Project Proposal

By David Fu(jiahaof4), Zhuoang Tao(zhuoang2), Xinming Zhai(xinming7), Aoyang Li(aoyangl2)

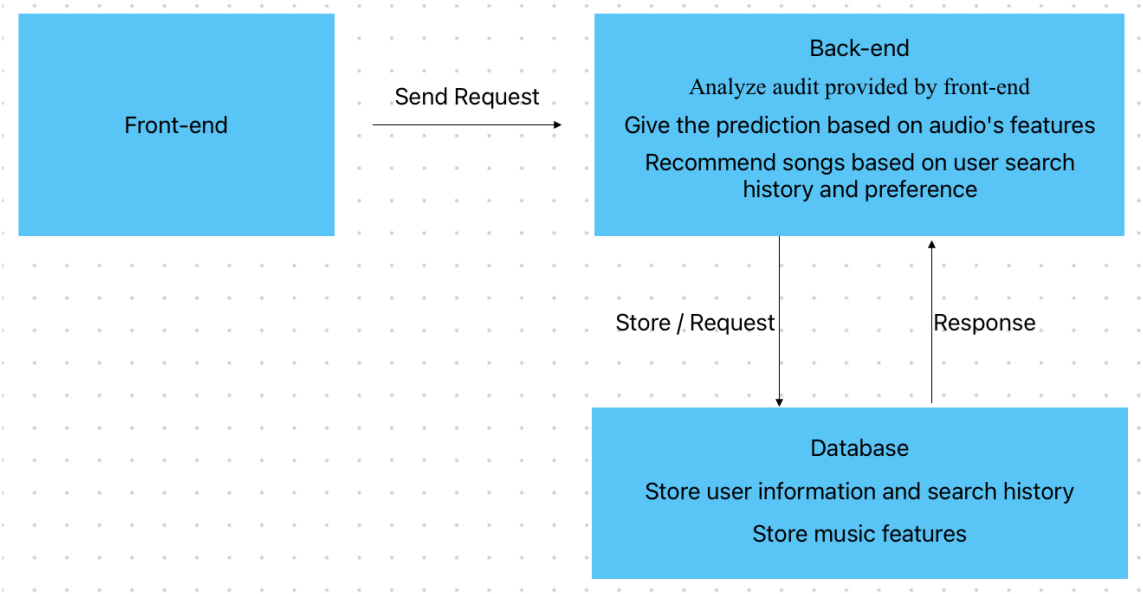
1. **Pitch**

Are you struggling looking through the music libraries? Are you annoyed finding the perfect song for the moment? Our music recommendation system is here to help you and perfect your music experience. Our system will offer a personalized set of music recommendations to match your mood and preferences under advanced audio analyzation and accurate classification. Moreover, our system can automatically classify music audios user uploads into different genres based on their audio features.

2. **Functionality**
    a. Recognize the genre of a specific song for users by uploading
    b. Personalize a playlist similar with users' preference
    c. Recommend more songs given music from users
    d. Create account and login
    e. Show personal search history
    f. Show users' music preferences
3. **Components**

We will be using a front-end back-end structure, a commonly used architecture in modern development. The front-end will be responsible for user interface and showing results retrieved from the back-end, while the back-end would perform necessary calculations and operations to acquire data from the database and return required information back to the front-end upon receiving HTTP requests. A brief summary of the

structure is shown below:



- Backend:
  - Language:
    - Python - backend and machine learning model
    - MySQL - database management
  - Libraries:
    - Librosa(Python)
      Analyze audio features of music
    - scikit-learn(Python)
      Build and train machine learning models
    - NumPy(Python)
      Help do mathematical/matrices calculations
    - PyTorch(Python)
      Build and train advanced machine learning/deep learning models
    - Pandas(Python)
      Dataset reading and analysis
  - Functionality
    - Analyze audio provided by front-end
    - Get user login information
    - Save user register information
    - Read and save user history and preferences
    - Give a list of recommended songs with respect to provided audio
  - Testing:
    - Unit tests on each functionality
    - Evaluating ML model with validation/test datasets

- Interaction

  We will use Python to backend will interact with frontend with HTTP request and respond

- Frontend:
  - Language
    - HTML

      Build the main components of web pages
    - CSS

      Webpage decoration
    - JavaScript

      Interact with backend and perform necessary functions
    - React

      Advanced web page building tool
  - Functionality:
    - Visualize users' music preferences
    - Present users' profile and searching history
    - Present a login page
    - Create a searching and uploading page
  - Testing:
    - Test each functionality after the website is completely built with test libraries
    - Open Beta version of the website to users for extra testing
  - Interaction:

    We will use JavaScript to interact with backend API using HTTP requests

4. Schedule

Week # (Date)	**Tasks**
Week 1 (9/18-9/24)	1. Create database and install necessary tools and libraries 2. Data collection and cleaning
Week 2 (9/25-10/1)	1. Design the layout of the webpage 2. Build up the main components of the front-end webpage

	3. Database and environment testing
Week 3 (10/2-10/8)	1. Build the other extra components of the webpage 2. Search and select appropriate machine learning model
Week 4 (10/9-10/15)	1. Implement login and showing history and preferences for front-end 2. Train machine learning model on dataset
Week 5 (10/16-10/22)	1. Implement uploading functionality of the front-end 2. Evaluate trained model with validation set
Week 6 (10/23-10/29)	1. Build reading user information for back-end 2. Employ model on the back-end
Week 7 (10/30-11/5)	1. Build recommendation system for back-end 2. Build analysis system for back-end
Week 8 (11/6-11/12)	1. Connect front-end and back-end 2. Connect with music streaming platform
Week 9 (11/13-11/19)	1. Unit tests for back-end functionalities 2. Library test for front-end functionalities 3. Platform API test
Week 10 (11/27-12/3)	1. Release Beta version of the website for open testing

	2. Clean up and leave buffer for any risks

5. Potential Risks

a. Algorithm Performance

Achieving accurate recommendations is challenging because of complexity of musical taste and the randomness of the music users upload. Moreover, machine learning models may overfit to a user's short term preference, which makes recommendation not reflective to the user's overall preference.

Steps if encountered: reporting problem to the group → discuss the origin of the problem and try to solve the problem by other members → ask mentor for help if needed
Schedule impact: the scheduled progress might slow down significantly
Plan to address schedule impact: use buffer time wisely to keep up with planned schedule and swap the task when other members trying to fix the problem

b. Cold start problem

New users may present challenges in making accurate recommendations because of lack of historical search records.

Steps if encountered: asking users provide some preference information when creating accounts; or providing recommendations based on other users with similar search histories
Schedule impact: the finish data may be postpone since we need to add features to our website to solve the problem
Plan to address schedule impact: we can try to solve the problem after we built the main skeleton of the project since this problem is just about the details and not urgent

c. Building database on cloud:

We lack experience building such types of applications. It will be a little difficult for us to implement the whole system for the first time.

Steps if encountered: learn the related knowledge together → try to follow online tutorials to finish the task →ask mentor for help
Schedule impact: compress our time on constructing other contents
Plan to address schedule impact: we might have to omit some details in improving appearance of website for instance to save time

d. Some of the functionalities might be unsatisfying or incomplete

Given time constraints and the capability of members, some functionalities might be implemented without much details, or even failed to implement, though we will definitely finish most of the core functions first.

Steps if encountered: search for online resources, such as pseudocode → ask mentor for help

Schedule impact: we may have less time to do other steps and unable to catch the schedule

Plan to address schedule impact: try to solve the problem as quickly as possible and just let one person stay on that problem. Others should still follow the original schedule. Options like adjusting the functionality should also be considered if the problem persists.

6. Teamwork

We will implement a thorough onboarding and orientation process. Each team member will have an opportunity to introduce themselves, their skills and past projects. Understanding and appreciating the diverse expertise within the team fosters mutual respect and helps in assigning tasks that align with individual strengths. We will have casual social gatherings to create an environment where team members feel comfortable and connect with each other, which helps us to minimize conflicts and promote collaboration.

To divide work fairly, we will follow the initial skills of each member to assign the work of this project, such as user experience design, machine learning and data analysis. Moreover, we will consider the interests of each team member to set the related aspects of this project as work.

7. Continuous Integration

- Testing Library:

  We will use React Testing Library for the frontend part and MySQL Test Framework for the backend part.
- Style guide:

  We will be following PEP 8 Python style guide, using proper function names and variable names. We also highly recommend writing comments when appropriate for other members to read. Pull requests with unreadable code segments will be requested revision by the reviewer.
- Test coverage:

We will evaluate test coverage by line coverage metric: computing this metric by dividing the number of lines we tested by the total number of lines in the project.

- Pull request workflow:

  After one has finished his part and tested the code, push to the github in his own branch. Other team members help to solve merge conflicts and provide reviews. After peer review and no conflicts are found, we are able to merge the branch onto the main branch. Direct modification on the main branch is strictly forbidden.