LAB CAT

Project Documentation - 3

APRAJITA NANDKEULIAR
22BCE0544

Project Title

SwiftDrop: Fast Delivery Logistics System

Abstract

This project explores the AI logistics framework employed by SwiftDrop to enhance its delivery efficiency. The focus is on the system's ability to identify the nearest dark store with the required items in stock and to assign an available delivery partner who can fulfill the order within a stringent time frame. Additionally, the project will delve into how the AI logistics system determines optimal delivery routes to ensure timely deliveries.

When a customer places an order through SwiftDrop's app, the system immediately identifies the nearest dark store with the required items in stock. These warehouses are meticulously organized to allow packers to locate and pack items in less than 60 seconds. SwiftDrop's AI technology not only tracks inventory in real-time but also maintains product assortment, ensuring that customers always have access to a variety of goods, including fresh produce, dairy, and everyday essentials.

Once the order is packed, SwiftDrop's AI logistics system locates the nearest available delivery partner. By considering real-time location data and the partner's availability status, the system assigns the task to ensure minimal delays. The AI also maps out the fastest delivery route, taking into account factors like traffic patterns, road conditions, and weather. By bypassing congestion and avoiding unnecessary detours, delivery personnel can reach customers' doorsteps quickly and efficiently.

Methodology

The methodology for the **AI Logistics System for Ultra-Fast Delivery Optimization** follows a **step-by-step** approach:

1. Order Placement by Customer

- The customer selects items and places an order via the app.
- The system verifies item availability in nearby **dark stores**.

2. Dark Store Selection

- AI checks inventory across multiple **dark stores**.
- The nearest store **with stock availability** is selected.

3. Delivery Personnel Assignment

- AI identifies the closest **available** delivery partner.
- The partner is assigned based on **real-time location** and **workload**.

4. Route Optimization (Special Feature)

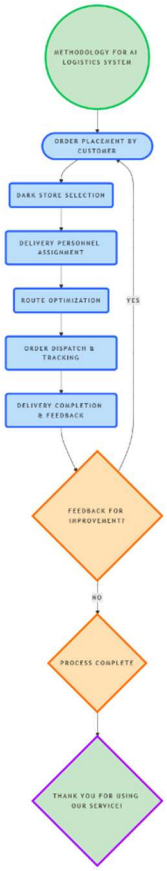
- The **best delivery route** is calculated using:
    - **Real-time traffic data**
    - **Historical traffic patterns**
    - **Weather conditions**

5. Order Dispatch & Tracking

- Order is packed and dispatched.
- Customers track **real-time** updates on the app.

6. Delivery Completion & Feedback

- Delivery partner completes the task.
- Customers provide feedback for **system improvement**.



Functional Requirements

1. Customer Order Placement:
   - Users must be able to place orders via a user-friendly mobile application.
   - The app should allow customers to browse items and check real-time availability.
2. Dark Store Identification:
   - The system must automatically identify the nearest dark store based on the customer's GPS location.
   - It should check inventory levels in real-time to ensure that ordered items are in stock.
3. Delivery Partner Assignment:
   - The logistics system should identify available delivery partners based on proximity to the dark store and their current status (available or busy).

- Notifications should be sent to delivery partners about new assignments.

4. Route Optimization:
   - The AI must calculate the best route for delivery partners using real-time traffic data and historical traffic patterns.
   - The system should provide turn-by-turn navigation to delivery partners.
5. Order Tracking:
   - Customers should be able to track their orders in real-time from dispatch to delivery.
   - Notifications about order status changes (e.g., dispatched, out for delivery) should be sent to customers.

Non-Functional Requirements

1. Performance:
   - The system should process orders and assign dark stores within seconds.
   - Delivery routes must be calculated in real-time to ensure timely deliveries within 10-15 minutes.

2. Scalability:
   - The architecture must support scaling as demand increases or as new dark stores are added.
   - It should handle multiple simultaneous orders without degradation in performance.

3. Reliability:
   - The system must ensure high availability with minimal downtime.
   - Data integrity must be maintained across all transactions.

4. Security:
   - Customer data and payment information must be securely handled and encrypted.
   - Access controls should be implemented to protect sensitive information.

5. User Experience:
   - The mobile application must have an intuitive interface for easy navigation.
   - Response times for user actions (e.g., placing an order) should be minimal.

