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:

Manith G Raju

1BM23CS182

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 Manith G Raju (1BM23CS182) during the 5th Semester August 2025-December 2025

Signature of the Faculty Incharge:

NAME OF THE Your Batch Incharge and designation:

Department of Computer Science and Engineering

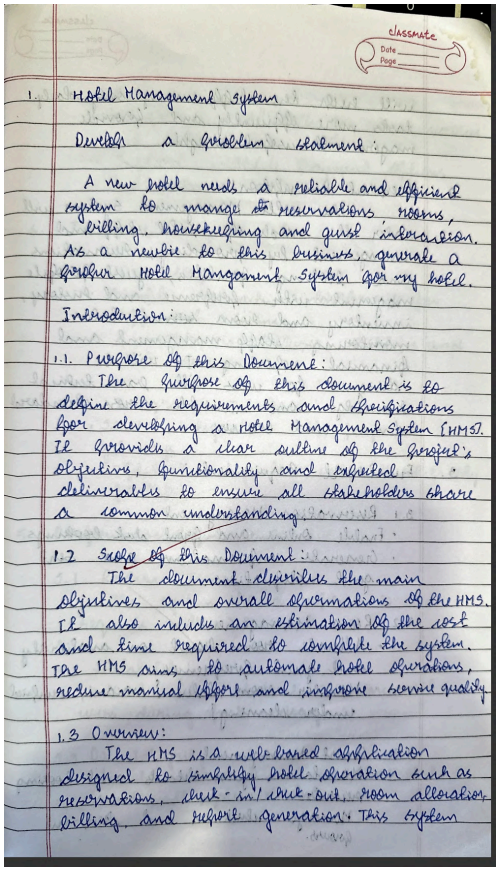
B.M.S. College of Engineering, Bangalore

Table of Contents

1. Hotel Management System

Problem Statement
SRS-Software Requirements Specification
Class Diagram Requirements: Minimum 7 classes, 4 attributes and 4 operations in each class, associations, association name, association end names, multiplicity, association class, enumeration, qualified association, aggregation, composition, generalization, ordered, sequencing, multiplicity of attribute, brief description of the Class diagram
State Diagram: one simple state diagram, one advance state diagram (either Sub Machine or Nested State, include any one of the concurrency in your design) Requirements: minimum of 6 states for both the state diagram with state name, do activities, events, guard condition, brief description of State diagram
Use-Case Diagram: Minimum 5 use cases to be identified, and design one simple use case diagram and one advanced use case diagram(ie using include,extend and generalization relationship) and explanation as there in the solution manual.
Sequence Diagram: Write the scenario for any two use case transaction completely. and design the simple sequence diagram with minimum of 5 objects and communication between them. Design advanced sequence diagram using passive and transient objects. Brief explanation for each
Activity Diagram: Design the simple activity diagram showing the algorithm or workflow. Design the advanced activity diagram with swimlanes showing the responsible person for swimlane and also write the brief explanation for both.

SRS Document



1. Hotel Management System

Develop a problem statement:

A new hotel needs a reliable and efficient system to manage its reservations, rooms, billing, housekeeping and guest interaction. As a newbie to this business, generate a further Hotel Mangement System for my hotel.

Introduction:

1.1. Purpose of this Document:

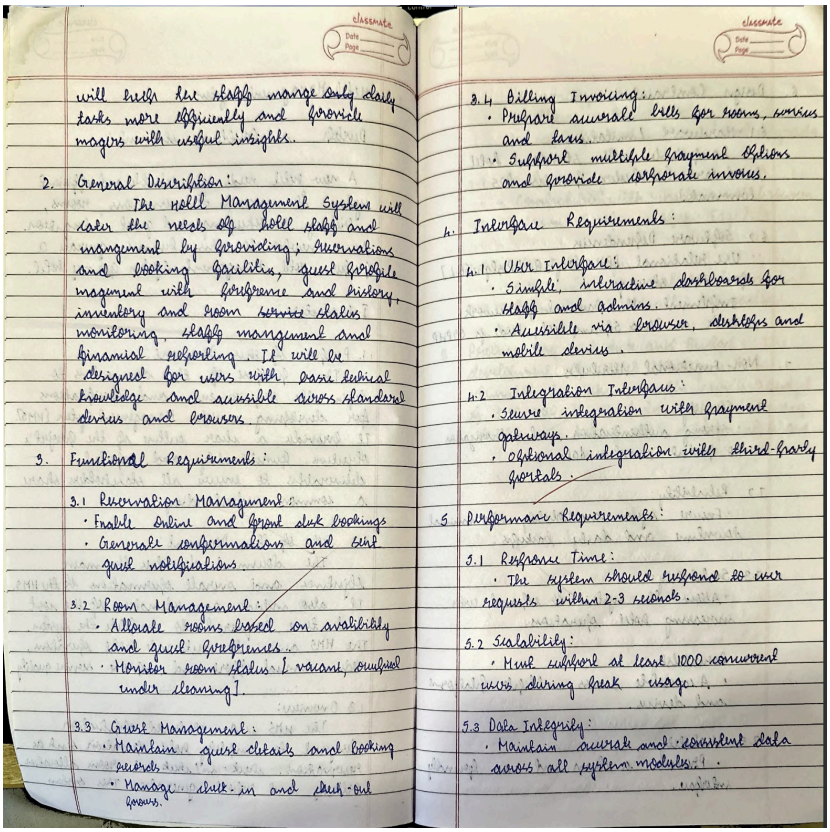
The purpose of this document is to define the requirements and specifications for developing a Hotel Management System (HMS). It provides a clear outline of the project's objectives, functionality and technical deliverables to ensure all stakeholders share a common understanding.

1.2 Scope of this Document:

The document describes the main objectives and overall operations of the HMS. It also includes an estimation of the cost and time required to complete the system. The HMS aims to automate hotel operations, reduce manual effort and improve service quality.

1.3 Overview:

The HMS is a web-based application designed to simplify hotel operation such as reservations, check-in/check-out, room allocation, billing and report generation. This system



will help the staffs manage daily tasks more efficiently and provide mgrs with useful insights.

2. General Description:

The Hotel Management System will cater the needs of hotel staffs and mangement by providing; reservations and booking facilities, guest profile mgmt with preference and history, inventory and room service status monitoring, staffs mangement and financial reporting. It will be designed for users with basic technical knowledge and accessible across standard devices and browsers.

3. Functional Requirements:

3.1 Reservation Management:
- Enable online and front desk bookings
- Generate confirmations and send guest notifications

3.2 Room Management:
- Allocate rooms based on availability and guest preferences.
- Monitor room status [vacant, occupied under cleaning].

3.3 Guest Management:
- Maintain guest details and booking records.
- Manage check-in and check-out process.

3.4 Billing Invoicing:
- Prepare accurate bills for rooms, services and taxes.
- Support multiple payment options and provide corporate invoices.

4. Interface Requirements:

4.1 User Interface:
- Simple, interactive dashboards for staffs and admins.
- Accessible via browser, desktops and mobile devices.

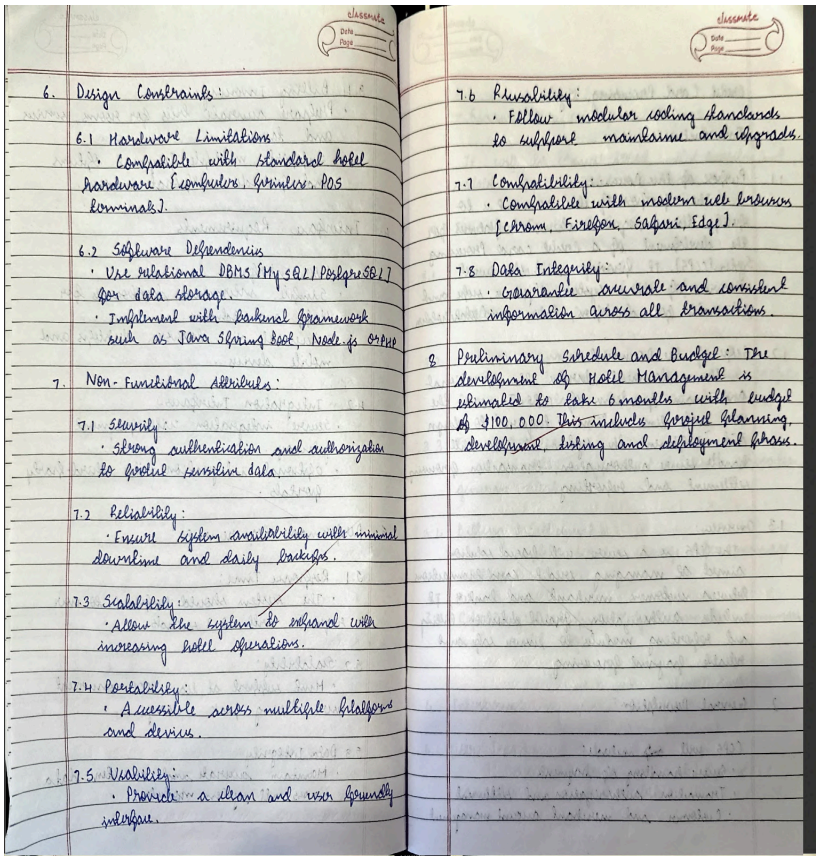
4.2 Integration Interfaces:
- Secure integration with payment gateways.
- Optional integration with third-party portals.

5 Performance Requirements:

5.1 Response Time:
- The system should respond to user requests within 2-3 seconds.

5.2 Scalability:
- Must support at least 1000 concurrent users during peak usage.

5.3 Data Integrity:
- Maintain accurate and consistent data across all system modules.



6. Design Constraints:

6.1 Hardware Limitations
- Compatible with standard hotel hardware [computers, printers, POS terminals].

6.2 Software Dependencies
- Use relational DBMS [MySQL/PostgreSQL] for data storage.
- Implement with backend framework such as Java Spring Boot, Node.js or PHP

7. Non-Functional Attributes:

7.1 Security:
- Strong authentication and authorization to protect sensitive data.

7.2 Reliability:
- Ensure system availability with minimal downtime and daily backups.

7.3 Scalability:
- Allow the system to expand with increasing hotel operations.

7.4 Portability:
- Accessible across multiple platforms and devices.

7.5 Usability:
- Provide a clean and user friendly interface.

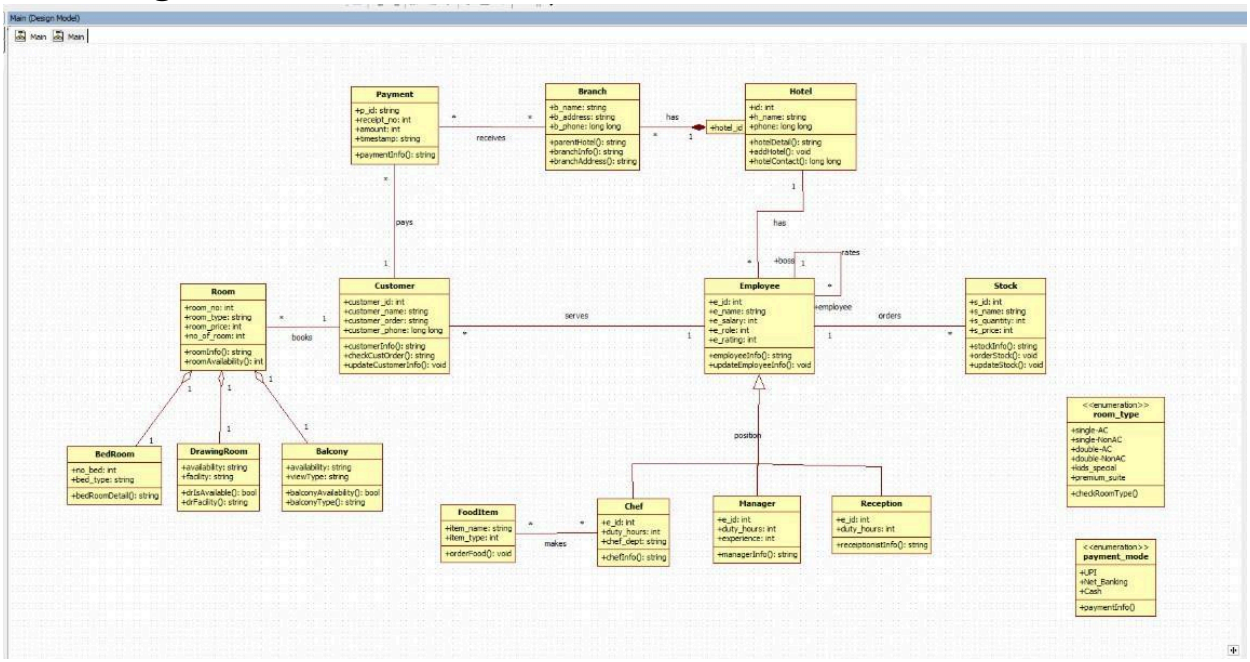
7.6 Reusability:
- Follow modular coding standards to support maintainance and upgrades.

7.7 Compatibility:
- Compatible with modern web browsers [Chrom, Firefox, Safari, Edge].

7.8 Data Integrity:
- Guarantee accurate and consistent information across all transactions.

8 Preliminary Schedule and Budget: The development of Hotel Management is estimated to take 6 months with budget of $100,000. This includes project planning, development, testing and deployment phase.

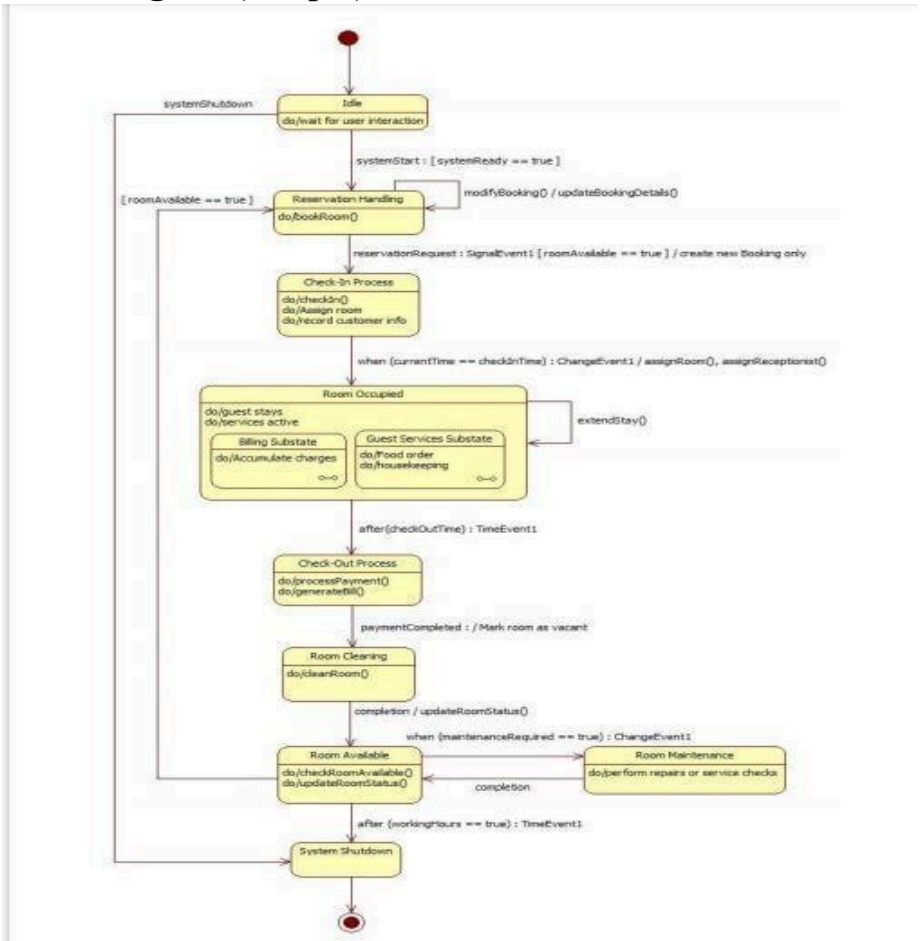
Class Diagram



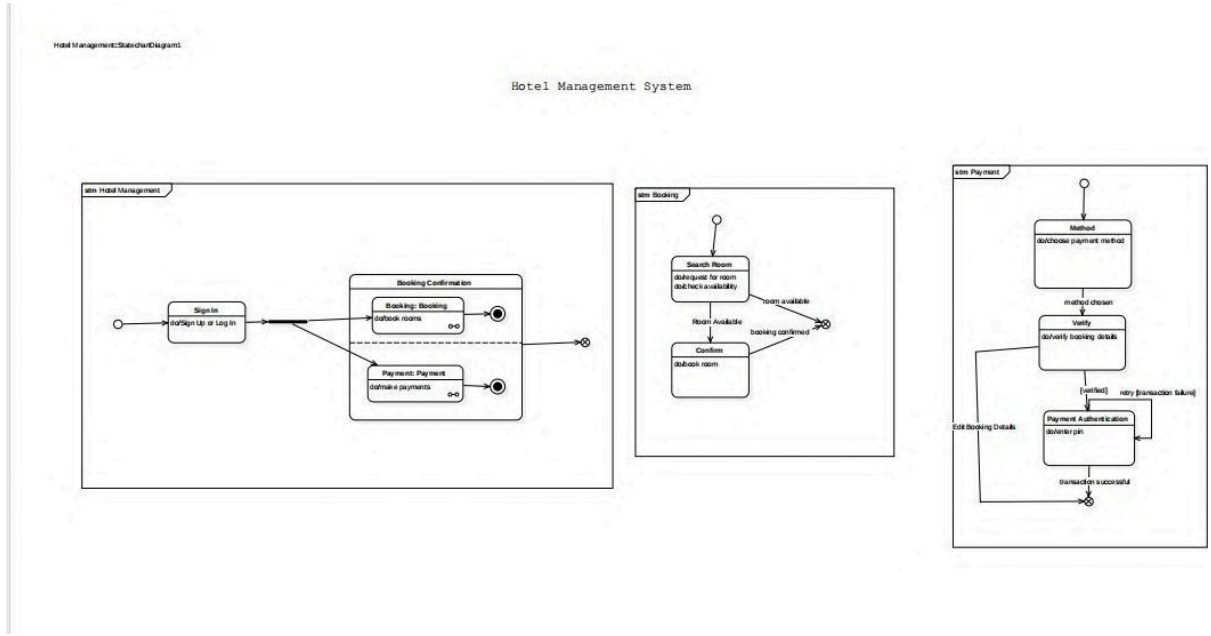
Explanation:

The class diagram for a Hotel Management System provides a detailed structure of the system by defining core classes—Hotel, Branch, Customer, Room, Employee, Payment, FoodItem, and Stock—alongside their attributes, methods, and intricate relationships. A hotel manages multiple branches, each with its own specifics, while customers can book various types of rooms, such as bedrooms or suites, and make secure payments. Employees are organized by roles like chef, manager, or receptionist, streamlining hotel operations, customer service, and food preparation. Additional components track food items and stock, ensuring an integrated approach to inventory and service management. Enumerations for room types and payment modes help standardize and automate bookings and transactions. By capturing both inheritance (e.g., specialized rooms and employee roles) and composition relationships, the diagram ensures a robust and adaptable design for managing hotel workflows efficiently.

State Diagram(Simple)



State Diagram(Advanced)



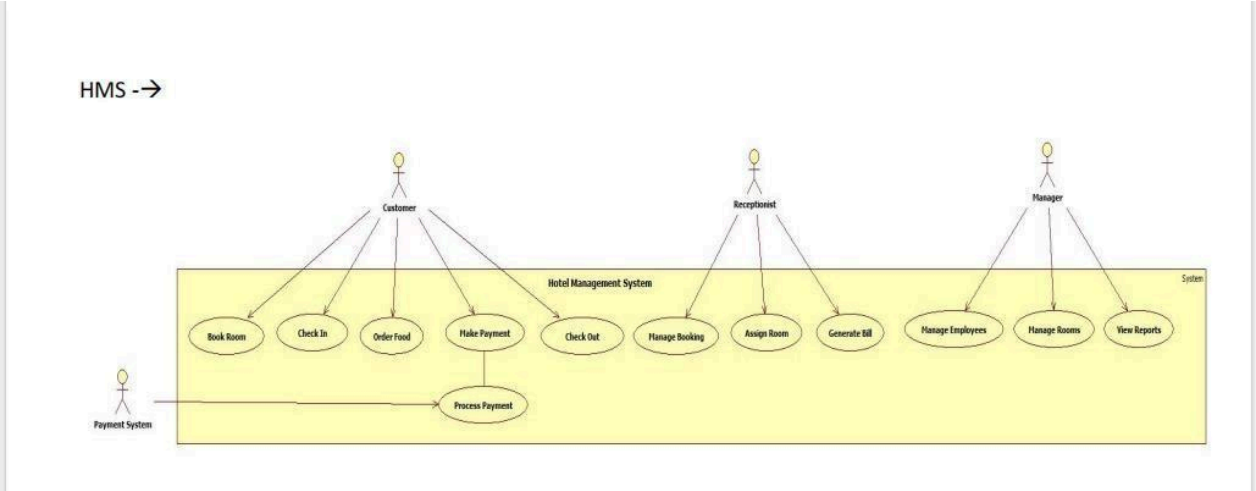
Simple State Diagram (Booking & Payment)

The simple state diagram shows only the basic steps a user follows while booking a room and making a payment. The process starts when the user logs into the system. After logging in, the user searches for available rooms, selects one, and confirms the booking. Once the booking is done, the user chooses a payment method and completes the transaction. This diagram focuses only on the main actions—searching rooms, checking availability, booking, and paying—making it easy to understand the basic workflow.

