

# PROJECT DOCUMENTATION

## ZING-INVENTORY-MANAGEMENT

### 1. INTRODUCTION :

**Project Title:** ZING-INVENTORY-MANAGEMENT

**Team ID:** NM2025TMID38525

**Team Leader:** SRI VISHNUPRIYA A & [srivishnupriya48@gmail.com](mailto:srivishnupriya48@gmail.com) [Coder]

**Team Members & [Roles]:**

1. SHREEYA M & [shrezz.2430@gmail.com](mailto:shrezz.2430@gmail.com) [Coder]
2. SARMITHA A & [sarmitha516@gamil.com](mailto:sarmitha516@gamil.com) [Documenter]
3. SANDHIYA E & [tamilsan2212@gmail.com](mailto:tamilsan2212@gmail.com) [Video Recorder]

### 2. PROJECT OVERVIEW :

- **Purpose:** To provide a simple and efficient platform for managing products, sales, and stock. It is designed to help store managers and businesses maintain accurate records of their inventory, streamline sales tracking, and ensure proper stock monitoring.
- **Features:**
  1. **Product Management**-Allows adding, updating, and deleting product information like name, price, and category.
  2. **Inventory Tracking**-Monitors real-time stock levels and alerts for lower or out-of-stock items.
  3. **Sales and Billing**-Records sales transactions and generates invoices, with automatic stock deduction.
  4. **Supplier Management**-Stores supplier details and tracks purchase orders for stock replenishment.
  5. **Mobile Accessibility**- Mobile app support for stock checking and updates.

### **3. ARCHITECTURE :**

- **Frontend:** the frontend is the part of a website or application that the user directly sees and interacts with, including the visual elements, layout, and interactive features.
- HTML, CSS, Java script, React.js with Bootstrap & Material UI.
  - **Html-** it uses tags to describe the structure and elements of a webpage (like headings, paragraphs, images, and links).
  - **CSS-** it is the language used to style and design the appearance of web pages created with HTML.
- **Backend:** The backend refers to the part of a software application or system that operates behind the scenes and is not directly visible to users. It handles all the core logic, data processing, and database interactions required to make the application work.
  - **Node.js (with Express.js)** — Lightweight, fast, and great for JavaScript-based stacks.
  - **React.js—React.js** (commonly referred to as React) is an open-source JavaScript library used for building user interfaces (UIs), especially for single-page applications (SPAs).
- **Data base:** A database is an organized collection of data that is stored and managed so it can be easily accessed, updated, and retrieved by computer systems.
  - **Mongo DB**—Mongo DB Is A No SQL Database That Stores Data In A Flexible, Document-Oriented Format, Using JSON-Like Documents (Called BSON — Binary JSON)
  - Mongo DB Store users data, application & chat messages

## **4. SETUP INSTRUCTIONS :**

### **• Prerequisites:**

- Node.js
- Mongo DB
- Git
- React.js
- Express.js
- Mongoose
- Visual Studio Code
- Git hub

### **• Installation Steps:**

- Navigate into the cloned repository directory and install libraries:
  - cd zing-inventory-management
  - npm install
- Start the development server, execute the command:
  - npm start
- Access the web browser & Navigate:
  - <https://localhost:3000>
- Integrated Development Environment:
  - Visual Studio Code

## **5. FOLDER STRUCTURE :**

Zing-Inventory-Management/

```
|— public/
|— src/
|   |— components/
|   |— Cart
|   |— Inventory
|   |— Product
|   |— Sales
|— Page/
|   |— .gitignore
|   |— {} package-lock.json
|   |— {} package.json
|   |— README.md
```

## **6. RUNNING THE APPLICATION :**

- **Frontend:**
  - Install dependencies  
-npm install
  - Run the frontend  
-npm start
- **Backend:**
  - Run the server  
-npm start
- **Access:**
  - Visit <http://localhost:3000>

