Software Engineering Software Requirements Specification (SRS) Document

< MarketHub>

<21th Dec2025>

<v 1.0>

Revision

Version	Primary Author(s)	Role	Description of Version	Date Completed
v1.0	Entire Team	QA	Initial Requirement Documentation	05/26/2025
V1.1	Entire Team	QA	Added Non-functional Requirement for optimization of product display page.	-

Review & Approval

Requirements Document Approval History

Approving Party	Project Role	Version Approved	Signature	Date
Bishal	Trainer	V1.0		

Requirements Document Review History

Reviewer	Project Role	Version Reviewed	Signature	Date
	Project Manager	V1.0		

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this document is to provide a detailed description of the **MarketHub Online Shop** application. It outlines the software's goals, core functionalities, user interfaces, and the technical constraints under which the system must operate. This document is intended for project stakeholders, developers, and testers.

1.2. Document Conventions

The IEEE template for System Requirement Specification Documents was used to create this document.

1.3. Intended Audience

- ✓ **Customers:** Users interested in browsing and purchasing products online.
- ✓ **Administrators:** Staff responsible for managing inventory, orders, and user accounts.
- ✓ **Developers and Testers:** Team members building and verifying the system.
- ✓ **Project Managers/Stakeholders:** Individuals managing the project lifecycle and requirements.

1.4. Scope

MarketHub is an online shopping platform designed to allow customers to browse products, manage carts, place orders, and track purchases, while enabling administrators to manage products, inventory, and orders.

The scope of this system includes user authentication, product catalog management, shopping cart functionality, secure payment processing, order tracking, and notification services. Features outside this scope, such as third-party seller onboarding and advanced analytics, are excluded from the current version of the system.

- ✓ **Administrators** can create product listings with details such as name, description, price, and category.
- ✓ **Customers** can view the product catalog, add items to a cart, and track their purchase history.

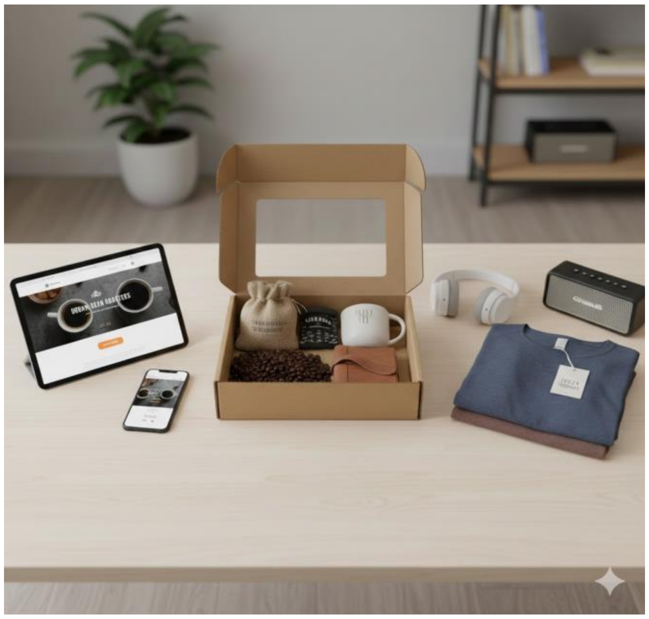
1.5. References

- ✓ IEEE Template for System Requirement Specification Documents https://goo.gl/nsUFwy\
- ✓ Payment Gateway API Documentation (e.g., Stripe or PayPal).
- ✓ The Agile Manifesto and Scrum Guide (2020)

2. General Description

2.1. Product Perspective

MarketHub is an evolving ecosystem. Unlike traditional models, the product perspective is **iterative**. Each Sprint will produce a working increment of the software. Feedback from **Sprint Reviews** will be used to update this SRS and the Product Backlog, ensuring the shop meets current market trends.



MarketHub is a standalone web-based application that operates within a client-server architecture. The system interacts with external services such as payment gateways and shipping providers while remaining independent of other internal software systems. The application is designed to run on modern web browsers.

2.2. Product Features

The MarketHub system provides the following high-level features:

Product Management

- ✓ Enables administrators to create, update, and remove product listings inventory, fulfillment, including product name, description, price, category, images & Tracking numbers.
- ✓ Supports inventory visibility by reflecting product availability status to customers.

User Shopping and Order Management

- ✓ Allows users to browse, search, and filter products based on categories and keywords.
- ✓ Enables users to add products to a shopping cart and place orders.
- ✓ Allows users to view their order history and cancel orders that have not yet been shipped.

