**WEEK 3**

**Entity Framework**

**-Aritra Das**

**Superset ID: 6362294**

**LAB – 1: Understanding ORM with a Retail Inventory System**

**Code –**

**Filename – Program.cs**

using System;

using System.Collections.Generic;

class Program

{

static void Main()

{

var category = new Category

{

CategoryId = 1,

Name = "Electronics",

Products = new List<Product>()

};

var supplier = new Supplier

{

SupplierId = 1,

Name = "Samsung",

Products = new List<Product>()

};

var product = new Product

{

ProductId = 1,

Name = "Smart TV",

Quantity = 10,

Category = category,

CategoryId = category.CategoryId,

Supplier = supplier,

SupplierId = supplier.SupplierId

};

category.Products.Add(product);

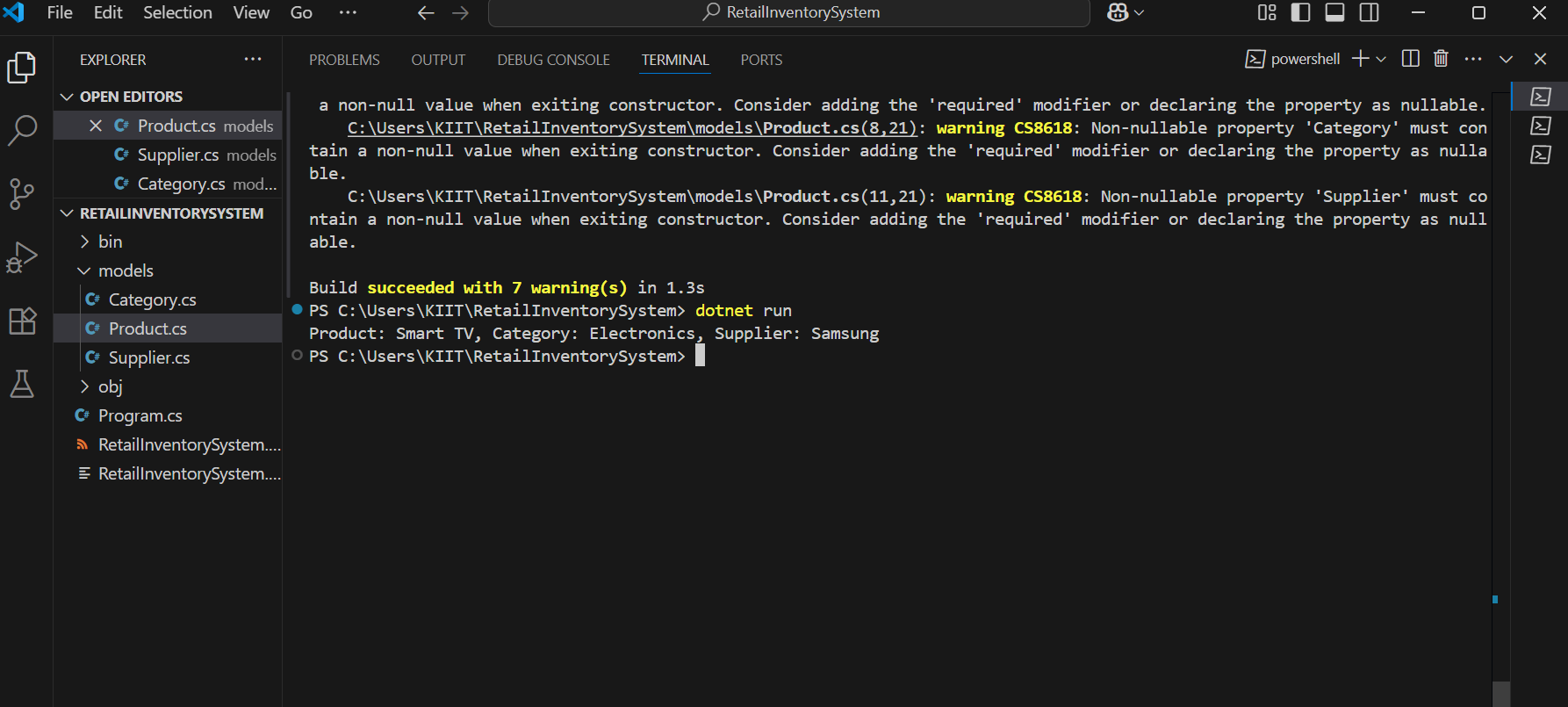
supplier.Products.Add(product);

