Dr. B. C. Roy Engineering College, Durgapur

Subject: Software Engineering Lab

Subject code: ESC 591

Semester: 5th

Academic Year: 2024-25

Topic: Music Recommendation System Using Machine Learning

Student Name	**University Roll No.**
1. Sandip Das	12030522014
2. Samar Manna	12030522012
3. Avijit Bauri	12030522005
4. Anik Kumar Sinha	12030522042
5. Mehuli Lahiri	12030522027

Abstract

The Music Recommendation System aims to revolutionize the way users discover and engage with music by leveraging advanced machine learning techniques such as collaborative filtering, content-based filtering, and deep learning models. This system analyses user listening patterns, preferences, and song metadata (e.g., genre, tempo, mood) to deliver highly personalized, real-time music suggestions.

Designed with scalability and performance in mind, the system ensures seamless integration into music streaming platforms, enhancing user engagement and satisfaction. Key deliverables include a robust recommendation engine, an intuitive web interface, and comprehensive documentation for development and maintenance.

Beyond personalization, the system focuses on introducing users to diverse music, fostering exploration across genres, artists, and albums. By continuously adapting to user behavior, it ensures recommendations evolve with individual preferences, providing a dynamic and engaging experience. This holistic approach not only enriches the user's music journey but also drives revenue growth and retention for streaming platforms.

Success will be measured through system accuracy (85% relevance to user preferences), scalability (efficiently handling large-scale data), and user satisfaction (80% positive feedback). The project encompasses stages of requirement analysis, architectural design, development, integration, testing, and deployment, culminating in a tool that transforms the music discovery experience for users worldwide.

TABLE OF CONTENT

LIST OF FIGURES

Chapter -1

Introduction

1.1 Problem Statement

With the growing availability of music streaming platforms, users face challenges in discovering songs that align with their unique preferences. Generic playlists often fail to capture personal tastes, leading to reduced user engagement and satisfaction. A robust solution is needed to deliver tailored music recommendations that evolve with users' listening habits.

1.2 Motivation

Music is deeply personal and plays a significant role in users' lives, influencing their moods, productivity, and emotions. The motivation for this project stems from the desire to create a system that not only enhances user experience but also drives engagement and retention on streaming platforms by providing personalized music suggestions.

1.3 Objectives

The primary objective of this project is to develop a music recommendation system that:

1. Delivers personalized song suggestions using machine learning techniques.
2. Introduces users to new music while aligning with their preferences.
3. Ensures scalability, accuracy, and efficiency to support large user bases.

1.4 Scope of the Project

The system will analyze user behavior and song attributes, such as genre, tempo, and mood, to provide real-time recommendations. It will integrate seamlessly into music streaming platforms, offering a user-friendly interface for exploring new music. The solution will also prioritize scalability to handle millions of users, ensuring long-term viability for large-scale adoption

Chapter-2

Defining

2.1 Project Definition:

The Music Recommendation System is designed to provide personalized, accurate, and diverse song recommendations to users by analyzing their listening behavior and music preferences. Utilizing machine learning techniques like collaborative filtering, content-based filtering, and deep learning models, the system offers dynamic suggestions based on user habits and song attributes such as genre, tempo, mood, and popularity.

2.2 Project Goals:

- **Personalized Recommendations:** Create a system that delivers tailored music suggestions, ensuring relevance and user engagement.
- **Music Discovery:** Introduce users to new and diverse music that aligns with their tastes, encouraging exploration beyond their usual preferences.
- **Scalability:** Ensure the system can handle a vast number of users and songs, offering seamless performance at scale.

2.3 Deliverables:

- A music recommendation engine using machine learning algorithms to analyze user behavior and song metadata.
- A web interface that integrates seamlessly with music platforms to deliver recommendations.
- Documentation detailing the setup, system architecture, and maintenance for developers.

