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

Lab Record

Software Engineering and Object-Oriented Modeling

Submitted in partial fulfillment for the 5th Semester Laboratory

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:

ADITYA
1BM22CS015

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

B.M.S. COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Object-Oriented Analysis and Design(22CS6PCSEO) laboratory has been carried out by Aditya (1BM22CS015) during the 5th Semester Oct24-Jan2025.

Signature of the Faculty In charge:

NAME OF THE FACULTY: Dr.Latha N R

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

Table of Contents

1.Hotel Management System

Problem Statement
SRS-Software Requirements Specification
Class Diagram Requirements: Minimum 7 classes, 4 attributes and 4 operations in each class, associations, association name, association end names, multiplicity, association class, enumeration, qualified association, aggregation, composition, generalization, ordered, sequencing, multiplicity of attribute, brief description of the Class diagram
State Diagram: one simple state diagram, one advance state diagram (either Sub Machine or Nested State, include any one of the concurrency in your design) Requirements: minimum of 6 states for both the state diagram with state name, do activities, events, guard condition, brief description of State diagram
Use-Case Diagram: Minimum 5 use cases to be identified, and design one simple use case diagram and one advanced use case diagram (ie using include,extend and generalization relationship) and explanation as there in the solution manual.
Sequence Diagram: Write the scenario for any two-use case transaction completely. and design the simple sequence diagram with minimum of 5 objects and communication between them. Design advanced sequence diagram using passive and transient objects. Brief explanation for each
Activity Diagram: Design the simple activity diagram showing the algorithm or workflow. Design the advanced activity diagram with swimlanes showing the responsible person for swimlane and also write the brief explanation for both.

Format to be Followed

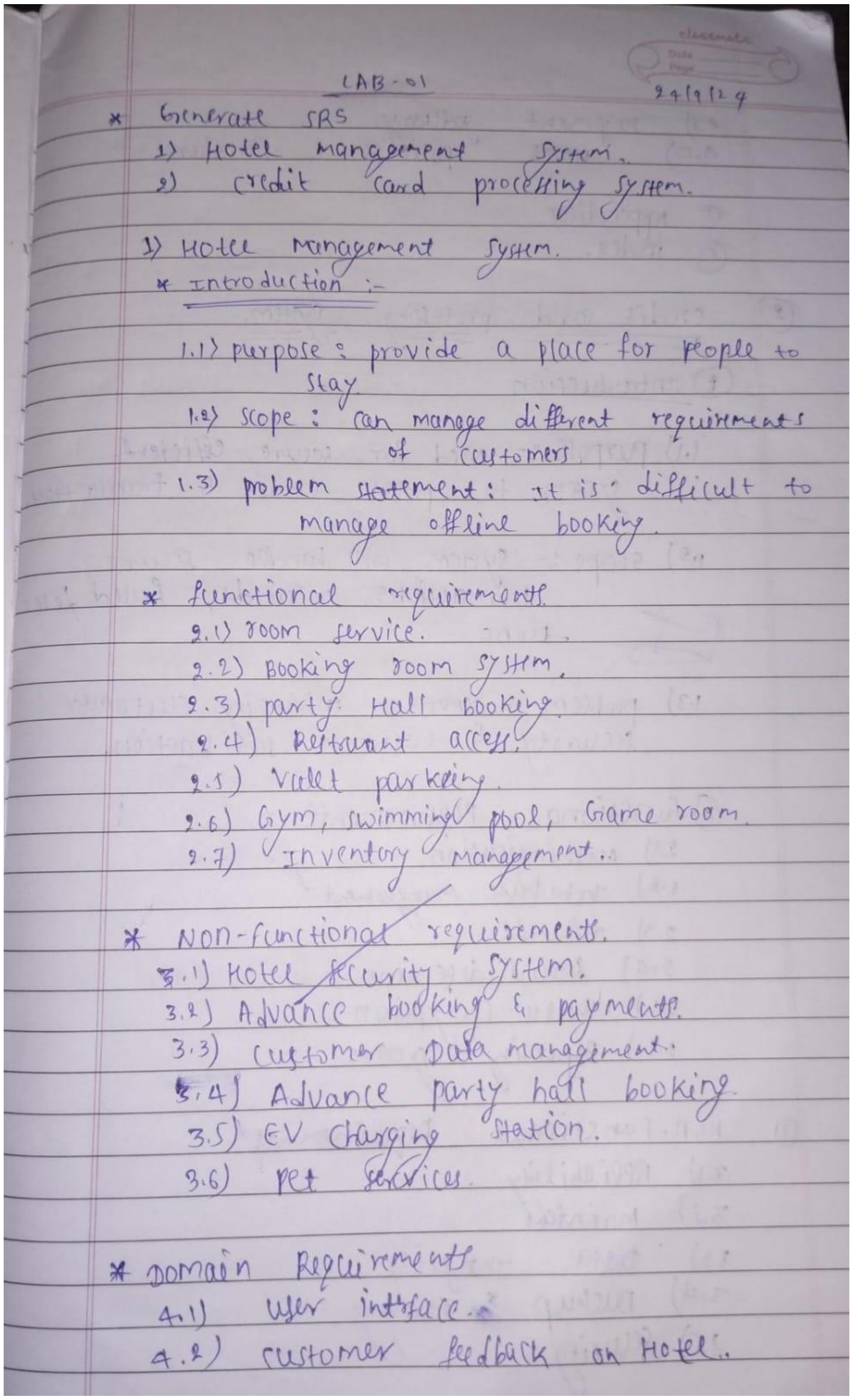
-Line spacing 1.5
-Font: 12 points - Times New Roman
-Margin: 1.5 inch left and 1 inch on all other sides

-Text should be justified
-Lab record report should be soft bounded (Note: Spiral bound will not be accepted). Soft bound should be with brown color cello tape and cream color sheet to be used.
-Figure and Tables must be numbered

1. Hotel Management System

1.1 Problem Statement: Hotels often face challenges in managing their daily operations efficiently, such as handling reservations, managing room availability, billing, staff management, and customer service. Traditional systems or manual methods are time-consuming, error-prone, and do not scale well with growing customer demands.

1.2 SRS-Software Requirements Specification



LAB-01

24/9/24

* Generate SRS
1) Hotel management system.
2) credit card processing system.

1) Hotel Management system.

* Introduction :-

1.1) purpose : provide a place for people to stay.

1.2) scope : can manage different requirements of customers.

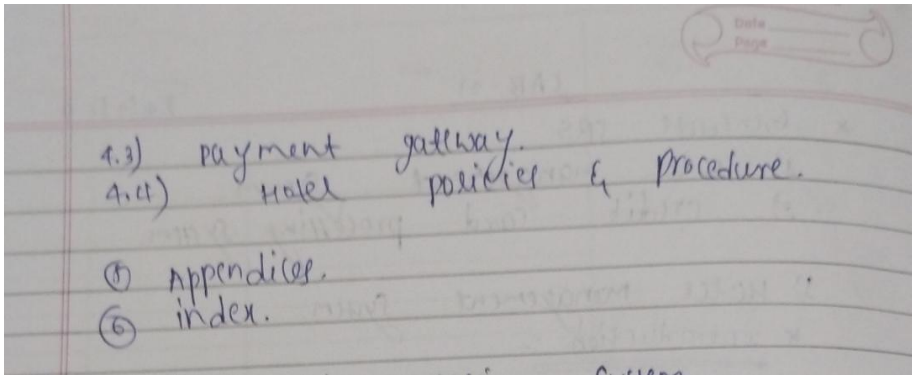
1.3) problem statement : It is difficult to manage offline booking.

* functional requirements.
2.1) room service.
2.2) Booking room system.
2.3) party Hall booking.
2.4) Restaurant access.
2.5) Valet parking.
2.6) Gym, swimming pool, Game room.
2.7) Inventory management.

* Non-functional requirements.
3.1) Hotel security system.
3.2) Advance booking & payments.
3.3) customer Data management.
3.4) Advance party hall booking.
3.5) EV charging station.
3.6) pet services.

* Domain Requirements
4.1) user interface.
4.2) customer feedback on Hotel.

Fig 1.2.1



4.3) payment gateway.
4.4) Hotel policies & procedure.

5) appendices.
6) index.

Fig 1.2.2

1.3 Class Diagram

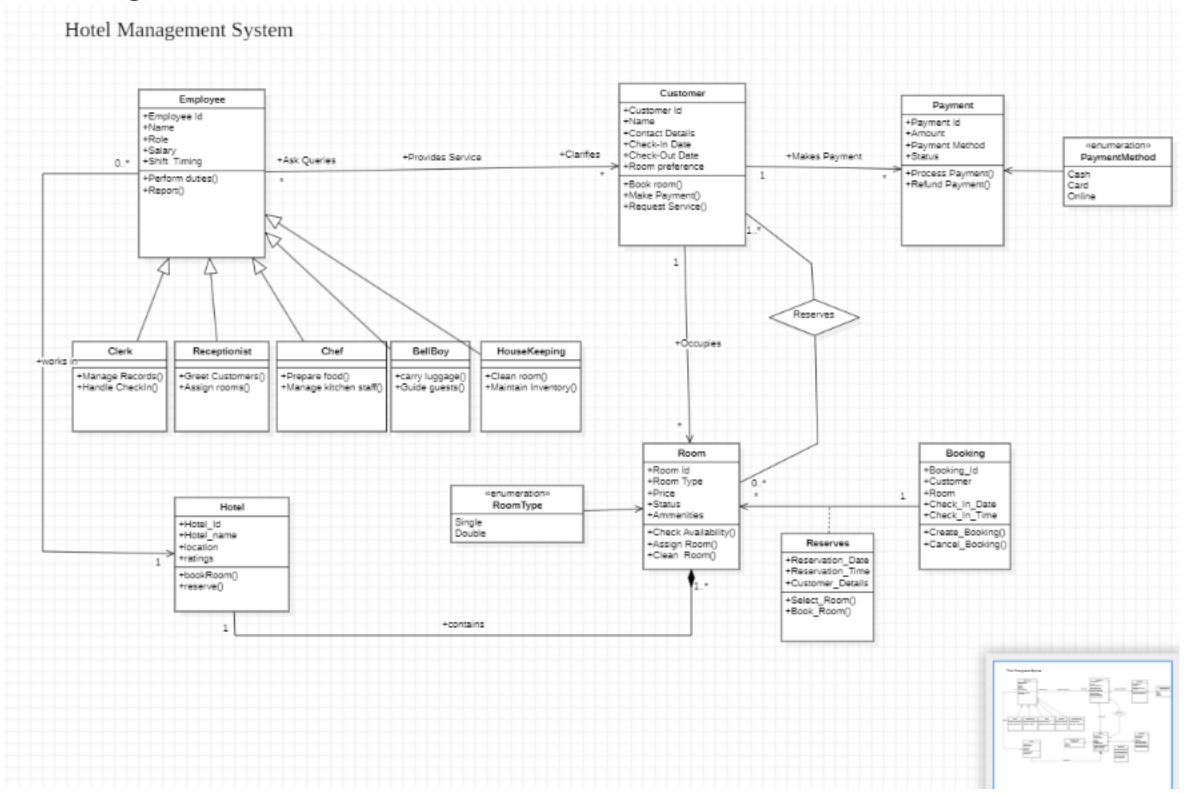


Fig 1.3.1

The class diagram represents the structure of a **Hotel Management System**, detailing the relationships and interactions among various entities. The primary entities include Employee, Customer, Room, Booking, Reserves, Payment, and Hotel. The Employee class serves as the parent for roles such as Clerk, Receptionist, Chef, BellBoy, and Housekeeping, each having specific duties like managing records, assigning rooms, preparing food, assisting guests, and maintaining room cleanliness. The Customer class holds details such as name, contact information, check-in/out dates, and room preferences, along with methods for booking rooms, making payments, and requesting services. The Room class defines attributes like room type (Single or Double), price, status, and amenities, and includes methods for checking availability, assigning, and cleaning rooms.

The Booking class manages the room booking process with details such as booking ID, customer, room, and check-in time, while the Reserves class handles room reservations and facilitates the selection and booking of rooms. The Payment class manages financial transactions, supporting various payment methods (cash, card, or online) and processes such as payment and refund handling. The Hotel class encapsulates the overall hotel entity, including details like the hotel name, location, and ratings, and connects to rooms for booking and reservation purposes. The diagram demonstrates how employees provide services to customers, who, in turn, reserve and occupy rooms while managing payments, ensuring a structured and efficient hotel management workflow.

1.4 State Diagram

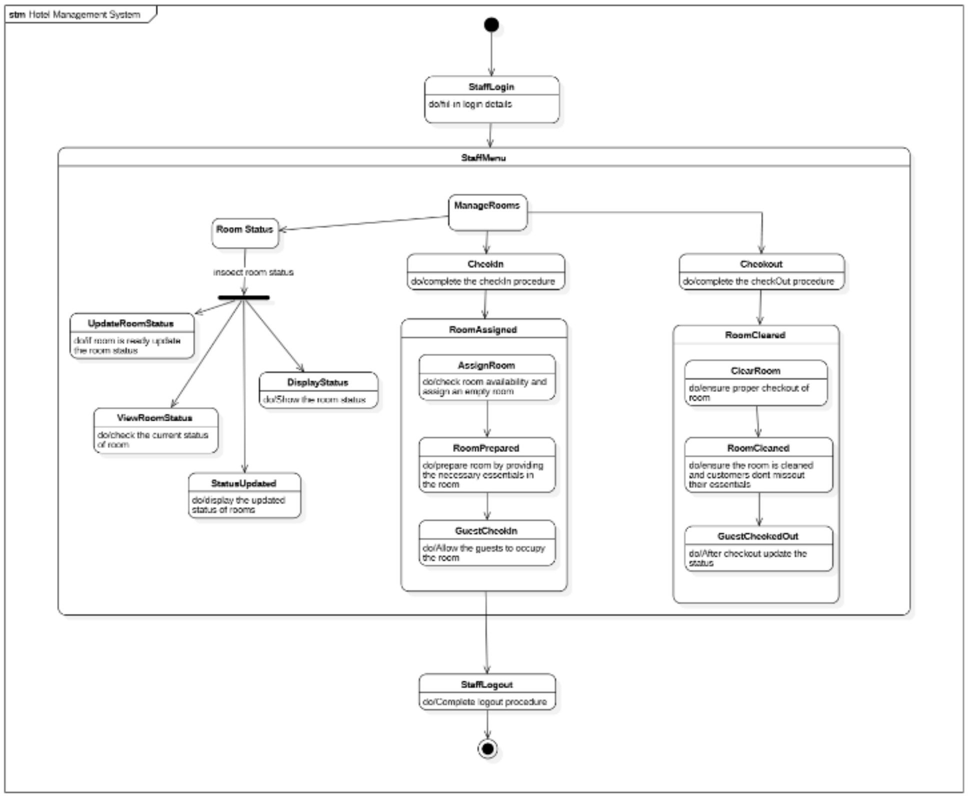

stm Hotel Management System
StaffLogin
do/fill in login details
StaffMenu
ManageRooms
Room Status
inspect room status
UpdateRoomStatus
do/if room is ready update the room status
DisplayStatus
do/Show the room status
ViewRoomStatus
do/check the current status of room
StatusUpdated
do/display the updated status of rooms
CheckIn
do/complete the checkIn procedure
RoomAssigned
AssignRoom
do/check room availability and assign an empty room
RoomPrepared
do/prepare room by providing the necessary essentials in the room
GuestCheckIn
do/Allow the guests to occupy the room
Checkout
do/complete the checkOut procedure
RoomCleared
ClearRoom
do/ensure proper checkout of room
RoomCleaned
do/ensure the room is cleaned and customers dont missout their essentials
GuestCheckedOut
do/After checkout update the status
StaffLogout
do/Complete logout procedure

Fig 1.4.1

The state diagram represents a Hotel Management System and outlines the workflow for managing hotel operations efficiently. It begins with the **Staff Login**, where employees access the system. The primary state is the **Staff Menu**, which branches into three key functions: **Room Status**, **Check-In**, and **Check-Out**. In **Room Status**, staff can view, update, and display the current status of rooms (e.g., vacant, occupied, or under maintenance). The **Check-In** process involves assigning a room, preparing it with necessary essentials, and allowing guests to occupy it. The **Check-Out** process handles clearing the room, cleaning it, and updating its status for the next guest. Finally, the system transitions to the **Staff Logout**, where employees log out after completing their tasks. This diagram ensures a structured approach to hotel management, covering room readiness, guest handling, and status updates.

