B.M.S. COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

Lab Record

Object Oriented Analysis and Design

Submitted in partial fulfillment for the 6th Semester Laboratory

Bachelor of Technology

in

Computer Science and Engineering

Submitted by:

VEDA C R

1BM20CS224

Department of Computer Science and Engineering

B.M.S. College of Engineering

Bull Temple Road, Basavanagudi, Bangalore 560 019

March-July 2023

B.M.S. COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Object-Oriented Analysis and Design(20CS6PCOMD) laboratory has been carried out by VEDA C R(1BM20CS224) during the 6th Semester March-July-2023.

Signature

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

TABLE OF CONTENTS

1. HOTEL MANAGEMENT SYSTEM

1.1 Problem statement

With the world now moving towards digitalization of every possible task, it is inevitable that libraries stop relying on physical staff for operation. Library management should allow users to borrow books, return books, pay fines and issue cards.

1.2 SRS Document

Introduction

Purpose

Operate a library efficiently using automation, streamlining tasks involved in running a library such as cataloging, issues, managing membership details, placing fines for delayed returns etc

Scope of the document:

To specify the functional, behavioral and non-functional requirements of the software application that automates and streamlines various tasks involved in running a library.

Overview

A library management system is a software application that helps librarians to manage the library operations, such as acquiring, cataloging, circulating, and reporting on library materials. It also helps users to find and access library resources easily and efficiently.

General description:

The Hotel Management software system provides hotels with a user-friendly interface to manage reservations, guest information, inventory, billing and payments, reporting and analytics, staff management, and loyalty programs. Additionally, it establishes a centralized database for efficient data storage and retrieval.

Functional requirements:

Room reservation/booking: The system should be able to record reservations, customer details, room number, room rate, confirmation number, check-in and check-out dates and times, etc. The system should also be able to modify or cancel reservations, mark assured rooms as “must pay”, and charge extra fees for late check-outs12.

Food service: The system should be able to track all meals purchased in the hotel (restaurant and room service), record payment and payment type for meals, and bill the current room if payment is not provided at the time of service1.

Management: The system should be able to generate reports on occupancy, revenue, expenses, customer feedback, etc. The system should also be able to manage inventory, staff, security, maintenance, etc12.

Customer service: The system should be able to provide online booking, check-in and check-out options for customers, send email notifications and reminders, offer loyalty programs and discounts, etc. The system should also be able to handle customer complaints and requests34.

Interface requirements

- User Interface
    - UI should be simple, intuitive, responsive and consistent
    - Interface should support different languages and devices as needed

        - UI should display availability and location of books in the library
    - Hardware Interface:
        - The system should be compatible with multiple operating systems such as Windows, Linux etc
        - The system should be reliable and secure when communicating with hardware devices
        - The system should use QR codes to scan books and cards, and to facilitate payments

Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes

Performance Requirements

- The number of pages should be minimized for the user's convenience
- Adding members, borrowing and renewing books should be very quick.
- There should be a procedure for when there is loss of data due to failure of storage device or similar reason.

Design Constraints

- The information of all users, staff must be stored in a database that is accessible by the website.
- The Hotel Management System is running 24 hours a day.
- Users may access from any computer that has Internet browsing capabilities and an Internet connection.
- Users must have their correct usernames and passwords to enter into their online accounts and do actions.

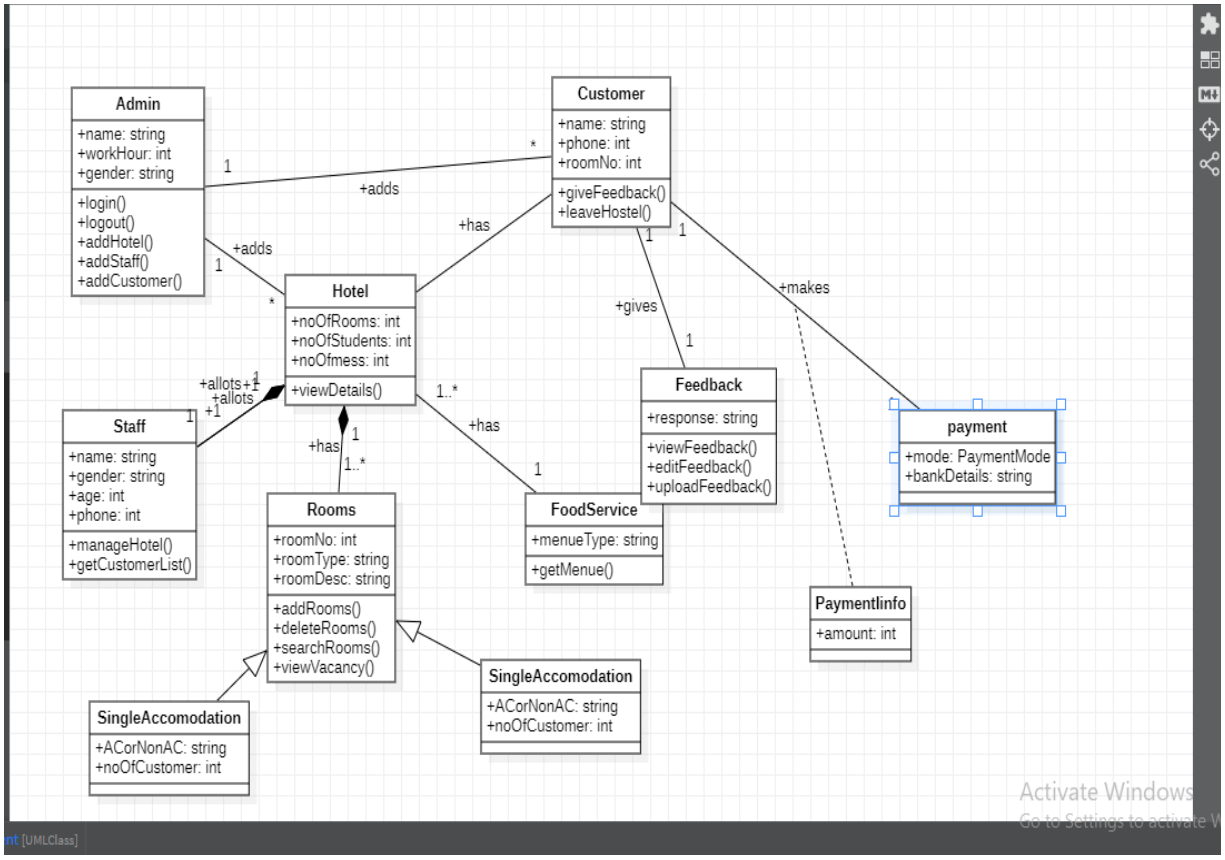
Non-functional Attributes

- Usability: The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.
- Accuracy: The data stored about the books and the fines calculated should be correct, consistent, and reliable.
- Availability: The System should be available for the duration when the library

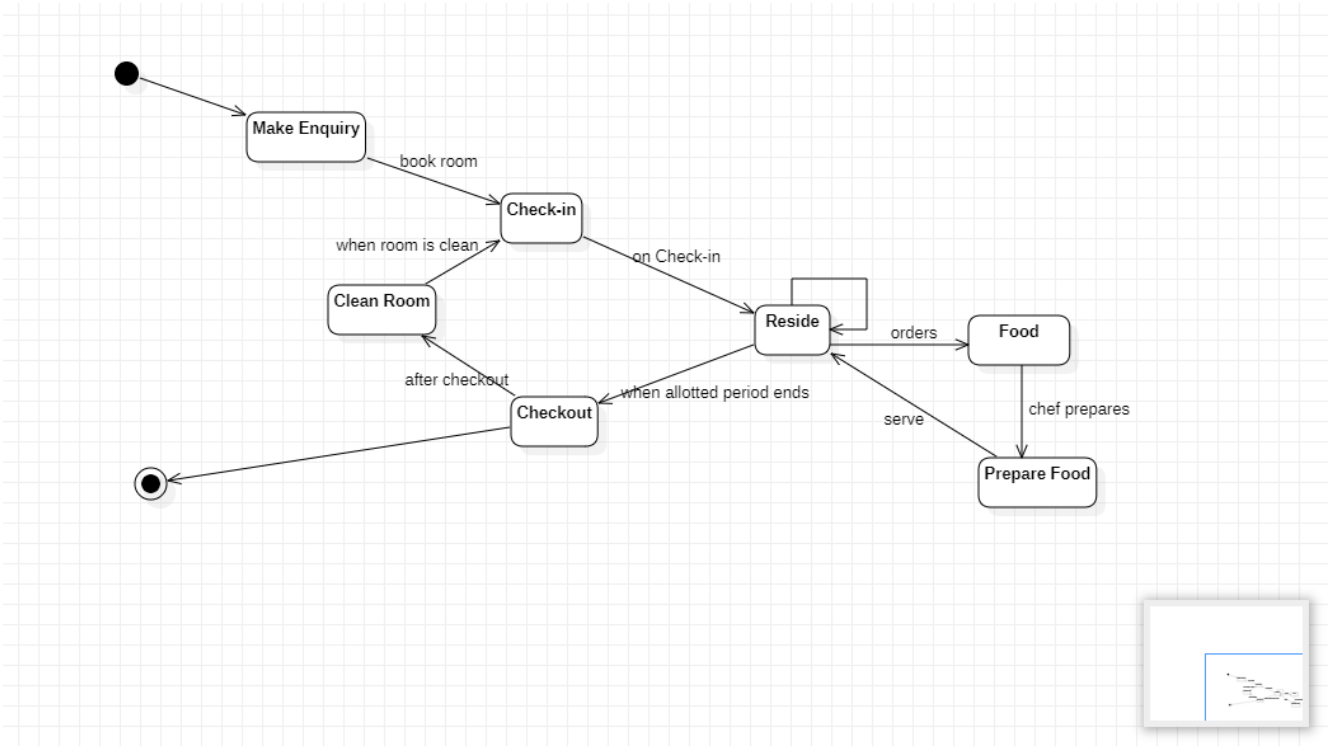
operates and must be recovered within an hour or less if it fails. The system should respond to the requests within two seconds or less.

- Performance: The system should be fast and accurate. It should handle expected and unexpected errors. It should be able to handle large amount of data.
- Security: The system should protect the data from unauthorized access and modification. It should use encryption and authentication techniques to ensure data security.
- Scalability: The system should be able to handle increasing number of users and books without degrading the performance or functionality.

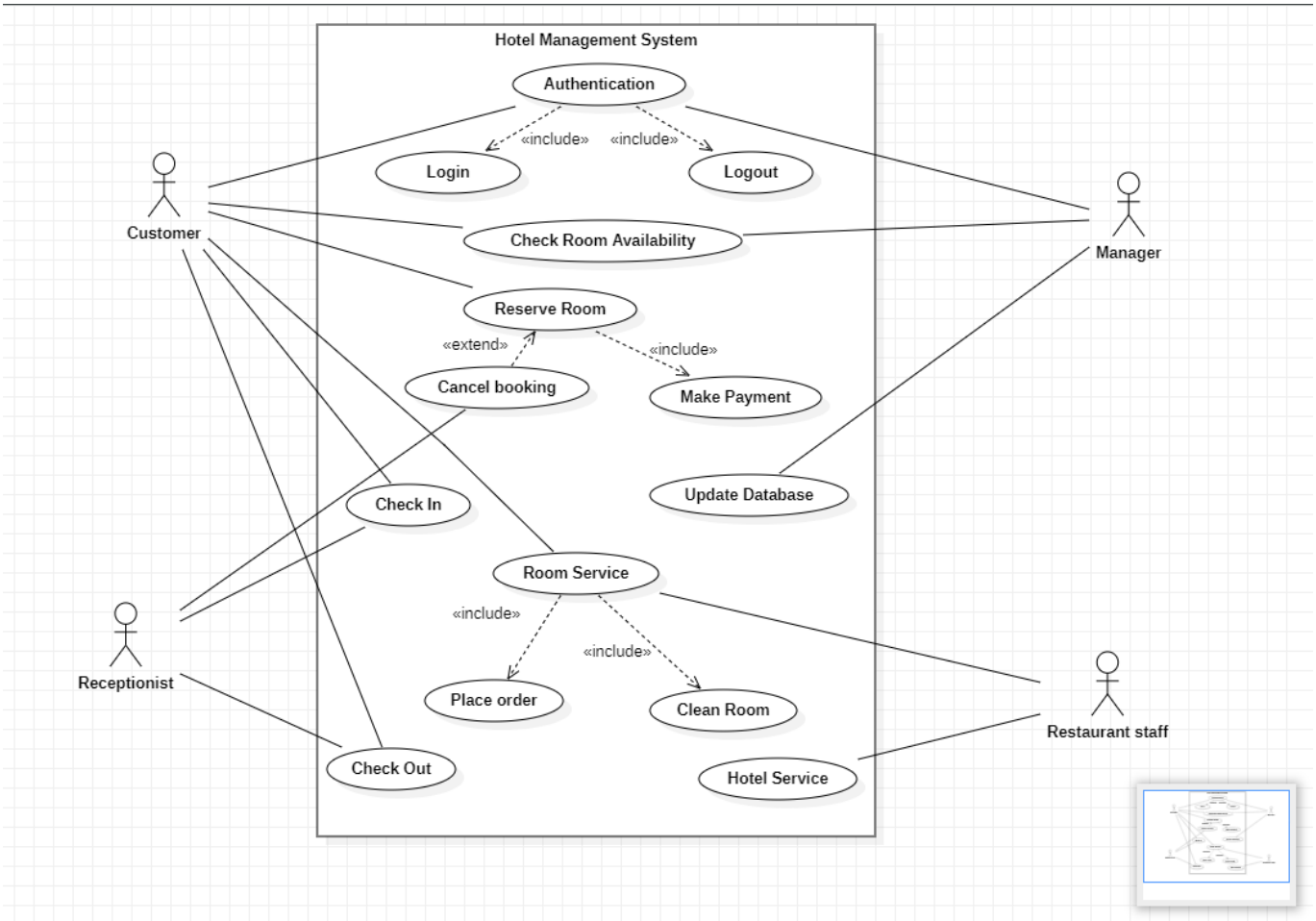
1.2 Class Diagram



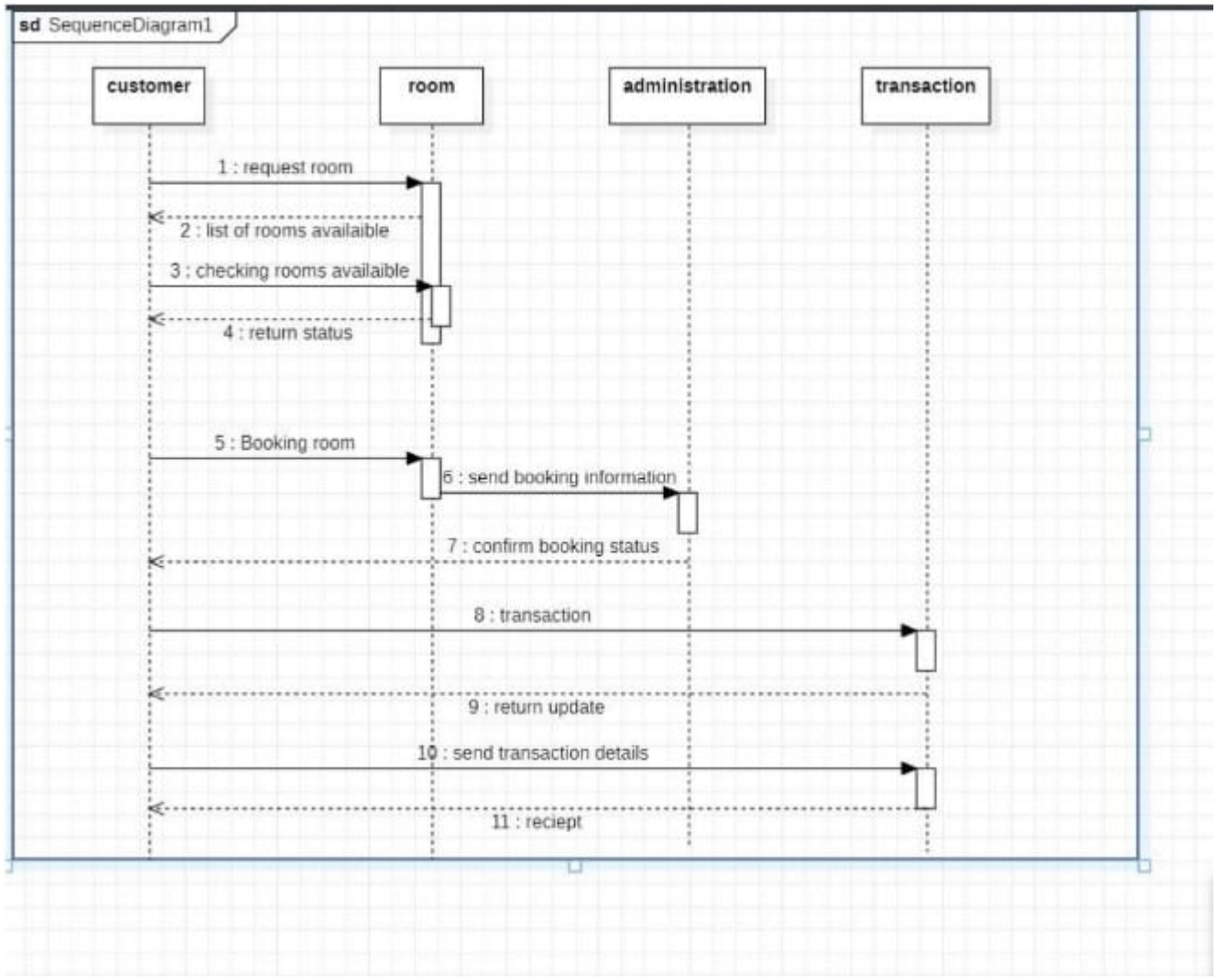
1.3 State Diagram



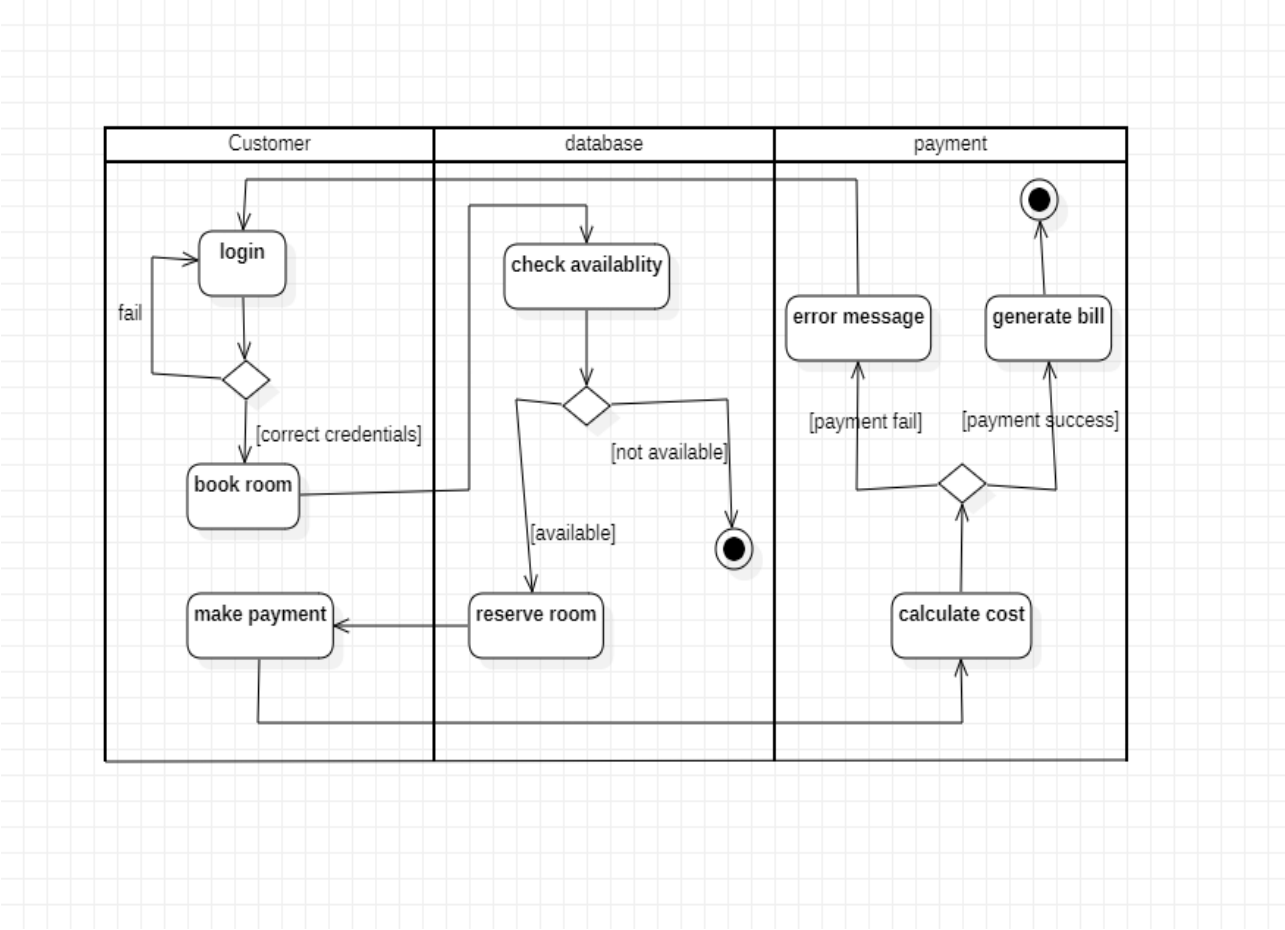
1.4 Use Case Diagram



1.5 Sequence Diagram



1.6 Activity Diagram



2. CREDIT CARD SYSTEM

2.1 Problem statement

To design and develop a software system that can perform online credit card processing securely and efficiently, by validating the card details, authorizing the transactions, sending receipts to customers and transferring funds to the merchants.

