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:

Hitesh Sharma
(1BM23CS114)

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 Hitesh Sharma (1BM23CS114) during the 5th Semester August 2025-December 2025

.

Signature of the Faculty Incharge:

NAME OF THE Your Batch Incharge and designation:

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

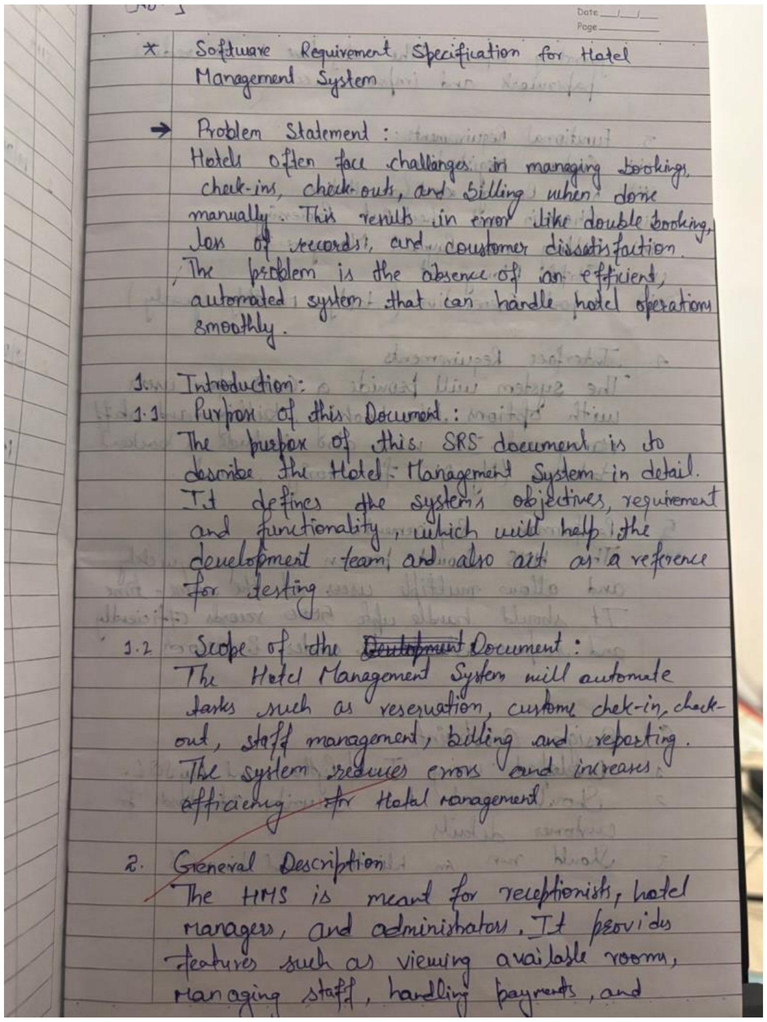
Table of Contents

1. Hotel Management System

Problem Statement – Hotel Management System

A Hotel Management System is required to automate and streamline these operations by providing a centralized platform where administrators, receptionists, and customers can manage reservations, room allocation, payments, and service requests seamlessly. The system must maintain complete guest records, ensure accurate billing, track room status dynamically, support multiple user roles, and generate essential reports. It should enhance operational efficiency, reduce human errors, improve customer experience, and ensure smooth functioning of the hotel's day-to-day activities.

SRS – Software Requirements Specification (Short Version)



Date __/__/___
Page ______

* Software Requirement Specification for Hotel Management System

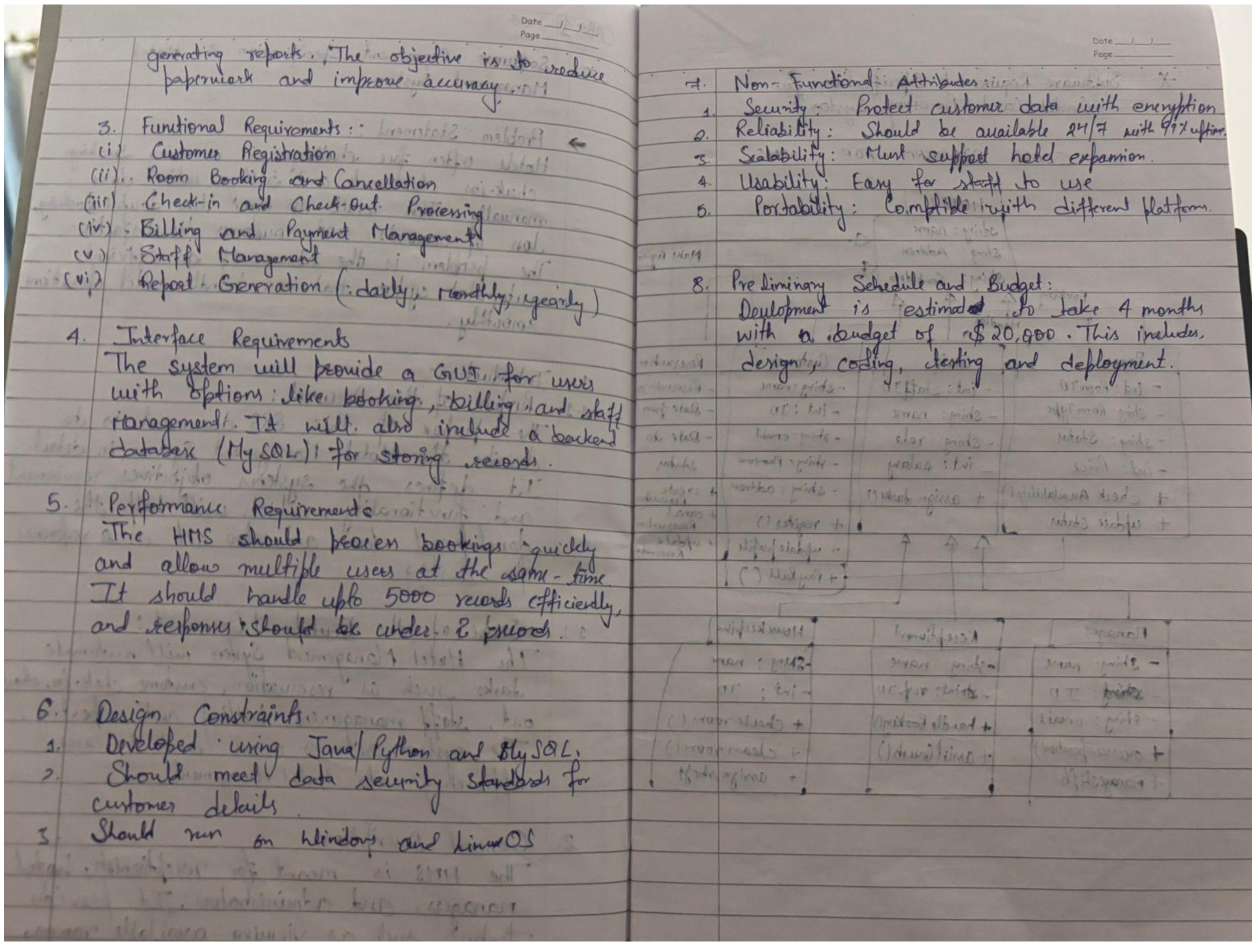
➔ Problem Statement :
Hotels often face challanges in managing bookings, check-ins, check-outs, and billing when done manually. This results in error like double booking, loss of records, and coustomer dissatisfaction. The problem is the absence of an efficient, automated system that can handle hotel operations smoothly.

1. Introduction:

1.1 Purpose of this Document :
The purpose of this SRS document is to describe the Hotel Management System in detail. It defines the system's objectives, requirement and functionality, which will help the development team and also act as a reference for testing.

1.2 Scope of the ~~Development~~ Document :
The Hotel Management System will automate tasks such as reservation, custome chek-in, check-out, staff management, billing and reporting. The system reduces errors and increases afficiency for Hotel management.

2. General Description
The HMS is meant for receptionists, hotel managers, and administrators. It provides features such as viewing available rooms, managing staff, handling payments, and



Date __/__/__
Page ____

generating reports. The objective is to reduce paperwork and improve accuracy.

3. Functional Requirements :
(i) Customer Registration.
(ii) Room Booking and Cancellation
(iii) Check-in and Check-Out Processing
(iv) Billing and Payment Management
(v) Staff Management
(vi) Report Generation (daily, monthly, yearly)

4. Interface Requirements
The system will provide a GUI for users with options like booking, billing and staff management. It will also include a backend database (My SQL) for storing records.

5. Performance Requirements
The HMS should process bookings quickly and allow multiple users at the same-time. It should handle upto 5000 records efficiently, and responses should be under 2 seconds.

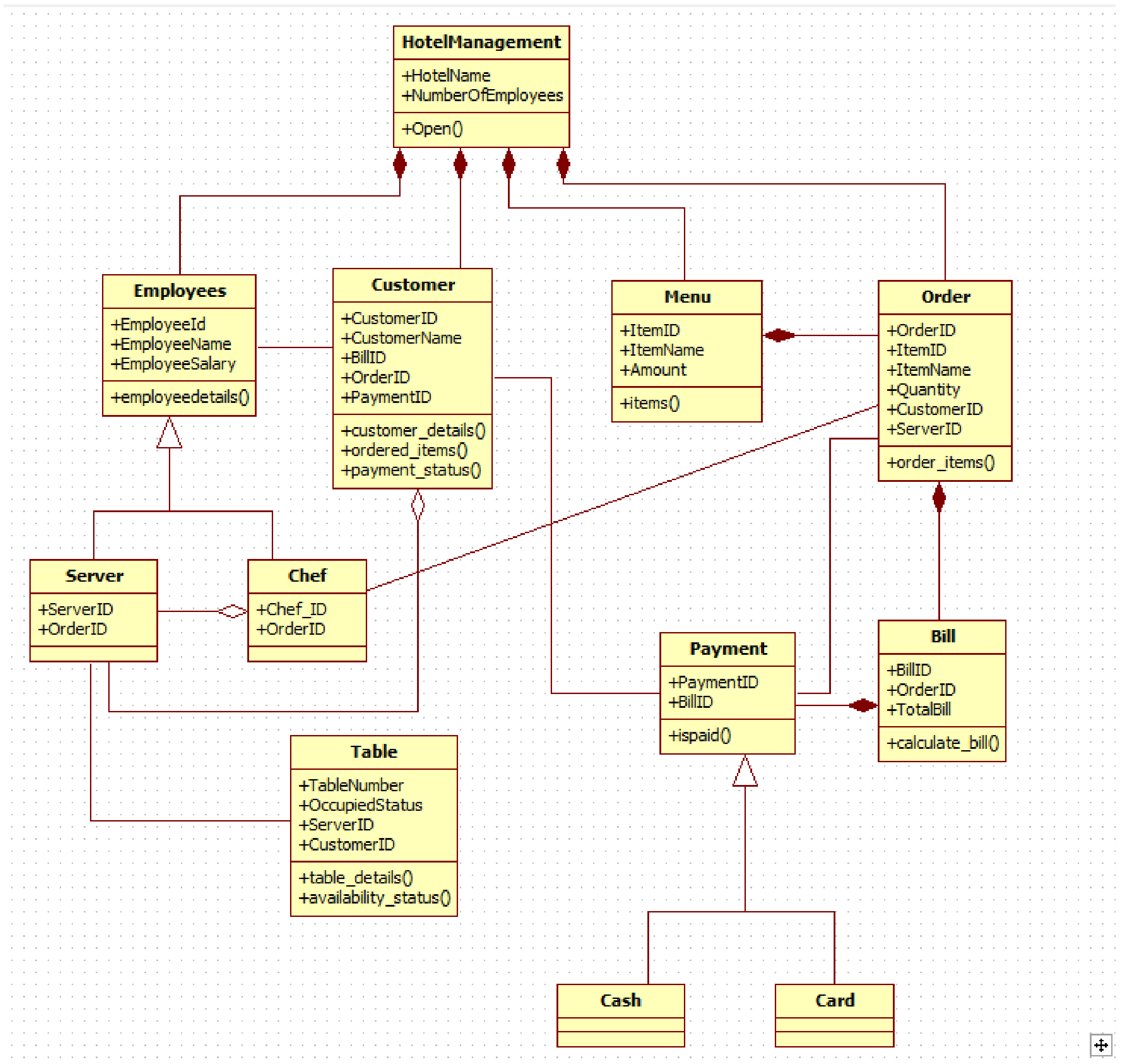
6. Design Constraints
1. Developed using Java/Python and MySQL.
2. Should meet data security standards for customer details.
3. Should run on Windows and LinuxOS

Date __/__/__
Page ____

7. Non-Functional Attributes
1. Security: Protect customer data with encryption
2. Reliability: Should be available 24/7 with 99% uptime.
3. Scalability: Must support hotel expansion.
4. Usability: Easy for staff to use
5. Portability: Compatible with different platforms.

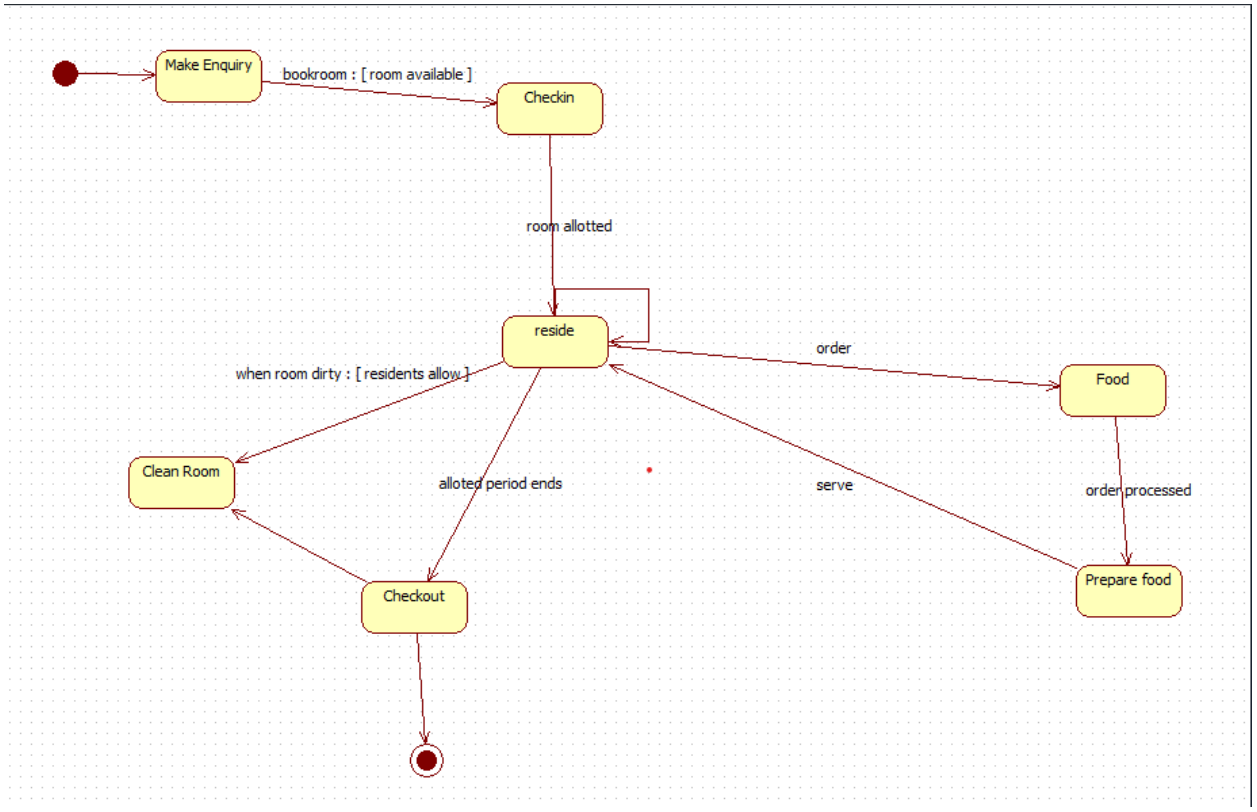
8. Preliminary Schedule and Budget:
Development is estimated to take 4 months with a budget of $20,000. This includes design, coding, testing and deployment.

