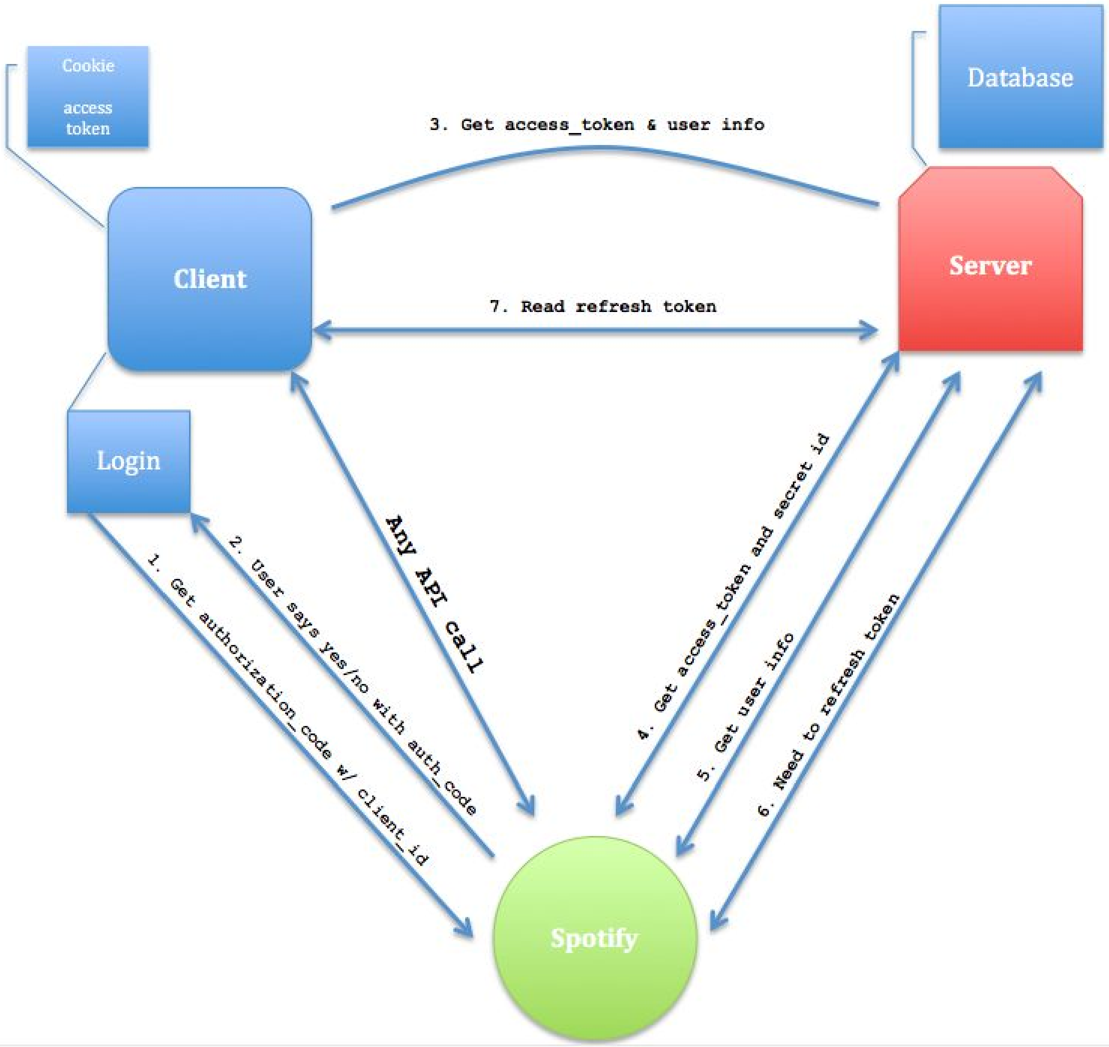
CS 408
Team #17

Ben Maxfield
Terry Lam
Christian Lock
Tom Fanella
Austin Miller

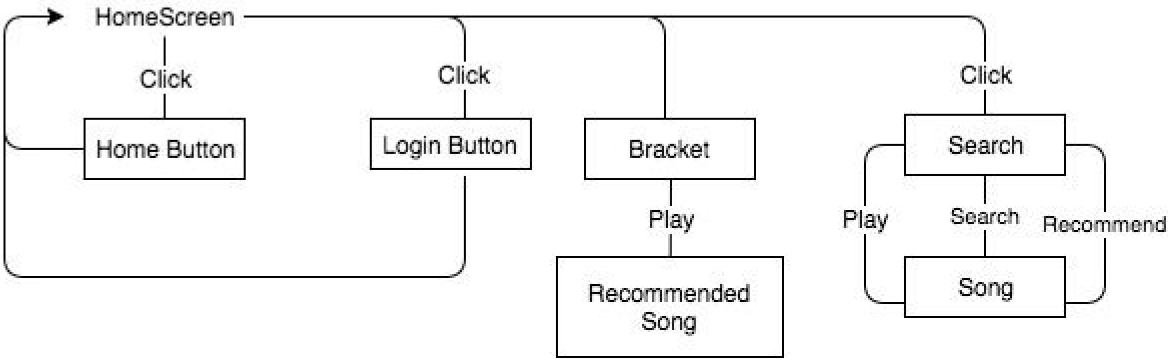
Incremental and Regression Testing Log

Classification:

Back-End Diagram


Cookie
access token
Database
3. Get access_token & user info
Client
Server
7. Read refresh token
Login
Any API call
1. Get authorization_code w/ client_id
2. User says yes/no with auth_code
4. Get access_token and secret id
5. Get user info
6. Need to refresh token
Spotify

Front-End Diagram



The form of incremental testing we used was bottom up. Because our backend is comprised of multiple lambda functions, the most feasible way of testing would be to treat each one as a separate component and individually test them as we built out our application. Since the functions map to a piece of functionality in the frontend, this method would allow us to cover and resolve most of the flaws on an individual component level and then combine once for final stage of testing to make it work with the browser UI, and nothing else (i.e. other services).

Module	**Component**
Component F	Refresh Token
Component G	Vote Song (Post Vote)
Component H	Vote Song (Get Vote)
Component I	History of Brackets
Component J	Bracket Statistics

Component F - Refresh Token

Incremental Testing

Defect #	Description	Severity	How to correct
1	Updated token should be unique specific to that user in the database table	1	Query by user’s specific ID to ensure the correct token is being updated.

Regression Testing

Defect #	Description	Severity	How to correct
1	Access token expected to update for specific user in table only.	2	Make a call to the Spotify API to retrieve their specific ID, and use it to query database table

1	Return value from refresh token lambda function doesn’t keep track of caller.	1	Change return value to just be the Spotify user ID (changes to database happens in the background).

Component G - Vote Song (Post Vote)

Incremental Testing

Defect #	Description	Severity	How to correct
1	A user shouldn’t be able to vote multiple times on the same song matchups	2	Keep track of song IDs that have already been voted on for that specific user.

Regression Testing

Defect #	Description	Severity	How to correct
