

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:

Dama Yohitesh Naveen Sai

(1BM23CS093)

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 **Dama Yohitesh Naveen Sai (1BM23CS085)** during the 5th Semester August 2025-December 2025

Faculty Incharge: Dr Pallavi GB

Batch Incharge : Sonika Sharma

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

Table of Contents

1. Hotel Management System
2. Credit Card Processing
3. Library Management System
4. Stock Maintenance System
5. Passport Automation System

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.

LAB -1

DATE: 18/8/25 PAGE: ①

⇒ Hotel Management System

Software Requirements Specification (SRS)

1. Introduction

1.1) Purpose:

- To provide an automated platform for managing hotel operations.
- Like room booking, check-in/out, billing and staff management.
- It aims to improve the efficiency of hotel staff and enhance user experience.

1.2) Scope:

- The system will handle room reservations, guest check-in/out, payment processing, room service requests and generate reports.
- Intended for use by the hotel staff and management.

1.3) Definitions and Abbreviations:

- HMS : Hotel Management System
- Admin : User with management rights
- Guest : Customer staying or booking rooms
- Booking : Reservation of a room for specific date.

1.4) Intended Audience and Reading Suggestions:

- Developers : To understand the technical requirements and specifications
- Project Managers : for overall project planning and resource allocation
- Hotel Administrators : To understand system functionalities for hotel operations.
- End-users : for understanding the system from a usability perspective.

1.5) References :-

- ISO/IEC 9126 - Software Engineering - Product quality
- IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications
- Hotel Management System Project Documentation
 - Internal documents or sources that can guide further system customization.

2. Overall Description :-

2.1) Product Perspective:-

The HMS is a standalone web-based/mobile application designed to replace manual hotel management methods.

2.2) Product Features:-

- User authentication and role-based access control.
- Room inventory management including availability and maintenance tracking.
- Booking creation, modification, cancellation and confirmation
- Guest check in and out with real-time room status update
- Billing module to generate invoices and process payments
- Reporting tools for occupancy, revenue and staff performance analysis
- Optional online booking interface for guests.

2.3) User classes and Characteristics:-

- Admin: Manages hotel resources, staff, reports and system settings
- Receptionist: Handles bookings, check ins/outs and guest services
- Guests: Views room availability and makes bookings.

2.4) Operating Environment:-

The system will operate on Windows/Linux servers with a web interface accessible through modern browsers and mobile devices.

2.5) Design and Implementation Constraints:

- The system should ensure data security and privacy.

- It should be scalable to accommodate multiple hotel branches.

2.6) User documentation:

- Online help guides and FAQs for system users.

- Step-by-step manuals for common tasks such as booking and billing.

- Training videos and quick-start tutorials for new staff.

2.7) Assumptions and Dependencies:

- Internet connectivity is required for online features.

- Payment gateways must support standard APIs for integration.

- Users have basic computer literacy.

- The system depends on reliable server infrastructure for uptime.

3. Specific Requirements:

3.1) Functional Requirements:

• User Authentication:

- ↳ Users must login using an username and password.

- ↳ Roles determine access rights.

• Room Management:

- ↳ Admin can add, edit or delete room details.

- ↳ Rooms have statuses (Available, booked, occupied, servicing).

• Booking Management:

- ↳ Guests/receptionists can create, update and cancel bookings.

- ↳ System checks room availability before confirming booking.

• Check-in / Check-out:

- ↳ Receptionists can check guests in and out.

- ↳ System updates room status accordingly.

• Billing and Payment:

- ↳ System generates bills including room charges and additional services.

3.2) System Features:

- Real time room Status tracking
- Reservation calendar
- Multi user access at a time
- Alerts and Notifications
- Data backup and Recovery to prevent data loss
- Audit trails for security and accountability.

3.3) Non-functional Requirements:

- Performance
- Security
- Usability (user friendly)
- Reliability (backup & recovery features)
- Maintainability

3.4) External Interface Requirements

- User interfaces
 - ↳ web based interface with dashboards for admin and user
- Hardware interfaces
 - ↳ integration with card payment
- Software interfaces

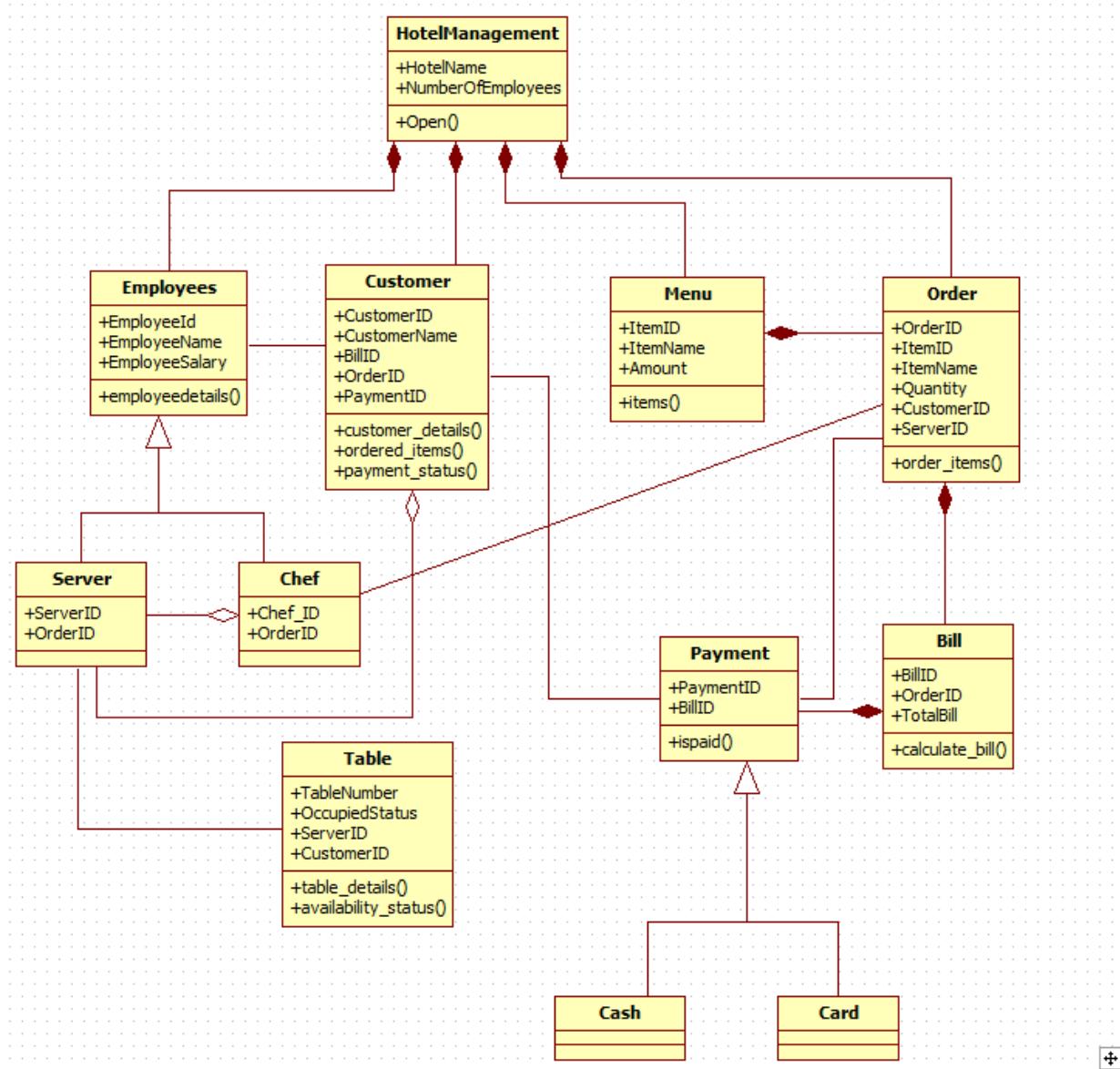
3.5) Other Requirements:

- Compliance with data protection laws
- Backup for all ~~plans~~ critical data daily.

4. Appendix:

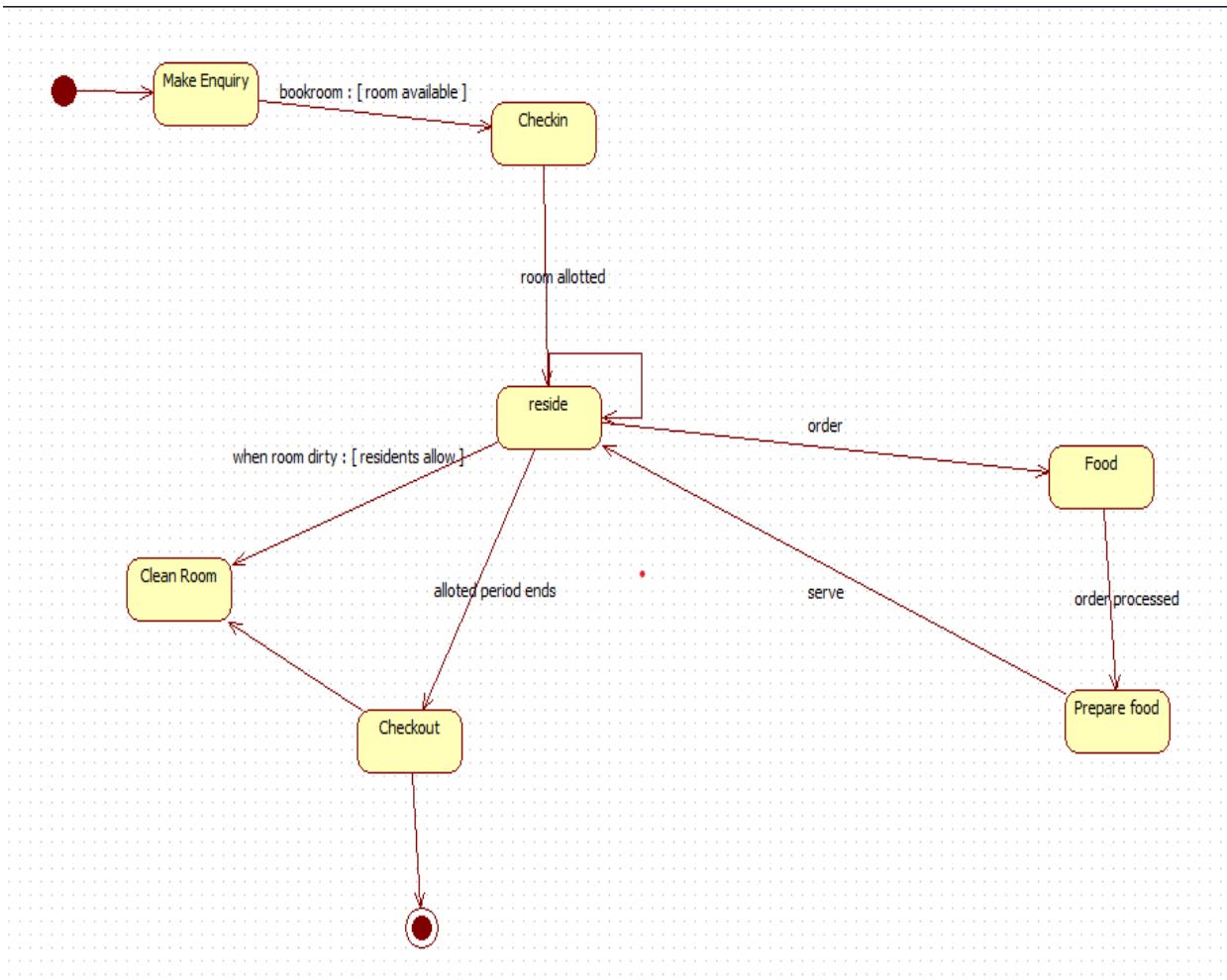
- HMS : Hotel Management System
- UI : User interface
- MFA : Multi-factor Authentication
- API : Application programming interface
- GDPR : General Data Protection regulation

