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:

Shamanth K Murthy

(1BM22CS251)

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
Oct 2024 - Jan 2025

B.M.S. COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Object Oriented Modelling (23CS5PCOOM) laboratory has been carried out by SHAMANTH K MURTHY (1BM22CS251) during the 5th Semester Oct 2024 - Jan 2025.

Signature of the Faculty Incharge:

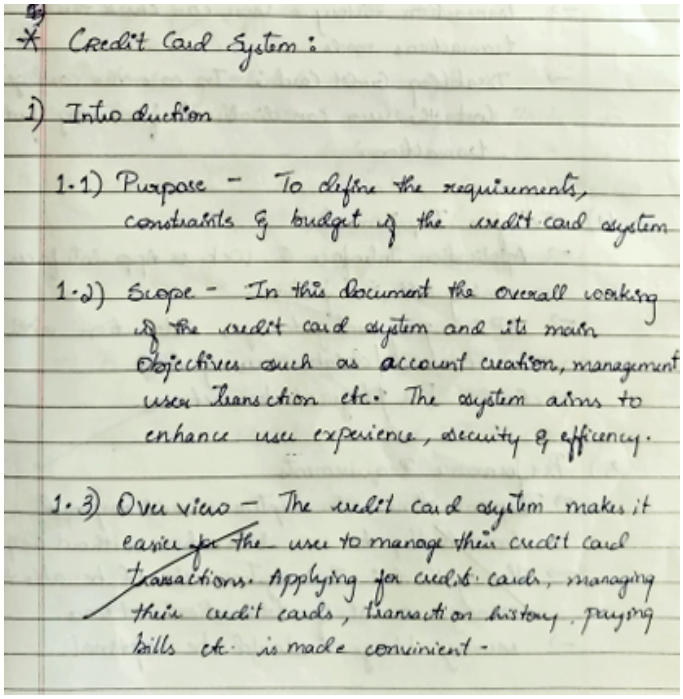
NAME OF THE FACULTY: Prof. Sunayana S

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

Table of Contents

1. Hotel Management System

SRS - Software Requirements Specification



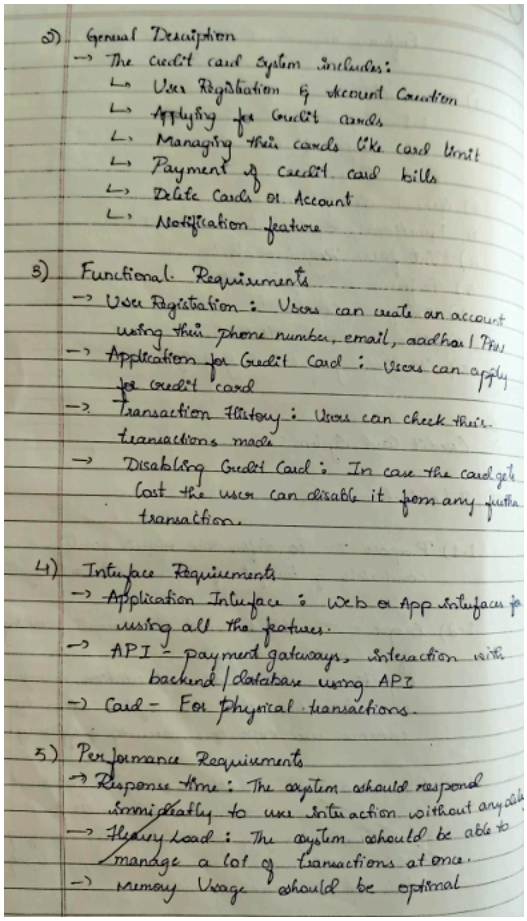
* Credit Card System :

1) Introduction

1.1) Purpose - To define the requirements, constraints & budget of the credit card system

1.2) Scope - In this document the overall working of the credit card system and its main objectives such as account creation, management user transaction etc. The system aims to enhance user experience, security & efficiency.

1.3) Overview - The credit card system makes it easier for the user to manage their credit card transactions. Applying for credit cards, managing their credit cards, transaction history, paying bills etc. is made convinient -

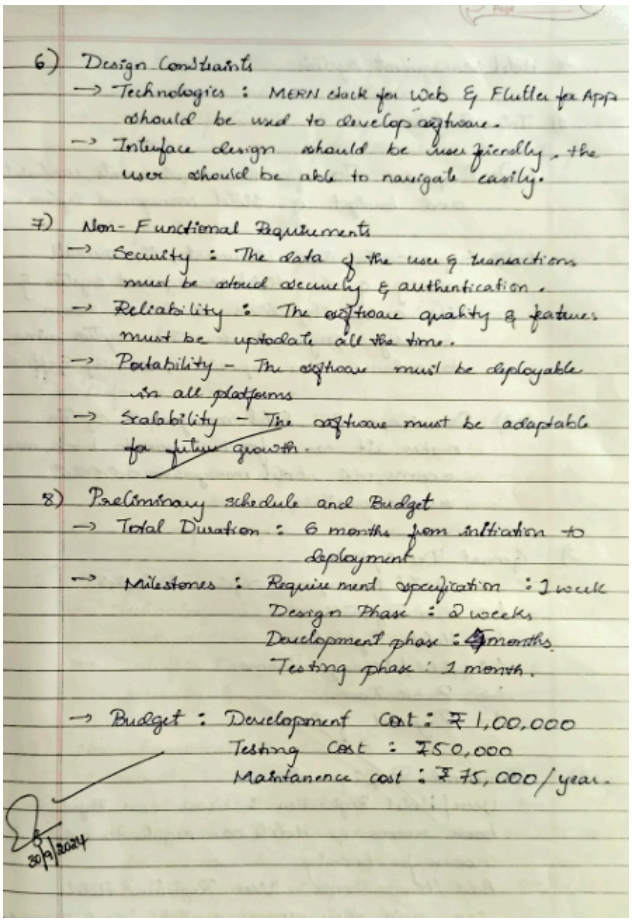


2) General Description
→ The credit card system includes:
- User Registration & Account Creation
- Applying for Credit cards
- Managing their cards like card limit
- Payment of Credit card bills
- Delete Cards or Account
- Notification feature

3) Functional Requirements
→ User Registration : Users can create an account using their phone number, email, aadhaar / PAN
→ Application for Credit Card : Users can apply for credit card
→ Transaction History : Users can check their transactions made
→ Disabling Credit Card : In case the card gets lost the user can disable it from any further transaction.

4) Interface Requirements
→ Application Interface : Web or App interfaces for using all the features.
→ API - payment gateways, interaction with backend / database using API
→ Card - For physical transactions.

5) Performance Requirements
→ Response time : The system should respond immediately to user interaction without any delay
→ Heavy Load : The system should be able to manage a lot of transactions at once.
→ Memory Usage should be optimal



6) Design Constraints
→ Technologies : MERN stack for Web & Flutter for Apps should be used to develop software.
→ Interface design should be user friendly, the user should be able to navigate easily.

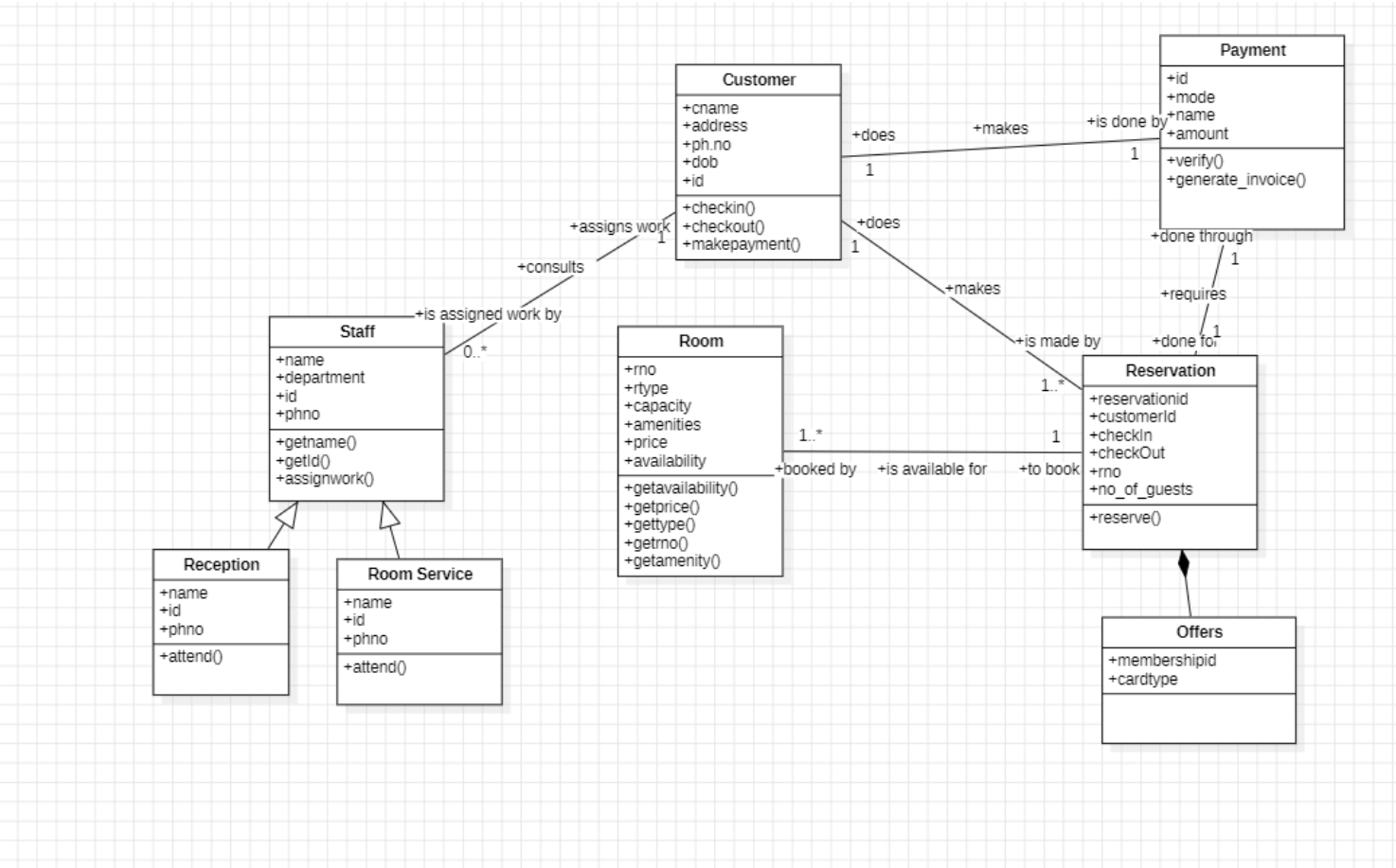
7) Non-Functional Requirements
→ Security : The data of the user & transactions must be stored securely & authentication.
→ Reliability : The software quality & features must be updated all the time.
→ Portability - The software must be deployable in all platforms
→ Scalability - The software must be adaptable for future growth.

8) Preliminary schedule and Budget
→ Total Duration : 6 months from initiation to deployment
→ Milestones : Requirement specification : 1 week
Design Phase : 2 weeks
Development phase : 3 months
Testing phase : 1 month.

→ Budget : Development cost : ₹ 1,00,000
Testing cost : ₹ 50,000
Maintanence cost : ₹ 75,000 / year.

30/9/2024

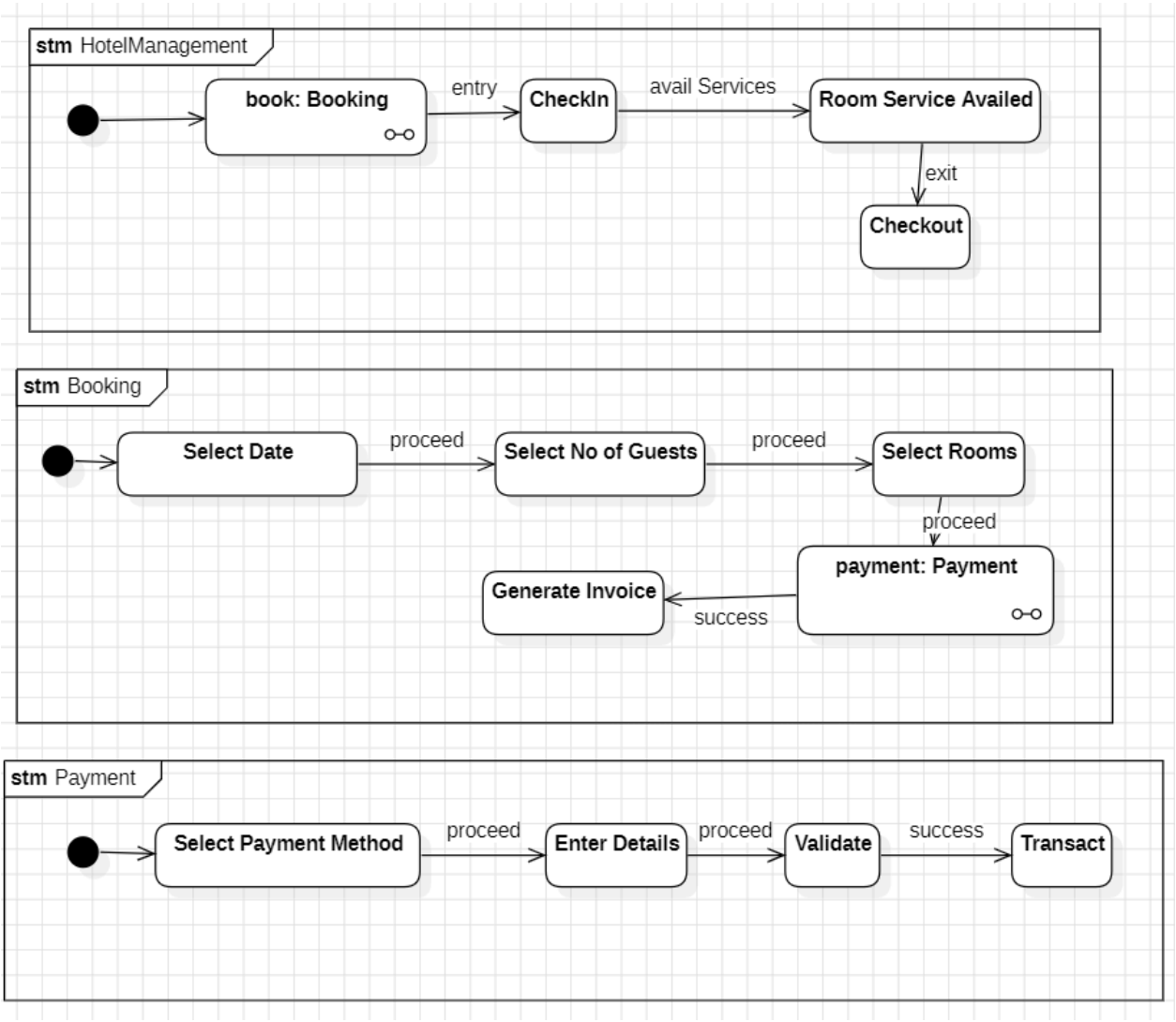
Class Diagram



Description:

- Customer - Class that has attributes and functions indicative of the customer who books the hotel.
- Room - Indicates the room that can be booked.
- Payment - A class that has attributes and functions for a transaction.
- Staff - Indicative of hotel staff.
- Reservation - Contains attributes and functions aiding the process of reservation.
- Generalization - Two classes Reception and Room Service inherit properties of the Staff class.
- Composition - The Offers class is a composition of Reservation i.e if reservation ceases to exist this class does not exist.

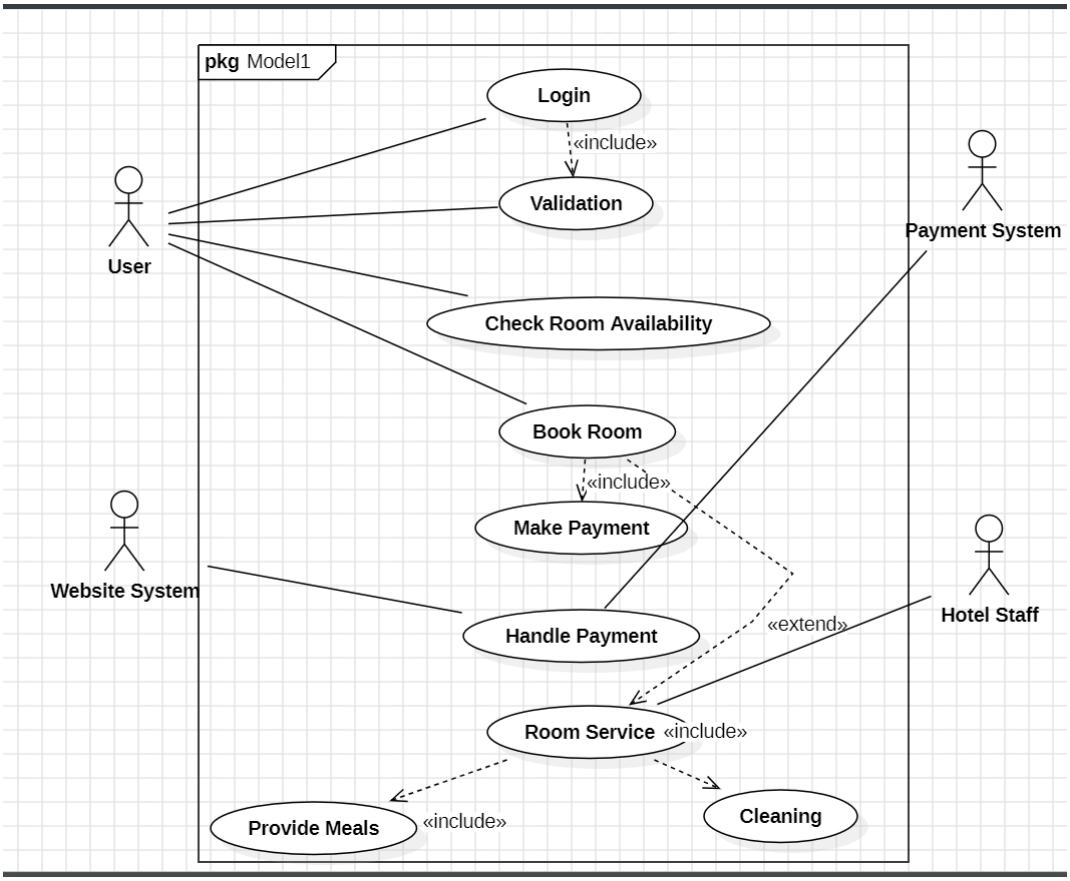
State Diagram



Description:

- There is a main Hotel Management state diagram and two submachine states called Booking and Payment.
- Hotel Management: Contains states such as Checkin - once the guest enters, Room Service Availed - if the guest opts for it and CheckOut on exit
- Booking: This submachine state has states to facilitate booking process i.e Select Date, Select no of guests, rooms, a submachine called payment and a state to generate invoice.
- Payment: This submachine has states for payment purpose such as Enter Details, Validate and Transact.