Class Diagram: fig 1.1



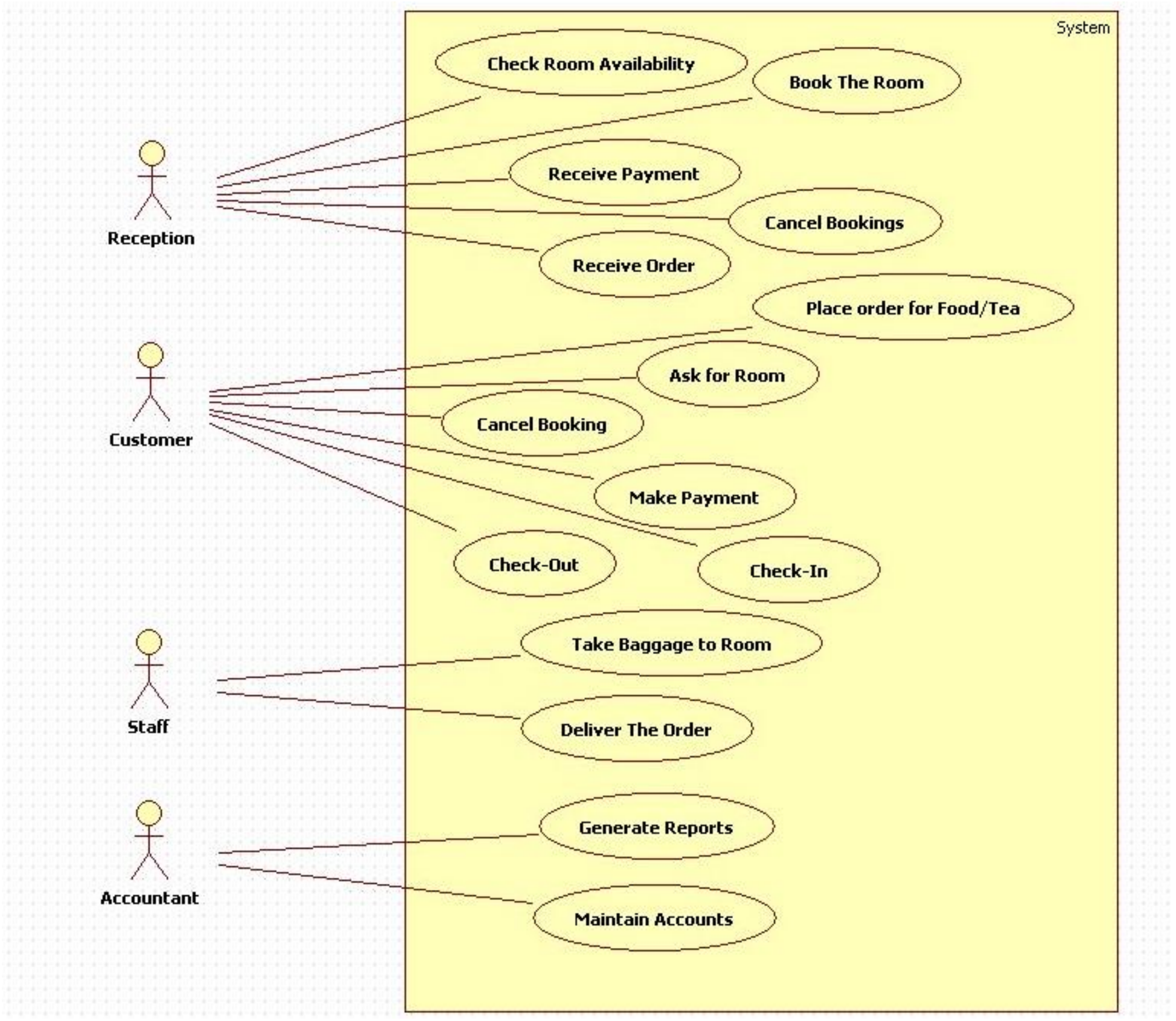
The class diagram of the Hotel Management System illustrates the main entities involved in hotel operations and how they interact. The central HotelManagement class is linked to major components such as Employees, Customer, Menu, Order, Bill, and Payment. Employees are modeled using a generalization relationship, with Server and Chef as specialized roles. Customers place orders based on items from the Menu, and each order is linked to both a server and a bill. Bills are connected to payments, which are further classified into Cash and Card types. The Table class maintains seating and availability details, associating customers and servers. The diagram includes associations, generalization, and aggregation to show how different classes collaborate to handle reservations, ordering, billing, and payment processes in the hotel.

State Diagram Fig 1.2



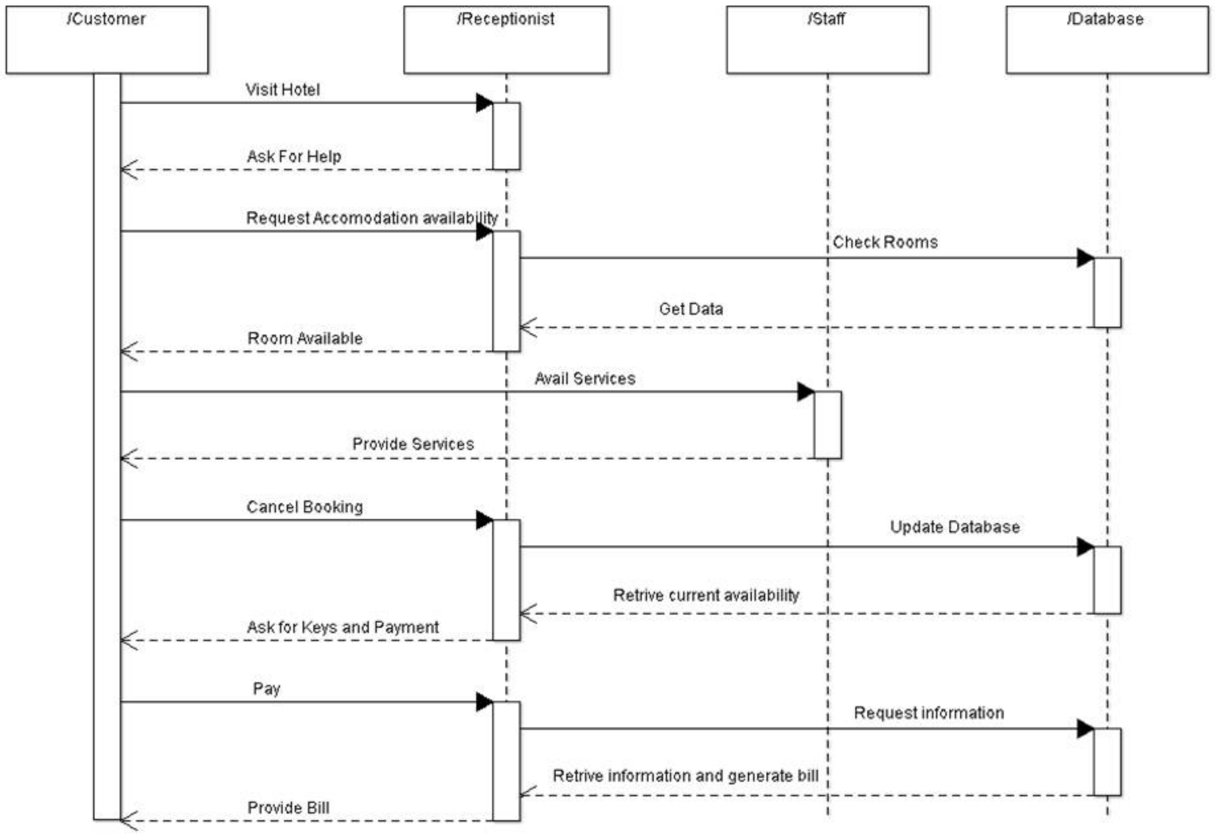
The state diagram represents the lifecycle of a hotel guest from enquiry to checkout. The process begins when a customer makes an enquiry and, if a room is available, moves to the check-in state. After the room is allotted, the guest enters the "reside" state, where they may request services such as food or room cleaning. Food orders transition to preparation and are then served back to the guest. If the room becomes dirty and residents permit, the system moves to the "Clean Room" state before returning to the residing state. Once the allotted stay period ends, the process transitions to the "Checkout" state, completing the customer's interaction with the hotel.

Use Case Diagram:fig 1.3



The use-case diagram illustrates the interactions between different users and the Hotel Management System. The **Receptionist** handles key operational tasks such as checking room availability, booking rooms, cancelling bookings, receiving orders, and processing payments. The **Customer** interacts with the system to ask for rooms, check in, place food orders, make payments, cancel bookings, and check out. The **Staff** assists by taking baggage to rooms and delivering food orders based on system instructions. The **Accountant** uses the system to generate financial reports and maintain accounts. Together, these use cases represent the essential functional requirements needed to manage hotel operations efficiently.

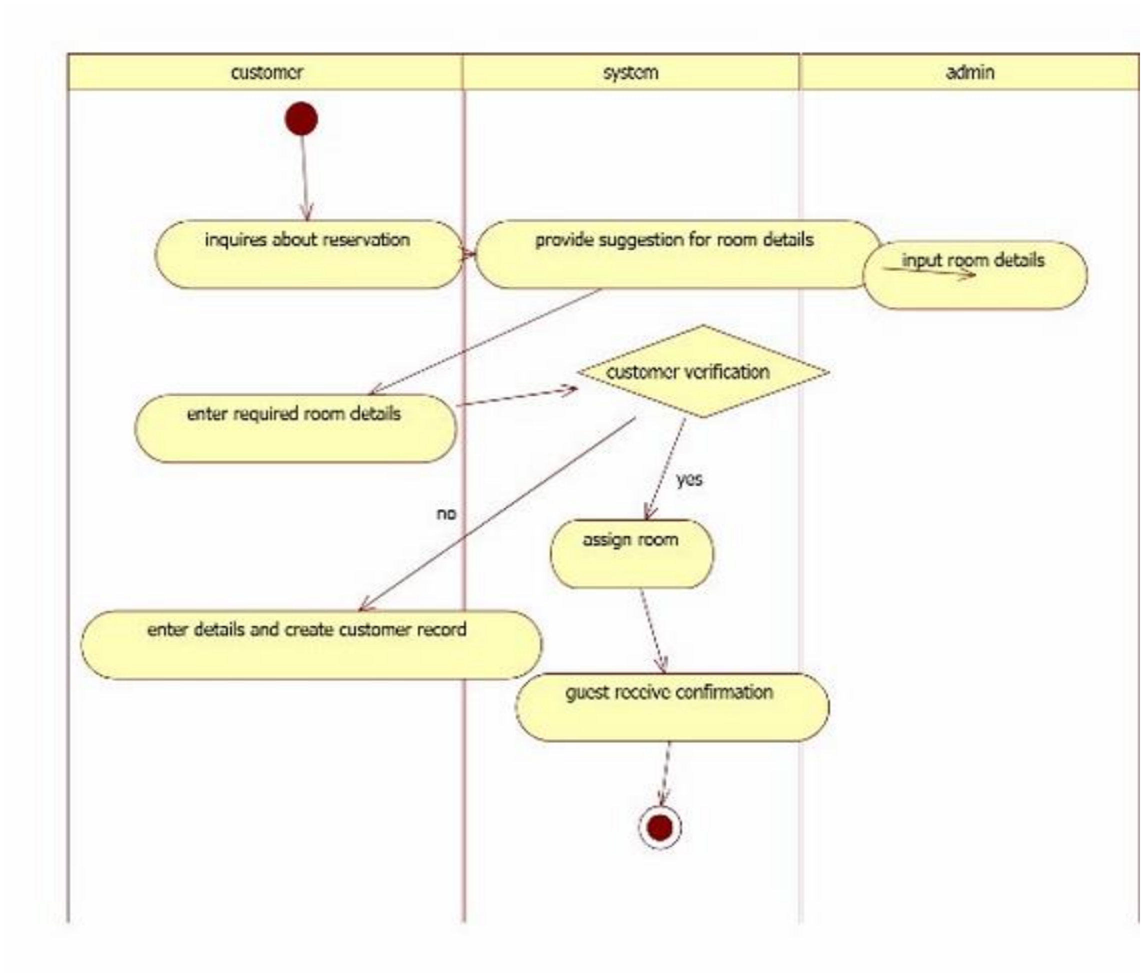
Sequence Diagram : Fig 1.4



The sequence diagram illustrates the interaction between the Customer, Receptionist, Staff, and Database during hotel service operations. The process begins when the customer visits the hotel and requests help, after which the receptionist checks room availability by requesting data from the staff, who then queries the database. Once availability is confirmed, the receptionist informs the customer, who then avails various services. If the customer cancels a booking, the receptionist instructs the staff to update the database. When the customer requests keys or proceeds to payment, the receptionist retrieves updated availability and billing information from the database. The customer completes the payment, and the receptionist retrieves the billing details from the database and provides the final bill, completing the interaction cycle.

fig 1.5

Advanced Activity Diagram

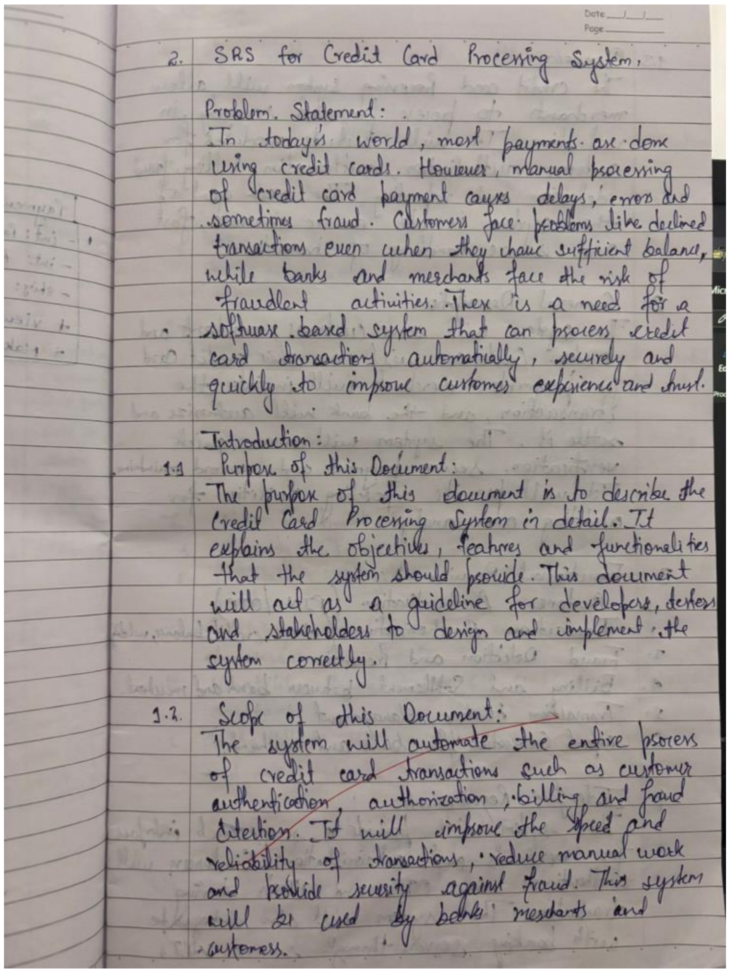

customer
system
admin
inquires about reservation
provide suggestion for room details
input room details
customer verification
enter required room details
yes
no
assign room
enter details and create customer record
guest receive confirmation

The advanced activity diagram illustrates the coordinated reservation workflow involving the customer, system, and admin. The process begins when the customer inquires about a reservation, prompting the system to suggest suitable room options. The admin enters room details into the system, enabling the customer to provide the required booking information. The system then performs customer verification; if verification fails, the customer must re-enter the details to create a valid record. If verification succeeds, the system assigns a room and sends a confirmation to the guest, completing the reservation process

2. Credit Card Processing

Problem Statement: A secure and efficient system is required to verify credit card details, authorize transactions, detect fraud, and maintain accurate payment records. Manual or outdated processing methods lead to errors, delays, and security risks. The Credit Card Processing System aims to automate transaction validation, ensure secure data handling, provide real-time authorization, and support merchants, customers, and administrators in managing credit card payments effectively.

