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

Lab Record

Object-Oriented Modeling

Submitted in partial fulfillment for the 5th Semester Laboratory

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:

HIMIKA KAKHANI

1BM23CS112

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 Analysis and Design(22CS6PCSEO) laboratory has been carried out by HIMIKA KAKHANI (1BM23CS112) during the 5th Semester September 2025-January 2026

.

Signature of the Faculty Incharge:

Name of Your Batch Incharge and designation:
Dr. Roopashree S,
Assistant Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

Table of Contents

1. HOTEL MANAGEMENT SYSTEM

Srs Document :

Page 1

LAB - 1

Aim: Develop a complete IEEE standard SRS document with several requirements for HOTEL MANAGEMENT SYSTEM

1. Introduction :-

(a) Purpose :-

This document outlines the functional and non functional requirements for the Hotel Management System. It will serve as a guide for developers, testers and stakeholders to understand the expected features and performance of the system. The goal is to automate hotel operations and improve customer service.

(b) Document conventions :-

All headings are bold and properly numbered for easy reference.

Requirements are labeled as Functional, Non-functional, External Interface.

Standard English is used throughout for clarity.

(c) Intended Audience and Reading Suggestions :-

The SRS document intended for developers, testers, project managers, hotel administrators. Developers should focus on requirements, testers on functionality and interfaces while manag

Page 4

(2) Specific Requirements :-

(a) Functional Requirements :-

- Users can book, modify and cancel reservations.
- Receptionist manage check-ins and check-outs.
- System generates and prints bill.
- Admin can add/edit rooms and view reports.
- Email confirmations are sent after booking.

(b) Non-functional Requirements :-

- System should respond within 2 seconds.
- 99.9% uptime is expected.
- Secure login and encrypted data structure.
- Must support 50 users at the same time.
- Should be user-friendly and fast.

(c) External Interface Requirements :-

- GUI for admin, staff and customers.
- Integration with payment gateways.
- Email servers for sending notification.
- Optional barcode scanner support at reception.

Appendices :-

- Glossary of terms used in HMS
- Screenshot mockups and interfaces.
- Use Case, ER and DFD diagrams.
- Sample test case for each feature.

Page 2

and clients may focus our scope and user roles.

(a) Project scope :-
The HMS will manage hotel operations such as reservations, room availability, customer check-in, check out, staff, billing.

(c) References :-
- IEEE 830 Software Requirements specification standard
- MySQL and Java Swing documentation
- API's for online payment and mail notification

2. Overall Description :-

(a) Product Perspective :-
The HMS is standalone application, but maybe extended to integrate with payment gateways and mail services. It is intended for both local use and online access.

(b) Product functions :-
Main functions include room reservation, check-in/checkout, billing, report generation and housekeeping status tracking. It also allows admin-level control for staff management.

Page 3

(c) User classes and characteristics :-
- Administrator :- full access to all modules and data
- Receptionist :- can manage manage bookings and billing
- Housekeeping staff :- can view and update room cleaning status
- Customers :- can book rooms online

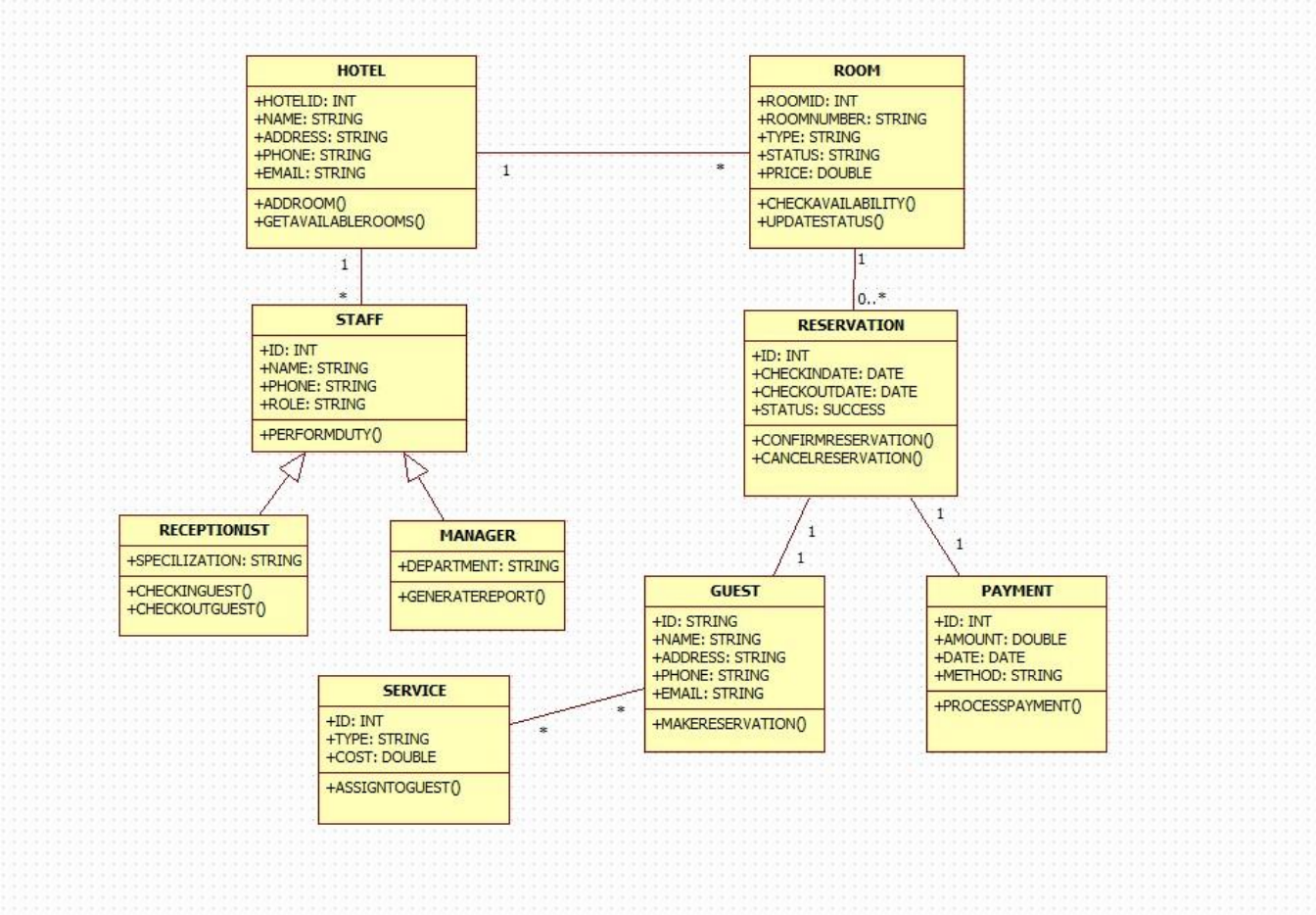
(d) operating Environment :-
The system will run on windows. It uses MySQL for backend and Java for frontend. Web components should be compatible with modern browsers.

(e) Design and Implementation Constraints :-
It must be scalable, secure and support integration with external services. The system should use only open source technologies.

(f) User Documentation :-
User guides and admin guides will be provided. These will be include system navigation, features, FAQ's and troubleshooting.

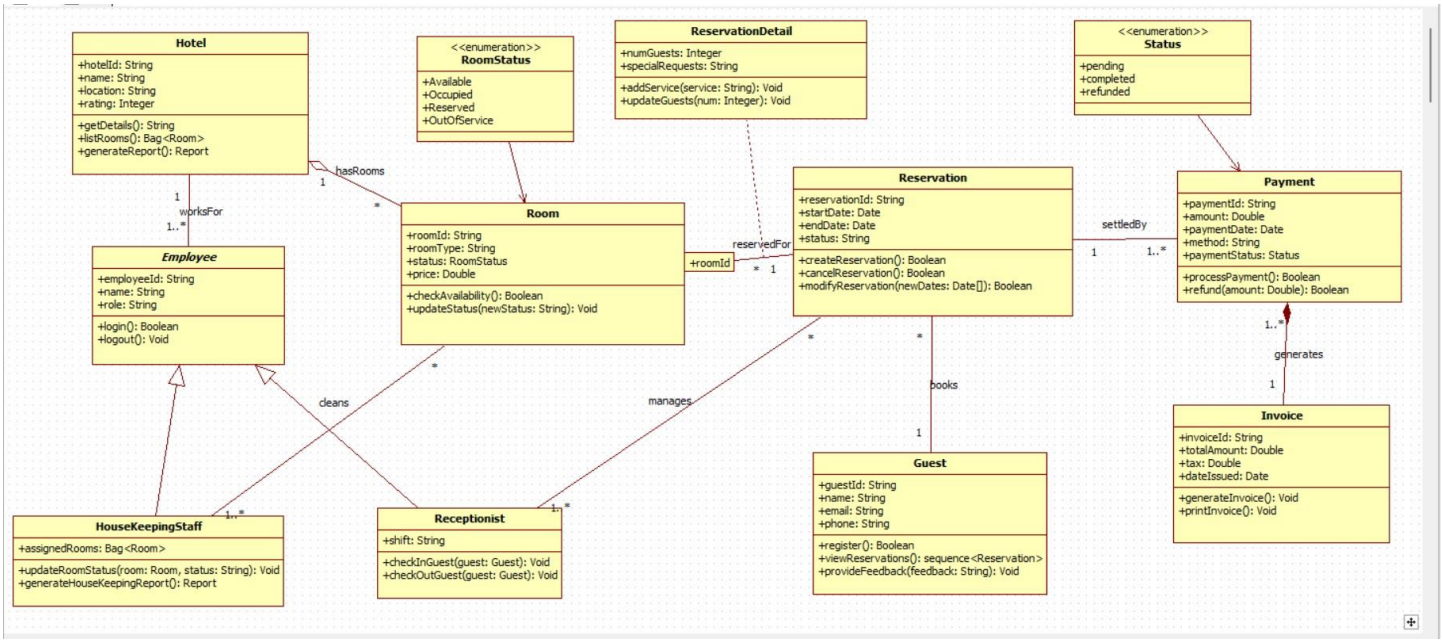
(g) Assumptions and Dependencies :-
Assumes users have basic computer knowledge. Online features depend on internet availability. The system also depends on correct setup of hosting environment.

Class Diagram(Simple)



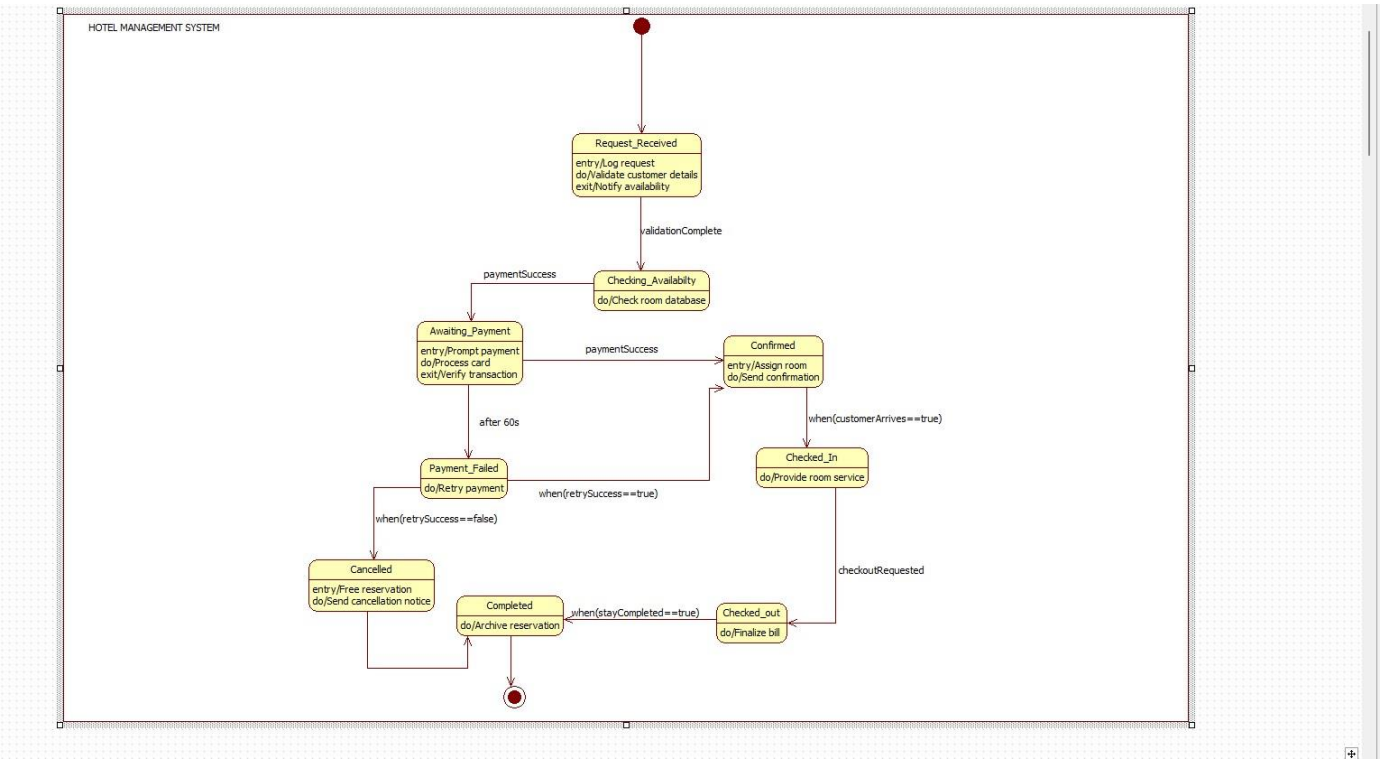
This class diagram represents a hotel management system where the Hotel contains multiple Rooms, which can be reserved by a Guest through a Reservation that includes detailed preferences in ReservationDetail. Receptionists manage reservations and interact with guests, while Housekeeping Staff clean and update room statuses, all of whom are specialized types of Employee. Each reservation may generate an Invoice, and payments for these invoices are handled through the Payment class. Overall, the diagram models the interactions and responsibilities required to manage hotel operations, guest services, room allocation, billing, and staff activities.

Class Diagram(Advanced)



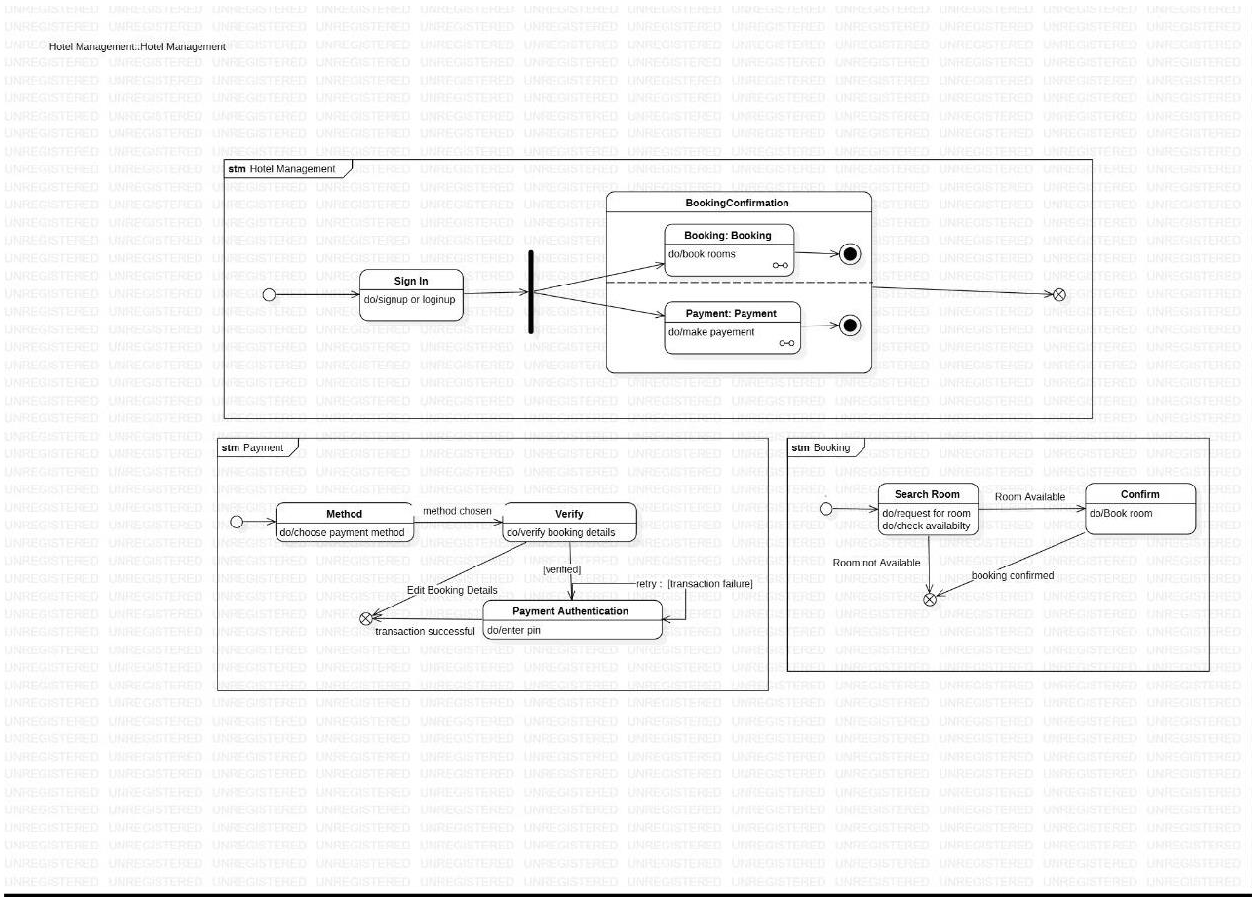
This class diagram illustrates a hotel reservation system where a Hotel manages multiple Rooms, each having a type defined by the RoomType enumeration. Guests can make reservations through the Reservation class, which stores check-in/check-out dates and reservation status. Payments for reservations are handled through an abstract Payment class with concrete types such as CreditCardPayment and CashPayment. The hotel operations are supported by Staff, including specialized roles like Receptionists for guest check-ins/check-outs and Managers for report generation. Additionally, Service objects represent extra services provided to guests, and Invoices are generated for completed reservations. Overall, the diagram models the key components and interactions required to manage rooms, guests, staff responsibilities, reservations, services, and payments in a hotel environment.

State Diagram(Simple)



The state machine diagram represents the life cycle of a hotel reservation request, beginning with Request_Received, where customer details are validated and availability is checked. Once validation is complete, the system transitions to Checking_Availability and, if successful, moves to Awaiting_Payment, where the customer is prompted for payment. Successful payment leads to a Confirmed reservation, while failed or timed-out payments transition to Payment_Failed and possibly Canceled. After confirmation, the reservation progresses to Checked_In once the customer arrives, and then to Checked_out upon checkout, where billing is finalized. The system ends in a Completed state once the stay is archived. This diagram clearly models the key operational states and transitions involved in handling a hotel reservation from request to completion or cancellation.

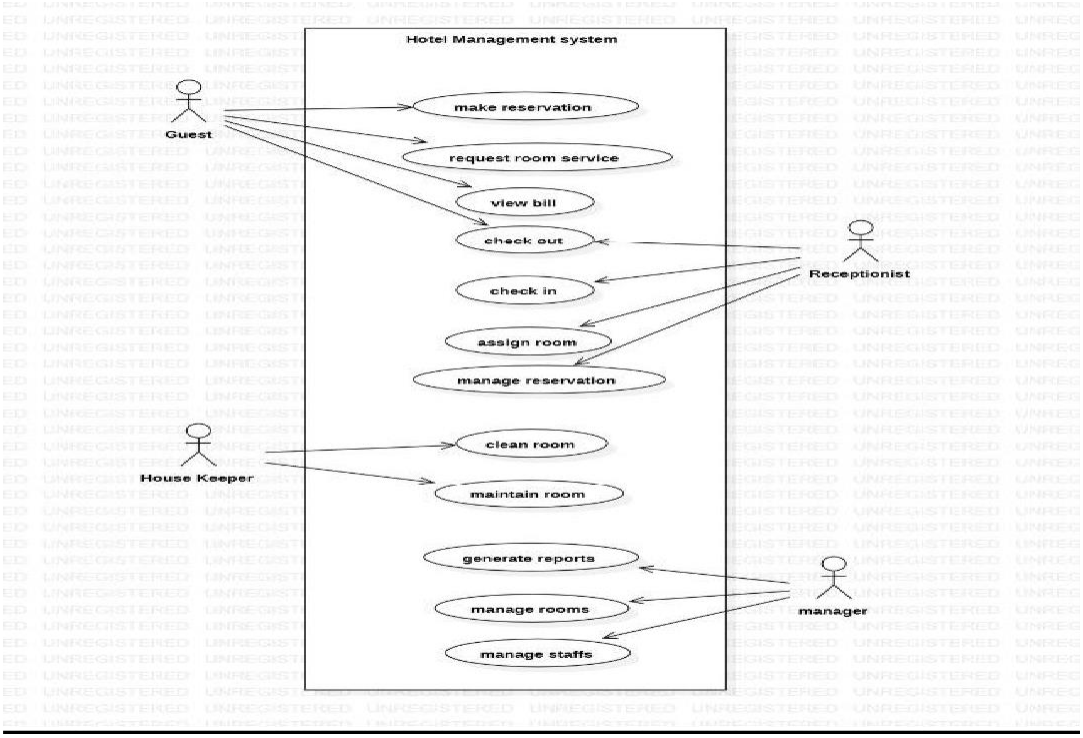
State Model(Advanced)



This diagram illustrates the overall behaviour of a hotel management system using multiple interacting state machines—one for hotel management, one for booking, and one for payment.
The process begins with the user signing in, after which they proceed to the BookingConfirmation region, where booking and payment activities occur in parallel. The Booking state machine handles room search, availability checking, and final confirmation of the booking. Simultaneously, the Payment state machine guides the user through selecting a payment method, verifying booking details, and authenticating the transaction. Each sub-system communicates synchronously to ensure that a booking is confirmed only after payment is

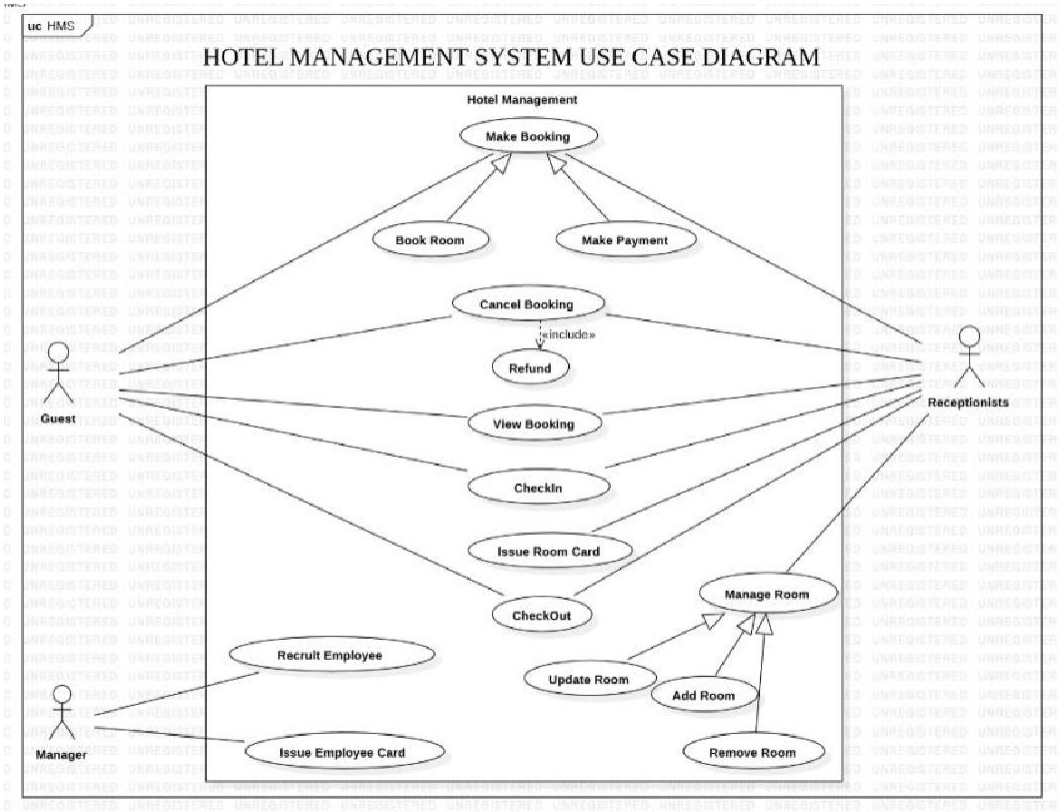
successfully authenticated. Overall, the diagram models how sign-in, room booking, and secure payment workflows integrate to complete the hotel reservation process.

