**TEE_BEE_DEE by JAMEs**
P02: Makers Makin' It, Act I
Devos: Ethan Cheung (PM), Amy Shrestha, Jason Chan, Matthew Ciu
2026-01-09
TARGET SHIP DATE: *2026-01-16*

Project Description: Our project is a website that creates the soon-to-be most popular song in the world. This is done by taking the most popular songs' lyrics and randomly picking lines and putting them in an assigned template. Users can choose two songs from 6 randomly rolled top songs to create a mashup of both pieces using a pre-set algorithm. Top the leaderboard by creating the most songs!

## Program Components:

A. Flask App (Python)
   a. data.py
      - Connects to SQLite3 database and creates this table:
        - **users** (stores personal info and stats in profile)
   b. __init__.py
      - Routes:
        - **/register** adds a username and password to the **users** table. Checks if username is unique, stores username in session then redirects to **/home**
        - **/login** checks if username is in **users** table and if password matches. If so, stores username in session then redirects to **/home**
        - **/home** displays various
          - **/TSG** "top song generator" uses songs from top billboard songs and uses their lines to create the next platinum-selling song
            - Finds song lyrics using Lyrics API
          - **/Speech-Text** will be the rolling page for songs where 6 randomly generated songs will be displayed with album cover, song name, and singer(s)
        - **/leaderboard** displays user's stats and displays stats of top 10 users

        - Rankings of leaderboard is decided based on who makes the most songs
    - **/logout** takes username out of session and redirects to /login

B. Templates
- Consistent navbar on all pages that redirects to different pages with a logout button
- **/login** & **/register** (form input boxes, handles authentication)
- **/home** (user stats, buttons towards following three)

C. Database (SQLite3) (stored in data.db)
- **user_data** table stores all usernames and matching passwords plus stats of users

D. RESTful API's
- Billboard API
    - https://github.com/mhollingshead/billboard-hot-100
        - Gives music charts, artists, and songs across various genres
- Lyrics API
    - https://lyricsovh.docs.apiary.io/#
        - Gives lyrics
- Album Cover API
    - https://musicbrainz.org/doc/Cover_Art_Archive/API

E. Frontend Frameworks
- Bootstrap
    - Grid for organizing leaderboard and profile stats
    - Buttons and cards for music that the user has created
    - Form styling, for use in login/register
    - Navbar functionality
    - Badges to indicate top/trending songs (in top 10 songs)
    - Spinners for loading page
    - Accordion for showing/hiding song lyrics