Console.WriteLine($"Product: {product.Name}, Category: {product.Category.Name}, Supplier: {product.Supplier.Name}");

}

}

**Output –**



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

**Code –**

**Filename – AppDbContext.cs**

using Microsoft.EntityFrameworkCore;

public class AppDbContext : DbContext

{

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

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

public DbSet<Supplier> Suppliers { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

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

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Category>()

.HasMany(c => c.Products)

.WithOne(p => p.Category)

.HasForeignKey(p => p.CategoryId);

modelBuilder.Entity<Supplier>()

.HasMany(s => s.Products)

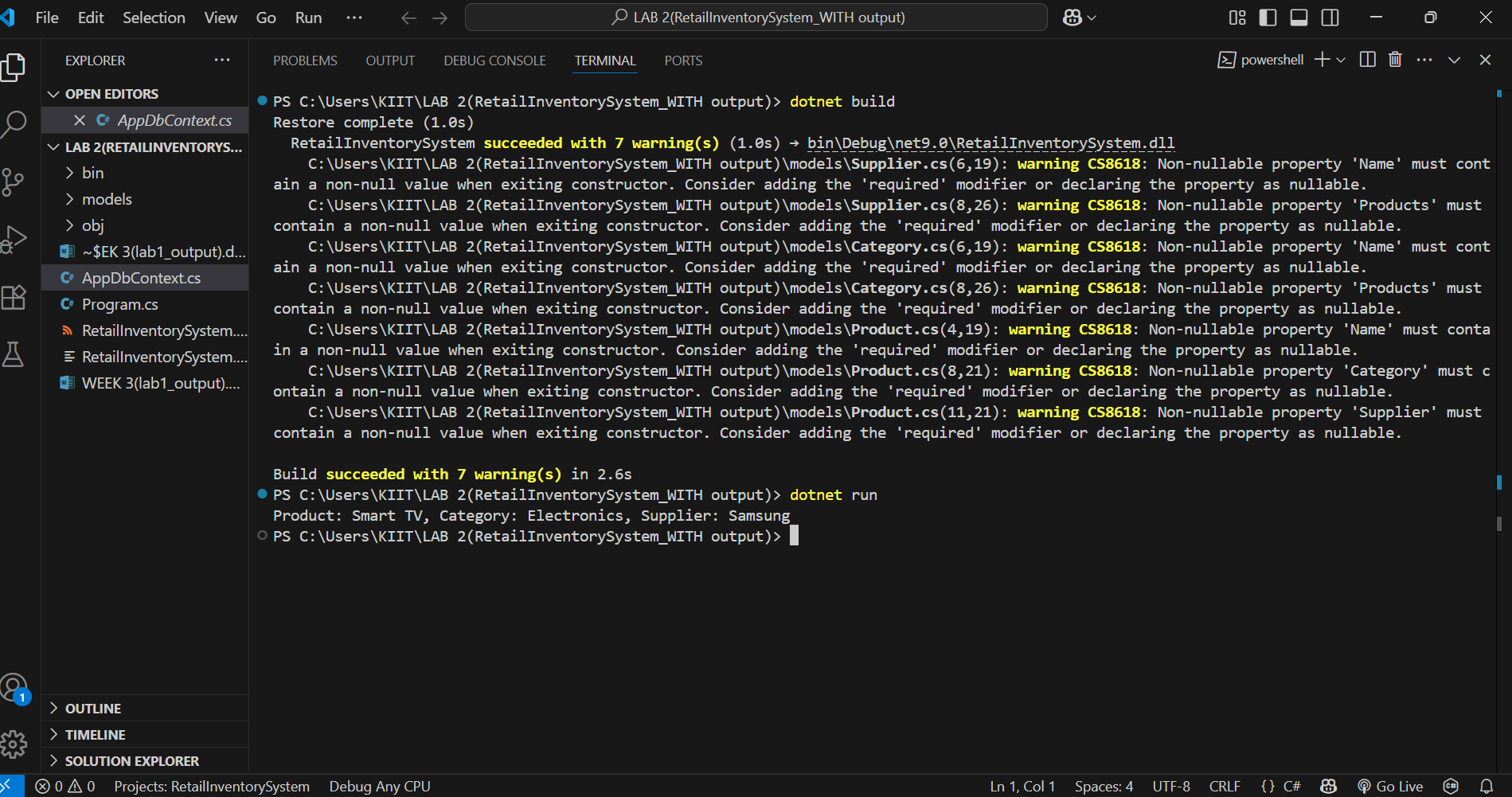
.WithOne(p => p.Supplier)

