**SOFTWARE REQUIREMENTS**

**SPECIFICATION**

for

**<MOVIE RECOMMENDATION**

**SYSTEM>**

**PREPARED BY**<A SAMUEL(RA201104710097),

N FADIL(RA2011047010079),

GEETESH NIHAL BALLA (RA2011047010087),

SIVAPRASATH S(RA2011047010118),

YASH GOEL(RA2011047010101)>

**Table of Contents**

**1. Introduction** .............................................................................................................................3

1.1 Purpose ...................................................................................................................................3

1.2 Document Conventions ...........................................................................................................3

1.3 Intended And Reading Suggestions.........................................................................................3

1.4 Project Scope...........................................................................................................................4

**2. Product Overview** .....................................................................................................................4

2.1 Product Perspective .................................................................................................................4 2.2 Product Functions ................................................................................................................... 4 2.3 Design and Implementation Constraints ................................................................................. 4 2.4 Assumptions and Dependencies ............................................................................................ 5

**3. System Features**........................................................................................................................5

3.1 Sorting Of Movies ……………………………………………………………………………………..5

3.1.1 Description And priority …………………………………………………………………………….5

3.1.2 Stimulus/Response Sequences …………………………………………………………………...5

3.1.3 Functional Requirements …………………………………………………………………………..5

**4. External Interface Requirements** ............................................................................................5

4.1 User Interfaces ........................................................................................................................6

4.2 Hardware Interfaces ................................................................................................................6

4.3 Software Interfaces …………………………………………………………………………………..6

4.4 Communications Interfaces ………………………………………………………………………...,6

**5.Non Functional Requirements**…….……………………………………………………………………..7

5.1 Security………………………………………………………………………………………………….7

5.2 Reliability ……………………………………………………………………………………………….7

5.3 Availability ………………………………………………………………………………………………7

5.4 Maintainability ………………………………………………………………………………………….7

5.5 Portability ……………………………………………………………………………………………….7

**1.Introduction**

A recommendation system has become an indispensable component in various e-commerce applications. Recommender systems collect information about the user's preferences of different items (e.g. movies, shopping, tourism, TV, taxi) by two ways, either implicitly or explicitly.

Simply put a Recommendation System is a filtration program whose prime goal is to predict the "rating" or "preference” of a user towards a domain-specific item or item. In our case, this domain-specific item is a movie, therefore the main focus of our recommendation system is to filter and predict only those movies which a user would prefer given some data about the user him or herself.

The Simple Recommender offers generalized recommendations to every user based on movie popularity and (sometimes) genre. The basic idea behind this recommendation system is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the average audience.

**1.1 Purpose**

The goal of this document is to provide support information on the project (current version v1.0). It will attempt to explain the functionality of the program and the features it provides.

It will illustrate the purpose and complete declaration for the development of the system. It will also explain system constraints, interface and interactions with other external applications. This document is primarily intended to be proposed to the customer company for their approval and a reference for development of the system.

**1.2 Document Conventions**

The first initial of each component name is in capital. The terms service and system are interchangeable in this document.

**1.3 Intended Audience and Reading Suggestions**

This Software Requirements document is intended for:

- Developers who can review a project's capabilities and more easily understand where their efforts should be targeted to improve or add more features to it (design and code the application - it sets the guidelines for future development).

- Project testers can use this document as a base for their testing strategy as some bugs are easier to find using a requirements document. This way testing becomes more methodically organized.

- End users of this application who wish to read about what this project can do.

**1.4 Project Scope**

Movie Recommender is a movie recommendation system, which provides users movies which they may like, based on the movies previously seen. Every logged in user should have access to the recommender system. The system will go through the movies that the user previously saw and rated, then according to that information it should provide movies to the user. The project's main aim is to provide accurate movie recommendations to the user. This project is beneficial for the users and the companies. For users, they may find movies that they may like without consuming time and they can even encounter new movies which they like from the recommendations. For the company, they make the website more attractive, so they draw more users to the website and the system makes the users of the website spend more time online.

**2. Overall Description :**

**2.1 Product Perspective :**

A recommender system predicts the attitude of a user towards an item. As to this project, it makes recommendations on movies to users. First, the system receives movie names, then it retrieves genre and reviews about the movie and finally it shows a list of top 10 movies to the user.

**2.2 Product Features :**

The recommender system is given a group of movie names and then it collects reviews & ratings referring to such movies via the TMDB API. For each movie, it finds the details (director ,actors ,rating etc.) via the TMDB API (more introduction of the TMDB API can be found in Section 3 of the Detailed Design document). For potential users, reviews(on the same movies) published by their native citizens and followers are called “related reviews”. A potential user with too few related reviews is not processed.

For each review, a sentiment classifier [1] is used to determine its polarity - negative, positive, or unknown. For each potential user, the system calculates the percentages of three types of related reviews.

After the percentage calculation, the system uses cosine similarity to predict potential users’ evaluations for the movie. The percentages are used as attributes for the cosine similarity.

Also, the system outputs to a file, SWW all potential users who are predicted to hold positive opinions on a specific movie.

