# YouTune

**Group Members: Sherwin Fernandes, Theo Bongolan, Kyle Torres, Ben Lopez**

# 1. Introduction and Overview

### 1.1 Introduction

This software will be a way of sharing music. The system should allow the user to make requests of other users to listen to a song and then will be able to give their own feedback to the requester user

### 1.2 Software Overview

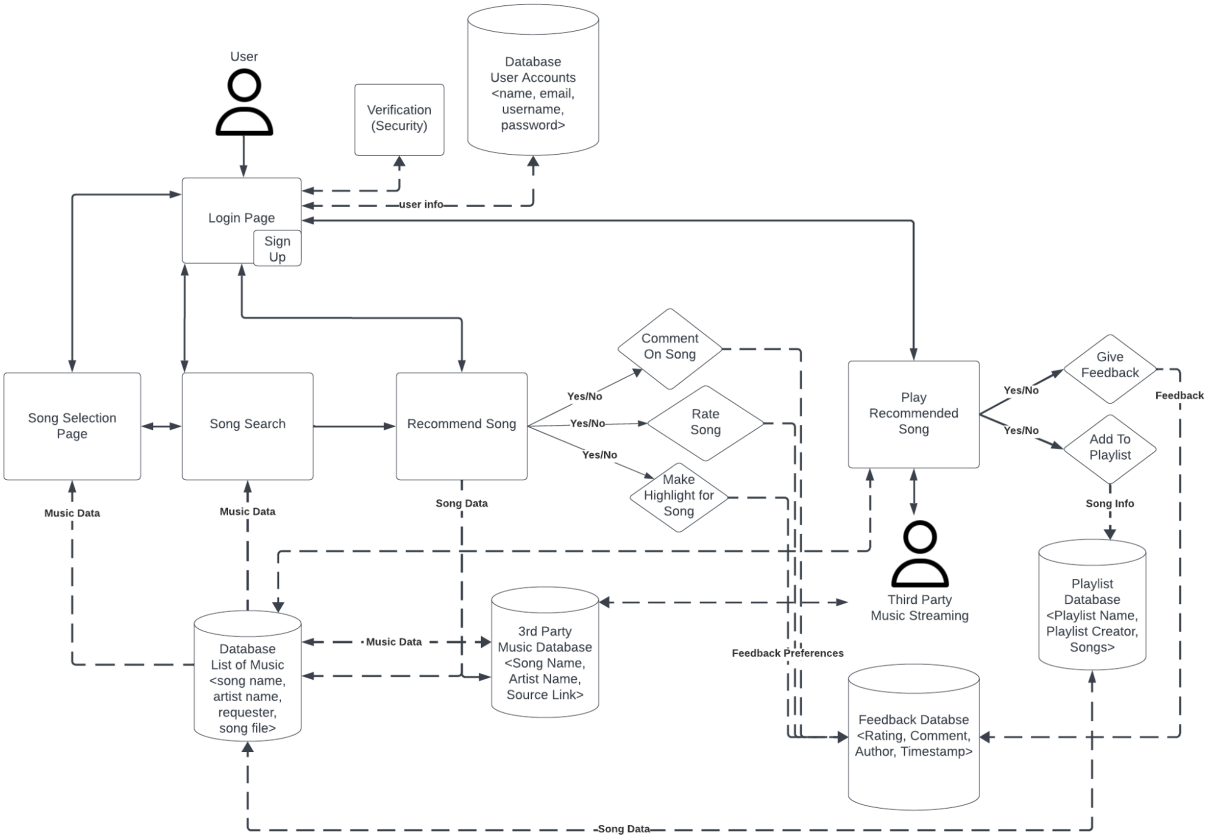
The software will mainly allow users to share songs from multiple music platforms and allow the listeners to browse requested songs made by other users. Every user will be a listener and requester. With every song, listeners will be able to give a rating and feedback.

