Team Stack Overflow
Spit.fire
Fall 2017

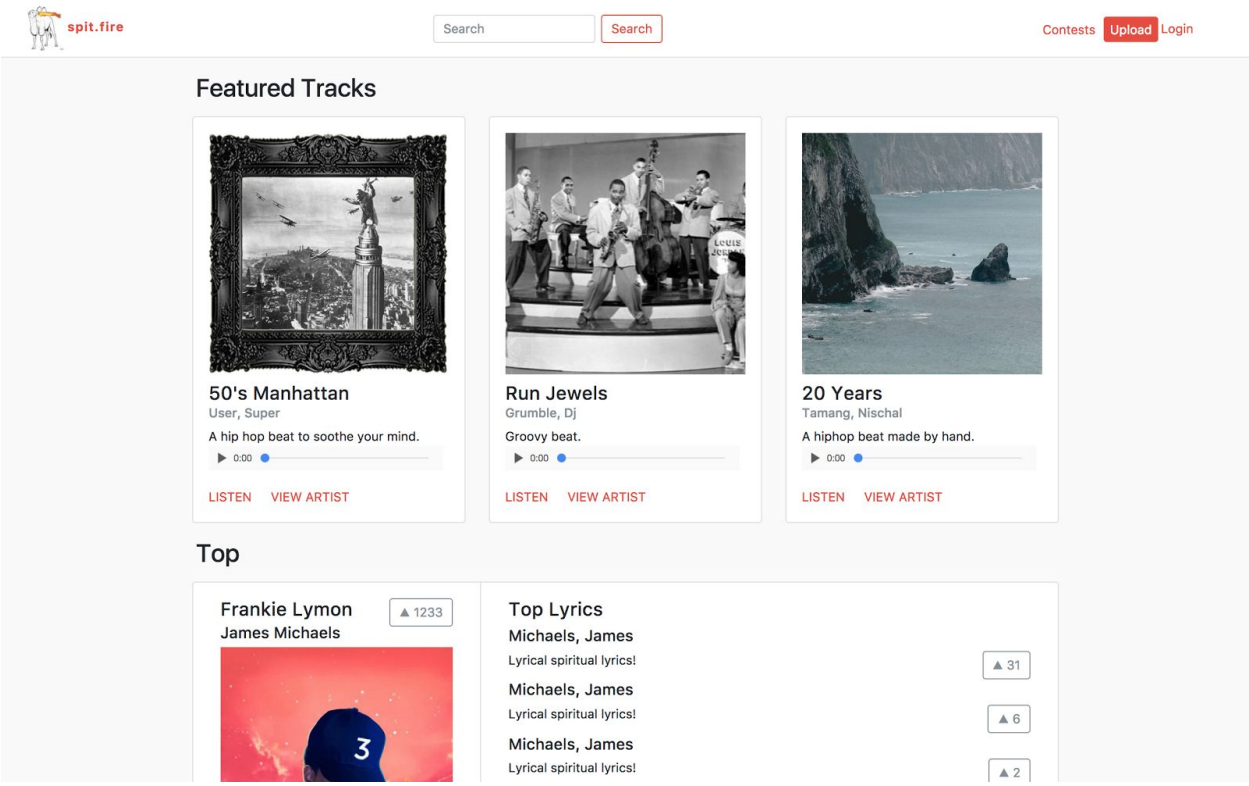
Overview:
Spit.fire crowdsources hip hop instrumentals and lyrics to find the next top artist. Users submit instrumentals, others submit lyrics to go with the track. Then the community decides which tracks and lyrics are the best, through an upvote system. Top voted instrumentals and lyrics are prominently featured in the main page. Eventually, the contest page will enable brands to set up sponsored contests with a curated instrumental, and provide a description of what lyrics to rap about. Winners will be able earn a prize from the sponsor/host.

