**FRONT-END**

**STORE MANAGER-KEEP TRACK OF INVENTORY**

**TEAM ID:** NM2025TMID37923

**TEAM LEADER:**

VARSHINI.S([varshinisenthil702@gmail.com](mailto:varshinisenthil702@gmail.com))

**TEAM MEMBERS:**

Shriharini B.N(shrinatchiyar@gmail.com)

Rasika JS(rasika.naksatra@gmail.com)

Sarah D(sarahdoss@gmail.com)

II BSC Computer science

Shri Shankarlal Sundarbai Shasun jain college for women,

Tnagar, Chennai-17

**REPORT CONTENTS**

1.Introduction to the Project

2.Project coverage

3.System Requirements

4.Setup instructions

5.Project Development

6.Implemented module

7.Screen short

8.Folder structure

9.Running Application

10.Test cases

11.Output

12.Conclusion

13.Reference

14.Future

Enhancements

15. Demo link

16.Github link

**INTRODUCTION TO THE PROJECT**

* The **Store Manager** project is a **frontend web application** designed to help users **manage and track inventory** in a store. It allows the user to **add, view, edit, and delete products** along with their quantities and prices. This project is built using **HTML, CSS, JavaScript, and ReactJS**, ensuring an interactive and responsive user interface.
* This project was created to demonstrate how modern frontend technologies can be combined to build a **simple, user-friendly, and functional inventory system** without using any backend or database. The interface is designed to be clean and intuitive, enabling even non-technical users to handle stock records efficiently.
* The main **objective** of this project is to provide a **quick, organized, and visually clear way to track inventory** .

**PROJECT COVERAGE**

**1. What does this project cover?**

* Adding new products with details like name, quantity, and price, stock.
* Viewing all products in an organized list or table format.
* Editing existing product details whenever required.
* Deleting products that are no longer in stock.
* Maintaining a simple, responsive, and user-friendly interface without needing backend setup or a database.

**2. What does this project not cover?**

* Multi-user access or user authentication.
* Permanent data storage; data will be lost on page refresh or browser close.
* Advanced features like sales tracking, reports generation, or analytics.
* Backend functionality or server-side operations.

**SYSTEM REQUIREMENTS**

**1. Hardware Requirements**

|  |  |  |
| --- | --- | --- |
| Component | Minimum Requirement | Recommended Requirement |
| Processor | Intel Core i3 or equivalent | Intel Core i5 or higher |
| RAM | 4 GB | 8 GB or more |
| Hard Disk | 250 GB | 500 GB or more |
| Monitor | 14-inch, 1366x768 resolution | 15.6-inch or higher |
| Input Devices | Keyboard and Mouse | Keyboard, Mouse, and Touchpad |

**2. software Requirements**

|  |  |  |
| --- | --- | --- |
| Component | Specification / Version | Purpose / Usage |
| Operating System | Windows 10 | To run the project environment |
| Code Editor | Visual Studio Code | Writing and editing code |
| Programming Languages | HTML, CSS, JavaScript | Frontend web development |
| Framework / Library | React.js | Building the user interface |
| Runtime Environment | Node.js with npm package manager | Running React.js and managing packages |
| Browser | Google Chrome | Testing and running the web application |
| Version Control | Git hub | Managing project versions and files |

**SETUP INSTRUCTIONS**

**Installation Steps**

1. Install Node.js and npm

Open your browser and search for Node.js official website.

Download the LTS version of Node.js (which includes npm).

Run the installer and complete the installation process.

To verify installation, open the terminal and run:

node -v

npm -v

1. Create React App

Open Visual Studio Code.

Open the terminal inside VS Code.

Run the following command to create a React application:

npx create-react-app store-manager

Wait until all the dependencies are installed.

1. Open and Run the Project

Navigate into the project folder:

cd store-manager

Start the development server:

npm start

The application will run in your default browser at:

[http://localhost:3000](http://localhost:3000/)

Usage Instructions

* Open the browser and go to the running link.

* Navigate through the Home, Inventory, Add Product, and Add cart pages.

* Products once added cannot be deleted or edited

**Project Development**

* The **Store Manager** project was developed as a frontend web application to manage inventory efficiently. The development involved using **React.js, HTML, CSS, and JavaScript** to create a responsive and interactive interface. The project focuses on basic inventory functionalities such as adding, editing, viewing, and deleting products.

