

# **Project Documentation: E-Commerce Website**

## **Table of contents:**

1. Introduction
2. Project Overview
3. Requirements
4. System Architecture
5. Database Design
6. User Interface Design
7. APIs and Integrations
8. Conclusion

**Introduction:** The purpose of this document is to provide comprehensive and detailed documentation for the e-commerce website project. This documentation covers all aspects of the e-commerce website project, including functional and non-functional requirements, system architecture, database design, user interface design, API integrations, Conclusion. It aims to provide a clear and detailed understanding of the project's objectives, functionalities, and technical specifications.

**Project Overview:** The e-commerce website is designed to facilitate online shopping for users while providing robust management tools for administrators. The system employs role-based authentication and authorization, ensuring that users and administrators have appropriate access to the website's features.

## **Features:**

### **1. Admin Features**

- **Product Management:** Admins can add, delete, edit, and search for products. They can also perform category-wise searches using AJAX, add new categories, and manage product attributes like colours and sizes through mapper tables in the database. The system notifies admins when product quantities drop below a specified threshold.
- **Coupon Management:** Admins can manage coupons by adding new coupons with details such as name, percentage discount, expiry date, and status (Active, Inactive, Expired). The system automatically updates the coupon status to 'Expired' once the expiry date is reached.
- **Order Management:** Admins can view all orders, access detailed information on specific orders, and update the status of orders (Placed, Shipped, Delivered).
- **Inventory Management:** The inventory management module allows admins to Check low stock products and add quantities to those products and Search order history within a specified date range and view total sales during that period.

## **2. User Features**

- **Product Browsing:** Users can browse all products, search by category, and view detailed information on each product.
- **Order Management:** Users can view their order history, check details of ongoing orders, and review specific order details.
- **Shopping Cart and Orders:** Users can add multiple products to their cart, choose specific sizes and colours, and place orders. During checkout, users can apply valid coupons to receive discounts. Upon placing an order, the system adjusts product quantities accordingly.
- **User Accounts:** Users can create accounts or shop as guests, providing flexibility in how they interact with the site. During the order process, users must provide a phone number for contact purposes.

### **Definitions, Acronyms, and Abbreviations:**

To ensure clarity and avoid misunderstandings, the following definitions, acronyms, and abbreviations are used throughout this document:

API: Application Programming Interface

DB: Database

UI/UX: User Interface/User Experience

CRUD: Create, Read, Update, Delete

HTTPS: HyperText Transfer Protocol Secure

### **Requirements:**

#### **1. User Management**

##### **1.1 User Registration**

Users must be able to register with an email address, password, and other necessary personal details.

The system should validate the email address to ensure it is unique.

## **1.2 User Authentication**

The system must allow users to log in using their registered email and password.

Implement role-based authentication to distinguish between admin and regular user roles.

## **1.3 Guest Checkout**

Allow users to checkout as a guest without the need to create an account.

Guest users must provide a valid phone number during the checkout process.

## **2. Product Management**

### **2.1 Add/Edit/Delete Products**

Admins must be able to add, edit, and delete products from the inventory.

Each product must have attributes such as name, description, price, category, quantity, colours, and sizes.

### **2.2 Category Management**

Admins can create, edit, and delete product categories.

Products can be associated with multiple categories.

### **2.3 Product Variants**

Allow products to have multiple variants, such as different sizes and colors.

Use mapper tables in the database to manage product variants.

### **2.4 Low Stock Notification**

The system should notify admins when product quantities fall below a specified threshold (e.g., less than 3 units).

### **2.5 Search and Filter**

Users must be able to search for products by category.

Admins can perform category-wise searches using AJAX for efficient data retrieval.

## **3. Coupon Management**

### **3.1 Add/Edit/Delete Coupons**

Admins can create, edit, and delete coupons.

Each coupon must have attributes such as name, discount percentage, expiry date, and status (Active, Inactive, Expired).

### **3.2 Automatic Expiry**

The system should automatically set the status of a coupon to "Expired" once the expiry date is reached.

### **3.3 Apply Coupons**

Users can apply valid coupons during checkout to receive discounts on their order total.

## **4. Order Management**

### **4.1 Place Order**

Users must be able to place orders for single or multiple products.

The system should update product quantities upon order placement.

### **4.2 Order History**

Users can view their past orders and the details of each order.

Admins can view all orders and access detailed information for each order.

### **4.3 Order Status Management**

Admins can update the status of orders (Placed, Shipped, Delivered).

Users can view the current status of their orders.

## **5. Inventory Management**

### **5.1 Low Stock Products**

Admins can view a list of products with quantities below the specified threshold.

Admins can add more stock to low inventory products.

### **5.2 Order History Search**

Admins can search for orders within a specific date range.

The system should display the total sales amount for the selected period.

# Technical Requirements

## 1. Technology Stack

**Frontend:** HTML, CSS, JavaScript (with AJAX for dynamic content updates), Bootstrap for responsive design.

**Backend:** ASP.NET MVC, C#.

**Database:** SQL Server or any relational database supporting Entity Framework.

**Version Control:** Git.

## System Architecture:

Overview: The system architecture of the e-commerce website is designed to ensure scalability, security, and maintainability. The architecture leverages a multi-tier approach, separating the application into distinct layers that handle different aspects of the application. This design enables better management of the codebase, facilitates team collaboration, and enhances the overall performance and reliability of the system.

Layers:

The architecture is divided into the following layers:

- Application Layer (Frontend)
- Business logic Layer (Backend)
- Data Layer (Database)

### 1. Application Layer (Frontend)

The presentation layer is responsible for the user interface and user experience. It is implemented using HTML, CSS, JavaScript, and Bootstrap to ensure responsive design and compatibility across various devices.

Bootstrap to ensure responsive design and compatibility across various devices.

