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

TOSHIN FELIX I
1BM20CS173

Department of Computer Science and Engineering

B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
April-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 **Toshin Felix I(1BM20CS173)** during the 6th Semester April-July 2023.

Signature

Dr. Seema Patil

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

TABLE OF CONTENTS

1.Credit Card Management System

1.1 Problem Statement

The current credit card management system is plagued by inefficiencies and limitations. It lacks automation, resulting in delays and errors in the card application process. Fraud detection capabilities are inadequate, leaving cardholders vulnerable to unauthorized transactions. Account management features are limited, making it difficult for users to monitor transactions and manage their credit card accounts effectively. Payment processes are inconvenient and lack options, leading to delays and difficulties in reconciliation. Integration with other financial systems is insufficient, causing data inconsistencies and manual redundancies. Customer support is ineffective, hampering prompt issue resolution and dispute resolution.

1.2 Software Requirements Specification

Overview – Credit card processing through offline involves the merchant collecting order information, storing this in a database on your site, and entering it using their on-site merchant credit card processing system. Takes time to manually enter credit card information for each order. This solution creates following cons: · Insecure – there is a possibility that a skilled hacker could break into the database and steal an entire list of credit card numbers, thereby damaging the merchant's reputation with current client. ·

General description - The system typically includes hardware and software components that work together to facilitate transactions. The hardware may include a point-of-sale (POS) terminal, card reader, or mobile device, while the software may include a payment gateway, merchant account, and fraud detection and prevention tools. The credit card processing system works by securely transmitting customer payment information to the card issuer or the card network, which then approves or declines the transaction based on factors such as available credit, fraud risk, and other security checks. Once approved, the payment is settled and funds are transferred from the customer's account to the merchant's account.

Functional Requirements

● Allows merchants to accept credit and debit card payments from customers. This includes authorizing transactions, verifying cardholder information, and settling payments.

● Integrated with a payment gateway that securely transmits payment information between the merchant and the card issuer.

● Has measures in place to detect and prevent fraudulent transactions, including address verification, card verification, and real-time fraud screening.

● Allows merchants to manage their accounts, view transaction histories, and generate reports on payment activity.

● The system allows merchants to issue refunds and handle chargebacks in a timely and efficient manner.

● Supports mobile payments, enabling customers to pay using their mobile devices.

Interface Requirements

● Has a user-friendly interface that makes it easy for merchants to access and manage their accounts, view transaction history, and generate reports.

● Provides an API (Application Programming Interface) that allows merchants to integrate the payment processing functionality into their own software applications.

● Integrated with a payment gateway that securely transmits payment information between the merchant and the card issuer.

● Supports mobile payments, enabling customers to pay using their mobile devices.

● The system should support transactions in multiple currencies and provide currency conversion functionality.

● Provides a reporting interface that allows merchants to generate reports on transaction activity, settlement, and reconciliation.

● Provides a security interface that enables merchants to configure security settings and monitor security events.

Performance Requirements

● The system should respond quickly to user requests and actions, with minimal latency and delay.

● The system should be scalable, capable of handling a large number of users and transactions, without compromising performance or reliability.

● The system should be available 24/7, with minimal downtime for maintenance or upgrades.

● The system should be reliable, with minimal errors or failures, and capable of recovering quickly from any disruptions.

● The system should be secure, protecting user data and transactions from unauthorized access or breaches.

● The system should undergo regular load testing, to ensure that it can handle peak loads and heavy traffic without compromising performance or availability.

● The system should be optimized for performance, with efficient algorithms, data structures, and processing techniques, to minimize resource usage and improve response time.

Non-Functional Requirements

● The system should be user-friendly, easy to learn, and intuitive, with a well-designed user interface that enables users to perform tasks quickly and efficiently.

● The system should be reliable, with a low error rate and minimal downtime, ensuring that users can access and use the system at all times.

● The system should be secure, protecting user data and transactions from unauthorized access or breaches, and complying with regulatory requirements.

● The system should be fast and responsive, with minimal latency and delay, enabling users to perform tasks quickly and efficiently.

● The system should be scalable, able to handle a large number of users and transactions, without compromising performance or reliability.

● The system should be easy to maintain, with well-documented code, clear error messages, and easy-to-use debugging tools, enabling developers to identify and fix issues quickly.

1.3 Class Diagram

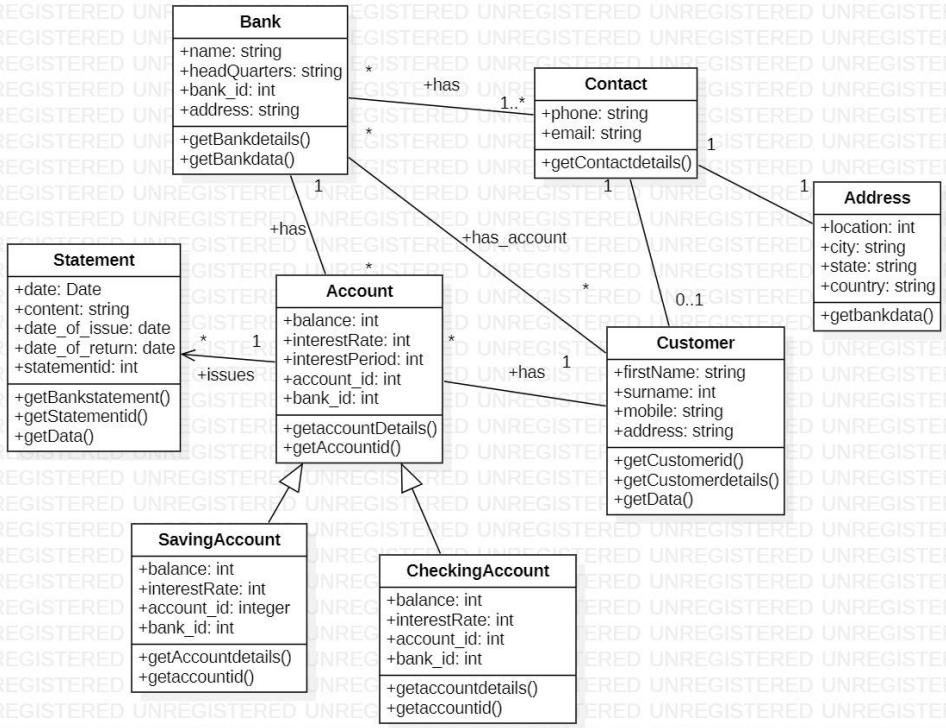

Fig 1.1 - Class Diagram for Credit Card Processing

1.4 State Diagram

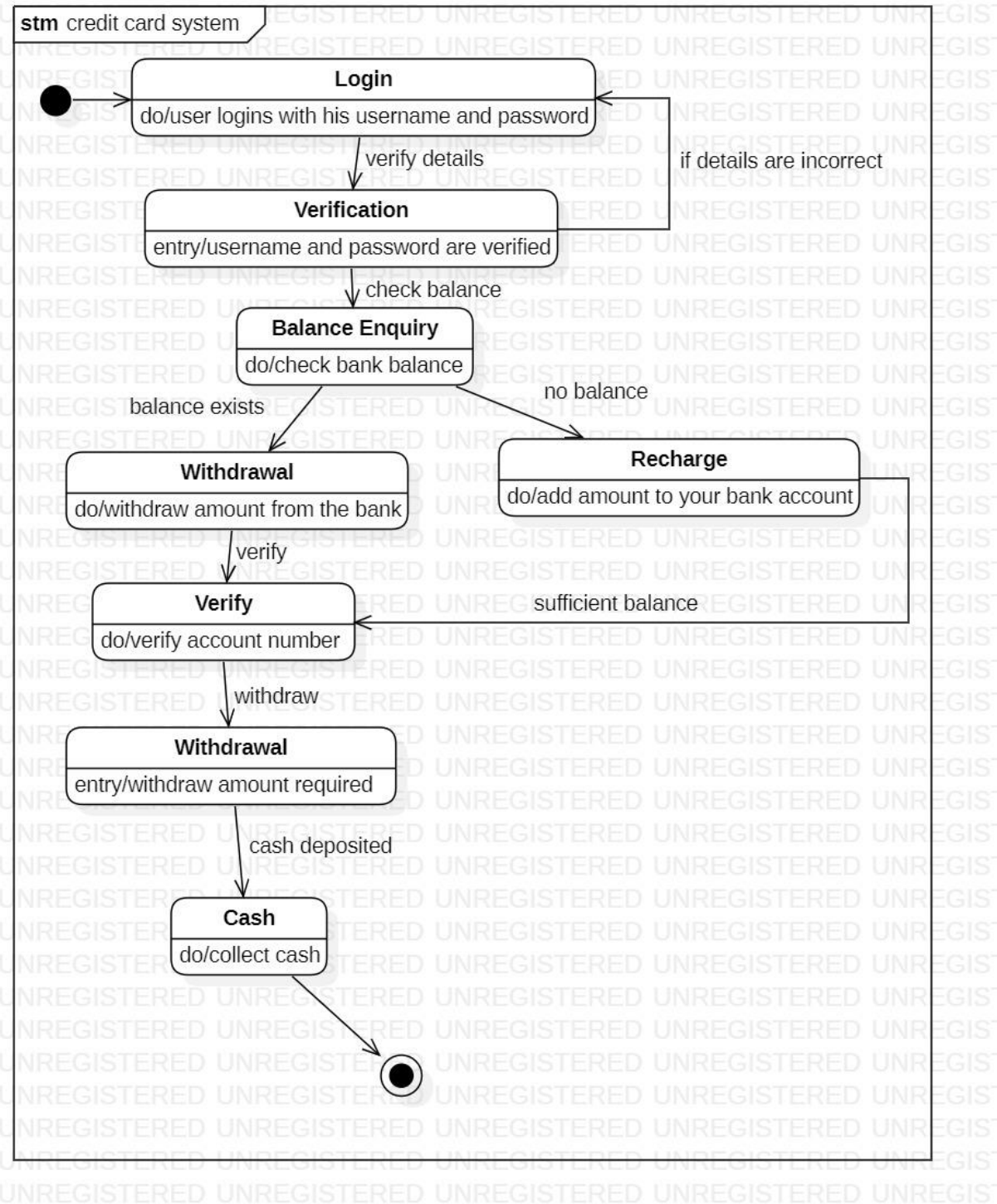

Fig 1.2 – State Diagram for Credit Card Processing

1.5 Interaction Diagrams

1.5.1 Use Case Diagram

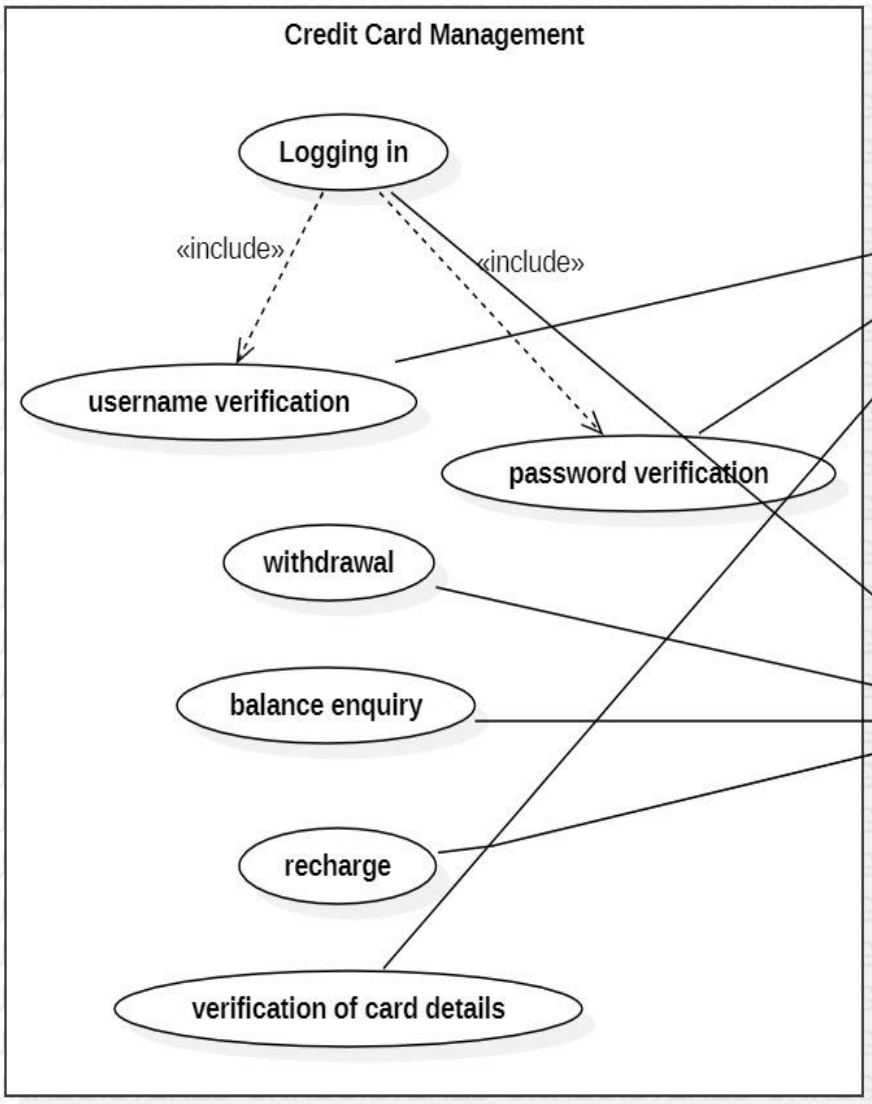

Fig 1.3 – Use Case Diagram for Credit Card Processing

1.5.2 Sequence Diagram

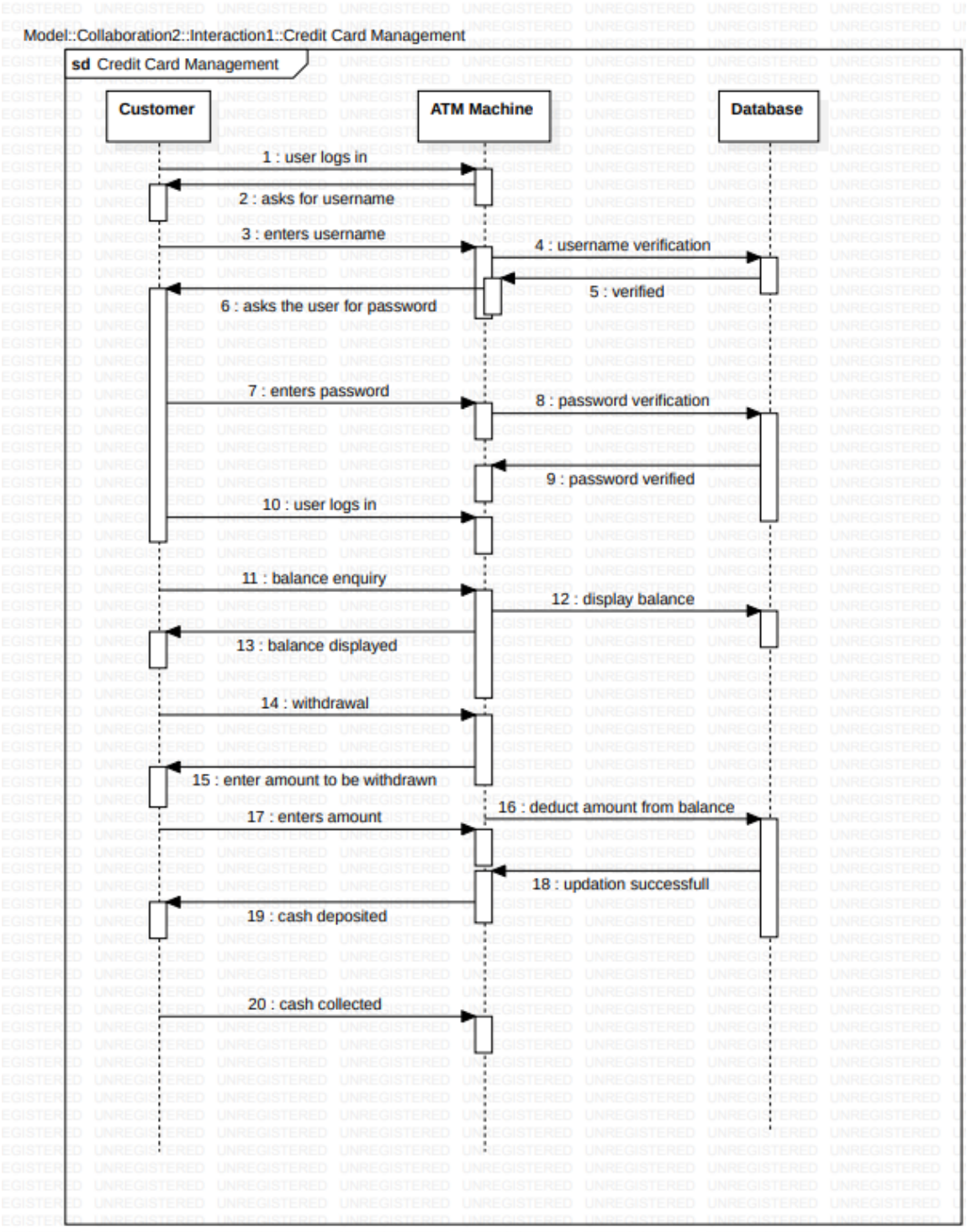

Fig 1.4 – Sequence Diagram for Credit Card Processing

1.5.3 Activity Diagram

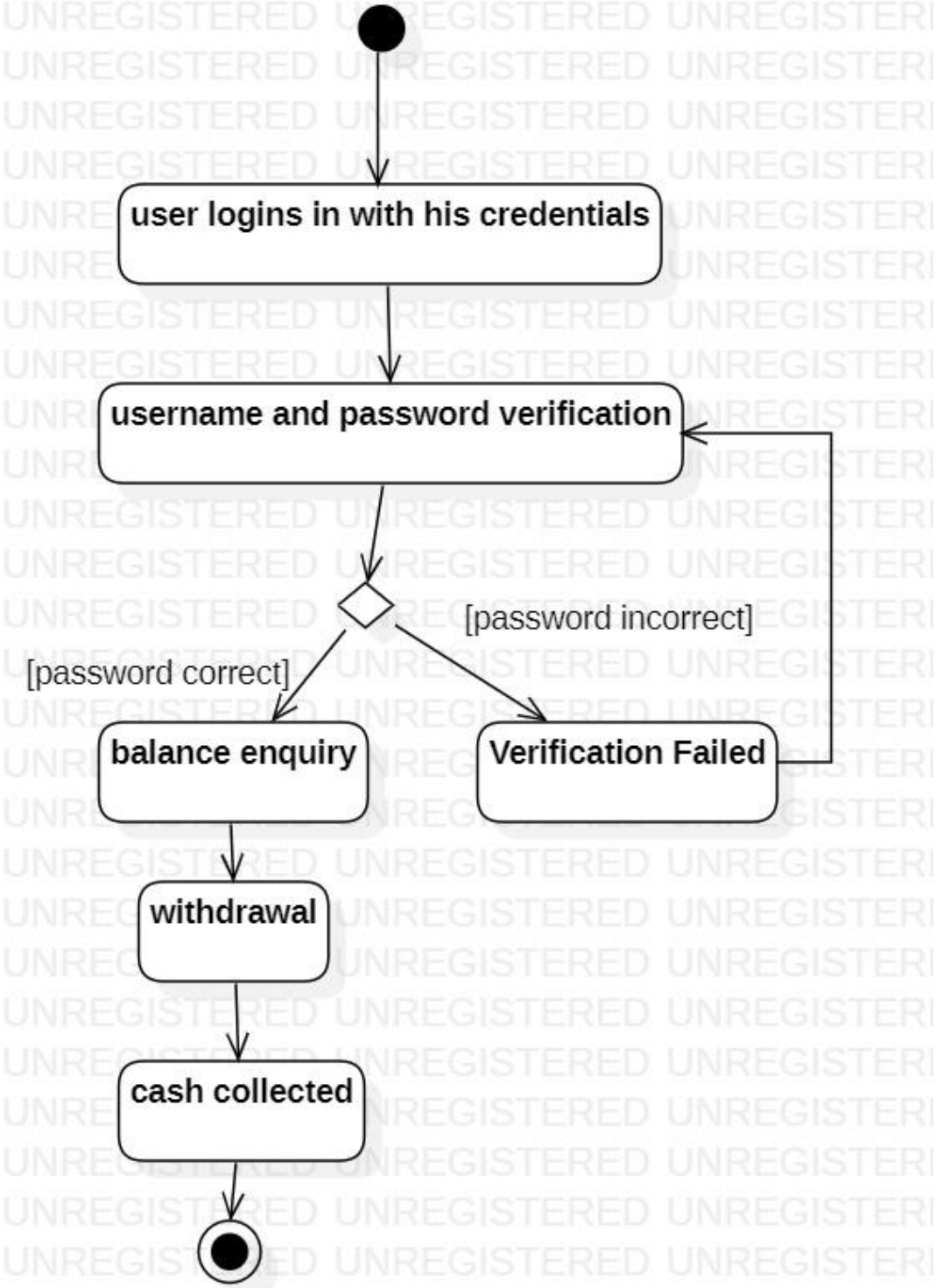

Fig 1.5 – Activity Diagram for Credit Card Processing

2. Hotel Management System

2.1 Problem statement

The existing hotel management system is plagued by various challenges that hinder its ability to efficiently handle hotel operations and provide a seamless guest experience. To address these challenges, a new hotel management system is needed that automates and streamlines processes, provides comprehensive room and rate management functionalities, enhances guest check-in/check-out experiences, improves housekeeping and maintenance management efficiency, enables seamless communication and collaboration between departments, and offers robust reporting and analytics capabilities. Implementing such a system would enhance operational efficiency, elevate guest satisfaction, and drive revenue growth for the hotel.

