Contents

[Browse theater- 2](#_Toc56448723)

[List of nouns: 2](#_Toc56448724)

[List of possible operations: 2](#_Toc56448725)

[Browse movies- 2](#_Toc56448726)

[List of nouns: 2](#_Toc56448727)

[List of possible operations: 2](#_Toc56448728)

[Use Case Scenario: Reserve Seats 3](#_Toc56448729)

[Good-Candidate Objects: Reserve Seats 3](#_Toc56448730)

[Potential Operations: Reserve Seats 3](#_Toc56448731)

[Use Case Scenario: Purchase Tickets 3](#_Toc56448732)

[Good-Candidate Objects: Purchase Tickets 4](#_Toc56448733)

[Potential Operations: Purchase Tickets 4](#_Toc56448734)

[Use Case Scenario: Manage Annual Fee 4](#_Toc56448735)

[Good-Candidate Objects: Manage Annual Fee 4](#_Toc56448736)

[Potential Operations: Manage Annual Fee 4](#_Toc56448737)

[Cancel reservation for a regular user: 4](#_Toc56448738)

[Candidate objects for regular user: 5](#_Toc56448739)

[Operations: 5](#_Toc56448740)

[Cancel reservation for a registered user: 5](#_Toc56448741)

[Candidate objects for registered user: 6](#_Toc56448742)

[Operations: 6](#_Toc56448743)

[Register User: 6](#_Toc56448744)

[Candidate objects: 6](#_Toc56448745)

[Operations: 7](#_Toc56448746)

[Use Case Scenario: View Showtimes 7](#_Toc56448747)

[Good-Candidate Objects 7](#_Toc56448748)

[Potential Operations 7](#_Toc56448749)

[Use Case Scenario: Send Receipt 7](#_Toc56448750)

[Good-Candidate Objects 8](#_Toc56448751)

[Potential Operations 8](#_Toc56448752)

# Short description about each actor

* An ordinary (or regular) user is someone who employs the movie theater ticket reservation app to search for a movie, select a specific theater, view available showtimes and seats, select a seat, make their payment and receive tickets and confirmation, but does not have their information registered on the system’s database.
* A registered user is an ordinary user who has had their information saved on the system’s database and is therefore entitled to certain attendant perquisites contingent upon their payment of an annual account fee.
* A financial institution is a corporation that is responsible for processing and finalizing transactions related to the purchasing of movie tickets in the reservation app.
* A database engine is a system that interacts with the application in order to perform the actions needed to communicate data to and from the database.
* A manager is responsible for coordinating all of the activities occurring within the system of the movie theater ticket reservation app.

# Browse theater

A user may start this use case either with or without completing the view movies use case. A user is to select a theater. Both types of users, regular and registered, can choose movie theater locations. Users will start the process of going to see a movie by either manually searching a theater by name or by viewing a list. Only one location can be selected in order to move forward. Once a theater is selected user will move onto the next use case, Browse Movie, or if the movie has been selected another use case such as reserve seats.

## List of nouns:

Regular User, Registered User, Movie, theater, Theater List

## List of possible operations:

* Get-theatername
* Get-theaterlist
* Select-theater
* Search-fortheaternames
* Show-theaterlist
* Show-theatername

# Browse movies

This use case can be started by either completing the browse theater prior to starting this, or without having a theater selected. A new screen will appear giving the User the option of seeing a list of movies that are currently being shown at that theater, or to manually search the name of a movie. Both types of users will see the same screen. All movies will display a check mark to indicate whether registered users are able to have early access to select the movie before regular users. No check mark means that all users can select the movie at the same time. As a regular user if the movie does not have a checkmark them they may proceed as normal. Each movie could have several times that it is being shown and could be viewed in different theaters at the same time. User must pick a movie in order to proceed to further. Once a movie is selected the User can move to select a showtime, or the theater if not previously selected.

## List of nouns:

Regular User, Registered User, Theater, MovieList, Movie, showTime

## List of possible operations:

* Get-movielist
* Select-movie
* Show-movies
* Search-movies
* Show-movie list

# Use Case Scenario: Reserve Seats

This use case begins when the user has already selected a showtime for a chosen movie and theater combination. At this point, the system will display a graphical representation of the seating map of the theater room, showing the seats that have already been sold in one colour and those that are still available in another. For the films that have not yet had a public announcement, registered users will be able to purchase up to 10% of seats on a first come, first serve basis. After this 10% of seats has been reserved, registered users will need to reserve their seats after announcement. For all users, if there are no more or not enough seats available, the system will inform a user that the show is sold out and encourage them to consider another film. Otherwise, the user will have the opportunity to reserve one or more specific seats or to simply press cancel to annul seat selection entirely. Any available seat selected by the user will be highlighted and if a user selects a seat that has already been reserved and confirmed, the system will display a message to the user and ask them to pick another seat. After the correct seats have been highlighted, the user will confirm their selection with the system. After the user confirms that these seats are the ones they want to reserve, the system will immediately ensure that said seats may no longer be accessed by future users. The user will then be informed that they have successfully selected their preferred seat(s) and can move forward with the process. The database engine will craft an update with this new information and both the available seat count and images displayed in the app will change accordingly. After this point, the use case ‘Purchase Tickets’ will become available to the user.