Class Diagram: fig 1.1



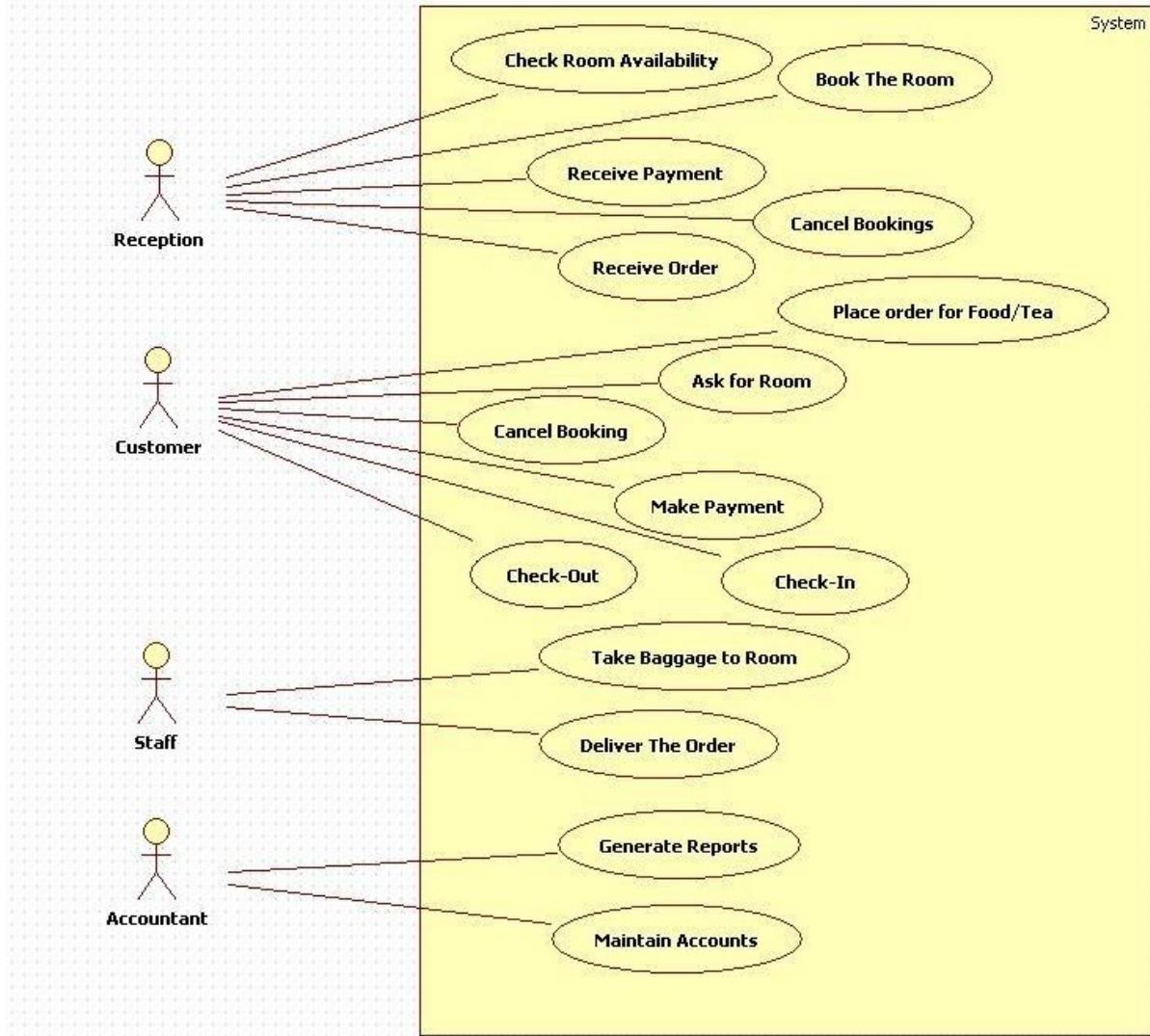
The class diagram of the Hotel Management System illustrates the main entities involved in hotel operations and how they interact. The central **HotelManagement** class is linked to major components such as **Employees**, **Customer**, **Menu**, **Order**, **Bill**, and **Payment**. **Employees** are 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

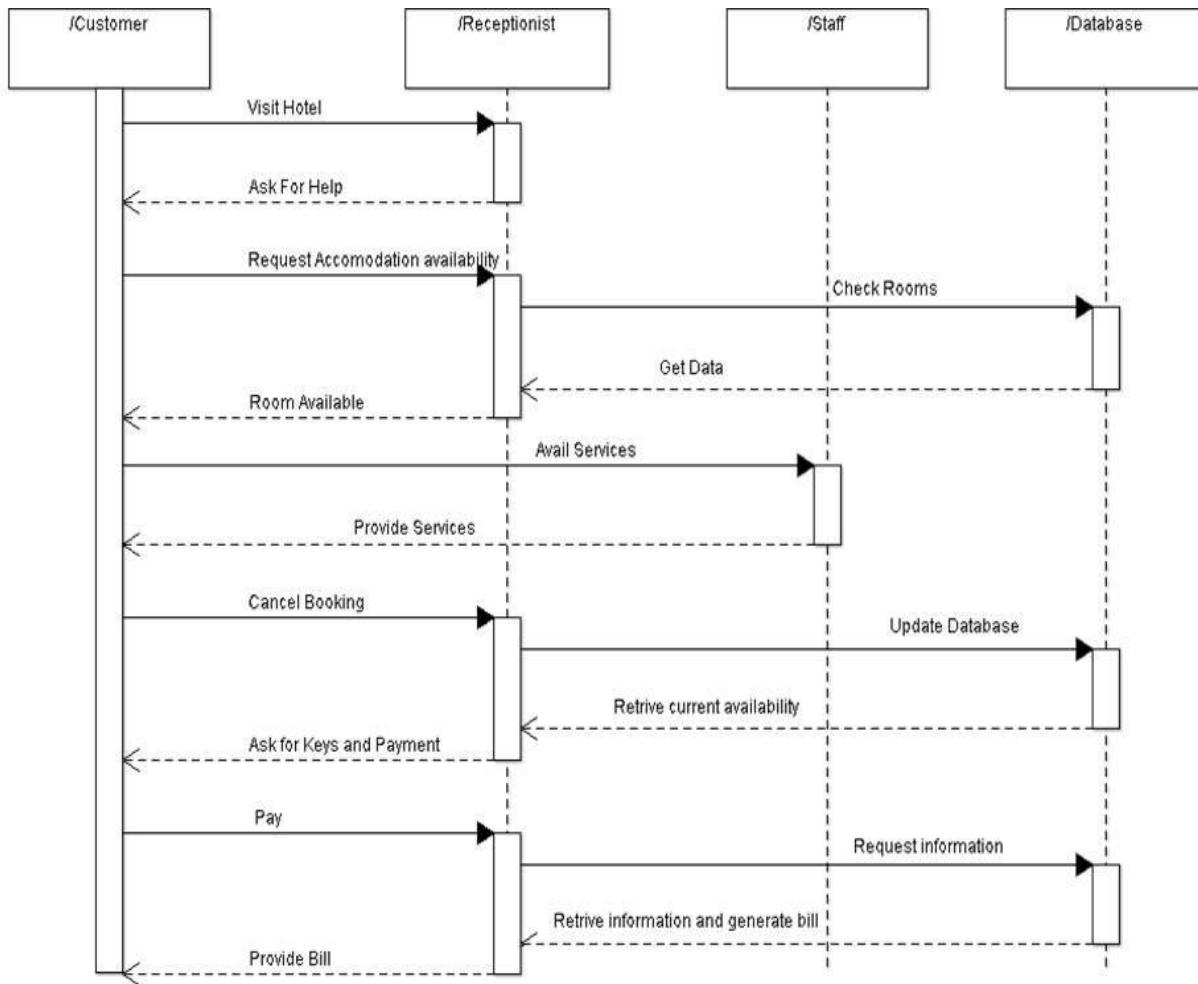
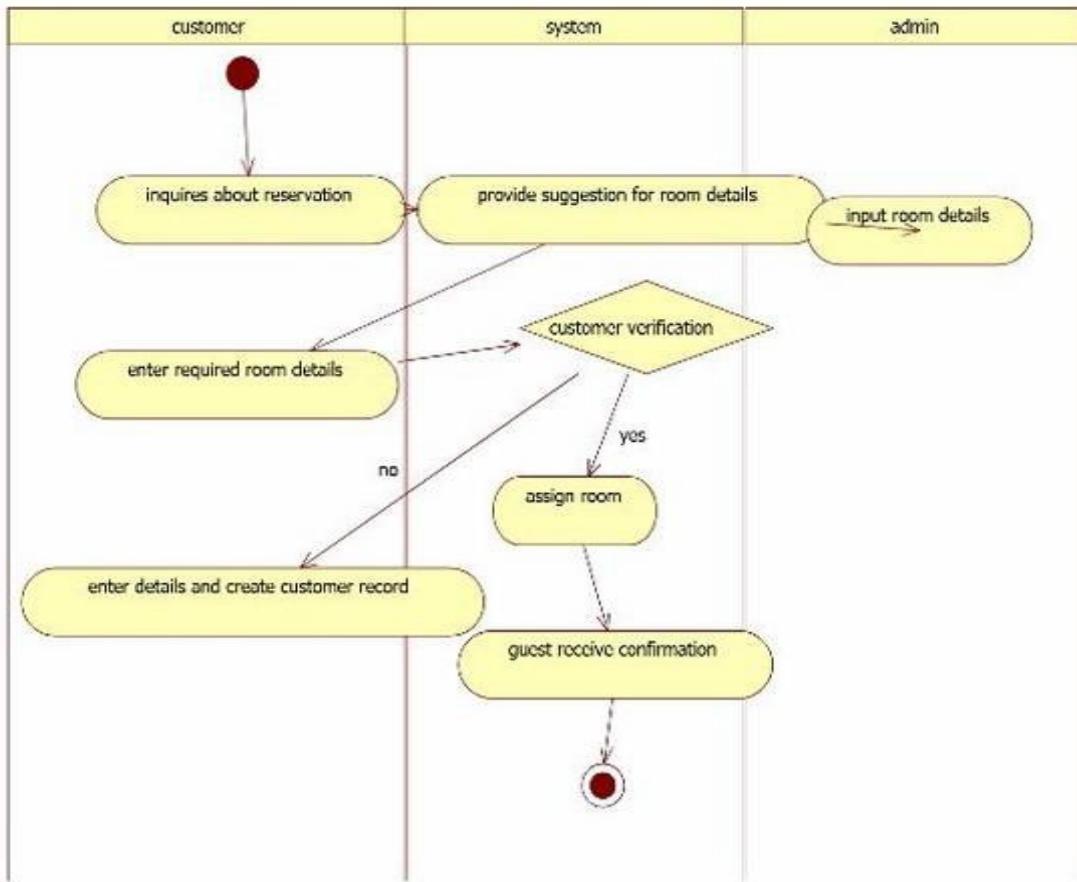


fig 1.5

Advanced Activity Diagram



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: (5)

⇒ Credit card Processing system SRS :

1. Introduction:

1.1) Purpose
System automates the authorization, billing and transaction recording of credit card payments securely and efficiently.

1.2) Sope:
The system allows user to make payments, validate credit card data, process transactions with secure authentication and maintain log of all operations.

1.3) Definitions:

- CCPS : Credit card Processing System
- CVV : Card Verification Value
- FR : Functional Requirements
- NFR : Non-functional Requirements

2. Overall Description:

2.1) Product Perspective:
A web service or component integrated into e-commerce or banking systems.
Acts as an intermediary between merchant and bank.

2.2) Product functions:

- Accept card information
- Validate card details and CVV
- Authorize transactions with bank
- Generate transaction receipts

2.3) User Classes:

- Customer: Input card details
- Merchant: Receive transaction results
- Admin: Manage logs and exports

2.4) Constraints:

- Must comply with PCI-DSS
- HTTPS required for all communication
- Data must be encrypted

3. System Requirements:

3.1) Functional Requirements (FR):

- FR1: Accept credit card
- FR2: Validate card details using Luhn algorithm
- FR3: Communicate with bank gateway API
- FR4: Confirm queue or failure of transactions.
- FR5: Generate digital receipt and store transaction log

3.2) Non Functional Requirements (NFR):

- NFR1: Transaction processed < 8 sec
- NFR2: 99.99% uptime for transaction services
- NFR3: All data must be encrypted (AES-256)

3.3) Domain Requirements:

- Must support Visa, Mastercard and AMEX
- Must flag and log suspicious transactions
- Fraud detection based on rule engine.

3.4) External Interfaces:

- UI: Payment form (card number, expiry, CVV)
- Software: Integration with payment gateways (Razorpay, stripe)
- Communication: REST API over HTTPS

3.5) Other Requirements:

- System must log all transactions securely
- System should support roll back for failed transactions
- Daily summary report for admins

4. Appendices:

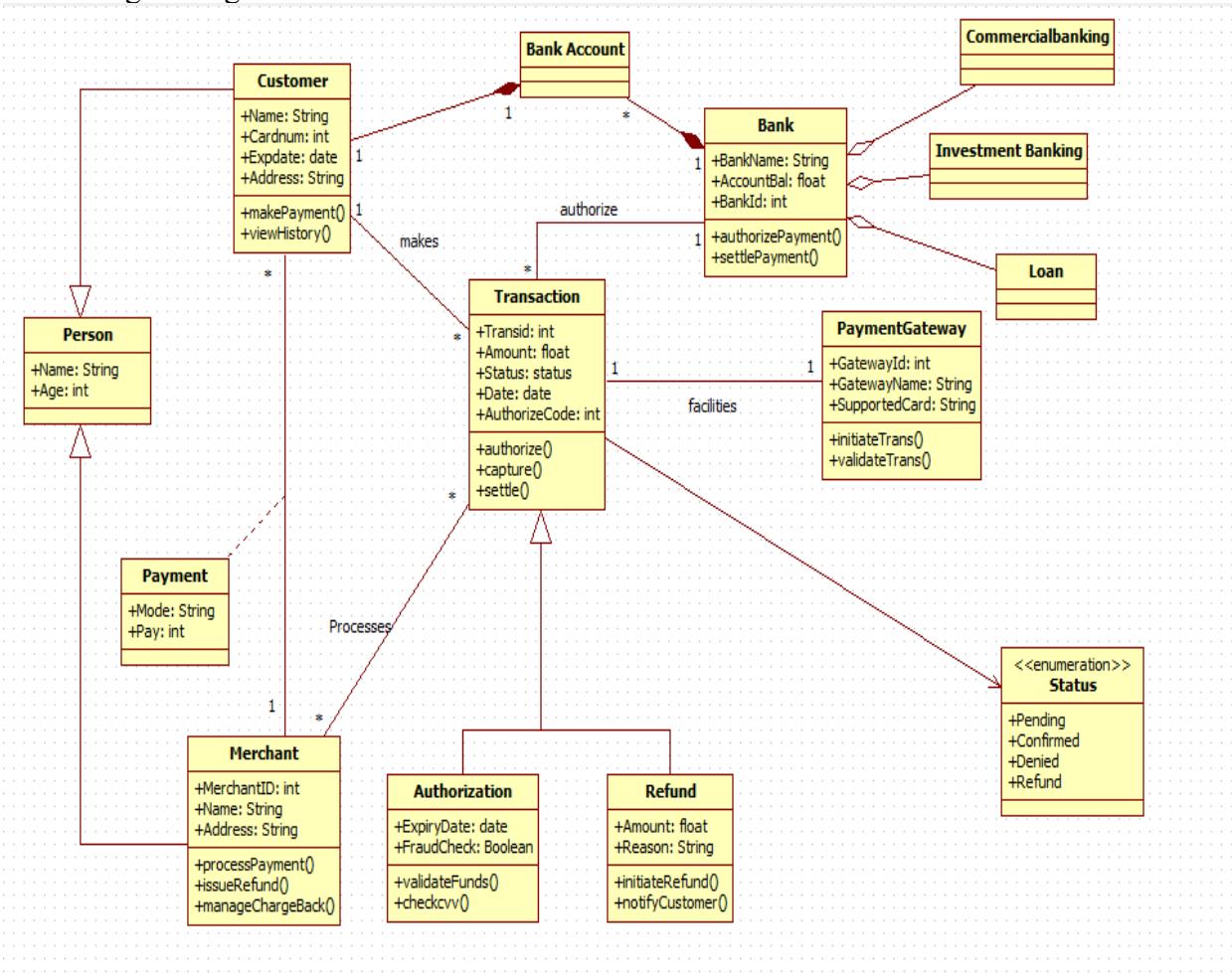
A: Example transaction log format

B: Simplified system architecture diagram

C: Sample fraud detection rule (multiple failed attempts)

