**Lab1: Understanding ORM with a Retail Inventory System**

**Product.cs:**

namespace RetailInventory.Models

{

public class Product

{

public int ProductId { get; set; }

public string Name { get; set; }

public int StockLevel { get; set; }

public int CategoryId { get; set; }

public Category Category { get; set; }

}

}

**Category.cs:**

namespace RetailInventory.Models

{

public class Category

{

public int CategoryId { get; set; }

public string Name { get; set; }

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

}

}

**AppDbContext.cs:**

using Microsoft.EntityFrameworkCore;

using RetailInventory.Models;

public class AppDbContext : DbContext

{

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

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

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

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

}

}

**Program.cs:**

using Microsoft.EntityFrameworkCore;

using RetailInventory.Models;

using var context = new AppDbContext();

context.Database.EnsureCreated();

if (!context.Categories.Any())

{

var category = new Category { Name = "Clothing" };

category.Products.Add(new Product { Name = "T-Shirt", StockLevel = 100 });

category.Products.Add(new Product { Name = "Jeans", StockLevel = 50 });

context.Categories.Add(category);

context.SaveChanges();

}

var products = context.Products.Include(p => p.Category).ToList();

foreach (var product in products)

{

Console.WriteLine($"{product.Name} ({product.Category.Name}) - {product.StockLevel} in stock");

}

**Output:**