Technologies: HTML, CSS, JavaScript, jQuery, Bootstrap

Components:

3. Login and Registration Pages
4. Admin Dashboard
5. Product Management Interface
6. Category Management Interface
7. Coupon Management Interface
8. Order Management Interface
9. Inventory Management Interface
10. User Shopping Experience (Product Browsing, Cart, Checkout)

## 2. Business logic Layer (Backend)

This layer contains the business logic and manages communication between the Application layer and the data layer. It is built using ASP.NET MVC and C#.

Technologies: ASP.NET MVC, C#

Components:

Controllers: Handle HTTP requests and responses.

Models: Represent the data structures.

Views: Render the user interface.

Services: Contain business logic and interact with the data layer.

Authentication and Authorization: Role-based access control to differentiate between admin and user functionalities.

## 3. Data Layer (Database)

The data layer handles the storage and retrieval of data. It uses a relational database to manage various entities such as products, categories, orders, users, and coupons.

Technologies: SQL Server or any relational database compatible with Entity Framework

Components:

Database Schema: Tables for Categories, Colors, Coupons, Orders, Products, ProductColor, ProductSize, Sizes, Users.

Entity Framework: ORM to interact with the database.

## **Database Structure:**

The database structure for the e-commerce web application consists of several tables to handle different aspects of the system, such as product management, user management, order processing, and coupon management. Below is the detailed database schema.

## **Tables and Relationships**

### **1. Category Table**

Id: int (Primary Key)

Name: string

Relationships:

One-to-Many with Product (One category can have multiple products).

### **2. Color Table**

Id: int (Primary Key)

Name: string

Relationships:

Many-to-Many with Product through ProductColor.

Many-to-Many with Order through ProductColorOrderMapper.

### **3. Cupon Table**

Id: int (Primary Key)

Name: string

Percentage: int (Nullable)

Expire\_date: datetime (Nullable)

Status: string



#### **4. Order Table**

Id: int (Primary Key)

Date: datetime (Nullable)

UserId: int (Nullable, Foreign Key to User)

Status: string

TotalOrderPrice: string

Number: string

Quantity: string

Relationships:

Many-to-Many with Product through ProductOrderMapper.

Many-to-Many with Color through ProductColorOrderMapper.

Many-to-Many with Size through ProductOrderSizeMapper.

#### **5. Product Table**

Id: int (Primary Key)

Name: string

price: int (Nullable)

Description: string

Quantity: int (Nullable)

Image: string

Category\_Id: int (Nullable, Foreign Key to Category)

Relationships:

Many-to-One with Category (Many products can belong to one category).

Many-to-Many with Order through ProductOrderMapper.

Many-to-Many with Color through ProductColor.

Many-to-Many with Size through ProductSize.

## **6. ProductColor Table**

id: int (Primary Key)

ProductId: int (Foreign Key to Product)

ColorId: int (Foreign Key to Color)

Relationships:

Many-to-One with Product (Many product colors can belong to one product).

Many-to-One with Color (Many product colors can belong to one color).

## **7. ProductColorOrderMapper Table**

Id: int (Primary Key)

ProductId: int (Foreign Key to Product)

OrderId: int (Foreign Key to Order)

ColorId: int (Foreign Key to Color)

Relationships:

Many-to-One with Product (Many product color orders can belong to one product).

Many-to-One with Order (Many product color orders can belong to one order).

Many-to-One with Color (Many product color orders can belong to one color).

## **8. ProductOrderMapper Table**

Id: int (Primary Key)

ProductId: int (Foreign Key to Product)

OrderId: int (Foreign Key to Order)

Relationships:

Many-to-One with Product (Many product orders can belong to one product).

Many-to-One with Order (Many product orders can belong to one order).

## **9. ProductOrderSizeMapper Table**

Id: int (Primary Key)

ProductId: int (Foreign Key to Product)

SizeId: int (Foreign Key to Size)

OrderId: int (Foreign Key to Order)

Relationships:

Many-to-One with Product (Many product size orders can belong to one product).

Many-to-One with Size (Many product size orders can belong to one size).

Many-to-One with Order (Many product size orders can belong to one order).

## **10. ProductSize Table**

Id: int (Primary Key)

ProductId: int (Foreign Key to Product)

SizeId: int (Foreign Key to Size)

Relationships:

Many-to-One with Product (Many product sizes can belong to one product).

Many-to-One with Size (Many product sizes can belong to one size).

## **11. Size Table**

id: int (Primary Key)

sizeOfProduct: int (Nullable)

Relationships:

Many-to-Many with Product through ProductSize.

Many-to-Many with Order through ProductOrderSizeMapper.

## **12. User Table**

Id: int (Primary Key)

Name: string

Email: string

Password: string

Gender: string

User\_Type: string

Relationships:

One-to-Many with Order (One user can have multiple orders).

