**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:*

**Sindhuja Narasimhan**

1BM22CS279

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
Oct 24-Jan 2025

# B.M.S. COLLEGE OF ENGINEERING

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

## *CERTIFICATE*

This is to certify that the Object-Oriented Modeling(23CS5PC00M) laboratory has been carried out by Sindhuja Narasimhan (1BM22CS279) during the 5th Semester Oct24-Jan2025.

Signature of the Faculty Incharge:

Prameetha Pai
Assistant Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

Table of Contents

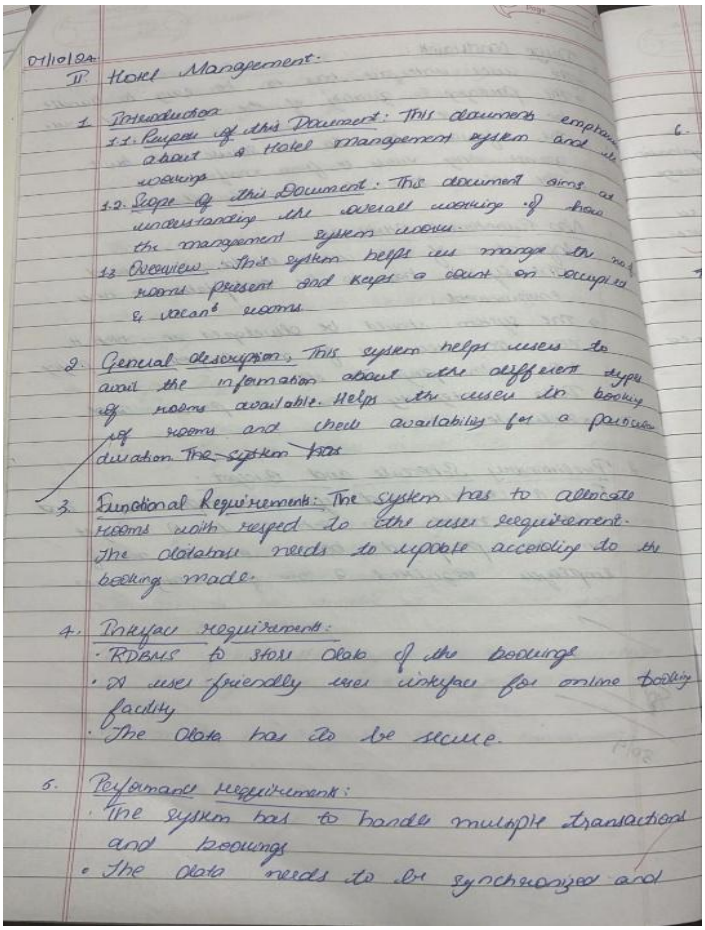
# 1. Hotel Management System

**Problem Statement:**

Managing hotel operations efficiently is a complex task due to the diverse range of activities involved, such as room reservations, guest check-ins and check-outs, staff coordination, inventory management, and billing. Many hotels still rely on manual processes or outdated systems, which can lead to inefficiencies, human errors, and poor customer experiences.

The lack of a centralized, user-friendly system makes it challenging to manage guest bookings, track room availability, handle billing, and maintain seamless communication between departments. Additionally, there is often no integration with modern features like online bookings, digital payments, or data analytics, which are essential for staying competitive in the hospitality industry

**SRS-Software Requirements Specification:**



07/10/24
IV Hotel Management:

1. Introduction
1.1. Purpose of this Document: This document emphasizes about a Hotel management system and its working
1.2. Scope of this Document: This document aims at understanding the overall working of hotel management system.
1.3. Overview: This system helps in manage the no. rooms present and keeps a count on occupied & vacant rooms

2. General description: This system helps users to avail the information about the different types of rooms available. Helps the users in booking of rooms and check availability for a particular duration. The system has

3. Functional Requirements: The system has to allocate rooms with respect to the user requirement. The database needs to update according to the booking made.

4. Interface requirements:
- RDBMS to store data of the bookings
- A user friendly user interface for online booking facility
- The data has to be secure.

5. Performance requirements:
- The system has to handle multiple transactions and bookings
- The data needs to be synchronized and

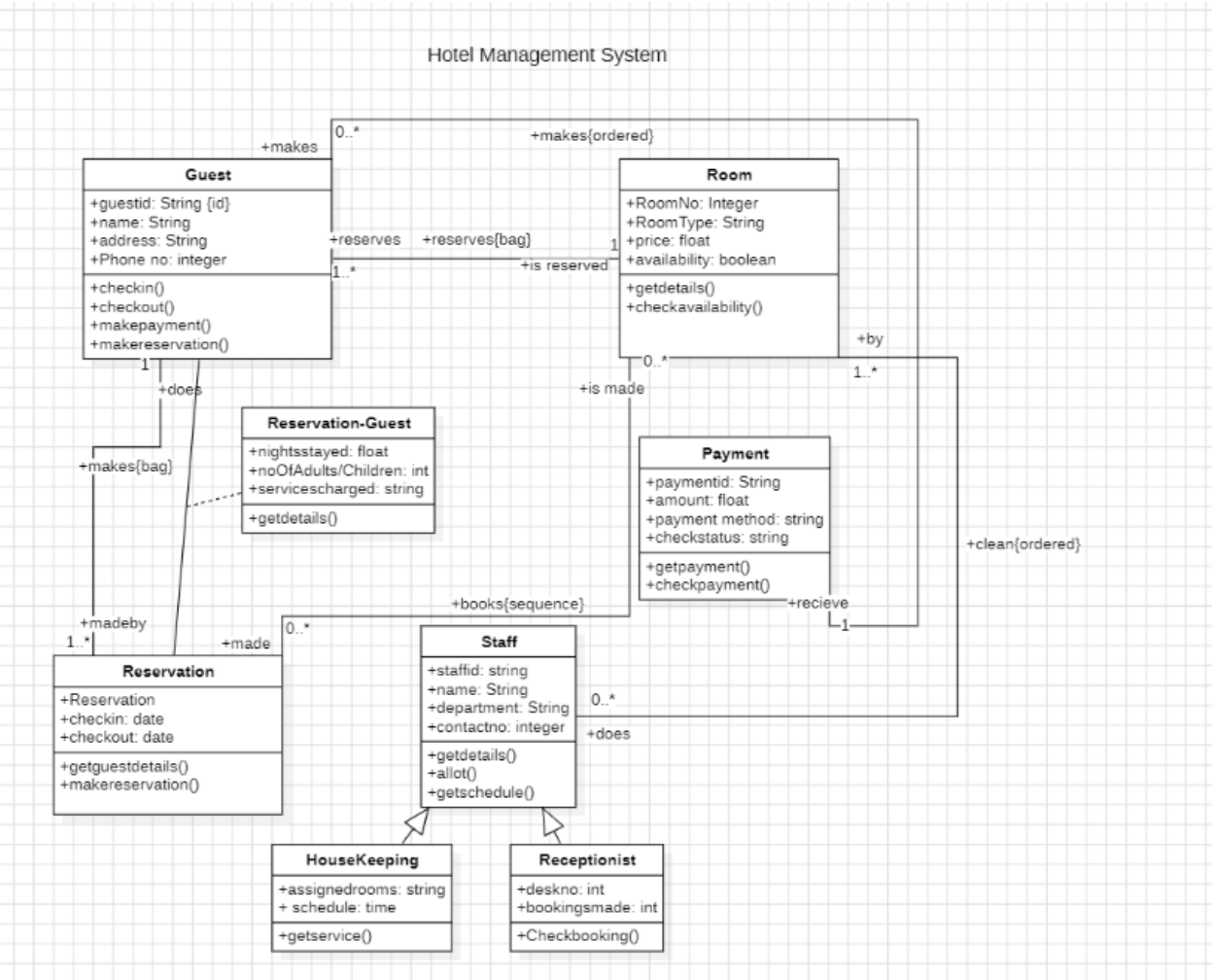
previous bookings needs to recorded.

6. Design Constraints:
-> The software should be easy to use.
-> Transactions to be done with help of Online transaction.
-> Software needs to be secure against any viruses.

7. Non-Functional Requirements:
-> The hotel staff members room allotment has to be recorded
-> Hygiene of the rooms to be maintained.

8. Preliminary Schedule and Budget:
The budget has to cover the charges for maintainance of the rooms, hotel staff salary and security.

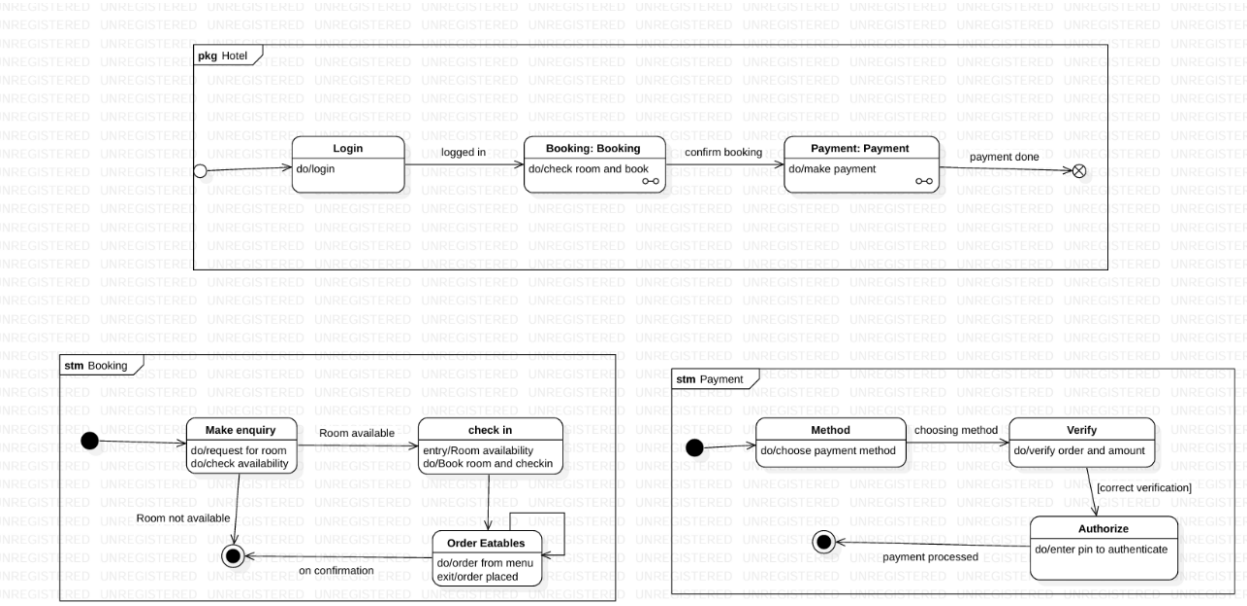
**Class Diagram:**




- Guest: Represents the hotel guests with attributes like ID, name, address, and phone number. Guests can make reservations, check in, check out, and make payments.
- Room: Holds details about rooms, such as room number, type, price, and availability. Guests can reserve rooms, and staff ensures they are cleaned and ready for use.
- Reservation: Tracks booking details, including check-in and check-out dates. It connects guests to the rooms they book and manages the reservation process.
- Reservation-Guest: Links reservation-specific details like the number of nights stayed, adults/children, and service charges. Helps associate personal details with the reservation.
- Payment: Manages payment information such as ID, amount, method, and status. Guests make payments for reservations, and the system tracks the transactions.

- Staff: Includes details of hotel staff such as ID, name, department, and contact info. Staff members perform tasks like room assignments, cleaning, and managing schedules.
- Housekeeping: Part of the staff, responsible for room cleaning and maintenance. Tracks assigned rooms and schedules for efficient service.
- Receptionist: Handles guest interactions, manages bookings, and assigns rooms. Tracks desk operations and the number of bookings made.
- Relationships: Guests make reservations and payments; staff ensures room readiness and assists with guest services. All components interact to manage hotel operations effectively.

**State Diagram:**




- LoginProcess:

  The user logs into the hotel management system using the provided credentials. Once logged in, the system allows access to booking and payment functionalities.

- BookingProcess:

  The user checks room availability and confirms the booking if a suitable room is available. Upon confirmation, the system proceeds to finalize the reservation details.

- PaymentProcess:

  The user chooses a payment method, verifies the order amount, and authenticates using a

PIN or similar mechanism. After successful verification, the payment is processed, completing the transaction.

- MakeEnquiry:
  Users can request room availability and check the hotel's capacity. If rooms are unavailable, the process stops, otherwise, they can proceed to booking.

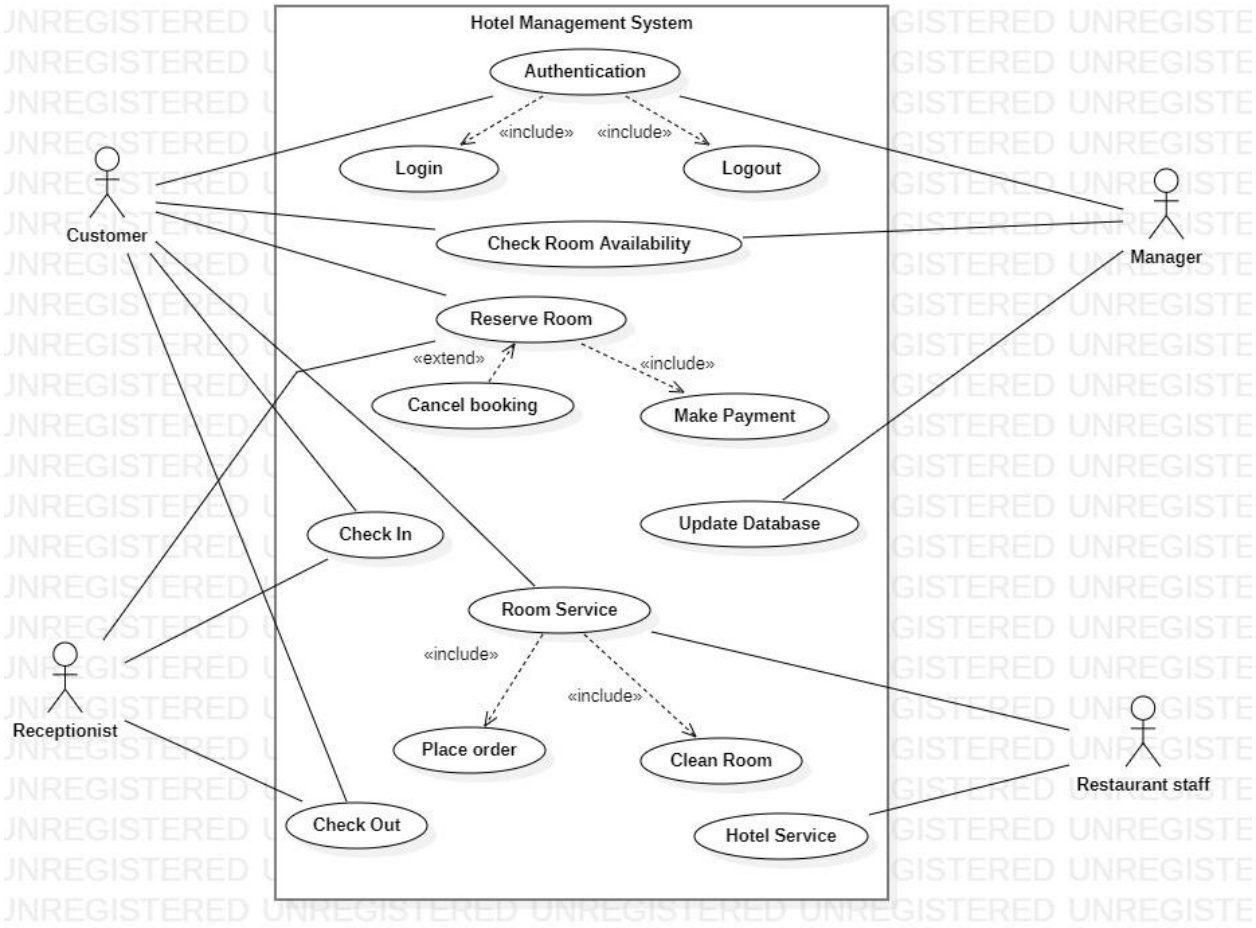
- Check-inProcess:
  Once a room is available, the user books the room and checks in. This step finalizes the allocation of the room to the user.

