Software Requirements Specification

for

Minerva – Movie Forum and Database

Version 1.0 under review

Prepared by:

Mukul Mehta (18CS10033) – mukul.csiitkgp@gmail.com

IIT Kharagpur

22 January 2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

A forum where people can discuss movie plots, spoilers and recommendations can come handy for movie buffs and critics who want their opinions heard. **Minerva** aims to provide such a platform for free discussion and debate related to movies. Additionally, users can also search for movies similar to the ones they enjoy watching. This document illustrates all features and implementational intricacies of the software.

1.2 Document Conventions

This SRS Document has been written using Free and Open Source writing tools such as LibreOffice typed in a monospace font **Source Code Pro**. The font size used is 11 for text and 13 for headings. All headings are highlighted appropriately in bold. The document is prepared using **UK English** convention.

1.3 Intended Audience and Reading Suggestions

This document lists all technical and non-technical aspects of the software. It is intended to assist developers and other end users to understand the motivation behind the software and understanding implementation intricacies in it. Anybody who wants to use the software can read the appropriate parts of the document, a list of which is given in the Table of Contents on Page 2.

1.4 Product Scope

Minerva is an android/web based application that is intended to be used by individuals who want to talk about anything related to movies and TV Shows (Discussing plots, alternate endings etc). The application would be based on the client-server model where the users can interact with the frontend system that is served by a backend server that also runs recommendation systems and other utilities required by the application.

1.5 References

This document is based on the IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specification given by the IEEE Computer Society in 1998.

2. Overall Description

2.1 Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

2.2 Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

2.3 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

2.4 Operating Environment

The application is based on a client-server model with the users interacting as clients with a GUI web application. The frontend application will be built using Flutter for mobile application and

React.JS for web based application. Since Flutter is a cross-platform framework, the app would be available to users independent of the OS and on all form factors ranging from Android smartphones to Windows PC's.

2.5 Design and Implementation Constraints

2.6 User Documentation

The software will be accompanied with a user manual aimed at allowing end users to study all components of the application and use it as intended. For developers and other technical users, all API's, classes and methods will be documented and presented online.

2.7 Assumptions and Dependencies

3. External Interface Requirements

3.1 User Interfaces

The product will consist of a mobile/web application with which the user will interact. The user will have the option to search for movies, create a new post regarding a particular movie, comment on other posts, like & dislike posts and create a bucket list on their profile which would be public for other users to view.

3.2 Hardware Interfaces

The backend server will require a decent processing unit to enable multiple threads for worker processes. The appropriate server will be purchased on Digital Ocean or set up on a machine inside the campus (To prevent complications due to existing network proxy). There is no specific hardware requirements for the user under the assumption that he/she has access to a decent internet connectivity via a decent mobile or modern web browser.

3.3 Software Interfaces

The application will interact with the Database (SQL server) using an ORM. The database will be secured with a username-password combination. To improve quality of software, there will be two databases - for local testing and a production database for deployment. The application also interacts with several API's for fetching movie information and suggestions. All API’s will be called with appropriate headers and parameters.

3.4 Communications Interfaces

4. System Features

4.1 User Authentication - Signup & Login

4.1.1 Description and Priority

As the first step to using **Minerva**, a user has to create an account by signing up on the application. This task is essential as it provides the user class needed for all other activities.

4.1.2 Stimulus/Response Sequences

User has to fill a form consisting of basic details such as Name, E-Mail, Username and Password to get started.

As soon as a new user signs up, the system will send a

verification mail to ensure that no malicious users enter the system.

4.1.3 Functional Requirements

Signup: As soon as the user enters information and the authenticity of the user is confirmed via email, the new user is created in the database. Then the user can add more information to the profile such as genres interested in and movies watched.

Login: Existing users can login with their authentication information once the system deems it to be correct.

4.2 Searching for Movies

4.2.1 Description and Priority

The user enters the name of a movie to view details such as cast, awards won, trivia etc. The data will be fetched using a third party API such as IMDB API. The user can also search for movies similar to the given movie using natural language queries. As an example, Batman Begins and The Dark Knight are related although the names are very different.