The development process followed a systematic approach to ensure clarity, usability, and proper functionality.

**Planning and Design Steps Followed**

**1.Requirement Analysis:**

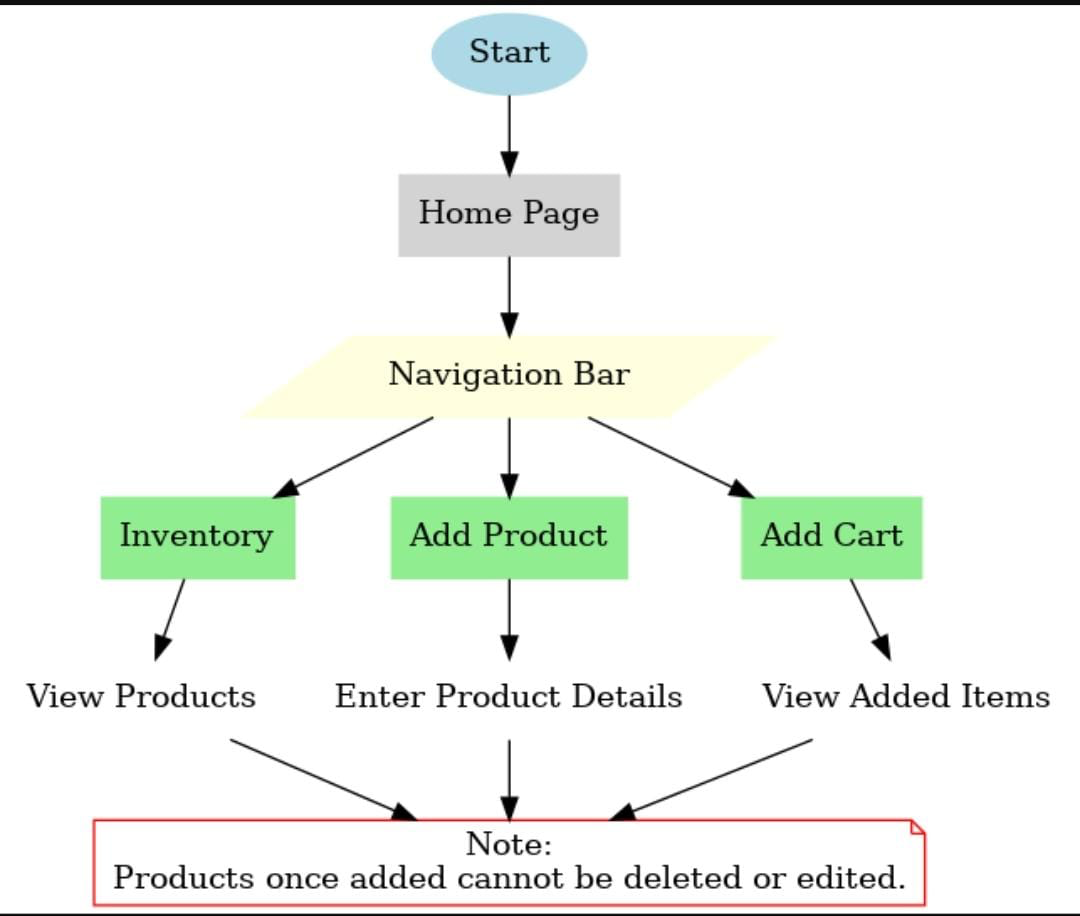
* + Identified the key features needed: add, view, edit, and delete products.
  + Determined the target users (small shop owners, non-technical users).

1. **Designing the Layout:**
   * Created a simple and intuitive interface for easy navigation.
   * Planned product display using tables or cards for clarity.
   * Decided the placement of buttons for add, edit, and delete actions.
2. **Choosing Tools and Technologies:**
   * Selected **React.js** for frontend development.
   * Used **HTML, CSS, JavaScript** for structure, styling, and functionality.
   * Chose **Node.js with npm** for runtime and package management.
   * Used **Visual Studio Code** as the code editor and **Git hub**for version control.
3. **Implementation Planning:**
   * Planned component structure in React for modular development.
   * Decided on state management for handling product data in the frontend.
   * Created a simple workflow for adding, editing.
4. **Testing and Debugging Plan:**
   * Planned step-by-step testing of each functionality.
   * Ensured proper validation of input fields.
   * Checked responsiveness and UI compatibility on different browsers.

