# Online Grocery Order System

## High Level Design (HLD)

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to provide a high-level overview of the Online Grocery Order System, including its functionality, architecture, and design. This document serves as a reference for project stakeholders, developers, and testers.

## 1.2 Scope

The Online Grocery Order System is an e-commerce platform that allows customers to browse, select, and purchase groceries online. It includes features such as user registration, product browsing, cart management, order placement, and payment processing.

## 1.3 Document Conventions

- **User:** Refers to the individuals who use the online grocery ordering system to purchase groceries.
- **Admin:** Refers to the system administrators responsible for managing the product catalog, user accounts, and orders.

## 1.4 Intended Audience

This document is intended for:

- Project Managers

- Developers
- Testers
- Designers
- Stakeholders

## 2. System Overview

### 2.1 System Architecture

The Online Grocery Order System follows a client-server architecture. It consists of a web-based user interface for customers and an admin dashboard for administrators. The system is built using a combination of front-end and back-end technologies.

### 2.2 Key Features

- User Registration and Authentication
- Product Browsing and Search
- Cart Management
- Order Placement
- Payment Processing
- Order History
- Notification System
- Admin Dashboard for Product Management

### 2.3 User Roles

1. **Customer**: Can browse products, add items to the cart, place orders, and manage their account.

2. **Admin**: Manages the product catalog, user accounts, and orders. Accesses the admin dashboard.

## 3. Functional Requirements

### 3.1 User Registration and Authentication

- Users can create accounts.
- Users can log in and log out.
- Password reset and account recovery options are available.

### 3.2 Browsing Products

- Users can browse products by category.
- Product details include name, description, price, and availability.
- Users can search for products.

### 3.3 Adding Products to Cart

- Users can add items to their shopping cart.
- Cart displays product details and quantities.

### 3.4 Managing Cart

- Users can view and edit the contents of their cart.
- Option to remove items from the cart.

### 3.5 Placing Orders

- Users can place orders from their cart.

- Specify delivery address and delivery time.
- Confirmation email sent to users.

### 3.6 Payment Processing

- Secure payment gateway integration.
- Support for multiple payment methods (credit card, online wallets, etc.).

### 3.7 Order History

- Users can view their order history.
- Details of past orders including items, dates, and statuses.

### 3.8 Notifications

- Email and in-app notifications for order updates.
- Admin receives notifications for new orders and product management.

## 4. Non-functional Requirements

### 4.1 Performance

- System should handle concurrent user requests efficiently.
- Fast loading times for web pages.
- Scalable architecture to handle increased load during peak times.

### 4.2 Security

- Secure user authentication and data encryption.
- Protection against SQL injection and other common vulnerabilities.
- Regular security audits and updates.

### 4.3 Usability

- Intuitive and user-friendly interface.
- Responsive design for mobile and desktop.
- Accessibility features for users with disabilities.

### 4.4 Availability

- High system availability (99.9% uptime).
- Regular maintenance and backups.

### 4.5 Scalability

- The system should be able to scale horizontally to accommodate increased traffic.

## 5. System Design

### 5.1 Database Schema

- Design the database schema to store user data, product information, orders, and cart details.

### 5.2 User Interface Design

- Create wireframes and mockups for the user interface.
- Ensure a responsive and user-friendly design.

### 5.3 Payment Integration

- Integrate a secure payment gateway for processing transactions.

### 5.4 Notification System

- Design a notification system for sending email and in-app notifications.

## 6. Testing

### 6.1 Test Scenarios

- Define test cases for each feature and functionality.
- Include positive and negative test scenarios.

### 6.2 User Acceptance Testing

- Conduct user acceptance testing with real users.
- Ensure the system meets user expectations.

### 6.3 Performance Testing

- Test the system's performance under load.
- Identify and address bottlenecks.

### 6.4 Security Testing

- Perform security testing to identify and fix vulnerabilities.
- Penetration testing and code reviews.

## 7. Deployment

### 7.1 Hardware Requirements

- Specify the hardware requirements for the server infrastructure.

### 7.2 Software Requirements

- List the software components and technologies used in the system.

### 7.3 Deployment Steps

- Provide detailed steps for deploying the system to production.

## 8. Maintenance and Support

### 8.1 Bug Reporting

- Set up a process for users to report bugs and issues.

### 8.2 Feature Requests

- Collect and prioritize feature requests from users.

### 8.3 Software Updates

- Plan for regular software updates and maintenance.

## 9. Conclusion

This high-level document provides an overview of the Online Grocery Order System, including its scope, functionality, and design. It serves as a reference for all stakeholders involved in the development, testing, deployment, and maintenance of the system.