Object Oriented Analysis and Design  
(INT3110 21)

*Course Project*

*Netflix.com Website*

**Instructor:** Assoc. Prof. Dr. Trương Ninh Thuận

**Project team:**

|  |  |  |
| --- | --- | --- |
| Trần Quang Vinh | K62-CACLC3 | Student No. 17021357 |
| Phạm Thái Sơn | Student No. 17021330 |

Table of Contents

[1 Requirements 3](#_Toc22131738)

[1.1 Problem statement 3](#_Toc22131739)

[1.1.1 Addressing the problem 3](#_Toc22131740)

[1.1.2 Solution 3](#_Toc22131741)

[1.2 Glossary 4](#_Toc22131742)

[1.3 Supplementary specifications 5](#_Toc22131743)

[1.4 Use-case models 7](#_Toc22131744)

[1.4.1 Create account 8](#_Toc22131745)

[1.4.2 Log in 9](#_Toc22131746)

[1.4.3 Upload movie 10](#_Toc22131747)

[1.4.4 Delete movie 12](#_Toc22131748)

[1.4.5 Edit movie’s information 13](#_Toc22131749)

[1.4.6 Watch movies 14](#_Toc22131750)

[1.4.7 Update account’s information 15](#_Toc22131751)

# Requirements

## Problem statement

### Addressing the problem

As quality of life standard increases, therefore, the need for entertainment increases. At the same time, people are more occupied than ever before. People wants a less time-consuming way to watch movies without ever standing in a queue or having to go to a store to purchase them. This evokes the need for a system for a service to help accommodate such needs. Users can watch a movie they want without the need of going out to a physical store or waiting at a queue in a movie theater. Movies distributors can distribute their movies on the system so that users can watch them. This has come to a demand for a system to solve this problem.

### Solution

Netflix.com is built as an online entertainment platform so that users can watch the movie directly on the website. Netflix is a streaming service that offers a wide variety of award-winning TV shows, movies, anime, documentaries, and more on thousands of internet-connected devices. You can watch as much as you want, whenever you want without a single commercial – all for one low monthly price. There's always something new to discover and new TV shows and movies are added every week!

**Description**

The system will be developed as a web application. End users will interact with the system over the Internet via a wide range of devices (smartphones, PCs, smart TV…).

People can register for an account then log into the system and buy a monthly subscription to start watching movies.

Users can search for movies and TV shows they are interested in and watch it. They can also view the information of a movie such as synopsis, trailers, etc. Moreover, they can add a movie to their watch list to watch later.

Content providers can upload movies and TV shows, edit their information directly on the system.

**End users**

The system’s end users are as follow:

* Users

Users are visitors registered to become registered users. They can buy subscription to watch movies and TV shows. Users can view information about the movies and add them to their watch list.

* Content providers

Content providers are responsible for managing their movies and TV shows on the system. They can upload movies, edit their information. They can also delete movies from a system if the contract expires.

## Glossary

**Introduction**

This document is used to define terminology specific to the problem domain, explaining terms, which maybe unfamiliar to the reader of the use-case descriptions or other project documents. Often, this document can be used as an informal data dictionary, capturing data definitions so that use-case descriptions and other project documents can focus on what the system must do with the information.

**Definitions**

The glossary contains the working definitions for the key concepts in the Netflix.com website.

**Account**

A record about a user/administrator containing information about his/her username, e-mail, password, subscription and payment information. Each account has a unique user ID, which are used to identify the user/administrator and grant them access to specific parts of the system.

**Content provider**

An organization whose job is to manage its movies and TV shows on the system. They are responsible for uploading movies, adding or editing movies’ information and delete the movies from the system when the contracts are expired.

**Movie**

A movie made by movie studios and published by a distributor.

**Subscription**

An amount of money that users pay regularly to be able to watch movies on Netflix.com website.

**TV Show**

A TV series consists of multiple episodes. Multiple episodes can be grouped into a season. A TV series may consist of one or many seasons.

