System Analysis & Design for an E-Commerce Web Application

1. Introduction

1.1 Purpose

This document provides a comprehensive analysis and design outline for an e-commerce web application with automated DevOps deployment. It covers requirements, architecture, data design, and the deployment pipeline.

1.2 Scope

1. Project Overview
2. . Problem Statement & Objectives
3. Use Case Diagram & Descriptions
4. Functional & Non-Functional Requirements
5. Software Architecture
6. Data Flow & System Behavior
7. UI/UX Design & Prototyping
8. System Deployment & Integration
9. Additional Deliverables
10. Conclusion

1.3 Audience

The primary audience includes:

- **Developers:** who need to understand the system's architecture and codebase.
- **DevOps Engineers:** responsible for creating and maintaining the automated deployment pipeline.
- **Project Managers:** overseeing timelines, deliverables, and resource allocation.
- **QA/Test Engineers:** validating functional and non-functional requirements.

Stakeholder	Role / Interest	Responsibilities	Impact on Project
Project Manager	Oversees project execution, timelines, and resource allocation	Coordinates between teams, manages risks, and ensures deliverables meet deadlines	**High**
Developers (Frontend & Backend)	Implement application features and business logic	Develop and integrate frontend (React-Vite-Tailwind) and backend (Node.js) components	**High**
DevOps Engineers	Automate deployment and manage infrastructure	Build CI/CD pipelines, configure Docker, Ansible, and Kubernetes clusters	**High**
UI/UX Designers	Design user interface and ensure usability	Create wireframes, prototypes, and maintain consistent design standards	**Medium**
Database Administrator	Maintain data integrity and performance	Manage MongoDB collections, indexing, backups, and security	**Medium**
Quality Assurance (QA) Team	Ensure quality and reliability	Test features, report bugs, and validate fixes across environments	**Medium**
End Users (Customers)	Use the e-commerce platform for purchases	Provide feedback, browse products, and place orders	**High**
Stakeholders / Investors	Provide funding and evaluate performance	Review progress, ensure ROI, and approve major milestones	**Medium**

2. Problem Statement & Objectives

2.1 Problem Statement

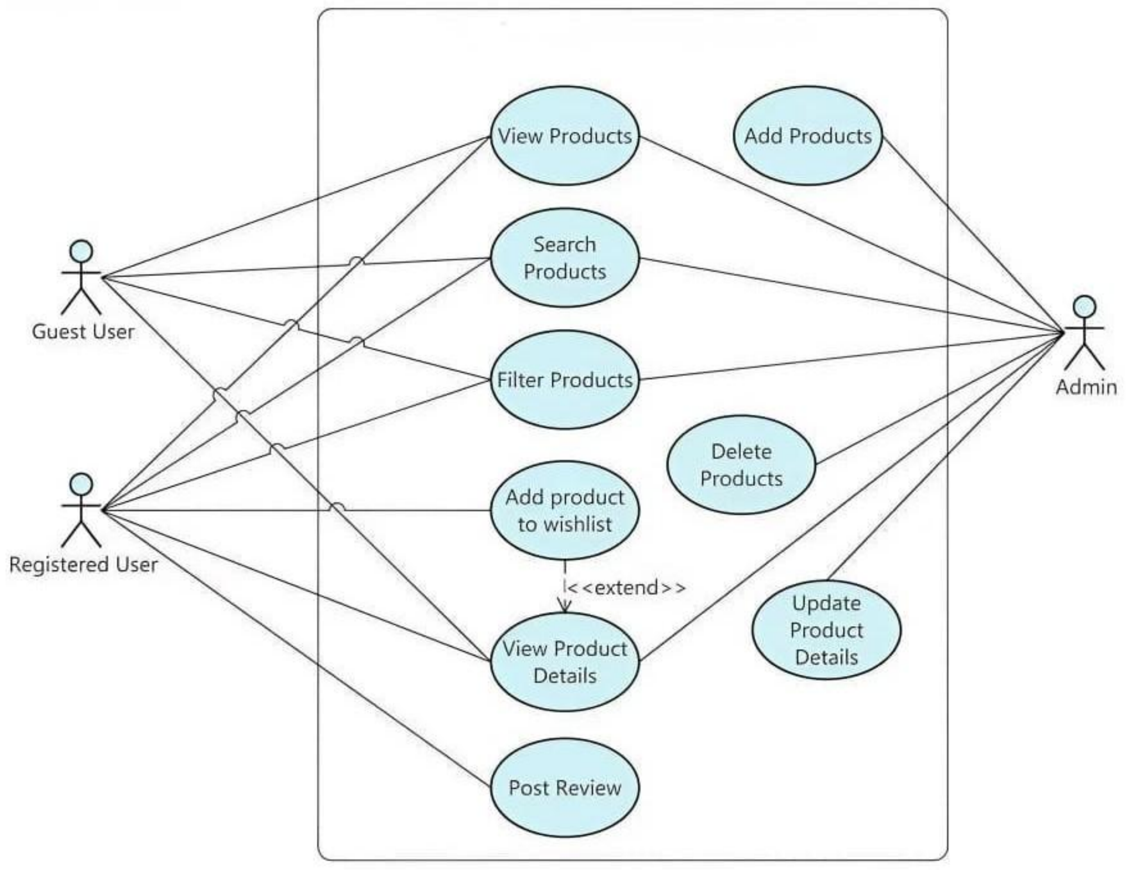
Manual deployment processes are time-consuming and error-prone, often leading to inconsistent environments and deployment failures.

2.2 Objectives

- Automate the deployment of an e-commerce web application using modern DevOps tools.
- Reduce errors and downtime by establishing a robust CI/CD pipeline.
- Improve scalability and performance by containerizing services and using orchestration platforms.

3. Use Case Diagram & Descriptions

2.1 Use Case Diagram



Actors & Their Roles

- **Guest User can:**
    - **View Products**
    - **Search Products**
    - **Filter Products**
    - **View Product Details**
- **Registered User can perform all guest actions plus:**
    - **Add Product to Wishlist**
    - **Add Product to Cart**
    - **Place Orders**
    - **Track Orders**
    - **Make Payments**
    - **Return Products**
    - **Post Reviews**
- **Admin can perform all registered user actions plus:**
    - **Add Products**
    - **Update Product Details**
    - **Delete Products**
    - **Manage Orders**
    - **Oversee Payment Transactions**

---

4. Functional & Non-Functional Requirements

4.1 Functional Requirements

- User Authentication and Authorization
- Product Listing, Searching, and Filtering
- Shopping Cart and Wishlist
- Secure Checkout and Payment Processing
- Order Tracking and Management
- Product Reviews and Ratings
- Admin Panel for Product & Order Management

4.2 Non-Functional Requirements

- Performance: Support up to [X] concurrent users.
- Security: Secure data transmission and storage.
- Scalability: Efficiently scale with user demand.
- Reliability: Implement failover and rollback mechanisms.

