

User Song Preferences using Preference Learning in Artificial Intelligence

By

Goh Beng Jun

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2022

REPORT STATUS DECLARATION FORM

Title: User Song Preferences using Preference Learning in Artificial Intelligence

Academic Session: Jan 2022

I GOH BENG JUN
(CAPITAL LETTER)


declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

Gob

(Author's signature)


Online Signature during COVID-19
SOONG HOONG CHENG (Lecturer)
FACULTY OF IET
UNIVERSITI TUNKU ABDUL RAHMAN (UTAR)
PERAK CAMPUS, JLN UNIVERSITI, BANDAR BARUK,
31110 KAMPAR

(Supervisor's signature)

Address:

30,Lorong Tropika 2, Tropika Residency,
14020,bm _____

Ts Soong Hoong Cheng
Supervisor's name

Date: 18/04/2022

Date: 20/04/2022

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 18 JAN 2022	Page No.: 1 of 1

FACULTY/INSTITUTE* OF Information and Communication Technology

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 18/04/2022

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that **Goh Beng Jun** (ID No: **1805435**) has completed this final year project/ dissertation/ thesis* entitled “ User Song Preferences using Preference Learning in Artificial Intelligence ” under the supervision of Ts Soong Hoong Cheng (Supervisor) from the Department of DDET, Faculty/Institute* of FICT , and _____ (Co-Supervisor)* from the Department of _____, Faculty/Institute* of _____.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,



(*Goh Beng Jun*)

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled “**User Song Preferences using Preference Learning in Artificial Intelligence**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : *Goh Beng Jun*

Name : Goh Beng Jun

Date : 18-04-2022

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, **Ts Soong Hoong Cheng** who has given me this bright opportunity to engage in a User Song Preferences using Preference Learning in Artificial Intelligence project. It is my first step to establish a career in Artificial Intelligence and Machine Learning field. A million thanks to you.

To a very special person in my life, Ts Soong Hoong Cheng, for his patience, unconditional support, and love, and for standing by my side during hard times. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

ABSTRACT

Existing music industries are working on building high-precision music recommender system. The music recommender system must provide an efficient way to manage songs and help their customers to classify all the songs based on genres, artists, age groups, locations, and language by giving quality recommendation. This is because users will experience difficulties choosing from the millions of songs. In this project, the main goal is to classify the millions of songs in accordance with the taste of the users. With a good recommender system, the user will spend less time looking for the songs and more time enjoying their favourite song and discovering new favourites. Thus, a good quality music recommender system will have a strong user base and will create a strong thriving market. This project will implement various algorithms to compare the results with one another to figure out which is the most effective algorithm that is suited for a music recommender system. The most common model is popularity-based model because it is simple and intuitive. That another algorithm is collaborative filtering, this algorithm aims to find similarity between users, songs, and artists. This experiment will use SVD model, KNN model, based on metadata. The problems faced when making this project is a metadata of song are too big and processing such a large dataset is GPU-intensive and requires a lot of system memory. The accuracy results of the experiments are very low due to large dataset. Collaborative filtering will be used in this project. The future work for this project is to provide enough storage for the song to be downloaded into the dataset so that the user can play it immediately rather than relying on YouTube links.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xiii
LIST OF ABBREVIATIONS	xiv
CHAPTER 1 INTRODUCTION	1
1.1 Background Information	1
1.2 Problem Statement and Motivation	2
1.3 Project Objectives	3
1.4 Project Scope	5
1.5 Impact, Significant and Contribution	6
1.6 Report Organization	7
CHAPTER 2 LITERATURE REVIEW	8
2.1 Music Recommender System CS365: Artificial Intelligence	8
2.2 How to Build a Simple Song Recommender System	10
2.3 Spotify	15
2.4 Apple Music	17
2.5 Last FM	19
CHAPTER 3 SYSTEM METHODOLOGY/APPROACH	21
3.1 Design Specifications	21
3.1.1 Methodologies and General Work Procedures	21
3.1.2 Tools to Use	24

3.1.3	User Requirements	28
3.1.4	System Performance Definition	28
3.2	System Design/Overview	29
3.2.1	System Flowchart	29
3.2.2	User Interface Design	35
3.2.3	Use Case Diagram	36
3.2.4	Use Case Description	37
3.2.5	Activity Diagram	43
3.3	Implementation Issues and Challenges	49
3.4	Verification Plan	50
3.5	Timeline	58
CHAPTER 4 EXPERIMENT AND MODEL IMPLEMENTATION		60
4.1	Experiment Conducted with Various Model	60
4.2	Model Implementation with Collaborative Filtering Method	63
CHAPTER 5 SYSTEM IMPLEMENTATION AND TESTING		67
5.1	Project Screenshot and Explanation	67
5.1.1	Login Interface	67
5.1.2	Login Fail	68
5.1.3	Sign Up	68
5.1.4	Reset Password	69
5.1.5	Firebase	71
5.1.6	Main Menu	72
5.1.7	Play Song	73
5.1.8	Refresh Song	74
5.1.9	Add to Favourite	76
5.1.10	Log Out	78
5.2	Module Testing	79
5.2.1	Account Login Testing	79
5.2.2	Account Registration Testing	79
5.2.3	Reset Password Testing	79
5.2.4	Play Song Testing	80

5.2.5	Refresh Song Testing	80
5.2.6	Add to Favourite Testing	80
5.2.7	Remove Favourite Testing	81
5.2.8	Logout Testing	81
CHAPTER 6 CONCLUSION		82
REFERENCES		84
APPENDIX A		
A.1	IEEE International Conference on Computing	A-1
A.2	Professional Conference Research Presentation ICOCO 2021	A-7
A.3	Certificate of Presentation	A-8
WEEKLY LOG		94
POSTER		100
PLAGIARISM CHECK RESULT		101
FYP2 CHECKLIST		109

LIST OF FIGURES

Figure Number	Title	Page
Figure 2.2.1	Define file.	10
Figure 2.2.2	Combine 2 files.	10
Figure 2.2.3	Visualize combined data.	11
Figure 2.2.4	Data transformation.	11
Figure 2.2.5	Output of data transformation.	12
Figure 2.2.6	SVD model.	12
Figure 2.2.7	SVD code.	13
Figure 2.2.8	Item based model.	14
Figure 2.2.9	Top 10 recommend songs.	14
Figure 2.3.1	Recommended function by Spotify.	15
Figure 2.4.1	Apple music interface.	17
Figure 2.5.1	Last fm interface.	19
Figure 3.1.1.1	System Development Life Cycle (SDLC).	21
Figure 3.1.1.2	Songs.csv file.	23
Figure 3.1.1.3	Videosongs.csv file.	23
Figure 3.1.2.1	User and Item Based Filtering.	25
Figure 3.1.2.3	Software Involved.	27
Figure 3.2.1.1	System flowchart.	29
Figure 3.2.1.2	Dataset of songs.csv file.	30
Figure 3.2.1.3	Dataset of videosongs.csv file.	30
Figure 3.2.1.4	Null values of songs.csv.	31
Figure 3.2.1.5	Null values of videosongs.csv.	31
Figure 3.2.1.6	Checking duplicate records for the 2 datasets.	31
Figure 3.2.1.7	Total number of songs sing by each artist.	32
Figure 3.2.1.8	Total no of songs sing in each year.	32
Figure 3.2.1.9	Visualizing songs by each artist with respect to year.	33
Figure 3.2.1.10	Merged dataframe and the percentage of the	33

	listen_count.	
Figure 3.2.1.11	Dividing dataset to test set.	34
Figure 3.2.2.1	User Interface Design for Recommender System.	35
Figure 3.2.3.1	Use Case Diagram.	36
Figure 3.2.5.1	Activity Diagram of Register Account.	43
Figure 3.2.5.2	Activity Diagram of Login Account.	44
Figure 3.2.5.3	Activity Diagram of Play Song.	45
Figure 3.2.5.4	Activity Diagram of Refresh Song.	46
Figure 3.2.5.5	Activity Diagram of Add to Favourite.	47
Figure 3.2.5.6	Activity Diagram of Logout.	48
Figure 3.5.1.1	Timeline of FYP 1.	58
Figure 3.5.1.2	Timeline of FYP 2.	59
Figure 4.1.1	Experiment Conducted of All Models.	60
Figure 4.1.2	Classification Report of Logistic Regression.	61
Figure 4.1.3	Actual vs Predicted Value.	62
Figure 4.2.1	Construct Cooccurrence Matrix.	63
Figure 4.2.2	Make Recommendations with Cooccurrence Matrix	64
Figure 4.2.3	Create and Use the Model to Make Recommendations	65
Figure 4.2.4	The Results of the Recommender System	66
Figure 5.1.1	Code of Unzip The Pickle File	67
Figure 5.1.1.1	Login Interface	67
Figure 5.1.2.1	Login Fail	67
Figure 5.1.3.1	Sign Up	68
Figure 5.1.3.2	Sign Up Format	69
Figure 5.1.4.1	Reset Password	69
Figure 5.1.4.2	Check Inbox	70
Figure 5.1.4.3	Send Link	70
Figure 5.1.4.4	Key in New Password	70
Figure 5.1.4.5	Expired Link	71
Figure 5.1.5.1	Authentication	71
Figure 5.1.6.1	Main Menu	72
Figure 5.1.7.1	Click Play Song	73
Figure 5.1.7.2	YouTube Video	73

Figure 5.1.8.1	Click Refresh Button	74
Figure 5.1.8.2	Refresh Song	74
Figure 5.1.8.3	Refresh Function	75
Figure 5.1.9.1	Click Favourite Button	76
Figure 5.1.9.2	Favourite List	76
Figure 5.1.9.3	Firebase	76
Figure 5.1.9.4	Existing Song	77
Figure 5.1.9.5	Delete Song	77
Figure 5.1.9.6	Click Home Button	77
Figure 5.1.10.1	Click Log Out Button	78
Figure 5.1.10.2	Log Out	78

LIST OF TABLES

Table Number	Title	Page
Table 1.3.2.1	Project Objective that Solved the Problem Statement	4
Table 2.5.1	Comparison of Existing System	20
Table 3.1.2.2	Specification of Laptop	27
Table 3.2.4.1	Use Case Description (Register Account)	37
Table 3.2.4.2	Use Case Description (Login Account)	38
Table 3.2.4.3	Use Case Description (Play Song)	39
Table 3.2.4.4	Use Case Description (Refresh Song)	40
Table 3.2.4.5	Use Case Description (Add to Favourite)	41
Table 3.2.4.6	Use Case Description (Log Out)	42
Table 3.4.1	Verification Plan P1 (Register Account)	50
Table 3.4.2	Verification Plan P2 (Login Account)	51
Table 3.4.3	Verification Plan P3 (Reset Password)	52
Table 3.4.4	Verification Plan P4 (Play Songs)	53
Table 3.4.5	Verification Plan P5 (Refresh Song)	54
Table 3.4.6	Verification Plan P6 (Add to Favourite)	55
Table 3.4.7	Verification Plan P7 (Remove Favourite)	56
Table 3.4.8	Verification Plan P8 (Log Out)	57
Table 5.2.1.1	Account Login Testing	79
Table 5.2.2.1	Account Registration Testing	79
Table 5.2.3.1	Reset Password Testing	79
Table 5.2.4.1	Play Song Testing	80
Table 5.2.5.1	Refresh Song Testing	80
Table 5.2.6.1	Add to Favourite Testing	80
Table 5.2.7.1	Remove Favourite Testing	81
Table 5.2.8.1	Logout Testing	81

LIST OF ABBREVIATIONS

<i>SVD</i>	Singular Value Decomposition
<i>KNN</i>	K-Nearest Neighbour
<i>SDLC</i>	Software Development LifeCycle

Chapter 1: Introduction

1.1 Background information

Nowadays a lot of music company like Spotify, Apple music are allowing us to access different music resources more conveniently. Most of the music industries are using music recommender systems and the old way of selling music has been replaced by a totally different cloud-based solution. This allows users can listen songs directly from the cloud. However, peoples will experience difficulties choosing from the millions of songs available in such services. In this project, the music recommender system must provide an efficient way to manage songs and help their customers to classify all the songs based on genres, artists, age groups, locations, and language by giving quality recommendation [1].

Currently, many music industries are working on building high-precision music recommender system. The main goal is to classify the millions of songs in accordance with the taste of the users. These companies are trying to improve their customer's ability to discover relevant music and the quality of the recommender service will attract more customers [2]. With a good recommender system, the user will spend less time to looking the songs and more time on enjoying their favourite song and discovering new favourite. Thus, a good quality music recommender system will have a strong user base and will create a strong thriving market [3].

Furthermore, the music recommender system will learn from the user's listening history and recommend song the user will likely enjoy based on their listening pattern. This project will implement various algorithms to compare the results with one another to figure out which is the most effective algorithm that is suited for a music recommender system. The most common model is popularity-based model because it is simple and intuitive. That another algorithm is collaborative filtering, this algorithm aims to find similarity between users, songs and artists. This experiment will use SVD model, KNN model, based on metadata. The problems face when making this project is a metadata of song are too big and processing such a large dataset is CPU-intensive and require a lot of system memory. So, to overcome system resources limitations, the original dataset is made into a subset for optimum usage. However, the creation of subset will require complex data pre-processing [4].

1.2 Problem Statement and Motivation

Without a doubt, music is an important element in the lives of mankind. The invention of music-playing websites and applications has made music accessible in the user's daily lives. With the rapid development of mobile devices and more people having internet access, people have access to music collections at an unprecedented scale. Music libraries can easily have more than 15 million songs, people will feel overwhelmed choosing from the ocean of song available. Thus, an efficient song recommender system is necessary for the music service providers and customers. However, algorithms that would satisfy all users expectations are yet to be perfected.

Suggest the best model for song recommendation.

The main goal is suggesting the best model for song recommendation. This is hard to decide which algorithms are the most suitable model for song recommendation. Moreover, music industries would like to build high precision recommender system to attract more customers.

Automatic playlist continuation

The second problem is automatic playlist continuation [5]. The task of the automatic playlist generation is referring to the automated creation of the sequences of tracks. Moreover, the automatic playlist generation should add one or many tracks to the playlist which fits the same target characteristics of the original playlist.

Cold start problem

The third problem is cold start problem. The meaning of cold start is when a new user registers to the system or a new song is added into the library, but the system does not have sufficient data associated with the new song or user. In the cold start problem, the music recommender system is unable to accurately recommend a song that suits the user's preferences due to insufficient user data. When a new song is added to the library, the recommender system will neglect to recommend the new song to existing users due to no user listening history available.

Huge metadata

The project is using English songs as our metadata. The problem faced is the metadata dataset is very large and needs to be broken down into smaller subsets. However, this will affect the result of training the model due to the subset containing less metadata of songs.

1.3 Project Objectives

1.3.1 Learning Objective

The learning objective of doing this project is to first learn about machine learning and its key concepts and various data mining techniques and algorithms. The second learning objective is to learn how to implement machine learning algorithms into song recommender system.

1.3.2 Project objective

I. To define the best algorithm for the recommender system.

The first project objective is to define the best algorithm for the recommender system. Using a matrix that is based on precision percentage various algorithm. This project will be applying all types of classification models such as Random Forest Classifier, Logistic Regression, Support Vector Machine, Naïve Bayes, Decision Tree Classifier and K Neighbours Classifier [6].

II. To find out the solution of automatic playlist continuation.

The collaborative-based model will be train by dataset to recommend song to the users. The model can solve the problem of creating automatic playlist continuation. This model will collect the information from many users and then make a prediction based on correlation measures between users and item.

III. To find out the solution of cold start problem.

This scenario can be overcome by collaborative-based model because this model will recommend the topmost popular songs. Song recommended by collaborative-based model will have the highest probability of the new user liking the song.

IV. To separate the huge dataset to smaller subset.

This objective is the huge metadata needs to be broken down into smaller subset to achieve higher accuracy and precision results. The larger dataset requires more GPU consumption and spend more time to train the models. Smaller subset will be more effective for this project as lack of resources to conduct the experiments with larger dataset.

Problem statement	Project Objective
To define the best algorithm for the recommender system.	Find the algorithm with the highest precision and accuracy percentage.
To find out the solution of automatic playlist continuation.	Train the collaborative-based model with dataset contains user id and song to recommend song.
To find out the solution of cold start problem.	Use collaborative-based model to recommend top ranking of songs.
To separate the huge dataset to smaller subset.	Break the dataset to one subset to achieve higher accuracy and save more time on training models.

Table 1.3.2.1 Project Objective that Solved the Problem Statement

1.4 Project Scope

In last project, which is FYP 1, we are developed “User Song Preferences using Preference Learning in Artificial Intelligence” model of a system. Moreover, this project is conducted using English songs and determine which model is most effective and suitable songs for users by giving quality algorithms. The system will be focus using **Collaborative filtering model** to conduct. This system can be used for every user and enjoy the songs which is arrange by the recommender system.

In this FYP 2, the project will developing web-based system to implement our recommender system by the artificial intelligence. The system will be using an application which is Heroku to deploy the project. Users may easily to enjoy the system by registration or login the account.

The purpose of the project is to design a helpful framework to help users get song recommendation. This project aims to find the correlation and similarity between music tastes and listening history. Song deems suitable by the algorithm to be like the user’s music taste are recommended to the user in a form of playlist.

1.5 Impact, Significance and Contribution

With the rapid development of mobile devices and more people having internet access, people have access to music collections at an unprecedented scale. Music libraries can easily have more than 15 million songs, people will feel overwhelmed choosing from the ocean of song available. Thus, an efficient song recommender system is necessary for the music service providers and customers. The music streaming companies can attract and retain users with a good recommender system. The problem of the previous project is huge dataset, but it will be separate to small subset for better results. Moreover, in the previous project will train different models to define best accuracy and prediction results. After the experiment, the collaborative filtering method will be used and trained for the project. Furthermore, the system will be deployed well-structured web-based system in this project which is FYP2. Users can easily listen song which is recommended by the system.

In the music recommender system field, many music streaming companies are working on building high-precision music recommender system [4]. Thus, this field have a high market demand for good quality music recommendation system.

1.6 Report Organization

Chapter 1: Introduction, talked about the problem statement of this project, background and the motivation are stated. In addition, the project objectives are used to solve the problem stated in the problem statement to achieve the improvement of the project. Moreover, the project scope will state how to propose the project and state the contribution of this project in the last part of Chapter 1.

Chapter 2: Literature Review is the research of the articles and existing system of this project and compute a summary of the functionality table.

Chapter 3: System Methodology or Approach, is about the methodology used in this project, the tools that were used for developing the system, the user requirement which is functional and nonfunctional requirements. Moreover, System Design also covered in this chapter. It will show the design of the whole system -System Flowchart, User Interface Design, Use Case Diagram and Activity Diagram. Verification plan used to verify for this project.

Furthermore, in Chapter 4: Experiment and Model Implementation, is about explain some comparison results and the model implementation.

Moreover, in Chapter 5: System implementation and testing are discussed about the work product and provide a testing plan for the system developed in this project.

Lastly, Chapter 6: Conclusion is about the summarize of this project such as research limit and future work.

CHAPTER 2: LITERATURE REVIEW

2.1 Music Recommender System CS365: Artificial Intelligence

In the study entitled Music Recommender System CS365: Artificial Intelligence [7] designed, applied and analysed a song recommendation system. The study is carried out by determining the users' listening patterns and learning from the users' listening history to provide suitable song recommendations according to user preferences.

The algorithms used for this study include the Popularity based Model, Collaborative based Model, SVD Model and KNN Model, and these algorithms are evaluated using Average Precision (mAP) to compare amongst themselves. The results of the SVD based latent factor model is generally better than the popularity-based model, however, the objective functions were prevented from converging to global optimum due to the sparse. On the other hand, the KNN did not perform well due to the limited song base of only 10,000 songs, which is less than 3 % of the whole dataset.

The results indicated that a memory-based collaborative filtering algorithm would be the most suitable for application in this situation. For the content-based model to work better, more memory and computational power are needed for the available metadata and training dataset to be utilized fully.

Popularity based Model

This model is the most basic and simple algorithm. In this algorithm, the popularity of each song is recorded and put into the training set and calculating the number of users who had listened to the song. Then sorted the song in descending order of their popularity. For new users, the system will recommend the topmost popular songs except those already in their profile. The disadvantage of the model is no personalization, and some songs may never get listened to.

SVD Model

SVD can collect the listening histories which are influenced by a set of factors (e.g. Genre, artist). These factors are not at all obvious in general, so we need to infer those called latent factors from the dataset. Users and songs will be characterized by latent factor.

KNN Model

KNN can be used to create a space of songs according to their features from the metadata and find out the neighborhood of each song. The example of features is loudness, mode genre and so more. After creating the feature space, look at all users' profiles and suggest songs which are neighbours to the songs present in his listening history. The KNN is quite personalized and uses metadata.

Evaluation Metrics

After using the four algorithms, the evaluation metrics can show the compare results with other algorithms. Furthermore, precision is more important than recalling because the false positives will lead to a poor user experience.

2.2 How to Build a Simple Song Recommender System

In this article [4], millions of songs are separated into 2 files called triplet file and meta file. The triplet file contains user_id, song_id and listen time. The metadata_file contains song_id, title, release_by and artist_name. The article is a helpful tutorial on using item similarity based collaborative filtering model to find similar songs to any songs in our dataset. The article discusses the different applications of 4 models to build the music recommendation system, namely content-based model, collaborative model, and popularity model. More models should be used in conducting this study for more accurate and precise results for example use K means clustering, Logistic Regression, Random Forests and so on.

Data preprocessing

Before taking the data to train and test for the algorithms require data pre-processing first.

```
triplets_file =
'https://static.turi.com/datasets/millionsong/10000.txt'

songs_metadata_file =
'https://static.turi.com/datasets/millionsong/song_data.csv'
```

Figure 2.2.1 Define file

This article [4] separates the metadata to 2 files which is triplet file and meta file. The figure shows to naming the 2 files from the links.

```
song_df_2 = pandas.read_csv(songs_metadata_file)

song_df = pandas.merge(song_df_1,
song_df_2.drop_duplicates(['song_id']), on="song_id", how="left")
```

Figure 2.2.2 Combine 2 files

CHAPTER 2 LITERATURE REVIEW

This figure shows to combine two files and drop duplicated data by using column `song_id`.

	<code>user_id</code>	<code>song_id</code>	<code>listen_count</code>	<code>title</code>	<code>release</code>	<code>artist_na</code>
0	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOAKIMP12A8C130995	1	The Cove	Thicker Than Water	Jack Johnson
1	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBBMDR12A8C13253B	2	Entre Dos Aguas	Flamenco Para Niños	Paco De Lucia
2	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBXHDL12A81C204C0	1	Stronger	Graduation	Kanye W
3	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBYHAJ12A6701BF1D	1	Constellations	In Between Dreams	Jack Johnson
4	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SODACBL12A8C13C273	1	Learn To Fly	There Is Nothing Left To Lose	Foo Fight

Figure 2.2.3 Visualize combined data

The diagram shows the combined data with song index, `user_id`, `song_id`, `listen_count`, `title`, `release`, and `artist_name`.

Data transformation

After preprocessing the data, the next step is data transformation. Doing transformation allows to simplify the dataset and make it easier and simpler to understand.

```
song_grouped = song_df.groupby(['song']).agg({'listen_count':  
'count'}).reset_index()  
grouped_sum = song_grouped['listen_count'].sum()  
song_grouped['percentage'] =  
song_grouped['listen_count'].div(grouped_sum)*100  
song_grouped.sort_values(['listen_count', 'song'], ascending = [0,1])
```

Figure 2.2.4 Data transformation

The first line of the code is grouping the `song_df` by the number of listen count in ascending. The second line is calculating sum of `listen_count` of each song into `grouped_sum` variable. The third line is create a new column name called `percentage` and calculate the value by dividing the `listen_count` by the sum of `listen_count` and multiply by 100. The last line is sorting the sequence and output in the ascending order.

	song	listen_count	percentage
3660	Sehr kosmisch - Harmonia	45	0.45
4678	Undo - Björk	32	0.32
5105	You're The One - Dwight Yoakam	32	0.32
1071	Dog Days Are Over (Radio Edit) - Florence + Th...	28	0.28
3655	Secrets - OneRepublic	28	0.28
4378	The Scientist - Coldplay	27	0.27
4712	Use Somebody - Kings Of Leon	27	0.27
3476	Revelry - Kings Of Leon	26	0.26

Figure 2.2.5 Output of data transformation

SVD Model

```

#constants defining the dimensions of our User Rating Matrix (URM)
MAX_PID = 4
MAX_UID = 5

#Compute SVD of the user ratings matrix

def computeSVD(urm, K):
    U, s, Vt = sparsesvd(urm, K)
    dim = (len(s), len(s))
    S = np.zeros(dim, dtype=np.float32)
    for i in range(0, len(s)):
        S[i,i] = mt.sqrt(s[i])
    U = csc_matrix(np.transpose(U), dtype=np.float32)
    S = csc_matrix(S, dtype=np.float32)
    Vt = csc_matrix(Vt, dtype=np.float32)
    return U, S, Vt

```

Figure 2.2.6 SVD model

This diagram shows the U represents user vector, S represents item vector. Vt is the joint of these U and S as collection of points in 2 dimensional spaces. The vectors are going to measure the distance of one preferences of user to another preferences of user.

```
#Compute estimated rating for the test user
def computeEstimatedRatings(urm, U, S, Vt, uTest, K, test):
    rightTerm = S*Vt

    estimatedRatings = np.zeros(shape=(MAX_UID, MAX_PID),
dtype=np.float16)
    for userTest in uTest:
        prod = U[userTest, :]*rightTerm
        #we convert the vector to dense format in order to get the
#indices
        #of the movies with the best estimated ratings
        estimatedRatings[userTest, :] = prod.todense()
        recom = (-estimatedRatings[userTest, :]).argsort()[:250]
    return recom
```

Figure 2.2.7 SVD code

The first line is used to define the `computeEstimatedRatings` for the test user. The following code is use for converting the vector to dense format to get the indices of the movie with best ratings.

Collaborative based Model

Collaborative based model can be classified by user based and item-based model. In this article using item similarity based model [4].

CHAPTER 2 LITERATURE REVIEW

```

#Create the item similarity based recommender system model
def create(self, train_data, user_id, item_id):
    self.train_data = train_data
    self.user_id = user_id
    self.item_id = item_id

#Use the item similarity based recommender system model to
#make recommendations
def recommend(self, user):

    #####
    #A. Get all unique songs for this user
    #####
    user_songs = self.get_user_items(user)

    print("No. of unique songs for the user: %d" % len(user_songs))

    #####
    #B. Get all unique items (songs) in the training data
    #####
    all_songs = self.get_all_items_train_data()

    print("no. of unique songs in the training set: %d" % len(all_songs))

    #####
    #C. Construct item cooccurrence matrix of size
    #len(user_songs) X len(songs)
    #####
    cooccurrence_matrix = self.construct_cooccurrence_matrix(user_songs, all_songs)

```

Figure 2.2.8 Item based model

This code shows it will get a unique count of user_id for each song and name it as recommendation score. The function then receives a user_id and use cooccurrence matrix to count the top 10 recommended songs. The recommendation function will show the top ten recommended song for given user_id.

	user_id	song	score	Rank
3194	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Sehr kosmisch - Harmonia	37	1
4083	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Undo - Björk	27	2
931	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Dog Days Are Over (Radio Edit) - Florence + Th...	24	3
4443	4bd88bfb25263a75bbdd467e74018f4ae570e5df	You're The One - Dwight Yoakam	24	4
3034	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Revelry - Kings Of Leon	21	5
3189	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Secrets - OneRepublic	21	6
4112	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Use Somebody - Kings Of Leon	21	7
1207	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Fireflies - Charttraxx Karaoke	20	8
1577	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Hey_ Soul Sister - Train	19	9
1626	4bd88bfb25263a75bbdd467e74018f4ae570e5df	Horn Concerto No. 4 in E flat K495: II. Romanc...	19	10

Figure 2.2.9 Top 10 recommend songs

2.3 Spotify

Spotify is a digital music, podcast, and video streaming service that gives you access to millions of songs and other content from artists all over the world. Spotify is available on both desktop and mobile applications, which is free of charge to download [8].