Notification Services

- ✓ Automatically notifies users regarding order confirmations, shipping updates, delivery status, and promotional offers.

Informational and Account Management Features

- ✓ Provides informational content such as company details, licensing information, and application version details.
- ✓ Allows users to manage personal profiles, delivery addresses, payment methods, and logout sessions.

2.3. User Class and Characteristics

- ✓ **Customers:** Regular users who browse the shop and make purchases. Authentication is required via a secure login system.
- ✓ **Shop Admin:** Users with elevated permissions to manage inventory and view sales reports.

2.4. Operating Environment

For Web Application:

- ✓ **Web Application:** Accessible via any modern browser (Chrome, Firefox, Safari etc.) on any operating system.
- ✓ **Mobile Application:** Optimized for Android devices (Version 8.0 and above).

2.5. Constraints

The following constraints impose limitations on the design, development, and operation of the MarketHub system:

- ✓ **Regulatory and Legal Constraints**

  The system shall comply with applicable data protection and privacy regulations, including GDPR and relevant local IT laws.
  Justification: MarketHub processes personal and transactional user data, making legal compliance mandatory to ensure data protection and avoid regulatory penalties.

- ✓ **Third-Party API Constraints**

  The system is constrained by usage limits, availability, and policies of external payment gateways and shipping service APIs.

  Justification: Core functionalities such as payment processing and shipment tracking rely on third-party services that may impose request rate limits or experience service interruptions.

- ✓ **User Interface and Device Constraints**

  The system shall maintain responsiveness across supported screen sizes and devices.
  Justification: Users may access the system from desktops, tablets, and mobile devices, requiring adaptable layouts to ensure usability.

- ✓ **Network and Performance Constraints**

  System performance depends on internet connectivity and server infrastructure capacity.
  Justification: As a web-based application, stable network communication is essential for real-time operations such as checkout and order status updates.

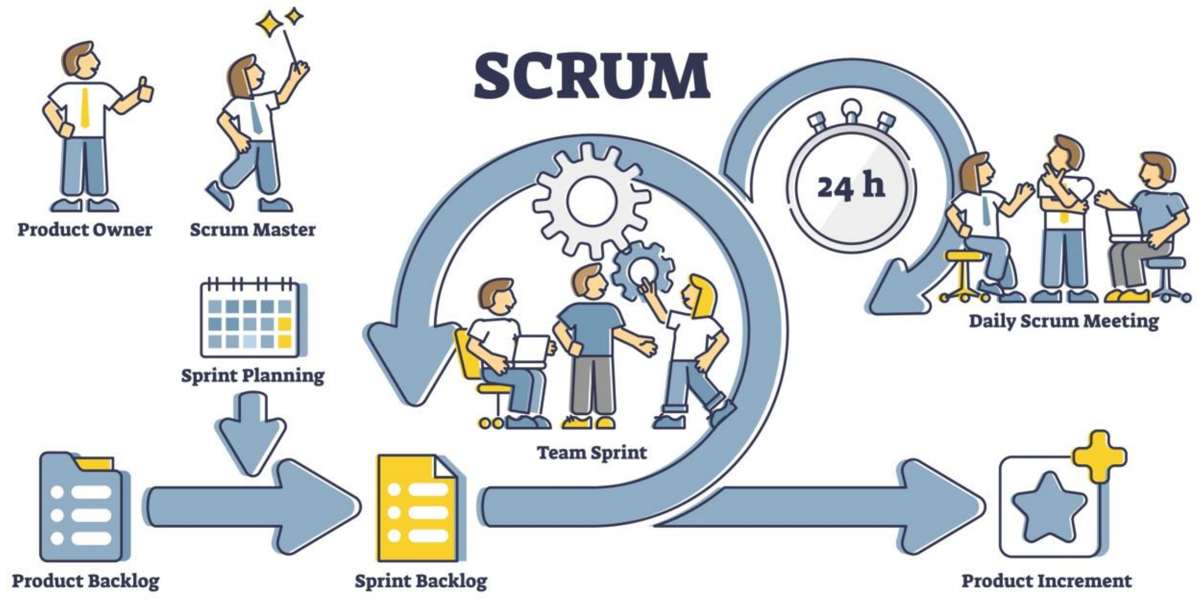
- ✓ **Security Constraints**

  The system shall enforce secure authentication and data protection mechanisms for all user interactions.

  Justification: The platform handles sensitive financial and personal data, making strong security controls essential.

