

**B.M.S. COLLEGE OF ENGINEERING BENGALURU**  
Autonomous Institute, Affiliated to VTU



Lab Record

**Object-Oriented Modeling – 23CS5PCOOM**

*Submitted in partial fulfillment for the 5<sup>th</sup> Semester Laboratory*

Bachelor of Engineering  
in  
Computer Science and Engineering

*Submitted by:*

**Dhruvdeep Nayak**

**(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 **Dhruvdeep Nayak (1BM23CS093)** during the 5<sup>th</sup> 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.

### **SRS – Software Requirements Specification (Short Version)**

Hotel management system	
Date _____	Page _____
1) Introduction	document describes the hotel management system.
1.1 Purpose	The purpose of HMS is to provide a digital platform to automate hotel operations such as reservations, check-in, billing and customer management system aiming to improve efficiency, reduce manual errors and enhance customer satisfaction.
1.2 Scope	The hotel management system will allow hotel staff to manage reservations, allocate rooms, handle customer check-in and check-out, and generate invoices. Customers will be able to make online bookings, request services, and receive automated confirmations.
1.3 Document Conventions	This document follows the IEEE SRS standard. Functional requirements, non-functional requirements, and external interface requirements are clearly stated.
1.4 Intended Audience and Reading suggestions	This document is intended for software developers, system architects, hotel administrators, project managers, and testers.
1.5 References	<ul style="list-style-type: none"> <li>• IEEE standard for SRS documentation</li> <li>• PCI DSS</li> <li>• Online hotel booking guidelines</li> </ul>
2) Product Functionality	<p>Overall Description: The HMS is a centralized web-based and mobile-supported solution. It integrates with payment gateways, room management systems, and other hotel management platforms.</p> <p>Product Functions:</p> <ul style="list-style-type: none"> <li>• Overall Description</li> <li>• Product Functionality</li> <li>• Overall Description: The HMS is a centralized web-based and mobile-supported solution. It integrates with payment gateways, room management systems, and other hotel management platforms.</li> <li>• Product Functions</li> <li>• Online room booking and cancellations</li> <li>• Customer check-in and check-out</li> <li>• Room allocation</li> <li>• Billing and invoice generation</li> <li>• Online payment support via multiple banks and payment gateways</li> <li>• Hotel guest information management system</li> <li>• User login and characteristics</li> <li>• Hotel Guests: Customer booking and staying at hotel</li> <li>• Receptionists: Handling check-in, check-out</li> <li>• Managers: Generating reports, occupancy and financial data</li> <li>• System Administrators: managing system security</li> </ul> <p>Design and implementation considerations:</p> <ul style="list-style-type: none"> <li>• Comply with PCI DSS for secure payments</li> <li>• Data must be encrypted</li> <li>• Multi-platform</li> </ul> <p>User documentation:</p> <ul style="list-style-type: none"> <li>• User manual for hotel staff and administrator</li> <li>• Online help guides for guests</li> <li>• API documentation for developers</li> </ul>

3 Strategic Requirements	
3.1 Functional requirements	Non-functional Requirements Performance, Security, Trustworthiness
<ul style="list-style-type: none"> <li>- System shall allow guests to book rooms online.</li> <li>- System shall authenticate and authorize participants.</li> <li>- System shall manage check ins and check outs.</li> <li>- System shall allow administrators to generate reports.</li> <li>- System shall generate invoices and receipts.</li> </ul>	Time to book a room is short
3.2 Non functional requirements	<ul style="list-style-type: none"> <li>- System shall ensure 99.9% uptime.</li> <li>- System shall support role-based access control.</li> <li>- System shall be available for multi branch hotels.</li> </ul>

~~System to generate guest account automatically. Also, direct user - guest relationship. User management, hotel configuration, payment integration, reporting.~~

~~Guests make bookings : comfortable & smooth~~

~~Guests can book rooms online 24x7 365 days a year~~

~~Billings are set based on room occupancy~~

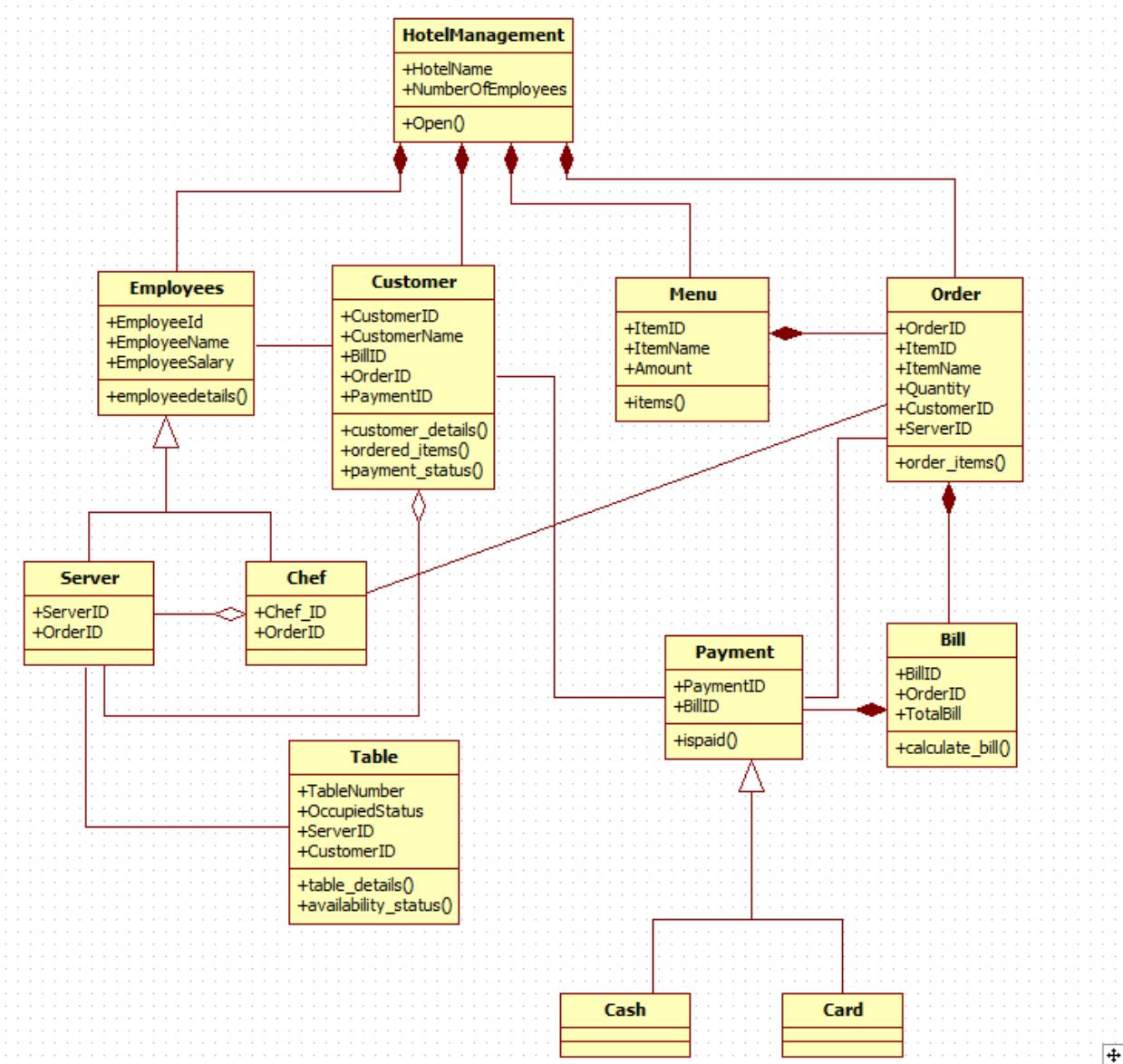
~~Administrators shall maintain guest information~~

~~Administrators shall maintain guest information~~

~~Guests shall be able to book rooms online easily~~

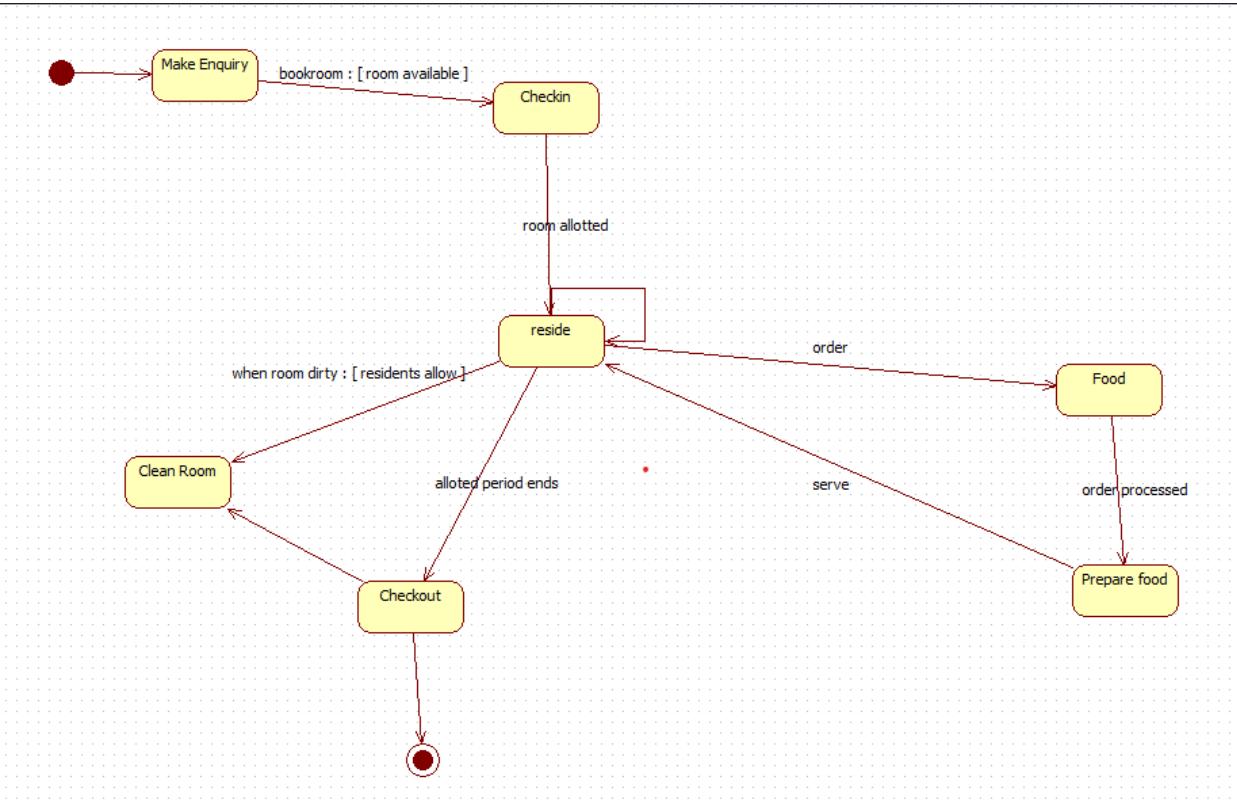
~~Guests shall be able to book rooms online easily~~