Use Case Diagram



Description:

Actors involved: Hotel Staff, Website System, User, Payment System

User:

Login - The user can login to the website.

Check Room Availability - They can check if rooms are available.

Book room - They can make a booking. It includes making a payment and extends room service (choosing it is optional)

Payment System:

Handle payment - It handles payment.

Website System:

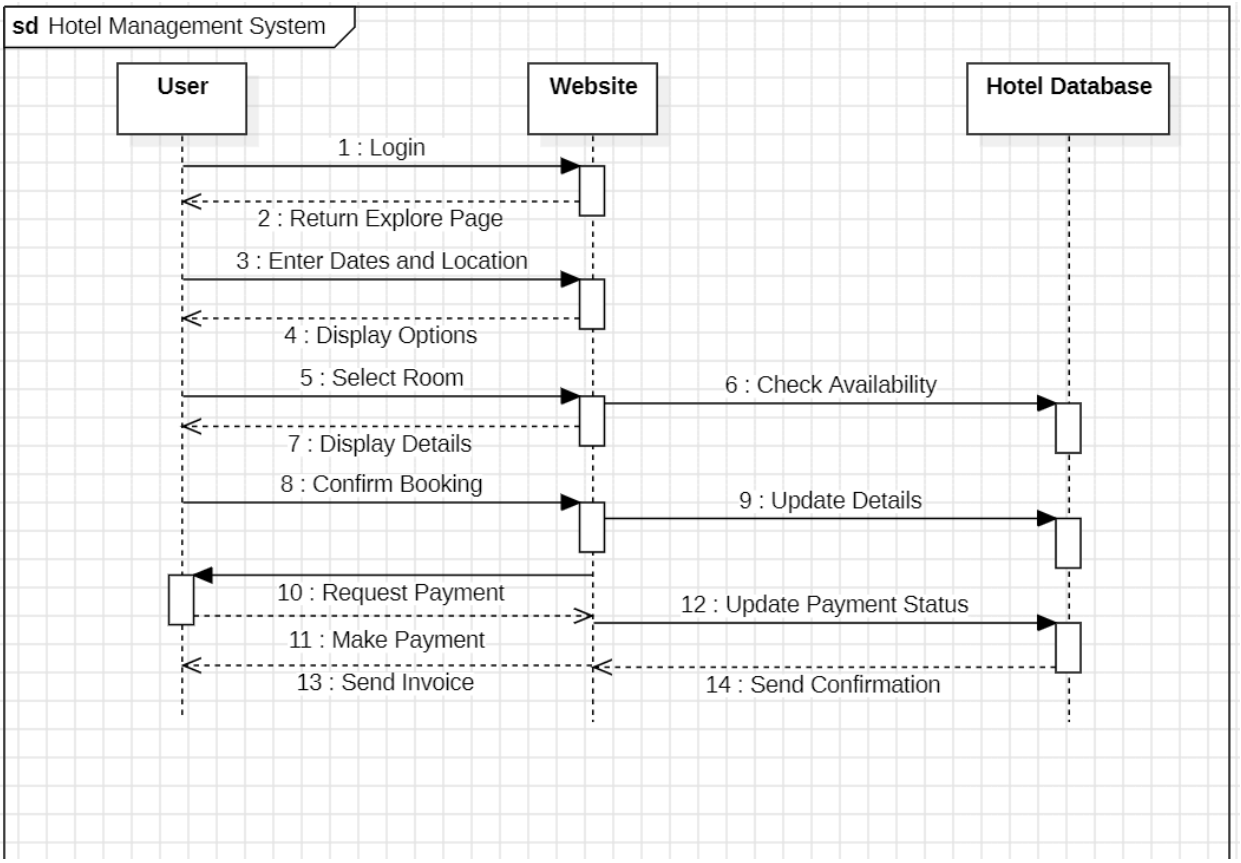
Validation - Validates the user from its database.

Handle payment - Also handles payment process.

Hotel Staff:

Room Service - Provide room service to the guests. Includes providing meals and cleaning.

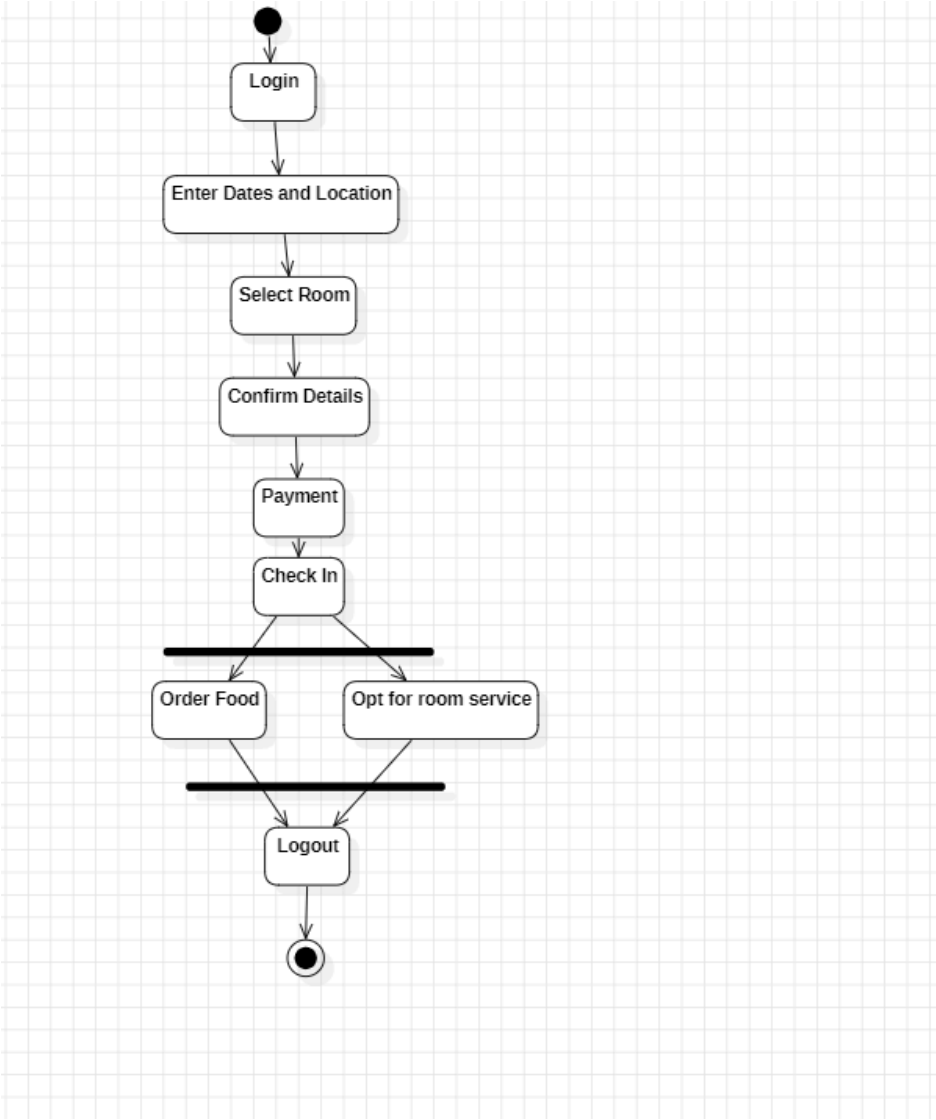
Sequence Diagram



Description:

- The user logs in to the website. If the user is valid then the explore page is returned.
- The user enters the date and location. The website displays the available hotel options.
- Once the user selects the room of their choice the website enquires the hotel database about its availability.
- The website displays the details. The user confirms their booking, the user's details are updated in the hotel database.
- There is a payment request initiated by the website, the user makes the payment and this is updated in the hotel database.
- On confirmation from hotel database an invoice is sent to the user.

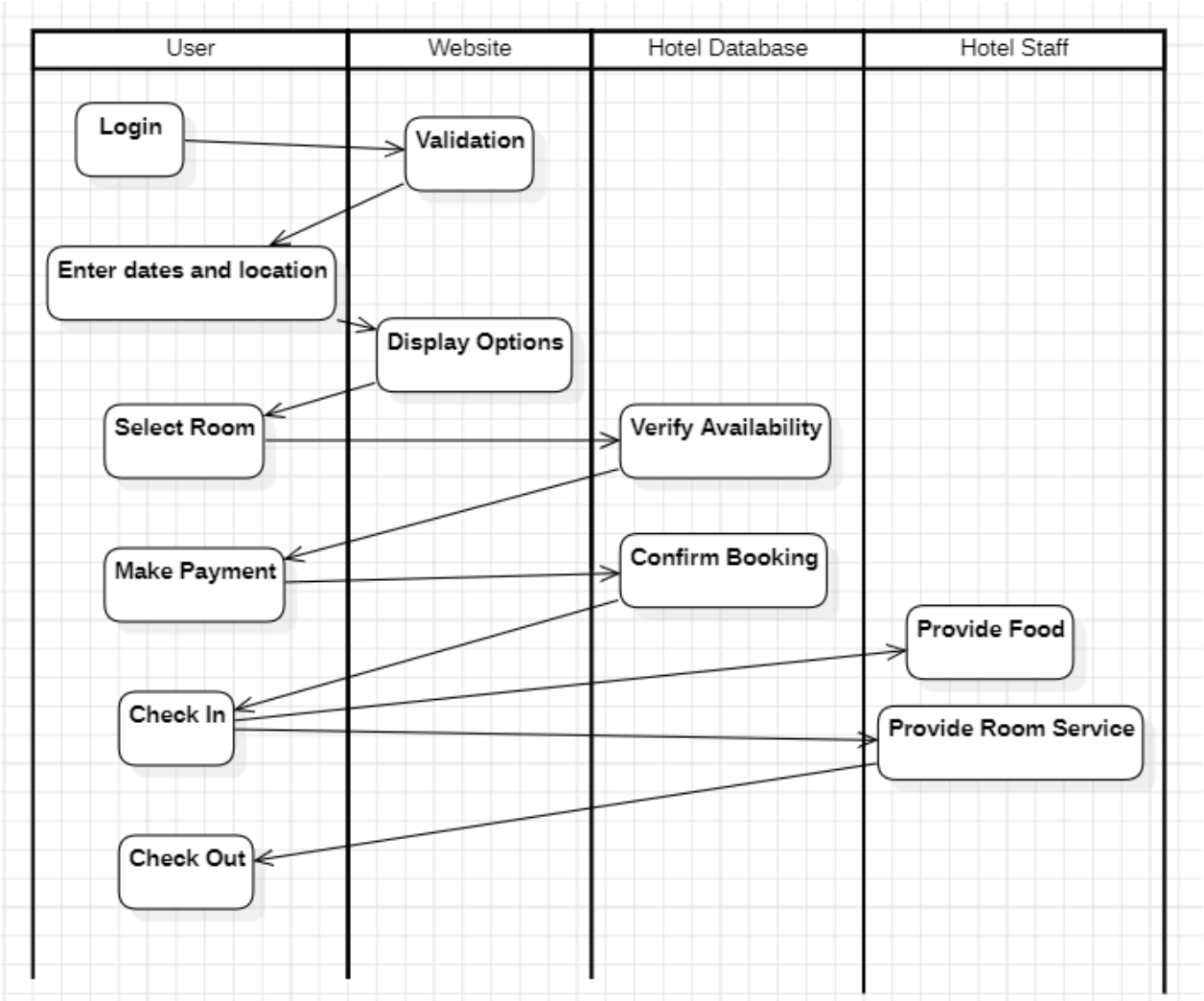
Activity Diagram



Description:

- The user logs in, enters the date and location in the website.
- Once the website displays the available options they select a room, confirm details and make payment.
- Post check in they may order food and opt for room service simultaneously.
- The user checks out.

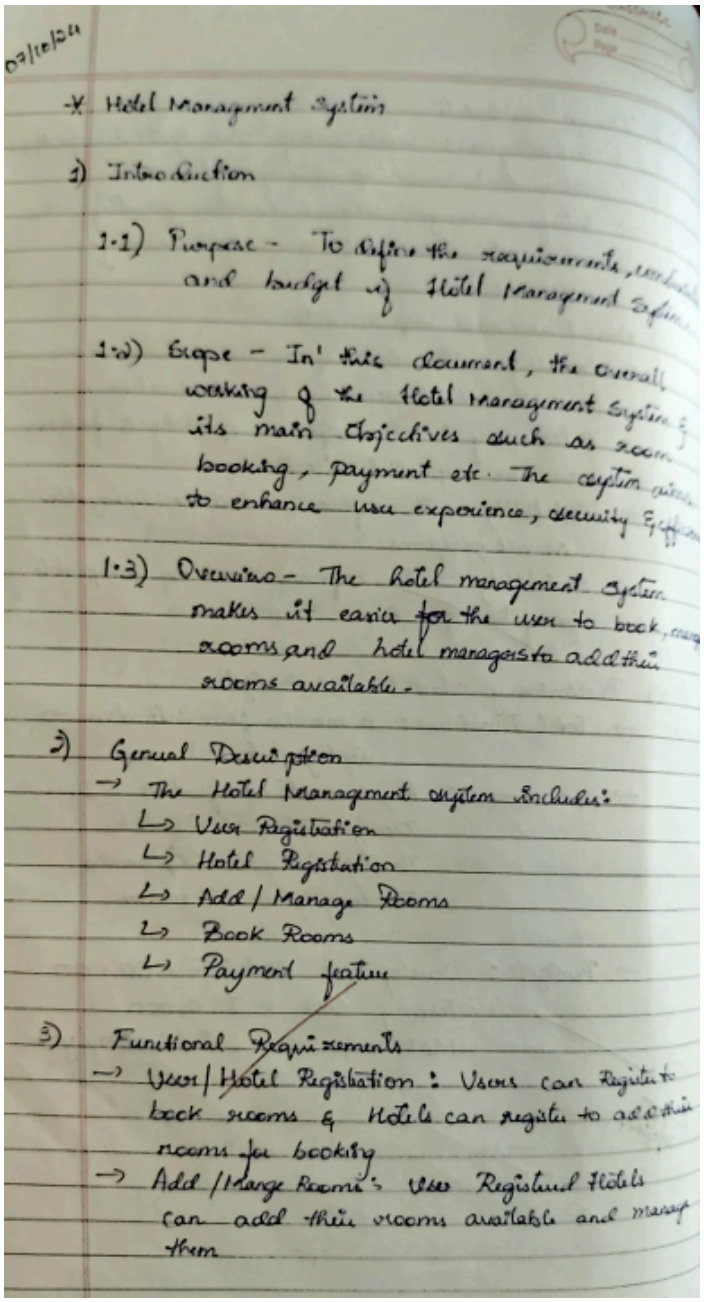
Activity Diagram using Swimlane



The diagram includes 4 swimlanes namely - User, Website, Hotel Database and Hotel Staff.

2. Credit Card Processing System

SRS - Software Requirements Specification



07/10/24

-X Hotel Management System

1) Introduction

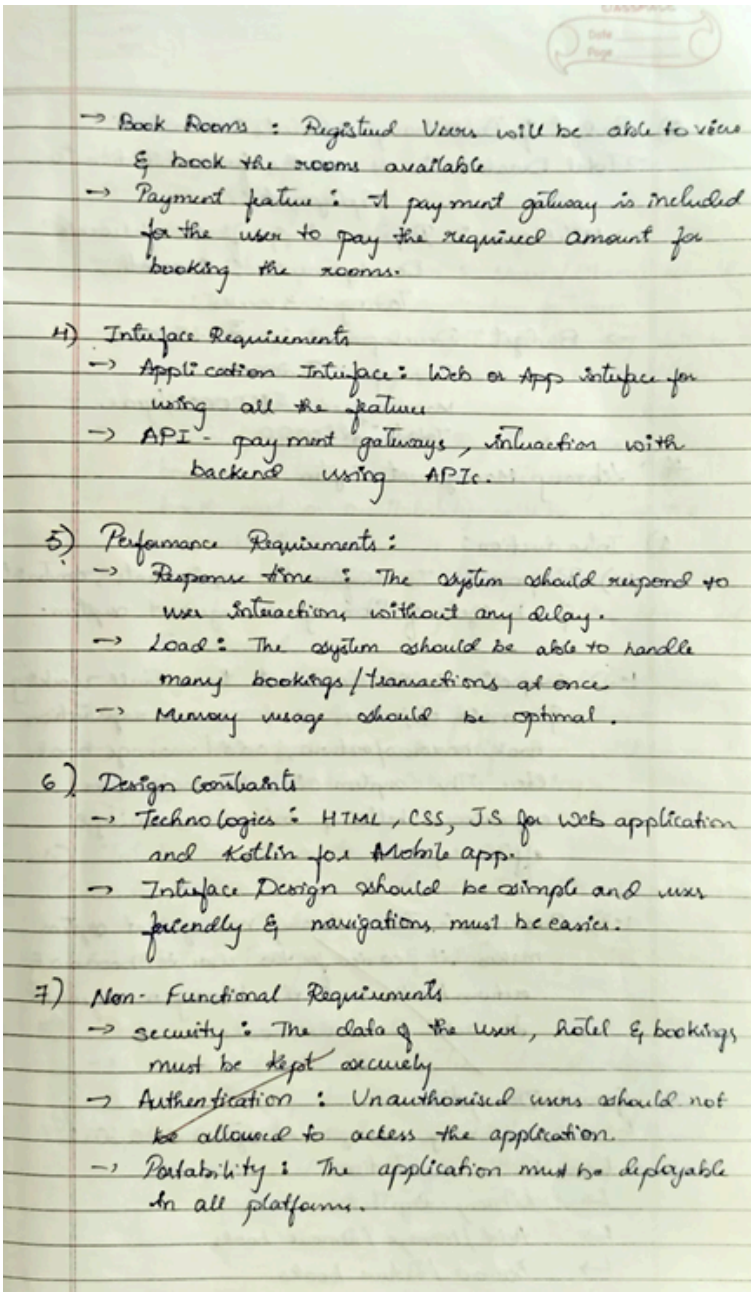
1.1) Purpose - To define the requirements, constraints and budget of Hotel Management System

1.2) Scope - In this document, the overall working of the Hotel Management System & its main objectives such as room booking, payment etc. The system aims to enhance user experience, security & efficiency

1.3) Overview - The Hotel management system makes it easier for the user to book, manage rooms and hotel managers to add their rooms available.

2) General Description
→ The Hotel Management system includes:
  ↳ User Registration
  ↳ Hotel Registration
  ↳ Add / Manage Rooms
  ↳ Book Rooms
  ↳ Payment feature

3) Functional Requirements
→ User / Hotel Registration : Users can Register to book rooms & Hotels can register to add their rooms for booking
→ Add / Manage Rooms : User Registered Hotels can add their rooms available and manage them



→ Book Rooms : Registered Users will be able to view & book the rooms available
→ Payment feature : A payment gateway is included for the user to pay the required amount for booking the rooms.

