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:

Deepthi M
1BM23CS088

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 Deepthi M(1BM23CS088) during the 5th Semester August 2025-December 2025

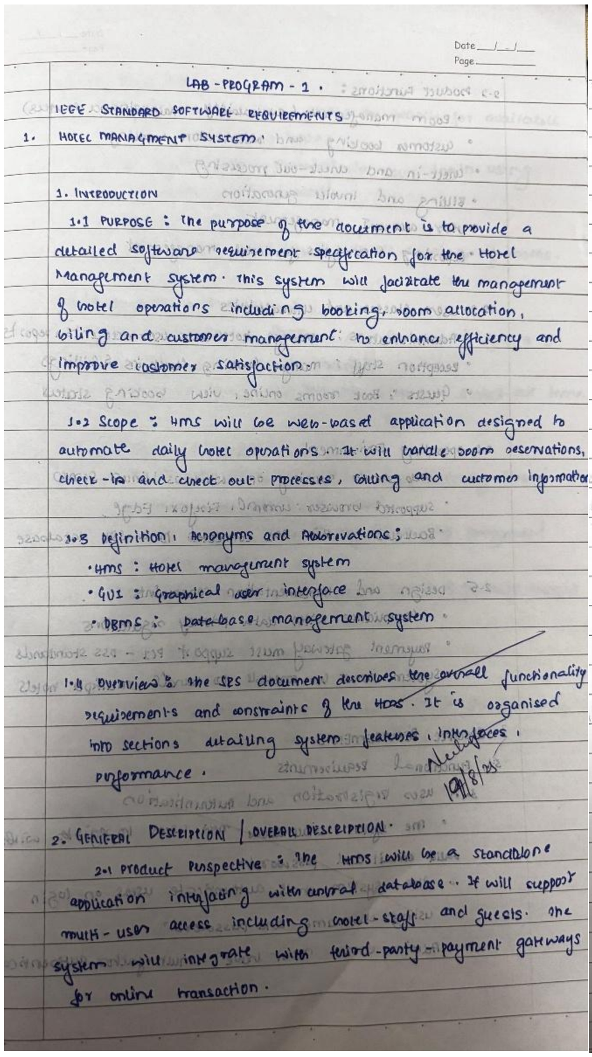
Signature of the Faculty Incharge:

Sonika Sharma D
Assistant Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

Table of Contents

1. Hotel Management System

SRS Document



Date __/__/__
Page ______

LAB - PROGRAM - 1 .

IEEE STANDARD SOFTWARE REQUIREMENTS

1. HOTEL MANAGMENT SYSTEM :

1. INTRODUCTION

1.1 PURPOSE : The purpose of the document is to provide a detailed software requirement specification for the Hotel Management system. This system will facilitate the management of hotel operations including booking, room allocation, billing and customer management to enhance efficiency and improve customer satisfaction.

1.2 Scope : HMS will be web-based application designed to automate daily hotel operations. It will handle room reservations, check-in and check out processes, billing and customer information

1.3 Definition, Acronyms and Abbrevations ;

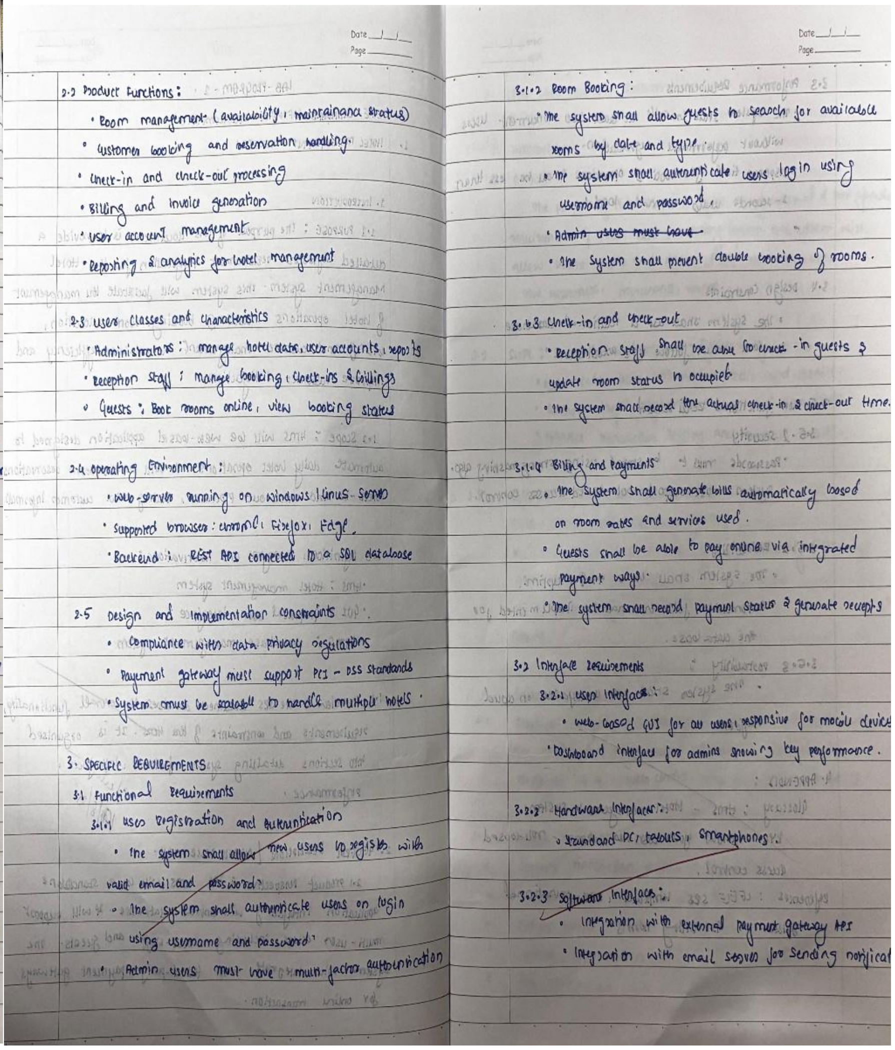
- HMS : Hotel management system
- GUI : Graphical user interface
- DBMS : Database management system

1.4 Overview : The SRS document describes the overall functionality requirements and constraints of the HMS. It is organised into sections detailing system features, interfaces, performance.

[illegible] 19/8/25

2. GENERAL DESCRIPTION / OVERALL DESCRIPTION.

2.1 Product Perspective : The HMS will be a standalone application interfacing with central database. It will support multi-user access including hotel-staff and guests. The system will integrate with third-party payment gateways for online transaction.



Date __/__/__
Page ______

2.2 Product Functions :

- Room management (availability, maintainance status)
- Customer booking and reservation handling.
- Check-in and check-out processing
- Billing and invoice generation
- User account management
- Reporting & analytics for hotel management

2.3 User Classes and characteristics

- Administrators : manage hotel data, user accounts, reports
- Reception staff : manage booking, check-ins & billings
- Guests : Book rooms online, view booking status

2.4 Operating Environment :

- Web-server running on windows / Linux servers
- Supported browser : chrome, Firefox, Edge.
- Backend : REST API connected to a SQL database

2.5 Design and Implementation constraints

- Compliance with data privacy regulations
- Payment gateway must support PCI - DSS standards
- System must be scalable to handle multiple hotels.

3. SPECIFIC REQUIREMENTS

3.1 Functional Requirements

3.1.1 User Registration and Authentication

- The system shall allow new users to register with valid email and password.
- The system shall authenticate users on login using username and password.
- Admin users must have multi-factor authentication

Date __/__/__
Page ______

3.1.2 Room Booking :

- The system shall allow guests to search for available rooms by date and type.
- The system shall authenticate users login using username and password.
- ~~Admin users must have~~
- The System shall prevent double booking of rooms.

3.1.3 Check-in and check-out

- Reception staff shall be able to check-in guests & update room status to occupied
- The system shall record the actual check-in & check-out time.

3.1.4 Billing and Payments

- The system shall generate bills automatically based on room rates and services used.
- Guests shall be able to pay online via integrated payment ways.
- The system shall record payment status & generate receipts

3.2 Interface Requirements

3.2.1 User Interface :

- Web-based GUI for all users, responsive for mobile devic
- Dashboard interface for admins showing key performance.

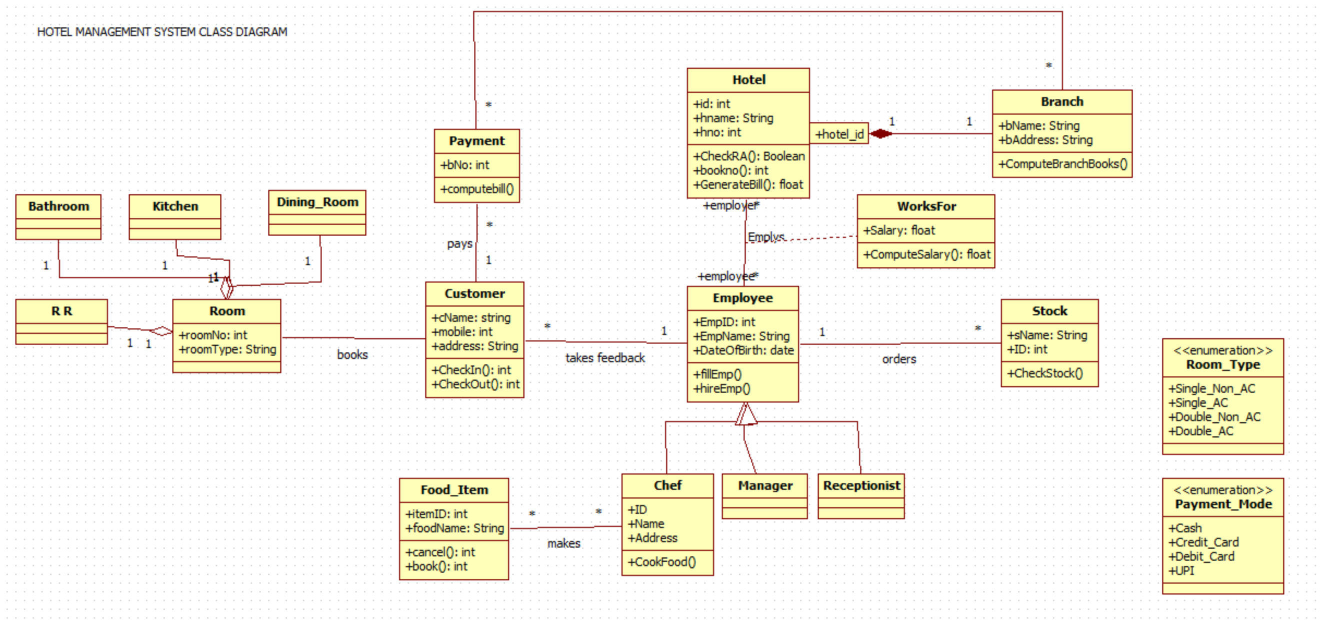
3.2.2 Hardware Interface :

- Standard PC, tablets, smartphones.

3.2.3 Software Interface :

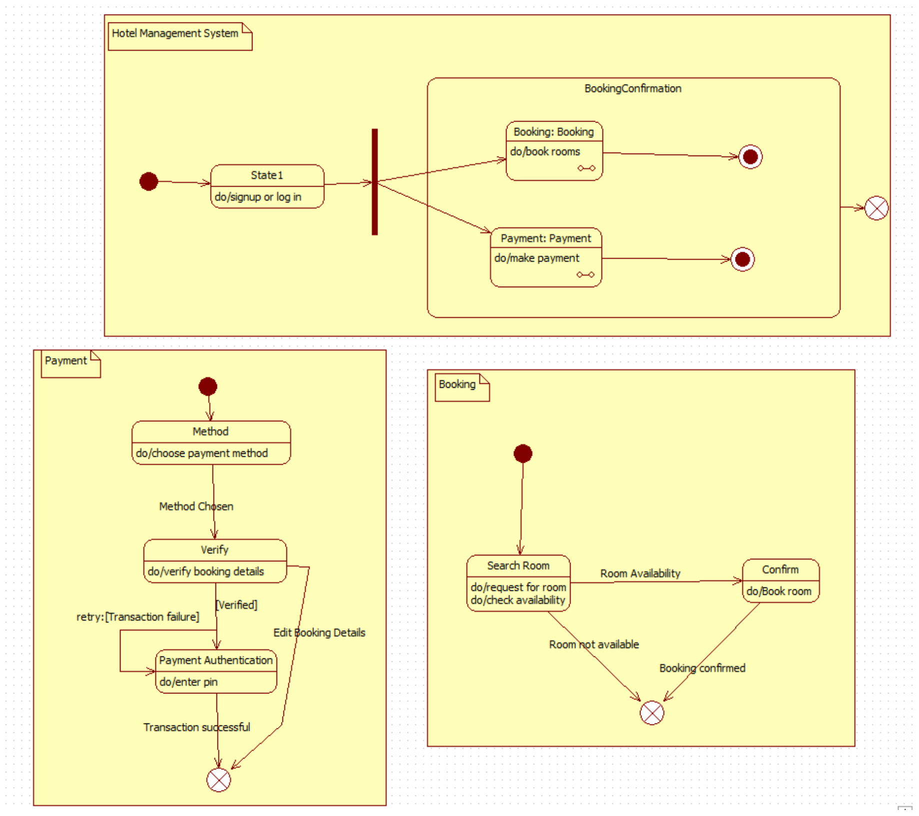
- Integration with external payment gateway APIs
- Integration with email server for sending notifica

