

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:

Hemanth Kumar R

1BM23CS110

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bengaluru 560019
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 Hemanth Kumar R (1BM23CS110) during the 5th Semester August 2025-December 2025.

Signature
Dr Roopashree S
Assistant Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bengaluru

Table of Contents

1. Hotel Management System	3-9
2. Credit Card Processing	10-16
3. Library Management System	17-23
4. Stock Maintenance System	24-30
5. Passport Automation System	31-37

GitHub Link: https://github.com/HK9876/OOMD_1BM23CS110

1. Hotel Management System

Problem Statement

A hotel needs a computerized system to manage its day-to-day operations efficiently. The current manual process used for managing room bookings, guest check-in/check-out, billing, and staff coordination is time-consuming, error-prone, and difficult to track. To overcome these problems, the hotel requires a Hotel Management System that automates major operational activities and improves service quality.

The system should allow customers to search for available rooms based on type, price, and date. It should support online and offline room reservations. When guests arrive, the receptionist should be able to check them in, assign rooms, and maintain guest records. Upon checkout, the system must generate accurate bills that include room charges, additional services (restaurant, laundry, spa, etc.), and applicable taxes.

SRS-Software Requirements Specification

Software Requirement System	
19/8/25	Date / / Page / /
1. Introduction	2.1 Product Functions Reservation, checkin/out, room & rate management, billing, payments, housekeeping reporting, user/role management, payment integration
1.1 Purpose: This SRS defines functional and non-functional requirements for Hotel Management System for stakeholders.	2.2 User classes and characteristics Guests, Receptionist, Housekeeping staff, Manager, System Administrator
1.2 Document Conventions: shall - mandatory, should - recommended, may - optional	2.3 Operating Environment Server: Modern browsers
1.3 Intended audience Project sponsor, developers, testers, end users	2.4 Design and Implementation constraints Comply with data protection laws, ACID financial transactions
1.4 Project Scope: HMS shall provide web application for single hotels covering bookings, checkin/checkout, billing, housekeeping, payment interface.	2.5 Used Documents API Docs, privacy policy
1.5 References IEEE SRS format	2.6 Assumption and Dependency Payment gateway availability, hotel provides internet
1. Overall description	3. Specific Requirements
2.1 Product Perspective: Application with web UI, REST API, DB.	3.1 Functional Requirements: FR-1 (Booking) shall create, modify or cancel reservations. FR-2 (Check In-out) shall support check in and check out. FR-3 (Payments): shall process payment

Date _____
Page _____
ng
via integrated gateway.
FR-1 (Room and rate management) shall
manage rooms, room types, rates,
rate plans.

3.2 Non-functional requirement

- NFR-1: Performance - Shall respond to 01 actions within 2s for 95% operations.
- NFR-2: Scalability - System should be scalable to many users.
- NFR-3: Backup Recovery - Daily backups.
- NFR-4: Security - Sensitive data shall not be stored.

3.3 External Interface Requirement

- UI - responsive web screens for all purposes.
- Payment gateway - secure API integrations.
- Peripherals - support for PDF printing (invoices / receipts).

3.4 Appendices

Glossary (OTA - Online Travel Agency)
HMS - Hotel Management System

Class Diagram

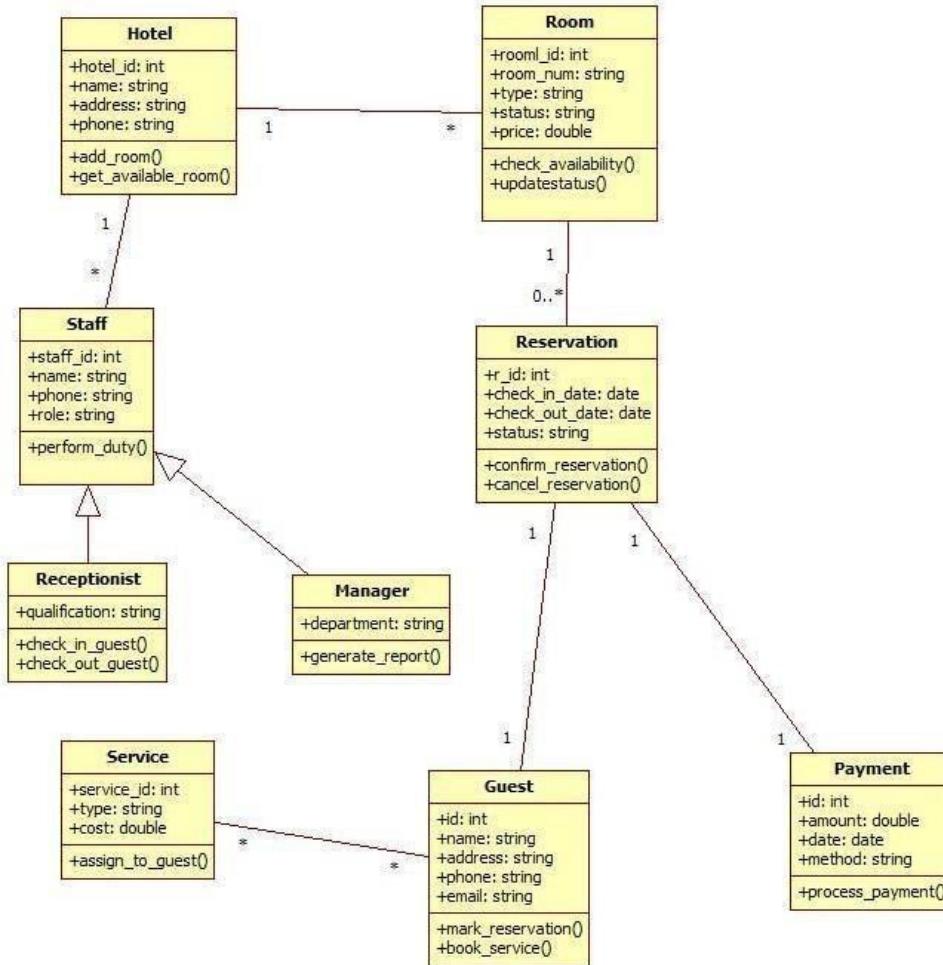


Figure 1.1: Class Diagram for Hotel Management System

A hotel management system keeps track of hotels, their rooms, and the staff who operate them. Each hotel has multiple rooms, and every room stores details such as room number, type, status, and price. Guests can make reservations for specific rooms, and each reservation contains check-in and check-out dates along with its status. Guests can also book additional services offered by the hotel, such as room service or laundry, and all payments made by guests are recorded with details like amount, date, and method. Staff members, including receptionists and managers, perform various duties such as checking guests in or generating reports, and every staff member belongs to a hotel. The system ensures that room availability is updated through reservations, guests can manage their bookings, and staff can perform their roles efficiently.

State Diagram

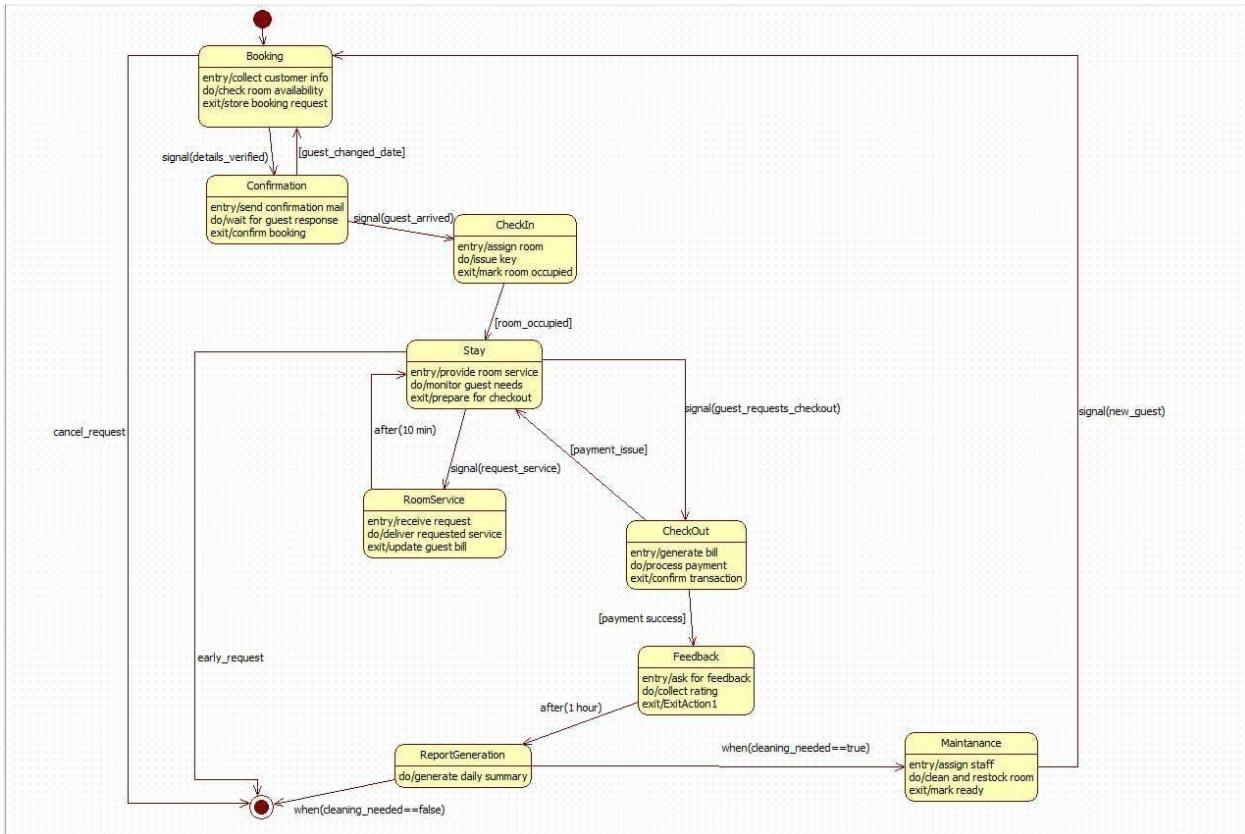


Figure 1.2: State Diagram for Hotel Management System

The hotel guest process begins with a booking stage where customer information is collected and room availability is checked, after which the request moves to confirmation, where the system sends a confirmation email and waits for the guest's response. Once the guest arrives, the process transitions to check-in, where a room is assigned and the key is issued before the guest enters the stay phase. During the stay, the hotel monitors guest needs and handles room-service requests, which temporarily move the workflow to a dedicated room-service state before returning to the stay state. When the guest requests checkout, the process moves to the checkout state, where the bill is generated and payment is processed; upon successful payment, the guest is directed to provide feedback. After checkout, if cleaning is required, the room enters a maintenance state where staff clean and restock the room before marking it ready; otherwise, the system proceeds to generate daily reports and waits for the next new guest to begin the cycle again.