### 1.3 Document Overview

Section 2: Software Architecture Diagram
Section 3: UML Class Diagram
Section 4: Development Timeline

# 2. Software Architecture Diagram

## 2.1 Diagram



## 2.2 Component Description

User

The user is an entity that will be able to interact with the Login/Signup Page

Login/Signup Page

Gives the user access to the software and serves as a gateway to provide the user with the services available, using the verification component and user account database for added security and

User Accounts Database

Stores sensitive account data for users (name, email, username, password)

Song Selection Page

Allows users to browse through the songs available in the database

Song Search

Allows users to search for specific songs by song name, artist, or requests from a user that will look through the Music Database

Music Database

Stores data about the song name, artist, user that made the rating request, and the song file

3rd Party Music Database

Stores information on songs that are connected to external streaming platforms that'll store data on the song name and artist, with a link to get the song file

Recommend Song

Allows users to make requests to share songs by either linking a song or uploading their own (feature for future release) and setting preferences for what feedback they will receive (Allow commenting, rating, and/or highlighting timestamp)

Play Recommended Song

Pulls Song Data from the Music Database if available or else pulls from the 3rd Party Music Database and streams it for the user. Allows users to give feedback and add to a playlist

Playlist Database

Stores data on playlists created by users (playlist name, playlist creator, songs list)

Third-Party Music Streaming Service

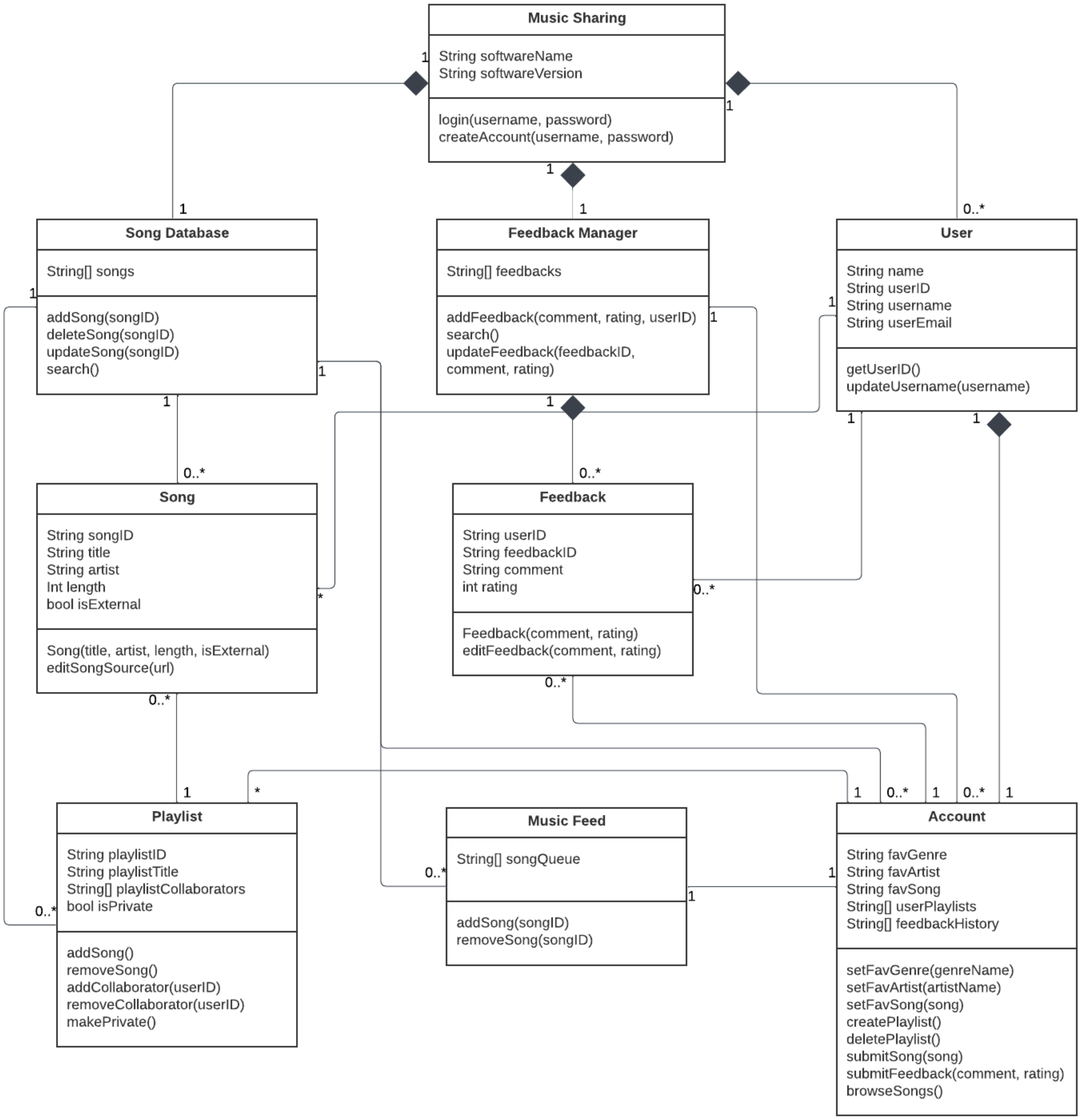
Pulls song data from a 3rd party streaming service if necessary to stream a song

