Contents

[I.Introduction: 3](#_Toc501183385)

[II.Use case and user story: 4](#_Toc501183386)

[1.Use case 1: 4](#_Toc501183387)

[2.Use case 2: 5](#_Toc501183388)

[3. Use case 3: 6](#_Toc501183389)

[4.Use case 4: 7](#_Toc501183390)

[5.Use case 5: 8](#_Toc501183391)

[6.Use case 6: 9](#_Toc501183392)

[7.Use case 7: 10](#_Toc501183393)

[8.Use case 8: 11](#_Toc501183394)

[9.Use case 9: 12](#_Toc501183395)

[10. Use case 10: 13](#_Toc501183396)

[11. Use case 11: 14](#_Toc501183397)

[12. Use case 12: 15](#_Toc501183398)

[13. Use case 13: 16](#_Toc501183399)

[14. Use case 14: 17](#_Toc501183400)

[15.Use case 15: 18](#_Toc501183401)

[16. Use case 16: 19](#_Toc501183402)

[17. Use case 17: 20](#_Toc501183403)

[III. Functional requirements: 21](#_Toc501183404)

[**Use case 1:** Search a movie 21](#_Toc501183405)

[**Use case 2:** Register 21](#_Toc501183406)

[**Use case 3:** Log in 21](#_Toc501183407)

[**Use case 4:** Log out 22](#_Toc501183408)

[**Use case 5:** Rate movie 22](#_Toc501183409)

[**Use case 6:** View a movie list 22](#_Toc501183410)

[**Use case 7:** View a movie detail 23](#_Toc501183411)

[**Use case 8:** Add movie to favorite 23](#_Toc501183412)

[**Use case 9:** View a movie favorite 23](#_Toc501183413)

[**Use case 10:** Delete a movie favorite 23](#_Toc501183414)

[**Use case 11:** Add account. 24](#_Toc501183415)

[**Use case 12:** Remove account. 24](#_Toc501183416)

[**Use case 13:** Add movie. 24](#_Toc501183417)

[**Use case 14:** Remove movie. 25](#_Toc501183418)

[**Use case 15:** Edit profile. 25](#_Toc501183419)

[**Use case 16:** Edit movie details. 25](#_Toc501183420)

[**Use case 17:** Play movie. 26](#_Toc501183421)

[IV. Non-functional requirements: 27](#_Toc501183422)

# I.Introduction:

Nowadays, movie and TV series are essential and irreplaceable parts of the people’s everyday entertainment. With the rapid development of the entertainment industry, there are various ways of watching movie and TV series. One of the fastest developing methods is video streaming with plenty of services such as Netflix, Amazon Prime, phimmoi and so on. But the problem is that some charge a very high subscription fee, some don’t but are embedded with annoying ads. With this in mind, we have decided to develop our own movie streaming service which is free, fast and convenient for users.

Our movies watching website have a database of movies information, movies categorized by genre and each of them has a unique name and id to easily access.

The system allows all guests view and search movies by their name, id and genre. If guest wants to have premium features, they could register account.

Each user will have a profile page containing age, avatar, favorite movies, genre and movies they watched. They can edit their information at any time they want. The biggest differences between users and guests are the ability to watch the most updated movies and comments on the movies.

In terms of the administrator, he/ she is able to upload the new movies, delete the obsolete movies, edit movies information, ban the account which has the inappropriate comments

# II.Use case and user story:

## 1.Use case 1:

**Name: Search Movies**

Identifier: UC1

Inputs:

* Movie name

Outputs:

* A page with the movies with name most similar to the searches
* A page that shows fail search (no result)

Precondition:

* None

Basic course:

|  |  |
| --- | --- |
| Actors: Registered user, admin, guest | System |
| 1. Enter name and chooses category |  |
| 1. Submit | * 1. Check database for user’s searches   2. If found, return all results for the result page   3. If not, return a page with fail result |

Post condition:

* None

## 2.Use case 2:

**Name: Register**

Identifier: UC2

Inputs:

* Username
* Password
* Email
* Date of birth

Outputs:

* A page with that shows successful user registration
* The register page again to ask for correct input if fail

Precondition:

* None

Basic course

|  |  |
| --- | --- |
| Actors: guest | System |
| 1. Request register page | 1.1 Show page |
| 1. Fill information |  |
| 1. Submit | * 1. Check if the information is suitable   2. If success return success page   3. If fail return register page again with added information |

Post condition:

* Database has been uploaded.