.HasForeignKey(p => p.SupplierId);

}

}

**Output -**



**LAB – 3: Using EF Core CLI to create and apply Migrations**

**Code –**

**Under Migrations :**

**Filename – initialCreate.cs**

using Microsoft.EntityFrameworkCore.Migrations;

#nullable disable

namespace RetailInventorySystem.Migrations

{

/// <inheritdoc />

public partial class InitialCreate : Migration

{

/// <inheritdoc />

protected override void Up(MigrationBuilder migrationBuilder)

{

migrationBuilder.CreateTable(

name: "Categories",

columns: table => new

{

CategoryId = table.Column<int>(type: "INTEGER", nullable: false)

.Annotation("Sqlite:Autoincrement", true),

Name = table.Column<string>(type: "TEXT", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Categories", x => x.CategoryId);

});

migrationBuilder.CreateTable(

name: "Suppliers",

columns: table => new

{

SupplierId = table.Column<int>(type: "INTEGER", nullable: false)

.Annotation("Sqlite:Autoincrement", true),

Name = table.Column<string>(type: "TEXT", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Suppliers", x => x.SupplierId);

});

migrationBuilder.CreateTable(

name: "Products",

columns: table => new

{

ProductId = table.Column<int>(type: "INTEGER", nullable: false)

.Annotation("Sqlite:Autoincrement", true),

Name = table.Column<string>(type: "TEXT", nullable: false),

Quantity = table.Column<int>(type: "INTEGER", nullable: false),

CategoryId = table.Column<int>(type: "INTEGER", nullable: false),

SupplierId = table.Column<int>(type: "INTEGER", nullable: false)

},

constraints: table =>

{

table.PrimaryKey("PK\_Products", x => x.ProductId);

table.ForeignKey(

name: "FK\_Products\_Categories\_CategoryId",

column: x => x.CategoryId,

principalTable: "Categories",

principalColumn: "CategoryId",

onDelete: ReferentialAction.Cascade);

table.ForeignKey(

name: "FK\_Products\_Suppliers\_SupplierId",

column: x => x.SupplierId,

principalTable: "Suppliers",

principalColumn: "SupplierId",

onDelete: ReferentialAction.Cascade);

});

migrationBuilder.CreateIndex(

name: "IX\_Products\_CategoryId",

table: "Products",

column: "CategoryId");

migrationBuilder.CreateIndex(

name: "IX\_Products\_SupplierId",

table: "Products",

column: "SupplierId");

}

/// <inheritdoc />

protected override void Down(MigrationBuilder migrationBuilder)

{

migrationBuilder.DropTable(

name: "Products");

migrationBuilder.DropTable(

name: "Categories");

migrationBuilder.DropTable(

name: "Suppliers");

}

}

}

**Filename – AppDbModel.cs**

using Microsoft.EntityFrameworkCore;

using Microsoft.EntityFrameworkCore.Infrastructure;

using Microsoft.EntityFrameworkCore.Storage.ValueConversion;

