VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
On

Object Oriented Modelling

(23CS5PCOOM)

Submitted by

RUTH MARY PAUL (1BM22CS360)
Semester & Section: 5 F
Batch: 4

In partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
In
COMPUTER SCIENCE AND ENGINEERING

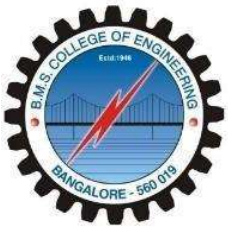
B. M. S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Sep-2024 to Feb-2025

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled "**Object Oriented Modelling (23CS5PCOOM)**" carried out by **Ruth Mary Paul (1BM22CS360)**, who is bonafide student of B.M.S. College of Engineering. It is in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements of the above-mentioned subject
and the work prescribed for the said degree.

Prof. Rekha G S Assistant Professor Department of CSE, BMSCE	Dr.KavithaSooda Professor & HOD Department of CSE, BMSCE

INDEX

1. Hotel Management System

1.1 SRS Document:

1. Introduction 1.1 Purpose

This document defines the software requirements for the Hotel Management System (HMS). It serves as a blueprint for the development team to achieve project goals efficiently.

1.2 Scope

The HMS will streamline hotel operations, improve customer satisfaction, and enhance efficiency. It includes functionalities such as reservations, room management, billing, and reporting, with an estimated development cost and timeline.

1.3 Overview

The HMS is a web-based platform designed to manage hotel operations efficiently, ensuring a user-friendly experience for administrators, staff, and guests.

---

2. General Description

The HMS automates hotel operations like reservations, billing, and reporting. It enhances productivity and service quality while reducing errors.

Features and Benefits:

- User Objectives: Simplify operations and improve guest experiences.
- User Characteristics: Supports administrators, staff, and guests.
- Importance: Streamlines operations, improves accuracy, and boosts customer satisfaction.

---

3. Functional Requirements

1. Reservation Management:
    - Book, modify, or cancel reservations.
    - Search for available rooms by date and type.
2. Room Management:
    - Track room status (occupied, available, maintenance).

    - Categorize rooms (Single, Double, Suite).
3. Billing and Payments:
    - Generate invoices.
    - Integrate with payment gateways.
4. User Management:
    - Role-based access control.
5. Reporting:
    - Generate operational and financial reports.

---

4. Interface Requirements

- Software Interfaces: Integration with payment gateways and external booking platforms.
- User Interfaces: Web-based dashboards for admins, staff, and guests.
- Communication Interfaces: Secure API endpoints and HTTPS protocols.

---

5. Performance Requirements •

Support up to 1000 concurrent users.

- Response times under 2 seconds.
- Maintain 99.9% uptime.

---

6. Design Constraints

- Hardware: Standard desktop and mobile devices.
- Software: Use secure HTTPS and comply with GDPR and PCI-DSS.
- Algorithms: Ensure efficiency for minimal response times.

---

7. Non-Functional Attributes

- Security: Data encryption during transmission and storage.
- Portability: Accessible via modern web browsers.
- Reliability: Minimize downtime.
- Scalability: Support growing user and transaction volumes.
- Data Integrity: Prevent data loss or corruption.

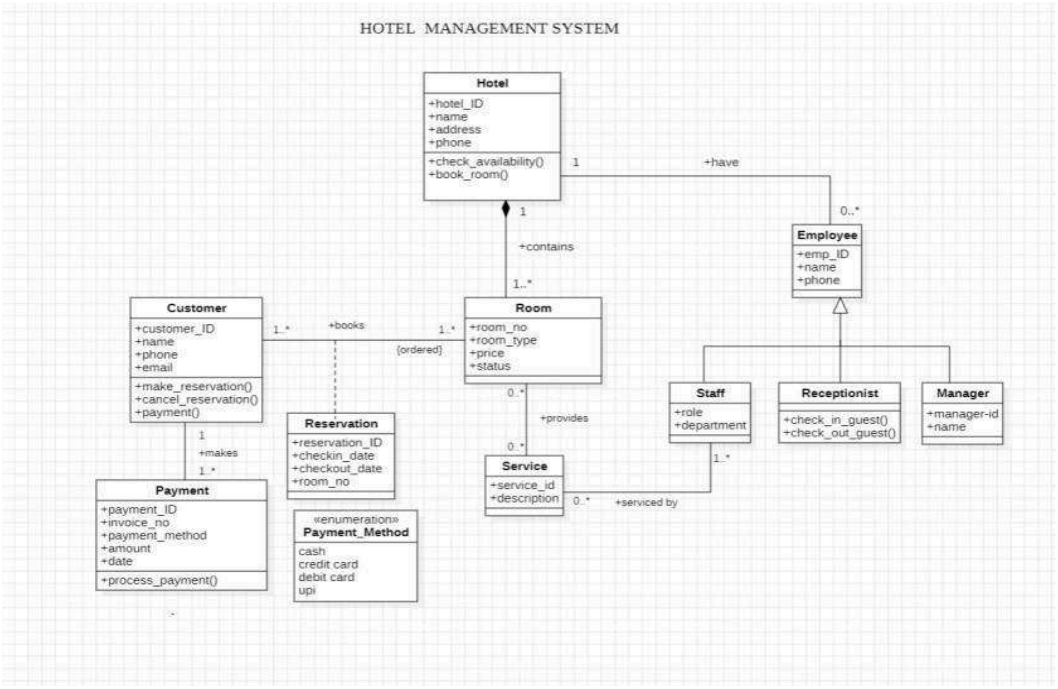
---

8. Schedule and Budget

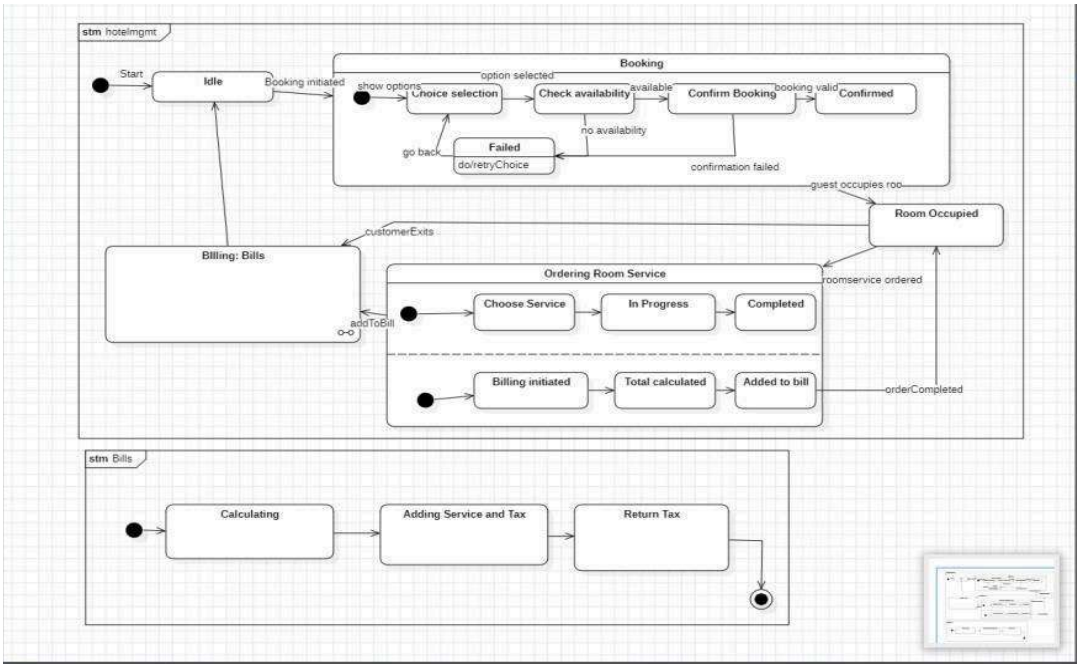
Schedule

- Phase 1: Requirements and analysis - 1 month.
- Phase 2: Design - 1 month.
- Phase 3: Development and testing - 3 months.
- Phase 4: Deployment and training - 1 month. Budget
- Development: $50,000.
- Testing and QA: $10,000.
- Deployment: $15,000.
- Total: $75,000.

