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:

Nidhi Pramod Nambiar

1BM23CS212

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 Nidhi Pramod Nambiar (1BM23CS212) during the 5th Semester August 2025-December 2025

.

Signature of the Faculty Incharge:

Sunayana S
Assistant Professor

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

Table of Contents

Github link: https://github.com/nidhinambiar7/OOM

1. Hotel Management System

1.1 Problem Statement

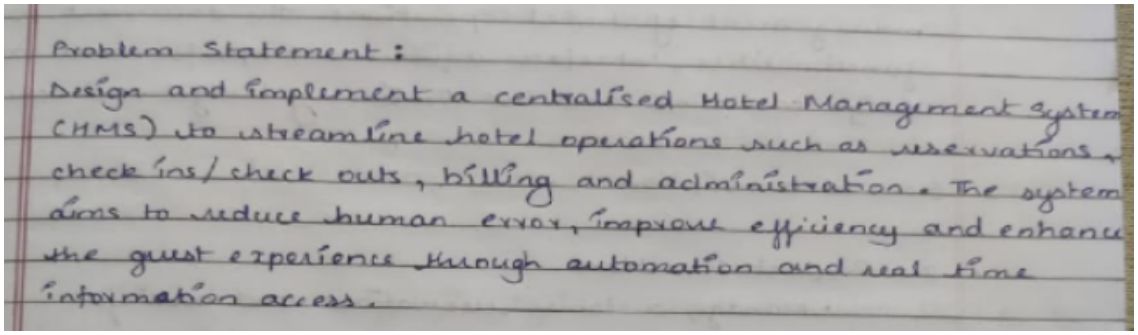
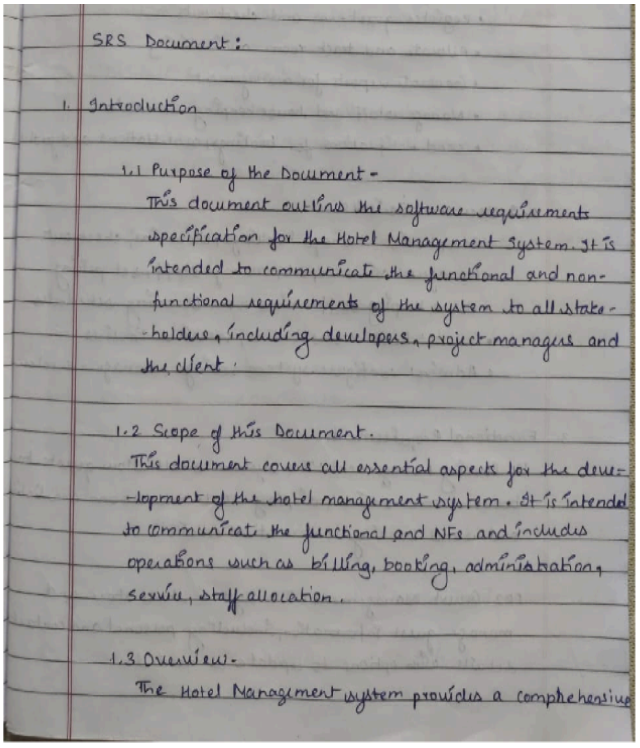

Problem Statement :
Design and implement a centralised Hotel Management System (HMS) to streamline hotel operations such as reservations, check ins/check outs, billing and administration. The system aims to reduce human error, improve efficiency and enhance the guest experience through automation and real time information access.

Fig 1.1.1

1.2 SRS-Software Requirements Specification


SRS Document :

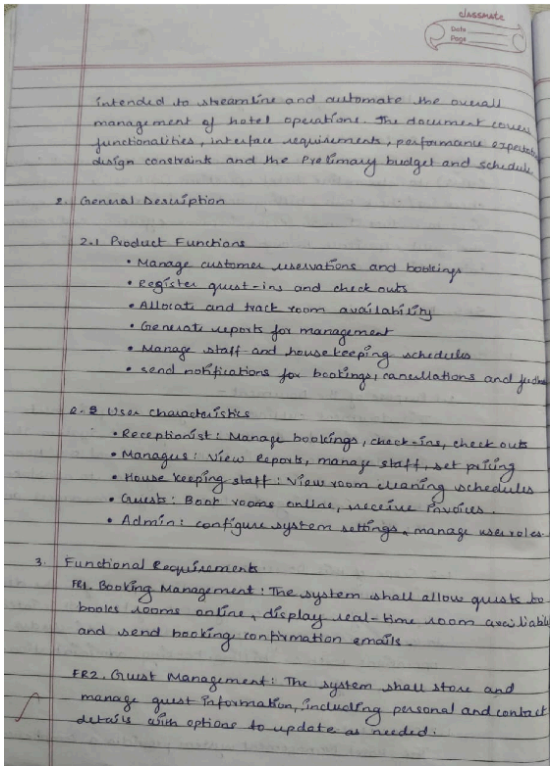
1. Introduction

1.1 Purpose of the Document -
This document outlines the software requirements specification for the Hotel Management System. It is intended to communicate the functional and non-functional requirements of the system to all stake-holders, including developers, project managers and the client.

1.2 Scope of this Document.
This document covers all essential aspects for the deve-lopment of the hotel management system. It is intended to communicate the functional and NFs and includes operations such as billing, booking, administration, service, staff allocation.

1.3 Overview.
The Hotel Management system provides a comprehensive

Fig 1.2.1


intended to streamline and automate the overall management of hotel operations. The document covers functionalities, interface requirements, performance expecta design constraints and the preliminary budget and schedule

2. General Description

2.1 Product Functions
- Manage customer reservations and bookings
- Register guest-ins and check outs
- Allocate and track room availability
- Generate reports for management
- Manage staff and housekeeping schedules
- send notifications for bookings, cancellations and fe

2.2 User characteristics
- Receptionist: Manage bookings, check-ins, check outs
- Manager: View reports, manage staff, set pricing
- House keeping staff: View room cleaning schedules
- Guests: Book rooms online, receive invoices.
- Admin: configure system settings, manage user roles

3. Functional Requirements
FR1. Booking Management: The system shall allow guests to books rooms online, display real-time room availabi and send booking confirmation emails.

FR2. Guest Management: The system shall store and manage guest information, including personal and contact details with options to update as needed.

Fig 1.2.2

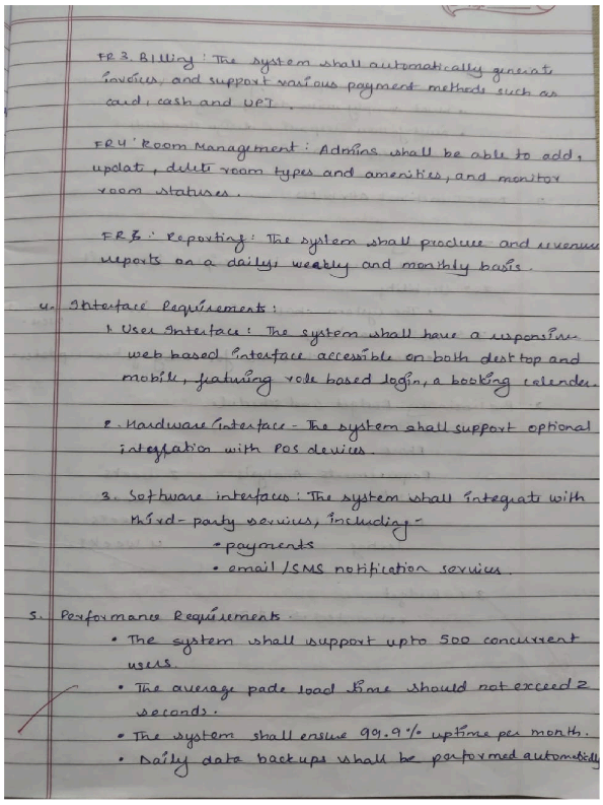


Fig 1.2.3

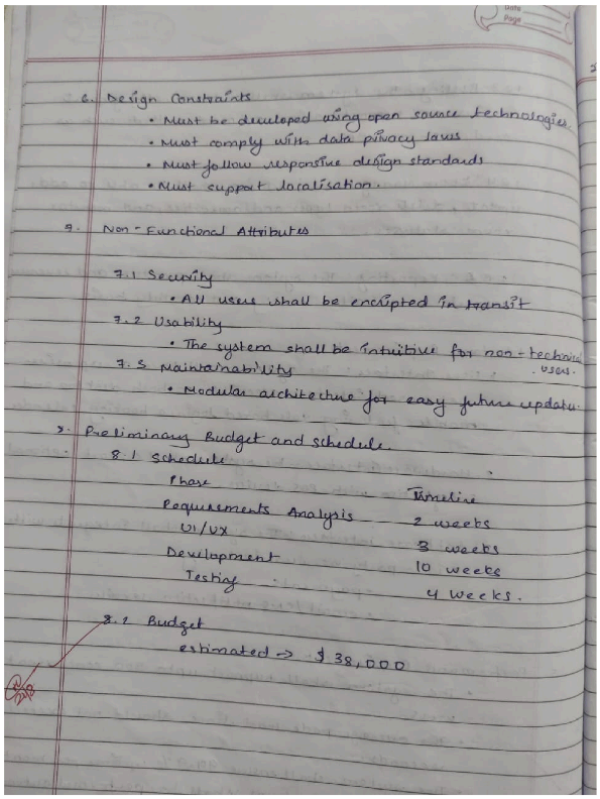


Fig 1.2.4

1.3 Class Diagram

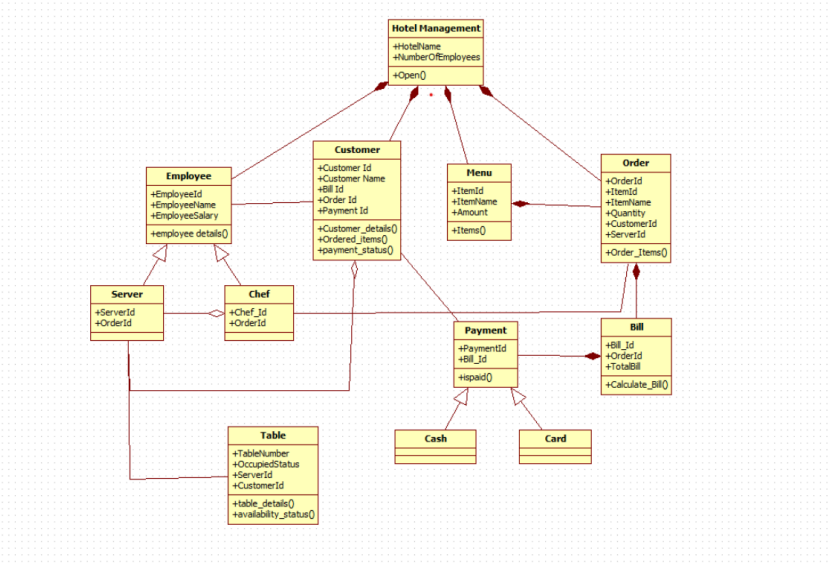


Fig 1.3.1 Hotel Management Class Diagram

This UML class diagram models a hotel management system and describes how key entities interact within the software.

- The Hotel Management class is central, connecting employees, customers, menu, and orders.
- Employee includes general staff details and links to specific roles like Server and Chef.
- Customer holds information about each guest, their orders, assigned tables, and payment status.
- Menu lists available items and prices, while Order tracks individual orders with data like item name, quantity, and server assignment.
- Table shows seating status and allocation to customers and servers.
- Bill and Payment handle the financial side, including billing calculation and cash/card payment processing.

