B.M.S. COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

Lab Record

Object-Oriented Modeling – 23CS5PCOOM

Submitted in partial fulfillment for the 5th Semester Laboratory

Bachelor of
Engineering in
Computer Science and Engineering

Submitted by:

Harshitha H G

1BM23CS108

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
August 2025-December 2025

B.M.S. COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Object-Oriented Modeling(23CS5PCOOM) laboratory has been carried out by Harshitha H G (1BM23CS108) during the 5th Semester August 2025-December 2025

.

Signature of the Faculty Incharge:
Dr Roopashree S
Associate Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

Table of Contents

1. Hotel Management System
2. Credit Card Processing
3. Library Management System
4. Stock Maintenance System
5. Passport Automation System

1.Hotel Management System

Problem Statement

Hotels face challenges in managing reservations, guest check-in/check-out, room availability, billing, and staff operations due to manual or outdated processes. These inefficiencies lead to booking errors, reduced guest satisfaction, and operational delays. A centralized, automated Hotel Management System is required to streamline operations, improve accuracy, and enhance customer service.

Software Requirements Specification (SRS)

Lab-1

Bafna Gold
Date: 19/8/25

1) Software Requirements Specification (SRS) for Hotel - Management System :-

1. Introduction :-

1.1 Purpose

This document specifies the requirements for the Hotel Management System (HMS), which automates the booking, check-in/check-out, billing, and management processes of a hotel. It is intended for developers, testers, and stakeholders to understand the system functionalities and constraints.

1.2 Scope

HMS will manage reservations, customer profiles, room assignments, billing and generate reports to improve operational efficiency in hotel environment. The system supports both online and on-site booking.

1.3 Document Conventions

→ Shall : Indicates requirement that is mandatory

→ Should : Indicates requirement that is recommended but not mandatory.

→ May/Can : Indicates optional features/behaviours

→ Requirement ID's (eg., FR1, NFR1) are used to reference specific requirements

1.4 Intended Audience
→ Developers: To implement specified functionality
→ Testers / QA: To validate system features against requirements.
→ Project Managers: To manage timelines, scope and deliverables
→ Hotel staff & Admins: For operational understanding.
→ Clients / Stakeholders: For approval & clarity on system behaviour.

1.4 References.
→ IEEE standard for Software requirements Specifications, IEEE Std 830-1998
→ PCI DSS standards for payment security
→ GDPR Compliance guidelines.

2. Overall Description

2.1 Product Perspective
HMS is a standalone system with a database backend and user interface accessible via web and desktop applications.

2.2 Product functions.
→ Room availability search
→ Booking and cancellation
→ Customer management
→ Check-in / Check-out processing
→ Billing and invoice generation.
→ Staff and payroll management

2.3 User characteristics
→ Customers: Basic computer / mobile knowledge
→ Hotel staff: Moderate IT skills
→ Admin / Managers: High-level access and management responsibility.

2.4 Constraints
HMS must support both web and mobile platforms. Database must be secure and reliable and it should be available 24/7. Data privacy and accounting rules should be maintained.

2.5 Assumptions and Dependencies.
Users have stable internet connections. Payment gateways are available and reliable. Hotel should provide accurate input data.

3. Specific Requirements
3.1 Functional Requirements.
FR1: System shall allow customers to search and book available rooms.
FR2: System shall generate unique booking IDs for reservations.
FR3: It should allow staff to check-in and check-out customers
FR4: It should generate invoices with tax calculation
FR5: It should maintain records of staff schedules and payroll by providing daily, weekly, monthly reports.
FR7: System shall allow administrators to manage rooms, prices and discounts

3.2 Non functional Requirements.

Performance : System shall handle atleast 500 concurrent users

Reliability : System should provide 99.5% uptime.

Security : Data can be encrypted using AES-256

Usability : It should be simple and intuitive

Scalability : It shall support future hotel chain integration.

3.3 External Interface Requirements.

User-Interface : web (HTML, CSS, React/Angular) and mobile (Flutter / React Native)

Hardware Interfaces : Standard desktop, laptops, mobile devices.

Software Interfaces : Integration with payment gateway API's.

Communication Interface : HTTPS, REST APIs

4. Appendices

4.1 Glossary :

A Booking ID is a unique-number for each reservation, while check-in/check-out record guest arrivals & departures. An invoice summarizes charges and taxes, & a payment Gateway handles secure online payments as key System users.

4.2 Future Enhancements

System can be upgraded with AI for dynamic pricing & chatbot-based customer support. It also include IOT smart room controls, loyalty programs & hotel chain integration.

Class Diagram:

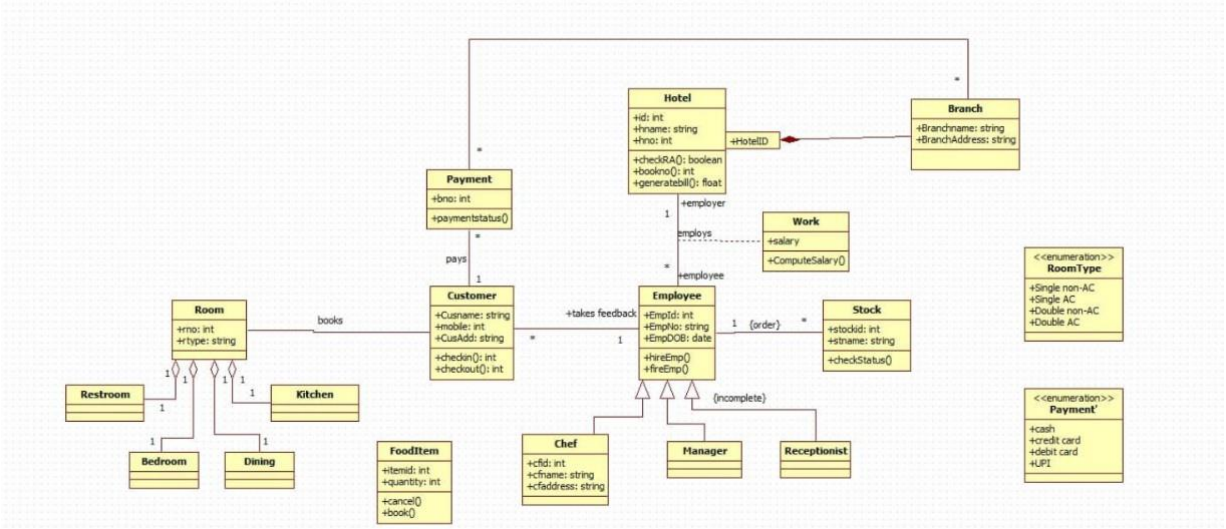

Fig 1.1 Class diagram of Hotel Management System

The diagram shows a Hotel Management System where a hotel has multiple branches, rooms, customers, employees, and payments. Customers book rooms and make payments. Employees (like chefs, managers, and receptionists) work for the hotel and handle tasks such as food orders and customer service. Rooms have different types and facilities, while stock and food items are also managed within the system.

State Diagram:

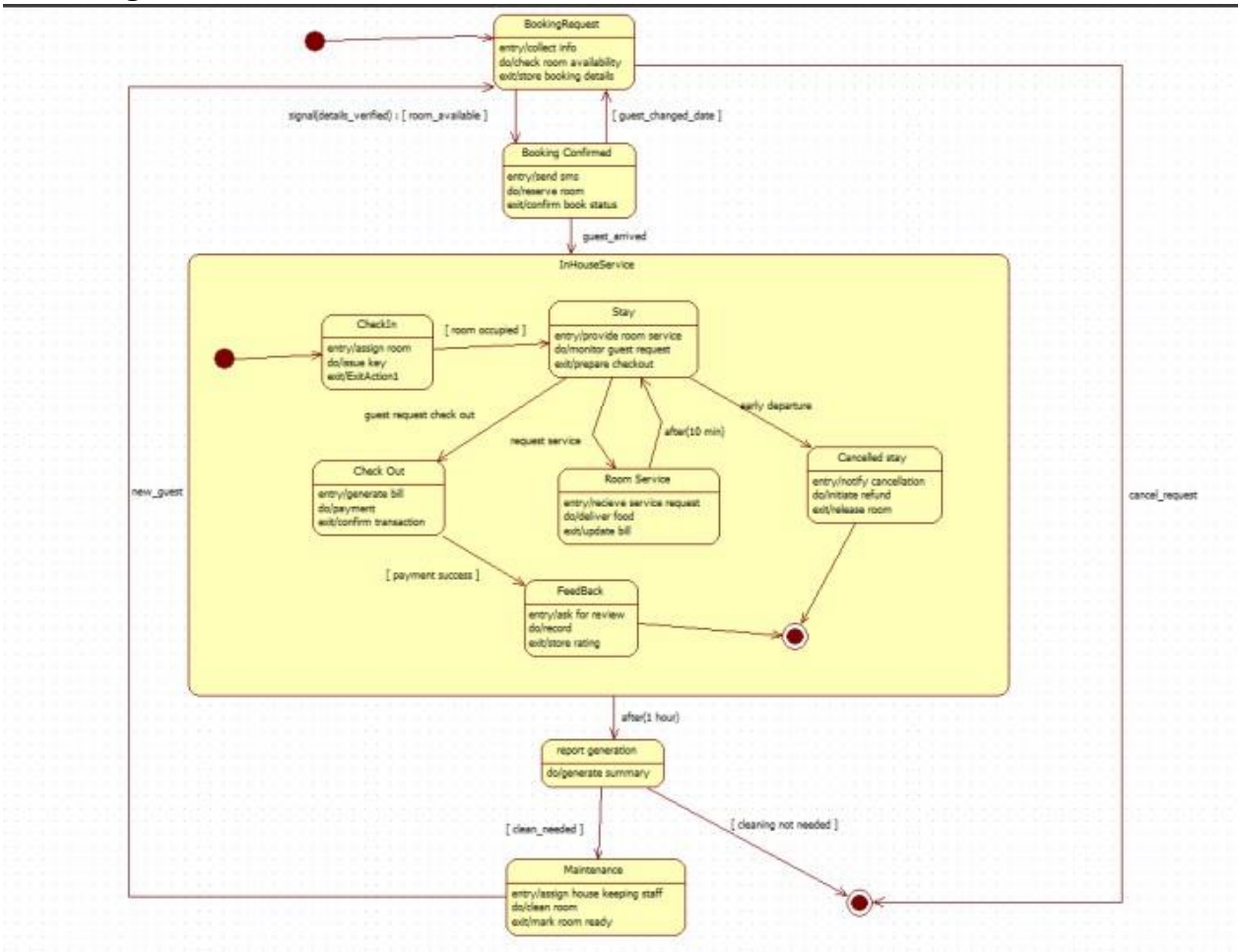

Fig 1.2 State diagram of Hotel Management System

The state diagram shows the lifecycle of a hotel booking and stay. It begins with a Booking Request, which becomes Booking Confirmed if a room is available. When the guest arrives, they move through states such as Check-in, Stay, requesting Room Service, giving Feedback, and finally Check-out. A guest may also enter a Cancelled Stay if the booking is canceled. After checkout, the system generates reports, and rooms may go into Maintenance if cleaning is needed.