F. CSS as necessary

G. Javascript Files
- tsg.js
- speechText.js

**Database Organization:**

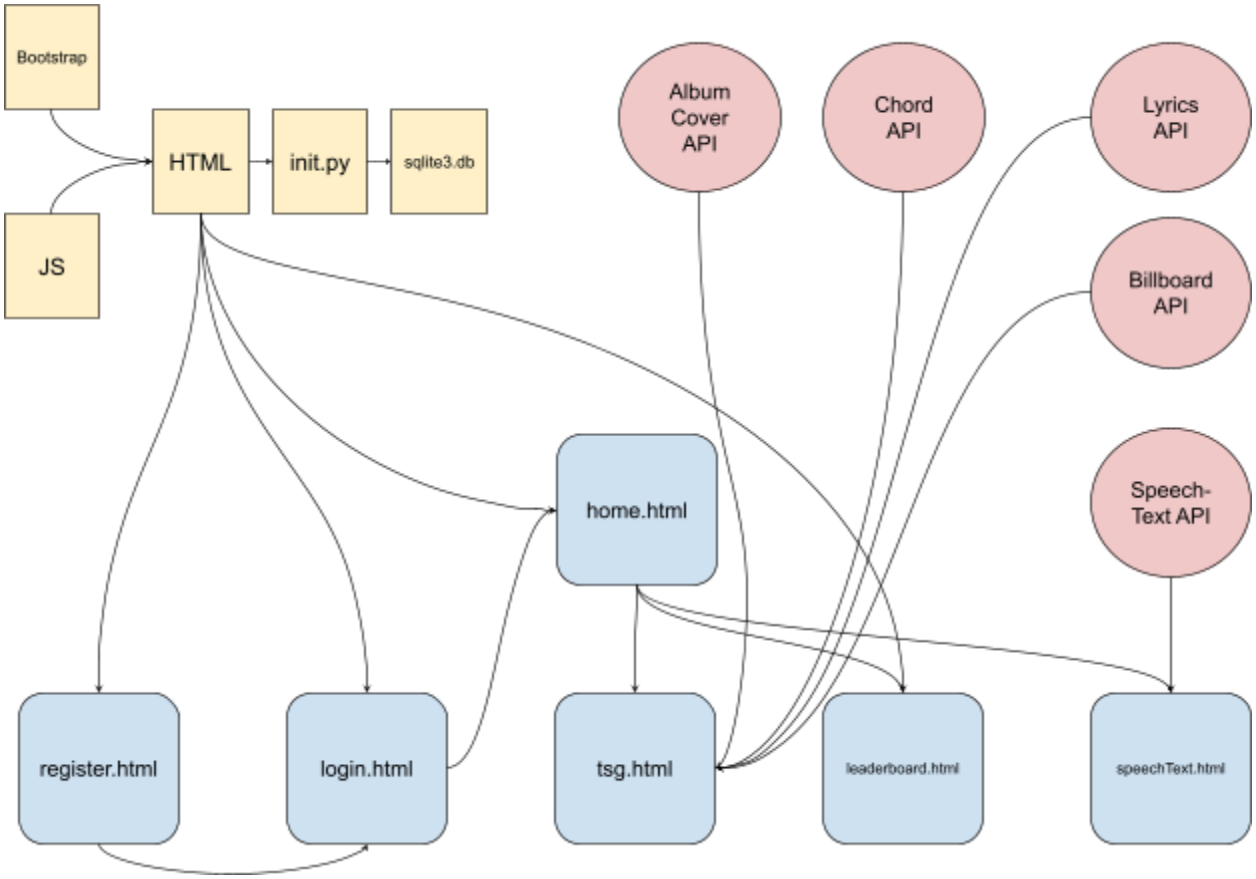
USERDATA

| | | |
|---|---|---|
| INTEGER | user_id | PK NOT NULL |
| TEXT UNIQUE | username | NOT NULL |
| TEXT | password | NOT NULL |
| TEXT | saved_songs | |
| INTEGER | total_songs | |

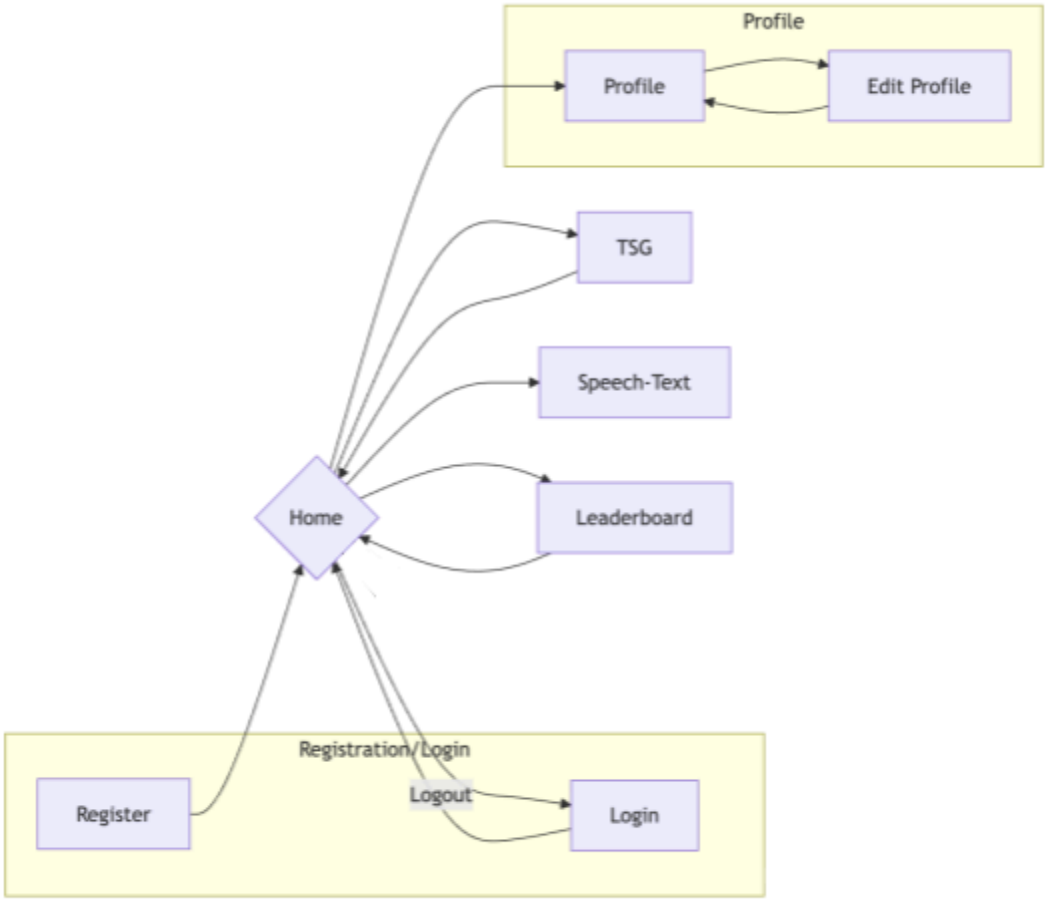
CREATESONG

| | | |
|---|---|---|
| TEXT | new_song_name | PK NOT NULL |
| TEXT | song_name1 | NOT NULL |
| TEXT | song_name2 | NOT NULL |
| TEXT | lyrics | NOT NULL |

## Component Map:



**Site Map:**




DEADLINES

- 1/14
  - [x] ~~PM stand ups~~
- 1/15
  - [ ] readme in keys dir
  - [ ] PM stand ups
- 1/16
  - [ ] Due date
  - [ ] Final repo pull
  - [ ] Design dock x1

DESIGN DOC WHO DOES WHAT

| **Task** | **Devo(s)** |
|---|---|
| Program components & explanation | Amy |
| Component map | Jason |
| DB Organization | Ethan |
| Site map | Matthew |
| API section | Amy |
| FEF section | Jason |
| Task Breakdown | Ethan |

PROJECT WHO DOES WHAT

| **Task** | **Devo(s)** |
|---|---|
| Handle middleware | Amy |
| Login page | Ethan |
| Register page | Ethan |
| Home page | Amy |
| Top Song page | Matthew + Amy |
| Speech-Text page | Jason + Ethan |
| Leaderboard page | Ethan |
| FEF and CSS implementation | All |