Each class and relationship helps manage hotel operations smoothly by clearly organizing how staff, guests, orders, and payments work together

1.4 State Diagram

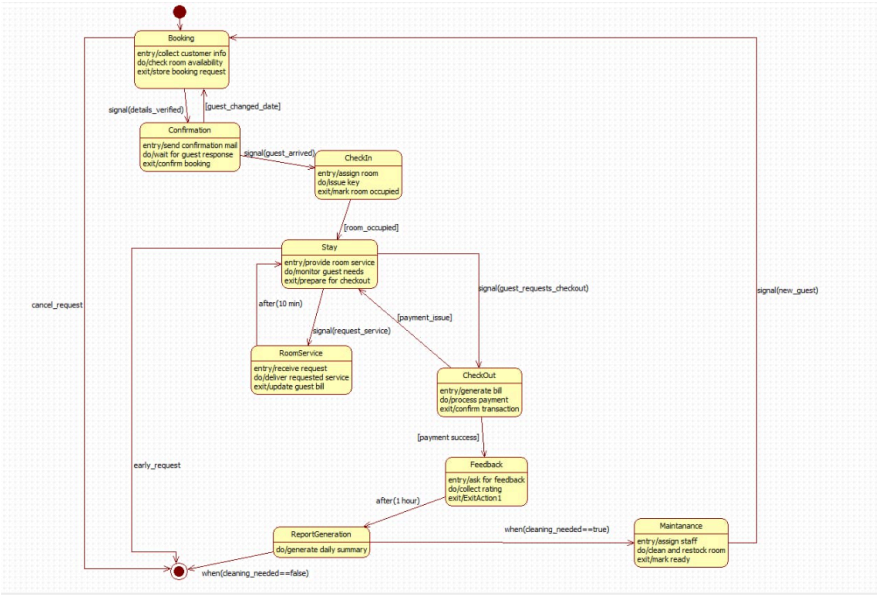


Fig 1.4.1 Simple State diagram

This diagram represents a state machine (statechart) for a hotel room booking and management system.

- It starts with Booking and Confirmation, where a customer requests and receives confirmation for a room.
- Upon arrival, the process moves to CheckIn, changing room status to occupied.

- The room may then switch between being in use (occupied), and go through RoomService if required.
- When the guest requests checkout, the flow moves to Checkout, where payment and feedback are handled.
- After checkout, the process may involve Maintenance if cleaning or repairs are needed, or ReportGeneration for daily summaries.

Transitions show the steps a room takes from booking to vacating, including key actions like arrival, payment, feedback, and maintenance, reflecting the hotel's end-to-end room lifecycle management.

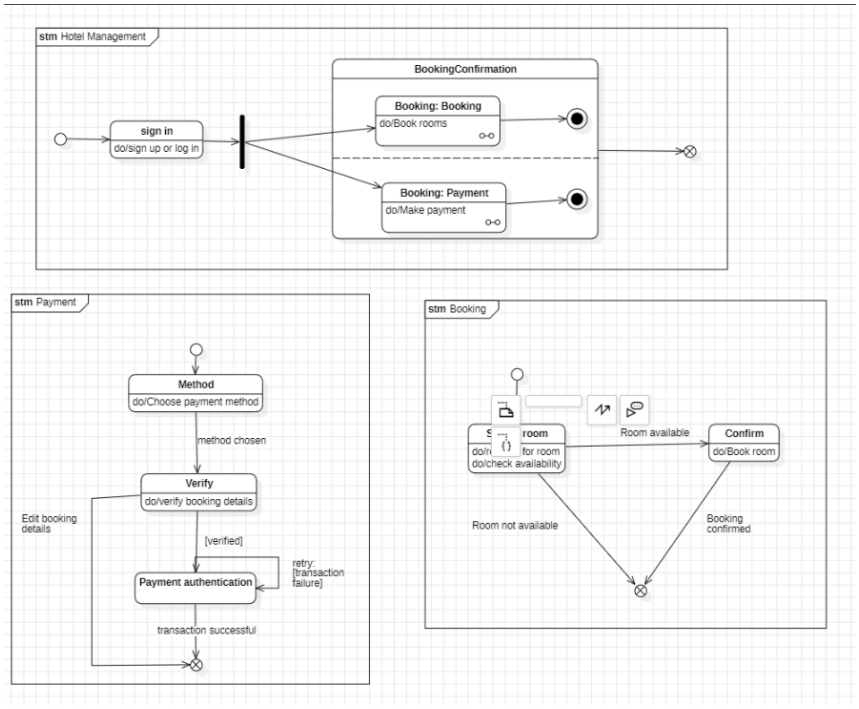


Fig 1.4.2 Advanced State diagram

This image shows three state machine diagrams for a hotel management system focused on booking, payment, and user authentication processes.

- Hotel Management (top left): Users start by signing up or logging in. After successful authentication, they can proceed to booking and payment activities.
- Payment (bottom left): The flow guides users through choosing a payment method, verifying booking details, and authenticating the transaction. It includes a retry option for failed payment attempts and editing booking details.

- Booking (bottom right): Users check room availability. If a room is available, they book the room and receive confirmation; if unavailable, the booking attempt ends.

These diagrams capture the essential steps for secure sign-in, making hotel bookings, and handling payments, with clear transitions for successful and unsuccessful scenarios.

1.5 Use-Case Diagram

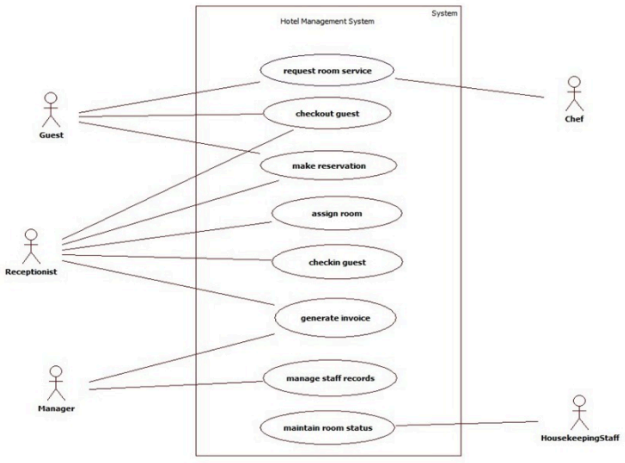


Fig 1.5.1 Simple Use Case diagram

This use-case diagram illustrates the main functions and actors in a hotel management system. Guests can request room service and checkout, while receptionists handle reservations, room assignments, guest check-ins, guest checkouts, and invoice generation. Managers are responsible for managing staff records and overall room status. Chefs fulfill room service requests, and housekeeping staff maintain the status and cleanliness of rooms. The diagram highlights how each role interacts with core hotel processes to ensure efficient service and administration.

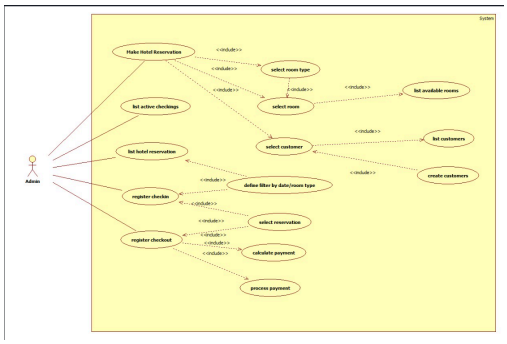


Fig 1.5.2 Advanced Use Case diagram

This diagram is a UML use case diagram for hotel management showing the functions accessible to an admin user.

- The admin can make hotel reservations, view active check-ins, register check-in and checkout, and manage reservations.
- Each main action can include sub-tasks, such as selecting room type, listing available rooms, selecting a customer, creating customers, and using filters by date or room type.
- Further use cases allow the admin to select reservations, calculate payments, and process payments.

The diagram structures hotel operations around the admin's workflow, visually connecting related tasks needed for day-to-day management.

1.6 Sequence Diagram

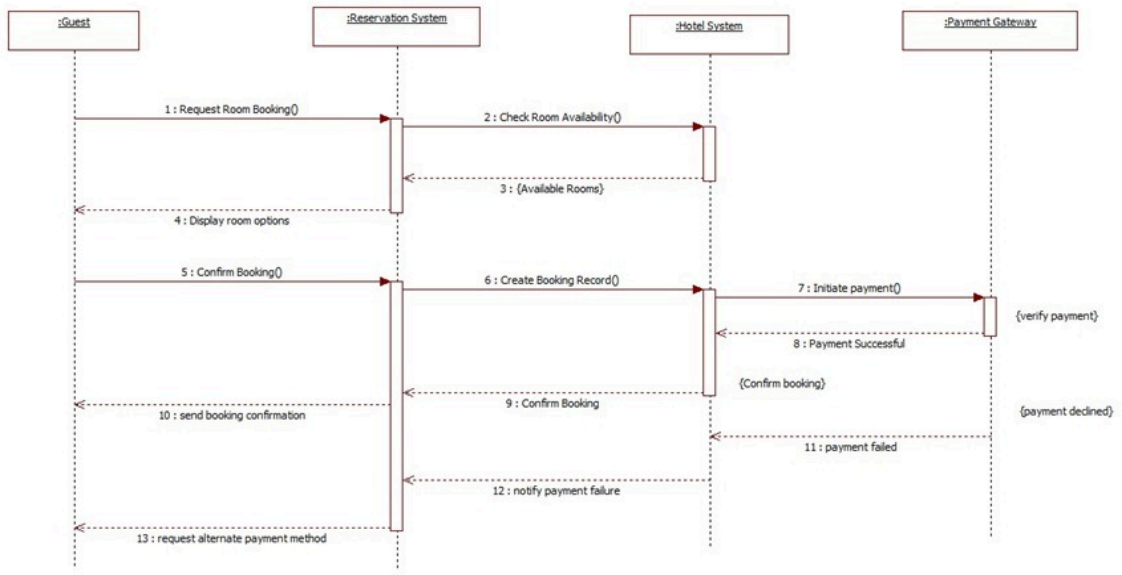


Fig 1.6.1 Simple Sequence diagram

This sequence diagram shows the main steps and system interactions involved in booking a hotel room and processing payment. The guest initiates a room booking request, after which the reservation system checks room availability with the hotel system. Available rooms are displayed to the guest, who then confirms the booking. The reservation system creates a booking record and initiates payment via the payment gateway. If payment is successful, the booking is

confirmed and the guest is notified. If payment fails, the guest receives a failure notification and may be prompted to choose another payment method.

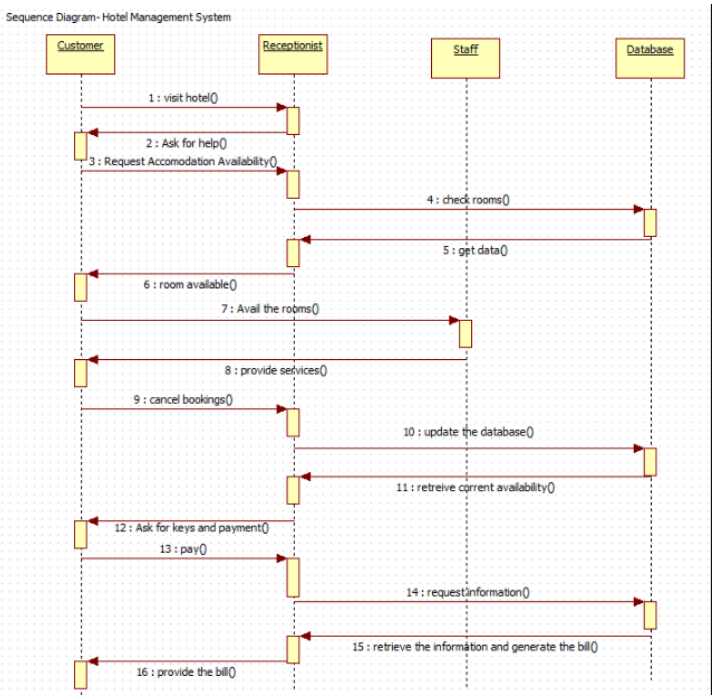


Fig 1.6.2 Advanced Sequence diagram

This sequence diagram shows the flow of hotel management operations step by step among the customer, receptionist, staff, and database.

- The customer arrives, asks for accommodation, and the receptionist checks room availability with help from staff and the database.
- If a room is available, the customer can avail services, register, pay, and even cancel bookings. The receptionist coordinates with the staff and updates information in the database throughout the process.
- Database interactions include retrieving current availability, updating records when bookings change, and generating the bill after payment.