---

5. Software Architecture

4.1 High-Level Design

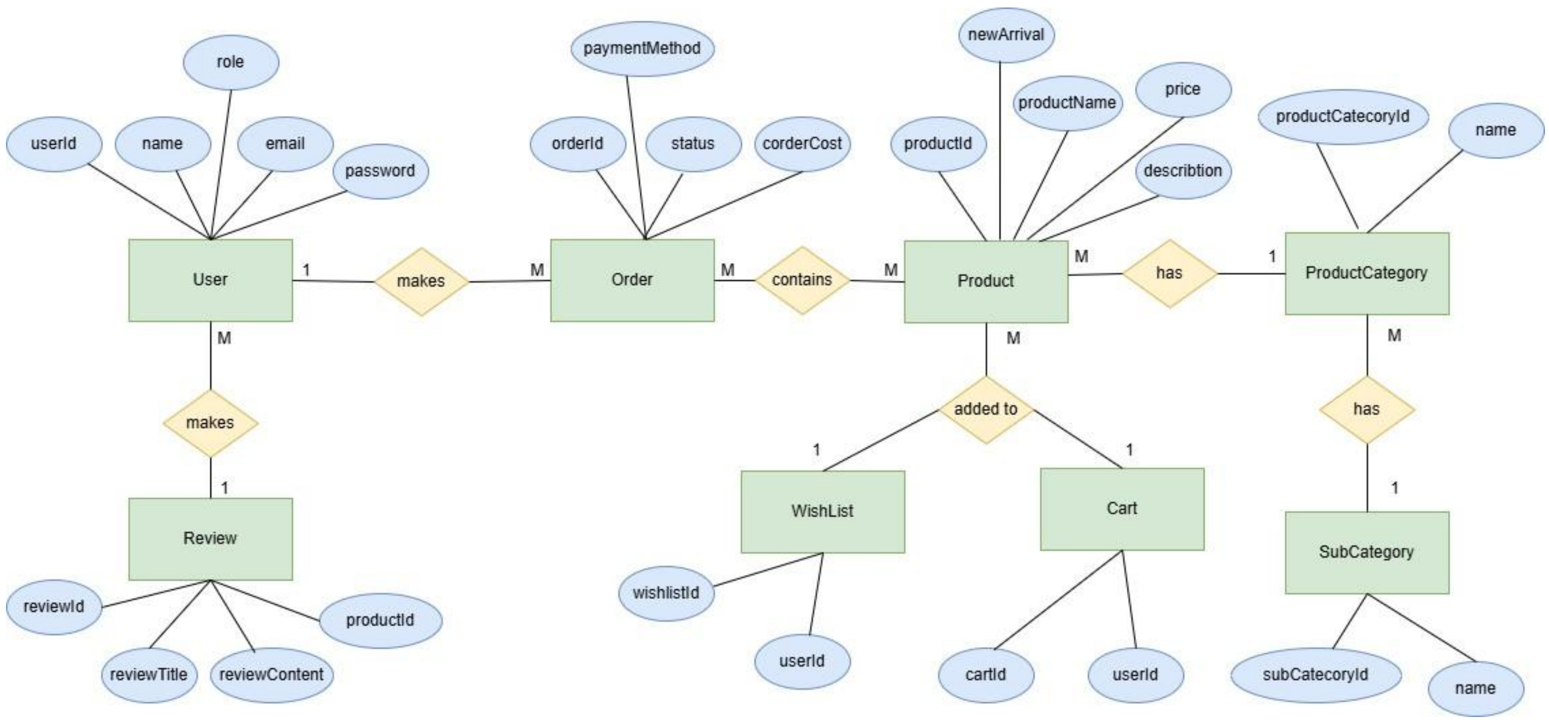
- **Frontend:** React.js /vite
- **Backend:** Node.js
- **Database:** MongoDB
- **DevOps Tools:** Jenkins, Docker, Kubernetes, Ansible

4.2 Architecture Style

- **Monolithic Architecture:** The entire application is deployed as a single containerized system for simpler management and deployment.