Justification of Process Model

The Agile Development Model is deemed most appropriate for this project due to several key factors like its ability to be flexible and adaptable, incremental delivery, constant collaboration and communication with the stakeholders, continuous improvement and rapid prototyping. In summary, by employing an Agile approach, this project can effectively address the complexities of developing AI logistics for Zepto's dark store model while ensuring responsiveness to market demands and user needs.

Work Progress According to Defined WBS

Phase 1: Requirement Analysis (Time Taken: 2 Weeks)

Tasks Completed:

1.1 Identify Functional Requirements:

- Defined key features such as dark store selection, delivery personnel assignment, route optimization, and real-time tracking.

1.2 Identify Non-Functional Requirements:

- Addressed performance (order processing within seconds), scalability (handling multiple orders), reliability (high availability), security (customer data protection), and user experience (minimal response time).

1.3 Conduct Stakeholder Interviews:

- Gathered insights from Zepto managers, delivery personnel, and customers on system expectations.

1.4 Define User Stories:

- Created user stories for customers (placing orders, tracking delivery), delivery personnel (accepting and completing deliveries), and dark store managers (inventory tracking).

1.5 Document Requirement Specification:

- Prepared detailed Software Requirement Specification (SRS) outlining functional and non-functional requirements.

Phase 2: System Design (Time Taken: 3 Weeks)

Tasks Completed:

2.1 Architecture Design for AI Logistics System:

- Defined system architecture, including data flow, API interactions, and AI-powered decision-making modules.

2.2 Database Schema Design for Inventory & Personnel Tracking:

- Designed database schema for tracking dark store inventory, delivery personnel locations, and order processing.

2.3-2.5 Algorithm Design for Dark Store Selection, Delivery Personnel Assignment & Route Optimization:

- Developed logic for selecting the nearest dark store with required stock.
- Created personnel assignment logic based on real-time location and workload.
- Designed a route optimization algorithm using real-time and historical traffic data.

2.6 API Design for Third-Party Integration:

- Designed APIs for integrating with maps (Google Maps), payment gateways, and notification services.

2.7 UI/UX Design:

- Wireframed and designed user interfaces for customers, delivery personnel, and administrators.

Phase 3: Implementation (Time Taken: 5 Weeks)

Tasks Completed:

3.1 Develop Dark Store Inventory Tracking Module:

- Built an inventory management system to track product availability at dark stores.

3.2 Implement Delivery Personnel Tracking & Assignment Module:

- Integrated real-time GPS tracking for delivery personnel.
- Developed an availability status update system.
- Implemented logic to assign personnel based on proximity and readiness.
- Tested assignment logic with simulated orders.

3.3 Develop Route Optimization Module:

- Implemented an AI-powered route optimization algorithm that factors in real-time and historical traffic patterns.

3.4 Integrate Mapping & Notification APIs:

- Connected the system with Google Maps API for route planning.
- Integrated notification systems for order updates via SMS/email.

3.5 Build User Interfaces (Web/App):

- Developed mobile and web apps for customers, delivery personnel, and admins.

3.6 Implement Order Tracking & Notification Systems:

- Enabled real-time order tracking for customers and delivery personnel.

Phase 4: Testing and Validation (Time Taken: 3 Weeks)

Tasks Completed:

4.1 Unit Testing:

- Created and executed test cases for inventory management, delivery assignment, and route optimization.
- Performed regression testing after every major update.

4.2 Integration Testing:

- Verified that modules such as order placement, personnel assignment, and routing work seamlessly together.

4.3 Performance Testing:

- Simulated peak order times to test system scalability.

4.4 User Acceptance Testing (UAT):

- Conducted real-world testing with selected delivery personnel and customers.
- Collected feedback to refine UI and performance.

4.5 Bug Fixing & Optimization:

- Addressed performance bottlenecks and security vulnerabilities.

Phase 5: Deployment & Maintenance (Time Taken: 2 Weeks)

Tasks Completed:

5.1 Deploy System on Production Servers:

- Deployed the system on cloud infrastructure with high availability.

5.2 Configure & Test Third-Party APIs:

- Ensured smooth integration with maps, notifications, and payment services.

5.3 Monitor System Performance Post-Deployment:

- Set up monitoring tools for real-time system tracking and debugging.

