0B.M.S. COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

Lab Record

Object Oriented Analysis and Design

Submitted in partial fulfillment for the 5th Semester Laboratory

Bachelor of Technology
in
Computer Science and Engineering

Submitted by:

Dhruva S Rao

(1BM23CS092)

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
Aug-Dec 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(23CS6PCSEO) laboratory has been carried out by Dhruva S Rao (1BM23CS092) during the 5th Semester Aug-Dec-2025.

Signature of the Faculty In charge:

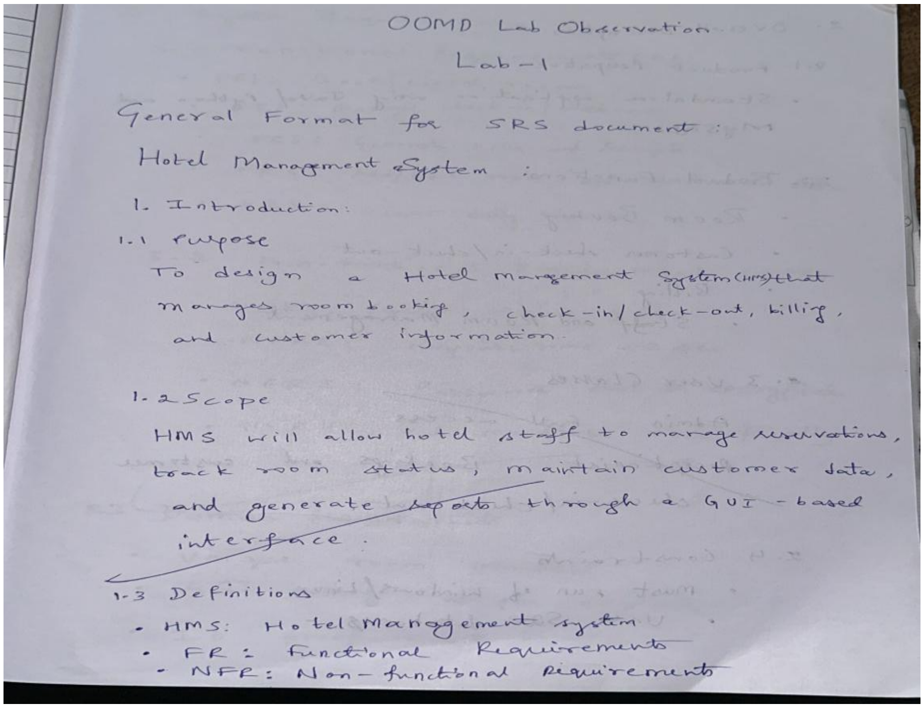
NAME OF THE FACULTY: Sonika Sharma

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

Table of Contents

1. Hotel Management System

SRS DOCUMENT



OOMD Lab Observation

Lab-1

General Format for SRS document :

Hotel Management System :

1. Introduction:

1.1 Purpose

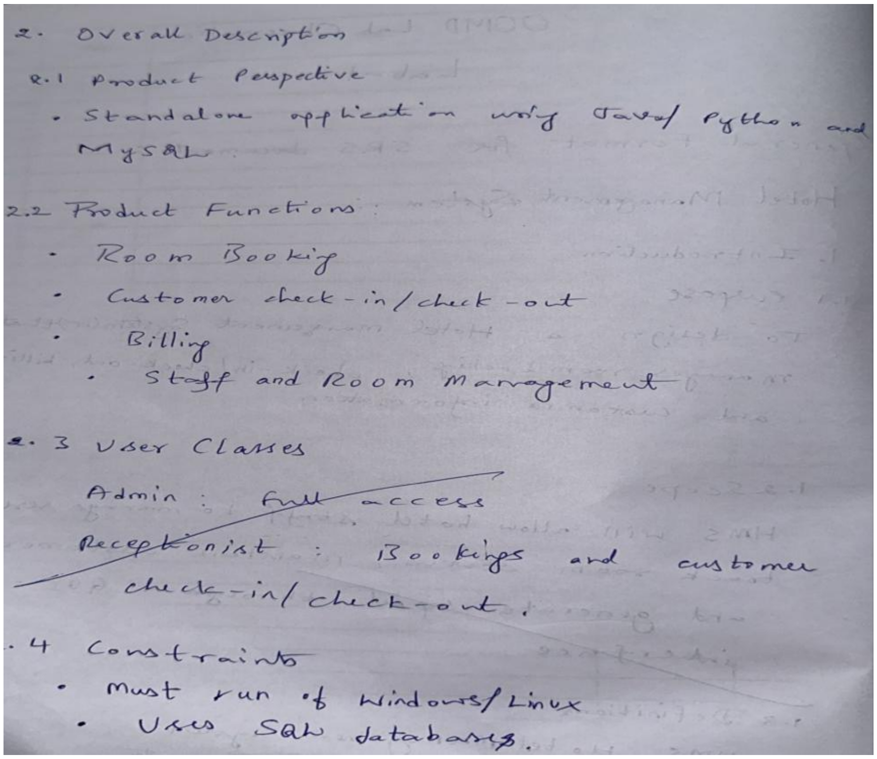
To design a Hotel management System (HMS) that manages room booking, check-in/check-out, billing, and customer information.

1.2 Scope

HMS will allow hotel staff to manage reservations, track room status, maintain customer data, and generate reports through a GUI-based interface.

1.3 Definitions

- HMS: Hotel Management system
- FR: Functional Requirements
- NFR: Non-functional Requirements



2. Overall Description

2.1 Product Perspective

- Standalone application using Java/ Python and MySQL.

2.2 Product Functions:

- Room Booking
- Customer check-in/check-out
- Billing
- Staff and Room Management

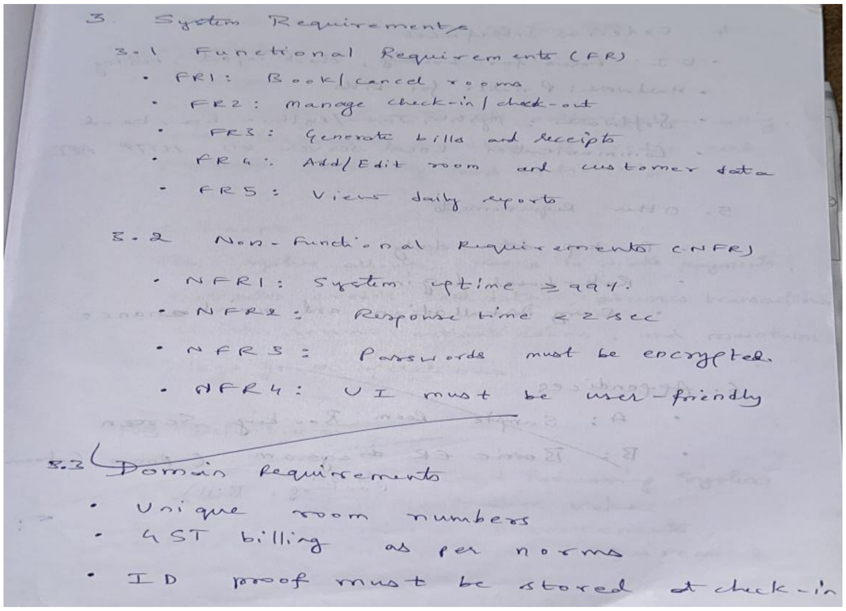
2.3 User Classes

Admin : full access

Receptionist : Bookings and customer check-in/check-out.

.4 Constraints

- Must run of Windows/Linux
- Uses SQL databases.



3. System Requirements

3.1 Functional Requirements (FR)

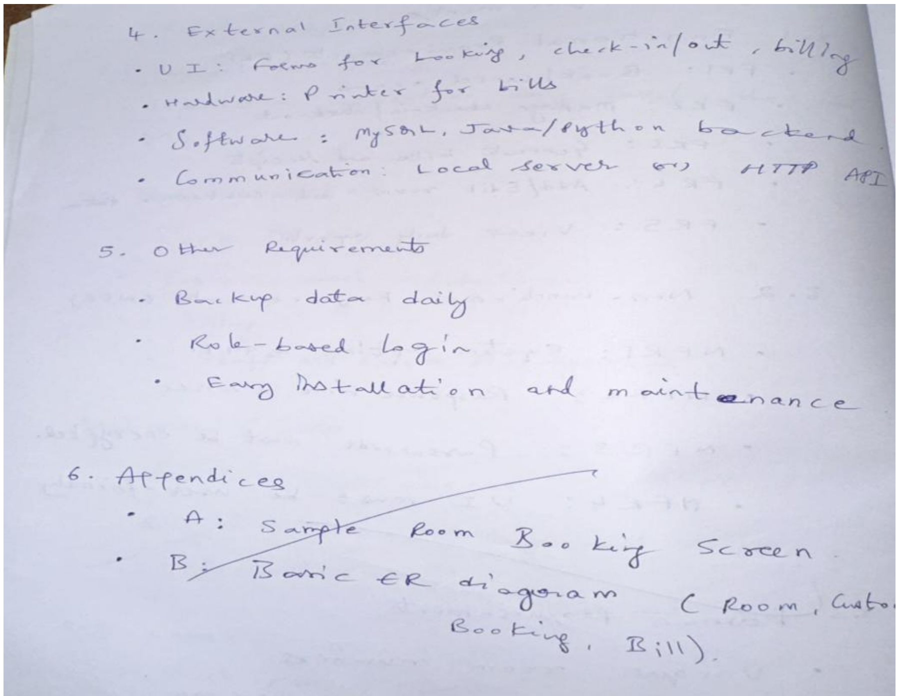
- FR1: Book/cancel rooms
- FR2: manage check-in/check-out
- FR3: Generate bills and receipts
- FR4: Add/Edit room and customer data
- FR5: View daily reports

3.2 Non-functional Requirements (NFR)

- NFR1: System uptime ≥ 99%.
- NFR2: Response time < 2 sec
- NFR3: Passwords must be encrypted.
- NFR4: UI must be user-friendly

3.3 Domain Requirements

- Unique room numbers
- GST billing as per norms
- ID proof must be stored at check-in



4. External Interfaces

- UI: Forms for booking, check-in/out, billing
- Hardware: Printer for bills
- Software: MySQL, Java/Python backend
- Communication: Local server (or) HTTP API

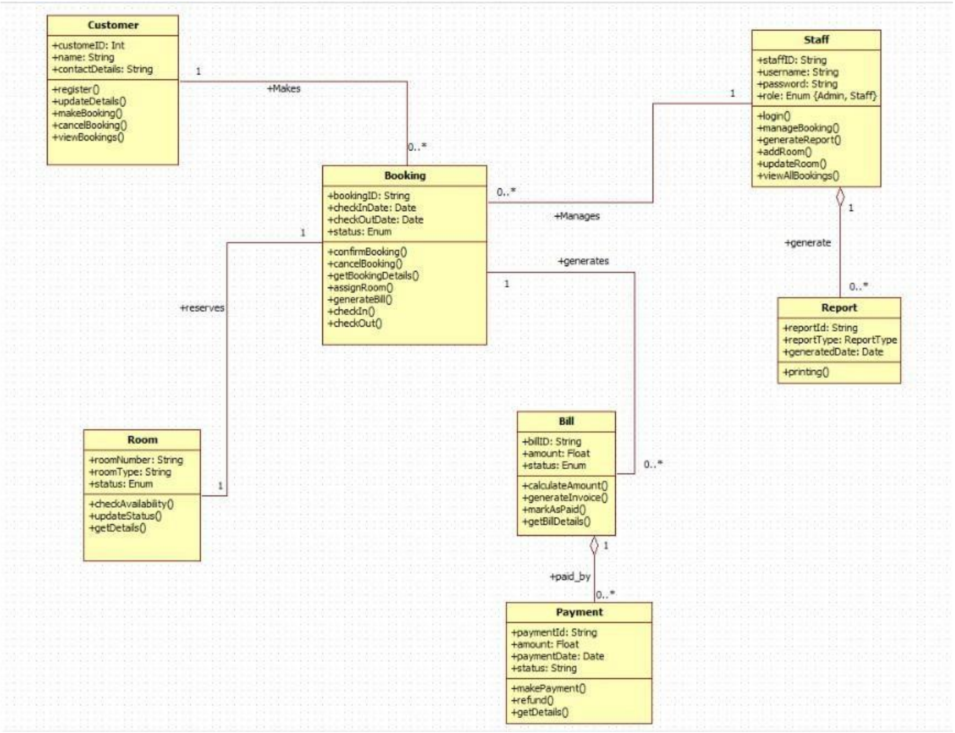
5. Other Requirements

- Backup data daily
- Role-based login
- Easy Installation and maintenance

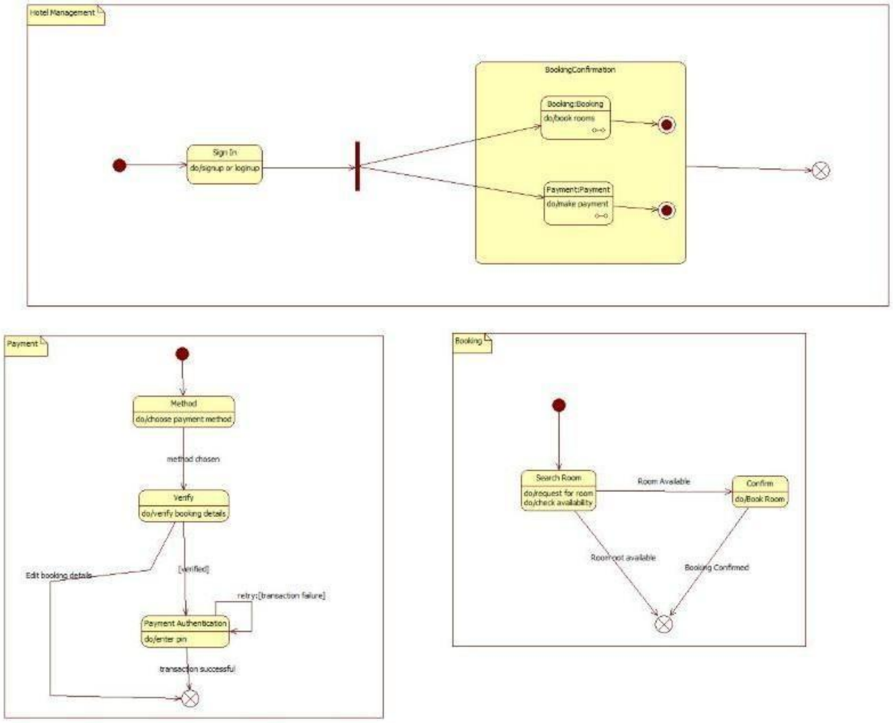
6. Appendices

- A: Sample Room Booking Screen
- B: Basic ER diagram (Room, Custo
Booking, Bill).