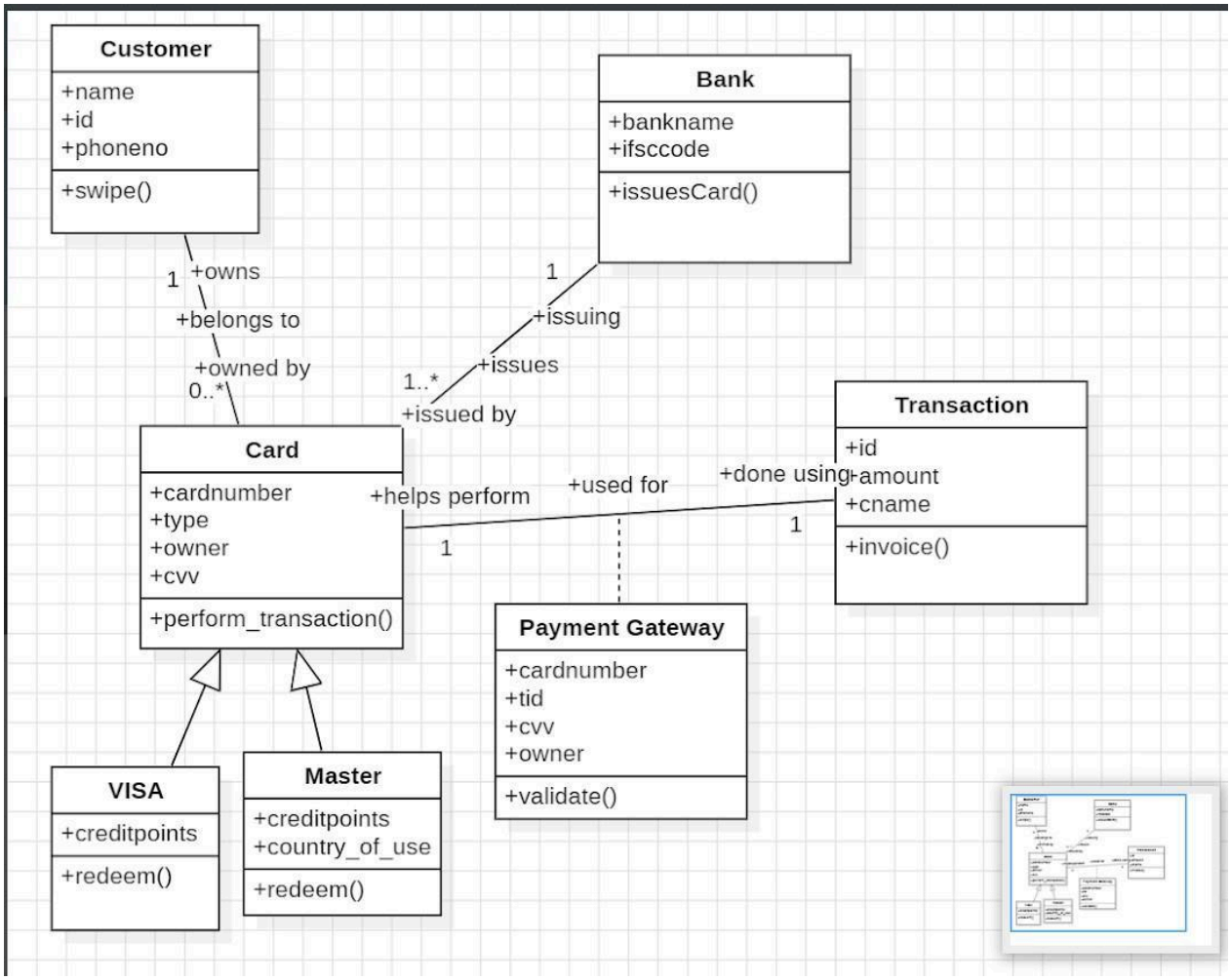
4) Interface Requirements
→ Application Interface : Web or App interface for using all the features
→ API - payment gateways, interaction with backend using APIs.

5) Performance Requirements :
→ Response time : The system should respond to user interactions without any delay.
→ Load : The system should be able to handle many bookings / transactions at once
→ Memory usage should be optimal.

6) Design Constraints
→ Technologies : HTML, CSS, JS for web application and Kotlin for Mobile app.
→ Interface Design should be simple and user friendly & navigations must be easier.

7) Non-Functional Requirements
→ Security : The data of the user, hotel & bookings must be kept securely
→ Authentication : Unauthorised users should not be allowed to access the application.
→ Portability : The application must be deployable in all platforms.

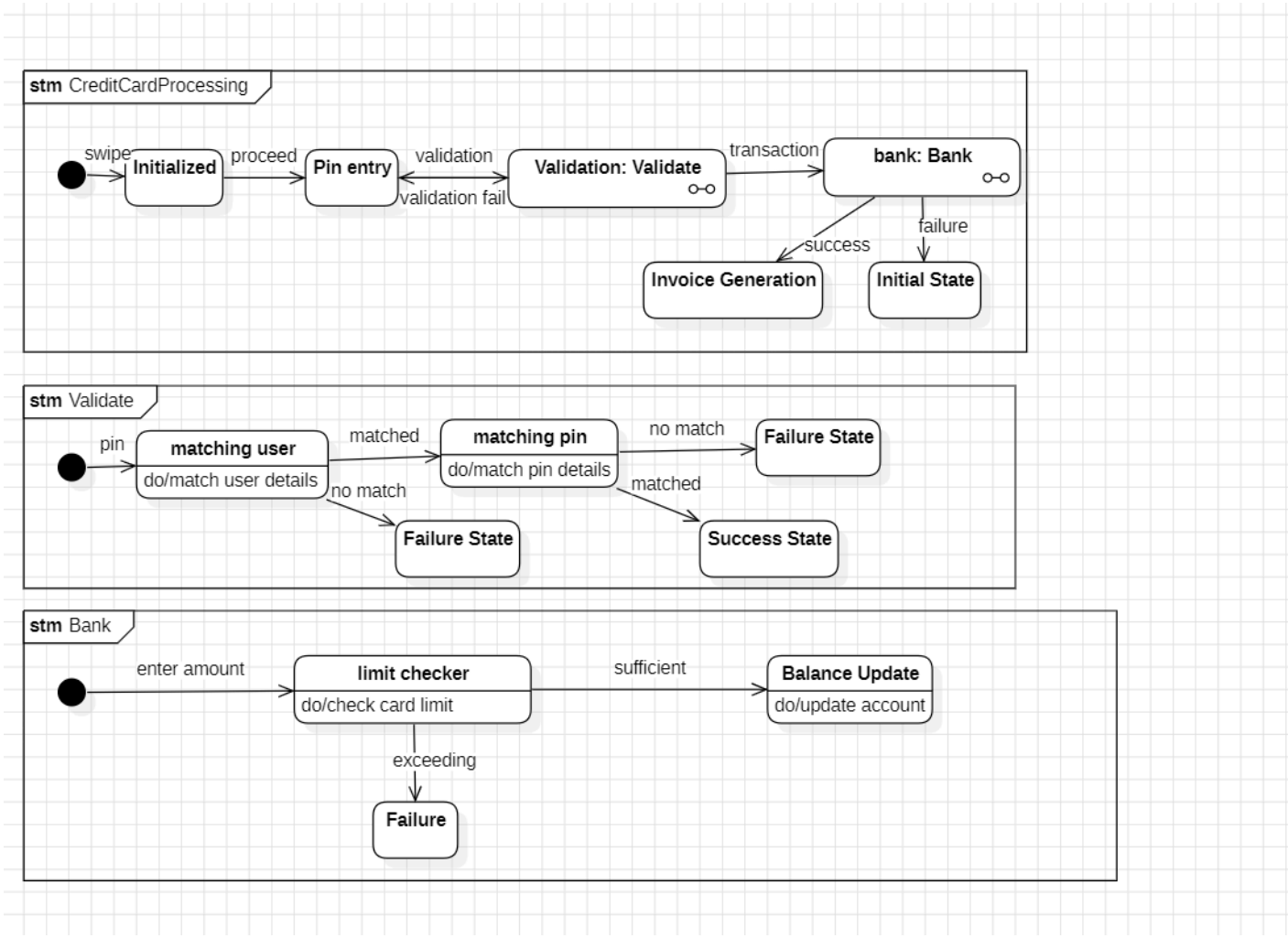
Class Diagram



Description:

- Customer - Class indicative of customer using the card.
- Bank - Indicative of the bank of the credit card.
- Card - Indicative of the card used by the customer.
- Transaction - Indicative of the transaction that occurs on using credit card.
- Generalization - VISA, Master inherit properties from Card.
- Association Class - Payment Gateway is the association class associating Card and Transaction.

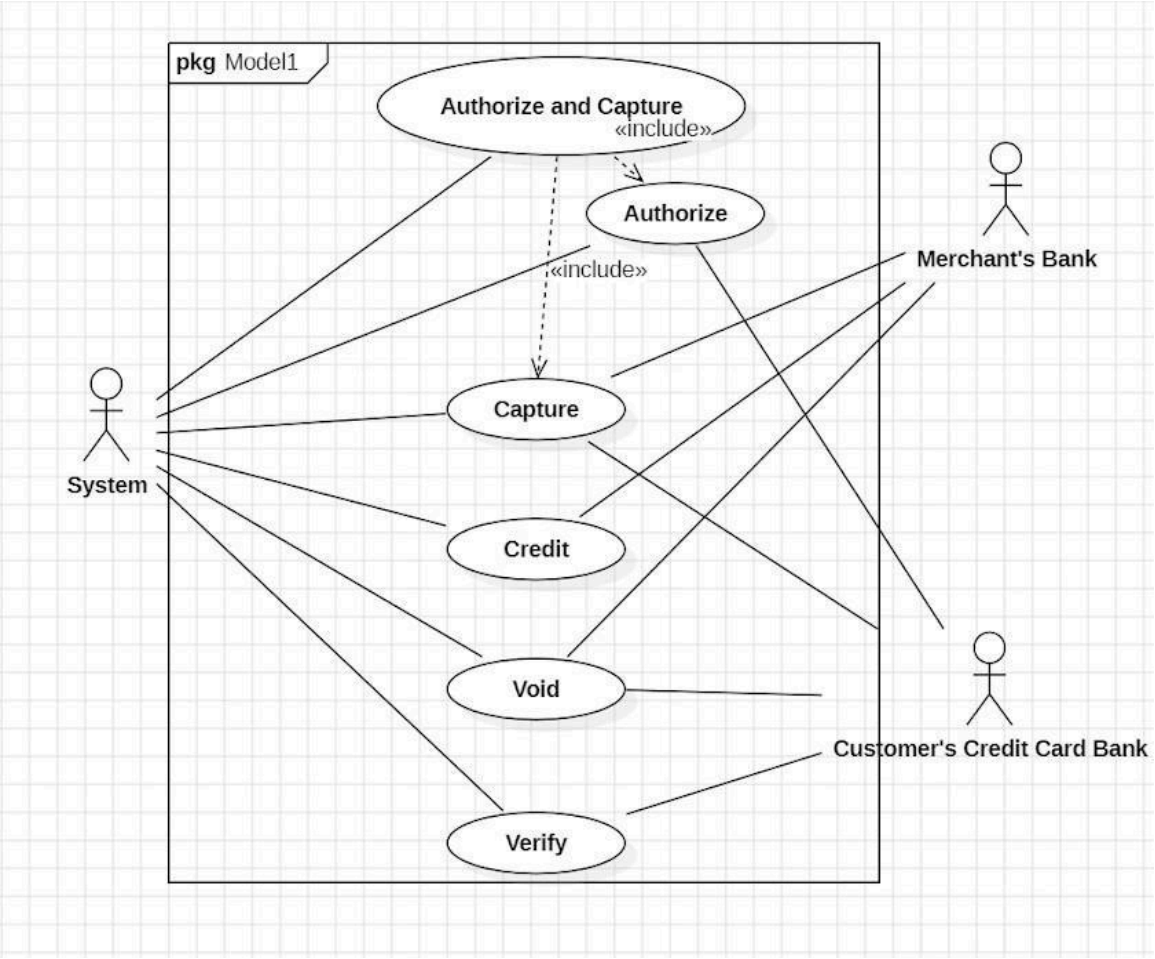
State Diagram



Description:

- The state diagram represents a CreditCardProcessing system with two submachines, Validate and Bank.
- The main system begins with the Initialized state (card swipe) and transitions to Pin Entry, followed by validation via the Validate submachine.
- If successful, it proceeds to the Bank submachine for transaction processing. The Validate submachine handles user and PIN matching, leading to success or failure.
- The Bank submachine checks transaction limits and updates balances, with transitions for success or failure. Successful transactions lead to Invoice Generation, completing the process.

Use Case Diagram



Description:

Actors Involved: System, Merchant's Bank, and Customer's Credit Card Bank.

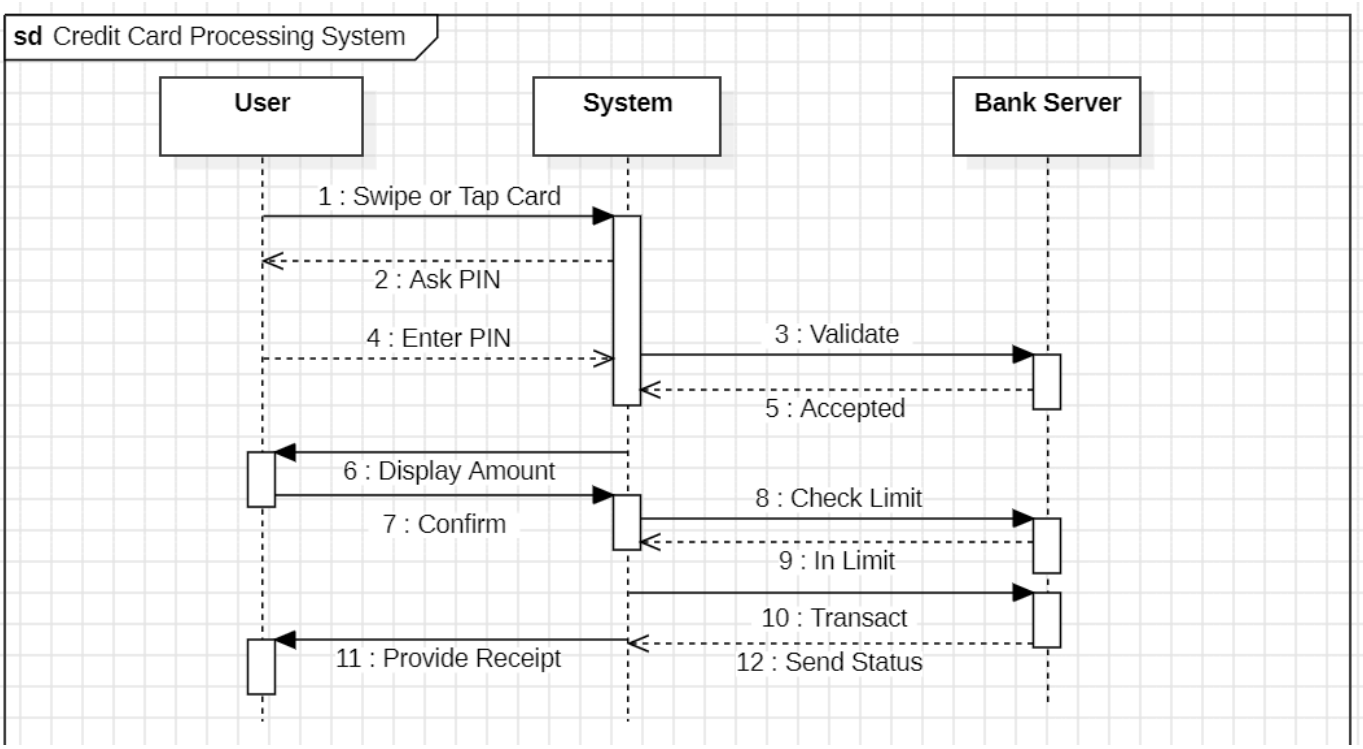
System: Initiates and manages multiple use cases, including:

- Authorize and Capture: A composite process that includes Authorize and Capture operations.
- Authorize: Validates the payment with the Customer's Credit Card Bank.
- Capture: Completes the payment process by confirming the transaction with the Merchant's Bank.
- Credit: Handles refunding or crediting the customer's account.
- Void: Cancels an authorization or transaction.
- Verify: Verifies the cardholder's information.

Merchant's Bank: Processes authorization and captures funds from the transaction.

Customer's Credit Card Bank: Validates the authorization and ensures sufficient funds for the transaction.

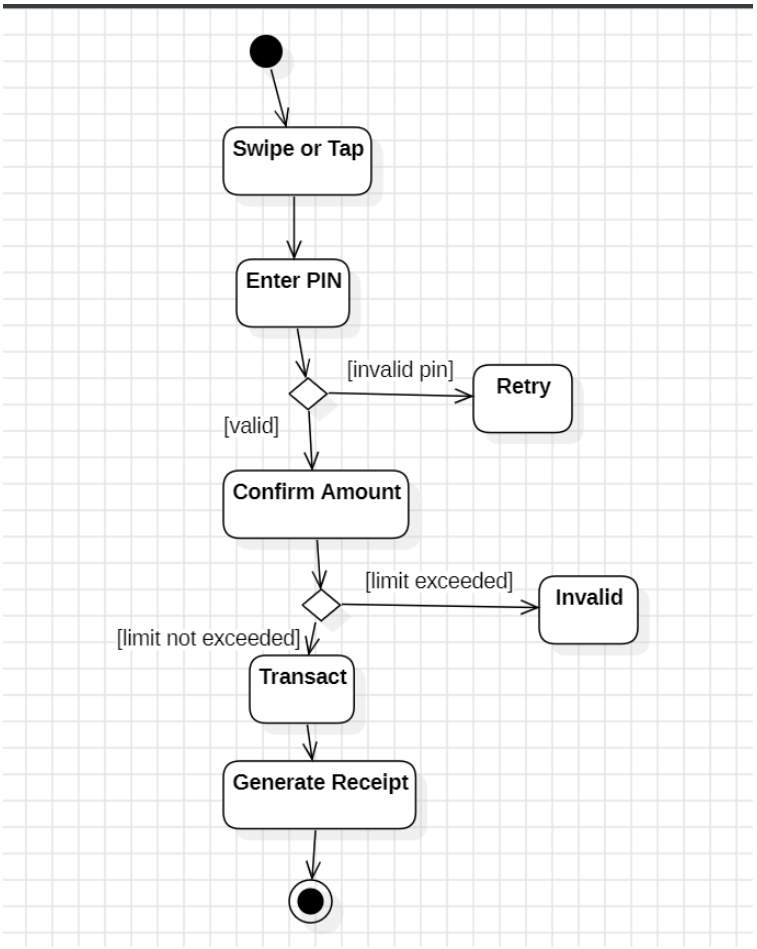
Sequence Diagram



Description:

- The user swipes or taps their card.
- The system asks the user to enter the PIN.
- The user enters the PIN, and the system validates it with the bank server.
- The bank server responds with acceptance if the PIN is valid.
- The system displays the transaction amount to the user.
- The user confirms the amount.
- The system checks the credit limit with the bank server.
- If the limit is sufficient, the bank server processes the transaction.
- The transaction status is sent back to the system.
- The system provides a receipt to the user.