Class Diagram:



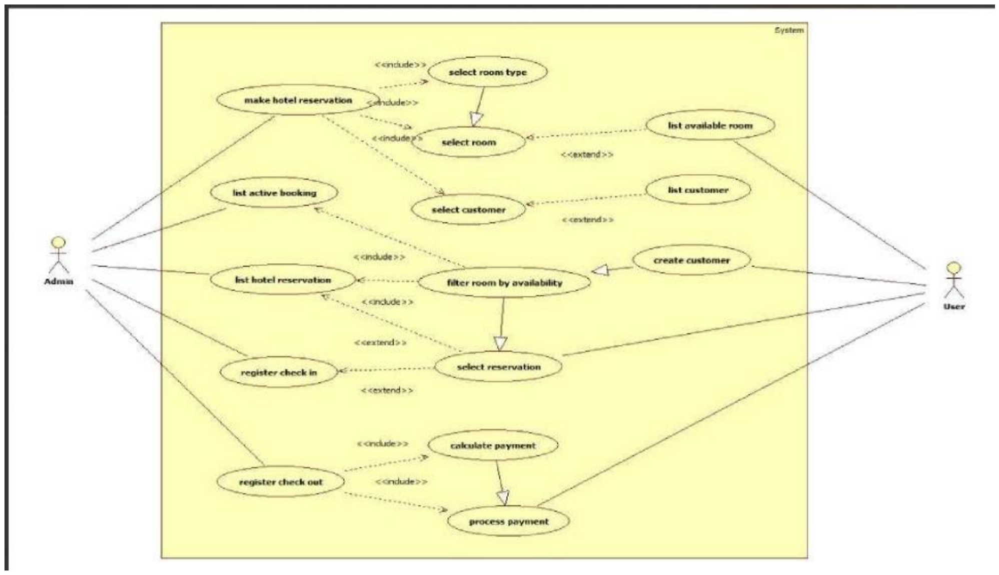
The class diagram for a Hotel Management System outlines the system's architecture by specifying key classes—Hotel, Branch, Customer, Room, Employee, Payment, FoodItem, and Stock—along with their attributes, behaviors, and detailed interconnections. A hotel oversees several branches, each with its own 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.

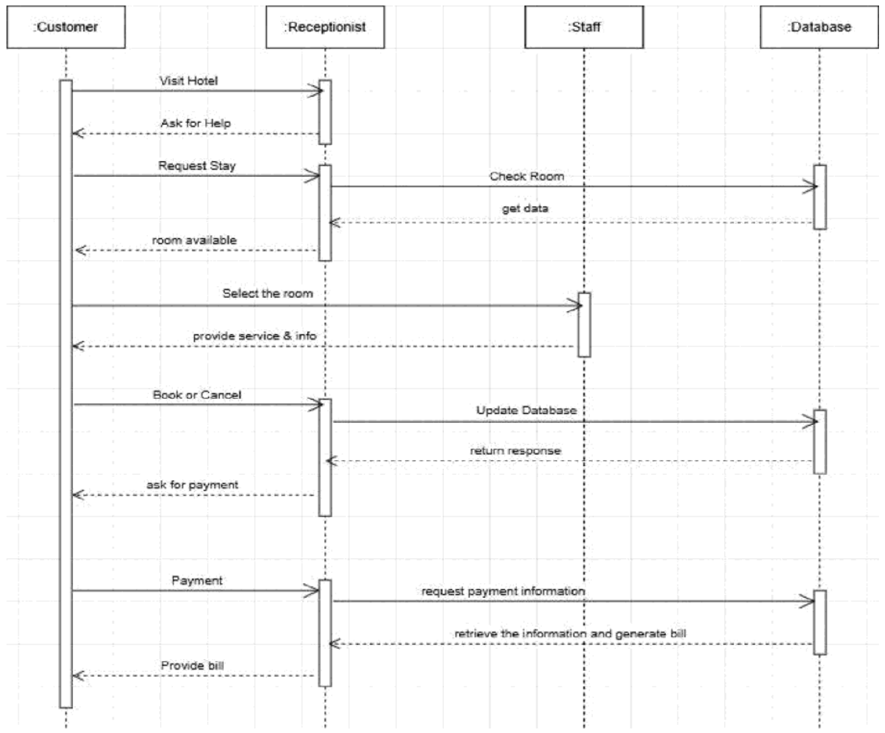
The customer enters required payment details.

The payment gateway verifies the details and processes the transaction.

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

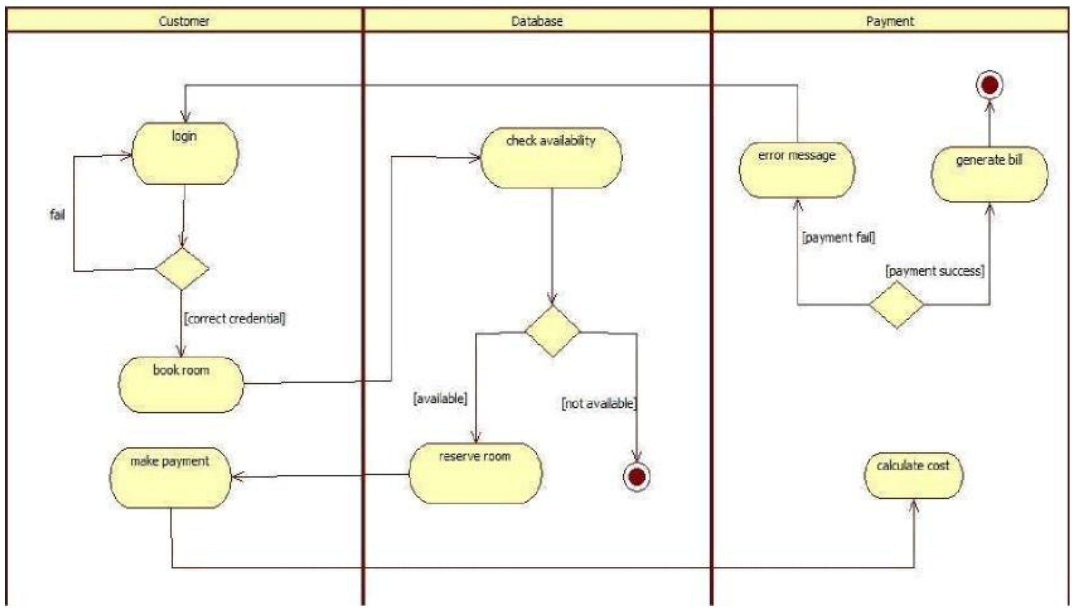
The system marks the booking as “Paid” and generates a payment receipt.

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



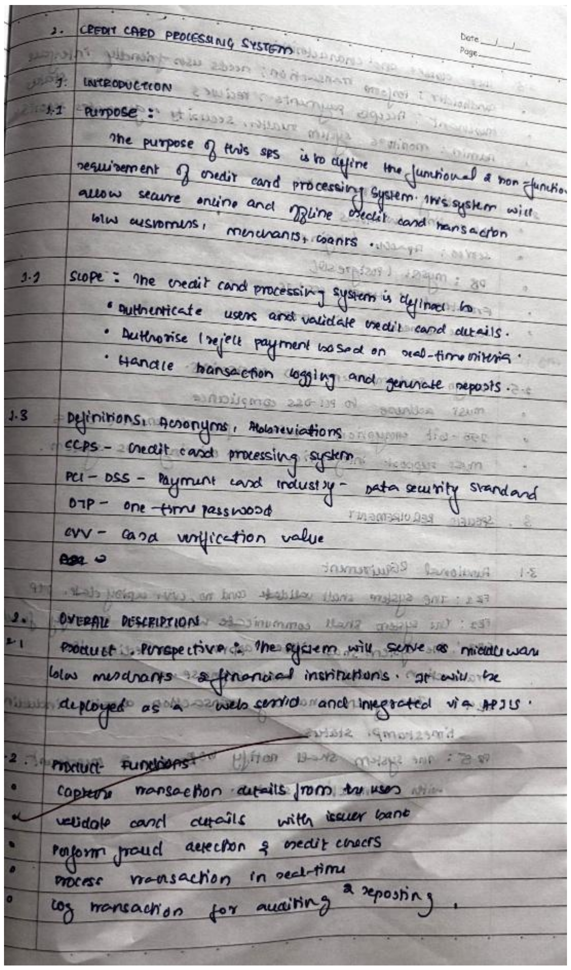
The advanced activity diagram shows the same process in more detail and separates the actions into three swimlanes: Customer, Database, and Payment.

- 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 togethe

2.Credit Card Processing

SRS Document



2. CREDIT CARD PROCESSING SYSTEM

Date __/__/__
Page ____

1. INTRODUCTION

1.1 Purpose :

The purpose of this SRS is to define the functional & non-functio requirement of credit card processing system. This system will allow secure online and offline credit card transaction b/w customers, merchants, banks.

1.2 Scope : The credit card processing system is defined to

- Authenticate users and validate credit card details.
- Authorise / reject payment based on real-time criteria.
- Handle transaction logging and generate reports.

1.3 Definitions, Acronyms, Abbreviations

CCPS - credit card processing system

PCI - DSS - Payment card industry - Data security standard

OTP - one-time password

CVV - card verification value

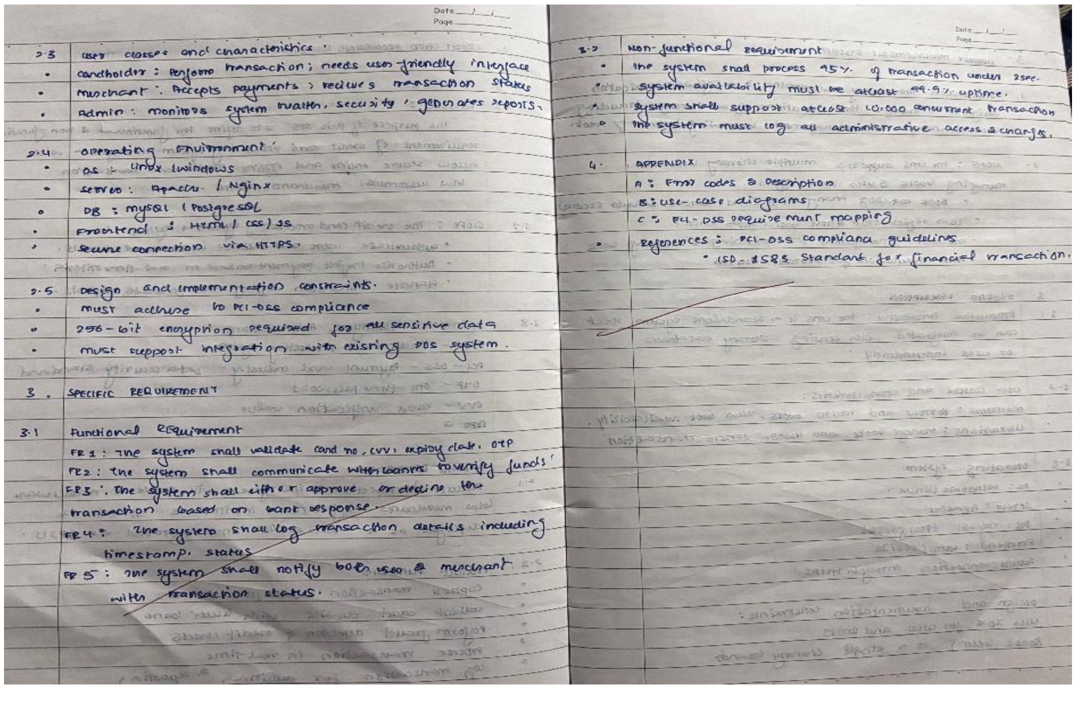
~~[illegible]~~ ~~[illegible]~~

2. OVERALL DESCRIPTION

2.1 Product Perspective : The system will serve as middleware b/w merchants & financial institutions. It will be deployed as a web service and integrated via APIs.

2.2 Product Functions

- Capture transaction details from the users
- validate card details with issuer bank
- Perform fraud detection & credit checks
- process transaction in real-time
- log transaction for auditing & reporting.



Date __/__/__
Page ____

2.3 User classes and characteristics :

- Cardholder : Perform transaction; needs user-friendly interface
- Merchant : Accepts payments, receives transaction status
- Admin : monitors system health, security, generates reports.

2.4 Operating Environment :

- OS : Linux / Windows
- Server : Apache / Nginx
- DB : MySQL / PostgreSQL
- Frontend : HTML / CSS / JS
- Secure connection via HTTPS.

2.5 Design and implementation constraints.

- must adhere to PCI-DSS compliance
- 256-bit encryption required for all sensitive data
- must support integration with existing POS system.

3. SPECIFIC REQUIREMENT

3.1 Functional Requirement

FR 1 : The system shall validate card no, CVV, expiry date, OTP

FR2 : The system shall communicate with banks to verify funds.

FR3 : The system shall either approve or decline the transaction based on bank response.

FR4 : The system shall log transaction details including timestamp, status

FR 5 : The system shall notify both user & merchant with transaction status.

Date __/__/__
Page ____

3.2 Non-functional requirement

- The system shall process 95% of transaction under 2sec.
- System availability must be atleast 99.9% uptime.
- System shall support atleast 10,000 concurrent transaction
- The system must log all administrative access & changes.

