PROM02 Computer Master's Project

Movie Recommender System

Functional Specification

Name:	Chu Siu Kay Alan
Student No:	189222006
Version:	0.6
Date:	Feb 17, 2021

1 Purpose

This purpose of this document is to define both the functional requirements and non-functional requirements of the movie recommender system. Besides that, this document also provides an overview on the architecture of the movie recommender system.

2 Business Requirement

The main goal of the movie recommender system is to suggest movies that are potentially interested by the users based on either users' preferences or the ratings that they gave in the past. In addition, the system should provide a dashboard that shows the statistical information about the user preferences and ratings. The statistical information should allow the company to discover which movie genres are popular so that the company could consider uploading more of the movies with those genres in the future. Beside the statistical information, the dashboard should show the graphs of the resulting recommendations so that the staff can have an idea how the recommendation is made by the system.

2.1 Use Cases

The online movie system is WEB based system which allows users to access using either WEB browser or mobile app. The movie recommender system is a subsystem of the online movie system. It should provide the two main functionalities, "view recommendation" and "view statistics" as depicted in the green ovals in Figure 2.1. The "view recommendation" function should show the recommended movies to the customer. The recommendation is based on his/her ratings on the movies in the past and his/her favourite movie genres. The "view statistics" function should allow the company staff to check on the statistical information about the movies and ratings. The information should be useful to the company to determine its strategy.

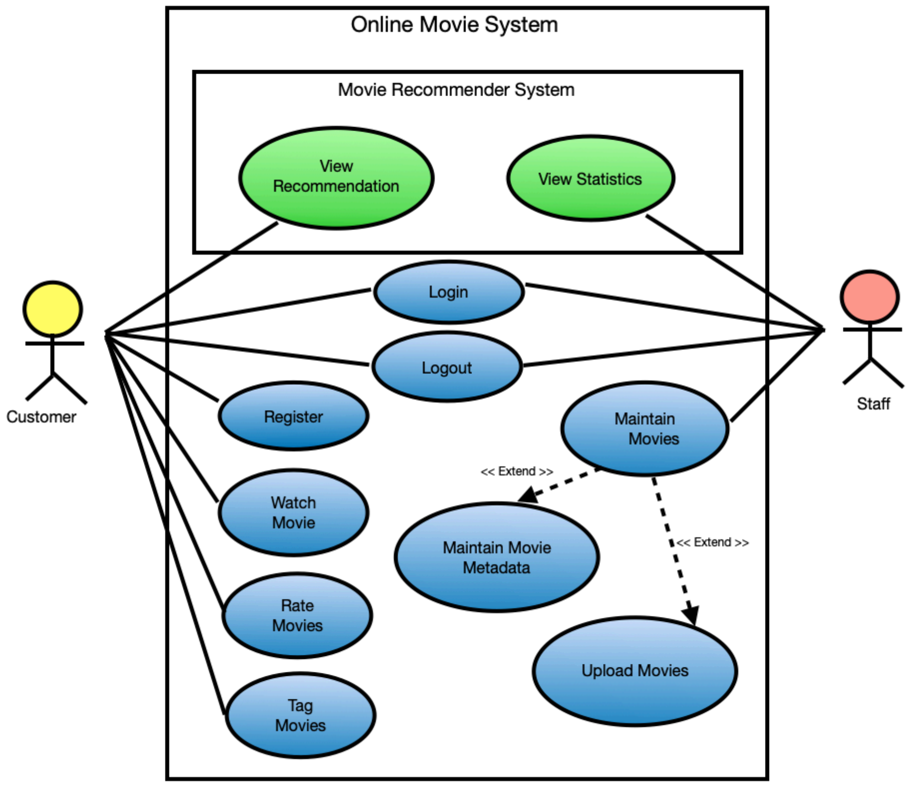

Figure 2.1 Use Case diagram of the online movie system

3 Data Source

GroupLens Research has provided MovieLens datasets for research and educational use (F. Maxwell Harper and Joseph A. Konstan., 2015). The machine learning models of the movie recommender prototype was built upon the latest small movie lens dataset (ml-latest-small). This dataset describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. It contains 100836 ratings and 3683 tag applications across 9742 movies. The data are contained in four CSV files as described in Table 3.1.