**Class Diagram: fig 1.1**



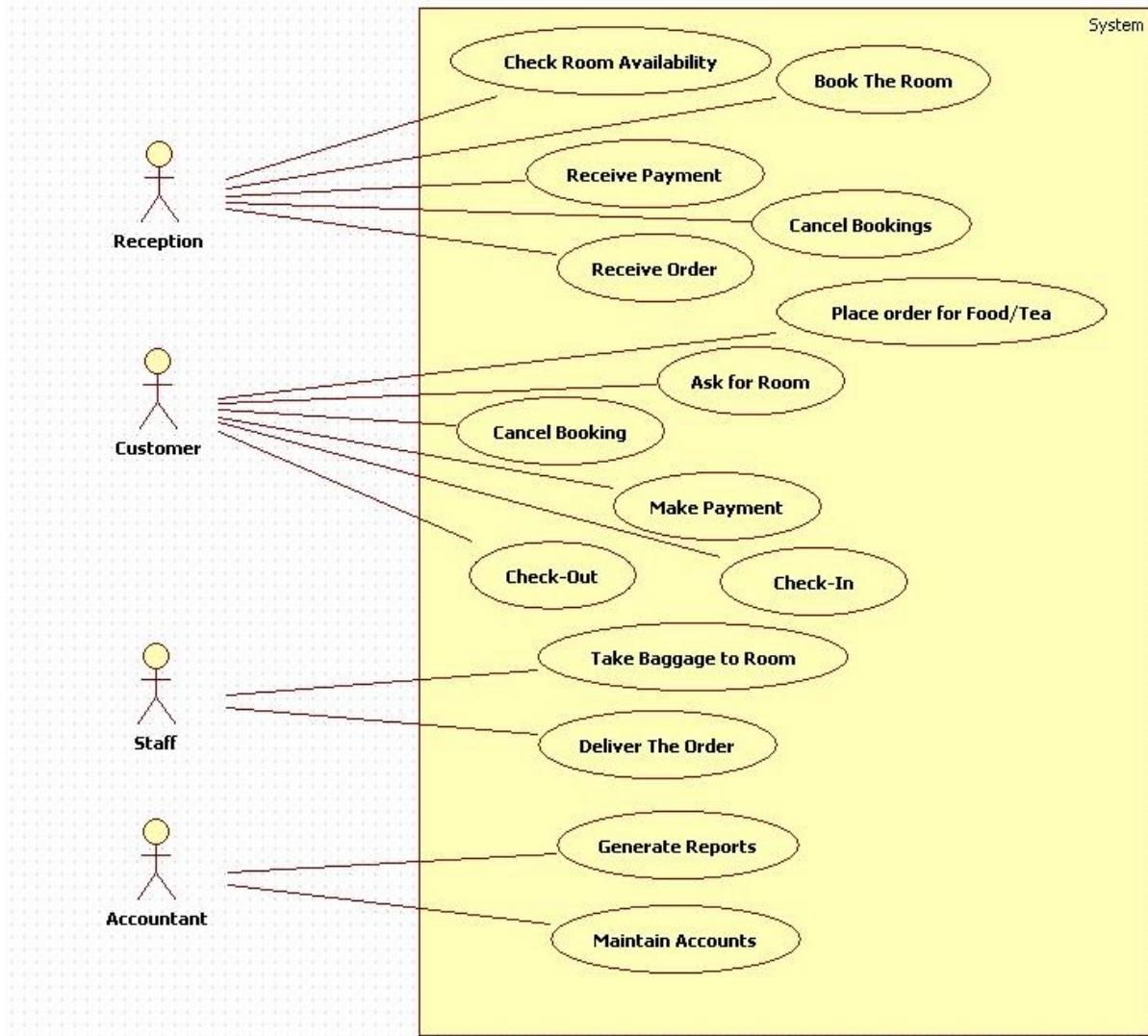
The class diagram of the Hotel Management System illustrates the main entities involved in hotel operations and how they interact. The central HotelManagement class is linked to major components such as Employees, Customer, Menu, Order, Bill, and Payment. Employees are modeled using a generalization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



Sequence Diagram : Fig 1.4

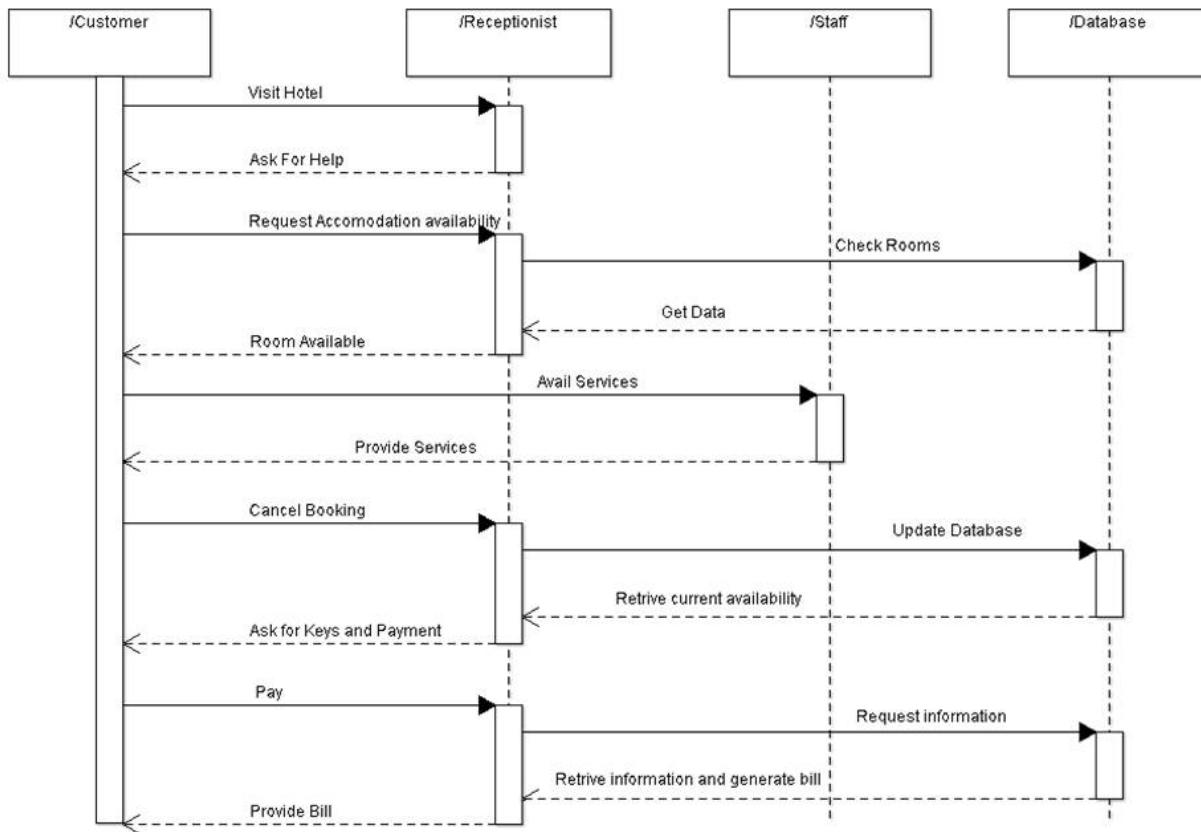
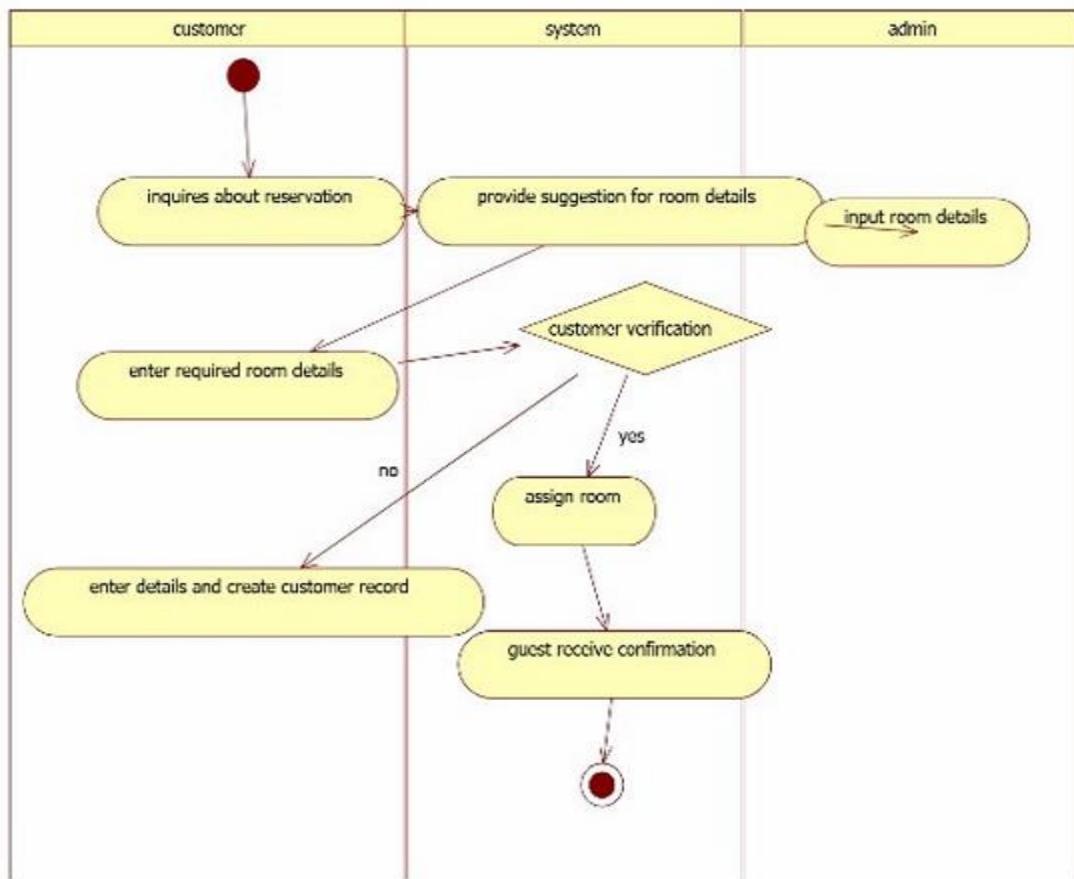


fig 1.5

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

Credit Card Processing	
1) <u>Introduction</u>	1.1 Purpose The purpose of credit card processing system is to provide secure and efficient platform for processing transactions. This system is designed for merchants, customers and banks to ensure seamless interaction during purchases. It helps validate card details, authorize payments and generate receipts quickly. Overall, CCPS aims to ensure trust and reliability in digital payment environments.
1.2 Scope	The credit card processing system will automate card validation, authorize billing and detect any instance of fraud. The system needs to save time, reduce errors and ensure safe payments. It also ensures PCI DSS compliance by integrating with external banking API's and provides a dashboard for monitoring and reporting various metrics. A detailed description of system features and architecture is provided.
1.3 Document Convention	This document follows IEEE 1016 standard. Functional requirements, non-functional requirements and test cases are described in a structured manner.
1.4 <u>Intended Audience and Reviewer Suggestions</u>	This document is intended for developers, system architects, project managers and stakeholders from financial institutions.
1.5 References	<ul style="list-style-type: none"><li>IEEE standard for SRS</li><li>PCI DSS security standard</li></ul>
2) <u>Overall Description</u>	2.1 Product Structure The system is a distributed, cloud-based and API-driven solution. It integrates with banks, merchants and fraud detection services to handle end-to-end processing. It also includes various modules such as authentication, authorization, and reporting.
2.2 Product Functions	<ul style="list-style-type: none"><li>Credit card authorization and authentication service</li><li>Transaction approval and settlement service</li><li>Fraud detection and prevention mechanism</li><li>Billing and monthly statement generation</li><li>Refund and chargeback service</li><li>Logging and monitoring system</li></ul>
2.3 User Classes and Characteristics	<ul style="list-style-type: none"><li>Cardholders: Individuals making payments during credit purchases.</li><li>Merchants: Businesses accepting payments via the system.</li><li>Administrators: System managers and moderators.</li><li>Bank integration: External entities for authorization.</li></ul>
2.4 Operating Environment	<ul style="list-style-type: none"><li>Web-based system compatible with major browsers.</li><li>Use HTTPS for secure communication.</li></ul>

Date \_\_\_\_\_  
Page \_\_\_\_\_

**3.5 Design and implementation constraints**

- Must comply with PCI DSS security standards
- Use HTTPS for secure communication
- Data must be encrypted using AES 256

**3.6 System Requirements** (3.6.1 to 3.6.7) Section 1.1

**3.6.1 Functional Requirements** (Detailed description of workflow)

- System shall authenticate card details before processing
- System shall authorize transaction initiation
- System shall allow refunds against bank source

**3.6.2 Non-functional Requirements** Appendix B.1

- (a) System shall authenticate card details before processing and system shall ensure 99.9% uptime guarantee.
- (b) System shall use role-based access control, and
- (c) System shall comply with PCI DSS and ISO 27001.