## 7. API DOCUMENTATION :

### ❖ Login :

```
<div>
```

```
    <h1 className="text-3xl lg:text-4xl font-extrabold tracking-tight bg-gradient-to-r from-slate-900 to-purple-800 bg-clip-text text-transparent mb-8 text-center">Welcome Back</h1>
```

```
<form onSubmit={onSubmit} className="space-y-5">
```

```
    <div>
```

```
        <label htmlFor="username" className="block text-sm font-semibold text-slate-700 mb-1.5">Username</label>
```

```
        <input id="username" type="text" required className="w-full px-4 py-3 rounded-xl border border-slate-200 bg-white text-slate-800 placeholder-slate-400 shadow-sm focus:outline-none focus:ring-2 focus:ring-purple-300 focus:border-purple-400 transition" placeholder="Enter your username" />
```

```
    </div>
```

```
    <div>
```

```
        <label htmlFor="password" className="block text-sm font-semibold text-slate-700 mb-1.5">Password</label>
```

```
        <input id="password" type="password" required className="w-full px-4 py-3 rounded-xl border border-slate-200 bg-white text-slate-800 placeholder-slate-400 shadow-sm focus:outline-none focus:ring-2 focus:ring-purple-300 focus:border-purple-400 transition" placeholder="Enter your password" />
```

```
    </div>
```

```
    <button type="submit" className="w-full py-3 rounded-xl font-semibold text-white shadow-md bg-gradient-to-r from-purple-600 to-blue-600 hover:from-purple-500 hover:to-blue-500 transition">Log In</button>
```

```
</form>
```

```
    <p className="mt-5 text-center text-sm text-slate-600">New user? <a href="/signup" className="text-purple-700 hover:underline">Create an account</a></p>
```

## ❖ Signup :

</div>

<h1 className="text-3xl lg:text-4xl font-extrabold tracking-tight bg-gradient-to-r from-slate-900 to-purple-800 bg-clip-text text-transparent mb-8 text-center">Create Account</h1>

<form onSubmit={onSubmit} className="space-y-5">

<div>

<label htmlFor="username" className="block text-sm font-semibold text-slate-700 mb-1.5">Username</label>

<input id="username" type="text" required className="w-full px-4 py-3 rounded-xl border border-slate-200 bg-white text-slate-800 placeholder-slate-400 shadow-sm focus:outline-none focus:ring-2 focus:ring-purple-300 focus:border-purple-400 transition" placeholder="Choose a username" />

</div>

<div>

<label htmlFor="password" className="block text-sm font-semibold text-slate-700 mb-1.5">Password</label>

<input id="password" type="password" required className="w-full px-4 py-3 rounded-xl border border-slate-200 bg-white text-slate-800 placeholder-slate-400 shadow-sm focus:outline-none focus:ring-2 focus:ring-purple-300 focus:border-purple-400 transition" placeholder="Create a password" />

</div>

<div>

<label htmlFor="confirm" className="block text-sm font-semibold text-slate-700 mb-1.5">Confirm Password</label>

<input id="confirm" type="password" required className="w-full px-4 py-3 rounded-xl border border-slate-200 bg-white text-slate-800 placeholder-slate-400 shadow-sm focus:outline-none focus:ring-2 focus:ring-purple-300 focus:border-purple-400 transition" placeholder="Confirm your password" />

</div>

<button type="submit" className="w-full py-3 rounded-xl font-semibold text-white shadow-md bg-gradient-to-r from-purple-600 to-blue-600 hover:from-purple-500 hover:to-blue-500 transition">Sign Up</button>

</form>

<p className="mt-5 text-center text-sm text-slate-600">Already have an account? <a href="/login" className="text-purple-700 hover:underline">Log in</a></p>

