**📘 Frontend Development with React.js**

**Project Documentation**

**1. Introduction**

* **Project Title**: **Store Manager – Keep Track of Inventory**
* **Team ID:NM2025TMID40300**
* **Team Leader**: Joshua Baskar PG & joshuabaskar106@gmail.com
* **Team Members**:
  + **Hariharan S** & eswariraj376@gmail.com
  + **Arun Kumar V** & arunbhuvan99@gmail.com
  + **Rohit Kumar K** & krohitkumar200708@gmail.com

**2. Project Overview**

* **Purpose**:  
  The **Store Manager** application helps businesses manage and track inventory through an intuitive dashboard built with React. It allows for efficient product listing, categorization, and updates.
* **Features**:
  + Product listing interface
  + Responsive layout with Tailwind CSS
  + Modular component structure
  + Routing for page navigation

**3. Architecture**

* **Component Structure**:
  + App.js: Root component housing routing and layout
  + Navbar, Sidebar: Shared layout components
  + ProductList, AddProduct, EditProduct: Functional pages/components
* **State Management**:  
  Currently uses **React’s built-in useState/useEffect** for local state. (No Redux/ContextAPI used as of latest commit)
* **Routing**:  
  Powered by **React Router DOM** – handles navigation between pages like Home, Products, Add Product, etc.

**4. Setup Instructions**

* **Prerequisites**:
  + Node.js (v14+)
  + npm
* **Installation**:
* git clone https://github.com/24au212405121-hue/str\_jb-project.git
* cd str\_jb-project
* npm install
  + To start development server:
  + npm start

**5. Folder Structure**

Store-Inventory-Manager/

|-- client/ # React frontend |

|-- components/ |

|-- AddProduct.jsx | |

|-- Cart/ | |

|-- Cart.jsx | |

|-- CartItem.jsx | |

|-- Inventory/ | |

|-- Inventory.jsx | |

|-- Product.jsx |

|-- NavBar.jsx | |

|-- Product/ | |

|-- Product.jsx | |

|-- ProductCatalog.jsx | |

|-- ProductList.jsx |

|-- Sales/ | |

|-- Sales.jsx | |

|-- SaleRecord.jsx |

|-- context/ | |

|-- CartContext.js | |

|-- InventoryContext.js | |

|-- SalesContext.js | |

-- hooks/ | |

|-- useLocalStorage.js |

|-- pages/ | |

|-- NotFound.jsx |

|-- App.css |

|-- App.js |

|-- App.test.js |

|-- index.css |

|-- index.js |

|-- reportWebVitals.js |

|-- setupTests.js |

|-- .gitignore |

|-- package.json |

|-- package-lock.json |

|-- README.md |

|-- tailwind.config.js

|-- server/ # Node.js backend (placeholder, not implemented)

**6. Running the Application**

* To run the frontend server:
* npm start

**7. Component Documentation**

* **Key Components**:
  + App.js: Entry component, contains routes
  + ProductList.js: Displays all inventory items
  + AddProduct.js: Form to add a new item
  + EditProduct.js: Form to update existing item
* **Reusable Components**:
  + Button.js, InputField.js, etc. (if implemented under components/)

**8. State Management**

* **Global State**:  
  None implemented yet.
* **Local State**:
  + Managed via useState() for form data, product listing, etc.
  + useEffect() used for initial rendering and API/data mocking (if applicable)

**9. User Interface**

* **UI Features**:
  + Minimalist UI using Tailwind CSS
  + Responsive design (mobile + desktop)
  + Navigation via navbar/sidebar