## **User Interface Design**

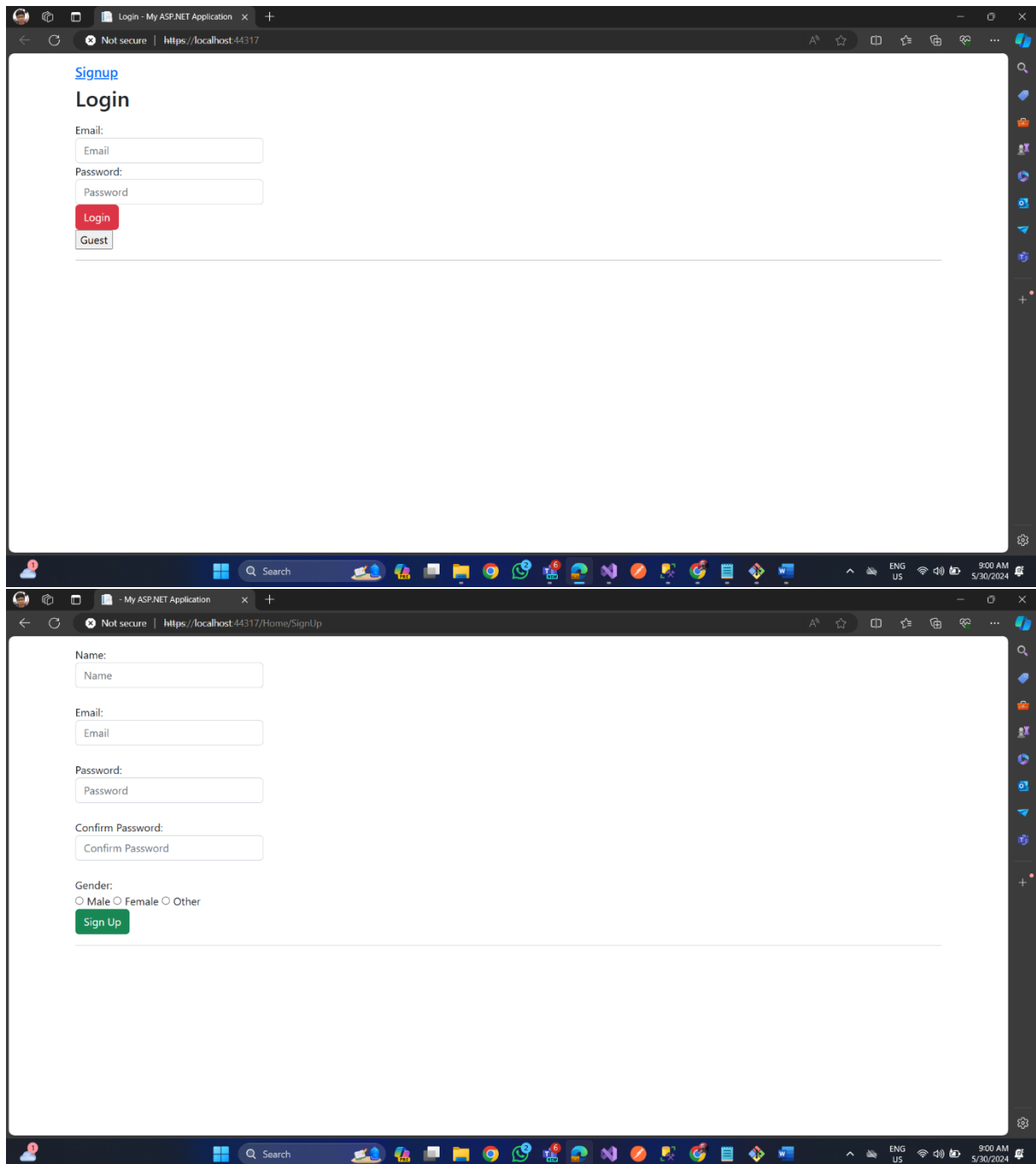
The user interface (UI) design for the e-commerce web application focuses on providing a seamless and intuitive experience for both the admin and the end-users. The design will be responsive, ensuring usability across various devices such as desktops, tablets, and smartphones. Below is a detailed description of the UI design for different modules.

### **1. Login and Registration Pages**

Login Page

Components: Email Input Field, Password Input Field, Login Button, Registration Link, Registration Page

Components: Name Input Field, Email Input Field, Password Input Field, Confirm Password Input Field, Gender Selection (Radio Buttons), Sign Up Button, Login Link

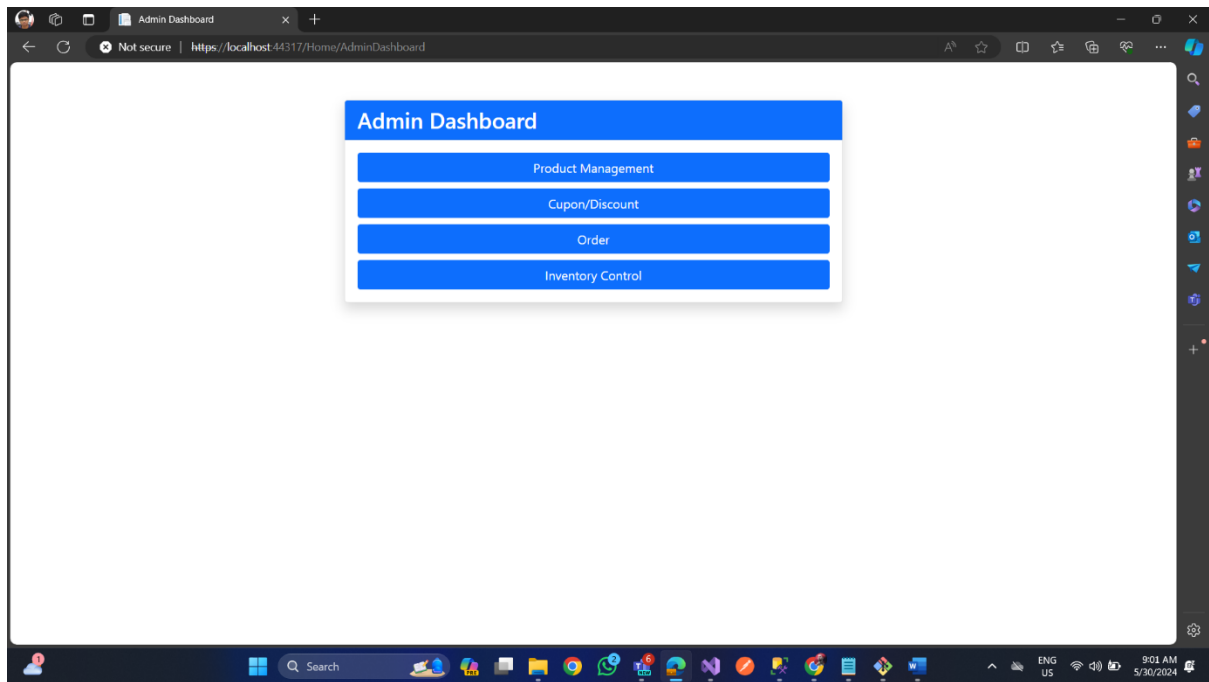


## 2. Admin Dashboard

Components:

Dashboard Link, Product Management Link, Category Management Link, Coupon Management Link, Orders Management Link, Inventory Management Link

Components: Overview of Key Metrics (Total Sales, Total Orders, Low Stock Alerts), Recent Orders List, Low Stock Products List



