

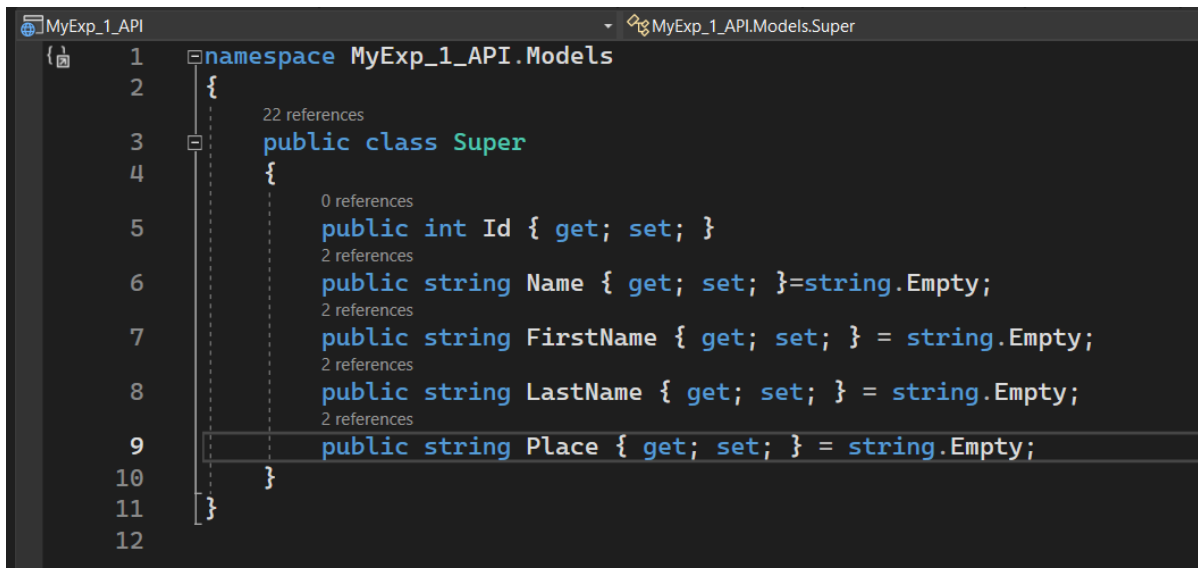
.NET REST API STEPS

Step1: Open Visual Studio 2022 → Create New Project → ASP.NET Core Web API Click on Next.

Step2: Give The Name for the project and Location to Store then Click on Next

Step3: It Shows Additional information leave as default and Click on Create Button.

Step4: Goto Solution Box Right Click on Project Name and Add new Folder “Models”and Right click on “Models” folder and Add Class “Employees”.Add Properties Inside Employees Class like (public int Id {get,set}; etc).



```
1 namespace MyExp_1_API.Models
2 {
3     22 references
4     public class Super
5     {
6         0 references
7         public int Id { get; set; }
8         2 references
9         public string Name { get; set; } = string.Empty;
10        2 references
11        public string FirstName { get; set; } = string.Empty;
12        2 references
13        public string LastName { get; set; } = string.Empty;
14        2 references
15        public string Place { get; set; } = string.Empty;
16    }
17 }
```

Step5:Right Click on Controller folder → Add→Controller.It shows window in left side **Select API→MVC Controller-Empty** Click on Add.It shows the new window, give it name (EmployeesController) and Click on Add.