2.6. Assumptions and Dependencies

- ✓ The project assumes an **active Product Owner** is available to prioritize the backlog every Sprint
- ✓ It is assumed that optional system features may be deferred or excluded if they conflict with technical feasibility, performance constraints, or external dependency limitations.
- ✓ The system depends on external libraries for secure payment processing and database management.
- ✓ Dependency on third-party payment providers (Khalti) remaining free or cost-effective for the platform.
- ✓ The system will use a reliable shipping partner API for tracking shipments.
- ✓ Dependencies include the availability of a stable testing environment for continuous integration/continuous deployment (CI/CD).



2.7. User documentation

✓ User Manual for Customers
✓ User Manual for Administrators
✓ Online Help and FAQs
✓ Installation and Configuration Guide (for Admin users)

3. System Requirements

3.1. Functional requirements

3.1.1. Authentication System

The authentication system shall ensure that only authorized users can access protected features of the MarketHub application based on their assigned roles and permissions.

Functional Requirements:

S.N.	Requirement	Priority
REQ-1	User Registration & Login	Must Have
REQ-2	Secure Password Reset (Email-based)	Must Have
REQ-3	Multi-Factor Authentication (MFA)	Should Have

- ✓ Users must be able to create an account using an email address or via social logins (e.g., Google) to participate in the shop ecosystem.
- ✓ The system shall provide a secure login session that persists across browser refreshes but times out after a period of inactivity.
- ✓ If a user forgets their password, the system shall provide a "Forgot Password" link to send a secure reset token to their registered email address.

3.1.2. Product Catalog & Management

This module allows users to browse items and enables the **Product Owner/Admin** to manage the shop’s inventory

Functional Requirements:

S.N.	Requirement	Priority
REQ-4	Create/Update/Delete Product Listing	Must Have
REQ-5	Search and Category Filtering	Must Have
REQ-6	Inventory Stock Tracking	Should Have

- ✓ The **Admin** can create new product listings including images, price, description, and category.
- ✓ Users can search for products by name or filter them by categories such as "Electronics," "Home," or "Clothing."
- ✓ The system shall display a "Sold Out" badge automatically when stock levels for a specific item reach zero.

3.1.3. Shopping Cart & Checkout

The core commerce engine where users select products and finalize their purchases.

Functional Requirements:

S.N.	Requirement	Priority
REQ-7	Virtual Shopping Cart Management	Must Have
REQ-8	Secure Payment Gateway Integration	Must Have
REQ-9	Apply Promotional/Discount Codes	Could Have

- ✓ Users can add, remove, and modify the quantity of items in their virtual shopping cart.
- ✓ The system shall calculate the total cost, including taxes and shipping fees, before the final checkout step.
- ✓ The checkout process must securely hand off transaction data to third-party providers (e.g., Stripe, Khalti-IME etc.) without storing sensitive credit card numbers locally.

3.1.4. Order Tracking & History

This module provides transparency to the user regarding their past and current transactions.

Functional Requirements:

S.N.	Requirement	Priority
REQ-10	View Active Order Status	Must Have
REQ-11	View Comprehensive Order History	Should Have
REQ-12	Download Invoice (PDF)	Should Have

- ✓ Users can view a real-time status of their active orders (e.g., "Processing," "Shipped," "Delivered").
- ✓ The system shall maintain a record of all past successful transactions for user reference.
- ✓ Admins can update the tracking number for an order, which immediately reflects in the user's "Active Orders" page.

3.1.5. Notification System

Automated alerts keep users informed about their account and purchase activity.

S.N.	Requirement	Priority
REQ-13	Order Confirmation Notifications	Must Have
REQ-14	Push Notifications for Sales/Promos	Could Have

- ✓ A notification (Email or Push) will be sent to the user upon successful order placement.
- ✓ The system shall alert the user when an item on their wishlist goes on sale or is back in stock.

4. External Interface Requirements

4.1. User Interfaces

MarketHub shall provide a user-friendly and responsive graphical interface designed for both desktop and mobile users.

- ✓ **Design Standards:** The application shall provide a consistent and intuitive user interface across web and mobile platforms, following widely accepted usability and accessibility standards.
- ✓ **Web Portal:** A browser-based interface featuring a product dashboard, shopping cart sidebar, and account management settings.
- ✓ **Mobile App:** A touch-optimized interface for Android and iOS, focusing on smooth navigation between product categories and a simplified "One-Tap" checkout process.

4.2. Hardware Interfaces

The application is designed to be hardware-agnostic, requiring only standard internet-connected devices.