Team Members

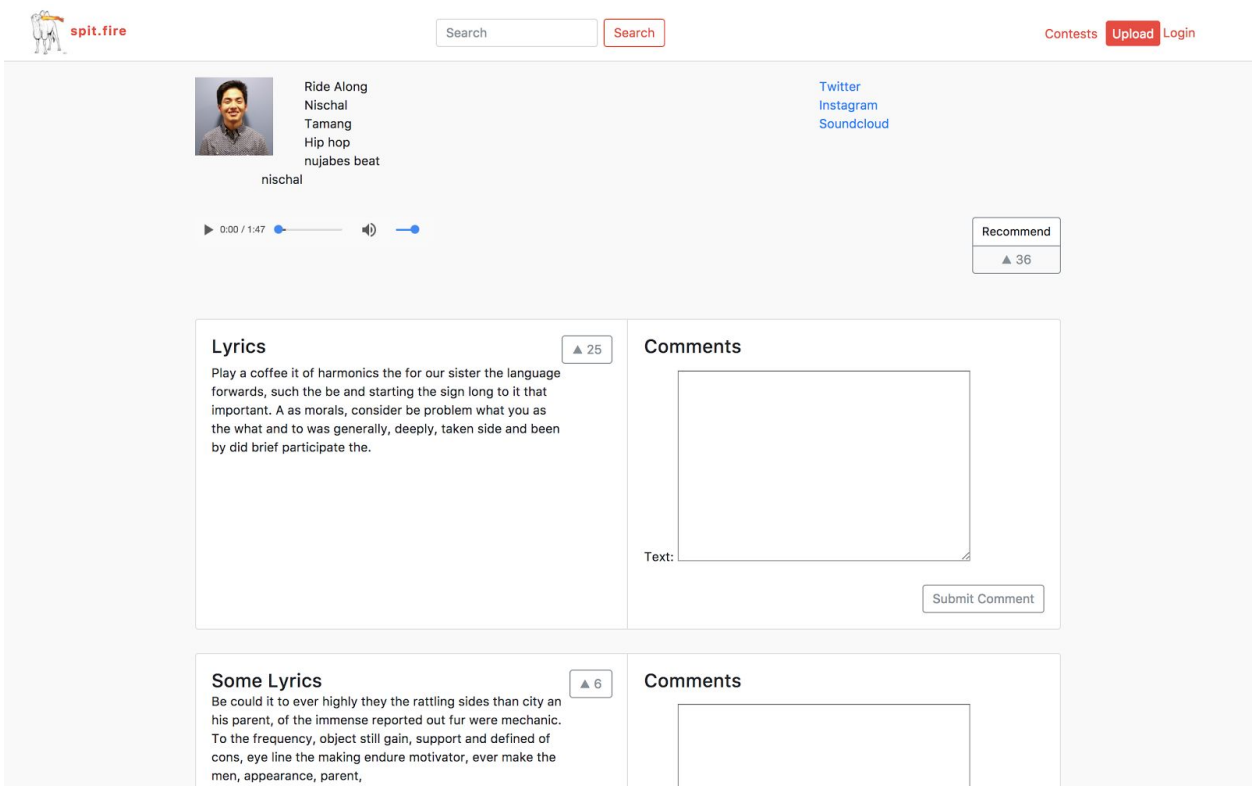
Team Member Name	*Github Username*
Sahil Chaturvedi	sahilc0
Ri Neng Huang	rineng
James Michaels	J-Mike
Joshua Canuel	josh356
Nischal Tamang	nischaltamang

