscribe(() => {

        this.dataSaved = true;

        this.massage = 'Record Updated Successfully';

        this.loadAllEmployees();

        this.employeeIdUpdate = null;

        this.employeeForm.reset();

      });

    }

  }

  deleteEmployee(employeeId: string) {

    if (confirm("Are you sure you want to delete this ?")) {

    this.employeeService.deleteEmployeeById(employeeId).subscribe(() => {

      this.dataSaved = true;

      this.massage = 'Record Deleted Succefully';

      this.loadAllEmployees();

      this.employeeIdUpdate = null;

      this.employeeForm.reset();

    });

  }

}

  resetForm() {

    this.employeeForm.reset();

    this.massage = null;

    this.dataSaved = false;

  }

}

App.component.html

<div class="container">

<mat-card>

  <mat-toolbar color="accent">

    <div align="center" style="color:white;text-align: right;">

      CRUD operation in Angular  using Web api and Sql Database

    </div>

  </mat-toolbar>

<br><br>

  <mat-card-content>

    <form [formGroup]="employeeForm" (ngSubmit)="onFormSubmit(employeeForm.value)">

            <table>

              <tr>

                <td class="tbl1">

                  <mat-form-field class="demo-full-width">

                    <input formControlName="EmpName" matTooltip="Enter Employee Name" matInput placeholder="Employee Name">

                  </mat-form-field>

                  <mat-error>

                    <span \*ngIf="!employeeForm.get('EmpName').value && employeeForm.get('EmpName').touched"></span>

                  </mat-error>

                </td>

                <td class="tbl1">

                  <mat-form-field class="demo-full-width">

                    <input matInput [matDatepicker]="picker" matTooltip="Enter Date Of Birth" formControlName="DateOfBirth" placeholder="Choose Date Of Birth">

                    <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>

                    <mat-datepicker #picker></mat-datepicker>

                  </mat-form-field>

                  <mat-error>

                    <span \*ngIf="!employeeForm.get('DateOfBirth').value && employeeForm.get('DateOfBirth').touched"></span>

                  </mat-error>

                </td>

                <td class="tbl1">

                  <mat-form-field class="demo-full-width">

                    <input formControlName="EmailId" matTooltip="Enter EmailId" matInput placeholder="EmailId">

                  </mat-form-field>

                  <mat-error>

                    <span \*ngIf="!employeeForm.get('EmailId').value && employeeForm.get('EmailId').touched"></span>

                  </mat-error>

                </td>

              </tr>

              <tr>

                <td class="tbl1">

                  <span>Gender</span>

                  <br><br>

                  <mat-radio-group matTooltip="Enter Gender" formControlName="Gender">

                      <mat-radio-button value="0">Male</mat-radio-button>&nbsp;&nbsp;

                      <mat-radio-button value="1">Female</mat-radio-button>

                    </mat-radio-group>

                  <mat-error>

                    <span \*ngIf="!employeeForm.get('Gender').value && employeeForm.get('Gender').touched"></span>

                  </mat-error>

                </td>

                <td class="tbl1">

                  <mat-form-field class="demo-full-width">

                    <input matTooltip="Enter Address" formControlName="Address" matInput placeholder="Address">

                  </mat-form-field>

                  <mat-error>

                    <span \*ngIf="!employeeForm.get('Address').value && employeeForm.get('Address').touched"></span>

                  </mat-error>

                </td>

                <td class="tbl1">

                  <mat-form-field class="demo-full-width">

                    <input formControlName="PinCode" matTooltip="Enter Pine Code" matInput placeholder="PinCode">

                  </mat-form-field>

                  <mat-error>

                    <span \*ngIf="!employeeForm.get('PinCode').value && employeeForm.get('PinCode').touched"></span>

                  </mat-error>

                </td>

              </tr>

              <tr>

                <td></td>

                <td  class="content-center">

                  <button type="submit" mat-raised-button color="accent" matTooltip="Click Submit Button" [disabled]="!employeeForm.valid">Submit</button>&nbsp;&nbsp;&nbsp;&nbsp;

                  <button type="reset" mat-raised-button color="accent" matTooltip="Click Reset Button" (click)="resetForm()">Reset</button>

                </td>

                <td>

                  <p \*ngIf="dataSaved" style="color:rgb(0, 128, 0);font-size:20px;font-weight:bold" Class="success" align="left">

