The supermarket store App

Development of online supermarket store with different genre of products.

Requirements

1. There should be 2 types of users.
   a. Customer
   b. Admin
2. Both the users must be able to login into system using their ID and Password. (ID and Password needs to be matched with database)

Non-User Functionality

a. System should generate order that must be visible to respective Customer and Admin.
b. System should handle order cancel request generated by Customer/Admin.
c. System should mark product out of stock automatically based on available quantity
d. System Should process login functionality for Customer/Admin
e. System must not show inactivate products to any Customer.
f. Customer inactivation by Admin should cancel all the pending order automatically.
g. System should generate and display re-payment schedule to customer if EMI mode is selected based on below ROI
   i. Tenure till 1 Year – 8%
   ii. Tenure more than 1 Year to 2 Year – 10%
   iii. Tenure more than 2 Year to 5 Year – 14%

User Functionality

a. Customer
   iv. Should able to search products by their name, category/genre or description.
   v. Should able to exclude out of stock products
   vi. Should able to add products (1 or more quantity) into cart.
   vii. Should able to see orders and delete them
   viii. Should able to checkout the cart (Do not need to implement payment gateway, dummy page with payment button will be enough)
   ix. Should able to opt for EMI option against the product
b. Admin Should
   x. Add more product(s) with its name, category, price, quantity, description
   xi. Able to cancel any order
   xii. Able to inactivate/activate any product
   xiii. Able to inactivate/activate any Customer
   xiv. Able to see list of inactivate products/Customers

Technology

Frontend: React
Backend: Spring Boot, log4j, Hibernate, Java8
Database: Any
Build Tool – Maven
Repo – Git

Assessment Parameters

- Frontend must communicate with backend using REST API only.
- Code completeness
- Code quality (Formatting, hardcoding, comments, exception handling, logger, Reusability)
- Design (SOLID Principle, Design pattern, Maven parent-child structure)
- Adherence to coding guidelines and best practises
- All the master data must be read from database.
- Unit Testing and Coverage
- Usage of java 8 features