Chapter – 3

System requirement

3.1 Hardware Requirements:

- **Processor:** Intel Core i7 or equivalent
- **RAM:** Minimum 8GB (16GB recommended)
- **Storage:** SSD with at least 512 GB

3.2 Software Requirements:

- **Operating System:** Windows 10 or Linux
- **Dataset:** Kaggle
- **Programming Language:** Python 3.x
- **IDE:** Jupyter Notebook, PyCharm, or VS Code
- **NLP Libraries:** NLTK
- **ML Libraries:** scikit-learn, pandas, NumPy, Pickle, Streamlit, Spotify

Chapter – 4

System Designing

4.1 Data Flow Diagram for a music recommendation system.

Level 0: Context Diagram

This is a high-level overview of the system showing the primary data flows between the system and external entities.

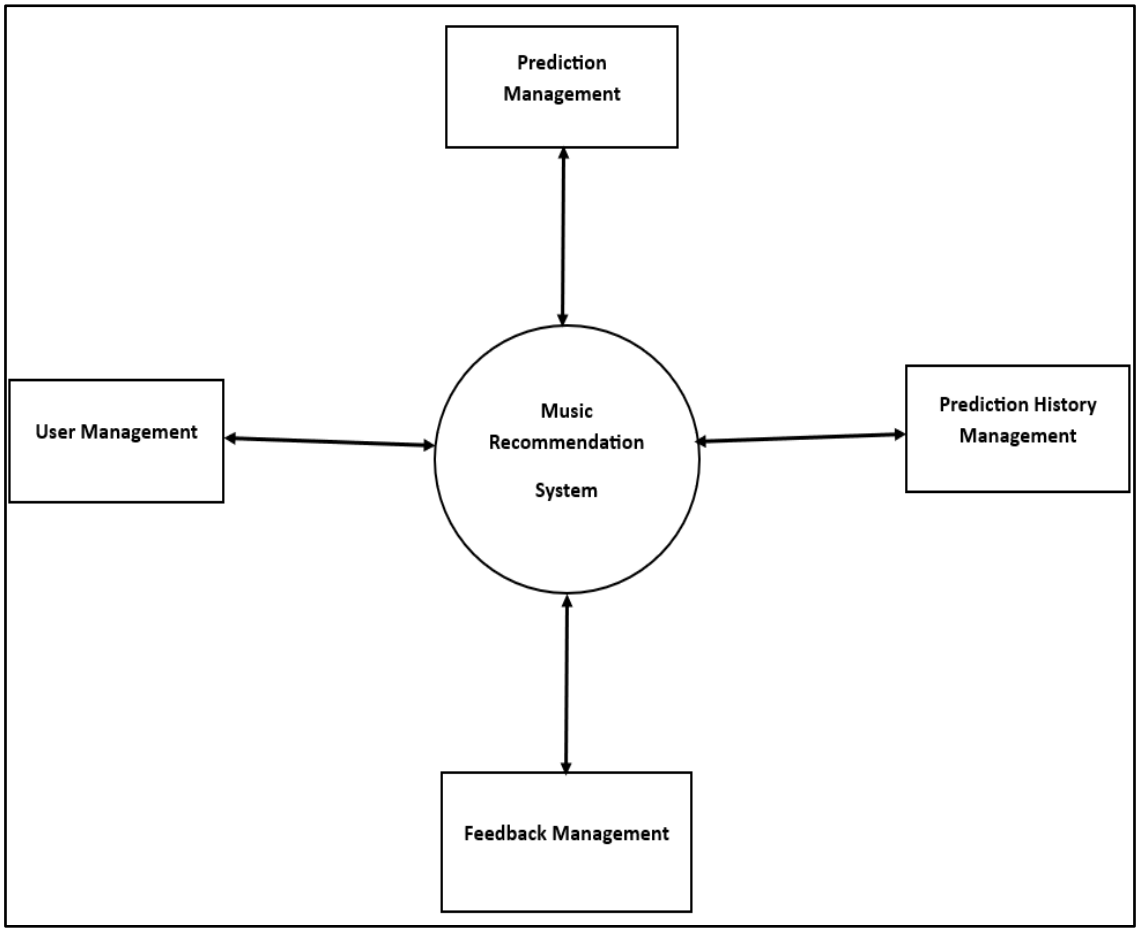


Figure 1: 0-level DFD

Level 1: Detailed DFD

This level shows more detailed data processes and data stores within the system.

