**How to create a Web API project.**

**Step: 1**

**Create Web API Project**

**Step 2**

**Install package:**

**Microsoft.EntityFrameworkCore  
Microsoft.EntityFrameworkCore.SqlServer**

**Step: 3**

**Create a Connection String in appSeting.json**

"ConnectionStrings": {

"defaultConnection": "Server=DESKTOPR1TOFP9; Database=EmpleyeeDB; TrustServerCertificate=True; Trusted\_Connection=True; MultipleActiveResultSets=true"

}

//"conn": "Server=ServerName; Database=DBName; TrustServerCertificate=True; Trusted\_Connection=True; MultipleActiveResultSets=true"

//"conn": "Server=serverName; Initial Catalog=DBName; Persist Security Info=True; User ID= admin; Password=123456; Trusted\_Connection=false; MultipleActiveResultSets=true; Connection Timeout=15; Connection Lifetime=0;"

**Step 4:**

Create DbContext Class

using Microsoft.EntityFrameworkCore;

public class AppDbContext : DbContext

{

public AppDbContext(DbContextOptions<AppDbContext> options) : base(options) { }

public DbSet<Employee> Employee { get; set; }

}

**Step 5:**

Register DbContext in start-up or program file

**program file:**

using Microsoft.EntityFrameworkCore;

builder.Services.AddDbContext<AppDbContext>(options => options.UseSqlServer(builder.Configuration.GetConnectionString("defaultConnection")));

**Start-Up file:**

using Microsoft.EntityFrameworkCore;

services.AddDbContext<AppDbContext>(options =>

options.UseSqlServer(Configuration.GetConnectionString("conn")));

Step 6 : Create Property Class-> database table wali Class

public class Employee | Employee.cs

{

[Key]

public int EmployeeID { get; set; }

public string? FirstName { get; set; }

public DateTime? HireDate { get; set; }

public string? Email { get; set; }

}

**Ab is table ko DbContext Class m Add kro DB Set mai**

**public DbSet<Employee> Employee { get; set; }**

**Step 7: CommonReposneModel Class**

public class ResposeModel | ResposeModel .cs

{

public int MessageCode { get; set; }

public string? Message { get; set; }

public bool Status { get; set; }

public Object? Data { get; set; }

}

Step 8:

Create services |EmployeeServices.cs |

Services m Constructor Create karna hai

Services class ke Constructor m DbClass ko Inject karna hai

public class EmployeeServices: IEmployee

{

private IConfiguration configuration;

private AppDbContext dbContext;

public EmployeeServices(AppDbContext dbContext, IConfiguration configuration)

{

this.dbContext = dbContext;

this.configuration = configuration;

}

public ResposeModel GetEmployee()

{

ResposeModel resposeModel = new();

var empList = dbContext.Employee.ToList();

resposeModel.Data = empList;

return resposeModel;

}

}

Step 9: Create Interface |IEmployee.cs|

public interface IEmployee

{

ResposeModel GetEmployee();

}

Step 10:

Create API Controller |EmployeeController.cs|

Controller m constructor create karna hai

Constructor m service ko inject karna hai

[Route("api/[controller]")]

[ApiController]

public class EmployeeController: ControllerBase

{

private IEmployee iEmplyee;

public EmployeeController(IEmployee iEmplyee)

{

this.iEmplyee = iEmplyee;

}

[HttpGet]

public IActionResult GetEmployee()

{

ResposeModel repoaseModel = new();

repoaseModel= iEmplyee.GetEmployee();

return Ok(repoaseModel);

}

}

Step 11:

Start-up/ program file mai ja ke AddScope<> add kar ke Interface and services Register karni hai

Program.cs

builder.Services.AddScoped<IEmployee, EmployeeServices>();

Startup.cs file

services.AddScoped<IAdmin, Admin>();