- ✓ **Web Interface:** Any computer or tablet capable of running a modern web browser (Chrome, Firefox, Safari, Edge etc.).
- ✓ **Mobile Interface:** Smartphones running **Android (Version 8.0+)** or **iOS (Version 13.0+)**.
- ✓ **Storage:** Local caching on devices will be used to improve load times for product images and search results.

4.3. Communications Interfaces

Secure communication is critical for protecting user payment data and personal information.

- ✓ **Protocols:** MarketHub shall use the **HTTPS (SSL/TLS)** protocol for all communication over the internet to ensure data encryption.
- ✓ **Data Format:** Data exchange between the front-end (web/mobile) and the back-end servers will be handled via **RESTful APIs** using **JSON** format.
- ✓ **Email/SMS:** Integration with communication services (like Twilio or SendGrid) for sending automated order confirmations and shipping updates.

4.4. Software Interfaces

The system relies on various software layers and third-party integrations to function effectively.

- ✓ **Operating Systems:** Compatible with Windows, macOS, and Linux for the web version, and Android/iOS for mobile.
- ✓ **Database: Firebase or MongoDB** will be used to store product details, user profiles, and order records.
- ✓ **Payment APIs:** Integration with **Stripe, Khalti-IME, or Square** for processing credit cards and digital wallets securely.
- ✓ **Shipping APIs:** Connections to logistics providers (e.g., FedEx, UPS, Nepal Can Move) to provide real-time shipping rates and tracking status updates.

5. Non-Functional Requirements

5.1. Performance Requirements

To ensure a smooth shopping experience and prevent cart abandonment, the following performance benchmarks must be met:

- ✓ **Response Time:** The system shall provide a response within 3 seconds after any user action, such as adding an item to the cart or loading a product page.
- ✓ **User-Interface:** The UI screens, especially the search results page, shall respond and be fully interactive within 5 seconds.
- ✓ **Reduced Real-Time Latency:** The system must minimize latency during the checkout process to ensure payment gateway handshakes are completed without timeout errors.
- ✓ **Scalability:** The system should be capable of handling a 300% increase in traffic during seasonal sales (e.g., Black Friday, 11.11) without performance degradation.

5.2. Safety Requirements

Safety in an e-commerce context focuses on data integrity and legal compliance:

- ✓ **Data Integrity:** To ensure that users do not lose their cart data or order history due to a crash, the development team shall implement regular system updates and state-persistence.
- ✓ **Bug Tracking:** A bug tracker shall be available for users and testers to report issues so they can be addressed in the next **Sprint**.
- ✓ **Privacy Policy:** MarketHub shall strictly maintain a privacy policy regarding user data, ensuring it is abided by local IT laws and GDPR/CCPA regulations where applicable.
- ✓ **API Compliance:** The application shall strictly maintain the terms and conditions of integrated third-party APIs (e.g., Stripe, Khalti-IME, or Google Maps)

5.3. Security Requirements

Security is paramount for protecting financial transactions and user identities:

- ✓ **Authentication:** Users can only access checkout and order history after fulfilling two conditions: successful login with verified credentials and a valid session token.
- ✓ **Encryption:** All sensitive data, including user credentials and payment-related tokens, shall be encrypted both in transit and at rest using industry-standard cryptographic techniques such as AES-256 and TLS
- ✓ **Access Control:** Only authorized Shop Admins shall have access to the administrative dashboard and API keys for backend services.

5.4. Software Quality Attributes

These attributes ensure the long-term health and reliability of the platform:

- ✓ **Maintainability:**
  - o **Error Logs:** The system shall keep a detailed error log of all server-side and client-side issues to facilitate quick debugging.
  - o **Routine Backup:** The system shall perform automated backups of the database on a daily basis to prevent permanent data loss.
- ✓ **Software Quality:** A high-quality framework (such as React or Flutter) shall be used to produce bug-free and robust software that meets all user satisfaction requirements.
- ✓ **Availability:** The system shall aim for "three nines" availability (99.9%), ensuring the shop is accessible to customers at all times.

6. Use Case Statements

6.1 Use Case: Customer Registration

Actor : Customer

Description : A new customer creates an account to be able to make purchases

Preconditions : Customer has a valid email address.

Basic Flow:

- Customer navigates to the registration page.
- Customer enters personal information (name, email, password).
- System validates the information.
- System creates a new account.
- System sends a confirmation email.
- Customer is redirected to the account dashboard.

Alternative Flows:

- If the email is already registered, system displays an error message.
- If any required field is missing or invalid, system displays appropriate error messages.

