University of Bahrain
College of Information Technology
Department of Computer Science

ITSE302 - Software Design & Architecture

CINEMA MANAGEMENT SYSTEM (CMS)

PP4 - Project Phase 4 - Final Project

Submitted to: Dr. Fawzi Albalooshi

- Zahraa Fadhel 202209444
- Israa Alwedaei 202204345
- Malak Zakareya 202208286
- Manar Ali 202208628

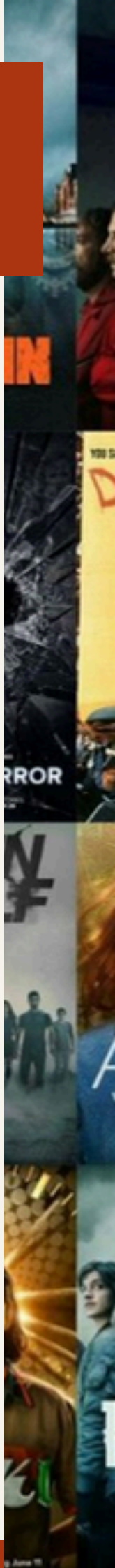
TABLE OF CONTENTS

Requirements Description

Technology, which touches every aspect of our lives, is a key driver of progress in our current technological age. From the necessities of life, like communication, to its enjoyment. The role that technology plays in our lives is growing every day. Cinemas serve as valuable entertainment venues, but their effectiveness can be hindered by outdated and inefficient management practices. A Cinema Management System (CMS) solves this issue. It is software that aims to manage and track the daily operations of the cinema so that all stakeholders (customers, staff, and admins) can easily access the functions they need, ensuring smooth workflow.

The cinema management system serves two primary purposes. First, it provides customers with a simple and comfortable booking experience by offering a straightforward way to book tickets and food in advance, allowing them to avoid long queues and ensure that their preferred seats are selected. Second, it works as a centralised platform for managing movie availability, seat inventory, and all movie-related details. It facilitates the reservation process by enabling the cinema staff to manage movie inventory and handle bookings, modifications, and cancellations seamlessly. The Cinema Management System (CMS) also makes communication with customers easier by sending e-tickets, payment confirmations, and movie updates or delays via email.

The system is accessible to users in two ways: either as guests, who enter the system without registration and hence are limited to viewing available movies and their showtimes, filtering/searching through them but they cannot make reservations, or as registered users, who enjoy all guest features along with additional capabilities such as booking tickets, selecting showtimes, choosing seats, pre-ordering food, making online payments, receiving confirmations/e-tickets/updates about movie details via email, viewing current bookings, cancelling them, and viewing or updating profile details.



To become a registered user, a form with the required fields must be filled out. The email or username along with password are required to log in each time. After a successful registration, the user can update his information except the birth date; it can only be changed once after registration.

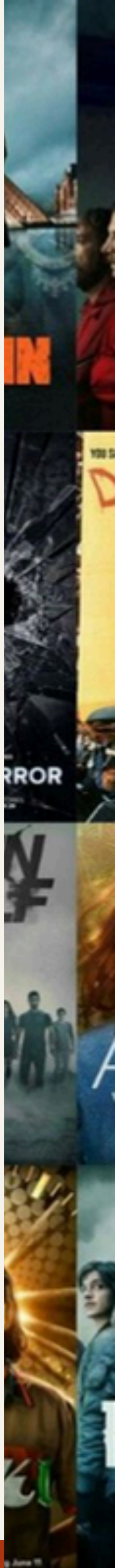
During the reservation process, registered users can access the system to choose the preferred movie and book multiple tickets for the same movie or different movies. While making a reservation, they can select the type of seat—VIP or standard—as well as the food option for each ticket separately, if desired. There are multiple types of payment methods. Following a successful payment, customers will receive an email confirmation along with their e-tickets.

If a customer wishes to cancel an existing booking, the cancellation and its' following procedure will be done depending on the time of cancellation and the starting time of the show.

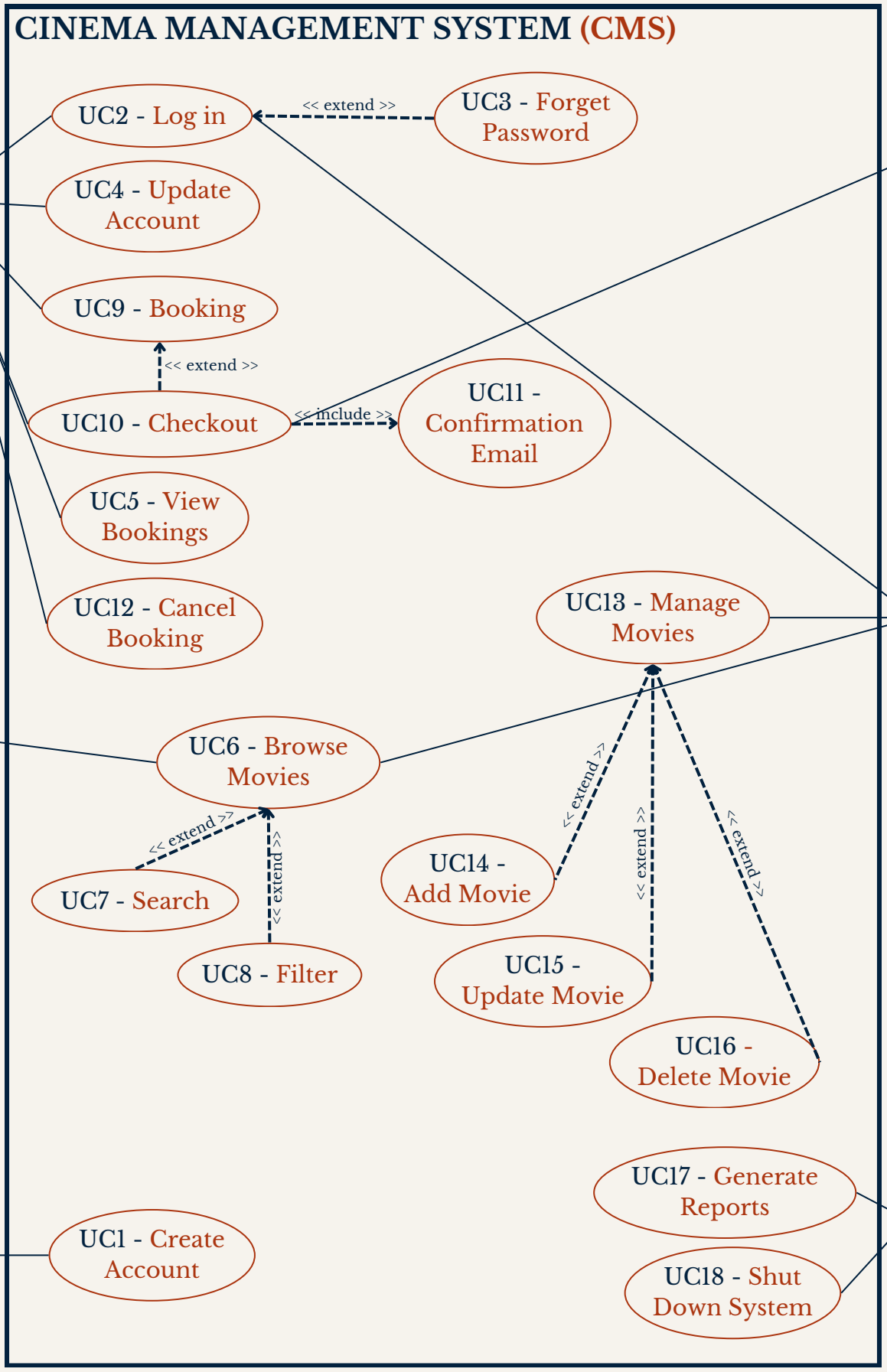
Apart from customers, the staff can perform a variety of tasks to ensure that operations run smoothly and help customers meet their needs without any problems. These tasks include managing movies by updating movie inventory, scheduling their showtimes and choosing a suitable hall for each.

A list of required fields must be filled out on the new movie-adding form. The staff has the authority to extend the movie's screening period. However, based on the movie's performance, the system will automatically extend the period of the movie or not. If not, the movie will be deleted from the movie list, and if any changes regarding the time or place of the movie were made, an alert email will be sent immediately to customers who have reserved this movie.

Aside from staff members' duties, administrators can perform additional tasks that regular staff members are not permitted to perform. The administrator can create sales and revenue reports to keep tracking the system's operations and assess its performance. In emergencies, such as system maintenance or a global pandemic, the administrator can shut down the system.

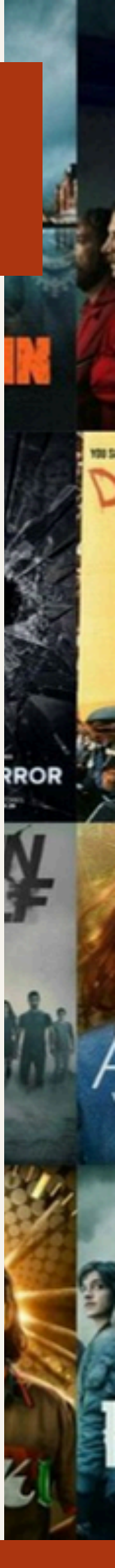


Use Case Model

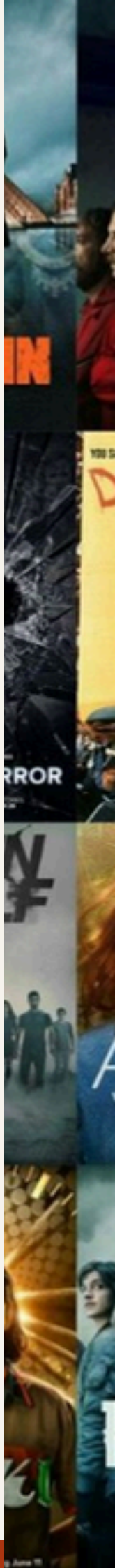


Use Case Descriptions

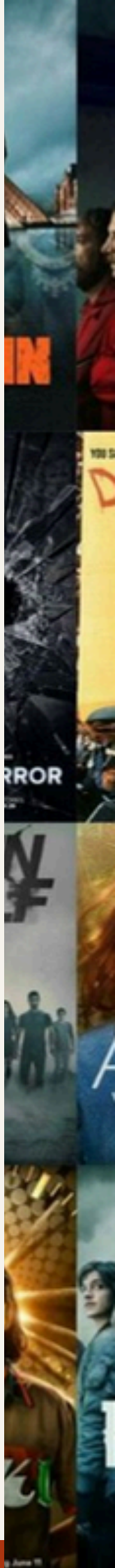
UC1 - Create Account	To become a registered customer, users must first create an account. They must complete a form with the following fields: first and last name, phone number, email address, username, password, and birth date. The username must be unique, and the user must create a password with a minimum length of eight characters. After successfully registering, the user can begin a smooth cinema ticket booking experience.
UC2 - Log in	Every time a registered user or staff member needs to use the system, they must log in with their username or email, along with their password.
UC3 - Forget Password	If a registered user or staff member forgets the password, he will be prompted to enter his phone number or email address to reset and verify the new password. After successfully resetting the password, the user can log in to the system using the new password.
UC4 - Update Account	Registered users can update their information, such as their first and last name, email address, and phone number, but the date of birth can only be updated once per user.
UC5 - View Bookings	Users can easily view their booking details by just clicking on the view bookings button that is located in the sidebar. This feature lets customers see a list of their reservations with their details, such as booking dates, showtimes, movie title, and food associated with each ticket.



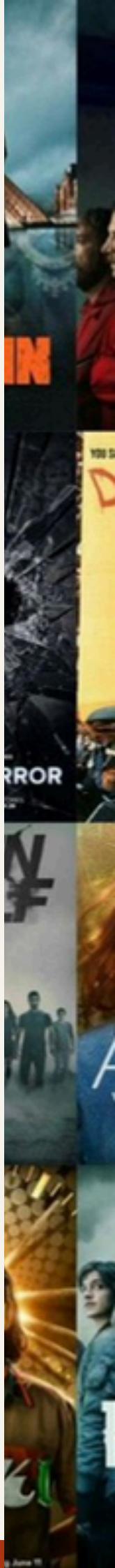
UC6 - Browse Movies	All users, whether registered or not, can easily browse the entire catalogue of available movie tickets by navigating via an interactive display of movie posters and titles. Each movie is showcased with its poster and title, allowing users to identify movies of interest quickly. Upon selecting a movie, they can view its details, such as a summary, actor's names, producing company, genre, showtimes, IMDb rating, age restriction, language, duration, and the remaining seats for each showtime (updated in real-time). Anytime during browsing, users can search or filter to narrow down their options.
UC7 - Search	All users can search for a specific movie by entering any word contained in the movie title. So, only movies with matched titles will be displayed to browse through them.
UC8 - Filter	All users can filter movies based on genre, price range, age restriction, and start and end showtimes. So, only movies that match the filter criteria will be displayed to browse through them.
UC9 - Book Movie(s)	Only the registered user can book any movie(s) after browsing or searching it. To book a movie, the user must choose the movie, showtime, hall number, and preferred seat. The user has the option to book for more than one movie at the same time. After adding the tickets, the user also has the option to pre-order the food and beverages for each ticket. The system will show the total amount to pay based on all tickets that are added to the cart and their details.
UC10 - Checkout	In the checking-out process, the user must pay by the provided methods, which are credit and debit cards, benefit pay, or by a previous card that is saved in the system. After a successful payment, the customer will receive a confirmation email, and the booking will appear in the view bookings section.



