Project Documentation: Inventory Document

# Introduction

Project Title: Store Manager: Keep Track of Inventory

Team Members: [SRINIVASAN.C, KAMESH.M, JEFF KALLIS.J,  RAGUL PRASHAD.D]

# Project Overview

Purpose: This project is a frontend application built with React.js to provide a modern, responsive, and efficient user interface.

Features:

- Responsive UI built with React and Tailwind CSS

- Component-based architecture

- State management using React hooks

- Routing support with react-router

# Architecture

Component Structure: The application is structured into reusable components under the 'src' folder.

State Management: Local state is managed using React hooks. Future scope may include Context API or Redux for global state.

Routing: react-router can be used to handle multiple routes in the application.

# Setup Instructions

Prerequisites: Ensure Node.js and npm are installed.

Installation:

1. Clone the repository

2. Navigate to the project folder

3. Run `npm install` to install dependencies

4. Run `npm start` to start the development server

# Folder Structure

The project folder contains the following files and directories:

- node\_modules/ : Contains project dependencies  
- public/ : Public assets (HTML, images, icons)  
- src/ : Source code (components, pages, hooks)  
- .gitignore : Git ignore file  
- package.json : Project configuration and dependencies  
- package-lock.json : Dependency lock file  
- README.md : Project documentation  
- tailwind.config.js : Tailwind CSS configuration

# Running the Application

Use the following command to run the application locally:

npm start

# Component Documentation

Key Components:

- App.js: Main entry component

- Reusable components in 'src/components/'

- Pages located in 'src/pages/'