The free version of Spotify has simple functions such as playing music on shuffle and creating playlists. Users can search for songs with the Browse function, as Spotify has a wide range of genres for different occasions. If the users happen to come across a song that catches their ears, they can save the song into their playlists or simply set the song as one of their “Favourites”. Users can also choose to upgrade to Spotify Premium to avoid advertisements. The paid Spotify Premium would also allow you to play songs on-demand, download songs to listen to offline, and skip songs unlimitedly.

Furthermore, the other advantage of the Spotify is that the users would get daily and weekly song recommendations from personalized features, such as Discover Weekly, Release Radar, and Daily Mix. The system will collect the data from the users listening pattern to recommend some trending songs or songs from the same artist. Spotify’s song base is also multilingual as it has songs in Korean, Japanese, Chinese, English, and Bahasa Melayu.

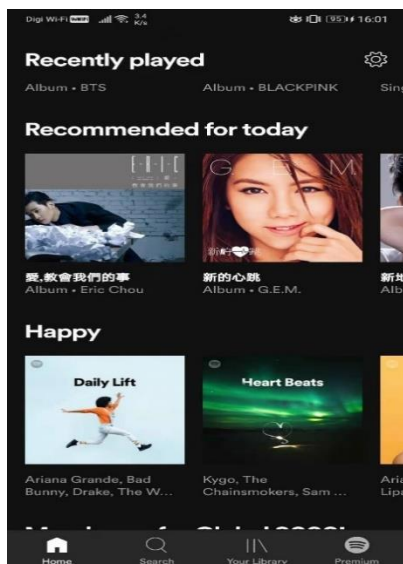


Figure 2.3.1 Recommended function by Spotify

Next, Spotify is available across a range of devices and platforms, including computers, tablets, phones, TVs, speakers, and car infotainment systems and you can easily transition from one to another with Spotify Connect. Users do not have to upgrade to premium to use this function, as it is available without any extra charges or fees.

CHAPTER 2 LITERATURE REVIEW

However, Spotify is not very user friendly to the users who choose to not upgrade to Spotify Premium. There would be ads every 30 minutes, which could be annoying for some users. Without premium, users are only allowed to play in shuffle mode instead of play specific song, and with 6 skips per hour, users may not be able to listen to one specific song. Without Premium, offline listening is unavailable, which means users must have access to the internet to listen to music. The audio quality accessible by free users is also lower than those who have premium.

Lastly, some songs of other languages are not available on Spotify, for instance, Spotify would have a complete collection of English albums, but only popular or trending Chinese and Malay songs. Some songs are also only accessible for Premium users.

2.4 Apple Music

Apple Music is a music and video streaming service developed by Apple Inc. Users select music to stream to their device on-demand, or they can listen to existing playlists. Apple Music is available for IOS users only. IOS users can download Apple music from the App Store on their Apple devices.

Apple music support song sharing features. It is a very good function to allow users to share their favourite song with friends and to social media such as Instagram. This social function enables the promotion of songs as well as the application itself [8].

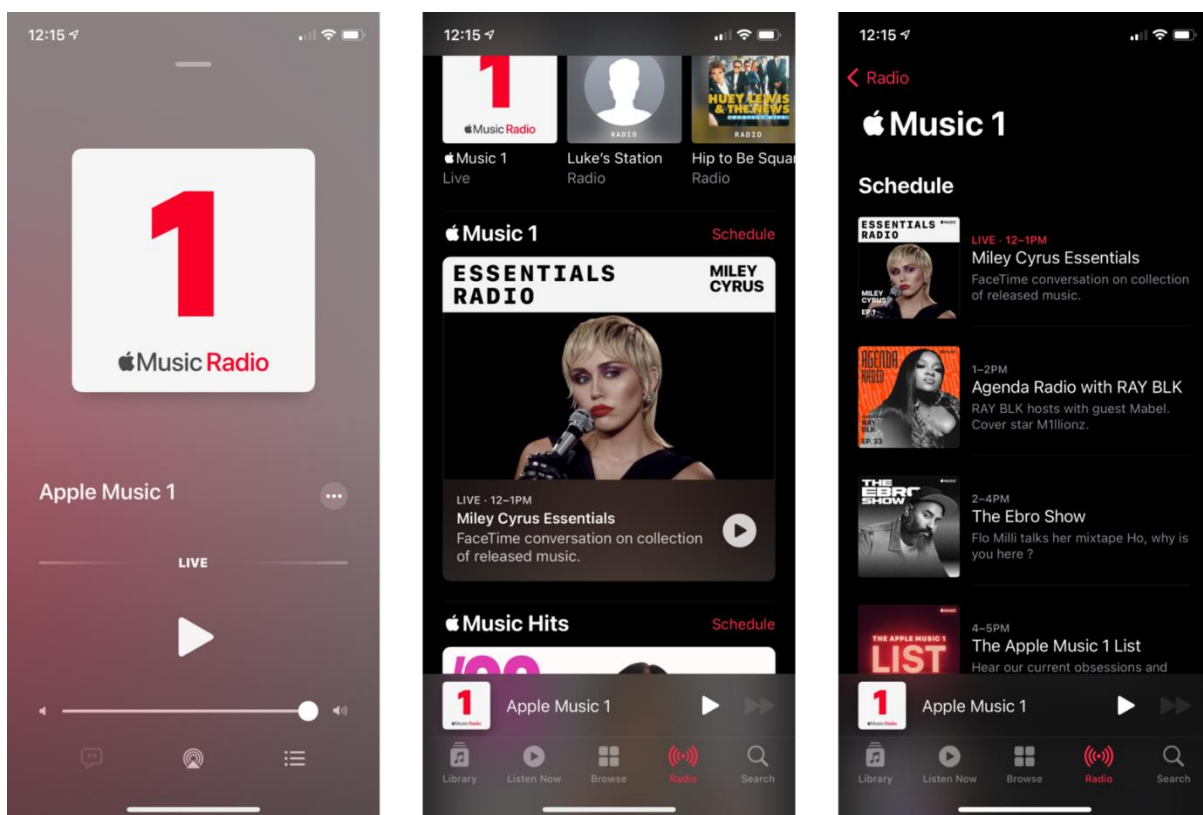


Figure 2.4.1 Apple music interface

The next strength of apple music is that it has more music content libraries compared to other music recommendation applications. According to the article [8], Apple music has over 60 million songs, users can find music from the artists they have yet to discover. Apple Music also makes exclusive deals with artists like Taylor Swift and Drake for early content release, which benefits the anticipating fans. The exclusive content is enabled by Apple's paid-only service, and users are required to pay for the subscription.

CHAPTER 2 LITERATURE REVIEW

Moreover, Apple Music has one more strong function which is users can search for songs using only lyrics. This feature is very useful to some users who want to listen to a song but can only remember the lyrics and not the title. Now users can type the lyrics directly into the app's search bar, the system will match the songs to the users accordingly.

On the downside, Apple music does not have an organized and clean user interface. The Apple Music UI does not provide instant access to your own content. The home page should show your recently played songs and playlists, which is what most people want to see when opening a music app. Every part of Apple Music's design should consider putting users' content at the forefront, followed by providing users with suggestions after users have seen what users wanted to see.

Finally, Apple Music should provide its own voice assistance functionality for the app without Siri. Apple Music can control the next, skip, pause song by using voice assistance functionalities.

2.5 Last Fm

Last Fm can use to track the music you stream by connecting Last.fm to a music service, app, or a browser plugin. Last Fm also provides users to view user's stats in real time, receive the weekly reports and access user listening history and so on. Last Fm also allows users to connect streaming services for example Spotify, Imusic, YouTube, Tidal and SOUNDCLOUD.

The one of the strengths of Last Fm is to provide free downloadable music tracks when applicable. Some applications will limit users to download songs without any subscription fees. In Last Fm, users can easily download the song for iOS and android users.

Moreover, another strong feature is being unable to find an artist or song in Last Fm's library, the system will automatically redirect to the YouTube link of the song.

On the downside, Last Fm's recommendation engine is not too strong because the engine is based on user input, not the back-end data. It will cause limitation of the recommendation system to the users because of a lack of data set.

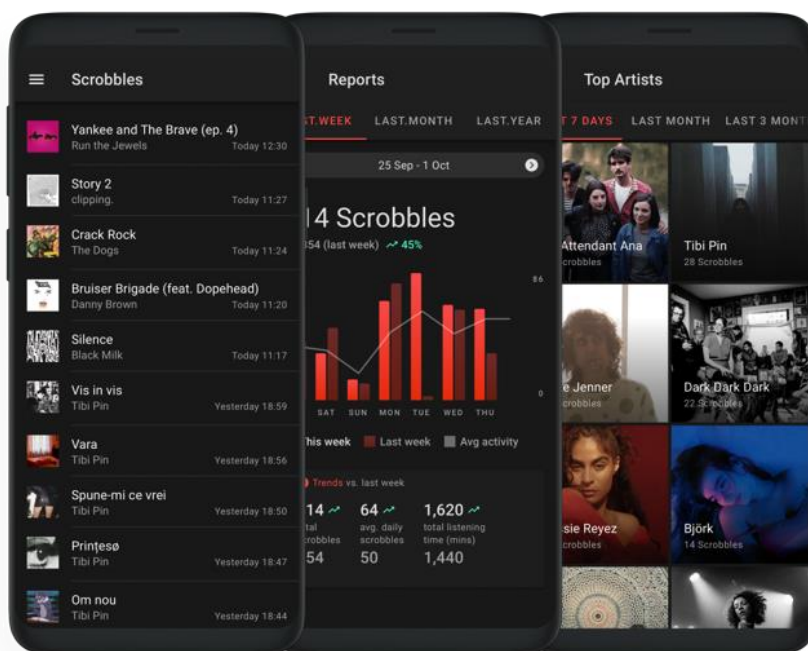


Figure 2.5.1 Last FM Interface

CHAPTER 2 LITERATURE REVIEW

	Spotify	Apple Music	Last FM
Platform	IOS and Android	IOS only	Android only
User interface	Friendly	Not friendly	Not friendly
Song Recommender system	√	√	×
Music library	Normal	Many	Less
Free of charge	Yes	No	Yes
Advertisement	No if subscribes premium package	No	Yes
Free download songs	√	√	√
Own service	Spotify Connect	Siri	No

Table 2.5.1 Comparison of Existing System

Chapter 3: System Methodology/Approach

3.1 Design Specifications

3.1.1 Methodologies and General Work Procedures

Methodologies

First, the project needs a dataset to train algorithms to build an efficient recommendation system. This recommendation system requires users to install Anaconda which is a popular Python language to run the python script. We will use different algorithms to train and test the dataset to calculate the accuracy. After the release of the result, choose the model with the highest accuracy and precision for the music recommendation system. Moreover, the project will develop by Visual Studio Code, so the python scripts need to be zipped to a pickle file so that it can be used in our program. Furthermore, this system also requires building a database to store the user's record or history.

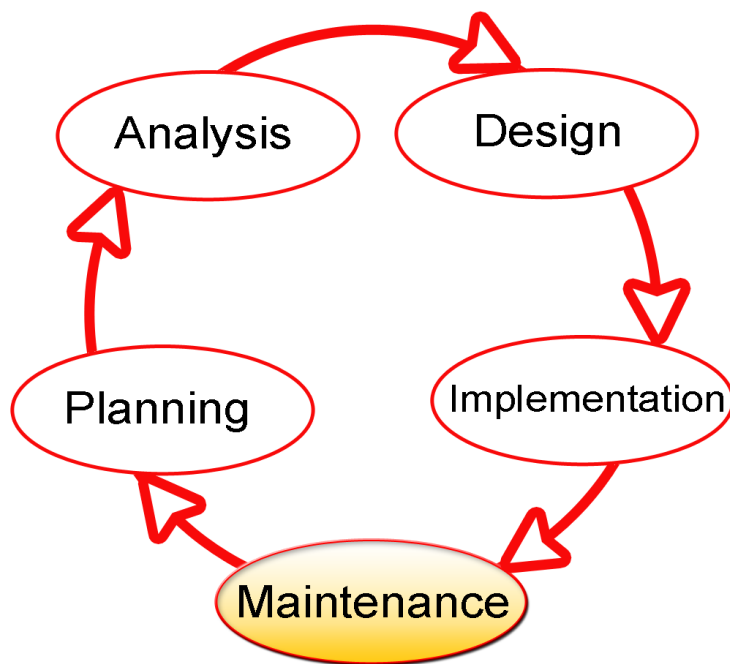


Figure 3.1.1.1 System Development Life Cycle (SDLC)

The methodology that will be used in this proposal is System Development Life Cycle (SDLC). SDLC consists of 5 phases which are Planning, Analysis, Design, Implementation and Maintenance (SDLC Overview, 2019). **Planning phase** is the fundamental process of understanding why a system should be built and produce ideas how the project will go about building it. During **analysis phase**, it will analyze the questions of who will use the system, what the system will do and where it will be used. On this phase, it will investigate any current systems, identify improvement opportunities, and develop a concept for the new system. The next phase is **design**, the design phase will decide how the system will operate, the user interface, forms, reports, specific programs, database, and files that will be needed. Moreover, the next phase is **implementation**. During this phase, the system is built and usually gets the most attention because for most systems it is the longest and most expensive single part of the development phase. Finally, the final phase of SDLC is **maintenance**. Developers need to provide upkeep to the system, need to perform maintenance activities and monitor system performance to make sure the system is always runnable [9].

Dataset

There are 2 datasets in this project. Each dataset is downloaded from Kaggle. First dataset call songs.csv includes metadata for example user_id, song_id, listen_count, release, artist_name and year. The dataset is under modified to get better train results.

	A	B	C	D	E	F	G	H	I	J
1	user_id	song_id	listen_count	title	release	artist_name	year			song
2	b80344d0fSOAKIMP:	1	The Cove	Thicker Than Water	Jack Johns		0 0			
3	b80344d0fSOBBMDR	2	Entre Dos	Flamenco	Paco De Lucia		1976	0		
4	b80344d0fSOBHDLJ	1	Stronger	Graduatio	Kanye We		2007	0		
5	b80344d0fSOBYHAJ1	1	Constellat	In Between	Jack Johns		2005	0		
6	b80344d0fSODACBLJ	1	Learn To F	There Is N	Foo Fighte		1999	0		
7	b80344d0fSODDNQT	5	Apuesta P	Antologã	Hã@roes		2007	0		
8	b80344d0fSODXRTY1	1	Paper Gar	The Fame	Lady GaGã		2008	0		
9	b80344d0fSOXRGUAY:	1	Stacked A	There Is N	Foo Fighte		1999	0		
10	b80344d0fSOFRQTD:	1	Sehr kosn	Musik vor	Harmonia		0 0			
11	b80344d0fSOHQWY2	1	Heaven's	JHãtel Co	Theftery C		2002	0		
12	b80344d0fSOIYTOA1	1	Let It Be	S If I Had	Ey Jack Johns		2007	0		
13	b80344d0fSOIZAZL1:	5	I'll Be	Mis: No Way	O Puff Dadd		0 0			
14	b80344d0fSOJNNUA	1	Love Shac	Original H	The B-52's		1989	0		
15	b80344d0fSOJPFQJ	1	Clarity	As/Is: Clef	John May		0 0			
16	b80344d0fSOXRMJP1	5	P'n A Ste	Digger!	De Robert Jol		0 0			
17	b80344d0fSOJLGNJ	1	The Old	S: incrediba	The Lone		2009	0		
18	b80344d0fSOMGIYR1	6	Behind Th	Live In Chi	Panic At T		0 0			
19	b80344d0fSOMLMK1:	1	Champion	Graduatio	Kanye We		2007	0		
20	b80344d0fSOMSQJY:	1	Breakout	There Is N	Foo Fighte		1999	0		
21	b80344d0fSONSAEZ1	1	Ragged W	Fleet Foxe	Fleet Foxe		2008	0		
22	b80344d0fSODKGRB	1	Mykonos	Sun Giant	Fleet Foxe		2008	0		
23	b80344d0fSOQCVGE	1	Country R	En Concer	Jack Johns		2009	0		
24	b80344d0fSOQJVUD:	1	Oh No	Noble Beã	Andrew B		2009	0		
25	b80344d0fSOQJLDY1	1	Love Song	Room For	John May		0 0			

Figure 3.1.1.2 songs.csv file

The second dataset named videosongs.csv which is self-create and record down the YouTube links. After that, find correlations between the songs and users from the dataset. Both datasets will be merged and used to calculate the similarity between songs listened by the user and all unique songs for the recommender system [10].

	A	B	C	D	E	F	G	H	I
1		Link							
2	0	https://www.youtube.com/watch?v=fm115y212ry							
3	1	https://www.youtube.com/watch?v=2oyhlad64-s							
4	2	https://www.youtube.com/watch?v=PsO6ZnUZi0g							
5	3	https://www.youtube.com/watch?v=1sZ15UeS9w							
6	4	https://www.youtube.com/watch?v=1VQ_3sBZE0M							
7	5	https://www.youtube.com/watch?v=6Tju36z2EcQ							
8	6	https://www.youtube.com/watch?v=RP3zMJ0AKIg							
9	7	https://www.youtube.com/watch?v=kqmTxaSrnio							
10	8	https://www.youtube.com/watch?v=P0hFr6i8y9c							
11	9	https://www.youtube.com/watch?v=qyYMHfPGYDY							
12	10	https://www.youtube.com/watch?v=NXI9fTKNnT0							
13	11	https://www.youtube.com/watch?v=NKMtZm2Yube							
14	12	https://www.youtube.com/watch?v=9SOrYjVTAGs							
15	13	https://www.youtube.com/watch?v=wpQ_R85VFJc							
16	14	https://www.youtube.com/watch?v=OxXczxbTbdQ							
17	15	https://www.youtube.com/watch?v=vXkalE0STPM							
18	16	https://www.youtube.com/watch?v=R40vaJwBX-0							
19	17	https://www.youtube.com/watch?v=wcDgUiy-IJY							
20	18	https://www.youtube.com/watch?v=IIOJdMdS56k							
21	19	https://www.youtube.com/watch?v=a2KO18Lt2Lc							
22	20	https://www.youtube.com/watch?v=DT-dxG4WWF4							
23	21	https://www.youtube.com/watch?v=1vrEljMfXYo							
24	22	https://www.youtube.com/watch?v=fXLiic00Crvk							
25	23	https://www.youtube.com/watch?v=zxkRSMmX4w							

Figure 3.1.1.3 videosongs.csv file

3.1.2 Tools to use

Programming language

The project will use Python to develop. Moreover, Python focuses on code readability. The language is adaptable, tidy, simple to learn and use, readable, and well-structured. From web development to game development to machine learning, there is a library for almost anything you can think of. Python, as previously stated, is simple to understand and quick to develop with. Python allows you to do more with less code, which means you can create prototypes and test concepts faster than with other languages. This means that Python not only saves a lot of time [11][12].

Visual Studio Code

The project will be developed on the Visual Studio Code platform. As Visual Studio Code supports Python, it is very user-friendly for developing projects. Additionally, .CSS, JSON, and HTML are required to implement this project. Furthermore, the platform console allows you to host private addresses to run the project. In comparison with Jupyter Notebook used in FYP1, this platform's debugging will be simpler.

Collaborative based model

In this project, collaborative based model is used because collaborative method can be either user-based or item-based. Then stochastic aggregation is used to combine them. Stochastic aggregation is randomly choosing one of them according to top scored items and their probability distribution. The reason for not choosing other models is because collaborative filtering method is most suitable for the recommender system.

The collaborative-based model can collect the information from many users and then make a prediction based on similarity measures between items and users. Which is classified into item-based and user-based models.

In the item-based model, it will collect the songs that are often listened to together by some users tend to be similar and some users are more likely to be listened together in the future.

In the user-based model, it will collect users who have similar listening histories. Some users will probably listen to the same songs in future because some songs have listened to in the past.

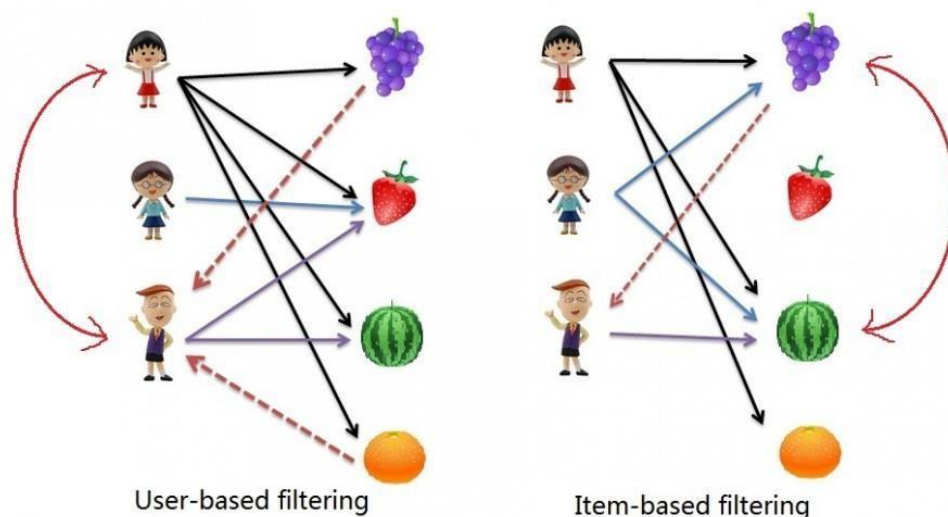


Figure 3.1.2.1 User and Item Based Filtering

Flask

Flask is a web framework and a Python module that allows you to easily create web applications. It has a tiny and easy-to-extend core: it's a microframework without an ORM (Object Relational Manager) or similar functionality. It has a lot of amazing features, such as url routing and a template engine. A Web Application Framework, often known as a Web Framework, is a set of tools and modules that allow web application developers to construct applications without having to worry about low-level issues such as protocol, thread management, and so on. Flask is also known as a microframework. It is intended to keep the application's core simple and scalable. Moreover, Flask allows extensions to be added to the application to provide such features [13].

Heroku

Heroku is a cloud Platform as a Service that is container-based (PaaS). Heroku is used by developers to launch, manage, and grow contemporary programmers. Heroku platform is sleek, adaptable, and simple to use, providing developers with the quickest path to market for their products. Heroku is entirely managed, allowing developers to focus on their core product without having to worry about servers, hardware, or infrastructure. The Heroku experience includes services, tools, processes, and polyglot support, all of which are intended to boost developer productivity [14].

Firebase

Firebase is a Backend-as-a-Service provider (Baas). In research from the articles [15], it offers several tools and services to developers to help them create high-quality apps, expand their user base, and profit. It is based on the infrastructure of Google. Firebase is a NoSQL database application. It accepts passwords, phone numbers, Google, Facebook, Twitter, and other methods of authentication. The Firebase Authentication (SDK) may be used to integrate one or more sign-in methods into a project manually. Moreover, Firebase is a real-time database which is suitable for our project. Firebase used for saved the users registration info such as email, password, and the song history.

In a nutshell, the hardware and the software used for the project is shown below:

1. Hardware Involved

- Laptop

Description	Specification
Model	Inspiron 5502
Processor	11 th Gen Intel Core i7-1165G7 CPU@2.80GHz
Operating System	Windows 10 64-bit
Memory	16GB DDR4 RAM
Storage	NVMe IM2P33F3A NVMe ADATA 512GB

Table 3.1.2.2 Specification of laptop

2. Software Involved:

- Jupyter Notebook
- Python
- Heroku
- Firebase
- Flask



Figure 3.1.2.3 Software Involved

3.1.3 User Requirements

Functional Requirements

The functional requirement of the project is mainly categorized as user requirements.

i. User Requirement: User have account on this system and client must have somewhere around listened one song and to investigate the identity for the music suggestion.

Non-Functional Requirements

The non-functional requirements of the project are security requirement and device requirement.

i. Performance: The system should have a speedy, exact, and dependable outcomes.

ii. Availability: The system will be accessible to client or users whenever at whatever point there is an Internet association.

iii. Capacity and Scalability: The system will record the user's song history into database.

iv. Flexibility and Portability: System should be available from any areas.

3.1.4 System Performance Definition

The performance of the system will be evaluated on accuracy and prediction of the songs to the users. Moreover, the system must be able to accurately accomplish the functions that its users have asked. The song should be play immediately for the users which is suggested by the system. Furthermore, the system also provides login function to keep user's history of the songs. User also allow to refresh the song list without repeated songs as well. Finally, users also allow to add ore remove favorite for the song. The recommended songs should be fast and correct to listen to.

3.2 System Design/Overview

3.2.1 System Flowchart

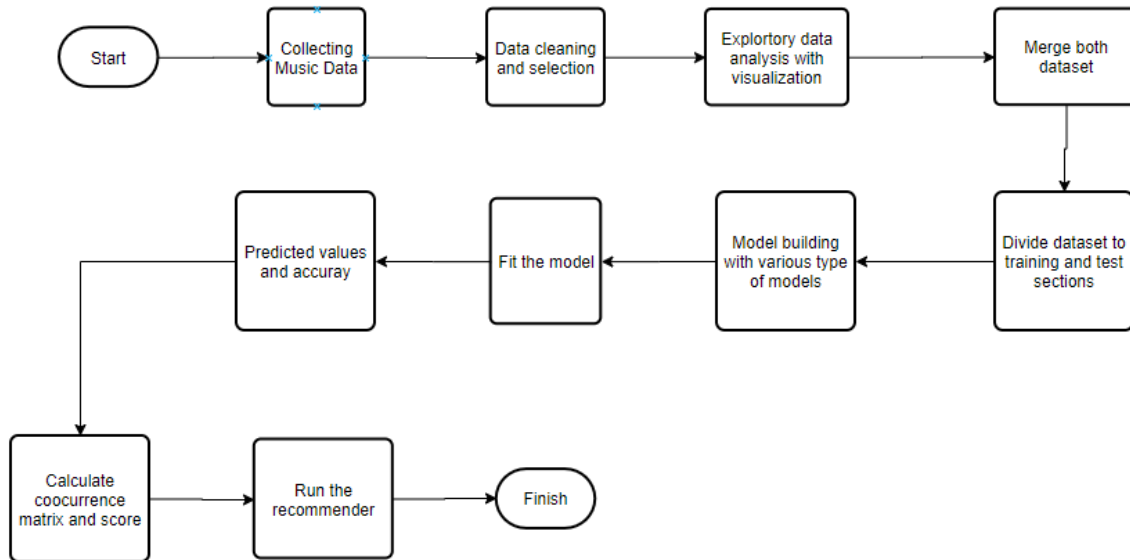


Figure 3.2.1.1 System Flowchart

In Figure 3.2.1.1 as the system flowchart, the first step is collecting dataset and clean the dataset. This step call data pre-processing. Data pre-processing is a very important step such as remove null values and duplicate records.