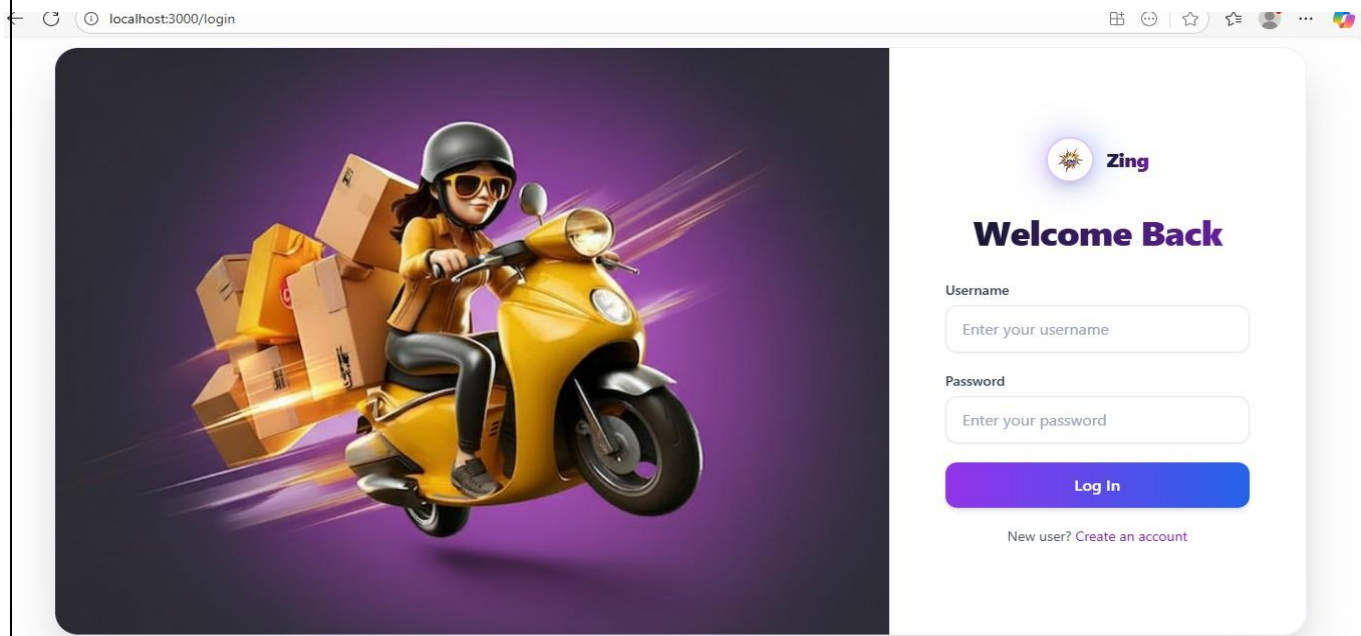
</div>

## **8. AUTHENTICATION :**


- Check minimum length (e.g., at least 8 characters).
- Require combination of:
  - Uppercase letters (A-Z)
  - Lowercase letters (a-z)
  - Numbers (0-9)
  - Special characters (e.g., !@#\$%^&\*)
- No common or easily guessable passwords (e.g., "password123").
- Optional: Check against breached password databases (e.g., using Have I Been Pwned API).


## **9. USER INTERFACE :**

### ***LOG IN***



## SIGN UP



**ZING**

### Create Account

Username


Password

Confirm Password

**Sign Up**

Already have an account? [Log in](#)