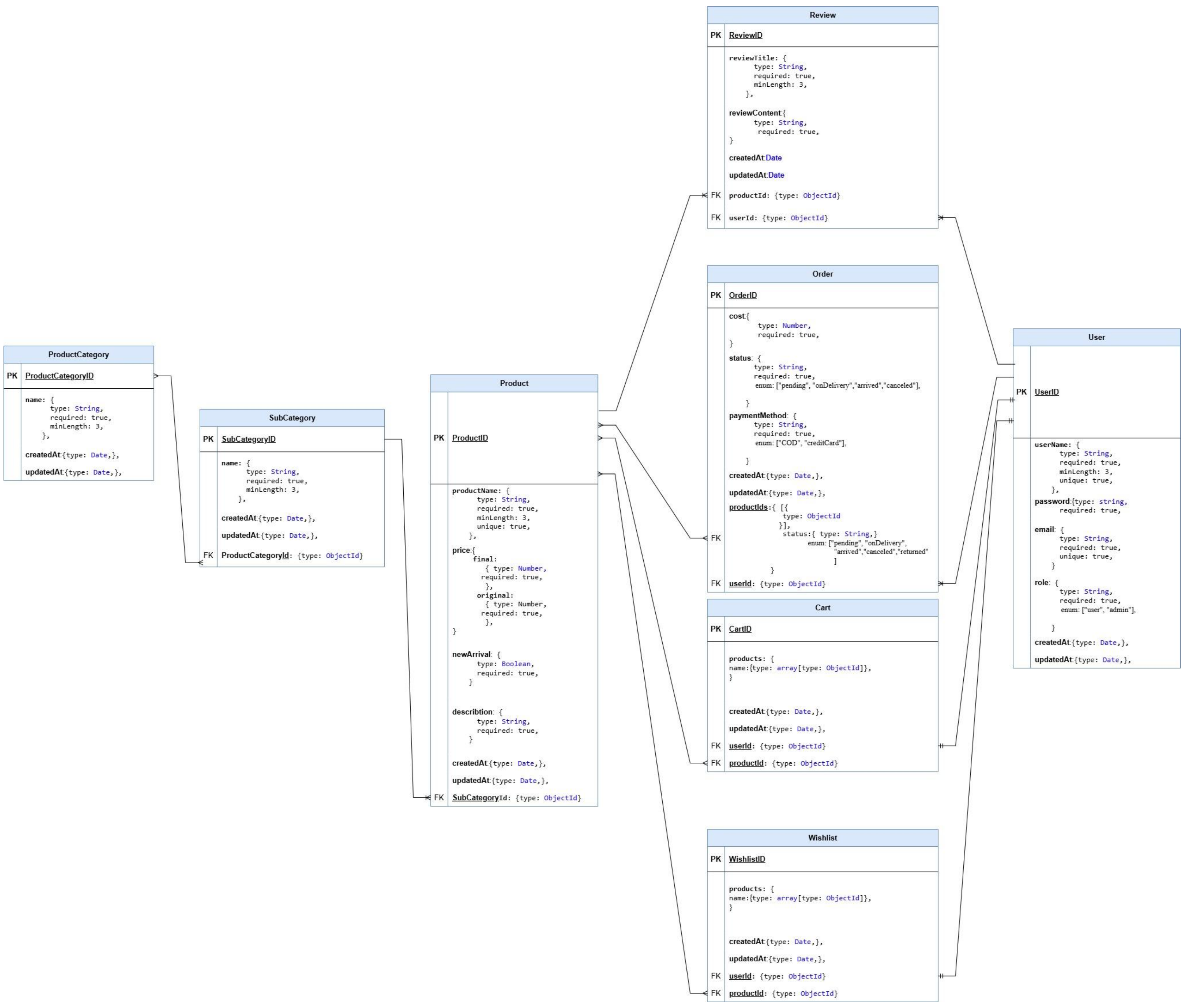
5. Database Design & Data Modeling

5.1 ER Diagram



5.2 Logical & Physical Schema

- Tables, attributes, keys, and normalization considerations.



```
ProductCategory
PK  ProductCategoryID
    name: {
          type: String,
          required: true,
          minLength: 3,
        },
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
```

```
SubCategory
PK  SubCategoryID
    name: {
          type: String,
          required: true,
          minLength: 3,
        },
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
FK  ProductCategoryId: {type: ObjectId}
```

```
Product
PK  ProductID
    productName: {
          type: String,
          required: true,
          minLength: 3,
          unique: true,
        },
    price:{
         final:
            { type: Number,
           required: true,
            },
          original:
            { type: Number,
           required: true,
            },
    }
    newArrival: {
          type: Boolean,
          required: true,
        }
    describtion: {
          type: String,
          required: true,
        }
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
FK  SubCategoryId: {type: ObjectId}
```

```
Review
PK  ReviewID
    reviewTitle: {
          type: String,
          required: true,
          minLength: 3,
        },
    reviewContent:{
          type: String,
           required: true,
    }
    createdAt:Date
    updatedAt:Date
FK  productId: {type: ObjectId}
FK  userId: {type: ObjectId}
```

```
Order
PK  OrderID
    cost:{
           type: Number,
           required: true,
    }
    status: {
          type: String,
          required: true,
          enum: ["pending", "onDelivery","arrived","canceled"],

        }
    paymentMethod: {
          type: String,
          required: true,
          enum: ["COD", "creditCard"],

        }
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
    productIds:{ [{
                 type: ObjectId
                }],
FK               status:{ type: String,}
                       enum: ["pending", "onDelivery",
                             "arrived","canceled","returned"
                             ]
              }
FK  userId: {type: ObjectId}
```

```
Cart
PK  CartID
    products: {
    name:{type: array[type: ObjectId]},
    }
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
FK  userId: {type: ObjectId}
FK  productId: {type: ObjectId}
```

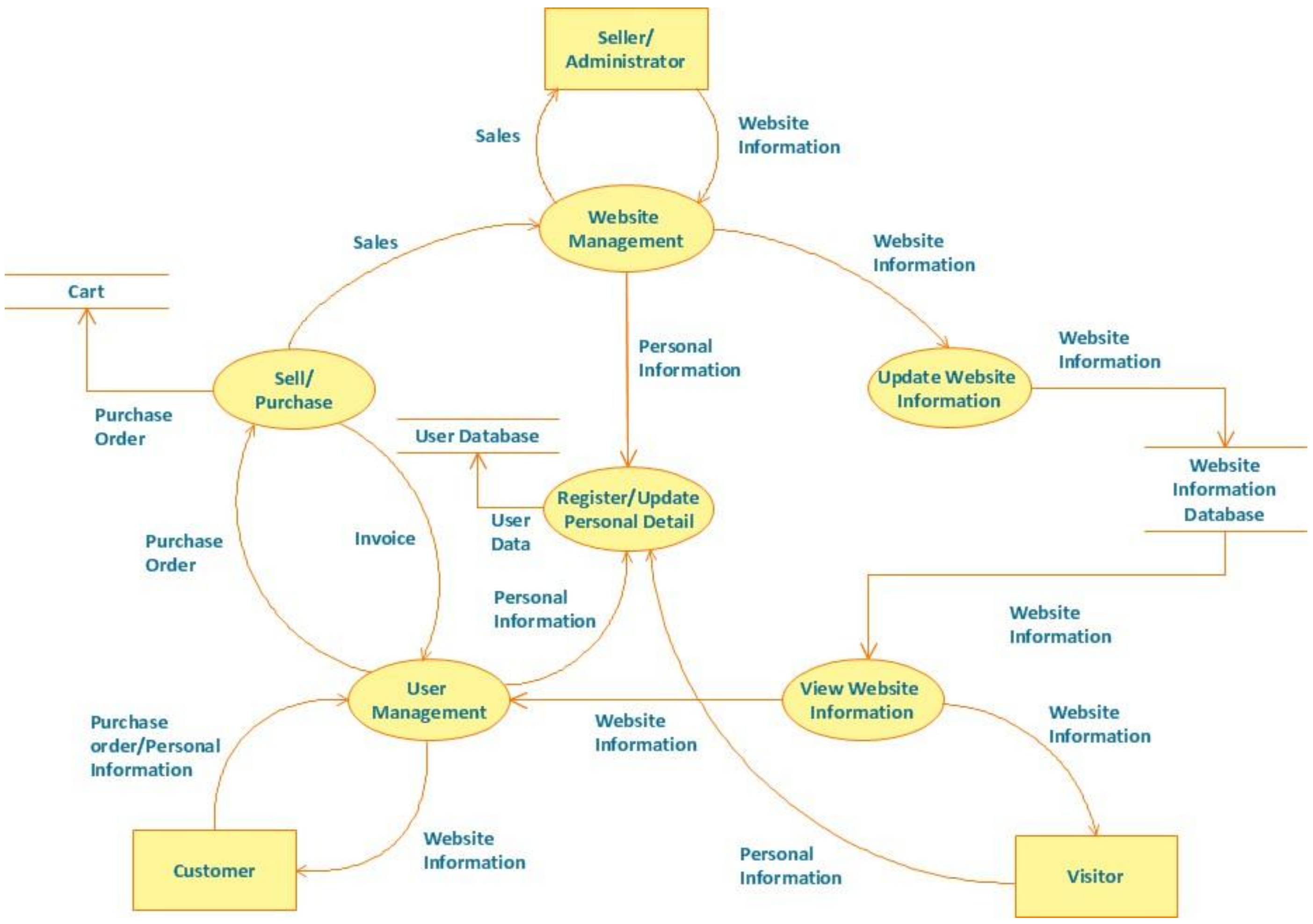
```
Wishlist
PK  WishlistID
    products: {
    name:{type: array[type: ObjectId]},
    }
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
FK  userId: {type: ObjectId}
FK  productId: {type: ObjectId}
```

```
User
PK  UserID
    userName: {
          type: String,
          required: true,
          minLength: 3,
          unique: true,
        },
    password:{type: string,
          required: true,

    email: {
          type: String,
          required: true,
          unique: true,
        }
    role: {
          type: String,
          required: true,
          enum: ["user", "admin"],

        }
    createdAt:{type: Date,},
    updatedAt:{type: Date,},
```