CHAPTER 3 SYSTEM METHODOLOGY/APPROACH

```
In [11]: song_df1
```

```
Out[11]:
```

	user_id	song_id	listen_count	title	release	artist_name	year	song	video_id
0	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOAKIMP12A8C130995	1	The Cove	Thicker Than Water	Jack Johnson	0	0 The Cove	0
1	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBBMDR12A8C13253B	2	Entre Dos Aguas	Flamenco Para Niños	Paco De Lucia	1976	0 The Cove	1
2	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBXHDL12A8C1204C0	1	Stronger	Graduation	Kanye West	2007	0 The Cove	2
3	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBYHAJ12A6701BF1D	1	Constellations	In Between Dreams	Jack Johnson	2005	0 The Cove	3
4	b80344d063b5ccb3212f76538f3d9e43d87dca9e	SODACBL12A8C13C273	1	Learn To Fly	There Is Nothing Left To Lose	Foo Fighters	1999	0 The Cove	4
...
2495	9fba771d9731561eba47216f6fbc0023d88641b	SOUNNWW12AB018795D	2	Opposite Of Adults	Now That's What I Call Music! 75	Chiddy Bang	2009	0 The Cove	2495
2496	9fba771d9731561eba47216f6fbc0023d88641b	SOURBIU12AC3DF683C	1	Sally Dog	Live at the Greek Theatre	Flogging Molly	1999	0 The Cove	2496
2497	9fba771d9731561eba47216f6fbc0023d88641b	SOVAMBN12AB0187FEF	1	Dingue_ Dingue_ Dingue	On Trace La Route	Christophe Maé	2010	0 The Cove	2497
2498	9fba771d9731561eba47216f6fbc0023d88641b	SOVDLZN12AB0185BEA	3	Tighten Up	Tighten Up	The Black Keys	2010	0 The Cove	2498
2499	9fba771d9731561eba47216f6fbc0023d88641b	SOVGKPF12A8AE45594	3	Come On Eileen	Kids Sing Along Pac	Dexys Midnight Runners	1982	0 The Cove	2499

2500 rows x 9 columns

Figure 3.2.1.2 Dataset of songs.csv file

```
In [12]: song_df2
```

```
Out[12]:
```

	video_id	Link
0	0	https://www.youtube.com/watch?v=fm1S2i2rY
1	1	https://www.youtube.com/watch?v=2oyhlad64-s
2	2	https://www.youtube.com/watch?v=PsO6ZnUZl0g
3	3	https://www.youtube.com/watch?v=1sZ15SueS9w
4	4	https://www.youtube.com/watch?v=1VQ_3sBZEm0
...
2495	2495	https://www.youtube.com/watch?v=s3dni9BYAtg
2496	2496	https://www.youtube.com/watch?v=1rDbav58XcU
2497	2497	https://www.youtube.com/watch?v=Rwv1djaV6Fc
2498	2498	https://www.youtube.com/watch?v=bQPW2MuEPkg
2499	2499	https://www.youtube.com/watch?v=GbpnAGajyMc

2500 rows x 2 columns

Figure 3.2.1.3 Dataset of videosongs.csv file

Data Cleaning

```
In [23]: for col in song_df1.columns:
          print(col , song_df1[col].isnull().sum())

user_id 0
song_id  0
listen_count 0
title 0
release 0
artist_name 0
year 0
```

```
In [24]: for col in song_df2.columns:
          print(col , song_df2[col].isnull().sum())

video_id 0
```

Figure 3.2.1.4 Null values of songs.csv Figure 3.2.1.5 Null values of videosongs.csv

From the Figure 3.2.1.4 and 3.2.1.5, both datasets do not have any null values.

Checking Duplicate Records

```
# Duplicate records
```

```
In [26]: song_df1.duplicated(subset=None,keep='first').sum()
Out[26]: 0
```

```
In [27]: song_df2.duplicated(subset=None,keep='first').sum()
Out[27]: 0
```

Figure 3.2.1.6 Checking duplicate records for the 2 datasets

The duplicate records for the two datasets are zero, it is good for our system because all data are unique.

Exploratory Data Analysis with Visualization

After cleaning the data, next step is exploratory data analysis with visualization. The purpose of the visualization is simplifying and easily understand the data in diagram or graph form. The following visualization will be shown below.

Total Number of Songs Sing by Each Artist

```
In [20]: song_df1['artist_name'].value_counts().to_frame()
```

Out[20]:

	artist_name
	Daft Punk
	82
	Coldplay
	63
	Cut Copy
	34
	The Black Keys
	33
	Kings Of Leon
	32
	...
	Spiderbait
	1
	Montell Jordan
	1
	Barry Manilow
	1
	Scottish Chamber Orchestra/Jukka-Pekka Saraste
	1
	João Pedro Pais
	1

817 rows x 1 columns

Figure 3.2.1.7 Total Number of Songs Sing by Each Artist

Total Number of Songs Sing in Each Year

```
In [32]: song_df1['year'].value_counts().to_frame()
```

Out[32]:

	year
	0
	384
	2007
	255
	2009
	251
	2008
	247
	2005
	171
	2006
	149
	2003
	108
	2010
	102
	2004
	101
	2002
	101
	2001
	100
	2000
	72
	1996
	44
	1999
	36
	1997
	34
	1998
	26
	1995
	22
	1994
	21
	1992
	19
	1990
	18
	1991
	18
	1993
	17
	1988
	16
	1987
	16

1987	16
1985	14
1976	13
1986	12
1979	11
1982	11
1983	9
1969	9
1981	9
1974	9
1968	7
1970	7
1965	6
1973	6
1978	6
1977	5
1967	5
1975	5
1984	5
1980	4
1972	3
1989	3
1963	3
1964	2
1962	2
1971	2
1954	1
1966	1
1956	1
1960	1

Figure 3.2.1.8 Total no of Songs Sing in Each Year

```
In [33]: sns.countplot(y="year",data=song_df1,orient="v")
```

```
Out[33]: <AxesSubplot:xlabel='count', ylabel='year'>
```

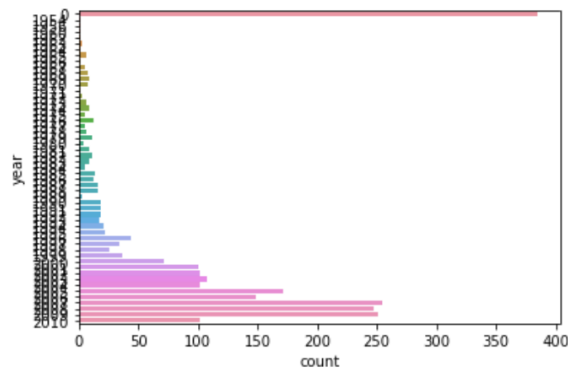


Figure 3.2.1.9 Visualizing Songs by Each Artist with Respect to Year

From the Figure 3.2.1.9, the reason of the results is the dataset too large so the information will be compressed by the count plot.

Merge two dataset

In this stage, we need to merge both dataset to one data frame with using video_id to connect each other. Next, we will create a data frame which will be a subset of the given dataset. Moreover, merge song_title and artist_name columns to make a new column. Last but not least, the column listen_count denotes the number of times the song has been listened. Using the column, we will find the data frame consisting of popular songs.

```
In [15]: song_gr = song_df1.groupby(['song', 'Link']).agg({'listen_count': 'count'}).reset_index()
grouped_sum = song_gr['listen_count'].sum()
song_gr['percentage'] = song_gr['listen_count'].div(grouped_sum)*100
song_gr.sort_values(['listen_count', 'song'], ascending = [0,1])
```

```
Out[15]:
```

	song	Link	listen_count	percentage
1317	Sehr kosmisch - Harmonia - https://www.youtube...	https://www.youtube.com/watch?v=P0hFr6i8y9c	13	0.529316
1715	Use Somebody - Kings Of Leon - https://www.you...	https://www.youtube.com/watch?v=gnhXHvRoUd0	9	0.366450
935	Lucky (Album Version) - Jason Mraz & Colbie Ca...	https://www.youtube.com/watch?v=D2f6iNmmd1M	8	0.325733
1850	You're The One - Dwight Yoakam - https://www.y...	https://www.youtube.com/watch?v=IsVVRi0Tg1w	8	0.325733
363	Dog Days Are Over (Radio Edit) - Florence + Th...	https://www.youtube.com/watch?v=cIXQjSo4GCE	7	0.285016
...
1866	high fives - Four Tet - https://www.youtube.co...	https://www.youtube.com/watch?v=z6A2LHGx8_A	1	0.040717
1867	re.stacks - Bon Iver - https://www.youtube.com...	https://www.youtube.com/watch?v=GhDnyPsQsB0	1	0.040717
1868	sleep_eat food_have visions - Four Tet - htt...	https://www.youtube.com/watch?v=HPHhTZeg_UM	1	0.040717
1869	sun drums and soil - Four Tet - https://www.yo...	https://www.youtube.com/watch?v=bPZ-jgYC5JA	1	0.040717
1870	you were there with me - Four Tet - https://ww...	https://www.youtube.com/watch?v=6pztaxT-IXc	1	0.040717

```
1871 rows x 4 columns
```

Figure 3.2.1.10 Merged dataframe and the Percentage of the listen_count

Divide Dataset to Training and Test Sections

The next step is splitting the dataset to training and test set. In this project, there will be 20% for testing the model and 80% for training the model. The reason of using train-test split evaluation is because the train set will be used to fit the machine learning model while the test set used to evaluate the fit machine learning model [10]

```
In [17]: train, test_data = train_test_split(song_df1, test_size = 0.20, random_state=0)
```

Figure 3.2.1.11 Dividing Dataset to Test Set

3.2.2 User Interface Design

Visual Paradigm Online Free Edition

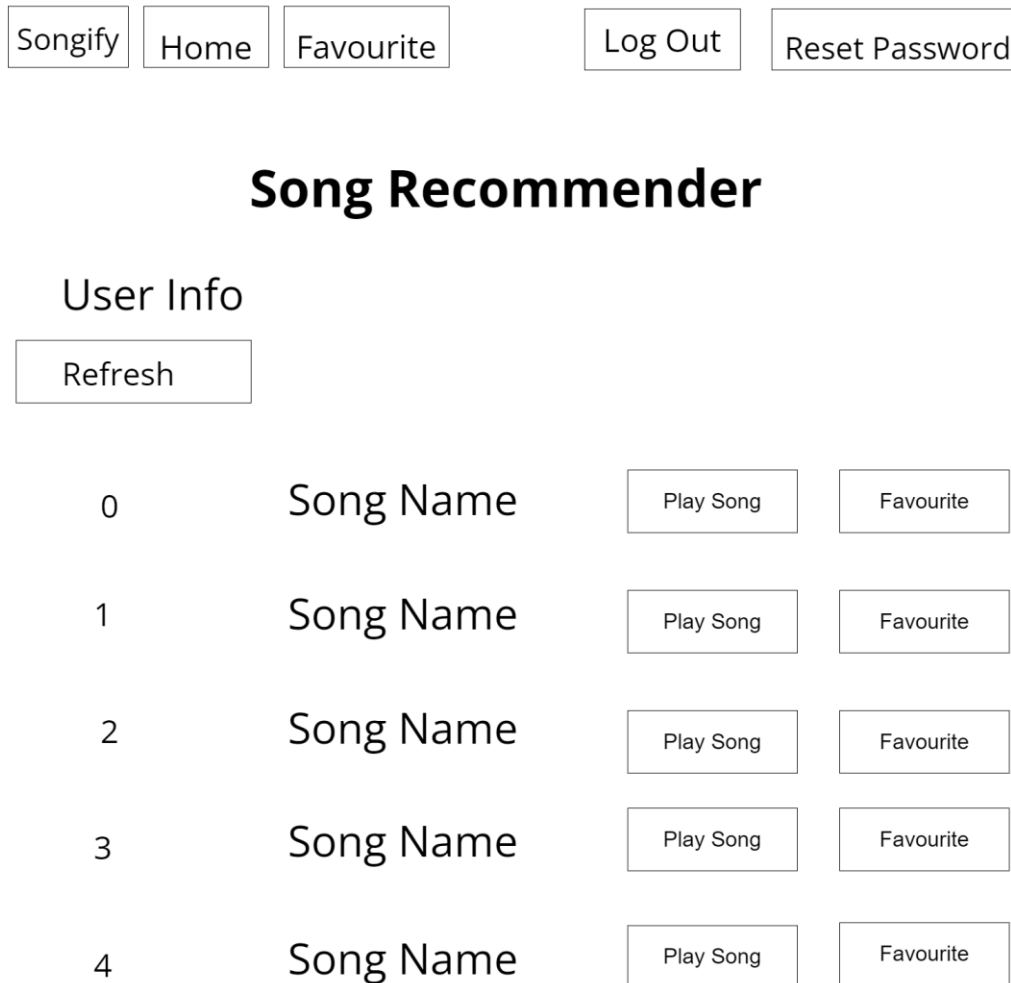


Figure 3.2.2.1 User Interface Design for Recommender System

From the *Figure 3.2.2.1*, the design of the user interface will be clean and simple. The title of the system will be put on center. Next, there will be got Home, Favourite, Log Out and Reset Password function on the top bar. Moreover, there will be one refresh button to refresh the recommended songs for the users. While in the bottom part, there will be the recommended songs with Play and add to Favourite button.

3.2.3 Use Case Diagram

The use case diagram of the song recommender system will be shown as follow:

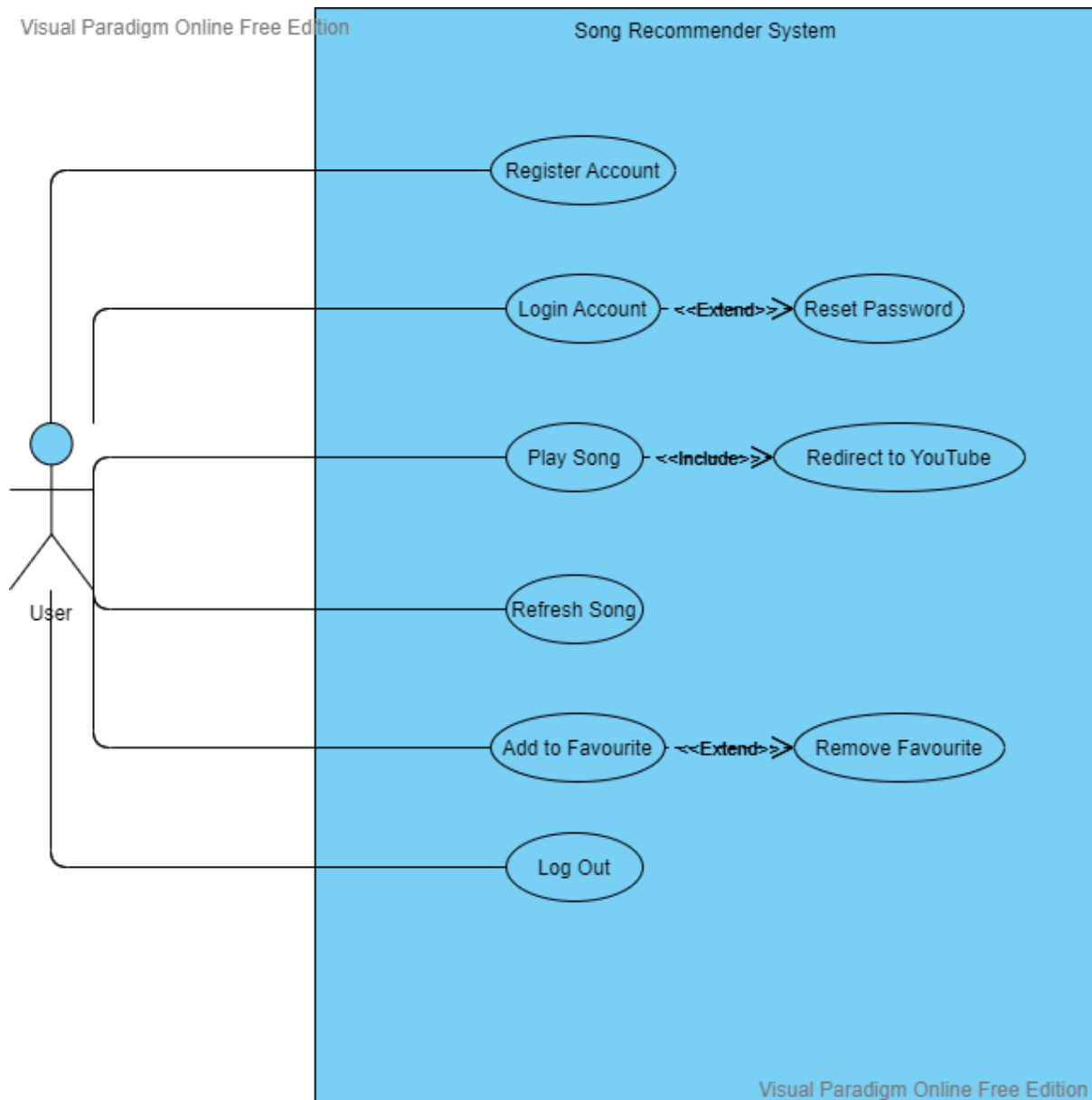


Figure 3.2.3.1 Use Case Diagram

3.2.4 Use Case Description

Register Account

Use Case Name: Register Account	ID: 1	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to register account.		
Brief Description: This use case describes the process of register account as a user.		
Trigger: User wants to register account.		
Type: -		
Relationships: Association: - Include: - Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The user enters username. 2. The user enters password with at least length of 6. 3. The user enters email with correct format. E.g @gmail.com. 4. User register account successfully. 		
SubFlows: -		
Alternate/Exceptional Flows:		

Table 3.2.4.1 Use Case Description (Register Account)

Login Account

Use Case Name: Login Account	ID: 2	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to login account.		
Brief Description: This use case describes the process of login account as a user.		
Trigger: User wants to login account.		
Type: -		
Relationships: Association: - Include: - Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The user enters correct email and password. 2. System verifies the email and password. 3. User login successful. 		
Alternate Flow-Invalid Email and Password: <ol style="list-style-type: none"> 1.1 User inputs invalid email or password. 1.2 User can choose forgot password option. 1.3 System will send the links to the registered email for reset password. 		

Table 3.2.4.2 Use Case Description (Login Account)

Play Song

Use Case Name: Play song	ID: 3	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to play song.		
Brief Description: This use case describes the process of play song.		
Trigger: User wants to play song.		
Type: -		
Relationships: Association: - Include: YouTube Extend: - Generalization: -		
Normal Flow of Events: 1. The user presses the Play button. 2. The system will redirect the user to the song of the YouTube video.		
SubFlows: -		
Alternate/Exceptional Flows: -		

Table 3.2.4.3 Use Case Description (Play song)

Refresh Songs

Use Case Name: Refresh songs	ID: 4	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to refresh the songs.		
Brief Description: This use case describes the process of refresh songs.		
Trigger: User wants to refresh the songs.		
Type: -		
Relationships: Association: - Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. The user presses the refresh button. 2. The system displays new and non-repeated 10 recommend songs and to user.		
SubFlows: -		
Alternate/Exceptional Flows: -		

Table 3.2.4.4 Use Case Description (Refresh songs)

Add to Favourite

Use Case Name: Add to Favourite	ID: 5	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to add the song to favourite.		
Brief Description: This use case describes the process of add song to favourite.		
Trigger: User wants to add song to favourite.		
Type: -		
Relationships: Association: - Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. The user presses the favourite button. 2. The system will add and save the song in favourite menu.		
SubFlows-Remove Favourite: 1. The user removes the song from the favourite lists. 2. The system will delete the song from the favourite list.		

Table 3.2.4.5 Use Case Description (Add to Favourite)

Log Out

Use Case Name: Log Out	ID: 6	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wants to log out the system.		
Brief Description: This use case describes the process of log out.		
Trigger: User wants to log out the system.		
Type: -		
Relationships: Association: - Include: - Extend: - Generalization: -		
Normal Flow of Events: 1. The user presses the Log out button. 2. The system will back to the login menu.		
SubFlows-		

Table 3.2.4.6 Use Case Description (Log Out)