**2.3 Design and Implementation Constraints :**

1. The database may affect system performance as data increases.

2. The accuracy of the polarity classifier may affect the final prediction accuracy.

3. Unrelated reviews may affect the prediction accuracy. The TMDB cannot guarantee that all reviews it collects are definitely related to movies. For example, a review referring to “Harry Potter” may possibly talk about the book, not the movie.

4. Missing reviews may affect the prediction accuracy. The TMDB cannot guarantee that it collects all reviews talking about specific movies. For example, TMDB API does not allow collecting past reviews.

**2.4 Assumptions and Dependencies :**

The final product will be a REST-compliant Web service. A working Internet connection is required.

**3. System Features**

This project includes sorting of different movies according to the mood of the user which is based on the past experiences of the user. It is based on different past preferences of the user and what kind of Films user would like to watch.We only implement a simple user interface for showing system recommendations In the main system,user logins first and starts giving options of movies which are created on our recommendation service data. The output is shown in the main applications interface.

**3.1 SORTING OF MOVIES**

**3.1.1.Description and Priority**

Recommending movies on preferences of the user.High priority is given to preferences of the user.

**3.1.2.Stimulus/Response Sequences**

GENERATE DATA:User can get movie recommendations on the device. Movie information will be collected according to the time of action,mood,actor information,genre and rating. The database will be filled with info.

RECOMMENDATION:It can suggest movies as recommendations based on data set by user’s approach.The main function will show movies based on recommendation algorithm.When a user will choose movie recommendation the system will give it recommended movies through their past experiences.

**3.1.3Functional Requirements**

>Top 10 recommended movies

>A feature which use NLP for sentiment analysis and predicts the movie whether it is good or bad.

>A feature showing overview,genre,release date,run time,status of the movie.

>A feature showing top cast of the movie and know more option of the top cast.

Database:The database based on past preferences of the user will be created.

Algorithm:It will capture preferences and suggest movies based on it.

Historical Data:It will be collected by the System. Searching reporting requirement:Will tell users how to search data.

**4. External Interface Requirements**

**4.1 User Interfaces**

The user interface for the software shall be compatible with any browser such as Internet Explorer, Mozilla or Google Chrome by which user can access to the system. For the Front-end: HTML and CSS For the Back-end: JavaScript For Machine Learning: Python

**4.2 Hardware Interfaces**

Since the application must run over the internet, this brings out the requirement of a network interface on the device. User should have: A device with valid internet connection or Wi-Fi. A browser that supports HTML and JavaScript.

**4.3 Software Interfaces**

The data of the project is organized in multiple datasets containing data of films (their genre, language, etc.) as it makes it easier to curate data with a large number of attributes. Python 3.10.0 will be used as the predominant programming language for this project. Since the project is based on Machine Learning Python is the clear as it has built-in modules for handling and cleansing data e.g., Pandas and NumPy and also for training the recommendation model e.g., Scikit-Learn

**4.4 Communications Interfaces**

The movie recommender system shall use the HTTP protocol for communication over the internet and for the intranet communication will be through TCP/IP protocol suite.

**5.Other Nonfunctional Requirements**

Nonfunctional Requirements (NFRs) define the system attributes such as security, reliability, performance, maintainability, scalability and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

NFRs are just as critical as functional requirements to the software. Failing to meet any one of them can result in the system failing to satisfy the business expectations, user or market needs. They are revisited for each iteration, program increment or release. Therefore, proper definition and implementation of NFRs is critical and necessary to the system.

Performance Requirements

The steps involved to perform the implementation of movie recommendation is as follows:

A) E-R Diagram

The E-R Diagram is a technique to represent the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalizing relation and finally obtaining a relation database.

ENTITIES: Which specify distinct real-world items in an application.

PROPERTIES/ATTRIBUTES: Which specify properties of an entity and relationships.

RELATIONSHIPS: Which connect entities and represent meaningful dependencies between them.

B) Normalization

The basic objective of normalization is to reduce the redundancy which means that information is to be stored only once. If a database is not properly designed it can give rise to modification anomalies.

Modification anomalies arise when data is added to, changed or deleted from a database table. Similarly, in traditional databases as well as improperly designed relational databases, data redundancy can be a problem. These can be eliminated by normalizing a database.

**5.1) Security**

The movie recommendation system does not require login or transactions or any other confidential information.

The system is safe and does not inquire about cookies so there is no fear of the computer’s password being stolen.

The back-end servers are accessible only to the administrators.

The connection is secure and the system is safe to browse.

All data obtained from the dataset are encrypted before being sent over in the net.

**5.2) Reliability**

The reliability of the system depends on the reliability of the separate components. The main and most important component of the system is the movie datasets which are continuously maintained and updated accordingly.

Thus, the overall stability of the system depends on the stability of the datasets and given that the datasets are encrypted and secure, it can be said that the overall system is reliable.

**5.3) Availability**

The system will be available at all times, meaning the user can access it using a web browser. In case of any failure or corruption of the database, it would be retrieved from the server by the administrator and then the service will be restarted.

**5.4) Maintainability**

The software is done in increments with modularity in mind so that maintainability can be done efficiently.

**5.5) Portability**

The end user part of the system is fully portable and any system using any web browser should be able to use it on any operating system.

The system can run on PC, Laptops, Smartphones etc.