EmotionTune

EMOTION TUNE PROJECT REPORT

Were Vincent

19/January/2025

Introduction

The EmotionTune project aims to develop a music streaming application that caters to users' emotional preferences. It leverages a relational database to store and manage user information, music details, emotions, recommendations, playback history, and user preferences. This report delves into the project's database design, user manual, SQL queries, and potential distribution of work among team members.

Database Scheme

Tables

a. Users

Columns

- **i.** UserID (INTEGER PRIMARY KEY AUTOINCREMENT): Unique identifier for each user.
- **ii.** Username (TEXT NOT NULL UNIQUE): User's chosen username for login.
- **iii.** Email (TEXT NOT NULL UNIQUE): User's email address for registration and communication.
- **iv.** Password (TEXT NOT NULL): Encrypted user password for secure authentication.

b. Music

❖ Columns

- **i.** MusicID (INTEGER PRIMARY KEY AUTOINCREMENT): Unique identifier for each song.
- **ii.** Title (TEXT NOT NULL): Title of the song.
- **iii.** Artist (TEXT NOT NULL): Artist or band who created the song.
- **iv.** Genre (TEXT): Genre classification of the song (e.g., Pop, Rock, Jazz).
- **v.** ReleaseDate (DATE): Date the song was officially released.
- **vi.** Album (TEXT): Album the song belongs to (if applicable).
- **vii.** LastPlayed (TIMESTAMP): Timestamp of the most recent playback for this song (updated by trigger).

c. Emotions

❖ Columns

- **i.** EmotionID (INTEGER PRIMARY KEY AUTOINCREMENT): Unique identifier for each emotion.
- **ii.** EmotionName (TEXT NOT NULL UNIQUE): Name of the emotion (e.g., Happy, Sad, Energetic).

d. Recommendations

❖ Columns

- **i.** RecommendationID (INTEGER PRIMARY KEY AUTOINCREMENT): Unique identifier for each recommendation.
- **ii.** UserID (INTEGER): Foreign key referencing the Users table.
- **iii.** MusicID (INTEGER): Foreign key referencing the Music table.
- **iv.** EmotionID (INTEGER): Foreign key referencing the Emotions table.
- **v.** FOREIGN KEY (UserID) REFERENCES Users(UserID): Ensures recommendations are linked to valid users.

vi. FOREIGN KEY (MusicID) REFERENCES Music(MusicID): Ensures recommendations point to existing songs.
vii. FOREIGN KEY (EmotionID) REFERENCES Emotions(EmotionID): Ensures recommendations are associated with valid emotions.

e. PlaybackHistory

❖ Columns

i. PlaybackID (INTEGER PRIMARY KEY AUTOINCREMENT): Unique identifier for each playback event.
ii. UserID (INTEGER): Foreign key referencing the Users table.
iii. MusicID (INTEGER): Foreign key referencing the Music table.
iv. EmotionID (INTEGER): Foreign key referencing the Emotions table.
v. PlayTime (TIMESTAMP DEFAULT CURRENT_TIMESTAMP): Timestamp of the playback event.
vi. FOREIGN KEY (UserID) REFERENCES Users(UserID): Ensures playback history entries are linked to valid users.
vii. FOREIGN KEY (MusicID) REFERENCES Music(MusicID): Ensures playback history entries reference existing songs.
viii. FOREIGN KEY (EmotionID) REFERENCES Emotions(EmotionID): Ensures playback history entries are associated with valid emotions.

f. UserPreferences

❖ Columns

i. PreferenceID (INTEGER PRIMARY KEY AUTOINCREMENT): Unique identifier for each user preference.
ii. UserID (INTEGER): Foreign key referencing the Users table.
iii. PreferredEmotionID (INTEGER): Foreign key referencing the Emotions table.
iv. FOREIGN KEY (UserID) REFERENCES Users(UserID): Ensures user preferences are linked to valid users.
v. FOREIGN KEY (PreferredEmotionID) REFERENCES Emotions(EmotionID): Ensures user preferences align with valid emotions.

g. Indexes

i. ***Users.Email* -** Enables efficient user lookups by email for login and other operations.
ii. ***Music.Title* -** Allows for faster searches and filtering of songs by title.