SRS – Software Requirements Specification



Date __/__/__
Page ____

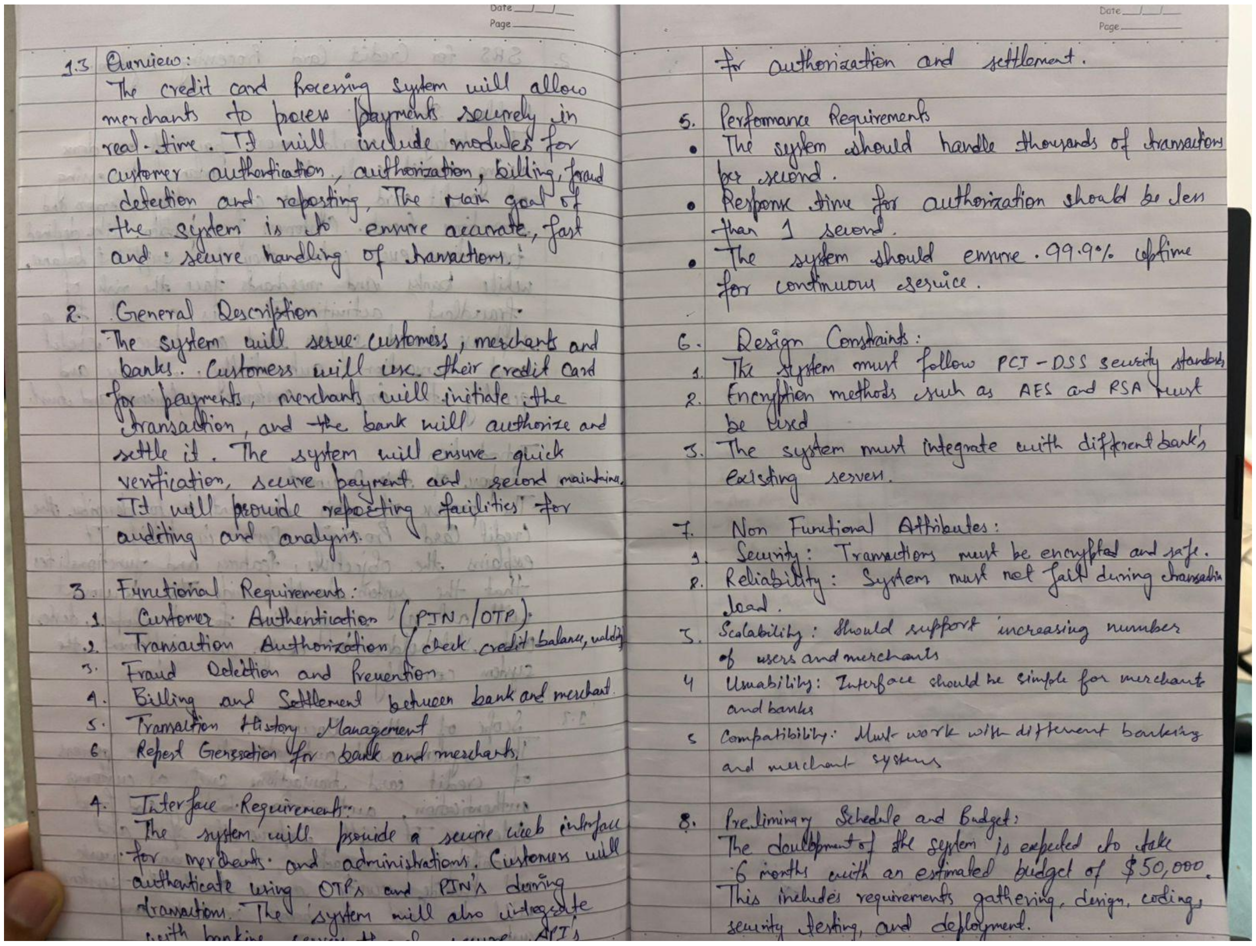
2. SRS for Credit Card Processing System.

Problem Statement:
In today's world, most payments are done using credit cards. However, manual processing of credit card payment causes delays, errors and sometimes fraud. Customers face problems like declined transactions even when they have sufficient balance, while banks and merchants face the risk of fraudulent activities. There is a need for a software based system that can process credit card transactions automatically, securely and quickly to improve customer experience and trust.

Introduction:

1.1 Purpose of this Document:
The purpose of this document is to describe the Credit Card Processing System in detail. It explains the objectives, features and functionalities that the system should provide. This document will act as a guideline for developers, testers and stakeholders to design and implement the system correctly.

1.2. Scope of this Document:
The system will automate the entire process of credit card transactions such as customer authentication, authorization, billing and fraud detection. It will improve the speed and reliability of transactions, reduce manual work and provide security against fraud. This system will be used by banks, merchants and customers.



Date __/__/__
Page ____

1.3 Overview:
The credit card processing system will allow merchants to process payments securely in real-time. It will include modules for customer authentication, authorization, billing, fraud detection and reporting. The main goal of the system is to ensure accurate, fast and secure handling of transactions.

2. General Description
The system will serve customers, merchants and banks. Customers will use their credit card for payments, merchants will initiate the transaction, and the bank will authorize and settle it. The system will ensure quick verification, secure payment and record maintaince. It will provide reporting facilities for auditing and analysis.

3. Functional Requirements:
1. Customer Authentication (PIN / OTP).
2. Transaction Authorization (check credit balance, validity
3. Fraud Detection and Prevention
4. Billing and Settlement between bank and merchant.
5. Transaction History Management
6. Report Generation for bank and merchants.

4. Interface Requirements:
The system will provide a secure web interface for merchants and administrations. Customers will authenticate using OTP's and PIN's during transactions. The system will also integrate with banking servers through secure API's

Date __/__/__
Page ____

for authorization and settlement.

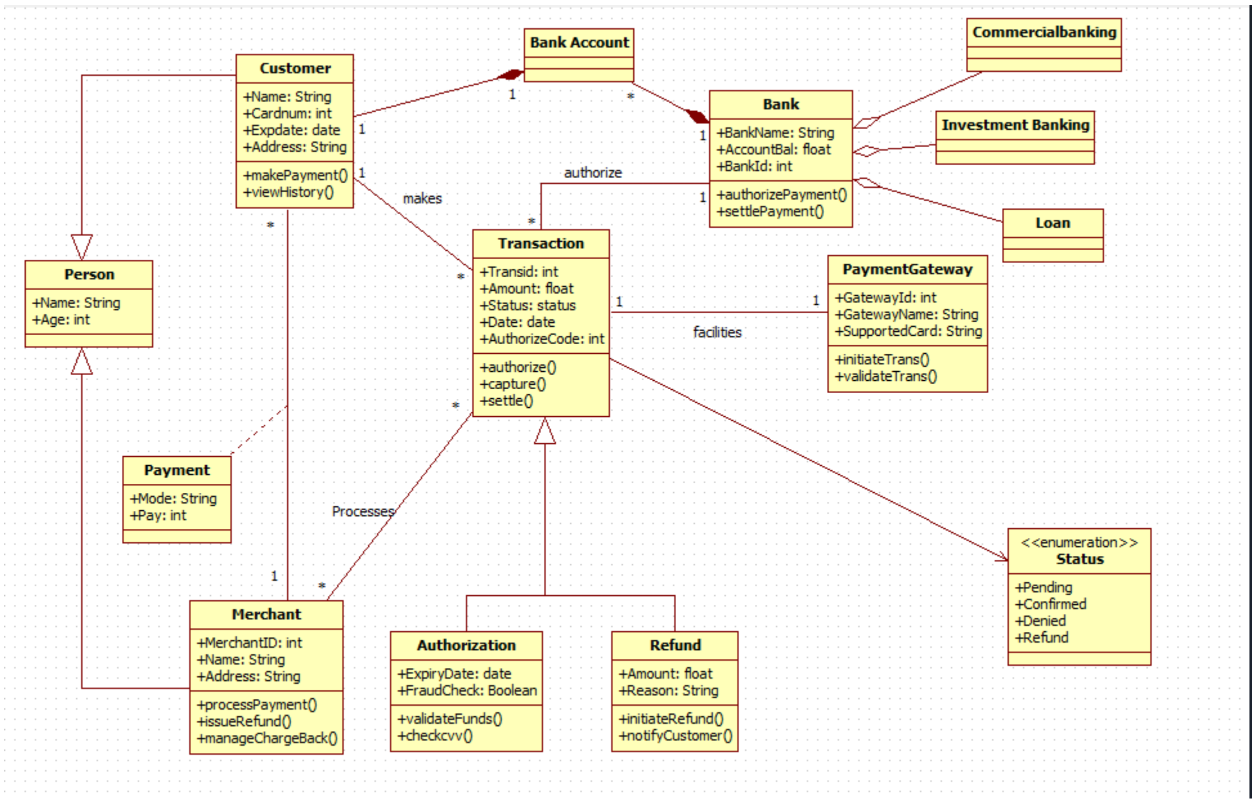
5. Performance Requirements
- The system should handle thousands of transactions per second.
- Response time for authorization should be less than 1 second.
- The system should ensure 99.9% uptime for continuous service.

6. Design Constraints:
1. The system must follow PCI-DSS security standards
2. Encryption methods such as AES and RSA must be used
3. The system must integrate with different bank's existing servers.

7. Non Functional Attributes:
1. Security: Transactions must be encrypted and safe.
2. Reliability: System must not fail during transaction load.
3. Scalability: Should support increasing number of users and merchants
4 Usability: Interface should be simple for merchants and banks
5 Compatibility: Must work with different banking and merchant systems

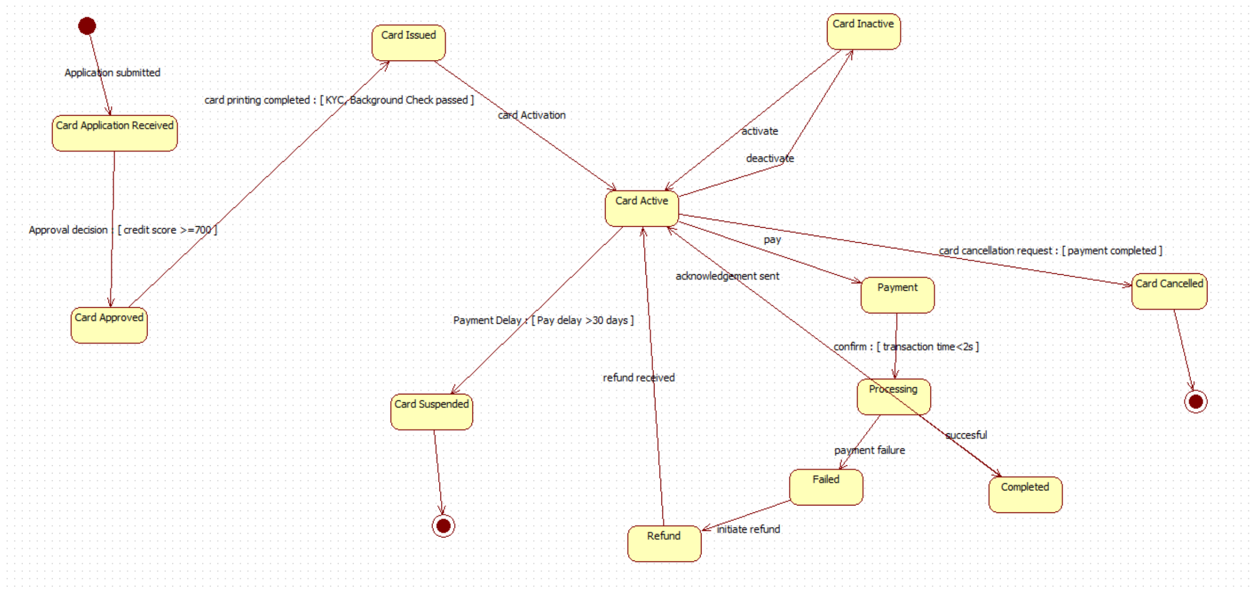
8. Preliminary Schedule and Budget:
The development of the system is expected to take 6 months with an estimated budget of $50,000. This includes requirements gathering, design, coding, security testing, and deployment.

Class Diagram Fig:2.1



The class diagram for the Credit Card Processing System shows how customers, merchants, banks, and payment gateways interact to process credit card transactions. A customer, inheriting details from a general person class, initiates payments that create transaction objects containing information such as amount, date, and status. Each transaction is linked to a bank account, and the bank is responsible for authorizing and settling the payment. The payment gateway helps facilitate the transaction by initiating and validating payment requests. Merchants receive the payment and can also manage refunds or chargebacks, which are handled through dedicated authorization and refund classes that perform fraud checks and validate funds. An enumeration class defines the possible transaction statuses, including pending, confirmed, denied, and refunded. Overall, the diagram captures the complete flow from initiating a payment to authorization, settlement, and refund processing.

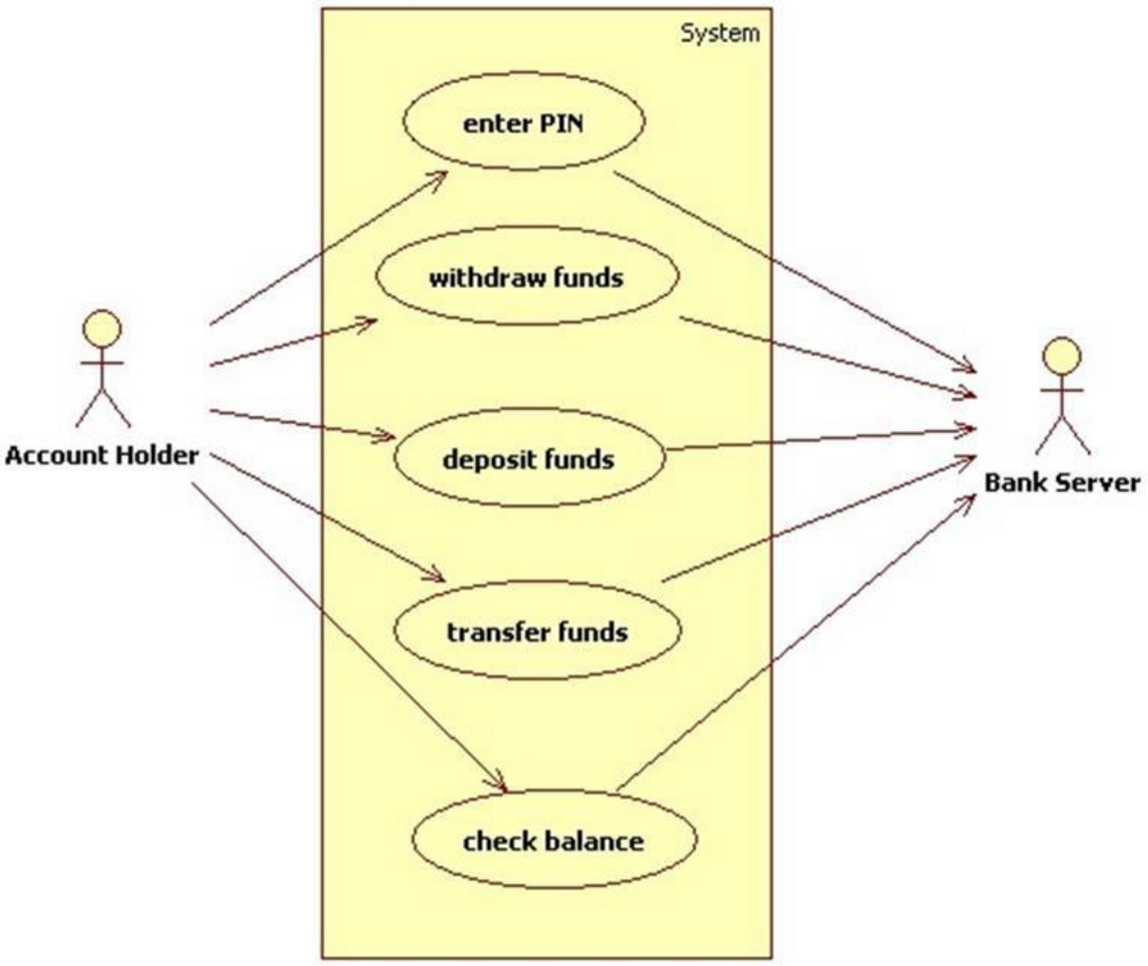
State Diagram Fig 2.2



Card Issued
Card Inactive
Application submitted
card printing completed : [KYC, Background Check passed]
card Activation
Card Application Received
activate
deactivate
Card Active
Approval decision : [credit score >=700]
pay
card cancellation request : [payment completed]
acknowledgement sent
Payment
Card Cancelled
Card Approved
Payment Delay : [Pay delay >30 days]
confirm : [transaction time<2s]
refund received
Processing
Card Suspended
succesful
payment failure
Failed
Completed
initiate refund
Refund

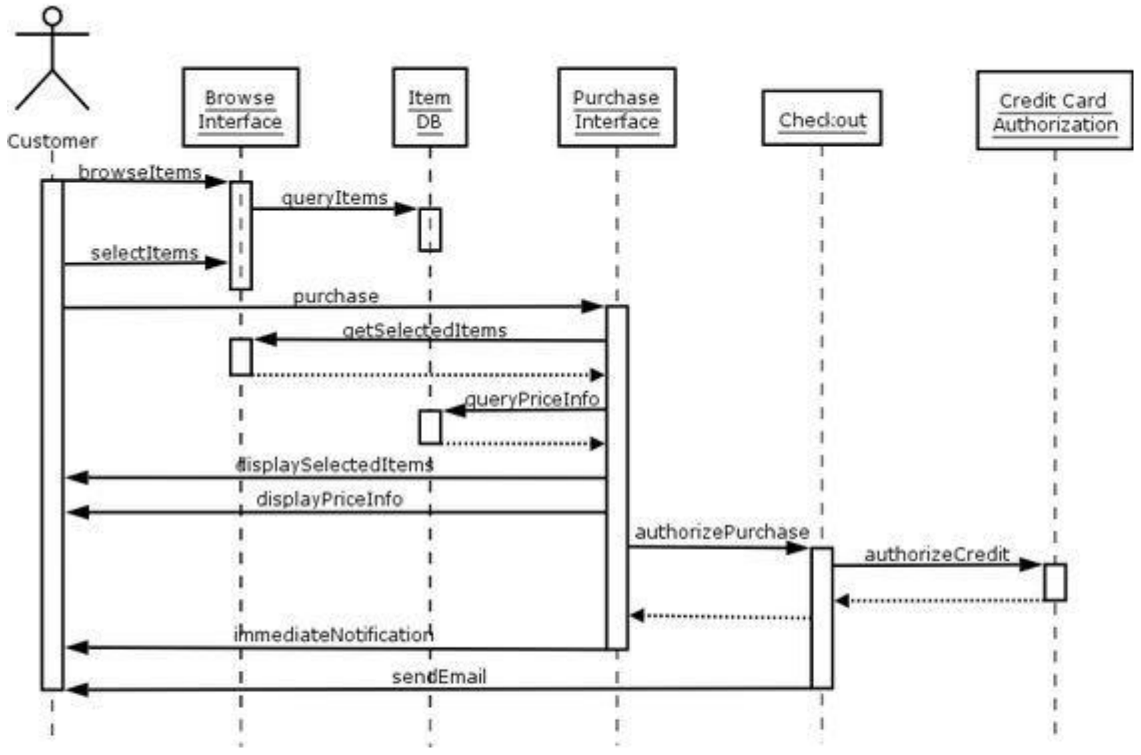
The state diagram shows the complete lifecycle of a credit card, beginning with the application submitted and moving to the card application received and approved states. Once approved and printed, the card enters the card issued state and becomes card active after activation. In the active state, the card can be used for payments, which then go through processing and end as either completed or failed. The active card can also be deactivated, reactivated, refunded, or cancelled based on user actions or system conditions. If payment delays exceed the allowed limit, the card moves to the suspended state. When a refund is issued or normal activity resumes, the system returns the card to the active state. The cycle ends either when the card is cancelled or permanently suspended.