Step6:

```
{
    [Route("api/[controller]")]
    [ApiController]
    public class SuperContriller : ControllerBase
    {
        private readonly ISuperHeroService _superHeroService;

        public SuperContriller(ISuperHeroService superHeroService)
        {
            _superHeroService = superHeroService;
        }

        [HttpGet]
        public async Task<ActionResult<List<Super>>> GetAllHeros()
        {
            return await _superHeroService.GetAllHeros();
        }

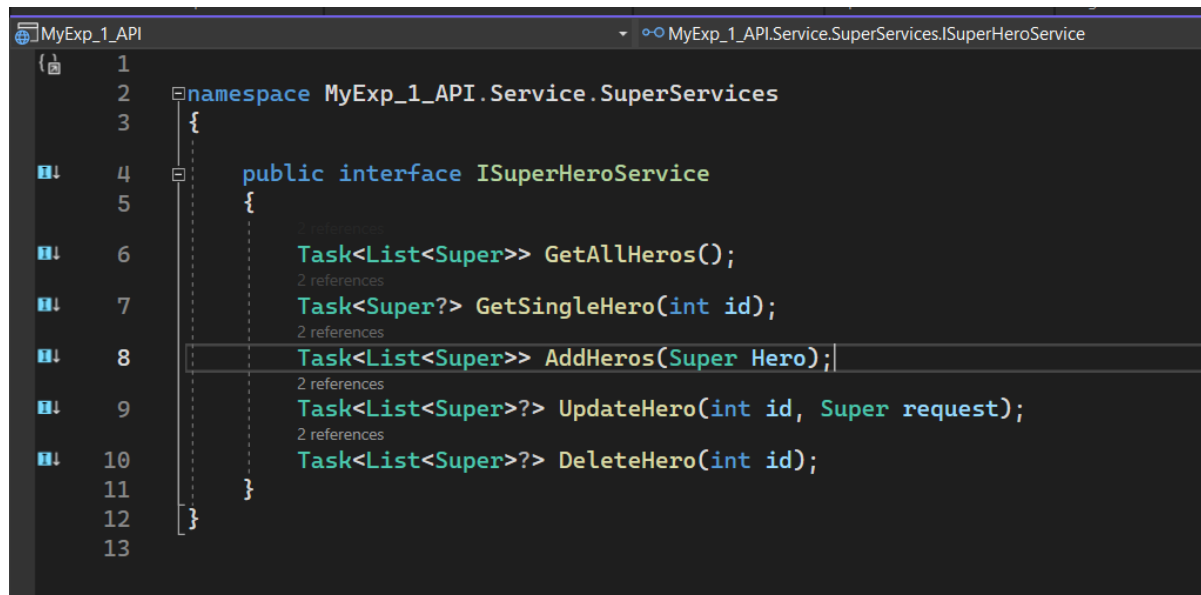
        [HttpGet("{id}")]
        public async Task<ActionResult<Super>> GetSingleHero(int id)
        {
            var result = await _superHeroService.GetSingleHero(id);
            if (result == null)
                return NotFound("Hero not found.");
            return Ok(result);
        }
    }
}

1.API
MyExp_1_API.Controllers.SuperContriller
UpdateHero(ir

34 [HttpPost]
35 public async Task<ActionResult<List<Super>>> AddHeros(Super Hero)
36 {
37     var result = await _superHeroService.AddHeros(Hero);
38     return Ok(result);
39 }
40 [HttpPost]
41 [HttpPut("{id}")]
42 public async Task<ActionResult<List<Super>>> UpdateHero(int id, Super request)
43 {
44     var result = await _superHeroService.UpdateHero(id, request);
45     if (result == null)
46         return NotFound("Hero not found.");
47     return Ok(result);
48 }
49
50 [HttpDelete("{id}")]
51 public async Task<ActionResult<List<Super>>> DeleteHero(int id)
52 {
53     var result = await _superHeroService.DeleteHero(id);
54     if (result == null)
55         return NotFound("Hero not found.");
56     return Ok(result);
57 }
58 }
59 }
```

Step7:Add new Folder “**Services**”.Right click and add another Folder called **EmployeesServices**.Right click on **EmployeesServices** and **Add**→ **New Item**→ **Interface**. Give it name “**IEmployeeService.cs**” and also Add new Class Called “**EmployeesService.cs**”.

IEmployeeService.cs

A screenshot of a Visual Studio code editor window. The title bar shows 'MyExp_1_API'. The breadcrumb navigation at the top right indicates the current file is 'MyExp_1_API.Service.SuperServices.ISuperHeroService'. The code is written in a dark-themed editor. It defines a namespace 'MyExp_1_API.Service.SuperServices' and a public interface 'ISuperHeroService'. The interface contains five methods: 'GetAllHeros()', 'GetSingleHero(int id)', 'AddHeros(Super Hero)', 'UpdateHero(int id, Super request)', and 'DeleteHero(int id)'. Each method signature is preceded by 'Task<List<Super>>' or 'Task<Super?>'. The code is formatted with syntax highlighting, and there are small '2 references' pop-ups for some of the types used. The line numbers 1 through 13 are visible on the left side of the editor.

```
1
2 namespace MyExp_1_API.Service.SuperServices
3 {
4     public interface ISuperHeroService
5     {
6         Task<List<Super>> GetAllHeros();
7         Task<Super?> GetSingleHero(int id);
8         Task<List<Super>> AddHeros(Super Hero);
9         Task<List<Super?>> UpdateHero(int id, Super request);
10        Task<List<Super?>> DeleteHero(int id);
11    }
12 }
13
```

For **EmployeesService.cs**

```
DataContext\ViewModels\Snapshot.cs 223072303284...InitialCreate.cs DataContext.cs superHeroService.cs Program.cs
MyExp_1_API MyExp_1_API.Services.SuperServices.superHeroService
1 using Microsoft.AspNetCore.Http.HttpResults;
2 using MyExp_1_API.Service.SuperServices;
3
4 namespace MyExp_1_API.Services.SuperServices
5 {
6     public class superHeroService : ISuperHeroService
7     {
8
9         private readonly DataContext _context;
10
11         public superHeroService(DataContext context)
12         {
13             _context = context;
14         }
15
16         public async Task<List<Super>> AddHeros(Super Hero)
17         {
18             _context.superHeros.Add(Hero);
19             await _context.SaveChangesAsync();
20             return await _context.superHeros.ToListAsync();
21         }
22
23         public async Task<List<Super>?> DeleteHero(int id)
24         {
25             var Hero = await _context.superHeros.FindAsync(id);
26             if (Hero == null)
```

```
25             if (Hero == null)
26             {
27                 return null;
28                 _context.superHeros.Remove(Hero);
29                 await _context.SaveChangesAsync();
30                 return await _context.superHeros.ToListAsync(); ;
31             }
32
33         public async Task<List<Super>> GetAllHeros()
34         {
35             var heroes = await _context.superHeros.ToListAsync();
36             return heroes;
37         }
38
39         public async Task<Super?> GetSingleHero(int id)
40         {
41             var Hero = await _context.superHeros.FindAsync(id);
42             if (Hero == null)
43             {
44                 return null;
45             }
46             return Hero;
47         }
48
49         public async Task<List<Super>?> UpdateHero(int id, Super request)
50         {
```

```

45     }
46     return Hero;
47 }
48
49 2 references
50 public async Task<List<Super>?> UpdateHero(int id, Super request)
51 {
52     var Hero = await _context.superHeroes.FindAsync(id);
53     if (Hero == null)
54     {
55         return null;
56         Hero.Name = request.Name;
57         Hero.FirstName = request.FirstName;
58         Hero.LastName = request.LastName;
59         Hero.Place = request.Place;
60
61         await _context.SaveChangesAsync();
62         return await _context.superHeroes.ToListAsync();
63     }
64 }
65

