Midterm Project: Online Grocery Delivery System

By: Yamil Diaz

Table of Contents

Online Grocery Delivery System

Yamil A. Diaz Rosa

Problem Statement

In many communities, grocery shopping takes a lot of time and can be stressful. Some people don’t have transportation or have mobility issues, and this makes shopping even harder. At the same time, grocery stores can have trouble keeping track of online orders and inventory. Sometimes customers only find out that a product is out of stock after placing the order, which is frustrating. This project will focus on building an online grocery delivery system where customers can order products from home and get them delivered. The goal is to save time, reduce stress, and make it easier for both customers and grocery stores to handle orders.

Objectives of the System

The main objectives of this system are:

- Let customers order groceries online and get them delivered.
- Help grocery store managers keep their product lists and inventory updated.
- Show availability of items so customers don’t waste time.
- Provide delivery staff with a clear way to see their tasks.

System Requirements

The system will support three types of users: customers, store managers, and delivery staff.

- Customers: browse products, add to cart, place and pay for orders, track delivery.
- Store managers: add, edit, or remove products, manage incoming orders, update delivery slots.

- Delivery staff: see assigned deliveries, mark pickup and delivery status, update customers.

Typical Customers

The main users for this system will be:

1. Busy families and professionals who don’t have time to shop in-store.
2. Elderly or people with mobility problems who need delivery service.
3. Grocery managers who want easier order management.
4. Delivery staff who need a simple way to track tasks.

Project Planning

To build this system, I plan to use:

- Software: React.js for the frontend, Python with Django for the backend, and MySQL for the database.
- Hardware: regular PC for development and testing, and cloud server for deployment.
- Network: stable internet with HTTPS for secure communication.

Development Approach

I will use Python (Django) for the backend and JavaScript (React.js) for the frontend. The system will connect to a MySQL database. For extra features, I might use Stripe for payments and Google Maps API for delivery tracking. Success will be measured by testing if customers can place orders, managers can update stock, and delivery staff can update status without problems.

Development Plan

This is my plan for the semester:

- Weeks 1-2: Set up database and connect frontend with backend.
- Weeks 3-4: Create login and registration for customers, managers, and delivery staff.

- Weeks 5-7: Build product catalog, shopping cart, and checkout with payments.
- Week 8: Test the first features (mid-term demo).
- Weeks 9-11: Add delivery tracking and notifications.
- Weeks 12-14: Get feedback from users and fix problems.
- Week 15: Final testing and prepare the final demo

Customer Problem Statements & System

Requirements

Yamil Diaz

Customer Problem Statement:

Going to the grocery store can take a lot of time and be stressful. Some people do not have cars, or they have health and mobility issues that make it even harder. Stores also have problems keeping track of orders and items in stock. Sometimes customers only find out that an item is not available after they have already ordered it. This project is about making an online grocery delivery system where people can order from home and get the food delivered. This will help save time, reduce stress, and make things easier for both customers and stores.

Glossary of Terms

Customer: A person who orders groceries through the system.

Store Manager: A person who manages the products and orders in the system.

Delivery Staff: A worker who picks up and delivers the groceries.

Cart: A digital basket where products are placed before checkout.

Order Tracking: A way for customers to see the status of their order and delivery.

Functional Requirements

REQ-1 (High): Customers must be able to sign up, log in, and manage their account.

REQ-2 (High): Customers can look at products, add them to a cart, and place orders.

REQ-3 (High): Store managers must be able to add, edit, or remove products.

REQ-4 (High): The system must show if a product is available or out of stock.

REQ-5 (Medium): Customers should be able to pay using a credit or debit card.

REQ-6 (Medium): Delivery staff should be able to see assigned orders and update them.

REQ-7 (Medium): Customers must be able to check where their order is in the delivery process.

REQ-8 (Low): Customers should get messages when their order is confirmed, shipped, and delivered.

Nonfunctional Requirements

NFR-1 (High): Functionality - Must support customers, store managers, and delivery staff.