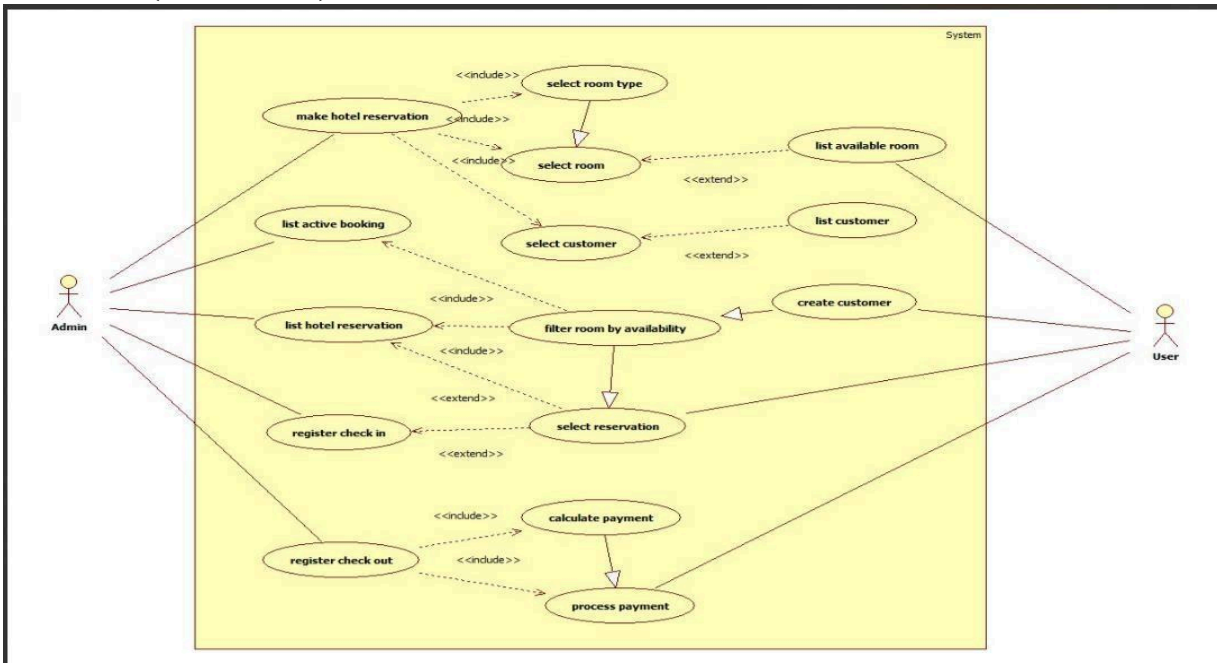
Advanced State Diagram (Full Hotel Room Lifecycle)

The advanced state diagram explains the entire hotel room management process. It begins with the system waiting for the user. When a room is booked, the system handles reservation details and then moves to the check-in process, where customer information is recorded and a room is assigned. During the guest's stay, the system enters the Room Occupied state, which includes billing and guest services like food delivery or housekeeping. After the guest checks out, the room goes through cleaning and, if needed, maintenance. Once completed, the room becomes available again. This diagram shows the full cycle of a room—from booking to check-in, stay, check-out, cleaning, and maintenance—providing a deeper view of hotel operations.

Use Case(Simple)



Use Case(Advanced)



Simple Use Case Diagram

The simple use case diagram shows the basic everyday functions of the Hotel Management System. It involves four main actors: Customer, Receptionist, Manager, and the external Payment System. The Customer can perform common tasks like booking a room, checking in, ordering food, making payments, and checking out. The Receptionist handles bookings, assigns rooms, and prepares bills. The Manager has administrative responsibilities such as managing employees, handling rooms, and viewing reports. The Payment System helps process customer payments. Overall, this diagram focuses only on the essential hotel operations and is easy to understand because it highlights basic interactions between users and the system.

Advanced Use Case Diagram

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

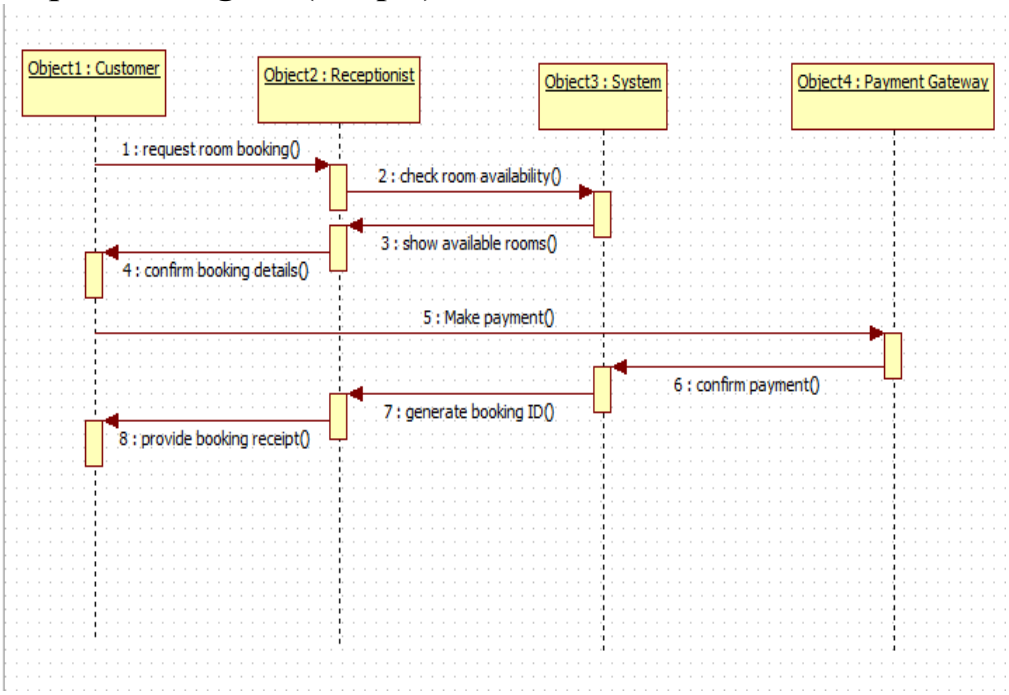
1: Book Room

- 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.

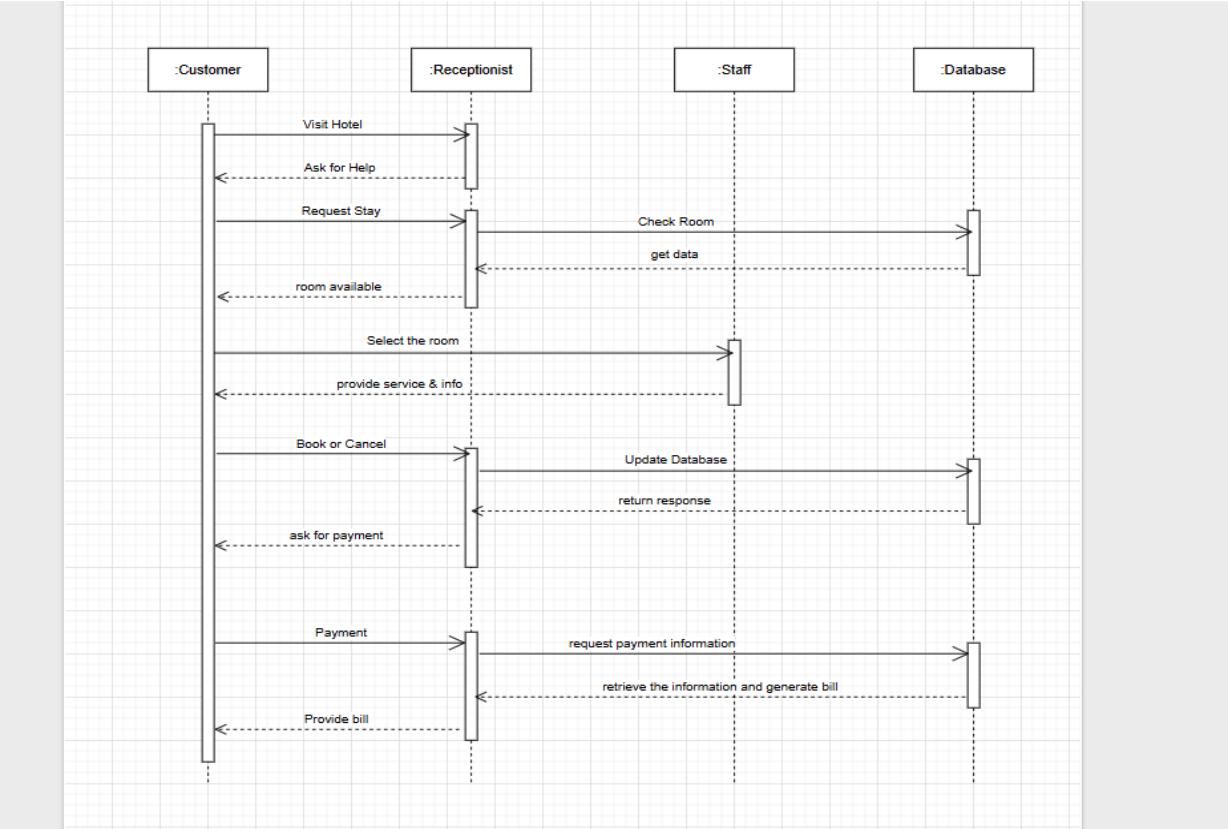
2:Process Payment

- 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.

Sequence Diagram(Simple)



Sequence diagram(Advance)



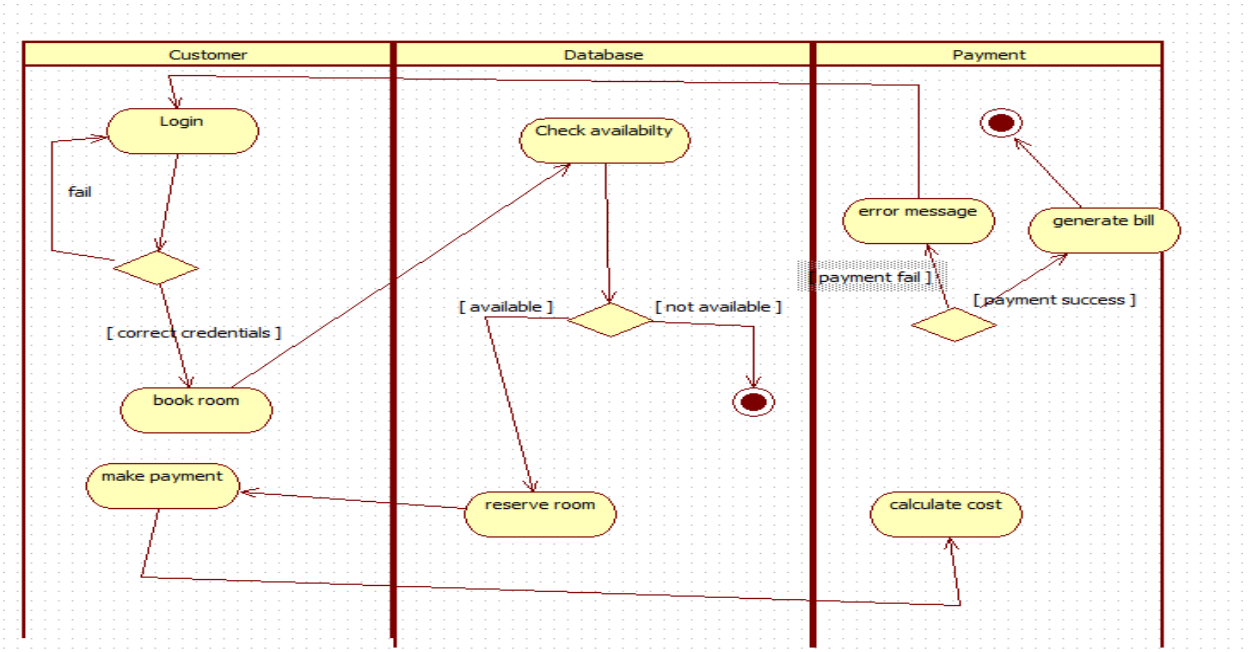
Simple Sequence Diagram (Customer–Receptionist–System–Payment Gateway)

The simple sequence diagram explains the basic steps involved in booking a room. The process begins when the Customer asks the Receptionist to book a room. The receptionist checks availability by sending a request to the System, which returns a list of available rooms. After seeing the options, the customer confirms the booking.Next, the customer proceeds to make the payment. The receptionist sends the payment request to the Payment Gateway, which verifies the transaction and returns the result. If the payment is successful, the system creates a booking ID, and the receptionist gives the customer a booking receipt.This diagram focuses only on the key tasks— checking rooms, confirming booking, and processing payment—without showing extra steps or internal system details.

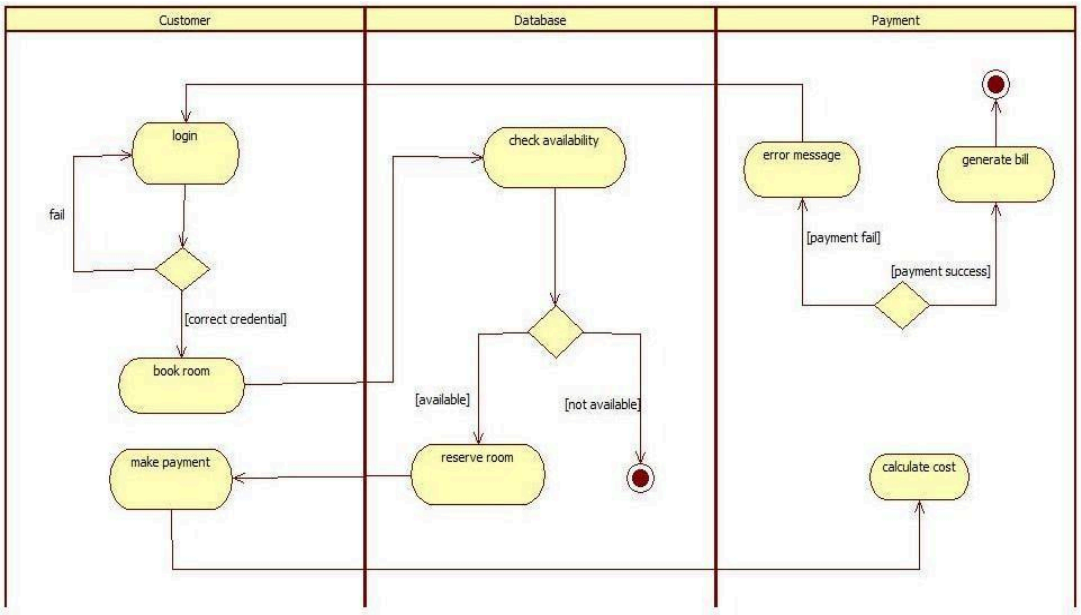
Advanced Sequence Diagram (Customer–Receptionist–Staff–Database)

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.

Activity Diagram(Simple)



Activity Diagram(Advanced)



Simple Activity Diagram

The simple activity diagram shows the basic steps a customer follows to book a room and make a payment. It includes three main actions:

- Login: The customer enters their details. If the login fails, they try again; if it succeeds, they move forward.
- Book Room: The system checks room availability. If no rooms are available, the process ends. If a room is available, it is reserved.
- Make Payment: The customer pays for the booking. If the payment fails, an error message appears. If it succeeds, the system generates the final bill.

This simple diagram focuses only on the essential steps—login, booking, and payment—making it easy for beginners to understand the basic flow.

Advanced Activity Diagram

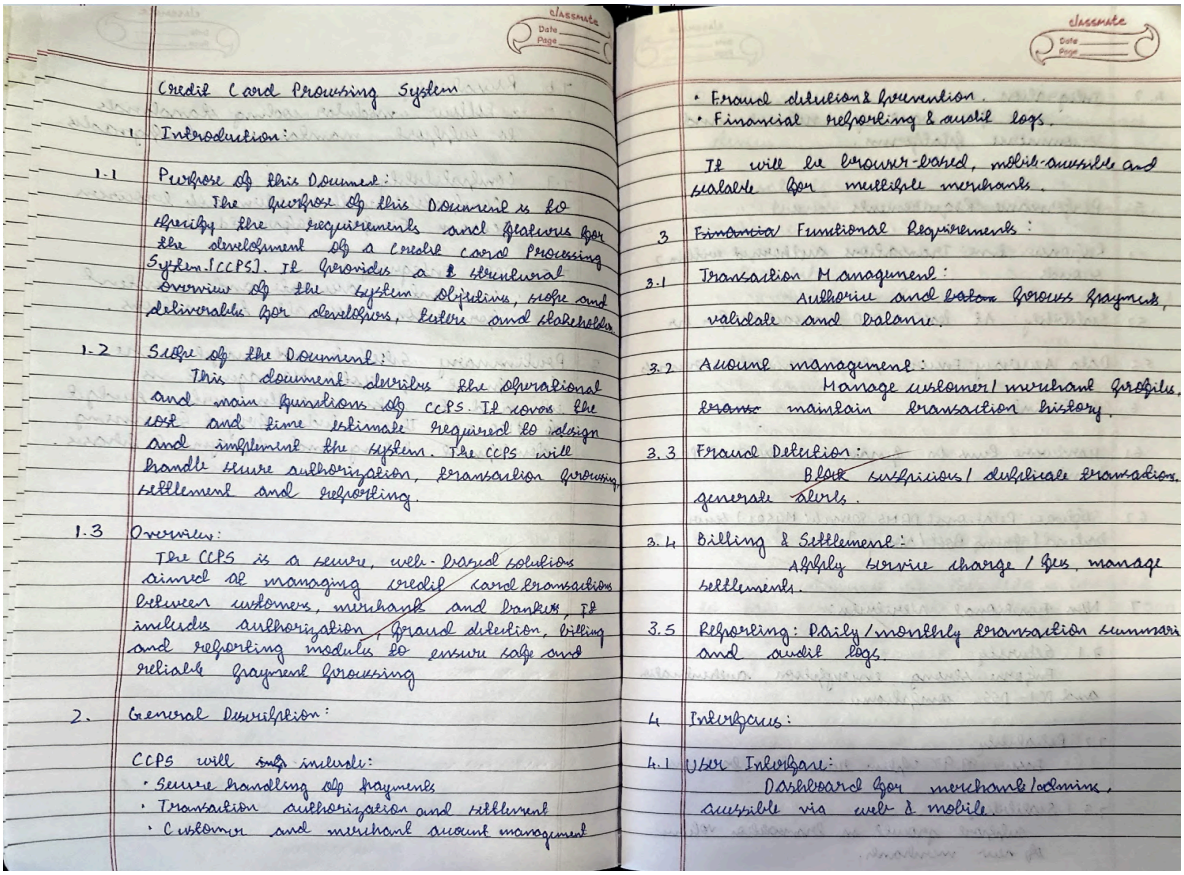
The advanced activity diagram shows the same process in more detail and separates the actions into three swimlanes: Customer, Database, and Payment.

- 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



Credit Card Processing System

1 Introduction:

1.1 Purpose of this Document:
The purpose of this Document is to specify the requirements and features for the development of a Credit Card Processing System [CCPS]. It provides a structural overview of the system objectives, scope and deliverables for developers, testers and stakeholders.

1.2 Scope of the Document:
This document describes the operational and main functions of CCPS. It covers the cost and time estimate required to design and implement the system. The CCPS will handle secure authorization, transaction processing, settlement and reporting.

1.3 Overview:
The CCPS is a secure, web-based solution aimed at managing credit card transactions between customers, merchants and banks. It includes authorization, fraud detection, billing and reporting modules to ensure safe and reliable payment processing.