1.5 Use Case Diagram

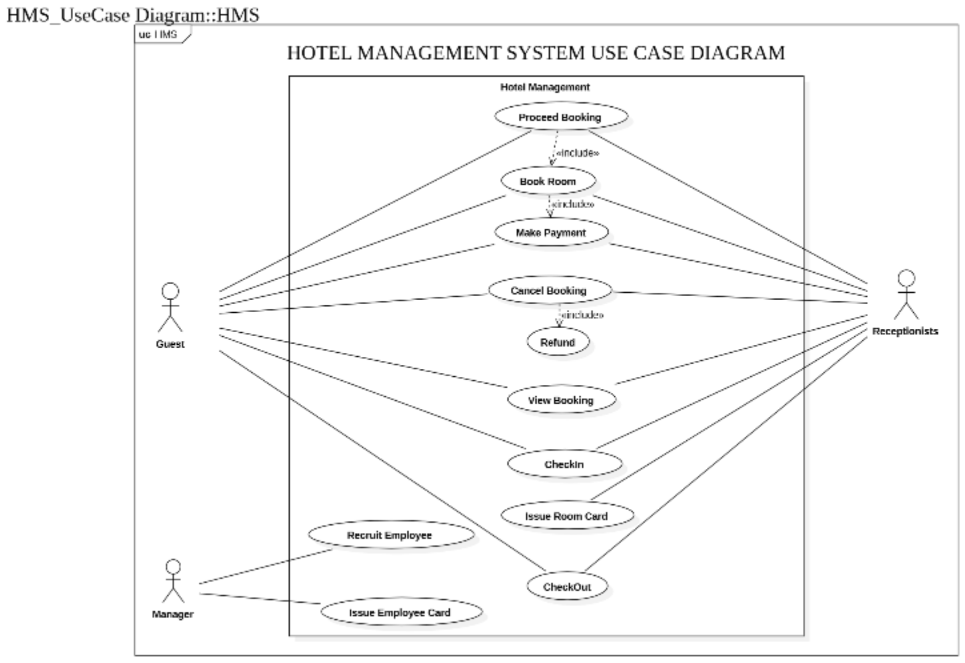


Fig 1.5.1

The use case diagram illustrates the functionality of a Hotel Management System by detailing the interactions between different actors and system processes. The primary actors include **Guests**, **Receptionists**, and a **manager**. Guests can perform actions such as booking rooms, making payments, cancelling bookings, requesting refunds, viewing bookings, checking in, and checking out. Receptionists facilitate these operations by assisting with room bookings, issuing room cards, and managing the check-in and check-out processes. The **Manager** has additional responsibilities like recruiting employees and issuing employee cards. Key use cases such as **Proceed**

Booking, **Book Room**, and **Make Payment** are interconnected to ensure a smooth booking workflow. The system is designed to handle cancellations and refunds effectively while maintaining efficient guest services. This diagram provides a clear overview of the roles and responsibilities within the hotel management workflow.

1.6 Sequence Diagram

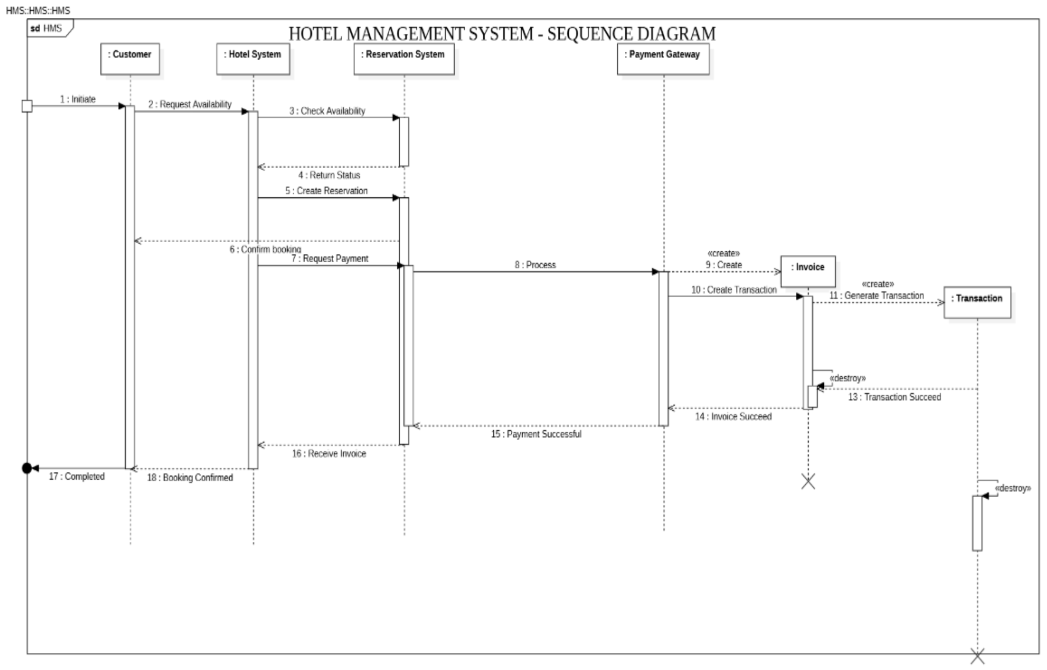


Fig 1.6.1

The sequence diagram illustrates the booking and payment process in a Hotel Management System. It begins with a **Customer** initiating a request to check room availability through the **Hotel System**, which communicates with the **Reservation System** to verify availability. The **Reservation System** returns the status to the **Hotel System**, and if a room is available, a reservation is created. The **Hotel System** then confirms the booking and sends a payment request to the **Payment Gateway**. The **Payment Gateway** processes the payment by generating an invoice and a transaction. Once the transaction is successful, the invoice is returned to the **Hotel System**, which acknowledges the payment as successful. Finally, the customer receives the booking confirmation, marking the process as completed. This diagram ensures a streamlined interaction between customer requests, reservation handling, and secure payment processing.

1.7 Activity Diagram

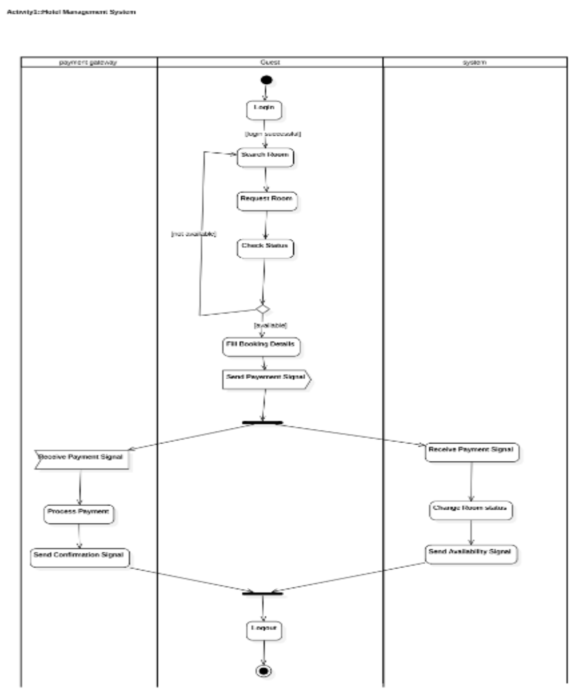


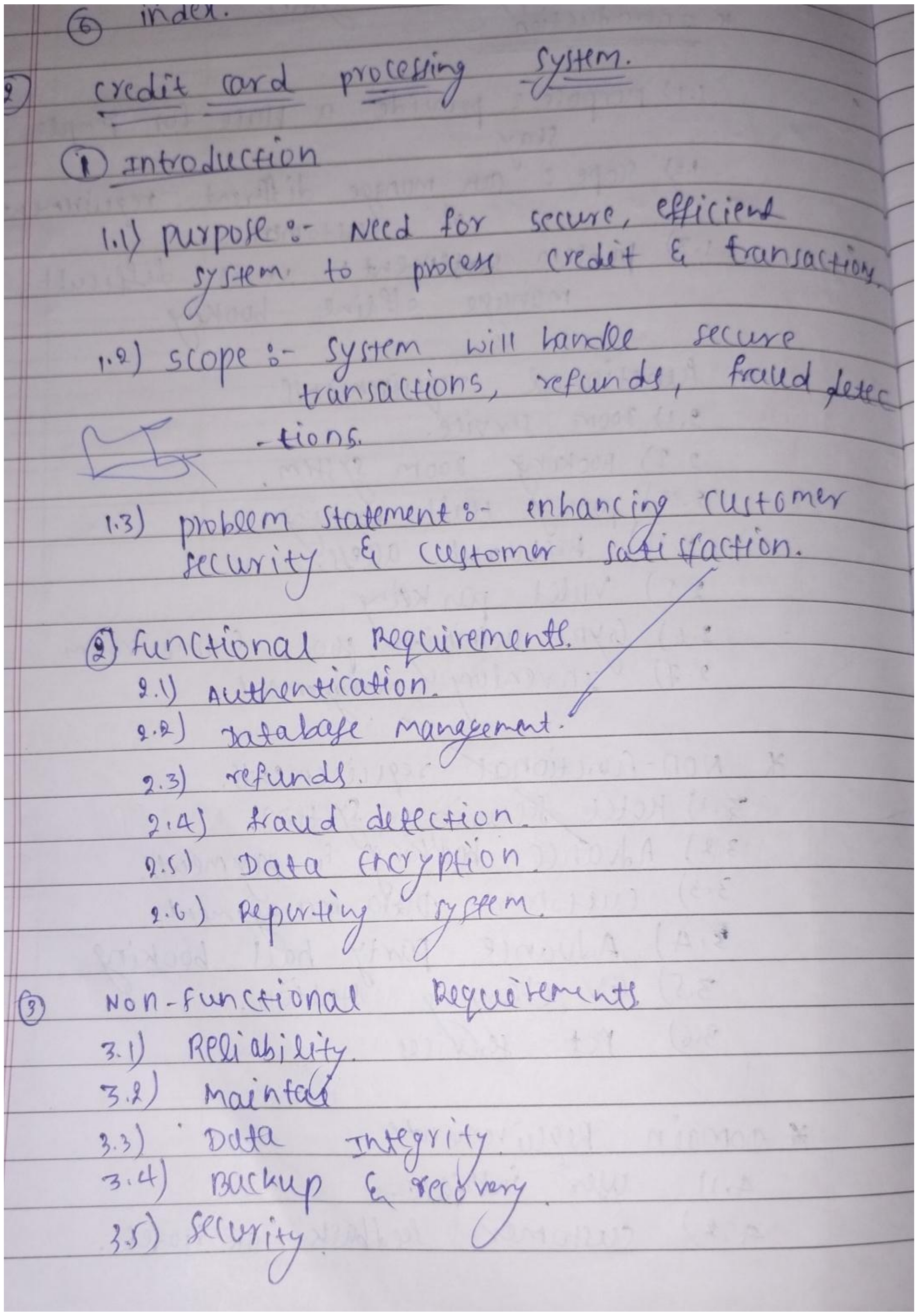
Fig 1.7.1

This activity diagram depicts the activity flow for a hotel room booking process. It involves three participants: the guest, the system, and the payment gateway. The guest begins by logging in and searching for available rooms. Upon finding a suitable room, they request it, and the system checks its availability. If available, the guest fills in booking details and sends a payment signal. This triggers parallel actions: the payment gateway processes the payment and sends a confirmation signal if successful, while the system simultaneously changes the room's status to booked and sends an availability signal. Finally, upon receiving both signals, the process concludes with the guest logging out. This diagram illustrates the sequential and parallel activities involved in a typical online hotel room booking scenario.

2. Credit Card Processing System

2.1 Problem Statement: Financial institutions and merchants face challenges in securely and efficiently managing credit card transactions. The process involves verifying customer details, processing payments, fraud detection, and ensuring regulatory compliance. An unreliable or inefficient system can lead to transaction delays, increased risks of fraud, and poor customer experience.

2.2 SRS-Software Requirements Specification



(6) index.

(2) credit card processing system.

(1) Introduction

1.1) purpose :- Need for secure, efficient system to process credit & transactions.

1.2) scope :- System will handle secure transactions, refunds, fraud detec -tions.

1.3) problem statement :- enhancing customer security & customer satisfaction.

(2) functional Requirements.

2.1) Authentication.

2.2) Database management.

2.3) refunds.

2.4) fraud detection.

2.5) Data Encryption.

2.6) Reporting system.

(3) Non-functional Requirements

3.1) Reliability.