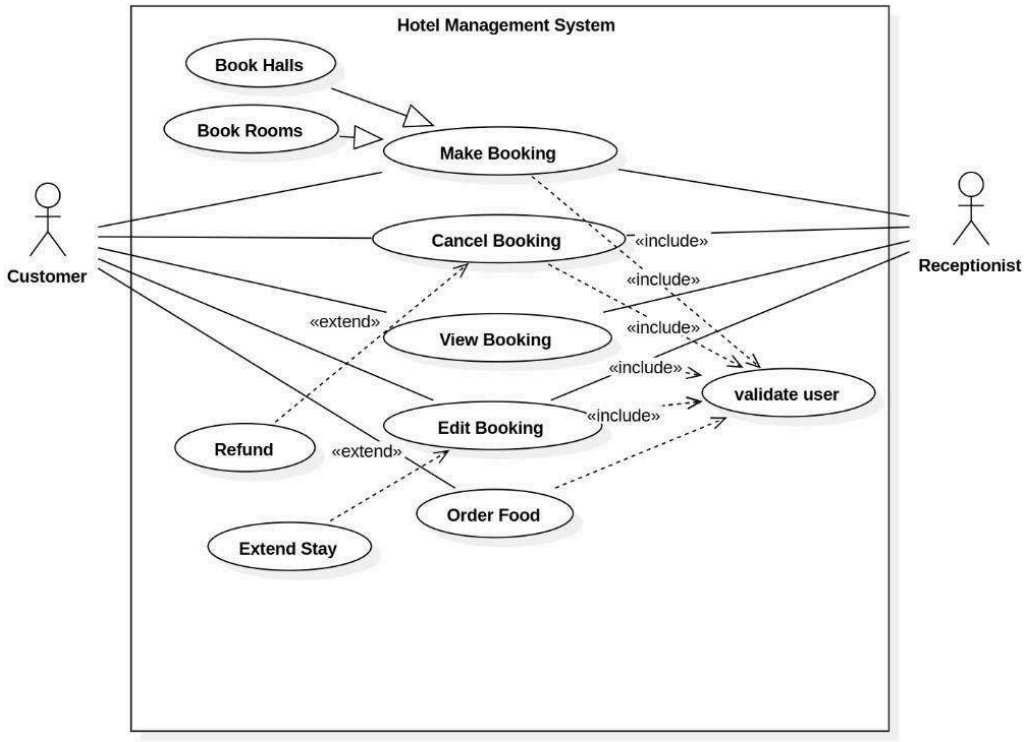
1.2 Class Diagram:



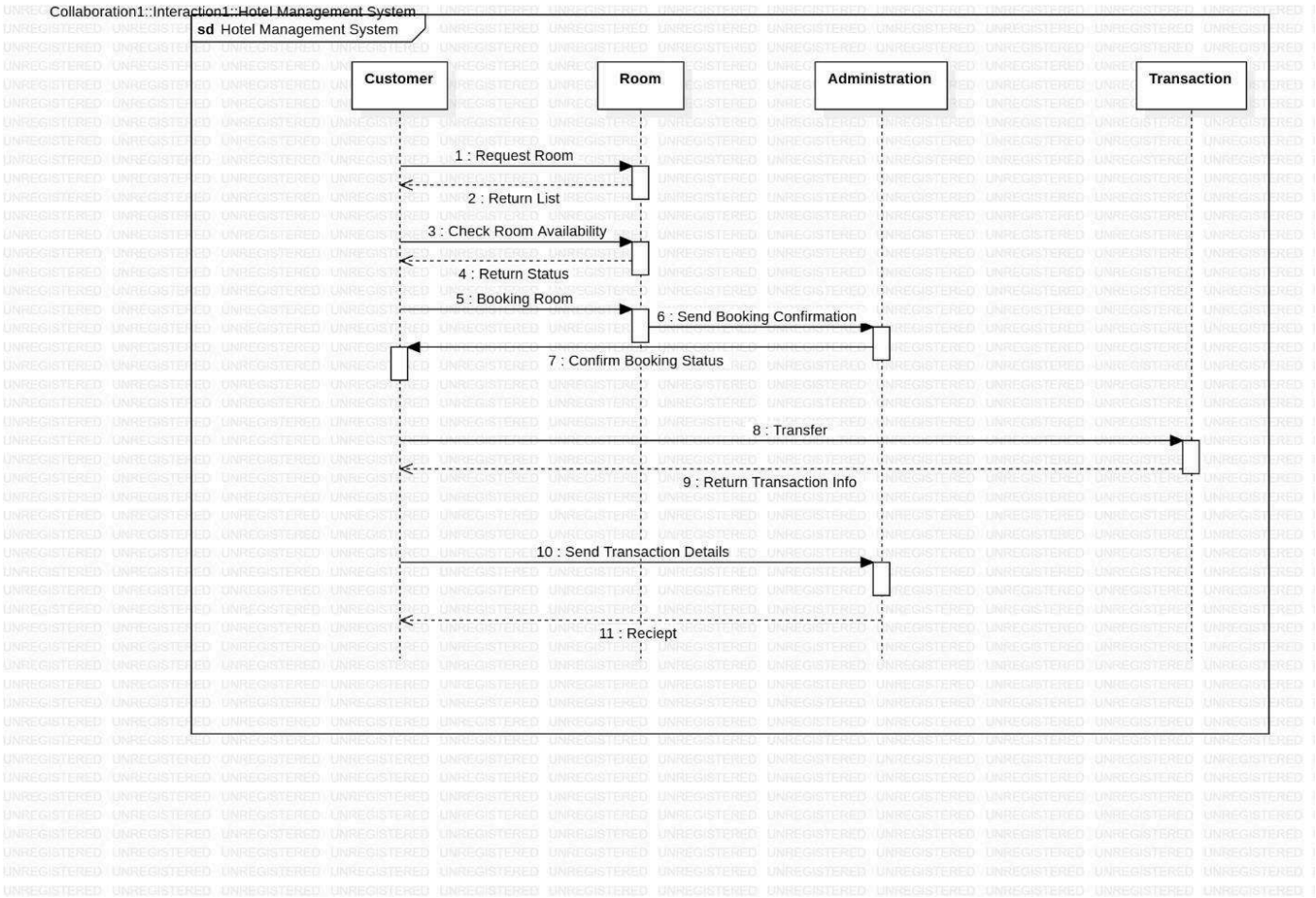
1.3 State Diagram:



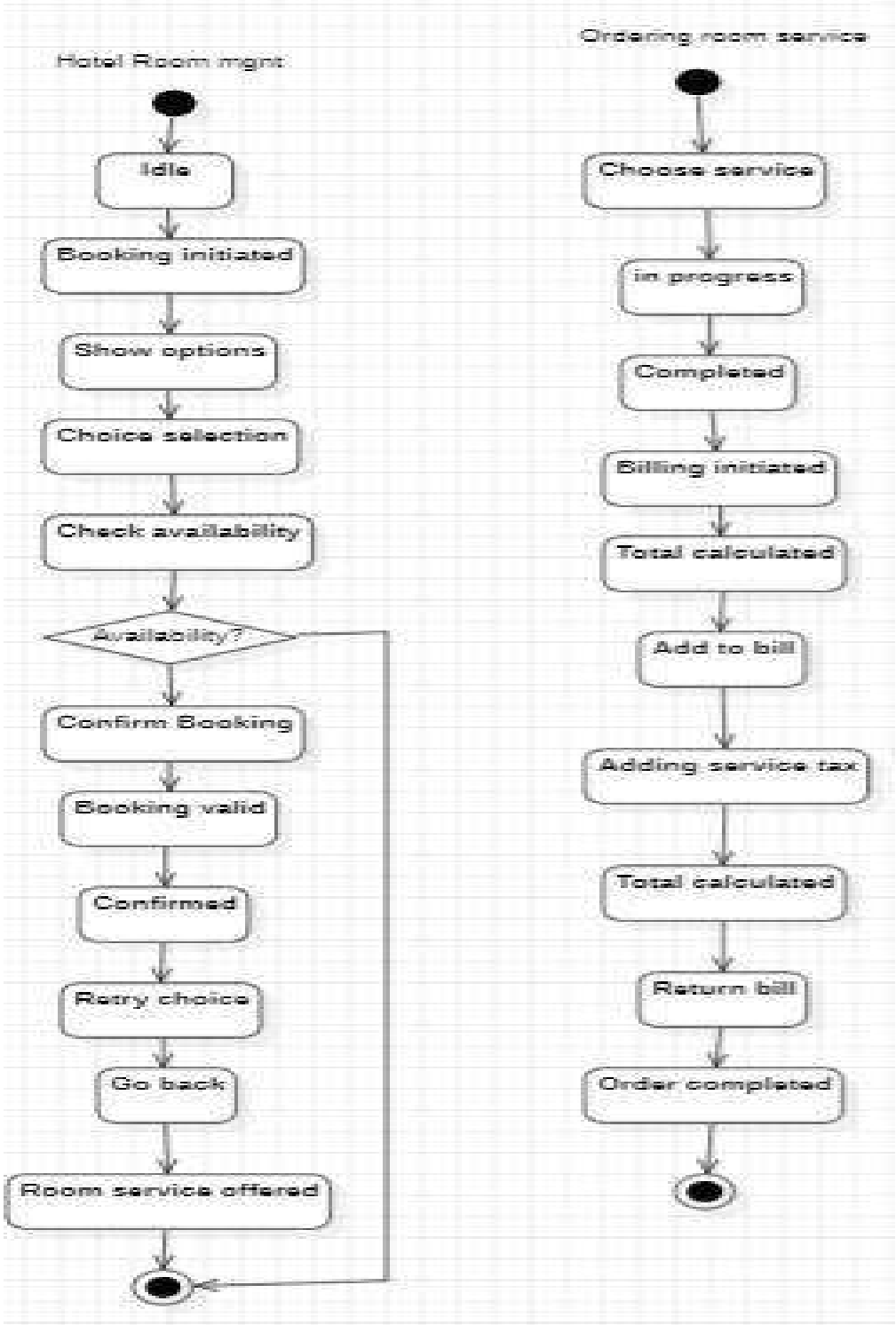
1.4 Use Case Diagram:



1.5 Sequence Diagram:



1.6 Activity Diagram



2. Credit Card Processing

2.1 SRS Document:

1. Introduction

1.1 Purpose

This document defines the requirements for a Credit Card Processing System (CCPS). It serves as a comprehensive guide for developers to build a secure, efficient, and scalable system for processing credit card transactions.