Use Case Diagram(Simple)



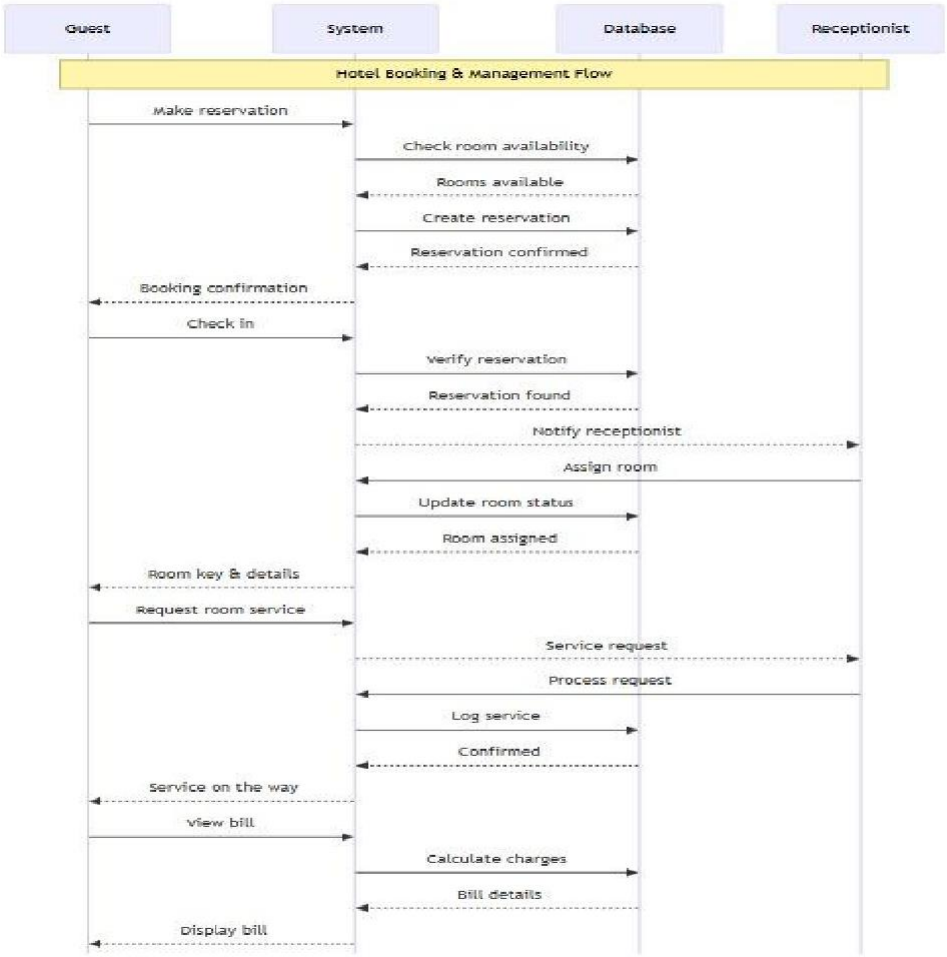
This use case diagram models the interactions within a hotel management system, showing how different actors—Guest, Receptionist, Housekeeper, and Manager—use the system to perform their respective tasks. Guests interact with the system to make reservations, request room services, view bills, check in, and check out. Receptionists support guest operations by handling check-ins, check-outs, assigning rooms, and managing reservations. Housekeepers interact with the system to clean and maintain rooms, ensuring room readiness. Managers oversee higher-level administrative functions such as generating reports, managing rooms, and supervising staff. Overall, the diagram highlights how each role collaborates with the system to streamline hotel operations and deliver smooth guest experiences.

Use Case Diagram(Advanced)



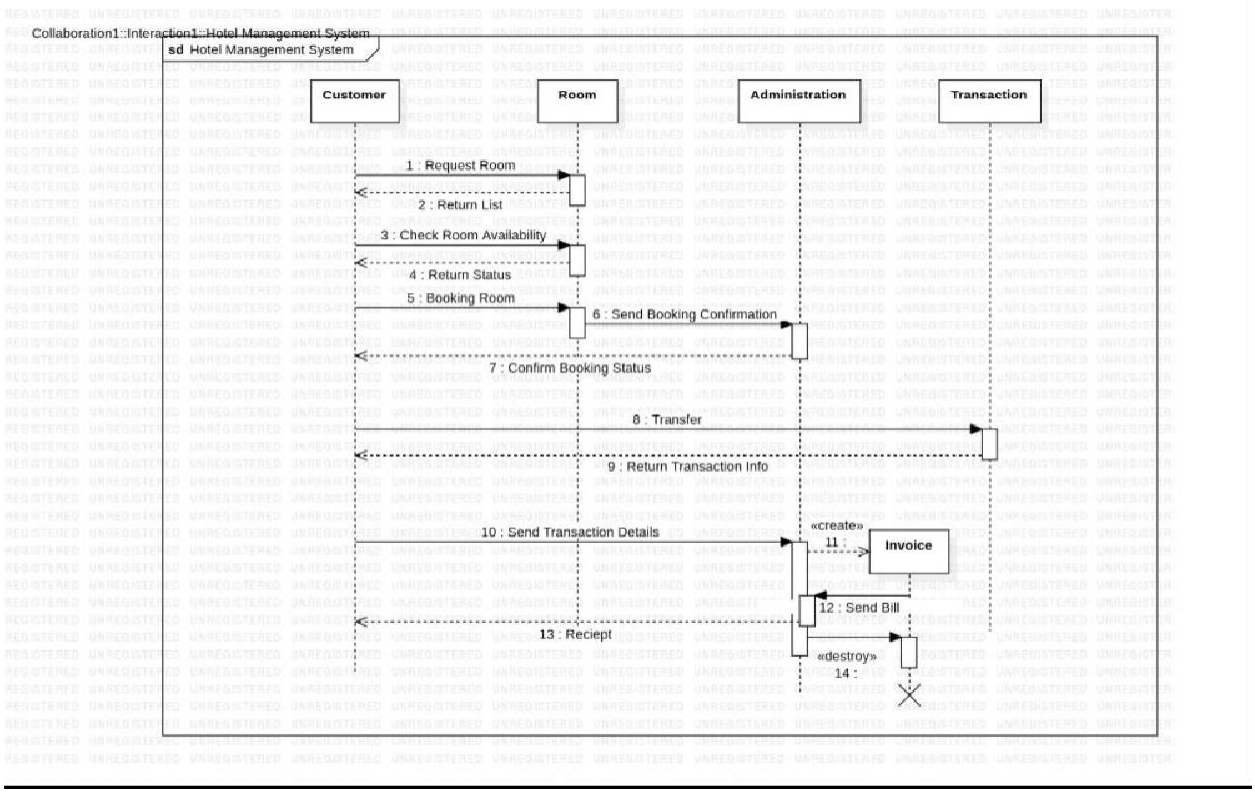
The use case diagram illustrates the interactions between different actors—Guest, Receptionist, and Manager—within the Hotel Management System. Guests can perform key operations such as making bookings, viewing bookings, booking rooms, making payments, checking in, checking out, issuing room cards, canceling bookings, and receiving refunds when necessary. Receptionists support and manage these guest-related activities by handling reservations, payments, check-ins, cancellations, and room card issuance, ensuring smooth hotel operations. The Manager oversees administrative tasks such as recruiting employees, issuing employee cards, and managing rooms through adding, updating, or removing rooms. Overall, the diagram captures the major functional requirements of the system and clearly shows how different users interact with various hotel processes.

Sequence Diagram(Simple)



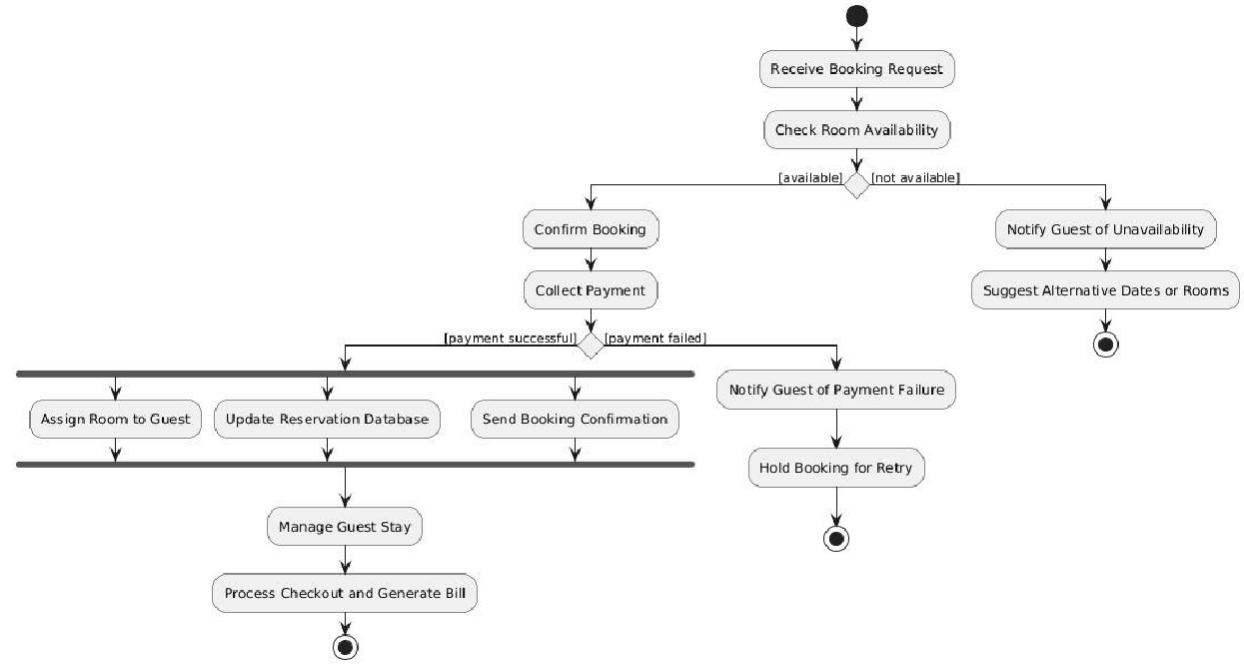
The sequence diagram illustrates the end-to-end flow of hotel booking and management, involving the Guest, System, Database, and Receptionist. The process begins when the guest initiates a reservation, prompting the system to check room availability with the database and create the reservation once a room is found. After confirmation, the guest checks in, and the system verifies the reservation and notifies the receptionist, who assigns a room and updates its status. Throughout the stay, the guest may request room services, which the system logs and forwards for processing, sending a confirmation back once the service is arranged. At checkout, the system calculates the charges based on the guest's stay and services used, retrieves billing details from the database, and displays the final bill to the guest. Overall, the diagram clearly models coordinated communication between system components to support booking, check-in, room assignment, service requests, and billing.

Sequence Diagram(Advanced)



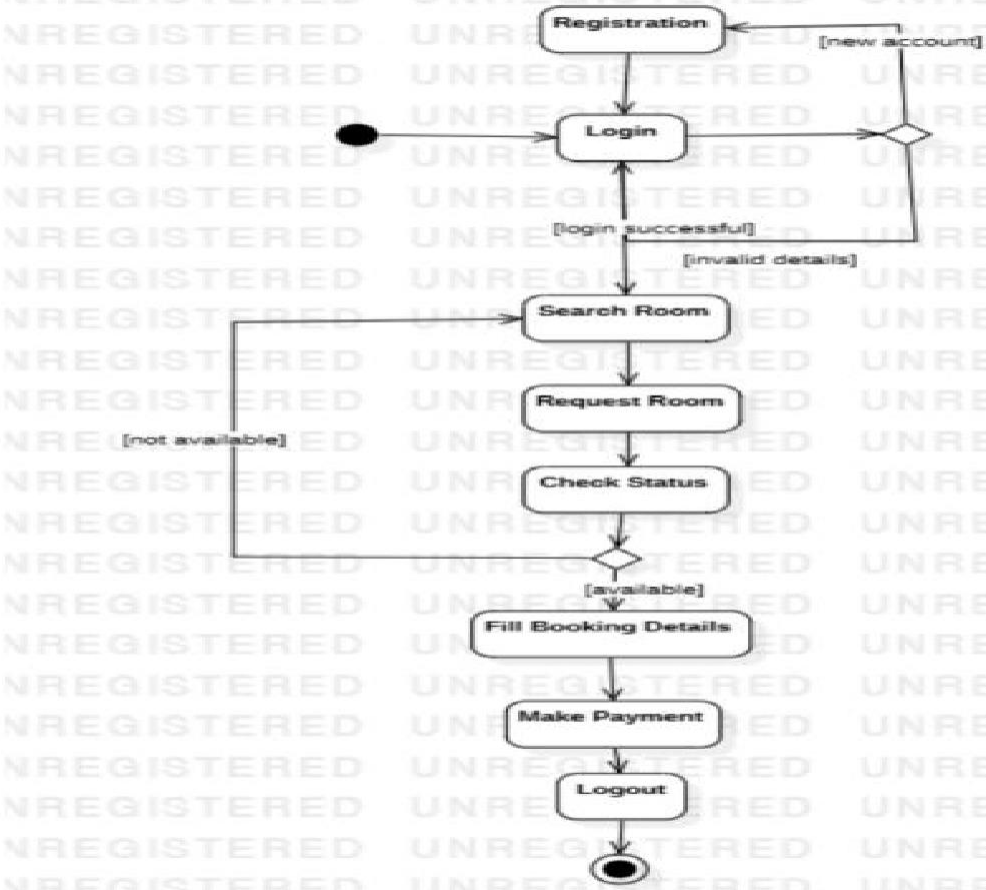
The collaboration diagram illustrates how different components of the hotel management system —Customer, Room, Administration, and Transaction—interact to complete a room booking and payment process. The sequence begins with the customer requesting a room, after which the Room component checks availability and returns the status. Once the customer proceeds with the booking, Administration confirms the reservation and initiates a payment transfer through the Transaction component. The Transaction module processes the payment, returns the transaction details, and triggers the creation of an invoice. Finally, Administration sends the bill to the customer, and the interaction concludes with the customer receiving the receipt. This diagram effectively demonstrates the flow of messages and coordination among system entities to handle booking, confirmation, payment, and invoicing in a hotel environment.

Activity Diagram(Simple)



This activity diagram illustrates the end-to-end workflow of the hotel booking and management process, starting from receiving a booking request to final checkout. The process begins when the system receives a booking request and checks room availability; if no rooms are available, the guest is notified and offered alternative options. When rooms are available, the system proceeds to confirm the booking and collect payment. Successful payments lead to room assignment, updating the reservation database, and sending a booking confirmation, after which the guest stay is managed until checkout, where the system generates the final bill. If payment fails, the system informs the guest and temporarily holds the booking for retry. Overall, the diagram captures decision points, parallel activities, and the complete life cycle of a guest reservation in a hotel management system.

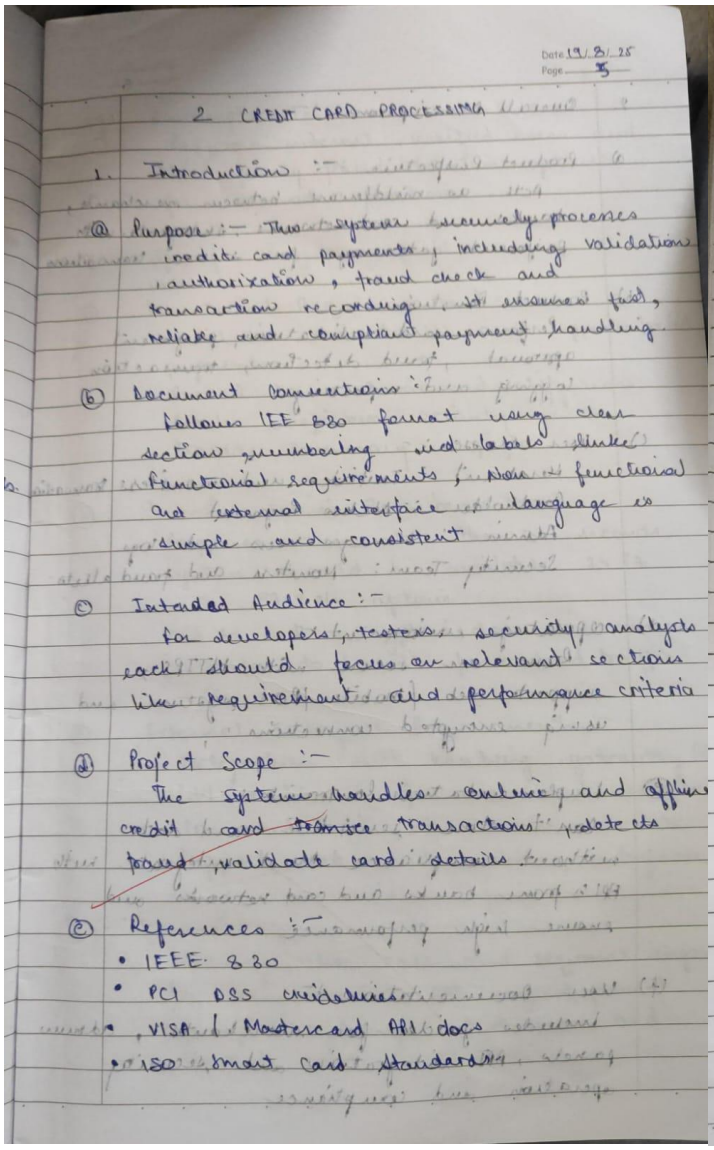
Activity Diagram(Advanced)



This activity diagram represents the complete flow of a hotel booking system, beginning with Login and handling Registration for new users. After a successful login, the customer can Search Room and Request Room, followed by a Check Status step where the system verifies availability. If rooms are not available, the process loops back to allow the customer to search again; if available, the user proceeds to Fill Booking Details and then Make Payment to finalize the reservation. Invalid login attempts redirect the user back to re-enter details, ensuring secure access. The process ends with the user Logging Out, capturing the end-to-end sequence from account access to room booking and payment.

2.CREDIT CARD PROCESSING

SRS Document :



2 CREDIT CARD PROCESSING

1. Introduction :-

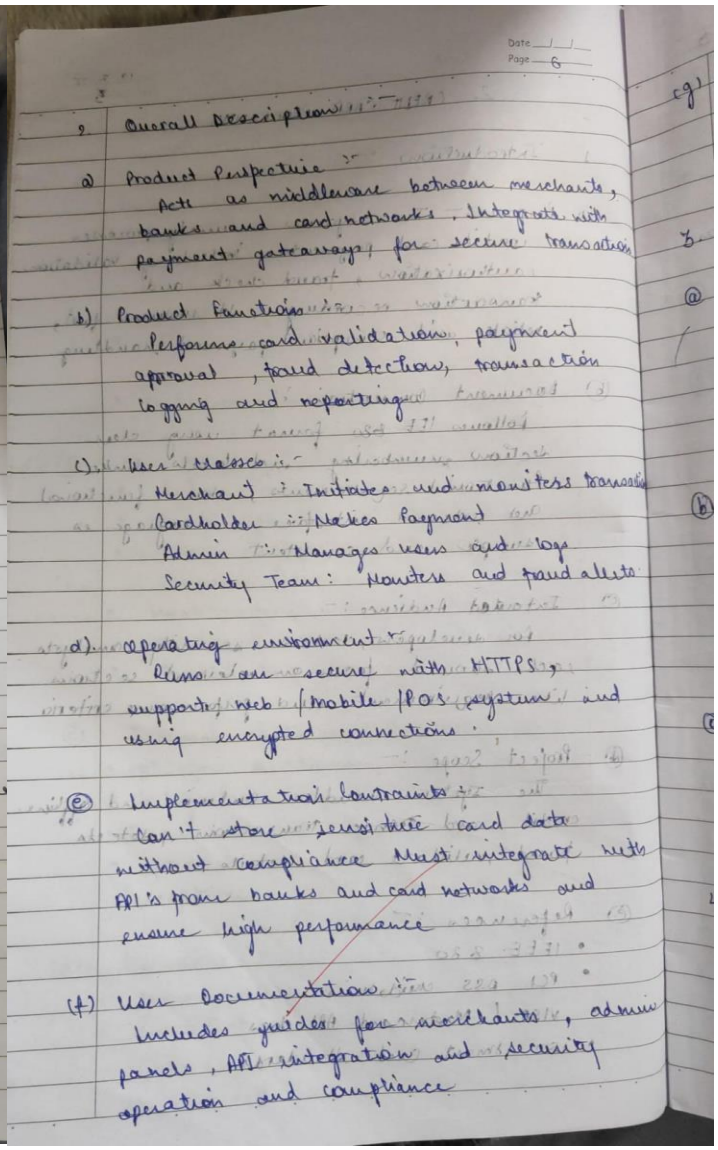
(a) Purpose :- The system securely processes credit card payments, including validation, authorization, fraud check and transaction recording. It ensures fast, reliable and compliant payment handling.

(b) Document Conventions :- Follows IEEE 830 format using clear section numbering and labels. Functional requirements, Non-functional and external interface language is simple and consistent.

(c) Intended Audience :- For developers, testers, security analysts each should focus on relevant sections like requirement and performance criteria.

(d) Project Scope :- The system handles online and offline credit card transactions, detects fraud, validate card details.

(e) References :-
- IEEE 830
- PCI DSS guidelines
- VISA / Mastercard API docs
- ISO smart card standards



2. Overall Description :-

(a) Product Perspective :- Acts as middleware between merchants, banks and card networks. Integrates with payment gateways for secure transaction.

(b) Product Functions :- Performs card validation, payment approval, fraud detection, transaction logging and reporting.

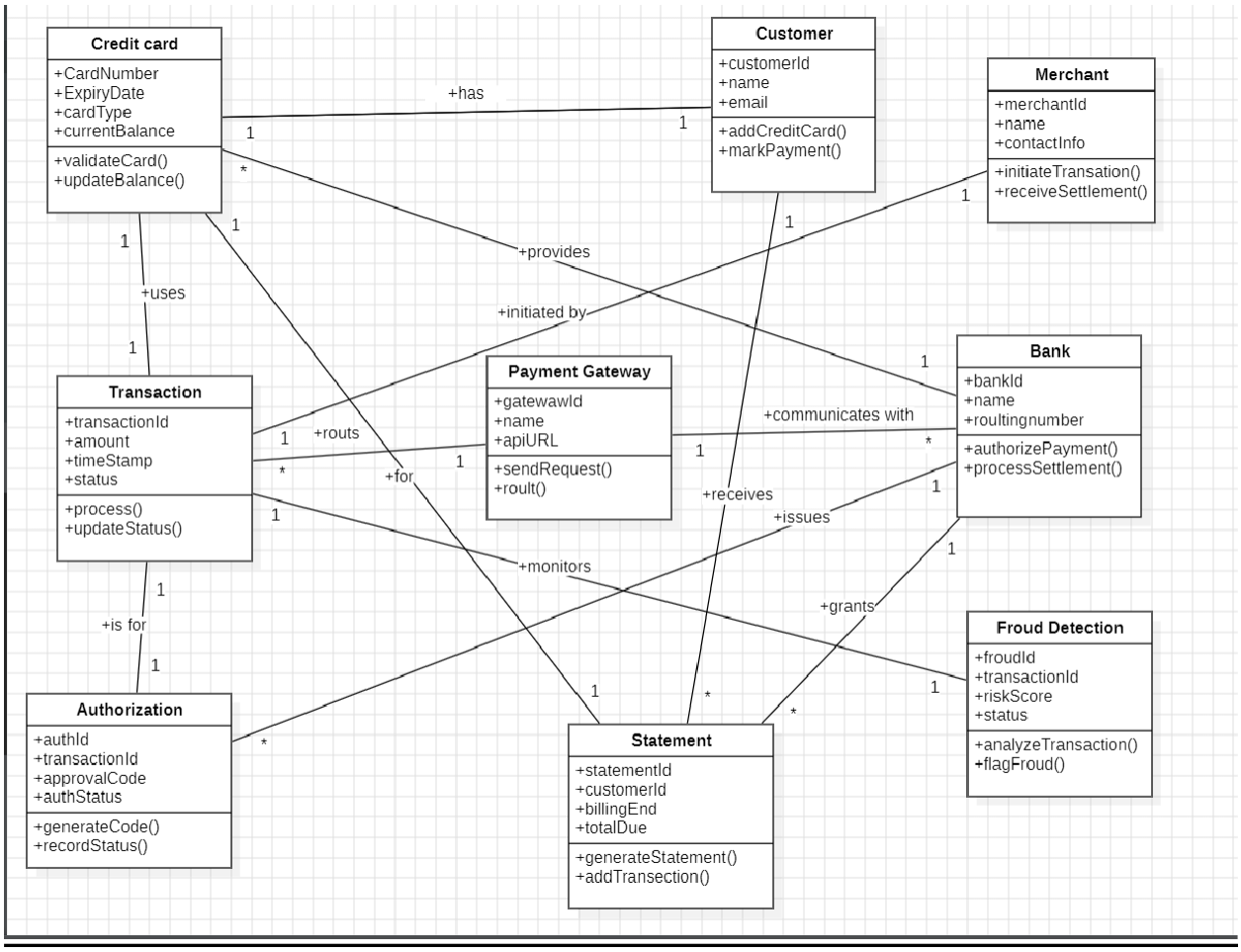
(c) User classes :-
Merchant : Initiates and monitors transaction
Cardholder : Makes payment
Admin : Manages users and logs
Security Team : Monitors and fraud alerts

(d) Operating environment :- Runs on secure with HTTPS, supports web / mobile / POS system and using encrypted connections.