- OrderExtras:
  Guests can place additional orders for services or food during their stay. Once the order is placed, the system tracks the details and updates the user's account.

- PaymentMethodSelection:
  Users select a preferred payment method for their booking or orders. The system guides them to verify the details and complete the transaction.

- PaymentVerificationandAuthorization:
  After selecting the method, the system verifies the payment details and prompts the user for authentication (e.g., entering a PIN). If authenticated correctly, the payment is successfully processed.

**Use Case Diagram**:

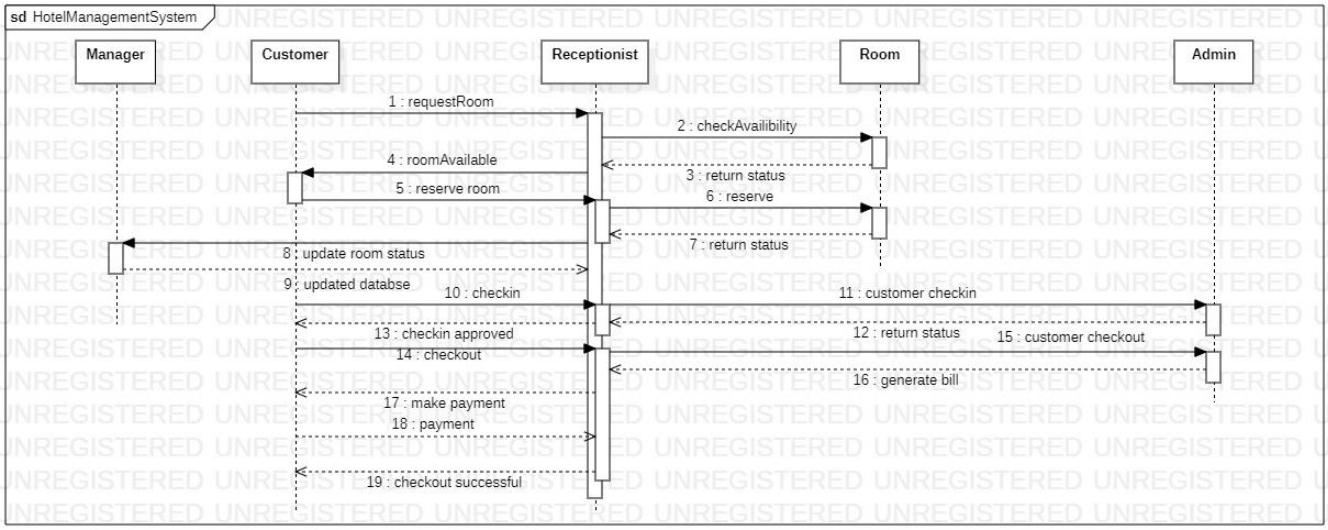



1. **Actors**:
    - Customer: Interacts with the system to log in, check room availability, reserve rooms, cancel bookings, make payments, and request room services.
    - Receptionist: Handles check-in, check-out, and assists with reservations and room services.
    - Manager: Updates the database and oversees the overall operations.
    - Restaurant Staff: Manages food orders placed by customers.
2. **Key Use Cases**:
    - Authentication: Allows users to log in and log out securely.

- Check Room Availability: Customers can search for available rooms before making a reservation.
- Reserve Room: Includes making payments and extends to cancel bookings if needed.
- Room Service: Includes placing food orders and requesting cleaning services.
- Check-in/Check-out: Handled by the receptionist for smooth customer management.

3. **Interactions**:
   - The customer initiates most activities like booking, payments, and services.
   - The receptionist and manager ensure seamless operations through system updates and service handling.
   - Restaurant staff and housekeeping handle specific customer service needs like food and cleaning.

**Sequence Diagram:**

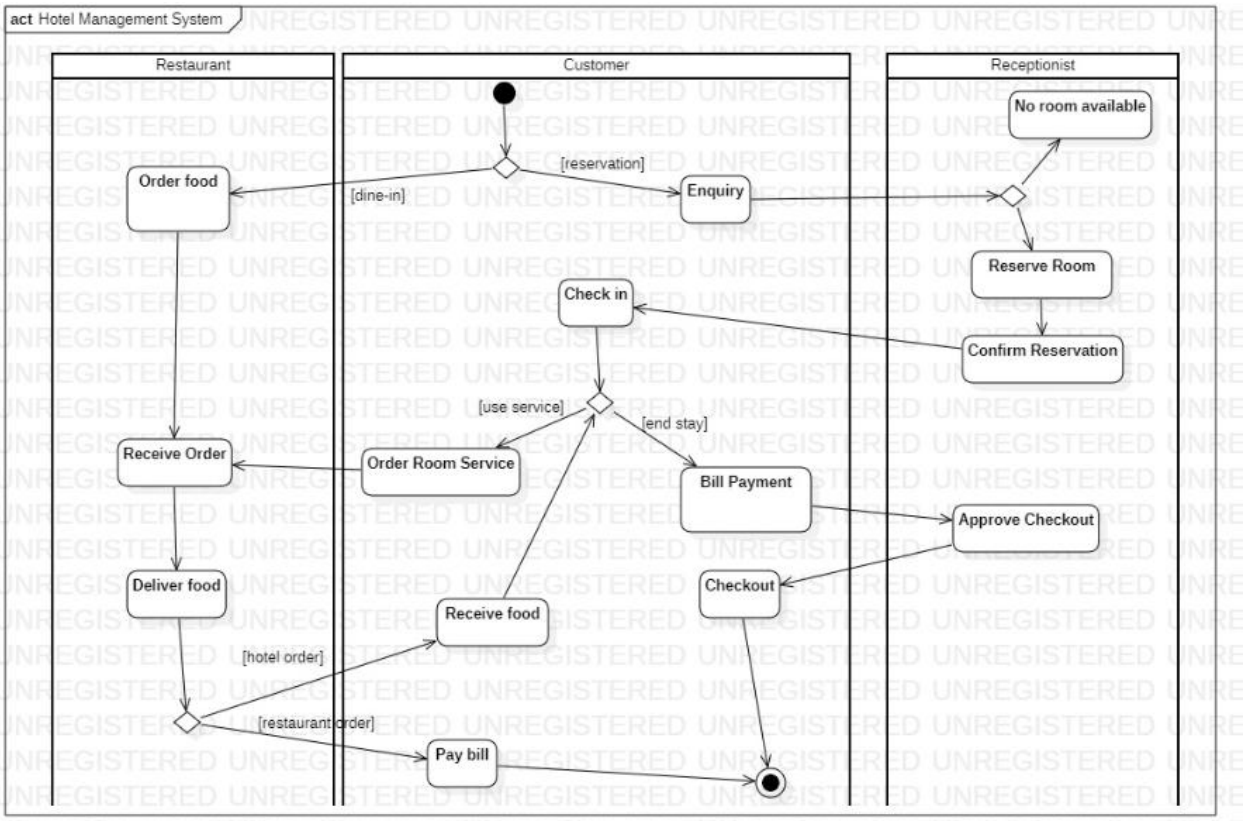



1. Actors:
   - Customer: Requests rooms, checks in, checks out, and makes payments.

- Receptionist: Facilitates room availability checks, reservations, customer check-ins, and check-outs.
- Manager: Oversees room status updates and ensures the database is kept current.
- Room: Represents the availability and status of hotel rooms.
- Admin: Handles billing and payment processes.

2. Key Interactions:
    - Request Room:
        - The customer requests a room, and the receptionist checks availability with the Room entity.
        - Once availability is confirmed, the customer reserves the room, and the system updates the room's status in the database.
    - Check-in Process:
        - The customer proceeds to check in, which is approved by the receptionist.
        - The Room status is updated to indicate that it is occupied.
    - Check-out and Payment:
        - The customer checks out, triggering the bill generation by the Admin.
        - The customer makes the payment, which is verified for successful checkout.
3. Workflow Summary:
    - The customer initiates key actions, such as booking, check-in, and check-out.
    - The receptionist coordinates between the customer and the Room/Admin for smooth operations.
    - The manager ensures accurate room status and database updates.
    - The Admin completes the payment and billing process to finalize the workflow.

**Activity Diagram:**




1. Actors:
    - Customer: Engages in enquiries, reservations, check-ins, room service requests, and bill payments.
    - Receptionist: Handles enquiries, room reservations, confirmations, and checkout approvals.
    - Restaurant: Manages food orders (dine-in or room service) and delivery.
2. Main Activities:
    - Room Enquiry and Reservation:
        - The customer makes an enquiry.
        - If a room is available, the receptionist reserves and confirms it; otherwise, the process ends.
    - Check-in and Stay:

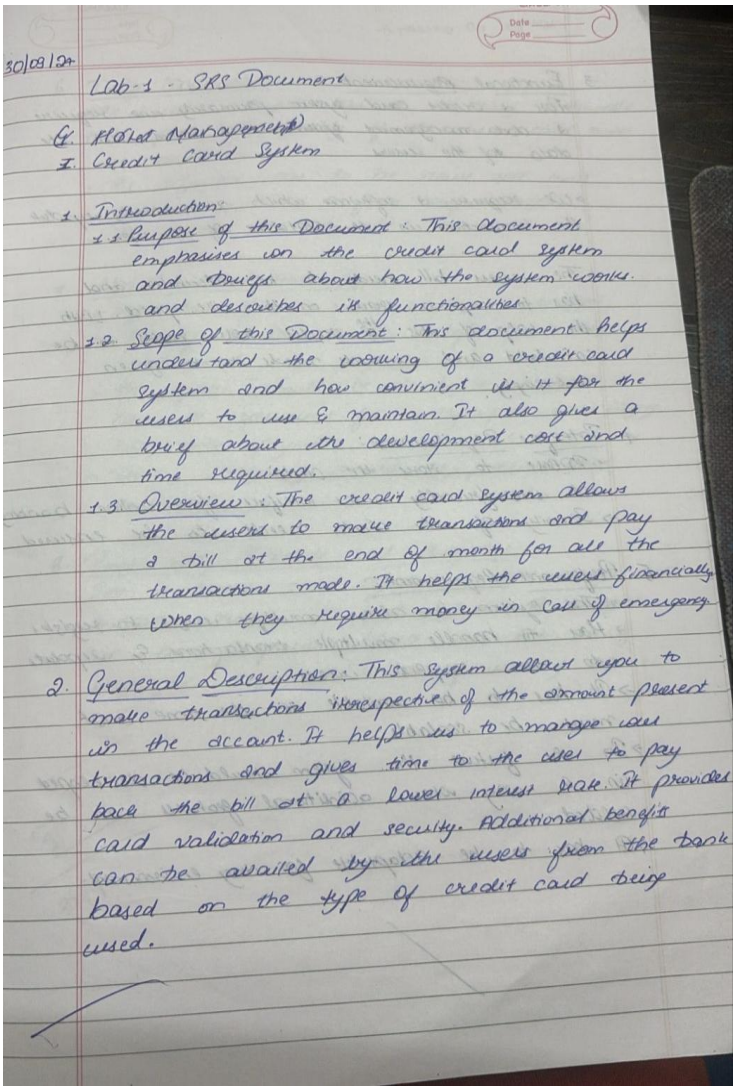
        - After confirmation, the customer checks in.
        - During their stay, the customer can request room service or order food.
    - Food Services:
        - For dine-in or room service, the restaurant receives the order and delivers food.
        - The customer pays for the food ordered.
    - Checkout and Payment:
        - At the end of the stay, the customer proceeds with bill payment and checkout.
        - The receptionist approves the checkout, completing the process.
3. Workflow Summary:
    - The customer initiates key processes, such as room enquiries, reservations, and service requests.
    - The receptionist ensures seamless room management and handles customer interactions.
    - The restaurant facilitates dining and room service needs.

# 2.Credit Card Processing

**Problem Statement**

The growing reliance on credit cards for transactions demands a robust, secure, and efficient system for processing payments. Traditional credit card processing systems often face challenges such as slow transaction speeds, security vulnerabilities, limited support for multiple payment methods, and difficulty in handling disputes or errors. These issues lead to customer dissatisfaction, financial losses, and non-compliance with industry standards like PCI DSS (Payment Card Industry Data Security Standard). Additionally, businesses need real-time insights into their transactions, seamless integration with existing accounting systems, and support for modern payment technologies like mobile wallets and contactless payments.

**SRS-Software Requirements Specification**



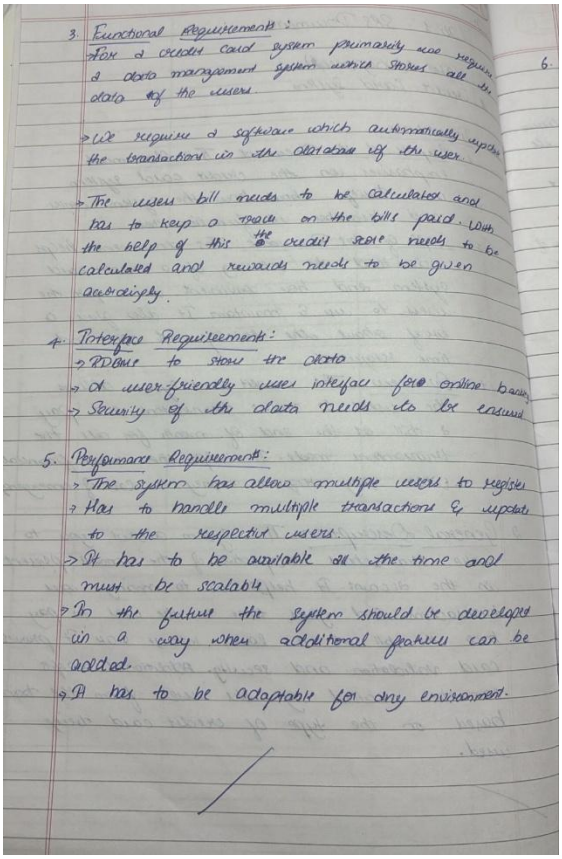
30/08/24

Lab-1 - SRS Document

G. Project Management
I. Credit Card System

1. Introduction:
1.1 Purpose of this Document: This document emphasises on the credit card system and throws about how the system works and describes it's functionalities
1.2 Scope of this Document: This document helps understand the working of a credit card system and how convinient it is for the users to use & maintain. It also gives a brief about the development cost and time required.
1.3 Overview: The credit card system allows the users to make transactions and pay a bill at the end of month for all the transaction made. It helps the user financially when they require money in call of emergency.

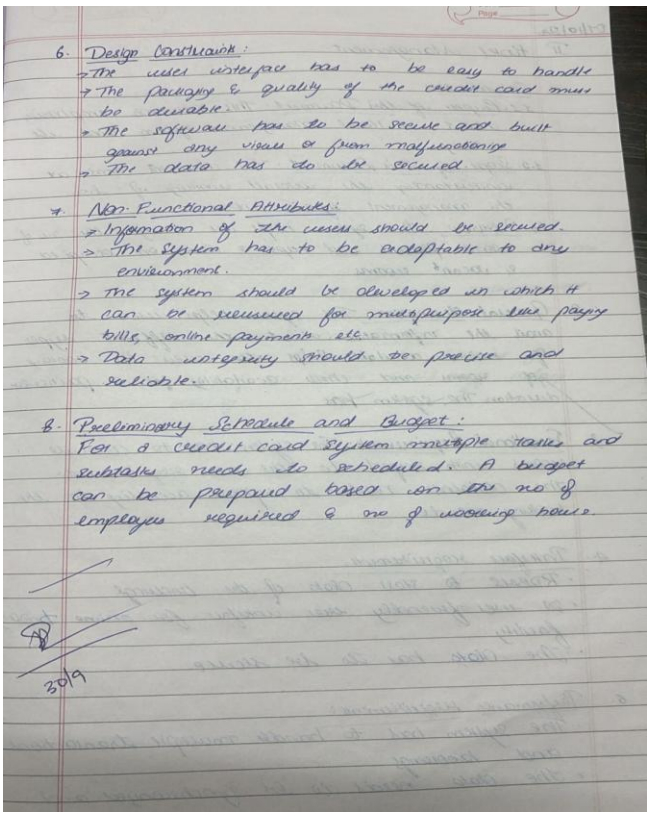
2. General Description: This system allows you to make transactions irrespective of the amount present in the account. It helps us to manage our transactions and gives time to the user to pay back the bill at a lower interest rate. It provides card validation and security. Additional benefits can be availed by the users from the bank based on the type of credit card being used.