It visually represents how requests move between people and systems, ensuring each task is completed in the right order for booking, availing rooms, and processing payments.

1.7 Activity Diagram

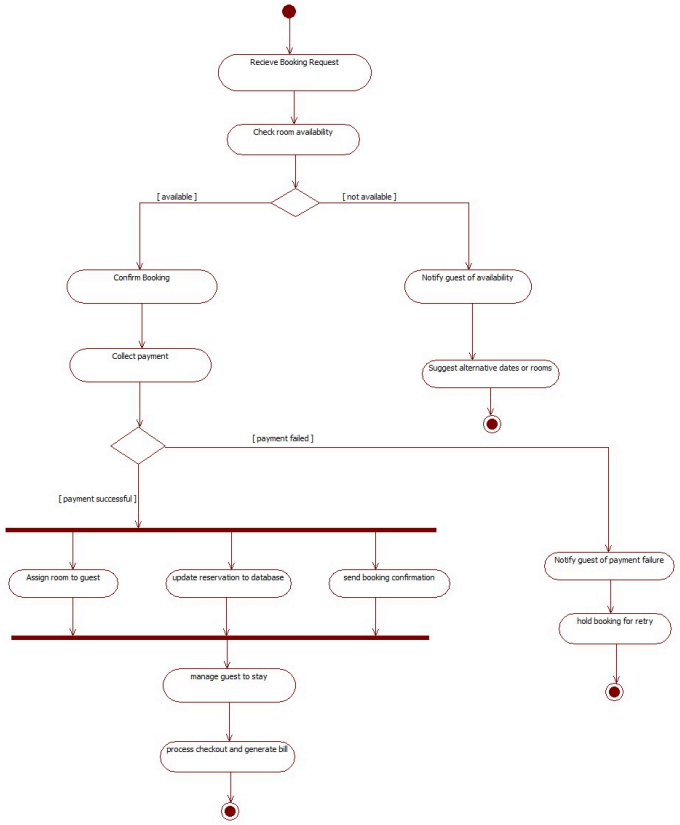


Fig 1.7.1 Simple Activity Diagram

This activity diagram maps the hotel booking process from receiving a booking request to managing guest stay and checkout. The workflow begins with a booking request and a room availability check. If a room is available, the booking is confirmed and payment is collected. Upon successful payment, the system assigns a room, updates the reservation database, sends a booking confirmation, and manages the guest's stay until checkout and bill generation. If at any point a room is unavailable or payment fails, the guest receives notifications and options such as retrying payment or selecting alternative rooms or dates.

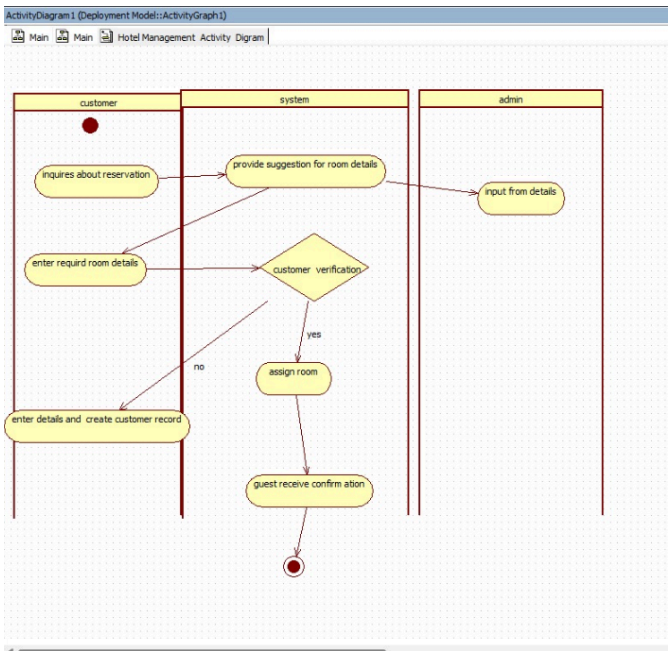


Fig 1.7.2 Advanced Activity Diagram

This activity diagram illustrates the hotel reservation workflow from inquiry to confirmation.

- The customer starts by inquiring about a reservation and providing required room details.
- The system suggests room options, verifies the customer's details, and (if verified) assigns a room.
- The admin gives input from the database as needed for reservation creation.
- If verification fails, the customer data is entered and a record is created; otherwise, the guest receives confirmation of the room assignment.

The diagram clearly shows responsibilities divided among the customer, system, and admin, and the main steps to complete a room booking.

2. Credit Card Processing System

2.1 Problem Statement

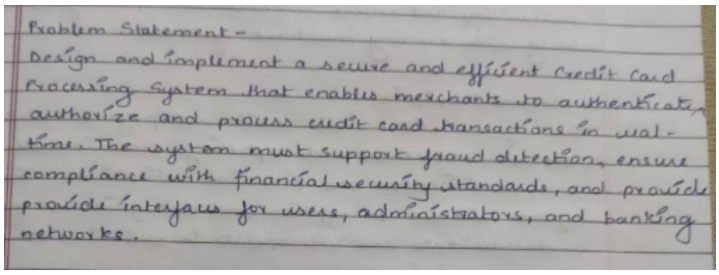
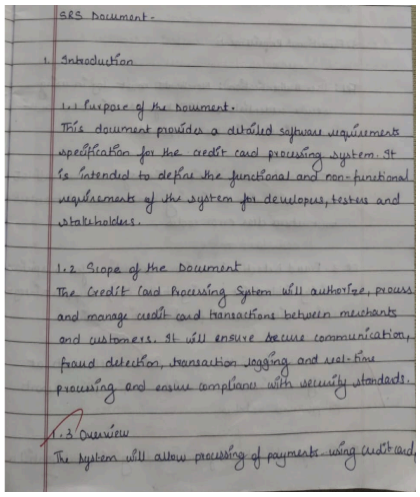

Problem Statement -
Design and implement a secure and efficient Credit Card Processing System that enables merchants to authenticate, authorize and process credit card transactions in real-time. The system must support fraud detection, ensure compliance with financial security standards, and provide provide interfaces for users, administrators, and banking networks.

Fig 2.1.1

2.2 SRS-Software Requirements Specification


SRS Document -

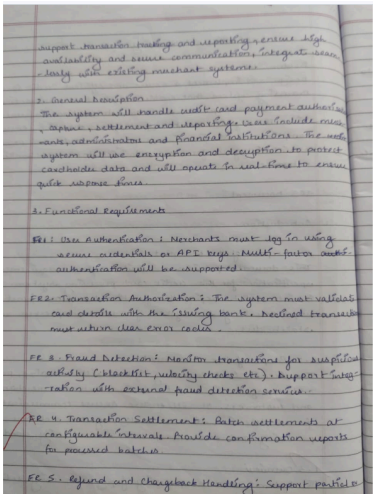
1. Introduction

1.1 Purpose of the Document.
This document provides a detailed software requirements specification for the credit card processing system. It is intended to define the functional and non-functional requirements of the system for developers, testers and stakeholders.

1.2 Scope of the Document
The Credit Card Processing System will authorize, process and manage credit card transactions between merchants and customers. It will ensure secure communication, fraud detection, transaction logging and real-time processing and ensure compliance with security standards.

1.3 Overview
The system will allow processing of payments using credit card.

Fig 2.2.1


support transaction routing and reporting, ensure high availability and secure communication, integrate seamlessly with existing merchant systems.

2. General Description
The system will handle credit card payment authorization, capture, settlement and reporting. Users include merchants, administrators and financial institutions. The web system will use encryption and decryption to protect cardholder data and will operate in real-time to ensure quick response times.

3. Functional Requirements

FR1: User Authentication: Merchants must log in using secure credentials or API keys. Multi-factor authentication will be supported.

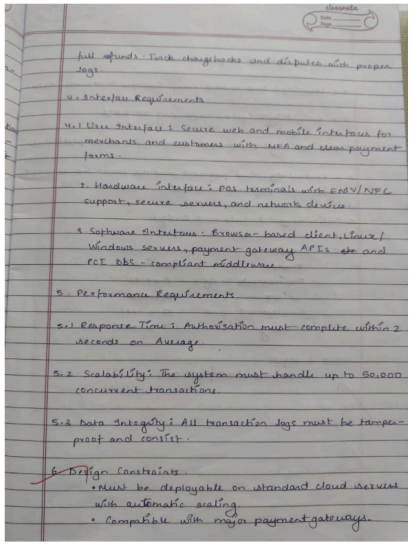
FR2: Transaction Authorization: The system must validate card details with the issuing bank. Declined transactions must return clear error codes.

FR 3: Fraud Detection: Monitor transactions for suspicious activity (blacklist, velocity checks etc.). Support integration with external fraud detection services.

FR 4: Transaction Settlement: Batch settlements at configurable intervals. Provide confirmation reports for processed batches.

FR 5: Refund and Chargeback Handling: Support partial a

Fig 2.2.2


full refunds. Track chargebacks and disputes with proper logs.

4. Interface Requirements

4.1 User Interface: Secure web and mobile interfaces for merchants and customers with PCI and user payment forms.

4.2 Hardware Interface: POS terminals with EMV/NFC support, secure servers, and network devices.

4.3 Software Interfaces: Browser-based client, Linux / Windows servers, payment gateway APIs etc and PCI DSS-compliant middleware.

5. Performance Requirements

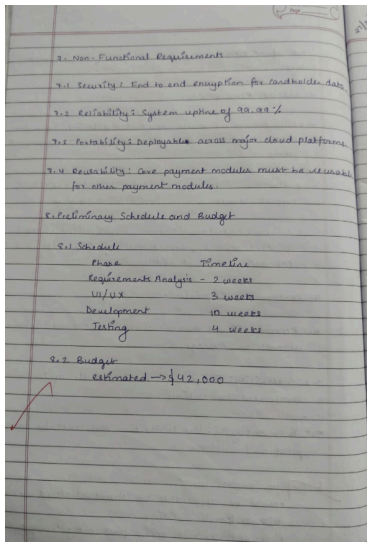
5.1 Response Time: Authorization must complete within 2 seconds on Average.

5.2 Scalability: The system must handle up to 50,000 concurrent transactions.

5.3 Data Integrity: All transaction logs must be tamper-proof and consist.

6. Design Constraints
- Must be deployable on standard cloud servers with automatic scaling
- Compatible with major payment gateways

Fig 2.2.3


7. Non-Functional Requirements

7.1 Security: End to end encryption for cardholder data

7.2 Reliability: System uptime of 99.99%

7.3 Portability: Deployable across major cloud platforms

7.4 Reusability: Core payment modules must be reusable for other payment modules.

8. Preliminary Schedule and Budget

8.1 Schedule

Phase	Timeline
Requirements Analysis	2 weeks
UI/UX	3 weeks
Development	10 weeks
Testing	4 weeks

8.2 Budget
estimated → $42,000

Fig 2.2.4

2.3 Class Diagram

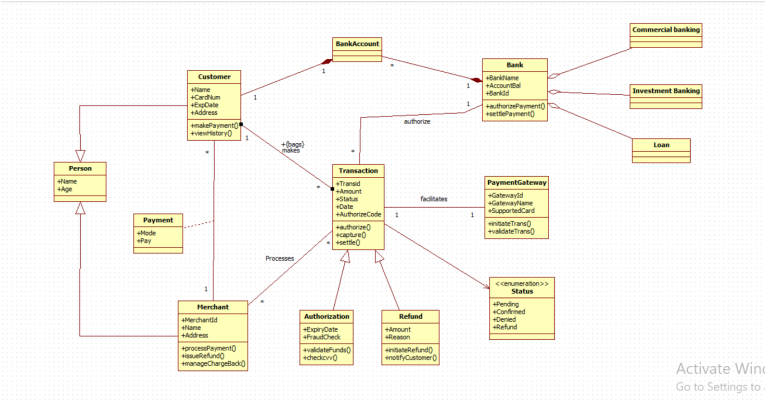


Fig 2.3.1 Credit Card Processing Class Diagram

This class diagram illustrates the main components and relationships in a banking payment system. Customers and merchants interact via payment transactions, which are processed through a payment gateway and authorized by banks. Bank accounts link customers to banks for handling funds. The system supports transaction validation, authorization, and refunds, with transaction status tracked from pending to confirmed or refunded. Specialized bank services like loans and investment banking are also depicted.

