Software Requirements Specification Document

For

< E-Commerce System >

< By: Sanjeet Dhakal >

< Version: 01 > **< Date: 04/12/2024 >**

Table of Contents:

1. *Introduction:*

1.1 Purpose

This Software Requirements Specification (SRS) document provides a detailed description of the E-Commerce system. The document is intended for use by the development team, project managers to ensure all parties have a clear understanding of the system requirements and expectations.

1.2 Scope

The E-Commerce system is a web-based platform that allows users to browse, purchase, and review products online. It also provides an admin panel for managing products, users, and orders. The system will include features such as user authentication, product search, shopping cart, payment gateway integration, and order management.

1.3 Definitions

- **E-Commerce:** Online buying and selling of products or services.
- **Admin Panel:** A back-end interface for administrators to manage the platform.
- **Payment Gateway:** A service that securely processes payments online.
- **User Authentication:** The process of verifying the identity of users.

1.4 References

- Payment Gateway Documentation
- Web Accessibility Guidelines

1.5 Overview

This document describes both functional and non-functional requirements for the E-Commerce system. It will outline the system's features, design, and constraints.

2. *Overall Description:*

2.1 Product Perspective

The system will be a web-based platform, accessible via modern browsers on desktops, tablets, and mobile devices. It integrates third-party services like payment gateways, shipping services, and customer support tools.

2.2 Product Features

- **User Registration:** Users can create an account and log in to the platform.
- **Product Search:** Users can search for products based on various criteria.
- **Shopping Cart:** Users can add products to their cart and proceed to checkout.
- **Order Management:** Admins can manage orders and update their statuses.

2.3 User Classes and Characteristics

- **Customers:** Users who browse and purchase products.
- **Admin:** Users who manage products, users, and orders.
- **Guest Users:** Users who browse products without creating an account.

2.4 Operating Environment

The system will run on standard web servers (Apache/Nginx) and will be compatible with modern web browsers (Chrome, Firefox, Edge, Safari). It will support both desktop and mobile platforms.

2.5 Design and Implementation Constraints

- Compliance with PCI DSS standards for payment processing.
- Must be responsive to work on mobile devices.
- Integration with external payment gateways and shipping providers.

2.6 Assumptions and Dependencies

- The system assumes that users will have internet access.
- The system depends on third-party payment gateway.

3. *System Features:*

3.1 User Registration and Authentication

- **Description:** Users must be able to register an account, log in, and recover their password.
- **Functional Requirements:**
    - Users must provide email, password, and personal details to create an account.
    - Login should support email and password authentication.

3.2 Product Search and Browsing

- **Description:** Users should be able to search for products by category, price range, and other filters.
- **Functional Requirements:**
    - A search bar will allow users to input keywords.
    - Filters will include categories, price range, ratings, etc.

3.3 Shopping Cart and Checkout

- **Description:** Users should be able to add products to their cart and complete the checkout process.
- **Functional Requirements:**
  - Users can add/remove products to/from the cart.
  - Cart displays product details, quantities, and prices.
  - Checkout process should include shipping details, payment options, and confirmation.

3.4 Payment Gateway Integration

- **Description:** The system should integrate with a secure payment gateway to process transactions.
- **Functional Requirements:**
  - The payment gateway should support multiple payment methods (credit cards, PayPal, etc.).
  - Secure data transmission via SSL/TLS encryption.

3.5 Order Management

- **Description:** Admins will be able to manage customer orders and update their status.
- **Functional Requirements:**
  - Admins can view all orders, update status (pending, shipped, delivered, etc.), and manage refunds.

3.6 User Profile Management

- **Description:** Users should be able to view and update their profile information.
- **Functional Requirements:**
  - Users can update their name, email, shipping address, and password.
  - Users can view their order history.

3.7 Admin

- **Description:** Admins can manage products, categories, and customers.
- **Functional Requirements:**
  - Admins can add/edit/remove products.
  - Admins can view user profiles and order history.

4. External Interface Requirements:

4.1 User Interfaces

The web interface will include:

- A homepage with featured products.
- A product catalogue with filtering options.
- A shopping cart page.
- A checkout and payment page.
- User and admin dashboards.

4.2 Hardware Interfaces

- The system should be compatible with any device with internet access (desktop, tablet, mobile).

4.3 Software Interfaces

- Integration with third-party payment gateway APIs.
- Integration with shipping service APIs.