**3.7 External interface Requirements** (notified) B.1

- User interface is web based for merchants and administrators
- API interface via REST APIs for external merchants
- Banking interface via secure ISO 8583 communication with banks

**3.8 Appendices** (notified) 1.1

- Glossary of financial terminologies in Transactional IoT T
- List of supported payment gateways Notified ~~and~~ Nedline  
09/09/23
- Transaction flow diagram Notified ~~and~~ 09/09/23

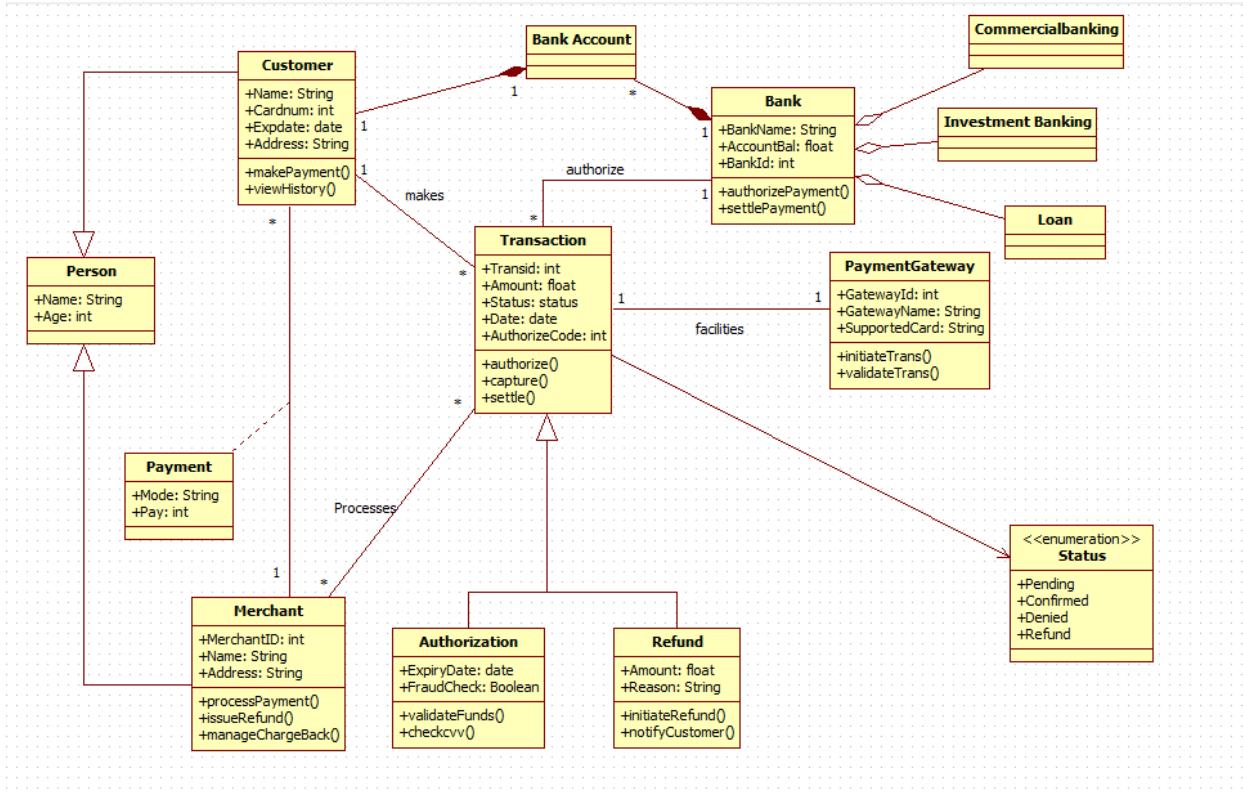
notified ~~and~~ B.1

notified ~~and~~ B.1

notified ~~and~~ B.1

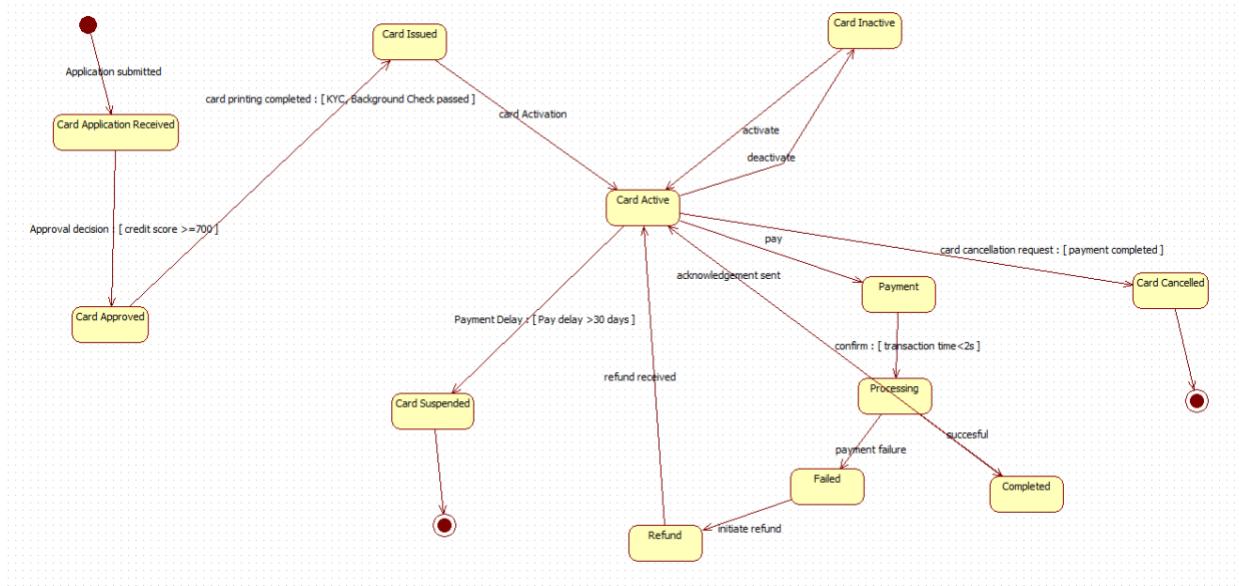
notified ~~and~~ B.1

## Class Diagram Fig:2.1



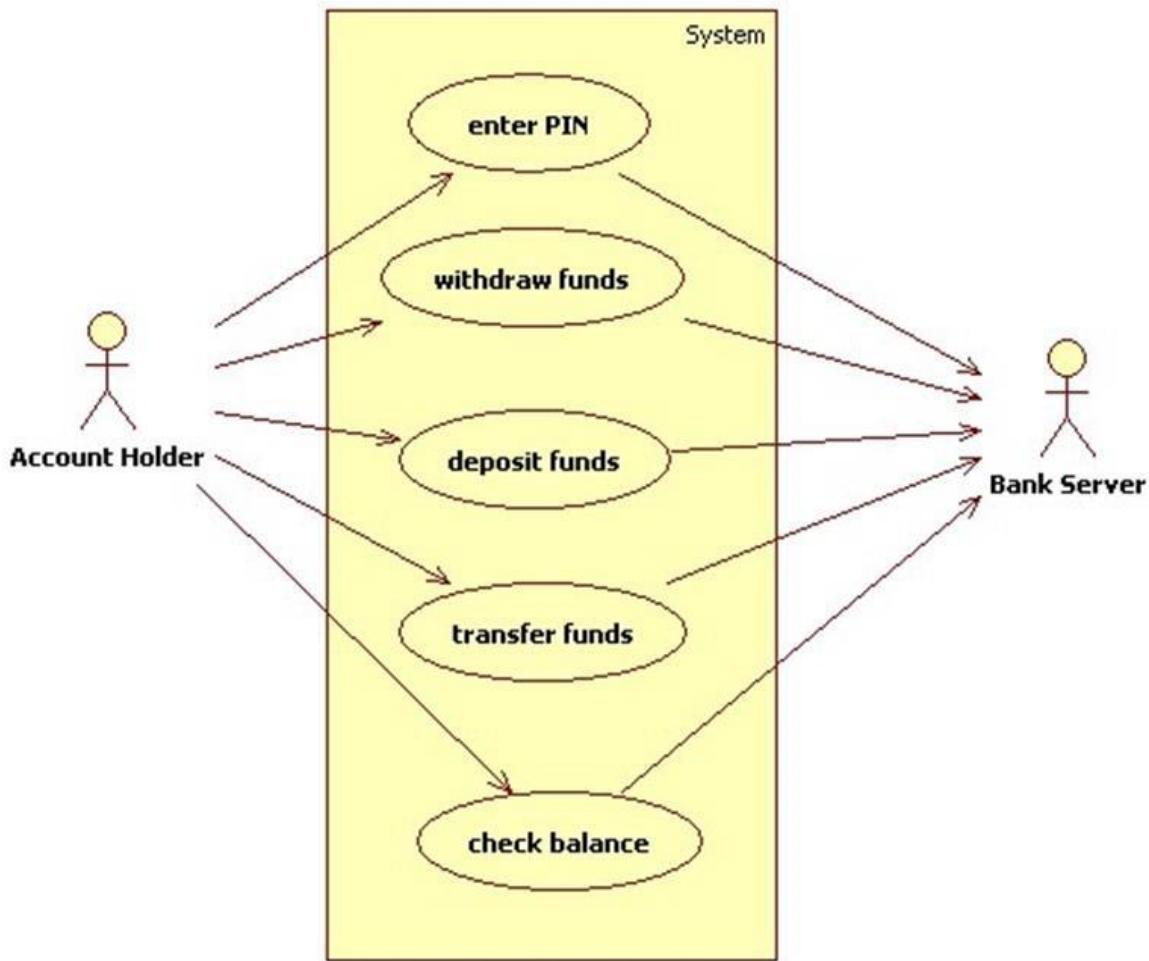
The class diagram for the Credit Card Processing System shows how customers, merchants, banks, and payment gateways interact to process credit card transactions. A customer, inheriting details from a general person class, initiates payments that create transaction objects containing information such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



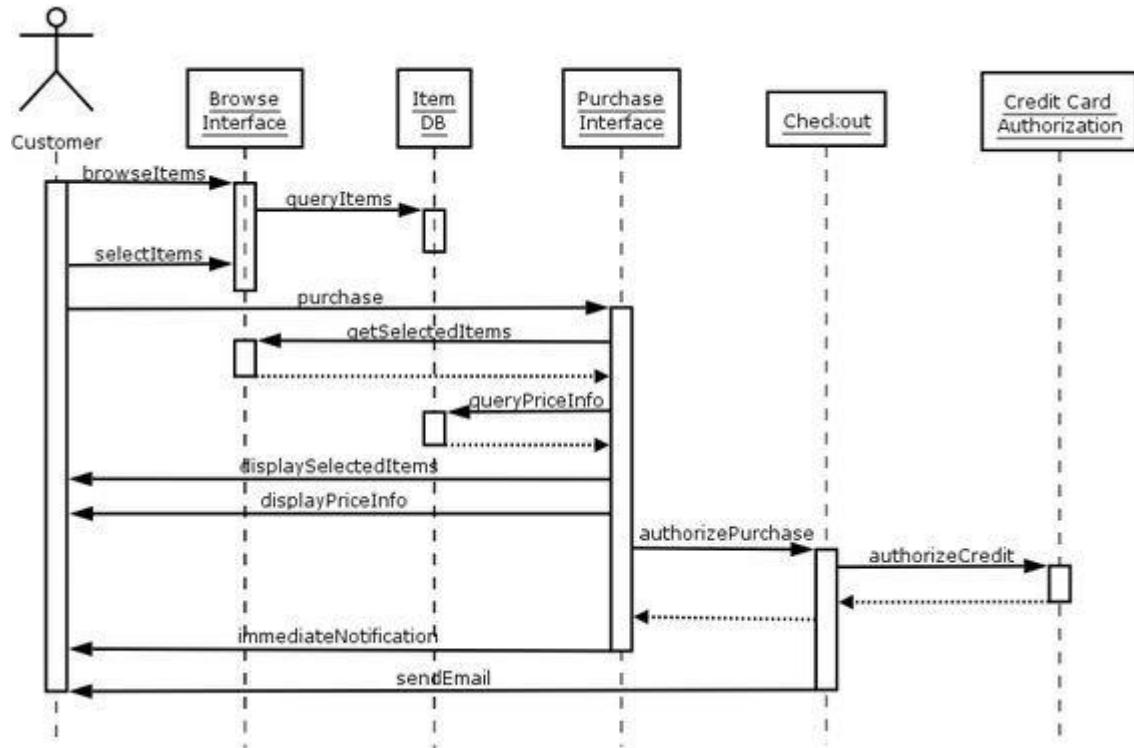
The state diagram shows the complete lifecycle of a credit card, beginning with the application submitted and moving to the card application received and approved states. Once approved and printed, the card enters the card issued state and becomes card active after activation. In the active state, the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