Postconditions: Customer has a registered account and is logged in.

6.2 Use Case: Product Purchase

Actor: Customer

Description: A customer browses products, adds items to cart, and completes a purchase.

Preconditions: Customer is logged in.

Basic Flow:

- Customer searches or browses for products.
- Customer views product details.
- Customer adds product(s) to cart.
- Customer reviews cart contents.
- Customer proceeds to checkout.
- Customer enters or selects shipping information.
- Customer selects payment method.
- Customer confirms order.
- System processes payment.
- System creates order.
- System sends order confirmation.

Alternative Flows:

- If a product is out of stock, customer is notified and cannot add it to cart.
- If payment fails, customer is notified and can retry or change payment method.

Postconditions: Customer has a confirmed order and the inventory is updated.

6.3 Use Case: Cancel Order

Actor: Customer

Description: A customer cancels an order that has been placed but not yet shipped.

Preconditions:

- Customer is logged in.
- Customer has an existing order with status "Processing" (not yet "Shipped").

Basic Flow:

- Customer navigates to their "Order History" or "Dashboard."
- Customer selects an active order to view details.
- Customer clicks the "Cancel Order" button.

- System asks for confirmation.
- Customer confirms the cancellation.
- System validates that the order status is still "Processing."
- System updates the order status to "Cancelled."
- System initiates the refund process (if payment was already made).
- System updates the product inventory (increments stock).
- System sends a cancellation confirmation email to the Customer.

Alternative Flows:

If the order status has already changed to "Shipped" while the customer was viewing it, the System displays an error message: "Order cannot be cancelled as it has already been shipped.

Postconditions: Order status is updated to "Cancelled," and inventory is restored.

6.4 Use Case: Product Management

Actor: Administrator

Description: An administrator adds, updates, or removes products from the catalog.

Preconditions: Administrator is logged in with appropriate permissions.

Basic Flow:

- Administrator navigates to product management section.
- Administrator selects to add, edit, or delete a product.
- For add/edit: Administrator enters product information.
- For delete: Administrator confirms deletion.
- System validates the information.
- System updates the product catalog.

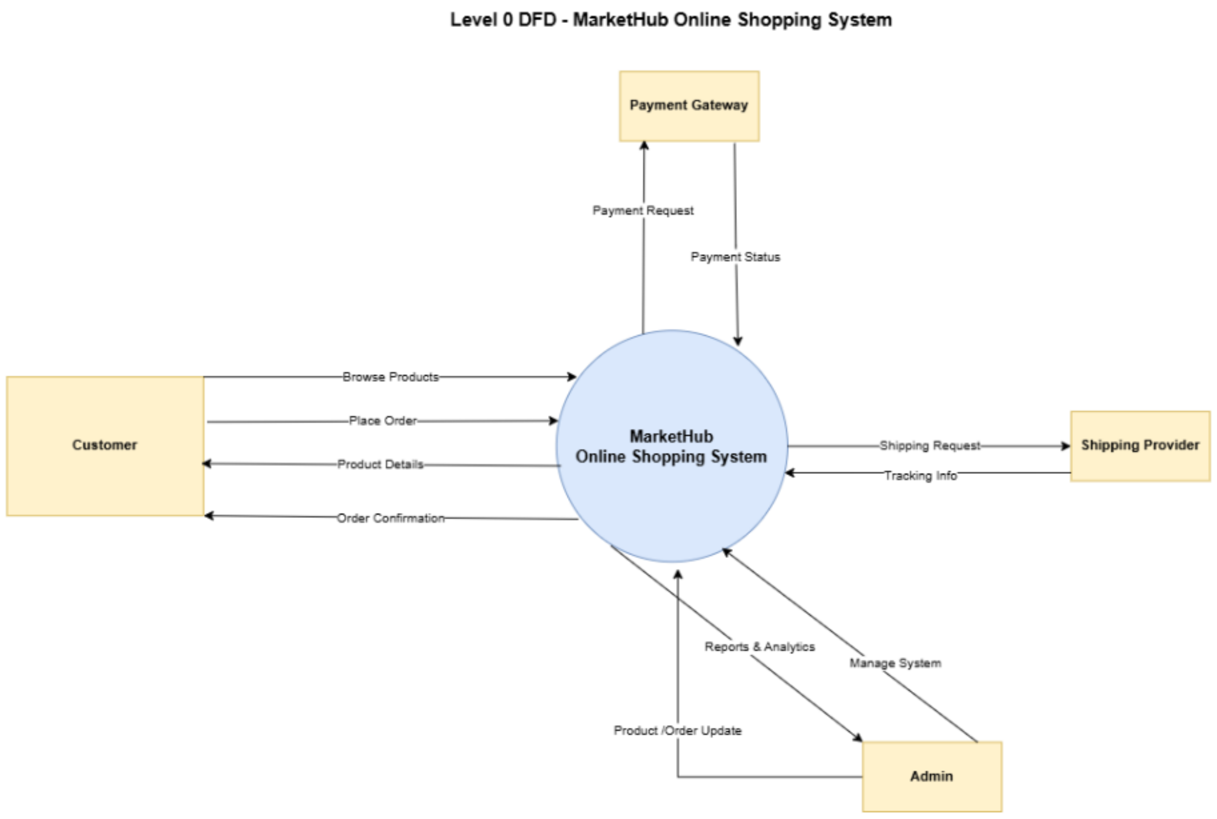
Alternative Flows:

- If required information is missing or invalid, system displays error messages.
- If trying to delete a product with active orders, system warns administrator.

Postconditions: Product catalog is updated accordingly.

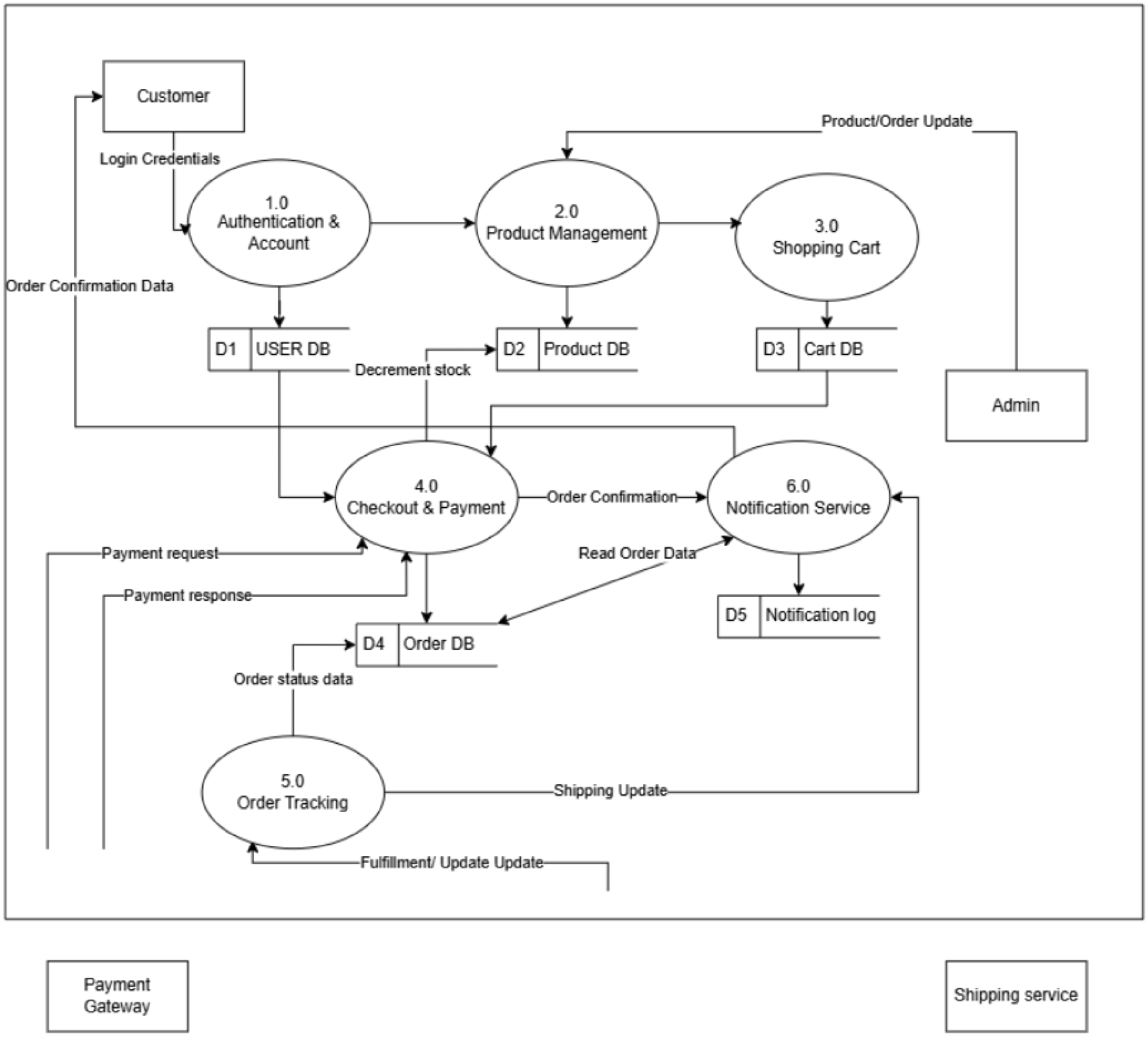
7. DFD

7.1 Level 0 DFD



Raw File:
https://drive.google.com/drive/u/1/folders/1BUffgsk4CdUCnsAXF6pCaaBOSTSZt1Tq

