PlayNext
Project Proposal and Requirements

CS 157C - Team 2

Rafael Auerswald, Prabhat Edupuganti, Jonathan Stewart Thomas

Project Description

Goal

The goal of our project is to develop a sophisticated song and playlist recommendation engine that leverages machine learning algorithms to provide personalized recommendations to users based on their liked categories, mood or emotional state. By analyzing songs and user's emotions, we aim to capture the semantic and emotional content of each song, enabling us to identify similar songs and curate playlists that resonate with the user's current mood. Ultimately, our objective is to enhance the music listening experience by offering tailored recommendations that align with the user's emotional context.

Motivation

The motivation behind this project stems from the increasing demand for personalized content recommendation systems in the digital entertainment industry. Music streaming platforms have revolutionized the way people discover and consume music, but there remains a gap in delivering recommendations that reflect the user's emotional state or mood. By incorporating natural language processing techniques to analyze songs, we can bridge this gap and offer recommendations that resonate with the user on a deeper emotional level. This not only enriches the user experience but also fosters greater engagement and loyalty to the platform.

Stakeholders

- Advisor: Dr. Ching-seh (Mike) Wu
- Backend Developer
- Frontend Developer
- Customers

Application Domain

Our project operates at the dynamic intersection of machine learning, music, and emotions, striving to pioneer innovative solutions within this unique domain.

Benefit to Our Users

1. **Personalized Recommendations**: Users will receive tailored song and playlist recommendations based on their current mood or emotional state, enhancing their listening experience and fostering a deeper connection with the music.

2. **Discoverability**: Our recommendation engine helps users discover new music that aligns with their tastes and preferences, introducing them to artists and genres they may not have encountered otherwise.

3. **Enhanced Mood Regulation**: By selecting a mood descriptor, users can actively regulate their emotions through music, whether they seek motivation, relaxation, or inspiration.

4. **Time Efficiency**: Instead of spending time searching for the perfect song or playlist, users can rely on our recommendation engine to quickly find music that suits their mood, saving time and effort.

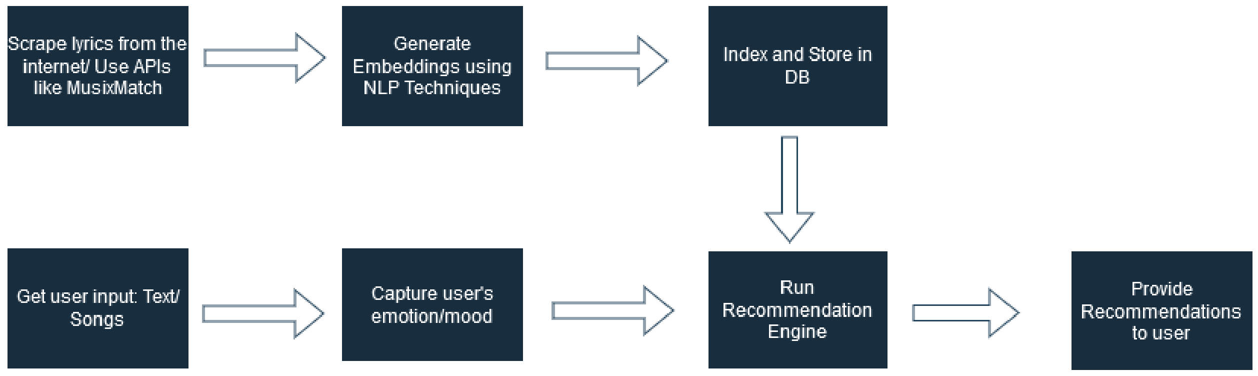
5. **Continuous Improvement**: As users interact with the recommendation engine, providing feedback and preferences, the system can continually refine its recommendations, ensuring relevance and accuracy over time.

Development Environment

MongoDB

MongoDB is a document based NoSQL database. It utilizes JSON-like documents to store data. We will use the most recent MongoDB 7.0.

Structure of Our System


Scrape lyrics from the internet/ Use APIs like MusixMatch
Generate Embeddings using NLP Techniques
Index and Store in DB
Get user input: Text/ Songs
Capture user's emotion/mood
Run Recommendation Engine
Provide Recommendations to user

Hardware and Software Used

- Frontend: Gradio/Dash
- Backend: Flask, FastAPI, Asyncio, PyMongo
- Machine Learning and Deployment: HuggingFace, SentenceTransformers, Pandas, Sklearn, Numpy, Pytorch
- Communication/Collaboration: Github, Discord
- OS: MacOS 13.6.4 (22G513), 14.3.1 (23D60) and Windows 10 and 11

- Computers: Base model M1 Pro Macbook Pro 16in, base model M2 MacBook Air 13in, Custom PC with 5600x, RTX 4070, 32GB 3200 RAM.

Application Languages

- Python
- PySpark
- Json

Functional Requirements

Features for Users

Creating Accounts and Logging In

Users can create an account and log in

Changing Email and Password

Users can change their email and password

Favorite Songs from the database

Users can favorite songs they like from the database

Add or Remove Categories of songs from likes/dislikes

Users can add or remove categories of songs they like or dislike

Pick Types of Emotions

Users can pick types of emotions, such as happy, sad, energized, etc, to generate a list of songs that fit that emotion.