3. Functional Requirements:
→ For a credit card system primarily we require a data management system which stores all the data of the users.
→ We require a software which automatically updates the transactions in the database of the user.
→ The users bill needs to be calculated and has to keep a track on the bills paid. With the help of this the credit score needs to be calculated and rewards needs to be given accordingly.

4. Interface Requirements:
→ RDBMS to store the data
→ A user-friendly user interface for online banking
→ Security of the data needs to be ensured

5. Performance Requirements:
→ The system has allow multiple users to register
→ Has to handle multiple transactions & updates to the respective users
→ It has to be available all the time and must be scalable
→ In the future the system should be developed in a way when additional features can be added.
→ It has to be adaptable for any environment.



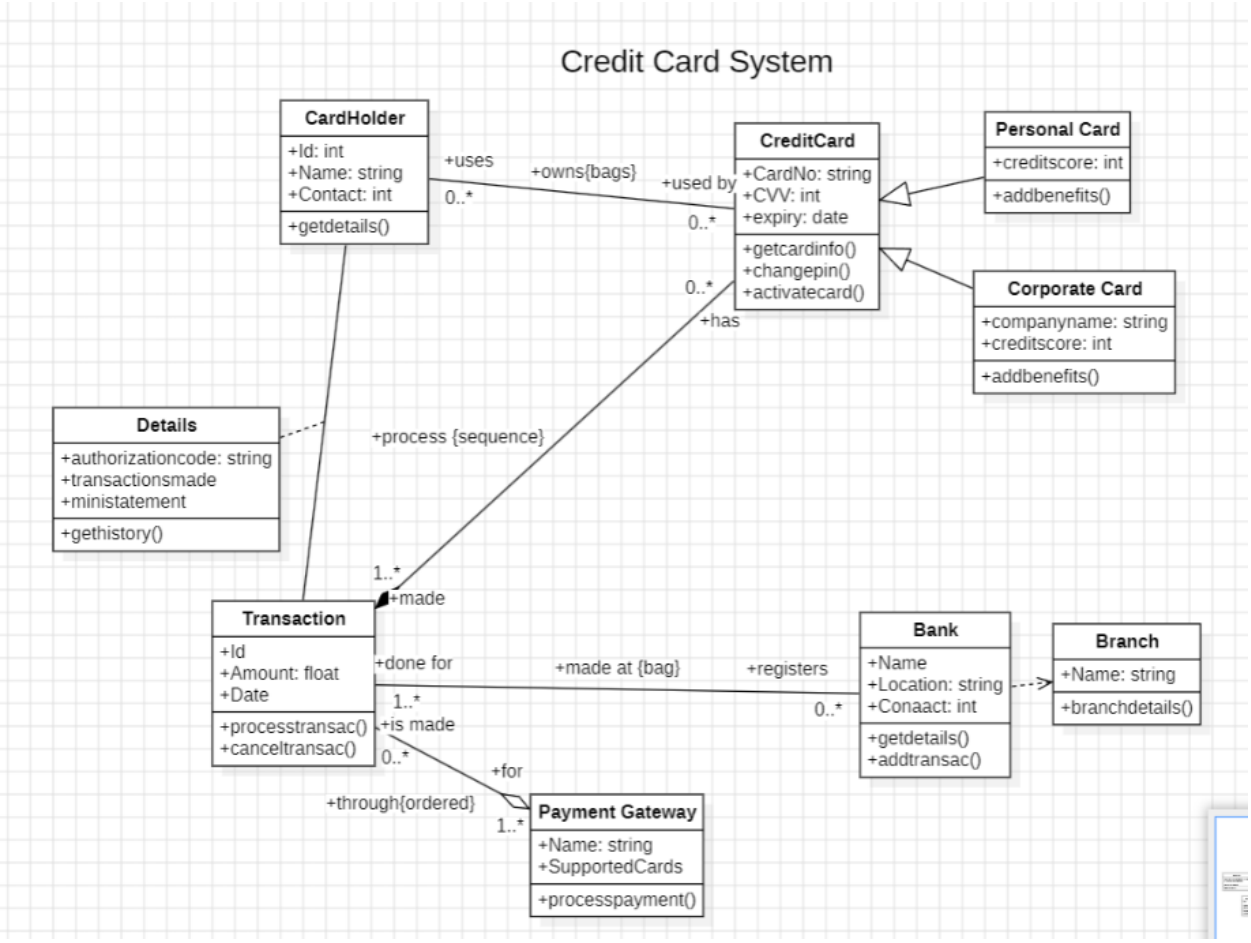
6. Design Constraints:
→ The user interface has to be easy to handle
→ The packaging & quality of the credit card must be durable.
→ The software has to be secure and built against any virus & from malfunctioning
→ The data has to be secured

7. Non-Functional Attributes:
→ Information of the users should be secured.
→ The system has to be adaptable to any environment.
→ The system should be developed in which it can be reassessed for multipurpose like paying bills, online payments etc.
→ Data integrity should be precise and reliable.

8. Preliminary Schedule and Budget:
For a credit card system multiple tasks and subtasks needs to scheduled. A budget can be prepared based on the no of employees required & no of working hours

3/9

**Class Diagram**

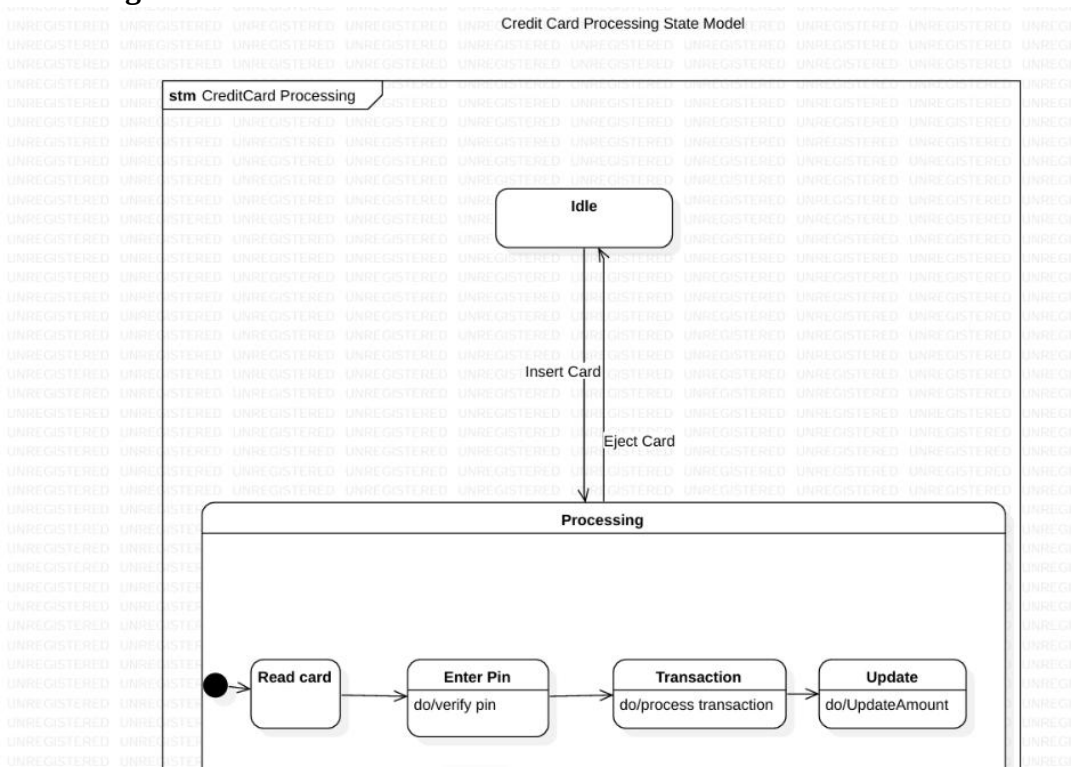



- Cardholder:
    - Represents individuals who own credit cards, with attributes like ID, name, address, phone number, and credit card limit.
    - Cardholders can perform activities like making purchases, paying bills, and checking account details.
- CreditCard:
    - Holds details about credit cards, including card number, card type, expiration date, CVV, and available credit limit.
    - Credit cards are issued to cardholders and are used for transactions and payments.
- Transaction:

- Tracks details of purchases or payments made using a credit card, such as transaction ID, date, amount, and merchant details.
- Transactions impact the available credit limit and are stored for tracking and statements.

- Payment:
  - Manages payment information, including payment ID, date, amount, and payment method.
  - Payments reduce outstanding balances and ensure the credit card remains active.
- Bank:
  - Represents the financial institution issuing the credit card, with attributes like bank ID, name, and contact details.
  - The bank oversees credit card issuance, processing payments, and monitoring accounts.
- RewardProgram:
  - Tracks details of reward points earned by cardholders through eligible transactions.
  - Points can be redeemed for rewards or discounts.
- FraudDetection:
  - Monitors unusual or unauthorized activities, with attributes like detection ID, suspicious transactions, and flags.
  - Helps in ensuring secure transactions and preventing fraudulent use of cards.
- Relationships:
  - Cardholders use credit cards for transactions, make payments to reduce balances, and interact with the bank for support.
  - Fraud detection and reward programs enhance the cardholder experience while ensuring security.

## State Diagram

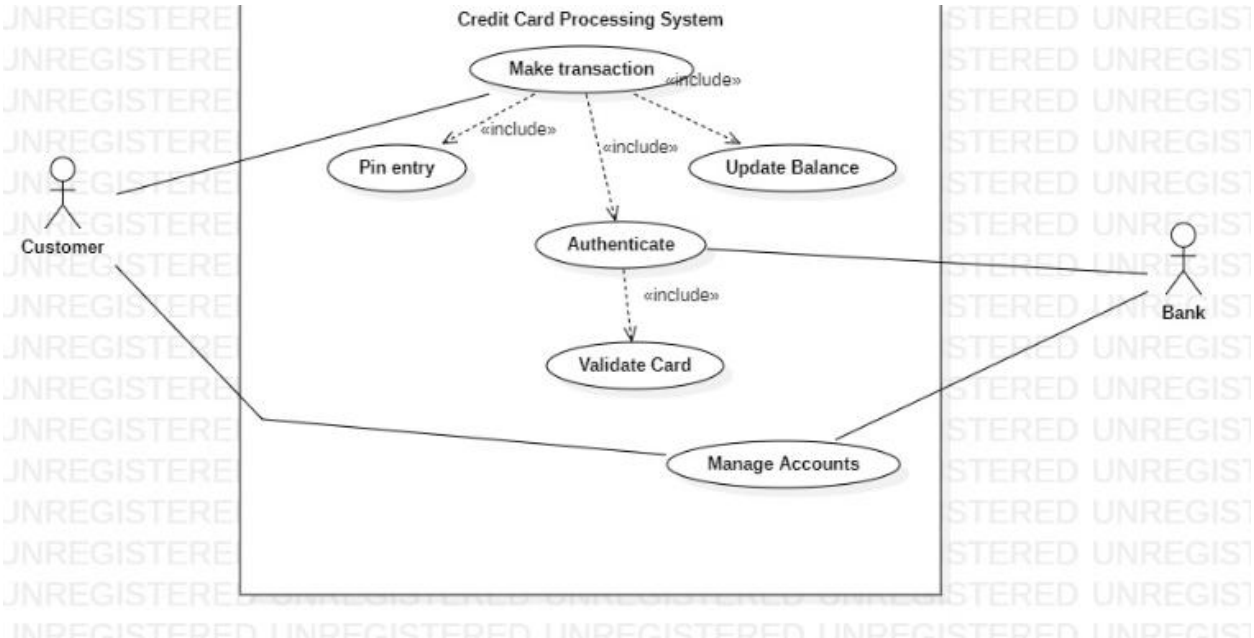



- Card Issuance Process:
    - The system verifies cardholder details and approves or rejects the application based on eligibility.
    - Upon approval, the card is issued and activated for use.
- Transaction Process:
    - Cardholders make purchases using their credit card, and the transaction details are recorded.
    - The system updates the available credit limit and tracks the transaction history.
- Payment Process:
    - Cardholders make payments to reduce their outstanding balance.
    - Payments are verified, processed, and reflected in the account statement.
- Fraud Detection Process:

- The system monitors transactions for unusual activity, such as high-value purchases or transactions in foreign locations.
- If fraud is detected, the system blocks the card and notifies the cardholder.

- Reward Redemption Process:
  - Cardholders check their reward points balance and select items for redemption.
  - The system processes the redemption and updates the reward points balance.

**Use Case Diagram**



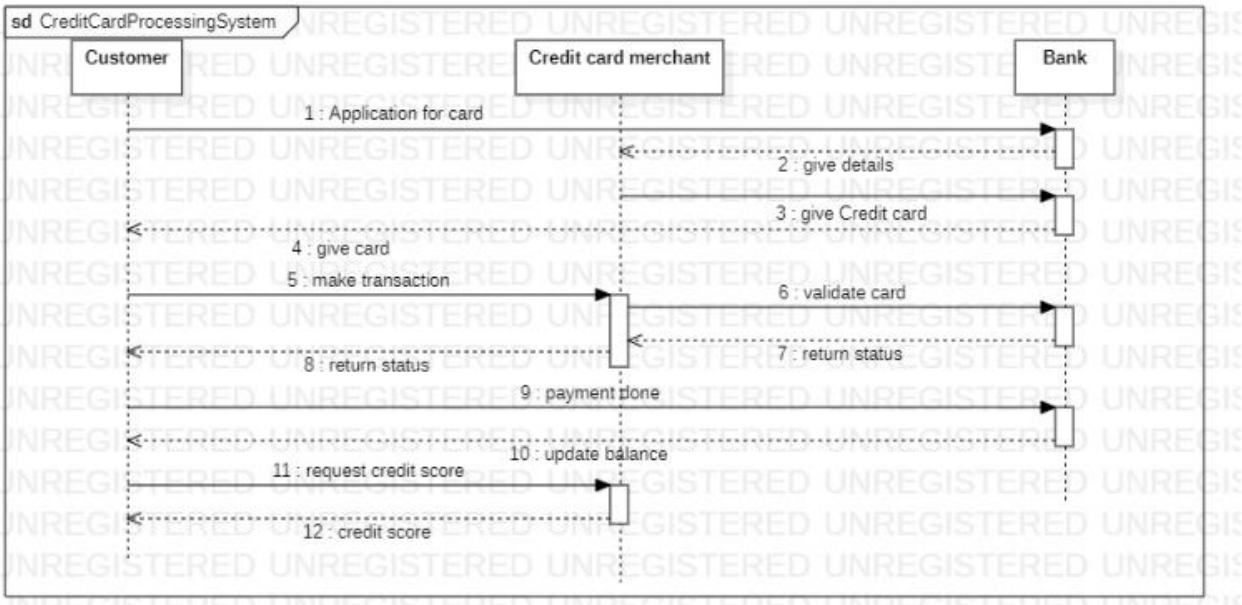

- Actors:
  - Cardholder: Uses the system to make transactions, pay bills, and manage rewards.
  - Bank: Issues credit cards, monitors transactions, and ensures compliance with regulations.
  - Merchant: Accepts credit card payments and communicates with the system to process transactions.
  - Fraud Analyst: Investigates flagged transactions and ensures secure operations.

- Key Use Cases:

- Apply for Credit Card: Allows users to submit applications and receive approval or rejection notifications.
- Make Transaction: Tracks purchases made using the credit card and updates the credit limit.
- Pay Bill: Enables cardholders to settle their credit card dues using various payment methods.
- Redeem Rewards: Provides an interface for cardholders to use reward points for benefits.
- Monitor Fraud: Tracks suspicious transactions and alerts the cardholder and bank.

- Interactions:
  - Cardholders initiate actions like purchases, payments, and reward redemptions.
  - Merchants process transactions and report them to the system.
  - The bank ensures all processes run smoothly and manages backend operations.

