**Project Documentation**

**Project Title: Inventory Management System**

**1. Introduction**

* **Project Title**: Inventory Management System
* **Team Leader**: Sanjana B
* **Team Members**:
  + Monika S
  + Nithyavalli T
  + Reeta Therosy S
  + Nivedha D

**2. Project Overview**

* **Purpose**:  
  The Inventory Management System helps businesses track, manage, and update product stocks efficiently. It reduces manual effort, avoids overstock/stock-out situations, and provides real-time insights for better decision-making.
* **Features**:
  + Add, update, and delete products
  + Track stock levels in real-time
  + Sales and purchase history
  + Low stock alerts and notifications
  + Role-based access: Admin and Staff

**3. Architecture**

* **Frontend**: React.js with Tailwind CSS / Bootstrap
* **Backend**: Node.js and Express.js (server logic and API)
* **Database**: MongoDB (stores product details, stock, sales, users)

**4. Setup Instructions**

**Prerequisites**

* Node.js
* MongoDB
* Git
* React.js
* Express.js
* Mongoose
* Visual Studio Code

**Installation Steps**

# Clone the repository

git clone <repo-link>

# Install client dependencies

cd client

npm install

# Install server dependencies

cd ../server

npm install

**5. Folder Structure**

Inventory-Manager/

│-- client/ # React frontend

│ └── components/

│ └── pages/

│-- server/ # Node.js backend

│ └── routes/

│ └── models/

│ └── controllers/

**6. Running the Application**

**Frontend:**

cd client

npm start

**Backend:**

cd server

npm start

**Access:** <http://localhost:3000/inventory>

**7. API Documentation**

* **User**
  + /api/user/register
  + /api/user/login
* **Products**
  + /api/products/add
  + /api/products/:id/update
  + /api/products/:id/delete
  + /api/products/list
* **Inventory**
  + /api/inventory/stock
  + /api/inventory/low-stock
* **Sales & Purchase**
  + /api/sales/create
  + /api/purchase/create

**8. Authentication**

* JWT-based authentication for secure login
* Role-based access (Admin, Staff)

**9. User Interface**

* Admin Dashboard
* Add / Manage Products
* Stock Overview
* Sales & Purchase Reports
* Low Stock Alerts