File name	Column	Column Description
movies.csv	movieId	Identifier of a movie used in MovieLens
	title	Title of a movie
	genres	A list of genres of a movie in which genres are delimited by the character "|"
links.csv	movieId	Identifier of a movie used in MovieLens
	imdbId	Identifier of a movie used in IMDB
	tmdbId	Identifier of a movie used in TMDB
ratings.csv	userId	User ID of MovieLens
	movieId	Identifier of a movie used in MovieLens
	rating	Rating given by a user on a movie
	timestamp	The time when the rating was given by the user
tags.csv	userId	User ID of MovieLens
	movieId	Identifier of a movie used in MovieLens
	tag	Tag given by a user to associated with a movie
	timestamp	The time when the tag was given by the user

Table 3.1

4 Functional Requirements

This section describes the functional requirements (FR) of the movie recommender system. The functional requirements can be described by a set of user stories which is summarised in Table 4.. There are 11 FR in total. 6 of them belong to the customer user cases and 5 of them belong the staff use cases.

User Story ID	As A <Type of user>	I want to <perform some task>	So that I can <achieve some goal>
C1	As a customer	I want to be given a set of recommendations based on the ratings that I have given in the past.	So that I can select the movies that I like without much navigation.
C2	As a customer	I want to be given a set of recommendations based on the popularity of the movies.	So that I can select the movies that I like without much navigation.
C3	As a customer	I want to be given a set of recommendations based on the similarity of the movies that I liked before	So that I can select the movies that I like without much navigation.
C4	As a customer	I want to see the recent movies that I have rated with my ratings	So that I can be reminded which movies I have rated with good ratings or bad ratings
C5	As a customer	I want to see the average ratings of the recommended movies	So that I can have an ideas how the others rated the recommended movies
C6	As a customer	I want to be given a set of recommendations based on what the others similar to me like	So that I can select the movies that are liked by the people who are similar to me
S1	As a staff	I want to check the top 10 movies at a specific period. Also the number of ratings given has to be at least equal to the minimum number of ratings specified	So that I can know the top 10 movies that are popular at what period of time
S2	As a staff	I want to check which movie genres and tags are popular / unpopular	So that I can determine which types of movies the company should upload to the system
S3	As a staff	I want to check the distribution of the movie ratings	So that I can have an ideas what ratings are generally given by the customers
S4	As a staff	I want to check the number of ratings given per customer for each year	So that I can have an ideas how active customers are regarding to rating movies
S5	As a staff	I want to see the reasoning behind the recommendations	So that I can determine whether the recommendations make any sense

Table 4.1

5 Non-Functional Requirement

This section describes the non-functional requirements (NFR) of the movie recommender system. There are 7 NFR as summarised in Table 5.1 below.

NFR ID	Description
NFR1	The page that displays the recommended movies to customers should be loaded within 3 seconds
NFR2	The page that displays the statistics information to staff should be loaded within 3 seconds
NFR3	The system should be accessible by using WEB browser or mobile application
NFR4	The system should have resilience
NFR5	The system should be 99.99% available
NFR6	The system should be able to recover or failover to DR within 15 minutes when disaster happens in the primary site
NFR7	The system should be scalable

Table 5.1

6 Architecture Design

This section discusses the architecture design of the movie recommender system, which addresses the functional and non-functional requirements shown in Section 4 and Section 5. Figure 6.1 shows the proposed architecture for the production movie recommender system.