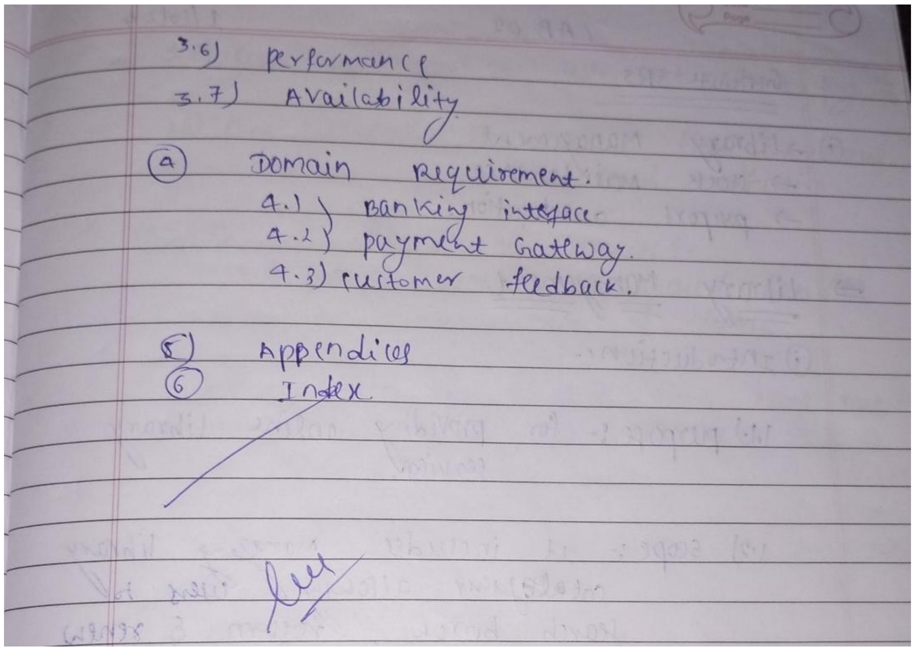
3.2) Maintain

3.3) Data Integrity.

3.4) Backup & recovery.

3.5) Security.

Fig 2.2.1


3.6) performance
3.7) Availability

4 Domain requirement.
4.1) Banking interface
4.2) payment Gateway.
4.3) customer feedback.

5) Appendices
6 Index

[illegible]

Fig 2.2.2

2.3 Class Diagram

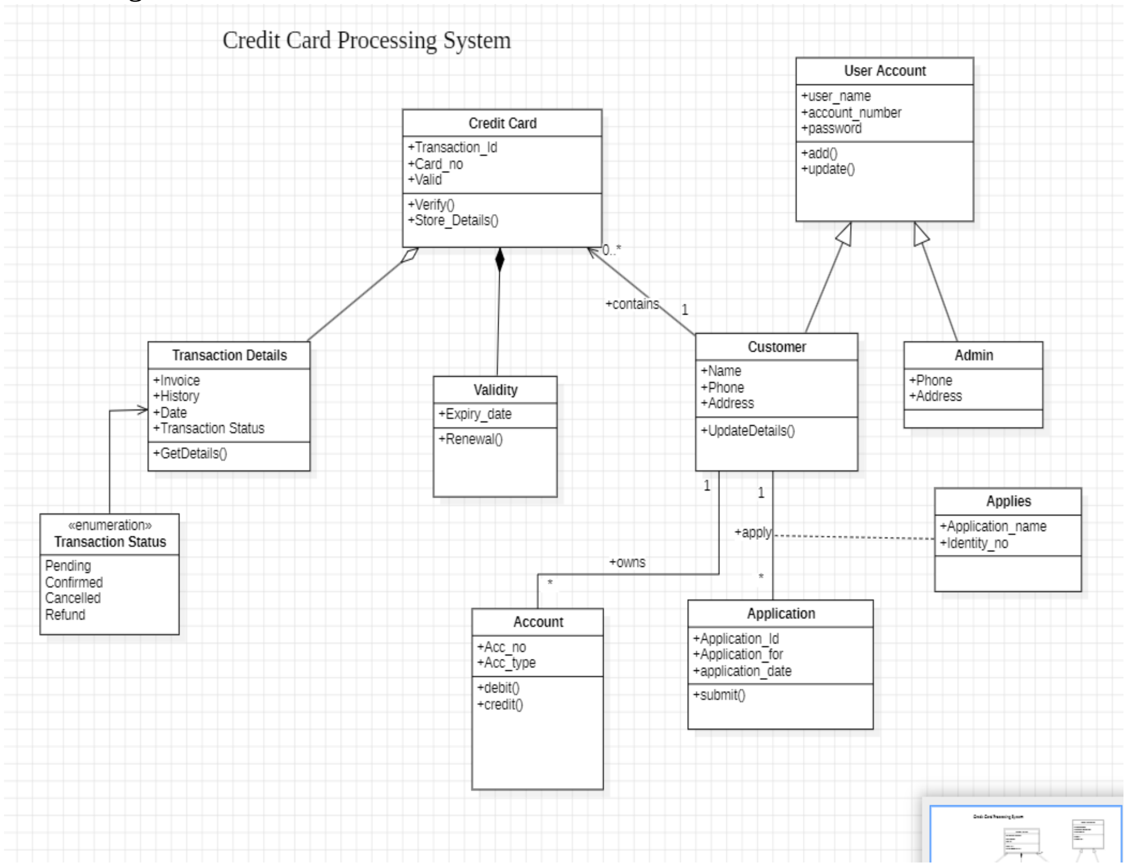


Fig 2.3.1

This UML class diagram models a credit card processing system. It shows that a Transaction uses a Credit Card and contains Transaction Details. Each Credit Card has a Validity status. A Customer can have multiple Accounts and submit Applications for credit cards, which are also linked to an Account. An Admin manages Customers and their Applications. The diagram also lists attributes for each class, like card number for Credit Card and transaction date for Transaction, as well as operations like verify() for Credit Card, debit() and credit() for Account, and submit() for Application. This structure defines the core components and their interactions within the credit card processing system.

2.4 State Diagram

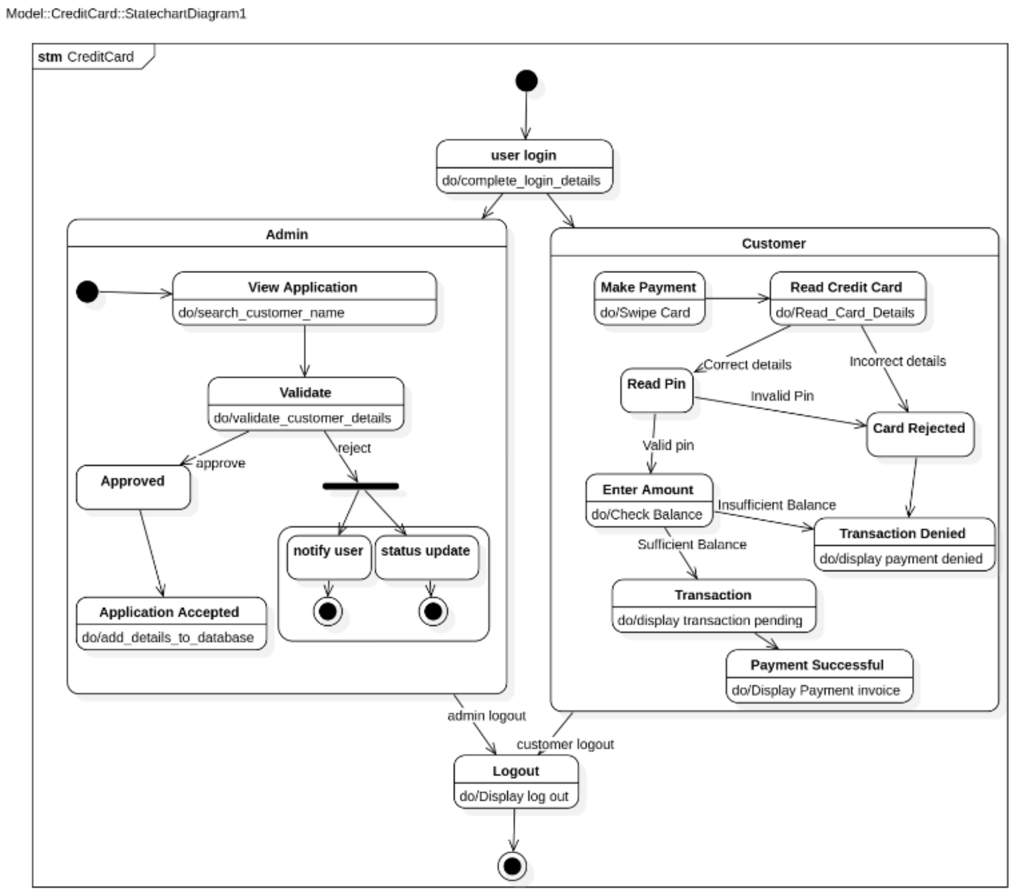


Fig 2.4.1

This state diagram models user interaction with a credit card system. Users begin by logging in, then proceed as either a Customer or Admin. Customers can "Make Payment" (involving card reading, PIN entry, and amount confirmation, leading to either successful payment or transaction denial) or "View Application." Admins can "View Application," which includes searching for applications and subsequently approving or rejecting them, updating the application status. Actions like reading card details or displaying payment invoices are associated with state transitions. The diagram clearly outlines the different paths and possible outcomes for both customer and admin users within the system.

2.5 Use Case Diagram

CPS_UseCase Diagram::CPS

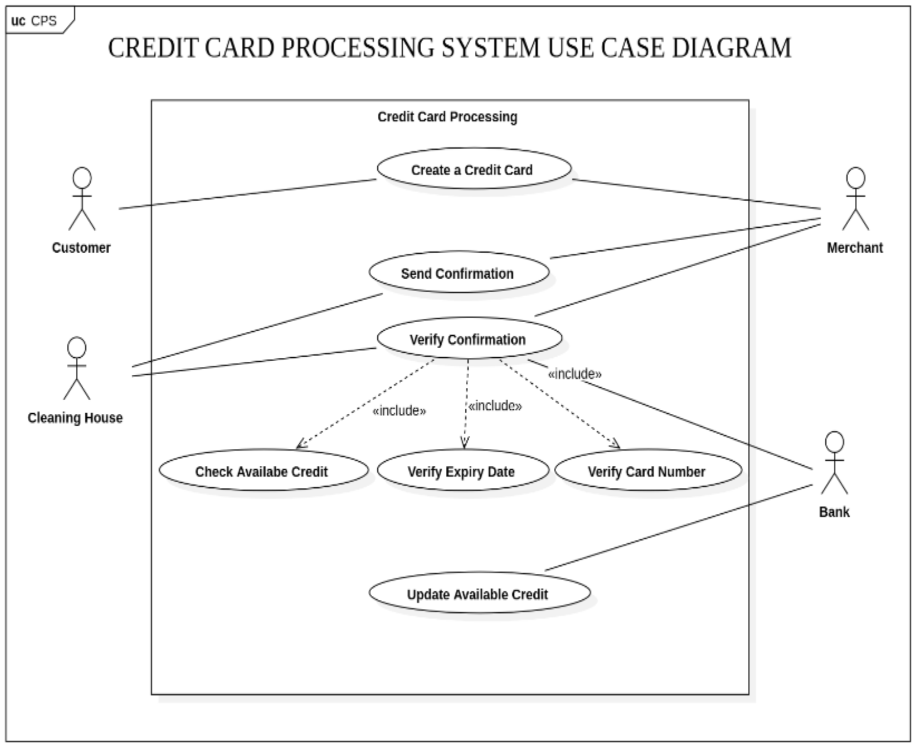


Fig 2.5.1

This use case diagram illustrates interactions within a Credit Card Processing System (CPS). It shows four actors: Customer, Merchant, Cleaning House (likely for transaction reconciliation), and Bank. The primary use case is "Credit Card Processing," which involves several included use cases. The Customer can "Create a Credit Card." The Merchant "Sends Confirmation" of a transaction, which triggers the "Verify Confirmation" use case. This verification process *includes* three sub-processes handled by the Bank: "Check Available Credit," "Verify Expiry Date," and "Verify Card Number." Finally, the Bank also "Updates Available Credit" after a transaction. The diagram highlights the dependencies and interactions between the different actors and their respective roles in the credit card transaction process.

2.6 Sequence Diagram

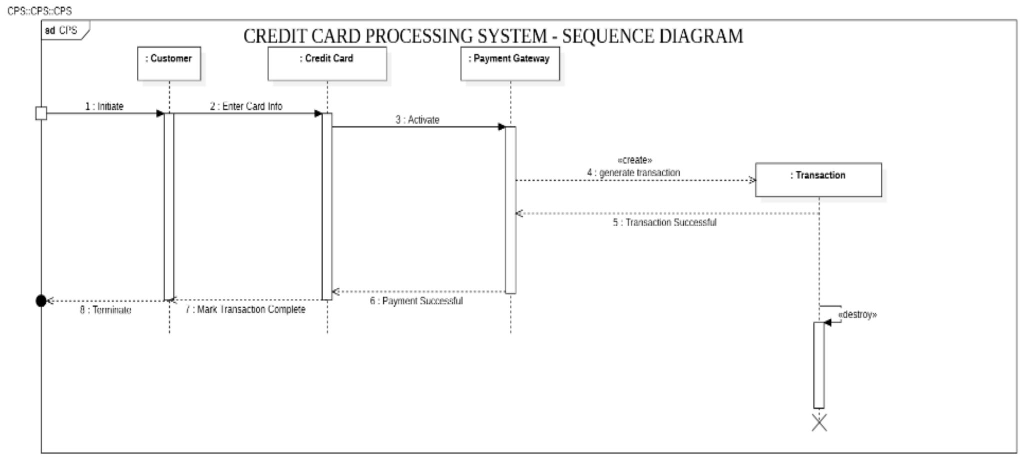


Fig 2.6.1

This sequence diagram illustrates the flow of a credit card transaction. It involves three participants: Customer, Credit Card (representing the card itself/card reader interaction), and Payment Gateway. The process begins with the Customer initiating a transaction (1: Initiate). The Customer then enters their card information (2: Enter Card Info), which activates the Payment Gateway (3: Activate). The Payment Gateway then generates a new Transaction object (4: generate transaction). Assuming the transaction is successful, the Payment Gateway sends a success message (5: Transaction Successful) back to the Customer. The Customer then marks the transaction as complete (Mark Transaction Complete), and the Payment Gateway confirms payment success (6: Payment Successful). Finally, the Transaction object is destroyed, completing the sequence. In short, the diagram shows the step-by-step interaction between the customer, their card, and the payment system during a successful transaction.