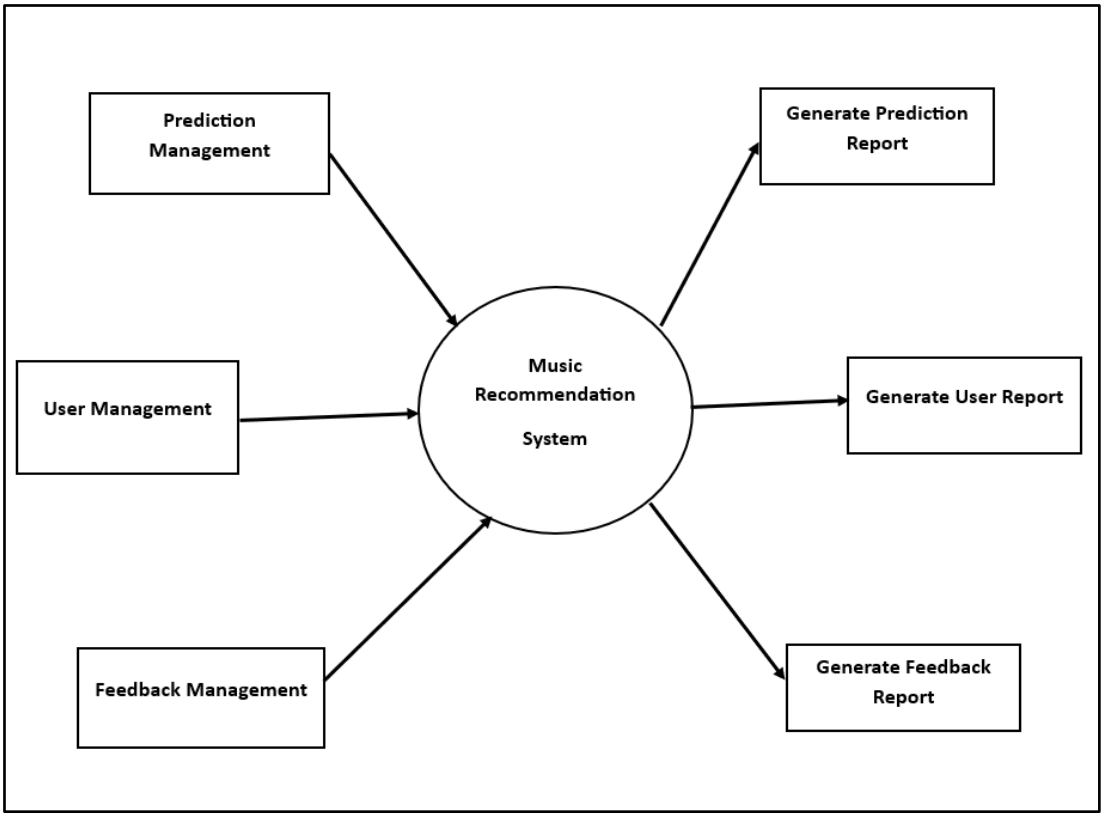


Figure 2: 1-level DFD

4.2 Flowchart of the music recommendation algorithm:

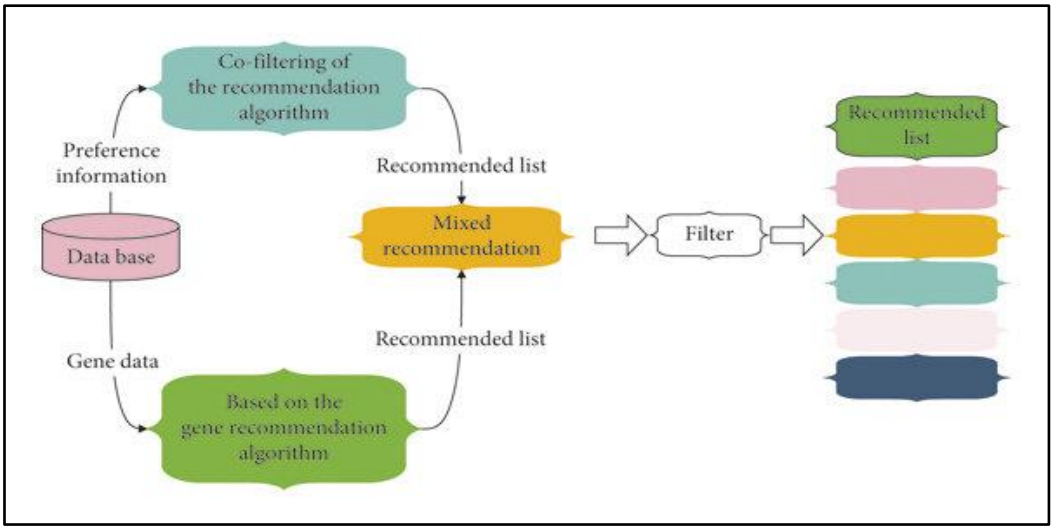


Figure 3: Flow chart

4.3 UML Diagram: Use case diagram

A Use Case Diagram in Unified Modeling Language (UML) is a visual representation that illustrates the interactions between users (actors) and a system.

