Software Requirements Specification

Version 1.0

November 29, 2016

Movie Hunter System

Alexandros Michailidis

Table of Contents

List of Figures

1.0. Introduction

1.1. Purpose

The purpose of this document is to present a detailed description of the Movie Hunter System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react.

1.2. Scope of Project

This software will be a movie / series streaming system. This system will be designed to efficiently provide a set of movies and series in client's video player. The goal of this software is to help users organize their favorite movies and series and also explore new, based on their tastes. By keeping everything in the server's hard drive it will also minimize the required hard drive space on the client's side. The software will meet the user’s needs while remaining easy to understand and use.

More specifically, this system is designed to allow a user to manage and explore new movies and series, read reviews and comments by others, and efficiently stream files to clients.

1.3. Glossary

Term	Definition
User	A user that will, or already accessing the system.
Administrator	A user that is the administrator of the system. Responsible for keeping the system in good status.
Database	Collection of all the information monitored by this system.
External API	An external server that keeps and updates information about movies and series.
Server	An internal server that handles every action in the system.
Movie	An object that represents an actual movie. Including title,category and the real file path of the movie.
Show	An object that represents an actual tv show. Including title,category and other.
Season	An object that represents an actual season of a tv show. Including season number year etc.
Episode	An object that represents an actual episode of a season. Including title,date and the real file path of the episode.

1.4. References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

1.5. Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

2.0. Overall Description

2.1 System Environment

The Movie Hunter System has two active actors and one cooperating system.

The User accesses the Movie Hunter through a program and the Internet. The

Administrator accesses the entire system directly through a website.

2.2 Functional Requirements Specification

This section outlines the use cases for each of the actors separately.

2.2.1 User Use Cases

Use case: **Login**

Brief Description
The User accesses the Movie Hunter program, hit the login button and stalls in a form, requesting credentials.

Initial Step-By-Step Description

1. The User types his email and password.
2. The system request the external api to verify the user's credentials.
3. The external api responds whether the credentials belong to a user or not.
4. The system based on the response decides to redirect the user to main page or ask for credentials again.

Use case: **Register**

Brief Description
The User accesses the Movie Hunter program and hit the register button.

Initial Step-By-Step Description

1. The user redirects to a new registration form handled by the external api.
2. After the user completes the form with his information the external api redirects him back to the program, including a unique identification.
3. The system gets the unique identification and saves the new user in the database.
4. After the successful registration the user redirects to the main page
5. Otherwise the process starts again from step 1.

Use case: **Display Main Page**

Brief Description
The User accesses the Movie Hunter program main page.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user sees a list of popular movies fetched from the external api.
2. He has the option to change this list and see trending movies or popular tv shows or trending tv shows, all fetched from the external api.

Use case: **Search Content**

Brief Description
The User accesses the Movie Hunter program main page and search for a specific movie/series.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user types the title of a movie or TV show in a text-box.
2. The external api fetches the appropriate results.
3. The system displays the results in a list.

Use case: **Display A Movie**

Brief Description
The User clicks a movie from the main page or after a search.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user clicks a movie from the list.
2. The system redirects him in a new page which shows information about the specific movie.
3. The user has the option read the reviews about the movie or start streaming content to his local machine.

Use case: **Display A TV Show**

Brief Description
The User clicks a TV show from the main page or after a search.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user clicks a tv show from the list.
2. The system redirects him in a new page which shows information about the specific TV show and a list of seasons attached to the show.
3. The user has the option read the reviews about the show or click a specific season

Use case: **Display A Season**

Brief Description
The User clicks a TV show from the main page or after a search and a then selects a specific season.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user clicks a season from the list.
2. The system redirects him in a new page which shows information about the specific season and a list of episodes attached to the show.
3. The user has the option read the reviews about the season or click a specific episode.

Use case: **Display An Episode**

Brief Description
The User clicks a tv show from the main page or after a search, selects a specific season and then a specific episode.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user clicks an episode from the list.
2. The system redirects him in a new page which shows information about the specific episode.
3. The user has the option to read the reviews about the episode or start streaming content in his local machine.

Use case: **Streaming**

Brief Description
The User selects to stream a movie or an episode.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user selects to stream content.
2. The server starts to progressively streams a file to the client.
3. The system is responsible to attach the file chunks to a video player and handle pause/resume/skip on the stream.

Use case: **Commenting**

Brief Description
The User reviewing or creating a comment for a tv show or movie.

Initial Step-By-Step Description
This assumes the user has already logged in or registered.

1. The user sees a list of comments associated with a show or movie .
2. The user types a new comment inside a text-box.
3. The system is responsible to send the comment to the external api and store it.

2.2.2 Administrator Use Case

Use case: **Login**

Brief Description
The administrator enters a website requesting credentials to proceed.