6. Data Flow & System Behavior

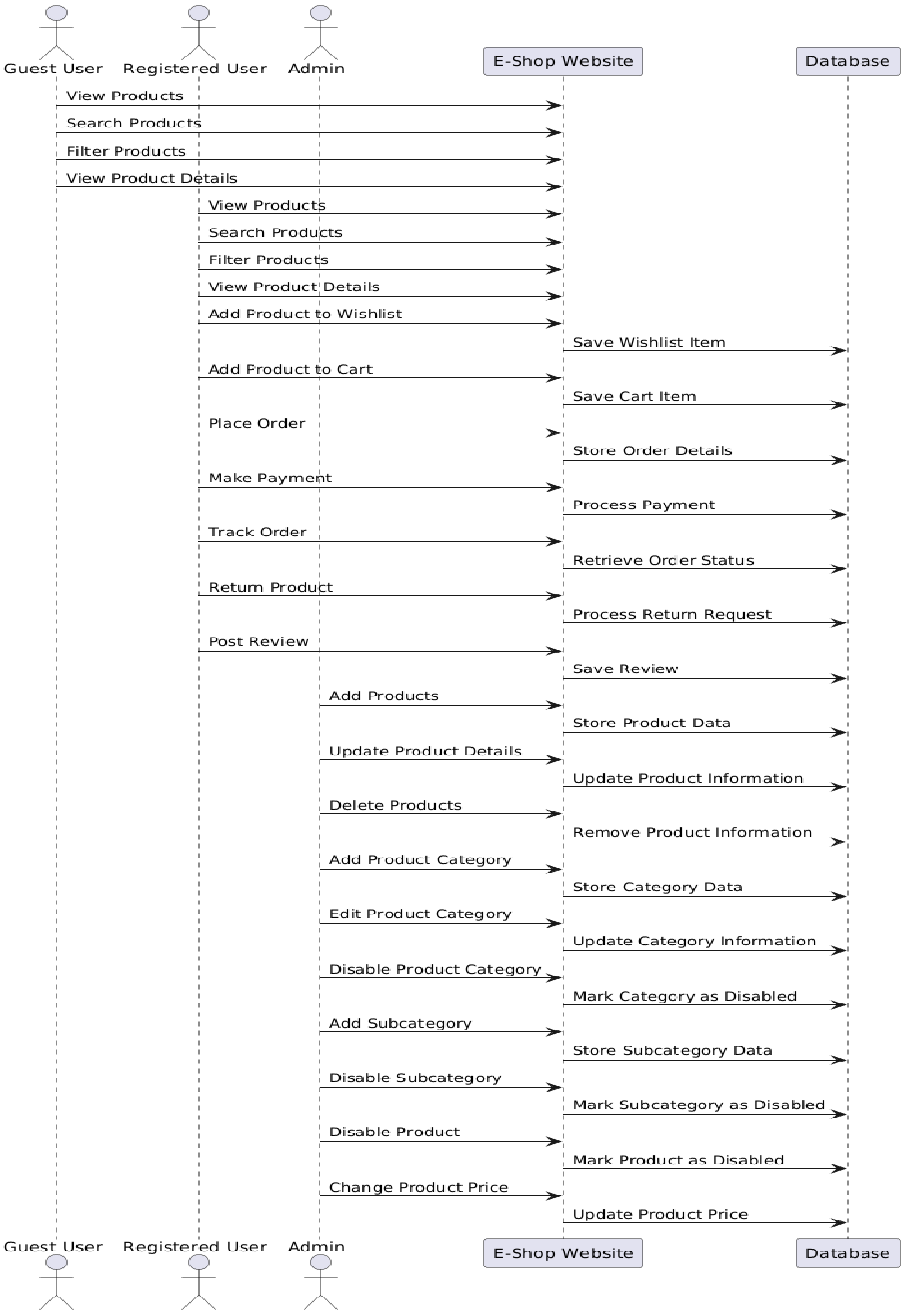
6.1 Data Flow Diagram (DFD)

- Context-level and detailed-level diagrams showcasing data movement.



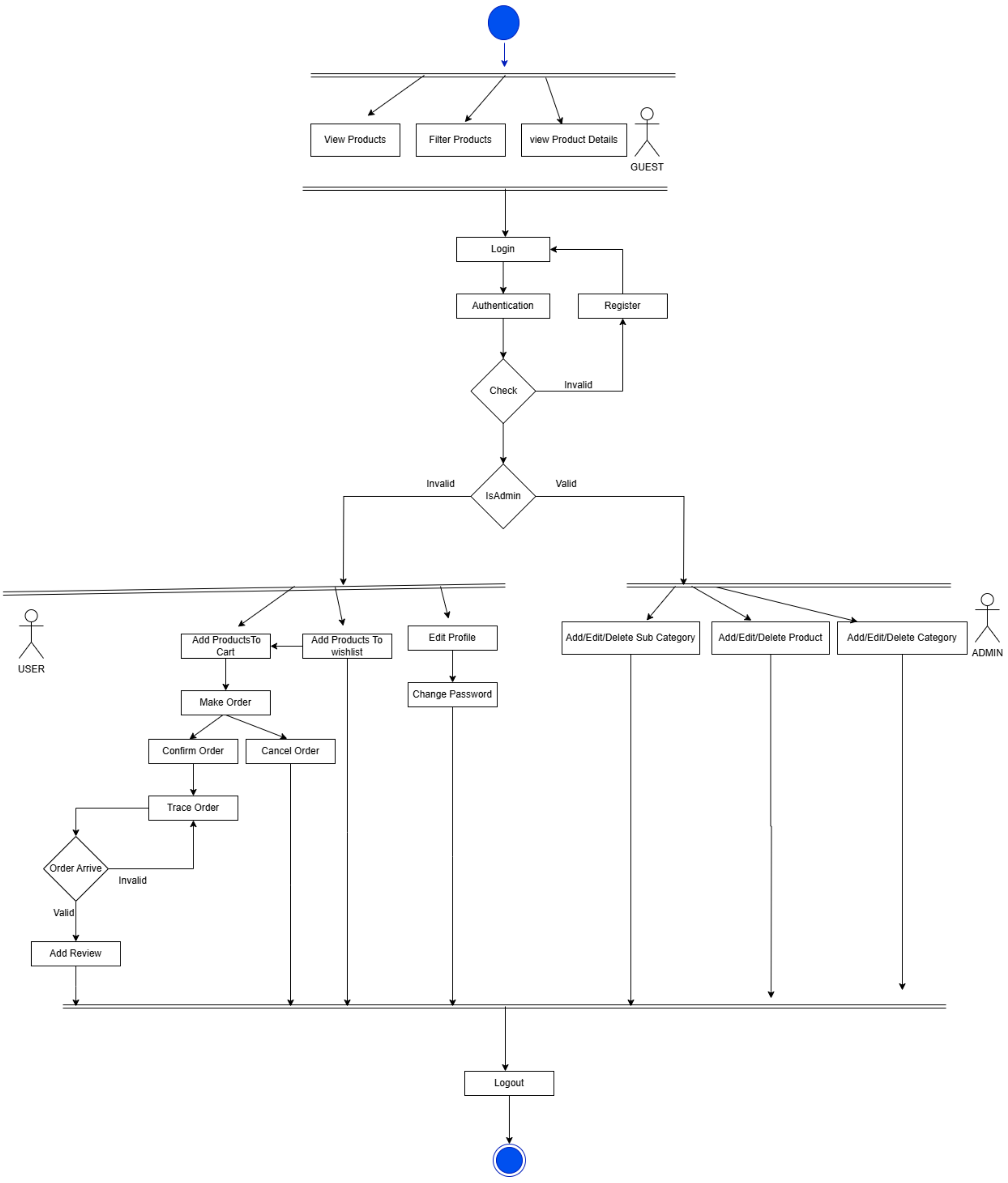
6.2 Sequence Diagrams

- Process flow representation of key interactions.


Guest User
Registered User
Admin
E-Shop Website
Database
View Products
Search Products
Filter Products
View Product Details
View Products
Search Products
Filter Products
View Product Details
Add Product to Wishlist
Save Wishlist Item
Add Product to Cart
Save Cart Item
Place Order
Store Order Details
Make Payment
Process Payment
Track Order
Retrieve Order Status
Return Product
Process Return Request
Post Review
Save Review
Add Products
Store Product Data
Update Product Details
Update Product Information
Delete Products
Remove Product Information
Add Product Category
Store Category Data
Edit Product Category
Update Category Information
Disable Product Category
Mark Category as Disabled
Add Subcategory
Store Subcategory Data
Disable Subcategory
Mark Subcategory as Disabled
Disable Product
Mark Product as Disabled
Change Product Price
Update Product Price
Guest User
Registered User
Admin
E-Shop Website
Database