## PRODUCT CATLOG

**ZING**

Home Cart Inventory Sales **Add Product** Logout

### Product Catalog

**Cheese**




**Pure Milk** 100

Price: ₹ 50.00

**Add to Cart**


**Panner**



Price: ₹ 75.00

**Add to Cart**


**Butter**




Price: ₹ 209.98

**Add to Cart**


**Milk**



**aashirvaad atta**

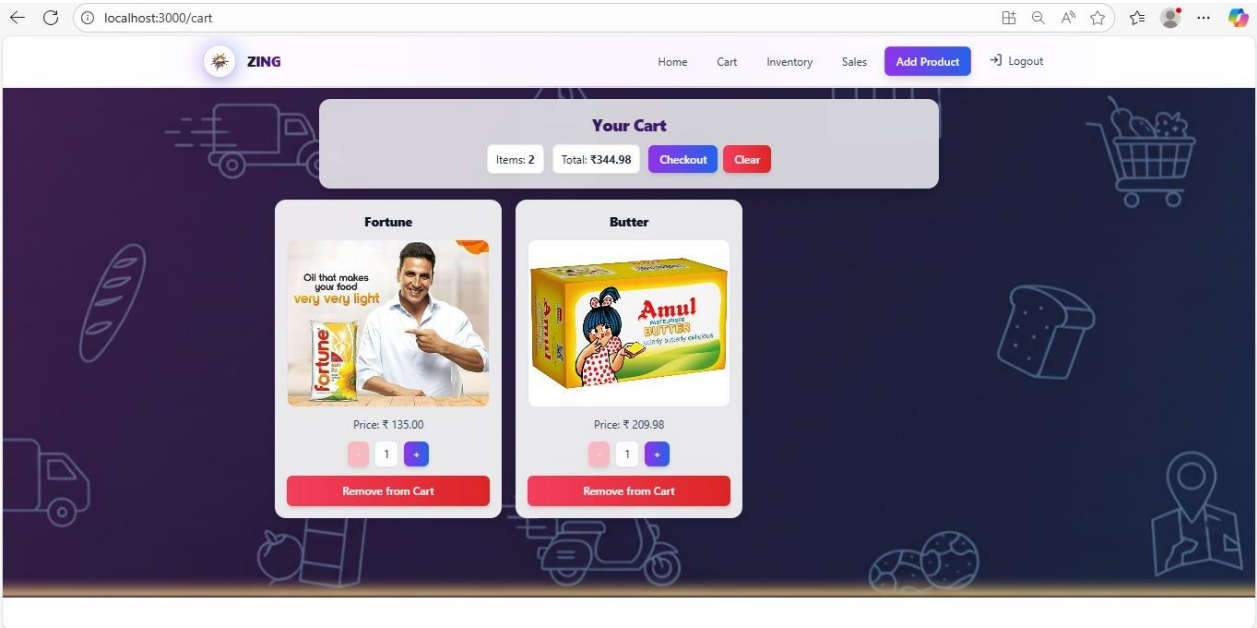


**Groundnut**

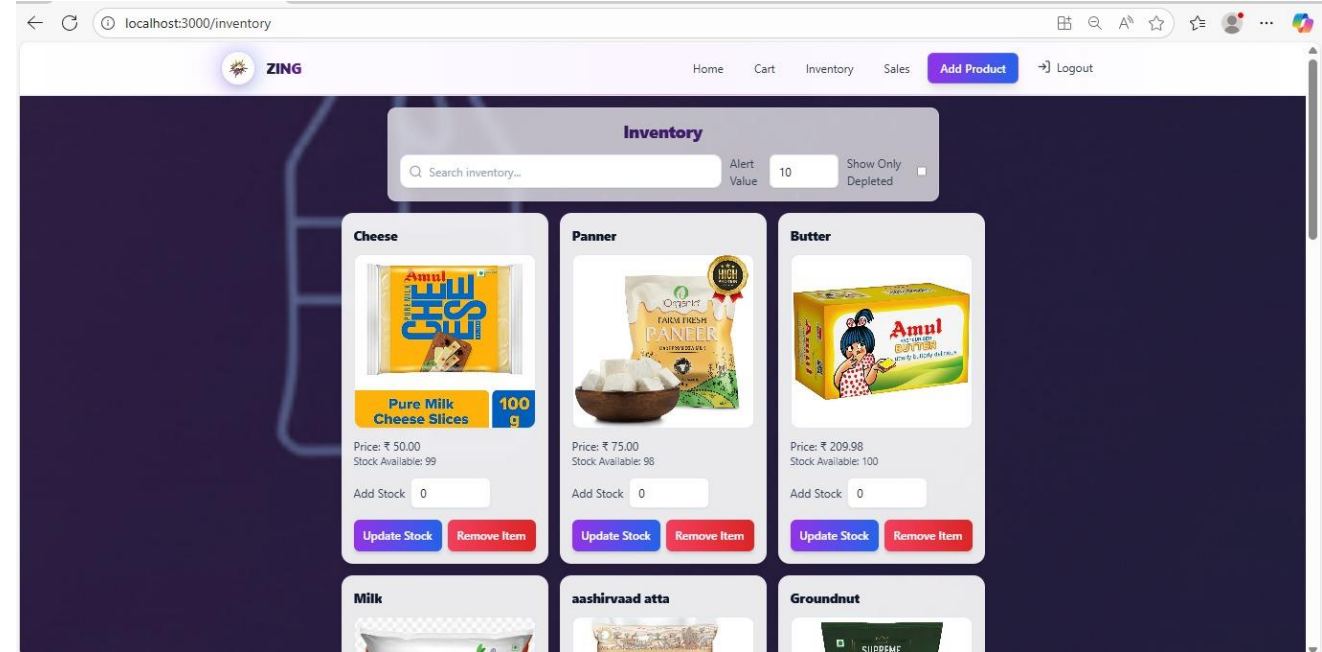




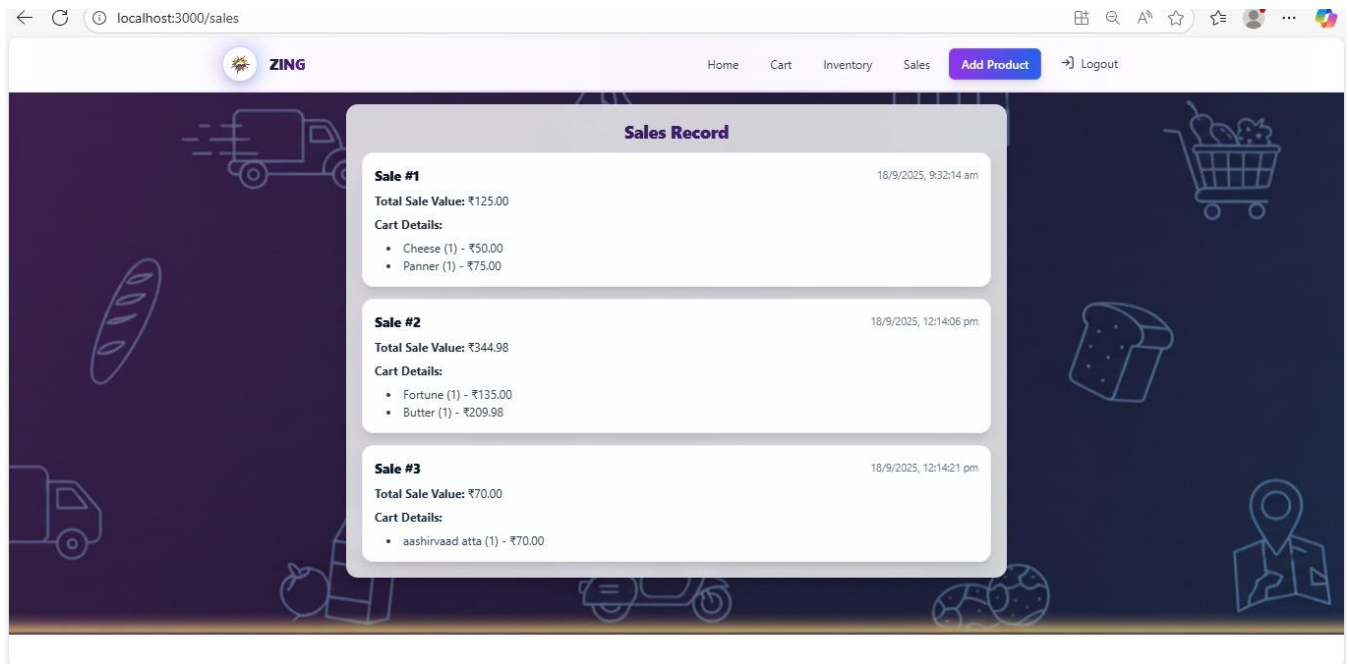
CART



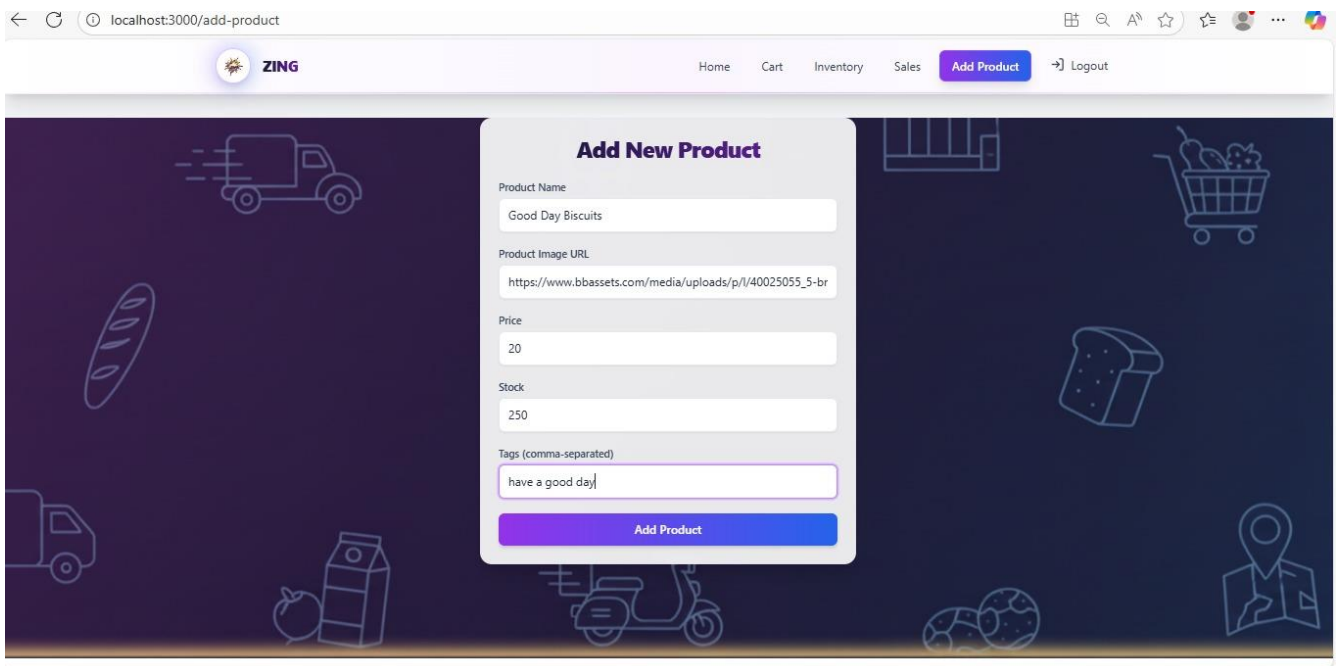
INVENTORY



