# EF CORE LAB 5

THE CODE:

using System;

using System.Collections.Generic;

using System.Threading.Tasks;

using Microsoft.EntityFrameworkCore;

namespace RetailInventorySystem

{

public class Category

{

public int Id { get; set; }

public string Name { get; set; }

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

}

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=RetailInventoryDB;Trusted\_Connection=True;");

}

}

class Program

{

static async Task Main(string[] args)

{

using var context = new AppDbContext();

await context.Database.EnsureDeletedAsync();

await context.Database.EnsureCreatedAsync();

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(" Initial data inserted.\n");

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

Console.WriteLine(" All Products:");

foreach (var p in products)

{

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

}

var productById = await context.Products.FindAsync(1);

if (productById != null)

{

Console.WriteLine($"\n Product with ID 1: {productById.Name}");

}

else

{

Console.WriteLine("\n Product with ID 1 not found.");

}

var expensiveProduct = await context.Products.FirstOrDefaultAsync(p => p.Price > 50000);

if (expensiveProduct != null)

{

Console.WriteLine($"\n Expensive Product: {expensiveProduct.Name} - ₹{expensiveProduct.Price}");

}

else

{

Console.WriteLine("\n No expensive products found.");

}

}

}

}