2.7 Activity Diagram

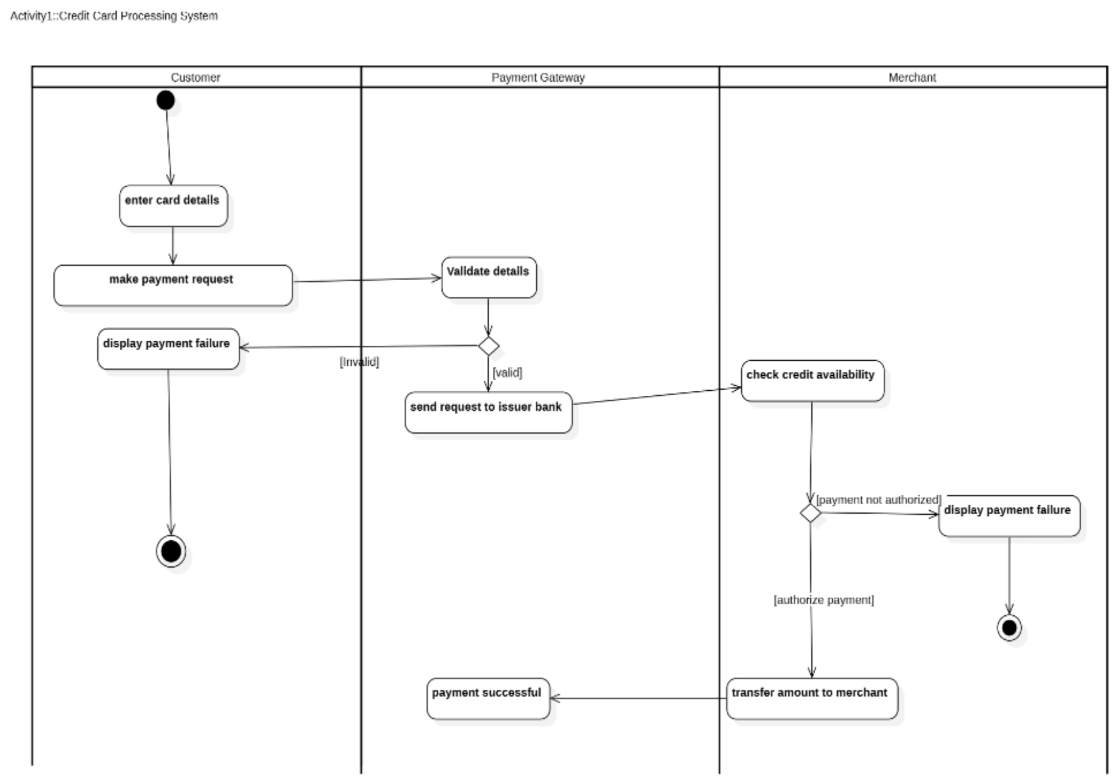


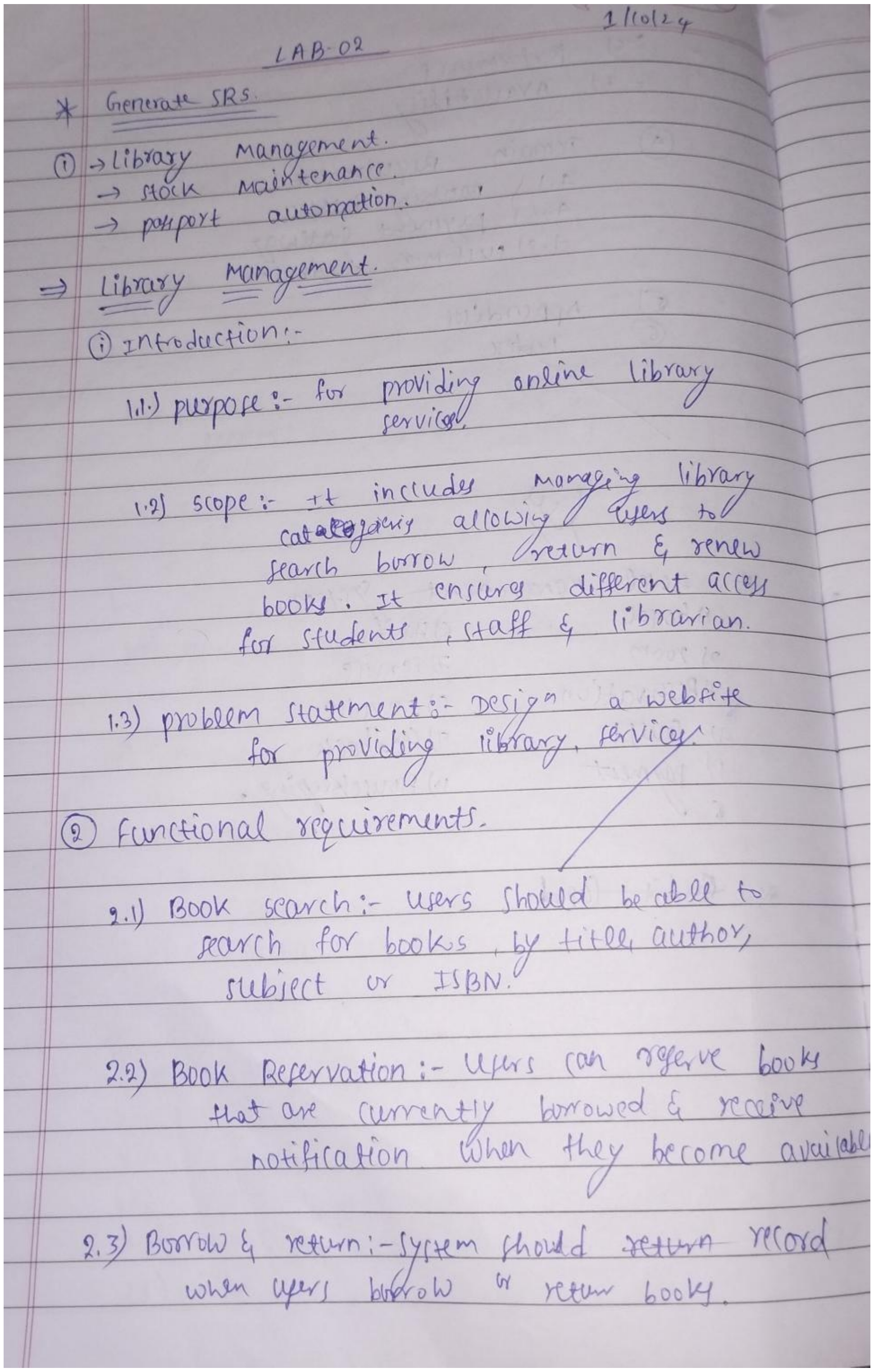
Fig 2.7.1

This activity diagram depicts the flow of a credit card transaction involving a Customer, Payment Gateway, and Merchant. The Customer begins by entering card details and making a payment request. The Payment Gateway then validates these details. If invalid, a payment failure is displayed to the Customer, and the process ends. If the details are valid, the Payment Gateway sends a request to the issuer bank, represented here by the Merchant's action of checking credit availability. If the payment is not authorized, a payment failure is displayed to the Merchant, and the process ends. However, if the payment is authorized, the amount is transferred to the Merchant, and a "payment successful" message is sent back through the Payment Gateway, completing the transaction. This diagram clearly shows the sequential steps and decision points in a credit card transaction, highlighting the interactions between the three parties involved.

3. Library Management System

3.1 Problem Statement: Managing the operations of a library manually, such as tracking borrowed books, overdue returns, and inventory updates, is time-consuming and prone to errors. A robust system is needed to streamline these operations, improve efficiency, and enhance the experience for library users.

3.2 SRS-Software Requirements Specification



1/10/24

LAB-02

* Generate SRS.

(1) → Library Management.
→ stock maintenance.
→ passport automation.

⇒ Library Management.

(i) Introduction:-

1.1) purpose:- for providing online library services.

1.2) scope:- It includes managing library catalogories allowing users to search borrow, return & renew books. It ensures different access for students, staff & librarian.

1.3) problem statement:- Design a website for providing library services.

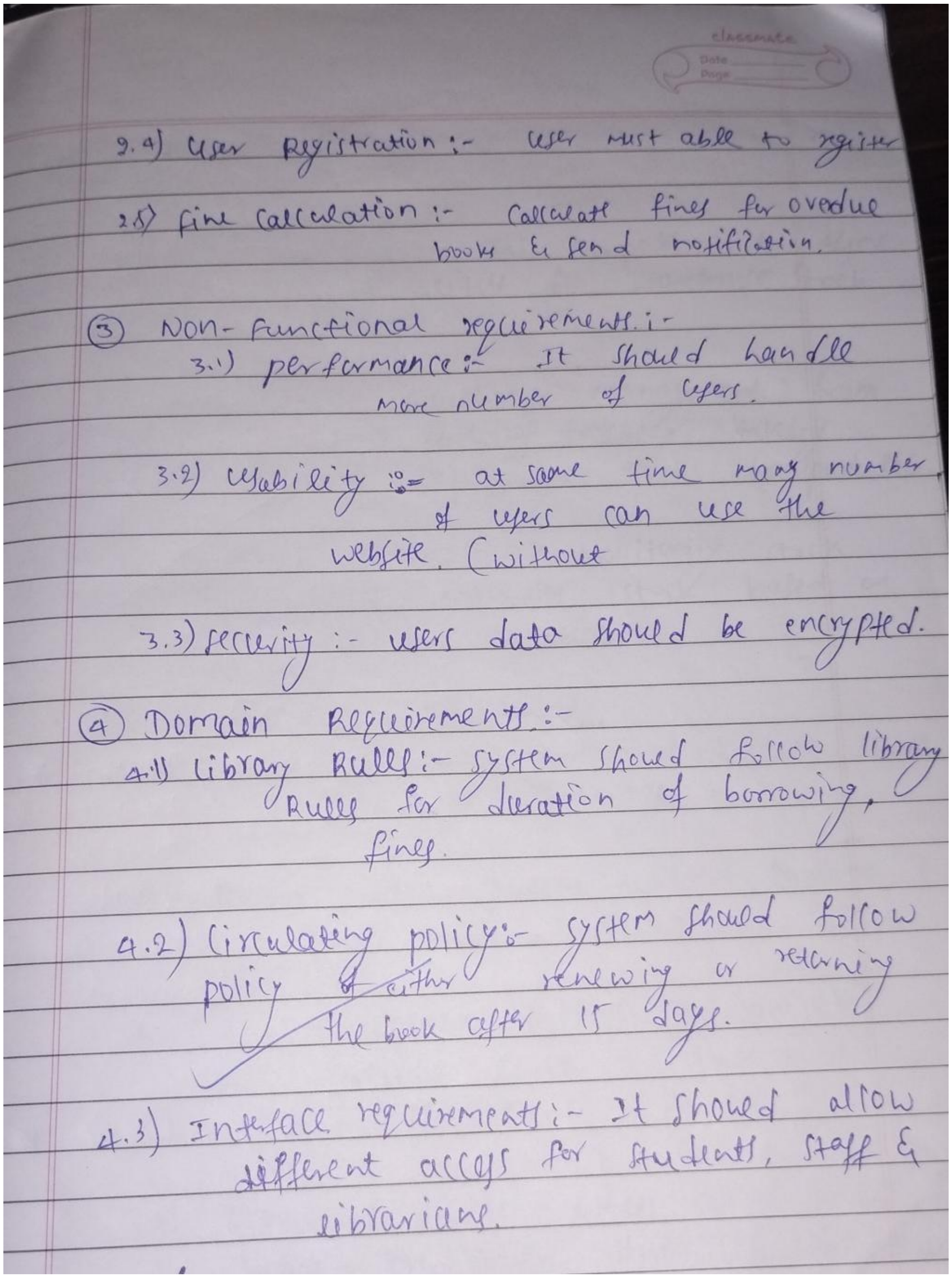
(2) Functional requirements.

2.1) Book search:- users should be able to search for books, by title, author, subject or ISBN.

2.2) Book Reservation:- Users can reserve books that are currently borrowed & receive notification when they become availabl

2.3) Borrow & return:- System should ~~return~~ record when users borrow or return books.

Fig 3.2.1



2.4) User Registration :- user must able to register

2.5) fine calculation :- Calculate fines for overdue books & send notification.

③ Non-functional requirements :-

3.1) performance :- It should handle more number of users.

3.2) usability :- at same time many number of users can use the website. (without

3.3) security :- users data should be encrypted.

④ Domain Requirements :-

4.1) library Rules :- system should follow library Rules for duration of borrowing, fines.

4.2) Circulating policy :- system should follow policy of either renewing or returning the book after 15 days.

4.3) Interface requirements :- It should allow different access for students, staff & librarians.