2. General Description:

CCPS will include:
- Secure handling of payments
- Transaction authorization and settlement
- Customer and merchant account management
- Fraud detection & prevention.
- Financial reporting & audit logs.

It will be browser-based, mobile-accessible and scalable for multiple merchants.

3 Financial Functional Requirements:

3.1 Transaction Management:
Authorize and batch process payments, validate and balance.

3.2 Account management:
Manage customer/merchant profiles, maintain transaction history.

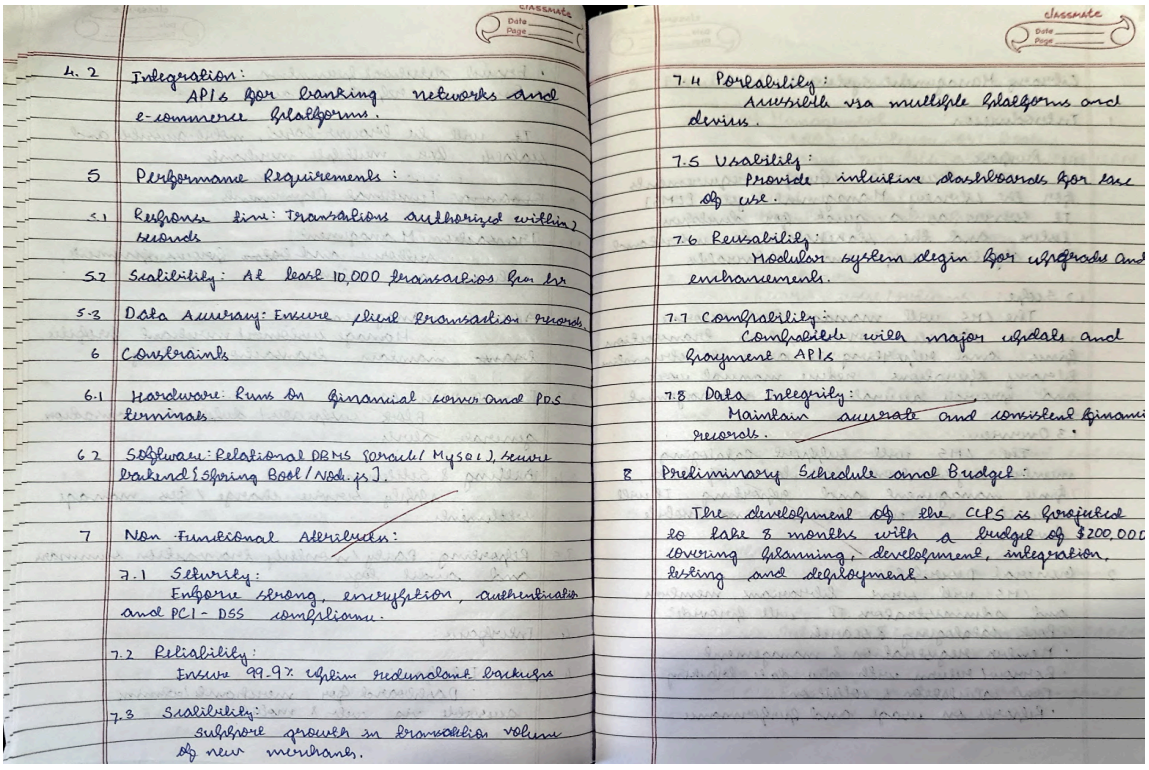
3.3 Fraud Detection:
Block suspicious/duplicate transaction, generate alerts.

3.4 Billing & Settlement:
Apply service charge / fee, manage settlements.

3.5 Reporting: Daily/monthly transaction summary and audit logs.

4 Interfaces:

4.1 User Interface:
Dashboard for merchant/admin, accessible via web & mobile.



4.2 Integration:
APIs for banking networks and e-commerce platforms.

5 Performance Requirements:

5.1 Response time: Transactions authorized within 2 seconds.

5.2 Scalability: At least 10,000 transactions per hr

5.3 Data Accuracy: Ensure client transaction records

6 Constraints

6.1 Hardware: Runs on financial server and POS terminals

6.2 Software: Relational DBMS (Oracle/ MySQL), secure backend (Spring Boot / Node.js).

7 Non Functional Attributes:

7.1 Security:
Enforce strong encryption, authentication and PCI-DSS compliance.

7.2 Reliability:
Ensure 99.9% uptime redundant backups

7.3 Scalability:
Support growth in transaction volume of new merchants.

7.4 Portability:
Accessible via multiple platforms and devices.

7.5 Usability:
Provide intuitive dashboards for ease of use.

7.6 Reusability:
Modular system design for upgrades and enhancements.

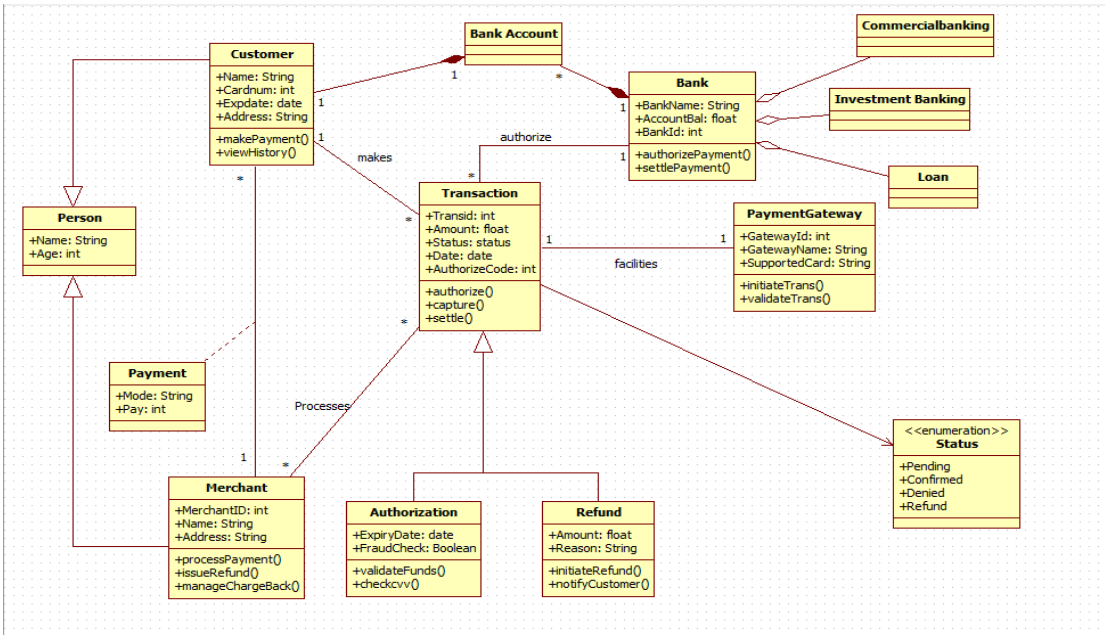
7.7 Compatibility:
Compatible with major wallets and payment APIs

7.8 Data Integrity:
Maintain accurate and consistent financial records.

8 Preliminary Schedule and Budget:

The development of the CCPS is projected to take 8 months with a budget of $200,000 covering planning, development, integration, testing and deployment.

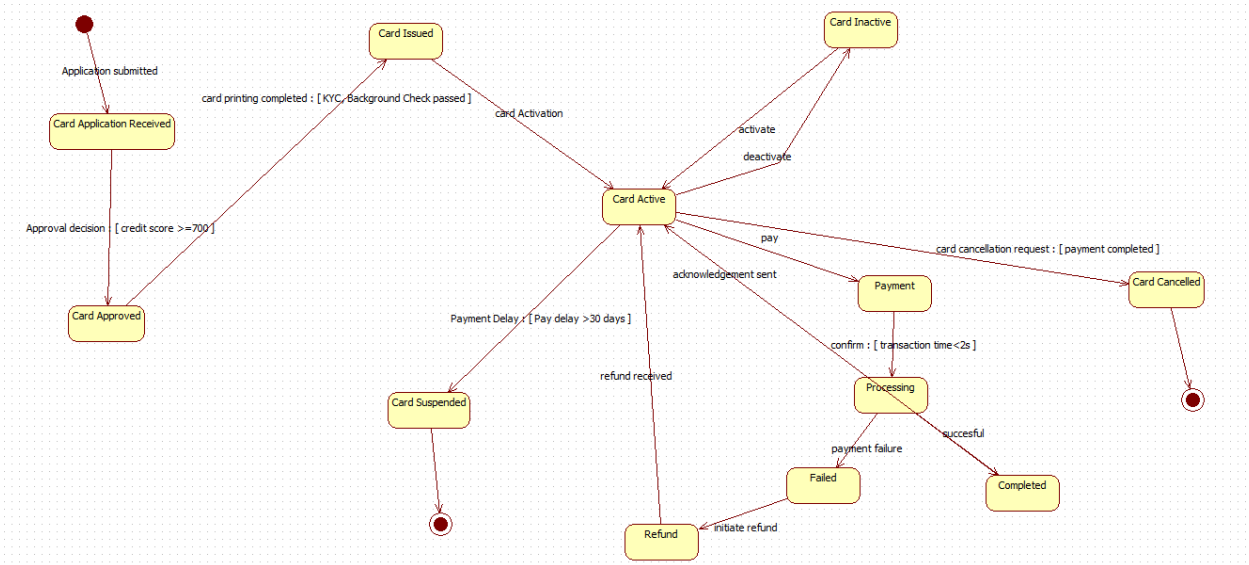
Class Diagram



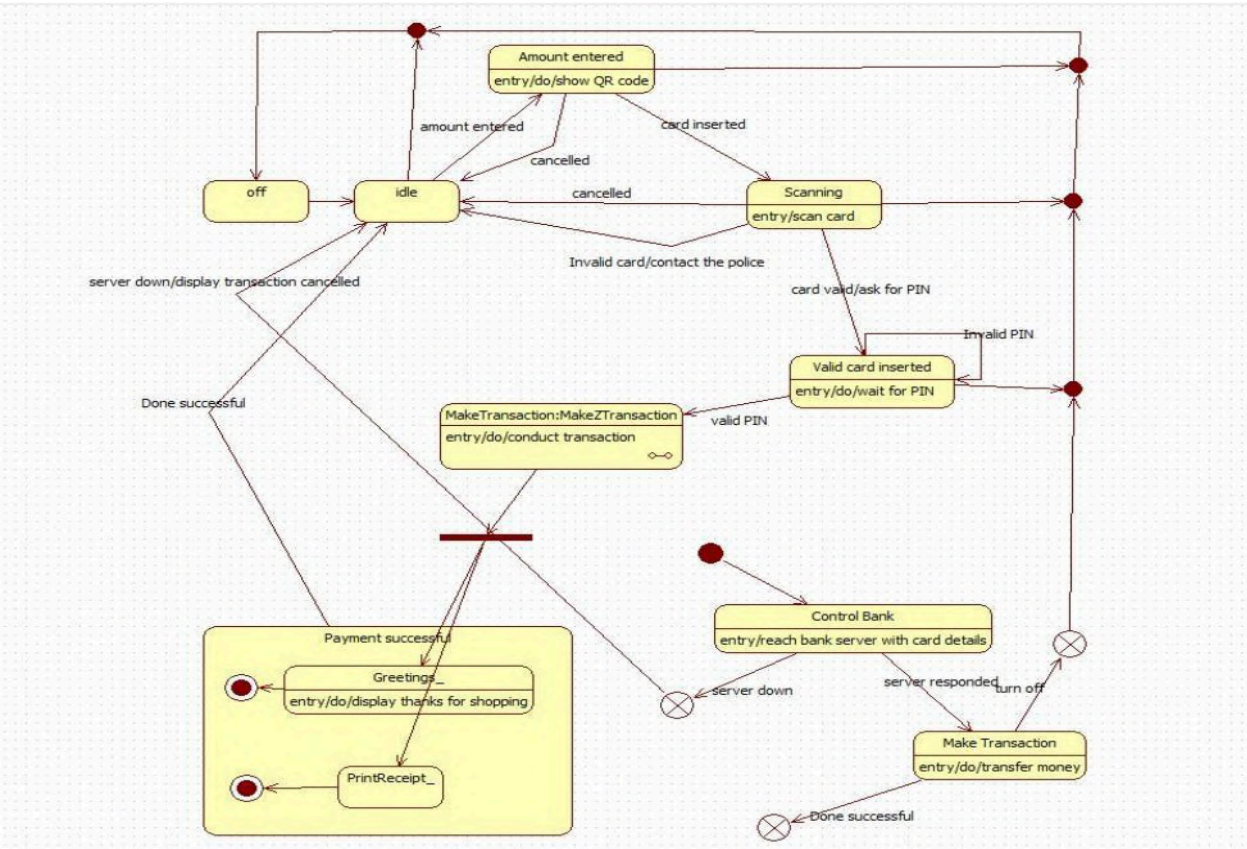
Explanation

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(Simple)



State Diagram(Advanced)



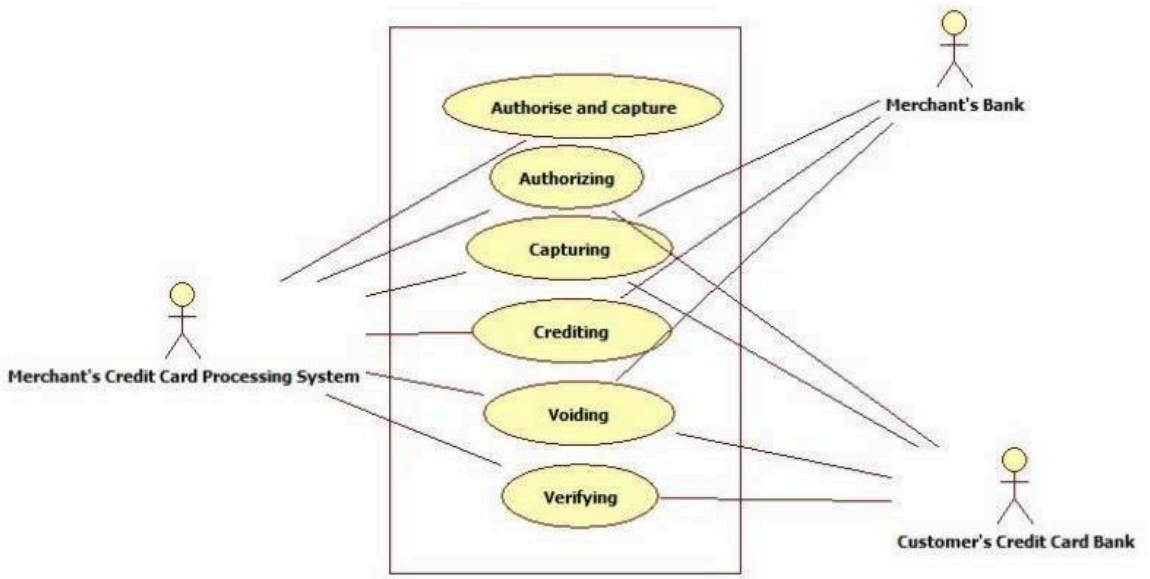
Simple State Diagram (Card Lifecycle & Payment Flow)

The simple state diagram shows the basic life cycle of a credit/debit card, starting from the application stage and ending with payment completion or card cancellation. The flow begins when a user submits an application and the system receives it. If the user meets the approval conditions (such as having a sufficient credit score), the card is approved and then issued. Once the card becomes Active, the user can make payments. Each payment goes through a processing state, where it can either succeed or fail. Successful payments lead to a Completed state, while failures move to a Failed state. The diagram also includes paths for refund, payment delays (which may suspend the card), and cancellation requests. This simple diagram focuses mainly on card activation, payment, success/failure, and cancellation.

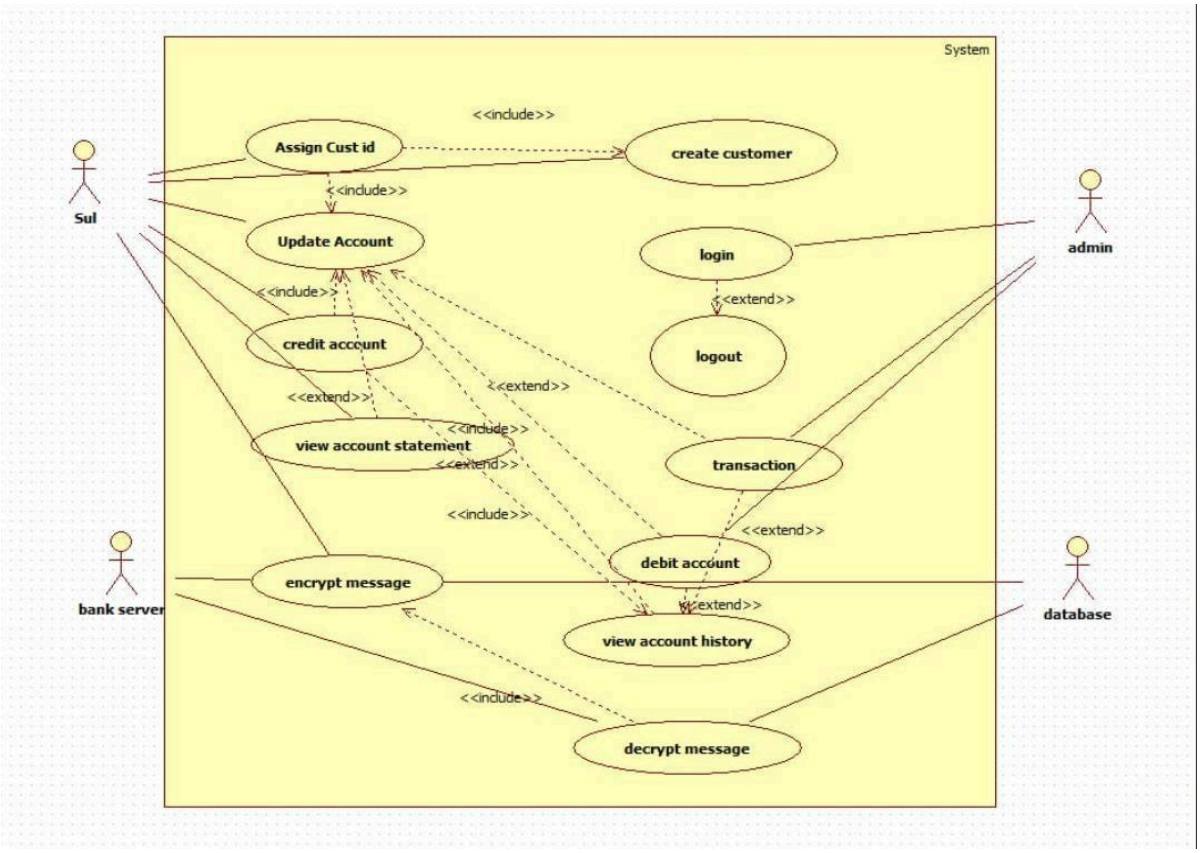
Advanced State Diagram (ATM/Payment Terminal Transaction Flow)

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 scanning 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(Simple)



Use Case(Advanced)



Simple Use Case Diagram (Credit Card Processing System)

The simple use case diagram focuses on the main functions involved in credit card processing between a merchant and the banks. The primary actor is the Merchant's Credit Card Processing System, which communicates with both the Merchant's Bank and the Customer's Credit Card Bank. The system performs core actions such as Authorizing, Capturing, Crediting, Voiding, and Verifying transactions. These actions represent the essential steps required to approve a payment, capture funds, issue credit, reverse a payment, or validate card details. Since the diagram only shows the main operations without detailing internal workflows, it provides an easy-to-understand overview of how basic credit card transactions are handled.