5. System Attributes:

5.1 Performance Requirements

- The system should be able to handle up to 10,000 concurrent users.
- The system should provide a response time of less than 2 seconds for most actions.

5.2 Security Requirements

- Data encryption for sensitive information (passwords, payment data).
- Two-factor authentication for admin users.

5.3 Usability Requirements

- The system should have an user-friendly interface.
- The system should be accessible to users with disabilities.

5.4 Reliability Requirements

- The system should have an uptime of 99.9% per month.
- Regular backups of critical data should be performed.

5.5 Maintainability and Supportability

- The system should have logging and monitoring to identify errors.
- The system should allow for easy updates and feature additions.

6. Other Non-Functional Requirements:

6.1 Scalability

- The system should be able to scale to handle increased traffic during peak seasons (e.g., Black Friday, Christmas).

6.2 Localization and Internationalization

- The system should support multiple languages.

6.3 Data Integrity and Privacy

- User data must be protected and comply with relevant data privacy laws.

7. Appendices

- **1. Use Case Diagrams**
- **2. Data Flow Diagrams**
- **3. Entity-Relationship Diagrams**

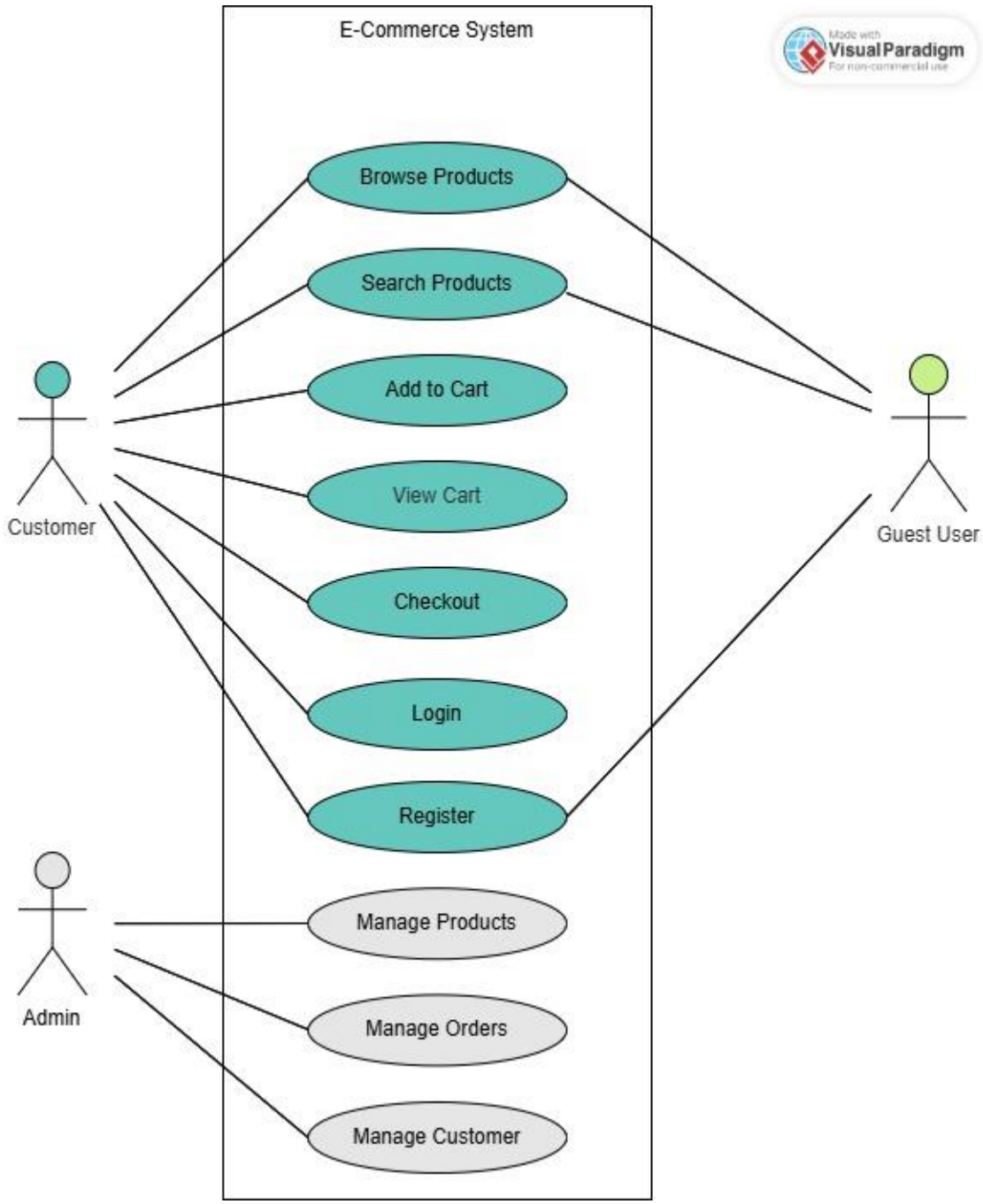
1. Use Case Diagram

A use case diagram for an E-Commerce system represents the various actors or users and their interactions with the system's functionalities.

Key Components:

a. Actors:

- **Customer:** The person who buys products.
- **Admin:** The person managing the products, orders, and customers.
- **Guest:** A non-logged-in user browsing the website.



b. Use Cases (system functions):

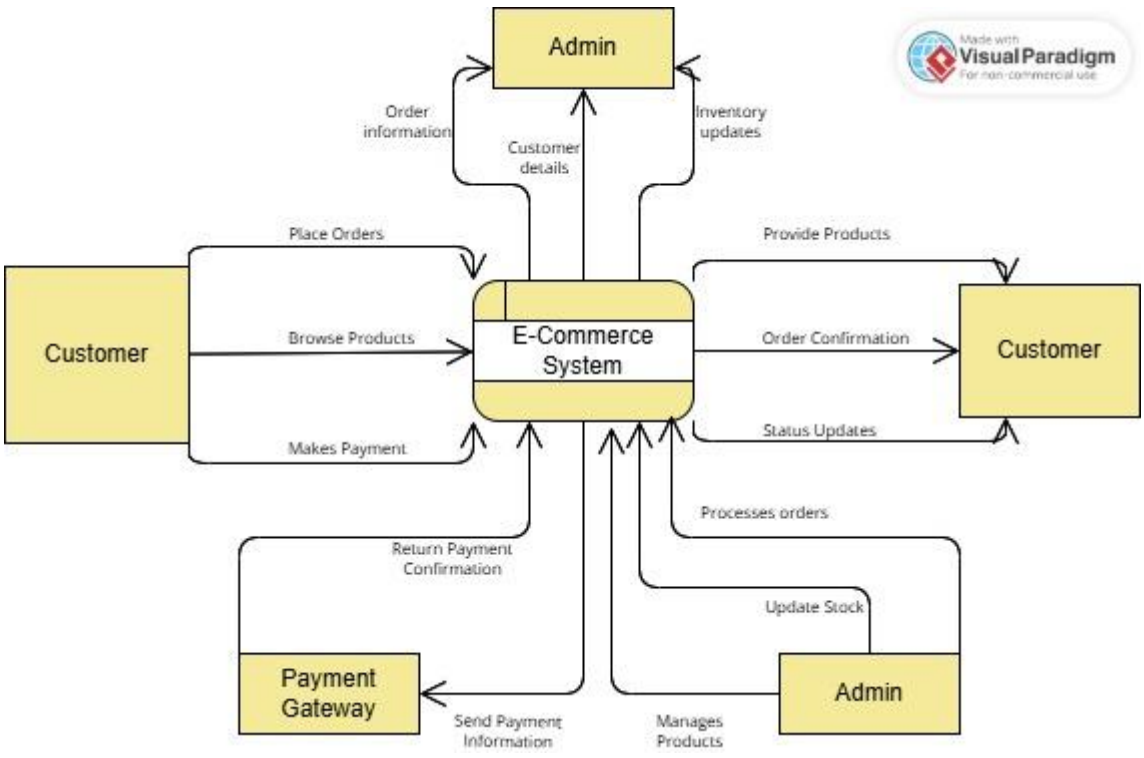
- **Browse Products:** Customers or guests can browse the available products.
- **Add to Cart:** Customers can add products to their shopping cart.
- **View Cart:** Customers can view the products they've added to their cart.
- **Checkout:** Customers proceed with the checkout process.
- **Login:** Customers can log into their accounts.
- **Register:** Guest User can register and create their account.

- **Manage Products:** Admin can add, update, or remove products from the catalogue.
- **Manage Orders:** Admin manages customer orders (view, update, or cancel).
- **Manage Customers:** Admin manages customer accounts.

