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:

Chitrashree K

1BM23CS081

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 Chitrashree K(1BM21CS081) during the 5th Semester August 2025-December 2025

.

Signature of the Faculty Incharge:
Sonika Sharma D
Assistant Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

Table of Contents

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in managing reservations, guest check-in/check-out, room availability, billing, and staff operations due to manual or outdated processes.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)

19/08/2025

LAB-01 - SRS for Hotel Management System and Credit Card Processing

Hotel Management System:-

1. Introduction:-

1.1 Purpose:-

This document specifies the requirements of Hotel Management System. This allows the hotel staff to look into guest check-ins/outs, billing, managing reservations etc.
The system ensures accuracy, efficiency in day-to-day hotel operations.
It will serve for seamless interaction b/w hotel staff, administrators, reception staff, housekeeping and customers. It is designed to improve customer satisfaction.

1.2 Document Conventions:-

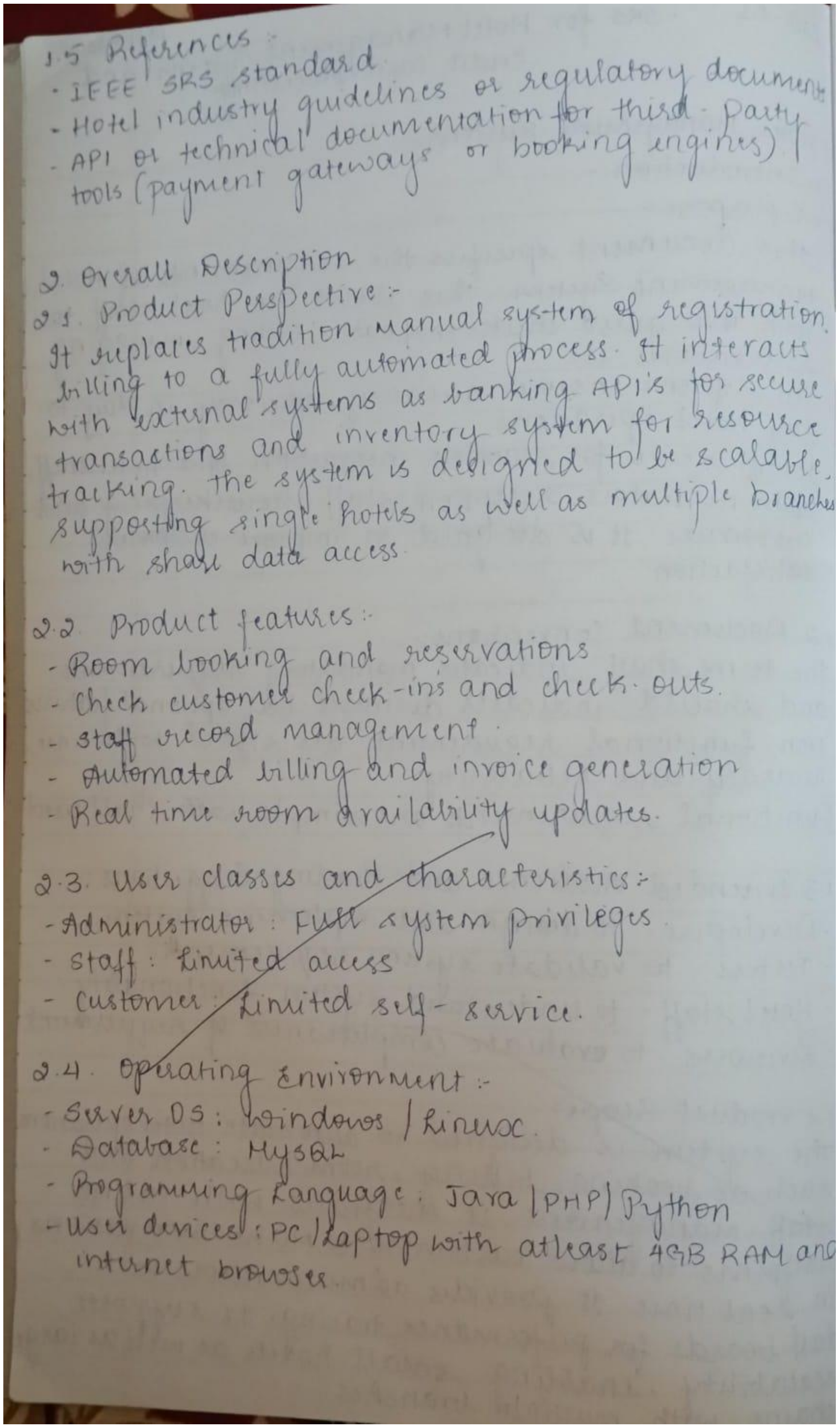
The terms 'shall' indicates mandatory requirements. and 'should' indicates desirable but optional features
Non functional requirements are stated with time, accuracy and reliability.
Functional requirements are simple with shall used

1.3 Intended Audience and Reading Suggestions:-

- Developers - to design and implement system
- Testers - to validate system requirements
- Hotel staff - to understand system functionality.
- Reviewers - to evaluate completedness of requirements

1.4 Product Scope:-

The system is designed to automate hotel operations such as booking, billing, room allocation and staff management. It integrates with online platforms to allow customers to make reservations in real time. It provides administrators with dashboards for performance tracing. It supports scalability, enabling small hotels as well as large chains with multiple branches.



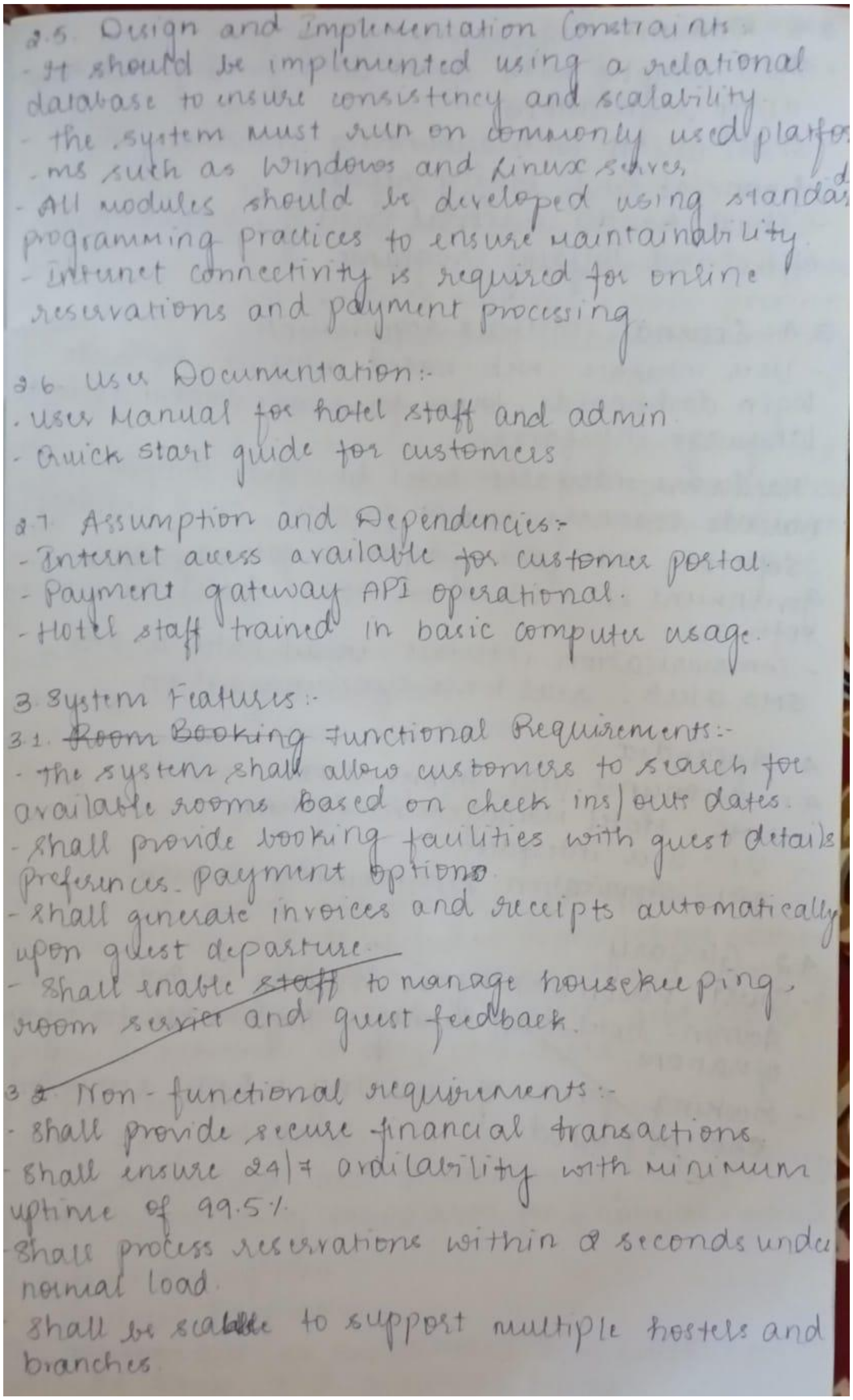
1.5 References :
- IEEE SRS standard
- Hotel industry guidelines or regulatory documents
- API or technical documentation for third - party tools (payment gateways or booking engines) |

2. Overall Description
2.1. Product Perspective :-
It replaces tradition manual system of registration billing to a fully automated process. It interacts with external systems as banking API's for secure transactions and inventory system for resource tracking. The system is designed to be scalable supporting single hotels as well as multiple branches with share data access.

2.2 Product features :-
- Room booking and reservations
- Check customer check-ins and check-outs.
- staff record management.
- Automated billing and invoice generation
- Real time room availability updates.

2.3. User classes and characteristics :-
- Administrator : Full system privileges
- Staff : Limited access
- Customer : Limited self - service.

2.4. Operating Environment :-
- Server OS : Windows / Linux.
- Database : MySQL
- Programming Language : Java | PHP | Python
- User devices : PC / Laptop with atleast 4GB RAM and internet browser



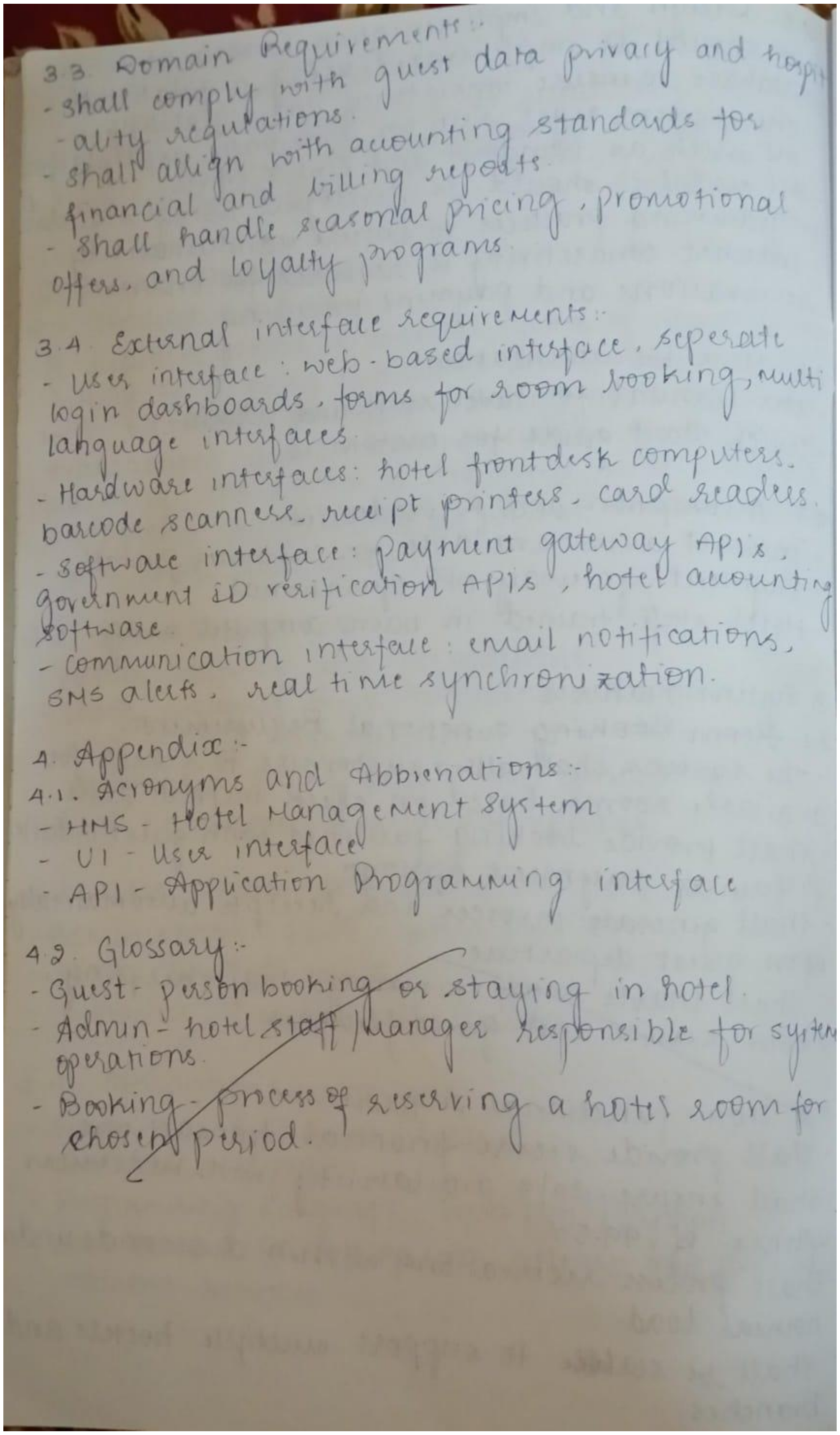
2.5. Design and Implementation Constraints
- It should be implemented using a relational database to ensure consistency and scalability
- the system must run on commonly used platfo-ms such as Windows and Linux servers
- All modules should be developed using standard programming practices to ensure maintainability
- Internet connectivity is required for online reservations and payment processing

2.6 User Documentation:-
. User Manual for hotel staff and admin
- Quick start guide for customers

2.7 Assumption and Dependencies:-
- Internet access available for customer portal
- Payment gateway API operational.
- Hotel staff trained in basic computer usage.

3 System Features:-
3.1 ~~Room Booking~~ Functional Requirements:-
- The system shall allow customers to search for available rooms based on check ins/outs dates.
- shall provide booking facilities with guest details preferences. payment options
- shall generate invoices and receipts automatically upon guest departure.
- Shall enable ~~staff~~ to manage housekeeping, room service and guest feedback.

3.2 Non-functional requirements:-
- Shall provide secure financial transactions
Shall ensure 24/7 availability with minimum uptime of 99.5%
Shall process reservations within 2 seconds under normal load
Shall be scalable to support multiple hostels and branches



3.3 Domain Requirements:-
- shall comply with guest data privacy and hospit
-ality regulations.
- shall allign with accounting standards for
financial and billing reports.
- shall handle seasonal pricing, promotional
offers, and loyalty programs.

3.4 External interface requirements:-
- User interface: web-based interface, seperate
login dashboards, forms for room booking, multi
language interfaces.
- Hardware interfaces: hotel frontdesk computers.
barcode scanners. receipt printers. card readers.
- Software interface: payment gateway API's,
government iD verification APIs, hotel accounting
software.
- communication interface: email notifications,
SMS alerts. real time synchronization.

4. Appendix:-

4.1. Acronyms and Abbrevations:-
- HMS - Hotel Management System
- UI - User interface
- API - Application Programming interface

4.2 Glossary:-
- Guest - person booking or staying in hotel.
- Admin - hotel staff/manager responsible for system
operations.
- Booking - process of reserving a hotel room for
chosen period.

Class Diagram:

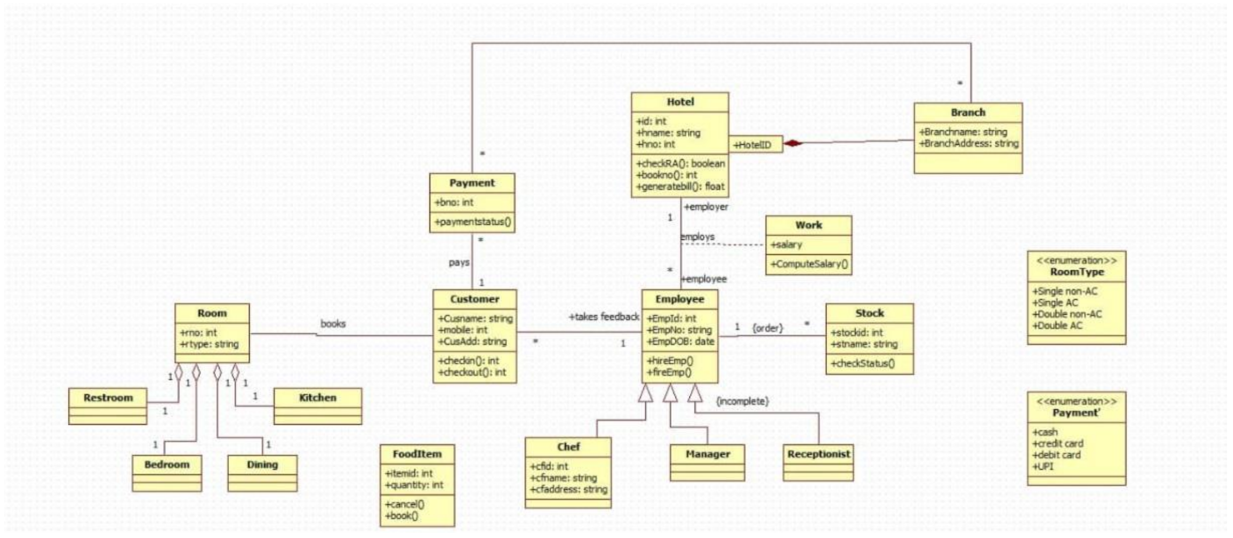


Fig 1.1 Class diagram of Hotel Management System

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:

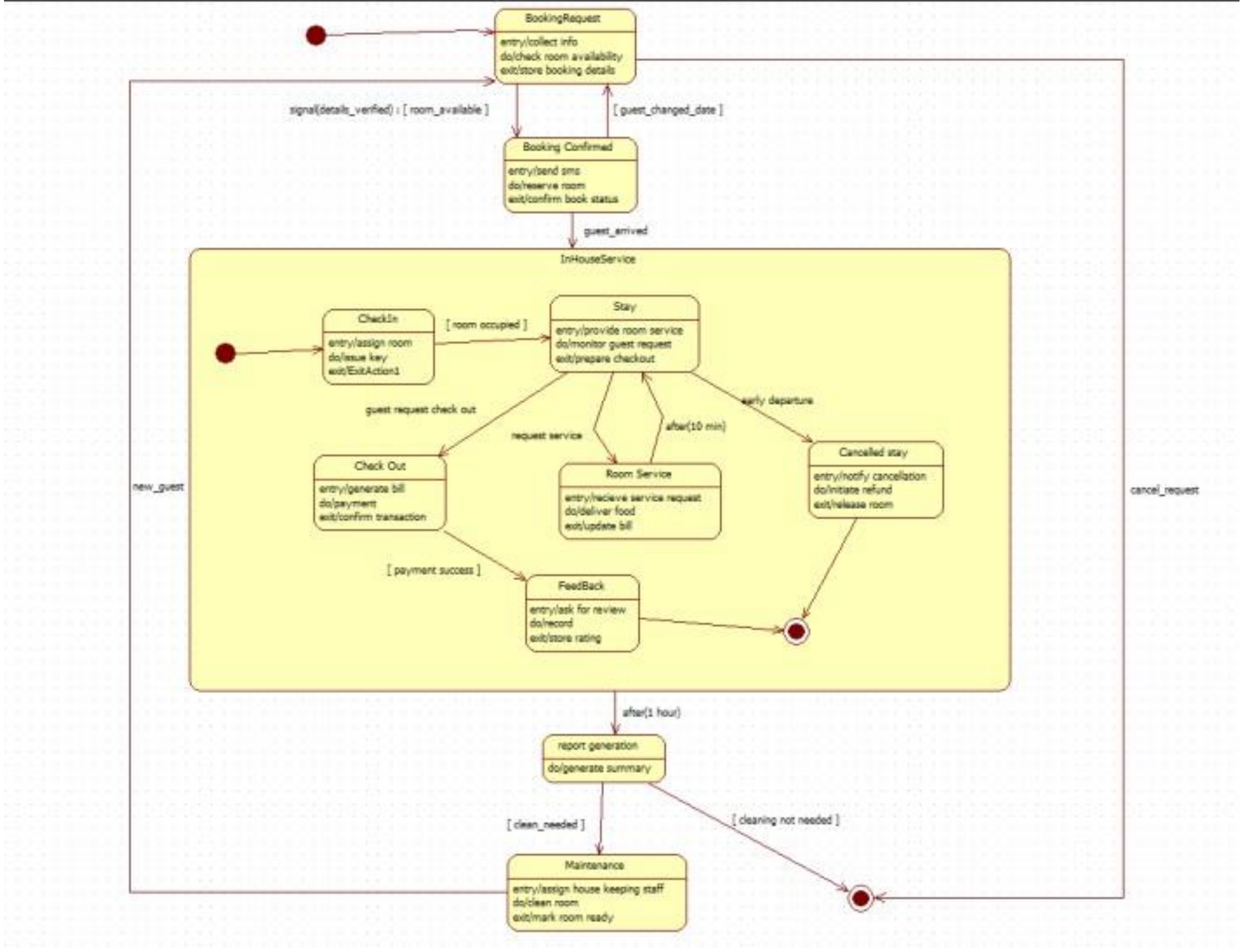


Fig 1.2 State diagram of Hotel Management System

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:

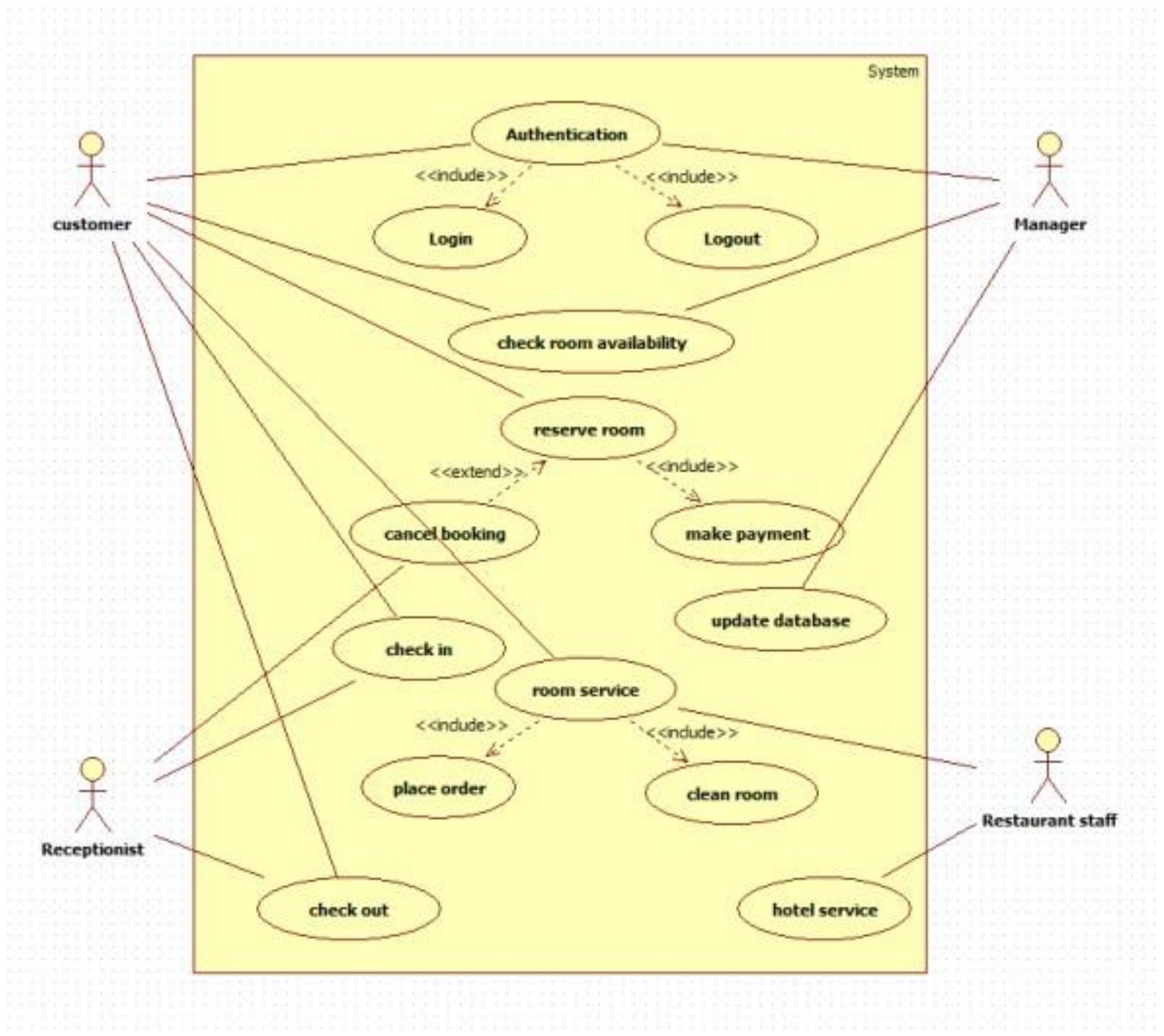


Fig 1.3 Use case diagram of Hotel Management System

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:

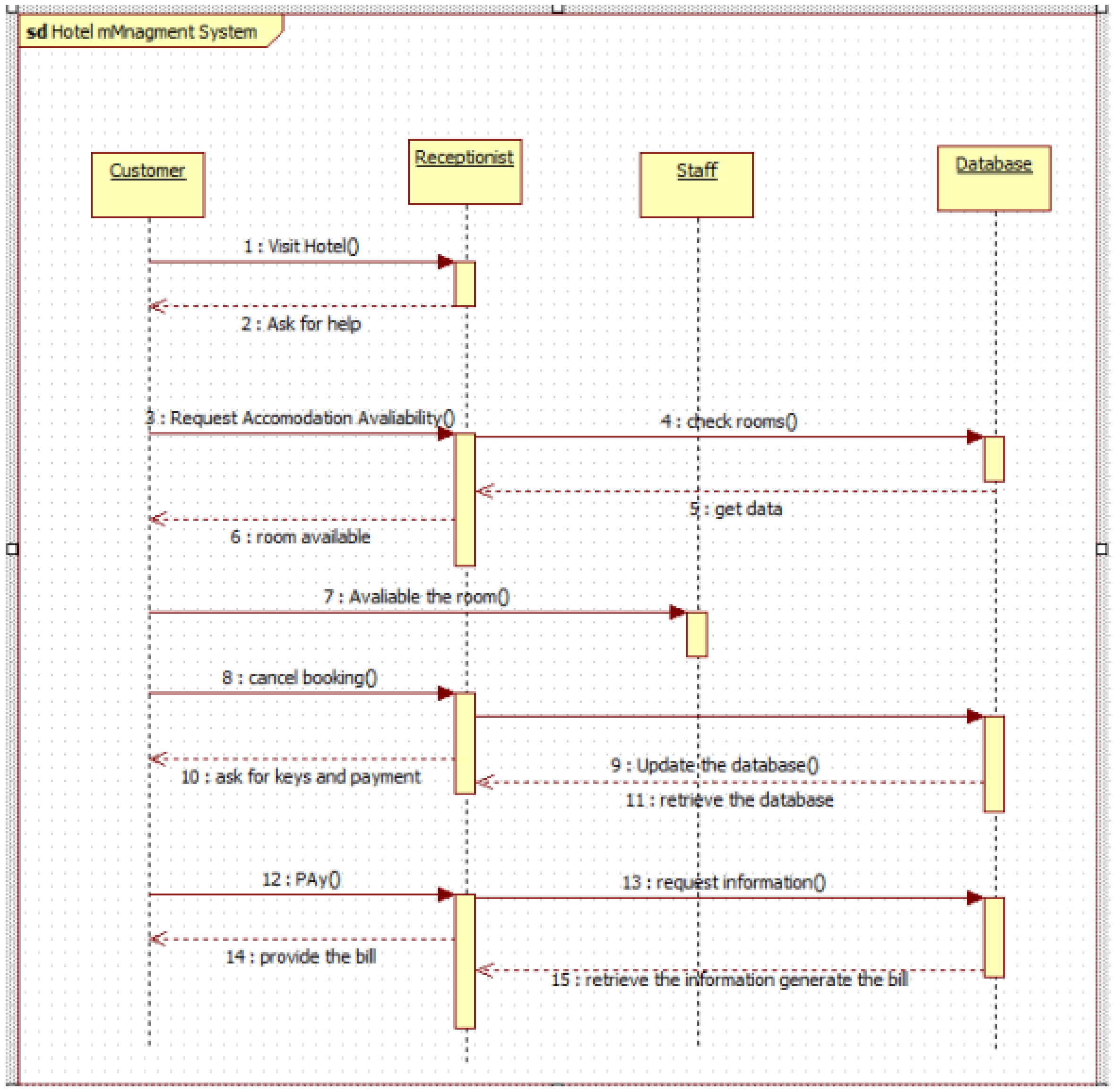

sd Hotel mMnagment System
Customer
Receptionist
Staff
Database
1 : Visit Hotel()
2 : Ask for help
3 : Request Accomodation Avaliability()
4 : check rooms()
5 : get data
6 : room available
7 : Avaliable the room()
8 : cancel booking()
9 : Update the database()
10 : ask for keys and payment
11 : retrieve the database
12 : PAy()
13 : request information()
14 : provide the bill
15 : retrieve the information generate the bill

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:

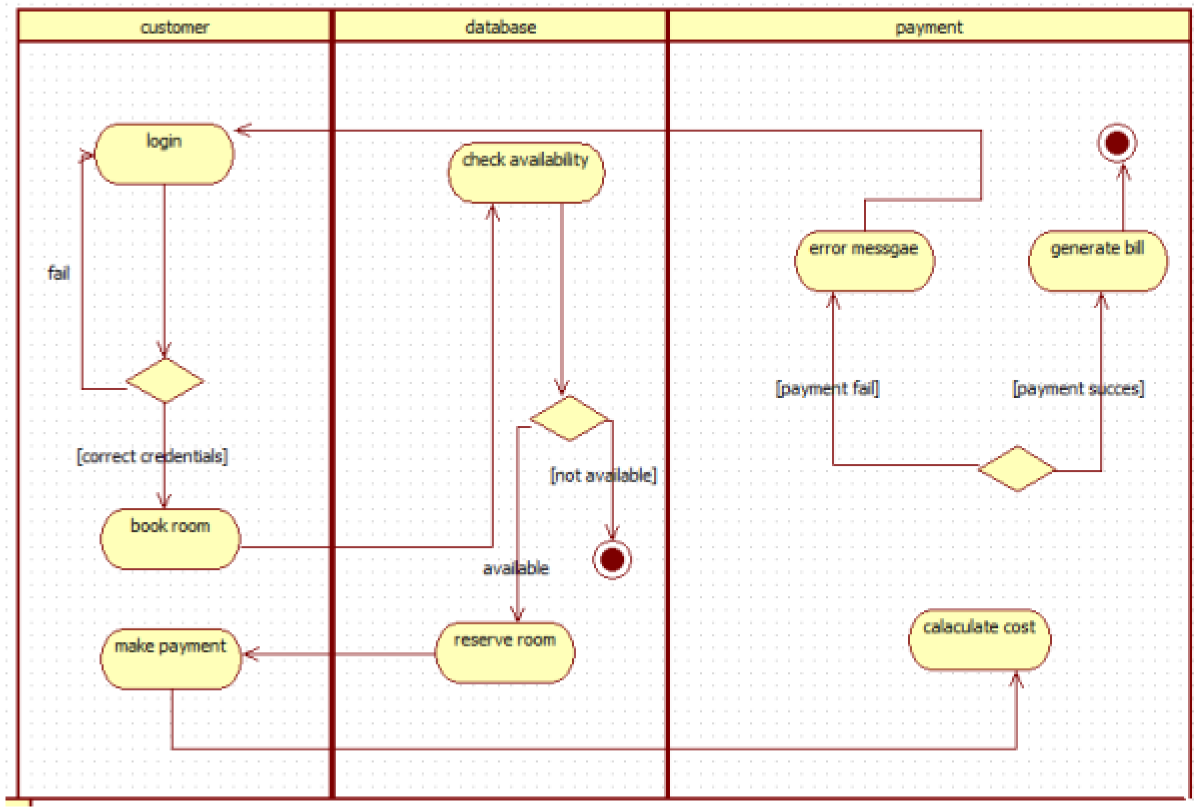


Fig 1.5 Activity diagram of Hotel Management System

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.

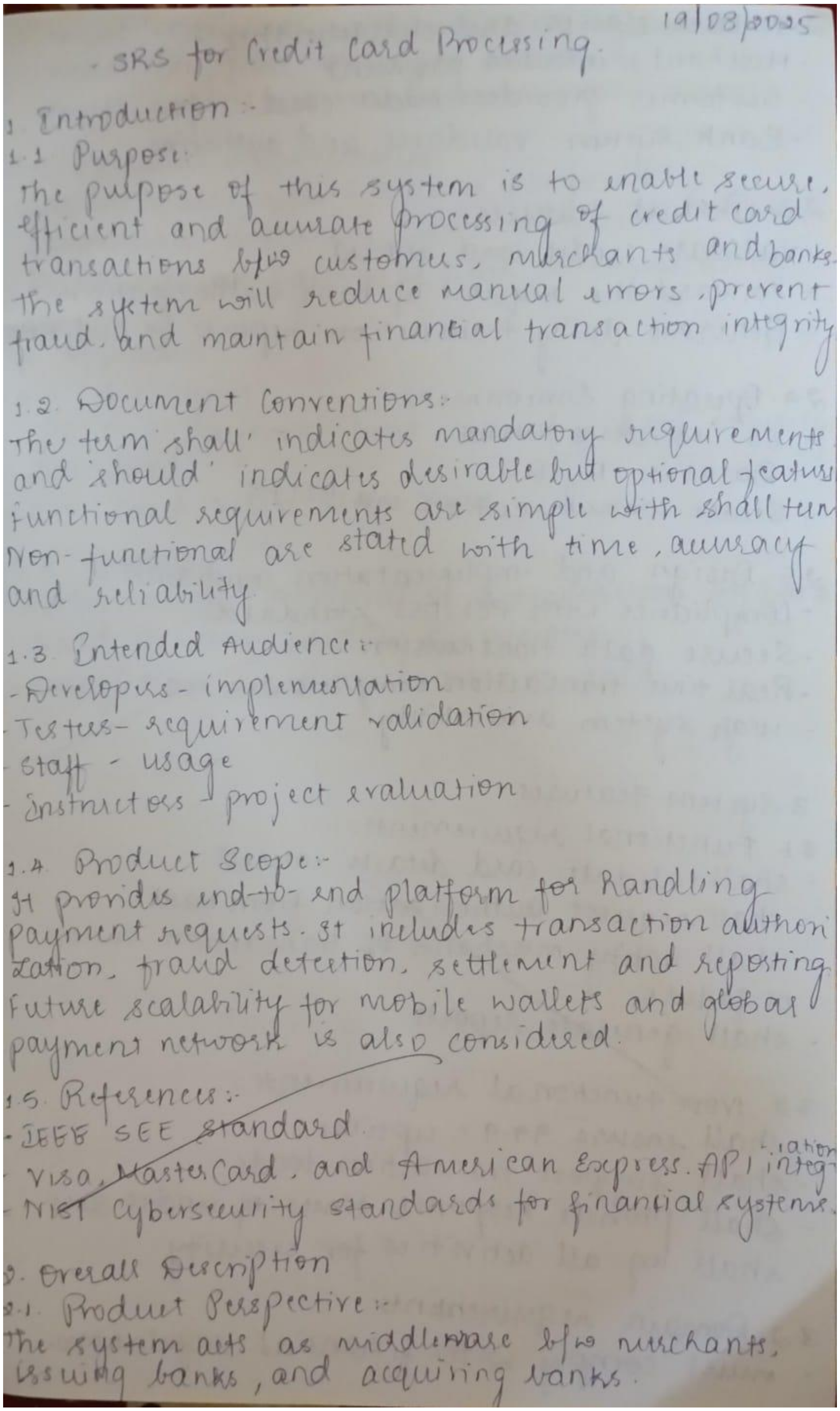
2. Credit Card Automation System

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)



19/08/2025

- SRS for Credit Card Processing.

1. Introduction :-

1.1 Purpose

The purpose of this system is to enable secure, efficient and accurate processing of credit card transactions b/w customers, merchants and banks. The system will reduce manual errors, prevent fraud and maintain financial transaction integrity

1.2. Document Conventions:-

The term 'shall' indicates mandatory requirements and 'should' indicates desirable but optional features functional requirements are simple with shall term Non-functional are stated with time, accuracy and reliability.

1.3 Intended Audience:-

- Developers - implementation
- Testers - requirement validation
- Staff - usage
- Instructors - project evaluation

1.4 Product Scope:-

It provides end-to-end platform for handling payment requests. It includes transaction authorization, fraud detection, settlement and reporting. Future scalability for mobile wallets and global payment network is also considered.

1.5. References:-

- IEEE SEE standard
- Visa, MasterCard, and American Express. API integration
- NIST Cybersecurity standards for financial systems.

2. Overall Description

2.1. Product Perspective:-

The system acts as middleware b/w merchants, issuing banks, and acquiring banks.

2.2. User classes and characteristics :-
- Merchant initiates payment
- Customer provides credit card info
- Bank system: validates and authorize

2.3. Product Features:-
- Validate credit card entered
- Forward the request of bank authorization
- Generate daily transaction reports for auditing

2.4. Operating Ennironment
- OS: Windows | Linux
- Database: MySQL
- Secure communication via HTTPS

2.5. Design and implementation constraints :-
- Compliance with PCI DSS standards.
- Secure data transmission
- Real time transaction complete 2-3 sec
- High system availability.

3 System features :-
3.1. Functional requirements :-
- shall validate card details entered.
- shall request authorization from bank.
- shall notify merchants of transaction success or failure
- shall generate reports.

3.2. Non functional requirements :-
- shall ensure 99.9% uptime
- shall support transaction loads
- shall provide response times of under 3sec.
- shall log all activities for security

3.2 Domain requirements.
- must comply with finantial regulations

- must integrate with existing gateways.
- must support multiple currencies.
- must maintain transaction

3.4. External Requirements:-
- User interface- web & MOBile platform
- Hardware: Software, Communication interface.
- Banking interface- real time communication

4. Appendix:-
4.1. Acronyms:-
- PCIDSS- Payment card industry data security standard.
- CVV- card verification value

4.2. Glossary:-
- Authorization - Approval of a transaction by bank
- Fraud detection: identify suspicious.
- encryption: securing sensitive data.

Class Diagram:

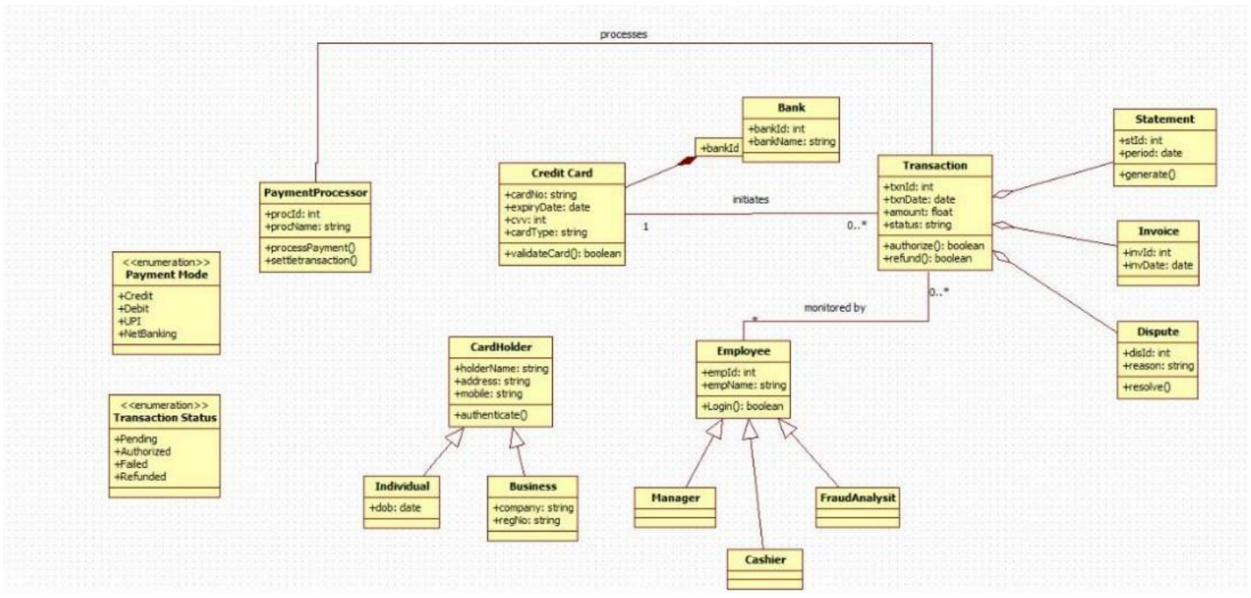


Fig 2.1 Class diagram of Credit Card Processing System

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:

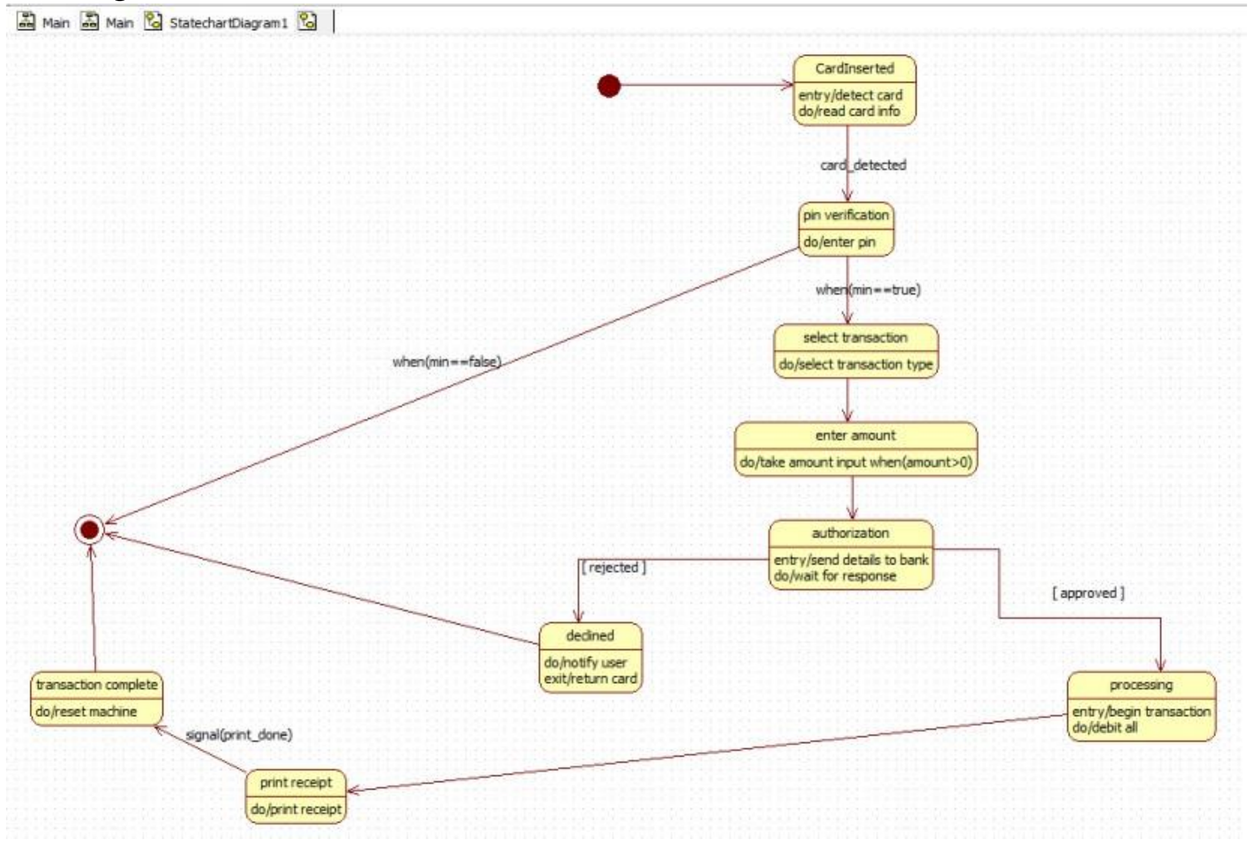


Fig 2.2 State diagram of Credit Card Processing System

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:

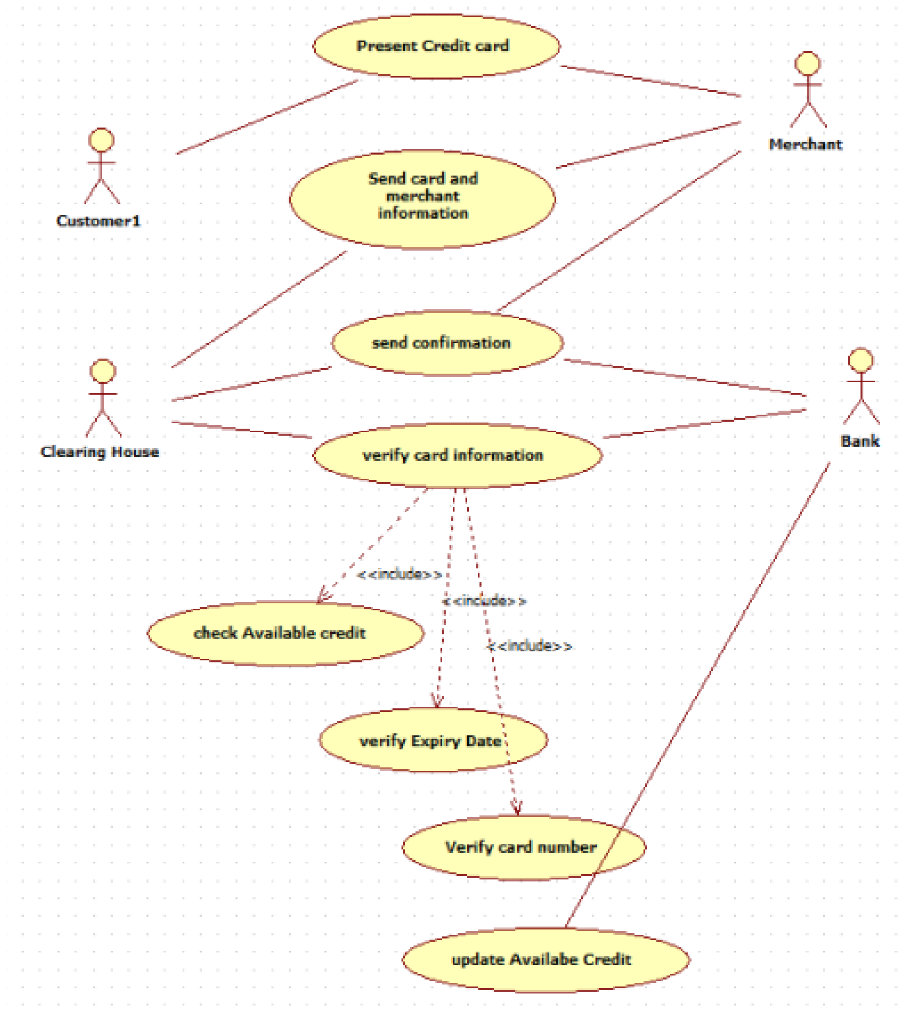


Fig 2.3 Use-case diagram of Credit Card Processing System

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:

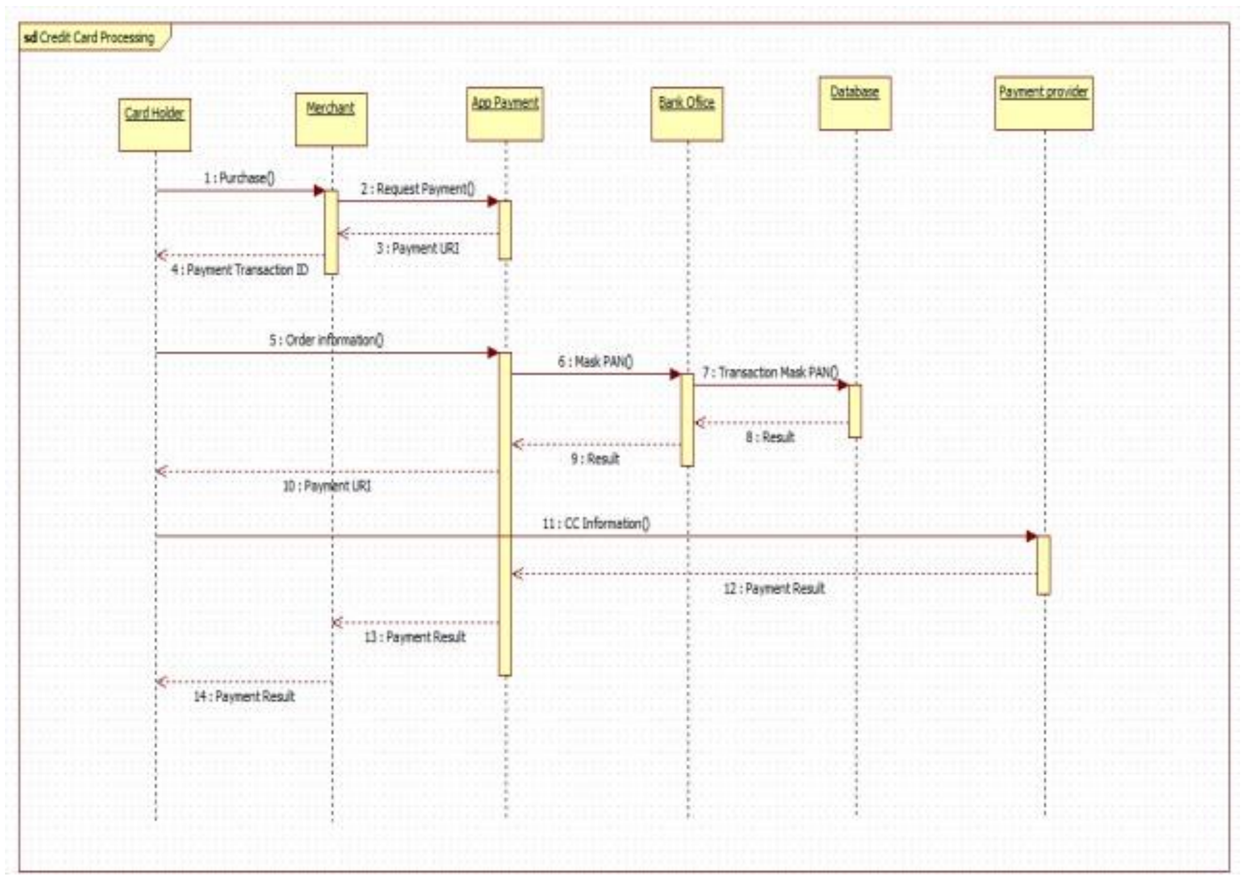
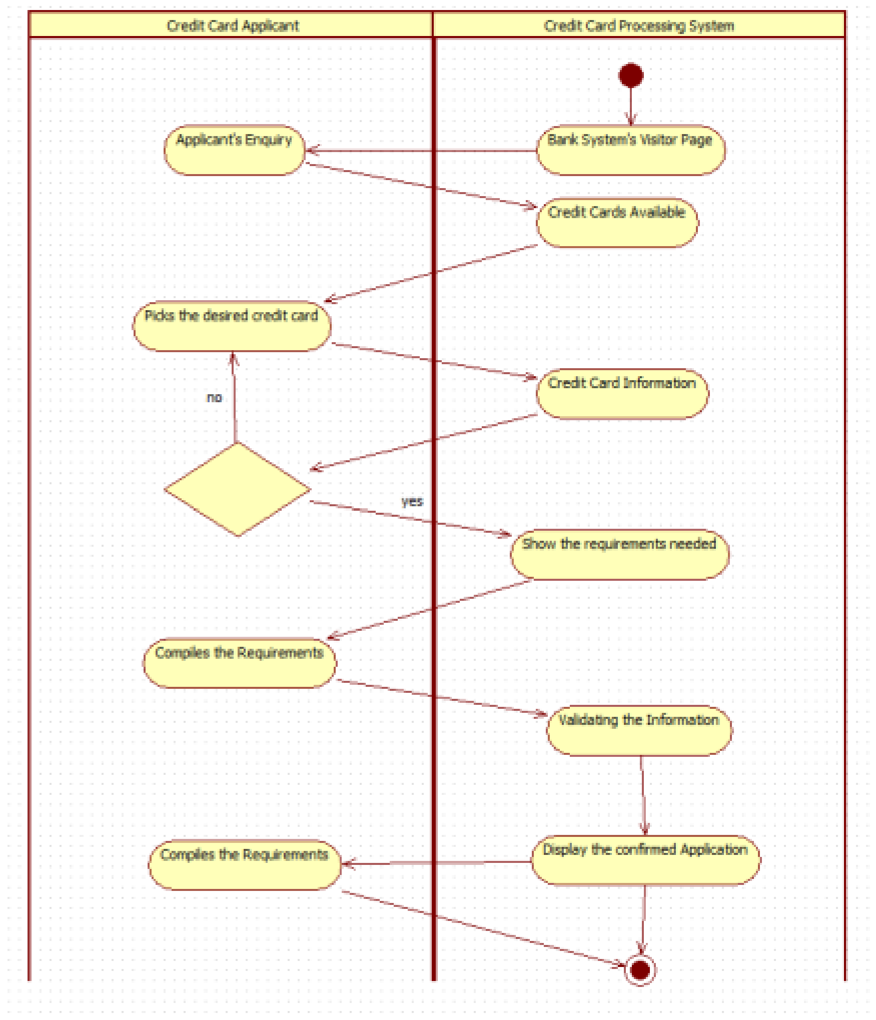


Fig 2.4 Sequence diagram of Credit Card Processing System

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:



Credit Card Applicant
Credit Card Processing System
Applicant's Enquiry
Bank System's Visitor Page
Credit Cards Available
Picks the desired credit card
Credit Card Information
no
yes
Show the requirements needed
Compiles the Requirements
Validating the Information
Compiles the Requirements
Display the confirmed Application

Fig 2.5 Activity diagram of Credit Card Processing System

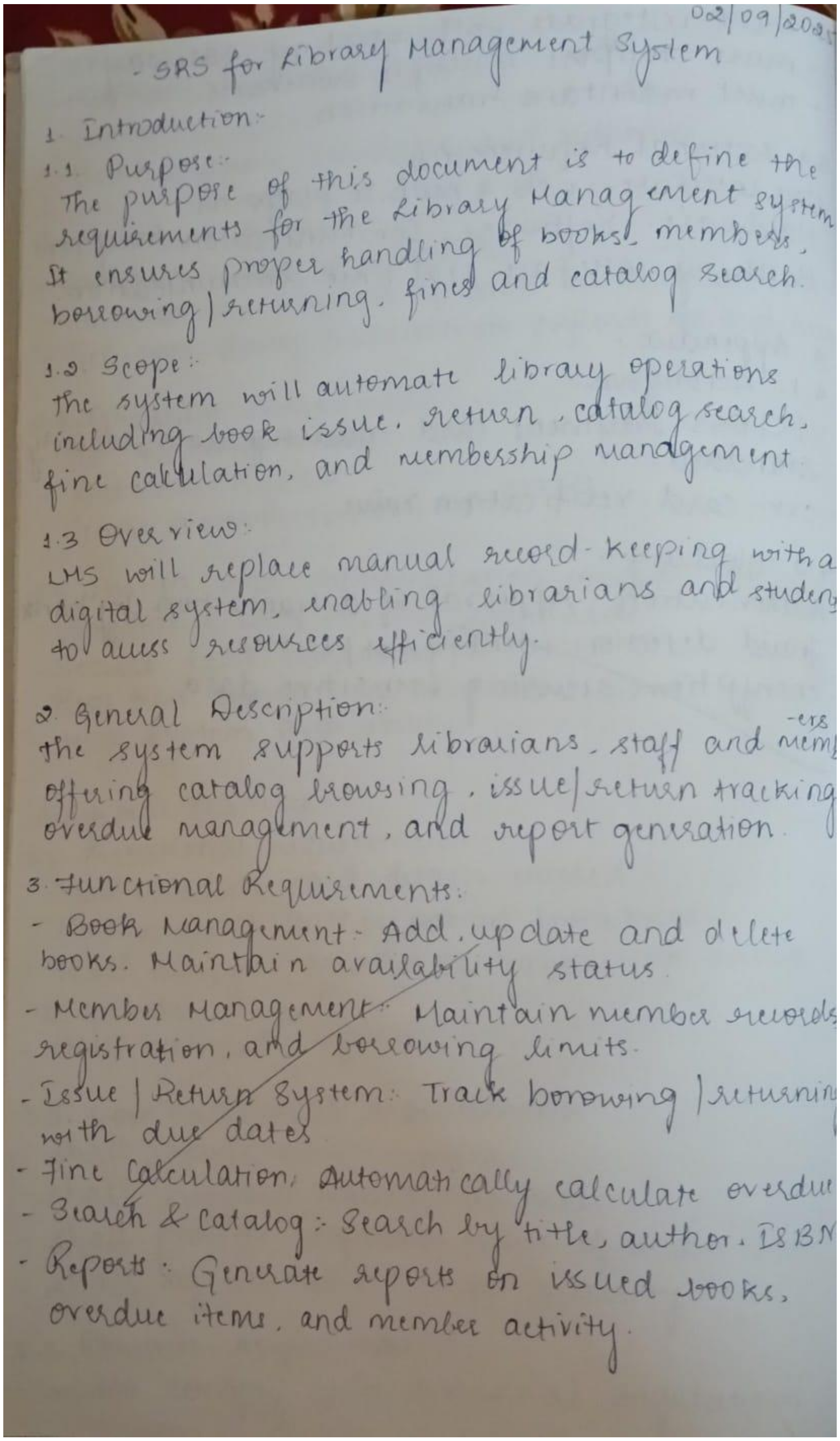
The activity diagram illustrates the process of a credit card applicant applying through the bank's credit card processing system. The applicant sends an enquiry, views available credit cards, and selects one. The system displays the card information and required documents. After the applicant submits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



02/09/202

- 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 studen to access resources efficiently.

2. General Description:-
The system supports librarians, staff and memb-ers 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/returnin with due dates
- Fine Calculation: Automatically calculate overdue
- Search & Catalog:- Search by title, author, ISBN
- 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 upto 500 concurrent users. Ensure real-time inventory update.

6. Design Constraints:-
- Relational DB, Web based architecture with Java | Python backend.

7. Non-functional Requirements:
- Security : Role based access [admin, student)
- 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

Class diagram:

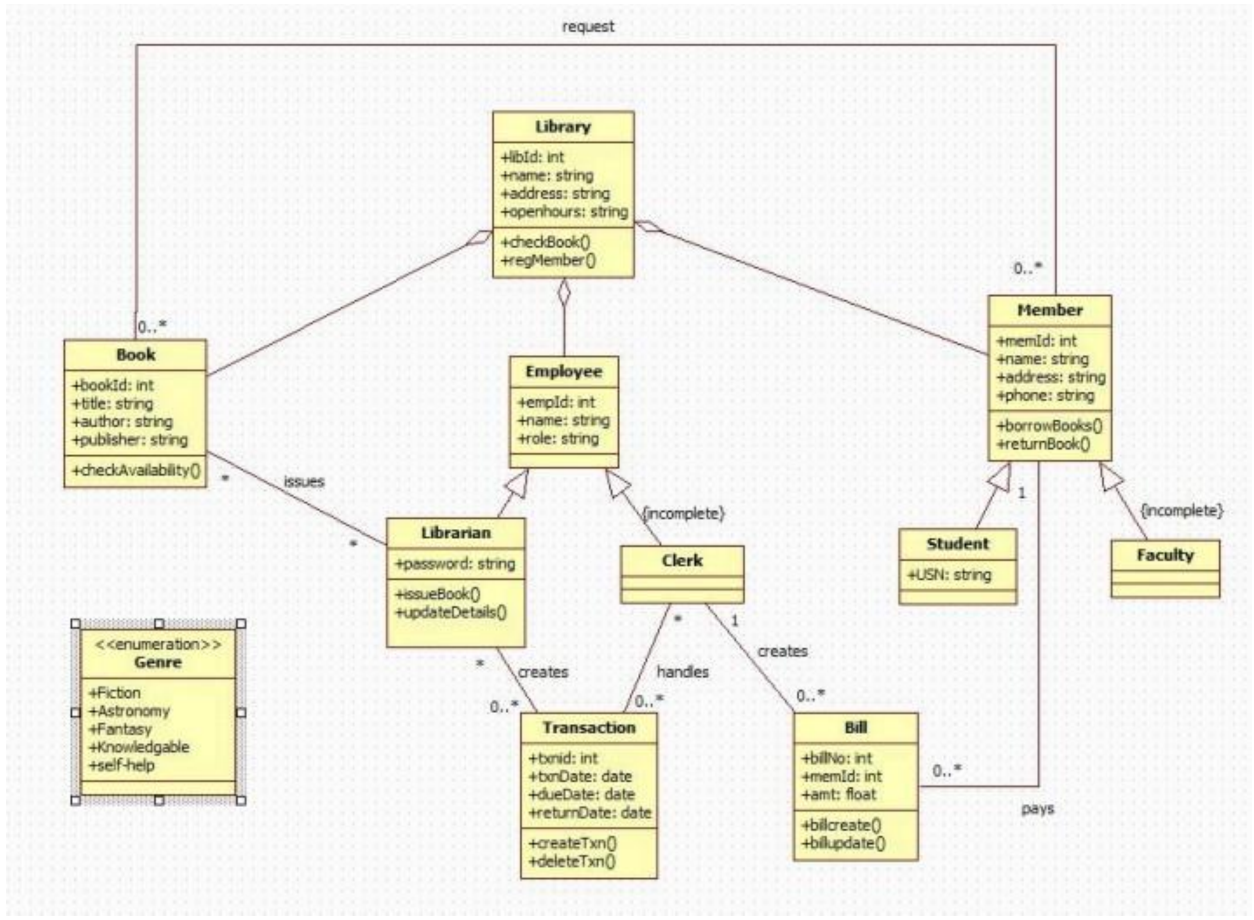


Fig 3.1 Class diagram of Library management system

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:

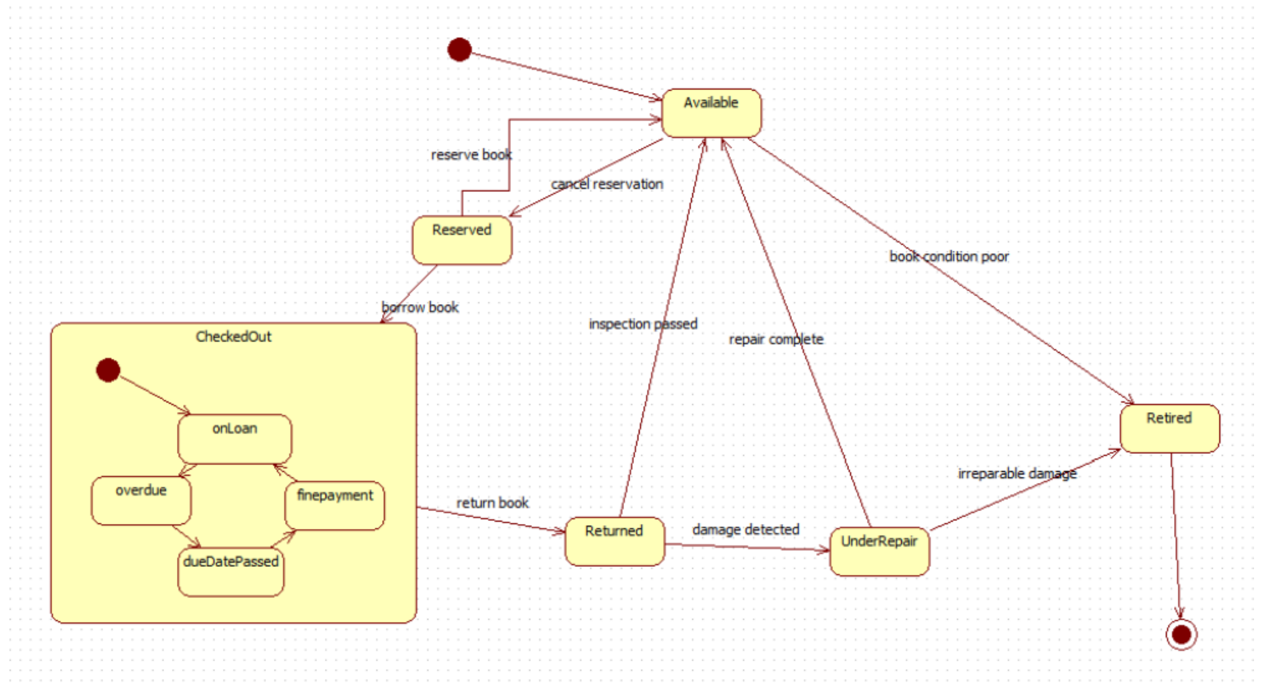


Fig 3.2 State diagram of Library management system

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:

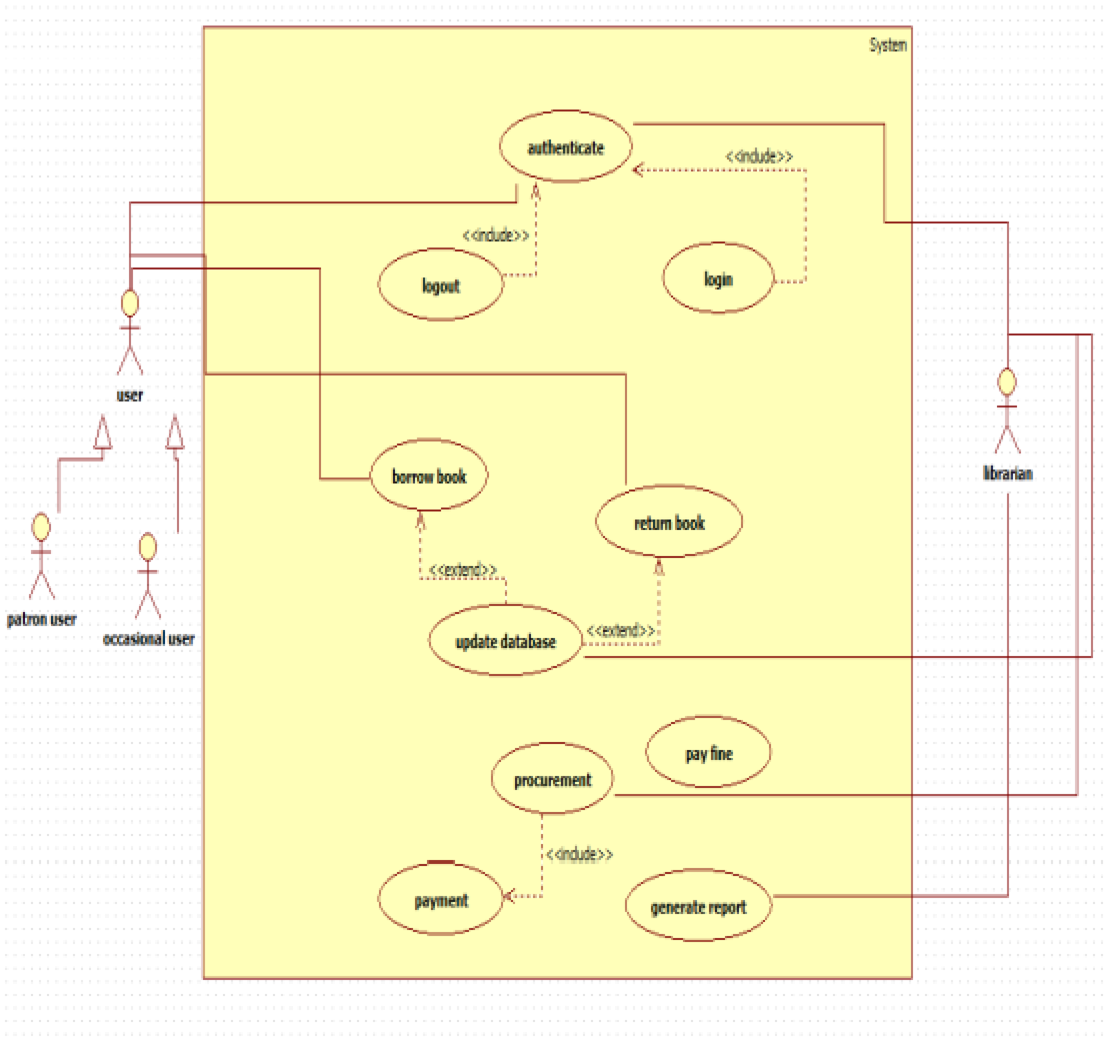


Fig 3.3 Use case diagram of Library management system

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:

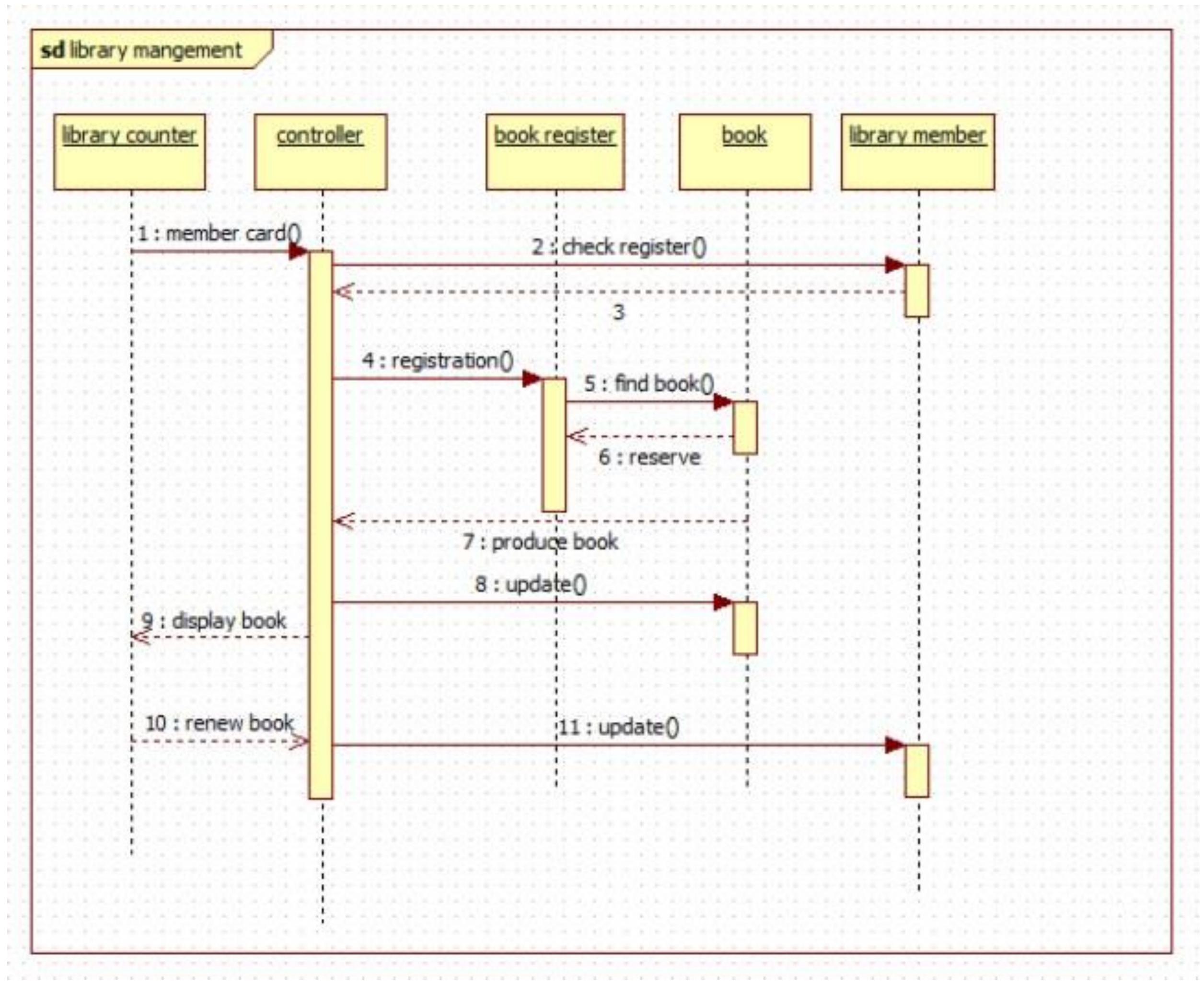


Fig 3.4 Sequence diagram of Library management system

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:

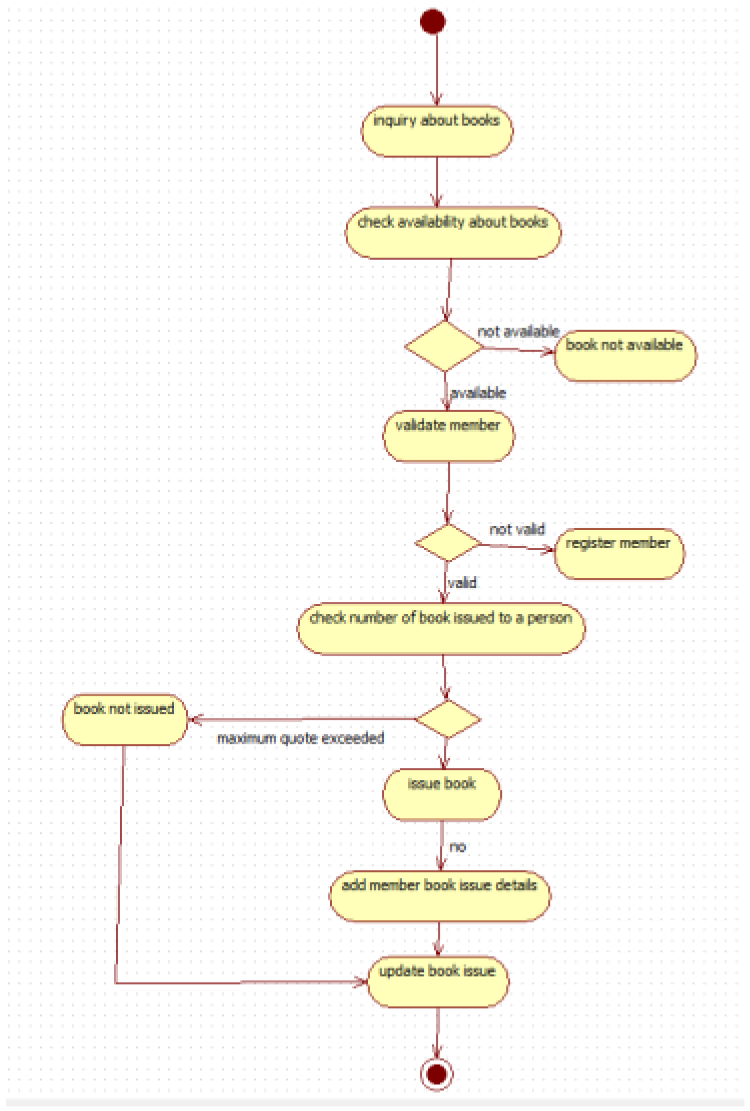


Fig 3.5 Activity diagram of Library management system

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.

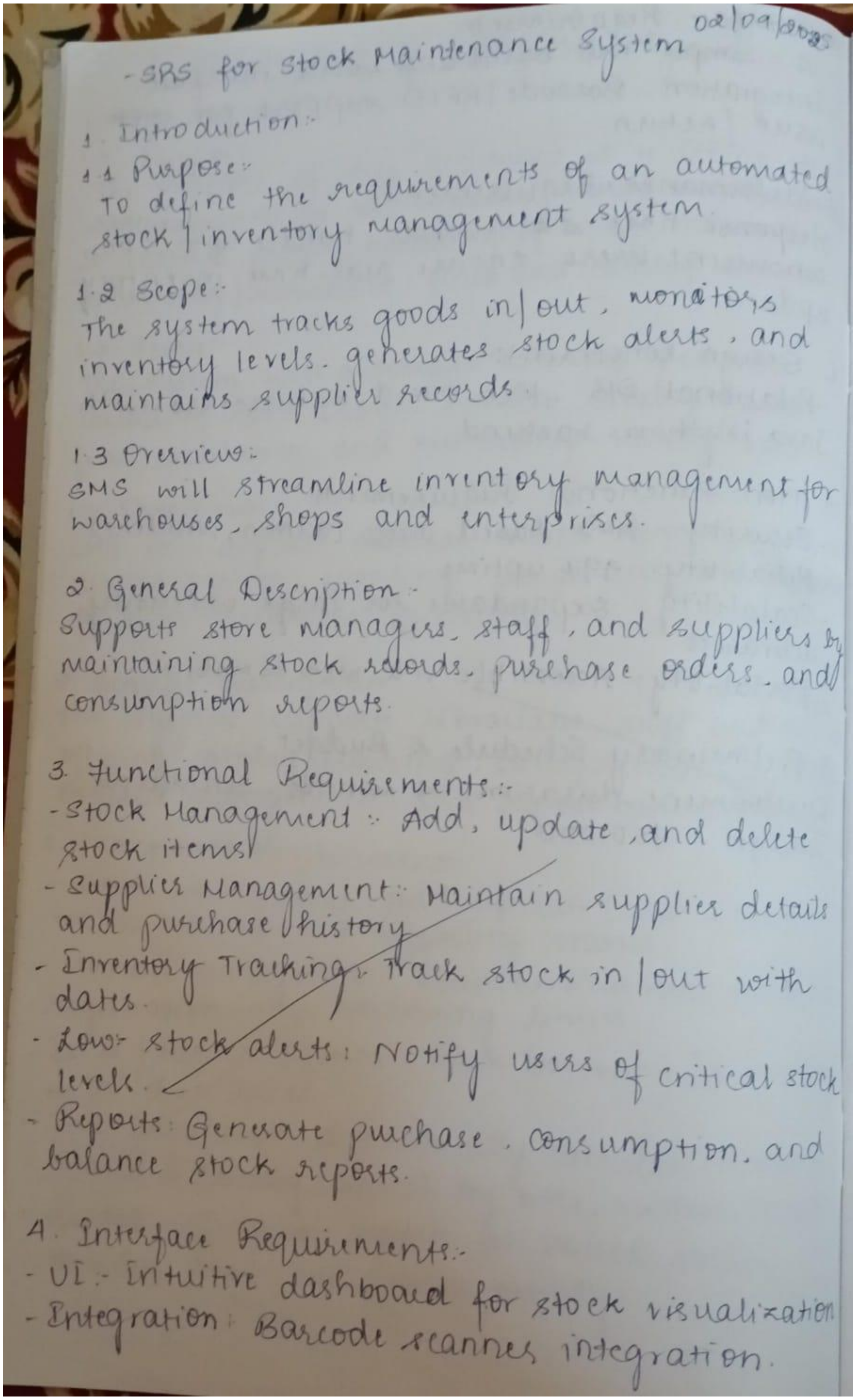
4. Stock Maintenance System

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)



02/09/2025

-SRS for Stock Maintenance System

1. Introduction:-

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

1.2 Scope:-
The system tracks goods in/out, monitors inventory levels, generates stock alerts, and maintains supplier 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 200 concurrent users, Response time ≤ 3 seconds, Real-time update of stock levels.

6. Design Constraints:-
Relational Database (MySQL), Backend can be Java/Spring Boot.

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

8. Preliminary Schedule and Budget:-
Development duration: 5 months
Estimated budget: $75000

Class diagram:

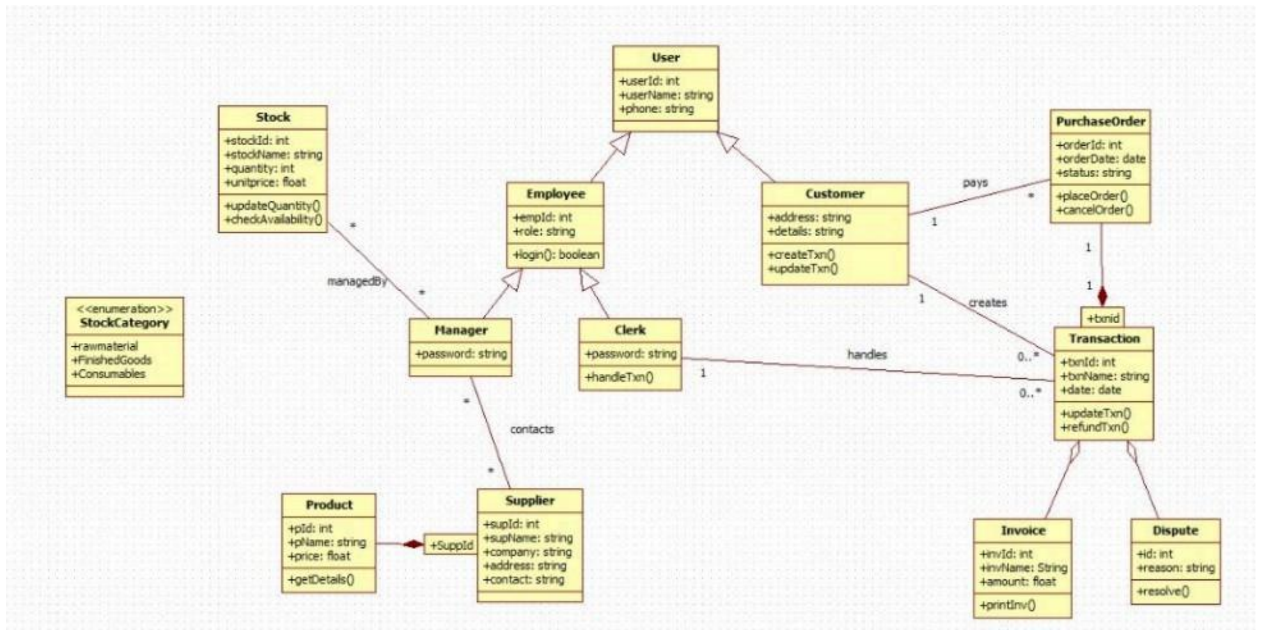


Fig 4.1 Class diagram of Stock management system

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:

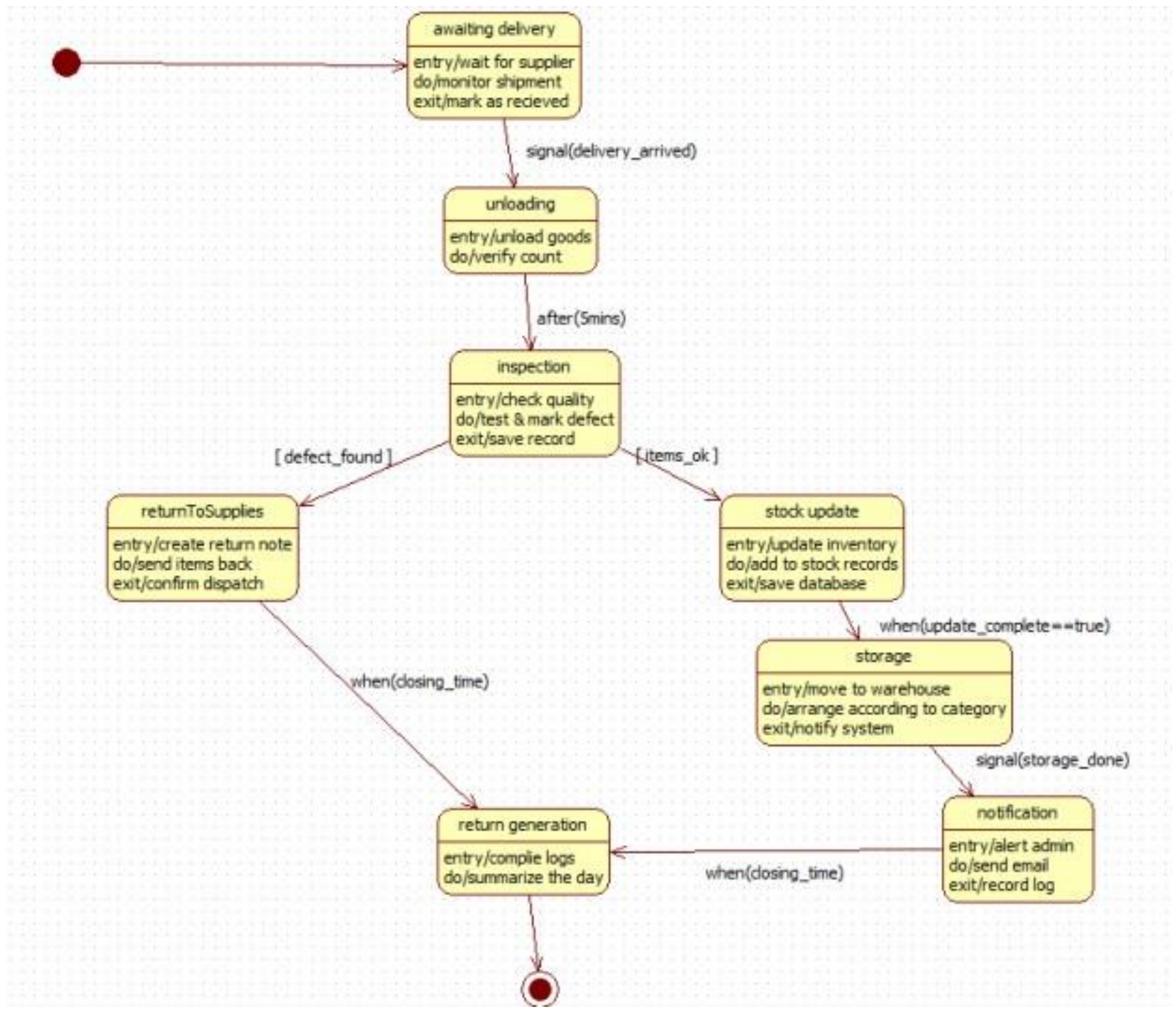


Fig 4.2 State diagram of Stock management system

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:

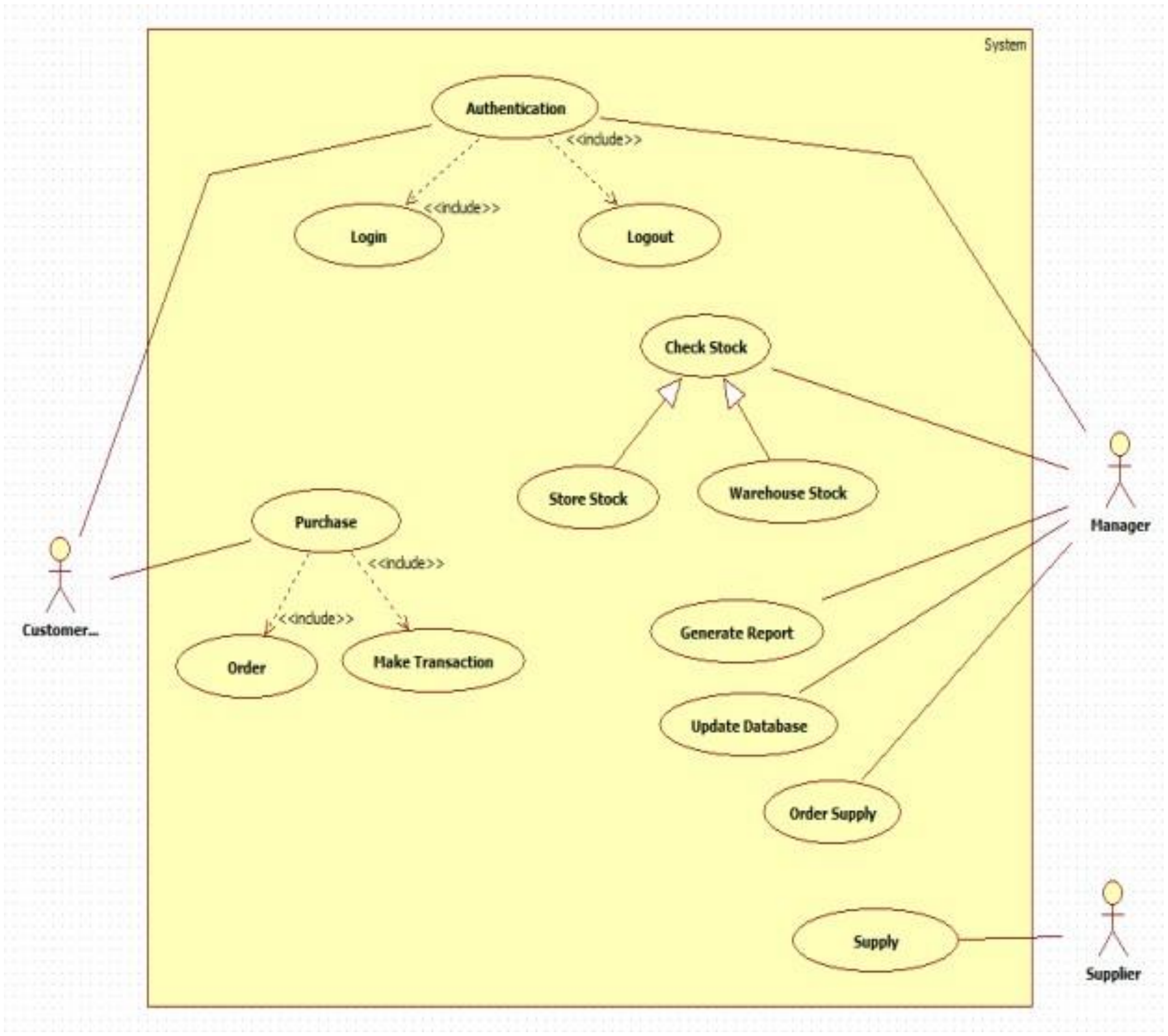


Fig 4.3 Use-case diagram of Stock management system

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:

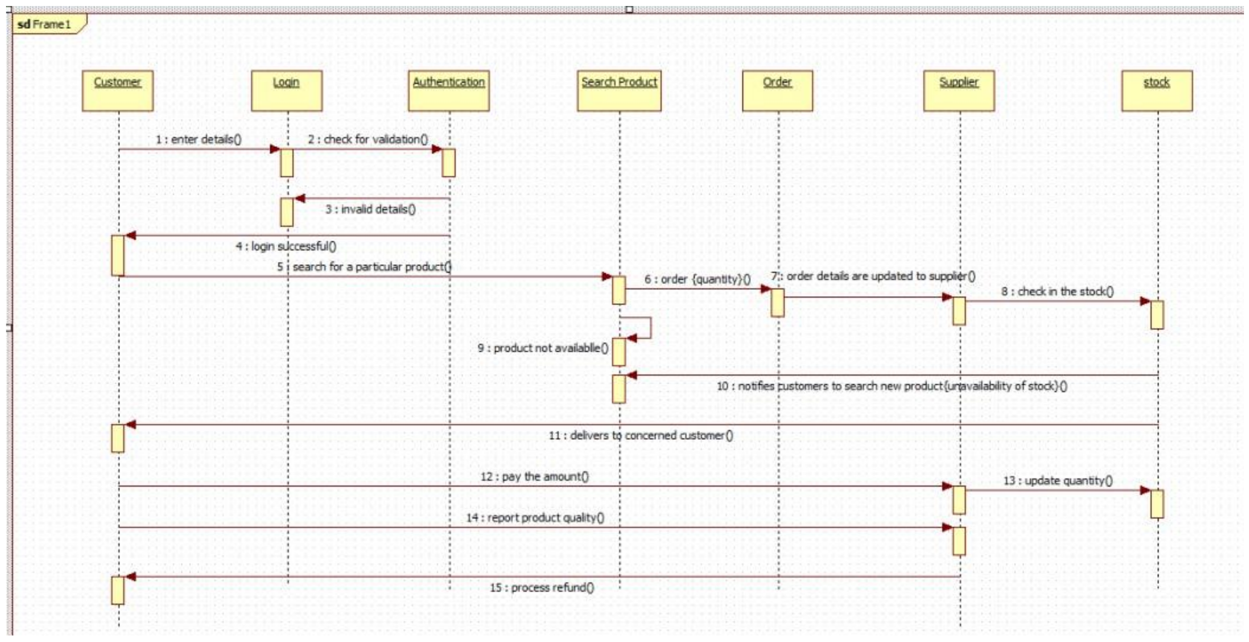

sd Frame1
Customer
Login
Authentication
Search Product
Order
Supplier
stock
1 : enter details()
2 : check for validation()
3 : invalid details()
4 : login successful()
5 : search for a particular product()
6 : order {quantity}()
7 : order details are updated to supplier()
8 : check in the stock()
9 : product not availablle()
10 : notifies customers to search new product{unavailability of stock}()
11 : delivers to concerned customer()
12 : pay the amount()
13 : update quantity()
14 : report product quality()
15 : process refund()

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:

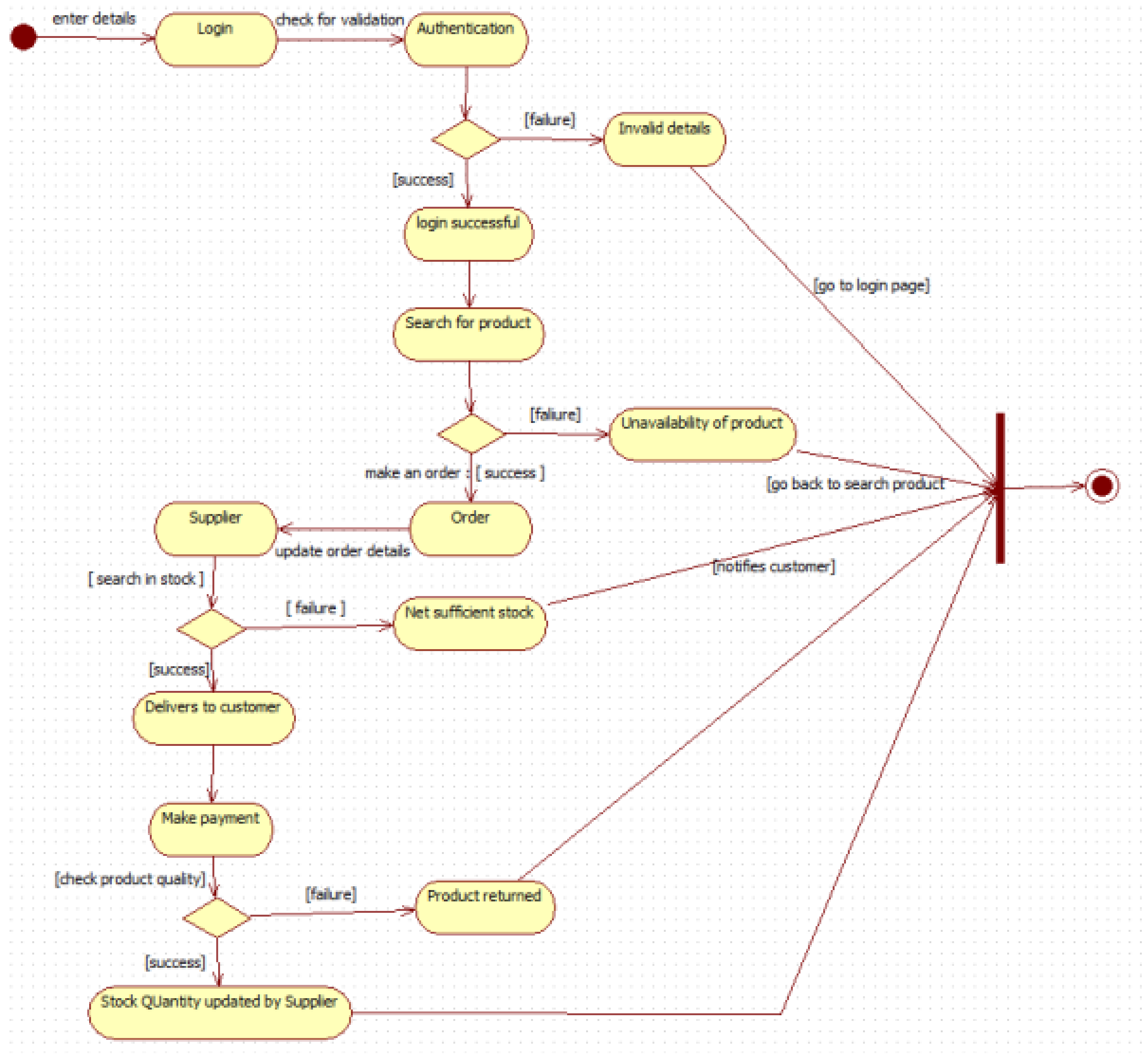


Fig 4.5 Activity diagram of Stock management system

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.

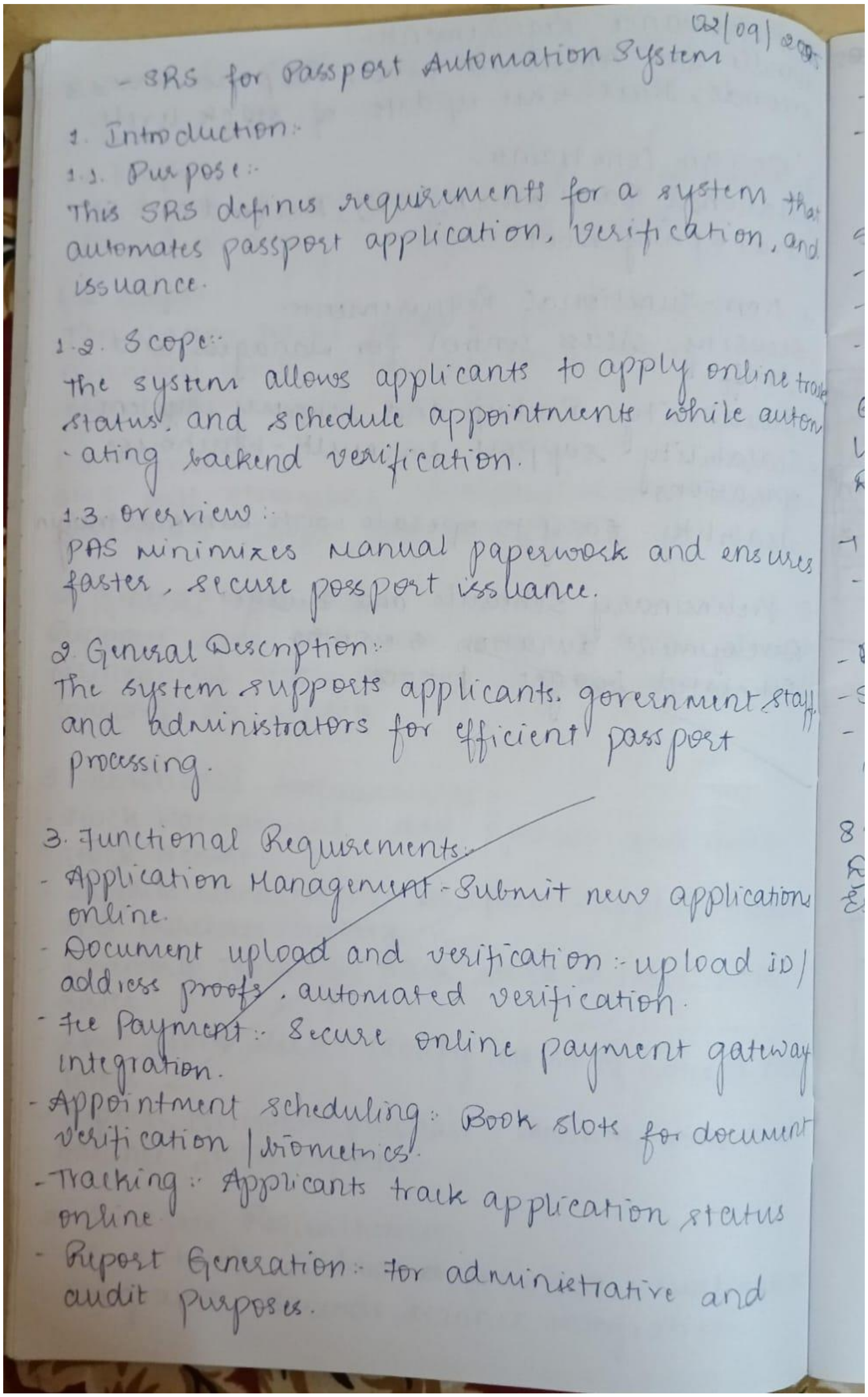
5. Passport Automation System

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)



02/09/20

- 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 trac status, and schedule appointments while autom -ating backend verification.

1.3. Overview:
PAS minimizes manual paperwork and ensures faster, 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 and verification:- upload ID/address proofs, automated verification.
- Fee Payment:- Secure online payment gateway integration.
- 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 and admin dashboard.
- Integration: Biometric devices, payment gateway, police verification system

5. Performance requirements:
- Response time ≤ 2 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 and kiosks.

8. Preliminary schedule and Budget.
Development duration 12 months.
Estimated budget: $500,000.

Class diagram:

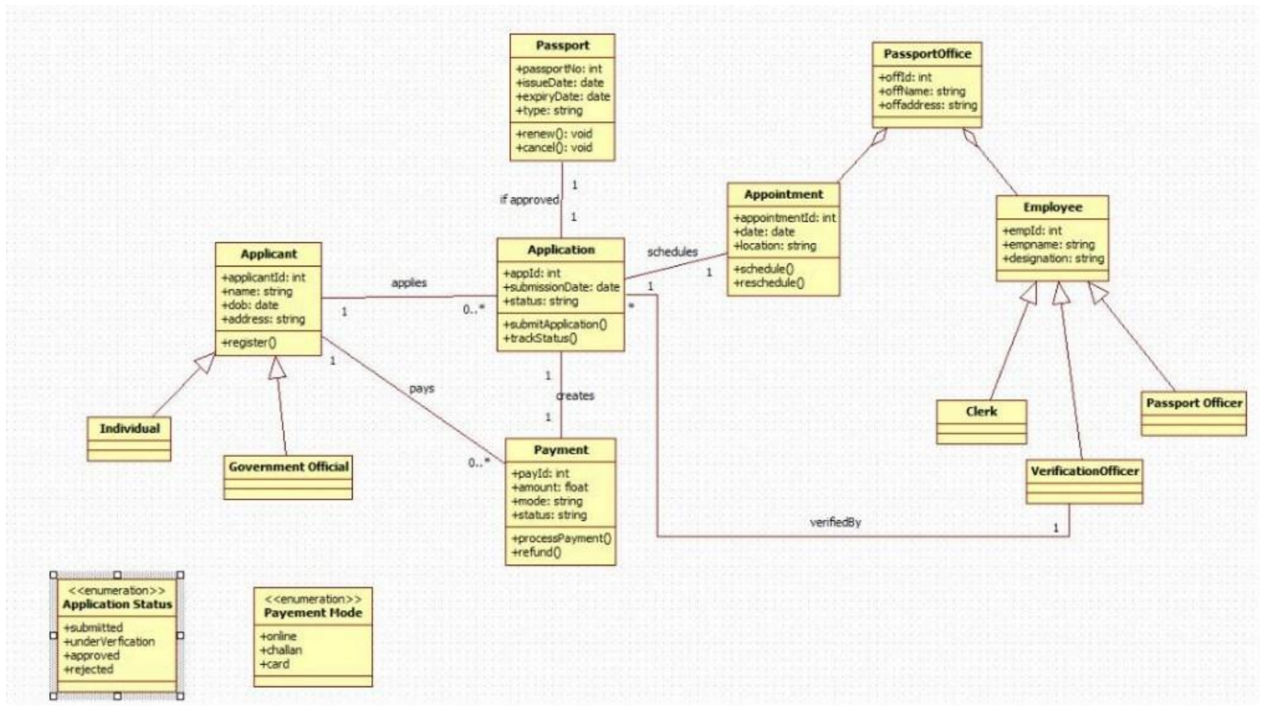


Fig 5.1 Class diagram of Passport Automation System

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:

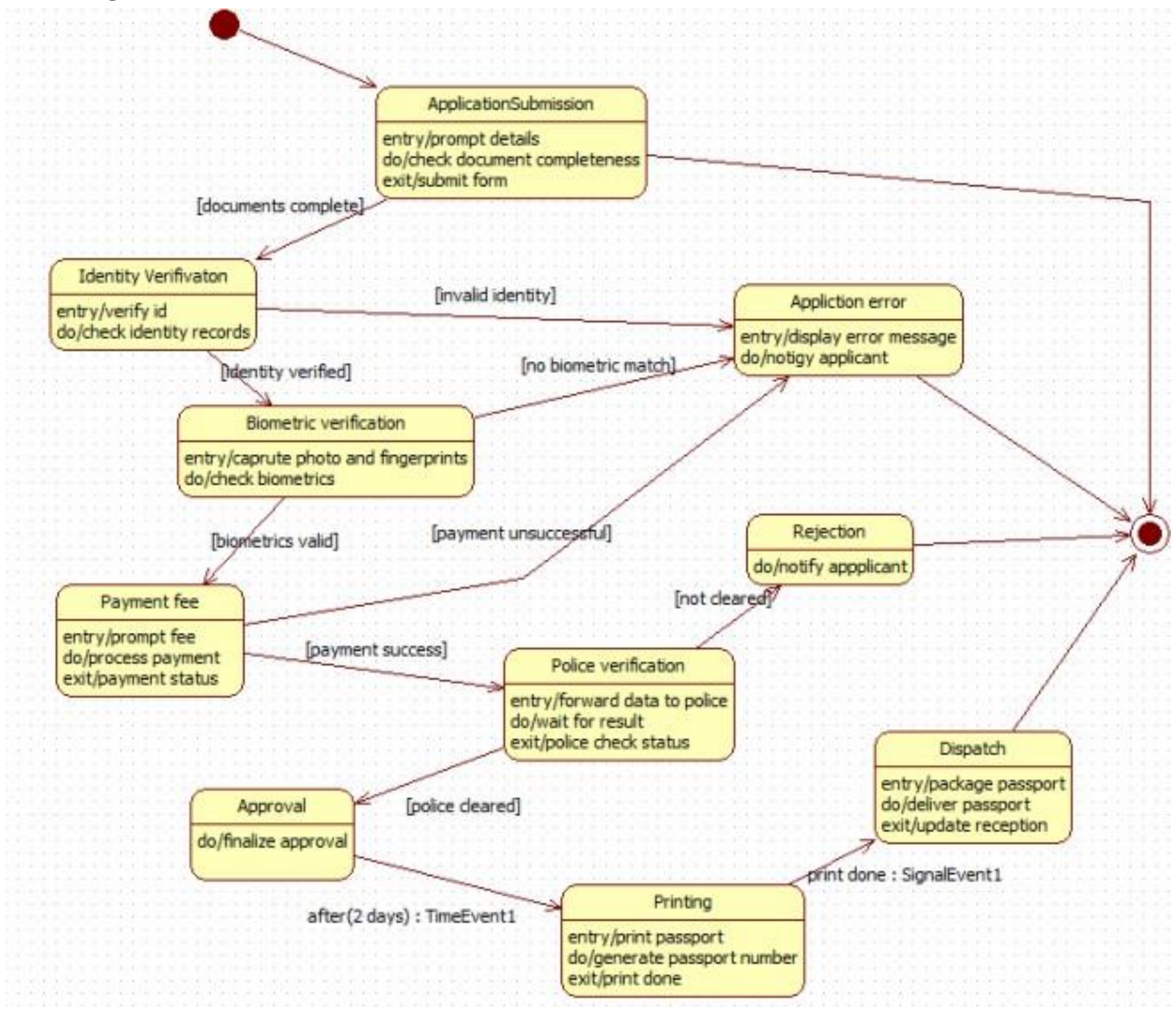


Fig 5.2 State diagram of Passport Automation System

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:

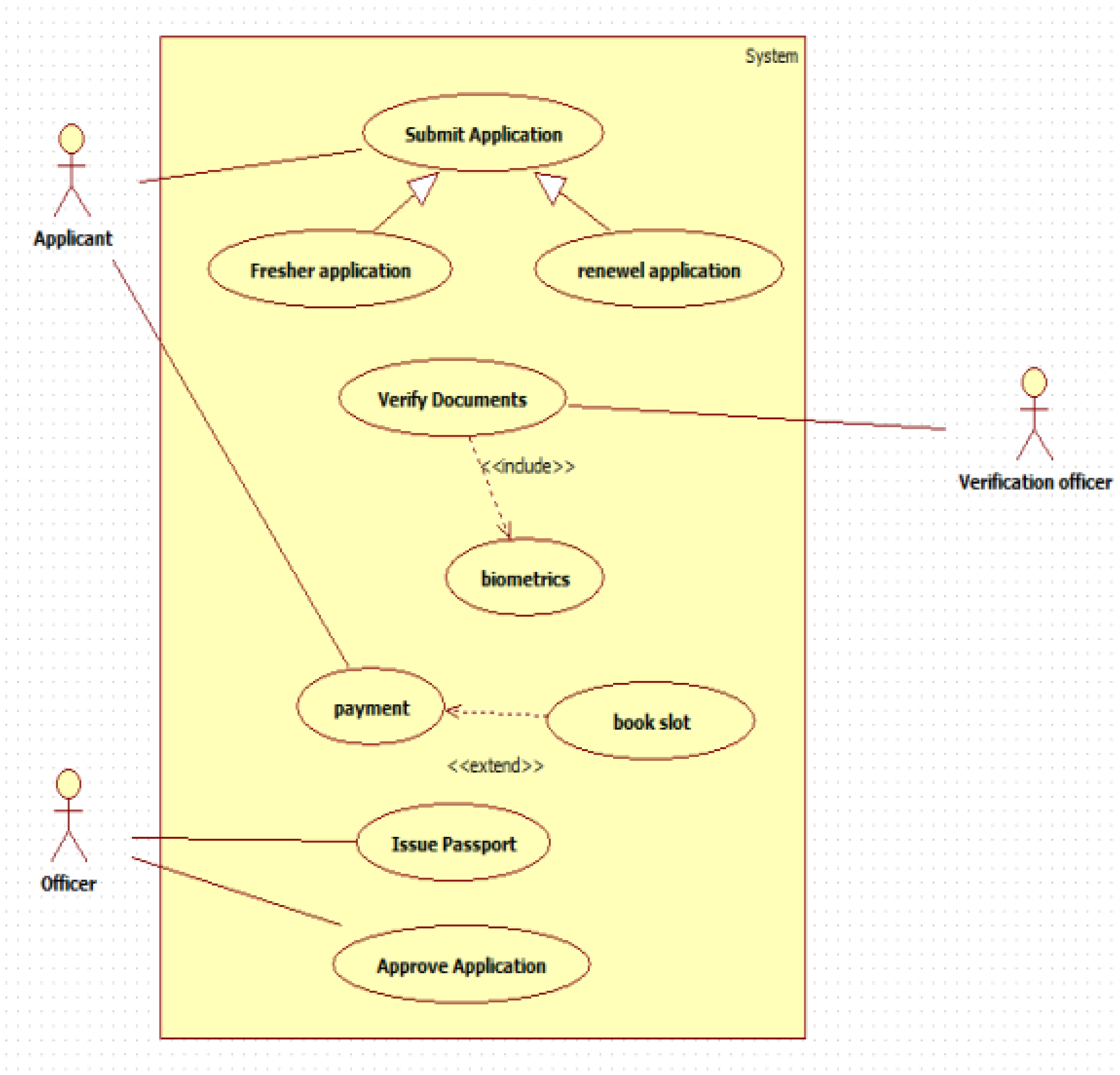


Fig 5.3 Use-case diagram of Passport Automation System

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:

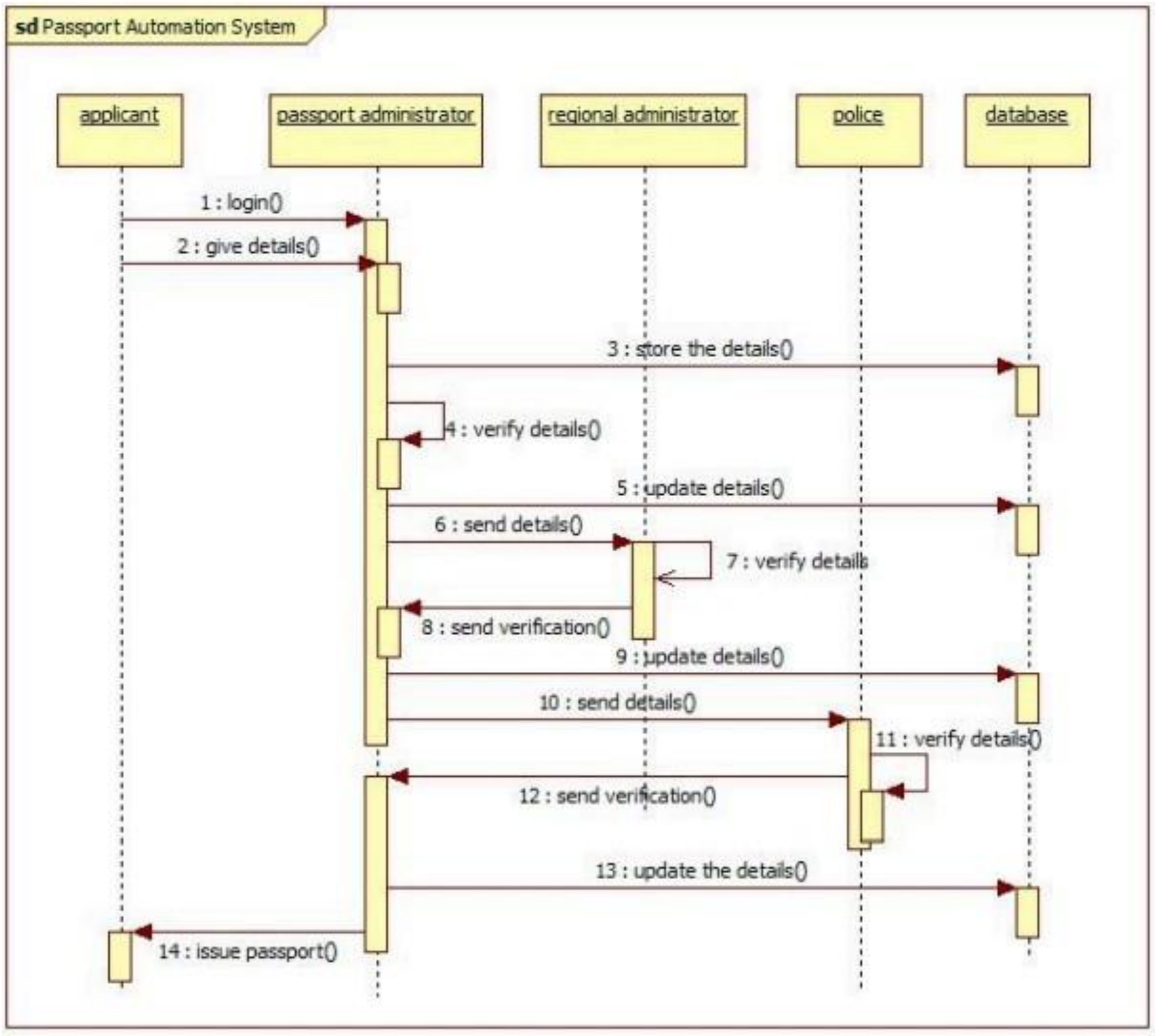


Fig 5.4 Sequence diagram of Passport Automation System

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:

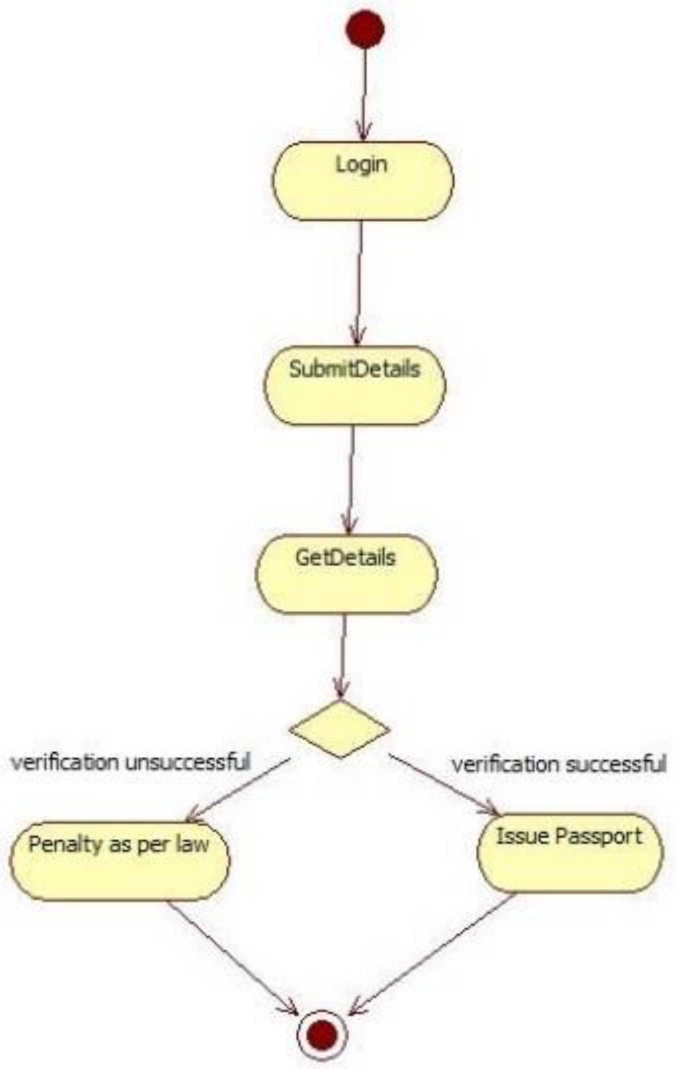


Fig 5.5 Activity diagram of Passport Automation System

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.