(e) Implementation constraints :- Can't store sensitive card data without compliance. Must integrate with API's from banks and card networks and ensure high performance.

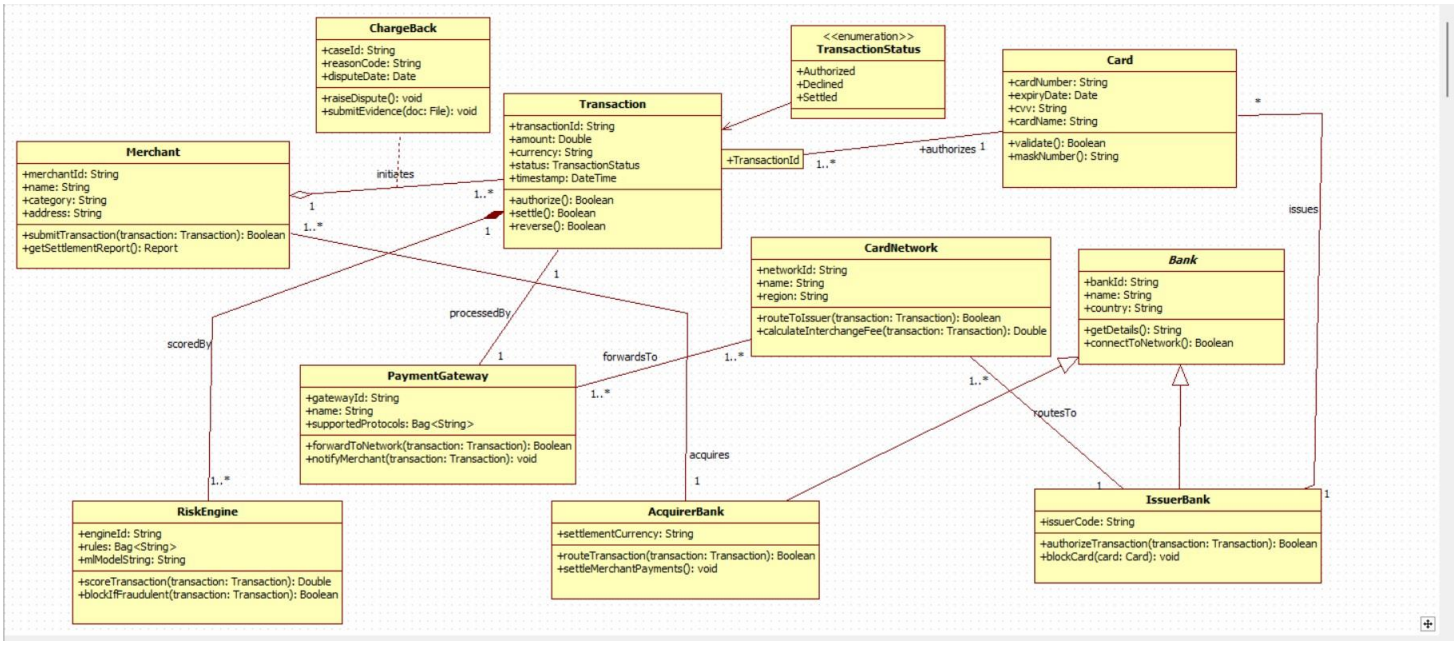
(f) User Documentation :- Includes guides for merchants, admin panels, API integration and security operation and compliance.

Class Diagram



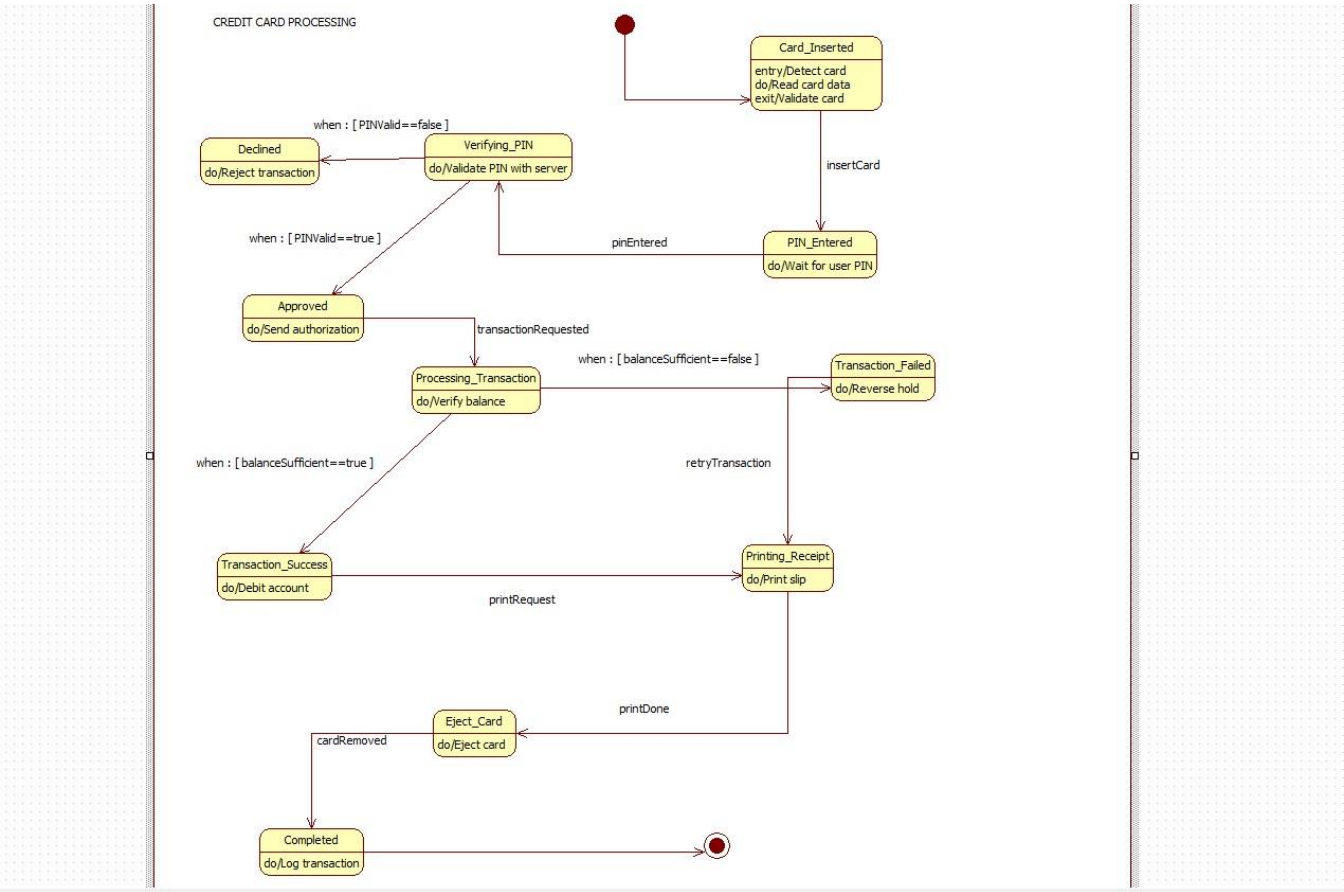
This class diagram models a credit-card-based payment processing system involving multiple entities such as Customers, Merchants, Banks, and a Payment Gateway. A Customer makes payments using a Credit Card, which is used to initiate a Transaction routed through the Payment Gateway. The gateway communicates with the Bank to authorize payments and issue responses, while an Authorization record stores approval details for each transaction. The Merchant initiates transactions and receives settlements once the bank approves them. All completed transactions are recorded in a Statement associated with the customer, and a Fraud Detection system monitors transactions to identify potential risks. Overall, the diagram illustrates how data and operations flow across various components to validate, process, authorize, and record credit card payments securely.

Class Diagram(Advanced)



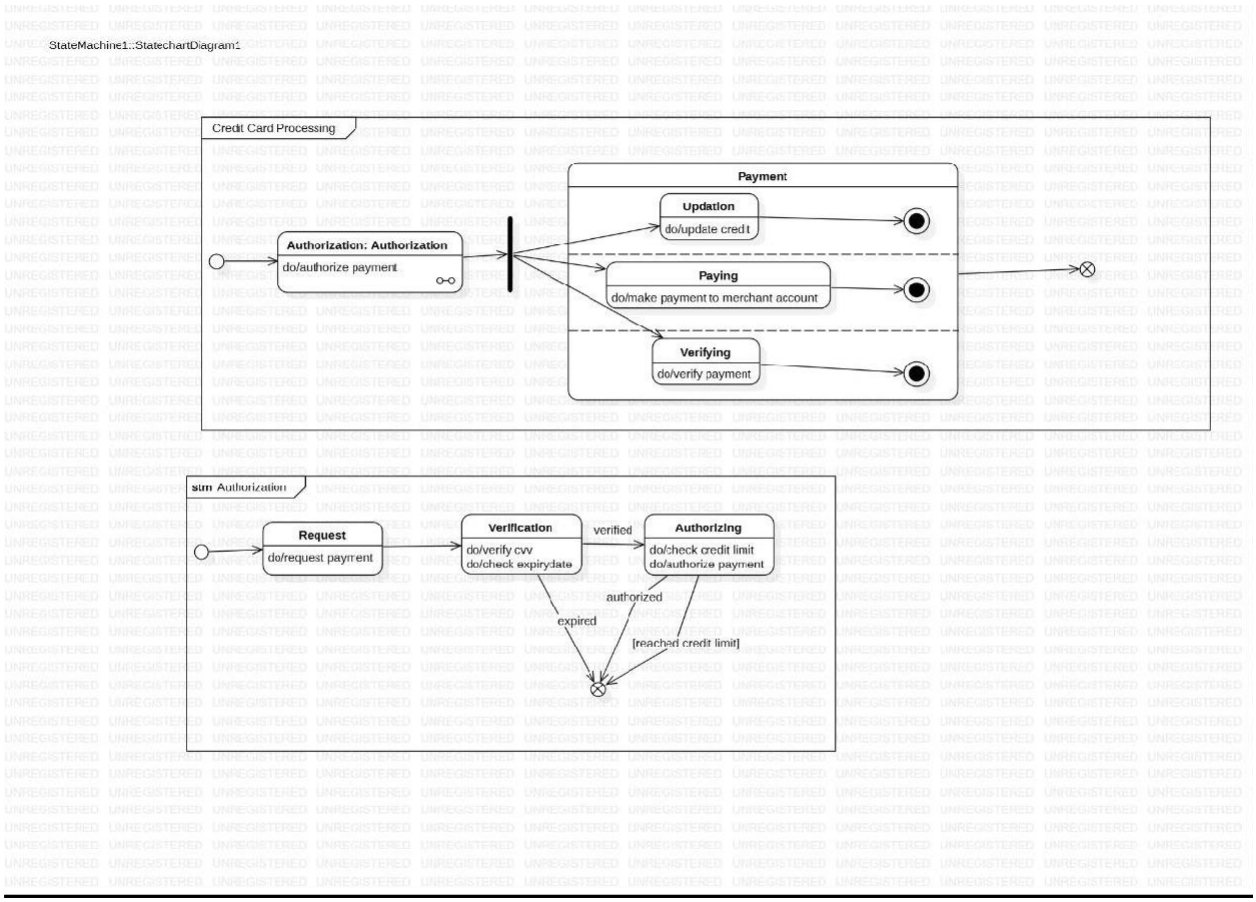
This class diagram models a complete digital payment ecosystem involving customers, merchants, banks, and transaction-processing entities. A Customer uses a Card to initiate a Transaction, which is routed through a PaymentGateway and processed using a CardNetwork that communicates with the IssuerBank for authorization and the AcquirerBank for settlement. The RiskEngine evaluates transactions for fraud, while ChargeBack handles disputes raised by customers. The Bank class supports issuing, processing, and validating payments, and the Merchant records submitted transactions and receives settlements. Overall, the diagram captures the full lifecycle of electronic payment processing, including authorization, risk evaluation, settlement, and dispute handling.

State Diagram(Simple)



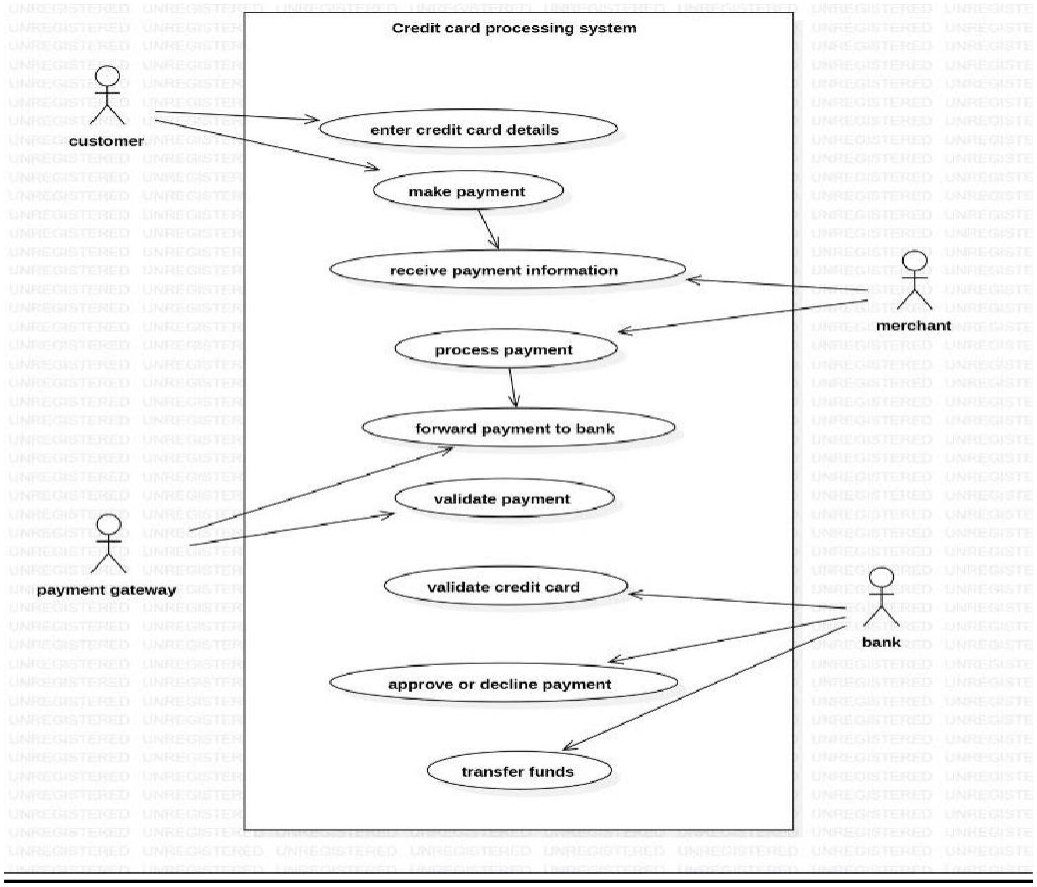
This state machine diagram represents the complete flow of a credit card transaction, beginning with the Card Inserted state where the terminal detects and validates card data. The process continues to the PIN Entered state, where the user provides their PIN, followed by verification in the Verifying PIN state. If the PIN is invalid, the system transitions to Declined and rejects the transaction; if valid, it moves to Approved, where authorization is granted. The system then enters the Processing Transaction state to verify account balance. Insufficient funds lead to Transaction Failed, triggering a reverse hold, whereas sufficient balance results in Transaction
Success, where the account is debited. After the transaction, the system prints a receipt in the Printing Receipt state and then proceeds to Eject Card, concluding the process in the Completed state, where the transaction is logged. This diagram clearly captures the decision points, validations, and outcomes involved in secure credit card transaction processing.

State Diagram(Advanced)



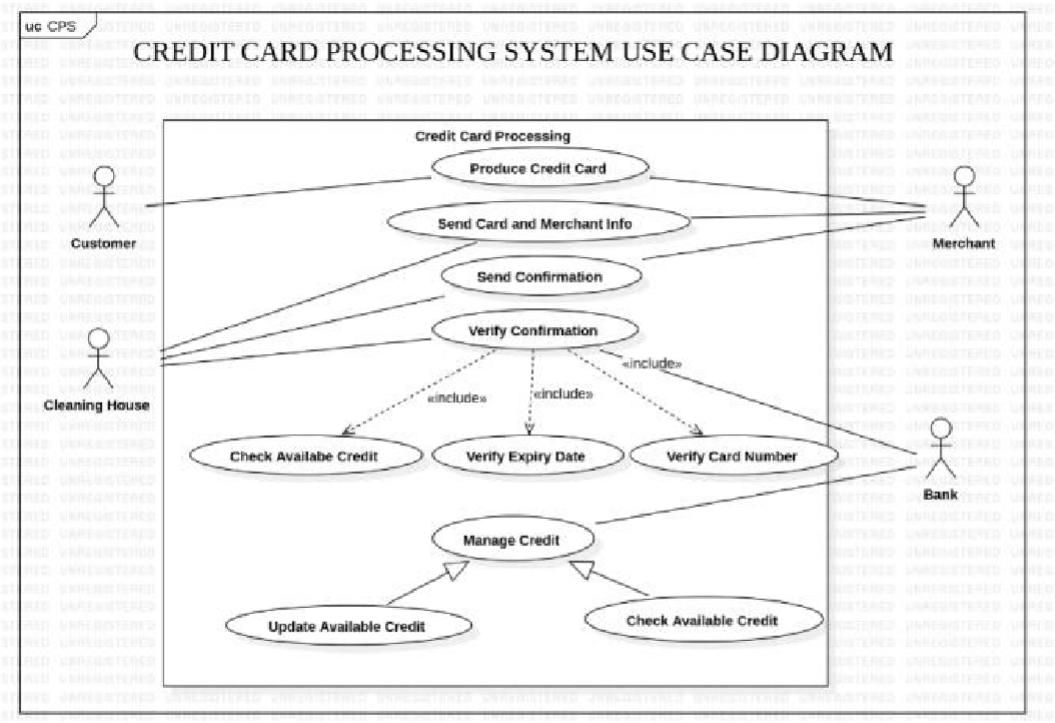
A credit card processing state diagram shows how a transaction moves from Transaction Created through an advanced sequence of states such as Validation, Authentication (OTP or 3-D Secure), Authorization by the issuing bank, and then either Capture and Settlement or failure states like Rejected, Declined, or Cancelled. After settlement, the transaction may enter Refunded (full or partial) or Disputed/Chargeback states depending on customer actions. The process ends in a final Closed state, making it a complete lifecycle representation of a real credit-card payment.

Use Case(Simple)



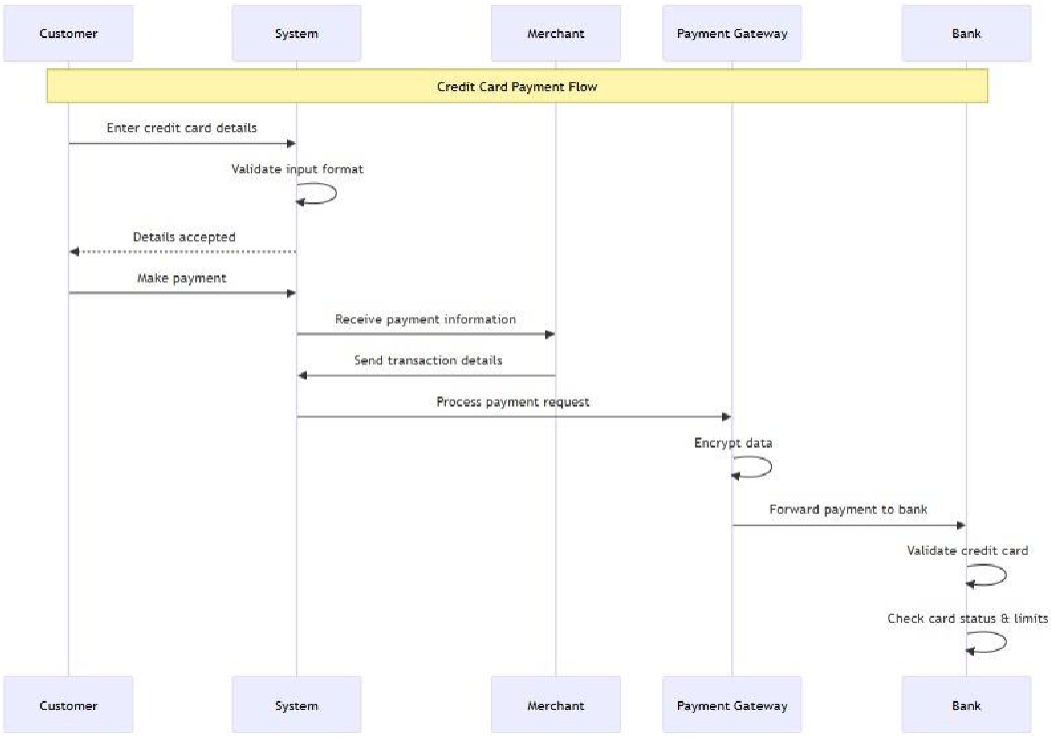
This state machine diagram illustrates the workflow of a credit card payment process by dividing it into two coordinated state machines: Authorization and Payment. The Authorization state machine begins with a payment request and moves through verification steps such as checking the CVV and expiry date. If verified, the system checks the credit limit and authorizes the transaction; if details are invalid or the card is expired or exceeds the limit, the process terminates. Once authorization is approved, control shifts to the Payment composite state, where three parallel substates—Verification, Paying, and Updation—handle payment validation, transferring funds to the merchant, and updating the credit balance. Together, these state machines show how card details are validated, payments are processed, and account balances are updated to complete a secure credit card transaction.

Use Case(Advanced)



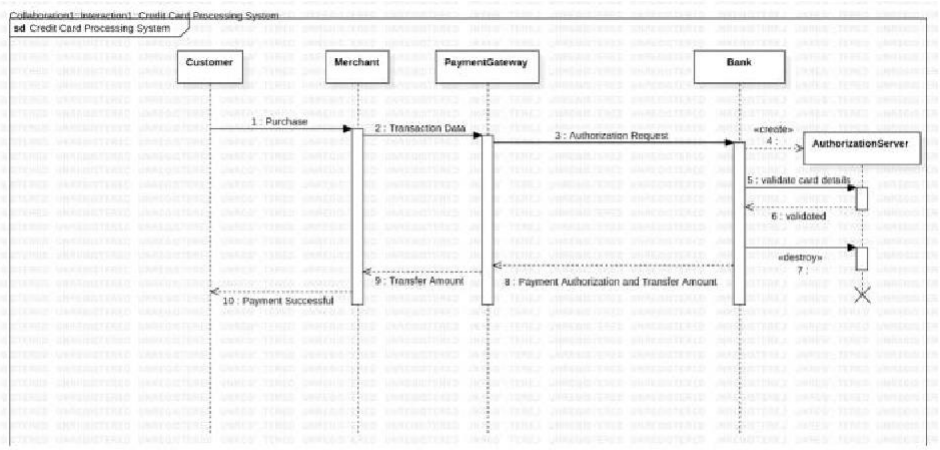
The use case diagram represents the interactions between key actors—Customer, Merchant, Bank, and the Clearing House—in a credit card processing system. The Customer initiates the credit card processing, which includes producing the card, sending card and merchant information, and receiving confirmation. The Merchant interacts with the system to receive card information and verify confirmation, while the Bank is responsible for validating card details through included use cases such as verifying the card number, checking available credit, and validating the expiry date. The Clearing House manages credit operations, including updating available credit and checking credit balances. Overall, the diagram clearly illustrates how multiple external entities collaborate with the system to ensure secure, validated, and accurate credit card transactions.