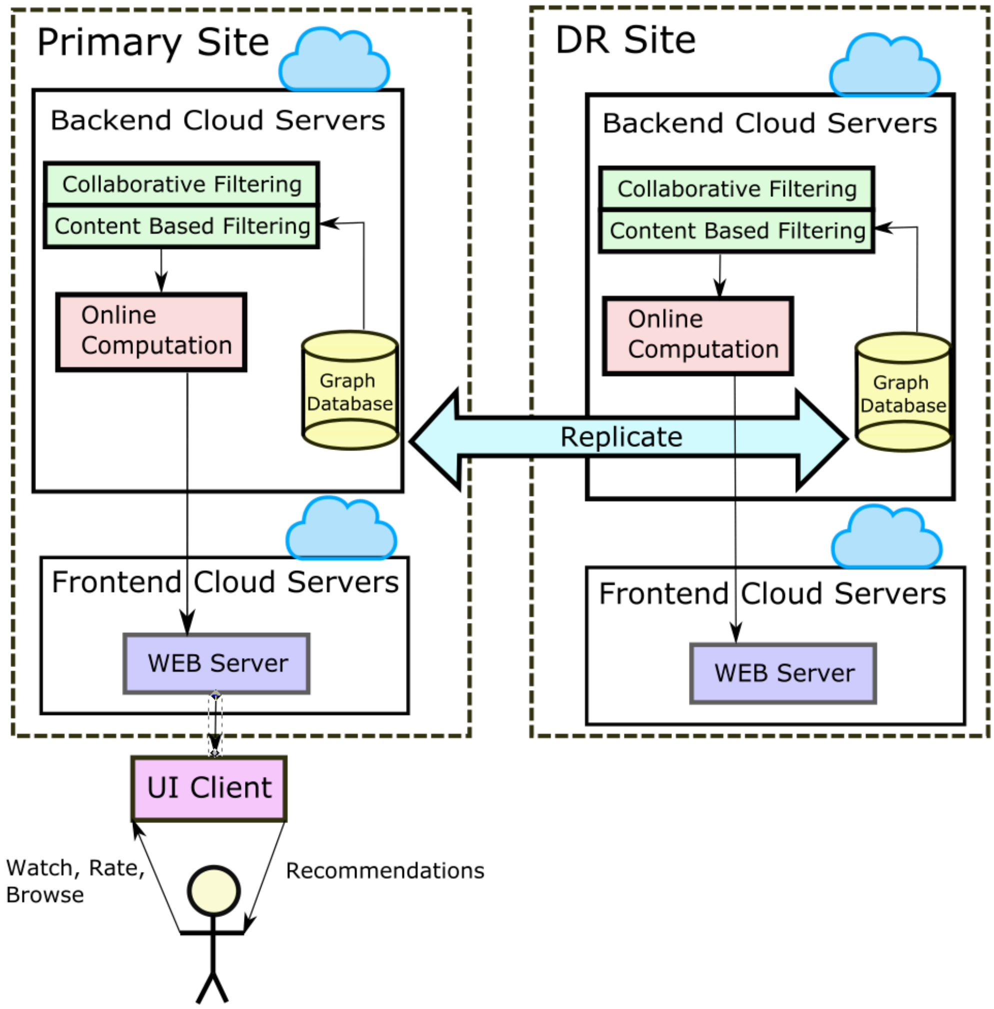

Figure 6.1 Architecture of the production system

In order to satisfy the resilience and availability requirements stated in Table 5.1, the proposed architecture design consists of a primary site and a disaster recovery (DR) site. The data is synchronised from primary to DR in real time to guarantee no data lost after failover. In the frontend, there is a WEB server to serve the WEB application to the customers. When the frontend servers need to obtain the recommendations, it will request the backend server to perform the online computation by applying both collaborative filtering and content based filtering to the graph database. Once the recommendations are generated, the result will be transferred back to the frontend server. The online computation module is also responsible for updating the graph database so that the graph data can be kept up to date all the time. For example, the customer may rate a new movie during the session. The online computation will re-compute the recommendation based on the updated data. This proposed design is inspired by the architecture design used in Netflix (Xavier A., 2019).

