LAB_3

RetailInventoryApp > Program.cs > ...

```
using RetailInventoryApp;
using RetailInventoryApp.Models;
using System;
using System.Linq;

using System;

0 references
class Program
{
    0 references
    static void Main()
    {
        using var context = new AppDbContext();

        if (!context.Categories.Any())
        {
            var electronics = new Category { Name = "Electronics" };
            var phone = new Product { Name = "Smartphone", Price = 699.99M, Category = electronics };

            context.Categories.Add(electronics);
            context.Products.Add(phone);
            context.SaveChanges();

            Console.WriteLine("Seed data added.");
        }
        else
        {
            Console.WriteLine("Data already exists.");
        }
    }
}
```

RetailInventoryApp > RetailInventoryApp.csproj

```
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <OutputType>Exe</OutputType>
<TargetFramework>net9.0</TargetFramework>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="7.0.0" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="7.0.0" />
  </ItemGroup>
</Project>
```

```
RetailInventoryApp > AppDbContext.cs > AppDbContext > OnConfiguring
1  using Microsoft.EntityFrameworkCore;
2  using RetailInventoryApp.Models;
3
4  namespace RetailInventoryApp
5  {
       1 reference
6      public class AppDbContext : DbContext
7      {
           2 references
8          public DbSet<Category> Categories { get; set; }
           1 reference
9          public DbSet<Product> Products { get; set; }
10
           0 references
11         protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
12         {
13 optionsBuilder.UseSqlServer(
14     "Server=localhost;Database=RetailDB;Trusted_Connection=True;Encrypt=False;");
15         }
16     }
17 }
```

OUTPUT:

```
PS C:\Users\Lenovo\OneDrive\Desktop\RetailInventoryApp\RetailInventoryApp> dotnet ef database update
>> dotnet run
>>
Build started...
Build succeeded.
No migrations were applied. The database is already up to date.
Done.
Data already exists.
```

Lab-4

AppDbContext.cs > ...

```
using Microsoft.EntityFrameworkCore;
using Core.Models;

3 references
public class AppDbContext : DbContext
{
    1 reference
    public DbSet<Category> Categories { get; set; }
    1 reference
    public DbSet<Product> Products { get; set; }

    0 references
    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseSqlite("Data Source=retail.db");
    }
}
```

Program.cs > ...

```
using System;
using System.Threading.Tasks;
using Core.Models;

0 references
class Program
{
    0 references
    static async Task Main(string[] args)
    {
        using var context = new AppDbContext();

        var electronics = new Category { Name = "Electronics" };
        var groceries = new Category { Name = "Groceries" };

        await context.Categories.AddRangeAsync(electronics, groceries);

        var product1 = new Product { Name = "Laptop", Price = 75000M, Category = electronics };
        var product2 = new Product { Name = "Rice Bag", Price = 1200M, Category = groceries };

        await context.Products.AddRangeAsync(product1, product2);
        await context.SaveChangesAsync();

        Console.WriteLine("Initial data inserted successfully.");
    }
}
```

```
namespace Core.Models;

4 references
public class Product
{
    0 references
    public int Id { get; set; }
    2 references
    public string Name { get; set; }
    2 references
    public decimal Price { get; set; }
    2 references
    public Category Category { get; set; }
}

```

AppDbContext.cs > ...

```
using Microsoft.EntityFrameworkCore;
using Core.Models;

3 references
public class AppDbContext : DbContext
{
    1 reference
    public DbSet<Category> Categories { get; set; }
    1 reference
    public DbSet<Product> Products { get; set; }

    0 references
    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseSqlite("Data Source=retail.db");
    }
}
```

OUTPUT:

```
Build started...
Build succeeded.
Acquiring an exclusive lock for migration application. See https://aka.ms/efcore-docs-migrations-lock for more information if this tak
o long.
Applying migration '20250705153242_InitialCreate'.
Done.
Initial data inserted successfully.
PS C:\Users\Lenovo\core> dotnet run
Initial data inserted successfully.
```

Follow link (ctrl + click)

LAB-5

Program.cs > Program

```
using System;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;
using LAB5.Models;

0 references
class Program
{
    0 references
    static async Task Main(string[] args)
    {
        using var context = new AppDbContext();

        if (!await context.Products.AnyAsync())
        {
            var category = new Category { Name = "Electronics" };

            context.Categories.Add(category);

            context.Products.AddRange(
                new Product { Name = "Smartphone", Price = 69999, Category = category },
                new Product { Name = "Laptop", Price = 109999, Category = category },
                new Product { Name = "Headphones", Price = 2999, Category = category }
            );

            await context.SaveChangesAsync();
            Console.WriteLine("Seeded sample data.");
        }

        var products = await context.Products.ToListAsync();
        foreach (var p in products)
            Console.WriteLine($"{p.Name} - ₹{p.Price}");

        var product = await context.Products.FindAsync(1);
        Console.WriteLine($"Found: {product?.Name}");

        var expensive = await context.Products.FirstOrDefaultAsync(p => p.Price > 50000);
        Console.WriteLine($"Expensive: {expensive?.Name}");
    }
}
```

Models > Category.cs > ...

```
using System.Collections.Generic;

namespace LAB5.Models
{
    3 references
    public class Category
    {
        0 references
        public int Id { get; set; }
        1 reference
        public string Name { get; set; } = "";
        0 references
        public List<Product> Products { get; set; } = new();
    }
}
```