4. APPENDIX

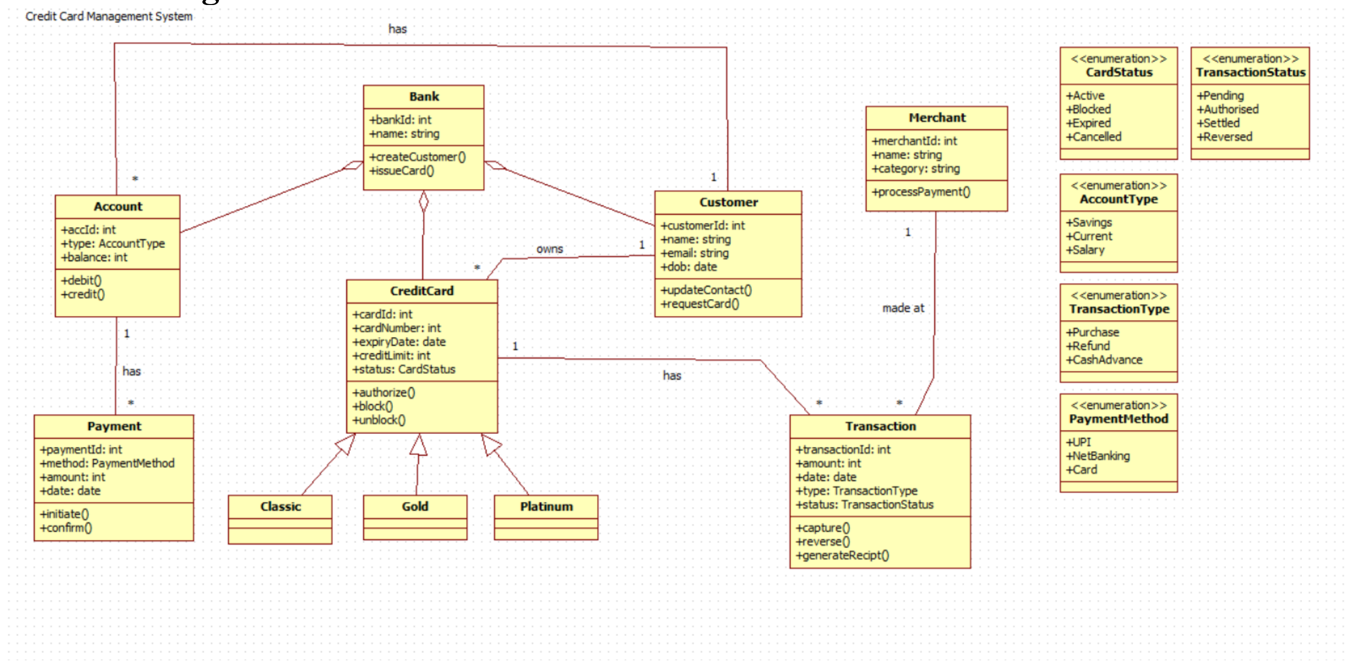
A : Error codes & Description

B : use-case diagrams

C : PCI-DSS requirement mapping

- References : PCI-DSS compliance guidelines
  - ISO 8583 Standard for financial transaction.

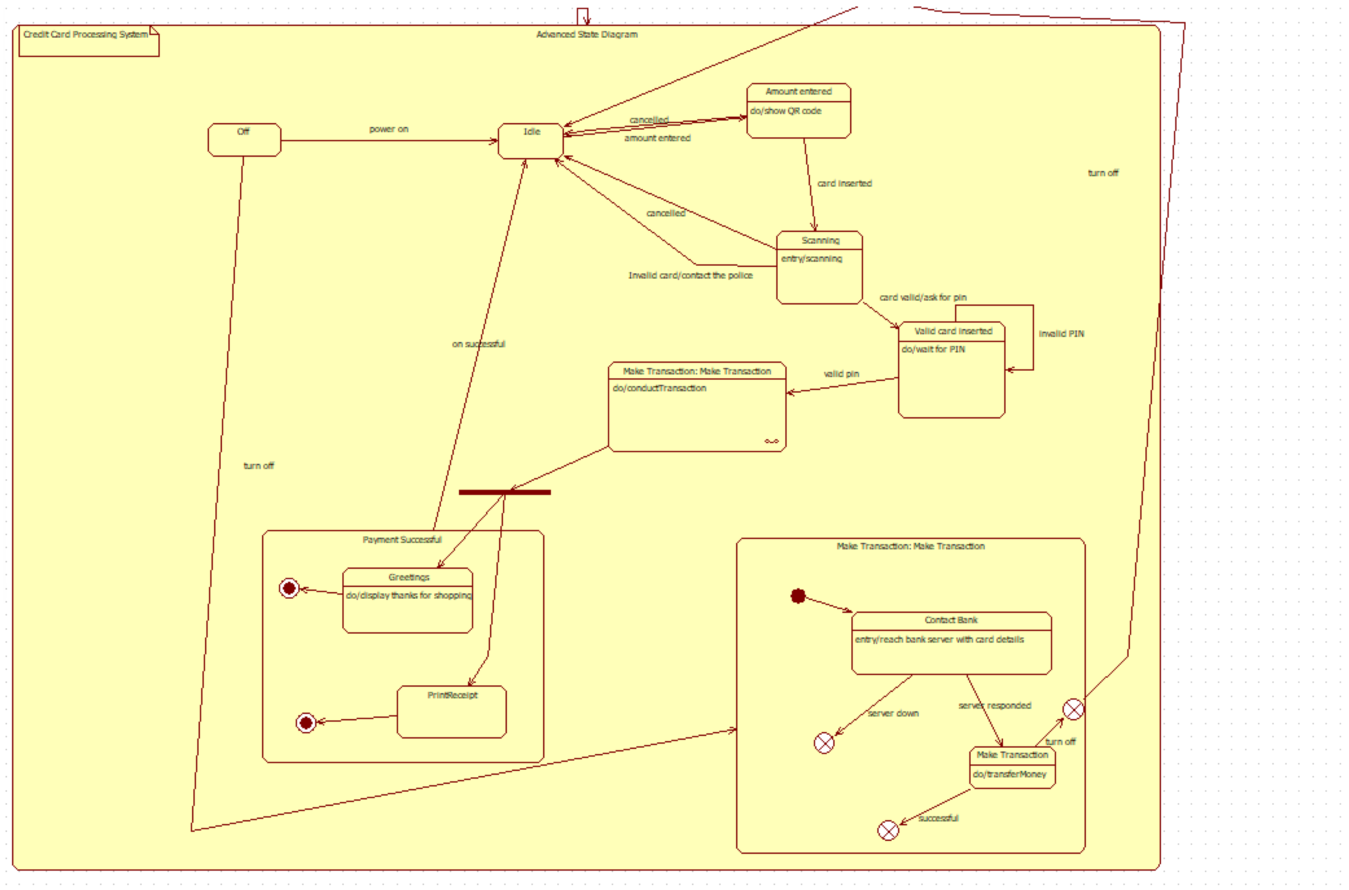
Class Diagram



Explanation

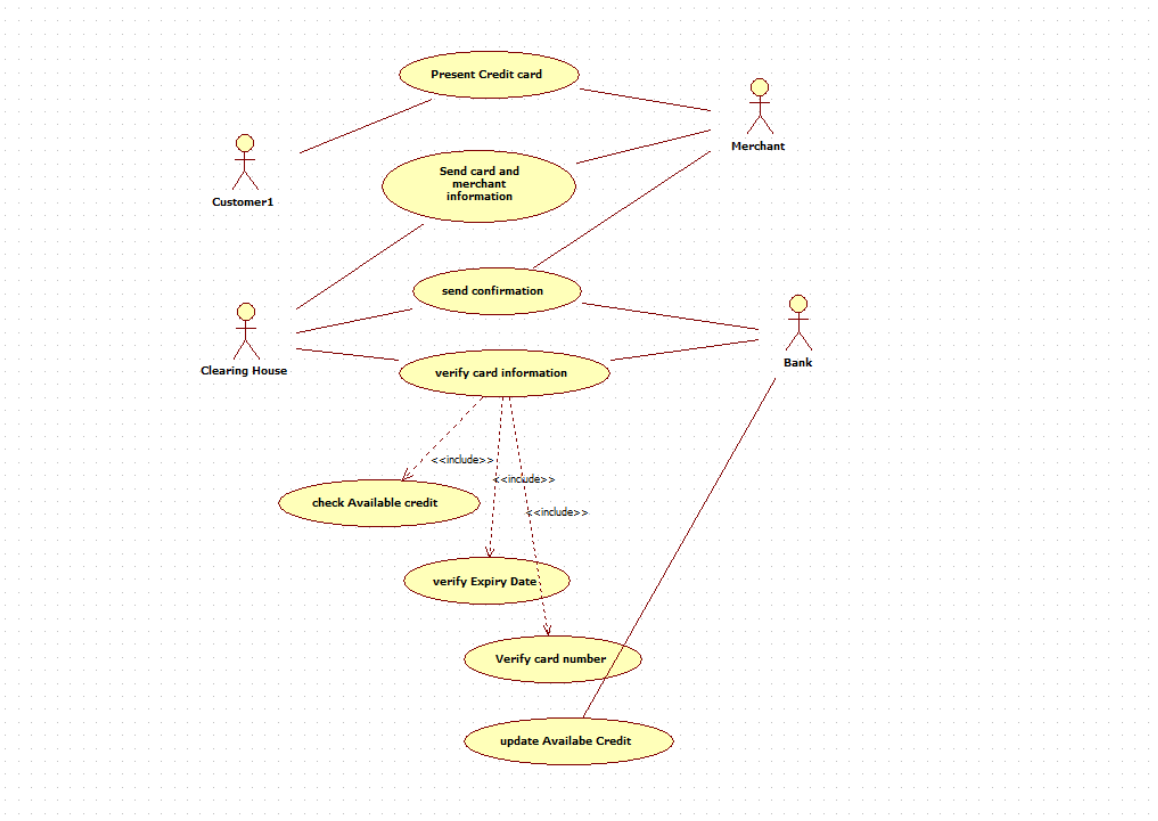
The class diagram represents an online payment processing system that manages customers, merchants, banks, and transactions. The Customer initiates payments using their card, which is linked to a Bank Account. A Transaction is created for each payment, containing details such as amount, date, status, 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 14 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

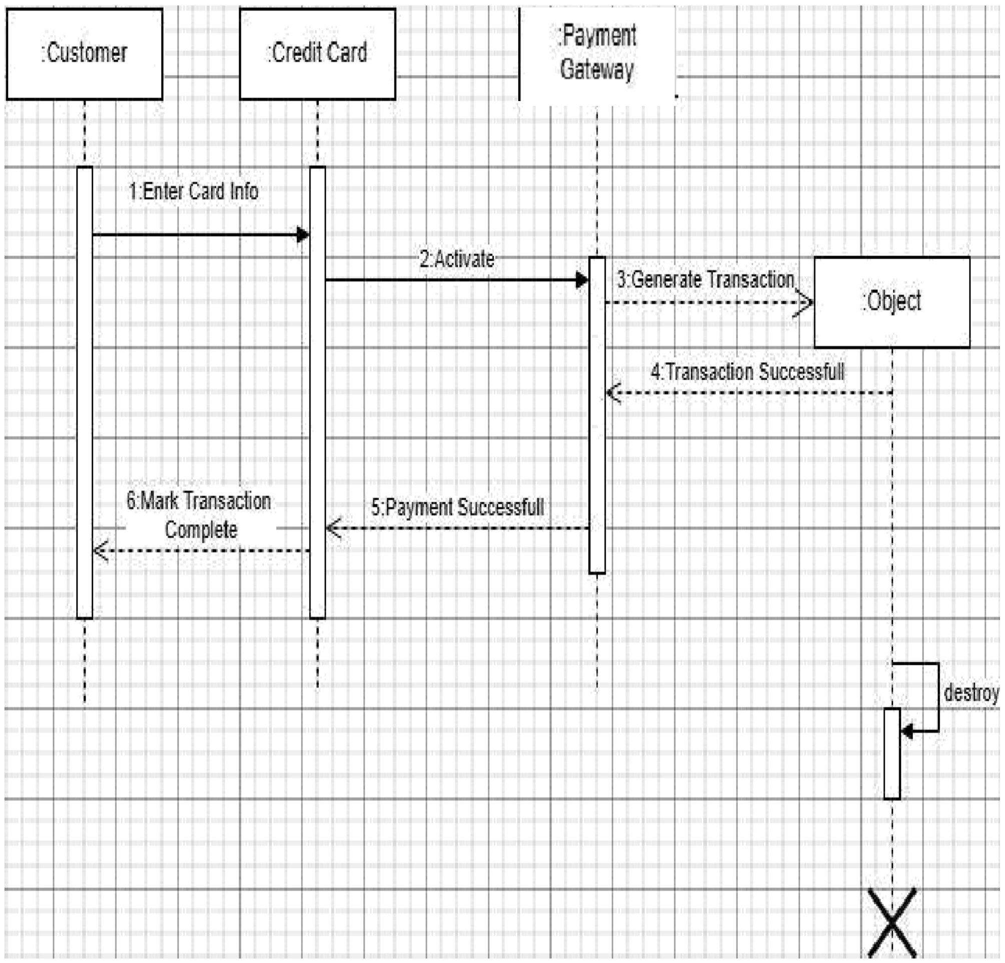
- The customer selects the “Make Payment” option. The system displays the total bill amount.
- The customer chooses a payment method (UPI, card, banking, etc.). The system sends the payment request to the payment gateway.
- The customer enters payment details.
- The payment gateway validates the details 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.

