Design Document for Cyclone Sounds

Group 1_srijita_7

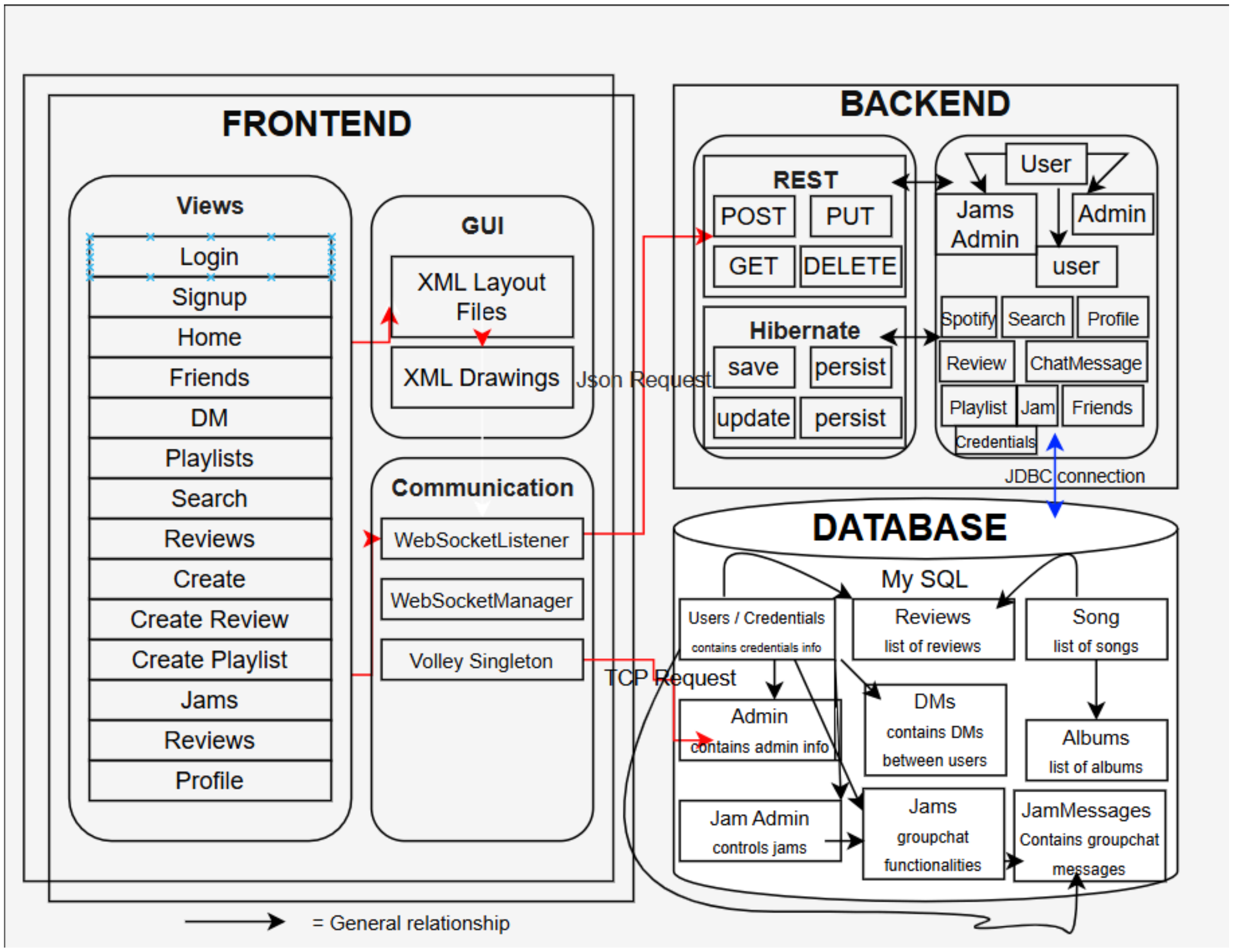
Alex DeWitt: 25% contribution

Jack Danby: 25% contribution

Mark Seward: 25% contribution

Martin Kochadampally: 25% contribution

Block Diagram



Frontend

- Authentication & User Management Users authenticate via Login (GET) or register via Signup (POST) with role selection (User, Admin, JamManager). The ProfileActivity displays user details and supports updates (PUT) or account deletion (DELETE). The HomeActivity acts as the central navigation hub connecting all features.
- Social Features FriendsActivity manages connections (GET) and handles friend requests (POST). DMActivity provides real-time bidirectional messaging between users using WebSockets and a RecyclerView interface.
- Music Discovery & Playlists SearchActivity allows querying for Songs or Profiles (GET). Users manage collections in MyPlaylistsActivity, using TableLayouts to create playlists and add/remove songs via Volley requests (POST/DELETE).
- Jams (Group Sessions) JamsActivity lists active sessions; creation is restricted to Admin or JamManager accounts. IndividualJamActivity connects users via WebSockets for live group chat and features a song suggestion system where users send requests for Admin approval.
- Reviews Users submit song ratings (0-5) and descriptions via CreateReviewActivity (POST). MusicActivity lists these reviews, allowing community interaction through upvotes/downvotes (PUT) and entry deletion.