### 3. Product Management

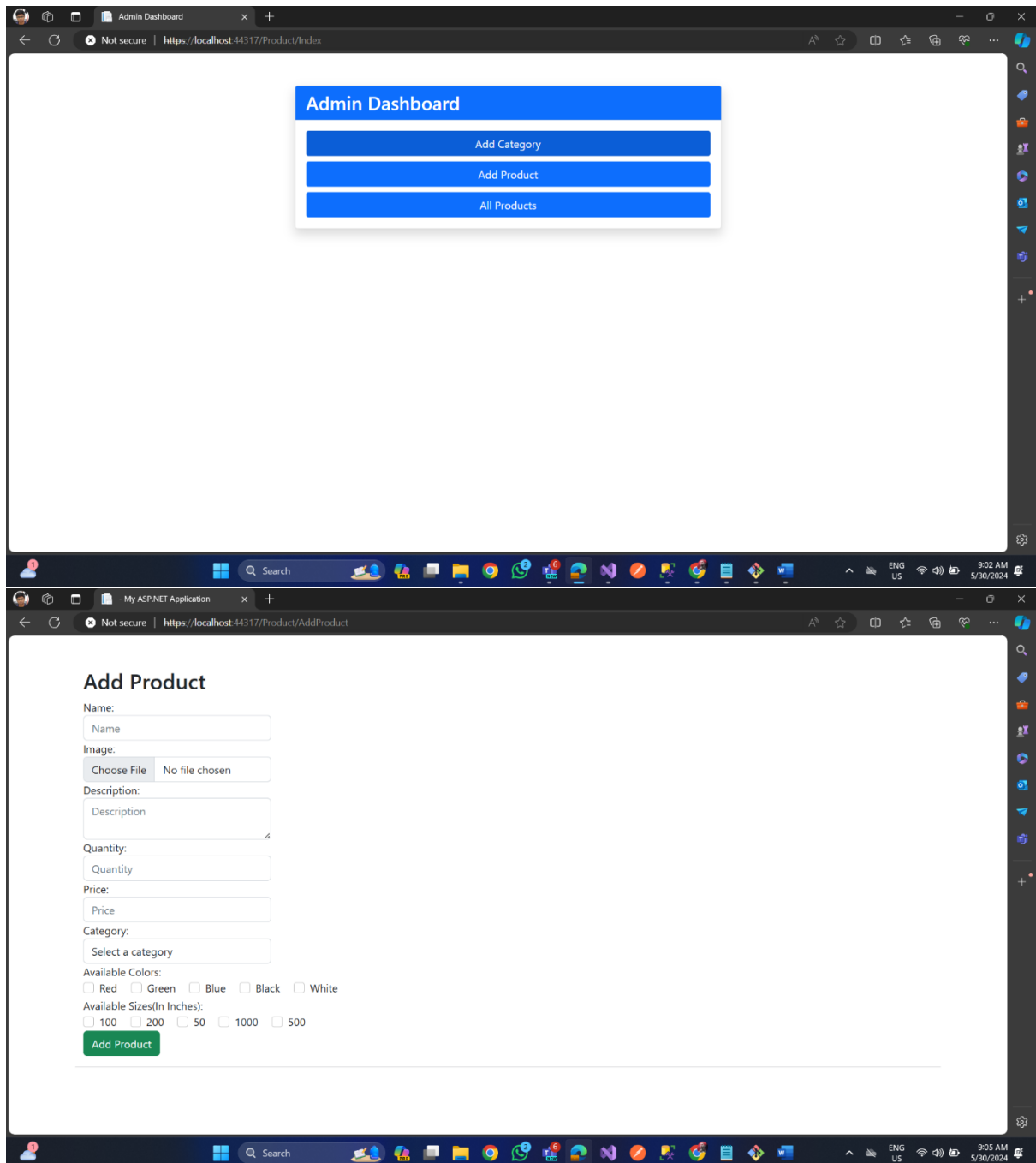
#### Product List Page

##### Components:

Search Bar with AJAX for Category-wise Search, Add New Product Button, Product Table (Columns: Name, Category, Price, Quantity, Actions), Edit and Delete Buttons for Each Product, Add/Edit Product Page

##### Components:

Product Name Input Field, Category Dropdown, Description Text, Price Input Field, Quantity Input Field, Image Upload Field, Color Selection (Multiple Select), Size Selection (Multiple Select), Save Button



Product Details - My ASP.NET A...

Not secure | https://localhost:44317/Product/Search


## Product Details

### Category-wise Search


Select Category:

All Categories

#### Product Details

Name	watch
Description	asvf
Quantity	2 (Low Stock)
Price	200
Category	Electronic
Color	Red Green
Size	100 1000
Image	

#### Product Details

Name	watch
Description	this is awesome
Quantity	2 (Low Stock)
Price	1500
Category	Gagets
Color	Red Blue
Size	100 50
Image	