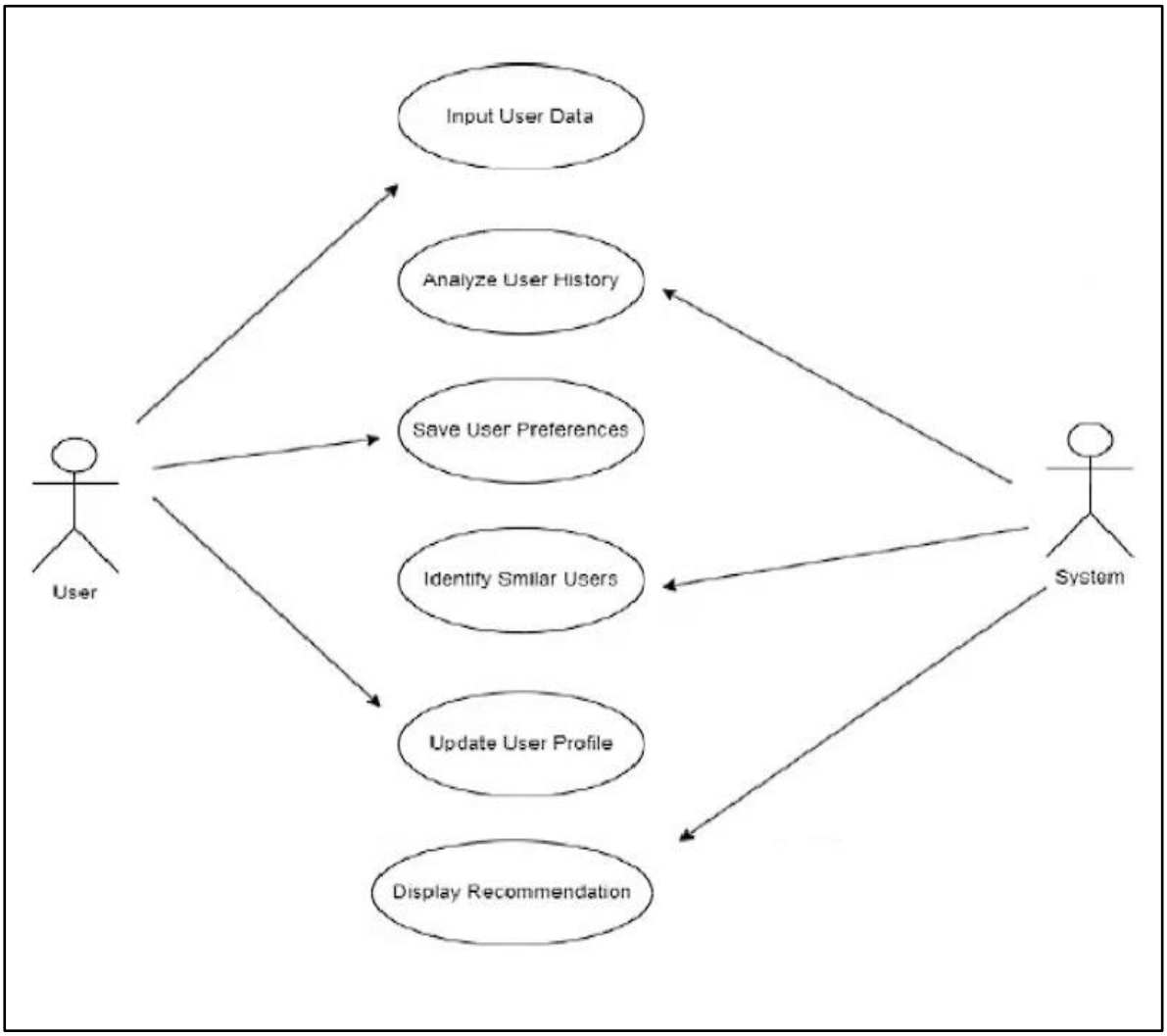
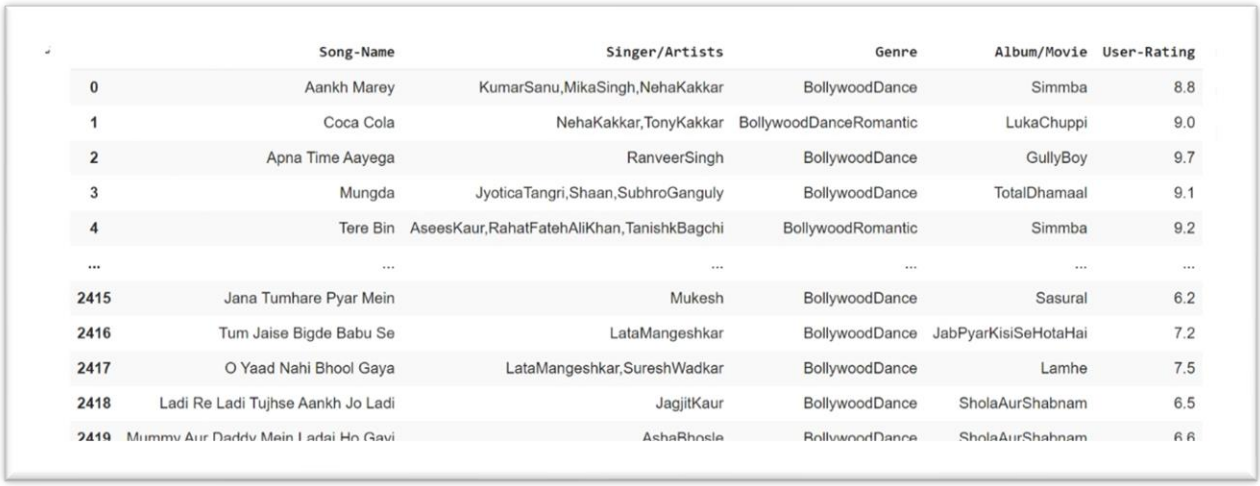


Figure 3: Use case diagram

Chapter – 5

Implementation and Results

Input Dataset:



	Song-Name	Singer/Artists	Genre	Album/Movie	User-Rating
0	Aankh Marey	KumarSanu,MikaSingh,NehaKakkar	BollywoodDance	Simmba	8.8
1	Coca Cola	NehaKakkar,TonyKakkar	BollywoodDanceRomantic	LukaChuppi	9.0
2	Apna Time Aayega	RanveerSingh	BollywoodDance	GullyBoy	9.7
3	Mungda	JyoticaTangri,Shaan,SubhroGanguly	BollywoodDance	TotalDhamaal	9.1
4	Tere Bin	AseesKaur,RahatFatehAliKhan,TanishkBagchi	BollywoodRomantic	Simmba	9.2
...	...	...	...	...	...
2415	Jana Tumhare Pyar Mein	Mukesh	BollywoodDance	Sasural	6.2
2416	Tum Jaise Bigde Babu Se	LataMangeshkar	BollywoodDance	JabPyarKisiSeHotaHai	7.2
2417	O Yaad Nahi Bhool Gaya	LataMangeshkar,SureshWadkar	BollywoodDance	Lamhe	7.5
2418	Ladi Re Ladi Tujhse Aankh Jo Ladi	JagjitKaur	BollywoodDance	SholaAurShabnam	6.5
2419	Mummy Aur Daddy Mein Ladai Ho Gayi	AshaBhosle	BollywoodDance	SholaAurShabnam	6.6