## 3. Use case 3:

**Name: Log in**

Identifier: UC3

Inputs:

* User Name
* Password

Outputs:

* The home page with user’s authentication [if success]
* The login page [if fail]

Basic Course

|  |  |
| --- | --- |
| Actor: User(User/ Administrator) | System |
| 1. Open the login page | * 1. Display the login page |
| 1. Enter user name and password |  |
| 1. Submit | * 1. Check user’s info.   2. If success, return the home page   3. If fail, the login page returns |

Precondition

* User has an account which was registered before

Post condition: None

## 4.Use case 4:

**Name: Log out**

Identifier: UC4

Inputs:

* The home page with authentication

Outputs:

* The home page without authentication

Basic Course

|  |  |
| --- | --- |
| Actor: User(User/ Administrator) | System |
| * + - 1. Click logout | * 1. Return home page |

Precondition

* User is in home page with their authentication

Post condition: None

## 5.Use case 5:

**Name: Rate a movie**

Identifier: UC5

Input:

* Number of star from 1(worst) to 5(best) by clicking star in vote section

Output:

* New voting instance based on (movie, user) in database
* New order of trendy movie in home page

Basic course:

|  |  |
| --- | --- |
| Actor: User(User) | System |
| 1. Choose the movie | * 1. Display the page with movie details |
| 1. Click the play button | * 1. Count the time user is watching the movie |
| 1. Choose number of star | * 1. If the time is greater than a third of movie duration, enable vote section   2. Otherwise, disable vote section |
|  | 4.1. Update the voting instance in database |

Precondition:

* User has an account which was registered before

Post condition: None

## 6.Use case 6:

**Name: View Movie List**

Identifier: UC6

Input: Movie category

Output: The movie list

Basic Course:

|  |  |
| --- | --- |
| Actor: Registered and Unregistered User | System: |
| 1. Click one specific Category | 1.1 Display movies (title and image) in the same Category |

Preconditions:

* User are viewing movie information

Post conditions: None

## 7.Use case 7:

**Name: View Movie Detail**

Identifier: UC7

Inputs:

* Specific movie

Outputs:

* A page with movie’s introduction, length, actor, director, etc

Precondition:

* There must be some movie at the screen, maybe from search method or view movie list

Basic course:

|  |  |
| --- | --- |
| Actors: Registered user, admin, guest | System |
| * + - 1. Click on the movie on the page | 1.1 Display all buttons |
| * + - 1. Click on the “Detail” button | * 1. Display all the information of the movie |

Post condition: None

## 8.Use case 8:

**Name: Add the movie to the favorite**

Identifier: UC8

Inputs:

* Specific movie

Outputs:

* The notification of successful adding the movie to favorite

Precondition:

* There must be some movie at the screen, maybe from search method or view movie list
* The user must be login with their username, password

Basic course:

|  |  |
| --- | --- |
| Actors: Registered user, admin | System |
| * + - 1. Click on the movie on the page | 1.1 Display all buttons |
| * + - 1. Click on “Add favorite” button | * 1. Add the movie into the database of favorite   2. The icon of favorite movie will be display at the right-side of the movie   3. Show the success notification |

Post condition:

* Add the movie into database

## 9.Use case 9:

**Name: View the favorite**

Identifier: UC9

Inputs:

Outputs:

* A page with your favorite movie

Precondition:

* The user must be login with their username, password

Basic course:

|  |  |
| --- | --- |
| Actors: Registered user, admin | System |
| Click on the “Favorite” in the menu | * 1. Display all the favorite movie |

Post condition:

* If there is no movie in the Favorite. There is a message “You should add films to the favorite”

## 10. Use case 10:

**Name: Remove favorite**

Identifier: UC10

Inputs:

* Movie want to be removed

Outputs:

* The notification of successful removing the movies from Favorite

Precondition:

* The user must be login with their username, password
* There must be some films in the Favorite
* Already go to the Favorite page

Basic course:

|  |  |
| --- | --- |
| Actors: Registered user, admin | System |
| 1. Click on the movie | * 1. Display all buttons |
| 2. Click on “Remove from favorite” button | 2.1. Remove the movie from the database of favorite  2.2 The icon of favorite movie will be disappear at the right-side of the movie  2.3 Return the movie page |

Post condition:

* The movie is remove from the database

## 11. Use case 11:

**Name: Add accounts**

Identifier: UC11

Inputs:

Outputs:

* The notification of successfully adding the account

Precondition:

* The users must be login as administrator

Basic course:

|  |  |
| --- | --- |
| Actors: Admin | System |
| 1. Go to the “Request” tab | * 1. Display all the request   2. Display the button “Accept” and “Ignore” next to the request |
| 2. Click on “Accept” button | * 1. The account is add to the database   2. Show the success notification |

Post condition:

* If the admin clicks to the “Ignore” button the request is removed and the registration of the user is unsuccessful

## 12. Use case 12:

**Name: Remove accounts**

Identifier: UC12

Inputs:

Outputs:

* The notification of successfully removing the account

Precondition:

* The users must be login as administrator
* There must be some accounts in the page

Basic course:

|  |  |
| --- | --- |
| Actors: Admin | System |
| 1. Click on the “Accounts” tab | * 1. Display the “Account” page   2. Display all the accounts   3. Display some buttons which is next to the accounts |
| 2. Click on “Remove” button | * 1. The account is remove from the database   2. Show the success notification   3. Return the “Account” page |

Post condition:

* The account is remove from the database

## 13. Use case 13:

**Name: Add movies**

Identifier: UC13

Inputs:

* All the information of the movie
* The links to the movies

Outputs:

* The notification of successfully adding movies

Precondition:

* The users must be login as administrator

Basic course:

|  |  |
| --- | --- |
| Actors: Admin | System |
| 1. Click on the “Movie” tab | * 1. Display the “Movie” page   2. Display all movies   3. Display some buttons “Add new” “Remove”… |
| 2. Click on “Add new” button | * 1. Open the “Input information” page |
| 3. Input all of information |  |
| 4. Click “Submit” | 4.1 Update all information of the movie into database  4.2 The notification of successfully adding the movies |

Post condition:

* The information must be checked by the system, if the information is invalid, there is the message to show the error
* The information of the movie is added to the database

## 14. Use case 14:

**Name: Remove movies**

Identifier: UC14

Inputs:

* Movie want to be removed

Outputs:

* The notification of successful remove the movie

Precondition:

* The users must be login as administrator
* There must be some movies in the page

Basic course:

|  |  |
| --- | --- |
| Actors: Admin | System |
| 1. Click to the “Movies” tab | * 1. Display the “Movie” page   2. Display all the movies   1.3 Display some buttons “Add new” “Remove”… |
| 2. Click on “Remove” button | * 1. The movies is remove from the database   2. Show the success notification   3. Return “Movie” page |

Post condition:

* The movie is remove from the database

## 15.Use case 15:

**Name: Edit Profile**

Identifier: UC15

Inputs:

* New profile information

Outputs:

* The notification of successfully edit profile

Precondition:

* The users must be login

Basic course:

|  |  |
| --- | --- |
| Actors: Registered user, admin | System |
| 1. Click to the “Profile” tab | * 1. Display the information of the user |
| 2. Click the “Edit” button | 2.1 Edit tab is displayed |
| 3.Input new information |  |
| 4. Click on “Confirm” button | * 1. The information is updated to database   2. Show the success notification   3. Return the “Profile” tab |

Post condition:

* The information must be checked by the system, if the information is invalid, there is the message to show the error instead of the “Confirm” message

The accounts’ information is update to the database

## 16. Use case 16:

**Name: Edit Movie details**

Identifier: UC16

Inputs:

* New moview details

Outputs:

* The notification of successfully edit movie details

Precondition:

* The users must be login as administrator
* There must be some movies in the page

Basic course:

|  |  |
| --- | --- |
| Actors: Admin | System |
| 1. Click to the “Movies” tab | * 1. Display the “Movie” page   2. Display all the movie   1.3 Display some buttons “Add new” “Remove”… |
| 2. Click on “Edit movie” button | * 1. Display the information of the movies |
| 3. Edit the information |  |
| 4. Click on “Confirm” button | * 1. The movies’ information is updated to the database   2. Show the success notification   3. Return “Movie” page |

Post condition:

* The information must be checked by the system, if the information is invalid, there is the message to show the error instead of the “Confirm” message, and return back to the page to edit the information
* The accounts’ information is update to the database

## 17. Use case 17:

**Name: Play Movie**

Identifier: UC17

Inputs:

Outputs:

* The movie

Precondition:

* The guest logged in.

Basic Course:

|  |  |
| --- | --- |
| Actor: User | System |
| Click the play button | * 1. Play the movie in default server |
| 1. Choose the server | * 1. Play the movie in requested server |

Post condition: None

# III. Functional requirements:

## **Use case 1:** Search a movie

**1. Functional requirements**

- Shall display the search section.

- Shall use auto-complete to suggest keywords with results.

- Shall sort the problems that match from the database.

- Shall display the search result .

**2. Data requirements**

None

## **Use case 2:** Register

**1. Functional requirements**

- Shall display the page to require account information from the user.

- Shall check the validity of account ID.

- Shall insert a new account of user into database.

- Shall send an email to the registered user with his account details.

- Shall encrypt the password before inserting it into the database.

- Shall display the announcing page the success of registration process.

**2. Data requirements**

Valid account information (ID is not used, email is valid).

## **Use case 3:** Log in

**1. Functional requirements**

- Shall display the log in so that user can input the information of account.

- Shall retrieve the information of the account that is matched with the ID that user logs in from the

database.

- Shall decrypt password of user account from the database.

- Shall check the validity of user login account.

**2. Data requirements**

The log in information (ID and password) must be valid.

## **Use case 4:** Log out

**1. Functional requirements**

* Shall display with user authentication and logout button so user

can click to log out

**2. Data requirements**

None

## **Use case 5:** Rate movie

**1. Functional requirements**

* Shall display with user authentication and movie detail section
* Shall allow the user to rate the movie

**2. Data requirements**

None

## **Use case 6:** View a movie list

**1. Functional requirements**

- Shall display list of movie based on the keyword and category

- Shall “next page” button when the list is too long.

**2. Data requirements**

Data is taken from the database.

## **Use case 7:** View a movie detail

**1. Functional requirements**

-Shall have a movie description with the following information: introduction, length, actors, etc…

-Shall have “view movie” button.

-Shall have “add to favorite” button when the guest already logged in.

**2. Data requirements**

Data is taken from the database.

## **Use case 8:** Add movie to favorite

**1. Functional requirements**

-Shall have “add to favorite” when the guest already logged in.

**2. Data requirements**

Data is taken from the database.

## **Use case 9:** View a movie favorite

**1. Functional requirements**

-Shall have a movie favorite list

-Shall have “add/delete favorite list” button..

**2. Data requirements**

Data is taken from the database.

## **Use case 10:** Delete a movie favorite

**1. Functional requirements**

-Shall have a movie favorite list

-Shall have “add/delete favorite list” button.