NFR-2 (High): Usability - The system should be simple and easy to use.

NFR-3 (Medium): Reliability - The system should work most of the time without crashing.

NFR-4 (Medium): Performance - The system should handle many users at the same time.

NFR-5 (Low): Supportability - The system should be easy to update and fix.

User Interface Requirements

UI-1 (High): Login and sign-up page for customers, managers, and delivery staff.

UI-2 (High): Product page with categories and a search box.

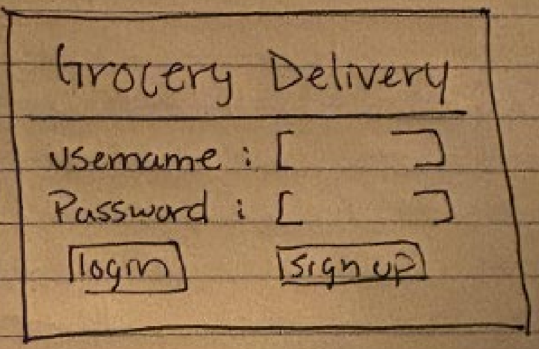
UI-3 (High): Shopping cart page to see selected items, total cost, and checkout.

UI-4 (Medium): Order tracking page with delivery progress.

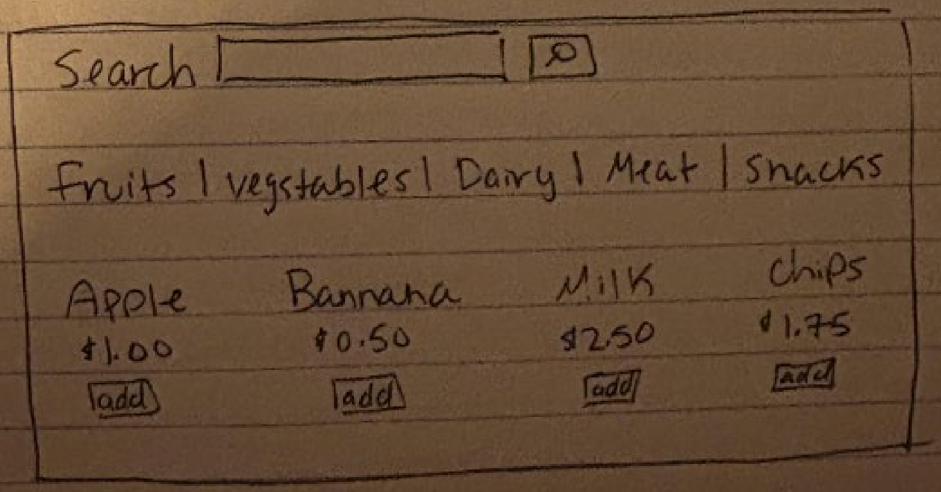
UI-5 (Medium): Manager page to update products and view orders.

UI-6 (Low): Delivery staff page to see orders and update status.

UI-1 (high) login and sign up page



UI-2 (high) Product Page



Functional Requirements Specification

Stakeholders

The people and groups who care about this project are:

- **Customers** – the people who will order the groceries online with the purpose of saving time and avoiding going to the physical store.
- **Store Managers** – the people who will handle the job of updating product lists, keep track of the stock, and check orders.
- **Delivery Staff** – the workers who are bringing the groceries to the customer's home.
- **System Administrators** – the people who will be maintaining the store's website and will also fix any issues.
- **Business Sponsors** – the people who own the grocery store as well as any investors who would want the system to improve their profits and increase customer satisfaction.

Actors and Goals

Primary actors

- **Customer:** This actor can sign up for services, log into their account, browse products, place orders, pay, and track deliveries. Their main goal is to get groceries delivered easily and without issues.
- **Store Manager:** This actor can add, remove, or edit products that are available, as well as view and manage the orders. Their main goal is to keep inventory up to date and make sure the orders are handled correctly.
- **Delivery Staff:** These actors can view assigned deliveries, update the order status, and send updates to customers. Their main goal is to complete deliveries efficiently.