Use Case Diagram

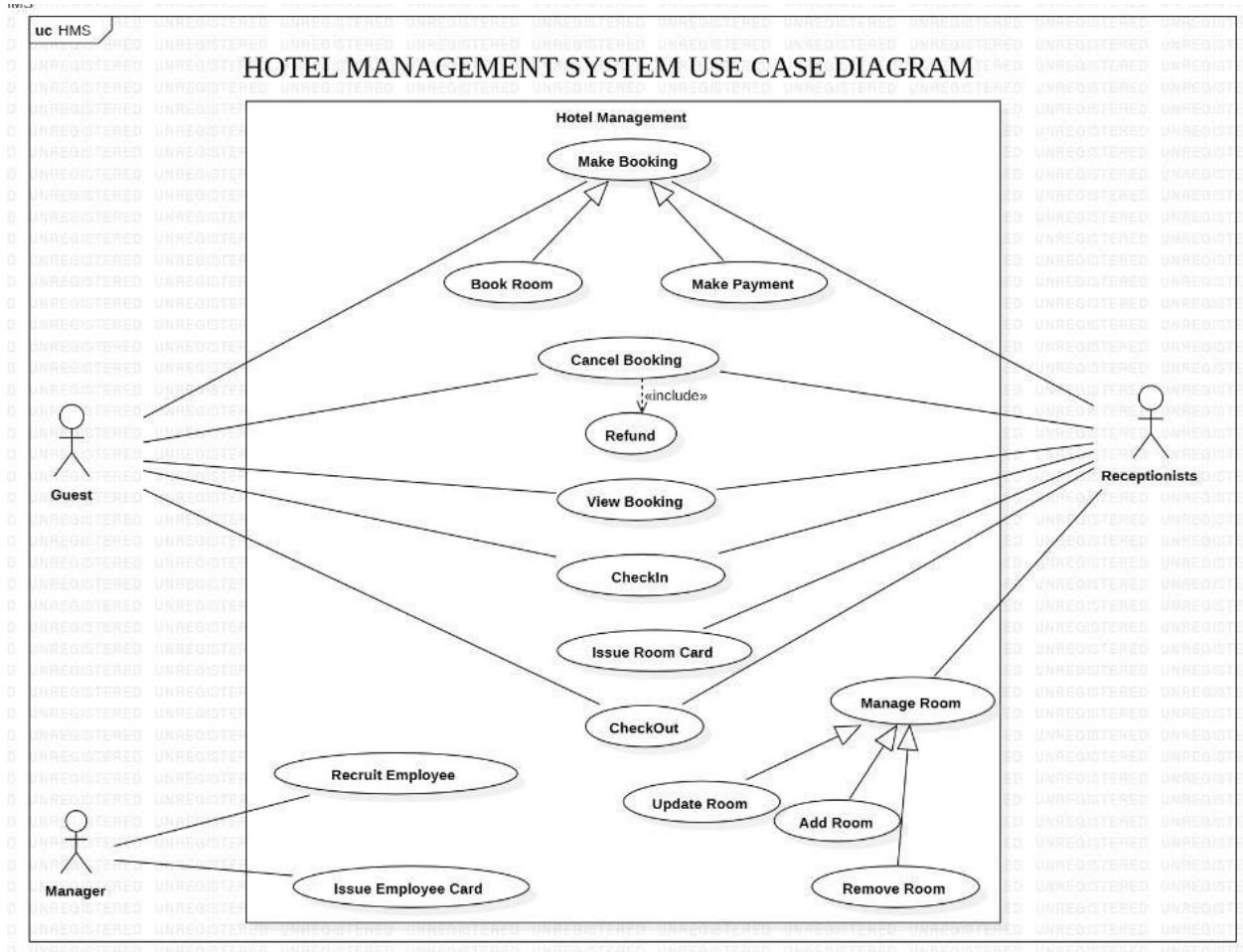


Figure 1.3: Use Case Diagram for Hotel Management System

The hotel management system allows guests to book rooms, make payments, cancel bookings, request refunds, view their bookings, check in, receive a room card, and check out. Receptionists support many of these operations by handling room bookings, payments, check-ins, issuing room cards, and managing room-related tasks such as updating, adding, or removing rooms. The manager, on the other hand, oversees staff-related functions including recruiting employees and issuing employee cards. The system organizes all these interactions under the broader hotel management process, ensuring that guests can access essential services while staff and managers maintain hotel operations smoothly.

Sequence Diagram

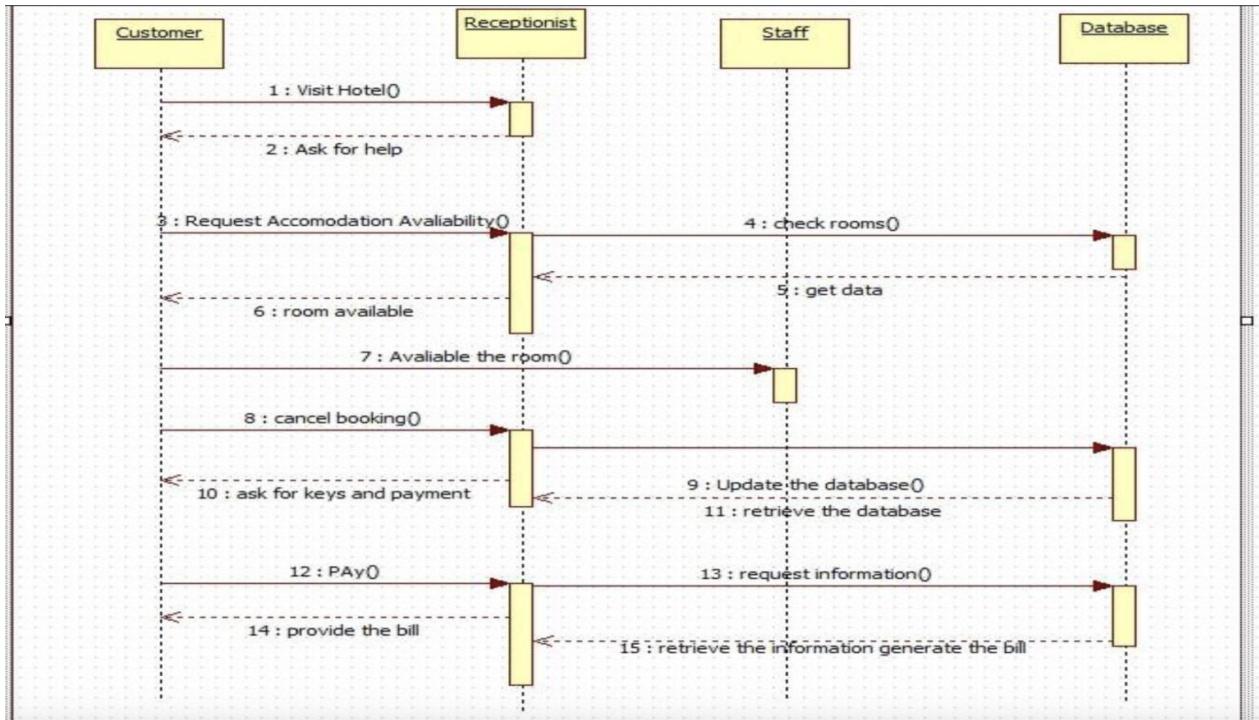


Figure 1.4: Sequence Diagram for Hotel Management System

In this hotel sequence, a customer first visits the hotel and asks the receptionist for assistance, then requests information about room availability. The receptionist contacts the staff, who check the rooms by querying the database and return the data, allowing the receptionist to inform the customer that a room is available. If the customer decides to cancel a booking, the receptionist updates the database accordingly. Later, when the customer asks for the room keys and proceeds to make payment, the receptionist requests billing information from the database, which retrieves the necessary details so the receptionist can provide the final bill to the customer.

Activity Diagram

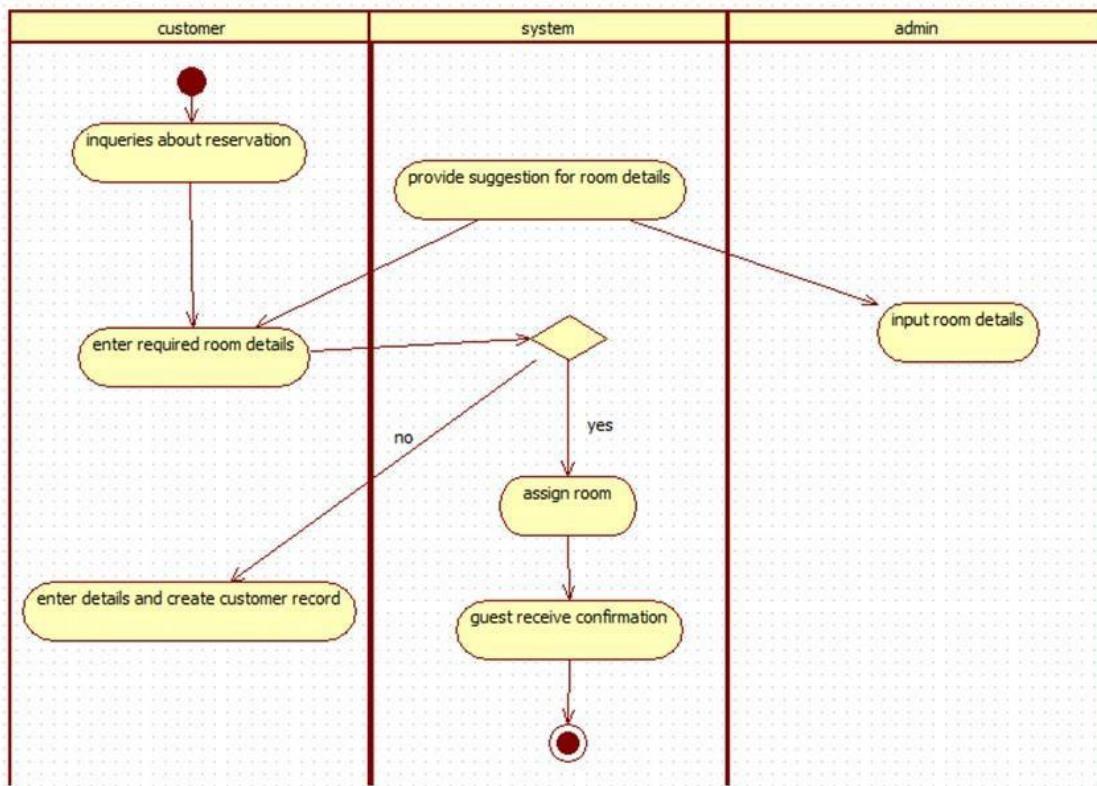


Figure 1.5: Activity Diagram for Hotel Management System

The activity begins when a customer inquires about making a reservation and proceeds to enter their required room details. The system then provides suggestions for suitable rooms while the admin inputs room information to help the system make a decision. Based on these details, the system checks if the room requirements can be met; if not, the customer is prompted to re-enter their details and a customer record is created. If the room is available, the system assigns the room and sends a confirmation to the guest, completing the reservation process.

2. Credit Card Processing

Problem Statement:

The Credit Card Processing System should allow cardholders to perform transactions such as purchases, cash withdrawals, balance inquiries, and bill payments. When a transaction is initiated, the system must verify card details, validate the cardholder's identity, check available credit limits, and ensure that the transaction is not fraudulent. The system must then approve or decline the transaction based on predefined rules and update the cardholder's account accordingly. Additionally, the system should generate periodic statements, track outstanding balances, calculate interest, and maintain accurate records of all transactions.

SRS-Software Requirements Specification

Credit Card Processing	
1. Introduction	UI, audit logging
1.1. Purpose : This SRS defines requirements for credit card processing module to securely tokenise, authorise, capture, refund,	2.3. User classes Integrating app : calls REST API, handles tokens End customer: Enters card data Finance Manager : uses Admin UI for reports and refunds.
1.2. Document Conventions : shall - mandatory, should - recommended may - optional	2.4. Operating Environment Clients - Browsers or mobile Database - Encrypted DB.
1.3. Intended Audience : Developers, testers, users	2.5. Design and implementation constraints Use gateway tokenisation to minimise PCI scope, idempotency keys for financial endpoints, all traffic over TLS
1.4. Project Scope : Tokenisation via hosted fields, authorise, capture, sale, refunds, recurring charges.	2.6. Used Documents Privacy Policy, Integration guide
1.5. References IEEE SRS format.	2.7. Assumption and Dependencies Merchant account provided, legal requirements supplied by notar.
2. Overall description	3. Specific Requirements
2.1. Product perspective : Modular backend service used by opp servers. Uses a trusted payment gateway for actual card handling	3.1. Functional Requirements
2.2. Production Functions : Tokenise card, authorize, refund, admin	✓ FR-1 Tokenisation : System shall accept gateway issued tokens from client hosted fields.