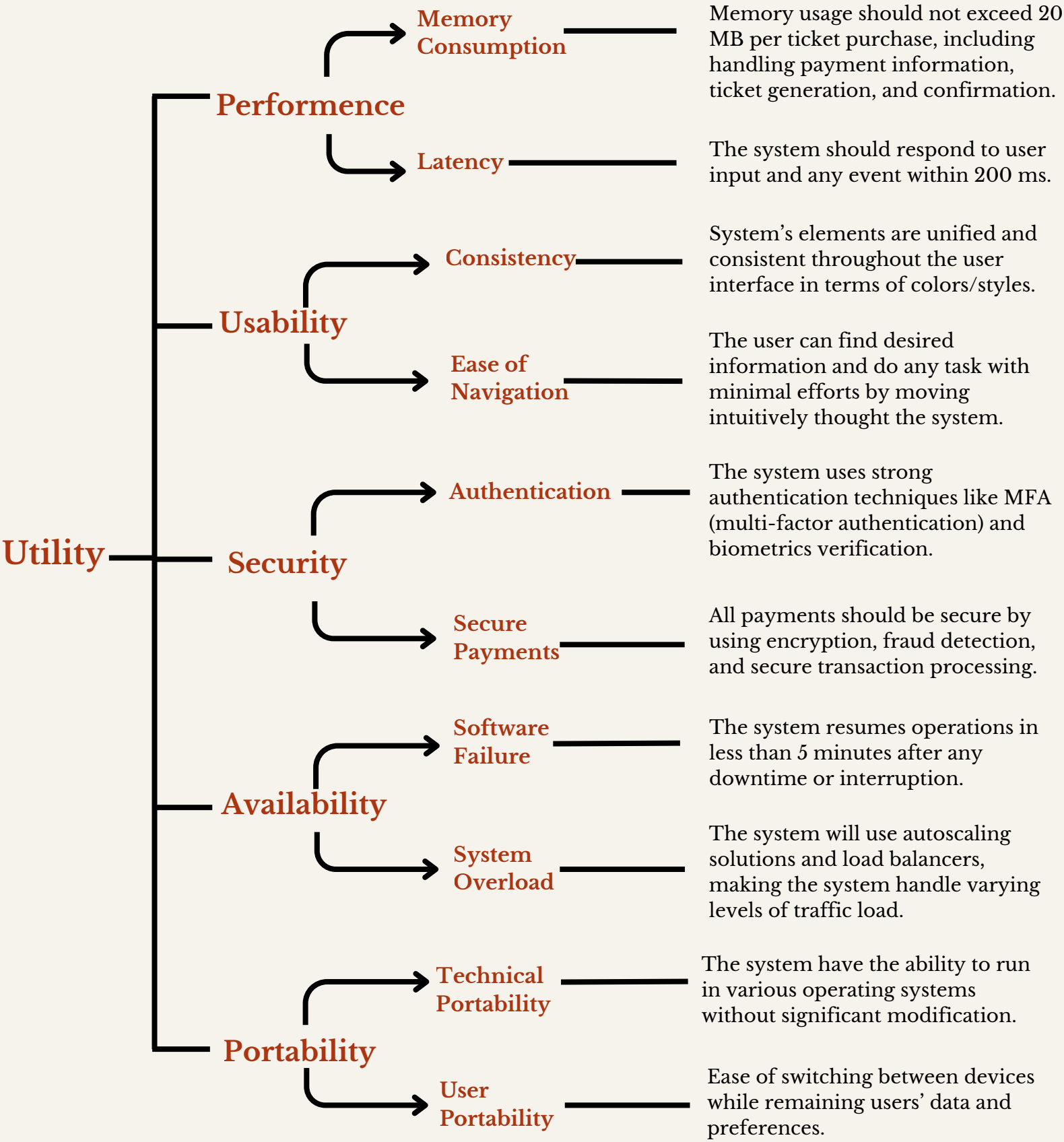
UC11 - Confirmation Email	The system shall send a confirmation email after a successful payment, providing customers with a QR ticket and detailed reservation information (receipt) that shows the reserved ticket(s), each with its specific showtime, theatre location, any food items ordered, and the total amount paid.
UC12 - Cancel Booking	The customer has the option to cancel the current bookings through the cancel booking function. Before cancellation, the system will ask the user to confirm that he wants to cancel the booking. According to the showtime and the timing of cancellation, the system will determine whether the user will receive a refund or not. If more than 24 hours are remaining before the show starts, the user will refund his money, but if it is less than 24 hours, the user will not. For refunding, the user must choose the way of refunding either as a balance in his account directly or through his original payment card within seven working days.
UC13 - Manage Movies	It provides an interface that staff members access after logging into the system. From there, they may administer the system by adding movies, then either deleting it afterward or updating its streaming period manually. Otherwise, the system will delete or update its streaming period automatically depending on how well the movie is performing. The performance of the movies is measured by the average of percentages of seats booked each show.
UC14 - Add Movie	To add a movie, staff members must log in and fill out a form that includes the title, poster, main actors, summary, age restriction, IMDb rating, language, duration, and the movie show times, including information about the available halls (VIP, 3D, IMAX, or standard). After choosing the hall type, the system will automatically input the seat type(s) available (VIP, or standard) with their number(s).



UC15 - Update Movie	Staff can update all details of movies and their shows manually or they can allow the system to manage these updates automatically based on movie performance, as indicated by a seat booking percentage of 60%. After the first two trial weeks, the process will be carried out. If the movie meets or exceeds the performance criteria, the assessment will continue weekly until the booking percentage drops below 60%, at which point the movie will be removed from the schedule. After updating any details regarding the time or place of the movie, an alert email will be sent immediately to the customers who reserved it.
UC16 - Delete Movie	Staff can delete any movie anytime or let the system handle deletions automatically based on performance. A movie will be deleted from the system if its percentage drops below 60%. In both cases, it will be taken out of the system and made unavailable for streaming.
UC17 - Generate Reports	Admins are allowed to generate sales and revenue reports, automatically created by the system. They provide insights into the overall financial performance of the platform by displaying the total number of tickets sold over a selected period, total revenue generated, and detailed breakdowns by hall, or date range. The data can be used to identify peak periods and help with future budgeting and forecasting.
UC18 - Shut Down System	The system administrator has the authority to shut down or close the system in an emergency, such as a pandemic, or for maintenance purposes. after the system displays a confirmation message for the admin, and the admin's username and password are required to ensure that the system will be shut down.



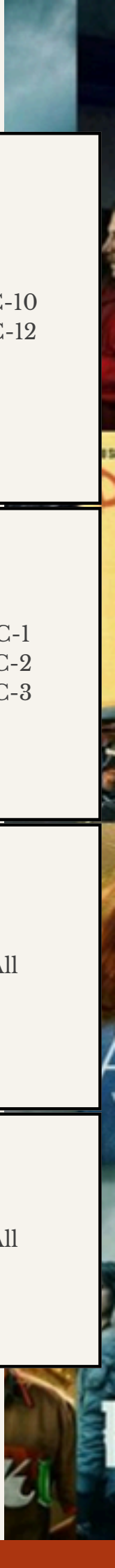
Utility Tree



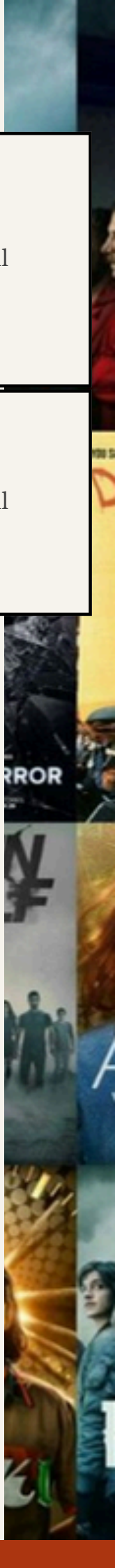
Quality Attribute List

ID	Quality Attribute		Scenario	Associated Use Case
QA-1	Performance	Latency	When a user requests a booking search, or filter, the system should proceed in less than 200ms seconds due to CON-2.	All
QA-2	Performance	Memory Consumption	The system shall complete the ticket purchase operation that will validate payment details, generate a ticket, and send the ticket to the user via email while monitoring memory usage to not exceed 20 MB due to CON-1.	UC-1, UC-7, UC-9, UC-10, UC-14
QA-3	Usability	Ease of Navigation	The system ensures easy navigation between all its functions and pages, so the user can book tickets with only 4 clicks on the screen.	UC-5, UC-6, UC-7, UC-8, UC-13
QA-4	Usability	Consistency	The system should have consistent components so that the user can access and interact with the system clearly, easily and with familiarity. The system maintains a high level of coherence through consistency in the way data is shown and in the way that fonts, color schemes, and style are applied.	All

QA-5	Security	Secure Payments	In order to comply with industry standards like as PCI DSS (Payment Card Industry Data Security Standard), the cinema management system places a high priority on security through the use of encryption algorithms, robust access controls, and frequent audits to guarantee a safe moviegoing experience for all users.	UC-10 UC-12
QA-6		Authentication	The system used strong authentication techniques like MFA (multi-factor authentication) and biometrics verifications to lower the danger of unauthorized login attempts. strengthening the system's defenses and making it much more difficult for hackers to gain entry.	UC-1 UC-2 UC-3
QA-7	Availability	Software Failure	It ensures that booking service is consistently accessible to users with minimum downtime. This system guarantees a 99.9% uptime rate, allowing customers to seamlessly reserve movie tickets anytime, enhancing user satisfaction and maximizing booking efficiency.	All
QA-8		System Overload	The system will use autoscaling solutions that would add or remove server instances based on current traffic levels and load balancers to spread the load over several servers so that no server is overloaded.	All



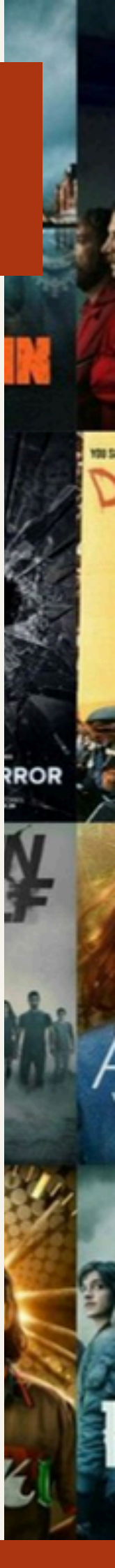
QA-9	Portability	Technical Portability	Users with different hardware platforms running different operating systems and in different web browsers can use the Cinema Management System smoothly and with equivalent capabilities and features due to CON-4.	All
QA-10		User Portability	Users can switch between their devices easily while maintaining their data, so they can view their upcoming bookings, profile details and past purchases from a different device.	All



QA Priority Table

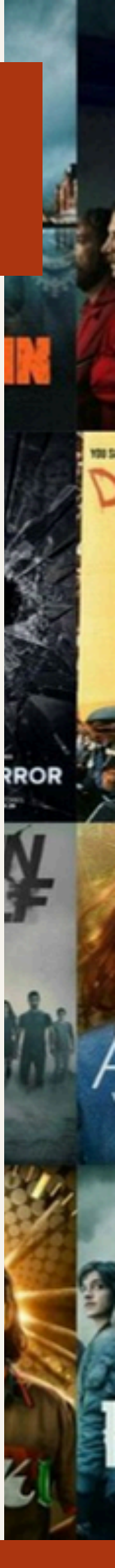
Technical Risk \ Business Importance	L	M	H
L	9	3	
M	10		4,6
H		2,8	1,5,7

*Numbers represent the Quality Attributes IDs.



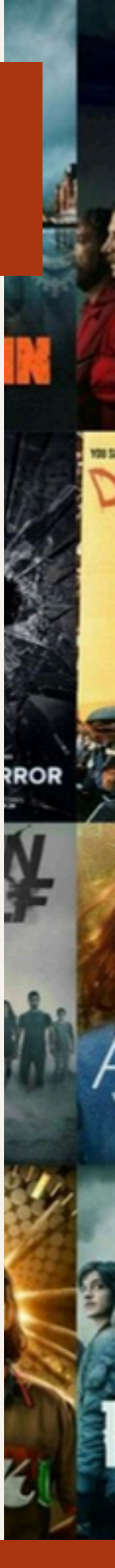
Constraints

ID	Constraint
CON-1	The system must complete all operations needed for ticket purchase while ensuring that memory usage does not exceed 20 MB for the entire transaction process.
CON-2	The system must process the ticket reservation process in under 3 seconds to avoid wasting customer time and ensure satisfaction.
CON-3	The system must support at least 1000 simultaneous users with no degradation in performance or user experience.
CON-4	The reservation functions are accessible across a variety of devices, web browsers (Microsoft Edge, Opera, Mozilla Firefox, Safari, and Google Chrome), and operating systems (Windows, macOS).
CON-5	The system needs to be completed in six months while successfully accomplishing the objectives specified in the design specifications.
CON-6	The system must implement event logging and monitoring feature so that it safely preserve events from the previous 60 days allowing administrators to keep track of user logins and see their actions.

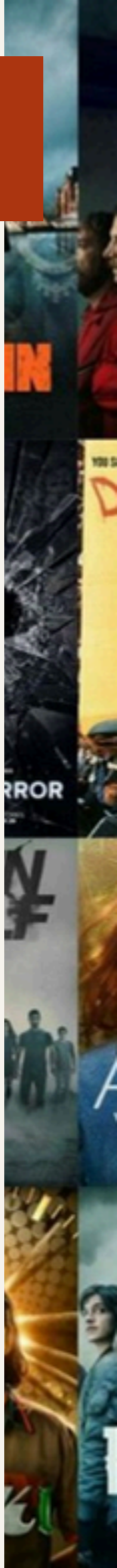


Concerns

ID	Concern
CRN-0	Establishing an overall system structure.
CRN-1	To leverage the team expertise in Java programming language, its technologies and Javascript along with its libraries.
CRN-2	Slow performance during peak ticket sales is critical concern as it can lead to revenue lost and customers dissatisfaction.
CRN-3	Ensuring that showtimes, seat availability, and pricing are accurate and up-dated in real time to avoid conflicts and hence achieve customer satisfaction.
CRN-4	Assign the design and development work to designated team members, making sure that every individual is given responsibilities that meet their set of abilities.
CRN-5	The development team should be aware of Bahraini government regulations and guidelines related to handling users data, its privacy and the policies regarding streamed material to make sure that age limitations, censorship, content licensing, and data protection rules are all followed.



First Iteration :
Establishing an Overall System Structure



Step 1 : Review Inputs

Before diving into design, we need to ensure that all design inputs — architectural drivers — are correct, complete and clearly understood. This will help us gain a comprehensive understanding of the overall design problem to be solved. Following is the summary table for them :

<table>
<tr><th>Category</th><th>Details</th></tr>
<tr><td>Design Purpose</td><td>This is a greenfield system from a mature domain. The purpose is to develop a comprehensive design to guide the system's development.</td></tr>
<tr><td>Primary Functionality Requirements</td><td>

UC-6 "Browse Movies" : Because it supports the core business by providing movie discovery & selection.
UC-9 "Book Movie(s)" : Because it directly supports the core business of reserving tickets.
UC-10 "Checkout" : Because it supports the core business of online payment to confirm booking.
UC-13 "Manage Movies" : Because it directly supports the core business by enabling administration of available movies.

</td></tr>
<tr><td>Quality Attribute Scenarios</td><td>
<table>
<tr><th>Scenario ID</th><th>Importance to the Customer</th><th>Difficulty of Implementation According to Architect</th></tr>
<tr><td>QA-1</td><td>High</td><td>high</td></tr>
<tr><td>QA-2</td><td>Medium</td><td>High</td></tr>
<tr><td>QA-4</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-5</td><td>High</td><td>High</td></tr>
<tr><td>QA-6</td><td>High</td><td>Medium</td></tr>
<tr><td>QA-7</td><td>High</td><td>High</td></tr>
<tr><td>QA-8</td><td>Medium</td><td>High</td></tr>
</table>
From the QA list, Only QA-1 / QA-5 / QA-7 are selected as drivers because of their importance.
</td></tr>
</table>

Constraints	All of the concerns are selected as drivers.
Concerns	All of the concerns are selected as drivers.

Step 2 : Establish Iteration Goal By Selecting Drivers.