Use Case Diagram fig : 2.3



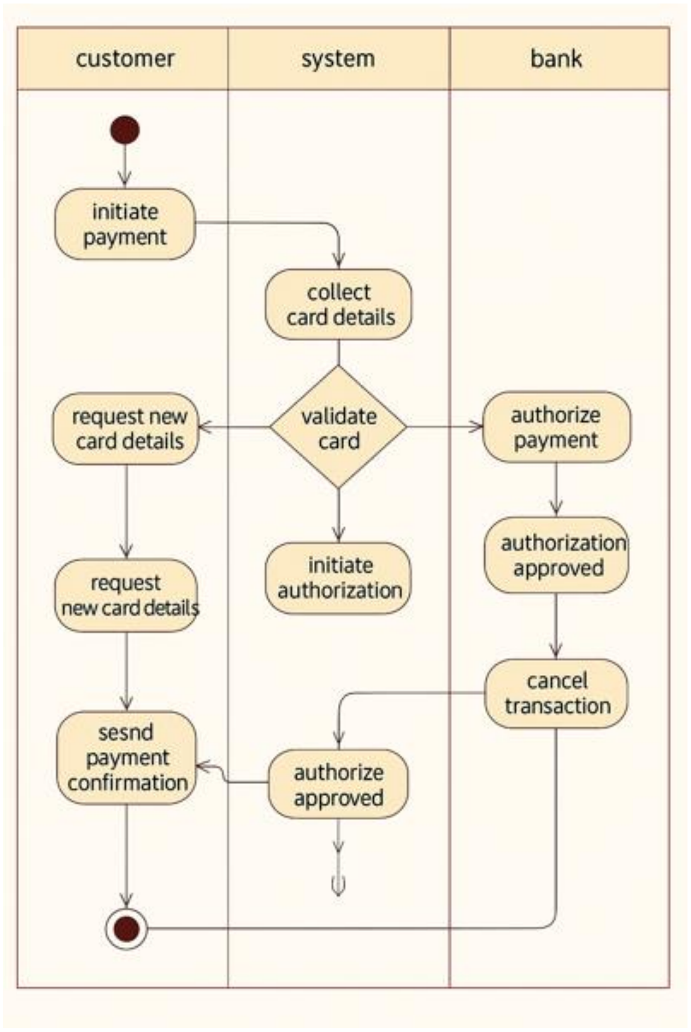
The use case diagram shows how an account holder interacts with the system to perform basic banking operations, such as entering a PIN, withdrawing funds, depositing money, transferring funds, and checking account balance. Each of these actions requires communication with the bank server, which verifies credentials, updates account information, and processes the requested transactions. The diagram highlights the role of the account holder as the primary user and the bank server as an external actor responsible for validating and completing all financial operations within the system.

Sequence Diagram: Fig 2.4



The sequence diagram illustrates the steps involved when a customer browses items, selects products, and completes a purchase. The customer begins by browsing items through the browse interface, which queries the item database to retrieve product information. After selecting items, the system fetches the selected items and checks price details from the database. The purchase interface then displays the selected items and their prices to the customer. When the customer proceeds to checkout, the system sends an authorization request to the credit card authorization service to validate the payment. Once authorized, the system confirms the purchase, sends an immediate notification, and delivers a confirmation email to the customer, completing the transaction flow.

Activity Diagram: Fig 2.5

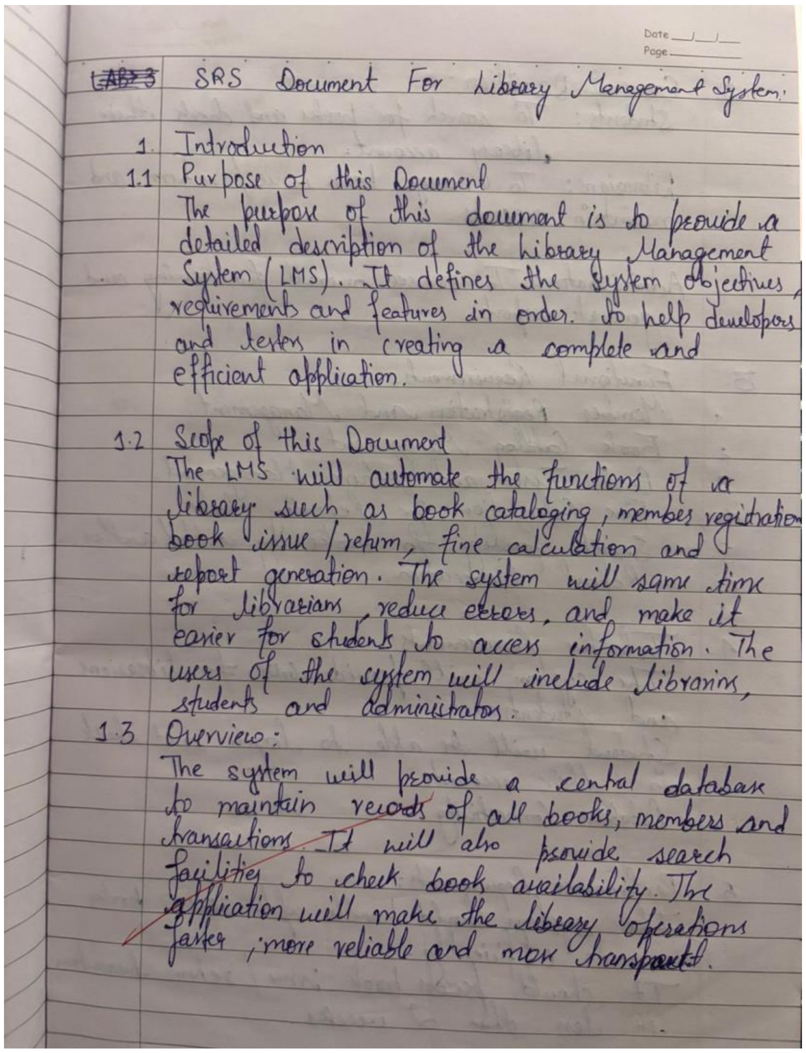


The advanced activity diagram shows the complete payment workflow involving the customer, the system, and the bank. The process begins when the customer initiates a payment, after which the system collects the card details and validates them. If the information is incorrect, the system sends a request for new card details and the customer must re-enter them. When the card is valid, the system initiates an authorization request to the bank, which verifies the details and either approves or cancels the transaction. If approved, the system sends a payment confirmation back to the customer. The diagram clearly separates responsibilities using swimlanes and shows the coordinated actions between the customer, the system, and the bank throughout the payment process.

3.Library Management System

Problem Statement: Managing library operations manually often leads to issues such as misplaced records, delayed book issuance, difficulty tracking borrowed items, and errors in maintaining member information. As libraries grow, maintaining accurate records of books, members, transactions, and availability becomes increasingly challenging. A Library Management System is required to automate these tasks by providing an organized platform for book cataloging, member registration, book issue/return, fine calculation, and inventory tracking. The system should improve efficiency, reduce human errors, and ensure quick access to information for both librarians and users.

SRS FOR Library Management System :



Date __/__/__
Page ______

~~LAB 3~~ SRS Document For Library Management System:

1. Introduction

1.1 Purpose of this Document

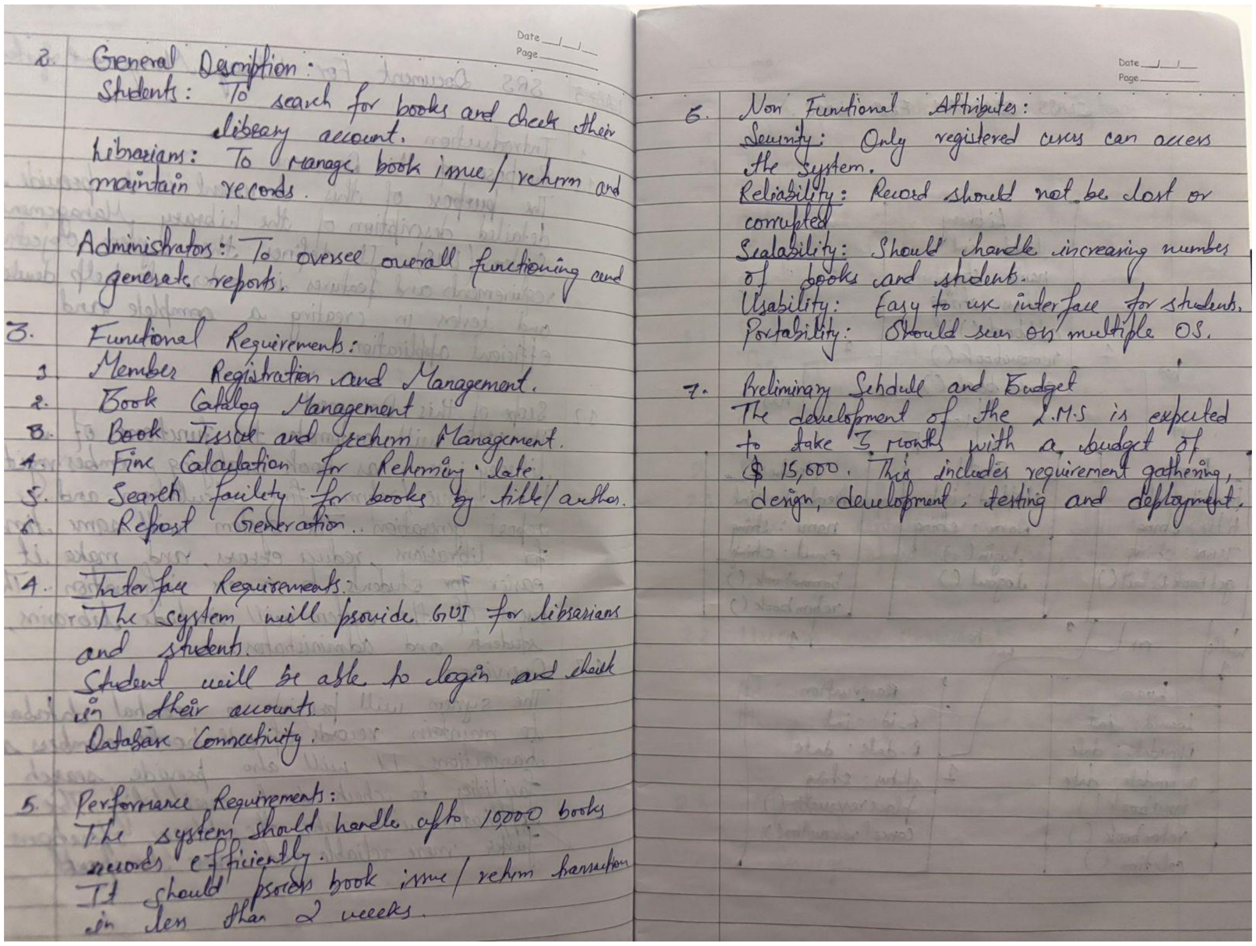
The purpose of this document is to provide a detailed description of the library Management System (LMS). It defines the system objectives, requirements and features in order to help developers and testers in creating a complete and efficient application.

1.2 Scope of this Document

The LMS will automate the functions of a library such as book cataloging, member registration book issue / return, fine calculation and report generation. The system will same time for librarians, reduce errors, and make it easier for students to access information. The users of the system will include librarins, students and administrators.

1.3 Overview:

The system will provide a central database to maintain records of all books, members and transactions. It will also provide search facilities to check book availability. The application will make the library operations faster, more reliable and more transparent.



Date __/__/__
Page ____

2. General Description:
Students: To search for books and check their library account.
Librarians: To manage book issue / return and maintain records.

Administrators: To oversee overall functioning and generate reports.

3. Functional Requirements:
1. Member Registration and Management.
2. Book Catalog Management
3. Book Issue and return Management.
4. Fine Calculation for Returning late.
5. Search facility for books by title / author.
6. Report Generation.

4. Interface Requirements:
The system will provide GUI for librarians and students.
Student will be able to login and check in their accounts.
Database Connectivity.

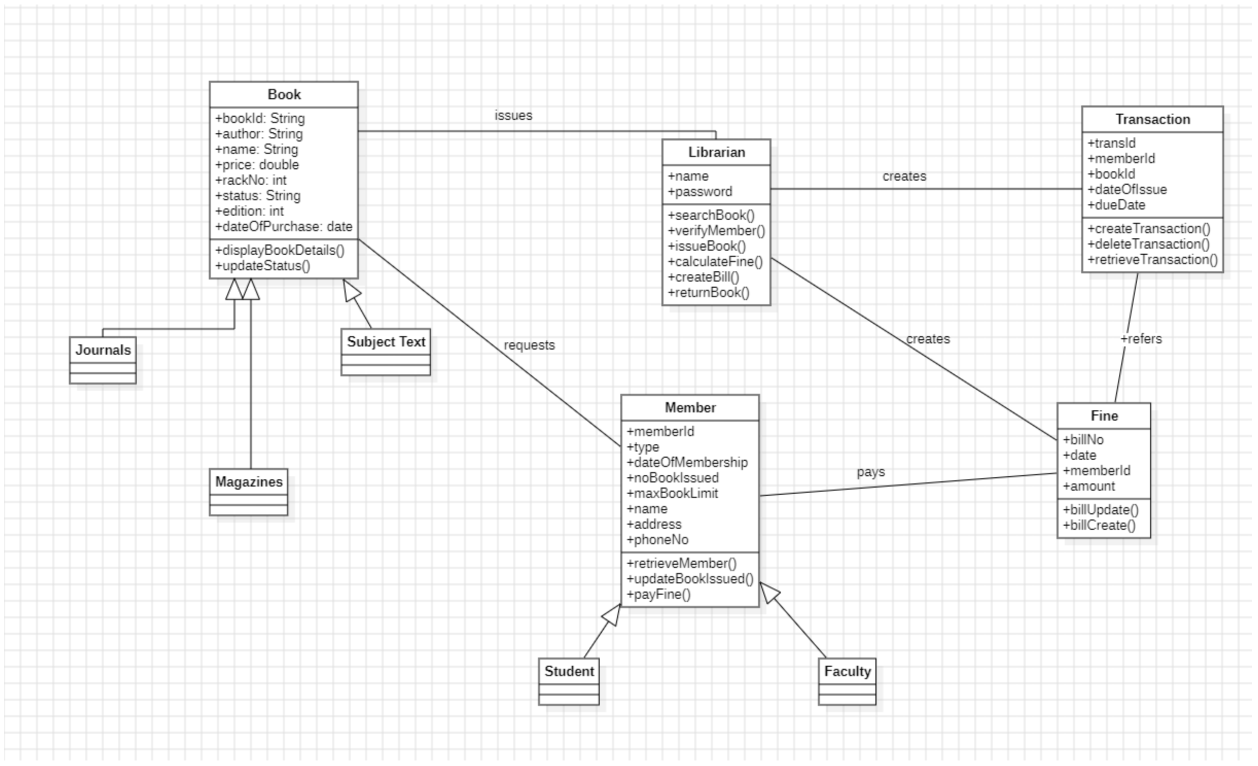
5. Performance Requirements:
The system should handle upto 10,000 books records efficiently.
It should process book issue / return transaction in less than 2 weeks.