	Date _____	Page _____
FR-2: Authorize :	shall authorize specified amount and return auth-id + status	
FR-3: Refund:	shall refund captured transactions and provide refund	
FR-4: Error Handling:	shall return structured error codes and indicate retryable vs non-retryable errors.	
3.2. Non-Functional Requirements		
NFR-1 Performance:	latency $\leq 3s$ for 95% request.	
NFR-2 Availability:	99.9% uptime	
NFR-3 Scalability:	Horizontally scalable API layer	
NFR-4 Privacy:	Customer consent for token reuse stored and auditable	
3.3 External interface requirements		
Backend API:	REST API end points	
Admin UI:	Transaction search, CSV export	
Gateway:	Server-to-Server HTTPS integration	
3.4 Appendices:		
Glossary,		
PCI - payment card industry		

Class Diagram

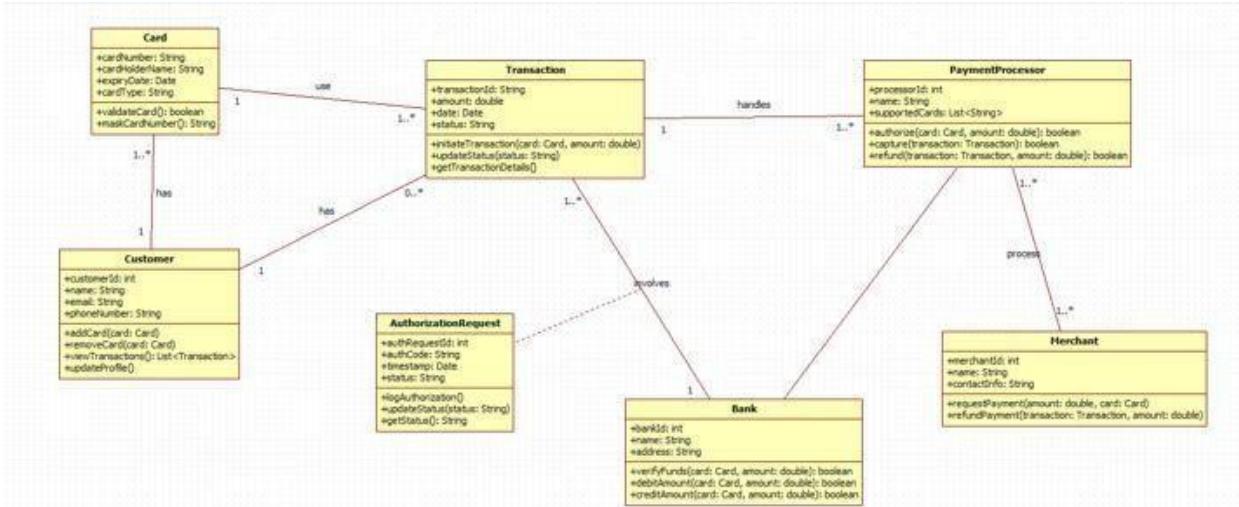


Figure 2.1: Class Diagram for Credit Card Processing

This payment processing system manages how customers use their cards to make transactions with merchants. Each customer owns one or more cards, and when a transaction is initiated, the card information is used to create a transaction request. The payment processor handles these transactions by forwarding authorization requests to the bank, which verifies card details, checks account status, and approves or declines the request. Once authorized, the transaction is processed for the merchant, who can also request refunds through the system. Authorization requests record details such as status and time, while the bank maintains operations for authorization, refund, and transaction validation. Throughout the process, all components—customer, card, payment processor, bank, and merchant—interact to ensure secure and accurate handling of payments.

State Diagram

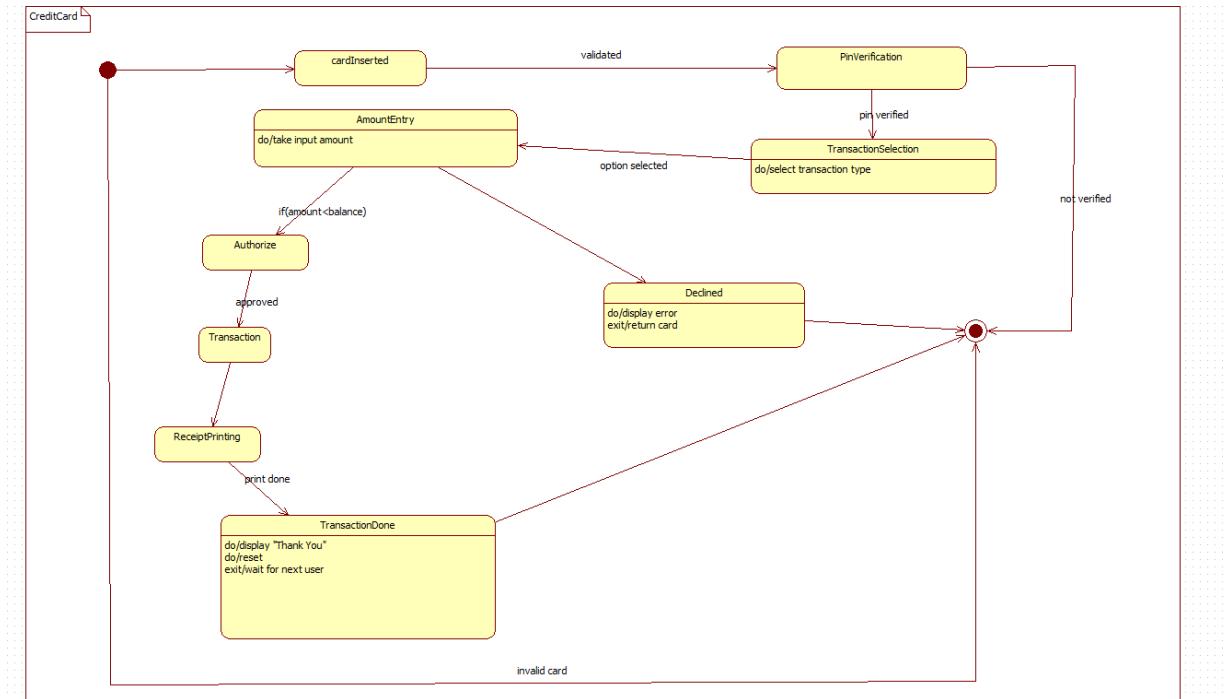


Figure 2.2: State Diagram for Credit Card Processing

The credit card transaction process begins when a user inserts their card, after which the system validates it and prompts for PIN verification. If the PIN is correct, the user proceeds to select the type of transaction and then enters the required amount. The system checks whether the entered amount is within the available balance; if it is, the transaction is authorized, executed, and followed by receipt printing. Once the receipt is generated, the system displays a thank-you message, resets itself, and waits for the next user. If the card is invalid, the PIN is incorrect, or the amount exceeds the balance, the process moves to a declined state where an error is displayed and the card is returned, ending the workflow.

Use Case Diagram

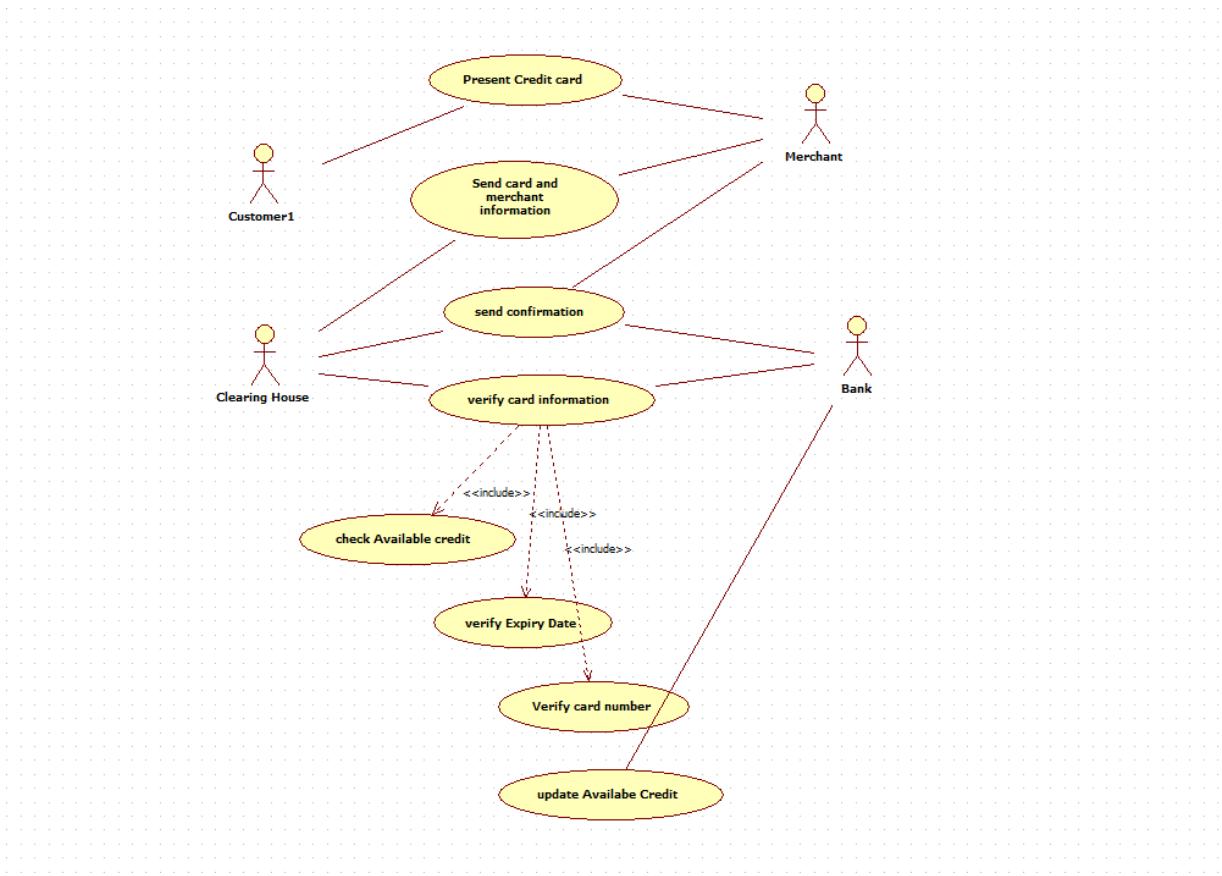


Figure 2.3: Use Case Diagram for Credit Card Processing

In this credit card authorization process, a customer presents their credit card to the merchant, who sends the card and transaction details to the clearing house for verification. The clearing house then communicates with the bank to verify the card by performing several checks, including validating the card number, confirming the expiry date, and checking whether sufficient available credit remains. Once these checks are completed, the clearing house sends a confirmation back to the merchant about whether the transaction is approved, and the bank updates the available credit accordingly. This cooperation among customer, merchant, clearing house, and bank ensures that every card transaction is securely authenticated before completion.