CLASS DIAGRAM

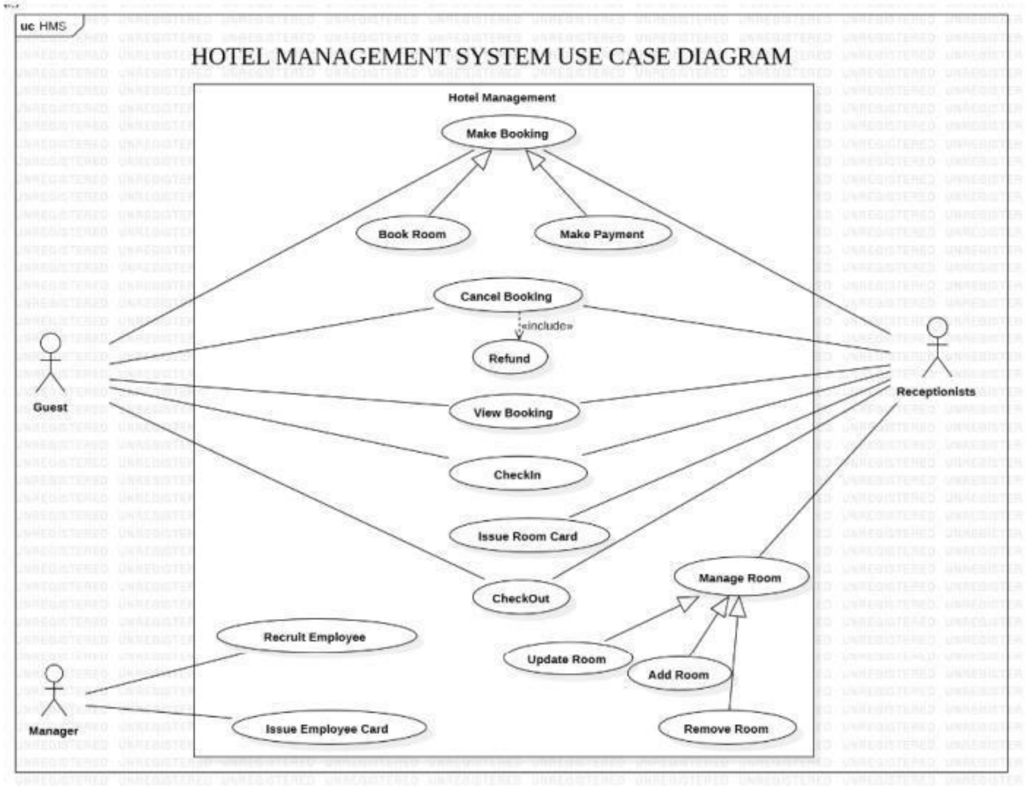


STATE DIAGRAM

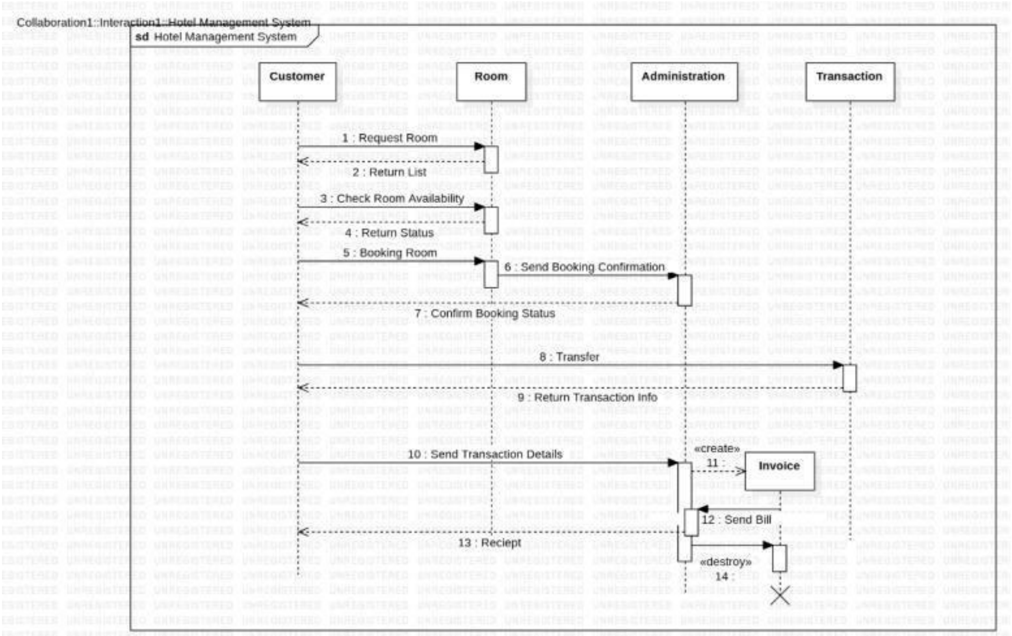


INTERACTION MODELS

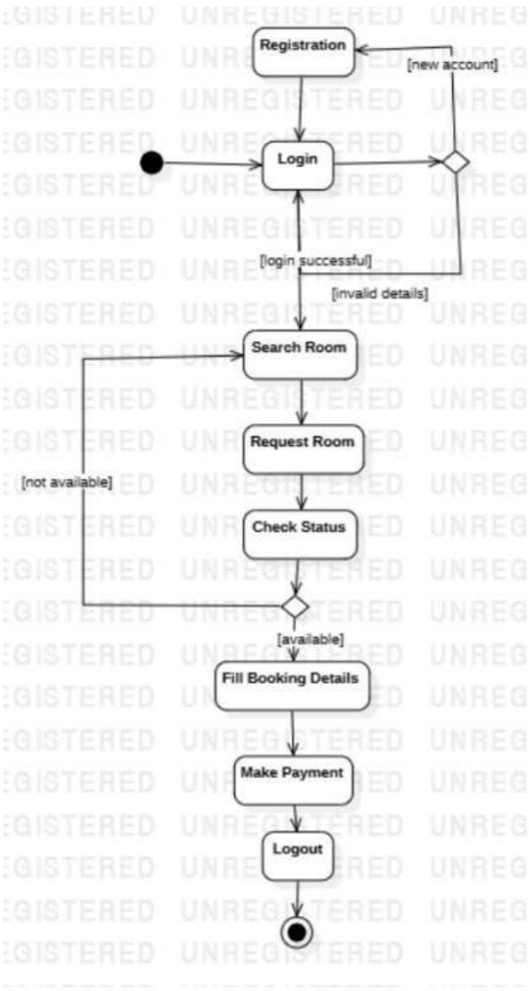
a. Use case model



b. Sequence model

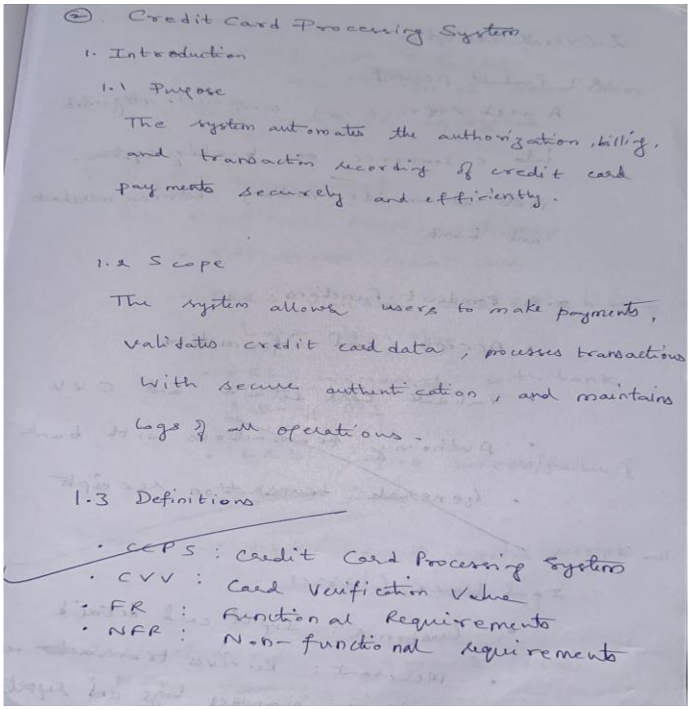

Collaboration1::Interaction1::Hotel Management System
sd Hotel Management System
Customer
Room
Administration
Transaction
1 : Request Room
2 : Return List
3 : Check Room Availability
4 : Return Status
5 : Booking Room
6 : Send Booking Confirmation
7 : Confirm Booking Status
8 : Transfer
9 : Return Transaction Info
10 : Send Transaction Details
«create»
11 :
Invoice
12 : Send Bill
13 : Reciept
«destroy»
14 :

c. Activity model



2.Credit Card Processing

2.1 SRS DOCUMENT



②. Credit Card Processing System

1. Introduction

1.1 Purpose

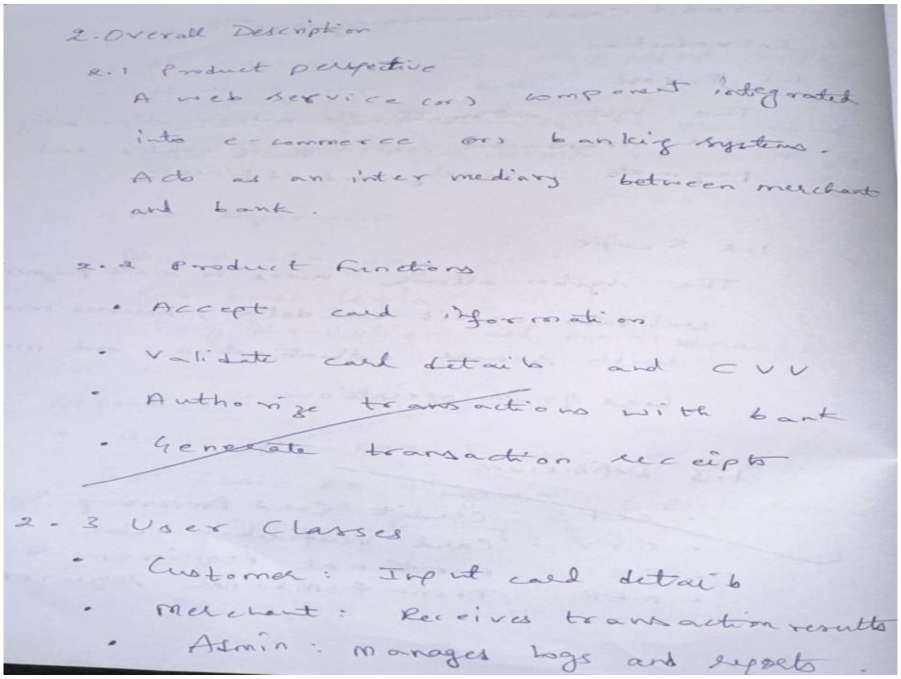
The system automates the authorization, billing, and transaction recording of credit card payments securely and efficiently.

1.2 Scope

The system allows users to make payments, validates credit card data, processes transactions with secure authentication, and maintains logs of all operations.

1.3 Definitions

- CCPS : Credit Card Processing System
- CVV : Card Verification Value
- FR : Functional Requirements
- NFR : Non-functional requirements



2. Overall Description

2.1 Product perspective

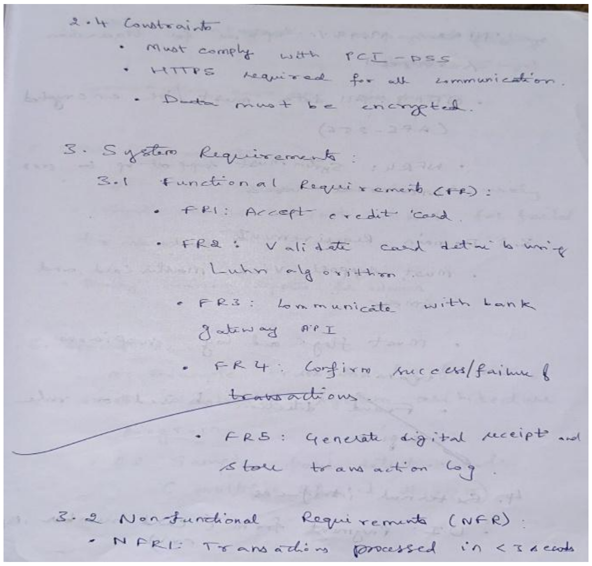
A web service (or) component integrated into e-commerce (or) banking systems. Acts as an intermediary between merchant and bank.

2.2 Product functions

- Accept card information
- Validate card details and CVV
- Authorize transactions with bank
- Generate transaction receipts

2.3 User Classes

- Customer : Input card details
- Merchant : Receives transaction results
- Admin : manages logs and reports.



2.4 Constraints

- Must comply with PCI-DSS
- HTTPS required for all communication.
- Data must be encrypted.

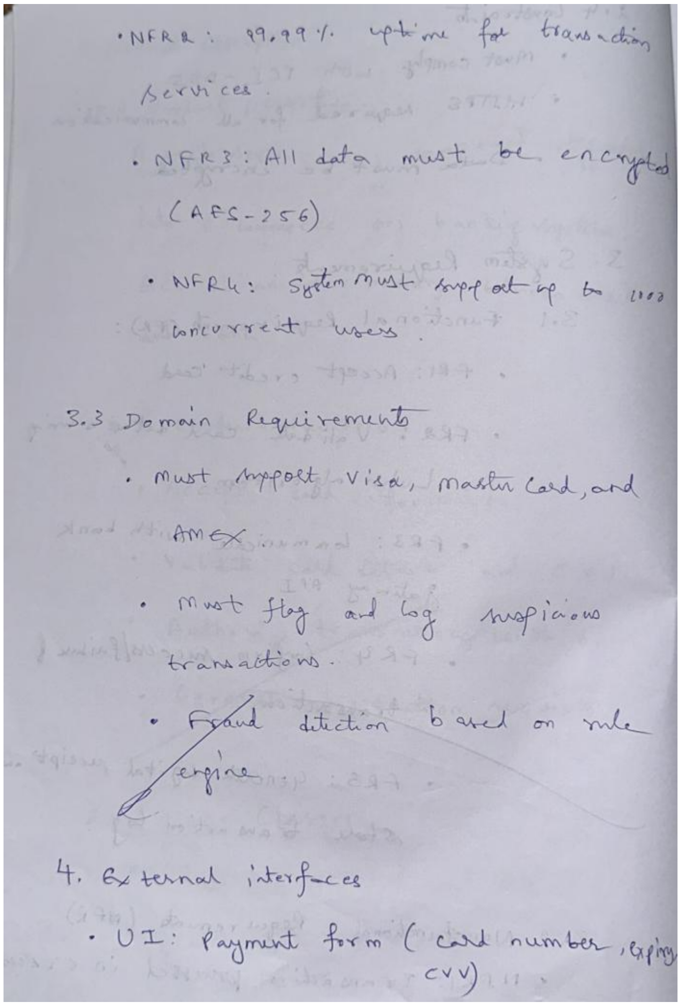
3. System Requirements :

3.1 Functional Requirements (FR) :

- FR1: Accept credit card.
- FR2: Validate card details using Luhn algorithm.
- FR3: Communicate with bank gateway API
- FR4: Confirm success/failure of transactions.
- FR5: Generate digital receipt and store transaction log.