Date __/__/__
Page ____

6. Non Functional Attributes:
Security: Only registered users can access the System.
Reliability: Record should not be lost or corrupted
Scalability: Should handle increasing numbes of books and students.
Usability: Easy to use interface for students.
Portability: Should run on multiple OS.

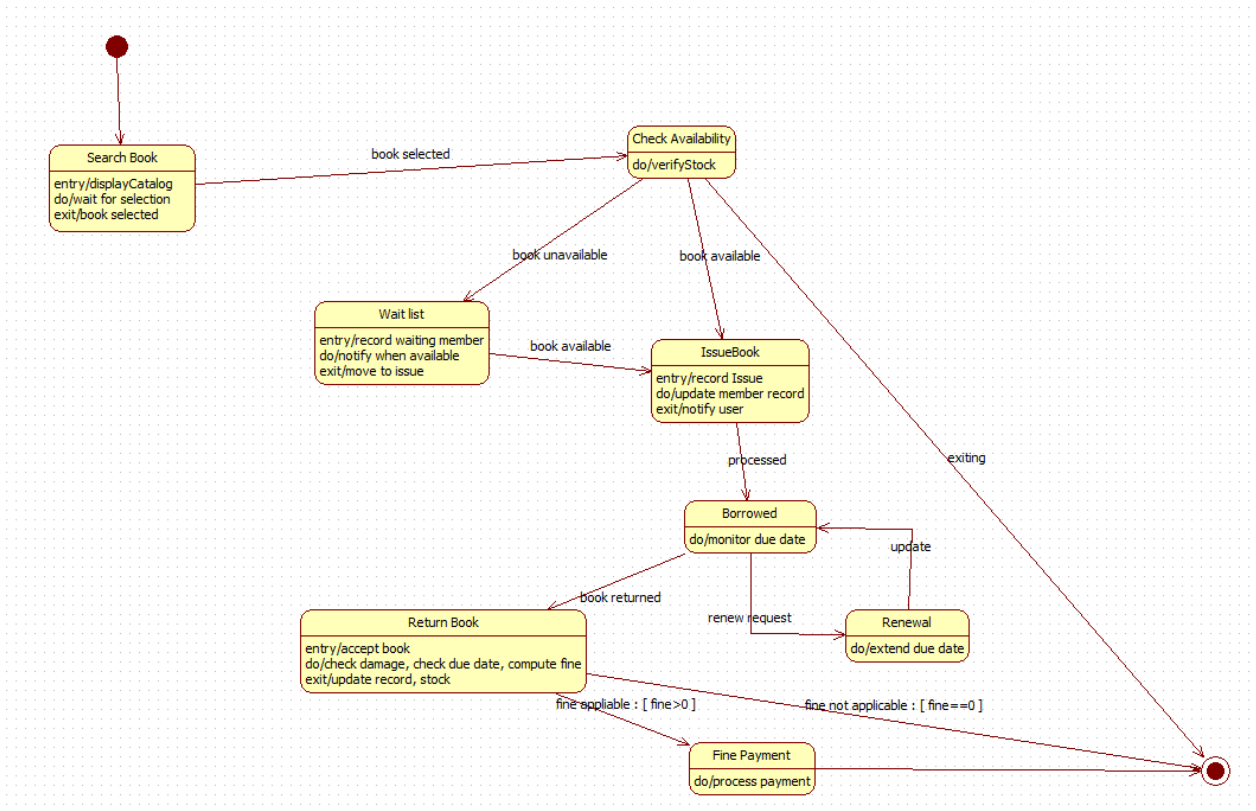
7. Preliminary Schedule and Budget
The development of the L.M.S is expected to take 3 months with a budget of $ 15,000. This includes requirement gathering, design, development, testing and deployment.

Class Diagram : Fig 3.1



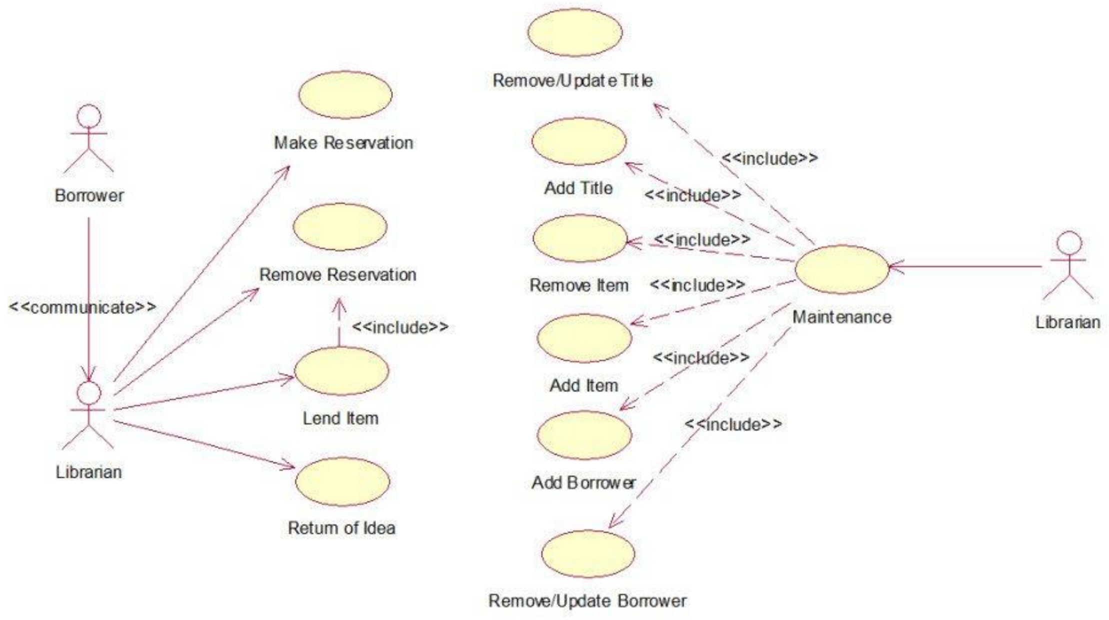
The class diagram for the Library Management System shows how books, members, librarians, transactions, and fines interact within the system. The Book class stores details about each book and is specialized into journals, magazines, and subject texts through inheritance. Members request books and may be either students or faculty, both inheriting attributes from the Member class. The Librarian manages core operations such as searching books, verifying members, issuing and returning books, calculating fines, and creating bills. Each issued book creates a Transaction record that stores issue details and due dates, while overdue returns generate entries in the Fine class, which is linked to both member and transaction information. Overall, the diagram captures how the system handles book management, member services, and fine processing through well-connected classes.

State Diagram : Fig 3.2



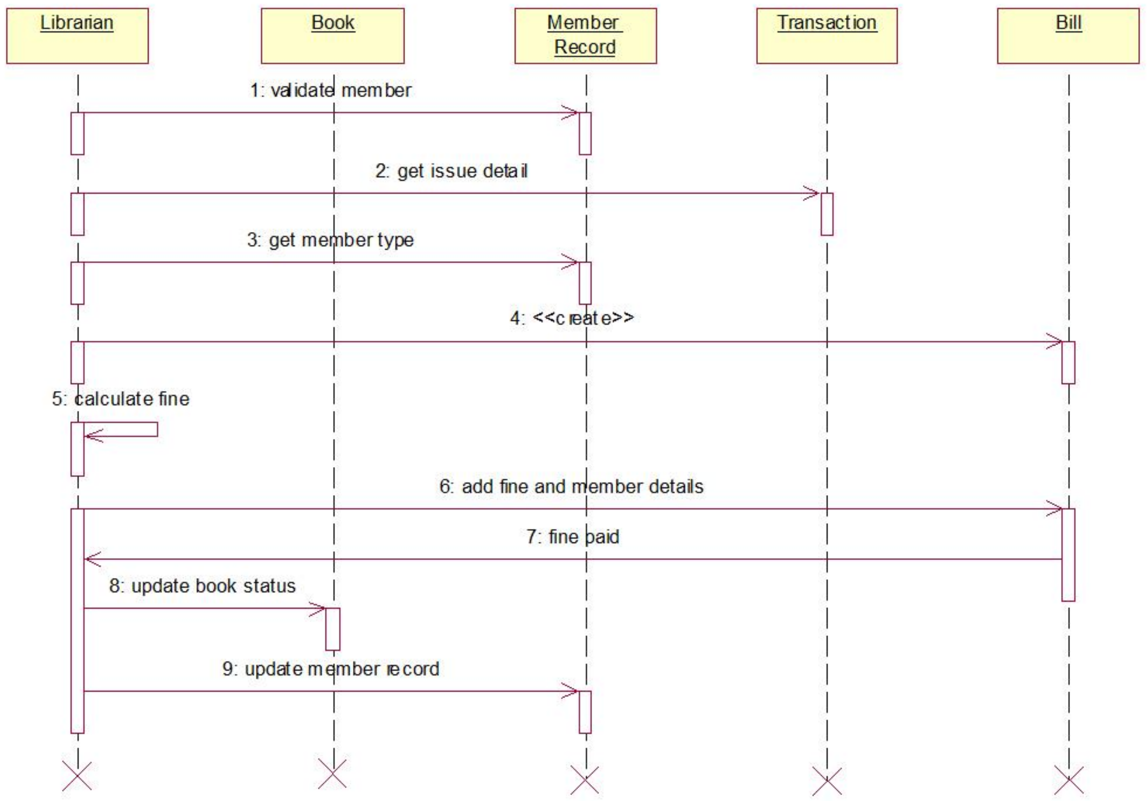
The state diagram shows the different stages a book goes through during the library borrowing process. It begins with the user searching for a book, after which the system checks its availability. If the book is unavailable, the request moves to the wait list state until a copy becomes available. When available, the system transitions to the issue book state, where member records and stock are updated. The book then enters the borrowed state, where the system monitors the due date. A renewal request extends the due date, while a return transitions the process to the return book state, where damage is checked, fines are calculated if necessary, and records are updated. If a fine is applicable, the flow moves to fine payment before completing the cycle. The diagram captures the lifecycle of a borrowed book from search to return, including renewal and fine handling.

Usecase Diagram: Fig 3.3



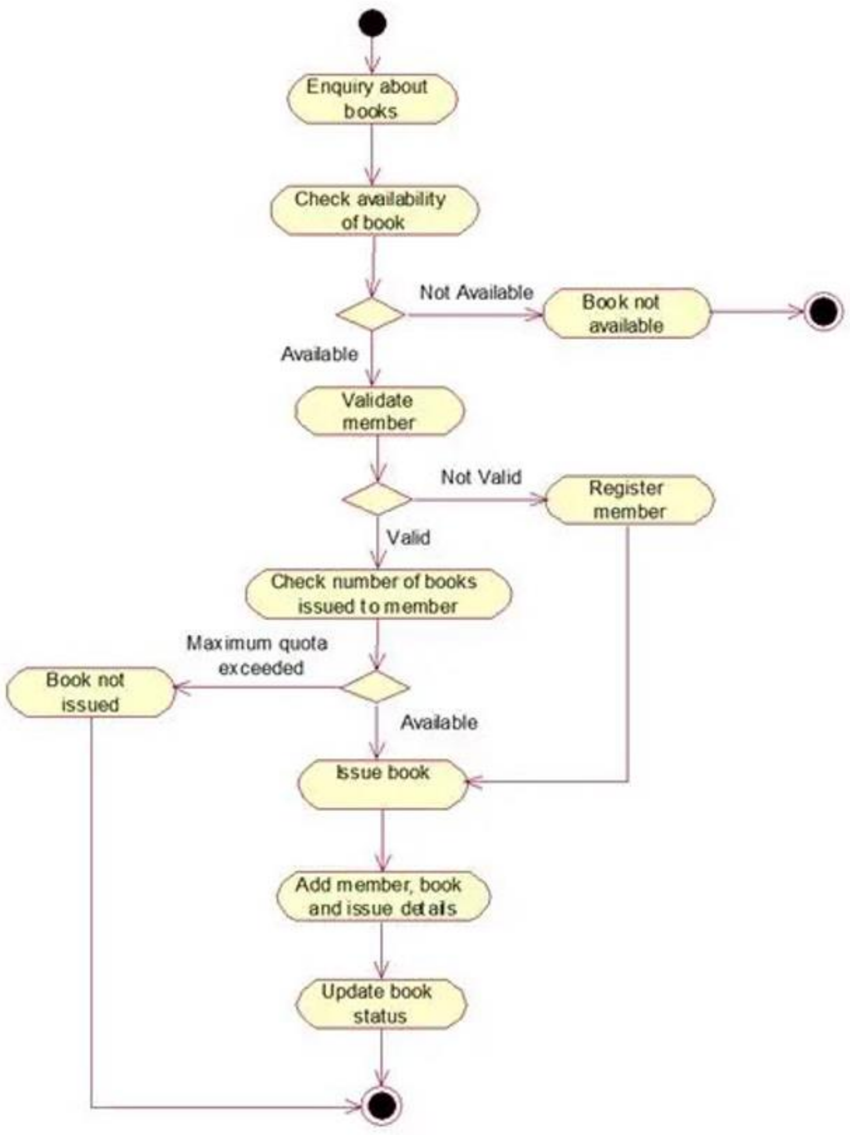
The use-case diagram shows how borrowers and librarians interact with the library system to manage reservations, item lending, returns, and maintenance tasks. Borrowers can make reservations, remove reservations, borrow items, and return items, all of which involve communication with the librarian. The librarian is responsible for performing core system operations such as adding or removing items, updating titles, adding or updating borrower details, and maintaining the overall catalog. These operations are grouped under the maintenance use case, which includes several supporting functions through include relationships. The diagram clearly separates user roles and illustrates how both borrowers and librarians interact with the system to manage library resources efficiently.