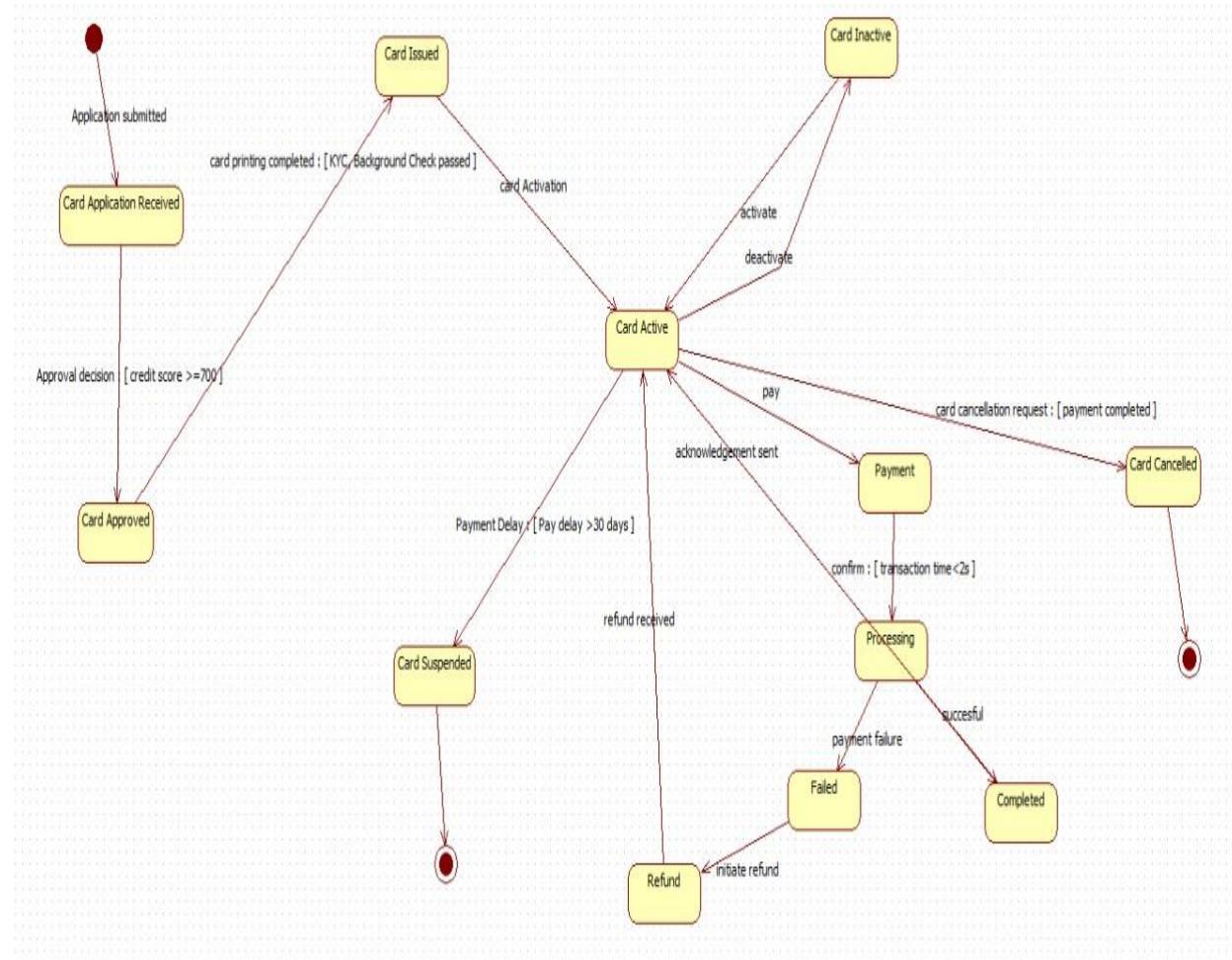
Nalinie
19/8/25

Class Diagram Fig:2.1



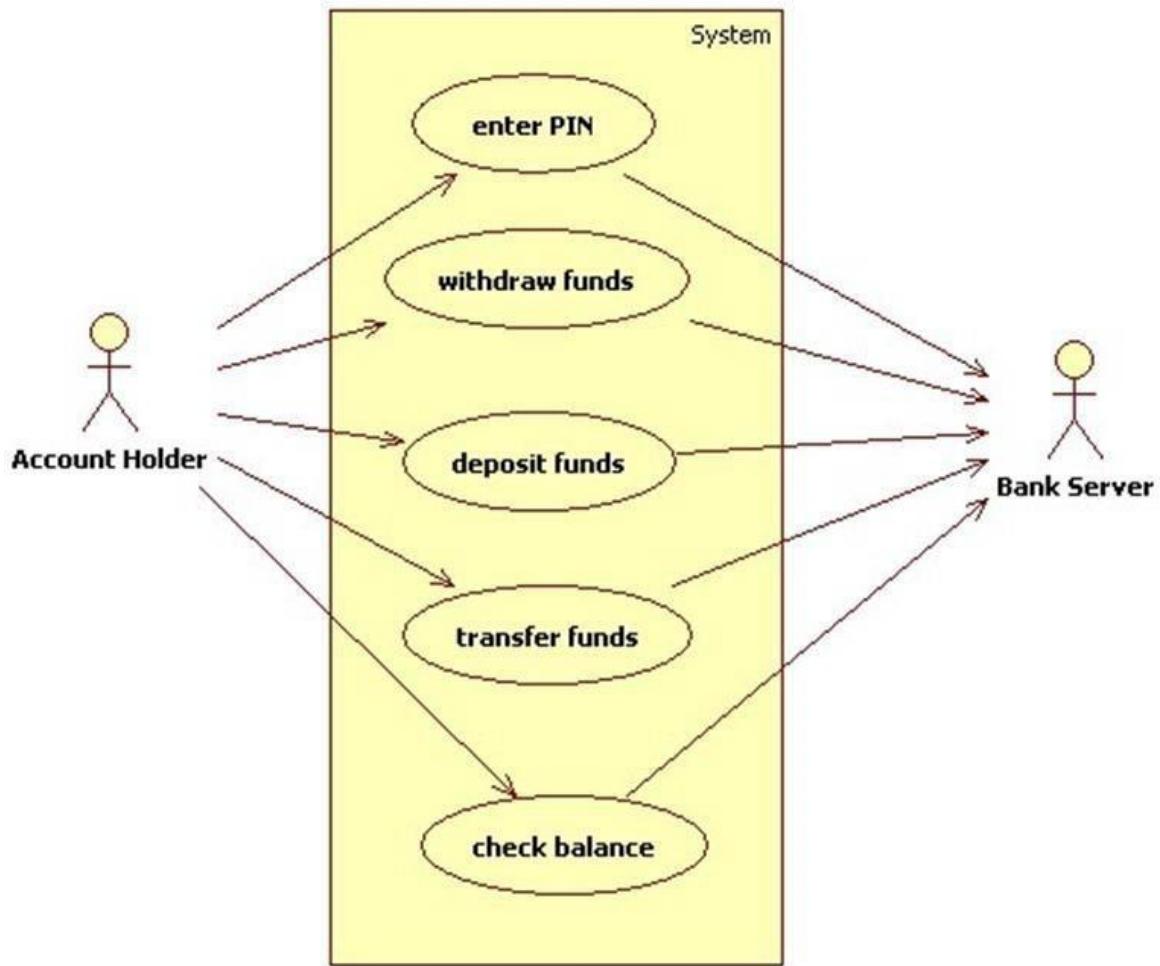
The class diagram for the Credit Card Processing System shows how customers, merchants, banks, and payment gateways interact to process credit card transactions. A customer, inheriting details from a general person class, initiates payments that create transaction objects containing information such 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



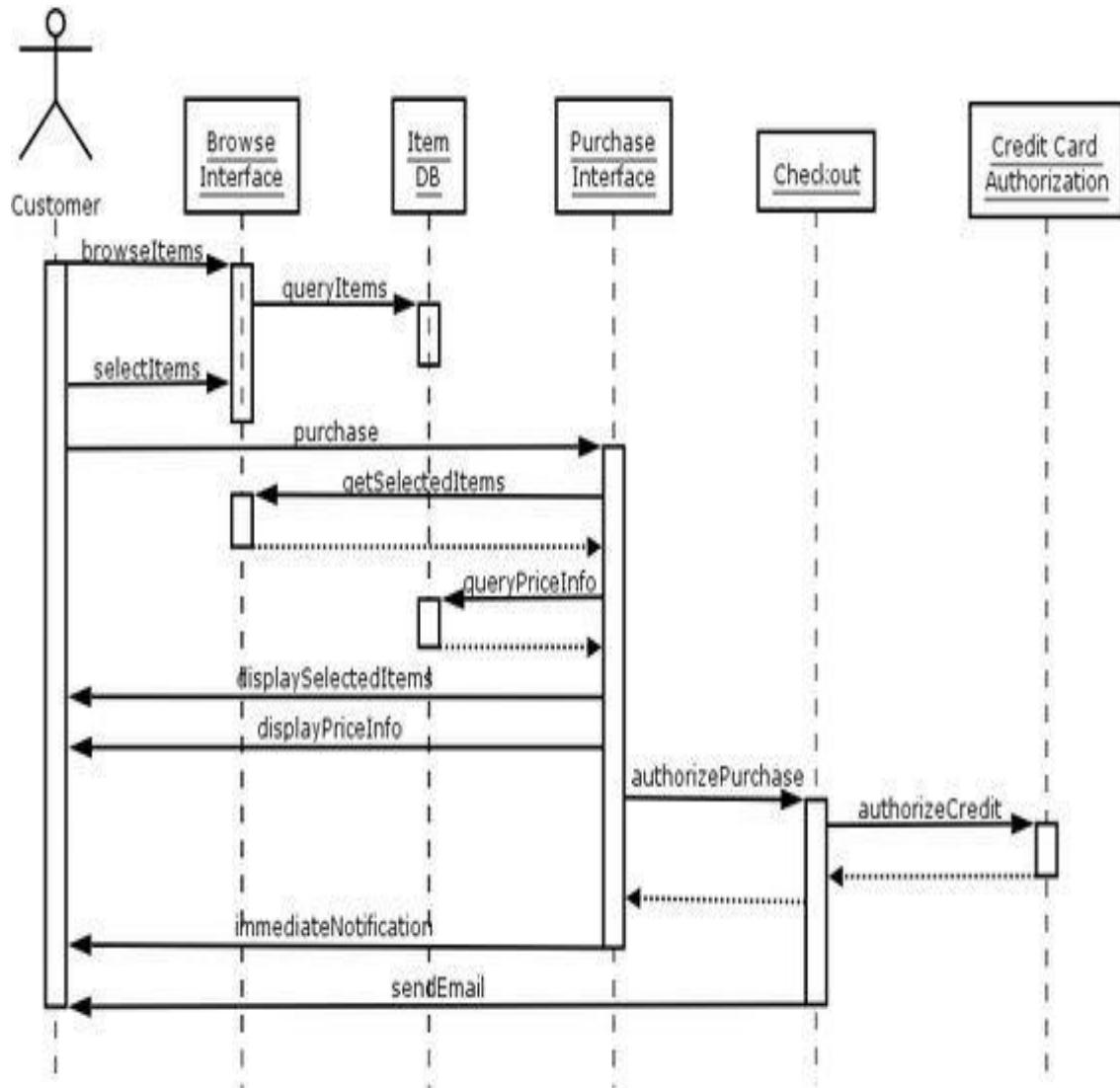
The state diagram shows the complete lifecycle of a credit card, beginning with the application submitted and moving to the card application received and approved states. Once approved and printed, the card enters the card issued state and becomes card active after activation. In the active state, the 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

