Software Modelling and Design

Cinema Management System Requirements Specifications

Version 2.0

May 24, 2024

Contributors:

-Rexhens Balla

-Suljon Pashaj

-Rosilda Bajrami

-Ledion Lamce

-Jord Thanasi

-Juna Dako

-Kevi Koro

1.1 Project overview

Our project is a cinema application, designed to optimize, simplify and enhance the cinema experience for both clients, as well as the working team. The app would be used by 4 distinct user groups, those being the administrator, managers, staff, and clients, with different functionalities for each access level, and tools given to them to provide the best possible experience.

1.2 Purpose and scope :

Purpose :

The purpose of the requirement specification outline is to outline the functional and non functional requirements for the app, serving as a comprehensive guide to it's development and use. It outlines the features, access levels, specific functionalities needed, etc, to make sure that everything is understood and planned out as perfectly as possible.

In scope :

- User access levels and permissions:

Outlining and defining the 4 distinct access levels of admin, manager, staff and client, as well as their functionalities and use cases.

- Admin functionalities:

Managing user data, cinema schedules, comprehensive overview of financial and performance information, etc.

-Manager functionalities:

Managing staff schedule, tracking attendance, reviewing financial and performance information for their specific location, modifying the movie listing, etc.

-Staff functionalities:
Selling tickets, processing reservations, handling user inquiries, etc.

-Client functionalities:

Booking tickets, pre ordering snacks, viewing current and upcoming movies, tracking upcoming movies, participating in loyalty programs, etc.

-User interface and experience:

A comprehensive design with easy navigation that’s simple to understand for all age ranges, available on both mobile and web platforms

-Data management:

Secure handling of user and financial data, as well as comprehensive overviews of them for the admin and managers.

-Payment methods:

Allowing the user to pay inapp or in person for their tickets or snacks

-Notification system:

Notifying the user for their upcoming movies they’re tracking, loyalty program updates, etc.

Out of scope :

-Third party integration

-Marketing campaign management

-A movie review system

-etc.

2.1 Product context

The cinema ap is designed to be an independent and self-contained app, providing it’s comprehensive functionalities in order to enhance user experience without the use of any third-party app. It has it’s own database and management system which contain the movies, financial information, user information, schedules, etc

payment gateways which facilitates secure transactions online using popular methods such as paying via paypal or credit/debit card, as well as a system to help the client prebook a ticket and pay for it at the cinema

notification services which notify the client for booking reminders, movies they’re tracking, loyalty program updates, etc

loyalty program system encourages the user to use the cinema services more to earn points, tracks those points, and then gives the user rewards based on them.

All of these systems connect with the app to make it have it’s full functionality.

2.2 User characteristics

-Admin:

Role : Owner

Experience: Extensive experience in cinema management and business operations
Technical expertise: Moderate, must be familiar with business software and financial tools

Characteristics: The decisionmaker overseeing multiple locations, focused on planning and financial performance

-Manager:

Role : Manages a specific cinema location

Experience: Significant experience in cinema operations and staff management
Technical expertise: Moderate, must be comfortable with scheduling software and basic financial tools

Characteristics: Focuses on overseeing and managing the operations of specific locations, and responsible for their financial performance as well as the staff.

-Staff:

Role : Frontline employees handling the day to day operations

Experience: Changes depending on the task of the employee.
Technical expertise: Basic, must be familiar with basic customer service tools and ticketing systems

Characteristics: Handling customers first-hand, including their purchases, inqueries, etc.

-Client:

Role : Moviergoers

Experience: None
Technical expertise: Extremely basic. Everything will be laid out in the most easy to understand way possible.

Characteristics: Interested in the movies and booking tickets and provided with the appropriate tools to make that as easy and optimal as possible.

2.3 Assumptions

Equipment availability : It’s assumed that all users have access to devices such as phones, tablets or computers with internet access

Operating Systems: The user’s device supports the application. These supported devices are running iOS, Android, or web browsers.

User expertise: A basic level of technical proficiency is assumed, appropriate to the needed expertise depending on the user type.

Data Security: The implementation of security measures to protect user data, transaction data and financial information is assumed.

Payment options: It’s assumed that the user has access to at least one of the payment options offered by the app.

2.4 Constraints

Hardware Limitations: Compatibility with existing hardware like ticket kiosks, digital screens, and POS systems.

Integration: Ability to integrate with third-party systems such as payment gateways, loyalty programs, and movie distributors' databases.

Performance: High performance to handle peak loads during popular movie releases and special events.

Booking System: Robust and intuitive booking system for both online and offline ticket purchases.

Show Scheduling: Flexible scheduling system to manage multiple screenings across different times and screens.

2.5 Dependencies

Databases: Reliable database systems (e.g., MySQL) for managing user data, booking records, and inventory.

Web Servers: Dependence on web servers like Apache or cloud-based solutions for hosting online booking systems.

SMS/Email Services: Services like Twilio, SendGrid, or SMTP servers for sending booking confirmations, notifications, and marketing communications.

Ticket Printers: Compatibility with specific ticket printing hardware for issuing physical tickets.

Internet Connectivity: Reliable internet connection for online functionalities such as booking, payment processing, and data synchronization.

3.1 Functional Requirements

Admin Requirements

Req#	Requirement	Comments	Priority	Date	SME
001	The system shall allow the admin to add new users.	This lets the admin easily add new users to the system.	1	22-Mar-2024	Rexhens Balla
002	The system shall allow the admin to delete existing users.	Admins need to be able to remove users who no longer need access.	1	21-Mar-2024	Rexhens Balla
003	The system shall allow the admin to update user information.	Admins may need to modify user details for accuracy or changes.	2	23-Mar-2024	Rexhens Balla
004	The system shall allow the admin to view a list of all users.	Admins should have access to an overview of all users in the system.	2	19-Mar-2024	Rexhens Balla
005	The system shall allow the admin to grant specific permissions to users.	Admins should have control over user access levels and permissions.	1	22-Mar-2024	Suljon Pashaj

006	The system shall allow the admin to revoke specific permissions from users.	Admins should be able to modify user permissions as needed.	1	22-Mar-2024	Suljon Pashaj
007	The system shall provide a user interface for the admin to manage user permissions.	Admins should have a user-friendly interface for managing permissions.	2	23-Mar-2024	Suljon Pashaj
008	The system shall allow the admin to generate financial reports for each cinema.	Admins need access to detailed financial data for decision-making.	1	20-Mar-2024	Suljon Pashaj
009	The system shall allow the admin to specify the period for which the financial report is generated.	Admins should be able to customize the timeframe for reports.	2	21-Mar-2024	Suljon Pashaj
010	The system shall provide detailed financial data, including revenue, expenses, and profit.	Admins need comprehensive financial information for analysis.	1	24-Mar-2024	Suljon Pashaj

Client Requirements

Req#	Requirement	Comments	Priority	Date	SME
011	The system shall allow clients to pay for tickets and purchases using the app.	Clients should have the convenience of making payments digitally.	1	22-Mar-2024	Ledion Lamce
012	The system shall allow the admin to send financial feedback to managers.	Admins need to communicate financial insights effectively to managers.	2	24-Mar-2024	Suljon Pashaj
013	The system shall provide a user interface for the admin to enter feedback and suggestions for budgeting and pricing.	Admins should have an interface for inputting feedback and suggestions.	2	24-Mar-2024	Suljon Pashaj
017	The system shall allow clients to provide feedback on the financial performance of each cinema.	Admins should be able to offer insights and suggestions for improvement.	1	24-Mar-2024	Suljon Pashaj
018	The system shall process payment information securely.	Payment transactions must be handled with high security standards.	1	22-Mar-2024	Ledion Lamce
019	The system shall send booking information to the staff for processing.	Staff should receive details of client bookings promptly.	2	24-Mar-2024	Ledion Lamce

020	The system shall allow clients to preorder snacks and beverages through the app.	Clients should have the option to preorder refreshments for their movie experience.	2	24-Mar-2024	Ledion Lamce
021	The system shall provide a pickup confirmation for clients who preorder snacks.	Clients should receive confirmation of their snack orders.	1	20-Mar-2024	Ledion Lamce
022	The system shall send notifications to clients about upcoming movies and updates.	Clients should stay informed about new releases and changes.	1	22-Mar-2024	Rosilda Bajrami
023	The system shall allow clients to manage their notification preferences.	Clients should have control over the types of notifications they receive.	2	22-Mar-2024	Rosilda Bajrami
024	The system shall allow clients to enroll in cinema loyalty programs through the app.	Clients should have the option to join loyalty programs for rewards.	1	19-Mar-2024	Rosilda Bajrami
025	The system shall collect necessary information such as name, email, and phone number for loyalty program enrollment.	Required details for loyalty program membership.	1	19-Mar-2024	Rosilda Bajrami
026	The system shall allow clients to claim benefits and rewards from loyalty programs.	Clients should be able to redeem rewards earned through the loyalty program.	2	23-Mar-2024	Rosilda Bajrami
027	The system shall provide access to exclusive discounts and promotions for loyalty program members.	Loyalty program members should have access to special offers.	1	23-Mar-2024	Rosilda Bajrami

028	The system shall allow clients to view and apply ongoing promotions.	Clients should be able to see available discounts and promotions.	2	21-Mar-2024	Rosilda Bajrami
029	The system shall provide a feature for clients to contact customer support within the app.	Clients should have access to assistance when needed.	2	20-Mar-2024	Rosilda Bajrami
030	The system shall allow clients to chat with support staff in real-time.	Real-time communication for quick issue resolution.	1	20-Mar-2024	Rosilda Bajrami
031	The system shall provide updates on support requests made by clients.	Clients should be informed about the status of their support inquiries.	1	21-Mar-2024	Rosilda Bajrami

Manager Requirements

Req#	Requirement	Comments	Priority	Date	SME
032	The system shall allow managers to track the revenue of their cinema.	Managers should have access to revenue data for performance evaluation.	1	22-Mar-2024	Jord Thanasi
033	The system shall provide detailed revenue reports for managers.	Detailed reports for informed decision-making by managers.	1	22-Mar-2024	Jord Thanasi
034	The system shall allow managers to track the income generated by each staff member.	Tracking staff performance and contributions to revenue.	2	24-Mar-2024	Jord Thanasi

035	The system shall allow managers to track the spending of each staff member.	Monitoring expenses and ensuring financial efficiency.	2	24-Mar-2024	Jord Thanasi
036	The system shall allow managers to add new films to the cinema catalog.	Updating the catalog with new releases and additions.	1	20-Mar-2024	Jord Thanasi
037	The system shall collect and store information such as the film title, genre, director, and release date.	Required details for maintaining an accurate catalog.	1	22-Mar-2024	Jord Thanasi
038	The system shall allow managers to delete films from the cinema catalog.	Removing outdated or no longer relevant films from the catalog.	2	23-Mar-2024	Jord Thanasi
039	The system shall allow managers to modify information about existing films in the cinema catalog.	Updating film details for accuracy and relevance.	2	19-Mar-2024	Juna Dako
040	The system shall authenticate users attempting to log in.	Ensuring secure access to the system for authorized personnel.	1	21-Mar-2024	Juna Dako
041	The system shall ensure only authorized users can access specific features.	Implementing role-based access control for data security.	1	21-Mar-2024	Juna Dako
042	The system shall allow users to access management features via a mobile application.	Providing flexibility and convenience for managing operations.	2	23-Mar-2024	Juna Dako

043	The system shall allow managers to create and edit staff schedules.	Efficient allocation of shifts considering employee availability and business needs.	1	20-Mar-2024	Juna Dako

Staff Requirements

Req#	Requirement	Comments	Priority	Date	SME
044	The system shall allow staff to sell tickets to customers.	Staff should be able to facilitate ticket purchases efficiently.	1	22-Mar-2024	Kevi Koro
045	The system shall process and record ticket sales transactions.	Recording sales for accurate tracking and reporting.	1	22-Mar-2024	Kevi Koro
046	The system shall allow staff to handle customer inquiries and assistance requests.	Assisting customers promptly for a positive experience.	2	24-Mar-2024	Kevi Koro
047	The system shall provide a user interface for staff to respond to inquiries.	Ensuring staff can address customer needs effectively.	2	24-Mar-2024	Kevi Koro
048	The system shall allow staff to check the validity of tickets presented by customers.	Ensuring only valid tickets are accepted for entry.	1	20-Mar-2024	Rexhens Balla
049	The system shall allow staff to report technical issues encountered with cinema equipment or software.	Ensuring prompt resolution of technical problems.	1	22-Mar-2024	Rexhens Balla

050	The system shall track and manage reported technical issues.	Monitoring reported issues for timely resolution.	2	22-Mar-2024	Rexhens Balla
051	The system shall allow staff to make transactions that include tickets and snacks.	Facilitating purchases of tickets and concessions in a single transaction.	1	19-Mar-2024	Rexhens Balla

3.2 Non-Functional Requirements

3.2.1 Product Requirements

- The Cinema Management System shall be available to all cinemas during normal working hours (Mon–Fri, 08:30–21:30).
- The MHC-PMS shall be available to all clinics during normal working hours (Mon–Fri, 0830–17.30).
- The system must ensure secure staff login: Secure login for staff members with role-based access control.
- The system must support online payment for a transaction made by a client.

3.2.1.1 User Interface Requirements

Customer Interface

In the Home Screen the system must display currently showing movies with posters, titles, and brief descriptions, highlighting upcoming movies and special events.
In the Movie Details Page the system must show detailed information about the movie, including synopsis, cast, crew, duration, genre, and rating and display available showtimes and theater locations. It shall allow customers to choose preferred showtime and theater.
Visual seat map for seat selection, indicating available, reserved, and occupied seats.
Step-by-step processes guiding the user from movie selection to payment must be supported.

Options to select ticket types (adult, child, senior) and apply discounts or promo codes.
Regarding snacks, the system must display a menu of available snacks and beverages with images, descriptions, and prices.

Staff member Interface

Secure login with username and password must be provided. Role-based access control to ensure appropriate permissions by different user levels.
Regarding the dashboard,Overview of daily transactions, upcoming showtimes, and inventory status must be displayed on the homepage of the staff member.
In the Ticket Section Page, Interface for scanning tickets using a barcode scanner or manual ticket ID entry must be provided. Display ticket validity status and details (movie, showtime, seat number) is another core functionality.
In the Transaction Management page, initiating new transactions composed of tickets and snacks is one of the most important functions of the staff member. Process payments and generate receipts must also be provided after finishing the transaction.

Manager Interface

The main screen of the manager must be composed of dashboards that will contain an overview of daily transactions, upcoming showtimes, and inventory status must be displayed on the homepage of the staff member.
In the Transaction Management Page initiating Transactions feature includes: Initiating new transactions composed of tickets and snacks is one of the most important functions of the staff member.
Meanwhile in the Movie section page, adding a new movie, editing an existing movie details or removing it from the list when it won't be displayed anymore shall be provided to the manager.
There shall be a dedicated page about managing employees that will provide the manager with all the information he needs in order for him to keep track of each employee he surveys.

Admin Interface

In the homepage of the admin the Admin Dashboard must be integrated and must be notifying him of an overview of system performance, sales statistics, and user activity.
In the User Management Section the full management functionalities for every employee of the cinema management system containing: Create, update, and delete staff accounts with role assignments. It will also give or restrict certain permissions to different user levels.
Monitor staff activity and performance metrics.
In the Reports and Analytics section the system shall generate detailed reports on sales, inventory, customer behavior, and system usage. Export options for various formats (PDF, Excel).

3.2.1.2 Usability

Usability of the cinema management software refers to the extent to which the product can be used to efficiently achieve the goals of the intended audience. Key points defining the usability of the software include:

Learnability: The software must be easy to learn by all types of users, including staff members, managers, and customers. Clear instructions for each command should be provided in the different views of the user interfaces, making it intuitive for all users to navigate and use the system effectively.

Accessibility: The software must be accessible remotely, allowing staff and customers to use it from various devices. The application should be available for both PC and mobile use, ensuring that users can access the system from anywhere at any time.

Responsiveness: The software must be highly responsive, successfully responding to user requests in real time. It should operate and update data in real time for both staff and customer interactions, ensuring a seamless experience.

Flexibility: The program should be flexible enough to accommodate different user needs and preferences. It will allow customization of certain parts, such as setting or changing passwords, and adding or updating movie and snack inventory. The developers should also be able to maintain and update the program as needed.

Effectiveness: The software must be simple and easy to use for all users, including admin, staff, managers, and clients. This will make it practical and effective in managing cinema operations, digitizing processes, and enhancing the customer experience.

Efficiency: Users should find it easy to complete tasks for example making a transaction for a certain movie, with straightforward commands and instructions, even if it is their first time using the application or they haven't used it in a while. The interface should facilitate quick and efficient task completion.

Familiarization: While staff and managers may need a bit more time to become completely familiar with all the software's capabilities, a week of use should be sufficient for them to become skilled in using the application. This ensures that administrators can efficiently manage all aspects of cinema operations.

By focusing on these usability aspects, the cinema management software will provide an efficient, user-friendly experience for all users, enhancing overall operational efficiency and customer satisfaction.

3.2.1.3 Efficiency

The efficiency of the software goes beyond merely completing assigned tasks; it must do so in the optimal manner, considering both space complexity and time complexity. Space complexity ensures the system completes tasks without wasting memory, while time complexity ensures tasks are performed as quickly as possible. Efficiency is determined by two main groups of requirements: performance and space requirements.

3.2.1.3.1 Performance Requirements

- **Response time:** the system should be able to respond to the user requests in real-time with no lag or delay.

- **Throughput:** the system should be able to handle a relatively high flow of data at a time as multiple clients may be using the application all at once. The program should be able to respond to every single request as soon as the command is given without compromising its speed or performance.

- **Compatibility:** the application should be compatible with different hardware such as computers or mobiles as long as they have a connection to the internet as most functions will require it. Regardless of the device the application is being used on, the performance and functionality should not be affected.

- **Latency:** Network latency for critical operations, such as ticket validation and payment processing, should not exceed 100 milliseconds.

3.2.1.3.1 Space Requirements

Web Availability: The application will be accessible via the web, eliminating the need for clients to download any software. This ensures ease of access and use from any device with an internet connection.

Mobile Application: The application will also be available as a mobile app. The mobile app will provide the same functionalities as the web version, offering a seamless user experience across different devices.

PDF Receipts: Transaction receipts downloaded as PDFs should be compressed and optimized to minimize their size. Users should have the option to download receipts immediately or receive them via email to reduce local storage impact.

Log Management: System logs should be managed to ensure they do not consume excessive storage space. Implement log rotation and retention policies to archive or delete old logs systematically.

3.2.1.4 Dependability

Dependability is critical for ensuring that the system is reliable, secure, and performs as expected under various conditions. Here are the key aspects of dependability:

1. Reliability

- Uptime: The software must be operational without interruption, especially during peak times like weekends and major movie releases.
- Error Handling: Robust error detection and handling mechanisms to ensure that failures do not lead to system crashes or data loss.
- Redundancy: Implementation of redundant systems and backups to mitigate the impact of hardware failures or other disruptions.

2. Availability

- High Availability Architecture: Use of load balancers, failover strategies, and distributed servers to ensure continuous availability.
- Maintenance Windows: Scheduling updates and maintenance during off-peak hours to minimize downtime and disruption to users.

3. Performance

- Response Time: Ensuring fast response times for user actions, especially for critical functions like booking tickets and processing payments.
- Resource Optimization: Efficient use of resources to maintain high performance without overloading the system.

4. Usability

- User-Friendly Interface: Designing an intuitive and easy-to-navigate interface to minimize user errors and enhance the overall user experience.
- Accessibility: Ensuring the software is accessible to users with disabilities by following accessibility standards and guidelines.

3.2.1.5 Security

Ensuring security involves implementing a comprehensive set of measures to protect the system from various threats, safeguard sensitive data, and maintain user trust. Here are the main security considerations for cinema software:

1. Data Protection

- Encryption: Use strong encryption algorithms (e.g., AES-256) to protect data at rest and in transit. Encrypt sensitive customer data, such as personal information and payment details.
- Secure Storage: Store sensitive data in secure databases with access controls and encryption.

2. Authentication and Authorization

- Secure Login: Implement secure authentication mechanisms, including multi-factor authentication for accessing the system.
- Role-Based Access Control: Restrict access to features and data based on user roles and responsibilities. Ensure that users only have access to the information necessary for their role.

3. Network Security

- Firewalls: Deploy firewalls to control incoming and outgoing traffic and protect against unauthorized access.
- VPNs: Implement virtual private networks for secure remote access to the cinema software.

4. Database Security

- Access Controls: Implement strict access controls and ensure that only authorized personnel can access the database.
- Regular Audits: Perform regular audits and monitoring of database activities to detect and respond to suspicious behavior.

5. Operational Security

- Incident Response Plan: Develop and maintain an incident response plan to quickly address and mitigate the impact of security breaches.
- Backup and Recovery: Regularly backup data and ensure that recovery procedures are in place to restore data in case of loss or corruption.

- Logging and Monitoring: Implement comprehensive logging and real-time monitoring to detect and respond to security incidents promptly.

3.2.2 Organizational Requirements

3.2.2.1 Environmental Requirements

- Internet Access:The web app requires a reliable and stable internet connection to function properly.
- Cross-Browser Support: The web application must be functional across all major web browsers to ensure wide accessibility.
- Device Accessibility: The web application should be accessible on various devices, including laptops, desktops, smartphones, and tablets.
- Technical Assistance:Any technical issues or bugs that impact the application's performance should be promptly addressed.
- User-Friendly Design:The application should have an intuitive and easy-to-navigate interface to enhance user engagement and satisfaction.

3.2.2.2 Operational Requirements

- User Authentication:Users should be able to create an account and log in using their email to access the app's features.
- Comprehensive Movie Catalog: The app should provide a detailed list of currently showing and upcoming movies, including information such as title, genre, duration, rating, and synopsis.
- Ticket Booking System: The app should enable users to book tickets for the movies they wish to watch, including seat selection and payment processing.
- Search and Filter:Users should be able to easily search and filter the movie listings by title, genre, release date, showtimes, and availability.
- Notification System:The app should notify users about their booking confirmations, showtime reminders, and any changes to the movie schedule.
- Customer Feedback:The app should allow users to provide feedback on their experience, including movie ratings and suggestions for improvement.
- Promotions and Offers:The app should display ongoing promotions, special deals, and discounts available for users, enhancing their movie-going experience.
- Food and Beverage Ordering: Users should be able to pre-order food and beverages through the app to pick up at the concession stand or have delivered to their seats.
- Loyalty Program:The app should include a loyalty program where users can earn points for each purchase, redeemable for discounts or free tickets.

- Customer Support: The app should provide easy access to customer support for resolving issues related to bookings, payments, and other inquiries.

3.2.2.3 Development Requirements

The development requirements for the cinema management system include using a robust technology stack (e.g., JavaScript, React, Node.js, MySQL), ensuring scalability and load balancing, integrating with external services and APIs, and implementing strong security measures for data protection and user authentication. The system must undergo comprehensive testing, including unit, integration, and user acceptance tests, and be supported by thorough technical and user documentation. Employing an agile project management methodology with clear milestones and deadlines is crucial. The system should be optimized for performance, include regular maintenance and updates, and feature an intuitive, accessible user interface to enhance the user experience.

3.2.3 External Requirements

3.2.3.1 Regulatory Requirements

- **Data Protection Compliance**: The system must comply with national and international data protection regulations such as the General Data Protection Regulation (GDPR) to safeguard staff personal data. This includes implementing features for data consent management, data anonymization processes, and secure data storage solutions.
- **Employment Laws**: The system must align with local employment laws regarding staff working hours, wages, and terms of employment. Features should include tracking work hours, managing leave requests, and automating salary calculations to ensure legal compliance.

3.2.3.2 Ethical Requirements

- **Privacy Ethics**: The system must uphold high ethical standards in handling staff data, ensuring the privacy and confidentiality of all staff members. This includes mechanisms for staff to control their personal data and opt out of non-essential data collection.

- **Fair Use and Accessibility**: The system should ensure equal access to all staff members, regardless of their role, seniority, or physical capabilities. This involves implementing adaptive technologies and user interface designs that accommodate a diverse workforce.

3.2.3.3 Legislative Requirements

- **Workplace Safety**: Compliance with occupational health and safety laws is critical. The system should support the management of health and safety records, incident reporting, and safety training logs, especially for staff operating in physical library spaces.
- **Intellectual Property Rights**: The system must respect and enforce copyright laws, particularly in handling digital content and multimedia resources. This includes features for copyright management, licensing agreements, and fair usage policies to prevent copyright infringement.

3.2.3.3.1 Accounting Requirements

- **Financial Reporting**: The system must support accurate and legally compliant financial reporting. Features should facilitate regular audits, budget tracking, and fiscal reporting related to library operations and staff-related financial management.
- **Transparency and Accountability**: The system must include mechanisms for transparent financial management accessible only to authorized personnel. This includes audit trails, role-based access controls, and detailed financial dashboards that provide real-time financial data.

3.2.3.3.2 Security Requirements

- **Physical Security**: Implement measures to secure physical assets of the library and safety of the staff. This should include access control systems to restrict and monitor entry into sensitive areas, and installation of surveillance systems throughout the library premises.
- **Cybersecurity**: Deploy robust cybersecurity measures to protect the library's digital infrastructure and sensitive staff data. This includes firewalls, intrusion detection systems, secure communication channels, and regular security audits to identify and mitigate threats.

4. SOFTWARE DESIGN

4.1 User Scenarios

1. ADMIN CASES

Use Case Name	**UC001 -** Managing users
Summary	This use case describes a set of actions that the admin of the system can perform in relation with user management.
Dependency	**Admin UC** - Add a new user **Admin UC** - Edit an existing user **Admin UC** - Remove an existing user
Actors	**Primary actor:** Admin **Secondary actor**: User being affected
Preconditions	1. The admin has been logged in successfully.
Description of main sequence	1. Admin navigates to the user management section. 2. Admin selects the user level he wants to make changes at(client, staff, manager). 3. Admin selects the action he wants to perform at this specific user. 4. Admin adds, edits or removes information regarding this user. 5. Admin saves changes.
Description of the alternative sequence	**Invalid input detected** 1. Admin navigates to the user management section. 2. Admin selects the user level he wants to make changes at(client, staff, manager). 3. Admin selects the action he wants to perform at this specific user. 4. Admin modifies or adds user details but the data is invalid. 5. Upon attempting to save changes, the system validates the input and detects the invalid data. 6. The system displays the appropriate message telling exactly where error is being displayed. 7. Admin is prompted to correct the data before saving the changes again.

Non functional requirements	**Performance:** The system saves the changes immediately **Security:** The admin should be the only one to change these permissions **Usability:** The user interface should be easy to understand and not overwhelming
Postconditions	System reflects the changes made by the admin, including newly created users, edited users and removed users.

Use Case Name	**UC002: Changing Specific User Permissions**
Summary	This use case describes the ability of the admin to give unique permission to users who originally would not have access to such abilities or restrict them from abilities they would originally have
Dependency	**Admin UC:** Add Users **Admin UC:** Manage Users
Actors	**Primary actor:** Admin **Secondary actor:** Manager, Staff, Client
Preconditions	1. The admin has been logged in successfully
Description of main sequence	2. The admin navigates to the “user management” section in his panel 3. The admin selects the option to “edit user information” 4. The admin chooses the user they want to edit 5. The admin selects which permissions they want the user to have or not have 6. The admin saves changes 7. If the changes save correctly, the admin gets a message
Description of the alternative sequence	1. The admin navigates to the “user management” section in his panel 2. The admin selects the option to “edit user information” 3. The admin chooses the user they

	want to edit 4. The admin selects which permissions they want the user to have or not have 5. The admin saves changes 6. If the changes fail to save, the system gives the admin a message, and gives him the option to retry or to cancel the changes 7. If the changes save correctly, the admin gets a message
Non functional requirements	**Performance:** The system saves the changes immediately **Security:** The admin should be the only one to change these permissions **Usability:** The user interface should be easy to understand and not overwhelming
Postconditions	After choosing to go back, the admin should be sent back to the financial reports section

Use Case Name	**UC003: View Financial Reports**
Summary	This use case describes the ability of the admin to access a detailed report of the finances of each cinema during any specific period of their liking.
Dependency	**Staff UC:** Ticket selling **Client UC:** Booking Tickets **Admin UC:** Add Users **Admin UC:** Manage Users **Manager UC:** Staff Spending
Actors	**Primary actor:** Admin
Preconditions	1. The admin has been logged in successfully
Description of main sequence	1. The admin navigates to the “financial management” section in his panel 2. The admin selects the option to view financial reports 3. The admin chooses which cinema/s they want a report on 4. The admin chooses the period for which they want a revenue report

	5. The system retrieves the financial data and displays it to the admin
Description of the alternative sequence	1. The admin navigates to the “financial management” section in his panel 2. The admin selects the option to view financial reports 3. The admin chooses which cinema/s they want a report on 4. If the admin enters an invalid period the application will let him know that he should add a valid one 5. The admin enters a valid period 6. The system retrieves the financial data and displays it to the admin
Non functional requirements	**Performance:** The system generates the financial report quickly **Security:** The reports should be only visible to the admin **Usability:** The information should be presented in a way that is both detailed and easy to understand
Postconditions	After choosing to go back, the admin should be sent back to the financial reports section

Use Case Name	**UC004: Financial Feedback**
Summary	This use case enables the admin to give feedback to the managers regarding their cinema’s performance and give them information for budgeting, price changes and plans for the future
Dependency	None
Actors	**Primary actor:** Admin **Secondary actor:** Manager
Preconditions	1. The admin has been logged in successfully 2. The admin has up to date reports on the financial situation of the cinema/s 3. The manager has the permission to get feedback from the admin
Description of main sequence	1. The admin navigates to the “financial

	management" section in his panel 2. The admin selects the option to give feedback to the managers 3. The admin chooses which cinema/s they want to give feedback to 4. The admin chooses which manager to give feedback to 5. The admin writes their feedback 6. The admin sends the feedback
Description of the alternative sequence	1. The admin navigates to the "financial management" section in his panel 2. The admin selects the option to give feedback to the managers 3. The admin chooses which cinema/s they want to give feedback to 4. The admin chooses which manager to give feedback to 5. The admin writes their feedback 6. If the admin encounters a loss of connection or system crash while writing their message, it will get saved as a draft
Non functional requirements	**Performance:** The system sends the feedback fast **Security:** The feedback is sent only to the managers the admin chooses and no one else can read them **Reliability:** Feedback data should be accurately captured, stored, and transmitted without loss or corruption.
Postconditions	The app should let the admin know that the feedback message was sent, and then send him back to the "financial management" section

Use Case Name	**UC005: Managing Cinema Facilities**
Summary	This use case describes the process for the cinema owner to manage and maintain the facilities of their cinemas.
Dependency	None
Actors	**Primary actor:** Admin

Preconditions	1. The admin has been logged in successfully
Description of main sequence	1. The admin navigates to the "Facilities Management" section in their dashboard. 2. The admin views a list of all facilities within their cinemas (e.g., screens, seats, sound systems, projectors). 3. The admin selects a specific facility to manage. 4. The admin can perform actions such as scheduling maintenance, repairs, or upgrades for the selected facility. 5. The admin confirms the changes and updates the facility status.
Description of the alternative sequence	1. The admin navigates to the "Facilities Management" section in their dashboard. 2. The admin selects a specific cinema to manage. 3. The admin views a list of all facilities within their cinemas (e.g., screens, seats, sound systems, projectors). 4. The admin can perform actions such as scheduling maintenance, repairs, or upgrades for the selected facility. 5. If a facility requires immediate attention (e.g., equipment breakdown), the admin can expedite the maintenance process by marking it as urgent. 6. If there are budget constraints or resource limitations, the admin can prioritize maintenance tasks based on criticality. 7. The admin confirms the changes and updates the facility status.
Non functional requirements	**Performance:** The system should provide real-time updates on facility status and maintenance schedules. **Usability:** The facilities management interface should be intuitive and easy to navigate. **Reliability:** The system should accurately track maintenance activities and ensure timely resolutions.

Postconditions	After choosing to go back, the admin should be sent back to the financial reports section

2. CLIENT CASES

UC Name	UC006: View movies and their information
Summary	*The client should be able to browse all currently showing movies, or movies that are confirmed to be available in the future using the app.*
Dependency	None.
Actors	The primary actor in this case is the client.
Preconditions	The client must have the app downloaded. The app must have a list of movies that it's displaying along with available information on each movie
Description of the Main Sequence	● *Step 1: The client opens the app* ● *Step 2: They click on "Currently showing"* ● *Step 3: They browse all the current movies that are showing, finding whichever one interests them* ● *Step 4: They click the movie that interests them and see it's information*
Description of the Alternative Sequence	● *Step 1: The client opens the app* ● *Step 2: They click on "Showing soon"*

	● *Step 3: They browse all the movies that are confirmed to be showing in cinemas in the future, finding whichever one interests them.*
Non functional requirements	*The user interface must respond quickly to the user input, within a second.* *The display when browsing through the movies must be optimised for both, and change depending on if the user is viewing on landscape or portrait mode* *The system should be able to handle letting the user scroll through multiple movies at a time* *Clicking on a movie should take the user to it's page quickly.*
Postconditions	*The client is able to see all the movies that are currently screening or that are confirmed to be screening in the future*

UC Name	UC007: Book tickets in app
Summary	The client should be able to book tickets using the app, selecting their prefered seating, screening and cinema. This information then is sent to the staff through the app.
Dependency	UC006 : View Movies
Actors	The primary actor in this case is the client who sends the information through the app. The secondary actor is the staff member who receives the information.

Preconditions	The client must have the app downloaded. The app must have a listing of all current movies, their times of showing, the seating, and the prices. The client must select the movie they want to watch.
Description of the Main Sequence	● Step 1: The client opens the app ● Step 2: They find the movie they're interested in ● Step 3: They're able to see all the available dates, times, cinemas and seats for the movie they want. ● Step 4: They press "book ticket" and are sent to the ticket booking screen.
Description of the Alternative Sequence	● Step 1: The client opens the app ● Step 2: They find the movie they're interested in ● Step 3: They're able to see all the available dates, times, cinemas and seats for the movie they want. ● Step 4: They try to book a ticket, but are met with a screen telling them that they're seat was already taken, while showing them all other available seats they can choose.
Non functional requirements	The user interface must respond quickly to the user input. The information on the app must always be up to date. Everything should be as easy and intuitive as possible. The seating information must be updated as quickly as possible when there's a change, so 2 clients don't book the same seat
Postconditions	The client is able to select the movie going experience they want, and book a ticket.

UC Name	UC008: Pay in app
Summary	The client should be able to pay for everything using the app, and then this information is processed in the app, and if the purchase is successful, the information about what was purchased should be sent to the staff, and the payment info should be sent to finance management.
Dependency	UC006 : View movies UC007 : Book tickets
Actors	The primary actor is the client, who makes the purchase, and the secondary actors are the staff who receive the information about the purchase, and finance management who gets the information on the payment.
Preconditions	The client must have the app downloaded. The app must have a listing of all current movies, their times of showing, the seating, and the prices. The client must select the movie they want to watch.
Description of the Main Sequence	● Step 1: The client opens the app ● Step 2: They find the movie they're interested in ● Step 3: They select the date, time, cinema, seat for the movie ● Step 4 : They press "Make purchase" ● Step 5 : They enter their credit card information, completing the transaction
Description of the Alternative Sequence	● Step 1: The client opens the app ● Step 2: They find the movie they're interested in ● Step 3: They're select the date, time and cinema for the movie ● Step 4 : They press "Pay in theatre"

Non functional requirements	The user interface must respond quickly to the user input. The app must accept purchases in all popular modes of online payment (paypal, mastercard, visa, etc) The payment process must be easy and intuative. Everything must be secure, insuring that the payment process is as safe for the client as possible.
Postconditions	The client is able to book a ticket through the app, either paying online, or in person (in which case they cant reserve a seat)

UC Name	UC009: Preorder snacks in app
Summary	The client should be able to preorder the snacks, beverages or both that they want, and then pick them up from a staff member at the cinema.
Dependency	UC006 : View movies UC007 : Book tickets UC008 : Pay in app
Actors	The primary actor is the client, who makes the order, and the secondary actors are the staff who receive the information about the purchase.
Preconditions	The client must have the app downloaded. The app must have a listing of all current movies, their times of showing, the seating, and the prices. The client must have selected and booked a ticket for a movie.