5.4 Regularly Update Algorithms:

- Implemented a feedback-driven approach for refining delivery assignment and route optimization.

5.5 Provide User Support & Gather Feedback:

- Established a support team to assist users.
- Collected feedback for continuous improvements.

Total Project Duration:

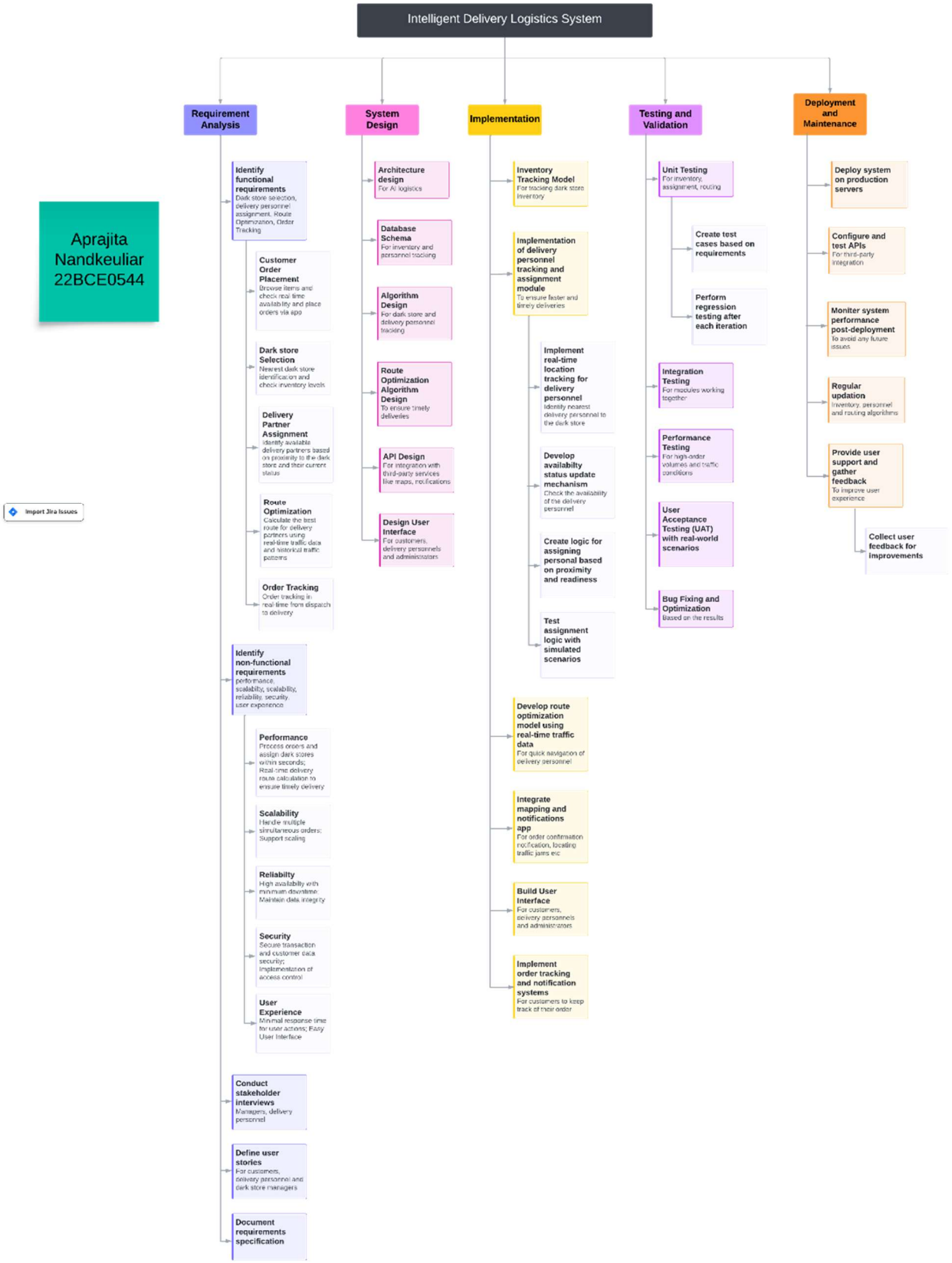
15 Weeks (Approximately 3.5 months)

Learning Outcomes

By using ER diagram, context flow diagram, data flow diagram and work breakdown structure, I learned the following:

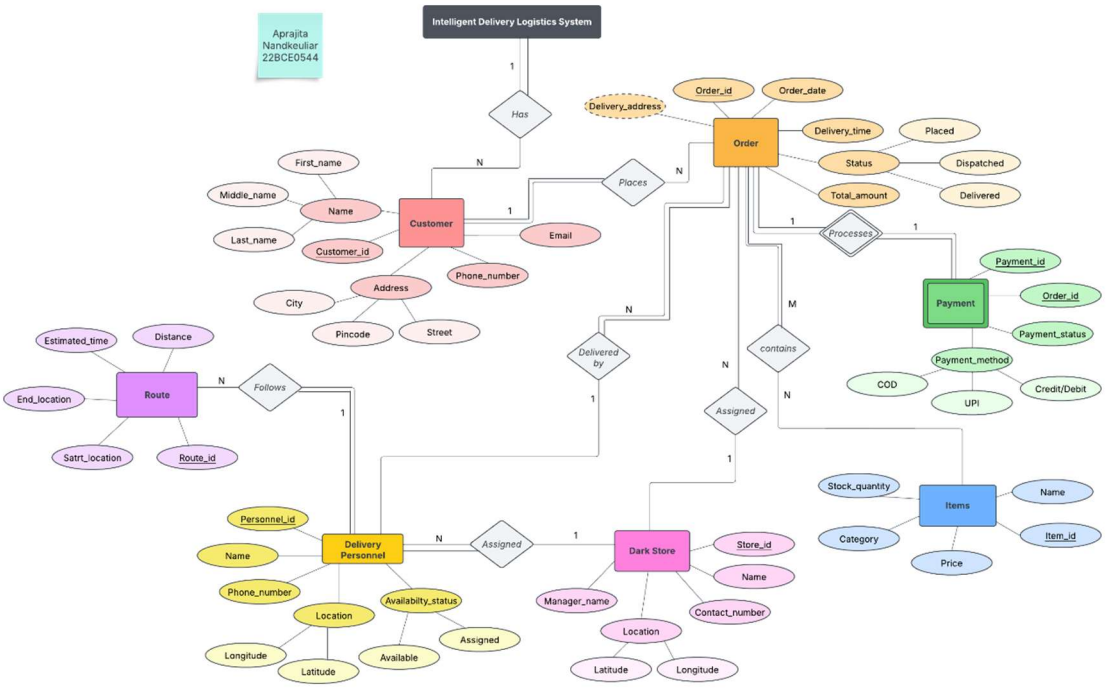
1. Work Base Structure

A work breakdown structure (WBS) is a visual, hierarchical and deliverable-oriented deconstruction of a project. It helps in project management by dividing the project into manageable sections, ensuring clarity and efficiency in execution. It improves planning, scheduling, and resource allocation, ensuring that all aspects are covered.


Intelligent Delivery Logistics System
Requirement Analysis
System Design
Implementation
Testing and Validation
Deployment and Maintenance
Aprajita Nandkeuliar 22BCE0544
Import Jira Issues
Identify functional requirements
Dark store selection, delivery personnel assignment, Route Optimization, Order Tracking
Customer Order Placement
Browse items and check real time availability and place orders via app
Dark store Selection
Nearest dark store identification and check inventory levels
Delivery Partner Assignment
Identify available delivery partners based on proximity to the dark store and their current status
Route Optimization
Calculate the best route for delivery partners using real-time traffic data and historical traffic patterns
Order Tracking
Order tracking in real-time from dispatch to delivery
Identify non-functional requirements
performance, scalability, reliability, security, user experience
Performance
Process orders and assign dark stores within seconds; Real-time delivery route calculation to ensure timely delivery
Scalability
Handle multiple simultaneous orders; Support scaling
Reliabilty
High availability with minimum downtime; Maintain data integrity
Security
Secure transaction and customer data security; Implementation of access control
User Experience
Minimal response time for user actions; Easy User Interface
Conduct stakeholder interviews
Managers, delivery personnel
Define user stories
For customers, delivery personnel and dark store managers
Document requirements specification
Architecture design
For AI logistics
Database Schema
For inventory and personnel tracking
Algorithm Design
For dark store and delivery personnel tracking
Route Optimization Algorithm Design
To ensure timely deliveries
API Design
For integration with third-party services like maps, notifications
Design User Interface
For customers, delivery personnels and administrators
Inventory Tracking Model
For tracking dark store inventory
Implementation of delivery personnel tracking and assignment module
To ensure faster and timely deliveries
Implement real-time location tracking for delivery personnel
Identify nearest delivery personnel to the dark store
Develop availabilty status update mechanism
Check the availability of the delivery personnel
Create logic for assigning personal based on proximity and readiness
Test assignment logic with simulated scenarios
Develop route optimization model using real-time traffic data
For quick navigation of delivery personnel
Integrate mapping and notifications app
For order confirmation notification, locating traffic jams etc
Build User Interface
For customers, delivery personnels and administrators
Implement order tracking and notification systems
For customers to keep track of their order
Unit Testing
For inventory, assignment, routing
Create test cases based on requirements
Perform regression testing after each iteration
Integration Testing
For modules working together
Performance Testing
For high-order volumes and traffic conditions
User Acceptance Testing (UAT) with real-world scenarios
Bug Fixing and Optimization
Based on the results
Deploy system on production servers
Configure and test APIs
For third-party integration
Moniter system performance post-deployment
To avoid any future issues
Regular updation
Inventory, personnel and routing algorithms
Provide user support and gather feedback
To improve user experience
Collect user feedback for improvements