2.2 Software Requirements Specification

General description: The Hotel Management System is a software application designed to automate the operations of a hotel. It provides an efficient and user-friendly interface for hotel staff to manage various functions such as reservations, check-ins, check-outs, room assignments, guest profiles, billing, and reporting. The system is accessible through desktop and mobile devices, and is integrated with other hotel systems and ensures smooth communication and coordination among different departments.

The system allows hotel staff to create, modify, and cancel reservations for guests, capturing guest information such as name, contact details, and special requests. It also enables hotel staff to manage room availability, room types, and room rates, and assign rooms to guests based on their preferences and availability. Guest profiles are maintained, storing guest information and history for personalized services and targeted marketing. The system also generates bills and invoices for guests, supports various payment methods, and provides reports and analytics on hotel performance for financial analysis and decision-making.

Functional Requirements

- The system should allow guests to reserve rooms based on availability.
- Staff should be able to view all reservations made.
- The system should allow hotel staff to manage the allocation of rooms to guests, based on availability and guest preferences.

● Allows for room maintenance, cleaning and inspection.

● Allows hotel staff to manage the check-in and check-out of guests.

● Generate invoices for hotel items and services used

● Generate reports on occupancy, customer preferences.

Interface Requirements

● The interface should have a modern and visually appealing design.

● The interface should be easy to navigate.

● The interface should be responsive and adapt to different screen sizes.

● The interface should allow for customization, such as the ability to change colors, logos, and fonts.

● The interface should support multiple languages.

● The interface should allow for different user roles and permissions, with restricted access to sensitive information.

● The interface should provide alerts and notifications to keep staff informed of important events, such as room availability or upcoming reservations.

● The interface should allow for easy search and filtering of information.

● The interface should be able to integrate with other systems used by the hotel, such as payment gateways or property management systems.

Performance Requirements

In this, how a software system performs desired functions under specific condition is explained. It also explains required time, required memory, maximum error rate, etc.