**Sequence Diagram**

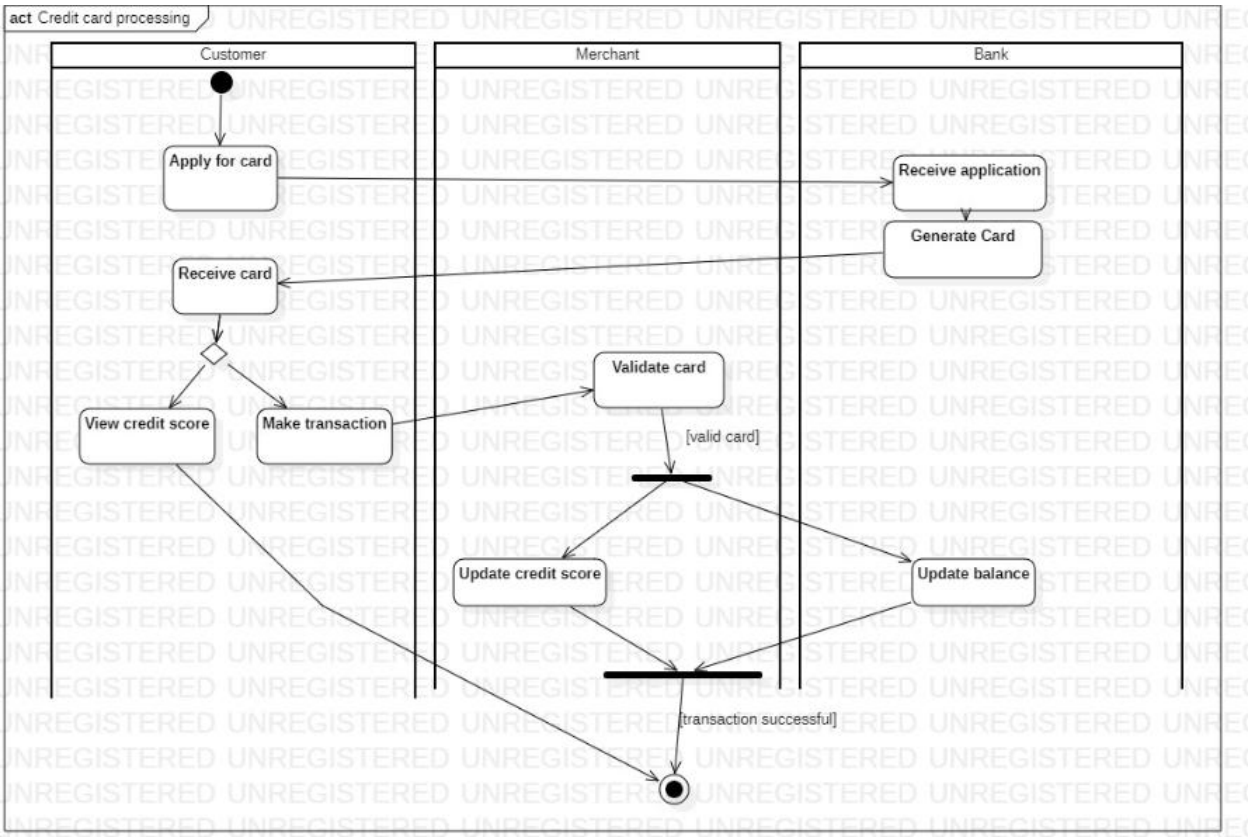



- Actors:
  - Cardholder: Initiates transactions, payments, and rewards redemption.

- Bank System: Verifies applications, processes transactions, and tracks fraud.
- Merchant: Communicates with the system to complete payment processing.
- Fraud Analyst: Manages flagged transactions for review.

- Key Interactions:
  - Apply for Credit Card:
    - The cardholder submits an application to the bank.
    - The bank reviews the application, approves/rejects it, and notifies the applicant.
  - Make Transaction:
    - The cardholder initiates a transaction, and the merchant communicates with the bank system.
    - The system verifies the card details, approves the transaction, and updates the credit limit.
  - Pay Bill:
    - The cardholder submits payment details, which are processed by the bank.
    - The system updates the outstanding balance and generates a confirmation.
  - Redeem Rewards:
    - The cardholder selects rewards for redemption, and the bank processes the request.

**Activity Diagram**




- Actors:
    - Cardholder: Engages in applying for credit cards, making purchases, paying bills, and redeeming rewards.
    - Bank: Handles credit card applications, transaction approvals, and fraud detection.
    - Merchant: Facilitates transactions by accepting payments.
- Main Activities:
    - Apply for Credit Card:
        - The cardholder submits their details through an application.
        - The bank reviews and processes the application, issuing the card if eligible.
    - Make Transaction:
        - The cardholder makes a purchase, and the merchant processes the payment through the bank.

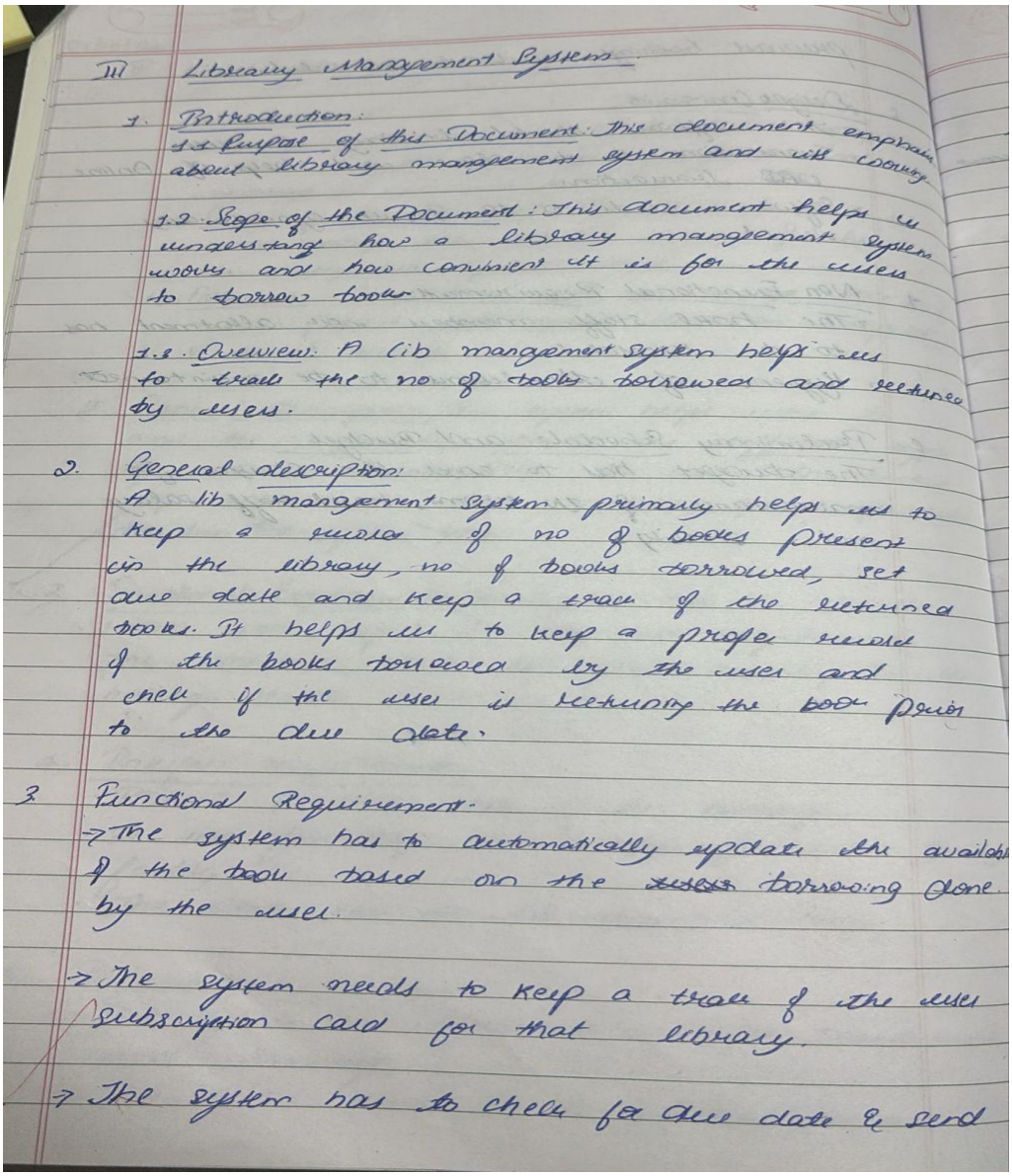
        - The transaction is recorded, and the available credit is updated.
    - Pay Bill:
        - The cardholder initiates a payment to reduce their outstanding balance.
        - The bank processes the payment and updates the account.
    - Fraud Detection:
        - The system monitors transactions in real-time to identify anomalies.
        - If fraud is detected, the system blocks the card and alerts the cardholder and bank.
    - Redeem Rewards:
        - Cardholders review their reward points balance and request redemption.
        - The bank approves the request, processes the redemption, and updates the points balance.
- Workflow Summary:
    - The cardholder drives activities like transactions, payments, and rewards redemption.
    - The bank ensures secure operations, approves or flags actions, and manages cardholder accounts.
    - Merchants process transactions in collaboration with the bank system, ensuring a smooth payment experience.

# 3.Library Management System

**Problem Statement**

Traditional library management systems often rely on manual processes, leading to inefficiencies in tracking books, managing member records, and handling transactions such as book borrowing and returns. These methods result in time-consuming tasks, data inaccuracies, limited accessibility for users, and challenges in generating meaningful insights for library staff. A lack of integration with digital technologies further hampers the library's ability to meet modern user expectations for convenience and efficiency**.**

**SRS-Software Requirements Specification**



III Library Management System

1. Introduction:
1.1 Purpose of this Document: This document emphasis about library mangement system and its working

1.2 Scope of the Document: This document helps us understand how a library mangement system works and how convinient it is for the users to borrow books

1.3. Overview: A Lib mangement system helps us to track the no of books borrowed and returned by users.

2. General description:
A lib mangement system primarily helps us to keep a record of no of books present in the library, no of books borrowed, set due date and keep a track of the returned books. It helps us to keep a proper record of the books borrowed by the user and check if the user is returning the book prior to the due date.

3 Functional Requirement:
→ The system has to automatically update the availabl of the book based on the ~~users~~ borrowing done by the user.

→ The system needs to keep a track of the user subscription card for that library.

→ The system has to check for due date & send

a notification

4. Interface Requirement:
- RDBMS to keep a track of the books borrowed & returned by users.
- Previous books borrowed by a user has to be stored in the memory
- The software has to show always the updated availability of the books to the users with the help of an online portal.

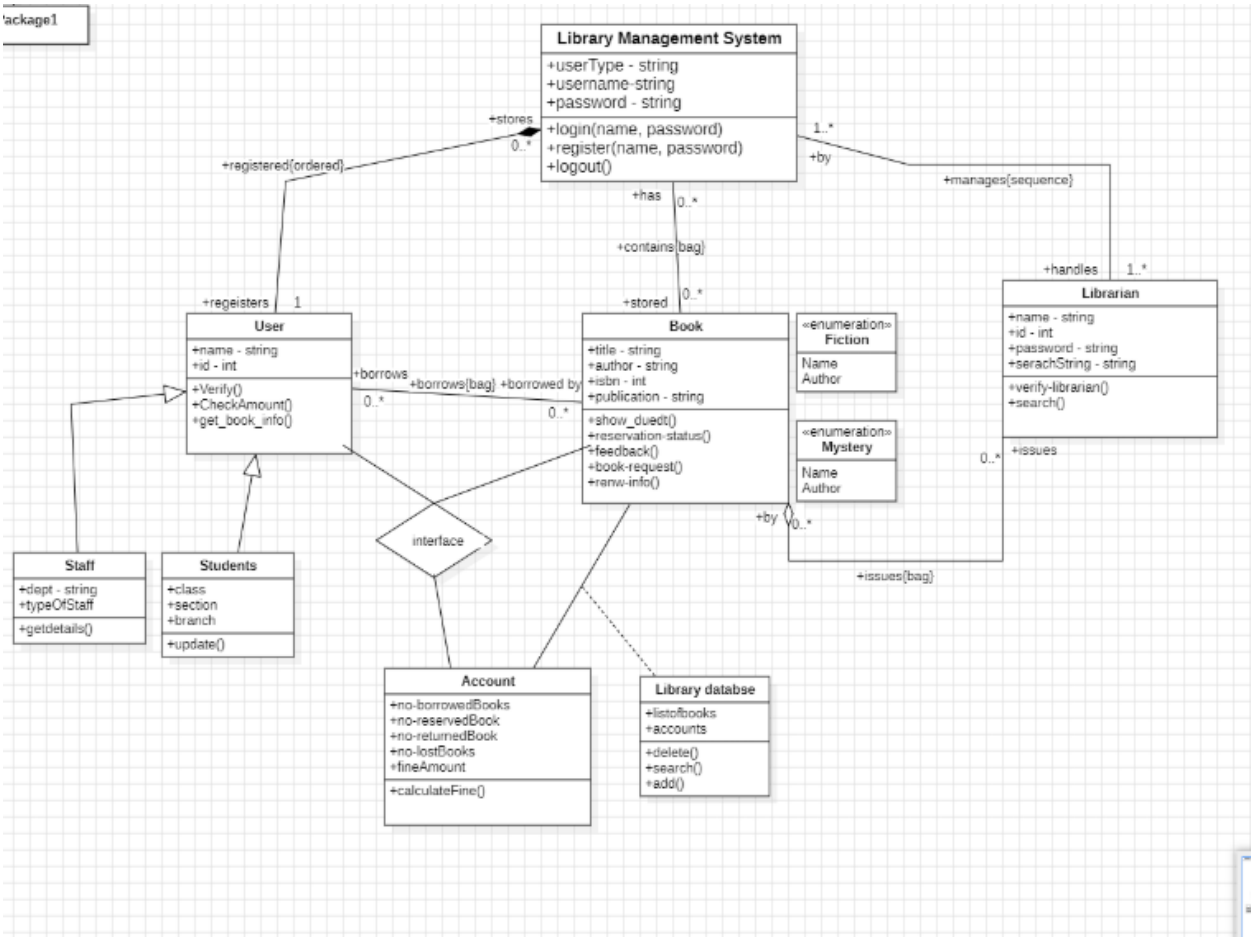
5. Performance Requirement:
- The system has to always be updated with the current availability of books.
- Has to keep a check of the the books returned and borrowed
- Users have to be notified about the due dates for the borrowed book.

6. Design Constraints:
- A systematic view of the different genres available in the library
- The librarian's interface has to be updated about the book borrowed by the user as well the availability of that book.

7. Non Functional attributes:
- The system needs to ensure that all books are returned and none goes missing.
- The data has to be reliable and secured.

8. Preliminary Schedule & Budget
- For a library management system multiple tasks (for eg borrowing, returning, updating) needs to be scheduled. A budget for no of books to be purchased & staffs required needs to be prepared.