1.2 Scope

The CCPS will handle authorization, transaction processing, settlement, and reporting for credit card payments. The system is designed for use by merchants, financial institutions, and customers, providing a seamless and secure payment experience.

1.3 Overview

The CCPS ensures secure and real-time processing of credit card transactions, fraud detection, and compliance with industry standards. It aims to provide a reliable platform for financial transactions with minimal errors.

---

2. General Description

The CCPS facilitates secure and efficient payment processing for merchants and customers. It includes core functionalities such as transaction authorization, settlement, chargeback handling, and reporting, while ensuring compliance with PCI-DSS.

Features and Benefits:

- User Objectives: Enable fast and secure credit card transactions.
- User Characteristics: Supports merchants, customers, and financial institutions.
- Importance: Enhances trust in digital payments and reduces operational complexities for merchants.

---

3. Functional Requirements
    1. Transaction Processing:
        - Authorize credit card transactions in real-time.
        - Process refunds and chargebacks.
    2. Fraud Detection:
        - Implement AI-based fraud detection mechanisms.
        - Alert merchants of suspicious activities.
    3. Settlement:
        - Reconcile daily transactions with financial institutions.
        - Generate settlement reports for merchants.
4. User Management:
    - Provide role-based access for administrators and merchants.
5. Reporting:
    - Generate transaction, settlement, and chargeback reports.

---

4. Interface Requirements

- Software Interfaces: Integration with payment gateways, banks, and fraud detection services.
- User Interfaces: Web-based dashboards for merchants and administrators.
- Communication Interfaces: Use secure HTTPS and API endpoints for data exchange.

---

5. Performance Requirements

- Handle up to 10,000 transactions per second.
- Response times under 1 second for transaction processing.
- Ensure 99.99% uptime.

---

6. Design Constraints

- Hardware: Deployable on cloud-based infrastructure.
- Software: Use PCI-DSS-compliant frameworks.
- Algorithms: Leverage secure encryption algorithms for data handling.

---

7. Non-Functional Attributes

- Security: Ensure end-to-end encryption and secure storage of cardholder data.
- Portability: Accessible via web and mobile applications.
- Reliability: Guarantee continuous transaction processing with disaster recovery mechanisms.
- Scalability: Support increasing transaction volumes as the business grows.
- Data Integrity: Maintain accurate records for all transactions.

---

8. Schedule and

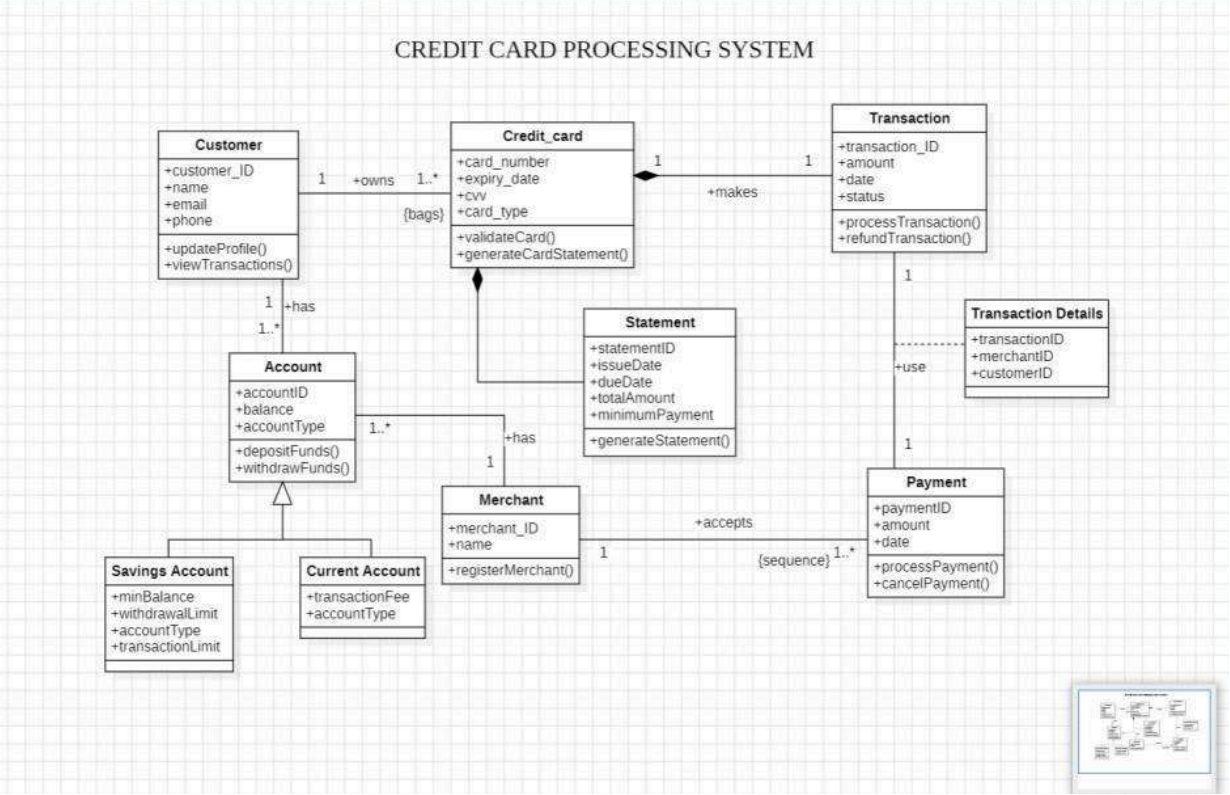
Budget Schedule

- Phase 1: Requirements analysis - 2 weeks.
- Phase 2: System design - 3 weeks.
- Phase 3: Development and testing - 8 weeks.
- Phase 4: Deployment and training - 2 weeks.

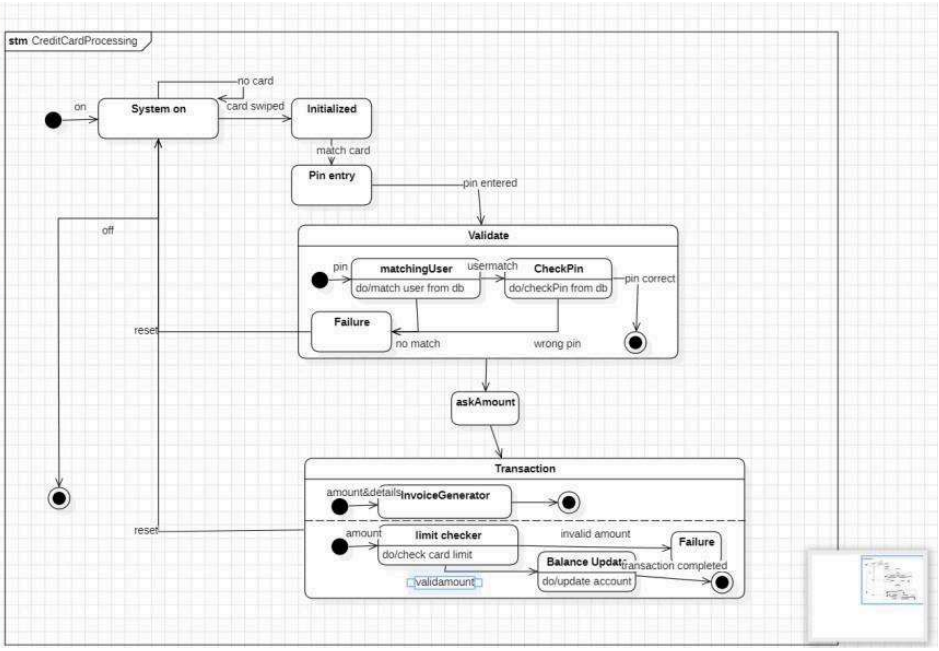
Budget

- Development: $100,000.
- Testing and QA: $25,000.
- Deployment: $20,000.
- Total: $145,000.

