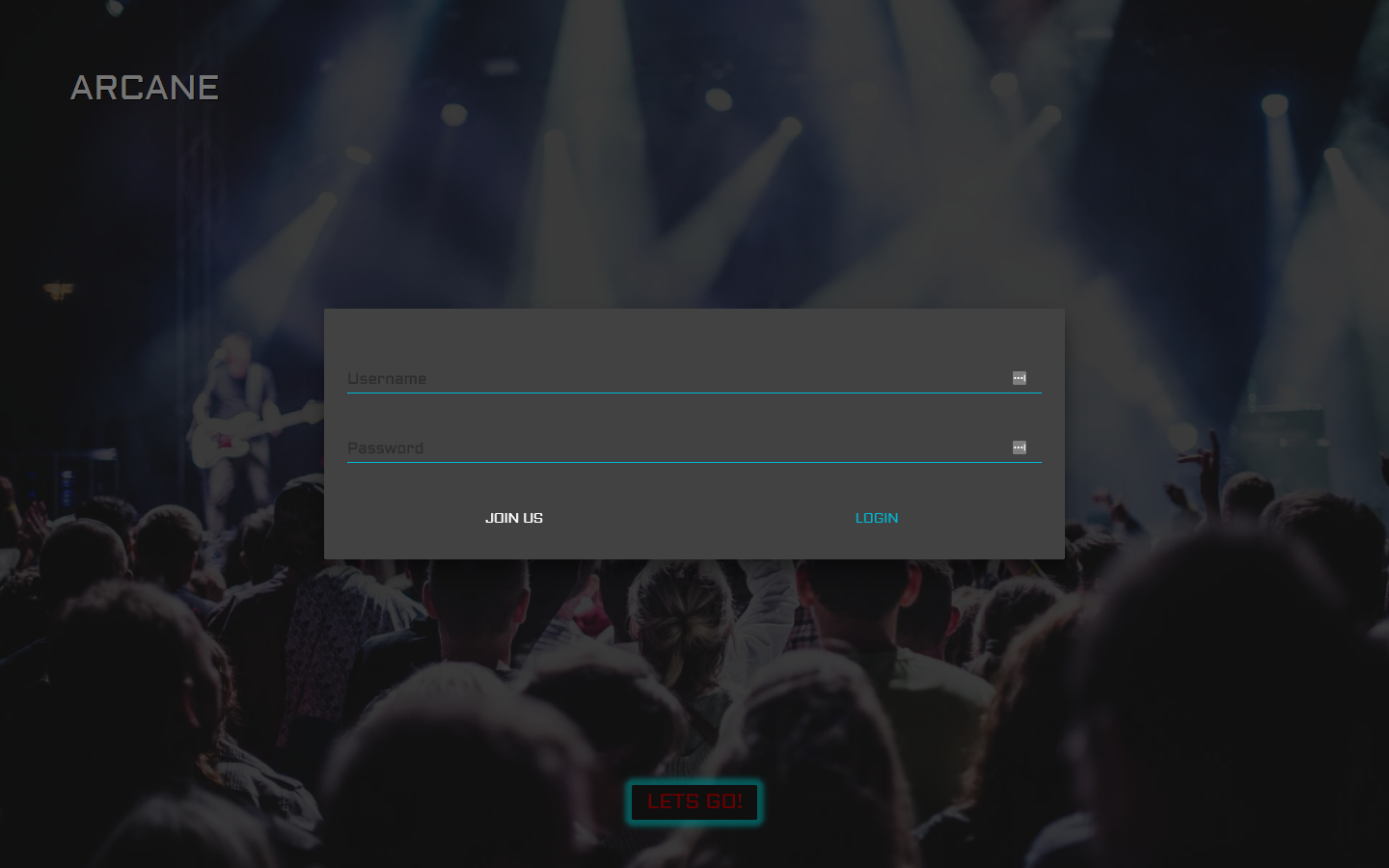
ARCANE

Tyler Scott, Shem Sedrick

https://github.com/tscott8/sr\_project\_research

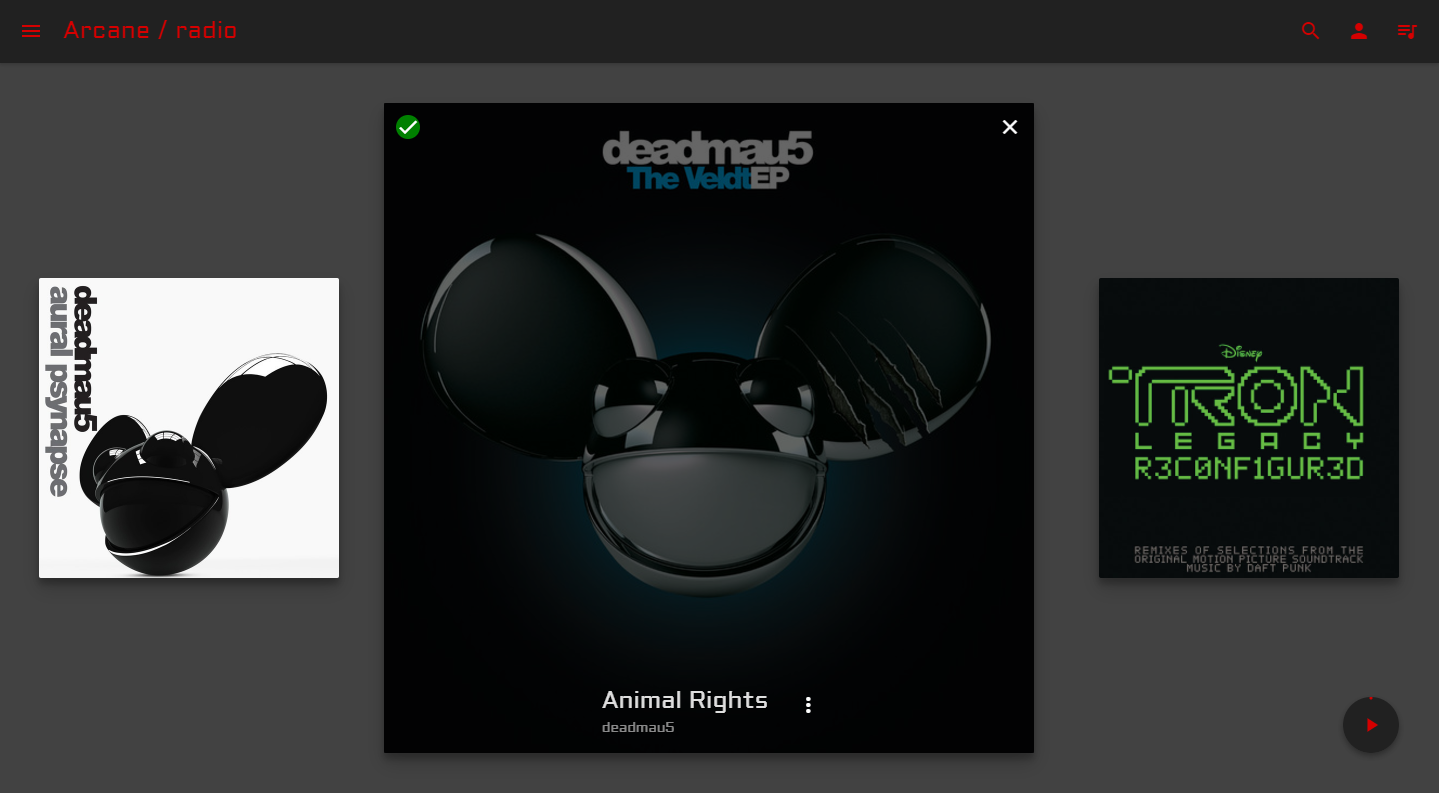
**Goal:**

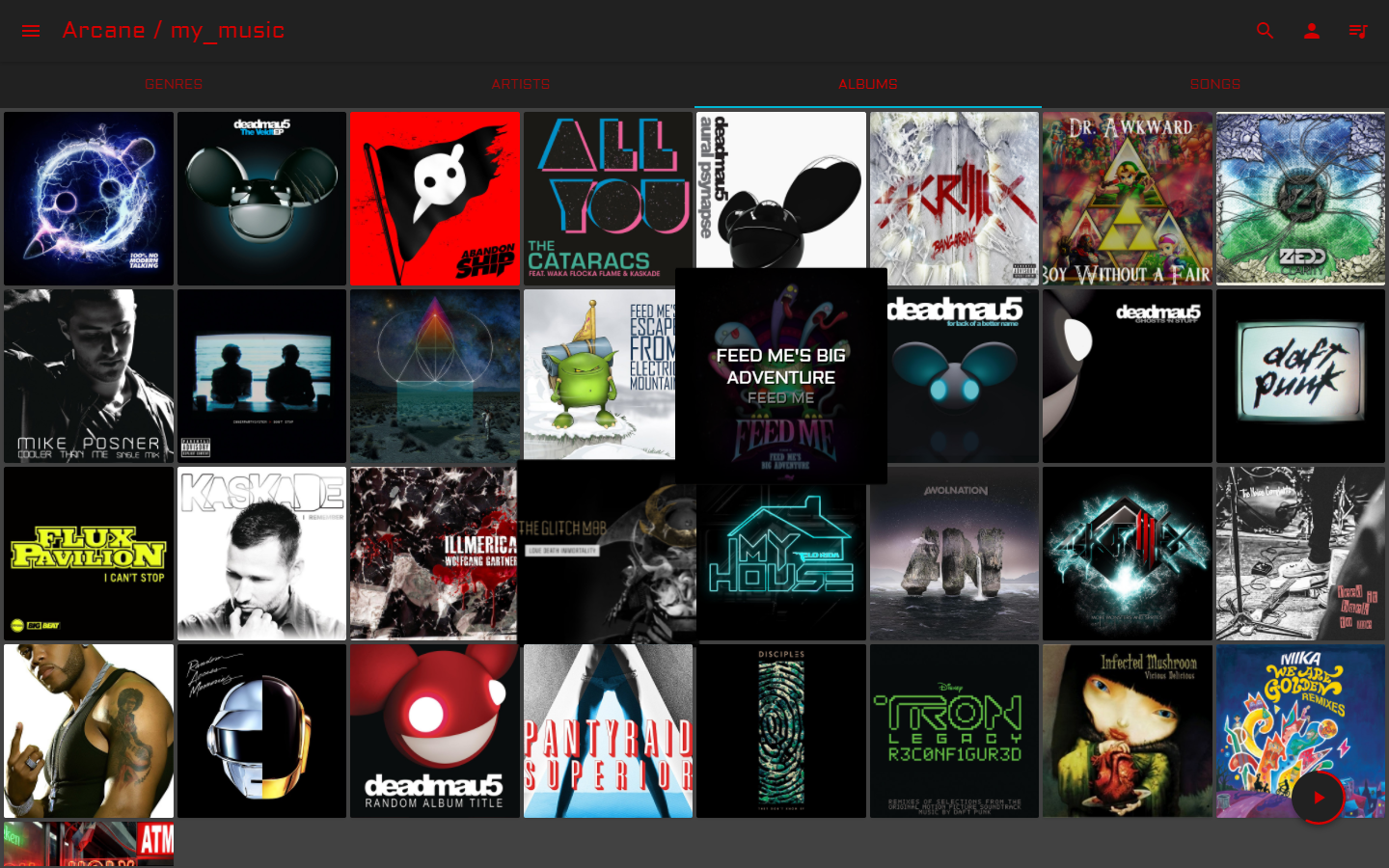
Common music applications on the web exist to service the needs of artists and listeners. Spotify and Pandora are examples of applications that focus on the needs of the listener, while SoundCloud or BandCamp focus more on creating a platform focused on artist. Arcane Streaming is based on the idea that many music enthusiasts want to find exciting, new artists that they have not yet heard. Arcane strives to bring listeners and artists to one location where they can share new music and find new bands.