Secondary actors

- **System:** This is responsible for handling stock checking, order confirmations, sending notifications, and processing the payments.
- **Payment Service:** This is for processing the payments.
- **Map/Tracking Service:** This helps show delivery tracking to the customer.

Use Cases

Below are the use cases based on the requirements. I used 2 points = 1 day of work for one developer.

Customer (Total: 16)

- **Sign up / Log in:** Create or access an account (2)
- **Browse products:** View categories and search (2)
- **Add to cart:** Select products for purchase (2)
- **Checkout & pay:** Confirm the order and pay (4)
- **Track order:** See order status and delivery progress (4)
- **Receive notifications:** Get updates for confirmed, shipped, delivered (2)

Store Manager (Total: 12)

- **Add product:** Add a new grocery item (2)
- **Edit/Remove product:** Update or delete products (2)
- **View orders:** See customer orders (2)
- **Update stock:** Mark items as available/out of stock (2)
- **Manage delivery slots:** Adjust delivery time availability (4)

Delivery Staff (Total: 8)

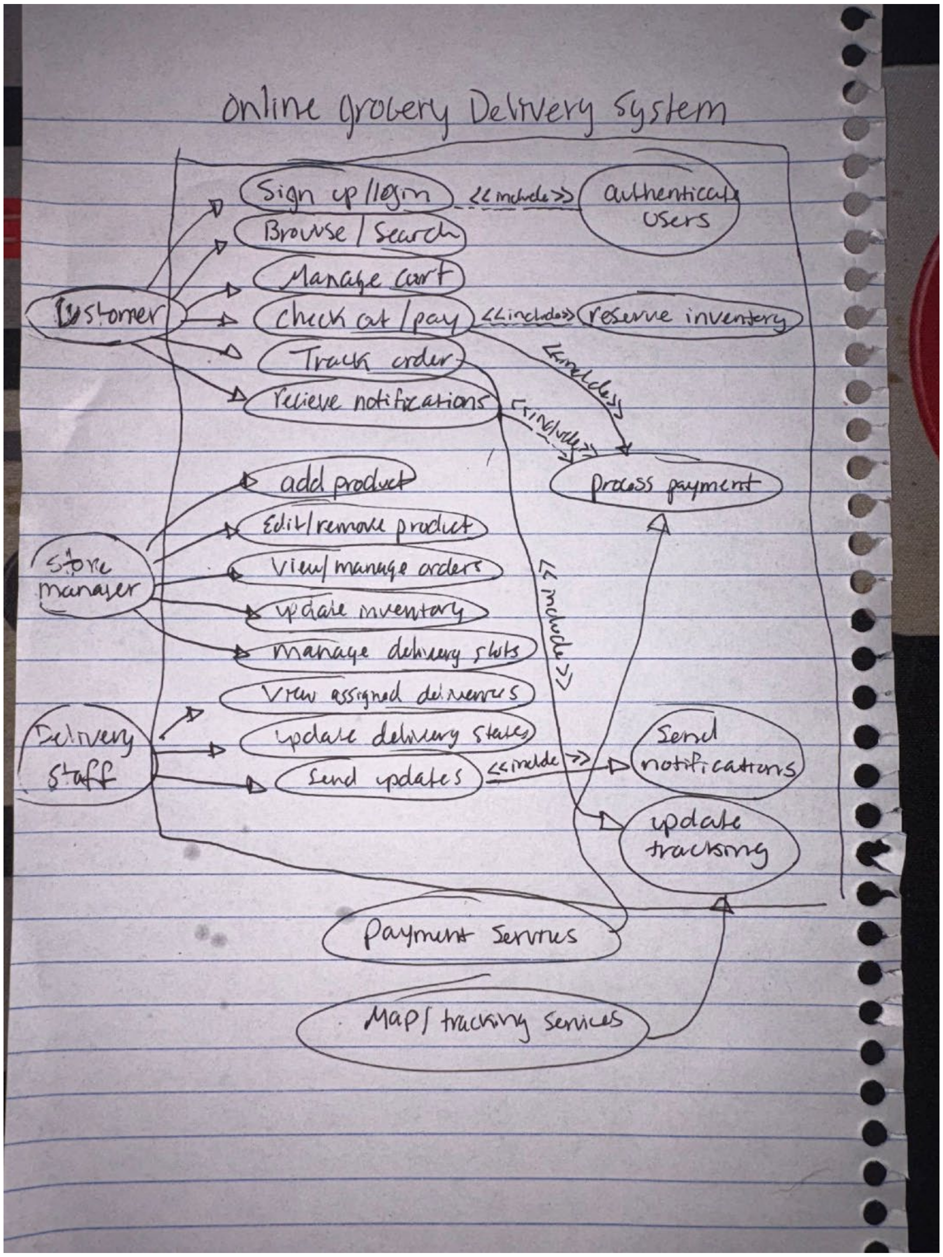
- **View assigned deliveries:** Check list of tasks (2)
- **Update status:** Mark "picked up" or "delivered" (2)
- **Send updates:** Trigger notifications to customers (2)
- **Log in/out:** Access delivery dashboard (2)