3.2.5 Activity Diagram

Register Account

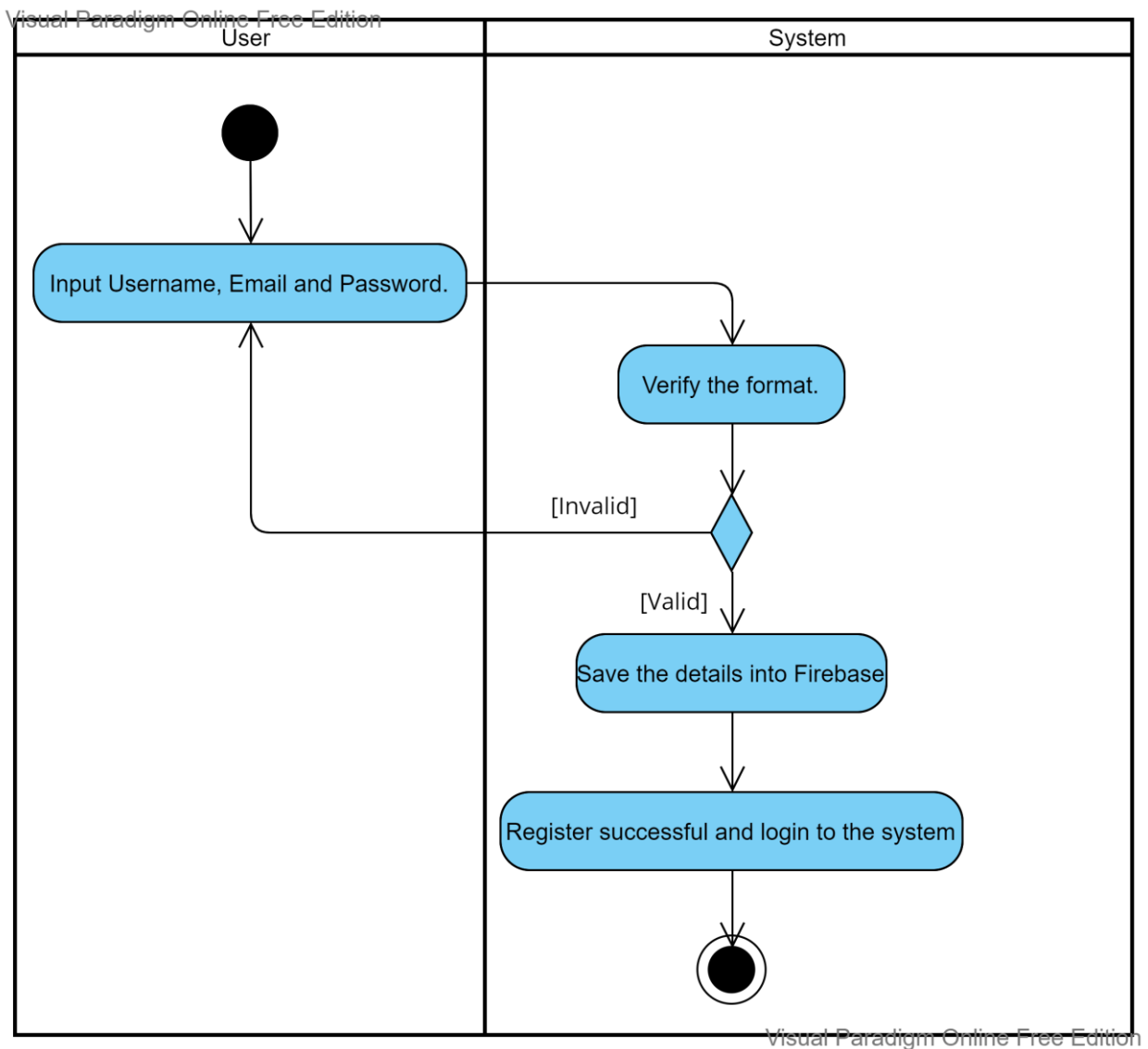


Figure 3.2.5.1 Activity Diagram of Register Account

Login Account

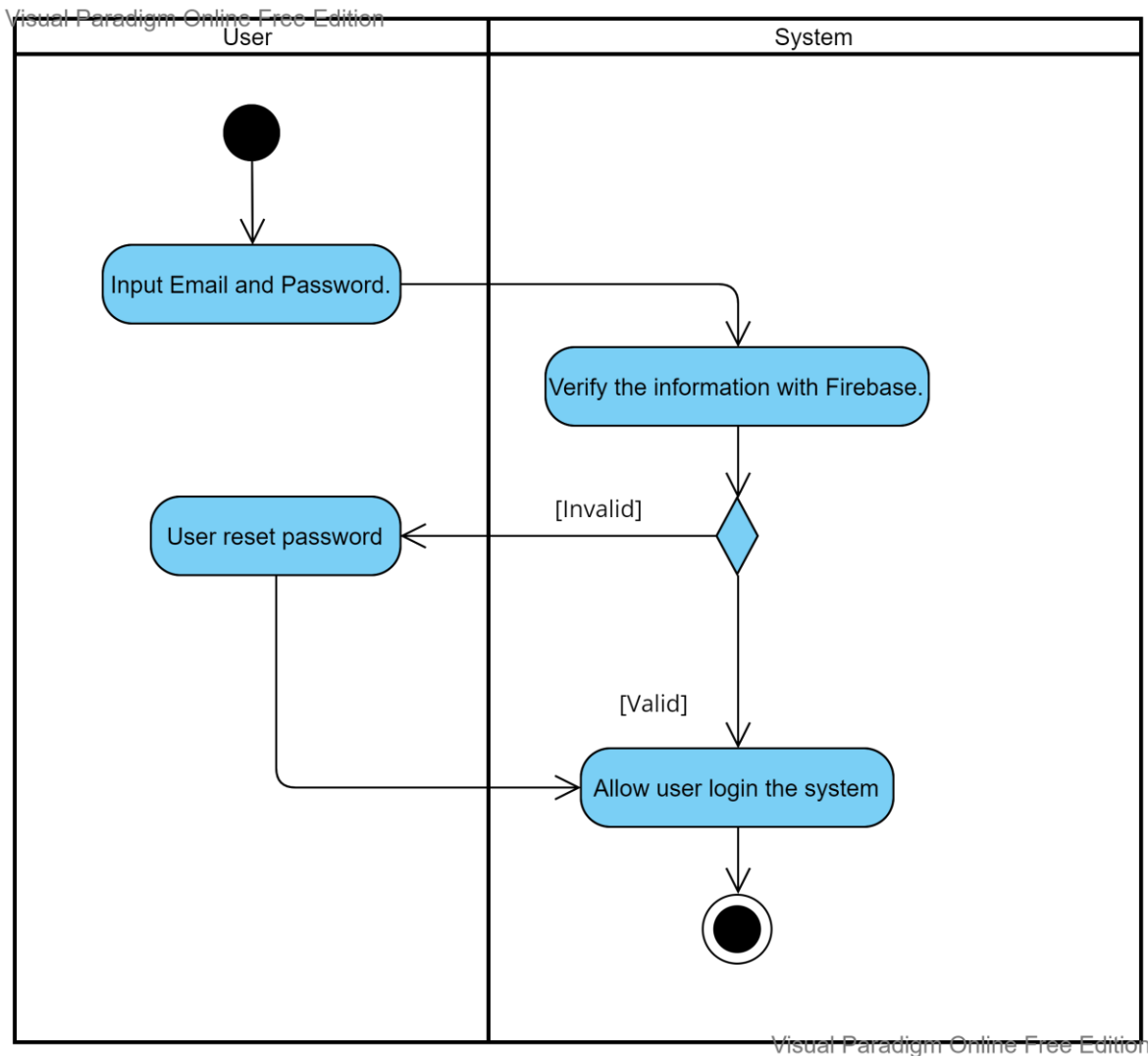


Figure 3.2.5.2 Activity Diagram of Login Account

Play Song

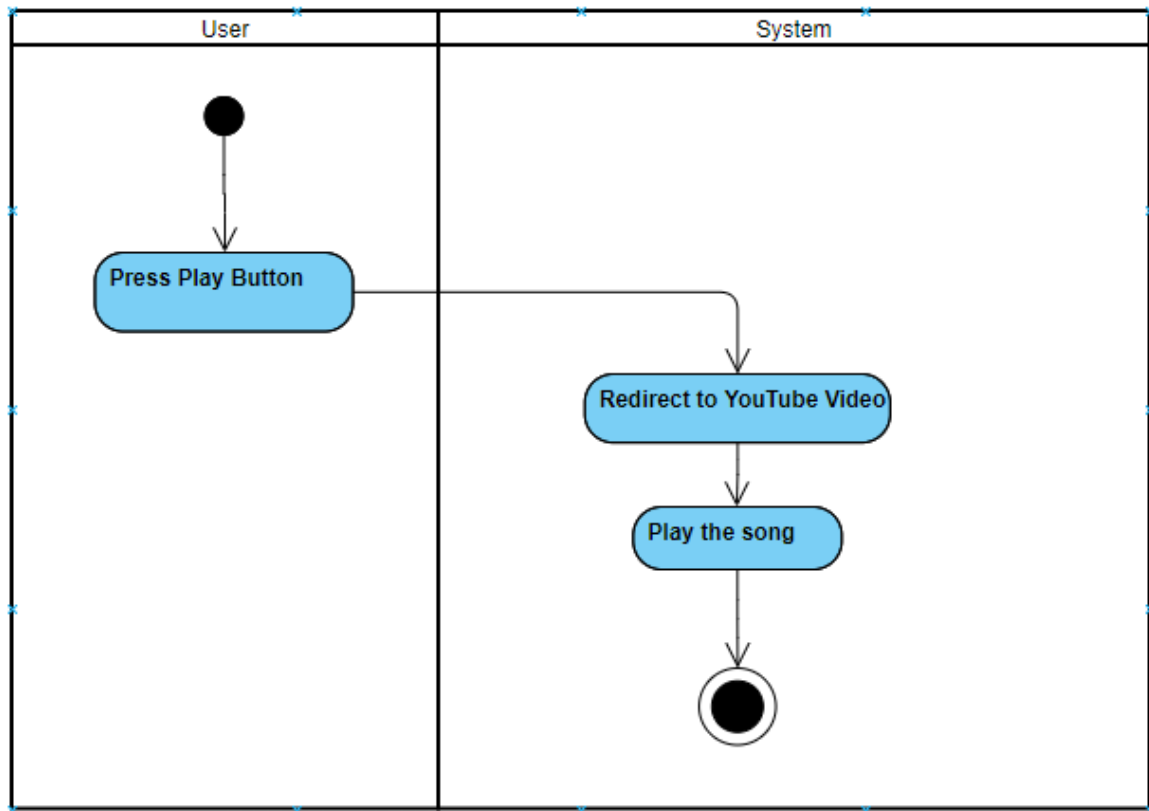


Figure 3.2.5.3 Activity Diagram of Play Song

Refresh Song

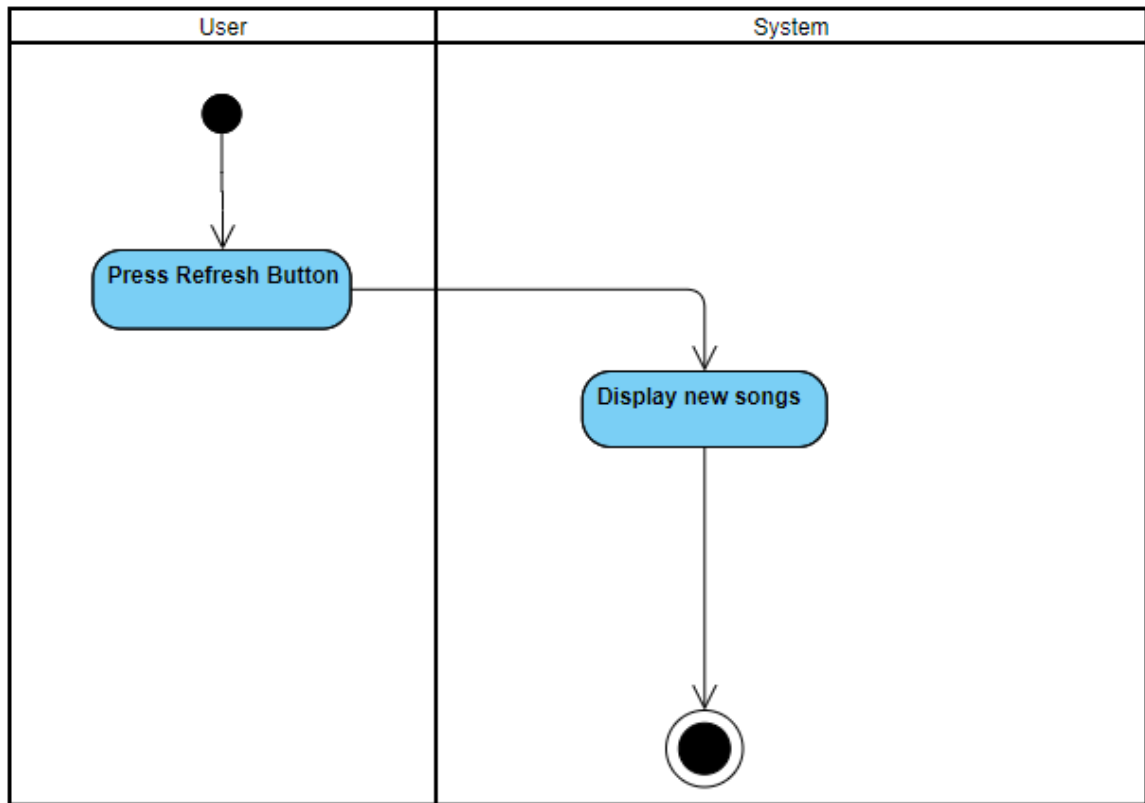


Figure 3.2.5.4 Activity Diagram of Refresh Songs

Add to Favourite

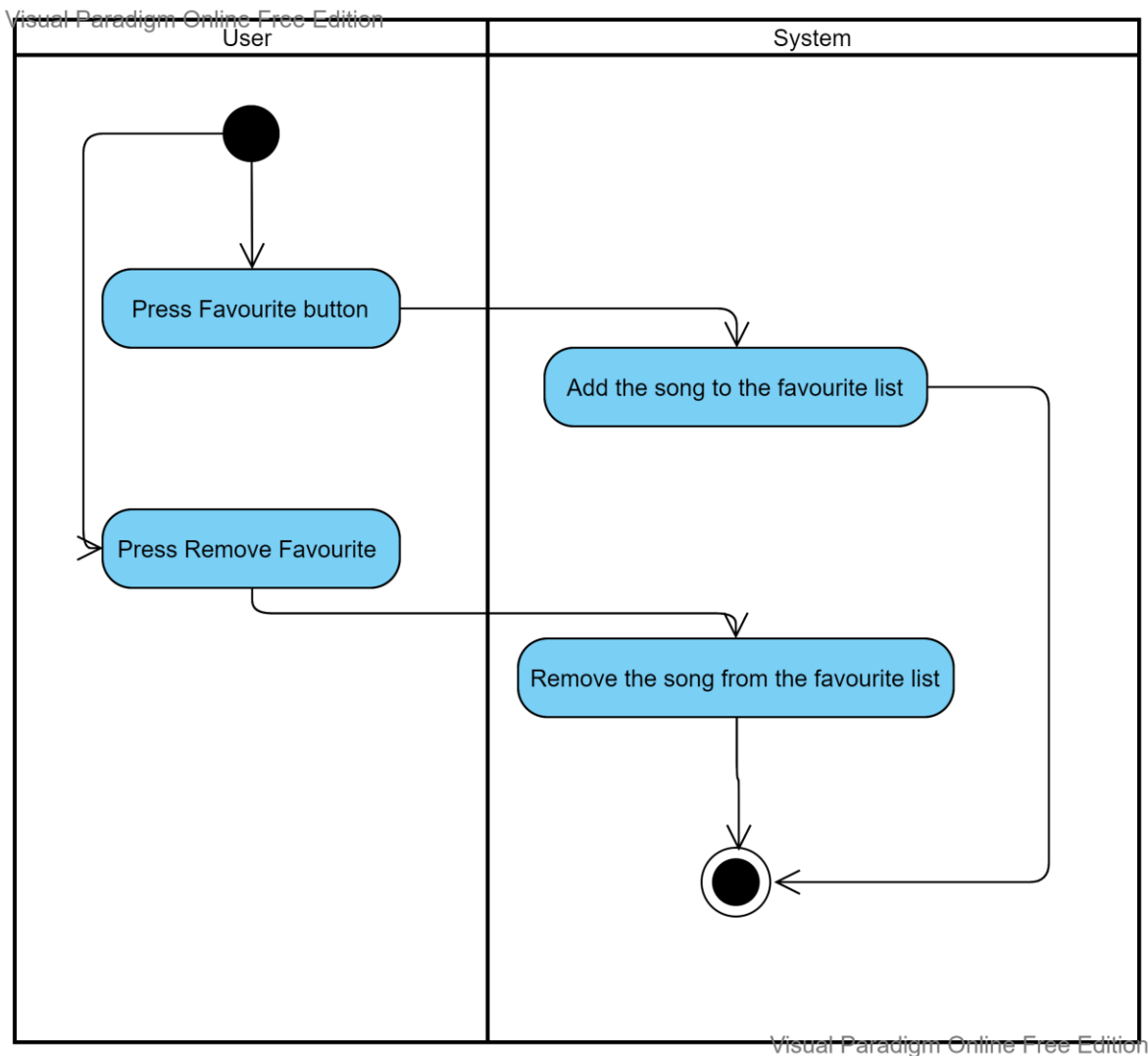


Figure 3.2.5.5 Activity Diagram of Add to Favourite

Logout

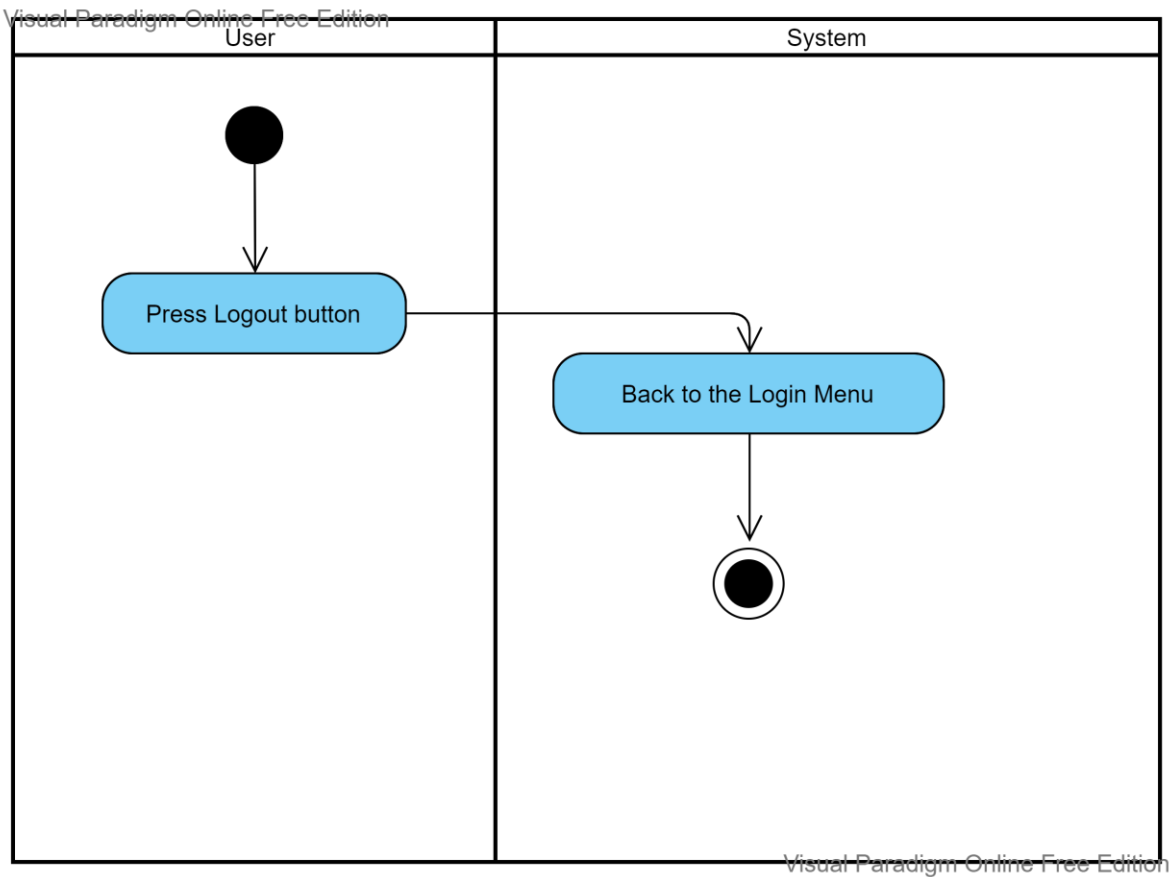


Figure 3.2.5.6 Activity Diagram of Logout

3.3 Implementation Issues and Challenges

There are several issues and challenges faced on this project include waste a lot of time of finding suitable dataset. A good dataset is hard to find, and a lot of job need to do such as data pre-processing and data cleaning. Moreover, we need to understand and analyze the dataset before train the recommender model. If the dataset not defined properly, it would affect the model and the quality of the recommender system. Furthermore, another implementation issues are providing the song links only because it is impossible to download all songs in the system because lack of storage.

In a nutshell, the other challenges faced include lack of knowledge of the Artificial Intelligence skill. This project will necessitate a significant amount of study time. I need to manage the time for this project and other academic subjects. Moreover, it is difficult to discuss my problems with my supervisor due to the COVID-19 pandemic as everything are held online. Therefore, I will spend more time on for searching the solution from online such as YouTube, Google and so more.

3.4 Verification Plan

The verification plans for the scenarios of the system will be shows below.

1. Register Account

Procedure Number	P1
Method	Testing
Purpose/Scope	To ensure the account created successfully.
Special Conditions/Limitations	-Email with correct @gmail.com format. -Password with minimum length of 6.
Acceptance Criteria	The system will create the account for the user.
Prerequisites	The system only run registered user.
Procedures	<ol style="list-style-type: none"> 1. Launch the system 2. Enter username. 3. Enter email. 4. Enter password. 5. Login to the system.
Troubleshooting	None

Table 3.4.1 Verification Plan P1(Register Account)

2. Login Account

Procedure Number	P2
Method	Testing
Purpose/Scope	To ensure the user login account with correct email and password.
Special Conditions/Limitations	-Enter correct email and password.
Acceptance Criteria	The system will verify the login information with Firebase.
Prerequisites	The system only run with correct user account.
Procedures	<ol style="list-style-type: none"> 1. Launch the system 2. Enter email. 3. Enter password. 4. Login to the system.
Troubleshooting	None

Table 3.4.2 Verification Plan P2(Login Account)

3. Reset Password

Procedure Number	P3
Method	Testing
Purpose/Scope	To ensure the user reset password successful.
Special Conditions/Limitations	-Enter correct email for reset the password.
Acceptance Criteria	The system will send a reset password link to registered email.
Prerequisites	The system allow user to reset password with registered email.
Procedures	<ol style="list-style-type: none"> 1. Launch the system 2. Enter email. 3. System will send reset password link to registered email. 4. User reset the password. 5. User login the account successful with new password.
Troubleshooting	None

Table 3.4.3 Verification Plan P3(Reset Password)

4. Play Songs

Procedure Number	P4
Method	Testing
Purpose/Scope	To ensure the songs is play correct and must related to the title of the song.
Special Conditions/Limitations	Need to click the song links and redirect to the YouTube.
Acceptance Criteria	The system will redirect the users to the YouTube.
Prerequisites	None
Procedures	<ol style="list-style-type: none"> 1. Press the Play button. 2. System will redirect the user to the YouTube page. 3. Listen song.
Troubleshooting	None

Table 3.4.4 Verification Plan P4(Play Songs)

5. Refresh Songs

Procedure Number	P5
Method	Testing
Purpose/Scope	To ensure the songs has been refresh without duplicate.
Special Conditions/Limitations	None
Acceptance Criteria	The system will refresh the songs without duplicate.
Prerequisites	The system requires users press the refresh button.
Procedures	<ol style="list-style-type: none"> 1. Press refresh button. 2. Observe song lists.
Troubleshooting	None

Table 3.4.5 Verification Plan P5(Refresh Song)

6. Add to Favourite

Procedure Number	P6
Method	Testing
Purpose/Scope	To ensure the songs has been added to favourite list.
Special Conditions/Limitations	None
Acceptance Criteria	The system will add the song to the favourite list.
Prerequisites	The system requires users press the Favourite button.
Procedures	<ol style="list-style-type: none"> 1. Press Favourite button. 2. System add song to favourite list. 3. System saves the song to the Firebase.
Troubleshooting	None

Table 3.4.6 Verification Plan P6(Add to Favourite)

7. Remove Favourite

Procedure Number	P7
Method	Testing
Purpose/Scope	To ensure the songs has been remove from favourite list.
Special Conditions/Limitations	None
Acceptance Criteria	The system will remove the song to the favourite list.
Prerequisites	The system requires users press the remove button.
Procedures	<ol style="list-style-type: none"> 1. Press remove button. 2. System remove song to favourite list. 3. System removes the song from the Firebase.
Troubleshooting	None

Table 3.4.7 Verification Plan P7(Remove Favourite)

8. Log Out

Procedure Number	P8
Method	Testing
Purpose/Scope	To ensure the user log out the system successful.
Special Conditions/Limitations	User log in the system.
Acceptance Criteria	The system will log out the account successful.
Prerequisites	The system requires users press the Log Out button.
Procedures	<ol style="list-style-type: none"> 1. Press Log Out button. 2. System log out the account. 3. System redirect to Login Menu.
Troubleshooting	None

Table 3.4.8 Verification Plan P8(Log Out)

3.5 Timeline

3.5.1 FYP 1

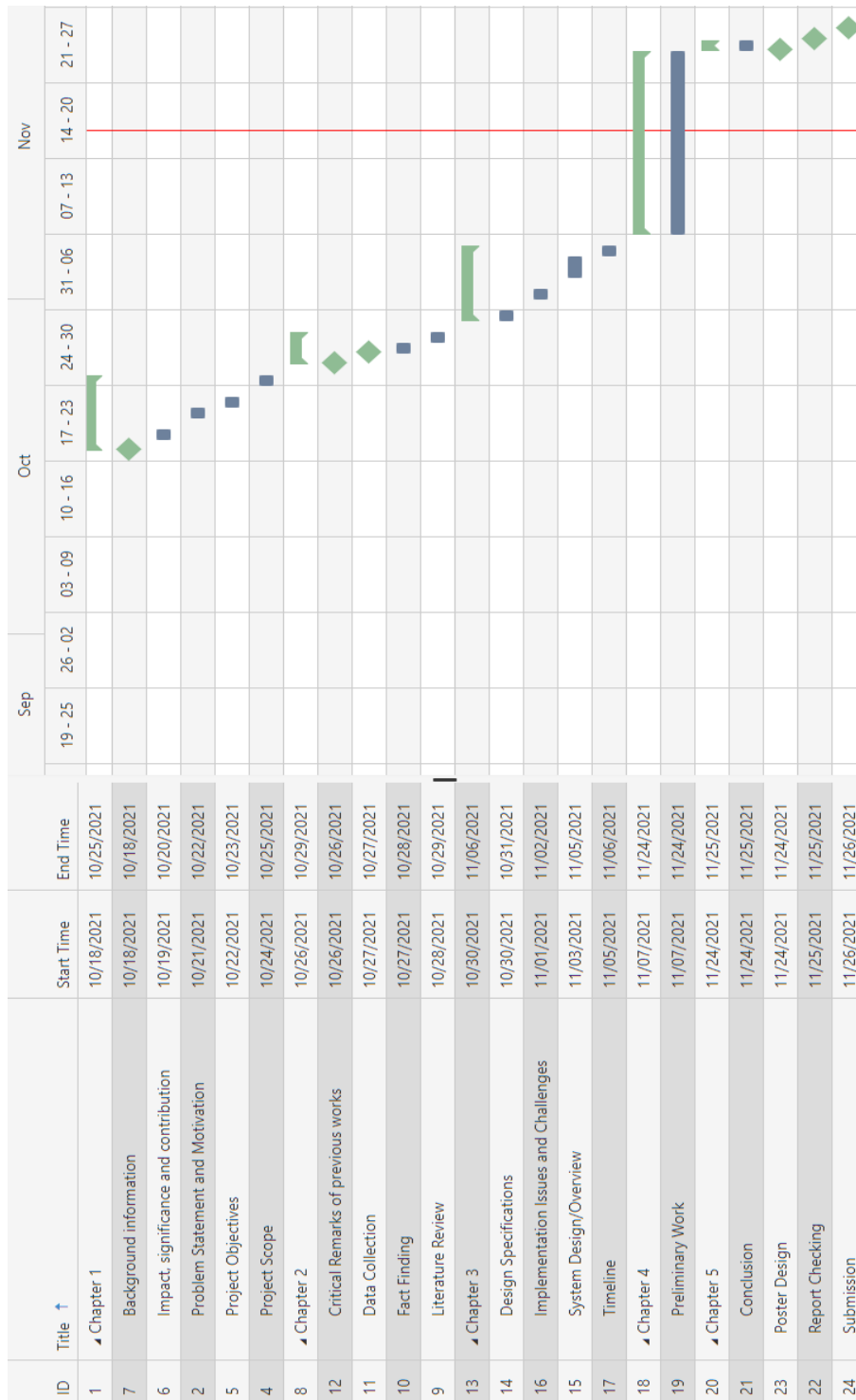


Figure 3.5.1.1 Timeline of FYP 1

3.5.2 FYP 2

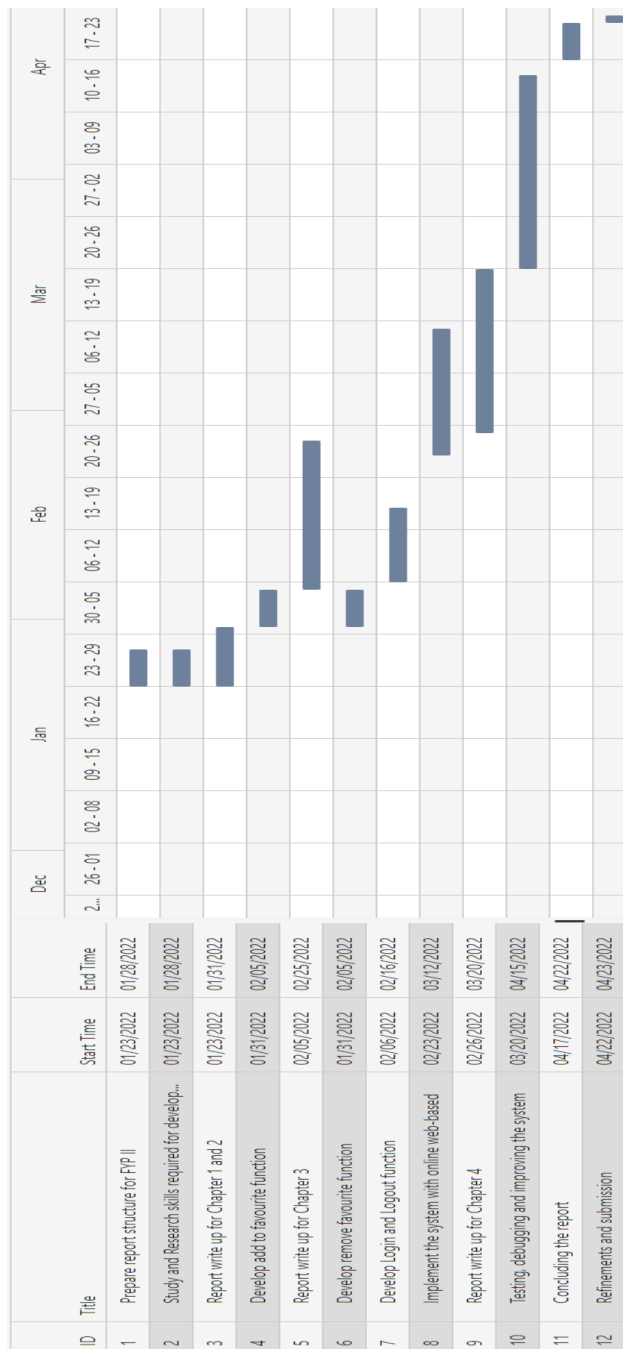


Figure 3.5.1.2 Timeline of FYP 2

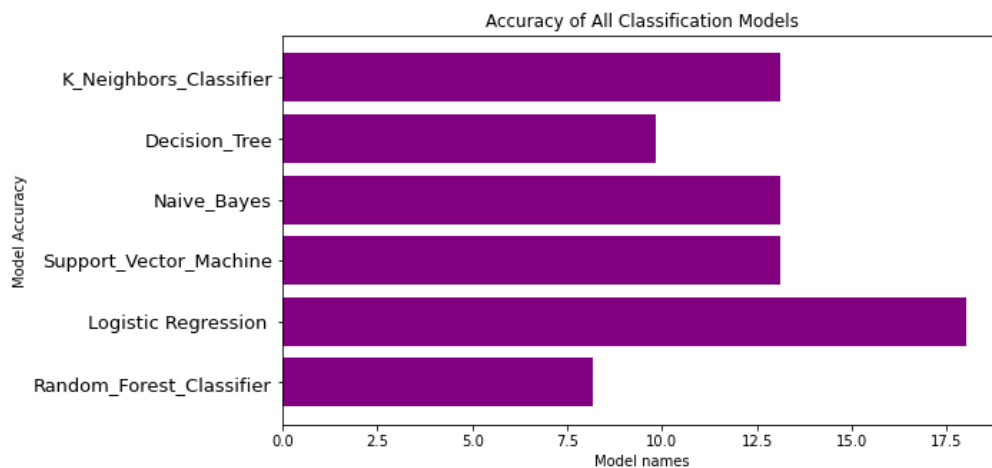
Chapter 4 Experiment and Model Implementation

In this chapter, the project will be shown and explain some comparison results and the model implementation. The figure below shows the comparison with various models and the accuracy. Moreover, the model implementation will be shows below as well.

4.1 Experiment Conducted with Various Model

Visualizing Accuracy of All Classification Models

```
In [61]: Index = [1,2,3,4,5,6]
plt.figure(figsize=(9,5))
plt.barh(Index,Model_Accuracy , color='purple' ,align='center', height=0.8)
plt.ylabel("Model Accuracy")
plt.xlabel("Model names")
plt.title("Accuracy of All Classification Models")
plt.yticks(Index, Model_Name , fontsize=13)
plt.show()
```



As you see above **Logistic_Regression** has highest accuracy

Figure 4.1.1 Experiment Conducted of All Models

Classification Report of Logistic Regression

```
In [64]: print(classification_report(y_test, Predicted_Value_Of_Logistic_Regression))
```

	precision	recall	f1-score	support
Bad Romance	0.00	0.00	0.00	9
Counting Stars	0.21	1.00	0.35	3
Dark Horse	0.18	0.25	0.21	8
Diamonds	0.25	0.40	0.31	5
Firework	0.00	0.00	0.00	8
Roar	0.00	0.00	0.00	4
See You Again	0.00	0.00	0.00	7
Someone Like You	0.29	0.29	0.29	7
Wake Me Up	0.17	0.67	0.27	3
We Found Love	0.00	0.00	0.00	7
accuracy			0.18	61
macro avg	0.11	0.26	0.14	61
weighted avg	0.10	0.18	0.12	61

Figure 4.1.2 Classification Report of Logistic Regression.