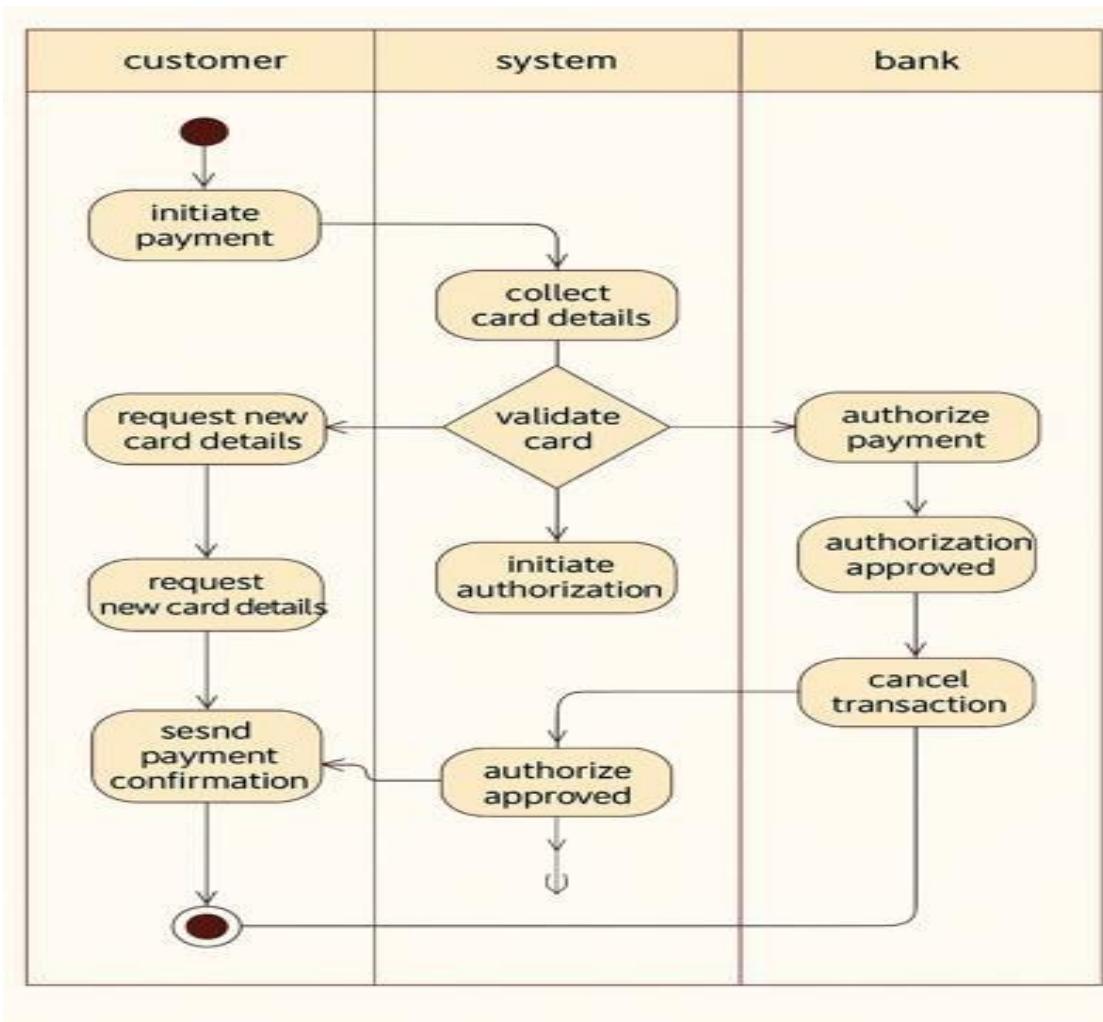


The use case diagram shows how an account holder interacts with the system to perform basic banking operations, such as entering a PIN, withdrawing funds, depositing money, transferring funds, and checking account balance. Each of these actions requires communication with the bank server, which verifies 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



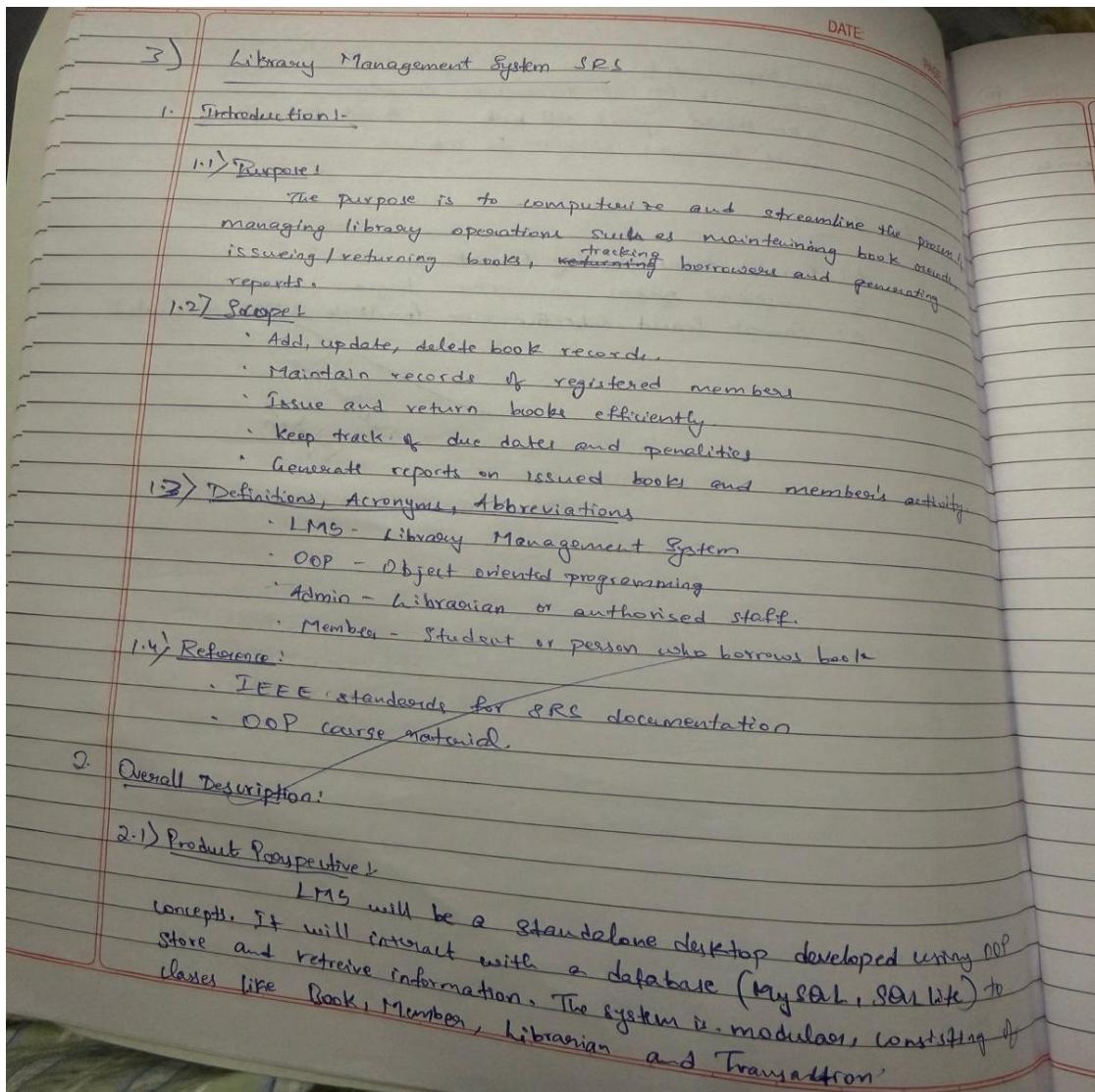
Activity Diagram: Fig 2.5



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.



2.2) Product Functions:

- User Authentication - login for librarians and members
- ' Book Management - add, delete, edit records and search books.
- ' Member Management - Register, update and delete members.
- ' Transaction Management - Issue and return books, track dues and fines.
- ' Reports - Generate summaries of library usage, overdue books, active members.

2.3) User Characteristics:

- ' Librarians / Admins - Basic computer knowledge, responsible for maintaining records.
- ' Members / Students - can search, borrow and return books.

2.4) Constraints:

- System should work on Windows / Linux environment
- ' Database size limited to institutional needs
- ' Multi user Concurrency support required.

2.5) Assumptions and Dependencies

- ' Users have basic computer literacy
- ' Reliable power and internet
- ' Proper database connectivity and hardware equipment.

3. System Features:

3.1) Functional Requirements:

- User Authentication - UID, password
- ' Book Management - Add, edit, delete, search
- ' Member Management - Add, delete, search
- ' Transaction Management - Borrow, return, fine
- ' Reports and Notifications - Due date, etc.

3.2) Non-functional Requirements

- Performance - results within 2 sec for over 1000 records
- Security
- Reliability and Availability
- Usability
- Portability

3.3) Domain Requirements:

- Library Rules and Policies
- Book Categorization
- Member Classification
- Transaction Policies.

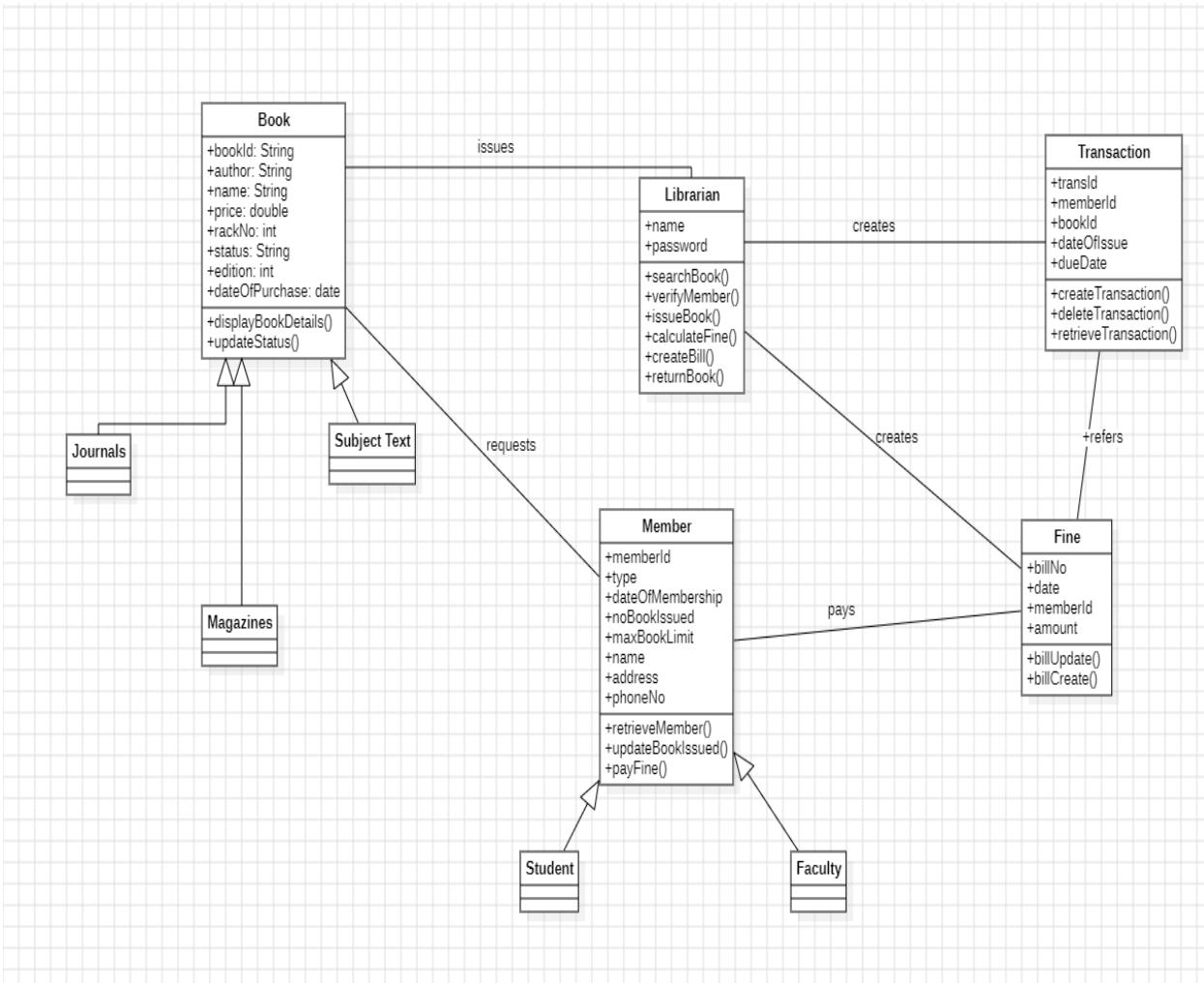
3.4) External Interfaces

- User interfaces
- Hardware interfaces
- Software interfaces
- Communications interfaces