Use-Case Diagram:

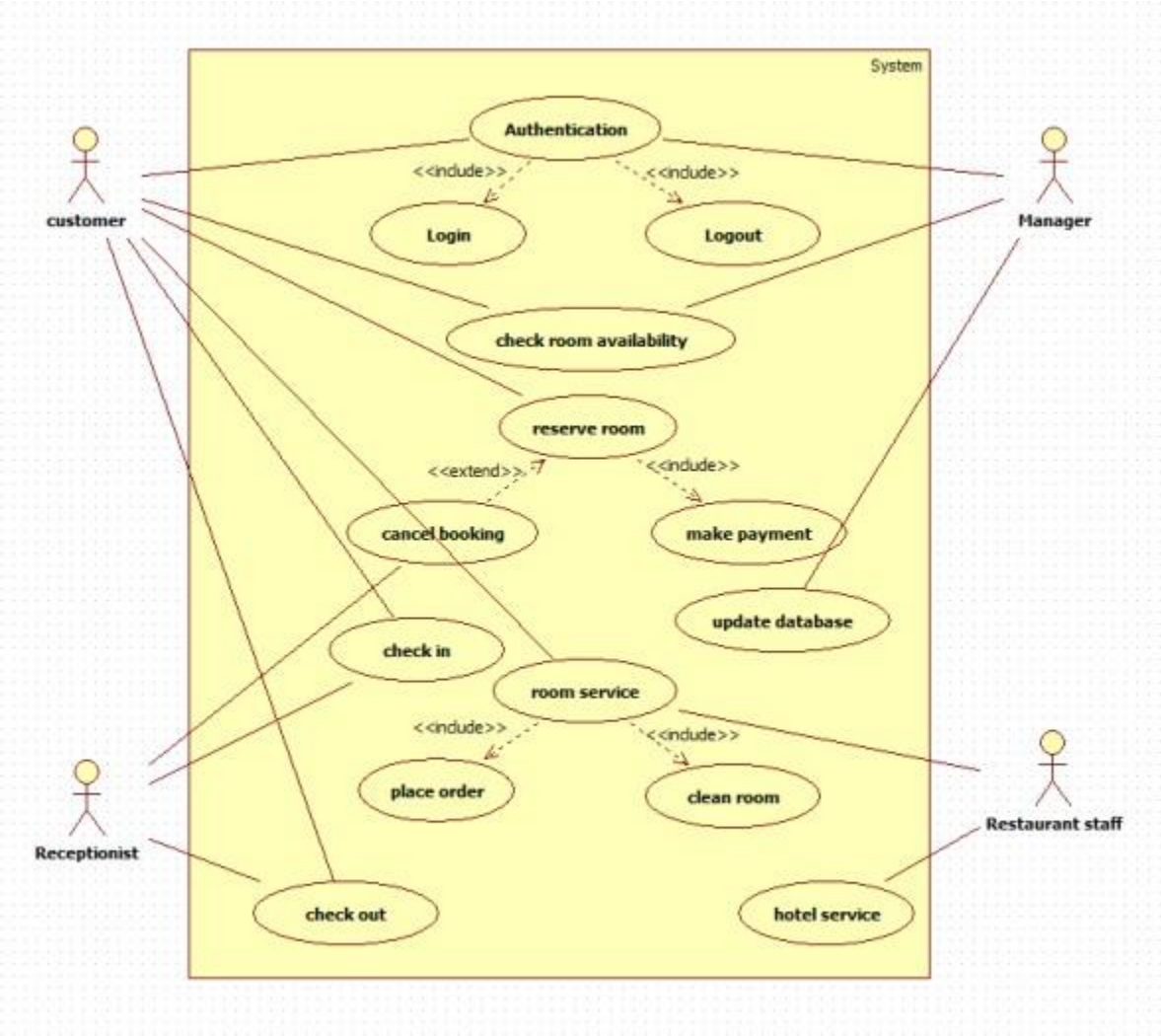

Fig 1.3 Use case diagram of Hotel Management System

The use-case diagram shows how different users interact with the Hotel Management System. Customers can log in, check room availability, reserve or cancel bookings, make payments, check in, request room service, and check out. Receptionists handle check-ins, check-outs, and room service tasks like placing orders or cleaning rooms. Managers can authenticate, update the database, and manage reservations. Restaurant staff provide hotel and food service support. The system centralizes all these actions to streamline hotel operations

Sequence Diagram:

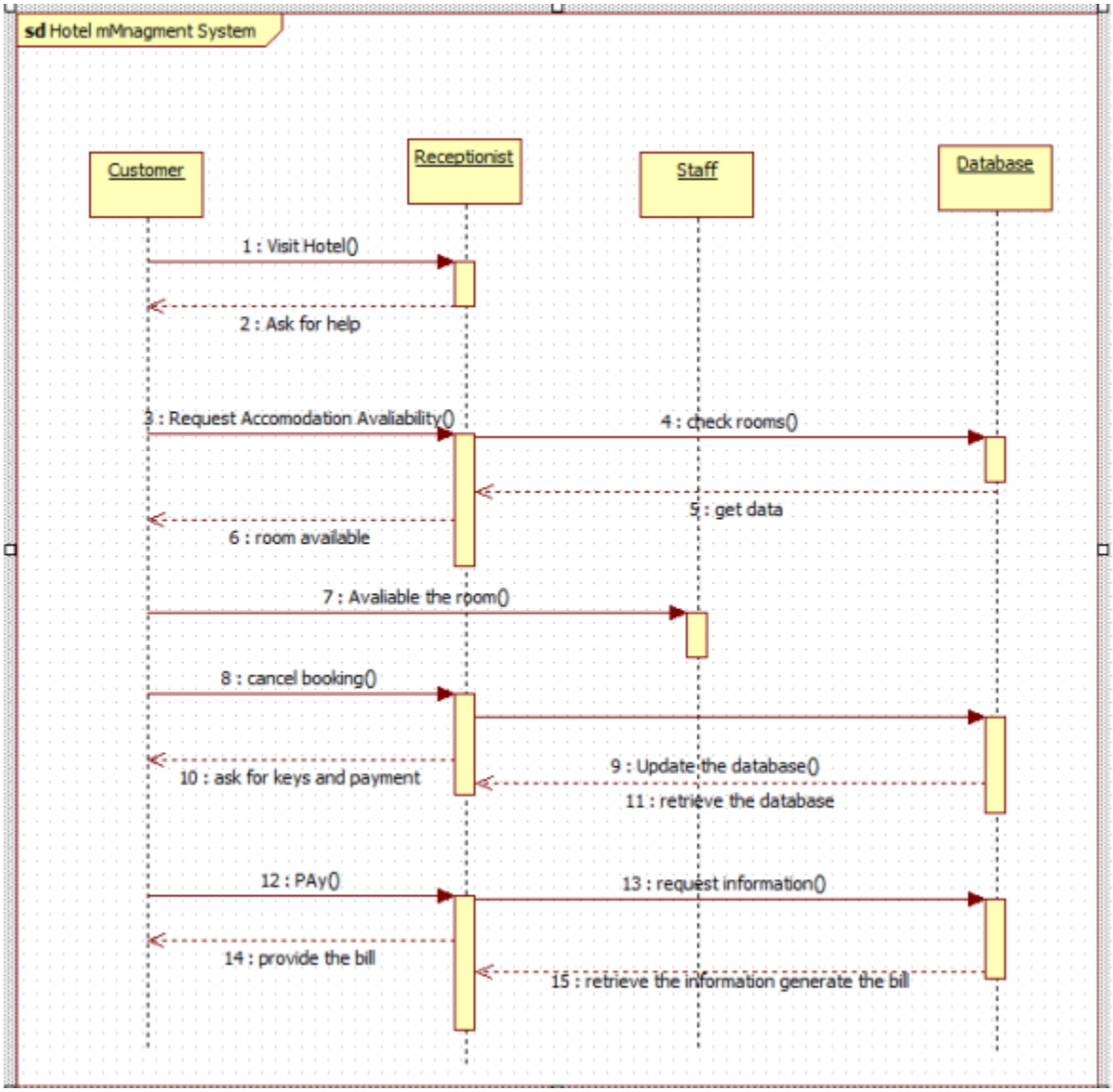

Fig 1.4 Sequence diagram of Hotel Management System

The sequence diagram shows the interaction between the customer, receptionist, staff, and database during a hotel booking process. The customer visits the hotel, requests accommodation, and the receptionist checks room availability through the staff and database. Once availability is confirmed, the customer can proceed to book or cancel. The receptionist updates the database, collects payment, and requests billing information. Finally, the receptionist provides the bill to the customer.

Activity diagram:

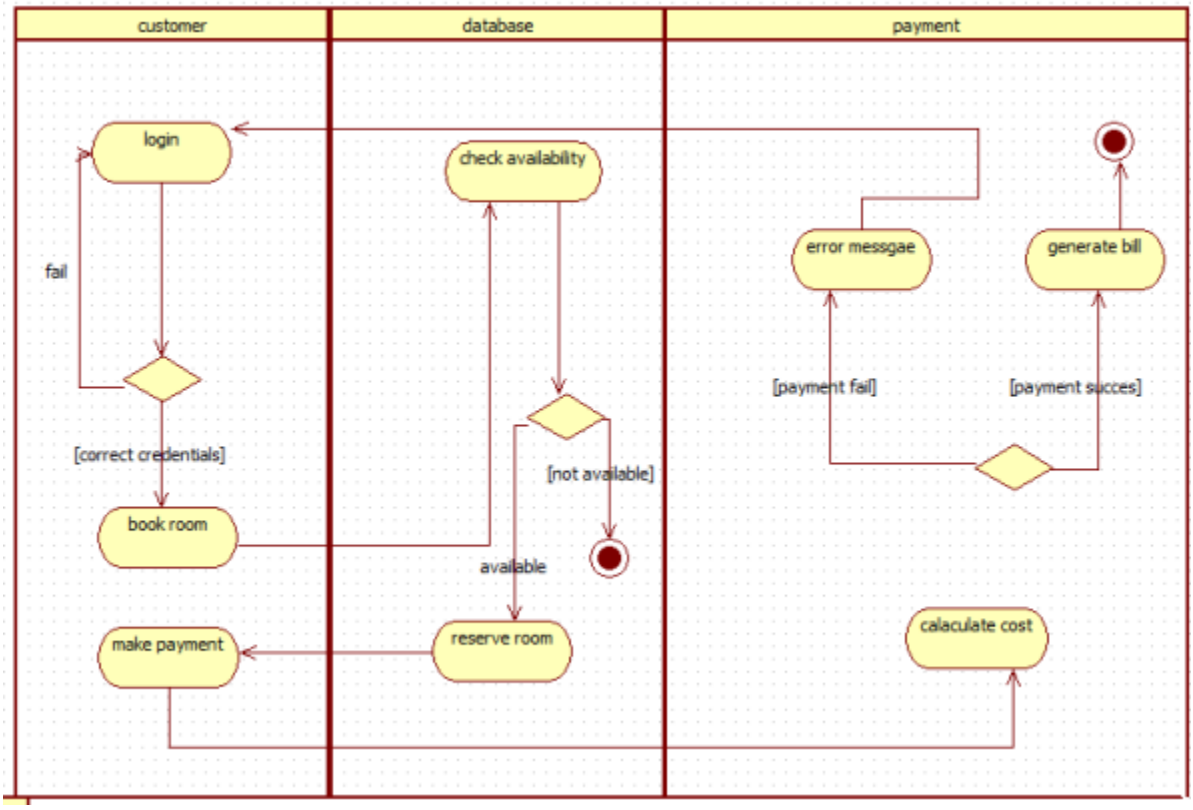

Fig 1.5 Activity diagram of Hotel Management System

The activity diagram shows the steps a customer follows to book a room in the hotel system. The customer logs in, and the database checks room availability. If a room is available, the customer proceeds to book and make a payment. The payment system calculates the cost and either generates the bill for a successful payment or shows an error message if the payment fails.

2. Credit Card Automation System

Problem Statement

Manual credit card processing—including application approval, transaction handling, billing, payments, and fraud detection—leads to delays, errors, and security risks. As customers and transactions increase, managing these tasks becomes more difficult for banks.

A Credit Card Automation System is required to automate account creation, real-time transaction processing, billing cycles, payment updates, fraud alerts, and reporting. The goal is to improve speed, accuracy, security, and overall efficiency through a centralized, secure, and user-friendly platform.

Software Requirements Specification (SRS)

Bafna Gold
Date: Page:

2) SRS for credit card processing:-

1. Introduction

1.1 Purpose

Purpose of this document is to define requirements for Credit card Processing system (CCPS). It is designed to securely authorize, process and settle credit card transactions between customers merchants & banks. This SRS will serve as a reference to stakeholders to ensure system meets security, reliability and performance expectations.

1.2 Scope

CCPS will handle the complete life cycle of credit card transactions, including authorization, authentication, transaction, logging, fraud detection & settlement. Banks and payment network will interact with the system to verify funds & complete settlements. The system aims to reduce transaction costs ensure regulatory compliance and provide high-speed, secure and reliable payment processing.

1.3 Document Conventions

This document follows IEEE 830 standard for SRS. Functional requirements are described in clear, structured paragraphs while non-functional requirements are expressed in plain language by implementing explanation for all technical terms.

1.4 Intended Audience

It includes Software developers, testers, project managers and system architects involved in development of Credit card Processing System. Additionally it is meant for banking officials, compliance officers and stakeholders who require an understanding of System's functionality & regulatory aspect.

1.5 References

→ IEEE Std 830-1998: IEEE recommended practice for software requirements specifications.
→ PCI DSS compliance standards
→ ISO 8583 standard for financial Transaction Messaging.

2. Overall Description

2.1 Product Perspective

Credit card Processing System will act as an intermediary between merchants, customers, issuing banks, and payment networks. It will integrate with Point of Sale [POS] terminals, online payment, gateways and banking system.

2.2 Product Functions

It will authorize transactions by validating card details, check balances with issuing banks and approve & decline requests. It will log each transaction, detect suspicious activity using fraud detection algorithms and handle settlements between bank & merchants.

2.3 User characteristics

Customers are expected to have minimal technical knowledge and simply use their credit cards for purchases. Merchants will use POS systems or online portals to initiate transactions. Bank administrators and auditors will use back-end tools for monitoring and compliance checks.

2.4 Constraints

System must comply with PCI-DSS standards to protect sensitive cardholder data. It must process transactions within 2-5 seconds and be available 24/7. Strong encryption is mandatory for data transmission.

2.5 Assumptions and Dependencies

The System assumes that customers provide valid card details and that payment networks are available. It also depends on reliable internet connectivity and bank services for processing requests.

3. Specific Requirements

3.1 Functional Requirements.

→ System must validate card details before processing. It shall also authorize & decline transactions based on available funds & fraud detection checks

→ System shall generate unique Transaction ID for each request.