3.2 Non-functional Requirements (NFR) :

- NFR1: Transactions processed in < 3 seconds



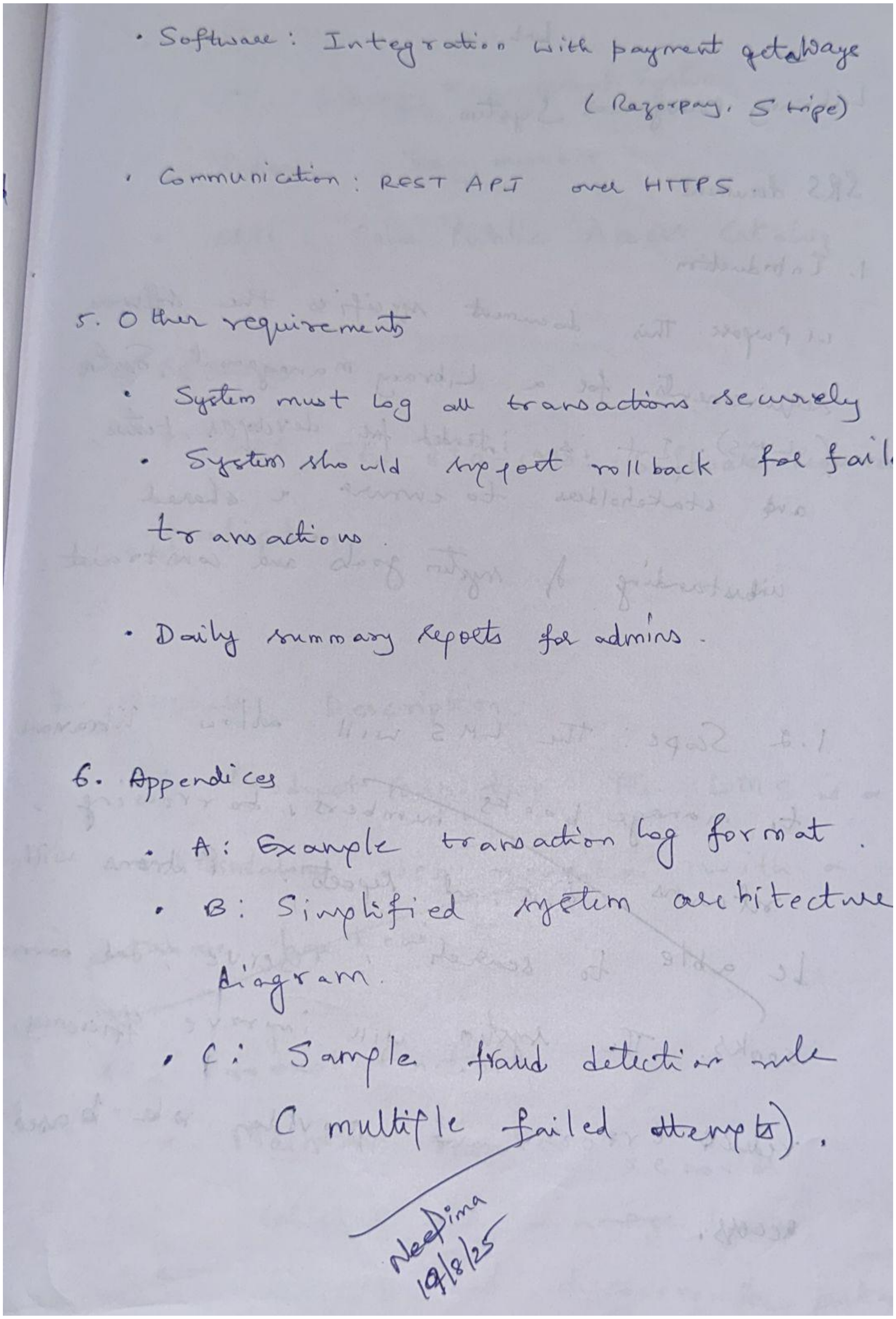
- NFR2: 99.99% uptime for transaction services.
- NFR3: All data must be encrypted (AES-256)
- NFR4: System must support up to 1000 concurrent users.

3.3 Domain Requirements

- Must support Visa, MasterCard, and AMEX
- Must flag and log suspicious transactions.
- Fraud detection based on rule engine

4. External interfaces

- UI: Payment form (card number, Expiry, CVV)



- Software: Integration with payment getaWaye (Razorpay. Stripe)
- Communication: REST API over HTTPS.

5. Other requirements

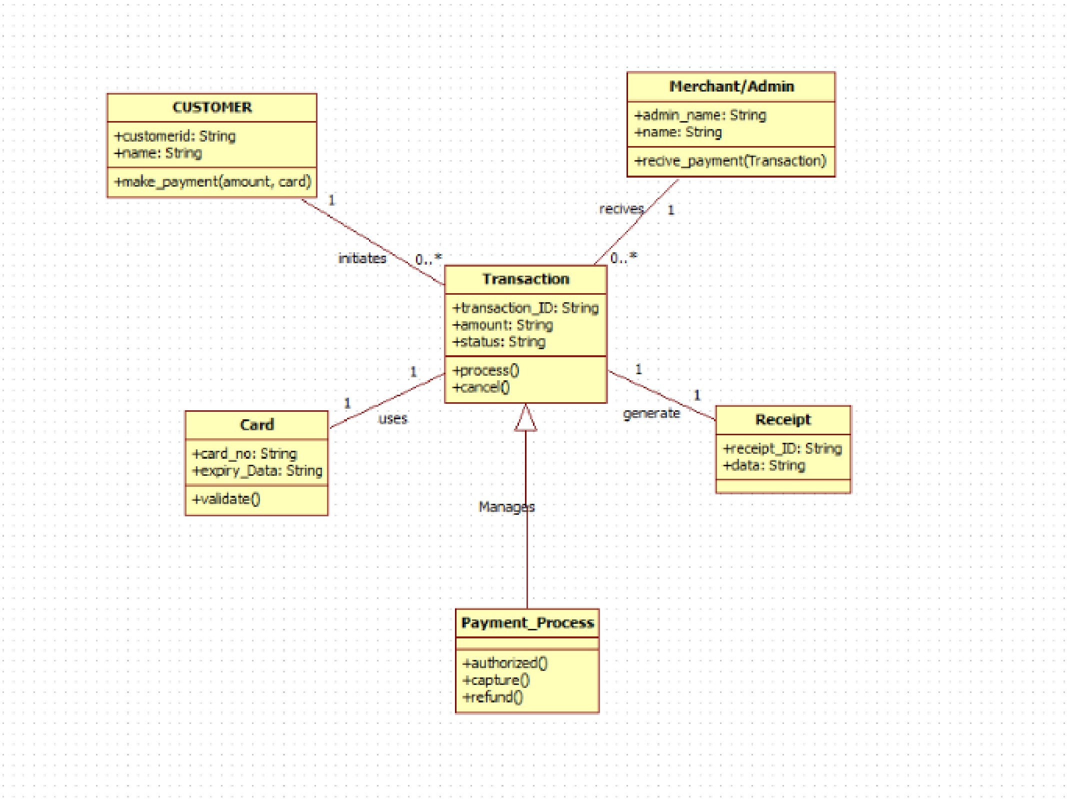
- System must log all transactions securely
- System should support rollback for fail transactions.
- Daily summary reports for admins.

6. Appendices

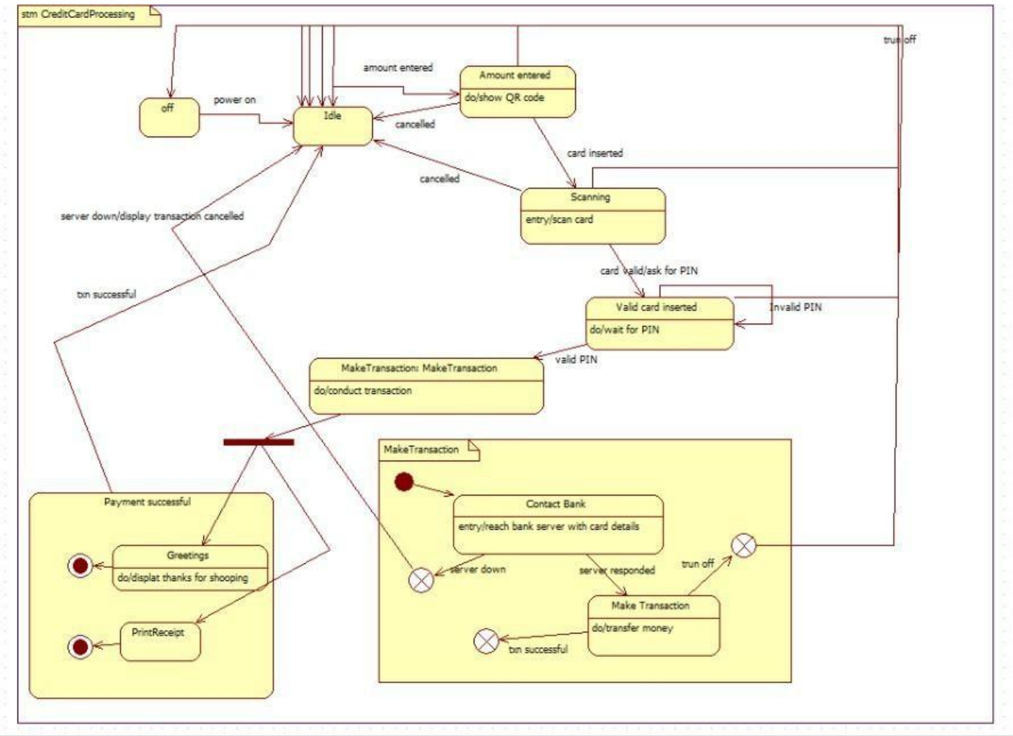
- A: Example transaction log format.
- B: Simplified system architecture diagram.
- C: Sample fraud detection rule (multiple failed attempts).