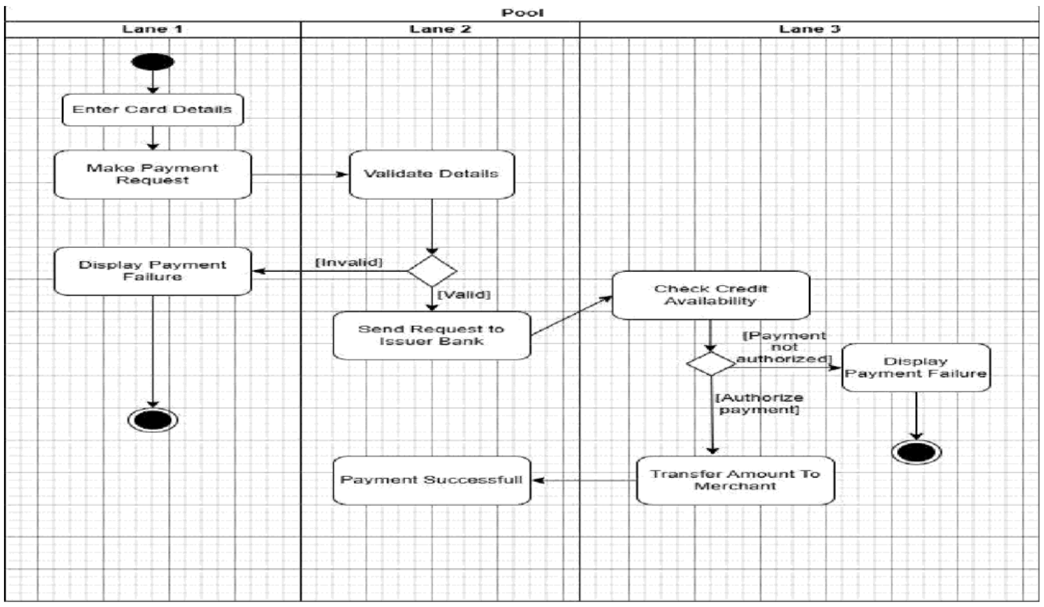
The customer account is created and stored in the database. The system confirms successful registration.

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.

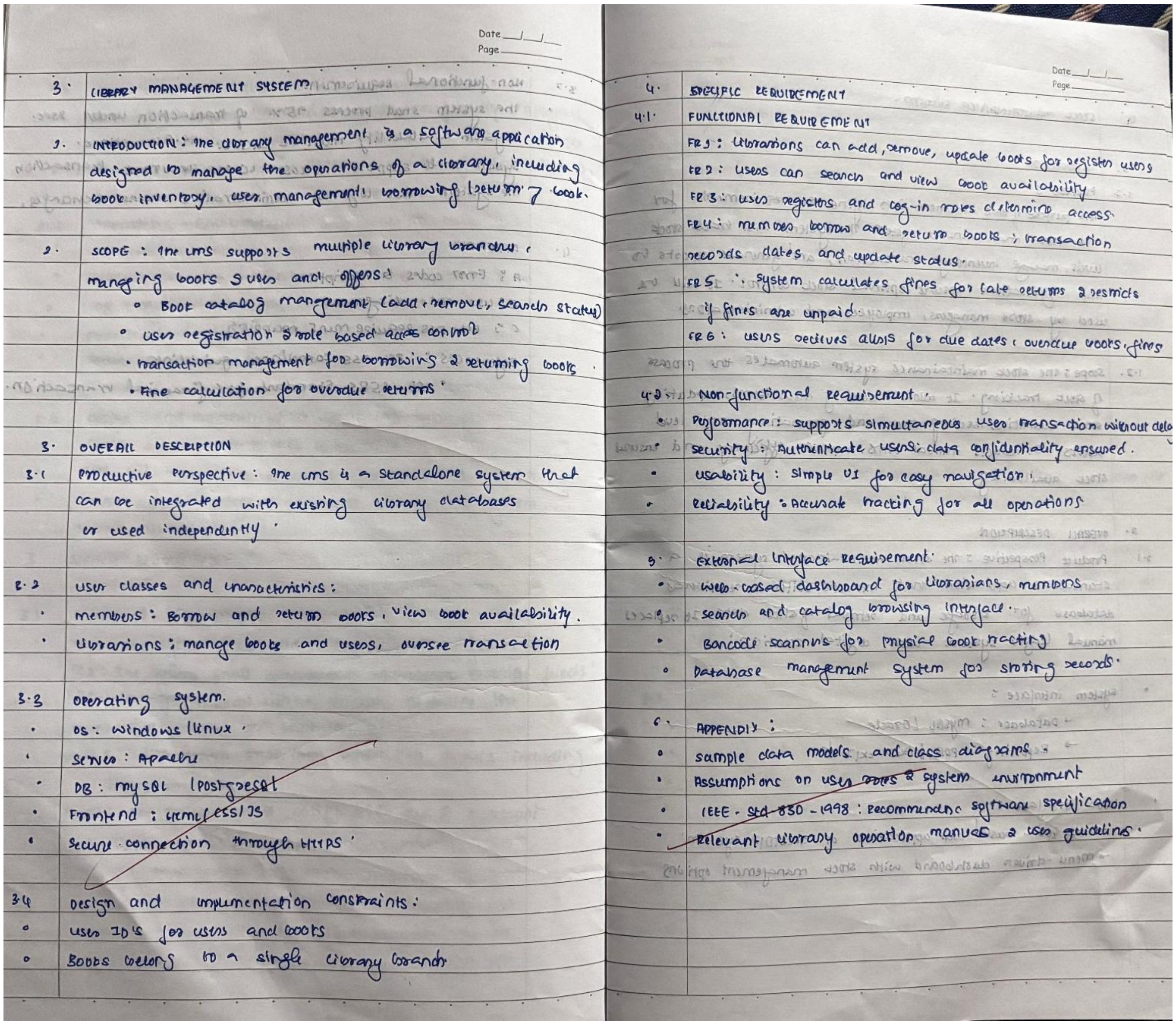
Activity Diagram(Advanced)



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



Date __/__/__
Page ____

3. LIBRARY MANAGEMENT SYSTEM

1. INTRODUCTION: The library management is a software application designed to manage the operations of a library, including book inventory, user management, borrowing [return] book.

2. SCOPE: The LMS supports multiple library branches, managing books, users and offers:

- Book catalog management (add, remove, search status)
- user registration & role based access control
- transaction management for borrowing & returning books.
- Fine calculation for overdue returns.

3. OVERALL DESCRIPTION

3.1 Productive perspective: The LMS is a standalone system that can be integrated with existing library databases or used independently.

3.2 user classes and characteristics:

- members: Borrow and return books, view book availability.
- Librarians: manage books and users, oversee transaction

3.3 operating system.

- OS: windows/linux.
- server: Apache
- DB: mySQL /postgreSQL
- Frontend: HTML/CSS/JS
- Secure connection through HTTPS.

3.4 Design and implementation constraints:

- uses ID's for users and books
- Books belong to a single library branch

Date __/__/__
Page ____

4. SPECIFIC REQUIREMENT

4.1. FUNCTIONAL REQUIREMENT

FR1: Librarians can add, remove, update books for register users

FR2: users can search and view book availability

FR3: users registers and log-in roles determine access.

FR4: members borrow and return books; transaction records dates and update status.

FR5: system calculates fines for late returns & restricts if fines are unpaid.

FR6: users receives alerts for due dates, overdue books, fines

4.2 Non-functional requirement:

- Performance: supports simultaneous user transaction without delay
- security: Authenticate users; data confidentiality ensured.
- usability: simple UI for easy navigation.
- Reliability: Accurate tracking for all operations

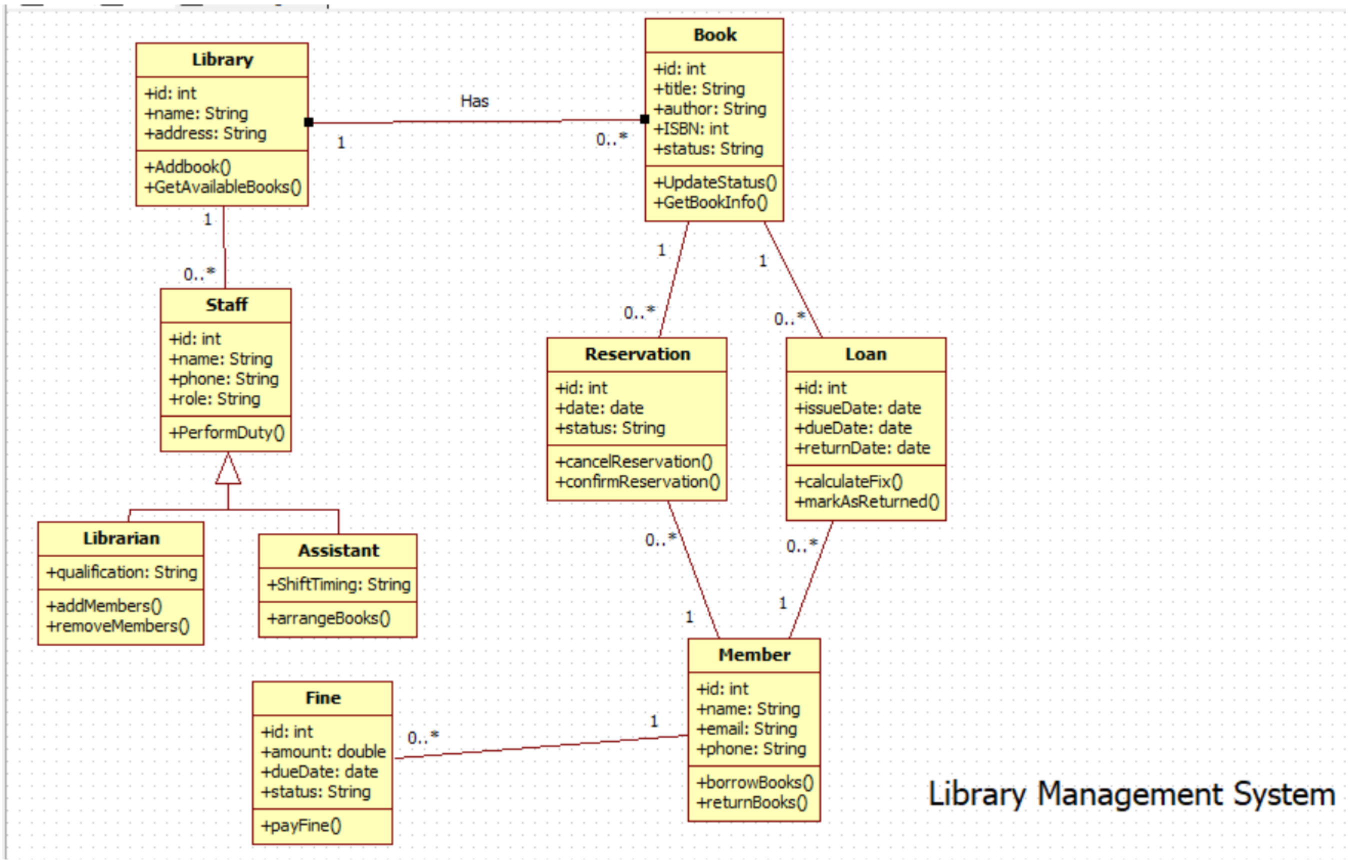
5. External Interface Requirement:

- web-based dashboard for librarians, members
- search and catalog browsing interface.
- Barcode scanners for physical book tracking
- Database management system for storing records.

6. APPENDIX:

- sample data models and class diagrams.
- Assumptions on user roles & system environment
- IEEE-Std-830-1998: Recommended software specification
- Relevant library operation manuals & user guidelines.

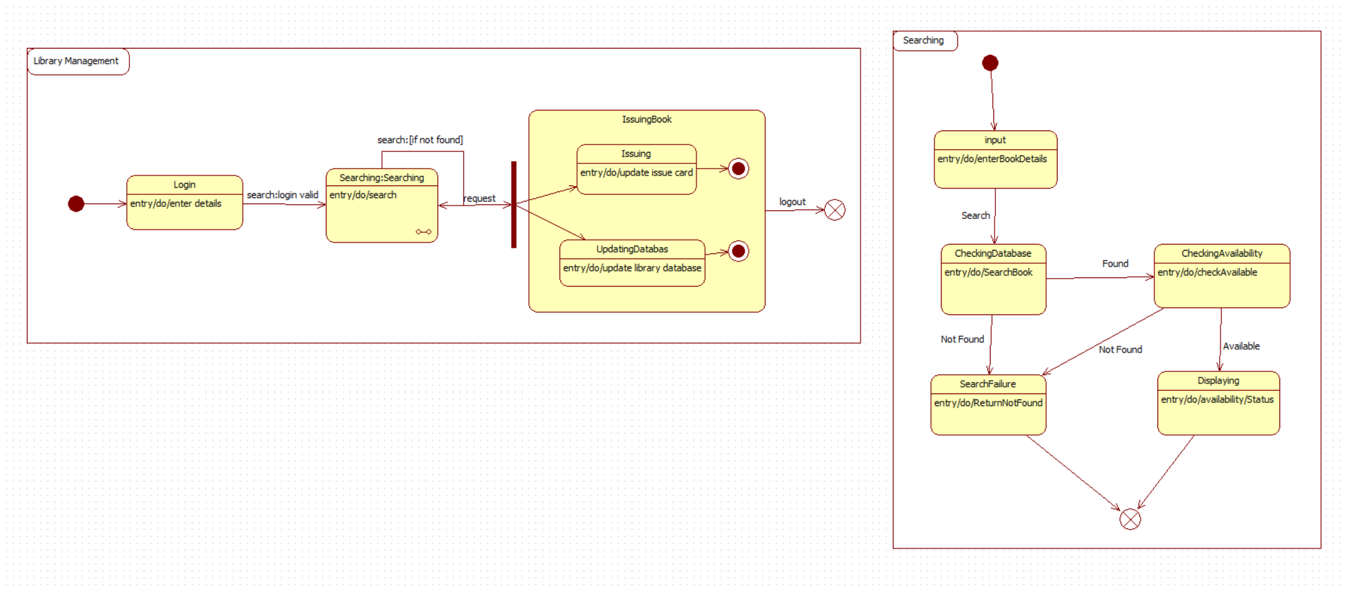
Class Diagram



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.

