Caroline Chen yc3756

Cherry Chu ccc2207

Adam Carpentieri ac4409

Haomin Gui hg2533

Part 0

We met with Rohit in the Discord chatroom on November 2nd. We discussed our new revised plan and the MVP with additional features later on.

Part 1

Background Information: We decided to change the project very significantly based on the feedback Rohit initially provided during the class meeting. He correctly pointed out the potential legal complications with a streaming audio broadcasting service.

1. Allow users to visit chatrooms that are based on musical genres. In the chatroom people can see other users' current Spotify song they are listening to as well as their previous recent history. Lastly, they will be able to add songs to each other's current play queue.
2. Our users are Spotify users who are interested in sharing and talking about the music they like with other similar minded people.
   For authentication the listener will create an account with a password on the chat server. We will use Oauth or OpenID to authenticate users and create tokens. We will also need them to authorize their Spotify account for access to the API.
3. For the demo, we will show the user interface for the chatroom. At least two users will demonstrate the chat functionality and the playlist sharing functionality.
4. We will use Sqlite database in Java to store user information - associating their accounts with the spotify API token. We also will have a chatroom table. We may add extra data for the chatrooms such as chat transcripts and shared songs.
5. We will be using the Spotify API which is a REST api to poll playlist data for the users.

Part 2

1. As a chatroom user, I want to see other participants to see what they are listening to or have previously listened to.

   My conditions of satisfaction are: The UI will show the participants current and recently played music.

2. As a chatroom user, I want to be able to talk to other chatroom users about the music we are listening to and the genre that the chatroom is about.

   My conditions of satisfaction are: the user can send messages into the chatroom and see the messages that other users type and send.

3. As an active participant I want to share new music with other users and also listen to shared music from them.

   My conditions of satisfaction are: 1. When you share the music, it should appear on other listener's playing queue in their Spotify app. 2. The UI will reflect that a song has been shared in the chat window.

Part 3 (acceptance tests)

1. When a user logs on to a room with other participants, they will see **only** the recently and currently playing songs of the **chatroom participants**. A failure condition would be if the user does not see the data at all, or that they see data from other chatrooms. Another way to test is for another user to change their song, and for the first participant to verify they see the change in 30 seconds or less.
2. The first user can send a message to the chatroom, and another user can verify that they see the corresponding message. And vice versa. After this we can scale to three users to make sure it works for all participants. Lastly, we will share screenshots to ensure the messages arrive in the correct order. A failure condition would be when the screenshots do not match.
3. When a user shares a song, the song should appear at the end of the queue of the other participant(s). A failure condition would be when their app does not display the song to be played. The chatroom should have a log for the shared song. The failure condition would be when the chatroom does not display this information.

Part 4

For the backend plan to utilize the same stack that was introduced in the Tic tac toe. That would include Checkstyle, Junit, Emma, spotbugs.

For frontend, we will use Javascript, MaterialUI and React. Tools include Eslint for static code analysis and JEST for testing.