**📂 Folder Structure**

dotnetapp/

│

├── Controllers/

│ └── StudentsController.cs

│

├── Models/

│ └── Student.cs

│

├── Data/

│ └── ApplicationDbContext.cs

│

├── appsettings.json

├── Program.cs

├── dotnetapp.csproj

**1. Models/Student.cs**

using System.ComponentModel.DataAnnotations;

namespace dotnetapp.Models

{

public class Student

{

[Key]

public int Id { get; set; }

[Required]

public string Name { get; set; }

[Required]

public string Department { get; set; }

[Required]

public string PhoneNumber { get; set; }

}

}

**2. Data/ApplicationDbContext.cs**

using Microsoft.EntityFrameworkCore;

using dotnetapp.Models;

namespace dotnetapp.Data

{

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options)

{

}

public DbSet<Student> Students { get; set; }

}

}

**3. Controllers/StudentsController.cs**

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using dotnetapp.Data;

using dotnetapp.Models;

namespace dotnetapp.Controllers

{

[ApiController]

[Route("[controller]")]

public class StudentsController : ControllerBase

{

private readonly ApplicationDbContext \_context;

public StudentsController(ApplicationDbContext context)

{

\_context = context;

}

// GET /getAllStudent

[HttpGet("/getAllStudent")]

public async Task<ActionResult<IEnumerable<Student>>> GetStudents()

{

var students = await \_context.Students.ToListAsync();

return Ok(students);

}

// POST /addStudent

[HttpPost("/addStudent")]

public async Task<ActionResult<Student>> CreateStudent(Student student)

{

if (student == null)

{

return BadRequest();

}

\_context.Students.Add(student);

await \_context.SaveChangesAsync();

return CreatedAtAction(nameof(GetStudents), new { id = student.Id }, student);

}

}

}

**4. Program.cs**

using dotnetapp.Data;

using Microsoft.EntityFrameworkCore;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

// CORS: allow React frontend on port 8081

builder.Services.AddCors(options =>

{

options.AddPolicy("AllowReactApp",

policy => policy

.WithOrigins("http://localhost:8081")

.AllowAnyHeader()

.AllowAnyMethod());

});

// Connect to SQL Server

builder.Services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

var app = builder.Build();

app.UseCors("AllowReactApp");

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();

**5. appsettings.json**

{

"ConnectionStrings": {

"DefaultConnection": "User ID=sa;password=examlyMssql@123;server=localhost;Database=appdb;trusted\_connection=false;Persist Security Info=False;Encrypt=False"

},

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"AllowedHosts": "\*"

}

**6. dotnetapp.csproj**

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

<PropertyGroup>

<TargetFramework>net6.0</TargetFramework>

<Nullable>enable</Nullable>

<ImplicitUsings>enable</ImplicitUsings>

<RootNamespace>dotnetapp</RootNamespace>

</PropertyGroup>

<ItemGroup>

<PackageReference Include="Microsoft.EntityFrameworkCore" Version="6.0.6" />

<PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="6.0.6" />

<PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="6.0.6">

<PrivateAssets>all</PrivateAssets>

</PackageReference>

<PackageReference Include="Swashbuckle.AspNetCore" Version="6.2.3" />

</ItemGroup>

</Project>

**🧩 Entity Framework Setup**

Run these commands in terminal inside dotnetapp/:

dotnet restore

dotnet build

# EF Setup

dotnet new tool-manifest

dotnet tool install --local dotnet-ef --version 6.0.6

# Create initial migration

dotnet dotnet-ef migrations add "InitialSetup"

# Apply to database

dotnet dotnet-ef database update

Make sure you already created the appdb database in SQL Server:

sqlcmd -U sa -P examlyMssql@123

1> create database appdb

2> go

**🚀 Run the Backend**

dotnet run

Then open Swagger UI:

http://localhost:8080/swagger

Endpoints:

* GET http://localhost:8080/getAllStudent
* POST http://localhost:8080/addStudent

Both are fully CORS-enabled for React running on http://localhost:8081.

Would you like me to add **a GET /getStudentById/{id} endpoint** next (to improve search accuracy in your React app)?