Code / Program:

Song_recomendation_system.ipynb

File Edit View Insert Runtime Tools Help Last saved at 12:18 AM

+ Code + Text

```
[ ] import numpy as np
    import pandas as pd
```

Double-click (or enter) to edit

```
[ ] df = pd.read_csv('ex.csv')
[ ] df.head()
[ ] df.isnull().sum()
[ ] df.dropna(inplace = True)
[ ] df.isnull().sum()
[ ] df.duplicated().sum()
[ ] df = df.drop_duplicates()
[ ] df.duplicated().sum()
[ ] df.shape
[ ] df.head()
[ ] df['User-Rating']
```

Song_recomendation_system.ipynb

File Edit View Insert Runtime Tools Help Last saved at 12:18 AM

+ Code + Text

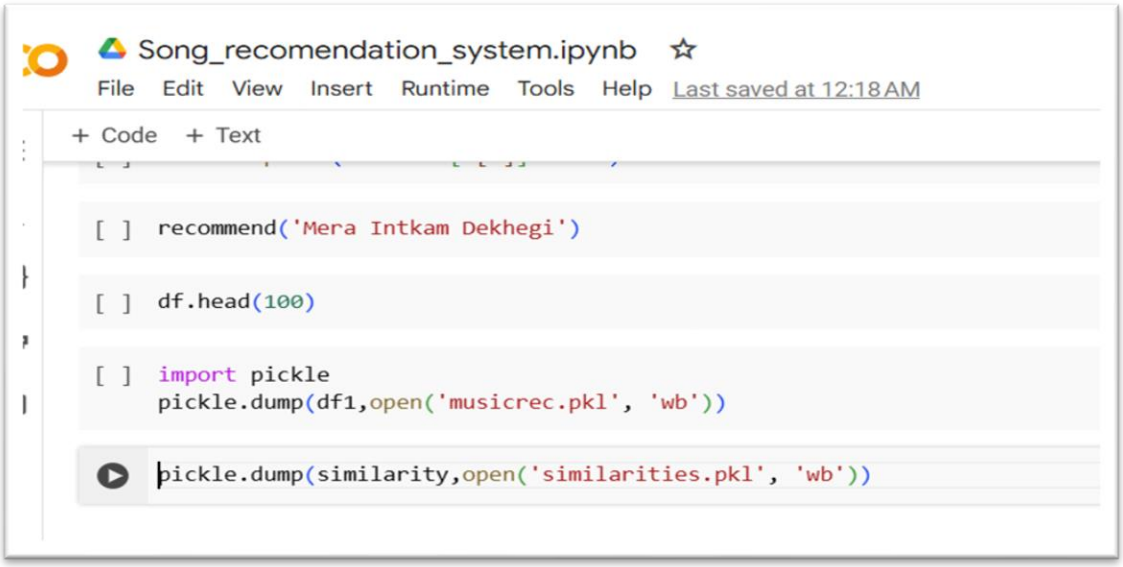
```
[ ] l=[]
    for i in df['User-Rating']:
        l.append(i[:3])

[ ] l

    df['User-Rating']=l
    df

[ ] df['Album/Movie'] = df['Album/Movie'].str.replace(' ','')
    df['Singer/Artists'] = df['Singer/Artists'].str.replace(' ','')

[ ] df

[ ] df['Singer/Artists']=df['Singer/Artists'].str.replace(',','')

[ ] df

[ ] df['tags']=df['Singer/Artists']+' '+df['Genre']+' '+df['Album/Movie']+' '+df['User-Rating']

[ ] df['tags'].head()

[ ] df1 = df[['Song-Name','tags']]
    df1

[ ] df1['tags'] = df1['tags'].apply(lambda x:x.lower())
    df1
```