**10. Testing**

## ****Testing Strategy****

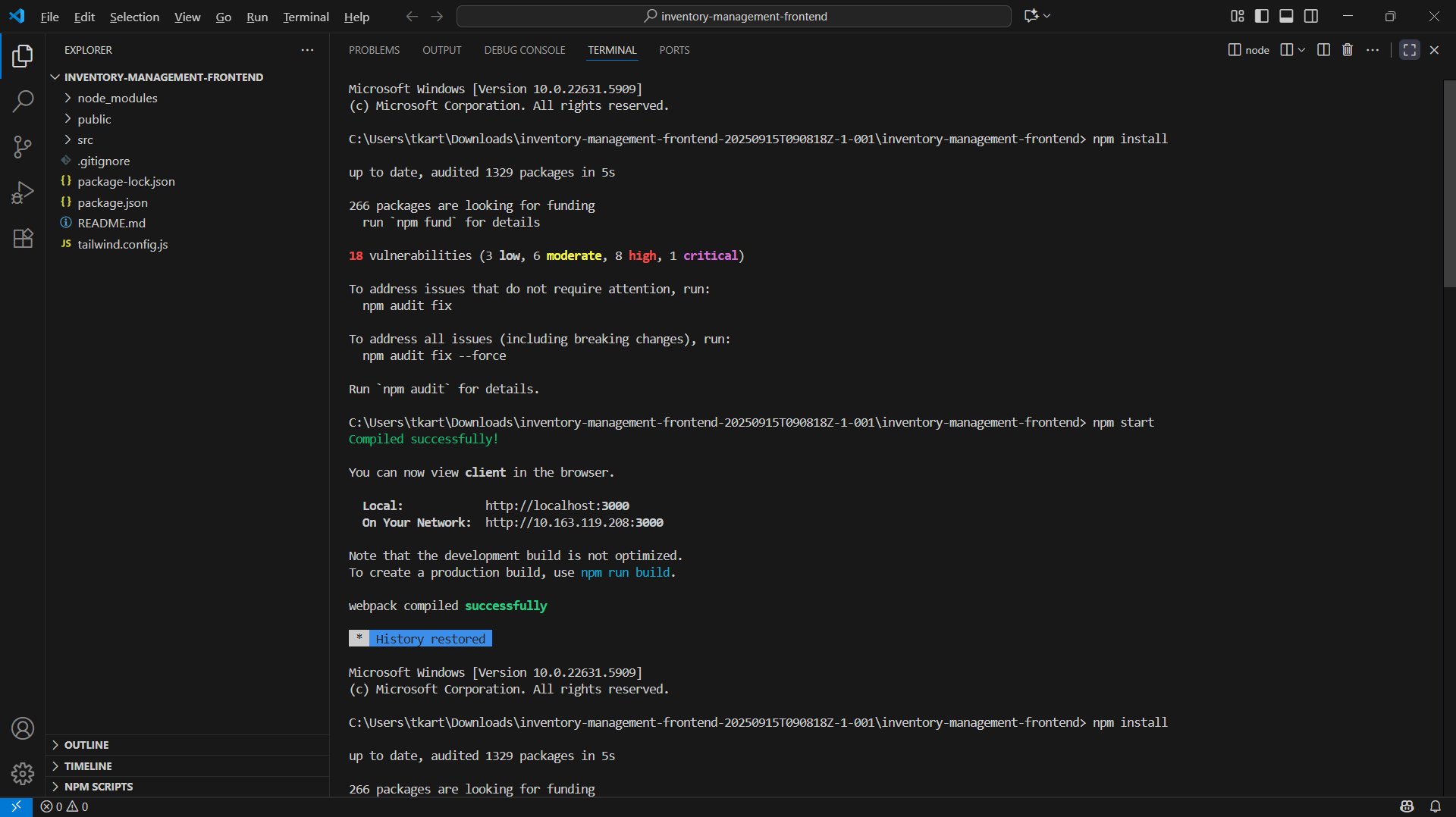
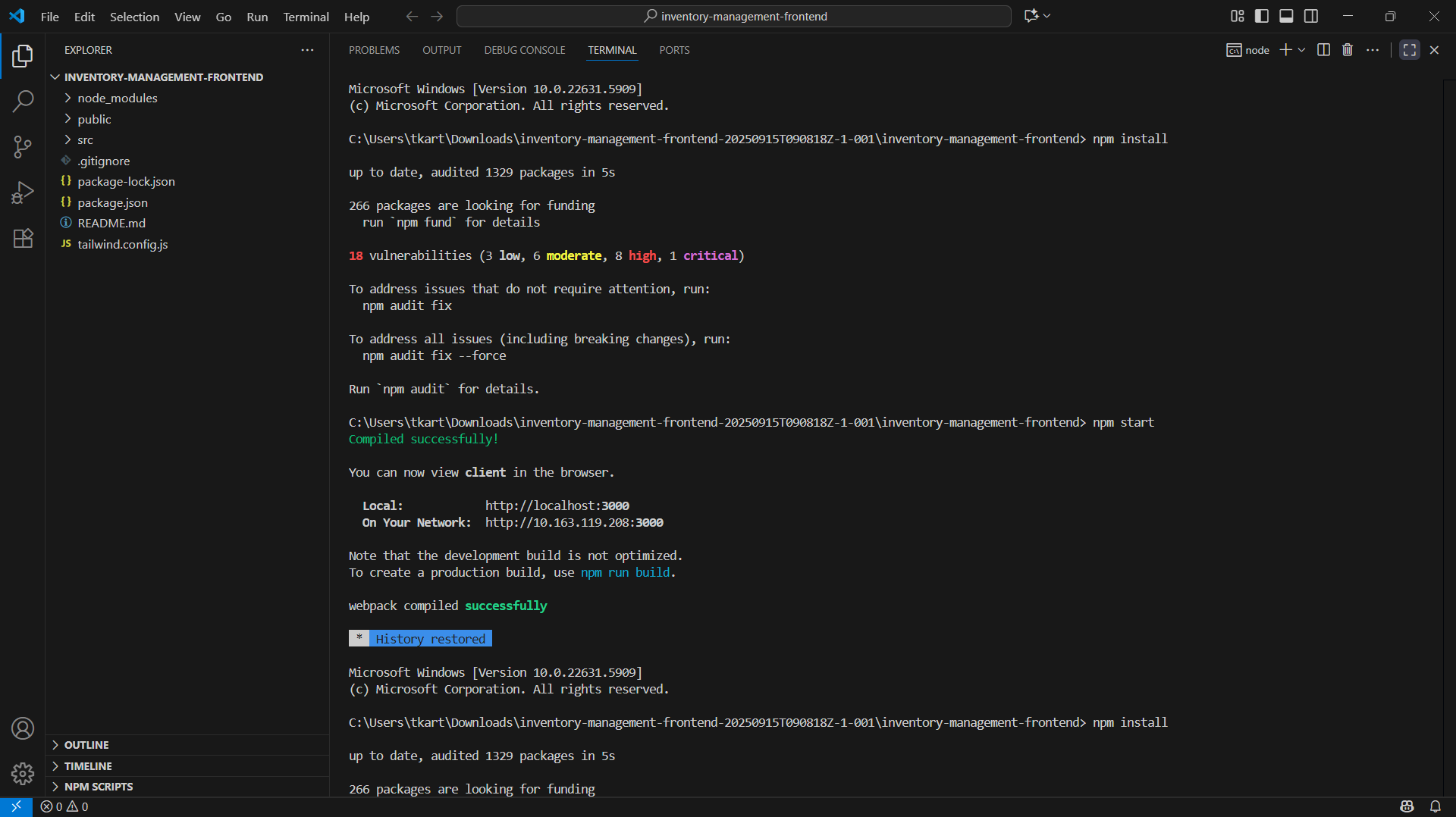
* Manual and functional testing of core features (add/update products, stock tracking, sales, low-stock alerts).
* API endpoints tested using **Postman**.
* UI tested in browser for responsiveness and navigation.
* Authentication verified with role-based access (Admin/Staff).