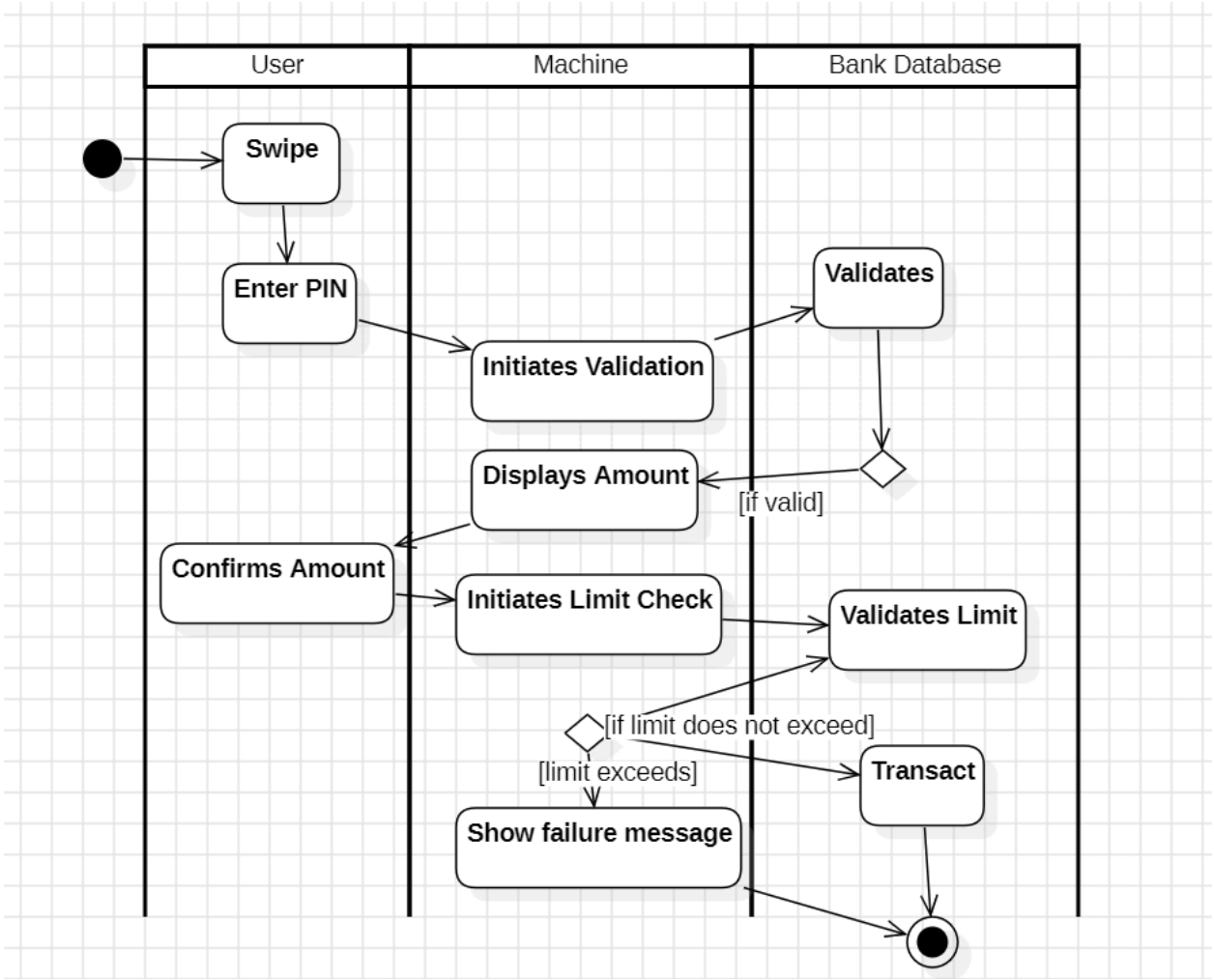
Activity Diagram



Description:

- The user swipes or taps the card.
- They enter the PIN. If the PIN is right the user is prompted to confirm if the amount displayed is right.
- Further if the limit has not been reached then the transaction happens and a receipt is generated else the transaction is invalid.

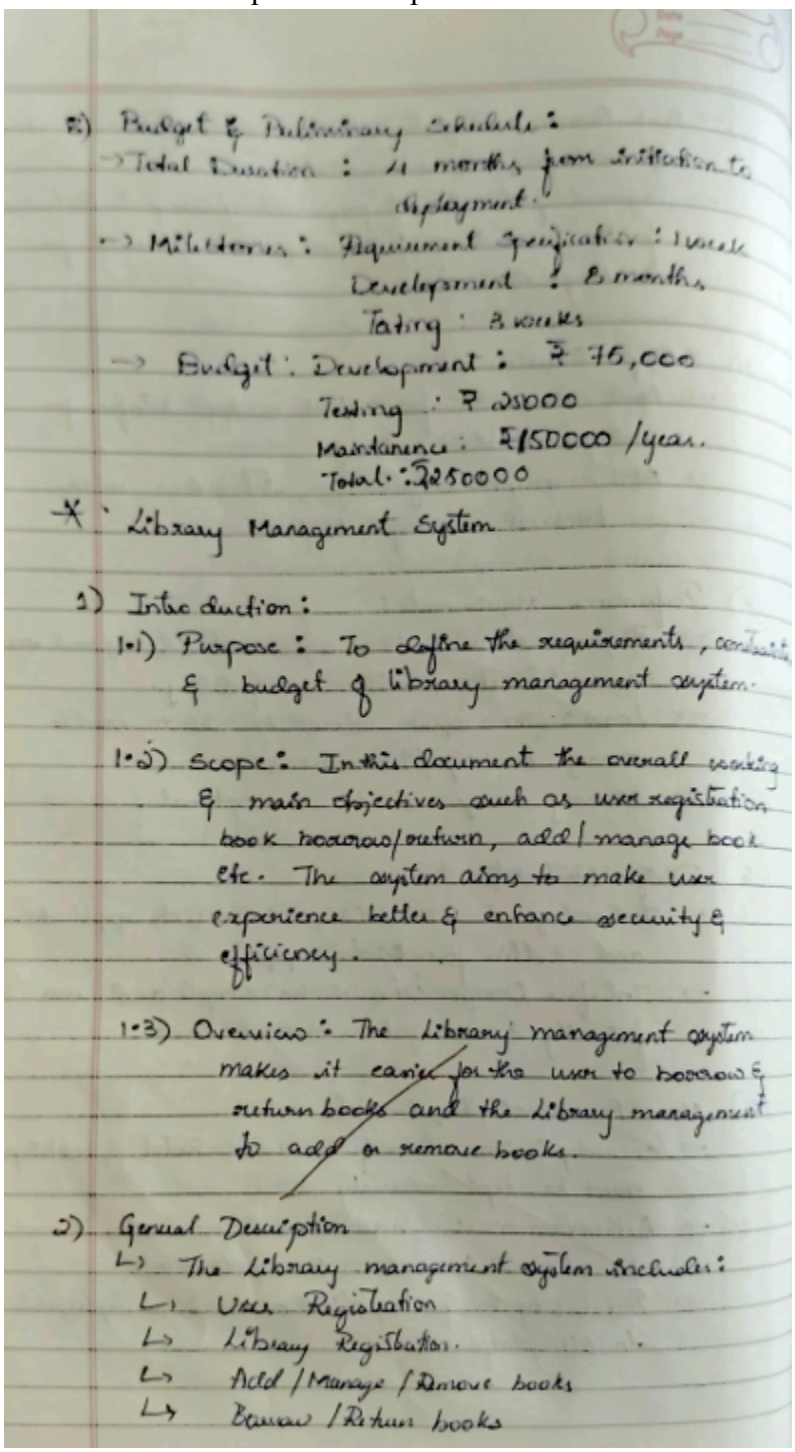
Activity Diagram using Swimlane



There are 3 swimlanes namely User, Machine and the Bank Database.

3. Library Management System

SRS - Software Requirements Specification



5) Budget & Preliminary schedule:
-> Total Duration : 11 months from initiation to deployment.
-> Milestones: Requirement Specification : 1 week
Development : 8 months
Testing : 2 weeks
-> Budget: Development : ₹ 75,000
Testing : ₹ 25000
Maintenance : ₹150000 /year.
Total : ₹250000

X Library Management System

1) Introduction:
1.1) Purpose : To define the requirements, constraints & budget of library management system.

1.2) Scope : In this document the overall working & main objectives such as user registration book borrow/return, add/manage book etc. The system aims to make user experience better & enhance security & efficiency.

1.3) Overview : The Library management system makes it easier for the user to borrow & return books and the library management to add or remove books.

2) General Description
↳ The Library management system includes:
↳ User Registration
↳ Library Registration.
↳ Add / Manage / Remove books
↳ Borrow / Return books

↳ Borrow History
↳ Payment feature.
↳ Reminder feature

3) Functional Requirements
→ User/Library Registration : User/Library should register to access the application features.
→ Add/Manage Books : Registered Library will be able to add/manage/remove books available in the library.
→ Borrow/Return books : Registed users can borrow/return books from library & the book count or availability will be updated accordingly.
→ Borrow History : Both User & Librarian will be able to view the history of the books borrowed & returned
→ Payment : It should be included for the users to pay the amount for borrowed books
→ Reminder feature : A notification will be sent when the return date is nearing.

4) Interface requirements
→ Application Interface : A website with all fatures implemented
→ API - For payment & interacting with backend

5) Performance requirements
→ Response time - The system should respond immediately to user interactions
→ Load - The system should be able to handle many transactions at once without being crashed.

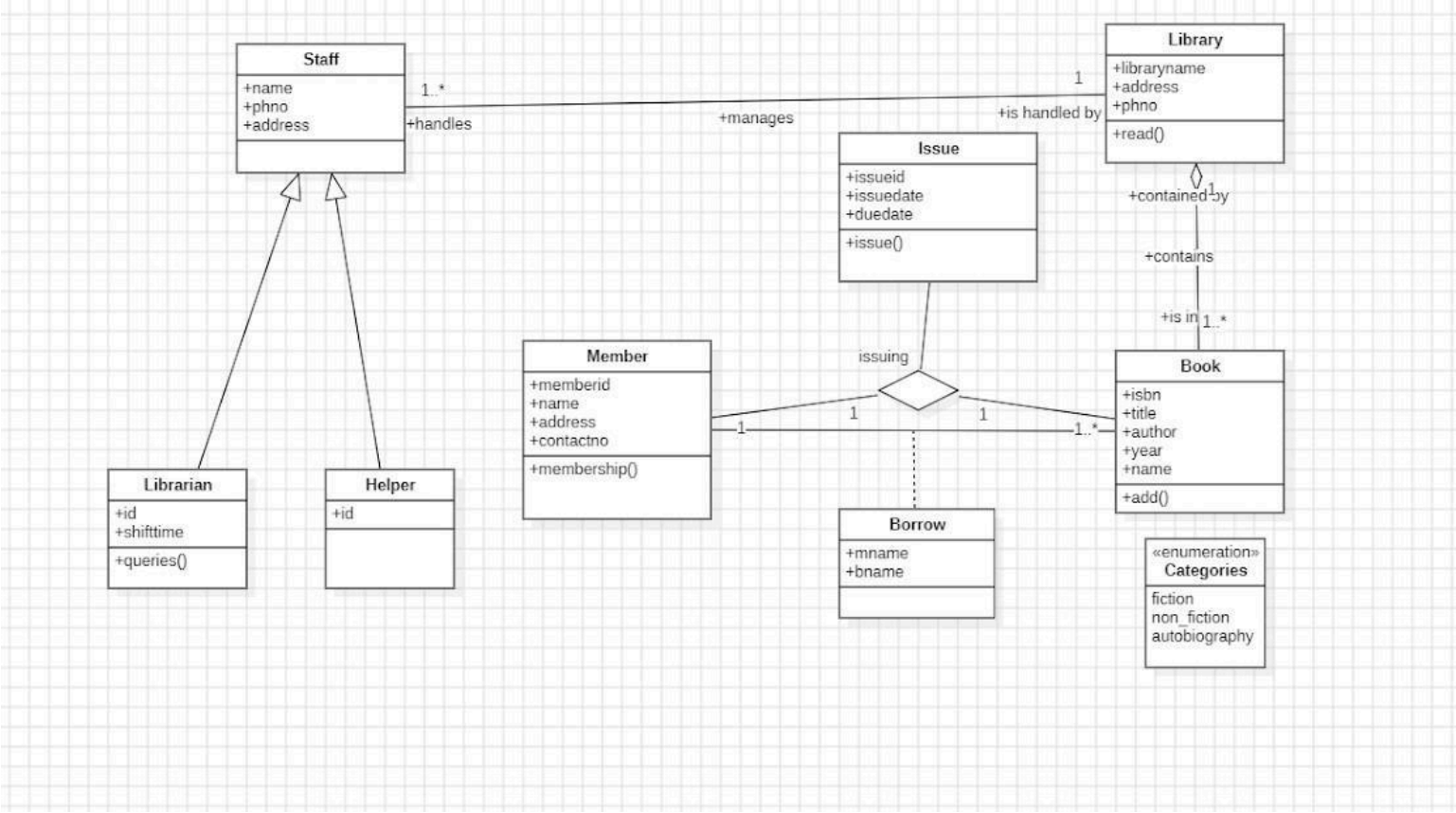
6) Design Constraints :
→ Technologies : React for frontend, Node.js for backend & MongoDB for database
→ Interface design should be clean & responsive.

7) Non-functional Requirements
→ Security : The information of user & library & books must be stored securely
→ Authentication : Unauthorised user cannot access the application features.
→ Reliability : The software features & quality must be updated all the time.
→ Scalability : The software must be adaptable for future growth.

8) Preliminary Schedule & budget
→ Total Duration : 2 Months
→ Milestones : Requirement specification : 1 week
Development : 1 Month 2 weeks
Testing : 1 week
→ Budget : Development : ₹ 500 00
Testing : ₹ 20,000
Maintenance : ₹ 50 000 /yr.
Total ₹ 120000

4/9/2024

Class Diagram

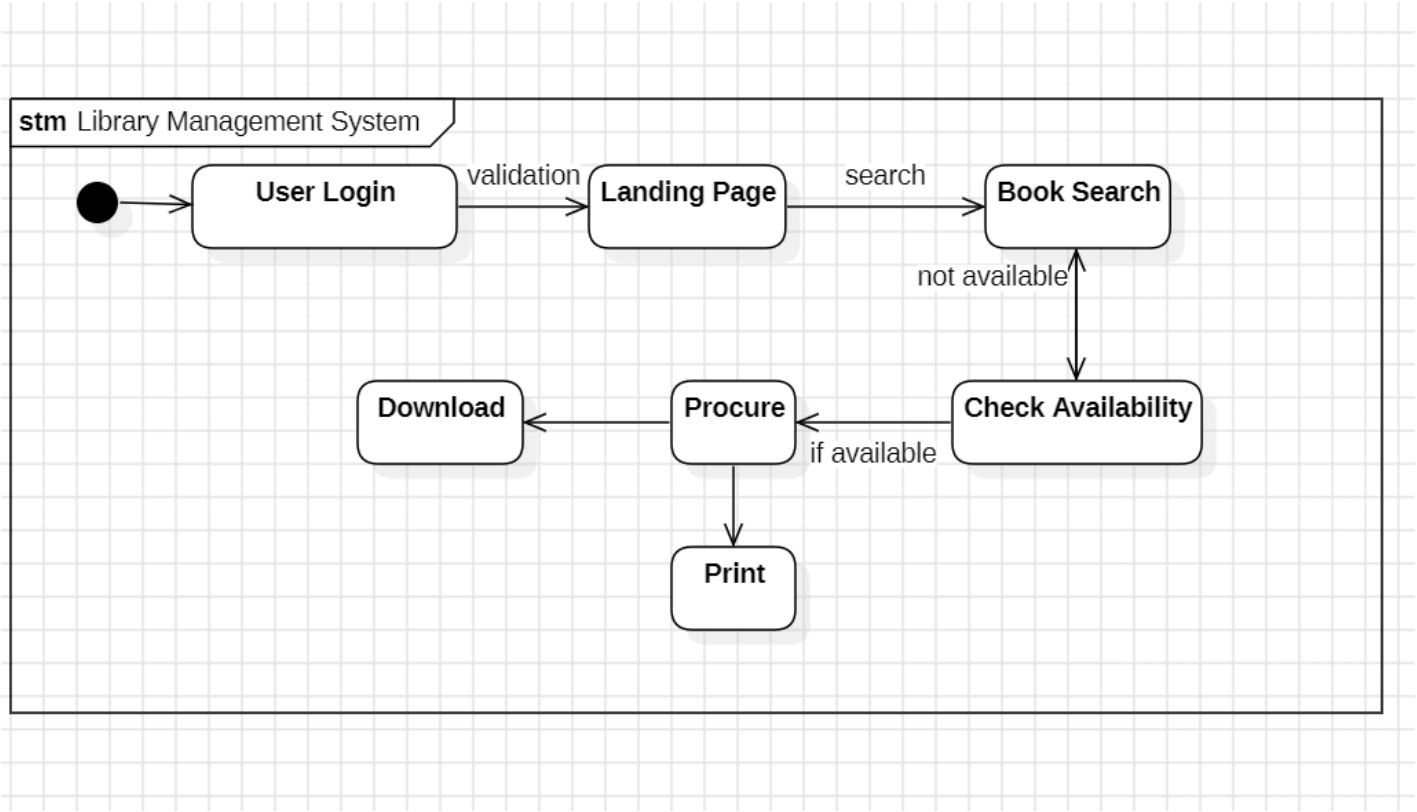


Description:

- Staff Class: Represents library staff with attributes such as name, phone number, and address. It has generalization with subclasses Librarian (with shift time and queries functionality) and Helper, which inherit its properties.
- Library Class: Contains attributes like library name, address, and phone number. It is associated with the Staff class through aggregation, as the library is "handled by" one or more staff members.
- Book Class: Represents books with attributes such as ISBN, title, author, year, and category. It has a composition relationship with the Library class, as books are a part of the library and cannot exist independently of it.
- Member Class: Represents library members with attributes like member ID, name, address, and contact number. Members are associated with the Borrow class, which connects them to books.
- Issue Class: Manages the issuing of books with attributes like issue ID, issue date, and due date. It acts as an intermediary between Member and Book, forming an association.
- Borrow Class: Represents the act of borrowing, connecting Member and Book. It contains attributes for member name and book name.

- Categories: Represent an enumeration for book classifications (fiction, non-fiction, autobiography).
- Through these relationships, the diagram incorporates generalization (Staff to Librarian/Helper), composition (Library to Book), and aggregation (Library to Staff).