Sequence Model(Simple)



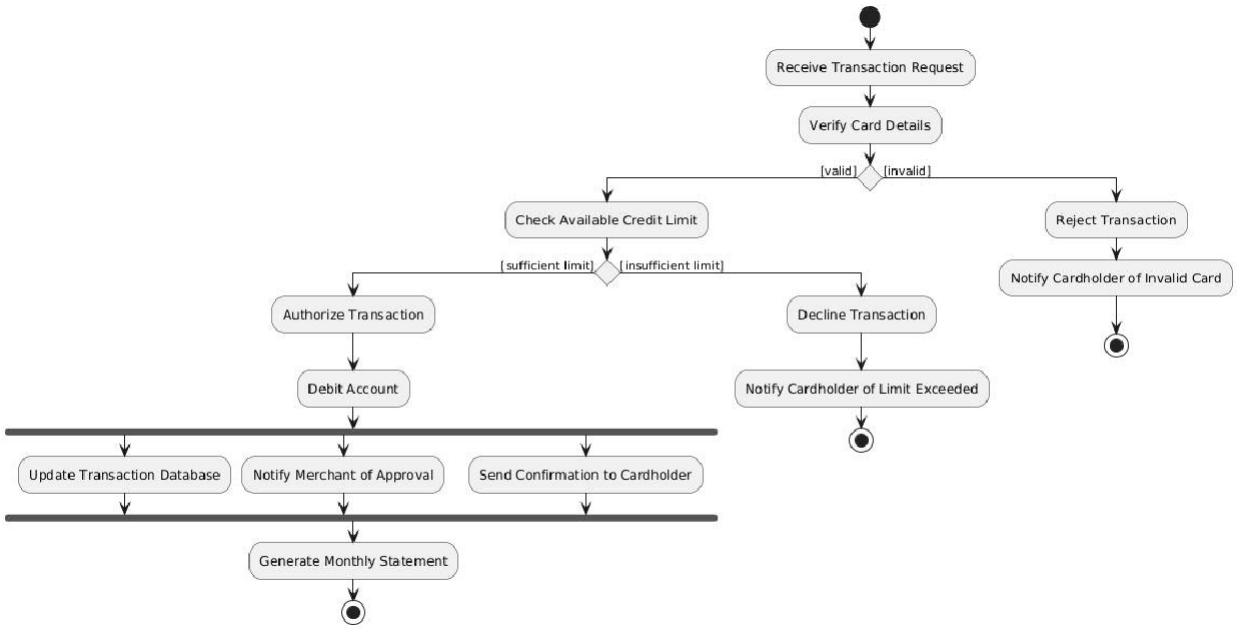
The sequence diagram illustrates the complete flow of a credit card payment transaction involving five key entities: Customer, System, Merchant, Payment Gateway, and Bank. The process begins when the customer enters their credit card details, which the system validates before accepting the payment request. The merchant receives the payment information and forwards the transaction details back to the system for processing. The system then sends the payment request to the payment gateway, where the data is securely encrypted and forwarded to the bank. The bank validates the credit card, checks the card's status and spending limits, and returns the authorization decision. This coordinated interaction ensures secure, accurate, and efficient processing of the customer's payment.

Sequence Model(Advanced)



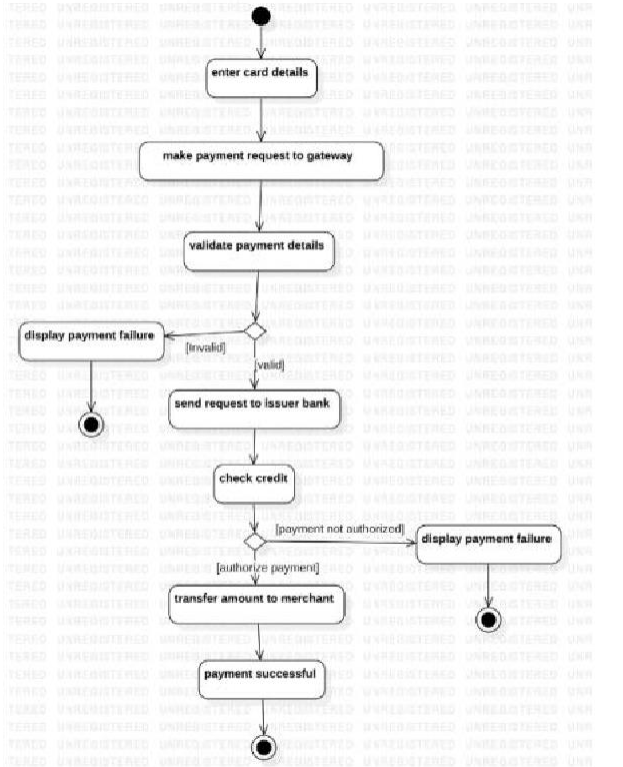
The collaboration diagram illustrates the flow of a credit card payment process involving four key participants: the Customer, Merchant, Payment Gateway, and Bank. The process begins when the customer initiates a purchase, after which the merchant sends the transaction details to the payment gateway. The gateway forwards an authorization request to the bank, which creates an authorization server instance to validate card details such as authenticity and available credit. Once validation succeeds, the bank authorizes the payment and transfers the approved amount to the merchant through the payment gateway. The system concludes by notifying the customer of successful payment. Overall, the diagram highlights the sequential and collaborative communication required to securely authenticate and process a credit card transaction.

Activity Diagram(Simple)



This activity diagram illustrates the workflow of a credit card transaction, beginning when the system receives a transaction request. The card details are first verified, and if they are invalid, the system immediately rejects the transaction and notifies the cardholder. If the card is valid, the system proceeds to check the available credit limit, branching into approval when the limit is sufficient or declining the transaction when the limit is insufficient. An approved transaction triggers steps to authorize the transaction, debit the cardholder's account, update the transaction database, notify the merchant of approval, and send confirmation to the cardholder. Finally, the system completes the cycle by generating a monthly statement. This diagram clearly captures the decision points and sequential actions involved in secure credit card payment processing.

Activity Diagram(Advanced)



This activity diagram illustrates the credit card payment process, beginning with the customer entering card details and the system sending a payment request to the payment gateway. The gateway validates the payment information, and if any details are invalid, the process immediately ends with a payment failure notification. For valid requests, the gateway forwards the transaction to the issuer bank, which checks the customer's credit availability. If the credit limit is insufficient or the bank declines authorization, the system displays a payment failure. When authorization is granted, the bank transfers the approved amount to the merchant, and the process concludes with a successful payment confirmation. This flow clearly represents the decision points and actions involved in completing or rejecting a credit card transaction.

3.LIBRARY MANAGEMENT SYSYTEM

Srs Document :

Date / / Page 14

5. LIBRARY MANAGEMENT SYSTEM

1. Introduction :-

(a) Purpose :- the system automatis library operations including book issue, return, cataloging and member management to save time & reduce errors

(b) Document Conventions :- All req. should use "shall", categories are function, non-functional and External interface

(c) Intended audience and Reading suggestion Librarians, students, faculty, developers, testers and managers

(d) Project Scope :- the system manages book records, member accounts, transaction and generate reports

(e) References :- IEEE std 830, library automation standard, and institutional library guidelines

2. Overall Description :-

(a) Product Perspective :- A studalone solution replacing manual register with options for future integration

Date / / Page 15

(b) Product functions :- Allows adding, removing books, issuing, returning generating reports and provide search functionality

(c) User classes and characteristics :- Librarians manages books and members students borrow book and admin accss the system

(d) operating environment :- works on windows / LINUX with MySQL DB and accessible via web browsers

(e) Design implementation constraints :- Must ensure data consistency, prevent issues of same copy and follow institutional policies

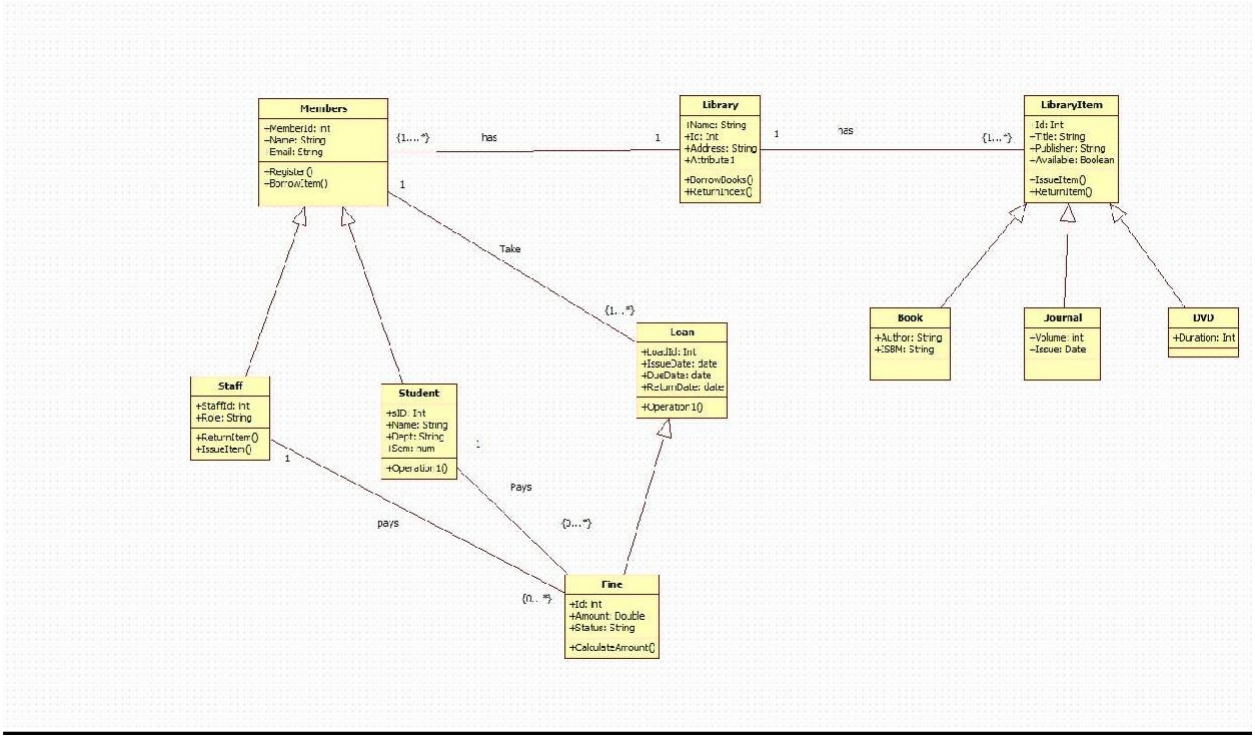
(f) User Documentation :- Includes manuals for staff, FAQ's for students and training guides

(g) Assumptions and Dependencies :- Users should have valid accounts, system depends on reliable database and network connection.

3. Specific Requirements :-

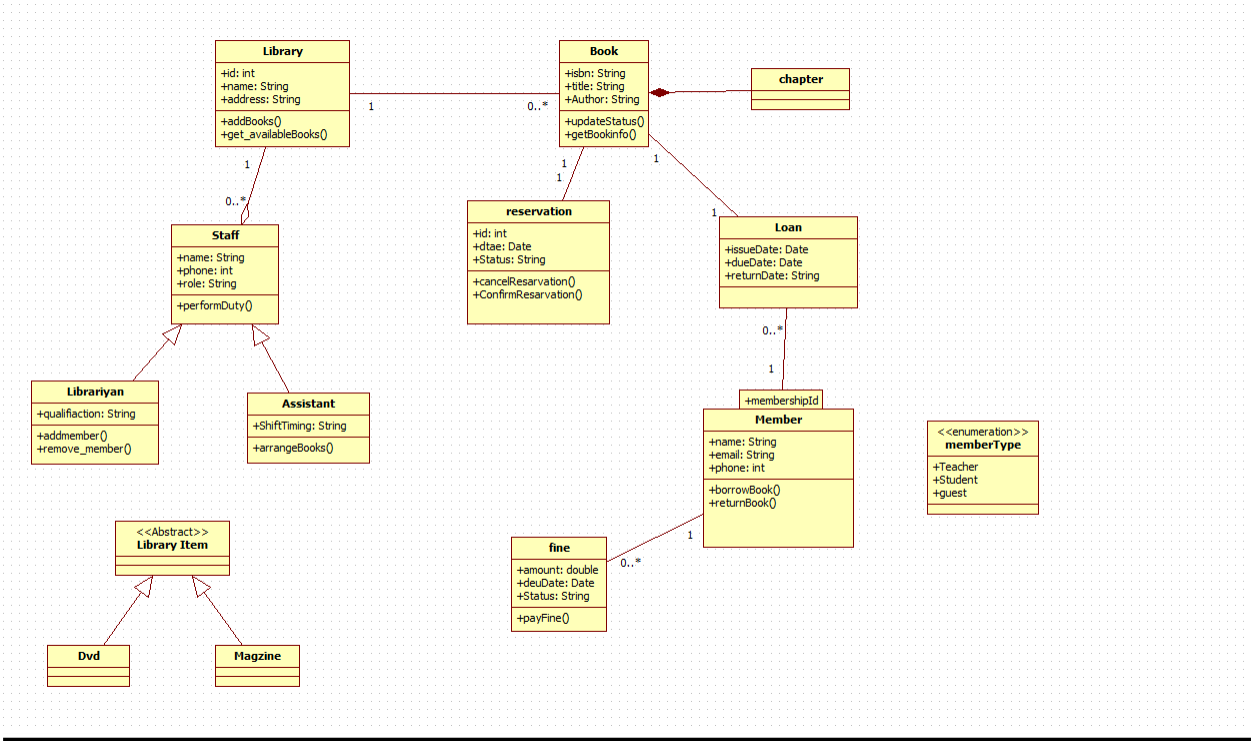
(a) Functional Requirements :- The system should allow book issue / return, manage due dates and fines,

Class Diagram(Simple)



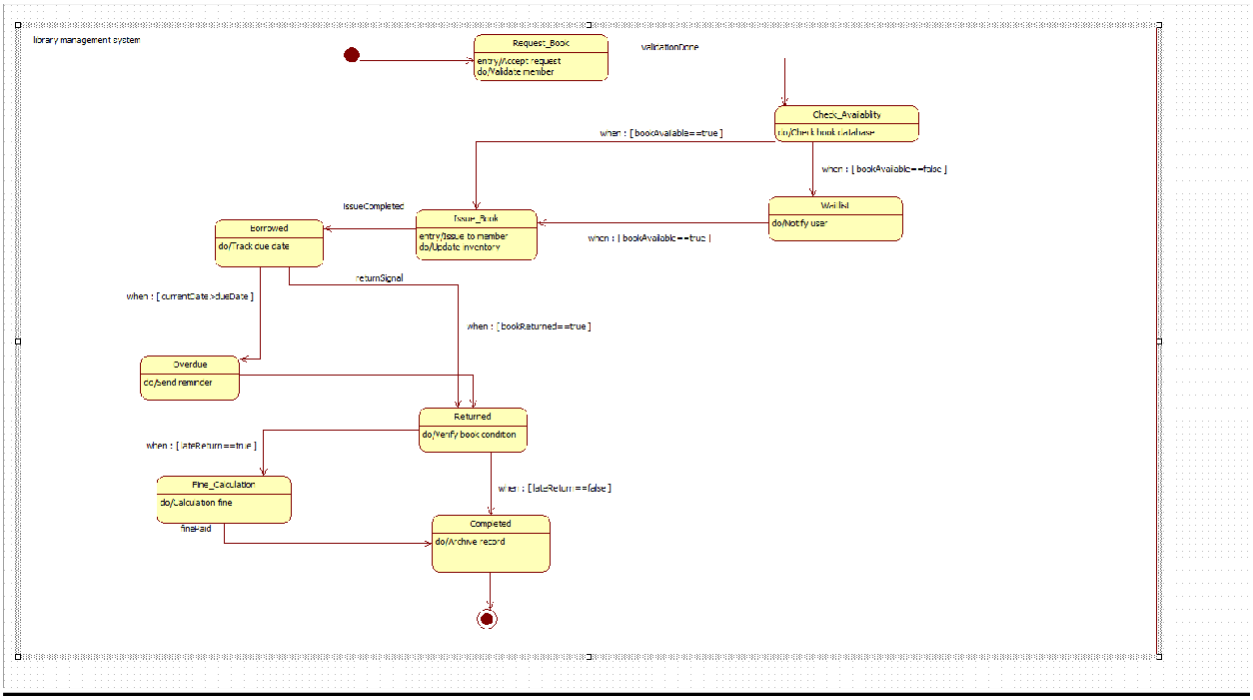
This class diagram represents a Library Management System where Members—including specialized types such as Staff and Students—can register and borrow items from the Library, which maintains multiple LibraryItems such as Books, Journals, and DVDs. Each borrowed item results in a Loan, recording issue and return dates, while overdue returns generate a Fine that members must pay. Library staff handle issuing and returning items, and the system tracks item availability, member borrowing activities, and fine calculations. Overall, the diagram illustrates how users, items, loans, and fines interact to support the operations of a typical library.

Class Diagram(Advanced)



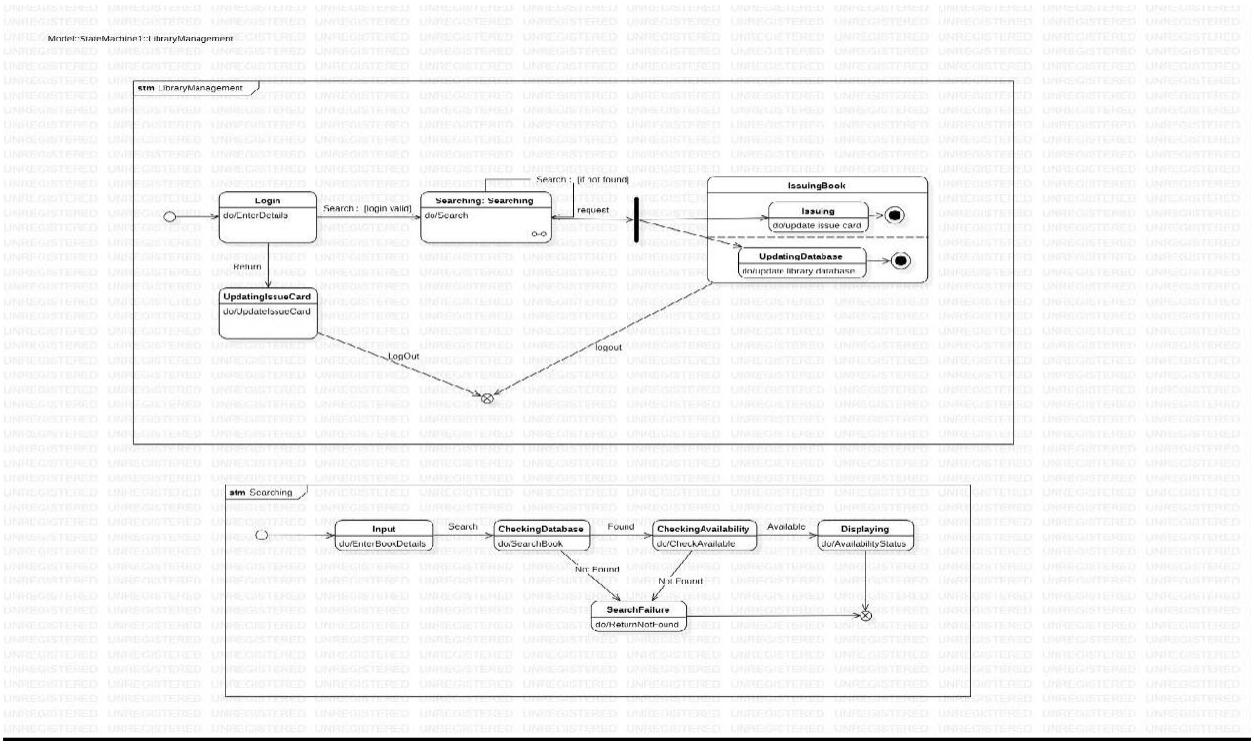
The class diagram represents a Library Management System where a Library manages various Books and other LibraryItems such as DVDs and Magazines, organized through an abstract superclass. Members of different types (Teacher, Student, Guest) can borrow books through the Loan process and make Reservations when items are not immediately available. The system includes Staff, such as Librarians who manage memberships and Assistants who organize books. Each book may contain multiple Chapters, and fines are generated through the Fine class for overdue returns. Overall, the diagram models key entities and their relationships required to support book borrowing, reservations, fines, and staff operations within a library.

State Diagram(Simple)



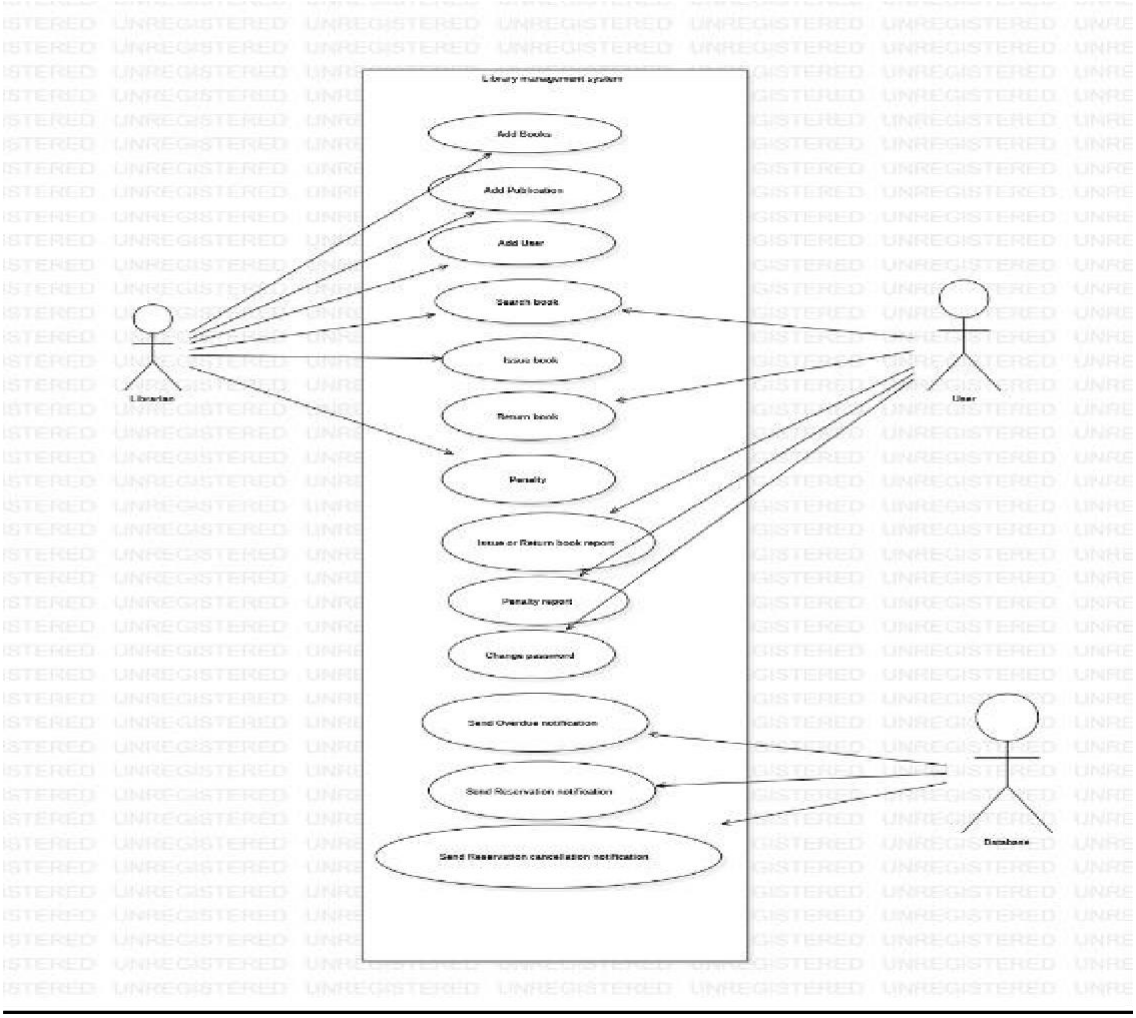
The state machine diagram illustrates the complete lifecycle of a book request in a library management system, beginning with a user initiating a Request_Book action where the system validates membership. If the book is available, the process moves to Issue_Book, updating inventory and handing the item to the member, after which the book enters the Borrowed state where due dates are tracked. If the due date passes without return, the item transitions to Overdue, triggering reminders. When the book is returned, the system moves to the Returned state to verify its condition. If the return is late, the flow proceeds to Fine_Calculation for computing penalties before completion; otherwise, it moves directly to Completed, where the transaction is archived. If the book is not available initially, the system directs the request to Waitlist, notifying the user accordingly.

