**WEEK 3 ASSIGNMENT**

**Task1 : Understanding ORM with a Retail Inventory System**

**Product.cs**

public class Product

{

    public int Id { get; set; }

    public string Name { get; set; }

    public int Quantity { get; set; }

    public decimal Price { get; set; }

}

**RetailContext.cs**

using Microsoft.EntityFrameworkCore;

public class RetailContext : DbContext

{

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

    protected override void OnConfiguring(DbContextOptionsBuilder options)

    {

        options.UseSqlite("Data Source=retail.db");

        // For SQL Server:

        // options.UseSqlServer("Server=(localdb)\\mssqllocaldb;Database=RetailDB;Trusted\_Connection=True;");

    }

}

**Program.cs**

using System;

using System.Linq;

class Program

{

    static void Main()

    {

        using var context = new RetailContext();

        context.Database.EnsureCreated();

        // Add product

        var product = new Product { Name = "Laptop", Quantity = 10, Price = 60000 };

        context.Products.Add(product);

        context.SaveChanges();

        // Read

        Console.WriteLine("\nProducts:");

        var products = context.Products.ToList();

        foreach (var p in products)

            Console.WriteLine($"{p.Id}: {p.Name} - Qty: {p.Quantity}, Price: {p.Price}");

        // Update

        var first = context.Products.First();

        first.Quantity += 5;

        context.SaveChanges();

        Console.WriteLine($"\nUpdated Quantity of '{first.Name}' to {first.Quantity}");