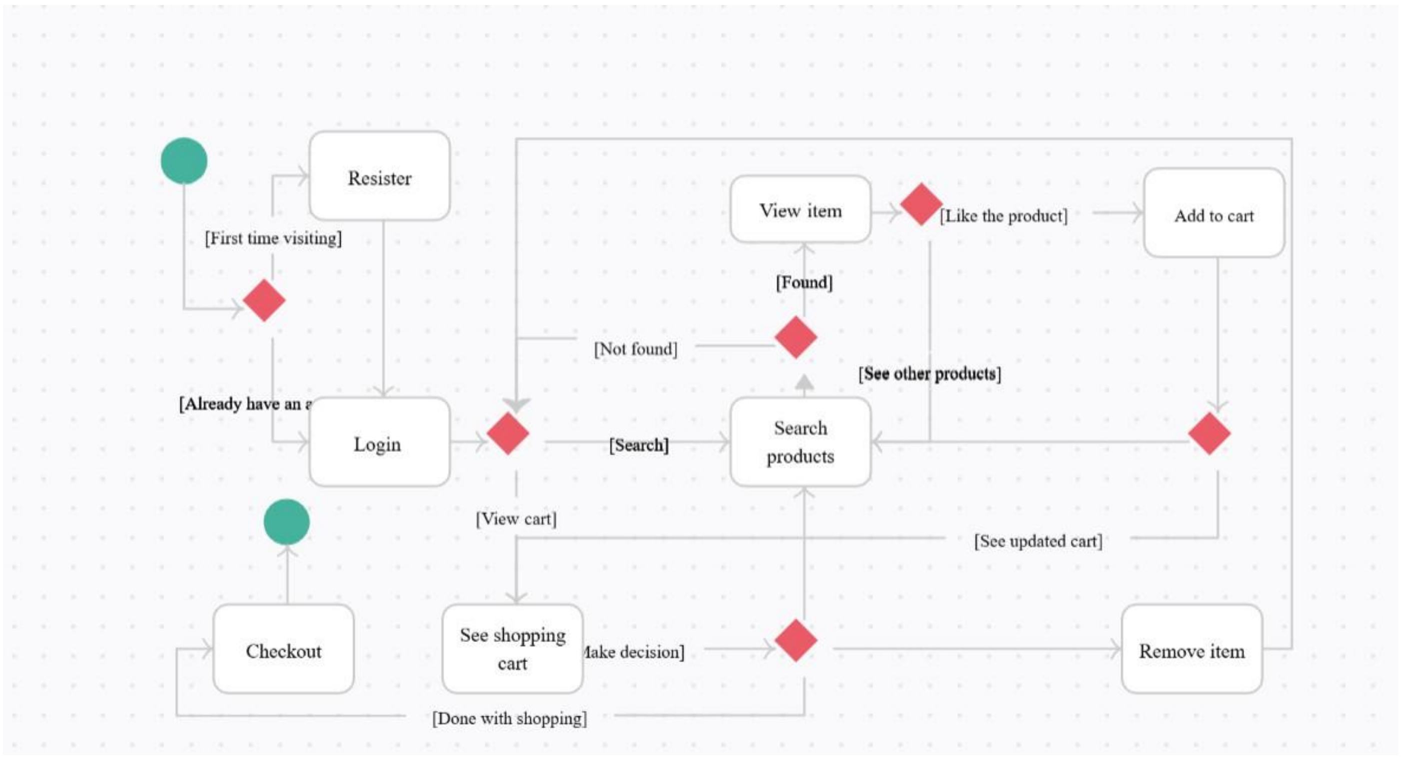
6.3 Activity Diagram

- Workflow visualization of system processes.



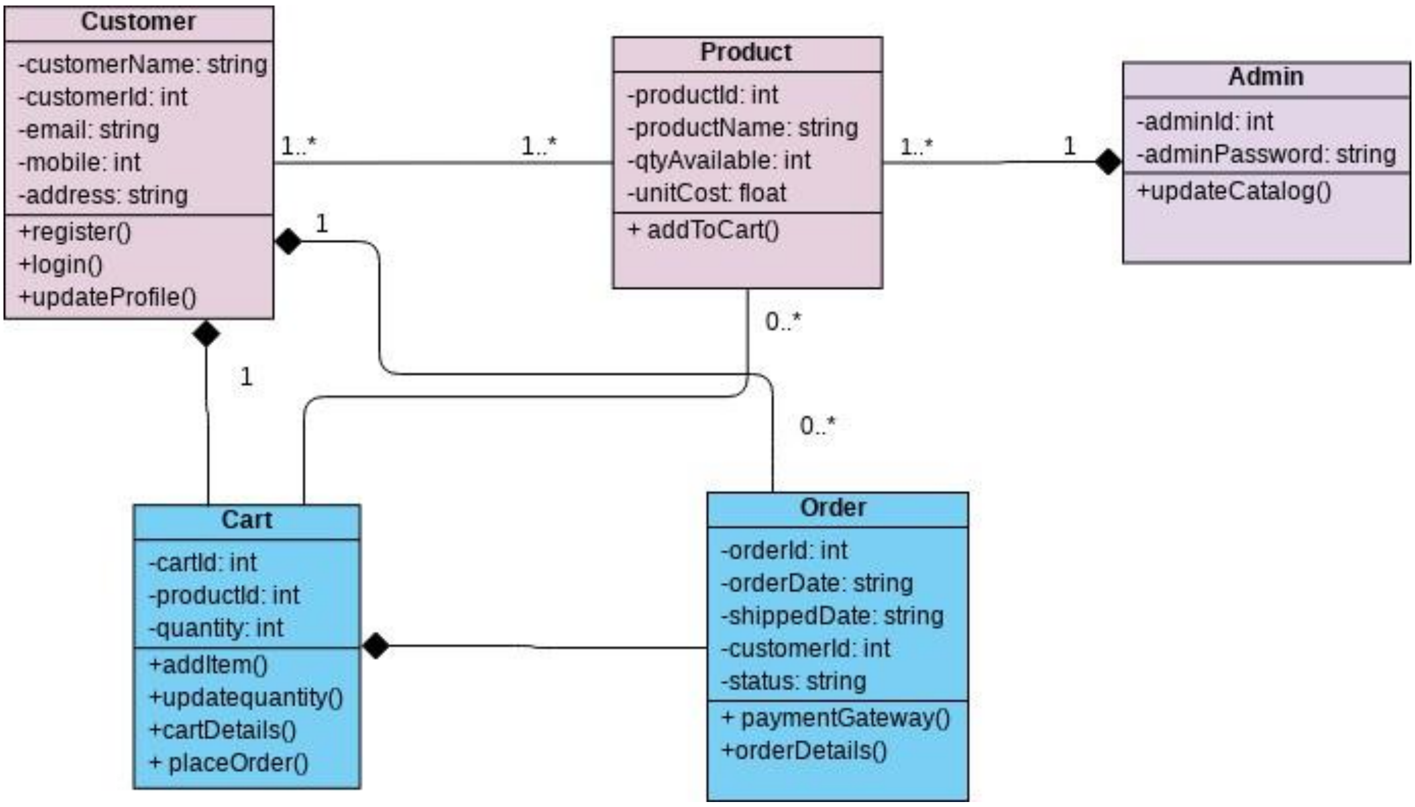
6.4 State Diagram

- Represents different states of system objects.