2.2 Software Requirement Specification

Introduction

Purpose

Operate a credit card machine efficiently using automation, streamlining tasks involved in transactions.

Scope of the document:

To specify the functional, behavioral and non-functional requirements of the software application that automates and streamlines various tasks involved in running a library.

Overview

A credit card processing system can perform online credit card processing securely and efficiently, by validating the card details, authorizing the transactions, sending receipts to customers and transferring funds to the merchants.

Functional requirements:

- Data entry: The system should be able to accept and validate credit card information from customers, such as card number, expiration date, CVV code, billing address, etc. The system should also be able to accept and process payment methods, such as cash, check, debit card, etc12.

- Transaction processing: The system should be able to communicate with the card issuer or bank to authorize or decline transactions, and handle errors or exceptions. The system should also be able to calculate and apply fees, taxes, discounts, rewards, etc. The system should also be able to generate and send receipts and invoices to customers and merchants132.

- Security: The system should be able to encrypt and protect the credit card data from unauthorized access or theft. The system should also be able to detect and prevent fraud, such as identity theft, chargebacks, etc. The system should also be able to comply with the relevant laws and regulations, such as PCI DSS14.

- Reporting: The system should be able to provide reports and analytics on the credit card transactions, such as sales volume, revenue, profit, customer behavior, etc. The system should also be able to provide alerts and notifications on any issues or anomalies in the transactions13.

- Received message. Based on web search results, here is a list of some functional requirements for a credit card system: - Data entry: The system should be able to accept and validate credit card information from customers, such as card number, expiration date, CVV code, billing address, etc. The system should also be able to accept and process payment methods, such as cash, check, debit card, etc . - Transaction processing: The system should be able to communicate with the card issuer or bank to authorize or decline transactions, and handle errors or exceptions. The system should also be able to calculate and apply fees, taxes, discounts,

rewards, etc. The system should also be able to generate and send receipts and invoices to customers and merchants . - Security: The system should be able to encrypt and protect the credit card data from unauthorized access or theft. The system should also be able to detect and prevent fraud, such as identity theft, chargebacks, etc. The system should also be able to comply with the relevant laws and regulations, such as PCI DSS . - Reporting: The system should be able to provide reports and analytics on the credit card transactions, such as sales volume, revenue, profit, customer behavior, etc. The system should also be able to provide alerts and notifications on any issues or anomalies in the transactions .

Interface requirements

- User Interface
- UI should be simple, intuitive, responsive and consistent
- Interface should support different languages and devices as needed
- UI should display availability and location of books in the library

Hardware Interface:

- The system should be compatible with multiple operating systems such as Windows, Linux etc
- The system should be reliable and secure when communicating with hardware devices
- The system should use QR codes to scan books and cards, and to facilitate payments

Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes

Performance Requirements

The credit card processing software system must efficiently handle a large number of concurrent transactions, ensuring quick response times and high availability. It should be scalable to accommodate increased transaction loads and maintain optimal performance.

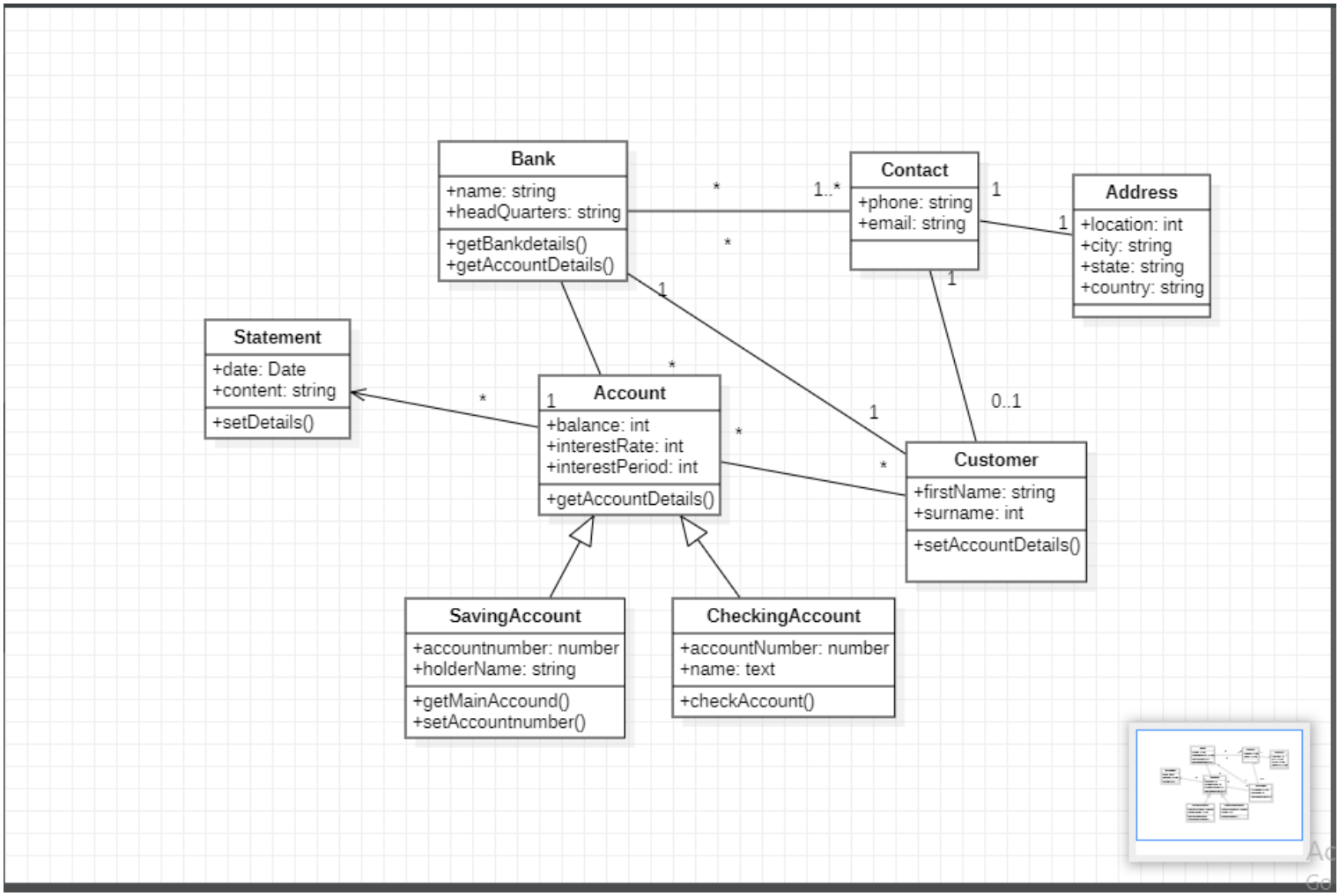
Design Constraints

The credit card processing software system must comply with industry standards and regulations, ensuring adherence to data security and privacy requirements. It should utilize secure encryption methods, follow PCI DSS guidelines, and implement robust authentication mechanisms.

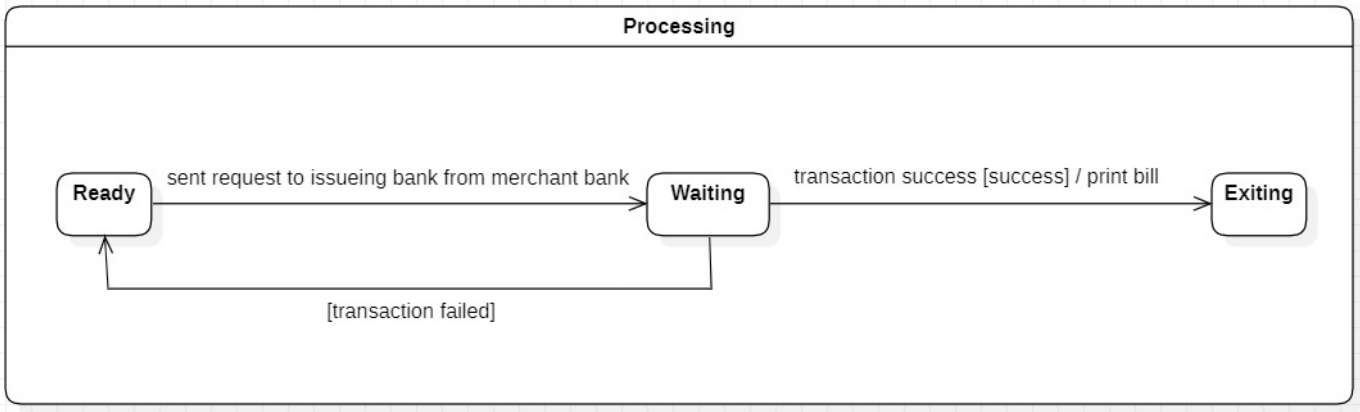
Non-functional Attributes

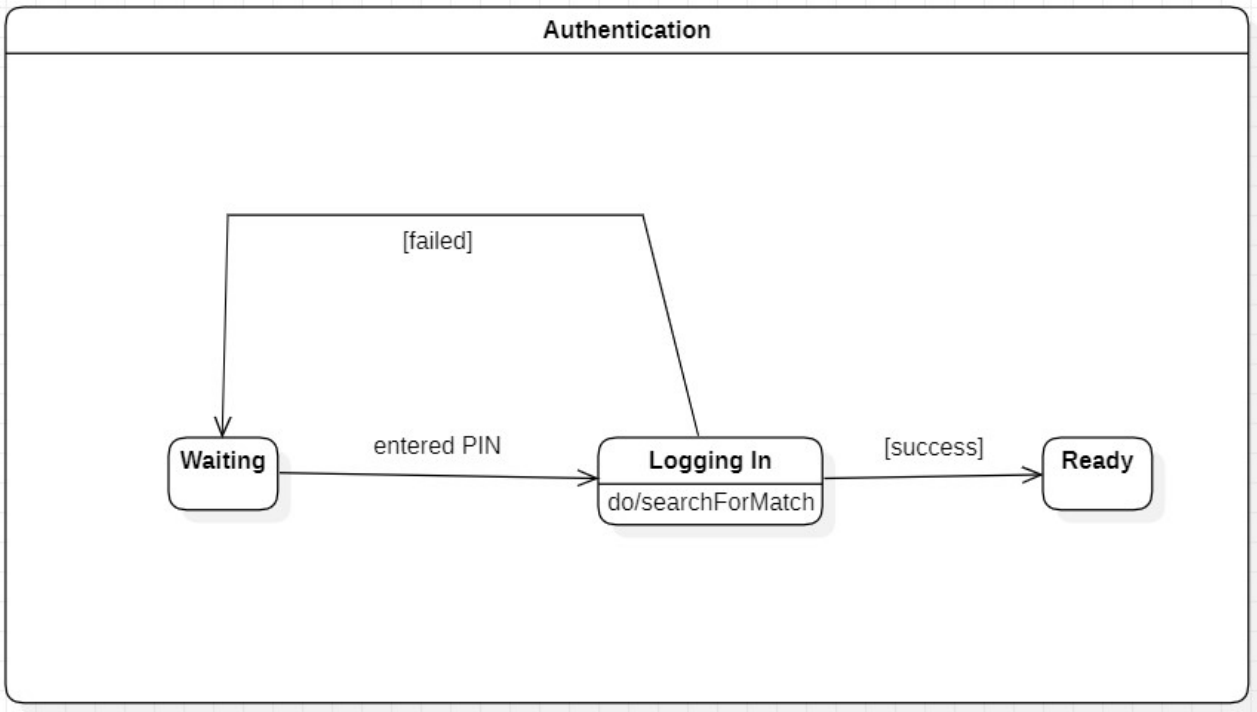
- Usability: The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.
- Accuracy: The data stored about the books and the fines calculated should be correct, consistent, and reliable.
- Availability: The System should be available for the duration when the library operates and must be recovered within an hour or less if it fails. The system should respond to the requests within two seconds or less.
- Performance: The system should be fast and accurate. It should handle expected and unexpected errors. It should be able to handle large amount of data.
- Security: The system should protect the data from unauthorized access and modification. It should use encryption and authentication techniques to ensure data security.
- Scalability: The system should be able to handle increasing number of users and books without degrading the performance or functionality.