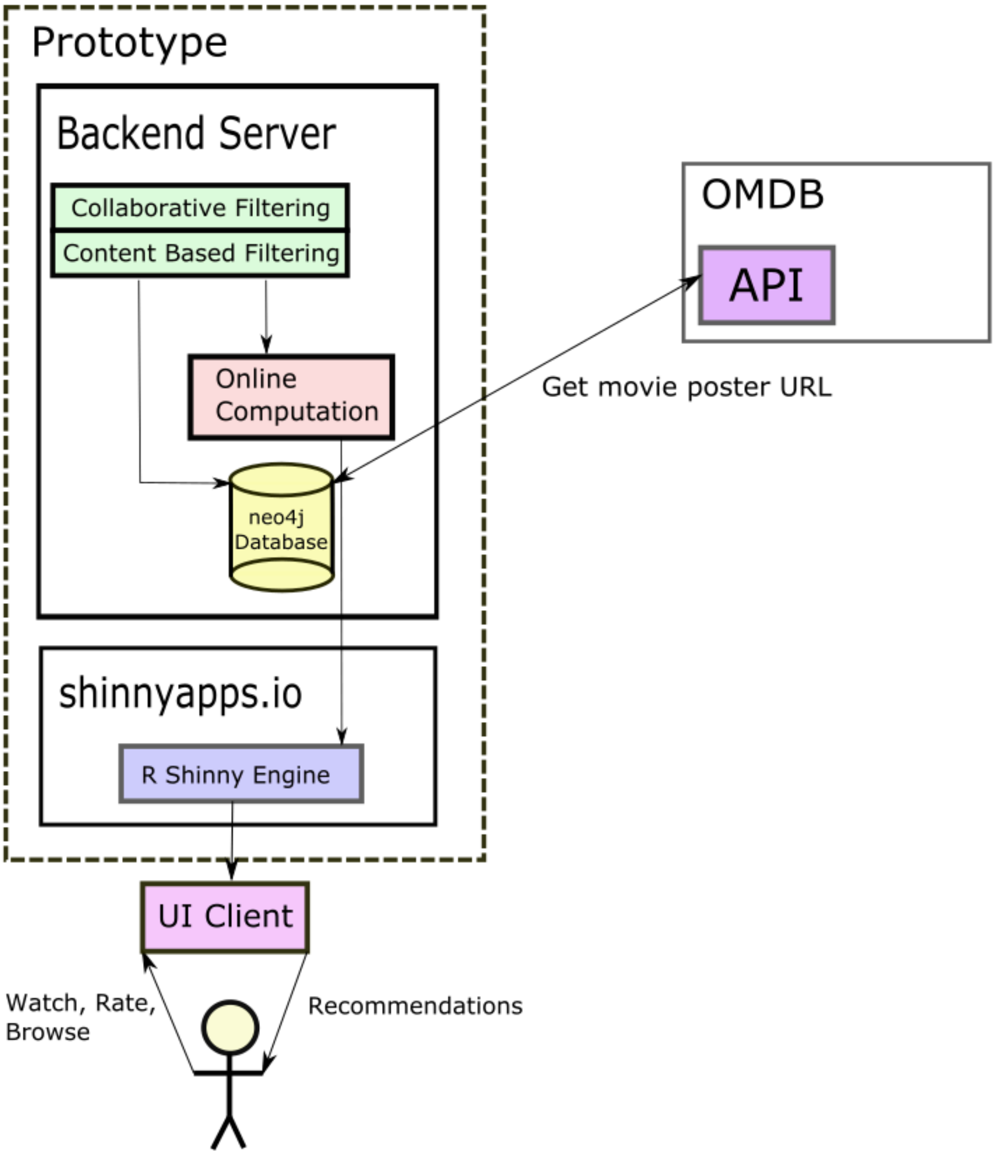

Figure 6.2 Architecture of the prototype

As the prototype is to mainly demonstrate the functional requirements of the movie recommender system, the architecture of the prototype is simplified as shown in Figure 6.2. Since the original data set does not include the movie poster images, a third party API provided by OMDB is used in the backend server to obtain the URLs retrieving the movie poster images. The URLs are then loaded to the Movie objects in the graph database. The R Shinny server connects directly to the backend server to access the neo4j graph database.