Advanced Use Case Diagram (Banking Customer Management System)

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

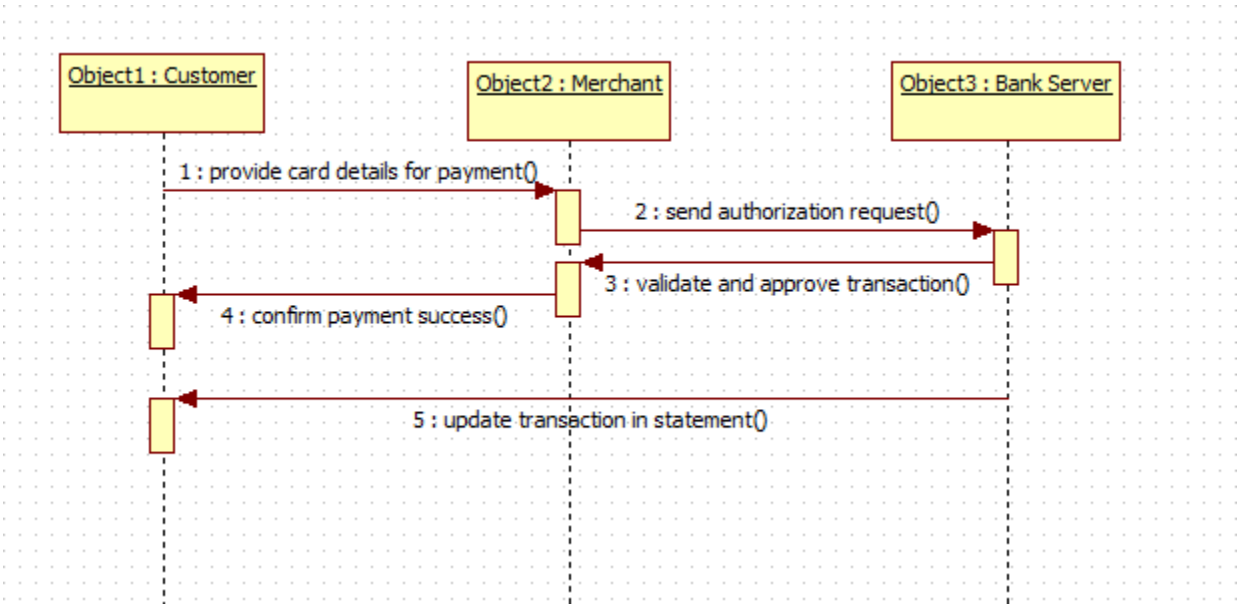
address. The system validates the information.

A unique customer ID is assigned automatically.

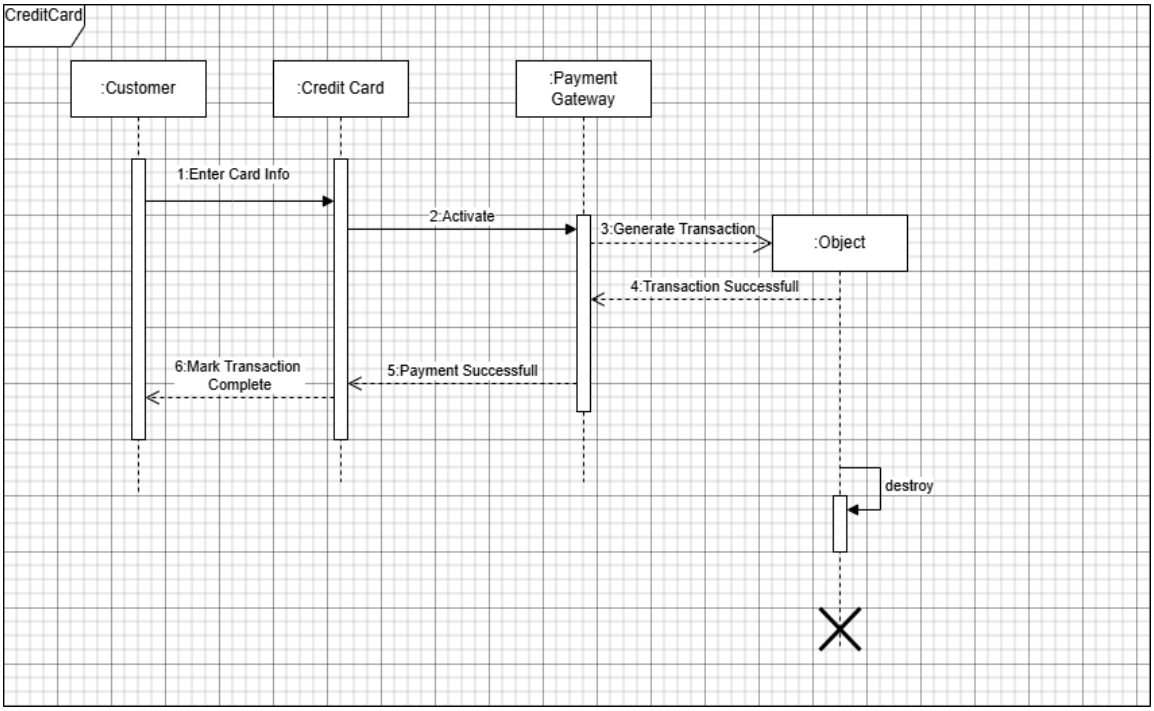
The customer account is created and stored in the database.

The system confirms successful registration.

Sequence Diagram(Simple)



Sequence Diagram(Advanced)



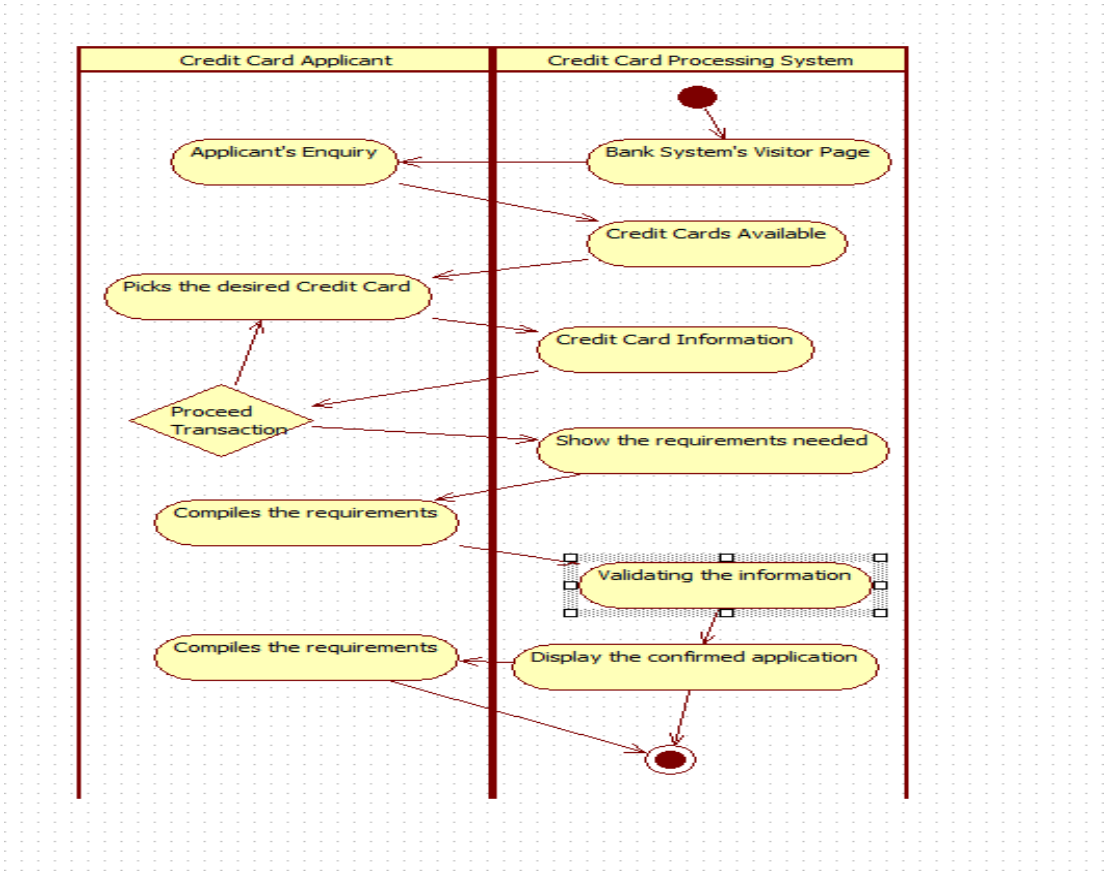
Simple Sequence Diagram (Customer – Credit Card – Payment Gateway)

The simple sequence diagram shows the basic steps involved when a customer makes a payment using a credit card. The process starts when the Customer enters card information, which is then sent to the Credit Card system. The credit card activates the request and sends it to the Payment Gateway. The payment gateway generates the transaction and forwards it to the internal processing object. Once the transaction is approved, a Transaction Successful message is sent back through the payment gateway to the credit card system. The credit card then informs the customer that the payment is successful, and the transaction is marked as complete. This diagram only focuses on the essential steps: entering card data, verifying payment, and completing the transaction.

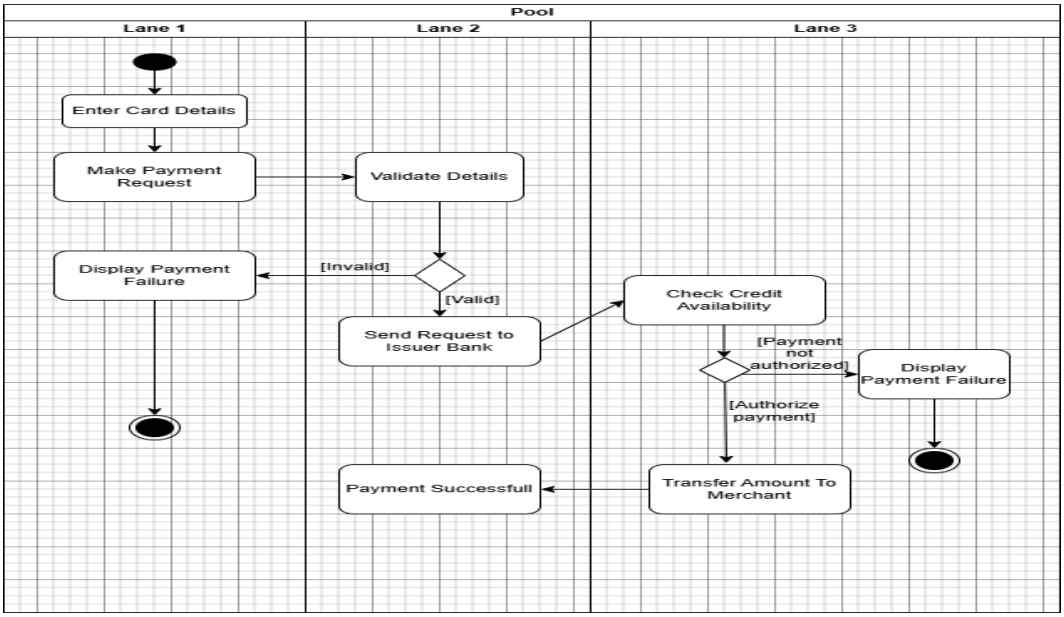
Advanced Sequence Diagram (Customer – Merchant – Bank Server)

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(Simple)



Activity Diagram(Advanced)



Simple Activity Diagram (Credit Card Payment Workflow)

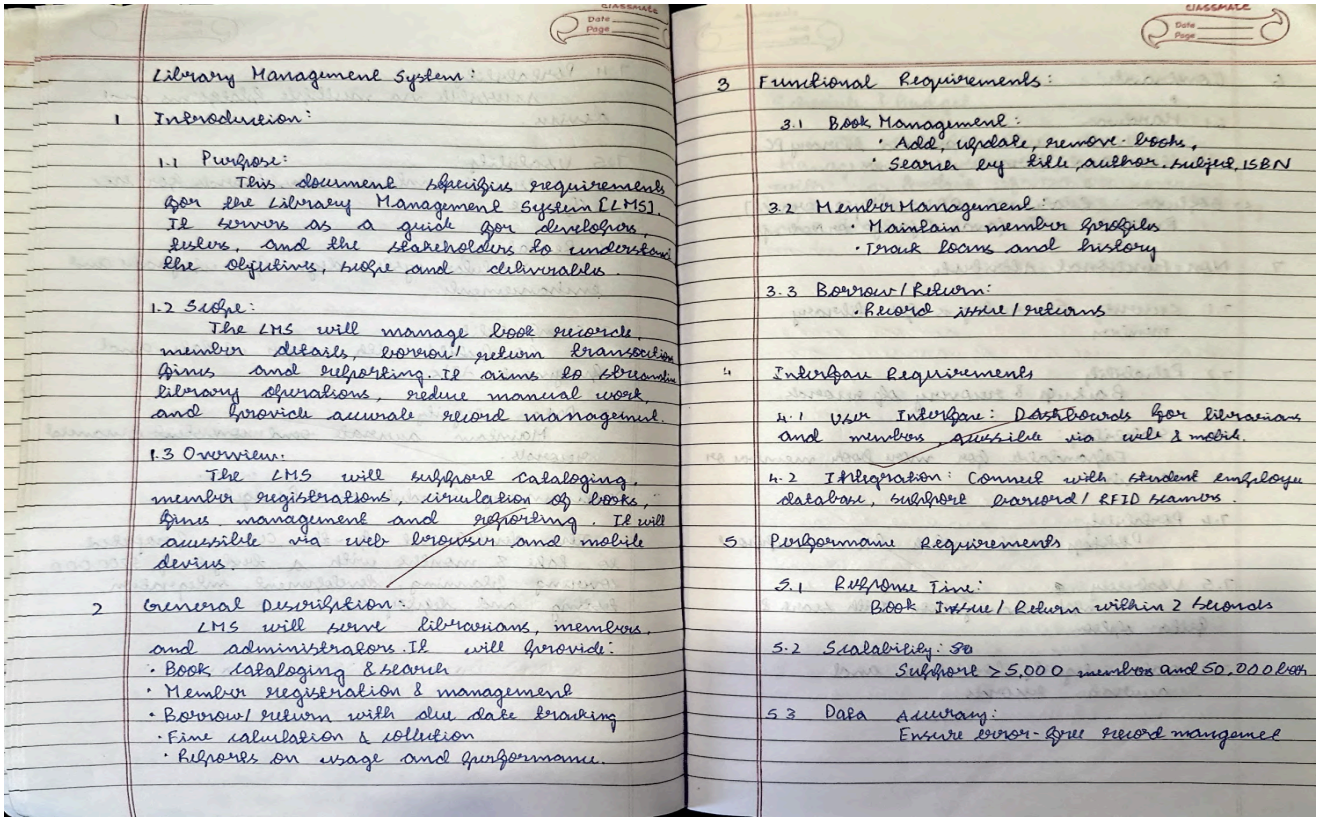
The simple activity diagram explains the basic steps in completing a credit card payment. The process starts when the customer enters card details and submits a payment request. The system then validates the details. If the card information is invalid, the system immediately displays a payment failure message. If the details are valid, the request is sent to the issuer bank, where the system checks the customer's available credit. If the bank does not authorize the payment, failure is shown again. If everything is correct, the bank authorizes the transaction, and the amount is transferred to the merchant. Finally, the user is shown a payment successful message. This diagram focuses only on the main steps—entering details, validating, checking credit, and approving or rejecting the payment.

Advanced Activity Diagram (Credit Card Application & Processing System)

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:
This document specifies requirements for the Library Management System [LMS]. It serves as a guide for developers, testers, and the stakeholders to understand the objectives, scope and deliverables.

1.2 Scope:
The LMS will manage book records, member details, borrow/return transactions, fines and reporting. It aims to streamline library operations, reduce manual work, and provide accurate record management.

1.3 Overview:
The LMS will support cataloging, member registrations, circulation of books, fines management and reporting. It will be accessible via web browser and mobile devices.

2 General Description:
LMS will serve librarians, members, and administrators. It will provide:
- Book cataloging & search
- Member registeration & management
- Borrow/return with due date tracking
- Fine calculation & collection
- Reports on usage and performance.

3 Functional Requirements:

3.1 Book Management:
- Add, update, remove books.
- Search by title, author, subject, ISBN

3.2 Member Management:
- Maintain member profiles
- Track loans and history

3.3 Borrow/Return:
- Record issue/returns

4 Interface Requirements

4.1 User Interface: Dashboards for librarians and members, accessible via web & mobile.

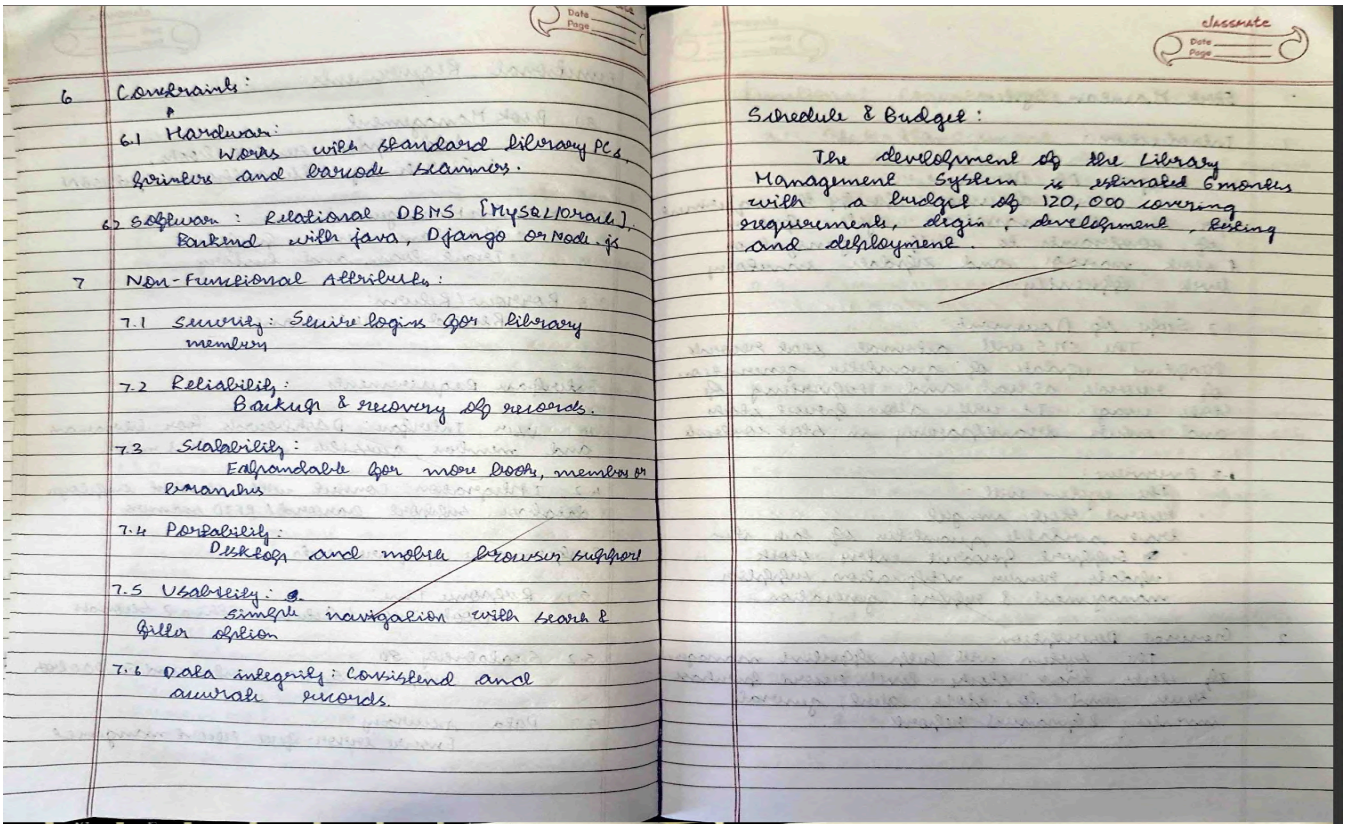
4.2 Integration: Connect with student employee database, support barcode/RFID scanners.