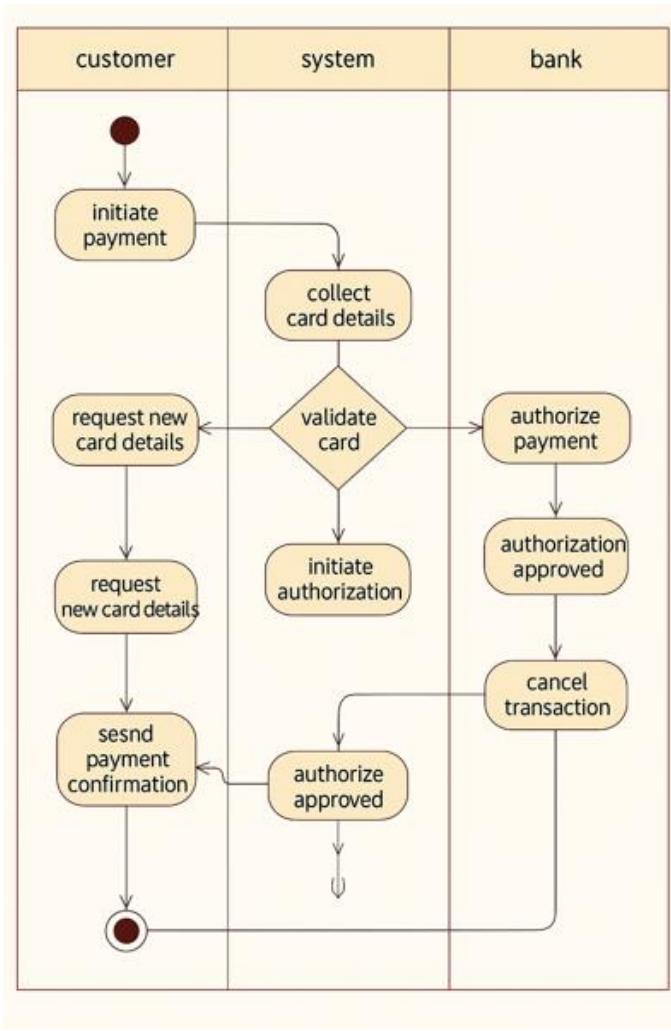


The use case diagram shows how an account holder interacts with the system to perform basic banking operations, such as entering a PIN, withdrawing funds, depositing money, transferring funds, and checking account balance. Each of these actions requires communication with the bank server, which verifies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

Date _____ Page _____	Date _____ Page _____
SRS for Library management system	Overall description
1) Introduction	Product perspective
2) Requirements	The LMS is a centralized system that replaces manual book records. It connects librarians, members and admin.
3) Purpose	The purpose of library management system is to automate library operations such as book cataloging, issuing, returning, fine calculations, user management, provides an easy way to search, borrow and return books.
4) Scope	Product Functions
5) Details	<ul style="list-style-type: none"> <li>Book cataloging and classification</li> <li>Book issue and return management</li> <li>Fine calculations for late returns</li> <li>Book reservation system</li> </ul>
6) Document Conventions	Member registration and profile management
7) Intended Audience and Reading suggestions	User classes <ul style="list-style-type: none"> <li>Students / members</li> <li>Librarians / staff</li> <li>Administrators</li> </ul>
8) References	Design constraints <ul style="list-style-type: none"> <li>Unique ID required for each book and member</li> <li>must support barcode / RFID</li> <li>data must be encrypted and have consistency checking</li> </ul>
9) Database design standard	Security features <ul style="list-style-type: none"> <li>data integrity</li> <li>data consistency</li> <li>data availability</li> <li>data security</li> </ul>
10) Existing library guidelines	Business rules <ul style="list-style-type: none"> <li>books can be issued for 3 months</li> <li>books can be renewed</li> <li>books can be returned early</li> </ul>

3 Specific Requirements Analysis

3.1 Functional Requirement

- System shall allow members to search books by title or author
- System shall manage lending returns
- System shall maintain history of issued and returned books
- System shall maintain history of late returns and fines

3.2 Non Functional Requirements

- shall ensure 99.9% uptime
- search query shall return within 2 seconds
- Role based access control should handle multiple users with different roles

4.1

initially maintained books - 2000

new queued books - 1000

available staff members - 500

transferred books - 1000

transferred members - 3000

available members - 2000

late returns - 100

overdue books - 50

lost books - 100

unfilled books - 100

transferred books - 2000

available books - 1000

available members - 2000

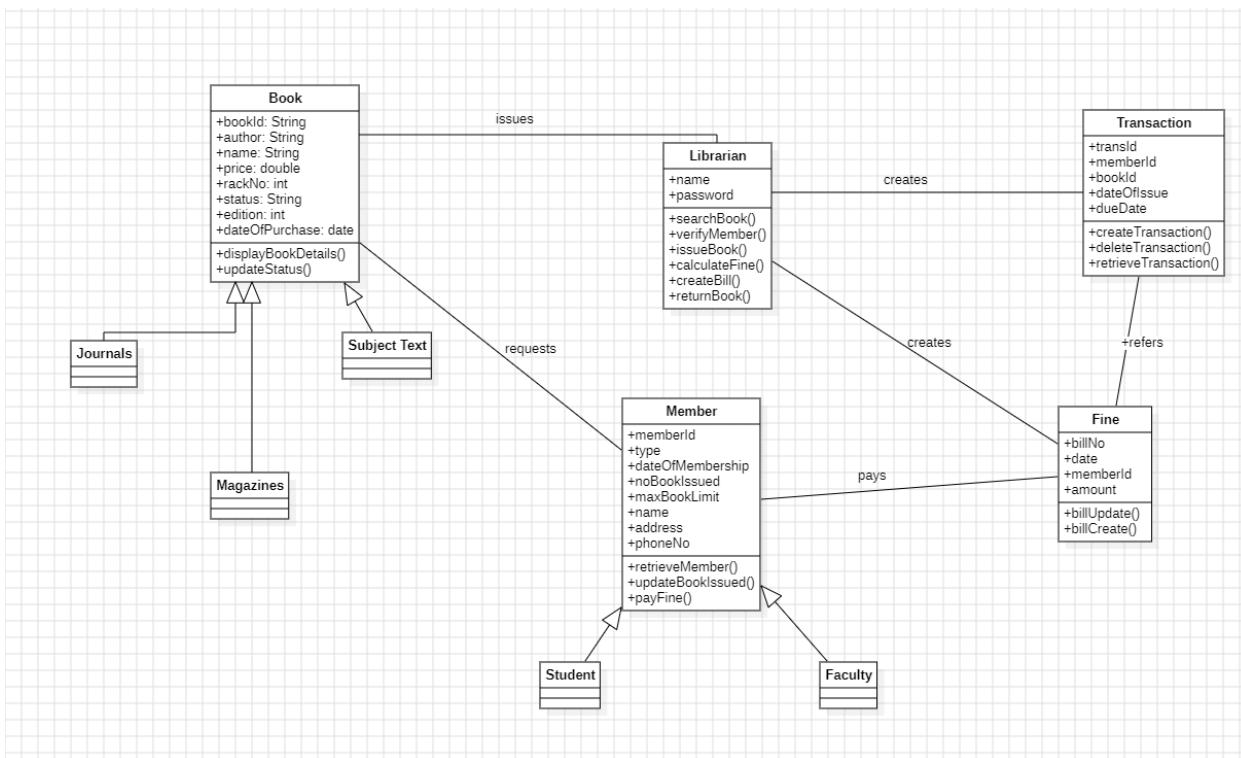
late returns - 100

overdue books - 50

lost books - 100

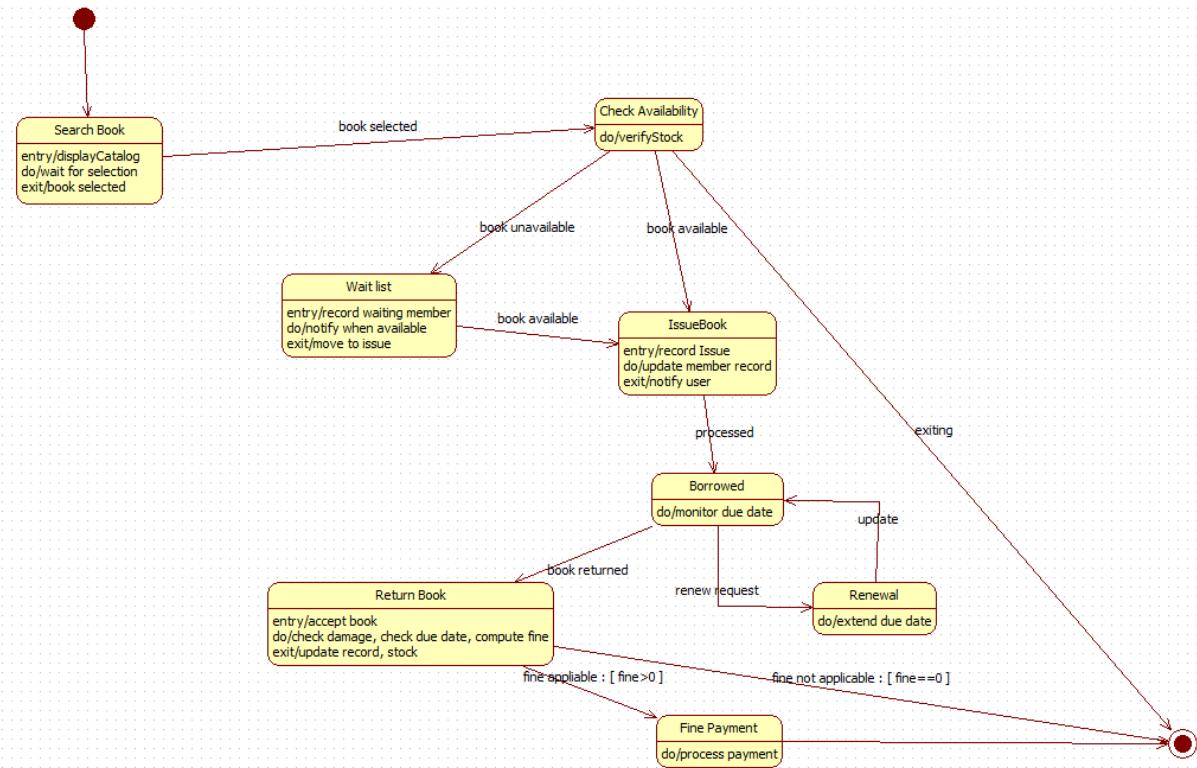
unfilled books - 100

Class Diagram : Fig 3.1



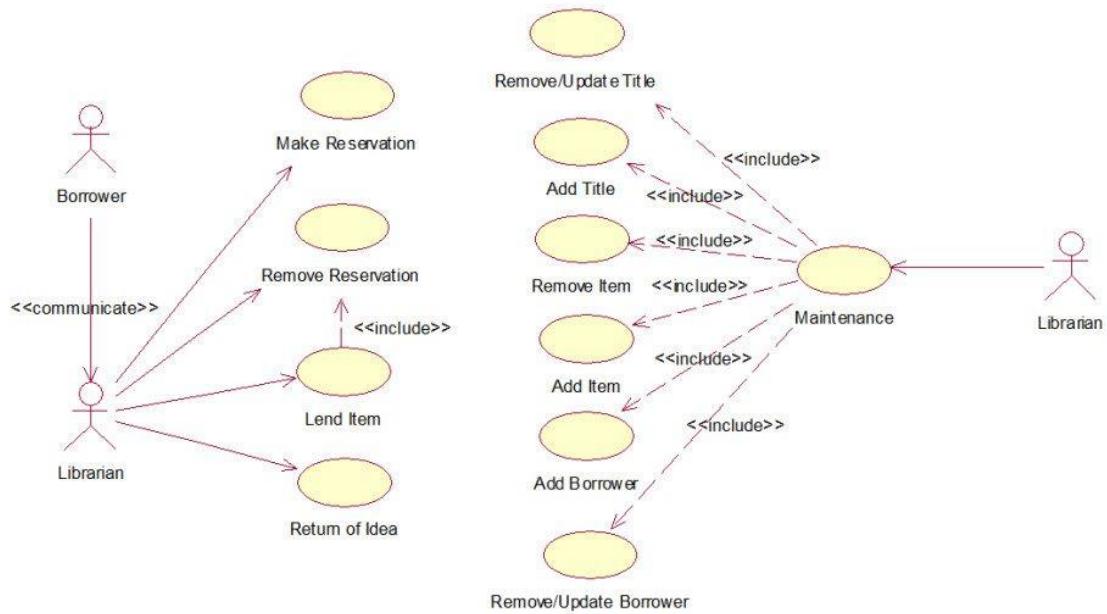
The class diagram for the Library Management System shows how books, members, librarians, transactions, and fines interact within the system. The Book class stores details about each book and is specialized into journals, magazines, and subject texts through inheritance. Members request books and may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