4. Appendix

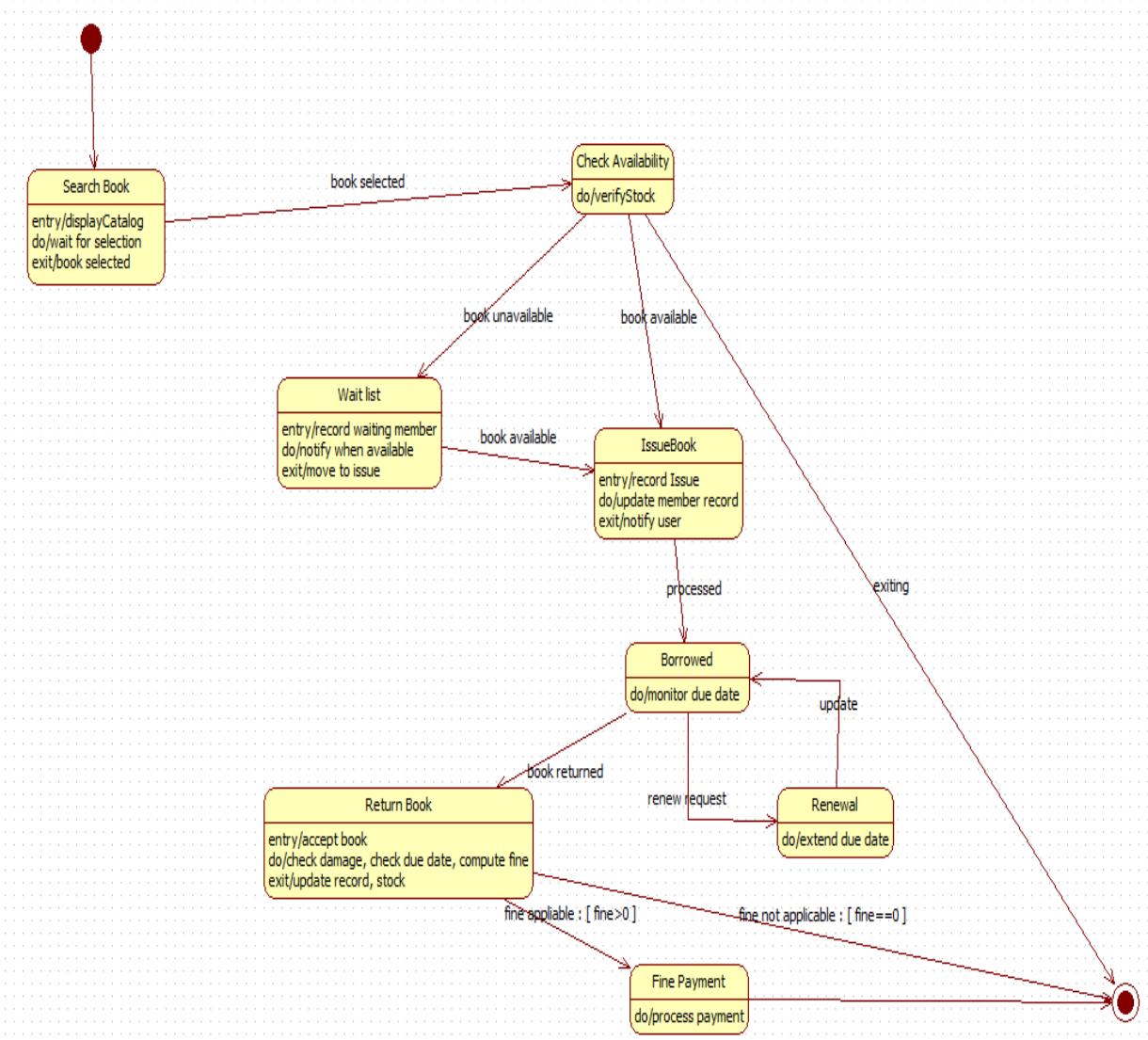
Future scope : Integration with e-books, online payments for fines, mobile app support.

Class Diagram : Fig 3.1



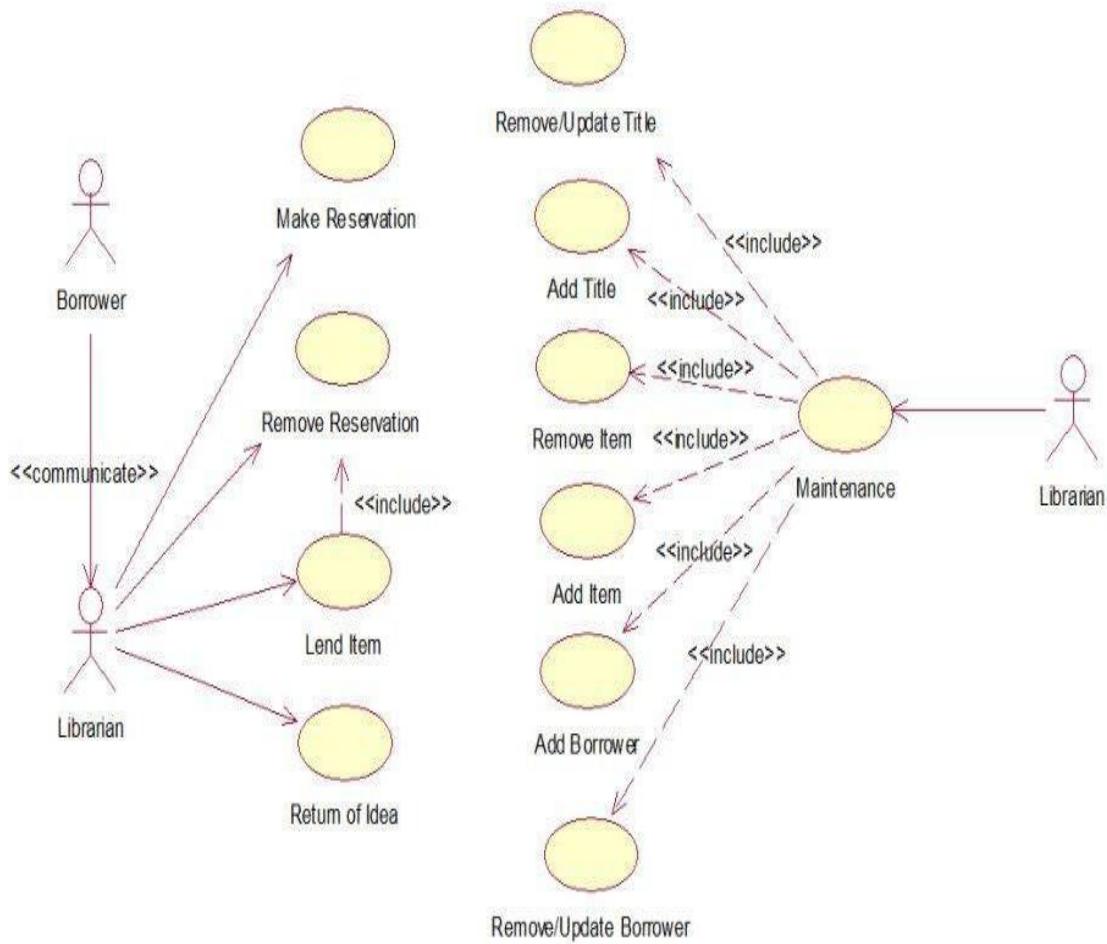
The class diagram for the Library Management System shows how books, members, librarians, transactions, and fines interact within the system. The Book class stores details about each book and is specialized into journals, magazines, and subject texts through inheritance. Members request books and may 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



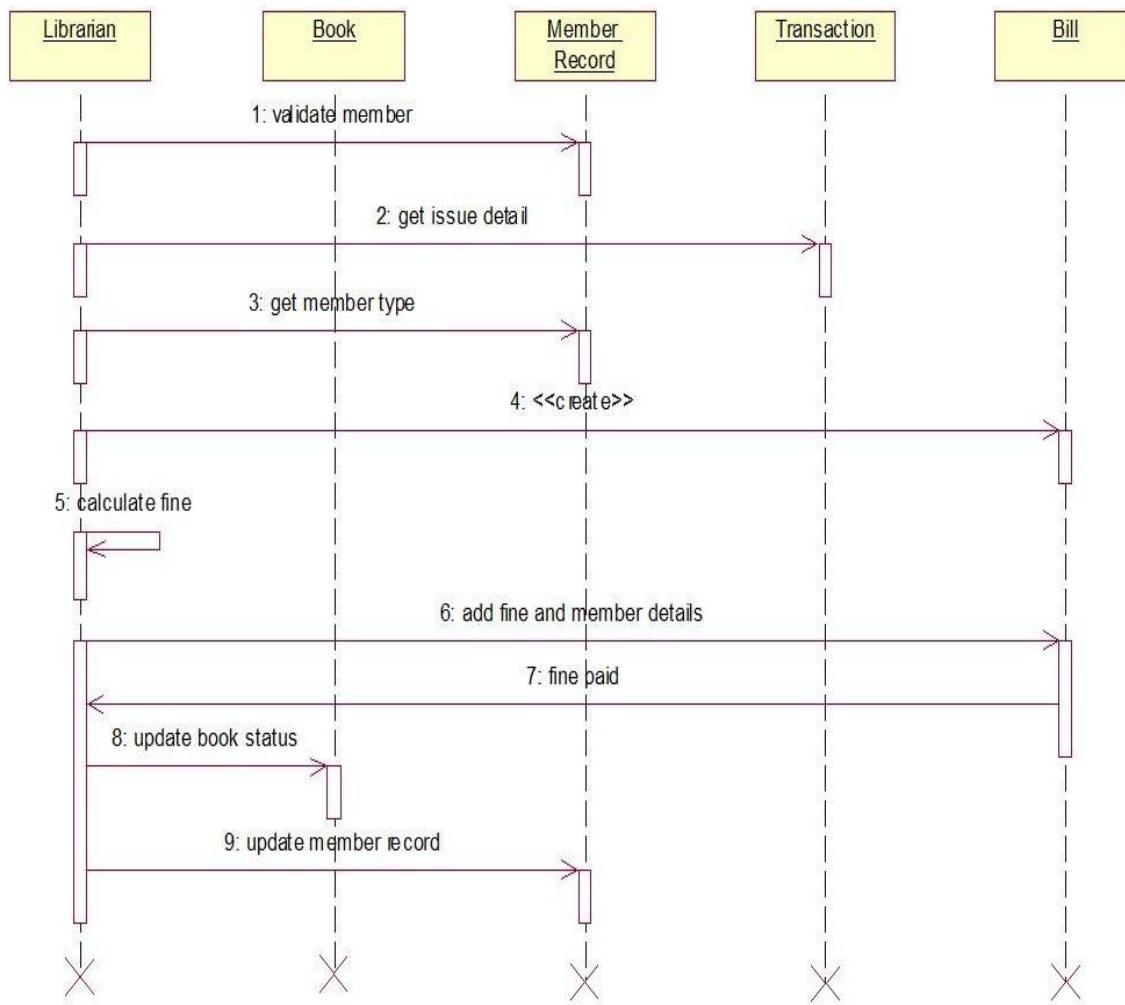
The state diagram shows the different stages a book goes through during the library borrowing process. It begins with the user searching for a book, after which the system checks its availability. If the book is unavailable, the request moves to the wait list state until a copy becomes available. When 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



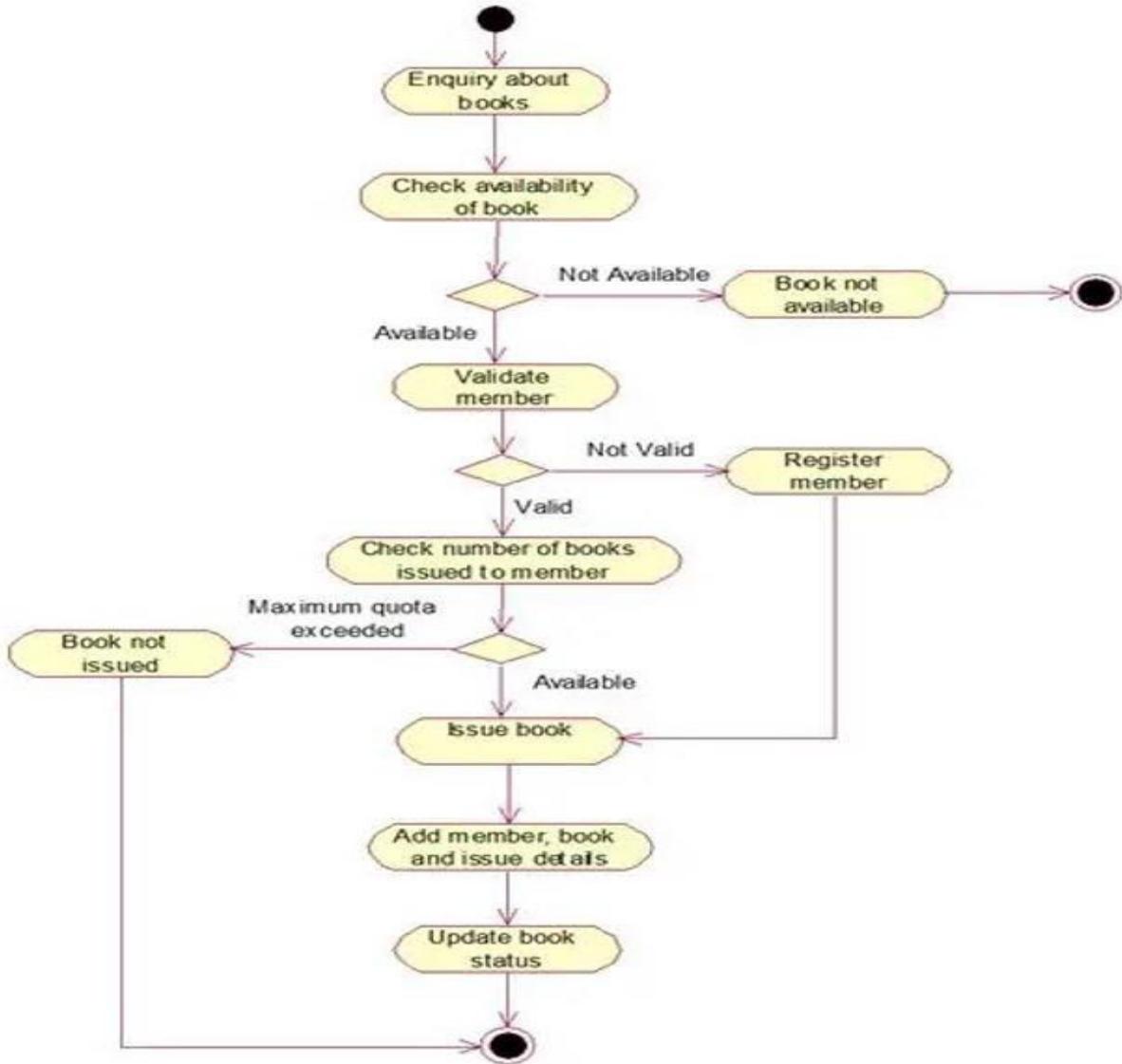
The use-case diagram shows how borrowers and librarians interact with the library system to manage reservations, item lending, returns, and maintenance tasks. Borrowers can make reservations, remove reservations, borrow items, and return items, all of which involve communication with the librarian. The 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.