Since it is the first iteration of greenfield system, the iteration goal will be to achieve architectural concern CRN-0 of establishing an overall system structure. This requires outlining the high-level structure of the system, its components, and how they will interact with each others. Although this iteration is driven by a general architectural concern, the architect must keep in mind all of the drivers that may influence the general structure of the system. In particular, the architect must be mindful of the following: UC-6 / UC-9 / UC-10 / UC-13 / QA-1 / QA-5 / QA-7 / CON-4 / CRN-0 / CRN-4

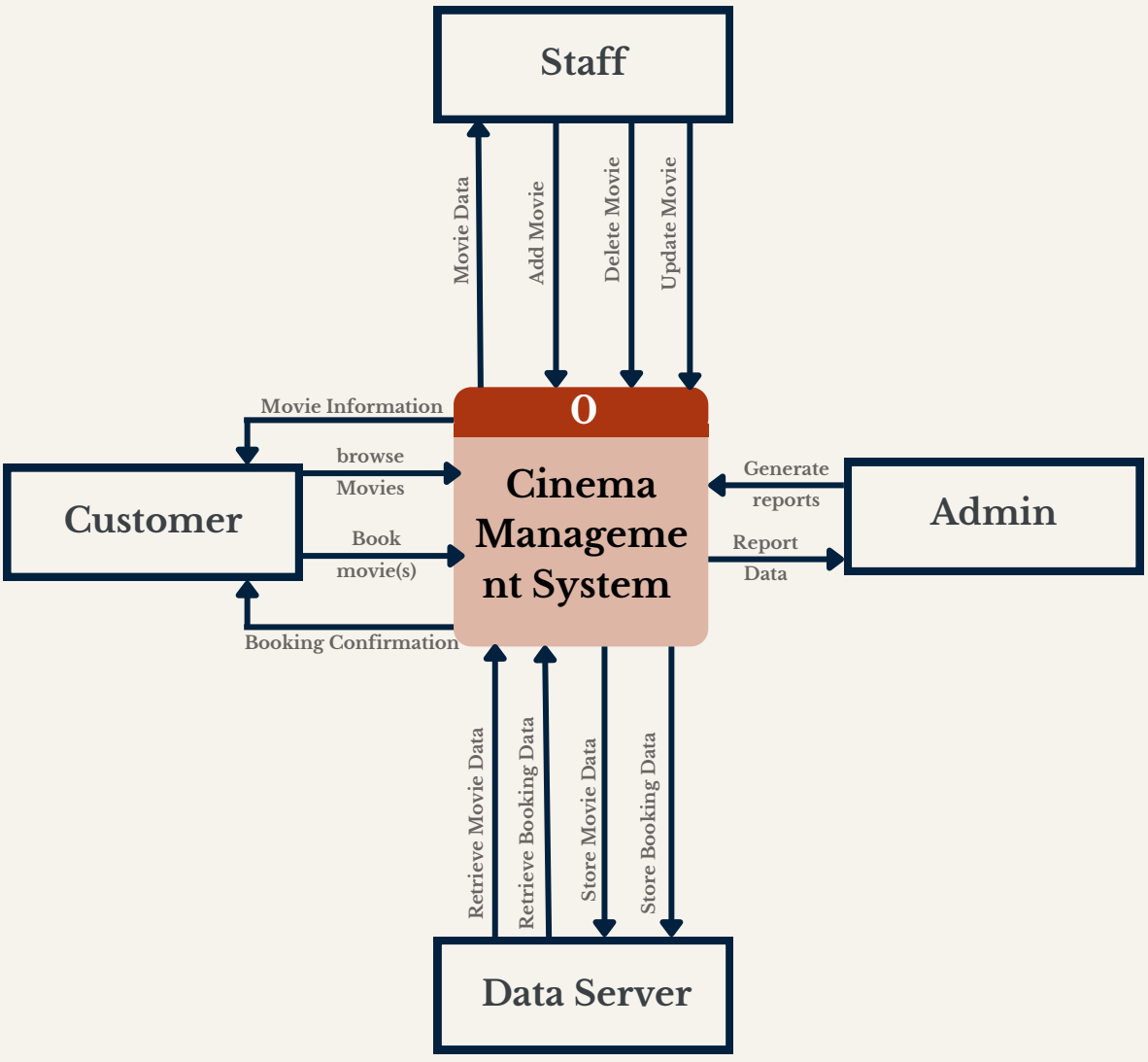


Figure 1.1 : Context Diagram (DFD) for CMS

Step 3: Choose One or More Elements of the System to Refine.

This is a greenfield development effort, so in this case to refine is the entire CMS, and refinement is performed through decomposition.

Step 4: Choose One Or More Design Concepts That Satisfy The Selected Drivers.

In this initial iteration, given the goal of structuring the entire system, design concepts are selected according to figure 1.2, which suggests Reference Architectures, Deployment Patterns and optionally Externally Developed Components for the first iteration. The following table summarizes the selection of design decisions :

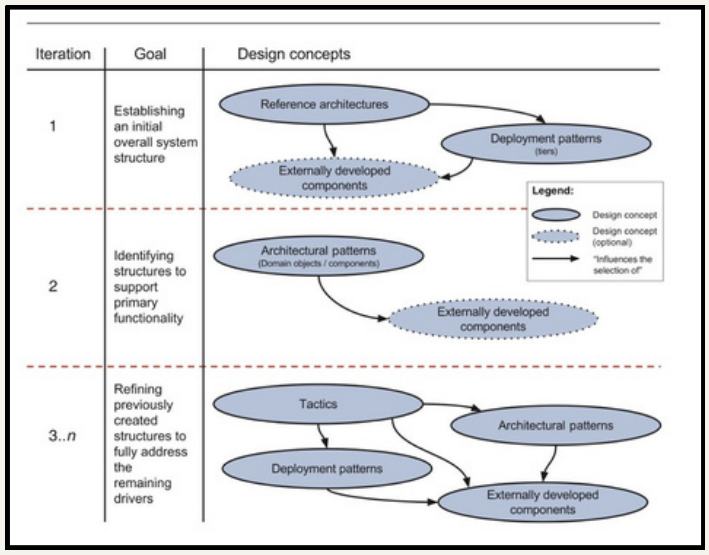


Figure 1.2 : Design Concept Selection Roadmap for Greenfields Systems

Design Decision and Locations	Rationale
logically structure the client of the system by using **Web Application Reference Architecture**	The Web Application Reference Architecture is a standardized framework that outlines the key components and their interactions within a web application (typically initiated from a web browser that communicates with a server using HTTP protocol). It serves as a guide for developers and architects to design, build, and deploy web applications effectively using three primary layers: presentation, business and data. In addition to cross-cutting layer. It is useful to achieve CON-4 and QA-8. Users can easily access the application through any web browser, and browse through it to fulfill their needs.

<table>
<tr><td></td><td>Web Application Reference Architecture is a good choice for the CMS because it do not require a rich user interface, not intended to be installed in users' machines but portability of user interface is needed. Also, the application need to be accessible over the internet. The application will be developed using Java for backend and JavaScript (React Library) for frontend.
<table>
<tr><th>Alternative</th><th>Reason for Discarding</th></tr>
<tr><td>Mobile Application</td><td>Because it focuses on the development of applications that are deployed in handheld devices. These devices was not considered for accessing the CMS.</td></tr>
<tr><td>Rich Client Application</td><td>Because it supports development of applications that are installed in the users' PC and do not run in web browsers (does not meet CON-4). It also supports rich user interface that is not needed for CMS.</td></tr>
<tr><td>Rich Internet Application</td><td>Due to its complexity and the limited availability of required plugins across different platforms which hinders consistent user experience. Also, loading time is slow and access to local resources is limited.</td></tr>
<tr><td>Python language</td><td>Although Python is an excellent language for structure data interchange between system, but JS has been chosen because it excels at creating dynamic and interactive web applications. Moreover, python is an interpreted language making it slower (does not meet Q-1).</td></tr>
</table></td></tr>
<tr><td>logically structure the server part of the system by using the Service Application Reference Architecture</td><td>Service applications are non-interactive applications that expose functionality through public interfaces. Therefore, They are responsible for handling all client requests and operations within the system. No other alternatives were considered.</td></tr>
<tr><td>Physically structure the application using the Three-tier Deployment Pattern</td><td>Due to CON-4, CMS must be accessed through a web browser, hence, database server must be used so three-tier deployment is appropriate (figure 1.3). It consists of: Presentation layer (Client Tier) responsible for user interface and user experience which it will interact directly with the user. Application layer(Business logic Tier) which will process the requests from the client tier. Data layer (Database Tier) responsible for data storage and managing the data. This separation simplifies development and maintenance.</td></tr>
</table>

Build the user interface of the client application using **JS (React Library)**	React is a modern javascript library that helps building interactive user interface. It uses components to make developing maintainable, scalable and it makes responsiveness with different devices (CON-4) much easier to implement. Also, it is what developers familiar with according to CRN-1.
Deploy the application using the **Java Web Start Technology**	Instead of requiring a manual download and installation, Java Web Start handles the entire process automatically. This technology allows users to launch all featured java applications in their latest versions directly from the web.

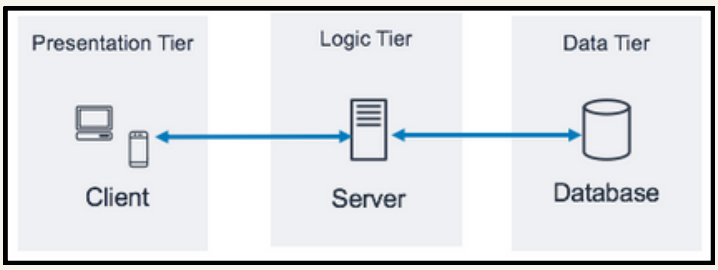


Figure 1.3 : Three-tier Deployment Pattern

Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces.

Design Decision and Location	Rationale
Remove Cross-Cutting Concerns: in the Web Application Reference Architecture	While security remains a critical aspect for any web application, other cross-cutting concerns like operational management, communication and logging may not be as relevant for a Cinema Management System (CMS). These components could potentially be streamlined or reduced to focus on the core functionalities specific to managing a cinema.
Create a module dedicated for integrating with external systems	It must establish transparent interfaces for smooth integration while guaranteeing data consistency when interacting with systems such as payment gateways or movie databases.

Step 6: Sketch Views and Record Design Decisions.

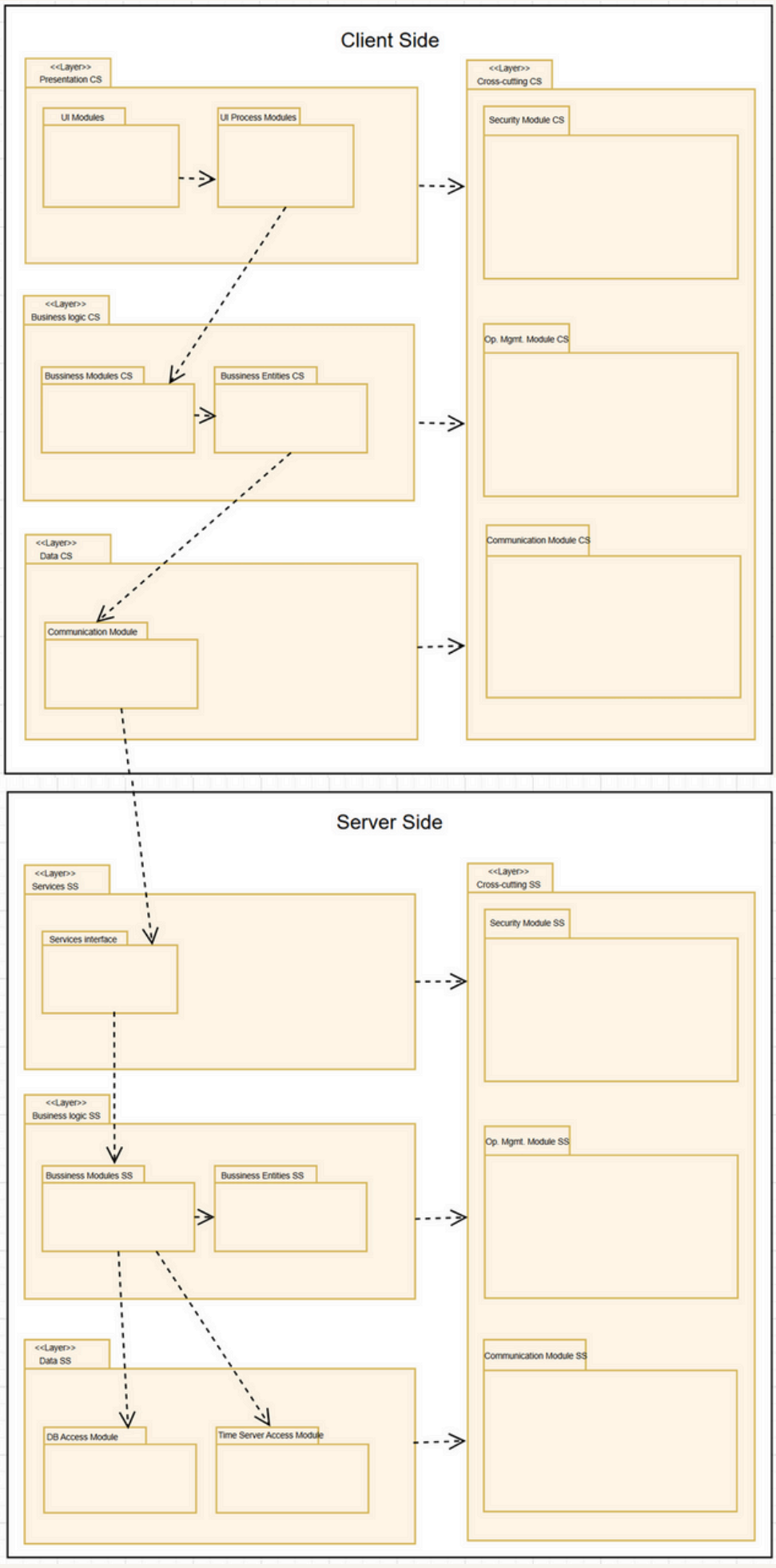


Figure 1.4 : Modules obtained from selected reference architectures

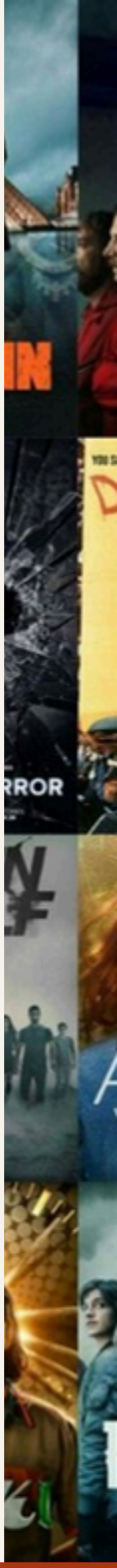
The following tables display the preceding sketch elements, outlining each element's fundamental responsibilities without diving into detail.

- **Cross-cutting in both the Client Side and Server Side:**

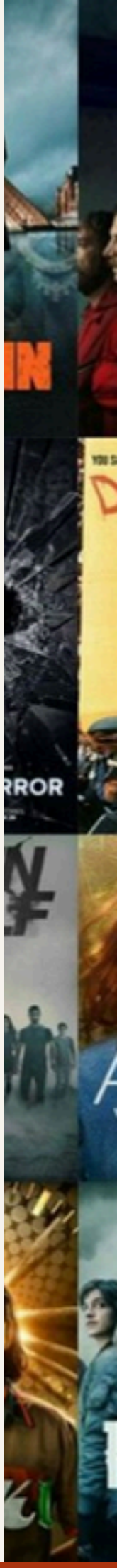
Element	Responsibility
Cross-cutting layer CS Cross-cutting layer SS	Rather than focusing on a single functional component, this layer usually addresses issues that impact several areas of the program. such as operational management, communication, and security.
Security Module CS Security Module SS	secures user data and prevents vulnerabilities by managing encryption, data protection, unauthorized access and other security-related concerns.
Op. Mgmt. Module CS Op. Mgmt. Module SS	This module oversees general functionality, performance, and health. It guarantees that the application operates seamlessly, effectively, and securely in production settings without compromising its essential business logic.
Communication Module CS Communication Module SS	This module is in charge of overseeing every facet of communication and data exchange between the application and outside components, services, or systems.

- **Client Side other elements:**

Element	Responsibility
Presentation layer CS	This layer focuses on the system's visual and functional interactions with the end user through the management of the user interface (UI) and user experience (UX).
UI Modules	In order to process requests, this module interacts with other layers, including the business logic layer, and manages user inputs, such as (clicks, keystrokes, and gestures).

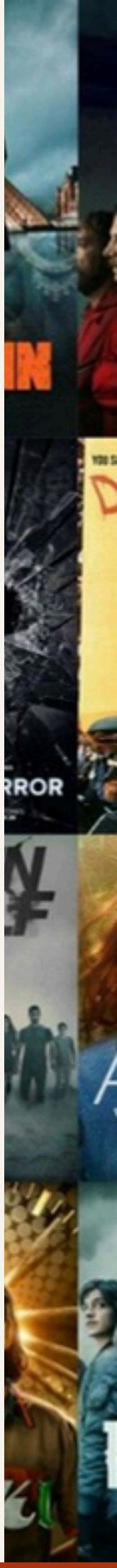


UI Process Modules	This module ensures a responsive and intuitive experience by managing user interactions, updating the interface, binding data to UI elements, validating inputs, and providing feedback.
Business logic layer CS	This layer manages the application's fundamental functions, including implementing business rules and processing user input. It oversees data processing, validation, computations, and decision-making, insuring that the client side executes the appropriate actions.
Bussiness Modules CS	This model exposes functionality from the application server or implements business activities locally. They preserve the application state while managing certain logic.
Bussiness Entities CS	In order to manage operations, enforce business rules, and preserve data integrity inside the domain model, business modules utilize this model, which represents fundamental objects (such as "User," "Order") with data and logic.
Data layer CS	Data retrieval and storage are handled by this layer. In order to ensure that data is fetched, stored, and synchronized effectively and preserve consistency throughout the application, it manages interactions with back-end services, APIs, or local storage.
Communication Module	Data communication between the client application and external services, like back-end servers or APIs, is handled by this module. Through abstracting the complexity of data retrieval, allowing other layers to focus without worrying about low-level communication details.