KNeighbors, Decision Tree, Naïve Bayes, Support Vector Machine, Logistic Regression and Random Forest Classifier are the classification models used to train the dataset in this experiment. *Figure 4.1.1* shows that the accuracy of Random Forest (8.19%), Logistic Regression (18.03%), Support Vector Machine (13.11%), Naïve Bayes(13.11%), Decision Tree(9.83%), and K-Neighbors(13.11%). The F1-score, precision, and recall (*figure 4.1.2*) for the songs are not high, as they are typically less than 0.5, apart from Counting Star, which has a perfect recall of 1.0.

```
In [43]: A.head(15)
```

	Actual_Values	Predicted_Values_Of_RFC
1670	Cry Me One Tear	Intro
13379	Wahnsinn (Live)	Intro
10234	Europa And The Pirate Twins	Intro
4719	Cincinnati_ Ohio	Intro
7003	January; By The Hearth	Intro
2831	David Hudson - Laura River	Intro
13014	Ti Shih Wu Ke Erh Tung	Caballo Viejo
11979	I Ain't Gonna Give Nobody None O' This Jelly Roll	Intro
8610	Parade: Choral	Intro
519	It Won't Escape Those Lips	Intro
13264	Sike	Intro
7329	Un Lugar En El Cielo	Intro
452	Han Yan Cui	Intro
12374	More Than Anything	Intro
4067	Waiting On Me	Intro

Figure 4.1.3 Actual vs Predicted value

From the simple experiment, the results shows that the classification models are not suitable for the recommender system because results of actual vs predicted values from the *Figure 4.1.3* are very bad. To improve accuracy, it is recommended that data be separated into hundreds of datasets as subsets, as processing millions of songs is time demanding and heavy. To overcome the challenges mentioned, preference learning with a good collaborative model will be employed.

4.2 Model Implementation with Collaborative Filtering Method

```

38     #Get users for all songs in user_songs.
39     u_songs_users = []
40     for i in range(0, len(u_songs)):
41         u_songs_users.append(self.get_i_users(u_songs[i]))
42     #Initialize the item cooccurrence matrix of size len(user_songs) X len(songs)
43     co_matrix = np.matrix(np.zeros(shape=(len(u_songs), len(a_songs))), float)
44
45     #Calculate similarity between songs listened by the user and all unique songs in the training data
46     for i in range(0, len(a_songs)):
47         #Calculate unique listeners (users) of song (item) i
48         songs_i_data = self.t_data[self.t_data[self.i_id] == a_songs[i]]
49         users_i = set(songs_i_data[self.u_id].unique())
50
51         for j in range(0, len(u_songs)):
52
53             #Get unique listeners (users) of song (item) j
54             users_j = u_songs_users[j]
55
56             #Calculate the songs which are in common listened by users i & j
57             users_intersection = users_i.intersection(users_j)
58
59             #Calculate cooccurrence_matrix[i,j] as Jaccard Index
60             if len(users_intersection) != 0:
61                 #Calculate all the songs listened by i & j
62                 users_union = users_i.union(users_j)
63
64                 co_matrix[j,i] = float(len(users_intersection))/float(len(users_union))
65             else:
66                 co_matrix[j,i] = 0
67
68
69     return co_matrix

```

Figure 4.2.1 Construct Cooccurrence Matrix

From the Figure 4.2.1 shows that how to construct a cooccurrence matrix for the model. First, get users for all songs in the variable user_songs. The matrix will be initializing the size len(user_songs) x len(songs). After the initialize, we need to calculate the similarity between songs listened by the user and all unique songs in the training data. The variable i and j will be used for calculating the intersection between the users and songs. Then insert the variables to the cooccurrence matrix. The function will return the results of the cooccurrence matrix.

```

71 | #Use the cooccurrence matrix to make top recommendations
72 | def generate_top_r(self, user, cooccurrence_matrix, a_songs, u_songs):
73 |     # print("%d" % np.count_nonzero(cooccurrence_matrix))
74 |
75 |     #Calculate the average of the scores in the cooccurrence matrix for all songs listened by the user.
76 |     user_sim_scores = cooccurrence_matrix.sum(axis=0)/float(cooccurrence_matrix.shape[0])
77 |     user_sim_scores = np.array(user_sim_scores)[0].tolist()
78 |
79 |     #Sort the indices of user_sim_scores based upon their value also maintain the corresponding score
80 |     s_index = sorted(((e,i) for i,e in enumerate(list(user_sim_scores))), reverse=True)
81 |
82 |     #Create a dataframe from the following
83 |     columns = ['user_id', 'song', 'score', 'rank']
84 |     #index = np.arange(1) # array of numbers for the number of samples
85 |     df1 = pandas.DataFrame(columns=columns)
86 |
87 |     #Fill the dataframe with top 10 songs
88 |     rank = 1
89 |     for i in range(0,len(s_index)):
90 |         if ~np.isnan(s_index[i][0]) and a_songs[s_index[i][1]] not in u_songs and rank <= 10:
91 |             df1.loc[len(df1)]=[user,a_songs[s_index[i][1]],s_index[i][0],rank]
92 |             rank = rank+1
93 |
94 |     #Handle the case where there are no recommendations
95 |     if df1.shape[0] == 0:
96 |         # print("The current user don't have any song for similarity based recommendation model.")
97 |         return -1
98 |     else:
99 |         return df1

```

Figure 4.2.2 Make Recommendations with Cooccurrence Matrix

Next step is using the cooccurrence matrix to make top recommendations. The function will calculate the average of the scores in the cooccurrence matrix for all songs listened by the user. Then will sort the indices of user_sim_scores based on their value also maintain the corresponding score. Then create and fill the dataframe for user_id, song, score and the rank with Top 10 songs.

```

101     #Create the system model
102     def create_s(self, t_data, u_id, i_id):
103         self.t_data = t_data
104         self.u_id = u_id
105         self.i_id = i_id
106     #Use the model to make recommendations
107     def recommend_s(self, u):
108
109         #A. Get all unique songs for this user
110         u_songs = self.get_u_items(u)
111
112         # print("No. of songs for the user: %d" % len(u_songs))
113
114         #B. Get all the songs in the data
115         a_songs = self.get_all_items_t_data()
116
117         # print("No. of songs in the list: %d" % len(a_songs))
118
119         #C. Make the cooccurrence matrix of size len(user_songs) X len(songs)
120         co_matrix = self.construct_co_matrix(u_songs, a_songs)
121
122         #D. Use the matrix to make recommended songs
123         df_r = self.generate_top_r(u, co_matrix, a_songs, u_songs)
124         return df_r

```

Figure 4.2.3 Create and Use the Model to Make Recommendations

Next step is creating the system model. Then get all unique songs according to the user and get all the songs in the dataset. Then insert to the matrix and used to make recommended songs for the user.

CHAPTER 4 EXPERIMENT AND MODEL IMPLEMENTATION

```
In [22]: is_model = Recommenders.similarity_recommender()
is_model.create_s(train, 'user_id', 'song')
```

```
In [23]: #Print the songs for the user
user_id1 = u[15]
user_items1 = is_model.get_u_items(user_id1)
print("-----")
print("Songs played by user %s:" % user_id1)
print("-----")
for user_item in user_items1:
    print(user_item)
print("-----")
print("Similar songs recommended for the user:")
print("-----")
#Recommend songs for the user using personalized model
is_model.recommend_s(user_id1)
```

Songs played by user ed7d4c476013b1c3dd91982b61494bf7436083ba:

Sehr kosmisch - Harmonia - <https://www.youtube.com/watch?v=P0hFr6i8y9c>
Dans Ma Bulle (Edit Radio - Live 2006) - Diam's - <https://www.youtube.com/watch?v=Ft1Y4deA2mo>

Similar songs recommended for the user:

```
Out[23]:
```

	user_id	song	score	rank
0	ed7d4c476013b1c3dd91982b61494bf7436083ba	Revelry - Kings Of Leon - https://www.youtube...	0.181818	1
1	ed7d4c476013b1c3dd91982b61494bf7436083ba	Undo - Björk - https://www.youtube.com/watch?v...	0.166667	2
2	ed7d4c476013b1c3dd91982b61494bf7436083ba	Learn To Fly - Foo Fighters - https://www.yout...	0.136364	3
3	ed7d4c476013b1c3dd91982b61494bf7436083ba	Dog Days Are Over (Radio Edit) - Florence + Th...	0.125000	4
4	ed7d4c476013b1c3dd91982b61494bf7436083ba	You're The One - Dwight Yoakam - https://www.y...	0.107143	5
5	ed7d4c476013b1c3dd91982b61494bf7436083ba	Superman - Eminem / Dina Rae - https://www.you...	0.100000	6
6	ed7d4c476013b1c3dd91982b61494bf7436083ba	Use Somebody - Kings Of Leon - https://www.you...	0.100000	7
7	ed7d4c476013b1c3dd91982b61494bf7436083ba	Just Dance - Lady GaGa / Colby O'Donis - https...	0.090909	8
8	ed7d4c476013b1c3dd91982b61494bf7436083ba	Ain't Misbehavin - Sam Cooke - https://www.you...	0.090909	9
9	ed7d4c476013b1c3dd91982b61494bf7436083ba	Drops Of Jupiter - Train - https://www.youtube...	0.090909	10

Figure 4.2.4 The Results of the Recommender System

From the *Figure 4.2.4* shows that the system will display songs played by the user with `user_id`. Then the system will also display the Top 10 recommended song according by the score. Moreover, the system will also show the song's links as well.

Chapter 5 System Implementation and Testing

5.1 Project Screenshot and Explanation

Since the model was developed in Jupyter Notebook, it needs to be zipped to a pickle file so that it can be used in our program. A website is implemented by this program using Visual Studio Code to access HTML and CSS. Moreover, Visual Studio can also unzip a pickle file to access all the useful functions created by Jupyter Notebooks.

```
# to un zip the pickle file to use the dataset in our program
train = pickle.load(open('trainset.pkl','rb'))
# from Recommenders we need to import similarity_recommender() class so that we use our all usefull functions
is_model = Recommenders.similarity_recommender()
# in this line of code we give the argument to the function named create_s() in which our program can create similarities
is_model.create_s(train , 'user_id' , 'song')
u = train['user_id'].unique()
```

Figure 5.1.1 Code for Unzip the Pickle File

5.1.1 Login Interface

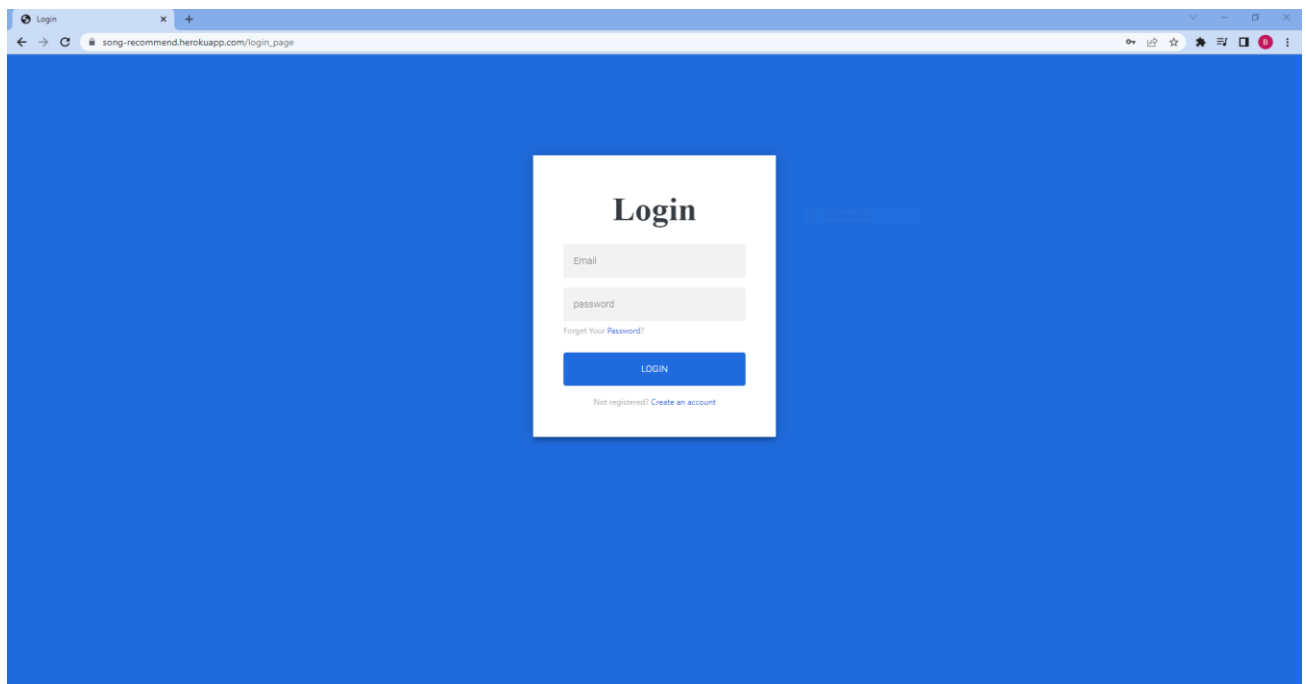


Figure 5.1.1.1 Login Interface

The is the interface of the User Login page. This function will request user to key in registered email and password.

5.1.2 Login Fail

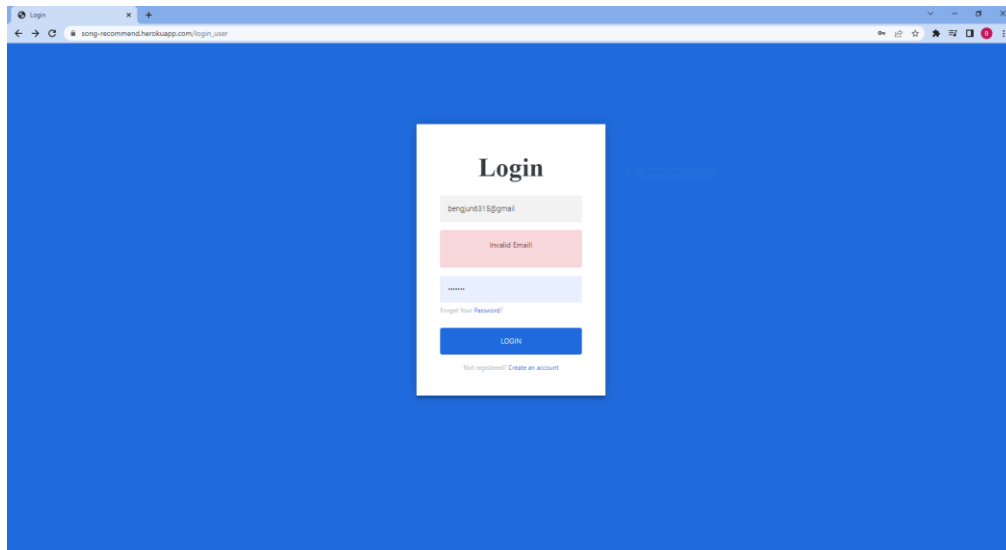


Figure 5.1.2.1 Login Fail

Next is the user input not under registered email or incorrect password, the system will prompt error message such as “Invalid Email”. In this situation, user can choose to reset password or the create an account.

5.1.3 Sign Up

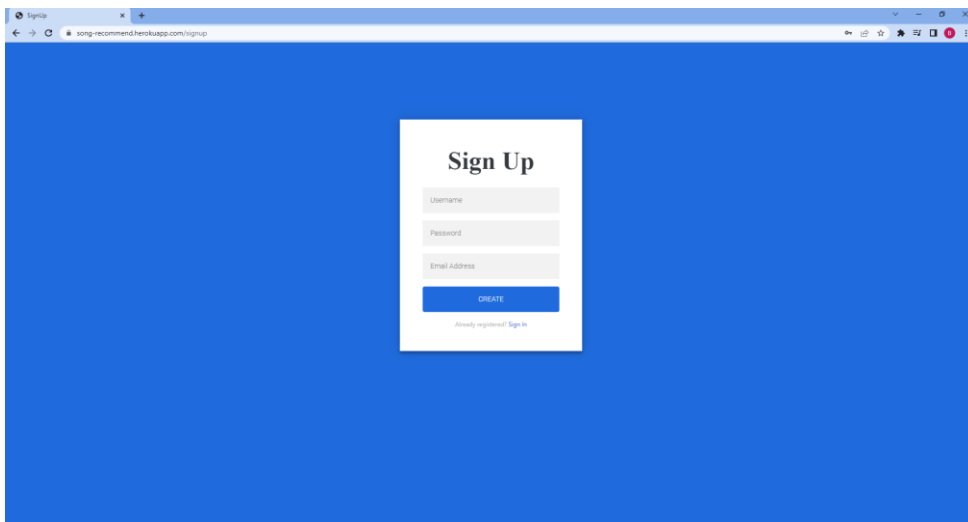


Figure 5.1.3.1 Sign Up

In the Sign-Up page, the system will request user to input username, password, and email address.

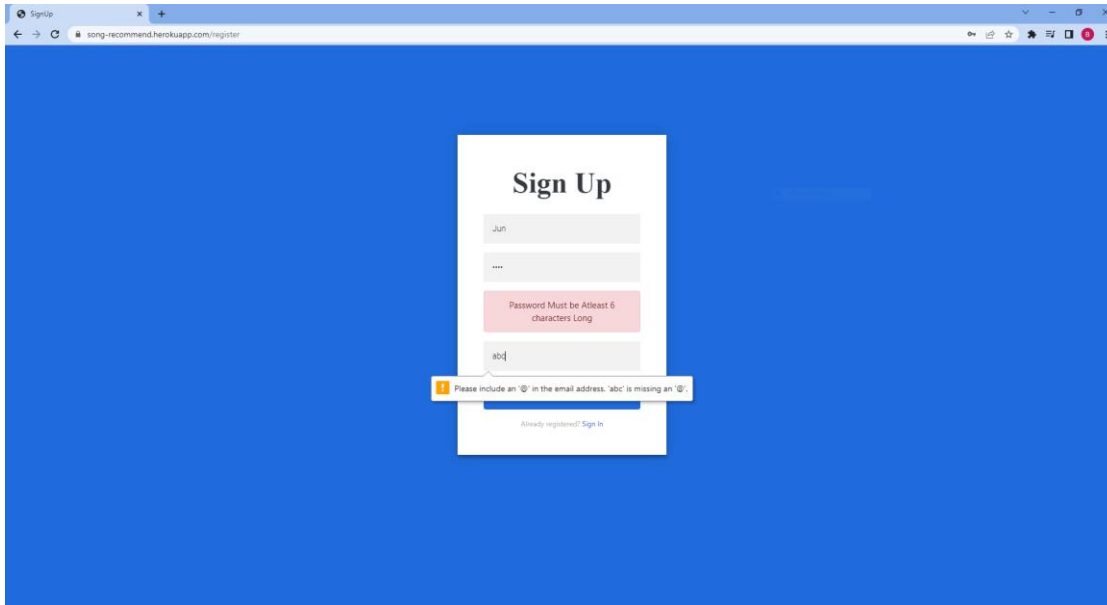


Figure 5.1.3.2 Sign Up format

The system will require user to key in the password with at least 6 characters long. Moreover, the email format also includes @ in the email address. Moreover, the system only accepts unique email addresses, repeated emails are not accepted.

5.1.4 Reset Password

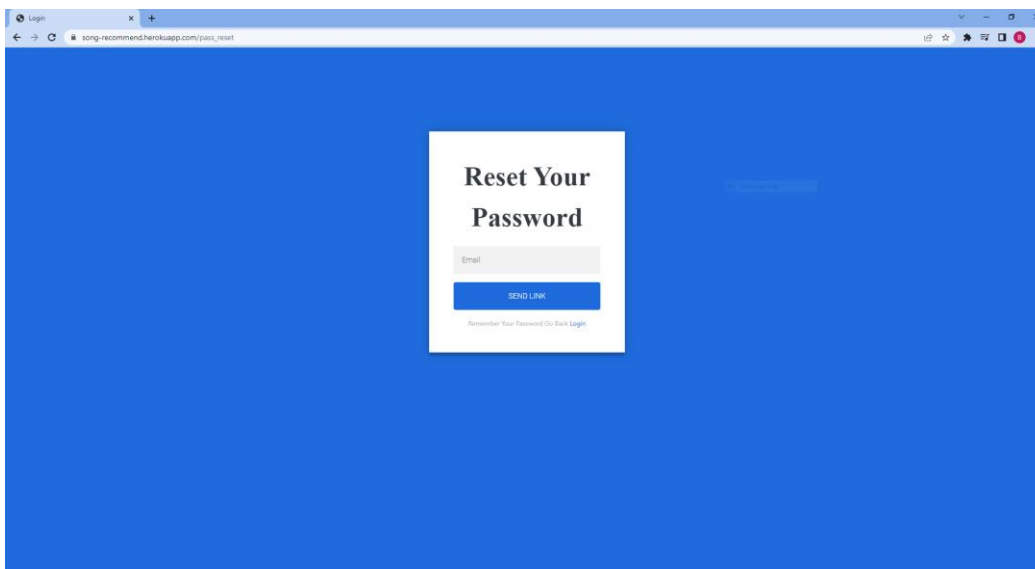


Figure 5.1.4.1 Reset Password

The user can reset their password if they are unable to log in. After entering the registered email address, the system will send a link to reset the password.

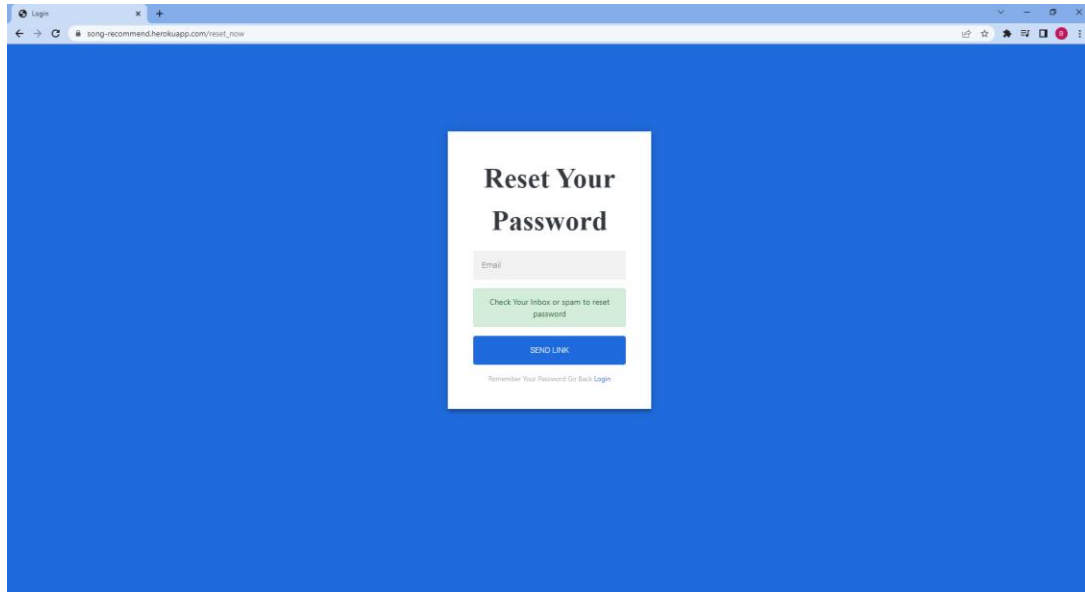


Figure 5.1.4.2 Check Inbox

After key in the registered email, system will prompt message inform user to check their inbox or spam to reset the password.

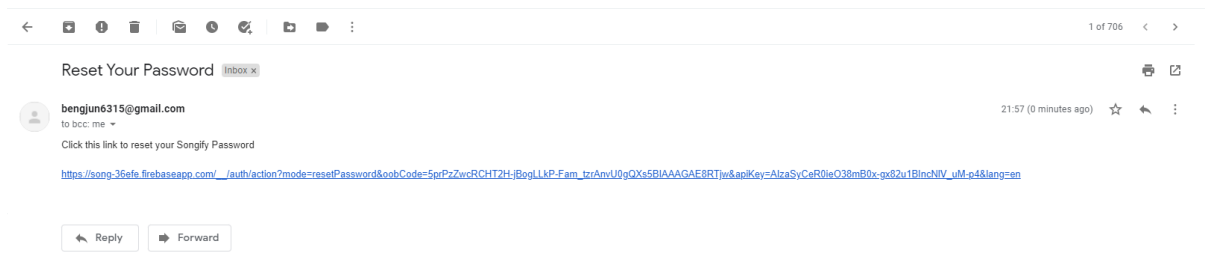


Figure 5.1.4.3 Send Link

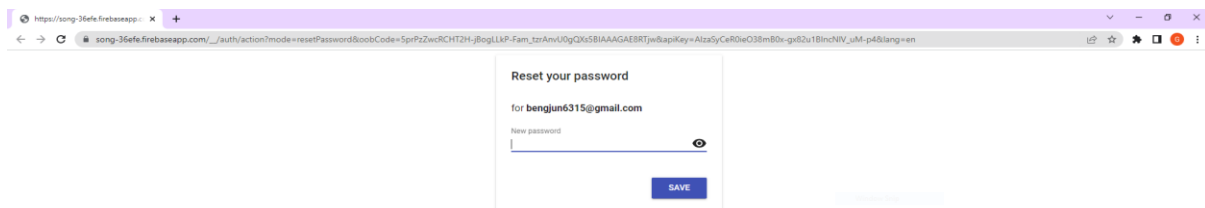


Figure 5.1.4.4 Key in New Password

From the Figure 5.1.4.3, the system will send a link to the user's email address. User can click the link to reset the new password as shown in Figure 5.1.4.4.

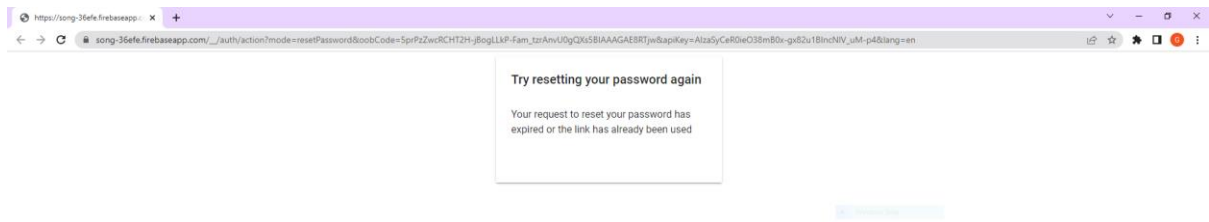


Figure 5.1.4.5 Expired Link

If the user does not reset the password within one hour, the link will expire.