-Shall have “confirm delete” message prompt.

**2. Data requirements**

Data is taken from the database.

## **Use case 11:** Add account.

**1. Functional requirements**

-Shall have administrator control panel.

-Shall have “request” tab contains all account request.

-Shall have “accept” and “ignore” button.

**2. Data requirements**

Data is taken from the database.

## **Use case 12:** Remove account.

**1. Functional requirements**

-Shall have administrator control panel.

-Shall have “account” tab contains all account information.

-Shall have “remove account” button.

-Shall have “confirm/cancel remove account” button.

**2. Data requirements**

Data is taken from the database.

## **Use case 13:** Add movie.

**1. Functional requirements**

-Shall have administrator control panel.

-Shall have “movie” page contains all movie information.

-Shall have “add new”, ”remove movie” button.

-Shall have “input information” page to input the movie details.

-Shall have “submit” button.

**2. Data requirements**

Data is taken from the database.

## **Use case 14:** Remove movie.

**1. Functional requirements**

-Shall have administrator control panel.

-Shall have “movie” page contains all movie information.

-Shall have “add new”, ”remove movie” button.

-Shall have “confirm/cancel remove” button

**2. Data requirements**

Data is taken from the database.

## **Use case 15:** Edit profile.

**1. Functional requirements**

-Shall have “profile” tab contains all user information.

-Shall have “edit profile” button.

-Shall have “submit/cancel” button.

**2. Data requirements**

Data is taken from the database.

## **Use case 16:** Edit movie details.

**1. Functional requirements**

-Shall have administrator control panel.

-Shall have “movie” page contains all movie information.

-Shall have “add new”, ”remove movie” button.

-Shall have “confirm/cancel edit” button

**2. Data requirements**

Data is taken from the database.

## **Use case 17:** Play movie.

**1. Functional requirements**

-Shall have “play movie” in movie details page.

-Shall “play” button.

-Shall server list to choose.

**2. Data requirements**

Data is taken from the database.

# IV. Non-functional requirements:

1. **Operational requirements**  
• All of the components must function and work properly with one another.  
• The server side must always be online so that users/guests can use all components. This is an important requirement that must be met  
• The system will have an administrator, who can maintain the site and resolve hardware and software issues and maintenance.

2. **Legal requirements**  
• Use open-source technology to build the system if possible, and if proprietary software is used, it should be clearly quoted so to avoid copyright violation and to help the customer in the calculation of maintenance costs.  
• Must secure the code to minimize the risk of hacks

3. **Legal requirements**  
• Use open-source technology to build the system if possible, and if proprietary software is used, it should be clearly quoted so to avoid copyright violation and to help the customer in the calculation of maintenance costs.  
• Must secure the code to minimize the risk of hacks

4. **Performance requirements  
4.1 System Response and throughput Requirements**  
•The time it takes to execute queries should be fast.  
•Time to execute functionalities of the system should be quick. Generally, the response time for the webpage should be fast.  
• All the items on the web page must be generated correctly.  
**4.2 Availability Requirements** (is the system available when is requested by the user)  
•System should have good memory management

5. **Maintainability requirements**  
• Offer the functions that help user to update their profile.  
• Write well-defined and clean source code with necessary comment lines like appropriate naming of variables.  
• Design a proper database that is convenient for the further maintenance

6. **Support requirements**  
• Provide a remote maintenance tool (like TeamViewer) to support the administrator if they encounter errors that they cannot resolve

7. **Security requirements**  
• The website should protect the personal information and account details of the users on the system from hacking attempts.   
• All communications between the system’s data server and clients must be encrypted (SSL, SSH protocol for HTTP, etc.)   
• All system data should be backed up and there should be one copy in an external hard drive.

8. **Interface requirements**  
•The interface should be consistent across different browsers

• Buttons should include feature symbols equivalent to their functionalities