6.**Finalization and Documentation:**

* + Prepared the final version of the project with all features working.
  + Documented the project structure, features, and limitations for future reference

**Project workflow**

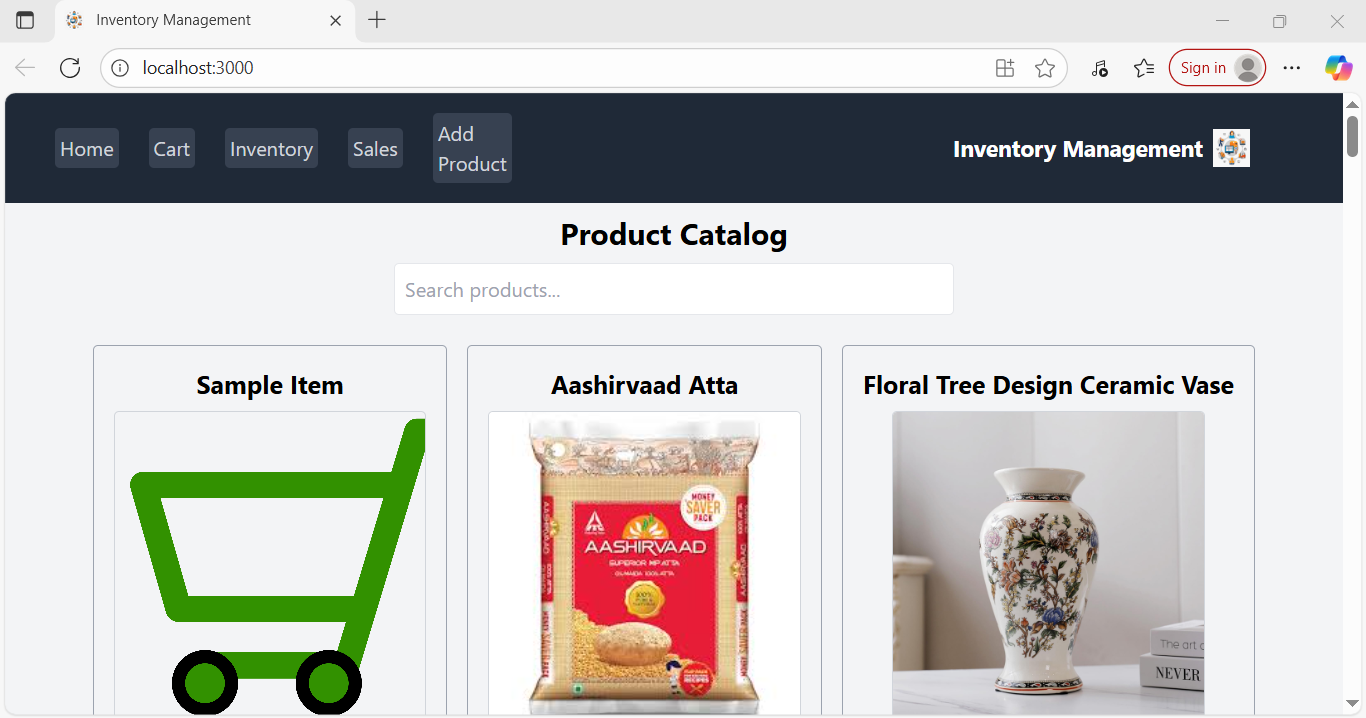


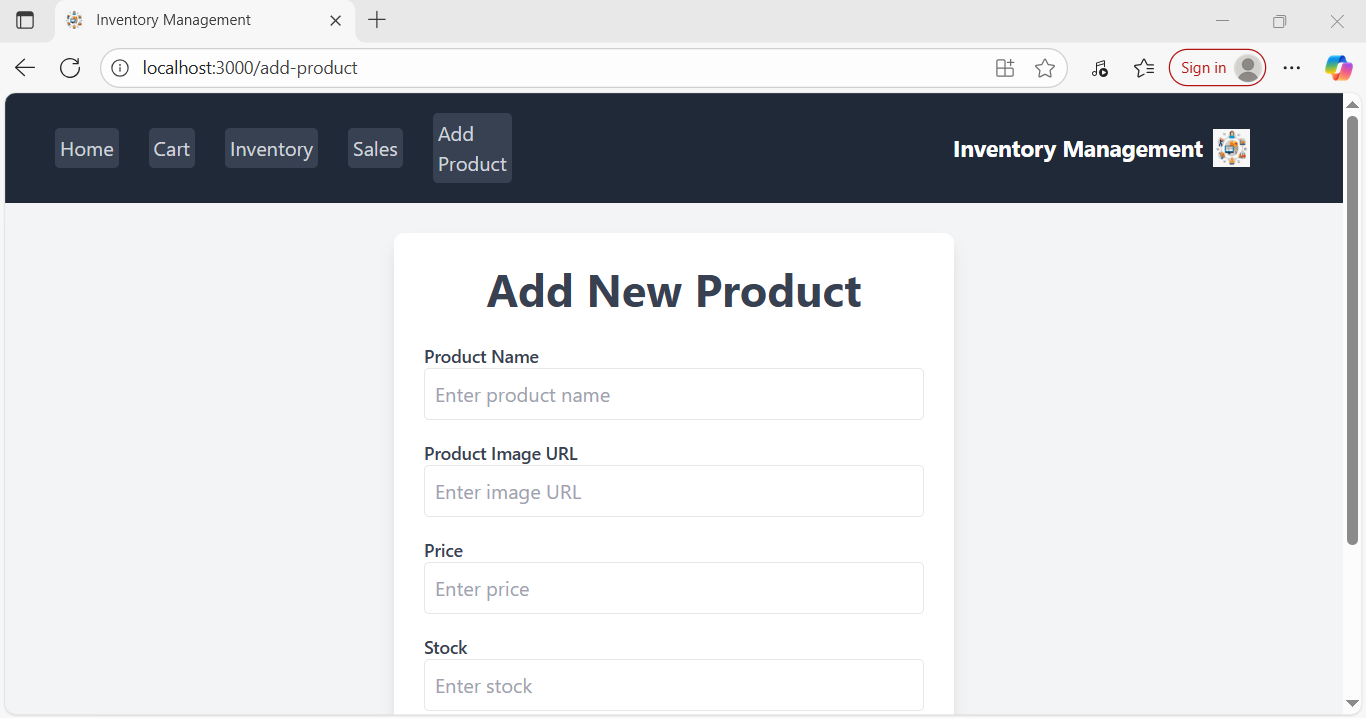
**Implemented modules**

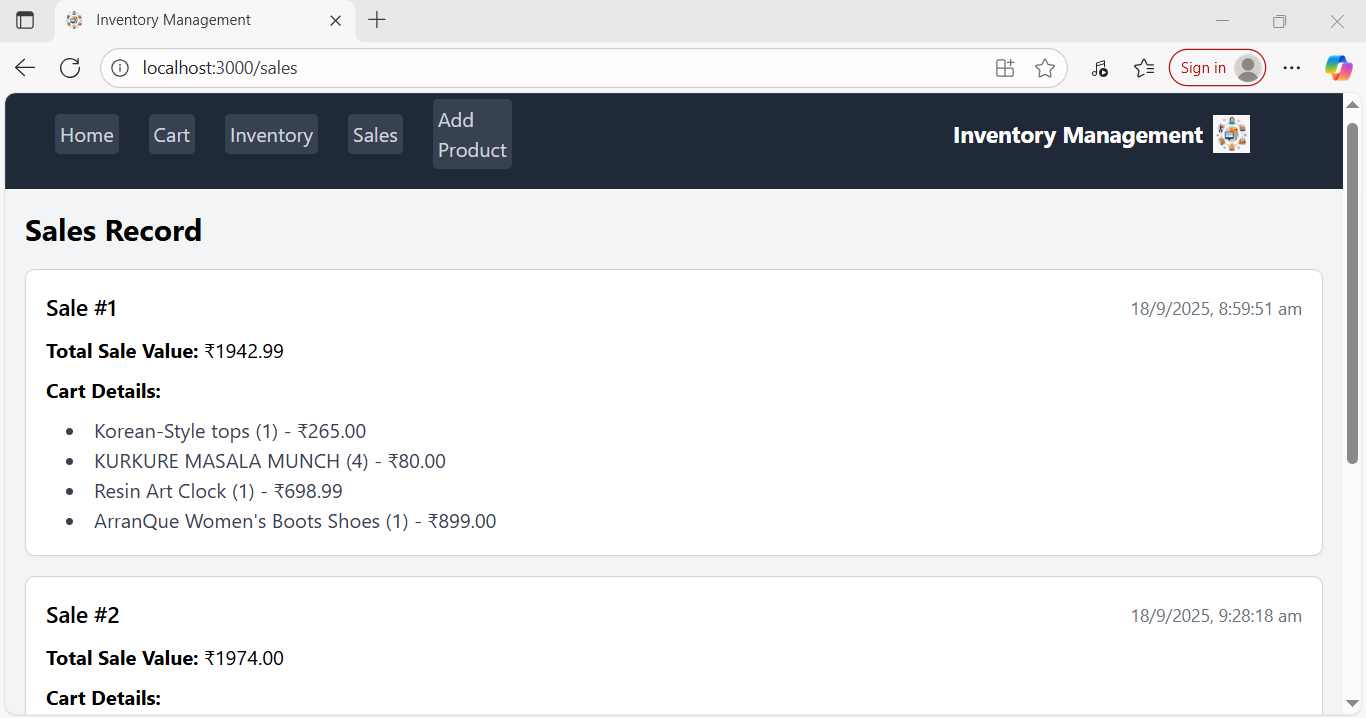
Here are some key features and a description of an inventory management system:

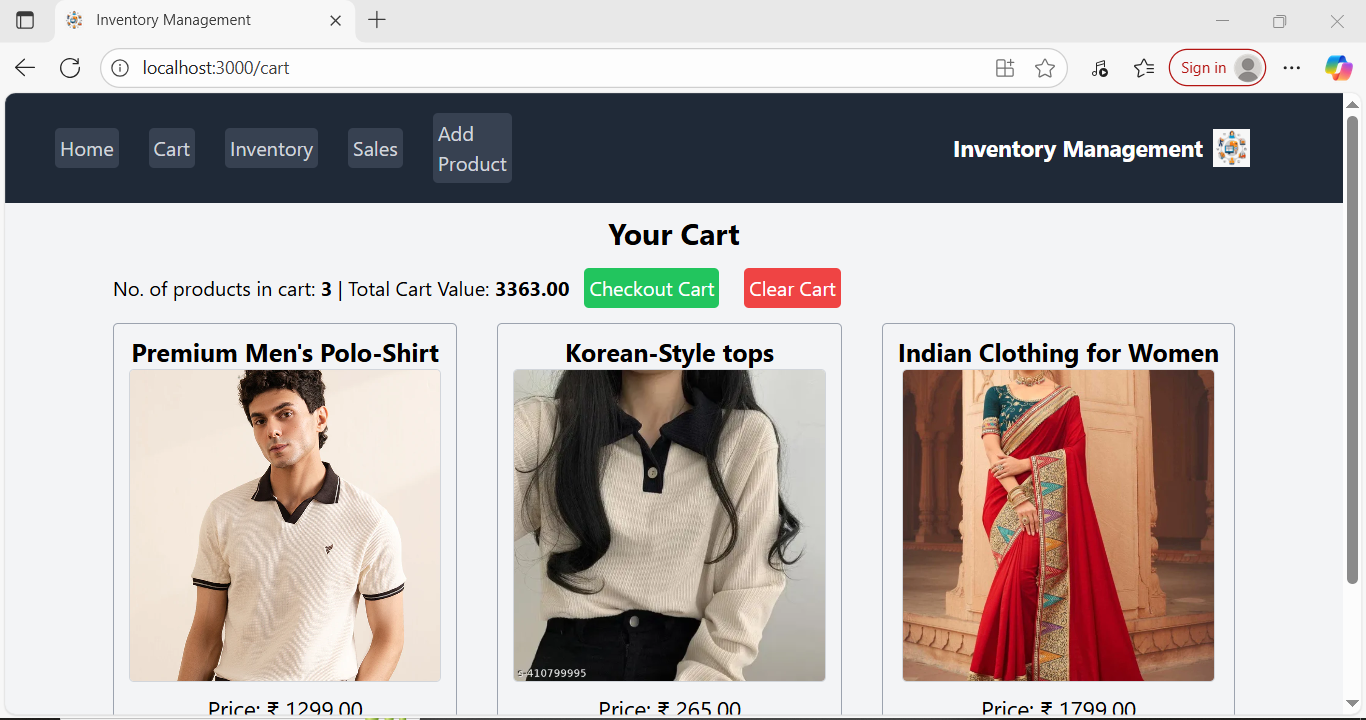
* **Inventory Management:** Inventory Management helps maintain healthy stock levels in a store and acquire them in time.
* **Stock Updates:** Stock will automatically update on sale of products, and it can be updated on adding new stock.
* **Cart:** Products can be added to cart for a particular sale and quantity can be added to each product.
* **Checkout at Cart**: Upon checkout, cart is cleared, inventory is updated, and a sale record is made.
* **Adding New Products to Inventory:** New products can be added to the inventory by providing product name, image URL, price, stock, tags.
* **Alert View for Depleting Stock:** Depleting stocks are shown in red background, and alert count can be updated as per requirement.
* **Search Functionality for Products:** Products in inventory and product catalog can be searched.
* **Sale Records:** All sale records are stored with sale value, products and datetime.