## Class Diagram




- User:
    - Attributes: UserType (string), username (string), password (string).
    - Methods: Login (username, password), searchBooks (string), feedback (string).
- Book:
    - Attributes: Title, Author, ISBN, Availability.
    - Methods: CheckAvailability(), ReserveBook(), UpdateDetails().
- Librarian:
    - Attributes: ID, Name, ContactInfo.
    - Methods: ManageMemberships(), ApproveBookRequests(), UpdateDatabase().
- LibraryDatabase:

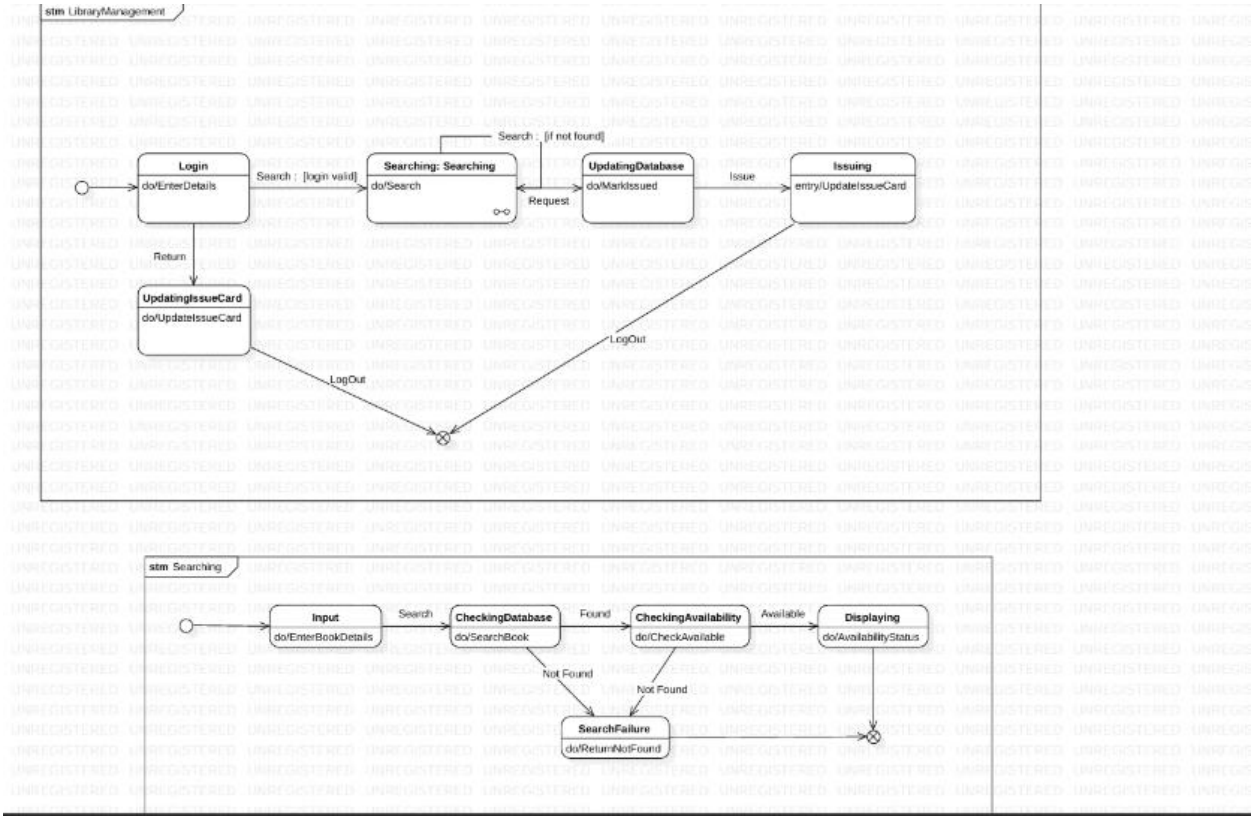
- Attributes: DatabaseID, BookRecords, UserRecords.
- Methods: SearchBooks(string), AddBook(Book), RemoveBook(Book).

- Transaction:
  - Attributes: TransactionID, BookIssued, BookReturned, FineAmount.
  - Methods: CalculateFine(), IssueBook(), ReturnBook().

- Relationships:
  - Users interact with the library database for searching books.
  - Librarians manage transactions and oversee user operations.

- Books are associated with transactions, and their availability status is updated in the database.

**State Diagram**

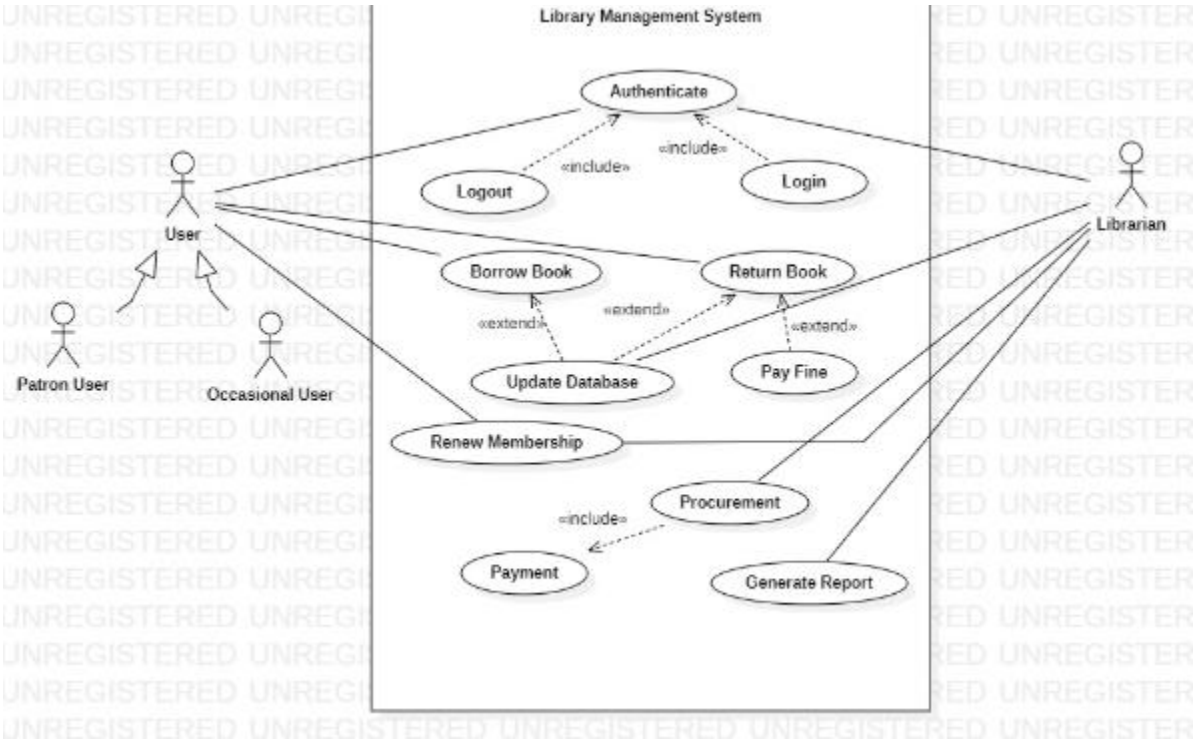



- Search Process:

- Start with user login.
- Enter search criteria.
- Check availability in the database.
- Display results (success or no match).

- Issue/Return Process:
  - Verify user and book details.
  - Update transaction status.
  - Calculate fine for overdue books (if any).

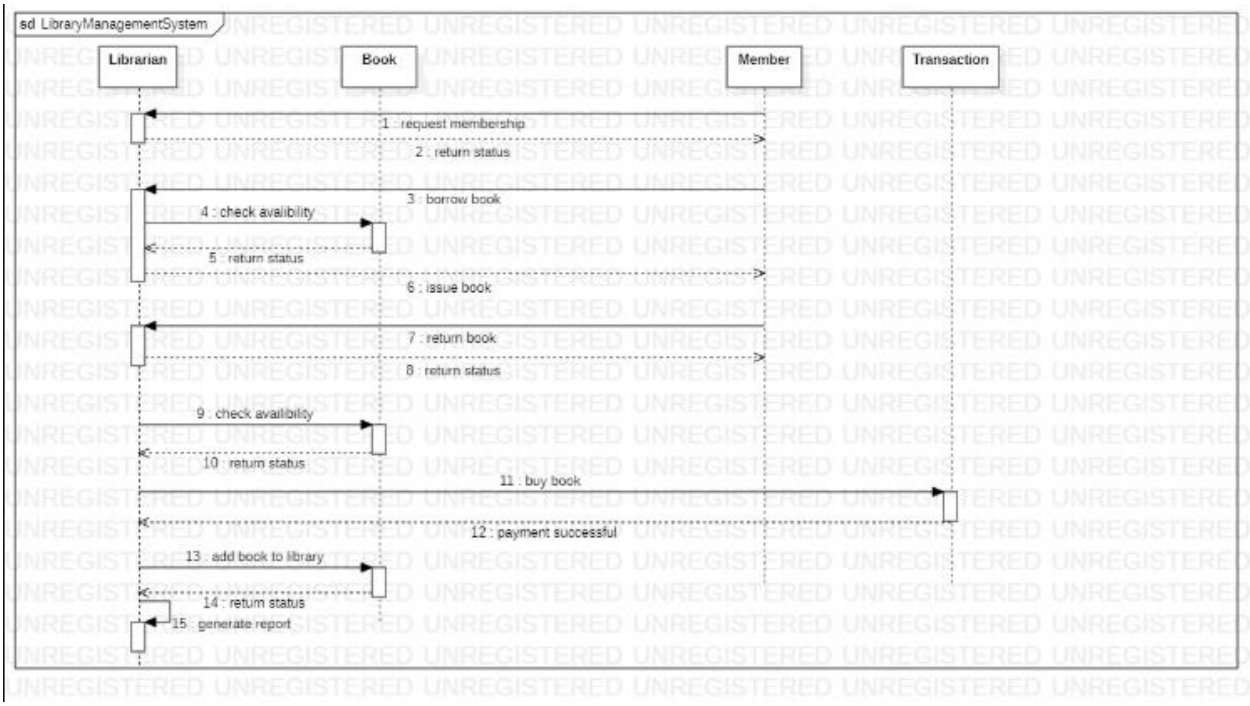
**Use Case Diagram**




- Actors:
  - User: Searches for books, borrows books, provides feedback.
  - Librarian: Approves membership, issues/returns books, manages the system.

- Key Use Cases:
    - Search for Books: Users search by title, author, or ISBN.
    - Borrow Book: Users request a book, and librarians approve the transaction.
    - Return Book: Users return the borrowed book, and fines (if any) are calculated.
    - Renew Membership: Librarians manage user renewals for continued access.
    - Manage Database: Librarians update book records, add new books, and remove old ones.

**Sequence Diagram**

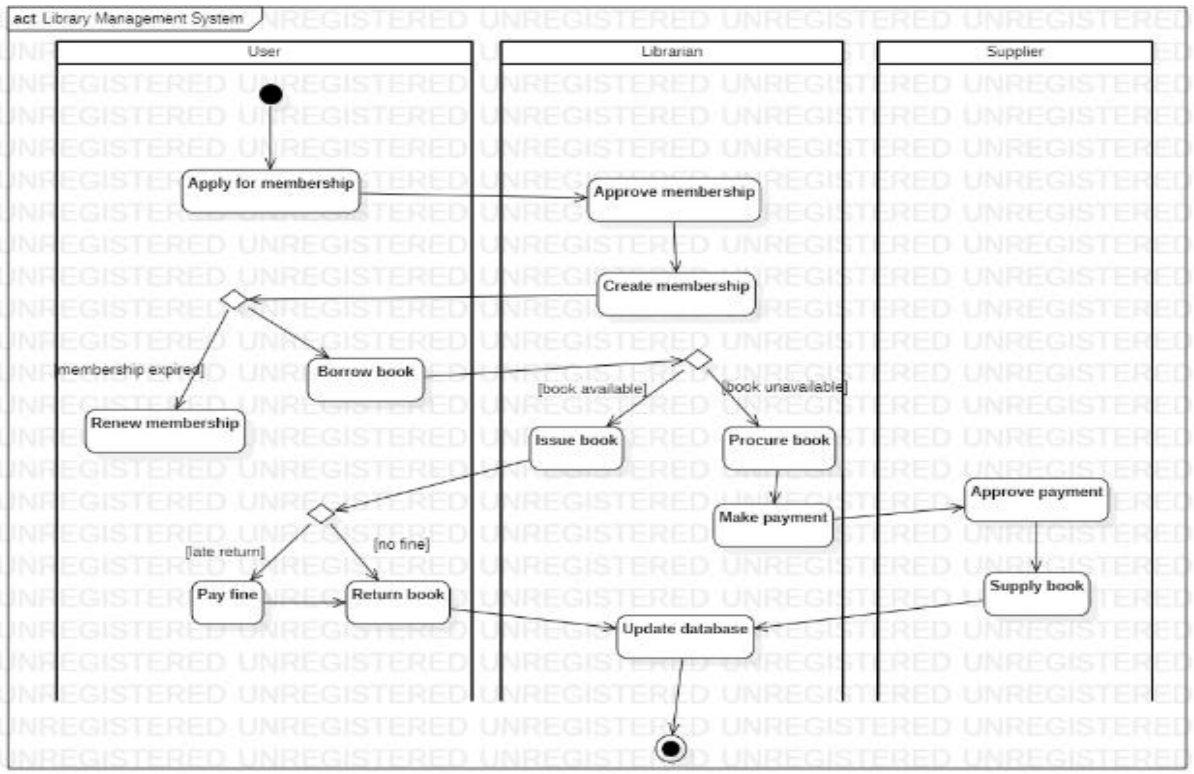



- Actors:
    - User: Applies for membership, searches for books, and manages loans.
    - Librarian: Verifies membership, issues books, and manages overdue fines.
    - Supplier: Supplies books requested by the library.
- Interactions:
    - Membership Request:

- The user applies for membership.
- The librarian verifies details and approves/rejects the request.

- Book Borrowing:
  - The user searches for a book.
  - The librarian checks availability, issues the book, and updates the system.
- Book Return:
  - The user returns the book.
  - The librarian processes the return, calculates fines (if any), and updates the system.
- Procurement Process:
  - The librarian requests unavailable books from suppliers.
  - The supplier processes the request and updates the library inventory.

**Activity Diagram**



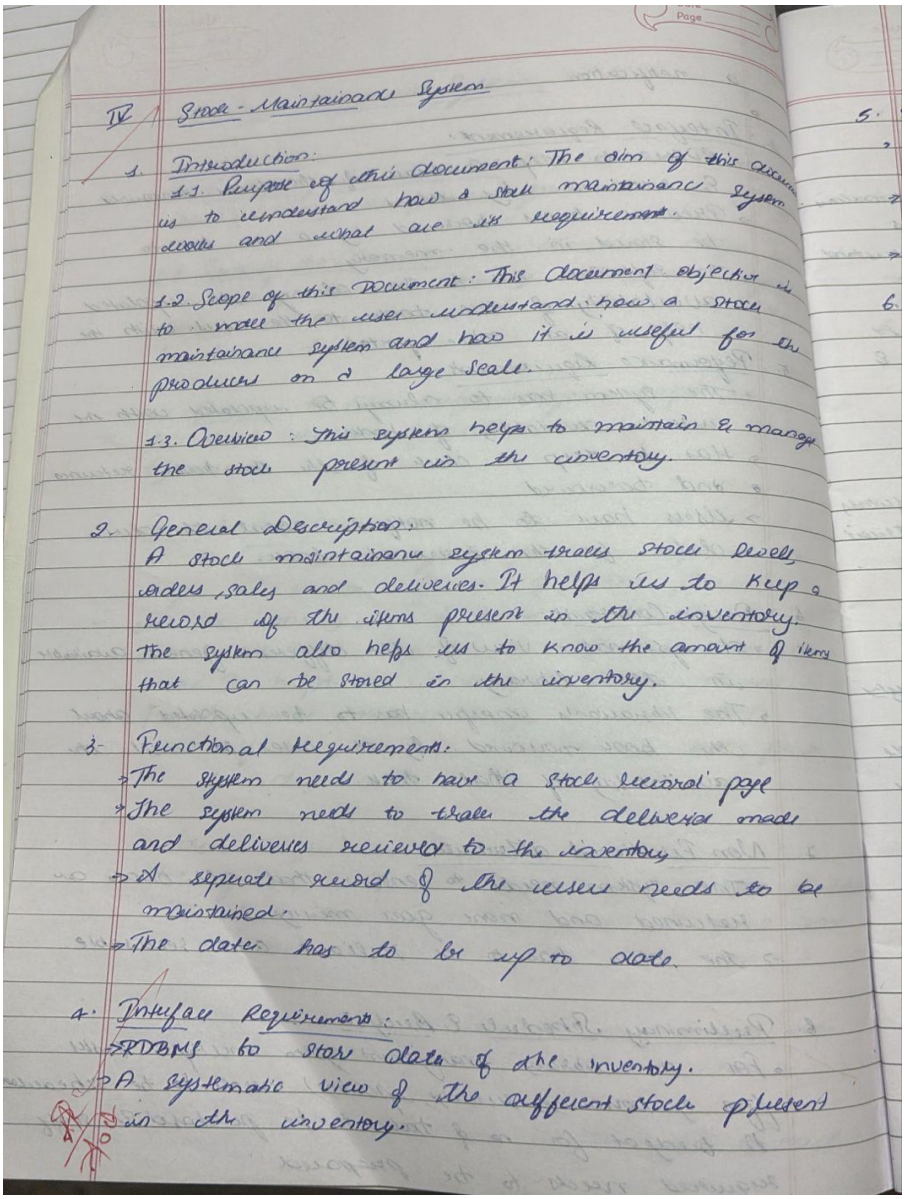
- Actors:
    - User: Initiates actions like applying for membership, borrowing/returning books, and making fine payments.
    - Librarian: Manages approvals, loans, and database updates.
    - Supplier: Provides unavailable books on request.
- Key Activities:
    - Membership Application:
        - The user fills out an application.
        - The librarian verifies details, approves/rejects the request, and updates the database.
    - Book Search and Borrowing:
        - The user searches for a book.
        - If available, the librarian issues it; otherwise, the librarian initiates a procurement request to the supplier.
    - Book Return:
        - The user returns a borrowed book.
        - The librarian checks for overdue fines, processes payment if needed, and updates the database.
    - Database Updates:
        - The librarian ensures that all transactions, loans, and book procurements are logged into the system.