Simple Diagram(Advanced)



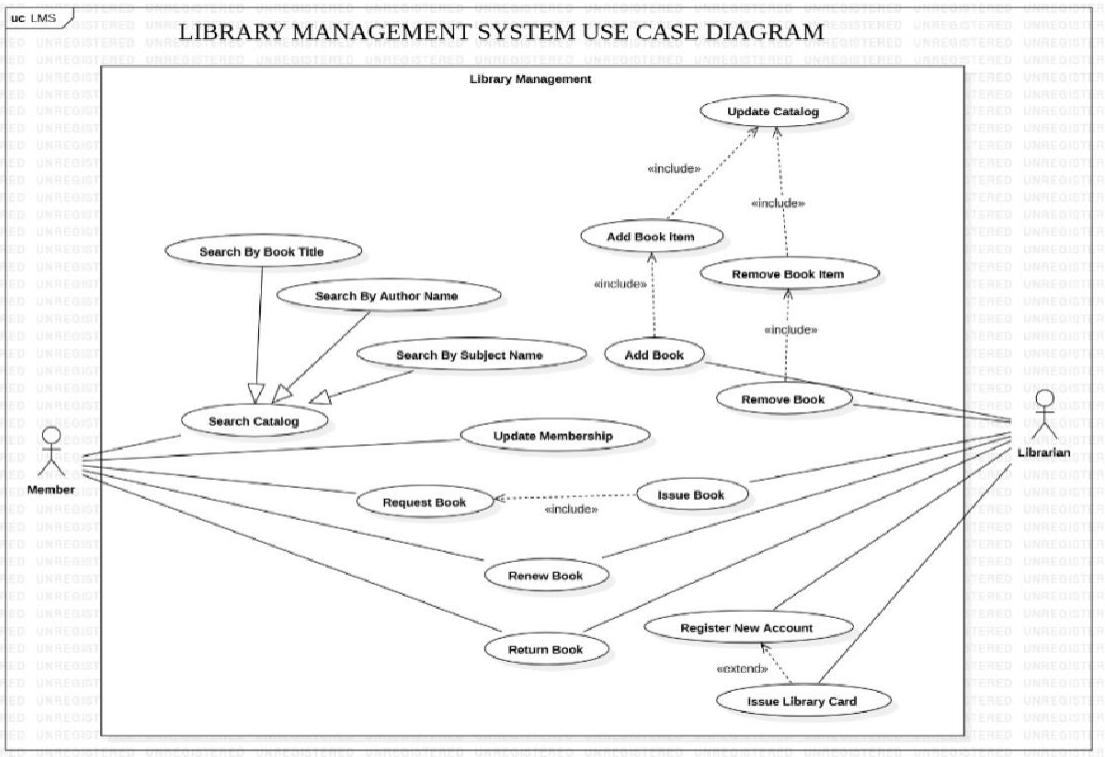
This state machine diagram represents the workflow of a library management system, beginning with the Login state where user credentials are entered and validated. Upon successful login, the system transitions to the Searching state, where the user searches for a book. The search process is further detailed in the nested *Searching* state machine, which verifies book details, checks availability, and displays results. If the requested book is available, the system proceeds to the IssuingBook region, consisting of two parallel activities—Issuing the book by updating the user's issue card and UpdatingDatabase to record the transaction. If the book is not found or unavailable, the system returns a failure message. After issuing or updating, the process can conclude with logout, returning the system to its final state. Overall, the diagram models login validation, book search, issuance, and database update operations in a coordinated flow.

Use Case Model(Simple)



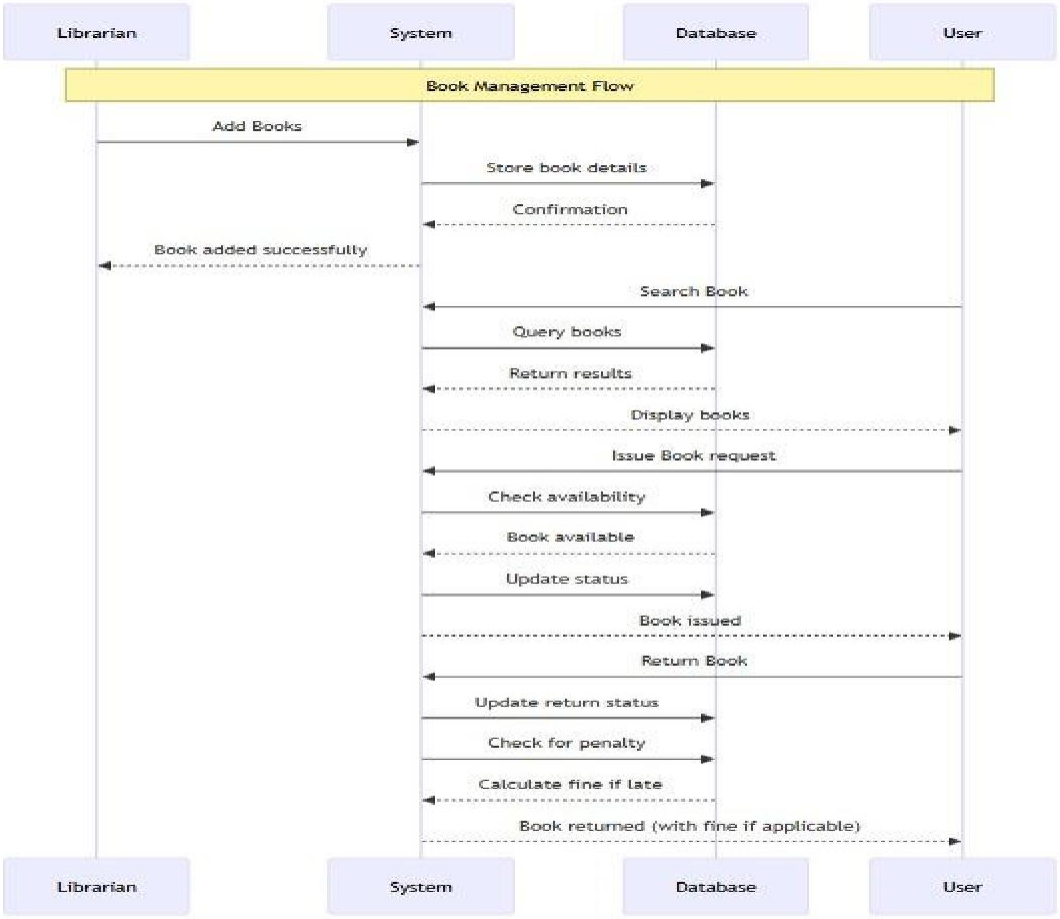
This use case diagram illustrates the interactions between three main actors—Librarian, User, and Database—within a library management system. The Librarian performs administrative tasks such as adding books, adding publications, adding users, issuing and returning books, generating penalty reports, and managing overdue, reservation, and cancellation notifications. The User interacts with the system mainly to search for books, request issue or return, pay penalties, and receive notifications. The Database supports backend operations by storing user data, book details, publications, and system-generated notifications. Overall, the diagram represents the functional requirements of the system, showing how different users utilize various features for efficient library operations and book management.

Use Case Model(Advanced)



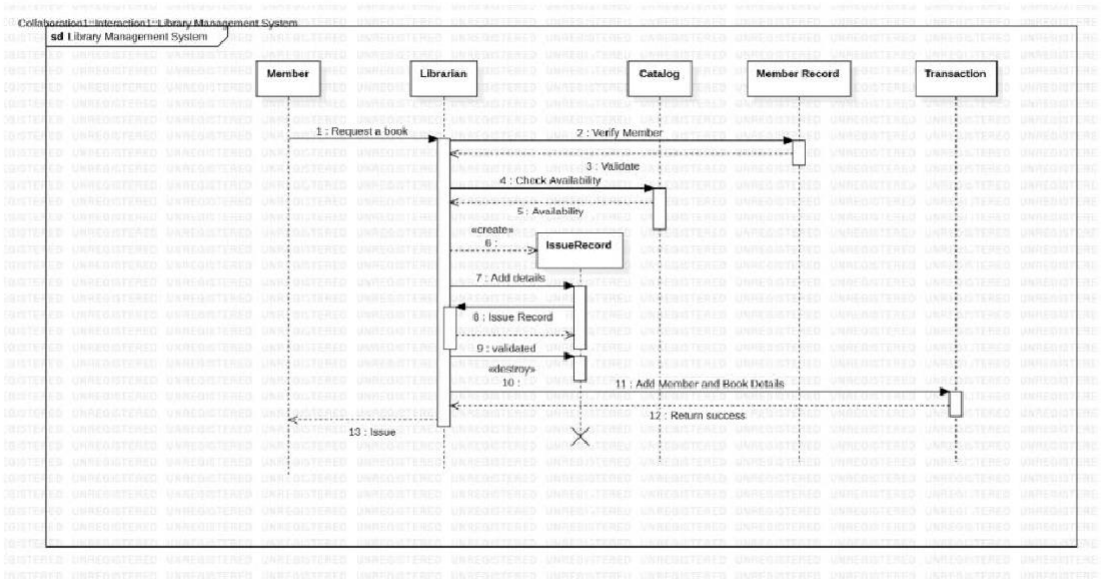
The use case diagram illustrates how the Library Management System supports both members and librarians in managing library operations. Members can search the catalog using book title, author name, or subject name, as well as request, renew, and return books, and update their membership details. Librarians handle administrative tasks such as registering new accounts, issuing library cards, adding and removing books, and updating the catalog. Several actions use *include* and *extend* relationships to show shared processes—such as issuing a book including the "Request Book" use case or registering an account extending to issuing a library card. Overall, the diagram clearly represents how different users interact with the system and how core library services are organized.

Sequence Model(Simple)



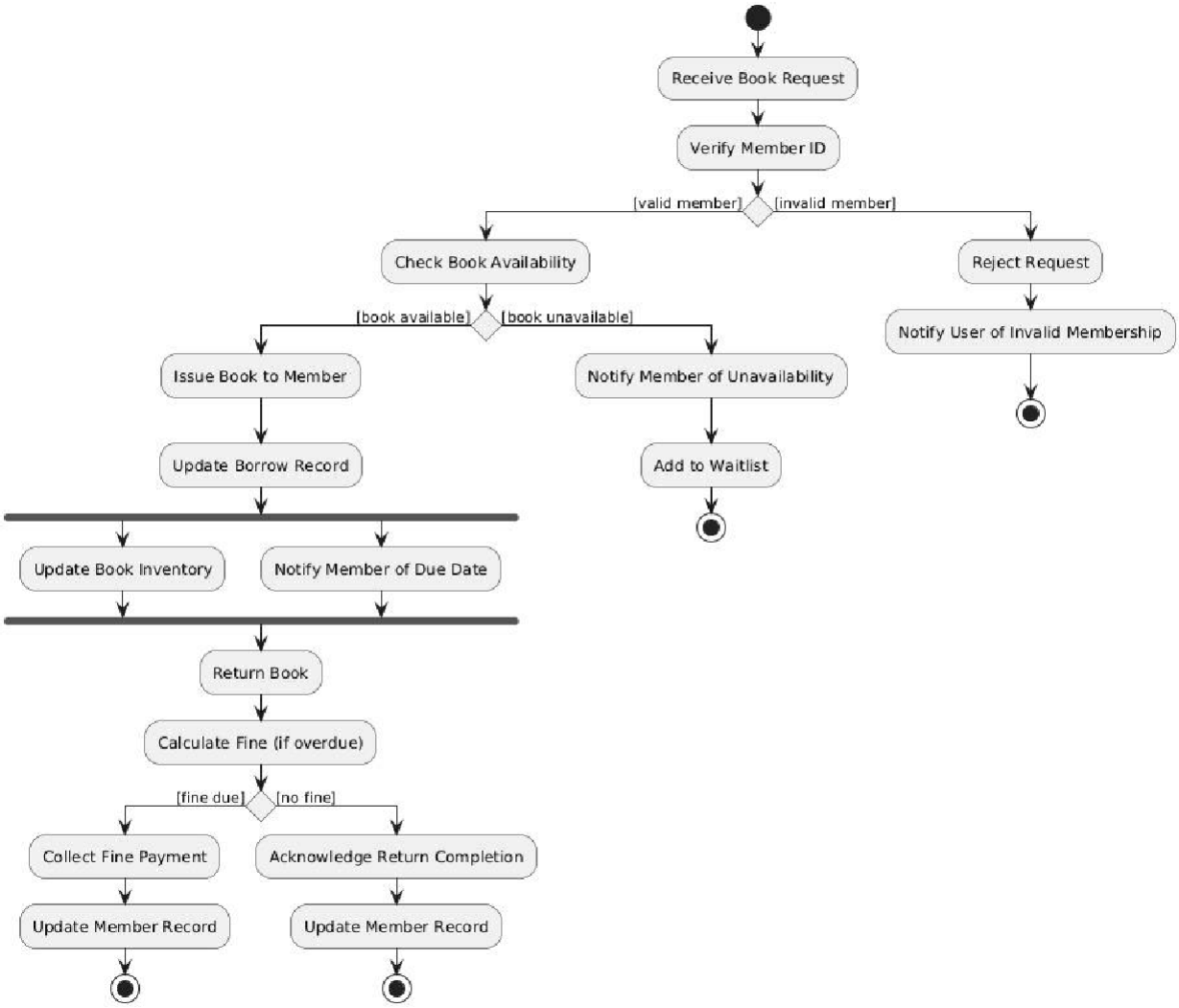
The Book Management Flow sequence diagram illustrates the interactions among the Librarian, System, Database, and User during key library operations. The process begins when the librarian adds new books, and the system stores the details in the database and sends a confirmation. Users can then search for books, and the system queries the database and displays results. When a user requests to issue a book, the system checks availability, updates the book status, and confirms issuance. During return, the system updates the return status, checks for penalties, calculates any applicable fines, and records the final return in the database. This flow ensures efficient management of book addition, search, issuance, and return activities within the library system.

Sequence Model(Advanced)



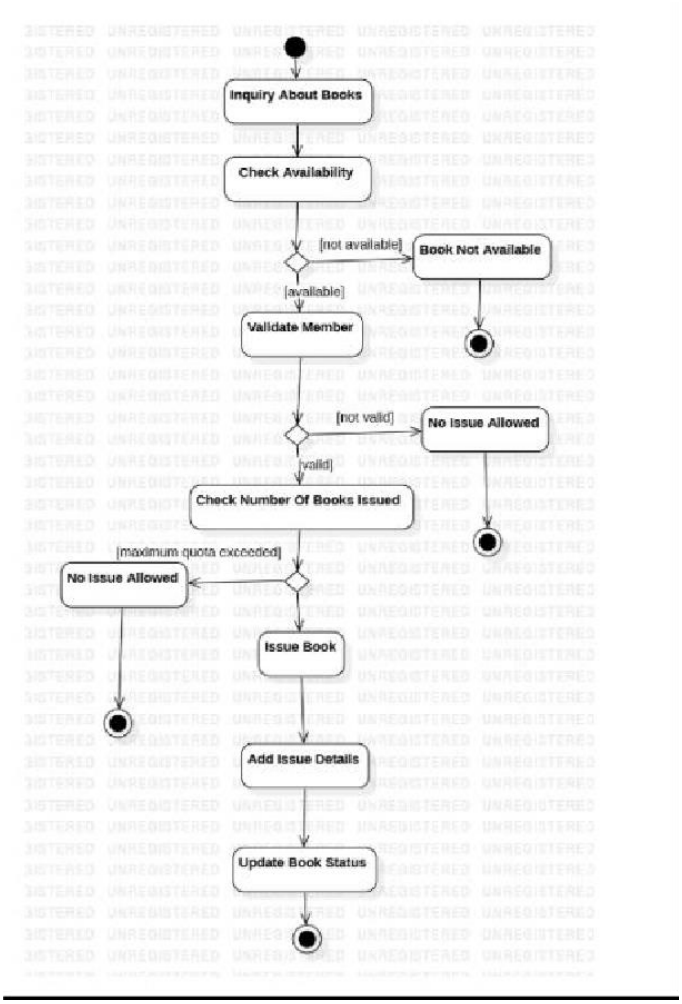
This collaboration diagram illustrates the interaction between different components of a Library Management System during the book-issuing process. The sequence begins with a Member requesting a book, after which the Librarian verifies the member's details and checks the book's availability through the Catalog. If the book is available, an IssueRecord is created to store the transaction details. The Librarian updates both the Member Record and the Catalog to reflect the issued book before sending the transaction information to the Transaction component, which logs and confirms the issue. Finally, a success response is returned to the Member, completing the transaction flow. This diagram clearly models how system components collaborate to validate, process, and record a book-issuance request.

Activity Diagram(Simple)



This activity diagram illustrates the complete workflow of issuing and returning a book in a library management system. The process begins when a librarian receives a book request and verifies the member's ID. If the membership is valid, the system checks the availability of the requested book. Available books are issued to the member, the borrow record is updated, inventory is adjusted, and the member is notified of the due date. If the book is unavailable, the member is informed and added to a waitlist. For invalid members, the request is rejected with an appropriate notification. The return process involves calculating any applicable fines for overdue books; if a fine is due, payment is collected before updating the member's record, otherwise the return is simply acknowledged. The workflow ensures proper tracking of book availability, member status, due dates, and fines within the library system.

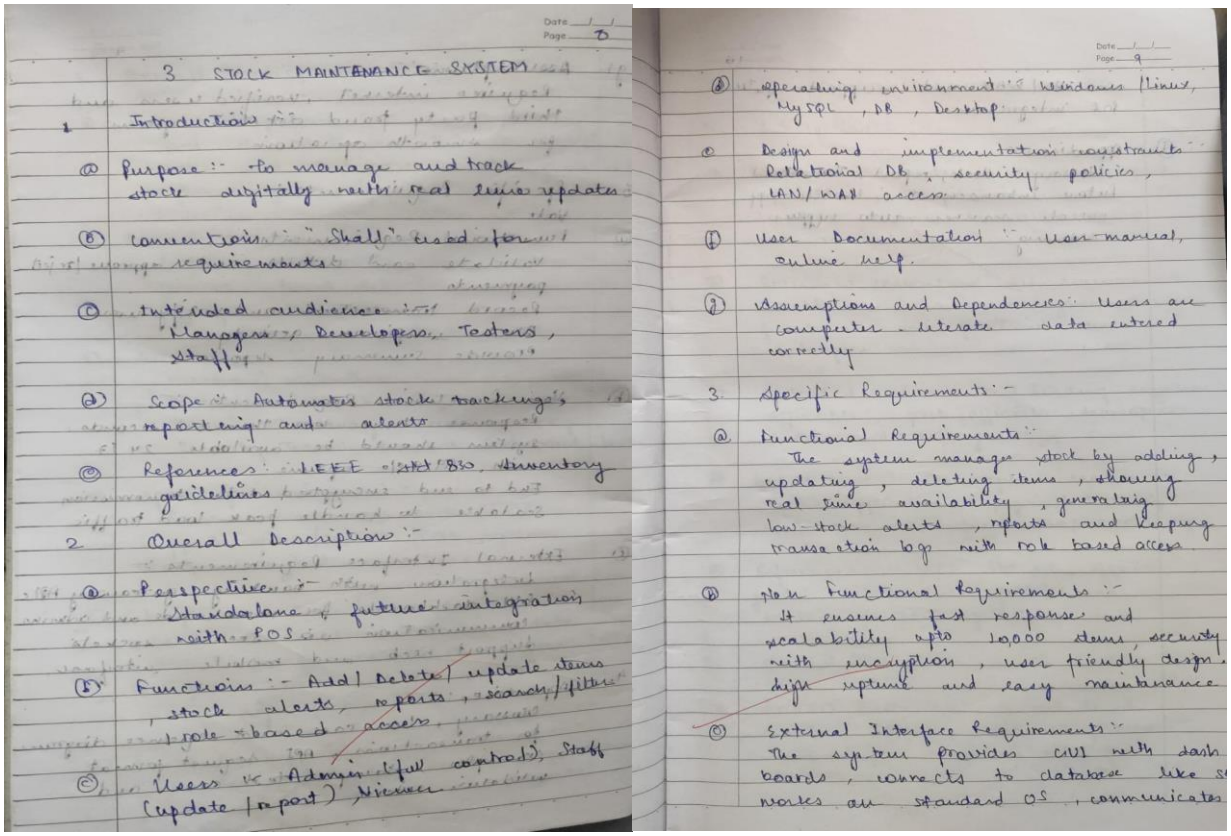
Activity Diagram(Advanced)



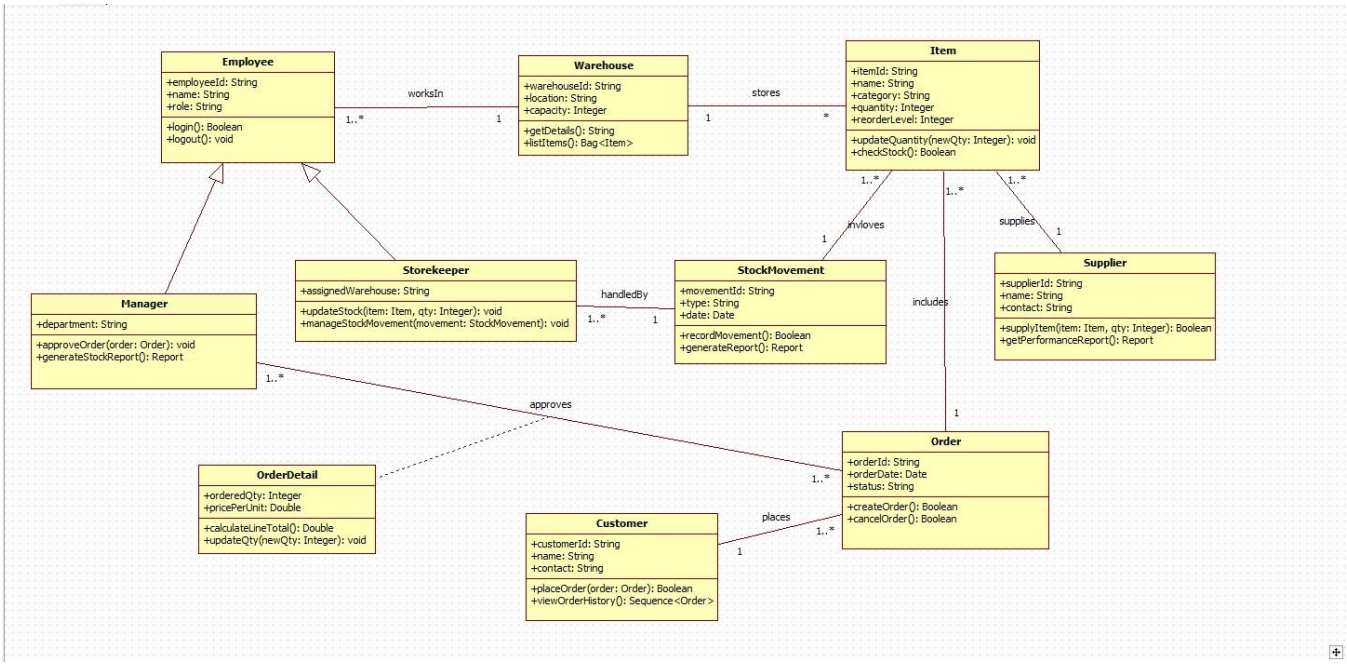
This activity diagram illustrates the workflow followed when a user inquires about a book in the library system. The process begins with checking the availability of the requested book; if it is unavailable, the system notifies the user and terminates the process. If the book is available, the system proceeds to validate the member's credentials. Invalid members are denied access and cannot issue books. For valid members, the system checks how many books they have already issued; if the maximum quota is exceeded, issuing is not allowed. If all conditions are met, the book is issued to the member, the issue details are recorded, and the book status is updated in the library database. The diagram captures all decision points and system actions required to manage book inquiries and ensure proper issuing rules are followed.