2.4 State Diagram

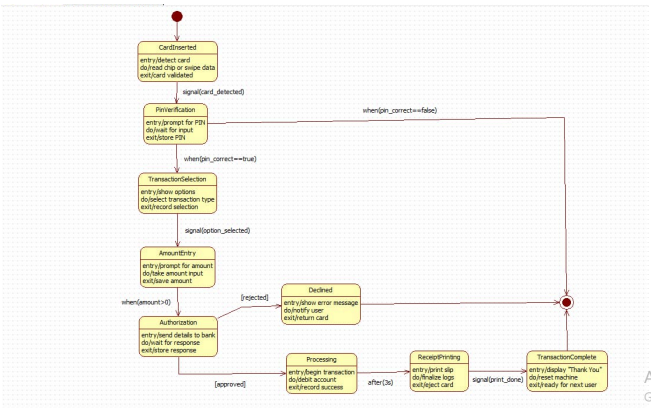


Fig 2.4.1 Simple State Diagram

This state machine diagram shows the flow of a card payment transaction in a banking system. The process starts with card details input and verification, then moves to authorization and payment processing. If all checks pass, the transaction completes successfully and the user

receives a notification or receipt. If any step fails, error handling such as invalid card or failed authorization is triggered, guiding the user accordingly.

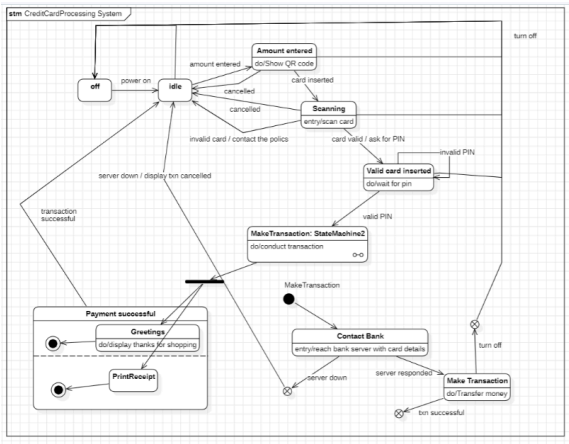


Fig 2.4.2 Advanced State Diagram

This state machine diagram describes the operation of a credit card processing system. The process begins with the system powering on and going idle, then accepts an entered amount, scans a card, and checks for a valid card and PIN. If authorized, the transaction is made and the bank is contacted. Upon successful payment, a greeting and receipt are displayed for the user. Any cancellation or error diverts the flow to transaction cancelled or server down states.

2.5 Use-Case Diagram

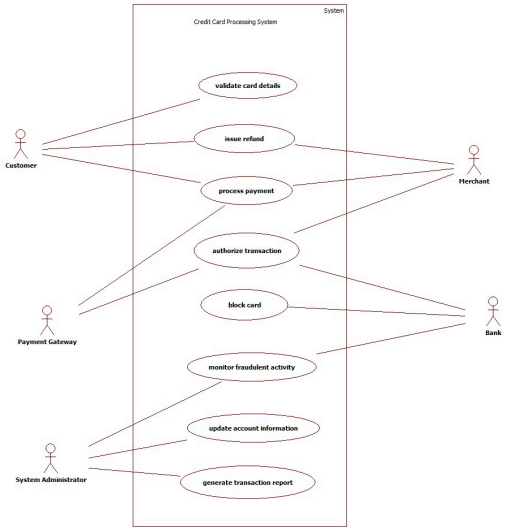


Fig 2.5.1 Simple Use Case Diagram

This use case diagram represents the credit card processing system. Customers can validate card details, process payments, and request refunds. Merchants interact for payments and refunds. Banks authorize transactions and handle card blocking. The payment gateway is responsible for processing payments and authorizations. The system administrator manages account updates, monitors for fraud, and generates transaction reports. Each actor is linked to specific actions that enable secure, efficient management of card transactions and account status.

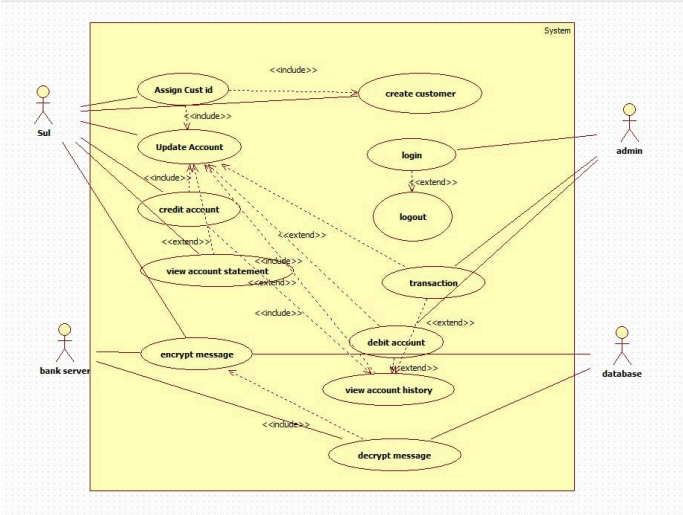


Fig 2.5.2 Advanced Use Case Diagram

This use-case diagram depicts the main functionalities of a banking software system. The system allows users (customers, bank servers, admins, and databases) to perform operations such as creating and updating accounts, viewing statements, logging in and out, managing transactions, and handling encryption/decryption of messages. Each actor interacts with relevant use cases, which show how account management and security tasks are coordinated for banking operations.

2.6 Sequence Diagram

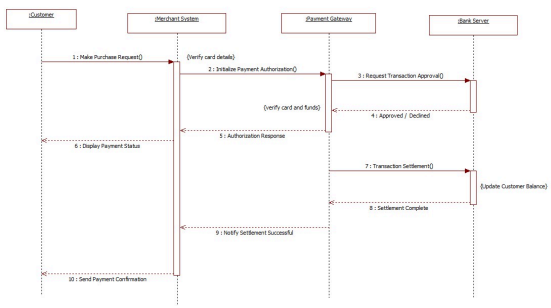


Fig 2.6.1 Simple Sequence Diagram

This simple sequence diagram shows how a customer makes a purchase using a credit card. The process begins when the customer sends a purchase request to the merchant system, which verifies the card and initiates payment authorization through a payment gateway. The bank server then approves or declines the transaction. If approved, the bank settles the payment and updates the customer's balance. The merchant receives settlement confirmation and displays the payment status to the customer, who also receives a payment confirmation message.

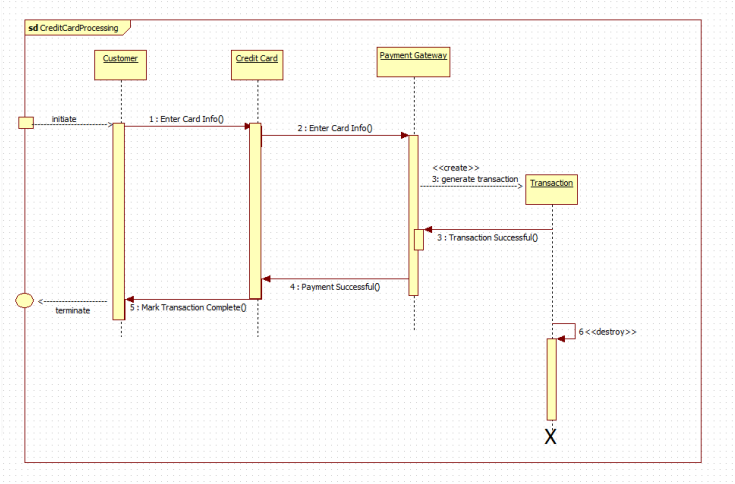


Fig 2.6.1 Advanced Sequence Diagram

This sequence diagram outlines the steps in a credit card payment transaction. The customer enters card information, which is passed from the Credit Card system to the Payment Gateway. The payment gateway generates a transaction and, after processing, returns a successful payment response. Finally, the transaction is marked as complete and the process terminates.

2.7 Activity Diagram

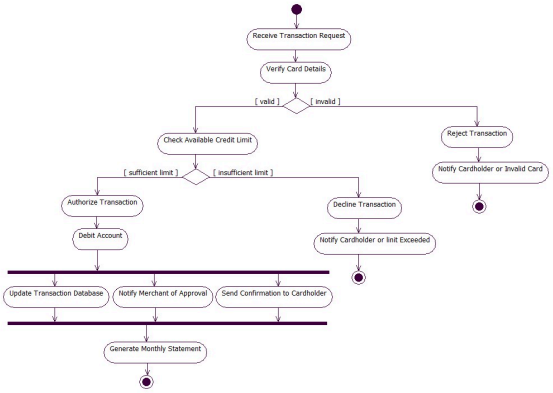


Fig 2.7.1 Simple Activity diagram

This activity diagram outlines the steps in processing a credit card transaction. The process begins with receiving the transaction request and verifying card details. If the card is valid, the system checks the available credit limit. With a sufficient limit, the transaction is authorized, the account is debited, and notifications are sent to the merchant and cardholder. The database is updated and a monthly statement is generated. If the card is invalid or the limit is insufficient, the transaction is declined and the cardholder is notified.

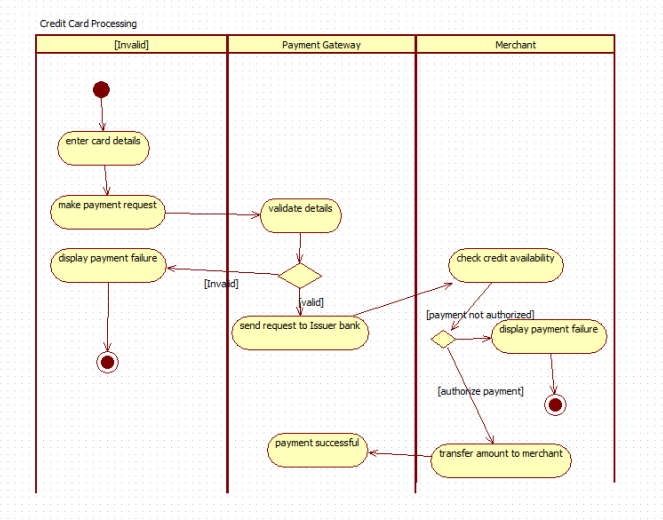


Fig 2.7.2 Advanced Activity diagram

This activity diagram shows the steps in credit card payment processing. The customer enters card details, which are validated by the payment gateway. If valid, a request goes to the merchant to check credit availability. Payment failures can occur if validation or authorization fails. If payment succeeds, the amount is transferred to the merchant, successfully completing the transaction.

3. Library Management System

3.1 Problem Statement

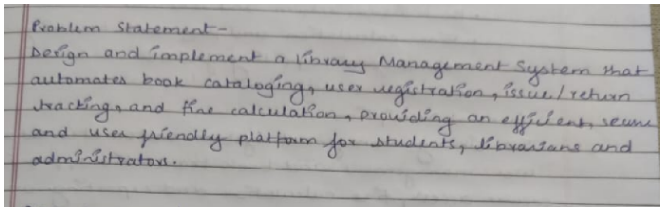
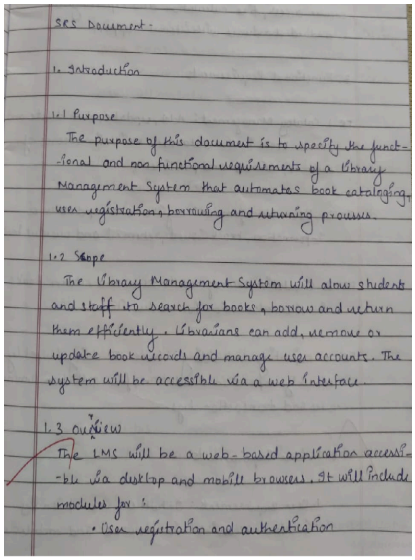

Problem Statement-
Design and implement a library Management System that automates book cataloging, user registration, issue/return tracking, and fine calculation, providing an efficient, secure and user friendly platform for students, librarians and administrators.