*Screenshot*

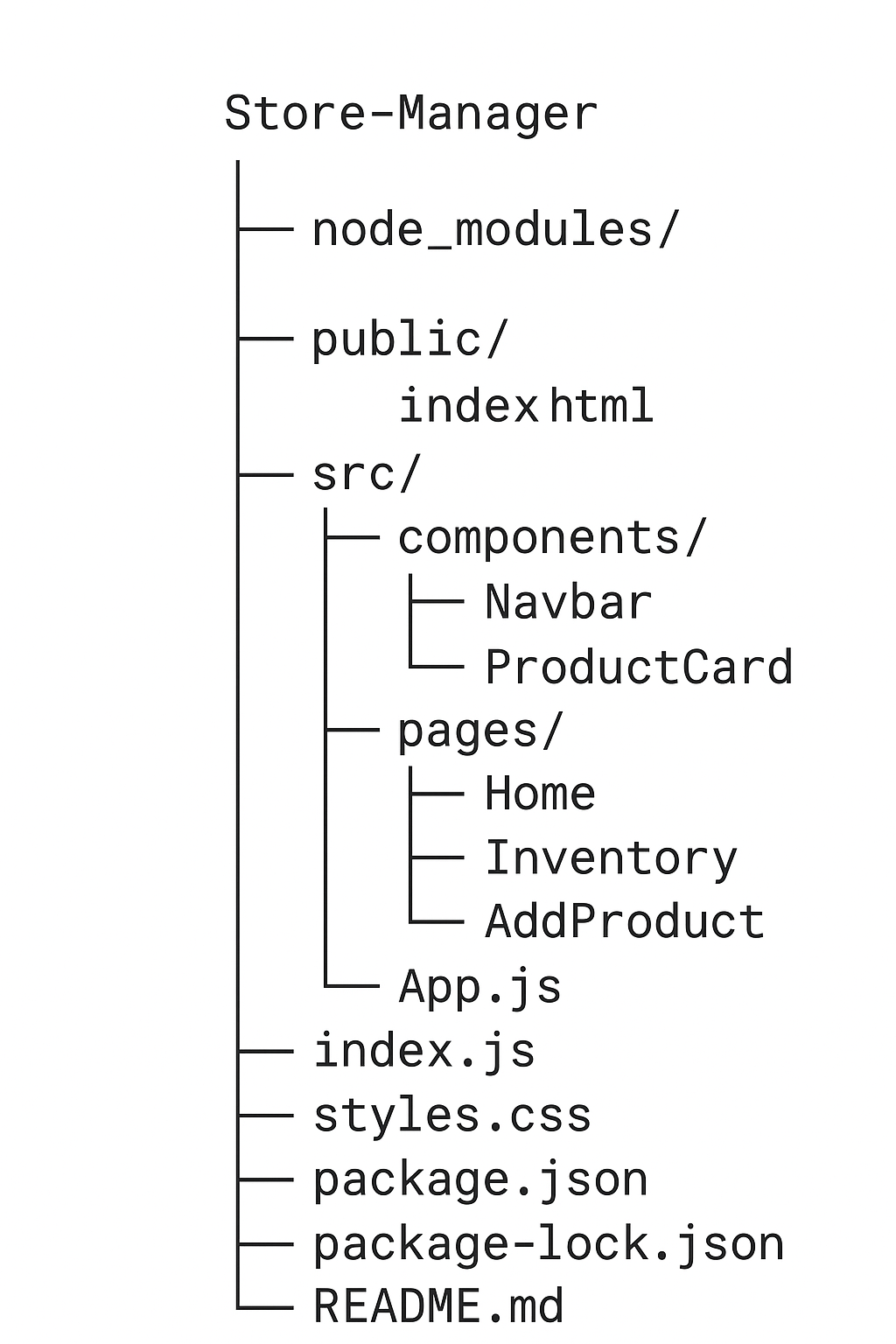








***Folder Structure***



**Running the application**

1. Open the Project Folder

Open Visual Studio Code.

Navigate to the folder where the React app was created.

1. Install Dependencies

Open the terminal in VS Code.

Run the command:

npm install

This will install all required node modules.

To check whether installation was successful, verify that the node\_modules folder appears in the project directory.

1. Start the Application

In the terminal, type:

npm start

Wait for the development server to compile the project.

The application will automatically open in your default browser at:

http://localhost:3000

4. View the Output

Once the browser opens, the Store Manager Keep Inventory Track interface will be displayed.

From here, you can navigate through Home, Inventory, Add product , Add Cart.

**Test cases**

**Navigation Testing:** Clicking on Home, Inventory, Add Product, Add Quote links works correctly.

1. **Add Product Testing:** Products are added successfully with all details.
2. **Add to Cart Testing:** Clicking a product adds it to the cart, and the cart updates correctly.
3. **Display Testing:** Newly added products and cart items appear immediately on the pages.
4. **UI Testing:** Buttons, links, and forms respond correctly to user actions.

**Output**

* **Home Page:** Shows all products with details (name, stock, price, image) and Add to Cart buttons.
* **Inventory Page:** Displays a list of all products with stock information.
* **Add Product Page:** Form allows users to enter product details and adds them dynamically.
* **Cart:** Shows products added via Add to Cart with updated information.
* **Navigation:** Users can switch pages smoothly using the navigation bar.

**Known issue**

During the development and execution of the Store Manager Keep Inventory Track project, the following issues were faced:

1. Errors in the Given Code

The initial code provided contained around five errors.

These errors included , incorrect syntax, and mismatched component names.

After identifying them, corrections were made and the application ran successfully.