Description of the Main Sequence	● Step 1: The client opens the app ● Step 2: They find the movie they're interested in ● Step 3: They book a ticket for it. ● Step 4 : They press "Preorder snacks/beverages" ● Step 5 : They select what they want to order ● Step 6 : They enter in their payment information, completing the purchase
Description of the Alternative Sequence	● Step 1: The client opens the app ● Step 2: They find the movie they're interested in ● Step 3: They book a ticket for it. ● Step 4 : They press "Preorder snacks/beverages" ● Step 5 : They select what they want to order. ● Step 4 : They press "Pay in theatre"
Non functional requirements	The user interface must respond quickly to the user input. The app must accept purchases in all popular modes of online payment (paypal, mastercard, visa, etc) The app must have all current snacks available, as well as the promotional deals. The payment process must be easy and intuative. Everything must be secure, insuring that the payment process is as safe for the client as possible.
Postconditions	The client is able to preorder snacks through the app, which then they can easily go and pick up at the theatre before the movie starts, making their experience faster.

Use Case Name	**UC010:Track upcoming movies**
Summary	This use case describes the ability of a client to view upcoming movies with details and receive notifications or updates.
Dependency	None
Actors	Primary actor: Clients
Preconditions	1.The client has been logged in successfully.
Description of main sequence	1.Client navigates to the “Upcoming Movies” section. 2.Client views a list of upcoming movies with their details, such as release dates, trailers, summaries and cast information. 3.Client selects a movie to view more details. 4.Client sets reminders for release dates or marks movies as favorites. 5.Client receives notifications or updates for the selected movies.
Description of the alternative sequence	1. If the client experiences a connectivity issue, the app displays a message indicating the inability to load upcoming movies. 2.The client may choose to retry the list or wait for the connection to be restored.
Non functional requirements	**-Reliability** Reminders should be reliable, ensuring clients receive notifications at the right time to stay informed about their favourite movie. **-Performance** The app should load the list of upcoming movies quickly, so clients do not have to wait too long to see what is new.
Postconditions	After using the app, clients have seen the list of movies and set reminders for their favourites.

Use Case Name	**UC011:Enroll in loyalty programs**
Summary	This use case describes the ability of a client to enroll in cinema loyalty programs through

	the app, providing necessary information such as name, email, and phone number.
Dependency	**Client case:** Claim benefits and apply promotions
Actors	Primary actor: Clients
Preconditions	1.The client has been logged in successfully.
Description of main sequence	1.Client navigates to the “Loyalty programs” section. 2.Client selects the option to enroll in a loyalty program. 3.Client provides necessary information such as name, email and phone number. 4.Client submits the enrollment form. 5.The system processes the enrollment request and adds the client to the loyalty program. 6.Client receives confirmation of enrollment.
Description of the alternative sequence	If the client encounters an error: 1.The system displays and error message asking the client to put the information again. 2.Client modifies the necessary fields and resubmits the form. 3.The system reprocesses the enrollment request.
Non functional requirements	**-Security** The information of the client should be securely stored and managed. **-Performance** The process of enrolling should be easy to understand and navigate quickly for clients.
Postconditions	After using the app, clients are successfully enrolled in the loyalty program.

Use Case Name	UC012: Claim benefits
Summary	Once clients are enrolled in loyalty programs, they can claim benefits directly from the app. This could include redeeming rewards points for free tickets or concessions, accessing exclusive discounts or promotions, or receiving personalized offers based on their preferences and past activity. Additionally, even without enrollment, clients can still enjoy various marketing promotions by viewing the list of prices in the app.
Dependency	**Client Case:** Enroll in loyalty programs **Client Case:** Contact customer support
Actors	Primary actor: Clients Secondary actor: Staff
Preconditions	1.The client has been logged in successfully.
Description of main sequence	1.Client navigates to the “Loyalty Program” or “Promotions” section. 2. If the client is enrolled in the loyalty program: -Client views the desired benefit, such as redeeming reward points for free tickets, accessing exclusive discounts, or receiving personalized offers. -Client selects the desired benefit. -Client follows the instructions to claim the benefits. -The system processes the request and applies the benefit to the client’s account. 3.If not enrolled: -Client views the list of prices or promotions available in the app. -Client selects the desired promotion. -Client follows the instructions to apply the promotion. -The system processes the request and applies the promotion to the client’s account.
Description of the alternative sequence	If the app has technical issues: -Client receives a notification informing them for the issue. -Client is asked to try again later or contact customer support.
Non functional requirements	**Security** -The personal information of the client, such as loyalty program details and transaction history should be securely stored and managed. **Availability of instructions** -The app should provide clear instructions for clients to claim benefits and apply for promotions.

Postconditions	After using the app, client has successfully claimed benefits or applied promotions.

Use Case Name	**UC013: Get Support**
Summary	Clients should have access to a customer support feature within the app, allowing them to easily reach out for assistance with any issues. They can chat with someone from staff in real-time, or find answers to common questions. They can also see what's happening with their request and get updates through the app.
Dependency	**Client case:** Claim benefits and apply promotions
Actors	Primary actor: Clients Secondary actor: Staff
Preconditions	1.The client has been logged in successfully.
Description of main sequence	1.Client navigates to the “Help” or “Support” section. 2. If the client selects the “Support” section : -Client initiates a chat with a staff member. -Client describes the issue or the question and sends the message. -Staff member responds in real time. 3.If the client selects the “Help” section : -Client browses through the FAQ section. -Client searches for relevant topics or questions. -Client reads the provided information.
Description of the alternative sequence	1.If the client cannot find answers to their question in the FAQ or knowledge base: -The app asks the client to submit a support request. -Client fills out a form detailing their issue or question. -The app notifies the client that their request has been submitted and will be addressed by staff service.

	2.If the client does not receive updates on their support request: -The app provides a "Check Status" option within the support section. -Client selects this option to view the status of their request. If there are no updates, the app informs the client that their request is still being processed.
Non functional requirements	**-Security** The conversation between the client and the staff should be secure and can't be seen by anyone else. **-Performance** The messages get sent in real time, without any delay. **-Usability** The user interface should be easy to understand and not overwhelming.
Postconditions	After finishing the conversation with the staff member, the chat gets saved in the system database, and the user goes back to the previous page.

3. MANAGER CASES

Use Case Name	**UC014: Cinema revenue**
Summary	This use case describes the involvement of Managers in tracking the revenue of the cinema using the cinema software
Dependency	None
Actors	**Primary actor-** Manager
Preconditions	1. The manager has logged on successfully
Description of main sequence	1. The manager opens the performance tab from the main view. 2. The manager chooses the "Overall Revenue" button 3. The manager enters a period or clicks the "All Time" button

	4. If the manager has entered a period the system displays the data during that period, otherwise it shows the all-time data
Description of the alternative sequence	1. The manager opens the performance tab from the main view. 2. The manager chooses the “Overall Revenue” button 3. If the manager enters a period that is not valid the system alerts him to enter a valid period 4. The manager enters a period or clicks the “All Time” button 5. If the manager has entered a period the system displays the data during that period, otherwise it shows the all-time data
Non functional requirements	**Security:** The overall revenue should be viewed only by the manager **Usability:** The overall revenue should be displayed in an easy-to-read format
Postconditions	After choosing to go back, the Manager should be sent back to the Performance section

Use Case Name	**UC015: Tracking Staff Member Income**
Summary	This use case describes the involvement of Managers in tracking the income made by staff using the cinema software
Dependency	**Staff UC:** Ticket selling
Actors	**Primary actor-** Manager **Secondary actor-** Staff
Preconditions	1. The manager has logged on successfully 2. The Staff account already exists
Description of main sequence	1. The manager opens the performance tab from the main view. 2. The manager picks from a series of already existing staff.

	3. The manager chooses the “Income” button. 4. The manager enters a period or clicks the “Overall Income” button 5. If the manager has entered a period the system displays the data during that period, otherwise it shows the all-time data
Description of the alternative sequence	1. The manager opens the performance tab from the main view. 2. The manager picks from a series of already existing staff. 3. The manager chooses the “Income” button. 4. If the Manager enters a period that is not valid the system alerts him to try a valid period. 5. The manager enters a period or clicks the “Overall Income” button 6. If the manager has entered a period the system displays the data during that period, otherwise it shows the all-time data
Non functional requirements	**Security:** The staff income should be viewed only by managers. **Performance:** The program should display the information quickly, to make it possible to check a vast number of staff faster.
Postconditions	After choosing to go back, the Manager should be sent back to the Performance section

Use Case Name	**UC016: Tracking Staff Member Spending**
Summary	This use case describes the involvement of Managers in tracking the spending made by staff using the cinema software
Dependency	**None**
Actors	**Primary actor-** Manager **Secondary actor-** Staff
Preconditions	1. The manager has logged on successfully 2. The Staff account already exists
Description of main sequence	1. The manager opens the performance tab from the main view.

	2. The manager picks from a series of already existing staff. 3. The manager chooses the “Spending” button. 4. The manager enters a period or clicks the “Overall Spending” button 5. If the manager has entered a period the system displays the data during that period, otherwise it shows the all-time data
Description of the alternative sequence	1. The manager opens the performance tab from the main view. 2. The manager picks from a series of already existing staff. 3. The manager chooses the “Spending” button. 4. If the Manager enters a period that is not valid the system alerts him to try a valid period. 5. The manager enters a period or clicks the “Overall Spending” button 6. If the manager has entered a period the system displays the data during that period, otherwise it shows the all-time data
Non functional requirements	**Security:** The staff spending should be viewed only by managers. **Performance:** The program should display the information quickly, to make it possible to check a vast number of staff faster.
Postconditions	After choosing to go back, the Manager should be sent back to the Performance section

Use Case Name	**UC017: Add a new film**
Summary	This use case describes the involvement of Managers in adding a new film to the catalogue of the cinema using the cinema

	software
Dependency	**None**
Actors	**Primary actor-** Manager
Preconditions	1. The manager has logged on successfully
Description of main sequence	1. The manager opens the manage catalogue tab from the main view 2. The manager chooses the “Add new Film” button 3. The manager enters the information regarding the film 4. The manager clicks the “Save” button 5. The system validates the data entered and adds the film if they are correct
Description of the alternative sequence	1. The manager opens the manage catalogue tab from the main view 2. The manager chooses the “Add new Film” button 3. The system displayed the type of information needed 4. The manager enters the information regarding the film 5. The manager clicks the “Save” button 6. The system validates the data entered shows which ones are entered incorrectly and asks the user to re-enter if they are incorrect
Non functional requirements	**Security:** Only the manager can add new films to the catalogue **Performance:** The program should process the values entered quickly, to make the work of adding films more efficient
Postconditions	After choosing to go back, the Manager should be sent back to the Manage Catalogue section

Use Case Name	**UC018: Delete an existing film**
Summary	This use case describes the involvement of Managers in deleting an existing film from the catalogue of the cinema using the cinema software
Dependency	**None**
Actors	**Primary actor-** Manager
Preconditions	1. The manager has logged on successfully 2. The film to be deleted already exists
Description of main sequence	1. The manager opens the manage catalogue tab from the main view 2. The manager chooses the “Delete Film” button 3. The manager picks from a series of already existing films 4. The system displays all of the information regarding the film 5. The manager clicks the “Delete” button 6. The system displays a message that informs the manager that the film is deleted

Description of the alternative sequence	None
Non functional requirements	**Security:** Only the manager can delete existing films from the catalogue **Performance:** The program should handle deleting films rapidly to make the work of deleting films more efficient
Postconditions	After choosing to go back, the Manager should be sent back to the Manage Catalogue section

Use Case Name	**UC019: Modifying an existing film**
Summary	This use case describes the involvement of Managers in modifying the data of an existing film from the catalogue of the cinema using the cinema software
Dependency	**None**
Actors	**Primary actor-** Manager
Preconditions	1. The manager has logged on successfully 2. The film to be modified already exists
Description of main sequence	1. The manager opens the manage catalogue tab from the main view 2. The manager chooses the “Modify Film” button 3. The manager picks from a series of already existing films 4. The system displays all of the information regarding the film 5. The manager modifies the data that he/she wants 6. The manager clicks the “Save” button 7. The system validates the data entered and modifies the film if they are correct
Description of the alternative sequence	1. The manager opens the manage catalogue tab from the main view

	2. The manager chooses the “Modify Film” button 3. The manager picks from a series of already existing films 4. The system displays all of the information regarding the film 5. The manager modifies the data that he/she wants 6. The manager clicks the “Save” button 7. The system validates the data entered shows which ones are entered incorrectly and asks the user to re-enter if they are incorrect
Non functional requirements	**Security:** Only the manager can modify the data of existing films from the catalogue **Performance:** The program should process the values entered quickly, to make the work of modifying films more efficient
Postconditions	After choosing to go back, the Manager should be sent back to the Manage Catalogue section

UC Name	UC020: User Login
Summary	This use case allows users to authenticate themselves and gain access to the system. It verifies the identity of users attempting to log in and ensures that only authorized individuals can access the system.
Dependency	None
Actors	**Primary Actor:**Manager **Secondary actor:** Staff
Preconditions	The user must have a valid username and password.
Description of the Main Sequence	1-The user enters their username and password. 2-The system validates the entered credentials against stored records. 3-If the credentials are valid, the system grants access to the user, allowing them to proceed to use the system.
Description of the	If the entered credentials are invalid, the system prompts the user to re-enter

Alternative Sequence	their username and password, or it may provide an error message indicating the incorrect credentials.
Non functional requirements	1-Secure encryption of passwords to ensure confidentiality. 2-Authentication mechanism should be efficient to provide a seamless user experience. 3-The system should be resilient against common security threats like brute force attacks or credential stuffing.
Postconditions	The user gains access to the system upon successful authentication, allowing them to perform their intended actions based on their role and permissions within the system.

UC Name	**UC021: Access System via Mobile App**
Summary	This use case allows users to access the system using a dedicated mobile application, providing flexibility and convenience for users who need to manage staff schedules or track attendance while on the go.
Dependency	User Login
Actors	Primary Actor:Manager Secondary actors: Staff
Preconditions	The mobile application is installed on the user's device, and the user must be logged into the system.
Description of the Main Sequence	1-The user launches the mobile application on their device. 2-The user enters their username and password into the mobile app. 3-The system authenticates the user's credentials and grants access to the mobile interface. 4-The user can then view staff schedules, record attendance, or perform other relevant tasks using the mobile app.
Description of the Alternative Sequence	If the mobile network connection is poor or unavailable, the user may experience delays in accessing the system or encounter issues with data synchronization.

Non functional requirements	1-The mobile interface should be responsive and user-friendly, optimized for various screen sizes and devices. 2-Offline access capabilities should be provided to allow users to perform essential tasks even when not connected to the internet. 3-The mobile app should adhere to security best practices to ensure the confidentiality and integrity of data transmitted between the device and the system.
Postconditions	The user gains access to the system upon successful authentication, allowing them to perform their intended actions based on their role and permissions within the system.

UC Name	**UC022: Create/Edit Schedule**
Summary	This use case empowers managers to create or modify schedules for staff members. It facilitates the efficient allocation of shifts to ensure adequate coverage while taking into account factors such as employee availability and business needs.
Dependency	User Login
Actors	Manager
Preconditions	The manager must be logged into the system.
Description of the Main Sequence	1-The manager selects the staff member or group for whom the schedule needs to be created or modified. 2-The manager specifies the date and time slots for shifts, assigning staff members accordingly. 3-The system updates the schedule to reflect the changes made by the manager.
Description of the Alternative Sequence	If conflicts arise, such as overlapping shifts or unavailable staff members, the manager may be prompted to resolve these conflicts before finalizing the schedule.

Non functional requirements	1-The scheduling interface should be intuitive and user-friendly to streamline the process for managers. 2-The system should support recurring schedules as well as one-time events for flexibility in scheduling. 3-It should provide real-time updates and notifications to keep staff members informed of any changes to their schedules.
Postconditions	The schedule for the selected staff member or group is successfully created or updated in the system, ensuring that shifts are allocated effectively to meet operational requirements.

4. STAFF CASES

UC Name	**UC023: Selling Tickets**
Summary	This use case outlines the process for staff to sell tickets to customers.
Dependency	None
Actors	Primary actor: Staff Secondary actor: Customer
Preconditions	Staff member is logged into the system. Ticket selling functionality is accessible.
Description of the Main Sequence	1.Staff member navigates to the ticket selling section of the system. 2. Staff member selects the movie, showtime, and seat(s) based on customer preference.

	3.Staff member confirms the ticket details and calculates the total price. 4.Staff member collects payment from the customer. 5.Staff member issues the ticket to the customer. 6. Staff member updates the system to reflect the sold ticket.
Description of the Alternative Sequence	If the selected seats are not available, the staff member informs the customer and helps them choose alternative seats.
Non functional requirements	1. Performance: The system should handle ticket sales to minimize customer waiting time. 2. Usability: The ticket selling interface should be intuitive for staff members to navigate. 3. Security: Staff members should only have access to sell tickets and not perform unauthorized actions.
Postconditions	The customer receives the ticket, and the system updates the available seat inventory accordingly.

UC Name	**UC024:Handling Customer Inquiries**
Summary	This use case describes how staff handle customer inquiries and assistance requests.
Dependency	None
Actors	Primary actor: Staff Secondary actor: Customer

Preconditions	Staff member is logged into the system. Customer inquiries functionality is accessible.
Description of the Main Sequence	1.Staff member receives a customer inquiry through in-person interaction, phone call, or online chat. 2. Staff member listens to the customer query and gathers necessary information. 3.Staff member provides relevant information or assistance to resolve the customer's issue. 4.If necessary, staff member escalates the inquiry to a manager or higher authority.
Description of the Alternative Sequence	If the staff member is unable to resolve the inquiry immediately, they inform the customer of the steps being taken to address the issue and provide an estimated resolution time.
Non functional requirements	1. Responsiveness: Staff members should respond promptly to customer inquiries. 2. Communication: Staff members should effectively communicate with customers to understand and address their concerns. 3. Professionalism: Staff members should maintain a professional demeanor while assisting customers
Postconditions	The customer's inquiry is resolved satisfactorily, or appropriate steps are taken to address the issue.

UC Name	**UC025:Checking Ticket Validity**
Summary	This use case outlines the process for staff to check the validity of tickets

	presented by customers..
Dependency	None
Actors	Primary actor: Staff Secondary actor: Customer
Preconditions	Staff member is logged into the system. The customer provides their ticket.
Description of the Main Sequence	1.Staff member navigates to manage ticket section. 2. Staff member scans the ticket or manually searches the ticket by ticketId. 3. Staff member checks the validity of the ticket. (If it exists in the database or not). 4.If the ticket is valid, the staff member changes the status of the ticket. 5.If the ticket is invalid, staff member may create a new ticket for the client.
Description of the Alternative Sequence	**Invalid ticket format** 1.Staff member navigates to manage ticket section. 2. Staff member scans the ticket or manually searches the ticket by ticketId but the format is incorrect. 3. Staff member will stay on the same page and the system will provide an error message that the format of the ticked Id is not correct

Non functional requirements	1. Reliability: The ticket scanning system should accurately detect valid and invalid tickets. 2. Efficiency: The ticket validation process should be swift to minimize customer waiting time. 3. Customer Service: Staff members should handle invalid ticket situations courteously and efficiently.
Postconditions	Customers with valid tickets are allowed entry, while appropriate action is taken for customers with invalid tickets.

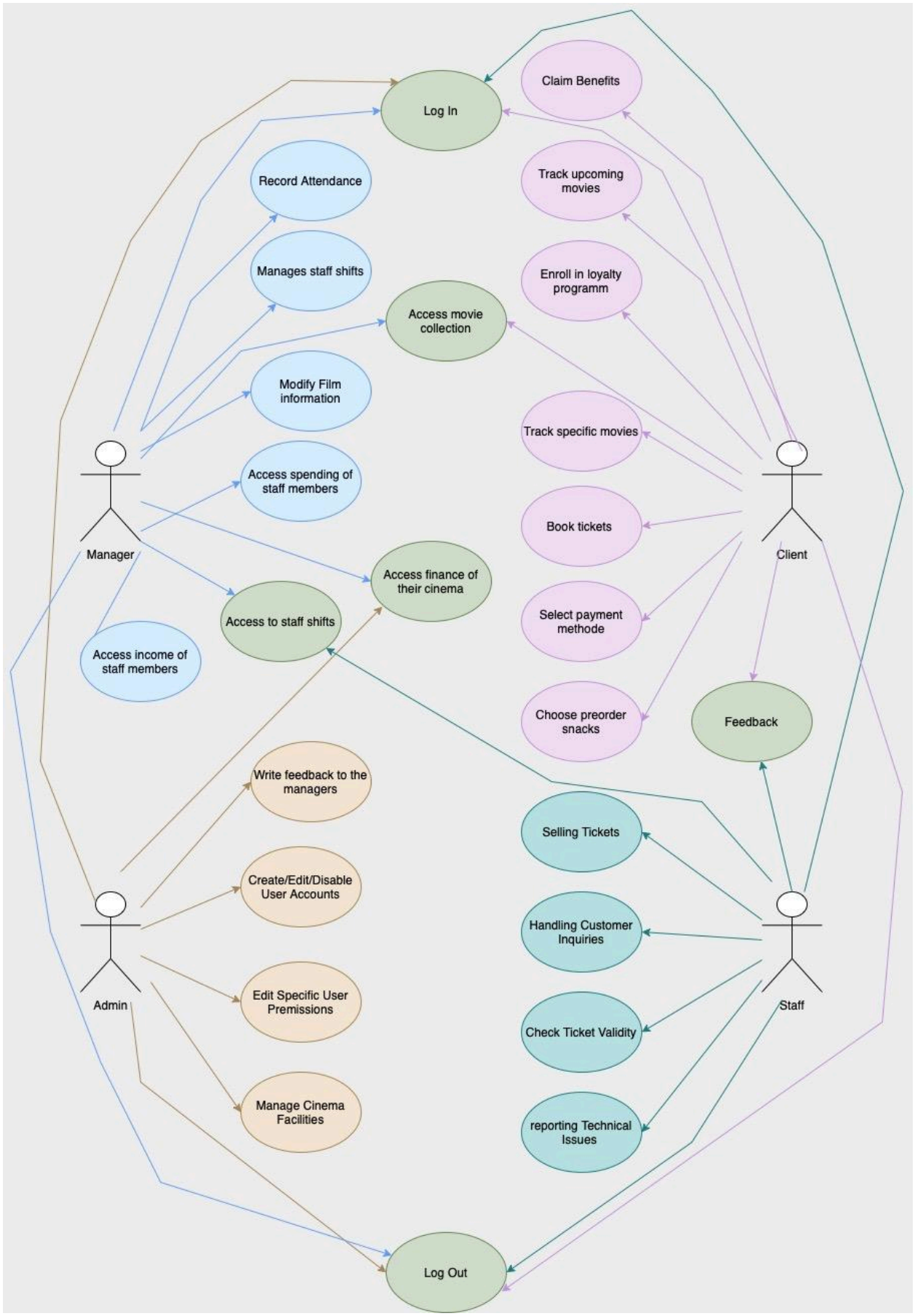
UC Name	**UC026:Reporting Technical Issues**
Summary	This use case describes how staff report technical issues encountered with cinema equipment or software.
Dependency	None
Actors	Primary actor: Staff Member Secondary actor: Manager
Preconditions	Staff member is logged into the system. Technical issue reporting functionality is accessible..
Description of the Main Sequence	1.Staff member navigates to the Technical Issue section within the system. 2.Staff member creates an issue. 3.Staff member enters a detailed description of the problem. 4.System will notify the staff member, type of technician via email.

	5.If a staff member, type of technician, resolves the technical issue, the system will notify the staff member via email. 6. If a staff member, type of technician, resolves the technical issue, it must report to the manager.
Description of the Alternative Sequence	1.Staff member navigates to the Technical Issue section within the system. 2.Staff member creates an issue. 3.Staff member enters a detailed description of the problem. 4.System will notify the staff member, type of technician via email. 5.Technician has not solved the issue for 7 days, then the status of the issue changes to important.
Non functional requirements	1. Clarity: Staff members should provide clear and detailed descriptions of technical issues to facilitate resolution. 2.Timeliness: Staff members should report technical issues promptly to minimize disruption to cinema operations. 3. Collaboration: Staff members should collaborate with technical support teams to resolve reported issues effectively.
Postconditions	Technical support teams receive the reported issue and initiate appropriate troubleshooting and resolution procedures.

UC Name	**UC027: Making a Transaction**
Summary	This use case outlines the process for staff to make a new transaction composed of tickets and snacks.
Dependency	None

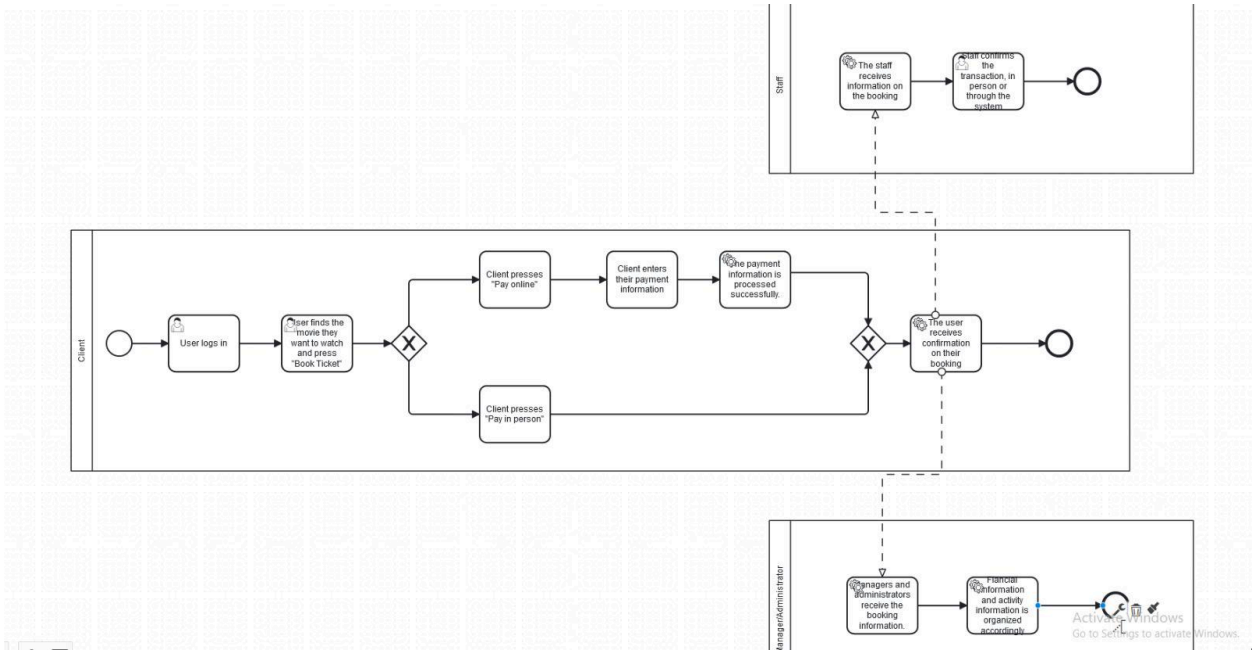
Actors	Primary actor: Staff
Preconditions	Staff member is logged into the system.
Description of the Main Sequence	1. The staff member navigates to the Transaction section in the system. 2. The staff member creates a new Transaction. 3.The staff member selects the type and number of tickets the customer wants to purchase. 4. The system returns available seating and pricing options. 5. New ticket(s) are created. 6. Optionally, the staff member adds the customer's selected snacks to the transaction. 7. The system calculates the total cost of the transaction.. 8. The system displays the bill for the transaction.
Description of the Alternative Sequence	1. The staff member navigates to the Transaction section in the system. 2. The staff member creates a new Transaction. 3.The staff member selects the type and number of tickets the customer wants to purchase. 4. The system returns available seating and pricing options. 5. There are no seats available. 6. System won’t allow to make a new ticket for this movie in this hour.

Non functional requirements	1. Performance: The system should handle ticket sales to minimize customer waiting time. 2. Usability: The ticket selling interface should be intuitive for staff members to navigate. 3. Security: Staff members should only have access to sell tickets and not perform unauthorized actions.
Postconditions	The customer receives the ticket(s), snack(s) and the system updates the available seat inventory accordingly and generates a new bill in the end.


Log In
Claim Benefits
Record Attendance
Track upcoming movies
Manages staff shifts
Enroll in loyalty programm
Access movie collection
Modify Film information
Track specific movies
Access spending of staff members
Book tickets
Manager
Client
Access finance of their cinema
Select payment methode
Access to staff shifts
Access income of staff members
Choose preorder snacks
Feedback
Write feedback to the managers
Selling Tickets
Create/Edit/Disable User Accounts
Handling Customer Inquiries
Admin
Staff
Edit Specific User Premissions
Check Ticket Validity
Manage Cinema Facilities
reporting Technical Issues
Log Out

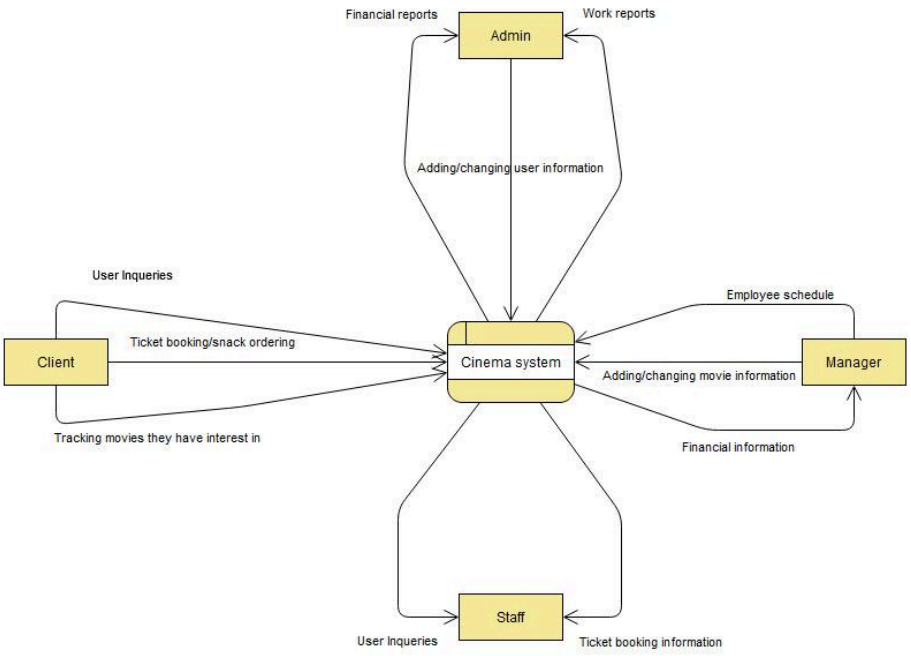
4.2 BPMN

Manager:

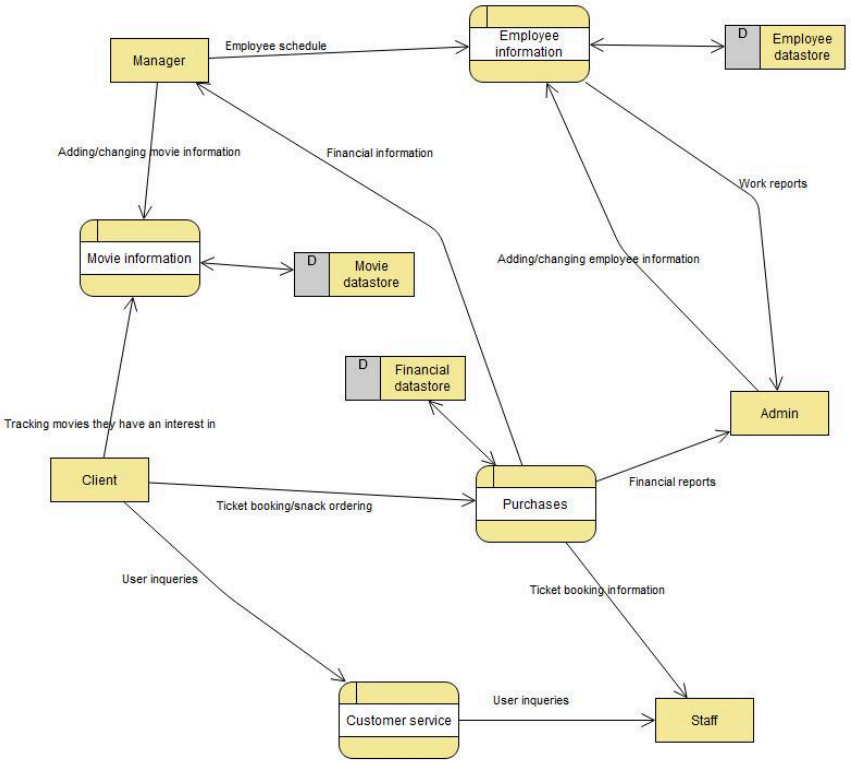


4.3 Data Flow Diagrams

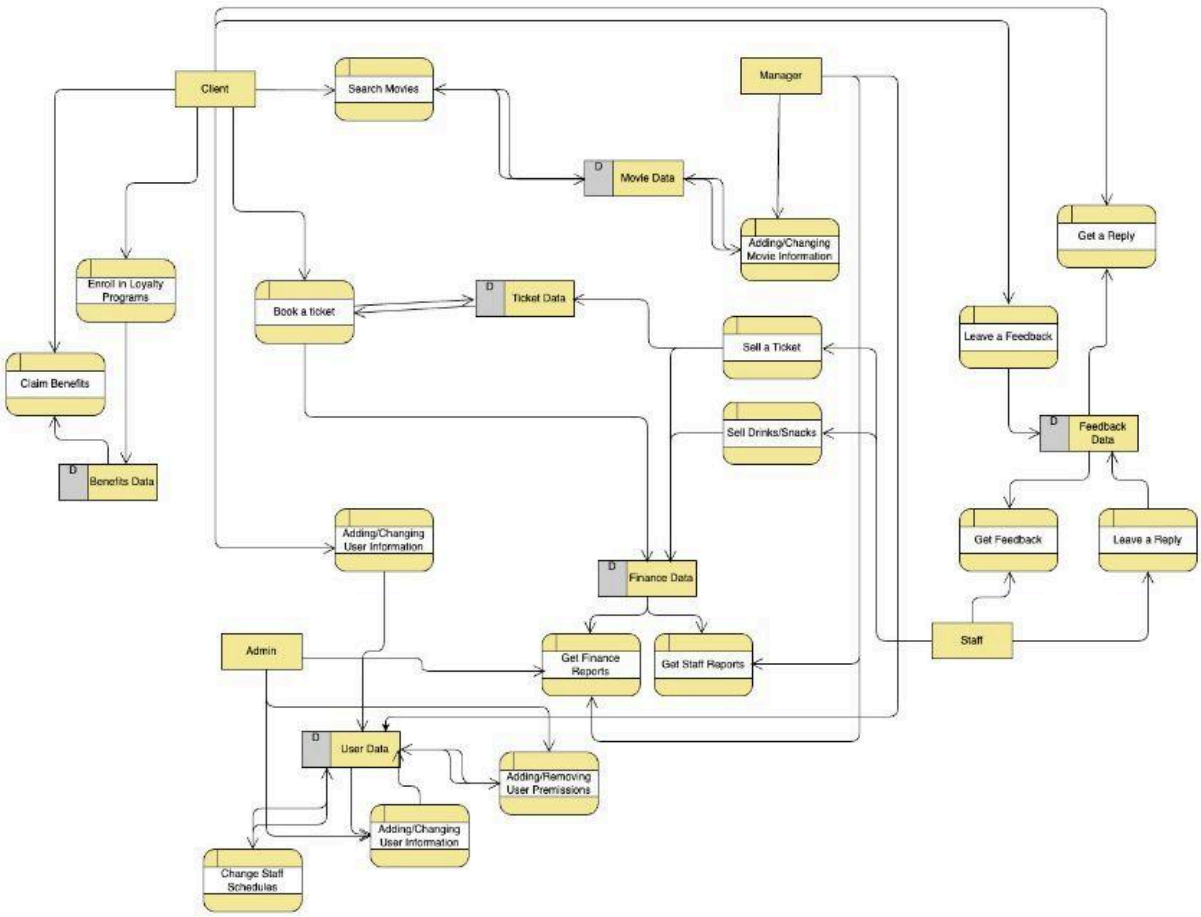
1 - Level 0



2 - Level 1

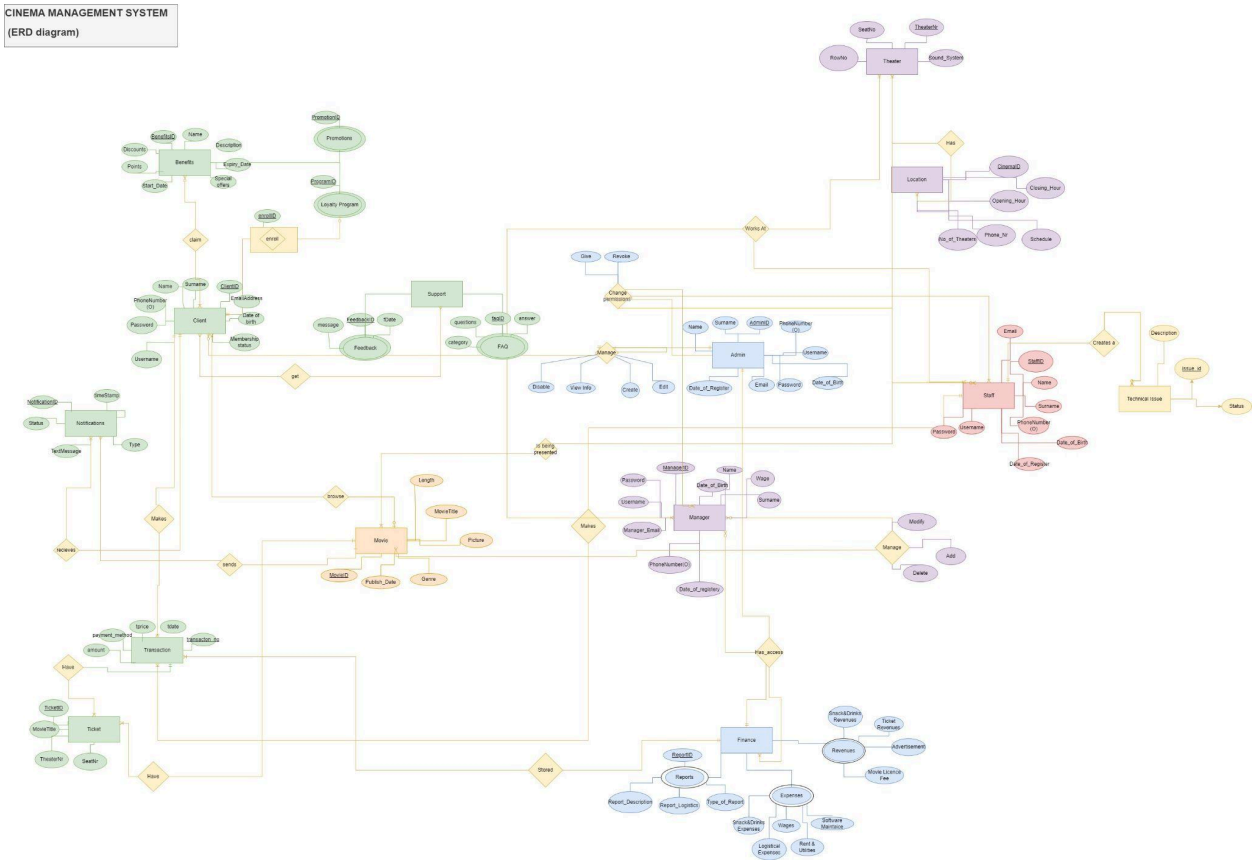

Manager
Employee schedule
Employee information
D
Employee datastore
Adding/changing movie information
Financial information
Work reports
Movie information
D
Movie datastore
Adding/changing employee information
D
Financial datastore
Admin
Tracking movies they have an interest in
Client
Ticket booking/snack ordering
Purchases
Financial reports
User inqueries
Ticket booking information
Customer service
User inqueries
Staff

3 - Level 2



4.4 Entity-Relationship Diagrams

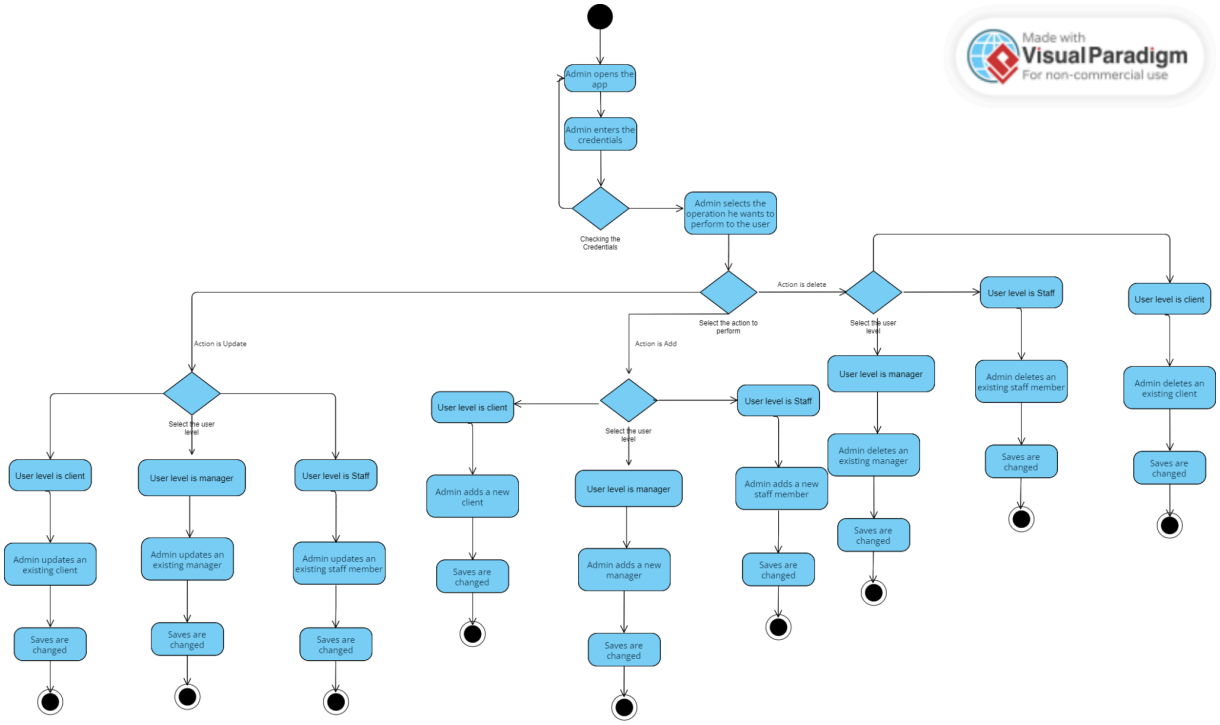
https://app.diagrams.net/#G1c-AjWGb-0AXxIeBBHY279NkJm2GfrkkE#%7B%22pageId%22%3A%225O1PYR-_gxztDaC10rXU%22%7D



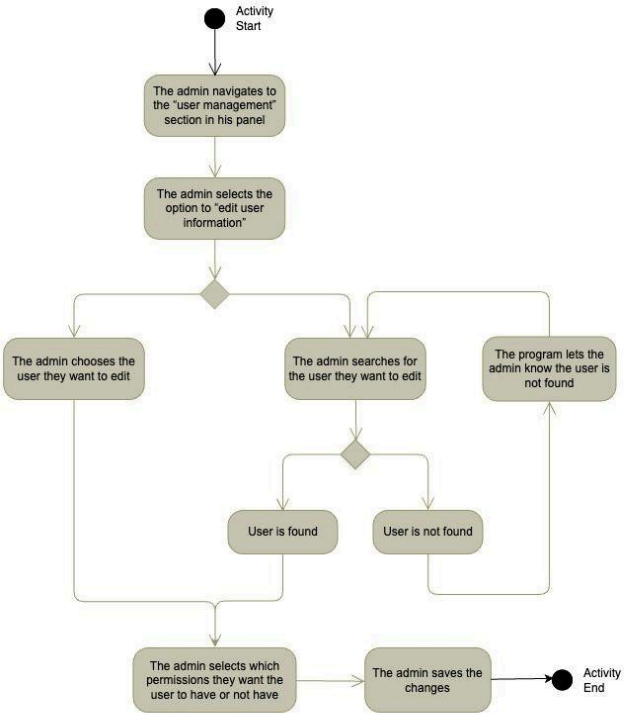
4.5 Activity Diagrams

1.ADMIN

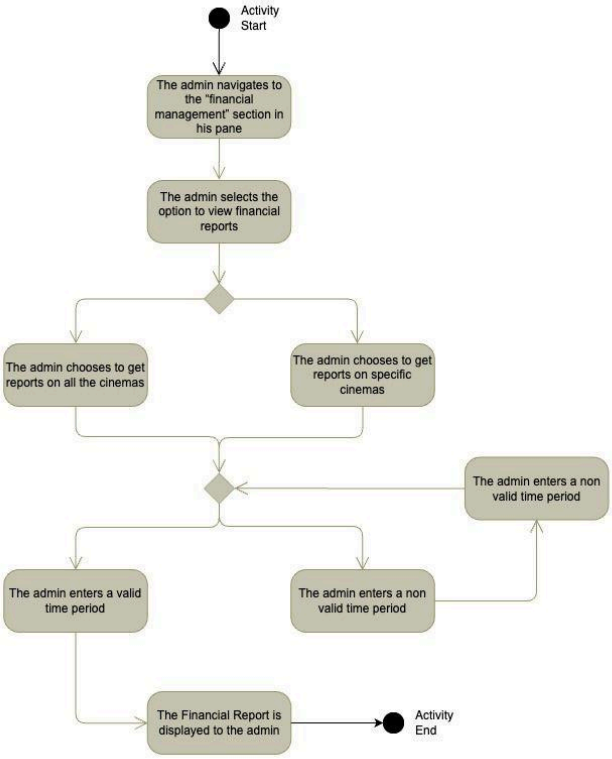
1-CRUD Operations



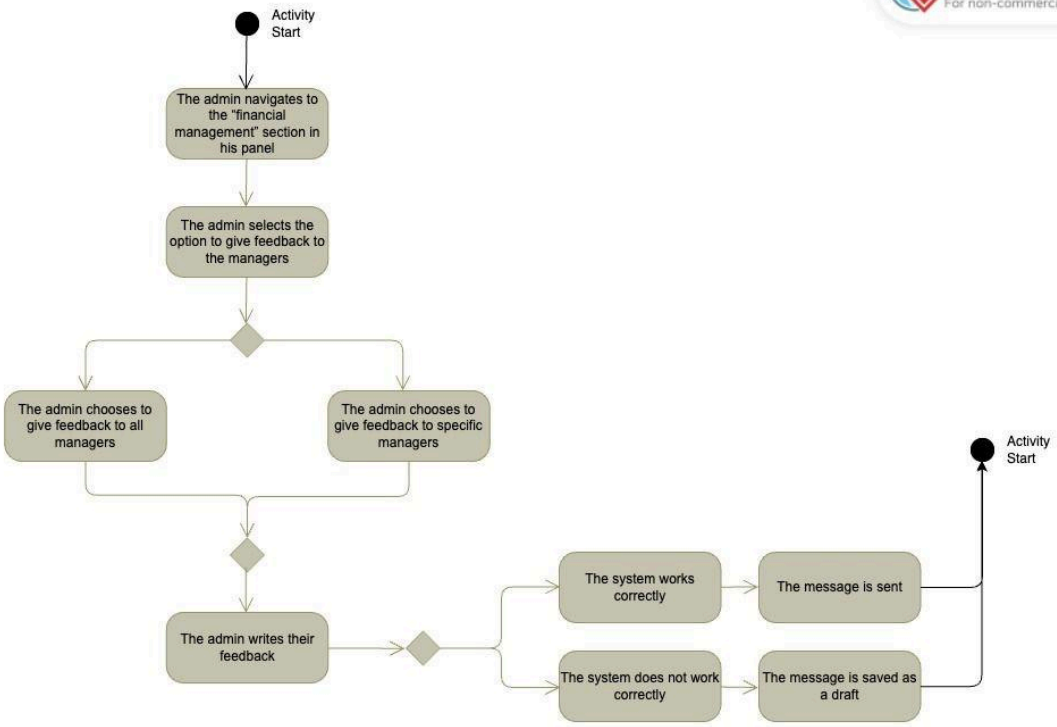
2 - Changing specific User Permissions



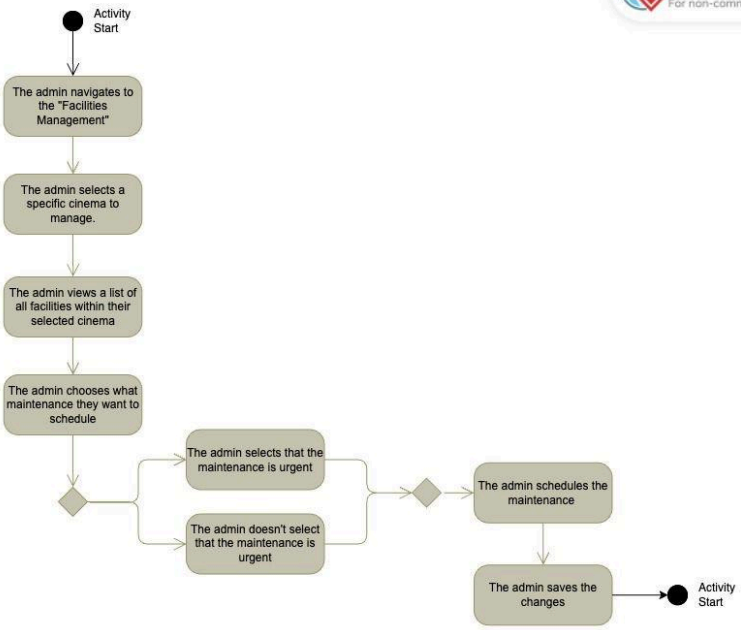
3 - View Financial Reports



4 - Financial Feedback

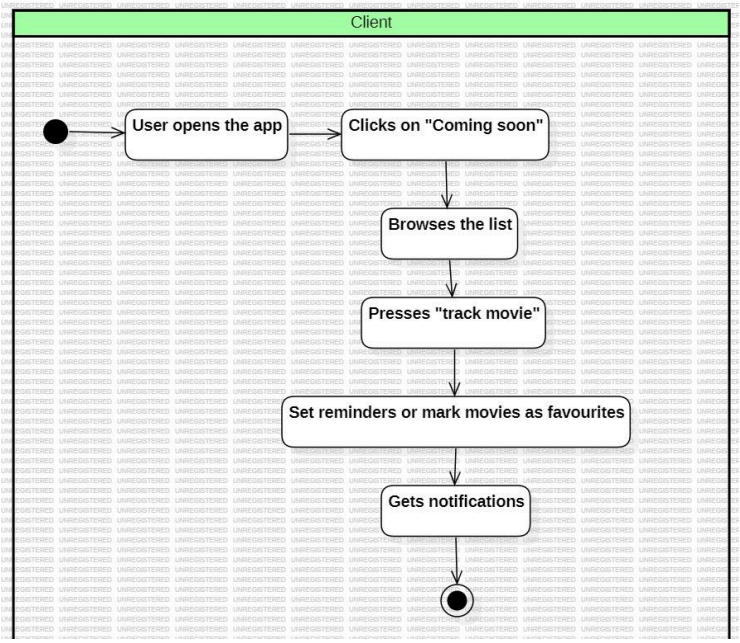


5 - Managing Cinema Facilities

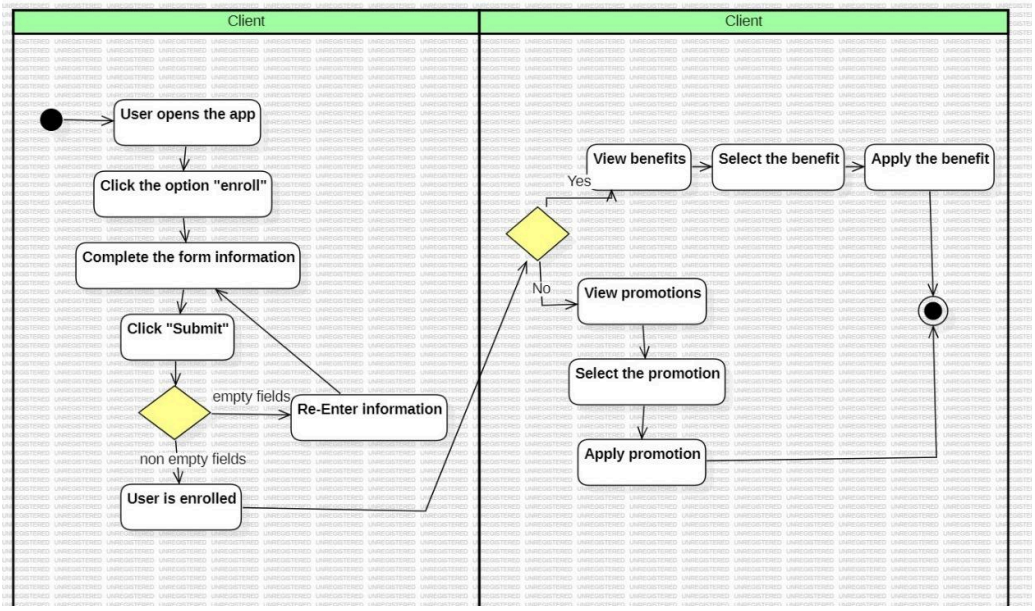


2.CLIENT

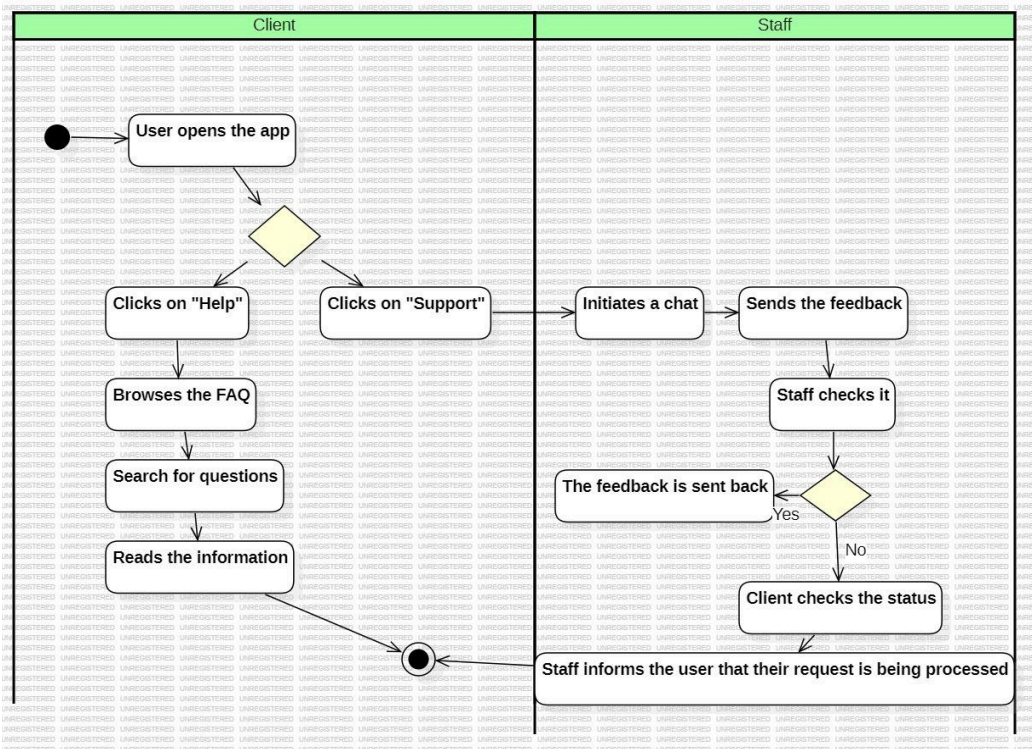
1 - Track upcoming movies



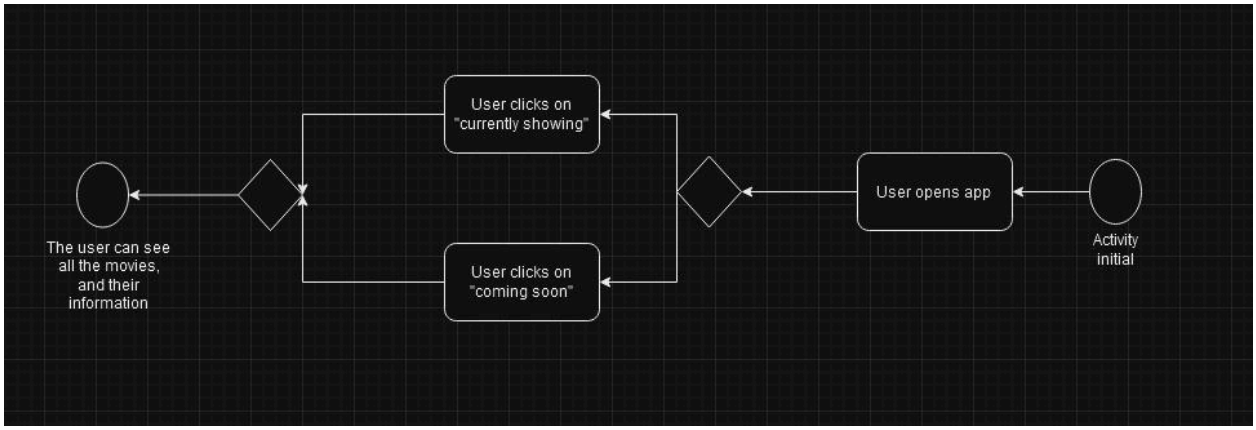
2 - Enroll in loyalty program and 3- Claim Benefits



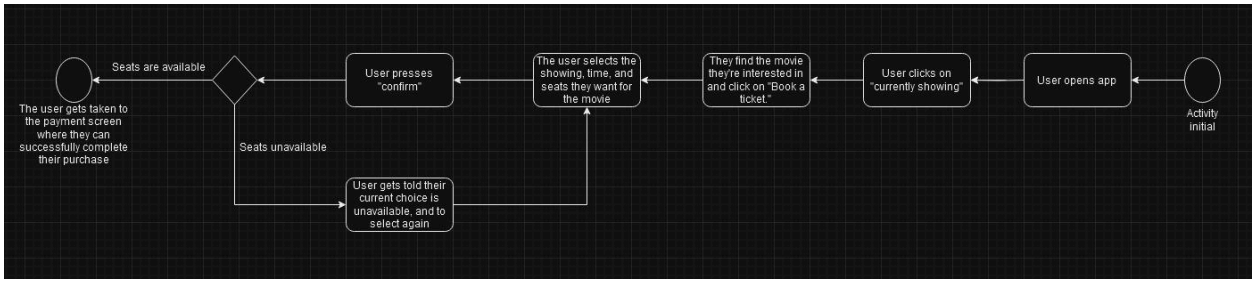
4.Get Support



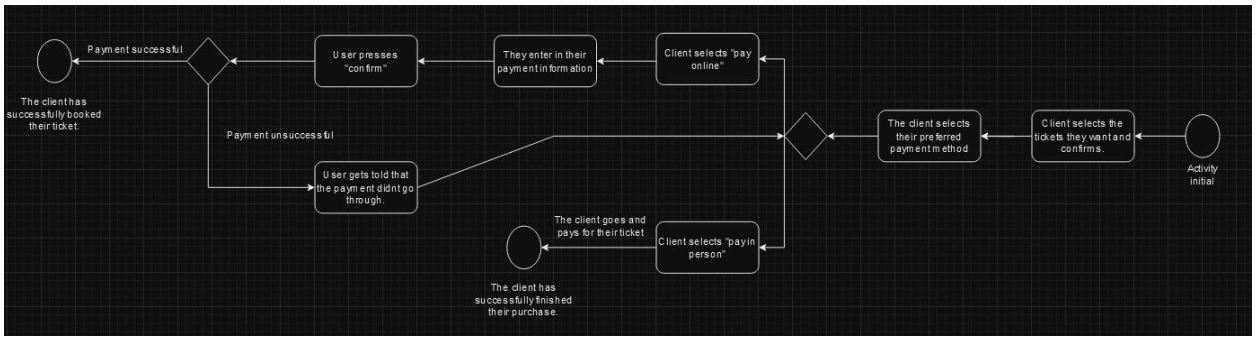
5- See current and upcoming movies



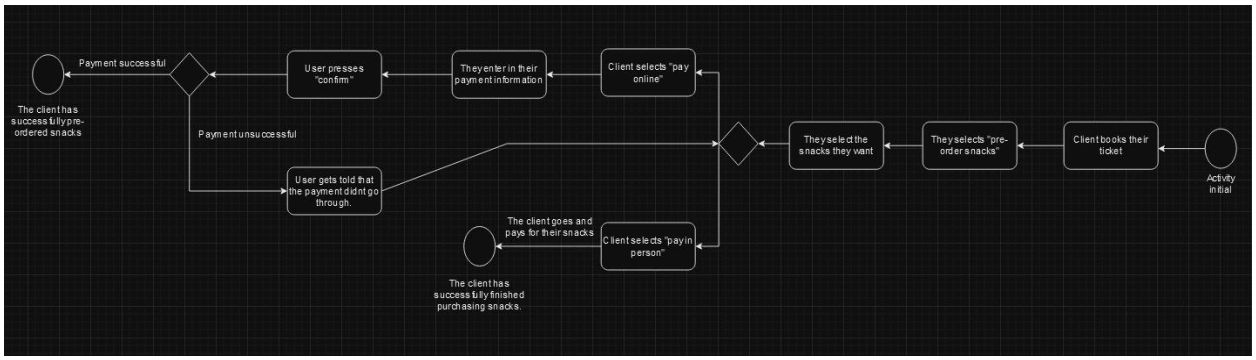
6- book tickets



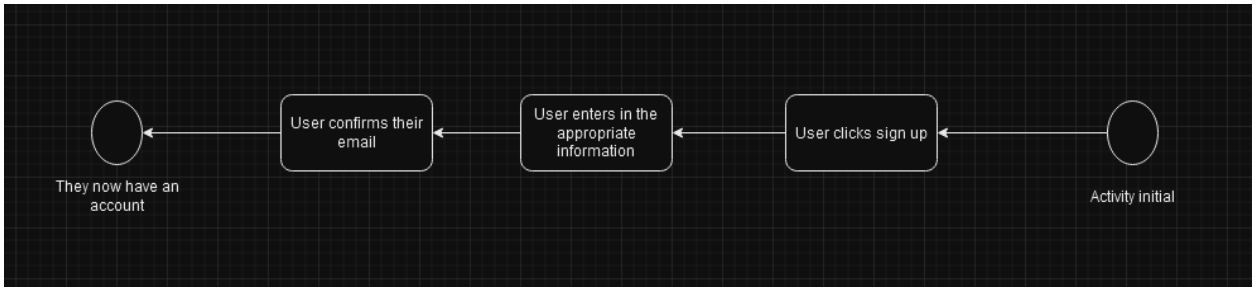
7- pay in app



8- pre-order snacks

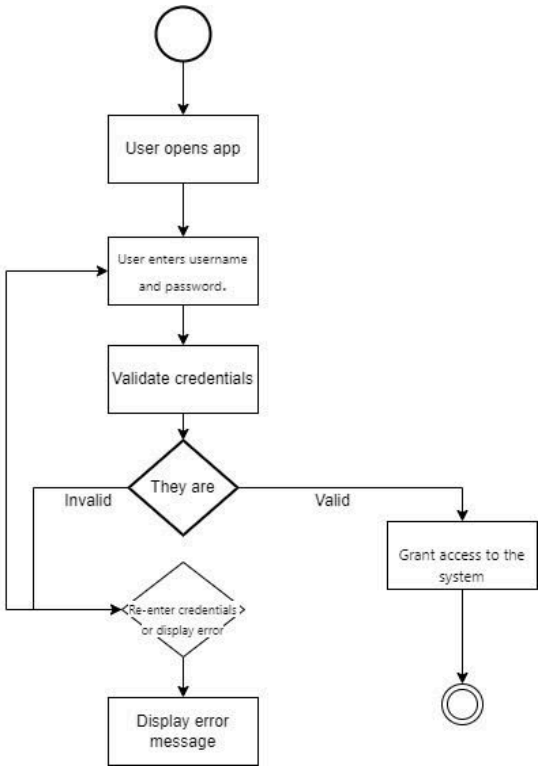


9- log in as a user

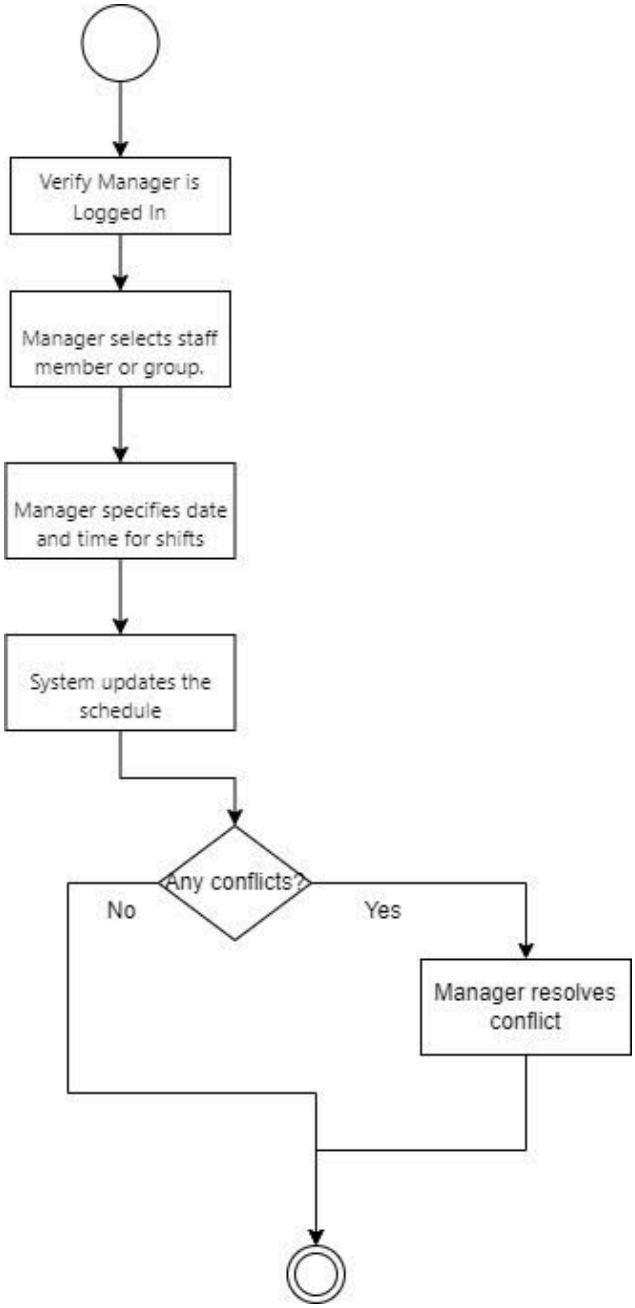


3.MANAGER

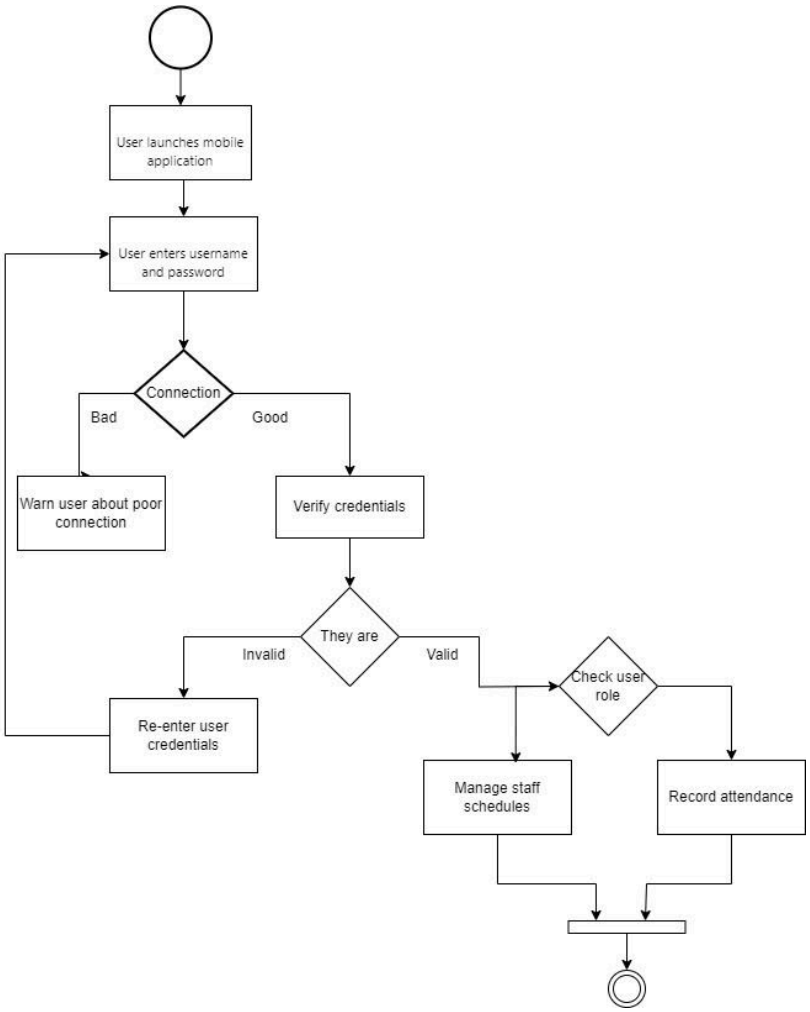
1.User Login:



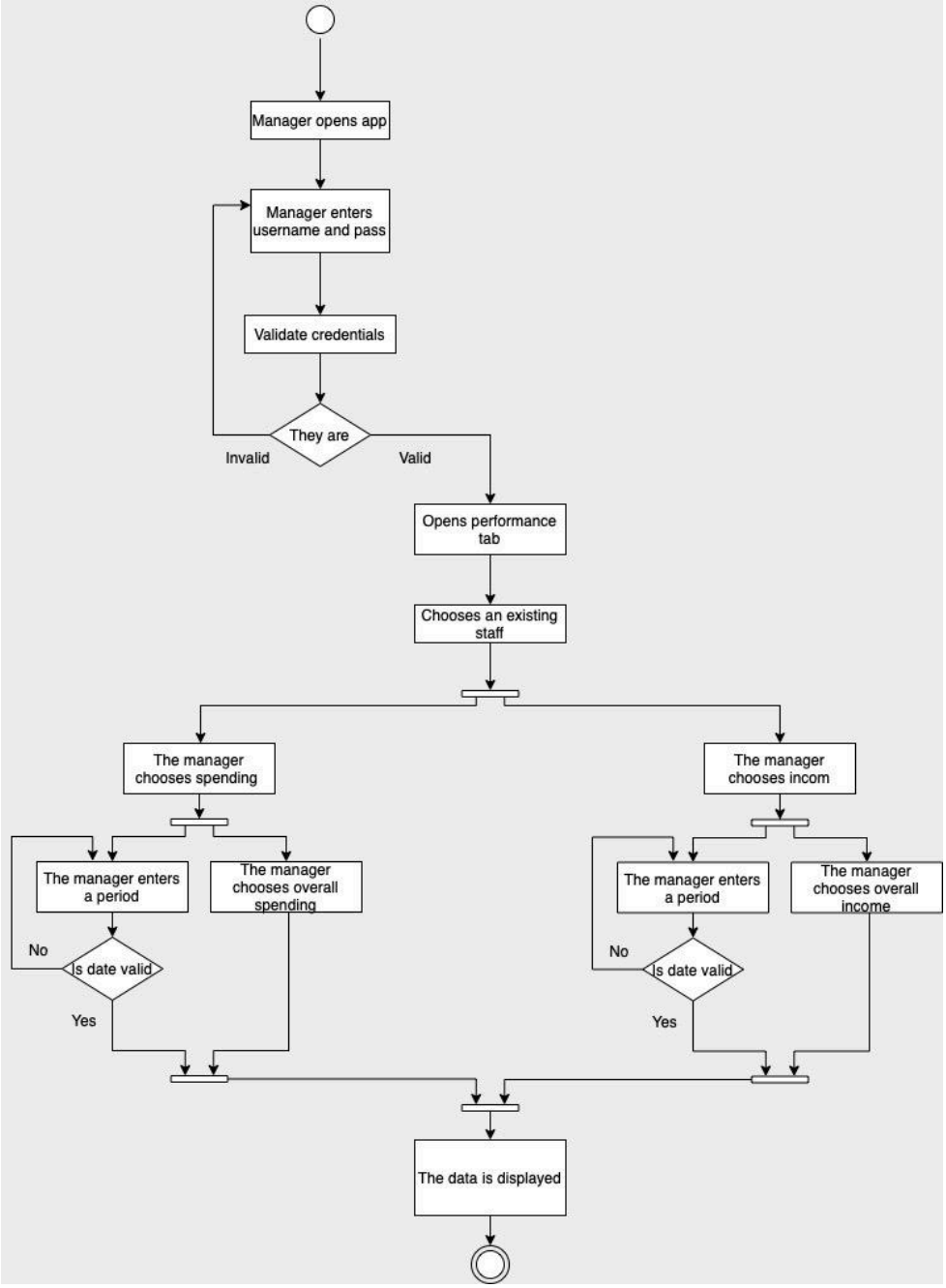
2.Create/Edit Schedule


Verify Manager is Logged In
Manager selects staff member or group.
Manager specifies date and time for shifts
System updates the schedule
Any conflicts?
No
Yes
Manager resolves conflict

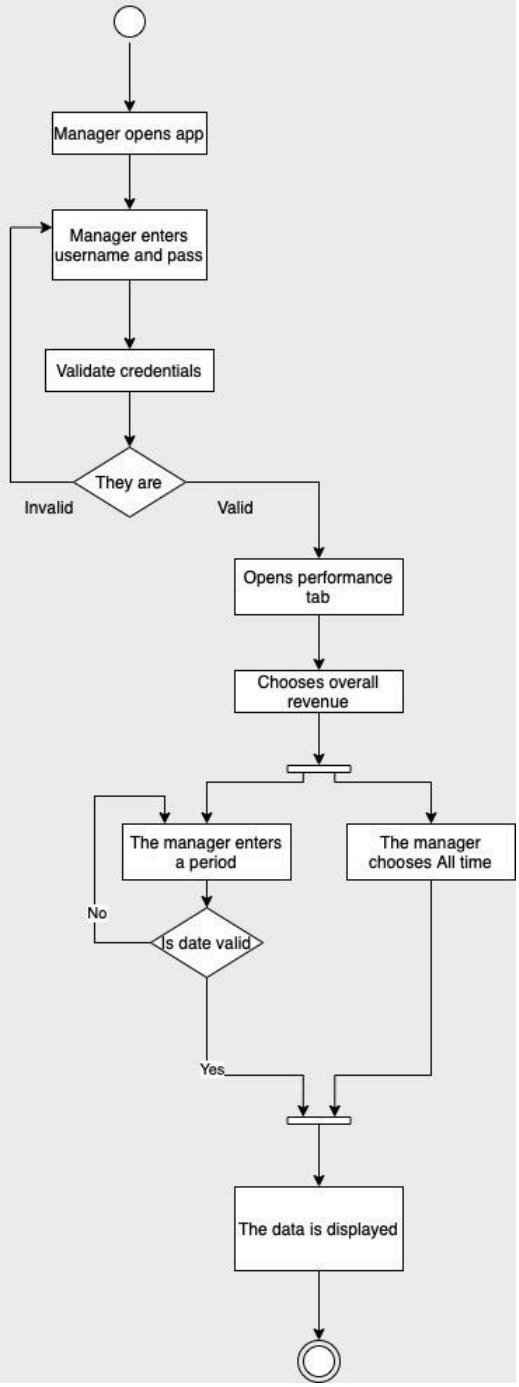
3.Access System via mobile app:



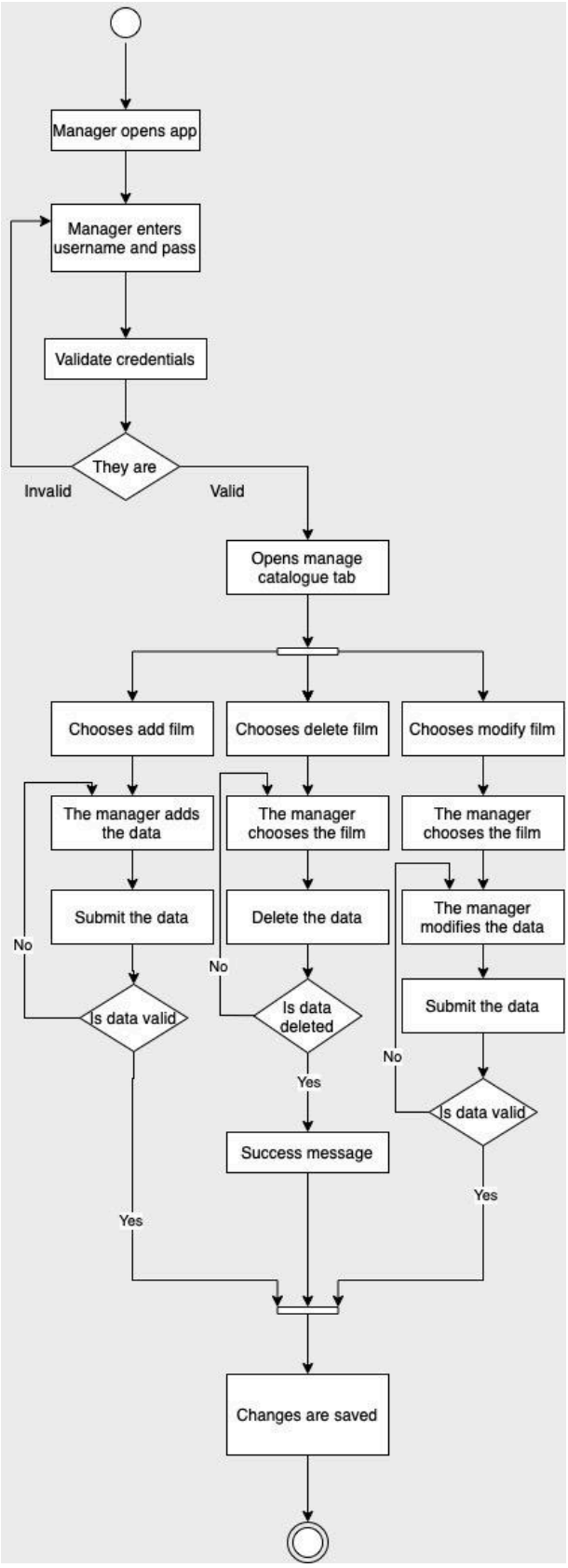
Staff spending/income:



Overall revenue:

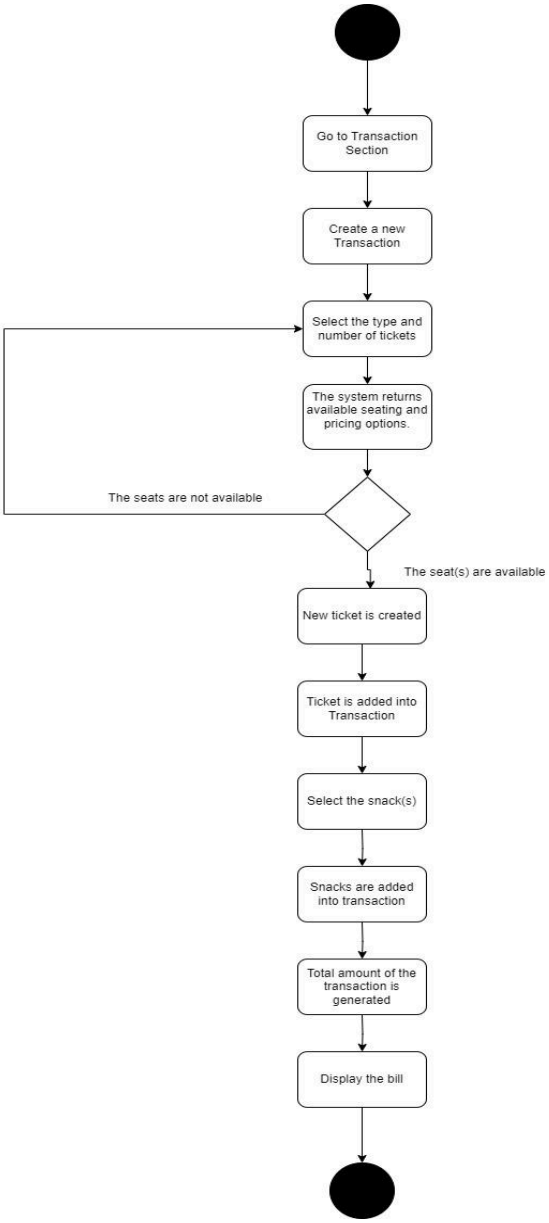


Modify Movies:

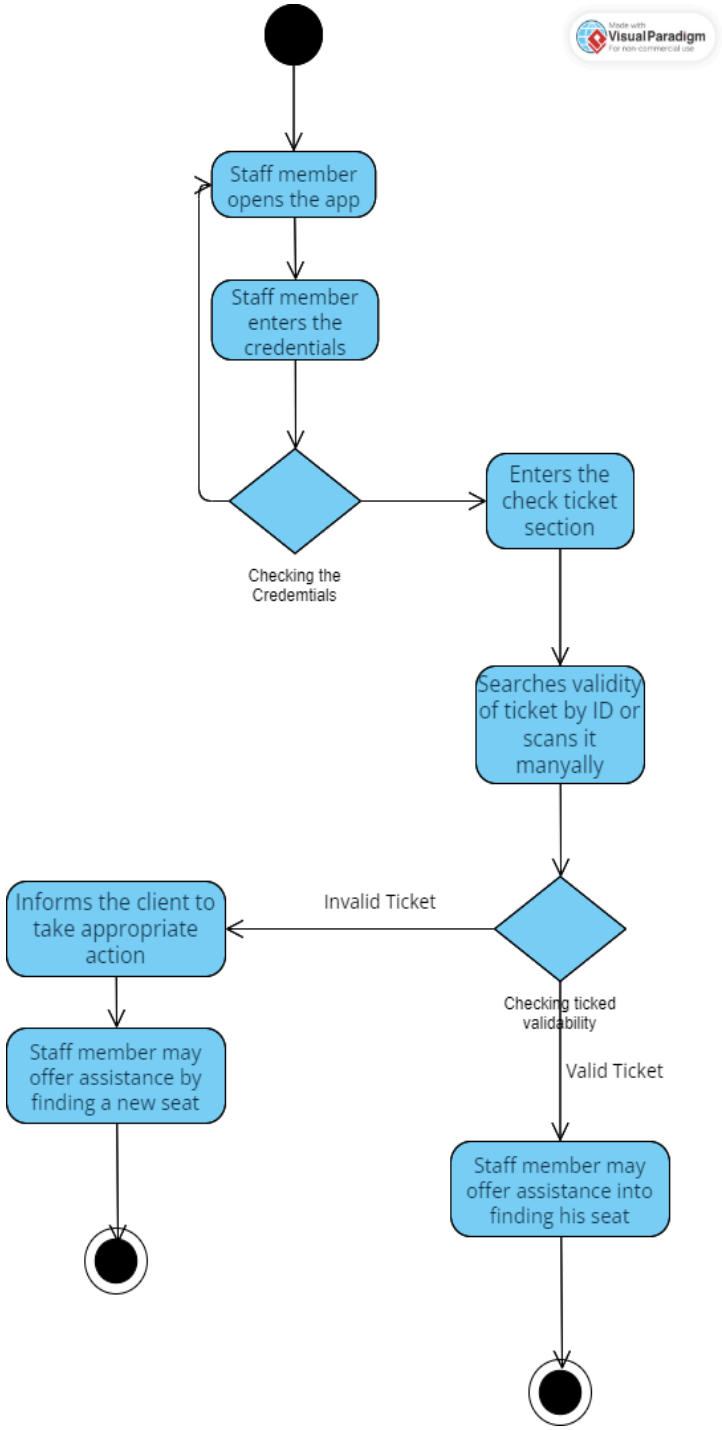


4.STAFF

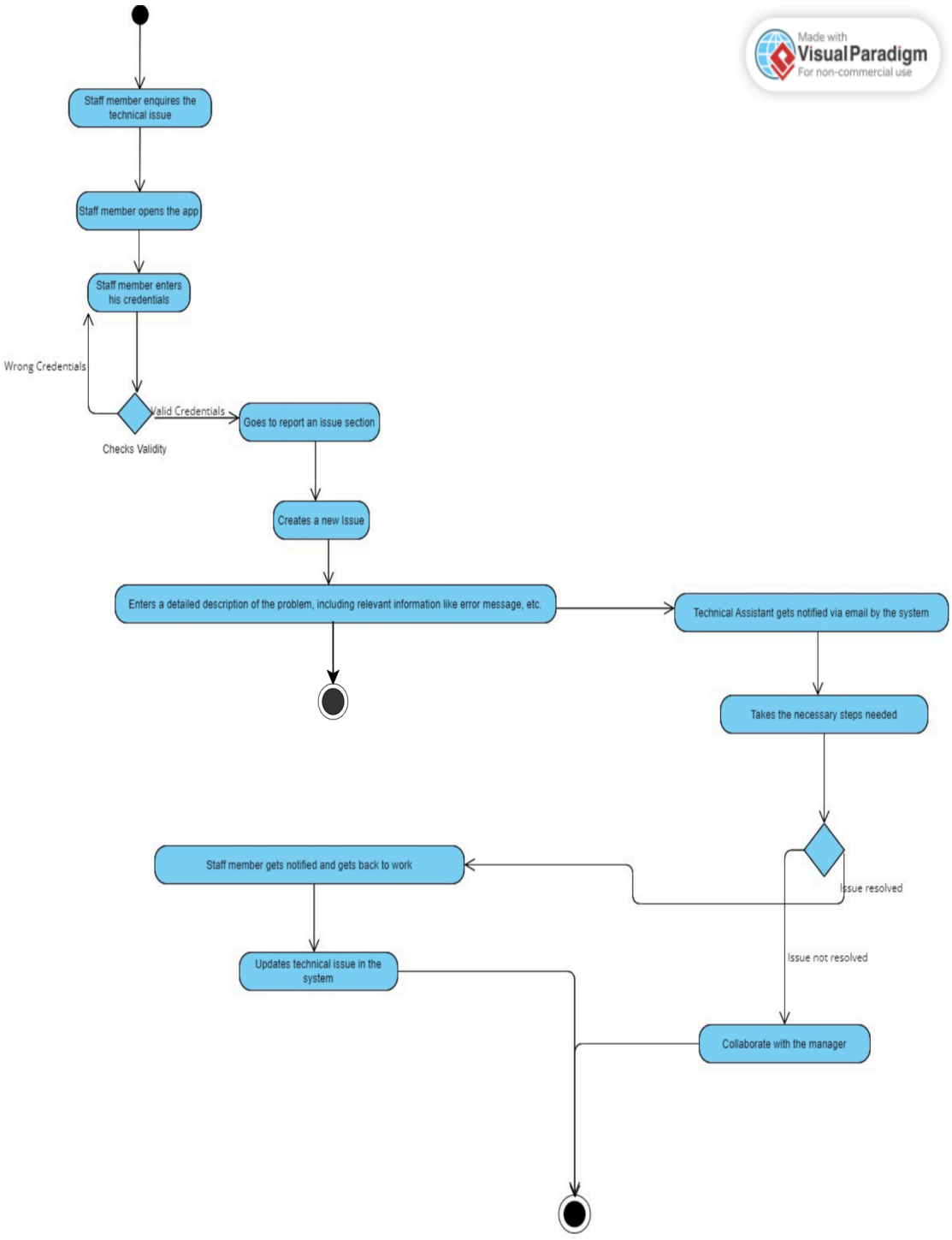
1-Create Transaction

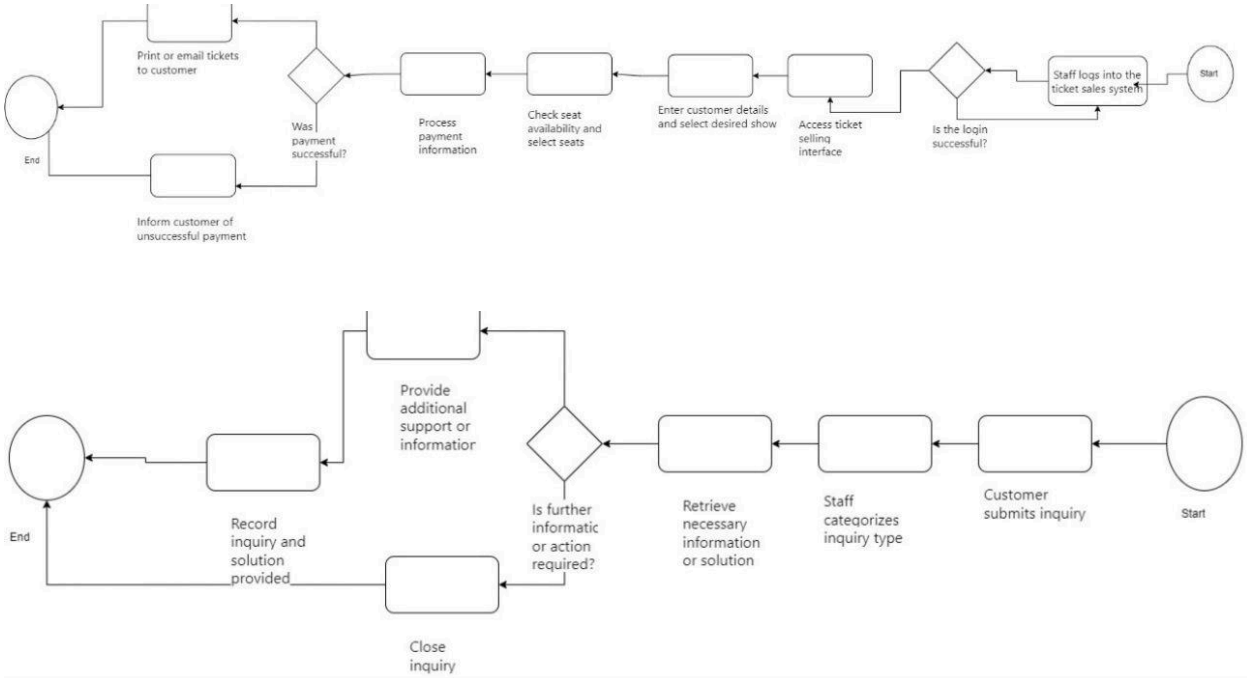


2- Valid Ticket



3-Technical Issue

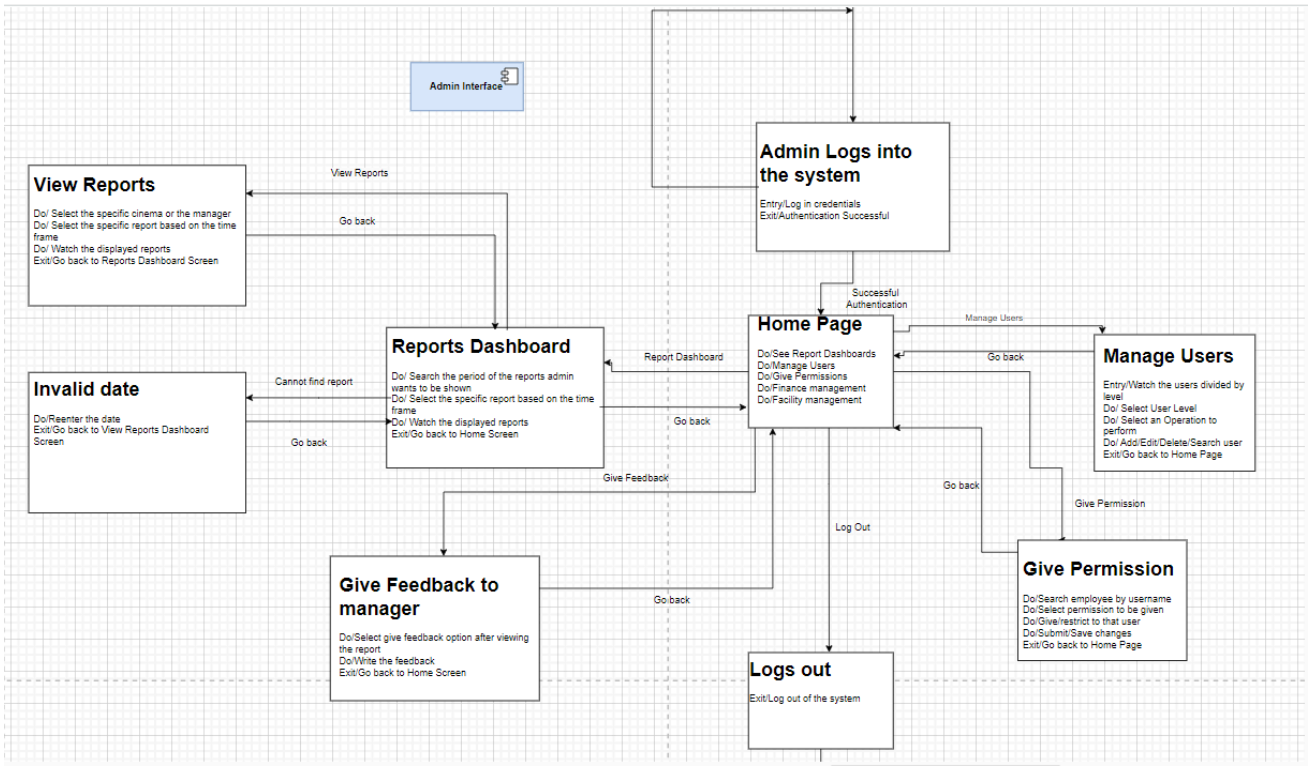

Staff member enquires the technical issue
Staff member opens the app
Staff member enters his credentials
Wrong Credentials
Valid Credentials
Checks Validity
Goes to report an issue section
Creates a new Issue
Enters a detailed description of the problem, including relevant information like error message, etc.
Technical Assistant gets notified via email by the system
Takes the necessary steps needed
Issue resolved
Issue not resolved
Staff member gets notified and gets back to work
Updates technical issue in the system
Collaborate with the manager



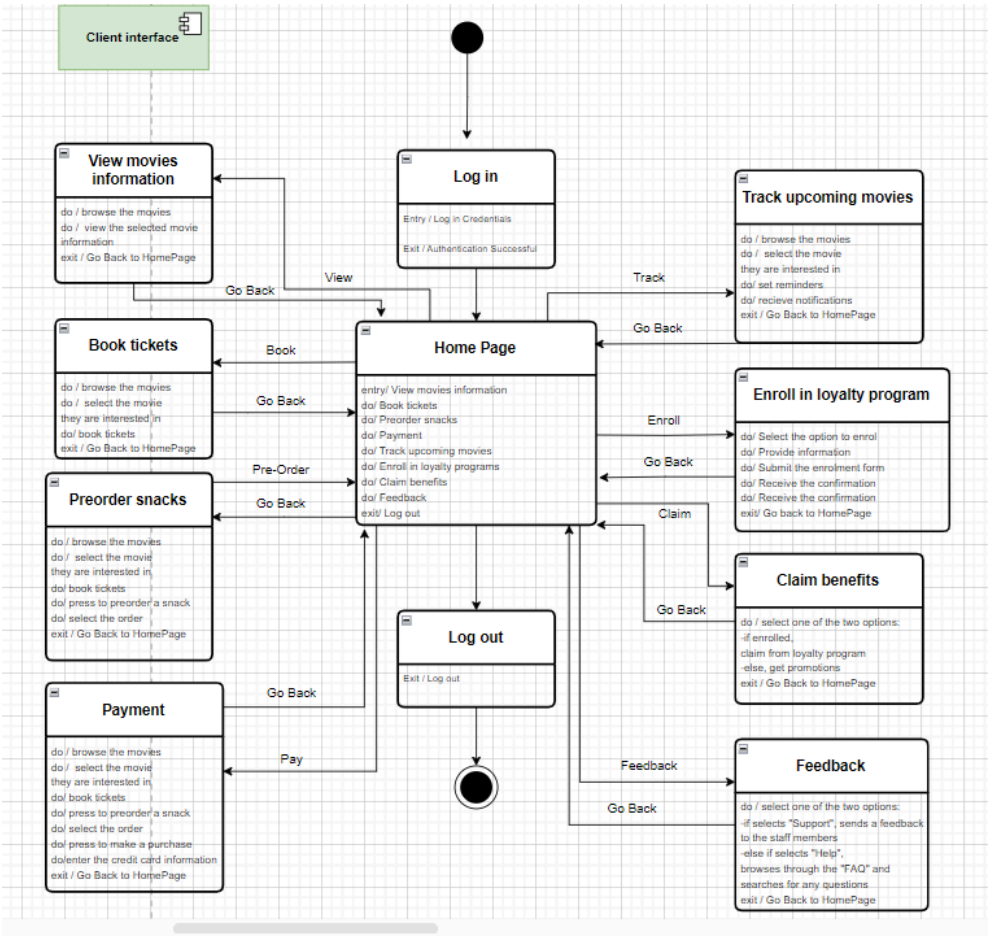
4.6 State Diagrams

https://app.diagrams.net/#G1meB0WYDs5drEXGsMbmOd44BlBobYQR3F#%7B%22pageId%22%3A%22jCCpQF_MnF1TEBN8lRtw%22%7D

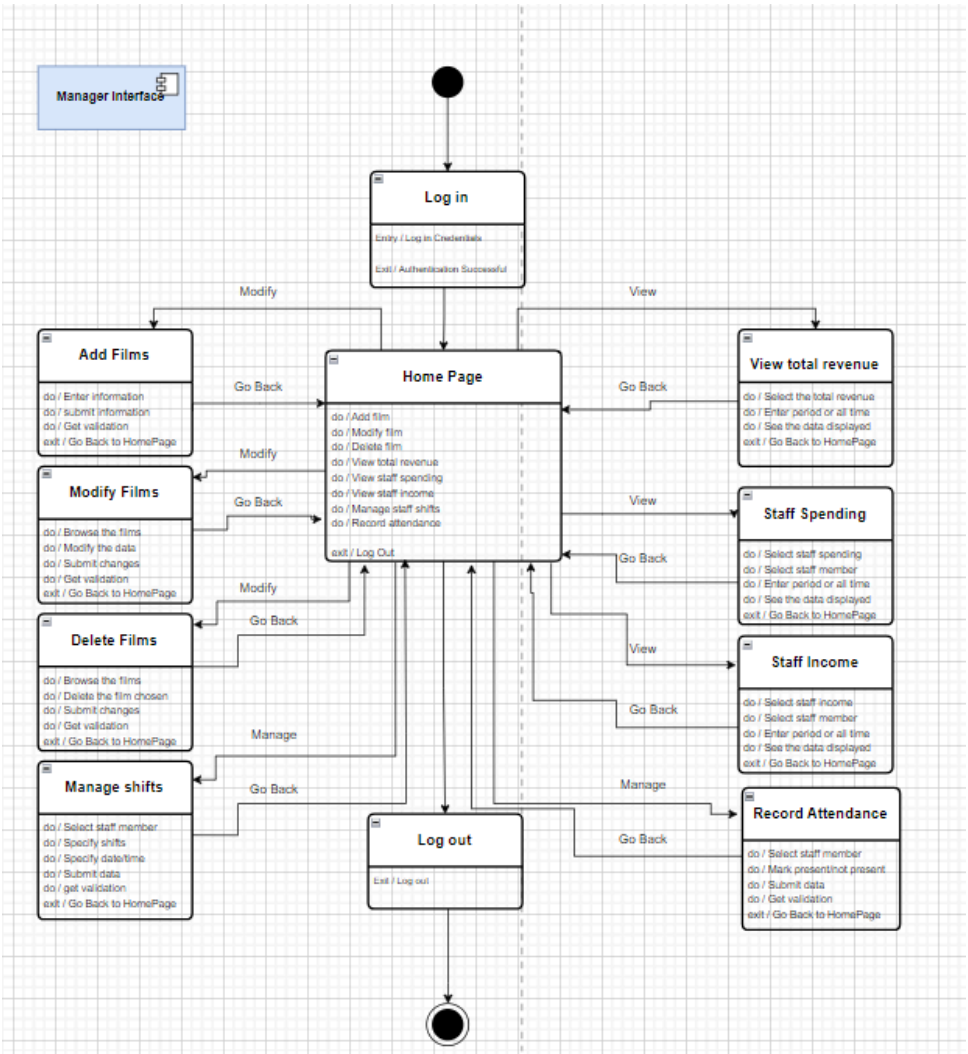
1.ADMIN


Admin Interface
Admin Logs into the system
Entry/Log in credentials
Exit/Authentication Successful
Successful Authentication
View Reports
Do/ Select the specific cinema or the manager
Do/ Select the specific report based on the time frame
Do/ Watch the displayed reports
Exit/Go back to Reports Dashboard Screen
View Reports
Go back
Reports Dashboard
Do/ Search the period of the reports admin wants to be shown
Do/ Select the specific report based on the time frame
Do/ Watch the displayed reports
Exit/Go back to Home Screen
Report Dashboard
Go back
Home Page
Do/See Report Dashboards
Do/Manage Users
Do/Give Permissions
Do/Finance management
Do/Facility management
Manage Users
Go back
Manage Users
Entry/Watch the users divided by level
Do/ Select User Level
Do/ Select an Operation to perform
Do/ Add/Edit/Delete/Search user
Exit/Go back to Home Page
Invalid date
Do/Reenter the date
Exit/Go back to View Reports Dashboard Screen
Cannot find report
Go back
Give Feedback
Go back
Give Permission
Log Out
Give Feedback to manager
Do/Select give feedback option after viewing the report
Do/Write the feedback
Exit/Go back to Home Screen
Go back
Give Permission
Do/Search employee by username
Do/Select permission to be given
Do/Give/restrict to that user
Do/Submit/Save changes
Exit/Go back to Home Page
Logs out
Exit/Log out of the system

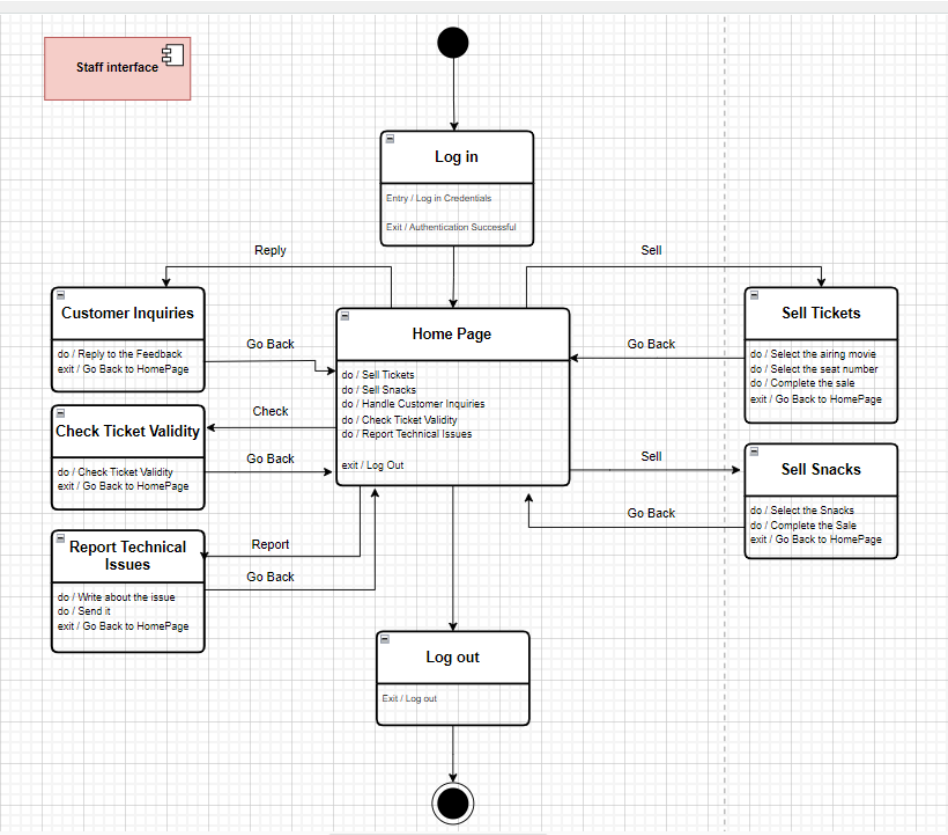
2.CLIENT


Client interface
View movies information
do / browse the movies
do / view the selected movie information
exit / Go Back to HomePage
Log in
Entry / Log in Credentials
Exit / Authentication Successful
Track upcoming movies
do / browse the movies
do / select the movie they are interested in
do/ set reminders
do/ recieve notifications
exit / Go Back to HomePage
Go Back
View
Track
Go Back
Book tickets
do / browse the movies
do / select the movie they are interested in
do/ book tickets
exit / Go Back to HomePage
Book
Home Page
entry/ View movies information
do/ Book tickets
do/ Preorder snacks
do/ Payment
do/ Track upcoming movies
do/ Enroll in loyalty programs
do/ Claim benefits
do/ Feedback
exit/ Log out
Enroll in loyalty program
do/ Select the option to enrol
do/ Provide information
do/ Submit the enrolment form
do/ Receive the confirmation
do/ Receive the confirmation
exit/ Go back to HomePage
Go Back
Enroll
Go Back
Pre-Order
Go Back
Preorder snacks
do / browse the movies
do / select the movie they are interested in
do/ book tickets
do/ press to preorder a snack
do/ select the order
exit / Go Back to HomePage
Claim
Claim benefits
do / select one of the two options:
-if enrolled,
claim from loyalty program
-else, get promotions
exit / Go Back to HomePage
Go Back
Log out
Exit / Log out
Payment
do / browse the movies
do / select the movie they are interested in
do/ book tickets
do/ press to preorder a snack
do/ select the order
do/ press to make a purchase
do/enter the credit card information
exit / Go Back to HomePage
Go Back
Pay
Feedback
Feedback
do / select one of the two options:
-if selects "Support", sends a feedback to the staff members
-else if selects "Help",
browses through the "FAQ" and searches for any questions
exit / Go Back to HomePage
Go Back