Song_recomendation_system.ipynb

File Edit View Insert Runtime Tools Help Last saved at 12:18 AM

+ Code + Text

```
[ ] df1.duplicated().sum()

[ ] from sklearn.feature_extraction.text import CountVectorizer
    vectorizer = CountVectorizer(max_features=2000)

[ ] X = vectorizer.fit_transform(df1['tags']).toarray()
    feature = vectorizer.get_feature_names_out()

[ ] X.shape

[ ] vectorizer.get_feature_names_out()

[ ] from sklearn.metrics.pairwise import cosine_similarity
    similarity=cosine_similarity(X)

    sorted_list = sorted(list(enumerate(similarity[0])), reverse=True, key=lambda x: x[1])

[ ] sorted_list

[ ] df1.rename(columns={'Song-Name':'title'},inplace=True)

[ ] def recommend(music):
        music_index = df1[df1['title'] == music].index[0]
        distances = similarity[music_index]
        music_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[1:6]
        for i in music_list:
            print(df1.iloc[i[0]].title)

[ ] recommend('Mera Intkam Dekhegi')
```



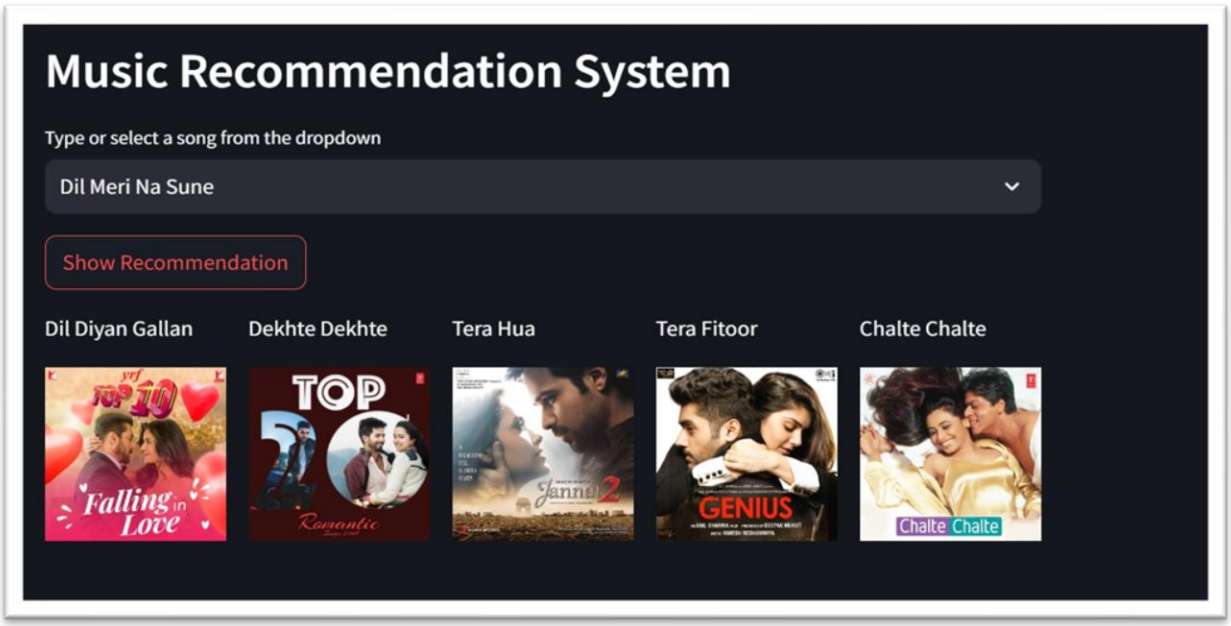
Creating User Interface:

```
import pickle
import streamlit as st
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials

CLIENT_ID = "b096a7de26614e89b431beb1b03a1d01"
CLIENT_SECRET = "196e909545bb45d4ac848a16f4ec3e75"

client_credentials_manager = SpotifyClientCredentials(client_id=CLIENT_ID, client_secret=CLIENT_SECRET)
sp = spotipy.Spotify(client_credentials_manager=client_credentials_manager)

def get_song_album_cover_url(song_name):  1 usage
    search_query = f"track:{song_name}"
    try:
        results = sp.search(q=search_query, type="track", limit=1)
        if results and results["tracks"]["items"]:
            track = results["tracks"]["items"][0]
            album_cover_url = track["album"]["images"][0]["url"]
            return album_cover_url
        else:
            return "https://i.postimg.cc/0QNxYz4V/social.png"
    except Exception as e:
        print(f"Error fetching album cover for {song_name} : {e}")
        return "https://i.postimg.cc/0QNxYz4V/social.png"
```

```
def recommend(song):  1 usage
    index = music[music['title'] == song].index[0]
    distances = sorted(list(enumerate(similarity[index])), reverse=True, key=lambda x: x[1])
    recommended_music_names = []
    recommended_music_posters = []
    for i in distances[1:6]:
        song_title = music.iloc[i[0]].title
        print(f"Fetching data for {song_title}")
        recommended_music_names.append(song_title)
        recommended_music_posters.append(get_song_album_cover_url(song_title))
    return recommended_music_names, recommended_music_posters

st.header('Music Recommendation System')

music = pickle.load(open('music.pkl', 'rb'))
similarity = pickle.load(open('similarity.pkl', 'rb'))


music_list = music['title'].values
selected_song = st.selectbox("Type or select a song from the dropdown", music_list)
```

```
if st.button('Show Recommendation'):
    recommended_music_names, recommended_music_posters = recommend(selected_song)

    cols = st.columns(5)
    for idx, col in enumerate(cols):
        with col:
            st.text(recommended_music_names[idx])
            st.image(recommended_music_posters[idx])
```