Fig 3.1.1

3.2 SRS-Software Requirements Specification


SRS Document-

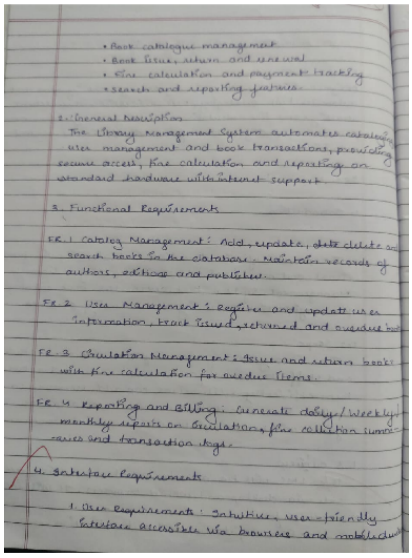
1. Introduction

1.1 Purpose
The purpose of this document is to specify the functional and non functional requirements of a library Management System that automates book cataloging, user registration, borrowing and returning processes.

1.2 Scope
The library Management System will allow students and staff to search for books, borrow and return them efficiently. Librarians can add, remove or update book records and manage user accounts. The system will be accessible via a web interface.

1.3 Overview
The LMS will be a web-based application accessible via desktop and mobile browsers. It will include modules for:
- User registration and authentication

Fig 3.2.1


- Book catalogue management
- Book issue, return and renewal
- Fine calculation and payment tracking
- search and reporting features

2. General Description
The library Management System automates cataloging, user management and book transactions, providing secure access, fine calculation and reporting on standard hardware with internet support.

3. Functional Requirements

FR 1 Catalog Management: Add, update, delete or search books in the database. Maintain records of authors, editions and publishers.

FR 2 User Management: Register and update user information, track issued/returned and overdue books.

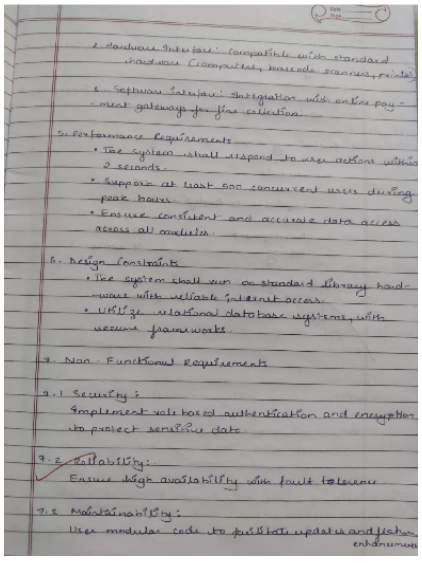
FR 3 Circulation Management: Issue and return books with fine calculation for overdue items.

FR 4 Reporting and Billing: Generate daily/weekly/monthly reports on circulation, fine collection summaries and transaction logs.

4. Interface Requirements

4.1 User Requirements: Intuitive, user-friendly interfaces accessible via browsers and mobile devices

Fig 3.2.2


4.2 Hardware Interface: Compatible with standard hardware components, barcode scanners, printers

4.3 Software Interface: Integration with online payment gateways for fine collection.

5. Performance Requirements
- The system shall respond to user actions within 2 seconds.
- Support at least 500 concurrent users during peak hours.
- Ensure consistent and accurate data access across all modules.

6. Design Constraints
- The system shall run on standard library hardware with reliable internet access.
- Utilize relational database systems with secure frameworks.

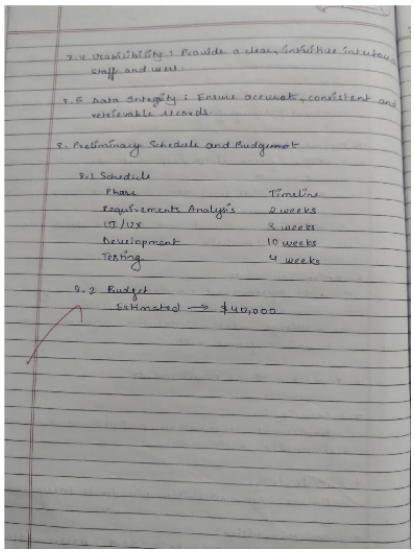
7. Non Functional Requirements

7.1 Security:
Implement role based authentication and encryption to protect sensitive data.

7.2 Reliability:
Ensure high availability with fault tolerance.

7.3 Maintainability:
User modular code to facilitate updates and feature enhancements.

Fig 3.2.3


7.4 Usability: Provide a clear, intuitive interface for staff and users.

7.5 Data Integrity: Ensure accurate, consistent and retrievable records.

8. Preliminary Schedule and Budgemet

8.1 Schedule

Phase	Timeline
Requirements Analysis	2 weeks
UI/UX	3 weeks
Development	10 weeks
Testing	4 weeks

8.2 Budget
Estimated → $40,000

Fig 3.2.4

3.3 Class Diagram

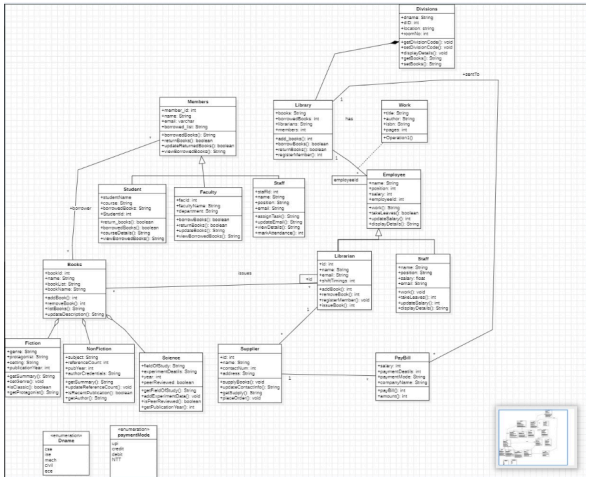


Fig 3.3.1 Class diagram

This is a class diagram for a library management system. It includes classes for books, members, staff, librarians, departments, courses, students, and records such as issues, reservations, and fines. Relationships connect how books are managed, issued, reserved, and returned; staff oversee operations; members borrow and reserve books; and records track all library transactions

3.4 State Diagram

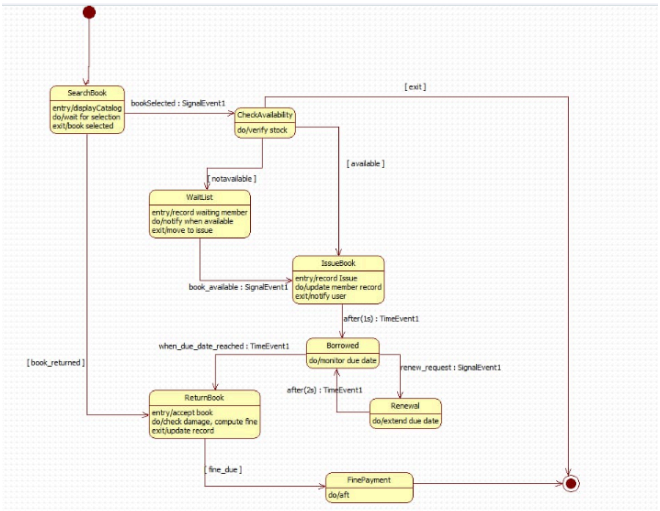


Fig 3.4.1Simple State Diagram

This state machine diagram shows the book borrowing process in a library management system. It starts with searching for a book and checking its availability. If the book is available, it is

issued to the member and enters the "Borrowed" state. When the due date approaches, a reminder is sent. If the book is returned, fines are computed if overdue, and the transaction is completed.

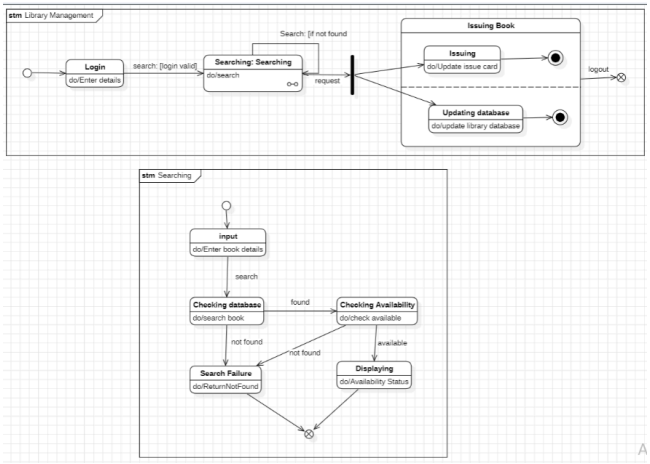


Fig 3.4.2 Advanced State Diagram

This advanced state diagram illustrates how a library management system handles book searches and issuing. Users log in and search for books; if the book is found, the system moves to issuing and updates the database. If the book is not found, it displays a search failure. The lower diagram details the searching logic, including input, database checks, and displaying availability or failure results.

3.5 Use-Case Diagram

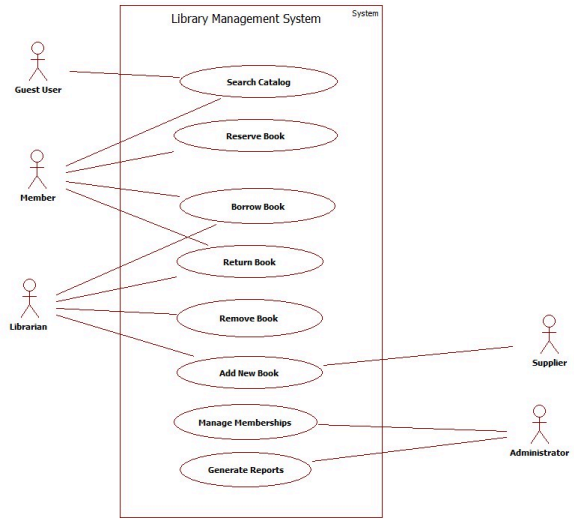


Fig 3.5.1 Simple Use-Case Diagram

This use-case diagram for a library management system shows how different users interact with the system. Guest users can search the catalog, members can reserve, borrow, and return books,

while librarians manage the collection by adding or removing books and handle memberships. Administrators oversee membership and report generation, and suppliers are involved when new books are added. Each actor is connected to specific system functionalities relevant to their role, enabling efficient library operations.

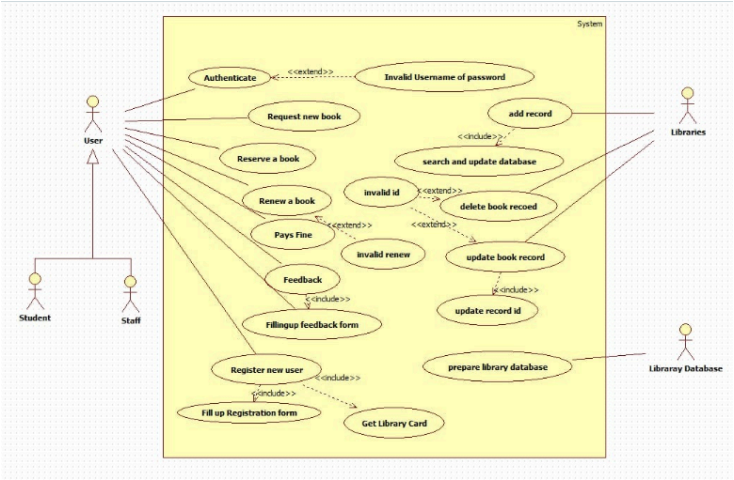


Fig 3.5.2 Advanced Use-Case Diagram

This use-case diagram shows the main actions available to different users in a library management system. Users—including students, staff, librarians, and the library database—can authenticate, search for books, reserve or renew, pay fines, and register for new accounts. Librarians maintain records and update the database, while the system ensures proper authentication and handles operations like feedback, registration, and card issuance.