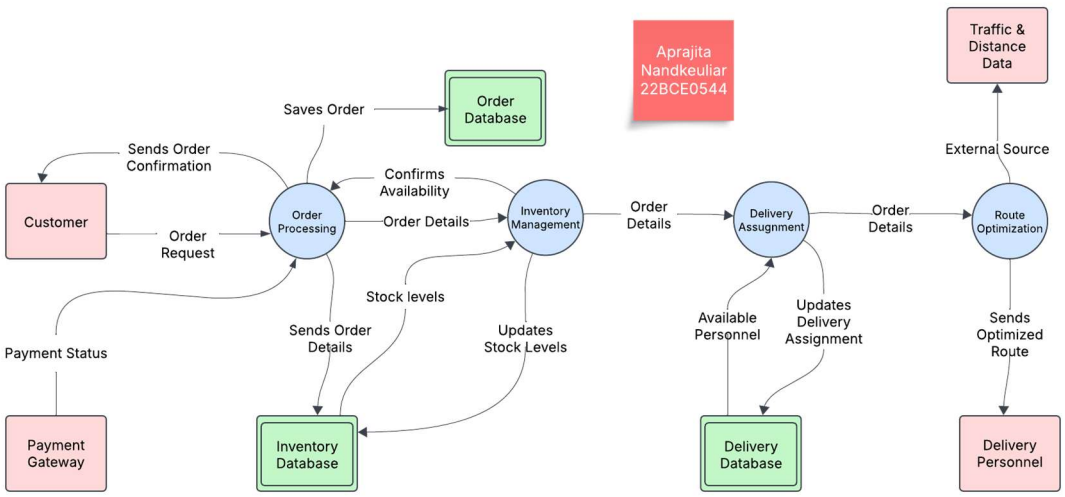
2. ER Diagram

An Entity-Relationship (ER) Diagram is a graphical representation of a database that illustrates the relationships between entities within a system. It helps to construct efficient database design by defining primary keys, foreign keys, and constraints.