5 Performance Requirements

5.1 Response Time:
Book Issue/Return within 2 seconds

5.2 Scalability: Su
Support > 5,000 members and 50,000 boo

5.3 Data Accuracy:
Ensure error-free record management



6 Constraints:

6.1 Hardware:
Works with standard library PCs, printers and barcode scanners.

6.2 Software: Relational DBMS [MySQL/Oracle].
Backend with Java, Django or Node.js

7 Non-Functional Attributes:

7.1 Security: Secure logins for library members

7.2 Reliability:
Backup & recovery of records.

7.3 Scalability:
Expandable for more books, members or branches

7.4 Portability:
Desktop and mobile browser support

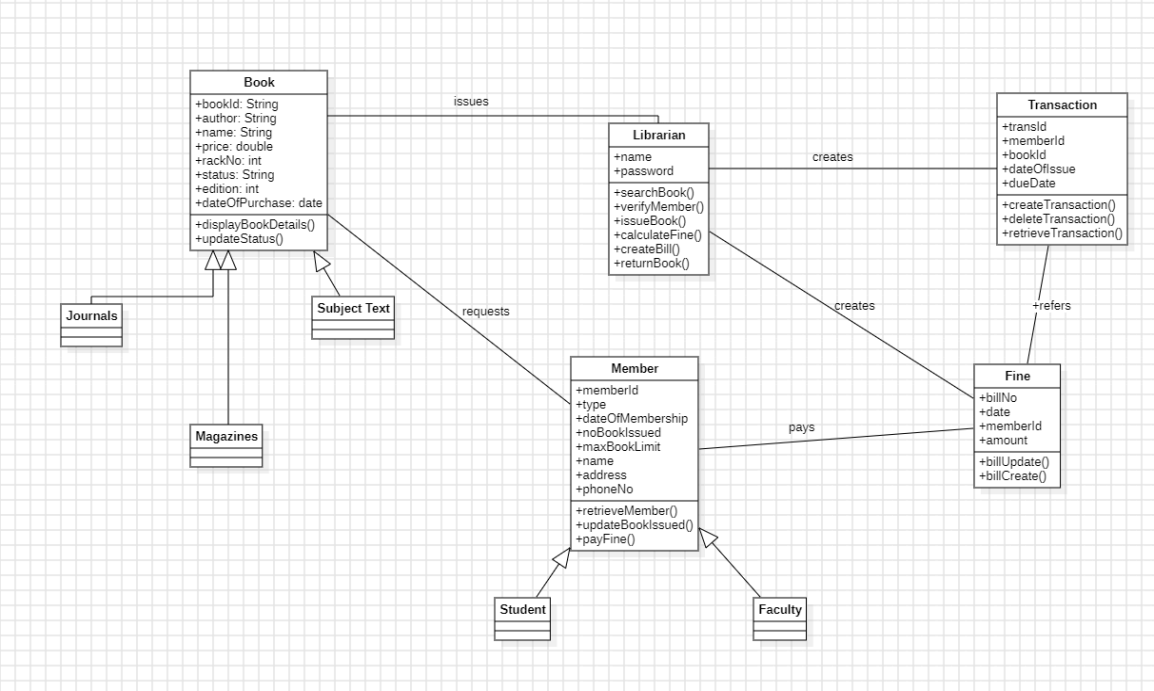
7.5 Usability:
simple navigation with search & filter option

7.6 Data integrity: Consistend and accurate records.

Schedule & Budget:

The development of the Library Management System is estimated 6 months with a budget of 120,000 covering requirements, design, development, testing and deployment.

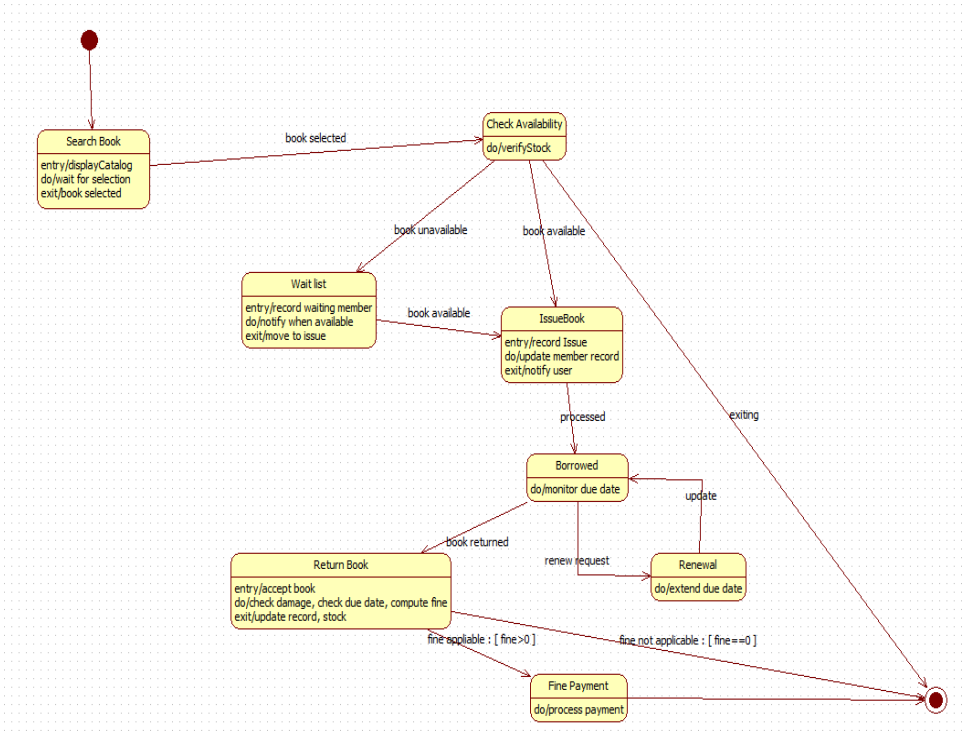
Class Diagram



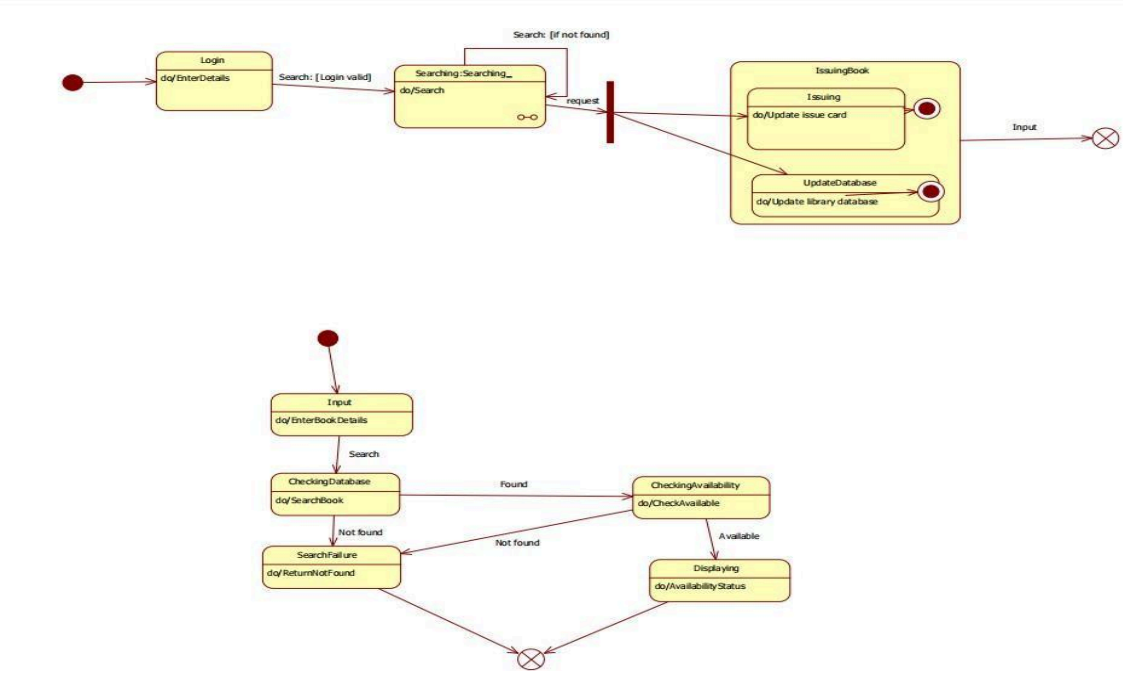
The class diagram represents a Library Management System where books, members, librarians, and transactions interact to manage library operations. The Book class stores details such as author, edition, price, and purchase date, and it is specialized into different types like Journals, Magazines, and 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(Simple)



State Diagram(Advanced)



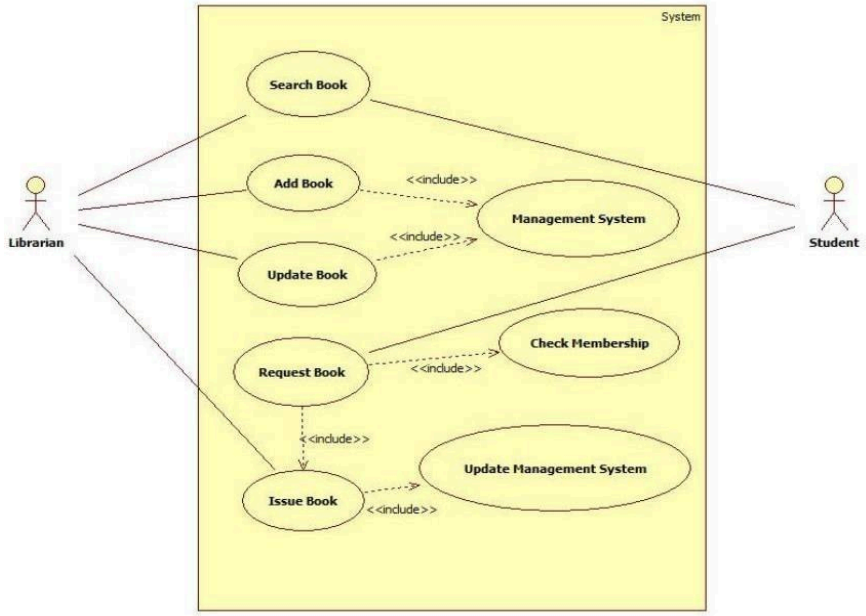
Simple State Diagram

The simple state diagram shows the basic steps involved in searching for and issuing a book in the library system. The process begins when the user logs in and enters book details. The system checks the database to see whether the book exists. If the book is not found, the system displays a failure message and the process ends. If the book is found, the system checks its availability. If the book is available, the issuing process begins, where the system updates the library records and the issue card. If the book is unavailable, the state transitions to a not-found or failure end state. This simple diagram focuses only on the essential steps: login, search, availability check, and issuing the book, making it easy to understand the basic workflow.

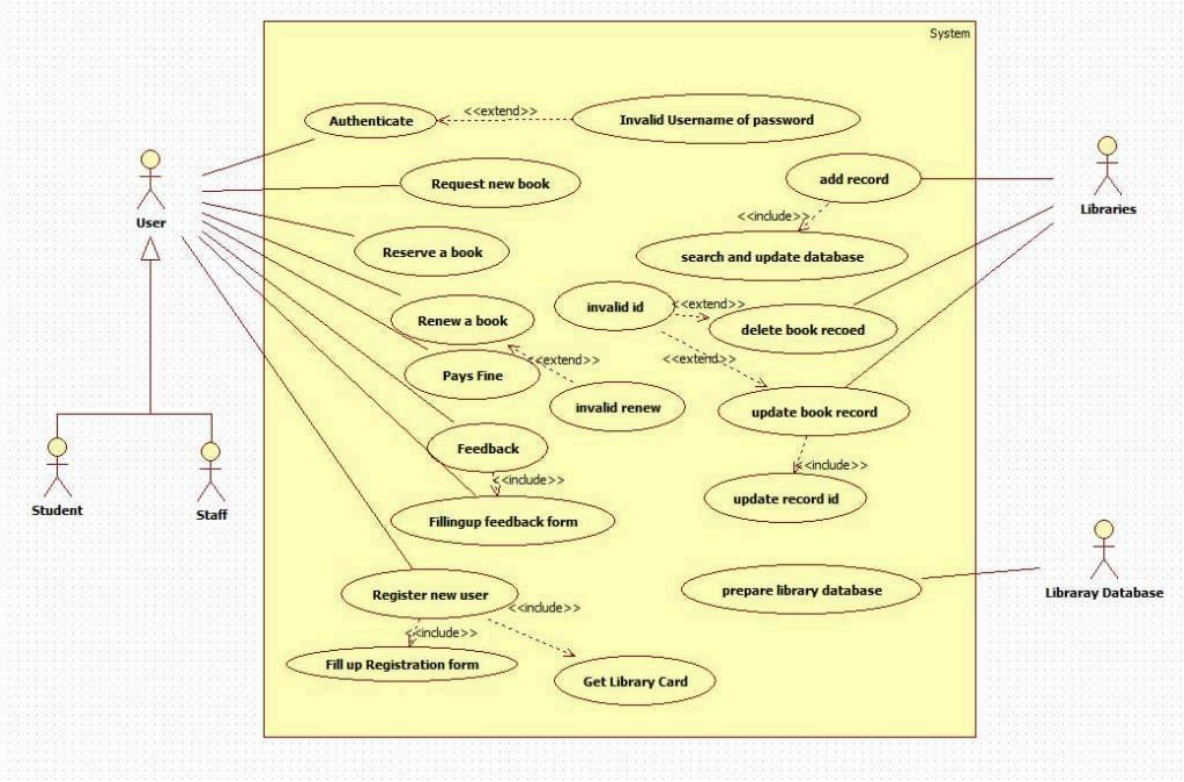
Advanced State Diagram

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 user 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.

UseCase(Simple)



UseCase(Advanced)



Simple Use Case Diagram

The simple use case diagram focuses on the core interactions in the Library Management System between the Librarian and the Student. The librarian performs essential tasks such as Searching Books, Adding Books, Updating Book Details, Requesting Books, and Issuing Books. Many of these actions include interactions with the Management System, such as checking membership and updating records. The student mainly interacts through the system to request or search for books. The diagram highlights only the fundamental operations that support the basic flow of library activities—finding, adding, updating, and issuing books—making it easy to understand the primary responsibilities of the librarian and how students request books.

Advanced Use Case Diagram

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.

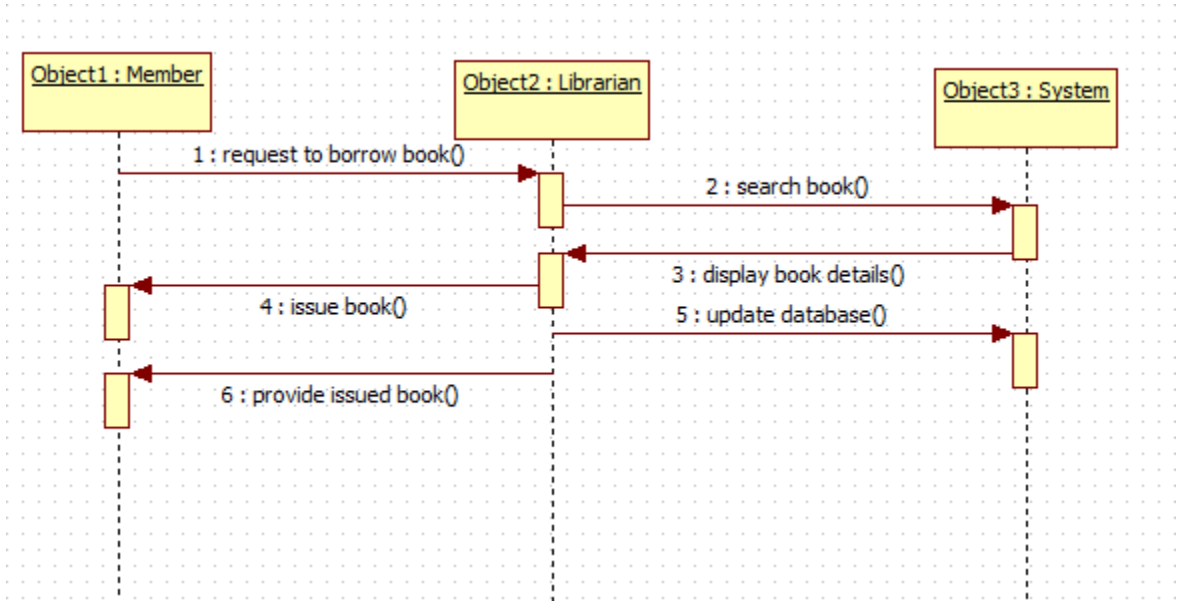
The system prompts for the book title/author/ID.

The student enters the details.

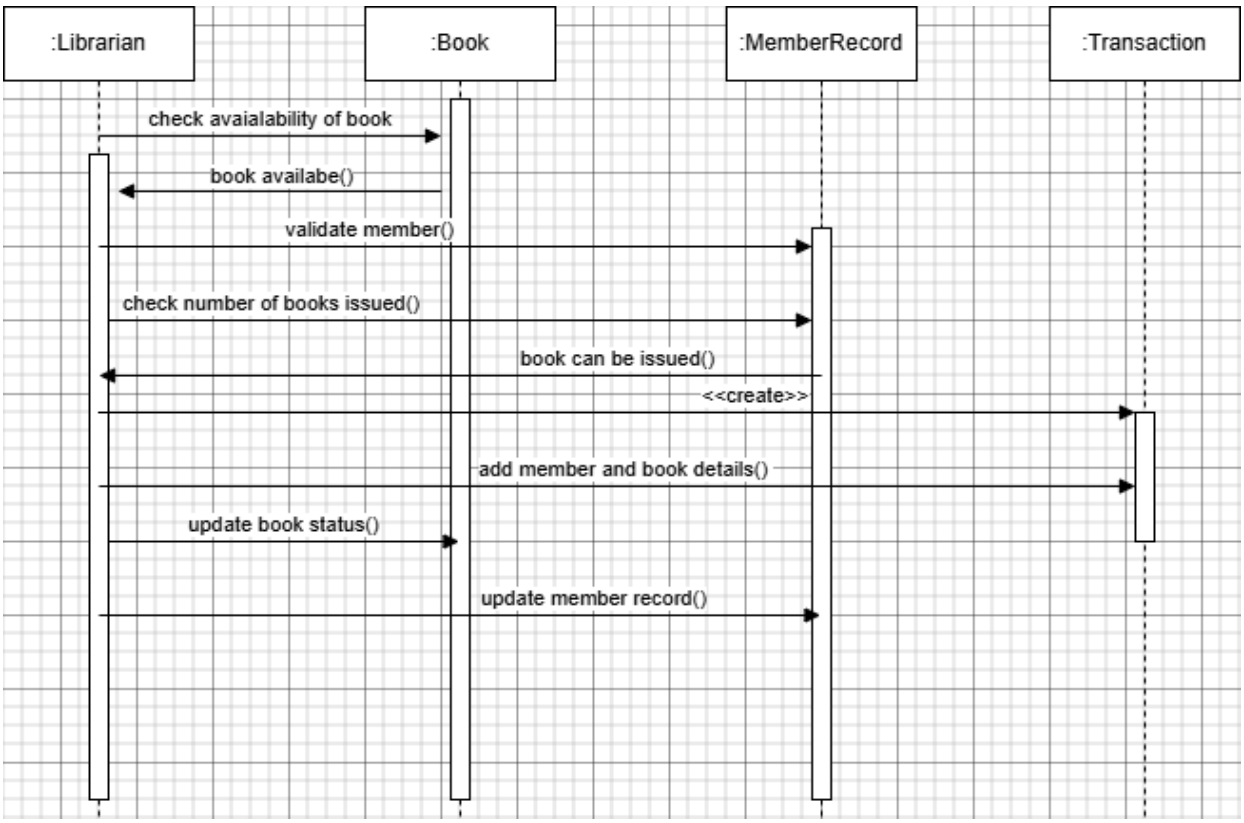
The system scans the database for matching books.

The system displays the availability status (Available / Issued / Not Found).

Sequence Diagram(Simple)



Sequence Diagram(Advanced)



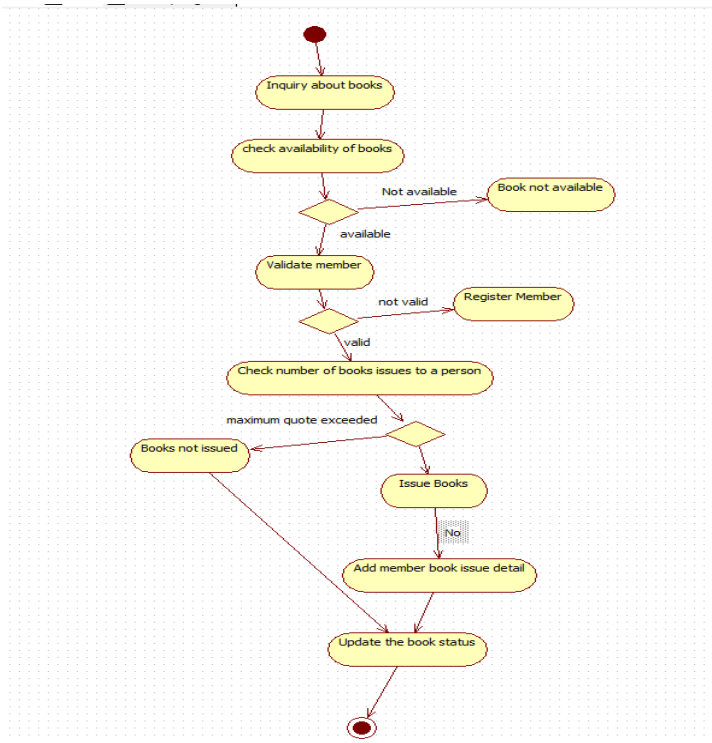
Simple Sequence Diagram (Member – Librarian – System)

The simple sequence diagram shows the basic process of borrowing a book in the library. The interaction begins when the Member requests to borrow a book. The Librarian receives this request and sends a command to the System to search for the book. The system returns the book details, after which the librarian proceeds to issue the book to the member. Once the book is issued, the librarian updates the library database, and finally the member receives the issued book. This diagram focuses only on the essential steps—requesting, searching, issuing, and updating— making it easy to understand how a book is borrowed in a simple workflow.