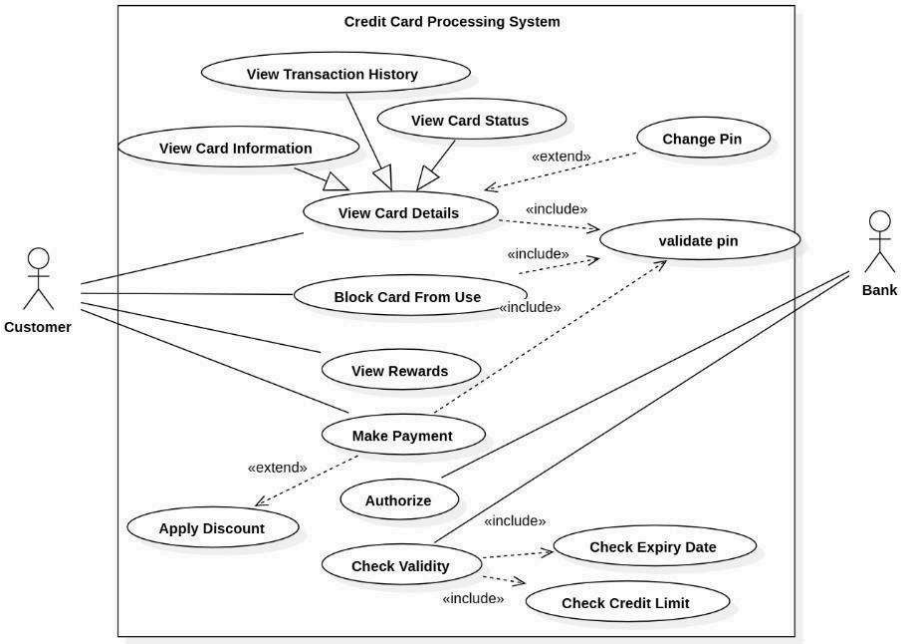
2.2 Class Diagram:



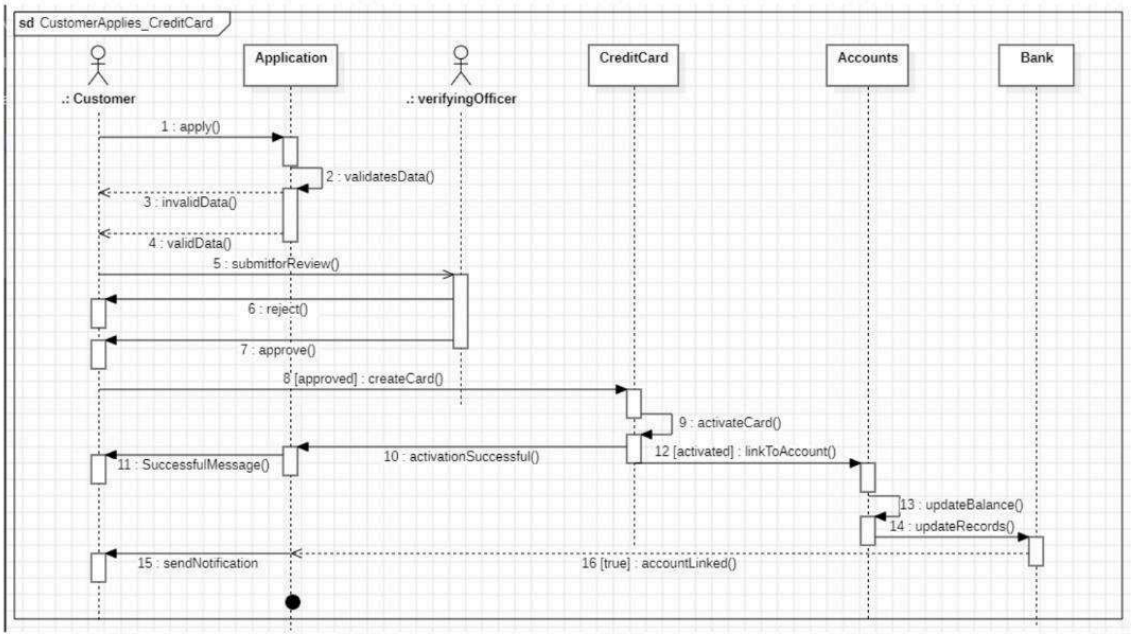
2.3 State Diagram:



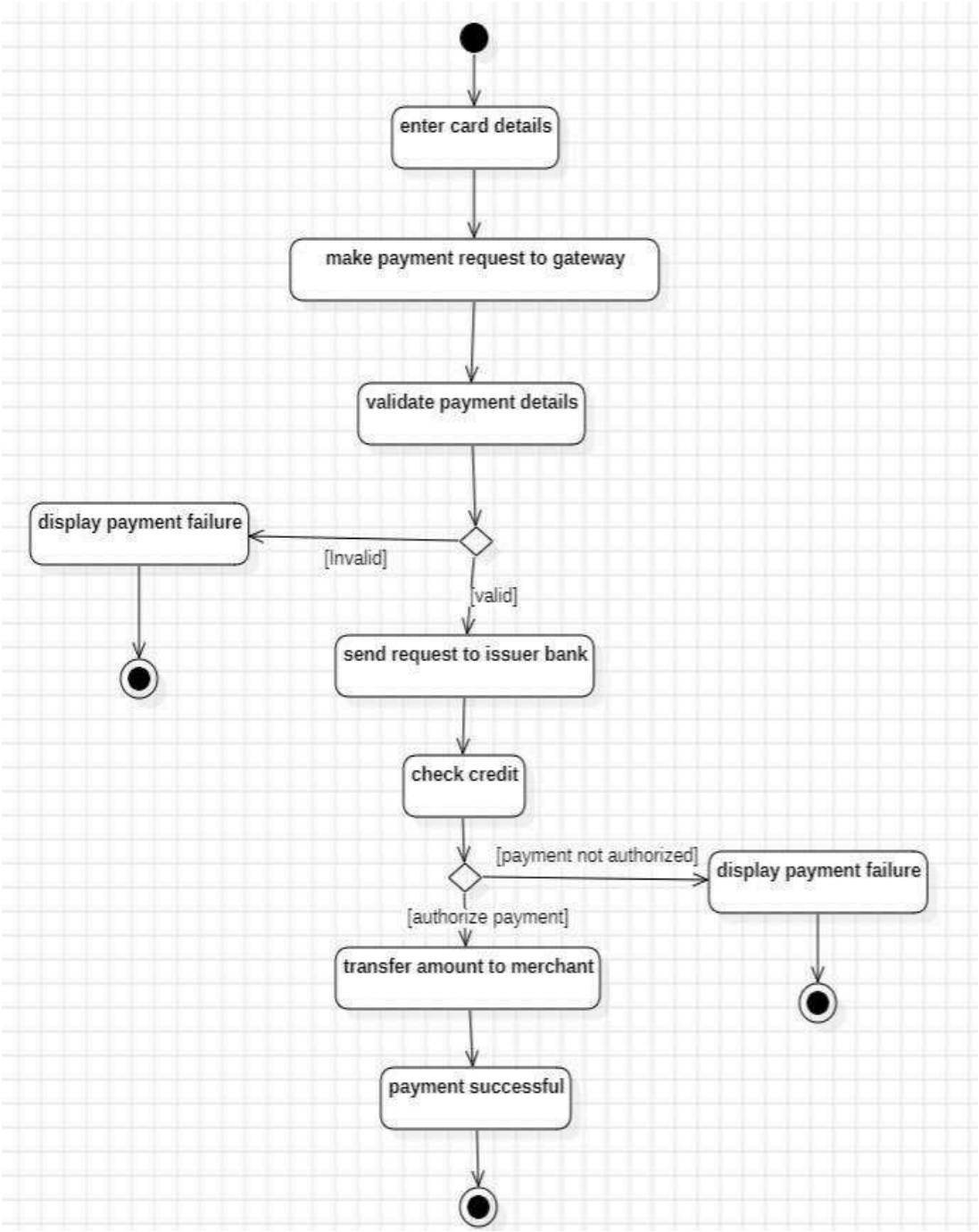
2.4 Use Case Diagrams:



2.5 Sequence Diagrams:



2.6 Activity Diagram



3. Library Management System

3.1 SRS Document:

1. Introduction

1.1 Purpose

This document outlines the requirements for the Library Management System (LMS). It serves as a guide for the development team to create a robust, efficient, and user-friendly system for managing library operations.

1.2 Scope

The LMS will automate key library functions such as book management, user registration, lending, and returns. It aims to improve operational efficiency, reduce manual errors, and enhance user experience for library staff and members.

1.3 Overview

The LMS is a web-based application designed to manage library resources and services effectively. It supports administrators, librarians, and members by providing seamless access to library functionalities and real-time information.

---

2. General Description

The LMS is developed to streamline the management of library resources, including books, users, and transactions. It facilitates efficient book lending and returning, reduces manual work, and improves resource tracking.

Features and Benefits:

- User Objectives: Simplify library operations and enhance resource accessibility.
- User Characteristics: Supports librarians, administrators, and library members.
- Importance: Ensures accurate tracking of resources, reduces operational delays, and enhances user satisfaction.