LAB-2

DATE: PAGE: (1)

4) Stock Management System - ~~CRIS~~ SRS

1. Introduction

1.1) Purpose:
The purpose is to automate and streamline the process of managing inventory within an organization.

1.2) Scope:

- Maintain a record of all stock items with details like name, category, quantity, price and supplies
- Enable to add, update, delete and ~~see~~ search records
- Record purchase and sales transactions automatically updating stocks.

1.3) Definitions, abbreviations and acronyms:

- SMS - Stock Management System
- Admin - System administrator responsible for full control
- Stock item - A product or material stored in inventory
- Transaction - A record of sales or purchases.

1.4) References:
→ IEEE Standards for SRS.

2. Overall Description

2.1) Product Perspective:
The SMS will be a standalone application (desktop-web-based) interacting with a

2.2) Product functions:

- Authentication: Secure login for different roles (Admin, Staff, Viewer)
- Stock Management: Add, update, delete, & search stock items.
- Supplier Management
- Transaction Management
- Reporting

2.3) User Characteristics

- Admin: Knowledge of system administration and business rules
- Staff: Basic computer skills for handling stock updates and transactions.
- Viewer/Manager: Decision-Makers reports to monitor stock-flow.

2.4) Constraints

- System must run on Windows or Linux OS
- Requires database connectivity (MySQL, SQLite or equivalent)
- Limited to organizational stock need, suitable for future expansion.

2.5) Assumptions and Dependencies

- Users have basic computer literacy
- stable power + internet supply (if web-based)
- Reliable supplier and customer records for accurate tracking.

3. System Requirements

3.1) Functional Requirements:

1. User Authentication
2. Stock Item Management
3. Inventory Tracking
4. Transaction Management
5. Reports + Analysis

3.2) Non-functional Requirements

1. Performance
2. Security and Privacy
3. Reliability and Availability
4. Usability
5. Portability

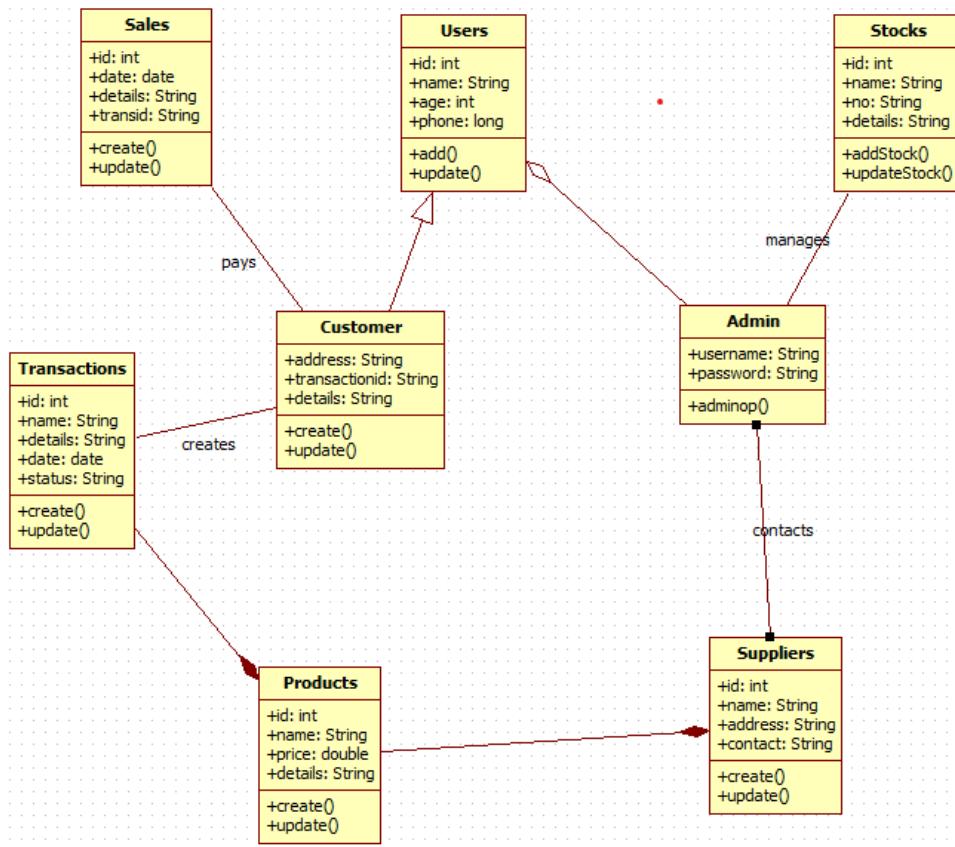
3.3) Domain Requirements

1. Stock Policies
2. Reorder Policies
3. Supplier and Customer Management
4. Transaction Rules

4. Appendix:

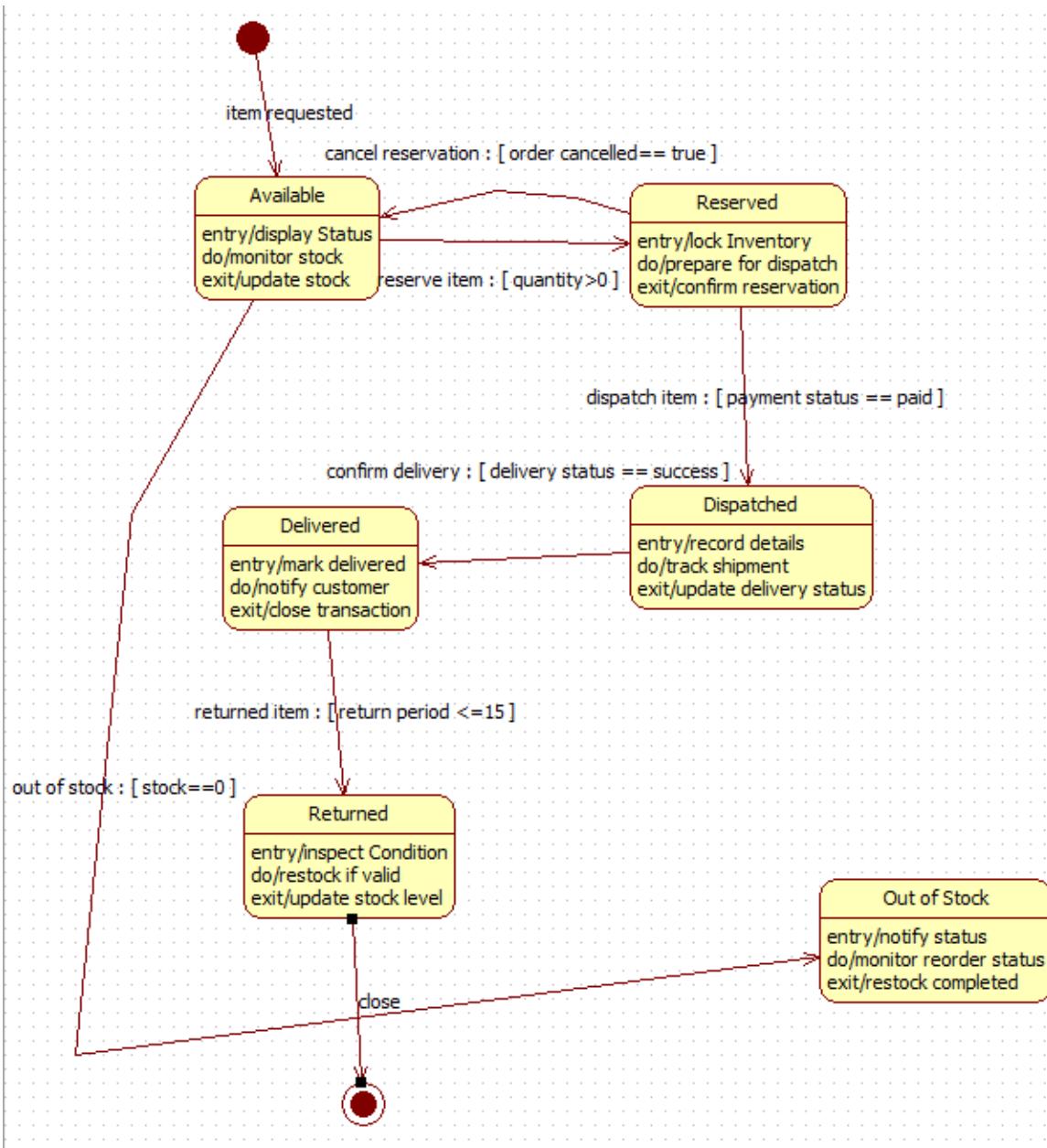
• IEEE Standards for Software Requirements Specification

Class Diagram : Fig 4.1



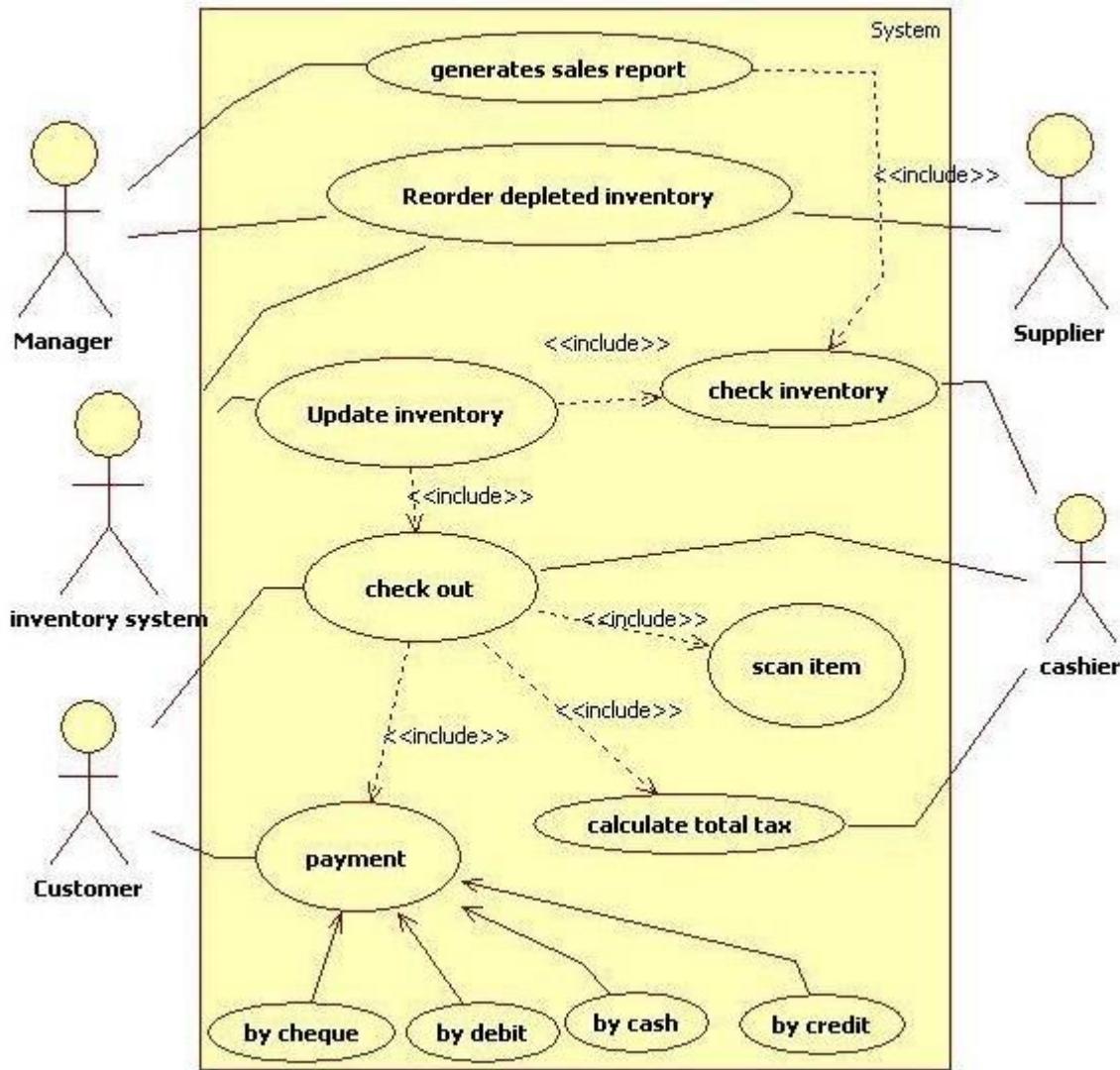
The class diagram for the Stock Maintenance System shows how different entities work together to manage inventory, products, sales, and supplier information. The Admin oversees the system by managing stock records and communicating with suppliers, who provide product details to maintain inventory levels. 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.