Triggers

- ***UpdateLastPlayed (AFTER INSERT ON PlaybackHistory)* -** Automatically updates the LastPlayed field in the Music table whenever a new playback entry is added, ensuring the most recent play time is always reflected.

Relationships

- One user can have many playback history entries (one-to-many).
- One song can be in many playback history entries and have many recommendations (one-to-many).
- One emotion can be associated with many recommendations and playback history entries (one-to-many).
- One user can have one set of preferences (one-to-one).

User Manual

Getting Started

1. Download and install the EmotionTune application.
2. Create an account by providing a username, email address, and secure password.
3. Customize Preferences

- Access the "User Preferences" page.
- Use the radio buttons to rate your preference level (1-5) for different emotions (Happy, Energetic, Calm, Sad, Nervous).
- Select your favorite genres from the provided list (Pop, Jazz, Hip Hop, Rap, Country, Rock).
- Click the "Save Preferences" button to save your selections.

4. Explore Music

- Browse through the music library, exploring songs by artists, genres, or emotions.
- Use the search bar to find specific songs or artists.
- View song details, including album art, artist information, and release date.

5. Play Music

- Select a song to play.
- Use the playback controls (play, pause, skip, etc.) to manage playback.
- Adjust volume and playback speed (if available).

6. View Playback History

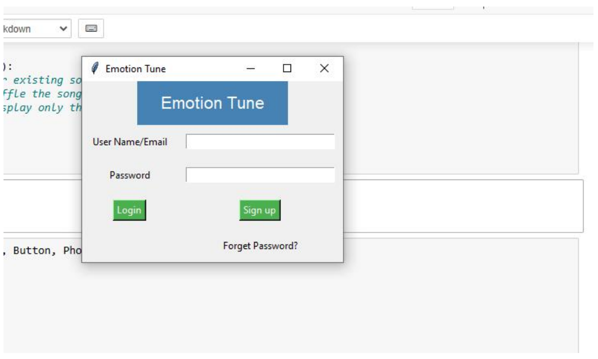
- Access the "Playback History" page.
- View a list of recently played songs with song title, artist, and play time.
- Filter the history by date, genre, or emotion.
- Use the "Replay" button to play a previously listened song.

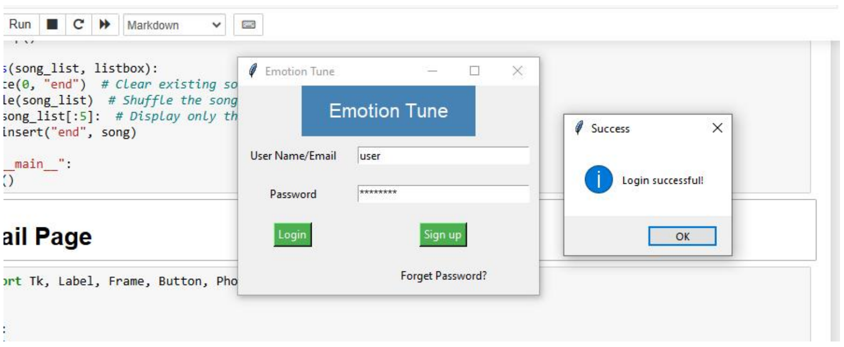
7. Manage Account

- Update account information (username, email, password).

Screenshots

❖ Login Screen





```
s(song_list, listbox):
ce(0, "end")  # Clear existing so
le(song_list)  # Shuffle the song
song_list[:5]:  # Display only th
insert("end", song)

_main__":
()
```

ail Page

```
ort Tk, Label, Frame, Button, Pho
```

Emotion Tune

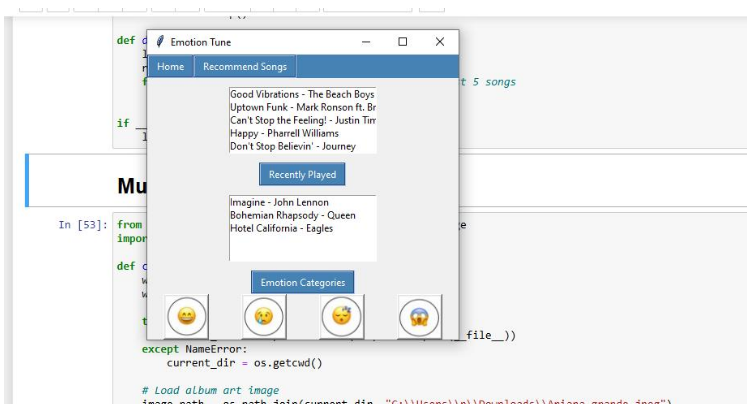
User Name/Email: user

Password: ********

Success

Login successful!

