Lab 1: Understanding ORM with a Retail Inventory System

Code:

// File: Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using Microsoft.EntityFrameworkCore;

namespace RetailInventorySystem

{

// 1. Product model (represents the database table)

public class Product

{

public int ProductID { get; set; }

public string Name { get; set; }

public string Category { get; set; }

public int Quantity { get; set; }

public decimal Price { get; set; }

}

// 2. Database context (acts as a bridge between C# and the database)

public class RetailDbContext : DbContext

{

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

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

// Use SQLite for simplicity (creates retail.db file)

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

}

}

// 3. Main logic

class Program

{

static void Main(string[] args)

{

using (var db = new RetailDbContext())

{

// Ensure the database is created

db.Database.EnsureCreated();

// Add a new product

var newProduct = new Product

{

Name = "Wireless Mouse",

Category = "Electronics",

Quantity = 25,

Price = 15.99m

};

db.Products.Add(newProduct);

db.SaveChanges();

Console.WriteLine("Added: Wireless Mouse");

// List all products

var products = db.Products.ToList();

Console.WriteLine("\nAll Products:");

foreach (var p in products)

{

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

}

// Update quantity of a product

var mouse = db.Products.FirstOrDefault(p => p.Name == "Wireless Mouse");

if (mouse != null)

{

mouse.Quantity += 5;

db.SaveChanges();

Console.WriteLine("\nUpdated Quantity of Wireless Mouse");

}

// Delete a product

var toDelete = db.Products.FirstOrDefault(p => p.Name == "Wireless Mouse");

if (toDelete != null)

{

db.Products.Remove(toDelete);

db.SaveChanges();

Console.WriteLine("Deleted: Wireless Mouse");

}

}

}

}

}

Output:  