5.1.5 Firebase

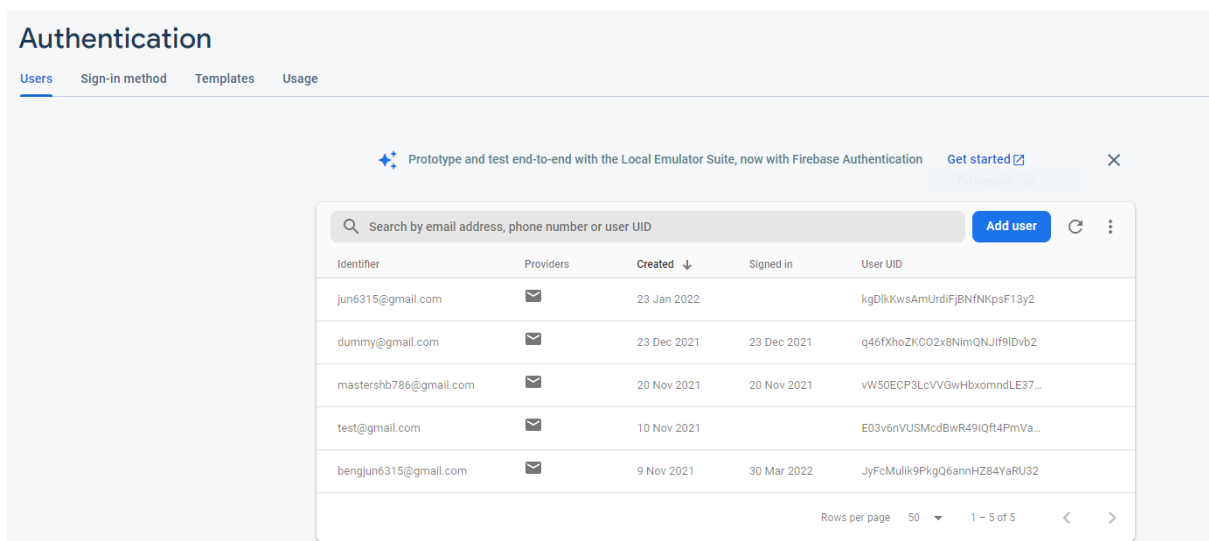


Figure 5.1.5.1 Authentication

The Firebase will save the user information if the user signs up for an account successfully. Moreover, we can also reset, disable, and delete an account under authentication.

5.1.6 Main Menu

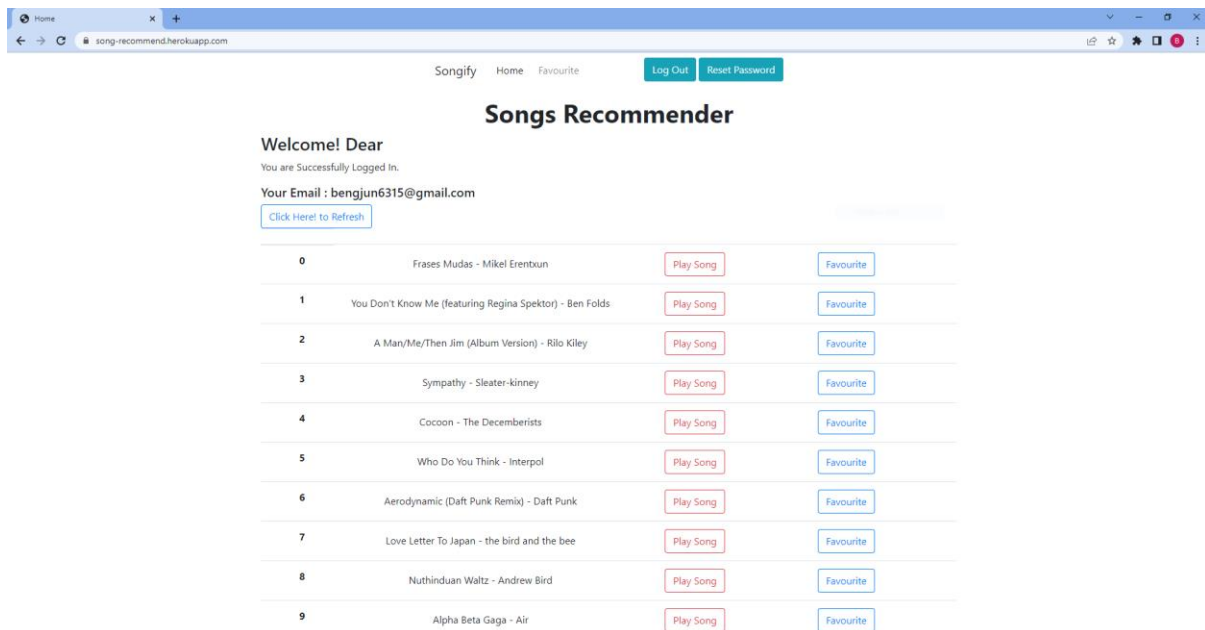


Figure 5.1.6.1 Main Menu

After successful login, this is the Main Menu. Users can view the Top 10 songs listed by the system. There are several functions, and they will be explained below with the figure.

5.1.7 Play Song

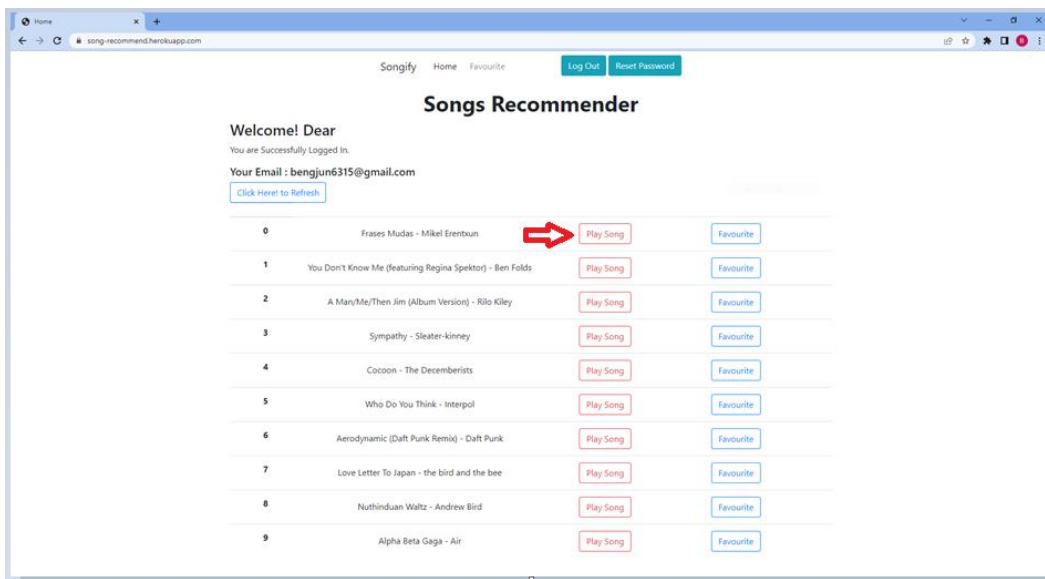


Figure 5.1.7.1 Click Play Song

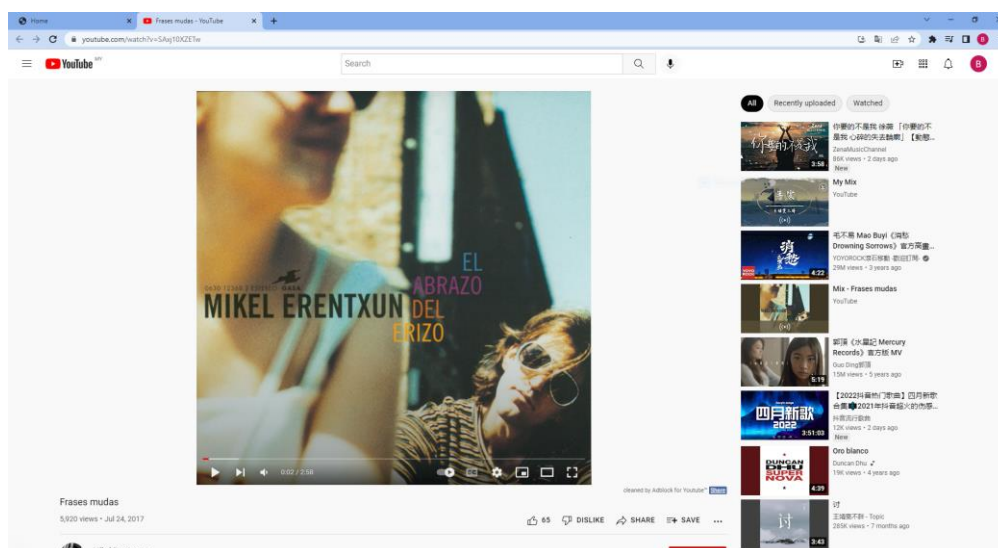


Figure 5.1.7.2 YouTube Video

After clicking the Play Song button, the system will automatically redirect user to the corresponding YouTube video. Thus, user can enjoy the song.

5.1.8 Refresh Song

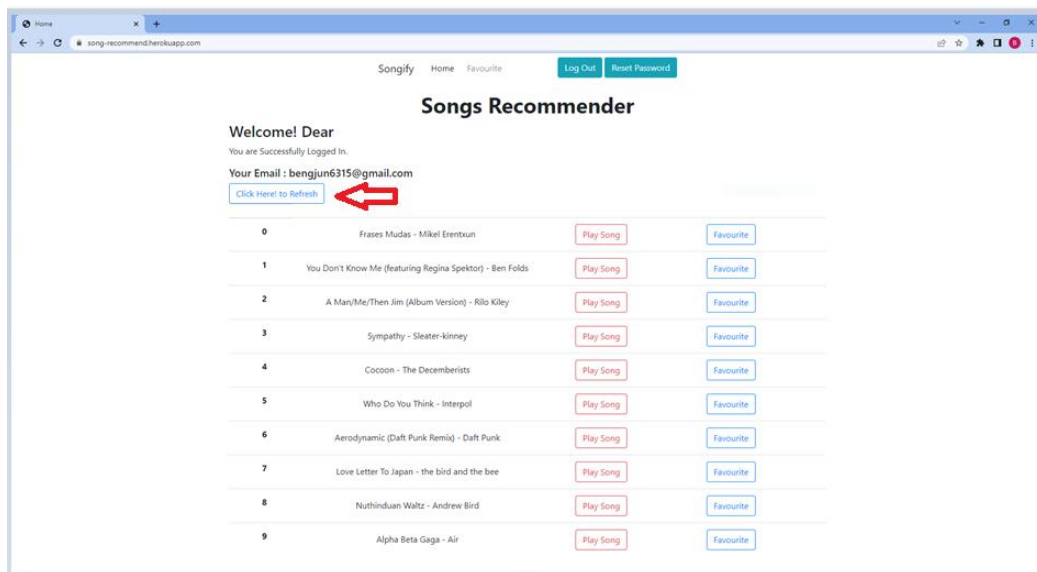


Figure 5.1.8.1 Click Refresh Button

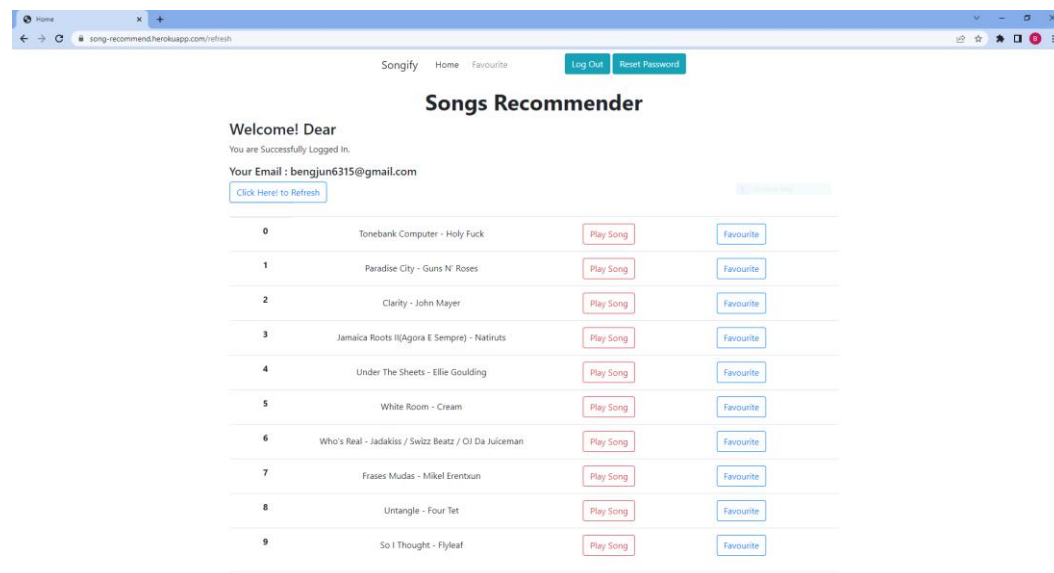


Figure 5.1.8.2 Refresh Song

In this page, user can click the Refresh Button if don't like the recommendations. The system will generate and display next 10 songs without duplicate to user.

CHAPTER 5 SYSTEM IMPLEMENTATION AND TESTING

```
# Refresh recommendations Function Goes Here
# //////////////////////////////////////
@app.route('/refresh', methods = ['GET', 'POST'])
def refresh():
    # now we have songs and there link in same column and both are merged so we need to seperate them to get the values
    # i can do it with slicing and store the values in below saporate lis
    songlink_refresh = []
    song_refresh = []
    # Function for seperate songs name and Links
    seperate_song_link_refresh(songlink_refresh , song_refresh)
    # to get the newer recommendation we need random values to recommend user the different songs
    # so that we choose set() function because we don't want any duplicate number and also not any user want duplicate songs
    a=set()
    # to done above procedure agian we need to run a Loop that generate 10 random numbers to get the newer recommendations
    for i in range(0,10):
        # we get and add the random values in out set() named (" a ")
        a.add(random.randint(0,1000))
    songlink_ = []
    song_ = []
    for i , v in enumerate(a):
        songlink_.append(songlink_refresh[v])
        song_.append(song_refresh[v])

    return render_template('refresh.html' , songlink_refresh = songlink_ , song_refresh = song_)

# //////////////////////////////////////
# Refresh recommendations Function End Here
```

Figure 5.1.8.3 Refresh Function

The figure shows how to generate the next 10 songs for the users without duplicating them.

5.1.9 Add to Favourite

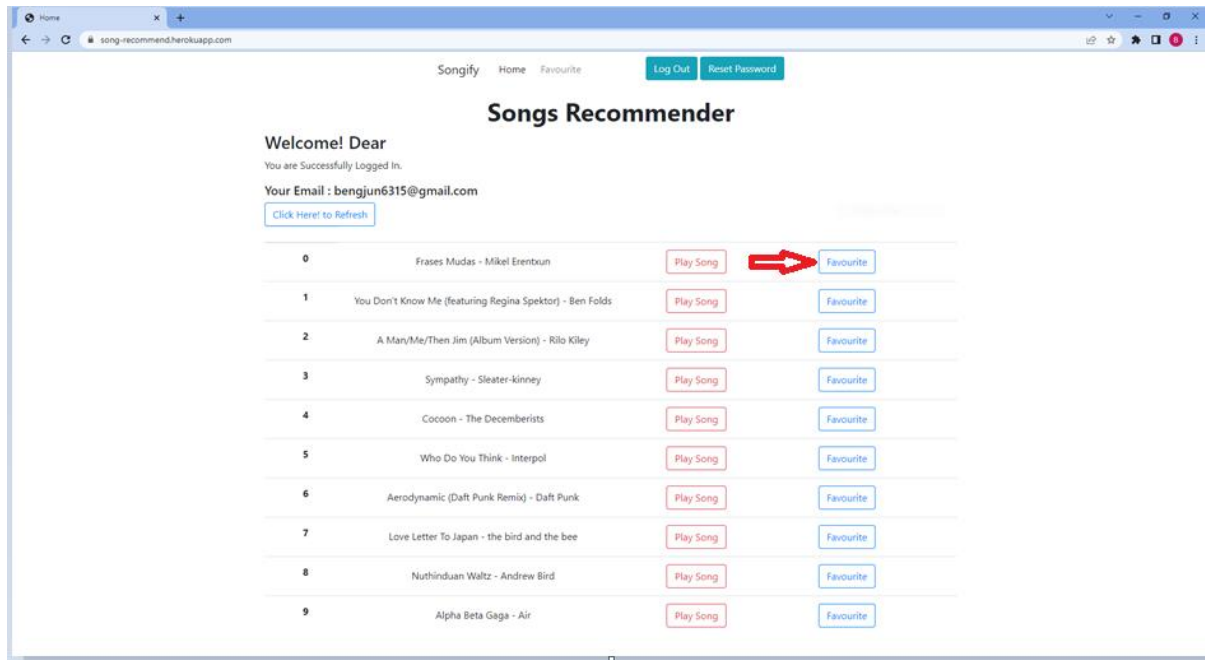


Figure 5.1.9.1 Click Favourite Button

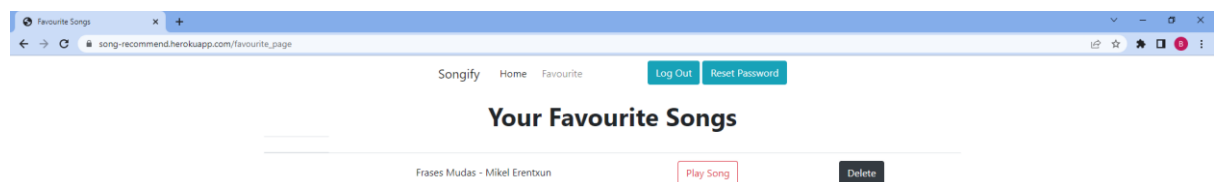


Figure 5.1.9.2 Favourite List

When user click Favourite Button, the system will add the song to the favourite list which is shown from Figure 5.1.9.1.

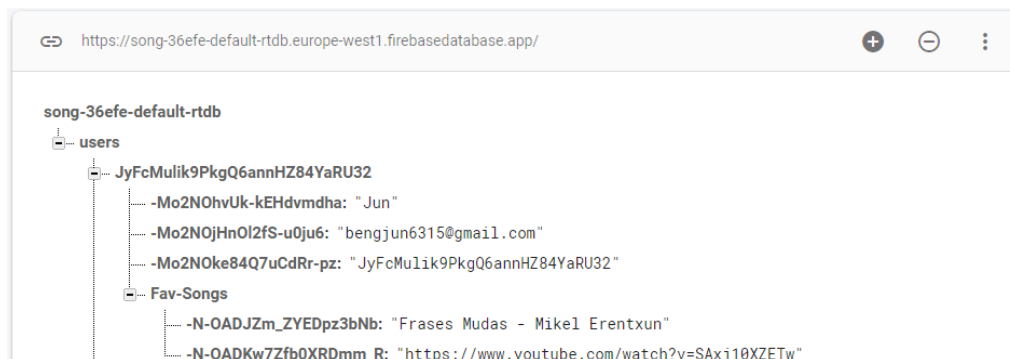


Figure 5.1.9.3 Firebase

The system will save the song into the Firebase with the favourite song list.

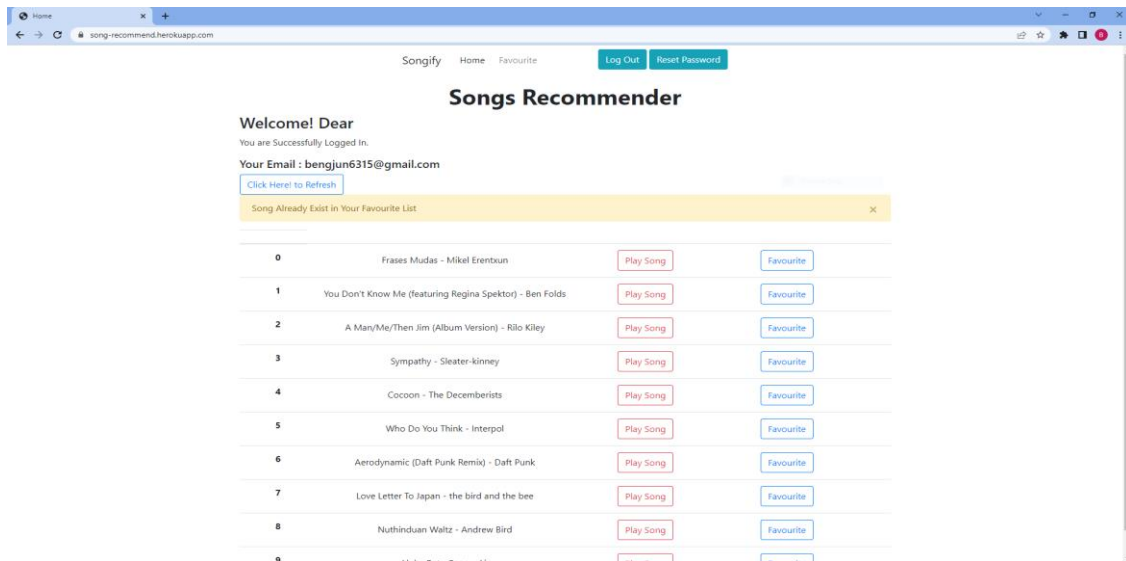


Figure 5.1.9.4 Existing Song

When a user clicks the Favourite button more than once with the same song, an alert message is displayed.



Figure 5.1.9.5 Delete Song

This is the interface when user remove the song from the favourite list.



Figure 5.1.9.6 Click Home Button

User can back to Main Menu page with the Home button function.

5.1.10 Log Out

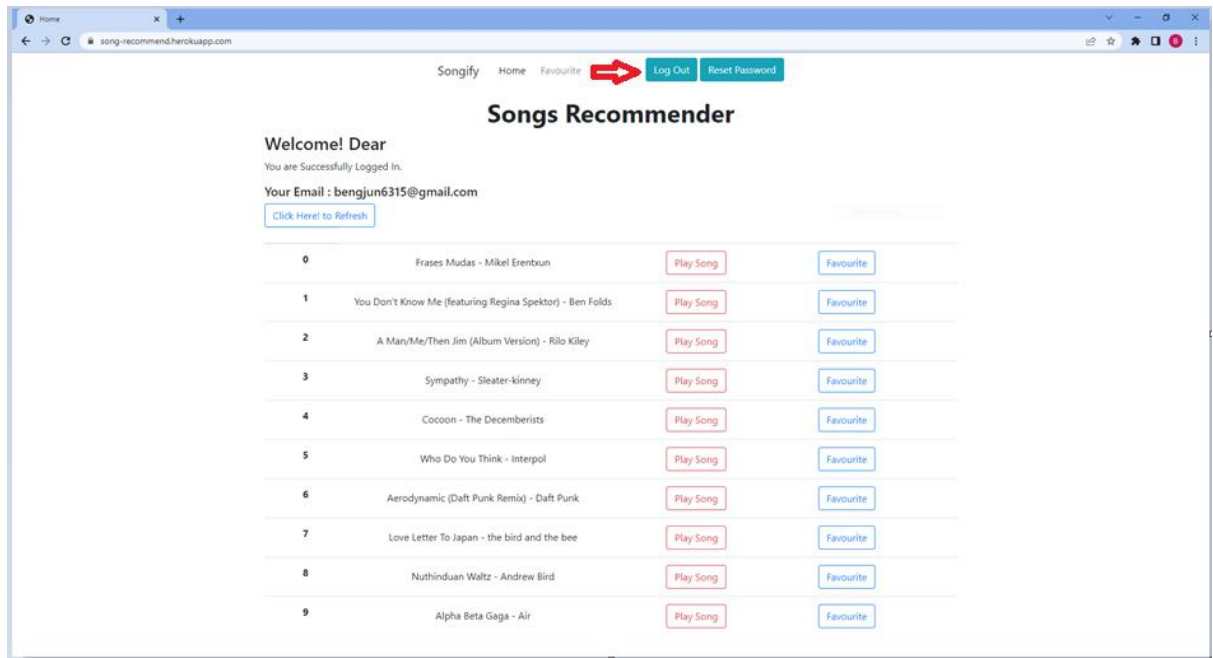


Figure 5.1.10.1 Click Log Out Button

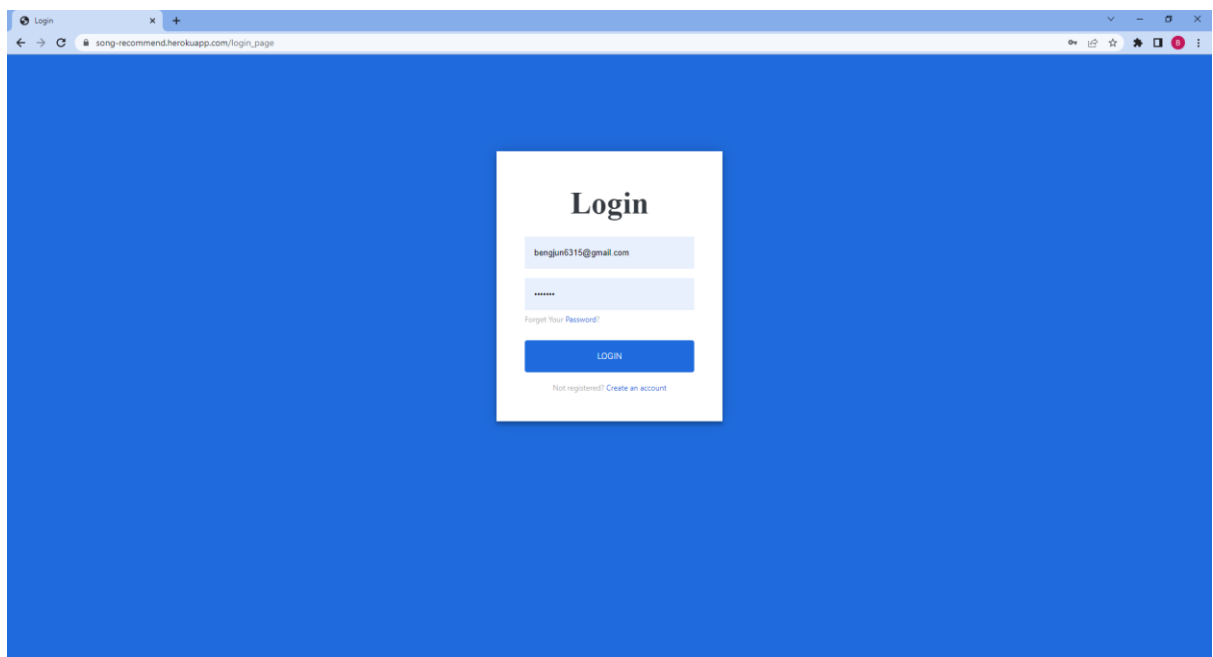


Figure 5.1.10.2 Log Out

When user click the Log Out button, the system will redirect the user back to login page.