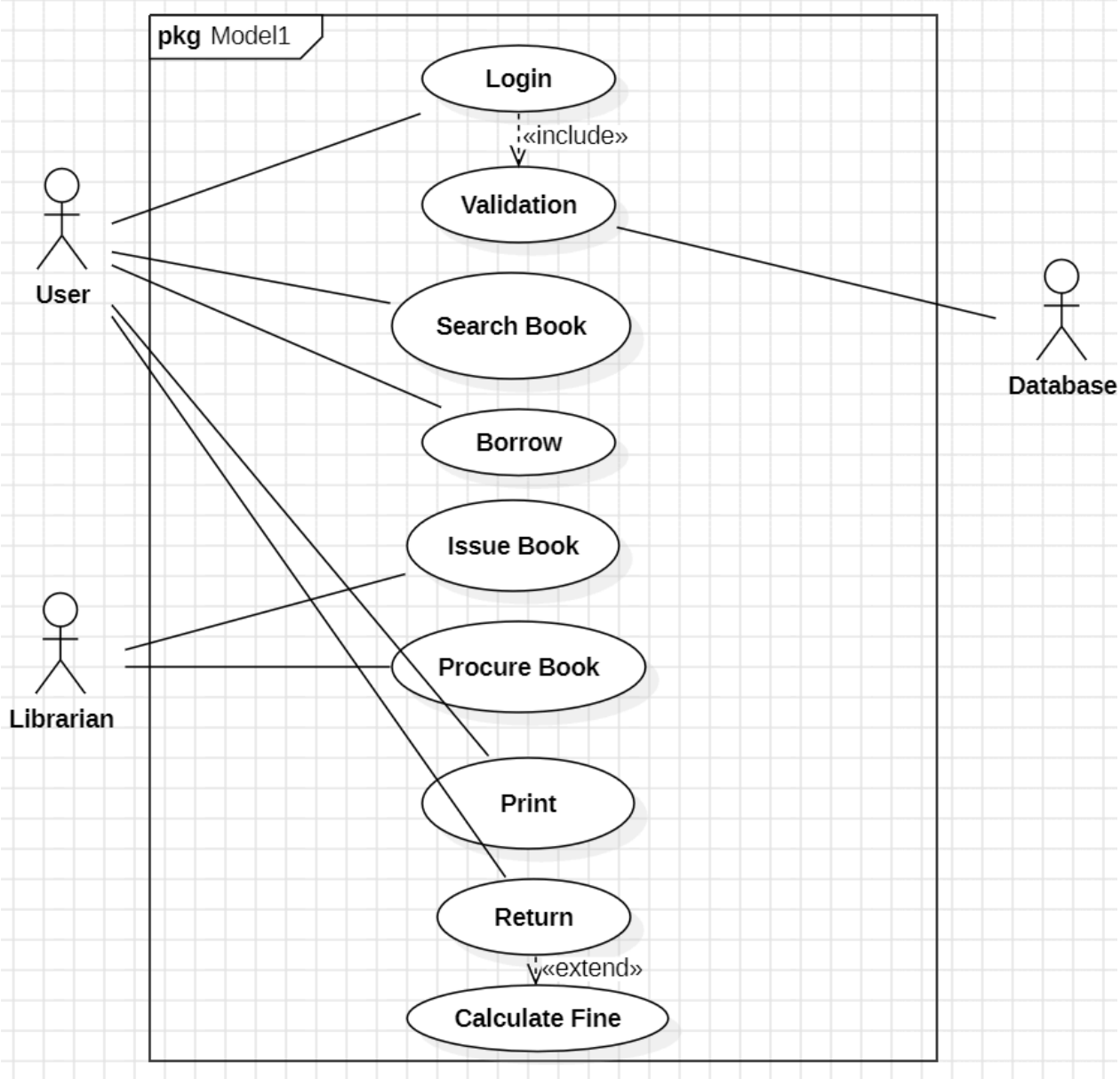
State Diagram



Description:

- The diagram represents a Library Management System with states for user interaction. The process starts at User Login, followed by validation and transition to the Landing Page.
- Users can search for books via the Book Search state, which leads to Check Availability. If the book is available, it transitions to Procure for obtaining the book, followed by options to Print or Download.
- If the book is not available, it loops back to Book Search for further searches.

Use Case Diagram



Description:

Actors Involved: User, Librarian, and Database.
User:

- Login: The user logs into the system, which includes Validation of their credentials against the Database.
- Search Book: Allows the user to search for books in the library's database.

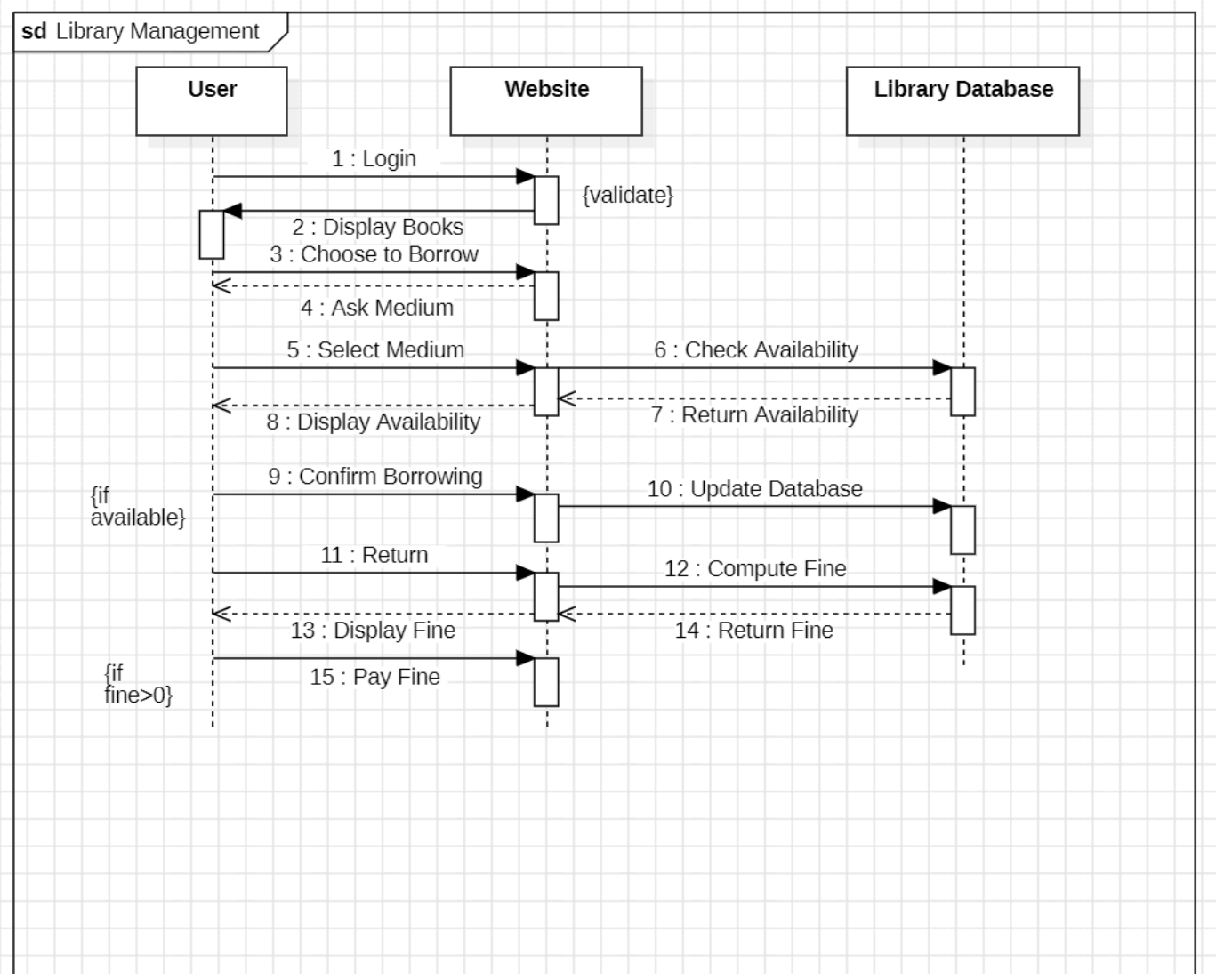
- Borrow: Users can borrow books, which triggers the Issue Book use case managed by the Librarian.
- Return: Handles the process of returning borrowed books and extends to Calculate Fine in case of overdue returns.

Librarian:

- Issue Book: Oversees the issuance of books to users.
- Procure Book: Handles the procurement of new books for the library.
- Print: Prints reports or details as needed for administrative purposes.

Database: Supports all the system's use cases, such as storing user information, book details, and transaction records.

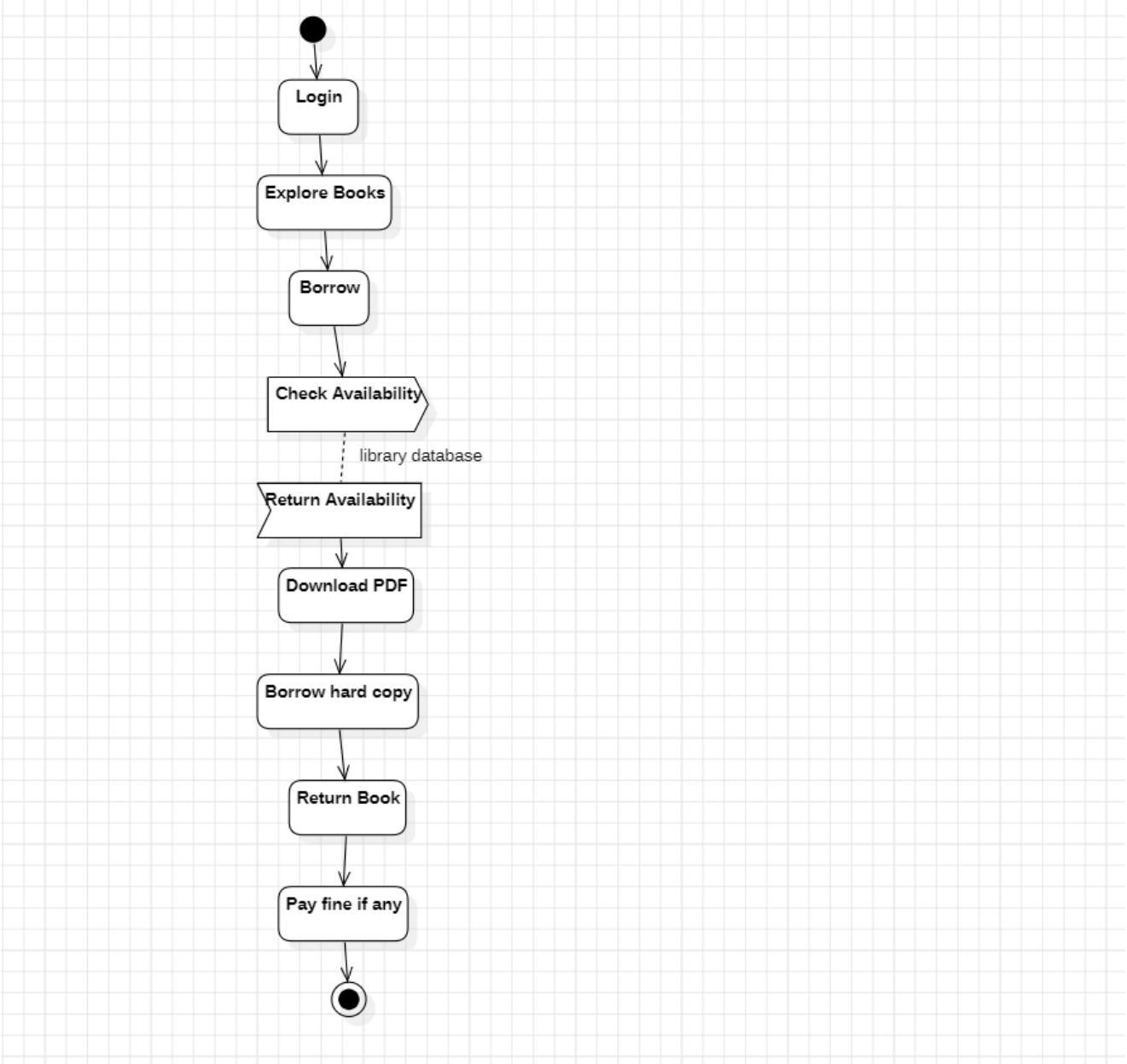
Sequence Diagram



Description:

Login: The user logs into the library management system via the website, and the system validates the user credentials using the library database.
Display Books: After successful login, the website displays a list of books available for borrowing.
Choose to Borrow: The user selects a book they wish to borrow.
Ask Medium: The website prompts the user to select the borrowing medium (e.g., physical or digital).
Select Medium: The user selects their preferred medium for borrowing.
Check Availability: The website checks the book's availability by querying the library database.
Return Availability: The library database returns the availability status of the selected book.
Display Availability: The website displays the availability information to the user.
Confirm Borrowing: If the book is available, the user confirms their intention to borrow it.
Update Database: The website updates the library database to reflect the borrowed status of the book.
Return: When the user is done with the book, they return it through the website.
Compute Fine: The website requests the library database to compute any late return fine.
Display Fine: If a fine exists, the website displays the fine amount to the user.
Pay Fine: If the fine is greater than zero, the user proceeds to pay it via the website.
Return Fine: The payment status is updated in the library database.

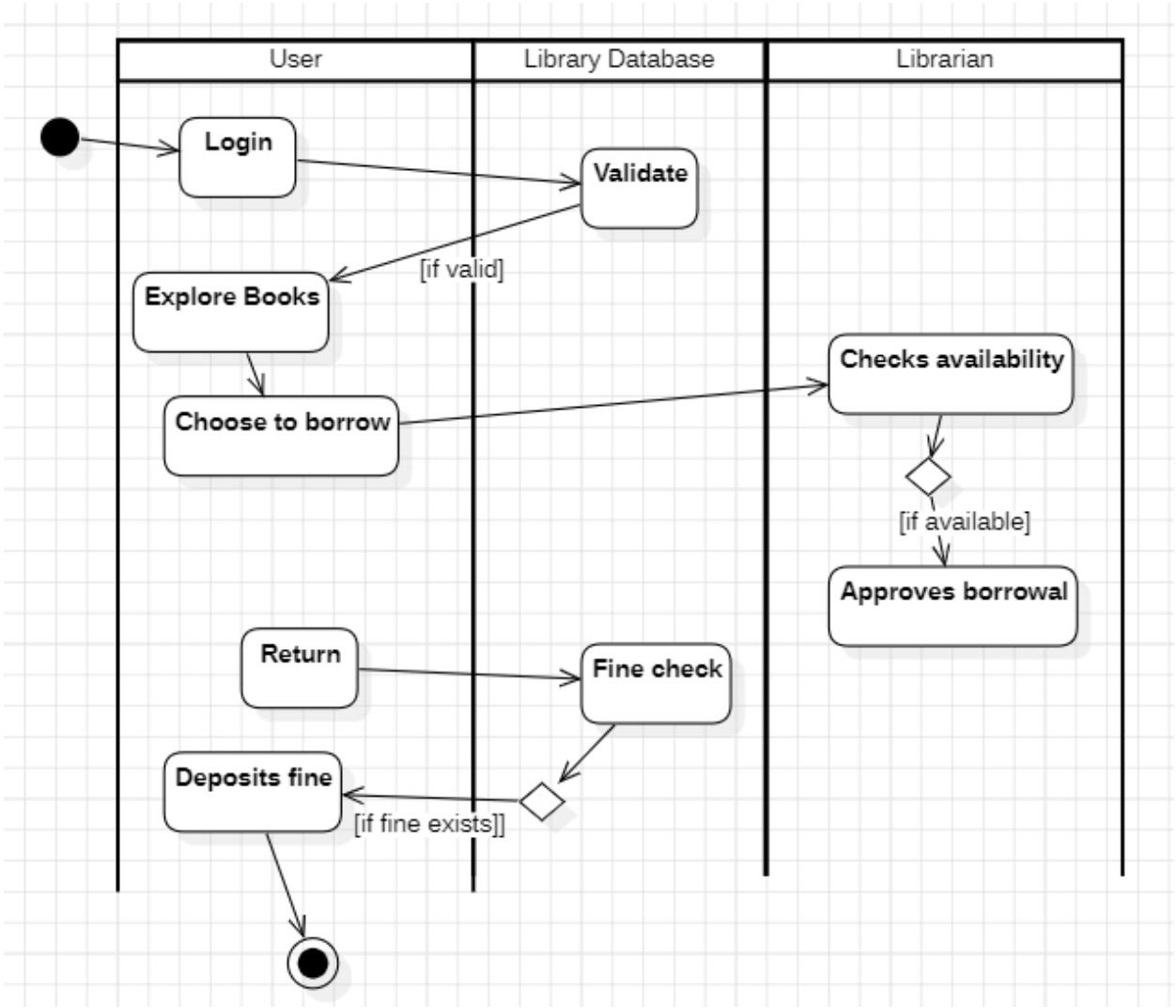
Activity Diagram



Description:

- The user logs in and explores the books.
- If they chose to borrow the availability is checked in the database and returned.
- If the book is available then the user may download a pdf and then borrow hardcopy.
- Once the user returns the book they may pay the fine if any.

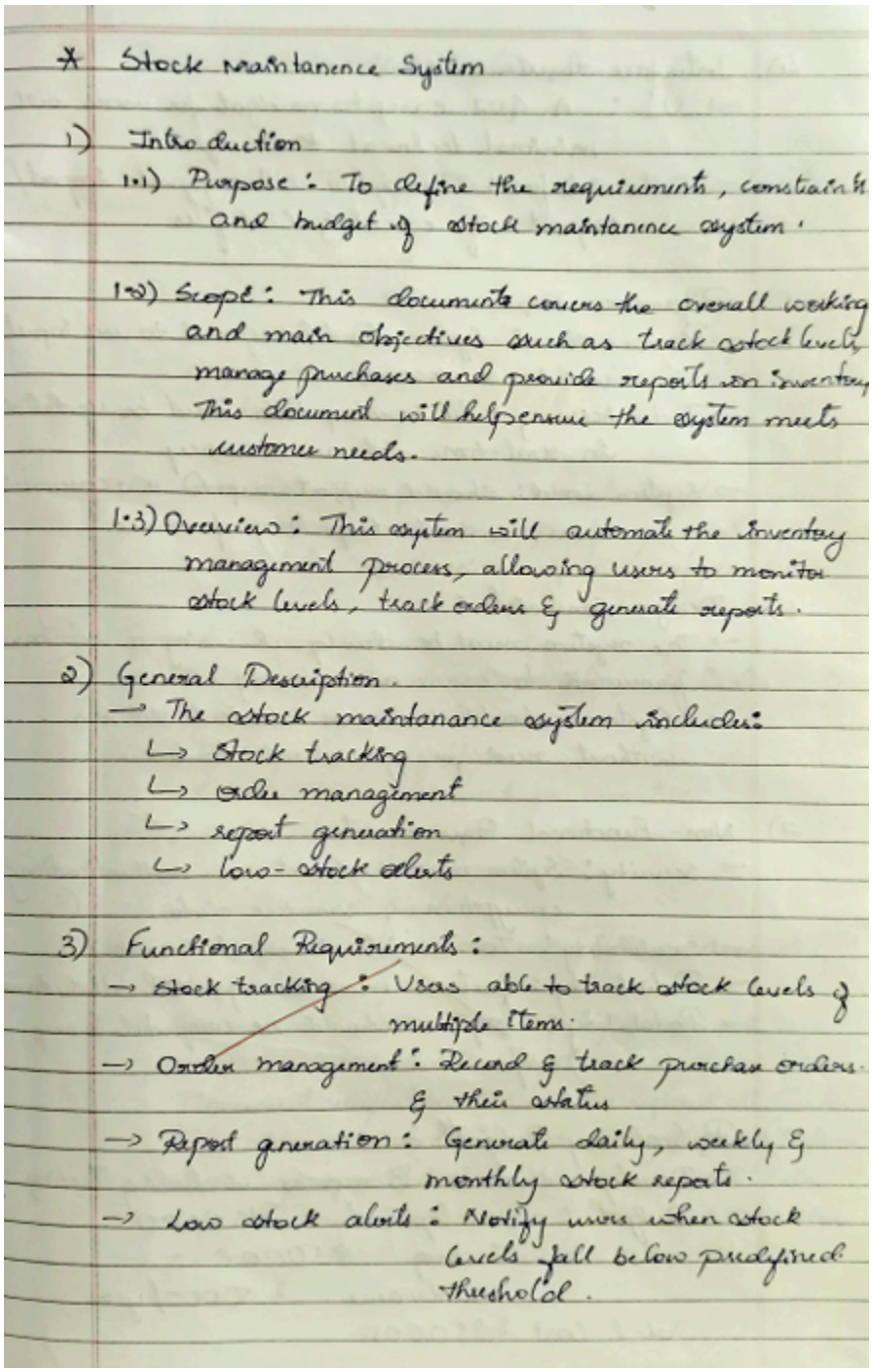
Activity Diagram using Swimlane:



There are 3 swimlanes namely User, Library Database and Librarian.