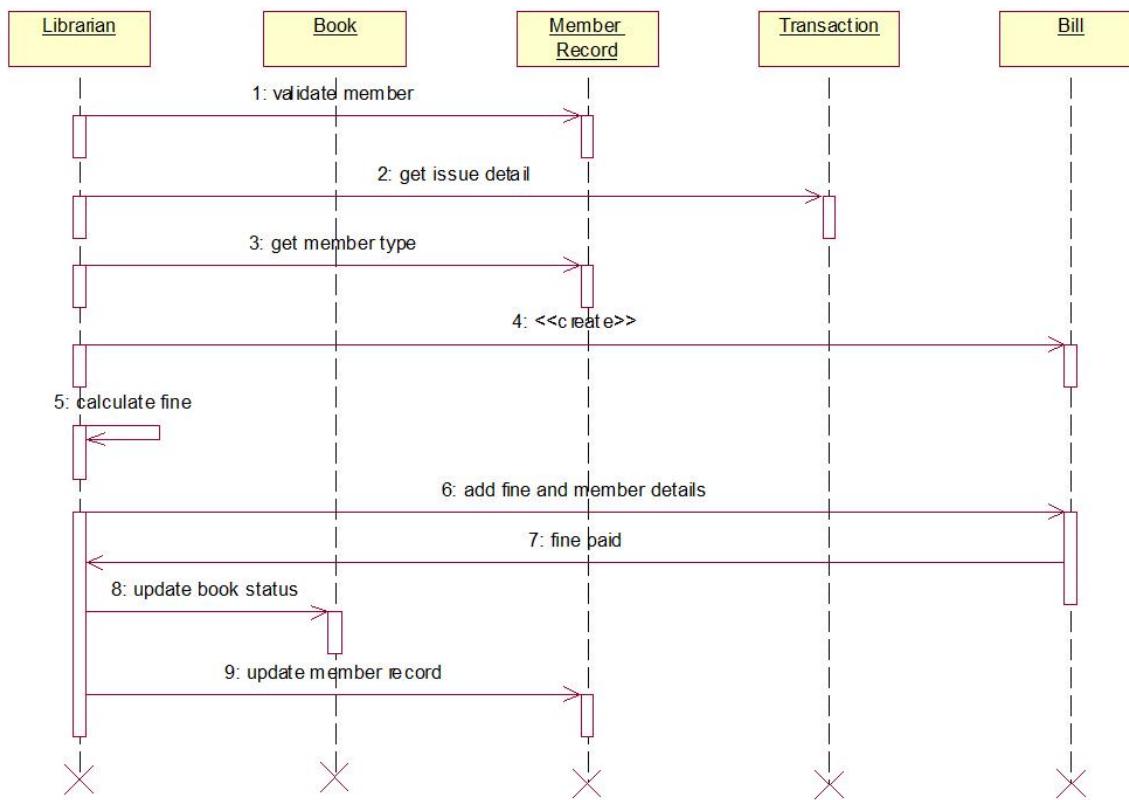
The state diagram shows the different stages a book goes through during the library borrowing process. It begins with the user searching for a book, after which the system checks its availability. If the book is unavailable, the request moves to the wait list state until a copy becomes available. When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

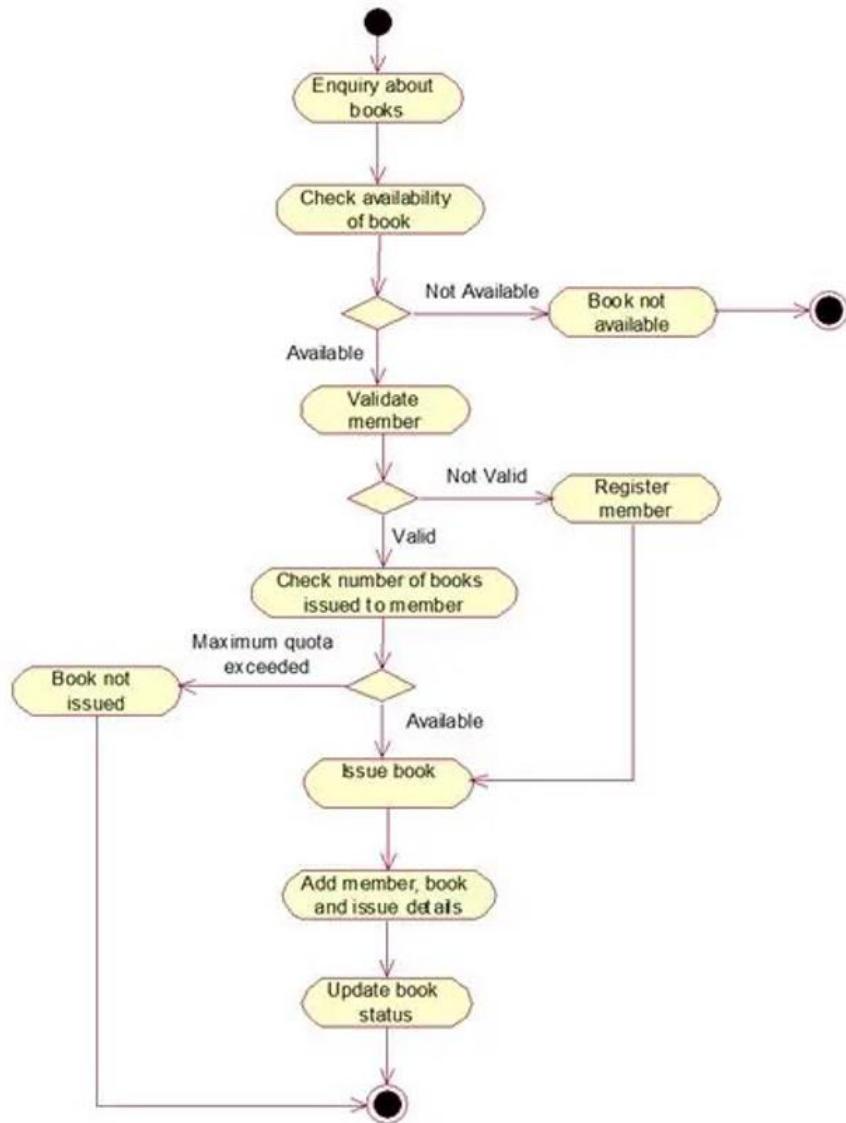


The use-case diagram shows how borrowers and librarians interact with the library system to manage reservations, item lending, returns, and maintenance tasks. Borrowers can make reservations, remove reservations, borrow items, and return items, all of which involve communication with the librarian. The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



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.

Stock maintenance system	
1. Introduction	Introducing the system
1.1 Purpose	The purpose specifies the requirements for a stock maintenance system intended for developers, testers, and stakeholders to ensure a shared understanding of the system
1.2 Scope	The S.M.S. will track stock levels, manage supplier record, purchase and sales, generate alerts for low inventory and streamline stock management
1.3 Definitions	<ul style="list-style-type: none"><li>SMS - stock maintenance system</li><li>SKU - stock keeping unit</li><li>Reorder level - threshold for triggering stock replenishment</li></ul>
1.4 References	<ul style="list-style-type: none"><li>IEEE std 836 - 1998 software requirement specification</li></ul>
1.5 Intended Audience	<ul style="list-style-type: none"><li>Business owners</li><li>Inventory staff</li><li>System developers</li><li>Testers</li></ul>

Overall description	
2.1 Product perspective	The S.M.S. is a standalone web application with database support, integrating with billing and supplier module.
2.2 Product Functions	<ul style="list-style-type: none"><li>Manage product and supplier information</li><li>Record stock in/out transaction</li><li>Track current inventory levels</li><li>Generate low stock alerts</li><li>Provide sales and stock reports</li></ul>
2.3 User classes	<ul style="list-style-type: none"><li>Store manager : Configure products, supplier policies</li><li>Staff : Record sales, update stock</li><li>Admin : Full system access</li></ul>
2.4 Design constraints	<ul style="list-style-type: none"><li>Must handle at least 10,000 stock items</li><li>Transaction must be in &lt; 3 seconds</li><li>AES encrypted data</li></ul>

### 3 System Requirements

Date \_\_\_\_\_  
Page \_\_\_\_\_

#### 3.1 Functional

- Product and supplier management
- Stock transaction
- Inventory tracking
- Reports