Feedback Database

Stores data on user feedback whenever users submit feedback on a song recommendation

# 3. UML Class Diagram

## 3.1 Diagram



### 3.2 Class Description

Music Sharing: main class for the software system

Attributes:

- String softwareName: name of the software
- String softwareVersion: tracks the current version of the system

Operations:

- Login(username, password): allows users to log into the system
- createAccount(name, username, email, password): takes user input to create an account for the user

Feedback Manager: Class for storing, tracking, and managing feedback made within the system

Attributes:

- String[] feedbacks: stores an array of all user feedbacks

Operations:

- addFeedback(comment, rating, userID): add a feedback to the database
- search(): look through the feedback database
- updateFeedback(feedbackID, comment, rating): update/edit a user's feedback

Song Database: Class for storing, tracking, and managing songs added within the system

Attributes:

- String[] songs: stores an array of all songs within the system

Operations:

- addSong(songID): adds song to the database
- deleteSong(songID): deletes song from the database
- updateSong(songID): updates information about the song within the database
- search(): allows searching through the database

User: Class for each user account

Attributes:

- String name: store user’s name on account
- String userID: stores generated userID for account
- String username: stores account username
- String userEmail: stores user’s email on account

Operations:

- getUserID(): returns the userID for an account
- updateUsername(username): updates the username

Song: Class for songs submitted by users

Attributes:

- String songID: generated id connected to a song
- String title: the name of a song
- String artist: name of whoever created the song
- int length: length of the song
- bool isExternal: true if the song is sourced from a 3rd party streaming service

Operations:

- Song(title, artist, length, isExternal): sets all the variables when creating a new song object
- editSongSource(url): allows source for externally linked songs to have the url updated if needed

Playlist: Class for letting users store a list of songs

Attributes:

- String[] playlistID: Playlist’s unique identifier
- String playlistTitle: name of the playlist
- String playlistCollaborators: list of collaborators for the playlist
- bool isPrivate: true if the playlist is not publicly accessible

Operations:

- addSong(): adds a song to the playlist
- removeSong(): removes a song from the playlist
- addCollaborator(userID): Gives user permission to add to this playlist
- removeCollaborator(userID ): Revokes user permission to add to this playlist
- toggleVisibility(): switches the playlist privacy between public/private

Music Feed:

Attributes:

- String[] songQueue: list of songIDs in a given queue

Operations:

- addSong(songID): adds a specified song to the song queue
- removeSong(songID): removes a specified song from the queue
- togglePlay(): allows user to play/pause the queue

Account:

Attributes:

- String favGenre: A string storing a listeners favorite Genre of song
- String favArtist: A string storing a listeners favorite musical Artist
- String favSong: A string storing a listeners favorite Song
- String[] userPlaylists: array of all playlistID's created on the account
- String[] feedbackHistory: array of all feedback submissions made on the account

Operations:

- setFavGenre(genreName): sets or updates a Listener's favGenre string
- setFavArtist(artistName): sets or updates a Listener's favArtist string
- setFavSong(song): sets or updates a Listener's favSong string
- createPlaylist(): allows user to create a new playlist on their account
- deletePlaylist(): allows users to delete their playlists on their account
- submitSong(song): allows users to submit a song for others to rate
- submitFeedback(comment, rating): allows user to submit feedback on a song
- browseSongs(): allows user to look through available song submissions

Feedback:

Attributes:

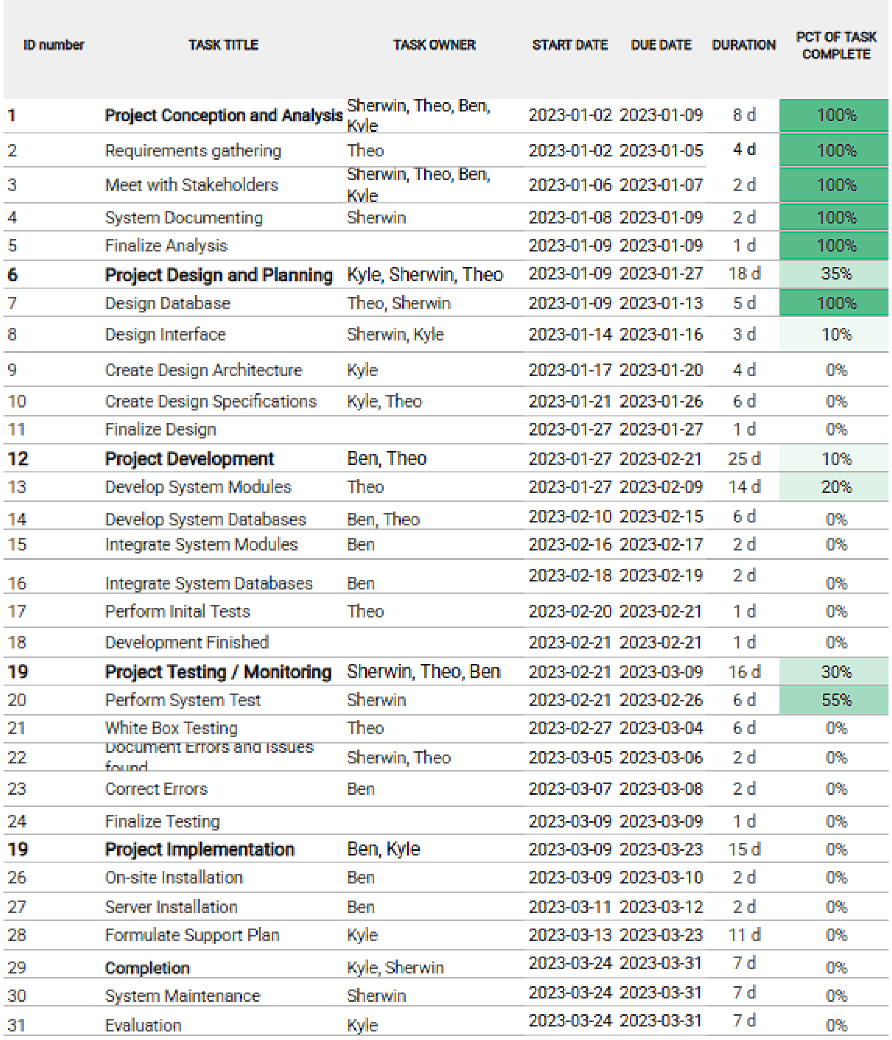
- String userID: user that created the feedback
- String feedbackID: generated ID for a user's feedback
- String comment: user's comment for a song
- int rating: user's rating for a song

Operations:

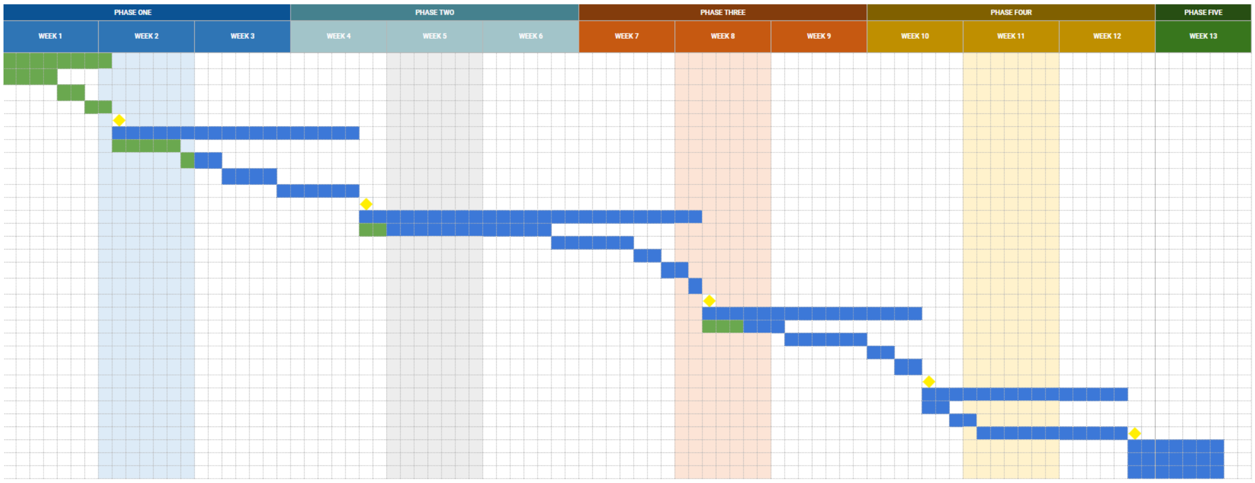
- Feedback(userID, comment, rating): constructor to store the comment/rating
- editFeedback(comment, rating): allows user to edit their feedback