2. Dependency Issues

Sometimes, while running npm install, dependencies were not installed correctly.

The issue was fixed by deleting the node\_modules folder and running the command again.

3. Browser Auto-Open

In a few cases, the project did not automatically open in the browser after running npm start.

The issue was resolved by manually opening the link:

http://localhost:3000

1. Uneditable Products (By Design)

Once a product is added, it cannot be edited or deleted.

This is part of the project design but may be considered a limitation for future improvement.

**Conclusion**

The **Store Manager project** successfully demonstrates a simple and interactive web-based application for managing products. Key features such as **viewing products, adding products, and adding items to the cart** are implemented effectively, providing a smooth and user-friendly experience.

The project highlights the importance of **dynamic data handling, responsive navigation, and interactive UI**, making it a practical example of a basic inventory management system.

Overall, this project strengthens understanding of **React.js, forms, state management, and dynamic rendering**, and serves as a foundation for more advanced web development projects in the future.

**References**

* NM Team Tutorial – provided step -by step guidance for developing store manager project

**Future Enhancement**

1. **Edit Product Feature (Optional):** Allow users to update product details like name, stock, price, or image.
2. **Cart Management:** Enable removing items from the cart and updating quantities.
3. **Search and Filter:** Implement search and filter options to quickly find products.
4. **Database Integration:** Connect the app to a database for persistent storage instead of temporary state.
5. **User Authentication:** Add login/signup functionality for secure access to the system.
6. **Delete Product (Optional):** Future versions could allow deleting products if needed.

**Demo link**

https://drive.google.com/file/d/1tkSs9nk3\_XlwWzDQIdxmrzSZHYRS1o47/view?usp=sharing

**GitHub link**

https://github.com/rasika-js/Store-Manager--Keep-Track-of-Inventory.git