**Week-03 HandsOn Solution**

* **EF Core 8.0 HOL**

**Lab 1: Understanding ORM with a Retail Inventory System**

* **Code:**

**//Model.cs**

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace Models

{

    public class Category

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public List<Product> Products { get; set; } = new();

}

    public class Product

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }

}

    public class AppDbContext : DbContext

    {

        public DbSet<Product> Products { get; set; }

        public DbSet<Category> Categories { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

            optionsBuilder.UseSqlServer("Server=(localdb)\\mssqllocaldb;Database=RetailDb;Trusted\_Connection=True;");

        }

    }

}

//**Program.cs**

Lab 1: ORM Concepts - No code required

**Lab 2: Setting Up the Database Context for a Retail Store**

* **Code:**

**//Model.cs**

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace Models

{

    public class Category

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public List<Product> Products { get; set; } = new();

}

    public class Product

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }

}

    public class AppDbContext : DbContext

    {

        public DbSet<Product> Products { get; set; }

        public DbSet<Category> Categories { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

           optionsBuilder.UseSqlServer("Server=localhost\\SQLEXPRESS01;Database=RetailDb;Trusted\_Connection=True;Encrypt=False;");

        }

    }

}

**//Program.cs**

using System;

using Microsoft.EntityFrameworkCore;

using Models;

class Program

{

    static void Main()

    {

        using var context = new AppDbContext();

        if (context.Database.CanConnect())

        {

            Console.WriteLine("Lab 2 setup successful  — Models and DbContext are working.");

        }

        else

        {

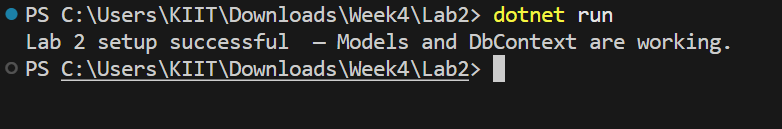
            Console.WriteLine("Lab 2 — Cannot connect to the database. Check your connection string.");

        }

    }

}

* **Output:**



**Lab 3: Using EF Core CLI to Create and Apply Migrations**

* **Code:**

**//Models.cs**

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace Models

{

    public class Category

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public List<Product> Products { get; set; } = new();

}

    public class Product

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }

}

    public class AppDbContext : DbContext

    {

        public DbSet<Product> Products { get; set; }

        public DbSet<Category> Categories { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

            optionsBuilder.UseSqlServer("Server=localhost\\SQLEXPRESS01;Database=RetailDb;Trusted\_Connection=True;Encrypt=False;");

        }

    }

}

**//Program.cs**

using Models;

using System;

class Program

{

    static void Main()

    {

        Console.WriteLine("Run migrations using the following commands:");

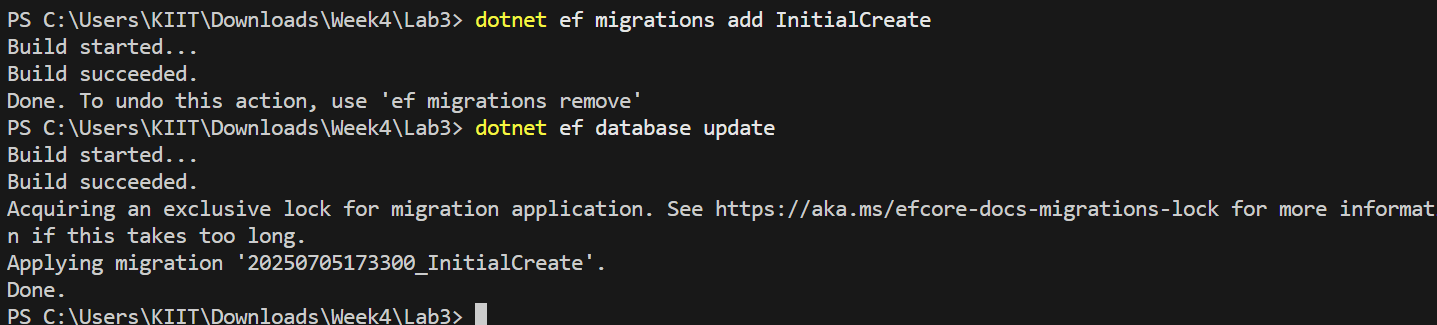
        Console.WriteLine("dotnet ef migrations add InitialCreate");

        Console.WriteLine("dotnet ef database update");

    }

}

* **Output:**



**Lab 4: Inserting Initial Data into the Database**

* **Code:**

**//Models.cs**

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace Models

{

    public class Category

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public List<Product> Products { get; set; } = new();

}

    public class Product

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }

}

    public class AppDbContext : DbContext

    {

        public DbSet<Product> Products { get; set; }

        public DbSet<Category> Categories { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

            optionsBuilder.UseSqlServer("Server=localhost\\SQLEXPRESS01;Database=RetailDb;Trusted\_Connection=True;Encrypt=False;");

        }

    }

}

**//Program.cs**

using System;

using System.Threading.Tasks;

using Models;

class Program

{

    public static async Task Main()

    {

        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 = 75000, Category = electronics };

        var product2 = new Product { Name = "Rice Bag", Price = 1200, Category = groceries };

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

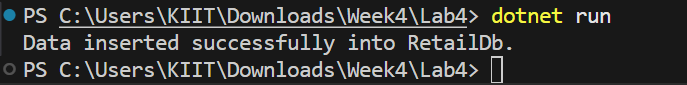
        await context.SaveChangesAsync();

        Console.WriteLine("Data inserted successfully into RetailDb.");

    }

}

* **Output:**



**Lab 5: Retrieving Data from the Database**

* **Code:**

**//Models.cs**

using System.Collections.Generic;

using Microsoft.EntityFrameworkCore;

namespace Models

{

    public class Category

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public List<Product> Products { get; set; } = new();

}

    public class Product

    {

        public int Id { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }

}

    public class AppDbContext : DbContext

    {

        public DbSet<Product> Products { get; set; }

        public DbSet<Category> Categories { get; set; }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

            optionsBuilder.UseSqlServer("Server=localhost\\SQLEXPRESS01;Database=RetailDb;Trusted\_Connection=True;Encrypt=False;");

        }

    }

}

**//Program.cs**

using System;

using System.Threading.Tasks;

using Microsoft.EntityFrameworkCore;

using Models;

class Program

{

    public static async Task Main()

    {

        using var context = new AppDbContext();

        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}");

    }

}

* **Output:**