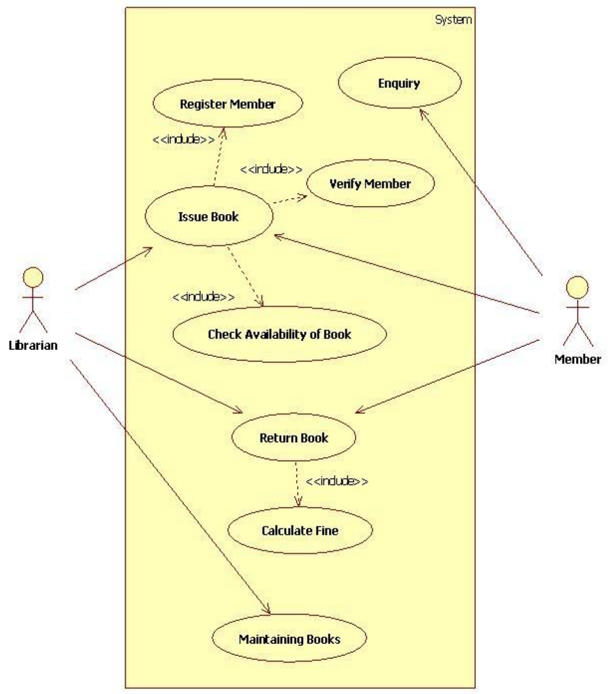
The system also keeps track of borrowing activities using the Transaction class, which stores the transaction ID, book ID, issue date, and due date. When members return books late, the Fine class is used to generate bills, update fines, and store payment information. Members pay these fines, and the 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 25 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

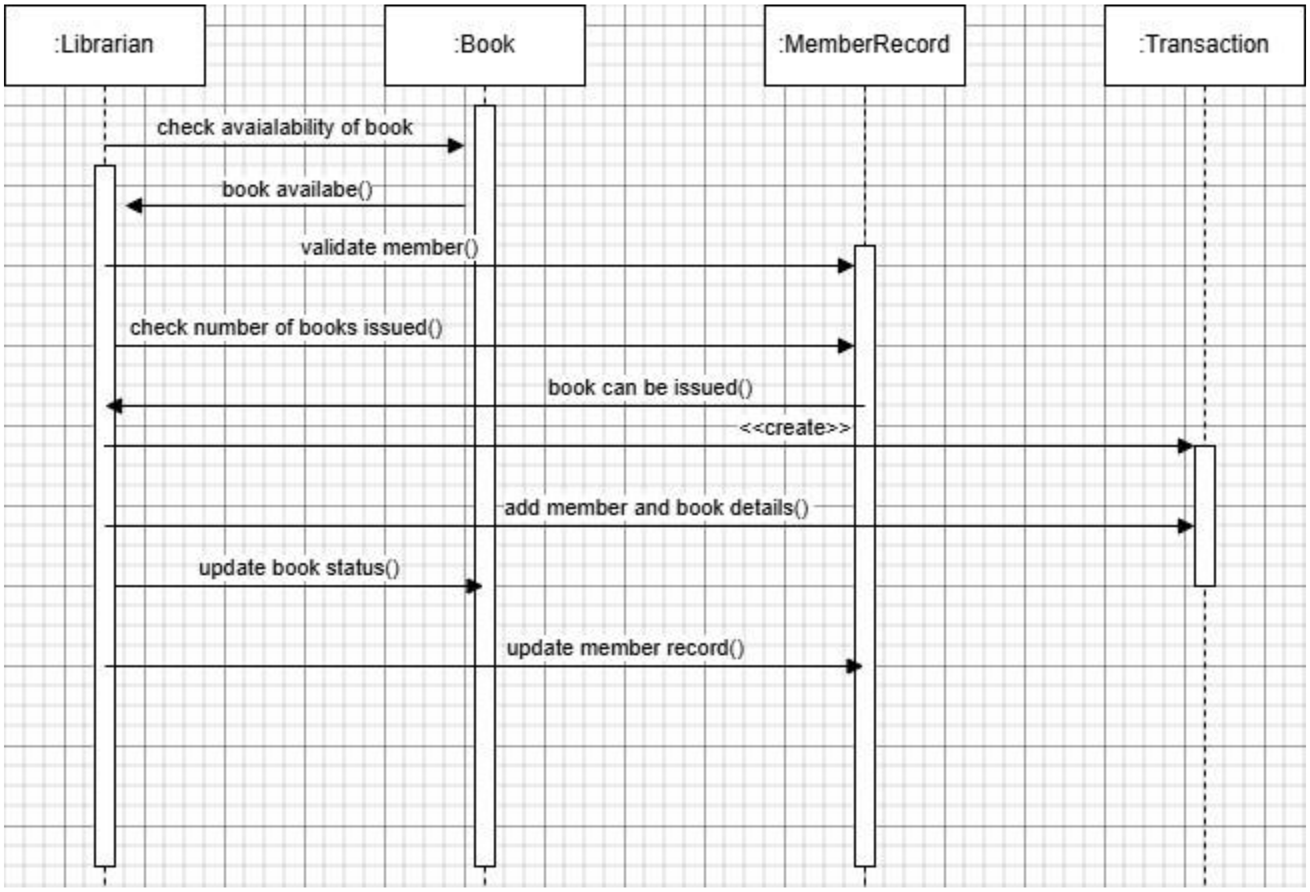
- The librarian selects the “Issue Book” option.
- The librarian enters the member ID and book ID.
- The system checks the member's status and borrowing limit.
- The system verifies the availability of the book.
- If valid, the system updates the issue record and marks the book as borrowed.
- The librarian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

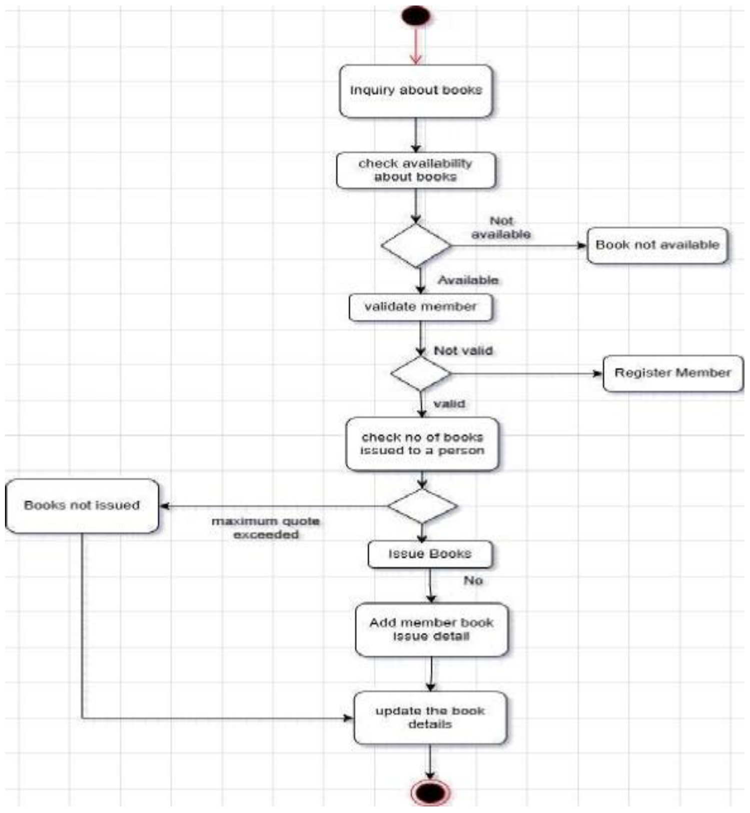
  The student enters the details.
- The system scans the database for matching books.
- The system displays the availability status (Available / Issued / Not Found).

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.

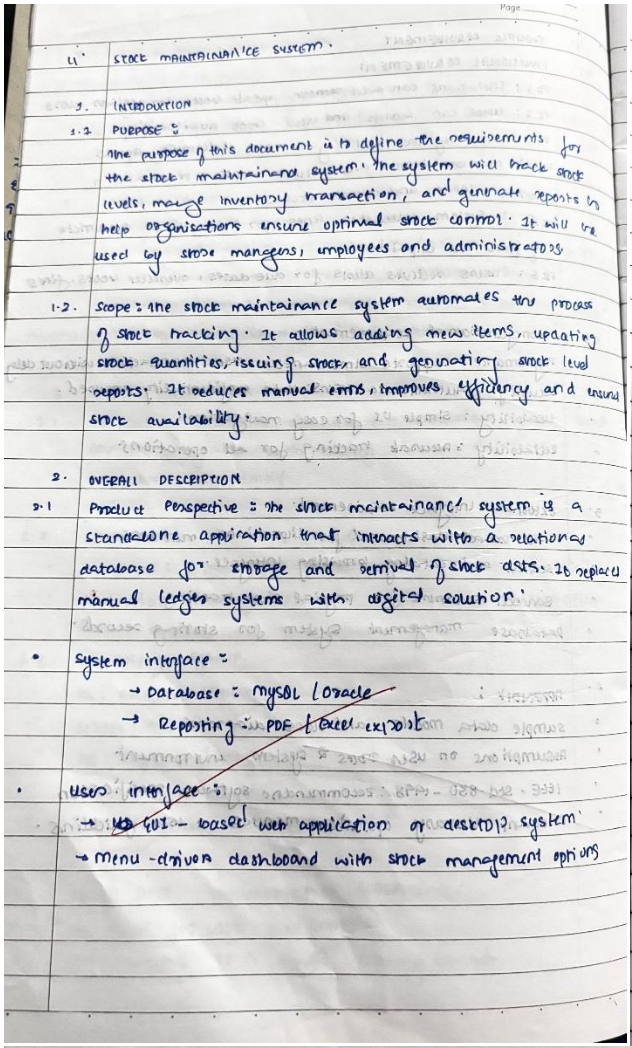
Activity Diagram(Advanced)



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:



4. STOCK MAINTAINANCE SYSTEM.

1. INTRODUCTION

1.1 PURPOSE :

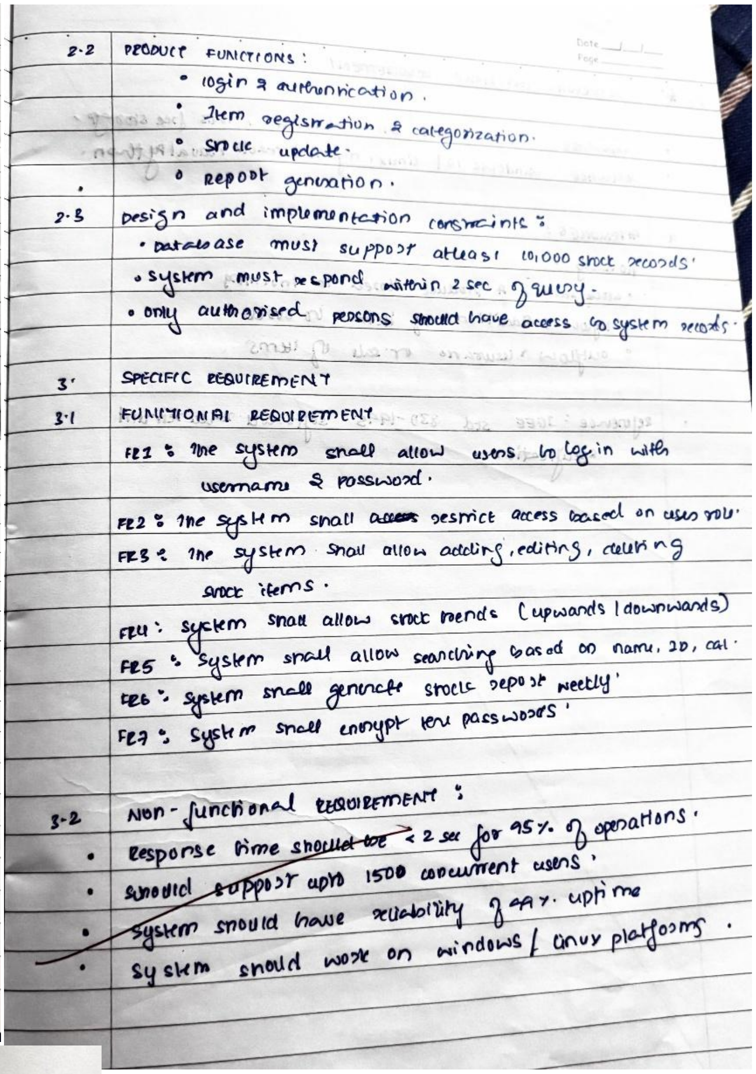
The purpose of this document is to define the requirements for the stock maintainance system. The system will track stock levels, manage inventory transaction, and generate reports to help organisations ensure optimal stock control. It will be used by store managers, employees and administrators.