2.3 Class Diagram

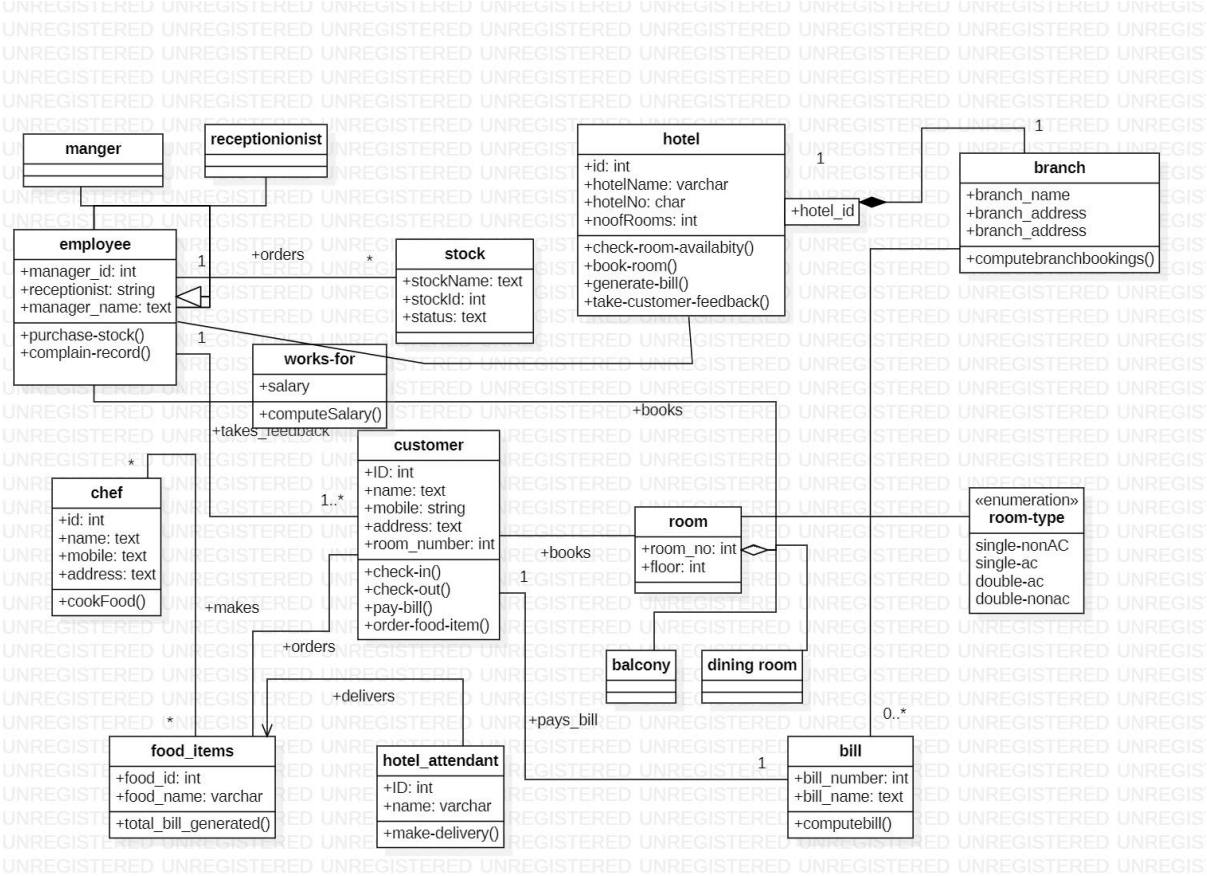

Fig 2.1 – Class Diagram for Hotel Management System

2.4 State Diagram

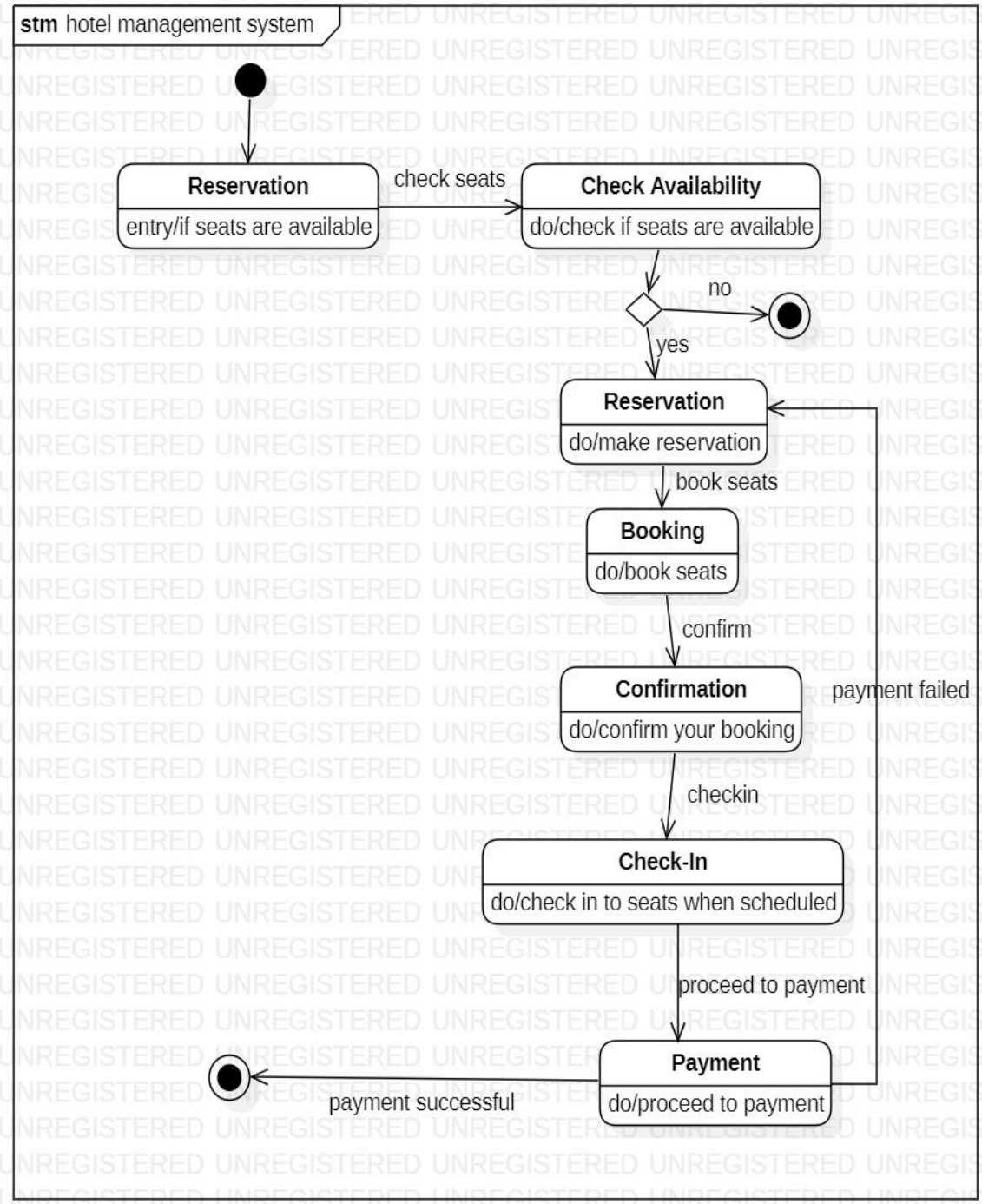

Fig 2.2 – State Diagram for hotel management system

2.5 Interaction Diagrams

2.5.1 Use Case Diagram

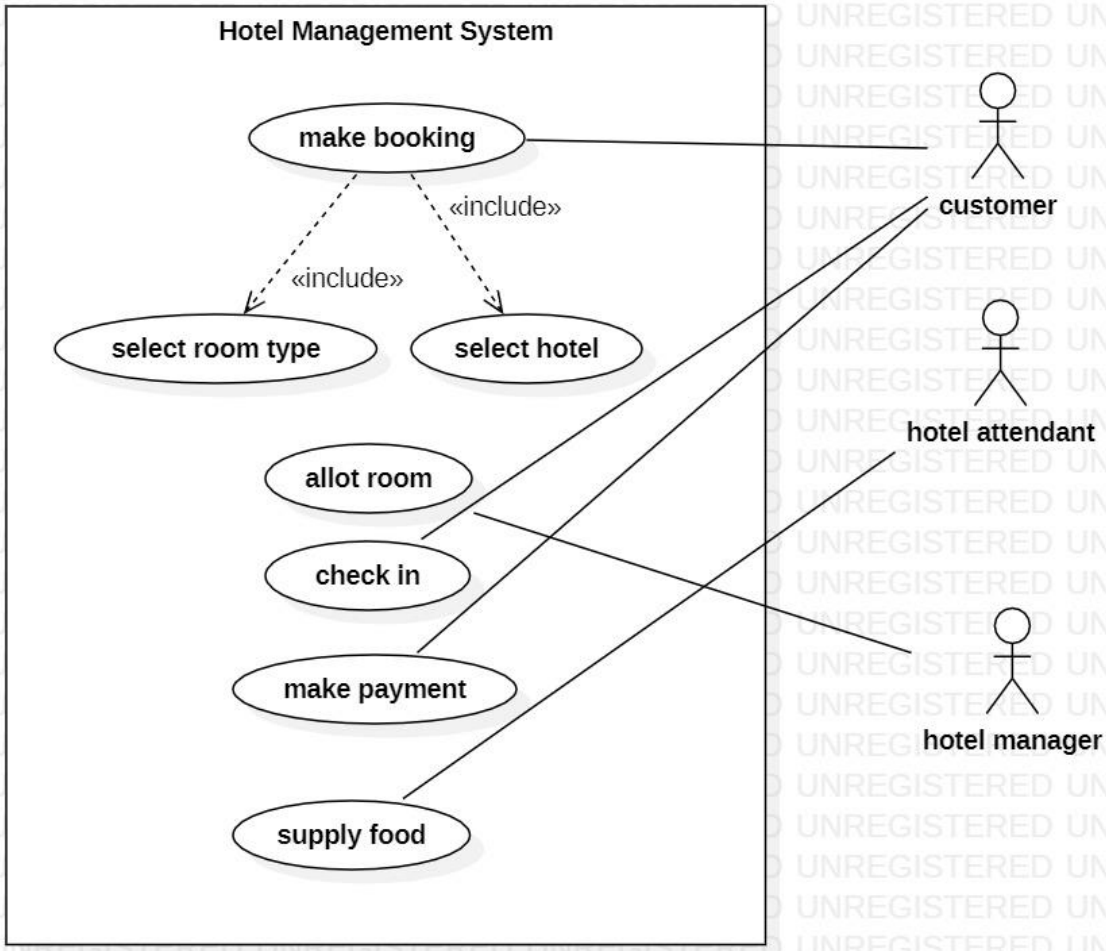

Fig 2.3 – Use Case diagram for Hotel Management

2.5.2 Sequence Diagram

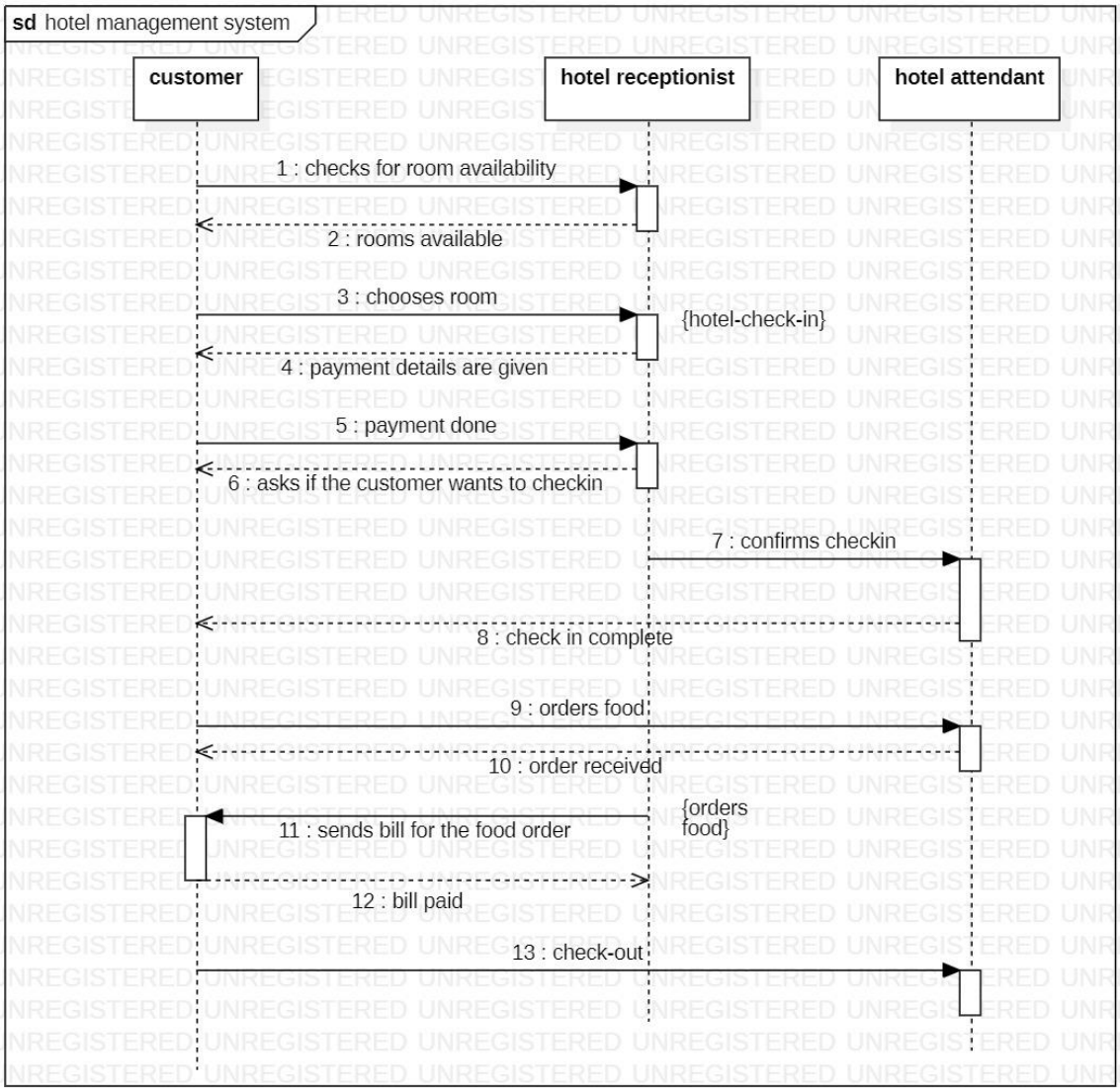

Fig 2.4 – Sequence Diagram for hotel management system

2.5.3 Activity Diagram

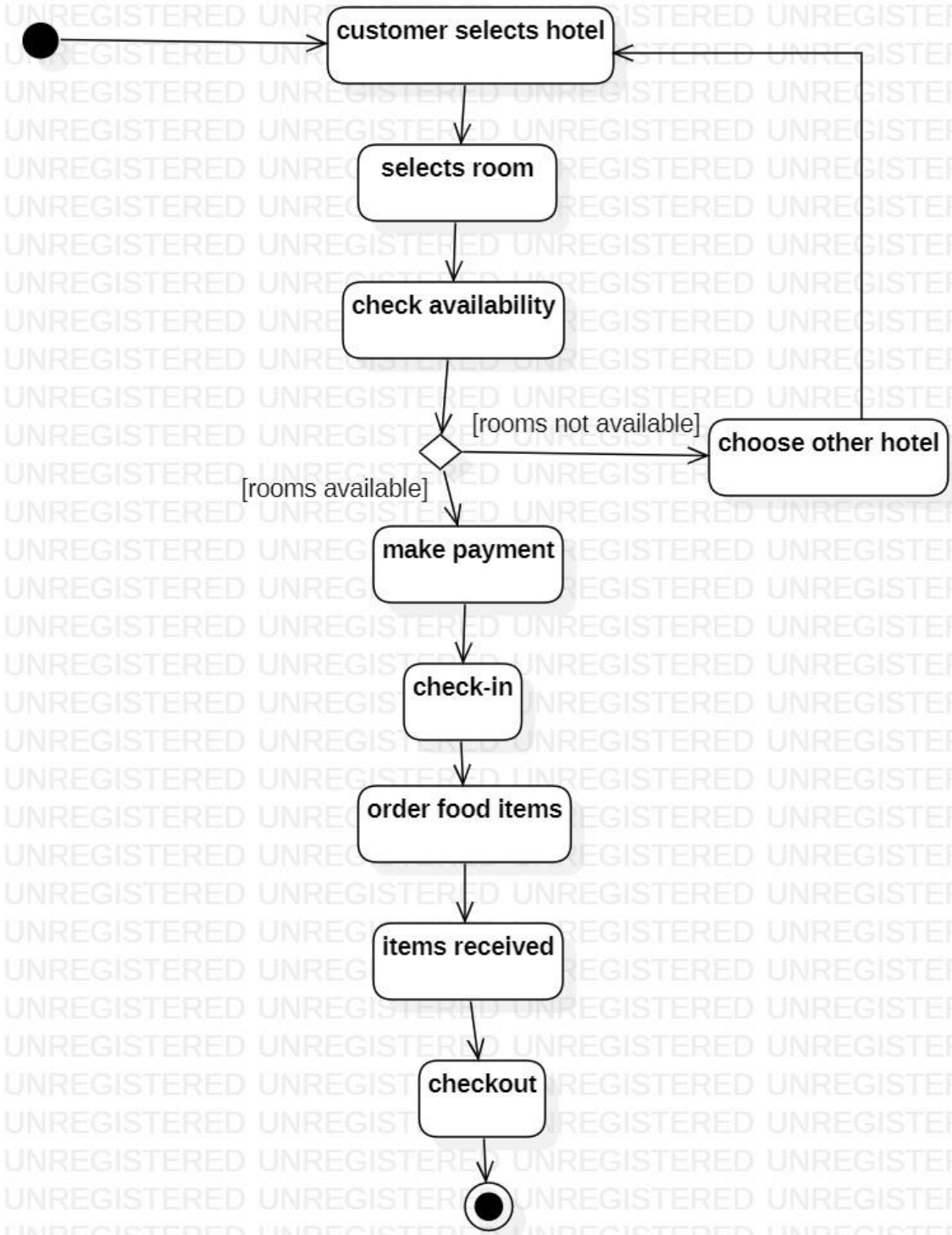

Fig 2.5 – Activity Diagram for hotel management system

3.Library Management System

3.1 Problem Statement

The current library management system faces several challenges that hinder efficient library operations and the delivery of exceptional library services. To address these challenges, a new and improved library management system is required. The system should automate and streamline cataloging processes, provide comprehensive user management functionalities, enable efficient reservation and queue management, offer robust inventory management and stock control features, provide enhanced reporting and analytics capabilities, and facilitate seamless integration with digital platforms. Implementing such a system would enhance operational efficiency, improve user satisfaction, and enable the library to adapt to the evolving needs of its users in the digital age.

3.2 Software Requirements Specification

Overview – A library management system is a software application designed to automate and manage the tasks and operations of a library. The main purpose of a library management system is to help librarians manage library resources more efficiently and effectively. The system typically includes modules for cataloging, circulation, acquisitions, and administration. The cataloging module is used to manage the library's collection, including books, periodicals, media, and other materials. It allows librarians to add, modify, and delete items in the collection and track their availability and location. The circulation module handles the loaning and returning of library materials. It tracks the status of borrowed items, manages fines and fees, and generates reports on circulation activity. The acquisitions module is used to manage the procurement of new materials for the library. It tracks purchase orders, invoices, and budgets, and helps librarians manage their collection development process. Overall, a library management system is a critical tool for librarians to manage the vast amount of information and resources in their libraries and provide efficient and effective services to their patrons.

General description: A library management system is a software solution that helps libraries manage their day-to-day operations. It is designed to automate many of the tasks that librarians perform manually, such as cataloging, inventory management, circulation, and patron management. The system can be used by librarians and staff to manage the library's collection, track books and other resources, and handle circulation tasks such as checking out and returning items.A typical library

management system will have several modules or components that work together to provide a comprehensive solution. These modules may include: Cataloging, Circulation, Patron management, Reporting, Digital resources.

Functional Requirements: A library management system is a software application that helps to manage the operations of a library. The functional requirements of a library management system include:

- User Management: The system should allow the librarian to create and manage user accounts. This includes registering new users, updating user information, and deleting user accounts.
- Book Management: The system should allow the librarian to manage the books in the library. This includes adding new books, updating book information, and removing books from the library.
- Cataloging: The system should provide a catalog of books in the library that can be easily searched by the users. The catalog should contain information such as the author, title, publication date, ISBN, and availability status of each book.
- Circulation: The system should manage the circulation of books in the library. This includes checking out and returning books, maintaining a record of who has borrowed each book, and managing fines for overdue books.
- Reservation: The system should allow users to reserve books that are currently checked out. The system should also notify users when reserved books become available.
- Reporting: The system should provide reports on various aspects of library operations, such as circulation statistics, overdue books, and popular books

Interface Requirements: Interface requirements of a library management system are related to the user interface design and how it interacts with the users. Some of the key interface requirements for a library management system include:

- User-friendly interface: The system should have an easy-to-use interface that can be easily navigated by users. It should have a simple and intuitive design that enables users to quickly and easily locate the information they need.
- Search functionality: The system should provide a powerful search feature that allows users to search for books by various criteria such as author, title, keyword, publication date, and subject. The search results should be presented in a clear and organized manner.
- Book details display: The system should display the detailed information about each book in a clear and concise manner. This includes the book cover image, author, title, publication date, ISBN, and availability status.
- Borrowing and returning books: The system should have a simple and easy-to-use interface for borrowing and returning books. Users should be able to check out books, renew them, and return them with minimal effort.

● Notifications: The system should provide notifications to users about the status of their library account, such as overdue books, reserved books, and pending fines.
● Accessibility: The system should be accessible to all users, including those with disabilities. It should comply with accessibility standards and provide features such as screen readers and keyboard navigation.

Performance Requirements: Performance requirements for a library management system would depend on the specific needs and objectives of the library. However, here are some general performance requirements that could be considered:
● Response time: The library management system should be able to respond quickly to user requests. This includes searching for books, checking out books, and other transactions.
● Concurrent users: The system should be able to handle a large number of users simultaneously. Libraries can have a high volume of users during peak hours, and the system should be able to handle this without slowing down.
● Scalability: The system should be scalable, allowing for future growth and expansion of the library's collection and user base. This means that the system should be able to handle increasing amounts of data and users without affecting performance.
● Reliability: The system should be reliable, with minimal downtime and errors. This is important to ensure that users can access the system when they need to and that library staff can perform necessary tasks without interruption.

Non-Functional Attributes: Non-functional requirements for a library management system might include:
● Performance: The system should be able to handle a large number of simultaneous users, and should respond quickly to user requests.
● Availability: The system should be available 24/7, with minimal downtime for maintenance and upgrades.
● Security: The system should be secure, with appropriate access controls, encryption, and authentication mechanisms to protect user data.
● Reliability: The system should be reliable, with a low probability of failure or errors.
● Scalability: The system should be able to handle growth in terms of both users and data.
● Maintainability: The system should be easy to maintain, with clear documentation and well-structured code.
● Usability: The system should be easy to use, with a user-friendly interface and clear instructions for common tasks.