Sequence Diagram : Fig 3.4



The sequence diagram illustrates the process of handling a returned book and calculating any associated fines. The interaction begins when the librarian validates the member and retrieves issue details from the member record and transaction. The system then checks the member type and proceeds to calculate the fine if applicable. Once the fine is determined, the transaction and member records are updated with fine and member details, and the librarian waits for confirmation of fine payment. After the fine is paid, the librarian updates the book status to indicate its availability and finally updates the member record to reflect the returned item. The diagram clearly shows the flow of messages among the librarian, book system, member record, transaction, and billing components

Activity Diagram : Fig 3.5

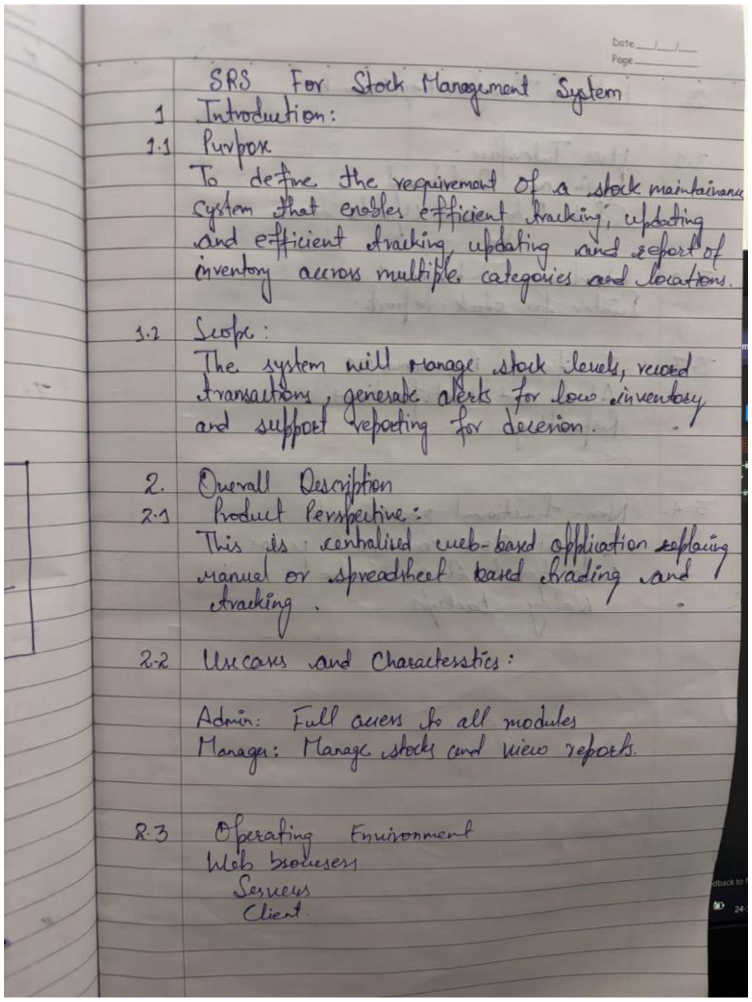


The activity diagram illustrates the workflow for issuing a book to a library member. It begins when a user makes an enquiry about a book, after which the system checks its availability. If the book is not available, the process ends. If available, the system validates the member; invalid members are directed to registration before proceeding. Once validated, the system checks whether the member has reached the maximum quota of issued books. If the quota is exceeded, the book is not issued. If eligible, the system issues the book, records the member and book details, and finally updates the book's status to reflect the issue. The diagram clearly represents the decision points and actions involved in ensuring proper book issuance

4. Stock Maintenance System

Problem Statement: Managing stock manually often leads to issues such as inaccurate inventory levels, misplaced records, delays in updating stock information, and difficulty tracking incoming and outgoing items. As the volume of products increases, maintaining accurate and real-time stock information becomes challenging. A Stock Maintenance System is needed to automate these tasks by recording stock details, monitoring quantities, updating inventory after sales or purchases, generating alerts for low stock, and maintaining proper logs of stock movements. The system should help organizations reduce errors, improve efficiency, and ensure smooth stock management across departments.

SRS:



Date __/__/__
Page ____

SRS For Stock Management System

1 Introduction:

1.1 Purpose

To define the requirement of a stock maintainance system that enables efficient tracking, updating and efficient tracking, updating and report of inventory across multiple categories and locations.

1.2 Scope:

The system will manage stock levels, record transactions, generate alerts for low inventory and support reporting for decision.

2. Overall Description

2.1 Product Perspective:

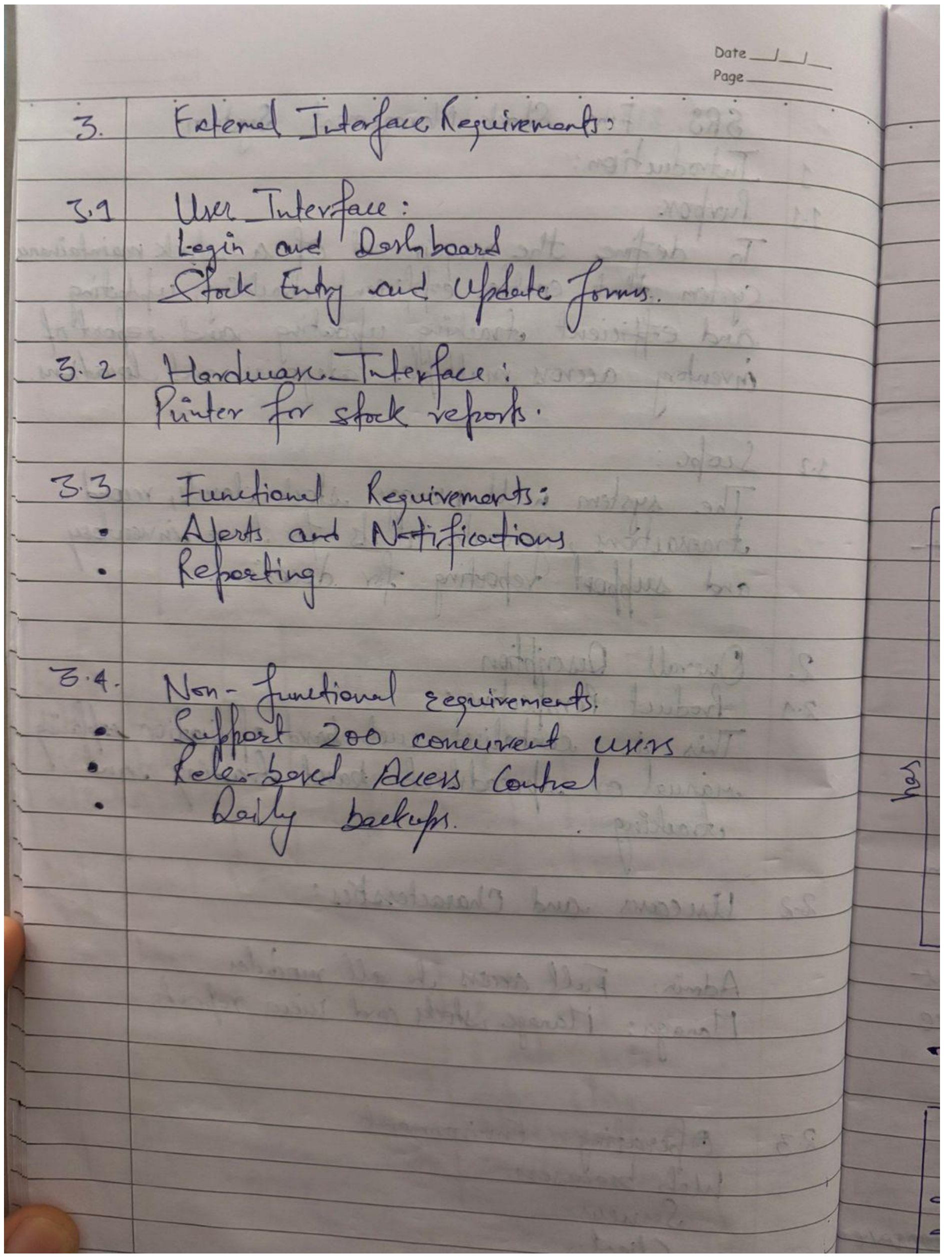
This is centralised web-based application replacing manual or spreadsheet based trading and tracking.

2.2 Usecases and Characteristics:

Admin: Full access to all modules
Manager: Manage stocks and view reports.

2.3 Operating Environment

Web browsers
Servers
Client.



Date __/__/__
Page ____

3. External Interface Requirements:

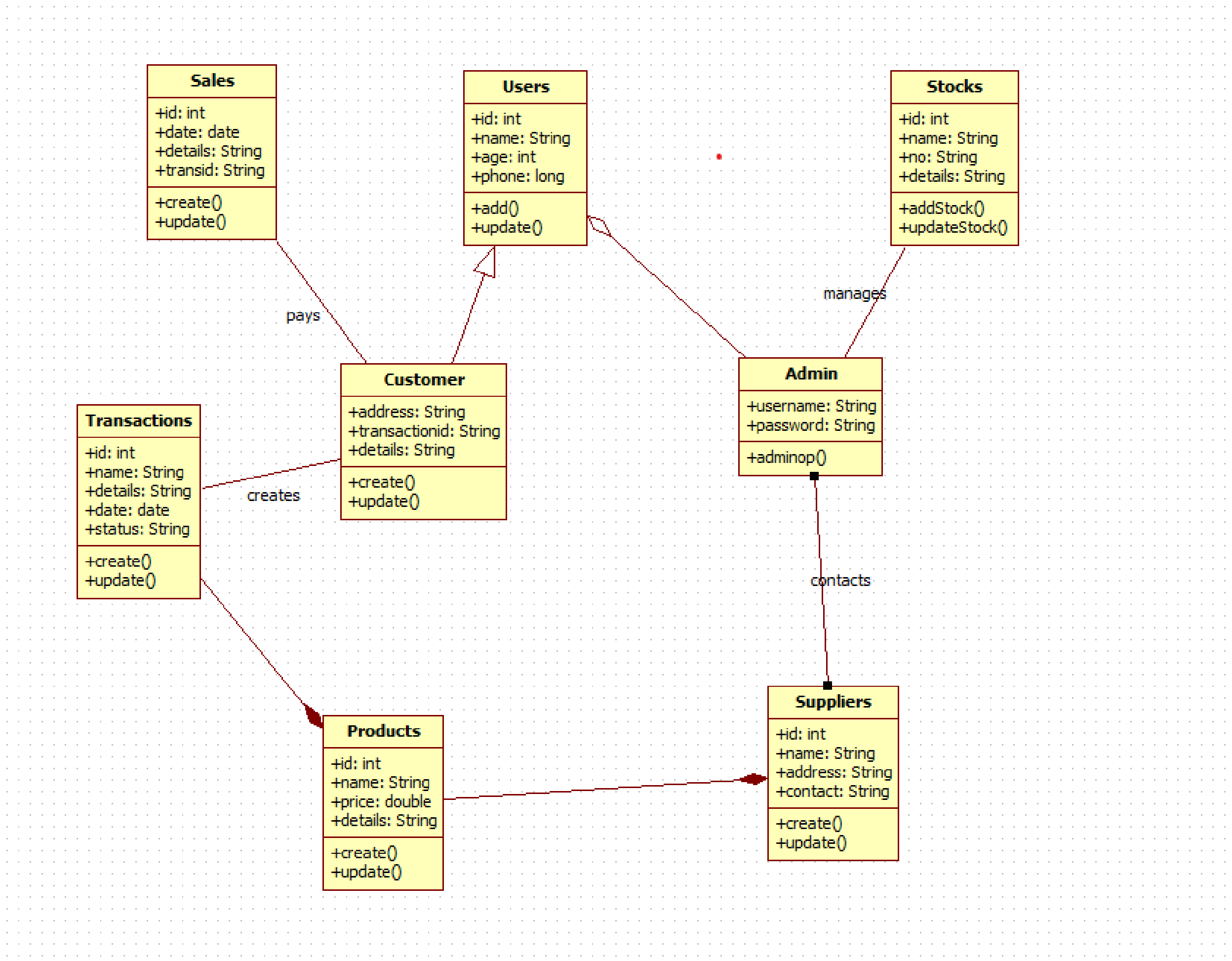
3.1 User Interface:
Login and Dashboard
Stock Entry and Update forms.

3.2 Hardware Interface:
Printer for stock reports.

3.3 Functional Requirements:
- Alerts and Notifications
- Reporting

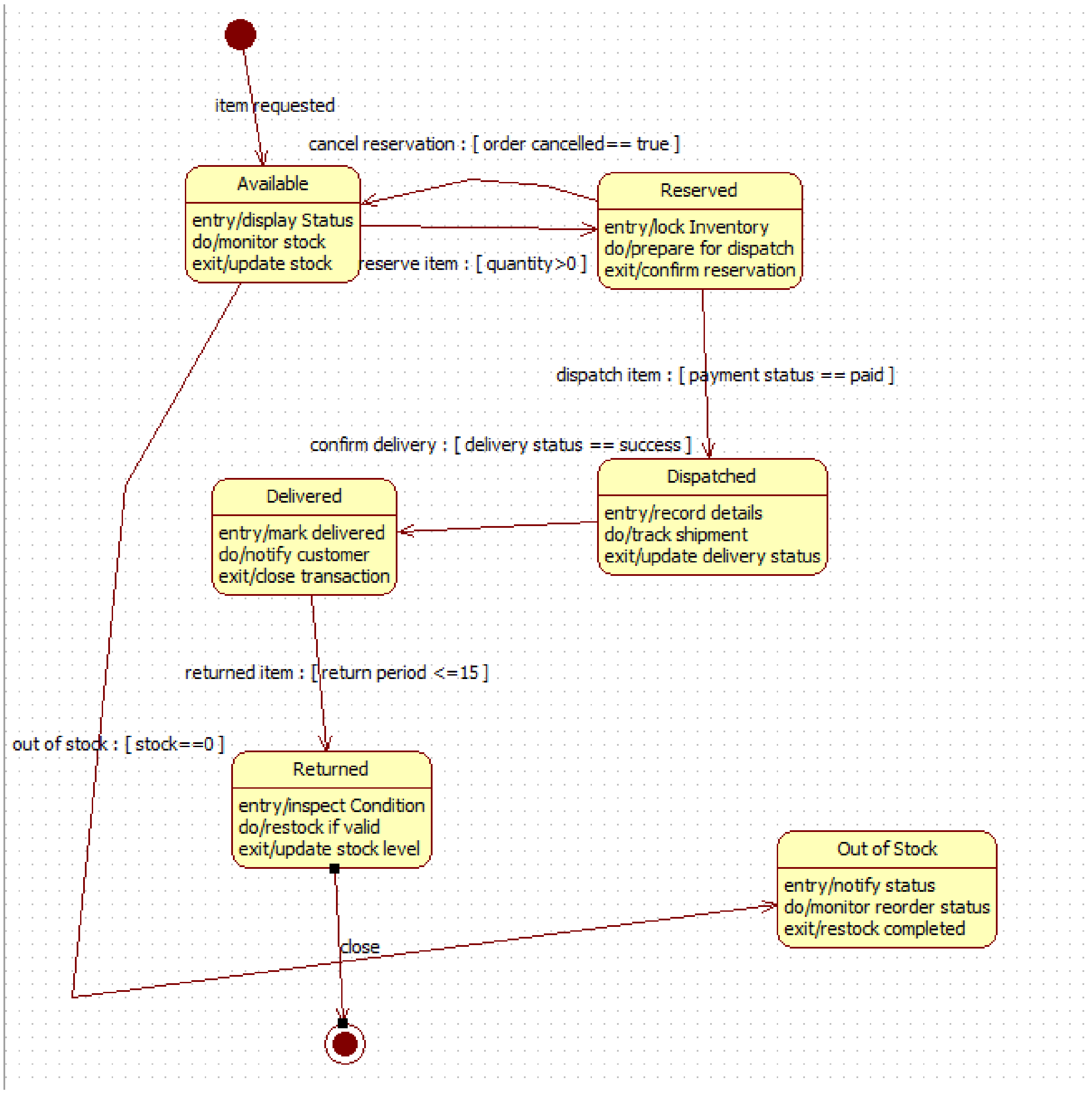
3.4. Non-functional requirements.
- Support 200 concurrent users
- Role-based Access Control
- Daily backups.

Class Diagram : Fig 4.1



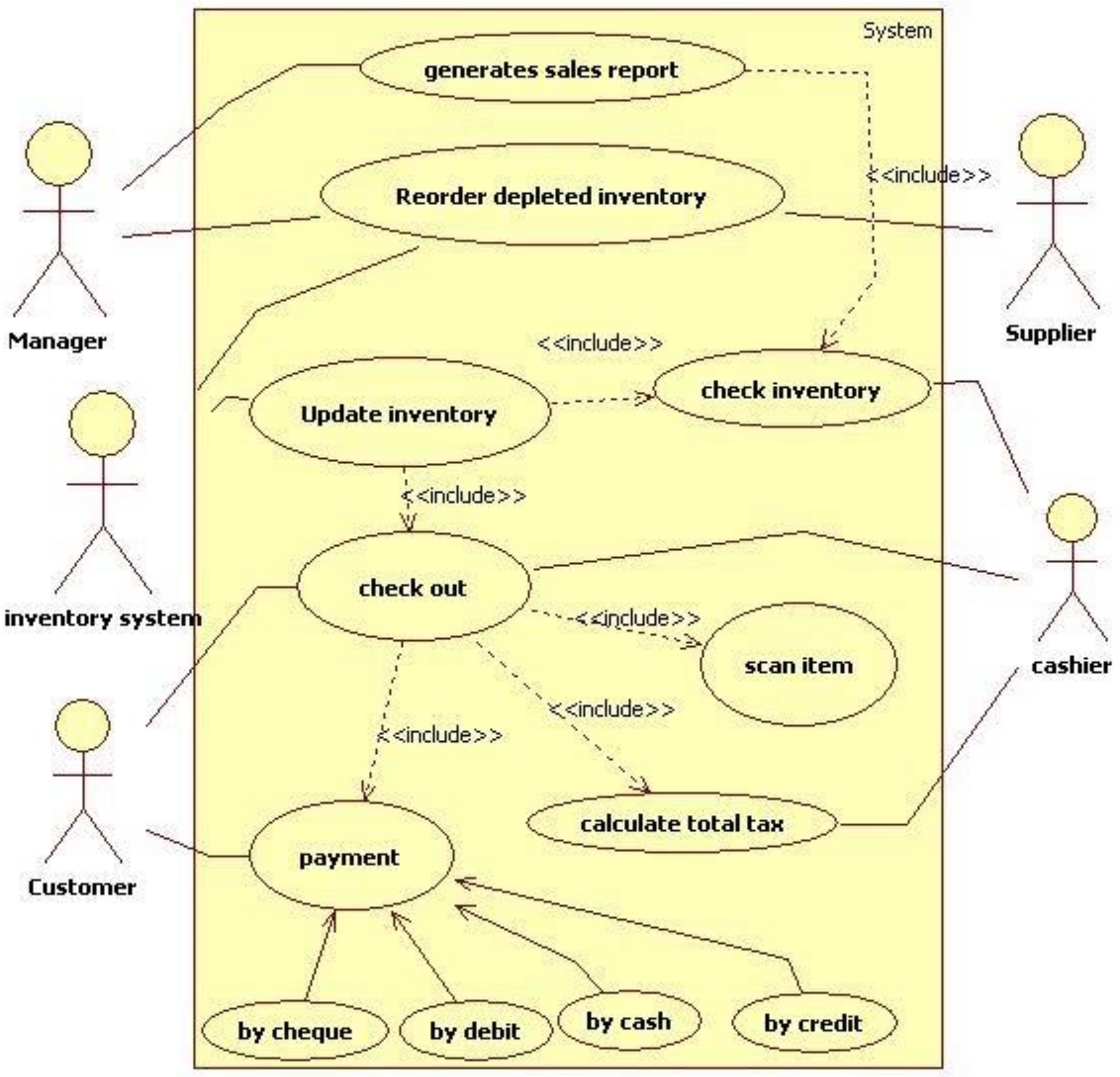
The class diagram for the Stock Maintenance System shows how different entities work together to manage inventory, products, sales, and supplier information. The Admin oversees the system by managing stock records and communicating with suppliers, who provide product details to maintain inventory levels. Products are associated with transactions, which record sales or stock movements. Customers create transactions when purchasing items, and Sales records the payment details linked to each transaction. Users represent staff members who interact with the system and update customer details when needed. Stocks store information about product quantities and are updated by the Admin as new supplies arrive. Overall, the diagram captures the relationships and responsibilities needed to maintain accurate stock levels and manage inventory operations efficiently.