EditDelete

My ASP.NET Application

Not secure | https://localhost:44317/Product/UpdateProduct/23

## Update Product

Name:  
watch

Image:  
Choose File No file chosen

Description:  
this is awesome

Quantity:  
2

Price:  
1500

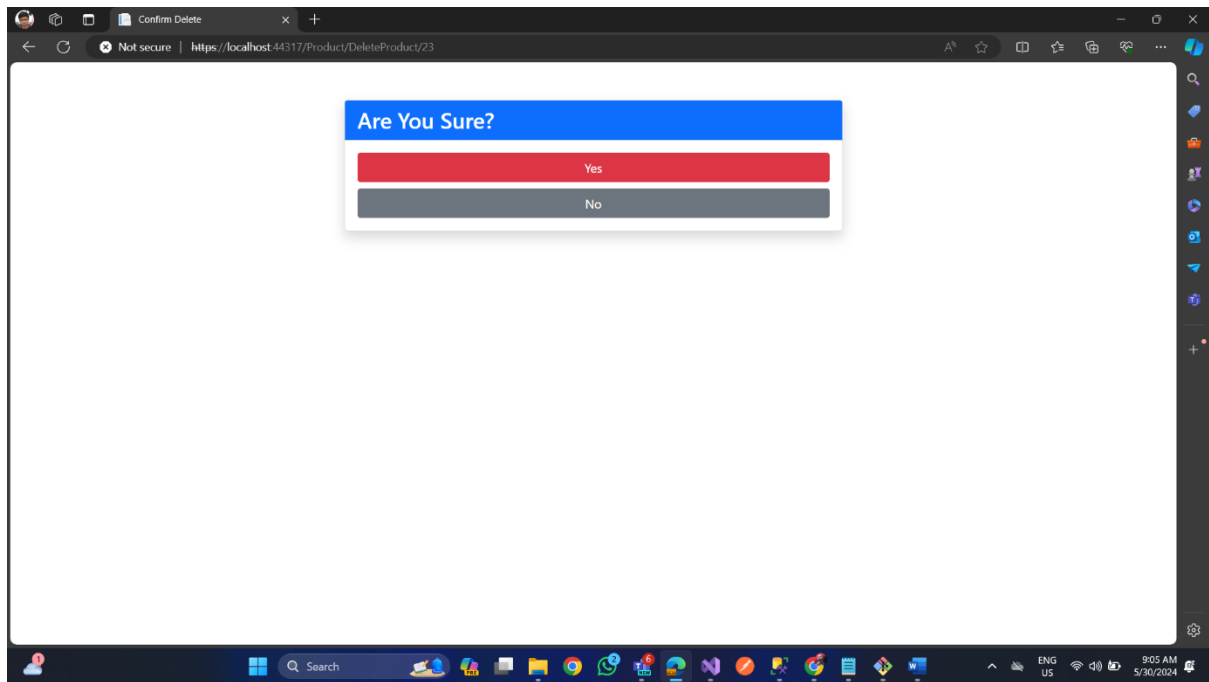
Category:  
Select a category

Available Colors:  
☐ Red ☐ Green ☐ Blue ☐ Black ☐ White

Available Sizes (In Inches):  
☐ 100 ☐ 200 ☐ 50 ☐ 1000 ☐ 500

Update Product





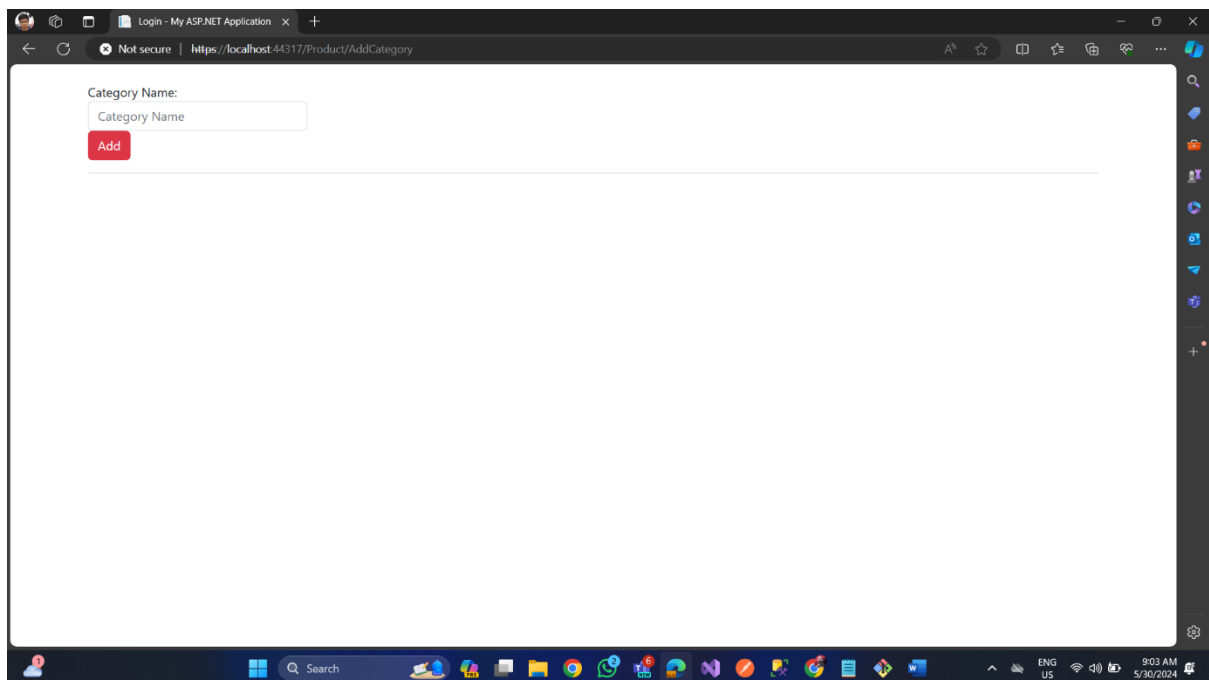
## 4. Category Management

Category List Page

Components:

Add New Category Button, Category Table (Columns: Name)

Components: Category Name Input Field, Save Button



## 5. Coupon Management

Coupon List Page

Components: Add New Coupon Button, Coupon Table (Columns: Name, Percentage, Expiry Date, Status, Actions), Edit and Delete Buttons for Each Coupon, Add/Edit Coupon Page

Components: Coupon Name Input Field, Percentage Input Field, Expiry Date Picker, Status Dropdown (Active, Inactive, Expired), Save Button

The screenshot displays two pages of a web application. The top page, titled 'Add Cupon', contains a form for adding a new coupon. The bottom page, titled 'Available Coupons', displays a table of existing coupons with columns for Coupon Id, Coupon Name, Coupon Percentage, Status, and Actions (Edit, Delete).