3.3 Class Diagram

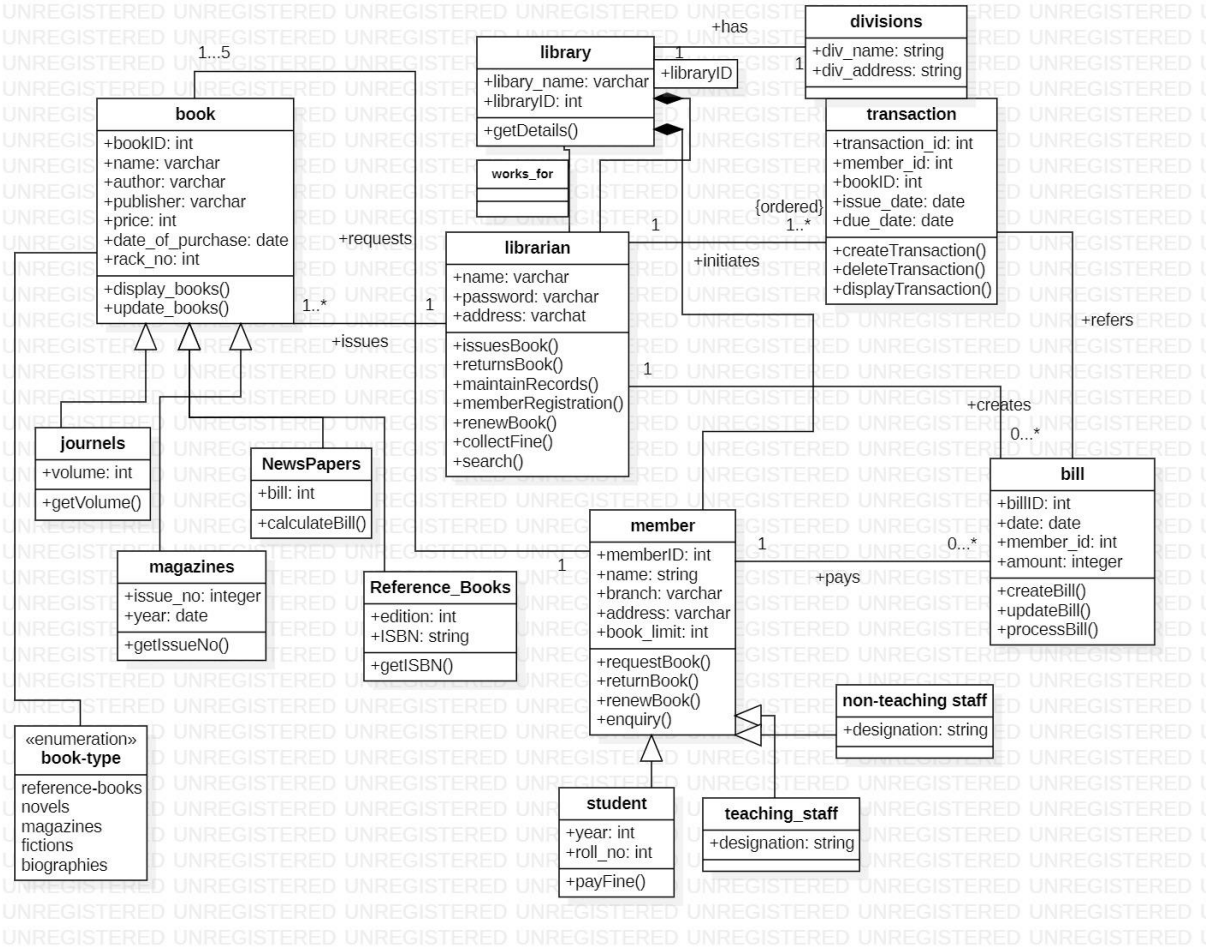

Fig 3.1 – Class Diagram for library management

3.4 State Diagram

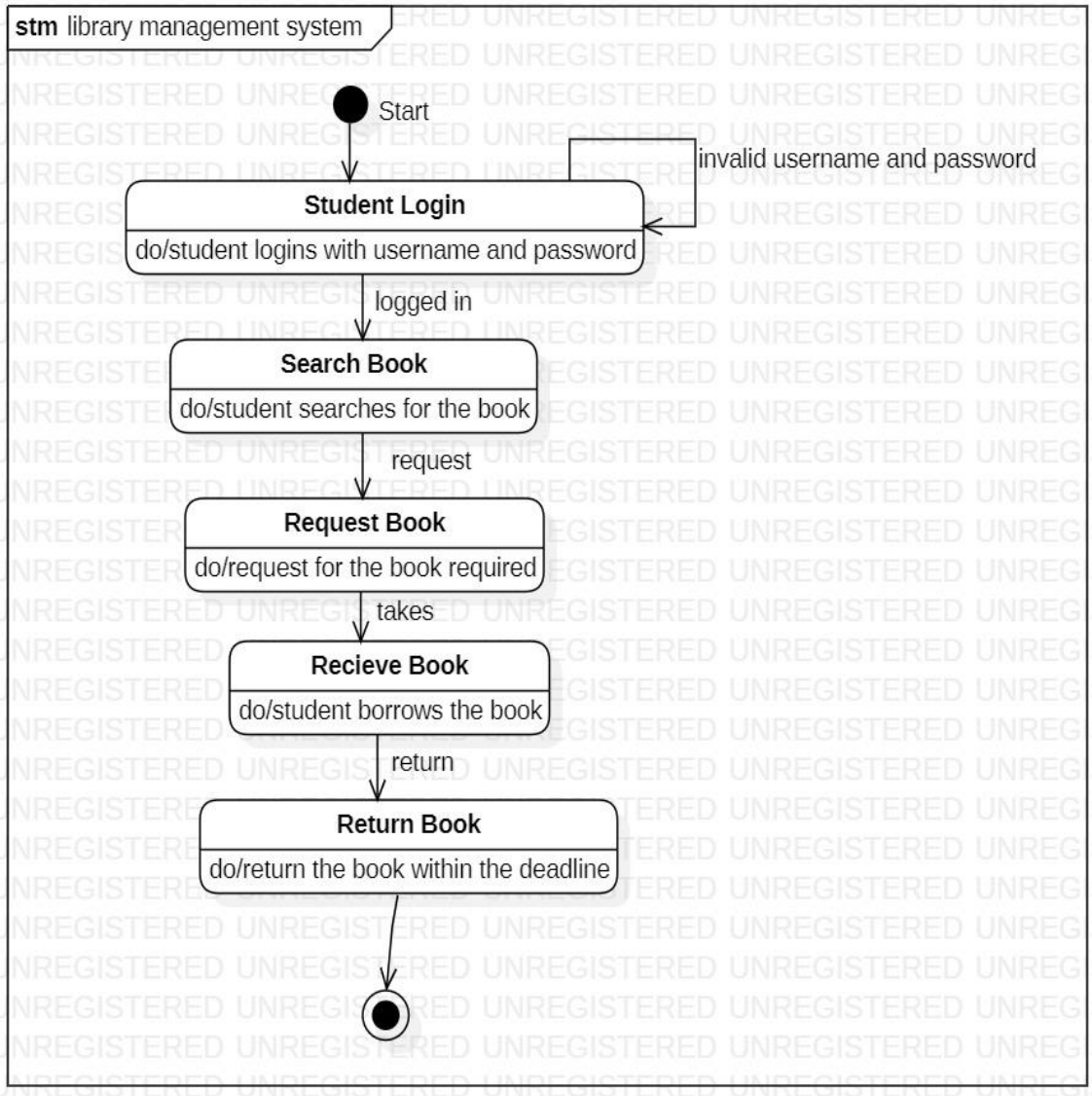

Fig 3.2 – State Diagram for library management

3.5 Interaction Diagram

3.5.1 Use Case Diagram

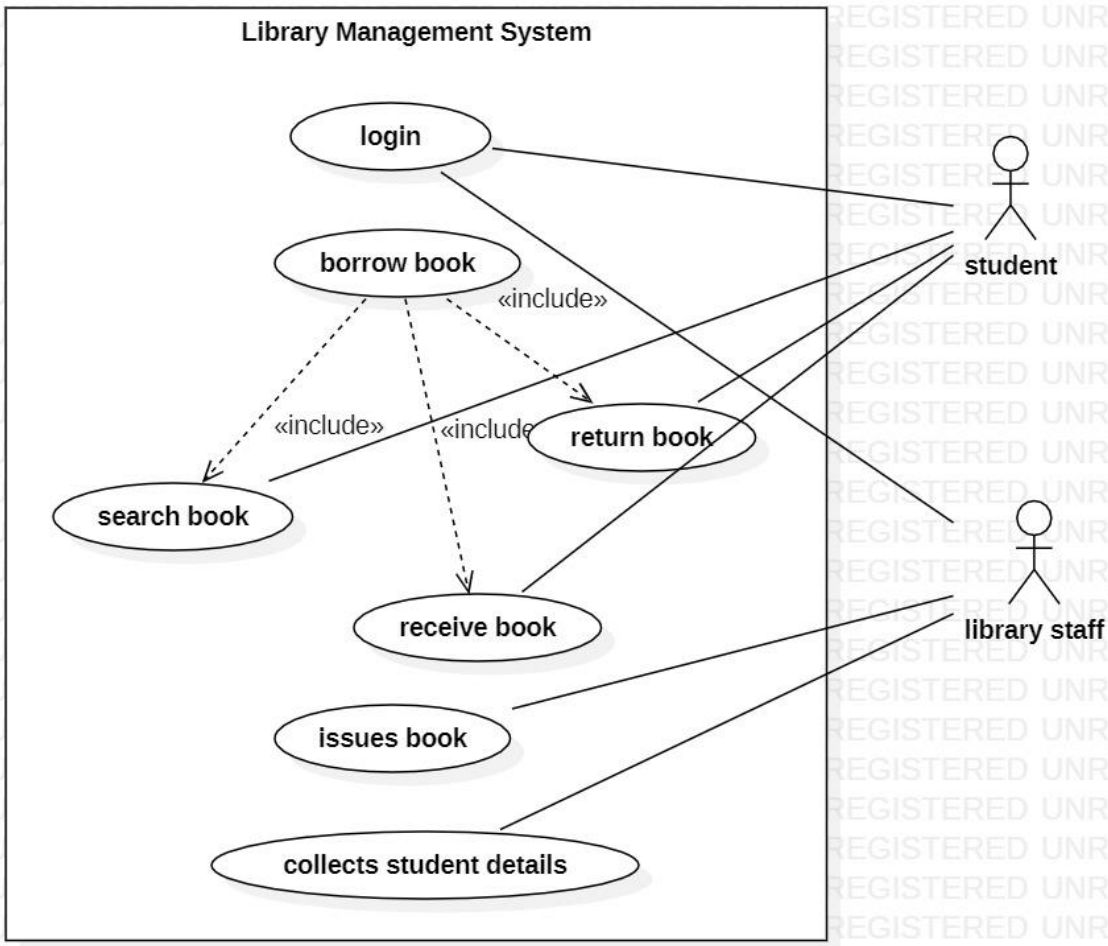

Fig 3.3 – Use Case Diagram for library management

3.5.2 Sequence Diagram

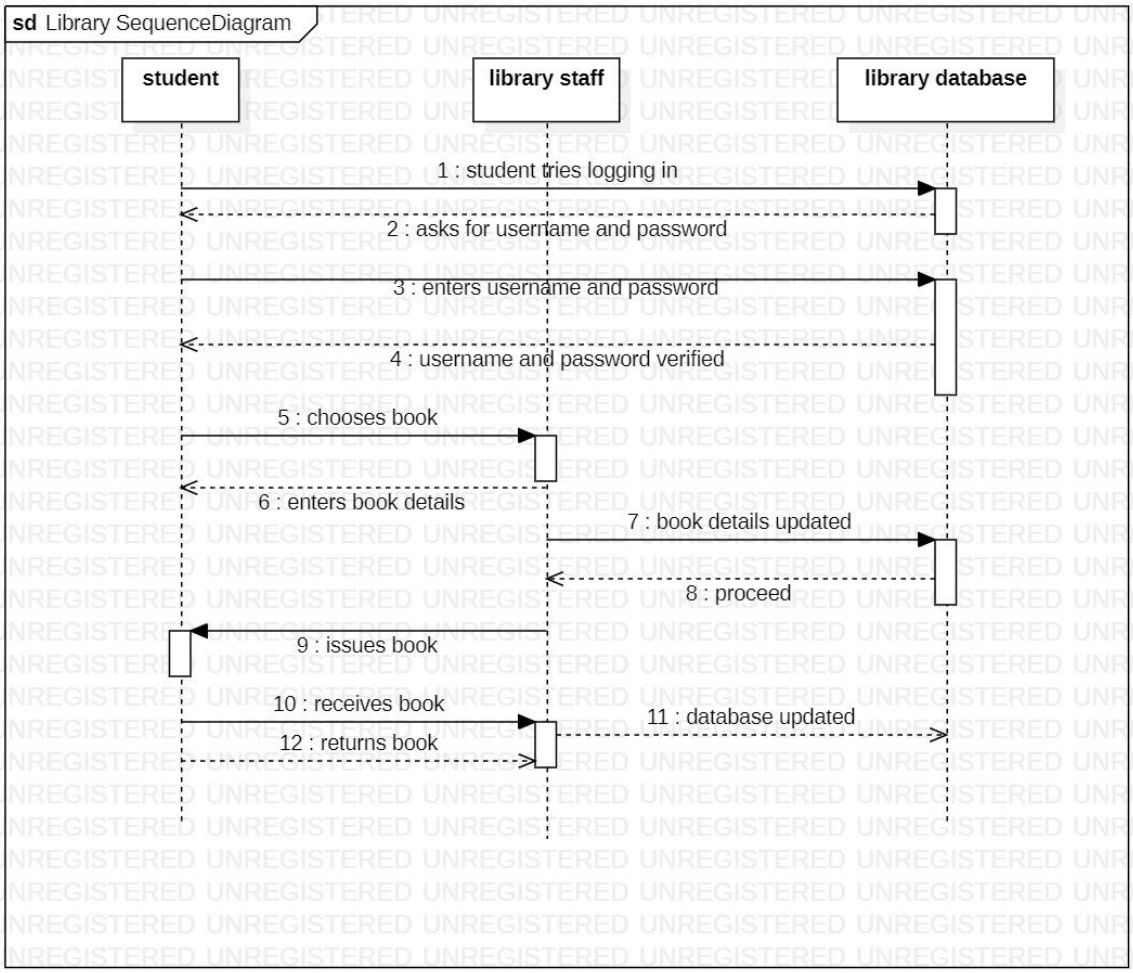

Fig 3.4 - Sequence Diagram for library management

3.5.3 Activity Diagram

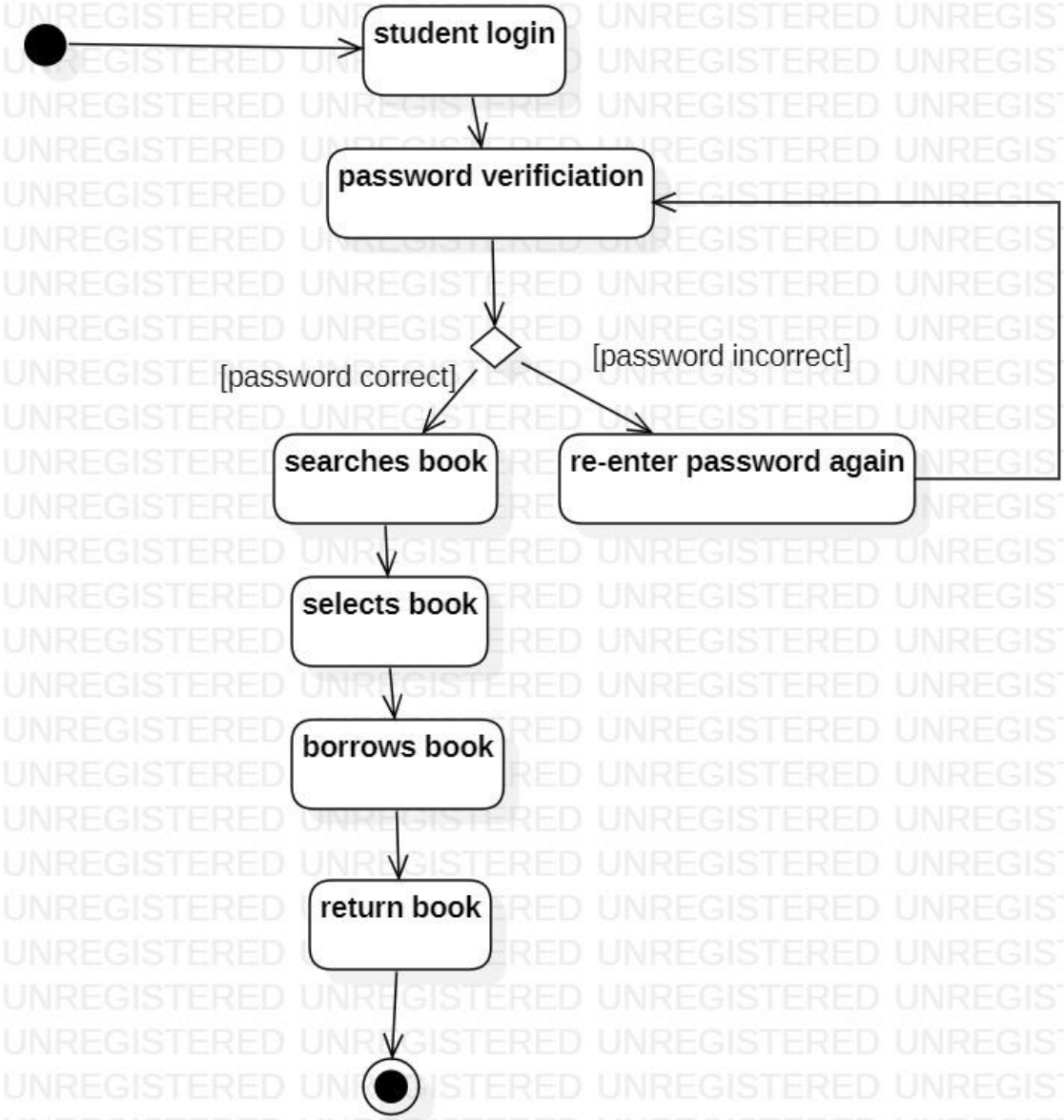

Fig 3.5 - Activity Diagram for library management

4. Stock Maintenance System

4.1 Problem Statement

The Stock Maintenance System is a software application that manages the inventory of a retail store. The purpose of this document is to provide a comprehensive description of the requirements for the Stock Maintenance System.

4.2 Software Requirements Specification

Overview: The Stock Maintenance System is a software application that will be used by a retail store to manage their inventory. The software will allow the store to keep track of their inventory levels, receive alerts when inventory levels are low, and automatically order new inventory when necessary.

General Description: The Stock Maintenance System will have a database to store all inventory information. Users will be able to add new inventory items and track inventory movement. The system will have a reporting feature to generate reports on inventory levels and movement.

Functional Requirements

- The system should allow users to add new inventory items with relevant information such as item name, description, cost, and quantity.
- The system should allow users to view inventory levels in real-time and generate alerts when inventory levels are low.
- The system should automatically order new inventory when inventory levels are low.
- The system should be able to track inventory movement, such as sales or returns.
- The system should have a reporting feature that allows users to generate reports on inventory levels, inventory movement, and sales data.
- The system should allow users to edit and delete inventory items as necessary
- The system should provide a search function to easily find inventory items by name, description, or other relevant information.

Interface Requirements

- The interface should be user-friendly and easy to navigate.
- The system should have clear and concise menus and buttons to perform tasks.
- The system should have a dashboard that displays important information such as inventory levels, sales data, and alerts.

Performance Requirements

- The system should be able to handle a large amount of data without slowing down.
- The system should be responsive and have a fast response time.
- The system should be able to generate reports quickly.
- The system should be able to handle a large number of simultaneous users.
- The system should be able to process inventory updates and generate alerts in real-time.

Non-Functional Attributes

- The system should be reliable and have a low error rate.
- The system should be easy to maintain with clear documentation and easy troubleshooting.
- The system should be user-friendly and intuitive.
- The system should be secure and protect inventory information
- The system should be able to import and export inventory data in a variety of file formats.
- The system should be designed to minimize the amount of manual data entry required.

Design Constraints

- The system should be designed to run on Windows operating systems.
- The system should be modular and scalable for future expansion.
- The system should have a secure database to store inventory information.
- The system should be designed to be compatible with a variety of hardware configurations.
- The system should have a clear and easy-to-understand user interface that minimizes the need for training.