State Diagram : Fig 4.2



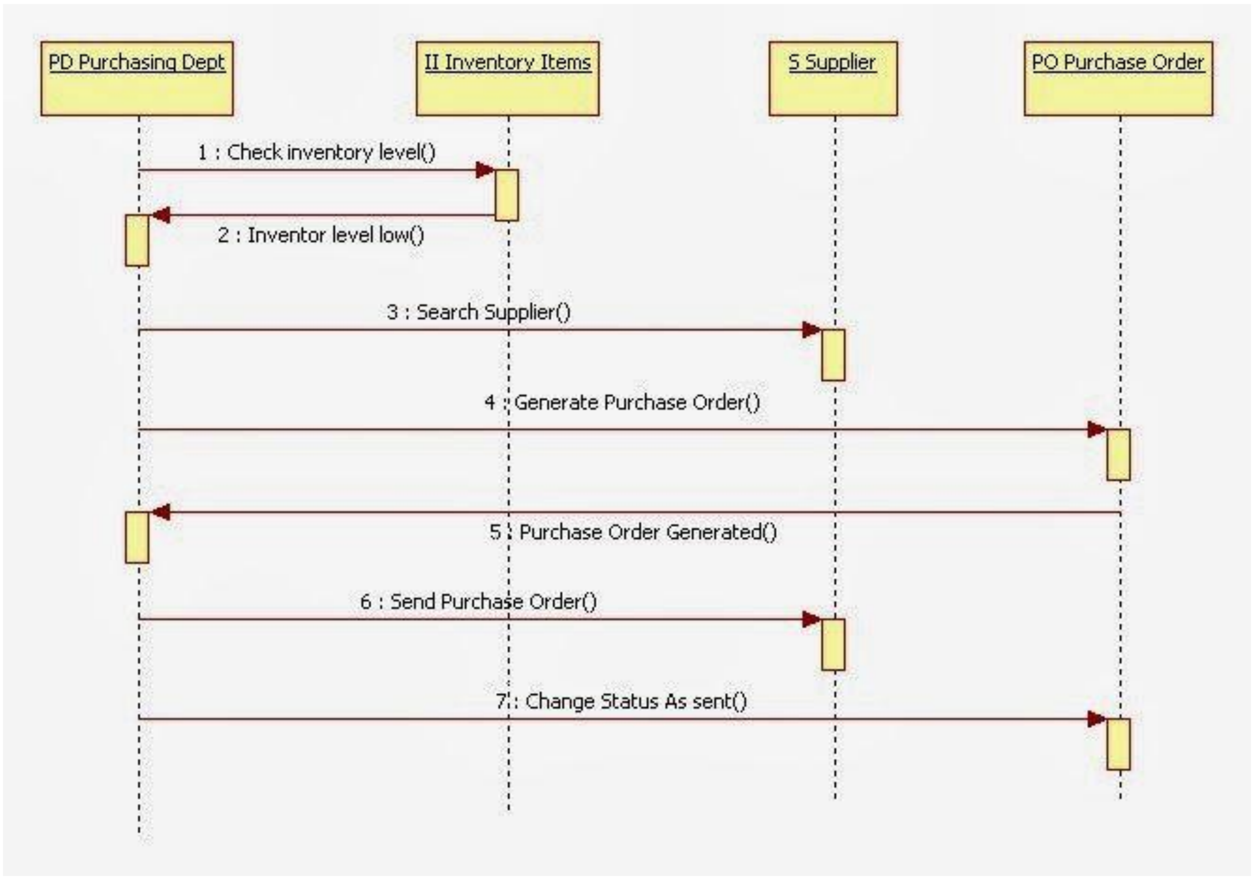
The state diagram describes the lifecycle of an item in the stock maintenance process. It begins when an item is requested, moving to the Available state where stock is monitored and displayed. If a customer reserves the item, the system transitions to the Reserved state, where inventory is locked and prepared for dispatch. Once payment is confirmed, the item moves to the Dispatched state, where shipment details are recorded and delivery status is tracked. A successful delivery shifts the item to the Delivered state, where the transaction is closed and the customer is notified. If the item is returned within the allowed period, it enters the Returned state for inspection and possible restocking.

Use case Diagram : Fig 4.3



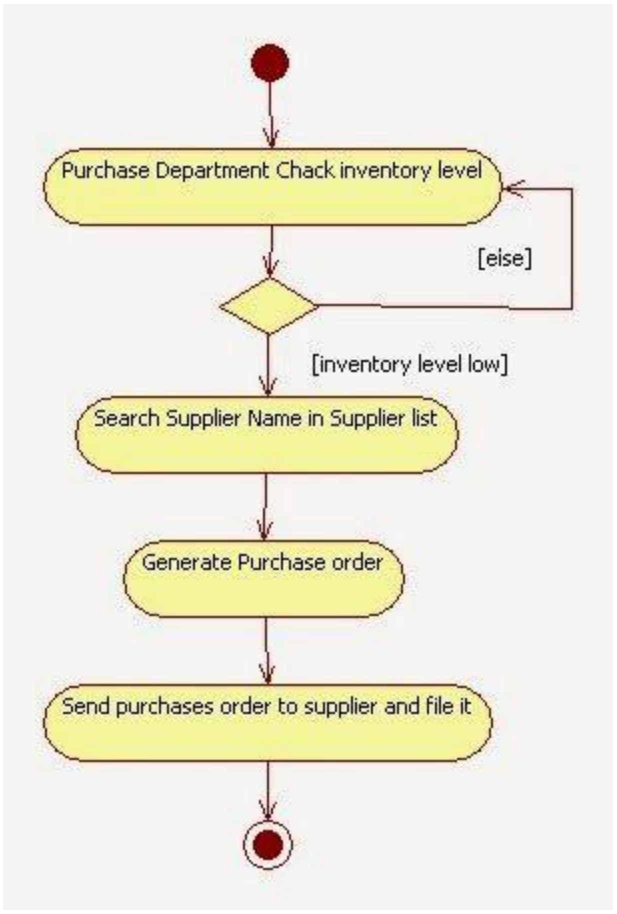
The use-case diagram shows how different users interact with the stock maintenance system to manage inventory, sales, and payments. The manager can generate sales reports, reorder depleted inventory, and update stock levels, all of which include checking the current inventory. The supplier provides inventory information to support stock updates. The cashier is responsible for scanning items, calculating the total bill, and completing the checkout process. Customers participate in payment activities and can pay by cheque, debit, cash, or credit. The system groups related actions through include relationships, showing how scanning items, checking inventory, and calculating tax are part of the checkout and payment processes. Overall, the diagram demonstrates how managers, suppliers, cashiers, and customers collaborate with the system to maintain accurate stock levels and complete sales transactions

Sequence Diagram : Fig 4.4


PD Purchasing Dept
II Inventory Items
S Supplier
PO Purchase Order
1 : Check inventory level()
2 : Inventor level low()
3 : Search Supplier()
4 : Generate Purchase Order()
5 : Purchase Order Generated()
6 : Send Purchase Order()
7 : Change Status As sent()

The sequence diagram illustrates how a purchase order is generated when stock levels become low. The process begins with the purchasing department checking inventory levels, after which the inventory system reports that the stock is low. The purchasing department then searches for an appropriate supplier, and once identified, instructs the system to generate a purchase order. After the purchase order is created, the system sends it to the supplier, and the purchasing order status is updated to indicate that it has been sent. The diagram clearly shows the interaction among the purchasing department, inventory system, supplier, and purchase order components to ensure timely restocking.

Activity Diagram : Fig 4.5

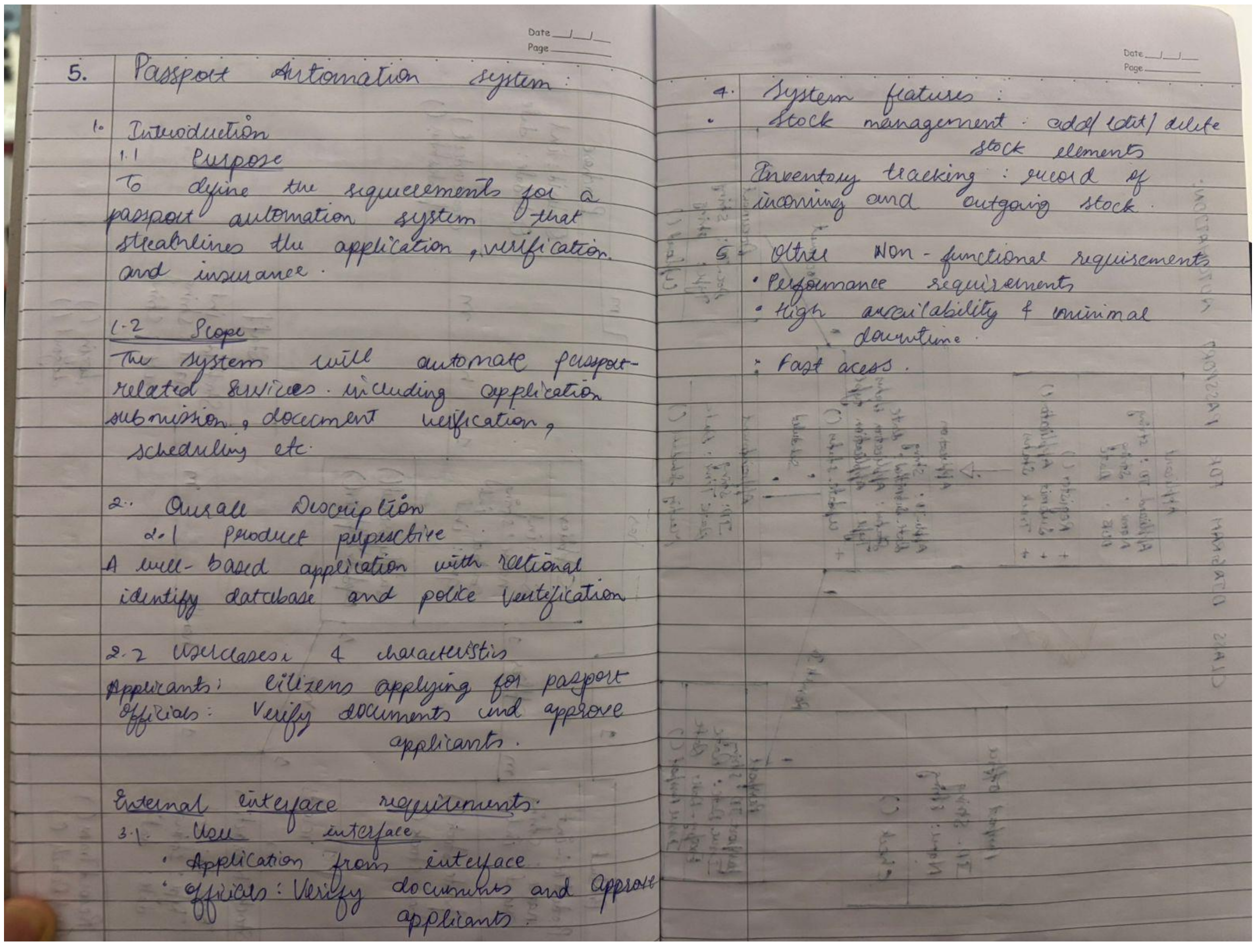

Purchase Department Chack inventory level
[eise]
[inventory level low]
Search Supplier Name in Supplier list
Generate Purchase order
Send purchases order to supplier and file it

The activity diagram illustrates the process followed by the purchasing department when stock levels run low. The workflow begins with checking the current inventory levels; if the stock is sufficient, the process ends. If the inventory level is low, the purchasing department searches for the appropriate supplier from the supplier list. Once the correct supplier is identified, the system generates a purchase order. Finally, the purchase order is sent to the supplier and filed for record-keeping, completing the restocking cycle. The diagram clearly shows decision-making based on stock availability and the steps taken to replenish inventory.

5. Passport Automation System

Problem Statement: The traditional passport application process involves long queues, manual verification, repeated form submissions, and delays caused by inefficient handling of applicant information. Managing records manually often leads to errors, misplaced documents, slow processing, and difficulty tracking application status. A Passport Automation System is needed to streamline the entire workflow by enabling users to apply online, upload required documents, schedule appointments, and track the progress of their applications. The system should support secure data handling, automated verification, faster processing, and seamless communication between applicants and passport officials. This will improve efficiency, reduce workload, minimize errors, and ensure timely passport issuance.

SRS:



5. Passport Automation system :

1. Introduction

1.1 Purpose

To define the requirements for a passport automation system that streamlines the application, verification and insurance.

1.2 Scope

The system will automate passport-related services including application submission, document verification, scheduling etc.

2. Overall Discription

2.1 Product perspective

A web-based application with national identify database and police verification

2.2 Userclasses & characteristics

Applicants: Citizens applying for passport

Officials: Verify documents and approve applicants.

External interface requirements.

3.1. User interface

- Application from interface
- Officials: Verify documents and Approve applicants.