# 4.Stock Maintainence System

**Problem Statement**

Effective stock management is critical for businesses to ensure optimal inventory levels, reduce wastage, and meet customer demands. However, traditional manual or semi-automated methods of stock maintenance often result in challenges such as inaccurate inventory tracking, overstocking or understocking, delayed replenishments, and inefficient reporting. These issues can lead to increased operational costs, missed sales opportunities, and poor decision-making due to a lack of real-time data and analytics.

**SRS-Software Requirements Specification**



IV Stock - Maintainance System

1. Introduction:
1.1 Purpose of this document: The aim of this document is to understand how a stock maintainance system works and what are its requirements.

1.2 Scope of this Document: This document objective is to make the user understand how a stock maintainance system and how it is useful for the producers on a large scale.

1.3. Overview: This system helps to maintain & manage the stock present in the inventory.

2. General Description:
A stock maintainance system tracks stock levels, orders, sales and deliveries. It helps us to keep a record of the items present in the inventory. The system also helps us to know the amount of items that can be stored in the inventory.

3. Functional Requirements:
→ The system needs to have a stock record page
→ The system needs to track the deliveries made and deliveries recieved to the inventory
→ A seperate record of the users needs to be maintained.
→ The data has to be up to date.

4. Interface Requirements:
→ RDBMS to store data of the inventory.
→ A systematic view of the different stock present in the inventory.

5.

6.

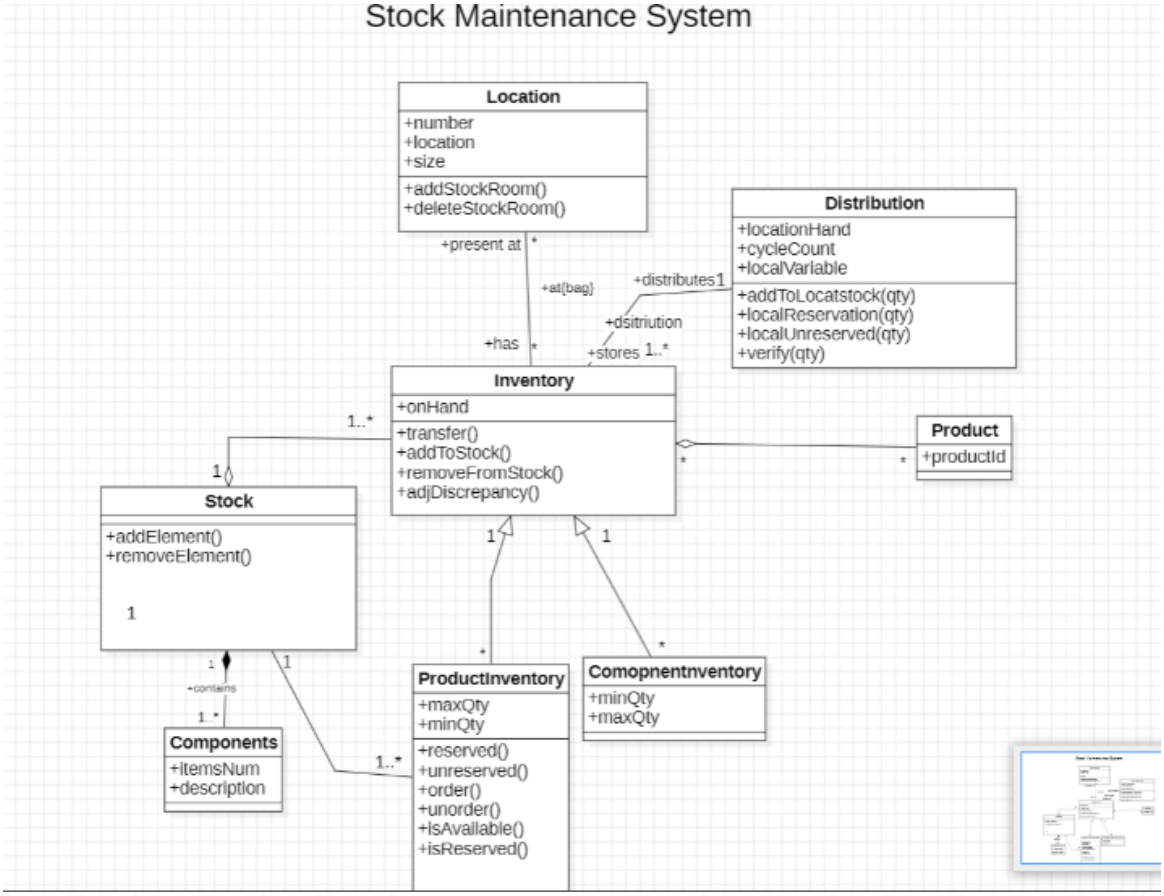
5. Performance requirements :
   - → The system's data has to synchronised with the stock present in the inventory
   - → The duration of the stocks present in the inventory has to be maintained.
   - → The system's data has to be secured.

6. Design Constraints:
   - → The interface has to be easy to use.
   - → The orders for storing should not exceed the capacity of the inventory
   - → The software should automatically update the stocks based on the deliveries.

4. Non-Functional Attributes:
   - → The data should be secure & portable
   - → The data should be reliable.

8. Preliminary schedule & budget: A budget can be prepared based on the staff required for maintaining the inventory and the deliveries to be made.

**Class Diagram**




- LoginProcess:

  The user or store owner logs into the system using credentials. This grants access to the relevant functionalities, such as product purchasing or stock management.

- ProductSearch

  Customers search for products in the system's inventory. Upon finding the desired items, they proceed to make a purchase.

- StockVerification:

  The system checks the stock levels in the store or warehouse to ensure availability. If stock is insufficient, the system triggers a restocking process.

- OrderSupply:

  The store owner places an order with the supplier for products that are out of stock. The supplier verifies the payment and dispatches the required items.

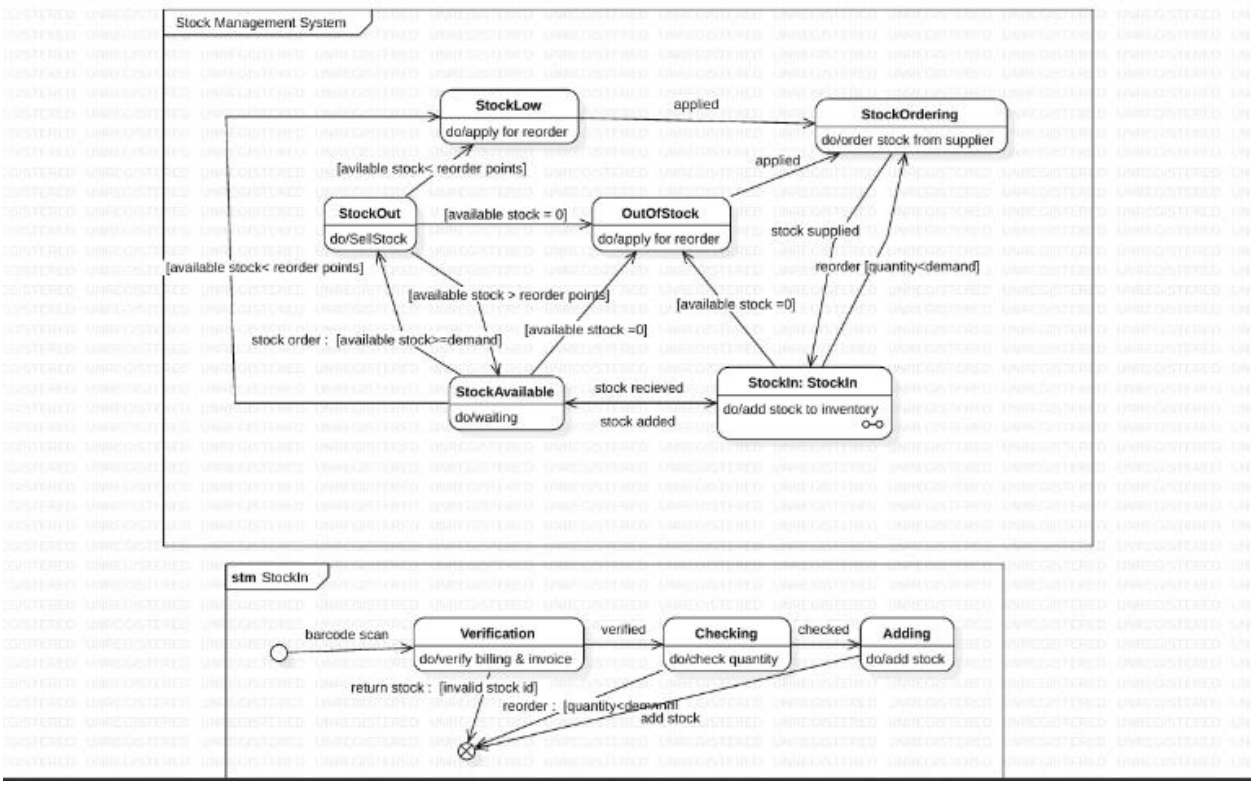
- ReceiveProduct:

  After supply delivery, the store owner receives and verifies the products. The system updates inventory records to reflect the new stock levels.

- UpdateDatabase:

  The stock and transaction records are updated in the database after any purchase or stock addition. This ensures accurate inventory tracking and reporting.

**State Diagram**

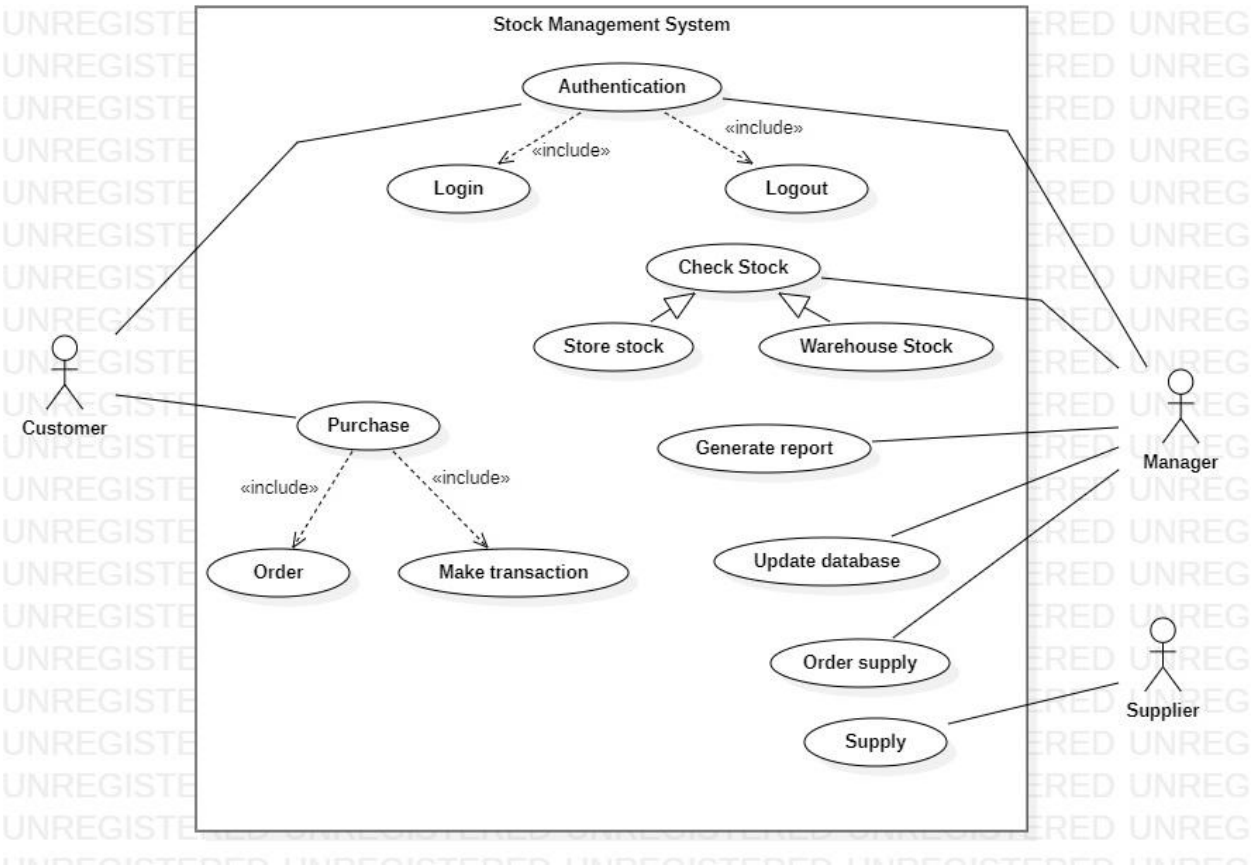



1. Stock Available:
   - Represents the initial state when sufficient stock is present.
   - Actions: Monitor the stock level and fulfill customer orders.
2. Stock Low:

- Triggered when stock levels reach reorder points.
- Actions: Generate a reorder request to maintain inventory levels.

3. Out of Stock:
    - Occurs when available stock falls to zero.
    - Actions: Halt order fulfillment and prioritize restocking.
4. Stock Ordering:
    - Represents the process of ordering new stock from a supplier.
    - Actions: Place an order with the supplier and track the stock shipment.
5. Stock In:
    - The state where new stock has been received and is being added to the inventory.
    - Actions: Verify stock quantity and update the system inventory.
6. Stock Out:
    - Represents the sale or distribution of stock to customers or stores.
    - Actions: Deduct the quantity from inventory and update stock levels.

**Use Case Diagram**




Actors

- Customer:

  Interacts with the system to purchase products and view available stock.

- Store Owner:

  Manages the inventory, checks stock, and places orders for supply replenishment.

- Supplier:

  Supplies stock when ordered by the store owner.

- Manager:

  1. Oversees overall operations, generates reports, and updates the database.

Key Use Cases

- Authentication:

- Description: Enables users (customers, store owners, and managers) to log in and log out securely.
- Includes: Logging in and logging out.

