Online Shopping

Abstraction

The online shopping system is a web application built using Asp.net Core Web API, Entity Framework Core, SQL Server, and React. It allows users to register, log in, and search for products. Users can view product statistics, add products to their favorite list, and manage their favorite products. The system includes an admin interface for managing users and products. Key functionalities are secured with optional JWT authentication, and the application supports sorting and pagination. The architecture ensures scalability, performance, and security, with thorough unit testing and comprehensive API documentation using Swagger.

Requirements

- User authentication (register, login, logout)
- Search products by name
- View product statistics
- Add products to favorite list
- View favorite products

Technical Requirements

- **Backend**
  - Asp.net Core Web API
  - Entity Framework Core for ORM
  - SQL Server as the database
- **Frontend**
  - React for building the user interface
- **APIs**
  - External API for fetching product details

Modules for Users

- ***User Interface (React)***

- **Register Page**: Form for user registration.
- **Login Page**: Form for user login.
- **Search Page**: Search bar to find products by name.
- **Product Statistics Page**: Display product statistics.
- **Favorite List Page**: List of user's favorite products with options to add, view, and remove products.
- **Logout**: Option to log out.

- ***User Service (Asp.net Core Web API)***

  - Register a new user.
  - Authenticate a user.
  - Log out a user.
  - Get user details.
  - Update user details.

- ***Favorite Service (Asp.net Core Web API)***

  - Get all favorite products for a user.
  - Add a product to favorites.
  - Remove a product from favorites.
  - Get details of a favorite product.

Modules for Admin

- ***Admin Interface (React)***

- **Admin Dashboard**: Overview of system statistics and user activities.
- **Manage Users**: View, update, and delete user accounts.

- **Manage Products**: Add, update, and delete product information.
- **View Product Statistics**: Detailed product statistics including user interactions.

- ***Admin Service (Asp.net Core Web API)***
    - Get a list of all users.
    - Update user details.
    - Delete a user account.
    - Get a list of all products.
    - Add a new product.
    - Update product details.
    - Delete a product.
    - Get detailed statistics of products.

Additional Common Modules

- ***Authentication Service***
    - User/Admin login.
    - User/Admin logout.
    - User registration.

- ***Product Service (Asp.net Core Web API)***
    - Search for products.
    - Get details of a product.
    - Get product statistics.
-

Functionality

User Functionality

User Interface (React)

- **Register Page**: User account creation.
- **Login Page**: User authentication.
- **Search Page**: Product search by name.
- **Product Statistics Page**: View product metrics.
- **Favorite List Page**: Manage favorite products.
- **Logout**: User log out.

User Service (Asp.net Core Web API)

- Register new user.
- User login.
- User logout.
- Get user details.
- Update user details.

Favorite Service (Asp.net Core Web API)

- Get user’s favorite products.
- Add product to favorites.
- Remove product from favorites.
- Get favorite product details.

Admin Functionality

Admin Interface (React)

- **Admin Dashboard**: Overview of statistics and activities.
- **Manage Users**: View, update, delete user accounts.
- **Manage Products**: Add, update, delete products.

- **View Product Statistics**: Detailed product metrics.

Admin Service (Asp.net Core Web API)

- Get all users.
- Update user details.
- Delete user.
- Get all products.
- Add new product.
- Update product details.
- Delete product.
- Get product statistics.

Common Modules

Authentication Service

- Login for user/admin.
- Logout for user/admin.
- User registration.

Product Service (Asp.net Core Web API)

- Search products.
- Get product details.
- Get product statistics.

User Entities

1. User

- **Attributes**:
  - `User Id`: Unique identifier for the user.
  - `Username`: Username of the user.

  - `Password`: Password for the user account (should be stored hashed).
  - `Email`: Email address of the user.
  - `FirstName`: First name of the user.
  - `LastName`: Last name of the user.
  - `Role`: Role of the user (e.g., User, Admin)

2. Favorite

- **Attributes**:
  - `FavoriteId`: Unique identifier for the favorite entry.
  - `UserId`: Identifier for the user who added the product to favorites.
  - `ProductId`: Identifier for the product added to favorites.
  - `AddedDate`: Date when the product was added to favorites.

Product Entities

3. Product

- **Attributes**:
  - `ProductId`: Unique identifier for the product.
  - `Name`: Name of the product.
  - `Description`: Description of the product.
  - `Price`: Price of the product.
  - `Category`: Category of the product.
  - `StockQuantity`: Quantity of the product in stock.
  - `ImageURL`: URL of the product image.

4. ProductStatistics

- **Attributes**:
  - `ProductId`: Unique identifier for the product.
  - `Views`: Number of times the product has been viewed.
  - `AverageRating`: Average rating of the product.
  - `NumberOfRatings`: Total number of ratings.

Authentication Entities

5. Authentication Token

- **Attributes**:
  - `TokenId`: Unique identifier for the token.
  - `UserId`: Identifier for the user associated with the token.
  - `Token`: The authentication token.
  - `ExpiryDate`: Expiry date of the token.

Admin Entities

6. Admin

- **Attributes**:
  - `AdminId`: Unique identifier for the admin.
  - `Username`: Username of the admin.
  - `Password`: Password for the admin account (should be stored hashed).
  - `Email`: Email address of the admin.
  - `Role`: Role of the admin (typically Admin).