3.6 Sequence Diagram

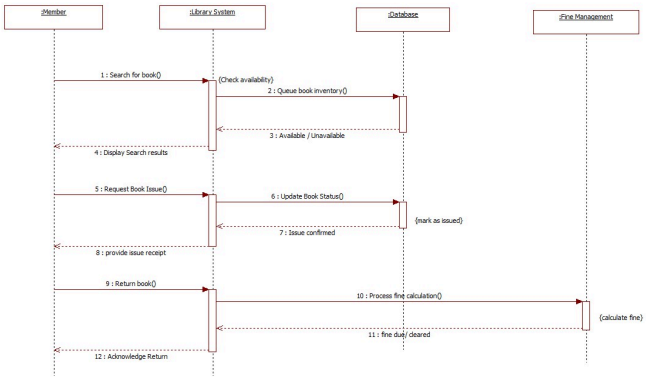


Fig 3.6.1 Simple Sequence Diagram

This simple sequence diagram shows how a library member searches for a book, checks its availability, and requests to borrow it. The library system updates the database and confirms the loan if the book is available. After reading, the member returns the book, triggering fine calculation if overdue. The process involves interactions between the member, system, database, and fine management to complete the transaction smoothly.

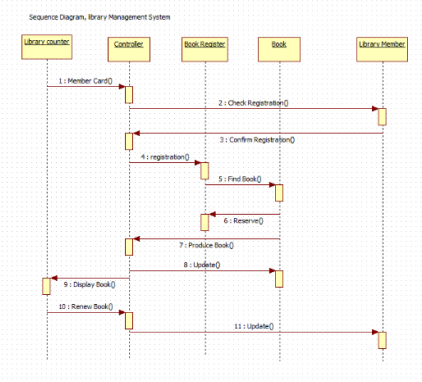


Fig 3.6.2 Advanced Sequence Diagram

This sequence diagram shows the steps a library member takes to register and reserve a book in the library management system. The process begins with presenting a member card, checking registration, and confirming it. The member then finds a book, requests reservation, and the book is produced by the system. Finally, the book register and member records are updated to reflect the transaction.

3.7 Activity Diagram

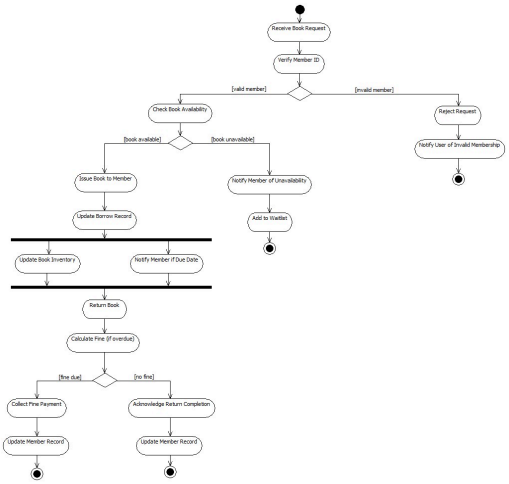


Fig 3.7.1 Simple Activity Diagram

This activity diagram illustrates the process for borrowing and returning a book in a library management system. It starts with a member's book request and membership verification. If the member is valid and the book is available, the book is issued, inventory updated, and return due date notified. For unavailable books, the member is notified and can be added to a waitlist. Upon returning the book, the system checks for overdue fines, collects payment if needed, updates records, and acknowledges completion. Invalid members have their requests rejected and are notified of their status.

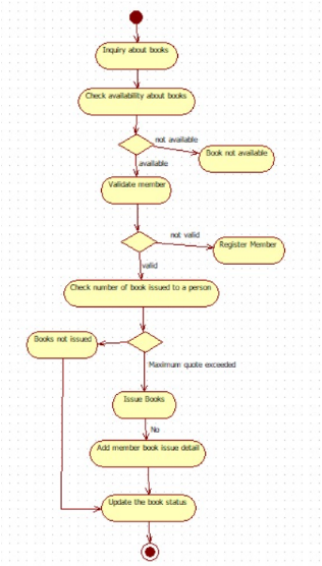


Fig 3.7.2 Advanced Activity Diagram

This activity diagram outlines the steps involved in issuing a book in a library system. The process begins with inquiring and checking book availability. If available, the member is validated and their quota is checked. If valid and within quota, the book is issued and records are updated; otherwise, errors or registration steps are triggered. The diagram visually maps each decision and action for controlled book issuance.

4. Stock Maintenance System

4.1 Problem Statement

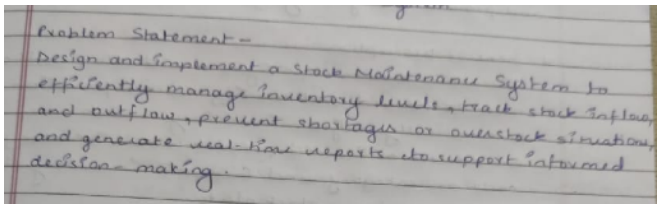
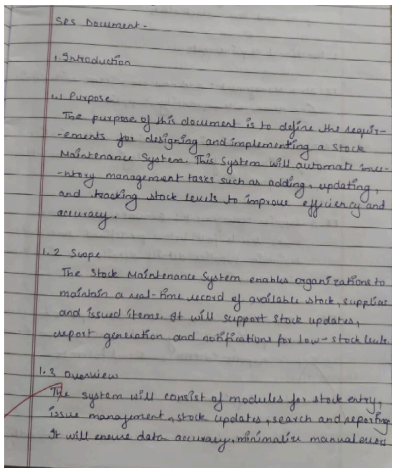

Problem Statement -
Design and implement a Stock Maintenance System to efficiently manage inventory levels, track stock inflow and outflow, prevent shortages or overstock situations, and generate real-time reports to support informed decision-making.

Fig 4.1.1

4.2 SRS-Software Requirements Specification


SRS Document -

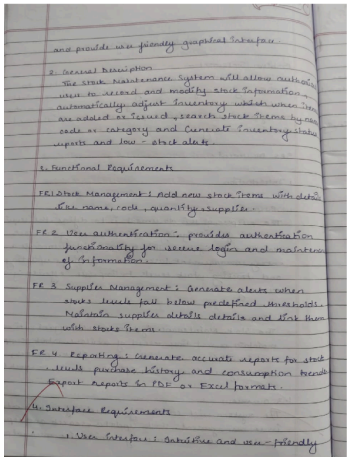
1. Introduction

1.1 Purpose
The purpose of this document is to define the requirements for designing and implementing a Stock Maintenance System. This System will automate inventory management tasks such as adding, updating, and tracking stock levels to improve efficiency and accuracy.

1.2 Scope
The Stock Maintenance System enables organizations to maintain a real-time record of available stock, supplier and issued items. It will support stock updates, report generation and notifications for low-stock levels.

1.3 Overview
The system will consist of modules for stock entry, issue management, stock updates, search and reporting. It will ensure data accuracy, minimize manual work

Fig 4.2.1


and provide user friendly graphical interface.

2. General Description
The Stock Maintenance System will allow authorized users to record and modify stock information automatically adjust inventory when items are added or issued, search stock items by name, code or category and generate inventory status reports and low-stock alerts.

3. Functional Requirements

FR1 Stock Management: Add new stock items with details like name, code, quantity, supplier.

FR 2 User authentication: provides authentication functionality for secure login and maintenance of information.

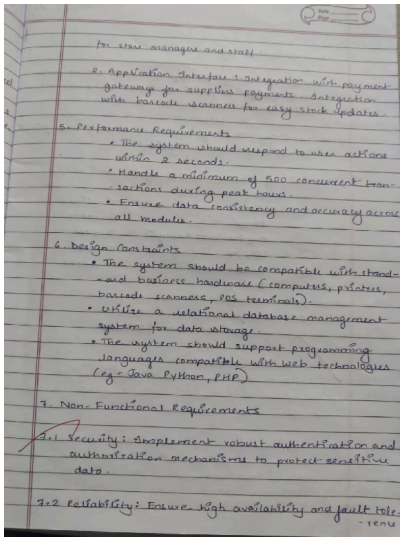
FR 3 Supplier Management: Generate alerts when stock levels fall below predefined thresholds. Maintain supplier details details and link them with stocks items.

FR 4 Reporting: Generate accurate reports for stock levels purchase history and consumption trends. Export reports to PDF or Excel formats.

4. Interface Requirements

1. User Interface: Intuitive and user-friendly

Fig 4.2.2


for store managers and staff.

2. Application Interface: Integration with payment gateways for supplier payments. Integration with barcode scanners for easy stock updates.

5. Performance Requirements
- The system should respond to user actions within 2 seconds.
- Handle a minimum of 500 concurrent transactions during peak hours.
- Ensure data consistency and accuracy across all modules.

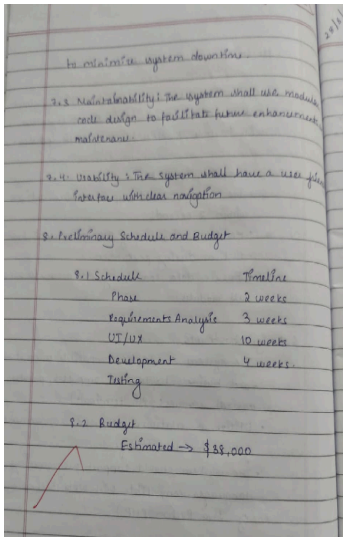
6. Design Constraints
- The system should be compatible with standard business hardware (computers, printers, barcode scanners, POS terminals).
- Utilize a relational database management system for data storage.
- The system should support programming languages compatible with web technologies (eg: Java, Python, PHP)

7. Non-Functional Requirements

7.1 Security: Implement robust authentication and authorization mechanisms to protect sensitive data.

7.2 Reliability: Ensure high availability and fault tolerance

Fig 4.2.3


to minimize system downtime.

7.3 Maintainability: The system shall use modular code design to facilitate future enhancements and maintenance.

7.4 Usability: The system shall have a user friendly interface with clear navigation

8. Preliminary Schedule and Budget

8.1 Schedule

Phase	Timeline
Requirements Analysis	2 weeks
UI/UX	3 weeks
Development	10 weeks
Testing	4 weeks.

8.2 Budget
Estimated -> $38,000

Fig 4.2.4

4.3 Class Diagram

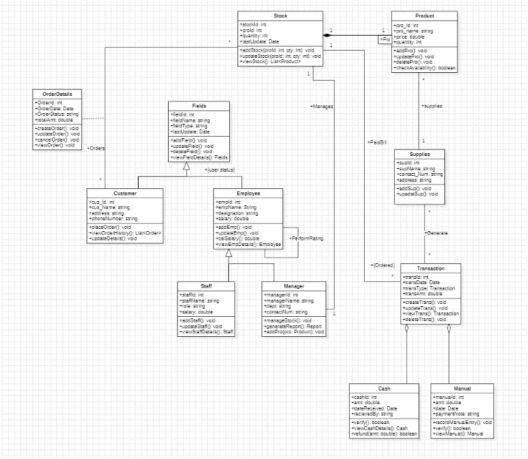


Fig 4.3.1 Class Diagram

This class diagram models a library management system, showing the relationships between key classes like Book, Customer, Staff, Manager, Transaction, Database, and Payment. Books are managed in the database and linked to transactions, which record issues and returns. Customers, staff, and managers interact with books and transactions, while payment can be processed in cash or card. Attributes and methods for each class capture essential data and operations for managing library resources.