- **Server Side other elements:**

Element	Responsibility
Services layer SS	This layer It is linked to the client-side data layer and has models that reveal client-provided services.
Services interface	This module separates clients from implementation specifics by exposing high-level methods for accessing business logic. It gives other layers a standardized, modular method of interacting with the main features of the application.
Business logic layer SS	Implementing the business rules and logic that specify how the system functions and how data is handled and transformed is the main duty of this layer.
Bussiness Modules SS	based on a certain domain notion, this module is a logical collection of linked business functions, responsibilities, or features.
Bussiness Entities SS	This module uses the application to represent real-world business concepts (e.g., Customer, Order). stores data and applies relevant logic
Data layer SS	Data from several data sources, typically databases, in addition to file systems, third-party APIs, and other data repositories, are managed and made accessible by this layer.
DB Access Module	this module that communicates with the database directly. It abstracts the logic needed to execute database operations from the rest of the program and encapsulates it.
Time Server Access Module	By managing synchronization, time tracking, and communication with external time servers, this module is in charge of communicating with time-related services or systems.



Deployment diagram

Allocation view that shows the deployment locations of components associated with the modules in the earlier diagram.

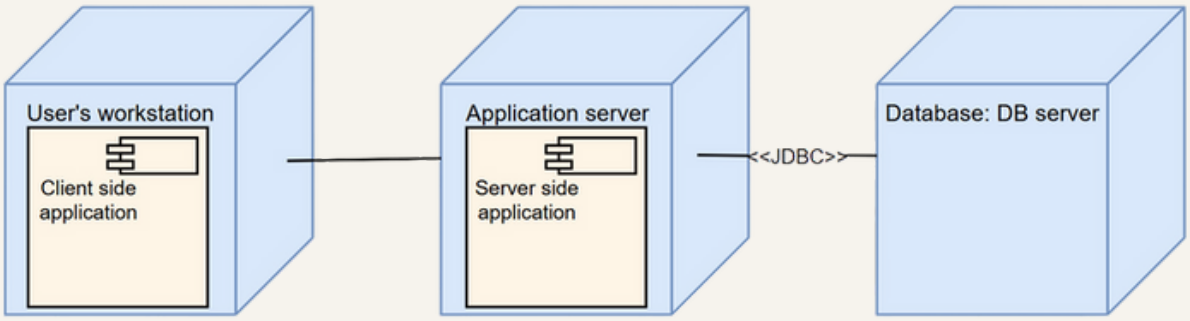


Figure 1.5 : Initial deployment diagram for CMS

Element	Responsibility
User's workstation	The server where the application's client side is hosted.
Application server	The server where the server-side logic is hosted.
Database server	The server where the database is hosted.

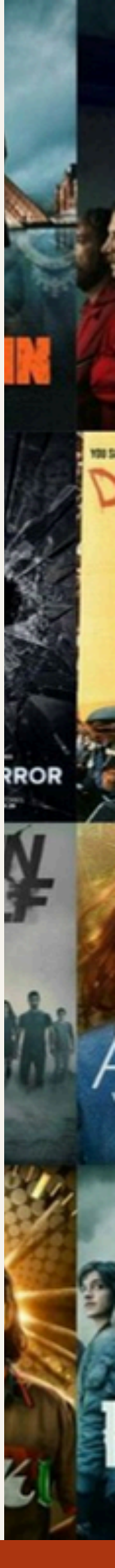
Relationships between some elements

Information on the relationships between some diagram elements that are worth noting is displayed in the following table.

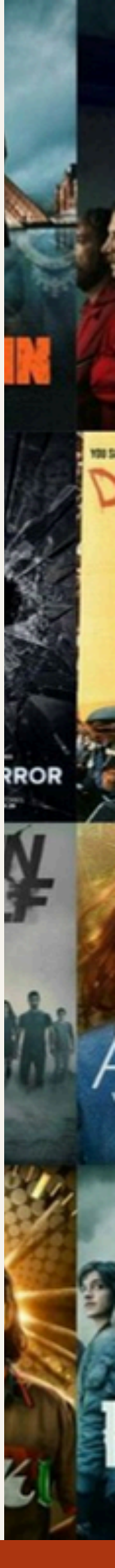
Element	Responsibility
Between application server and database serve	The Java Database Connectivity (JDBC) protocol will be used for database communication.

Step 7: Perform Analysis of Current Design and Review Iteration, Goal and achievement of Design Purpose.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	UC-6: Browse Movies		The chosen reference architecture determines the modules that will support this functionality.
	UC-9: Book Movie(s)		The chosen reference architecture determines the modules that will support this functionality.
	UC-10: Checkout		The chosen reference architecture determines the modules that will support this functionality.
	UC-13: Manage Movies		The chosen reference architecture determines the modules that will support this functionality.
QA-1: Latency			No relevant decisions made.
QA-5: Secure Payments			No relevant decisions made.



Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
	QA-7: Software Failure		Identify the elements derived from the deployment pattern that need to be replicated.
		CON-4	JavaScript in web applications facilitates rich client-side functionalities across diverse platforms like Windows, OSX, and Linux. In addition to different browsers, ensuring a seamless user experience.
		CRN-0	Selection of reference architectures and deployment pattern
CRN-4			No relevant decisions made.



Second Iteration :
Identifying Structures to Support Primary Functionality.

In this iteration, we move from generic and coarse-grained descriptions of functionality used in iteration 1 to more detailed decisions that will drive future critical steps such as implementation and formation of development teams. Since we have met the first iteration goal of establishing an overall system structure, our new goal is to reason about the units of implementation, which affect team formation, interfaces and how development tasks will be distributed and implemented.

Step 2: Establish Iteration Goal By Selecting Drivers

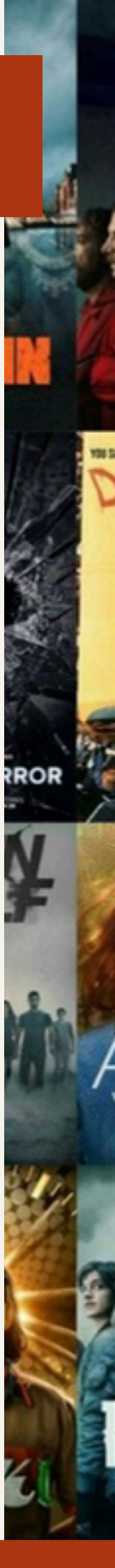
Finding structures to support primary functionality is a general architectural concern that this iteration aims to address. Finding these components helps address CRN-4, as well as for comprehending how functionality is supported.

In addition to CRN-4, the architect takes into account the main use cases for the system in this second iteration:

- UC-6: Browse Movies
- UC-9: Book Movie(s)
- UC-10: Checkout
- UC-13: Manage Movies

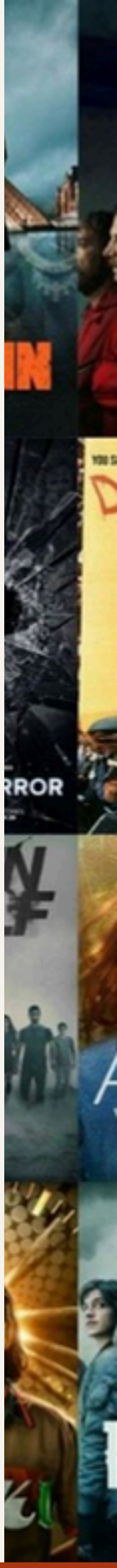
Step 3: Choose One or More Elements of the System to Refine.

The modules in the various layers defined by the two reference architectures from the previous iteration are the components that will undergo refinement in this iteration. In simple terms, the collaboration of components linked to modules located in the various layers is necessary to support functionality in this system.



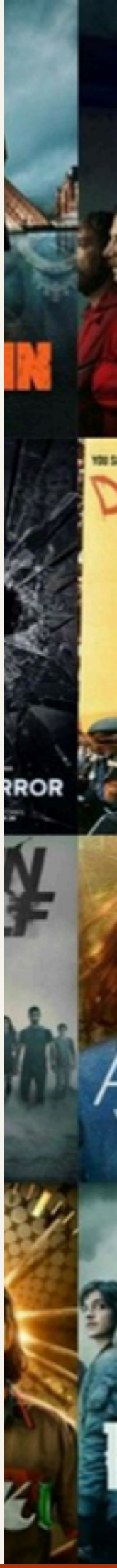
Step 4: Choose one or more design concepts that satisfy the selected drivers

Element	Responsibility
Create a Domain Model for the application	An initial domain model for the system must be created before beginning a functional decomposition. Identifying key entities, each has distinctive attributes and relationships. In addition to meeting requirements, this structure ensures modifiability and scalability. There is no good alternatives. A domain model must eventually be created to meet requirements and ensures modifiability and scalability. Otherwise, it will come out a suboptimal fashion, resulting in a ad hoc architecture that is difficult to understand and maintain..
Identify Domain Objects that map to the functional requirements	To guarantee that all needs are met and the drivers are satisfied, every functional component of the system needs to be enclosed within a self-contained domain object. This ensures that all business processes are appropriately represented and controlled by utilizing domain objects, preventing functional gaps and promoting modularity, maintainability, and adaptability. There is an alternative which is to directly decompose layers into modules without considering domain objects. However, this increases chance of not meeting some requirements.
Decompose Domain Objects into general and specialized Components	Domain objects represent sets of functionality in a system; however, this functionality is supported by finer-grained elements found in the various architectural layers. These layers divide the domain objects into specialized "components". The components in this case serve as the foundation for the more general domain objects, guaranteeing that every module is in charge of a particular area of the functionality. By dividing the system, we improve scalability, flexibility, and maintainability while also being sure that every domain object can be effectively expanded and controlled within its layer. There are no good alternatives to decomposing the layers into modules to support functionality.



Step 5: Instantiate Architectural Elements, Allocate Responsibilities and Define Interfaces

Design Decisions and Location	Rationale
Create an initial domain model	Identifying and modeling the entities involved in primary use cases (Book, Customer, Ticket, Payment, Movie, Show, Staff, Admin, Report), but just an initial domain model is created to accelerate the design phase.
Map the system usecases to domain objects	Through an analysis of the system's use cases, the domain objects (User Account Management, Movies Display, Booking Movies, Movies Management, Payment, Reporting) can be initially identified. All use cases have domain objects identified in order to address CRN-4.
Decompose the domain objects across the layers to identify layer-specific modules with an explicit interface	This method guarantees that modules supporting every functionality are identified; the architect will only carry out this effort for the primary use cases. Therefore, another team member can identify the remaining modules and allocate work to team members. In this stage, the architect recognizes the importance of testing the modules after they have been established. So, a new architectural concern is identified. CRN-7 : Most of the modules shall be unit tested. Because it is challenging to test the modules that implement user interface functionality individually, most but not all of the modules will be unit tested.
Associate frameworks with a module in the data layer	Encapsulated within the data layer modules, ORM (Object-Relational Mapping) is used to simplify database interactions. The object-oriented model of the system and the relational database underneath it are managed by these modules. In order to facilitate CRUD (Create, Read, Update, Delete) activities and abstract away the complexity of raw SQL queries, Hibernate, the selected ORM framework, is used to map Java objects to database tables. The Hibernate framework improves the system's data access layer's efficiency and maintainability by offering capabilities like transaction management, caching, and automated mapping. Movies, showtimes, clients, and reservations can all be handled effectively by the system thanks to this architecture.



Step 6 : Sketch Views and Record Decisions

As a result of the decisions made in step 5, several diagrams are created :