2. ***Data Flow Diagram***

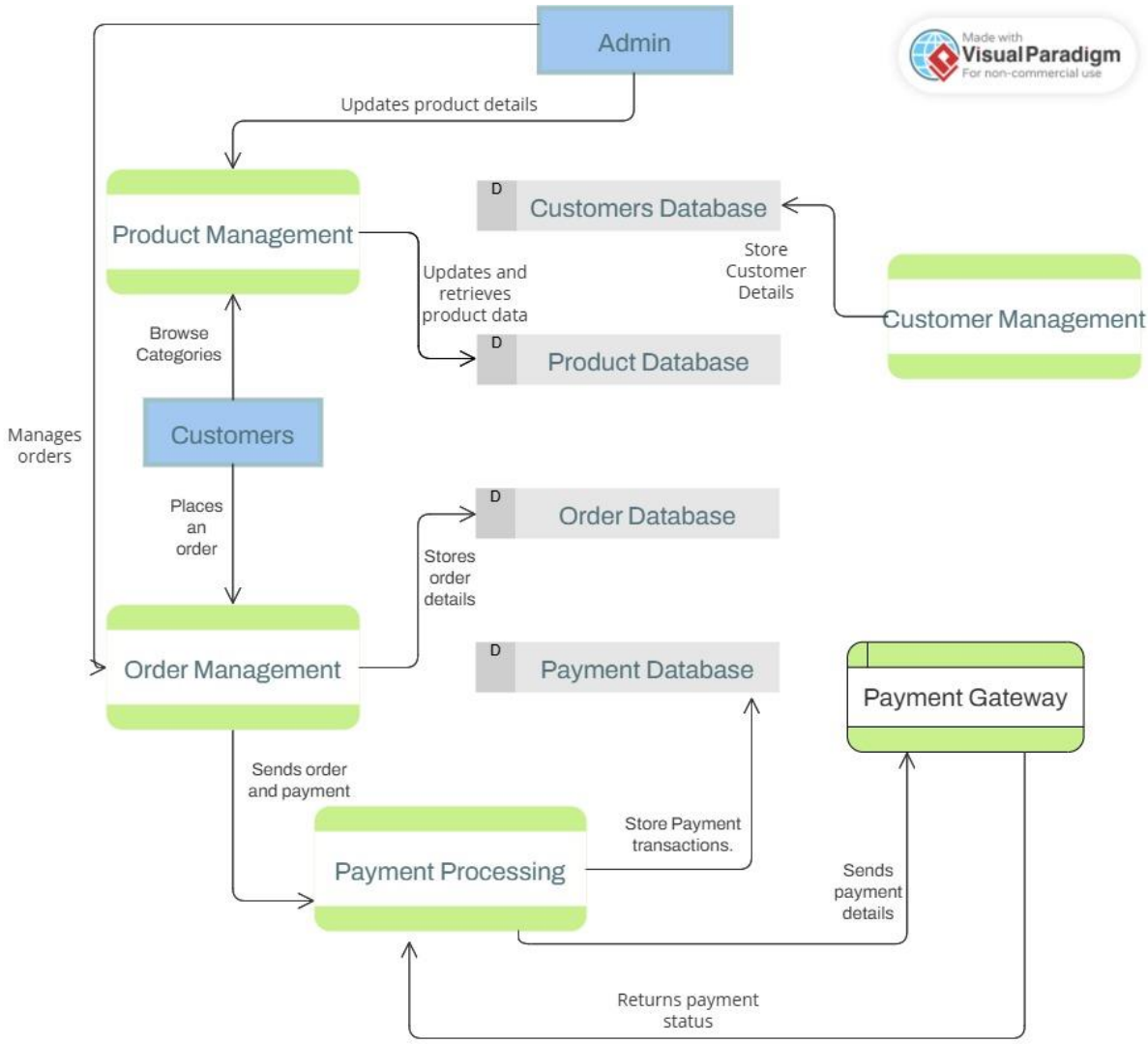
A Data Flow Diagram (DFD) for an e-commerce system typically depicts the flow of data between the different entities and processes involved in the system.

- **Level 0 (Context Diagram)**
  This is the Context Diagram (Level 0), which gives a high-level overview of the e-commerce system.



- **Level 1 DFD:**

  Level 1 expands on the single process in the context diagram by showing the primary functions of the e-commerce platform.