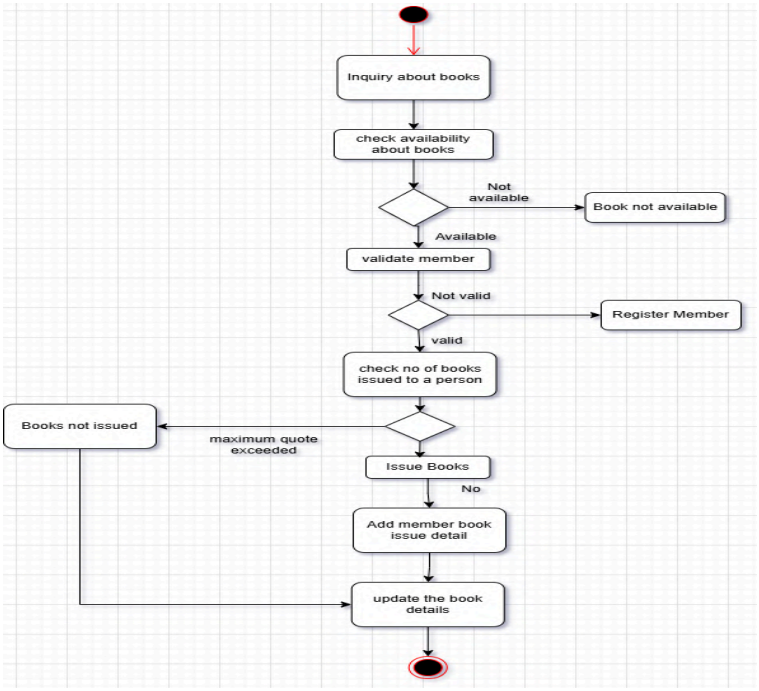
Advanced Sequence Diagram (Librarian – Book – MemberRecord – Transaction)

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(Simple)



Activity Diagram(Advanced)



Simple Activity Diagram

The simple activity diagram shows the basic steps followed when a user wants to issue a book. The process starts with an inquiry about the book, followed by checking book availability. If the book is not available, the process ends. If it is available, the system then validates the member. If the member is not valid, they are asked to register. After validation, the system checks how many books the member has already borrowed. If the maximum limit is exceeded, the book is not issued. Otherwise, the book is issued, and the system updates the book status. This simple diagram covers only the main flow of requesting, verifying, and issuing a book.

Advanced Activity Diagram

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:

Stock Maintain System:

1 Introduction:

1.1 Purpose Of Document:
This document specify the requirements for the stock maintain system feature, of constraints to help organization & stock, manage and update inventory levels efficiently.

1.2 Scope of Document:
The SMS will automate stock records, realtime update of quantities, generation of records alert and reporting of stocks usage. It will allow precise stocks and ensure transparency in stock control.

1.3 Overview:
The system will:
- record stock insight
- track avilable quantities of each item
- Support product entry, stock updats, render notifications, supplier management & report generation.

2 General Description:
The system will help effective management of stocks, track stocks level, record purchase issues, send low-stock alert, generate inventory & financial reports.

Functional Requirements:

3.1 Stock Management:
- Add, Update & delete product records.

3.2 Supplier Management
- Record supplier details, purchase history.

3.3 Purchase & Issue Tracking:
- Log purchase entries and stock addition
- Record product issues to departments or customers.

3.4 Reorder Management:
- Alert users when stock reaches minimum
- Generate purchase order automatically.

3.5 Reporting:
Generate stock summary reports.

4 Interface Requirement

4.1 User Interface:
Interactive dashboards showing analysis and statistics

4.2 Integration Interfaces:
Integration with accounting software purchase / expense tracking.

5 Performance Requirements:
- The system should update stock transactions within 2 seconds
- support up to 10,000 product records
- multiple concurrent users

6 Design Constraints:

6.1 Hardware Limitations:
- Must support standard PCs, barcode scanners & printers

6.2 Software Dependencies:
- Relational DBMS (MySQL (Oracle)).
- Computer language technology

7. Non-Functional Attributes:

7.1 Security:
Role-based access to restrict unauthorized services.

7.2 Reliability:
System should ensure backup and recovery of stocks.

7.3 Scalability:
Allow adding new products, supplies & warehouses.

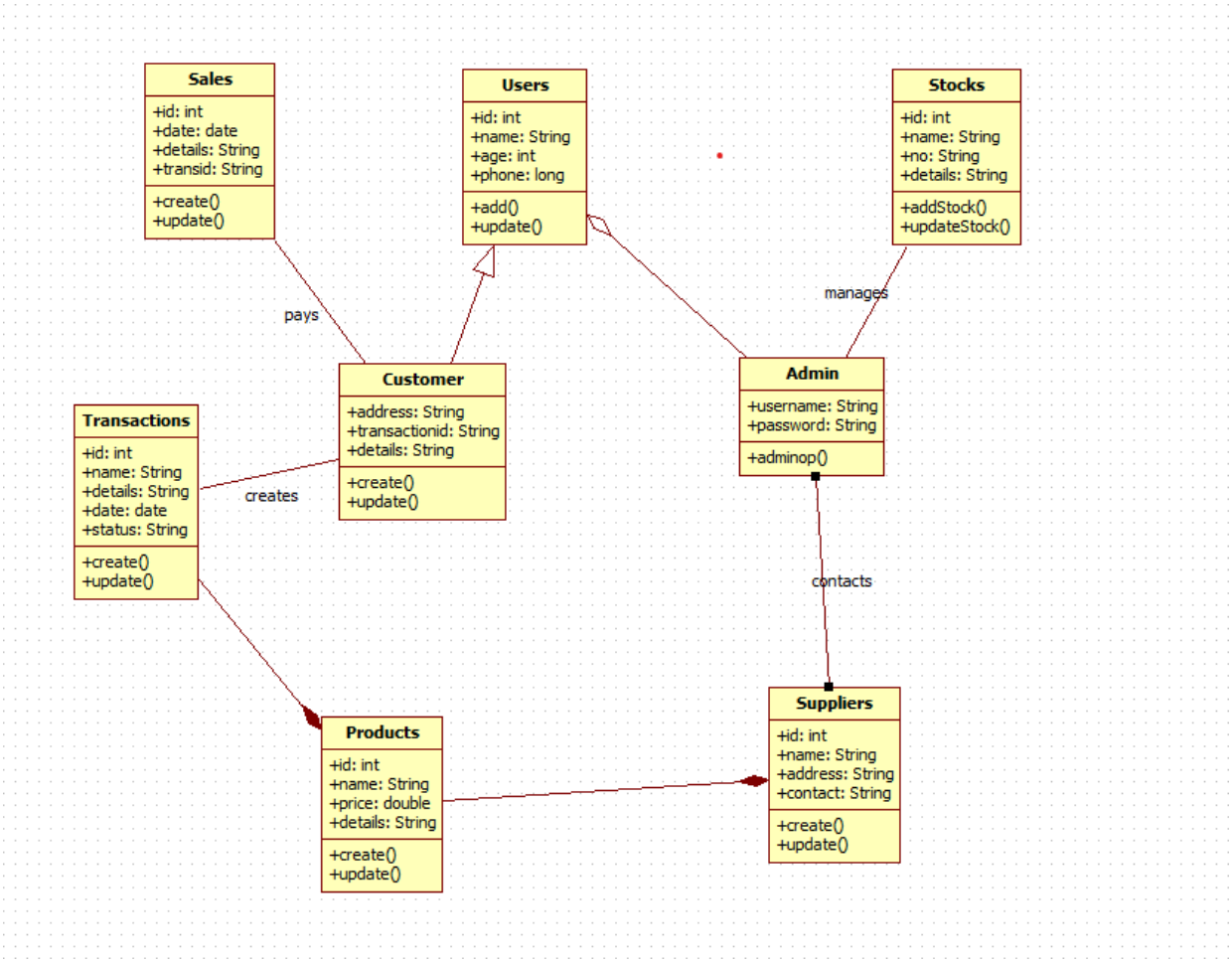
7.4 Portability:
Support all platforms in desktop and mobile

7.5 Data Integrity:
Guarantee accuracy of stock levels of sustainability

8 Preliminary Schedule and Budget

The development is estimated to take 6 months with a budget of 120,000 including design development and testing

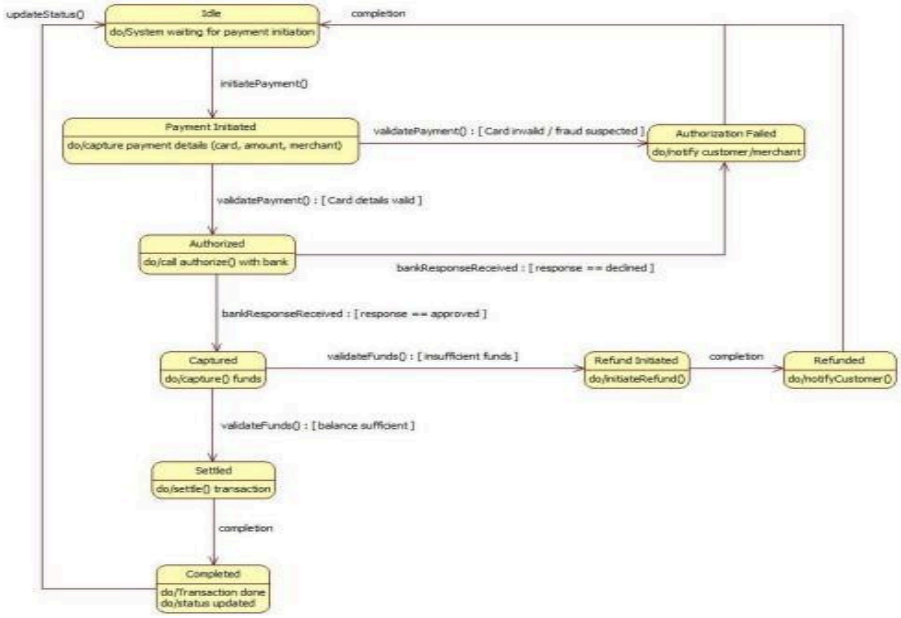
Class Diagram



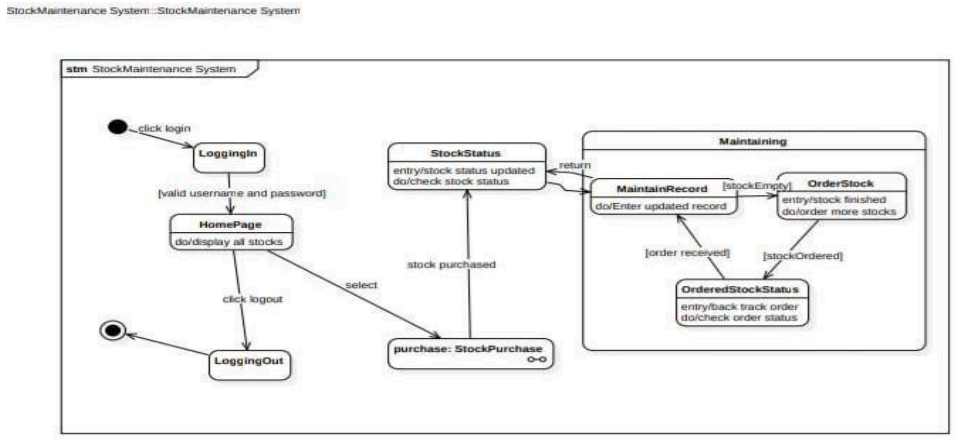
Explanation:

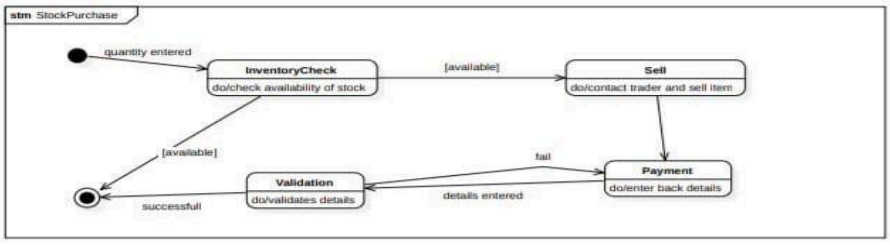
This class diagram visually represents the key entities and their relationships in a Sales Management System, showing how different objects such as Users, Customers, Admins, Suppliers, Stocks, Products, Transactions, and Sales interact within the system. Each class is defined with specific attributes 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.

State Diagram



Advanced State Diagram





Simple State Diagram:

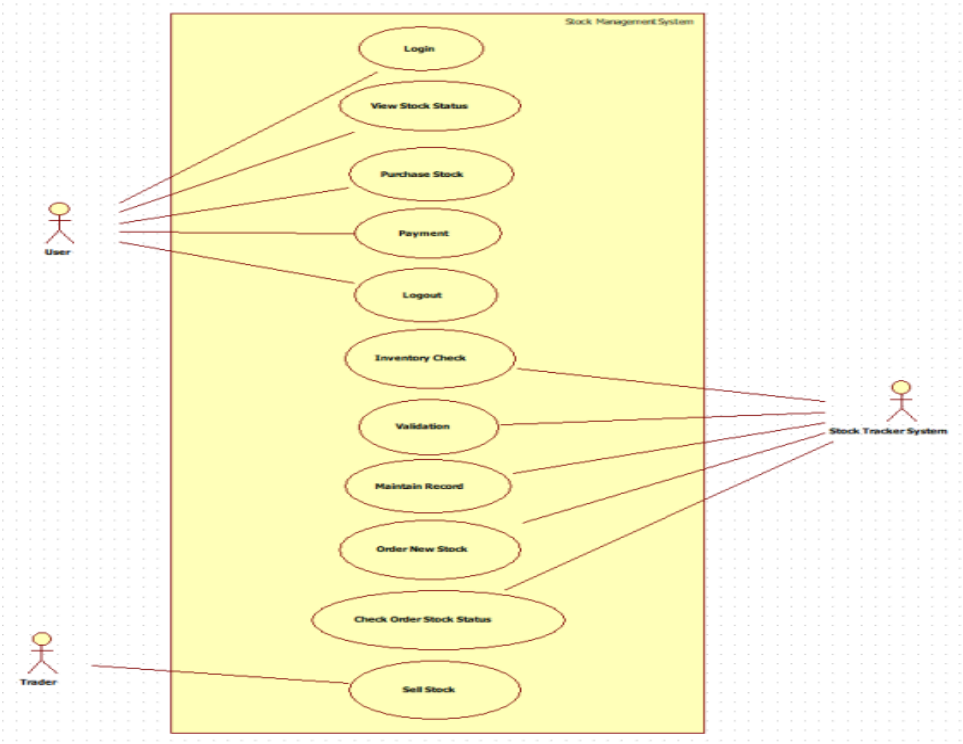
This state diagram illustrates the various stages an item goes through in an inventory or order management system, from being available to the possibility of being out of stock. The process begins when an item is requested, transitioning the item from "Available" to "Reserved" if the quantity is sufficient, or returning to "Available" if a reservation is canceled. After payment, the item is "Dispatched" and moves to "Delivered" once confirmed by the customer. If returned within the allowed period, it enters the "Returned" state for inspection and potential restocking. At any point, if stock runs out, the item moves to the "Out of Stock" state, where reordering and restocking actions are managed.

Advanced State Diagram:

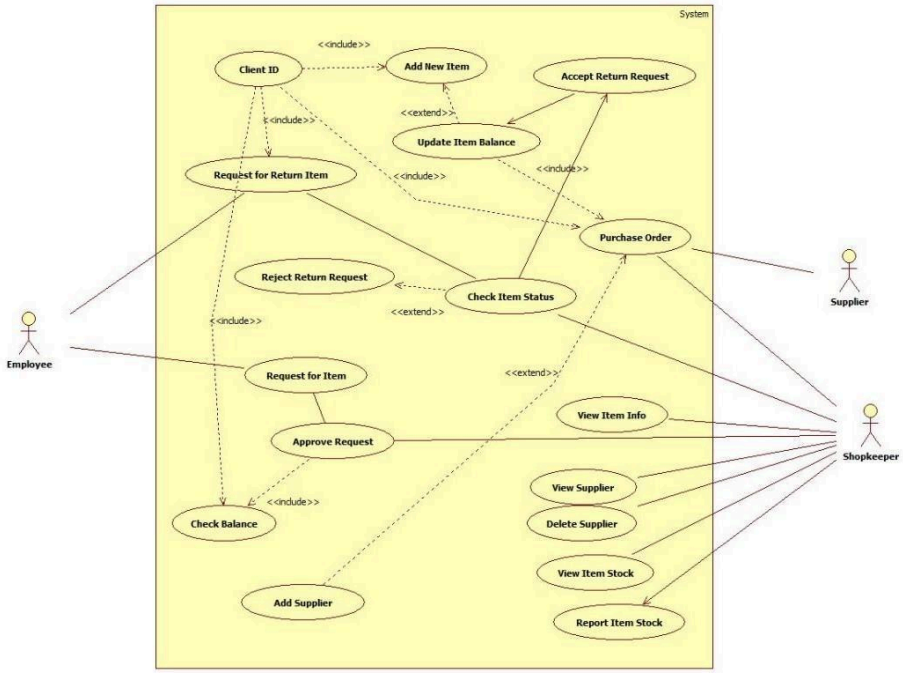
This advanced state diagram models the Stock Maintenance System, focusing on two core processes: managing overall stock status and handling stock purchase transactions. The top section illustrates user actions like logging in, navigating the homepage to view stocks, initiating stock purchases, and logging 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.

Simple Use Case:



Advanced Use Case:



Scenario 1: Successful Purchase and Stock Update (Happy Path)

- 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 → 80 units Failed
- 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

Simple Use case:

This use case diagram illustrates the interactions between two primary actors—Store Manager and Supplier—in a stock management system. The Store Manager handles inventory operations such as adding, updating, and viewing stock levels, as well as generating reports for analysis. The Supplier is responsible for requesting supply and delivering stock to the store. Each actor is linked to specific system functions, clarifying their roles and responsibilities. This visual representation helps in understanding system requirements, streamlining workflows, and guiding software development. It ensures that all user interactions are captured, making the system more efficient, user-centric, and aligned with business objectives.

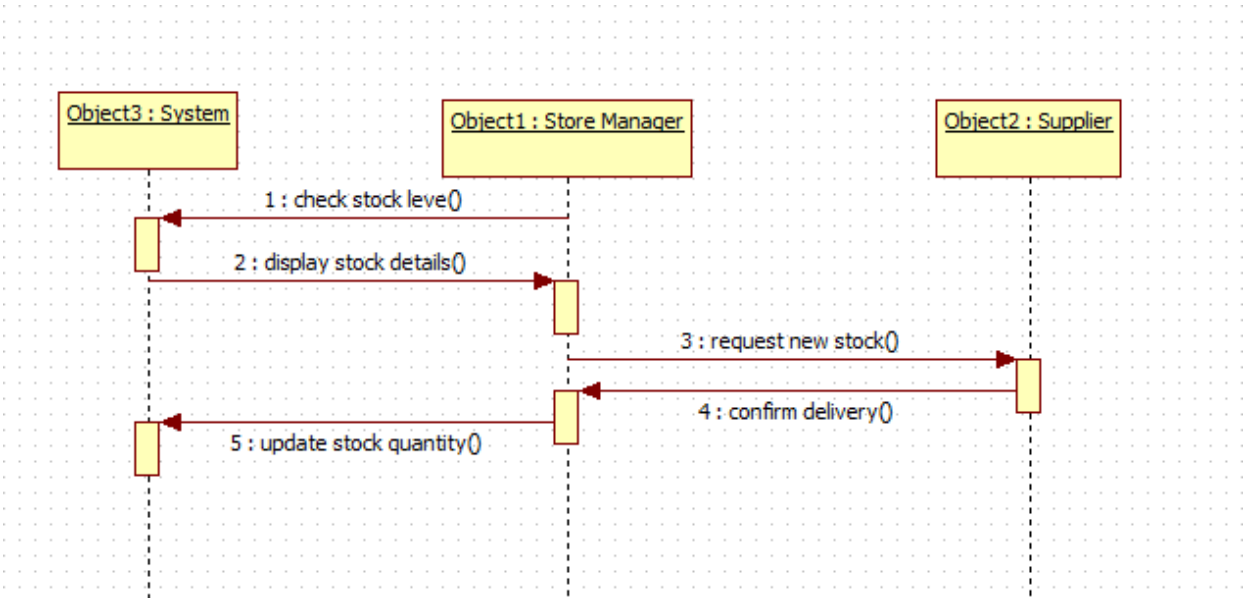
Advanced Use Case:

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 <> and <> 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.