3.MANAGER


Manager Interface
Log in
Entry / Log in Credentials
Exit / Authentication Successful
Modify
View
Add Films
do / Enter information
do / submit information
do / Get validation
exit / Go Back to HomePage
Go Back
Home Page
do / Add film
do / Modify film
do / Delete film
do / View total revenue
do / View staff spending
do / View staff income
do / Manage staff shifts
do / Record attendance
exit / Log Out
Go Back
View total revenue
do / Select the total revenue
do / Enter period or all time
do / See the data displayed
exit / Go Back to HomePage
Modify
Modify Films
do / Browse the films
do / Modify the data
do / Submit changes
do / Get validation
exit / Go Back to HomePage
Go Back
View
Staff Spending
do / Select staff spending
do / Select staff member
do / Enter period or all time
do / See the data displayed
exit / Go Back to HomePage
Go Back
Modify
Delete Films
do / Browse the films
do / Delete the film chosen
do / Submit changes
do / Get validation
exit / Go Back to HomePage
Go Back
View
Staff Income
do / Select staff income
do / Select staff member
do / Enter period or all time
do / See the data displayed
exit / Go Back to HomePage
Go Back
Manage
Manage shifts
do / Select staff member
do / Specify shifts
do / Specify date/time
do / Submit data
do / get validation
exit / Go Back to HomePage
Go Back
Manage
Record Attendance
do / Select staff member
do / Mark present/not present
do / Submit data
do / Get validation
exit / Go Back to HomePage
Go Back
Log out
Exit / Log out

