

Startup Submission 07 - Adding a Database

SoundRoom Team

December 14, 2016

1 Outstanding Bugs

- Need to add authorization to some routes
- Room does not get deleted if all participants exit room
- User is able to join multiple rooms
- We don't have routes working for the landing page, signup page, and the login page
- The host should be able to delete a song but we currently don't support that feature
- The save playlist feature only saves the playlist that is currently on the queue and not all of the songs that have been played previously
- Sometimes when you add a song, it automatically gets played even though it has lower likes than another song in the queue
- Twilio's API only allows the client to send text messages to numbers that are verified on the user's Twilio account. Thus you cannot share the access code to anyone
- The Currently Playing Song header on the participants page in the room doesn't show the right song that is currently playing from the host

2 Special Server Setup Procedure

This isn't a special server setup procedure, but his has to do with running our application properly. Once you start the server and head onto localhost, you need to add `?user_id = 000000000000000000000001` to the end of the url to sign in as user 1 (`localhost : 3001?user_id = 000000000000000000000001`). Currently we have 4 users, so you can sign in with each of the four users on different browser tabs, just make sure to change the 1 in the url to 1, 2, 3 or 4. To test the socket.io feature, just sign in with different users on different browser tabs and join a room to add songs to a playlist. The whole room component is real-time meaning that whenever you add a song with one user, it should appear in the room for a different user who is also in the room.

When you reset the db, it going to take you to the main homepage but its not going to have the ?user_id query parameter attached to it. All you need to do is add that user_id query (*user_id* = 000000000000000000000001) at the end of the url so that the app can function properly.

3 Individual Contributions

- **Aarsh Patel**

- GET /user/:userId/account.info → gets the user account info
- GET /user/:userId/playlists → gets the user's playlist
- PUT /user/:userId/account.info → updates the user's profile
- Implemented the SocketIO for real time event communication between participants in the room
- Implemented the share access code button to share the access code of a room to other friends by text using Twilio's API
- Added a Room Hosting section on the Sidebar
- Converted most of the helper functions in the src/server.js file to use mongodb instead of the mock database

- **Bhavik Jain**

- DELETE /room/delete → deletes a room
- POST /room/host/ → gets the room's host
- DELETE /room/:roomId/participants/:participantId → removes participant from room
- JSON Tokens and Authentication

- **Ronit Arora**

- POST /room/songlike → likes a song in the room playlist
- POST /room/participants → gets the room's participants
- Updated the schema according to mongoDB format
- Replaced numeric ID with Object ID according to mongoDB
- Implemented POST /room/save

- **Siddarth Patel**

- POST /resetdb → resets the mock database
- GET /song/:songId → gets the song from the song track id
- POST /room/sharecode/ → shares the code to another friend using Twilio's API
- Implemented the hide Close Room and the hide Exit Room button functionality

- **Justin Martinelli**

- POST /room/save → save a room's playlist to the user's account

- /room/:songId/new_song → adds a song to the room playlist
- Implemented components for the landing, signup and login pages

- **Lynn Samson**

- POST /createroom/ → creates a room
- POST /joinroom/:userId → lets a user join a room
- POST /room/data → returns back the room's data such as the participants, playlist and the host
- Implemented the feature that the Save Playlist button only shows up when there are songs in the playlist
- Implemented the queue liking feature: a song gets played next based on the number of likes it has