ProjectDocumentation

Store Manager: Keep Track of Inventory

# Introduction

## ProjectTitle: Store Manager

* + **TeamID:** NM2025TMID47750
  + **TeamLeader:** M.M.Dharshini & [mdharshini835@gmail.com](mailto:mdharshini835@gmail.com)

## TeamMembers:

## M.Diviya & [diviyam2007@gmail.com](mailto:diviyam2007@gmail.com)

## S.Divya & [divyasd25522006@gmail.com](mailto:divyasd25522006@gmail.com)

## P.Jayapriya & [jayapriyaj653@gmail.com](mailto:jayapriyaj653@gmail.com)

# ProjectOverview

* + **Purpose:** Ensure that the business runs smoothly, efficiently, and profitably. This role bridges the gap between upper management and frontline employees, focusing on customer satisfaction, staff performance, and operational excellence.

## Features:

* + - **Inventory Management**
    - **Stock Updates**
    - **Cart**
    - **Checkout at Cart**
    - **Adding New Products to Inventory**
    - **Search Functionality for Products**

# Architecture

* + **Frontend:** In a **retail store**, the **frontend** refers to the **customer-facing area**—usually the **entrance, sales floor, checkout area, and customer service desk**. A store manager’s role here is critical.
  + **Backend:** In a **physical store**, the **backend** refers to **behind-the-scenes operations** that customers don’t see, but that are crucial for running the store efficiently.

**Database:** A **database** is a structured collection of data that supports day-to-day operations. It stores everything from **product details to employee records**.

The **store manager doesn't usually manage the raw database directly**, but interacts with it through applications.

# SetupInstructions

## Prerequisites:

* **Node.jsandnpm**:
* **React.js**
* **HTML, CSS, and JavaScript**
* **Version Control**
* **Development Environment**

• Installation Steps:

* + • Add tailwindcss, react, react-dom, and other necessary packages.

A screen shot of a computer program

Description automatically generated

A black screen with white text

Description automatically generated

1. Folder Structure

store-manager/

├── 📁 config/ # Configuration files (DB, environment, etc.)

│ └── database.js

│ └── app-config.js

│

├── 📁 controllers/ # Handle user input and app logic

│ └── productController.js

│ └── userController.js

│ └── orderController.js

│

├── 📁 models/ # Database models (ORM/ODM schemas)

│ └── productModel.js

│ └── userModel.js

│ └── orderModel.js

│

├── 📁 services/ # Business logic (e.g., stock management)

│ └── inventoryService.js

│ └── authService.js

│

├── 📁 routes/ # Application routes / endpoints

│ └── productRoutes.js

│ └── userRoutes.js

│

├── 📁 middleware/ # Authentication, logging, validation

│ └── authMiddleware.js

│ └── errorHandler.js

│

├── 📁 utils/ # Utility functions (formatters, validators)

│ └── logger.js

│ └── validator.js

│

├── 📁 public/ # Static files (images, CSS, client JS)

│

├── 📁 views/ # UI templates (for web apps, if not API-only)

│ └── dashboard.ejs

│

├── 📁 tests/ # Unit and integration tests

│ └── product.test.js

│

├── 📁 scripts/ # CLI tools or DB seed scripts

│ └── seedProducts.js

│

├── 📄 .env # Environment variables

├── 📄 package.json # Project metadata & dependencies (Node.js)

├── 📄 README.md # Project documentation

└── 📄 server.js / app.js # Entry point of the application

5. Folder Structure  
A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

# RunningtheApplication

## Frontend:

The **frontend** is the **visual, interactive part** of the app that a **store manager uses** in their daily tasks. It's usually a **web app**, **desktop app**, or **mobile app** built using technologies

**Backend:**

The **backend** is the **behind-the-scenes part** of the application that handles:

* Business logic (e.g. calculating totals, enforcing rules)
* Data storage (e.g. inventory, employees, sales records)
* APIs that the frontend (UI) communicates with
* Authentication, security, permissions
* Reporting and analytics processing
  + **Access:**Visit[http://localhost:3000](http://localhost:3000/)

# APIDocumentation

## User:

* + - https://api.storemanager.com/v1/
    - [mdharshini835@gmail.com/tE6VXl0x](mailto:mdharshini835@gmail.com/tE6VXl0x)

## Projects:

* **Step 1: Create context files for Inventory, Cart, and Sales.**
  + Use createContext() and define default states.
* **Step 2: Define reducers for state management.**
  + Inventory Reducer: Handle stock addition, updates, and sales.
  + Cart Reducer: Manage adding/removing items and quantities.
  + Sales Reducer: Track sales records.
* **Step 3: Set up ContextProvider components.**
  + Wrap the app in these providers in index.js or App.js.
* **Step 4: Persist data in localStorage.**
  + Save and load context states to/from localStorage.