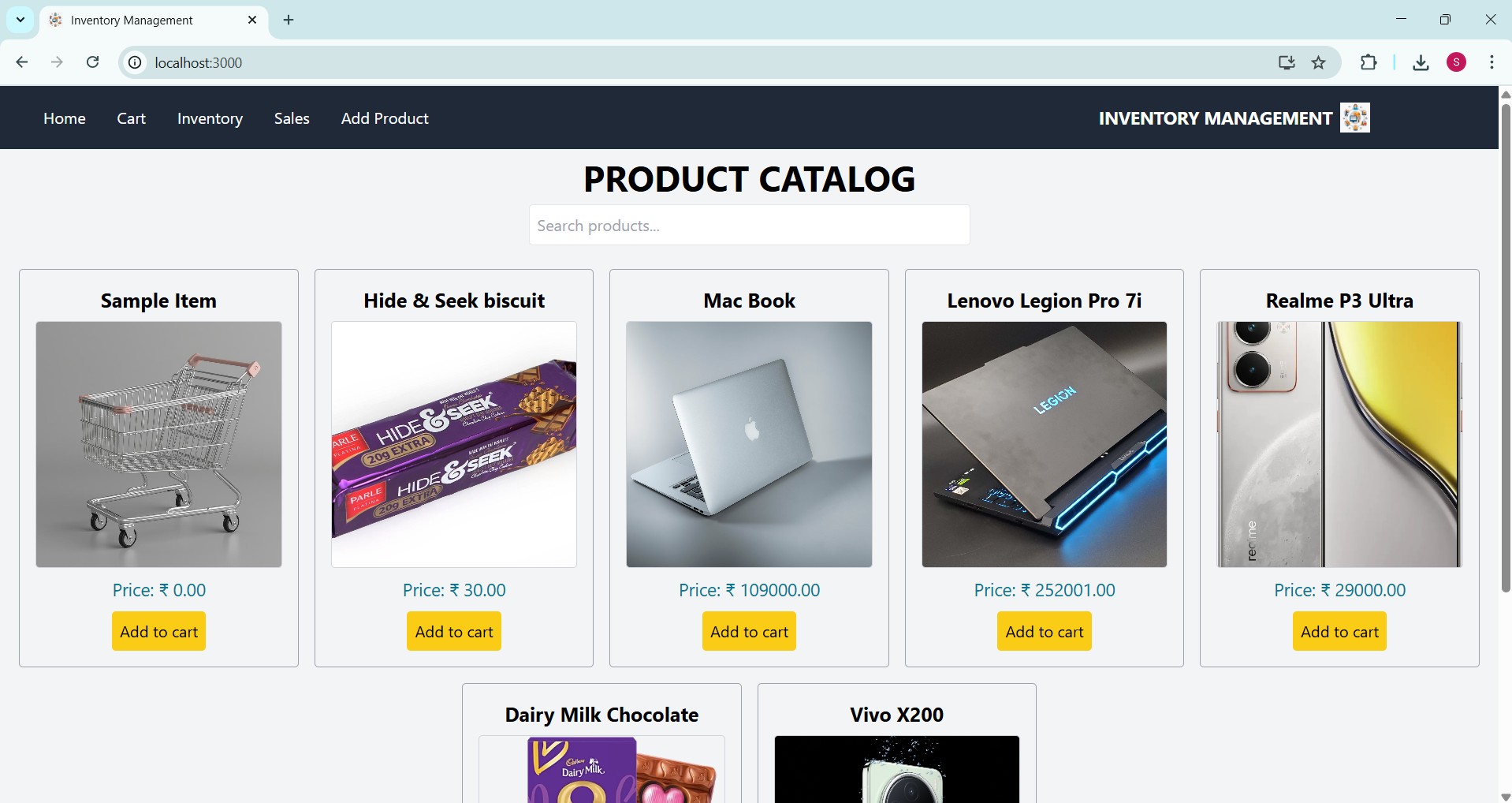
# Styling

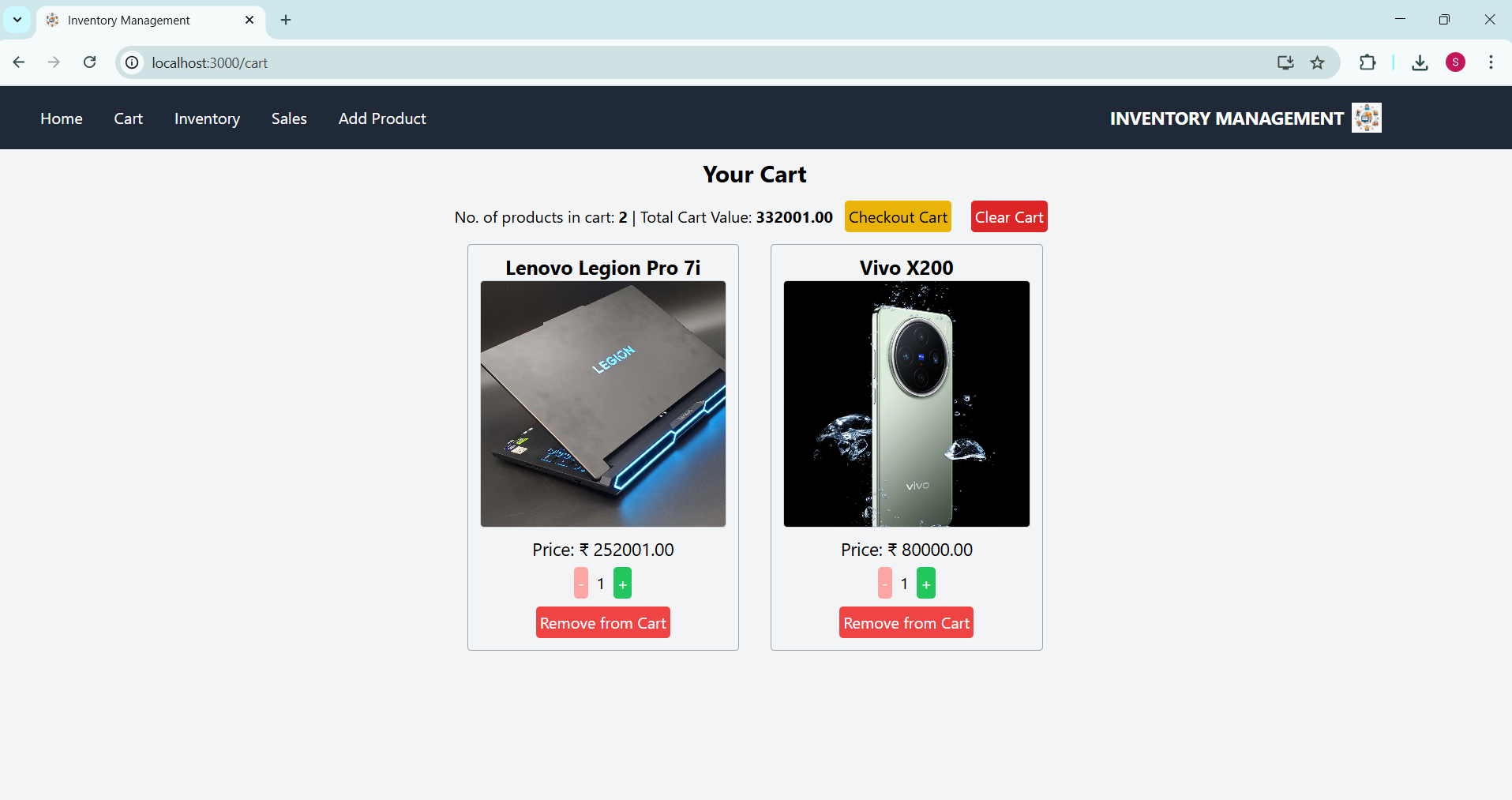
The application uses Tailwind CSS for styling and utility-first design. The configuration is managed in tailwind.config.js.