Backend

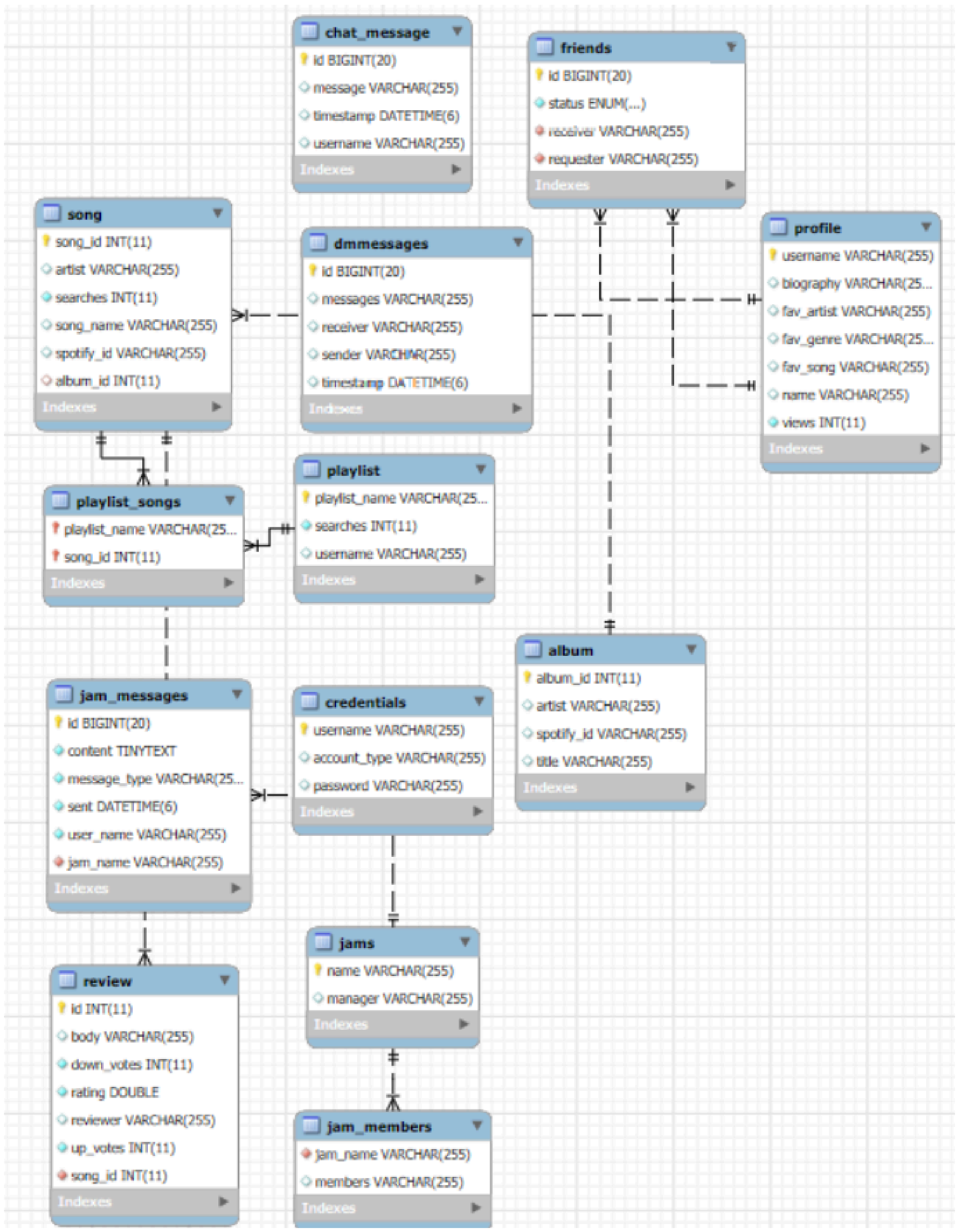
Communication

- Get: Used mainly for retrieving specific objects using an identifier and to search for objects using a search key, but is also used to list all.
- Put: Used to update a specific object in the database using an identifier.
- Post: Used to create new objects into the corresponding database.
- Delete: Used to delete an object based on a specified identifier.

Controllers

- Credentials (Users) - For full CRUD operation user credentials and has one to one relation with Profiles.
- Profile - Stores some info about the User and is connected to Credentials by username variable and has a One to Many relationship with Reviews
- Reviews - Allows users to review songs and has a Many to One relation with Songs and is connected to songs by the songs id and is connected to Profiles by username
- Songs - Has a Many to Many relation with Playlists and it stores information about various songs, which it pulls from the Spotify API.
- Playlist - One to One relation with Jams, and allows users to store songs they like in groups for easy access.
- Jams - Allows user to chat in a groupchat and make playlists together in real time.
- DMs - Allows users to to talk with one another, and its links the message with the two people talking based on thier usernames.
- Search - Users can search Songs ans other Profiles and results are ordered by popularity