*(Screenshots can be added here once available)*

**10. Styling**

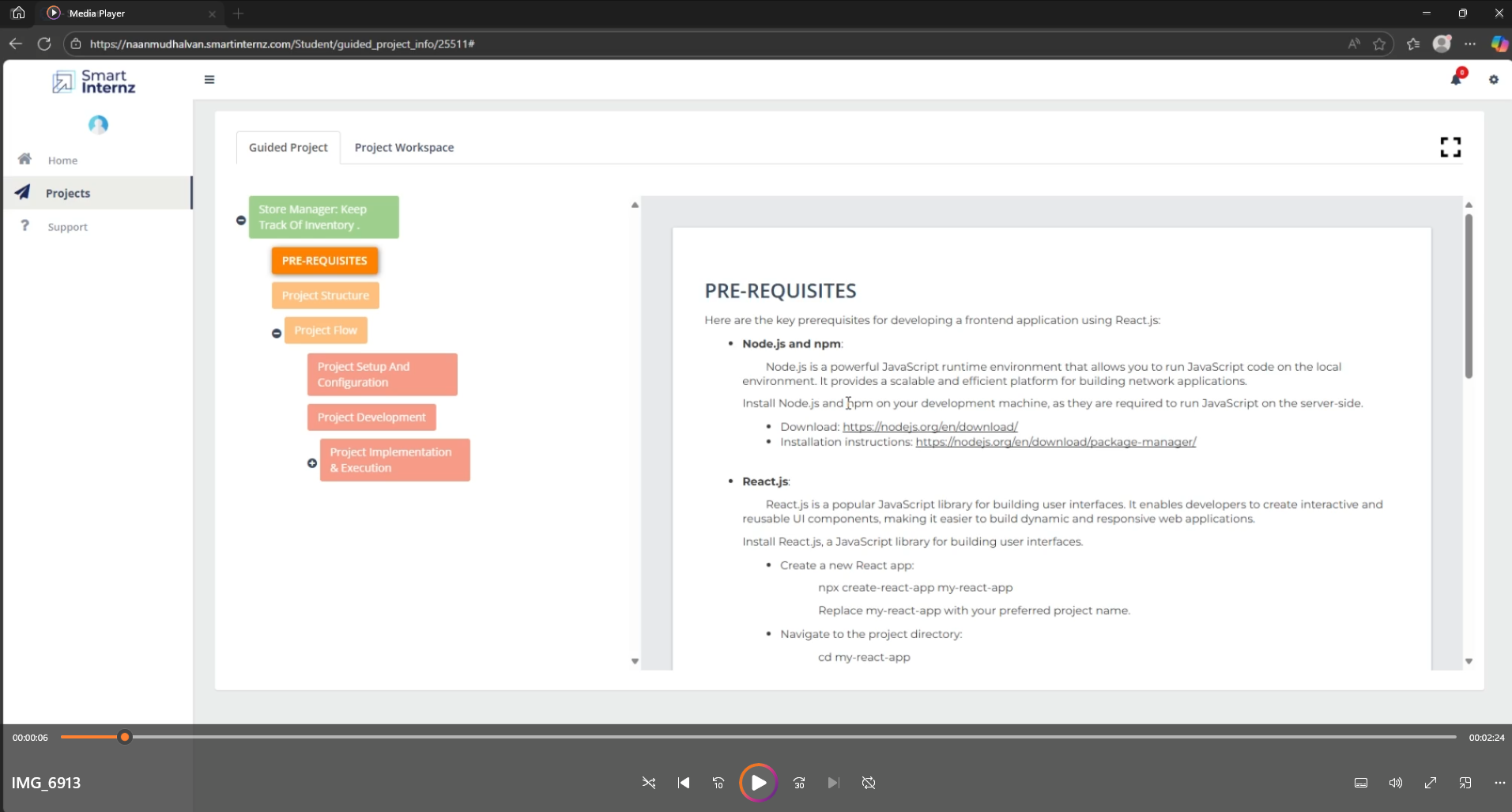
* **CSS Framework**:
  + [✔] Tailwind CSS (configured via tailwind.config.js)
* **Theming**:
  + Not implemented. Currently uses default Tailwind utility classes.