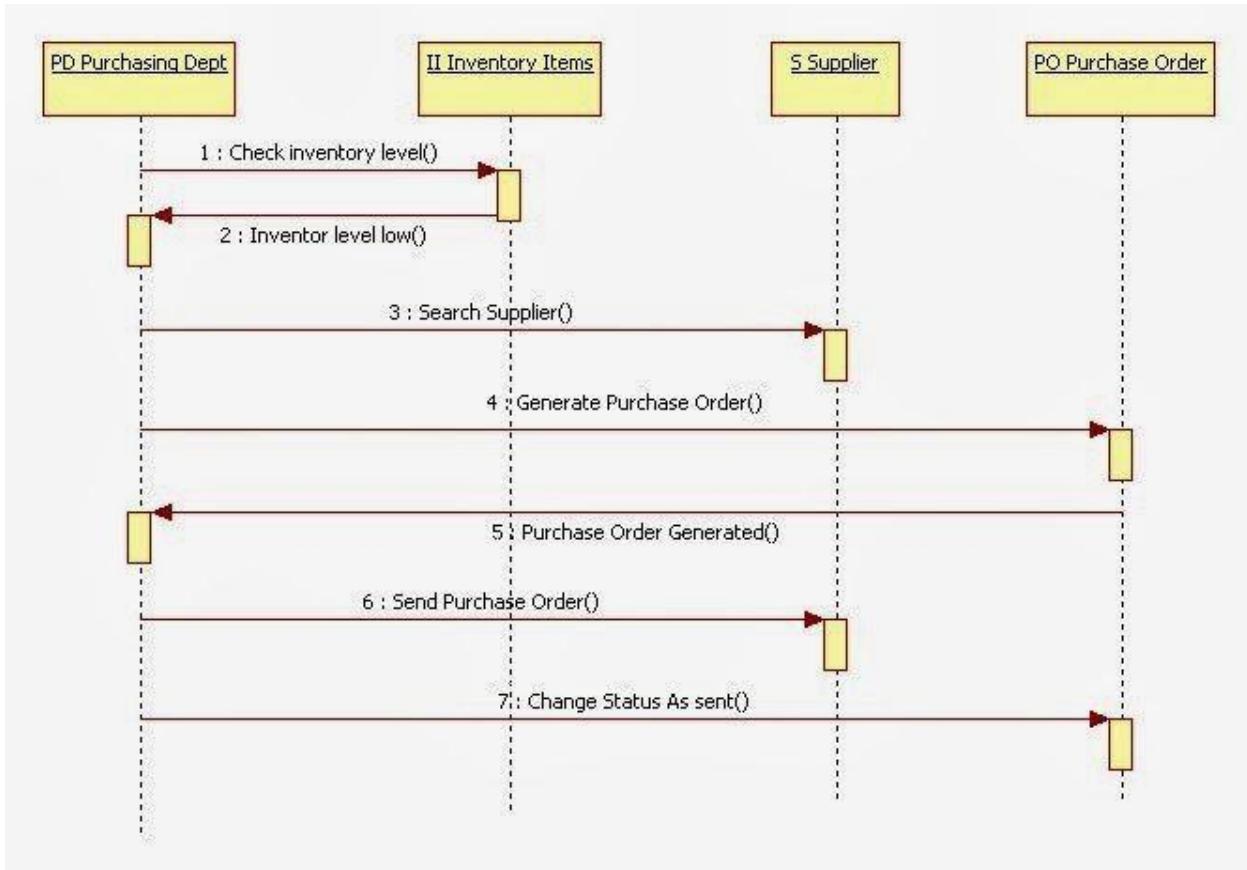
Use case Diagram : Fig 4.3



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

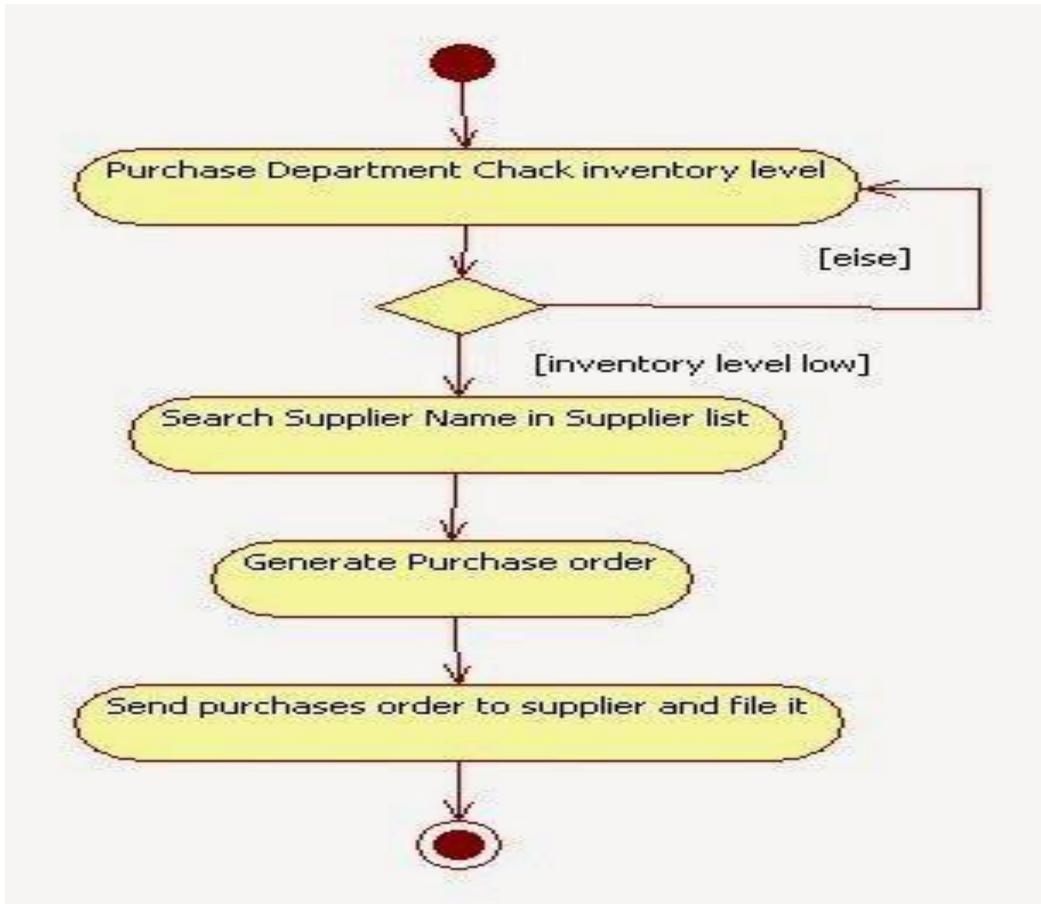
demonstrates how managers, suppliers, cashiers, and customers collaborate with the system to maintain accurate stock levels and complete sales transactions

Sequence Diagram : Fig 4.4



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 purchase 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



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.

5) Passport Automation System

1. Introduction

1.1) Purpose:
The PAS is designed to simplify and automate the process of applying for renewing and managing passports. It provides a digital solution for handling user applications, verifying documents, scheduling appointments, tracking application status and issuing passports.

1.2) Scope:
Allow citizens to apply for a new passport or renew an existing one.
Enable users to upload req documents online.
Users include:
• Applicants (citizens): Apply, upload documents, book appointments, track status.
• Passport officers/ Admin: Verify, approve, reject applications & manage system data.
• Supervisors: Monitor operations and generate reports.

2. Overall Description

2.1) Product Perspective:
The PAS will be a web-based application integrated with a secure database. Applicants will interact with the system through a front-end user portal, while administrators will use a backend interface for verification and approval.

2.2) Product functions:

- User registration and login
- Application Management
- Appointment Scheduling
- Verification & Approval
- Tracking & Notifications
- Reports

2.3) User Characteristics

- Applicants
- Admins / officers
- Supervisors

2.4) Constraints

- The system must comply with government passport rules and security regulations
- High-security standards must be enforced for sensitive data.

2.5) Assumptions and Dependencies:

- Applicants provide valid and authentic documents
- Government ID databases are accessible for verification.

3. System Requirements

3.1) Functional Req.

1. User Registration and authentication
2. Application Submission
3. Appointment Scheduling
4. Verification + Approval
5. Tracking & Notifications
6. Reports

3.2) Non-functional Req.

1. Performance
2. Security
3. Reliability & Availability
4. Usability
5. Portability

6.

3.3) Domain Req.

1. Government Policies
2. Security Application Rules
3. Appointment Policies
4. Verification & Approval rules

4. Appendix

4.1) Definitions, Acronyms, Abbreviations

PAS - Passport Automation System

Applicant - Citizens

Admin - Authorized passport officer managing applications

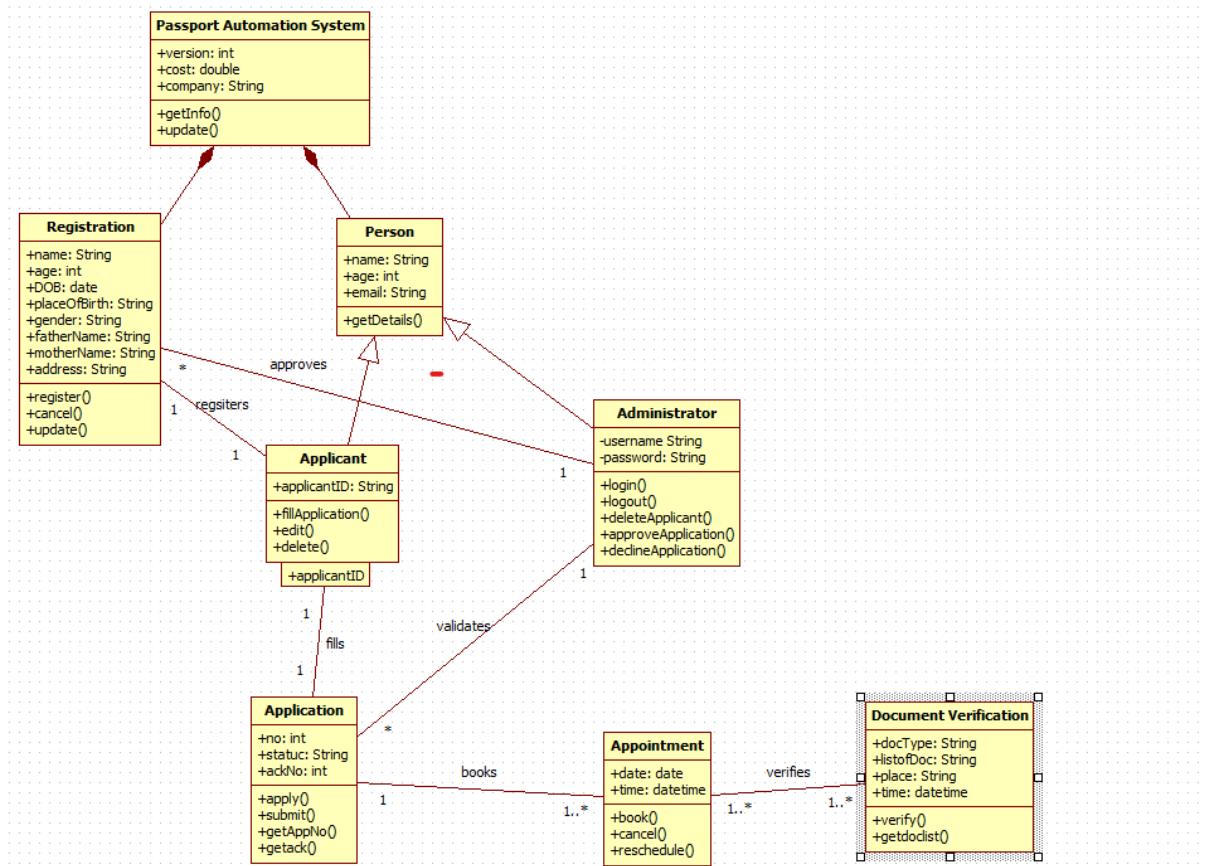
Appointment - Schedule date for biometric notification and/or interview