1.2. Scope : The stock maintainance system automates the process of stock tracking. It allows adding new items, updating stock quantities, issuing stock, and generating stock level reports. It reduces manual errors, improves efficiency and ensures stock availability.

2. OVERALL DESCRIPTION

2.1 Product Perspective : The stock maintainance system is a standalone application that interacts with a relational database for storage and retrival of stock data. It replaces manual ledger systems with digital solution.

- System interface :
  - → Database : MySQL / Oracle
  - → Reporting : PDF / Excel export
- User interface :
  - → ~~UI~~ GUI - based web application or desktop system.
  - → menu - driven dashboard with stock management options



2.2 PRODUCT FUNCTIONS :

- login & authentication.
- Item registration & categorization.
- Stock update.
- Report generation.

2.3 Design and implementation constraints :

- Database must support atleast 10,000 stock records.
- System must respond within 2 sec of query.
- Only authorised persons should have access to system records.

3. SPECIFIC REQUIREMENT

3.1 FUNCTIONAL REQUIREMENT

FR1 : The system shall allow users to log in with username & password.

FR2 : The system shall ~~access~~ restrict access based on user role.

FR3 : The system shall allow adding, editing, deleting stock items.

FR4 : System shall allow stock trends (upwards / downwards)

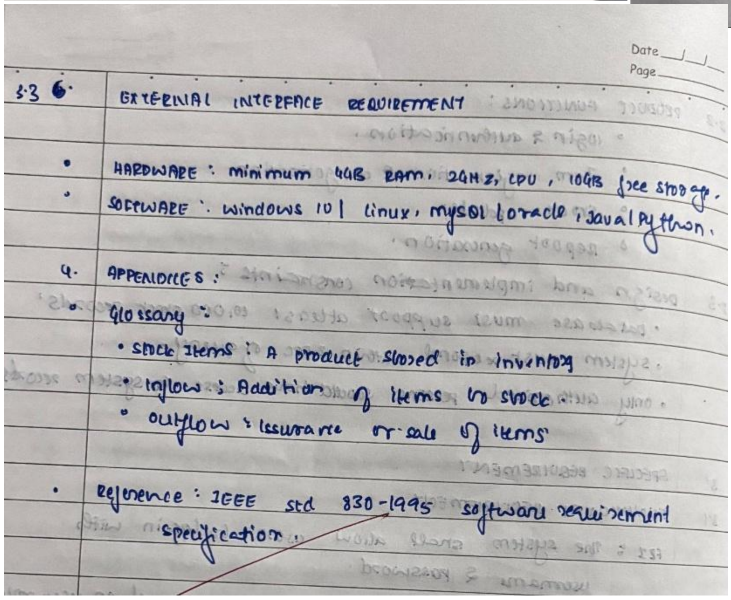
FR5 : System shall allow searching based on name, ID, cat.

FR6 : System shall generate stock report weekly.

FR7 : System shall encrypt the passwords.

3.2 Non-functional REQUIREMENT :

- Response time should be < 2 sec for 95% of operations.
- Should support upto 1500 concurrent users.
- System should have reliability of 99%. uptime
- System should work on windows / linux platforms.



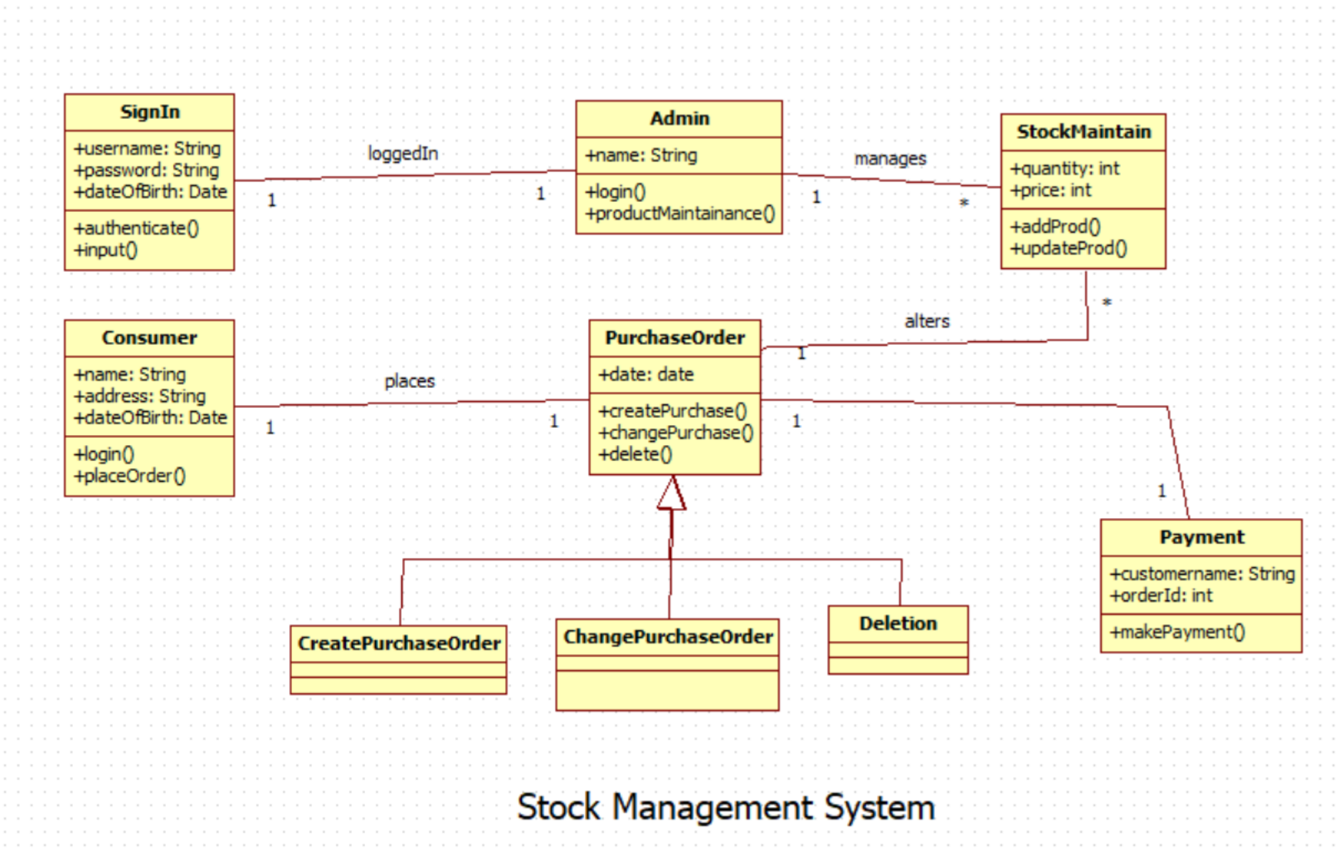
3.3 ~~6.~~ EXTERNAL INTERFACE REQUIREMENT

- HARDWARE : minimum 4GB RAM, 2GHz, CPU, 10GB free storage.
- SOFTWARE : Windows 10 / Linux, MySQL / Oracle, Java / Python.

4. APPENDICES :

- Glossary :
  - Stock Items : A product stored in inventory
  - Inflow : Addition of items to stock.
  - Outflow : Issurance or sale of items
- Reference : IEEE std 830-1995 software requirement specification.

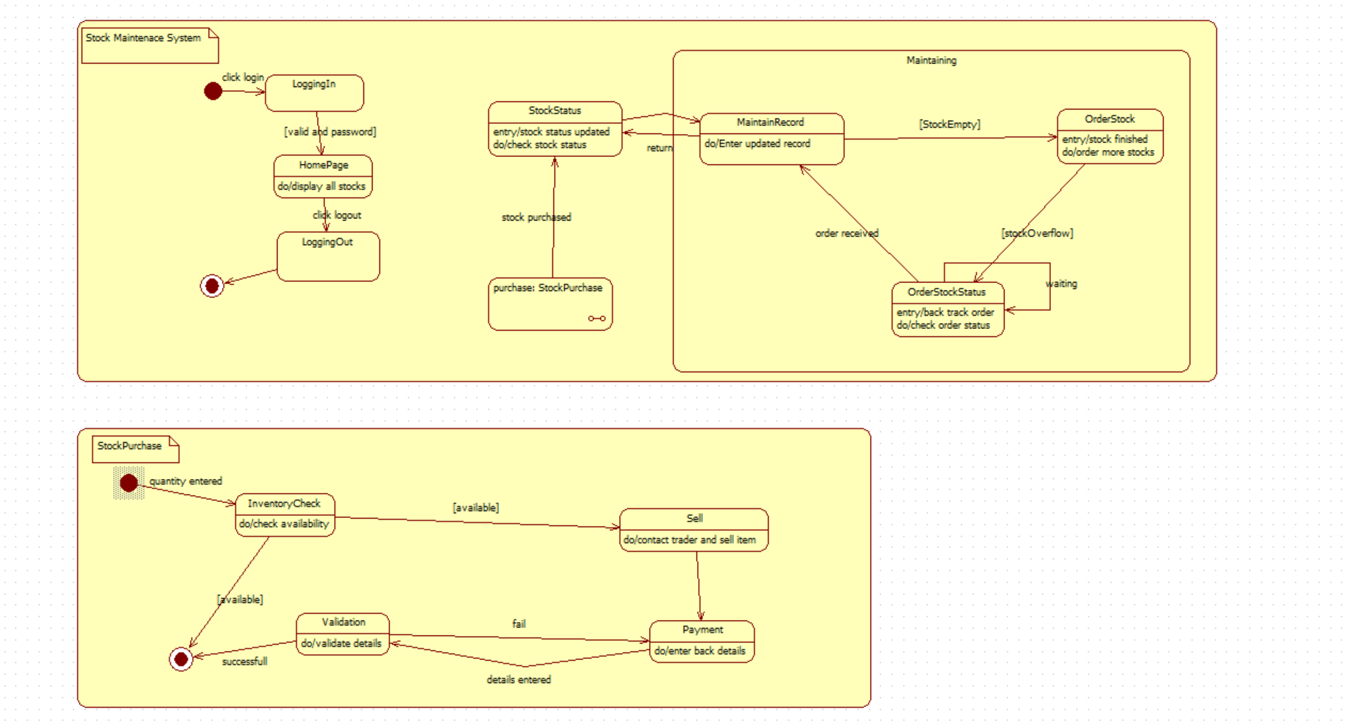
Class Diagram



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.

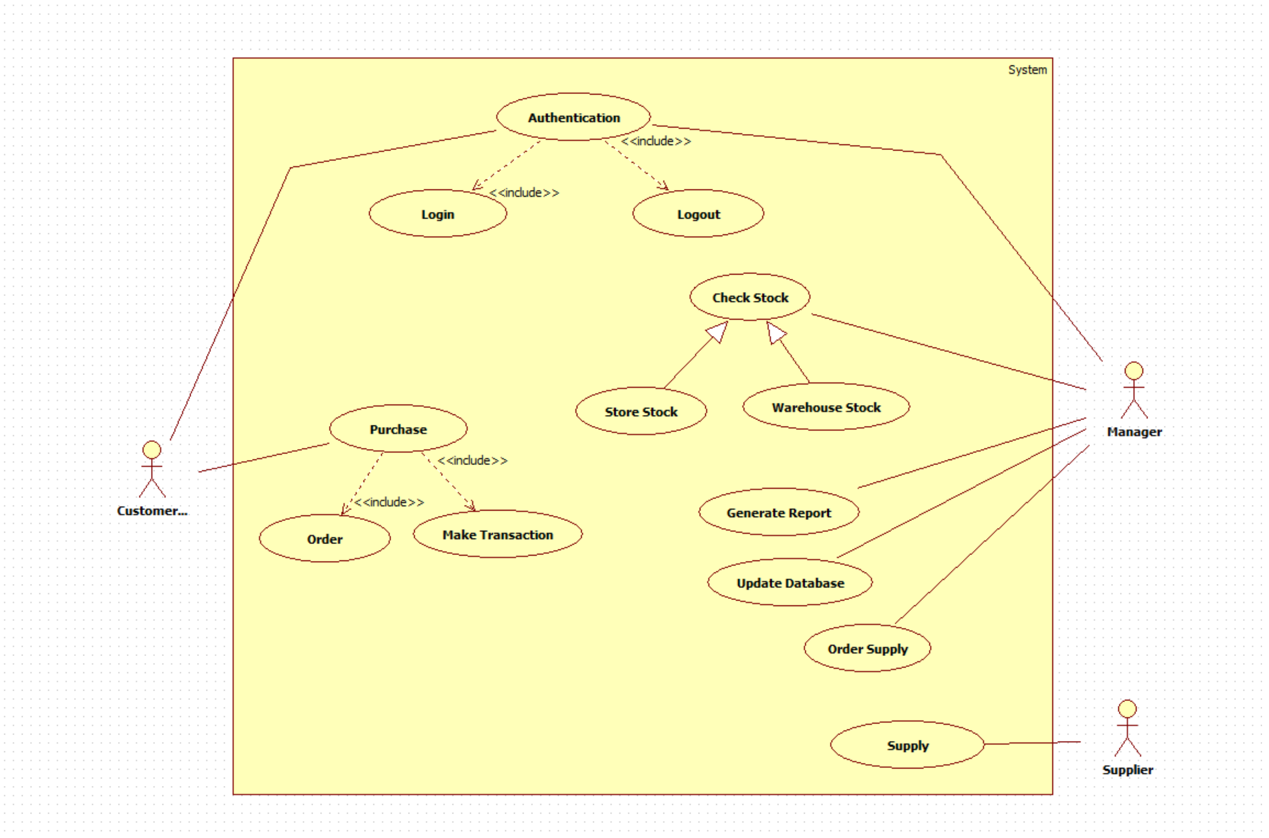
Advanced State Diagram



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 <> 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.