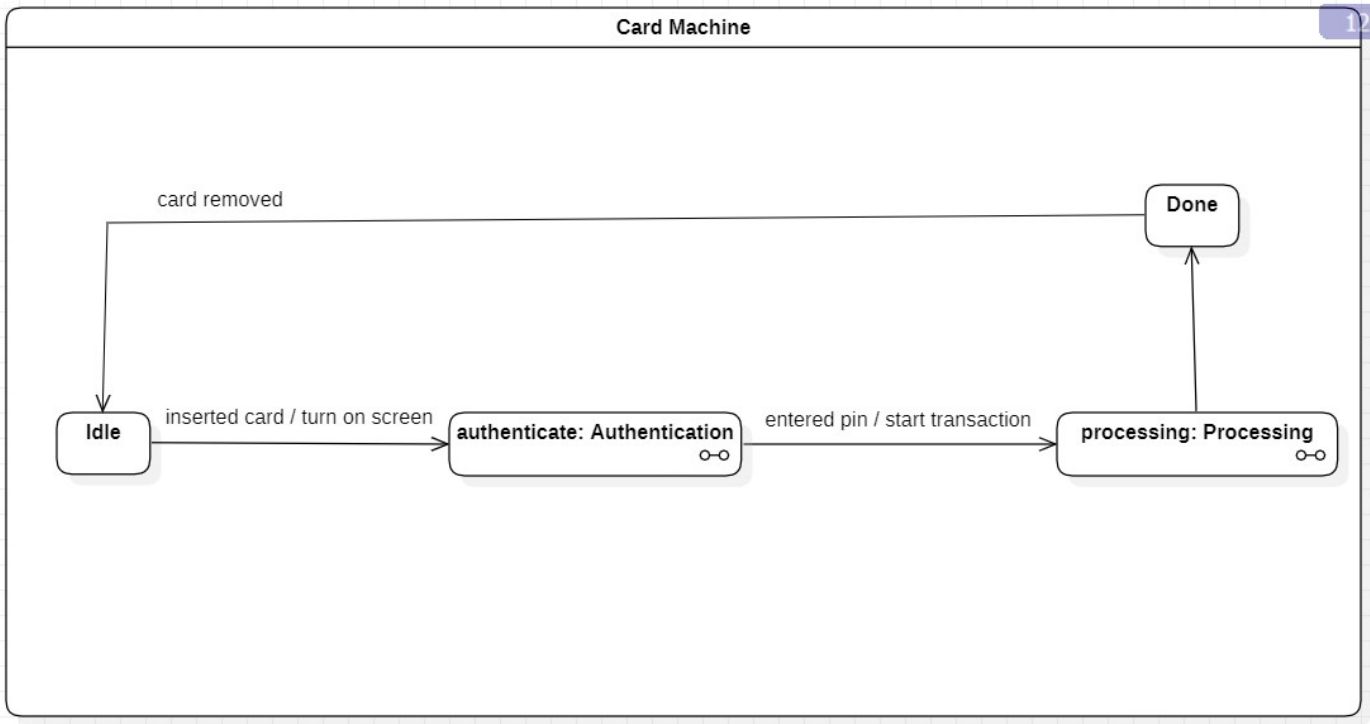
2.3 Class Diagram



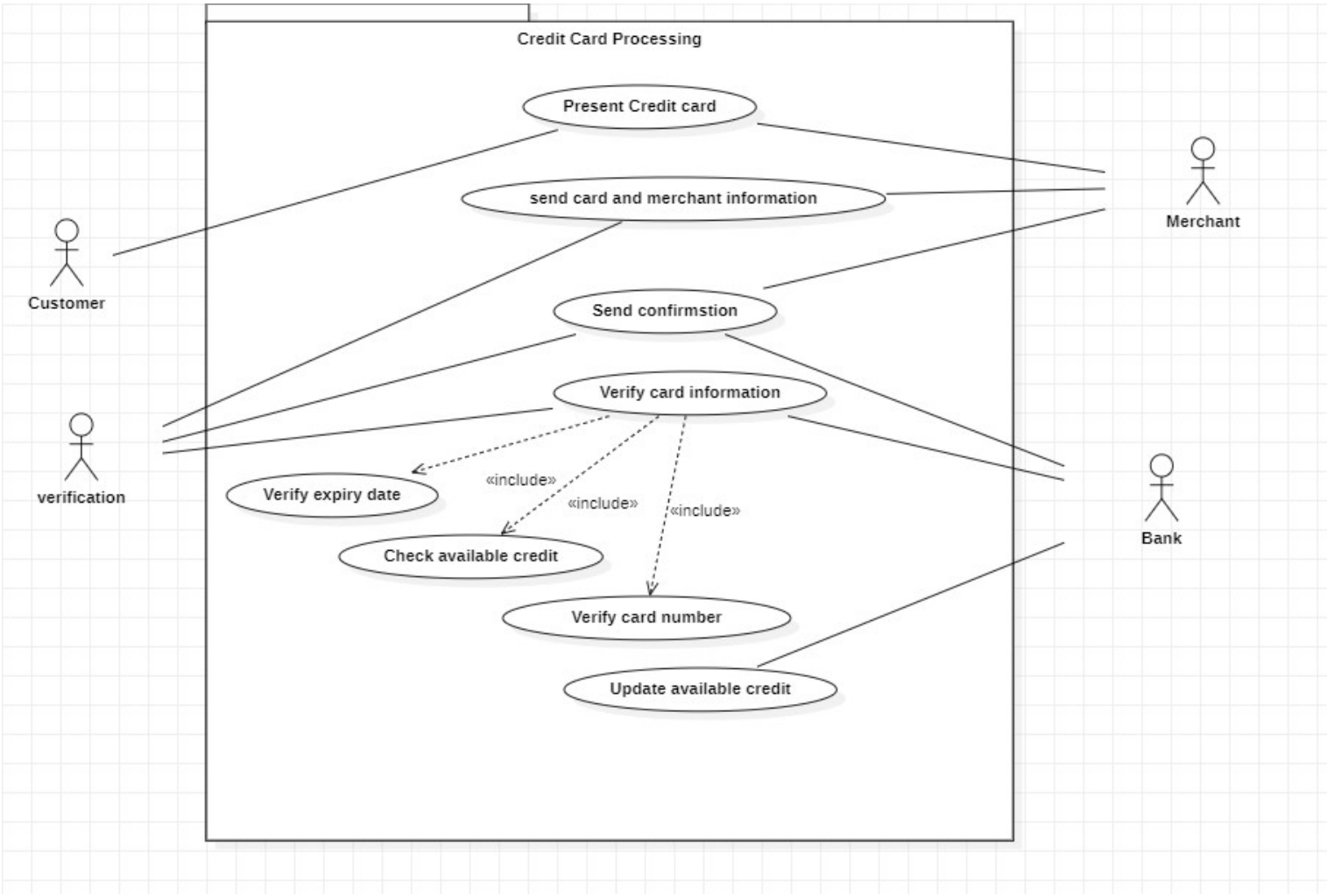
2.4 State Diagrams



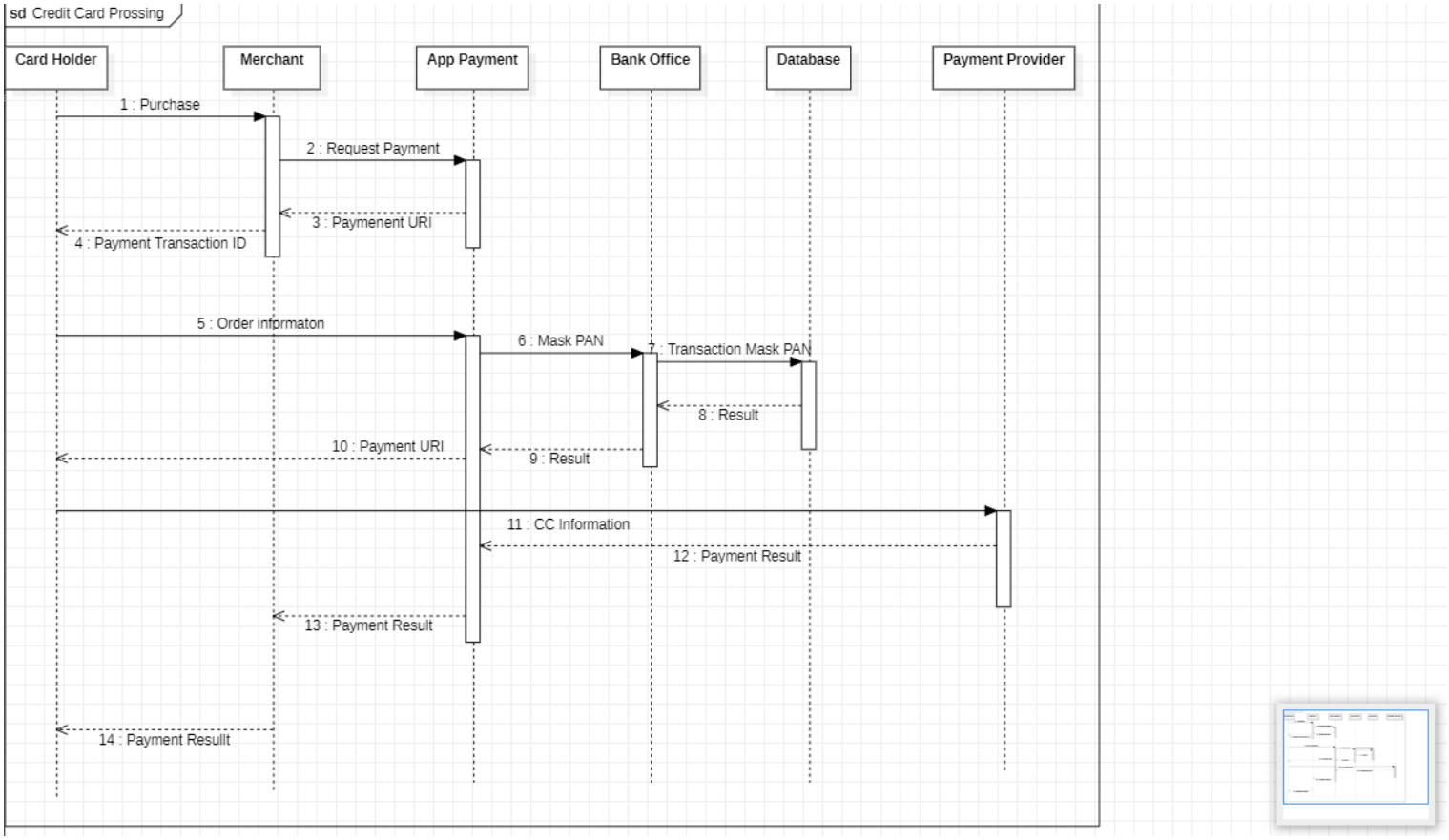



Card Machine
12 Jun
card removed
Done
Idle
inserted card / turn on screen
authenticate: Authentication
entered pin / start transaction
processing: Processing

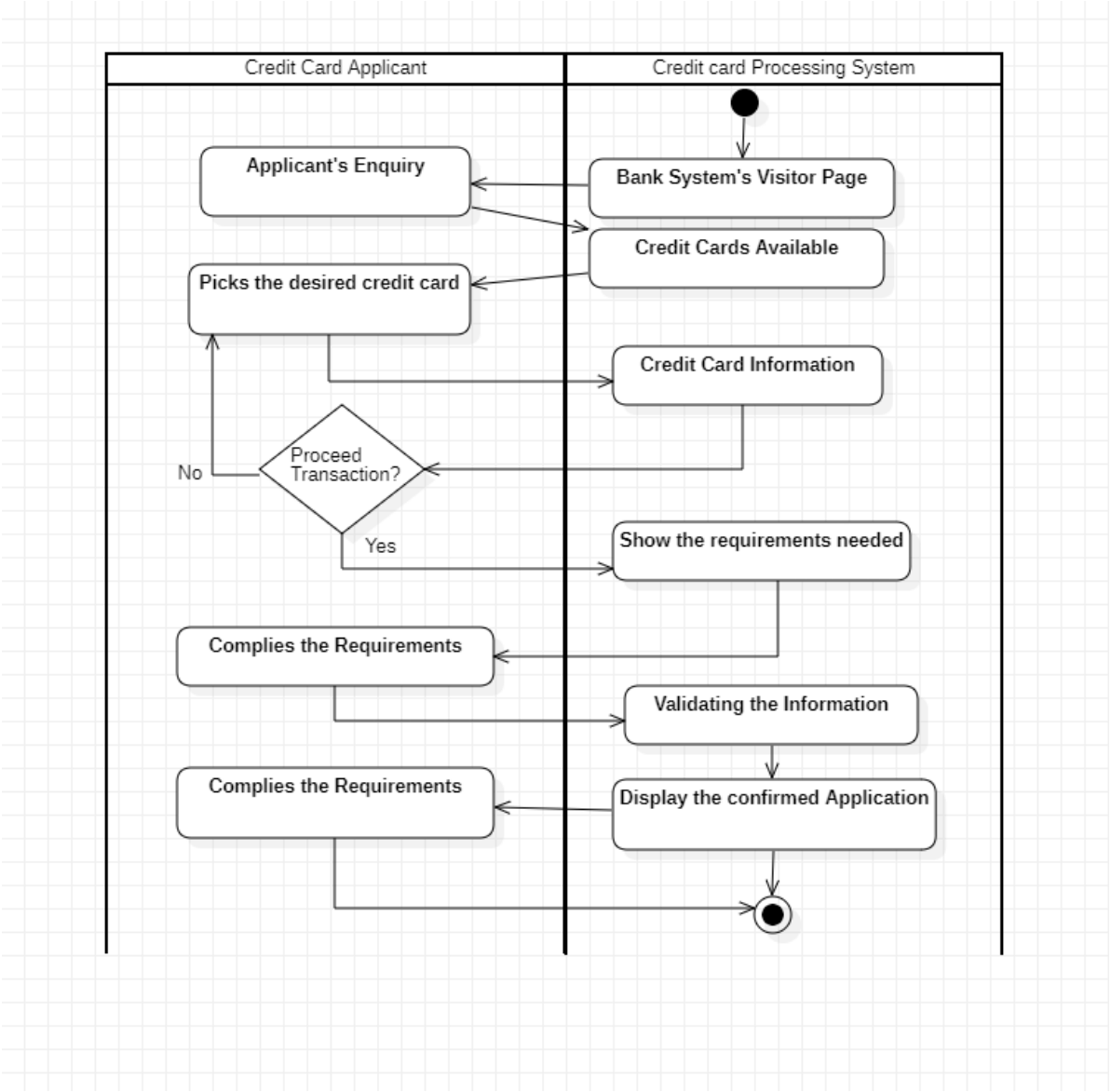
2.5 Use Case Diagram



2.6 Sequence Diagram



2.7 Activity Diagram



3. LIBRARY MANAGEMENT SYSTEM

3.1 Problem statement

With the world now moving towards digitalization of every possible task, it is inevitable that libraries stop relying on physical staff for operation. Library management should allow users to borrow books, return books, pay fines and issue cards.

3.2 Software Requirement Specification

Introduction

Purpose

Operate a library efficiently using automation, streamlining tasks involved in running a library such as cataloging, issues, managing membership details, placing fines for delayed returns etc

Scope of the document:

To specify the functional, behavioral and non-functional requirements of the software application that automates and streamlines various tasks involved in running a library.

Overview

A library management system is a software application that helps librarians to manage the library operations, such as acquiring, cataloging, circulating, and reporting on library materials. It also helps users to find and access library resources easily and efficiently.

General description:

A library management system is a software application that helps librarians to manage the library operations, such as acquiring, cataloging, circulating, and reporting on library

materials. It also helps users to find and access library resources easily and efficiently. Users can search for books, access e-resources, borrow and renew books, view their account details including due dates, participate in online forums etc

Functional requirements:

- System should allow the librarian to add, edit, delete and view the details of books, members and loans
- Users should be allowed to borrow, renew and return books using login credentials and barcode scanners
- System should send notifications to the members and librarian about the overdue books and pending payments
- System should generate reports on the inventory, circulation, and financial transactions of the library
- System should keep track of due dates of books issued and active membership period

Interface requirements

User Interface

- UI should be simple, intuitive, responsive and consistent
- Interface should support different languages and devices as needed
- UI should display availability and location of books in the library

Hardware Interface:

- The system should be compatible with multiple operating systems such as Windows, Linux etc
- The system should be reliable and secure when communicating with hardware devices
- The system should use QR codes to scan books and cards, and to facilitate payments

Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes

Performance Requirements

- The number of pages should be minimized for the user's convenience
- Adding members, borrowing and renewing books should be very quick.
- There should be a procedure for when there is loss of data due to failure of storage device or similar reason.

Design Constraints

The information of all users, books and libraries must be stored in a database that is

accessible by the website.

The Online Library System is running 24 hours a day.

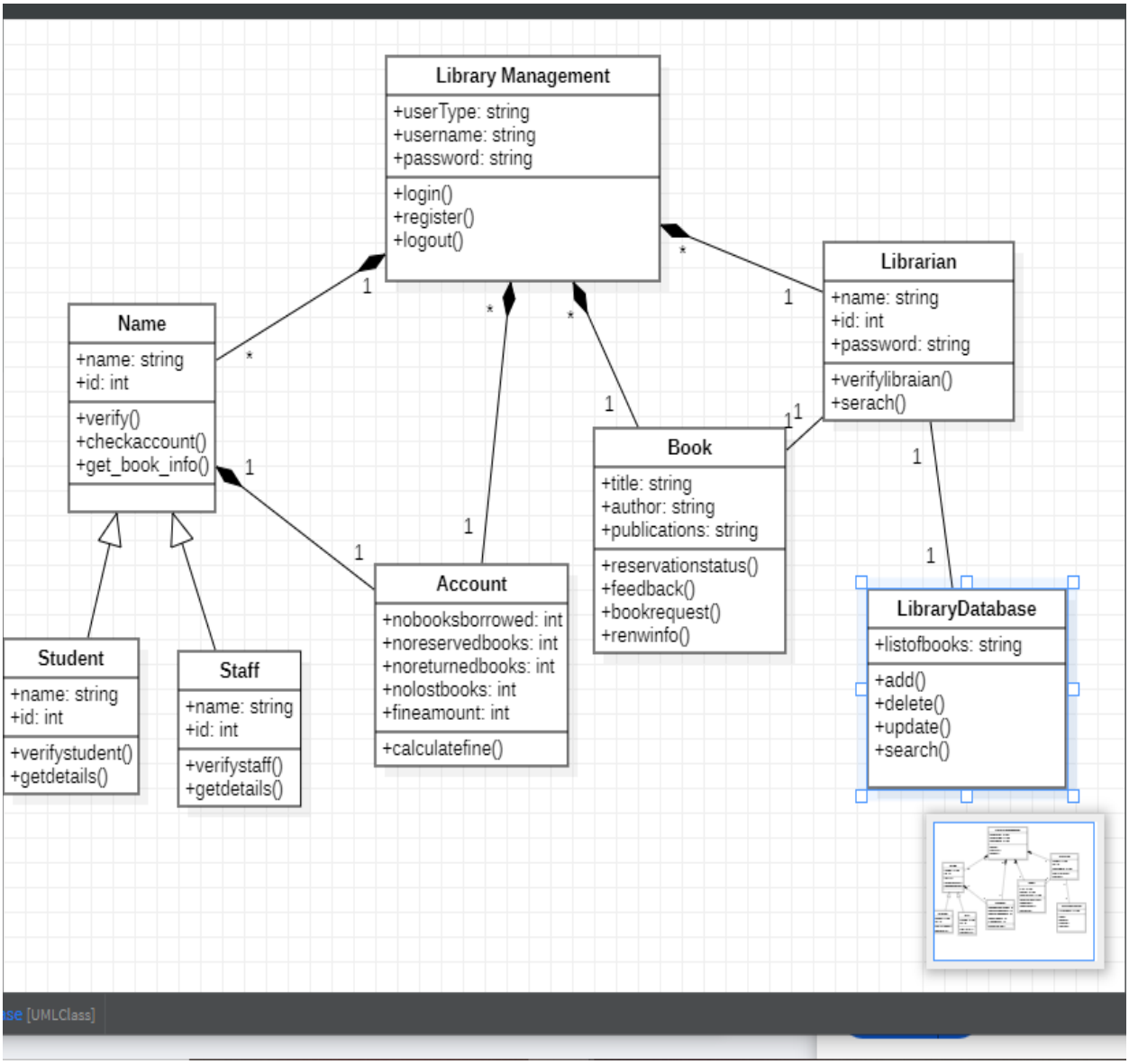
Users may access from any computer that has Internet browsing capabilities and an Internet connection.

Users must have their correct usernames and passwords to enter into their online accounts and do actions.

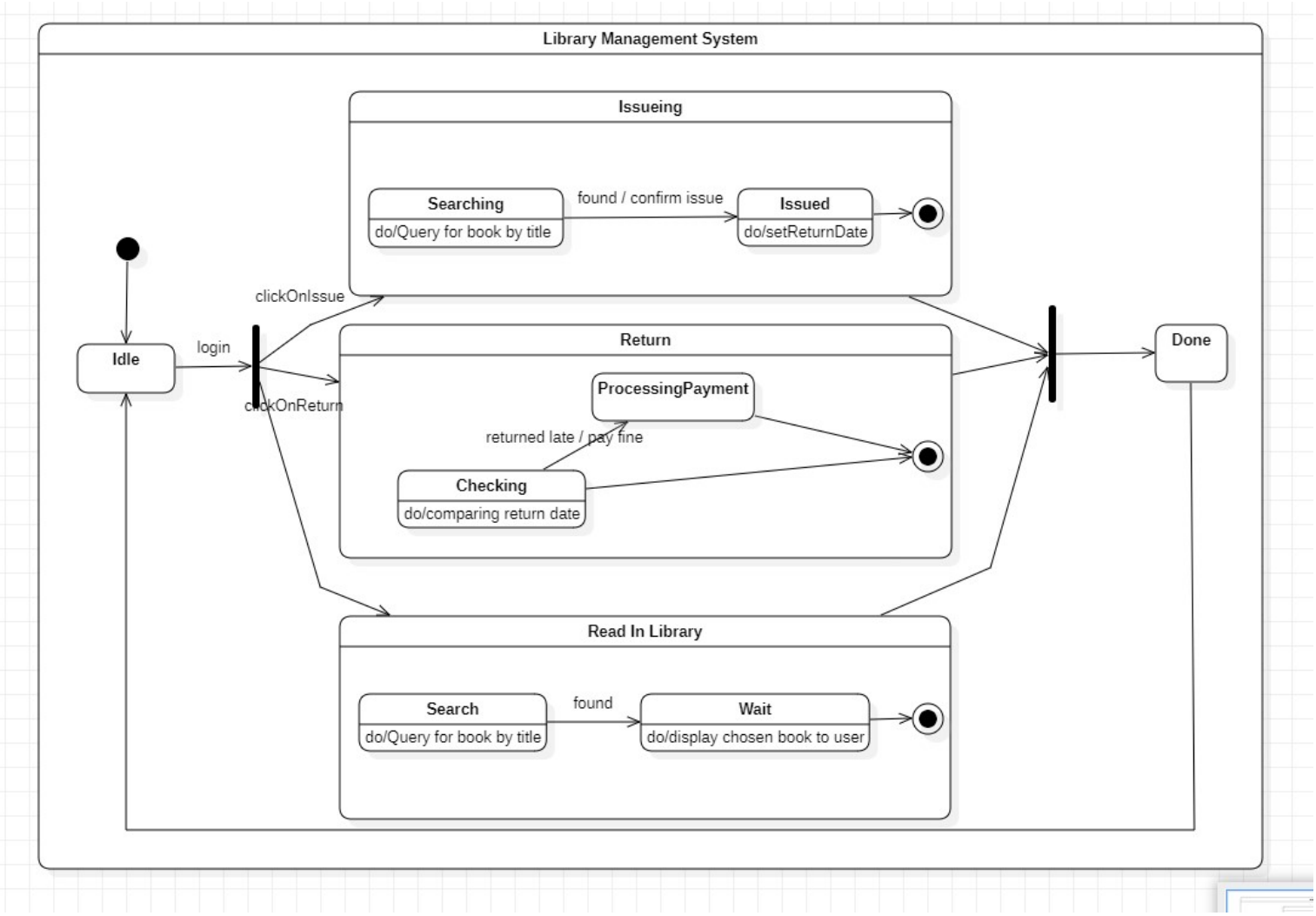
Non-functional Attributes

- Usability: The UI should be simple enough for everyone to understand and get the relevant information without any special training. Different languages can be provided based on the requirements.
- Accuracy: The data stored about the books and the fines calculated should be correct, consistent, and reliable.
- Availability: The System should be available for the duration when the library operates and must be recovered within an hour or less if it fails. The system should respond to the requests within two seconds or less.
- Performance: The system should be fast and accurate. It should handle expected and unexpected errors. It should be able to handle large amount of data.
- Security: The system should protect the data from unauthorized access and modification. It should use encryption and authentication techniques to ensure data security.
- Scalability: The system should be able to handle increasing number of users and books without degrading the performance or functionality.