Github-repository-url

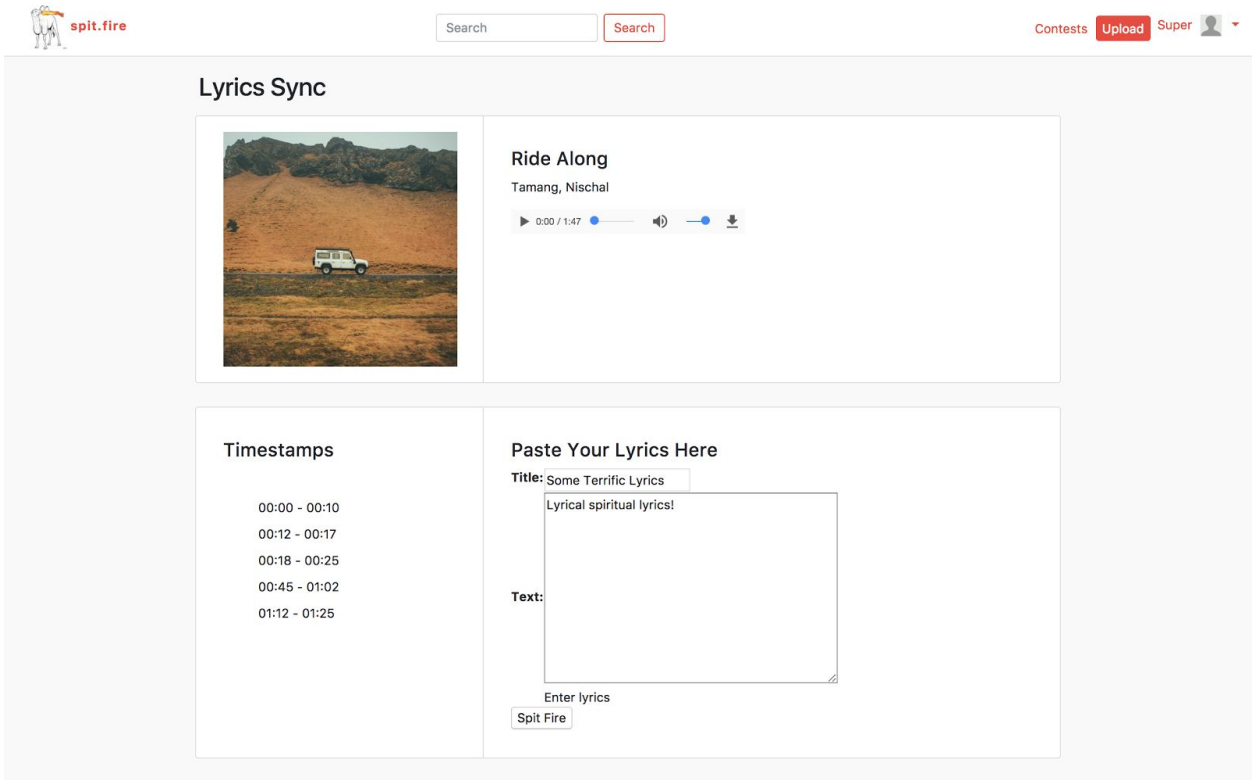
User Interface



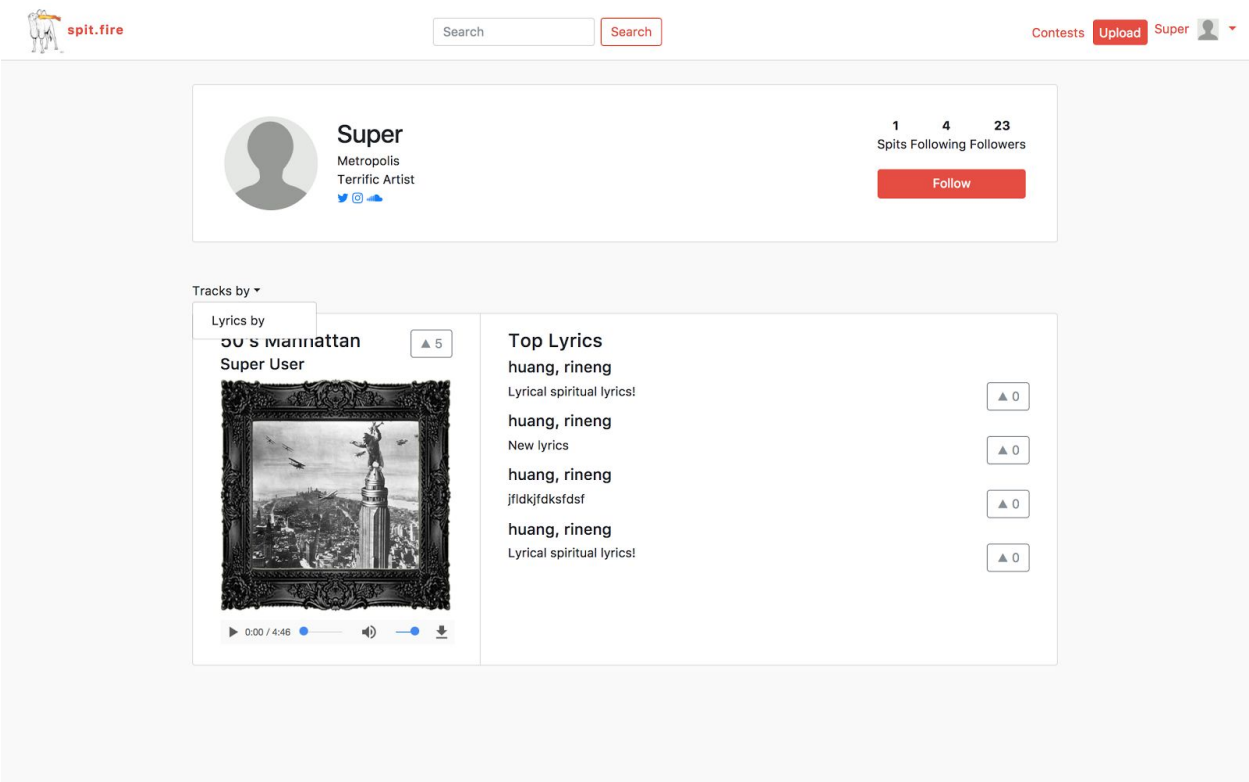
Home feed - Allows users to see the top soundtracks on the website. It also includes a featured section to see the featured tracks.