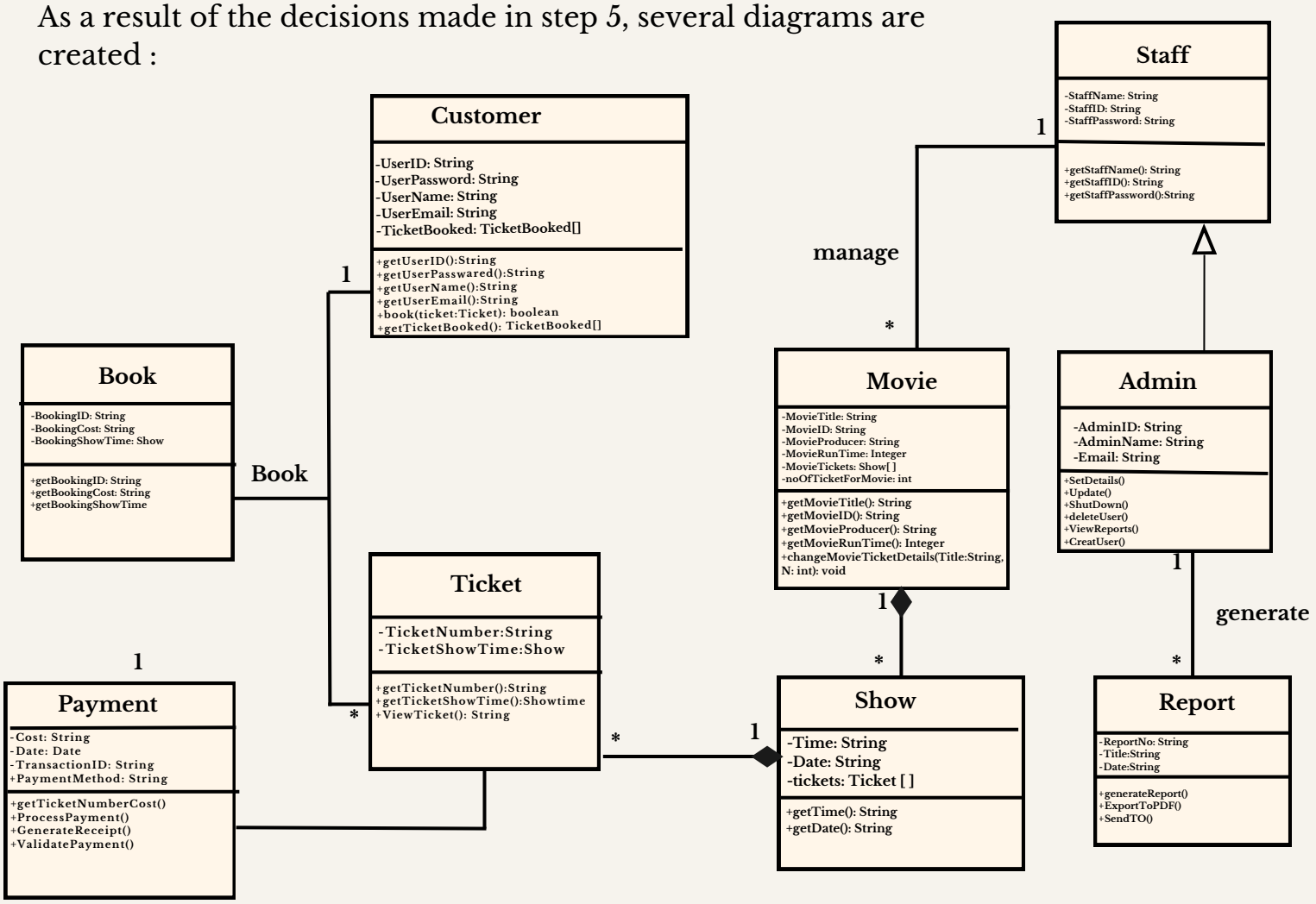


Figure 2.1 : initial domain model

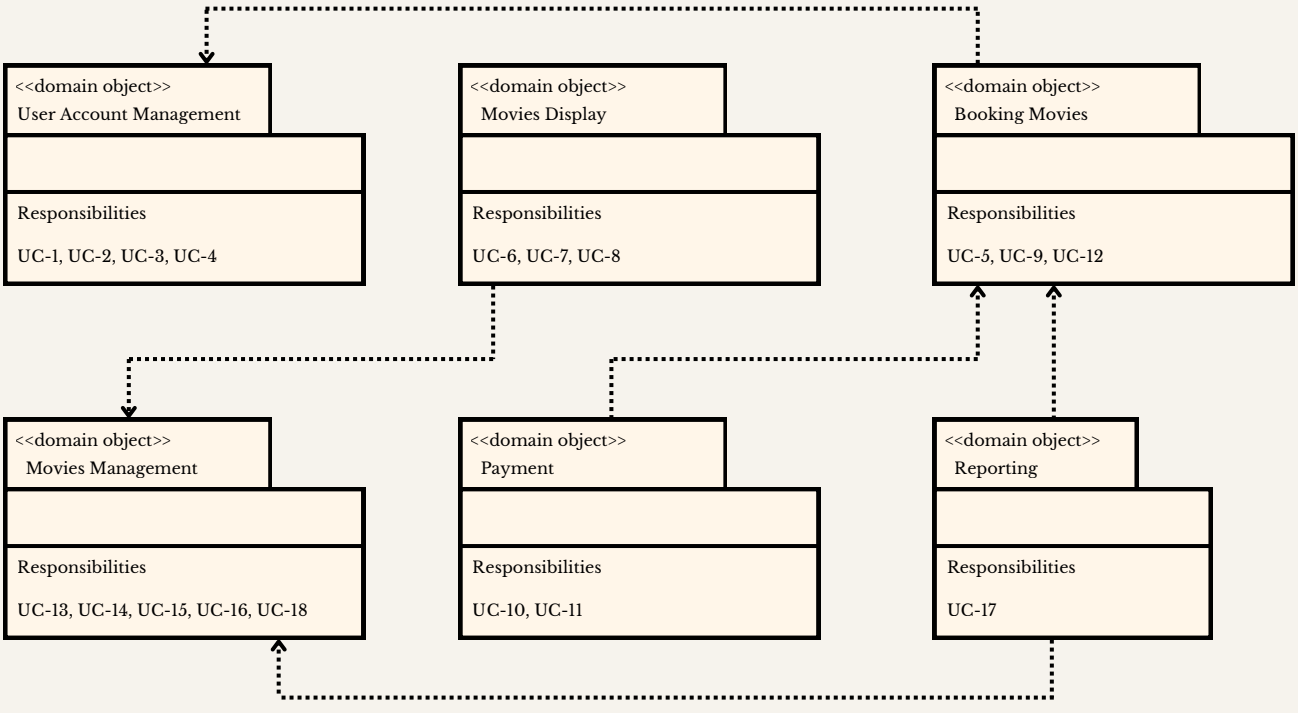


Figure 2.2 : Domain Objects associated with the use case model

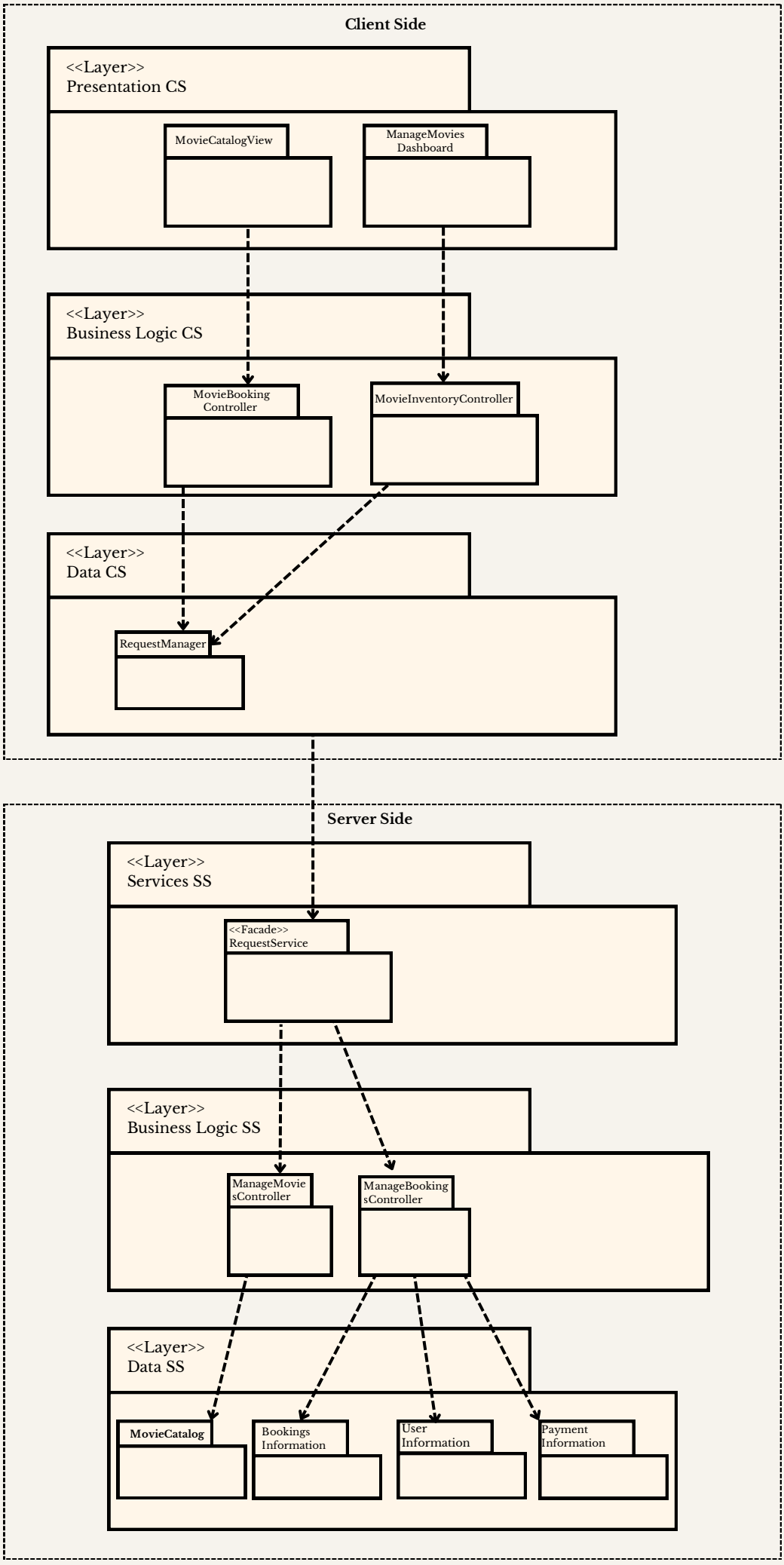
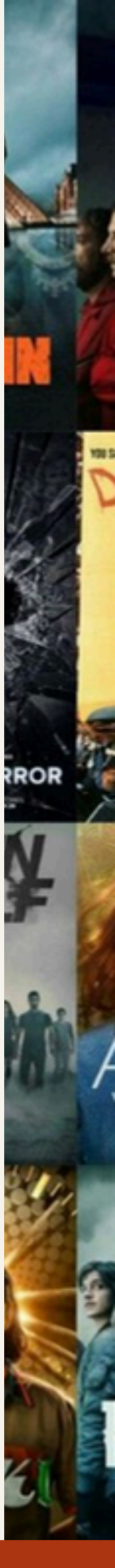


Figure 2.3 : Modules that support primary use cases

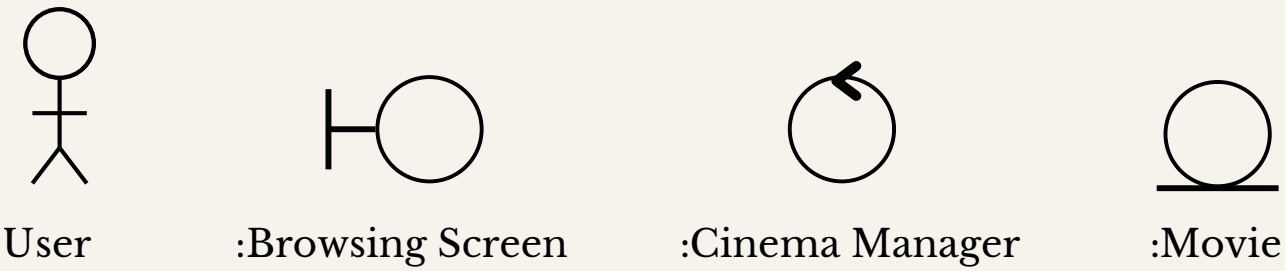
Each element in Figure 2.3 with its responsibility is in the following table :

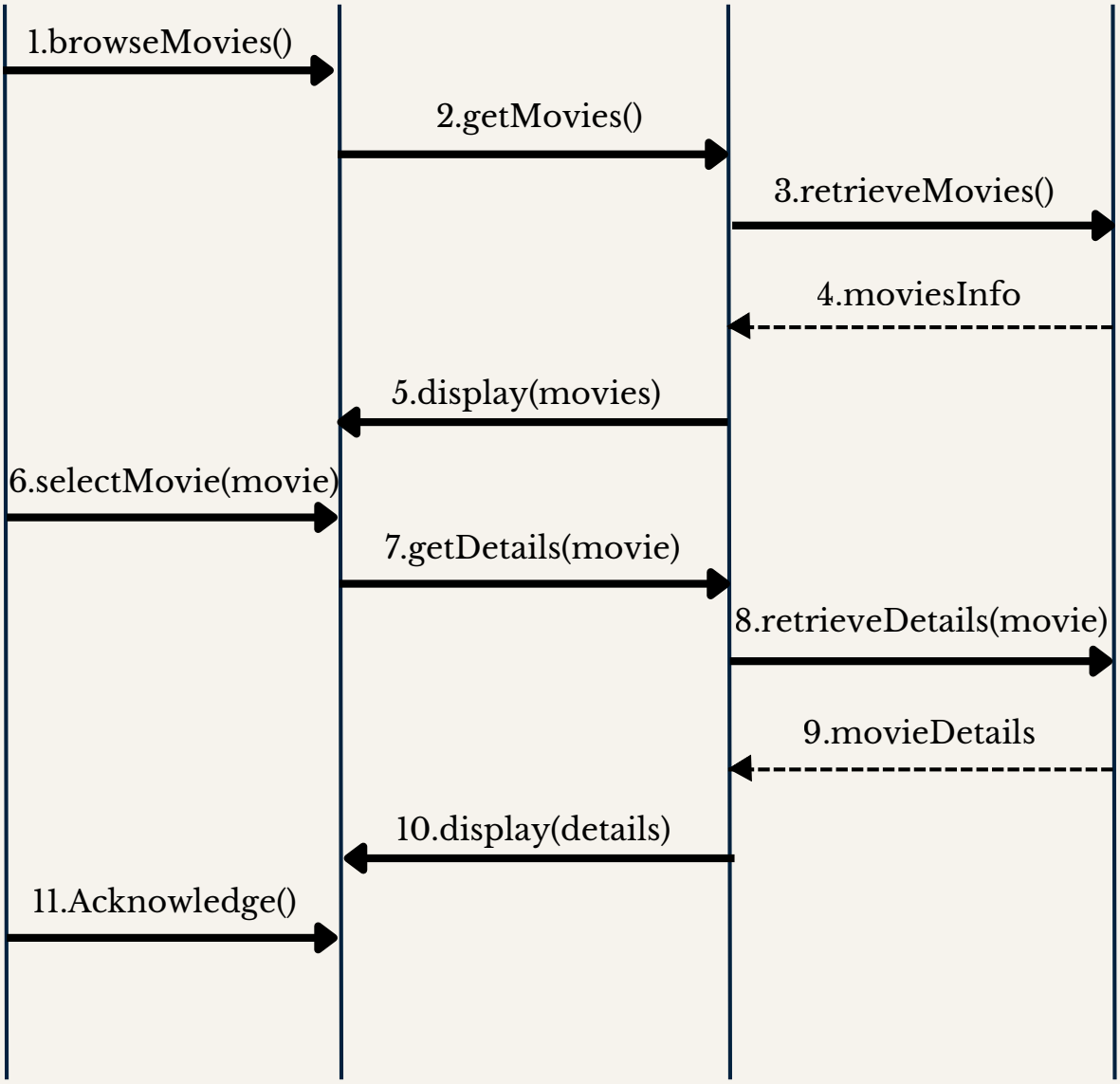
Element	Responsibility
MovieCatalogView	Data display, search, user interaction and accessibility.
StaffView	Updates, maintenance and content management
MovieBookingController	Responsible of process incoming booking requests from client, track user state, confirmation of booking, facilitate payment and allow updating.
RequestManager	manage user requests, monitor, allocate, facilitate and support the submitted requests.
<<Facade>> RequestService	Process the requests.
ManageMoviesController	Manage the movie requests, modification and manage all the updates, interaction and ensure data integrity.
MovieCatalog	organizing and managing the collection of movies.
Bookings Information	Contain booking information
User Information	Contain user information
Payment Information	Contain payment information



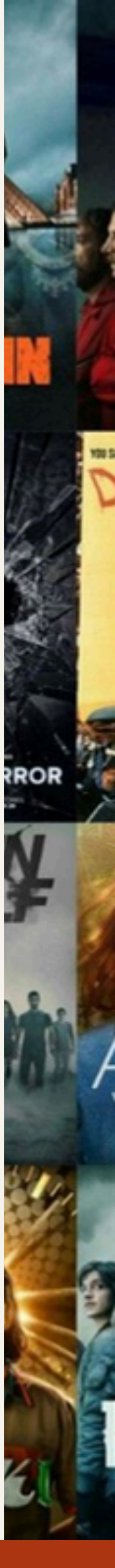
The followings are sequence diagrams for the primary use cases which are UC-6, UC-9, UC-10, UC-13

UC-6 (Normal Browsing)