**Add Cupon Page:**

Cupon Details

Cupon Name:

Discount:

Expiration Date:

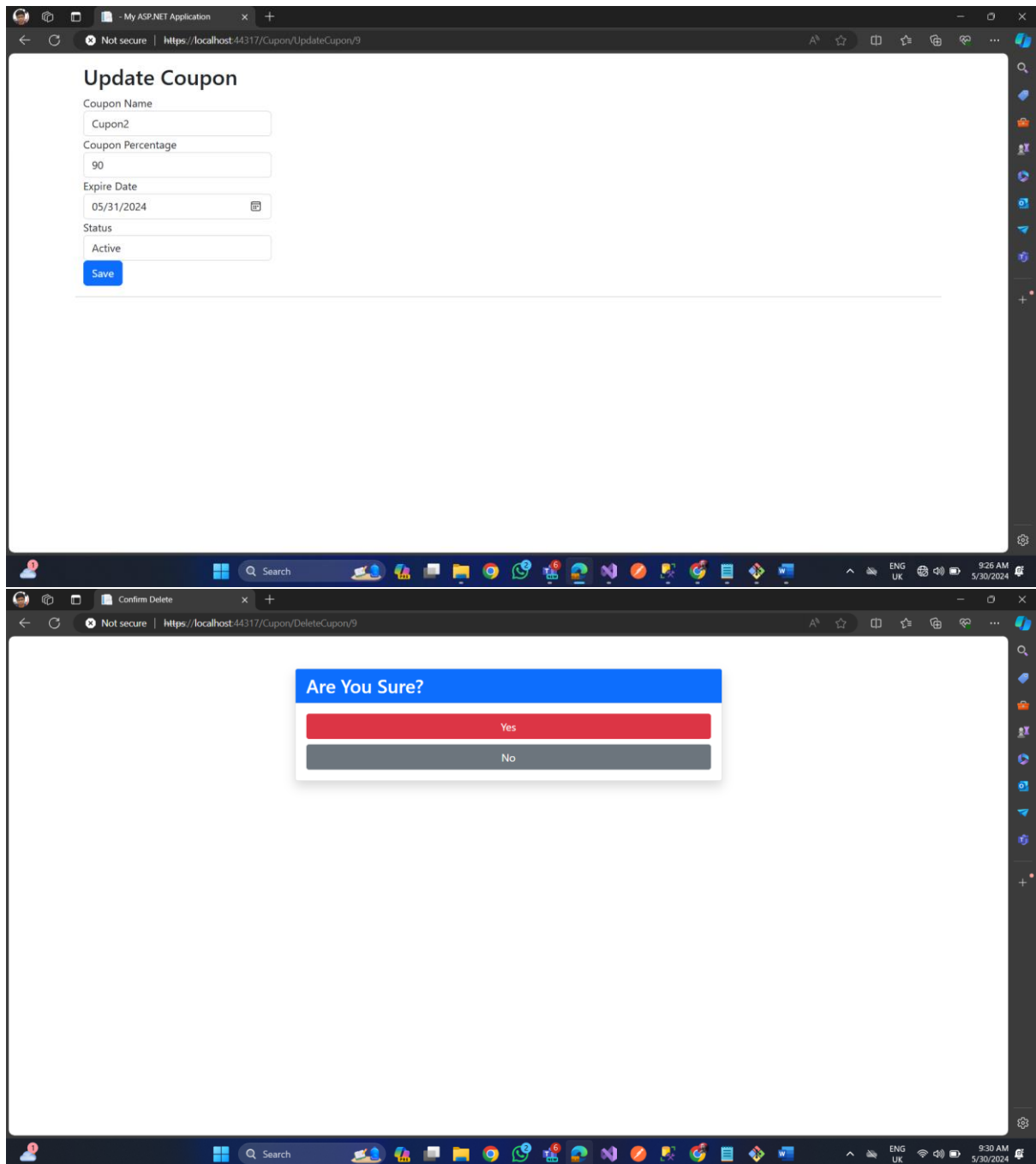
Status:

**Available Coupons Page:**

Available Coupons

Filter by Status:

Coupon Id	Coupon Name	Coupon Percentage	Status	Actions
7	Cupon9	50	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
9	Cupon2	90	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
10	Cupon3	60	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
11	Cupon7	70	Expired	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
12	Discount10	49	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
14	c19	10	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
15	cupon99	99	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
17	cupon999	99	Active	<input type="button" value="Edit"/> <input type="button" value="Delete"/>