PUT THE TABLE RELATIONSHIPS DIAGRAM on this fourth page! (Create the picture using MySQLWorkbench)


chat_message
id BIGINT(20)
message VARCHAR(255)
timestamp DATETIME(6)
username VARCHAR(255)
Indexes
friends
id BIGINT(20)
status ENUM(...)
receiver VARCHAR(255)
requester VARCHAR(255)
Indexes
song
song_id INT(11)
artist VARCHAR(255)
searches INT(11)
song_name VARCHAR(255)
spotify_id VARCHAR(255)
album_id INT(11)
Indexes
dmmessages
id BIGINT(20)
messages VARCHAR(255)
receiver VARCHAR(255)
sender VARCHAR(255)
timestamp DATETIME(6)
Indexes
profile
username VARCHAR(255)
biography VARCHAR(25...
fav_artist VARCHAR(255)
fav_genre VARCHAR(25...
fav_song VARCHAR(255)
name VARCHAR(255)
views INT(11)
Indexes
playlist_songs
playlist_name VARCHAR(25...
song_id INT(11)
Indexes
playlist
playlist_name VARCHAR(25...
searches INT(11)
username VARCHAR(255)
Indexes
album
album_id INT(11)
artist VARCHAR(255)
spotify_id VARCHAR(255)
title VARCHAR(255)
Indexes
jam_messages
id BIGINT(20)
content TINYTEXT
message_type VARCHAR(25...
sent DATETIME(6)
user_name VARCHAR(255)
jam_name VARCHAR(255)
Indexes
credentials
username VARCHAR(255)
account_type VARCHAR(255)
password VARCHAR(255)
Indexes
review
id INT(11)
body VARCHAR(255)
down_votes INT(11)
rating DOUBLE
reviewer VARCHAR(255)
up_votes INT(11)
song_id INT(11)
Indexes
jams
name VARCHAR(255)
manager VARCHAR(255)
Indexes
jam_members
jam_name VARCHAR(255)
members VARCHAR(255)
Indexes