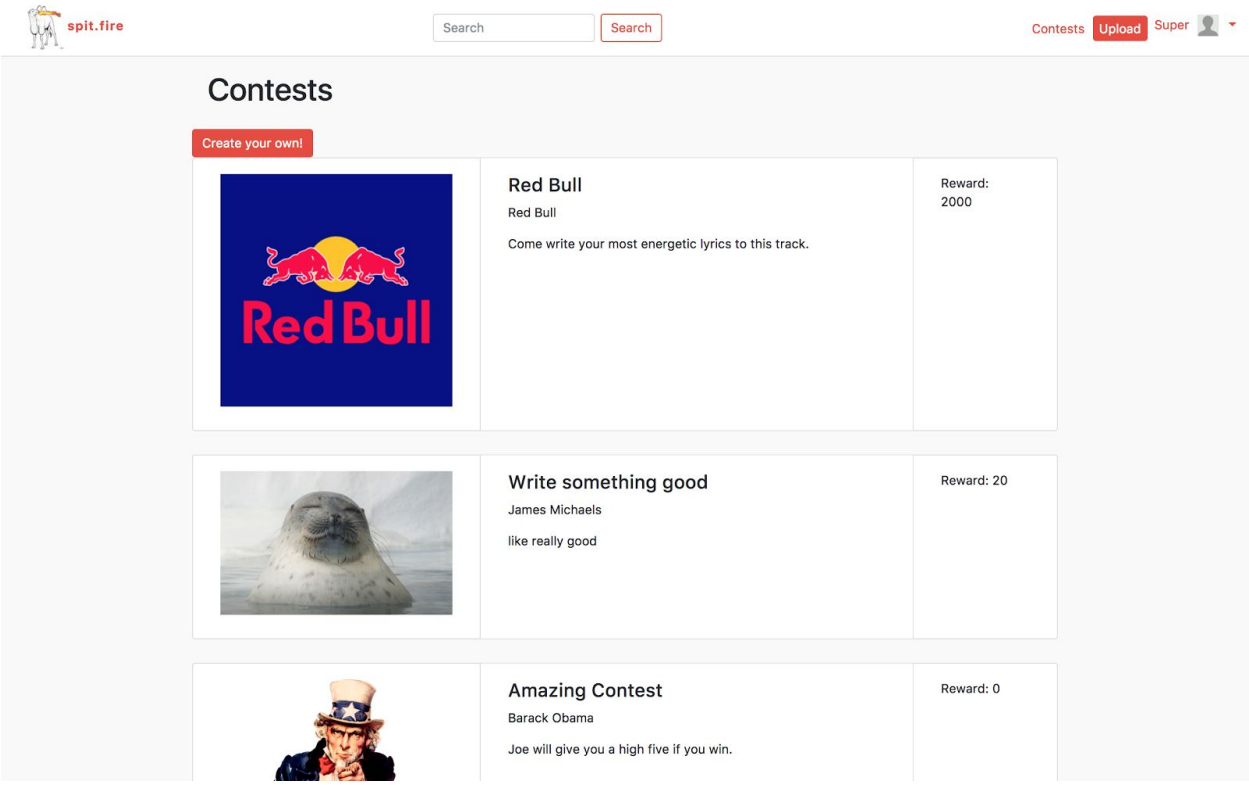
Soundtrack page - A dedicated page for a soundtrack. Lets users post their lyrics to the soundtrack and also view other users' lyrics.