Applicants - Citizens applying for a passport

4.2) References

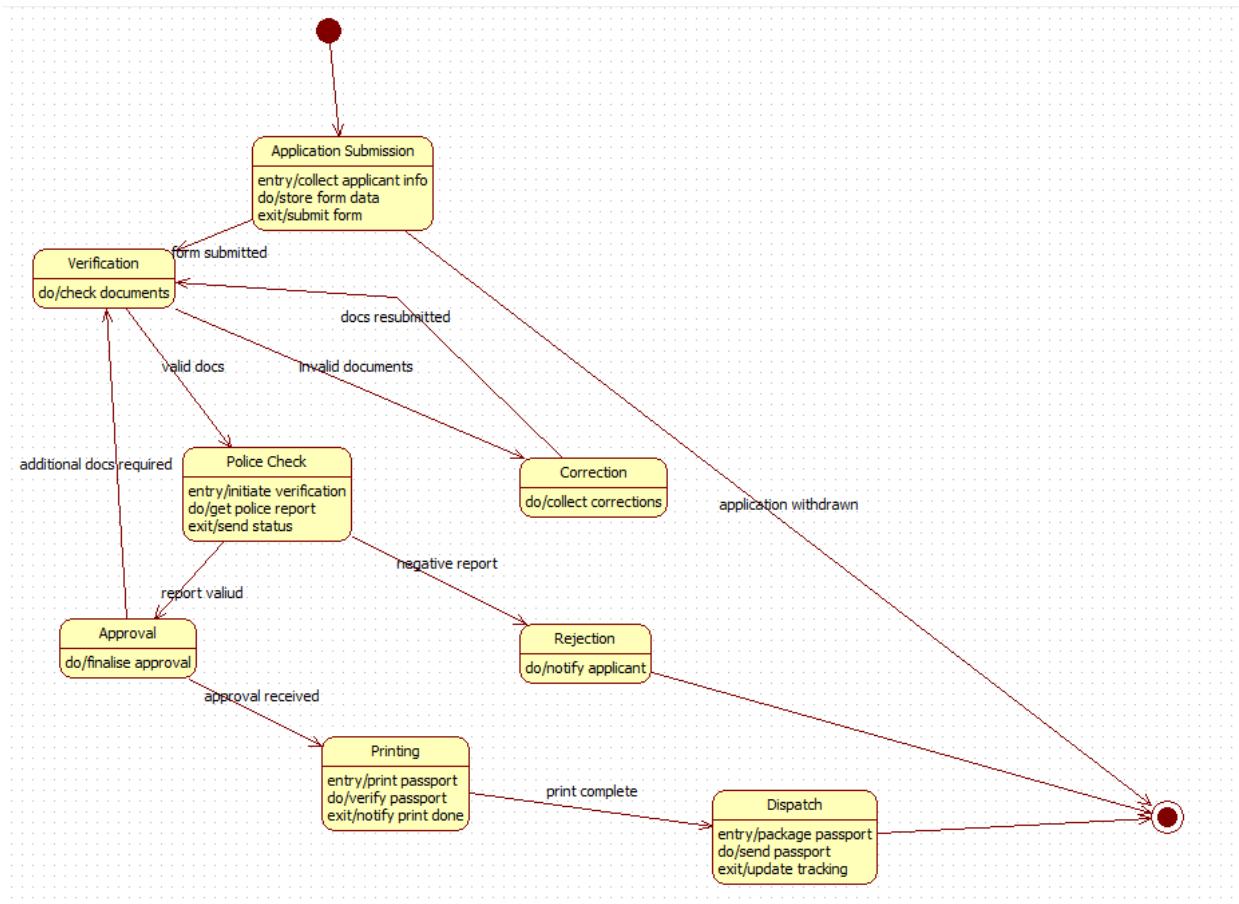
IEEE standards for SRS documentation

Class diagram : Fig 5.1

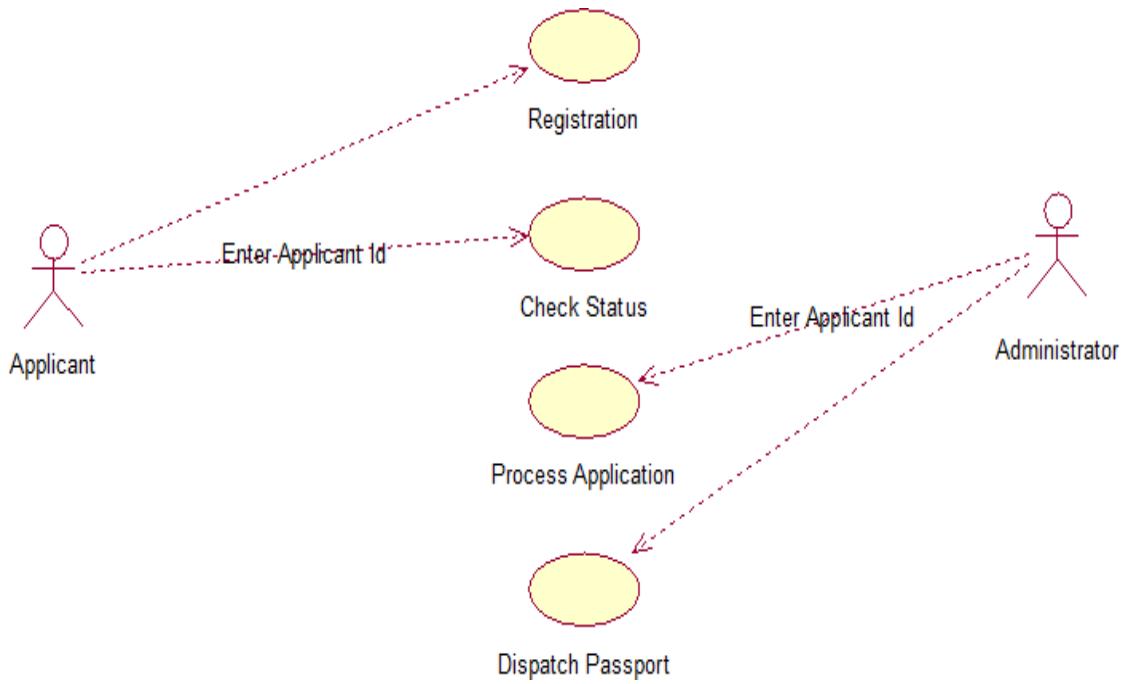


The class diagram for the Passport Automation System illustrates the major components involved in managing passport applications and their interactions. The Person class stores basic personal details and is used by the Registration process to record applicant information. Once registered, an Applicant 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

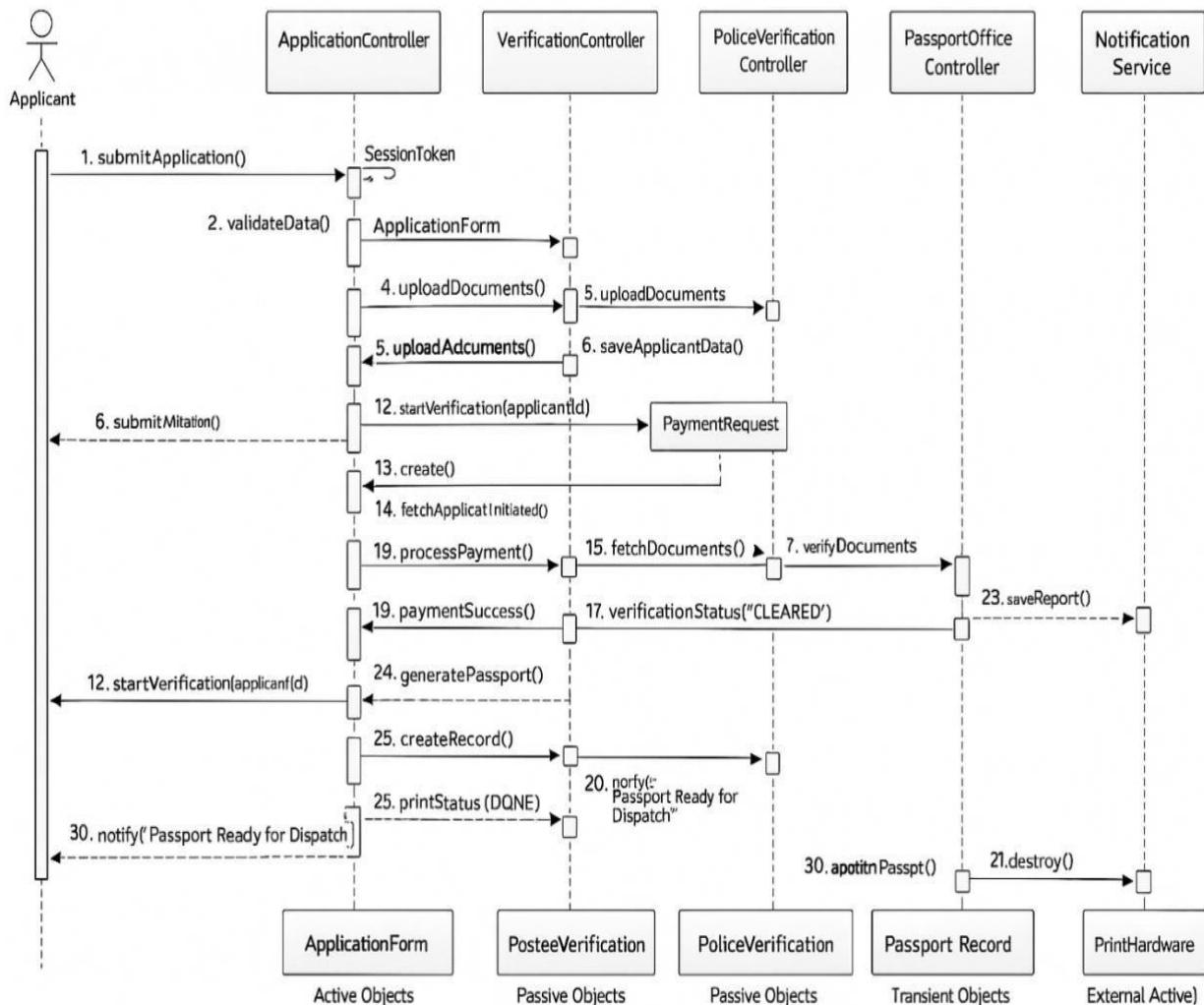


Use Case Diagram : Fig 5.3



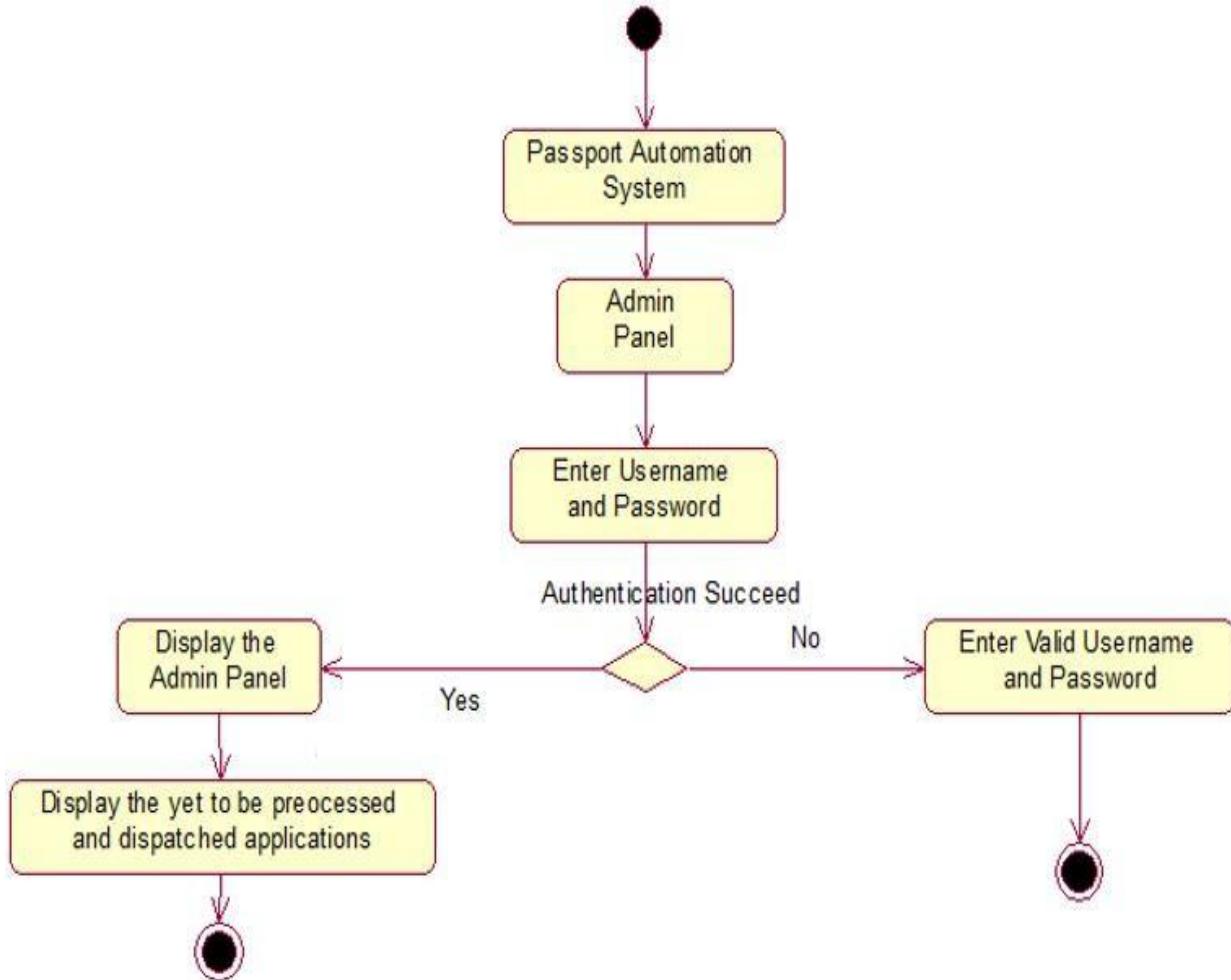
The use-case diagram shows the interaction between the applicant and the administrator during the passport application process. The applicant can register in the system, enter their applicant ID to check the status of their application, and eventually receive their dispatched passport. The administrator 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



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.