---

3. Functional Requirements

    1. Book Management:

        - Add, update, and delete book records.
        - Categorize books by genre, author, and availability.

    2. User Management:

        - Register and manage library members and staff.
        - Implement role-based access control.

    3. Lending and Returns:

        - Issue and return books with automated due date calculation.
        - Generate overdue notifications for members.

4. Search and Cataloging:

    - Enable keyword-based search for books.
    - Provide catalog views by genre, author, or availability.

5. Reporting:

    - Generate reports on issued books, overdue items, and membership statistics.

---

4. Interface Requirements

- Software Interfaces: Integration with databases for storage and retrieval of records.
- User Interfaces: Web-based dashboards for librarians and members.
- Communication Interfaces: Secure API endpoints for data exchange and mobile app integration.

---

5. Performance Requirements

- Support up to 500 concurrent users.
- Search and transaction response times under 2 seconds.
- Maintain 99.9% uptime.

---

6. Design Constraints

- Hardware: Compatible with standard desktop and mobile devices.
- Software: Use secure frameworks and adhere to GDPR standards.
- Algorithms: Optimize search and transaction processes for minimal latency.

---

7. Non-Functional Attributes

- Security: Protect user and book data with encryption.
- Portability: Accessible via web browsers and mobile apps.
- Reliability: Ensure consistent functionality with regular backups.
- Scalability: Handle growing user bases and book collections seamlessly.
- Data Integrity: Ensure accuracy in book and user records.

---

8. Schedule and

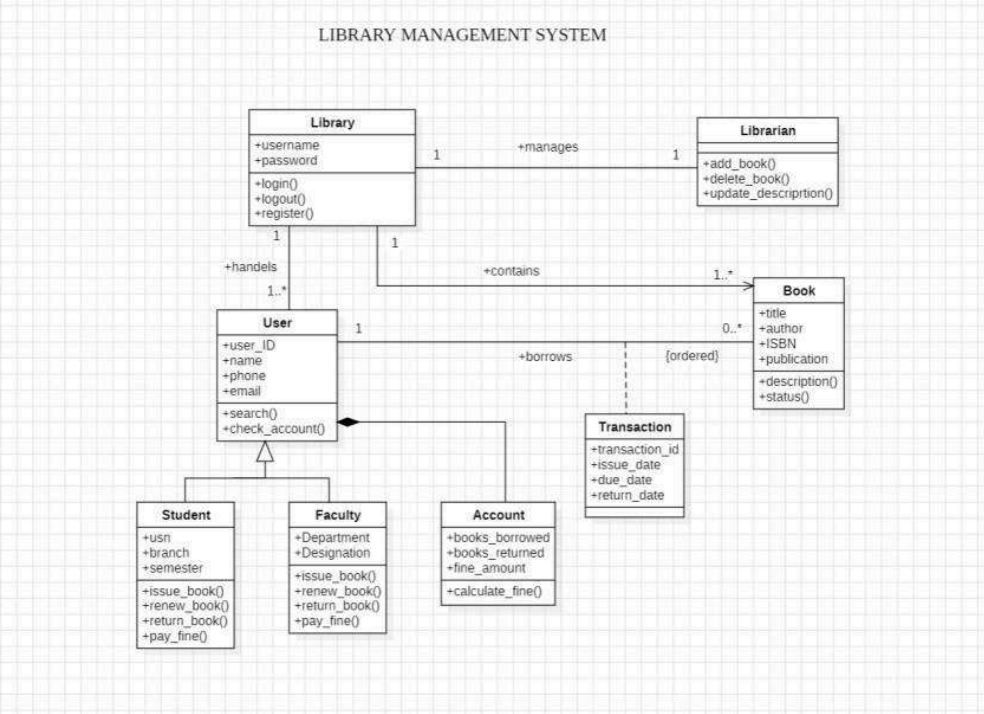
Budget Schedule

- Phase 1: Requirements analysis - 2 weeks.
- Phase 2: System design - 3 weeks.
- Phase 3: Development and testing - 8 weeks.
- Phase 4: Deployment and training - 2 weeks.

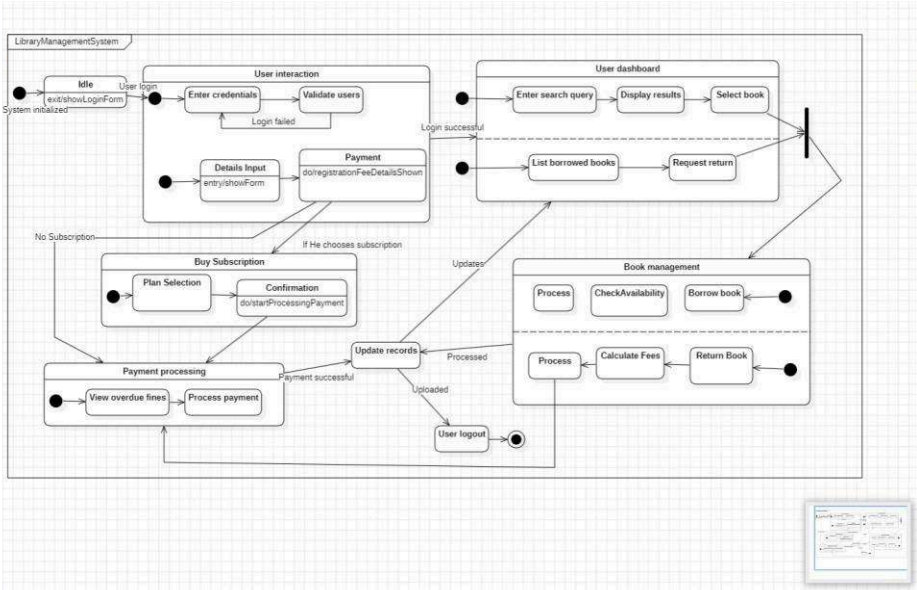
Budget

- Development: $70,000.
- Testing and QA: $15,000.
- Deployment: $10,000.
- Total: $95,000.

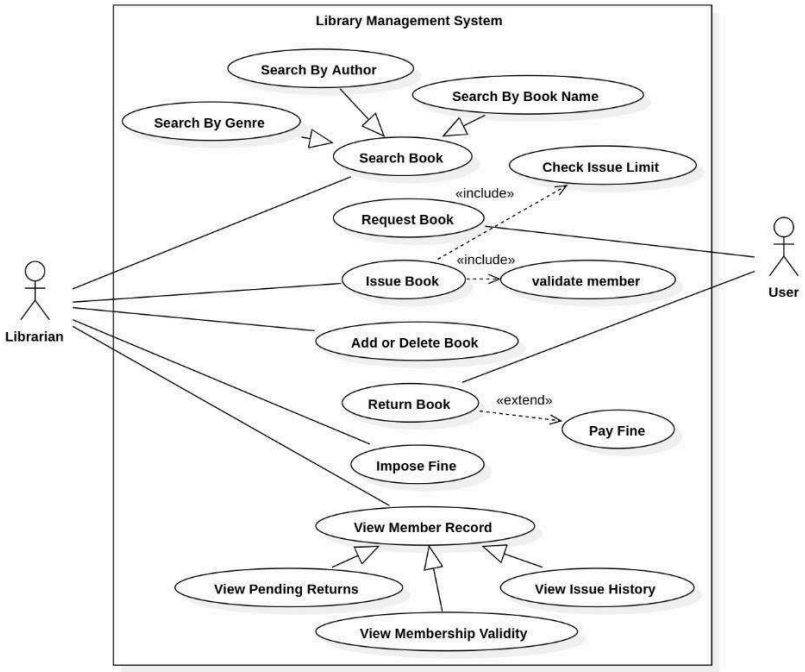
3.2 Class Diagram:



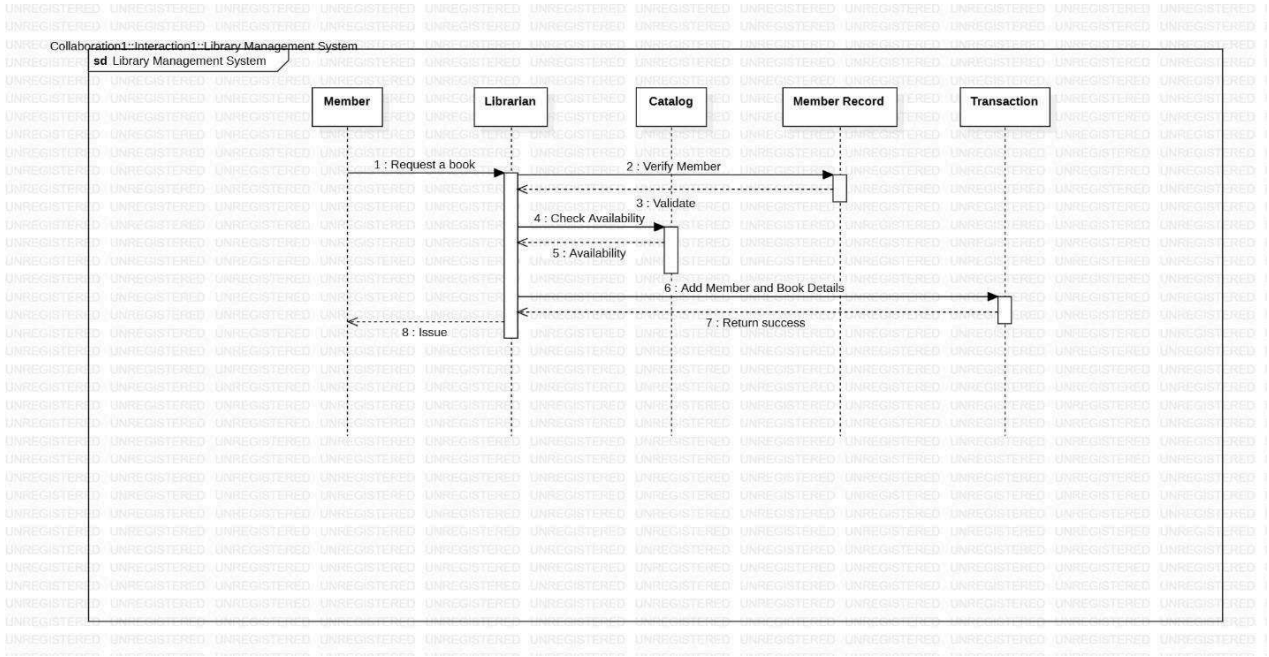
3.3 State Diagram:



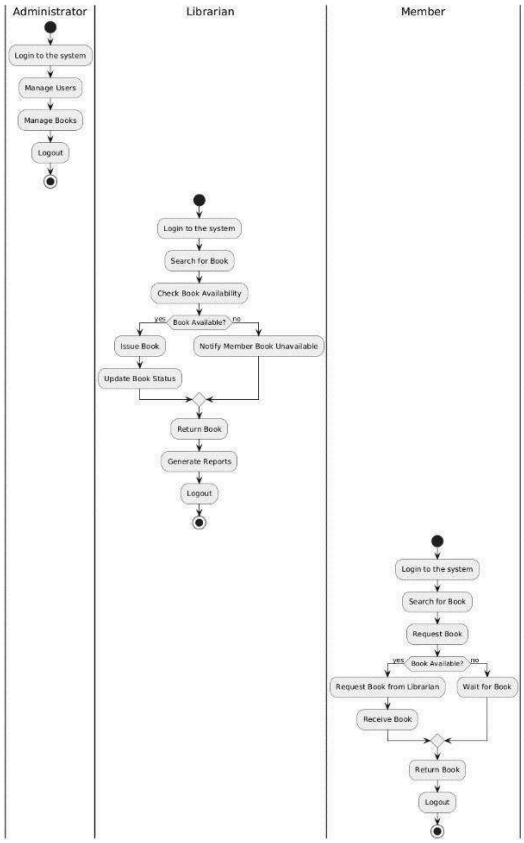
3.4 Use Case Diagram:



3.5 Sequence Diagram:



3.6 Activity Diagram



4. Stock Maintenance System

4.1 SRS Document:

1. Introduction

1.1 Purpose

This document specifies the requirements for the Stock Maintenance System (SMS). It is intended to guide the development of a robust and efficient system for managing inventory and stock-related operations.

1.2 Scope

The SMS will manage inventory tracking, stock updates, reporting, and supplier interactions. It aims to optimize inventory control, reduce stock discrepancies, and streamline restocking processes. The system is designed for use by warehouses, retail businesses, and supply chain managers.

1.3 Overview

The SMS automates stock monitoring and control, ensuring real-time updates and accuracy in inventory records. It reduces manual workload and enhances operational efficiency.

---

2. General Description

The SMS supports the management of inventory across multiple locations, tracks stock levels, and facilitates supplier communication for restocking. It provides real-time visibility of inventory and supports decision-making through comprehensive reporting.

Features and Benefits:

- User Objectives: Minimize stock discrepancies and improve restocking processes.
- User Characteristics: Designed for inventory managers, warehouse staff, and procurement teams.

- Importance: Enhances accuracy in stock records and ensures timely restocking, reducing operational delays.

---

3. Functional Requirements
    1. Inventory Management:
        - Add, update, and delete stock records.
        - Categorize items by type, location, and supplier.
    2. Stock Monitoring:
        - Real-time tracking of stock levels.
        - Generate low-stock alerts.
    3. Supplier Management:
        - Maintain supplier records.
        - Automate restocking requests based on stock thresholds.
4. Reporting:
    - Generate inventory status reports.
    - Provide insights on stock movement and trends.
5. User Management:
    - Role-based access control for admins and staff.

---

4. Interface Requirements

- Software Interfaces: Integration with ERP systems and supplier portals.
- User Interfaces: Web-based dashboard and mobile application for inventory management.
- Communication Interfaces: Secure API endpoints for third-party integrations.

---

5. Performance Requirements

- Support up to 1000 concurrent users.
- Response times under 2 seconds for stock updates and queries.
- Maintain 99.9% uptime.

---

6. Design Constraints

- Hardware: Compatible with standard desktop and mobile devices.
- Software: Use scalable and secure cloud-based infrastructure.
- Algorithms: Implement efficient search and sorting algorithms for inventory queries.

---

7. Non-Functional Attributes

- Security: Encrypt sensitive data such as supplier details and stock information.
- Portability: Accessible on modern web browsers and mobile devices.
- Reliability: Ensure continuous functionality with regular backups.
- Scalability: Handle increasing stock volumes and user base.
- Data Integrity: Prevent discrepancies in stock records.

---

8. Schedule and

Budget Schedule

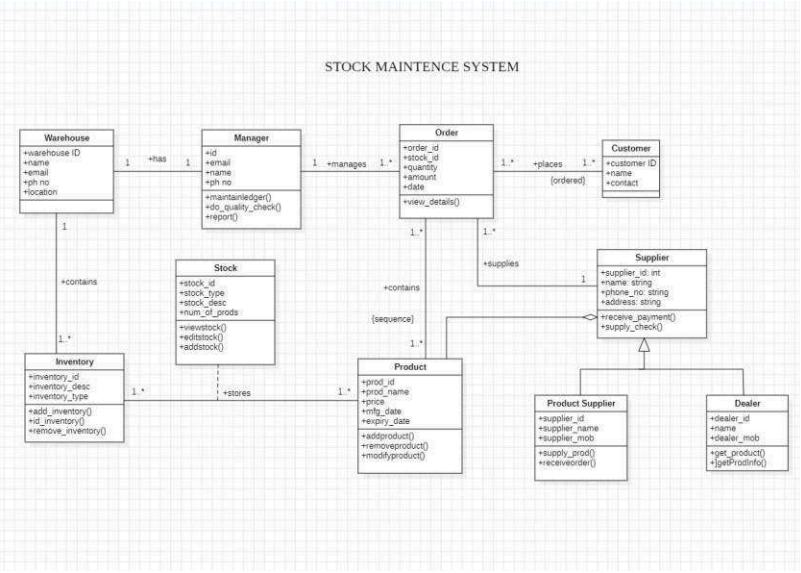
- Phase 1: Requirements analysis - 3 weeks.
- Phase 2: System design - 4 weeks.
- Phase 3: Development and testing - 10 weeks.

- Phase 4: Deployment and training - 3 weeks.

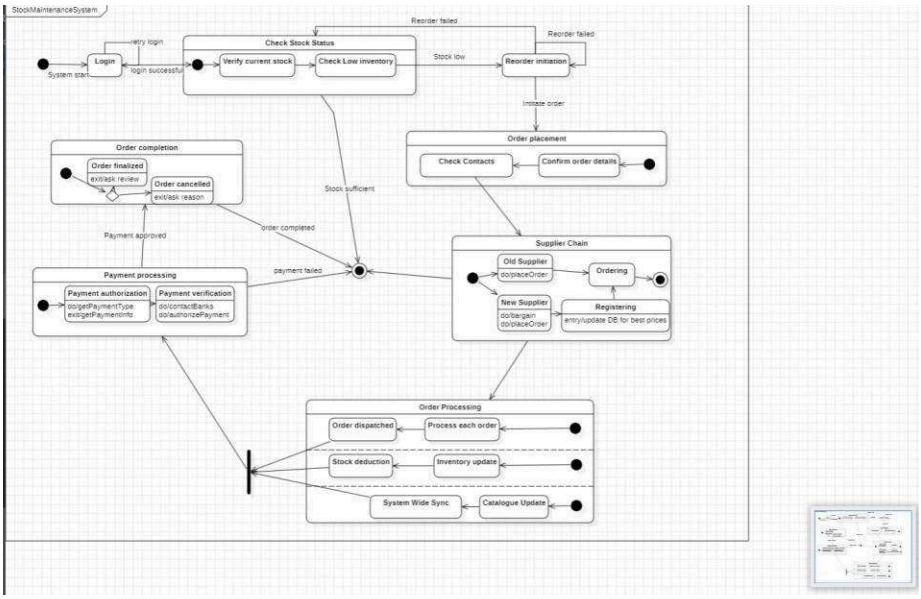
Budget

- Development: $80,000.
- Testing and QA: $20,000.
- Deployment: $15,000.
- Total: $115,000.