Neelima
19/8/25

2.3 Class Diagram

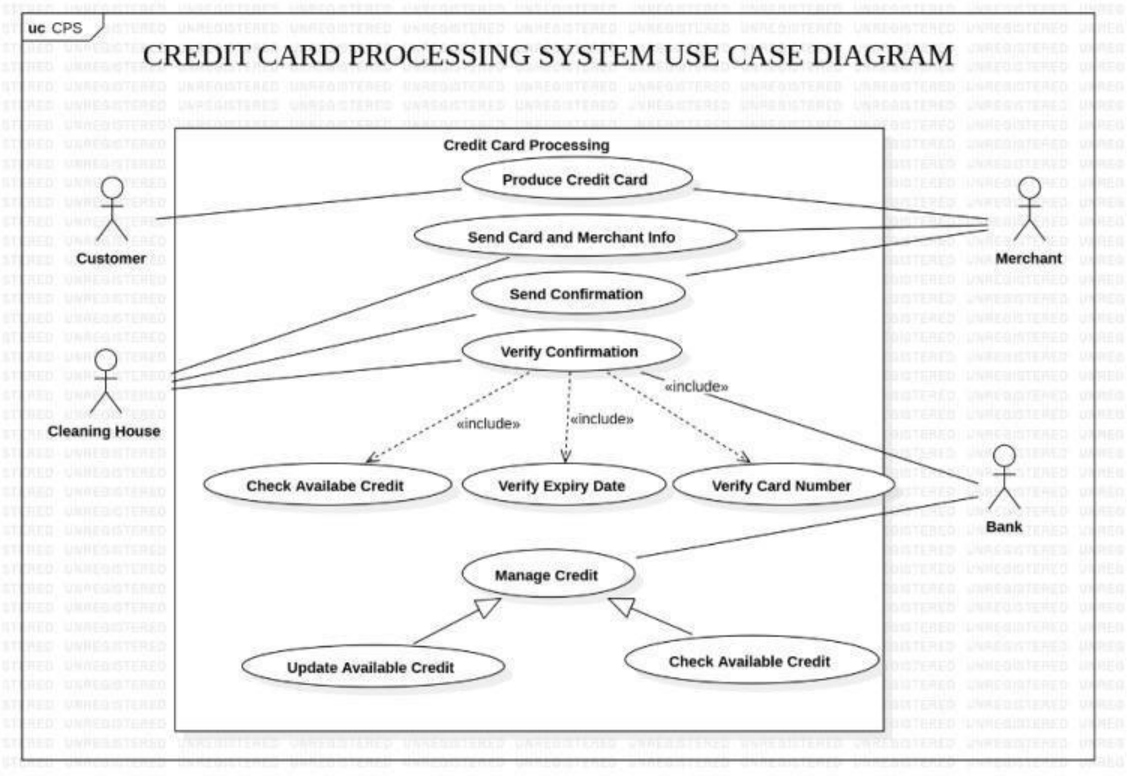


2.3 State Diagram

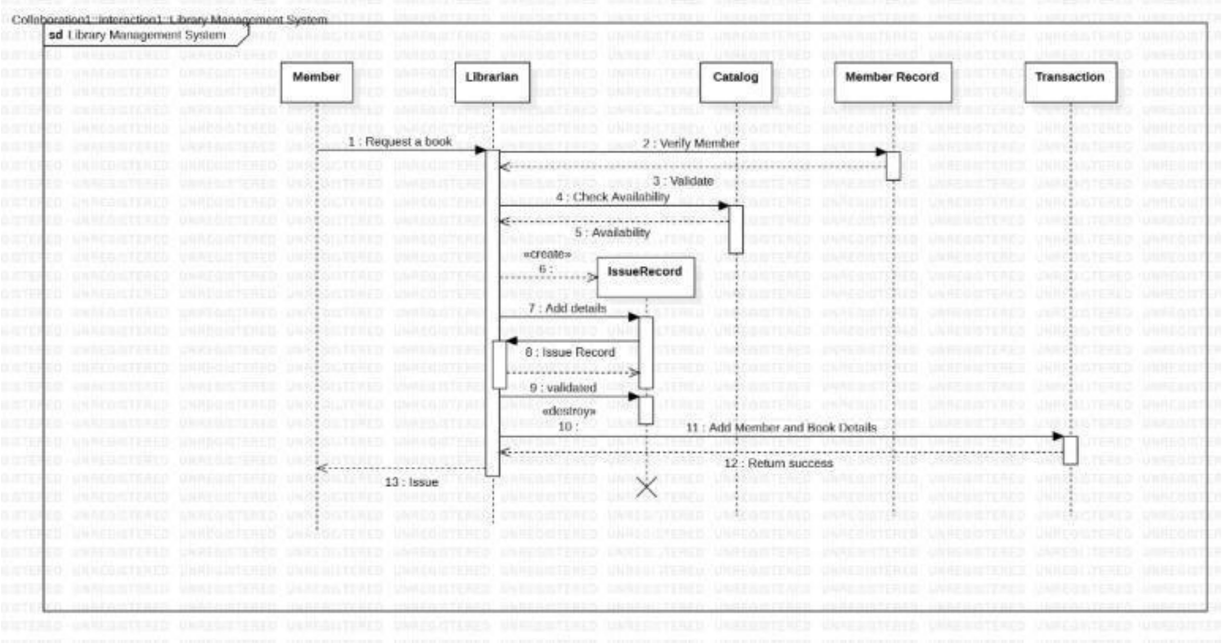


2.4 Interaction Models

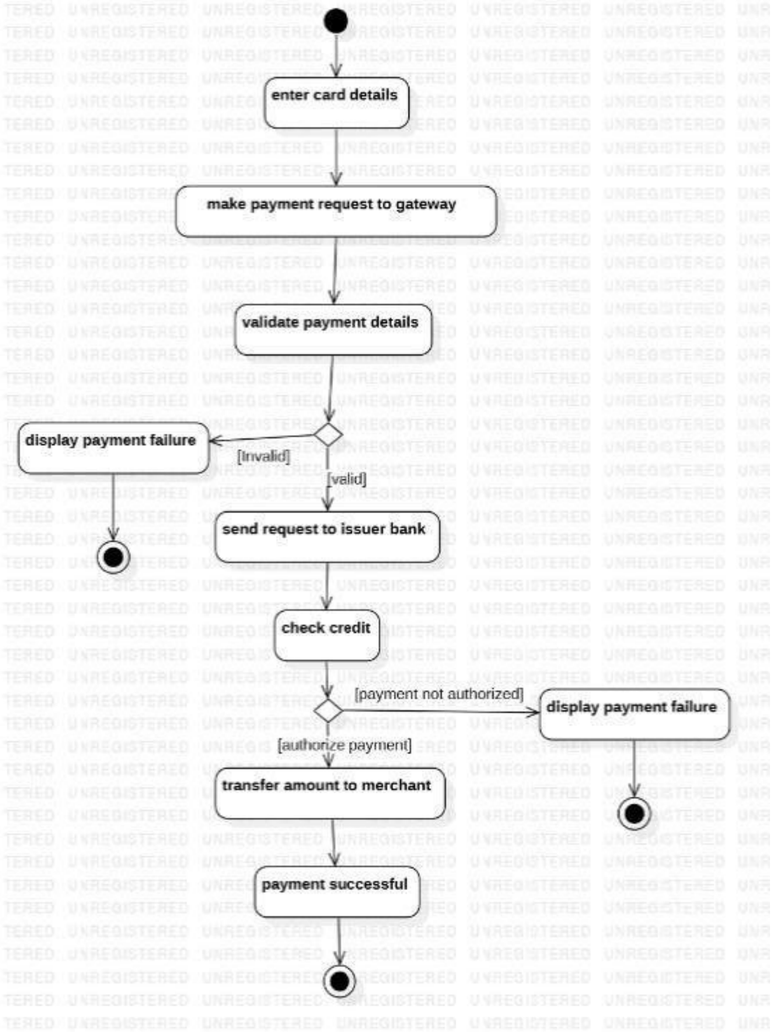
a. Use Case Model



d. Sequence model

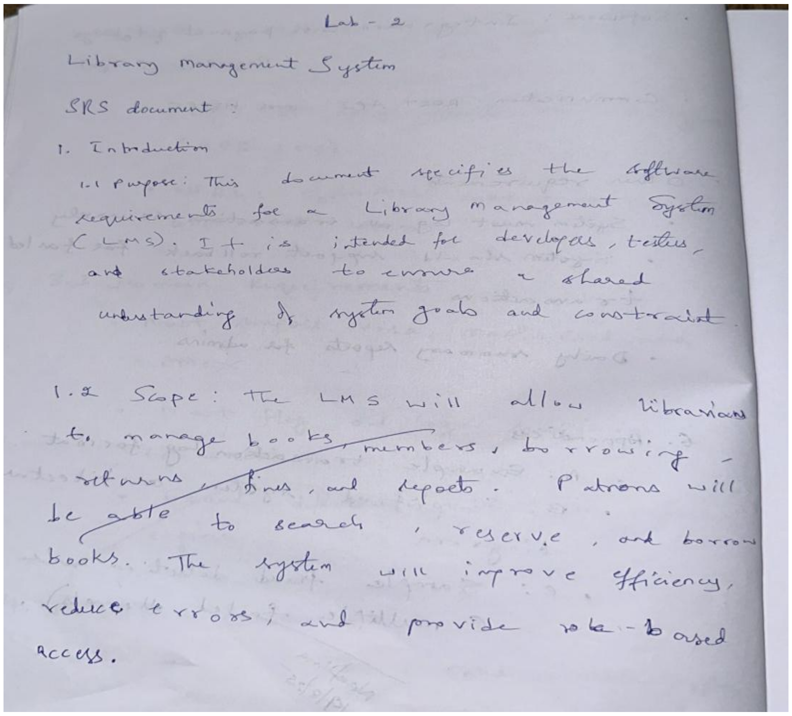


c. Activity Model



3.Library management system

3.1 SRS Document



Lab - 2

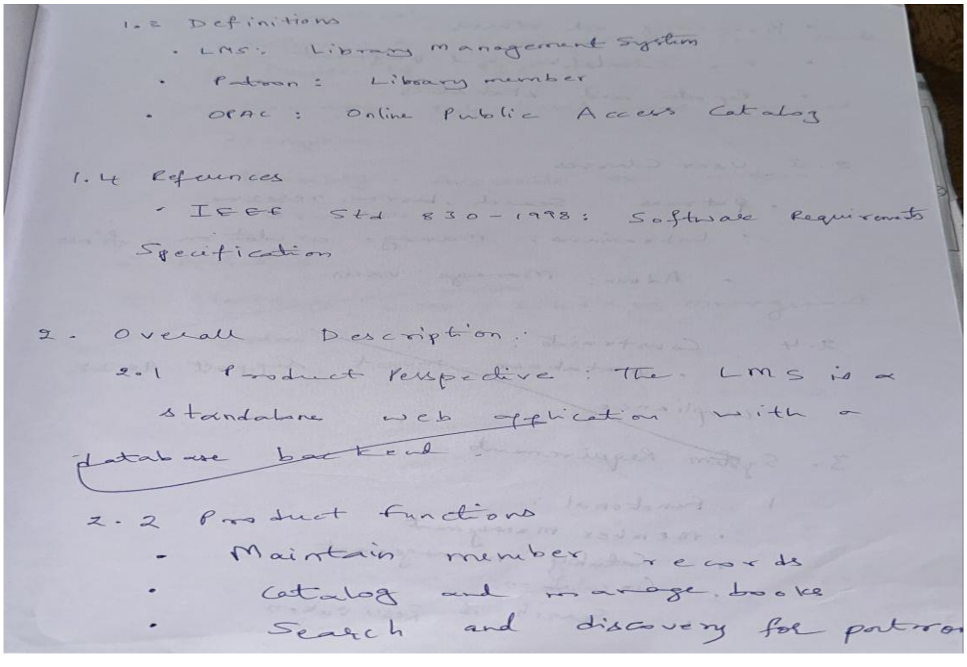
Library Management System

SRS document :

1. Introduction

1.1 Purpose: This document specifies the software requirements for a Library management System (LMS). It is intended for developers, testers, and stakeholders to ensure a shared understanding of system goals and constraint.

1.2 Scope: The LMS will allow librarians to manage books, members, borrowing - returns, fines, and reports. Patrons will be able to search, reserve, and borrow books. The system will improve efficiency, reduce errors; and provide role-based access.



1.3 Definitions

- LMS: Library Management System
- Patron : Library member
- OPAC : Online Public Access Catalog

1.4 References

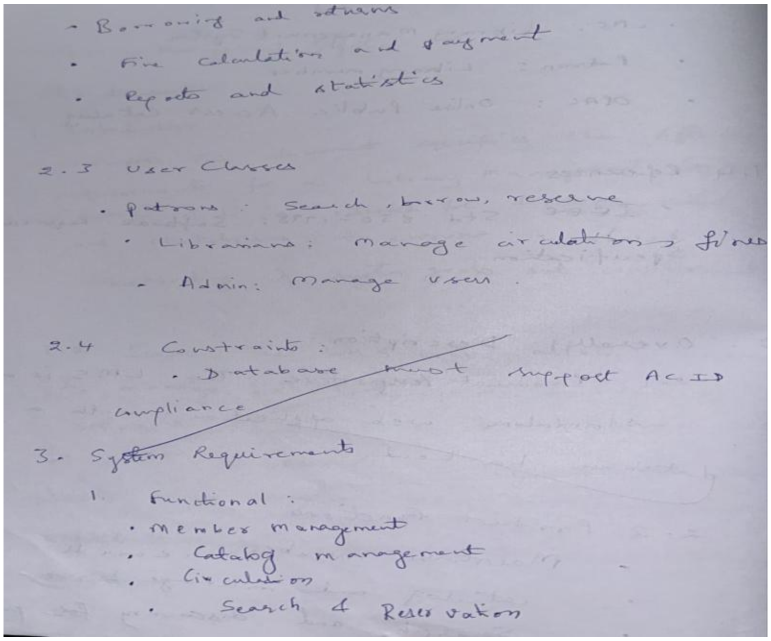
- IEEE Std 830-1998: Software Requirements Specification

2. Overall Description:

2.1 Product Perspective : The LMS is a standalone web application with a database backend.

2.2 Product functions

- Maintain member records
- Catalog and manage books
- Search and discovery for patron



- Borrowing and returns
- Fine calculation and payment
- Reports and statistics

2.3 User Classes

- Patrons: Search, borrow, reserve
- Librarians: manage circulations, fines
- Admin: Manage users.

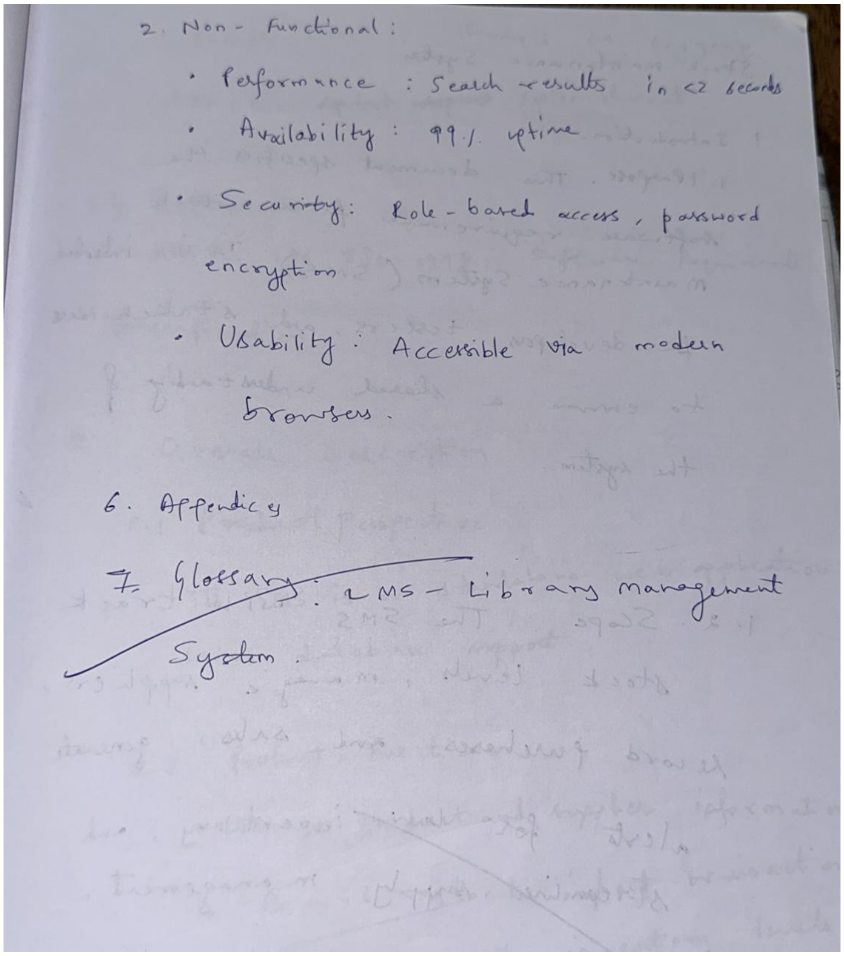
2.4 Constraints:

- Database must support ACID compliance

3. System Requirements

1. Functional:

- Member management
- Catalog management
- Circulation
- Search & Reservation



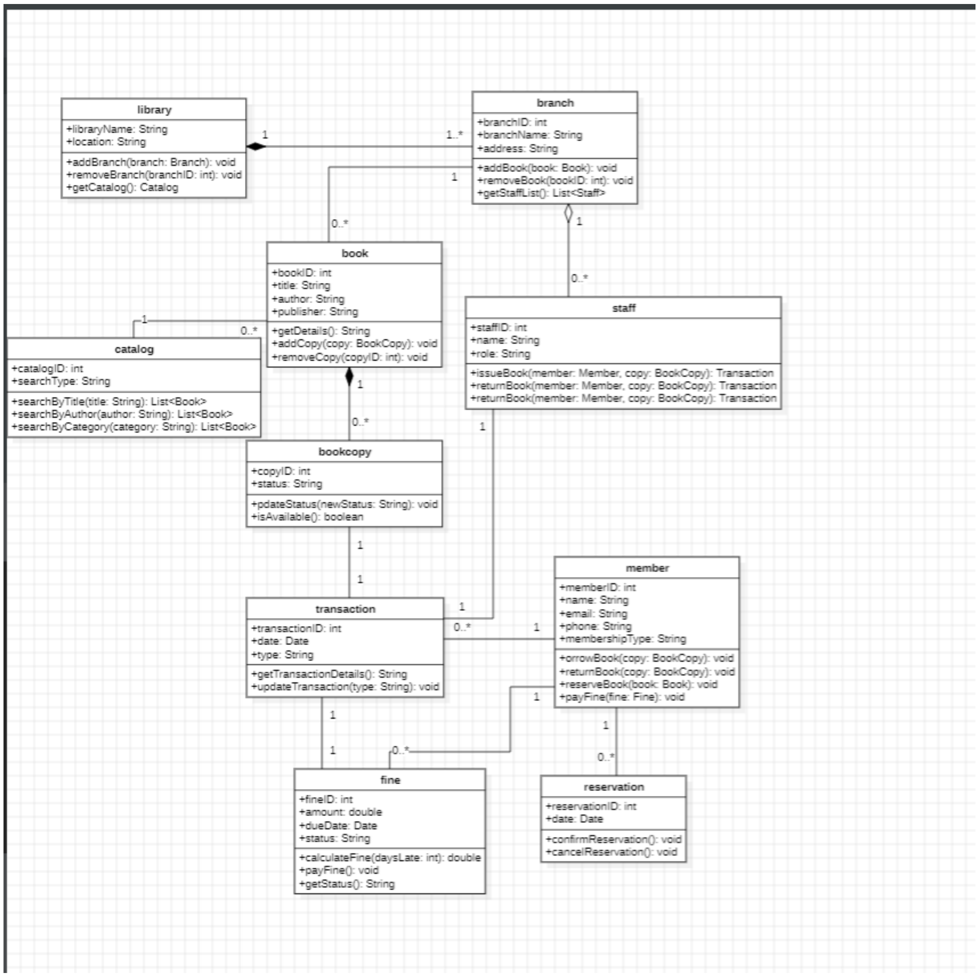
2. Non-Functional:

- Performance: Search results in <2 seconds
- Availability: 99% uptime
- Security: Role-based access, password encryption
- Usability: Accessible via modern browsers.