4.2.2 Stimulus/Response Sequences

User searches the database and the relevant backend API's return useful results to the query.

4.2.3 Functional Requirements

Search: The user can search the database and add filters such as language, country of origin, year of release etc. The database is queried appropriately and results are returned.

4.3 User Profile

4.3.1 Description and Priority

The user can create a profile consisting of movies that they have watched, items on their bucket list, a bio describing the genres they enjoy and some interesting trivia.

4.3.2 Stimulus/Response Sequences

User can edit his/her bio with a GUI tool that calls the relevant functions in the backend.

4.3.3 Functional Requirements

Edit Profile: The user can add relevant details to their profile. They can upload fan art such as posters and add a profile picture to their page.

4.4 Create Post

4.4.1 Description and Priority

The user can create a post for a movie to start a discussion on the topic. When the user is creating a post, the system will add movie info the post automatically by fetching it from the database and provide links to the movie page.

4.4.2 Stimulus/Response Sequences

User can create a post by adding text content. Later, the feature to add images will be added to the system.

4.4.3 Functional Requirements

Create Post: The user can create a discussion thread for a particular movie and related topics.
Suggest Threads: The system will show threads and posts similar to the user's post to allow the user to read more about the line of thought they want to pursue.
Edit Post: The user can also edit a post that they have previously created and the system should show the latest time at which the edit has been done.

4.5 Comments and Replies

4.5.1 Description and Priority

Once a post has been created, a user can also post replies to the post. The comments will be nested (as in Reddit) and users can reply to replies as well.

4.5.2 Stimulus/Response Sequences

User can add a comment on the post as soon as it is created. These comments will be publicly viewable and other users can visit user profiles of the comment creator.

4.5.3 Functional Requirements

Create comment: The user can add comments and the system will create an appropriate entry in the database with information such as when the comment was made and the user who created it.
Edit comment: A user can edit a comment made previously.

4.6 Post & Comment Ranking System

4.6.1 Description and Priority

The user can upvote and downvote posts and comments. The number of such reactions to a post or comment will be viewable by all users and they will be ranked by these numbers.

4.6.2 Stimulus/Response Sequences

User can upvote or downvote a post only once according to his/her liking.

4.6.3 Functional Requirements

React to post: The user can click on upvote or downvote and the post will receive a reaction. The user can also change his/her reaction to the post.

Rank posts/comments: The order in which posts and comments are displayed will depend on the number of likes/dislikes it receives. The comments which are heavily appreciated will be displayed on the top.

4.7 Bucket List & Recommendation System

4.7.1 Description and Priority

The user can add movies and TV-Shows to their bucket list which will be visible to all other users of the application. The system will suggest movies that might be similar to the user’s liking and allow the user to explore that genre.

4.7.2 Stimulus/Response Sequences

User adds a movie to his/her bucket list and once the user has added enough movies (5 or so), the system can start recommending movies.

4.7.3 Functional Requirements

Add item: Once the user adds an item to his/her bucket list, it will be added to the database and the system will add it to the dataset that is powering a recommendation system in the backend.

Recommendation System: TBD, will suggest movies similar to the movies a user has already seen.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The user should be able to query databases quickly and the results fetched must be appropriate. This can be done by finding the right balance between performance and accuracy by using an indexing tool such as Elasticsearch. The DB's that store user profile info are going to be optimized using in memory caching tools.

5.2 Safety Requirements

The application operates above multiple abstraction layers over the hardware and kernel and there is little possibility of damage to a user's device. To prevent sharing of malicious links and scripts, a sanity check will be added to all content before uploading to the system.

5.3 Software Quality Attributes

Maintainability
Different versions of the product should be easy to maintain. For development it should be easy to add code to an existing system, should be easy to upgrade for new features and new technologies time to time. Maintenance should be cost effective and easy.

Usability
This can be measured in terms of ease of use. Application should be user friendly. Should be easy to learn. Navigation should be simple.

Flexibility

Should be flexible enough to modify. Adaptable to other products with which it needs interaction. Should be easy to interface with other standard 3rd party components.

5.4 Business Rules

The software will be free to use for all users and the source code will be publicly hosted for free use and modification.

6. Other Requirements

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: To Be Determined List