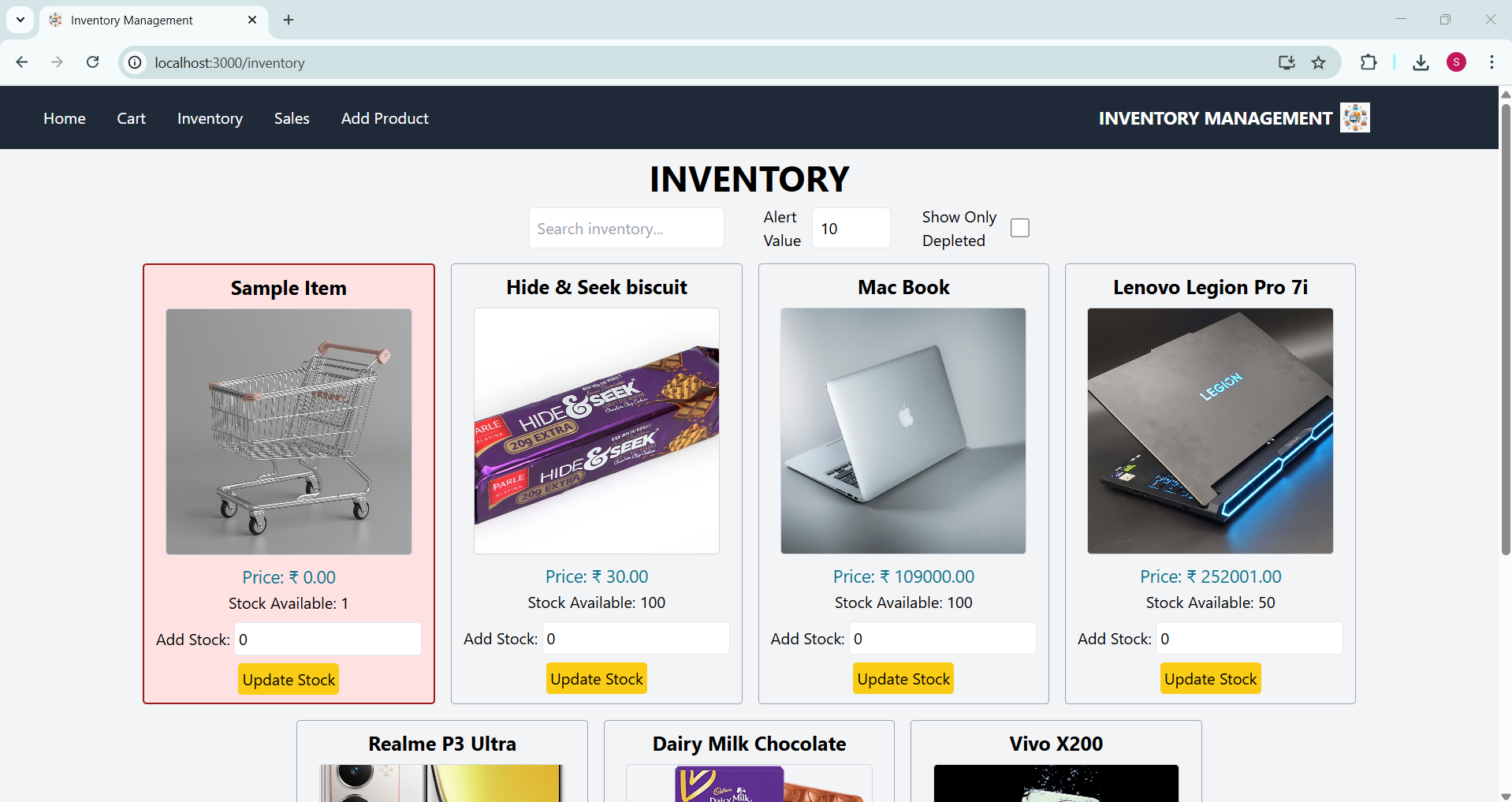
# Testing

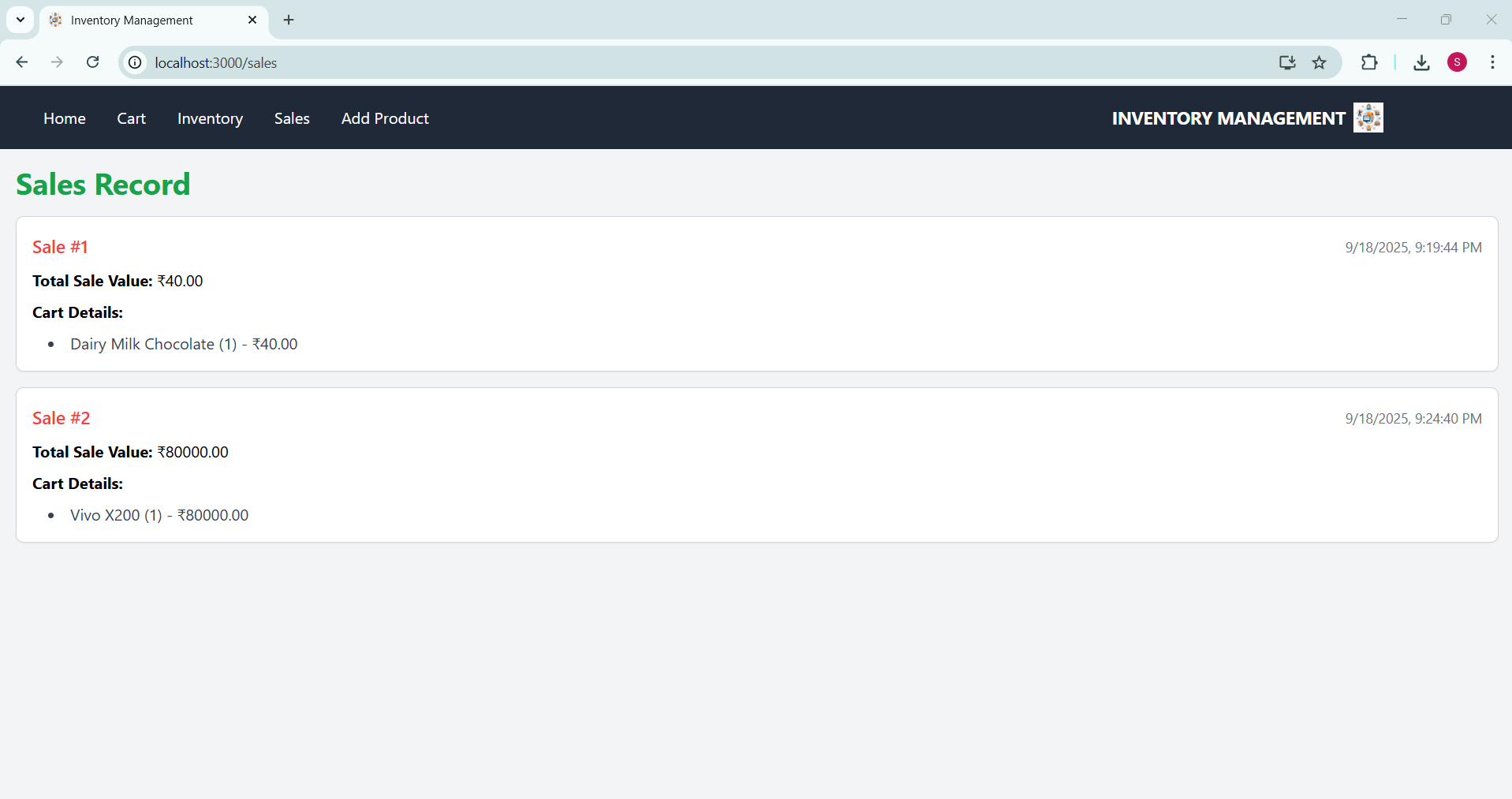
Testing Strategy: Components can be tested using Jest and React Testing Library.