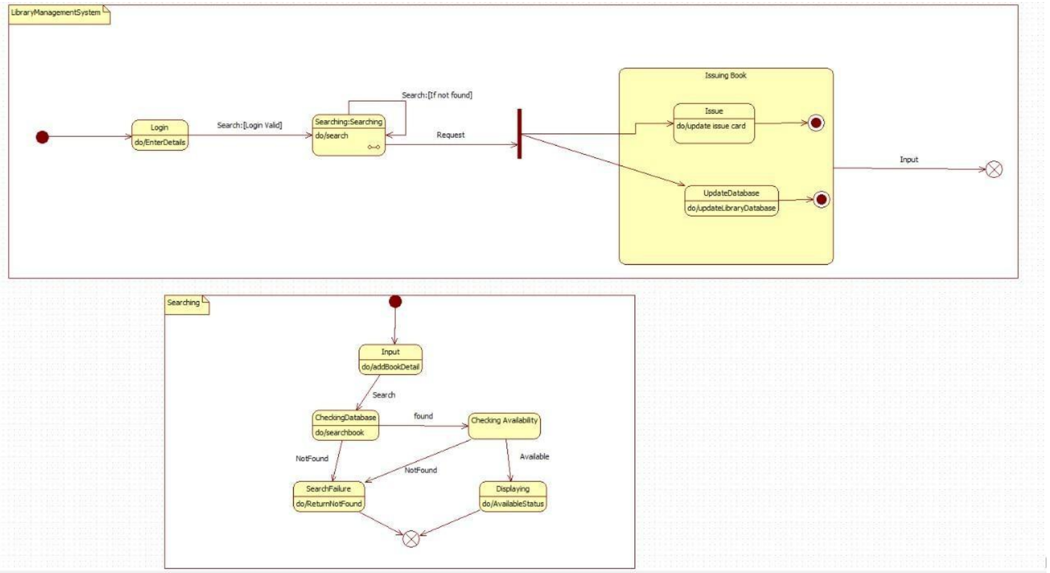
6. Appendices

7. Glossary: LMS – Library management System.

3.3 Class Model

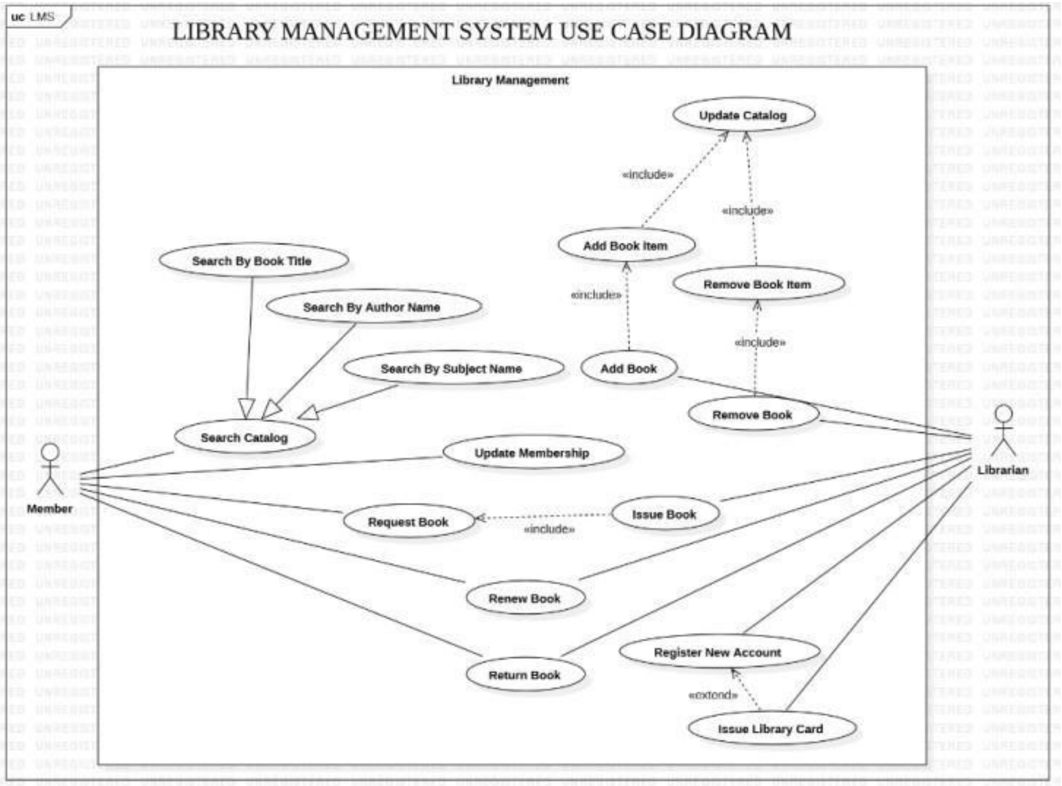

library
+libraryName: String
+location: String
+addBranch(branch: Branch): void
+removeBranch(branchID: int): void
+getCatalog(): Catalog
1
1..*
branch
+branchID: int
+branchName: String
+address: String
+addBook(book: Book): void
+removeBook(bookID: int): void
+getStaffList(): List<Staff>
1
0..*
1
0..*
book
+bookID: int
+title: String
+author: String
+publisher: String
+getDetails(): String
+addCopy(copy: BookCopy): void
+removeCopy(copyID: int): void
1
0..*
catalog
+catalogID: int
+searchType: String
+searchByTitle(title: String): List<Book>
+searchByAuthor(author: String): List<Book>
+searchByCategory(category: String): List<Book>
staff
+staffID: int
+name: String
+role: String
+issueBook(member: Member, copy: BookCopy): Transaction
+returnBook(member: Member, copy: BookCopy): Transaction
+returnBook(member: Member, copy: BookCopy): Transaction
1
0..*
bookcopy
+copyID: int
+status: String
+pdateStatus(newStatus: String): void
+isAvailable(): boolean
1
1
1
member
+memberID: int
+name: String
+email: String
+phone: String
+membershipType: String
+orrowBook(copy: BookCopy): void
+returnBook(copy: BookCopy): void
+reserveBook(book: Book): void
+payFine(fine: Fine): void
1
0..*
1
transaction
+transactionID: int
+date: Date
+type: String
+getTransactionDetails(): String
+updateTransaction(type: String): void
1
1
1
0..*
1
0..*
fine
+fineID: int
+amount: double
+dueDate: Date
+status: String
+calculateFine(daysLate: int): double
+payFine(): void
+getStatus(): String
reservation
+reservationID: int
+date: Date
+confirmReservation(): void
+cancelReservation(): void

3.4 State model

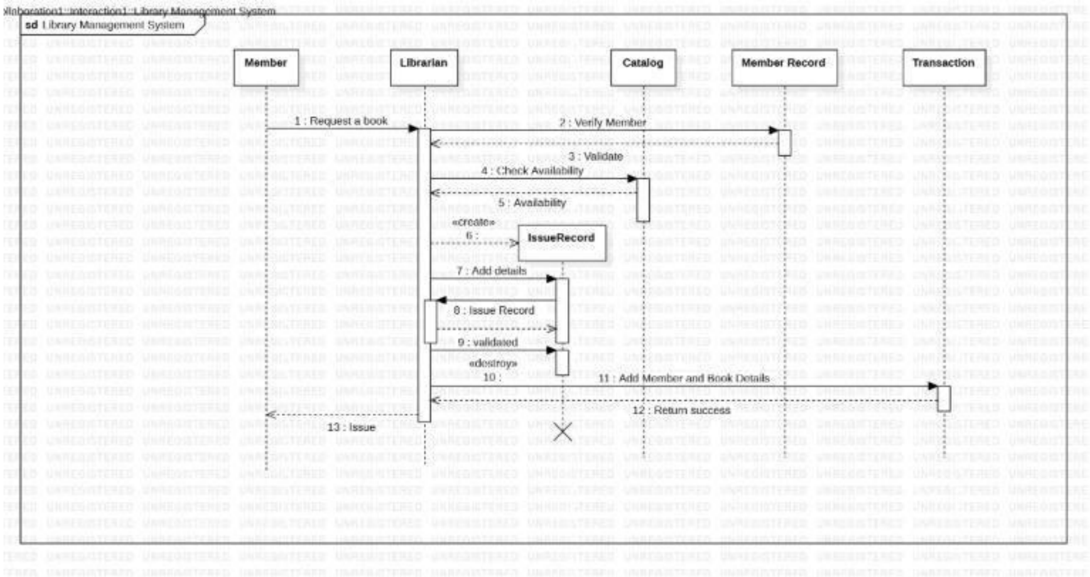


3.5 Interaction Models

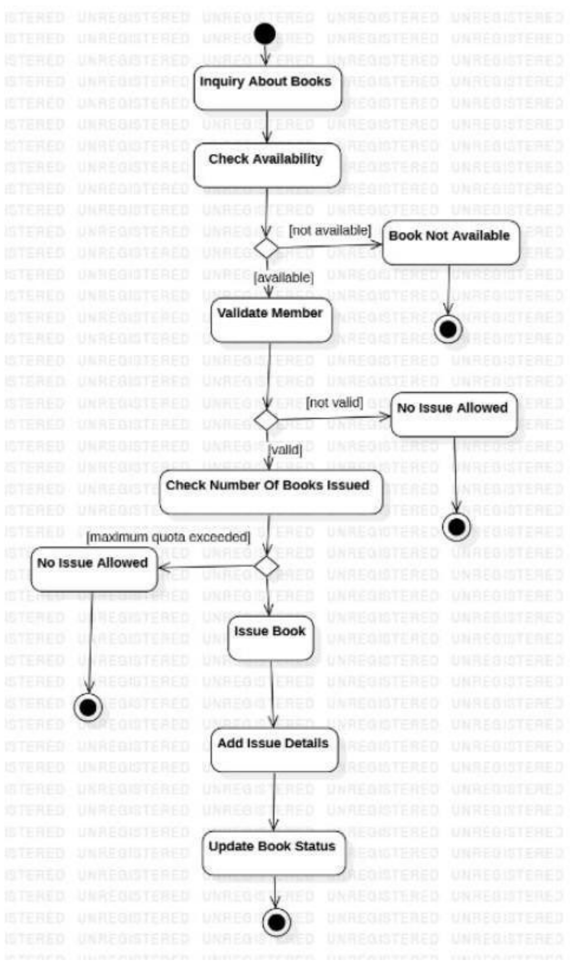
a. Use Case Model



b. Sequence Model

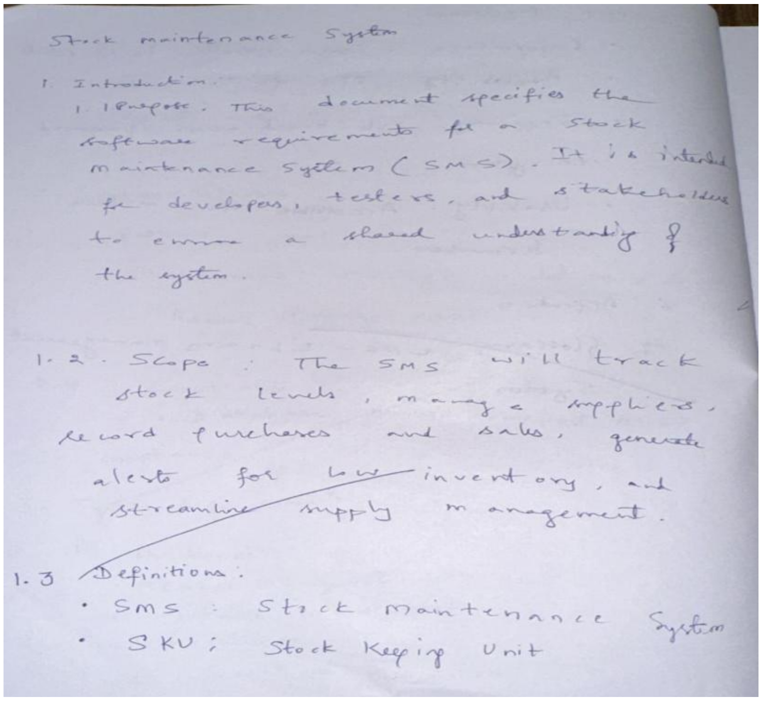


c. Activity Model



4.Stock Maintenance System

4.1 SRS Document



Stock maintenance System

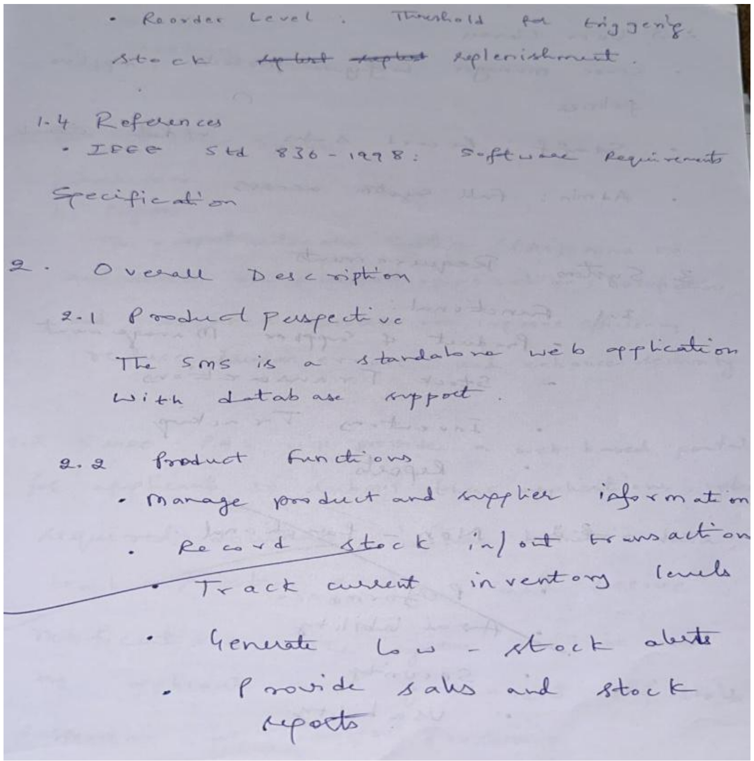
1. Introduction.

1. 1 Purpose. This document specifies the software requirements for a Stock maintenance System (SMS). It is intended for developers, testers, and stakeholders to ensure a shared understanding of the system.

1. 2. Scope: The SMS will track stock levels, manage suppliers, record purchases and sales, generate alerts for low inventory, and streamline supply management.