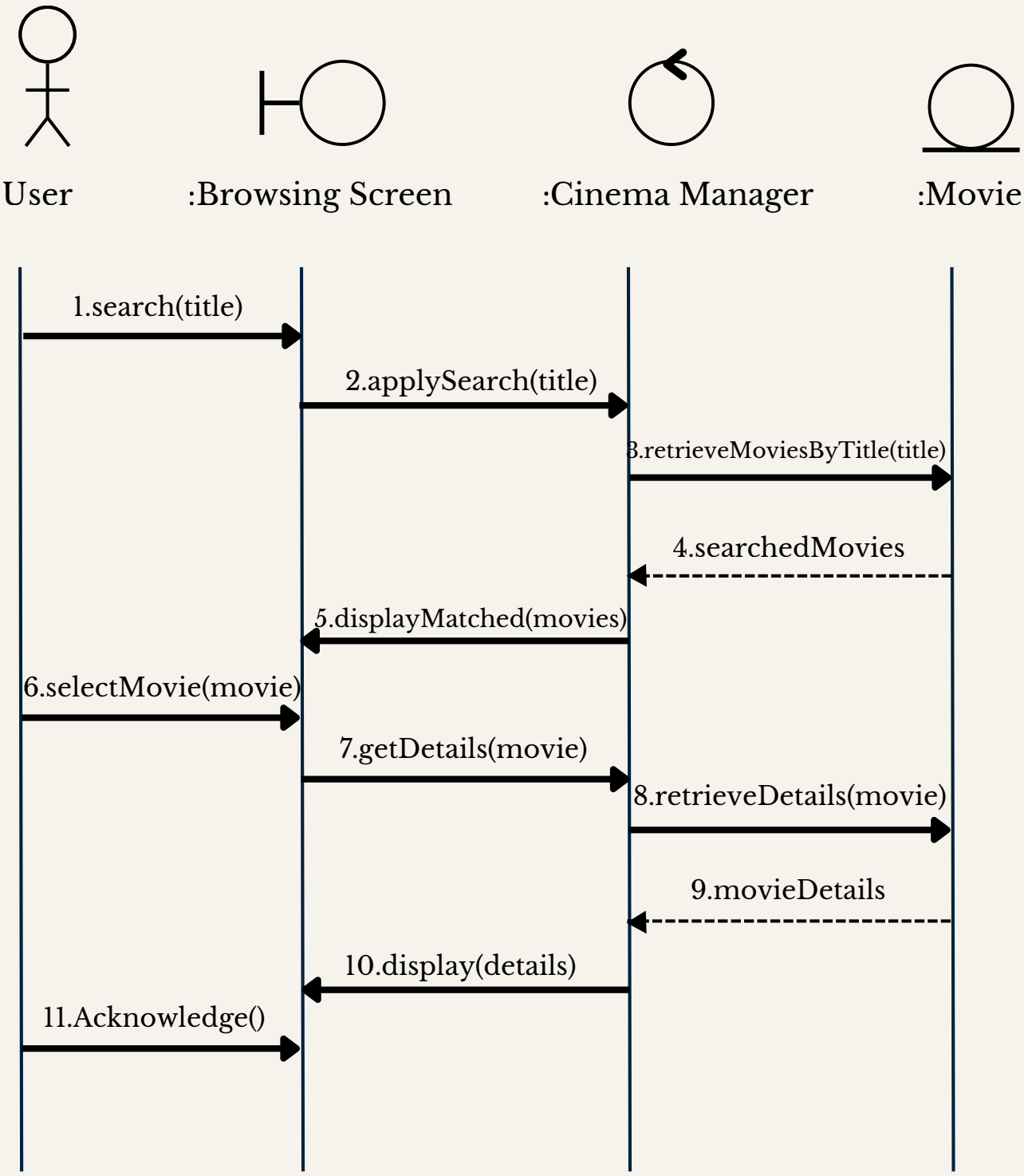




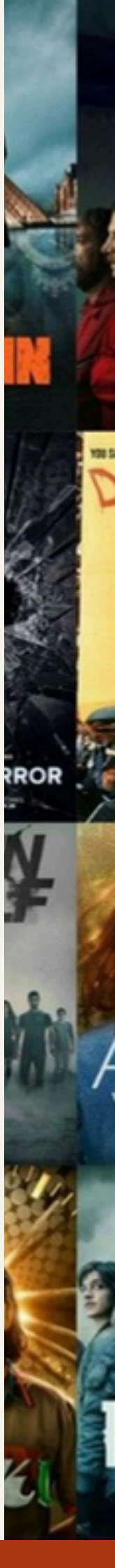
Method Name	Description
Element: Browsing Screen	
BrowseMovies()	Allow users to browse movies displayed in the screen
SelectMovie()	Allow users to choose a movie by clicking on it.
Acknowledge()	The user acknowledges the displayed details.
display(movies)	Takes a list of movies as parameter and display them in screen.
displayDetails(details)	Takes the details of a movie as parameter and display a page for movie details,
Element: Cinema Manager	
getMovies()	Return a list of all available movies.
getDetails(movie)	Return details of a specific movie.
Element: Movie	
retrieveMovies()	Retrieve data about all available movies from the data source.
retrieveDetails(movie)	Retrieve data about details of a specific movie from the data source.



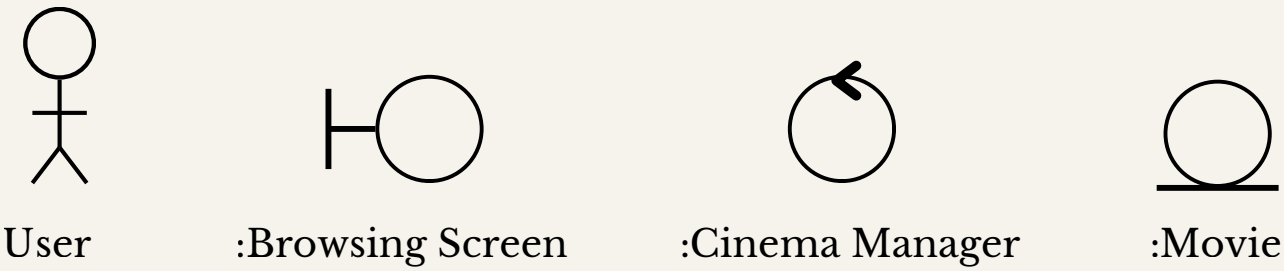
UC-6 (Searching)

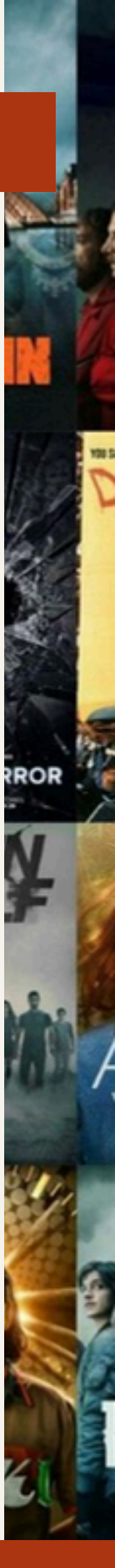


Method Name	Description
Element: Browsing Screen	
search(title)	Allow users to input a movie title in a search bar to search for it.
selectMovie(movie)	Allow users to choose a movie from the displayed list by clicking on it.
Acknowledge()	The user acknowledges the displayed details.
displayMatched(movies)	The browsing screen displays the list of matched movies to the user.
display(details)	The browsing screen displays the selected movie details to the user.
Element: Cinema Manager	
applySearch(title)	Takes the title to search as parameter and process the search request.
getDetails(movie)	Takes the selected movie as parameter and requests detailed information about it.
Element: Movie	
retrieveMoviesByTitle(title)	Fetches a list of movies that match the searched title from the data source.
retrieveDetails(movie)	Fetches the detailed information for the selected movie.

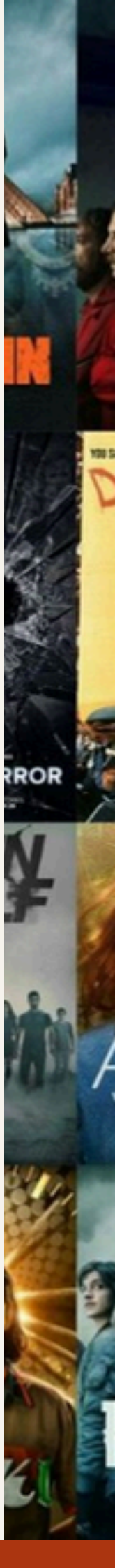


UC-6 (Filtering)

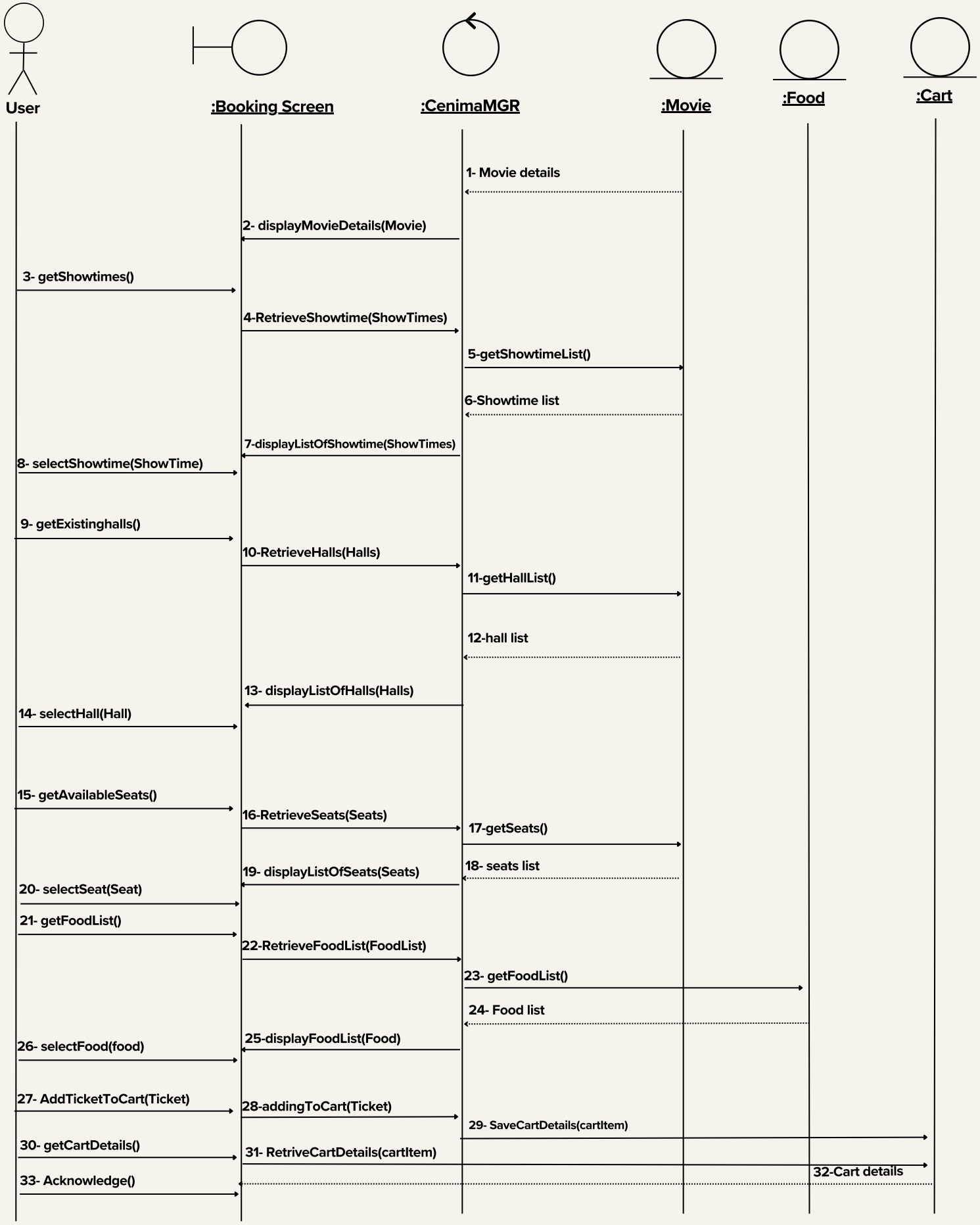




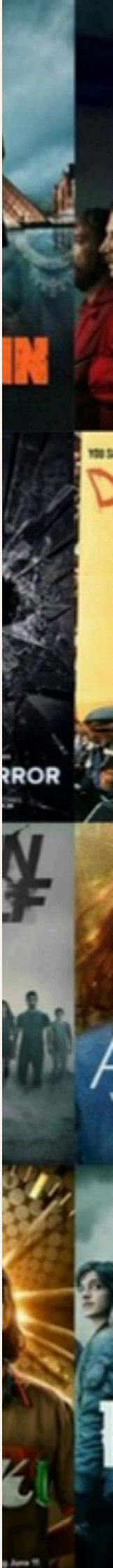
Method Name	Description
Element: Browsing Screen	
filter(criteria)	Allow users to select filter criteria to refine the displayed movies.
selectMovie(movie)	Allow users to choose a movie from the displayed list by clicking on it.
Acknowledge()	The user acknowledges the displayed details.
displayMatched(movies)	The browsing screen displays the list of matched movies to the user.
display(details)	The browsing screen displays the selected movie details to the user.
Element: Cinema Manager	
applyFilters(criteria)	Takes the criteria to filter as parameter and process the filter request.
getDetails(movie)	Takes the selected movie as parameter and requests detailed information about it.
Element: Movie	
retrieveMoviesByFilter(criteria)	Fetches a list of movies that match the filter criteria from the data source.
retrieveDetails(movie)	Fetches the detailed information for the selected movie.



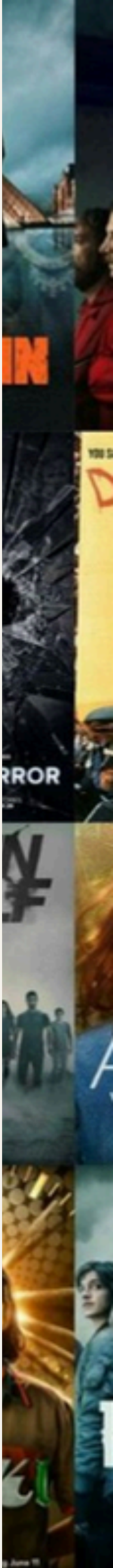
UC-9 (Booking)


User
:Booking Screen
:CenimaMGR
:Movie
:Food
:Cart
1- Movie details
2- displayMovieDetails(Movie)
3- getShowtimes()
4-RetrieveShowtime(ShowTimes)
5-getShowtimeList()
6-Showtime list
7-displayListOfShowtime(ShowTimes)
8- selectShowtime(ShowTime)
9- getExistinghalls()
10-RetrieveHalls(Halls)
11-getHallList()
12-hall list
13- displayListOfHalls(Halls)
14- selectHall(Hall)
15- getAvailableSeats()
16-RetrieveSeats(Seats)
17-getSeats()
18- seats list
19- displayListOfSeats(Seats)
20- selectSeat(Seat)
21- getFoodList()
22-RetrieveFoodList(FoodList)
23- getFoodList()
24- Food list
25-displayFoodList(Food)
26- selectFood(food)
27- AddTicketToCart(Ticket)
28-addingToCart(Ticket)
29- SaveCartDetails(cartItem)
30- getCartDetails()
31- RetriveCartDetails(cartItem)
32-Cart details
33- Acknowledge()

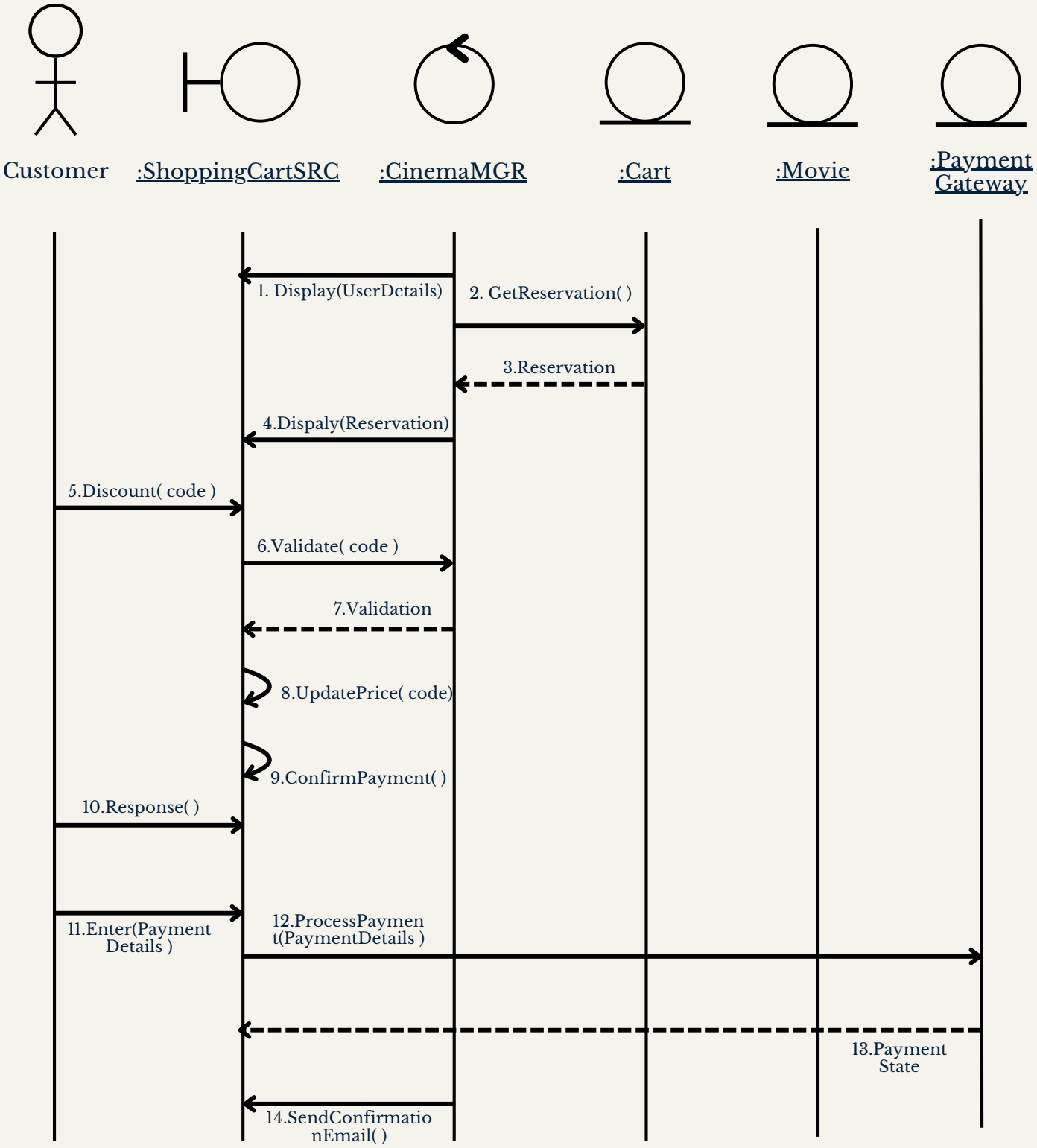
Method Name	Description
Element: Booking Screen	
displayMovieDetails()	The movie details will be displayed on the user screen.
displayListOfShowTime()	List of showtime will be displayed on the user screen.
selectShowtime(ShowTime)	User select showtime.
displayListOfHalls(Halls)	List of halls will be displayed on the user screen.
selectHall(Hall)	User select hall.
displayListOfSeats(Seats)	List of seats will be displayed on the user screen.
selectSeat(Seat)	user select seat/s.
displayFoodList(Food)	List of food will be displayed on the user screen.
selectFood(food)	user select food.
AddTicketToCart(Ticket)	user click the button (Add To Cart).
Acknowledge()	user confirmation.