## SALES RECORD



## ADD NEW PRODUCT



## 10. VIDEO :

[https://drive.google.com/file/d/1XtgfXD-mhH9BLVDugloQfXnx-e113VDV/view?usp=drive\\_link](https://drive.google.com/file/d/1XtgfXD-mhH9BLVDugloQfXnx-e113VDV/view?usp=drive_link)

## **11. KNOWN ISSUES :**

### **1. Lack of Real-Time Updates**

- **Cause:** Delays between POS transactions and inventory updates
- **Impact:** Users see outdated stock information.
- **Mitigation:** Sync systems in real-time or reduce sync intervals.

### **2. Missing or Incomplete Product Data**

- **Cause:** New items added without full details (e.g., SKU, category, cost).
- **Impact:** Affects reporting accuracy and search/filter functionality.
- **Mitigation:** Enforce mandatory fields when adding new items.

### **3. Duplicate Product Entries**

- **Cause:** Multiple entries for the same product under different names/SKUs.
- **Impact:** Inflates stock values, confuses staff.
- **Mitigation:** Implement SKU validation and product lookup on entry.

## **12. FUTURE ENHANCEMENT :**

- Real-Time Inventory Tracking
- Low Stock & Overstock Alerts
- Automated Reordering System
- Dashboard & Reporting Enhancements
- Mobile App Access