**User**

Any person who has a registered account on the website but is not an administrator. Users can buy or cancel subscriptions, watch movies and view their information.

**Visitor**

Any person who visits the website without having an account. Visitors can create account to become users.

## Supplementary specifications

**Objectives**

The purpose of this document is to define requirements of the Netflix.com system. This Supplementary Specification lists the requirements that are not rapidly captured in the use case of the use-case model. The Supplementary Specification and the use-case model together capture a complete set of requirements on the system.

**Scope**

This Supplementary Specification applies to the Netflix.com system, which is a streaming website in the United States.

This specification defines the non-functional requirements of the system: such as reliability, usability, performance and supportability, as well as functional requirements that are common across a number of use cases. (The functional requirements are defined in the Use Case Specification.)

**References**

None.

**Functionality**

Multiple users must be able to perform their work concurrently.

**Usability**

The software must be easy to use so that a new user can learn how to use the system within 30 minutes.

The user interface has to be friendly and intuitive.

**Reliability**

The system must be available 24 hours a day, 7 days a week. The system must also have less than 5% downtime.

**Performance**

The system shall handle up to 10 terabits per second of simultaneous peak traffic.

The system shall provide access to the database with no more than 5 seconds latency.

The system must be able to complete at least 95% of all transactions within 15 seconds.

**Supportability**

None.

**Security**

The system must prevent users from logging in if they do not provide a correct password.

A movie or TV show can only be edited and deleted by the content provider that uploaded it.

Only content providers can upload movies to the server.

**Design Constraints**

The system must provide a responsive web-based interface that is usable on multiple devices, such as computer, smartphone and smart TV.

## Use-case models

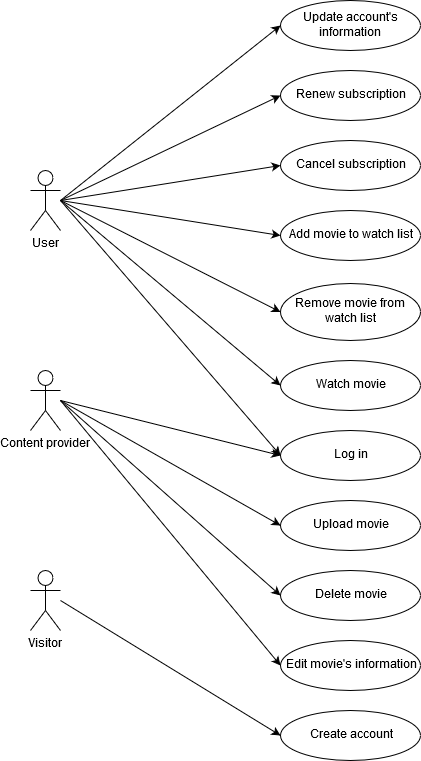


Figure 1‑1. Use-case model for the Netflix.com website system

### Create account

**Brief Description**

This use case describes how a visitor registers for an account on the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the visitor requests to create an account on the website.

1. Visitor clicks the “Try 30 days free” button.
2. The system displays a page that show basic information about plans.
3. Visitor clicks “See the plans” button.
4. The system shows more detailed information, features about each plan.
5. Visitor clicks “Continue” button.
6. The system prompt visitor to enter the following information:

* Email address (required)
* Password (required)
* Email his/her Netflix special offers (optional)

1. Visitor provides the information then press continue.
2. The system check if email address already exists in database and all required fields are filled. It then adds a new account with the specified information to the user database.
3. The system lists available payment method then asks visitor to choose his/her preferred payment methods.
4. Visitor choose a payment method, enter his/her name (required) and billing information (required), then select “Start membership”.
5. The system verifies billing information, then save the specified information to the previously added account.

*Alternative Flows*

**Missing Information**

If any of required fields are not filled in, the system will automatically display an error. The visitor can make changes to continue the registration or cancel it, which end the use case.