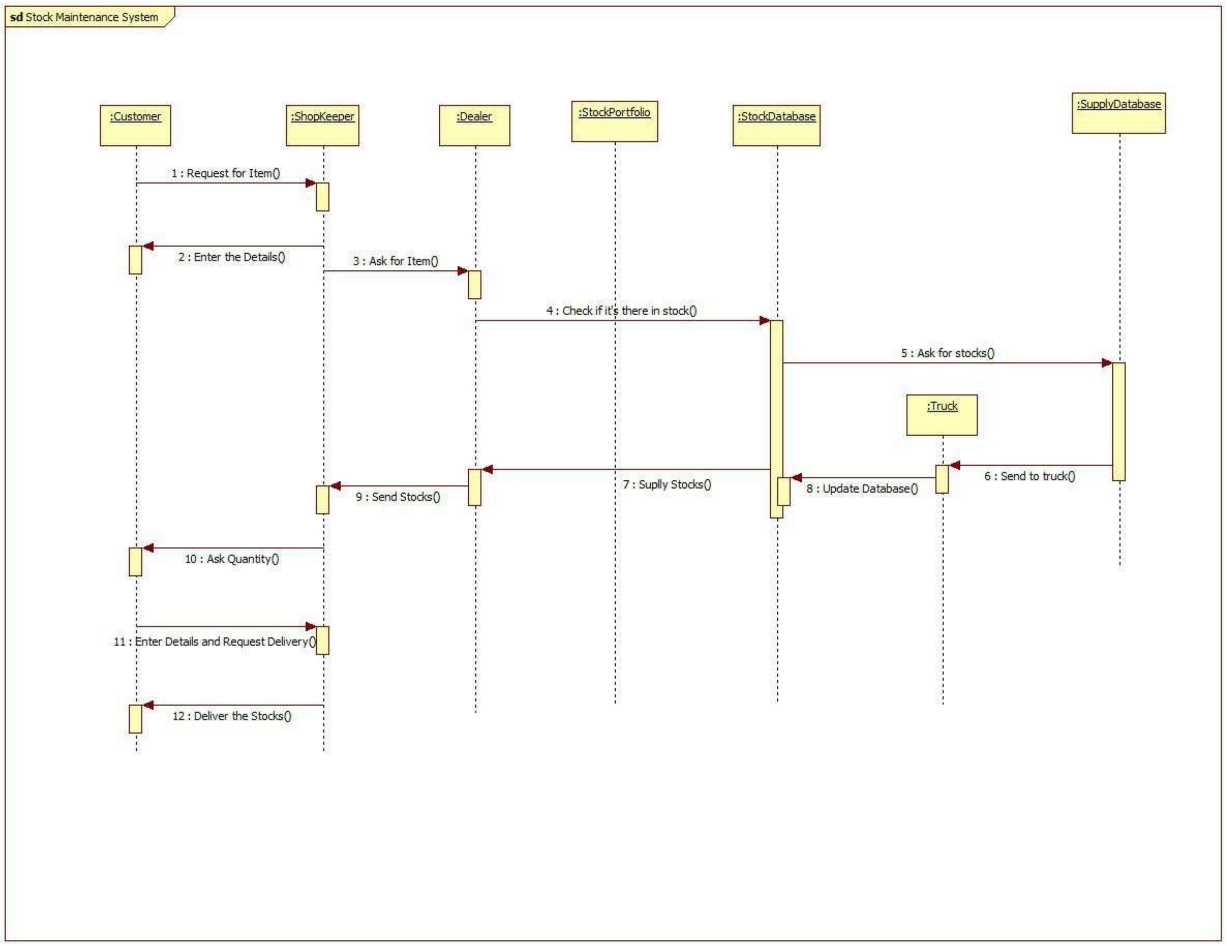
Scenario 1: Purchase and Stock Update (Succeeded)

- Purchase department raises a purchase requisition for 500 units of Item-X
- Supplier accepts PO and delivers 500 units with proper invoice
- Warehouse stuff 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 stuff performs quality check —8m 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

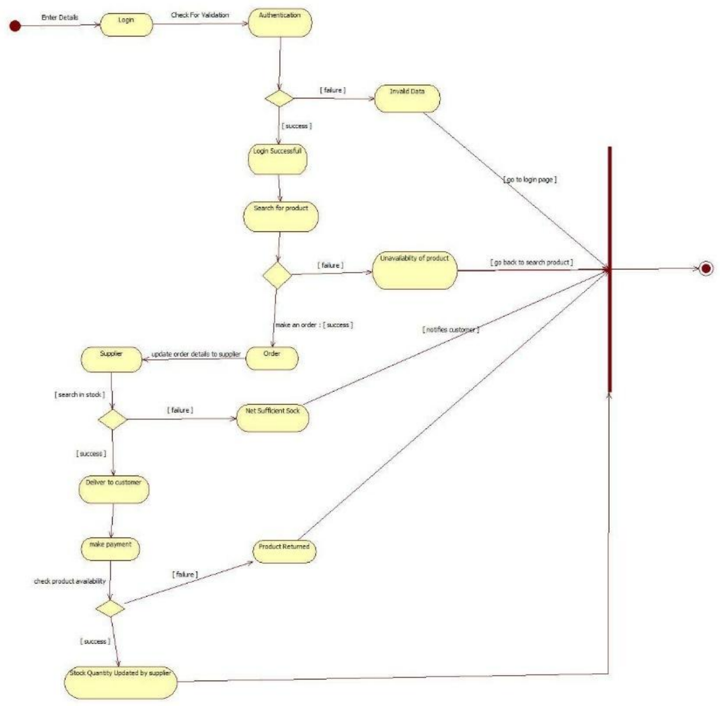
Advanced Sequence Diagram:



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:

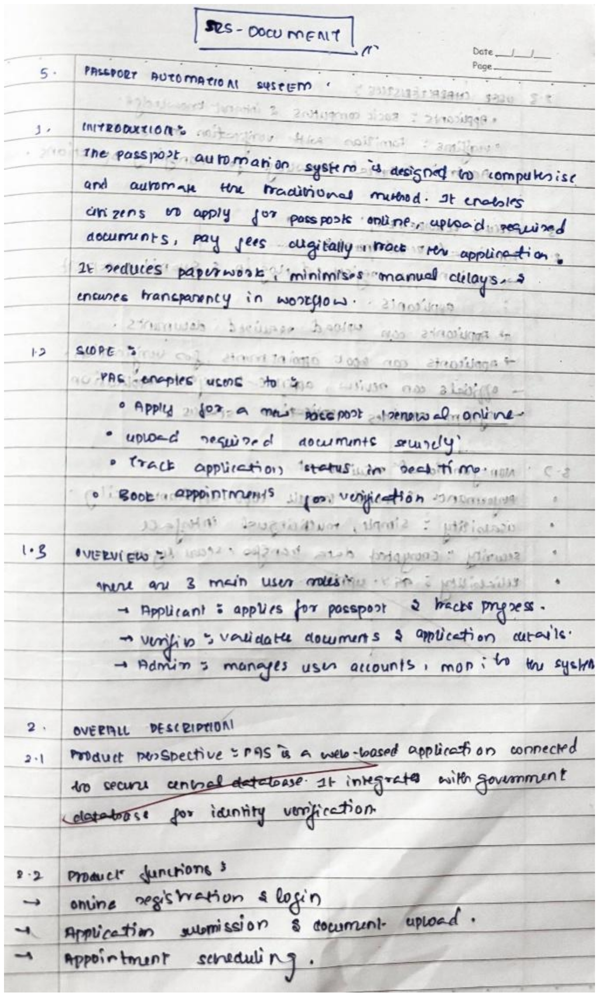


After the order is successfully placed, the system interacts with the supplier to fetch stock and order details. The supplier searches inventory and updates the system accordingly. If delivery fails, the customer is notified immediately. Successful deliveries lead to the payment phase, where transaction 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:



SRS - DOCUMENT

Date __/__/__
Page ____

5. PASSPORT AUTOMATIONAL SYSTEM

1. INTRODUCTION :

The passport automation system is designed to computerise and automate the traditional method. It enables citizens to apply for passports online, upload required documents, pay fees digitally, track the application. It reduces paperwork, minimises manual delays, & ensures transparency in workflow.

1.2 SCOPE :

PAS enables users to :

- Apply for a new passport / renewal online.
- upload required documents securely.
- Track application status in real time.
- Book appointments for verification

1.3 OVERVIEW :

There are 3 main user roles :

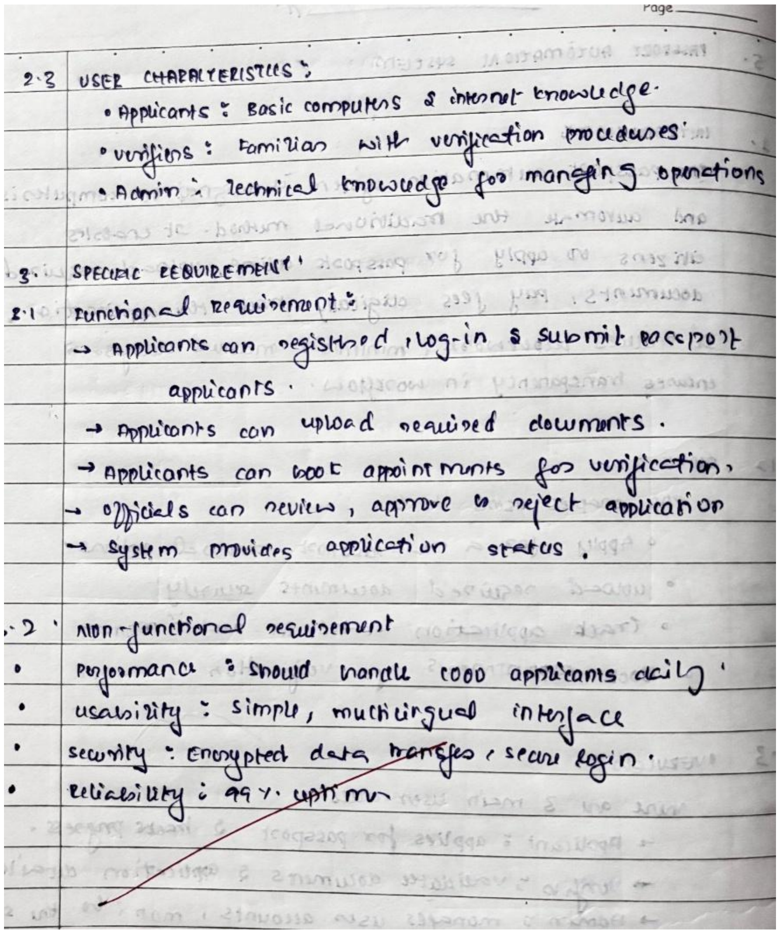
→ Applicant : applies for passport & tracks progress.
→ verifier : validates documents & application details.
→ Admin : manages user accounts, monitors the system

2. OVERALL DESCRIPTION

2.1 Product perspective : PAS is a web-based application connected to secure central database. It integrates with government database for identity verification.

2.2 Product functions :

→ online registration & login
→ Application submission & document upload.
→ Appointment scheduling.



Page ____

2.3 USER CHARACTERISTICS :

- Applicants : Basic computers & internet knowledge.
- verifiers : Familiar with verification procedures.
- Admin : Technical knowledge for managing operations

3. SPECIFIC REQUIREMENT :

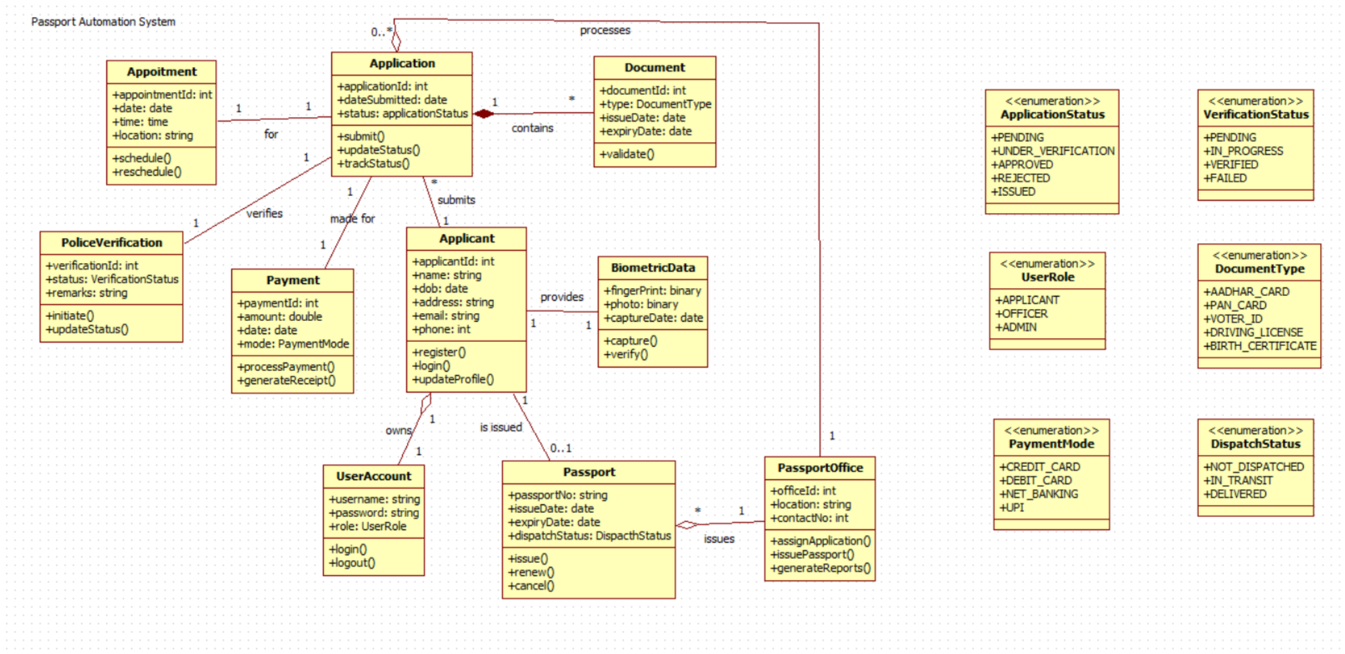
3.1 Functional requirement :

→ Applicants can register, log-in & submit passport applicants.
→ Applicants can upload required documents.
→ Applicants can book appointments for verification.
→ officials can review, approve or reject application
→ system provides application status.