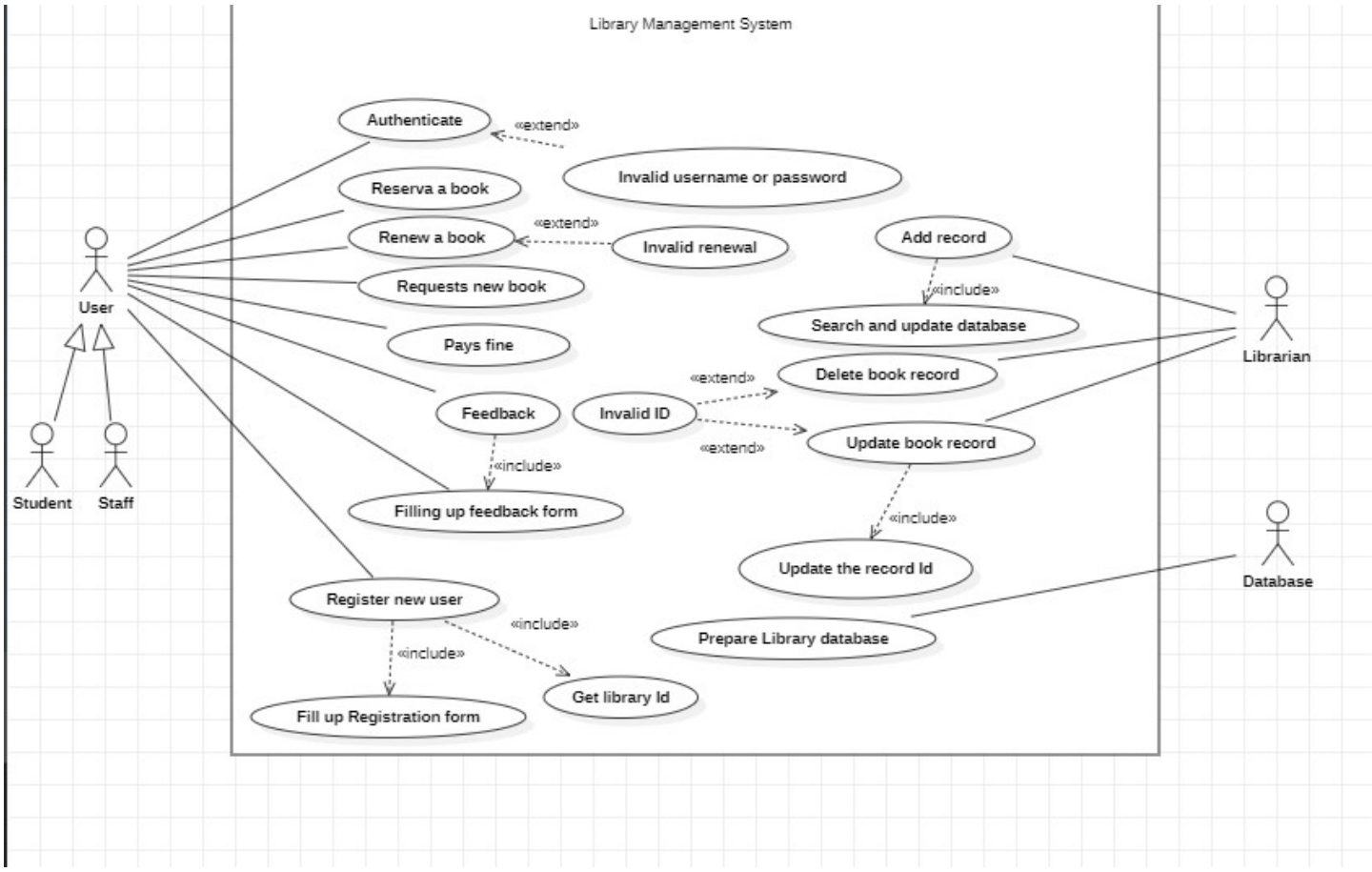
3.3 Class Diagram



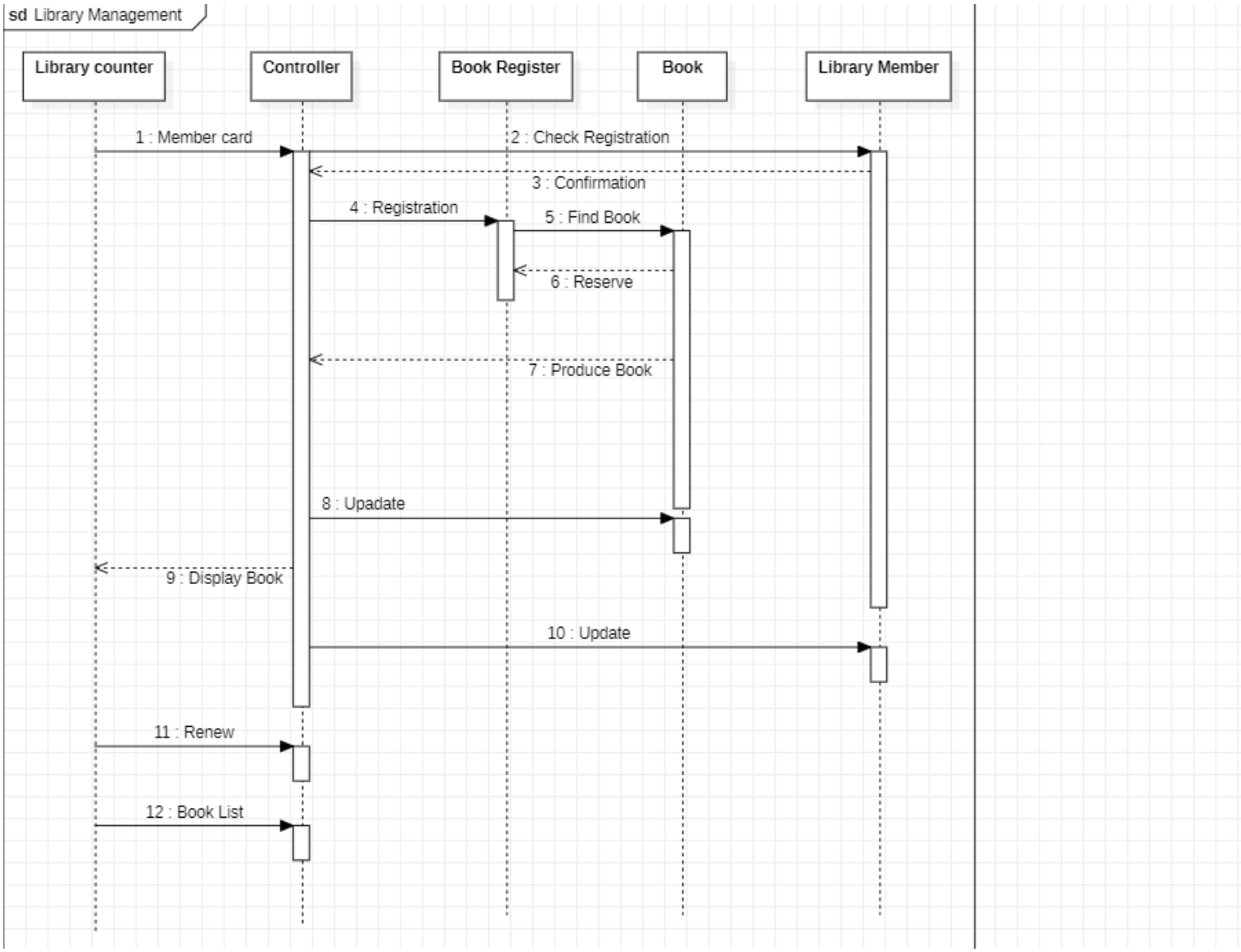
3.4 State Diagrams



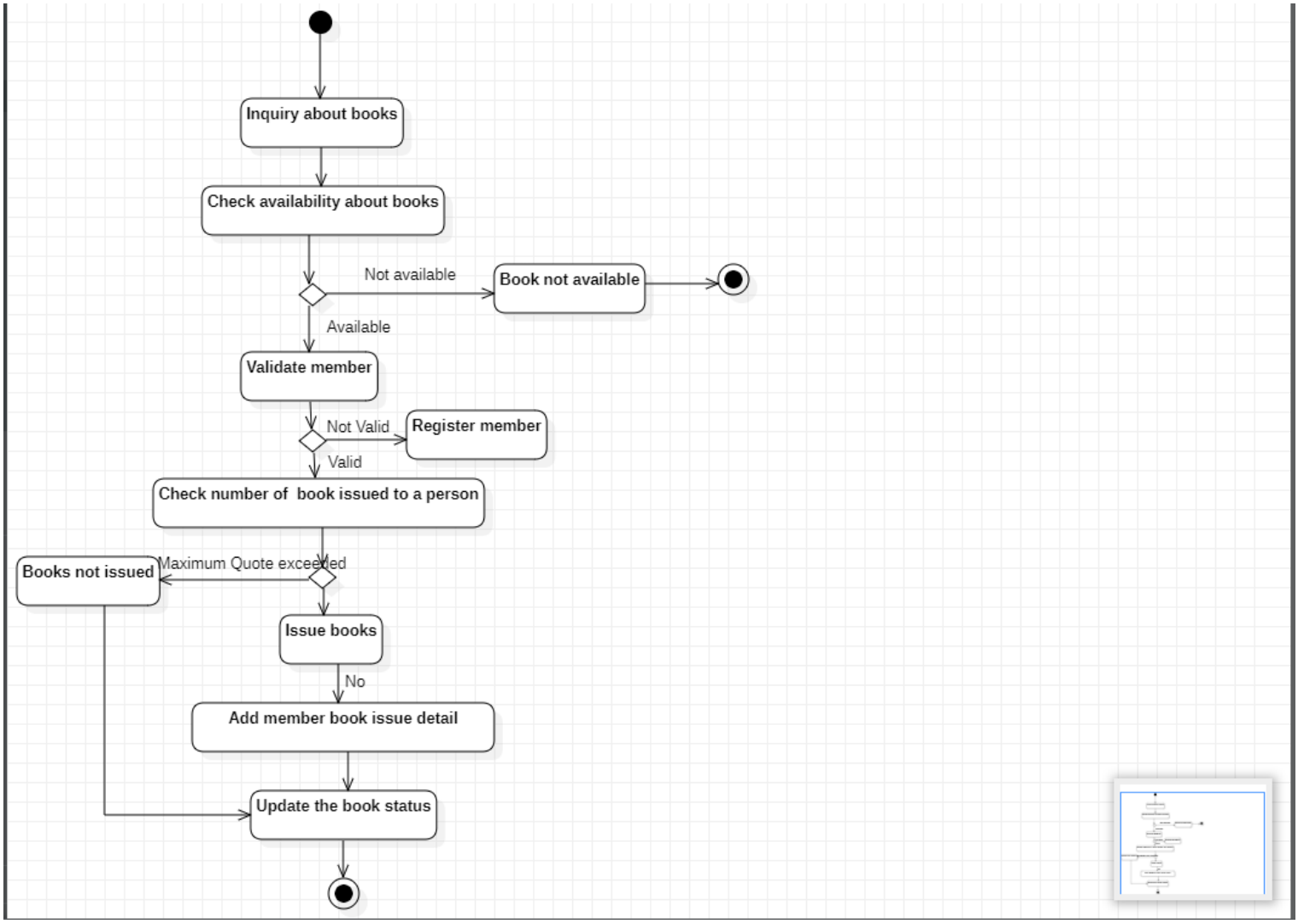
3.5 Use Case Diagram



3.6 Sequence Diagram



3.7 Activity Diagram



4. ONLINE SHOPPING SYSTEM

4.1 Problem statement

Online shopping is a convenient and popular way of buying products and services. However, online shoppers face many challenges such as finding the best deals, comparing products, ensuring security and privacy, and managing orders and deliveries. An online shopping system is needed to provide a user-friendly and reliable platform that can address these challenges and enhance the customer experience.

4.2 Software Requirement Specification

Introduction

Purpose:

The purpose of an online shopping system is to provide a convenient and efficient way for customers to buy products and services from various vendors over the internet. It also aims to offer a variety of features and functions that can help customers find, compare, and purchase the items they need or want. Additionally, it strives to ensure the security and privacy of the customers' personal and financial information. Furthermore, it seeks to facilitate the order processing and delivery management for both customers and vendors.

Scope of this document

The core of the system is to get the online registration form (with details such as name, address etc.,) filled by the applicant whose testament is verified for its genuineness by the Online Shopping system with respect to the already existing information in the database. It adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner.

Overview

Online shopping is a convenient and popular way of buying products and services. However, online shoppers face many challenges such as finding the best deals, comparing products, ensuring security and privacy, and managing orders and deliveries.

General Description:

An online shopping system is a software application that enables customers to buy products and services from various vendors over the internet. Customers can search, compare, and select the items they want and pay for them using different payment methods. The system also handles the order processing and delivery management for both customers and vendors. The system provides a user-friendly and secure interface that can be accessed from any device with an internet connection.

Functional Requirements:

The system should allow vendors to add, edit and delete their products and services

Customer should be able to browse through products

The system should enable the customers to add, remove, and update their items in their shopping cart

The system should redirect the user to a payment gate

Interface Requirements

User Interface

- UI should be simple, intuitive, responsive and consistent
- Interface should support different languages and devices as needed
- UI should display availability and location of books in the library

Hardware Interface:

- The system should be compatible with multiple operating systems such as Windows, Linux etc
- The system should be reliable and secure when communicating with hardware devices
- The system should use QR codes to scan passports, and to facilitate payments

Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes

Performance Requirements

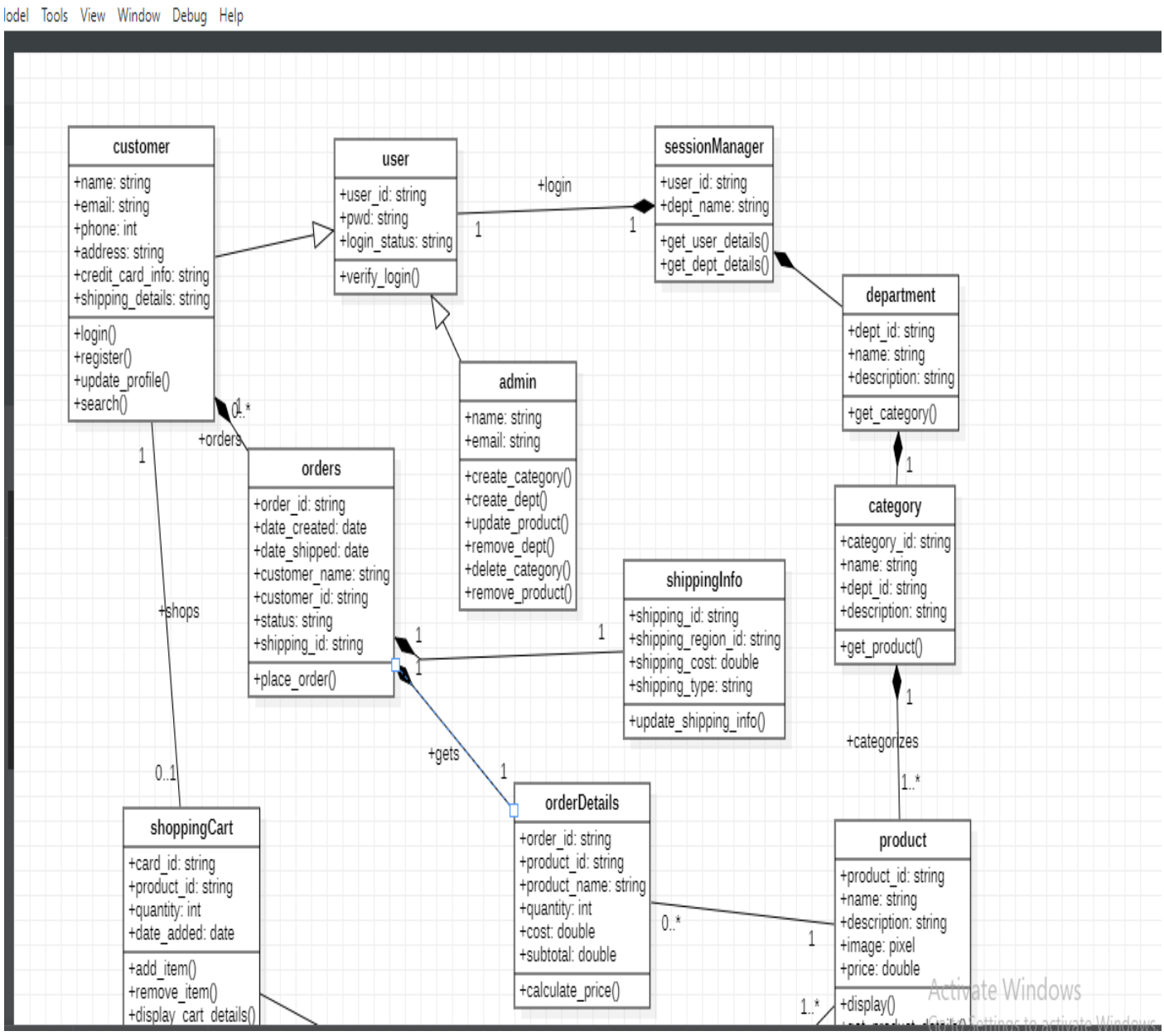
- The number of pages should be minimized for the user's convenience
- Page reloads should be fast
- There should be a procedure for when there is loss of data due to failure of storage device or similar reason.
- In case of a transaction failure, user is to be alerted