# 4. Development Timeline

## 4.1 Timeline: View the full timeline here



| ID number | TASK TITLE | TASK OWNER | START DATE | DUE DATE | DURATION | PCT OF TASK COMPLETE |
|---|---|---|---|---|---|---|
| **1** | **Project Conception and Analysis** | Sherwin, Theo, Ben, Kyle | 2023-01-02 | 2023-01-09 | 8 d | 100% |
| 2 | Requirements gathering | Theo | 2023-01-02 | 2023-01-05 | 4 d | 100% |
| 3 | Meet with Stakeholders | Sherwin, Theo, Ben, Kyle | 2023-01-06 | 2023-01-07 | 2 d | 100% |
| 4 | System Documenting | Sherwin | 2023-01-08 | 2023-01-09 | 2 d | 100% |
| 5 | Finalize Analysis | | 2023-01-09 | 2023-01-09 | 1 d | 100% |
| **6** | **Project Design and Planning** | Kyle, Sherwin, Theo | 2023-01-09 | 2023-01-27 | 18 d | 35% |
| 7 | Design Database | Theo, Sherwin | 2023-01-09 | 2023-01-13 | 5 d | 100% |
| 8 | Design Interface | Sherwin, Kyle | 2023-01-14 | 2023-01-16 | 3 d | 10% |
| 9 | Create Design Architecture | Kyle | 2023-01-17 | 2023-01-20 | 4 d | 0% |
| 10 | Create Design Specifications | Kyle, Theo | 2023-01-21 | 2023-01-26 | 6 d | 0% |
| 11 | Finalize Design | | 2023-01-27 | 2023-01-27 | 1 d | 0% |
| **12** | **Project Development** | Ben, Theo | 2023-01-27 | 2023-02-21 | 25 d | 10% |
| 13 | Develop System Modules | Theo | 2023-01-27 | 2023-02-09 | 14 d | 20% |
| 14 | Develop System Databases | Ben, Theo | 2023-02-10 | 2023-02-15 | 6 d | 0% |
| 15 | Integrate System Modules | Ben | 2023-02-16 | 2023-02-17 | 2 d | 0% |
| 16 | Integrate System Databases | Ben | 2023-02-18 | 2023-02-19 | 2 d | 0% |
| 17 | Perform Inital Tests | Theo | 2023-02-20 | 2023-02-21 | 1 d | 0% |
| 18 | Development Finished | | 2023-02-21 | 2023-02-21 | 1 d | 0% |
| **19** | **Project Testing / Monitoring** | Sherwin, Theo, Ben | 2023-02-21 | 2023-03-09 | 16 d | 30% |
| 20 | Perform System Test | Sherwin | 2023-02-21 | 2023-02-26 | 6 d | 55% |
| 21 | White Box Testing | Theo | 2023-02-27 | 2023-03-04 | 6 d | 0% |
| 22 | Document Errors and Issues found | Sherwin, Theo | 2023-03-05 | 2023-03-06 | 2 d | 0% |
| 23 | Correct Errors | Ben | 2023-03-07 | 2023-03-08 | 2 d | 0% |
| 24 | Finalize Testing | | 2023-03-09 | 2023-03-09 | 1 d | 0% |
| **19** | **Project Implementation** | Ben, Kyle | 2023-03-09 | 2023-03-23 | 15 d | 0% |
| 26 | On-site Installation | Ben | 2023-03-09 | 2023-03-10 | 2 d | 0% |
| 27 | Server Installation | Ben | 2023-03-11 | 2023-03-12 | 2 d | 0% |
| 28 | Formulate Support Plan | Kyle | 2023-03-13 | 2023-03-23 | 11 d | 0% |
| 29 | **Completion** | Kyle, Sherwin | 2023-03-24 | 2023-03-31 | 7 d | 0% |
| 30 | System Maintenance | Sherwin | 2023-03-24 | 2023-03-31 | 7 d | 0% |
| 31 | Evaluation | Kyle | 2023-03-24 | 2023-03-31 | 7 d | 0% |


PHASE ONE
PHASE TWO
PHASE THREE
PHASE FOUR
PHASE FIVE
WEEK 1
WEEK 2
WEEK 3
WEEK 4
WEEK 5
WEEK 6
WEEK 7
WEEK 8
WEEK 9
WEEK 10
WEEK 11
WEEK 12
WEEK 13