#### 3.2 Non Functional

- Scalability
- Security
- Usability

#### 3.3 Appendix

##### Glossary of inventory terms

with master, with history - price

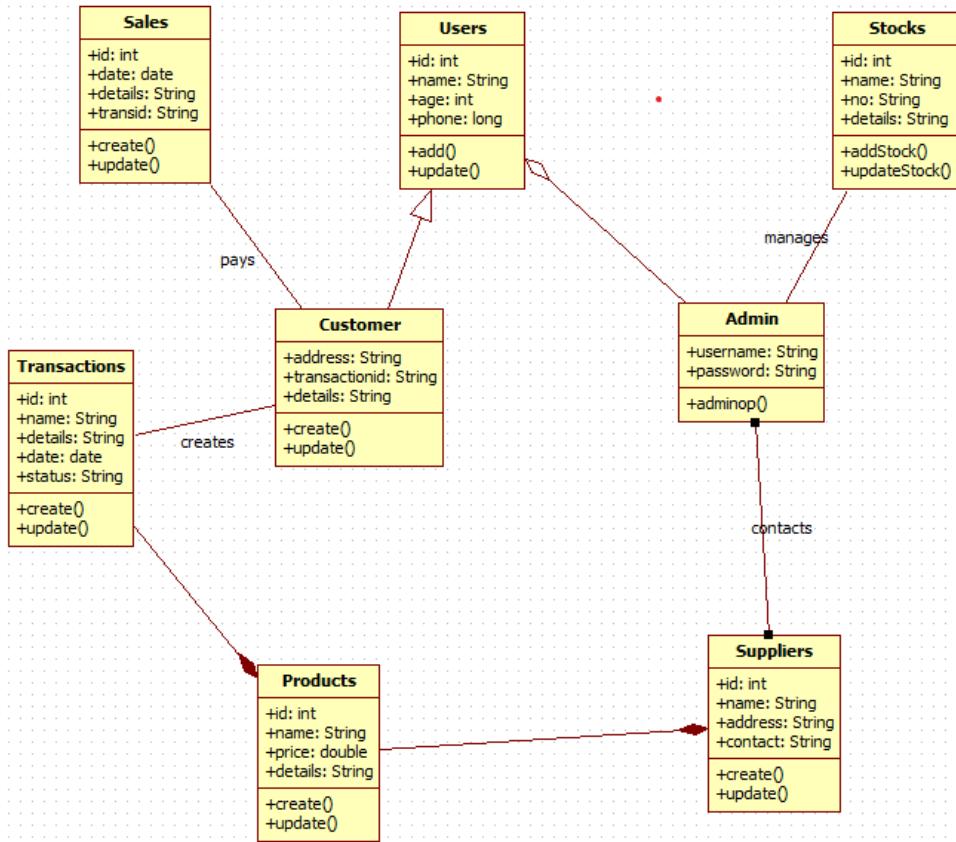
new item this month

will take over 01 task to eliminate

order > me at time introduced

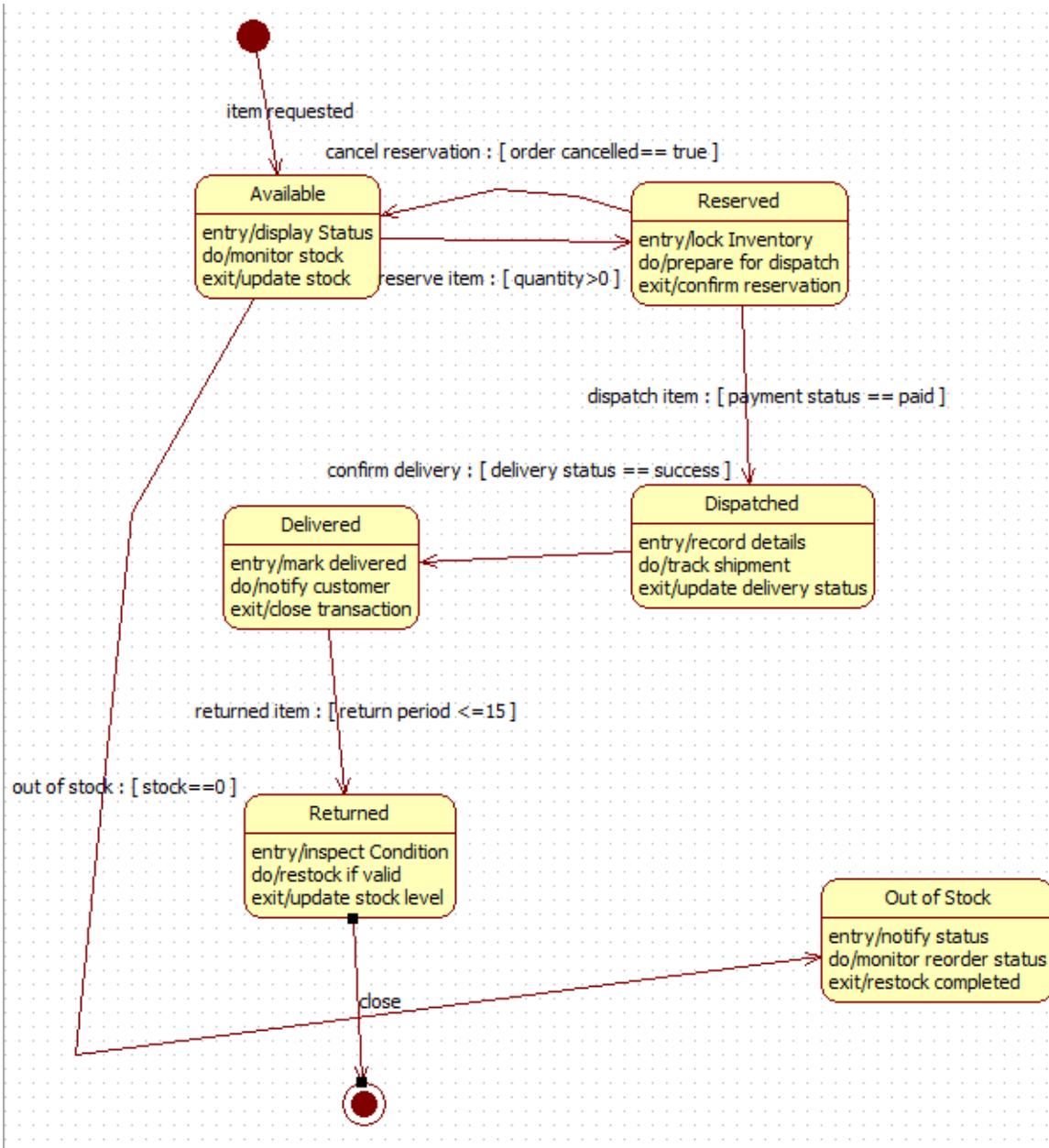
stock turnover 83%

Class Diagram : Fig 4.1



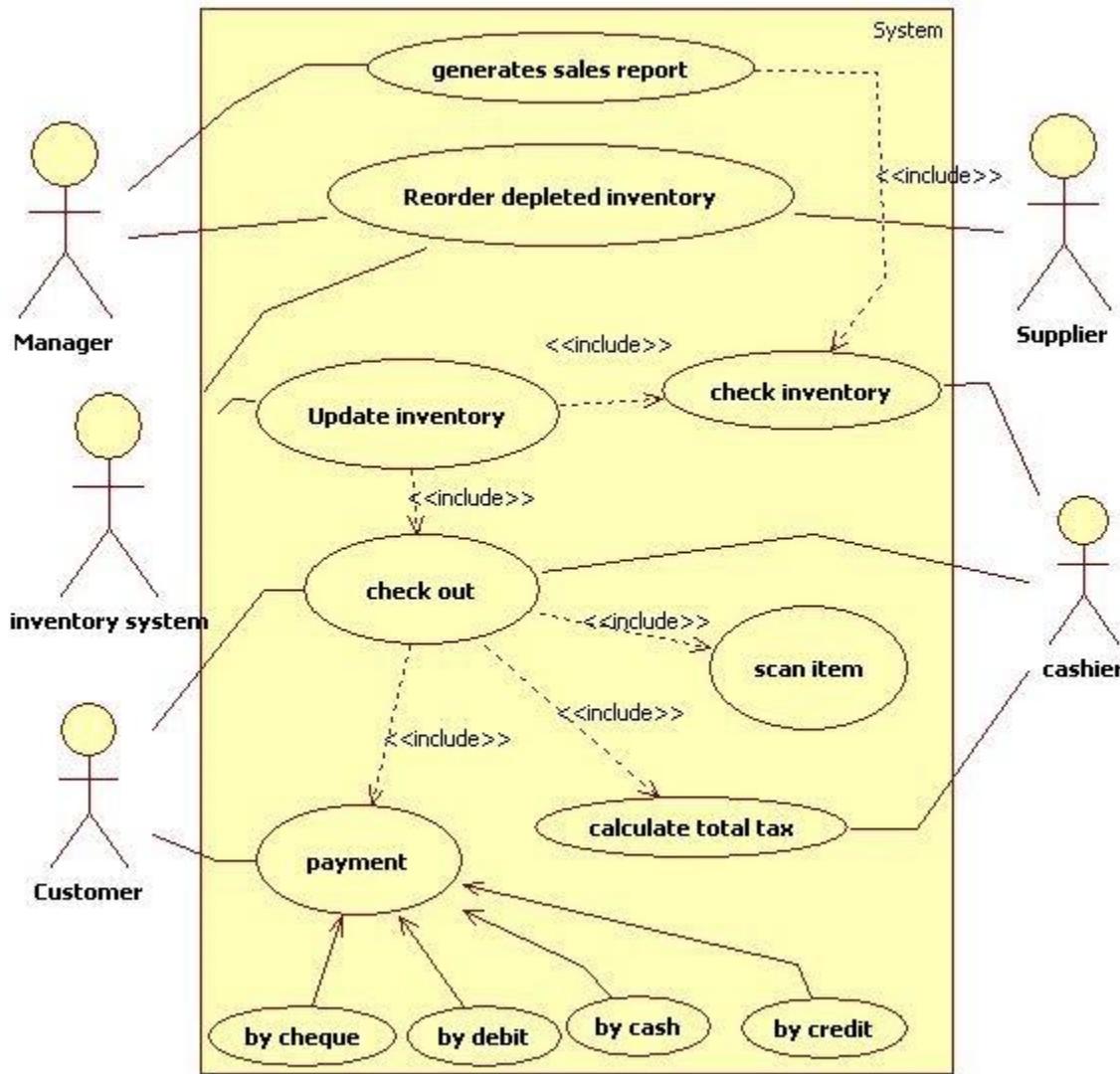
The class diagram for the Stock Maintenance System shows how different entities work together to manage inventory, products, sales, and supplier information. The Admin oversees the system by managing stock records and communicating with suppliers, who provide product details to maintain inventory levels.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

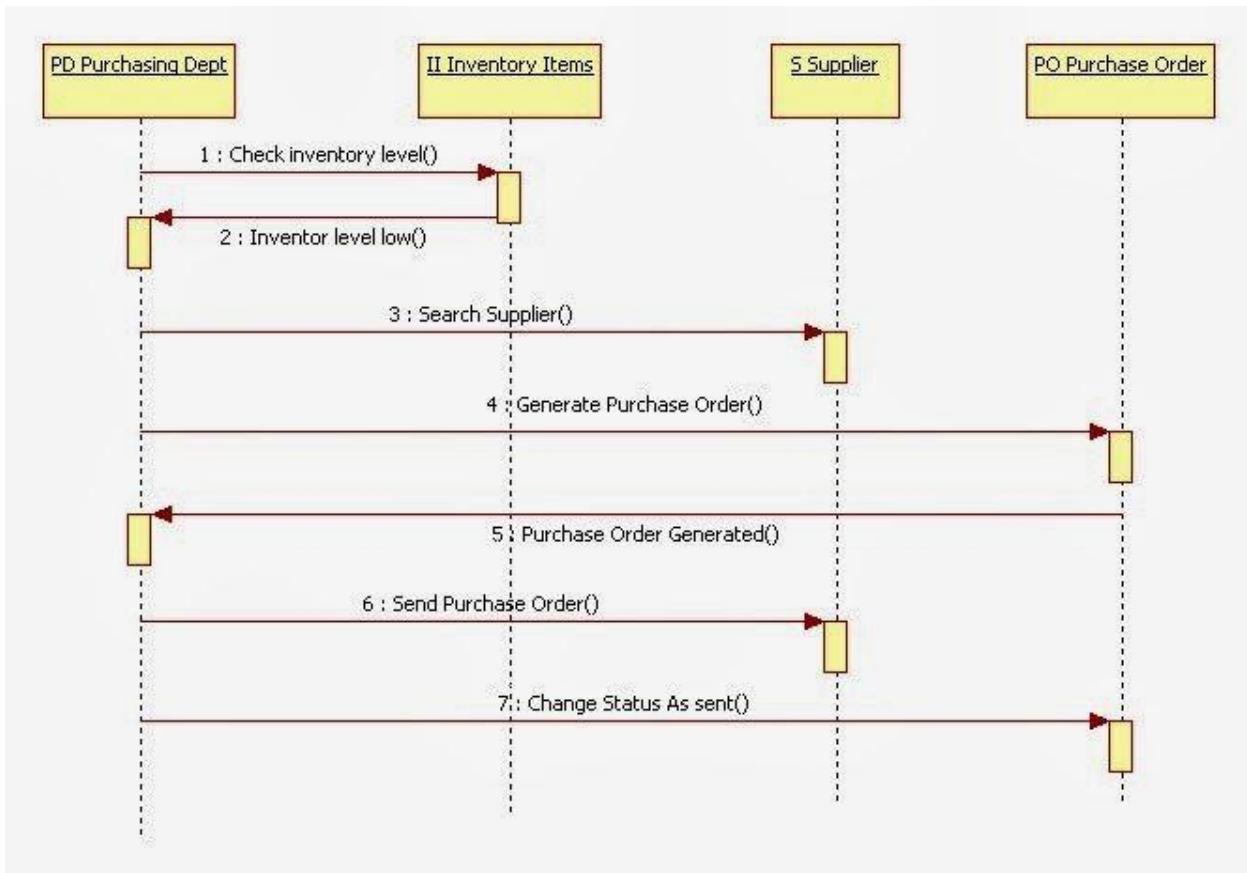
Use case Diagram : Fig 4.3



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

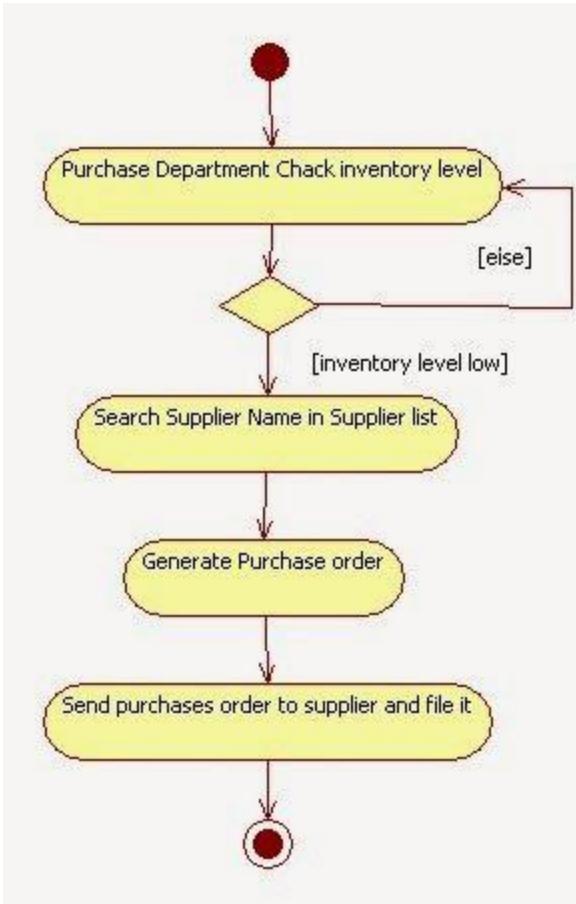
demonstrates how managers, suppliers, cashiers, and customers collaborate with the system to maintain accurate stock levels and complete sales transactions

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 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.