```

Step8:

Add this in Program.cs file for Data Context and Scoped

```

13 builder.Services.AddSwaggerGen();
14 builder.Services.AddScoped<ISuperHeroService, superHeroService>();
15 builder.Services.AddDbContext<DataContext>();

```

Step8: **Goto View Menu → Other Windows → Package Manager Console.**

And type a commands

For checking all the packages is install in our project or not

>> dotnet ef

If it shows an error message then install it.

>>dotnet tool install --global dotnet -ef

Then check it.

Step9: Right Click on project file in solution explorer Click on Manage **Nuget Package Manager**

Goto Browse

Search for:

Microsoft.EntityFrameworkCore

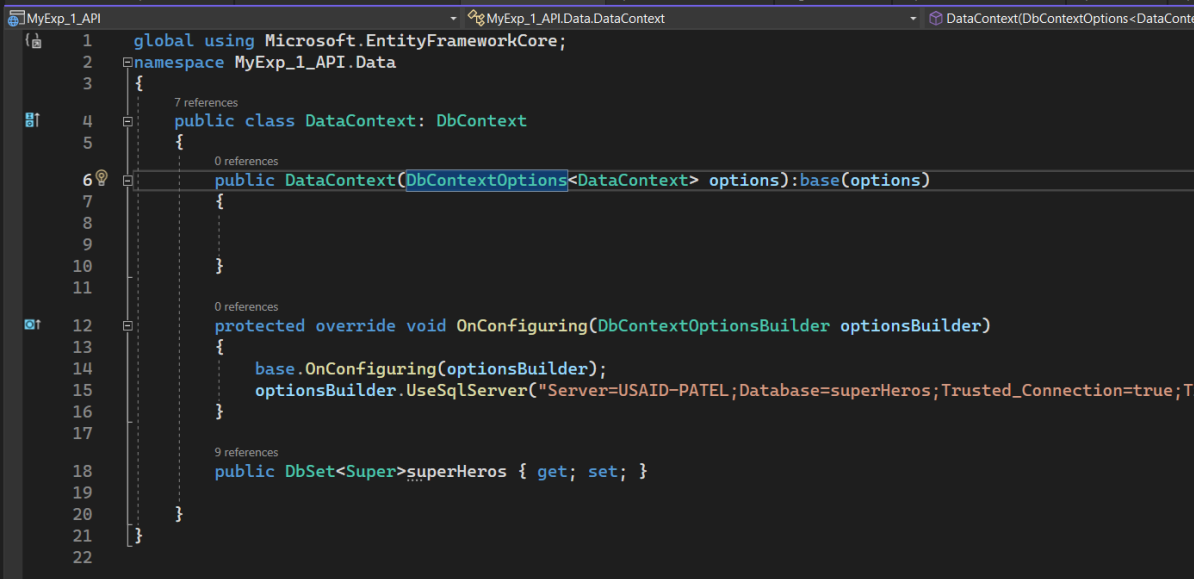
Microsoft.EntityFrameworkCore.Design

Microsoft.EntityFrameworkCore.SqlServer

Install all.

Step10: Add Folder “Data”

Right Click on Data folder and add the new Class Name “DataContext”.



```
1 global using Microsoft.EntityFrameworkCore;
2 namespace MyExp_1_API.Data
3 {
4     public class DataContext: DbContext
5     {
6         public DataContext(DbContextOptions<DataContext> options):base(options)
7         {
8         }
9
10    }
11
12    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
13    {
14        base.OnConfiguring(optionsBuilder);
15        optionsBuilder.UseSqlServer("Server=USAID-PATEL;Database=superHeros;Trusted_Connection=true;T
16    }
17
18    public DbSet<Super>superHeros { get; set; }
19
20 }
21
22 }
```

Step11: Goto Package Console Manager again type commands

```
>>ls //it show list of folder and files
>>cd project folder // in this case Employee
>>cd Employee
>>dotnet ef migrations add InitialCreate //It will create schema
for Database.
```

Step12 : dotnet ef database update

Step13: Run the project. Check all the operations