7 Risk Assessment

Risk assessment is another critical part of a software project. Failure to identify the risks could possibly lead to project failure. According to (Riaz, M. T. et al. (2019)), the risks can be identified and classified using Fishbone Analysis as shown in Figure 7.1.

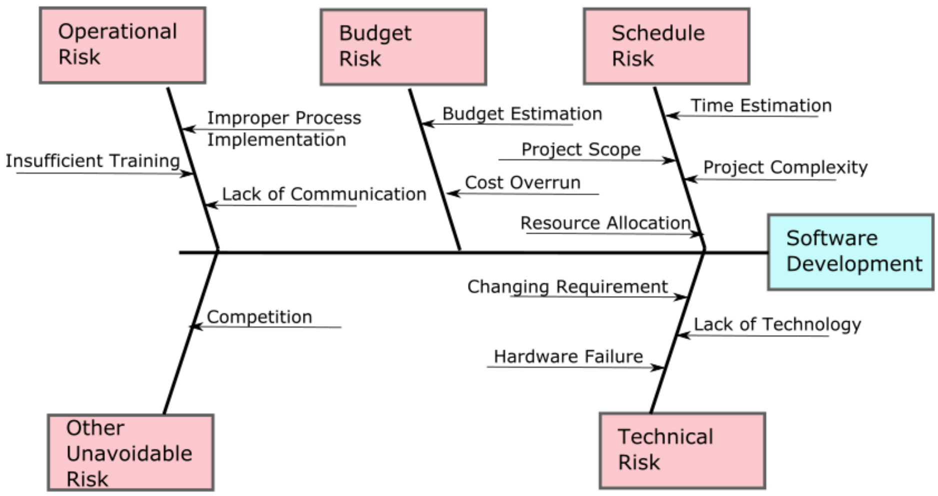

Figure 7.1 Fishbon Analysis

The next step is to derive a qualitative risk assessment for each risk identified as shown in Table 7.3. The assessment is based on the risk impact definition as shown in Table 7.2 and the risk matrix defined in Table 7.1.

	Rare	Unlikely	Possible	Likely	Certain
Catastrophic	Low Med	Medium	Med High	High	High
Critical	Low	Low Med	Medium	Med High	High
Moderate	Low	Low Med	Medium	Med High	Med High
Minor	Low	Low Med	Low Med	Medium	Med High
Neglectable	Low	Low	Low Med	Medium	Medium

Table 7.1 Risk Matrix

Impact	Definition
Catastrophic	Not able to deliver the project
Critical	1 year delay
Moderate	6 month delay
Minor	3 month delay
Neglectable	< 3 month delay