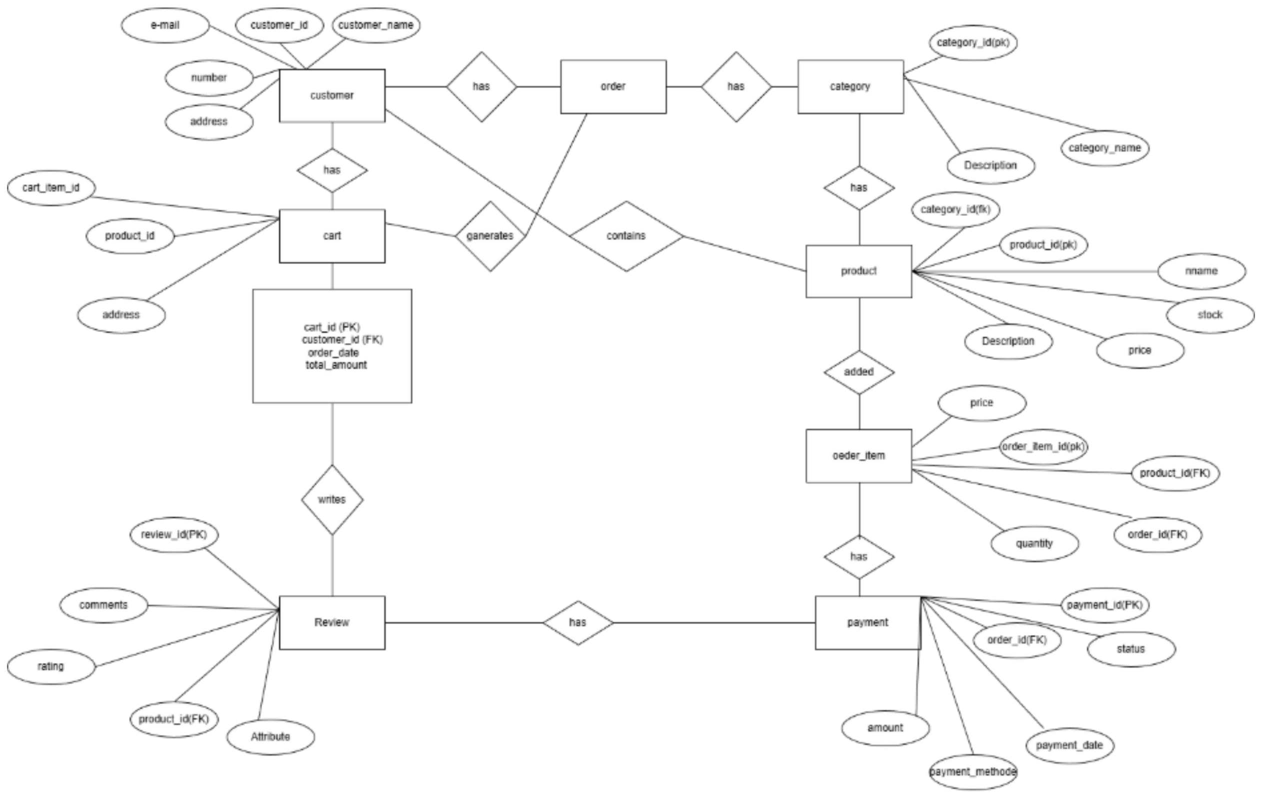
7.2 Level 1DFD



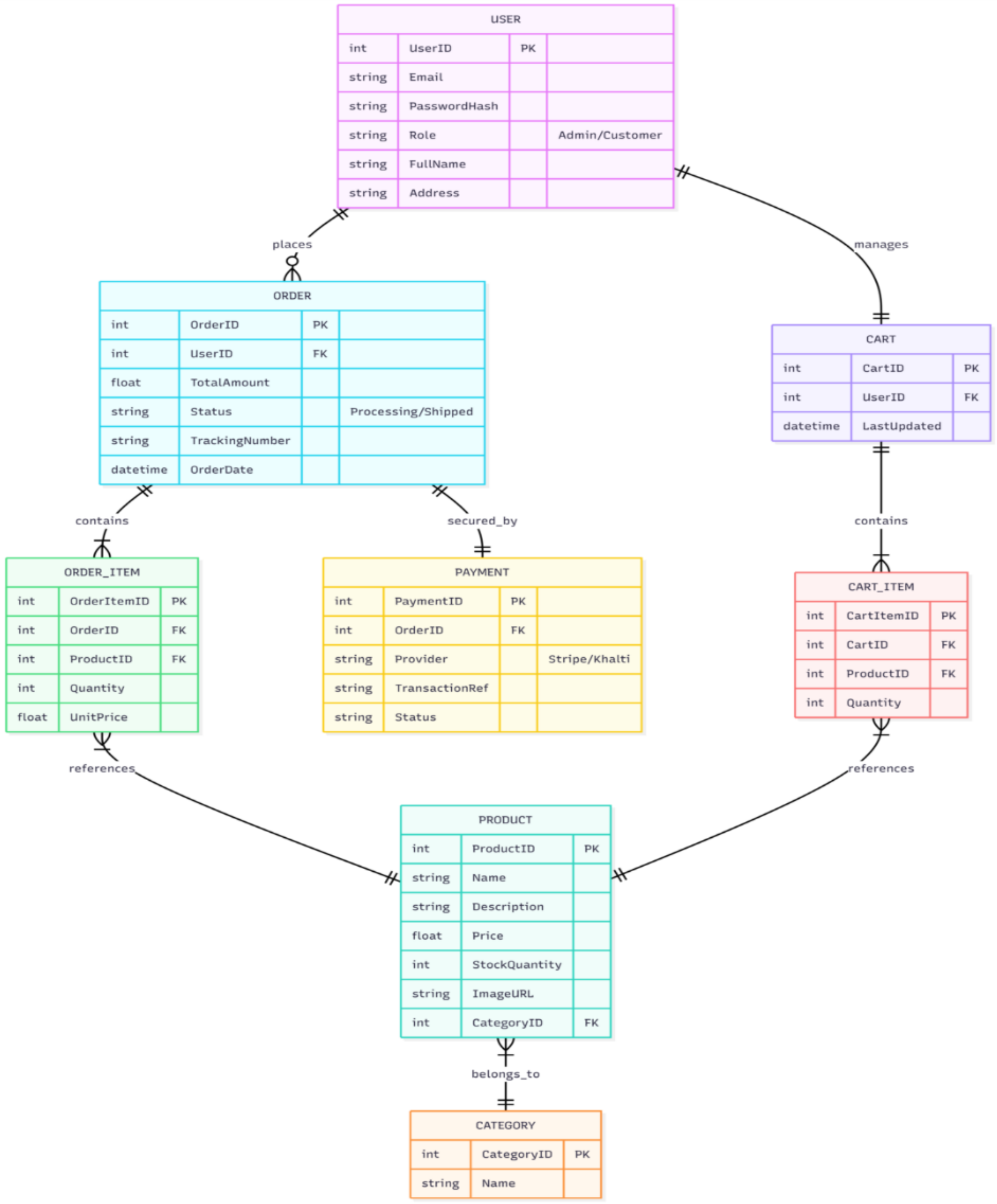
Raw file : https://drive.google.com/drive/u/1/folders/1BUffgsk4CdUCnsAXF6pCaaBOSTSZt1Tq

8. ER Diagram:

8.1 Conceptual ER Diagram (Chen's Notation)


e-mail
customer_id
customer_name
number
address
customer
has
order
has
category
category_id(pk)
Description
category_name
has
cart_item_id
product_id
address
cart
ganerates
contains
cart_id (PK)
customer_id (FK)
order_date
total_amount
category_id(fk)
product_id(pk)
product
nname
stock
Description
price
added
price
oeder_item
order_item_id(pk)
product_id(FK)
order_id(FK)
quantity
writes
has
review_id(PK)
comments
rating
product_id(FK)
Attribute
Review
has
payment
payment_id(PK)
order_id(FK)
status
amount
payment_methode
payment_date

8.2 Physical/Logical ER Diagram (Crow's Foot Notation)



Source: https://www.mermaidchart.com/d/4f197d22-9dda-42e8-8a2d-7cfa5f3044b7