1. 3 Definitions:

- SMS : Stock maintenance System
- SKU ; Stock Keeping Unit



- Reorder Level : Threshold for triggering stock ~~[illegible]~~ ~~[illegible]~~ replenishment.

1.4 References

- IEEE Std 836-1998: Software Requirements Specification

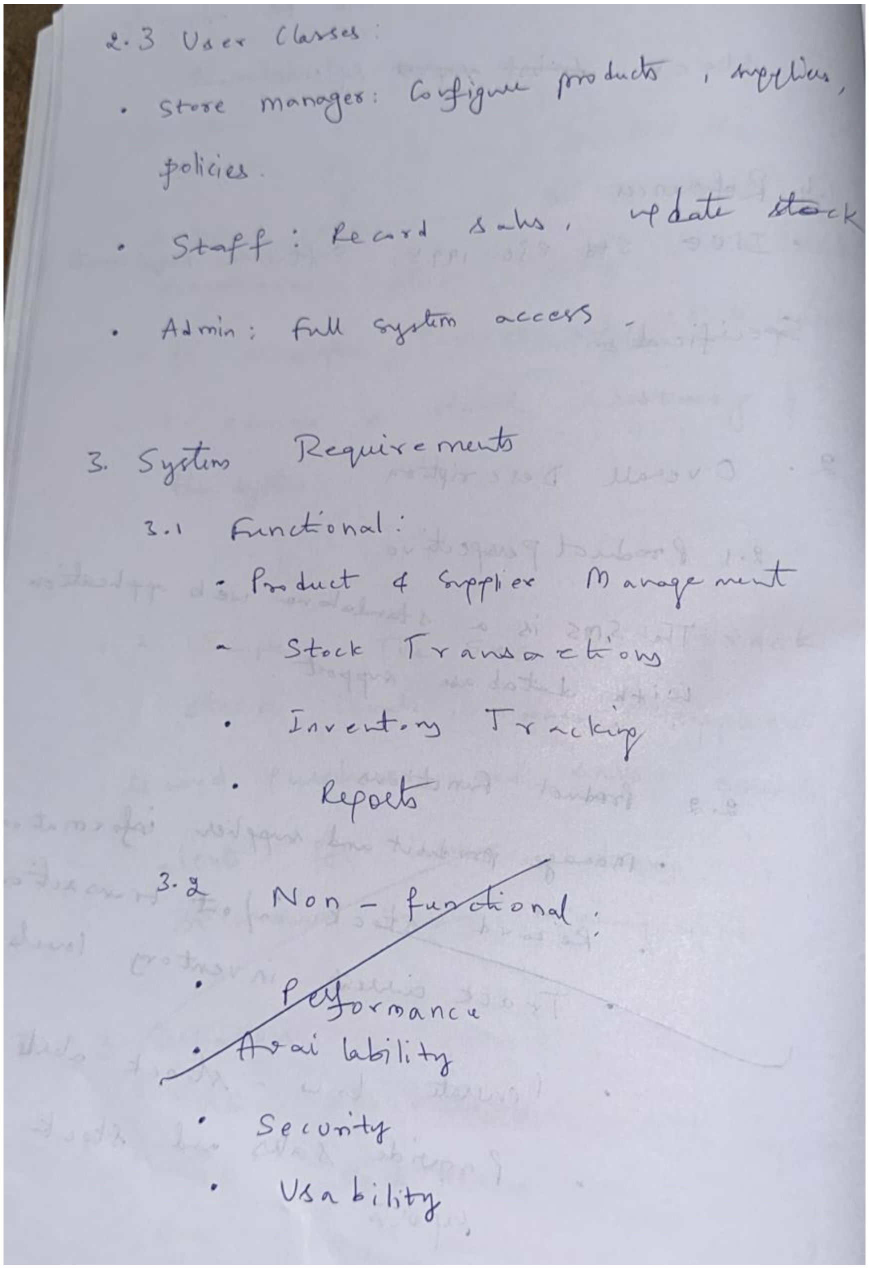
2. Overall Description

2.1 Product perspective

The SMS is a standalone web application with database support.

2.2 Product functions

- Manage product and supplier information
- Record stock in/out transaction
- Track current inventory levels
- Generate low-stock alerts
- Provide sales and stock reports



2.3 User Classes:

- Store manager: Configure products, suppliers, policies.
- Staff: Record sales, update stock
- Admin: Full system access.

3. System Requirements

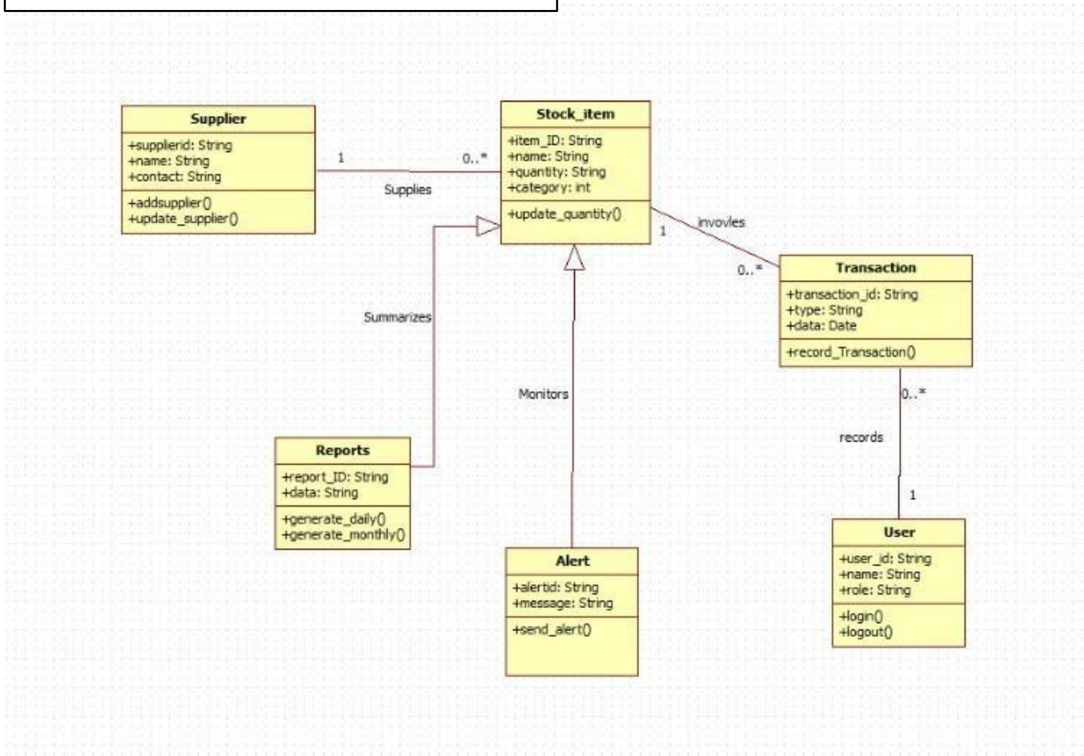
3.1 Functional:

- Product & Supplier Management
- Stock Transactions
- Inventory Tracking
- Reports

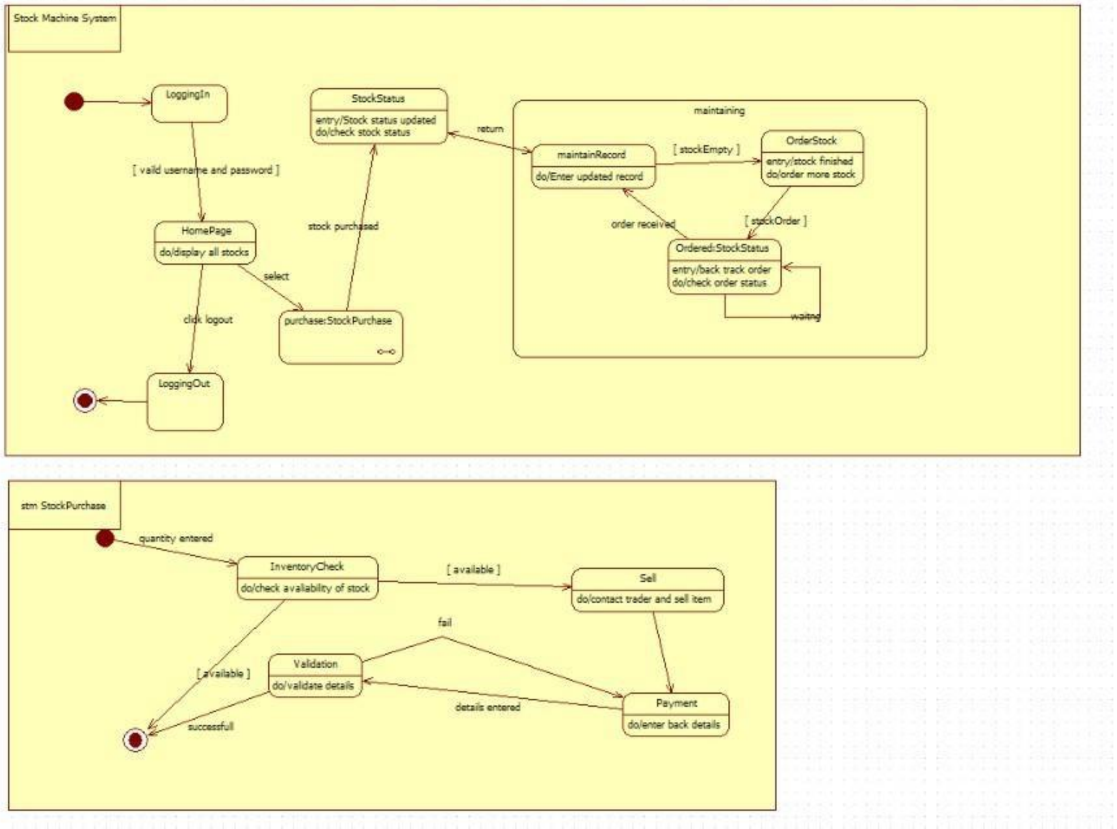
3.2 Non-functional:

- Performance
- Availability
- Security
- Usability

2.3 class model

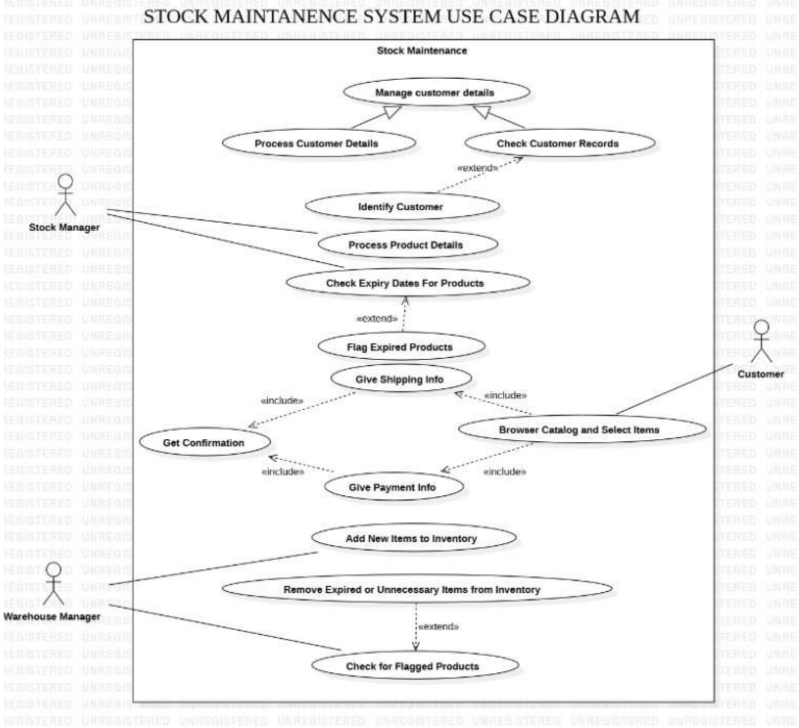


State Model

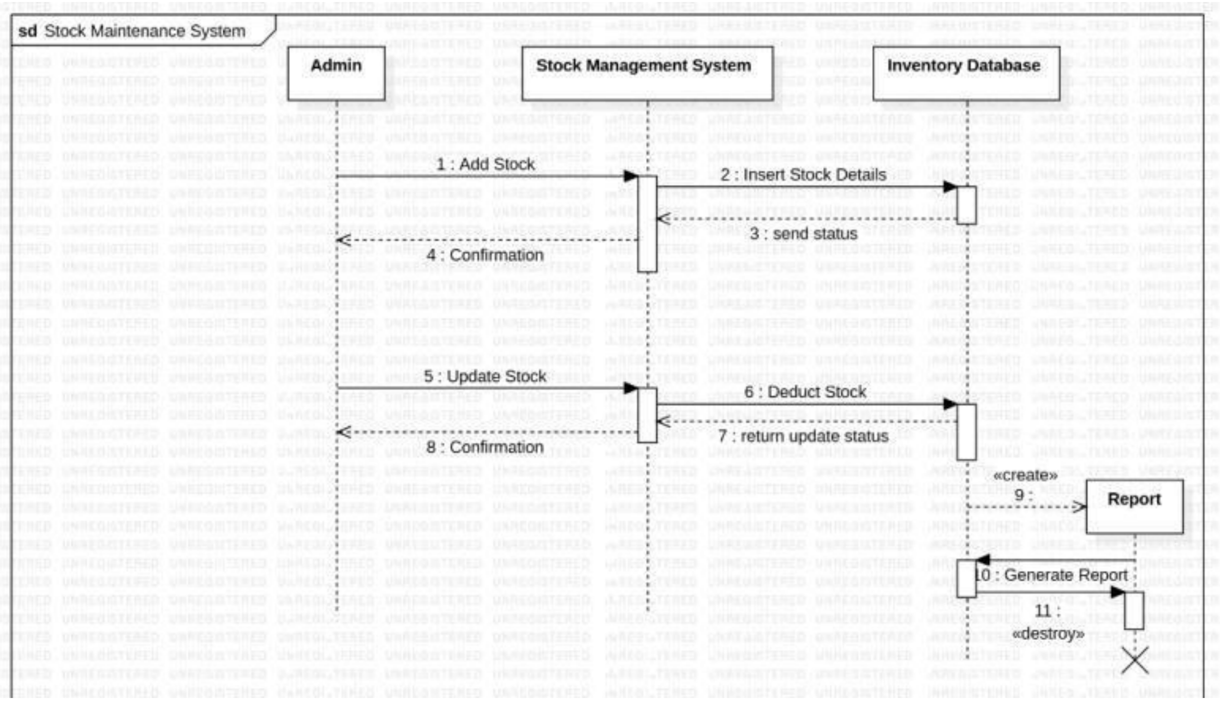

Stock Machine System
LoggingIn
[vaild username and password]
HomePage
do/display all stocks
click logout
select
LoggingOut
purchase:StockPurchase
stock purchased
StockStatus
entry/Stock status updated
do/check stock status
return
maintaining
maintainRecord
do/Enter updated record
[stockEmpty]
OrderStock
entry/stock finished
do/order more stock
order received
[stockOrder]
Ordered:StockStatus
entry/back track order
do/check order status
waiting
stm StockPurchase
quantity entered
InventoryCheck
do/check availability of stock
[available]
Sell
do/contact trader and sell item
fail
Validation
do/validate details
[available]
details entered
Payment
do/enter back details
successfull