#nullable disable

namespace RetailInventorySystem.Migrations

{

[DbContext(typeof(AppDbContext))]

partial class AppDbContextModelSnapshot : ModelSnapshot

{

protected override void BuildModel(ModelBuilder modelBuilder)

{

#pragma warning disable 612, 618

modelBuilder.HasAnnotation("ProductVersion", "9.0.6");

modelBuilder.Entity("Category", b =>

{

b.Property<int>("CategoryId")

.ValueGeneratedOnAdd()

.HasColumnType("INTEGER");

b.Property<string>("Name")

.IsRequired()

.HasColumnType("TEXT");

b.HasKey("CategoryId");

b.ToTable("Categories");

});

modelBuilder.Entity("Product", b =>

{

b.Property<int>("ProductId")

.ValueGeneratedOnAdd()

.HasColumnType("INTEGER");

b.Property<int>("CategoryId")

.HasColumnType("INTEGER");

b.Property<string>("Name")

.IsRequired()

.HasColumnType("TEXT");

b.Property<int>("Quantity")

.HasColumnType("INTEGER");

b.Property<int>("SupplierId")

.HasColumnType("INTEGER");

b.HasKey("ProductId");

b.HasIndex("CategoryId");

b.HasIndex("SupplierId");

b.ToTable("Products");

});

modelBuilder.Entity("Supplier", b =>

{

b.Property<int>("SupplierId")

.ValueGeneratedOnAdd()

.HasColumnType("INTEGER");

b.Property<string>("Name")

.IsRequired()

.HasColumnType("TEXT");

b.HasKey("SupplierId");

b.ToTable("Suppliers");

});

modelBuilder.Entity("Product", b =>

{

b.HasOne("Category", "Category")

.WithMany("Products")

.HasForeignKey("CategoryId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.HasOne("Supplier", "Supplier")

.WithMany("Products")

.HasForeignKey("SupplierId")

.OnDelete(DeleteBehavior.Cascade)

.IsRequired();

b.Navigation("Category");

b.Navigation("Supplier");

});

modelBuilder.Entity("Category", b =>

{

b.Navigation("Products");

});

modelBuilder.Entity("Supplier", b =>

{

b.Navigation("Products");

});