4.3 Class Diagram

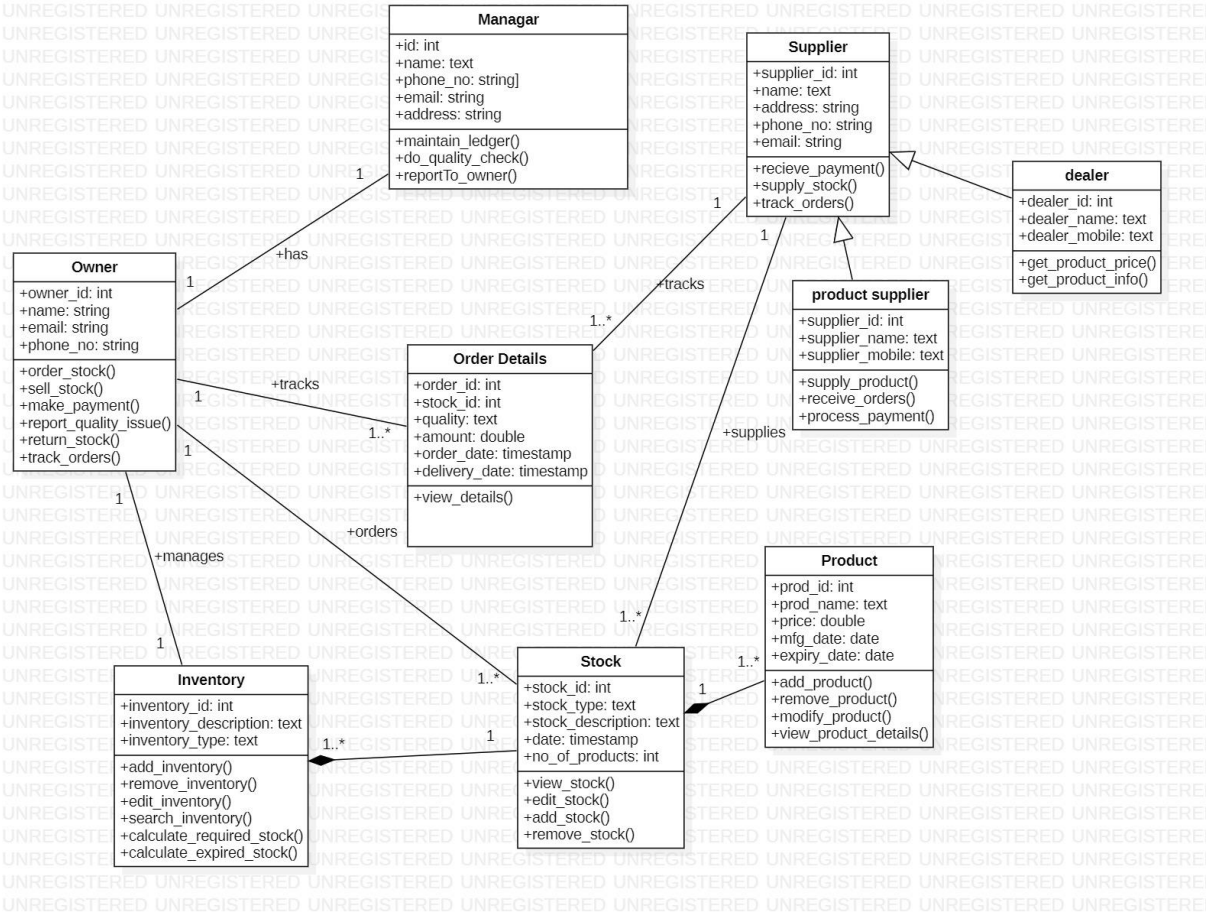

Fig 4.1 - Class Diagram for stock maintenance

4.4 State Diagram

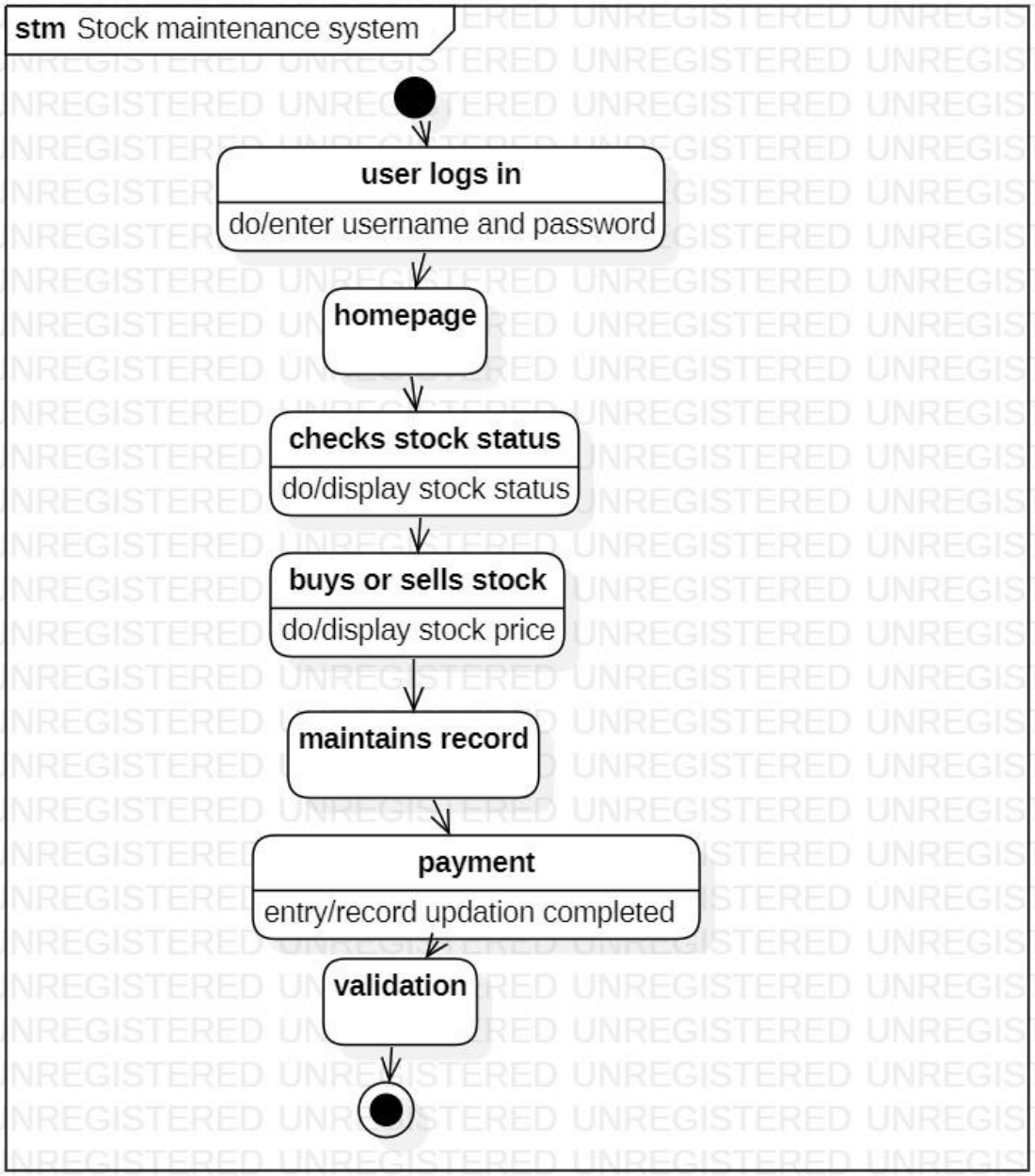

Fig 4.2 - State Diagram for stock maintenance

4.5 Interaction Diagram

4.5.1 Use Case Diagram

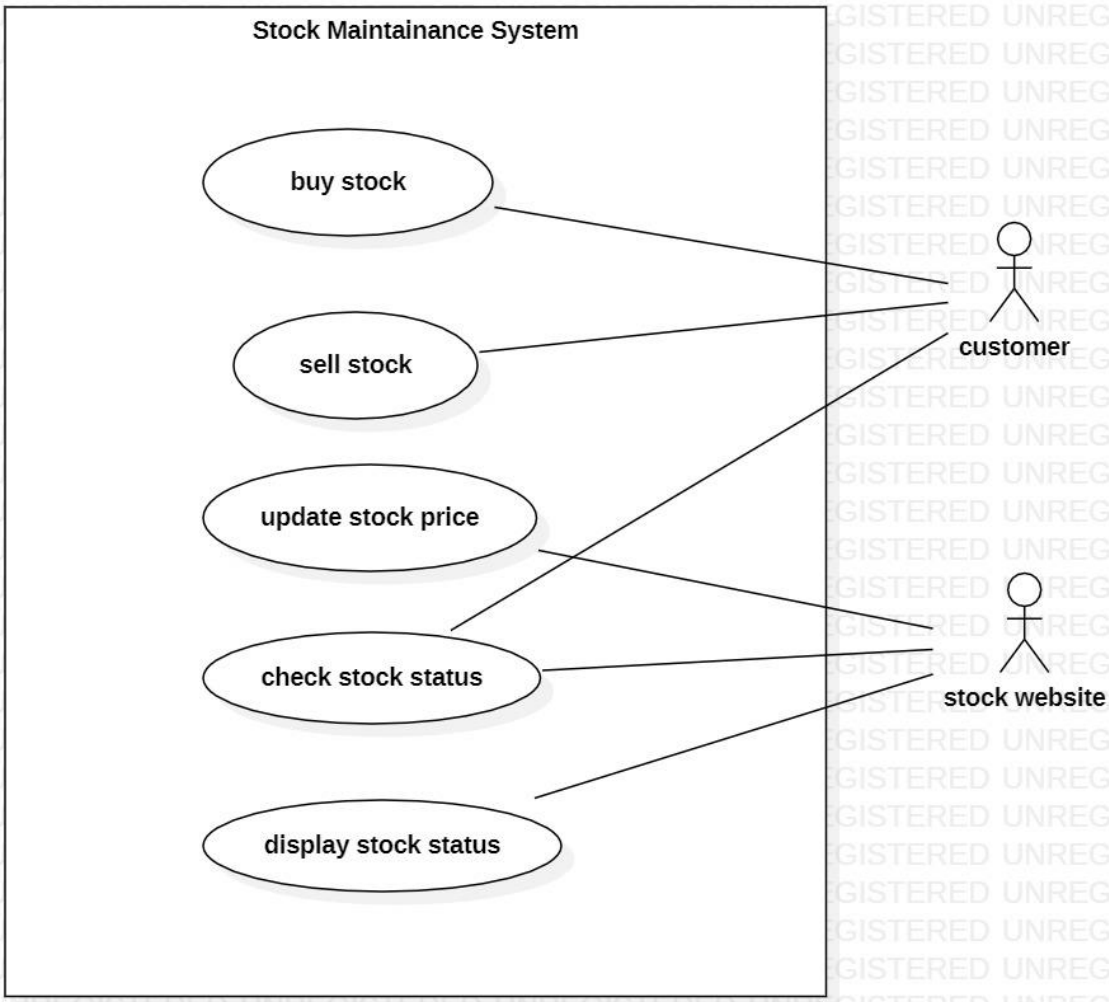

Fig 4.3 – Use Case Diagram for stock maintenance

4.5.2 Sequence Diagram

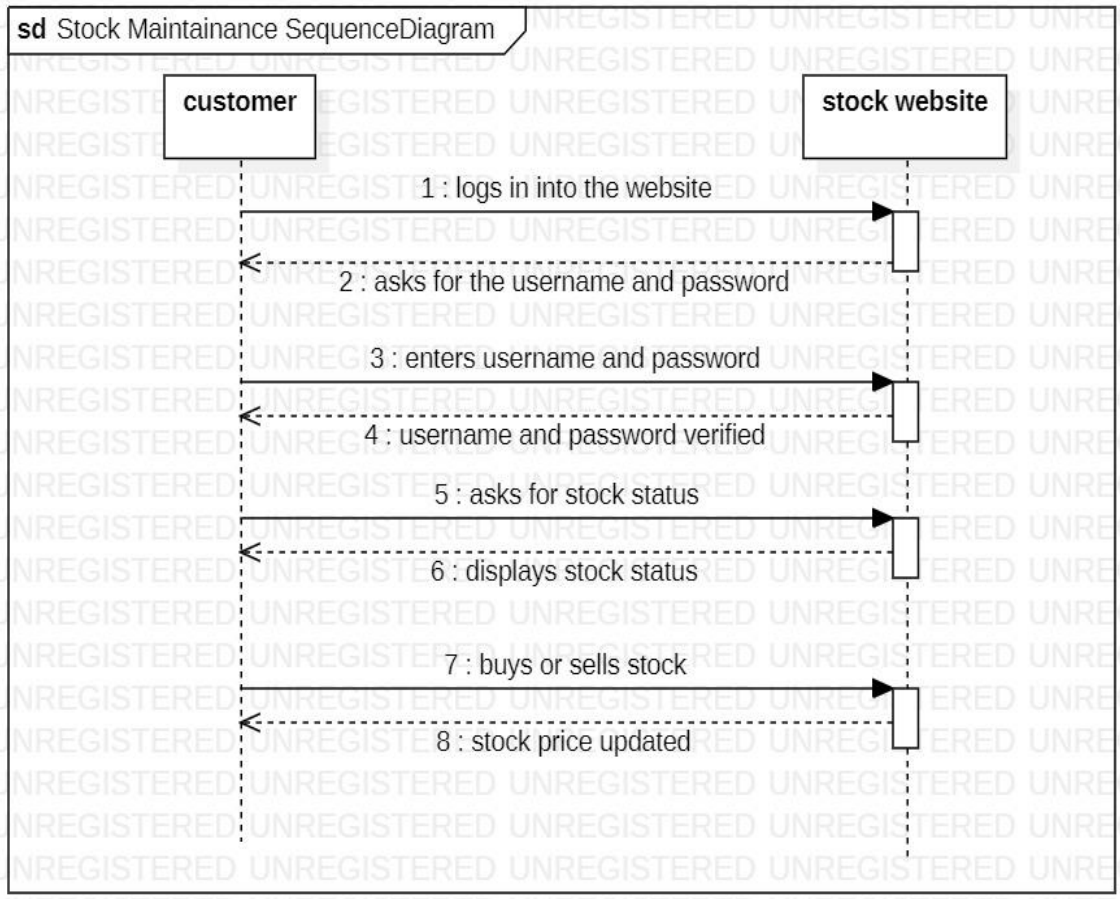

Fig 4.4 – Sequence Diagram for stock maintenance

4.5.3 Activity Diagram

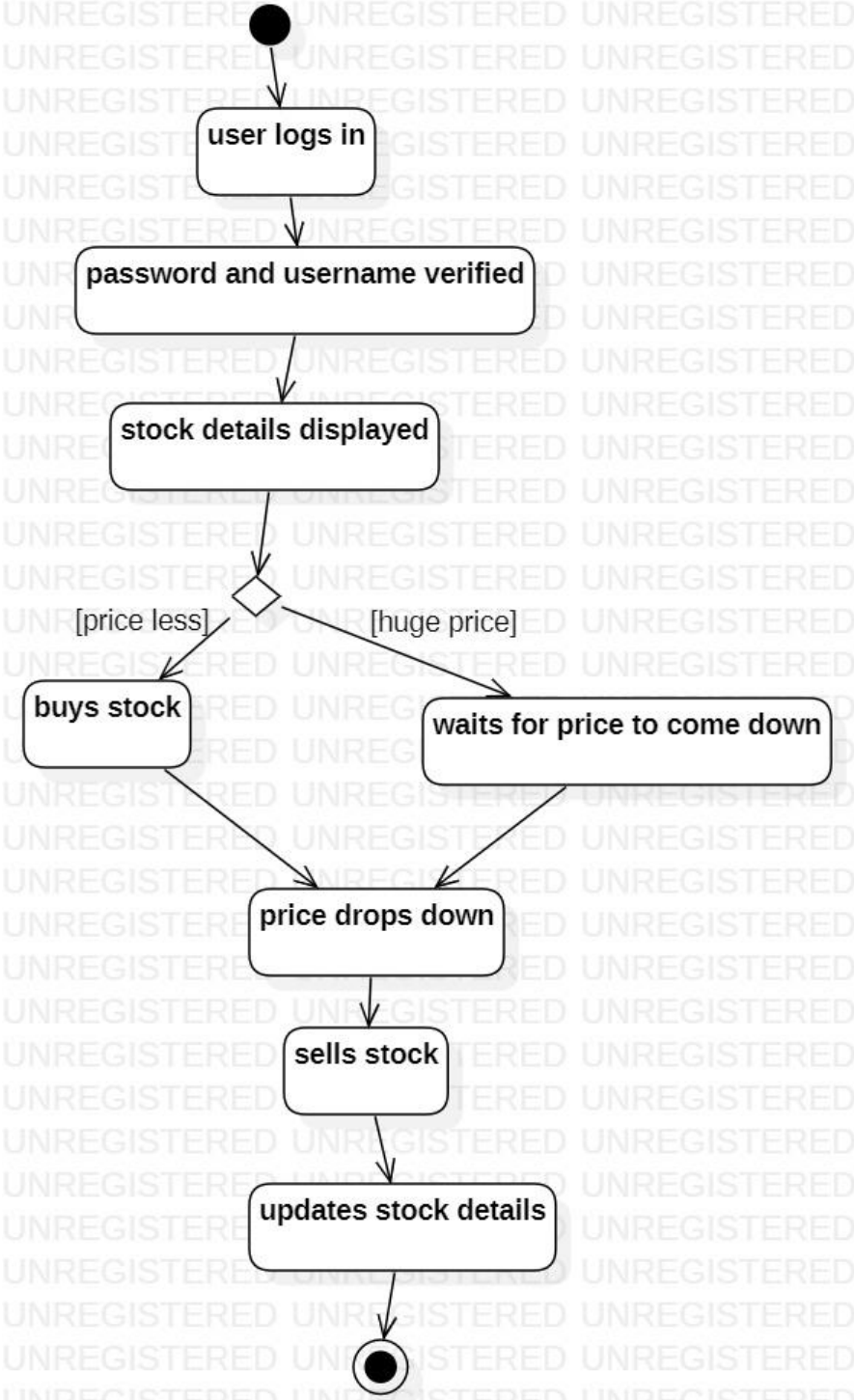

Fig 4.5 – Activity Diagram for stock maintenance

5. Passport Authentication System

5.1 Problem Statement

The existing passport authentication system faces several challenges that hinder efficient and secure passport verification processes. To address these challenges, a new and improved passport authentication system is needed. The system should incorporate advanced technologies, such as machine learning, biometrics, and document verification algorithms, to enhance security and accuracy. It should establish standardized verification procedures, ensure interoperability with international standards, integrate with relevant databases, and prioritize data privacy and security. Implementing such a system would streamline passport verification processes, enhance national security, and provide a more efficient and secure experience for passport holders and immigration authorities.

5.2 Software Requirements Specification

General description: A passport automation system is a digital system designed to automate and streamline the process of issuing passports to citizens. It typically includes a database of citizen information, which can be accessed and updated by authorized government officials. It may also include online application portals, allowing citizens to apply for passports from the comfort of their homes or through designated passport offices. The system uses biometric technology, such as facial recognition and fingerprint scanning, to verify the identity of the applicant and prevent fraud. Once an application is submitted, the system automatically checks the information provided against government databases to ensure accuracy and completeness. The system may also include an appointment scheduling feature, allowing applicants to schedule appointments for passport interviews and processing. Passport automation systems may also include features such as electronic payment processing, document scanning and verification, and automatic printing of passports once all requirements have been met.

Functional Requirements

- It must authenticate the identity of the user to ensure that only authorized individuals can access the system.
- It must allow citizens to submit passport applications online, as well as validate and process the application in a timely and accurate manner.
- It must maintain a central database of citizen information.
- It must allow citizens to schedule appointments for passport interviews and processing.
- It must enable citizens to pay the required fees for passport issuance and processing and process payments securely and accurately.

● It must allow citizens to upload and scan supporting documents required for passport issuance, such as birth certificates and proof of citizenship and verify the authenticity of these documents.
● It must use biometric technology, such as facial recognition and fingerprint scanning, to verify the identity of the applicant and prevent fraud.
● It must automatically generate and print passports once all requirements have been met.

Interface Requirements

● It must have a user-friendly interface that is easy to navigate and understand.
● It must be accessible through multiple channels, such as online portals and designated passport offices, to provide citizens with flexibility and convenience.
● It must support multiple languages.
● It must be designed to comply with accessibility requirements, such as providing alternative text for images and videos.
● It must use secure authentication protocols, such as two-factor authentication.
● It must provide clear feedback to users at every step of the application process.
● It must have a consistent design and layout to ensure that users can easily navigate and use the system.
● The system must be designed to integrate with other systems, such as border control and immigration systems.

Performance Requirements: In this, how a software system performs desired functions under specific condition is explained. It also explains required time, required memory, maximum error rate, etc.

Design Constraints: In this, constraints which simply means limitation or restriction are specified and explained for design team. Examples may include use of a particular algorithm, hardware and software limitations, etc.

Non-Functional Attributes: In this, non-functional attributes are explained that are required by software system for better performance. An example may include Security, Portability, Reliability, Reusability, Application compatibility, Data integrity, Scalability capacity, etc.

Preliminary Schedule and Budget: In this, initial version and budget of project plan are explained which include overall time duration required and overall cost required for development of project.