Fig 3.2.2

3.3 Class Diagram

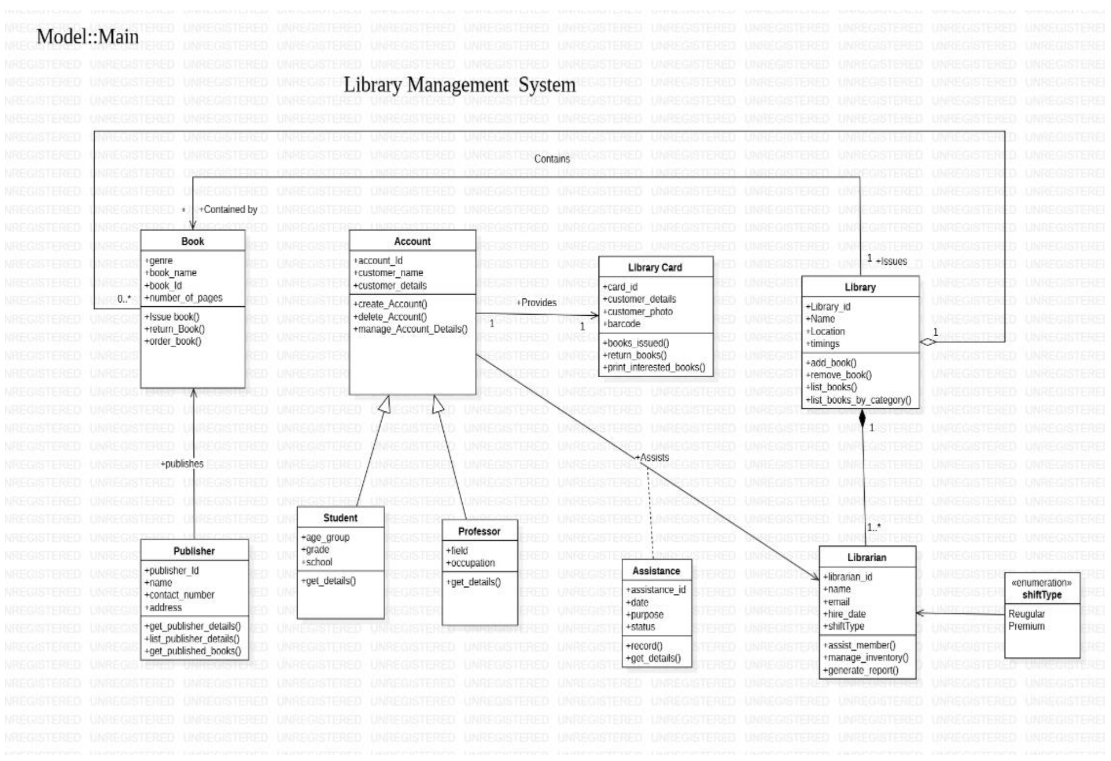


Fig 3.3.1

3.4 State Diagram

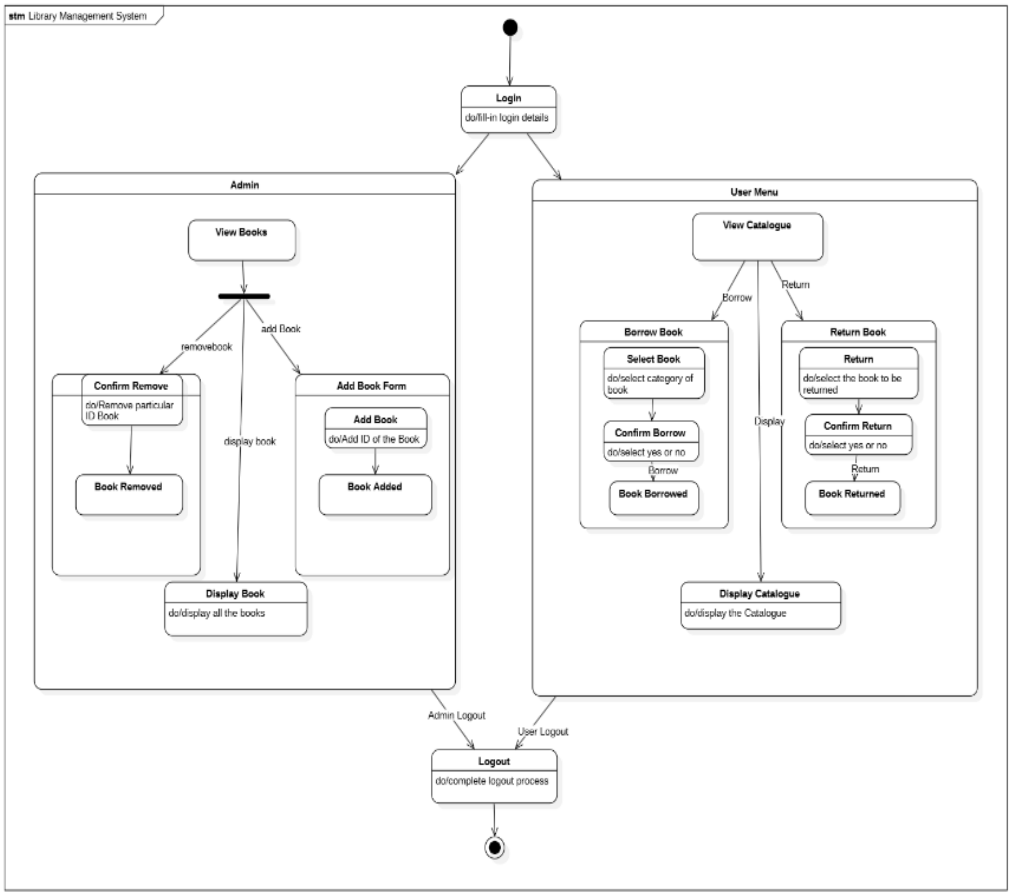


Fig 3.4.1

3.5 Use Case Diagram

LMS_UseCase Diagram::LMS

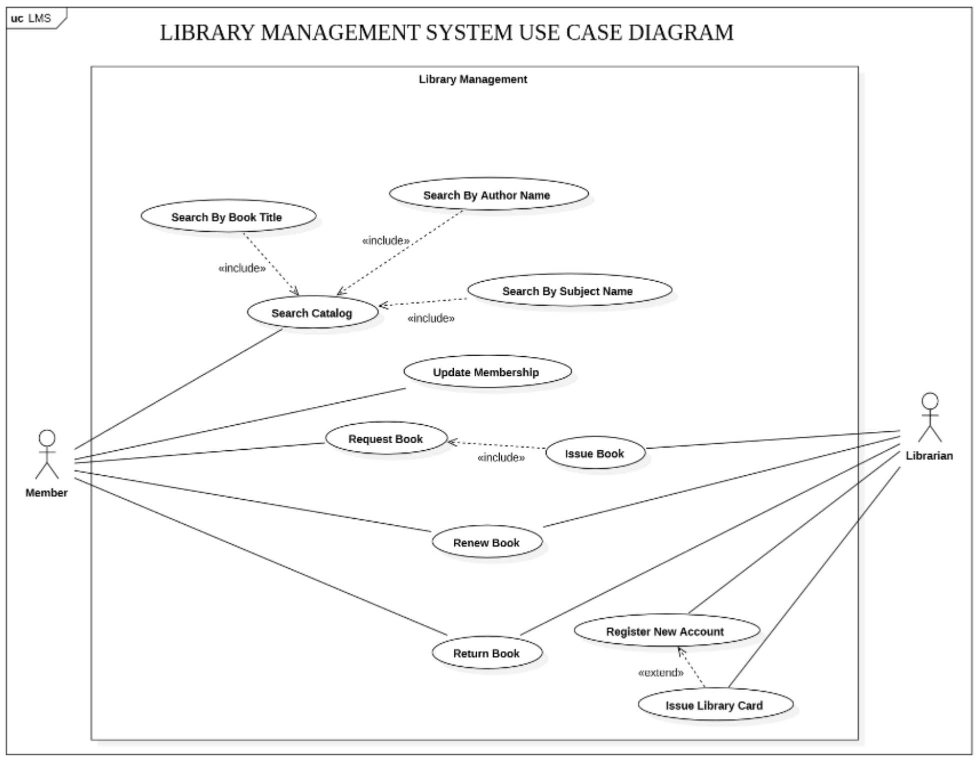


Fig 3.5.1

3.6 Sequence Diagram

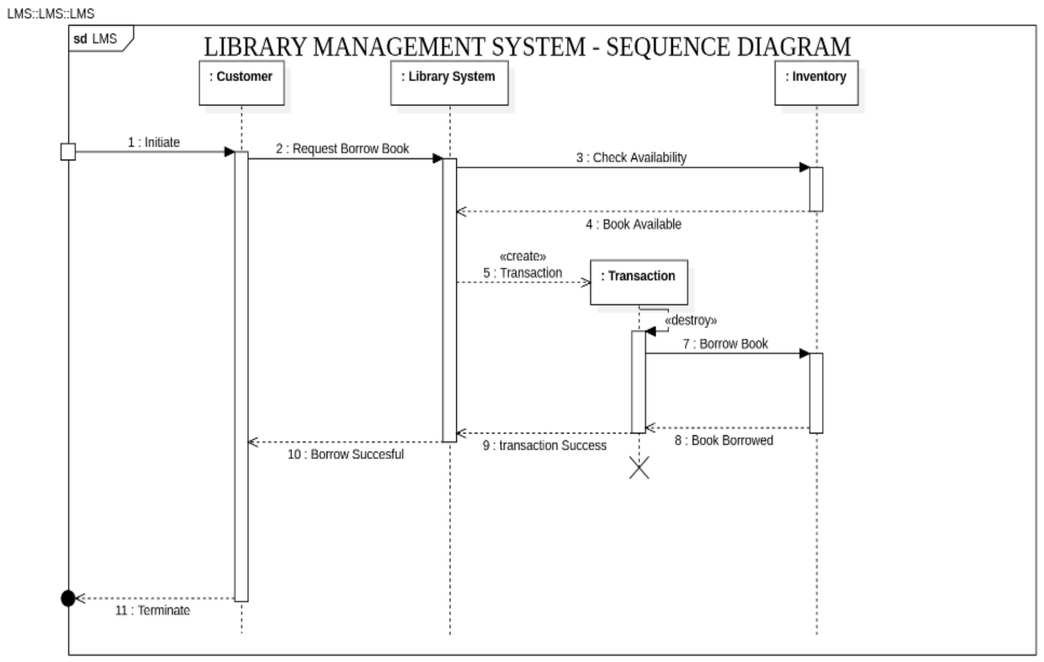


Fig 3.6.1

3.7 Activity Diagram

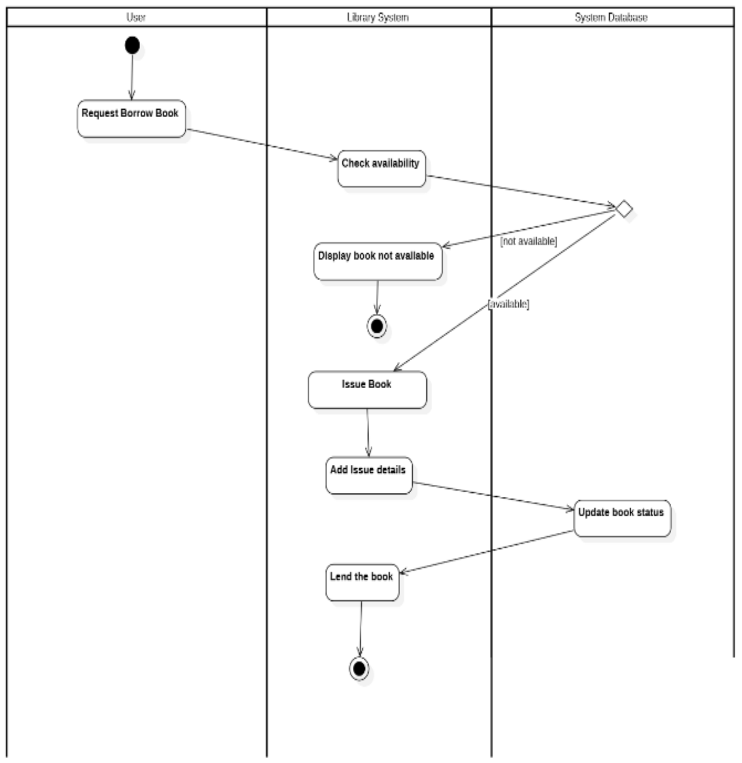
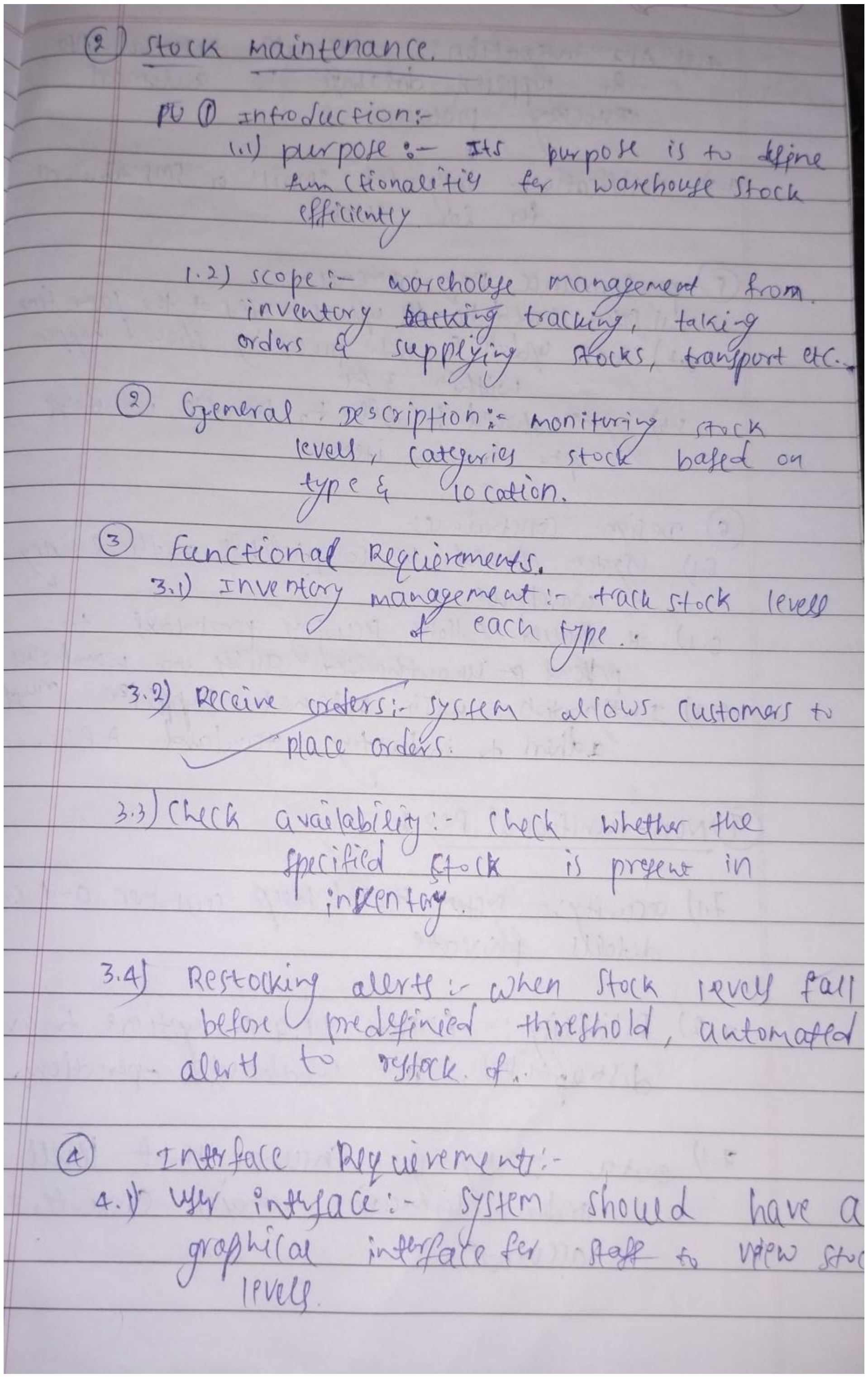

User
Library System
System Database
Request Borrow Book
Check availability
[not available]
Display book not available
[available]
Issue Book
Add Issue details
Update book status
Lend the book

Fig 3.7.1

4. Stock Management System

4.1 Problem Statement: Managing stock levels manually in a business is inefficient, prone to human error, and can lead to overstocking or understocking issues. A well-organized system is required to monitor inventory, track stock movements, and ensure the availability of items to meet customer demands efficiently.

4.2 SRS-Software Requirements Specification



② Stock maintenance.

~~PU~~ ① Introduction:-

1.1) purpose :- Its purpose is to define functionality for warehouse stock efficiently

1.2) scope :- warehouse management from inventory ~~tacking~~ tracking, taking orders & supplying stocks, transport etc.

② General Description:- monitoring stock levels, categories stock based on type & location.

③ Functional Requirements.

3.1) Inventory management :- track stock levels of each type.

3.2) Receive orders:- system allows customers to place orders.