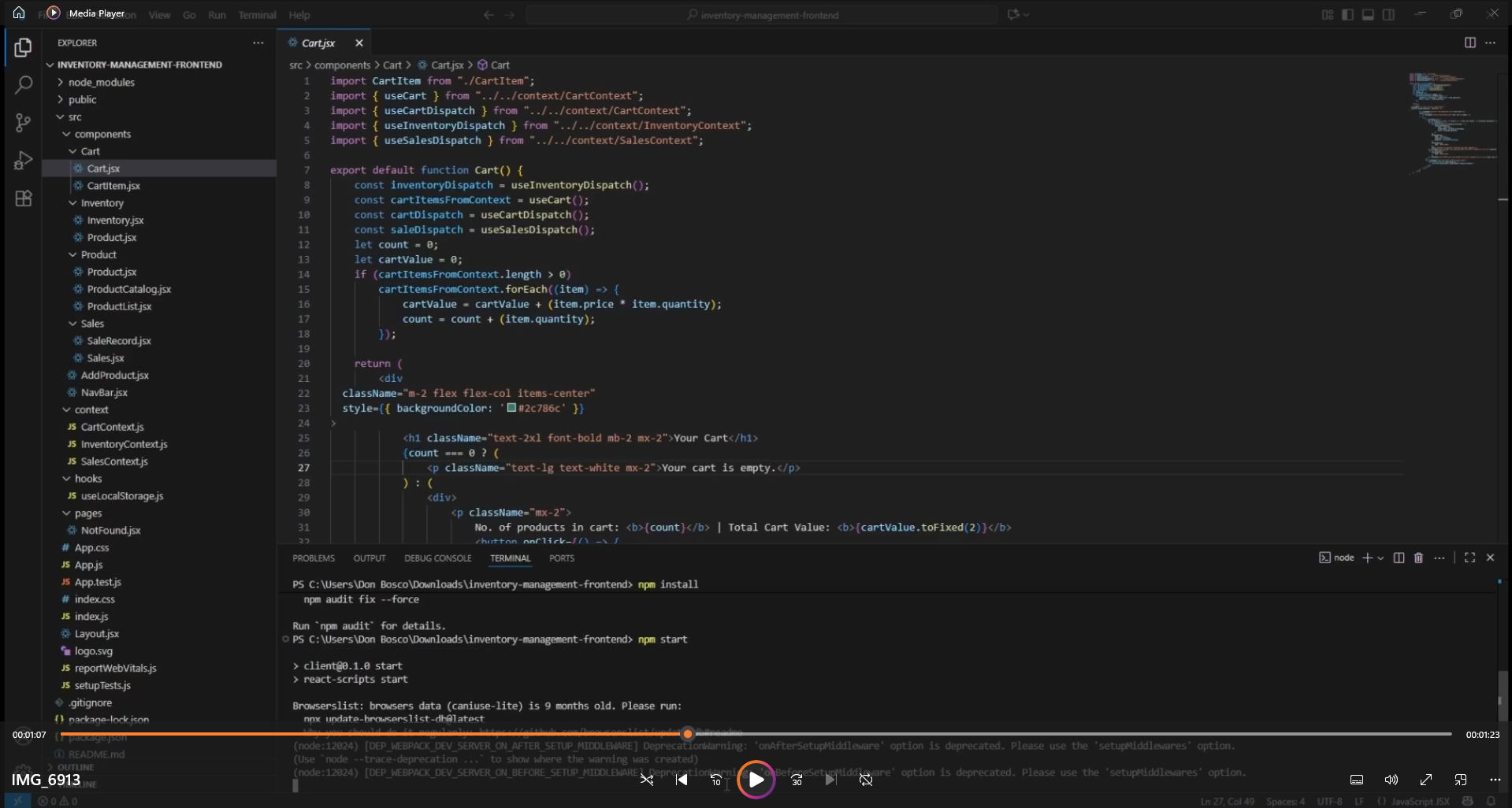
**11. Testing**

* **Testing Strategy**:
  + No test suites implemented yet.
  + Recommended: Jest + React Testing Library
* **Code Coverage**:
  + Not tracked currently.