Initial Step-By-Step Description

1. The Administrator enters his credentials.
2. The server checks if the credentials match and belongs to an administrator.
3. The Administrator redirects to the main page.

Use case: **Content Management**

Brief Description
The administrator manage a TV show or movie.

Initial Step-By-Step Description

1. The Administrator selects a TV show or movie.
2. The server redirects him to a new page with information about the selected show/movie.
3. The Administrator uploads or change the real file of the episode or movie.
4. The server handles the new file and associates it with the selected episode/movie for future streaming.

2.3 User Characteristics

The user is expected to be able to use a windows/osx/linux program based on the operating system he has. Also to be familiar with search boxes, buttons and be able to understand how to use a video player. Also it is expected that he user has a good Internet connection in order to stream content flawlessly.

2.4 Non-Functional Requirements

The Movie Hunter server will be on a server with high speed Internet capability. The physical machine to be used will be determined in the future. The software developed here assumes the use of a tool for connection between the Web pages and the database. The speed of the user's connection will depend on the hardware used rather than characteristics of this system. The user's program will not expect high end computers and it is expected to work across multiple platforms. (windows / osx / linux)

3.0. Requirements Specification

3.1 External Interface Requirements

The only link to an external system is the link to the Trakt TV API. The Trakt TV API fields of interest to the Movie Hunter System are: users, tv shows, movies and comments.

For the sake of efficiency and speed, the movie hunter database will not store movies, and shows but it will reference them from the external api. It will work like a foreign key which joins the movie hunter database with the Trakt TV api.

3.2 Detailed Non-Functional Requirements

3.2.1 Logical Structure of the Data stored in the **external api** database

Movie Data Entity

Data Item	Type	Description	Comment
ID	Text	Unique id of movie	
Title	Text	Title of movie	
Year	Integer	The release year of movie	
Overview	Text	Overview of the movie	
Genre	Text	The genre of the movie	It could be action,horror,etc..
Rating	Float	Rating of movie based on public opinion	This is calculated based on several movie reviewing sites.
Image	Text	A URL of an image for the movie	

TV Show Data Entity

Data Item	Type	Description	Comment
ID	Text	Unique id of TV show	
Title	Text	Title of TV show	
Year	Integer	The release year of TV show	
Overview	Text	Overview of the show	
Genre	Text	The genre of the show	It could be action,horror,etc..
Rating	Float	Rating of show based on public opinion	This is calculated based on several movie reviewing sites.
Image	Text	A url of an image for the show	
Seasons	Array of Pointers	An array of pointers which lead to the seasons for this show	

TV Season Data Entity

Data Item	Type	Description	Comment
ID	Text	Unique id of tv season	
Number	Integer	Number of season	It could be season 1, season 2, etc..
Show	Pointer	A pointer which leads to the show this season belongs	
Year	Integer	The release year of season	
Overview	Text	Overview of the season	
Rating	Float	Rating of season based on public opinion	This is calculated based on several movie reviewing sites.
Image	Text	A url of an image for the season	
Episodes	Array of Pointers	An array of pointers which lead to the episodes for this season	

TV Episode Data Entity

Data Item	Type	Description	Comment
ID	Text	Unique id of tv episode	
Number	Integer	Number of episode	It could be episode 1, episode 2, etc..
Season	Pointer	A pointer which leads to the season this episode belongs	
Release Date	Date	The release date of episode	
Overview	Text	Overview of the episode	
Rating	Float	Rating of episode based on public opinion	This is calculated based on several movie reviewing sites.
Image	Text	A url of an image for the episode	

User Data Entity

Data Item	Type	Description	Comment
ID	Integer	Unique id of user	
Email Address	Text	Email address of user	
Name	Text	User name	

3.2.2 Logical Structure of the Data stored in the **Movie Hunter Server database**

Movie Data Entity

Data Item	Type	Description	Comment
ID	Pointer	Unique id of movie	It will point to the actual data stored in the trakt tv api.
File Path	Text	File path of the movie	It will be the real file path on the server, which contains the movie.

TV Episode Data Entity

Data Item	Type	Description	Comment
ID	Pointer	Unique id of episode	It will point to the actual data stored in the trakt tv api.
File Path	Text	File path of the episode	It will be the real file path on the server, which contains the episode.

User Data Entity

Data Item	Type	Description	Comment
ID	Pointer	Unique id of user	It will point to the actual data stored in the trakt tv api.
Role	Text	The role of the user in system	It could be a simple user or administrator.

3.2.3 Security

The server on which the Movie Hunter Server resides will have its own security to prevent unauthorized *write*/read/*delete* access. The client will communicate with the server providing an access token on every request. The access token will be decoded in the server and if and only if the user is allowed to perform the action then it will be executed.

The PC on which the Movie Hunter Client resides will have its own security. Only if the user is logged in will have access to content. Also all the content is stored in the external api which is already secured.