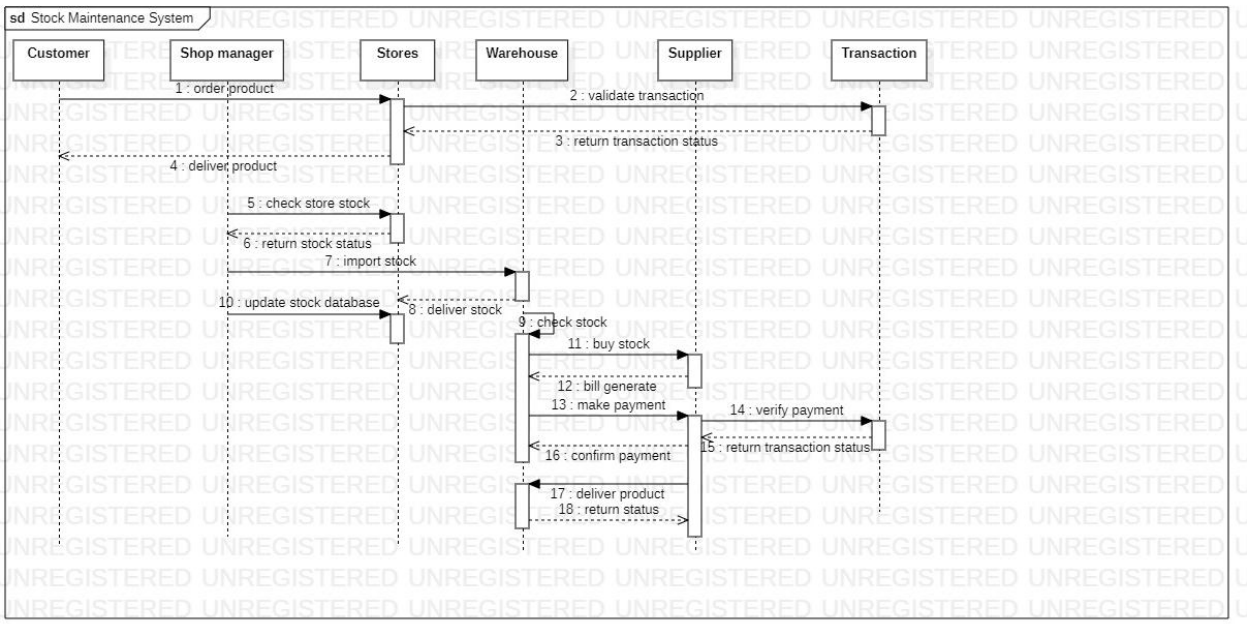
- Check Stock:
  - Description: Users can view the available stock levels in the store or warehouse.
  - Includes: Checking store stock and warehouse stock.
- Purchase Products:
  - Description: Customers search for and buy products, which deducts from the available stock.
  - Includes: Ordering and making transactions.
- Order Supply:
  - Description: Store owners place orders with suppliers to replenish stock.
  - Includes: Verifying payment and receiving stock.
- Update Database:
  - Description: Managers and the system ensure that all stock-related data is accurate and up-to-date.
  - Includes: Adding and removing inventory entries.
- Generate Reports:
  - Description: Managers generate reports to analyze stock levels, purchases, and sales trends.
  - Includes: Accessing historical stock data for insights.

Interactions

1. Customer:
   - Initiates purchase actions, views stock, and makes transactions.
2. Store Owner:
   - Ensures stock availability by monitoring levels and placing supply orders.
3. Supplier:
   - Supplies products and verifies payments for received orders.
4. Manager:

- Oversees operations and maintains the integrity of the system's data.

**Sequence Diagram**




Actors:

- Customer: Orders products.
- Shop Manager: Validates transactions, checks stock, manages imports, updates stock database.
- Stores: Provides stock status and updates stock database.
- Warehouse: Imports stock, checks stock, delivers stock, verifies payments.
- Supplier: Supplies requested stock, generates bills, processes payments.
- Transaction: Validates transactions, returns transaction status, verifies payments.

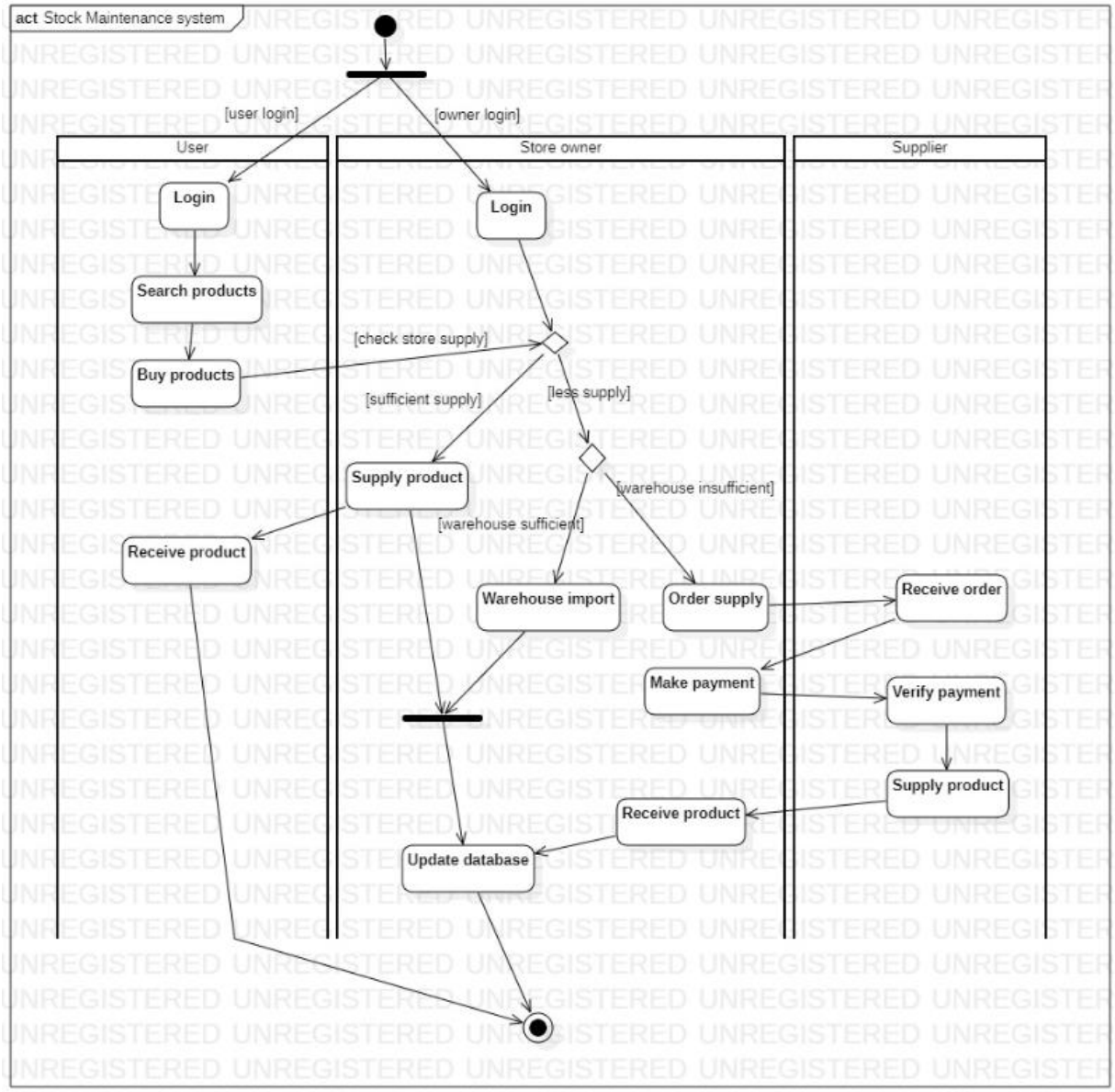
Interactions:

- Ordering Product:
    - The customer places an order with the shop manager.
    - The shop manager validates the transaction with the transaction system.
    - The transaction system returns the transaction status to the shop manager.

    - The shop manager delivers the product to the customer.

- Checking and Updating Stock:
    - The shop manager checks the stock status with the stores.
    - The stores return the stock status to the shop manager.
    - The shop manager requests stock from the warehouse if needed.
    - The warehouse delivers the stock to the shop manager.
    - The shop manager updates the stock database in the stores.
- Procurement Process:
    - The warehouse checks stock with the supplier.
    - The supplier delivers the stock to the warehouse.
    - The warehouse generates a bill and processes the payment with the supplier.
    - The transaction system verifies the payment.
    - The warehouse confirms the payment and the supplier delivers the product.

## Activity Diagram




- Login Process:
    - Flow:
        - Customers or store owners log in.
        - Authentication is verified.
        - Users proceed to their respective tasks (search products or manage stock).
- Search and Purchase Products:
    - Flow:
        - Customers search for products.

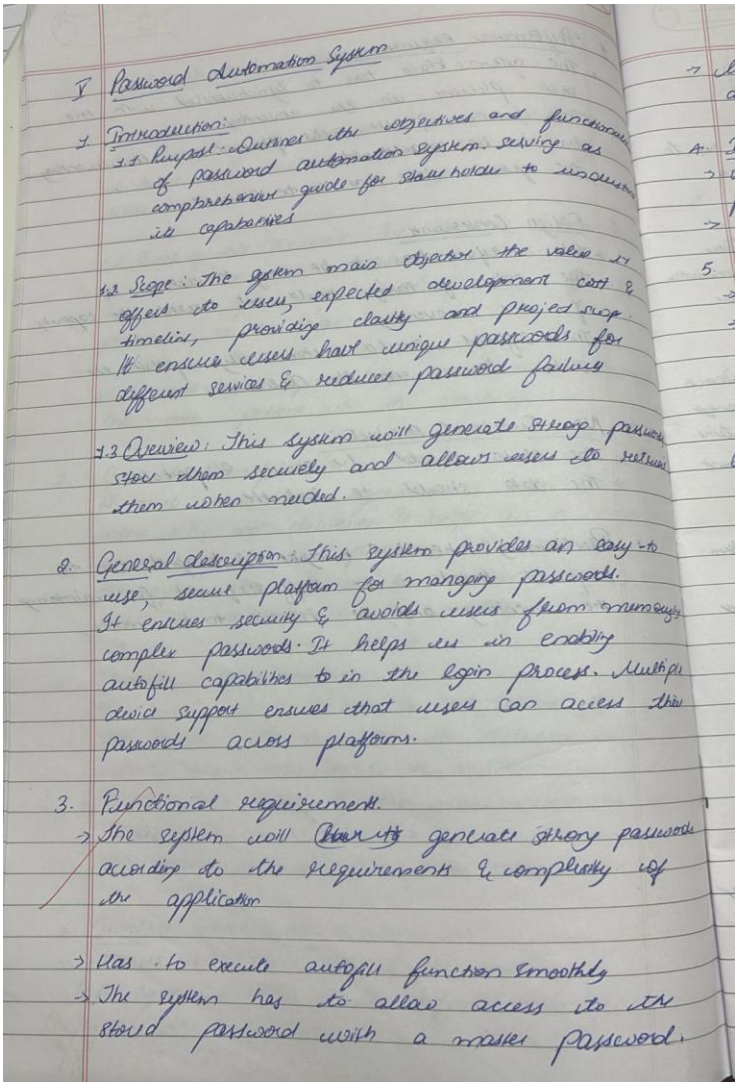
        - If products are available, they proceed to purchase.
        - The system processes the payment and initiates delivery.
- Stock Check and Replenishment:
    - Flow:
        - Store owners review stock levels.
        - If stock is sufficient, the process stops.
        - If stock is insufficient, the store owner orders supplies.
- Supply Order Management:
    - Flow:
        - Store owners place an order for supply.
        - Suppliers receive the order and verify payment.
        - Stock is delivered to the store, and the inventory is updated.
- Update Inventory Database:
    - Flow:
        - Once a product is sold or new stock is received, the database is updated automatically.

# 5.Password Automation System

**Problem Statement**

The manual processing of passport applications is often time-consuming, prone to errors, and inefficient, leading to delays in issuance and renewal. Applicants face challenges such as lengthy queues, incomplete information, and lack of real-time updates on application status. Additionally, managing large volumes of data manually can result in errors, security concerns, and difficulty in tracking and retrieving records. These inefficiencies hinder the overall process and compromise user satisfaction and operational effectiveness.

**SRS – Software Requirements Specification**



I Password Automation System

1. Introduction:
1.1 Purpose: Outlines the objectives and functionalities of password automation system. serving as comprehensive guide for stakeholders to understand its capabilities

1.2 Scope: The system main objective the value it offers to users, expected development cost & timelines, providing clarity and project scope. It ensures users have unique passwords for different services & reduces password failure

1.3 Overview: This system will generate strong passwords store them securely and allows users to retrieve them when needed.

2. General description: This system provides an easy-to use, secure platform for managing passwords. It ensures security & avoids users from memorizing complex passwords. It helps us in enabling autofill capabilities to in the login process. Multiple device support ensures that users can access their passwords across platforms.

3. Functional requirements.
→ The system will generate strong passwords according to the requirements & complexity of the application

→ Has to execute autofill function smoothly
→ The system has to allow access to the stored password with a master password.

→ Users can search & retrieve stored passwords thru a web interface or mobile app

4. Interface Requirements
→ A user interface to allow users to manage passwords
→ User authentication with master password.

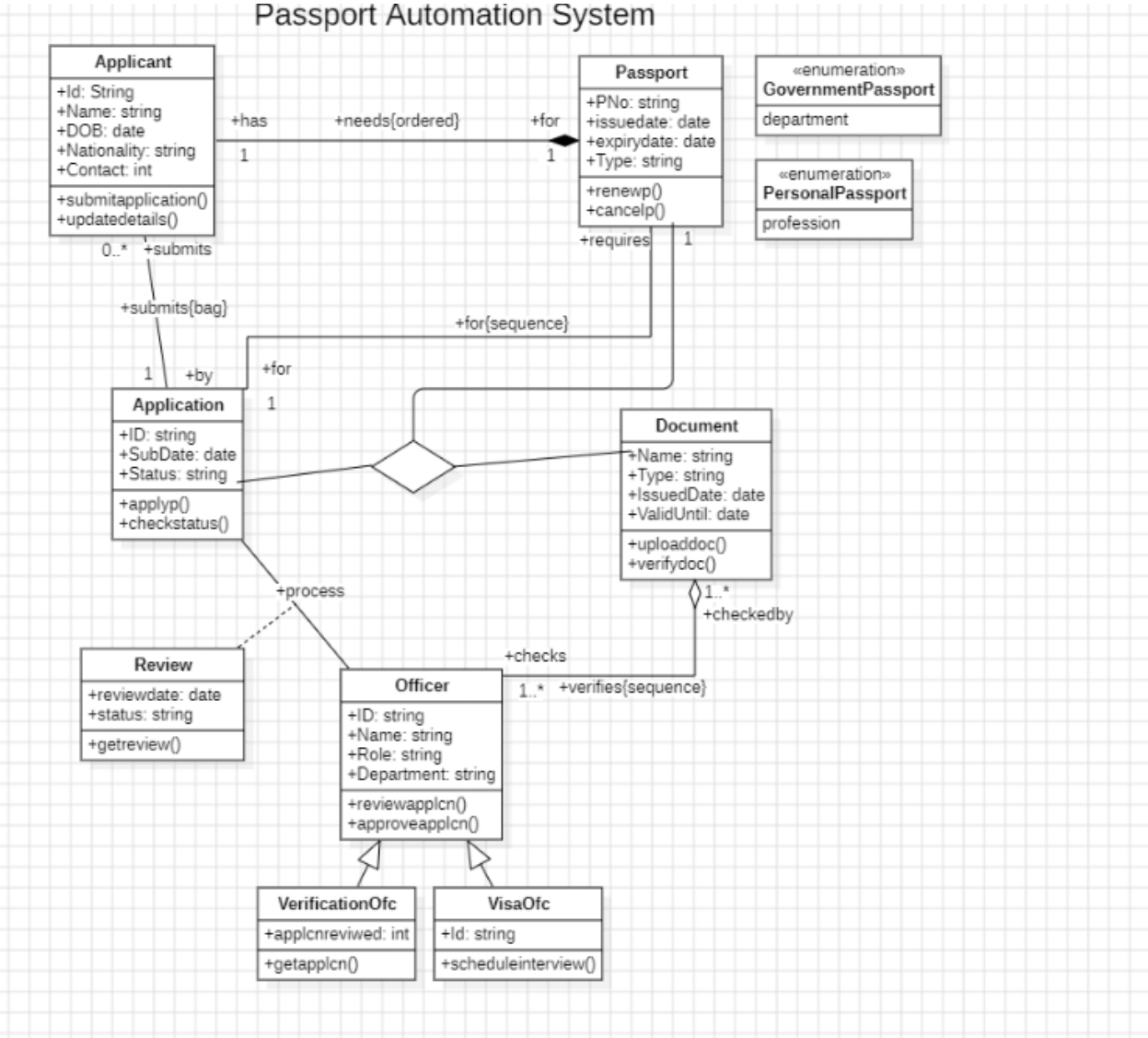
5. Performance Requirements
→ The system has to be secure
→ Password retrieval & and autofill operations response time should be fast
→ The system must be adaptable in any platform

6. Design Constraints
→ ~~OOA~~ RDBMS to store all data
→ AES - encryption to store & secure the passwords
→ Browser extensions to comply with browser policies to avoid installation issue

7. Non-Functional-attributes
→ The system has to be secure, portable & reliable
→ Passwords & user data must remain intact and unaltered unless changed by the user.