6.5 Class Diagram

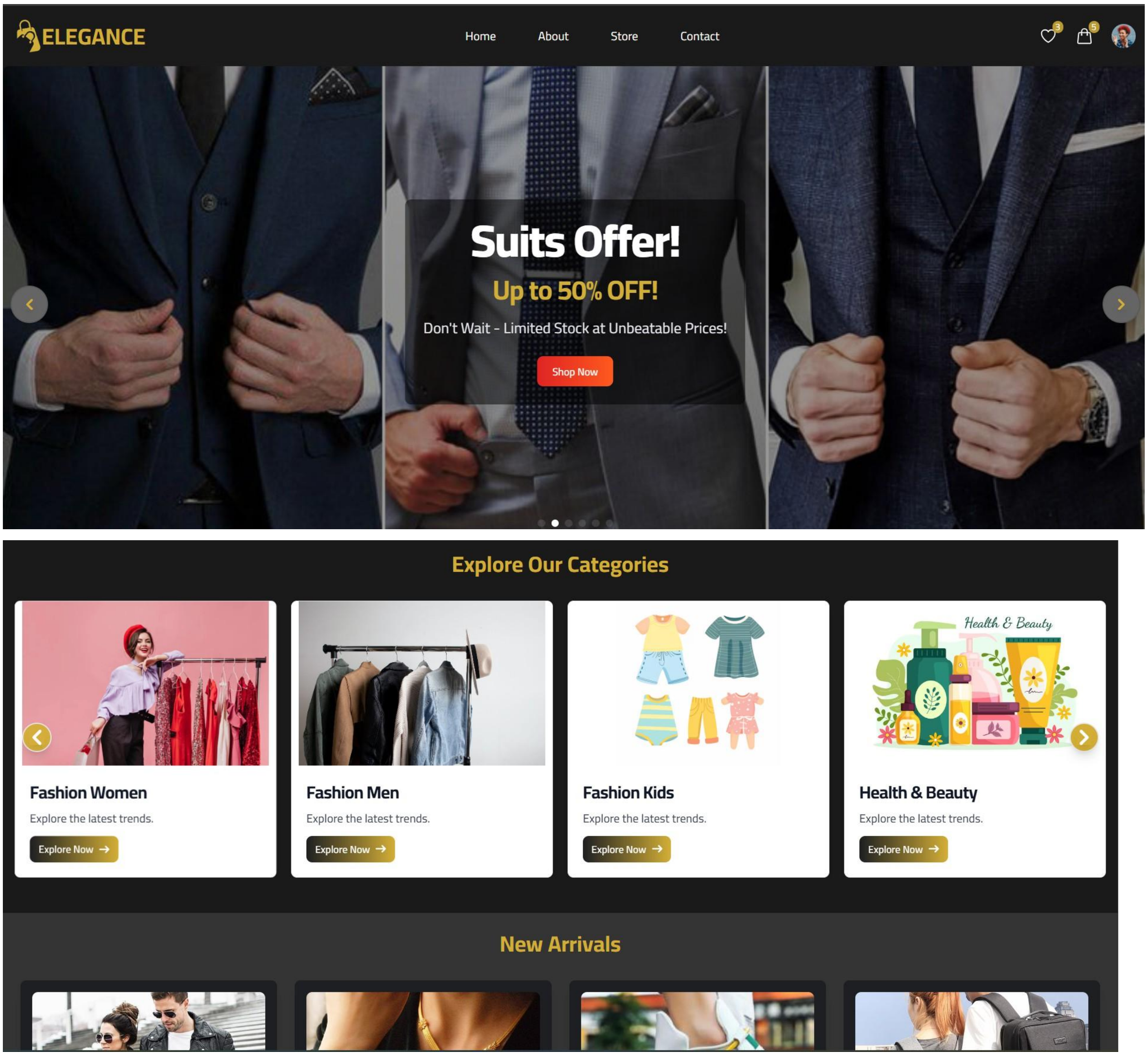
- Defines system structure through classes, attributes, methods, and relationships.

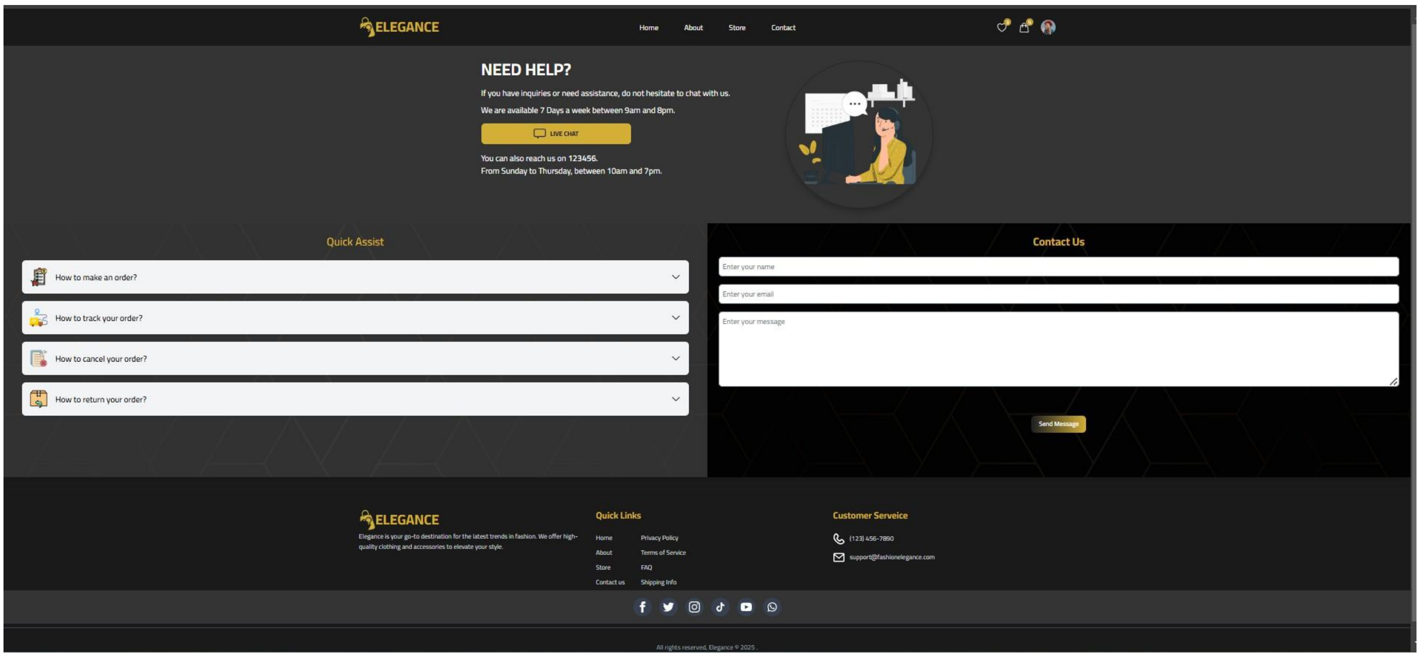


7. UI/UX Design & Prototyping

7.1 Wireframes & Mockups

- Homepage & store page: Displays featured products, search bar, categories.
- Product Detail Page: Shows product images, descriptions, reviews, and “Add to Cart” button.
- Cart & Checkout: Summarizes items, totals, and captures shipping/payment info.
- Order Confirmation: Displays receipt, estimated delivery, and order tracking.




ELEGANCE

Home About Store Contact

NEED HELP?

If you have inquiries or need assistance, do not hesitate to chat with us.

We are available 7 Days a week between 9am and 8pm.

LIVE CHAT

You can also reach us on 123456.
From Sunday to Thursday, between 10am and 7pm.

Quick Assist

How to make an order?

How to track your order?

How to cancel your order?

How to return your order?

Contact Us

Enter your name

Enter your email

Enter your message

Send Message

ELEGANCE

Elegance is your go-to destination for the latest trends in fashion. We offer high-quality clothing and accessories to elevate your style.