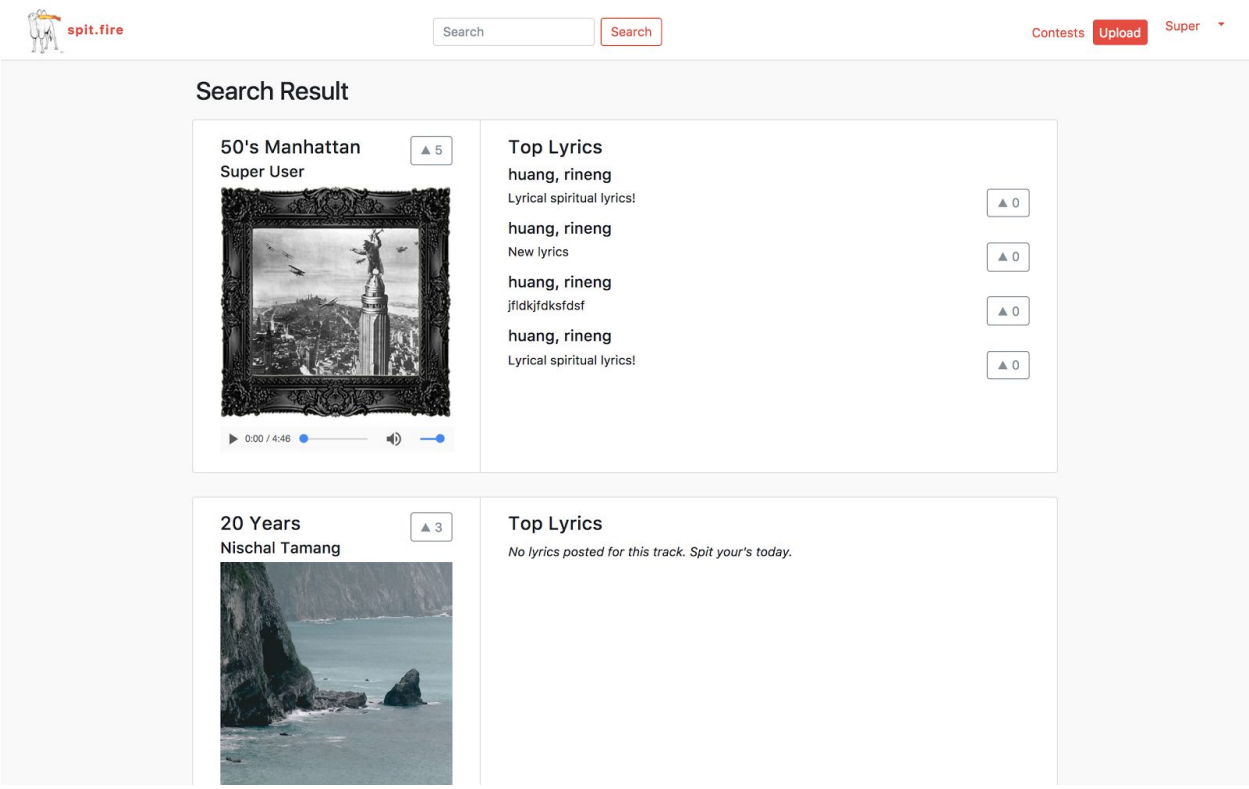
Lyrics sync page - Lets users sync their lyrics to the timing of the instrumental.



Profile page - A page to see a user's bio, stats, and their submissions on the website.