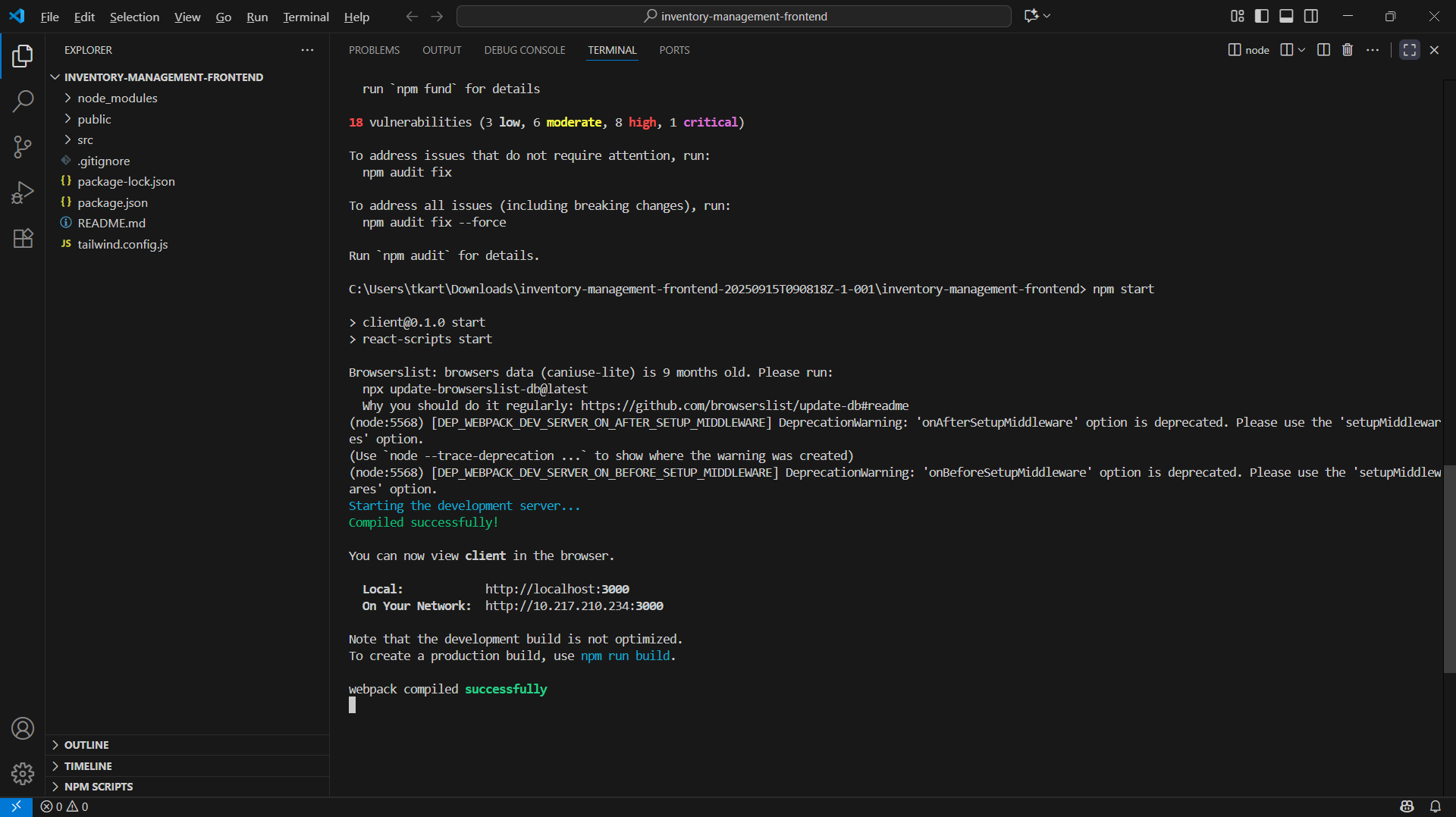
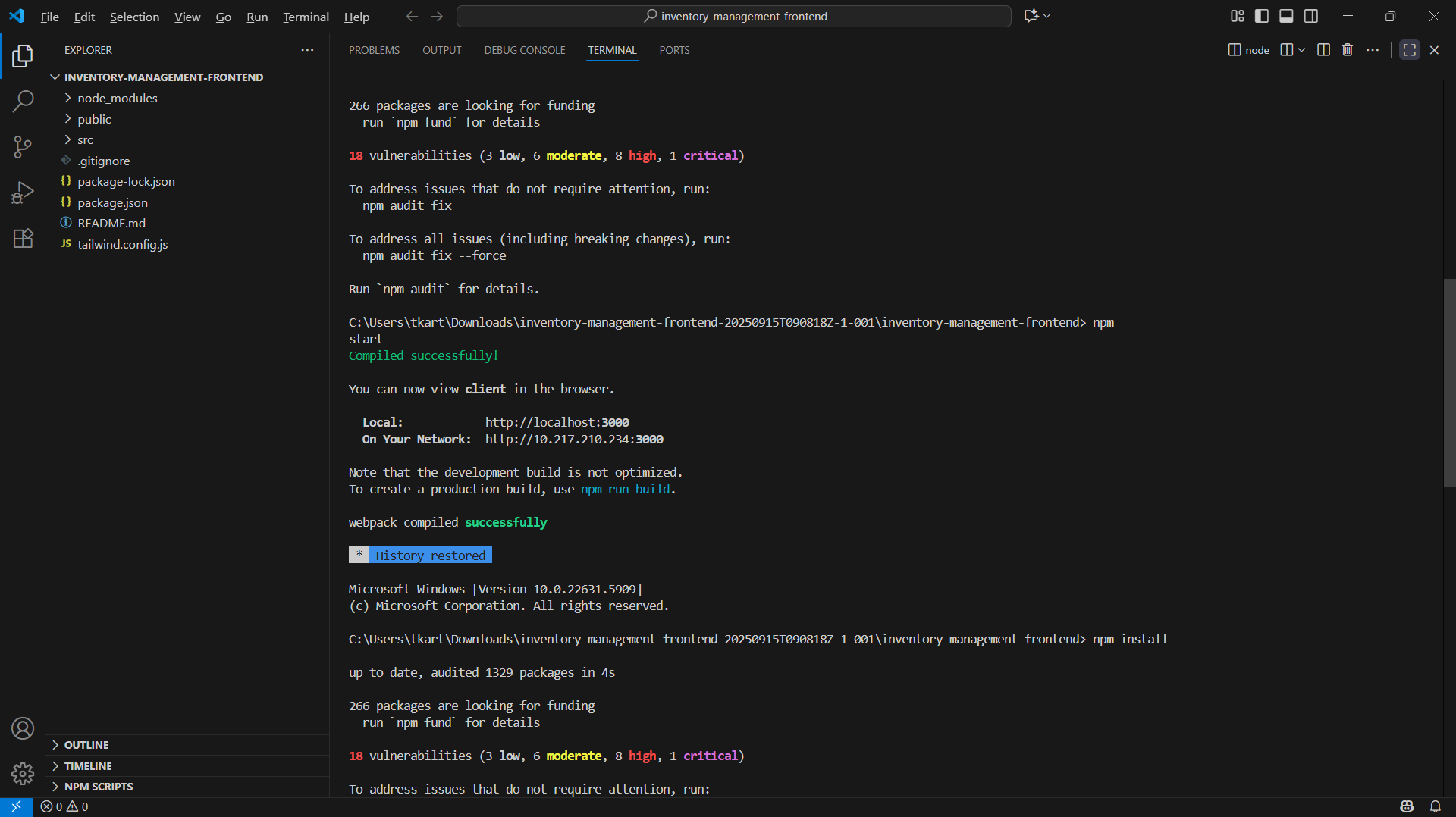
## ****Code Coverage****