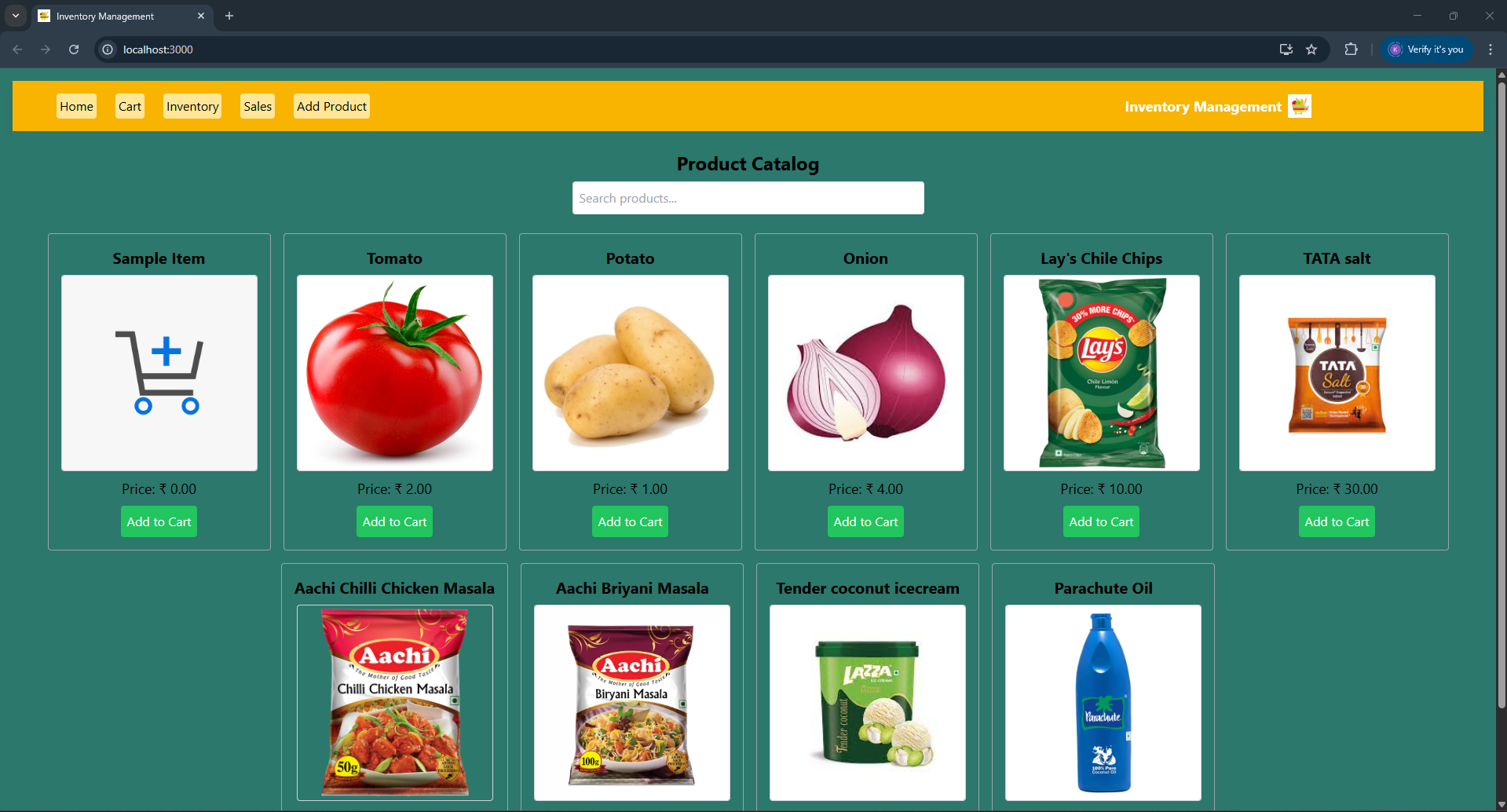
**12. Screenshots or Demo**

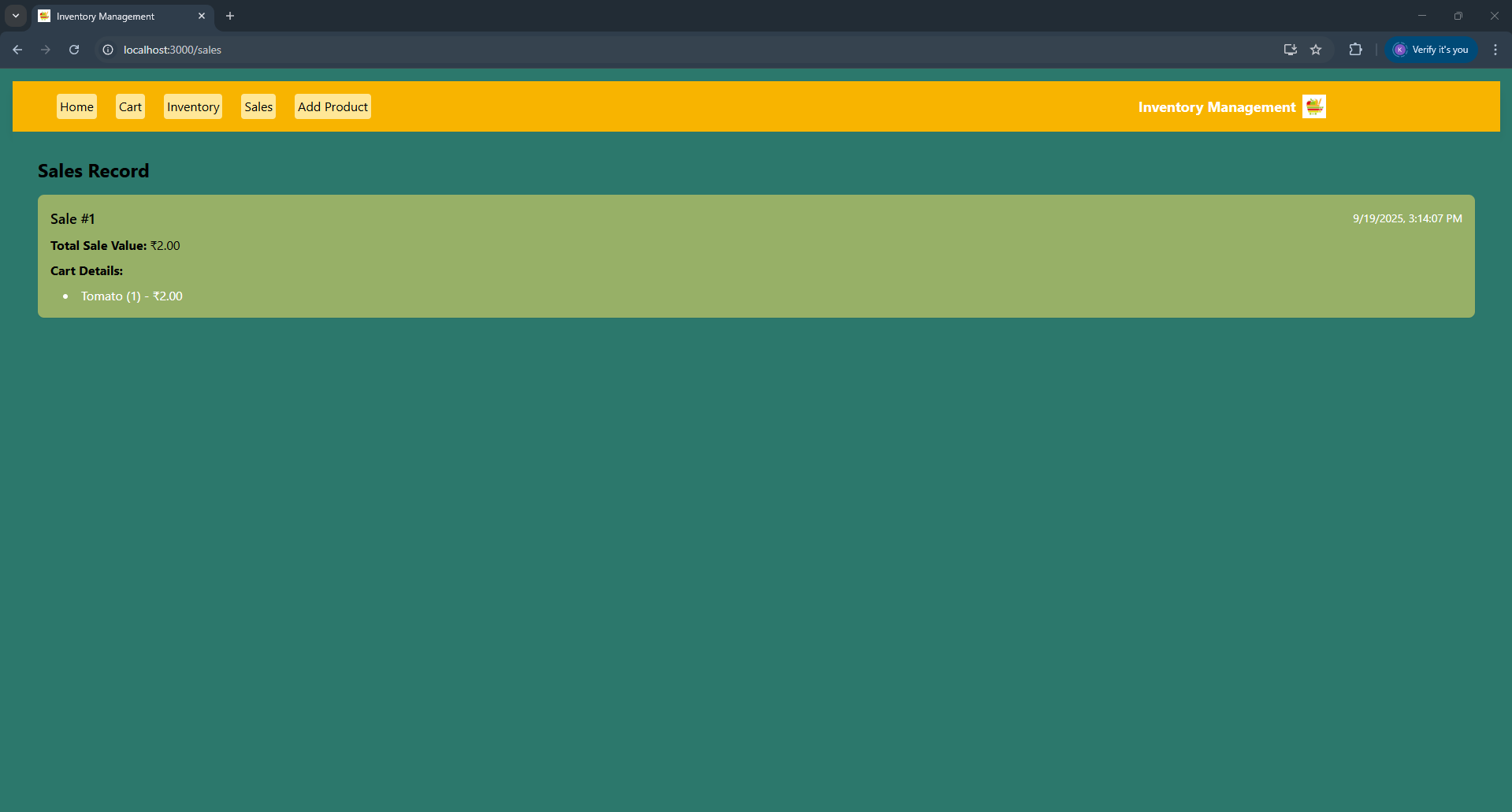
**Demo video**

****

****

**Website**





**13. Known Issues**

* Missing backend/API integration
* Form validation not implemented
* No global state (Context/Redux)
* No persistent storage (mock data only)

**14. Future Enhancements**

* Add login/authentication flow
* Connect to backend API for dynamic data
* Implement global state using Redux or Context API
* Add form validation using Formik or Yup
* Add search, filter, and sort functionality
* Deploy to Netlify or Vercel with CI/CD