4.STOCK MAINTENANCE SYSTEM

Srs Document :



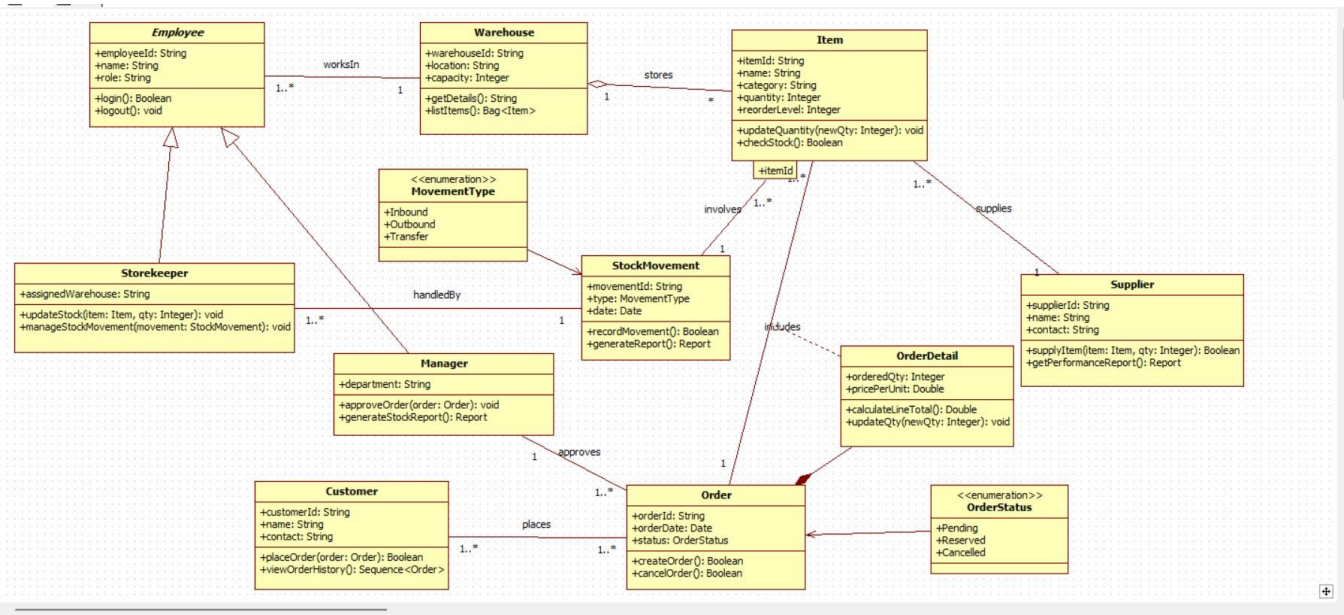
Class Diagram(Simple)



This class diagram represents an inventory management system where the Inventory maintains a collection of Products, each holding details such as name, price, quantity, and category.

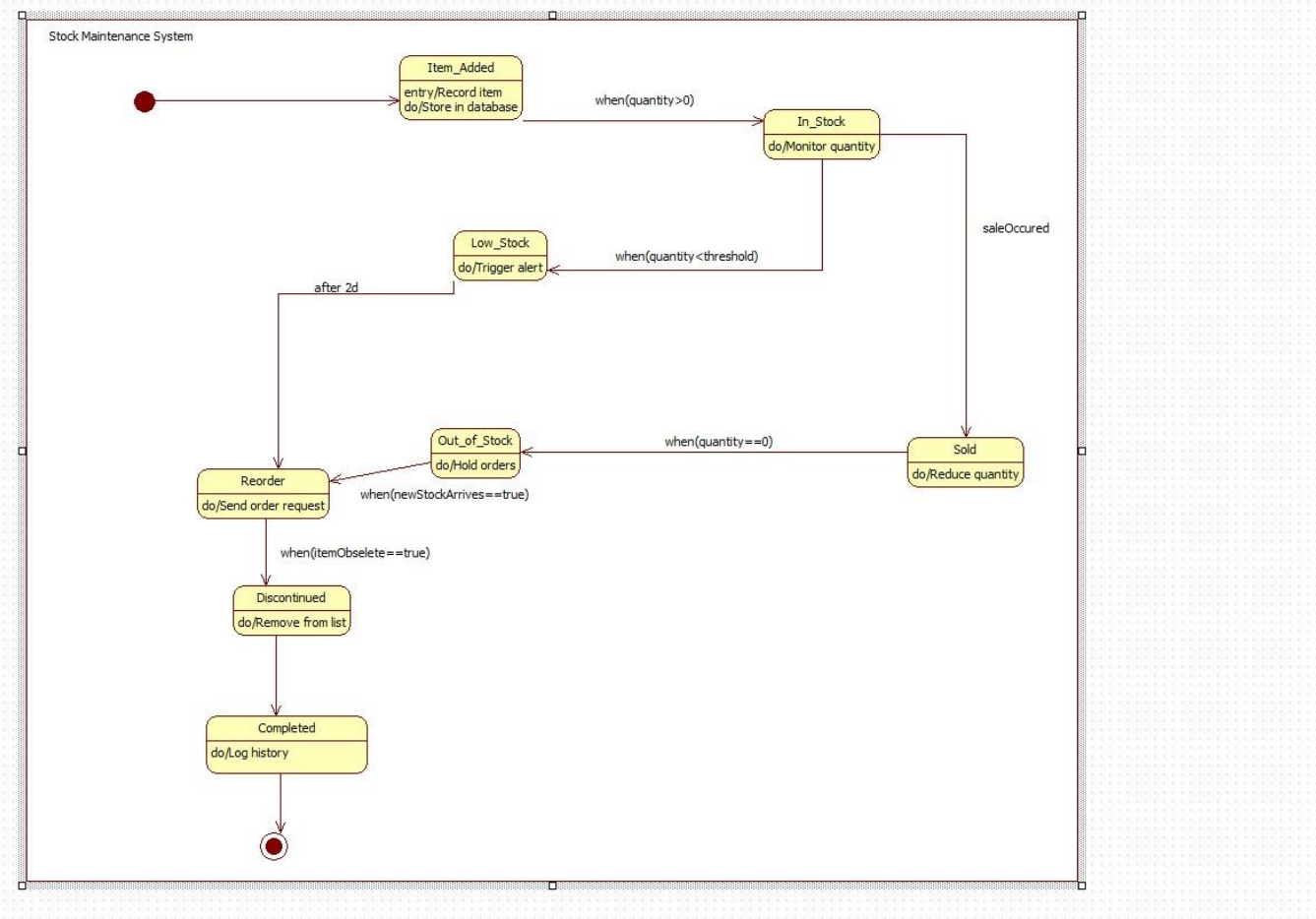
Suppliers provide products to the inventory, and their deliveries are recorded as StockTransactions, which track quantities and dates of stock updates. Customers place Orders, each linked to one or more products and processed through the system. The Staff class represents employees who manage inventory operations, with specialized roles such as Manager (who generates reports) and Clerk (who updates inventory). Overall, the diagram models the flow of products from suppliers into inventory, their management by staff, and their purchase by customers, ensuring a structured and efficient inventory control process.

Class Diagram(Advanced)



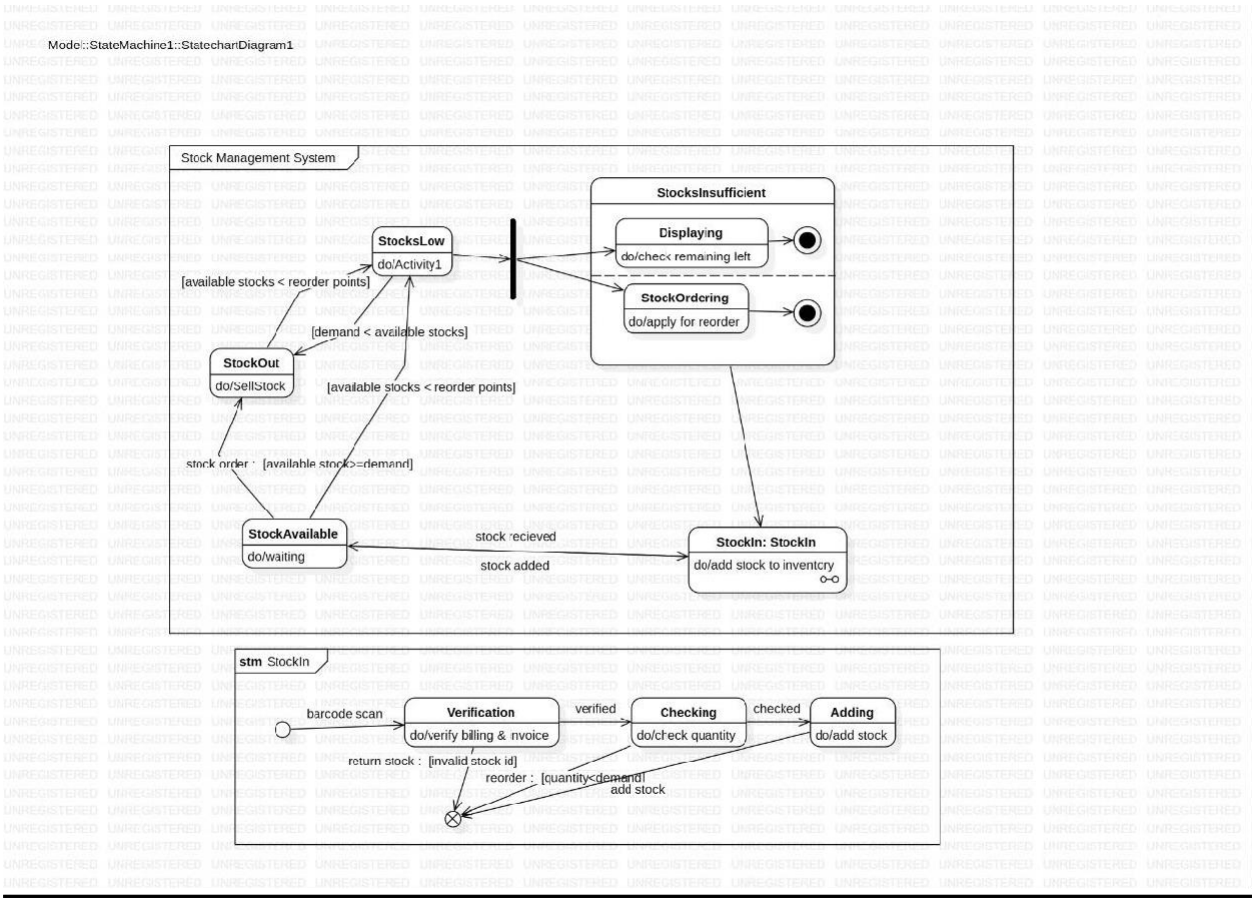
The class diagram represents an Inventory Management System where Inventory, Product, and Batch/StockItem form the core components of stock handling. The Inventory maintains details of stored products, while each Product may be linked to multiple stock items, categorized as RawMaterial or FinishedProduct. Suppliers provide products through supply operations, and StockTransaction records movements of stock within the system. Customer purchases are processed through Orders, which associate customers with products and quantities. The Staff entity oversees system operations, with specialized roles such as Manager for reports and Clerk for inventory updates. The StockItemType enumeration defines item conditions such as available, out of stock, or damaged. Overall, the system models how products are stored, updated, supplied, sold, and managed through well-defined interactions among inventory, people, and stock entities.

State Transition Diagram(Simple)



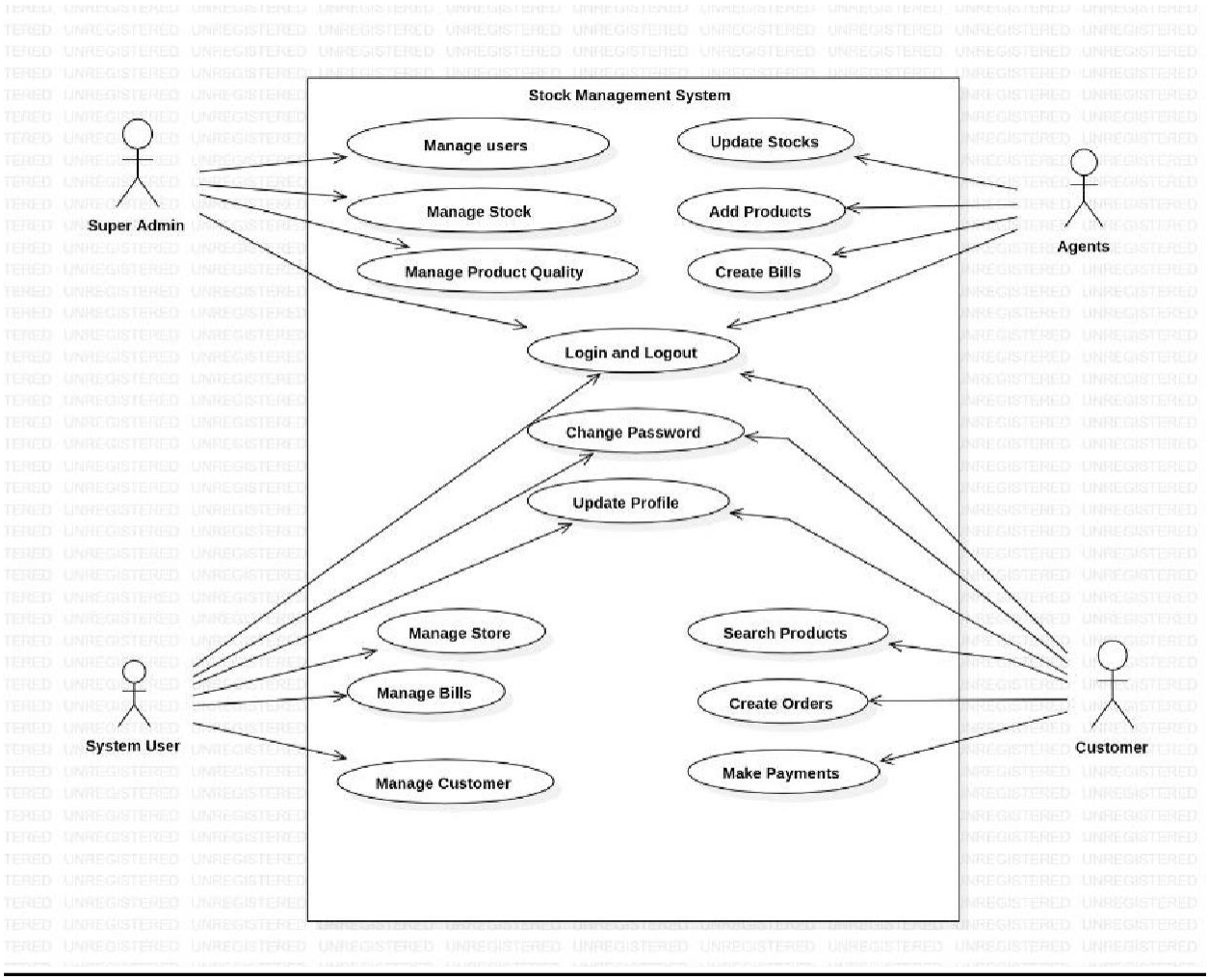
The Stock Maintenance System begins when a new item is added and recorded in the database, transitioning to the *In_Stock* state where its quantity is continuously monitored. When a sale occurs, the system reduces the quantity, and if the quantity falls below a defined threshold, it moves to the *Low_Stock* state and triggers an alert. After a specified time delay, the system initiates a *Reorder* process, sending an order request to replenish inventory. If the quantity reaches zero, the item enters the *Out_of_Stock* state, where incoming orders are put on hold until new stock arrives. Once new stock is received, the item is returned to normal stock flow; however, if the item becomes obsolete, it transitions to the *Discontinued* state and is removed from the active list. The workflow concludes in the *Completed* state, where the system logs the item's history for recordkeeping.

State Transition Diagram(Advanced)



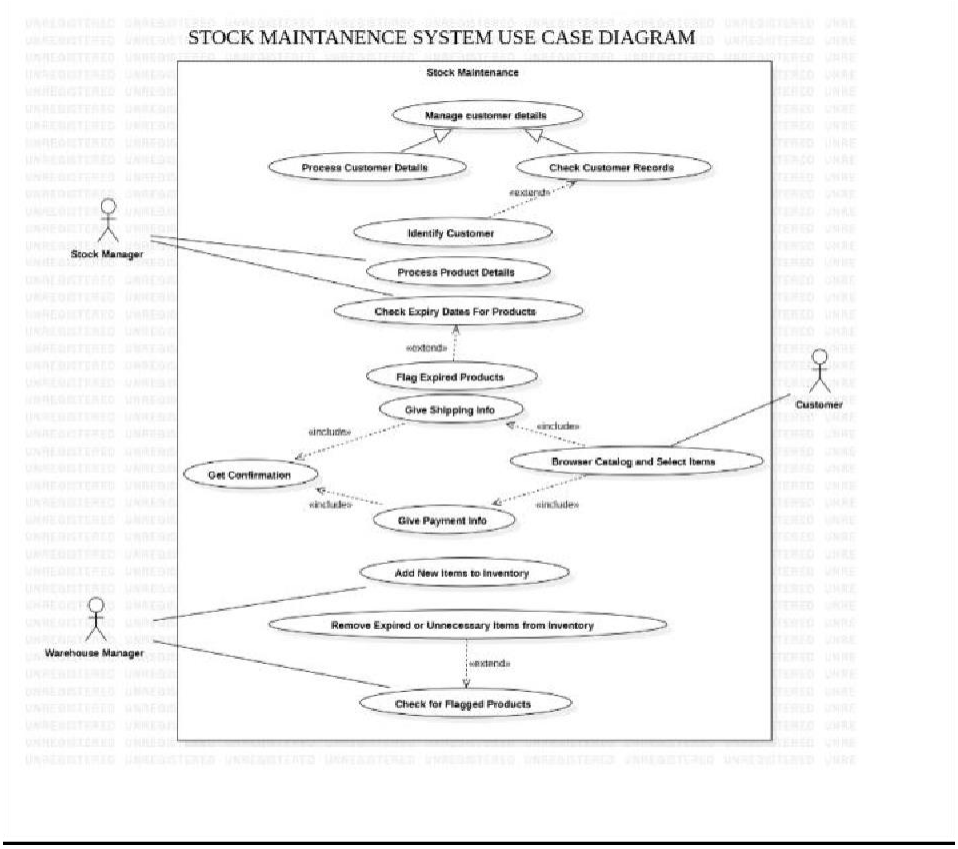
The Stock Management System tracks inventory levels by transitioning between states based on availability, demand, and reorder conditions. When stock levels drop below the reorder point, the system enters the *StocksLow* state and may move to *StockOut* if demand exceeds available stock, triggering a stock order. If stock is insufficient, the system transitions into the composite *StocksInsufficient* state, where it either displays the remaining quantity or begins the reorder process. Once new stock is received, the system moves to the *StockIn* state, where items are verified, checked, and added into inventory through a structured subprocess involving billing verification, quantity checking, and stock addition. After stock is successfully updated, the system returns to the *StockAvailable* state, ensuring continuous monitoring and smooth replenishment of inventory.

Use Case(Simple)



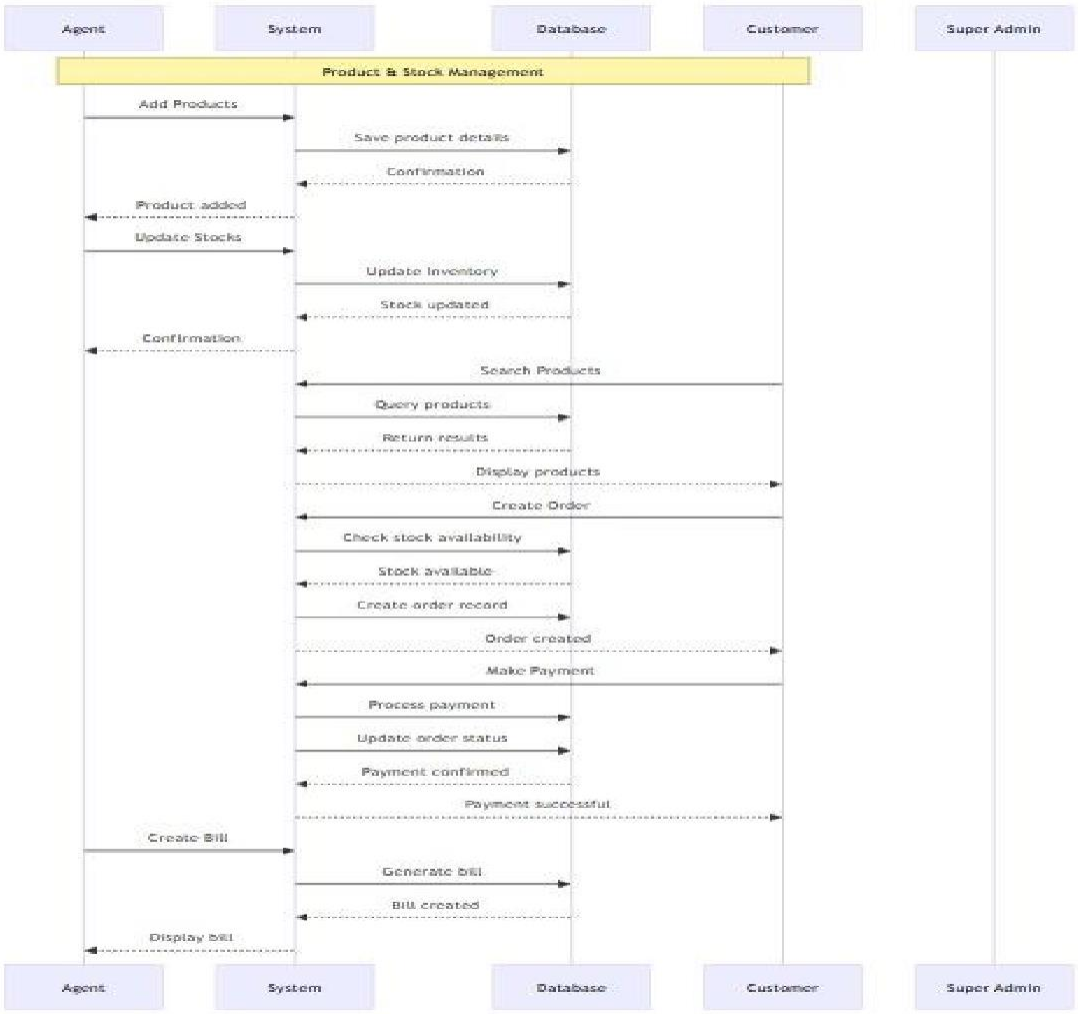
The use-case diagram illustrates the interactions between different types of users and the Stock Management System, highlighting the distinct roles and functionalities available to each. The Super Admin has the highest level of control, with the ability to manage users, stock, and product quality, ensuring smooth system administration. System Users handle day-to-day operational tasks such as managing stores, bills, and customer information. Agents interact with the system to update stocks, add products, and generate bills, supporting business operations. Customers are provided functionalities like searching products, creating orders, and making payments. Common use cases such as login/logout, password changes, and profile updates are shared across multiple actors, ensuring secure and personalized access for all users. Overall, the diagram clearly represents how each user category contributes to the efficient functioning of the Stock Management System.