Sequence Diagram

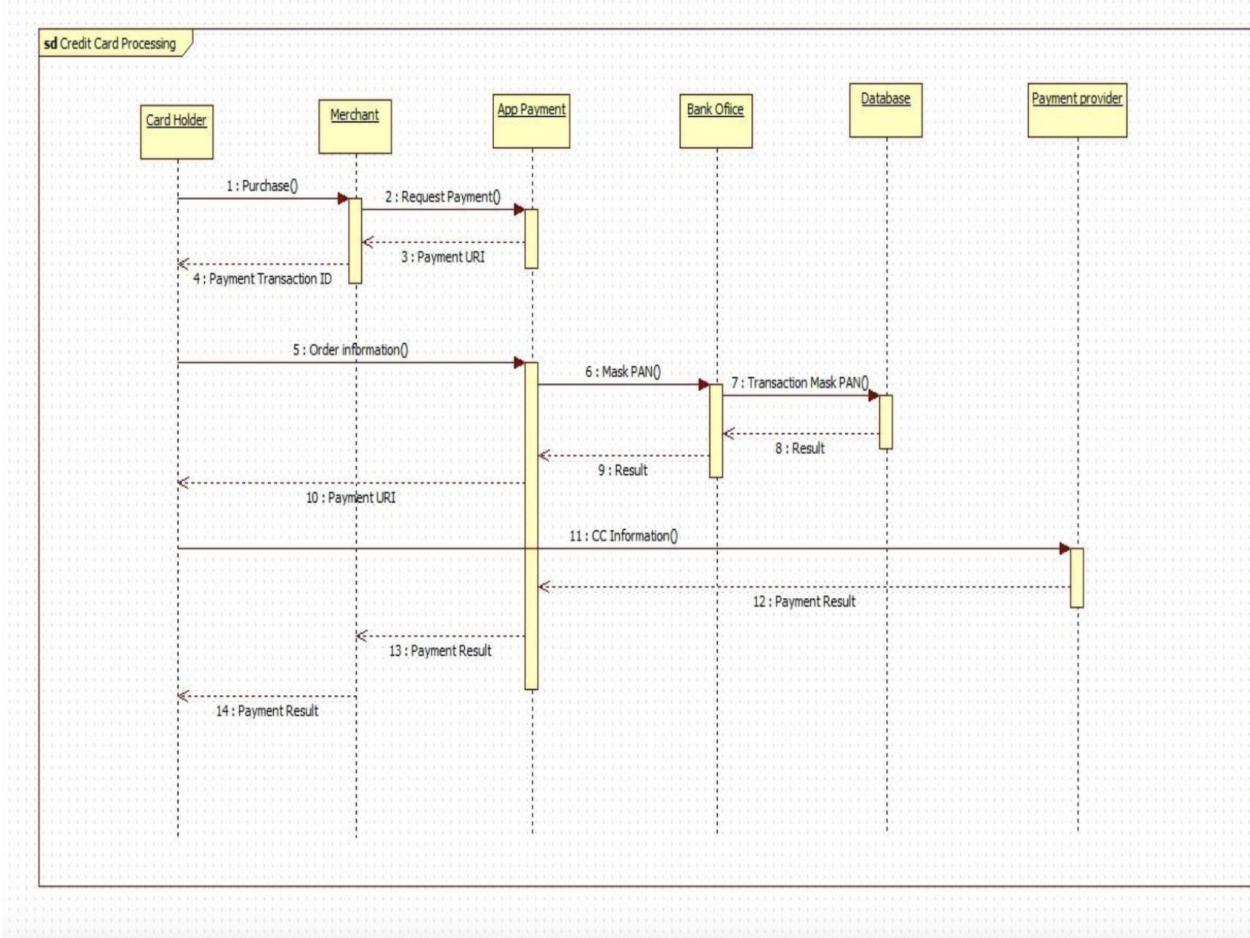


Figure 2.4: Sequence Diagram for Credit Card Processing

In this credit card processing sequence, the card holder begins a purchase request, which the merchant forwards to the app payment system to obtain a payment URL. After receiving the payment transaction ID and payment URL, the merchant sends the customer's order information to the bank office, which masks the card number (PAN) and forwards the masked transaction details to the database for verification. The database returns the result to the bank office, which passes it back through the app payment system along with the credit card information. The app payment system processes the details with the payment provider, receives the final payment result, and returns it to the merchant, who ultimately notifies the customer of the completed transaction.

Activity Diagram

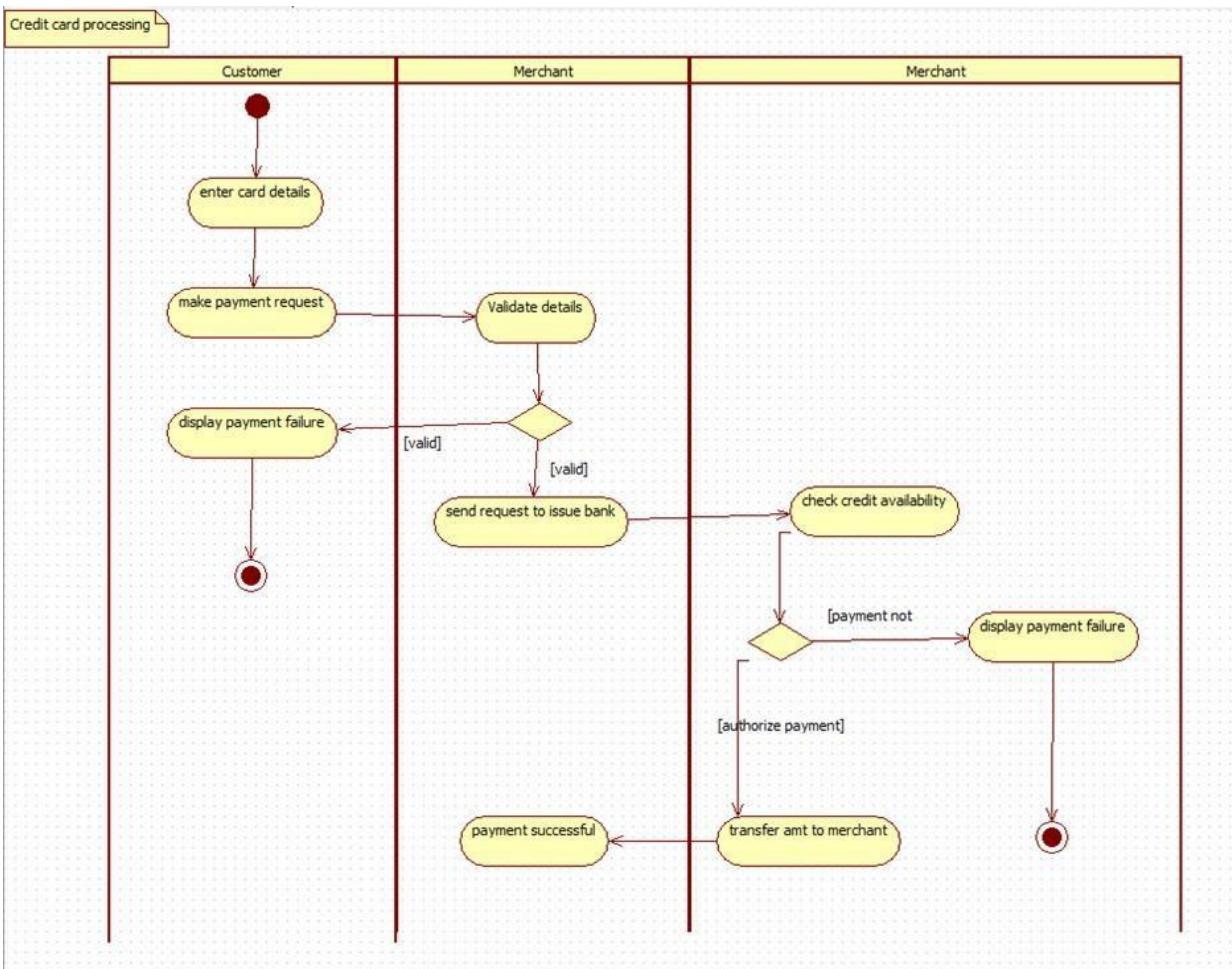


Figure 2.5: Activity Diagram for Credit Card Processing

The credit card payment process begins when the customer enters their card details and submits a payment request to the merchant. The merchant first validates the provided information; if the details are invalid, the system immediately displays a payment failure message to the customer. If the details are correct, the merchant forwards the request to the issuing bank, where the system checks the availability of credit. If sufficient credit is not available, the bank notifies the merchant, who then informs the customer of the payment failure. When credit is available and the payment is authorized, the bank transfers the approved amount to the merchant, and the system confirms that the payment was successful, completing the process.

3. Library Management System

Problem Statement:

Libraries today manage large volumes of books, journals, digital media, and member records. Many libraries still rely on manual methods such as record books, paper-based membership files, and handwritten issue-return logs. These manual processes are time-consuming, prone to human errors, difficult to update, and inefficient when handling large numbers of users and resources. To overcome these limitations and modernize library operations, an automated Library Management System is required.

The Library Management System should enable librarians to efficiently manage the acquisition, cataloging, tracking, and distribution of books and other resources. The system must allow members to search for books, check availability, borrow and return items, place reservations, and receive notifications for due dates or overdue fines. Librarians should be able to register new members, maintain member details, add or remove books, update catalog information, manage book categories, handle book issues, enforce return deadlines, calculate fines for late returns, and generate various reports related to inventory and member activity.

SRS-Software Requirements Specification

SRS	
	Date _____ Page _____
Library Management System	
Needs secure management of lending books and having account of the due, efficient management of cataloging, lending, fine collection. Automated system improve resource efficiency.	
1. Introduction	
1.1. Purpose	Defines requirements for library management system to automate book lending and cataloging.
1.2. Scope	Improve efficiency by digitizing library processes.
1.3. Overview	System manages books, users, borrowing, fines and reports.
2. General description:	Objectives: Digitize library operations. Users: Students, staff, admin. Features: Search, borrow, return, report. Benefits: Time saving, better service.
3. Functional requirements:	<ul style="list-style-type: none">Register library membersMaintain digital catalog
	<ul style="list-style-type: none">Borrow and return trackingFine calculationReport generation
	4. Interface requirement
	Barcode scanner support
	Online catalog access
	5. Performance Requirement
	Catalog search <sec
	6. Design constraints
	Multi branch library support
	7. Non-functional requirement
	Security, User authentication, Reliability, Data backup support, Usability: Simple UI
	8. Schedule and budget
	Development: 4-6 months Cost: Low to medium

Class Diagram

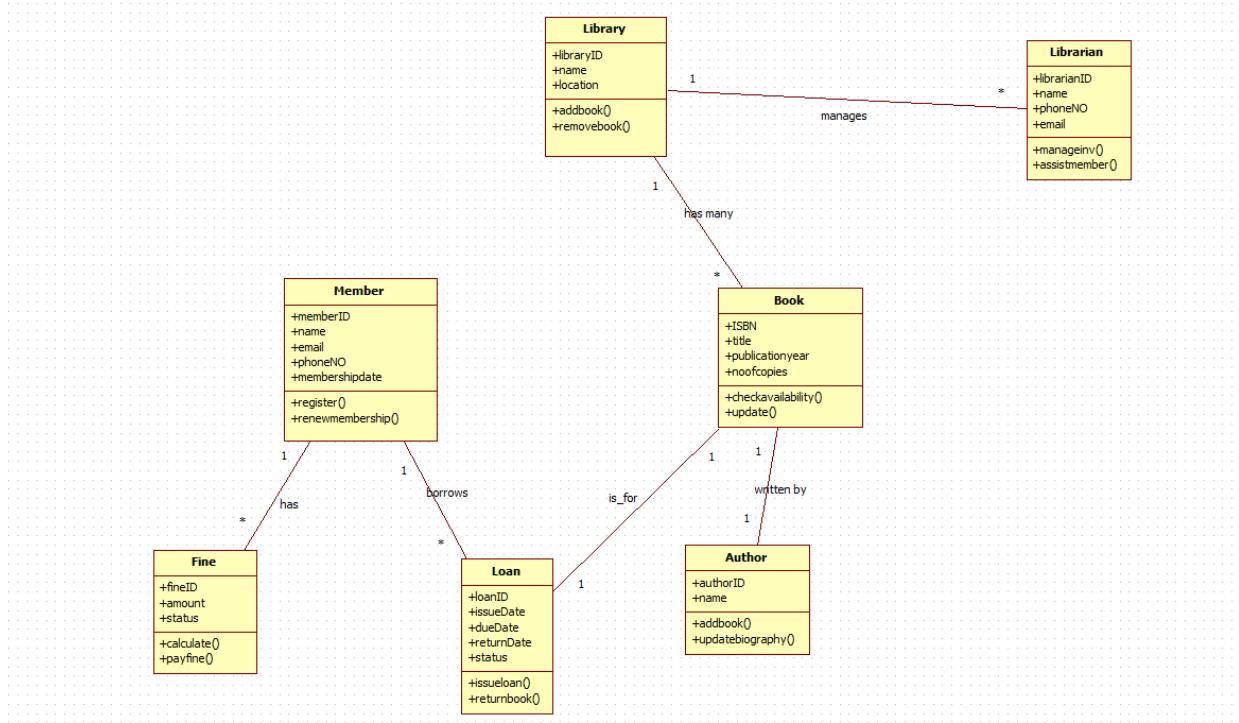


Figure 3.1: Class Diagram for Library Management System

In this library management system, a library contains many books, each written by one or more authors and managed by librarians who oversee book records and assist members. Members can register with the library and borrow books through loan records that track issue dates, due dates, and return dates. If a member returns a book late or violates library rules, fines are generated and linked to the member. The system ensures that book availability is updated, loans are properly recorded, fines are calculated, and librarians can manage inventory and support member needs, while authors and their books are also maintained for accurate cataloging.