Features for Admins

Adding songs to the database

Admins can add new songs to the database

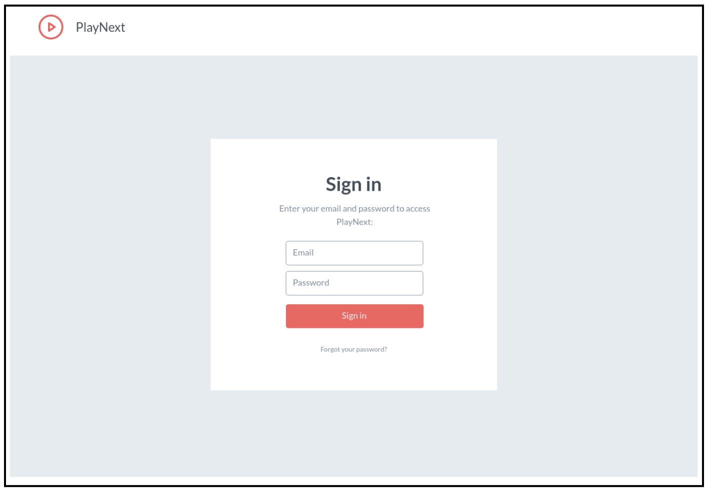
Adding Categories

Admins can add more categories for types of music

Non-Functional Issues

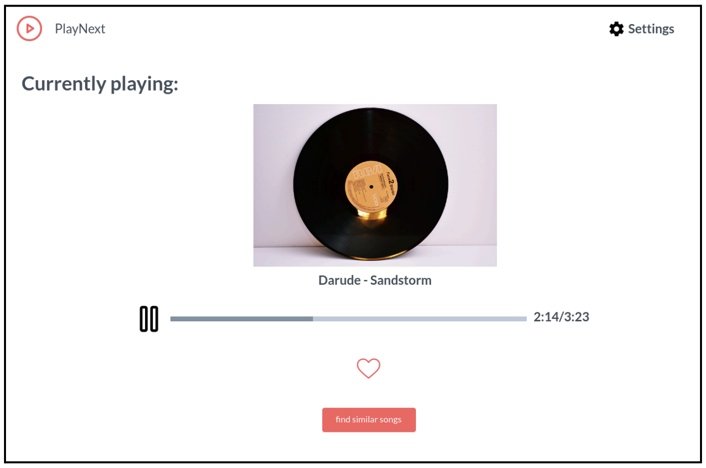
The GUI

We aim for a simple user-centered interface with few distractions which should be usable without requiring a tutorial. To illustrate how we imagine that, we created some prototype screenshots with Marvelapp.

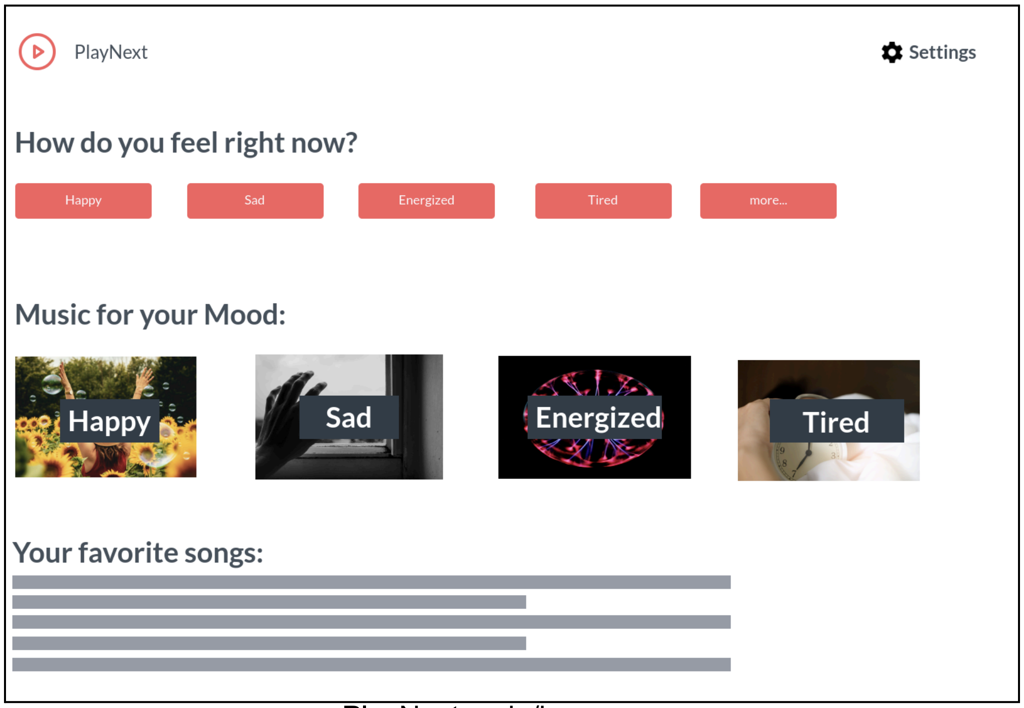


Enter your email and password to access PlayNext:

Sign-In screen



Currently playing screen



PlayNext main/home screen

Security

- **Use of passwords**: Users have to sign in with passwords that have to match the following requirements: At least 8 characters, at least 1 number, at least one special character.

- **Hashing of passwords**: The passwords stored on the server should be hashed and not stored in clear text.

- **Use of HTTPS**: In our production version, for security purposes, traffic should be encrypted.

Access Control

- **User/Administrator rights**: Users should be able to use the service as described in the functional requirements section of this document. Admins should have additional rights that users aren’t entitled to such as adding songs to the database or adding categories.

- **Unique account**: Each user needs their own account. There are no shared accounts as this application is highly personalized. Users can authenticate with the combination of their email and password.