→ It shall log every transaction in a secure database for auditing

→ It sends transaction responses back to merchant within 2-5 seconds.

→ System shall provide an API interface for integration with POS terminals and online payment gateways.

→ It shall allow administrators to monitor transactions, generate reports, and review suspicious activities.

3.2 Non-Functional Requirements

Security: System must comply with PCI DSS standards and use AES-256 encryption.

Performance: It must process upto 10,000 concurrent transactions per second.

Reliability: It provide 99.9 uptime & ensure disaster recovery within 1 hour of failure.

Scalability: Support future growth in transaction volume & merchant base.

Usability: Interface must be user-friendly with clear error-message for customers & merchants.

Maintainability: System must allow easy updates without disrupting ongoing operations.

3.3 External Interface Requirements:

System will provide APIs for integration with merchant applications, POS devices, and online payment gateways. It will support HTTPS communication & work with standard databases for transaction logging.

5. Appendices

5.1 Glossary

A Transaction ID is unique identifier generated for each credit card transaction. Authorization is process of verifying card details & funds availability, while Settlement refers to transferring funds from customer accounts to merchant accounts.

5.2 Future Enhancements

Future improvements may include biometric authentication for higher security, AI driven fraud detection systems, blockchain-based settlements for transparency & support for multi-currency international transactions.

Class Diagram:

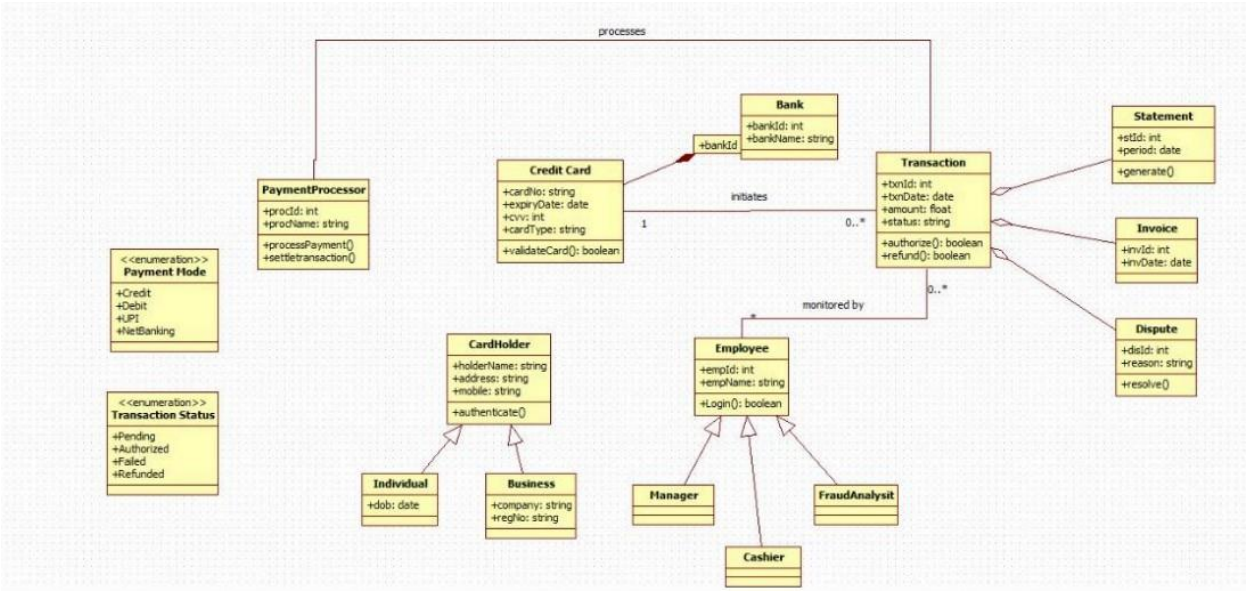

Fig 2.1 Class diagram of Credit Card Processing System

The class diagram represents a payment processing system that manages credit card transactions. A PaymentProcessor handles payment modes (like credit, debit, UPI, net banking) and interacts with CreditCard details, which are validated before initiating a transaction. Each Transaction is linked to a Bank and can generate related documents such as Statements, Invoices, or Disputes. Transactions carry statuses like pending, authorized, failed, or refunded.

A CardHolder (either an individual or business) owns the credit card and can authenticate themselves. Employees—including managers, cashiers, and fraud analysts—monitor transactions and log into the system. Overall, the diagram shows how payments flow from cardholder verification to bank interaction, transaction processing, and post-transaction documentation.

State diagram:

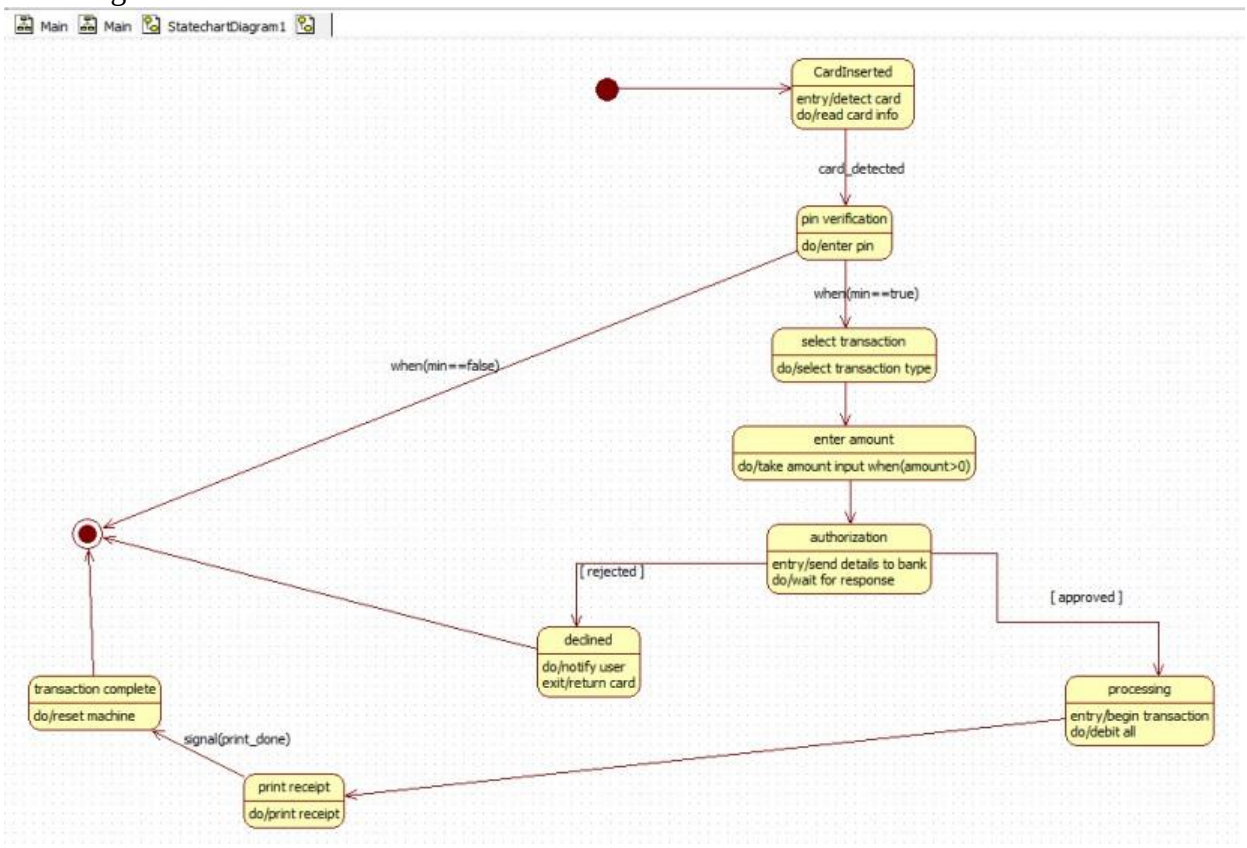

Fig 2.2 State diagram of Credit Card Processing System

The state diagram illustrates the workflow of a card-based transaction process in a payment machine (like an ATM or POS). The process begins when a card is inserted and detected, followed by PIN verification. If the PIN is valid, the user selects a transaction type and enters an amount. The machine then sends the transaction details to the bank for authorization.

If the bank approves the request, the system proceeds to process the transaction and later prints a receipt. If the request is rejected, the machine notifies the user and returns the card. In both scenarios, the process ends with the machine resetting itself and returning to the initial state, ready for the next transaction.

Use-case diagram:

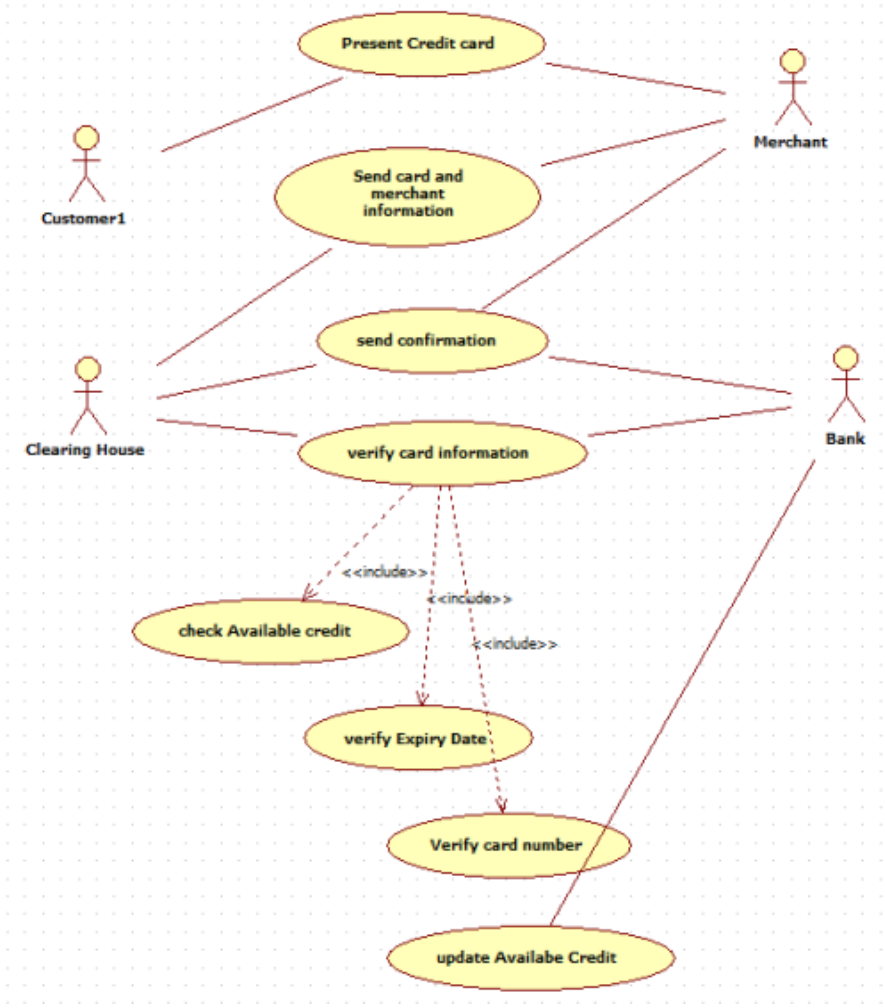

Fig 2.3 Use-case diagram of Credit Card Processing System

The use case diagram illustrates the overall credit card processing workflow involving the Customer, Merchant, Clearing House, and Bank. The process begins when the customer presents their credit card to the merchant, who sends the card and merchant details for verification. The Clearing House and Bank validate the card through a series of checks, including verifying the card number, expiry date, and available credit.After successful verification, the system updates the available credit and sends a confirmation back to the merchant. This enables the merchant to complete the transaction. The diagram highlights the interaction between different actors and the essential verification steps involved in processing a credit card payment.