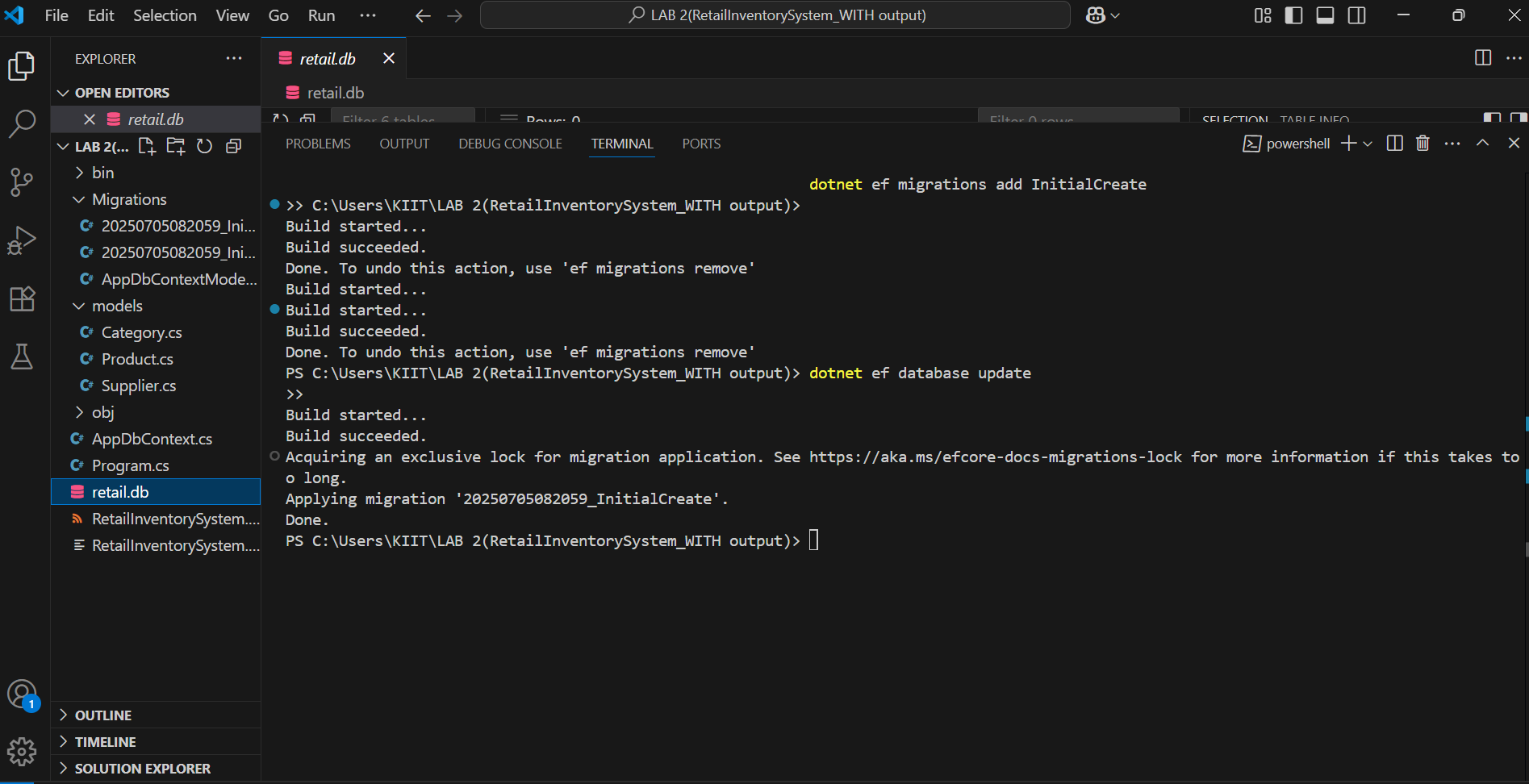
#pragma warning restore 612, 618

}

}

}

**Output –**



**LAB – 4: Inserting initial data into the Database**

**Code –**

**Filename – Program.cs**

using System;

using System.Threading.Tasks;

using RetailInventorySystem.Models;

using Microsoft.EntityFrameworkCore;

class Program

{

static async Task Main(string[] args)

{

using var context = new AppDbContext();

await context.Database.EnsureCreatedAsync();

context.Products.RemoveRange(context.Products);

context.Categories.RemoveRange(context.Categories);

await context.SaveChangesAsync();

var electronics = new Category { Name = "Electronics" };

var groceries = new Category { Name = "Groceries" };

await context.Categories.AddRangeAsync(electronics, groceries);

var laptop = new Product { Name = "Laptop", Price = 75000, Category = electronics };

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

await context.Products.AddRangeAsync(laptop, rice);

