

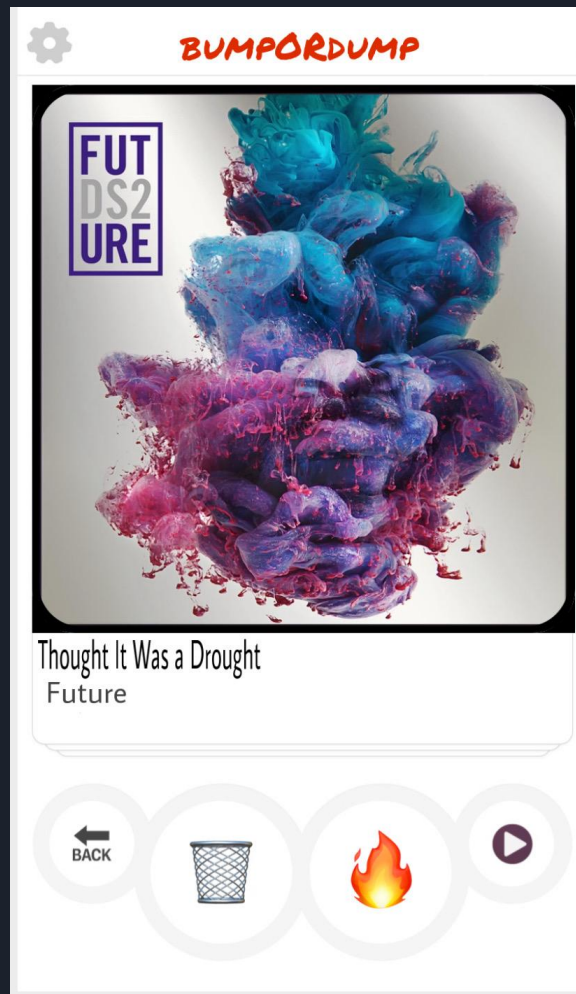


bumpORdump

Jacob Stallings & Quinton Chudik

Overview

- Web application
- Recommendations shown 1-by-1
- Spotify player plays the song
- Allows user to "Like" or "Dislike"
- Add the "likes" to the user's library





Rationale

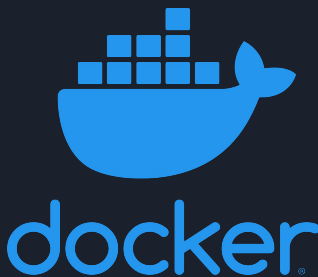
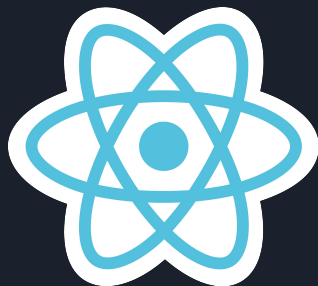
- Spotify offers an extensive API
- Apple Music's API was too rigid
- Learn about deploying web applications
- Fun way to discover new music
- Help artists gain new fans





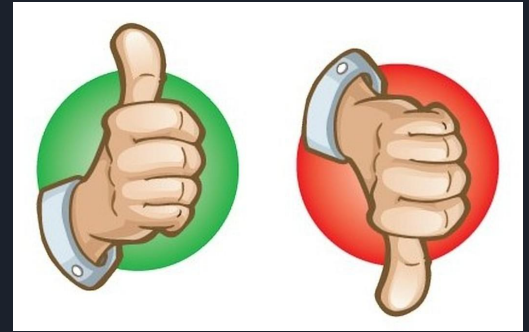
Stack


- React
 - Spotify Player
- Node.js
 - Express
- Docker
- Ngnix
- AWS (for deployment)



Web App Features

- React Spotify Player
 - Plays each recommended song one by one
- "Like"/"Dislike" buttons
- Users can see their score (how many songs they've discovered whether liked or disliked)
- History of "Liked" songs
 - Will be made into a Spotify playlist
- Long-term:
 - Social networking aspect
 - Leaderboard





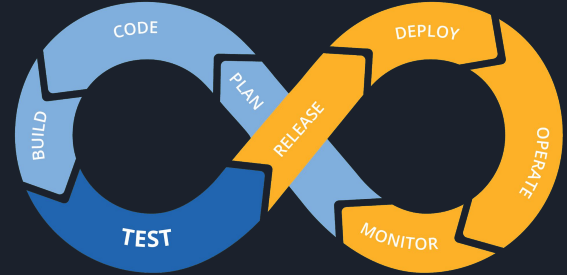
Spotify API Interactions



- User must login to Spotify account
- Recommendations
 - Utilize Spotify recommendations and song qualities
 - Take liked songs into account
- "Liked" songs will make two API calls
 - One to save to user's Songs library
 - One to add to playlist of "liked" songs
- Spotify Developer Token

DevOps

- Containerize the web application with **Docker**
- **Nginx** for load balancing
- **AWS** for hosting
 - Potential for using Amazon EKS (Elastic Kubernetes Service)



Demo

- Showcase functionality of running app
- Show results in Spotify library
- Briefly show backend code and start up procedure
- Explain challenges and what we learned



Challenges

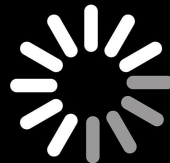
- Spotify API learning curve
- Dockerizing the app
- Load Balancing
- Networking
- Time





Status

- Application is up and running on `bumpORdump.media`
- Future changes include incorporating Kubernetes for auto scaling



Teamwork

Quinn:

- Front End

Jacob:

- Back End

Both:

- DevOps



Questions?