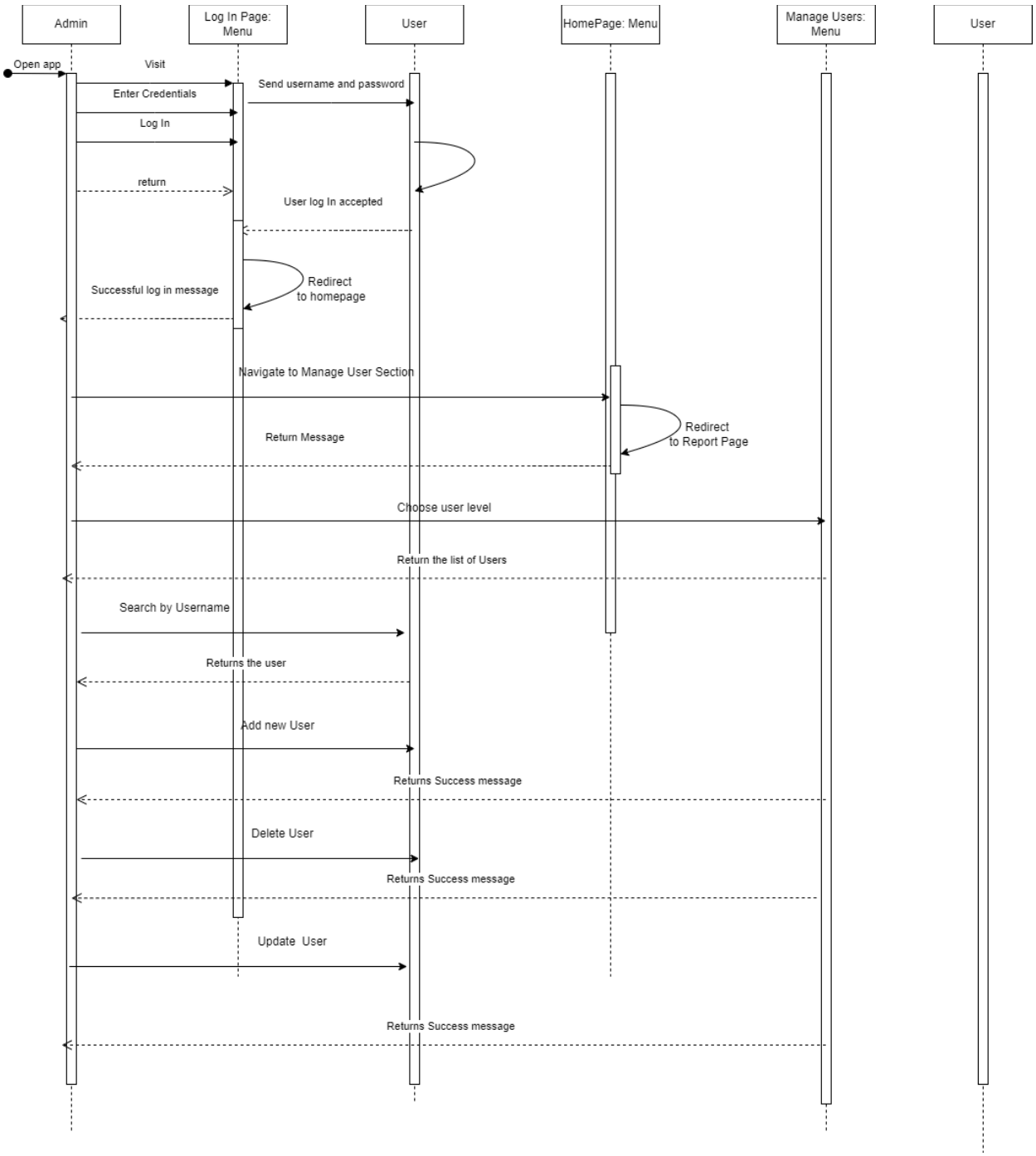
4.STAFF



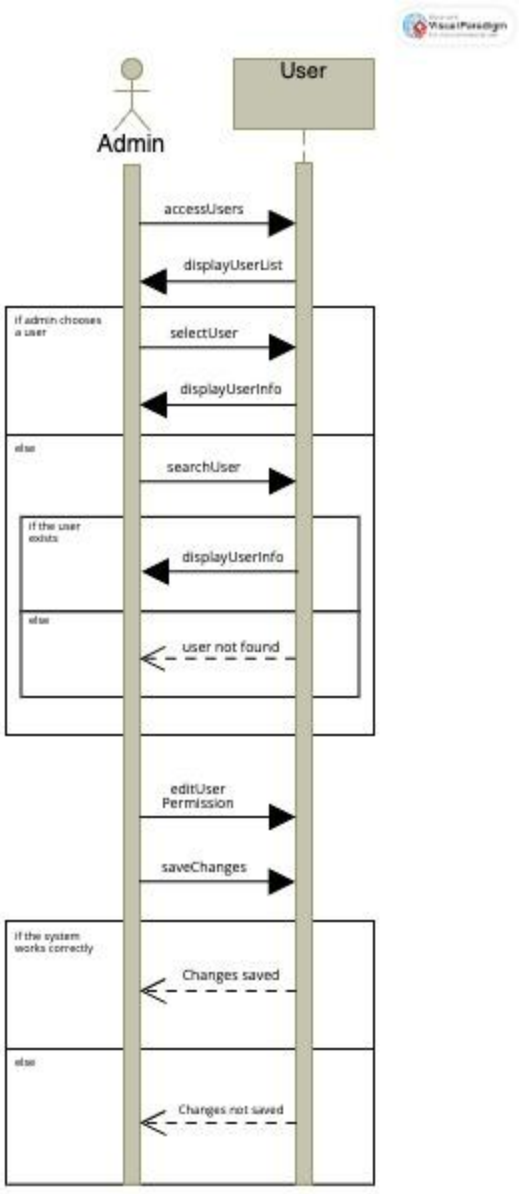
4.7 Sequence Diagrams

1.ADMIN

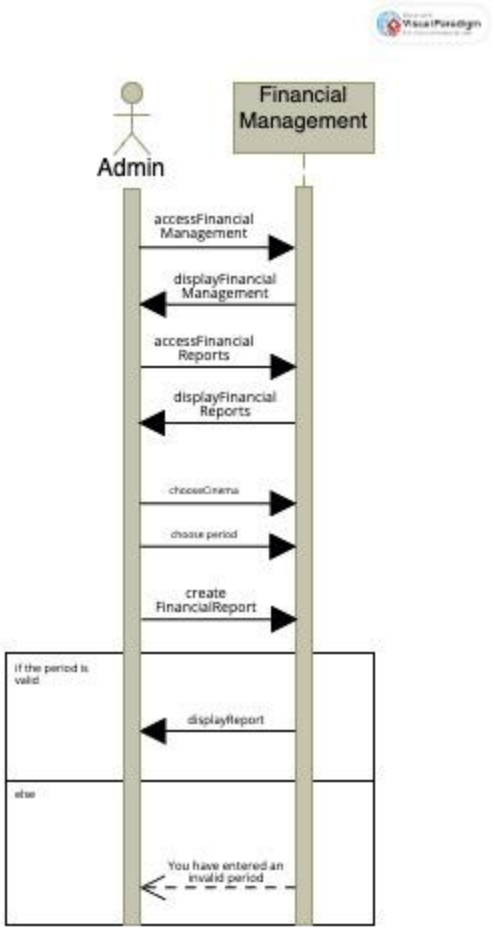
1-CRUD Operations



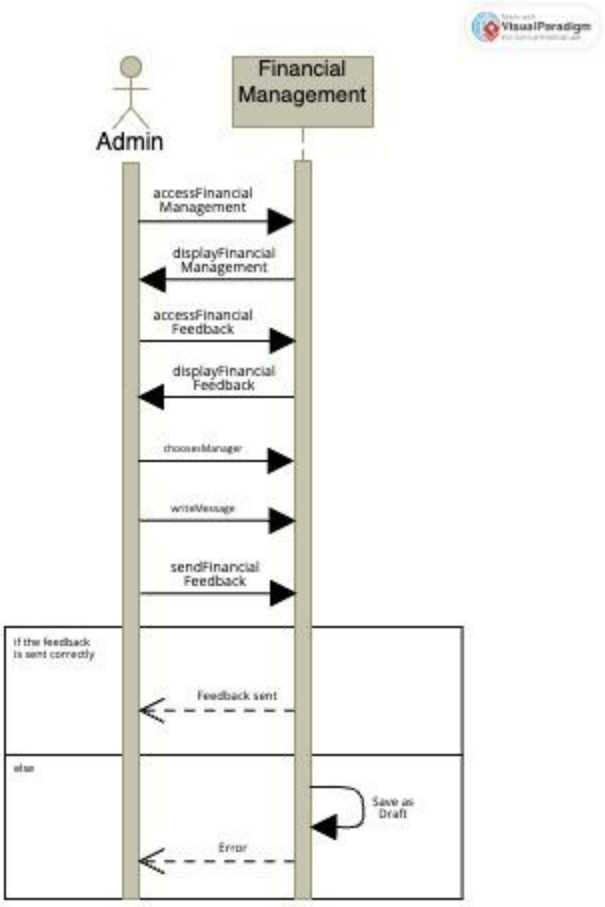
2 - Changing specific User Permissions



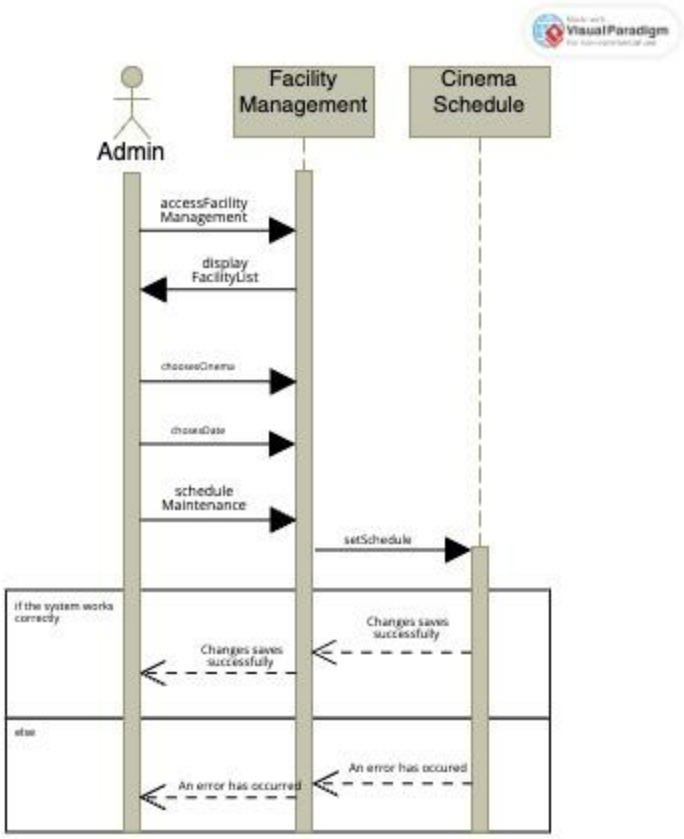
3 - View Financial Reports



4 - Financial Feedback

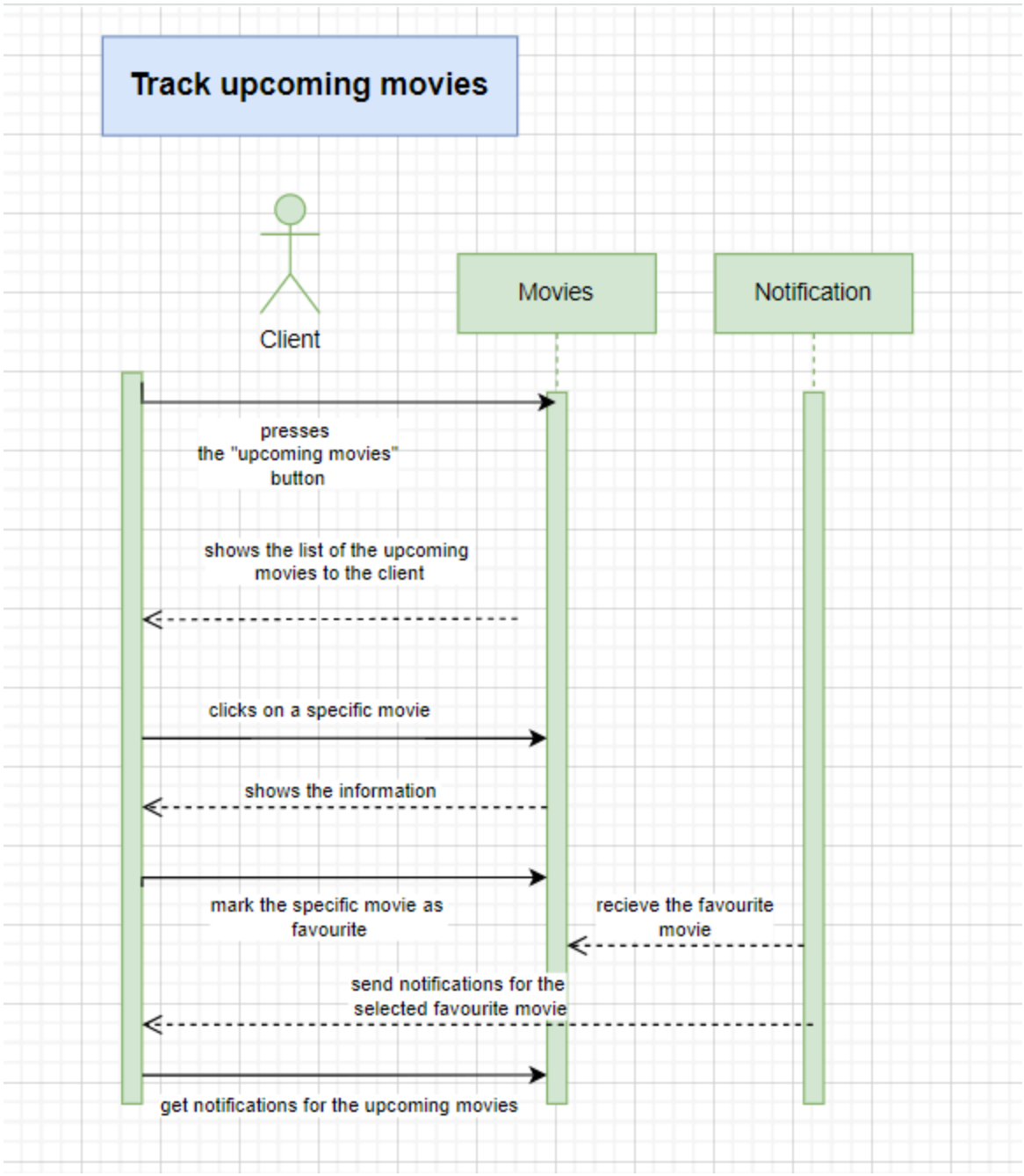


5 - Managing Cinema Facilities

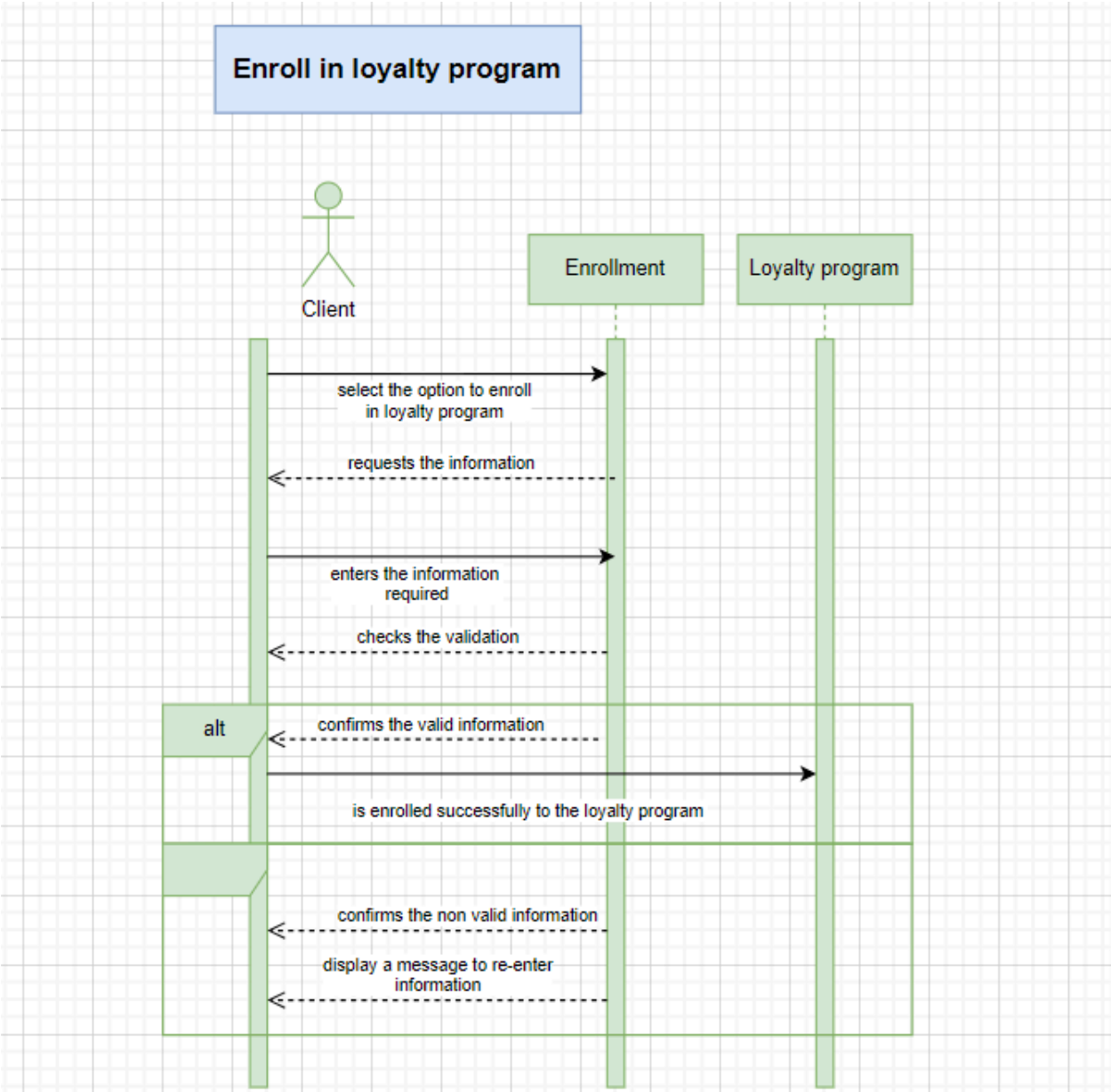


2.CLIENT

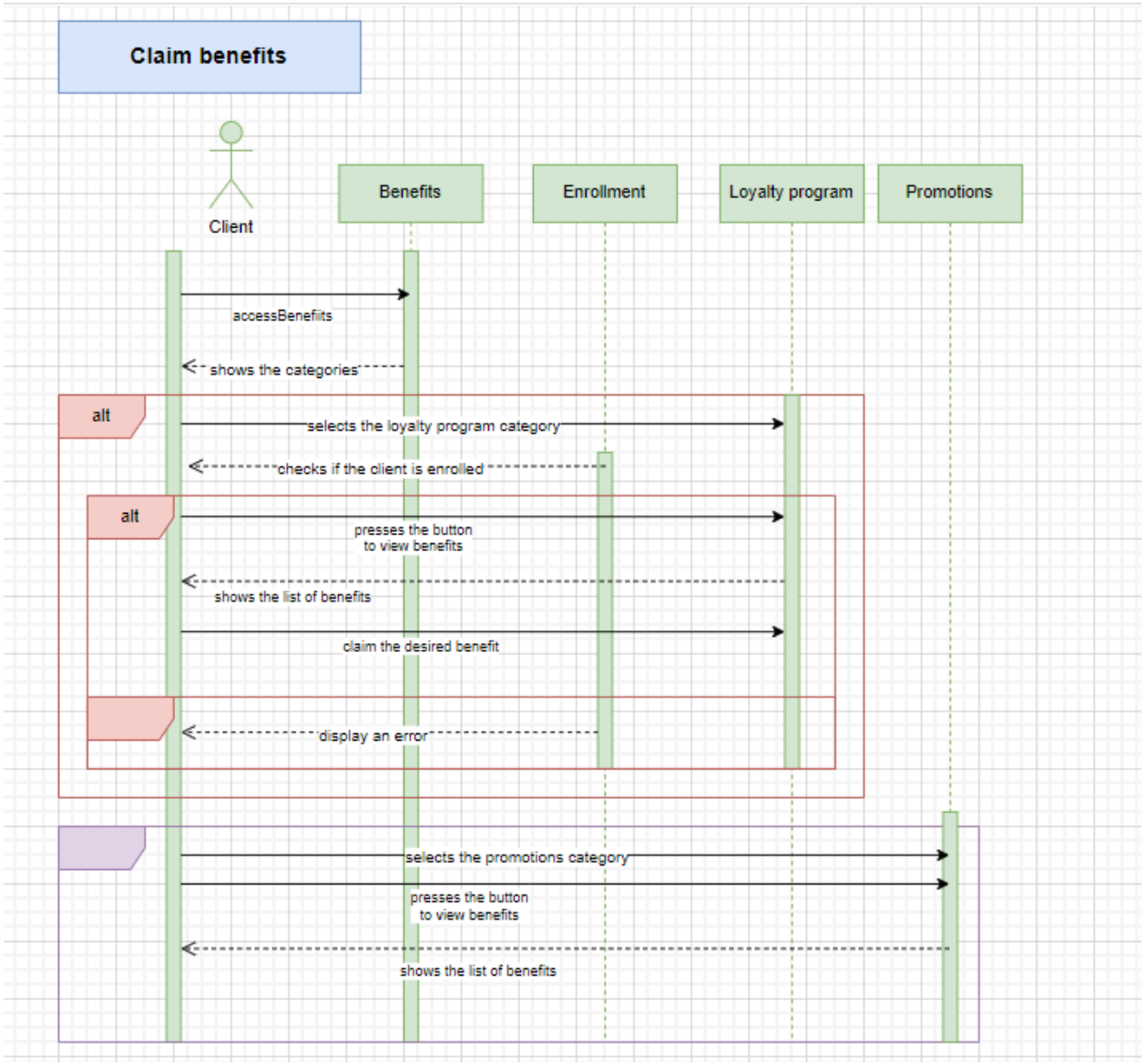
1 - Track upcoming movies



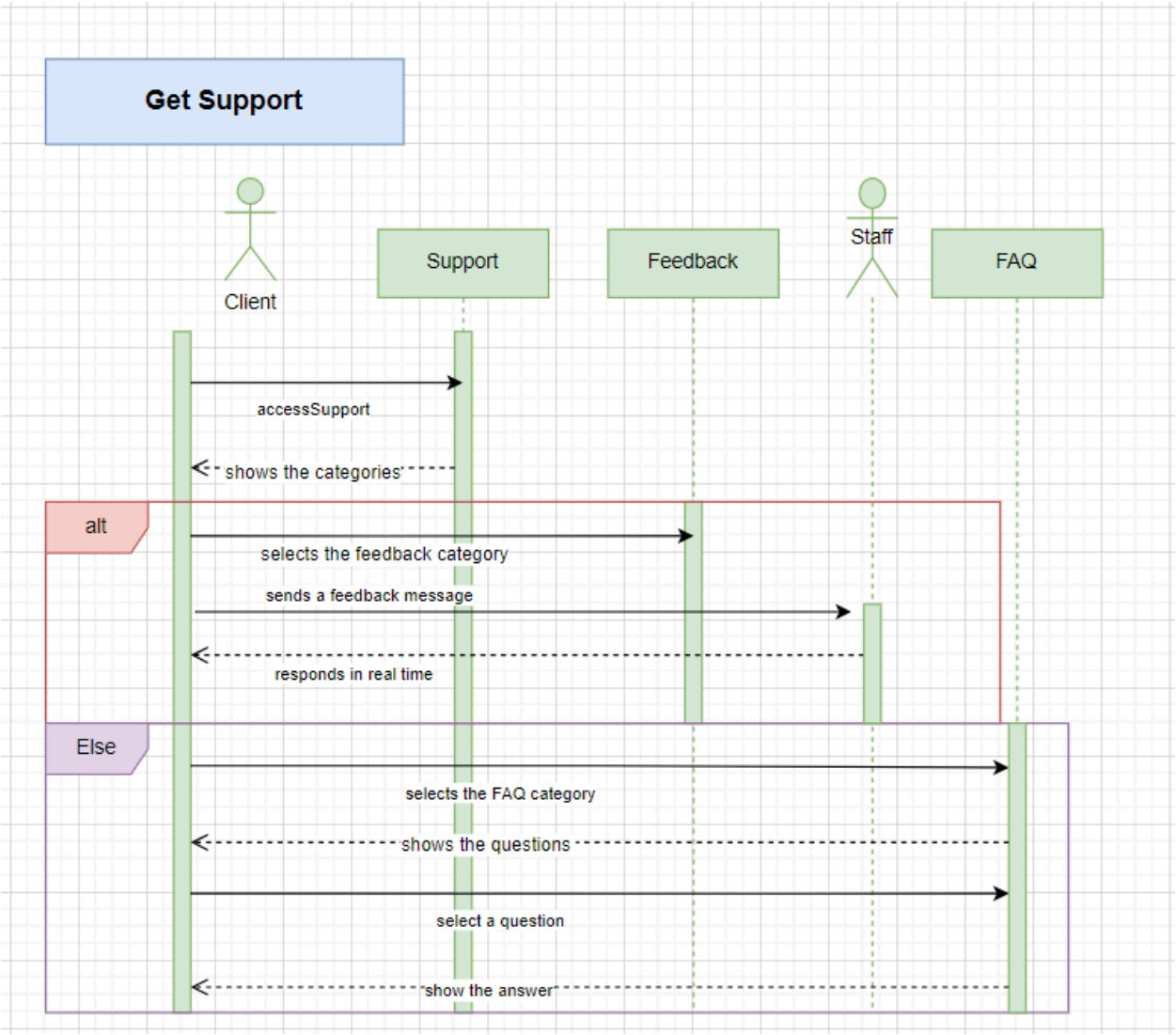
2 - Enroll in Loyalty Program



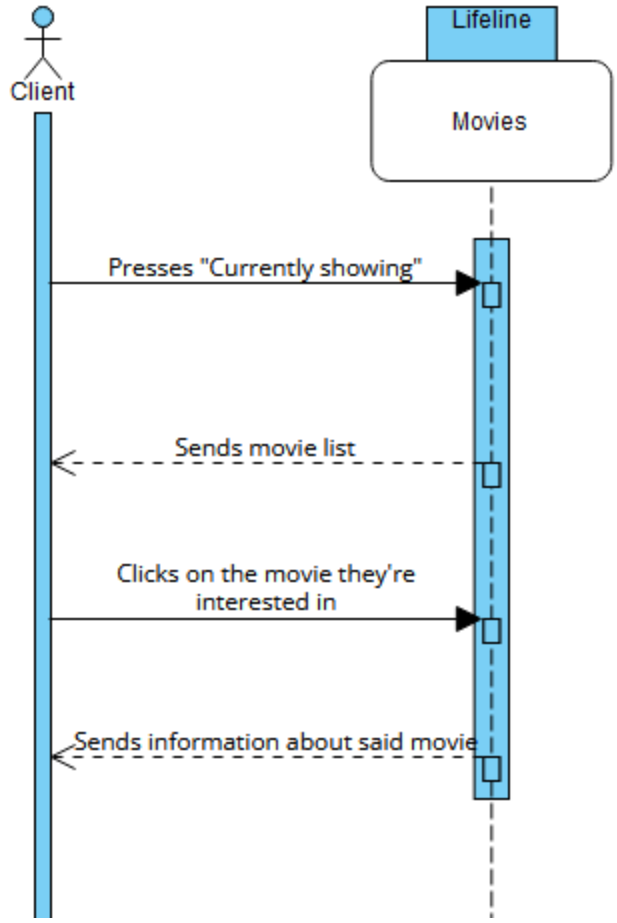
3 - Claim Benefits



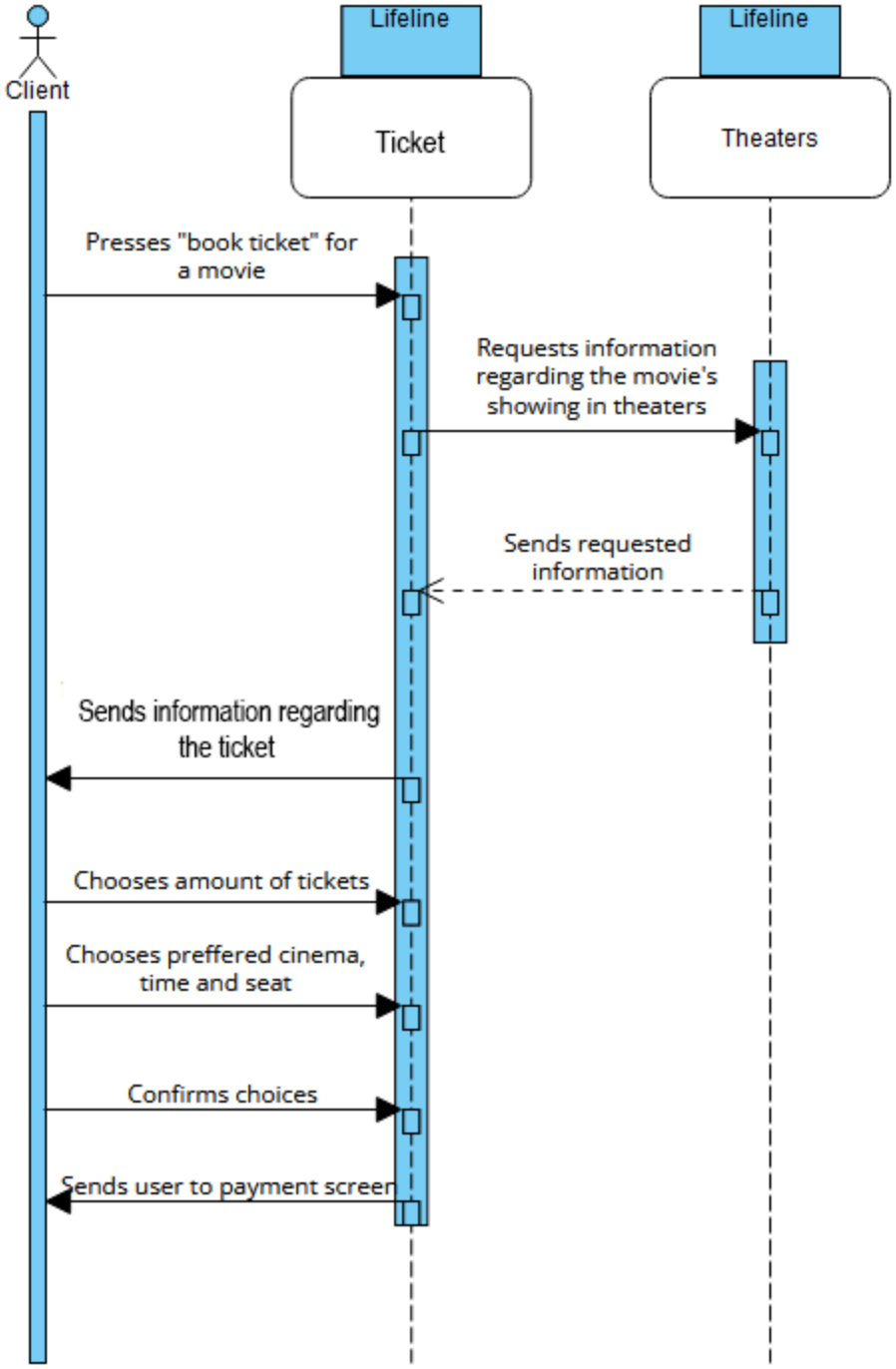
4 - Get Support



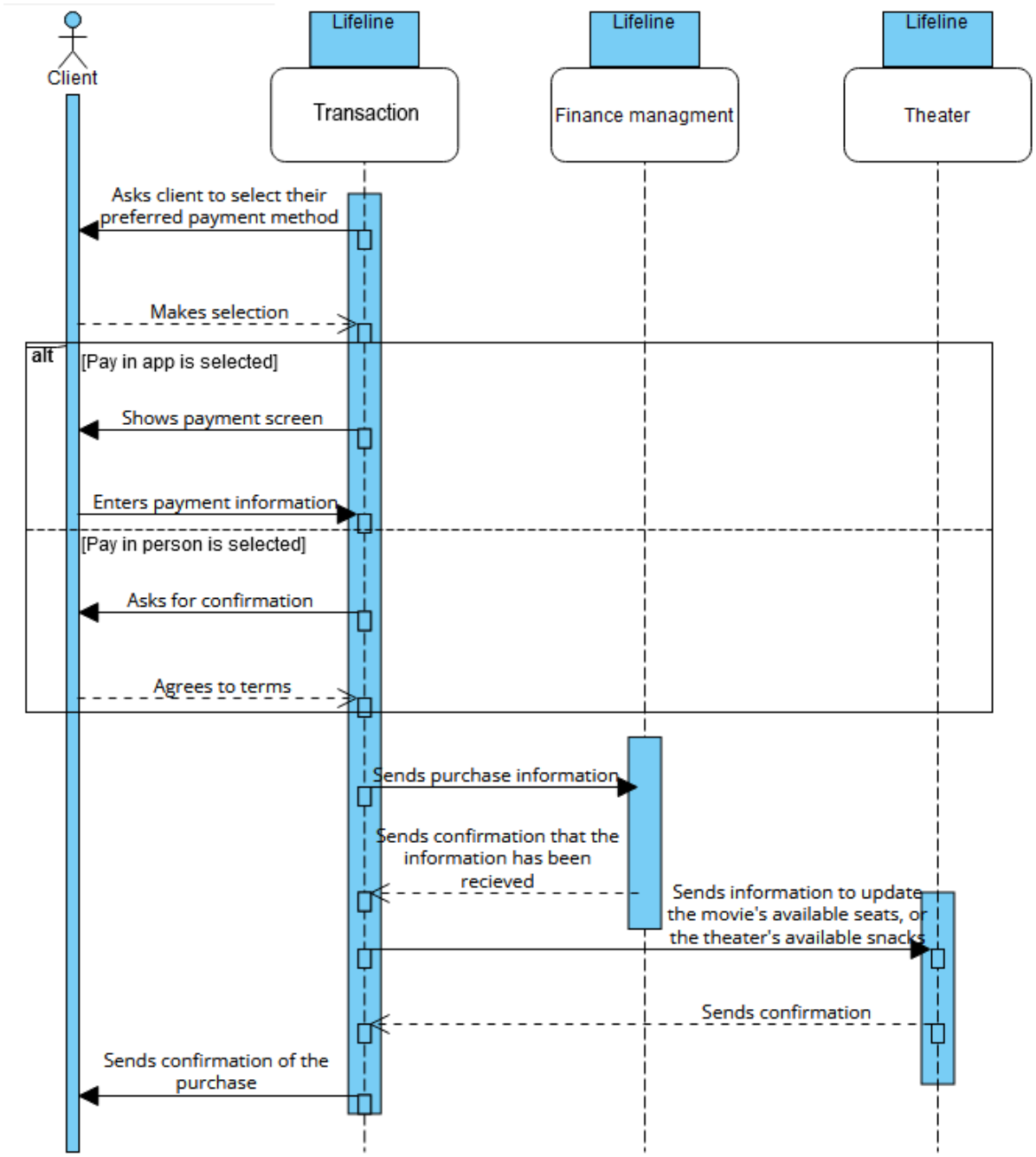
5- See current and upcoming movies



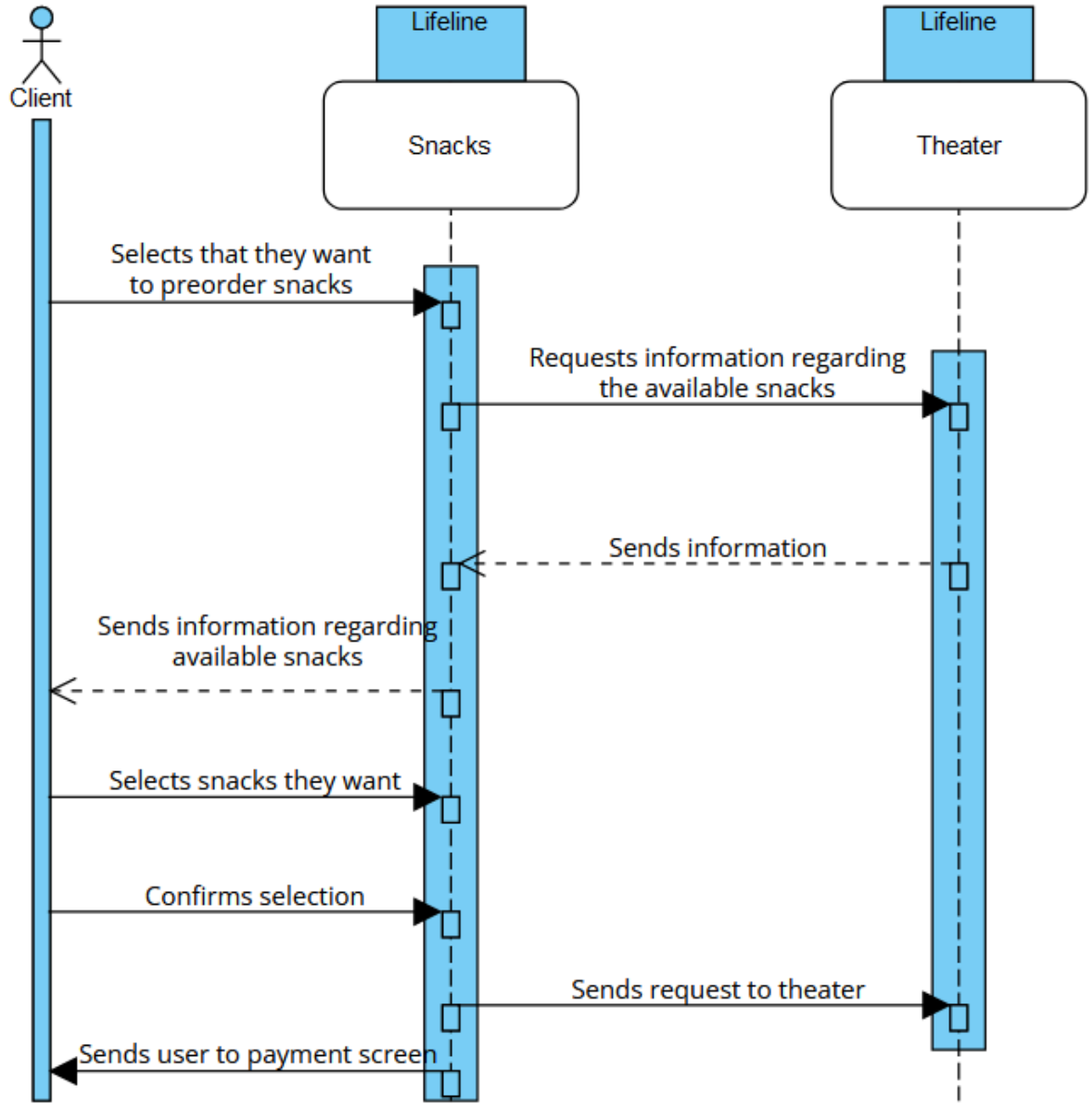
6-Book tickets in app


Client
Lifeline
Ticket
Lifeline
Theaters
Presses "book ticket" for a movie
Requests information regarding the movie's showing in theaters
Sends requested information
Sends information regarding the ticket
Chooses amount of tickets
Chooses preffered cinema, time and seat
Confirms choices
Sends user to payment screen

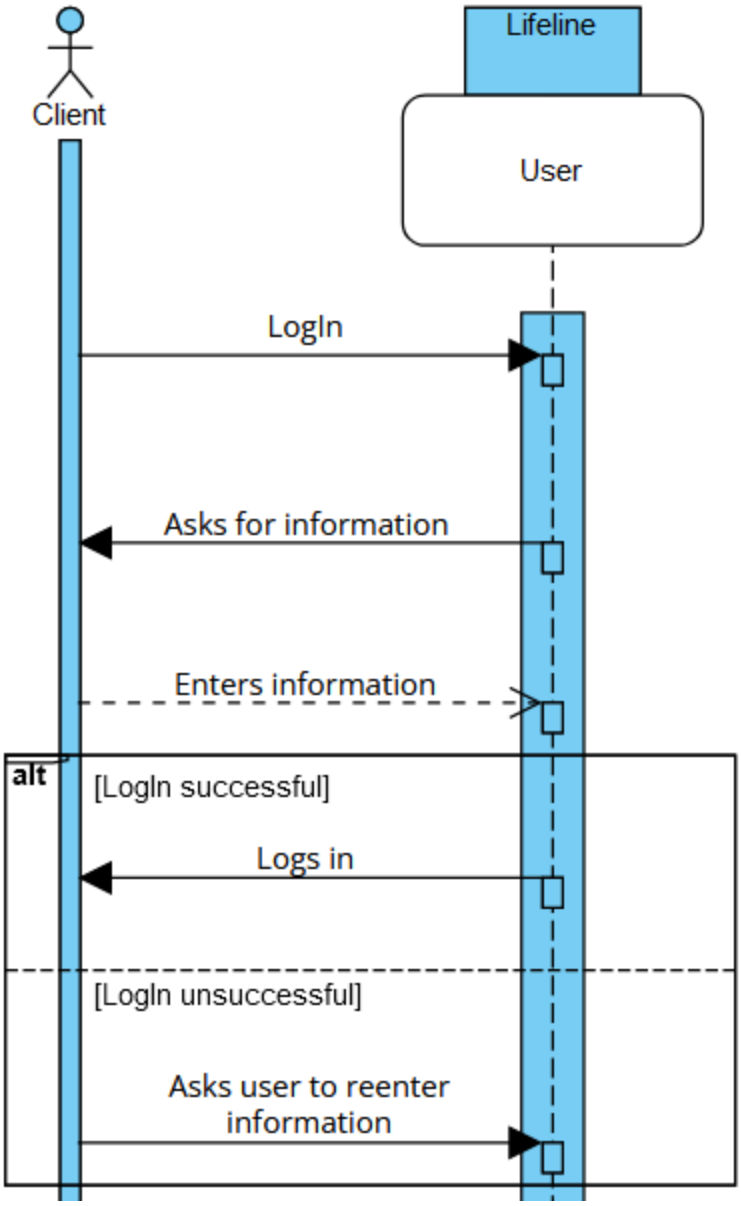
7-Pay in app



8- preorder snacks

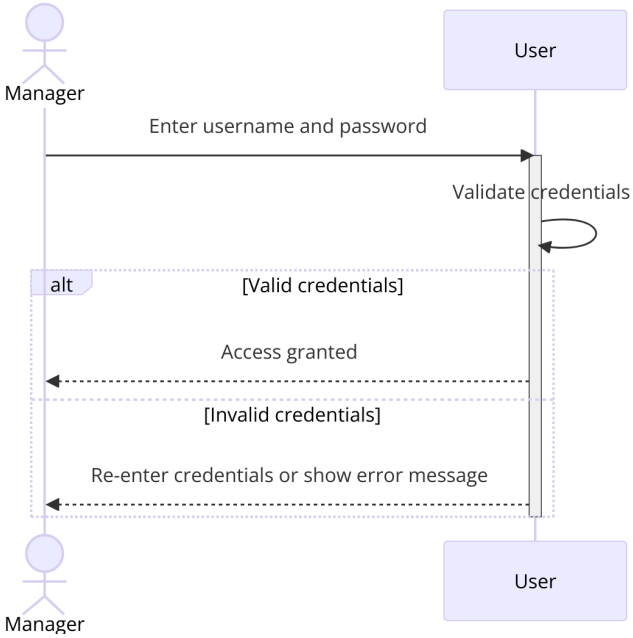

Client
Lifeline
Snacks
Lifeline
Theater
Selects that they want to preorder snacks
Requests information regarding the available snacks
Sends information
Sends information regarding available snacks
Selects snacks they want
Confirms selection
Sends request to theater
Sends user to payment screen

9- log in

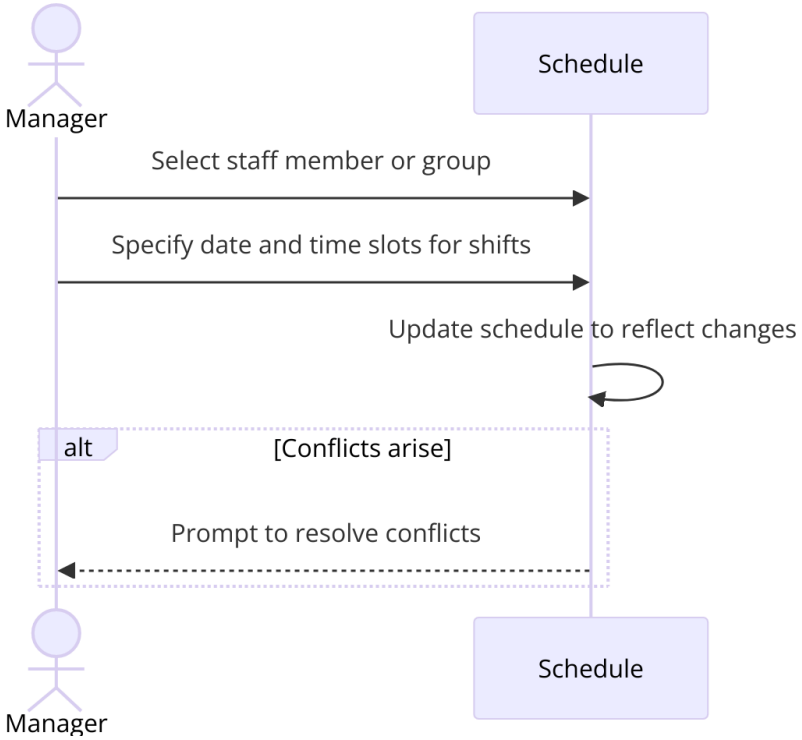


3.MANAGER

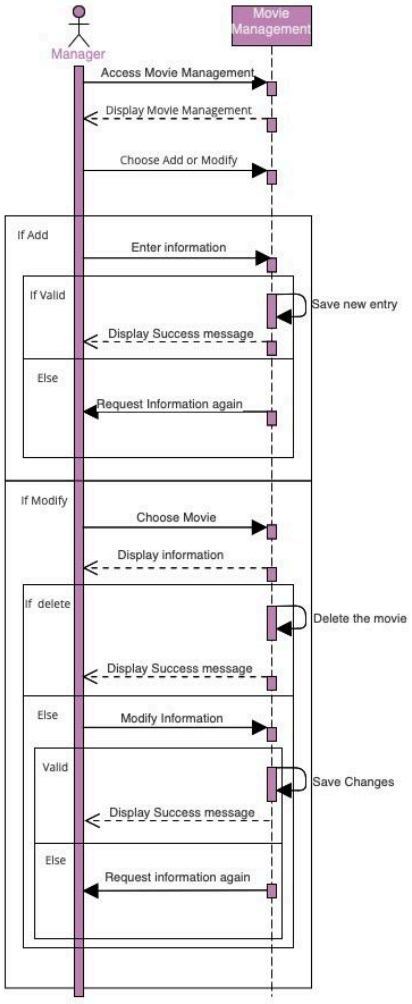
User login:



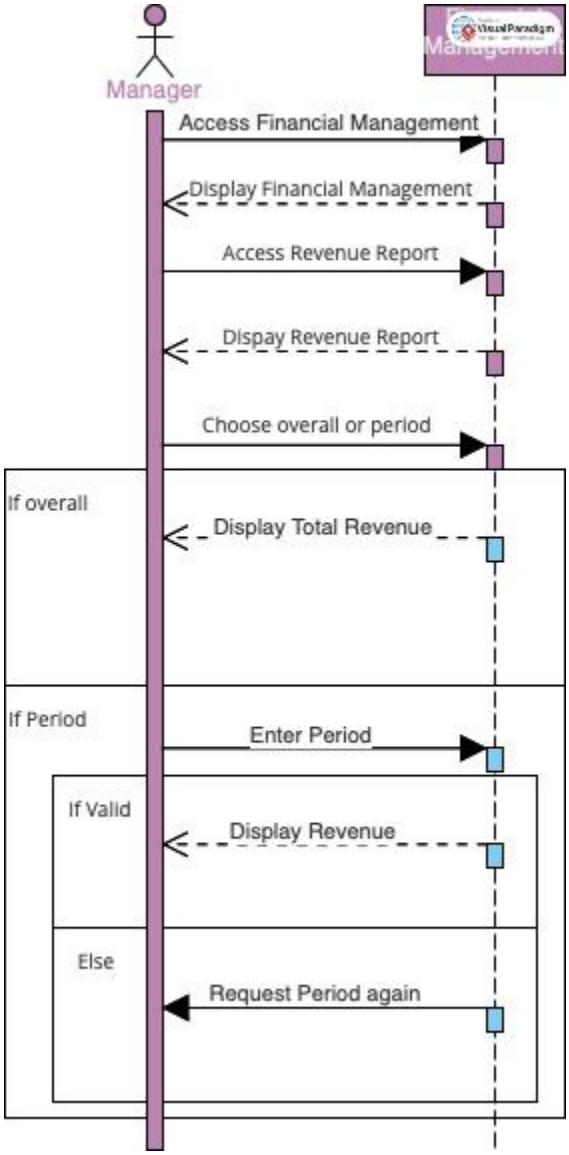
Create/Edit Schedule:



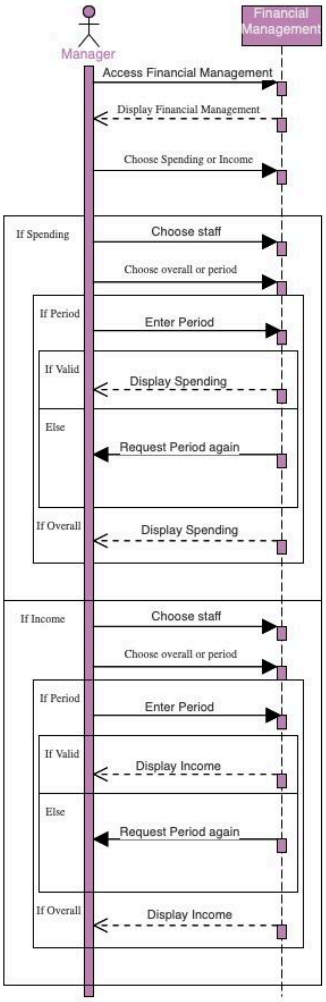
Modify movies:



Overall revenue:

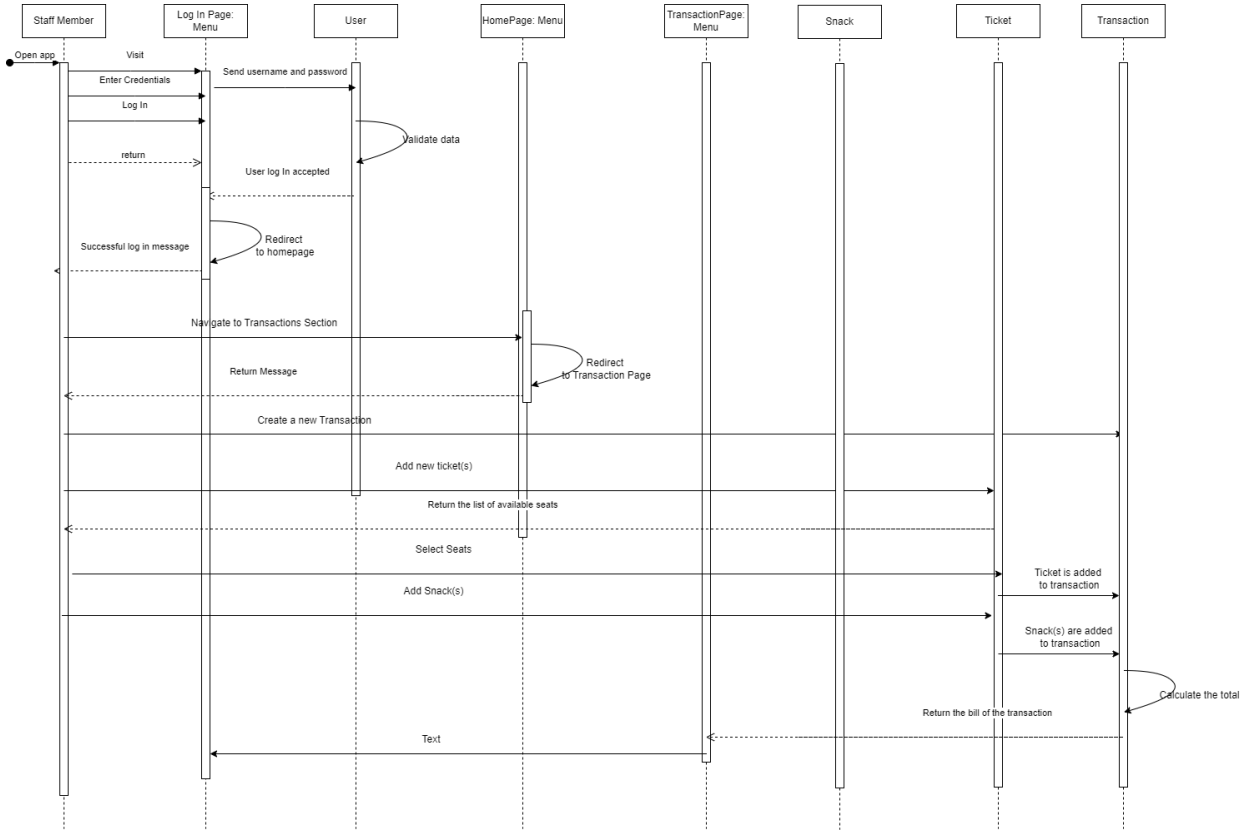


Staff spending/income:

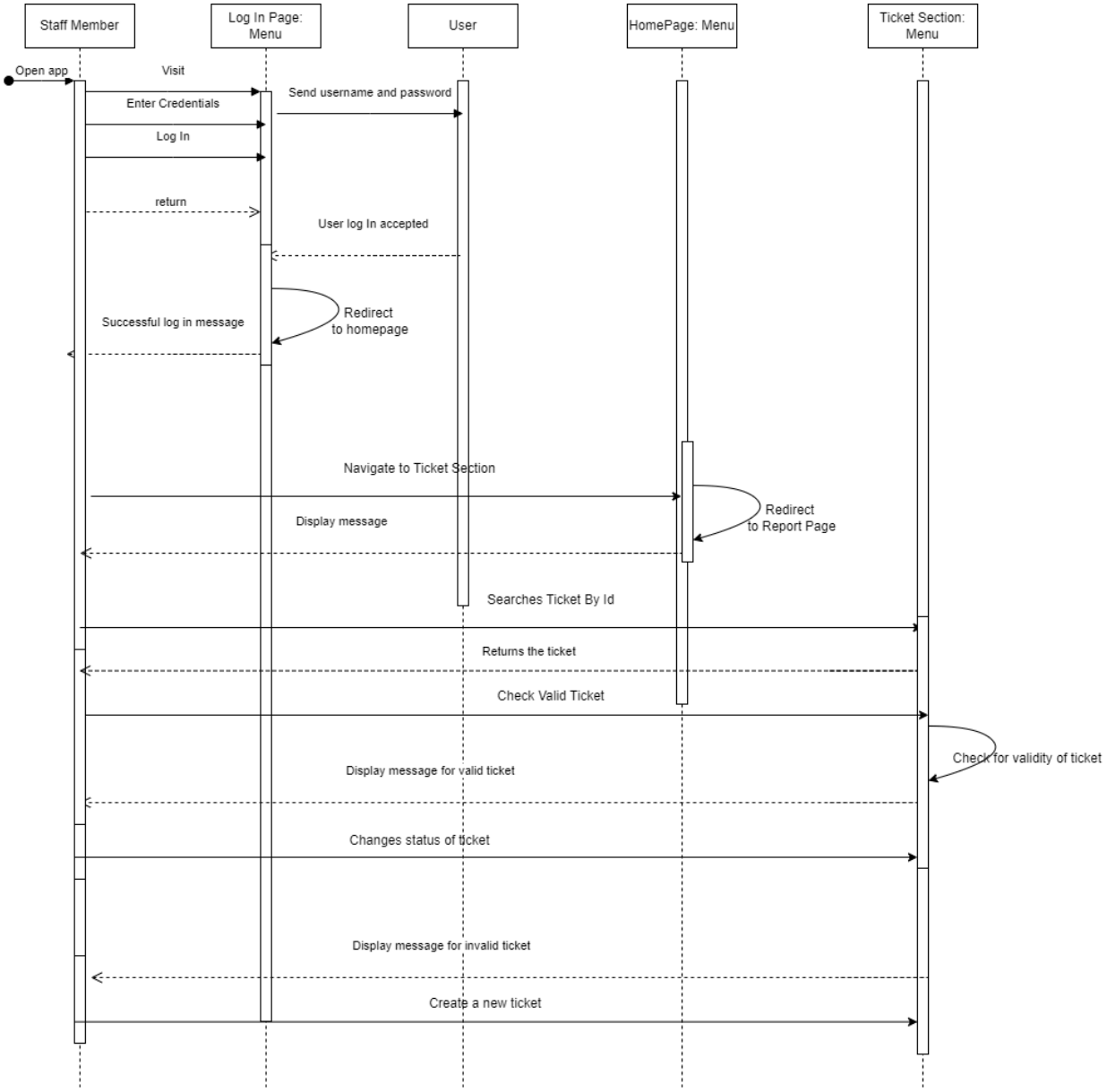


4.STAFF

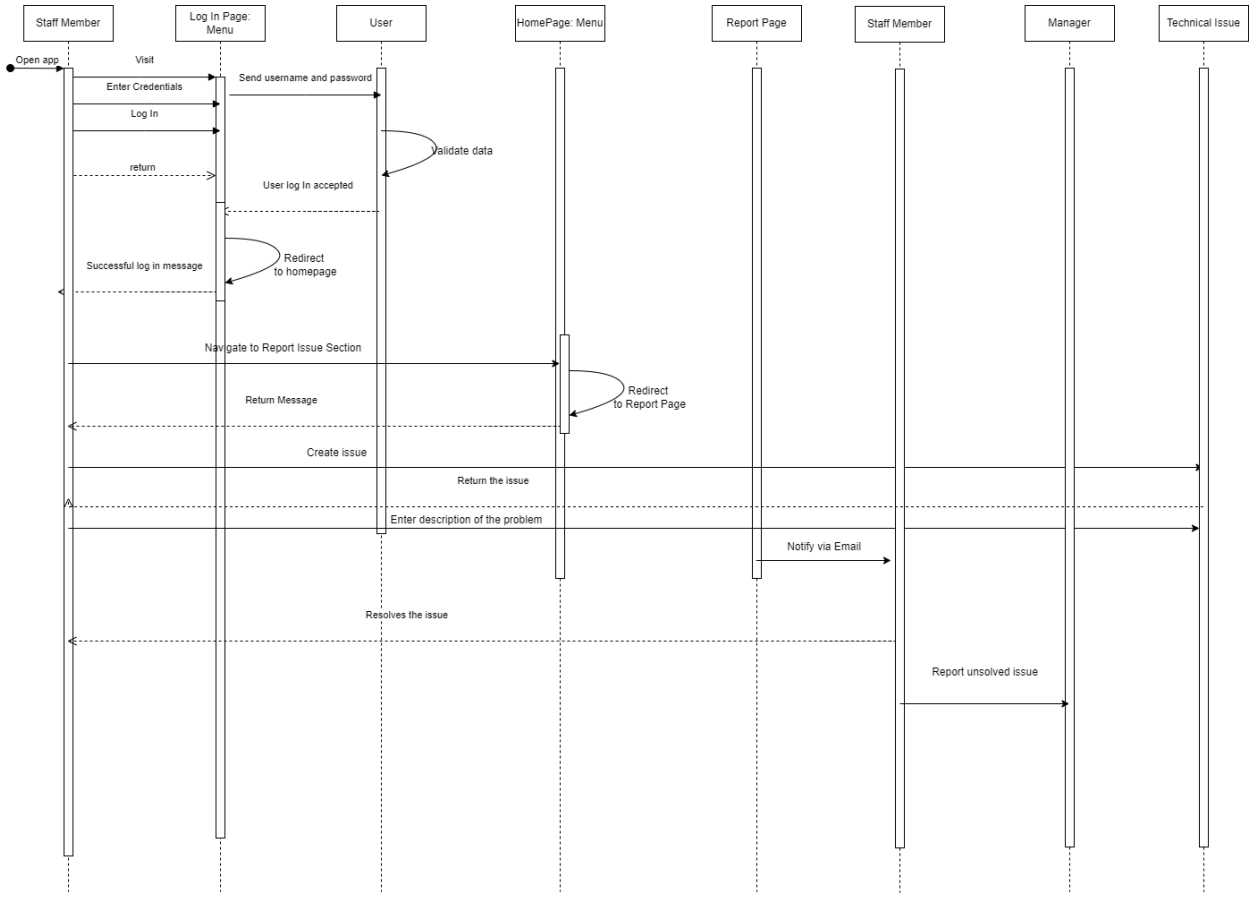
1-Create Transaction

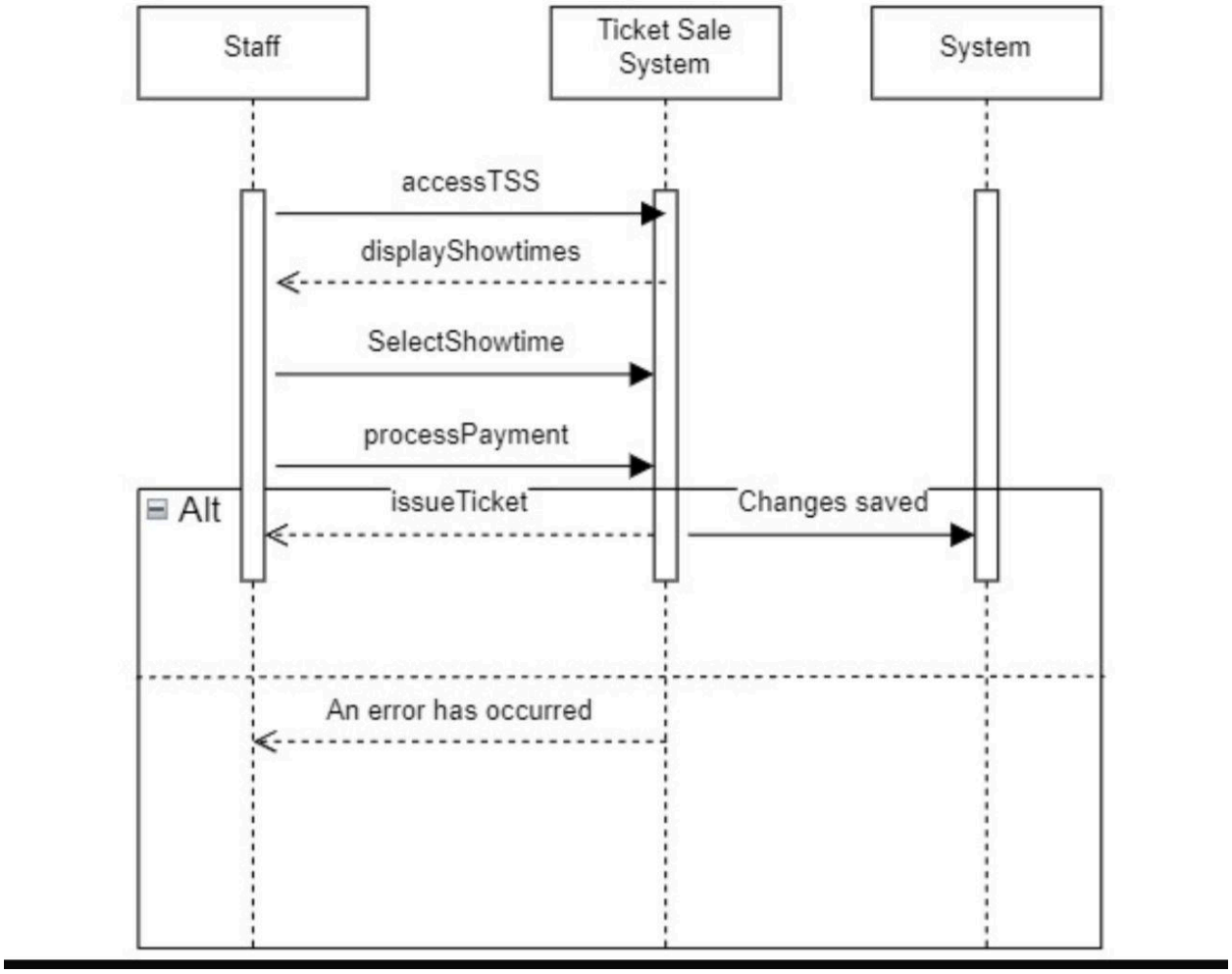


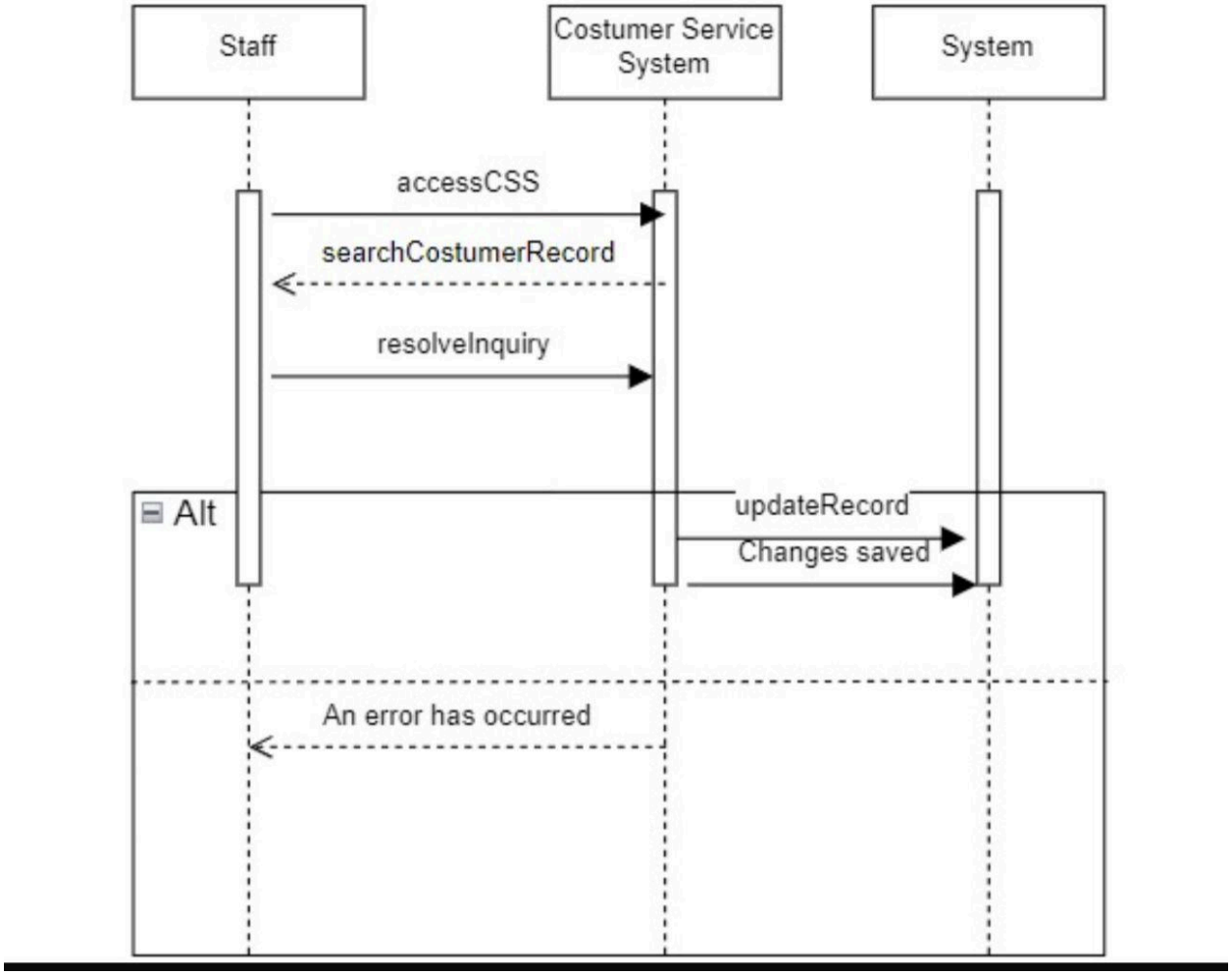
2- Validate Ticket



3-Technical Issue


Staff Member
Log In Page: Menu
User
HomePage: Menu
Report Page
Staff Member
Manager
Technical Issue
Open app
Visit
Enter Credentials
Log In
return
Send username and password
Validate data
User log In accepted
Successful log in message
Redirect to homepage
Navigate to Report Issue Section
Return Message
Redirect to Report Page
Create issue
Return the issue
Enter description of the problem
Notify via Email
Resolves the issue
Report unsolved issue