System (Total: 10)

- **Confirm order:** Validate items in stock and send confirmation (2)
- **Process payment:** Connect with payment gateway (2)
- **Send notification:** Email/SMS updates (2)
- **Update tracking:** Connect with map service (2)

- **Handle multiple users:** Keep site working under many users (2)

Use Case Diagram



Class Diagram Display

Main Classes

- **Account (abstract)**

- Attributes: accountID:int, name:string, email:string, password:string
- Operations: login(), logout(), updateProfile()
- Subclasses: Customer, StoreManager, DeliveryStaff

- **Product**
  - Attributes: productID:int, name:string, price:float, stock:int
  - Operations: addProduct(), editProduct(), removeProduct(), checkAvailability()
- **Cart**
  - Attributes: cartID:int, items:list<Product>, total:float
  - Operations: addItem(), removeItem(), calculateTotal()
- **Order**
  - Attributes: orderID:int, customerID:int, status:string, deliveryTime:datetime
  - Operations: placeOrder(), updateStatus(), trackOrder()
- **Payment (abstract)**
  - Attributes: paymentID:int, amount:float, status:PaymentStatus
  - Operations: processPayment()
  - Subclasses: CreditCard, DebitCard
- **Delivery**
  - Attributes: deliveryID:int, staffID:int, orderID:int, status:string
  - Operations: assignDelivery(), updateDeliveryStatus()
- **Notification**
  - Attributes: notificationID:int, message:string, timestamp:datetime
  - Operations: sendNotification()

Data Types & Enumerations

- **PaymentStatus:** pending, paid, failed, refunded
- **OrderStatus:** confirmed, preparing, shipped, delivered

- **AccountStatus:** active, suspended, closed

Relationships

- **Association:**
  - Customer ⇔ Order (one-to-many)
  - Order ⇔ Product (many-to-many through Cart)
  - DeliveryStaff ⇔ Delivery (one-to-many)
- **Composition:**
  - Order *includes* Cart and Payment
  - Delivery *includes* Order
- **Inheritance:**
  - Account → Customer, StoreManager, DeliveryStaff
  - Payment → CreditCard, DebitCard

Summary

This Online Grocery Delivery System is designed to make shopping easier for customers while also making sure the process of the system is easy for managers and delivery staff to operate. The main goal and benefit of the system is that it is a combination of real-time stock checking, online payments, and live delivery tracking into one system. The design covers stakeholders, actors and goals, use cases with time estimates, and a use case diagram and a class diagram description that explains how the system works.