4.4 State Diagram

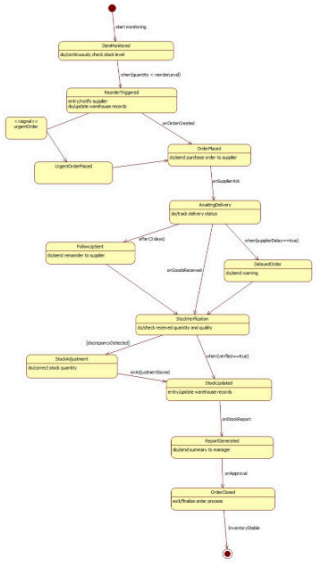


Fig 4.4.1 Simple State Diagram

This state diagram shows the different stages a book goes through in a library system. It starts from book acquisition/registration and moves through validation, issue, and renewal states.

Intermediate states include checking member status, updating records, handling renewals, and resolving returns or overdue issues. The transitions ensure that book management is organized and follows the library's rules at each step.

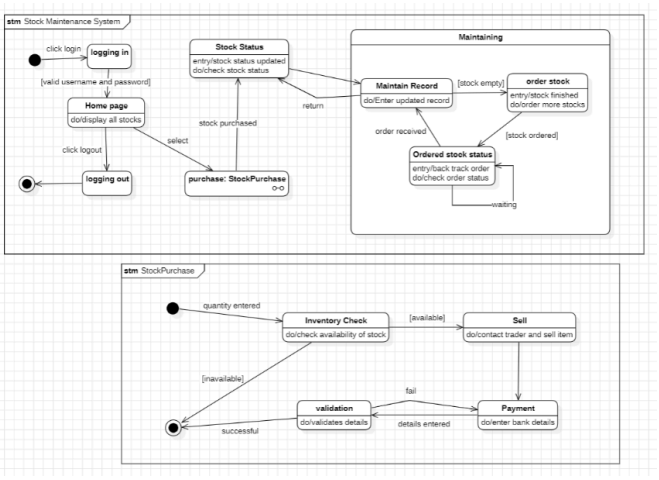


Fig 4.4.2 Advanced State Diagram

This advanced state diagram models the flow of actions in a stock maintenance system. It starts with user login, then allows navigation between home and maintenance pages for viewing and updating stock records, or ordering new stock. The lower diagram details the stock purchase process, including inventory checks, validation, selling, and payment. Actions and transitions ensure stock is correctly managed, maintained, and tracked in the system.

4.5 Use-Case Diagram

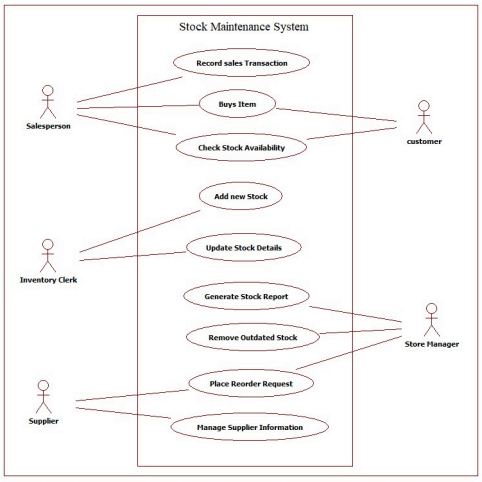


Fig 4.5.2 Simple Use-Case Diagram

This use-case diagram for a stock maintenance system shows how different roles interact with inventory processes. Salespersons and customers are involved in buying items, checking stock, and recording sales. Inventory clerks manage stock by adding new items, updating details, and removing outdated stock. Suppliers handle reorder requests and manage their information. Store managers oversee stock reports and control critical inventory actions. The diagram clearly outlines responsibilities for efficient stock tracking and replenishment.

Fig 4.5.2 Advanced Use-Case Diagram

This use-case diagram illustrates stock management functions for a system used by admins and traders. Key features include logging in, validating credentials, viewing and purchasing stock, checking availability, making payments, generating reports, handling returns, and reviewing order requests. The diagram maps out how both user types interact with core operations—like inventory check, updates, and order processes—within the stock management workflow.

4.6 Sequence Diagram

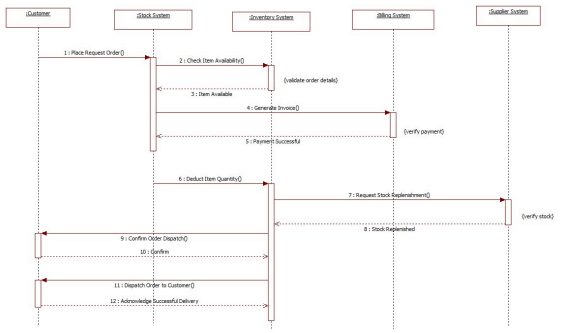


Fig 4.6.1 Simple Sequence Diagram

This sequence diagram shows the process for a customer order in a stock maintenance system. The customer places an order, prompting the system to check availability and generate an invoice. After payment is verified, the ordered quantity is deducted. If stock is insufficient, a reorder request is sent to the supplier. Once the order is dispatched, both order success and delivery are acknowledged, completing the transaction cycle.

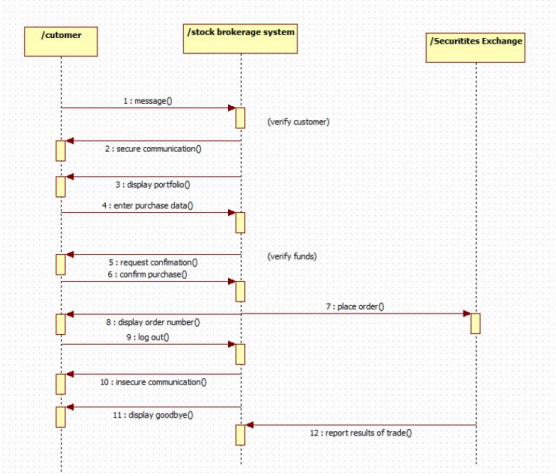


Fig 4.6.2 Advanced Sequence Diagram

This sequence diagram explains how a customer interacts with a stock brokerage system and a securities exchange to place and confirm a trade. The customer securely logs in, views their portfolio, and enters a purchase order. The brokerage system verifies the customer and funds, then places the order with the exchange. After confirmation, the system displays the order details, logs out the customer, and reports the result of the trade.

4.7 Activity Diagram

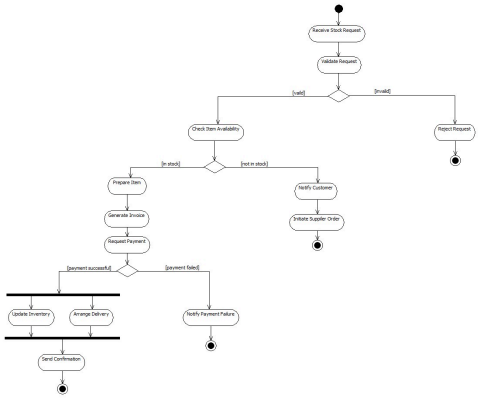


Fig 4.7.1 Simple Activity Diagram

This activity diagram outlines the stock request process in a stock maintenance system. When a stock request is received, it is validated. If valid, the system checks item availability. For in-stock items, the system prepares the item, generates an invoice, and requests payment. On successful payment, inventory is updated, delivery is arranged, and confirmation is sent. If payment fails or the item is out of stock, the customer is notified and a supplier order is initiated if needed. Invalid requests are rejected, ensuring only proper transactions proceed.

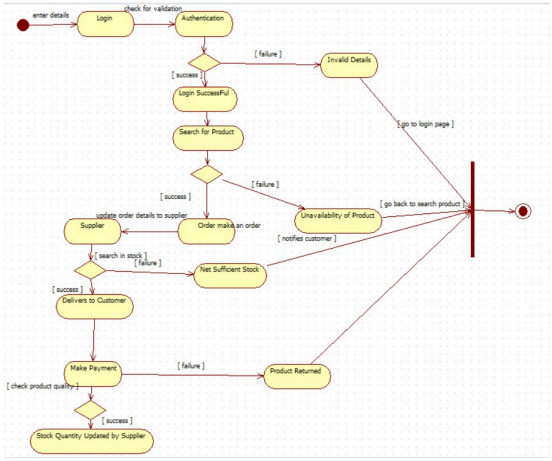


Fig 4.7.2 Advanced Activity Diagram

This activity diagram shows a customer ordering a product online. After login and authentication, the customer searches for a product and, if available and in stock, places an order. Payment is made, stock is updated, and returns are handled if needed. Unavailable products or invalid logins end the process early.

5. Passport Automation System

5.1 Problem Statement

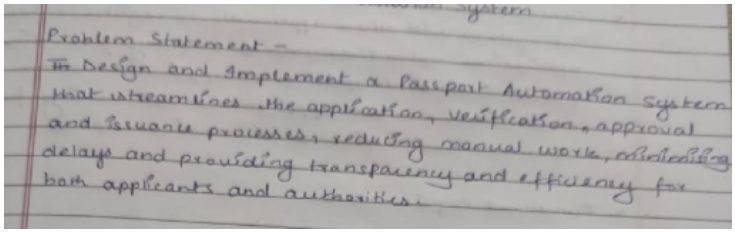


Fig 5.1.1

5.2 SRS-Software Requirements Specification

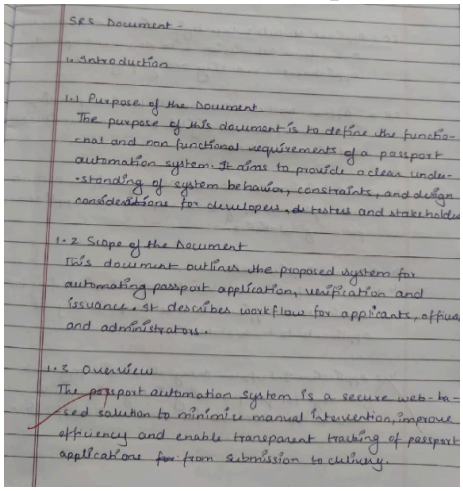


Fig 5.2.1

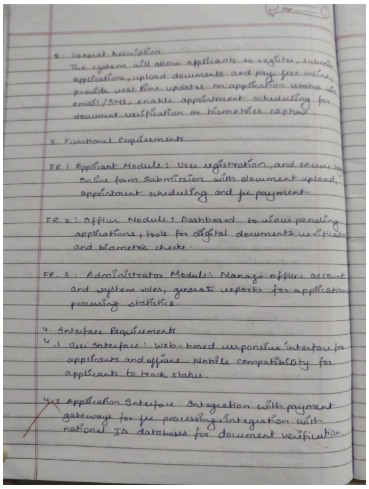


Fig 5.2.2

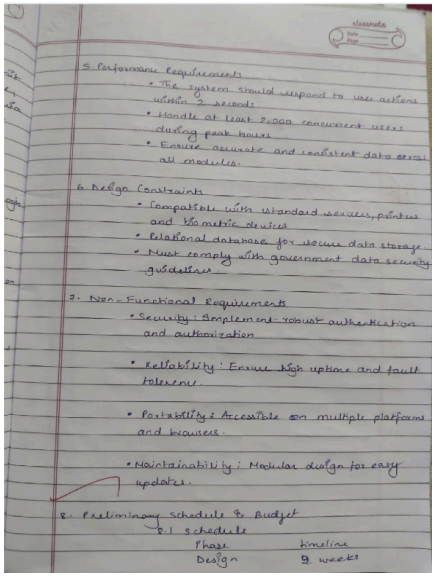


Fig 5.2.3

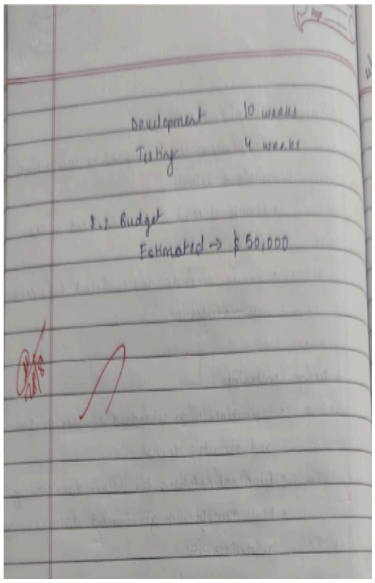


Fig 5.2.4

5.3 Class Diagram

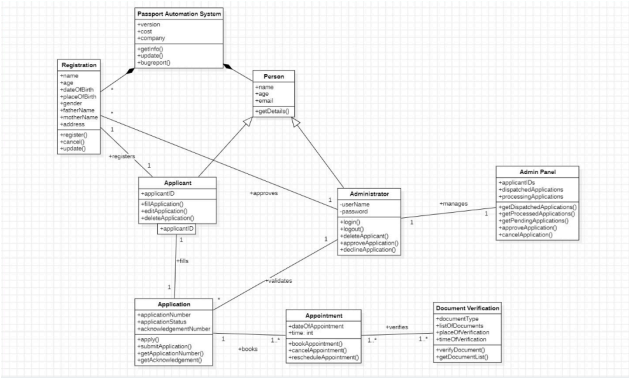


Fig 5.3.1 Class Diagram

This class diagram models a passport automation system, showing how registration, applicants, appointments, and document verification connect. The system consists of classes for Person, Applicant, Administrator, Appointment, Registration, Admin Panel, and Document Verification. Relationships capture how users register, book appointments, interact with administrators, and verify documents as part of the passport process.