5.3 Class Diagram

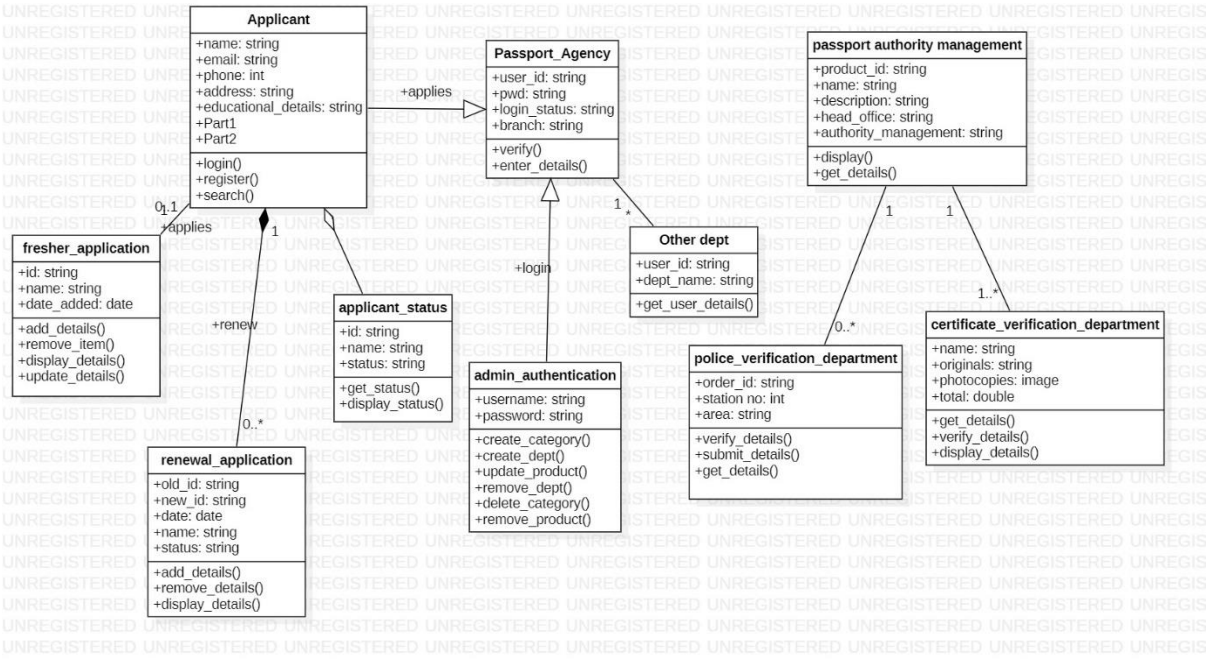

Fig 5.1 – Class Diagram for passport authentication

5.4 State Diagram

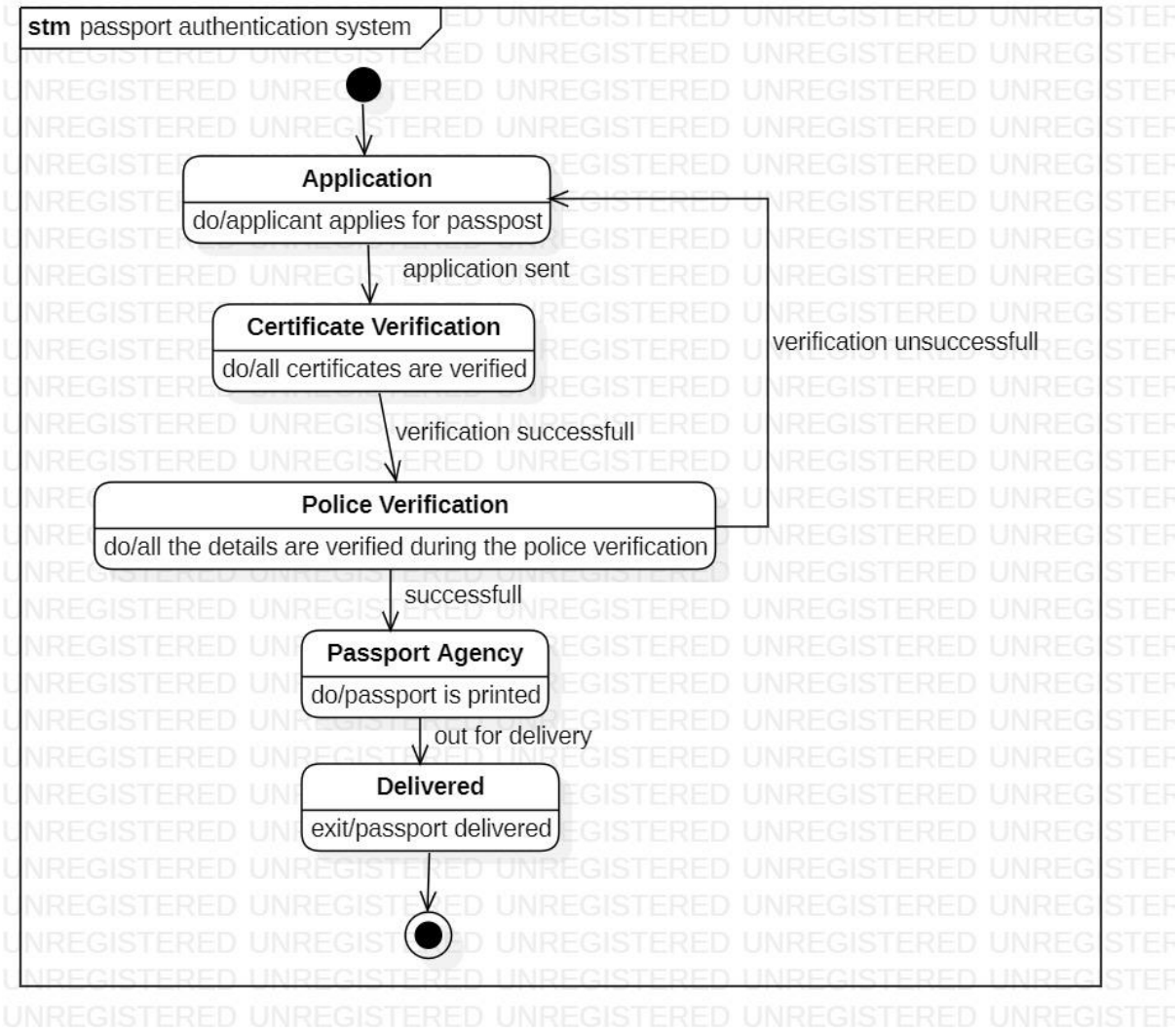

Fig 5.2 – State Diagram for passport authentication

5.5 Interaction Diagram

5.5.1 Use Case Diagram

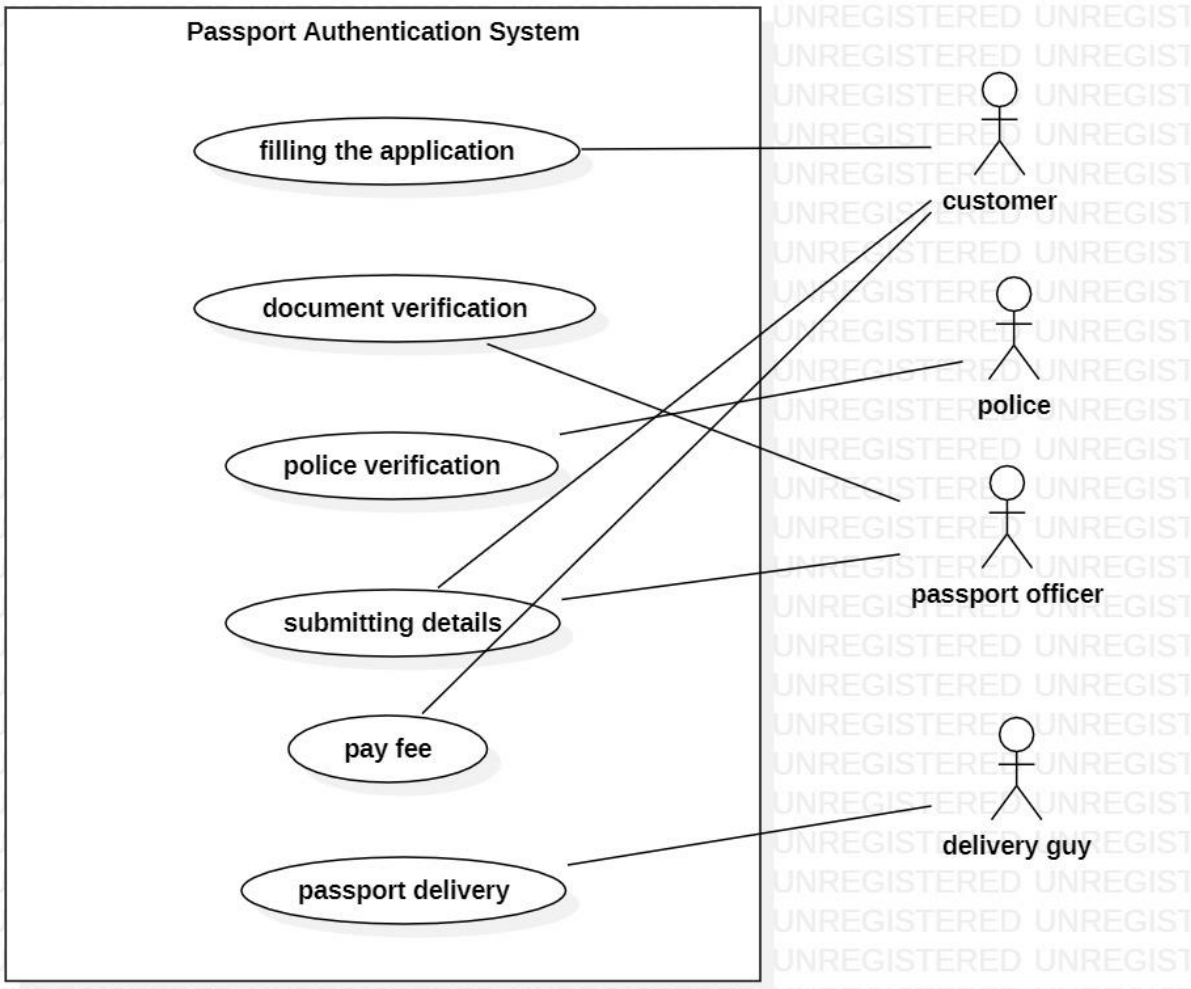

Fig 5.3 – Use Case Diagram for passport authentication

5.5.2 Sequence Diagram

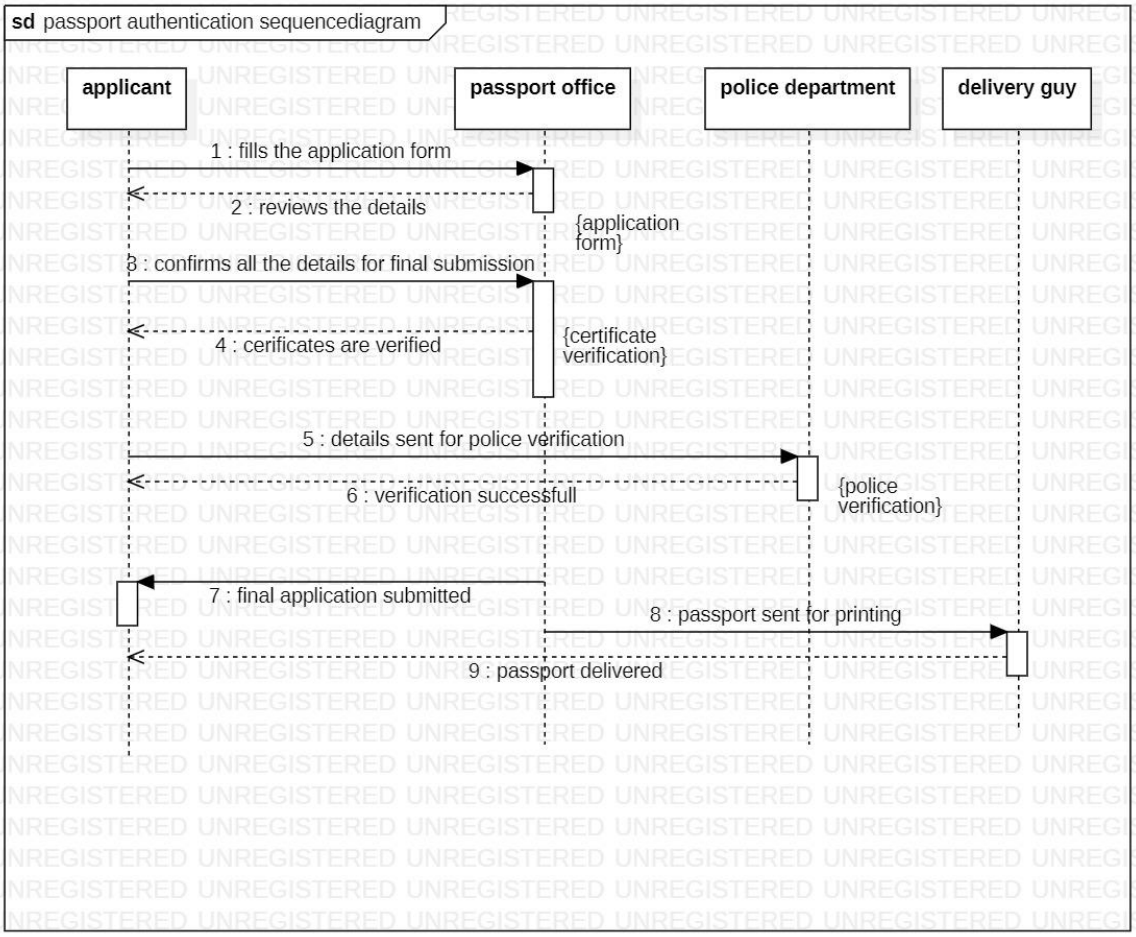

Fig 5.4 – Sequence Diagram for passport authentication

5.5.3 Activity Diagram

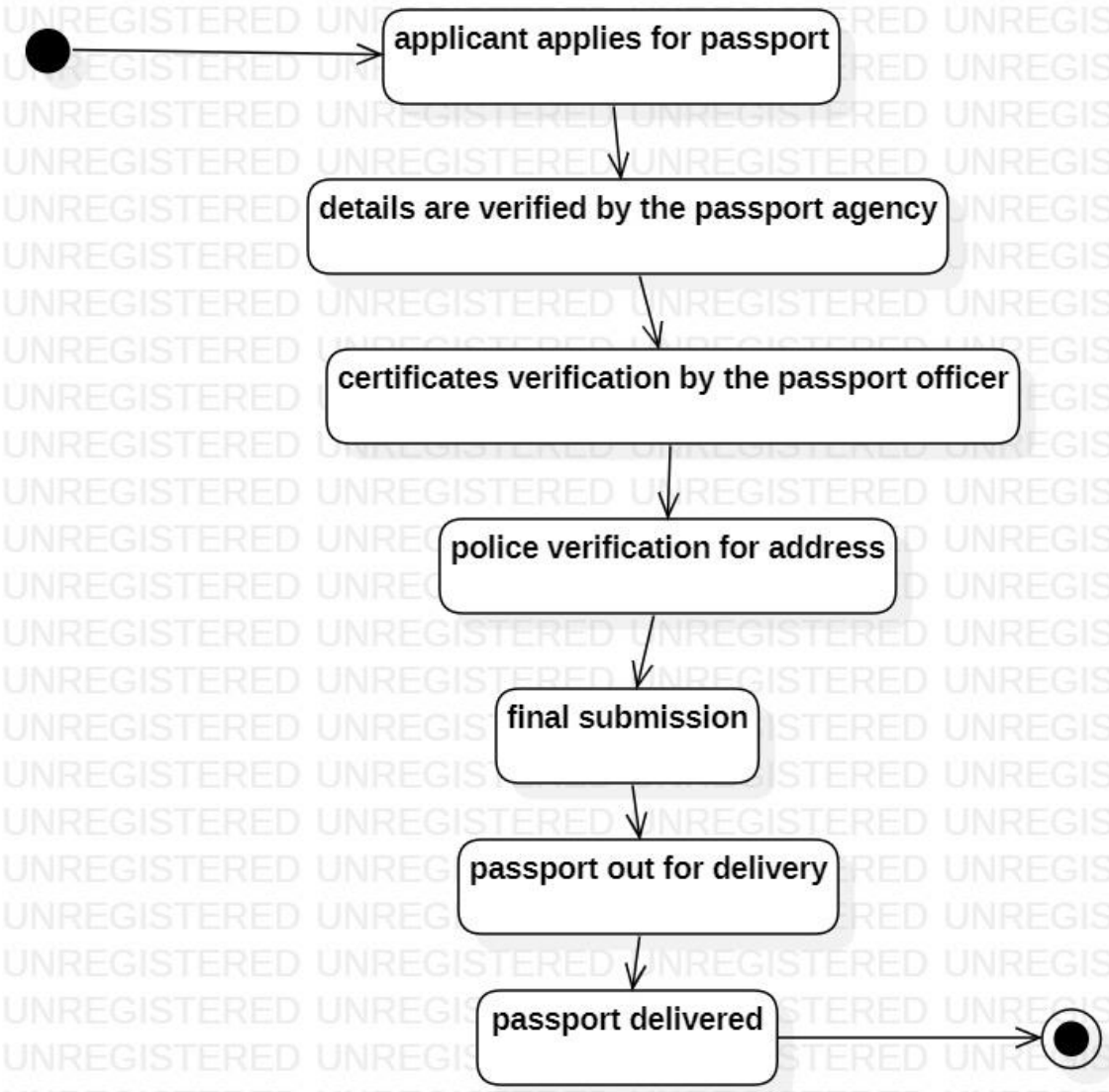

Fig 5.5 – Activity Diagram for passport authentication

6. Railway Reservation System

6.1 Problem Statement

The existing railway reservation system faces several challenges that hinder efficient and seamless booking processes for passengers. To address these challenges, a new and improved railway reservation system is required. The system should have enhanced online booking capacity, a simplified and user-friendly booking process, accurate real-time seat availability information, flexible payment and ticketing options, seamless integration with other transport systems, and mobile and offline booking alternatives. Implementing such a system would significantly enhance the booking experience for passengers, improve operational efficiency, and foster a positive perception of railway travel.

6.2 Software Requirements Specification

Overview – The railway reservation system is a computer-based application that enables passengers to book and manage train tickets, as well as for railway staff to manage the train schedules, seat availability, and other related information. The system typically involves multiple components, including a user interface for passengers to make reservations, a database to store ticket and train-related information, and an administrative interface for railway staff to manage the system.

General description: A railway reservation system is a software application that automates the process of booking train tickets and managing reservations. It enables passengers to reserve seats or berths in advance for a specific train, route, date, and time. The system helps to eliminate the need for long queues at railway stations, as passengers can make their reservations online or through designated reservation counters. It typically consists of several modules, including, user registration and login, seat availability and booking, payment and ticketing, cancellation and refund, reporting and analytics etc. The railway reservation system helps to improve the efficiency of the ticket booking process, reduce errors and fraud, and provide a better experience for passengers. It is an essential component of modern railway operations, helping to streamline the process of managing reservations, inventory, and revenue.

Functional Requirements: The functional requirements of this system are:

- The system should be able to track, record, and manage incidents that occur on the railway network, such as train delays, equipment failures, and accidents.
- The system should be able to notify relevant stakeholders, such as train drivers, maintenance crews, and passengers, of incidents and provide real-time updates on the status of incidents.

● The system should be able to allocate resources, such as maintenance crews, equipment, and alternative transport, to resolve incidents and minimize disruption to train services.
● The system should be able to generate reports and analytics on incident trends and performance metrics, such as incident response time, resolution time, and customer satisfaction.
● The system should be able to integrate with other railway systems, such as train scheduling systems, signaling systems, and passenger information systems, to provide a seamless and coordinated response to incidents.
● The system should have user management capabilities to control access to the system and assign roles and permissions to users based on their responsibilities.

Interface Requirements:

● The interface should have a modern and visually appealing design.
● The interface should be easy to navigate.
● The interface should be responsive and adapt to different screen sizes.
● The interface should allow for customization, such as the ability to change colors, logos, and fonts.
● The interface should support multiple languages.
● The interface should allow for easy search and filtering of information.
● The interface should provide alerts and notifications to keep passengers informed of important events.
● The interface should be able to integrate with other systems used by the system, such as payment gateways.

Performance Requirements:

● Response Time: The system should be responsive and provide real-time updates to users. Response times should be within acceptable limits, typically a few seconds for search queries, seat availability, and booking transactions.
● Throughput: The system should be able to handle a high volume of transactions simultaneously, including ticket bookings, cancellations. It should be able to handle peak loads without any service interruptions.
● Availability: The system should be available 24x7, with minimum downtime for maintenance and upgrades.
● Scalability: The system should be scalable to accommodate a growing number of users, trains, and routes. The system should be able to scale up or down without affecting performance, response times, or availability.
● Data Integrity: The system should ensure the integrity and consistency of data, with proper validation of user inputs, transaction logs, and backups. The system should also be able to recover from crashes or corruption quickly.

Non-Functional Attributes:

● The system should be able to handle a large number of incidents and users, respond quickly to incidents, and provide real-time updates to stakeholders.

● The system should be highly available and reliable, with a low likelihood of downtime or data loss.

● The system should be available 24/7, with minimal planned and unplanned downtime.

● The system should be easy to maintain and upgrade, with minimal impact on operations during maintenance activities.

● The system should be easy to use and navigate, with a user-friendly interface and intuitive workflows.