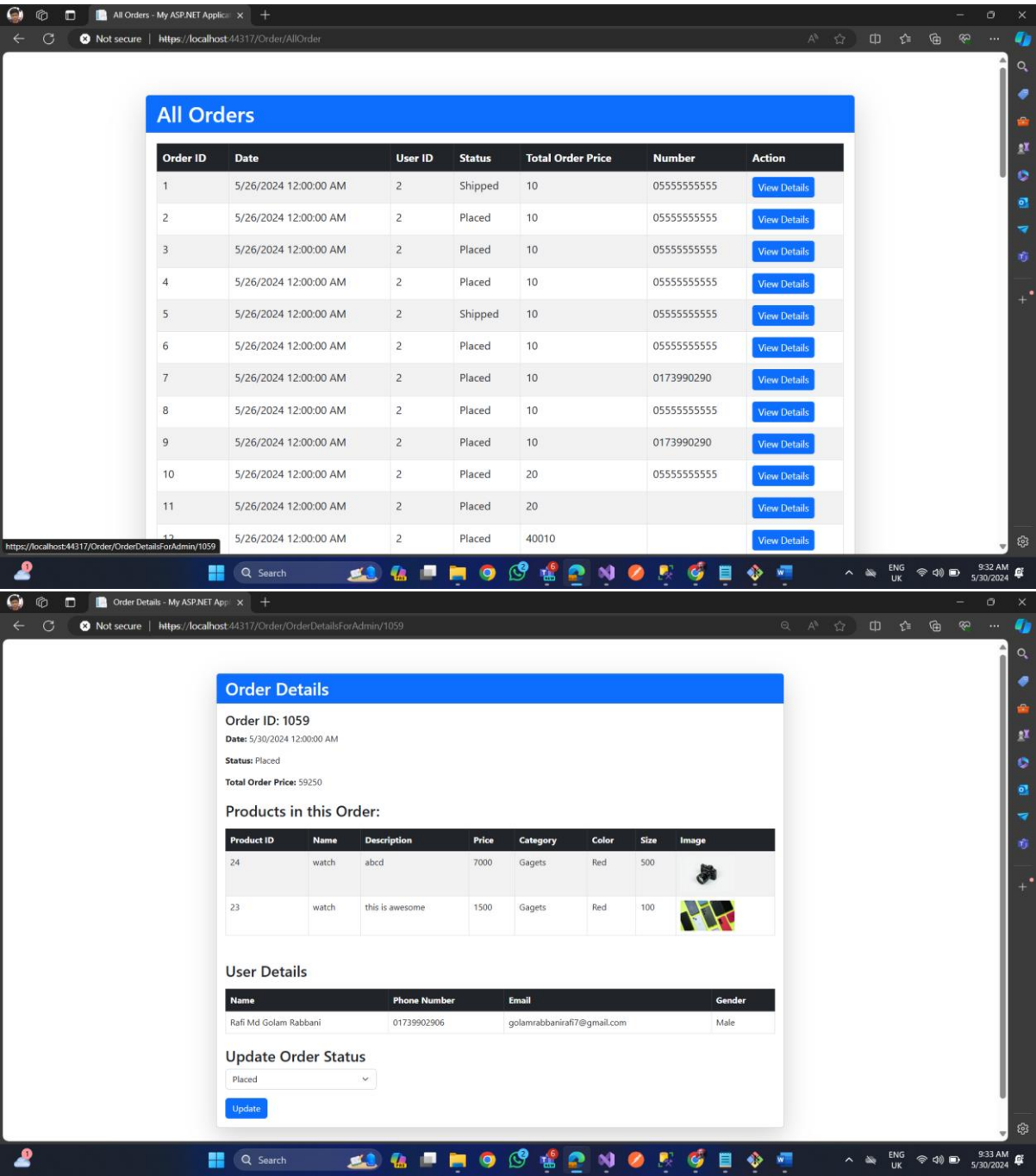
## 6. Orders Management

### Orders List Page

#### Components:

Orders Table (Columns: Order ID, Date, User, Status, Total Price, Actions),  
View Details and Change Status Buttons for Each Order, Order Details Page

Components: Order Summary (User Info, Order Date, Status), List of Products in the Order (Columns: Product Name, Color, Size, Quantity, Price), Change Order Status Dropdown, Save Button