**Email address already exists**

If visitor enters an email address already existed in database, the system displays an error message and asks the user to choose another one. The visitor can choose another email address to continue the registration or cancel it, which end the use case.

**Failed to verify billing information**

If the payment processor cannot verify the billing information, the system displays an error message and prompts the user to choose another billing method or billing information. The visitor can choose another billing method or billing information to continue the registration or cancel it, which end the use case.

**Special requirements**

None.

**Pre-Conditions**

None.

**Post-Conditions**

If the operation is successful, a new user is added to the system. Otherwise, the system state remains unchanged.

**Extension Points**

None.

### Log in

**Brief Description**

This use case describes how a user or a content provider logs into the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor requests to log into the website.

1. Actor clicks the “Sign in” button.
2. The system displays a page that prompts actor for email address and password.
3. Actor enters email address and password and, optionally, tick the “Remember me” checkbox then press “Sign in” button.
4. The system validates the entered information then logs the actor into the system.

*Alternative Flows*

**Missing Information**

If any of required fields are not filled in, the system will automatically display an error. The actor can make changes to continue to log in or cancel operation, which end the use case.

**Email address doesn’t exist in database**

If actor enters an email address doesn’t exist in database, the system displays an error message and asks the actor if he/she want to register a new account. The actor can choose another email address to continue to log in, create a new account or cancel the operation, which end the use case.

**Wrong password**

If the actor entered wrong password, the system displays an error message and asks the actor if he/she want to recover password. The actor can choose to recover password, retry another password or cancel the operation, which end the use case.

**Special requirements**

None.

**Pre-Conditions**

The system is in the login state and has the login screen displayed.

**Post-Conditions**

If the operation is successful, the actor is now logged into the system. Otherwise, the system state remains unchanged.

**Extension Points**

None.

### Upload movie

**Brief Description**

This use case describes how a content provider uploads a movie into the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the content provider requests to upload a movie to the system.

1. Content provider opens “Upload” page.
2. The system displays a page with a drag-and-drop box to upload the movie.
3. Content provider choose the movie he/she wanted to upload and press “Upload” button.
4. The system starts uploading the movie to the server and prompts the content provider for the movie’s information (movie’s name, genre, released year, actor, …).
5. The content provider enters movie’s information.
6. When the upload process is completed, the system let the content provider to preview the movie for any error and asks the provider to confirm the upload.
7. Content provider confirms the upload.
8. The system creates a new record in the database containing information about that movie and its location on the server for later use.

*Alternative Flows*

**Missing Information**

If any of required fields are not filled in, the system will automatically display an error. The content provider can make changes to continue or cancel operation, which end the use case.

**Content provider rejects the uploaded movie**

If the actor rejects the movie that he/she uploaded, the system deletes the movie from the system then goes back to step 1. Actor can choose to continue the upload or cancel the operation, which end the use case.

**Special requirements**

None.

**Pre-Conditions**

The content provider must be logged to the system.

**Post-Conditions**

If the operation is successful, the movie will be available for the users. Otherwise, the system state remains unchanged.

**Extension Points**

None.

### Delete movie

**Brief Description**

This use case describes how a content provider deletes a movie on the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the content provider requests to delete a movie on the system.

1. Content provider opens “Manage” page.
2. The system displays a list of movies uploaded by the content provider.
3. Content provider choose the movie he/she wished to delete and press “Delete” button.
4. The system displays a message asks if he/she really wants to delete the movie.
5. The content provider chooses “Yes” to confirm the delete.
6. The system deletes the movie’s record from database and movie file from the server then displays a message notify the user that the operation is complete.

*Alternative Flows*

**Content provider denied when asked to confirm the delete**

If the actor rejects the delete, the system goes back to step 2. Actor can choose to continue the process or cancel the operation, which end the use case.

