**Step 1:**

public class Student

{

public int Id { get; set; }

public string Name { get; set; }

public string Batch { get; set; }

public int Marks { get; set; }

}

**Step 2:**

public class StudentDbContext : DbContext

{

**public StudentDbContext(DbContextOptions<StudentDbContext> options)**

**: base(options) { }**

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

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Employee>().

HasData(new Employee

{

Id = 1,

Name = "Ajay",

Batch = "B001",

Marks = 90

}, new Employee

{

Id = 2,

Name = "Deepak",

Batch = "B002",

Marks = 98

}

);

}

}

**Microsoft.EntityFrameworkCore**

**Microsoft.EntityFrameworkCore.SqlServer**

**Microsoft.EntityFrameworkCore.Tools**

**Add the connectionString in AppSettings.json file**

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

**"ConnectionStrings": {**

**"StudentDbContext": "server=LAPTOP-53S2KQS8;database=BookStore;integrated security=true",**

"AllowedHosts": "\*"

}

}

"ConnectionStrings": {

"StudentDBContext": "server=LAPTOP-53S2KQS8\\SqlExpress;database=BookStore;integrated security=true",

"AllowedHosts": "\*"

}

**Step :**

**In Startup.cs file, add**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using EntityFrameDotNetCoreDemo.Models;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.AspNetCore.HttpsPolicy;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.DependencyInjection.Extensions;

using Microsoft.Extensions.Hosting;

namespace EntityFrameDotNetCoreDemo

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services)

{

**services.AddDbContext<StudentDbContext>(op => op.UseSqlServer(Configuration["ConnectionStrings:StudentDbContext"]));**

services.AddControllersWithViews();

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

else

{

app.UseExceptionHandler("/Home/Error");

// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

});

}

}

}

services.AddDbContext<StudentDbContext>(op => op.UseSqlServer(Configuration["ConnectionStrings:StudentDbContext"]));

**With EFCore you do not need to "enable" migrations - they are always-enabled. Just add new migration with Add-Migration.**

**Next Step:**

**Add-Migration name**

**It will build project**

**Next Step:**

**Update-database**

**It will create table in database**

Seed some test data

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Author>().HasData(new Author

{

AuthorId = Guid.NewGuid(),

FirstName = "Bob",

LastName = "Ross",

Genre = "Drama"

}, new Author

{

AuthorId = Guid.NewGuid(),

FirstName = "David",

LastName = "Miller",

Genre = "Fantasy"

});

}

}

After that ,

Add-Migration WebApi.Models.LibraryContextSeedData

Update-database