4. Stock Maintenance System

SRS - Software Requirements Specification



* Stock maintanence System

1) Introduction

1.1) Purpose : To define the requirements, constraints and budget of stock maintanance system.

1.2) Scope : This document covers the overall working and main objectives such as track stock levels, manage purchases and provide reports on inventory. This document will helpensure the system meets customer needs.

1.3) Overview : This system will automate the inventory management process, allowing users to monitor stock levels, track orders & generate reports.

2) General Description.

→ The stock maintanance system includes:

- ↳ Stock tracking
- ↳ order management
- ↳ report genration
- ↳ low-stock alerts

3) Functional Requirements :

→ Stock tracking : Users able to track stock levels of multiple items.

→ Order management : Record & track purchase orders & their status

→ Report genration : Genrate daily, weekly & monthly stock reports.

→ Low stock alerts : Notify users when stock levels fall below predefined threshold.

4) Interface Requirements:
→ UI: A GUI easy to navigate for users with minimal technical knowledge.
→ Data Interface: The system will support import/export of excel & CSV files

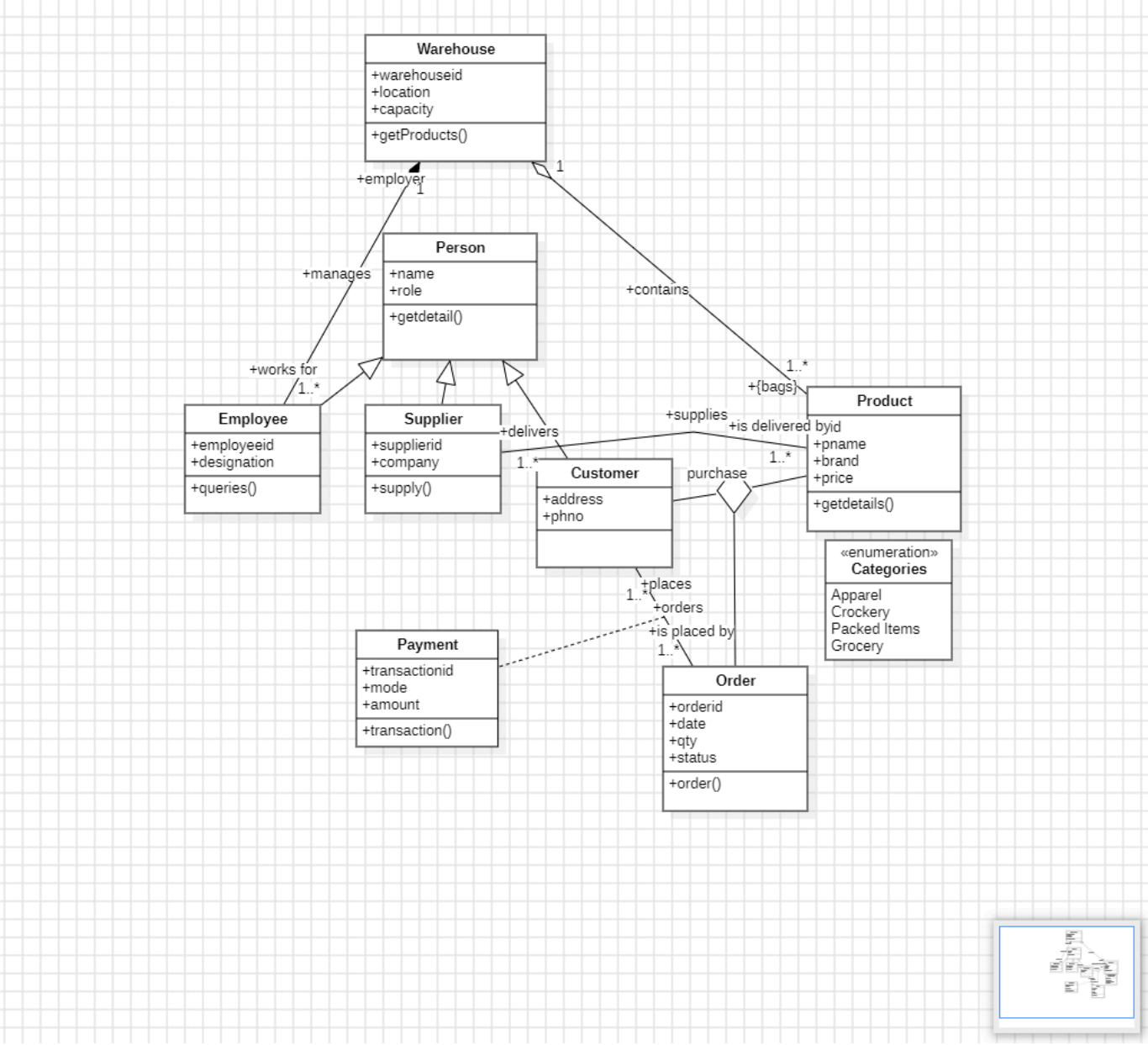
5) Performance Requirements:
→ Response time: System should respond to user inputs within 2 seconds
→ Data Accuracy: All inventory data must be updated in real-time with an accuracy
→ System load: should support upto 100 concurrent users without drop in performance.

6) Design Constraints:
→ The system must be developed using a web-based framework to ensure accessability
→ System should run on standard servers without need for specialized hardware.

7) Non-Functional Requirements:
→ Security: System must implement user authentication encryption of sensitive data
→ Scalability: To handle increased inventory
→ Reliability: Data must be updated all the time
→ Portability: System should be compatible to any platform.

8) Preliminary Budget:
→ Development time: 3 months including Testing
→ Budget: Development: ₹150000
Testing: ₹50000
Maintanance: ₹ 50000/year
Total Cost: ₹250000

Class Diagram

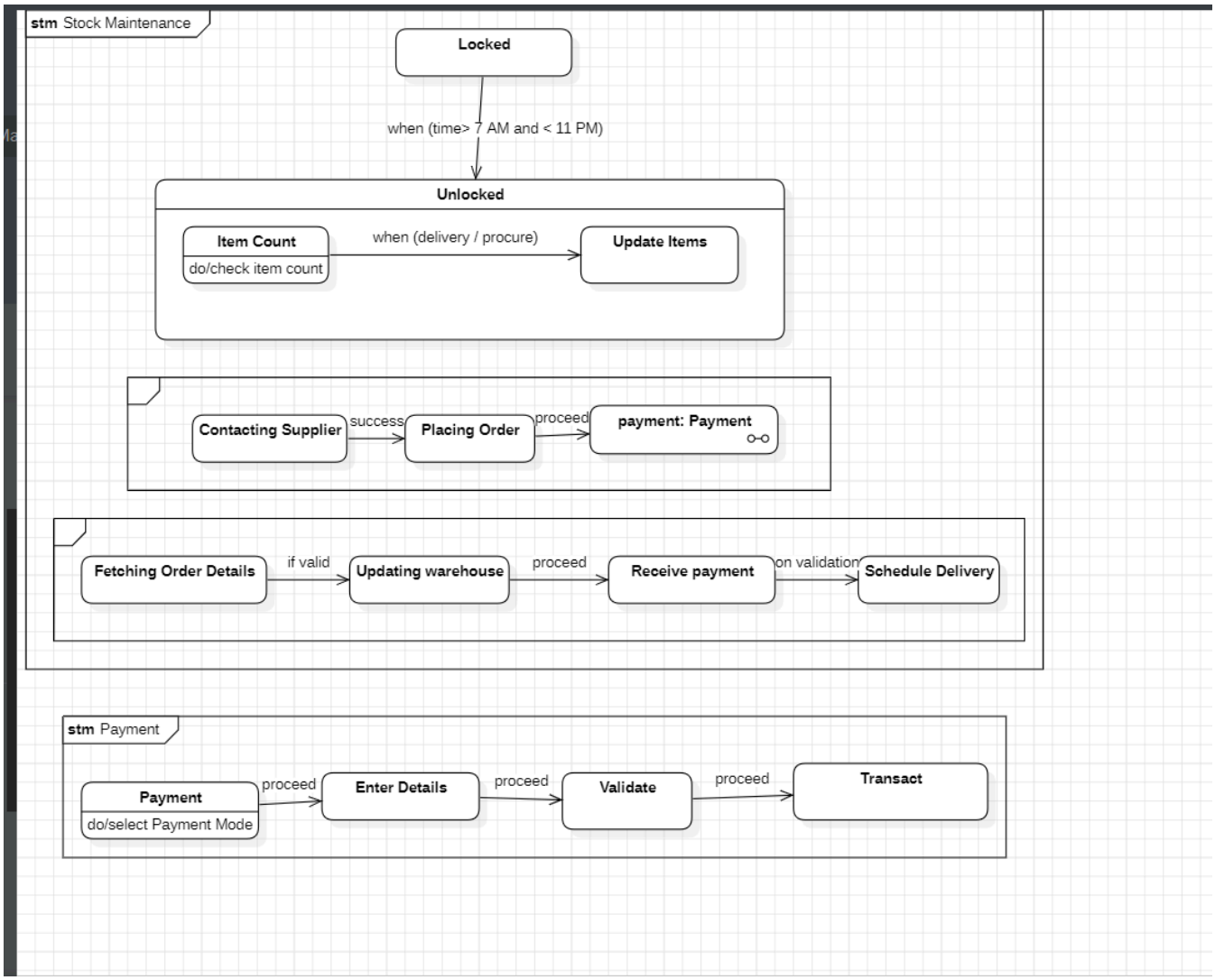


Description of Class Diagram:

- Warehouse: Represents storage facilities containing products (*composition* with Product).
- Person: General class (*generalization*) inherited by Employee, Supplier, and Customer.
- Employee: Represents staff working for the warehouse (*aggregation* with Warehouse).
- Supplier: Represents entities delivering products to the warehouse (*association* with Product).
- Customer: Represents buyers placing orders for products (*association* with Order).
- Product: Represents items stored in the warehouse, categorized by type (*composition* with Warehouse).

- Order: Represents customer purchases, linked to Product and Customer (*aggregation* with Customer).
- Payment: Captures transaction details for orders (*association* with Order).

State Diagram

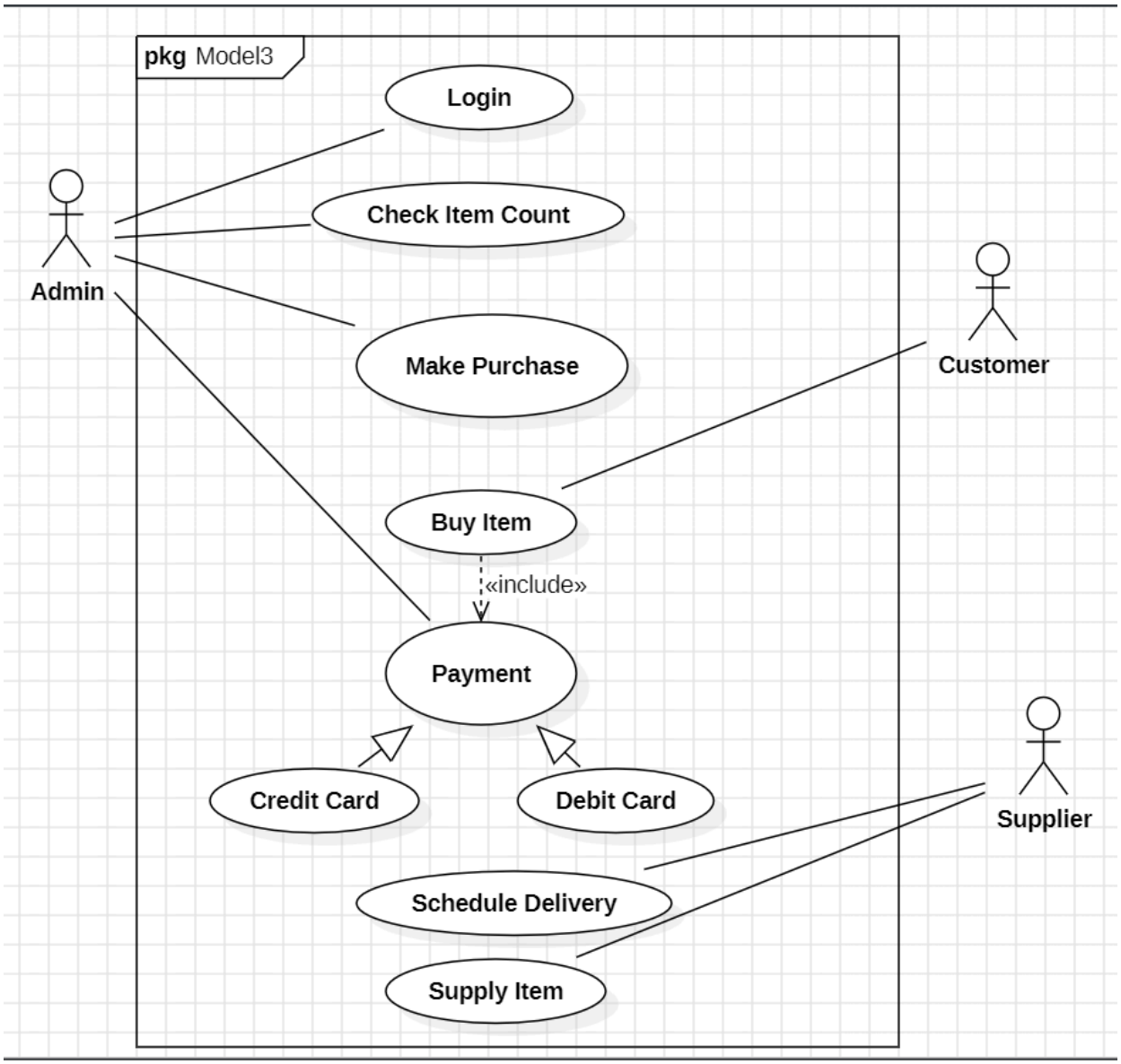


Description of State Diagram:

- The state diagram represents a Stock Maintenance system with states for managing inventory and deliveries.
- The system starts in the Locked state, operational only between 7 AM and 11 PM, transitioning to Unlocked.

- In the Unlocked state, stock is checked in Item Count, and items are updated if necessary. Processes include Contacting Supplier, Placing Order, and a Payment submachine for transaction handling.
- Upon success, it proceeds to Fetching Order Details, Updating Warehouse, and Receiving Payment, finally leading to Schedule Delivery.
- The Payment submachine includes states for selecting a mode, entering details, validating, and completing the transaction.

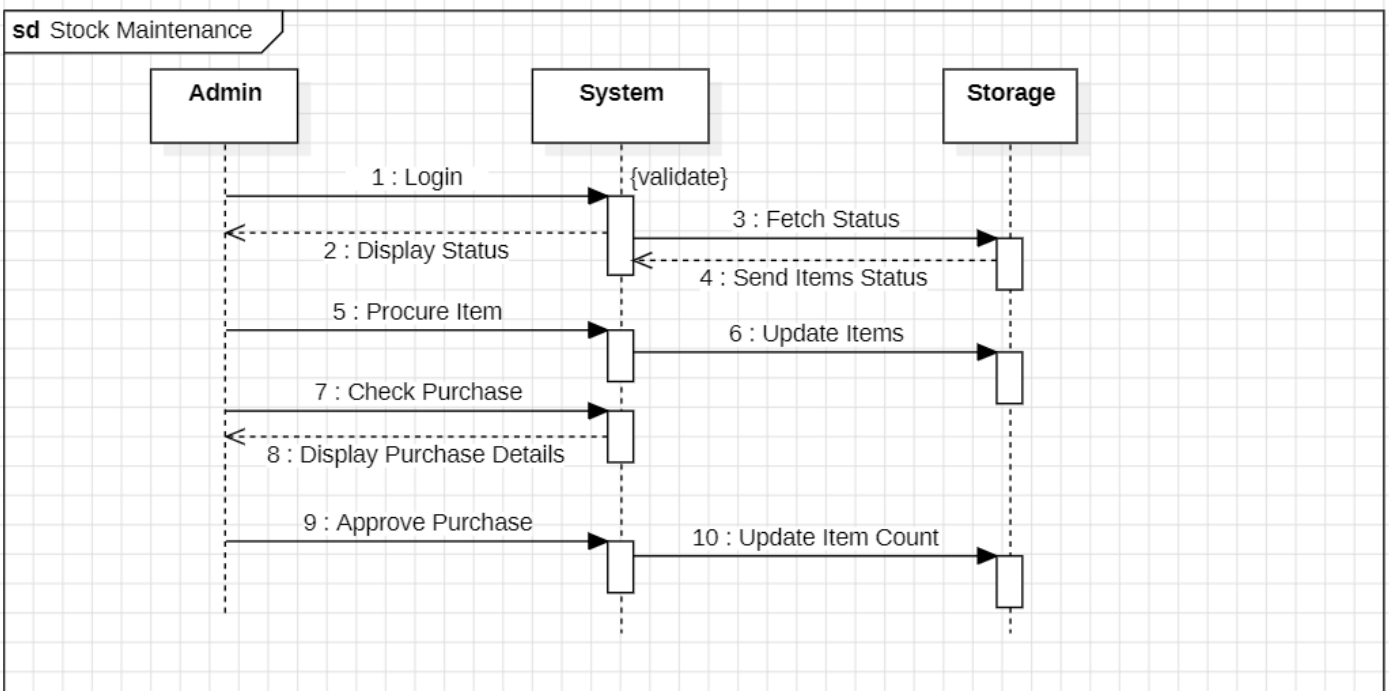
Use Case Diagram



Description of Use Case Diagram:

Actors Involved: Admin, Customer, and Supplier.
Admin: Logs into the system, checks item count, and makes purchases by buying items.
Includes: The Buy Item use case includes the Payment process, which involves selecting either credit or debit card as payment methods.
Supplier: Supplies items after the admin schedules delivery.
Extends: Delivery scheduling extends the purchase process to ensure timely item supply.