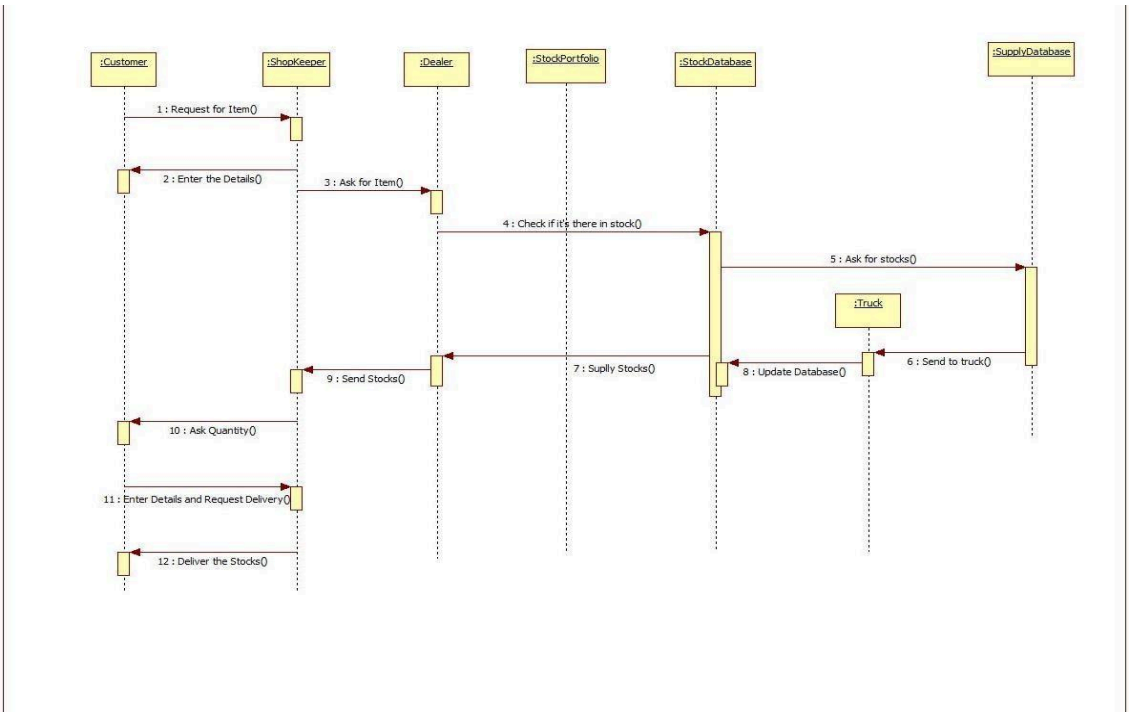
- The diagram maps three actors: Manager, Store Staff, and Supplier, each with distinct responsibilities. The Manager handles buying stock, making payments, and supervising staff.
- Store Staff are responsible for inventory reporting, product quality checks, and selling stock.
- The Supplier delivers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

- For example, "Buy Stock" includes "Give Payment" and may extend to "Report quality issue to supplier."

Simple Sequence Diagram:



Advanced Sequence Diagram:



Simple Sequence:

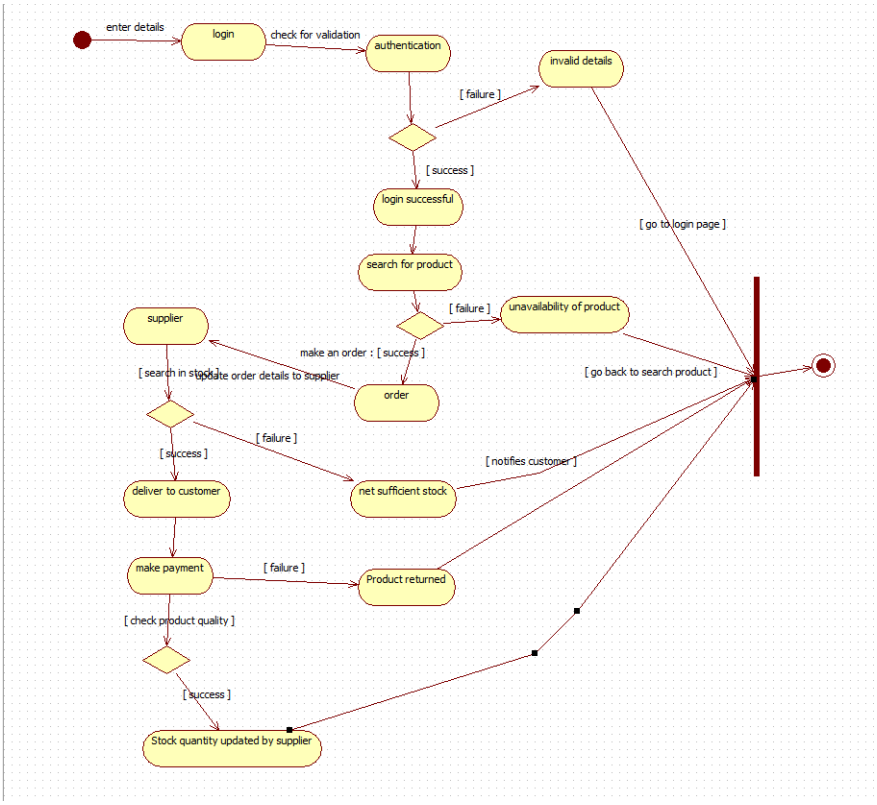
This UML sequence diagram illustrates the interaction between the Store Manager, Supplier, and System in a stock management workflow. The Store Manager initiates the process by checking stock levels through the System, which responds by displaying stock details. If stock is insufficient, the Manager requests new stock from the Supplier, who confirms delivery. Following this, the Manager updates the stock quantity in the System to reflect the new inventory. This sequence clearly outlines the communication flow and operational dependencies among the actors and system components, helping developers understand system behavior and ensuring accurate implementation of inventory-related functionalities.

Advanced Sequence Diagram:

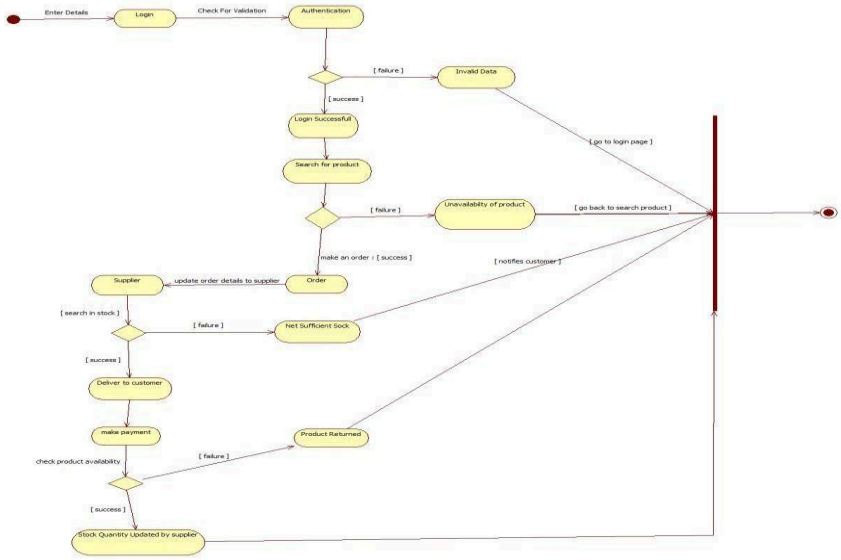
This UML sequence diagram for the Old Stock Management System outlines the step-by-step interaction between entities like Customer, Login, Authentication, Search Product, Order, Supplier, and Stock. The process begins with customer detail entry and validation. Upon successful login, the customer searches 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.

Simple Activity Diagram:



Advanced Activity Diagram:



Simple Activity:

This flowchart illustrates the complete journey of a customer in an online shopping system, starting from entering login details to receiving the product. The process begins with authentication, where invalid credentials redirect the user back to the login page. Upon successful login, the customer searches for a product. If the product is unavailable, they are prompted to search again. Once a product is selected, the system checks stock availability. If stock is insufficient or the product is returned, the customer is notified. This structured flow ensures that only valid users proceed and that product availability is verified before order placement.

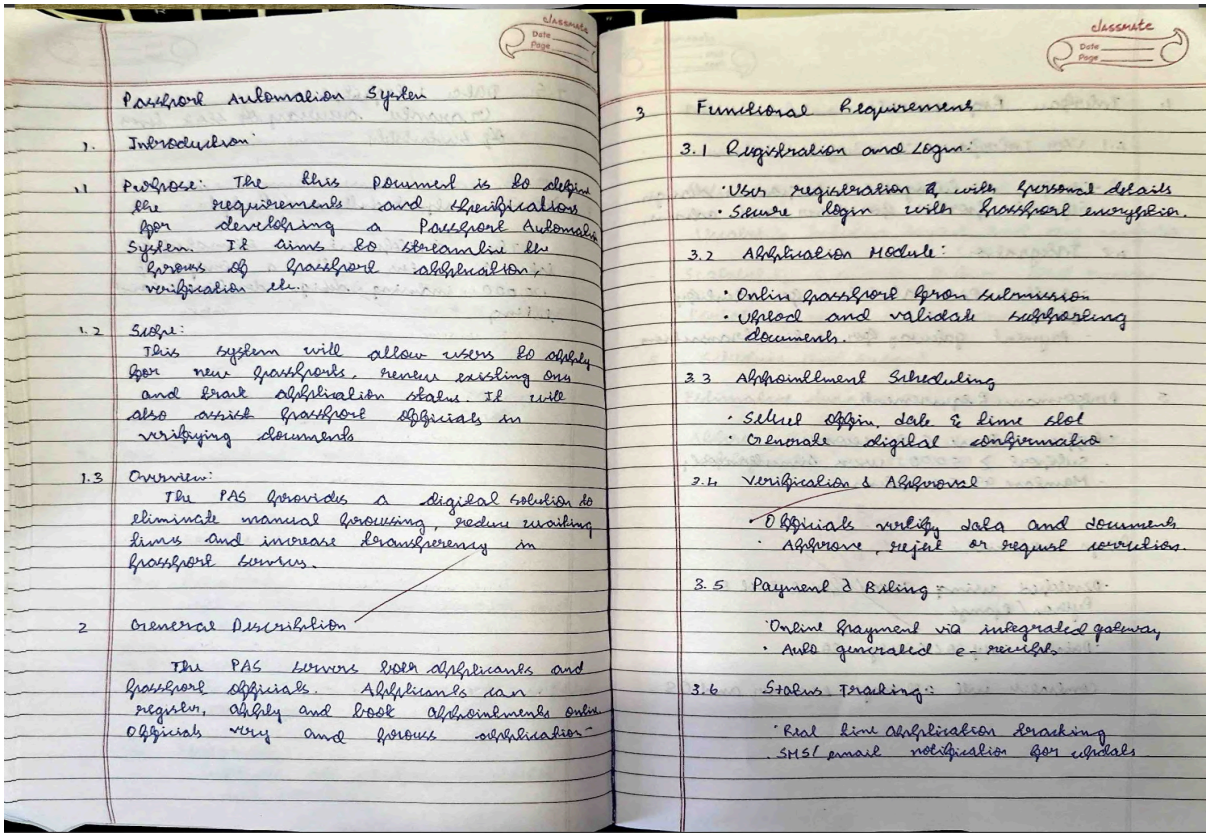
Advanced Activity:

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: The this document is to define the requirements and specifications for developing a Passport Automation System. It aims to streamline the process of passport application, verification etc.

1.2 Scope:
This system will allow users to apply for new passports, renew existing ones and track application status. It will also assist passport officials in verifying documents

1.3 Overview:
The PAS provides a digital solution to eliminate manual processing, reduce waiting time and increase transparency in passport services.

2 General Description

The PAS serves both applicants and passport officials. Applicants can register, apply and book appointments online officials verify and process applications

3 Functional Requirements

3.1 Registration and Login:

- User registeration & with personal details
- Secure login with password encryption.

3.2 Application Module:

- Online passport form submission
- Upload and validate supporting documents.

3.3 Appointment Scheduling

- Select office, date & time slot
- Generate digital confirmation

3.4 Verification & Approval

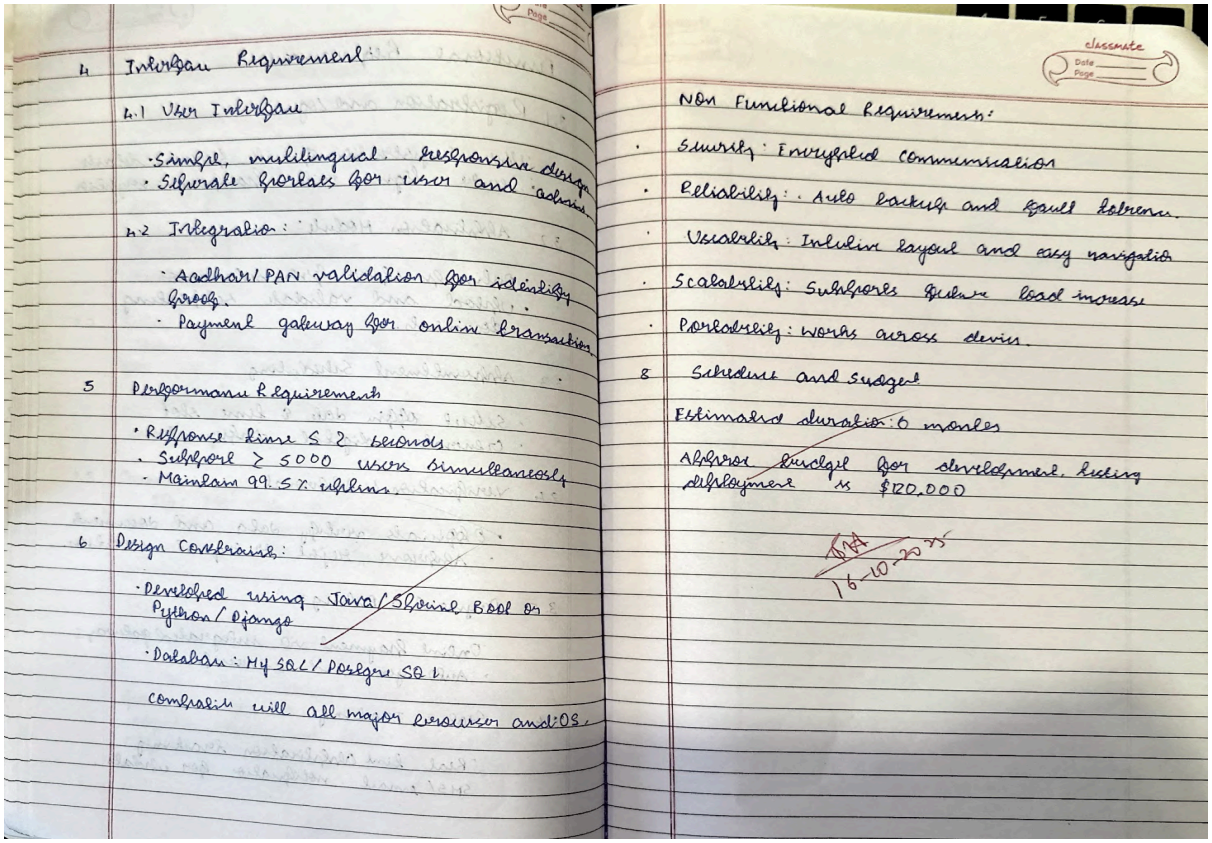
- Officials verify data and documents
- Approve, reject or request corrections.

3.5 Payment & Billing:

- Online payment via integrated gateway
- Auto generated e-receipt.

3.6 Status Tracking:

- Real time application tracking
- SMS/email notification for updates



4 Interface Requirement

4.1 User Interface

- Simple, multilingual responsive design
- Separate portals for user and admin

4.2 Integration:

- Aadhar/PAN validation for identity proof.
- Payment gateway for online transaction

5 Performance Requirement

- Response time ≤ 2 seconds
- Support ≥ 5000 users simultaneously
- Maintain 99.5% uptime.

6 Design Constraints:

- Developed using Java/Spring Boot or Python/Django
- Database: MySQL/PostgreSQL

compatible will all major browsers and OS.

Non Functional Requirements:

- Security: Encrypted communication
- Reliability: Auto backup and fault tolerance
- Usability: Intuitive layout and easy navigation
- Scalability: Supports future load increase
- Portability: Works across devices