Output:



Chapter – 6

Testing

This section will include the following:

- **Unit Testing:** Verification of individual components such as the recommendation engine, data preprocessing modules, and user interface elements to ensure each unit performs as expected.

- **Integration Testing:** Testing the interaction between different modules, including database connectivity, API responses, and the web interface's integration with the recommendation engine.

- **System Testing:** Evaluating the entire system to ensure it meets all specified requirements and operates seamlessly under various conditions.

- **Performance Testing:** Assessing the system's ability to handle large-scale data and concurrent user access without significant latency or errors.

- **User Acceptance Testing (UAT):** Gathering feedback from end-users to validate the system's usability, relevance, and overall performance.

Chapter – 7

DISCUSSION AND CONCLUSION

The development of a personalized music recommendation system demonstrates the transformative potential of artificial intelligence and machine learning in delivering tailored user experiences. This project integrates collaborative filtering, content-based filtering, and deep learning models to analyze user preferences, listening habits, and song attributes. The result is a highly accurate and dynamic system capable of suggesting songs that align with user tastes while introducing them to new and diverse music.

The system not only benefits individual users but also offers significant advantages to music streaming platforms. By enhancing user engagement, retention, and satisfaction, the recommendation system contributes to the overall growth of these platforms, fostering a deeper connection between users and the music they love. Furthermore, the focus on scalability and real-time processing ensures the system's ability to handle large datasets and deliver recommendations with minimal latency, even in a high-demand environment.

The user-centric design of the system, combined with its technical sophistication, reflects a commitment to improving how users discover and enjoy music. By continuously incorporating feedback and leveraging advancements in machine learning and data analytics, the system can evolve to meet changing user needs, provide a richer music discovery experience, and remain competitive in a rapidly advancing digital landscape.

In summary, this project not only achieves its objective of delivering personalized music recommendations but also serves as a foundation for future innovations in the field of intelligent recommendation systems. It underscores the importance of combining technical excellence with a focus on user satisfaction, laying the groundwork for a system that is as impactful as it is enjoyable.