Sequence Diagram



Description:

Login: The admin logs into the stock maintenance system through the interface, and the system validates the admin's credentials.

Display Status: The system displays the current stock status to the admin.

Fetch Status: The system retrieves the stock status by querying the storage system.

Send Items Status: The storage system sends the current items' status back to the system.

Procure Item: The admin initiates the process to procure an item based on the displayed stock status.

Update Items: The system updates the stock items in the storage after procurement.

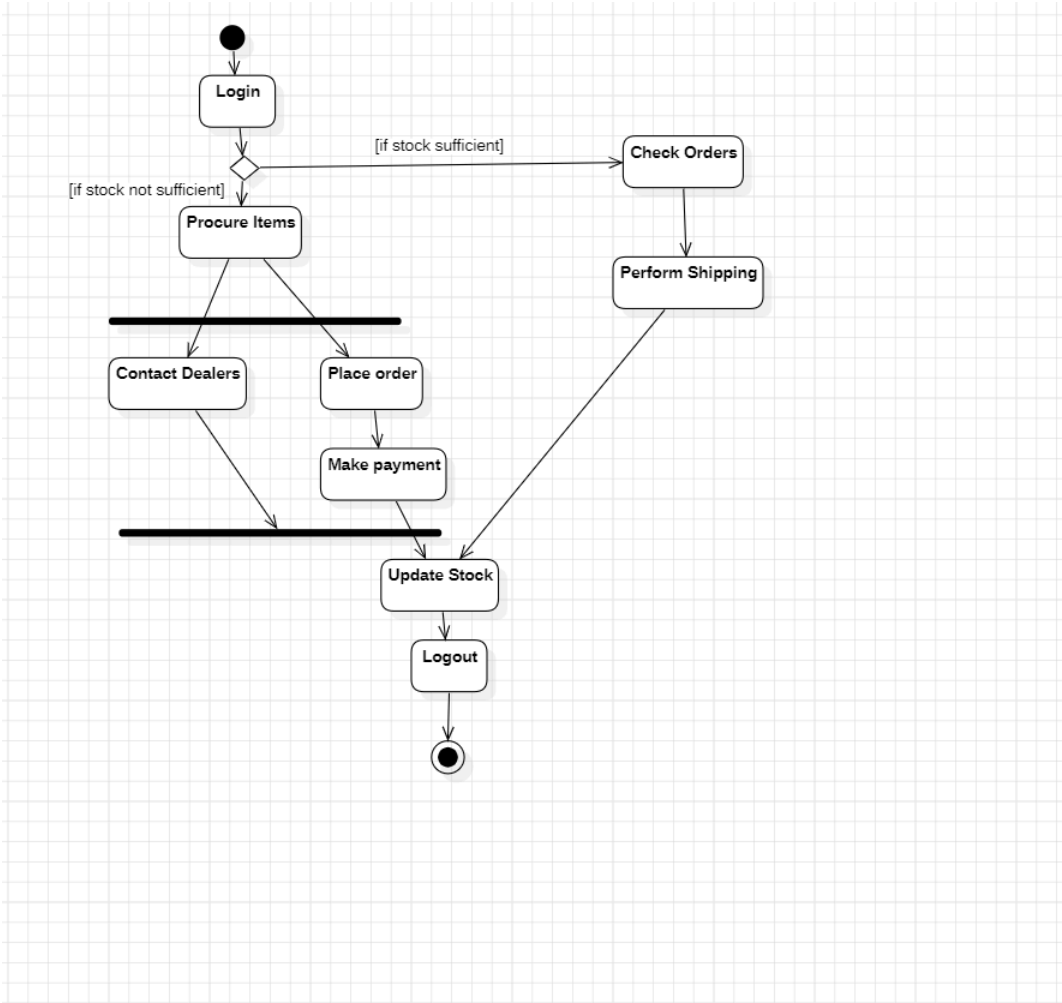
Check Purchase: The admin checks the purchase details to verify the procurement process.

Display Purchase Details: The system displays the details of the purchase to the admin.

Approve Purchase: If satisfied with the details, the admin approves the purchase.

Update Item Count: The system updates the item count in the storage system to reflect the approved purchase.

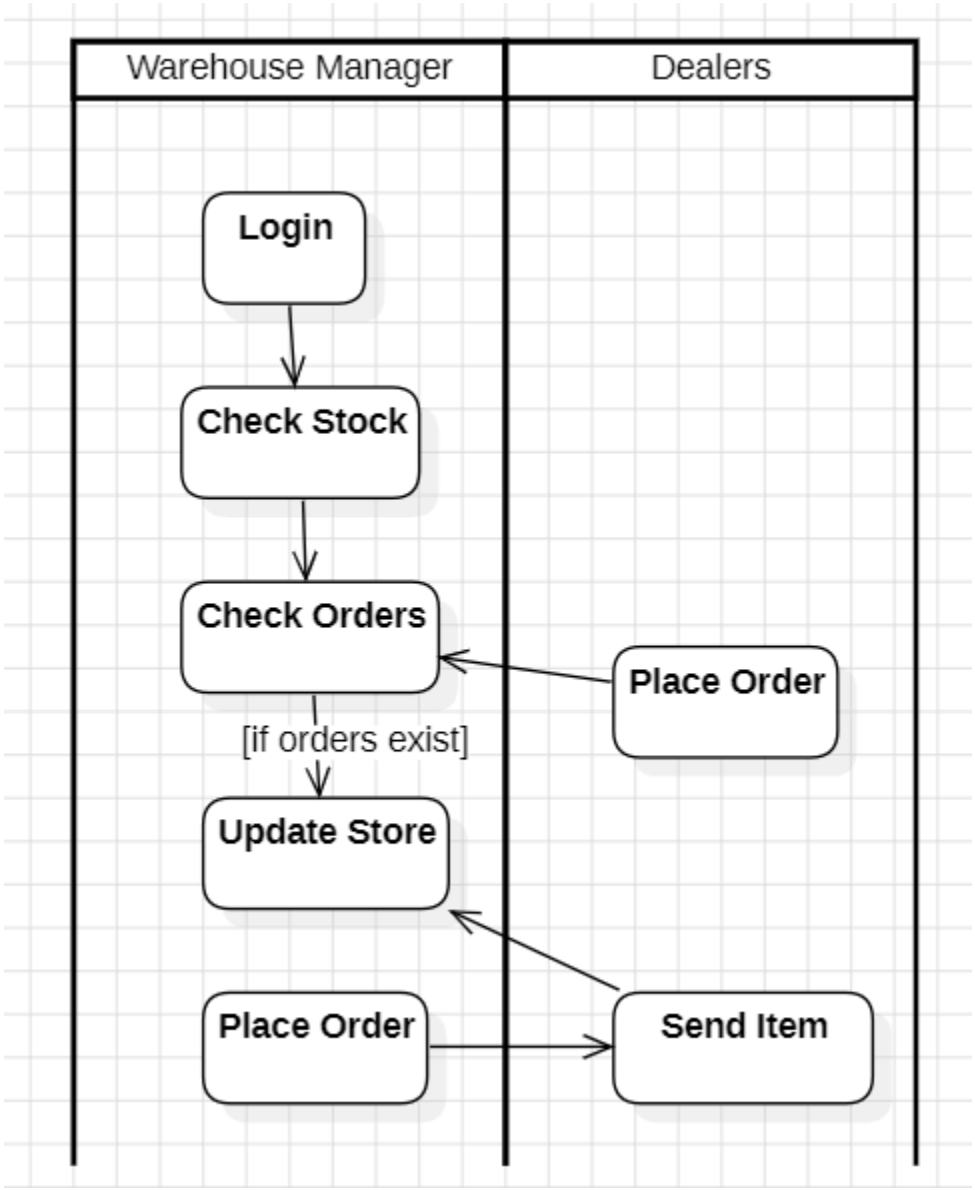
Activity Diagram



Description:

- The warehouse manager logs into the system.
- They check if there is sufficient stock. If there is, they proceed to check the orders and deliver those items.
- Otherwise they place the order by contacting the dealers and procuring the items by making payments.

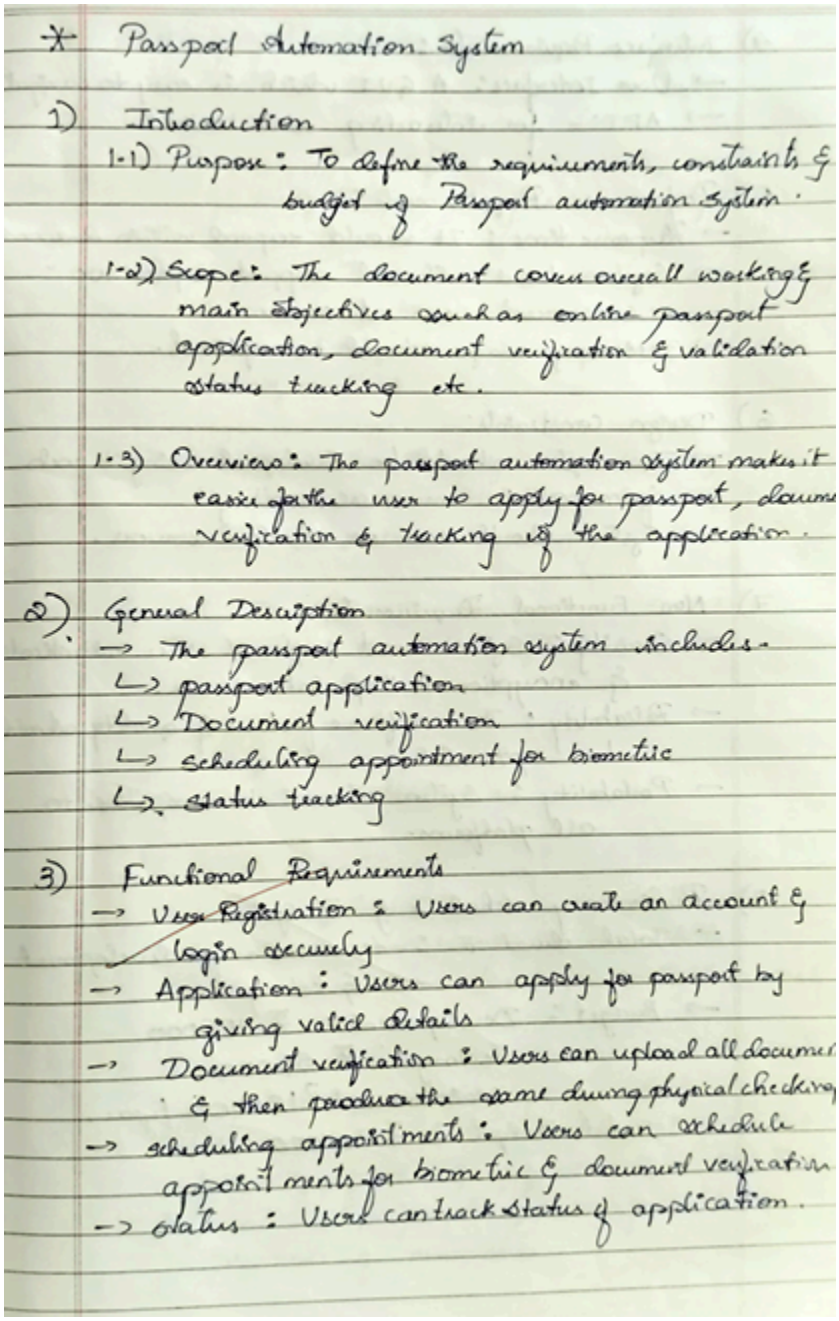
Activity Diagram using Swimlanes:



There are 2 swimlanes Warehouse Manager and Dealer.

5. Passport Automation System

SRS - Software Requirements Specification



* Passpot Automation System

1) Introduction

1-1) Purpose : To define the requirements, constraints & budget of Passpot automation system.

1-2) Scope : The document covers overall working & main objectives such as online passpot application, document verification & validation status tracking etc.

1-3) Overviews : The passpot automation system makes it easier for the user to apply for passpot, docum verification & tracking of the application.

2) General Description

→ The passpot automation system includes -
  ↳ passpot application
  ↳ Document verification
  ↳ scheduling appointment for biometric
  ↳ status tracking

3) Functional Requirements

→ User Registration : Users can create an account & login securely
→ Application : Users can apply for passpot by giving valid details
→ Document verification : Users can upload all documen & then produce the same during physical checking
→ scheduling appointments : Users can schedule appointments for biometric & document verification
→ status : Users can track status of application.

4) Interface Requirements :
→. User Interface: A GUI which is easy to navigate
→ API - for interacting with backend.

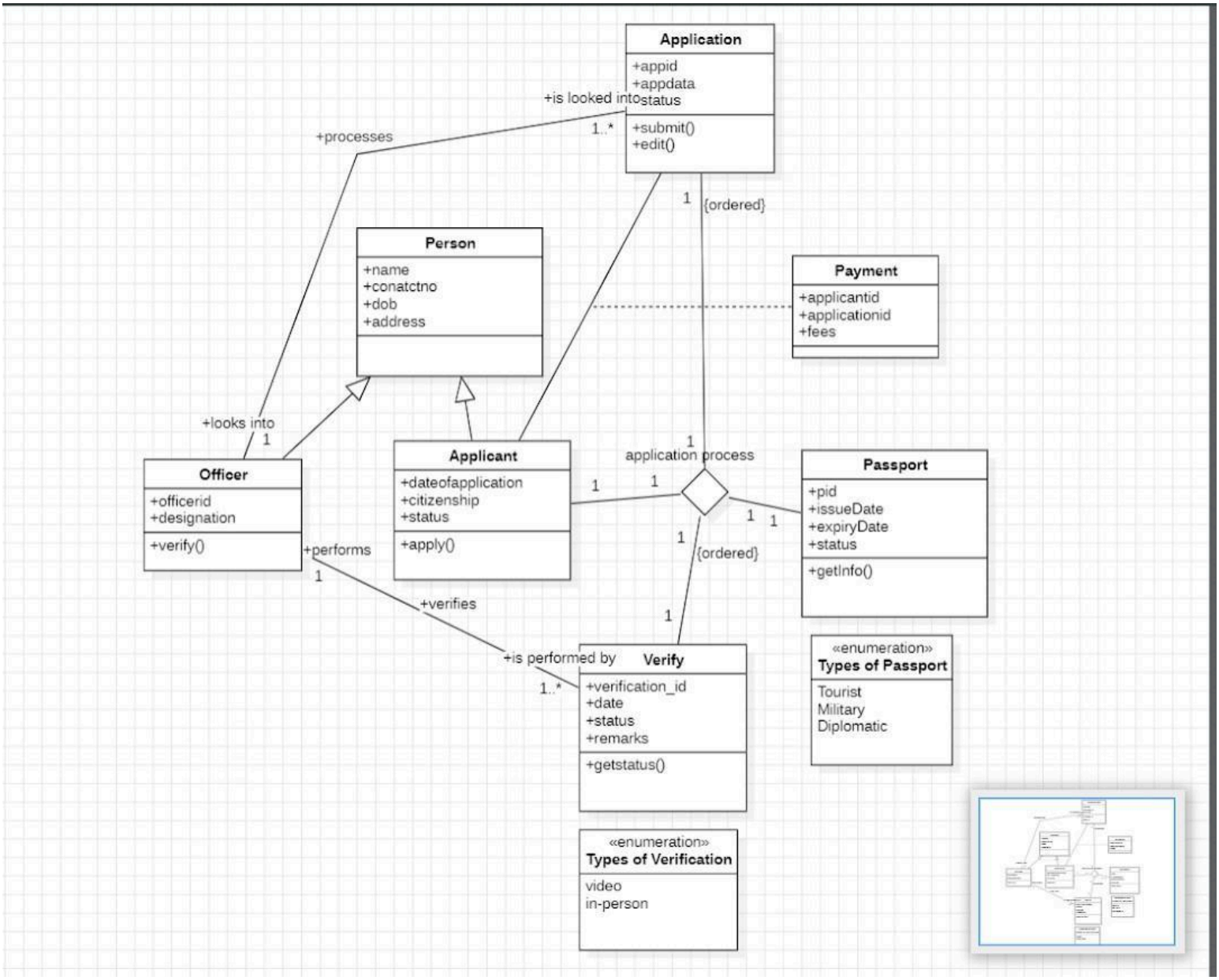
5) Performance Requirements :
→ Response time : It should respond within 2 seconds
→ System Load: Should support upto 100 concurrent users at once.
→ Memory usage should be optimal.

6) Design Constraints:
→ The system should be developed using a web framework to ensure accessability
→ System should run on standard servers.

7) Non- Functional Requirements
→ Security : System must implement user authentication & encryption to keep data secure.
→ Reliability : The software features & quality should be upto date.
→ Portability : System should be compatible to all platforms.