Design Constraints

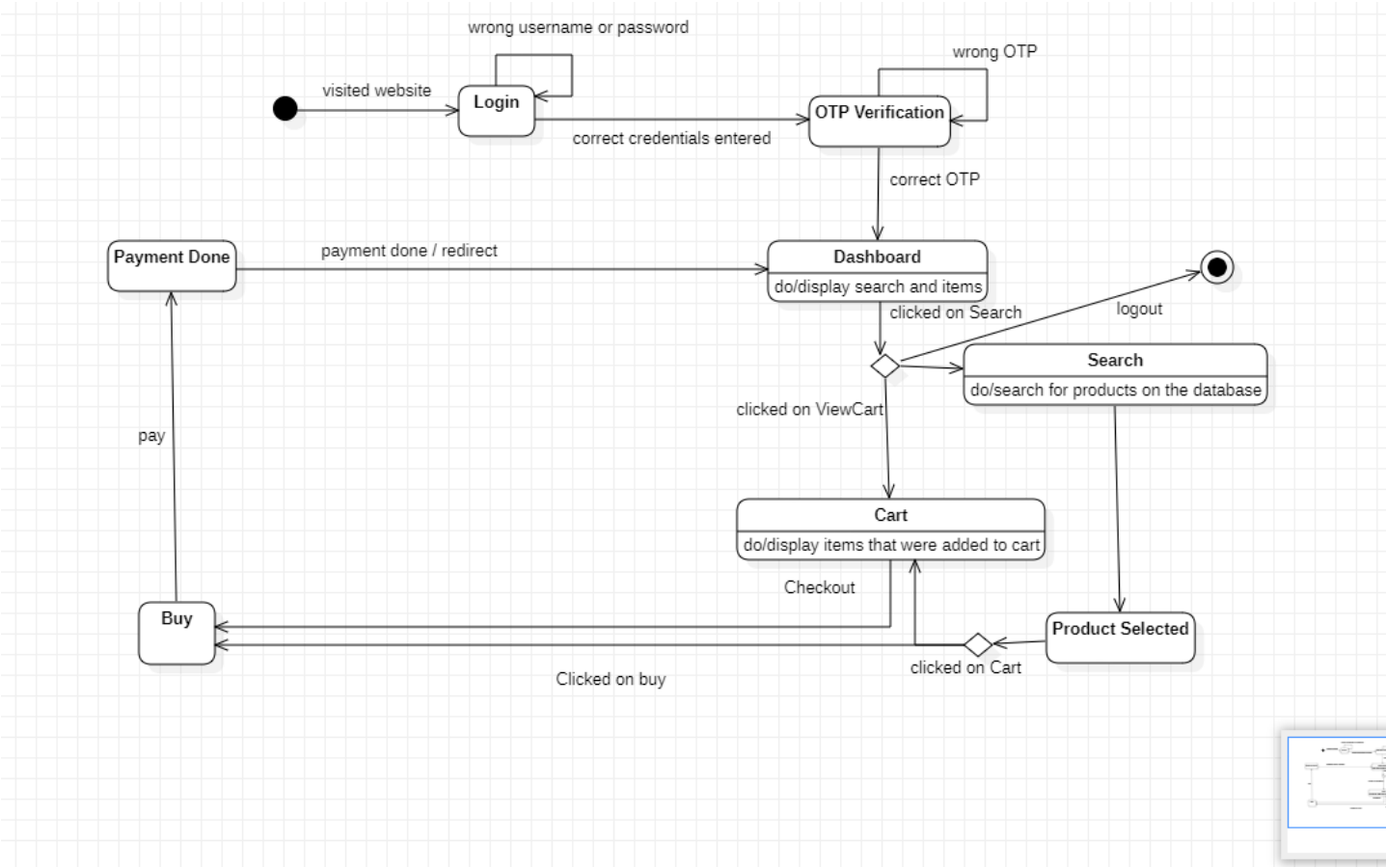
- Working with the constraints set by the government
- Respect industry standards and follow best practices of software development
- Non-Functional Requirements
- Performance: The system should be fast and accurate. It should handle expected and unexpected errors. It should be able to handle large amounts of data.
- Security: The system should protect the data from unauthorized access and modification. It should use encryption and authentication techniques to ensure data security.

- Scalability: The system should be able to handle an increasing number of users and applications without degrading the performance or functionality.

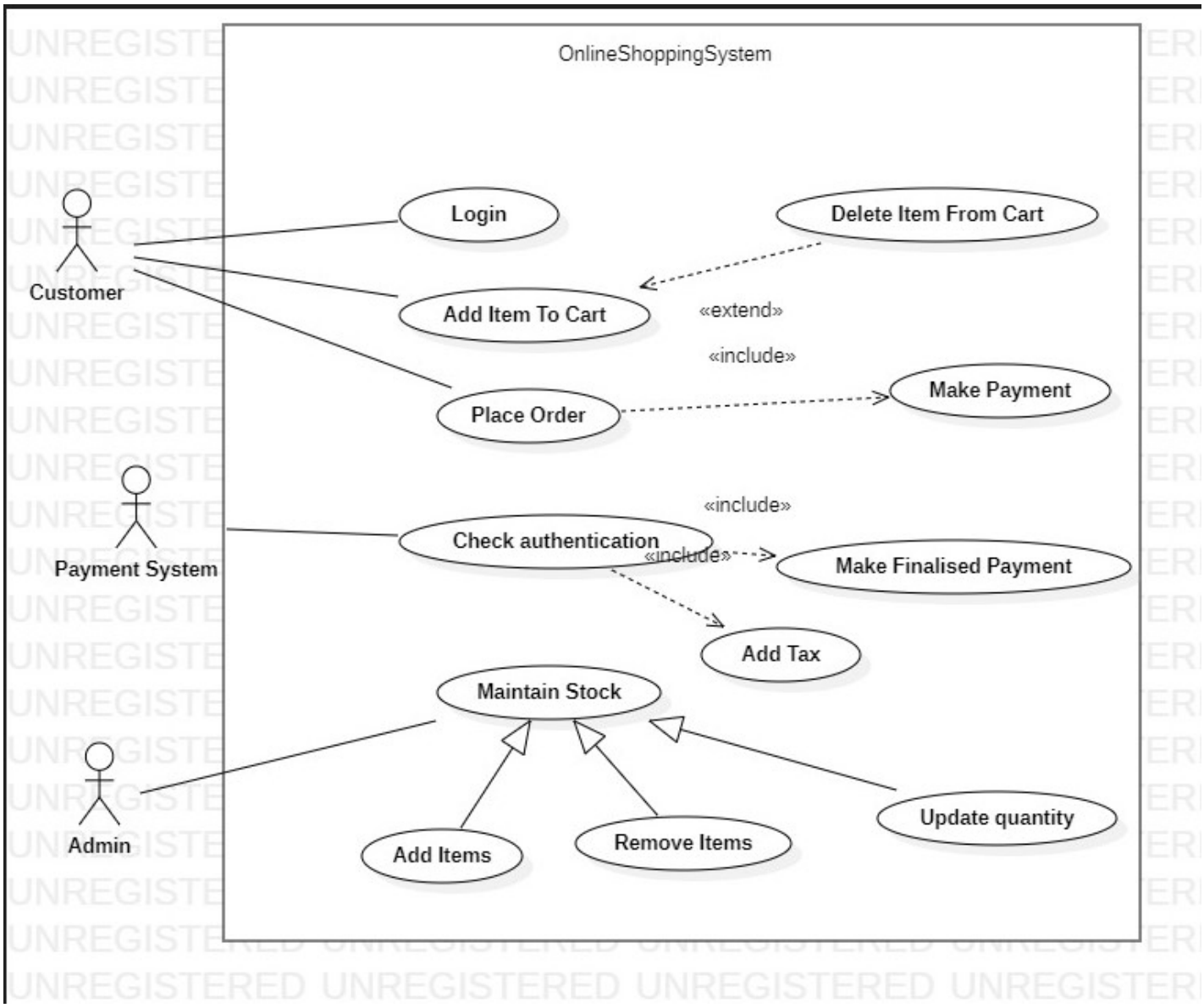
4.3 Class Diagram



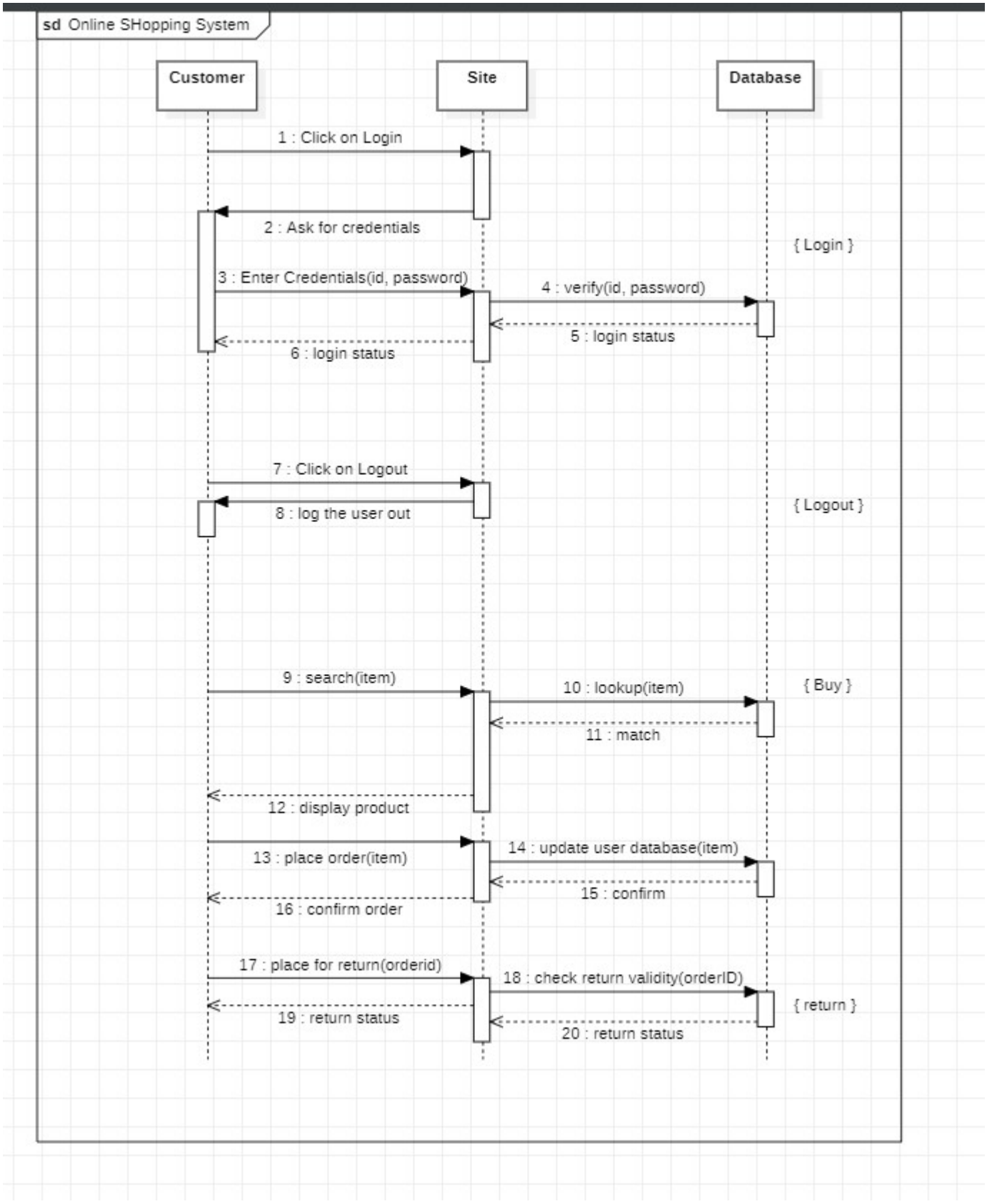
4.4 State Diagrams



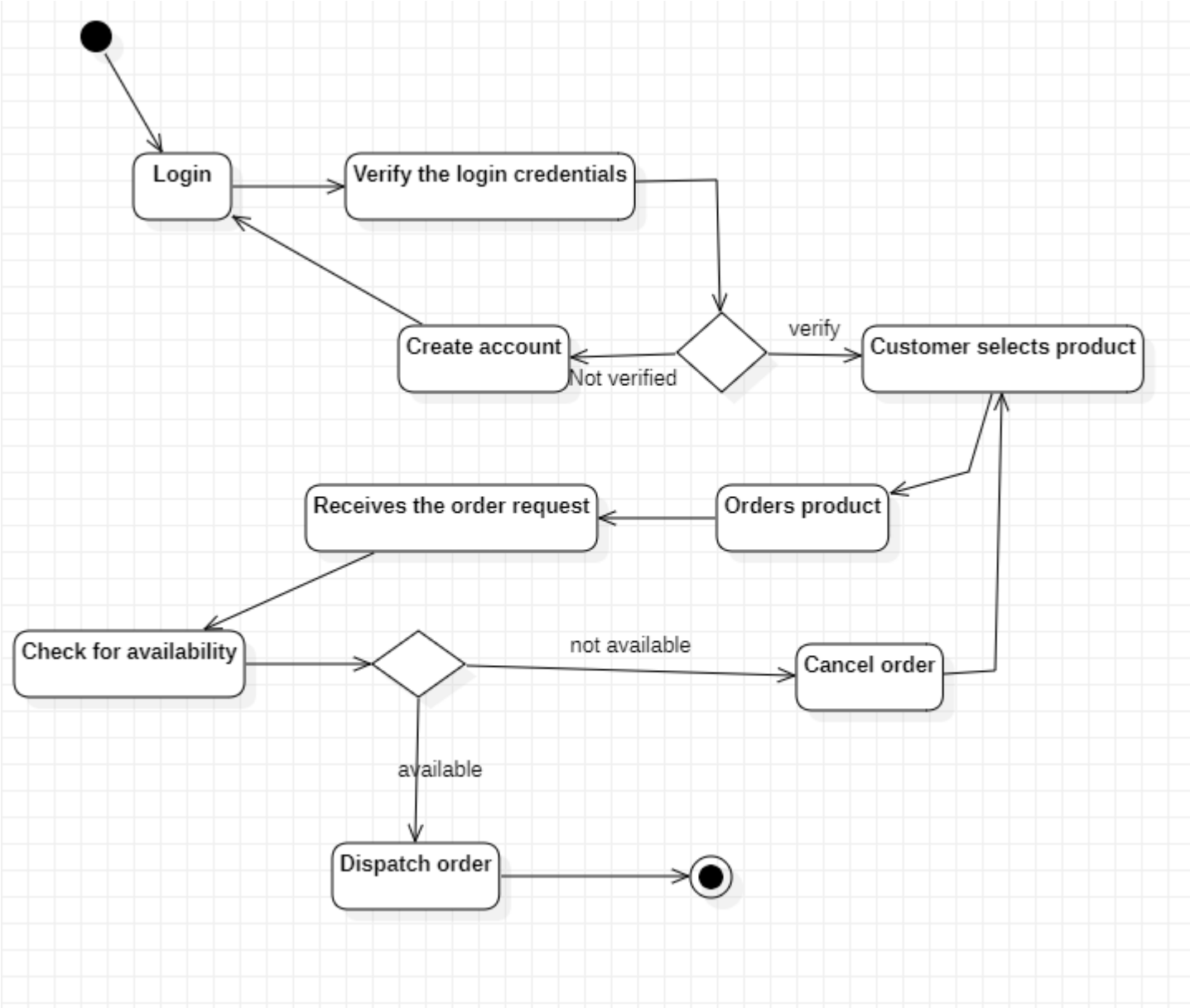
4.5 Use Case Diagram



4.6 Sequence Diagram



4.7 Activity Diagram



5. PASSPORT AUTHENTICATION

5.1 Problem statement

Passport automation systems streamline the process of passport issuance and management by using advanced technology to manage applications, track progress, and generate passports. Passport management involves the administration of passport-related processes, including application processing, verification, data entry, and record keeping. The benefits of passport automation systems and passport management include speed, accuracy, security, and convenience.

5.2 Software Requirement Specification

Introduction

Purpose:

The purpose of this document is to provide a detailed description of the requirements for the Passport Automation System. This system will be used to automate the process of passport issuance and management, enabling faster and more efficient processing of passport applications.

Scope of this document

The core of the system is to get the online registration form (with details such as name, address etc.,) filled by the applicant whose testament is verified for its genuineness by the Passport Automation System with respect to the already existing information in the database. It adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner.

Overview

The Passport Authentication System provides an online interface to the user where they can fill in their personal details and submit the necessary documents.

General Description:

A passport automation system is a software that helps to issue passports to applicants in a fast and efficient way. It allows the applicants to fill in their personal details and submit their documents online. It also verifies the authenticity of the information and documents with the existing database. It communicates with the passport authority and the local police for approval and verification. It notifies the applicants about the status of their application and the date of document verification.

Functional Requirements:

Passport Information: The system should provide an online interface to the user where they can fill in their personal details and submit the necessary documents (may be by scanning).

The system should allow the user to apply for a new passport, renew an existing passport, or change any details in their passport.

The system should allow the user to pay the required fees for their passport application through online mode or offline mode.

The system should authenticate the user by using their username and password or by scanning their barcode.

Interface Requirements

User Interface

- UI should be simple, intuitive, responsive and consistent
- Interface should support different languages and devices as needed
- UI should display availability and location of books in the library

Hardware Interface:

- The system should be compatible with multiple operating systems such as Windows, Linux etc
- The system should be reliable and secure when communicating with hardware devices
- The system should use QR codes to scan passports, and to facilitate payments

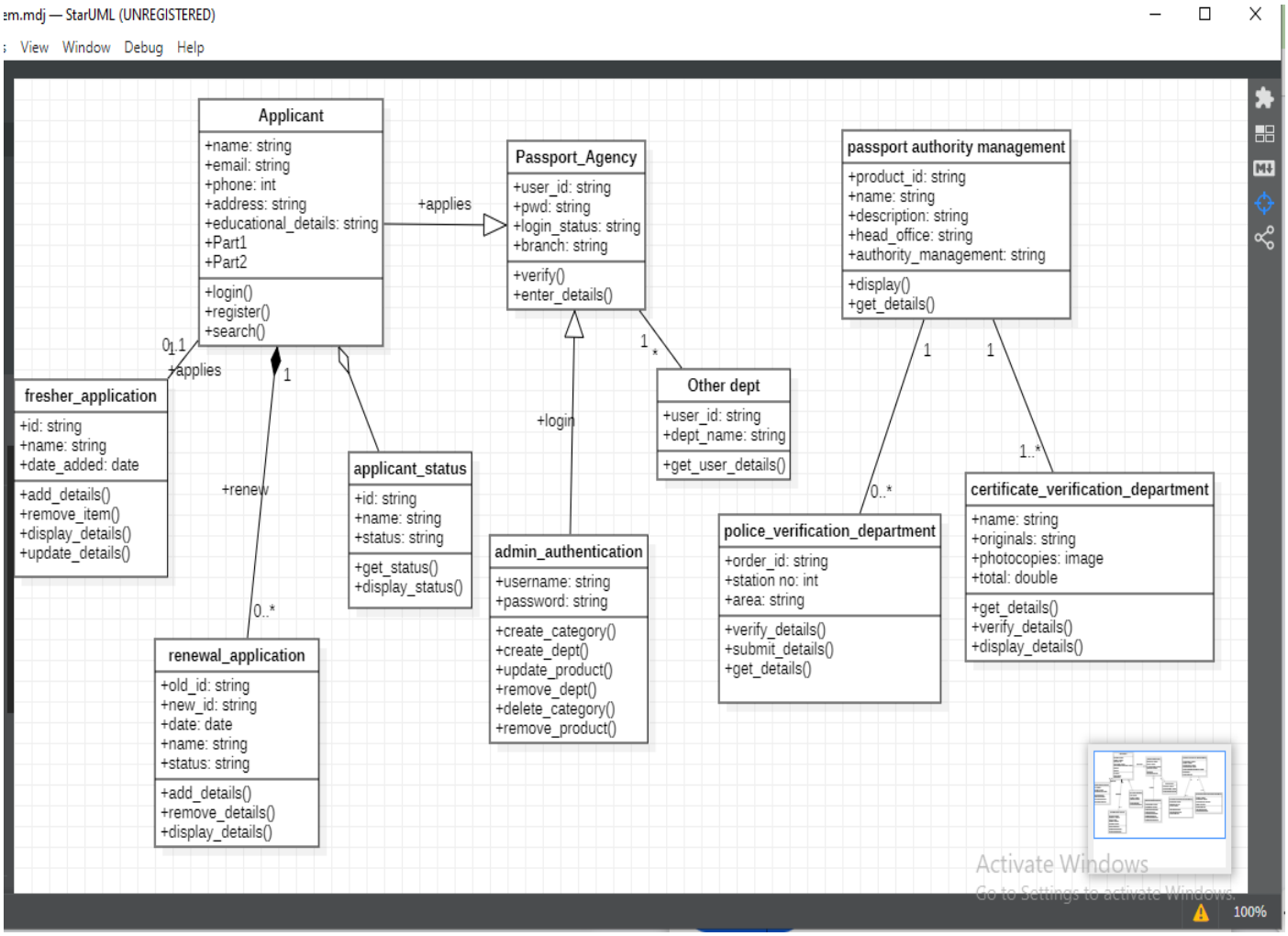
Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes
- Performance Requirements
- The number of pages should be minimized for the user's convenience
- Verifying passports should be a very swift process
- There should be a procedure for when there is loss of data due to failure of storage device or similar reason.