Use Case(Advanced)



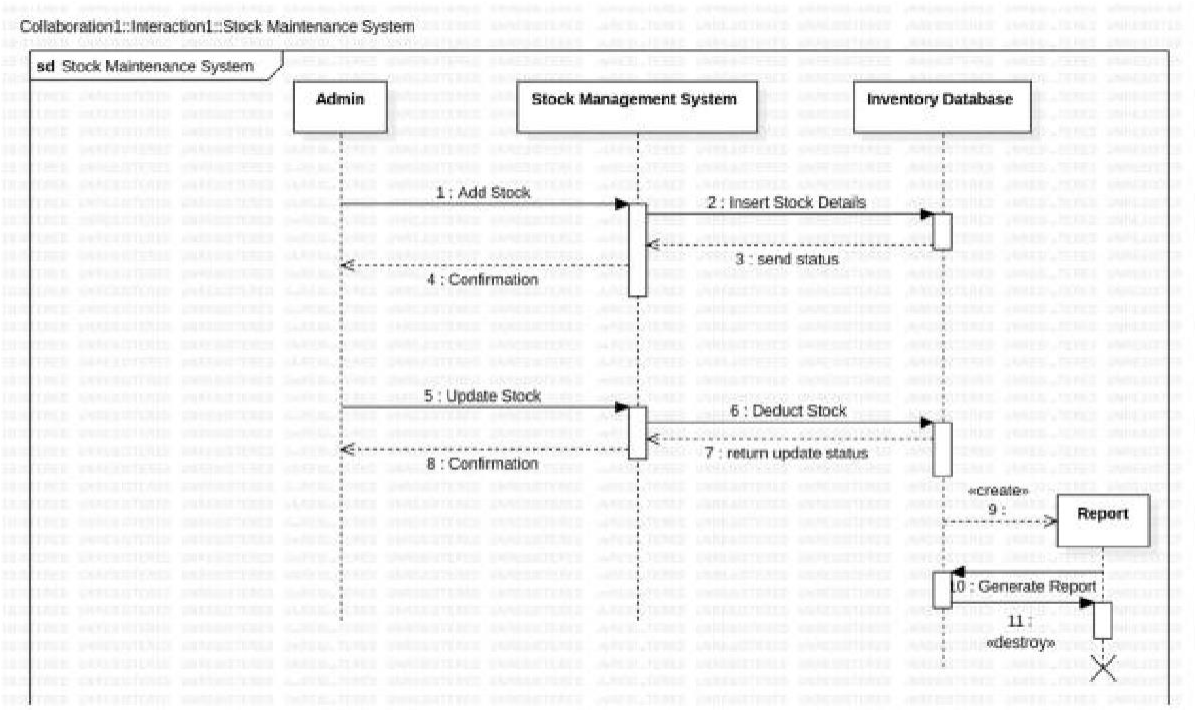
The Stock Maintenance System use case diagram illustrates the interactions between the Stock Manager, Warehouse Manager, and Customer as they perform various operations within the system. The Stock Manager manages customer details, processes customer records, identifies customers, handles product details, checks expiry dates, and flags expired products. The Customer interacts with the system to browse the catalog, select items, provide shipping and payment information, and receive confirmation, with these actions supported through include and extend relationships. The Warehouse Manager is responsible for updating physical inventory by adding new items and removing expired or unnecessary products, along with checking for flagged items. Overall, the diagram clearly represents how different actors collaborate to maintain stock accuracy, ensure product quality, and facilitate smooth customer transactions.

Sequence Diagram(Simple)



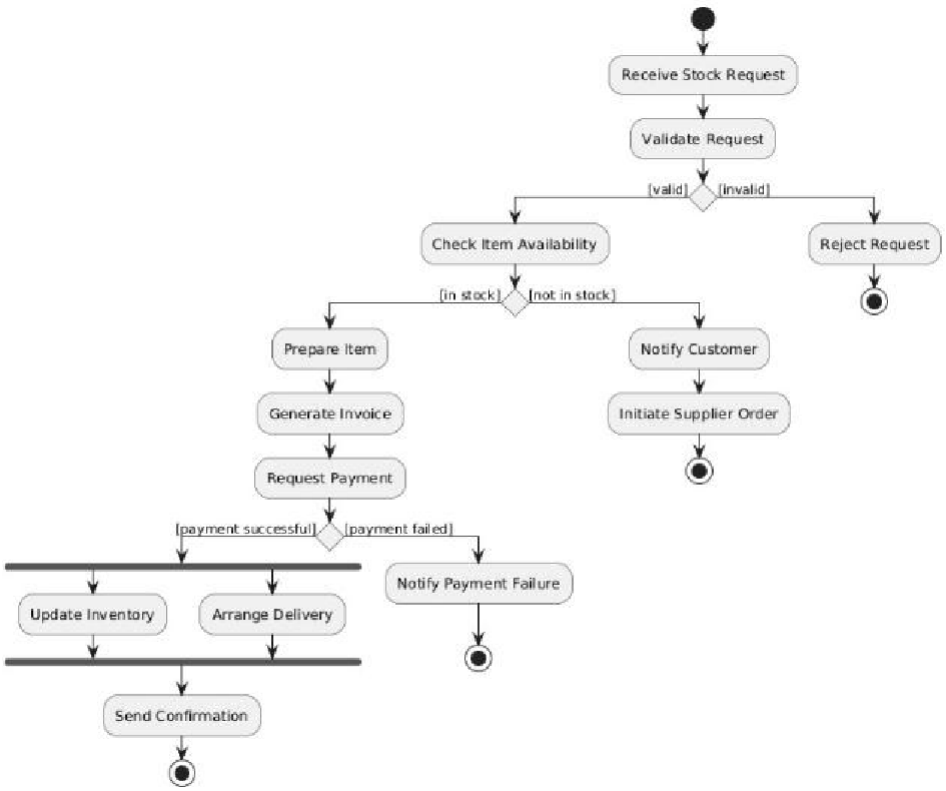
The sequence diagram illustrates the interaction flow within the Product and Stock Management module, detailing how agents, customers, the system, and the database communicate to complete product, stock, and order-related operations. The process begins when an agent adds products, prompting the system to store product details in the database and return a confirmation. Agents can also update stock levels, which the system records in the inventory. Customers interact by searching and browsing products, with the system querying the database and displaying relevant results. When a customer creates an order, the system checks stock availability, confirms item availability, and records the order. For payments, the customer initiates the transaction, the system processes it through the database, updates the order status, and sends back a confirmation. Finally, the agent generates a bill, which the system creates using stored order information and displays to complete the transaction workflow. This diagram effectively shows how different actors collaborate through the system to manage stock, process orders, and handle billing in a streamlined and systematic manner.

Sequence Diagram(Advanced)



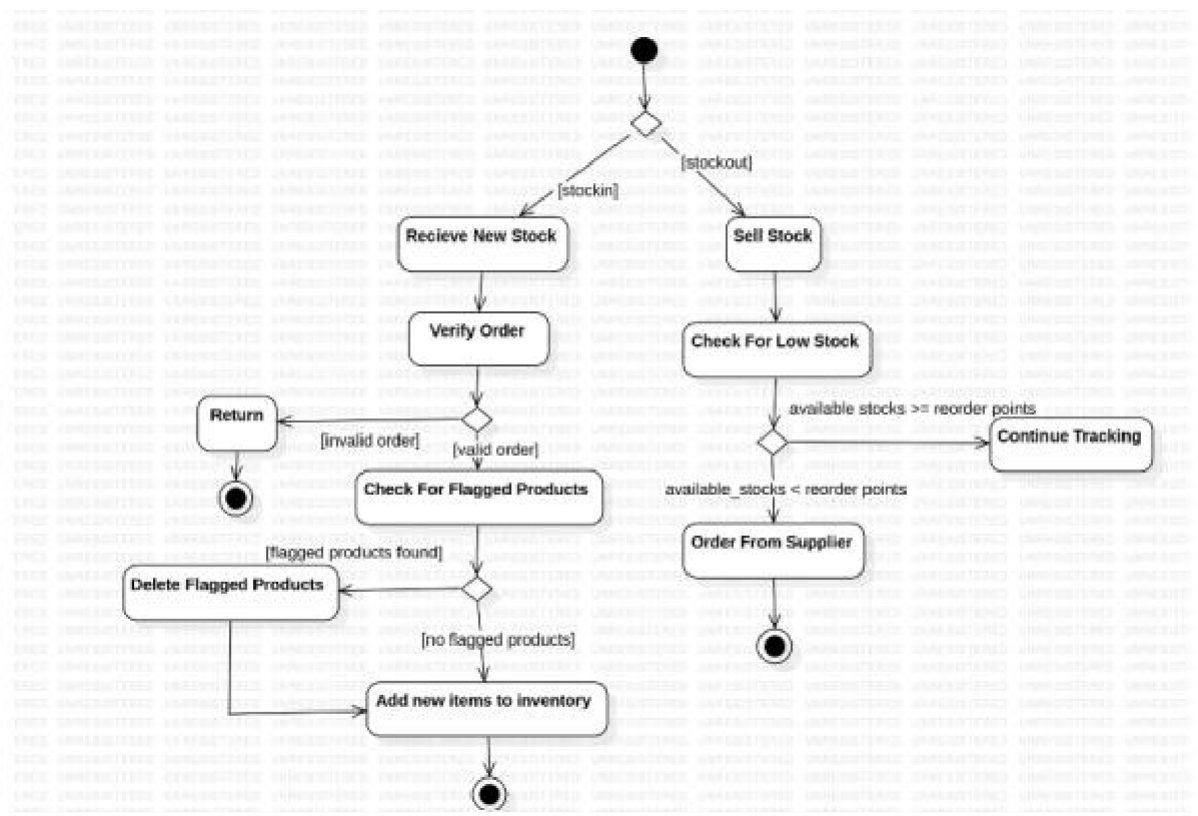
The sequence diagram represents the interaction flow within the Stock Maintenance System, showing how the Admin, Stock Management System, and Inventory Database collaborate to manage stock updates and generate reports. The process begins when the Admin initiates an *Add Stock* request, prompting the Stock Management System to insert stock details into the Inventory Database, which then returns a status update for confirmation. The Admin can also request to *Update Stock*, after which the system sends a request to deduct or modify stock levels in the database, receives an update status, and returns a confirmation to the Admin. Additionally, the system can create a Report object, instruct it to generate a stock report using inventory data, and then destroy the object after completion. This diagram clearly illustrates the step-by-step communication and message flow involved in maintaining stock information and generating administrative reports.

Activity Model(Simple)



The activity diagram illustrates the workflow for processing a stock request, beginning when the system receives and validates the request. If the request is invalid, it is immediately rejected; if valid, the system checks item availability. When the item is in stock, the process moves forward with preparing the item, generating an invoice, and requesting payment. A successful payment leads to two parallel activities: updating the inventory and arranging delivery, after which the system sends a confirmation to the requester. If the payment fails, a payment failure notification is sent and the process ends. When the item is not in stock, the system notifies the customer and initiates a supplier order before terminating the activity. Overall, the diagram clearly demonstrates decision points, parallel flows, and end states involved in handling stock requests efficiently.

Activity Diagram(Advanced)



The activity diagram illustrates the workflow of stock-in and stock-out processes within the stock maintenance system, beginning with a decision that branches into either receiving new stock or selling existing stock. When new stock is received, the order is verified; invalid orders are returned, whereas valid orders proceed to a check for flagged or defective products. If flagged products are detected, they are deleted before the remaining items are added to the inventory; if none are found, the system directly updates the stock with new items. On the stock- out side, once stock is sold, the system checks for low stock conditions. If available stock remains above the reorder point, normal tracking continues, but if stock falls below the threshold, an order is automatically placed with the supplier. The diagram effectively captures decision points, validations, and exception handling involved in maintaining accurate and reliable stock levels.

5.PASSPORT AUTHENTICATION SYSTEM

Srs Document :

Date / /
Page 4

4. PASSPORT AUTOMATION SYSTEM

1. Introduction:

(a) Purpose: the system automates passport application, verification and issuance online to reduce delay and manual work.

(b) Document conventions: All requirement would use "shall", categories are functional, non-functional and interface

(c) Intended audience and reading suggestion: Applicants, govt officials, developers, tester and managers

(d) Project Scope: The system enables online applications, payment, tracking verification and final passport delivery

(e) References: IEEE Std 830, official govt e-Governance policies and existing passport guidelines

2. Overall Description:

(a) Product Perspective: A web based solution replacing manual process designed for integration with govt databases.

(b) Product functions: Users apply online upload documents, pay fees, schedule appointments, track status and officials issue passports

(c) User classes and characteristics Applicants submit request; verifies check and approve document and admins control the system

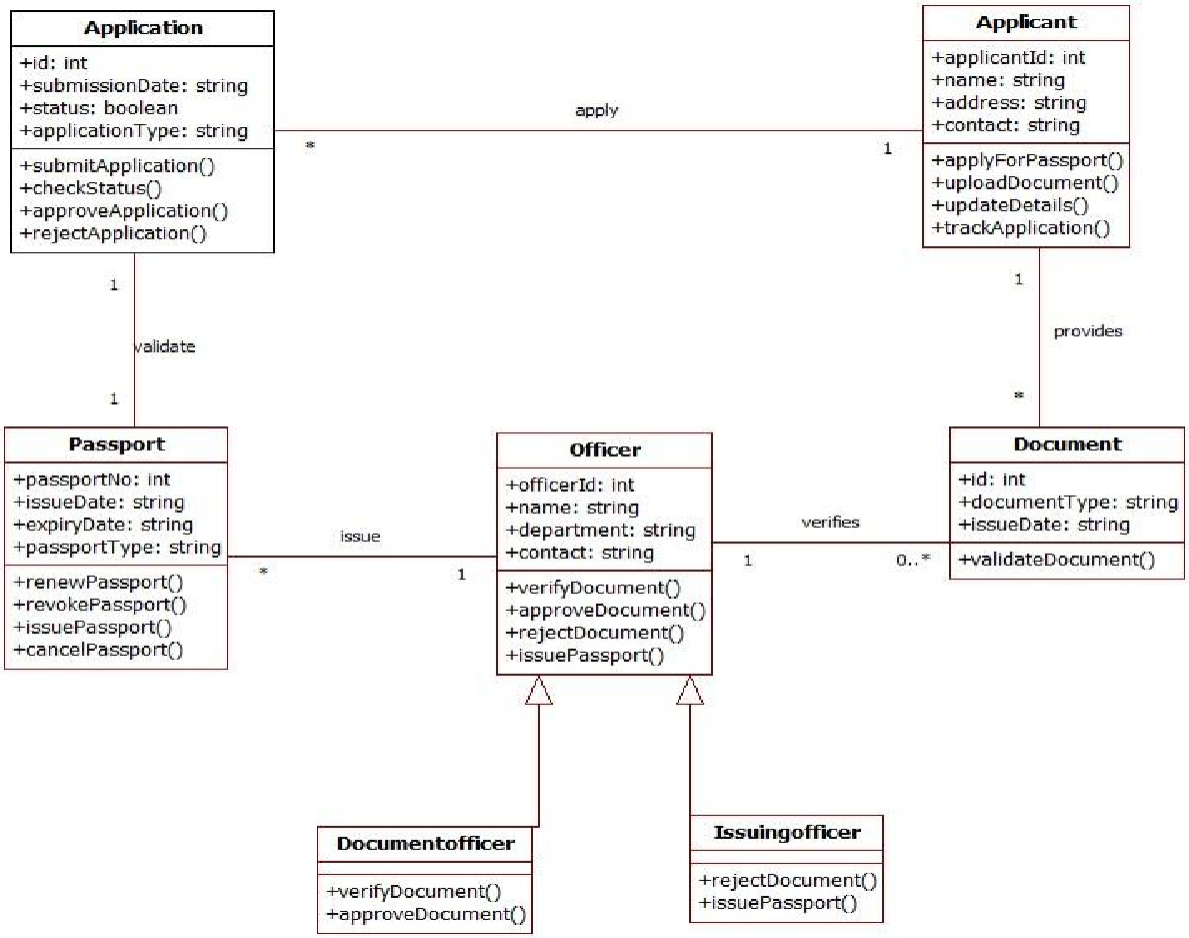
(d) Operating Environment: Works on windows/LINUX servers with MySQL DB, accessible via common browsers

(e) Design and implementation constraints: Must follow govt data protection, security standards and handle large user traffic

(f) User Documentation: Provides user manuals, online help and FAQ's for applicants

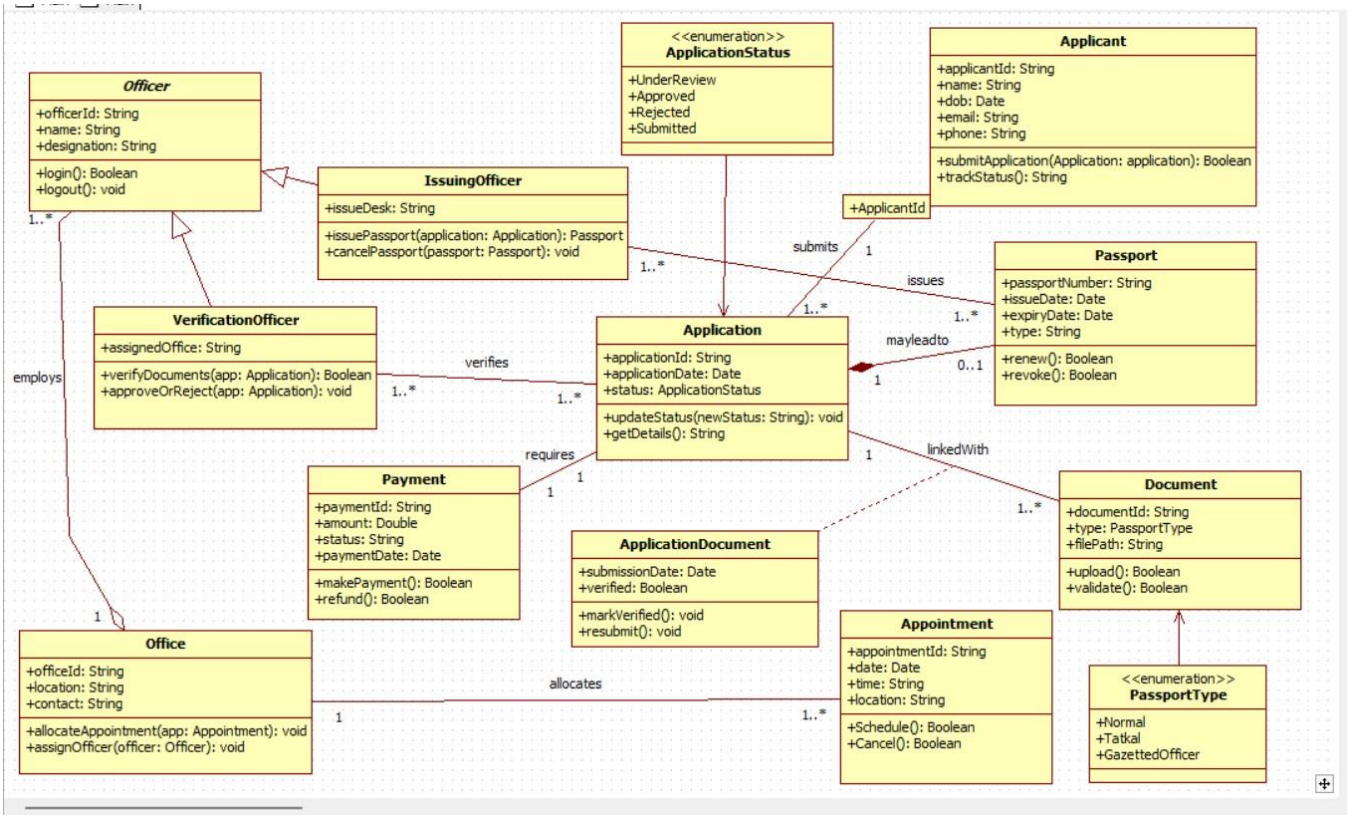
(g) Assumptions and Dependencies: Users have internet access and valid documents System depends on secure payment methods

Class Diagram(Simple)



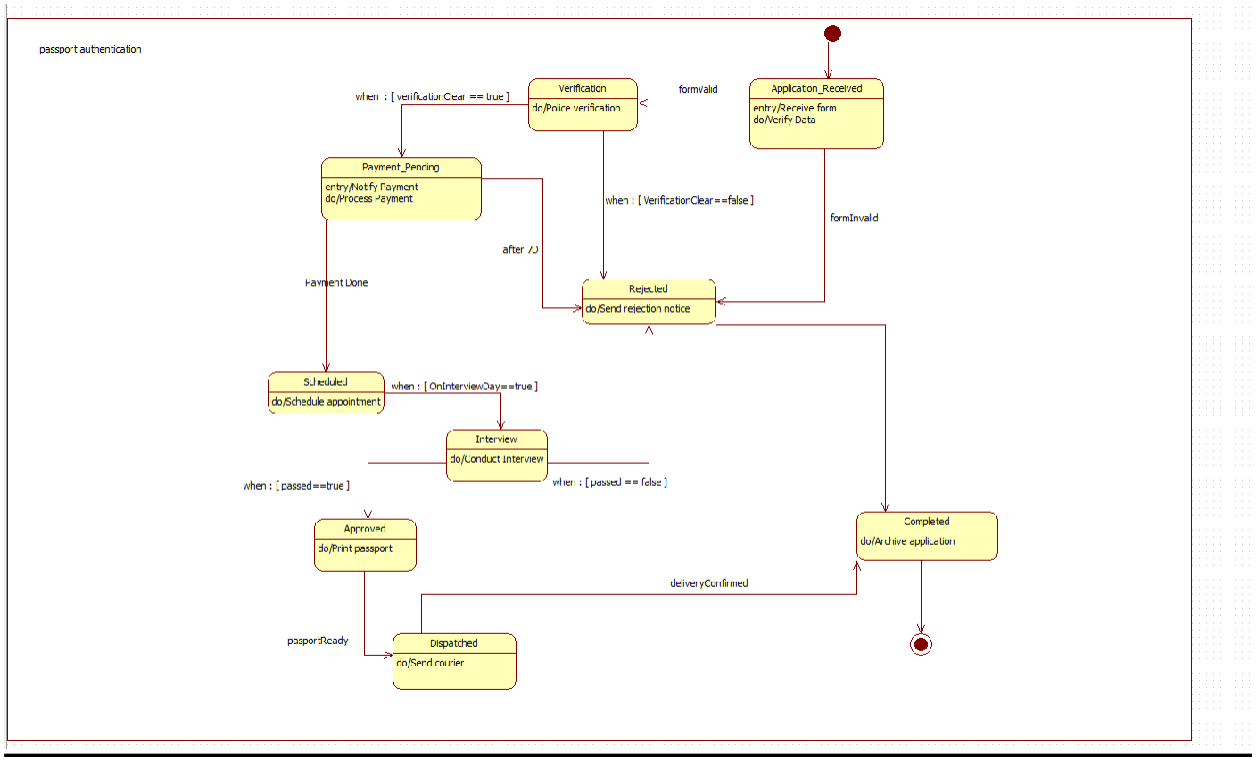
The UML class diagram represents a passport management system involving key entities such as Applicant, Application, Document, Officer, and Passport, along with specialized officer types. An Applicant submits an Application and provides multiple supporting Documents, each of which can be validated by an Officer. The Officer verifies, approves, or rejects documents and is responsible for issuing passports, while the Application is validated before a Passport is created. The Passport class contains details such as passport number, issue date, expiry date, and passport type, and includes operations for issuing, renewing, revoking, and cancelling passports. DocumentOfficer and IssuingOfficer inherit from Officer, specializing in document verification and passport issuance respectively. Overall, the diagram clearly shows relationships, multiplicities, and responsibilities among the classes involved in the passport application and issuance process.