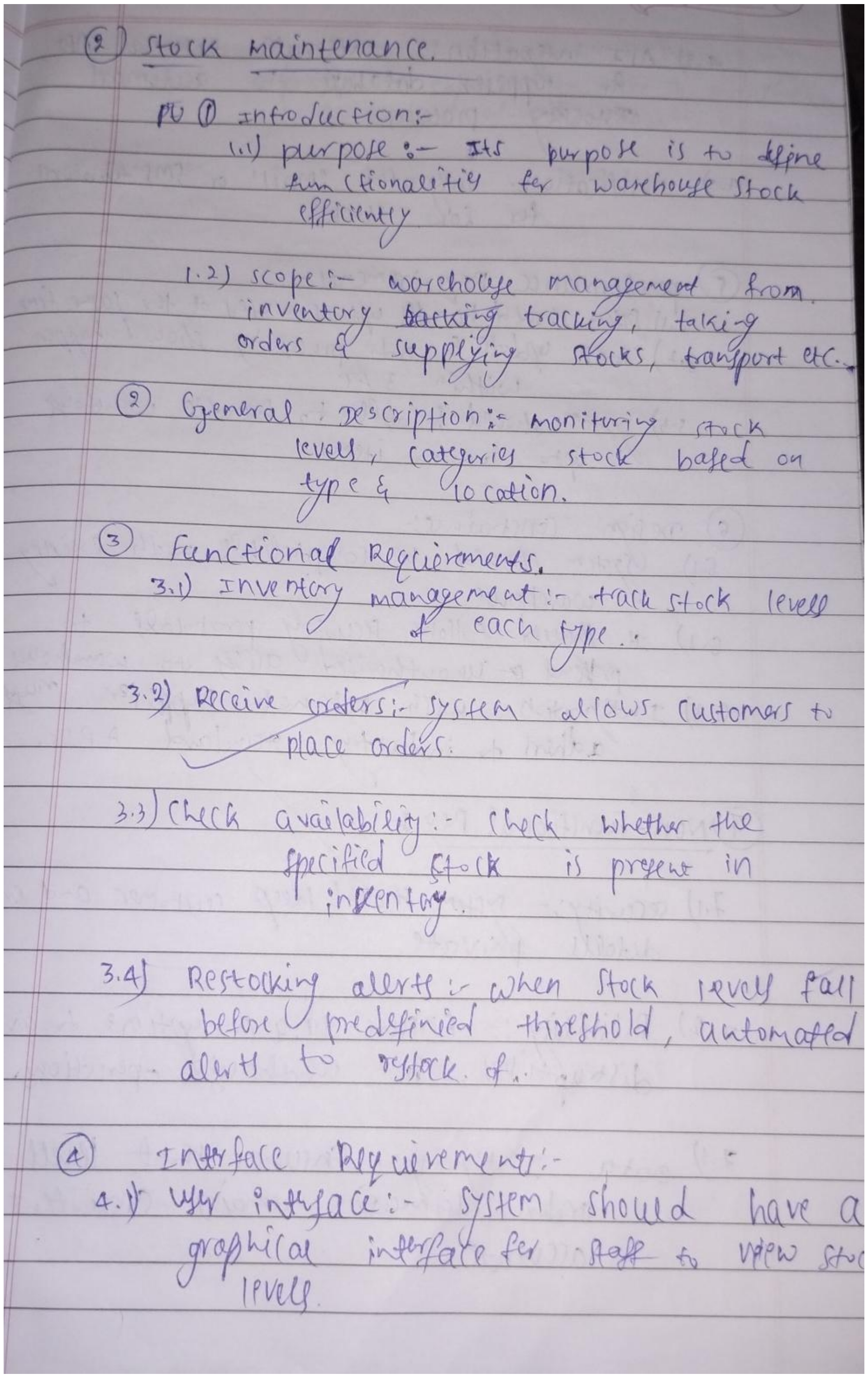
3.3) Check availability:- check whether the specified stock is present in inventory.

3.4) Restocking alerts:- when stock levels fall before predefinied threshold, automated alerts to restock of.

④ Interface Requirements:-

4.1) User interface:- System should have a graphical interface for ~~staff~~ to view stoc levels.

Fig 4.2.1



② Stock maintenance.

PU ① Introduction:-

1.1) purpose :- Its purpose is to define functionality for warehouse stock efficiently

1.2) scope :- warehouse management from inventory ~~backing~~ tracking, taking orders & supplying stocks, transport etc.

② General Description :- monitoring stock levels, categories stock based on type & location.

③ Functional Requirements.

3.1) Inventory management :- track stock levels of each type.

3.2) Receive orders :- system allows customers to place orders.

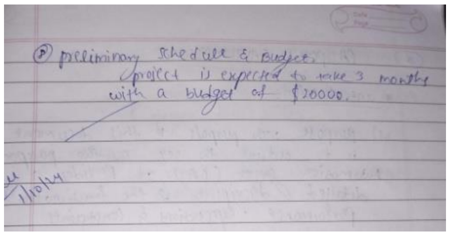
3.3) Check availability :- check whether the specified stock is present in inventory.

3.4) Restocking alerts :- when stock levels fall before predefinied threshold, automated alerts to restock of.

④ Interface Requirements:-

4.1) User interface :- System should have a graphical interface for ~~staff~~ to view stoc levels.

Fig 4.2.2



(8) preliminary schedule & Budget.
project is expected to take 3 months with a budget of $20000.
1/10/24

Fig 4.2.3

4.2 Class Diagram

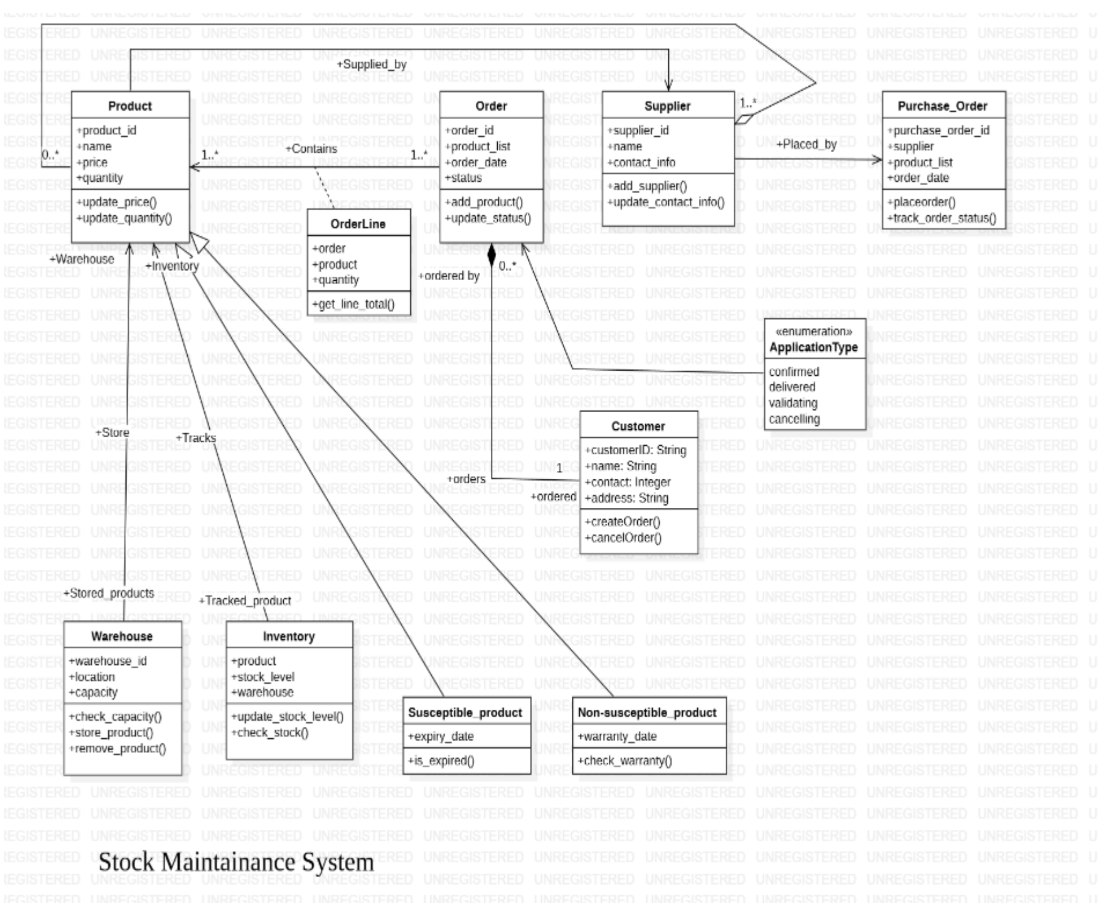


Fig 4.3.1

4.3 State Diagram

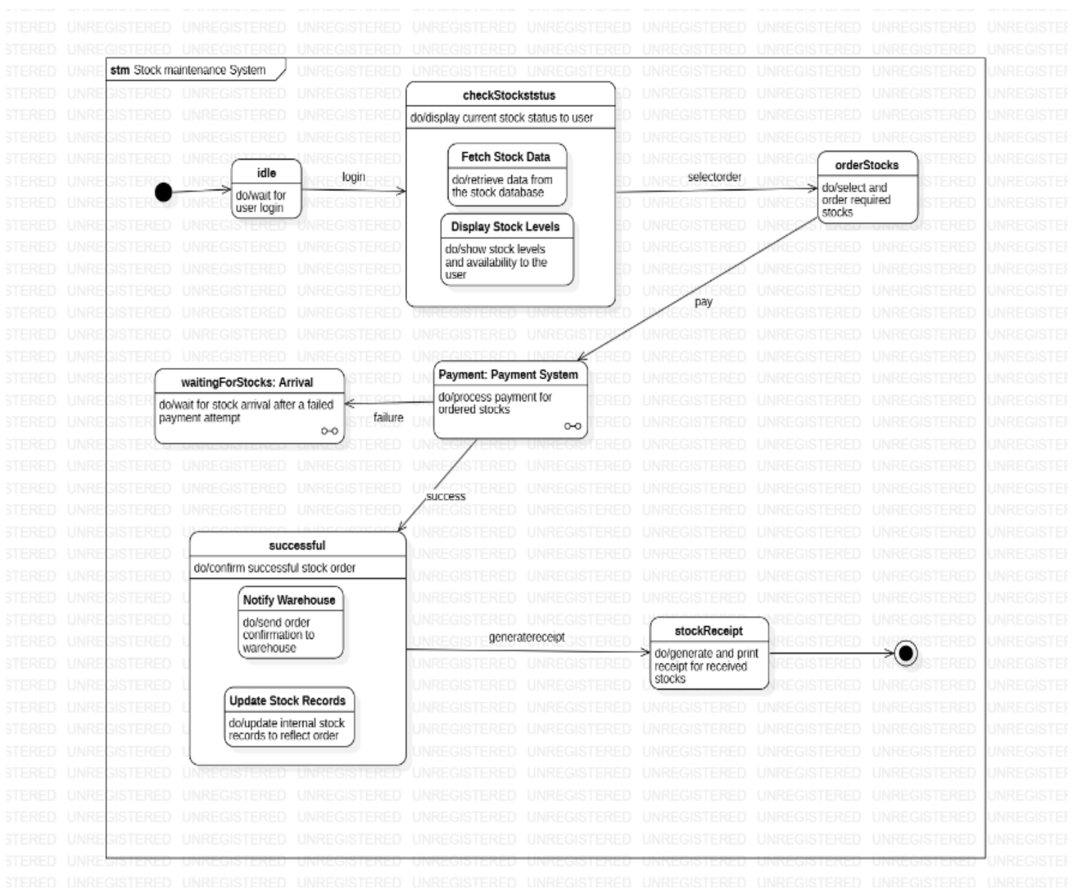

stm Stock maintenance System
idle
do/wait for user login
login
checkStockststus
do/display current stock status to user
Fetch Stock Data
do/retrieve data from the stock database
Display Stock Levels
do/show stock levels and availability to the user
selectorder
orderStocks
do/select and order required stocks
pay
Payment: Payment System
do/process payment for ordered stocks
failure
waitingForStocks: Arrival
do/wait for stock arrival after a failed payment attempt
success
successful
do/confirm successful stock order
Notify Warehouse
do/send order confirmation to warehouse
Update Stock Records
do/update internal stock records to reflect order
generatereceipt
stockReceipt
do/generate and print receipt for received stocks

Fig 4.4.1

4.4 Use Case Diagram

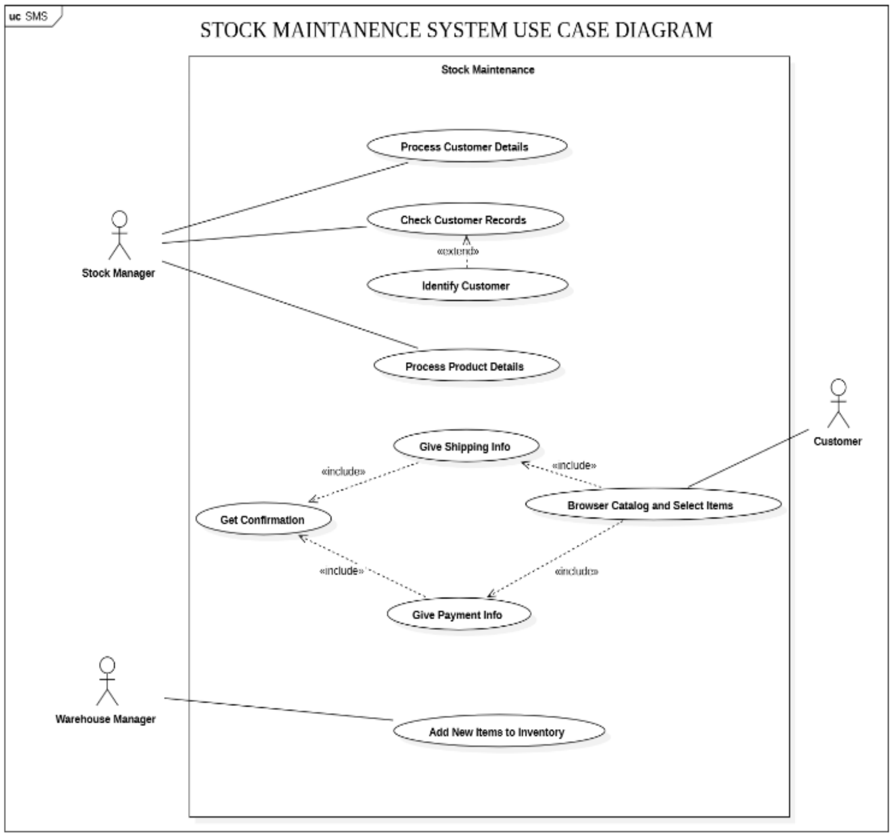


Fig 4.5.1

4.6 Sequence Diagram

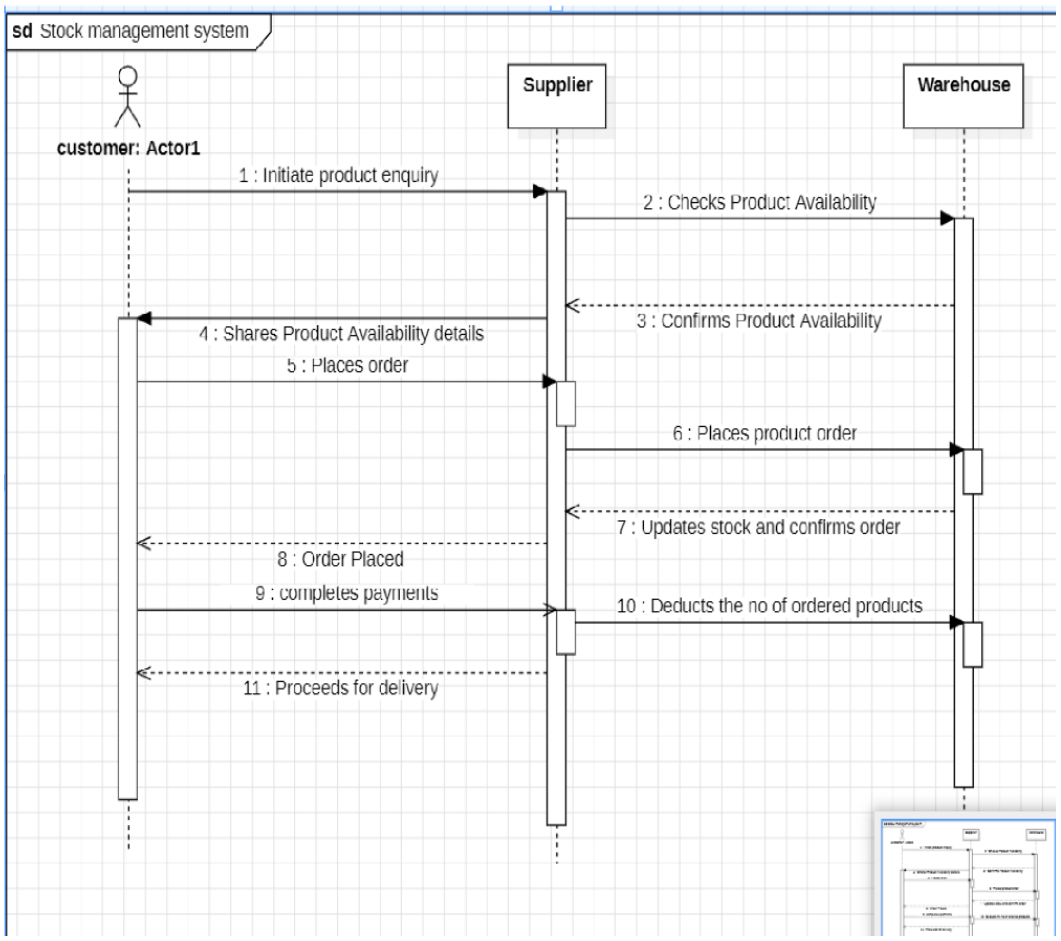

sd Stock management system
customer: Actor1
Supplier
Warehouse
1 : Initiate product enquiry
2 : Checks Product Availability
3 : Confirms Product Availability
4 : Shares Product Availability details
5 : Places order
6 : Places product order
7 : Updates stock and confirms order
8 : Order Placed
9 : completes payments
10 : Deducts the no of ordered products
11 : Proceeds for delivery