Sequence diagram:

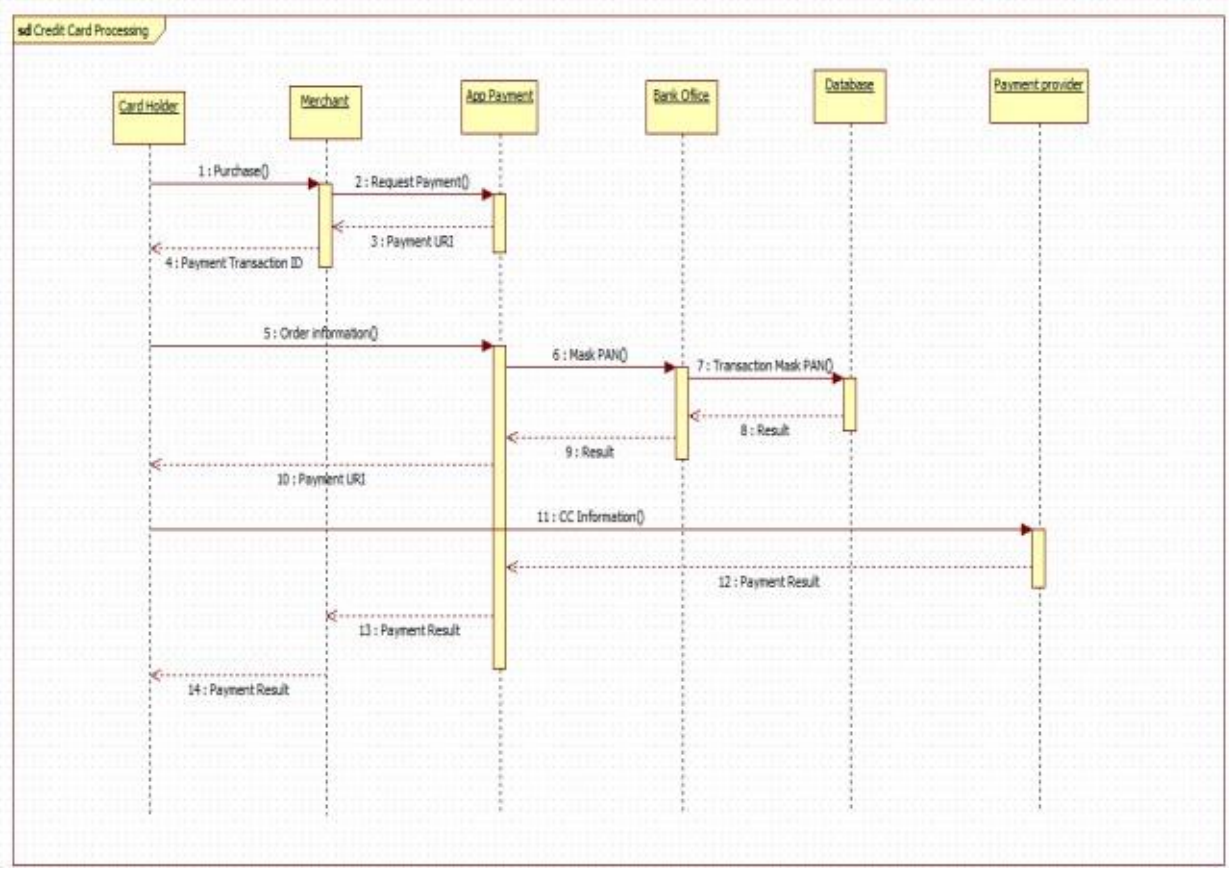

Fig 2.4 Sequence diagram of Credit Card Processing System

The sequence diagram shows the flow of a credit card payment from purchase to completion. The Card Holder initiates a payment, the Merchant sends the request, and the App Payment system processes the order by masking card details and communicating with the Bank Office and Database for verification. After validation, the payment result is returned through the App Payment system to the Merchant, Payment Provider, and finally to the Card Holder.

Activity diagram:

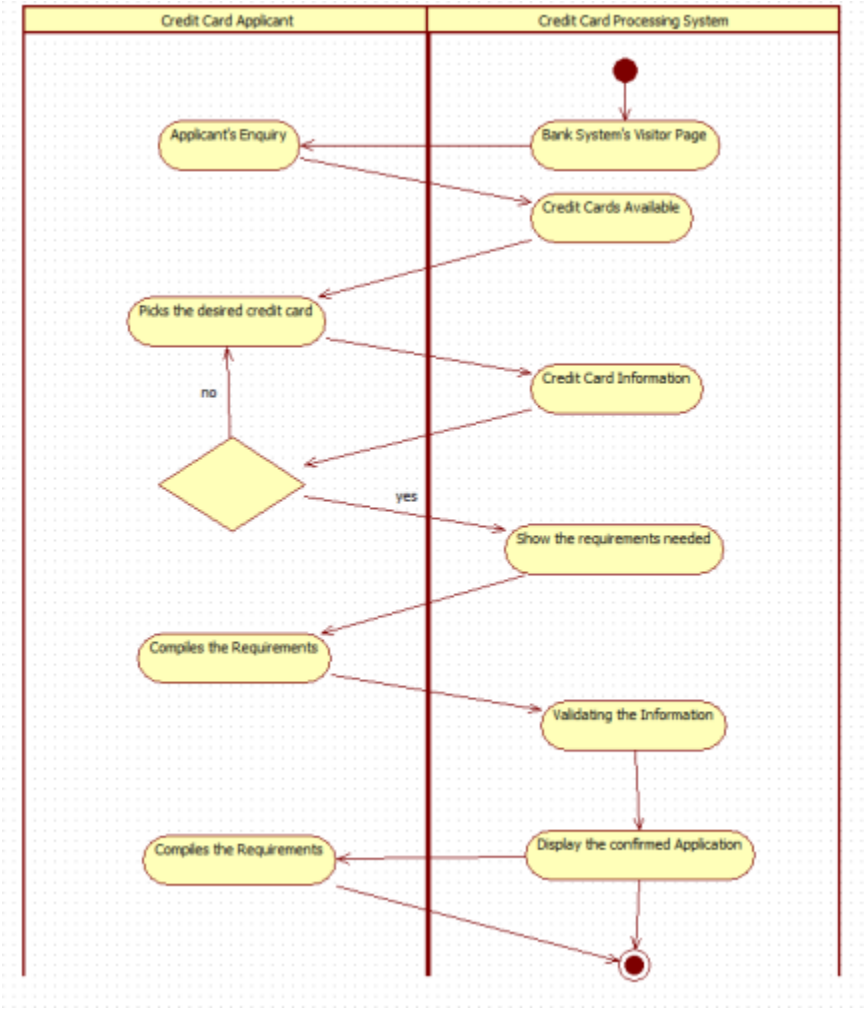

Fig 2.5 Activity diagram of Credit Card Processing System

The activity diagram illustrates the process of a credit card applicant applying through the bank's credit card processing system. The applicant sends an enquiry, views available credit cards, and selects one. The system displays the card information and required documents. After the applicant submits the necessary requirements, the system validates the information and finally displays the confirmed application.

3. Library Management System

Problem Statement

Traditional library operations such as book issuing, returning, cataloging, fine calculation, and record maintenance are often done manually, leading to errors, delays, and difficulty in tracking books and users. As the number of books and members increases, managing inventory, reducing book loss, and maintaining accurate records becomes challenging.

A Library Management System is needed to automate book management, user registration, transactions, and report generation. The goal is to improve efficiency, accuracy, and accessibility for librarians and users. The system should provide quick search, real-time inventory updates, and role-based access while ensuring secure data handling.

Software Requirements Specification (SRS)

2/09/2025

Lab-2

Bafna Gold
Date: Page:

3) SRS for library Management System:

1. Introduction:

1.1 Purpose: The purpose of this document is to define the requirements for the library Management System. It ensures proper handling of books, members, borrowing/returning, fines and catalog search.

1.2 Scope: The system will automate library operations including book issue, return, catalog search, fine calculation, and membership management.

1.3 Overview: LMS will replace manual record-keeping with a digital system enabling librarians and students to access resources efficiently.

2 General Description.

The system supports librarians, staff and members, offering catalog browsing, issue/return tracking, overdue management, and report generation.

3. Functional Requirements

Book Management: Add, update, and delete books. Maintain availability status.

Member Management: Maintain member records, registration and borrowing limits.

Issue/Return System: Track borrowing/returning with due dates.

Fine Calculation: Automatically calculate overdue fines.

Search & Catalog: Search by title, author ISBN & category.

Reports: Generate reports on issued books overdue items, and member activity.

4. Interface Requirements

UI: Simple web-based and mobile interface

Integration: Barcode/RFID Support for book issue/return.

5. Performance Requirements: Response time ≤ 2 seconds. Handle up to 500 concurrent users. Ensure real-time inventory update

6. Design Constraints: Relational DB (MySQL/PostgreSQL). Web-based architecture with Java/Python backend.

7. Non-Functional Requirements

Security: Role-based access (admin, librarian, student)

Class diagram:

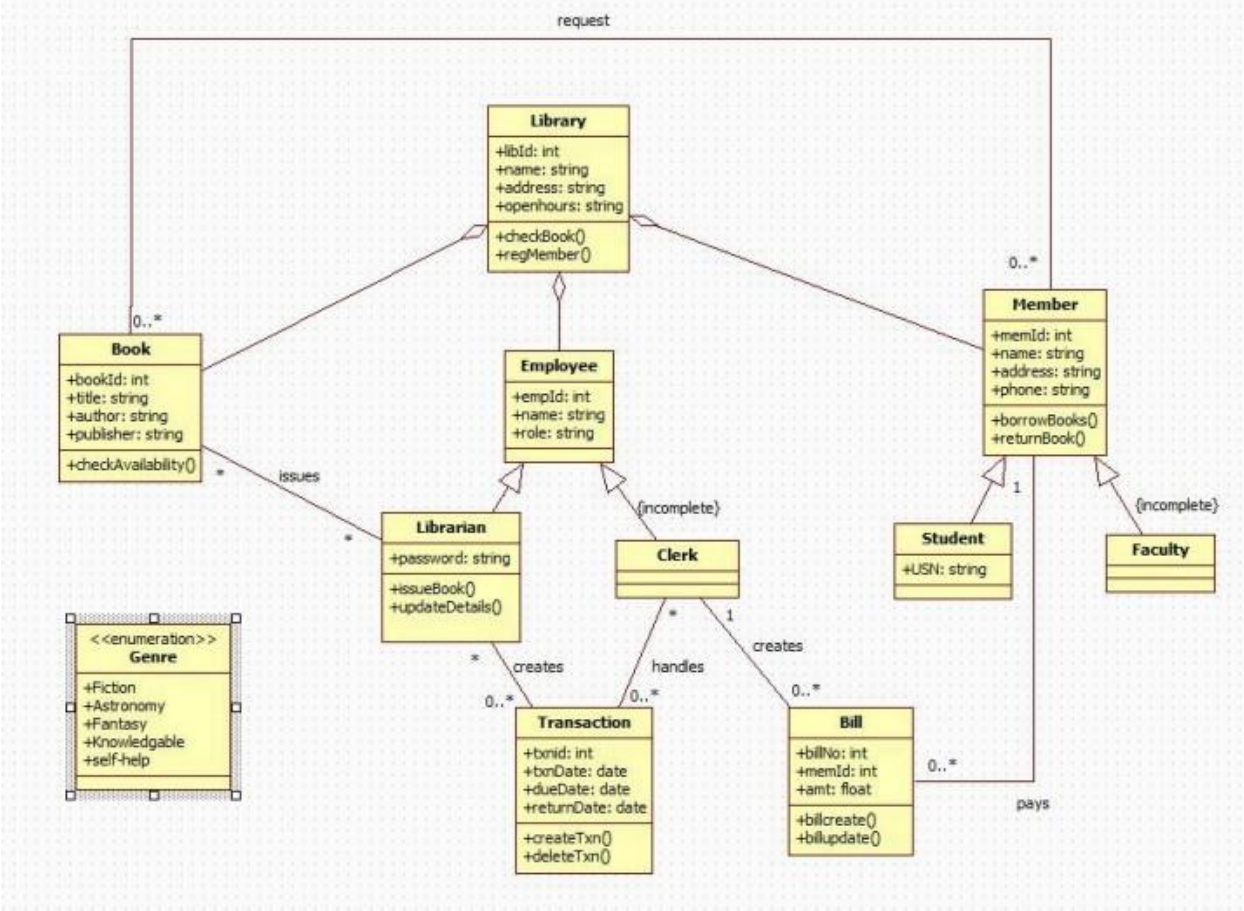

Fig 3.1 Class diagram of Library management system

The class diagram represents a library management system involving books, members, employees, and transactions. The Library maintains books and registered members. Members (students or faculty) can borrow and return books, while employees (librarians and clerks) manage book issuing, record updates, and transaction handling. Each transaction records borrowing and return dates, and bills are generated for any dues. The diagram also includes book genres and shows how different roles interact to support the overall library operations.

State diagram:

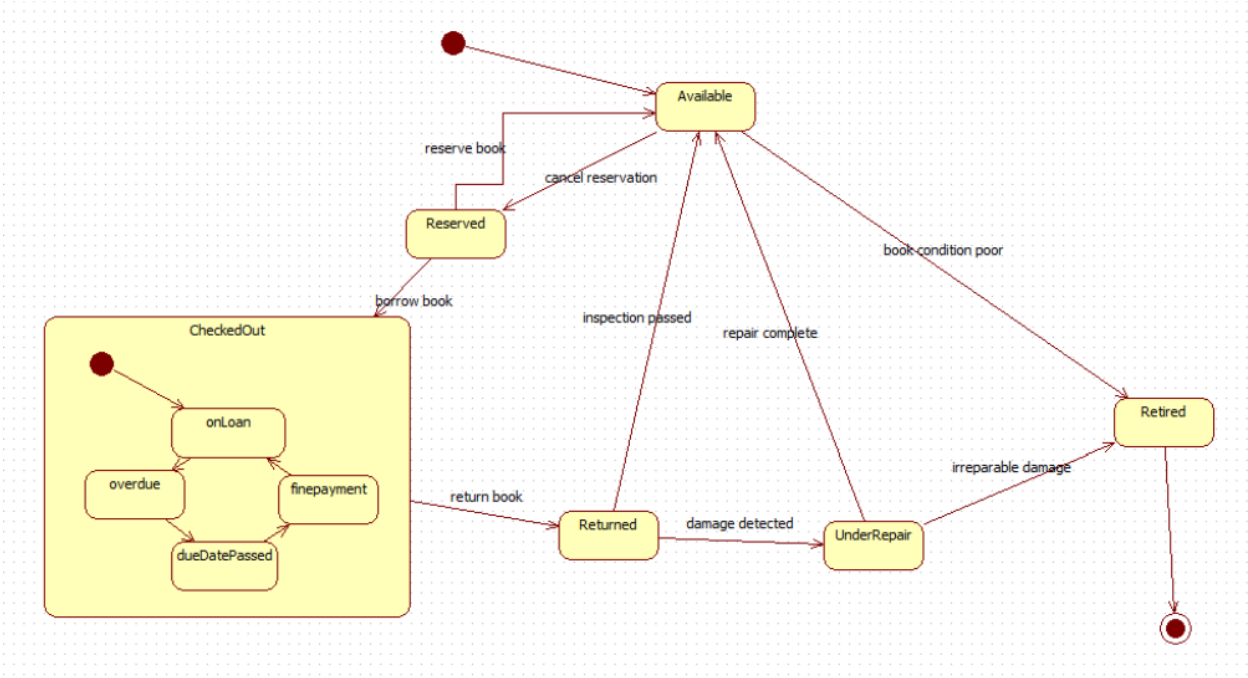

Fig 3.2 State diagram of Library management system

The state diagram shows the lifecycle of a library book. A book starts in the Available state and can be Reserved or Checked Out. When checked out, it may become overdue or require fine payment before return. Once returned, the book is inspected—if damaged, it goes Under Repair; if repair is completed, it becomes available again. If the book is irreparably damaged or in poor condition, it transitions to the Retired state and is removed from circulation.

Use-case diagram:

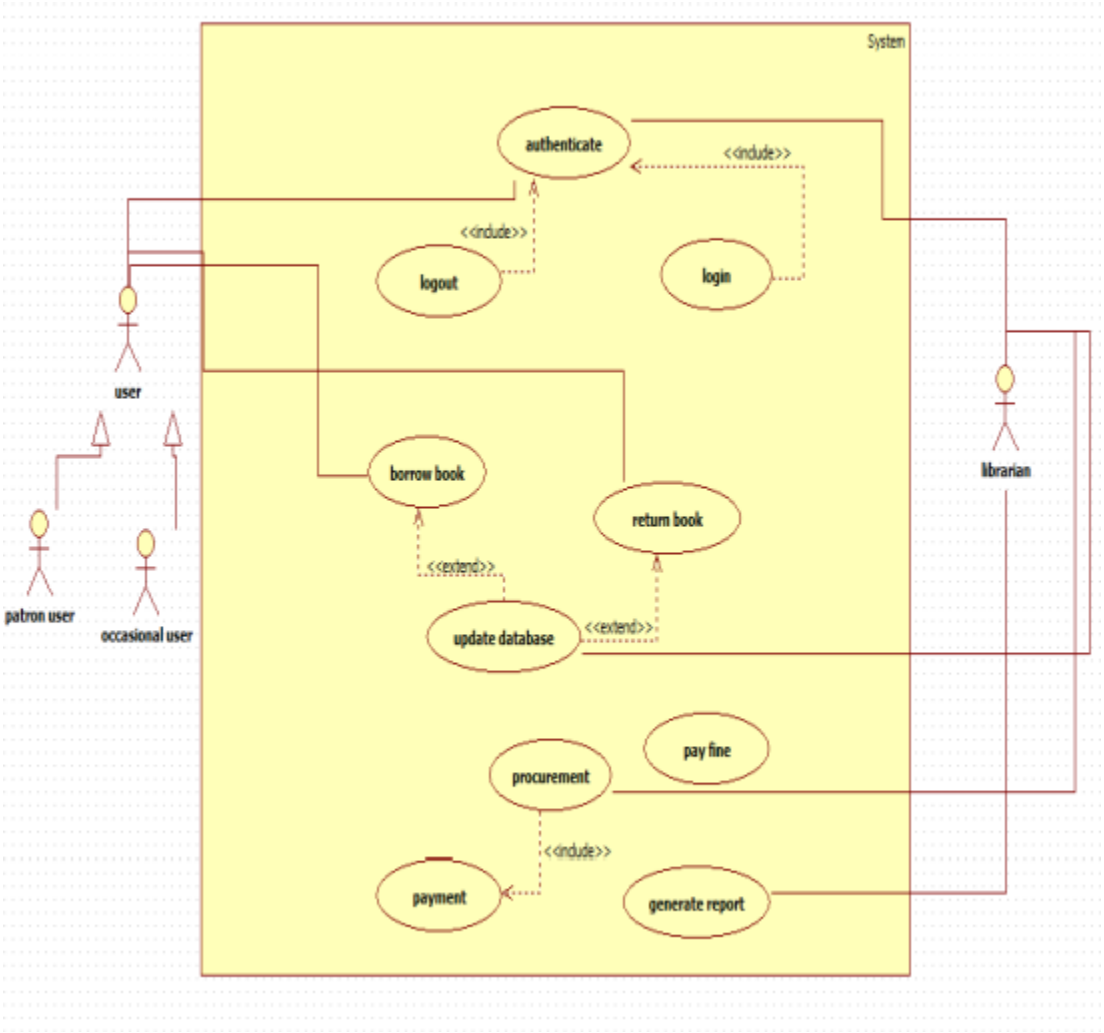

Fig 3.3 Use case diagram of Library management system

The use case diagram represents a library management system where users (patron and occasional users) and librarians interact with various system functions. Users can authenticate, log in, borrow books, return books, pay fines, and make payments. Librarians manage procurement, generate reports, and update the database. Several use cases include or extend others, showing how core actions like authentication and data updates support the overall library operations.

Sequence diagram:

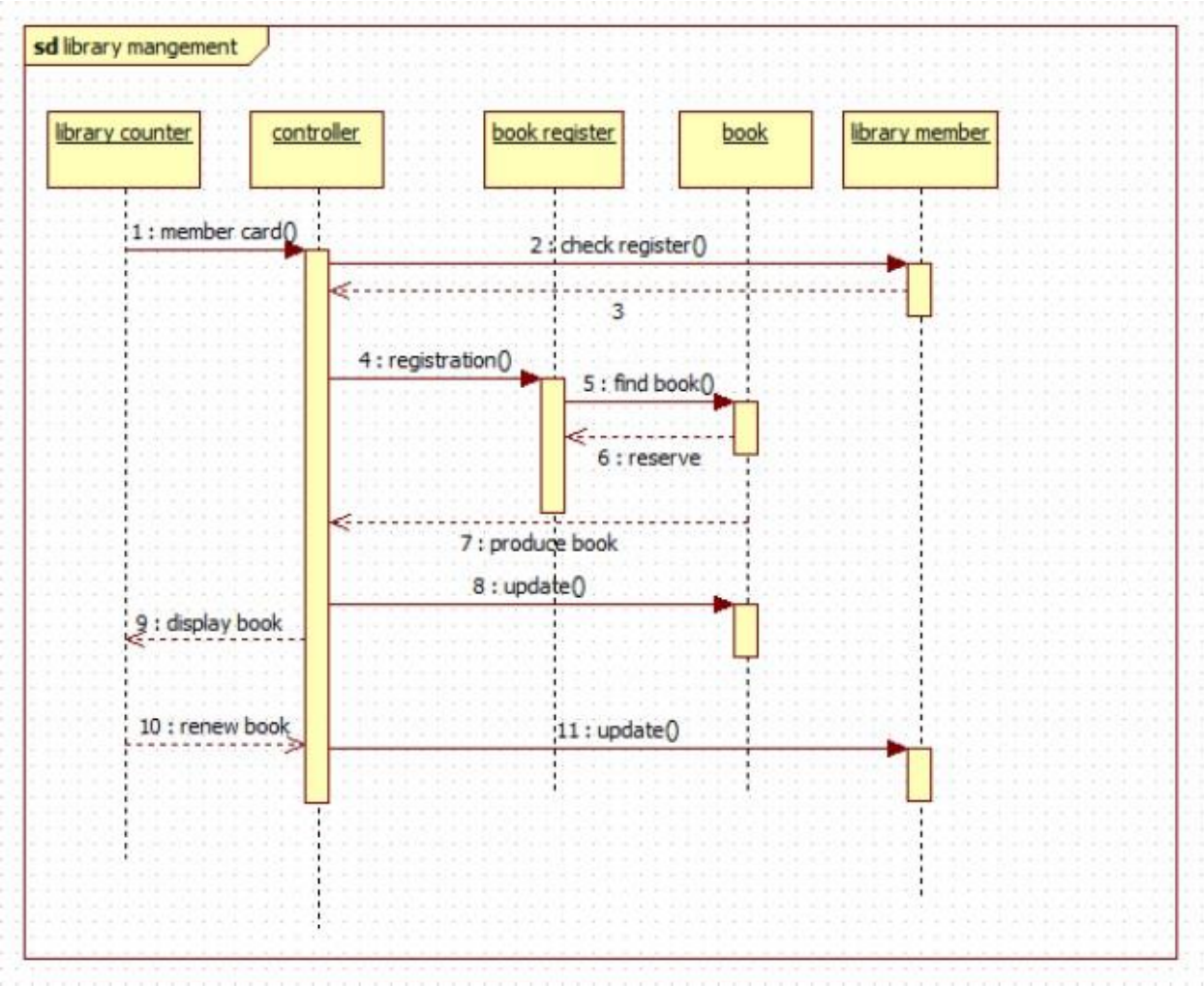

Fig 3.4 Sequence diagram of Library management system

The sequence diagram shows the process of managing book transactions in a library. A library member presents their card at the library counter, which triggers the controller to check the book register, register the member, and search for the requested book. Once the book is found and reserved, it is produced for the member. The system updates the book's status, displays it to the member, and also handles book renewal requests, updating the records accordingly.

Activity diagram:

Fig 3.5 Activity diagram of Library management system

The activity diagram outlines the process of issuing a book in a library. It begins with a user inquiring about a book and checking its availability. If available, the system validates the member; if not registered, the member is asked to register. The system then checks whether the member has exceeded the maximum number of issued books. If eligible, the book is issued, the member's issue details are added, and the book issue record is updated. Otherwise, the book is not issued.

4. Stock Maintenance System

Problem Statement

Many businesses still rely on manual methods to record and track stock levels, purchases, and sales. This often leads to errors, mismatched inventory counts, delays in restocking, and difficulty in monitoring product movement. As stock increases, it becomes harder to maintain accurate records and prevent shortages or overstocking.

A Stock Maintenance System is needed to automate inventory tracking, update stock levels in real time, and generate reports on stock usage, shortages, and reorder requirements. The goal is to ensure accuracy, reduce manual errors, improve efficiency, and help businesses make better inventory decisions through a centralized and user-friendly system.

Software Requirements Specification (SRS)

Reliability: 99% uptime
Scalability: Expandable for large university libraries
Portability: Accessible via web & mobile

8. Preliminary Schedule & Budget
Development duration: 4 months.
Estimated budget: $ 50,000

4) SRS for Stock Maintenance System (SMS)

1. Introduction

1.1 Purpose: To define the requirements of an automated stock/inventory management system.

1.2 Scope: System tracks goods in/out, monitors inventory levels, generates stock alerts, and maintains supplies records.

1.3 Overview: SMS will streamline inventory management for warehouses, shops and enterprises.

2. General Description
Supports store managers, staff, and suppliers by maintaining stock records, purchase orders, and consumption reports.

3. Functional Requirements
Stock Management: Add, update, and delete stock items.
Supplier Management: Maintain supplier details and purchase history.
Inventory Tracking: Track stock in/out with dates.
Low-Stock Alerts: Notify users of critical stock levels.
Reports: Generate purchase, consumption and balance stock reports.

4. Interface Requirements
UI: Intuitive dashboard for stock visualization.
Integration: Barcode scanner integration.

5. Performance Requirements: Handle 800 concurrent users. Response time < 3 seconds. Real-time update of stock levels.

6. Design Constraints: Relational DB (MySQL/Oracle)
Backend: Java / Spring Boot or Node.js

7. Non-Functional Requirements
Security: Access control for managers & staff
Reliability: Backup and recovery system
Scalability: Support for multi-warehouse operation
Usability: Easy to operate with minimal training.

8. Preliminary Schedule & Budget
Development duration: 5 months.
Estimated budget: $ 75,000

Class diagram:

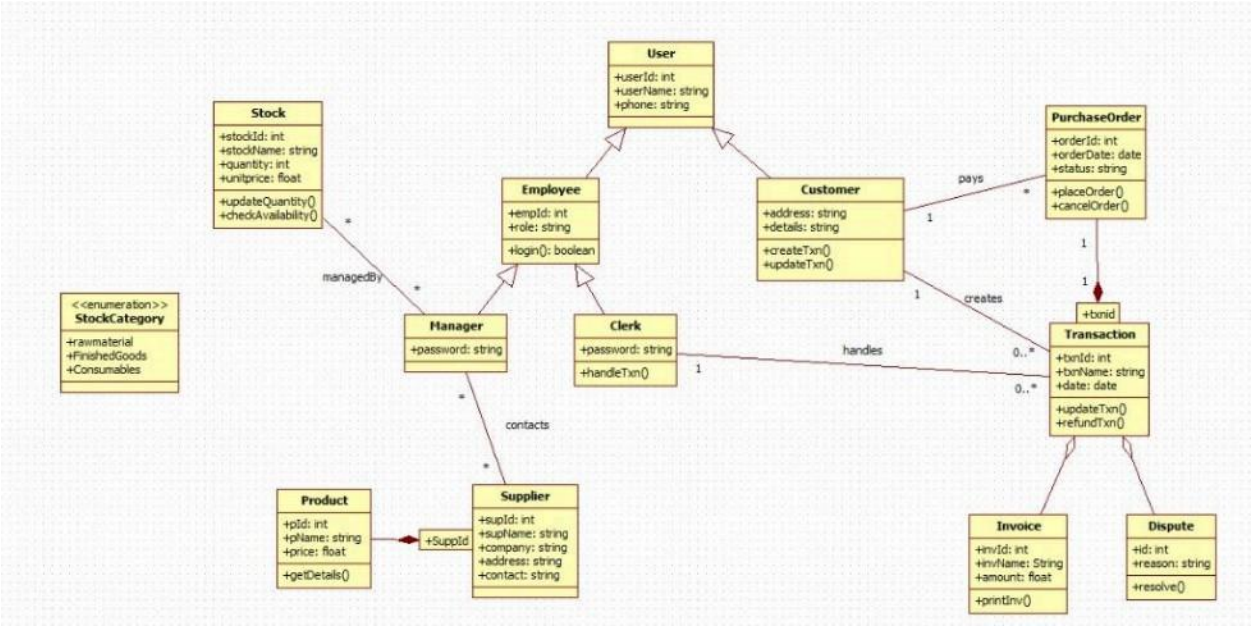

Fig 4.1 Class diagram of Stock management system

The class diagram represents an inventory and transaction management system involving users, employees, customers, suppliers, products, and stock. Customers create purchase orders, which generate transactions handled by clerks. Stock items belong to specific categories and are managed by managers who update quantity and availability. Suppliers provide products linked to stock, while invoices and disputes are associated with transactions for payment and issue resolution. Overall, the diagram shows how different entities interact to support purchasing, stock management, and transaction processing.

State diagram:

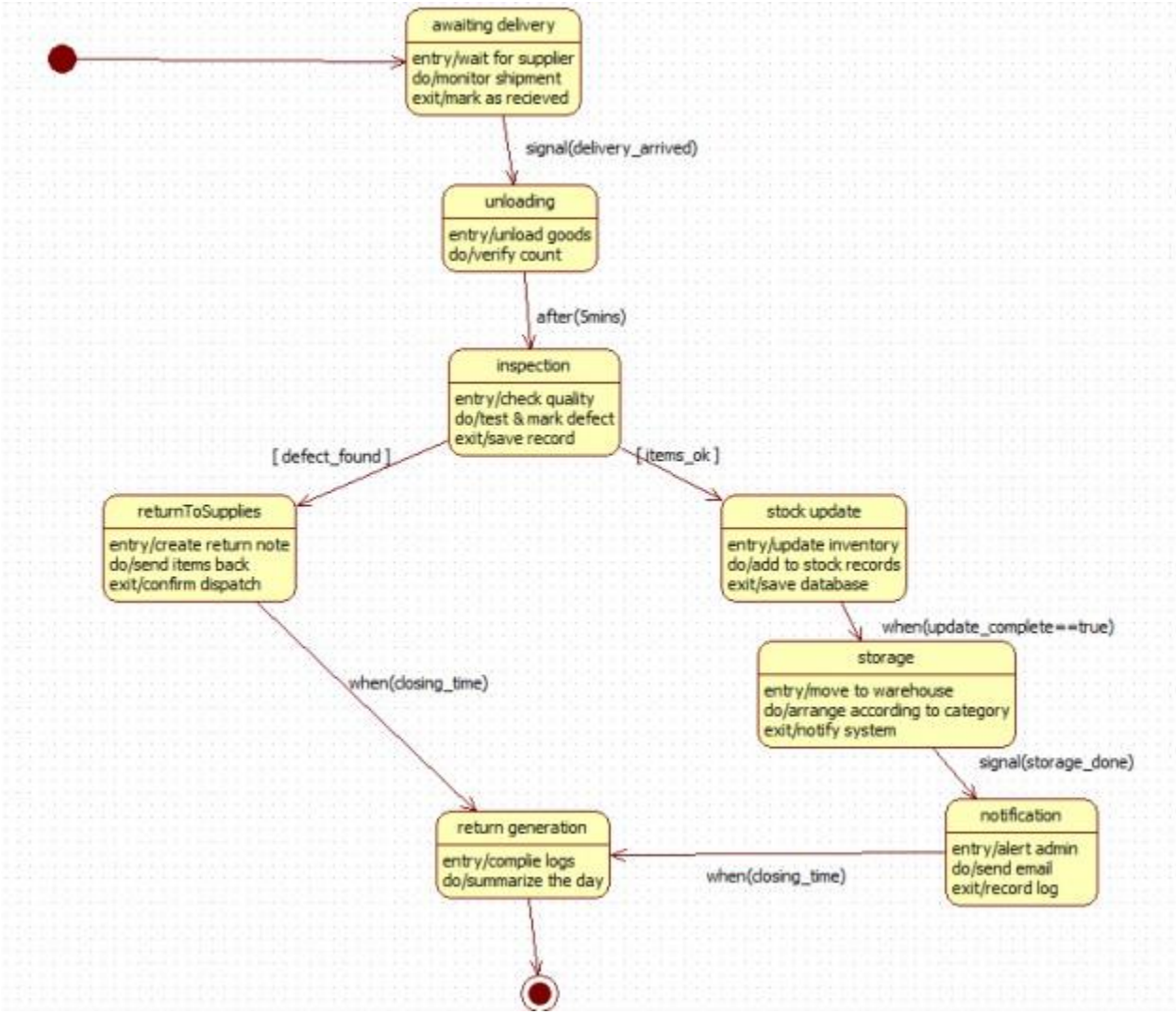

Fig 4.2 State diagram of Stock management system

The state diagram models the workflow of handling incoming goods in an inventory system. The process starts with awaiting delivery, followed by unloading and inspecting the goods. If defects are found, items are returned to the supplier; if the goods are acceptable, the system updates stock records and moves items into storage. After storage, a notification is sent to the administrator. At closing time, the system generates a daily return log, summarizing all activities before ending the process.

Use-case diagram:

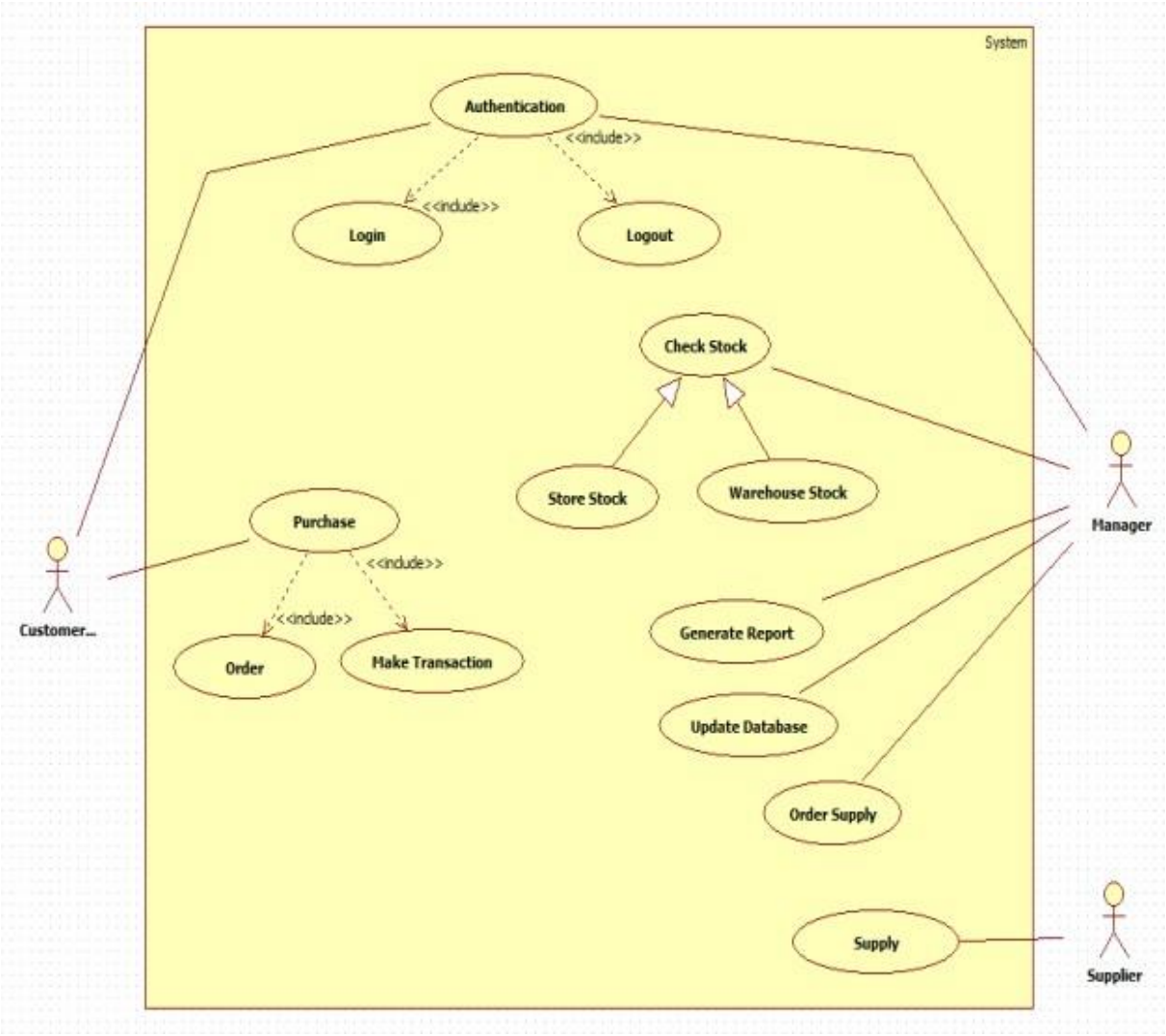

Fig 4.3 Use-case diagram of Stock management system

The use case diagram illustrates an inventory and order management system involving three actors: Customer, Manager, and Supplier. Customers can authenticate, log in, make purchases, place orders, and complete transactions. Managers oversee stock by checking warehouse and store quantities, generating reports, updating the database, and ordering supplies. Suppliers provide stock based on the manager's supply orders. Authentication supports all major activities, ensuring secure access to system functions.

Sequence diagram:

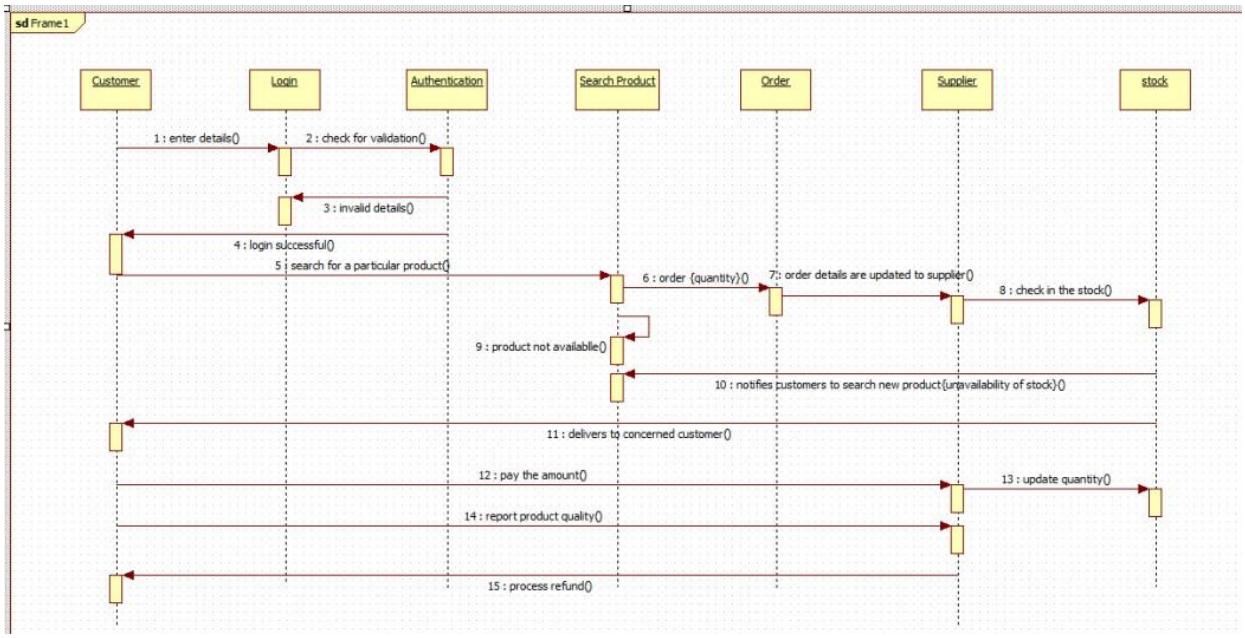

Fig 4.4 Sequence diagram of Stock management system

The sequence diagram illustrates the workflow of an online product ordering system. A customer logs in, searches for a product, and places an order. The system forwards order details to the supplier, who checks stock availability. If the product is unavailable, the system notifies the customer. If available, the supplier delivers the product, the customer makes the payment, and stock levels are updated. The customer may also report product quality, after which the system processes a refund if required.

Activity diagram:

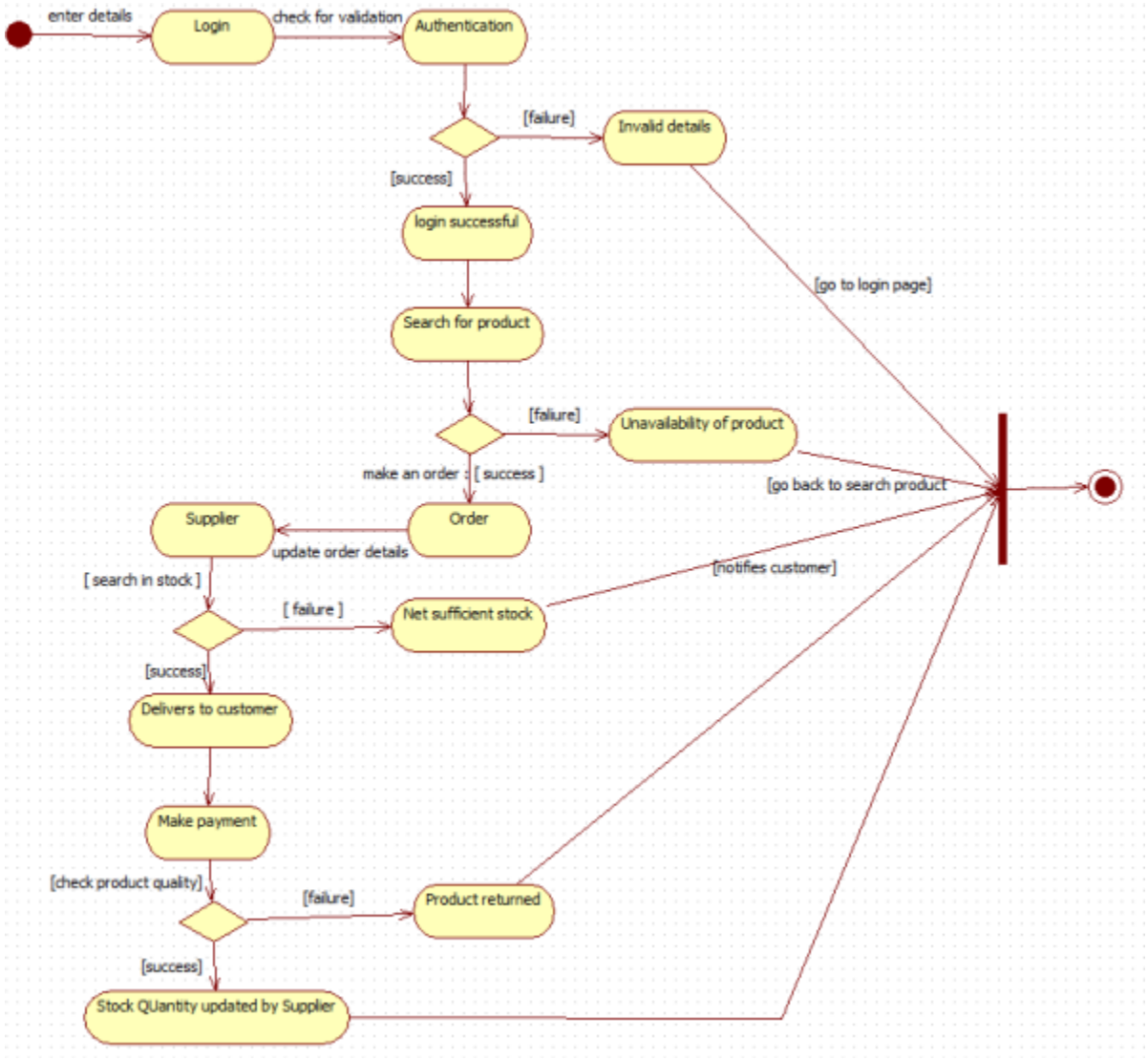

Fig 4.5 Activity diagram of Stock management system

The activity diagram shows the workflow of an online product purchase process. The customer logs in and is authenticated; if valid, they search for a product and place an order. The supplier checks stock availability and updates order details. If stock is sufficient, the product is delivered, payment is made, and product quality is checked. If the product fails the quality check, it is returned; if successful, stock quantity is updated. Invalid login or unavailable products redirect the user back to the appropriate steps.

5. Passport Automation System

Problem Statement

Traditional passport application and verification processes involve multiple manual steps, long queues, and paperwork, leading to delays, errors, and inefficiencies. Tracking application status, verifying documents, scheduling appointments, and issuing passports become increasingly difficult as the number of applicants grows.

A Passport Automation System is needed to streamline application submission, document verification, appointment scheduling, status tracking, and passport issuance. The goal is to reduce processing time, minimize human errors, improve transparency, and provide a faster, user-friendly experience for both applicants and officials through a centralized digital platform.

Software Requirements Specification (SRS)

5) SRS for Passport Automation System

1. Introduction

1.1 Purpose: This SRS defines requirements for a system that automates passport application, verification and issuance.

1.2 Scope: The system allows applicants to apply online, track status and schedule appointments while automating backend verification.

1.3 Overview: PAS minimizes manual paperwork and ensures fast, secure passport issuance.

2. General Description

The System supports applicants, government staff, and administrators for efficient passport processing

3. Functional Requirements

Application Management: Submit new applications online.

Document Upload & verification: Upload ID/ address proofs, automated verification.

Fee payment: Secure online payment gateway integration.

Bafna Gold
Date: Page:

Appointment Scheduling: Book slots for document verification & biometrics.

Tracking: Applicants track application status online.

Report Generation: for administrative and audit purposes.

4. Interface Requirements
UI: Applicant portal & admin dashboard
Integration: Biometric devices, payment gateway, police verification system.

5. Performance Requirements
Response time ≤ 3 seconds. Handle 10,000+ concurrent users. Process 1 million+ records securely.

6. Design Constraints: Use government-approved security standards.
Database: Oracle or PostgreSQL.

7. Non-Functional Requirements
Security: End-to-end encryption, strong authentication
Reliability: 99.9% uptime
Scalability: National-level deployment
Portability: Accessible on desktop, mobile & kiosks.

8. Preliminary Schedule & Budget
Development duration: 12 months
Estimated budget: $ 500,000

Class diagram:

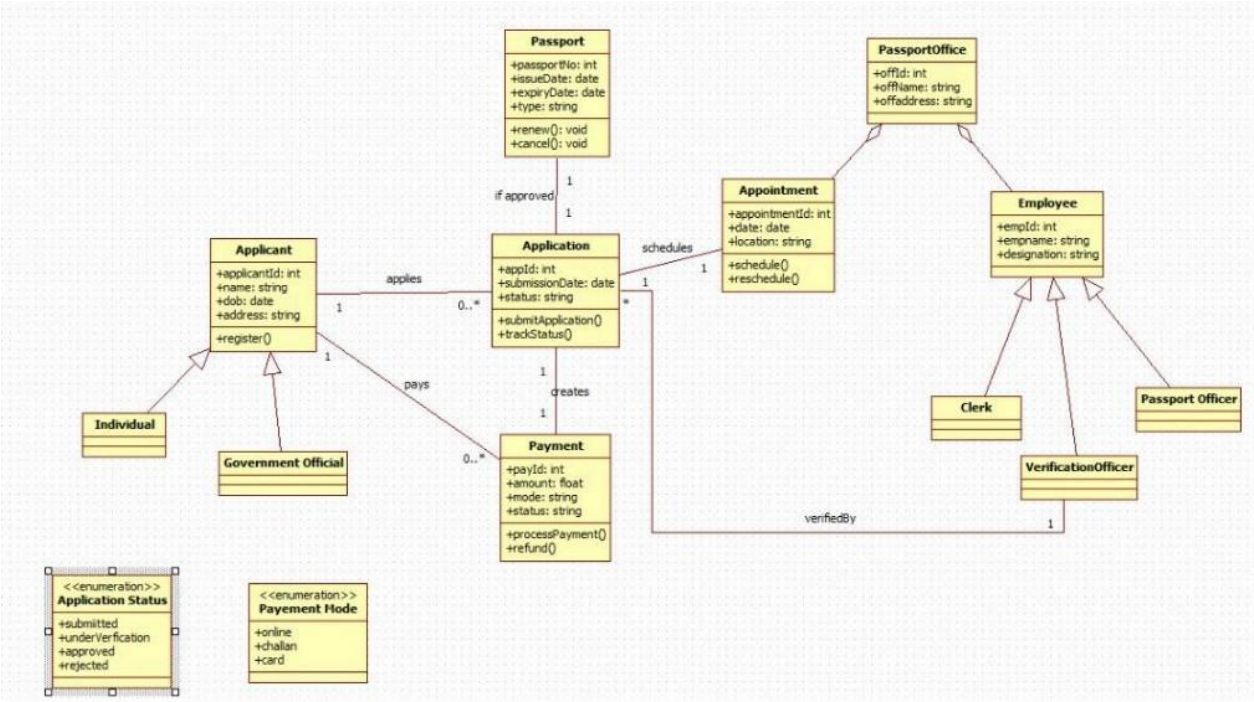

Fig 5.1 Class diagram of Passport Automation System

The class diagram represents a passport application management system. An Applicant—either an individual or government official—submits an Application and makes the required Payment. The Passport Office manages appointments and employs different staff roles, including clerks, passport officers, and verification officers, who verify applications. Upon approval, a Passport is issued or renewed. The diagram also includes application statuses and payment modes, showing how the system processes applications from submission to approval.

State diagram:

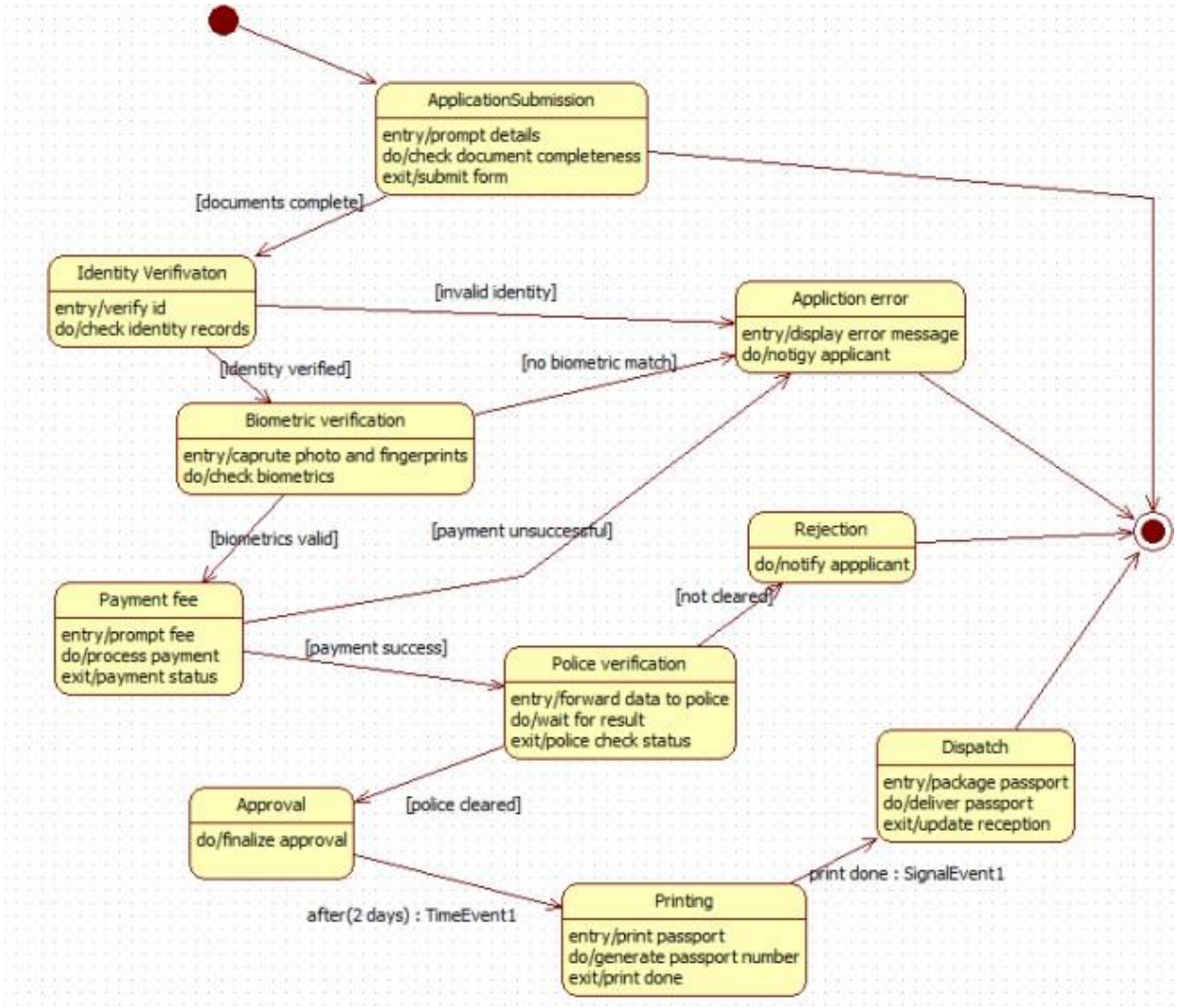

Fig 5.2 State diagram of Passport Automation System

The state diagram illustrates the complete workflow of a passport application process. It begins with application submission, followed by identity and biometric verification. If verification fails, the system moves to an error or rejection state. Upon successful verification, the applicant pays the required fee, and the application proceeds to police verification. Once cleared, the passport is approved, printed, and dispatched to the applicant. Any failure at intermediate steps leads to rejection.

Use-case diagram:

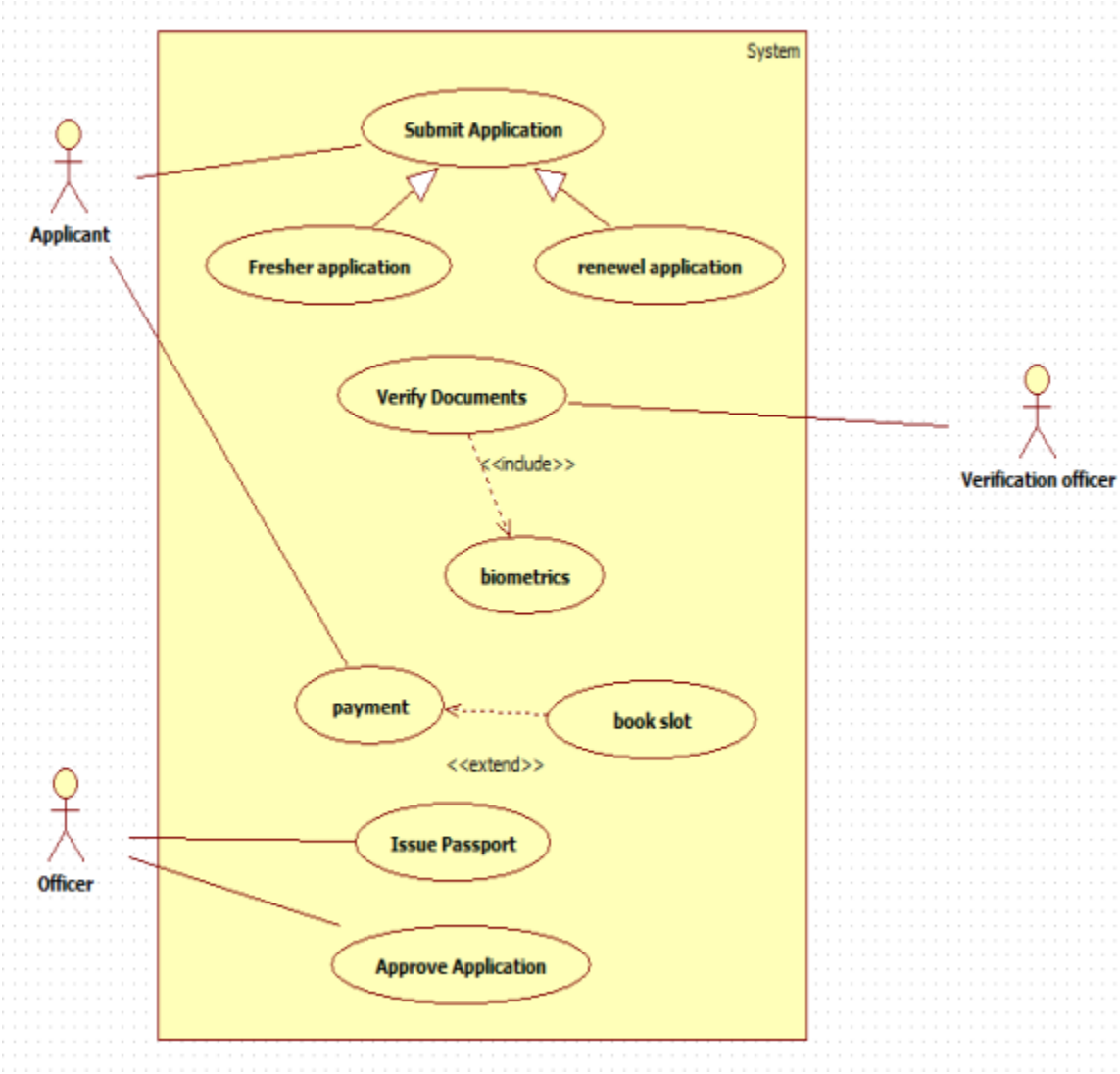

Fig 5.3 Use-case diagram of Passport Automation System

The use case diagram represents a passport application system involving Applicants, Officers, and Verification Officers. Applicants submit either a fresh or renewal application, after which the system verifies documents and captures biometrics. Applicants then make payments and may book a slot if required. Officers handle the approval and issuance of the passport, completing the application process.

Sequence diagram:

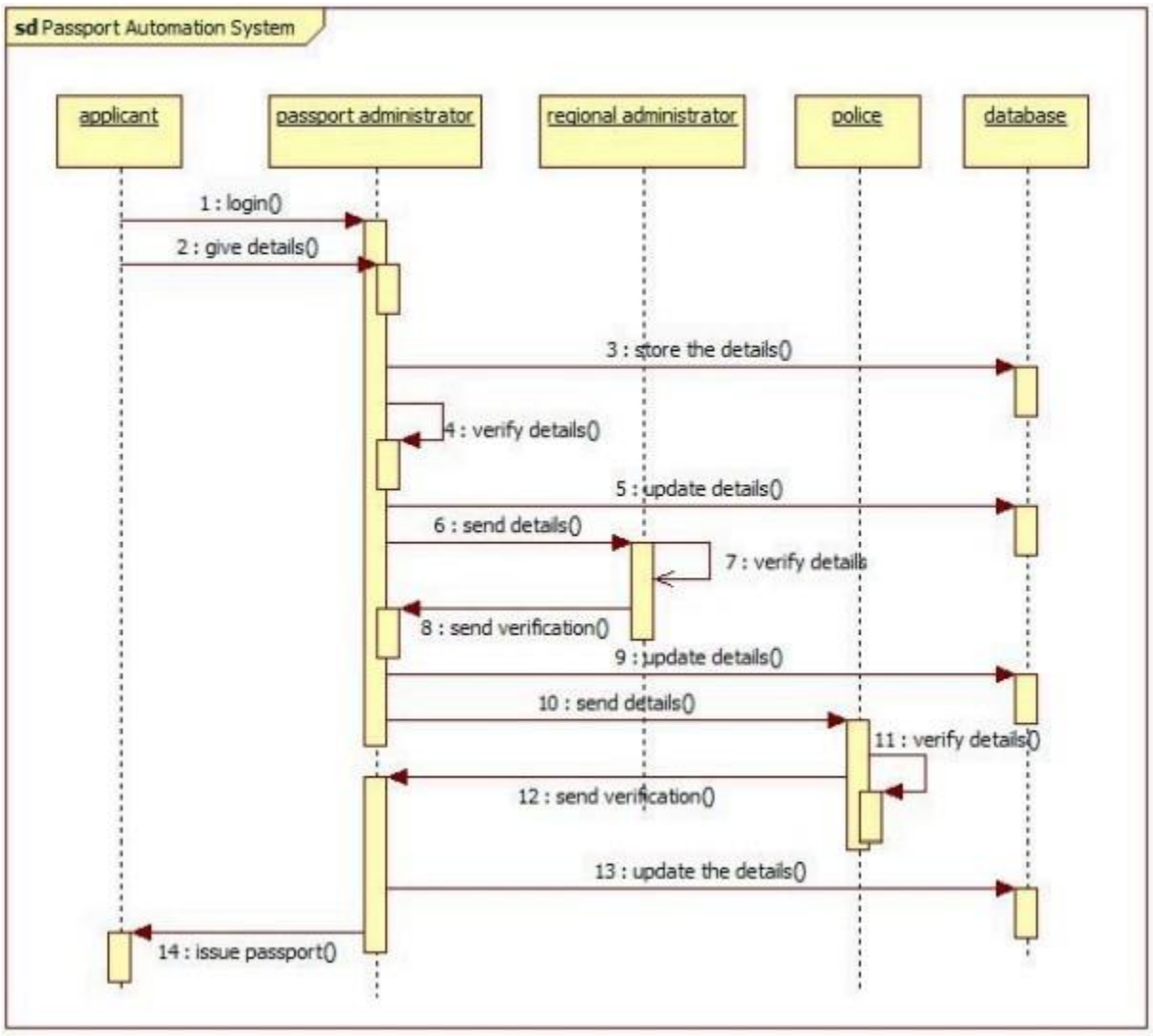

Fig 5.4 Sequence diagram of Passport Automation System

The sequence diagram shows the workflow of a passport automation system. The applicant logs in and submits details, which the passport administrator stores in the database. The regional administrator and police department sequentially verify the submitted information and update the records. After receiving verification from both authorities, the passport administrator issues the passport to the applicant.

Activity diagram:

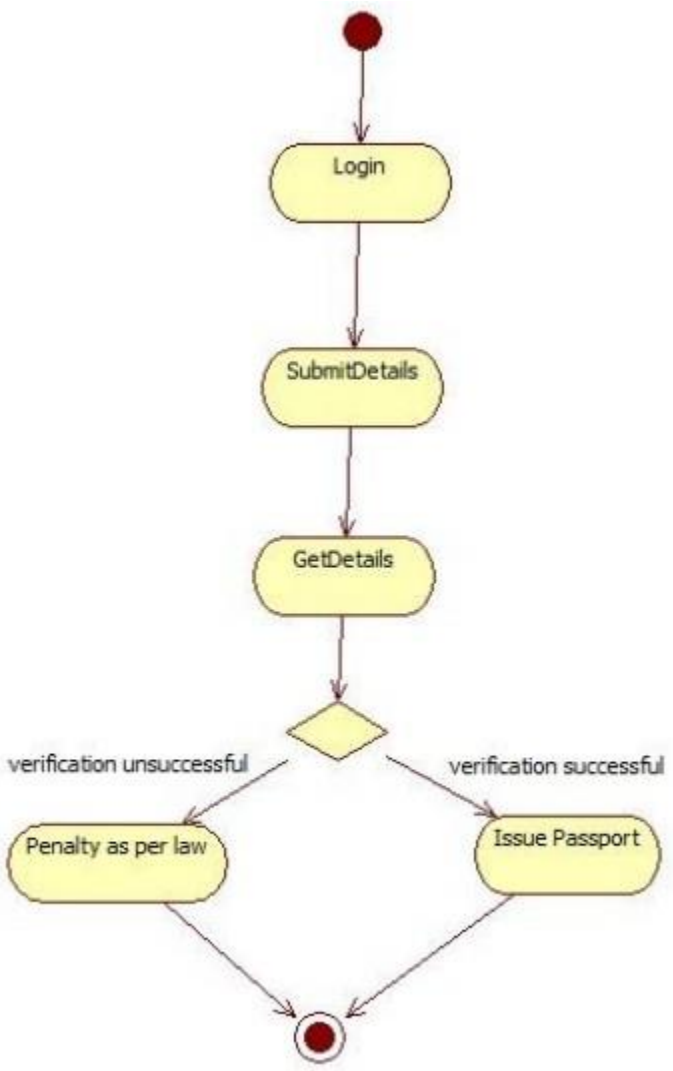

Fig 5.5 Activity diagram of Passport Automation System

The activity diagram shows a simple passport verification process. A user logs in, submits their details, and the system retrieves and checks the information. If verification is successful, a passport is issued; if verification fails, a legal penalty is applied. The process then ends.