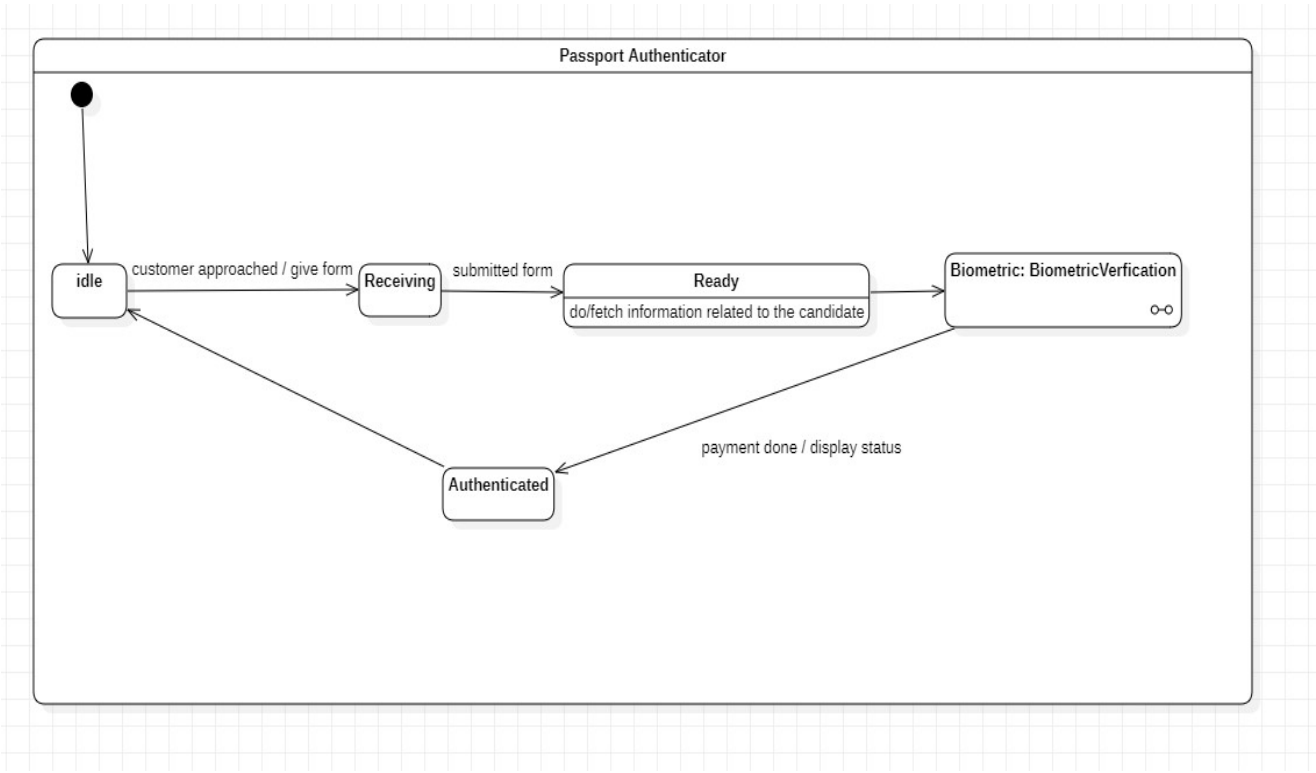
Design Constraints

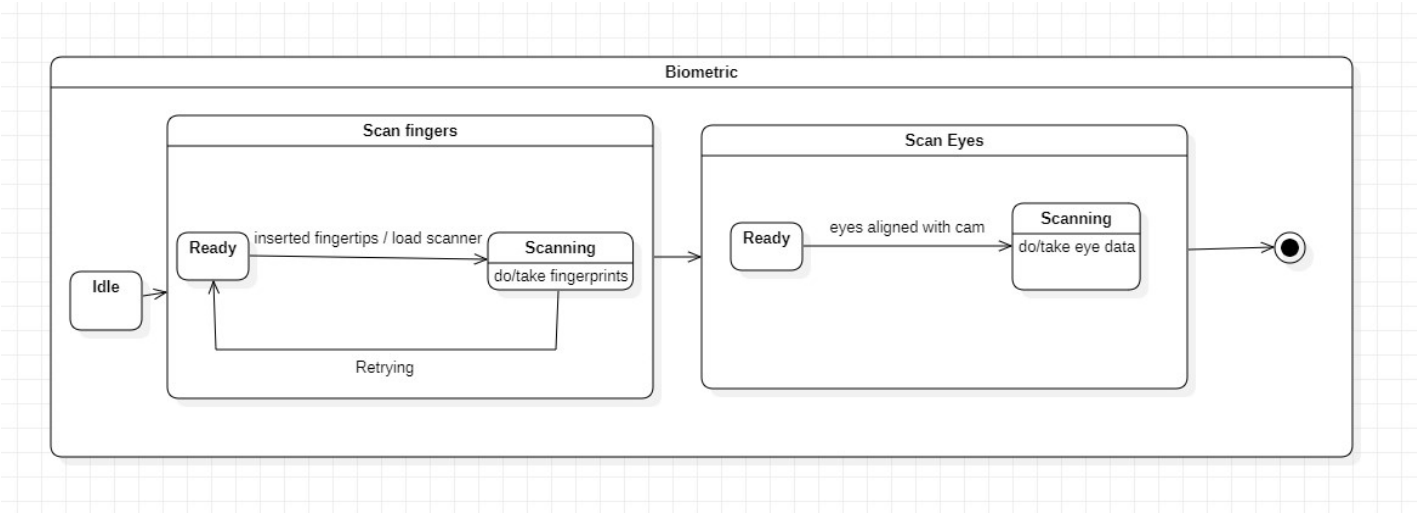
- Working with the constraints set by the government
- Respect industry standards and follow best practices of software development
- Non-Functional Requirements
- Performance: The system should be fast and accurate. It should handle expected and unexpected errors. It should be able to handle large amounts of data.
- Security: The system should protect the data from unauthorized access and modification. It should use encryption and authentication techniques to ensure data security.
- Scalability: The system should be able to handle an increasing number of users and applications without degrading the performance or functionality.

5.3 Class Diagram

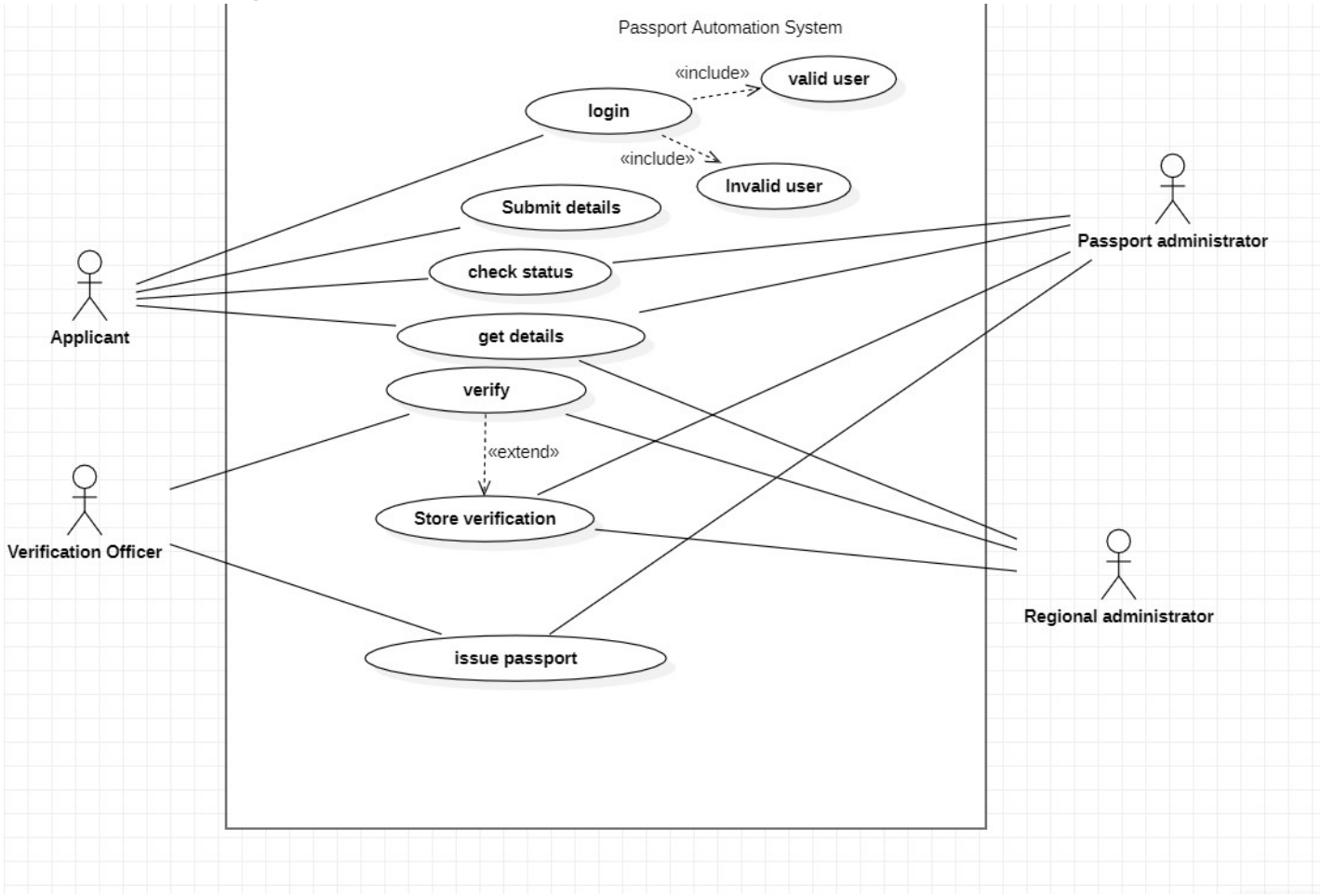


5.4 State Diagrams

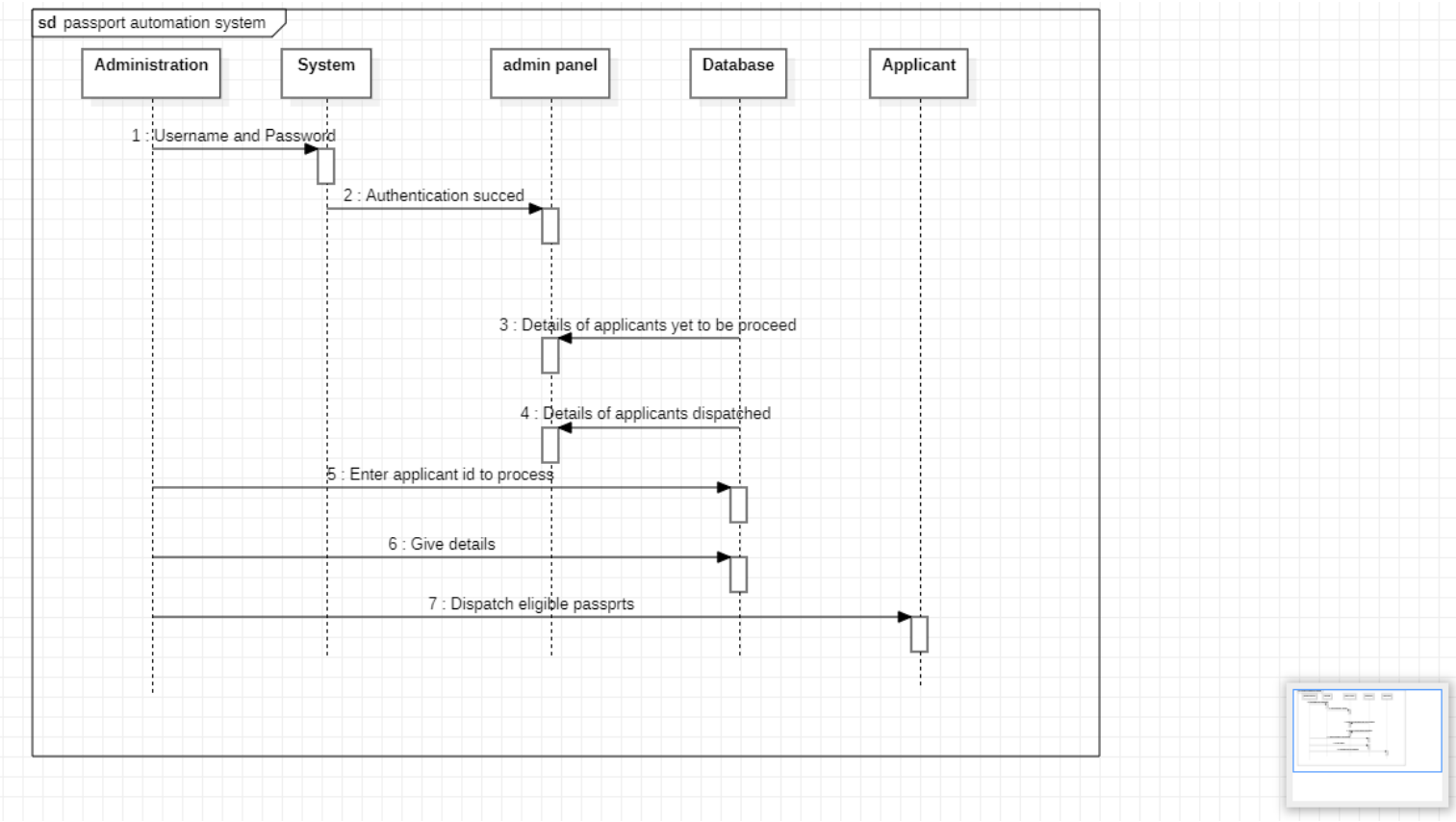




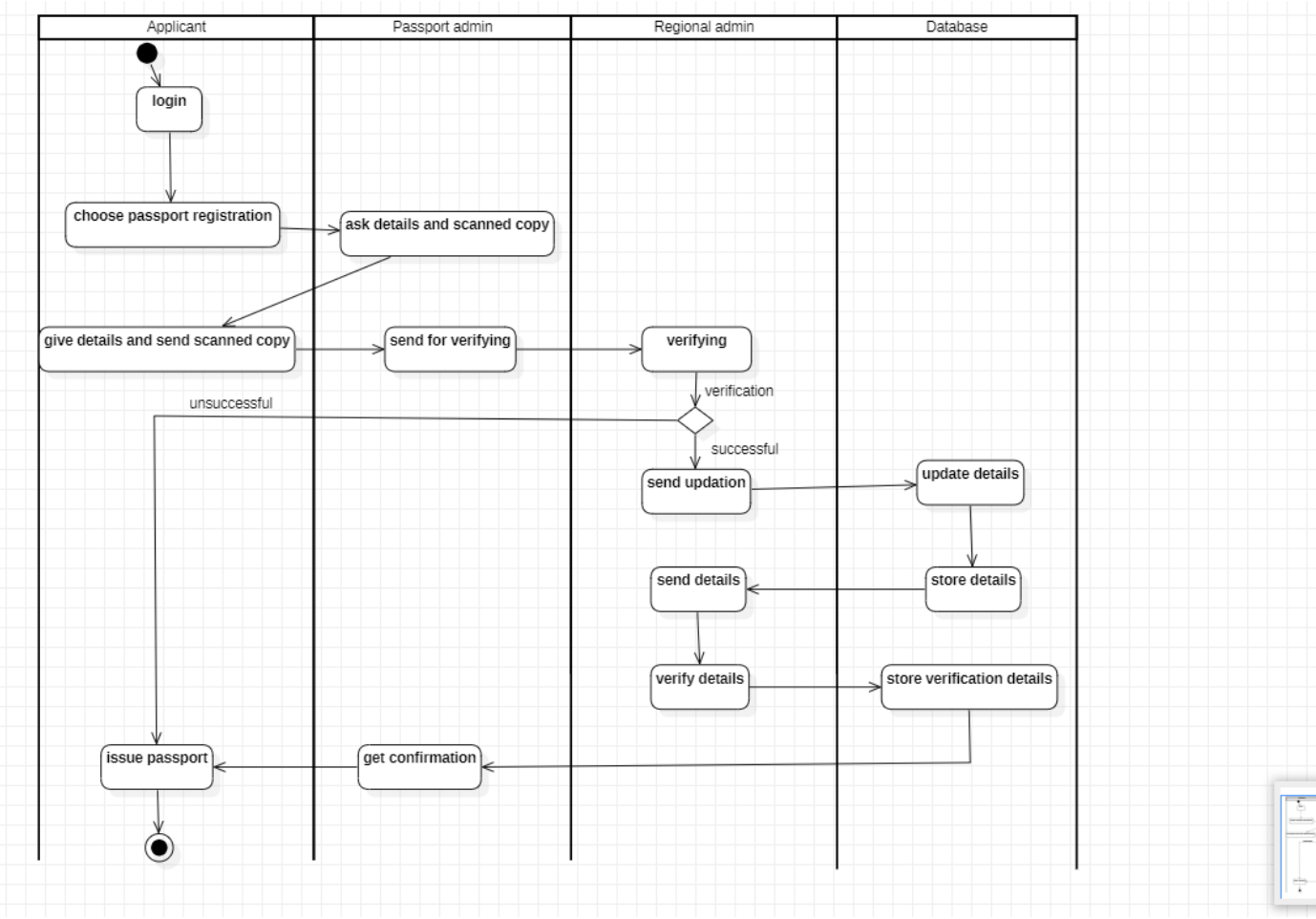
5.5 Use Case Diagram



5.6 Sequence Diagram



5.7 Activity Diagram



6. RAILWAY RESERVATION SYSTEM

6.1 Problem statement

Railway Reservation System is a system used for booking tickets over internet.Any Customer Can book tickets for different trains.Software has to be developed for automating the manual reservation system of railway.The system should be standalone in nature.It should be designed to provide functionalists like booking of tickets in which a user should be able to applied for tickets of any train and of any class.The software takes the current system date and time as the date of issue and calculates the amount to be paid by the user. It also provide the functionality of cancellation of tickets.