❖ Home Screen (main music library view)



Emotion Tune

Good Vibrations - The Beach Boys
Uptown Funk - Mark Ronson ft. Br
Can't Stop the Feeling! - Justin Tim
Happy - Pharrell Williams
Don't Stop Believin' - Journey

Imagine - John Lennon
Bohemian Rhapsody - Queen
Hotel California - Eagles

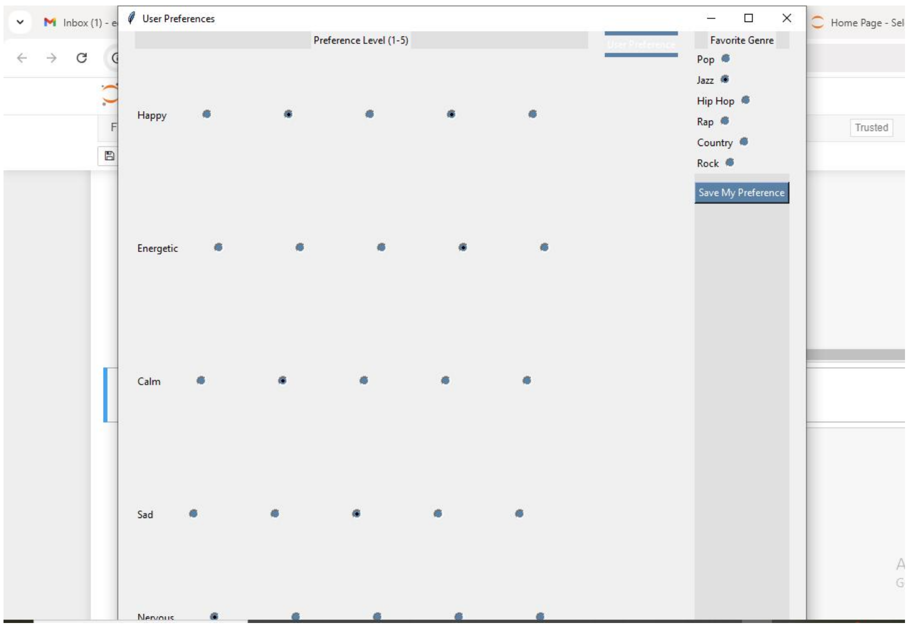
```
In [53]: from
         impor

         def

    except NameError:
        current_dir = os.getcwd()

    # Load album art image
```

❖ User Preferences Page



User Preferences

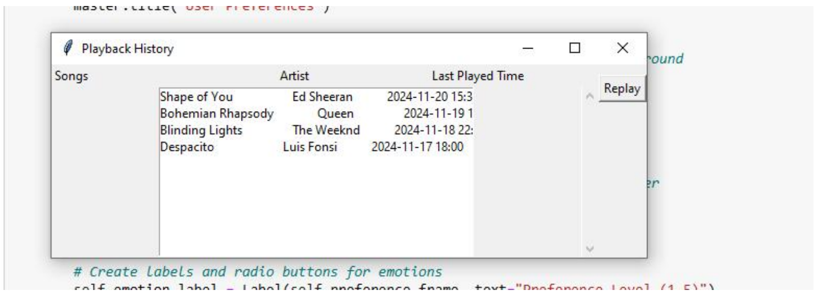
Preference Level (1-5)