3.***Entity-Relationships Diagram***

Entities and Attributes for the E-commerce Website

1. Product:

- **P-ID (Primary Key):** Unique identifier for each product.
- **Name:** Name of the product.
- **Price:** Price of the product.
- **Description:** Description of the product.

2. <u>Category:</u>

- **Category – ID (Primary Key):** Unique identifier for each category.
- **Name:** Name of the category.

3. <u>Order:</u>

- **Order – No (Primary Key):** Unique identifier for each order.
- **Order - Amount:**
- **Order - Date:**

4. <u>User:</u>

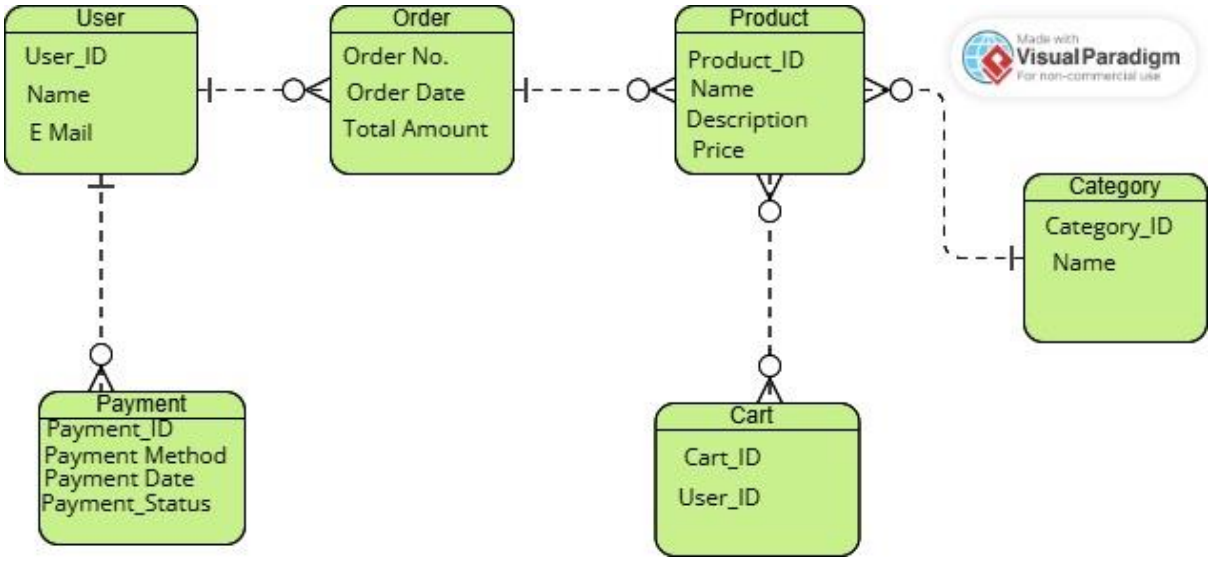
- **User – ID (Primary Key):** Unique identifier for each user or customer.
- **Name:** Name of the user.
- **Email:** Email of the user.

5. <u>Payment:</u>

- **Payment – ID (Primary Key):** Unique identifier for each payment.
- **Method:** Payment method like UPI or Credit Card etc.
- **Amount:** Total amount paid by user.

6. <u>Cart:</u>

- **Cart – ID (Primary Key):** Unique identifier for each cart.
- **User - ID (Foreign Key):** Reference to the user.



<u>Relationships Between These Entities:</u>

1. <u>Product - Category Relationship</u>:

- One product can belong to only one category.
- One category can have multiple products.
- So this is a **Many-to-one** relationship, showing that many products can belong to a single category.

2. <u>User-Order Relationship</u>:

- One user can place multiple orders.

- Each order is placed by exactly one user.
- This is a **one-to-many** relationship, showing that a user can place multiple orders, but each order is placed by exactly one user.

3.Product - Cart Relationship:

- One product can be added to multiple carts.
- Each cart can contain multiple products.
- This is **many-to-many** relationship.

4. User-Payment Relationship:

- One user can make multiple payments.
- Each payment is made by exactly one user.
- This is **one-to-many** relationship because each user can make multiple payments, and each payment is made by a single user.

5. Order-Product Relationship:

- One order can contains multiple products.
- Many products are get ordered in each order.
- So this is **one-to-many** relationship we can order multiple products on each order.