Grocery Delivery System

# Purpose

To develop an online **grocery delivery platform** where buyers can select products and post purchase quotations sent to nearby shop owners. Shop owners can respond with modified quotations (counter-offers), and buyers confirm one to proceed with the order. This buyer-driven model encourages flexible pricing, enhances local engagement, and supports efficient deliveries.

# Objective

* Enable buyers to browse product categories and submit customized quotations.
* Allow shop owners to respond with updated quotations (negotiation loop)
* Let buyers confirm one quotation response to lock the order and notify the chosen shop owner.
* Facilitate seamless order processing, order tracking, and communication between buyer and accepted shop owner.
* Provide dashboards for buyers and owners with features like profile management, order history, analytics, and product listings.
* Ensure role-based login for buyers and shop owners.

# Demand / Opportunity

This is a **demand-based** project driven by the need for a flexible, local grocery ordering platform where buyers can submit custom quotations. It addresses the growing demand for convenience and digital access to nearby shop owners.

# Business Requirement:

## Two user roles: Grocery Buyer and Shop Owner.

* + - **Buyer:** Browse products, submit quotations, receive counter-offers, confirm selected quotations, track orders, manage profile, set delivery location (manual or GPS).
    - **Owner:** Manage product listings, receive buyer quotations, respond with adjusted offers (price/quantity), fulfill confirmed orders, view analytics, manage profile.

## Quotation submission and acceptance system.

* + - Buyers submit quotations. Nearby owners can respond with revised offers. Buyers review responses and confirm one. Once confirmed, the order is locked to that owner.

## Cart management and address detection for delivery.

* + - Buyers can build a cart of multiple products, enter delivery addresses or use device based location detection.

## Order history, tracking, and real-time status updates.

* + - Both parties can access order status (e.g., Pending, Confirmed, Preparing, Delivered, Cancelled) and track it in real-time.

## Admin product listing and user management.

* + - Owners manage their inventory and track business performance through built-in

analytics tools.

# Technical Requirement:

### User Authentication

* + - UI and API routes for buyer and owner registration, login, forgot password, and reset password
    - Role-based access (buyer or shop owner)
    - Secure session handling via middleware

### Quotation Workflow & Negotiation

* + - Buyer-side UI for product selection and quotation submission
    - Owner-side UI/API to view and respond with adjusted offers
    - Quotation response aggregation on buyer side for review
    - Confirmation API to lock selected quotation and notify owner
    - Locking logic to restrict duplicate owner acceptances

### Cart and Order Placement

* + - Cart system for buyers to add/edit/remove multiple products
    - Address input via form or GPS-based location detection (Google Maps API or similar)
    - Order creation and linking accepted quotation with buyer and owner

### Real-Time Order Tracking

* + - API and UI for order status (Pending, Confirmed, Preparing, Delivered, Cancelled)
    - Notification system for buyers and owners (status updates)
    - Order tracking component on buyer dashboard

### Product & Category Management (Owner Side)

* + - UI and API to add/edit/delete products with name, description, price, and category
    - Category dropdowns on both buyer and owner interfaces

### Real-Time Order Tracking

* + - Buyers view previous and current orders with status
    - Owners view fulfilled orders, revenue, top-selling items

# Technological Requirement:

## Frontend

* + - React Js
    - Tailwind Css

## Backend

* + - Express Js
    - Node Js

## Database

* + - SQL → MySQL

## Tools

* + - Visual Studio Code [ Code IDE ]
    - GIT & GitHub [ version control ]
    - Postman [ api testing ]

# Stakeholder:

|  |  |  |
| --- | --- | --- |
| **Stakeholder** | **Name** | **Count** |
| Developers | Avishkar Pawar Arya Sadalage Shruti Deokar Akshit Dhake | 4 |
| End Users | Grocery Buyers Shop Owners | 2 |
| Testers | Srivaths Iyer  Prathamesh Shelke | 2 |
| Hosting Service Provider | Cpanel | 1 |
| Investor | TBD | 0 |

**Resources Needed:**

## Documentation

* + - React Js
    - Tailwind Css
    - Node Js
    - Express Js

## Hosting

* + - Cpanel
  + **Human Resource**

|  |  |  |
| --- | --- | --- |
| **Role** | **Count** | **Name** |
| Frontend Developer | 4 | UI Development - Avishkar Pawar  Akshit Dhake API Integration - Arya Sadalage  Shruti Deokar |
| Backend Developer | 2 | API Development & API Testing  - Arya Sadalage Shruti Deokar |
| DB Designer | 2 | Avishkar Pawar Shruti Deokar |
| Project Management | 4 | Avishkar Pawar Akshit Dhake Arya Sadalage Shruti Deokar |
| Documentation | 4 | Creator - Shruti Deokar  Akshit Dhake Reviewer - Avishkar Pawar  Arya Sadalage |
| Tester | 2 | Srivaths Iyer  Prathamesh Shelke |

# PESTEL Analysis:

* + **Political:** Local trade regulations and delivery norms.

### Economic:

* + - Encourages local economy and vendor growth.

### Social:

* + Addresses convenience needs and contactless delivery.

### Technological:

* + Leverages modern tech stack and APIs

### Legal:

* + Must comply with digital transaction and data privacy laws.
* **Environmental**: Reduces unnecessary trips; promotes local sourcing.

**Risk Analysis:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Impact** | **Description** | **Mitigation** |
| Delay in order acceptance | Medium | Orders may not be accepted promptly, causing buyer frustration and potential order cancellations. | Notify buyers when acceptance time exceeds a set limit to keep them informed. |
| Miscommunication on order | High | Incorrect or unclear order details can lead to wrong deliveries and disputes. | Implement real-time status updates and notifications to keep both parties aligned. |
| Server downtime | High | System unavailability can disrupt order processing and user experience. | Choose reliable hosting providers with high uptime SLAs to minimize downtime. |
| Location inaccuracy | Medium | Incorrect location data may cause delivery delays or failures. | Use GPS for location detection with manual address fallback options. |
| Budget overrun | Low | Project costs may exceed the planned budget, affecting feasibility. | Use affordable hosting services and open-source technologies to control expenses. |

**Timeline / Milestone:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Milestone** | **Tasks** | **Timeline** |
| 1 | Requirement Analysis & Planning | * Gather and Document project requirements * Define & construct functional flow of the application * Identify third-party services and dependencies * Design UI mockups and wireframes | Week 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | Database Design and API Development | * Design database schema * Set up database configurations * Set up middleware and API structure * Develop API endpoints * Implement user authentication setup * Perform API testing using postman | Week 2 |
| 2 | UI Development and API Integration | * Develop UI components and pages * Implement state management * Connect frontend with backend * Implement error handling & validation * Optimize API calls | Week 3 |
| 4 | Testing & Debugging | * Perform UI/UX testing * Perform API integration testing * Fix bugs and optimize performance | Week 4 |
| 5 | Deployment & Final Review | * Set up cPanel hosting environment * Configure FTP and database connections * Deploy application using cPanel file manager or FTP * Monitor website performance and resolve hosting issues * Conduct final review and complete documentation | Week 4 |

**Budget:**

|  |  |
| --- | --- |
| **Item** | **Estimated Cost (INR)** |
| cPanel Hosting | ₹10,000–₹15,000/year |
| Domain | ₹500/year |
| **Total** | **₹15,000–₹20,000/year** |