4.3 Interaction Models

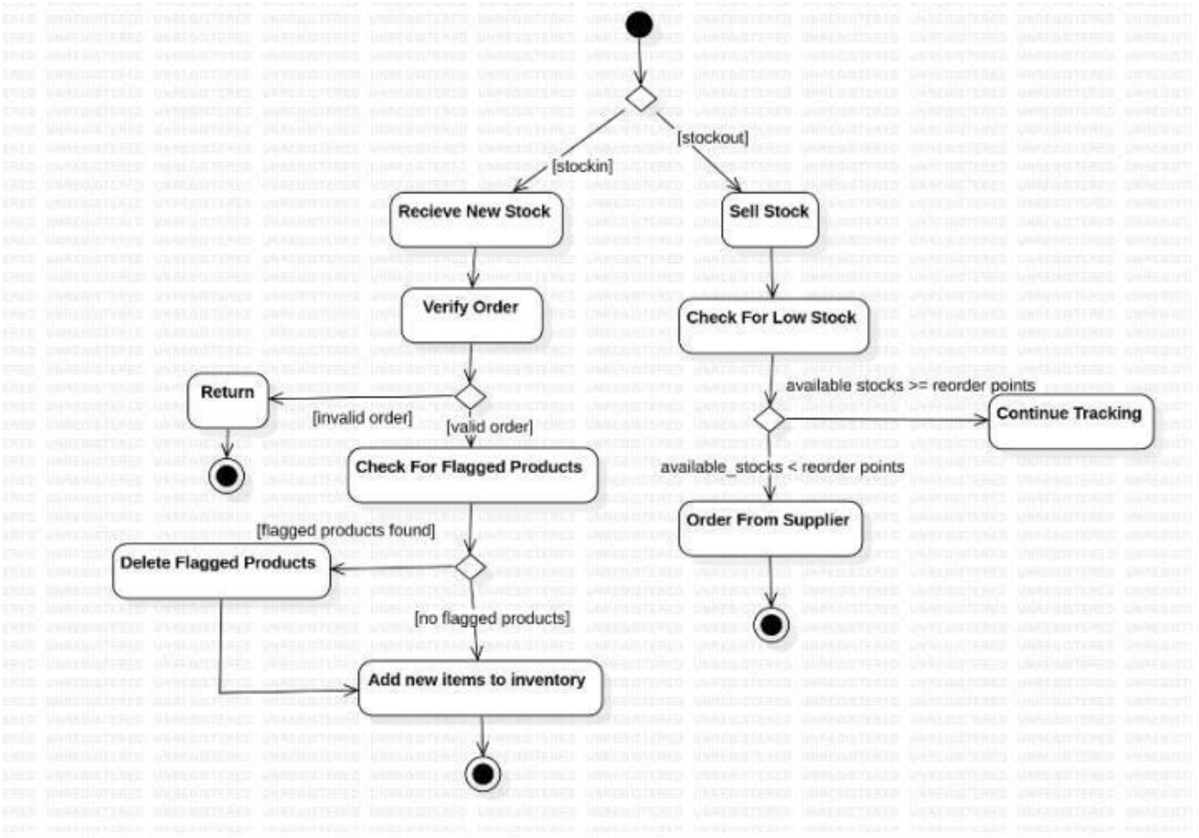
a. Use Case Model



d. Sequence Model

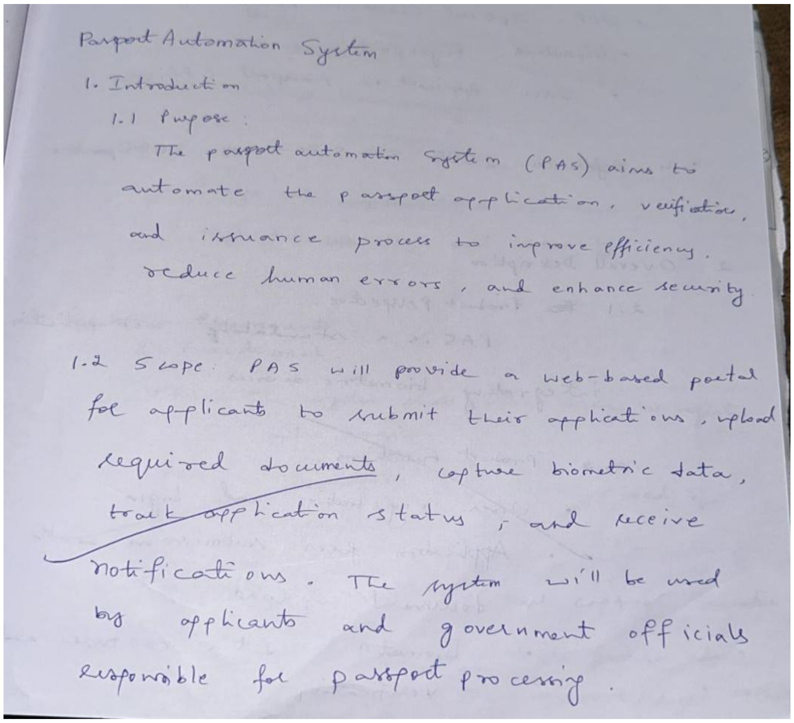


e. Activity Diagram



5 .Passport Automation System

5.1 SRS Document:



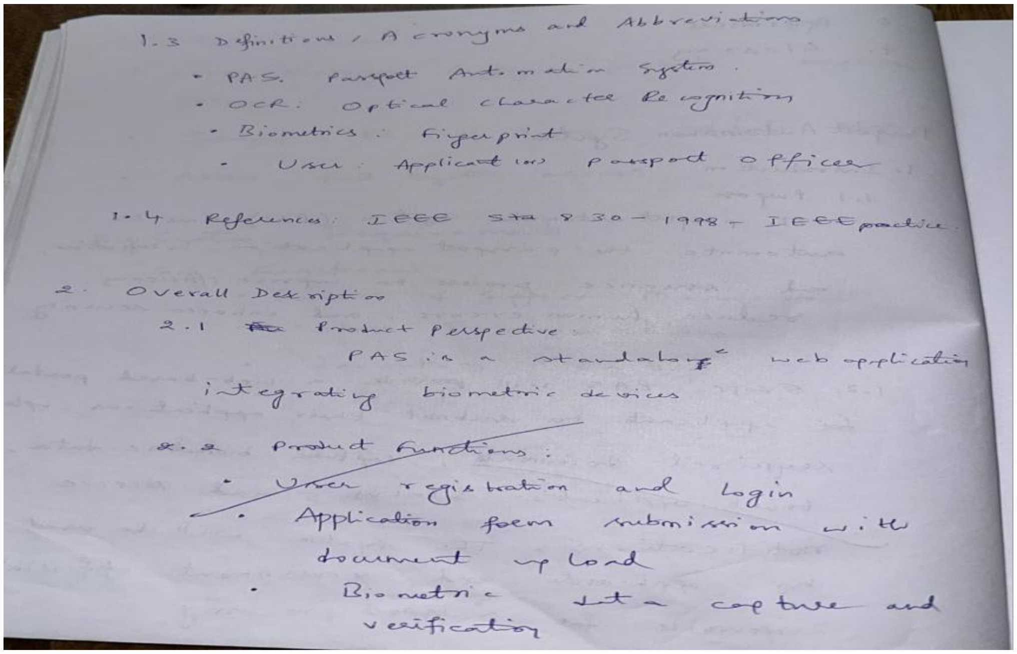
Passport Automation System

1. Introduction

1.1 Purpose:

The passport automation system (PAS) aims to automate the passport application, verification, and issuance process to improve efficiency. reduce human errors, and enhance security.

1.2 Scope: PAS will provide a web-based portal for applicants to submit their applications, upload required documents, capture biometric data, track application status; and receive notifications. The system will be used by applicants and government officials responsible for passport processing.



1.3 Definitions / Acronyms and Abbreviations

- PAS. Passport Automation System.
- OCR: Optical Character Recognition
- Biometrics: Fingerprint
- User: Applicant (or) passport officer

1.4 References: IEEE Std 830-1998 - IEEE practice

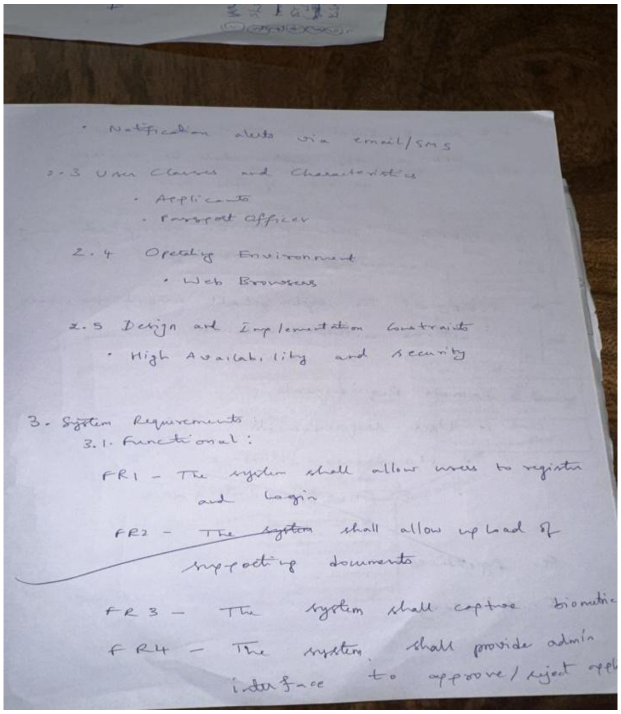
2. Overall Description

2.1 Product Perspective

PAS is a standalone web application integrating biometric devices

2.2 Product Functions:

- User registration and login
- Application form submission with document upload
- Biometric data capture and verification



- Notification alerts via email/SMS

2.3 User Classes and Characteristics

- Applicants
- Passport Officer

2.4 Operating Environment

- Web Browsers

2.5 Design and Implementation Constraints

- High Availability and security

3. System Requirements

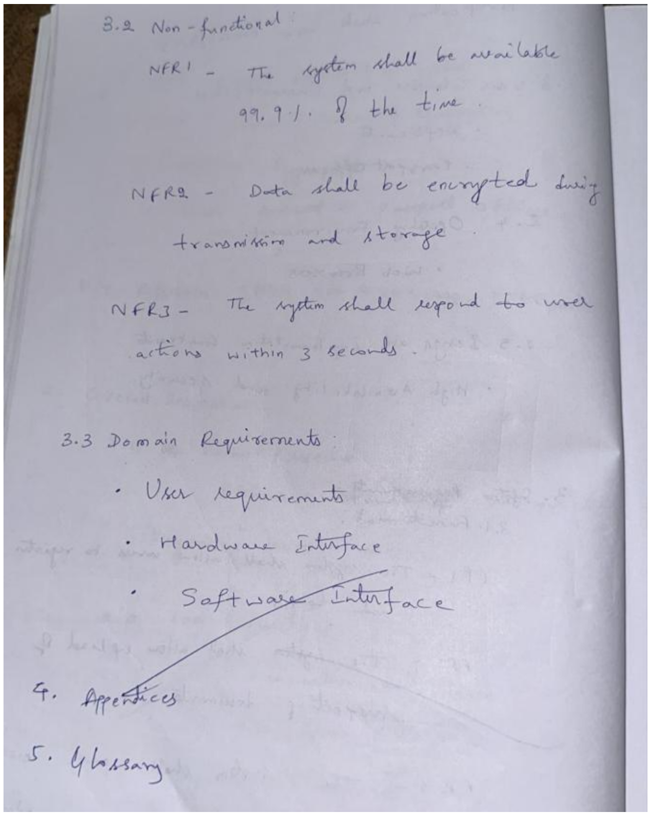
3.1. Functional:

FR1 – The system shall allow users to register and Login

FR2 – The system shall allow upload of supporting documents

FR3 – The system shall capture biometric

FR4 – The system shall provide admin interface to approve/reject appl



3.2 Non-functional

NFR1 – The system shall be available 99.9% of the time.

NFR2 – Data shall be encrypted during transmission and storage.

NFR3 – The system shall respond to user actions within 3 seconds.