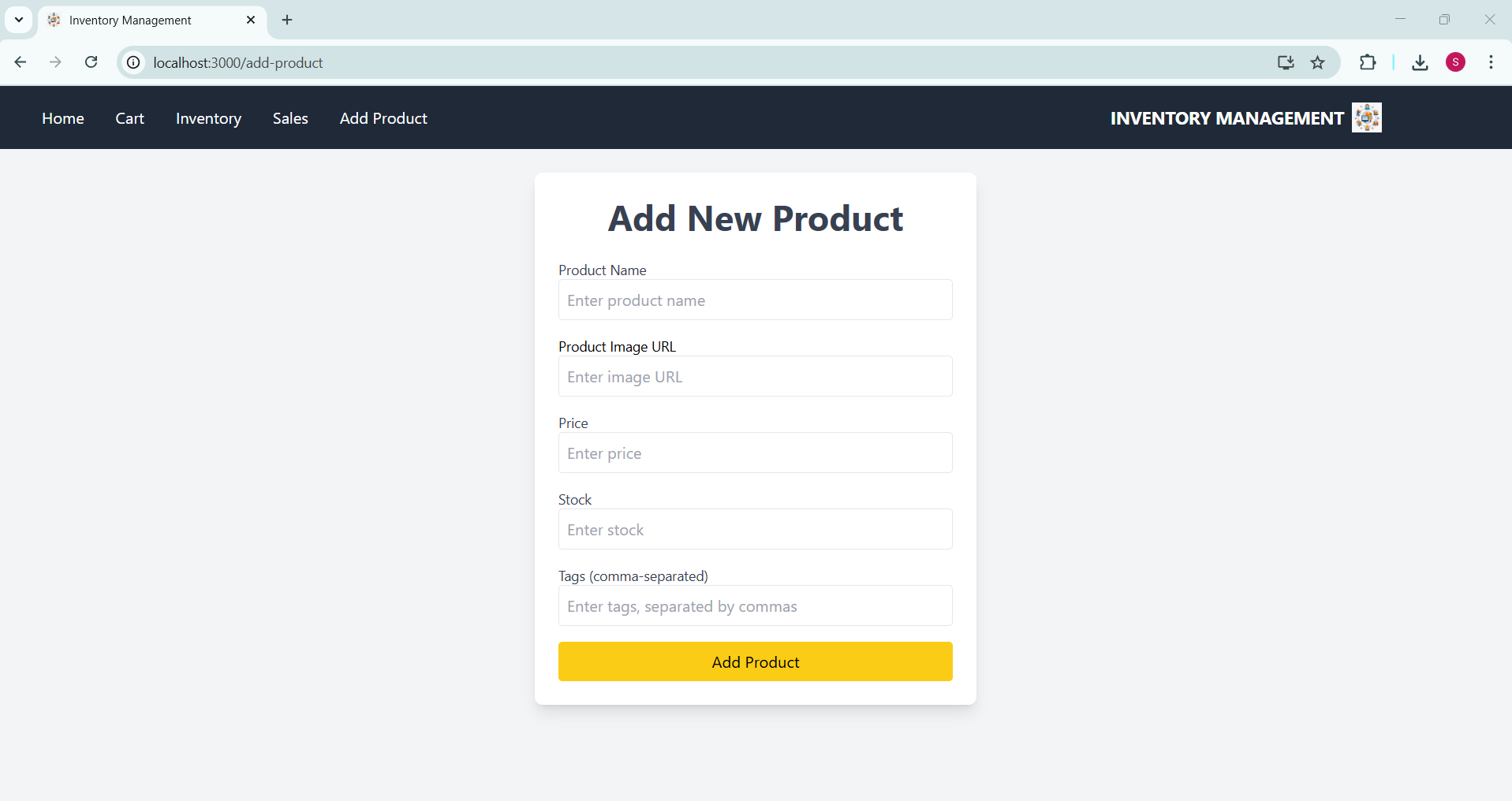
# Screenshots or Demo











# Known Issues

No known issues at the current stage.

# Future Enhancements

- Add global state management with Redux or Context API

- Improve unit and integration test coverage

- Enhance UI/UX with animations