5.2 Module Testing

5.2.1 Account Login Testing

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TA01	Check user login with valid Data.	Email=test@gmail.com Password=123456	User should login into the system.	As expected.	Pass
TA02	Check user login with invalid data.	Email=unregistered@gmail.com Password=123	User should not login into the system.	As expected.	Pass

Table 5.2.1.1 Account Login Testing

5.2.2 Account Registration Testing

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TB01	Check user registration with valid Data.	Email=test1@gmail.com Password=123456	User should register and login the system.	As expected.	Pass
TB02	Check user registration with invalid data.	Email=test1.gmail Password=123	User should not register and login the system.	As expected.	Pass

Table 5.2.2.1 Account Registration Testing

5.2.3 Reset Password Testing

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TC01	Check user reset password with valid email.	Email=registered@gmail.com	User should receive the password reset link email.	As expected.	Pass
TC02	Check user reset password with invalid email.	Email=unknown@gmail.com	User should not receive the password reset link email.	As expected.	Pass

Table 5.2.3.1 Reset Password Testing

5.2.4 Play Song Testing

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TD01	Check the function of Play Song.	-	Users should allow to click the play button and listen song.	Users can click the Play button and listen the song.	Pass
TD02	Check the correct YouTube link.	-	System should redirect user to the correct YouTube link.	As expected.	Pass
TD03	Check the multiple of play song.	-	Users allow to listen multiple songs.	As expected.	Pass

*Table 5.2.4.1 Play Song Testing***5.2.5 Refresh Song Testing**

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TE01	Check the function of Refresh Song.	-	System should generate new 10 songs.	As expected.	Pass
TE02	Check the duplicate of songs.	-	New 10 songs should not be duplicate.	As expected.	Pass

*Table 5.2.5.1 Refresh Song Testing***5.2.6 Add to Favourite Testing**

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TF01	Check the function of Add to Favourite.	Add the song to Favourite.	System should add the song to the Favourite Lists.	As expected.	Pass
TF02	Check the duplicate of songs.	Add the same song to Favourite.	New 10 songs should not be duplicate.	As expected.	Pass
TF04	Check adds multiple of songs to the list.	Add multiple of songs.	System should successful add multiple of songs to the list.	As expected.	Pass
TF04	Check the song save into the Firebase.	-	The Firebase should save the songs.	As expected.	Pass

Table 5.2.6.1 Add to Favourite Testing

5.2.7 Remove Favourite Testing

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TG01	Check the function of Remove Song.	Remove the song from Favourite List.	System should remove the song from the Favourite Lists.	As expected.	Pass
TG02	Check remove multiple of songs from the list.	Remove the multiple song from to Favourite list.	New 10 songs should not be duplicate.	As expected.	Pass
TG03	Check the song remove from the Firebase.	-	The Firebase should remove the songs.	As expected.	Pass

*Table 5.2.7.1 Remove Favourite Testing***5.2.8 Logout Testing**

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TH01	Check the user logout the system.	Click the logout button.	System should logout and redirect user to Login Menu.	As expected.	Pass

Table 5.2.8.1 Logout Testing

CHAPTER 6: CONCLUSION

The purpose of this project is to create a good song recommender system as there is currently a strong market demand in the economy. It is also a great opportunity to have a more in depth understanding about artificial intelligence and machine learning. This project will also be a good way to practice solving problem that arise from implementing artificial intelligence in real life. Firstly, from this project, the best song recommendation software for recommending song precisely will be determined. Furthermore, this project will also create an automatic playlist continuation to automatically create a sequence of recommended track to the user based on the taste in music. Other than that, this project will also solve the cold start problem so that new user and newly added song to a library will be recommended to users somewhat accurately by using popularity-based algorithms. Lastly, this project will solve the problem caused by having a large data set to limited hardware by pre-processing it into a subset containing fewer metadata.

Moreover, song recommender system is a vast and complicated topic that possess great potential in the future of being ever reliant on cloud-based service for our media consumption. Concepts and theories learnt from the creation of this song recommender system could be slightly adjusted and be used for recommender system for movies, tv shows, e-books, and so on. Such methods, on the other hand, are far from flawless and frequently give unacceptable suggestions. This is partly since users' musical preferences and needs are highly dependent on a variety of factors that are not adequately considered in current MRS approaches, which are typically centered on the core concept of user-item interactions, or occasionally content-based item descriptors.

Furthermore, the prototype is done and shown in Chapter 5, the system implementation. The basic GUI and some features have been implemented. The system can show the simple login, logout, song list, play song, add, and remove favourite and refresh the song list as well. The users can enjoy the system without any advertisement compare to other Spotify.

Last but not least, the future work for this project is to provide enough storage for the song to be downloaded into the dataset so that the user can play it immediately rather than relying on YouTube links. Moreover, the system also can work offline in the future as well. There will be more improvement on the quality of the play song generated by the system which will be further developed on the future.

CHAPTER 6 CONCLUSION

In a nutshell, the limitation of the system is a still long way to go and tune up a good algorithm. Different types of problems that need to be solved require different types of algorithms due to the nature of algorithms. There is no one-size-fits-all solution that can solve every problem. The only thing we can do now is continue to work on improving an algorithm. A perfect state is a state of being that is close to flawless, even though perfection is impossible to achieve.

REFERENCES

REFERENCES

- [1] Schedl, M., Zamani, H., Chen, CW. *et al.* (2018, April) "Current challenges and visions in music recommender systems research", *Int J Multimed Info Retr* 7, [Online], 95–116. Available:
<https://link.springer.com/article/10.1007/s13735-018-0154-2/>
- [2] Brain Srebrenik. (2018), "Introduction to Music Recommendation and Machine Learning". [Online]. Available: <https://medium.com/@briansrebrenik/introduction-to-music-recommendation-and-machine-learning-310c4841b01d/>
- [3] Gunawan, A.A. and Suhartono, D. "Music recommender system based on genre using convolutional recurrent neural networks." *Procedia Computer Science*, 157, 1998, pp.99-109.
- [4] Eric Le. (2017). "How to build a simple song recommender system." [Online] Available:
https://towardsdatascience.com/how-to-build-a-simple-song-recommender-296fcbc8c85#:~:text=There%20are%203%20types%20of,%2Dbased%2C%20collaborative%20and%20popularity.&text=Here%20we%20have%20the%20song,%2C%20title%2C%20release%20and%20artist_name/
- [5] Euge Inzaugarat. (2020, Mar). "The keys to creating a collaborative-filtering music recommender system." [Online] Available:
<https://towardsdatascience.com/the-keys-building-collaborative-filtering-music-recommender-65ec3900d19f/>
- [6] Rohit Garg, (2018, Jan). 7 Types of Classification Algorithms. [Online] Available:
<https://analyticsindiamag.com/7-types-classification-algorithms/>
- [7] Garg, S. and Fangyan, S.U.N., (2014, April). "Music Recommender System CS365: Artificial Intelligence." [Online] Available:
<https://cse.iitk.ac.in/users/cs365/2014/submissions/shefalg/project/report.pdf>

REFERENCES

- [8] Harley Maranan, (2021). “Apple music VS Spotify.” [Online] Available:
<https://www.soundguys.com/apple-music-vs-spotify-36833>
- [9] SDLC Overview, 2019. [Online] Available:
https://www.tutorialspoint.com/sdlc/sdlc_overview.htm/
- [10] Jason Brownlee, (2020, July). “Train-Test Split for Evaluating Machine Learning Algorithms.” [Online] Available:
<https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/>
- [11] AnujMehla, 2021. “Python Language advantage and applications.” [Online] Available:
<https://www.geeksforgeeks.org/python-language-advantages-applications/>
- [12] Krzysztof Basel, (2018, June). “Python Pros and Cons.” [Online] Available:
<https://www.netguru.com/blog/python-pros-and-cons/>
- [13] What is Flask Python - Python Tutorial. “What Is Flask Python - Python Tutorial.” [Online] Available:
<https://pythonbasics.org/what-is-flask-python/>
- [14] About Heroku | Heroku. (n.d.). “About Heroku|Heroku” [Online] Available:
<https://www.heroku.com/about>
- [15] Edpresso Team. (2022, Feb). “What is Firebase? Educative: Interactive Courses for Software Developers.” [Online] Available: <https://www.educative.io/edpresso/what-is-firebase>

IEEE International Conference on Computing

User Song Preferences using Artificial Intelligence

Beng Jun Goh
*Department of Computer Science
 (DCS),
 Faculty of Information and
 Communication Technology (FICT),
 Universiti Tunku Abdul Rahman
 (UTAR),
 Kampar, Malaysia
 Bengjun6315@utar.my*

Hoong-Cheng Soong
*Department of Information
 Systems(DIS),
 Faculty of Information and
 Communication Technology (FICT),
 Universiti Tunku Abdul Rahman
 (UTAR),
 Kampar, Malaysia
 soonghc@utar.edu.my*

Ramesh Kumar Ayyasamy
*Department of Information
 Systems(DIS),
 Faculty of Information and
 Communication Technology (FICT),
 Universiti Tunku Abdul Rahman
 (UTAR),
 Kampar, Malaysia
 rameshkumar@utar.edu.my*

Abstract— With the rapid development of mobile devices and more people having internet access, people have access to music collections at an unprecedented scale. Music libraries can easily have more than 15 million songs, people will feel overwhelmed choosing from the ocean of song available. Thus, an efficient song recommender system is necessary for the music service providers and customers. The music streaming companies can attract and retain users with a good recommender system. In the music recommender system field, many music streaming companies are working on building high-precision music recommender system. Thus, this field have a high market demand for good quality music recommendation system. In this research paper, it is the initial stage of the research project using preference learning for the music recommendation system as it has potential to be utilized as optimum user song preferences for the music recommendation systems. This research project conducted preliminary experiment for training data set for few classification models that are Random Forest, Logistic Regression, Support Vector Machine, Naive Bayes, Decision Tree and K-Neighbors for that can be used for the collaborative model. From the accuracy output, it was observed that Logistic Regression gave the highest score that is 18.03% and the Random Forest gave lowest score that is 8.19%.

Keywords—*Music Recommendation System, User Song Preferences, Preference Learning, User-Based Filtering, Collaborative model*

I. INTRODUCTION

Nowadays a lot of music company like Spotify, Apple [1] music is allowing us to access different music resources more conveniently. Most of the music industries are using music recommender systems [2] and the old way of selling music has been replaced by a totally different cloud-based solution. This allow users can listen songs directly from the cloud. However, peoples will experience difficulties choosing from the millions of songs available in such services. In this project, the music recommender system must provide and efficient way to manage songs and help their customers to classify all the songs based on genres, artists, age groups, locations, and language by giving quality recommendation.

Currently, many music industries are working on building high-precision music recommender system. The main goal is to classify the millions of songs in accordance to the taste of the users. These companies are trying to improve their customer's ability to discover relevant music and the quality of the recommender service will attract more customers. With a good recommender system, the user will spend less time to looking the songs and more time on enjoying their favourite song and discovering new favourite. Thus, a good quality

music recommender system will have a strong user base and will create a strong thriving market.

Furthermore, the music recommender system will learn from the user's listening history and recommend song the user will likely enjoy based on their listening pattern [3]. This research project will implement various algorithms to compare the results with one another to figure out which is the most effective algorithm that is suited for a music recommender system. The most common model is popularity-based model because it is simple and intuitive. That another algorithm is collaborative filtering, this algorithm aims to find similarity between users, songs and artists. This experiment will use SVD model, KNN model, based on metadata. The problems face when making this project is a metadata of song are too big and processing such a large dataset is CPU-intensive and require a lot of system memory [4]. So, to overcome system resources limitations, the original dataset is making into a subset for optimum usage. However, the creation of subset will require complex data pre-processing.

The following paragraphs will introduce and review three existing music or song recommendation systems that is available in the market.

A. Spotify

Spotify [5][6] is a digital music, podcast, and video streaming service that gives you access to millions of songs and other content from artists all over the world. Spotify is available on both desktop and mobile applications, which is free of charge to download.



Fig. 1. Recommended function by Spotify. [5][6]

The free version of Spotify has simple functions such as playing music on shuffle and creating playlists. Users can search for songs with the Browse function, as Spotify has a wide range of genres for different occasions. If the users happen to come across a song that catches their ears, they can

save the song into their playlists or simply set the song as one of their "Favourites". Users can also choose to upgrade to Spotify Premium to avoid advertisements. The paid Spotify Premium would also allow you to play songs on-demand, download songs to listen to offline, and skip songs unlimitedly. Furthermore, the other advantage of the Spotify is that the users would get daily and weekly song recommendations from personalized features, such as Discover Weekly, Release Radar, and Daily Mix. The system will collect the data from the users listening pattern to recommend some trending songs or songs from the same artist. Spotify's song base is also multilingual as it has songs in Korean, Japanese, Chinese, English, and Malay. Next, Spotify is available across a range of devices and platforms, including computers, tablets, phones, TVs, speakers, and car infotainment systems and you can easily transition from one to another with Spotify Connect. Users do not have to upgrade to premium to use this function, as it is available without any extra charges or fees.

However, Spotify is not very user friendly to the users who choose to not upgrade to Spotify Premium. There would be ads every 30 minutes, which could be annoying for some users. Without premium, users are only allowed to play in shuffle mode instead of play specific song, and with 6 skips per hour, users may not be able to listen to one specific song. Without Premium, offline listening is unavailable, which means users must have access to the internet to listen to music. The audio quality accessible by free users is also lower than those who have premium. Lastly, some songs of other languages are not available on Spotify, for instance, Spotify would have a complete collection of English albums, but only popular or trending Chinese and Malay songs. Some songs are also only accessible for Premium users.

B. Apple Music

Apple Music [6][7] is a music and video streaming service developed by Apple Inc. Users select music to stream to their device on-demand, or they can listen to existing playlists. Apple Music is available for IOS users only. IOS users can download Apple music from the App Store on their Apple devices.

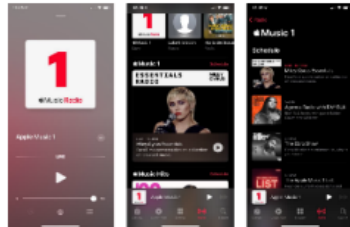


Fig. 2. Apple music interface. [6][7]

Apple music support song sharing features. It is a very good function to allow users to share their favourite song with friends and to social media such as Instagram. This social function enables the promotion of songs as well as the application itself. The next strength of apple music is that it has more music content libraries compared to other music recommendation applications. According to Harley Maranan, Apple music has over 60 million songs, users can find music from the artists they have yet to discover. Apple Music also makes exclusive deals with artists like Taylor Swift and Drake

for early content release, which benefits the anticipating fans. The exclusive content is enabled by Apple's paid-only service, and users are required to pay for the subscription. Moreover, Apple Music has one more strong function which is users can search for songs using only lyrics. This feature is very useful to some users who want to listen to a song but can only remember the lyrics and not the title. Now users can type the lyrics directly into the app's search bar, the system will match the songs to the users accordingly.

On the downside, Apple music does not have an organized and clean user interface. The Apple Music UI does not provide instant access to your own content. The home page should show your recently played songs and playlists, which is what most people want to see when opening a music app. Every part of Apple Music's design should consider putting users' content at the forefront, followed by providing users with suggestions after users have seen what users wanted to see. Last but not least, Apple Music should provide its own voice assistance functionality for the app without Siri. Apple Music can control the next, skip, pause song by using voice assistance functionalities.

C. Last Fm

Last Fm [8][9] can use to track the music you stream by connecting Last.fm to a music service, app or a browser plugin. Last Fm also provides users to view user's stats in real time, receive the weekly reports and access user listening history and so on. Last Fm also allows users to connect streaming services for example Spotify, Imusic, YouTube, Tidal and SOUNDCLOUD. The one of the strengths of Last Fm is to provide free downloadable music tracks when applicable. Some applications will limit users to download songs without any subscription fees. In Last Fm, users can easily download the song for iOS and android users.

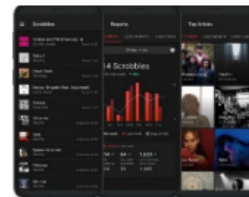


Fig. 3. Lst Fm interface. [8][9]

Moreover, another strong feature is being unable to find an artist or song in Last FM's library, the system will automatically redirect to the YouTube link of the song.

II. FACTS AND FINDINGS

In the study entitled Music Recommender System CS365: Artificial Intelligence [10], Garg & Fangyan (2014) designed, applied and analyzed a song recommendation system. The study is carried out by determining the users' listening patterns and learning from the users' listening history to provide suitable song recommendations according to user preferences.

The algorithms used for this study [10] include the Popularity based Model, Collaborative based Model, SVD Model and KNN Model, and these algorithms are evaluated using Average Precision (mAP) to compare amongst themselves. The results of the SVD based latent factor model is generally better than the popularity-based model, however, the objective functions were prevented from converging to

global optimum due to the sparse. On the other hand, the KNN did not perform well due to the limited song base of only 10,000 songs, which is less than 3 % of the whole dataset.

The results indicated that a memory-based collaborative filtering algorithm would be the most suitable for application in this situation. For the content-based model to work better, more memory and computational power are needed for the available metadata and training dataset to be utilized fully.

1) *Popularity-based Model*: [10] This model is the most basic and simple algorithm. In this algorithm, the popularity of each song is recorded and put into the training set and calculating the number of users who had listened to the song. Then sorted the song in descending order of their popularity. For new users, the system will recommend the top most popular songs except those already in their profile. The disadvantage of the model is no personalization and some songs may never get listened to.

2) *Singular Value Decomposition (SVD) Model*: [10] SVD can collect the listening histories which are influenced by a set of factors (e.g. Genre, artist). These factors are not at all obvious in general, so we need to infer those called latent factors from the dataset. Users and songs will be characterized by latent factors.

3) *k-nearest neighbors (KNN) Model*: [10] KNN can be used to create a space of songs according to their features from the metadata and find out the neighbourhood of each song. The example of features is loudness, mode genre and so more. After creating the feature space, look at all users' profiles and suggest songs which are neighbours to the songs present in his listening history. The KNN is quite personalized and uses metadata.

4) *Evaluation Metrics*: [10] After using the four algorithms, the evaluation metrics can show the compare results with other algorithms. Furthermore, precision is more important than recalling because the false positives will lead to a poor user experience.

A. Problem Formulation

Without a doubt, music is an important element in the lives of mankind. The invention of music-playing websites and applications has made music accessible in the user's daily lives. However, algorithms that would satisfy all users' expectations are yet to be perfected. The main goal is suggesting the best model for song recommendation. This is hard to decide which algorithms are the most suitable model for song recommendation.

The second problem is automatic playlist continuation [11][12]. The task of the automatic playlist generation is referring to the automated creation of the sequences of tracks. Moreover, the automatic playlist generation should add one or many tracks to the playlist which is fits the same target characteristics of the original playlist.

The third problem is cold start problem [13][14]. The meaning of cold start is when a new user register to the system or new song is added into library but the system does not have sufficient data associated with the new song or user. In the cold start problem, the music recommender system is unable to accurately recommend song that is suit the user's preferences due to insufficient user data. When a new song is

added to the library, the recommender system will neglect to recommend the new song to existing user due to no user listening history available.

The research project is using English song as our metadata. The problem faced is the metadata dataset is very large and needs to be broken down into smaller subset. However, this will affect the result of training model due to the subset containing less metadata of songs.

B. Preference Learning

Preference learning [15][16] mentions to the job of learning to forecast an order relation on an assortment of objects. Preference learning as the algorithms [15][16] will be able to access to examples for which the sought order relation is recognized fully or partially in the training phase. Object ranking problems can be determined between label ranking problems and object ranking problems as it depends on the alternatives to be ordered and formal modelling of the preference context [15][16]. Two fundamental approaches [15][16]: directly modelling the binary preference relation or by employing this relation not directly through utility function. Four of the basic learning tasks can be examined [15][16]: Learning Utility Functions, Learning Preference Relations, Model-based Preference Learning and Local Aggregation of Preferences. Active learning paradigm is able to give advantage to preference learning [31]. There are few studies conducted to investigate active preference learning for practical applications for instance collaborative filtering [31]. There are many classification models to classify data in preference learning such as Random Forest, Logistic Regression, Support Vector Machine, Naive Bayes, Decision Tree or K-Neighbors [32]. There are numerous machine learning algorithms able to implement in collaborative filtering. Between them, there are nearest-neighbor, clustering, and matrix factorization [32].

III. PROPOSED METHOD OR FRAMEWORK

A. How to build a simple song recommender system?

In this article by Eric Le (2017) [17], millions of songs are separated into 2 files called triple file and meta file. The triplet file contains `user_id`, `song_id` and listen time. The metadata file contains `song_id`, title, release by and artist name. The article is a helpful tutorial on using item similarity based collaborative filtering model to find similar songs to any songs in our dataset. The article discusses the different applications of 4 models to build the music recommendation system, namely content-based model, collaborative model and popularity model. More models should be used in conducting this study for more accurate and precise results for example use K means clustering, Logistic Regression, Random Forests and so on.

- Data preprocessing [17]: Before take the data to train and test (figure 4) for the algorithms require data pre-processing first. According by Eric Le (figure 5), he separates the metadata to 2 files which is triplet file and meta file. The figure shows to naming the 2 files from the links.

```
triplets_file =
'https://static.turi.com/datasets/millionsong/10000.txt'

songs_metadata_file =
'https://static.turi.com/datasets/millionsong/song_data.csv'
```

Fig. 4. Define files. [17]

```
song_df_2 = pandas.read_csv(songs_metadata_file)
song_df = pandas.merge(song_df_1,
song_df_2.drop_duplicates(['song_id'], on='song_id', how='left'))
```

Fig. 5. Combine 2 files. [17]

This figure 6 shows to combine two files and drop duplicated data by using column `song_id`. The diagram shows the combined data with `song_id`, `user_id`, `listen_count`, `title`, `release` and `artist_name`.

user_id	song_id	listen_count	title	release	artist_name
0	883544d683b0c32127953303d9e4387da9f	1	The Cove	Thicker Than Water	Jack Johnson
1	883544d683b0c32127953303d9e4387da9f	2	Entre Dos Aguas	Flamenco Para Niños	Paco De Lucia
2	883544d683b0c32127953303d9e4387da9f	1	Stanger	Overulation	Kanye West
3	883544d683b0c32127953303d9e4387da9f	1	Constellations	Between Dreams	Jack Johnson
4	883544d683b0c32127953303d9e4387da9f	1	Learn To Fly	There Is Nothing Left To Lose	Foo Fighters

Fig. 6. Visualize combined data. [17]

- Data transformation [17]: After preprocessing the data, the next step (figure 7) is data transformation. Doing transformation allows to simplify the dataset and make it easier and simpler to understand.

```
song_grouped = song_df.groupby(['song']).agg(['listen_count':
'count']).reset_index()
grouped_sum = song_grouped['listen_count'].sum()
song_grouped['percentage'] =
song_grouped['listen_count'].div(grouped_sum)*100
song_grouped.sort_values(['listen_count', 'song'], ascending = [0,1])
```

Fig. 7. Data transformation. [15]

- Output [17]: The first line of the code (figure 7) is grouping the `song_df` by the number of listen count in ascending. The second line is calculating sum of `listen_count` of each song into `grouped_sum` variable. The third line is creating a new column name called `percentage` and calculate the value by dividing the `listen_count` by the sum of `listen_count` and multiply by 100. The last line is sorting the sequence and output (figure 8) in the ascending order.

song	listen_count	percentage
3660 Sethr Kosmitch - Harmonia	45	0.45
4678 Undo - Björk	32	0.32
5165 You're The One - Dwight Yoakam	32	0.32
1071 Dog Days Are Over (Radio Edit) - Florence + Th...	28	0.28
3655 Secrets - OneRepublic	28	0.28
4378 The Scientist - Coldplay	27	0.27
4712 Use Somebody - Kings Of Leon	27	0.27
3476 Revelry - Kings Of Leon	26	0.26

Fig. 8. Output of data transformation. [17]

1) *SVD model*: [17] This diagram (figure 9) shows the U represents user vector, S represents item vector. Vt is the joint of these U and S as collection of points in 2 dimensional spaces. The vectors are going to measure the distance of one preferences of user to another preferences of user. The first

line is use to define the `computeEstimatedRatings` for the test user. The following code is use for convert the vector to dense format to get the indices of the movie with best ratings.

```
#Compute estimated rating for the test user
def computeEstimatedRatings(um, U, S, Vt, uTest, K, test):
    rightTerm = S*Vt

    estimatedRatings = np.zeros(shape=(UMX_UID, UMX_PID),
dtype=np.float64)
    for userTest in uTest:
        prod = U[userTest, :]*rightTerm
        #we convert the vector to dense format in order to get the
#indices
        #of the movies with the best estimated ratings
        estimatedRatings[userTest, :] = prod.todense()
        recom = (-estimatedRatings[userTest, :]).argsort()[1:250]
    return recom
```

Fig. 9. SVD code. [17]

2) *SVD model*: [17] This Collaborative based model can be classified by user based and item based model (figure 10). By Eric Le (2017) [17], he used item similarity based model. This code shows it will get a unique count of `user_id` for each song and name it as recommendation score. The function then receive a `user_id` and use cooccurrence matrix to count the top 10 recommended songs. The recommendation function will show the top ten recommended song for given `user_id`.

```
#Create the item similarity based recommender system model
def create(self, train_data, user_id, item_id):
    self.train_data = train_data
    self.user_id = user_id
    self.item_id = item_id

#Use the item similarity based recommender system model to
#make recommendations
def recommend(self, user):
    #####
    #A. Get all unique songs for this user
    #####
    user_songs = self.get_user_items(user)

    print("No. of unique songs for the user: %d" % len(user_songs))

    #####
    #B. Get all unique items (songs) in the training data
    #####
    all_songs = self.get_all_items_train_data()

    print("No. of unique songs in the training set: %d" % len(all_songs))

    #####
    #C. Construct item cooccurrence matrix of size
    #len(user_songs) X len(songs)
    #####
    cooccurrence_matrix = self.construct_cooccurrence_matrix(user_songs, all_songs)
```

Fig. 10. Item based model. [17]

This code (figure 10) shows it will get a unique count of `user_id` for each song and name it as recommendation score. The function then receives a `user_id` and use cooccurrence matrix to count the top 10 recommended songs. The recommendation function will show (figure 10) the top ten recommended song for given `user_id`.

user_id	song	score	Rank
2194 4bc989bf25293a78bbd9487e74018faa570e5cf	Sethr Kosmitch - Harmonia	37	1
4083 4bc989bf25293a78bbd9487e74018faa570e5cf	Undo - Björk	27	2
601 4bc989bf25293a78bbd9487e74018faa570e5cf	Dog Days Are Over (Radio Edit) - Florence + Th...	26	3
4443 4bc989bf25293a78bbd9487e74018faa570e5cf	You're The One - Dwight Yoakam	24	4
3034 4bc989bf25293a78bbd9487e74018faa570e5cf	Revelry - Kings Of Leon	21	5
3189 4bc989bf25293a78bbd9487e74018faa570e5cf	Secrets - OneRepublic	21	6
4112 4bc989bf25293a78bbd9487e74018faa570e5cf	Use Somebody - Kings Of Leon	21	7
1207 4bc989bf25293a78bbd9487e74018faa570e5cf	Fireflies - Chvrches Karaoke	20	8
1577 4bc989bf25293a78bbd9487e74018faa570e5cf	Hey, Soul Sister - Train	19	9
1628 4bc989bf25293a78bbd9487e74018faa570e5cf	Horn Concerto No. 4 in E Flat Major, Op. 35 - Franz...	19	10

Fig. 11. Top 10 recommend songs. [17]

B. Initial Proposed Framework

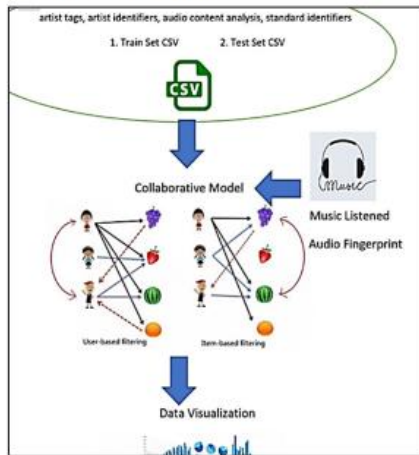


Fig. 12. Proposed Brief Framework.