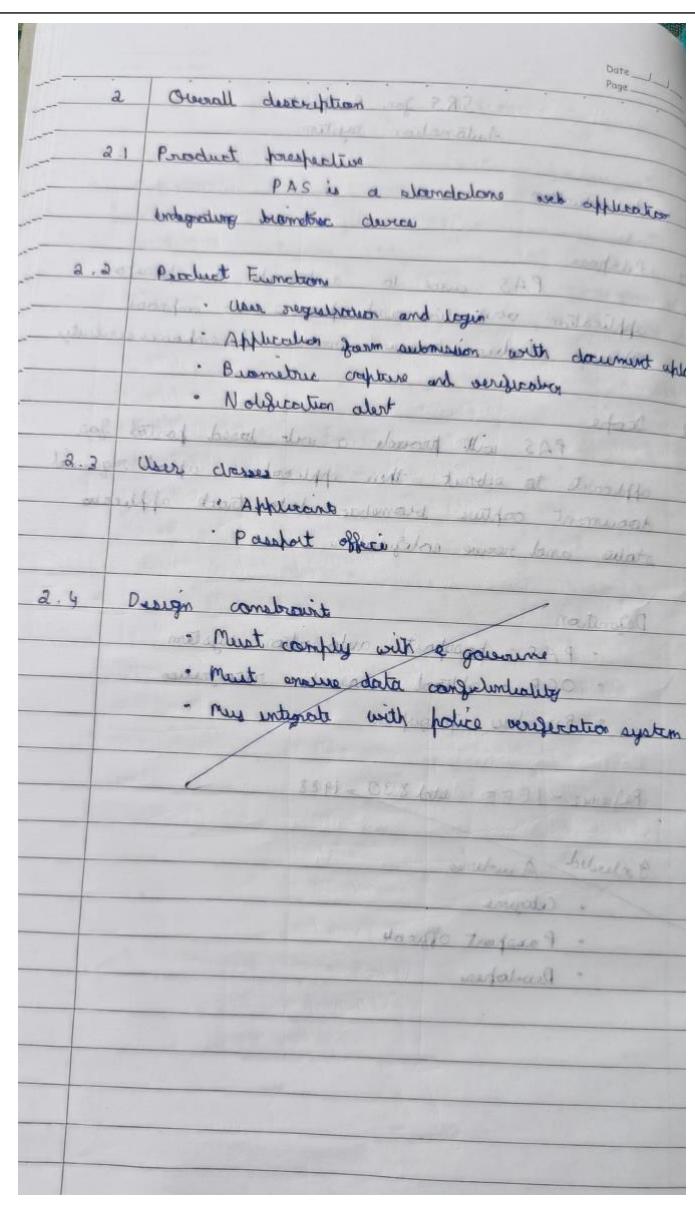
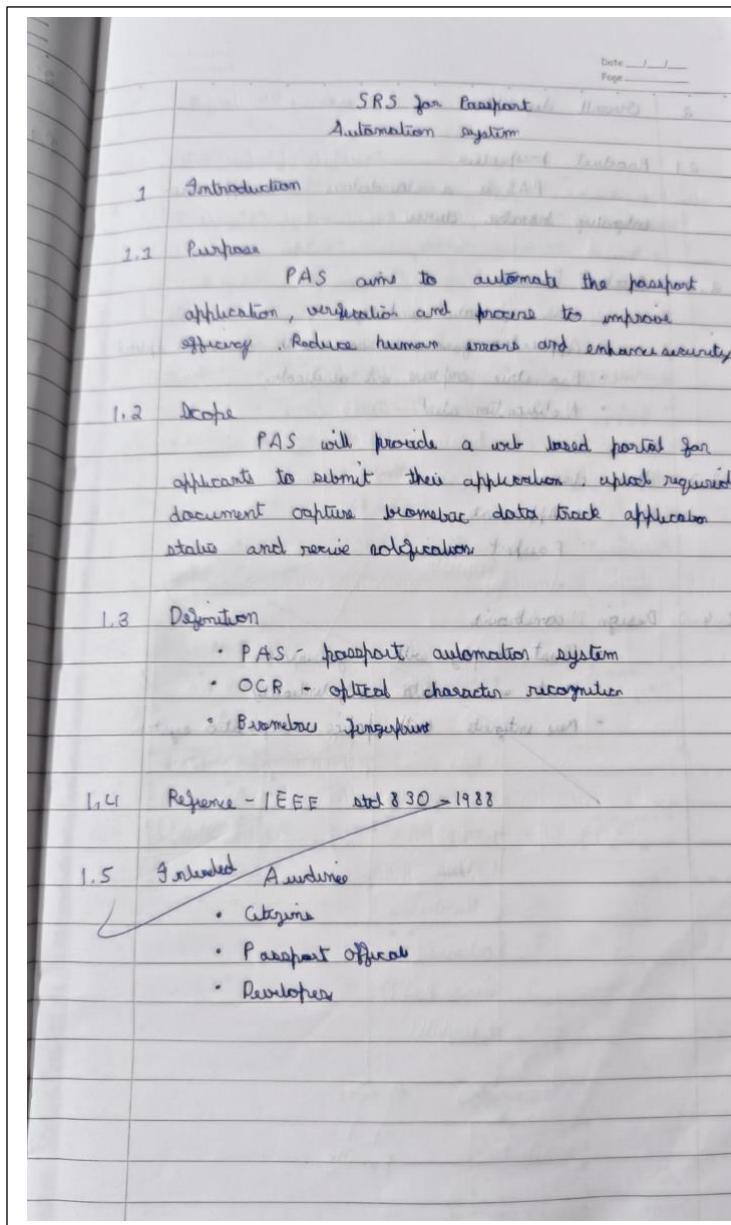
Activity Diagram : Fig 4.5



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.



### 3 System Requirements Engineering Methods

100

### 3.1 Functional requirements

- system shall allow users to regulate and log in
  - system shall allow upload of documents
  - system will capture biometric history +
  - system will provide admin interface +

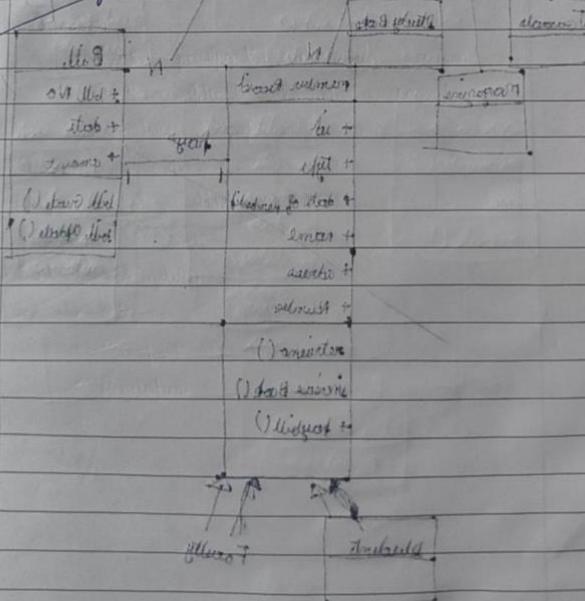
### 3.2 Non-functional based +

- system will maintain 99.9% uptime
  - data shall be integrated with +
  - system shall react to user action +