Models > Product.cs > ...

```
namespace LAB5.Models
{
    5 references
    public class Product
    {
        0 references
        public int Id { get; set; }
        6 references
        public string Name { get; set; } = "";
        5 references
        public decimal Price { get; set; }
        3 references
        public Category Category { get; set; }
    }
}
```

LAB5.csproj

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

  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net9.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="Microsoft.EntityFrameworkCore" Version="9.0.6" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="9.0.6">
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
      <PrivateAssets>all</PrivateAssets>
    </PackageReference>
    <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="9.0.6" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="9.0.6" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="9.0.6">
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
      <PrivateAssets>all</PrivateAssets>
    </PackageReference>
  </ItemGroup>

</Project>
```

OUTPUT:

```
PS C:\Users\Lenovo\LAB5> dotnet run
>>
C:\Users\Lenovo\LAB5\Models\Product.cs(8,25): warning CS8618: Non-nullable property 'Category' must contain a non-null value when exiting c
onstructor. Consider adding the 'required' modifier or declaring the property as nullable.
Seeded sample data.
Smartphone - ₹69999
Laptop - ₹109999
Headphones - ₹2999
Found: Smartphone
Expensive: Smartphone
```

Retail Inventory

Models > Category.cs > Category

```
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace RetailInventory.Models
{
    4 references
    public class Category
    {
        [Key]
        0 references
        public int Id { get; set; }

        [Required]
    3 references
    public string? Name { get; set; }

    0 references
    public List<Product>? Products { get; set; }
    }
}
```

Models > Product.cs > Product

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace RetailInventory.Models
{
    4 references
    public class Product
    {
        [Key]
        0 references
        public int Id { get; set; }

        [Required]
    3 references
    public string? Name { get; set; }

        3 references
        public int Stock { get; set; }

        [ForeignKey("Category")]
        0 references
        public int CategoryId { get; set; }

    4 references
    public Category? Category { get; set; }
    }
}
```

AppDbContext.cs > AppDbContext > OnConfiguring

```
using Microsoft.EntityFrameworkCore;
using RetailInventory.Models;

3 references
public class AppDbContext : DbContext
{
    2 references
    public DbSet<Product> Products { get; set; }
    2 references
    public DbSet<Category> Categories { get; set; }

    0 references
    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseSqlServer("Server=localhost;Database=RetailDB;Trusted_Connection=True;TrustServerCertifica
    }
}
```

Program.cs > Program > Main

```
using RetailInventory.Models;
using Microsoft.EntityFrameworkCore;
using System;
using System.Linq;

0 references
class Program
{
    0 references
    static void Main()
    {
        using var context = new AppDbContext();

        if (!context.Categories.Any())
        {
            var electronics = new Category { Name = "Electronics" };
            var grocery = new Category { Name = "Grocery" };

            context.Categories.AddRange(electronics, grocery);

            context.Products.AddRange(
                new Product { Name = "Laptop", Stock = 5, Category = electronics },
                new Product { Name = "Banana", Stock = 20, Category = grocery }
            );

            context.SaveChanges();
        }

        Console.WriteLine("Products in Inventory:");
        foreach (var p in context.Products.Include(p => p.Category))
        {
            Console.WriteLine($"{p.Name} ({p.Category.Name}) - Stock: {p.Stock}");
        }
    }
}
```

OUTPUT:

```
Build succeeded in 0.7s
PS C:\Users\Lenovo\RetailInventory> dotnet build
>>
Restore complete (0.7s)
  RetailInventory succeeded with 1 warning(s) (1.2s) → bin\Debug\net9.0\RetailInventory.dll
    C:\Users\Lenovo\RetailInventory\Program.cs(30,44): warning CS8602: Dereference of a possibly null reference.

Build succeeded with 1 warning(s) in 2.3s
PS C:\Users\Lenovo\RetailInventory> dotnet run
Products in Inventory:
Laptop (Electronics) - Stock: 5
Banana (Grocery) - Stock: 20
```

Retail Inventory

Program.cs > Program > Main

```
using System;
using System.Linq;
using Microsoft.EntityFrameworkCore;
using RetailInventory;
using RetailInventory.Models;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        using var context = new AppDbContext();

        if (!context.Categories.Any())
        {
            var electronics = new Category { Name = "Electronics" };
            var phone = new Product { Name = "Smartphone", Price = 699.99M, Category = electronics };

            context.Categories.Add(electronics);
            context.Products.Add(phone);
            context.SaveChanges();
            Console.WriteLine("Data added successfully!");
        }
        else
        {
            var products = context.Products.Include(p => p.Category).ToList();
            foreach (var product in products)
            {
                Console.WriteLine($"{product.Name} - {product.Price:C} - Category: {product.Category.Name}");
            }
        }

        Console.WriteLine("Press any key to exit...");
        Console.ReadKey();
    }
}
```

AppDbContext.cs > AppDbContext

```
using Microsoft.EntityFrameworkCore;
using RetailInventory.Models;

namespace RetailInventory
{
    4 references
    public class AppDbContext : DbContext
    {
        2 references
        public DbSet<Product> Products { get; set; }
        2 references
        public DbSet<Category> Categories { get; set; }

        0 references
        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            optionsBuilder.UseSqlServer("Server=localhost;Database=RetailDb;Trusted_Connection=True;TrustServerCertificate=True;");
        }
        0 references
        protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Product>()
        .Property(p => p.Price)
        .HasColumnType("decimal(18,2)");
}

    }
}
```

Output:

```
PS C:\Users\Lenovo\OneDrive\Desktop\RetailInventory> dotnet run
>>
Smartphone - ₹ 699.99 - Category: Electronics
Press any key to exit...
```