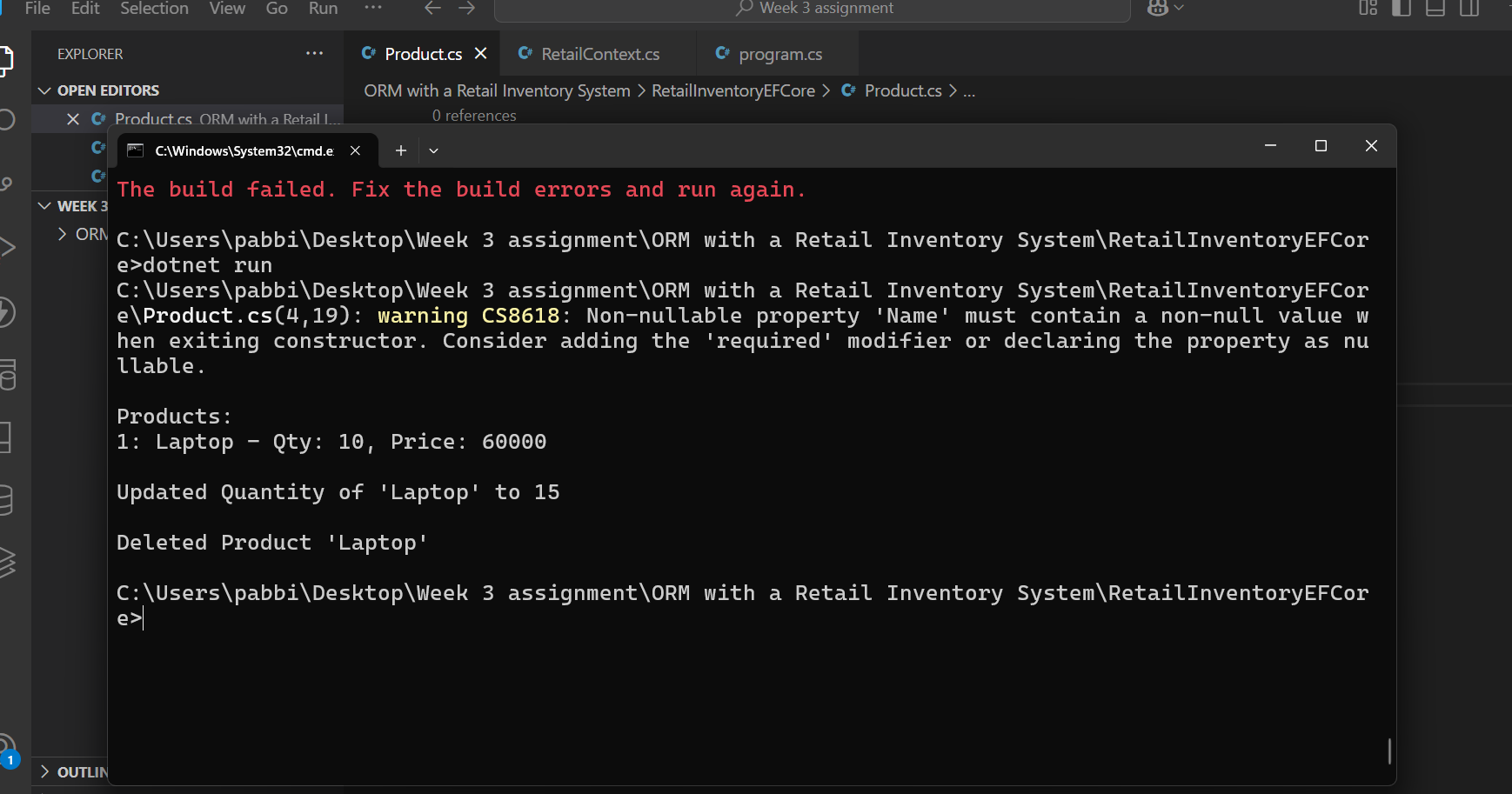
        // Delete

        context.Products.Remove(first);

        context.SaveChanges();

        Console.WriteLine($"\nDeleted Product '{first.Name}'");

    }

}

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

**Source code**

**Category.cs**

namespace RetailStoreEFCore

{

    public class Category

    {

        public int CategoryId { get; set; }

        public string Name { get; set; }

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

    }

}

**Product.cs**

namespace RetailStoreEFCore

{

    public class Product

    {

        public int ProductId { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }

    }

}

**RetailStoreContext.cs**

using Microsoft.EntityFrameworkCore;

namespace RetailStoreEFCore

{

    public class RetailStoreContext : DbContext

    {

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

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

        protected override void OnConfiguring(DbContextOptionsBuilder options)

        {

            options.UseSqlite("Data Source=retailstore.db");

        }

        protected override void OnModelCreating(ModelBuilder modelBuilder)

        {

            modelBuilder.Entity<Category>()

                .HasMany(c => c.Products)

                .WithOne(p => p.Category)

                .HasForeignKey(p => p.CategoryId);