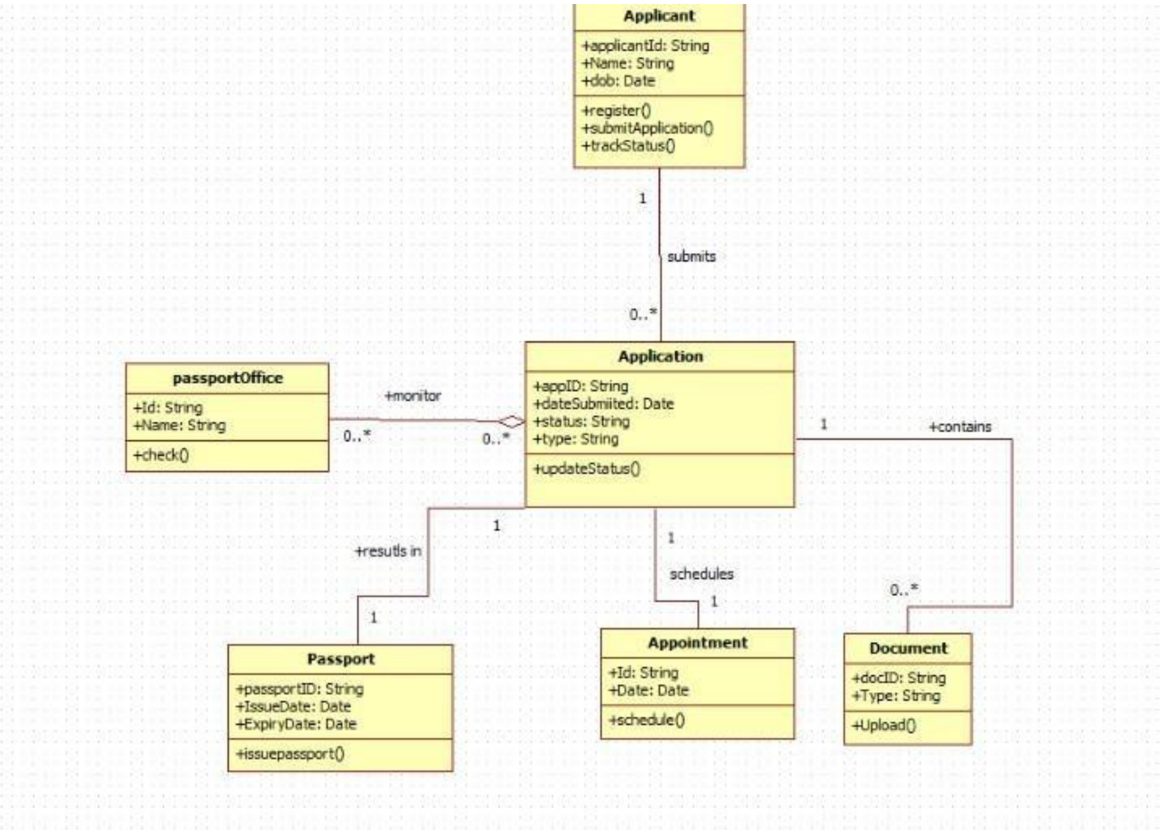
3.3 Domain Requirements

- User requirements
- Hardware Interface
- Software Interface

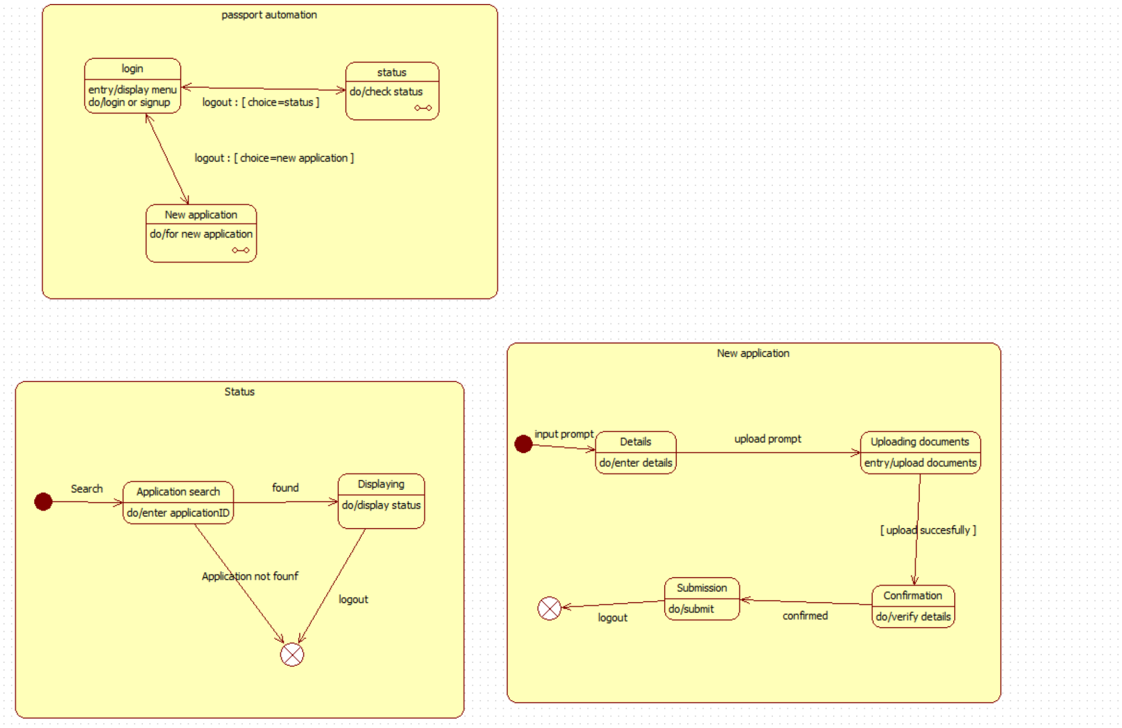
4. Appendices

5. Glossary

Class Model

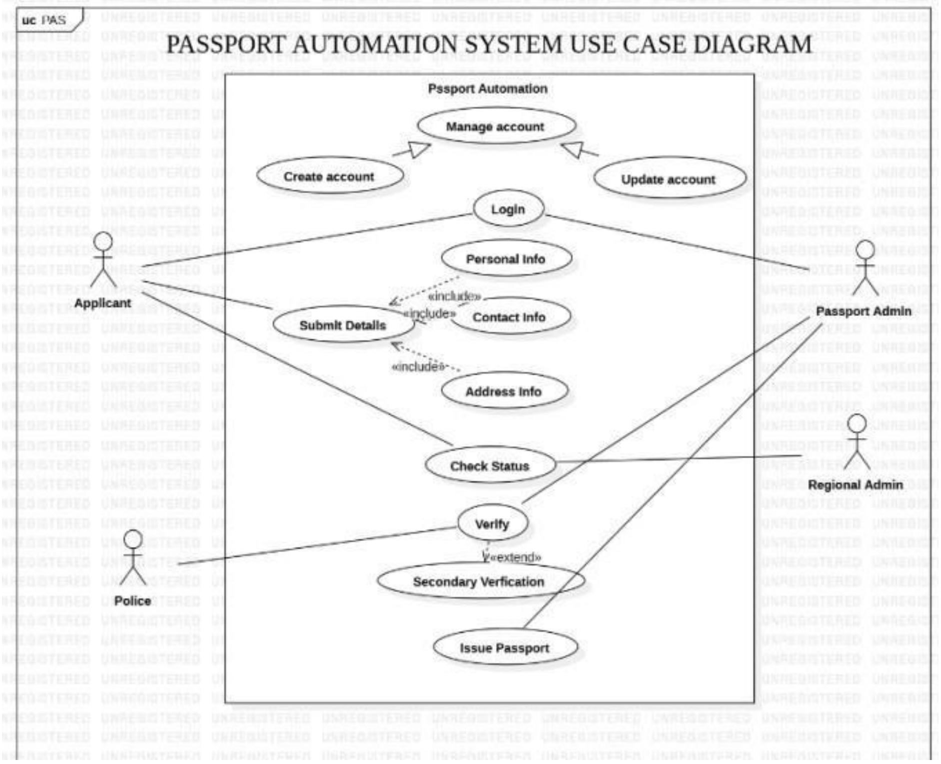


5.2 State Model

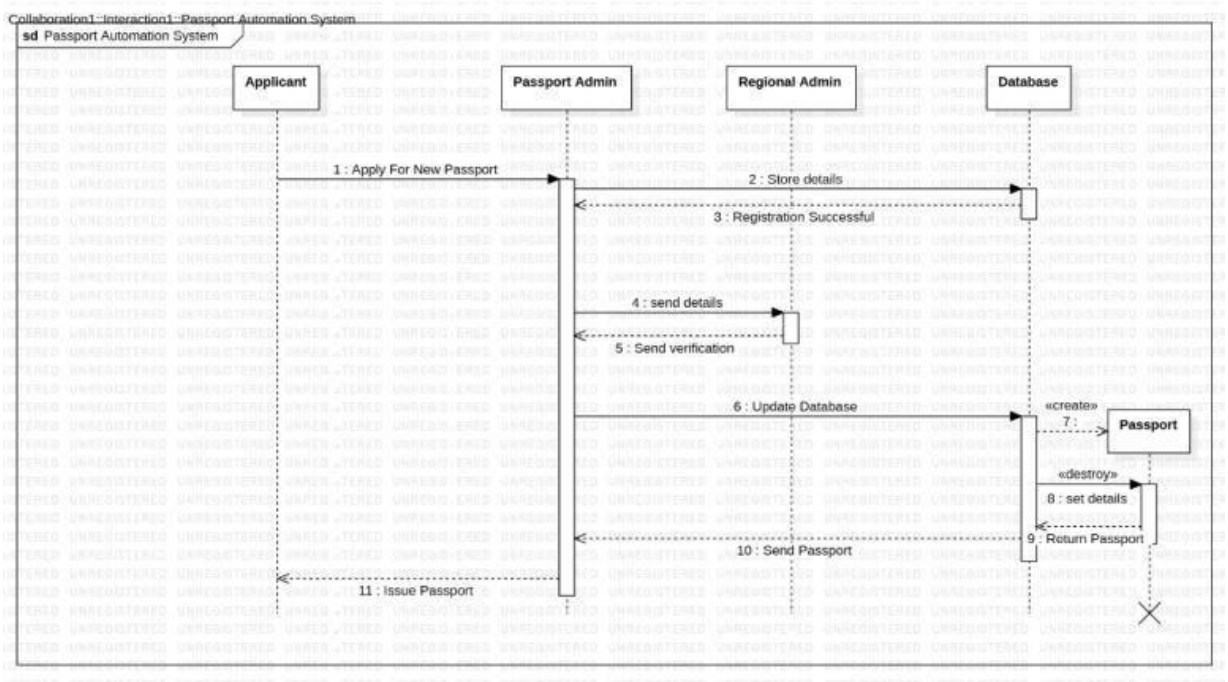


5.3 Interaction Models

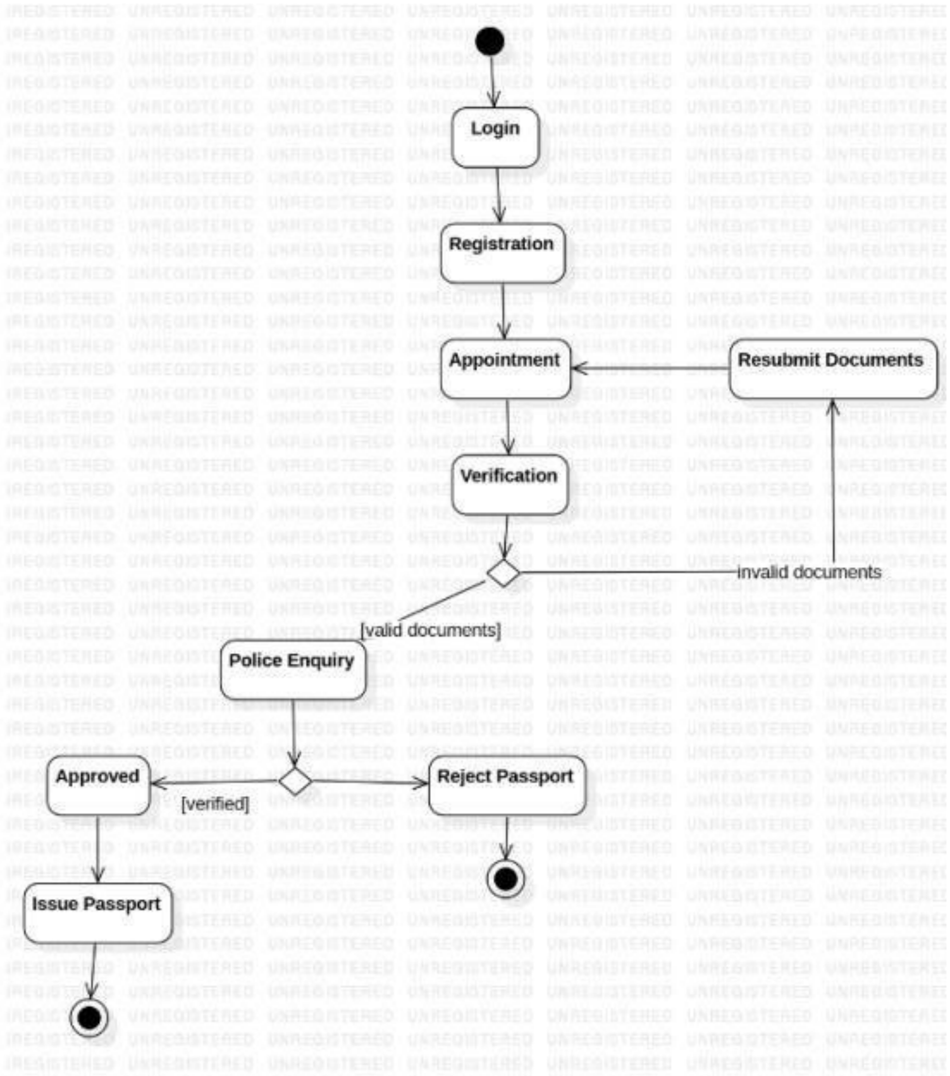
a. Use Case Model

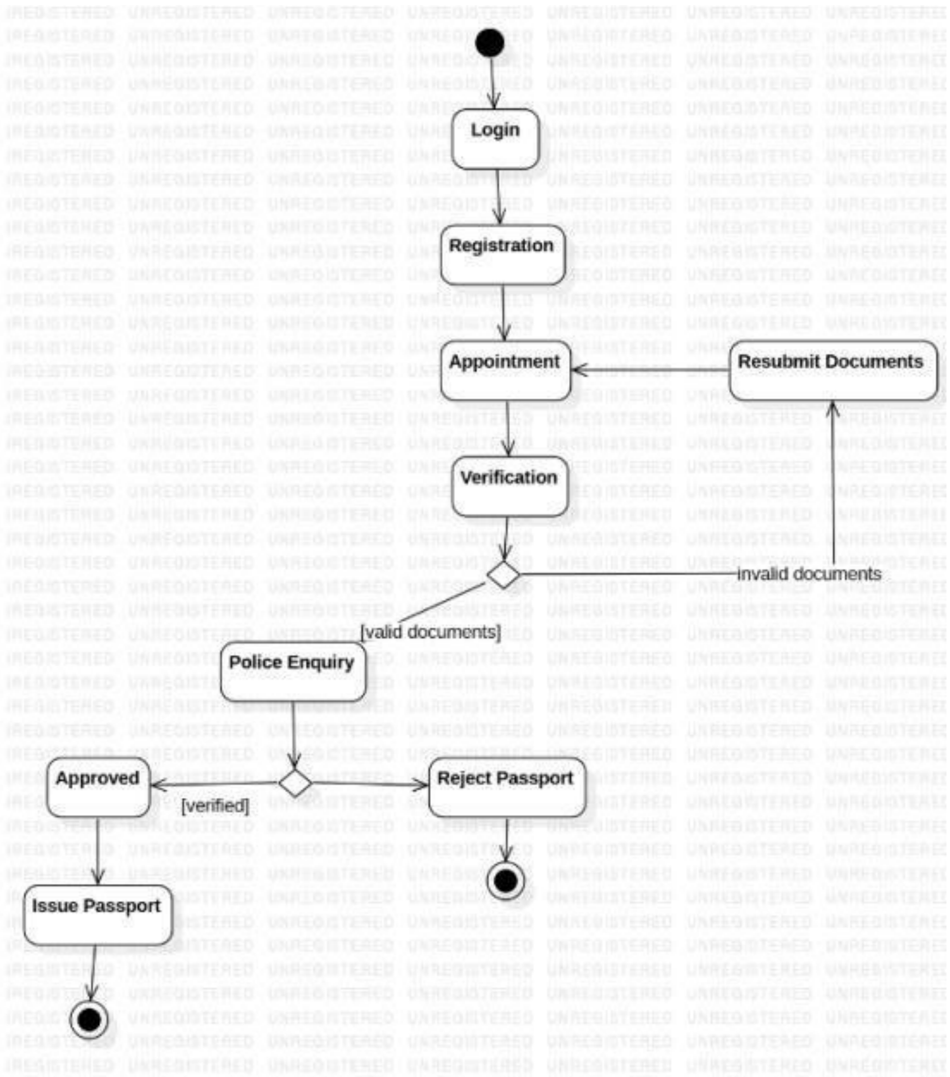


a. Sequence Model



b. Activity model




Login
Registration
Appointment
Resubmit Documents
Verification
invalid documents
[valid documents]
Police Enquiry
Approved
[verified]
Reject Passport
Issue Passport