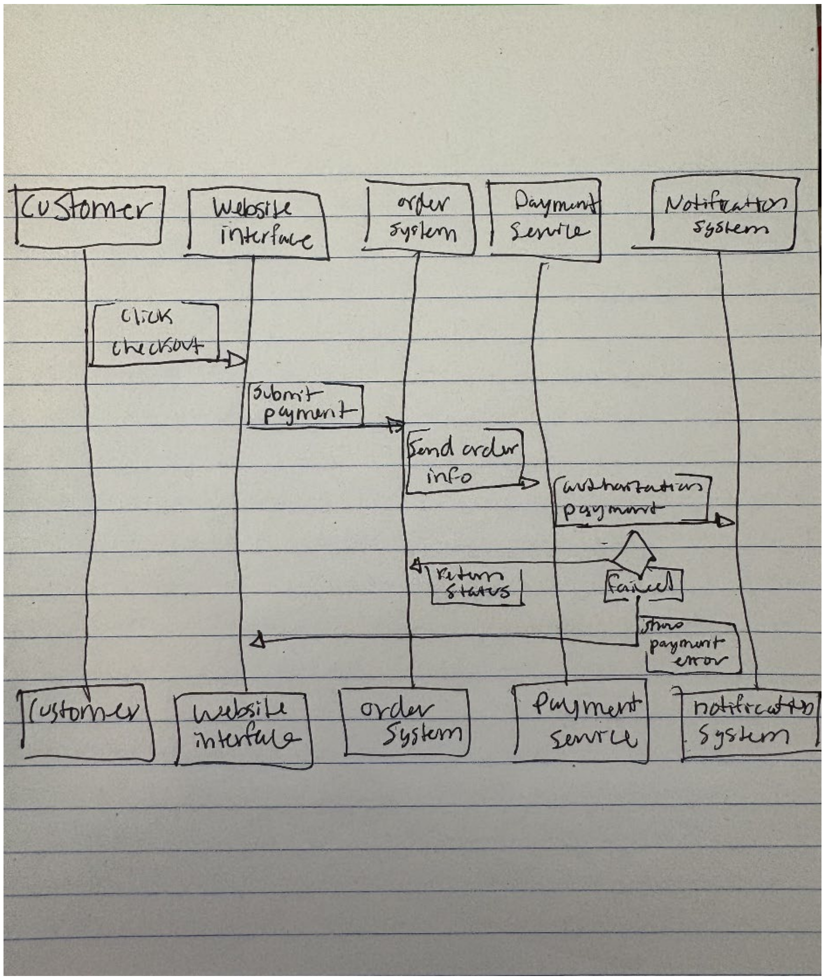
System Sequence Diagram and Activity Diagram

System Sequence Diagram 1: Checkout and Pay

Actor: Customer

Objects: Website Interface, Order System, Payment Service, Notification System

1. The customer clicks Checkout and submits payment information.
2. The website sends order details to the Order System.
3. The order System requests payment authorization from the Payment Service.
4. The p/ayment Service processes and returns a payment status.
5. If the payment is successful, the Order System confirms the order and sends a confirmation to the customer.
6. If the payment fails, the system shows a payment error message and allows the customer to try again.



System Sequence Diagram 2: Track Order
Actor: Customer

Objects: Website Interface, Order System, Tracking Service, Notification System

1. The customer selects Track Order.
2. The website sends requests to the Order System with order details.
3. The order System connects with the Tracking Service to get delivery status.
4. The Tracking Service returns the current delivery update.
5. The Order System displays the delivery progress to the customer.
6. The otification System sends updates if the delivery status changes.

Activity Diagram 1: Checkout and Pay
Initial State: Customer has items in the cart.

Actions:
- The customer clicks Checkout.
- The system checks the item availability.
- The customer enters their payment details.
- The payment Service processes the payment.
- If the payment is confirmed, the system saves the order and sends confirmation.
- If the payment fails, the system shows payment error and allows retry.

Final State: Customer receives confirmation or retry message.

Activity Diagram 2: Track Order
Initial State: Customer is logged in and selects Track Order.

Actions:
- The system retrieves order details.
- The system connects to the Tracking Service.
- The tracking Service sends current delivery status.
- The system updates and displays the delivery progress.

Final State: The customer views the latest order status or sees an error message if tracking is unavailable.