Table 7.2 Risk Impact Definition

Type	Risk	Risk Levels	Liklihood	Impact	Risk Management Technique
Schedule Risk	Project Scope	Medium	Possible	Moderate	Avoid This risk can be avoided by defining a very clear and detailed definition of done of the project.
	Project Complexity	Medium	Possible	Moderate	Avoid This risk can be avoided by defining a very clear functional requirement specification.
	Time Estimation	Medium	Likely	Minor	Reduce The risk can be reduced by regularly reviewing the progress and revising the project plan.
Technical Risk	Changing Requirements	Medium	Possible	Moderate	Avoid This risk can be avoided by defining a very clear and detailed definition of done of the project.
	Lack of Technology	Low Med	Unlikely	Critical	Reduce This risk is unlikely to happen as the requirement of the technology is no high. This risk can be reduced if the technical requirement is clearly defined.
	Hardware Failure	Low	Rare	Neglectable	Transfer As cloud solution is proposed. This risk is transferred to the cloud service provider.
Budget Risk	Budget Estimation	Med High	Possible	Catastrophic	Reduce This risk can be reduced by applying the controlling procedures and creating budgets that are regularly revised and updated.
	Cost Overrun	Med High	Possible	Catastrophic	Reduce This risk can be reduced by the following: 1) Track the project progress closely 2) Stay within the scope originally planned 3) Ensure stakeholders in the project on the same page
Operational Risk	Improper Process Implementation	Medium	Possible	Moderate	Reduce This risk can be reduced by regularly reviewing the processes.
	Lack of Communication	Low Med	Possible	Minor	Reduce This risk can be reduced by regularly arranging meetings for aligning people's ideas.
	Insufficient Training	Low Med	Unlikely	Moderate	Avoid This risk can be avoided if sufficient training is provided to staff.
Other Unavoidable Risk	Competition	Medium	Possible	Catastrophic	Accept This risk is not avoidable. There is always competition from another company that produces recommender systems.

Table 7.3 Quantitative Risk Assessment

8 References

Siebes, R., Mika, P., Menken, M., & Haase, P. (2004). Evaluation plan. *SWAP Project Deliverable D*, *10*.

Riaz, M. T. et al. (2019) ‘Risk Assessment on Software Development using Fishbone Analysis’, 2019 International Conference on Data and Software Engineering (ICoDSE), Data and Software Engineering (ICoDSE), 2019 International Conference on, pp. 1–6. doi: 10.1109/ICoDSE48700.2019.9092727.

Hahsler M., J. et al. (2020) ‘recommenderlab: A Framework for Developing and Testing Recommendation Algorithms’, Southern Methodist University Available at:https://cran.r-project.org/web/packages/recommenderlab/vignettes/recommenderlab.pdf.

F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4: 19:1–19:19. <https://doi.org/10.1145/2827872>

Carlos A. Gomez-Uribe and NEIL H. (2015) ‘The Netflix Recommender System: Algorithms, Business Value, and Innovation’, Netflix, Inc.

Jeffrey S. S., Ivan S., Kevin C. (2017) ’Comparing Data Science Project Management Methodologies via a Controlled Experiment’, Proceedings of the 50th Hawaii International Conference on System Sciences | 2017

Walek, B. and Fojtik, V. (2020) ‘A hybrid recommender system for recommending relevant movies using an expert system’, Expert Systems With Applications, 158. doi: 10.1016/j.eswa.2020.113452.

Olena Popova (2019) ‘Adaptation of flexible project management models based on Scrum and Kanban technologies’, Tehnologìčnij Audit ta Rezervi Virobnictva, 4(2(48)), pp. 4–10. doi: 10.15587/2312-8372.2019.180459.

Xavier A. (2019) ’Big & Personal: data and models behind Netflix recommendations’. BigMine '13: Proceedings of the 2nd International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications August 2013 Pages 1–6 Available at: https://doi.org/10.1145/2501221.2501222

Haitao Sun (2011) ‘Knowledge for Software Quality Control and Measurement’, 2011 International Conference on Business Computing and Global Informatization, Business Computing and Global Informatization (BCGIN), 2011 International Conference on, pp. 468–470. doi: 10.1109/BCGIn.2011.123.

Ashton Williams (2020) ‘Quality Management in Project Management and Agile practices’, Scrum Time, assessed 17 July 2020, <https://scrumtime.org/quality-management-in-project-management-and-agile-practices/>

Ed Caldeira (2012) ‘Construction Quality Assurance/Quality Control Blog’, Quality First Time, assessed 17 July 2020, <https://www.firsttimequality.com/Blog/bid/75546/How-to-Write-a-Construction-Quality-Control-Plan>

Limesh Parekh (2017) ‘How Start-ups Can Benefit by Using Customer Relationship Management’, Entrepreneur, assessed 17 July 2020, <https://www.entrepreneur.com/article/298189>