        }

    }

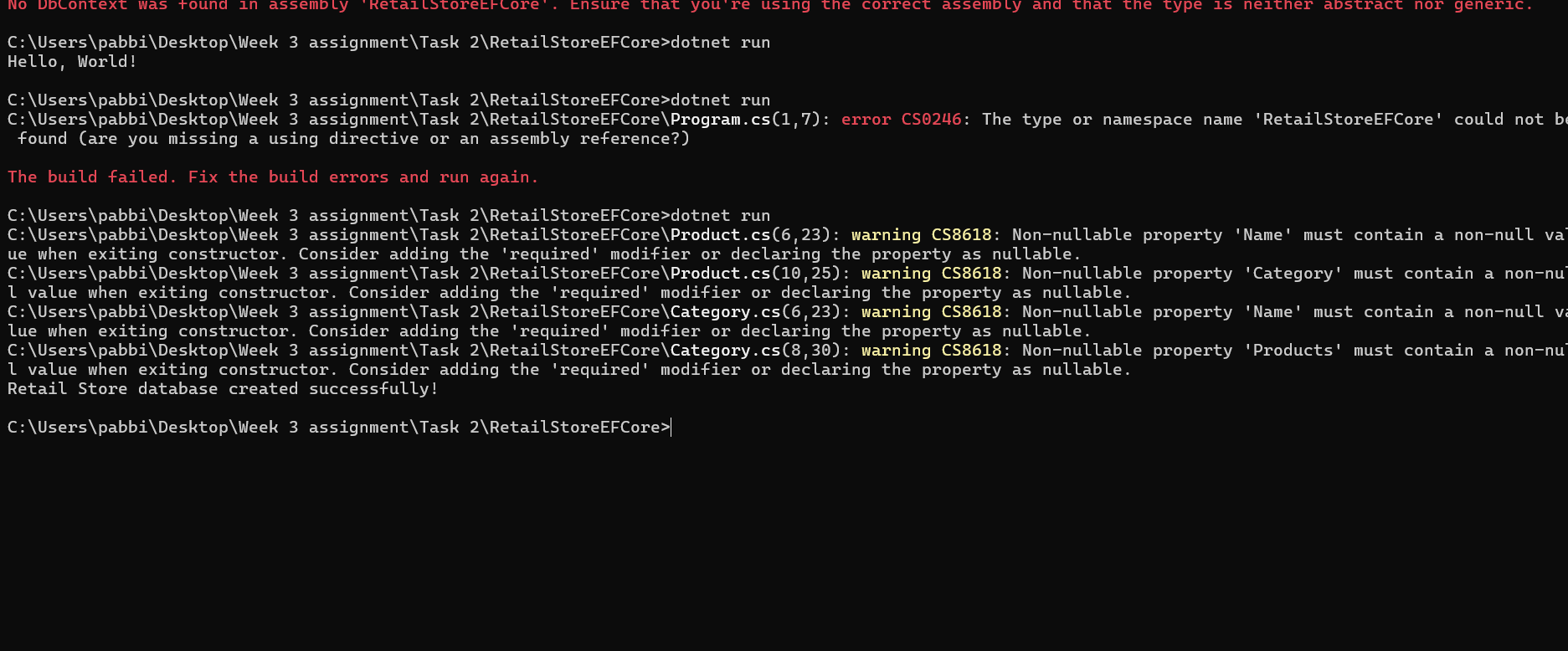
}

**Program.cs**

using var context = new RetailStoreContext();

context.Database.EnsureCreated();

Console.WriteLine("Retail Store database created successfully!");



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

**Source code**

**Product.cs**

namespace EFLab3MigrationDemo

{

    public class Product

    {

        public int ProductId { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

    }

}

**AppDbContext.cs**

using Microsoft.EntityFrameworkCore;

namespace EFLab3MigrationDemo

{

    public class AppDbContext : DbContext

    {

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

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

            optionsBuilder.UseSqlite("Data Source=store.db");