6.3 Class Diagram

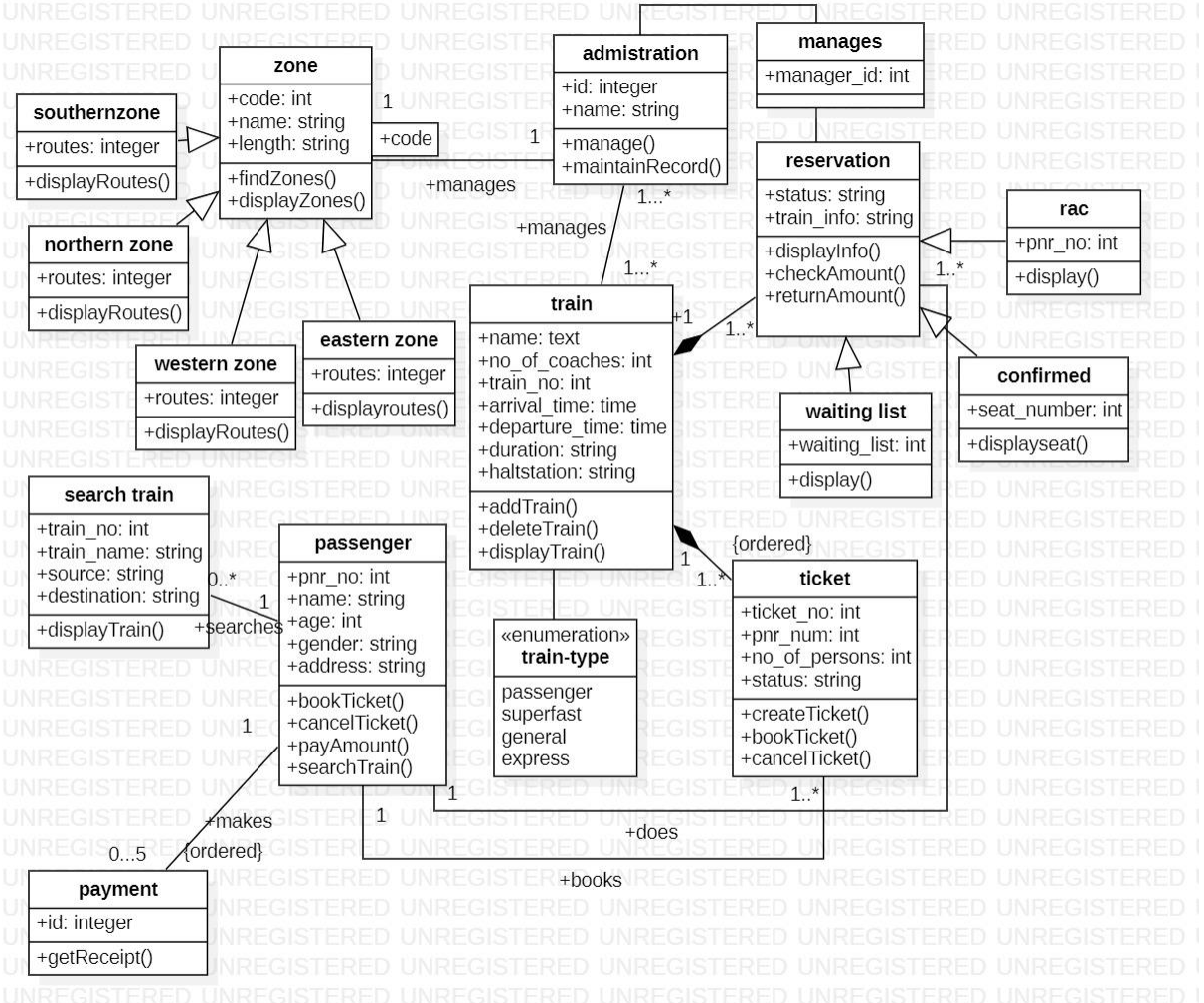

Fig 6.1 – Class Diagram for railway reservation

6.4 State Diagram

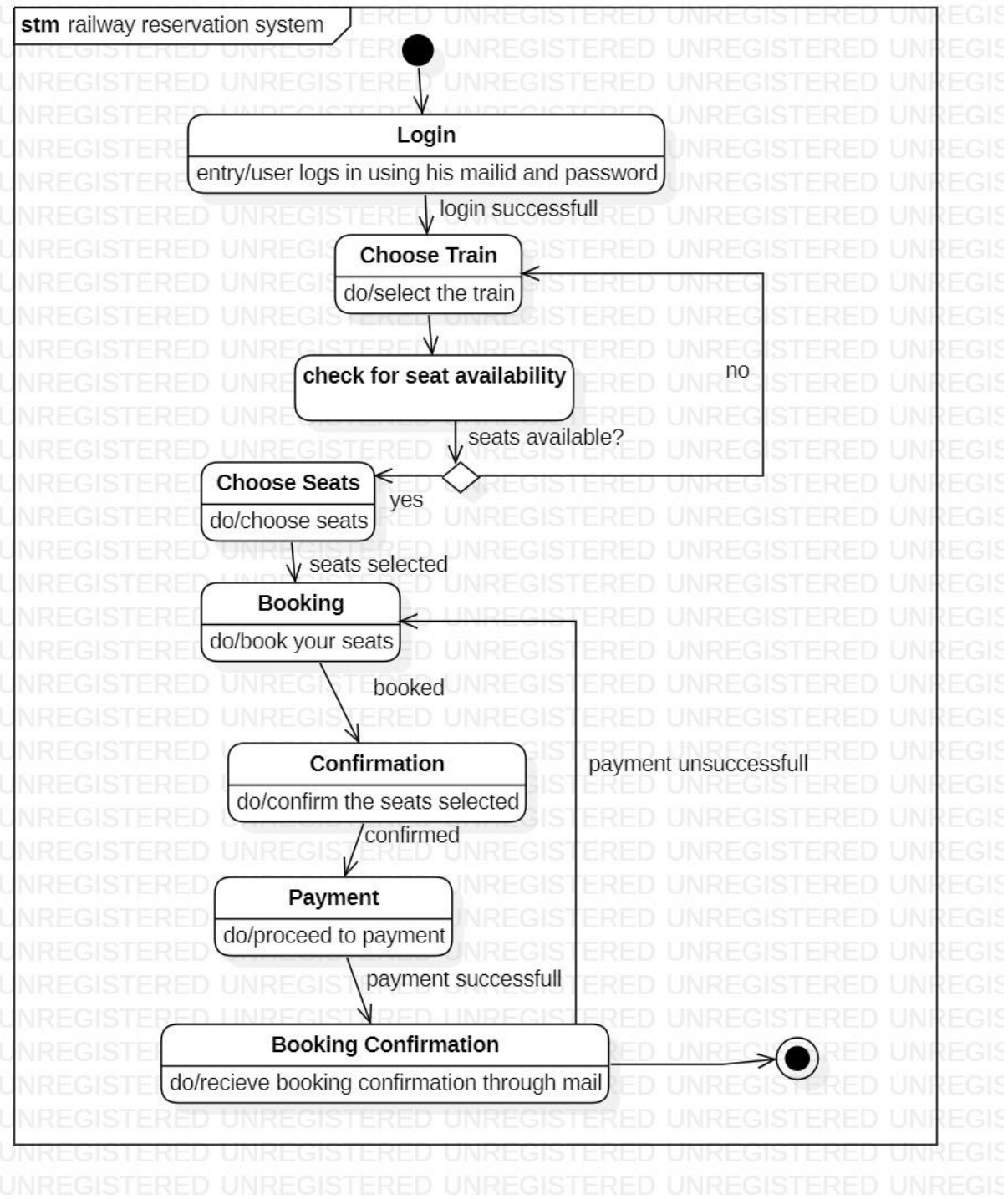

Fig 6.2 – State Diagram for railway reservation

6.5 Interaction Diagram

6.5.1 Use Case Diagram

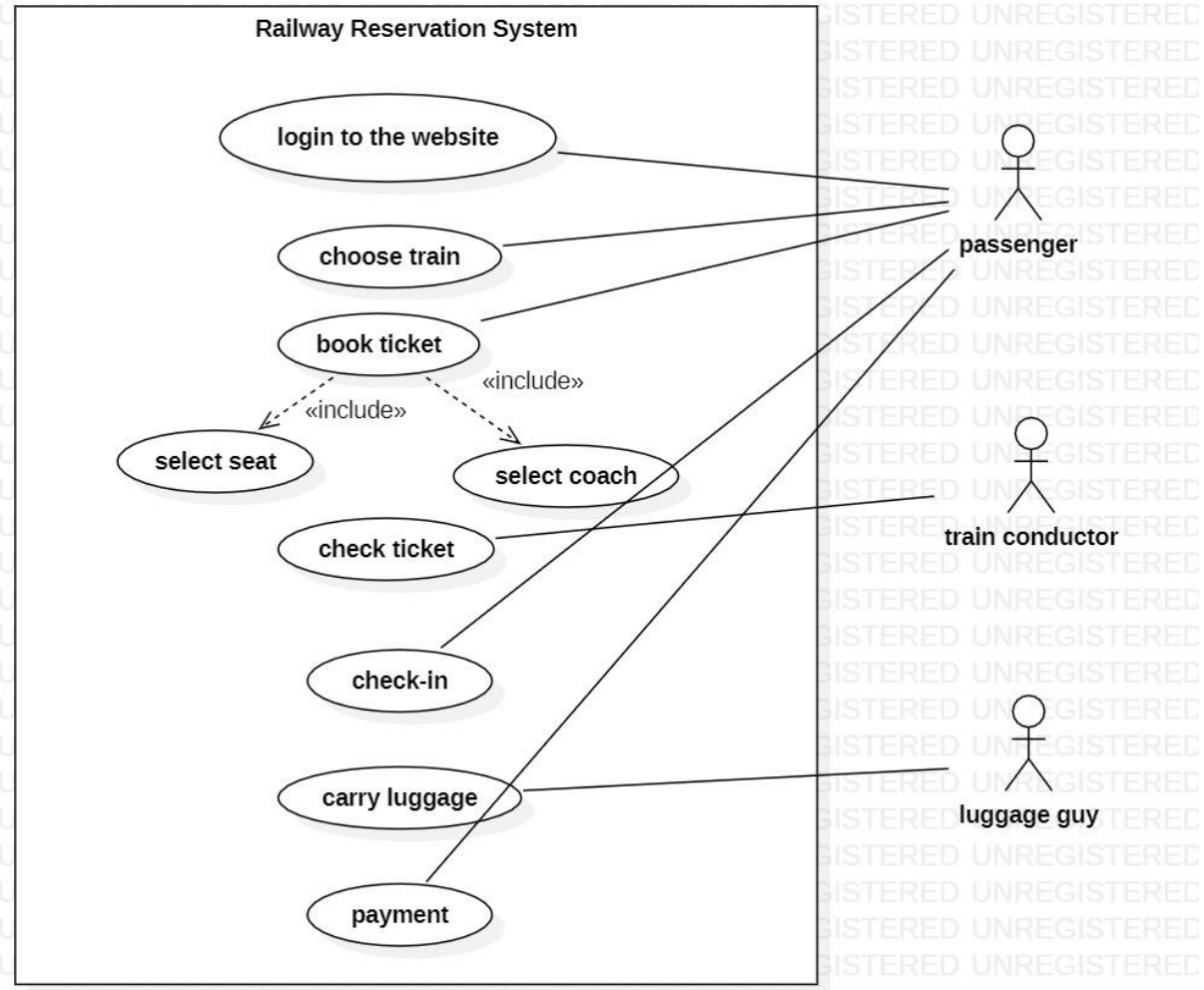

Fig 6.3 – Use Case Diagram for railway reservation

6.5.2 Sequence Diagram

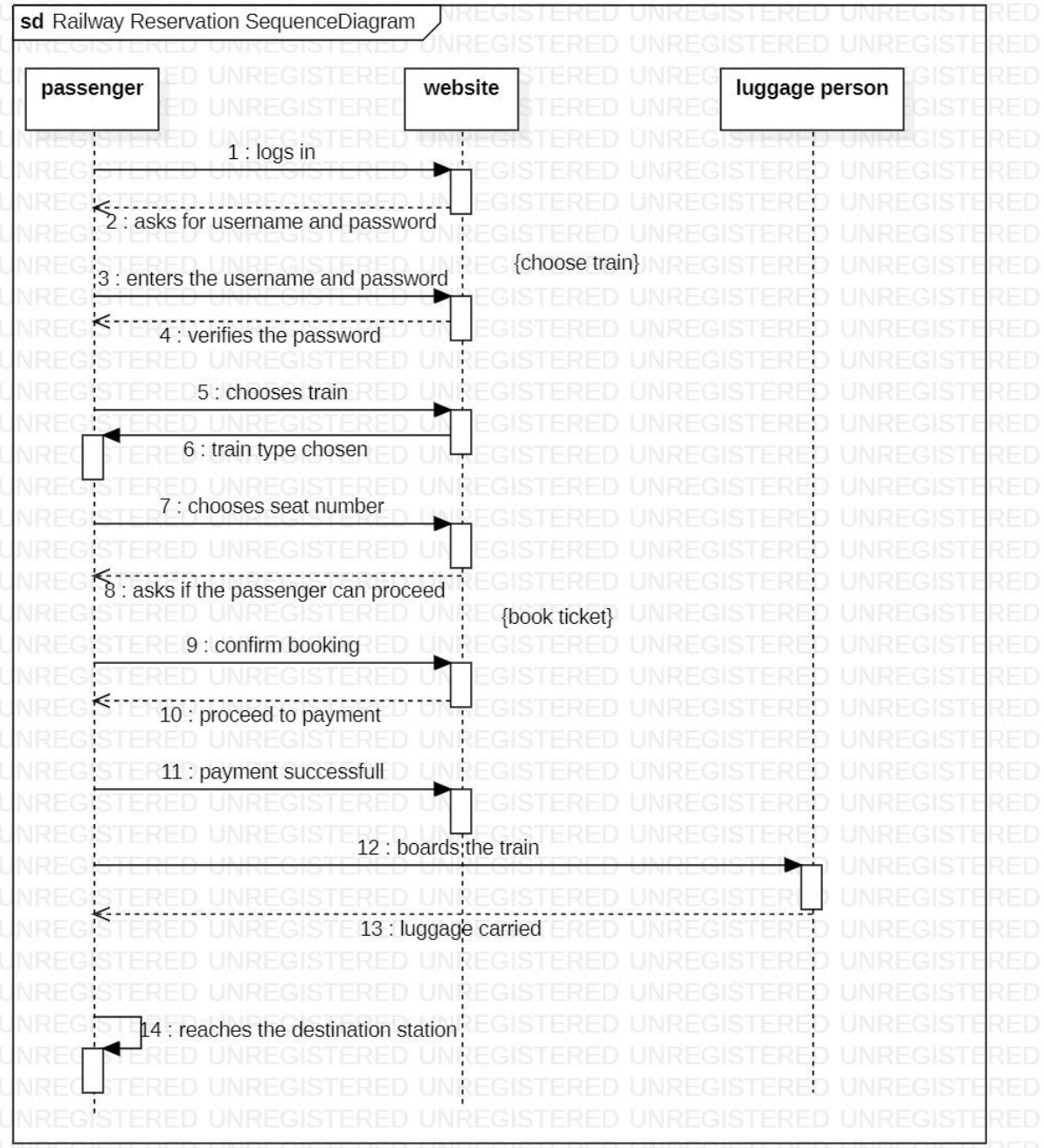

Fig 6.4 – Sequence Diagram for railway reservation

6.5.3 Activity Diagram

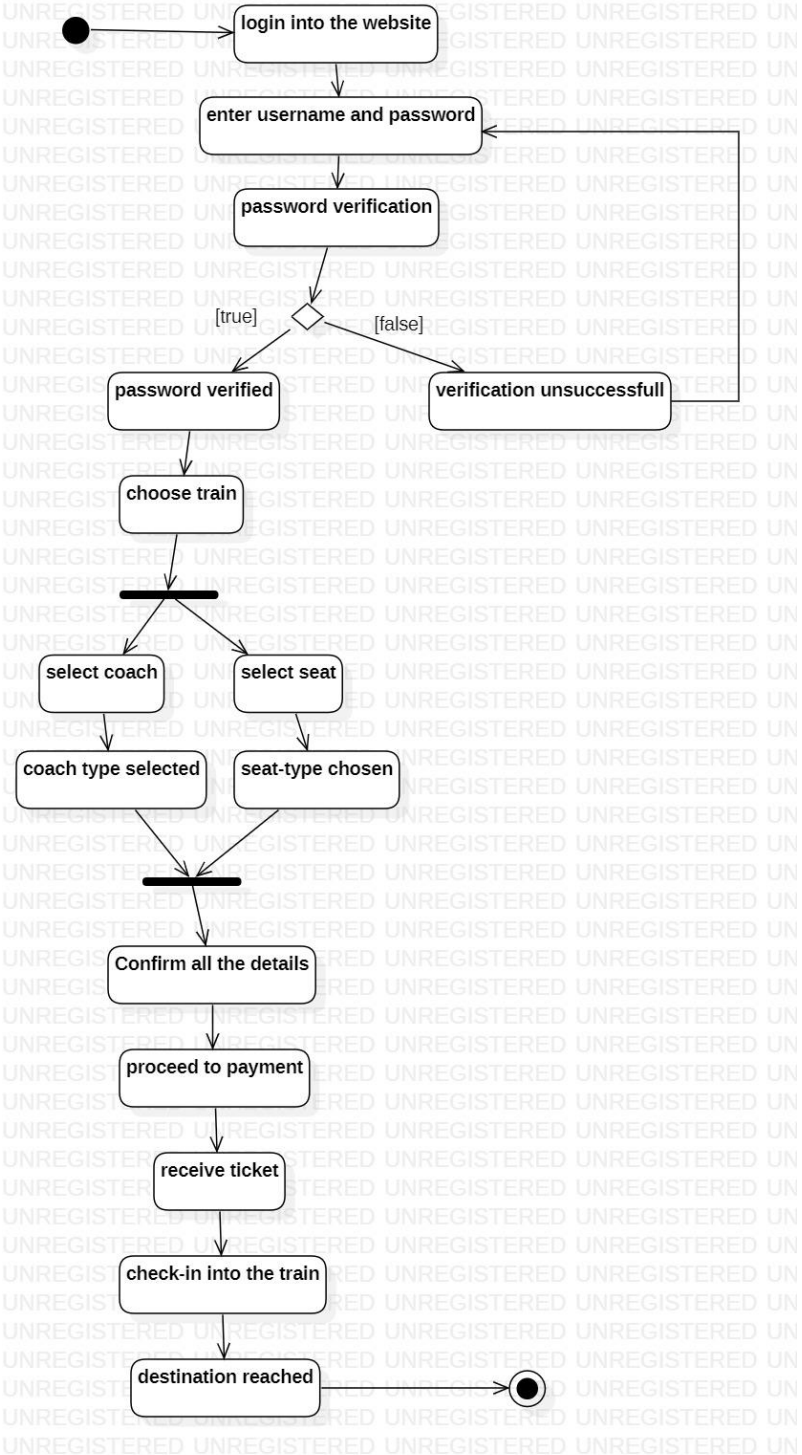

Fig 6.5 – Activity Diagram for railway reservation

7. Online Shopping System

7.1 Problem Statement

The existing online shopping system faces several challenges that hinder efficient and satisfying e-commerce experiences for customers. To address these challenges, a new and improved online shopping system is needed. The system should feature a user-friendly interface with intuitive navigation, comprehensive and accurate product information, robust search and filtering capabilities, efficient inventory management, transparent shipping and delivery processes with order tracking, and seamless customer support and returns processes. Implementing such a system would enhance the overall online shopping experience, increase customer satisfaction, and foster customer loyalty for the e-commerce platform.

7.2 Software Requirements Specification

Overview – This system provides an easy solution for customers to buy the product without going to the shop and also for the shop owner to sale the product. This proposed system can be used by any naïve users and it does not require any educational level, experience or technical expertise in computer field but it will be of good use if user has the good knowledge of how to operate a computer.

General description: An online shopping system is a software application that enables customers to purchase products and services over the internet. The system typically consists of several components, including a website or mobile app, a database, and payment processing software. Online shopping systems are used by retailers to sell products to customers in a convenient and efficient way, without the need for physical stores. The online shopping system provides customers with a range of features and services, including, product catalog, search and navigation, shopping cart, checkout process, payment processing, order fulfillment, order tracking, customer service etc. Online shopping systems help retailers to reach a wider audience, increase sales, and provide a convenient and flexible shopping experience for customers. They are a key component of modern retail operations, enabling retailers to compete in a global marketplace and adapt to changing customer needs and preferences.

Functional Requirements: Some of the functional requirements for an online shopping system may include:

- The system should display an up-to-date catalog of products or services available for purchase, including product descriptions, images, and prices.
- The system should allow customers to search for products or services based on keywords, categories, price ranges, and other filters.

● The system should allow customers to add items to a shopping cart, view the contents of their cart, and adjust quantities or remove items as needed.

● The system should enable customers to securely check out and pay for their purchases using a variety of payment options, such as credit cards, PayPal, or mobile payment services.

● The system should process orders, generate invoices or receipts, and initiate the fulfillment process, such as shipping or delivery.

● The system should allow customers to track the status of their orders, such as shipment tracking information or estimated delivery dates.

● The system should allow customers to create and manage their accounts, view their order history, save payment and shipping information, and manage their preferences

. ● The system should provide customer support and assistance, such as live chat, email, or phone support.

Interface Requirements:

● The interface should have a modern and visually appealing design.

● The interface should be easy to navigate.

● The interface should be responsive and adapt to different screen sizes.

● The interface should allow for customization, such as the ability to change colors, logos, and fonts.

● The interface should support multiple languages.

● The interface should allow for easy search and filtering of information.

● The interface should provide alerts and notifications to keep customers informed of the status of their product.

● The interface should be able to integrate with other systems used by the system, such as payment gateways.

Performance Requirements:

● Response Time: The system should be responsive and provide real-time updates to users. Response times should be within acceptable limits, typically a few seconds for search queries, listing products, and online transactions.

● Throughput: The system should be able to handle a high volume of transactions simultaneously. The system should be able to handle peak loads during festive seasons and special occasions without any service interruptions.

● Availability: The availability of the system should be at least 99.9%, which means that the system should be operational for at least 99.9% of the time, with only minimal unscheduled downtime.

● Scalability: The system should be scalable to accommodate a growing number of users, products, and orders. The system should be able to scale up or down based on demand without affecting performance,

response times, or availability.

● Security: The system should be designed to prevent unauthorized access, fraud, and cyber-attacks. The system should have robust security measures, including encryption, multi-factor authentication, regular security audits and testing.

Design Constraints:

● It must be designed to protect customer data, including personal information and payment details, from unauthorized access or use.

● It system must be designed to handle a large volume of users, products, and transactions, without experiencing performance issues or downtime.

● It system must be designed to meet the needs and expectations of different user groups, including customers, retailers, and administrators.

● The design and development of the online shopping system must be cost-effective, taking into account the available budget and resources.

Non-Functional Attributes: Some non-functional requirements for an online shopping system include:

● The system should be able to handle a large number of simultaneous users and transactions, respond quickly to user requests, and minimize latency and processing time.

● The system should be available 24/7, with minimal planned and unplanned downtime.

● The system should be highly available and reliable, with a low likelihood of downtime or data loss.

● The system should have robust security measures in place to protect against unauthorized access, data breaches, and other security threats, such as encryption, access controls, and intrusion detection.

● The system should be easy to use and navigate, with a user-friendly interface and intuitive workflows.

