

Project Documentation

Project Title: keep track of inventory.

1. Introduction

Project Title: keep track of inventory

Team ID : NM2025TMID42269

Team Leader : VARALAKSHMI .D
(Varalakshmiddevandiran5@gmail.com)

Teammember:SUDHA.K
(sudhakuppulingam24042007@gmail.com)

Team member : TAMILARASI. K
(Tamiltamilarasik.19.09@gmail.com)

Team member : SUMITHRA P
(Sumiparthi0002007@gmail.com)

2. Project Overview Purpose:

Sri muruga Shopping Complex is a simple e-commerce style web application that allows users to browse products, add items to the cart, check inventory, track sales, and add new products.

Features:

- Product Catalog with details (price, stock, tags, comments).
- Add to Cart functionality.
- Inventory management with stock tracking.
- Add new product form (Name, Image, Price, Stock, Tag, Comment).
- Sales page (future tracking system).
- User-friendly interface with responsive design.

3. Architecture

Frontend:

- HTML, CSS, JavaScript..

Responsive design with CSS flexbox and media queries..

Backend:

- Not implemented in this version (can be extended with Node.js & Express).

Database:

- Currently uses in-memory JavaScript array (products & cart).
- Can be extended with MongoDB for persistent storage.

4. Setup Instructions:

Prerequisites

- Web Browser (Chrome/Firefox/Edge).
- Code Editor (VS Code recommended).

Installation Steps::

1. Clone or download the project files.
2. Open the index.html file in a browser.
3. No server setup required (static HTML/JS app).

5. Folder Structure:

Sri muruga Shopping-Complex/

```
| — index.html
| — style.css
| — script.js
| — assets/
```

#

Main HTML file

Embedded/External CSS (optional separation)

#

#

JavaScript logic (optional separation)

Images & icons (optional)

6. Running the Application:

- Open index.html in a browser.

Navigate using the menu bar:

- Home: Browse products.
- Cart: View added products.
- Inventory: Check stock levels.
- Sales: Placeholder page for sales tracking.
- Add Product: Add new products dynamically.

7. API Documentation (Future Scope):

(Currently handled by JavaScript in-memory objects; can be extended to backend APIs)

Products API:

- GET /api/products → Fetch all products.
- POST /api/products → Add a new product.

Cart API:

- POST /api/cart/add → Add product to cart.
- GET /api/cart → Get cart items.

Sales API:

GET /api/sales → Fetch sales records.

8. Authentication (Future Scope):

Implement user login system with JWT authentication (optional).

9. User Interface:

- Home Page: Product catalog.
- Cart Page: Items added by user.
- Inventory Page: Stock & details.
- Sales Page: Placeholder for future enhancements.
- Add Product Page: Form to add products dynamically.

10. Testing

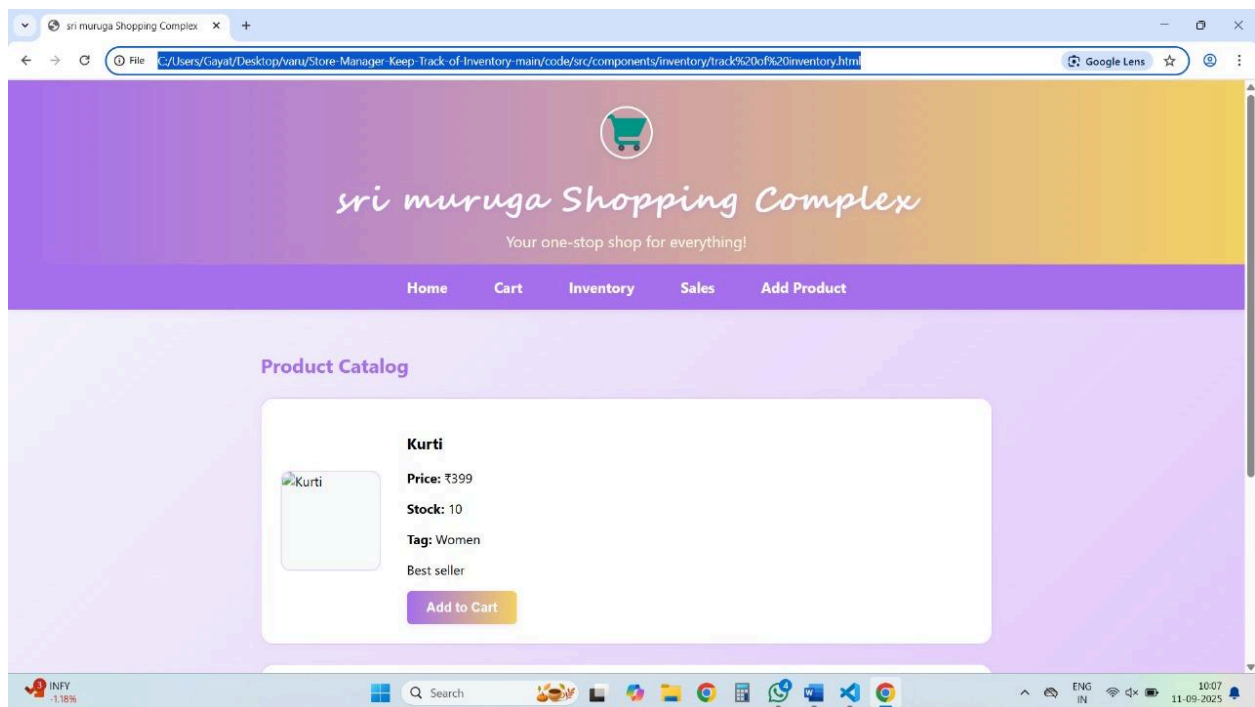
Manual Testing:

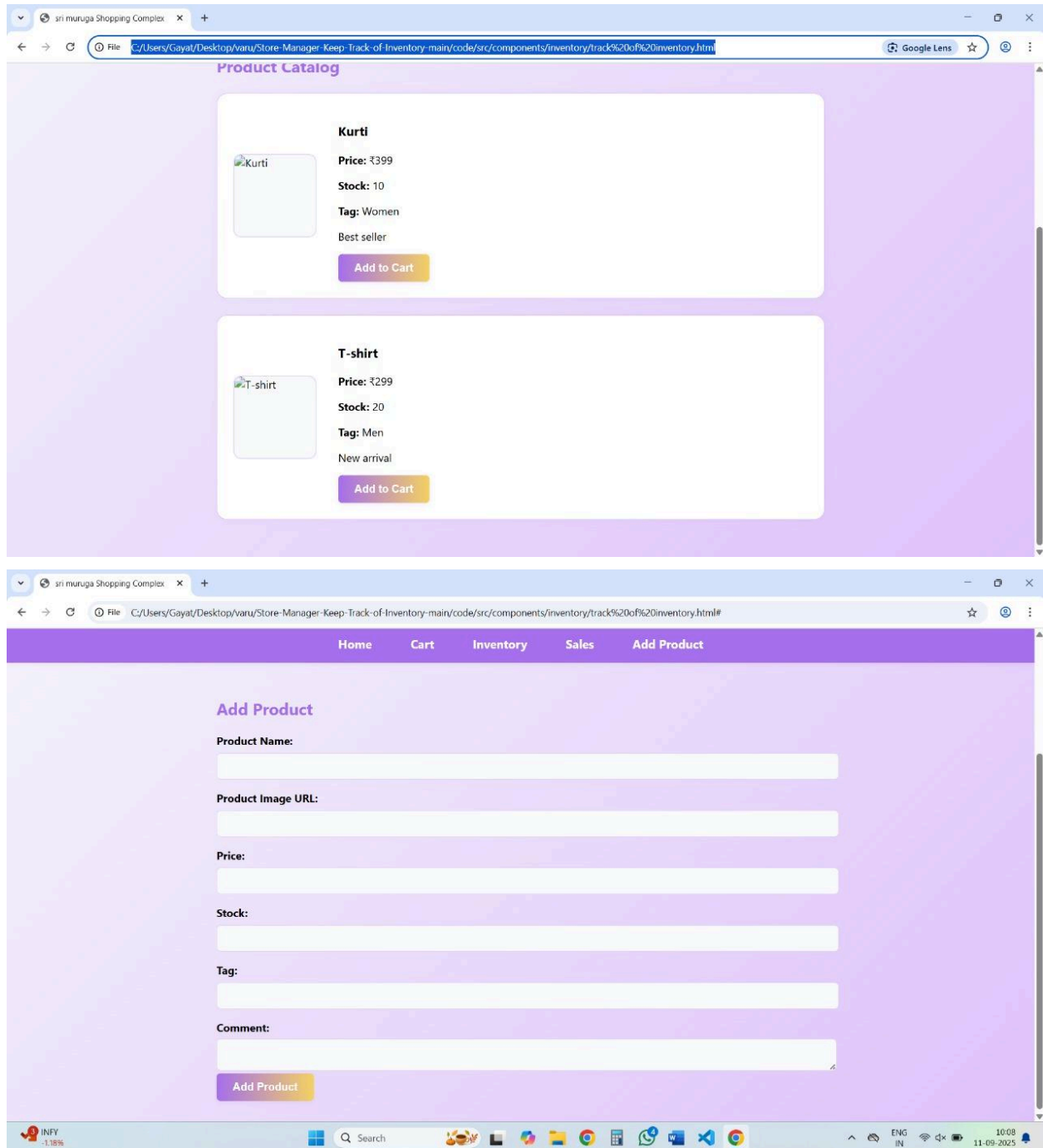
Add product, add to cart, stock decrement, inventory updates.

Tools: Browser console, Chrome DevTools.

11. Screenshots or Demo:

SCREENSHORT:





12. Known Issues:

- Data resets on page refresh (no persistent storage)..
- No authentication system implemented..
- Sales tracking not functional yet.

13. Future Enhancements::

Connect backend with Node.js + Express + MongoDB.
Implement user login & authentication..
Add checkout and payment system.
Create sales tracking & reports.
Deploy on Netlify/Heroku.