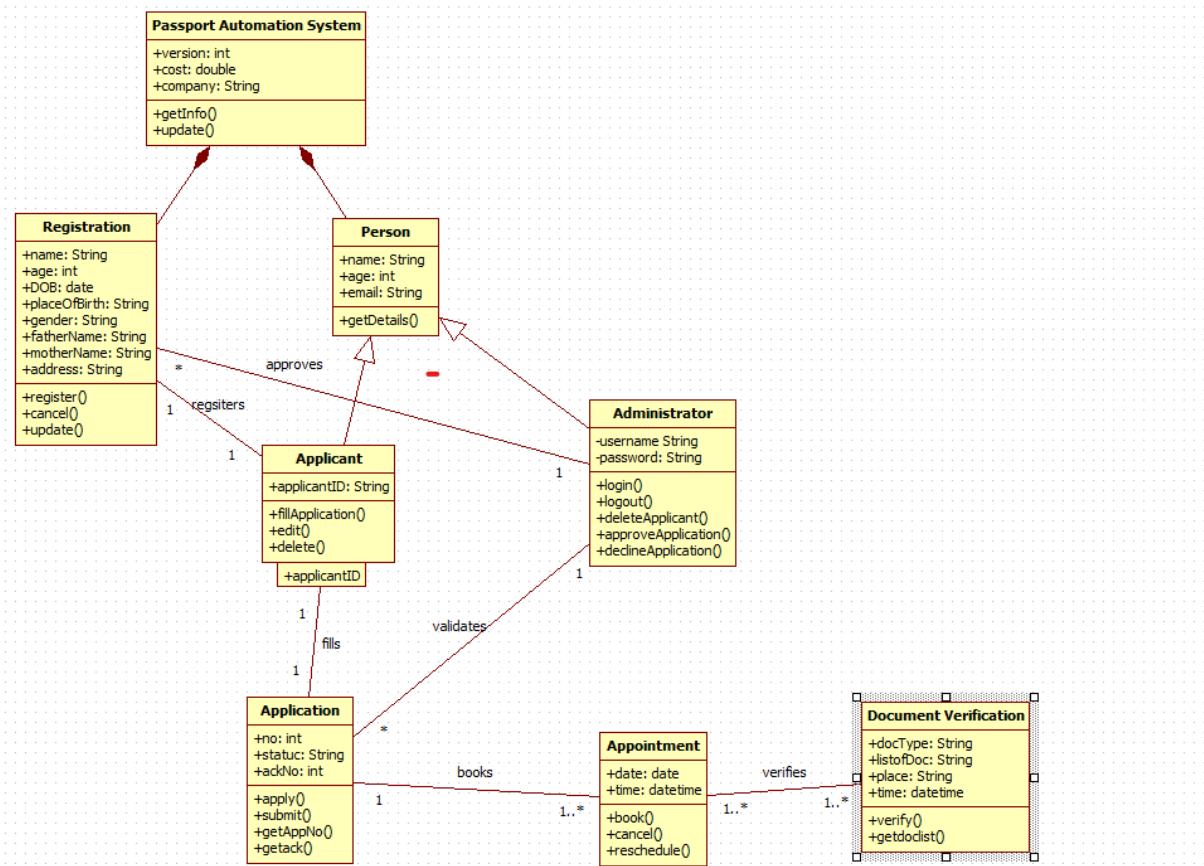
### 3.3 Appendix

- 3.3 Appendix

## 3.4 Glossary

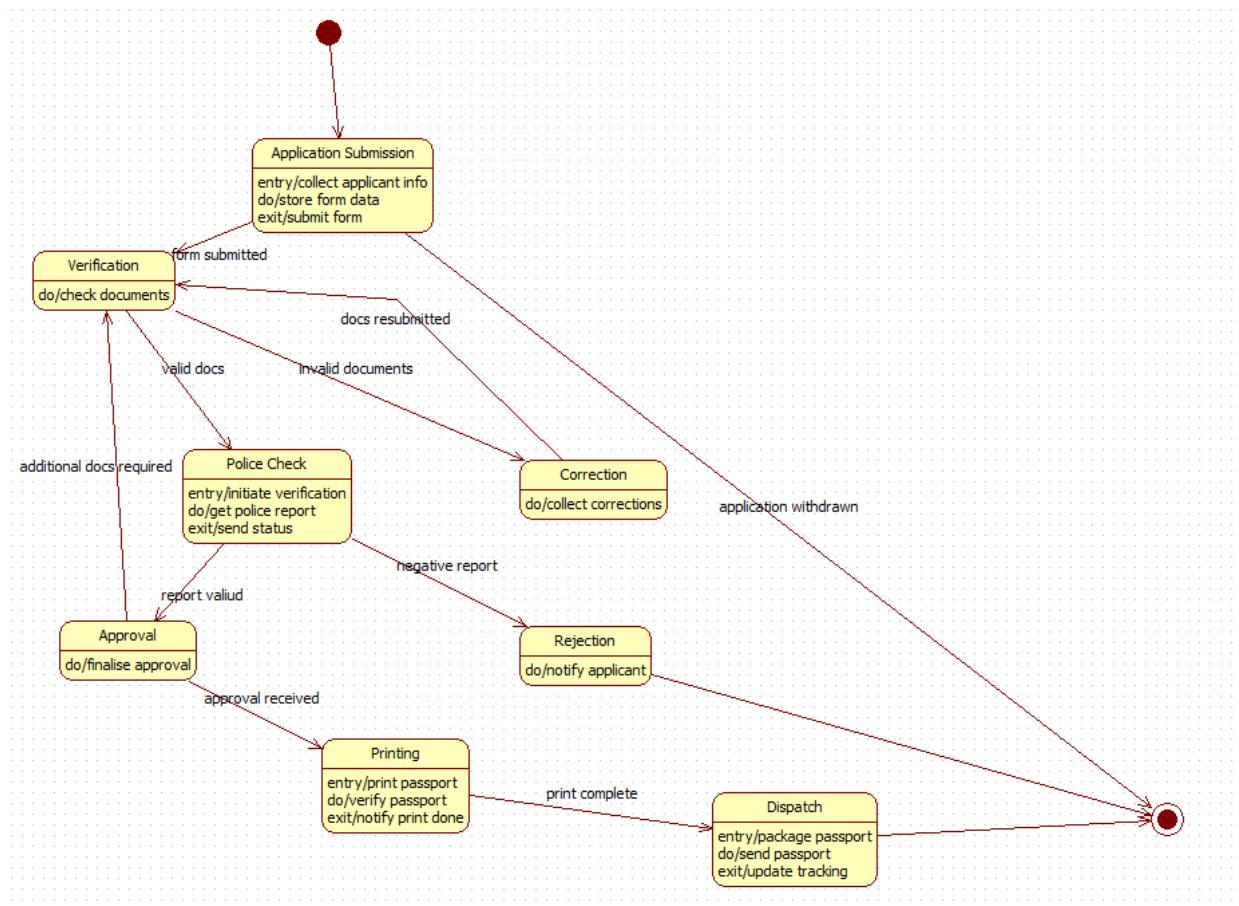


Class diagram : Fig 5.1



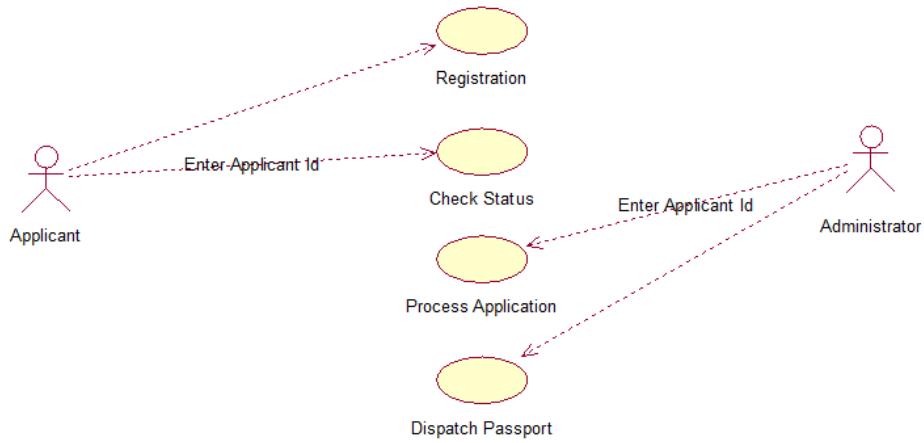
The class diagram for the Passport Automation System illustrates the major components involved in managing passport applications and their interactions. The Person class stores basic personal details and is used by the Registration process to record applicant information. Once registered, an Applicant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



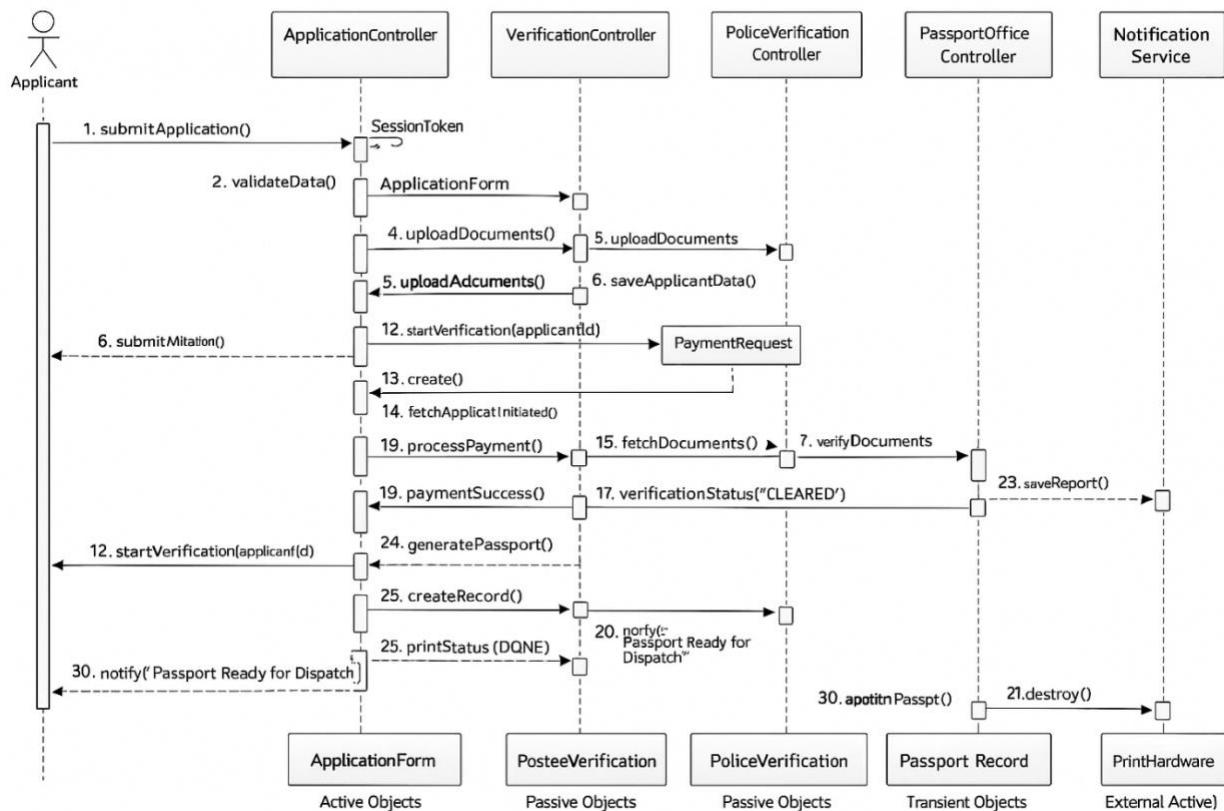
The state diagram illustrates the complete lifecycle of a passport application, beginning with the submission of the applicant's form. After submission, the application enters the verification state, where documents are checked and either accepted, rejected, or sent back for corrections. Valid documents 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



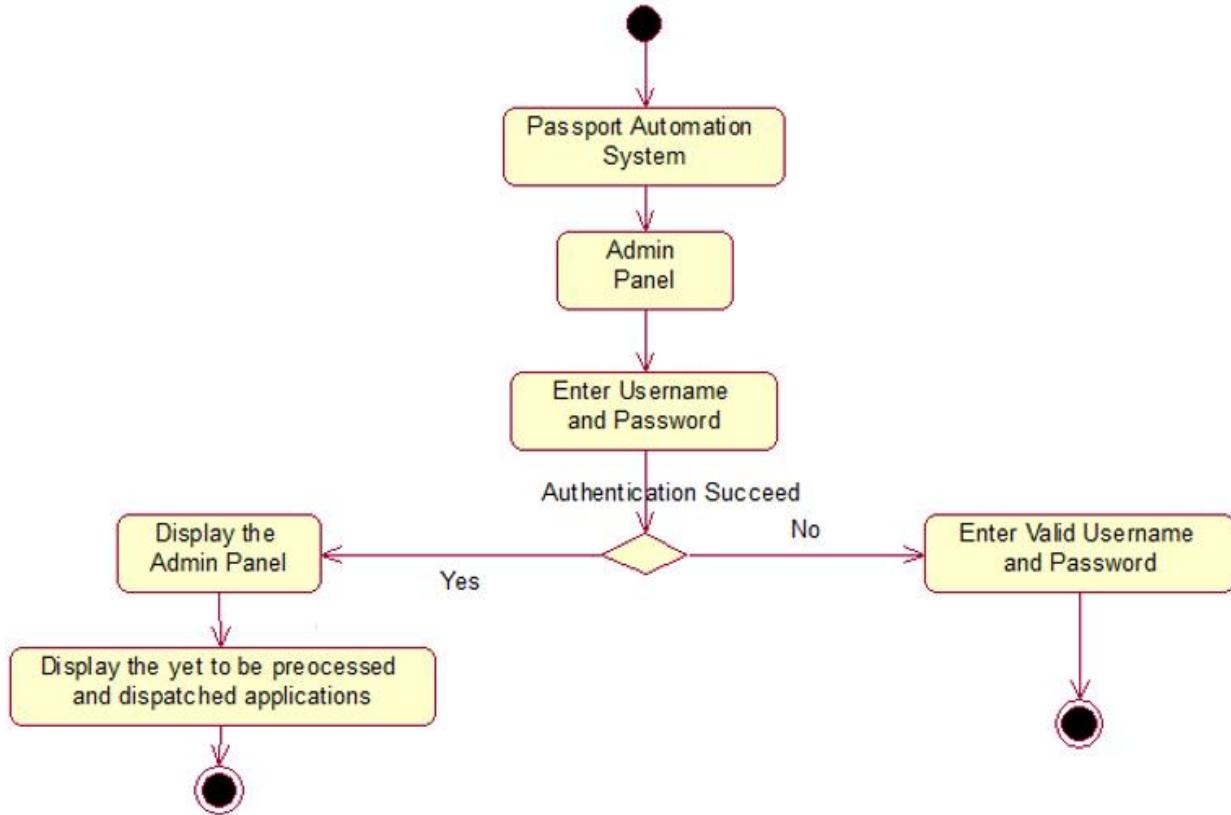
The use-case diagram shows the interaction between the applicant and the administrator during the passport application process. The applicant can register in the system, enter their applicant ID to check the status of their application, and eventually receive their dispatched passport. The administrator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



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.