**Challenges:**

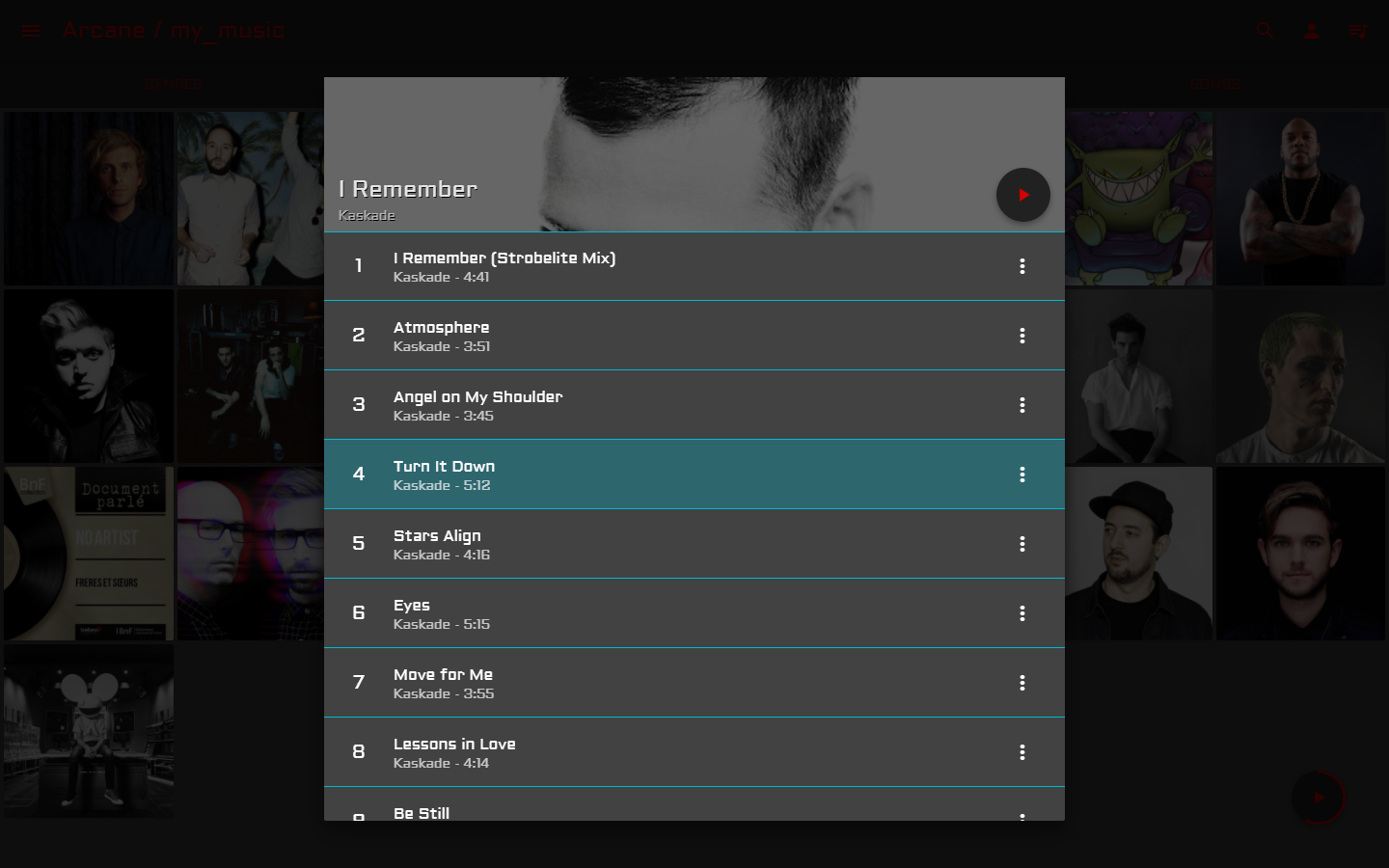
* Parse .mp3 file tags for relevant data
* Allow users to upload mp3 files
* Global access to audio tag
* Globally asynchronous audio control
* Adding songs to the queue
* RESTful API server setup
* Request pagination
* Dynamically changing app theme
* Full database search





**Resources:**

* Python
  + Django - Server
  + Django Rest Framework (DRF) – REST requests
  + Pillow – Image processing
  + SACAD – Retrieve Album Artwork
  + Mutagen – ID3 tags
* Node.js
  + React – User Interface
  + Webpack - Compiler
  + Redux – Data storage
  + Material UI – Material Design Components
  + React Dropzone – Drag and Drop uploading



**How:**

Our application runs with a Django and DRF backend. These python frameworks allowed us to develop models and views quickly in a language we were comfortable with. Our front end utilizes React and Redux with a lot of components from the Material UI framework.

Using Mutagen, we parsed the mp3 files uploaded to populate the genre, album, artist, and track names in our database. SACAD and Pillow were then employed to find and resize album and artist images.