State Diagram

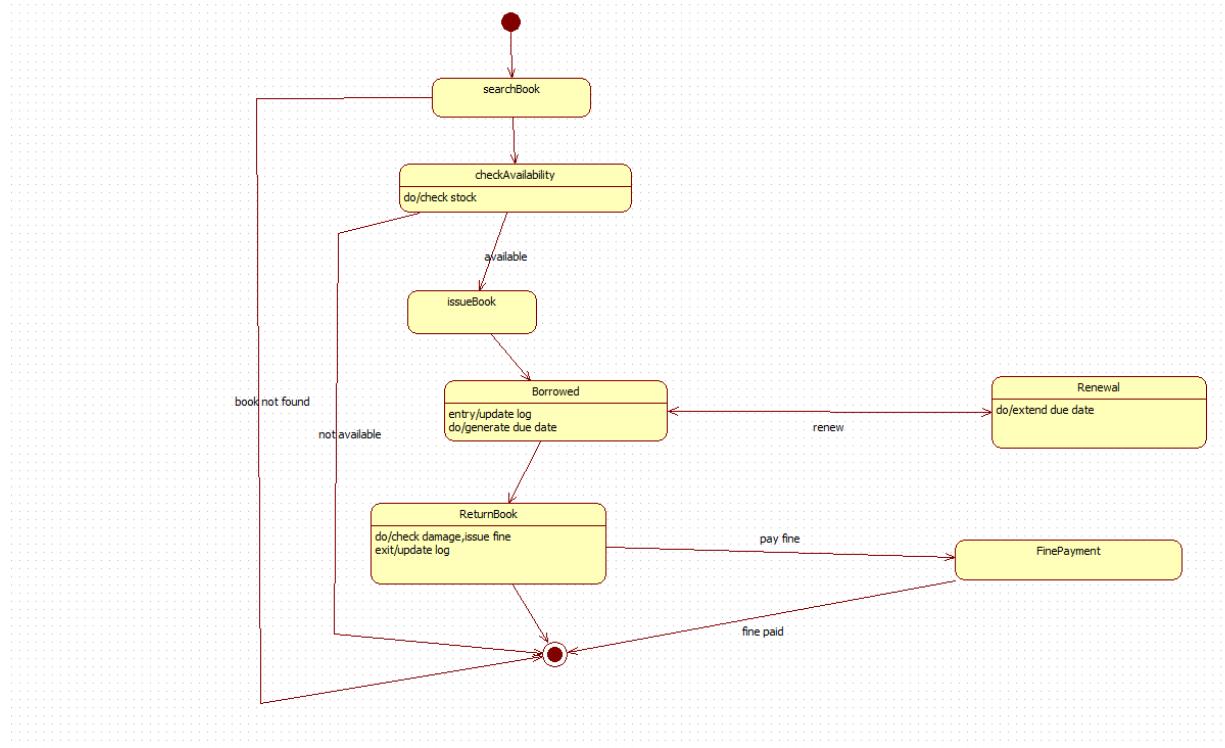


Figure 3.2: State Diagram for Library Management System

The book-borrowing process starts when a user searches for a book, after which the system checks its availability in the library's stock. If the book is found and available, it is issued to the user, who enters the borrowed state where the system logs the transaction and generates a due date. From this state, the user may choose to renew the book, extending the due date, or return it. Upon return, the system checks for any damage or late submission and issues a fine if necessary; the user can then proceed to pay the fine, after which the record is updated and the process ends. If the book is not found or unavailable at the start, the system immediately terminates the process with an appropriate message.

Use Case Diagram

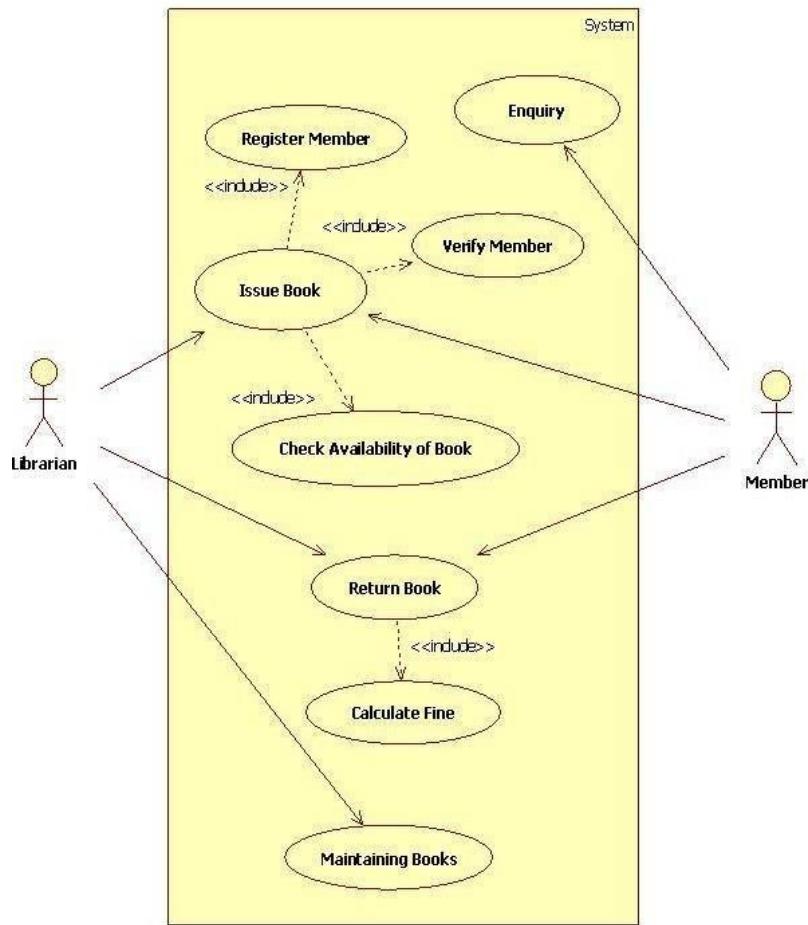


Figure 3.3: Use Case Diagram for Library Management System

In this library management system, members can make enquiries about books and have their membership verified before borrowing any items. The librarian is responsible for registering new members, issuing books, checking the availability of requested books, and processing book returns. When a book is returned, the system calculates any applicable fines for late or damaged returns. The librarian also oversees maintaining the collection, ensuring that books are properly managed and updated in the system. Each major task, such as issuing books or returning them, includes necessary supporting actions like verifying membership or checking book availability to maintain smooth and accurate library operations.

Sequence Diagram

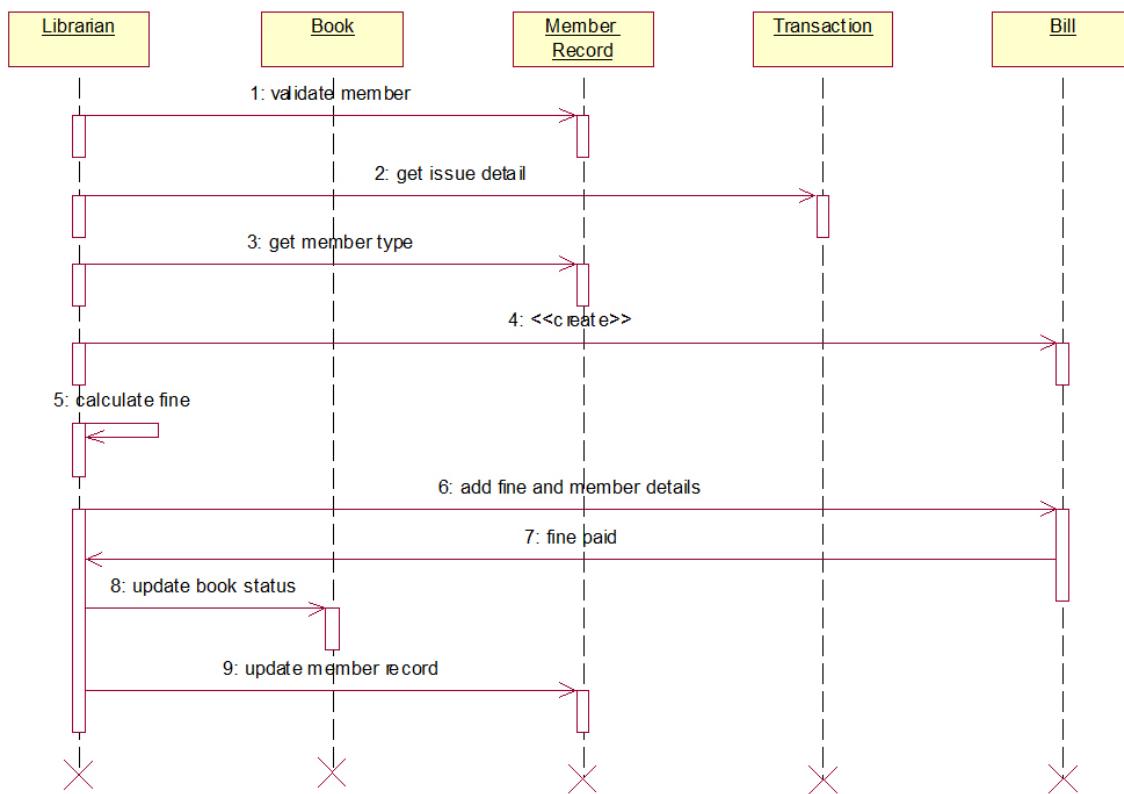


Figure 3.4: Sequence Diagram for Library Management System

In this fine-processing sequence, the librarian begins by validating the member and then requests issue details from the member record, followed by retrieving the member type to determine applicable rules. A fine calculation process is initiated, and the system creates a transaction entry before adding the fine and member information to the transaction. Once the member pays the fine, the system updates the status of the book and then updates the member record to reflect the completed transaction, concluding the process.

Activity Diagram

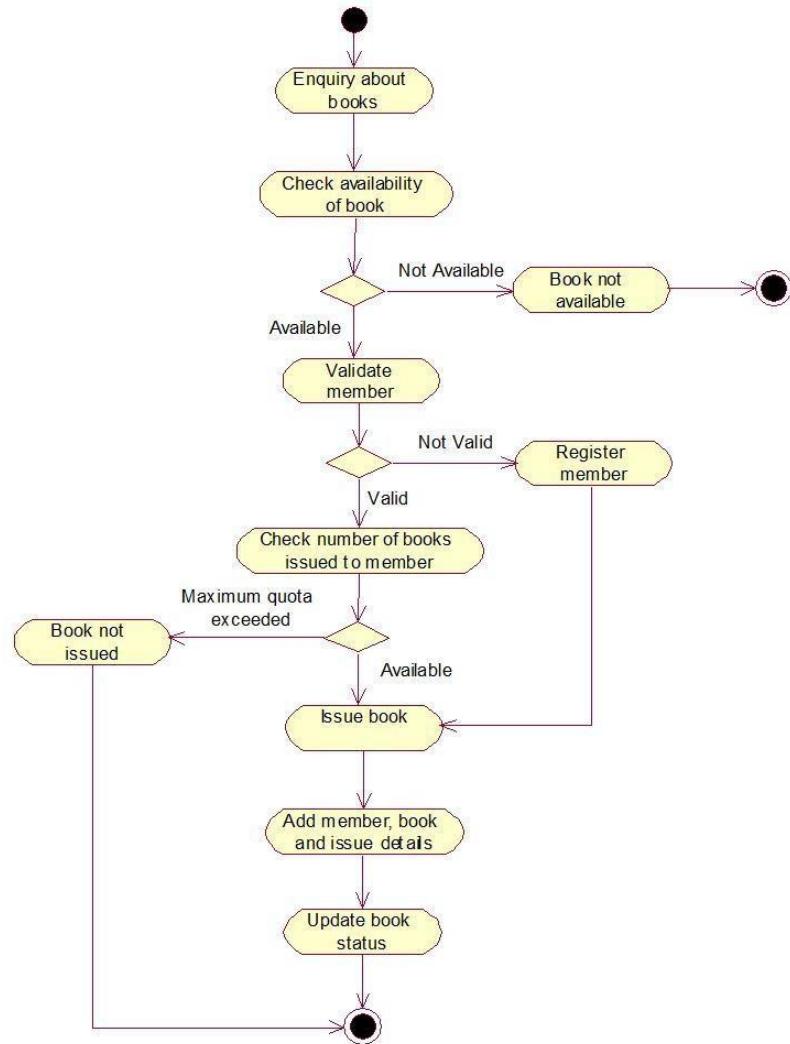


Figure 3.5: Activity Diagram for Library Management System

The library book-issuing process begins when a user makes an enquiry about a book, prompting the system to check its availability. If the book is not available, the process ends immediately. If available, the system proceeds to validate the member; an invalid member is redirected to the registration process before continuing. Once validated, the system checks how many books the member has already borrowed; if the borrowing limit is exceeded, the book cannot be issued. If the member is eligible, the book is issued, and the system records the member, book, and issue details before updating the book's status, successfully completing the transaction.