4. System features :

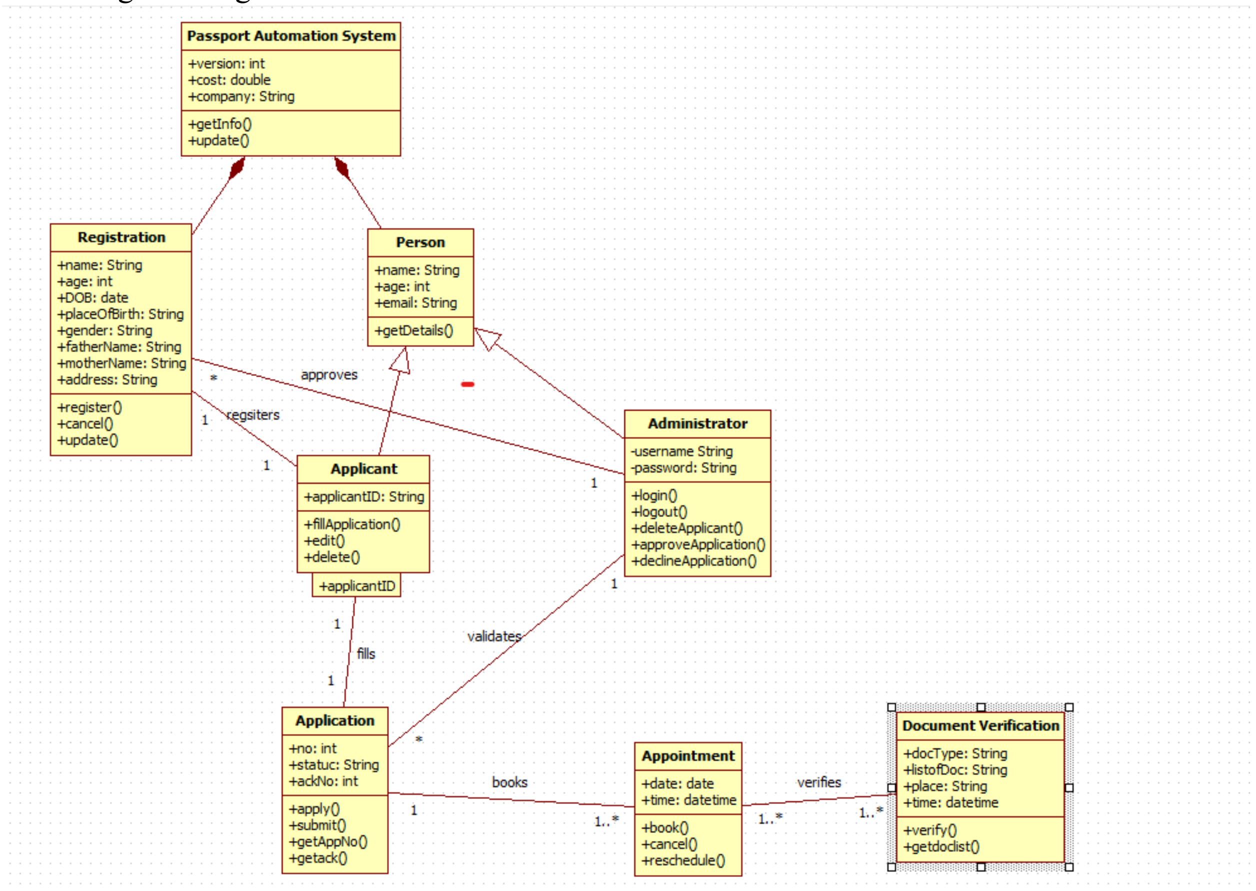
- Stock management : add/ edit/ delete stock elements

Inventory tracking : record of incoming and outgoing stock.

5. Other Non-functional requirements

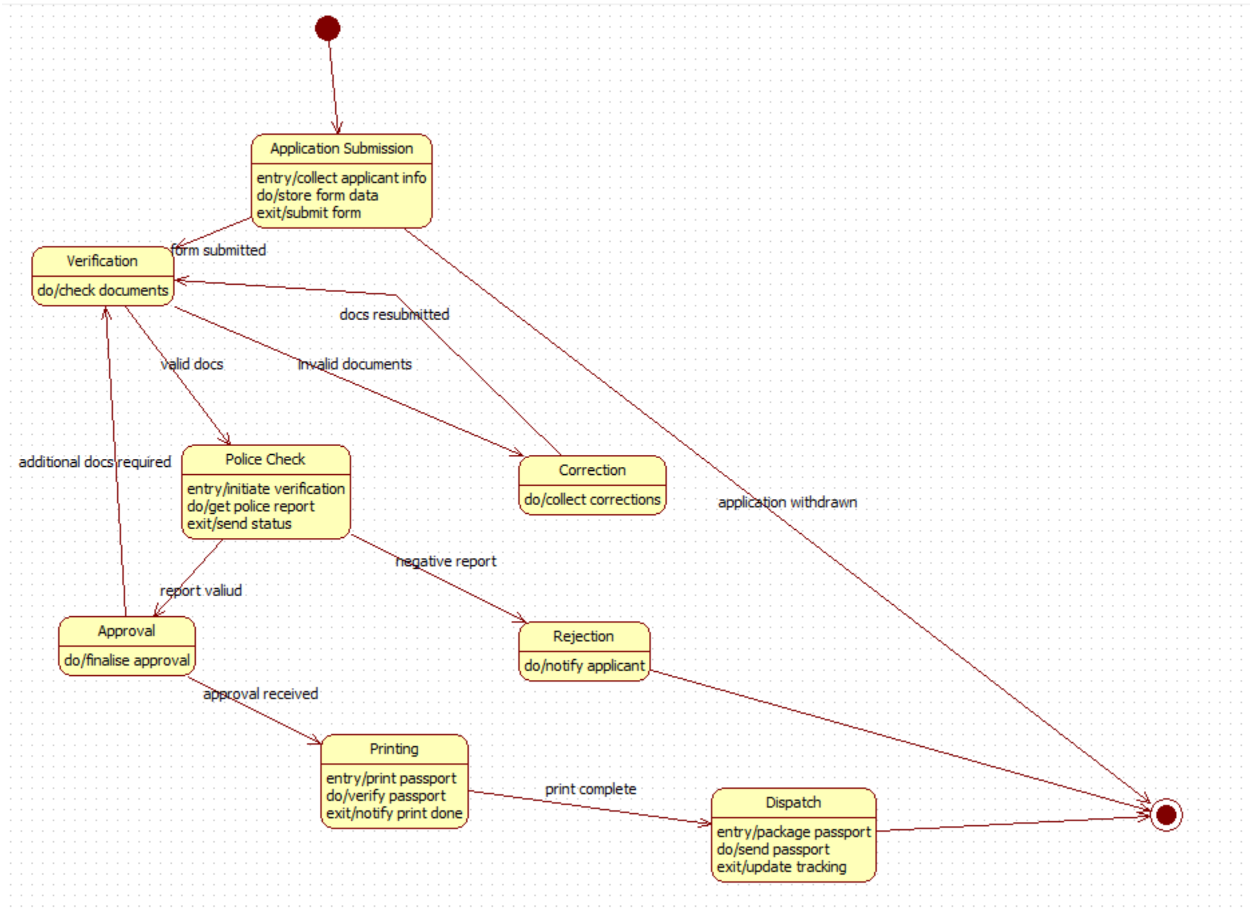
- Performance requirements
- High availability & minimal downtime.
- Fast acess.

Class diagram : Fig 5.1



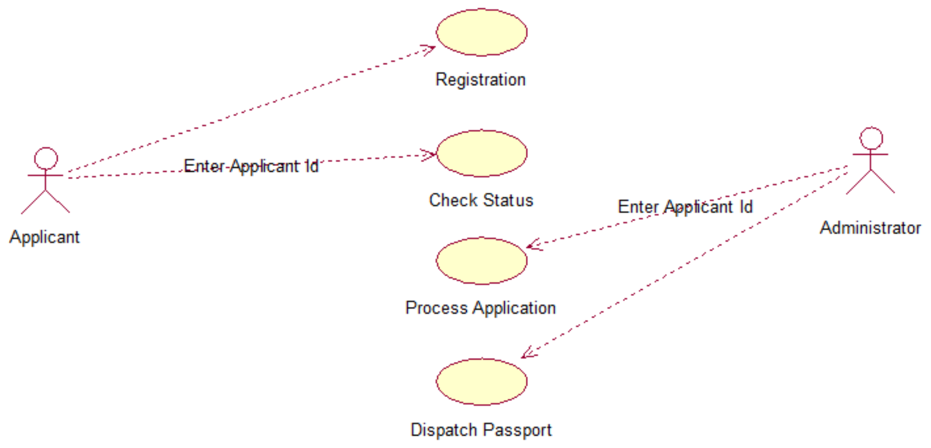
The class diagram for the Passport Automation System illustrates the major components involved in managing passport applications and their interactions. The Person class stores basic personal details and is used by the Registration process to record applicant information. Once registered, an Applicant can fill, edit, or delete passport applications, which are represented by the Application class containing application number, status, and application details. Applicants can book or reschedule appointments through the Appointment class, while their documents are verified using the Document Verification class, which records document type, list, place, and time. The Administrator oversees the system by validating applicants, approving or rejecting applications, and managing user accounts. The main Passport Automation System class interacts with these components by providing system information and updates. Overall, the diagram shows how registration, application submission, appointment scheduling, verification, and approval processes are structured and interconnected within the system.

State Diagram : Fig 5.2



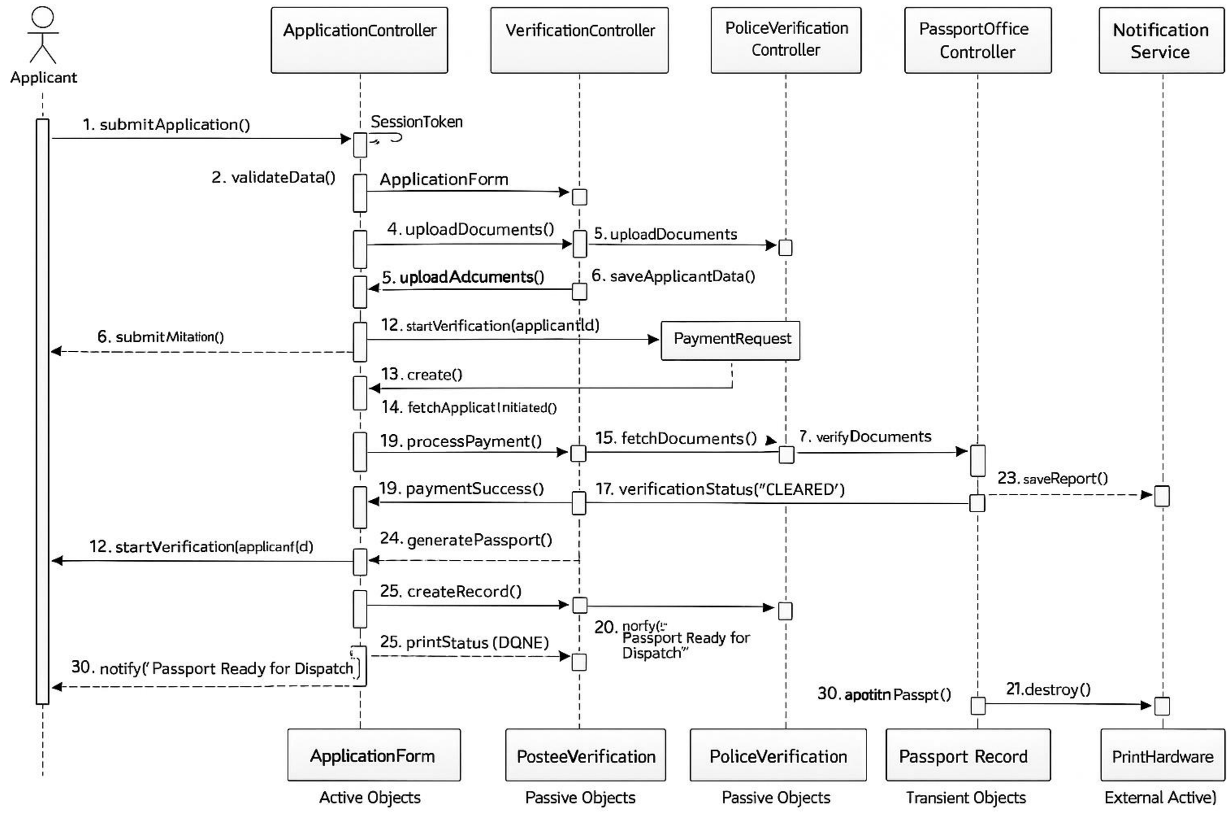
The state diagram illustrates the complete lifecycle of a passport application, beginning with the submission of the applicant’s form. After submission, the application enters the verification state, where documents are checked and either accepted, rejected, or sent back for corrections. Valid documents move the application to the police check state, where background verification is performed. A positive police report leads to the approval state, while a negative report results in rejection, notifying the applicant. Once approved, the application transitions to the printing state, where the passport is printed and verified. After printing is complete, the process moves to the dispatch state, where the passport is packaged, sent to the applicant, and tracking information is updated. The diagram shows how applications may loop back for corrections or terminate early if withdrawn or rejected, clearly representing all key stages from submission to final dispatch

Use Case Diagram : Fig 5.3



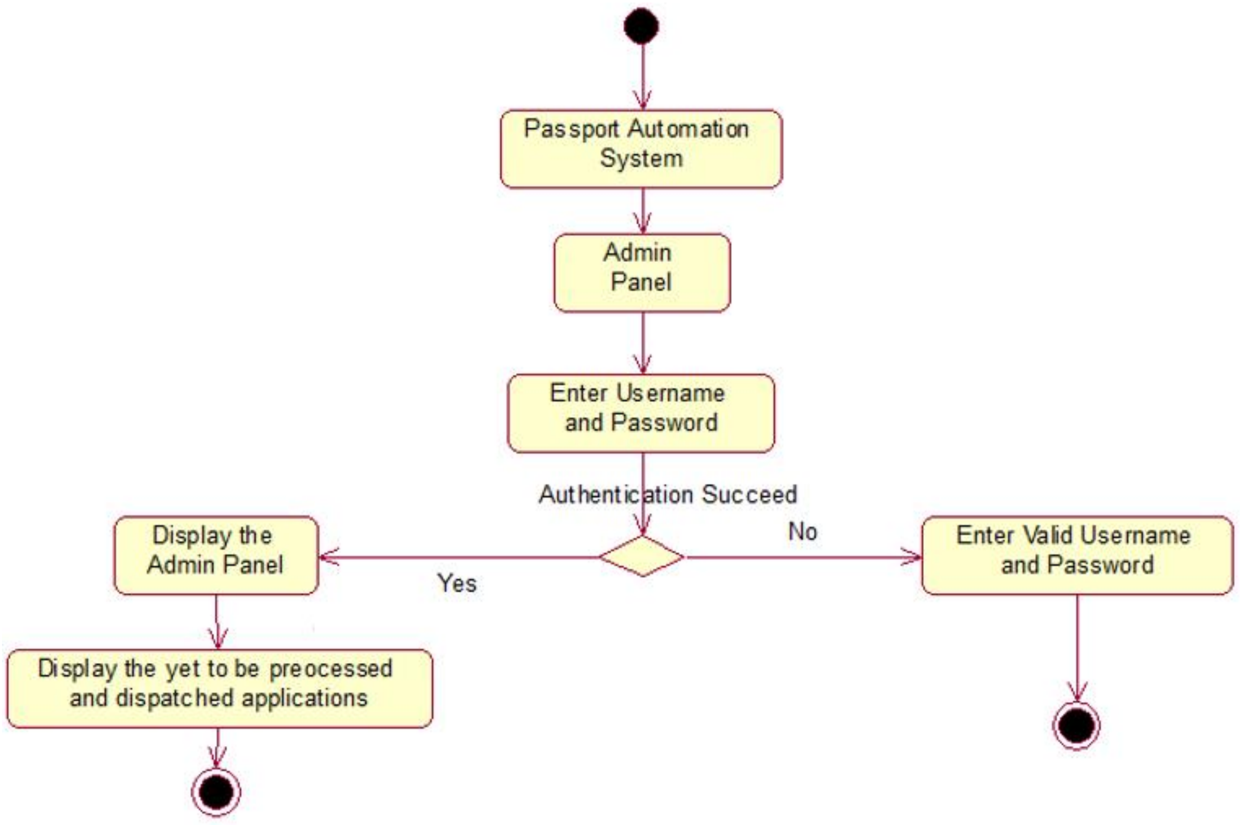
The use-case diagram shows the interaction between the applicant and the administrator during the passport application process. The applicant can register in the system, enter their applicant ID to check the status of their application, and eventually receive their dispatched passport. The administrator enters the applicant ID to access application details, processes the application, and handles the dispatch of the passport once it is approved. The diagram highlights the key actions performed by both users and demonstrates how applicant information flows through registration, processing, status checking, and final dispatch.

Sequence Diagram : Fig 5.4



The sequence diagram shows the complete workflow of the Passport Automation System. The applicant submits the application, after which the ApplicationController validates data, uploads documents, and processes payment using a temporary PaymentRequest. Once payment succeeds, the VerificationController checks the applicant's details and documents, and the PoliceVerificationController performs a background check and stores the PoliceReport. When verification is cleared, the PassportOfficeController generates the passport, creates the PassportRecord, and prints it through a transient print job. Finally, the NotificationService informs the applicant that the passport is ready.

Activity Diagram : Fig 5.5


Passport Automation System
Admin Panel
Enter Username and Password
Authentication Succeed
No
Yes
Display the Admin Panel
Enter Valid Username and Password
Display the yet to be preocessed and dispatched applications

The activity diagram illustrates the admin login workflow in the Passport Automation System. The process begins when the admin enters the system and navigates to the Admin Panel. The admin is then prompted to enter a username and password. A decision is made based on whether the authentication succeeds or fails. If authentication fails, the admin is redirected to re-enter valid login credentials. If authentication succeeds, the system displays the Admin Panel, where the admin can view all pending passport applications that are yet to be processed or dispatched. The diagram ends after displaying the list of pending applications.