8) Preliminary schedule & Budget
→ Total duration : 4 months for development & testing
→ Budget : Development : ₹ 150000
Testing : ₹ 50000
Maintenance : ₹100000 /year.
Total Budget : ₹ 300000

Class Diagram

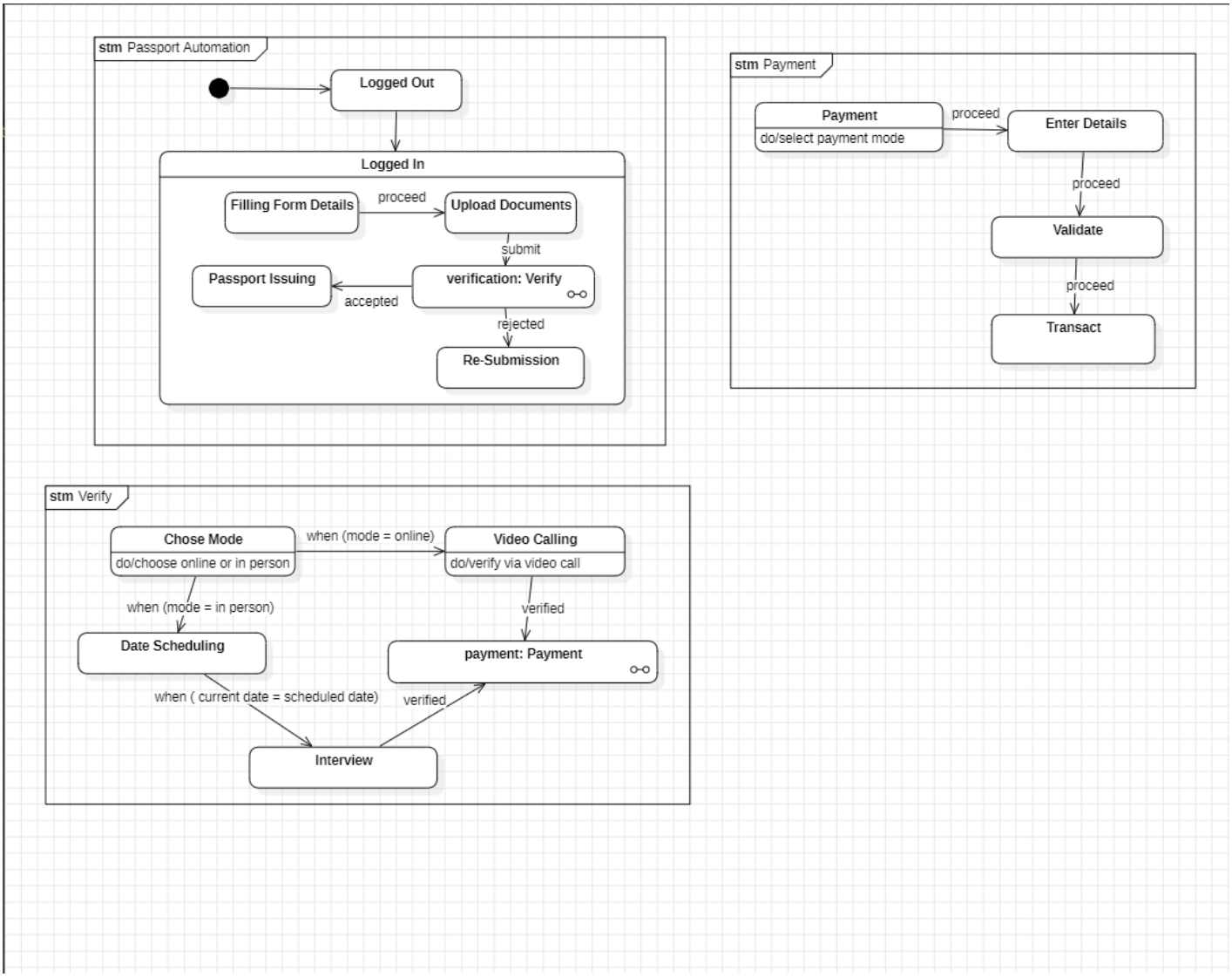


Description of Class Diagram

- Person is a general class associated with Applicant and Officer, representing entities involved in the system.
- Applicant applies for a passport, links to Application, and has attributes like citizenship and application status.
- Officer verifies applications, performing a verification process represented by the Verify class, which details the verification type (in-person or video).

- Application is associated with Payment for fees and leads to the generation of a Passport, categorized into different types (Tourist, Military, Diplomatic).
- The Passport includes essential attributes like issue and expiry dates, completing the process flow.

State Diagram

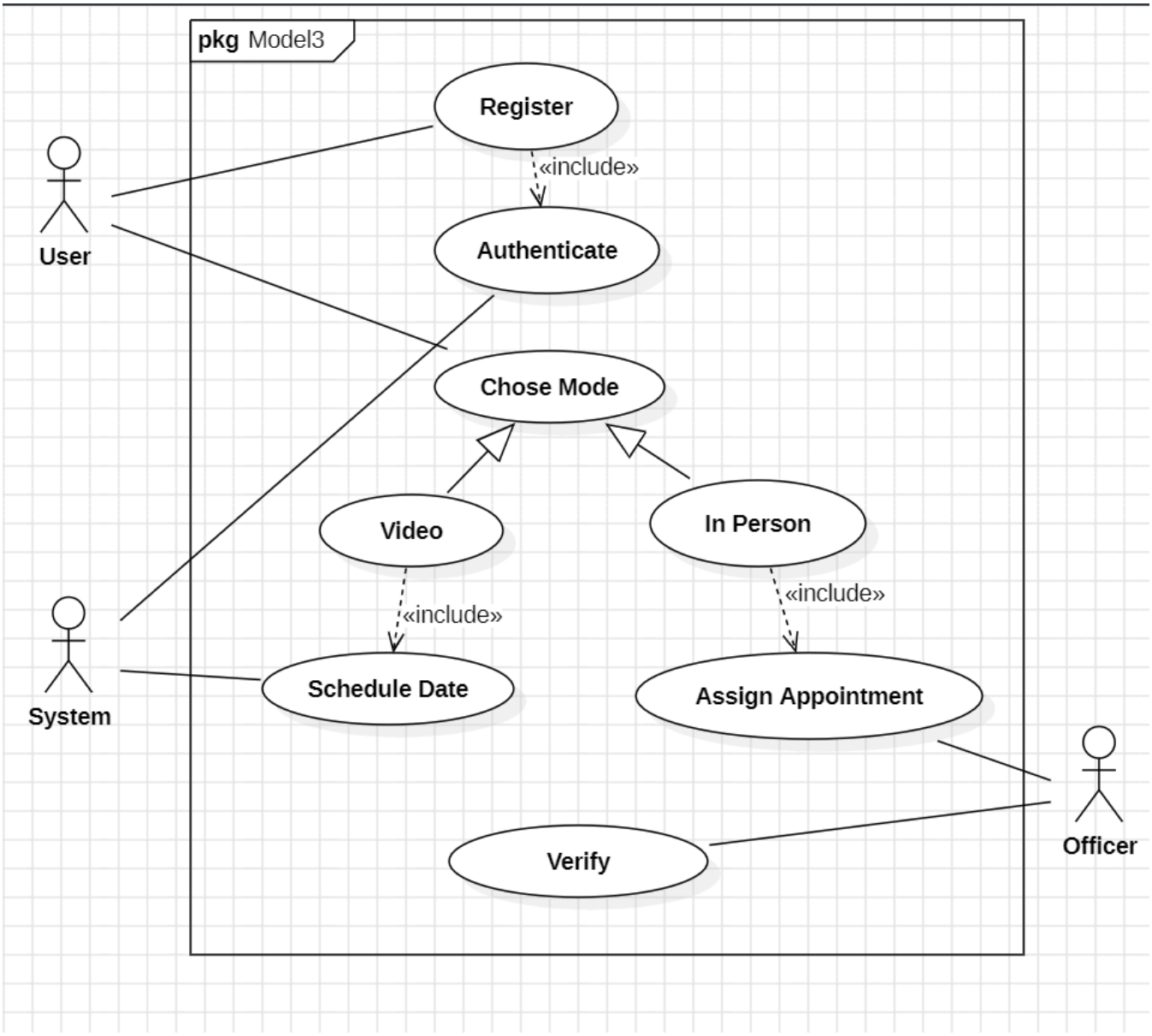


Description of State Diagram:

- This state diagram represents a Passport Automation system. It begins in the Logged Out state, transitioning to Logged In for processes such as Filling Form Details, Uploading Documents, and Passport Issuing.
- Verification involves choosing a mode (online or in-person) with subsequent video calling or scheduled interviews. Rejected applications allow re-submission.

- The Payment submachine handles mode selection, detail entry, validation, and transaction completion.

Use Case Diagram



Description of Use Case Diagram:

Actors Involved: User, System, and Officer.

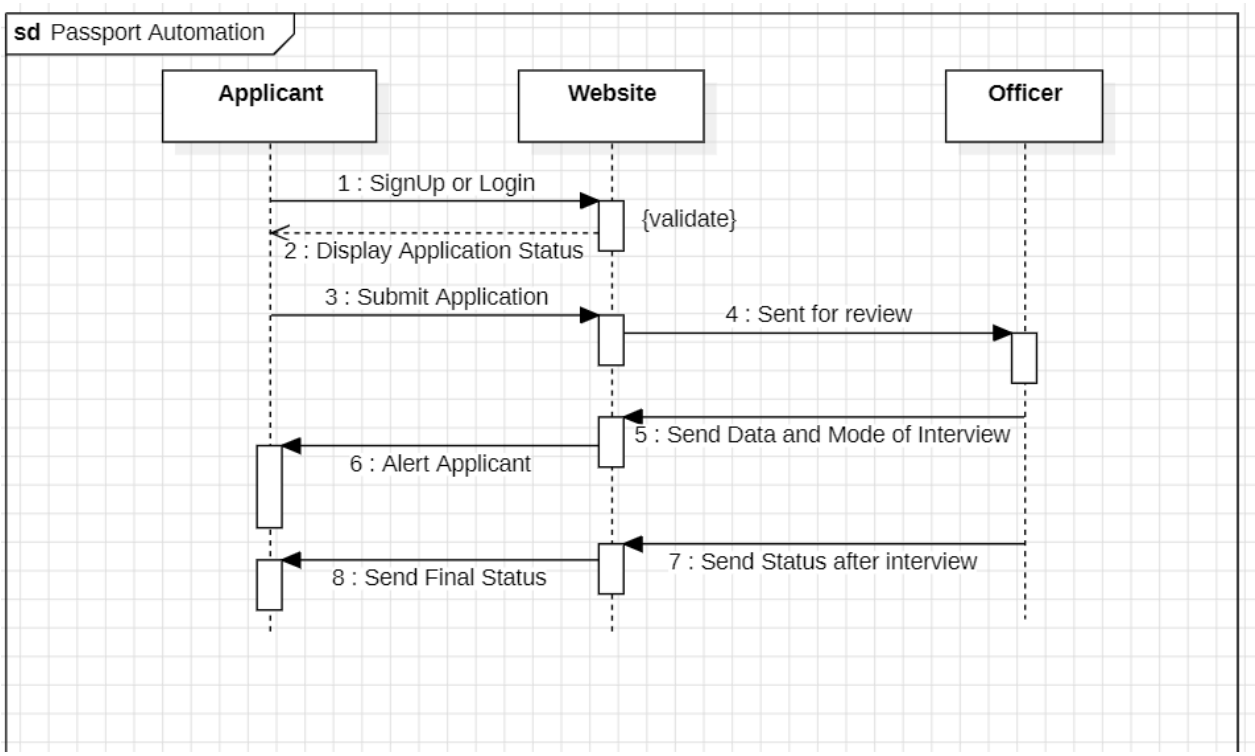
User: Registers and authenticates to access the system. They choose a mode—Video or In-Person.

System: Schedules dates for video mode or assigns appointments for in-person mode.

Includes: The Authenticate use case is included in Register, and Schedule Date and Assign Appointment include their respective mode-specific processes.

Extends: Verification extends to both appointment modes, ensuring successful completion by the Officer.

Sequence Diagram



Description:

SignUp or Login: The applicant signs up or logs into the passport automation system through the website, and their credentials are validated.

Display Application Status: The website displays the current status of the applicant's passport application.

Submit Application: The applicant submits their passport application via the website.

Sent for Review: The website forwards the submitted application data to the officer for review.

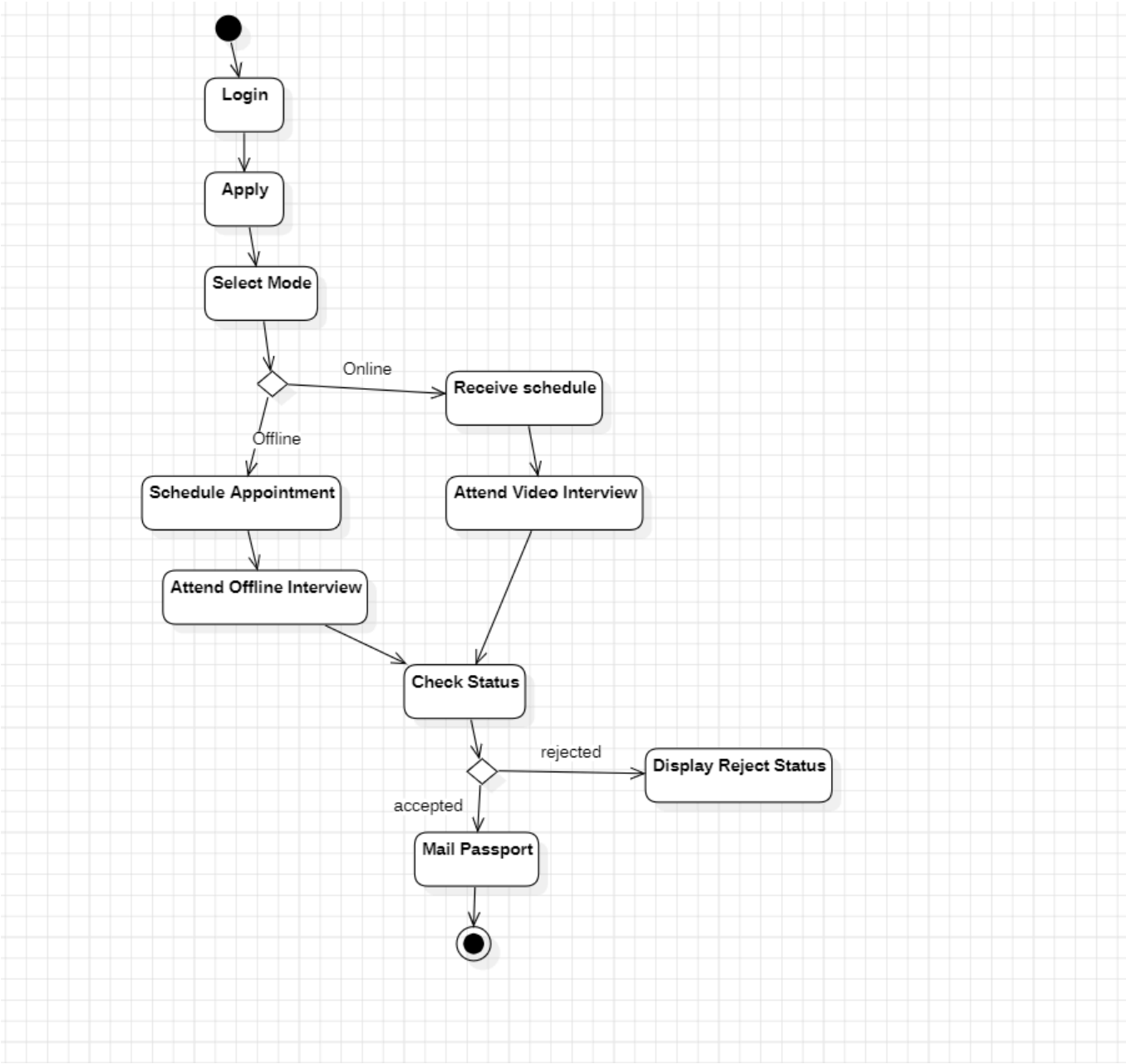
Send Data and Mode of Interview: The officer sends back the reviewed data along with the mode and schedule of the interview.

Alert Applicant: The website notifies the applicant about the interview details or the next steps.

Send Status after Interview: After the interview, the officer updates the application status, which is sent back to the website.

Send Final Status: The website communicates the final status of the application to the applicant.

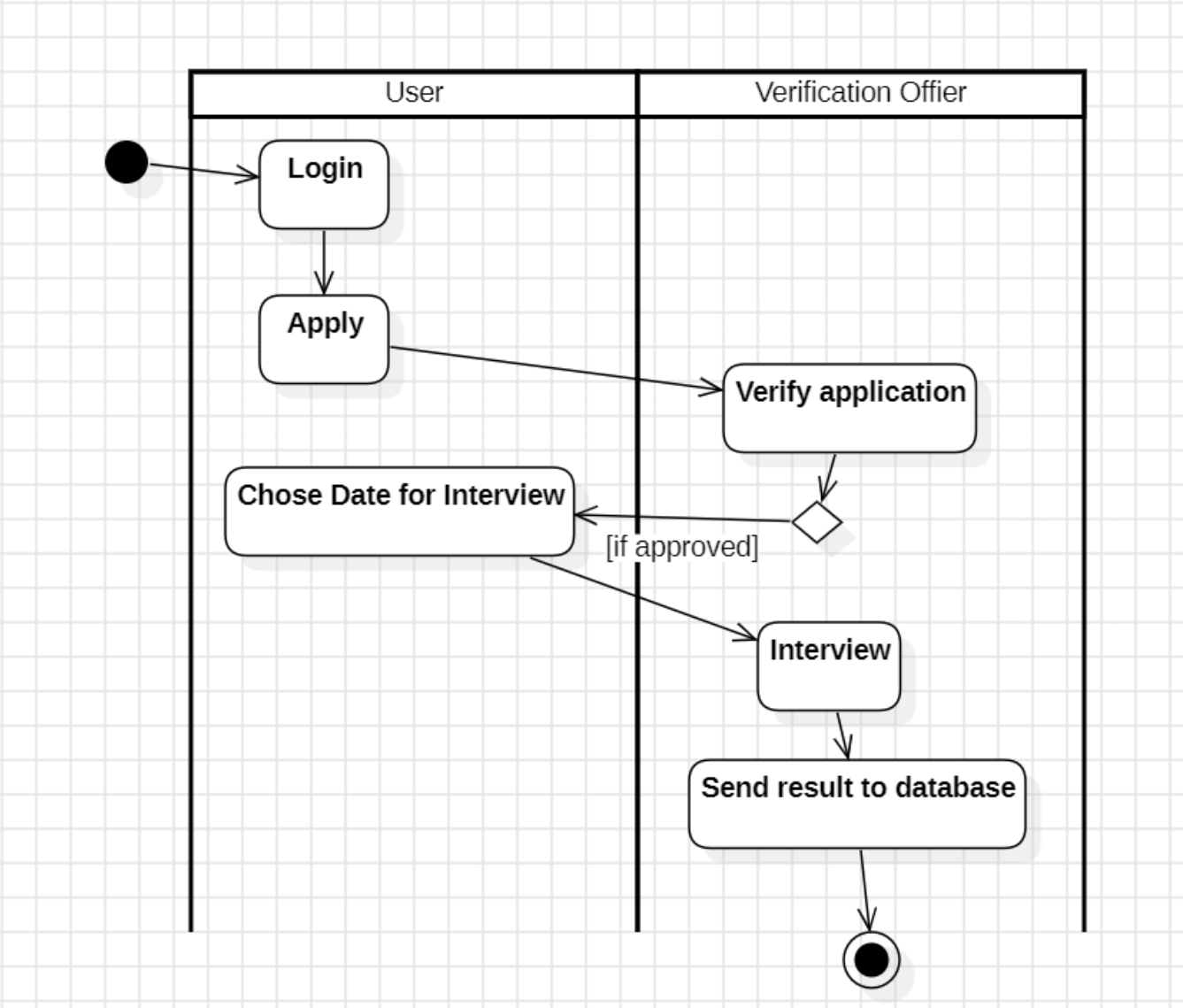
Activity Diagram



Description:

- The user logs in to the application. They apply for the passport.
- They select the mode i.e online or offline.
- If online they receive a schedule and attend the interview online.
- If offline they schedule an appointment and attend the interview offline.
- Post this they check the status of their application on the platform. If accepted the passport is mailed to them.

Activity Diagram with Swimlane



The swimlane has 2 lanes namely User and Verification Officer.