3.2 Non-functional requirement

- Performance : should handle 1000 applicants daily.
- usability : simple, multilingual interface
- security : Encrypted data transfer, secure login.
- reliability : 99% uptime

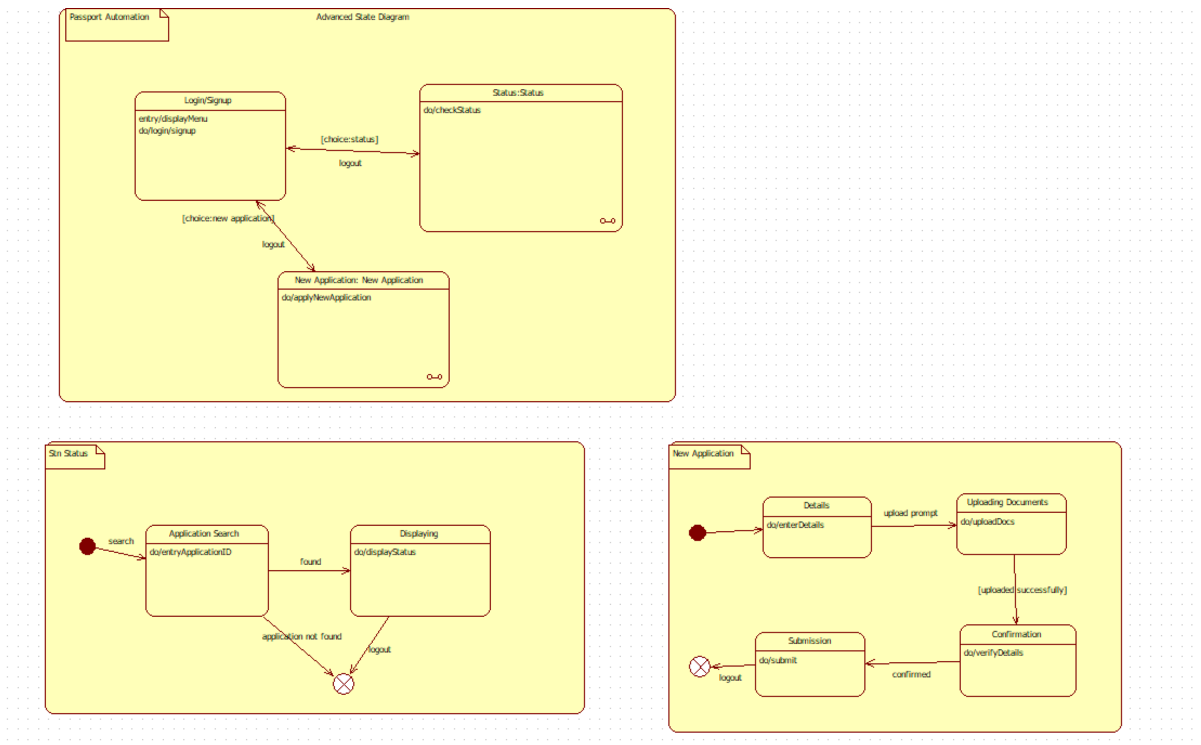
Class Diagram



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.

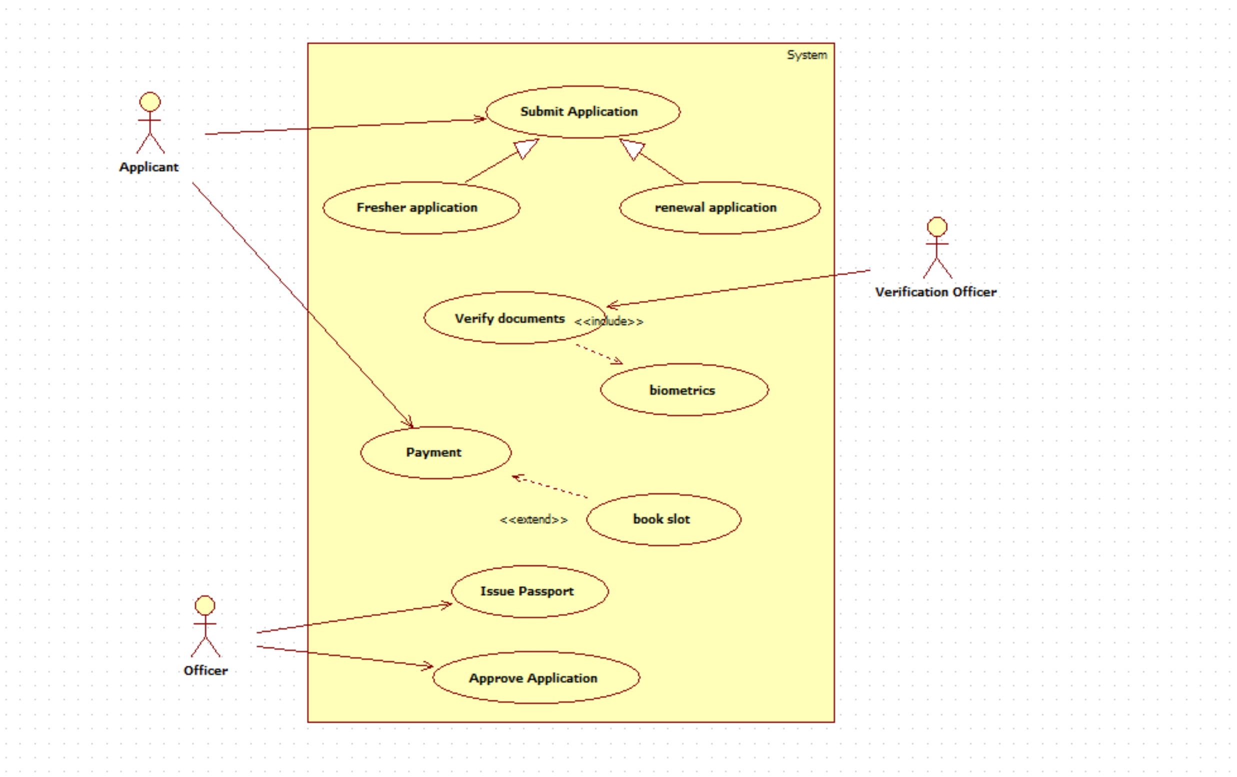
State Diagram:



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.

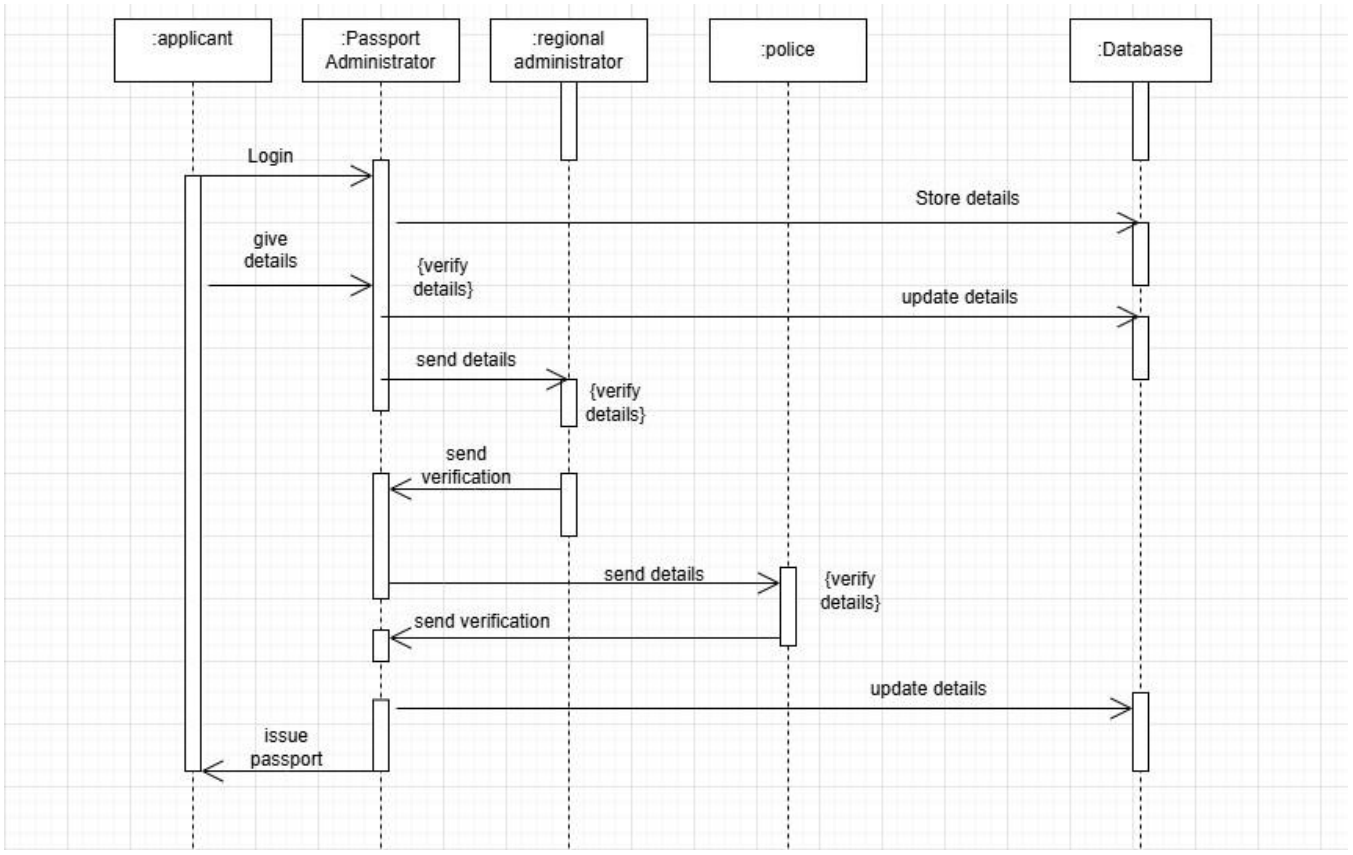
Scenario 1: Main Scenario (succeeded)

- 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)

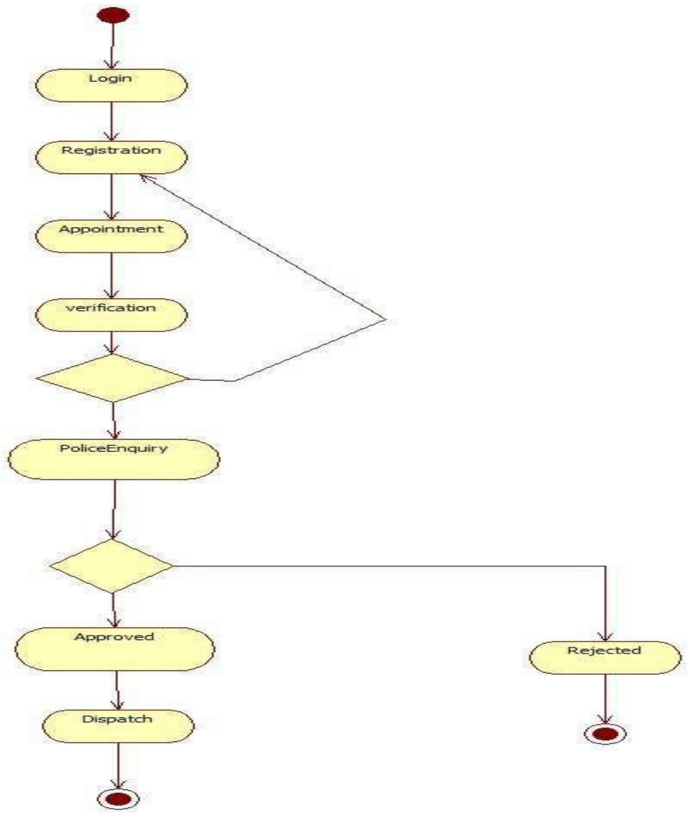
- Applicant completes registration, form filling and fee payment
- Attends appointment; document & biometric verification successful
- Police station visits applicant's address for enquiry
- System detects pending criminal case / FIR
- Police submit adverse verification report
- Regional authority reviews 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 (Objectl) 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.