5.4 State Diagram

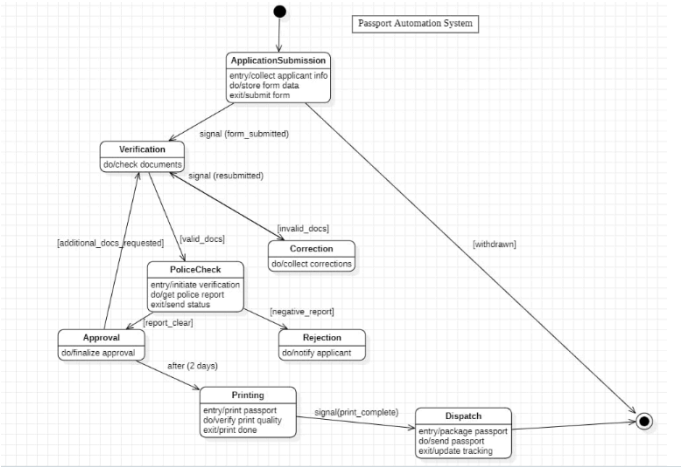


Fig 5.4.1 Simple State Diagram

This state diagram maps the passport application process in an automation system. It begins with applicant submission and document verification. If documents are missing, corrections are requested; valid applications proceed to police check and approval. Approved passports are printed and dispatched, while rejections or withdrawals end the process. Each state and transition records key checks and actions for secure passport processing.

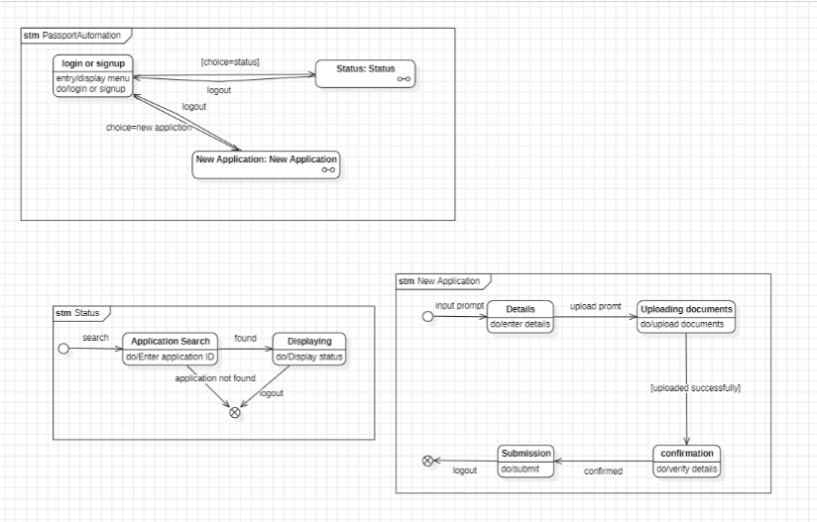


Fig 5.4.2 Advanced State Diagram

This diagram shows the passport automation system's state flow for user login, application status search, and new application submission. Users log in to check application status or submit new applications. The search checks for the entered ID and displays the result; the application state flows from entering details to uploading documents and confirming submission. Logout transitions can occur from any state.

5.5 Use-Case Diagram

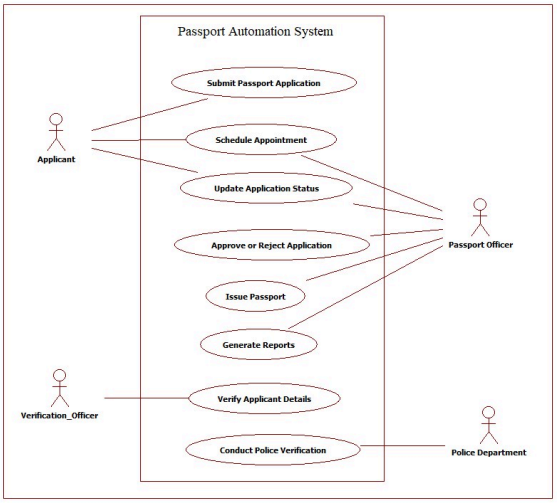


Fig 5.5.1 Simple Use-Case Diagram

This use-case diagram illustrates the main actions in a passport automation system. Applicants can submit passport applications, schedule appointments, and check their application status. Passport officers are responsible for approving or rejecting applications, issuing passports,

updating statuses, and generating reports. Verification officers handle the verification of applicant details, while the police department is involved in conducting police verification. The diagram shows how each actor interacts with the system to ensure a smooth passport processing workflow.

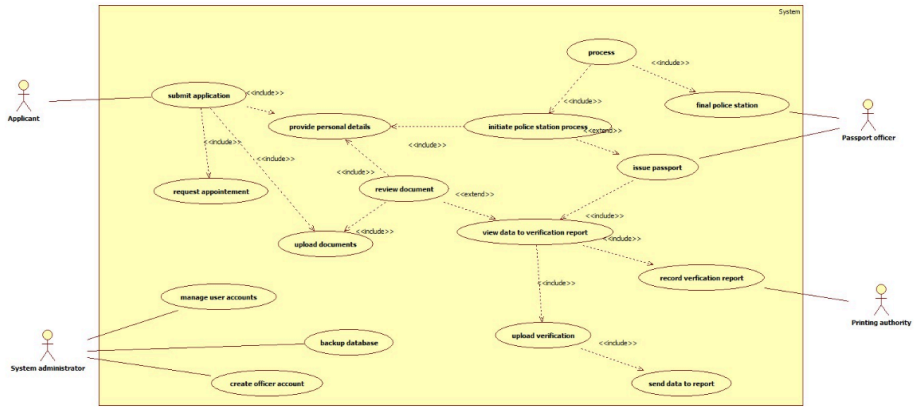


Fig 5.5.2 Advanced Use-Case Diagram

This use-case diagram models a passport automation system with actors such as applicants, system administrators, passport officers, and printing authorities. Applicants can submit applications, provide details, request appointments, review and upload documents, and view reports. The system coordinates processes like verification, police station procedures, issuing passports, and managing accounts. It also enables interaction between authorities for report recording and passport printing.

5.6 Sequence Diagram

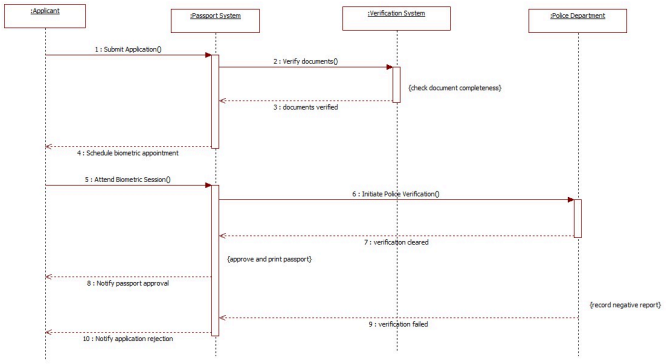


Fig 5.6.1 Simple Sequence Diagram

This sequence diagram shows the steps involved in processing a passport application. The applicant submits an application, which is verified for document validity. If the documents are verified, a biometric session is scheduled and police verification is initiated. After receiving the police verification result, the passport officer approves and prints the passport if successful, or rejects the application if issues are found. Throughout the process, the applicant is notified of the application's status.

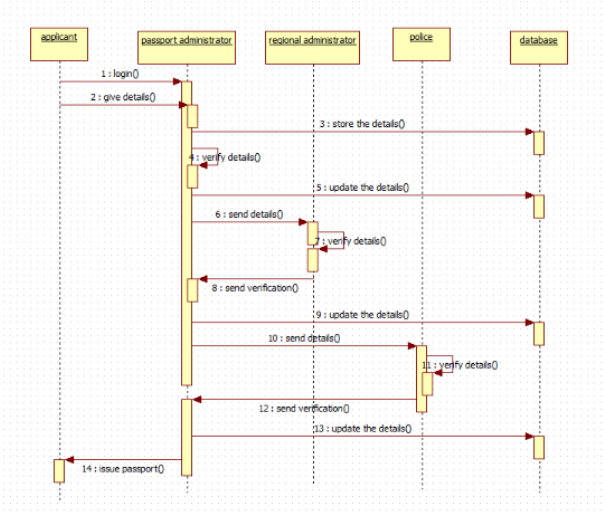


Fig 5.6.2 Advanced Sequence Diagram

This sequence diagram shows the steps in processing a passport application. The applicant logs in and submits details, which the passport administrator verifies and forwards to the regional administrator, police, and database. Verification steps are repeated at each level, and once approved, the database is updated and the passport is issued to the applicant.

5.7 Activity Diagram

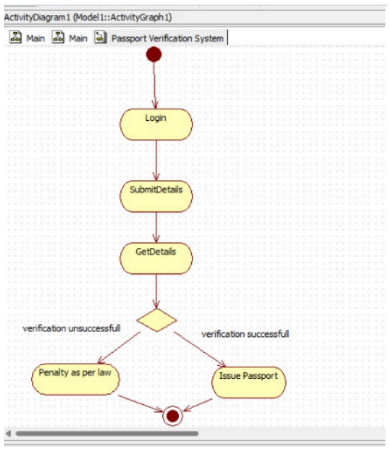


Fig 5.7.1 Simple Activity Diagram

This activity diagram shows the steps in a passport verification system. The user logs in, submits their details, and the system collects verification information. If verification is successful, a passport is issued; if unsuccessful, a penalty is applied as per law. The process clearly splits based on verification outcome, ending with either issuing or penalizing.

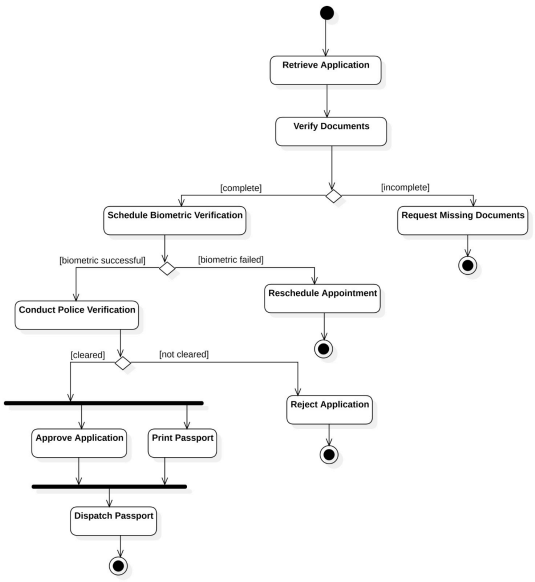


Fig 5.7.2 Advanced Activity Diagram

This advanced activity diagram outlines the steps for processing a passport application. The system starts by retrieving the application and verifying documents. If documents are complete, a biometric verification is scheduled; otherwise, missing documents are requested. A successful biometric check leads to police verification, after which the application is either approved and the passport printed and dispatched, or rejected if not cleared. Failed biometric checks or insufficient documents result in rescheduling or stopping the process.