6.2 Software Requirement Specification

Introduction

Purpose:

To provide a convenient and efficient way for customers to book and manage their train journeys. To improve the operational efficiency and revenue generation of the railway department.

Scope of this document

The core of the system is to get the online registration form (with details such as name, address etc.,) filled by the applicant whose testament is verified for its genuineness by the Railway Reservation system with respect to the already existing information in the database. It adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner.

Overview

The Railway Reservation System provides an online interface to the user where they can fill in their personal details, submit the necessary documents and book tickets.

General Description:

Railway reservation is a complex and tedious process that involves booking tickets, checking seat availability, finding train schedules, and paying fares. The current manual system is prone to errors, delays, and frauds. Customers face many challenges such as long queues, limited payment options, lack of information, and poor customer service. A railway reservation system is needed to provide a convenient and efficient way for customers to book and manage their train journeys. It also aims to improve the operational efficiency and revenue generation of the railway department.

Functional Requirements:

The system should allow the passengers to register themselves by providing their personal details, such as name, email, address, phone number, etc.

The system should allow the passengers to check their PNR status by entering their PNR number.

The system should allow the passengers to cancel their tickets by entering their PNR number and a cancel ticket request.

The system should allow the railway department to add, edit, and delete the train information, such as train name, train code, train schedule, train route, train fare, etc.

Interface Requirements

User Interface

- UI should be simple, intuitive, responsive and consistent
- Interface should support different languages and devices as needed

- UI should display availability and location of books in the library

Hardware Interface:

- The system should be compatible with multiple operating systems such as Windows, Linux etc
- The system should be reliable and secure when communicating with hardware devices
- The system should use QR codes to scan passports, and to facilitate payments

Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes

Performance Requirements

The number of pages should be minimized for the user's convenience

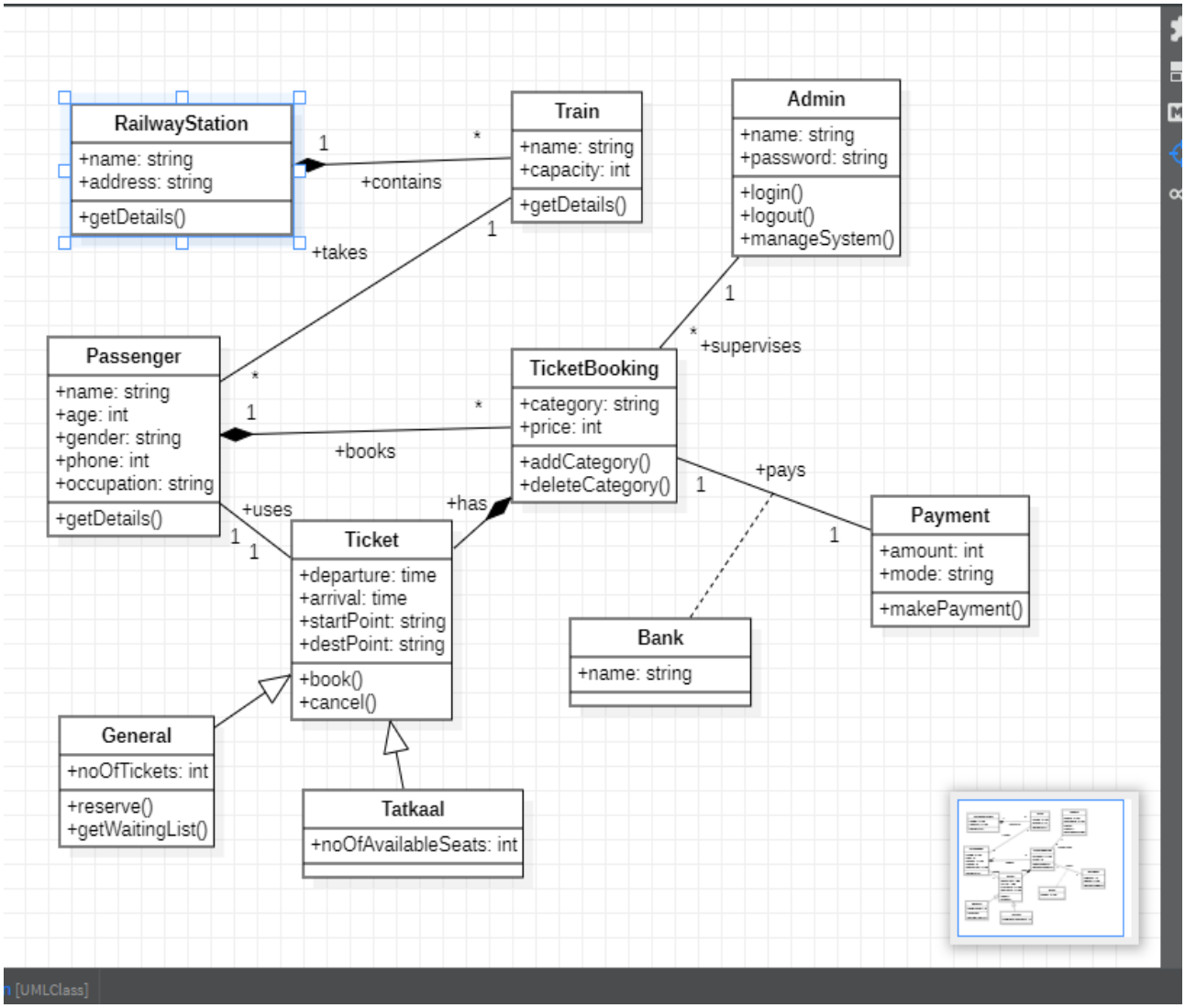
Verifying details should be a very swift process

There should be a procedure for when there is loss of data due to failure of storage device or similar reason.

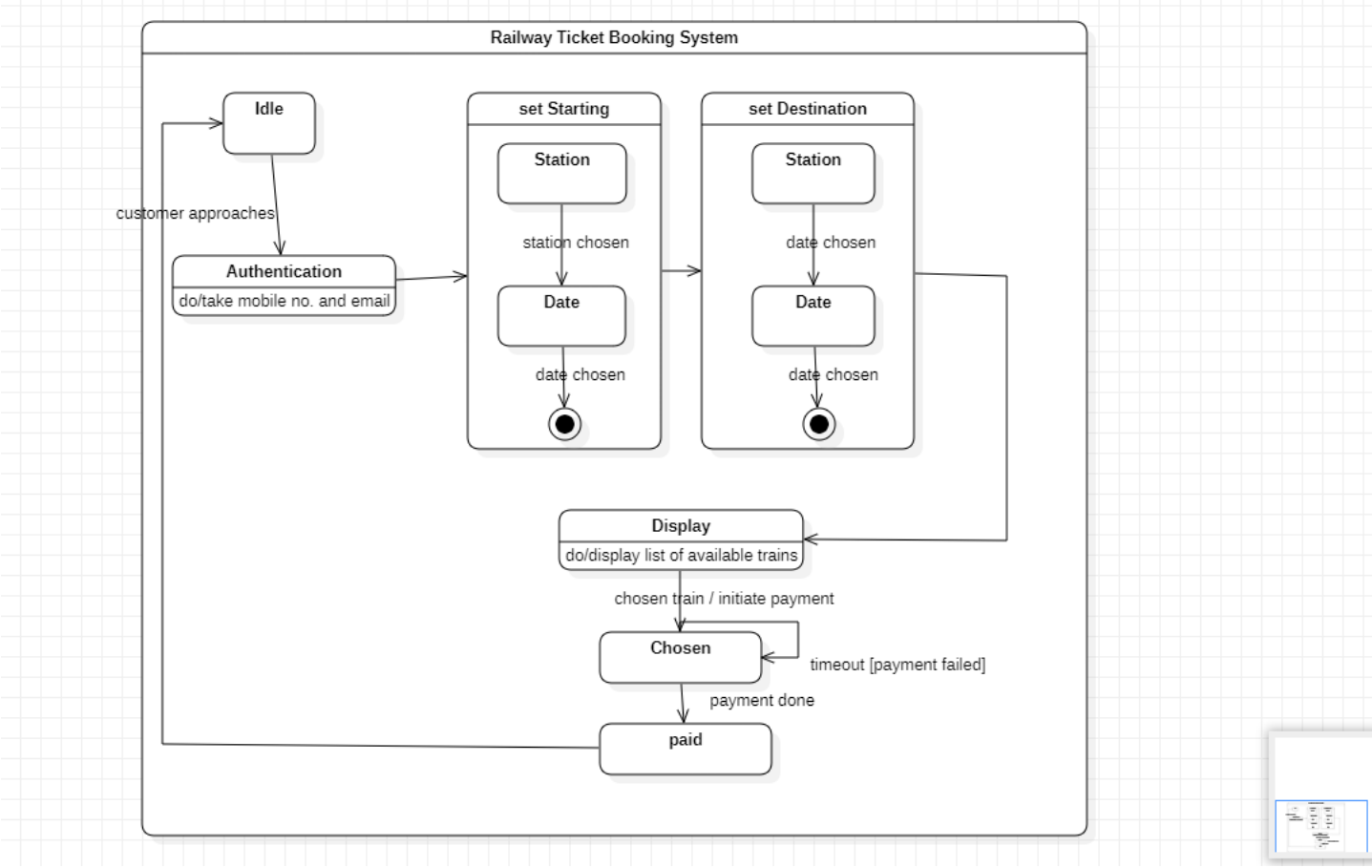
Design Constraints

- Working with the constraints set by the government
- Respect industry standards and follow best practices of software development

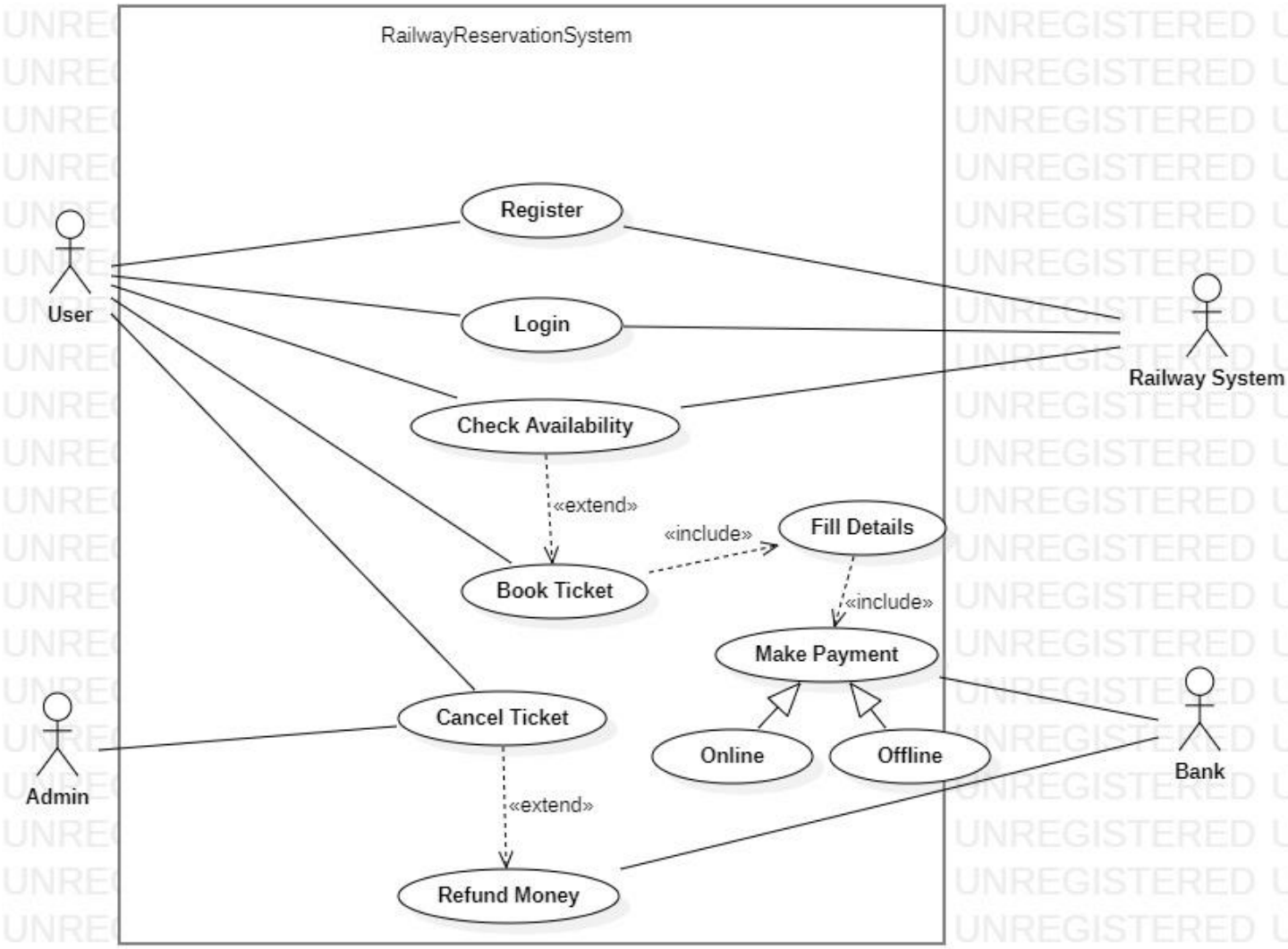
6.3 Class Diagram



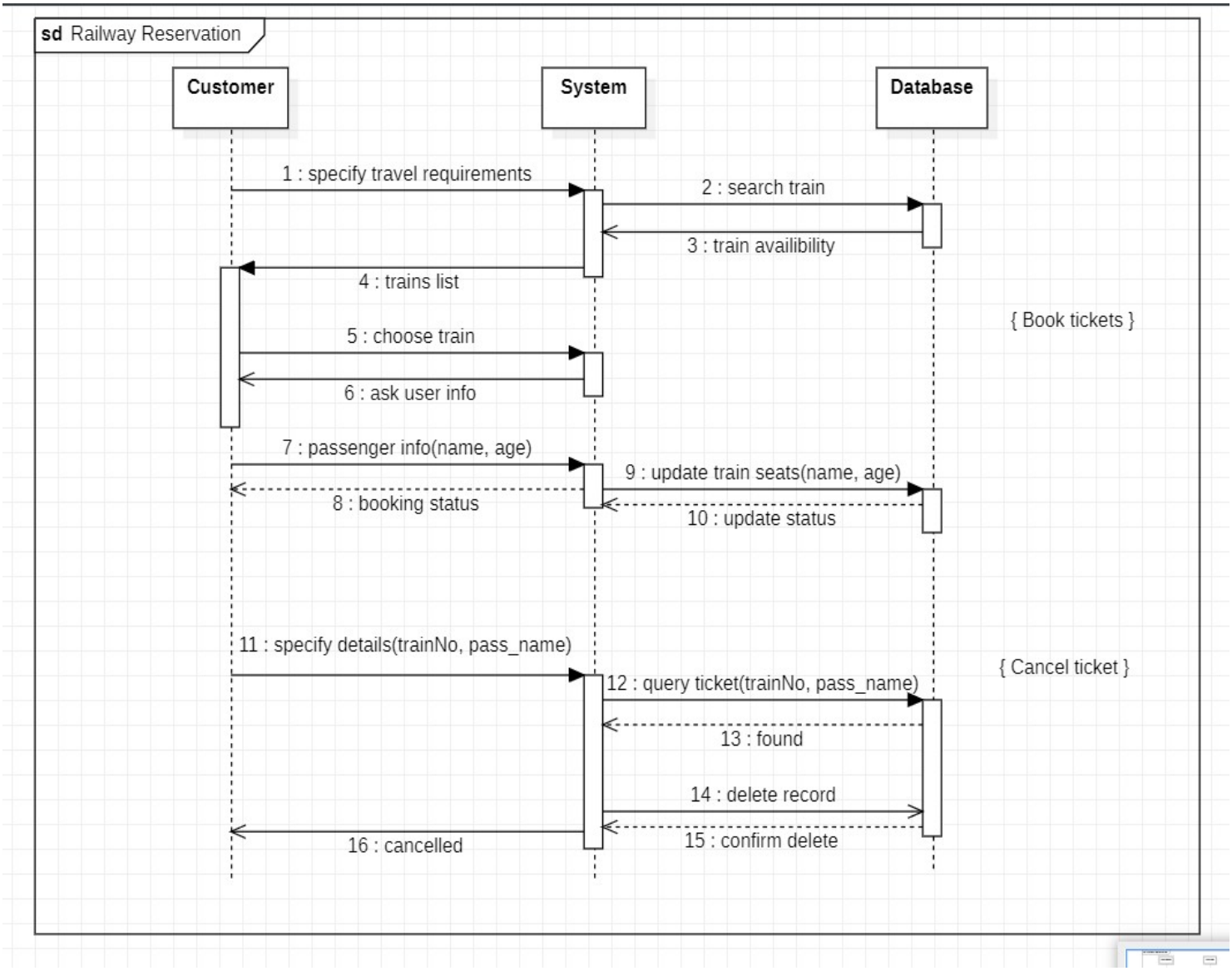
6.4 State Diagrams



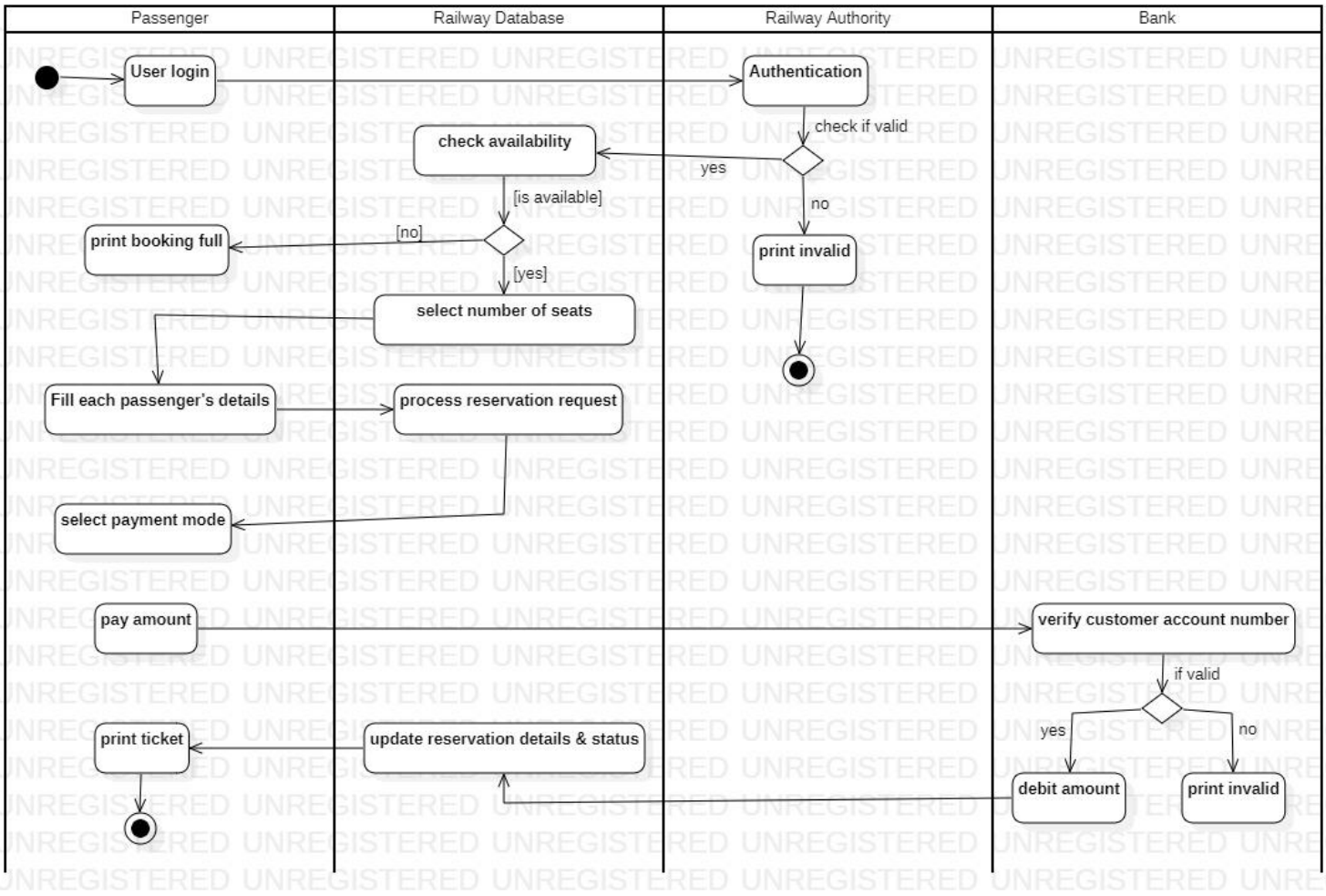
6.5 Use Case Diagram



6.6 Sequence Diagram



6.7 Activity Diagram



7. STOCK MANAGEMENT SYSTEM

7.1 Problem statement

The stock maintenance system must take care of sales information of the company and must analyze the potential of the trade. It maintains the number of items that are added or removed.