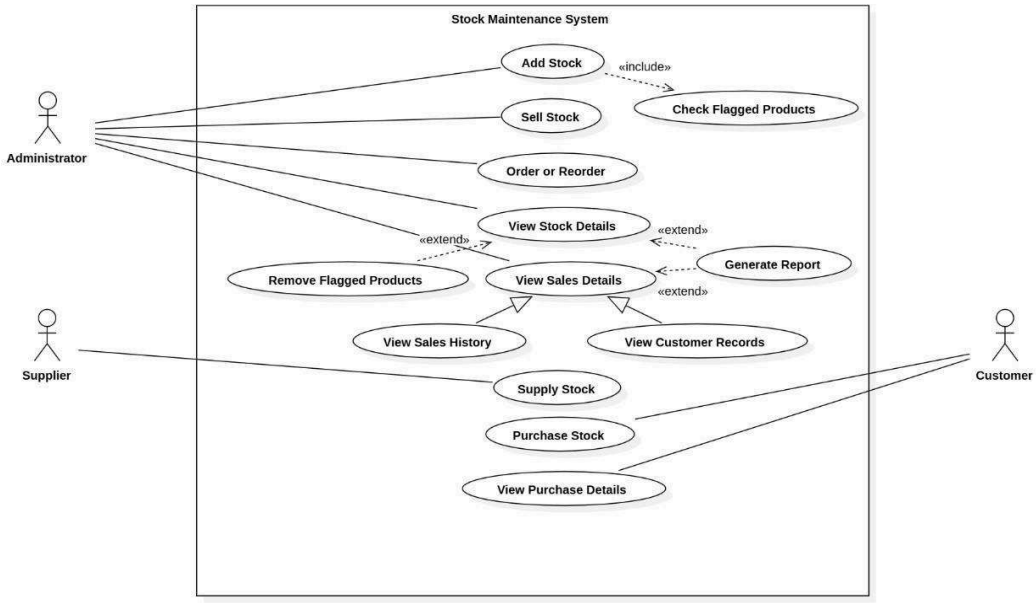
4.2 Class Diagram:



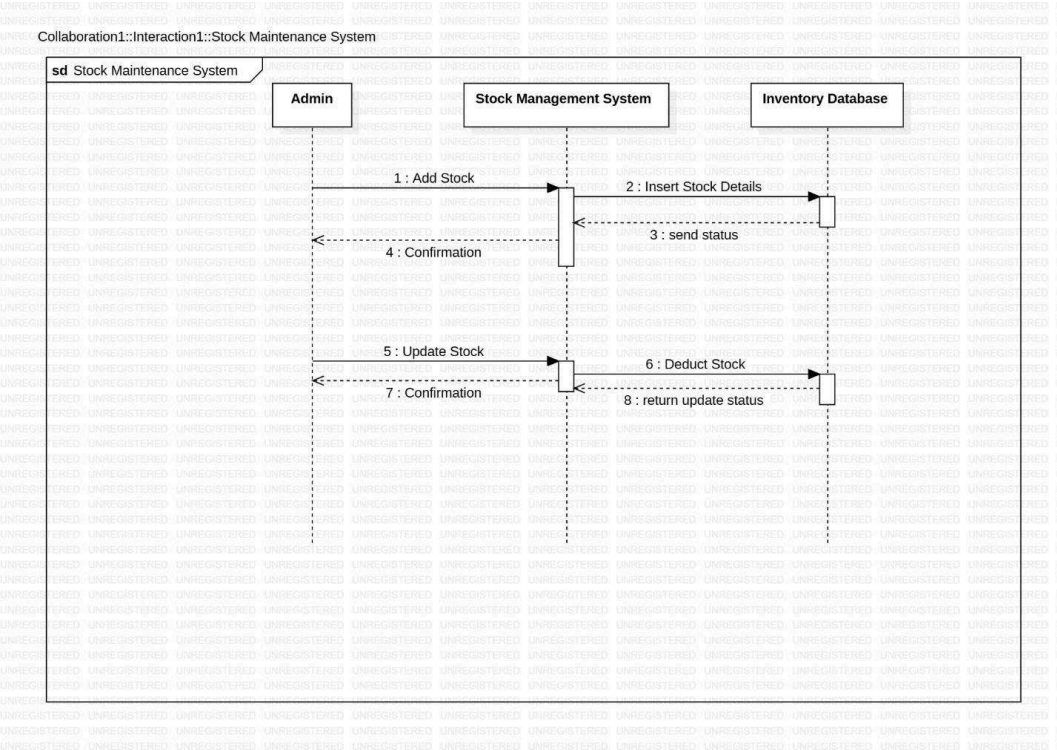
4.3 State Diagram:



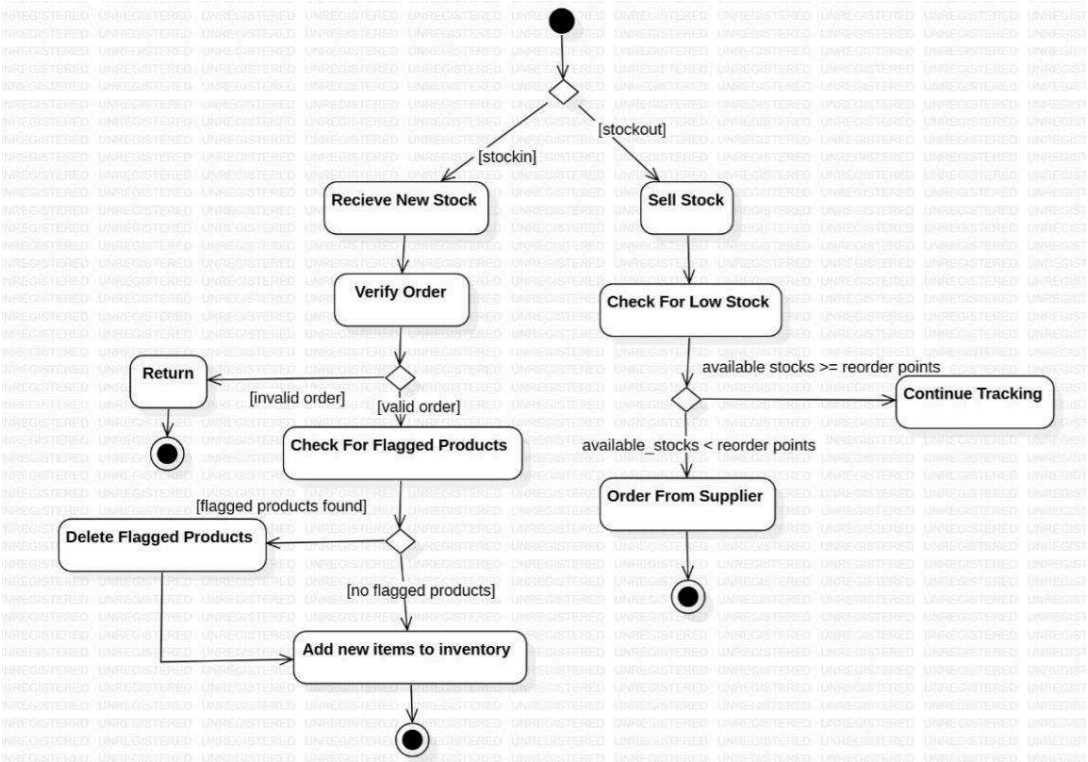
4.4 Use Case Diagram:



4.5 Sequence Diagram:



4.6 Activity Diagram:



5. Passport Automation System

5.1 SRS Document:

1. Introduction

    1.1 Purpose

This document specifies the requirements for the Passport Automation System (PAS). It serves as a guideline for the development of a system to automate passport application, processing, and issuance.

    1.2 Scope

The PAS will digitize and streamline the passport application process, including form submission, document verification, appointment scheduling, and passport issuance. It aims to reduce manual errors, improve efficiency, and enhance the applicant experience.

    1.3 Overview

The PAS is a web-based application designed to manage the end-to-end process of passport issuance, ensuring accuracy, security, and real-time tracking. It will serve applicants, government officials, and verification agencies.

---

2. General Description

The PAS automates the passport lifecycle, from application to issuance. It integrates with government databases for identity verification and provides status updates to applicants.

Features and Benefits:

- User Objectives: Simplify the passport application process and enhance operational efficiency.
- User Characteristics: Designed for applicants, passport office staff, and verification officers.

- Importance: Reduces processing time, eliminates redundancies, and improves service quality.

