Abstract

Jean Station, a retail chain specializing in apparels, is currently facing significant challenges with its traditional retail marketing methods and limited online presence. The existing system, which relies heavily on print and electronic media for product promotion and a basic website to showcase products, often leaves customers dissatisfied due to product unavailability in stores. Additionally, the absence of an e-commerce platform places Jean Station at a disadvantage compared to competitors who offer online shopping solutions. To overcome these issues, Jean Station's management has decided to develop an advanced e-commerce application named Jean Station Online. This application, to be built using DotnetCore and React, aims to automate the product selling process and expand the company's market reach.

In its initial phase, Jean Station Online will provide customers with the ability to browse products, manage an online shopping basket, and place orders with delivery options. Administrators will have capabilities to manage product inventories, view and update order statuses, and configure discounts through an API. The application will ensure a uniform design across all pages, utilize token-based authentication and role-based authorization, validate user data, and incorporate a microservice for discount notifications. Future development phases will enable individual stores to manage orders specific to their locations, thereby enhancing operational efficiency and customer satisfaction.

Requirements:

Customer Operations

1. **Browse Products**
    - Users can view a list of available products.
    - Users can filter and sort products based on various criteria (e.g., category, price, popularity).
2. **Shopping Basket Management**
    - Users can add products to their shopping basket.
    - Users can remove products from their shopping basket.
    - Users can view the contents of their shopping basket.
3. **Order Placement**
    - Users can place an order for the products in their shopping basket.
    - Users can provide delivery information (address, contact details).
    - The system should calculate the total price of the order including any applicable discounts.

Administrator Operations

4. **Product Management**

- Administrators can add new products to the store.
- Administrators can edit existing product information.
- Administrators can delete products from the store.

5. **Order Management**
    - Administrators can view the status of all placed orders.
    - Administrators can update the status of an order to "Delivered" or other relevant statuses.

Discount Configuration

6. **Discounts**
    - Discounts on the total price of an order can be configured via an API.
    - Discount rules and configurations are managed through a microservice.

Company Information

7. **Company Profile**
    - The application provides details about the company including its name, address, phone number, email, and website.

Non-Functional Requirements

8. **Technology Stack**
    - The application should be developed using Dot Netcore for the backend and React for the frontend.
9. **Design and Navigation**
    - The design, layout, and navigation of the application should be uniform across all web pages to ensure a consistent user experience.
10. **Session Tracking**
    - Shopping activities of customers should be tracked across sessions through a shopping cart mechanism.
11. **Authentication and Authorization**
    - User authentication should be token-based to secure access.
    - User authorization should be role-based to control access to different features.

12. **Data Validation**
    - Data provided by the user must be validated to ensure correctness and security.
13. **Discount Notifications**
    - Discount notifications should be easily configurable through a microservice, allowing for flexibility in discount management.

14. **Company Information**
    - The application should provide clear and accessible information about the company and its contact details.

Modules And Functionalities:

1.Customer Module

- **Browse Products**: Allows customers to view products available in the store.
- **Shopping Basket Management**: Enables customers to add and remove products from their shopping basket.
- **Order Placement**: Allows customers to place an order and provide delivery information.
- **User Authentication**: Token-based authentication for secure access.
- **User Authorization**: Role-based access to different features of the application.
- **Session Tracking**: Tracks customer shopping activities across sessions.
- **Company Information**: Provides details about the company and contact information.

2.Administrator Module

- **Product Management**: Includes functionality to add, edit, and delete products.
- **Order Management**: Allows viewing and updating the status of placed orders.
- **Discount Configuration**: Interface to configure discounts on the total price of an order through API.

3.Discount Microservice Module

- **Discount Notifications**: Manages discount configurations and notifications, interfacing with the main application through API.

4.Shopping Cart Module

- **Cart Management**: Handles the shopping cart operations, ensuring continuity of the customer's shopping activities.

5.Authentication and Authorization Module

- **Token-Based Authentication**: Manages secure access for users.
- **Role-Based Authorization**: Enforces access control based on user roles.

Entities:

1. Customer Role

- **Customer**: Represents a user who interacts with the application to browse products, manage their shopping basket, and place orders.
    - **Attributes**: Customer ID, Name, Email, Password (hashed), Address, Phone Number, Order History, Basket ID.
- **Shopping Basket**: Represents the collection of items a customer has selected to purchase.
    - **Attributes**: Basket ID, Customer ID, List of Products (with quantities), Total Price.
- **Order**: Represents a completed transaction made by the customer.
    - **Attributes**: Order ID, Customer ID, Basket ID, Delivery Address, Total Price, Order Status, Date and Time.

2. Administrator Role

- **Administrator**: Represents a user with elevated permissions to manage products and orders.
    - **Attributes**: Admin ID, Name, Email, Password (hashed), Role (e.g., Admin), Permissions.
- **Product**: Represents an item available for purchase in the store.
    - **Attributes**: Product ID, Name, Description, Price, Category, Stock Quantity, Image URL.
- **Order Status**: Represents the status of an order (e.g., Pending, Shipped, Delivered).
    - **Attributes**: Status ID, Order ID, Status Description, Date and Time Updated.

3. Discount Microservice Role

- **Discount**: Represents the discount configurations that can be applied to orders.
    - **Attributes**: Discount ID, Description, Discount Percentage or Amount, Validity Period, Applicable Products/Categories, API Endpoint for Application.

4. Shopping Cart Module

- **Shopping Cart**: Typically, part of the customer entity but focuses on the session-specific details of the cart.
    - **Attributes**: Cart ID, Customer ID, List of Products (with quantities), Total Price, Session ID.

5. Authentication and Authorization

- **User**: A generic entity that could represent both customers and administrators with a unified authentication system.
    - **Attributes**: User ID, Username, Password (hashed), Role (Customer, Administrator), Authentication Token, Permissions.