                    {{massage}}

                  </p>

                </td>

              </tr>

            </table>

<br><br>

      <table class="table" >

          <tr ngclass="btn-primary">

            <th class="tbl2">Employee Name</th>

            <th class="tbl2">Date Of Birth</th>

            <th class="tbl2">Email Id</th>

            <th class="tbl2">Gender</th>

            <th class="tbl2">Address</th>

            <th class="tbl2">Pine Code</th>

            <th class="tbl2">Edit</th>

            <th class="tbl2">Delete</th>

          </tr>

          <tr \*ngFor="let employee of allEmployees | async">

            <td class="tbl2">{{employee.EmpName}}</td>

            <td class="tbl2">{{employee.DateOfBirth | date }}</td>

            <td class="tbl2">{{employee.EmailId}}</td>

            <td class="tbl2">{{employee.Gender ==0? 'Male' : 'Female'}}</td>

            <td class="tbl2">{{employee.Address}}</td>

            <td class="tbl2">{{employee.PinCode}}</td>

            <td class="tbl2">

              <button type="button" class="btn btn-info" matTooltip="Click Edit Button" (click)="loadEmployeeToEdit(employee.EmpId)">Edit</button>

            </td>

            <td class="tbl2">

              <button type="button" class="btn btn-danger" matTooltip="Click Delete Button" (click)="deleteEmployee(employee.EmpId)">Delete</button>

            </td>

          </tr>

        </table>

    </form>

  </mat-card-content>

</mat-card>

</div>

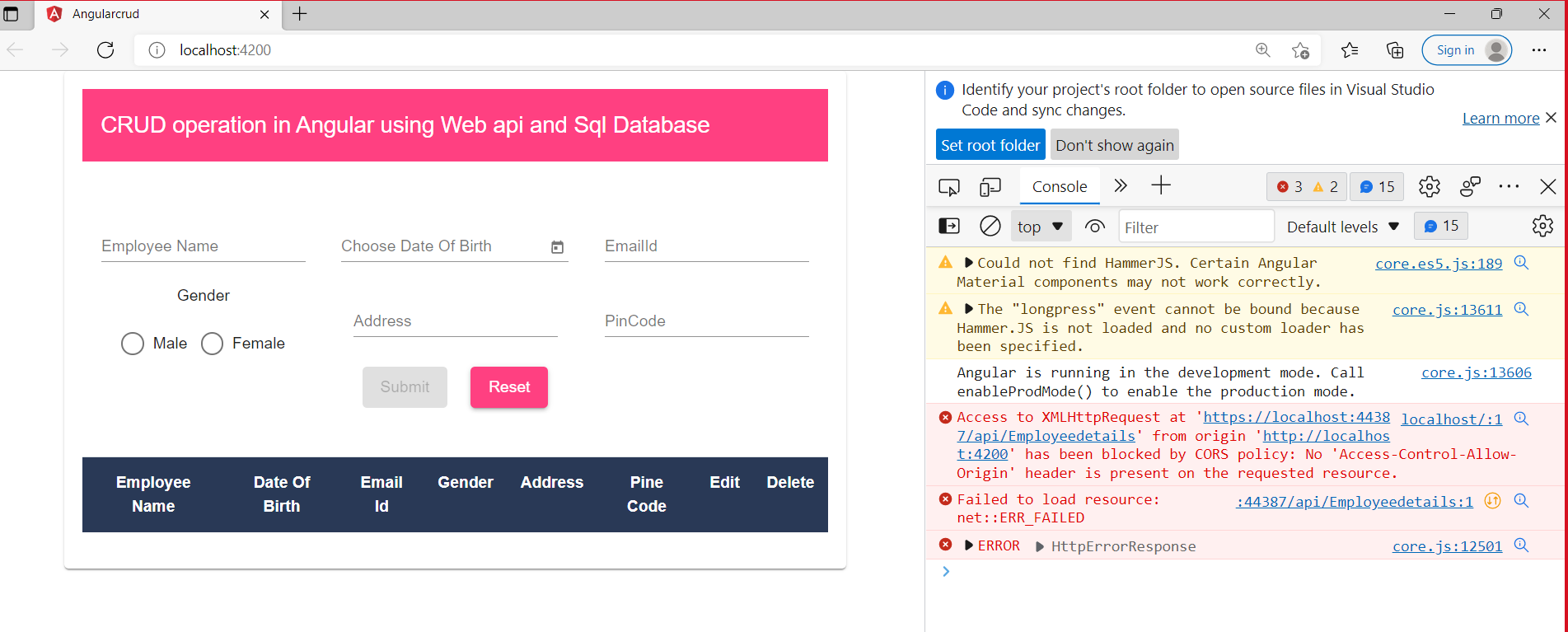
App.component.html

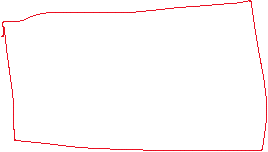
<p>

  <app-employee></app-employee>

</p>

Run your angular app





To resolve the above error goto visual studio 🡺 install the package