* **User Management**: Register, login (JWT protected).
* **Products & Inventory**: Add, update, delete, stock alerts.
* **Sales & Purchase**: Entry and reports.
* **Integration**: React frontend ↔ Node.js backend ↔ MongoDB database.

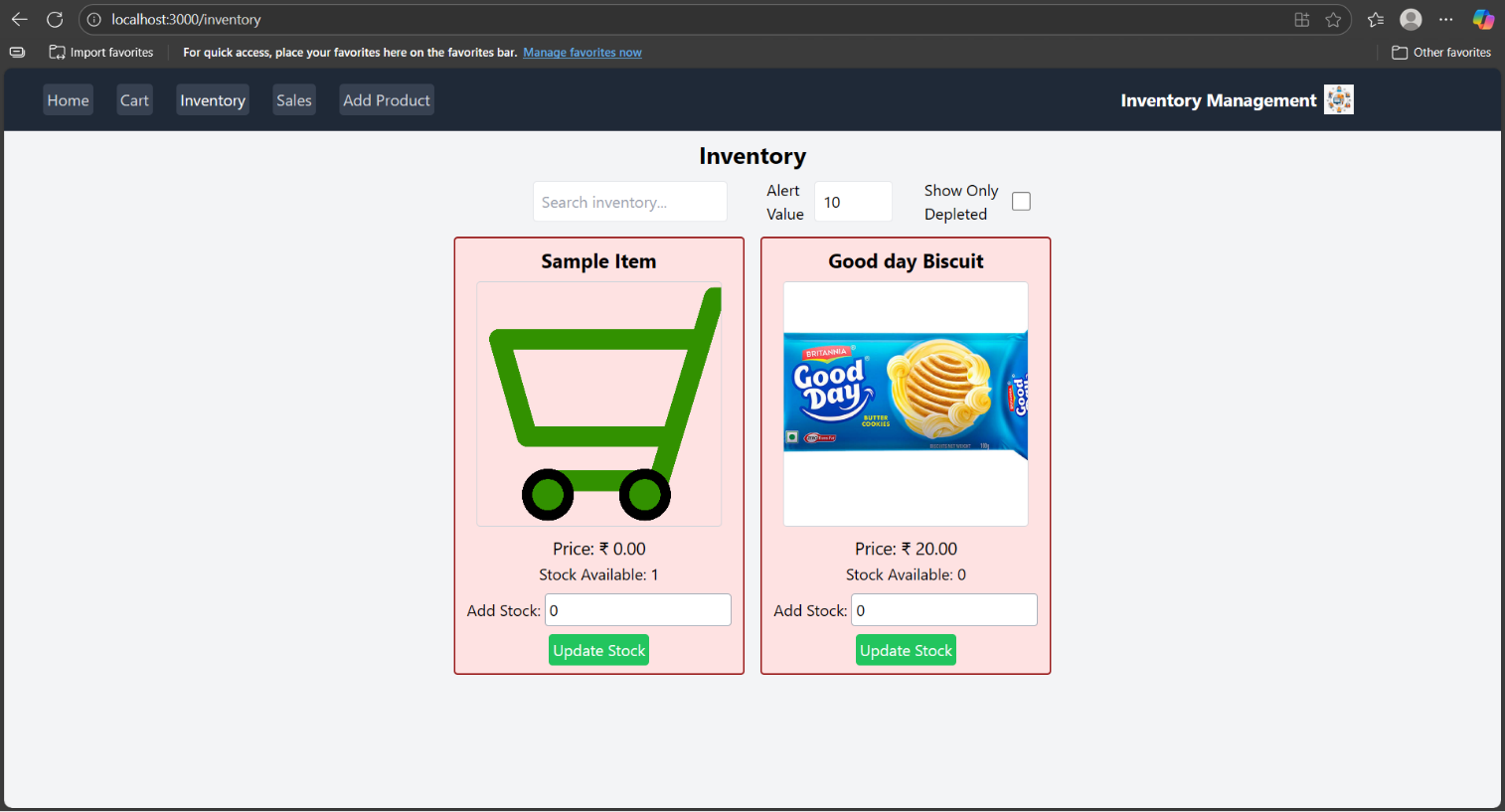
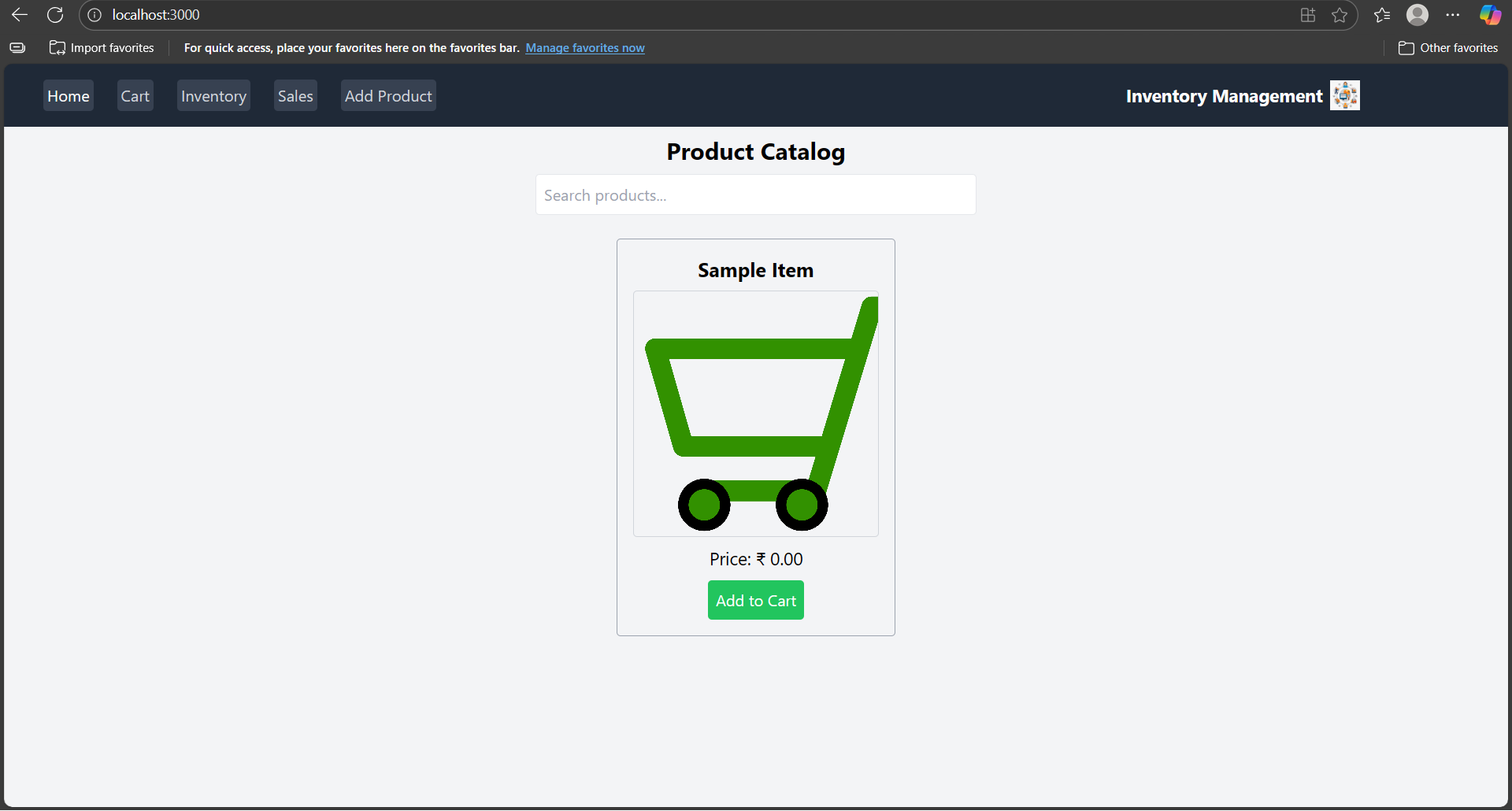
**11. Screenshots**