Fig 4.6.1

4.7 Activity Diagram

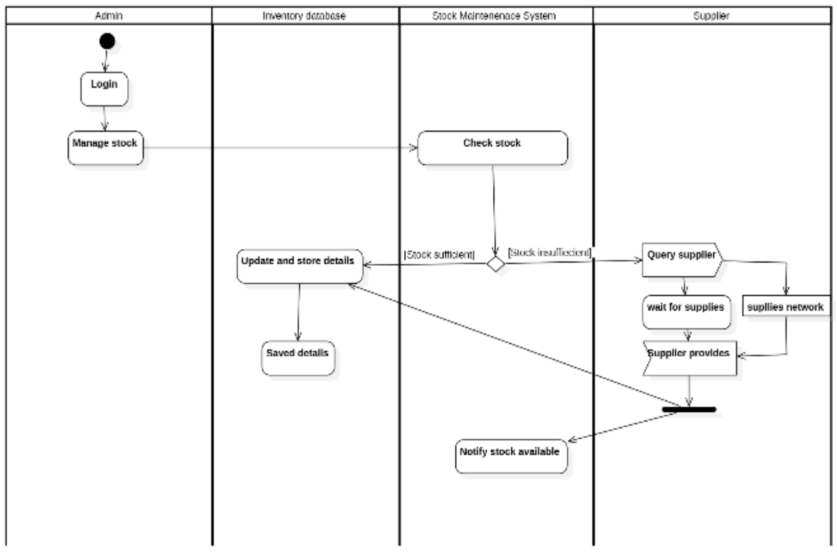
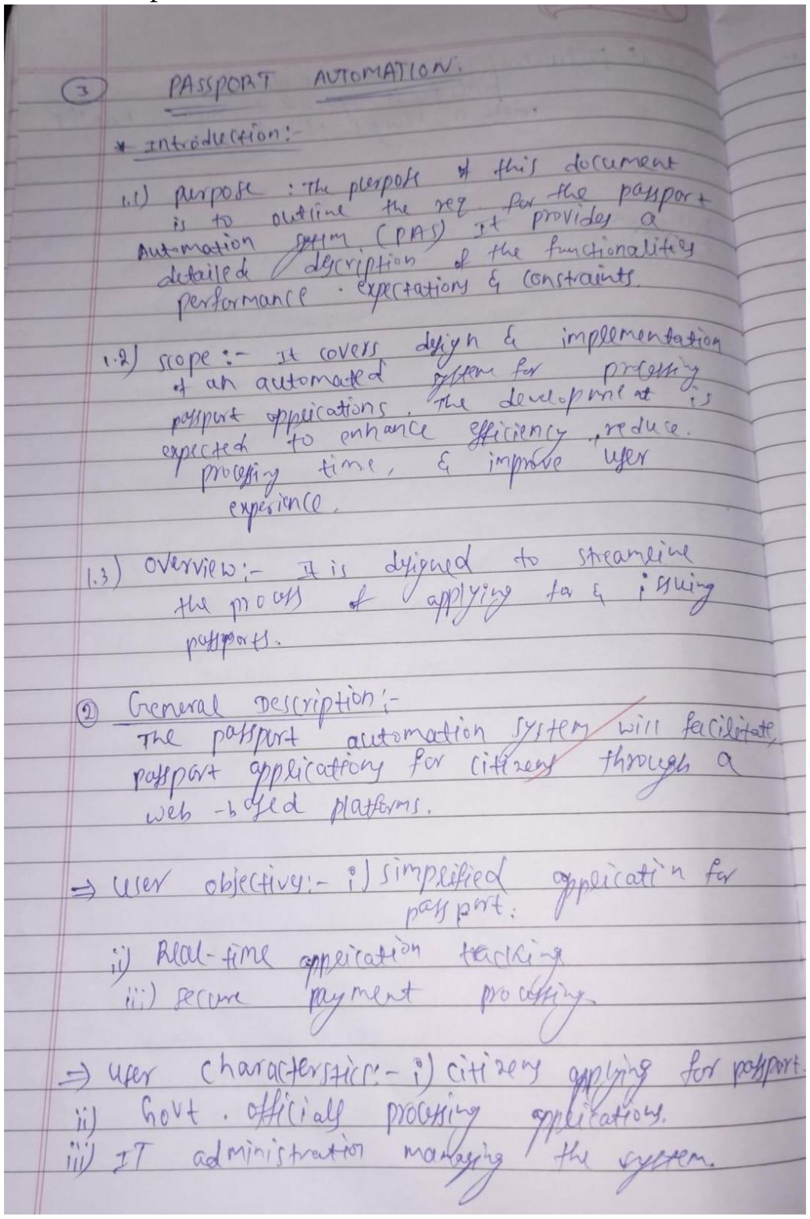


Fig 4.7.1

5 Passport Management System

5.2 Problem Statement: Managing passport applications and issuance is a complex process involving multiple stages, including document verification, application tracking, and appointment scheduling. A streamlined system is required to handle these tasks efficiently, minimize errors, and enhance user experience.

5.3 SRS-Software Requirements Specification



③ PASSPORT AUTOMATION.

* Introduction:-

1.1) purpose : The purpose of this document is to outline the req. for the passport Automation system (PAS) It provides a detailed description of the functionalities performance . expectations & constraints.

1.2) scope :- It covers design & implementation of an automated system for processing passport applications. The development is expected to enhance efficiency, reduce. processing time, & improve user experience.

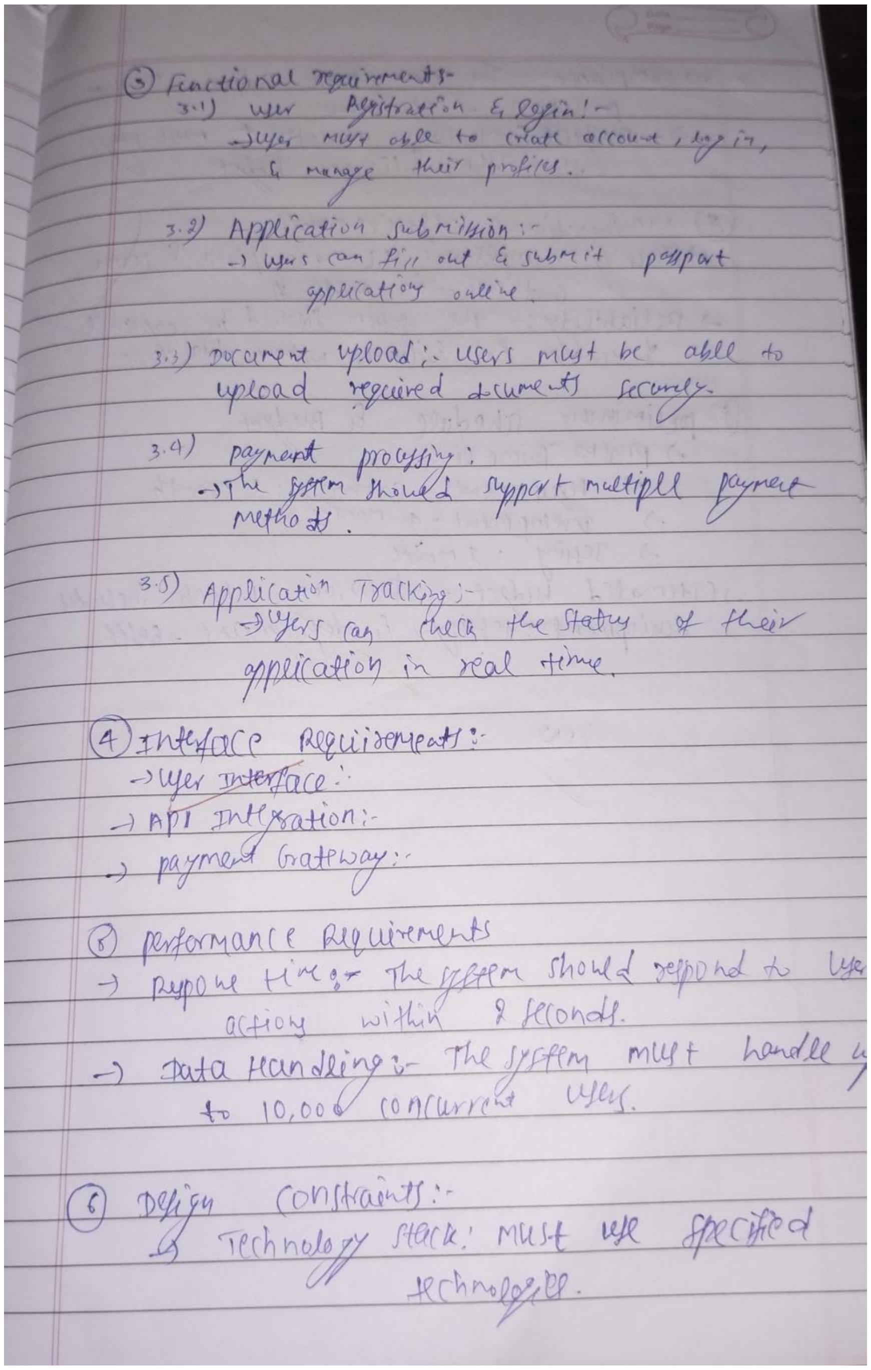
1.3) Overview:- It is designed to streamline the process of applying for & issuing passports.

② General Description:-
The passport automation system will facilitate, passport applications for citizens through a web-based platforms.

⇒ user objectives:- i) simplified application for passport:
ii) Real-time application tracking
iii) secure payment processing.

⇒ user characteristics:- i) citizens applying for passport.
ii) Govt. officials processing applications.
iii) IT administration managing the system.

Fig 5.2.1



③ Functional requirements:-

3.1) user Registration & login:-

→ user must able to create account, log in, & manage their profiles.

3.2) Application submission:-

→ users can fill out & submit passport application online.

3.3) Document upload:- users must be able to upload required documents securely.

3.4) payment processing:

→ The system should support multiple payment methods.

3.5) Application Tracking:-

→ users can check the status of their application in real time.

④ Interface Requirements:-

→ user Interface:

→ API Integration:-

→ payment Gateway:-

⑤ performance Requirements

→ Respone time:- The system should respond to use

action within 2 seconds.

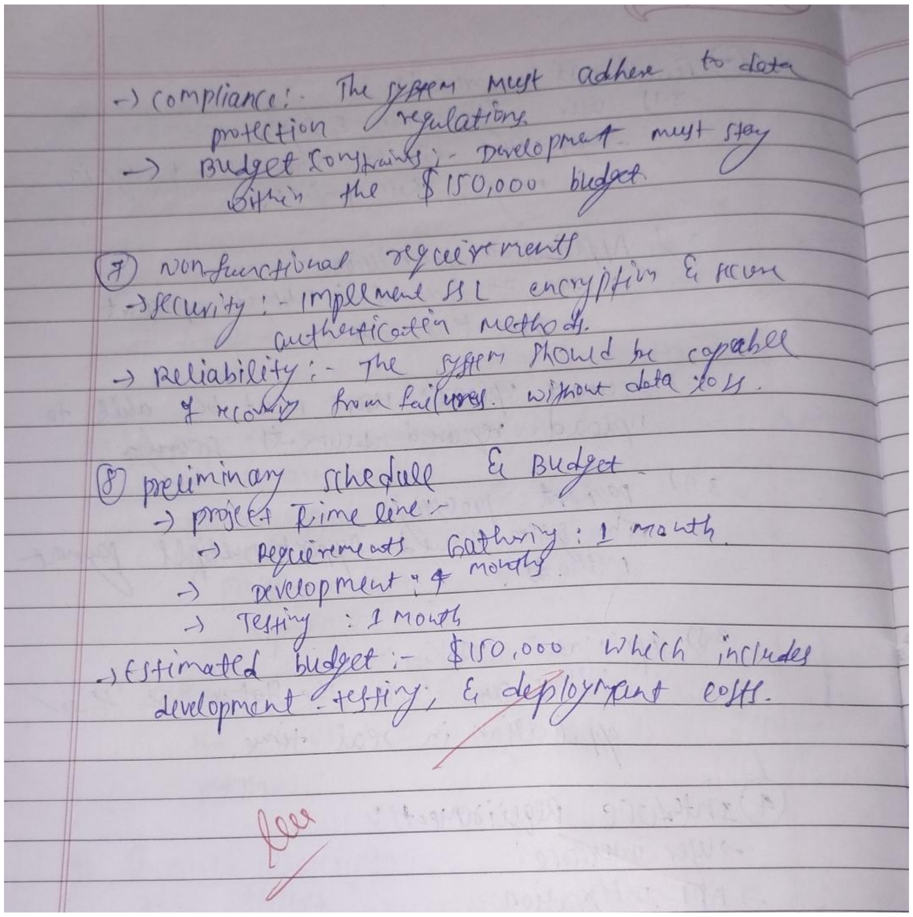
→ Data Handling:- The system must handle u

to 10,000 concurrent users.