Quick Links

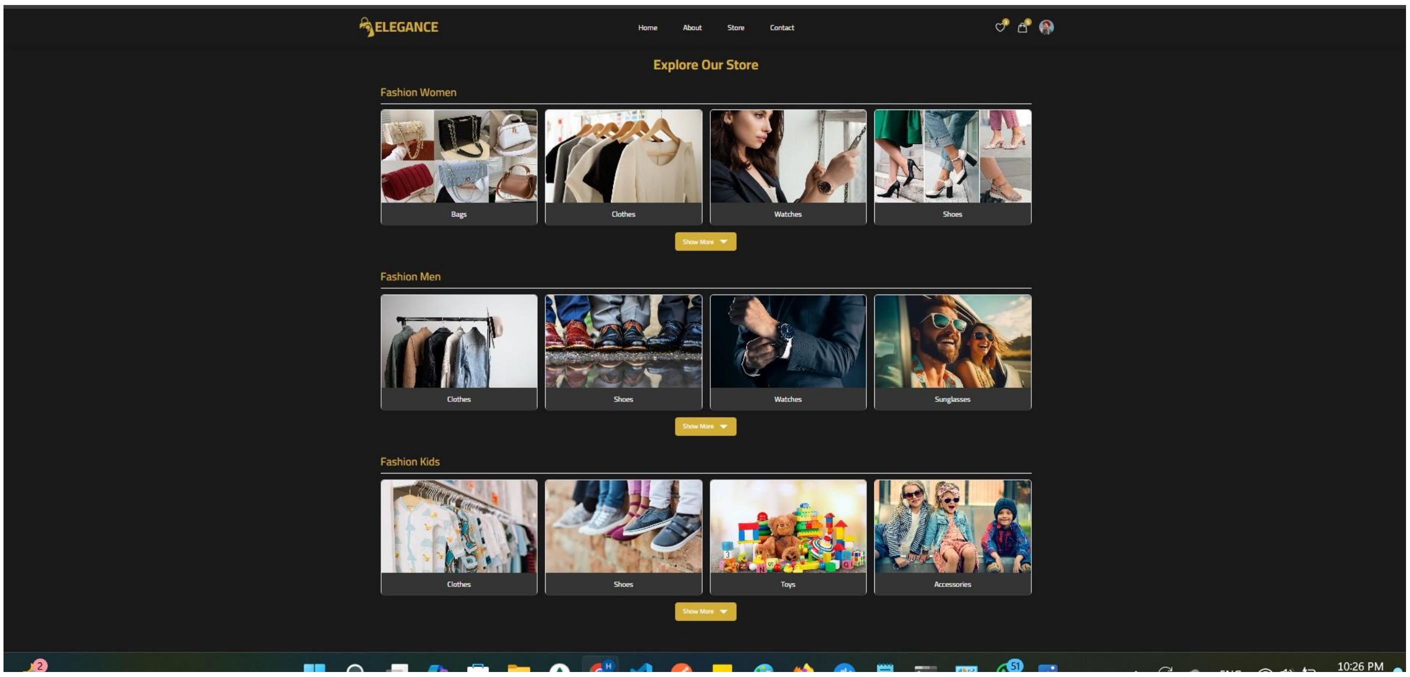
Home	Privacy Policy
About	Terms of Service
Store	FAQ
Contact us	Shipping Info

Customer Serveice

(123) 456-7890

support@fashionelegance.com

All rights reserved, Elegance © 2025 .


ELEGANCE

Home About Store Contact

Explore Our Store

Fashion Women

Bags Clothes Watches Shoes

Show More

Fashion Men

Clothes Shoes Watches Sunglasses

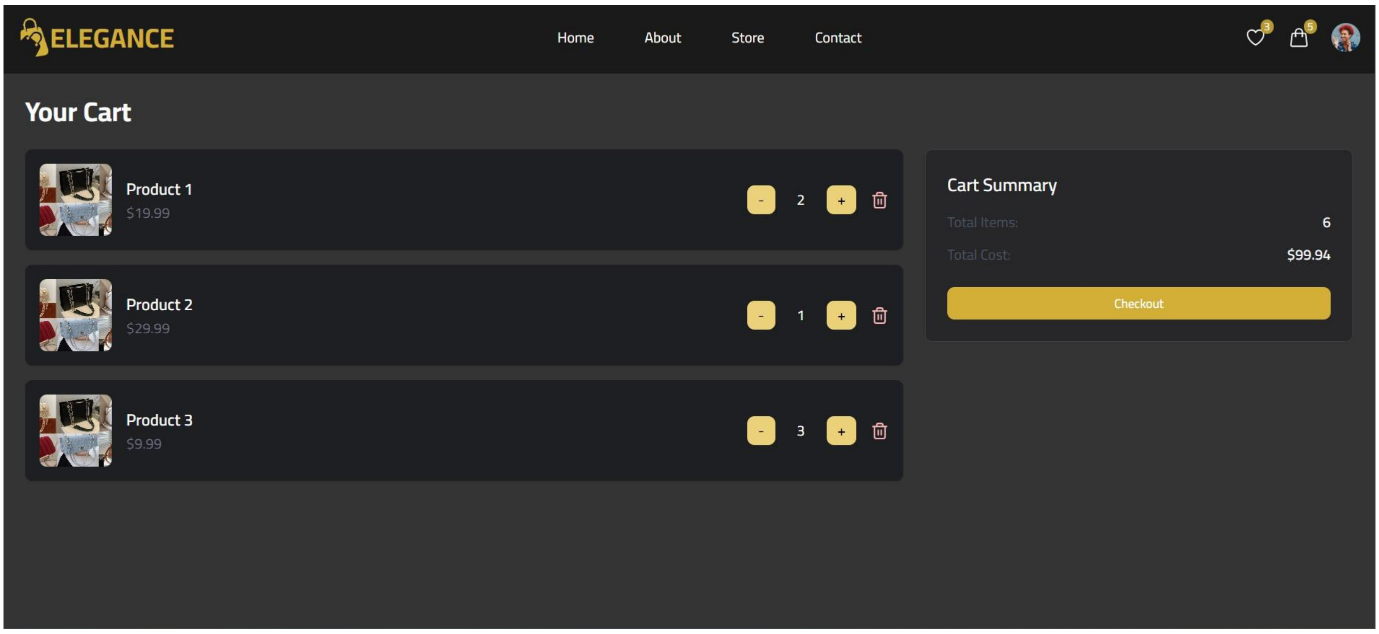
Show More

Fashion Kids

Clothes Shoes Toys Accessories

Show More

10:26 PM


ELEGANCE

Home About Store Contact

Your Cart

Product 1	$19.99	2
Product 2	$29.99	1
Product 3	$9.99	3

Cart Summary

Total Items:	6
Total Cost:	$99.94

Checkout

7.2 UI/UX Guidelines

- Design principles, color schemes, typography, and accessibility considerations.