4. Stock Maintenance System

Problem Statement

The Stock Maintenance System should enable businesses to track stock quantities, monitor product movements, and maintain updated records of items entering or leaving the inventory. The system must allow users to add new stock items, categorize products, update quantities, record purchases, process stock issues, and generate alerts when stock levels fall below a specified threshold. It should also support search and inquiry functions so that users can quickly check availability, item details, supplier information, and stock status in real time. The system must maintain historical records of stock transactions, including purchases, sales, returns, damaged goods, and adjustments, ensuring transparency and traceability.

SRS-Software Requirements Specification

Stock Management System	
Retailers' warehouse struggle with inaccurate and outdated stock records, backorders, outstocking and order errors, real-time inventory management system is required for tracking, alerting, managing stocks.	Supplier management module
1. Introduction	4. Interface requirements
1.1 Purpose	• System integration
Defined requirements for stock management system to automate inventory management.	• Supplier database API for integration
1.2 Scope	8. Performance Requirements
Automate inventory tracking, supplier management and reporting.	Handle look product records
1.3 Overview	6. Design constraints
Tracks products, sales, suppliers and generate reports.	Work across multiple warehouses.
2. General description:	7. Non-functional requirements
Objective: Manage stock efficiency	Security: Role based access control
Users: Store staff, admins	Reliability: Real time synchronization
Features: Stock tracking, supplier management	8. Schedule and Budget
Benefits: Prevents stockout	Development: 6-8 months
3. Functional Requirements:	Cost: Medium
• Product database	9. Conclusion
• Stock in and stock out	• Strong user interface
• low stock alerts	• Intuitive navigation
	• Real time data processing
	• Scalable architecture
	• High availability
	• Cost-effective
	• User friendly

Class Diagram

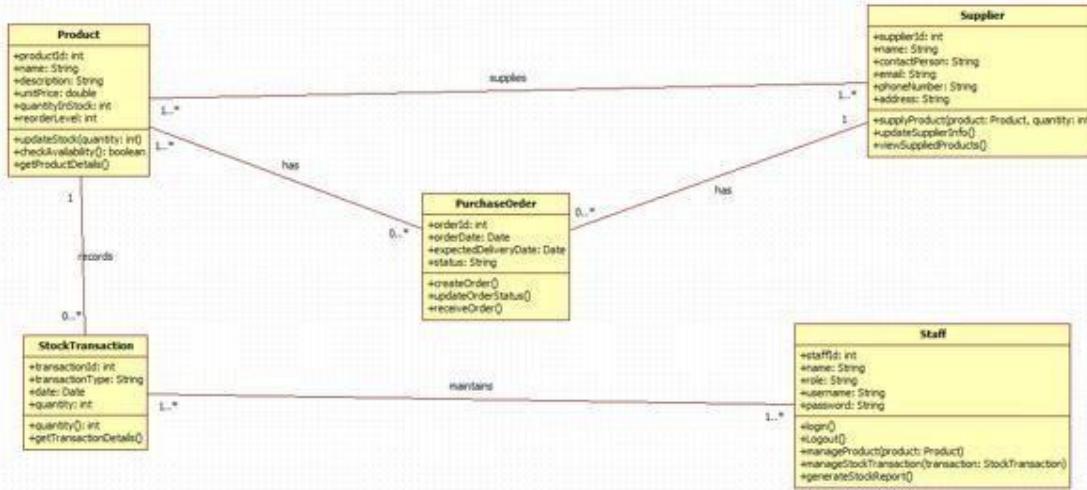


Figure 4.1: Class Diagram for Stock Maintenance System

This inventory management system tracks products supplied by various suppliers and maintained by staff members through purchase orders and stock transactions. Each supplier provides multiple products, and purchase orders are created to record order dates, expected delivery dates, quantities, and order status. When stock arrives or changes occur, stock transactions log the product, quantity, and transaction type to keep inventory accurate. Staff members manage the creation and updating of purchase orders, record stock transactions, and generate reports, while products maintain their own pricing, availability, and reorder information. Together, these components ensure smooth purchasing, stock maintenance, and supplier coordination.

State Diagram

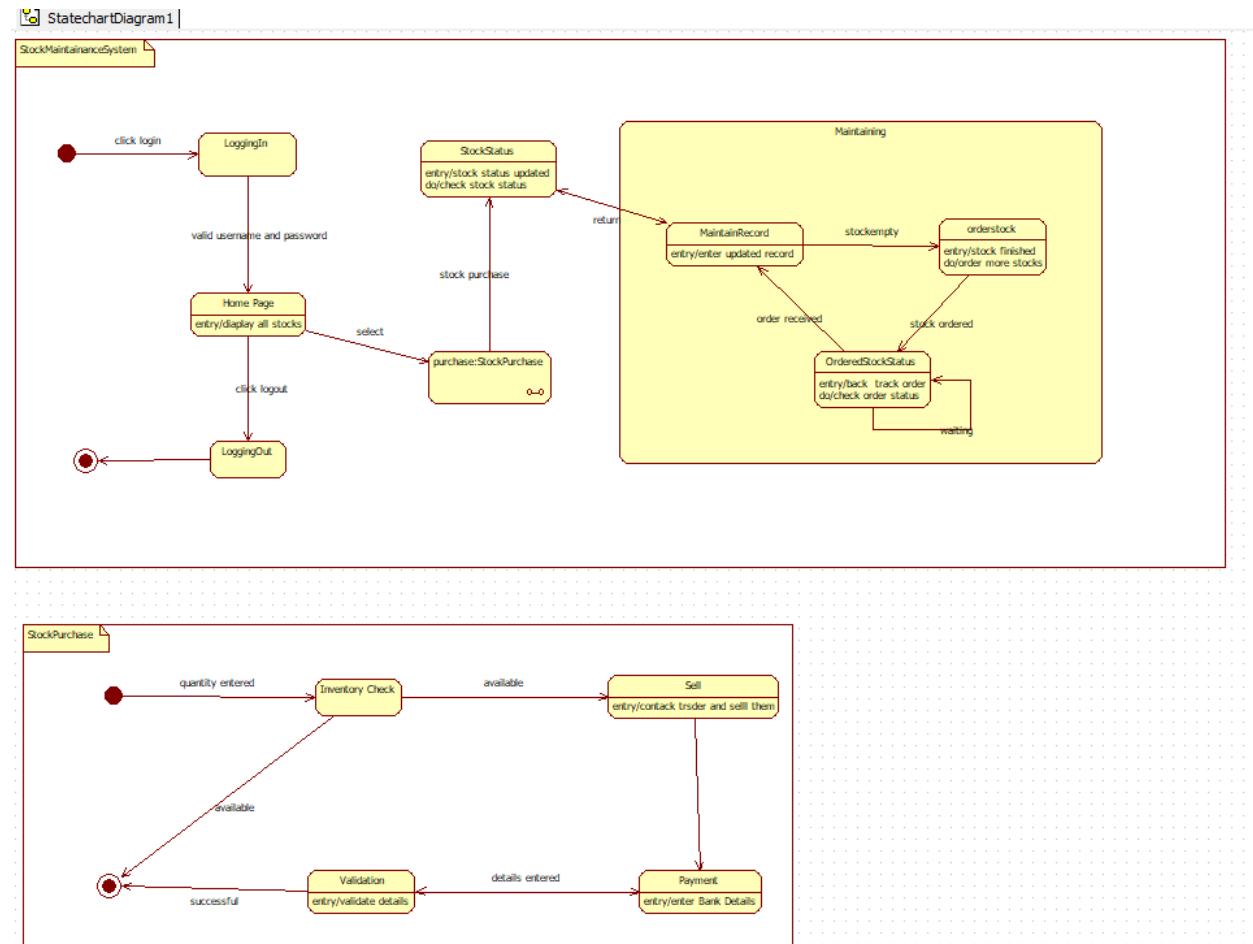


Figure 4.2: State Diagram for Stock Maintenance System

The stock maintenance process begins when a user logs into the system and is taken to the home page, where they can view all available stock items and choose to perform a stock purchase. When a stock purchase is initiated, the system checks current stock status and moves into the maintenance workflow, where records are updated, shortages are detected, and new stock orders are placed if inventory is low. The system then tracks the status of ordered stock until it arrives, updating records accordingly. Meanwhile, the stock purchase subprocess validates the requested quantity, determines whether sufficient stock is available, and either proceeds to sell the items and process payment or terminates the request if stock is insufficient. The overall process concludes when the user logs out or the purchase transaction successfully completes.

Use Case Diagram

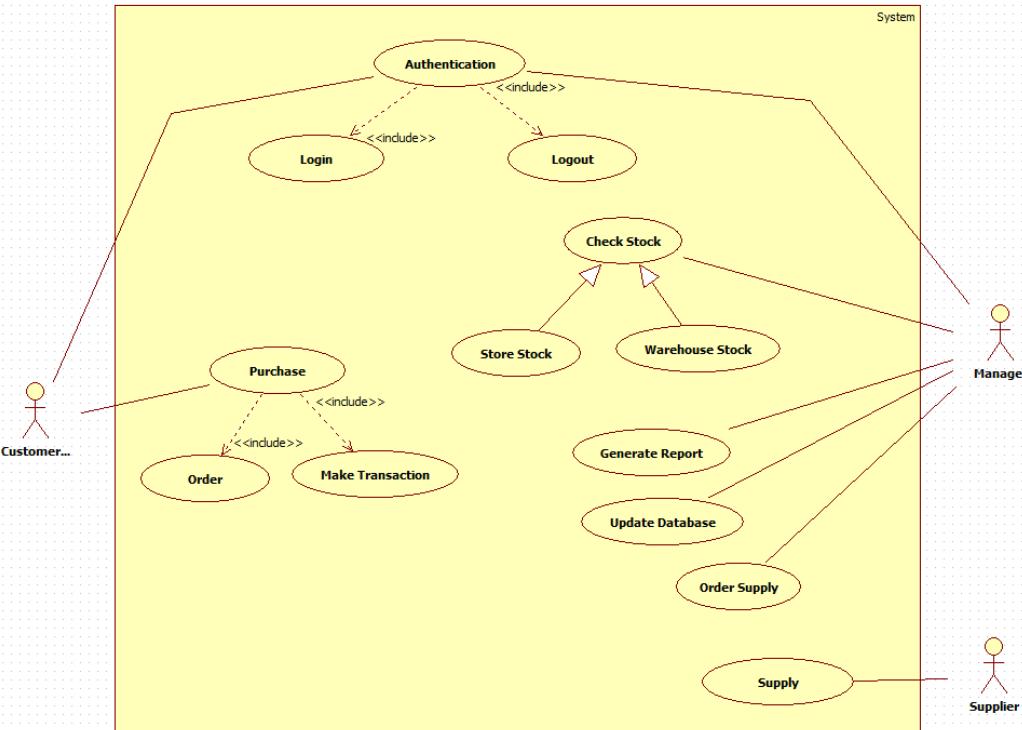


Figure 4.3: Use Case Diagram for Stock Maintenance System

In this stock management system, customers can purchase items by placing orders and making transactions, while the system ensures that user authentication through login and logout is always included. Managers interact with the system to check stock levels, whether in the store or the warehouse, and can generate reports or update the database based on stock conditions. When stock is low, the manager can initiate a supply order from suppliers, who then provide the required items to the system. Throughout the process, checking and managing stock is central, supporting smooth purchasing, inventory updates, and supply coordination.