        }

    }

}

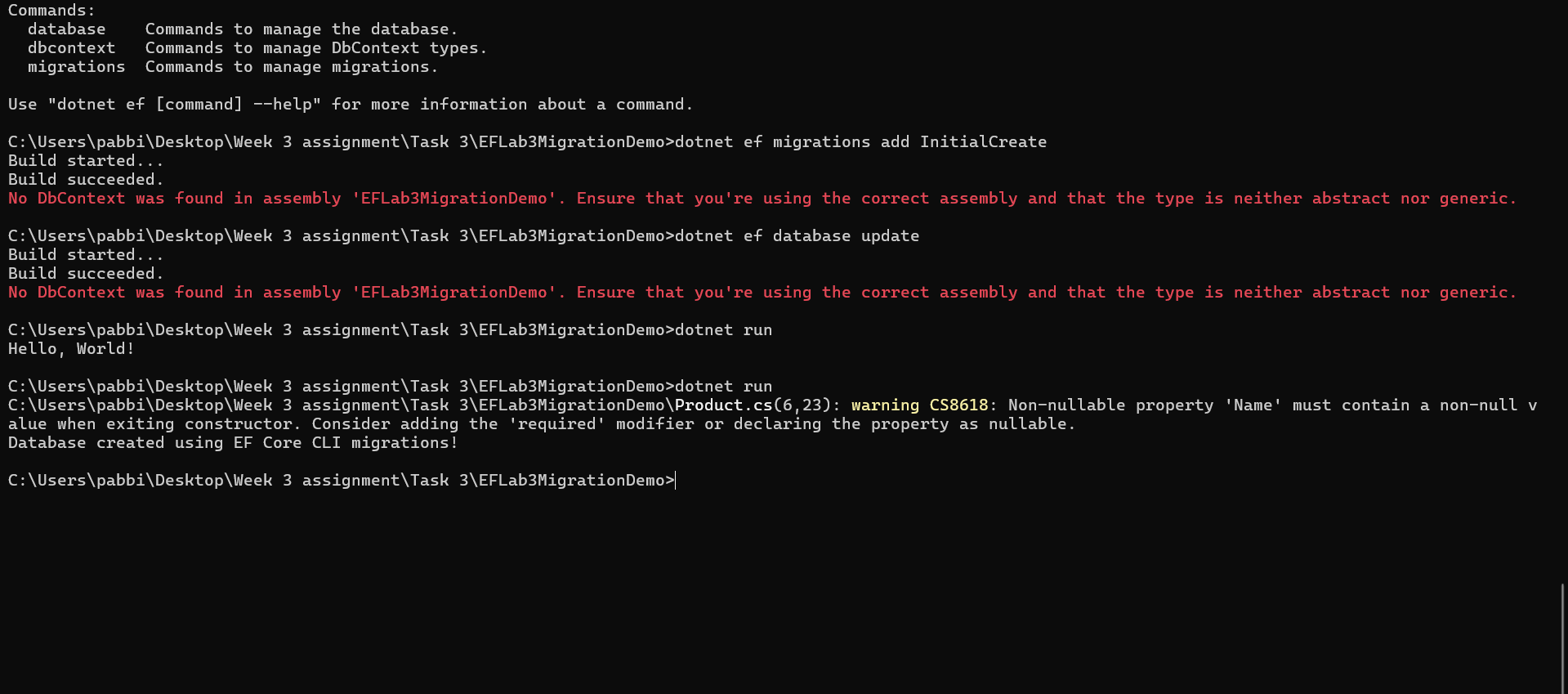
**Program.cs**

using EFLab3MigrationDemo;

using var context = new AppDbContext();

context.Database.EnsureCreated();

Console.WriteLine("Database created using EF Core CLI migrations!");



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

**Source code**

**Product.cs**

namespace EFLab3MigrationDemo

{

    public class Product

    {

        public int ProductId { get; set; }

        public string Name { get; set; }

        public decimal Price { get; set; }

    }

}

**AppDbContext.cs**

using Microsoft.EntityFrameworkCore;

namespace EFLab3MigrationDemo

{

    public class AppDbContext: DbContext

    {

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

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

        {

            optionsBuilder.UseSqlite("Data Source=store.db");

        }

        // Add this method to seed products

        protected override void OnModelCreating(ModelBuilder modelBuilder)

        {

            modelBuilder.Entity<Product>().HasData(

                new Product { ProductId = 1, Name = "Laptop", Price = 75000 },

                new Product { ProductId = 2, Name = "Smartphone", Price = 30000 },

                new Product { ProductId = 3, Name = "Keyboard", Price = 1500 }

            );

        }

    }

}

**Program.cs**

using System;

using System.Linq;

using EFLab3MigrationDemo; // match your other files' namespace

namespace EFLab3MigrationDemo

{

    class Program

    {

        static void Main()

        {

            using var context = new AppDbContext();

            Console.WriteLine("Seeded Products:");

            foreach (var product in context.Products.ToList())

            {

                Console.WriteLine($"{product.ProductId}: {product.Name} - ₹{product.Price}");

            }

        }

    }

}