## Good-Candidate Objects: Reserve Seats

User, Movie, Showtime, Theater, Seat, Ticket, Map, Pattern, Message

## Potential Operations: Reserve Seats

* Display-availability
* Select-seat
* Cancel-seat-selection
* Confirm-seat-reservation
* Check-announcement-status
* Inform-sold-out

# Use Case Scenario: Purchase Tickets

After a user has chosen the seats to reserve for their performance, and proceeded to checkout, this use case commences. First, the system will show the user a summary of the ticket costs including taxes. The system will also inform the user at this point that, if they so wish, they may now apply any existing coupon codes or points towards the purchase price. If the user opts to do so, they will need to enter in the relevant information and apply it here. The system will accept this and modify the summary accordingly. If there is still an outstanding balance, the system will then prompt user to proceed to selecting a method of payment. Next, the system will determine which type of user it is dealing with. If a user is registered, they will be given the option to skip immediately to final confirmation of their booking details. An ordinary user will at this point be asked by the system to select their financial institution of choice, enter relevant personal information such as name, address, and email and to enter credit card number, expiry date, and security code (if credit card was chosen as form of payment) or other pertinent payment information. The system will communicate with the appropriate institution to establish the validity of the given information to accept payment. Then, for both ordinary, and registered users, the payment will be submitted. After the payment has been submitted, the system will confirm the completion of the payment process within the application and end the interaction with the user, shuttling them back to the application’s landing page. The system will update the database with the concluded purchase’s information through the database engine. It will also update the database regarding the used codes/points. The ‘Send Receipt’ use case will follow.

## Good-Candidate Objects: Purchase Tickets

User-information, Card-information, Payment, Coupon, Summary

## Potential Operations: Purchase Tickets

* Submit-payment
* Confirm-payment
* Apply-code
* Present-summary
* Choose-payment-method
* Check-payment

# Use Case Scenario: Manage Annual Fee

When a user registers, they will be prompted by the system to make their first $20.00 annual fee payment to complete their registration. The system, which should now be keeping track of the registered user’s information, will from this point forward annually contact the user via their email to the remind them of their payment and ask that they make it before the deadline. This email will be sent one month before the registration anniversary. If the user makes their payment on time, the system will confirm that the fee has been paid and the user’s registration will remain in the system. If a user fails to pay their annual fee, the system will remove them from the list of registered users, and they will have to redo the process if they would like to rejoin.

## Good-Candidate Objects: Manage Annual Fee

Reminder, RegisteredUser

## Potential Operations: Manage Annual Fee

* Check-Fee-Status
* CheckDeadline
* Email-reminder

# Cancel reservation for a regular user:

This use case starts when a regular user selects the option of looking for a reservation and inputs the reservation number, when all the information is displayed the user selects the cancel a reservation option. The system will check if the movie starts within the next 72 hours. If it does, it displays a message saying cancellation is unavailable for this reservation and use case terminates. The system will display a message informing the user of the cancellation policy and wait for the user to accept it. If the user does not accept, a message will be displayed and use case terminates. If the user accepts, the system labels the seats in that reservation as available. The system will create a new coupon, with the store credit amount and a unique code number. The system will send an email, to the email the user used to make a reservation, including the details of the cancelation and the previously created code number.

## Candidate objects for regular user:

* Movie
* Message
* Reservation
* User
* Seats
* Email
* Coupon-code
* Store-credit-amount

## Operations:

* Display-message
* Cancel-reservation
* Check-movie-start
* Cancellation-unavailable
* Cancellation-policy
* Seats-label
* Send-email

# Cancel reservation for a registered user:

This use case starts when a registered user logs into their account, checks the reservation that is going to be cancelled and selects the cancel reservation option. The system will check if the movie starts within the next 72 hours. If it does, it displays a message saying cancelation is unavailable for this reservation and use case terminates. The system will display a message asking the user to confirm the cancelation and wait for the user to accept it. If the user does not accept, a message will be displayed and use case terminates. If the user accepts, the system labels the seats in that reservation as available. Also, the system adds the credit amount refunded to the user's account.