7.2 Software Requirement Specification

Introduction

Purpose:

The purpose of this system is to maintain stocks in an automated way.

Scope of this document

The core of the system is to get the online registration form (with details such as name, address etc.,) filled by the applicant whose testament is verified for its genuineness by the Stock Market Prediction with respect to the already existing information in the database. It adopts a comprehensive approach to minimize the manual work and schedule resources, time in a cogent manner.

Overview

General Description

A stock market management system is a software that helps manage the entire details of the stock market in an automated way. It can help maintain an optimal stock level, track goods during transport between locations, receive new items, manage warehouse processes such as picking, packing, and shipping, prevent product obsolescence and spoilage, and ensure your products are never out of stock.

Functional Requirements:

Passport Information: The system should provide an online interface to the user where they can fill in their personal details and submit the necessary documents (may be by scanning).

The system should allow the user to apply for a new passport, renew an existing passport, or change any details in their passport.

The system should allow the user to pay the required fees for their passport application through online mode or offline mode.

The system should authenticate the user by using their username and password or by scanning their barcode.

Interface Requirements

User Interface

- UI should be simple, intuitive, responsive and consistent
- Interface should support different languages and devices as needed
- UI should display availability and location of books in the library

Hardware Interface:

- The system should be compatible with multiple operating systems such as Windows, Linux etc
- The system should be reliable and secure when communicating with hardware devices
- The system should use QR codes to scan passports, and to facilitate payments

Software Interface:

- A working OS
- Front-end framework like React/Angular/Vue
- RDBMS like MySQL, PostgreSQL etc
- Containers and orchestration services like Kubernetes

Performance Requirements

- The number of pages should be minimized for the user's convenience
- The system should work 24/7
- There should be a procedure for when there is loss of data due to failure of storage device or similar reason.
- The system should update every few seconds

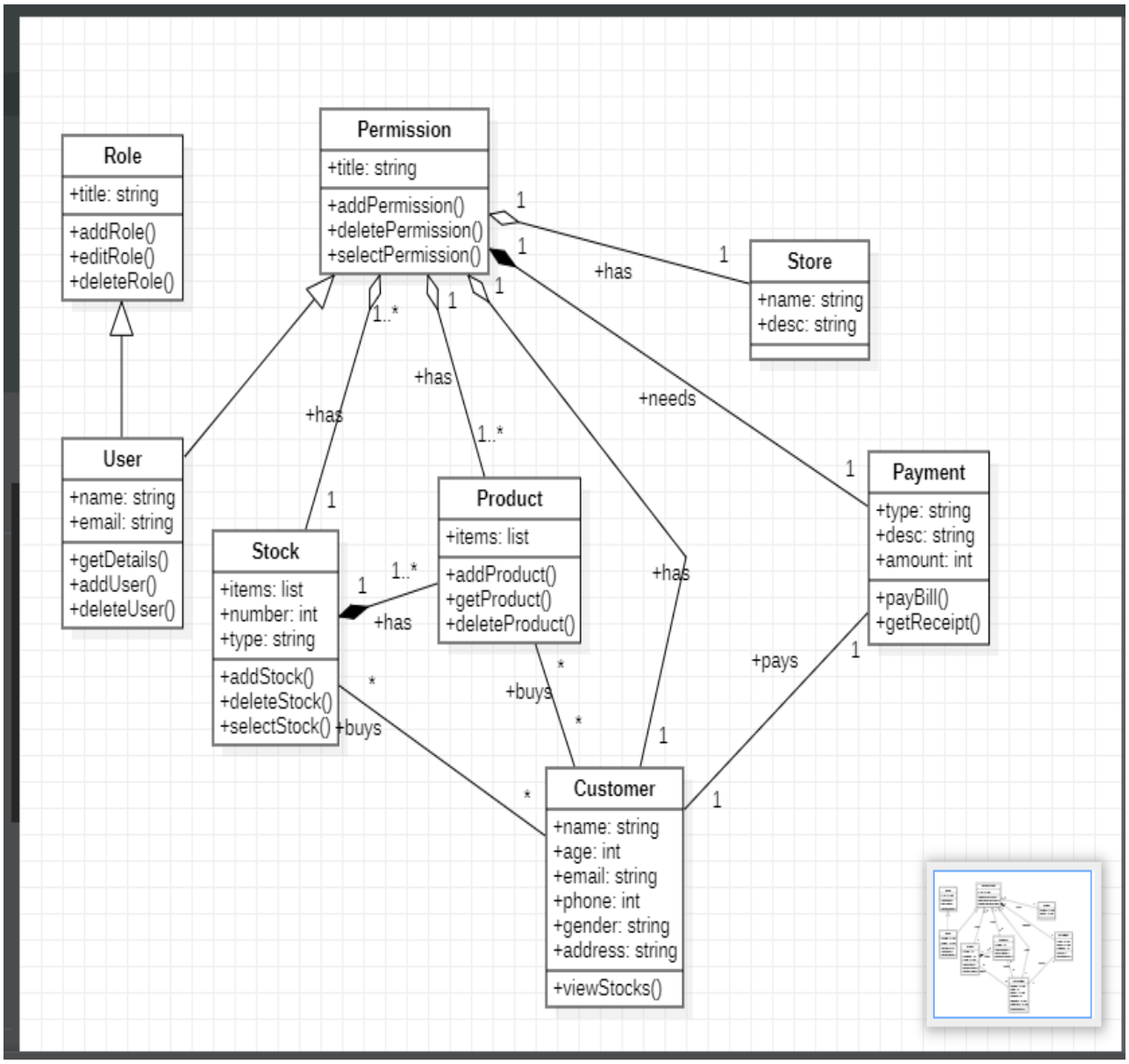
Design Constraints

- Working with the constraints set by the government
- Respect industry standards and follow best practices of software development

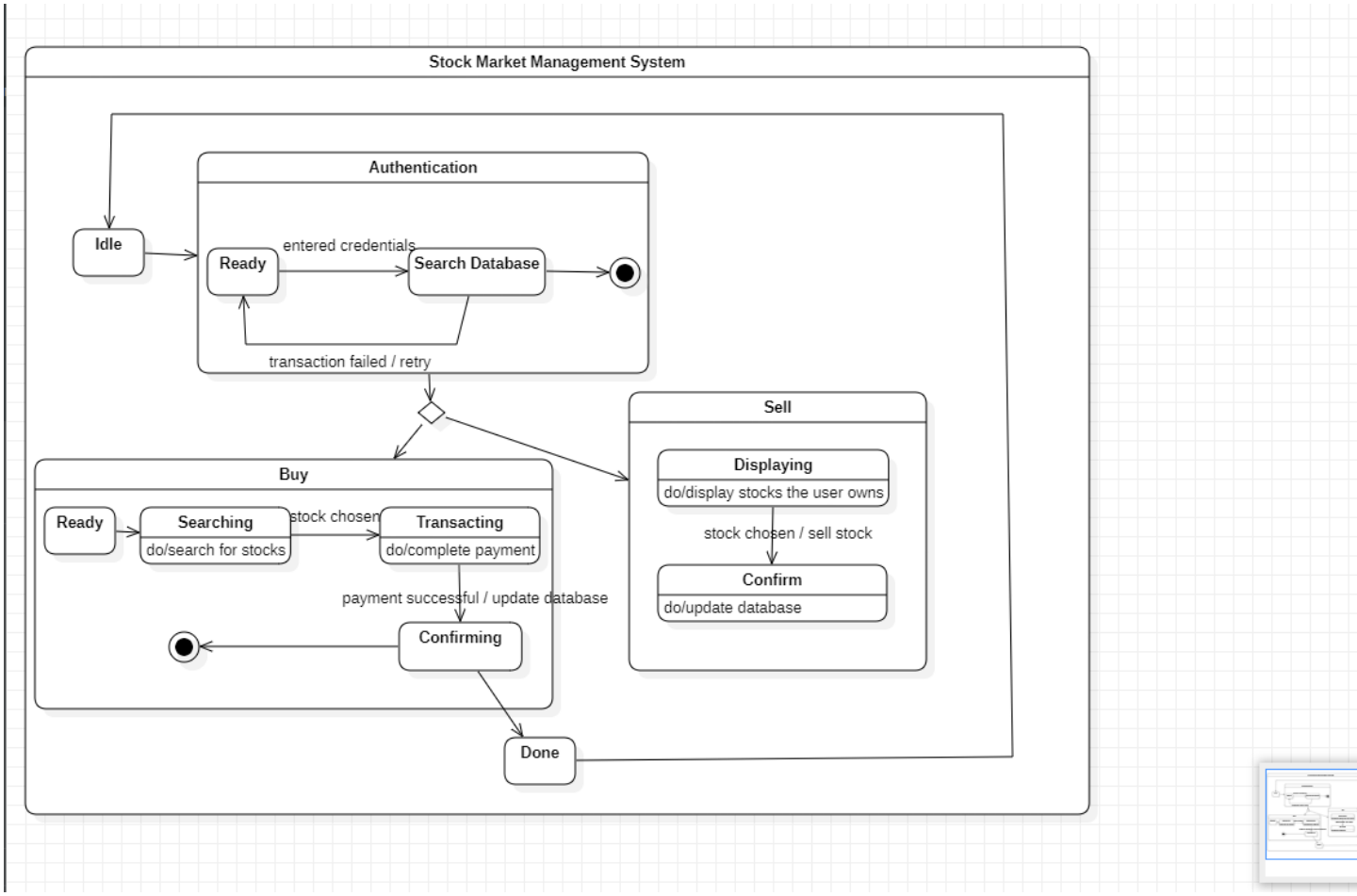
Non-Functional Requirements

- Performance: The system should be fast and accurate. It should handle expected and unexpected errors. It should be able to handle large amounts of data.
- Security: The system should protect the data from unauthorized access and modification. It should use encryption and authentication techniques to ensure data security.
- Scalability: The system should be able to handle an increasing number of users and applications without degrading the performance or functionality.

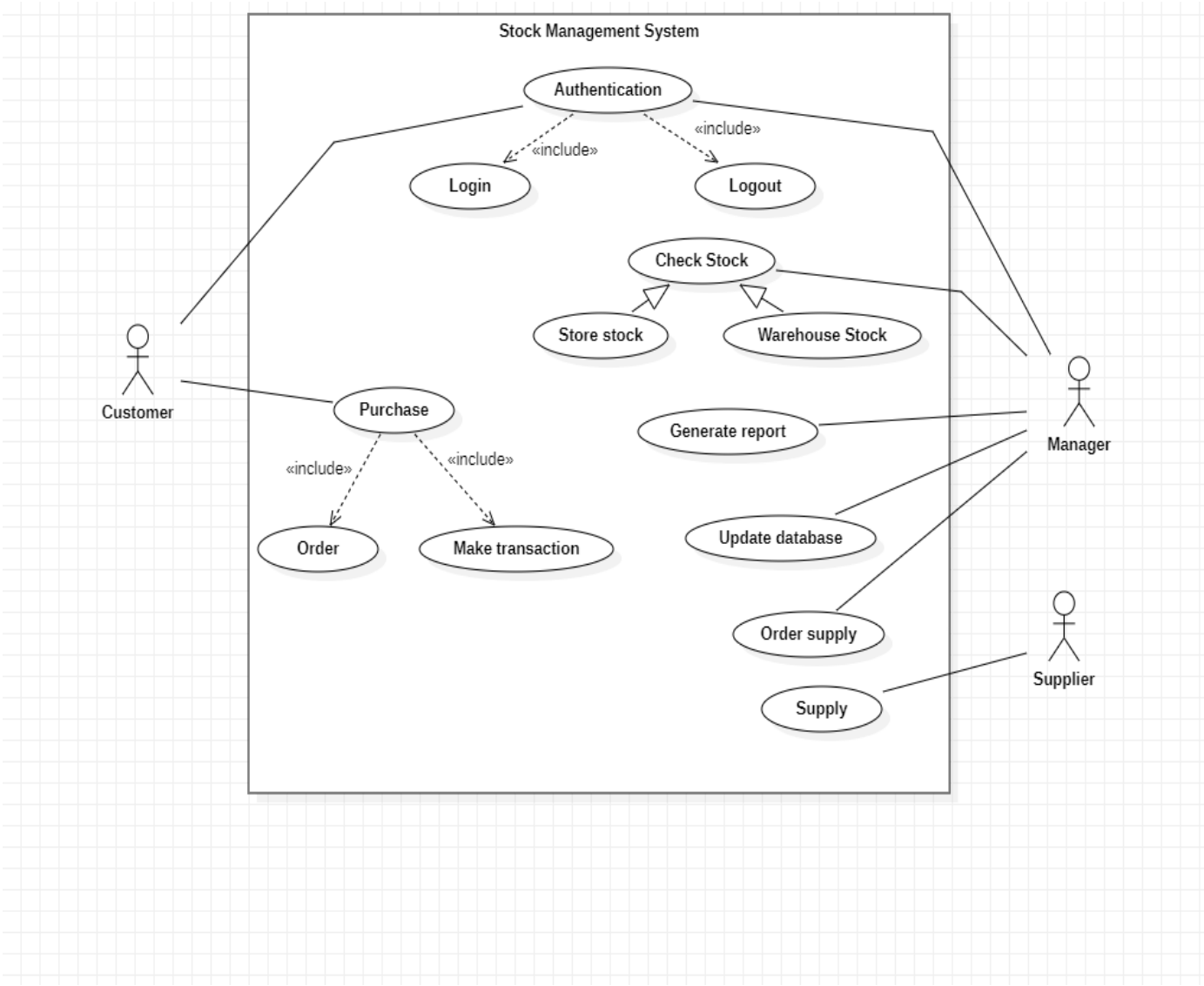
7.3 Class Diagram



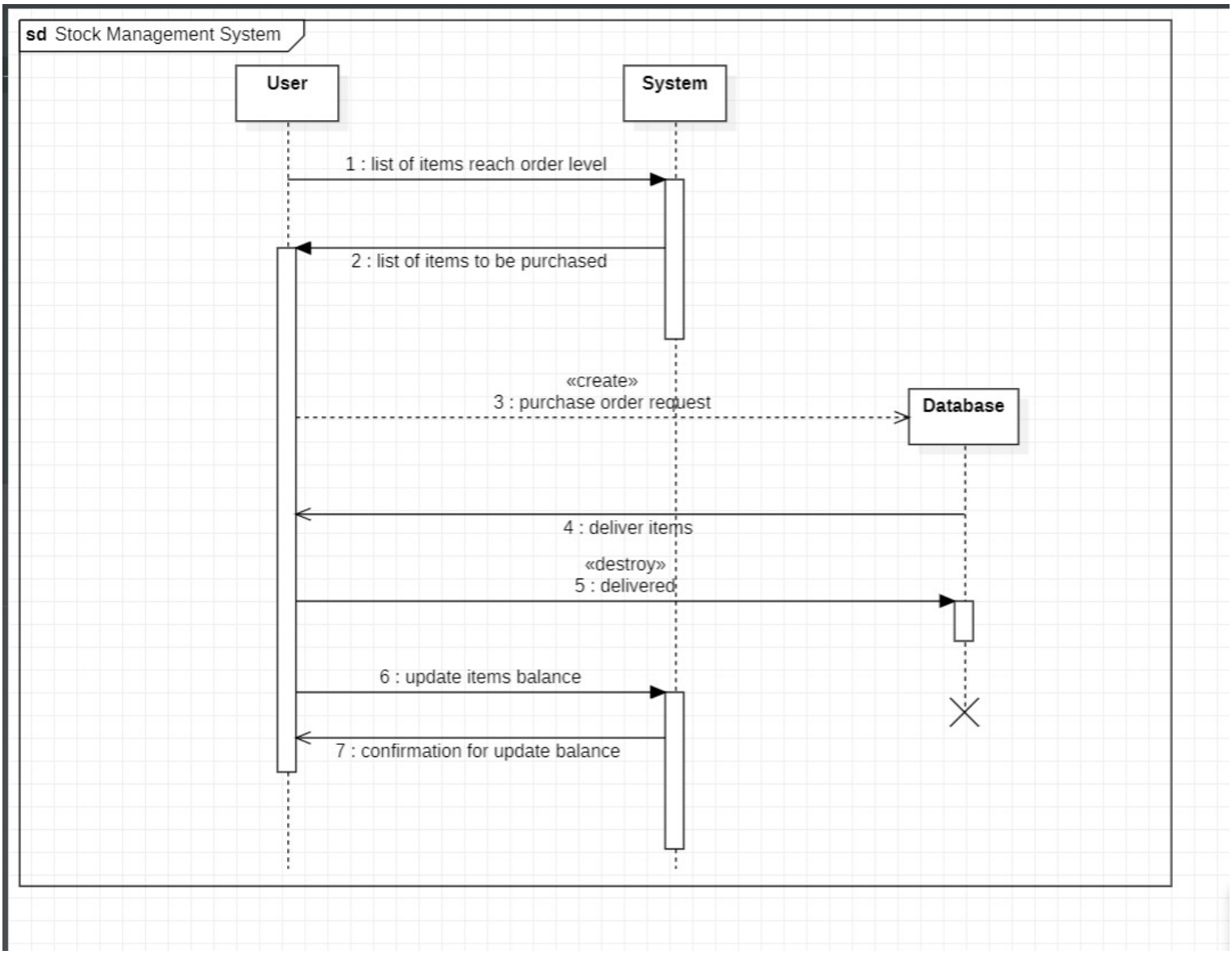
7.4 State Diagrams



7.5 Use Case Diagram



7.6 Sequence Diagram



7.7 Activity Diagram