**Case Study: Online Shopping Cart**

**🔍 Project Overview**

Design and develop an **Online Shopping Cart System** using **ASP.NET Core MVC** (for frontend) and **ASP.NET Core Web API** (for backend). The application allows users to browse products, add them to a shopping cart, register/login, and place orders.

**🎯 Objectives**

* Implement full-stack web development using ASP.NET Core MVC and Web API.
* Apply CRUD operations via API for products and cart.
* Learn authentication and authorization with JWT Token.
* Use ADO.NET for database access with proper design patterns.
* Host API and MVC separately to simulate real-world API scenarios.

**📦 Major Functional Modules**

| **Module** | **Description** |
| --- | --- |
| User Registration/Login | Allow customers to register, login. |
| Product Catalog | Show a list of available products with details. |
| Shopping Cart | Enable users to add/remove/update products in the cart. |
| Order Placement | Checkout functionality with order summary, billing, and confirmation. |
| Admin Panel (optional) | Manage products, categories, and view customer orders. |

**🛠️ Technology Stack**

| **Layer** | **Technology** |
| --- | --- |
| Frontend (UI) | ASP.NET Core MVC |
| Backend (API) | ASP.NET Core Web API |
| Database | SQL Server + ADO.NET |
| Authentication | ASP.NET Core JWT Bearer Token |
|  |  |

**🗃️ Suggested Folder Structure**

**✅ ASP.NET Core Web API (Backend)**

ShoppingCartAPI/

│

├── Controllers/

│ ├── ProductsController.cs

│ ├── UsersController.cs

│ ├── CartController.cs

│ └── OrdersController.cs

│

├── Models/

├── DTOs/

├── Services/

├── Data/

│ └── ApplicationDbContext.cs

├── Program.cs

└── appsettings.json

**✅ ASP.NET Core MVC (Frontend)**

CopyEdit

ShoppingCartWeb/

│

├── Controllers/

│ ├── HomeController.cs

│ ├── AccountController.cs

│ ├── CartController.cs

│ └── CheckoutController.cs

│

├── Views/

│ ├── Home/

│ ├── Account/

│ ├── Cart/

│ └── Checkout/

│

├── Models/

├── Services/

├── wwwroot/

└── Program.cs

**🧩 Database Design (ER Diagram Highlights)**

**Entities**

1. **Users**
   * UserId (PK)
   * FullName
   * Email
   * Password
   * Role (Customer/Admin)
2. **Products**
   * ProductId (PK)
   * Name
   * Description
   * Price
   * Stock
   * CategoryId (FK)
3. **Categories**
   * CategoryId (PK)
   * CategoryName
4. **CartItems**
   * CartItemId (PK)
   * UserId (FK)
   * ProductId (FK)
   * Quantity
5. **Orders**
   * OrderId (PK)
   * UserId (FK)
   * OrderDate
   * TotalAmount
6. **OrderItems**
   * OrderItemId (PK)
   * OrderId (FK)
   * ProductId (FK)
   * Quantity
   * UnitPrice

**🧪 Milestones & Tasks (Practice)**

**🔹 Week 1: Environment Setup & Backend API**

* Setup ASP.NET Core Web API project.
* Create database and Tables.
* Implement CRUD APIs for Cart and CartItems.
* Implement CRUD APIs for Orders and OrderItems.
* Implement CRUD APIs for Products and Categories (optional if time permits).
* Secure APIs using JWT.

**🔹 Week 2: MVC Frontend & API Integration**

* Setup ASP.NET Core MVC project.
* Build Product Catalog UI (consume Web API using HttpClient).
* Implement registration/login via JWT.

**🔹 Week 3: Cart Functionality**

* Create cart controller and views.
* Add/remove/update cart items via API.
* Fetch product prices dynamically from API.

**🔹 Week 4: Checkout & Order Processing**

* Add checkout flow (address, review, confirm).
* Save order in the database via API.
* Show order summary and user past orders.

**📘 Learning Outcomes**

* Clear understanding of ASP.NET Core MVC vs Web API roles.
* API integration with front-end via HttpClient.
* Real-world application of layered architecture and ADO.NET.
* Role-based authentication with JWT.
* Modular design and reusability using services and DTOs.