## Candidate objects for registered user:

* Login
* Reservation
* Message
* Movie-start
* Cancelation
* User
* Seats
* Store-credit-amount
* User-account

## Operations:

* Login-account
* Check-reservation
* Cancel-reservation
* Display-message
* Cancelation-unavailable
* Confirm-cancelation
* User-accept
* Seats-label

# Register User:

This use case starts when a regular user selects the register option. The system displays a message and text fields asking the user to input their first name, last name, date of birth, email and password. The system checks for that email address in the database, if found, the system displays a message saying that email is already registered, and the use case ends. Otherwise, the system requests the user for a payment method for the annual account fee. If the payment is unsuccessful, the system will display a message asking to try again and will let the user input the payment information again. If the payment is confirmed, the system adds the user’s information, including the method of payment, to the database and a message is displayed to the user confirming their registration.

## Candidate objects:

* Message
* Input
* Email
* Payment
* Account-fee
* User
* Registration

## Operations:

* Register-option
* Display-message
* Check-email
* Email-registered
* Request payment
* Confirm-payment
* Add-user
* Confirm-registration
* Add-payment-option

# Use Case Scenario: View Showtimes

This use case begins when user already selected a theater and a movie from the available movies list and selected ‘view showtimes’ function/button. Afterwards, the system displays a list of available showtimes of this movie (for today) at the selected theater in a list format. The top of the list begins with the showtime which is the closest to the actual time, and following showtimes are later in order. If a showtime is selected by the user, then use case ‘Reserve Seats’ become available.

## Good-Candidate Objects

User, Movie, Showtime, Theater, Seat, Ticket, Map, Button, Page, Date time, day

## Potential Operations

* Select showtime
* go to next day
* go to previous day,

# Use Case Scenario: Send Receipt

This use case begins when user already purchased a ticket. At this point, the system will display a button to the user to Send Receipt. It this is clicked, the system will create an electronic receipt containing name of the movie, theater, date, time and seats purchased. Then on the next page, the system will display the user the email address registered (for registered users) or the email address used during the purchase (for regular users) and ask the user to confirm. The user has the option to click on that email address and change it, as they wish, or simply click confirm if they wish the proceed with the email address which was already shown. If user changes the email address, this will be used as the recipient of the email but this won’t be used to update the email address on the database for this transaction (since this might be just a temporary email’s address). Then, system creates an email with receipt information in the body of mail and sends it to the user’s selected email address. Then it displays the user the result of the email sending operation (sent, connection error, etc).

## Good-Candidate Objects

User, User-information, email address, ticket, receipt, page,

## Potential Operations

* Send email
* Change email address
* Confirm email address
* Check email address
* Create email body

# Use Case Scenario: Make Payment

This use case starts when a user needs to be charged money. The system displays a message with the description of what the payment is for and the amount to be charged, below it will have a text field asking the user to input their payment information. When the user enters this, the system will send a request to the financial institution with the user’s information and amount to be charged. If the payment is denied the system will notify the user and the use case will restart. If the payment is successful, the system will display a message confirming the payment. Then the use case will switch into Send Receipt use case.

## Good-Candidate Objects

Amount, Message, Payment, Payment Information, Request, Financial Institution, User Information, Notification

## Potential Operations

* Process payment
* Decline payment
* Display message
* Prompt user

# Use Case Scenario: Start

This use case begins when the app is launched in a device. It instantiates the model, reads the csv to initialize the theater/movie data. It launches the graphical user interface and builds its elements. The use case ends when the app is successfully launched and user is landed onto the main GUI page.

## Good-Candidate Objects

App, Model, CSV, Data, GUI, Landing Page

## Potential Operations

* Launch
* Instantiate Mode
* Read csv
* Initialize data
* Launch GUI

# Use Case Scenario: Login / Logout

## This use case will start when the start use case has finished. Both users will be able to see the entire application. Regular users are free together choose to start the Register User use case or continue as a regular user and select any other use cases they want. Registered users would have the option to login to their account using a username and password that is assigned to them upon completing the Register User use case for the first time. Once logged in they may continue to use any use cases they wish.

## Good-Candidate Objects

## User, Regular User, Registered User, Login, Account, Username, Password

## Potential Operations

* Login
* Logout
* Validate password
* Accept login
* Reject login

**List of all nouns**

Regular User

Registered User

Theater List

Theater

Movie

Ticket

Receipt

Page //GUI

Showtime

Seat

Map

Button

Timestamp

Message //GUI

Input

Payment

Registration

Login

Reservation

Movie-start

Cancelation

Voucher (code and amount)