Class Diagram(Advanced)



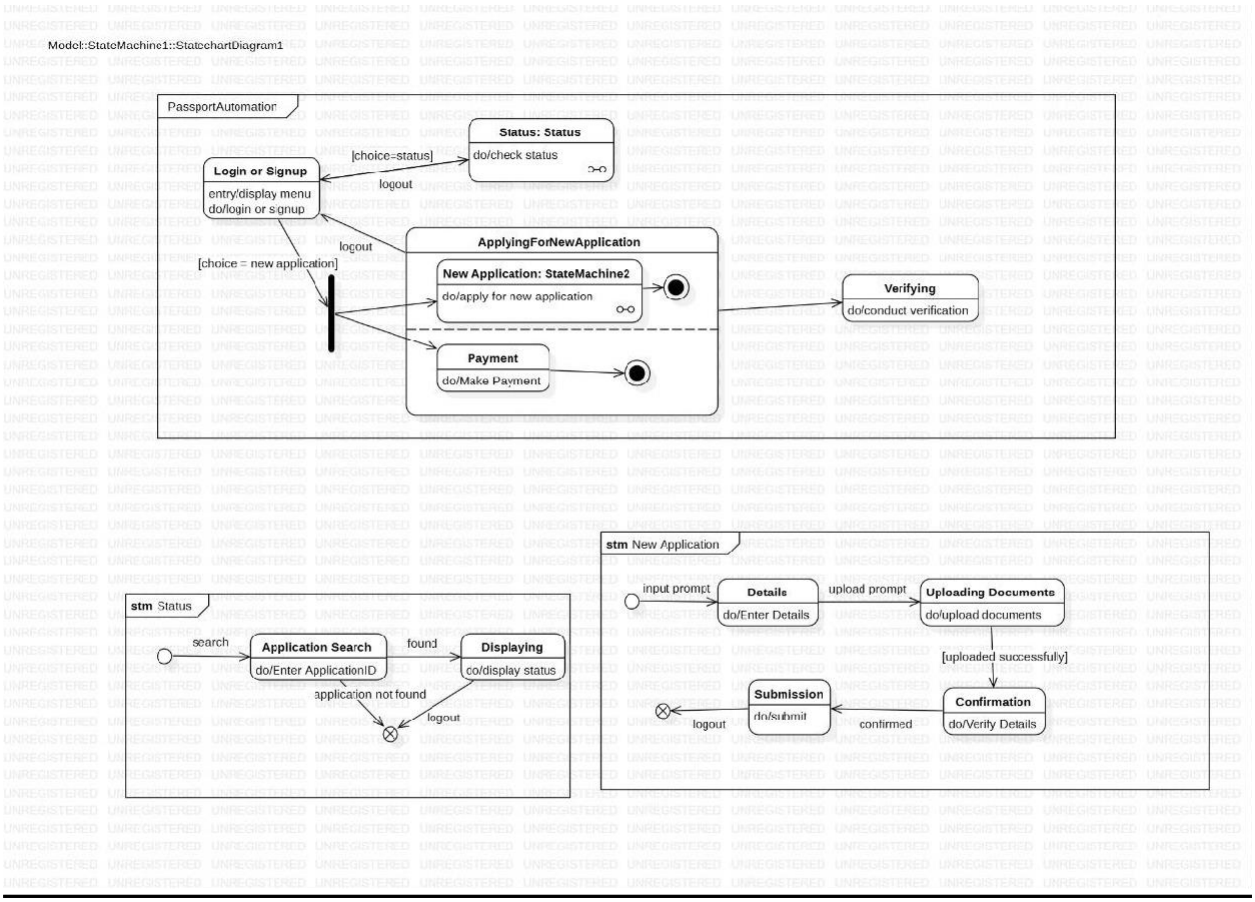
The UML class diagram models a passport processing system involving key entities such as PassportOffice, Applicant, Application, Appointment, Document, Payment, and various staff roles. The PassportOffice manages multiple staff members, including Officers and Clerks, each inheriting from the Staff class and performing duties such as approving applications or scheduling appointments. Applicants submit applications—either new or renewal—through the abstract Application class, which maintains submission details and application status using the ApplicationStatus enumeration. Each application is linked to an Applicant, associated Documents for verification, and corresponding Payment records. Appointments are scheduled and confirmed through the PassportOffice and can be cancelled or updated as needed. The system also links successful applications to the Passport entity, which further relates to the VisaPage component for travel validation. Overall, the diagram clearly captures class relationships, inheritance structures, and operational responsibilities within a complete passport management workflow.

State Transition Diagram(Simple)



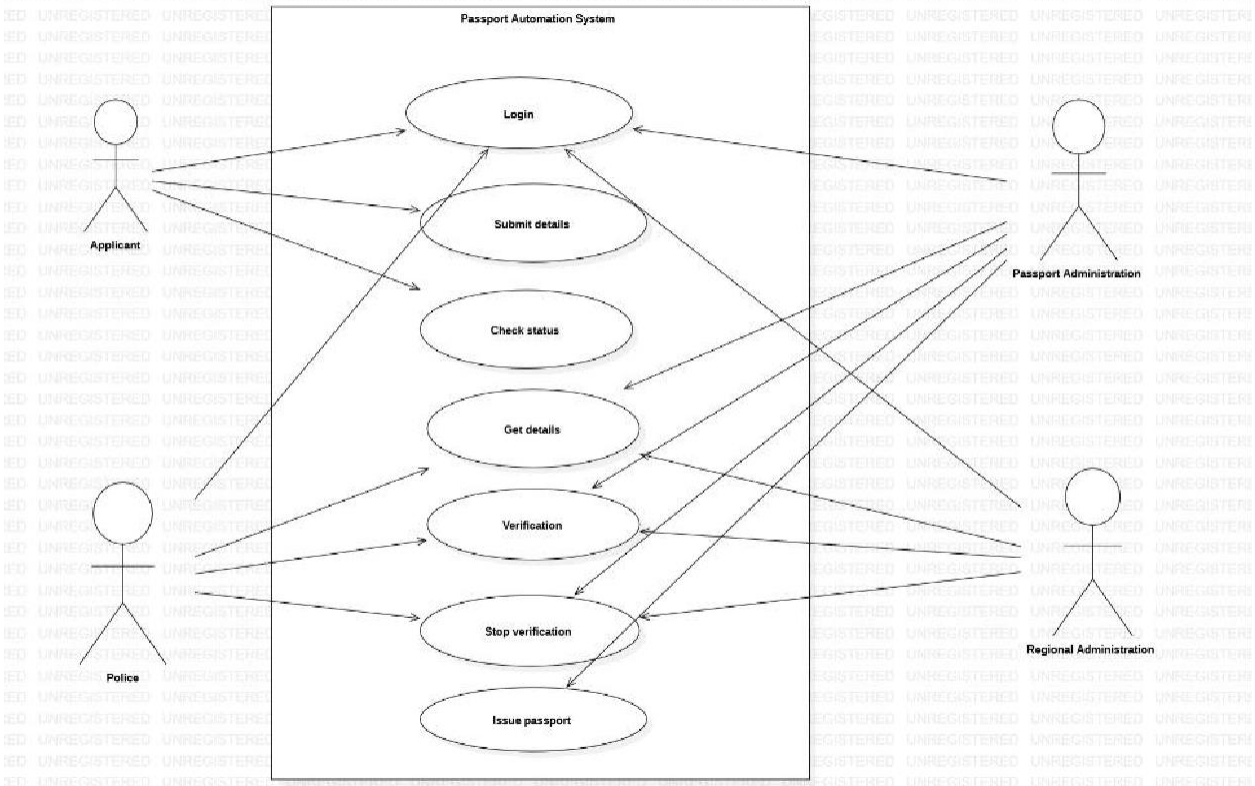
The statechart diagram illustrates the end-to-end workflow of passport authentication, starting with the *Application_Received* state where the system collects and verifies the submitted form data. If the form is valid, the process moves to *Verification* for police verification; invalid applications transition directly to *Rejected*. Once verification is cleared, the applicant enters the *Payment_Pending* state, where payment is processed before the system automatically proceeds. After successful payment, an appointment is *Scheduled*, followed by the *Interview* state on the designated interview day. Based on interview results, applicants either move to *Approved*— where the passport is printed—or return to *Rejected* if they fail. An approved passport enters the *Dispatched* state for courier delivery, and upon delivery confirmation the workflow ends in the *Completed* state, where the application is archived. This diagram effectively captures decision points, transitions, and actions involved in the entire passport authentication lifecycle.

State Transition Diagram(Advanced)



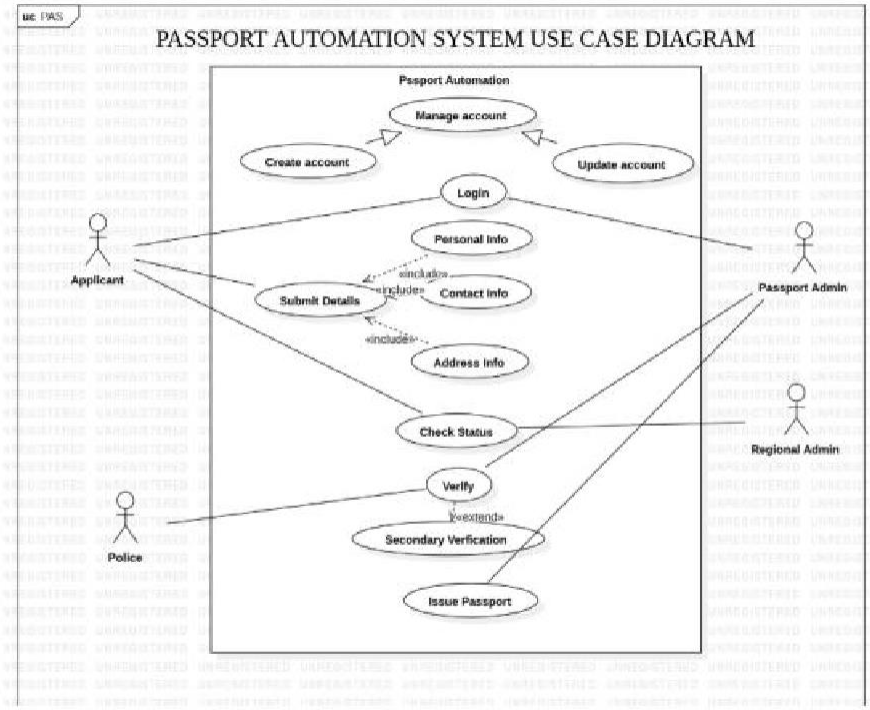
The statechart diagram represents the overall workflow of the Passport Automation system, beginning with the user logging in or signing up, after which they choose either to check application status or apply for a new passport. If the user selects status checking, the system transitions into the *Status* sub-state machine, where the application ID is entered, searched, and the current status is displayed. If the user opts to apply for a new passport, control moves into the *ApplyingForNewApplication* composite state, which contains another detailed state machine for completing a new application. This sub-process includes entering personal details, uploading required documents, submitting the application, and receiving confirmation after verification. Once the application is submitted, the workflow proceeds to payment, followed by a transition to the *Verifying* state where official verification is conducted. Throughout the diagram, various transitions support logout options and error handling, illustrating a complete, structured flow from user login to application submission and verification in the passport application process.

Use Case Diagram(Simple)



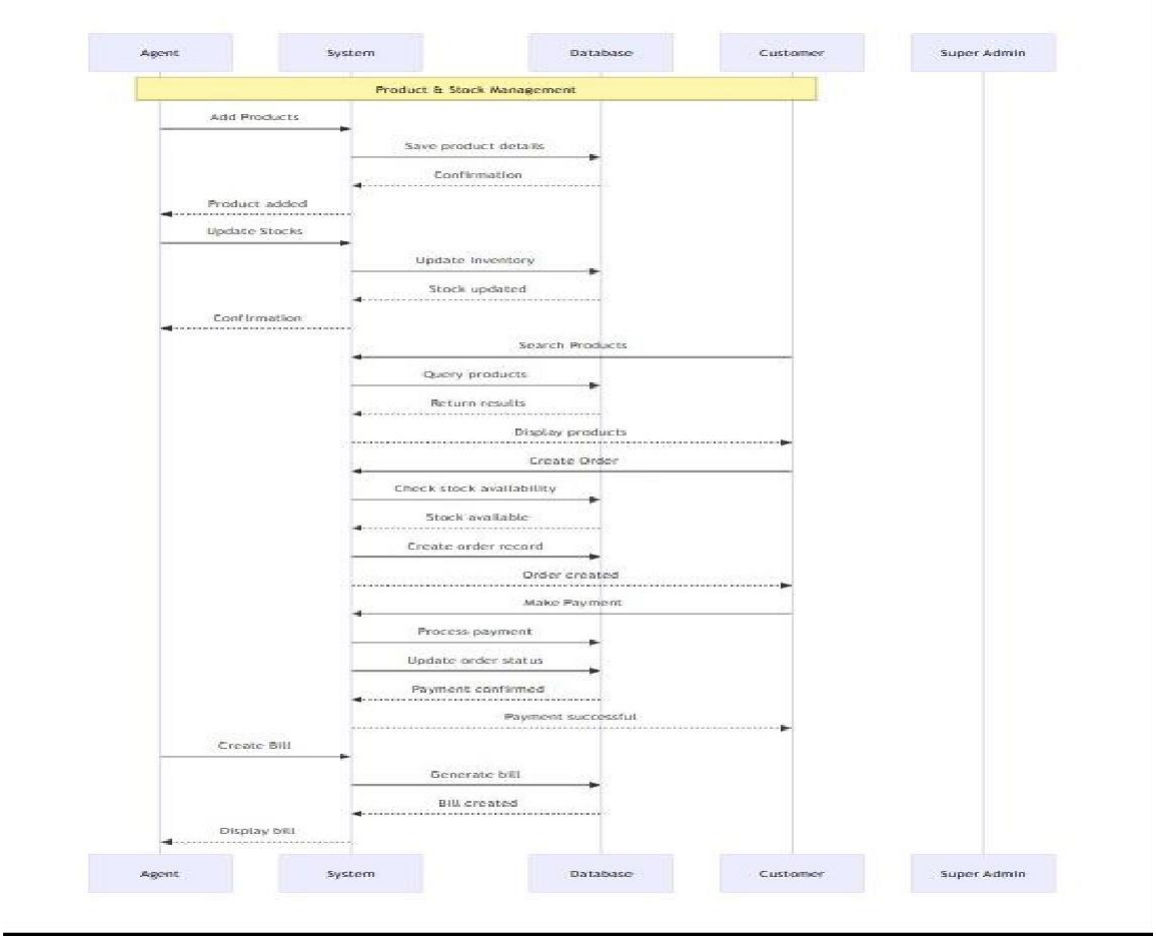
The use-case diagram illustrates the core interactions within the Passport Automation System, showing how different actors—including the Applicant, Passport Administration, Police, and Regional Administration—participate in various system functions. The Applicant initiates the process by logging in, submitting personal details, checking application status, and retrieving information. The Passport Administration interacts with the system to review submitted data, verify details, and proceed with issuing the passport, while the Police department conducts background verification and communicates verification results. The Regional Administration steps in for higher-level approval or escalation when needed. Together, these actors collaborate through use cases such as verification, stopping verification, and issuing passports, forming a complete digital workflow for streamlined passport processing. The diagram effectively highlights how each actor contributes to the overall lifecycle of a passport application.

Use Case(Advanced)



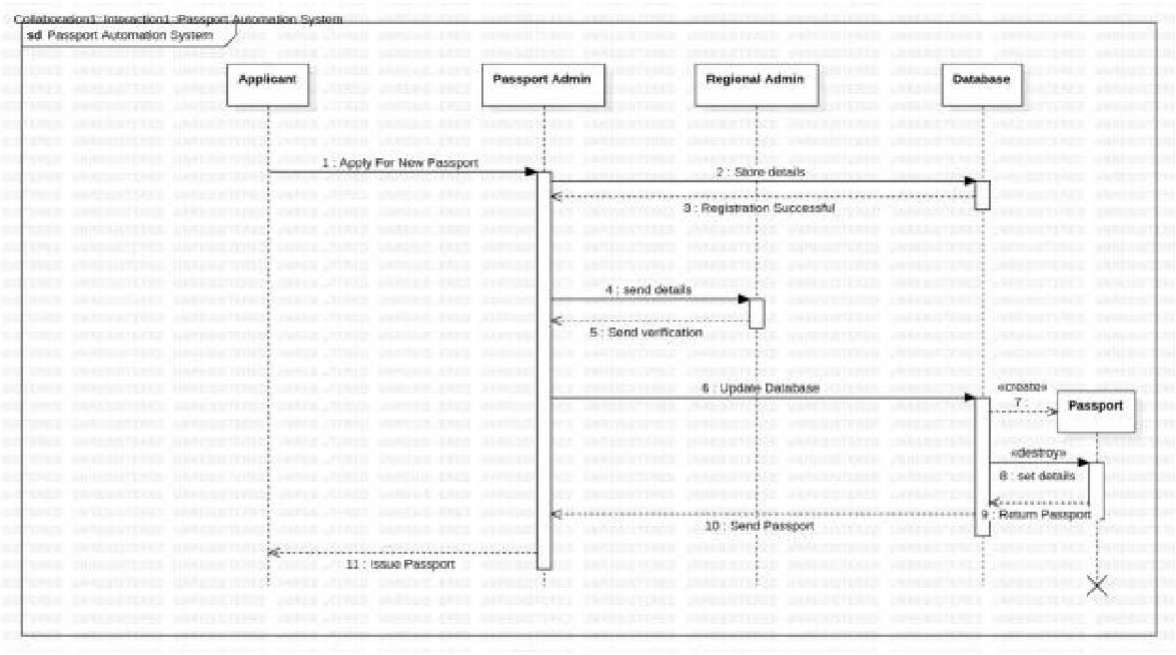
The use-case diagram for the Passport Automation System illustrates how different actors interact with the system to complete the passport application and verification process. The Applicant performs essential functions such as creating an account, logging in, submitting personal, contact, and address details, and checking the application status. These information submission actions are connected through *include* relationships, indicating that multiple details are required as part of a complete application. The Passport Admin manages account updates, verifies submitted information, performs secondary verification when needed, and ultimately issues the passport. The Police department plays a role in the verification process, contributing to identity and background checks that extend from the main verification use case. The Regional Admin participates in higher-level verification or approval processes to ensure authenticity and compliance. Together, these actors and use cases form a comprehensive workflow that supports efficient and secure passport processing within the automated system.

Sequence Model(Simple)



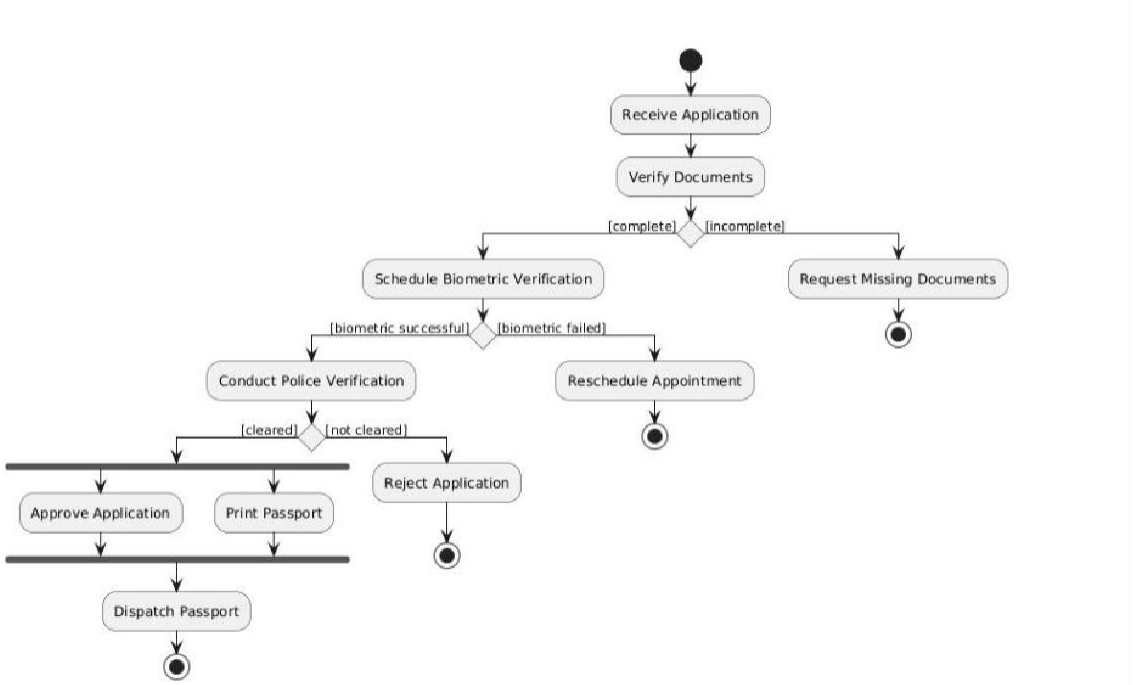
The sequence diagram illustrates the interaction between the Agent, System, Database, and Customer within the Product & Stock Management workflow. The process begins when an Agent adds new products, prompting the system to save product details in the database and return a confirmation. The Agent can also update stock levels, after which the system updates the inventory and acknowledges the change. On the customer side, a user searches for products, leading the system to query the database, retrieve the results, and display the products. When a customer creates an order, the system checks stock availability, confirms product availability, and records the order in the database. The customer then makes a payment, which the system processes by updating the order status and confirming successful payment. Finally, the Agent generates a bill, the system creates it using stored order information, and the bill is displayed, completing the transaction flow. This diagram clearly outlines the step-by-step communication and data exchange required for efficient product management and order processing.

Sequence Diagram(Advanced)



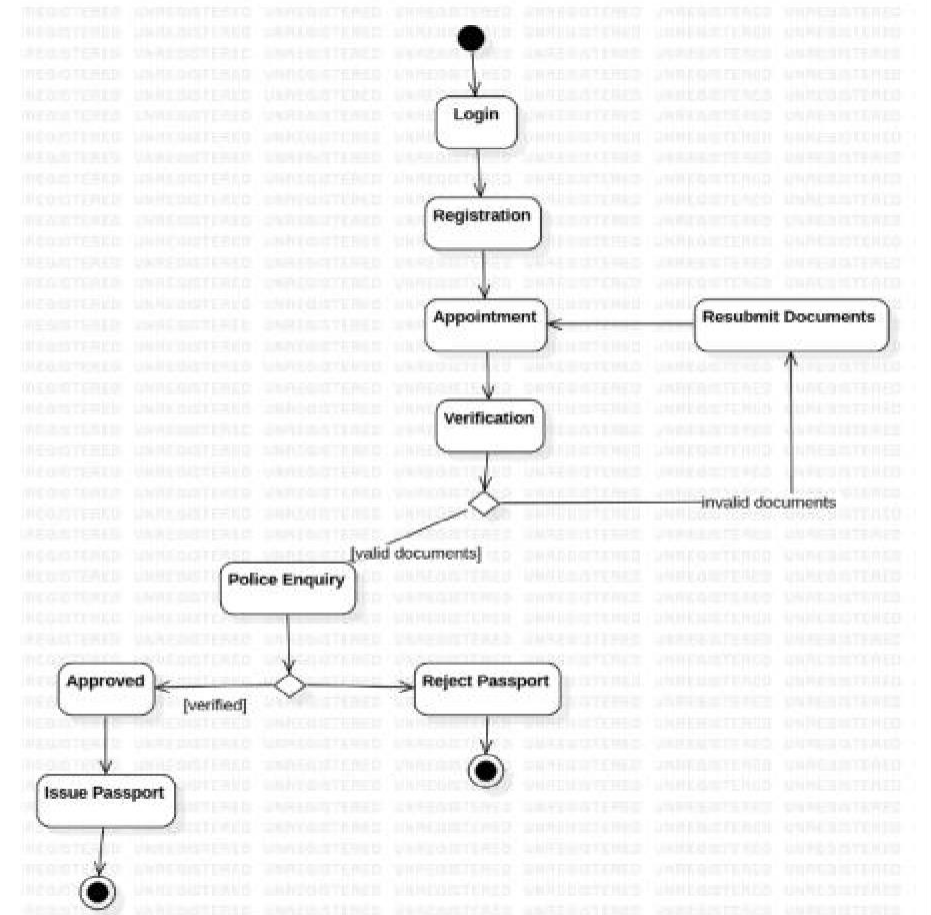
The sequence diagram illustrates the workflow of the Passport Automation System, showing how the Applicant, Passport Admin, Regional Admin, and Database interact during a new passport application. The process begins when the Applicant submits a request to apply for a passport, prompting the Passport Admin to store the application details in the database, which confirms successful registration. The Passport Admin then forwards the applicant's details to the Regional Admin for verification, after which verification results are sent back to the admin. Following verification, the admin updates the database, which triggers the creation of a Passport object, sets the relevant passport details, and then deletes the temporary object after returning the finalized passport information. Finally, the Passport Admin sends the completed passport back to the Applicant, completing the application process. This diagram clearly demonstrates step-by-step communication between system entities to ensure accurate passport processing and issuance.

Activity Diagram(Simple)



The activity diagram illustrates the complete workflow of the passport approval process, beginning with receiving the application and verifying the submitted documents. If the documents are incomplete, the system immediately requests the missing documents and ends the process. For complete applications, a biometric verification appointment is scheduled. Successful biometric verification leads to police verification, whereas failed biometric results trigger an appointment reschedule. After police verification, applications that are cleared proceed to simultaneous approval and passport printing activities, which together lead to dispatching the passport to the applicant. If police verification is not cleared, the application is rejected. Overall, the diagram clearly demonstrates decision points, parallel processes, exception handling, and the structured flow from document verification to final passport dispatch.

Activity Diagram(Advanced)



The activity diagram represents the sequential workflow of the passport application process, beginning with the user logging in and completing registration before scheduling an appointment for verification. During the verification stage, the application is assessed for valid documentation; if documents are invalid, the applicant is redirected to resubmit the required documents and return to the appointment stage. For valid documents, the process proceeds to a police enquiry, where background verification is conducted. If the enquiry is successful, the application moves to the approval stage and the passport is issued; if verification fails, the application is rejected. This diagram clearly depicts all decision points, alternations, feedback loops, and final outcomes involved in the passport approval lifecycle.