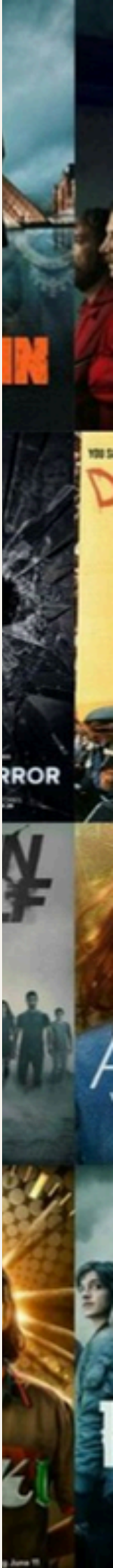
Element: Cinema Manager	
RetriveShowtime(ShowTime)	manager retrieve showtime from movie element.
RetriveHall(Hall)	manager retrieve Halls from movie element.
RetriveSeats(Seats)	manager retrieve seats from movie element.
RetriveFoodList(FoodList)	manager retrieve Food list from movie element.
AddingTicketToCart(Ticket)	manager Add ticket to cart.
RetriveCartDetails(cartItem)	manager retrieve cart details.
Movie	
getShowtimeList()	getting showtime list from movie element.
getHallList()	getting hall list from movie element.
getSeats()	getting seats from movie element.
Food	
getFoodList()	getting food from food element.
Cart	
SaveCartDetails(cartItem)	Saving cart details that consist of the user selection.



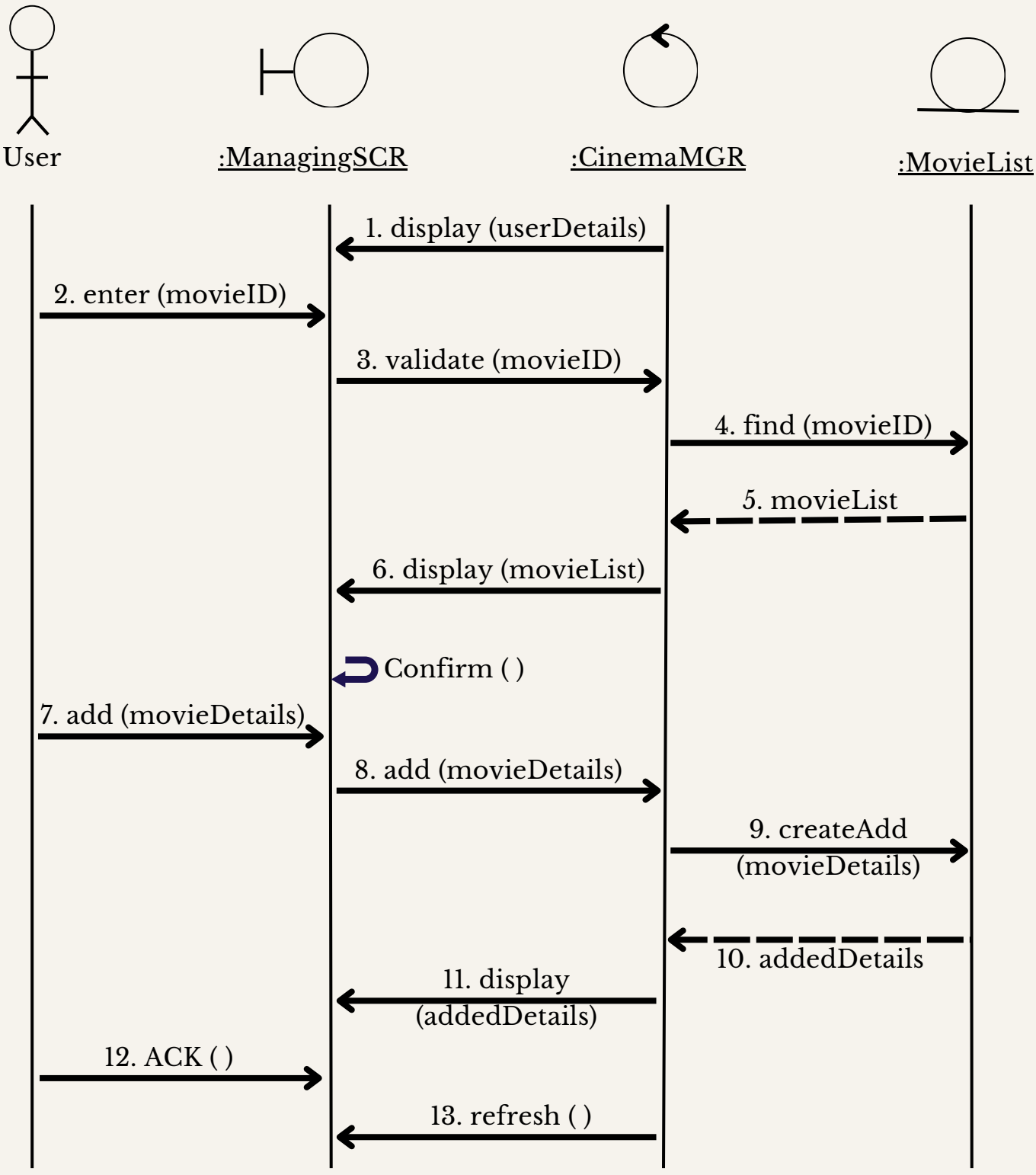
UC-10 (Checkout)



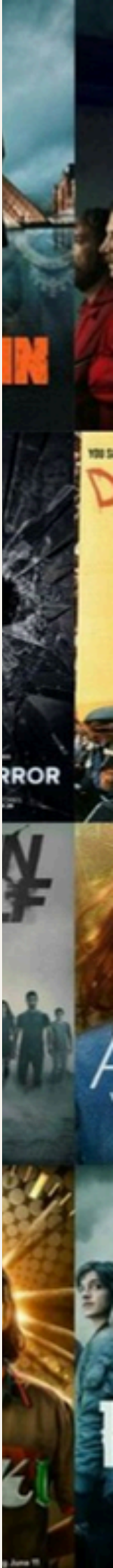
Method Name	Description
Element: Shopping Cart Screen	
Display(UserDetails)	Shows specific user details on the screen.
Dispaly(Reservation)	Shows user reservation retrieved from cart element
Discount(code)	Allows users to input a discount code if they have any to apply to the total reservation price
UpdatePrice(code)	updates the total price if a discount code was provided
ConfirmPayment( )	Ask the user if they want to proceed with the payment process
Response( )	Allow the user to confirm that they want to proceed with the payment process
Enter(PaymentDetails)	Allow the user to enterpayment details such as payment Method and Card Type
SendConfirmationEmail( )	Inform the user via email that the payment procedure succeeded without any problems.
Element: Cinema Manager	
Validate(code)	Verify whether the code exists and is still active
Element: Cart	
GetReservation( )	Get reservations the user has made from the cart element
Element: :Payment Gateway	
ProcessPayment(PaymentDetails)	Executes payment by finishing the transaction using the user's provided payment details



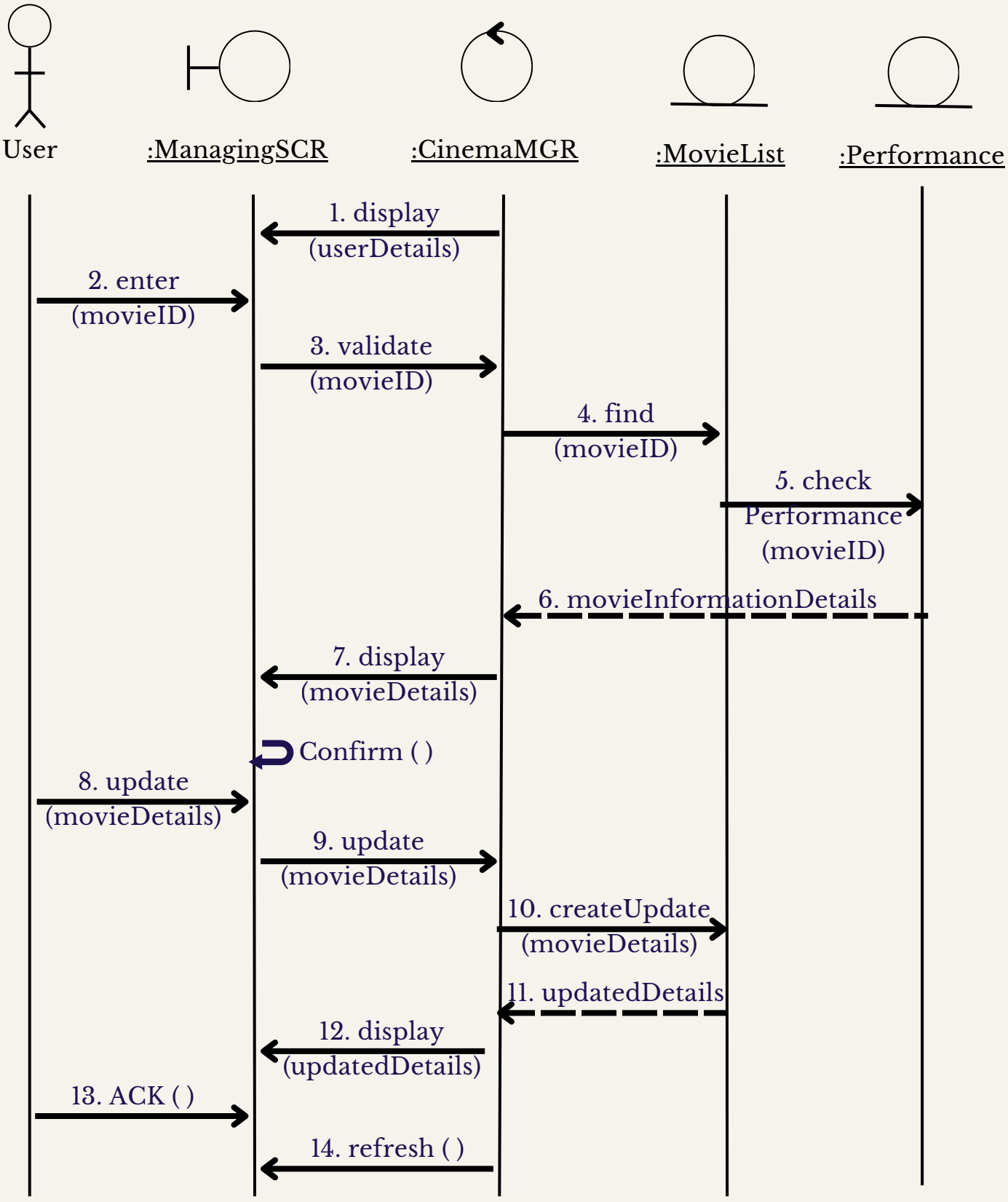
UC-13 (Add Movie)



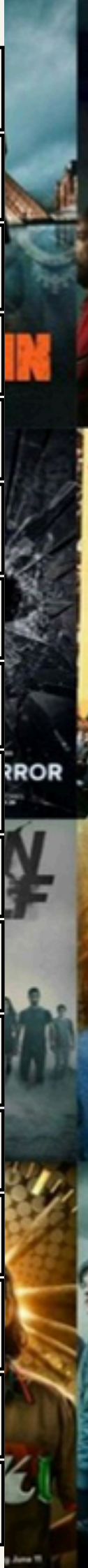
Method Name	Description
Element: Managing Screen	
display (userDetails)	Shows specific user details on the screen.
enter (movieID)	Allows users to input a movie ID for search and access
display (movieList)	Presents a list of available movies for user selection.
add (movieDetails)	Facilitates the addition of new movie details into the system.
display (addedDetails)	Exhibits details of a recently added movie on the screen.
ACK ( )	Confirms data reception in the system.
refresh ( )	Updates or reloads data in the system interface.
Element: Cinema Manager	
validate (movieID)	Verifies the correctness of the movie ID input within the system.
add (movieDetails)	Incorporates new movie details into the system for storage and display.
Element: Movie List	
find (movieID)	Locates details of a specific movie ID in the system.
createAdd (movieDetails)	Generates and incorporates new movie details into the system.