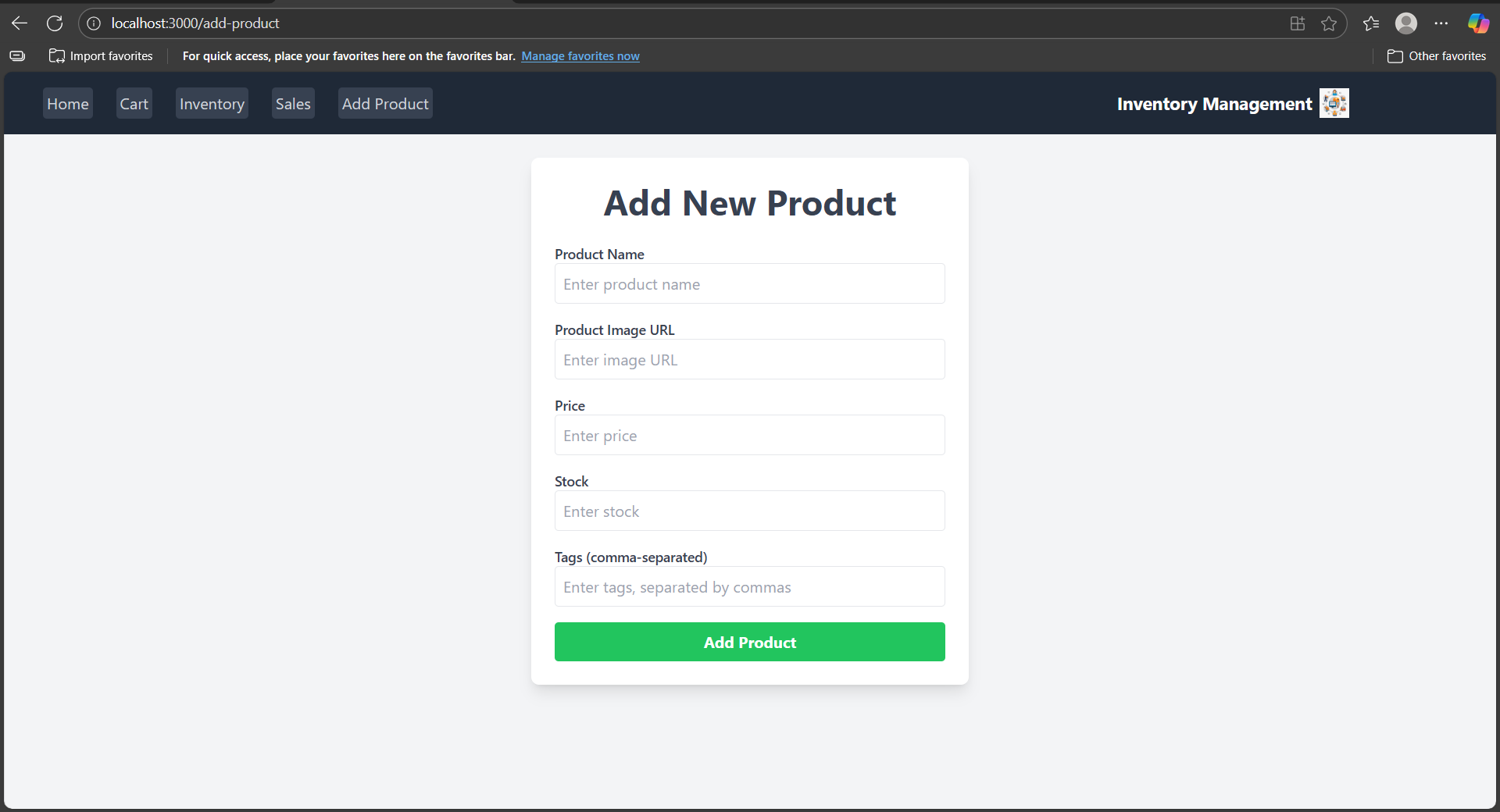
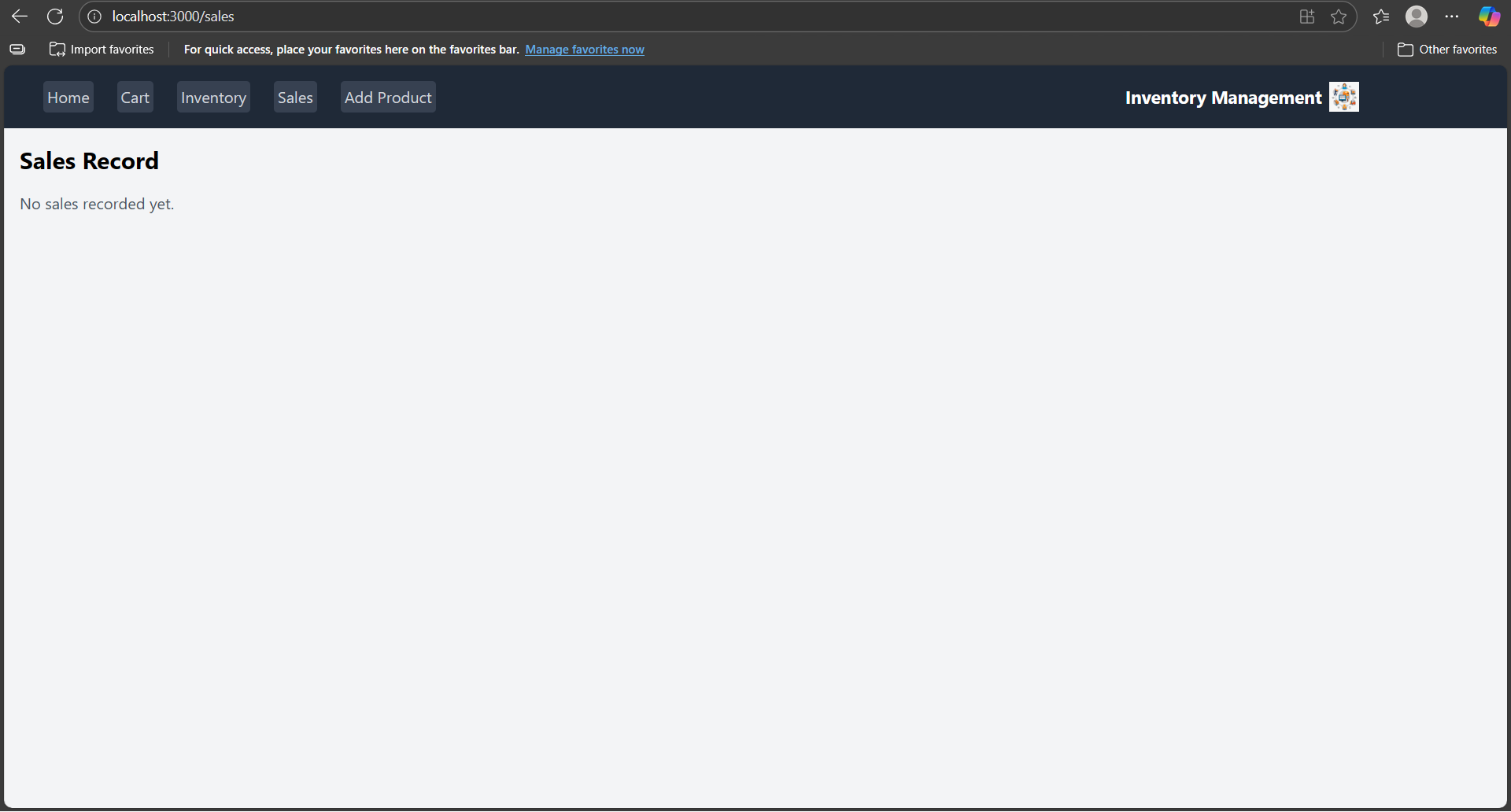
*Coding:*





*Output:*





## ****Known Issues & Future Enhancements****

**Known Issues**

* No barcode/QR support
* Basic sales and stock reports only
* No low-stock notifications via email/SMS
* Limited mobile optimization
* Simple error messages

**Future Enhancements**

* Add barcode/QR code support
* Provide advanced sales and inventory reports
* Enable email/SMS alerts for low stock
* Improve mobile-friendly UI
* Enhance error handling with user-friendly messages

1. **Demo video**

Video link:

[*https://drive.google.com/file/d/1cvlezxCmyW6FHhv4hD4DEBfRMA3s6xcY/view?usp=drive\_link*](https://drive.google.com/file/d/1cvlezxCmyW6FHhv4hD4DEBfRMA3s6xcY/view?usp=drive_link)