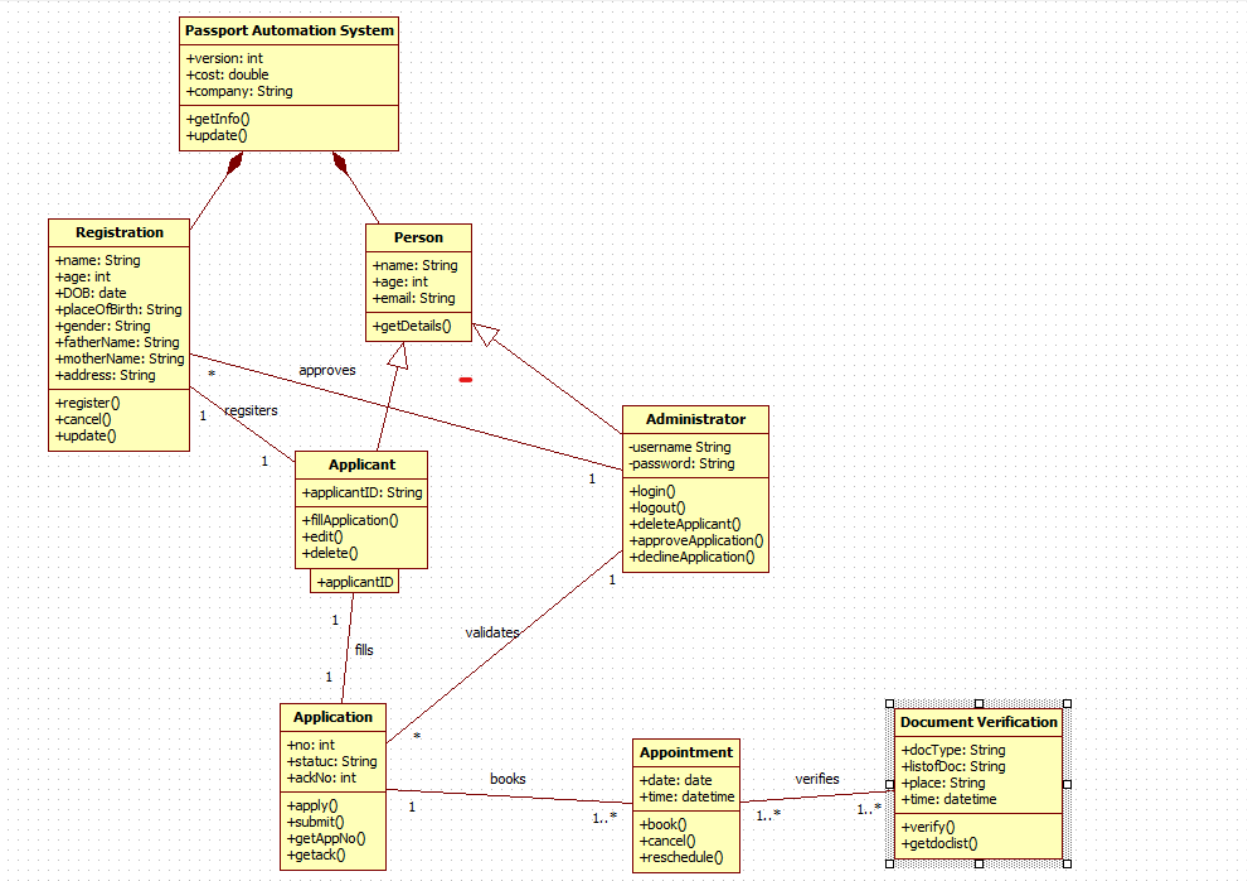
8 Schedule and Budget

Estimated duration: 6 months

Approx budget for development, testing deployment is $120,000

16-10-25

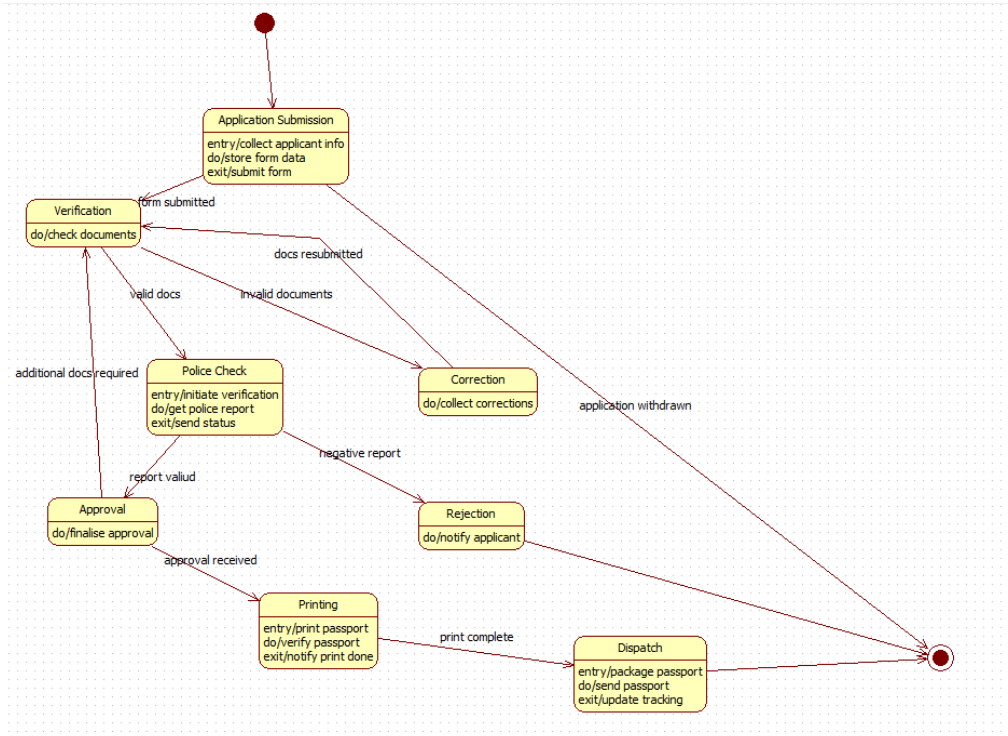
Class Diagram



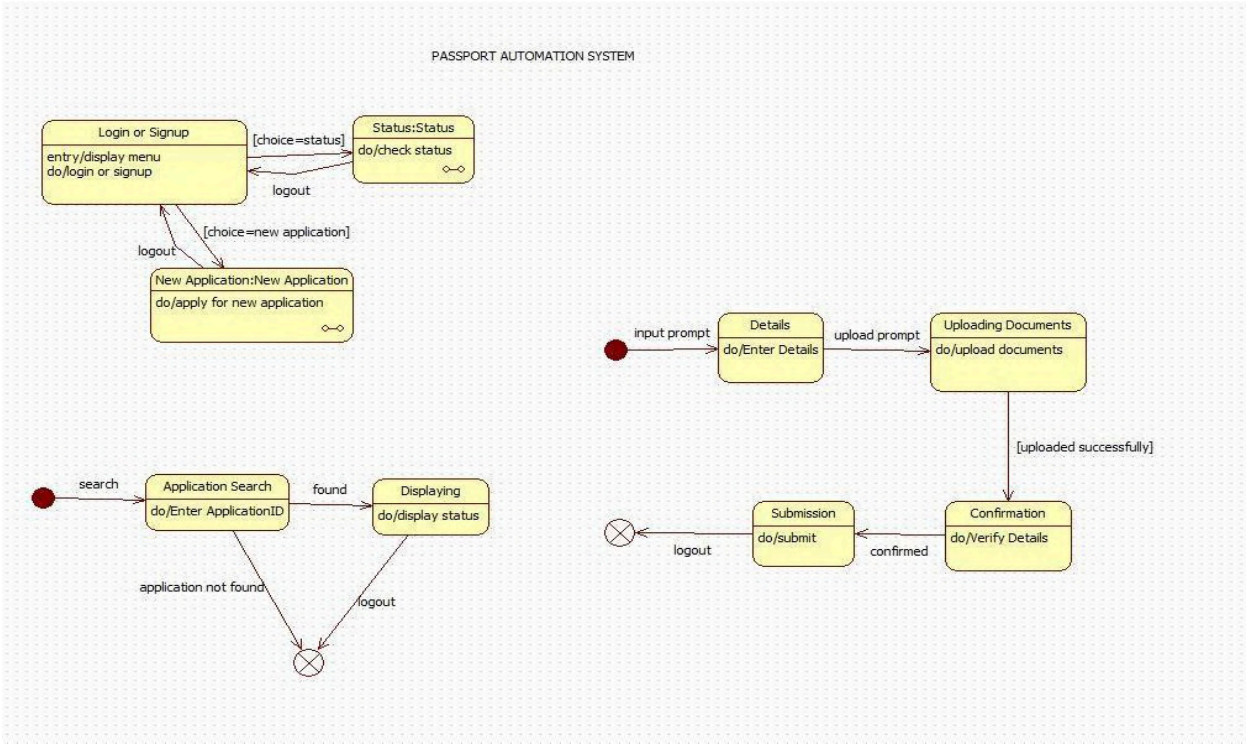
Explanation:

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.

Simple State Diagram:



Advanced State Diagram:



Simple State:

This activity diagram depicts the complete life-cycle of a passport application. It begins with the applicant submitting the online/offline form and documents. The application then undergoes Verification; if documents are invalid, corrections are sought or the application is withdrawn. Valid documents trigger Police Check; a negative report leads to Rejection, while a clear report moves to final Approval. Once approved, the passport proceeds to Printing and then Dispatch, completing the process. The diagram clearly shows the main success path (submission → verification → police check → approval → printing → dispatch) along with alternate paths for invalid documents, corrections, negative police reports, and rejection.

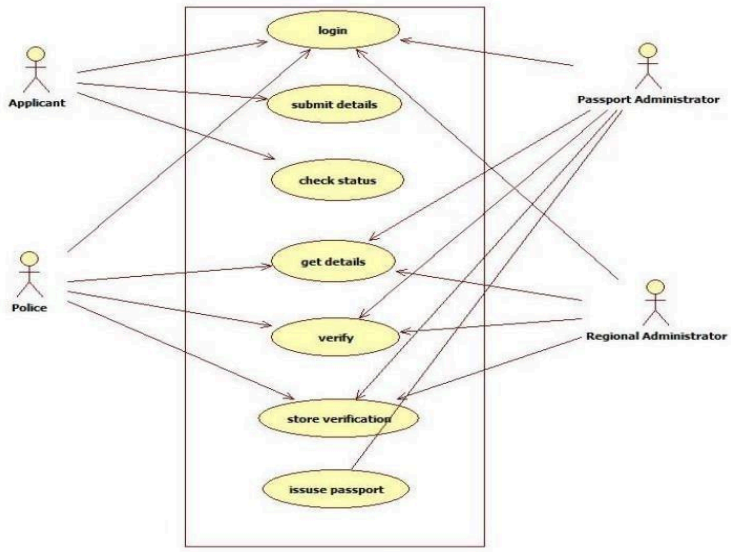
Advanced State:

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.

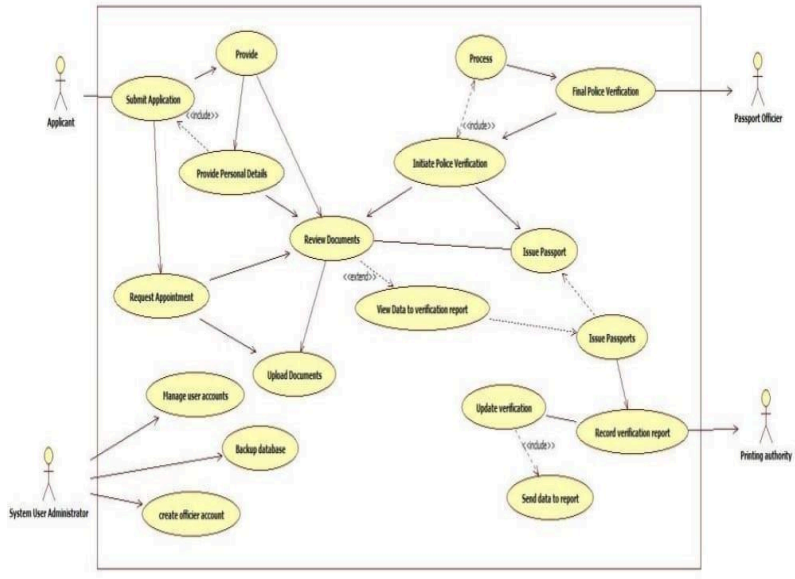
Selecting “New Application” initiates a dedicated subprocess where users first input personal details, then proceed to upload required documents. Successful upload triggers the “Submission” activity, followed by system-side confirmation and verification of the entered data. Only after verification is 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.

Simple Use Case:



Advanced Use Case:



Scenario 1: Main Success Scenario (Happy Path)

- 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 due to Adverse Police Report)

- 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

Simple Use Case:

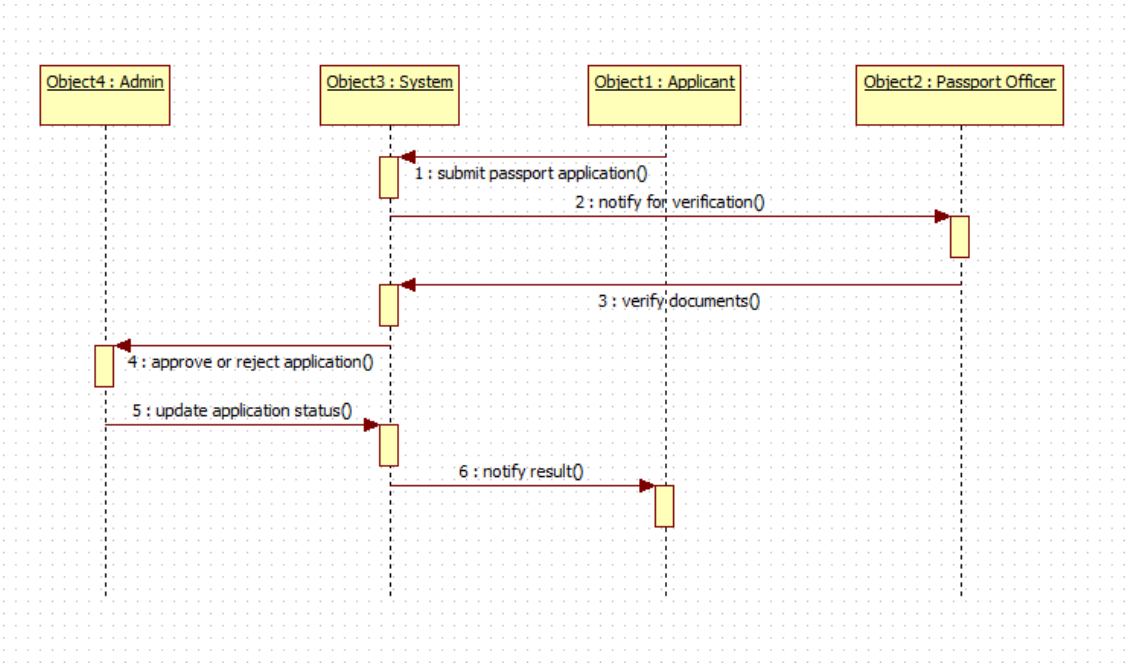
This use case diagram represents a Passport Automation System involving three actors: Applicant, Passport Officer, and Admin. The Applicant can Apply for Passport, Schedule Appointment for document verification, and Check Status of their application anytime. The Passport Officer is responsible for Verify Documents during the appointment and subsequently Issue or Reject the Passport based on authenticity and police verification reports. The Admin handles administrative tasks by managing user accounts (Manage Users) and updating application statuses (Update Application Status) throughout the process. The diagram clearly separates citizen-facing, officer- level, and administrative functions, ensuring smooth, role-based interaction with the system.

Advanced Use Case:

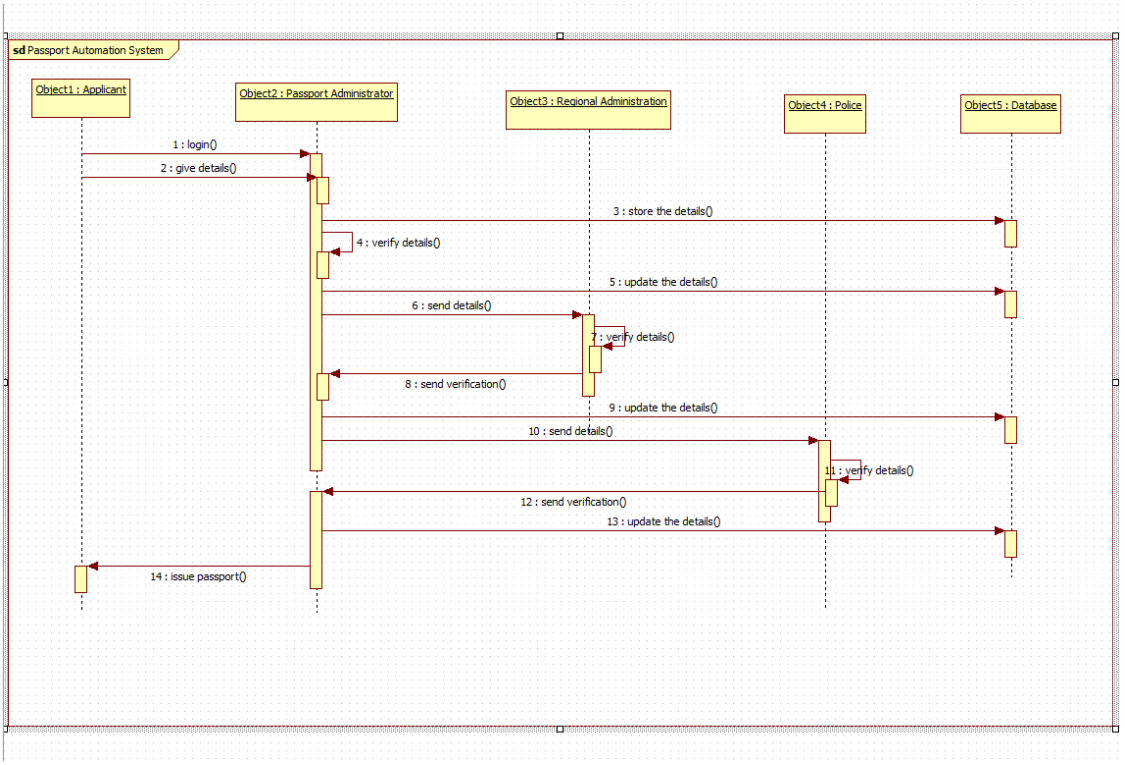
The use case diagram outlines a comprehensive Passport Automation System with four actors. The Applicant registers on the portal, logs in, fills the application form, pays fees via an external Payment Gateway, tracks status, and schedules an appointment. The "Schedule Appointment" use case includes receiving 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.

Simple Sequence Diagram:



Advanced Sequence Diagram:



Simple Sequence:

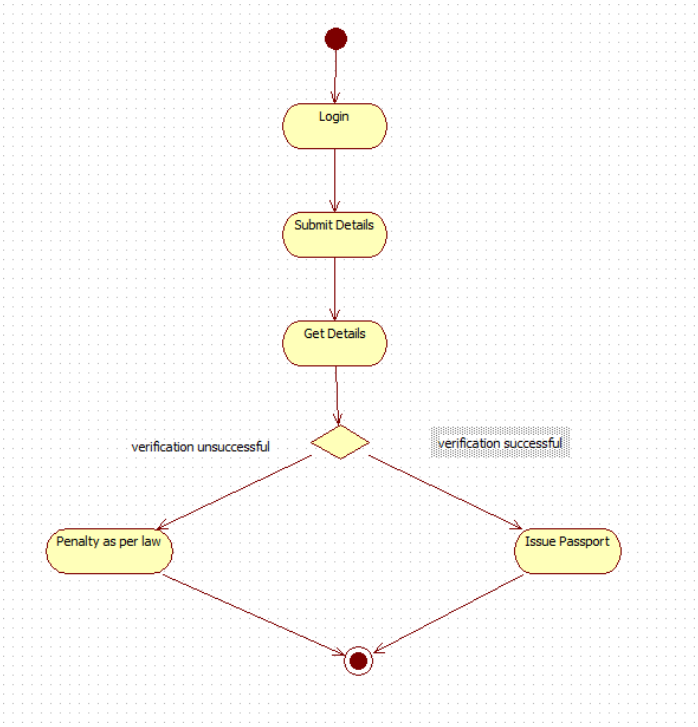
This sequence diagram illustrates the interaction flow in a Passport Automation System. The Applicant (Object1) initiates by submitting the passport application to the System (Object3). The System immediately notifies the Passport Officer (Object2) for verification. The Officer verifies the documents 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 Sequence:

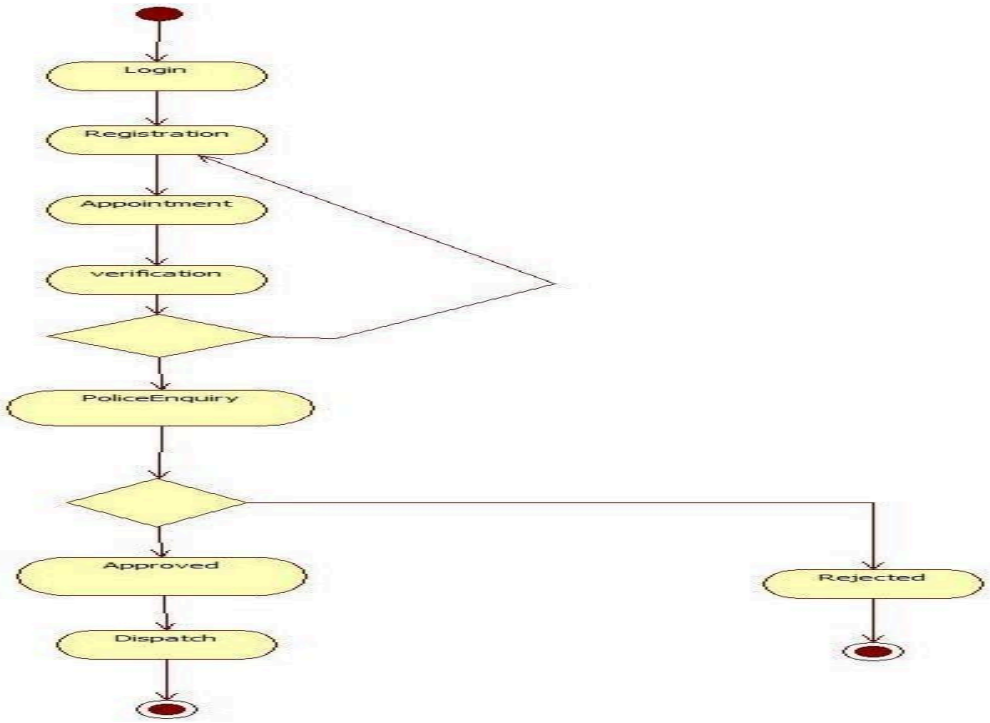
The sequence diagram depicts the complete passport application verification process. The Applicant first logs in and submits personal details to the Passport Administrator. The Administrator verifies the details, stores them in the Database, and forwards them to the Regional Administration. The Regional Administration updates the Database and sends the details to the Police for background verification. After the Police verify and return the report, the Regional Administration updates the Database again and notifies the Passport Administrator. This multi- level verification involving local, regional, and police authorities ensures thorough scrutiny of applicant information before final processing.

Once police verification is complete, the Passport Administrator receives confirmation, updates the Database, and the Regional Administration issues the final approval. The Database is updated with the final status, and the Passport Administrator issues the passport to the Applicant. Throughout the flow, the Database serves as the central repository, updated at every stage (initial submission, regional processing, police verification, and final issuance). This structured, step-by-step interaction among Applicant, Passport Administrator, Regional Administration, Police, and Database ensures transparency, accountability, and secure passport issuance with proper background checks and official validations at each level.

Simple Activity Diagram:



Advanced Activity Diagram:



Simple Activity:

This activity diagram illustrates the passport renewal/issuance process under strict verification. The applicant begins by logging in, submits personal and supporting details, and the system retrieves and verifies the stored information. A decision node checks verification outcome: if successful, the passport is issued immediately; if unsuccessful (due to discrepancies, criminal record, or invalid documents), a penalty is imposed as per law. The simple, linear flow with a single critical decision point emphasizes automated background checks and legal compliance, ensuring only eligible applicants receive passports while enforcing penalties for fraudulent or ineligible cases.

Advanced Activity:

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.