8. Preliminary Schedule & Budget
→ The project estimated development cost - $10,000 and 3-months to complete.

**Class Diagram**



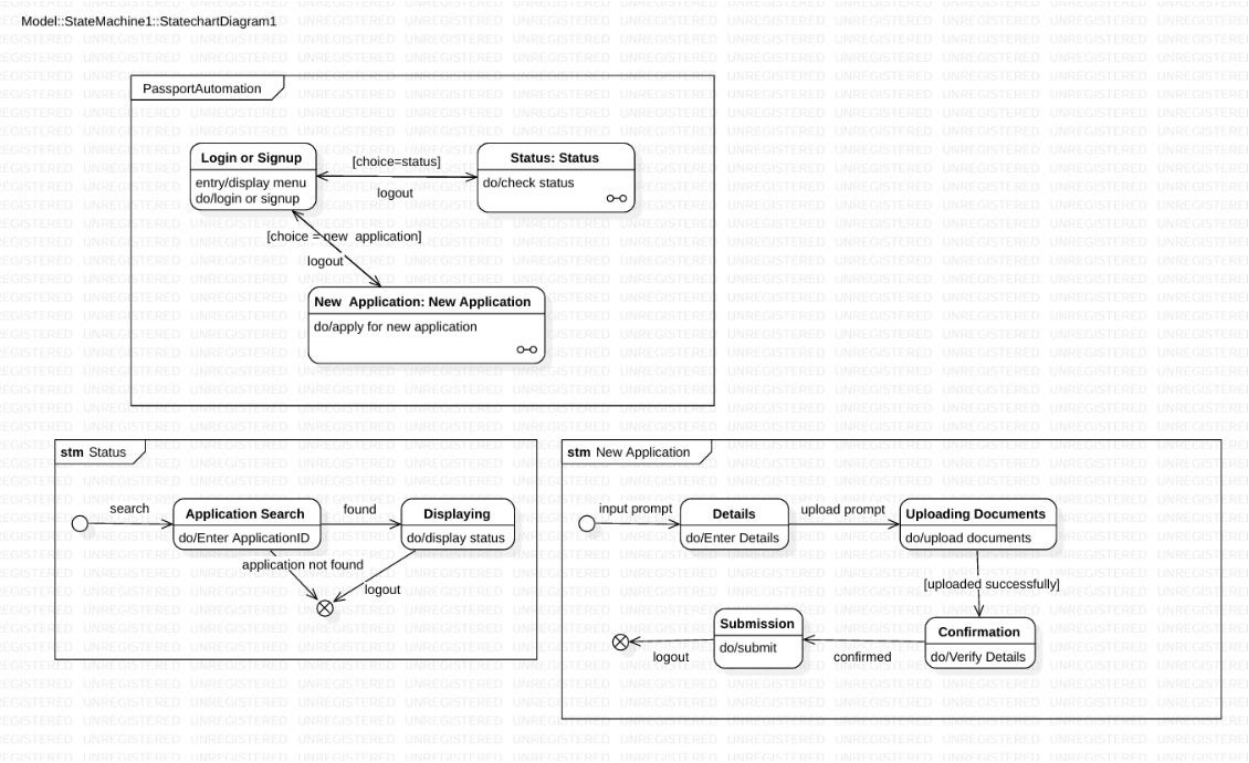

Classes:

- Applicant: Captures personal details such as name, date of birth, nationality, and contact information.
- Passport: Includes passport-related attributes like passport number, type (e.g., government or personal), and expiry date.
- Document: Represents uploaded documents with details like document type, issue date, and validity.
- Officer: Represents the admin responsible for verifying and approving applications. Attributes include name, role, and ID.

- Verification/Transaction System: Handles document validation and payment processing.

- Relationships:
  - Aggregation: The applicant is associated with one or more applications, and each application is linked to multiple documents.
  - Association: Verification and approval are associated with the officer role, and payments are tied to the applicant.
- Enumerations: Differentiates between various passport types (e.g., official, personal) for clarity.

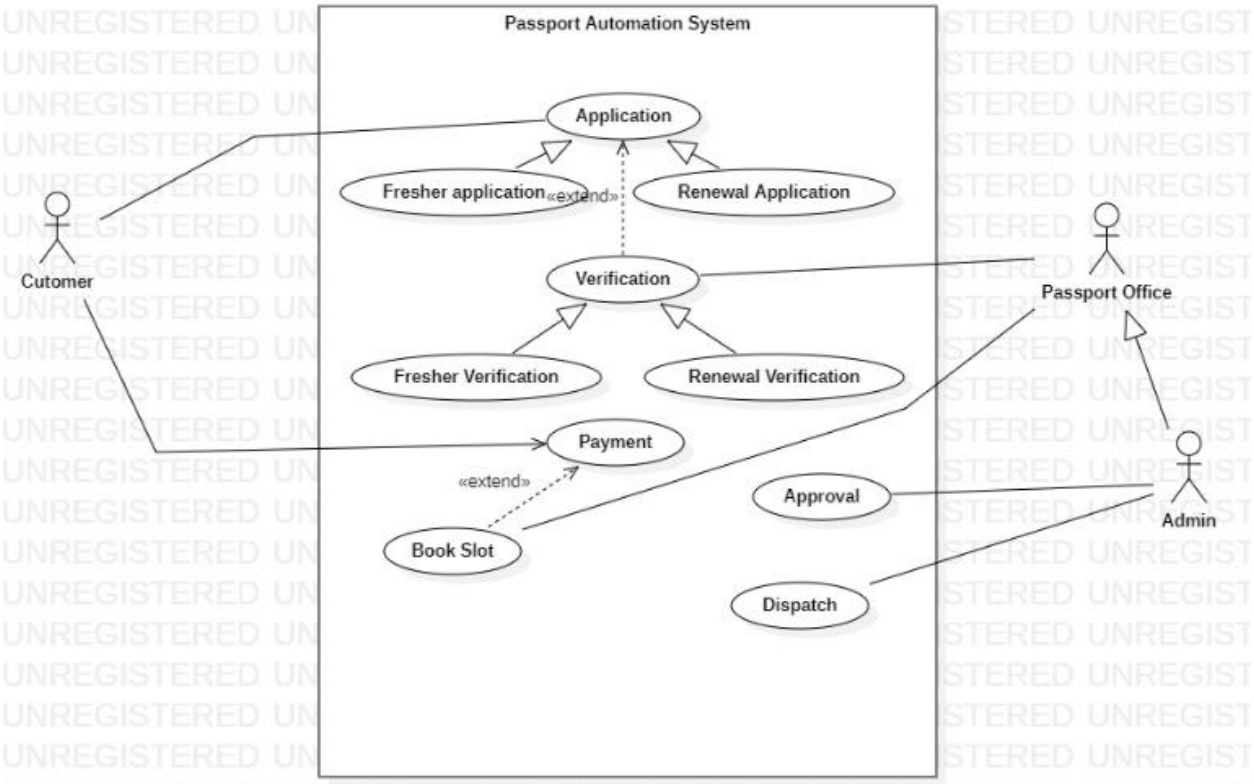
**State Diagram**




- Login/Signup: The applicant starts by logging into the system or signing up for a new account.
- New Application: The system transitions to the application phase, where the applicant provides details, uploads documents, and confirms submission.
- Verification and Approval: After submission, the application enters the verification state, where admins validate the information and approve or reject it based on compliance.

- Status Checking: Applicants can query the status of their application, such as "Pending Approval," "Rejected," or "Approved."
- Final State: The final state is reached when the passport is successfully issued and delivered.
- Transitions: Clear transitions show the flow from one state to another, such as moving from the "Submitted" state to the "Under Verification" state.

**Use Case Diagram**

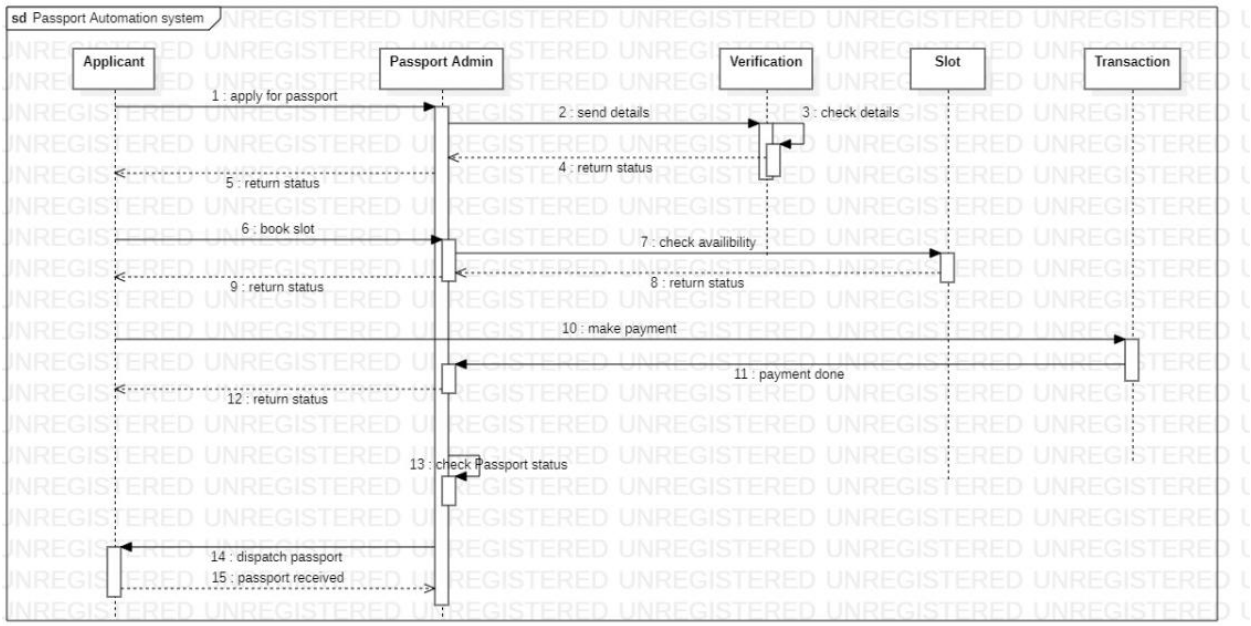



- Actors:
  - The applicant is the primary actor responsible for submitting applications, booking appointments, and tracking progress. The admin represents the Passport Office personnel who process and approve applications.
- Use Cases:
  - For Applicants: Apply for a new passport or renew an existing one, upload supporting documents, make payments, book appointment slots, and check the application status.

- For Admins: Verify submitted details, approve or reject applications, schedule document dispatch, and manage the overall application lifecycle.

- Relationships:
  - The use cases demonstrate relationships such as "includes" and "extends," e.g., making payments extends the application process, and approval includes document verification.

**Sequence Diagram**




- Actors:
  - The main actors are the applicant (user), the Passport Admin (system operator), and supporting systems such as Verification, Slot Management, and Payment Systems.

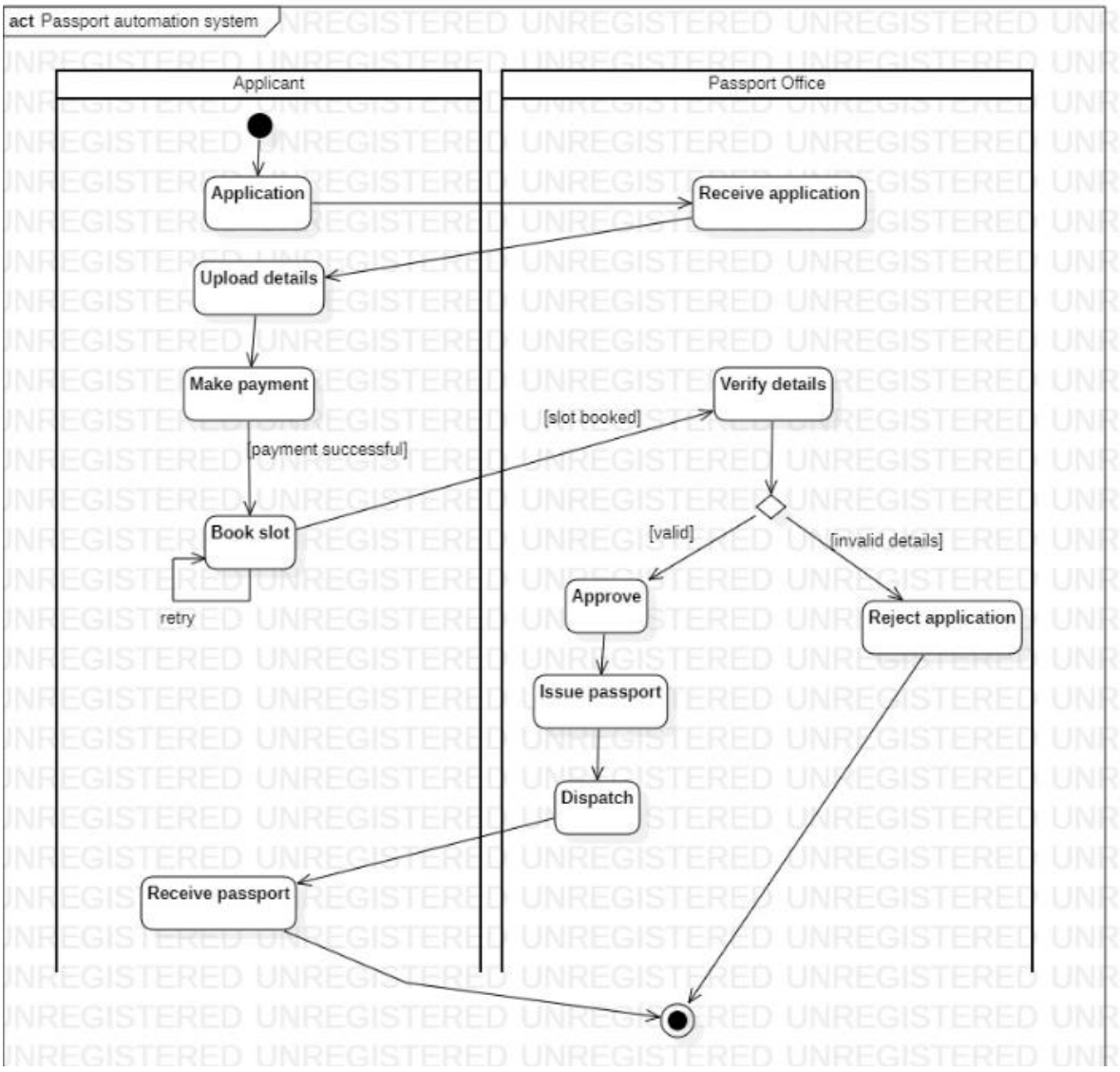
- Flow:
  - The applicant initiates the process by submitting an application through the system.
  - The system forwards the details to the Verification component, which validates the data and sends feedback (e.g., valid or invalid details).

- The applicant then selects a convenient slot for an in-person appointment, makes the required payment, and tracks the status of their application.
- Once approved, the system finalizes the application by dispatching the passport.

o Focus:

- Each interaction is represented clearly, showing how the system components collaborate to ensure smooth operation.

**Activity Diagram**



- Applicant Actions:
    - The applicant starts by logging into the system, filling in their details, uploading required documents (e.g., proof of address, ID proof), making payments for application fees, and booking an appointment for verification.
- Passport Office Actions:
    - After receiving the application, the office performs actions such as validating the provided documents, cross-checking details, scheduling verification, and making decisions on approval or rejection.
    - Once the application is approved, the passport is issued, printed, and dispatched to the applicant's address.
- Decision Points:
    - Conditional checks include verifying the completeness and correctness of documents and processing payment status.
    - Loops exist for retries in case of payment failure or invalid/insufficient information.
- End State:
    - The process concludes with the successful dispatch of the passport, ensuring all required verifications and steps are completed.