---

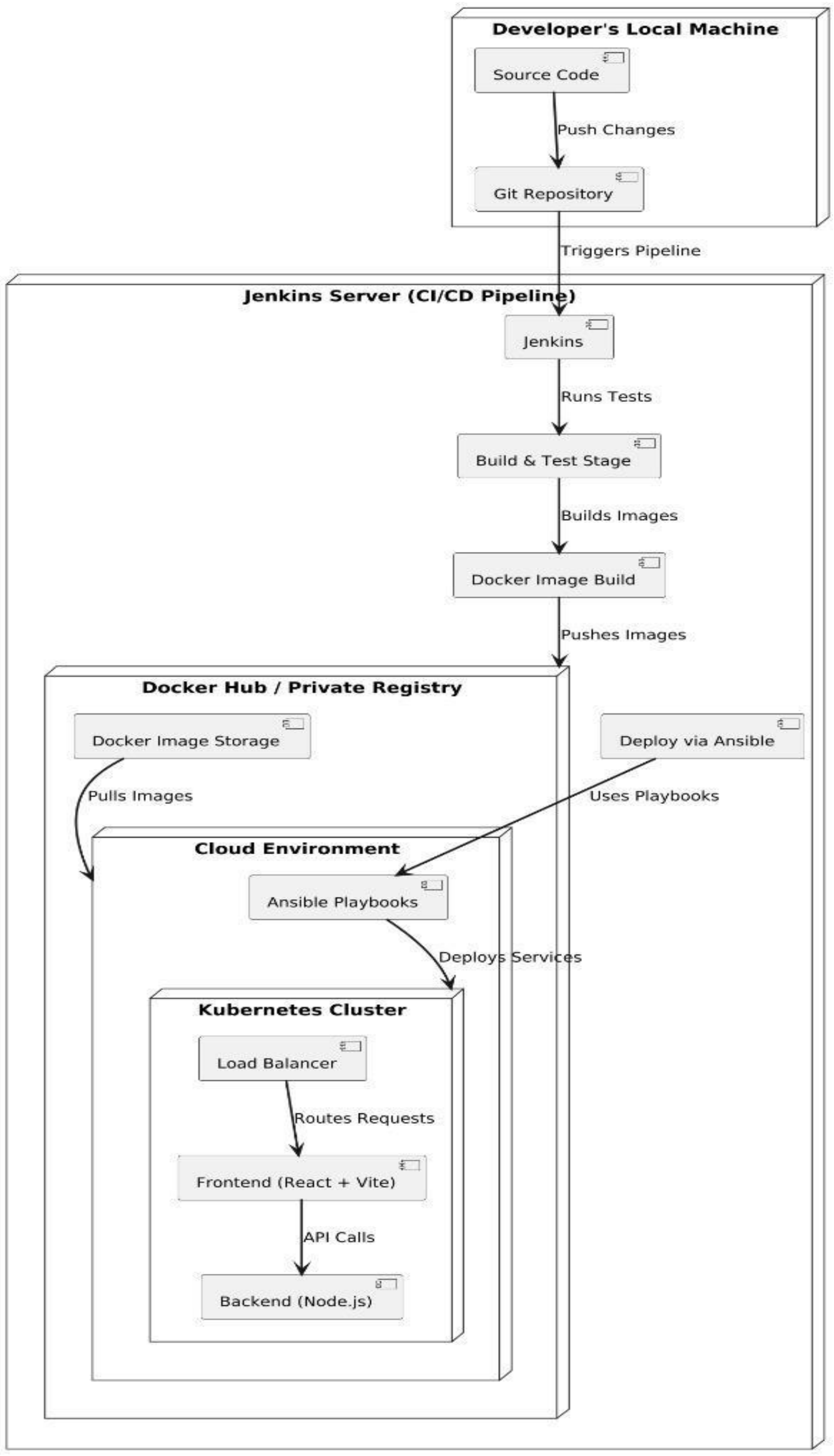
8. System Deployment & Integration

8.1 Technology Stack

- **Frontend:** React + Vite
- **Backend:** Node.js
- **Database:** MongoDB
- **DevOps:** Jenkins (CI/CD), Docker (containers), Ansible (configuration management), Kubernetes (orchestration)

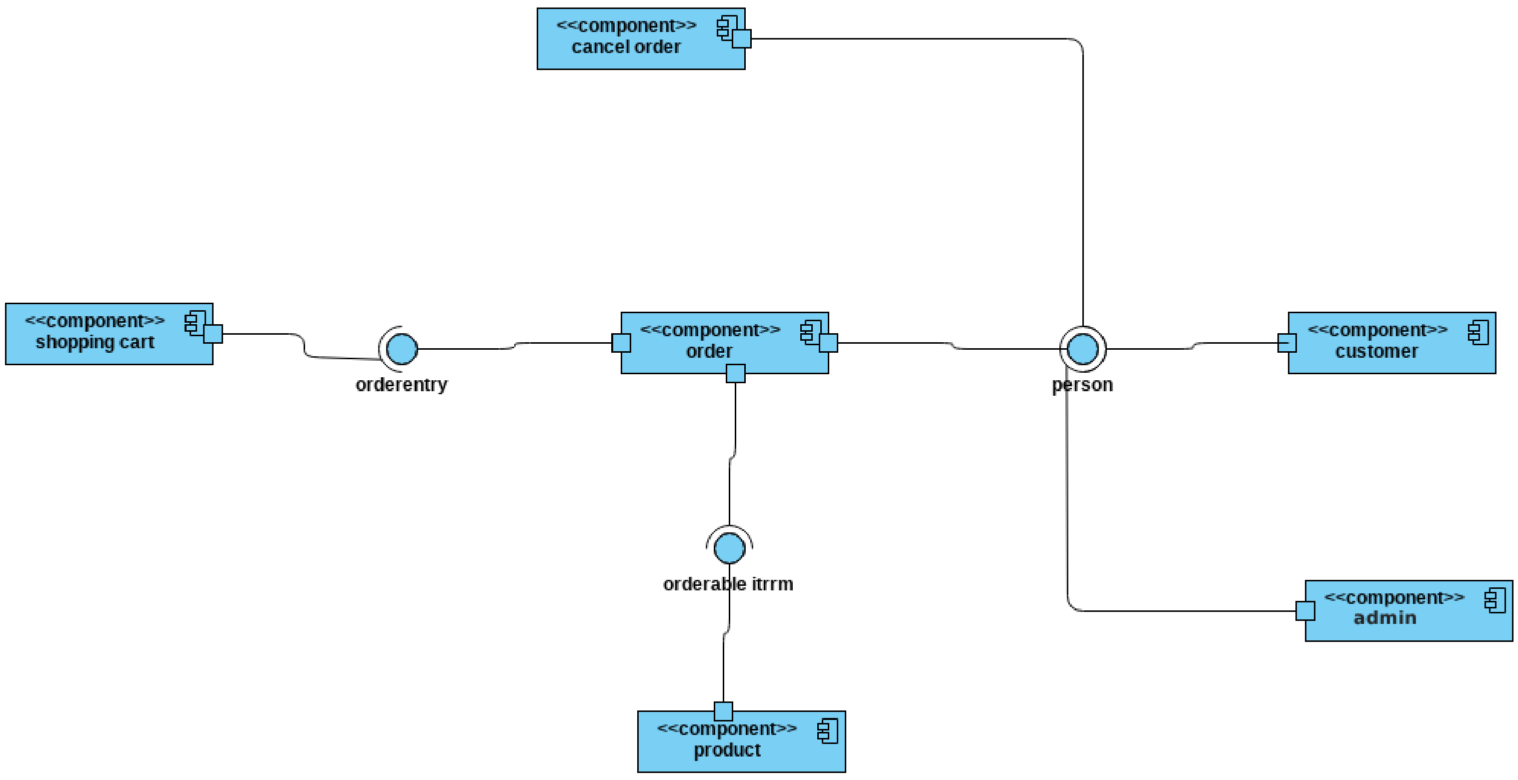
8.2 Deployment Diagram

- **Jenkins:** Polls Git repository for changes.
- **Docker:** Builds images for frontend & backend.
- **Ansible (optional):** Manages infrastructure configs or application environment.
- **Kubernetes:** Deploys Docker containers, manages scaling, load balancing, and service discovery.



8.3 Component Diagram

- High-level system components and their dependencies.



9. Additional Deliverables

9.1 API Documentation

- If the system includes APIs, provide documentation for endpoints and usage.

9.2 Testing & Validation

- Unit tests, integration tests, and user acceptance testing plan.

9.3 Deployment Strategy

- **Staging Environment:** Automated pipeline triggers build and test. If successful, deploy to staging.
- **Production Environment:** Manual approval or automated if tests pass. Rolling updates or blue-green deployment.
- **Monitoring & Logging:** Integrate with tools like Prometheus, Grafana, or ELK Stack for real-time metrics and logs.

9.4 Maintenance & Future Improvements

- **Automated Rollback:** If a deployment fails health checks, revert to the last stable version.
- **Infrastructure as Code:** Use Terraform or Ansible for consistent environment provisioning.
- **Security Hardening:** Regular scans for vulnerabilities, SSL certificates, and secrets management.

10. Conclusion

This documentation outlines the system analysis and design for an automated DevOps deployment of an e-commerce web application. By adhering to these guidelines—covering architecture, database design, DevOps integration, and UI/UX—the project will be well-structured, maintainable, and scalable.