**List of movies uploaded by the content provider is empty (Content provider hasn’t uploaded any movies to the system)**

If the list of movies uploaded by the content provider is empty, the “Delete” button is disabled and the use case is ended.

**Special requirements**

None.

**Pre-Conditions**

The content provider must be logged to the system.

**Post-Conditions**

If the operation is successful, the movie will no longer be available for the users. Otherwise, the system state remains unchanged.

**Extension Points**

None.

### Edit movie’s information

**Brief Description**

This use case describes how a content provider edit information of a movie on the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the content provider requests to edit information of a movie on the system.

1. Content provider opens “Manage” page.
2. The system displays a list of movies uploaded by the content provider.
3. Content provider choose the movie he/she wished to edit information and press “Edit” button.
4. The system displays all information fields with current information filled each field.
5. Actor edits information as needed then click “Save”.
6. The system displays a message asks the actor to confirm the changes.
7. The content provider chooses “Yes” to confirm the changes.
8. The system edits the movie’s record on database using the new information.

*Alternative Flows*

**Content provider denied when asked to confirm the changes**

If the actor rejects the changes, the system goes back to step 2. Actor can choose to continue the process or cancel the operation, which end the use case.

**List of movies uploaded by the content provider is empty (Content provider hasn’t uploaded any movies to the system)**

If the list of movies uploaded by the content provider is empty, the “Edit” button is disabled and the use case is ended.

**Special requirements**

None.

**Pre-Conditions**

The content provider must be logged to the system.

**Post-Conditions**

If the operation is successful, the movie’s new information will be available for the users. Otherwise, the system state remains unchanged.

**Extension Points**

None.

### Watch movies

**Brief Description**

This use case describes how a user watch a movie on the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the user requests to watch a movie on the system.

1. User searches for the movie he/she wants to watch by using search box and enters its name.
2. The system displays a list of movies matches the given name.
3. User choose the movie he/she wished to watch from the list.
4. The system displays all information about that movie, a “Play” and a “My list” button.
5. User press “Play” button.
6. The system starts streaming the movie to the user.

*Alternative Flows*

**No movies are found with the given name**

If the system cannot find any movies matches the given name, it prompts the user to enter a different name. User can continue making change to the search box or cancel the operation, which end the use case.

**Special requirements**

None.

**Pre-Conditions**

The user must be logged to the system and the account must have an active subscription.

**Post-Conditions**

If the operation is successful, the movie is streamed to user’s display.

**Extension Points**

None.

### Update account’s information

**Brief Description**

This use case describes how a user updates his/her account’s information on the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the user requests to edit account information on the system.

1. User clicks “Account” button.
2. The system displays account’s information.
3. User makes changes as needed. After that, user presses “Save” button to save all changes.
4. The system verifies the changes then updates the changes to the database.

*Alternative Flows*

**Verification detects invalid information**

If the system detects any invalid changes, it will display an error message and notify the user about those invalid changes. User can continue to make changes or cancel the operation, which end the use case.

**Special requirements**

None.

**Pre-Conditions**

The user must be logged to the system.

**Post-Conditions**

If the operation is successful, the new account information is displayed to user.

**Extension Points**

None.

### Renew subscription

**Brief Description**

This use case describes how a user renew his/her subscription on the Netflix.com website.

**Flow of Events**

*Basic Flow*

This use case starts when the user requests to renew account subscription on the system.

1. User clicks “Account” button.
2. The system displays account’s information and a “Restart membership” button.
3. User makes changes as needed. After that, user presses “Save” button to save all changes.
4. The system verifies the changes then updates the changes to the database.

*Alternative Flows*

**Verification detects invalid information**

If the system detects any invalid changes, it will display an error message and notify the user about those invalid changes. User can continue to make changes or cancel the operation, which end the use case.

**Special requirements**

None.

**Pre-Conditions**

The user must be logged to the system.

**Post-Conditions**

If the operation is successful, the new account information is displayed to user.

**Extension Points**

None.