● The system should be able to handle a growing number of users and transactions without significant degradation in performance.

7.3 Class Diagram

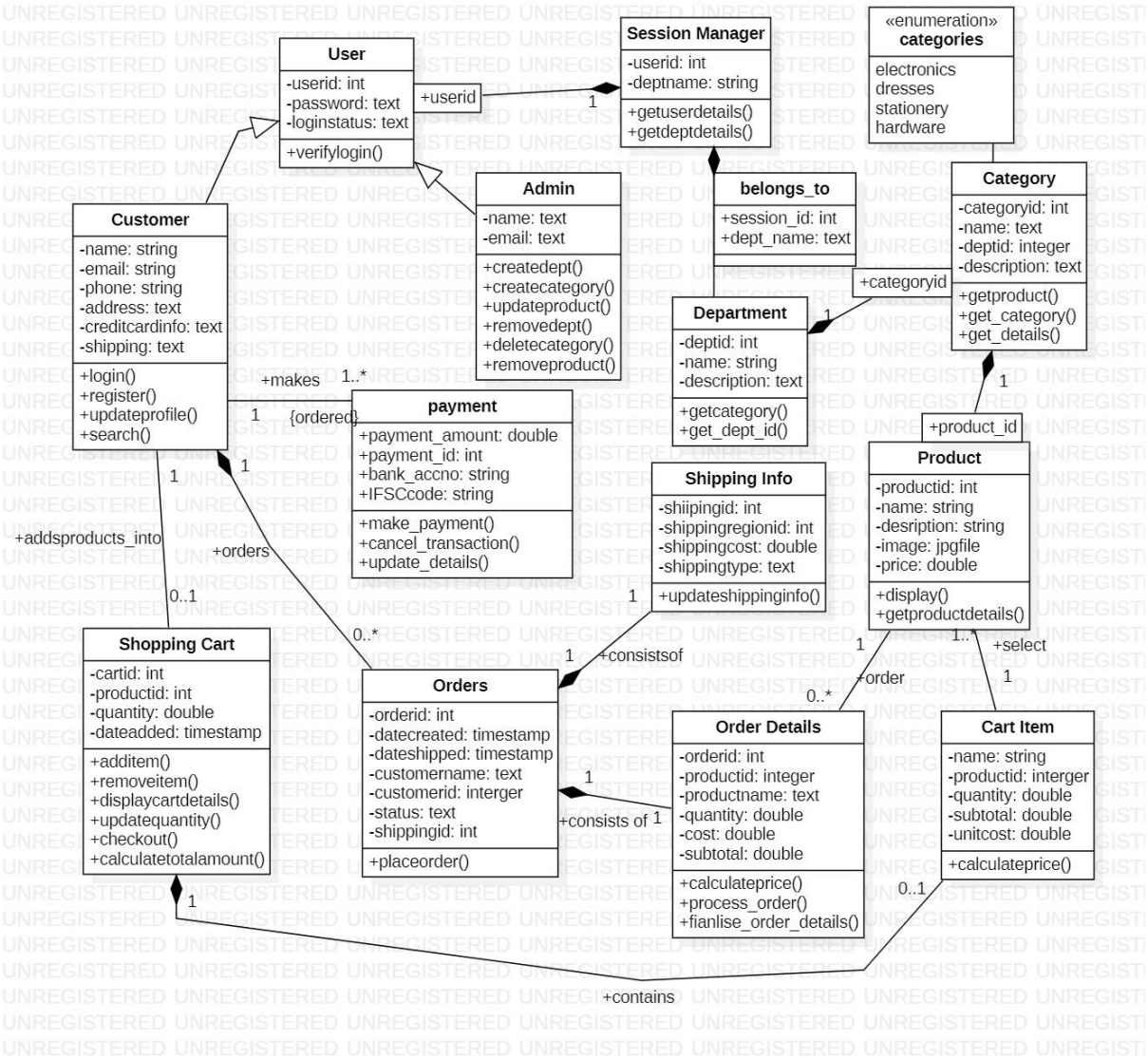

Fig 7.1 – Class Diagram for online shopping system

7.4 State Diagram

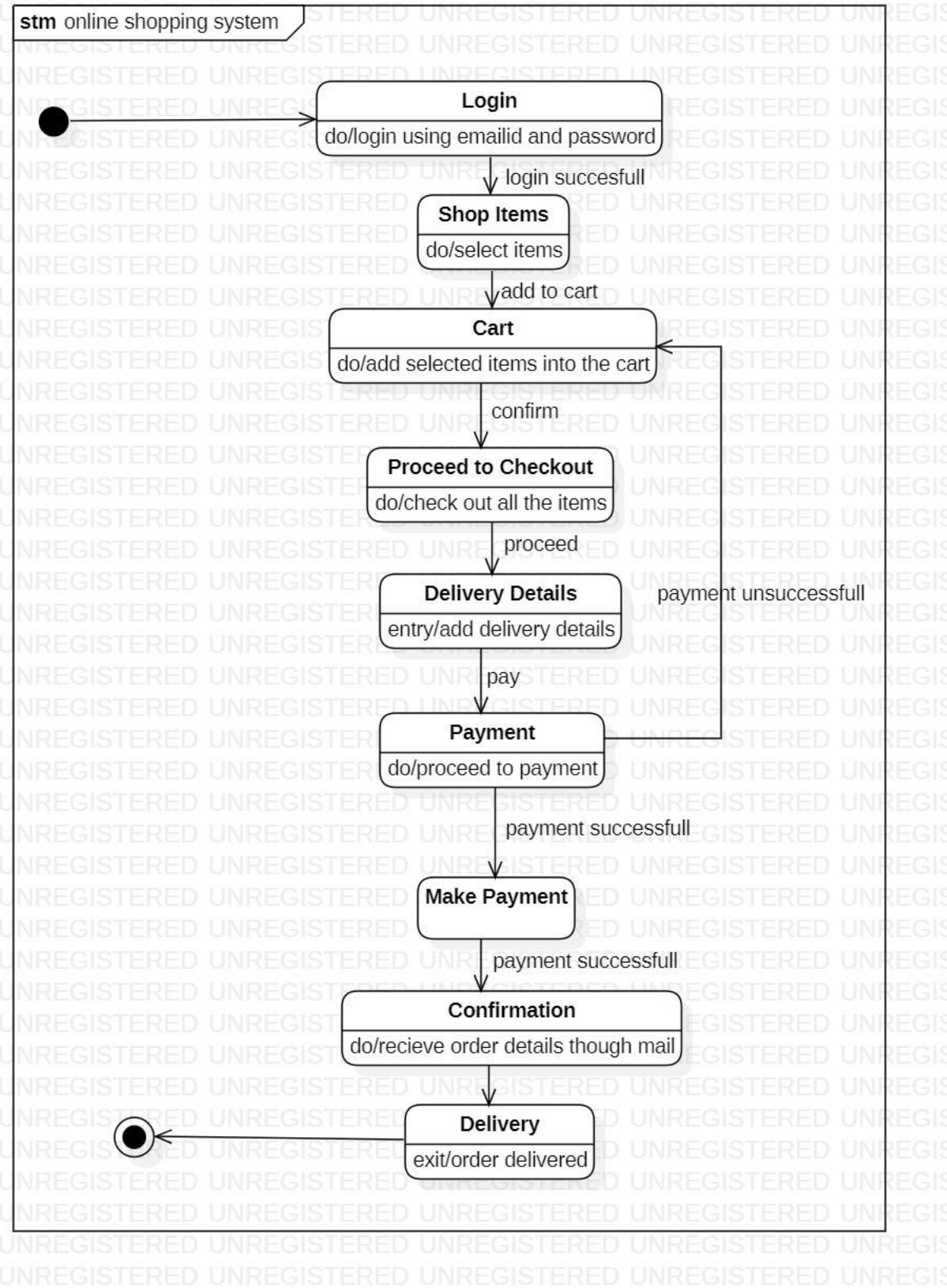

Fig 7.2 – State Diagram for online shopping system

7.5 Interaction Diagram

7.5.1 Use Case Diagram

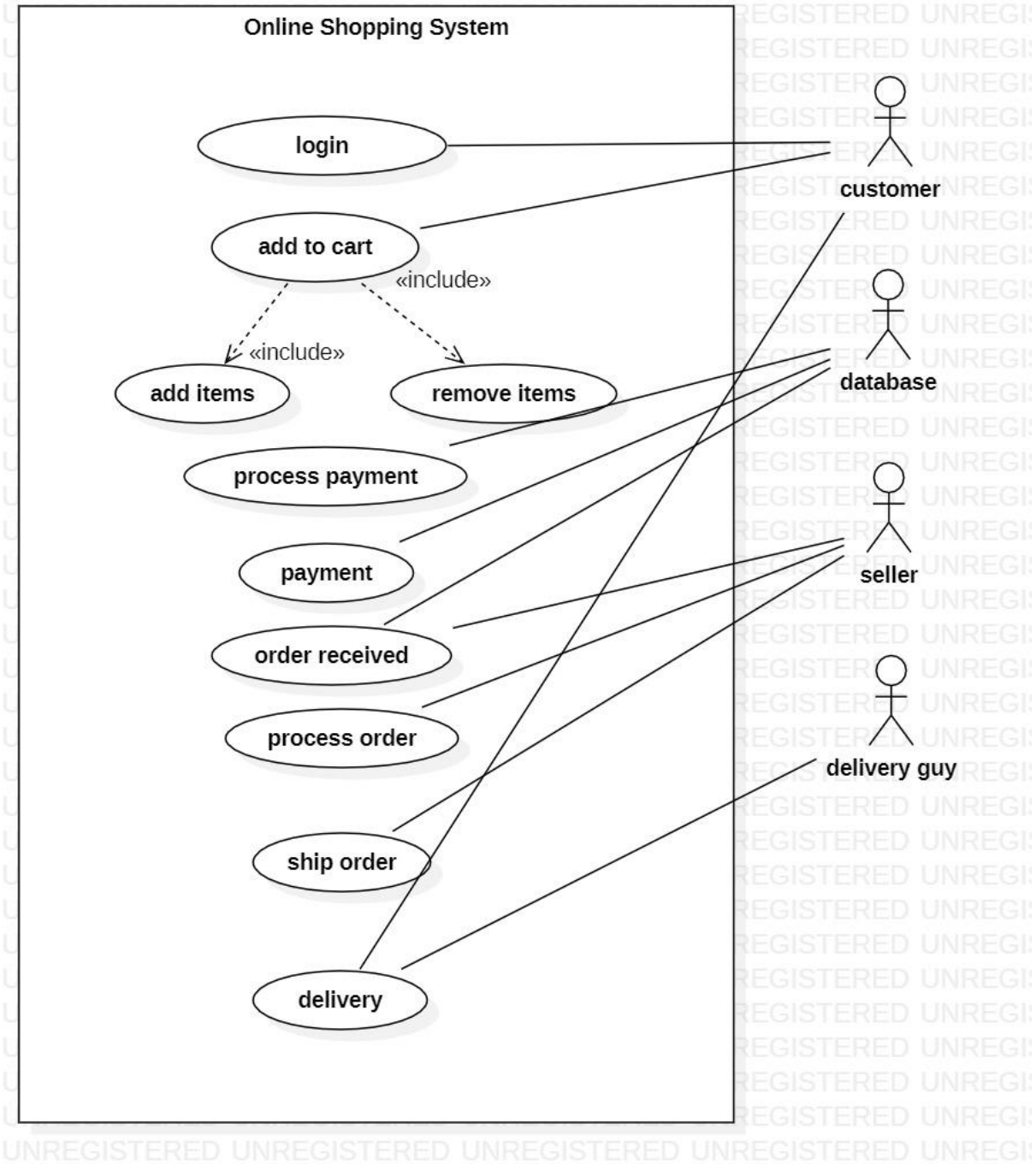

Fig 7.3 – Use Case Diagram for online shopping system

7.5.2 Sequence Diagram

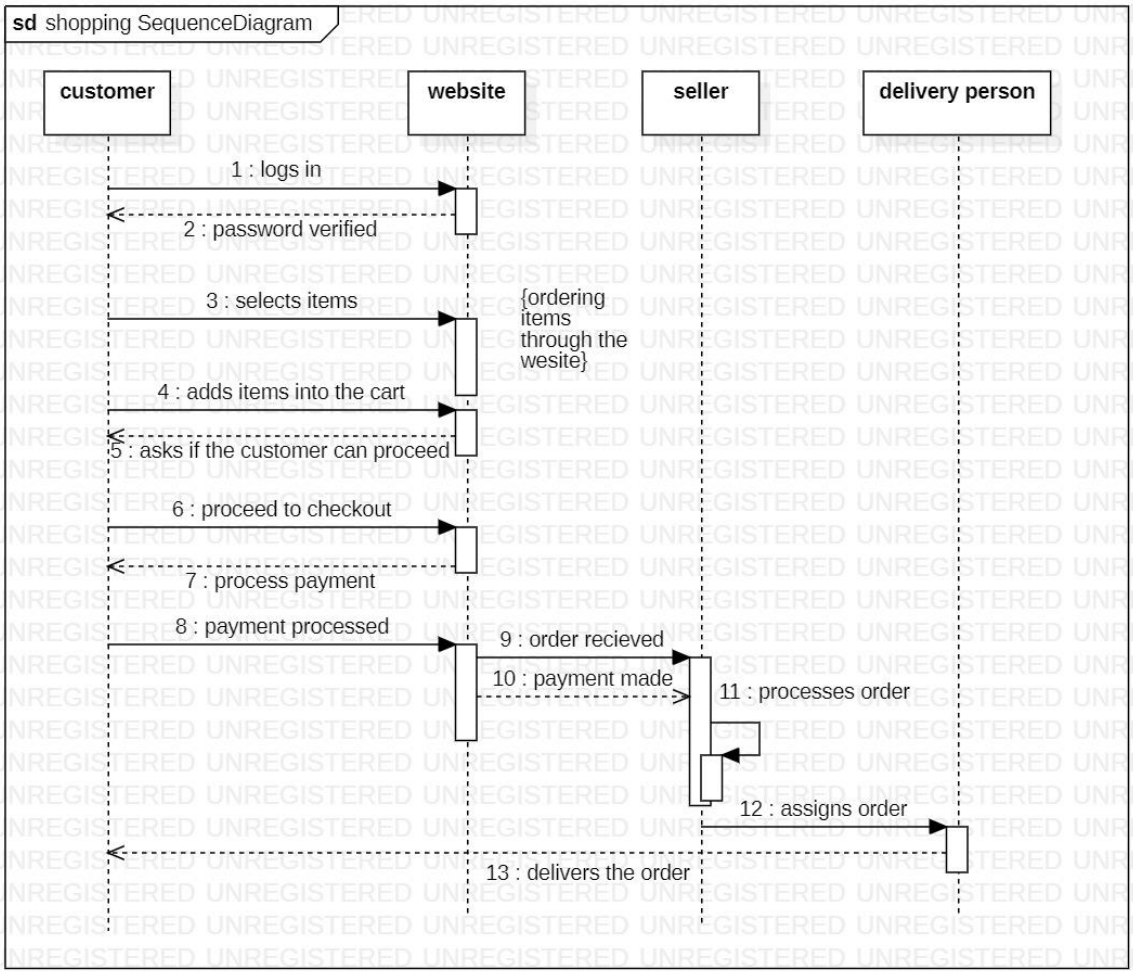

Fig 7.4 – Sequence Diagram for online shopping system

7.5.3 Activity Diagram

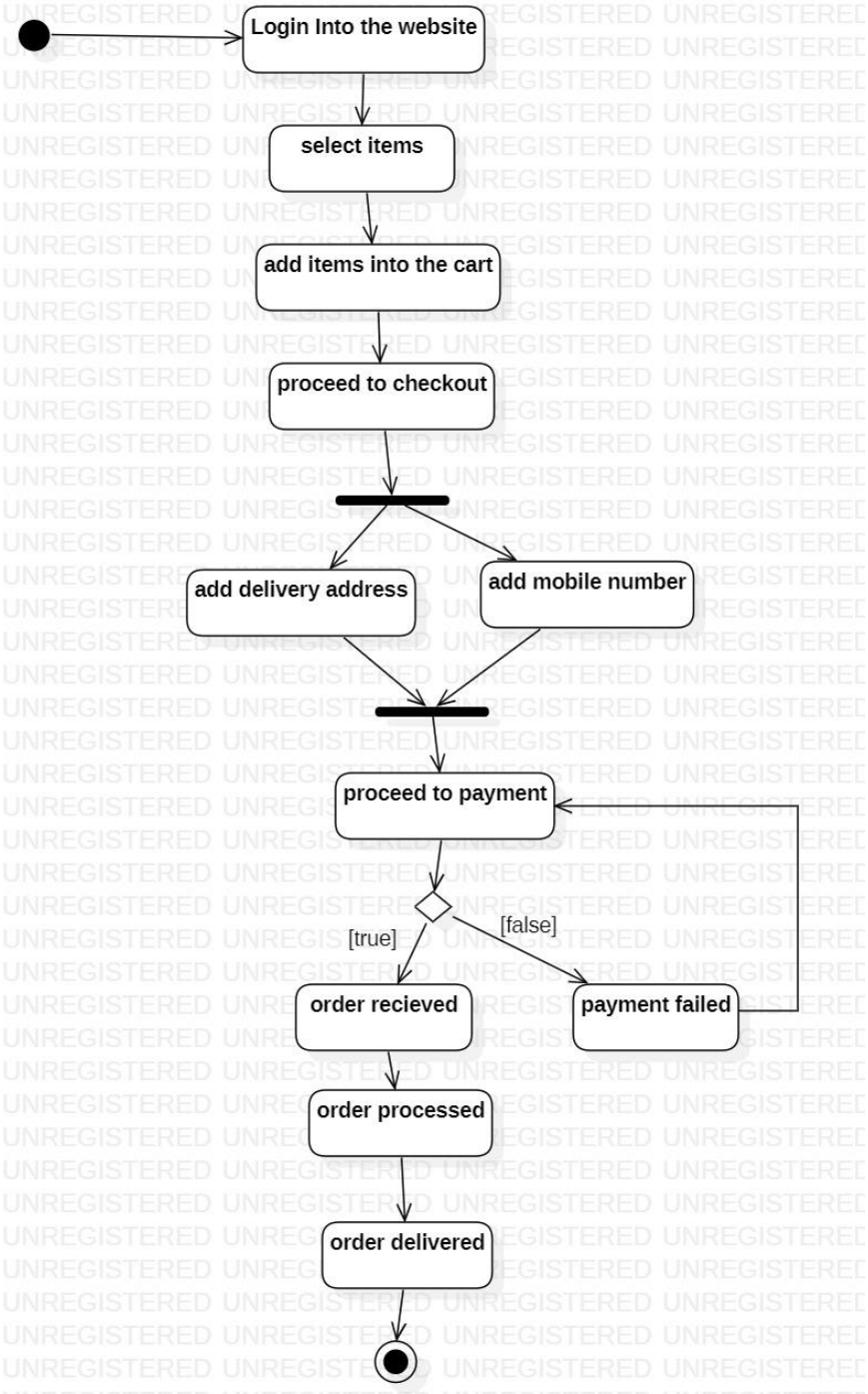

Fig 7.5 – Activity Diagram for online shopping system