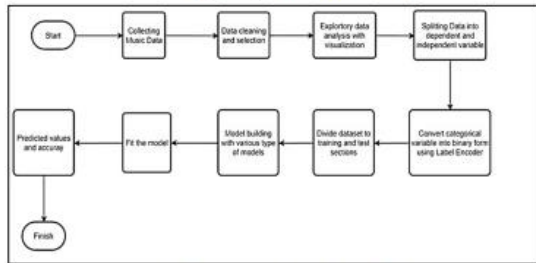


Fig. 13. Proposed Methodology Research Flowchart.

In this research project as in the proposed brief framework in figure 12 and figure 13 as the research flowchart, the dataset includes metadata for example artist tags, artist identifiers, audio content analysis, standard identifiers. After that, find correlations between the songs and users from the dataset. The dataset will divide into 2 subsets namely train set and test set and store in CSV files for pandas. Collaborative based model is used because collaborative method can be either user-based or item-based. Then stochastic aggregation is used to combine them. Stochastic aggregation is randomly choosing one of them according top scored items and their probability distribution. The reason for not choosing SVD is because no enough memory and computational power to use the whole metadata and dataset. The collaborative based model can collect the information from many users and then make a prediction based on similarity measures between items and users. Which is classified into item-based and user-based models. In the item-based model, it will collect the songs that are often listened to together by some users tend to be similar and some users are more likely to be listened together in the future. In the user-based model, it will collect users who have similar listening histories. Some users will probably listen to the same songs in future because some

songs have listened to in the past. Audio fingerprinting [18] or also known as acoustic fingerprinting is condensed digital summary from an audio files to find the audio samples especially the song titles and artists as to locate similar samples in the audio databases. Therefore, it is best to capture the user listened songs and by using the collaborative model from recommendation methods. In the end, the list of songs will be suggested in an interactive visualization as the final output for the users.

C. Experiment of Training Dataset for Different Classification Models

In this experiment, the classification models for training the dataset are Random Forest, Logistic Regression, Support Vector Machine, Naive Bayes, Decision Tree or K-Neighbors.

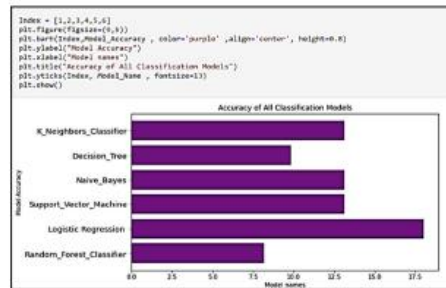


Fig. 14. Accuracy Output for the Five Classification Model.

```
print(classification_report(y_test, Predicted_Value_of_Logistic_Regression))
```

	precision	recall	f1-score	support
Bad Romance	0.00	0.00	0.00	9
Counting Stars	0.21	1.00	0.35	3
Dark Horse	0.18	0.25	0.21	8
Diamonds	0.25	0.40	0.31	5
Firework	0.00	0.00	0.00	6
Roar	0.00	0.00	0.00	4
See You Again	0.00	0.00	0.00	7
Someone Like You	0.29	0.29	0.29	7
Wake Me Up	0.17	0.67	0.27	3
We Found Love	0.00	0.00	0.00	7
accuracy			0.18	61
macro avg	0.11	0.26	0.14	61
weighted avg	0.10	0.18	0.12	61

Fig. 15. Precision, F1-Score and Recall for the Songs.

The accuracy obtained in figure 14: Random Forest (8.19%), Logistic Regression (18.03%), Support Vector Machine (13.11%), Naive Bayes (13.11%), Decision Tree (9.83%) or K-Neighbors (13.11%). The F1-score, precision and recall (figure 15) for the songs are not right either that is normally lower than 0.5 with the exception for one song that is Counting Star with the perfect recall 1.0. To increase the accuracy, it is suggested to separate data to hundred datasets as subset as million songs are time consuming and heavy processing. Therefore, preference learning with proper collaborative model will be used to solve the problems above.

IV. CONCLUSION

This research project is to give insight on how to create a good song recommender system as there is currently a strong market demand in the economy. It is also a great opportunity to have a more in depth understanding about artificial intelligence and machine learning as this research paper can

be considered as brief survey research paper or abbreviated review research paper for music recommendation system. This research project will also be a good way to practice to solve the problem arisen from implementing artificial intelligence in real life. Firstly, from this research project, the best song recommendation software for recommending song precisely is possible to be determined. Furthermore, this research project will also create an automatic playlist continuation that automatically creating a sequence of recommended track to the user based on their taste in music. Other than that, this research project will also solve the cold start problem [19] so that new user and newly added song to a library will be recommended to users somewhat accurately by using popularity-based algorithms. Lastly, this research project has the possibility to solve the problem caused by having a large data set [20] to limited hardware by pre-processing it into a subset containing fewer metadata. In a nut shell, song recommender system is a vast and complicated topic [21] that possess great potential in the future [22] of being ever reliant on cloud-based service for our media consumption. Concepts and theories learnt from the creation of this song recommender system could be slightly adjusted and be used for recommender system for movies, tv shows, e-books, and so on. On the interesting note, the music is often associated with emotions and thus it is possible to apply sentiment analysis as the combination methods for the song recommendation system [23]. For the reviews of sentiment analysis, you may refer to these papers [24][25]. To conclude, deep learning is emerging as new method from the machine learning and it can be utilized into the music recommendation system [26][27][28][29][30].

ACKNOWLEDGMENT

I would like to thank my supervisor, Ts. Soong Hoong Cheng and other members Faculty of Information and Communications Technology (FICT) in Universiti Tunku Abdul Rahman (UTAR) for the support.

REFERENCES

- [1] V. Chang, Y. Yang, Q. A. Xu, and C. Xiong, "Factors influencing consumer intention to subscribe to the premium music streaming services in China," *J. Glob. Inf. Manag.*, vol. 29, no. 6, pp. 1–25, 2021.
- [2] M. Kleč and A. Wiczkowska, "Music recommendation systems: A survey," in *Recommender Systems for Medicine and Music*, Cham: Springer International Publishing, 2021, pp. 107–118.
- [3] Y. Deldjoo, M. Schedl, and P. Knees, "Content-based music recommendation: Evolution, state of the art, and challenges," *arXiv [cs.LG]*, 2021.
- [4] C. Roy and S. S. Rautaray, "Challenges and issues of recommender system for big data applications," in *Studies in Computational Intelligence*, Singapore: Springer Singapore, 2021, pp. 327–341.
- [5] I. Gomes, I. Pereira, I. Soares, M. Antunes, and M. Au-Yong-Oliveira, "Keeping the beat on: A case study of spotify," in *Advances in Intelligent Systems and Computing*, Cham: Springer International Publishing, 2021, pp. 337–352.
- [6] B. Manolovitz and M. Oghira, "Repeated listens in the music discovery process," in *Recommender Systems for Medicine and Music*, Cham: Springer International Publishing, 2021, pp. 119–134.
- [7] O. Yessenbayev, Y. Ismailov, J. Youn, and J. Han, "The Influence of Cultural and Demographic Factors on Improving Music Streaming Experience: Apple Music Case Study," 2021, pp. 276–282.
- [8] Z. Yang *et al.*, "Fusion of internal similarity to improve the accuracy of recommendation algorithm," *Journal on Internet of Things*, vol. 3, no. 2, pp. 65–76, 2021.
- [9] T. Kang, H. Lee, B. Choe, and K. Jung, "Entangled bidirectional encoder to autoregressive decoder for sequential recommendation," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021.
- [10] F. S. Shefali Garg, "Music Recommender System CS365: Artificial Intelligence," 2014, pp. 1–6.
- [11] C.-W. Chen, P. Lamere, M. Schedl, and H. Zamani, "Recsys challenge 2018: Automatic music playlist continuation," in *Proceedings of the 12th ACM Conference on Recommender Systems - RecSys '18*, 2018.
- [12] H. Zamani, M. Schedl, P. Lamere, and C.-W. Chen, "An analysis of approaches taken in the ACM RecSys challenge 2018 for automatic music playlist continuation," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 5, pp. 1–21, 2019.
- [13] J. Misztal-Radecka, B. Indurkha, and A. Smywinski-Pohl, "Meta-User2Vec model for addressing the user and item cold-start problem in recommender systems," *User Model. User-adapt Interact.*, vol. 31, no. 2, pp. 261–286, 2021.
- [14] J. Feng, Z. Xia, X. Feng, and J. Peng, "RBPR: A hybrid model for the new user cold start problem in recommender systems," *Knowl. Based Syst.*, vol. 214, no. 106732, p. 106732, 2021.
- [15] C. Sammut and G. I. Webb, Eds., *Encyclopedia of machine learning and data mining*. New York, NY: Springer, 2017.
- [16] J. Furnkranz and E. Hüllermeier, "Preference Learning: An Introduction," in *Preference Learning*, S. Greco, M. Ehrgott, J. Figueira, Ed. Springer-Verlag, 2010, pp. 1–17.
- [17] E. Le, "How to build a simple song recommender system," *Towards Data Science*, 24-Apr-2017. [Online]. Available: <https://towardsdatascience.com/how-to-build-a-simple-song-recommender-296fbc8c85>. [Accessed: 15-Oct-2021].
- [18] N. Struharová, "Performance of the Dejavu audio fingerprinting framework in music identification in movies," 2021.
- [19] U. Kuźelewska, "Quality of recommendations and cold-start problem in recommender systems based on multi-clusters," in *Computational Science – ICCS 2021*, Cham: Springer International Publishing, 2021, pp. 72–86.
- [20] V. Chaturvedi, A. B. Kaur, V. Varshney, A. Garg, G. S. Chhabra, and M. Kumar, "Music mood and human emotion recognition based on physiological signals: a systematic review," 2021, pp. 1–24.
- [21] M. Ghajargar and A. M. Schröder, "Unboxing the Algorithm: Designing an Understandable Algorithmic Experience in Music Recommender Systems," 2021.
- [22] Y. Huo, "Music personalized label clustering and recommendation visualization," *Complexity*, vol. 2021, pp. 1–8, 2021.
- [23] S. J. Lee, B. G. Seo, and D. H. Park, "Development of Music Recommendation System based on Customer Sentiment Analysis," *Journal of Intelligence and Information Systems*, vol. 24, no. 4, pp. 197–217, 2018.
- [24] H.-C. Soong, R. K. Ayyasamy, and R. Akbar, "A review towards deep learning for sentiment analysis," in *2021 International Conference on Computer & Information Sciences (ICCOINS)*, 2021.
- [25] H. C. Soong, N. B. A. Jilil, R. K. Ayyasamy, and R. Akbar, "The essential of sentiment analysis and opinion mining in social media: Introduction and survey of the recent approaches and techniques," 2019, pp. 272–277.
- [26] X. Wen, "Using deep learning approach and IoT architecture to build the intelligent music recommendation system," *Soft Comput.*, vol. 25, no. 4, pp. 3087–3096, 2021.
- [27] R. Anand, R. S. Sabeenian, D. Gurang, R. Kirthika, and S. Rubeeena, "AI based Music Recommendation system using Deep Learning Algorithms," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 785, no. 1, p. 012013, 2021.
- [28] S. R. Chavare, C. J. Awati, and S. K. Shrivage, "Smart Recommender System using Deep Learning," in *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, 2021.
- [29] L. Haiming, W. Kaili, S. Yunyun, and M. Xuefeng, "Application of recommendation systems based on deep learning," in *Recent Challenges in Intelligent Information and Database Systems*, Singapore: Springer Singapore, 2021, pp. 85–97.
- [30] M. S. Fathollahi and F. Razzazi, "Music similarity measurement and recommendation system using convolutional neural networks," *International Journal of Multimedia Information Retrieval*, vol. 10, no. 1, pp. 43–53, 2021.
- [31] A. Birlutiu, *Machine learning for pairwise data: applications for preference learning and supervised network inference*. SI: sn, 2011.
- [32] P. Sagar, P. P., and I. L., "Analysis of Prediction Techniques based on Classification and Regression," *Int. J. Comput. Appl.*, vol. 163, no. 7, pp. 47–51, 2017.

Professional Conference Research Presentation ICOCO 2021

The screenshot shows a Zoom meeting interface. At the top, there is a Zoom Meeting title bar with a search bar for participants. Below this is a video gallery showing several participants, including Hoong-Cheng Soong, Beng Jun Goh, and others. The main content area displays a presentation slide titled "Introduction" for the IEEE International Conference on Computing (ICOCO 2021). The slide lists topics such as "Music Recommendation System, User Song Preferences, Preference Learning, User-Based Filtering, Collaborative model" and "Known System: Spotify, Apple Music, Last FM." To the right of the slide is a chat window with messages from participants, including a question about programming language and a response. At the bottom, there is a banner for Universiti Tunku Abdul Rahman (UTAR) with its logo and motto "Broadening Horizons Transforming Lives".

Zoom Meeting

Participants (12)

Find a participant

Hoong-Cheng Soong | UTAR (Me) [M] [V] [A]

Track B: IEEE CS Mala... (Host) [M] [V] [A]

Beng Jun Goh | UTAR [M] [V] [A]

Arcely Napalit | TIP [M] [V] [A]

Badrul Anuar [M] [V] [A]

Festus Uwasonba | Sunway Uni... [M] [V] [A]

Invite Unmute Me

Chat

good presentation

From Shilan Sameen Hameed to Everyone:

Dr Nuzli Mohamad Anas , which programming language used for implementing the algorithm?

thanks and well done

Who can see your messages? Recording On

To: Everyone [V] [A]

Type message here...

UTAR UNIVERSITI TUNKU ABDUL RAHMAN DU012(A) 拉曼大學 (優大) Wholly owned by UTAR Education Foundation Co. No. 578227-M Broadening Horizons Transforming Lives 德智体兼修 群美新并重

Type here to search

9:54 AM 19-Nov-21

Certificate of Presentation

IEEE International Conference on Computing (ICOCO 2021)



CERTIFICATE OF PRESENTATION

for the paper titled:

User Song Preferences Using Artificial Intelligence

has been presented at the IEEE International Conference on Computing (ICOCO 2021), held via virtual conference from November 17 – 19, 2021.

This paper is presented by:

Beng Jun Goh

A handwritten signature in black ink, appearing to read 'Ali Selamat', written over a horizontal line.

Prof. Ts. Dr. Ali Selamat,
General Chair

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.:2
Student Name & ID: Goh Beng Jun 18ACB05435	
Supervisor: Ts Soong Hoong Cheng	
Project Title: User Song Preferences using Preference Learning in Artificial Intelligence	

1. WORK DONE

- i. Research on how to host the project become public web services and markup language.

2. WORK TO BE DONE

- i. Host the project become public web services.
- ii. Design the website with markup language.

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

- i. Progress is slightly slow as more time need to be research.



Online Signature using COVID-19
SOONG HOONG CHENG (Lecturer)
FACULTY OF ICT
UNIVERSITI TUNKU ABRAHAM (UTAR)
PERAK CAMPUS, 3, JALAN UNIVERSITI, BANDAR BARU,
31000 KAMPAR.

Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.:4
Student Name & ID: Goh Beng Jun 18ACB05435	
Supervisor: Ts Soong Hoong Cheng	
Project Title: User Song Preferences using Preference Learning in Artificial Intelligence	

1. WORK DONE

- i. Done coding for the simple implementation and GUI.

2. WORK TO BE DONE

- i. Implement the Login and Logout functions.

3. PROBLEMS ENCOUNTERED

Convert the python file become pickle file so that can use the train model.

4. SELF EVALUATION OF THE PROGRESS

- i. Progress is slightly slow as more time need to be survey.



Online Signature during COVID-19
SOONG HOONG CHENG (Lecturer)
FACULTY OF ICT
UNIVERSITI SAINS AJLUN, BAHAWU LITAM,
PERAK CAMPUS, AJLUN LAUNDAU, BAHAWU BIRAH,
34300 KAMPAR.

Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT (Project II)

Trimester, Year: Y3S3	Study week no.:6
Student Name & ID: Goh Beng Jun 18ACB05435	
Supervisor: Ts Soong Hoong Cheng	
Project Title: User Song Preferences using Preference Learning in Artificial Intelligence	

1. WORK DONE

- i. Done the project architecture.

2. WORK TO BE DONE

- i. Save the user's results into Firebase.

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

- i. Work is on track but discovered issues are pending to solve.


Online Signature during COVID-19
TSOONG HOONG CHENG (Supervisor)
FACULTY OF ICT
UNIVERSITI TERAHUTU TEMAHAN (UTAR)
PERAK CAMPUS, 31100 UNIVERSITI, BANDAR BARU,
31000 KAMPAR.

Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.:8
Student Name & ID: Goh Beng Jun 18ACB05435	
Supervisor: Ts Soong Hoong Cheng	
Project Title: User Song Preferences using Preference Learning in Artificial Intelligence	

1. WORK DONE

- i. Working on new functions.

2. WORK TO BE DONE

- i. Done the add and remove songs from favourite list.
- ii. Reconstruct and writing the report.

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

- i. Progress is delayed due to workloads from other courses.



SOONG HOONG CHENG (Lecturer)
FACULTY OF ICT
UNIVERSITI TUNKU ABRAHAM (UTAR)
PERAK CAMPUS, JLN SIKERBISTE, BANGSAI BIRAH,
3100 KAMPAR.

Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.:10
Student Name & ID: Goh Beng Jun 18ACB05435	
Supervisor: Ts Soong Hoong Cheng	
Project Title: User Song Preferences using Preference Learning in Artificial Intelligence	

1. WORK DONE

- i. Review and modify the FYP 1.
- ii. Checking and testing the overall of the whole system.

2. WORK TO BE DONE

- i. Writing report.
- ii. Fix the bugs of the projects.

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

- i. Progress is delayed due to workloads from other courses.



SOONG HOONG CHENG (Lecturer)
FACULTY OF ICT
UNIVERSITI TUNKU ABRAHAM (UTAR)
PUNJIT CAMPUS, 35100 KAMPAR, PERAK

Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	Study week no.:12
Student Name & ID: Goh Beng Jun 18ACB05435	
Supervisor: Ts Soong Hoong Cheng	
Project Title: User Song Preferences using Preference Learning in Artificial Intelligence	

1. WORK DONE

- i. Continue writing on report.
- ii. Model testing.

2. WORK TO BE DONE

- i. Complete FYP 2 reports.
- ii. Code rearrangement.

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

- i. Work is on track.



Online Signature during COVID-19
SOONG HOONG CHENG (Lecturer)
FACULTY OF ICT
UNIVERSITI TUNKU ABDOLLAH RAHMAN (UTAR)
PENANG CAMPUS, 11A UNIVERSITY AVENUE, BANDAR BARU,
11800 KAMPAR.

Supervisor's signature



Student's signature

Poster

USER SONG PREFERENCES USING PREFERENCE LEARNING IN ARTIFICIAL INTELLIGENCE

Introduction

This project will implement various algorithms to compare the results with one another to figure out which is the most effective algorithm and focus on the development of the recommender system using Python. Users will enjoy the song which is recommended by the system.

Methodology

The system is developed using Jupyter Notebook and Python language and integrated by the Streamlit services. Moreover, the development of the system uses the SDLC model, namely Software Development Life Cycle.

Discussion

The system will be designed as simple as well for the users. Moreover, the system will be free to all users. It will simplify the step and users can enjoy the songs easily.

Conclusion

The project is successfully developed and users enjoy the songs recommended by the system. There will be more improvement on the features by the system which will be further developed during the final year of project 2.

Bachelor of Computer Science(Hons)
Final Year project

Project by: Goh Beng Jun
Project Supervisor: Ts Soong Hoong Cheng

PLAGIARISM CHECK RESULT

Turnitin Originality Report

Processed on: 18-Apr-2022 13:17 +08
 ID: 1813246007
 Word Count: 8220
 Submitted: 1

FYP2 Check By Goh Beng Jun

[Document Viewer](#)

Similarity by Source	
Internet Sources:	13%
Publications:	3%
Student Papers:	16%

[include quoted](#)
 [include bibliography](#)
 [excluding matches < 5 words](#)
 mode: quickview (classic) report
 Change mode
 [print](#)
 [download](#)

<p>2% match (student papers from 24-Apr-2014) Submitted to Indian Institute of Technology, Kanpur on 2014-04-24</p>
<p>2% match (Internet from 17-Sep-2020) https://towardsdatascience.com/how-to-build-a-simple-song-recommender-296fcbc8c85?gi=7139e611406e</p>
<p>2% match (Internet from 29-Sep-2021) http://www.ir.juit.ac.in:8080</p>
<p>1% match (Internet from 04-Dec-2020) https://www.soundguys.com/apple-music-vs-spotify-36833/</p>
<p>1% match (publications) Rahul Kataraya, Om Prakash Verma. "Efficient music recommender system using context graph and particle swarm", Multimedia Tools and Applications, 2017</p>
<p>1% match (student papers from 16-Apr-2020) Submitted to QA Learning on 2020-04-16</p>
<p>1% match (Internet from 12-Jan-2021) https://support.spotify.com/us/article/what-is-spotify/</p>
<p>1% match (Internet from 03-Feb-2022) https://www.studymode.com/essays/Miss-1891340.html</p>
<p>1% match (Internet from 14-Feb-2022) http://dspace.daffodilvarsity.edu.bd:8080</p>
<p>1% match (Internet from 14-Nov-2020) https://www.netguru.com/blog/python-pros-and-cons</p>



Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author: **Beng Jun Goh**
Assignment title: **FYP Check**
Submission title: **FYP2 Check**
File name: **FYP_check.docx**
File size: **56.94K**
Page count: **29**
Word count: **8,220**
Character count: **42,757**
Submission date: **18-Apr-2022 01:16PM (UTC+0800)**
Submission ID: **1813246007**

Normally a lot of music companies like Spotify, Apple music are allowing us to access different music resources more conveniently. Most of the music related to our using music recommender systems and the old way of writing music has been replaced by a newly different cloud-based solution. The artists users can listen songs directly from the cloud. However, people will experience difficulties choosing from the millions of songs available in such services. In this project, the music recommender system uses provide and efficient way to manage songs and help their customers to identify all the songs based on genre, artists, age groups, location, and language by using quality recommendation. (Marko,2018)

Currently, many music industries are working on building high-precision music recommender systems. The main goal is to identify the millions of songs in accordance to the taste of the users. These companies are trying to improve their customer's ability to discover relevant music and the quality of the recommender service will attract more customers. (Fadhil, 2018) With a good recommender system, the user will spend less time in looking the songs and more time on enjoying their favorite songs and discovering new favorites. Thus, a good quality music recommender system will have a strong user base and will create a strong listening market. (Zaman,2019)

Furthermore, the music recommender system will learn from the user's listening history and recommend song the user will likely enjoy based on their listening pattern. This project will implement an item algorithm to compare the music with our similar to figure out which is the most effective algorithm that is suited for a music recommender system. The most common model is popularity-based model because it is simple and intuitive. The another algorithm is collaborative filtering. This algorithm aims to find similarity between users, songs and artists. This experiment will use SVT model, KNN model, based on metadata. The problems face when making the project is a hundreds of song are being and processing such a large dataset is CPU intensive and require a lot of system memory. So, to overcome system resource limitations, the original dataset is made into a subset for optimum usage. However, the creation of subset will require complex data pre-processing. (Li,2017)

Without a doubt, music is an important element to the lives of mankind. The invention of music playing solutions and applications has made music accessible to the user's daily lives. With the rapid development of mobile devices and more people having internet access, people have access to music collections of an unprecedented scale. Most libraries can easily have more than 10 million songs, people will find overwhelming choosing from the ocean of song

PLAGIARISM CHECK RESULT

FYP2 Check

ORIGINALITY REPORT

19%	13%	3%	16%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Indian Institute of Technology, Kanpur Student Paper	2%
2	towardsdatascience.com Internet Source	2%
3	www.ir.juit.ac.in:8080 Internet Source	2%
4	www.soundguys.com Internet Source	1%
5	Rahul Katarya, Om Prakash Verma. "Efficient music recommender system using context graph and particle swarm", Multimedia Tools and Applications, 2017 Publication	1%
6	Submitted to QA Learning Student Paper	1%
7	support.spotify.com Internet Source	1%
8	www.studymode.com Internet Source	1%

PLAGIARISM CHECK RESULT

9	dspace.daffodilvarsity.edu.bd:8080 Internet Source	1 %
10	www.netguru.com Internet Source	1 %
11	Submitted to Liverpool Community College Student Paper	<1 %
12	Submitted to University of Technology, Sydney Student Paper	<1 %
13	Submitted to University of Westminster Student Paper	<1 %
14	Submitted to Georgia Gwinnett College Student Paper	<1 %
15	Submitted to Kingston University Student Paper	<1 %
16	analyticsindiamag.com Internet Source	<1 %
17	Submitted to University of Greenwich Student Paper	<1 %
18	Submitted to Segi University College Student Paper	<1 %
19	Submitted to Softwarica College Of IT & E- Commerce Student Paper	<1 %

PLAGIARISM CHECK RESULT

20	www.researchgate.net Internet Source	<1 %
21	Submitted to Jaypee University of Information Technology Student Paper	<1 %
22	Sejal Budhani, Roshni Kataria, Mahek Nagdev, Shikhar Niranjani, Pallavi Saindane. "Chapter 34 Moelleux—Music Recommendation System", Springer Science and Business Media LLC, 2022 Publication	<1 %
23	Submitted to The British College Student Paper	<1 %
24	Submitted to American Intercontinental University Online Student Paper	<1 %
25	Submitted to Embry Riddle Aeronautical University Student Paper	<1 %
26	www.analyticsvidhya.com Internet Source	<1 %
27	Submitted to University of Sunderland Student Paper	<1 %
28	Submitted to Indian School of Business Student Paper	<1 %

29	Submitted to Brigham Young University, Hawaii Student Paper	<1 %
30	thesai.org Internet Source	<1 %
31	link.springer.com Internet Source	<1 %
32	Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, Mehdi Elahi. "Current challenges and visions in music recommender systems research", International Journal of Multimedia Information Retrieval, 2018 Publication	<1 %
33	Submitted to National Institute of Technology, Kurukshetra Student Paper	<1 %
34	Submitted to Sydney Institute of Technology and Commerce Student Paper	<1 %
35	Submitted to Whitireia Community Polytechnic Student Paper	<1 %
36	Submitted to Charles Sturt University Student Paper	<1 %
37	Submitted to Colorado Technical University Online Student Paper	<1 %

PLAGIARISM CHECK RESULT

38 Submitted to American University in Dubai <1 %
Student Paper

39 Submitted to University of Sussex <1 %
Student Paper

40 ijedr.org <1 %
Internet Source

41 markets.pe.com <1 %
Internet Source

42 Submitted to Harrisburg University of Science and Technology <1 %
Student Paper

Exclude quotes On
Exclude bibliography On

Exclude matches < 5 words

PLAGIRISM CHECK RESULT

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 20/04/2022	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Goh Beng Jun
ID Number(s)	18ACB05435
Programme / Course	Computer Science
Title of Final Year Project	User Song Preferences using Preference Learning in Artificial Intelligence

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>19</u> % Similarity by source Internet Sources: <u>13</u> % Publications: <u>3</u> % Student Papers: <u>16</u> %	1. Some direct quotes are needed to explain the sentences. 2. The headings, footers, chapters used similar names such as Chapter 1: Introduction. 3. Some methods, formula and algorithms names/explanations are the same and fixed.
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.



Signature of Supervisor

Name: Ts Soong Hoong Cheng

Date: 20 April 2022

Signature of Co-Supervisor

Name: Wong Chee Siang

Date: 20 April 2022



UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	18ACB05435
Student Name	Goh Beng Jun
Supervisor Name	Ts Soong Hoong Cheng

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Plastic Cover (for hardcopy)
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)

Date: 20-04-2022