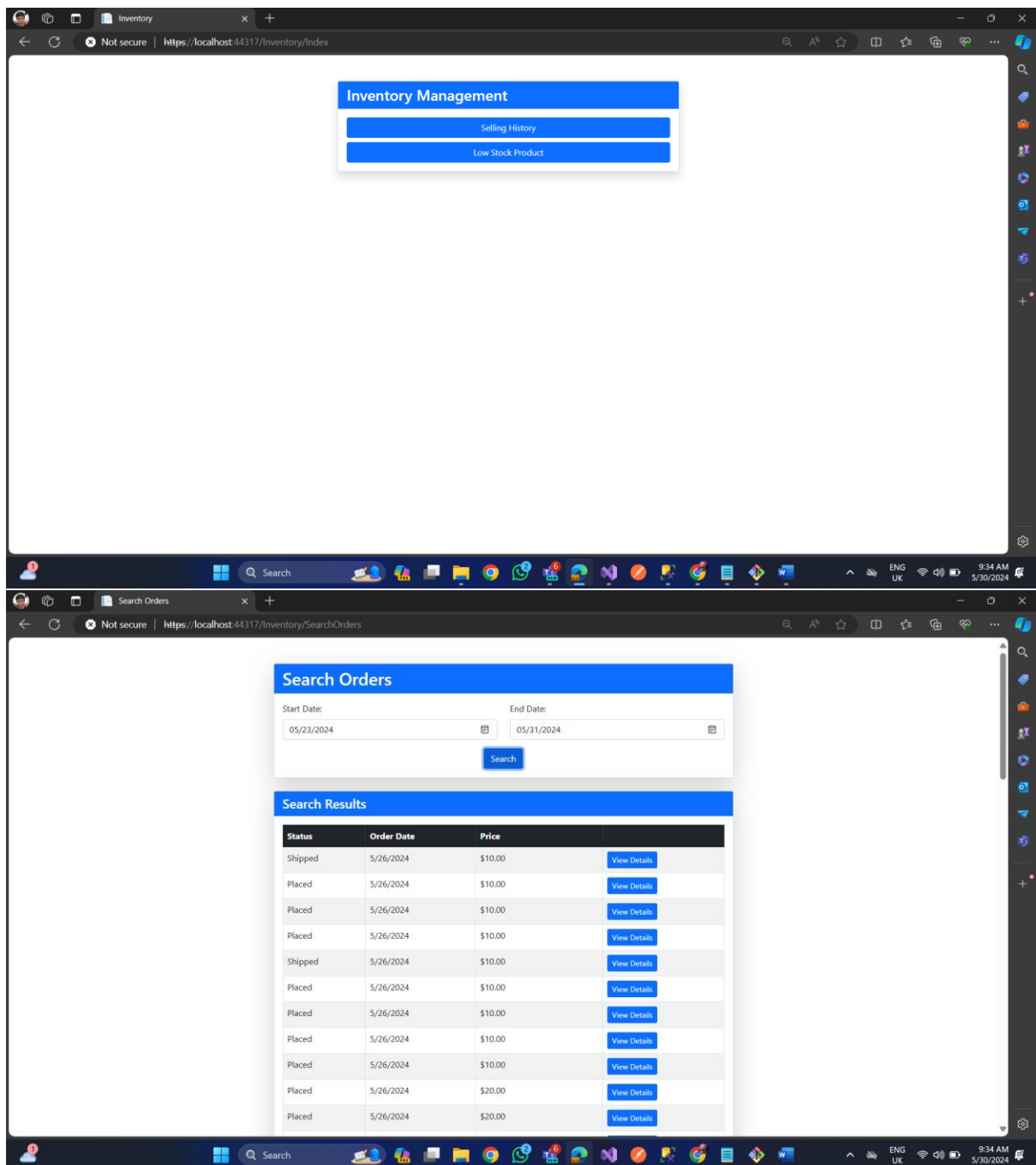


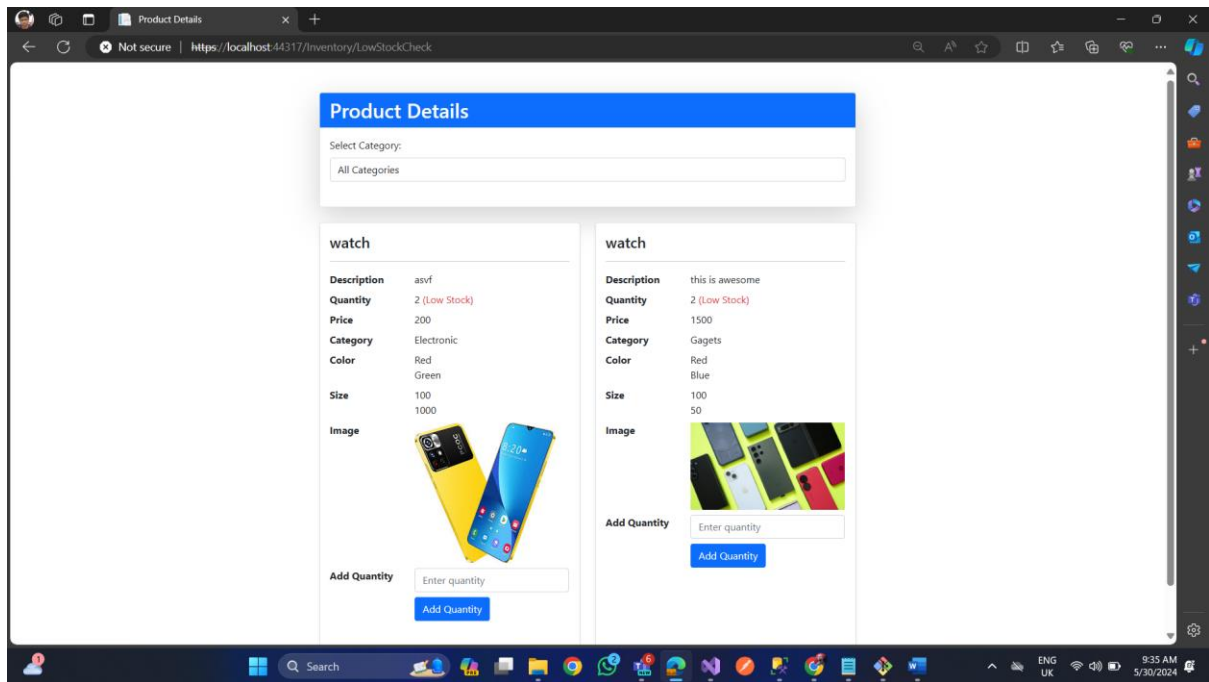
## 7. Inventory Management

Low Stock Products Page

Components: List of Low Stock Products (Columns: Product Name, Current Quantity, Add Quantity), Add Quantity Input Field for Each Product, Save Button, Order History Search Page

Components: Date Range Picker (Start Date, End Date), Search Button, Orders Table (Columns: Order ID, Date, User, Status, Total Price)





## APIs and Integrations:

1. Landing page: <https://localhost:44317/>
2. Signup page: <https://localhost:44317/Home/SignUp>
3. Admin Dashboard: <https://localhost:44317/Home/AdminDashboard>
4. Product Landing page: <https://localhost:44317/Product/Index>
5. Add category page: <https://localhost:44317/Product/AddCategory>
6. Add product page: <https://localhost:44317/Product/AddProduct>
7. Search all product page: <https://localhost:44317/Product/Search>
8. Edit product page: <https://localhost:44317/Product/UpdateProduct/{id}>
9. Delete Product page: <https://localhost:44317/Product/DeleteProduct/{id}>
10. Coupon Index page: <https://localhost:44317/Cupon/Index>
11. Add Coupon page: <https://localhost:44317/Cupon/AddCupon>
12. Available Coupon page: <https://localhost:44317/Cupon/AvailableCupon>
13. Edit coupon page: <https://localhost:44317/Cupon/UpdateCupon/{id}>
14. Delete coupon page: <https://localhost:44317/Cupon/DeleteCupon/{id}>
15. View all order page: <https://localhost:44317/Order/AllOrder>
16. Order details for admin page:  
<https://localhost:44317/Order/OrderDetailsForAdmin/{id}>
17. Inventory Index page: <https://localhost:44317/Inventory/Index>
18. Search Orders by date page:  
<https://localhost:44317/Inventory/SearchOrders>
19. Order details for User page:  
<https://localhost:44317/Order/OrderDetailsForAdmin?id=14>

- 20. Low stock product check page:  
<https://localhost:44317/Inventory/LowStockCheck>
- 21. User Dashboard page: <https://localhost:44317/Home/UserDashboard>
- 22. View cart page: <https://localhost:44317/Product/ViewCart>
- 23. Search All product by user page:  
<https://localhost:44317/Product/SearchAllProduct>
- 24. Details of a specific product page:  
<https://localhost:44317/Product/Details/{id}>
- 25. Order Placed page:  
<https://localhost:44317/Product/OrderPlaced?productId=18&price=200&colorId=1&sizeId=1&quantity=1>

## **Conclusion:**

In conclusion, the e-commerce website project represents a significant step towards providing a robust and efficient online shopping platform. By addressing key functionalities, ensuring user-friendly design, and leveraging a scalable architecture, this project sets a strong foundation for future growth and development. Through continuous improvement and adaptation to emerging trends, the website can evolve to meet the changing needs of users and administrators alike, ensuring long-term success in the competitive e-commerce landscape.