Introduction

Project Title: Store Manager: Keep Track of Inventory

Team Leader: Thanigaivasan.K

Team Members: Tamilselvan.B, Udith Narayanan.S, Uthith.S

Project Overview

Purpose: This project is an inventory management system designed to help stores maintain healthy stock levels and manage sales efficiently. It focuses on providing a comprehensive solution for tracking inventory, managing sales records, and handling product information.

Features:

* Inventory Management: Helps maintain and update stock levels for products.

* Stock Updates: Automatically updates stock when products are sold and allows manual updates when new stock is added.

* Cart Functionality: Enables adding products to a cart and adjusting quantities for individual sales.

* Checkout: Clears the cart, updates inventory, and creates a sales record upon checkout.

* Product Management: Allows adding new products to the inventory with details like name, image URL, price, stock, and tags.

* Low Stock Alerts: Highlights products with depleting stock levels using a red background and customizable alert counts.

* Search Functionality: Provides search capabilities for products within the inventory and product catalog.

* Sales Records: Stores a history of all sales, including the total value, products sold, and date/time.

Architecture

Component Structure: The application's major components are organized to handle specific functionalities, such as managing the product catalog, the cart, and the sales history.

State Management: The project uses a state management approach to handle both global and local states.

Routing: The application uses a routing library like react-router to handle navigation between different views, such as the product inventory, the cart, and the sales records page.

Setup Instructions

Prerequisites: Node.js is required to run the application.

Installation:

* Clone the project repository.

* Navigate to the client directory.

* Install dependencies using npm install.

* Configure environment variables as needed.

Folder Structure

Client:

* Components: Reusable UI elements such as buttons, cards, and forms.

* Pages: Top-level components representing different application views (e.g., InventoryPage, CartPage).

* Assets: Static files like images and CSS.

* Utilities: Helper functions, custom hooks, and other reusable logic.

Running the Application

Frontend: To start the frontend server, run the command npm start in the client directory.

Component Documentation

Key Components:

* ProductCard: Displays individual product information.

* CartItem: Represents a single product in the cart.

* SalesHistoryTable: Shows a list of past sales records.

Reusable Components: The project includes reusable components like input fields, buttons, and modal dialogs that are used across different parts of the application.

State Management

Global State: A global state management solution, such as the Context API or Redux, is used to manage the inventory, cart contents, and sales records, ensuring state consistency across the application.

Local State: The local state is handled within individual components to manage UI-specific data, such as form inputs or button states.

User Interface & Styling

User Interface: The UI is designed to be intuitive for managing inventory and sales. It features clear layouts for viewing products, a dynamic cart, and easy-to-read sales records.

Styling: The project utilizes a CSS framework or library, such as Sass or Styled-Components, to maintain a consistent and responsive design.

Testing

Testing Strategy: Components are tested using a framework like Jest or the React Testing Library to ensure they render correctly and behave as expected.

Code Coverage: Code coverage tools are used to monitor and ensure that a sufficient portion of the code is covered by tests.

Screenshots or Demo

[Link to a demo or a set of screenshots showcasing the application's functionality, including the inventory view, cart, and sales records.]

Known Issues

[List any known bugs or issues, such as minor display inconsistencies or performance issues with large datasets.]

Future Enhancements

* Adding user authentication and multiple user roles.

* Implementing a more advanced reporting and analytics dashboard for sales.

* Improving the user interface with animations and enhanced styling.