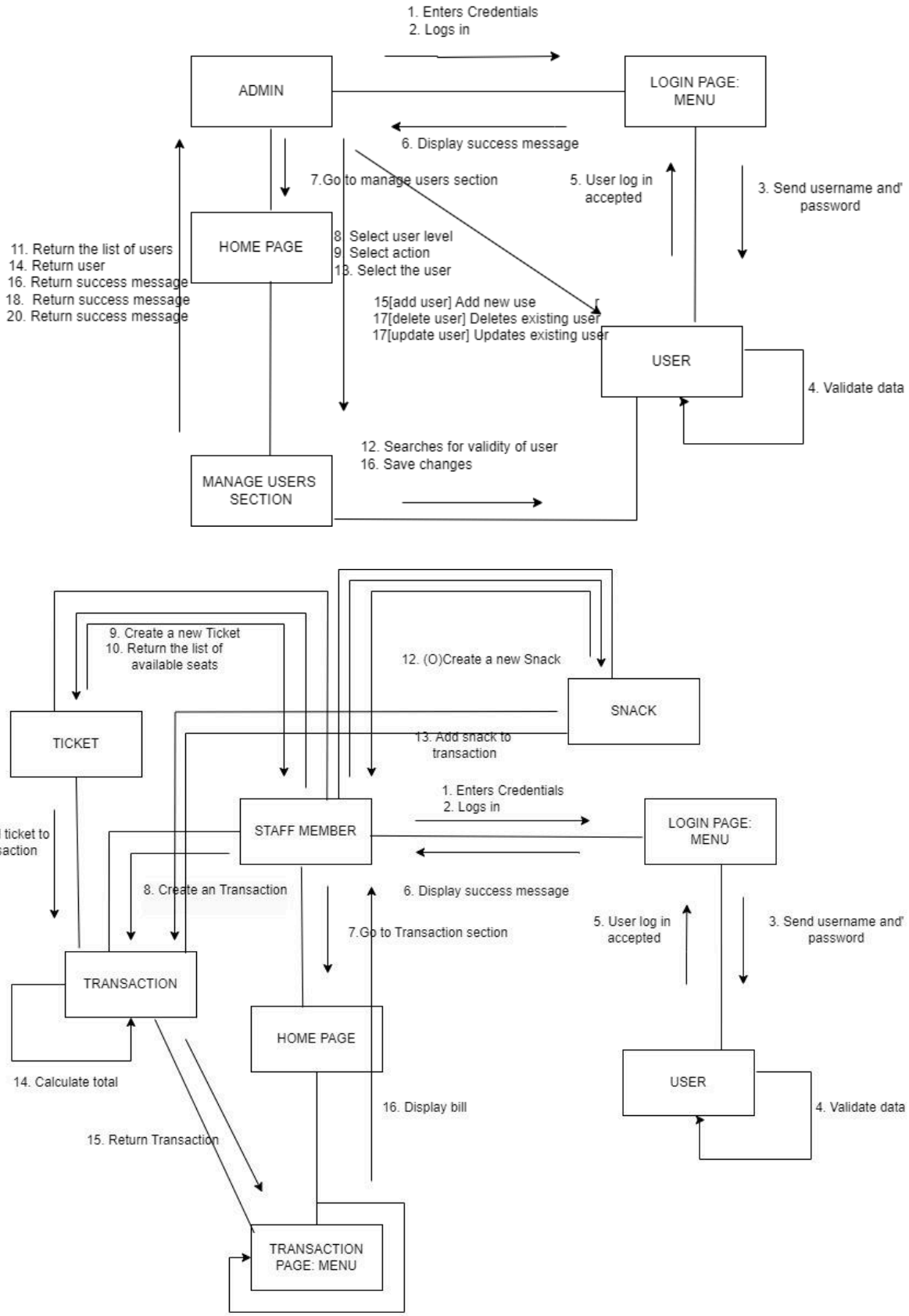

Staff
Ticket Sale System
System
accessTSS
displayShowtimes
SelectShowtime
processPayment
Alt
issueTicket
Changes saved
An error has occurred



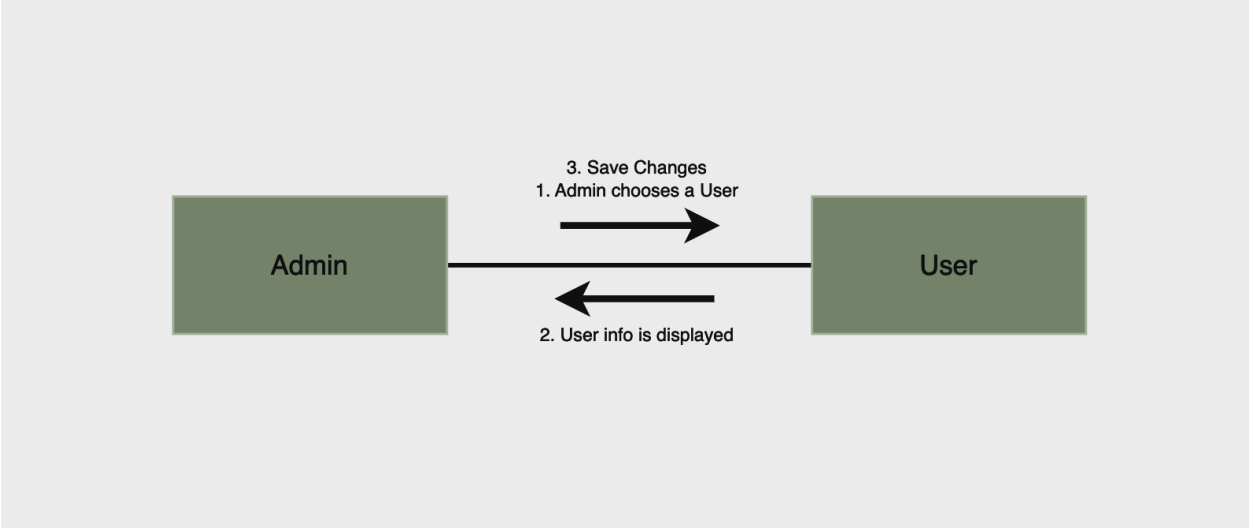
4.8 Collaboration Diagrams

1.ADMIN

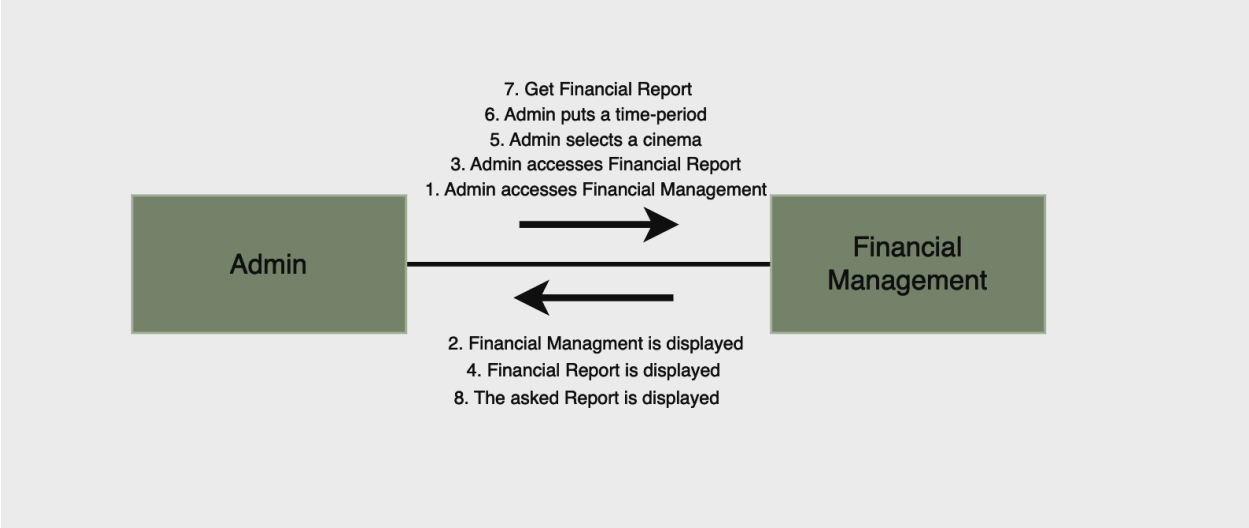
1-CRUD Operations


1. Enters Credentials
2. Logs in
ADMIN
LOGIN PAGE: MENU
6. Display success message
7.Go to manage users section
5. User log in accepted
3. Send username and' password
HOME PAGE
8. Select user level
9. Select action
13. Select the user
11. Return the list of users
14. Return user
16. Return success message
18. Return success message
20. Return success message
15[add user] Add new use
17[delete user] Deletes existing user
17[update user] Updates existing user
USER
4. Validate data
12. Searches for validity of user
16. Save changes
MANAGE USERS SECTION
9. Create a new Ticket
10. Return the list of available seats
12. (O)Create a new Snack
SNACK
TICKET
13. Add snack to transaction
1. Enters Credentials
2. Logs in
11. Add ticket to transaction
STAFF MEMBER
LOGIN PAGE: MENU
8. Create an Transaction
6. Display success message
7.Go to Transaction section
5. User log in accepted
3. Send username and' password
TRANSACTION
HOME PAGE
USER
14. Calculate total
16. Display bill
4. Validate data
15. Return Transaction
TRANSACTION PAGE: MENU

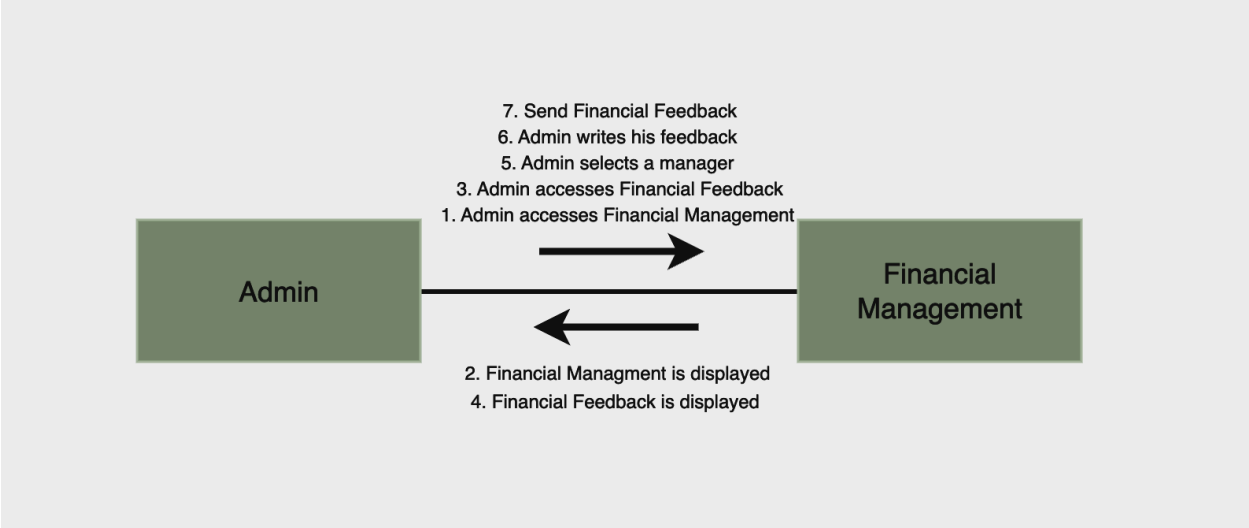
2 - Changing specific User Permissions



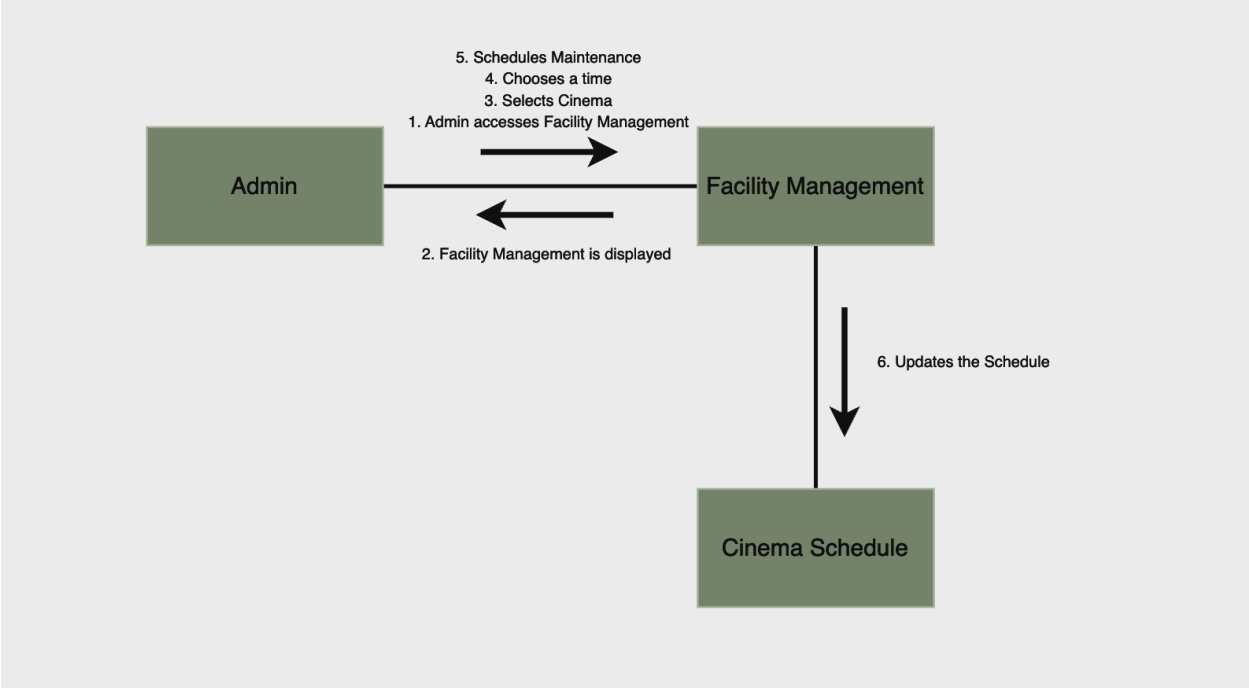
3 - View Financial Reports



4 - Financial Feedback

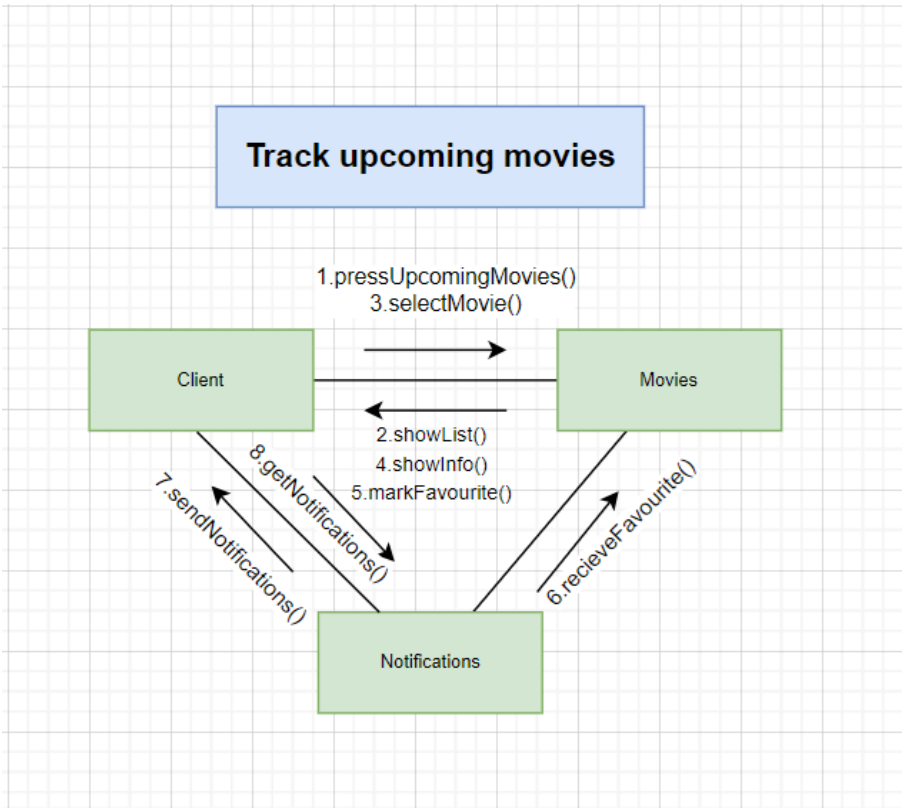


5 - Managing Cinema Facilities

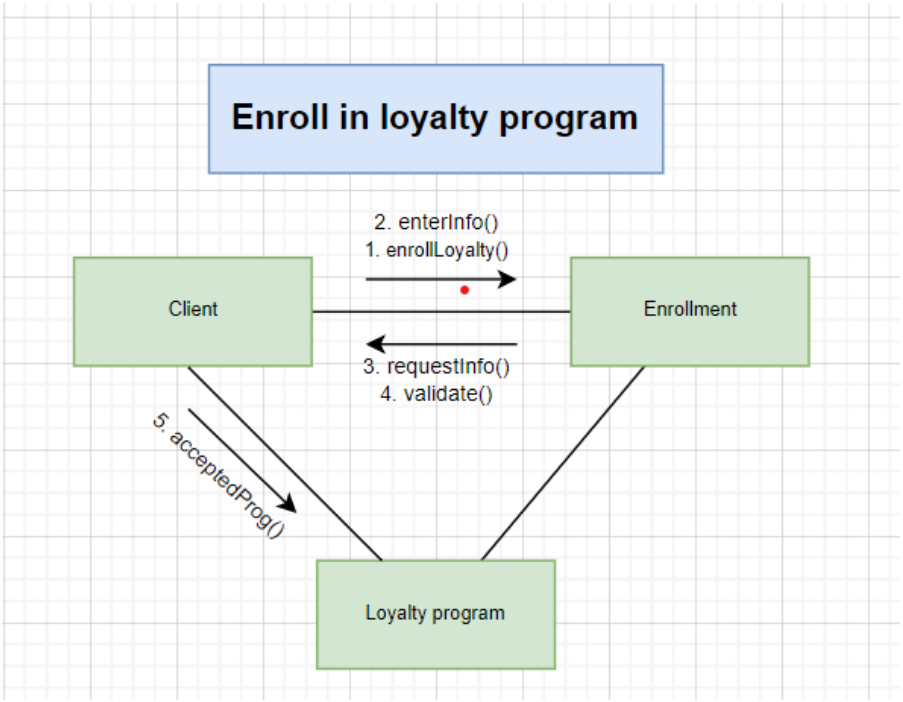


2.CLIENT

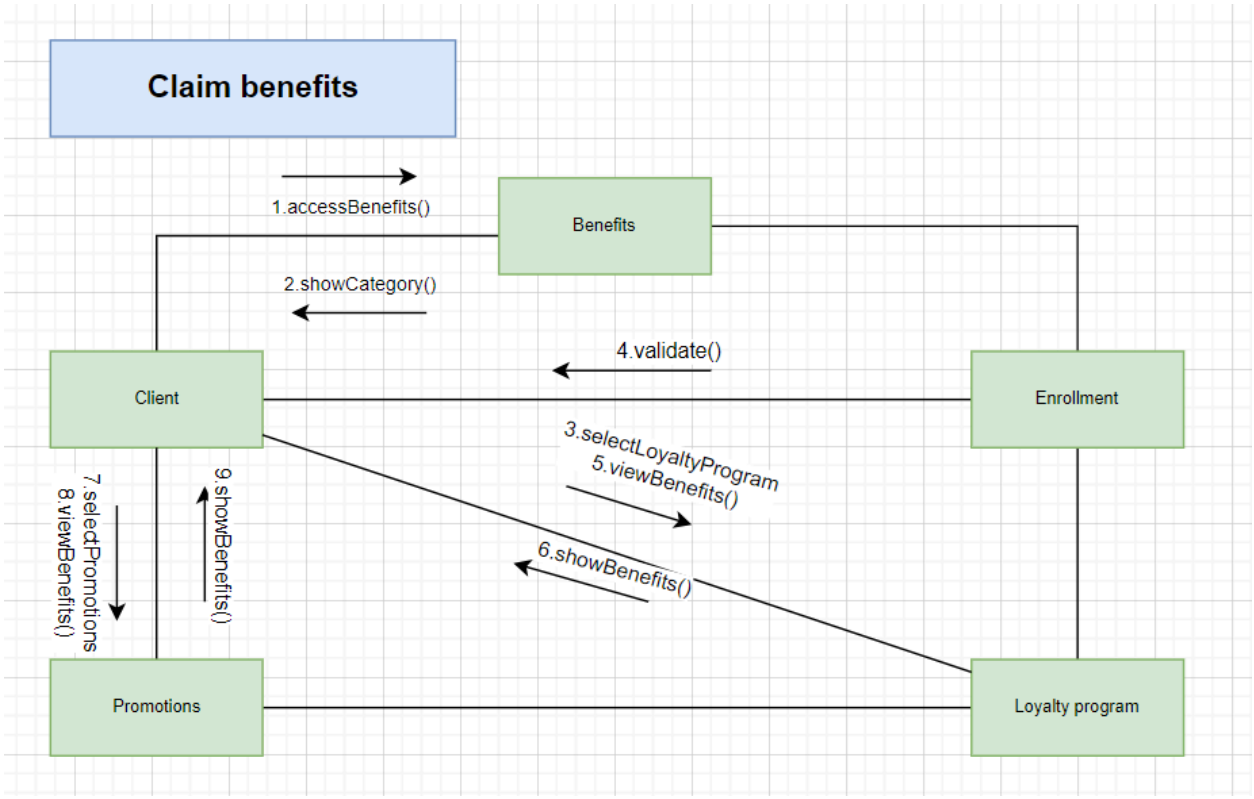
1 - Track upcoming movies



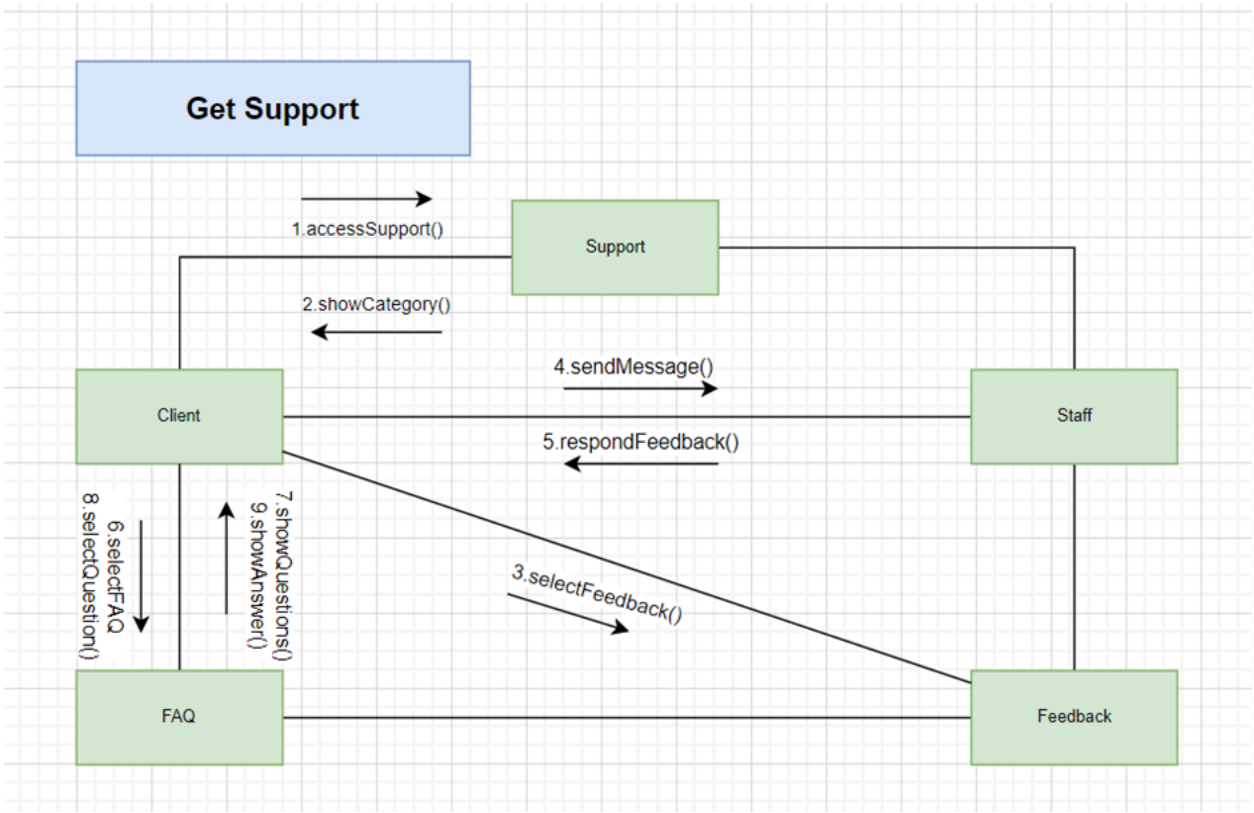
2 - Enroll in Loyalty Program



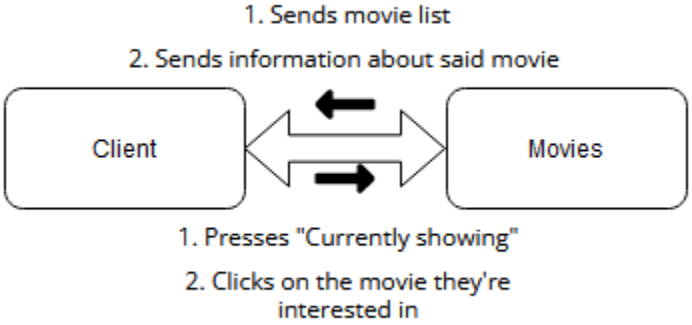
2 - Claim Benefits



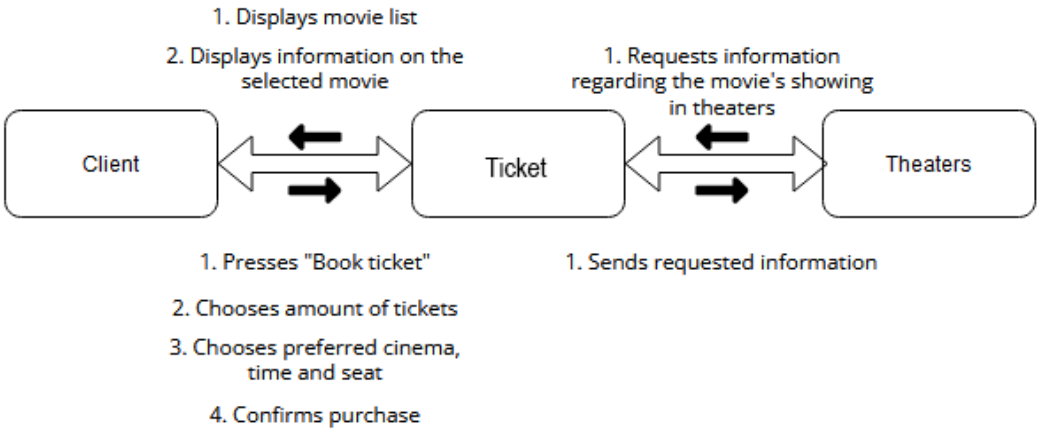
2 - Get Support



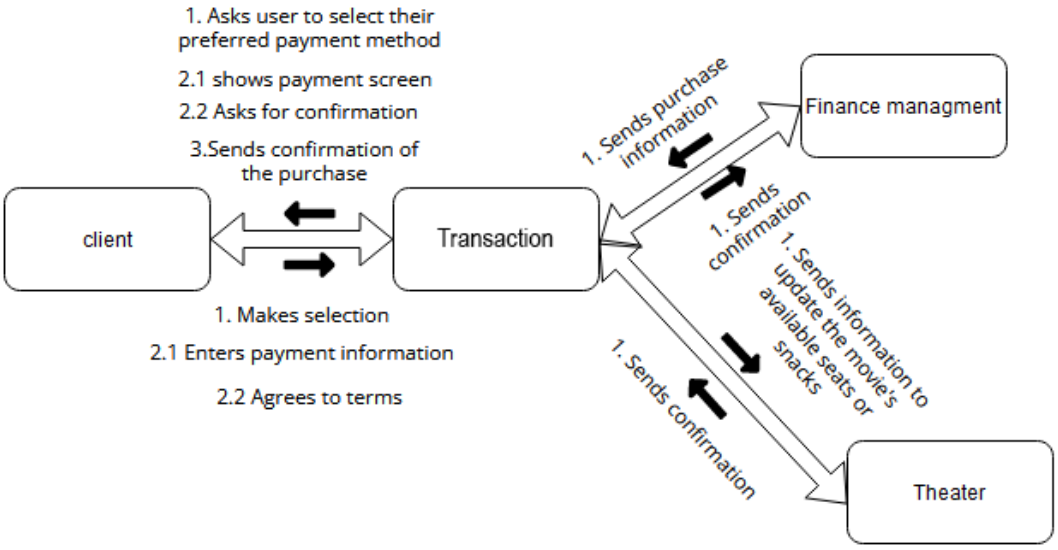
5-See current and upcoming movies



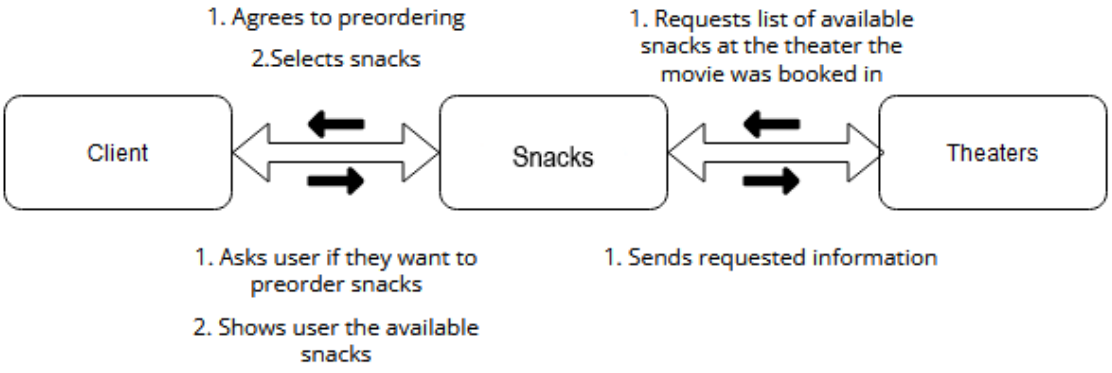
6-Book tickets in app



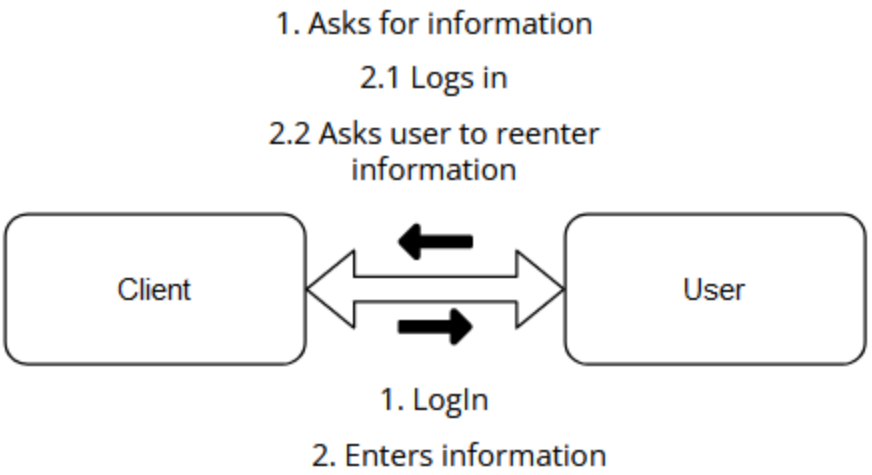
7-pay in app



8-Preorder snacks

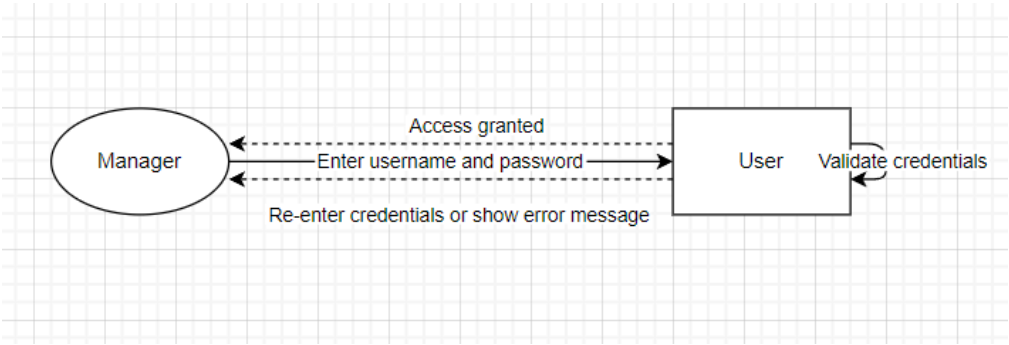


9-Log in

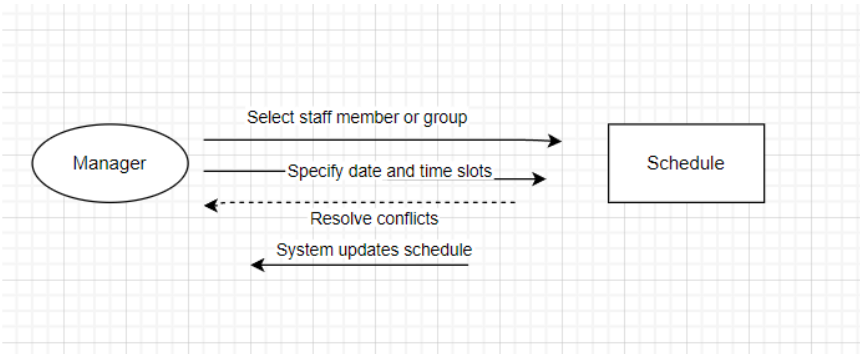


3.MANAGER

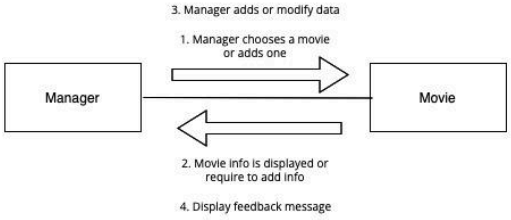
User login:



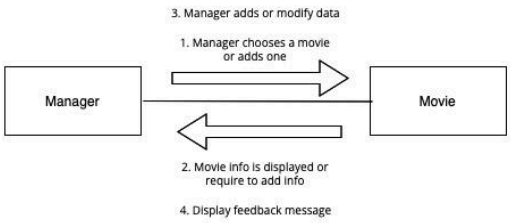
Create/Edit Schedule:



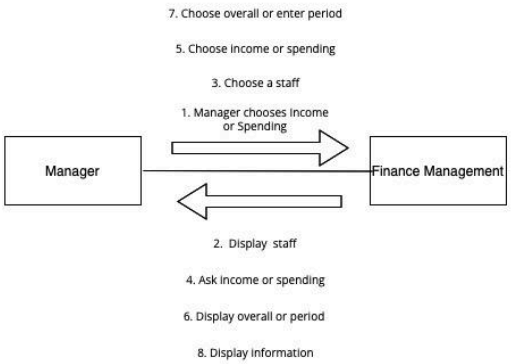
Modify movies:



Overall revenue:

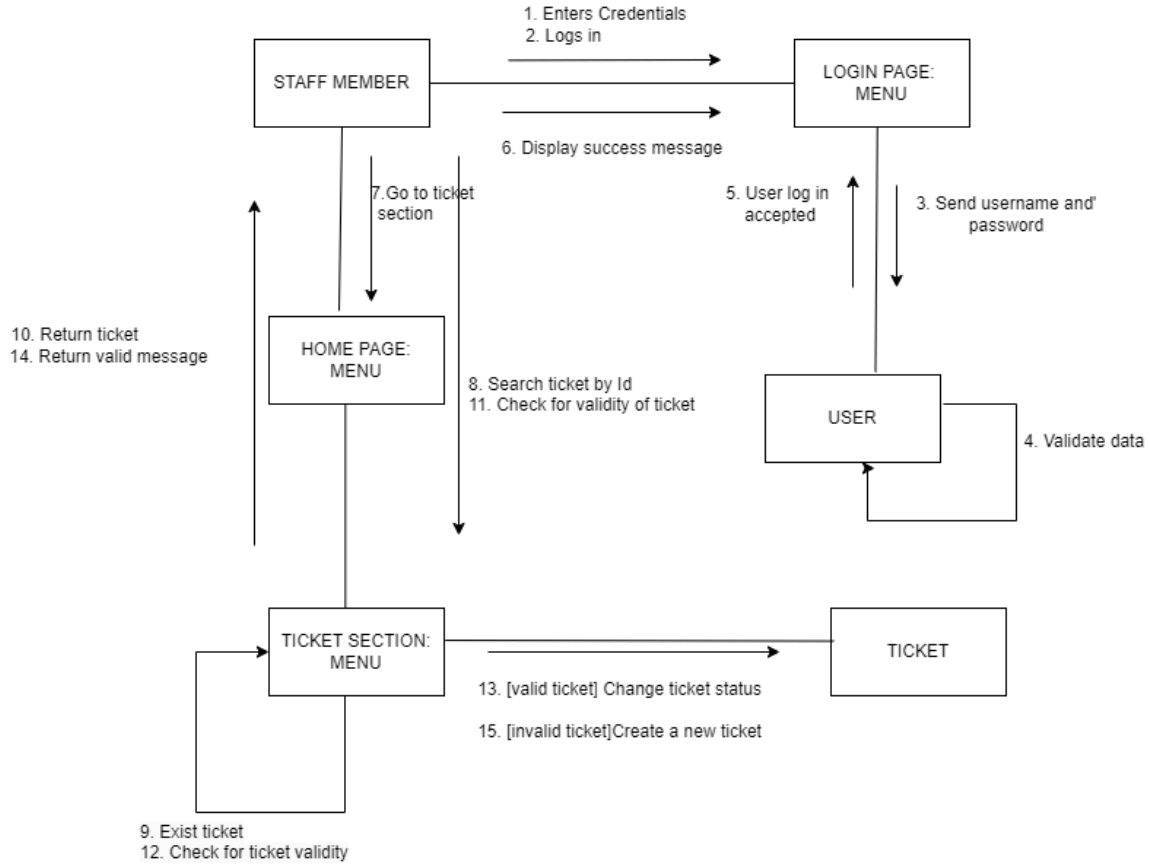


Staff spending/income:

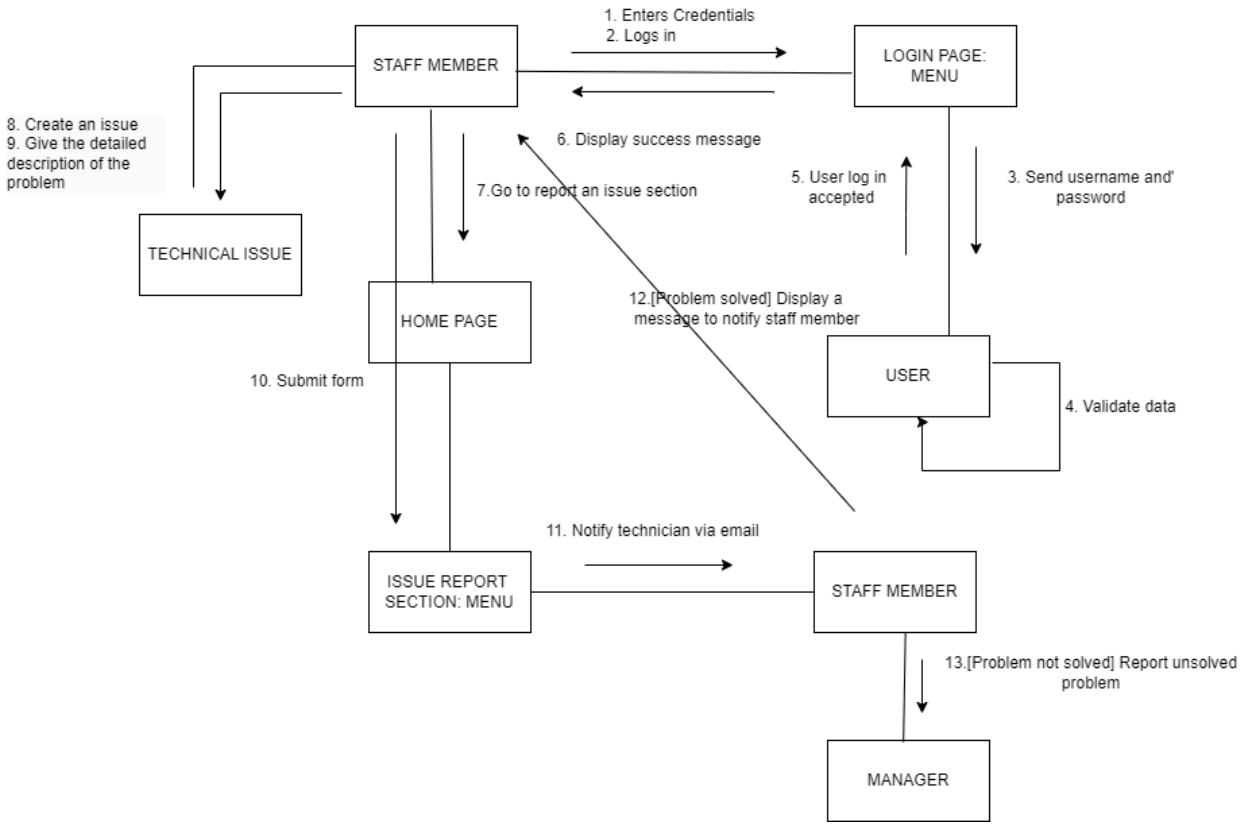


4.STAFF

1- Validate Ticket

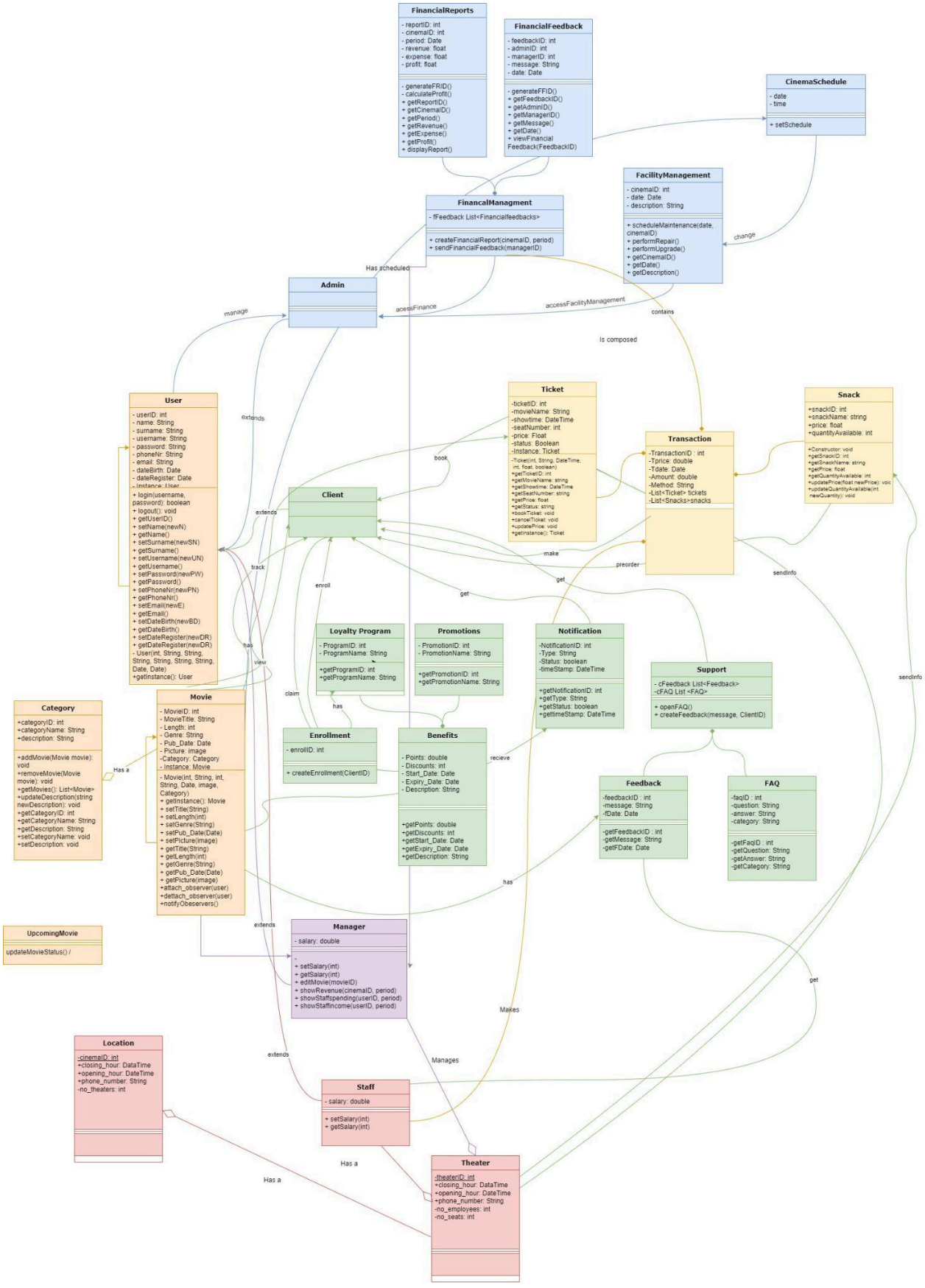


2-Technical Issue



4.9 Class Diagrams

https://app.diagrams.net/#G1N8wLu4P4O1CcI7-uvLLZhrCWWKYfzPjl#%7B%22pageId%22%3A%226133507b-19e7-1e82-6fc7-422aa6c4b21f%22%7D


FinancialReports
- reportID: int
- cinemaID: int
- period: Date
- revenue: float
- expense: float
- profit: float
- generateFRID()
- caliculateProfit()
+ getReportID()
+ getCinemaID()
+ getPeriod()
+ getRevenue()
+ getExpense()
+ getProfit()
+ displayReport()
FinancialFeedback
- feedbackID: int
- adminID: int
- managerID: int
- message: String
- date: Date
- generateFFID()
+ getFeedbackID()
+ getAdminID()
+ getManagerID()
+ getMessage()
+ getDate()
+ viewFinancial Feedback(FeedbackID)
CinemaSchedule
- date
- time
+ setSchedule
FacilityManagement
- cinemaID: int
- date: Date
- description: String
+ scheduleMaintenance(date, cinemaID)
+ performRepair()
+ performUpgrade()
+ getCinemaID()
+ getDate()
+ getDescription()
change
FinancalManagment
- fFeedback List<Financialfeedbacks>
+ createFinancialReport(cinemaID, period)
+ sendFinancialFeedback(managerID)
Has scheduled
Admin
manage
acessFinance
accessFacilityManagement
contains
Is composed
User
- userID: int
- name: String
- surname: String
- username: String
- password: String
- phoneNr: String
- email: String
- dateBirth: Date
- dateRegister: Date
- instance: User
+ login(username, password): boolean
+ logout(): void
+ getUserID()
+ setName(newN)
+ getName()
+ setSurname(newSN)
+ getSurname()
+ setUsername(newUN)
+ getUsername()
+ setPassword(newPW)
+ getPassword()
+ setPhoneNr(newPN)
+ getPhoneNr()
+ setEmail(newE)
+ getEmail()
+ setDateBirth(newBD)
+ getDateBirth()
+ setDateRegister(newDR)
+ getDateRegister(newDR)
- User(int, String, String, String, String, String, String, Date, Date)
+getInstance(): User
extends
Ticket
-ticketID: int
-movieName: String
-showtime: DateTime
-seatNumber: int
-price: Float
-status: Boolean
-Instance: Ticket
-Ticket(int, String, DateTime, int, float, boolean)
+getTicketID: int
+getMovieName: string
+getShowtime: DateTime
+getSeatNumber: string
+getPrice: float
+getStatus: string
+bookTicket: void
+cancelTicket: void
+updatePrice: void
+getInstance(): Ticket
book
Transaction
-TransactionID : int
-Tprice: double
-Tdate: Date
-Amount: double
-Method: String
-List<Ticket> tickets
-List<Snacks>snacks
Snack
+snackID: int
+snackName: string
+price: float
+quantityAvailable: int
+Constructor: void
+getSnackID: int
+getSnackName: string
+getPrice: float
+getQuantityAvailable: int
+updatePrice(float newPrice): void
+updateQuantityAvailable(int newQuantity): void
Client
extends
track
enroll
has
view
claim
make
preorder
get
sendInfo
Loyalty Program
- ProgramID: int
- ProgramName: String
+getProgramID: int
+getProgramName: String
Promotions
- PromotionID: int
- PromotionName: String
+getPromotionID: int
+getPromotionName: String
Notification
-NotificationID: int
-Type: String
-Status: boolean
-timeStamp: DateTime
+getNotificationID: int
+getType: String
+getStatus: boolean
+gettimeStamp: DateTime
Support
- cFeedback List<Feedback>
-cFAQ List <FAQ>
+ openFAQ()
+ createFeedback(message, ClientID)
Category
+categoryID: int
+categoryName: String
+description: String
+addMovie(Movie movie): void
+removeMovie(Movie movie): void
+getMovies(): List<Movie>
+updateDescription(string newDescription): void
+getCategoryID: int
+getCategoryName: String
+getDescription: String
+setCategoryName: void
+setDescription: void
Has a
Movie
- MovieID: int
- MovieTitle: String
- Length: int
- Genre: String
- Pub_Date: Date
- Picture: image
-Category: Category
- Instance: Movie
- Movie(int, String, int, String, Date, image, Category)
+ getInstance(): Movie
+ setTitle(String)
+ setLength(int)
+ setGenre(String)
+ setPub_Date(Date)
+ setPicture(image)
+ getTitle(String)
+ getLength(int)
+ getGenre(String)
+ getPub_Date(Date)
+ getPicture(image)
+attach_observer(user)
+dettach_observer(user)
+notifyObeservers()
Enrollment
- enrollID: int
+ createEnrollment(ClientID)
Benefits
- Points: double
- Discounts: int
- Start_Date: Date
- Expiry_Date: Date
- Description: String
+getPoints: double
+getDiscounts: int
+getStart_Date: Date
+getExpiry_Date: Date
+getDescription: String
recieve
Feedback
-feedbackID : int
-message: String
-fDate: Date
-getFeedbackID : int
-getMessage: String
-getFDate: Date
FAQ
-faqID : int
-question: String
-answer: String
-category: String
-getFaqID : int
-getQuestion: String
-getAnswer: String
-getCategory: String
has
UpcomingMovie
updateMovieStatus() /
Manager
- salary: double
-
+ setSalary(int)
+ getSalary(int)
+ editMovie(movieID)
+ showRevenue(cinemaID, period)
+ showStaffspending(userID, period)
+ showStaffincome(userID, period)
Makes
Manages
Location
-cinemaID: int
+closing_hour: DataTime
+opening_hour: DateTime
+phone_number: String
-no_theaters: int
Staff
- salary: double
+ setSalary(int)
+ getSalary(int)
Has a
Theater
-theaterID: int
+closing_hour: DataTime
+opening_hour: DateTime
+phone_number: String
-no_employees: int
-no_seats: int

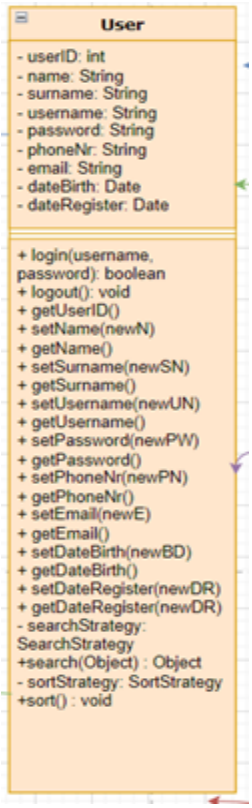
5. DESIGN PATTERNS

1.ADMIN

Strategy Pattern

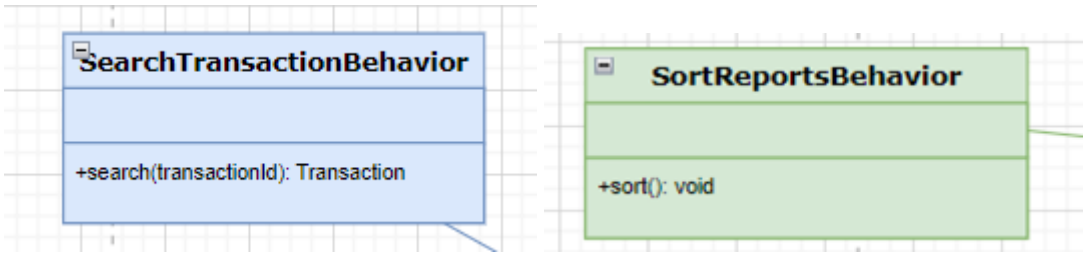
In the context of our cinema management system, the Strategy pattern is implemented through the SearchStrategy interface, which defines a common method search(Object) Object. This interface is then concretely implemented by several classes: SearchReportBehavior, SearchEmployeeBehavior, SearchMovieBehavior, and SearchTicketBehavior.

By using the Strategy pattern, we can dynamically assign search behaviors to various user roles within the system. For instance, an admin might be granted the SearchEmployeeBehavior to manage staff records, while a customer might use SearchMovieBehavior to find movies. This design allows for clear separation of concerns and promotes the open/closed principle, where new search behaviors can be added without modifying existing code. Furthermore, it enhances code maintainability and readability by encapsulating the search algorithms into distinct classes. This modular approach ensures that each search behavior can be independently developed, tested, and modified, leading to a robust and easily extendable system.
Things we have included in our class diagram:

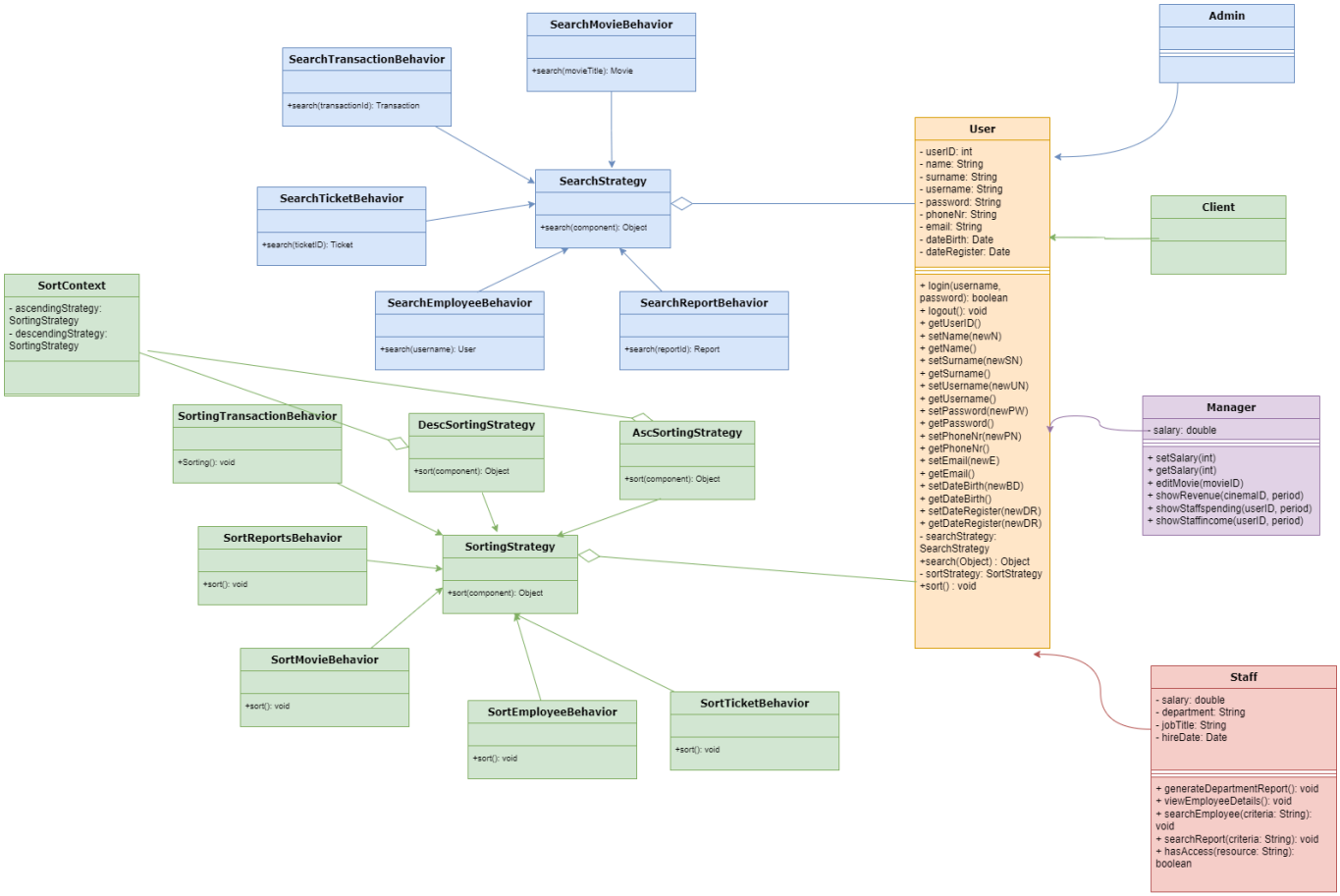

User
- userID: int
- name: String
- surname: String
- username: String
- password: String
- phoneNr: String
- email: String
- dateBirth: Date
- dateRegister: Date
+ login(username, password): boolean
+ logout(): void
+ getUserID()
+ setName(newN)
+ getName()
+ setSurname(newSN)
+ getSurname()
+ setUsername(newUN)
+ getUsername()
+ setPassword(newPW)
+ getPassword()
+ setPhoneNr(newPN)
+ getPhoneNr()
+ setEmail(newE)
+ getEmail()
+ setDateBirth(newBD)
+ getDateBirth()
+ setDateRegister(newDR)
+ getDateRegister(newDR)
- searchStrategy: SearchStrategy
+search(Object) : Object
- sortStrategy: SortStrategy
+sort() : void

Added searchStraregy and sortStrategy into user class, meaning that a user has a searchingStrategy and sortingStrategy, which means that all the subclasses of user, in our case the user roles will have different searching and sorting methods they can perform.

SearchStrategy interface that will have a generic search method depending on the type of search, more specific behaviour will be implemented in the concrete classes that will implement a SearchStretegy.



These are concrete classes that will implement the SearchStrategy or SortStrategy and will be very specific, for one functionality only. This is the only way we can differentiate the actions a certain user can perform, for example only admin and manager will be able to sort the reports, and no other user levels.


SearchMovieBehavior
+search(movieTitle): Movie
Admin
SearchTransactionBehavior
+search(transactionId): Transaction
User
- userID: int
- name: String
- surname: String
- username: String
- password: String
- phoneNr: String
- email: String
- dateBirth: Date
- dateRegister: Date
+ login(username, password): boolean
+ logout(): void
+ getUserID()
+ setName(newN)
+ getName()
+ setSurname(newSN)
+ getSurname()
+ setUsername(newUN)
+ getUsername()
+ setPassword(newPW)
+ getPassword()
+ setPhoneNr(newPN)
+ getPhoneNr()
+ setEmail(newE)
+ getEmail()
+ setDateBirth(newBD)
+ getDateBirth()
+ setDateRegister(newDR)
+ getDateRegister(newDR)
- searchStrategy: SearchStrategy
+search(Object) : Object
- sortStrategy: SortStrategy
+sort() : void
SearchStrategy
+search(component): Object
SearchTicketBehavior
+search(ticketID): Ticket
Client
SortContext
- ascendingStrategy: SortingStrategy
- descendingStrategy: SortingStrategy
SearchEmployeeBehavior
+search(username): User
SearchReportBehavior
+search(reportId): Report
SortingTransactionBehavior
+Sorting(): void
DescSortingStrategy
+sort(component): Object
AscSortingStrategy
+sort(component): Object
Manager
- salary: double
+ setSalary(int)
+ getSalary(int)
+ editMovie(movieID)
+ showRevenue(cinemaID, period)
+ showStaffspending(userID, period)
+ showStaffincome(userID, period)
SortReportsBehavior
+sort(): void
SortingStrategy
+sort(component): Object
SortMovieBehavior
+sort(): void
SortEmployeeBehavior
+sort(): void
SortTicketBehavior
+sort(): void
Staff
- salary: double
- department: String
- jobTitle: String
- hireDate: Date
+ generateDepartmentReport(): void
+ viewEmployeeDetails(): void
+ searchEmployee(criteria: String): void
+ searchReport(criteria: String): void
+ hasAccess(resource: String): boolean

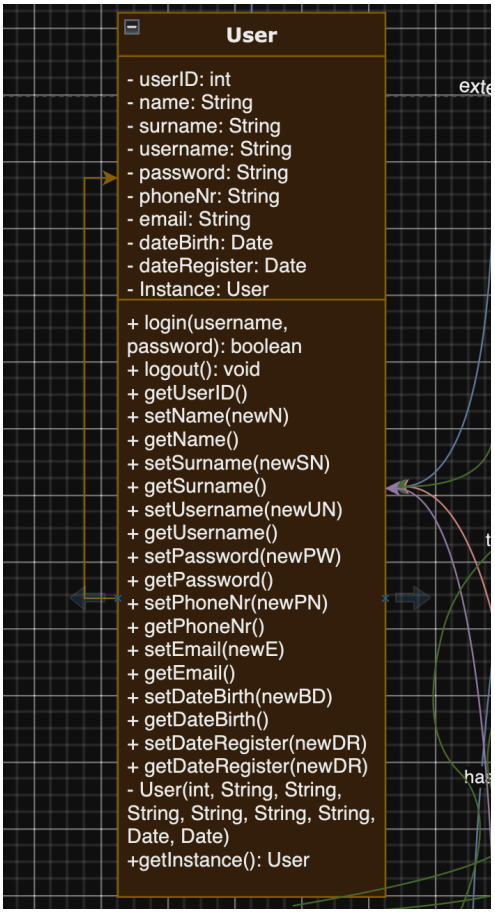
2.MANAGER

Singleton

The singleton is a fundamental design pattern, which belongs to the creational design patterns. It is used to prevent the instantiation of more than one object from a particular class.

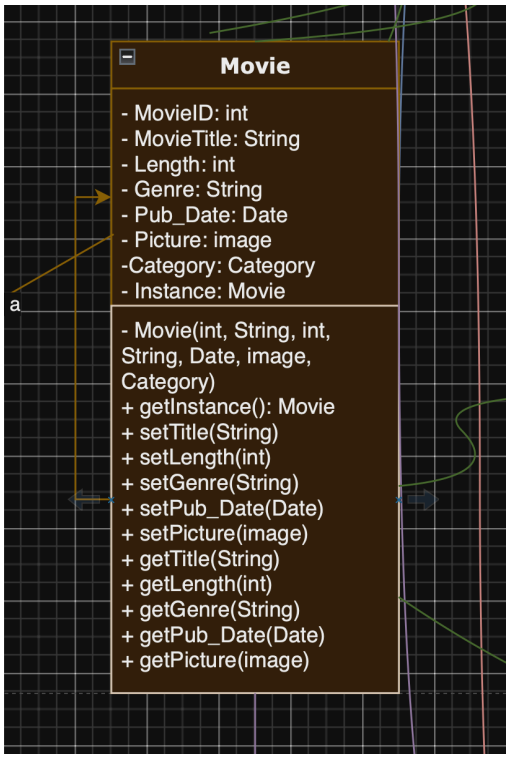
There are several benefits of using this particular design pattern, from which we can mention the added efficiency of sparing the memory from overloading with several instances of the same object. In our case, only one instance of the following classes needs to be created and used.

To achieve this, the following classes contain an object of themself, and the constructor which initiates the objects is declared as private. Instead of directly calling the constructor from outside of the class, which would lead to the creation of various objects from it, the constructor is only called once inside the getInstance method, only if it has not previously been done.



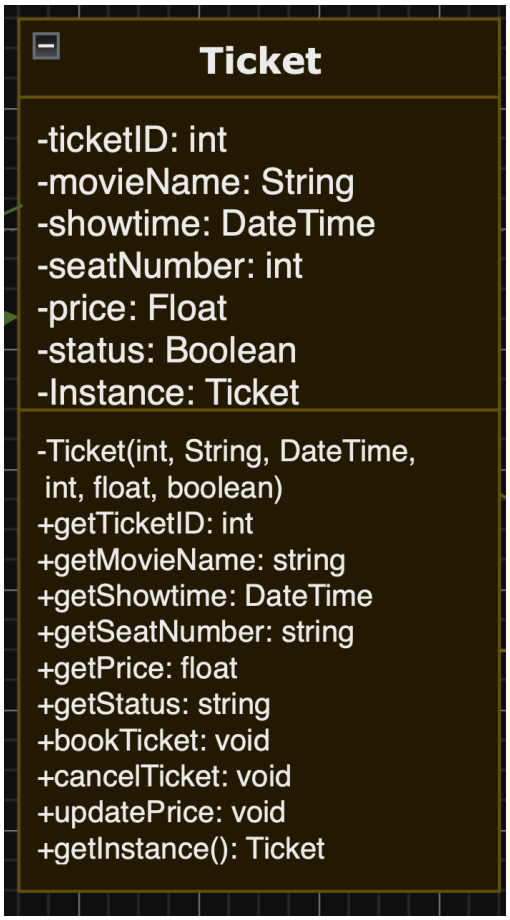
User Class (This extends to all of its children classes: Admin, Manager, Client, Staff)

As you see the class has a private constructor and a public getInstance function that checks if the instance of the class is created or not. If yes, return it, else create a new one.



Movie Class

As you see the class has a private constructor and a public getInstance function that checks if the instance of the class is created or not. If yes, return it, else create a new one.



Ticket Class

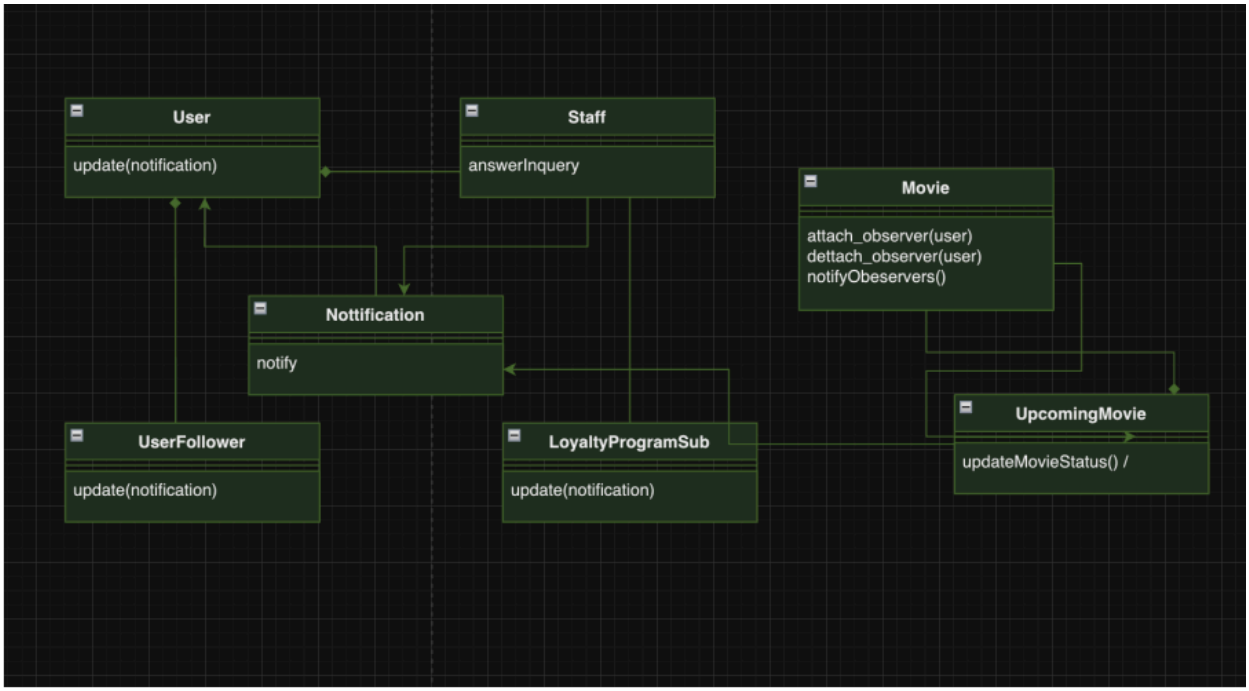
As you see the class has a private constructor and a public getInstance function that checks if the instance of the class is created or not. If yes, return it, else create a new one.

1.CLIENT

Observal

The Observer design pattern is a behavioral design pattern used to manage the communication between objects in a one-to-many relationship. This pattern ensures that
when one object changes state, all its dependents are notified and updated automatically.
Movie Class
- Methods: attach_observer(User), detach_observer(User), notifyObservers()
- This class acts as the subject in the Observer pattern. It maintains a list of observers (users) who are interested in updates. The attach_observer method adds a user to the list of
observers, and detach_observer removes a user. The notifyObservers method iterates through the list of observers and calls their update method to notify them of changes.
UpcomingMovie Class (inherits from Movie):
- Methods: updateMovieStatus()
- This subclass represents a specific type of movie that is yet to be released. The updateMovieStatus method changes the movie's status from upcoming to currently in theaters. When this status is updated, the notifyObservers method is triggered to inform all
registered users about the change.
User Interface
- Methods: update(notification)
- The User interface defines the update method that all concrete user classes must implement. This method is used to receive notifications from the Movie class.
UserFollower Class (implements User):
- Methods: update(notification)
- This class represents users who follow upcoming movies. The update method in this class will handle notifications about movie status changes, specifically alerting users when a
movie they've followed is released.
LoyaltyProgramSub Class (implements User):
- Methods: update(notification)
- This class represents users who are part of a loyalty program. The update method in this
class will handle notifications related to loyalty program details, ensuring that users receive
information about new benefits or offers.

Staff Class
- Methods: answerInquery()
- This class handles user inquiries. When a staff member responds to an inquiry, it sends
out a notification to the relevant user. The actual notification process would utilize the update
method of the User interface.
Notification Class
- This class contains the information to be sent out as updates. It holds details like movie
information, loyalty program updates, and responses to inquiries.
When a movie's status is updated from upcoming to currently in theaters, the updateMovieStatus method in the UpcomingMovie class is called. This triggers the notifyObservers method in the Movie class, which then calls the update method for each registered user. Users who are instances of UserFollower receive notifications about the movie release, while those in LoyaltyProgramSub might receive updates about related offers
or programs.
For user inquiries, the Staff class's answerInquery method would create a notification and
use the update method to inform the user who made the inquiry.
This design ensures a decoupled and efficient way to manage and distribute updates, adhering to the principles of the Observer pattern.


User
update(notification)
Staff
answerInquery
Movie
attach_observer(user)
dettach_observer(user)
notifyObeservers()
Nottification
notify
UpcomingMovie
updateMovieStatus() /
UserFollower
update(notification)
LoyaltyProgramSub
update(notification)