Sequence Diagram

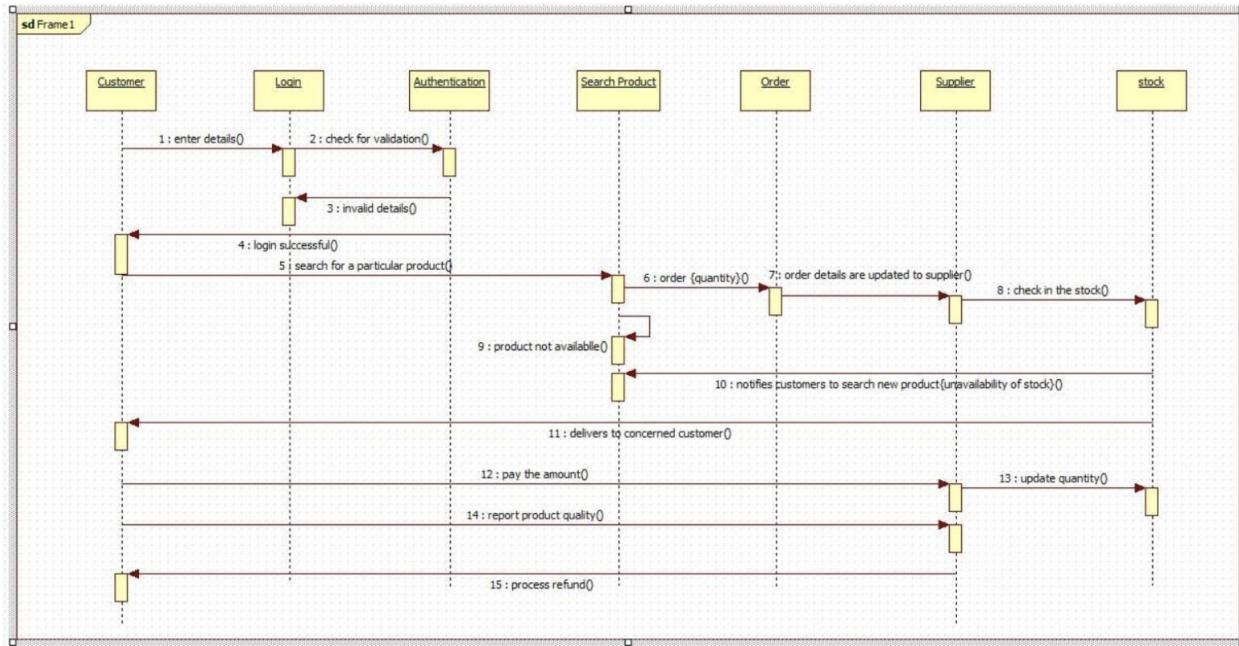


Figure 4.4: Sequence Diagram for Stock Maintenance System

In this sequence, a customer begins by entering login details, which the login system forwards to the authentication module for validation; if the details are invalid, the customer is notified, otherwise login succeeds. The customer then searches for a specific product, and the system initiates an order request with the desired quantity. The order information is sent to the supplier, who checks product availability in stock. If the product is unavailable, the supplier returns a notification prompting the customer to search for a new product. When the product is available, the supplier updates stock levels, processes the order, and delivers the product to the customer. The customer then pays for the product, reports product quality if needed, and the system may process a refund if a problem is reported, completing the transaction flow.

Activity Diagram

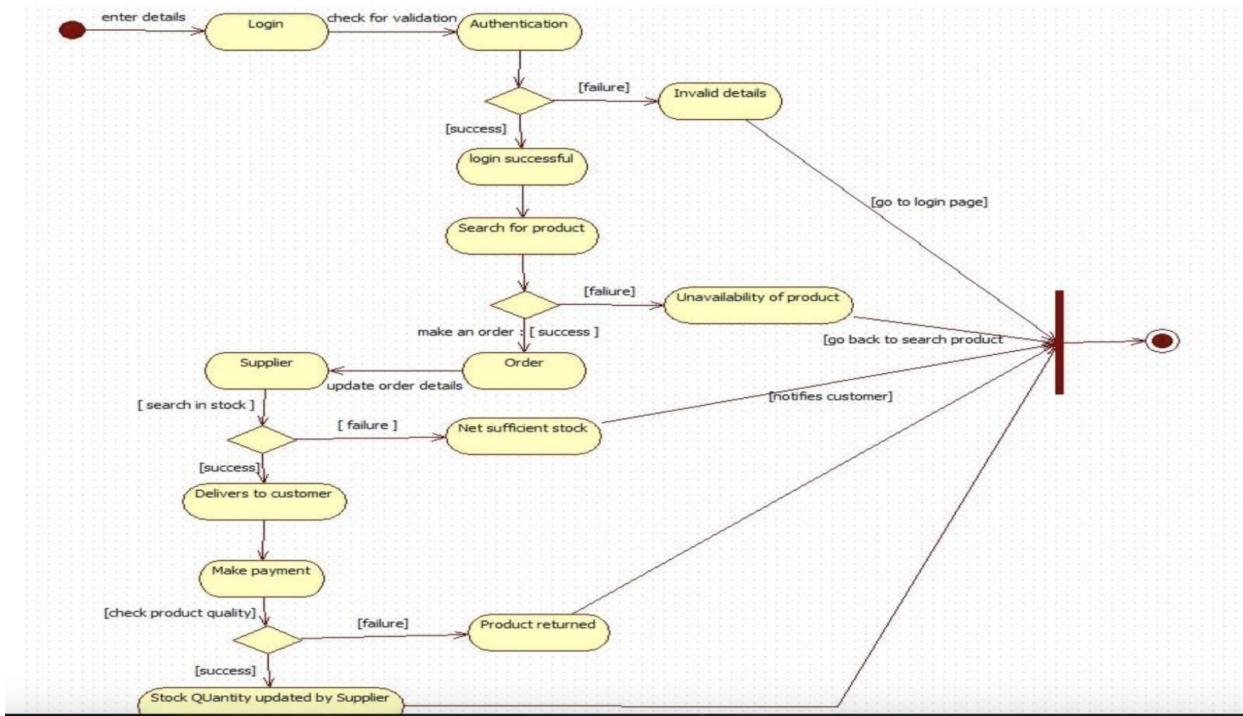


Figure 4.5: Activity Diagram for Stock Maintenance System

The activity begins when a customer enters their login details, which are validated by the authentication process; if the details are invalid, the flow ends by redirecting the user back to the login page. When login is successful, the customer searches for a product, and the system checks its availability; if the product is unavailable, the user is prompted to search again. If available, the customer places an order, and the supplier checks stock levels to confirm whether sufficient quantity exists. If stock is inadequate, the system notifies the customer and the process ends. When stock is sufficient, the supplier updates order details, delivers the product, and the customer makes payment. After purchase, product quality is checked; if the quality fails, the product is returned, otherwise the supplier updates stock quantity and the flow completes successfully.

5. Passport Automation System

Problem Statement

The process of passport application and issuance in many regions involves several manual steps such as filling out physical forms, submitting documents in person, waiting in long queues, and undergoing slow verification processes. These traditional methods often lead to delays, inconsistencies in data, missing documents, difficulty in tracking application status, and a lack of transparency for applicants. To address these challenges and improve the efficiency of passport-related services, a computerized Passport Automation System is required.

The Passport Automation System must automate the major functions involved in applying for, processing, and issuing passports. Applicants should be able to submit their applications online, upload required documents, schedule appointments, and track the status of their application at each stage. The system must validate applicant information, verify supporting documents, and provide alerts or notifications if corrections or additional documents are needed. It must support various stages of processing, including police verification, document scanning, fee payment, background checks, and final approval.

SRS-Software Requirements Specification

Passport automation system	
comprehensive passport automation system that can help easily dealing and tracking handling of passport related queries	
1. Introduction	1.1. Purpose: Defines requirements for passport automation system to enhance user experience and convenience of passport holders.
	1.2. Scope: Defines passport authentication, application submission, and document delivery.
	1.3. Overview: System overview. Enable online application, document submission, verification, and passport delivery.
2. General Description	Objective: Automate passport processing. Using: Citizens, govt. staff. Features: Online applications, status tracking, verification. Benefits: Fast, transparent and efficient.
3. Functional requirements	- Online application and registration - Document verification - Appointment scheduling - Police verification
4. Non-functional requirements:	
	• Performance requirements: Handle 10k+ applications daily
	• Design constraints: Govt. security compliance
	• Non-functional requirements: Security: End-to-end encryption b/w client and system. Availability: 99.9% uptime. Reliability: Nationwide usage.

Class Diagram

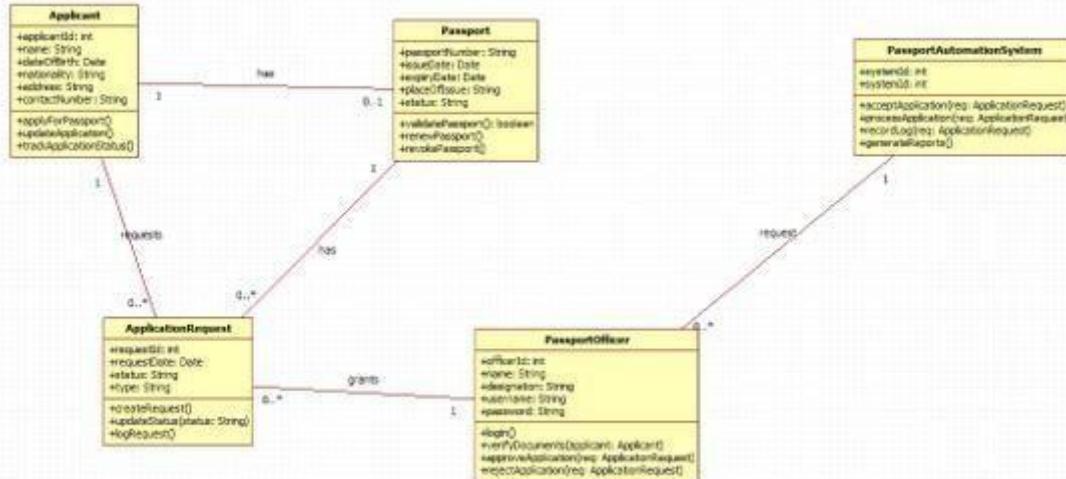


Figure 5.1: Class Diagram for Passport Automation System

This passport application system manages applicants who submit passport requests that are processed by passport officers through the passport automation system. Each applicant may already have an existing passport or apply for a new one by generating an application request containing details such as request type, submission date, and status. The passport officer reviews the application, verifies documents, approves or rejects the request, and updates the status accordingly. Approved requests result in the issuance of a passport linked to the applicant. Throughout the process, the passport automation system coordinates application submissions, retrieves applicant and passport data, forwards requests to officers, and returns confirmation or rejection results, ensuring a smooth and organized passport issuance workflow.