1	Response from AWS throws a null exception	1	Verify SQL query was updating the correct table and columns
~~2~~	Uploading to AWS fails and returns a class function error	1	Re-imported class packages and rebuild class path from source

Component H - Vote Song (Get Vote)

Incremental Testing

Defect #	Description	Severity	How to correct

1	Multiple Get Vote calls return duplicate song matchups (already voted on)	1	Error in sequence of operations of Posting Vote, counted the vote but didn’t log the user.
2	Multiple Get Vote calls for a round don’t return all possible matchups.	1	Make sure SQL query results for vote logs are transformed correctly into the round’s ‘positions’

Regression Testing

Defect #	Description	Severity	How to correct
1	Matchups are not returned in a seemingly random fashion.	3	Diversify the random number generator more.
1	Matchups return songs from round positions that shouldn’t be in the matchup selected.	1	Correct the position calculator to ensure SQL query is looking for the right positions.

Component I - History of Brackets

Incremental Testing

Defect #	Description	Severity	How to correct
1	User should be able to view historical data for past brackets	1	Create another database table that contains past bracket data, which will be able to be queried by week.

Regression Testing

Defect #	Description	Severity	How to correct
1	Should not be able to query for past bracket data that doesn’t exist	1	Post an error to the user to signify that the chosen historical bracket does not exist
1	Querying for historical data should display winners of matchups and advancement, not just the songs for that week	2	Keep a \`round\` and \`position\` column inside our bracket data tables to distinguish where each song belongs in the bracket.

Component J - Bracket Statistics

Incremental Testing

Defect #	Description	Severity	How to correct
1	Recommendation numbers for songs in the bracket were not saved after the recommendation stage.	2	Added column to table holding bracket history.

Regression Testing

Defect #	Description	Severity	How to correct
1	If data connection is disrupted, display graphs should not attempt to load	2	Remove graph structures and replace typical page data with a notification of poor connection.