---

3. Functional Requirements

1. Application Submission:
    - Online form filling and document upload.
    - Integration with Aadhaar or national ID for identity verification.
2. Appointment Scheduling:
    - Select and book appointment slots for document verification.
    - Automated reminders for upcoming appointments.
3. Document Verification:
    - Digital and manual verification of submitted documents.
    - Integration with police and government agencies for background checks.
4. Passport Issuance:
    - Automated status updates.
    - Generation and printing of passport documents.
5. User Management: ○ Role-based access for applicants, staff, and verification officers.
6. Reporting and Analytics:
    - Generate reports on application volumes, processing times, and issued passports.

---

4. Interface Requirements

- Software Interfaces: Integration with government databases, police verification systems, and payment gateways.

- User Interfaces: Web-based portal and mobile application for applicants and staff.
- Communication Interfaces: Secure API endpoints for third-party integrations.

---

5. Performance Requirements

- Support up to 5000 concurrent users.
- Process applications within 5 minutes on average.
- Ensure 99.9% uptime.

---

6. Design Constraints

- Hardware: Compatible with standard desktop and mobile devices.
- Software: Use government-approved secure frameworks and adhere to data protection laws.
- Algorithms: Efficient matching and validation algorithms for document verification.

---

7. Non-Functional Attributes

- Security: Implement multi-factor authentication and data encryption.
- Portability: Accessible via web browsers and mobile devices.
- Reliability: Ensure high availability and regular backups.
- Scalability: Support growing application volumes.
- Data Integrity: Maintain accuracy in applicant and passport records.

---

8. Schedule and

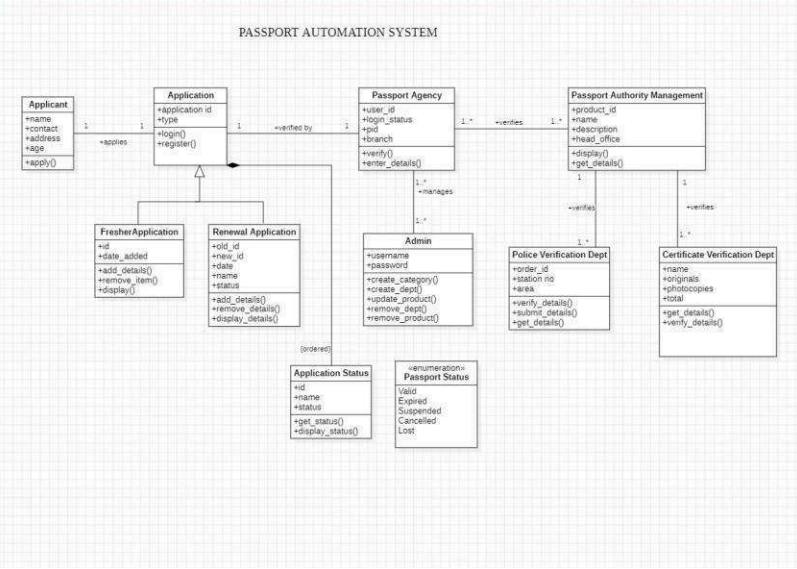
Budget Schedule

- Phase 1: Requirements analysis - 2 weeks.
- Phase 2: System design - 3 weeks.
- Phase 3: Development and testing - 8 weeks.
- Phase 4: Deployment and training - 3 weeks.

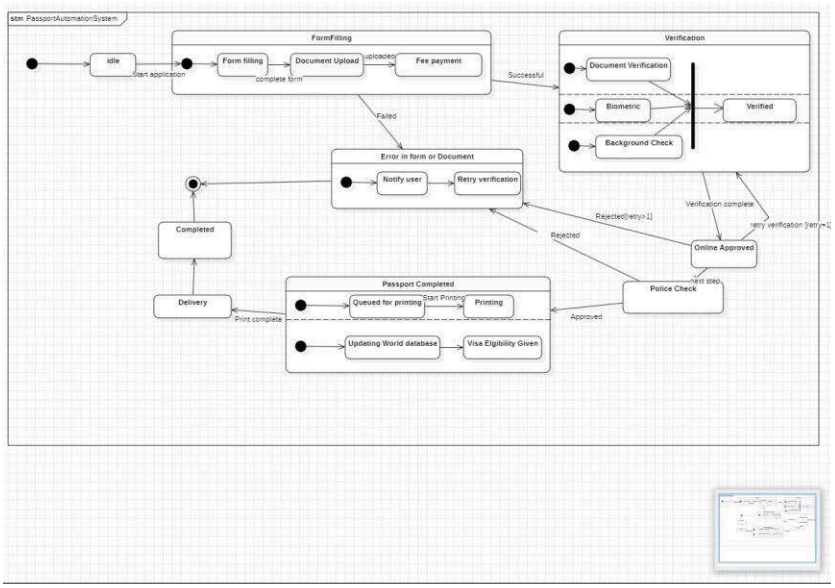
Budget

- Development: $120,000.
- Testing and QA: $30,000.
- Deployment: $20,000.
- Total: $170,000.

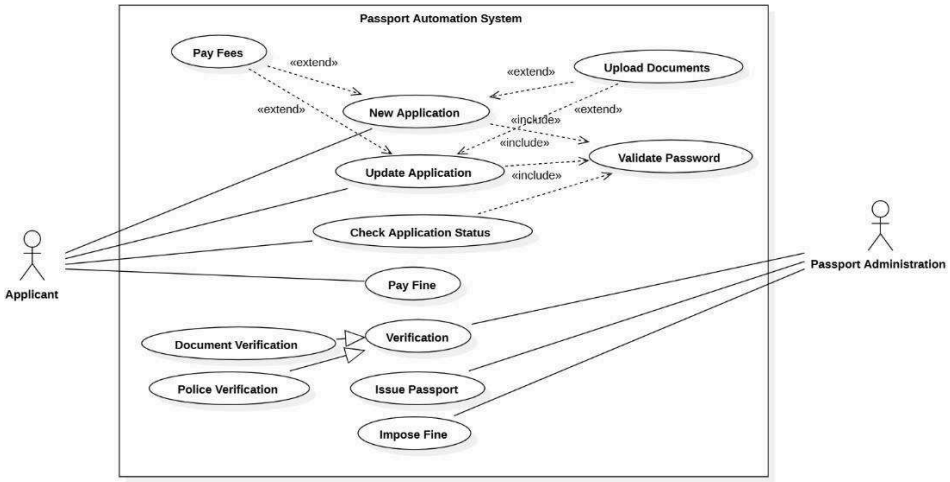
5.1 Class Diagram:



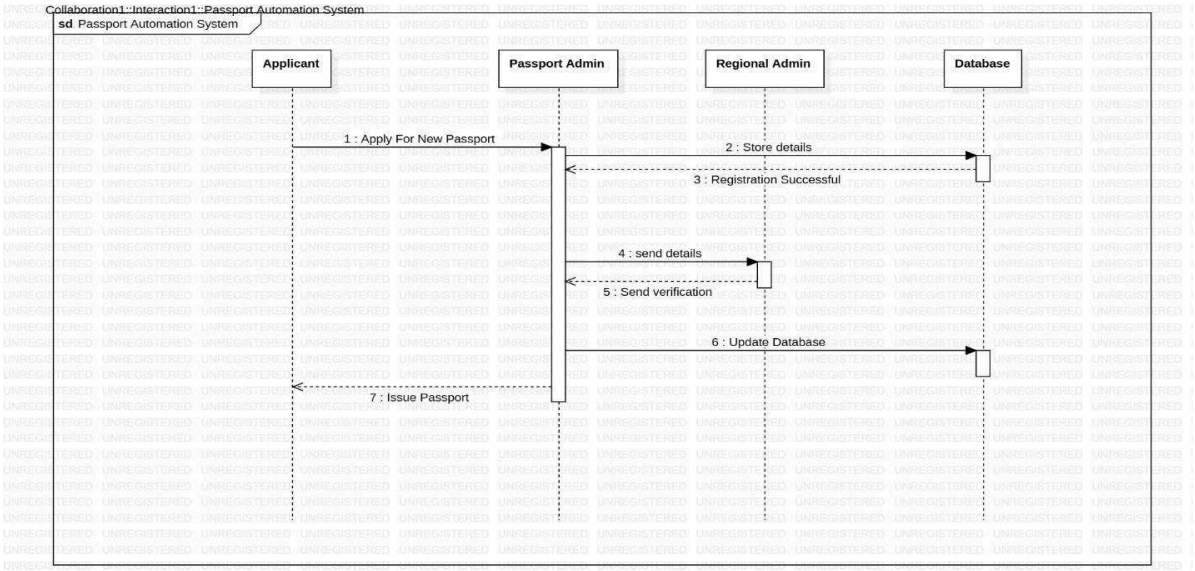
5.3 State Diagram:



5.4 Use Case Diagram:



5.5 Sequence Diagram:



5.6 Activity Diagram: