

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



**Lab Record**

**Object-Oriented Modeling — 23CS5PCOOM**

*Submitted in partial fulfillment for the 5<sup>th</sup> Semester Laboratory*

Bachelor of Engineering  
in  
Computer Science and Engineering

*Submitted by:*

Chethan K S  
1BM23CS074

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 Chethan K S(1BM23CS074) during the 5<sup>th</sup> Semester August 2025-December 2025

Signature of the Faculty Incharge:

Vikranth B M  
Assistant Professor  
Department of Computer Science and Engineering  
B.M.S. College of Engineering, Bangalore

## Table of Contents

1. Hotel Management System	1-7
2. Credit Card Processing	8-15
3. Library Management System	16-22
4. Stock Maintenance System	23-29
5. Passport Automation System	30-36

# 1. Hotel Management System

## SRS Document

<p>SRS Document.</p> <h3><u>Hotel Management System.</u></h3> <p>1. Introduction</p> <p>1.1 Purpose: To specify requirements for "HotelSphere" Hotel Management System.</p> <p>1.2 Document Conventions</p> <p>→ Following the IEEE SRS Standard.</p> <p>1.3 Audience</p> <p>→ Intended for project team (managers, developers) and stakeholders (hotel staff) to understand the project's scope and requirements.</p> <p>1.4 Product Scope</p> <p>→ Web-based system for managing reservations, front desk operations, room service, and billing.</p> <p>1.5 References</p> <p>→ IEEE std → Client Requirements and project</p> <p>2 Overall Description</p> <p>2.1 Product Perspective</p> <p>→ A stand-alone system that integrates with third-party APIs from Online Travel Agencies (OTAs) and payment gateways.</p> <p>2.2 Product Functions</p> <p>→ Manages reservations, check-in/check-out, room status, billing and reporting.</p> <p>2.3 User Classes and Characteristics</p> <p>→ Front Desk Staff: Manages daily operations; needs a simple interface. → Housekeeping Staff: Updates room status; needs a mobile friendly interface.</p>	<p>→ Hotel Manager: Oversees all operations; needs comprehensive reports.</p> <p>→ Guests: Views bookings and invoices; needs a secure, easy-to-use interface.</p> <p>2.4 Operating Environment</p> <p>→ Client-server environment</p> <p>→ Server runs on Windows and a database.</p> <p>→ Client - web-browser.</p> <p>2.5 Design and Implementation Constraints</p> <p>→ Must comply with hotel data privacy standards</p> <p>→ Limited to the capacity of hotel hardware infrastructure.</p> <p>2.6 User Documentation</p> <p>→ User manuals for staff and admin → Online help documentation.</p> <p>2.7 Assumptions and Dependencies</p> <p>→ Reliable internet connection (if online booking) → Availability of database server.</p> <p><u>3 Requirements.</u></p> <p>3.1 User Interface</p> <p>→ Simple, intuitive UI with menus for bookings, check-in, billing and reports.</p> <p>3.2 Hardware Interfaces</p> <p>→ PC or server with database connectivity. → Printers for receipts and bills.</p> <p>3.3 Software Interfaces</p> <p>→ Database: MySQL → Payment gateway integration.</p> <p><u>4 System Features</u></p> <p>→ Room Booking → Customer management → Billing and payments → Staff management</p>
--	---

## 5 Other Non-functional Requirements

### 5.1. Performance Requirements

- Support up to 100 concurrent users
- Booking transactions should complete within 2 seconds

### 5.2. Safety Requirements

- Regular Data backups
- Fail-safe mechanisms in case of server crash.

### 5.3. Security Requirements

- Role-based access control
- Encrypted storage of sensitive data

### 5.4. Software Quality Attributes

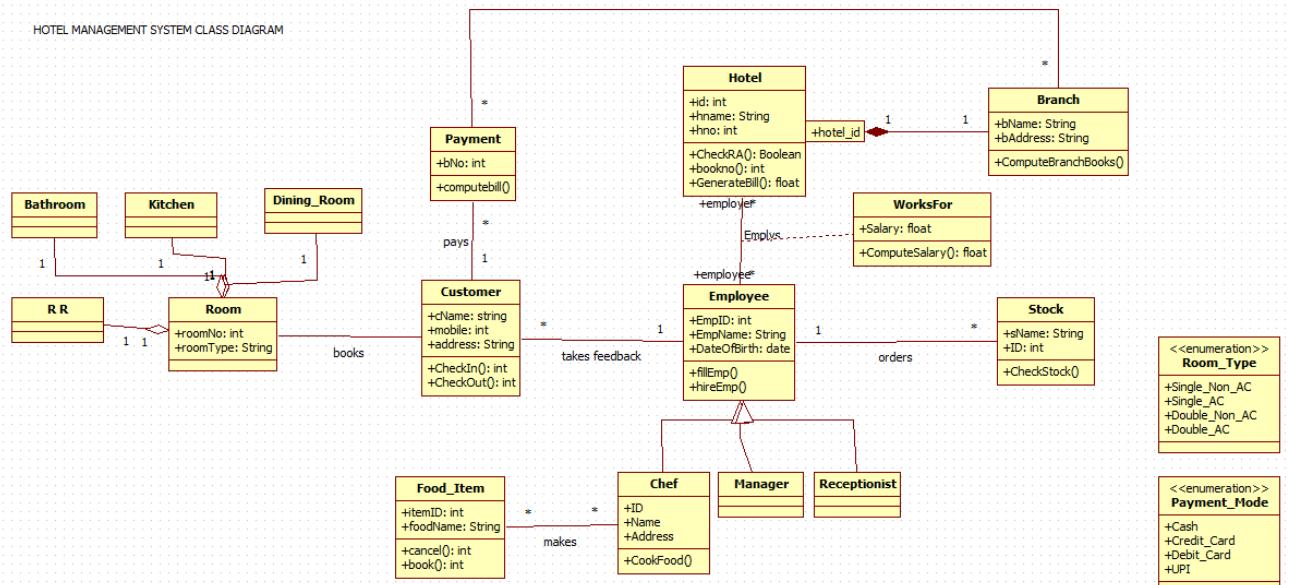
- Usability: Easy-to-use interface

## 6 Other Requirements

- Compliance with local hotel laws and taxation policies

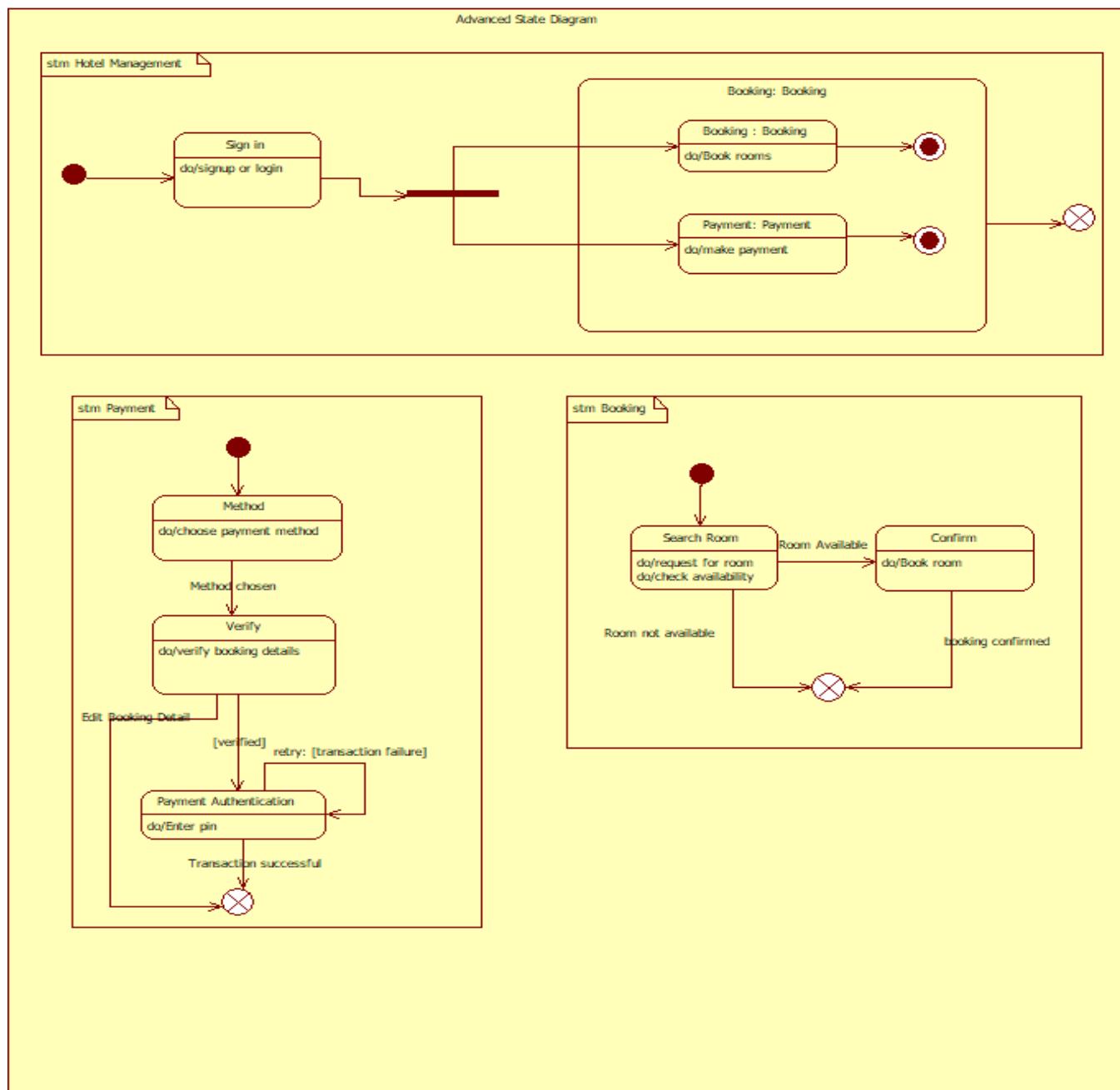
- Multi-language support for international customers

## Class Diagram:



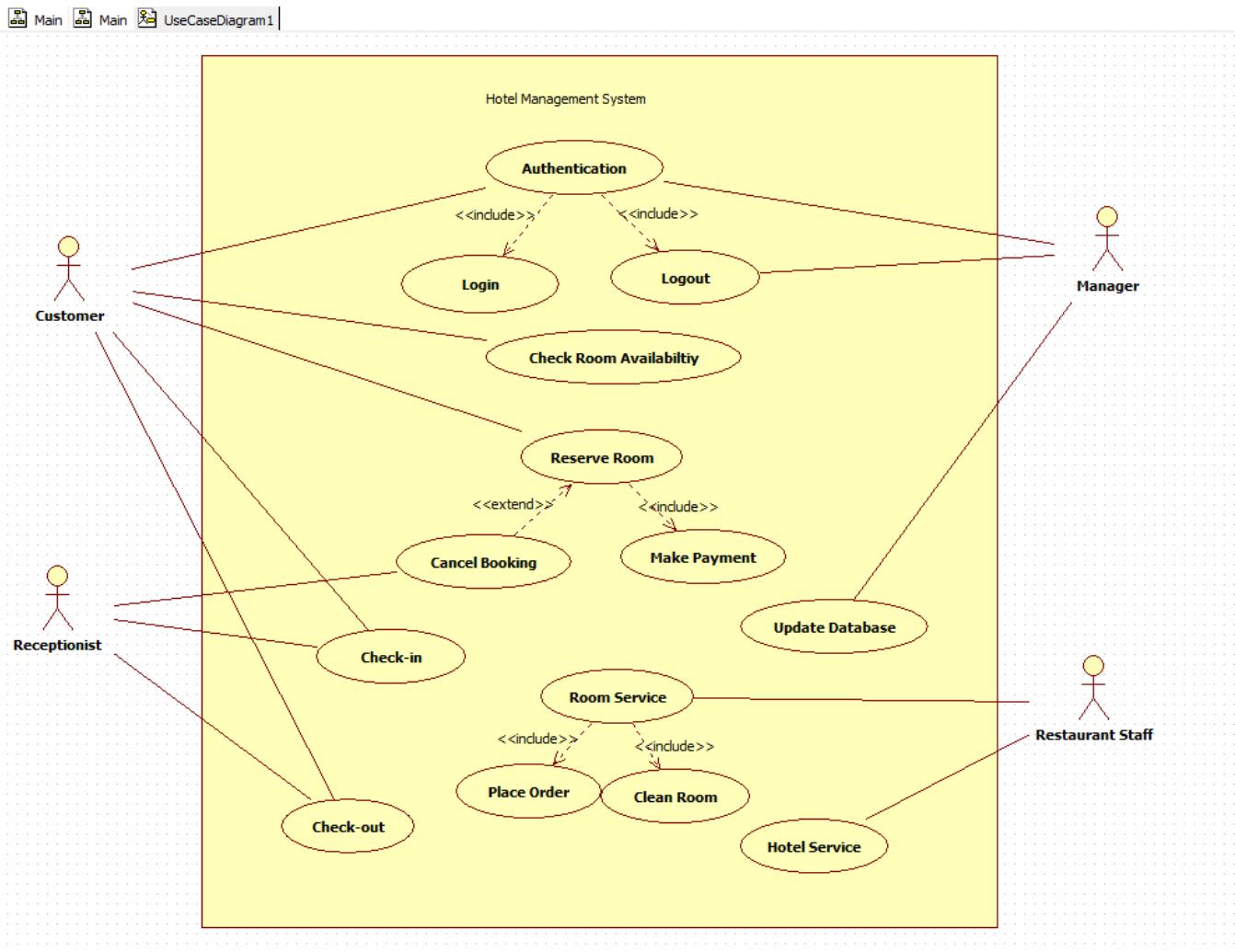
The class diagram for a Hotel Management System outlines the system's architecture by specifying key classes—Hotel, Branch, Customer, Room, Employee, Payment, FoodItem, and Stock—along with their attributes, behaviors, and detailed interconnections. A hotel oversees several branches, each with its own characteristics, while customers can reserve different room categories, such as standard rooms or suites, and complete payments securely. Employees are divided into roles like manager, chef, or receptionist, which supports efficient hotel operations, guest services, and food preparation. Additional elements monitor food items and inventory, creating a cohesive approach to stock control and service delivery. Enumerations for room categories and payment methods help standardize and automate booking and transaction processes. By representing both inheritance (for example, specialized room types and employee subclasses) and composition links, the diagram provides a solid and flexible framework for managing hotel activities effectively

## State Diagram:



The diagrams illustrate the workflow of a Hotel Management System by showing how users interact with booking and payment processes. After signing up or logging in, the system splits into two parallel activities: booking a room and making a payment. In the booking process, users search for available rooms and either confirm the booking if a room is available or exit the process if not. The payment process involves selecting a payment method, verifying booking details, and authenticating the payment, with options to retry if verification or the transaction fails. Each process concludes independently, and together they represent a complete and coordinated hotel booking and payment workflow.

## Use Case:



The advanced use case diagram shows a more detailed view of how hotel staff and users interact with the system. It includes both Admin and User roles along with all supporting processes. The Admin can make reservations, view bookings, handle check-ins and check-outs, calculate bills, and process payments. These actions may involve additional steps such as choosing room types, selecting customers, filtering available rooms, or finalizing a reservation. The User can create customer profiles, check available rooms, view customer lists, and select reservations. These use cases support the admin's work and help the system run smoothly. This diagram gives a clearer picture of the system's internal workflow, showing how different tasks are connected and how the hotel manages the entire reservation and payment process.

## Scenarios

### 1: Book Room

The customer opens the hotel booking page.

- The customer selects the desired room type (Single, Double, AC, Non-AC, etc.).

The system filters and displays all available rooms of that type.

- The customer chooses one room from the list.

The customer enters personal details (name, contact number, ID proof).

The system verifies the entered information.

- The customer confirms the booking.

The system generates a booking confirmation and stores reservation details.

### 2: Process Payment

The customer selects the “Make Payment” option.

The system displays the total bill amount.

The customer chooses a payment method (UPI, Card, Net Banking, etc.).

The system redirects the request to the external payment gateway.

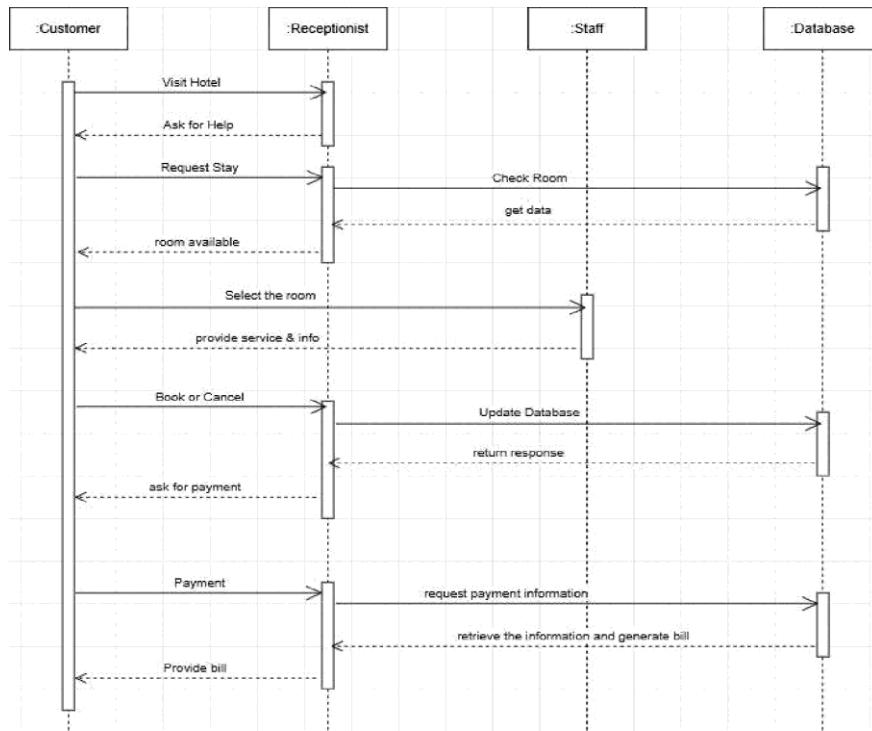
The customer enters required payment details.

The payment gateway verifies the details and processes the transaction.

On success, the gateway sends a payment success message to the hotel system.

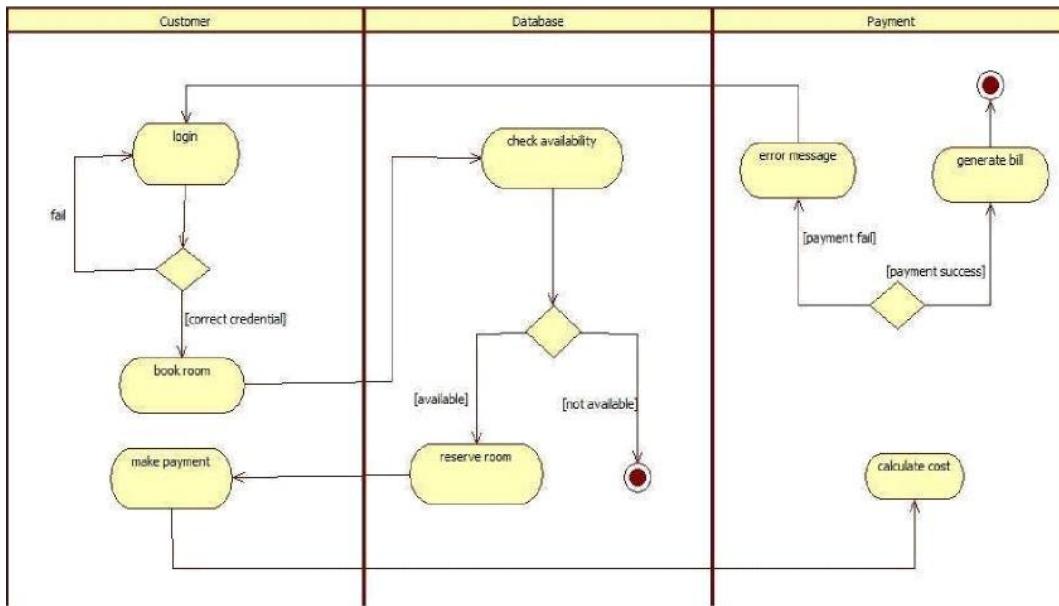
The system marks the booking as “Paid” and generates a payment receipt.

## Advanced Sequence Diagram (Customer—Receptionist—Staff—Database)



The advanced sequence diagram provides a more complete picture of the hotel booking process. It starts when the Customer approaches the Receptionist for help. The customer requests a room, and the receptionist forwards this request to the Staff, who checks room availability by interacting with the Database. The database returns the room information, and the receptionist shares it with the customer. The customer selects a room, and the receptionist provides the required details or services. If the customer decides to book or cancel, the receptionist updates the Database, which stores the new booking status and returns confirmation. After the booking is confirmed, the receptionist asks the customer for payment. The receptionist retrieves payment details from the database and prepares the final bill for the customer. This advanced diagram shows the full workflow—from request to billing—highlighting the roles of staff and database operations behind the scenes.

## Advanced Activity Diagram



The advanced activity diagram shows the same process in more detail and separates the actions into three swimlanes: Customer, Database, and Payment.

- In the Customer section, the user logs in, books a room, and makes the payment.
- In the Database section, the system checks if rooms are available and reserves the room when possible.
- In the Payment section, the system calculates the total amount, processes the payment, and generates the bill.

The advanced diagram also includes decision points such as incorrect login, room not available, and payment failure. It presents a clearer and more complete picture of how different parts of the hotel system work together.

## 2. Credit Card Processing

### SRS Document

<h3><u>II Credit Card Processing System</u></h3> <h4><u>1. Introduction</u></h4> <h5><u>1.1 Purpose</u></h5> <p>The credit card processing system (CCPS) is designed to securely authorize, authenticate, and settle credit card transactions for merchants and customers. It ensures accuracy, fraud prevention, and compliance with financial regulations.</p> <h5><u>1.2 Document Conventions</u></h5> <ul style="list-style-type: none"><li>CCPS = Credit Card Processing System</li><li>POS = Point of Sale</li><li>OTP = One-Time Password</li></ul> <h5><u>1.3 Intended audience and reading suggestions</u></h5> <ul style="list-style-type: none"><li>Merchants: For accepting card payments</li><li>Banks/Payment Gateways: For validating and settling transactions</li><li>Developers/Testers: For building and validating the system.</li><li>Admins: For monitoring and managing system operations.</li></ul> <h5><u>1.4 Product Scope</u></h5> <p>The CCPS provides a secure and reliable platform to handle card transactions in real time. It supports authorization, authentication, fraud detection, billing, and settlement between banks and merchants.</p> <h5><u>1.5 References</u></h5> <ul style="list-style-type: none"><li>PCI DSS Security Standards</li><li>ISO 8583 Financial Messaging Standard</li></ul>	<h4><u>2. Overall Description</u></h4> <h5><u>2.1 Product Perspective</u></h5> <p>The CCPS works as middleware between merchants, customer banks and payment gateways. It integrates with POS devices, ATMs and online payment portals.</p> <h5><u>2.2 Product Functions</u></h5> <ul style="list-style-type: none"><li>Transaction authorization and validation.</li><li>Customer authentication (PIN/OTP)</li><li>Settlement between <del>bank</del> bank and merchant</li><li>Reporting and fraud detection.</li></ul> <h5><u>2.3 User Classes and Characteristics</u></h5> <ul style="list-style-type: none"><li>Customer: Initiates payment with card details</li><li>Merchant: Accepts payments via POS/online.</li><li>Admin/Bank Staff: Monitors, verifies, and manages transactions</li></ul> <h5><u>2.4 Operating Environment</u></h5> <ul style="list-style-type: none"><li>Secure servers with 24/7 availability</li><li>POS terminals, ATMs and web applications</li><li>Encrypted communication channels</li></ul> <h5><u>2.5 Design and implementation constraints</u></h5> <ul style="list-style-type: none"><li>Must comply with PCI DSS and local regulations</li></ul> <h5><u>2.6 User Documentation</u></h5> <ul style="list-style-type: none"><li>Merchant user guide</li><li>Admin manuals</li><li>API documentation for integration.</li></ul> <h5><u>2.7 Assumptions and Dependencies</u></h5> <ul style="list-style-type: none"><li>Continuous internet connectivity</li><li>Support from banks and payment networks</li></ul> <h3><u>3 External Interface Requirements</u></h3> <h4><u>3.1 User interface</u></h4> <ul style="list-style-type: none"><li>POS interface for merchants</li><li>Customer card input + PIN/OTP screen.</li></ul>
--	--

- Admin dashboard for monitoring

### 3.2. Hardware Interfaces

- Card readers, POS machines, ATMs

### 3.3 Software Interfaces

- Bank APIs
- Payment gateway integration
- Fraud detection modules

### 3.4 Communications Interface

- Secure encrypted Internet channels
- Standard ISO 2583 message format.

## 4 System Features

↳ Authorization - validates card details and account balance.

→ Authentication - PIN or OTP

→ Reporting.

## 5 Other Nonfunctional Requirements

### 5.1 Performance Requirements

- Handle up to 5000 transactions per second
- Process each transaction in less than 3 seconds

### 5.2 Safety Requirements

- Backup and recovery mechanisms
- Rollback for incomplete transactions

### 5.3 Security Requirements

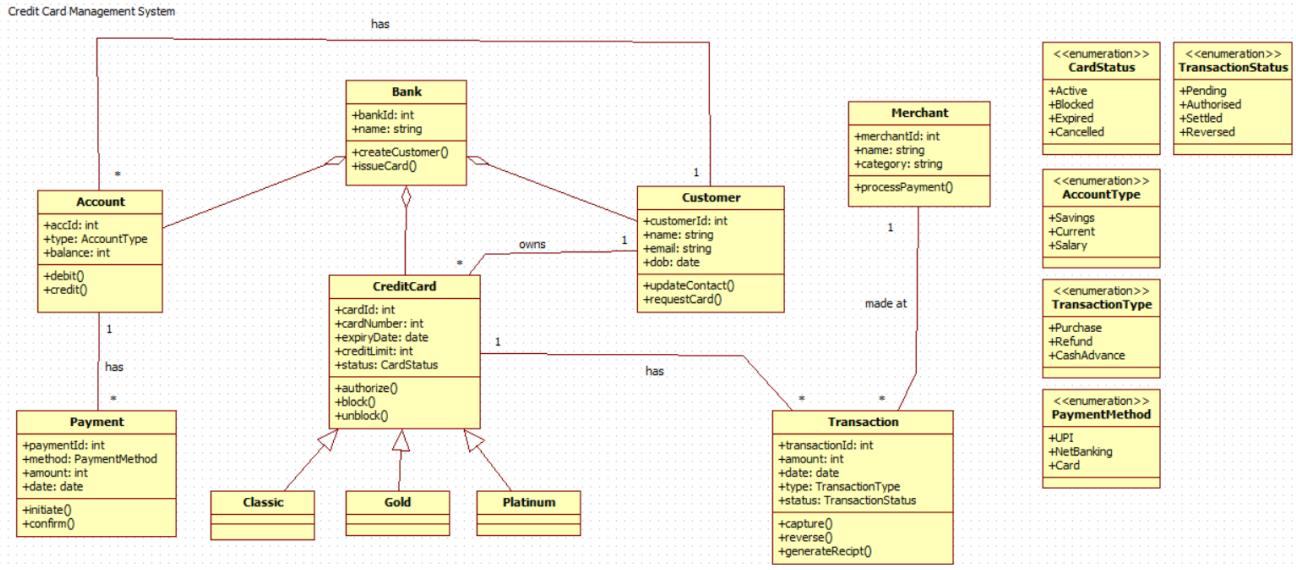
- Strong encryption (AES, RSA)
- Role-based access control.
- PCI DSS compliance.

## 6 Other Requirements

- Support for Visa, MasterCard, RuPay, and other networks
- Multi-currency support for global usage.

✓ 13/8/2025

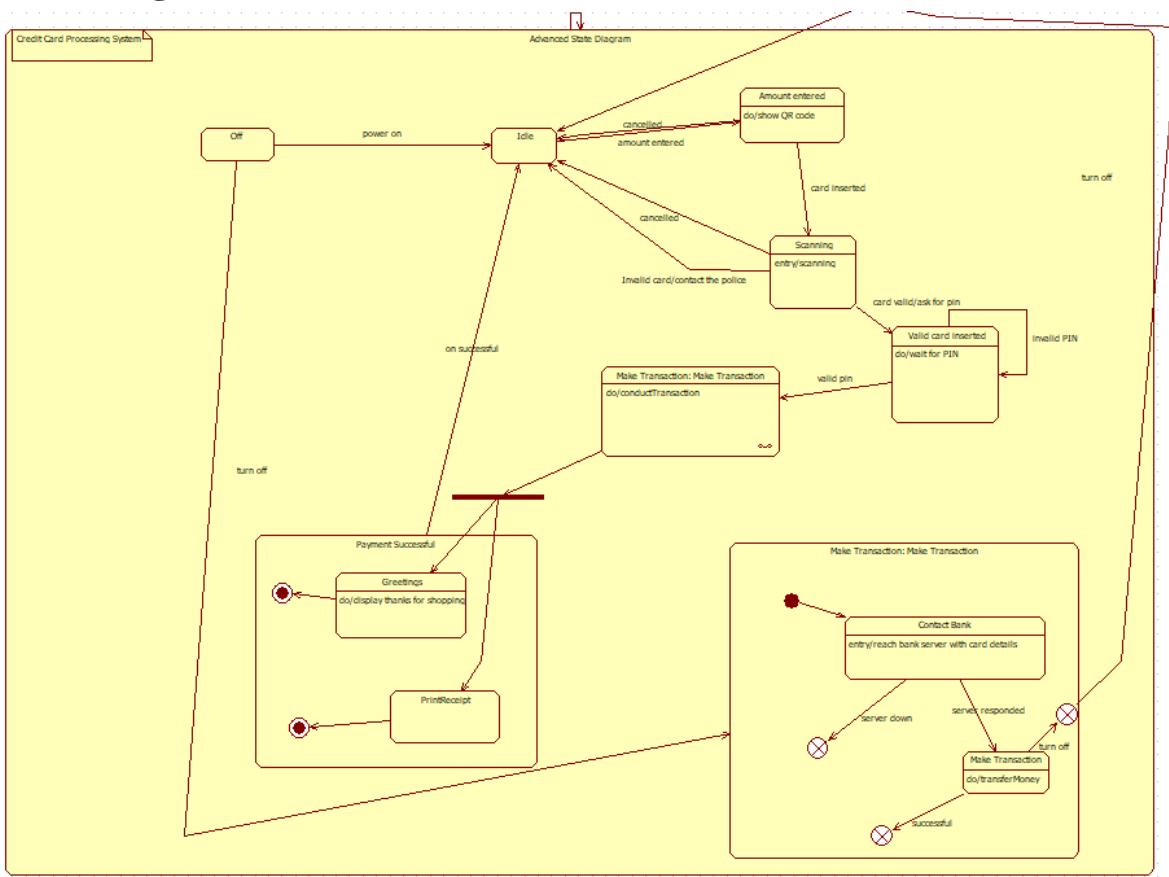
# Class Diagram



## Explanation

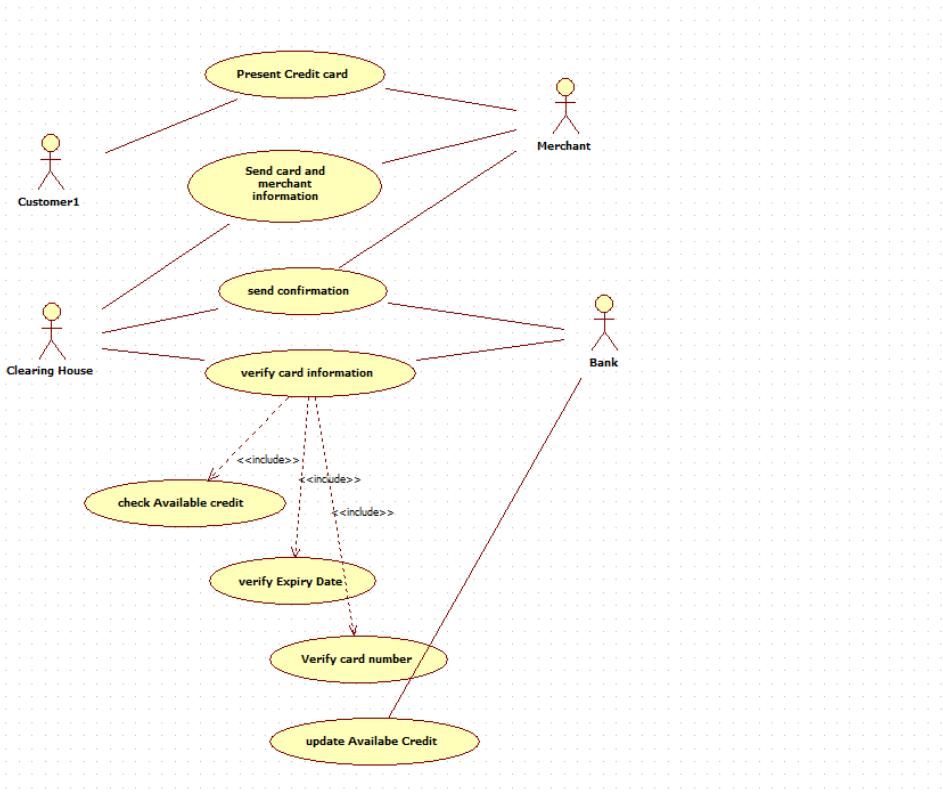
The class diagram represents an online payment processing system that manages customers, merchants, banks, and transactions. The Customer initiates payments using their card, which is linked to a Bank Account. A Transaction is created for each payment, containing details such as amount, date, status, and authorization code. The Bank authorizes and settles payments, while the PaymentGateway facilitates communication between the customer's bank and the merchant. The Merchant receives payments, issues refunds, and handles chargebacks. Additional classes such as Authorization and Refund manage fraud checks, verification, and money returns. Supporting entities like Person, Payment, and the status enumeration define basic attributes and types used throughout the system. Overall, the diagram shows how different components work together to process, validate, and complete electronic payment transactions securely.

## State Diagram(Advanced)



The advanced state diagram describes a more detailed and realistic payment or ATM terminal process. It begins in an Idle state, where the terminal waits for input. A user either enters an amount (for QR payment) or inserts a card. The system validates the card through a scan process. If the card is valid, the terminal asks for the PIN and moves to the Valid Card Inserted state. After the correct PIN is entered, the process moves to Make Transaction, where the transaction is conducted and sent to the bank for approval. The Control Bank state checks card details and verifies the transaction. If the bank responds successfully, the terminal processes the payment and shows a greeting message, followed by printing a receipt. The diagram also includes multiple exceptional paths such as invalid PIN, invalid card, server down, or cancelled actions. This advanced diagram shows a complete, realistic workflow including scanning, PIN verification, bank communication, success/failure handling, and receipt printing.

## Use Case (Advanced)



The advanced use case diagram presents a much more detailed view of how users and administrators interact with a banking system. Multiple actors participate: the Admin, Customer (User), Bank Server, and Database. The system supports a wider set of processes such as Creating Customer, Assigning Customer ID, Updating Account, Crediting and Debiting Accounts, Viewing Statements and Account History, Transactions, Login/Logout, and secure processes like Encrypting and Decrypting Messages. Many relationships use *include* and *extend*, showing how smaller actions support larger processes. This diagram provides a complete picture of how customer accounts are managed, how transactions flow through the system, and how security is maintained. It shows internal system behavior more clearly than the simple diagram.

## **Scenarios**

### **1: Make Payment**

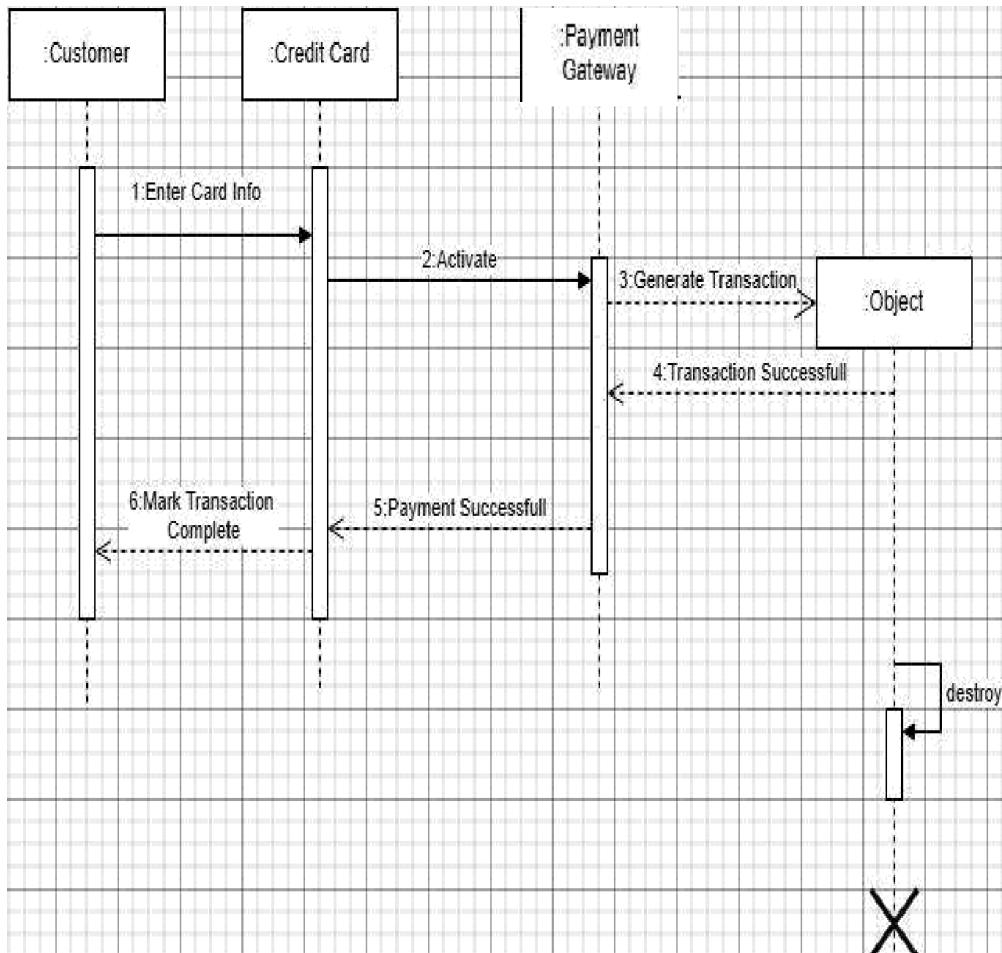
- The customer selects the “Make Payment” option. The system displays the total bill amount.
- The customer chooses a payment method (UPI, card, banking, etc.). The system sends the payment request to the payment gateway.
- The customer enters payment details.
- The payment gateway validates the details and processes the transaction. On success, the system updates the payment status.
- A receipt is generated and shown to the customer.

### **2.Create Customer Account**

- The admin selects the “Create Customer” option. The system displays a form to enter customer details.
- The admin enters name, ID proof, contact details, and address. The system validates the information.
- A unique customer ID is assigned automatically.

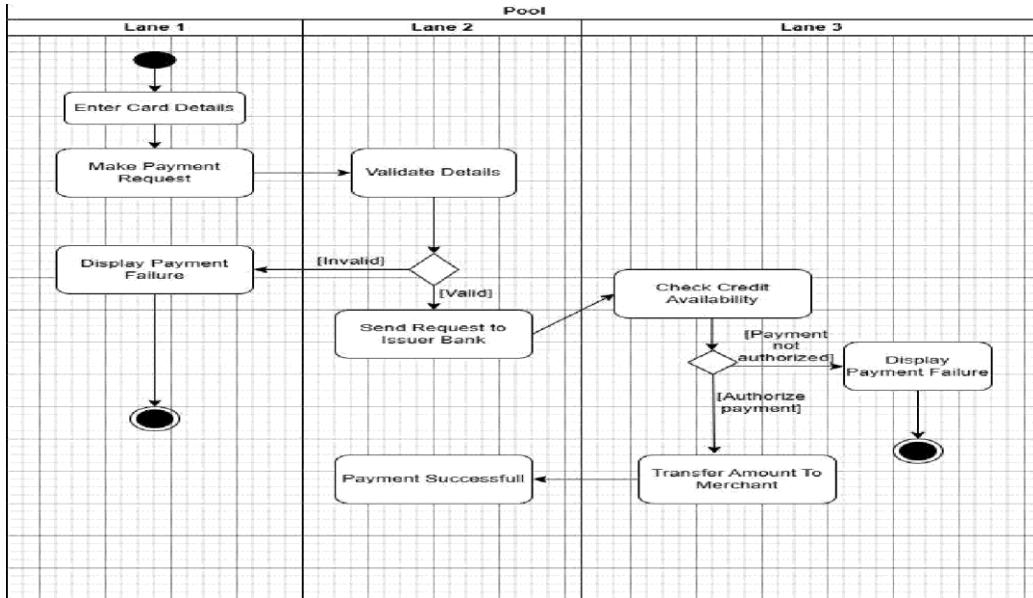
The customer account is created and stored in the database. The system confirms successful registration.

## Sequence Diagram(Advanced)



The advanced sequence diagram provides a more detailed view of how a credit card payment is handled behind the scenes. It begins when the Customer provides card details to the Merchant for payment. The merchant then sends an authorization request to the Bank Server. The bank server validates the card information, checks the account status, and approves or denies the transaction. If approved, the bank sends back a confirmation to the merchant, who then informs the customer that the payment was successful. Finally, the merchant updates the customer's statement or transaction history. This advanced diagram shows a more complete workflow, including authorization, validation, approval, and transaction recording.

## Activity Diagram(Advanced)



The advanced activity diagram presents a more detailed workflow of how a credit card application is handled. It begins when an applicant makes an inquiry about credit cards. The credit card processing system responds by displaying available card options. The applicant then selects a card and decides whether to proceed. If they continue, the system shows the required documents and information needed for application. The applicant gathers and submits these requirements. The system then validates the submitted information, checking eligibility criteria. Once everything is confirmed, the system displays the approved or completed application. This diagram covers more steps and interactions compared to the simple one and offers a clearer view of how credit card applications are processed from inquiry to validation.

### 3. Library Management System

#### SRS Document

Library Management System

#### 1. Introduction

##### 1.1 Purpose

→ To define the requirements for a library management system that automates book tracking, user management, and lending operations.

##### 1.2 Document Conventions

→ All dates follow the DD-MM-YYYY format.

##### 1.3 Intended Audience and Reading Suggestions

→ This document is intended for developers, testers, librarians and stakeholders.

##### 1.4 Product Scope

→ The system will manage book inventory, member records, issue or return processes, and overdue tracking. It will support both staff and member interactions.

##### 1.5 References

- IEEE SRS Standard 830 - 1998
- Library Science Best Practices
- PostgreSQL Documentation

#### 2. Overall Description

##### 2.1 Product Perspective

• This is a standalone web-based application replacing manual library operations.

##### 2.2 Product Functions

- Add/update/delete books and members
- Issue and return books
- Search catalog
- Generate reports.

##### 2.3 User Classes and Characteristics

- Librarians: Admin privileges

#### 2.4 Operating Environment

- Members: Limited access to search and request books
- Admins: Full system control.

#### 2.5 Design and Implementation Constraints

- Must use open-source technologies.
- Responsive UI for mobile and desktop.
- Data encryption for user credentials.

#### 2.6 User Documentation

- User manual (PDF)
- Online help section
- Admin training guide

#### 2.7 Assumptions and Dependencies

- Users have basic digital literacy
- Internet connectivity is available

#### 3. External Interface Requirements

##### 3.1 User Interfaces

- Login or registration page
- Dashboard for librarians and members
- Search and book request interface

##### 3.2 Hardware Interfaces

- A good server.

##### 3.3 Software Interfaces

- PostgreSQL database
- RESTful APIs for integration
- Authentication via OAuth 2.0

#### A. System features

- Book Management - categorise books by genre, author.
  - add or delete book records (edit too)
- Member management - Register and manage member profiles
- Lending Operations - issue or return books
  - Borrowing history
- Search and Reports
  - Calculate fines for overdue items
  - Search books by title, author  
or ISBN
  - Generate monthly usage reports.

#### 5 Other Nonfunctional Requirements

##### 5.1 Performance

- System should support 100 concurrent users
- Search results must load within 2 seconds

##### 5.2 Safety Requirements

- Regular data Backups
- Graceful error handling

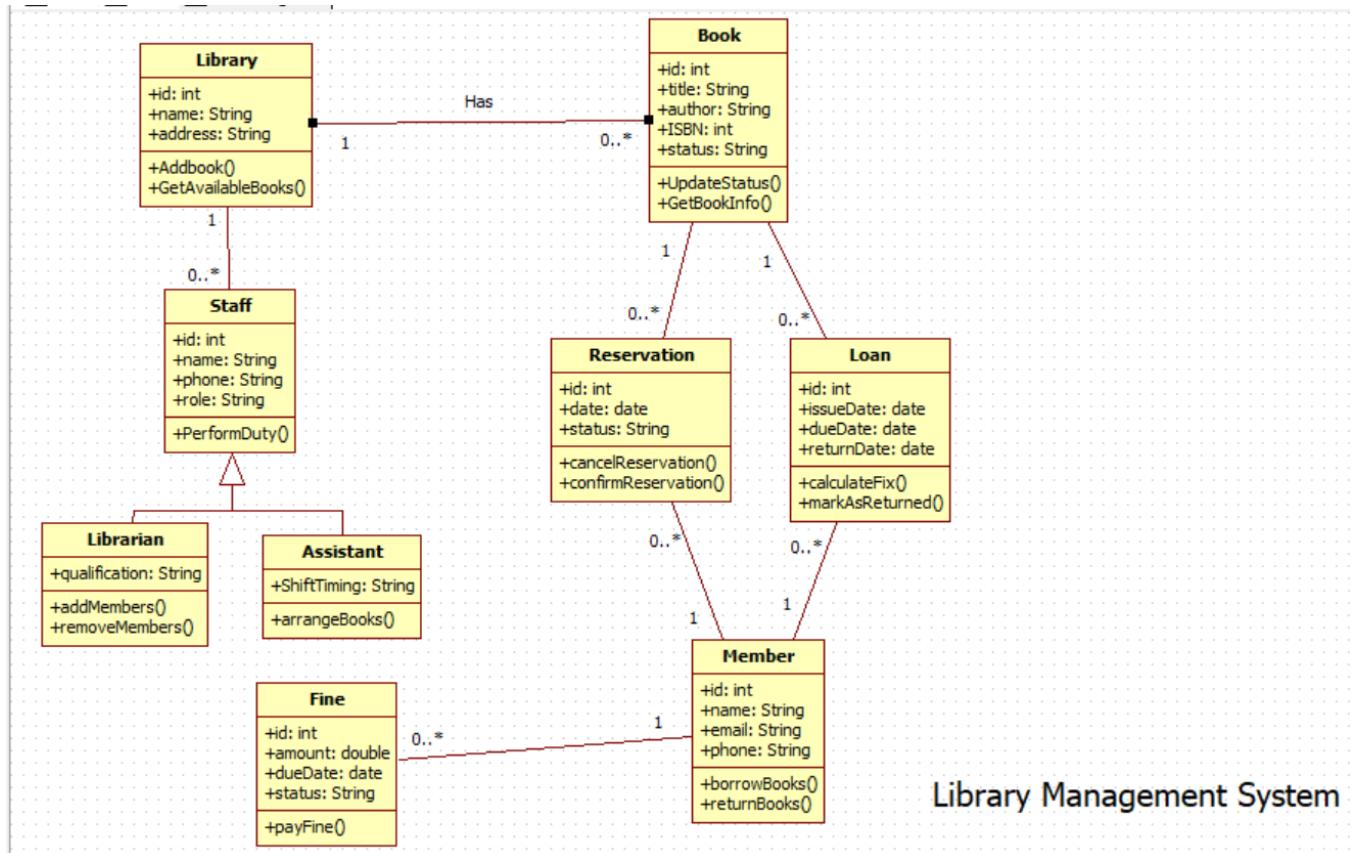
##### 5.3 Security requirements

- Role-based access control
- Passwords stored using hashing

#### 6 Other Requirements

- Integration with SMS gateway
- Support for multilingual UI.
- Audit logs for admin actions

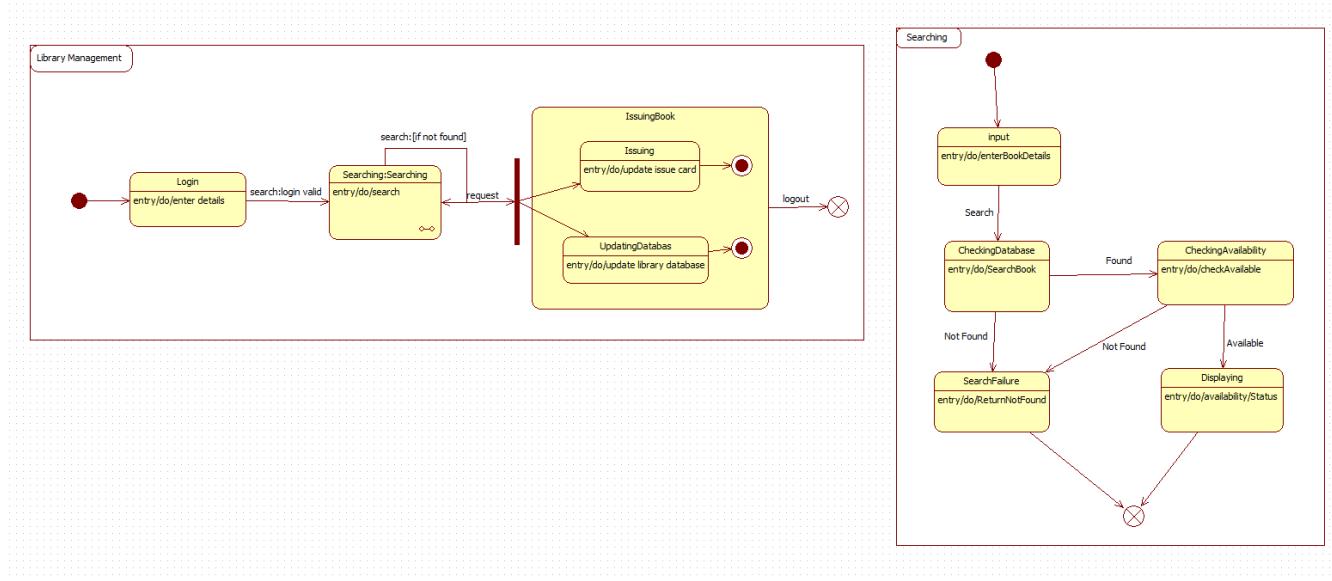
## Class Diagram



The class diagram represents a Library Management System where books, members, librarians, and transactions interact to manage library operations. The Book class stores details such as author, edition, price, and purchase date, and it is specialized into different types like Journals, Magazines, and Subject Text through inheritance. Members of the library—categorized as Students and Faculty—send book requests and borrow books. The Librarian plays a central role by searching books, verifying members, issuing books, calculating fines, and handling book returns. The librarian also updates the status of books and ensures that library policies like issuing limits are followed.

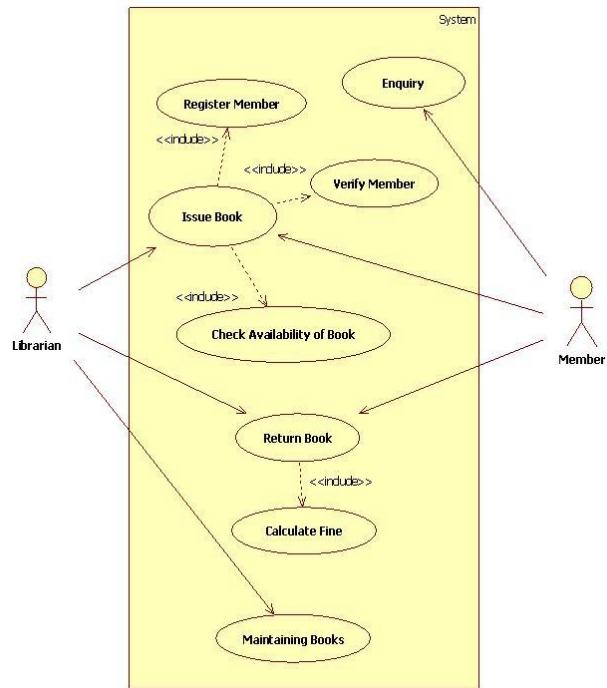
The system also keeps track of borrowing activities using the Transaction class, which stores the transaction ID, book ID, issue date, and due date. When members return books late, the Fine class is used to generate bills, update fines, and store payment information. Members pay these fines, and the librarian or system records them for future reference. Overall, the class diagram shows how different components—books, members, librarians, transactions, and fines—work together to ensure smooth operation of the library.

## State Diagram(Advanced)



The advanced state diagram provides a more complete picture of the library's book issuing and returning process. It starts with the user searching for a book. If the book is available, the system verifies stock and moves to the issue state, where member and library records are updated. If the book is unavailable, the system places the user in a waitlist and notifies them when the book becomes available. Once issued, the system moves into the Borrowed state, where the due date is monitored. From here, the user may renew the book or return it. Upon returning, the system checks for damage or late return and computes fines if applicable. If a fine exists, the process proceeds to Fine Payment; otherwise, the process ends. This diagram captures the full lifecycle of a borrowed book—from search and waitlist to issuing, borrowing, returning, renewal, and fine handling.

## Use Case(Advanced)



The advanced use case diagram presents a more detailed and comprehensive view of the library system by involving multiple actors—User (Student/Staff), Libraries, and the Library Database. It includes a wider range of use cases such as Authentication, Requesting New Books, Reserving Books, Renewing Books, Paying Fines, Providing Feedback, Registering New Users, and Filling Forms. The system also manages book records using operations like Adding, Updating, and Deleting Records, as well as preparing the library database. Several use cases use *include* and *extend* relationships to show dependencies and optional behaviors such as invalid login, invalid ID, or invalid renewal. This advanced diagram gives a full picture of how users interact with the system, how library data is maintained, and how operations extend beyond simple issuing and returning of books.

## **Scenarios**

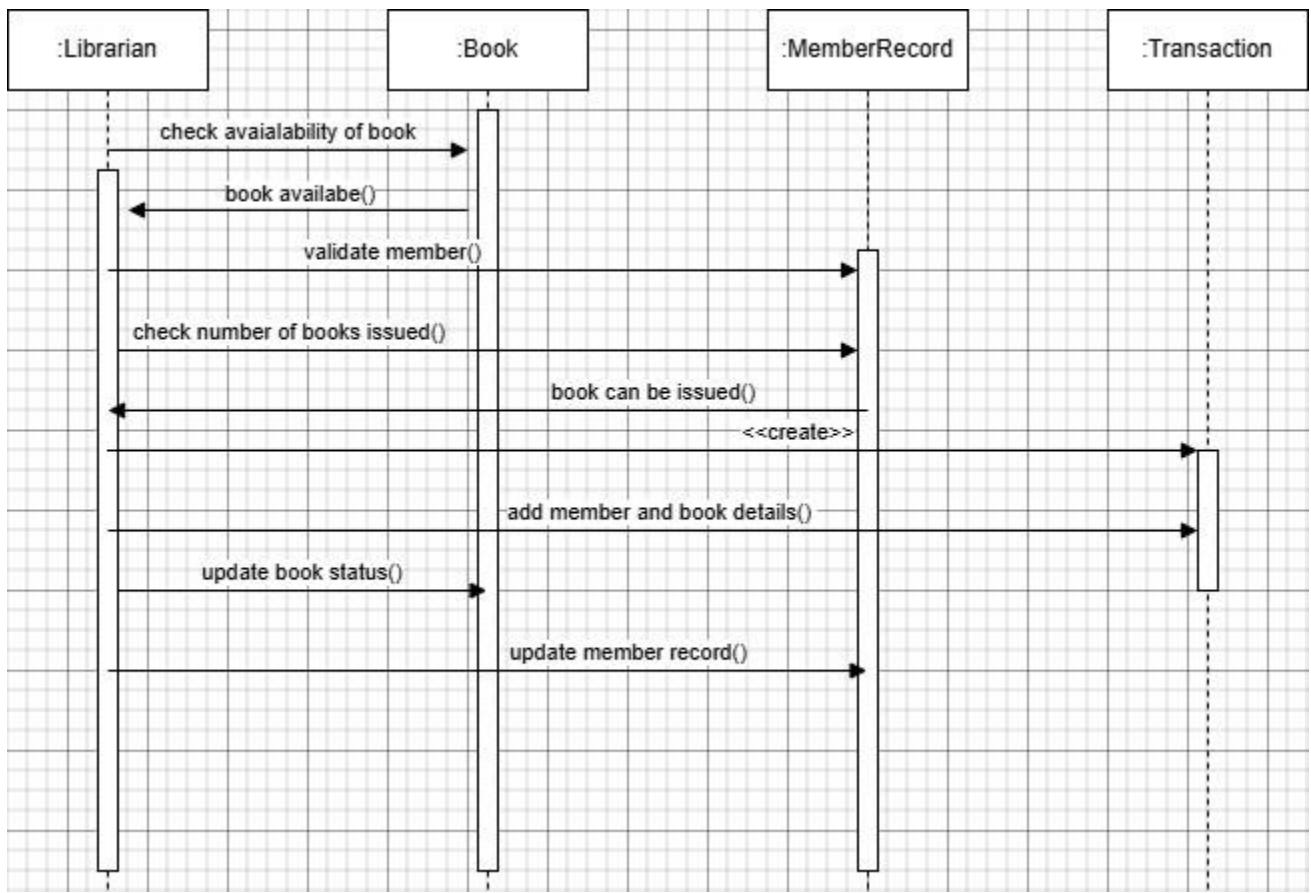
### **1: Issue Book**

- The librarian selects the “Issue Book” option.
- The librarian enters the member ID and book ID.
- The system checks the member's status and borrowing limit.
- The system verifies the availability of the book.
- If valid, the system updates the issue record and marks the book as borrowed.
- The librarian hands the book to the member.

### **2: Search Book**

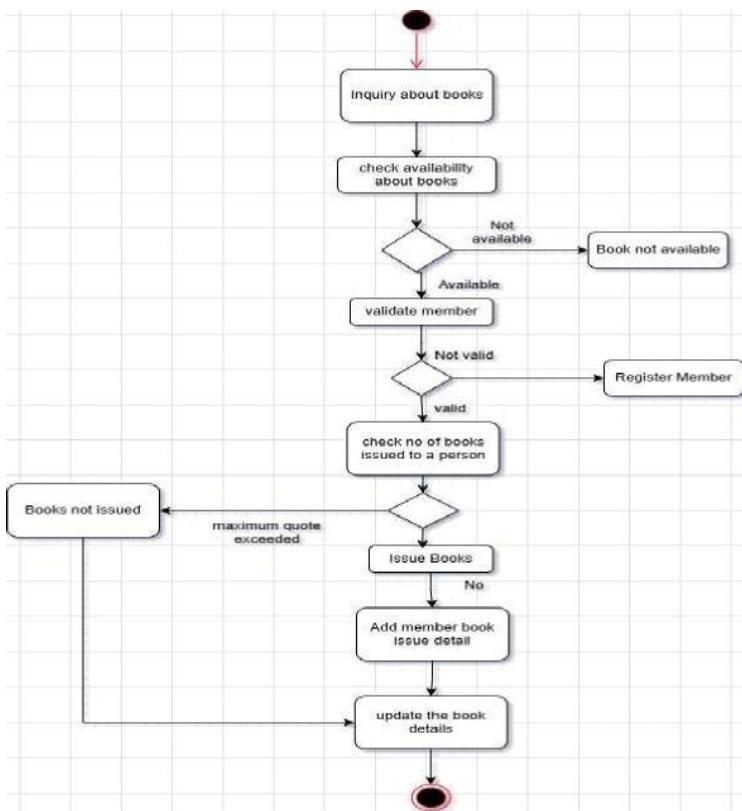
- The student selects the “Search Book” option. The system prompts for the book title/author/ID.  
The student enters the details.
- The system scans the database for matching books.
- The system displays the availability status (Available / Issued / Not Found).

## Sequence Diagram Advanced



The advanced sequence diagram provides a more detailed view of the book-issuing process. It starts when the Librarian checks the availability of a selected book. The Book object responds with availability status. The librarian then validates the member and checks how many books the member has already borrowed. If issuing is allowed, a Transaction record is created, containing details of the member and the book. The librarian then updates the book status, marking it as issued, and updates the member's record to reflect the new loan. This detailed diagram shows all internal checks—availability, member validation, issuing limits—and the creation of proper records, giving a complete view of how the system handles book issuing behind the scenes.

## Activity Diagram(Advanced)



The advanced activity diagram provides a more detailed and complete view of the book-issuing workflow. In addition to checking availability and validating the member, it clearly shows decision points such as book unavailability, invalid member status, and quota limits. It also includes additional actions like adding member book issue details, updating the complete book record, and handling both “Book not issued” and “Issue Books” outcomes. By showing all alternative paths and internal updates, the advanced diagram presents the full operational logic of how the library processes book inquiries, member eligibility, quota checks, issuing actions, and record maintenance. It reflects the complete backend process of issuing a book in a real library system.

## 4. Stock Maintenance System

### SRS Document:

IV Stock Maintenance System

1. Introduction

1.1 Purpose  
To define the requirements for a Stock Maintenance system that enables efficient tracking, updating and efficient tracking, updating, and reporting of inventory across multiple categories and locations.

1.2 Document Conventions  
• Currency: INR  
• Date: DD-MM-YYYY format.

1.3 Audience  
• This document is intended for developers, inventory managers, business stakeholders, and QA teams.

1.4 Scope  
• The system will manage stock levels, record transactions, generate alerts for low inventory and support reporting for decision-making. It will streamline operations for warehouse, retail outlets and procurement teams.

1.5 References  
• Inventory management  
• ISO/IEC 25010 Software Quality Model

2. Overall Description

2.1 Product Perspective  
This is a centralized web-based application replacing manual or spreadsheet-based stock tracking and trading.

2.2 Product Functions  
• Add / update / delete stock items  
• Monitor stock levels and movement  
• Generate alerts and reports  
• Support multi-location inventory

→ Alerts and Notifications  
→ Reporting

3. External Interface Requirements

3.1 User Interface  
• Login and Dashboard  
• Stock entry and update forms  
• Alerts and reporting module

3.2 Hardware Interface  
• Printer for Stock Reports

3.3 Software interfaces  
• MySQL database  
• Authentication via JWT

4. System Features

→ Stock Management - add / edit / delete stock items  
→ Categorize items  
→ Inventory tracking

5. Non-functional Requirements

5.1 Performance Requirements  
• Support 200 concurrent users  
• Stock updates reflected within 5 seconds

5.2 Safety Requirements  
• Role-based access control  
• Encrypted credentials and secure session

5.3 Safety requirements  
• Daily backups  
• Rollback mechanisms for erroneous entries

6. Other Requirements

• Support multi-language UI  
• Audit logs for stock changes  
• Integration with ERP systems

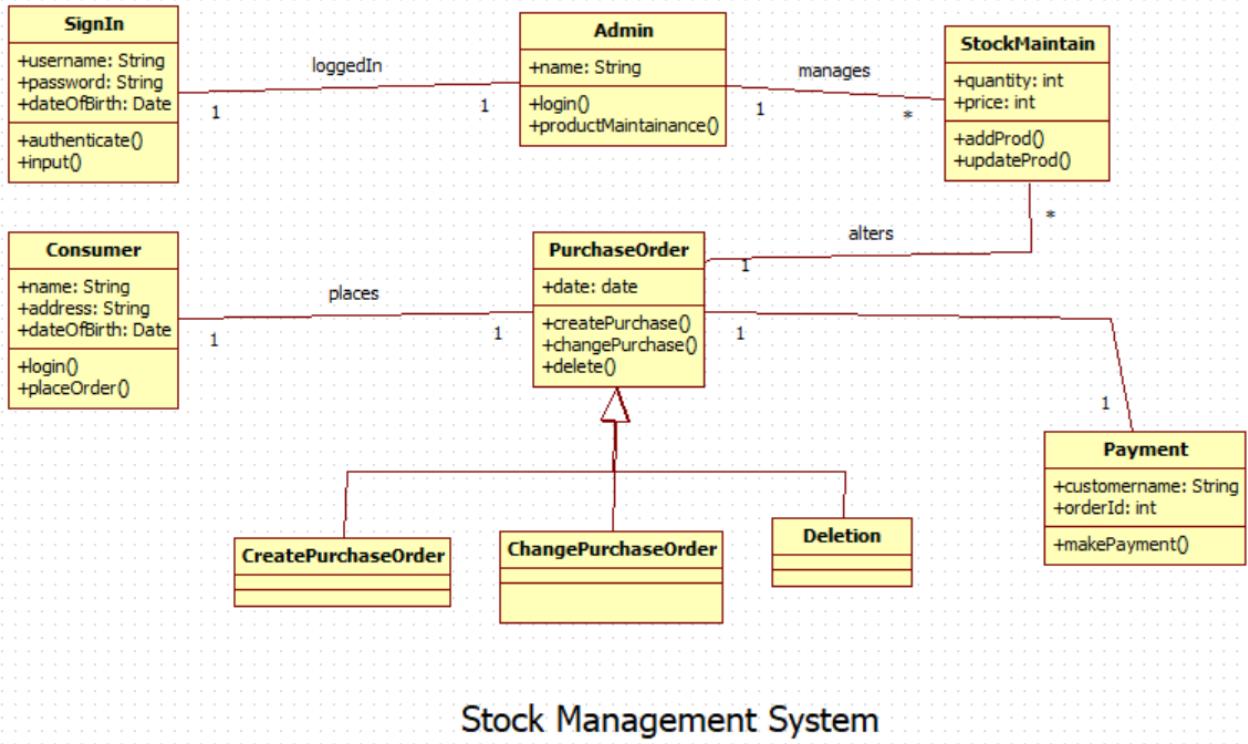
2.3 User classes and characteristics  
• Admin: full access to all modules  
• Inventory Manager: Manage stock and view reports  
• Staff: limited access to update stock entries

2.4 Operating environment  
• Web browsers (Chrome, Edge)  
• Server: Linux with MySQL  
• Client: Desktop or mobile devices

2.5 Design and implementation constraints  
• Must support role-based access  
• Use open-source stack (Node.js etc)  
• Responsive design for mobile compatibility.

2.6 Assumptions and Dependencies  
• Users have basic inventory knowledge  
• Stable internet connection  
• MySQL and Node.js installed on server

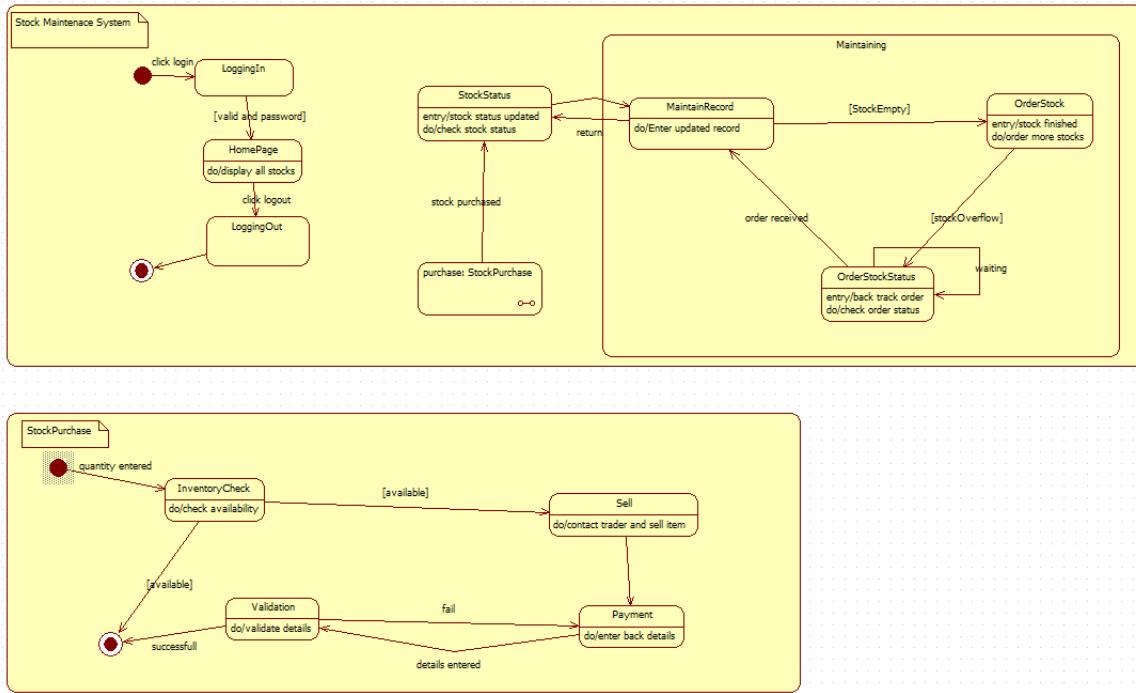
## Class Diagram



### Explanation:

This class diagram visually represents the key entities and their relationships in a Sales Management System, showing how different objects such as Users, Customers, Admins, Suppliers, Stocks, Products, Transactions, and Sales interact within the system. Each class is defined with specific attributes and methods, and the arrows indicate relationships—such as Customers making Transactions, Admins managing Stocks and contacting Suppliers, and Sales being associated with Customers—highlighting the flow of information and control throughout the platform. This structure helps ensure efficient handling of sales, inventory, user actions, and supplier coordination, forming a cohesive framework for managing business operations digitally.

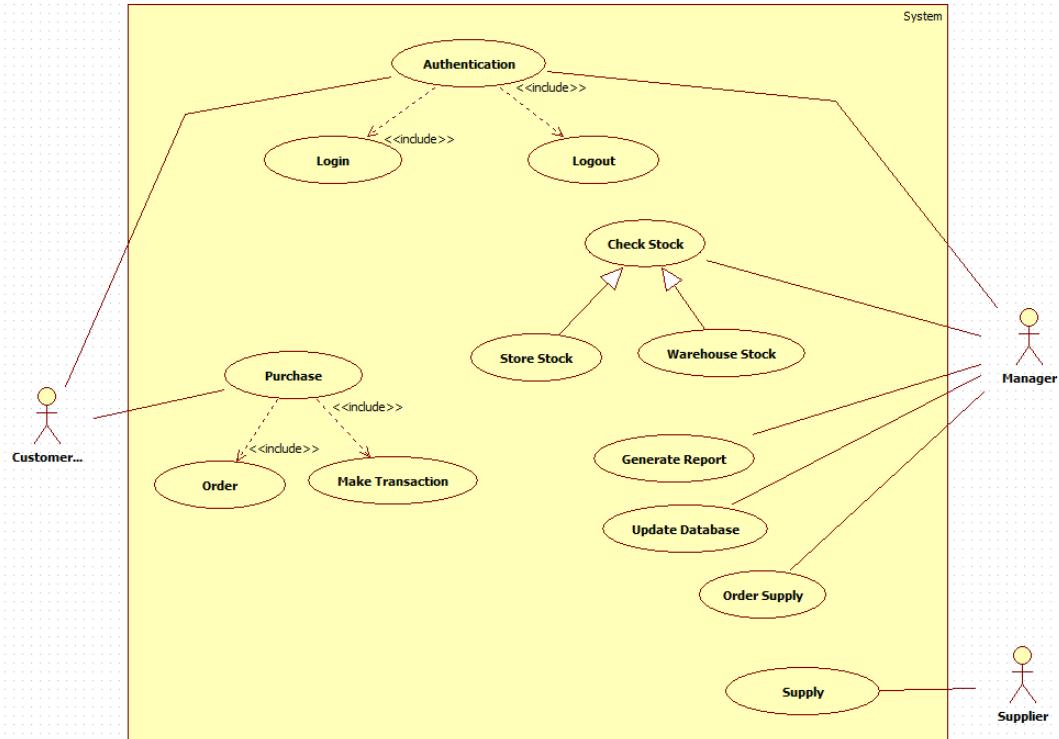
## Advanced State Diagram



This advanced state diagram models the Stock Maintenance System, focusing on two core processes: managing overall stock status and handling stock purchase transactions. The top section illustrates user actions like logging in, navigating the homepage to view stocks, initiating stock purchases, and logging out. Once inside the maintenance area, states transition between updating records, ordering new stock if inventory runs out, and tracking the status of those orders—ensuring stock levels remain sufficient through cyclic monitoring and updating.

The lower section details the Stock Purchase process, starting when a quantity is entered. The flow continues with an inventory check; if stock is available, the item is sold and the system validates and processes the payment. If any details are invalid, the process prompts re-entry of information. Successful transactions complete the state flow, while unavailability or validation failure leads to process termination. Overall, the diagram effectively captures the operational logic and interdependencies involved in maintaining stock and processing purchases within an inventory system.

## Advanced Use Case:



This use case diagram outlines the roles of Manager, Store Staff, and Supplier in a retail system. The Manager oversees buying stock, making payments, and supervising staff. Store Staff handle inventory reporting, product quality checks, and selling stock. The Supplier delivers orders, receives payments, and manages quality issue reports. Relationships like  $\leftrightarrow$  and  $\rightarrow$  show dependencies—for example, buying stock includes giving payment and may extend to reporting quality issues, which in turn includes returning damaged goods. This structured view clarifies responsibilities, enhances system design, and ensures smooth coordination among stakeholders for efficient inventory and supply chain management.

- The diagram maps three actors: Manager, Store Staff, and Supplier, each with distinct responsibilities. The Manager handles buying stock, making payments, and supervising staff.
- Store Staff are responsible for inventory reporting, product quality checks, and selling stock.
- The Supplier delivers orders, receives payments, and manages quality issue reports.

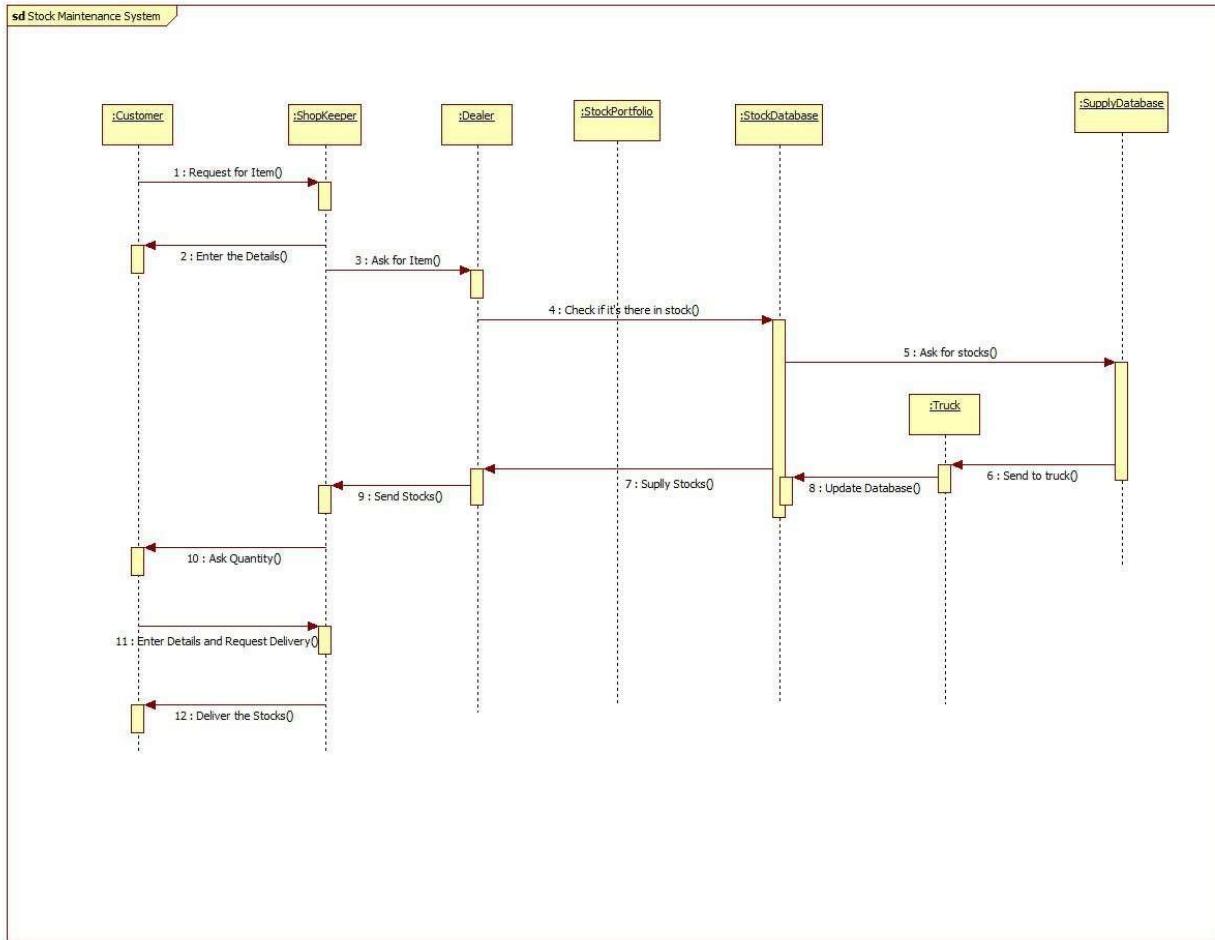
## Scenario 1: Purchase and Stock Update (Succeeded)

- Purchase department raises a purchase requisition for 500 units of Item-X
- Supplier accepts PO and delivers 500 units with proper invoice
- Warehouse staff receives goods, performs quality check — Passed
- Staff scans/enters GRN (Goods Receipt Note) into the system
- System automatically increases stock level of Item-X by 500
- Stock value updated in inventory ledger
- Finance receives GRN copy and processes supplier payment
- Re-order level alert for Item-X automatically turns off
- Dashboard shows updated stock: Item-X = 1,200 units

## Scenario 2: Stock Rejection and Return (Exception Scenario)

- Supplier delivers 300 units of Item-Y, but 80 units found damaged
- Warehouse staff performs quality check — 80m units
- Staff creates Partial GRN for 220 units only
- System adds only 220 units to stock, blocks 80 units
- Return Material Authorization (RMA) generated automatically
- Rejected 80 units returned to supplier with Debit Note
- Stock level of Item-Y updated correctly (only +220)
- Supplier acknowledges return, issues Credit Note
- Inventory report reflects accurate stock and pending returns
- Low-stock alert remains active since actual receipt is below requirement

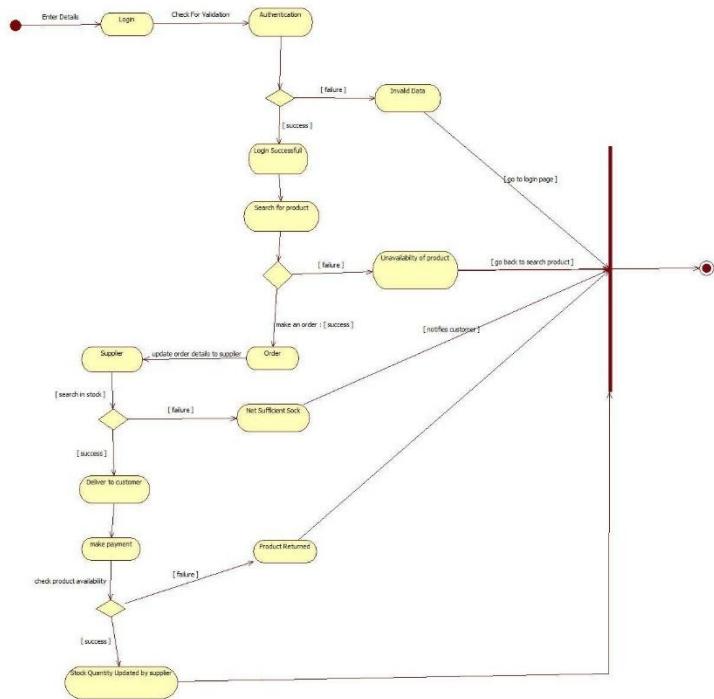
## Advanced Sequence Diagram:



This UML sequence diagram for the Old Stock Management System outlines the step-by-step interaction between entities like Customer, Login, Authentication, Search Product, Order, Supplier, and Stock. The process begins with customer detail entry and validation. Upon successful login, the customer searches for a product and checks its quantity. If unavailable, the system prompts them to search for alternatives. Once an order is placed, the supplier is notified, and delivery is arranged. The customer pays, stock quantities are updated, and any quality issues can be reported, triggering a refund process. This flow ensures efficient order handling, inventory tracking, and customer satisfaction.

- Customer details are validated before login is granted.
- Product search triggers quantity check in stock.
- If stock is unavailable, customer is prompted to choose a new product.
- Supplier receives order details and delivers to the customer.
- Payment, quantity update, and refund (if needed) complete the transaction flow.

## Advanced Activity Diagram:



After the order is successfully placed, the system interacts with the supplier to fetch stock and order details. The supplier searches inventory and updates the system accordingly. If delivery fails, the customer is notified immediately. Successful deliveries lead to the payment phase, where transaction failures are also communicated. This decision-based flow ensures transparency and error handling at each step. The supplier plays a critical role in maintaining stock accuracy and fulfilling orders. The system's ability to notify users at every failure point enhances customer experience and operational reliability, making the process robust and user-friendly.

The final steps involve quality checks and stock updates. Once the product is delivered, the customer can report any quality issues. If a defect is found, the system initiates a refund process. Meanwhile, the supplier updates stock quantities to reflect the latest inventory status. This ensures real-time tracking and accountability. The flowchart's design emphasizes decision points and feedback loops, allowing for efficient handling of exceptions. By integrating supplier coordination, customer notifications, and inventory updates, the system maintains a seamless shopping experience while ensuring operational integrity and customer satisfaction.

## 5. Passport Automation System

### SRS Document:

Passport Automation System

#### 1. Introduction

##### 1.1 Purpose

To define the requirements for a Passport Automation system that streamlines the application, verification and issuance of passports through a secure and user-friendly digital platform.

##### 1.2 Document Conventions

- Currency INR.

##### 1.3 Audience

- This document is intended for government officials, developers, testers, and system integrators.

##### 1.4 Scope

- The system will automate passport-related services including application submission, document verification, appointment scheduling and status tracking. It will reduce manual processing and improve transparency.

##### 1.5 References

- Government of India Passport Guidelines
- Aadhar API documentation

#### 2. Overall Description

##### 2.1 Product Perspective

- A web-based application with national identity databases and police verification systems.

##### 2.2 Product Functions

- Online passport application
- Document upload and verification
- Appointment booking
- Status tracking and notifications

#### 3. System Features

- Stock Management - add/edit/delete stock items
- Inventory Tracking - record of incoming and outgoing stock.
- Alerts and Notifications
- Reporting

#### 4. Other Non-functional

##### Performance Requirements

- High availability with minimal downtime.
- Fast processing

##### Usability

#### 2.3 User Classes and Characteristics

- Applicants: Citizens applying for passport
- Officials: Verify documents and approve applications
- Admins: Manage system settings and user roles

#### 2.4 Operating Environment

- Web browsers
- Desktop or mobile devices

#### 2.5 Design and Implementation Constraints

- Must comply with government security standards
- Aadhar and PAN integration required
- Multilingual support

#### 2.6 User Documentation

- Online help center

#### 2.7 Assumptions and Dependencies

- Applicants have valid identity documents
- Available internet access

#### 3. External Interface Requirements

##### 3.1 User interfaces

- Application form interface
- Document upload portal
- Appointment calendar
- Status dashboard

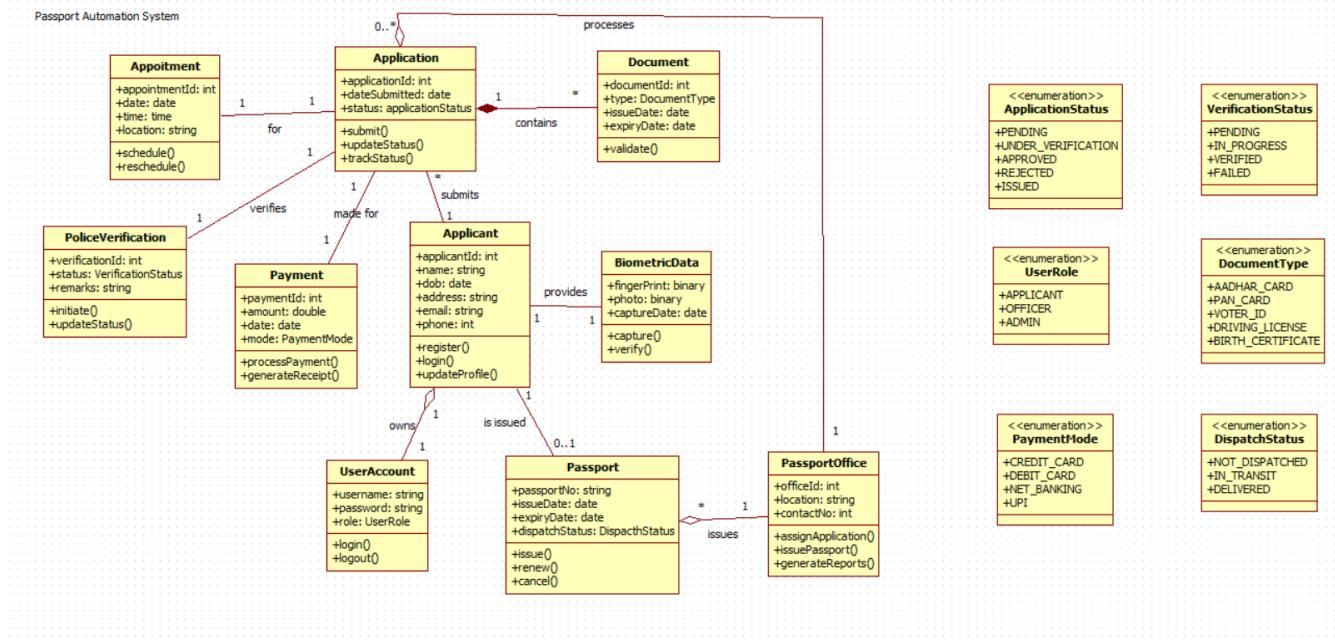
##### 3.2 Hardware interfaces

- Printers for application receipts

##### 3.3 Software interfaces

- Aadhar and PAN APIs
- Police verification system
- Payment gateway

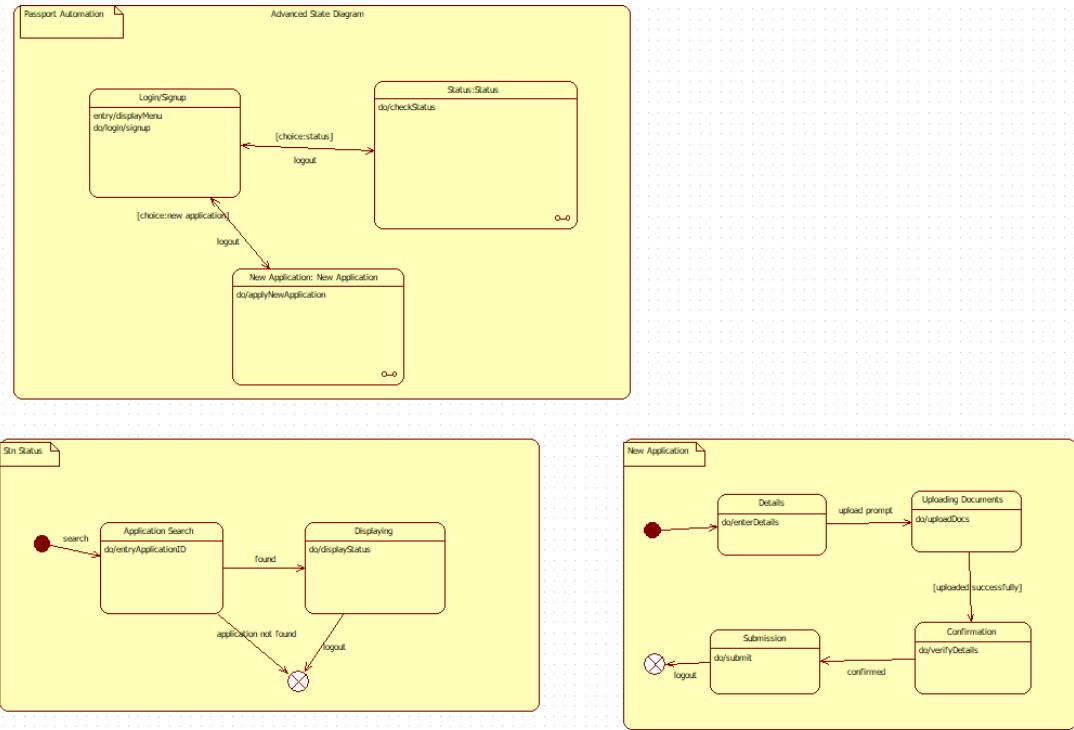
# Class Diagram



## Explanation:

This class diagram represents an online Passport Automation System. An Applicant (inheriting basic details from Person and Registration) fills and submits an Application, which an Administrator validates, approves, or declines. Once approved, the applicant books an Appointment for document verification. At the center, Document Verification staff verify the applicant's original documents (docType, place, etc.) during the scheduled slot. The system maintains applicant info, tracks application status, manages appointments, and logs all actions (register, update, cancel, reschedule) while ensuring only verified and approved applicants proceed to the final appointment stage.

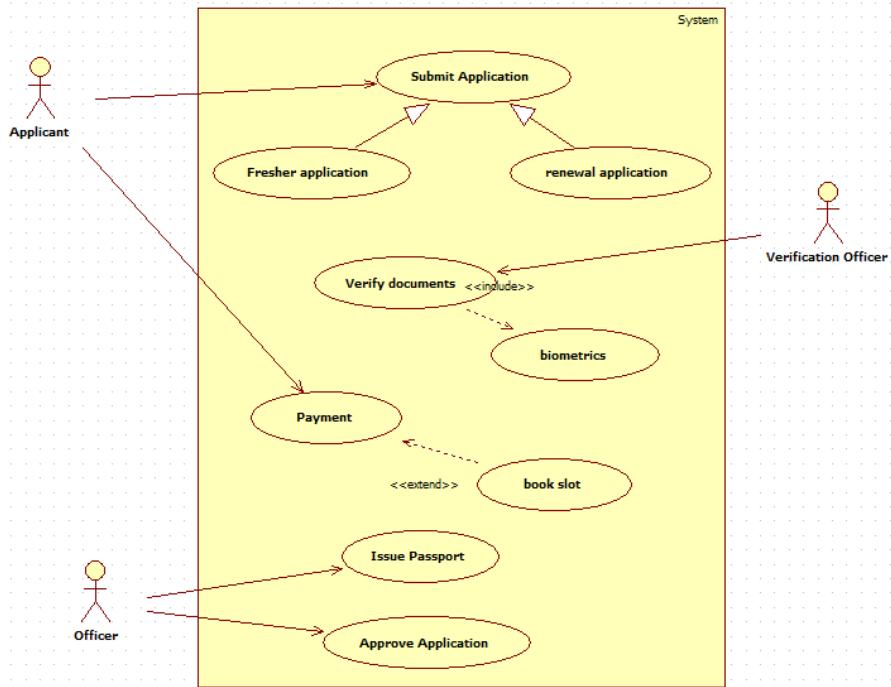
## State Diagram:



The activity diagram presents the main menu of a Passport Automation System, where users begin by logging in or signing up. After successful authentication, they are prompted to choose between two primary options: checking the status of an existing application or starting a new passport application. The “Status” path allows users to enter their Application ID; if the record is found, the current status is displayed instantly. If not found, the process terminates gracefully with a logout. This streamlined design enables applicants to quickly track progress without re-submitting documents, enhancing user convenience and reducing support queries.

Selecting “New Application” initiates a dedicated subprocess where users first input personal details, then proceed to upload required documents. Successful upload triggers the “Submission” activity, followed by system-side confirmation and verification of the entered data. Only after verification is complete does the applicant receive final confirmation. Logout options are strategically placed throughout to ensure secure sessions. The clear sequential flow — details —+ documents —→ submission —+ confirmation — minimizes errors, ensures all mandatory information is captured, and provides immediate feedback, making the online passport application process efficient, user-friendly, and compliant with official requirements.

## Advanced Use Case:



The use case diagram outlines a comprehensive Passport Automation System with four actors. The Applicant registers on the portal, logs in, fills the application form, pays fees via an external Payment Gateway, tracks status, and schedules an appointment. The "Schedule Appointment" use case includes receiving notifications and mandatory document verification. The Passport Officer verifies documents, conducts interviews, and approves or rejects the application. The Police actor performs background verification and submits the police report. This citizen-centric flow ensures online submission, transparent tracking, and seamless integration of payment and notification services.

The Admin manages the entire backend by handling user accounts, officer profiles, and generating reports. Key «include» relationships highlight that scheduling an appointment always triggers notification delivery and requires document verification, preventing incomplete applications from proceeding. The diagram clearly separates responsibilities: applicants handle submission and payments, officers manage verification and decision-making, police provide clearance, and admin oversees system governance. This structured, role-based design with included use cases ensures security, accountability, and efficient processing from registration to final decision, making the passport issuance process faster, paperless, and user-friendly.

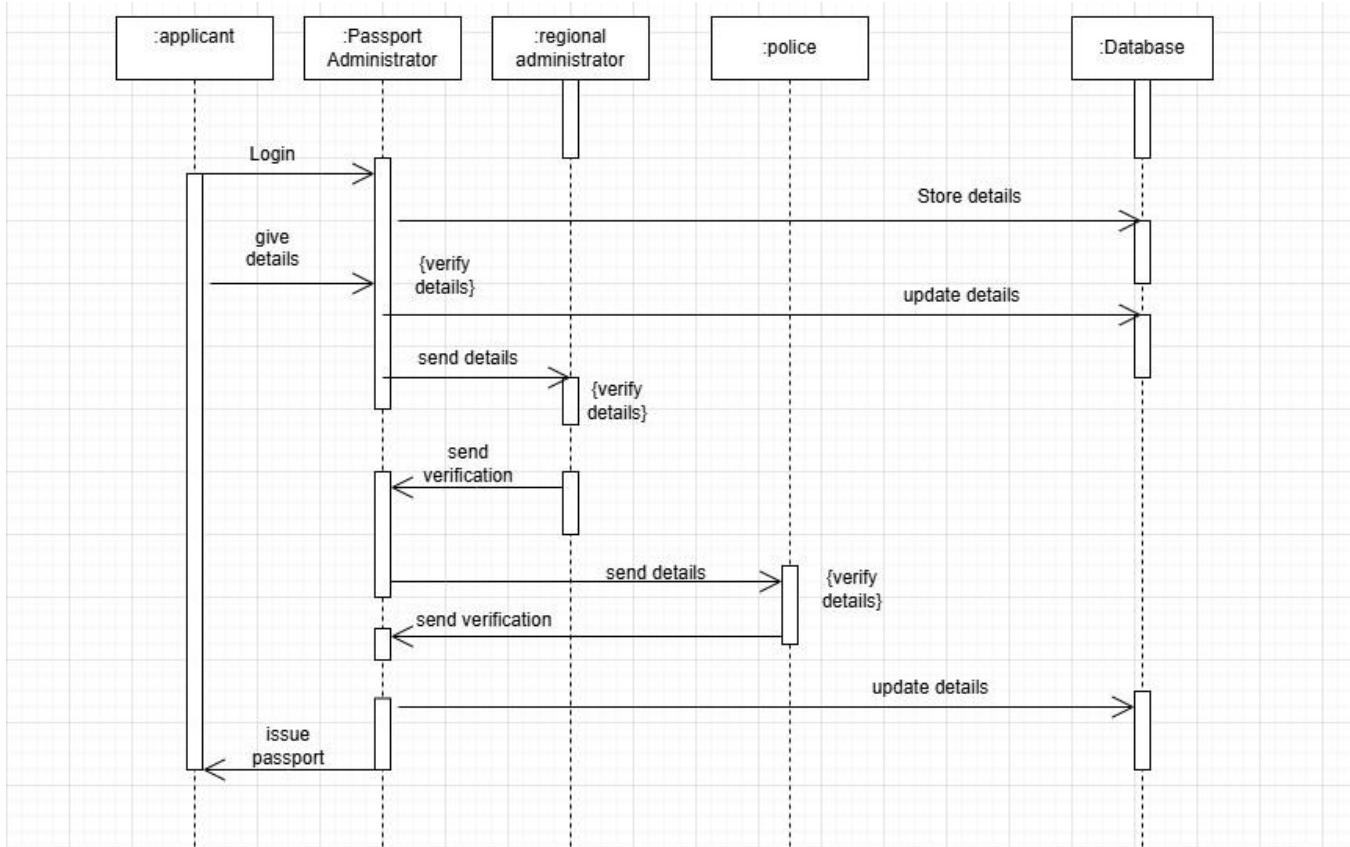
## **Scenario 1: Main Scenario (succeeded)**

- Applicant registers and logs into the passport portal
- Fills application form correctly and uploads clear documents
- Pays fees online successfully
- Books and attends appointment at PSK on scheduled date
- Officer verifies original documents and captures biometrics smoothly
- Police verification completed with clear/no-adverse report
- Regional authority approves the application
- Passport printed and dispatched via Speed Post
- Applicant tracks status online and receives passport within 25-30 days

## **Scenario 2: Alternative Scenario (Rejection)**

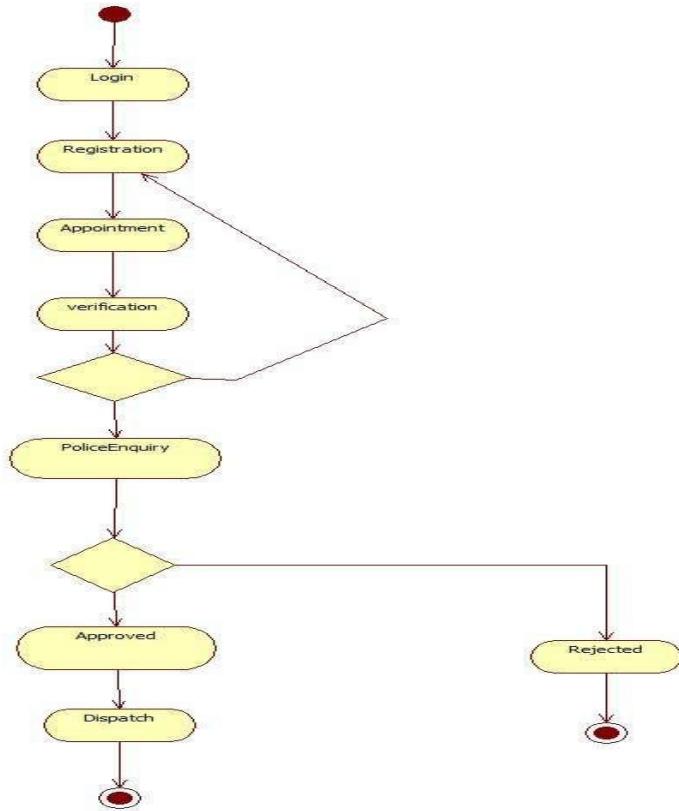
- Applicant completes registration, form filling and fee payment
- Attends appointment; document & biometric verification successful
- Police station visits applicant's address for enquiry
- System detects pending criminal case / FIR
- Police submit adverse verification report
- Regional authority reviews and rejects the application
- Applicant receives SMS + email notification sent with rejection reason
- Status updated as “Rejected” on portal

## Advanced Sequence Diagram:



This sequence diagram illustrates the interaction flow in a Passport Automation System. The Applicant (Object1) initiates by submitting the passport application to the System (Object3). The System immediately notifies the Passport Officer (Object2) for verification. The Officer verifies the documents and sends the approval/rejection decision back to the System. The System then forwards this decision to the Admin (Object4) to update the official application status. Finally, the System notifies the Applicant about the result (approved or rejected). This clear, step-by-step message exchange ensures transparent communication, proper verification, and timely status updates among applicant, officer, system, and admin.

## Advanced Activity Diagram:



The activity diagram outlines the complete passport application lifecycle. It begins with user Login followed by Registration of personal details. The applicant then books an Appointment for document verification, after which physical Verification takes place at the passport office. A crucial decision point follows verification: if documents and identity are valid, the process continues; otherwise, it may loop back or terminate. The flow then proceeds to Police Enquiry for background and criminal record checks, ensuring eligibility. This structured sequence guarantees that only verified applicants advance to the final approval stage.

After police enquiry, another decision node determines the outcome: if clearance is received, the application is Approved and proceeds to Dispatch of the passport; if any issue is found, the application is Rejected and the process ends. The diagram clearly shows two termination points — one for successful dispatch and one for rejection. By incorporating appointment scheduling, multi-level verification (document + police), and explicit decision points, the flow ensures security, transparency, and compliance with legal procedures, preventing issuance to ineligible persons while providing a smooth path for genuine applicants from registration to final delivery.