**Milestone 3 - Developing Core Components**

* **Step 1: Build the Inventory Component.**
* Display a list of products with stock details.
* Implement search and alert value input.
* **Step 2: Build the Product Component.**
* Show product details, including an image, price, and stock.
* Add functionality to update stock.
* **Step 3: Build the Cart Component.**
* Display items added to the cart with quantity and total value.
* Add buttons for incrementing, decrementing, and removing items.
* Include checkout functionality to update inventory and create sales records.
* **Step 4: Build the Sales Component.**
* Display a list of sales records with details like date, cart items, and total sale value.
* Sort records from the latest to the oldest.

# Authentication

* + JWT-**JWT Authentication** is ideal for secure and stateless access.
  + app.post("/products", authMiddleware, productController.addProduct);

# UserInterface

* + Admin
  + Store Staff
  + Warehouse Team
  + Accountant

# Testing

At each milestone (e.g., “Product CRUD complete”, “Authentication implemented”, “Orders working”), you manually verify:

Functional correctness

API responses

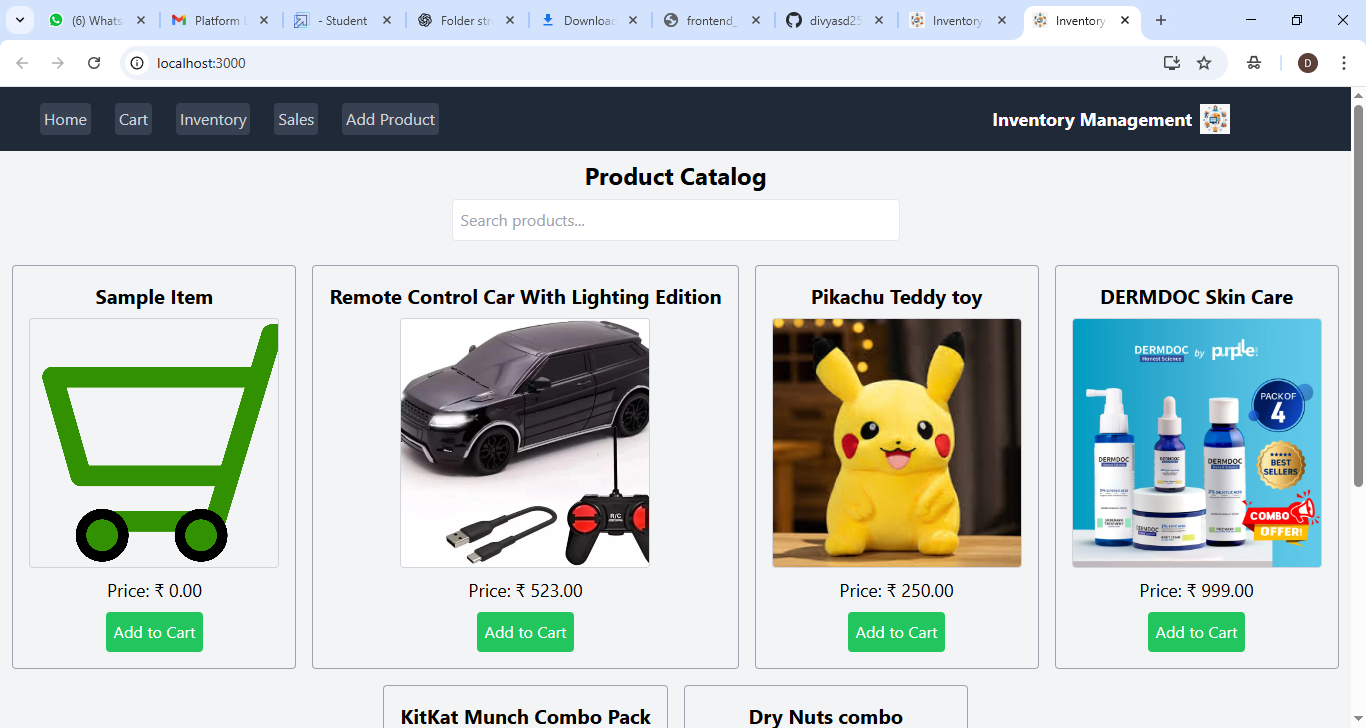
UI behavior

Data flow

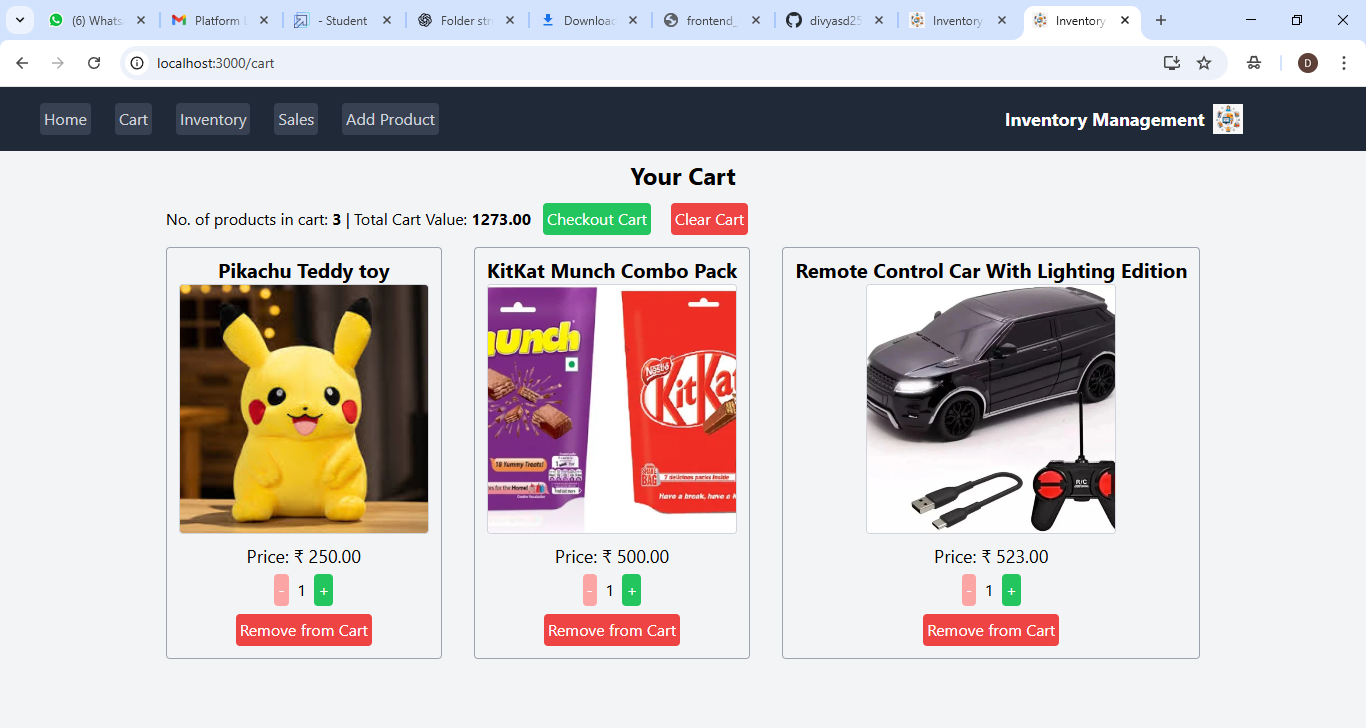
Error handling

* + Tools:Postman,ChromeDevTools

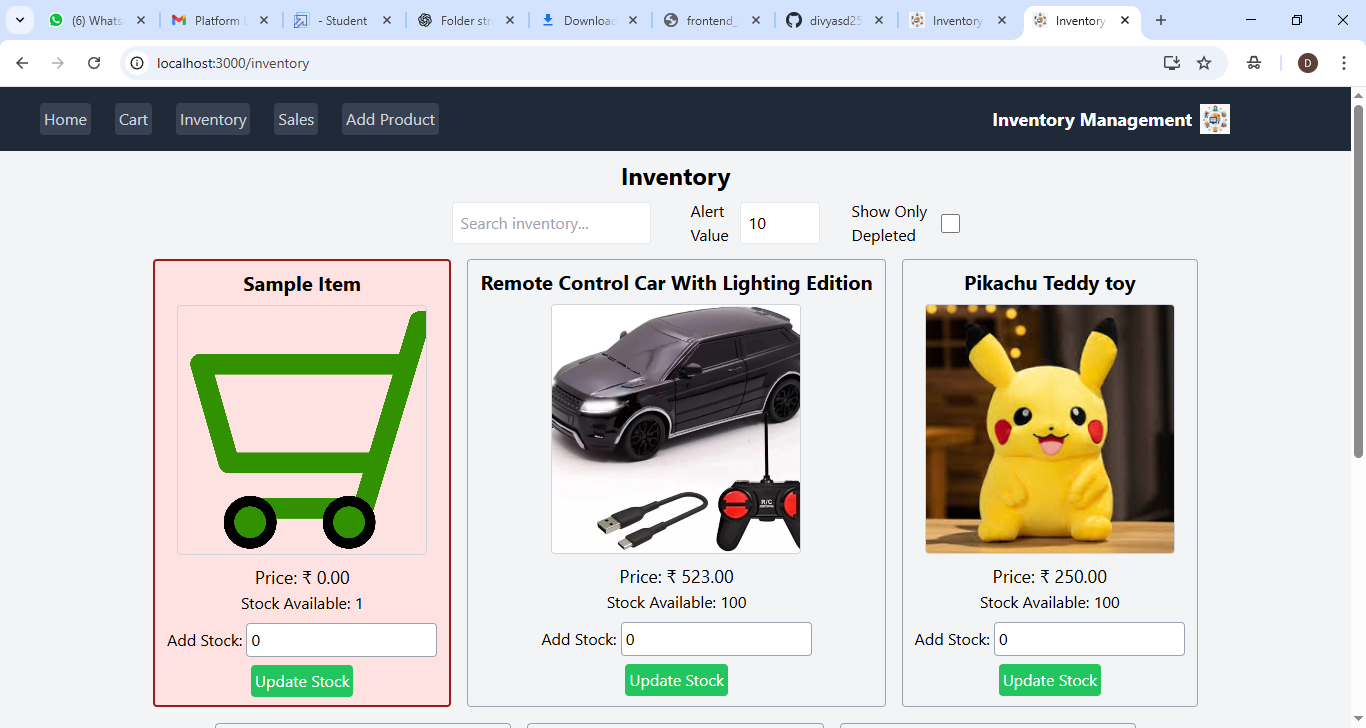
1. Demo Link: <https://drive.google.com/file/d/1i_BNxMjY5DM2irPCnzyKCRlwql__Kn7d/view?usp=sharing>
2. **Product Catalog:**



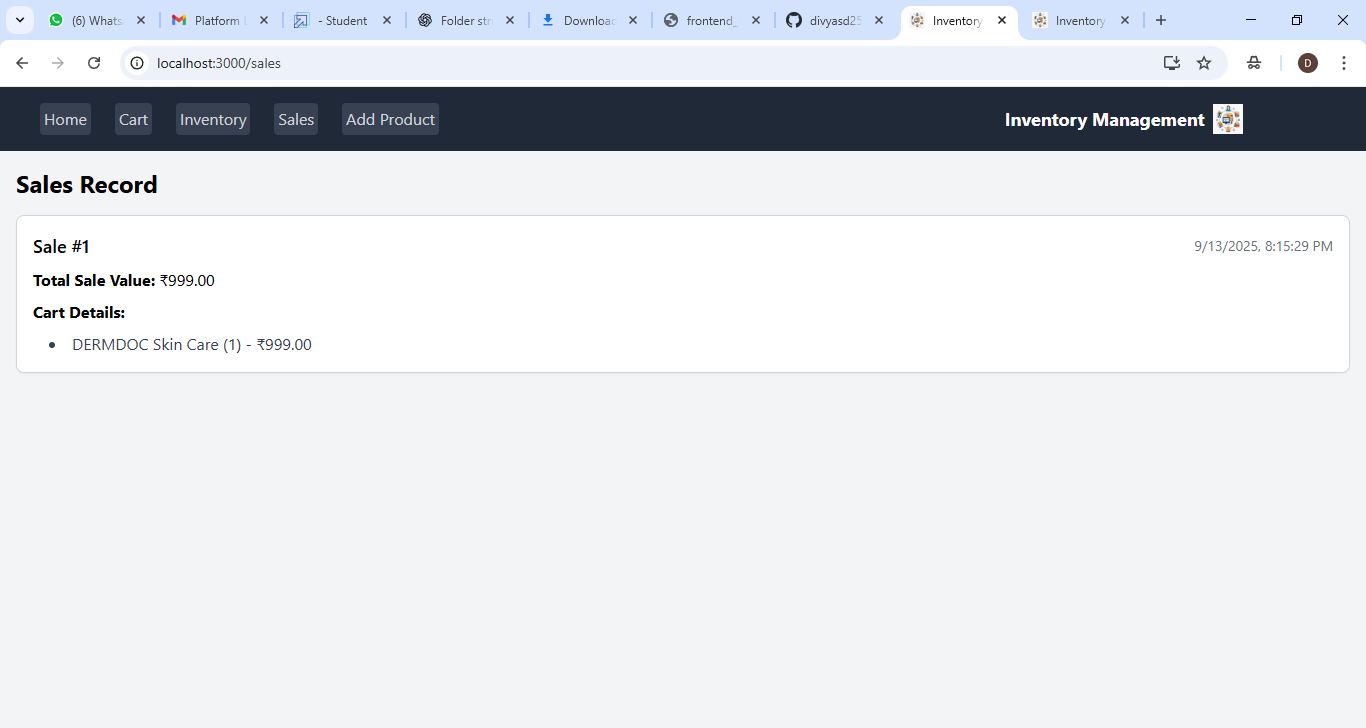
Cart:

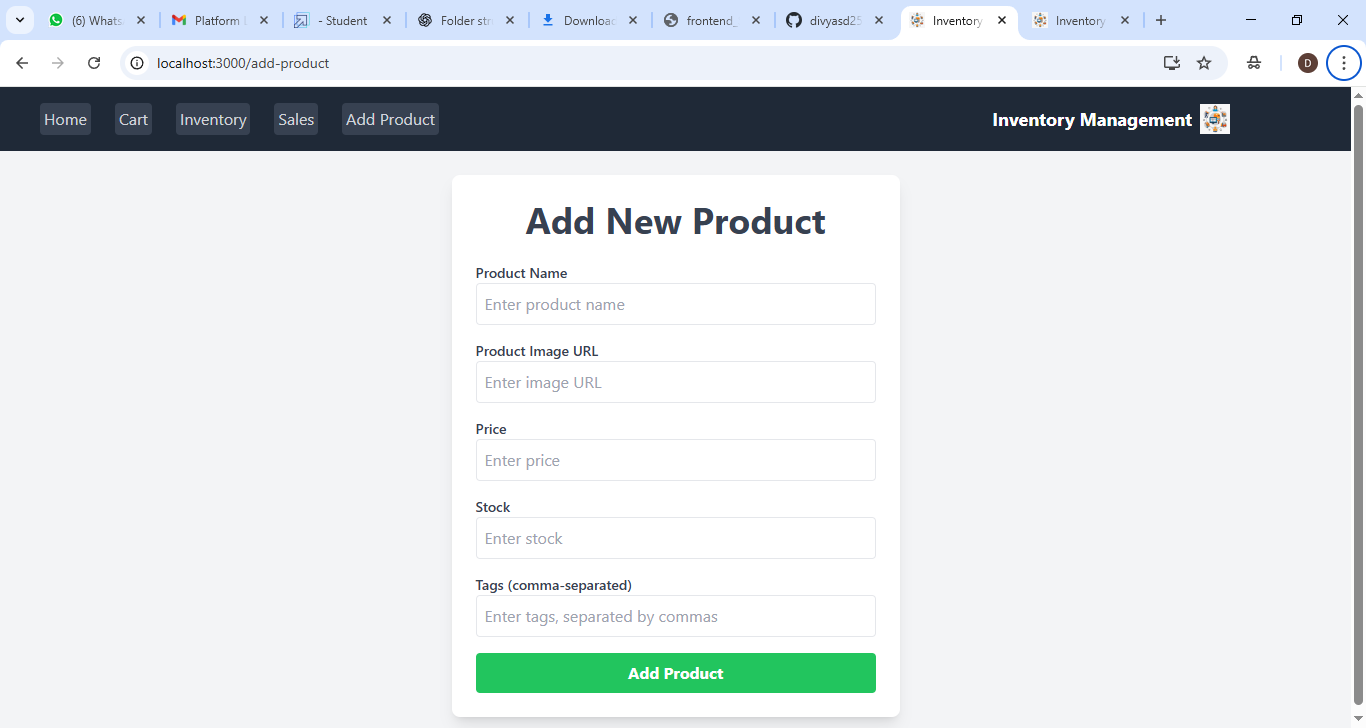
****

**Inventory:**

****

**Sales:**

****

****