Project Documentation

# 1. Introduction

Project Title: **Store Manager: Inventory Management System**

Team ID: NM2025TMID47175

## Team Members & Roles

|  |  |
| --- | --- |
| Name & Email | Role |
| SATHYA V sathyasathyavenkatesh@gmail.com | Backend Development |
| PAVITHRA M pavithrapavi0763@gmail.com | Product Module & Documentation |
| GOPIKA P gopikasumanth@gmail.com | Sales Module & Video Recording |
| MITHRA A mithraa251@gmail.com | Store Module & Debugging |

# 2. Project Overview

The Inventory Management System is a web-based application designed to simplify business inventory operations. It enables efficient management of products, stores, purchases, and sales while ensuring real-time data tracking. The system provides insights into stock availability, reducing errors and improving efficiency.

## Key Features:

* Product Management – CRUD operations
* Store Management – Track multiple stores
* Purchase Management – Record purchases
* Sales Management – Record and track sales
* User Authentication – Secure JWT login/register
* Dashboard – Inventory overview
* Search & Filters – Locate data quickly
* Responsive Design – Multi-device support

# 3. Architecture

The system follows a three-tier architecture with frontend, backend, and database layers.

|  |  |
| --- | --- |
| Layer | Description |
| Frontend | React.js with Redux and Material UI for UI components. Provides dashboard, product, sales, and store interfaces. |
| Backend | Node.js with Express.js providing RESTful APIs, authentication middleware, and business logic. |
| Database | MongoDB storing users, products, stores, purchases, and sales records with scalability and flexibility. |

System Flow: Frontend → Backend → Database → Backend → Frontend

# 4. Setup Instructions

Prerequisites: Node.js v14+, Git, MongoDB, Visual Studio Code

1. Steps to Run:

* Clone the repository: git clone <repo-url>
* Backend setup: cd backend → npm install → npm start
* Frontend setup: cd ../frontend → npm install → npm start
* Open http://localhost:3000 in browser

# 5. Folder Structure

Inventory-Management-System/  
│  
├── Backend/  
│ ├── controller/  
│ │ ├── product.js  
│ │ ├── purchase.js  
│ │ ├── purchaseStock.js  
│ │ ├── sales.js  
│ │ ├── soldStock.js  
│ │ └── store.js  
│  
│ ├── models/  
│ │ ├── index.js  
│ │ ├── product.js  
│ │ ├── purchase.js  
│ │ ├── sales.js  
│ │ ├── store.js  
│ │ └── users.js  
│  
│ ├── router/  
│ │ ├── product.js  
│ │ ├── purchase.js  
│ │ ├── sales.js  
│ │ └── store.js  
│  
│ ├── node\_modules/  
│ ├── .gitignore  
│ ├── package.json  
│ ├── package-lock.json  
│ ├── seed.js  
│ └── server.js  
│  
├── Frontend/  
│ ├── node\_modules/  
│ ├── public/  
│ └── src/  
│ ├── components/  
│ ├── pages/  
│ └── redux/  
│  
├── package.json  
└── README.md

# 6. Running the Application

1. Start MongoDB service or connect to MongoDB Atlas.

2. Run backend using npm start inside backend folder.

3. Run frontend using npm start inside frontend folder.

4. Open http://localhost:3000 in browser.

# 7. API Documentation

|  |  |  |
| --- | --- | --- |
| Module | Endpoint | Description |
| User | POST /api/user/register | Register new user |
| User | POST /api/user/login | Authenticate user and return JWT |
| Product | GET /api/products | Retrieve products with filters |
| Product | POST /api/products | Add new product |
| Product | PUT /api/products/:id | Update product details |
| Product | DELETE /api/products/:id | Delete product |
| Store | GET /api/stores | Retrieve all stores |
| Store | POST /api/stores | Add new store |
| Store | PUT /api/stores/:id | Update store details |
| Store | DELETE /api/stores/:id | Remove a store |
| Purchase | POST /api/purchases | Record new purchase |
| Purchase | GET /api/purchases | View purchases with filters |
| Sales | POST /api/sales | Record new sale |
| Sales | GET /api/sales | Retrieve sales records |

# 8. Authentication

JWT-based authentication. Tokens are required in headers for protected routes. Middleware validates tokens.

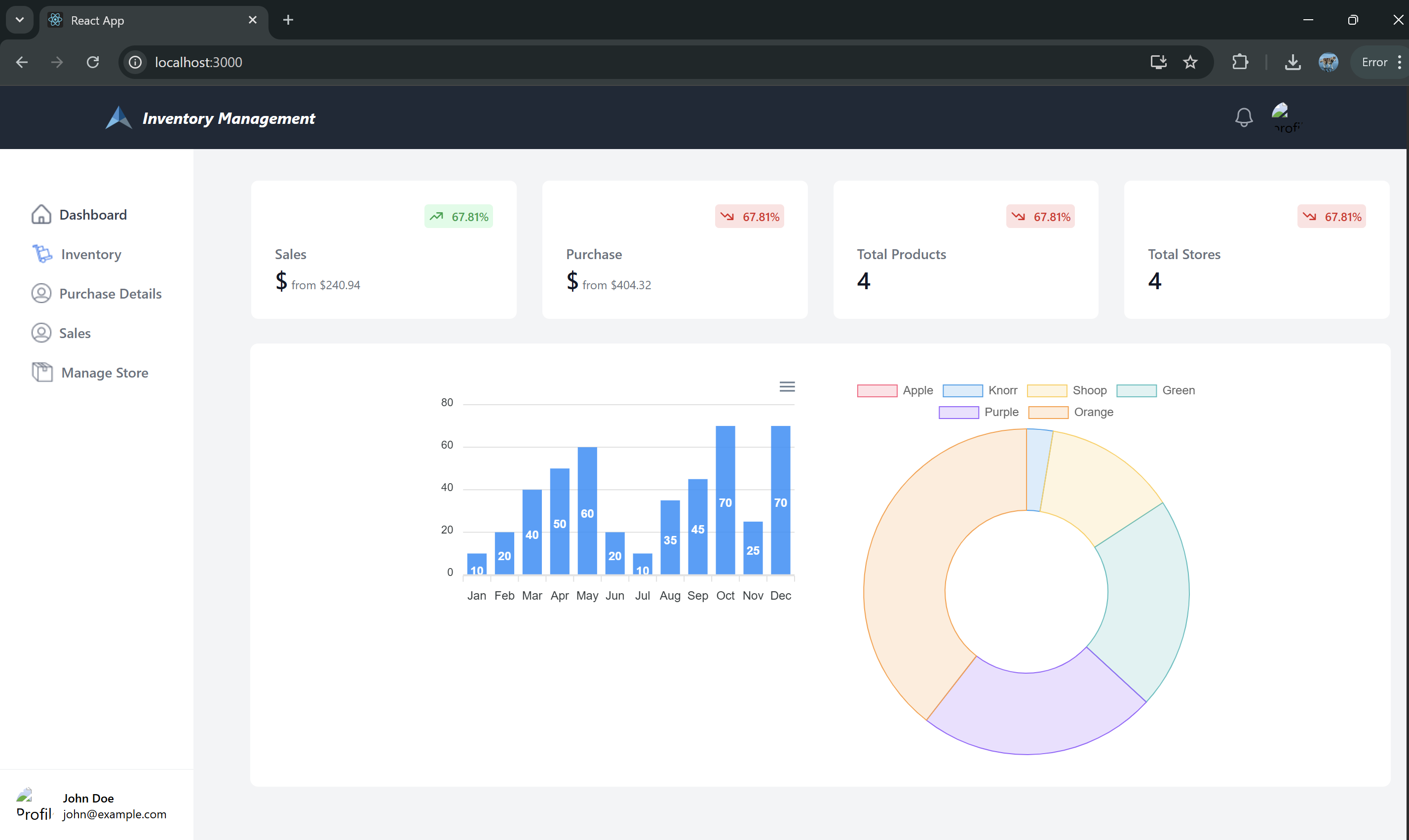
# 9. User Interface (UI Pages)

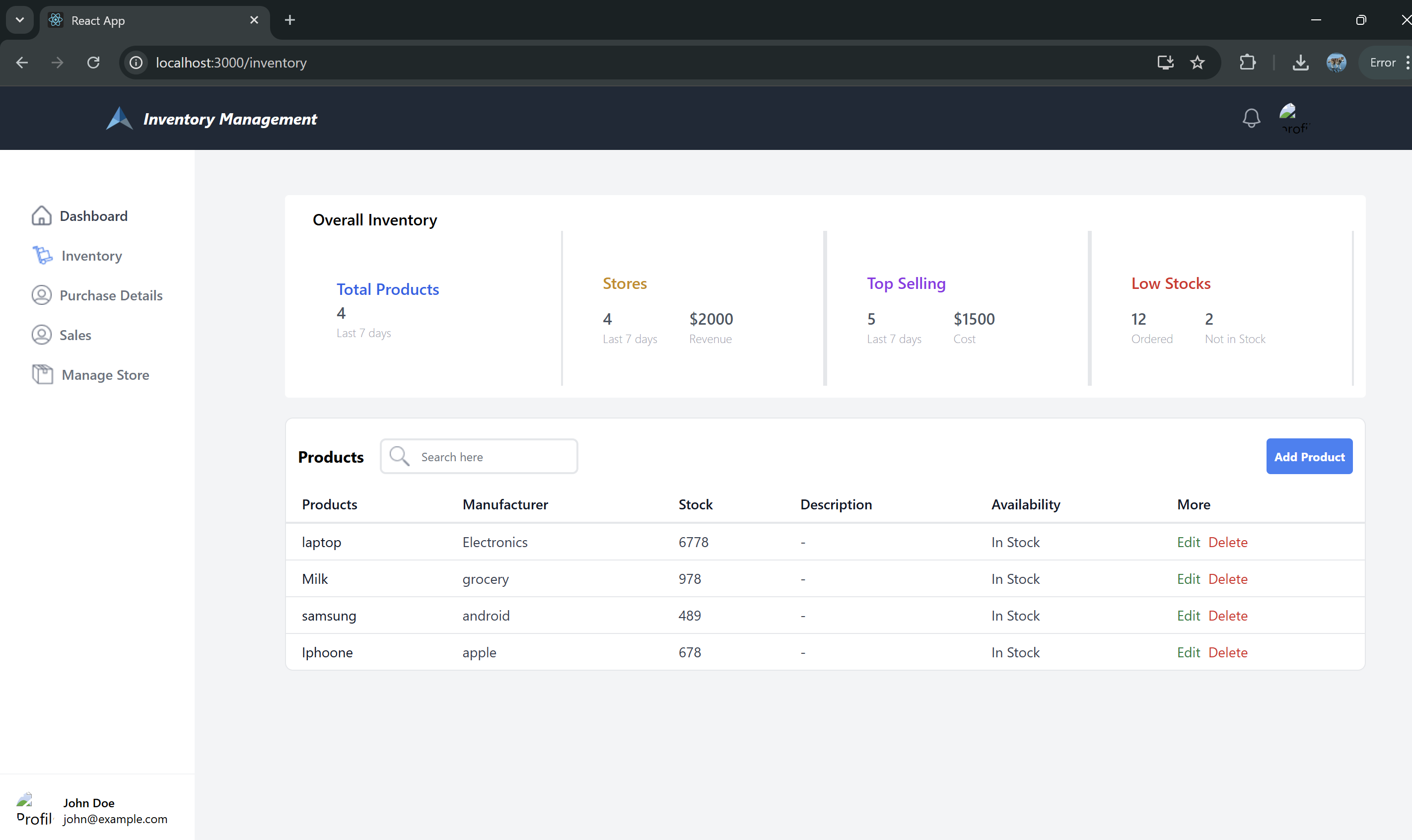
* Dashboard
* Product Management
* Store Management
* Purchase Management
* Sales Management
* Login/Register

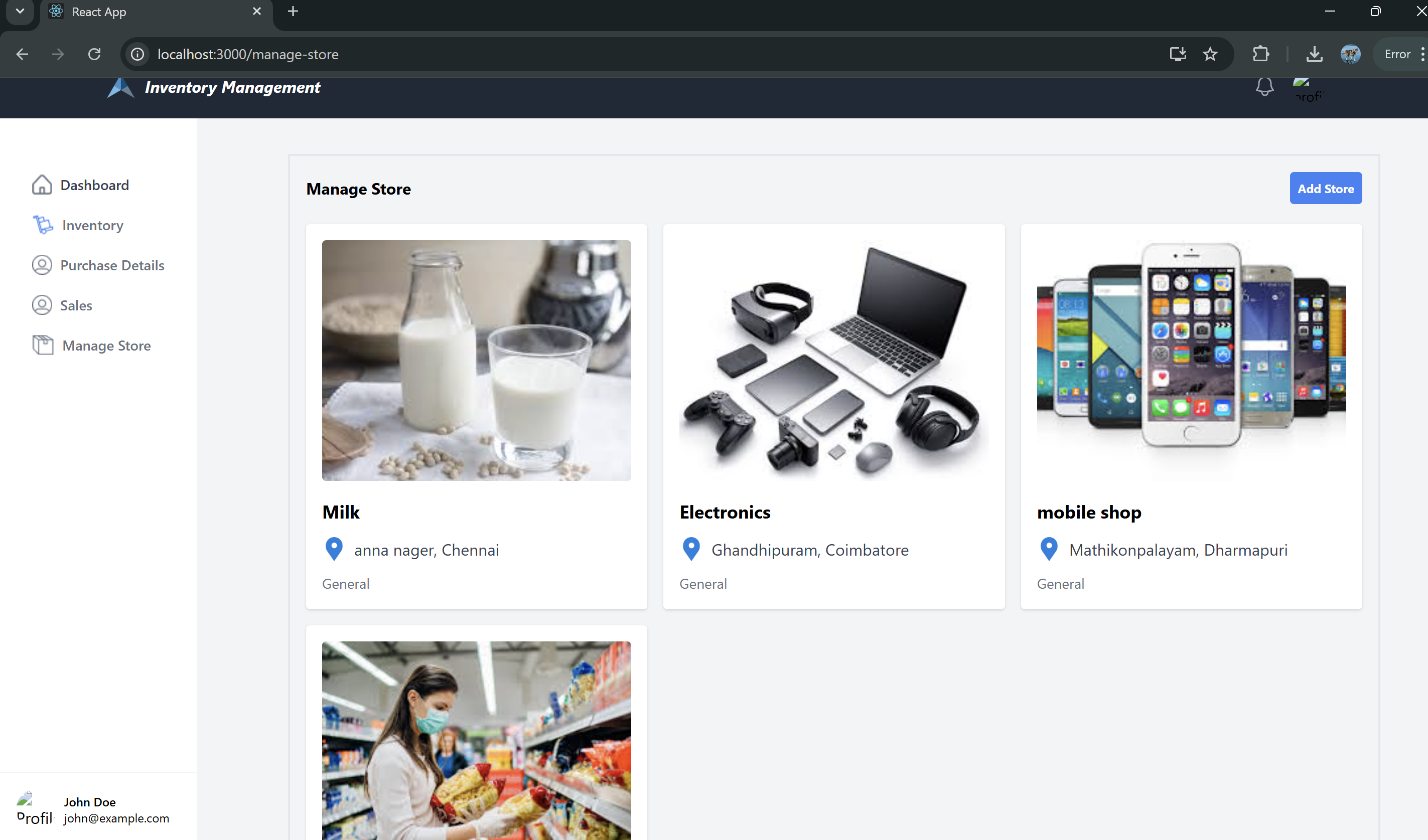
# 10. Testing

Manual testing + Postman for API validation. Debugging with DevTools.

# 11. Screenshots or Demo







# 12. Known Issues

* No live deployment yet
* Limited error handling
* No payment integration

# 13. Future Enhancements

* Live deployment
* Advanced reporting & analytics
* Barcode scanning
* Multi-role access
* Real-time notifications