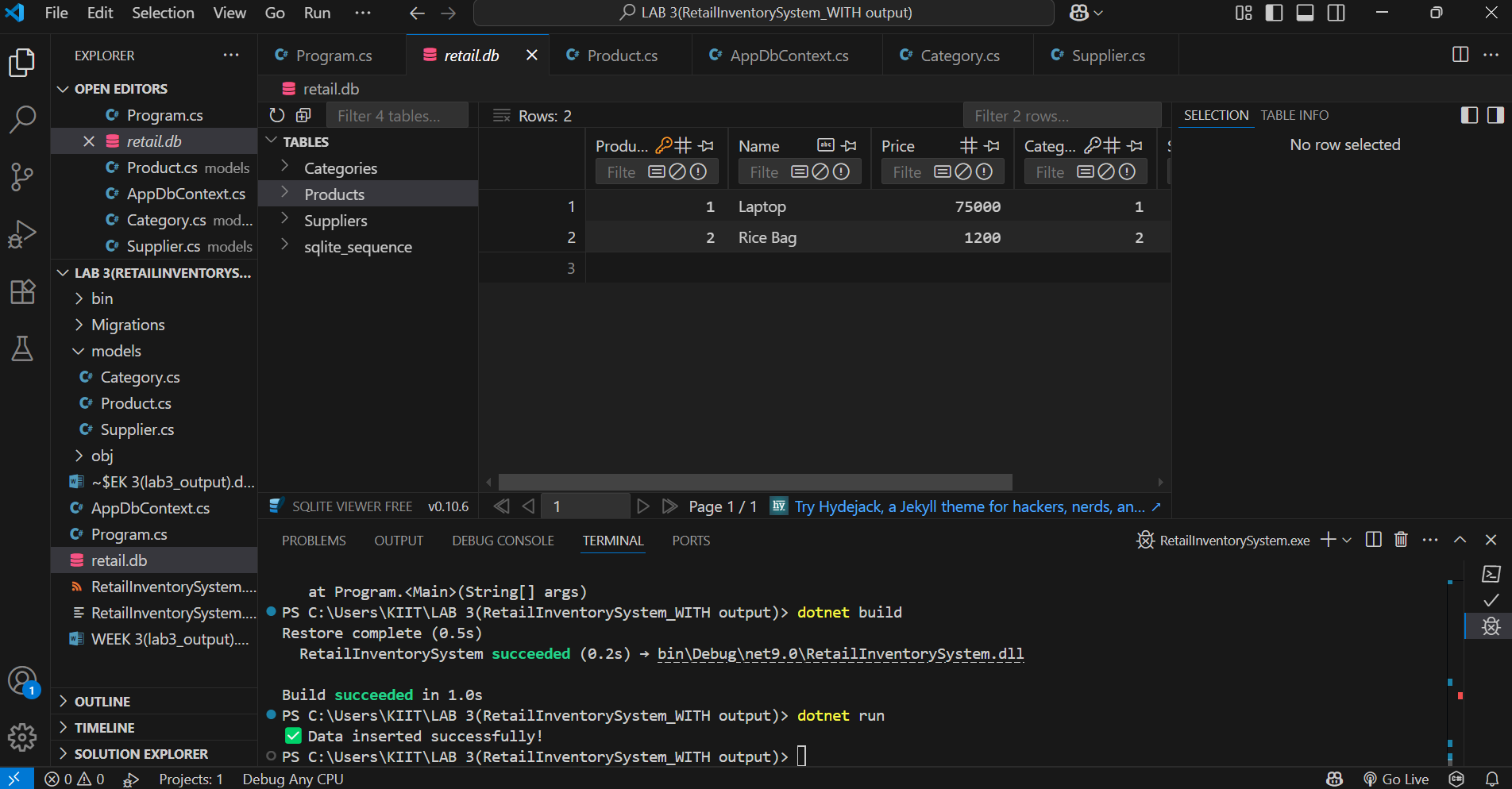
await context.SaveChangesAsync();

Console.WriteLine("Data inserted successfully!");

}

}

**Output –**



**LAB – 5: Retrieving data from the Database**

**Code –**

**Filename – Program.cs**

using System;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.EntityFrameworkCore;

using RetailInventorySystem.Models;

class Program

{

static async Task Main(string[] args)

{

using var context = new AppDbContext();

Console.WriteLine("Products in Inventory:\n");

var products = await context.Products

.Include(p => p.Category)

.Include(p => p.Supplier)

.ToListAsync();

foreach (var product in products)

{

string categoryName = product.Category?.Name ?? "No Category";

string supplierName = product.Supplier?.Name ?? "No Supplier";

Console.WriteLine($"Product: {product.Name}");

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

Console.WriteLine($"Category: {categoryName}");

Console.WriteLine($"Supplier: {supplierName}");

Console.WriteLine();

}

Console.WriteLine($"Total Products: {products.Count}");

}

}

**Output –**