⑥ Design constraints:-

→ Technology stack: must use specified technologies.

Fig 5.2.2



-) compliance: The system must adhere to data protection regulations.

-) Budget constraints :- Development must stay within the $150,000 budget.

(7) Non functional requirements

-) security :- implement SSL encryption & secure authentication methods.

-) Reliability :- The system should be capable of recovery from failures without data loss.

(8) preliminary schedule & Budget

-) project Time line :-

-) Requirements Gathering : 1 month.

-) Development : 4 months.

-) Testing : 1 month

-) Estimated budget :- $150,000 which includes development, testing, & deployment costs.

[illegible]

Fig 5.2.3

5.4 Class Diagram

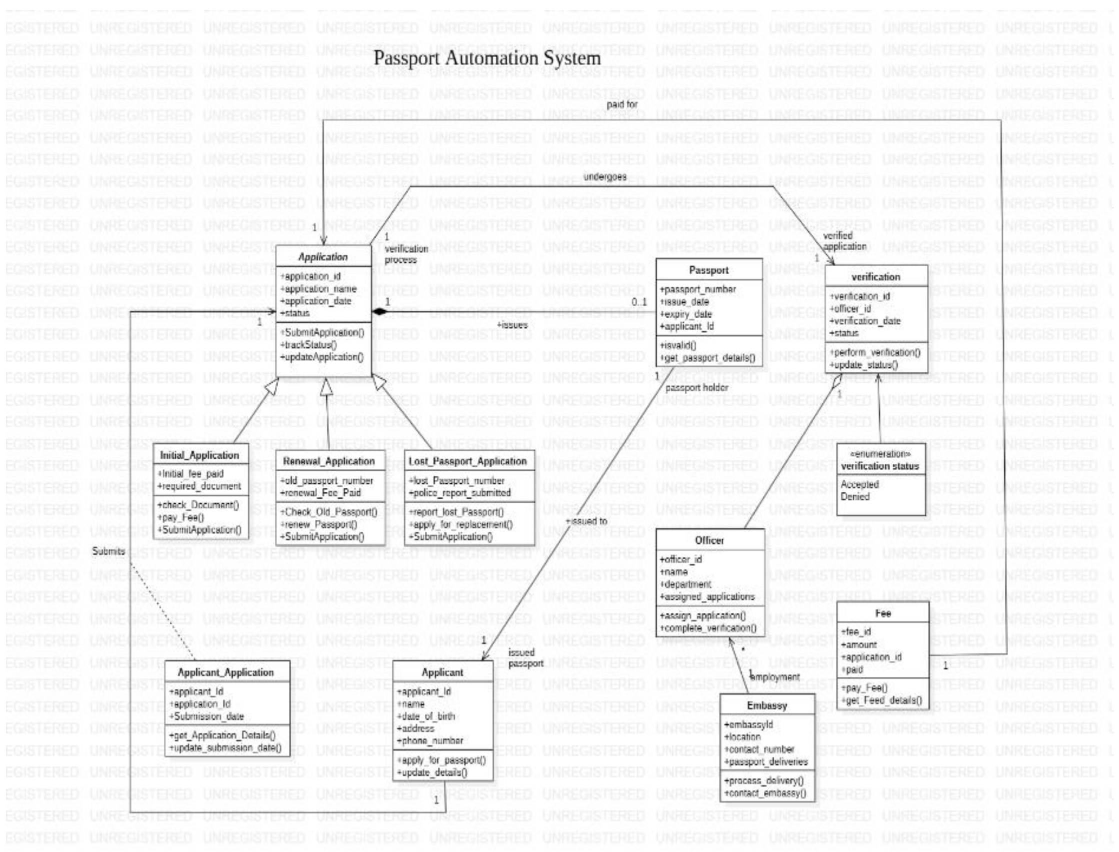


Fig 5.3.1

5.5 State Diagram

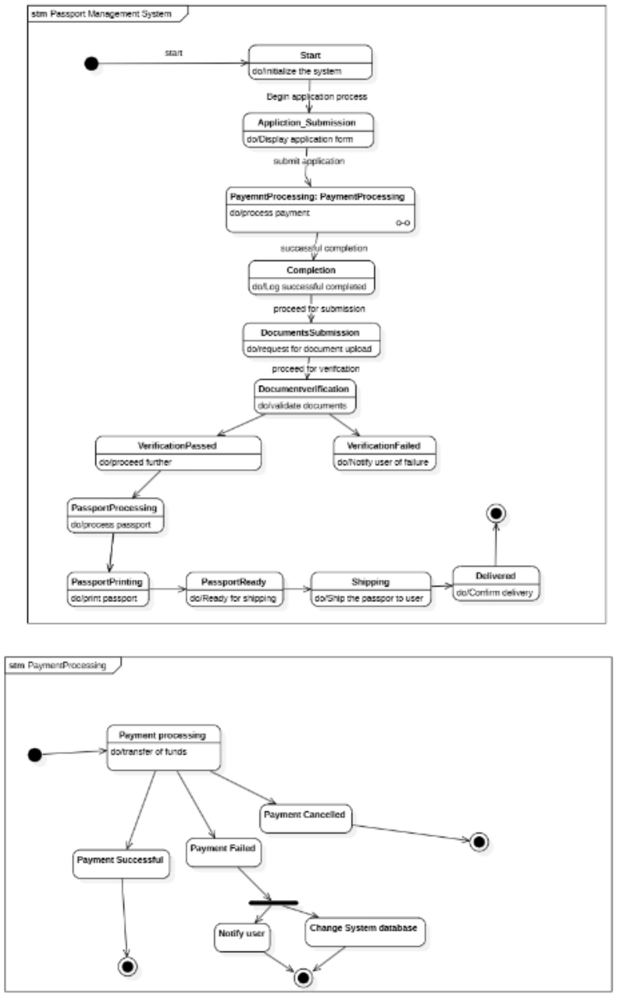


Fig 5.4.1

5.6 Use Case Diagram

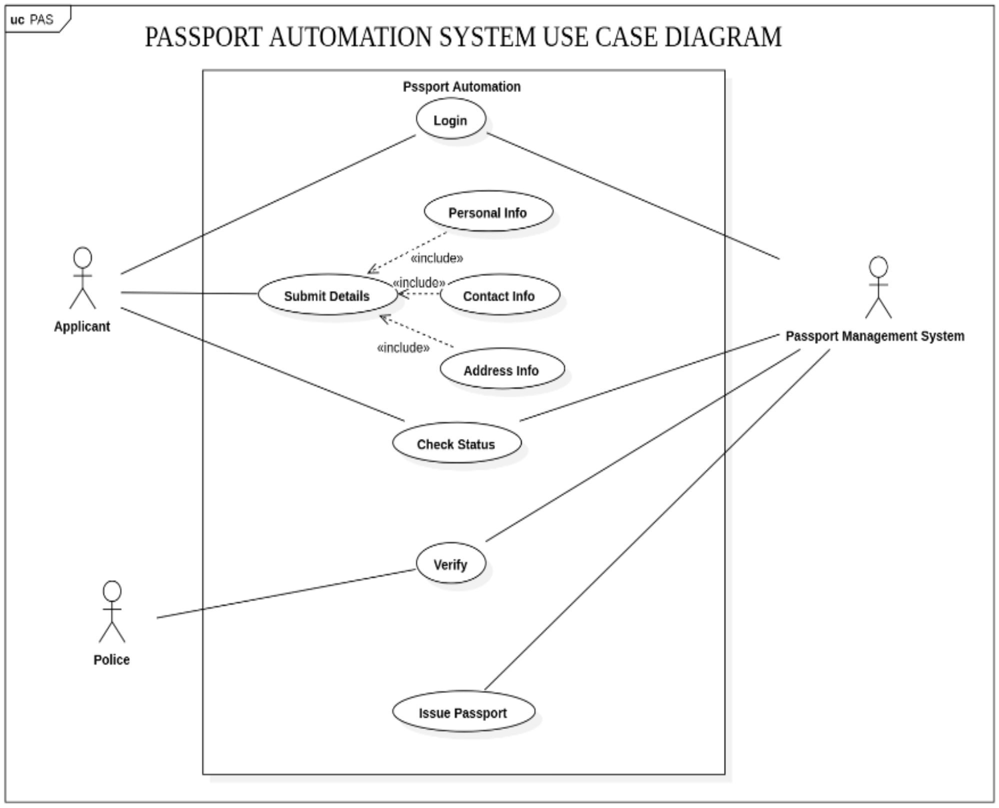


Fig 5.5.1

5.7 Sequence Diagram

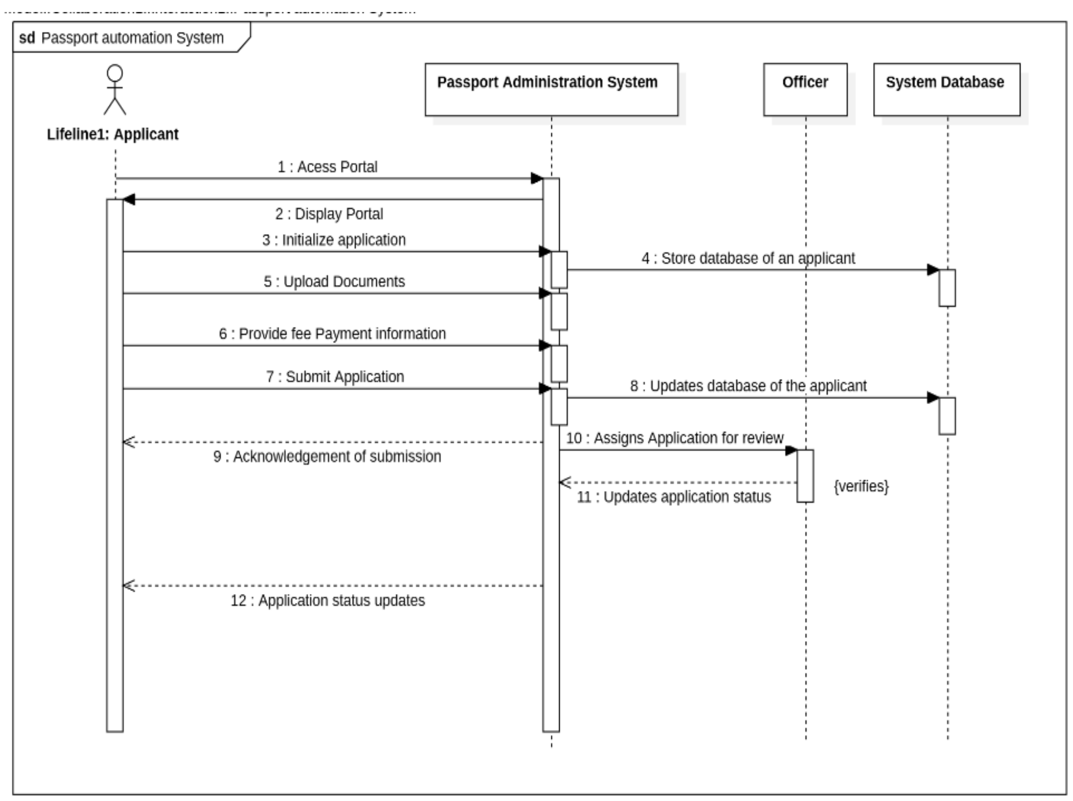

sd Passport automation System
Lifeline1: Applicant
Passport Administration System
Officer
System Database
1 : Acess Portal
2 : Display Portal
3 : Initialize application
4 : Store database of an applicant
5 : Upload Documents
6 : Provide fee Payment information
7 : Submit Application
8 : Updates database of the applicant
9 : Acknowledgement of submission
10 : Assigns Application for review
11 : Updates application status
{verifies}
12 : Application status updates

Fig 5.6.1

5.8 Activity Diagram

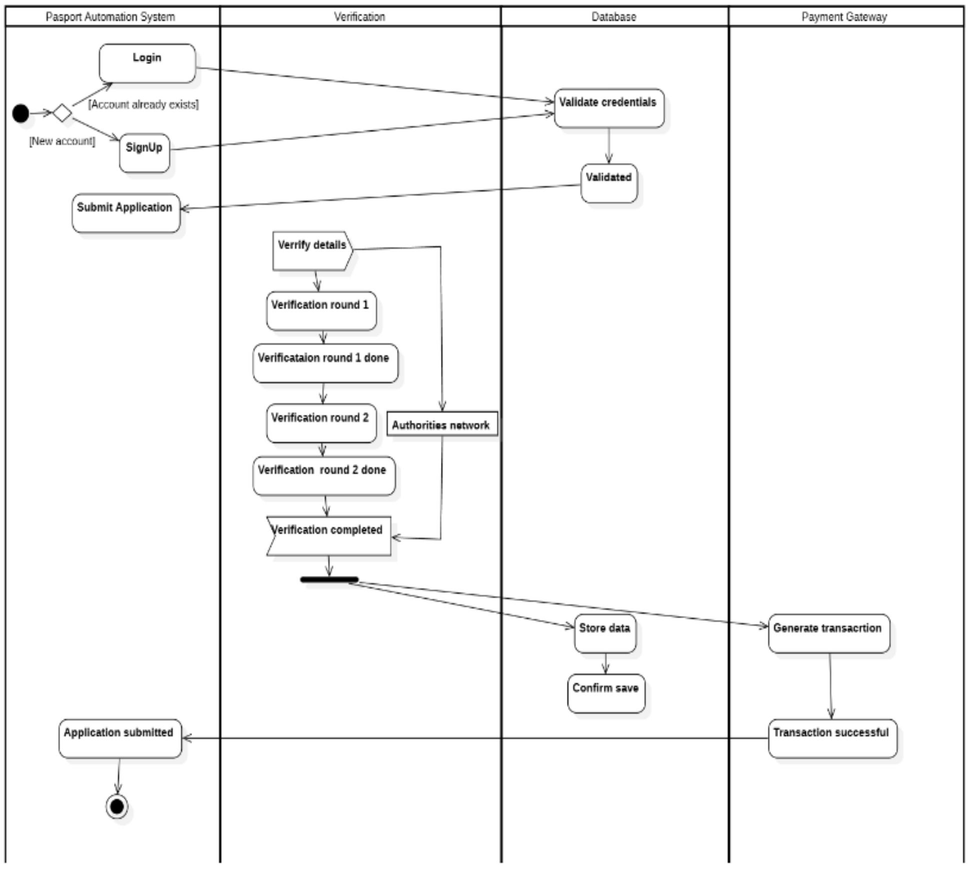

Pasport Automation System
Verification
Database
Payment Gateway
Login
[Account already exists]
[New account]
SignUp
Validate credentials
Validated
Submit Application
Verrify details
Verification round 1
Verificataion round 1 done
Verification round 2
Authorities network
Verification round 2 done
Verification completed
Store data
Generate transacrtion
Confirm save
Application submitted
Transaction successful

Fig 5.7.1