User Interface Specification

Preliminary Design:

Use case 1: Sign Up/Log in

If they have an account, the customer puts their email, username, and password into the text boxes on the login screen. Then the customer clicks the Login button to access their account. If they do not have an account, they click Sign Up and fill in their information. When they are done, they click Create Account. The system will show a welcome message when login is successful or an error message if the login failed. The steps go from Home to Login Page to Dashboard.

Use Case 2: Browse Products

To browse products customers can search or use filters to find their items. The customer can click Add to Cart to add products they want. The system then updates the page with search results or displays a message if no items are found. The steps go from Dashboard to Product Page to Category Filter to Add to Cart.

Use Case 3: Checkout and Pay

The customer clicks Checkout and is prompted to type their delivery address and payment details. After the customer clicks Confirm Payment, the system displays a message saying Payment Successful or an error if it fails. The steps go from Cart to Checkout Page to Payment Confirmation.

Use Case 4: Track Order

The customer clicks Track Order on their dashboard and is prompted to enter their order ID. The system shows a progress bar with steps including Confirmed, Packed, Out for Delivery, and Delivered. If tracking data is not found, the page shows Tracking not available yet. The steps go from Dashboard to Order History to Track Order.

Use Case 5: Manage Products

The store manager logs in and goes to the Product Management page. The manager can add, edit, or remove products by typing the product name, price, and number in stock. After the manager clicks Save or Remove, the system shows a confirmation message. The steps go from Manager Dashboard to Product List to Add or Edit Product.

User Effort Estimation

Scenario 1: Customer Checkout and Pay

1. Click Cart
2. Click Checkout
3. Click Delivery Address box and type 20 characters
4. Click Card Number box and type 16 numbers
5. Click Confirm Payment

Total: 5 clicks and around 36 keystrokes.

Scenario 2: Track Order

1. Click Track Order
2. Click Order ID field and type 6 numbers
3. Click Search

Total: 3 clicks and around 6 keystrokes.

Scenario 3: Add Products

1. Click Products
2. Click Add Product
3. Click and fill in name, price, and quantity
4. Click Save Product

Total: 6 clicks and about 30 keystrokes.

Scenario 4: Log In

1. Click Login
2. Click Username box and type 6 characters
3. Click Password box and type 6 characters
4. Click Submit

Total: 4 clicks and about 12 keystrokes.

Scenario 5: Browse Products

1. Click Search Bar and type 10 characters
2. Click Filter by Category
3. Click Add to Cart for two items

Total: 4 clicks and about 10 keystrokes.

Assignment6-Traceability Matrix

System Requirements

No.	Priority Weight (1 - 5: 1: lowest, 5: highest)	Description
REQ-1	5	Customers can sign up, log in, and manage their account.
REQ-2	5	Customers can browse products, add to cart, and place orders
REQ-3	5	Store managers can add, edit, or remove products.
REQ-4	5	System shows product availability
REQ-5	3	Customers can pay by credit/debit card.
REQ-6	3	Delivery staff see assigned orders and update their status
REQ-7	3	Customers can track delivery status/progress
REQ-8	1	System sends confirmed, shipped, and delivered order notifications

Use Cases

No.	Description
UC1	Sign up / Log in
UC2	Browse Products
UC3	Add to Cart
UC4	Checkout and Pay
UC5	Track Order
UC6	Manage Products
UC7	Update Stock Status
UC8	Delivery Staff: Manage Deliveries

Traceability Matrix

Req’t	**PW**	**UC1**	**UC2**	**UC3**	**UC4**	**UC5**	**UC6**	**UC7**	**UC8**
REQ-1	5	x							
REQ-2	5		x	x	x				
REQ-3	5						x	x	
REQ-4	5		x	x				x	
REQ-5	3				x				
REQ-6	3								x
REQ-7	3					x			
REQ-8	1				x	x			x
Max PW		**5**	**5**	**5**	**5**	**3**	**5**	**5**	**3**
Total PW		**5**	**10**	**10**	**9**	**4**	**5**	**10**	**4**