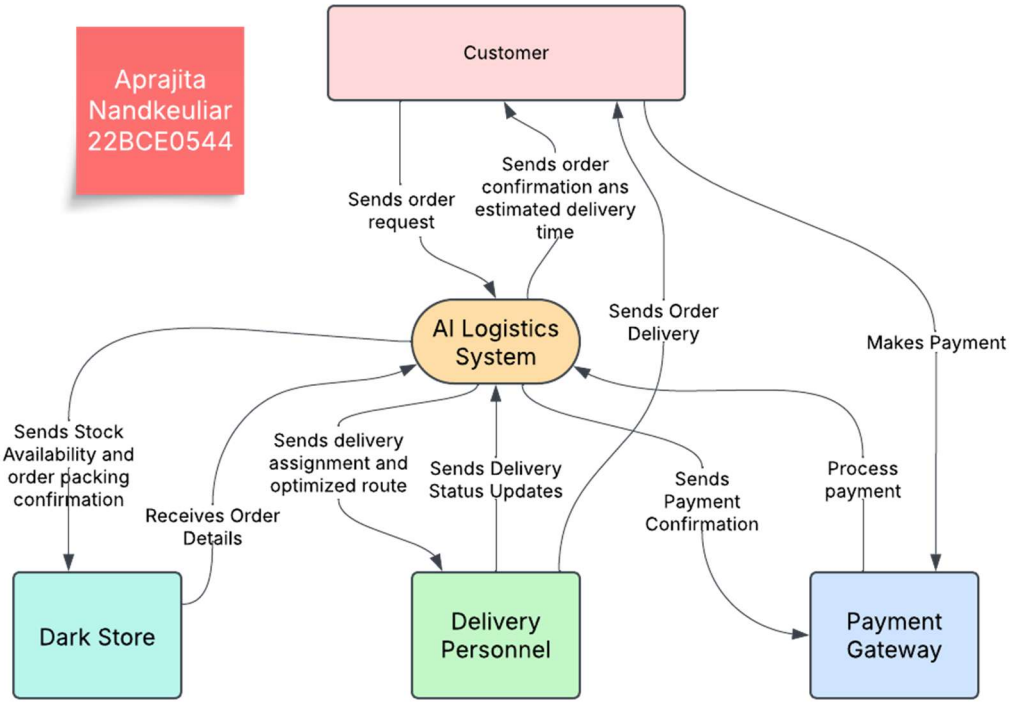
3. Data Flow Diagram

A Data Flow Diagram is a graphical representation of how data moves through a system, illustrating the processes, data stores, external entities, and data flow between them. It helps in understanding the system's inputs, outputs, and processing steps at various levels of detail (e.g., Level 0, Level 1, Level 2). It also improves system design by ensuring logical workflow and proper data handling



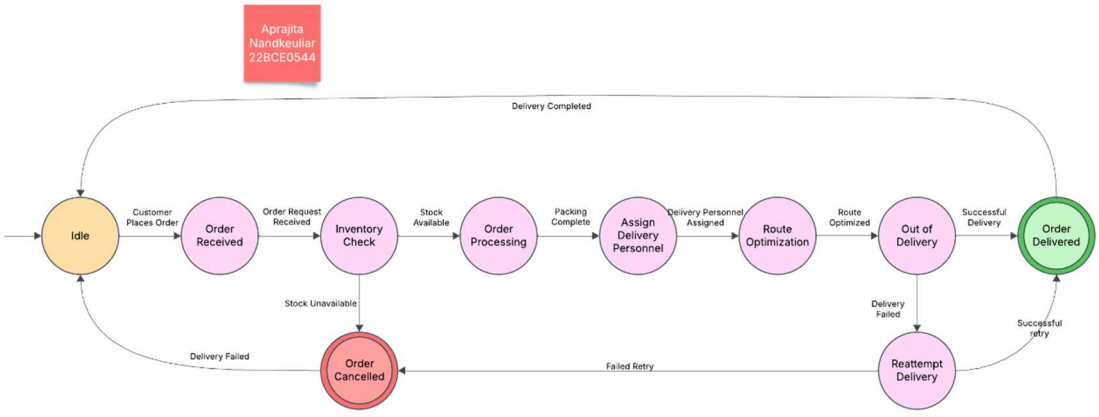
4. Context Flow Diagram

A Context Flow Diagram is the highest-level DFD (Level 0) that provides an overview of the entire system by how information is processed, store, communicated, between subsystems, enhancing the understanding of system functionality. It represents the system as a single process with input and output data flows.



5.State transition diagram

State Transition Diagram for AI Logistics System (Zepto) A State Transition Diagram (STD) represents the different states of the AI Logistics System and how it transitions between them based on events like customer orders, inventory checks, and delivery updates.



Conclusion

SwiftDrop enhances delivery efficiency by leveraging AI-driven inventory tracking, real-time delivery personnel assignment, and route optimization. By integrating advanced algorithms and real-time data processing, the system ensures faster order fulfilment while maintaining scalability, reliability, and security. Rigorous testing validates system performance under high-order volumes. The project demonstrates the potential of AI in optimizing logistics, reducing delivery times, and improving customer satisfaction. Future enhancements could include predictive analytics for demand forecasting and further automation. This system lays a strong foundation for next-generation ultra-fast delivery solutions.