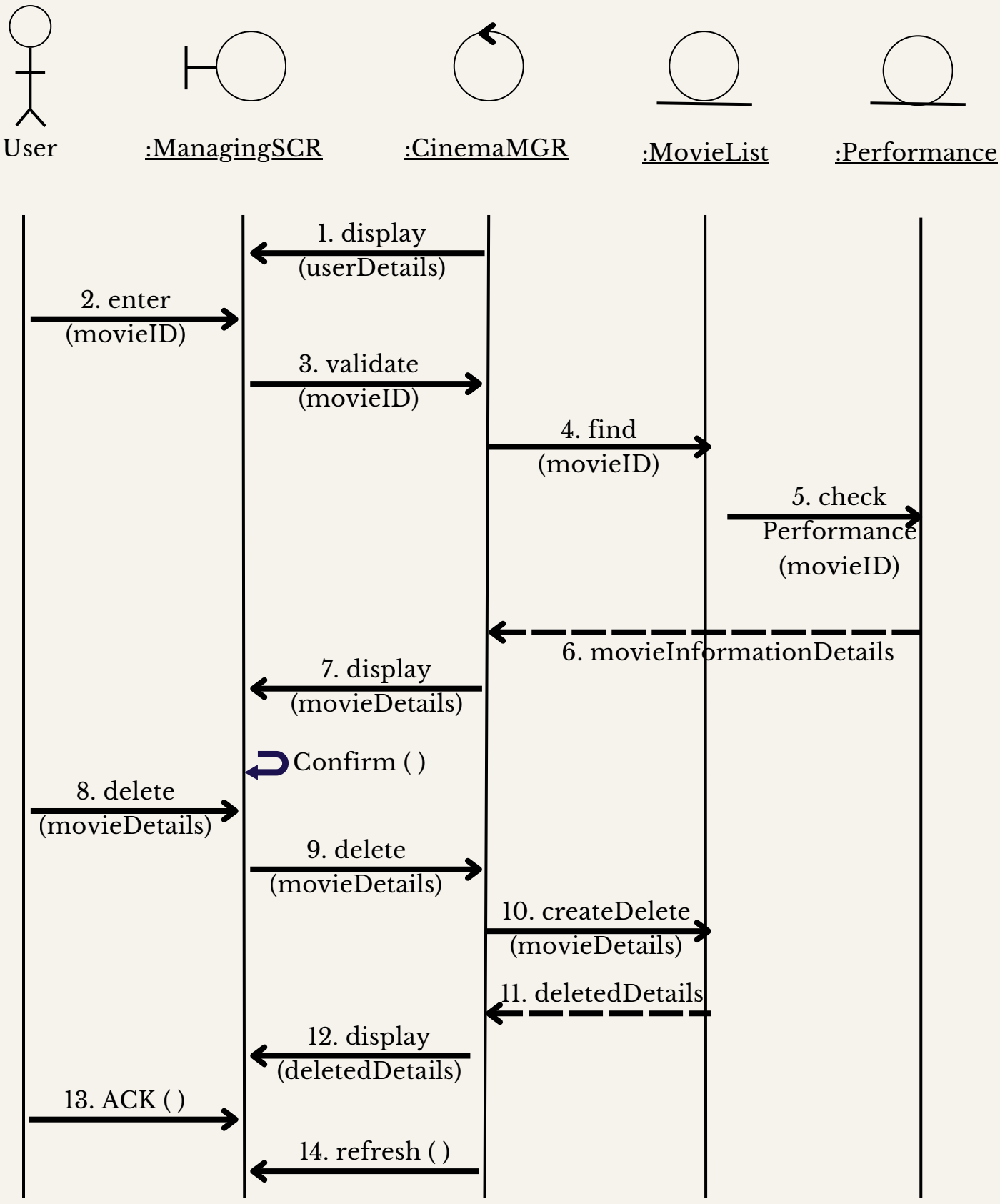
UC-13 (Update Movie)



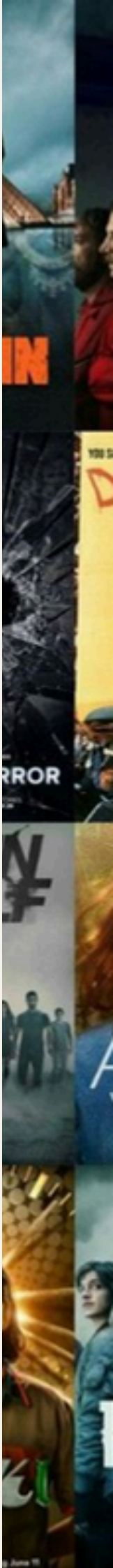
Method Name	Description
Element: Managing Screen	
display (userDetails)	Shows specific user details on the screen.
enter (movieID)	Allows users to input a movie ID for search and access
display (movieDetails)	Shows specific details of a movie on the screen.
update (movieDetails)	Incorporates new movie details into the system for storage and display.
display (updatedDetails)	Exhibits details of a recently updated movie on the screen.
ACK ( )	Confirms data reception in the system.
refresh ( )	Updates or reloads data in the system interface.
Element: Cinema Manager	
validate (movieID)	Verifies the correctness of the movie ID input within the system.
update (movieDetails)	Updates the system's database with fresh movie details for storage and display.
Element: Movie List	
find (movieID)	Locates details of a specific movie ID in the system.
createUpdate (movieDetails)	Adds and integrates new movie details into the system for storage and updating.
Element: Performance	
checkPerformance (movieID)	Confirms movie performance using its ID.



UC-13 (Delete Movie)



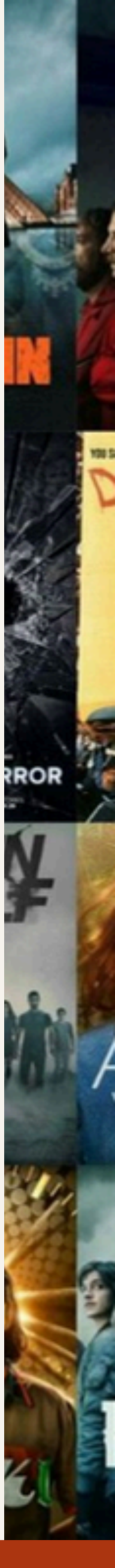
Method Name	Description
Element: Managing Screen	
display (userDetails)	Shows specific user details on the screen.
enter (movieID)	Allows users to input a movie ID for search and access
display (movieDetails)	Shows specific details of a movie on the screen.
delete (movieDetails)	Removes movie details from the system.
display (deletedDetails)	Shows details of a recently deleted movie on the screen.
ACK ( )	Confirms data reception in the system.
refresh ( )	Updates or reloads data in the system interface.
Element: Cinema Manager	
validate (movieID)	Verifies the correctness of the movie ID input within the system.
delete (movieDetails)	Removes movie details from the system.
Element: Movie List	
find (movieID)	Locates details of a specific movie ID in the system.
createDelete (movieDetails)	Adds movie details for deletion from the system.
Element: Performance	
checkPerformance (movieID)	Confirms movie performance using its ID.



Step 7 : Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose :

The decisions made in this iteration provided an initial understanding of how functionality is supported in the system. The modules associated with the primary use cases were identified by the architect, and the modules associated with the rest of the functionality were identified by another team member. From the complete list of modules, a work assignment table was created:

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
		UC-6, UC-9, UC-10, UC-13	Modules across the layers and preliminary interfaces to support this use case have been identified.
	QA-1		The elements that support the associated use case (UC-6) have been identified.
	QA-5		The elements that support the associated use case (UC-10) have been identified.
	QA-7		No related decisions made
		CRN-4	Modules associated with all of the user use cases have been identified and a work assignment matrix has been created.
	CRN-7		The concern —identified in step 5 in this iteration—, is partially solved through the use of inversion of control (IoC) approach to connect the components associated with the modules.



Third Iteration :
Addressing Quality Attribute Scenario Driver.

We can now begin to consider the fulfillment of some of the more significant quality attributes by building on the basic structural choices established in the first and second iterations. Only three of these quality attributes is the focus of this iteration.

Step 2 : Establish Iteration Goal By Selecting Drivers.

QA-1: Latency
When a user requests any normal operation, such as booking, searching, or filtering, the system should respond to user input and any event within 200 ms.

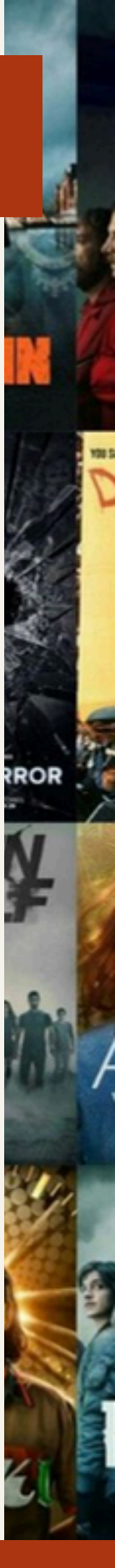
QA-5: Secure Payment
The system ensures that all payments are secure by using encryption algorithms, fraud detection, robust access controls, frequent audits, and secure transaction processing.

QA-7: Software Failure
The system should resume operations in less than 5 minutes after any downtime or interruption. This system guarantees a 99.9% uptime rate, allowing customers to seamlessly access the system anytime.

Step 3: Choose One or More Elements of the System to Refine

The elements that will be refined are the physical nodes that were identified in the first iteration:

- Application server
- Database Server
- User's workstation

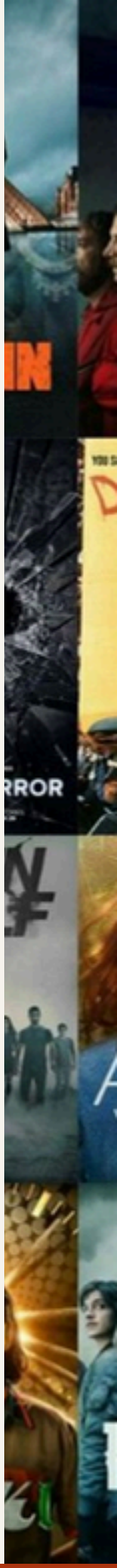


Step 4: Choose one or more design concepts that satisfy the selected drivers

Design Decisions and Location	Rationale
By replicating the application server and other crucial elements like the database, to implement the active duplication strategy	Replicating the essential components allows the system to function even if one of the replicated components fails.
Performance tactics: Introducing Concurrency	A load balancer evenly distributes incoming requests to multiple application servers, which allows to process requests simultaneously, enabling parallel processing.

Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces.

Design Decisions and Location	Rationale
Implement Load Balancing for Application Server	To split up incoming web traffic among several instances, implement a load balancing system for the Application Server. By avoiding overload on a single server, this will help increase scalability, optimize resource utilization, and guarantee high availability.
Database Replication	To generate redundant copies of data across several servers, configure database replication on the database server. By enabling read operations to be divided among replicated servers, this design idea can improve read performance, increase fault tolerance, and improve data availability.
Client-Side Caching for User's Workstation	Use client-side caching techniques on the user's workstation to locally store data that is frequently accessed. You can lower network latency, increase responsiveness, and improve the cinema management system's overall user experience by caching data on the client side.



Step 6: Sketch Views and Record Design Decisions.

A refined deployment diagram is created that includes the introduction of redundancy in the system, shown in figure 3.1

* <<JDBC>> is Java Database Connectivity

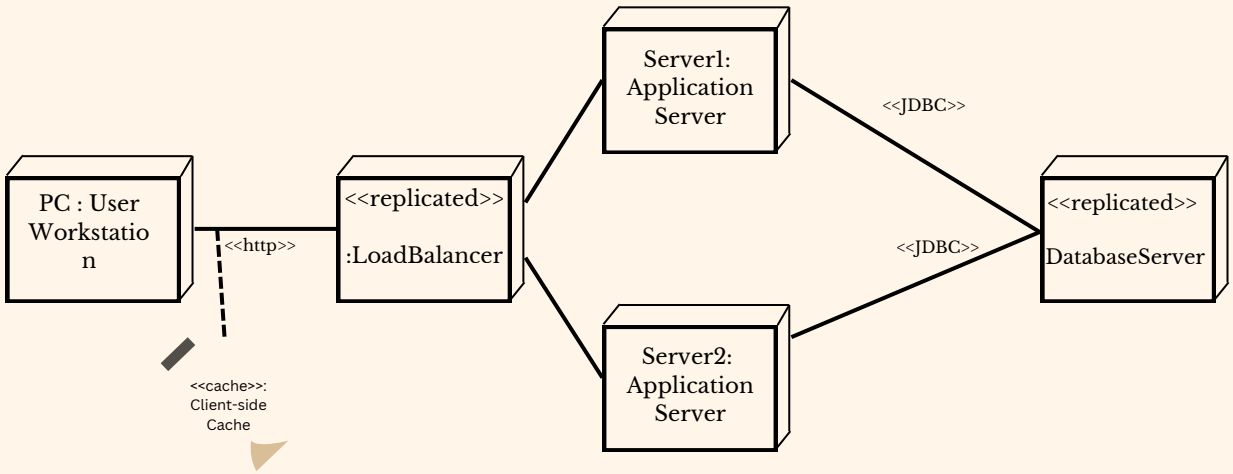


Figure 3.1 : Refined deployment diagram for CMS

Responsibilities of elements that were not listed in previously :

Element	Responsibility
Load Balancer	To distribute incoming traffic across multiple application servers. in order to ensure high availability and prevent overload on a single server. It is located between the user workstation and the application servers.

The following UML sequence diagram shows how load balancers that was introduced in this iteration exchanges messages with other elements, shown in deployment diagram figure 3.1, to support QA-1 (latency). The purpose of this diagram is to illustrates the communication between the physical nodes. Note: name of methods are preliminary.

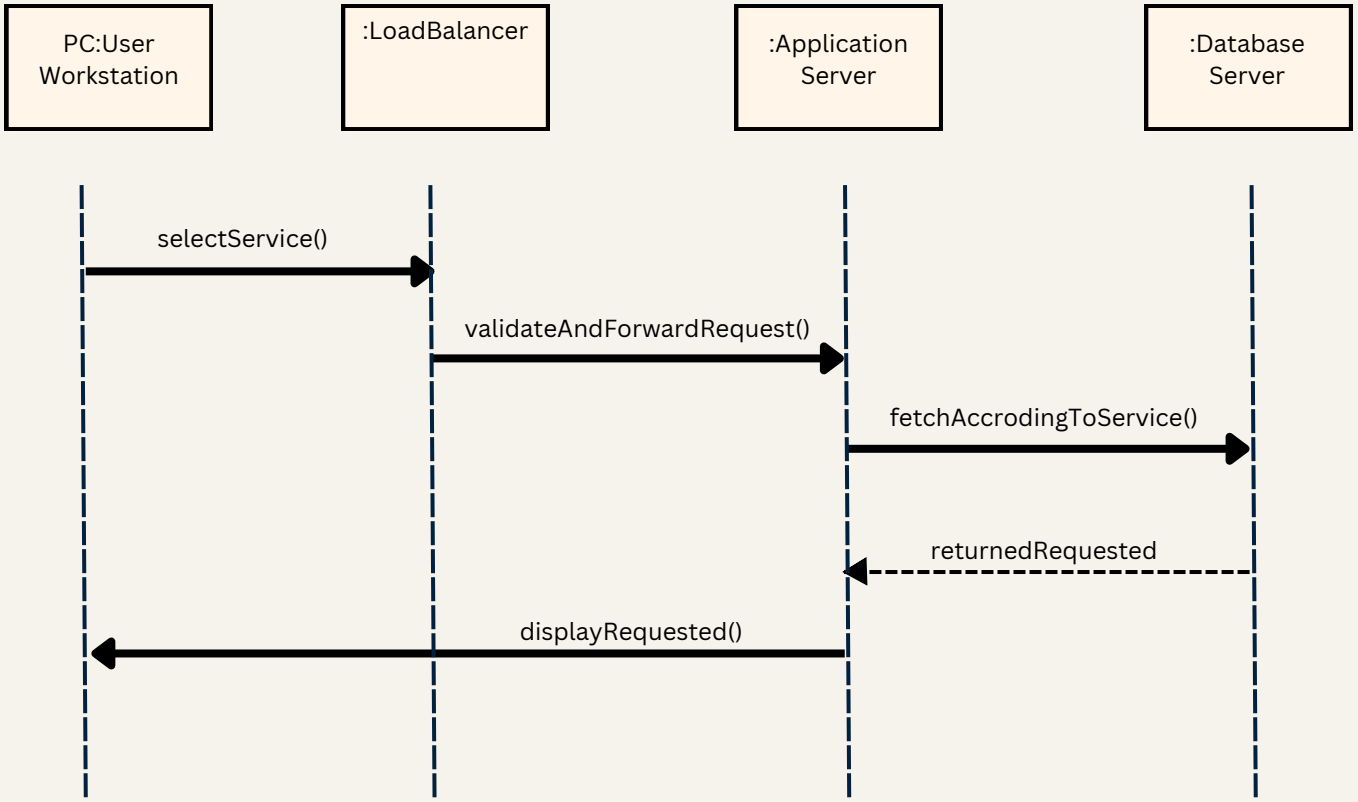


Figure 3.2 : Sequence Diagram illustrates the messages exchanged between the physical nodes to support QA-1

It represents the interaction between various nodes of a system to process a user's request. The user workstation component represents the end user initiating a request. This request could be any of the functionalities of the system such as selecting a movie, searching, filtering, booking, paying or an action to manage movies inventory. Then, the load balancer, which acts as an intermediatory between the user and backend servers, will validate the incoming request and forwards it to the appropriate server for processing based on the load on available servers. After that, the application server will process the forwarded request and communicate with the database server to fetch or manipulate data as needed. The database server handles operations related to the stored data by querying or updating database based on the request.

Step 7: Perform Analysis of Current Design and Review Iteration, Goal and achievement of Design Purpose.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
		QA-1	The elements that support the associated use case (UC-6) have been identified.
		QA-5	The elements that support the associated use case (UC-10) have been identified.
		QA-7	The elements that support the associated use case (UC-9) have been identified.
	CRN-7		No relevant decisions made