State Diagram

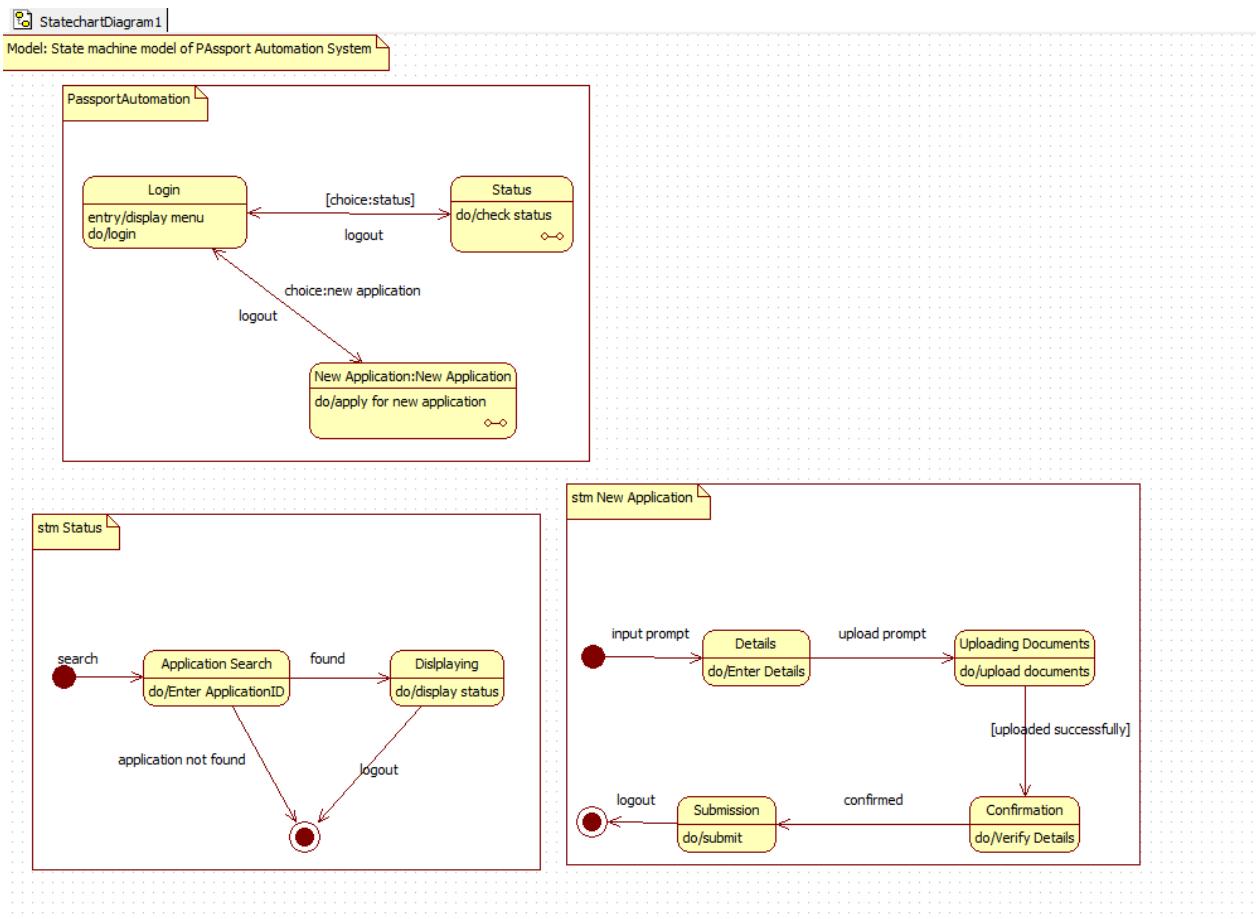


Figure 5.2: State Diagram for Passport Automation System

The passport automation system begins with the user logging in, after which they can either check the status of an existing application or start a new one. If the status option is chosen, the system enters the status state machine, where the user searches by application ID, and the system either displays the current status or ends the process if the application is not found. If the user selects to submit a new application, the system moves into the new-application workflow, prompting the user to enter personal details, upload required documents, and submit the form. The system then verifies the submitted information and confirms successful completion. At any point, the user can log out, returning the system to the initial login menu and resetting the workflow.

Use Case Diagram

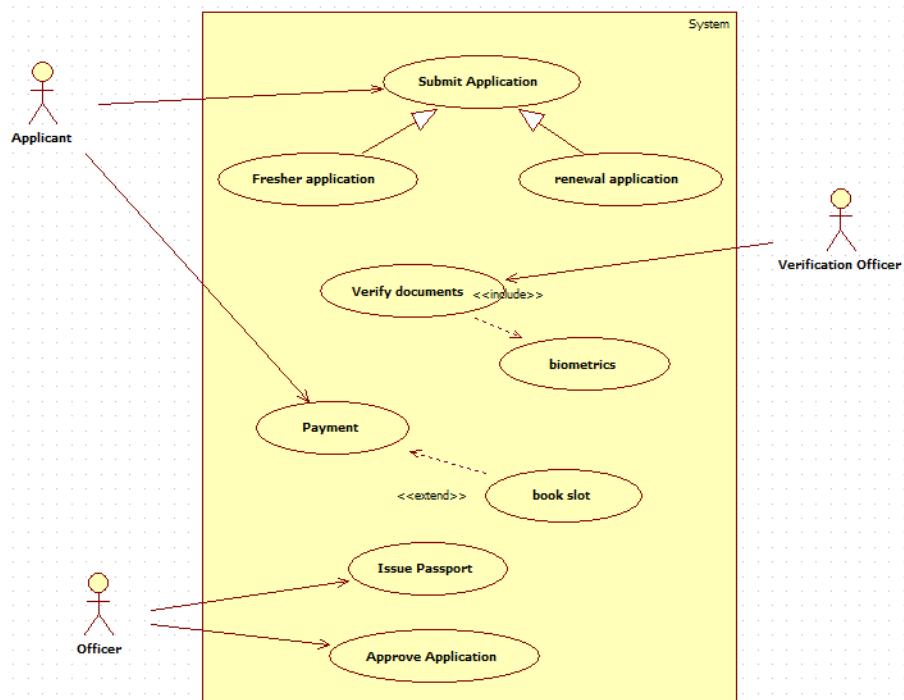


Figure 5.3: Use Case Diagram for Passport Automation System

This passport application system allows an applicant to submit either a fresh application or a renewal request, after which the system proceeds to verify the submitted documents with the involvement of a verification officer. As part of the verification process, the applicant also completes biometric procedures before selecting and booking an appointment slot. Once verification is completed, the applicant makes the necessary payment, and the system forwards the case to an officer who approves the application and issues the passport. Throughout the process, document verification is an essential included step, while slot booking extends the payment workflow to ensure proper scheduling and processing.

Sequence Diagram

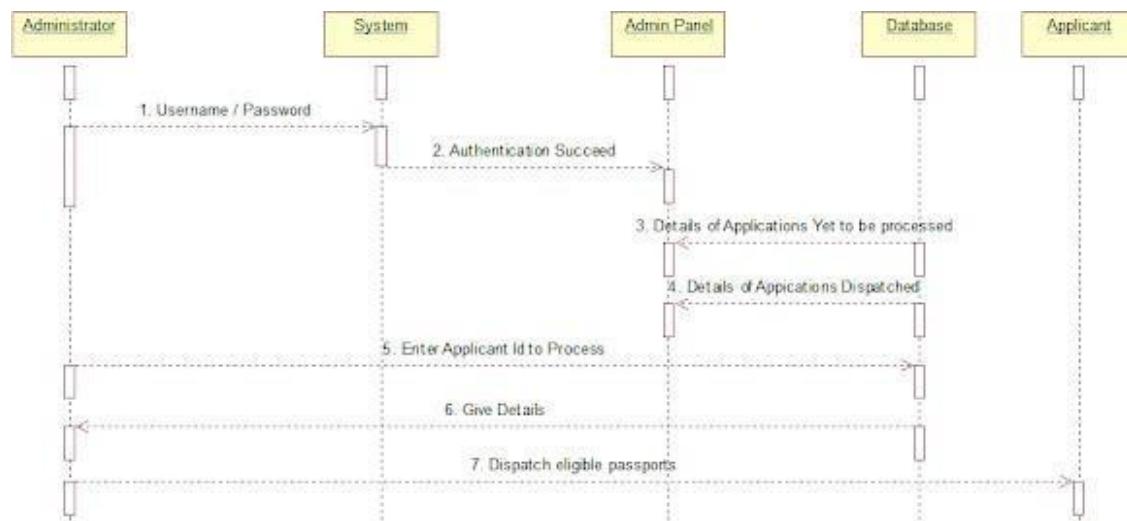


Figure 5.4: Sequence Diagram for Passport Automation System

In this passport dispatch workflow, an administrator begins by entering their username and password, which the system validates before confirming successful authentication. Once logged in, the system retrieves from the admin panel the list of passport applications that are still pending and those already dispatched, using the database as the data source. The administrator then selects an applicant ID to process, after which the admin panel sends the applicant's details back through the system. Finally, eligible passports are dispatched to the respective applicants, completing the sequence.

Activity Diagram

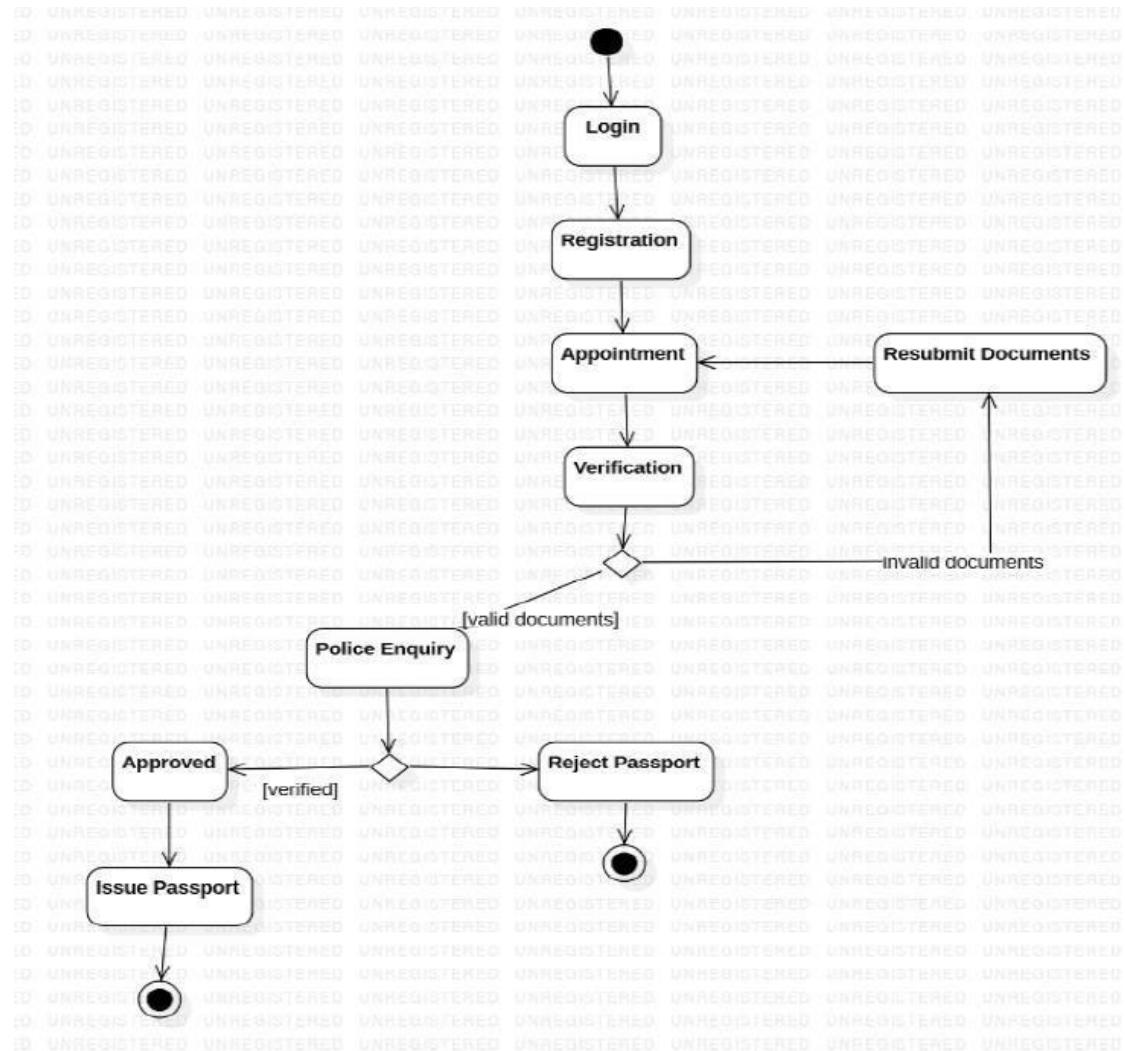


Figure 5.5: Activity Diagram for Passport Automation System

This passport approval process begins when the applicant logs into the system and completes the registration step, after which they schedule an appointment for document verification. During verification, the submitted documents are checked; if they are invalid, the applicant is asked to resubmit the documents and return to the appointment stage. If the documents are valid, the process proceeds to a police enquiry for background verification. Based on the police verification report, the application is either approved or rejected. Approved applications move to the final step where the passport is issued, while rejected applications terminate the process.