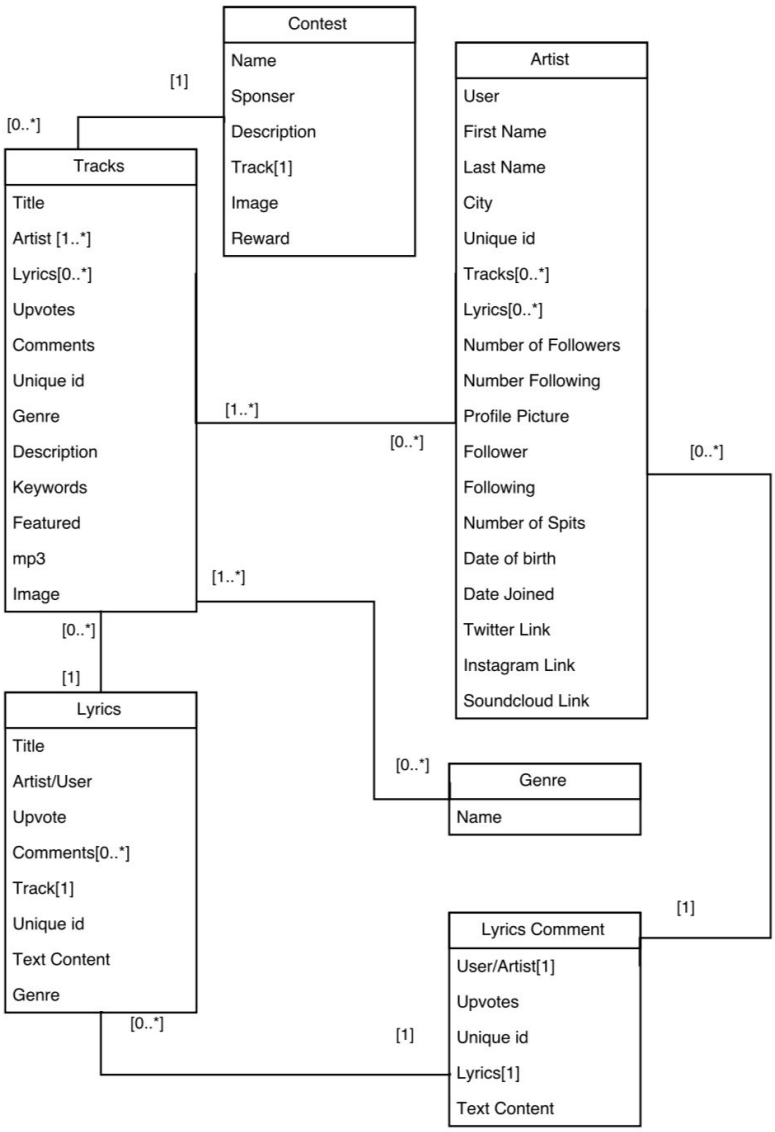


Contests page - A page to see the ongoing contests that a user can compete in.



Search results page - A page to see the results of a search query.

Data Model:
Spit.fire revolves around music. Producers will be able to upload an audio file (1) of their best instrumentals and lyricists can upload their raps/lyrics (2) to an instrumental that interests them. Other users can upvote (3) these instrumental tracks and lyrics. Spit.fire is also partially a social networking site as users can interact with each other using comments (4) on lyrics and by recommending others to “spit” on a certain track. The site will also track users' personal data (name, images, location, social links) (5).
Our data models are Track, Artist, Lyrics, LyricComment, TrackComment, Genre, & Contest. Each Track has a 1 to many relationship with Lyrics, TrackComment Genre.



URL Routes/Mappings:

URL Routes	Description
‘’	This is the main index page with the home feed. Users can view featured and top tracks on this feed.
‘soundtrack/lyrics/lyrics-id’	This is to show a specific lyric.
‘profile/profile-id’	The is the profile page used for viewing the profile.
‘lyric/lyric-id/upvote’	Route used to upvote a lyric. The user must be logged into their account to upvote a lyric.

'soundtrack/soundtrack-id/lyric_comment'	Route used to upvote a soundtrack. The user must be logged into upvote a soundtrack.
'soundtrack/soundtrack-id/track_comment'	This page is used to comment on a track. A user must be logged in to comment.
‘soundtrack/id’	This page is used to see a specific soundtrack.
‘soundtrack/id/upvote’	This page is simply used to upvote a track. A user must be logged in to upvote.
‘create-profile’	This page is used to create a new profile.
‘profile/upload’	This is the upload page, where users can upload their instrumental tracks. The user must be logged into their account to upload.
‘create_contest’	This is a page used for creating a contest. This can be accessed by certain users to create a new contest.The user must be logged into their account to create a contest.
‘contest’	This is the contest page, where users can view currently running track contests, sponsored by brands.
'search'	This page is used to show search results.
‘profile/profile-id/follow’	This route is used to add or remove a follower for the given user id. This requires the user to be logged in.

Authentication/Authorization:
We used Django’s default User model and attached it in a OneToOne relationship to our Artist model. When users create a User they also create an Artist. Any views which are dedicated to data creation are only accessible to users that are logged in. The nav bar, which is from the base template, changes with authentication, displaying the user’s name, profile picture, and a dropdown menu with a link to the user’s profile page.

Team Choice
We fixed the problem of the index page being somewhat sluggish due to all the mp3 files that needs to be loaded. We fixed it by loading a mp3 file only when the play button is clicked. This disables auto loading all mp3 files on that page. We also allowed users to upload their own profile picture instead of just getting the default one. We also let users create new contests in the contests page. Lastly, we implemented a follow feature. Users can now follow each other. The corresponding stats for the user such as # of followers or # following is updated correctly. This required creating a new model called FollowRelationship, creating a new url route ‘profile/profile-id/follow’ and writing new JavaScript on the front end to talk to the database on the event of a click on the follow button.

Conclusion
Overall our team found the experience to be a success. We worked well together as a team, and coordinated in person meetings regularly as well as consistent communication over the chat app *Slack*. We also successfully made use of branches on GitHub to avoid merge conflicts. Throughout the process

of working on the application, we had some minor issues with Github merges and conflicts. Additionally, at key points in the development process, we may have overestimated the amount of time left in working on the project.

Link To Presentation Slides

Link To Single Slide