Happy
Energetic
Calm
Sad
Nervous

Favorite Genre

Pop
Jazz
Hip Hop
Rap
Country
Rock

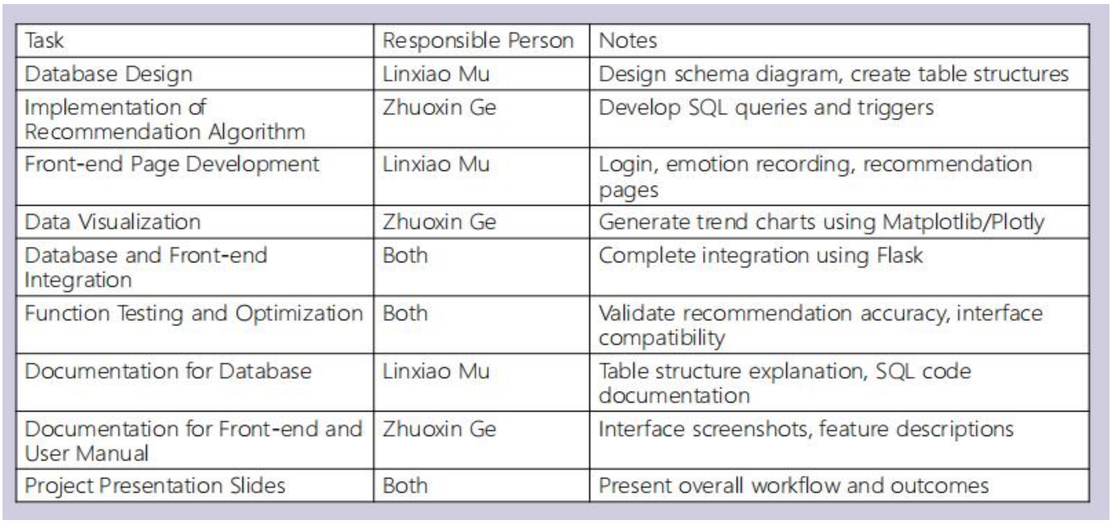
❖ Playback History Page



```
Songs                         Artist              Last Played Time
Shape of You           Ed Sheeran      2024-11-20 15:3
Bohemian Rhapsody           Queen        2024-11-19 1
Blinding Lights        The Weeknd      2024-11-18 22:
Despacito            Luis Fonsi     2024-11-17 18:00
```

```
# Create labels and radio buttons for emotions
```

Team Member Contributions



Task	Responsible Person	Notes
Database Design	Linxiao Mu	Design schema diagram, create table structures
Implementation of Recommendation Algorithm	Zhuoxin Ge	Develop SQL queries and triggers
Front-end Page Development	Linxiao Mu	Login, emotion recording, recommendation pages
Data Visualization	Zhuoxin Ge	Generate trend charts using Matplotlib/Plotly
Database and Front-end Integration	Both	Complete integration using Flask
Function Testing and Optimization	Both	Validate recommendation accuracy, interface compatibility
Documentation for Database	Linxiao Mu	Table structure explanation, SQL code documentation
Documentation for Front-end and User Manual	Zhuoxin Ge	Interface screenshots, feature descriptions
Project Presentation Slides	Both	Present overall workflow and outcomes

References

Van Rossum, G., & Drake, F. L. (2009). *Introduction to python 3: python documentation manual part 1*. CreateSpace.

https://dl.acm.org/doi/abs/10.5555/1592885

Venkatesan, V., & Hariharan, G. An Implementation Approach Towards The Automation Of Document Formatting Using Python. *International Journal of Aquatic Science*, *12*(02), 3946-3959.

https://www.journal-aquaticscience.com/article_135903_3cb3777731dafc0976d56144e5e48a19.pdf

Kreibich, J. (2010). *Using SQLite*. " O'Reilly Media, Inc.".

https://books.google.co.ke/books?hl=en&lr=&id=HFIM47wp0X0C&oi=fnd&pg=PR7&dq=%EF%82%B7SQLite3+documentation&ots=Fj-BfYtm6R&sig=-aSiCWwEt3YKdt8hZrXSCEG2TsQ&redir_esc=y#v=onepage&q=%EF%82%B7SQLite3%20documentation&f=false

Ronacher, A. (2021). *Flask documentation*.

https://readthedocs.org/projects/flask-russian-docs/downloads/pdf/0.9/

Dirix, M., Muller, A., & Aranega, V. (2013). Genmymodel: an online uml case tool. In *ECOOP*. https://hal.science/hal-01251417/