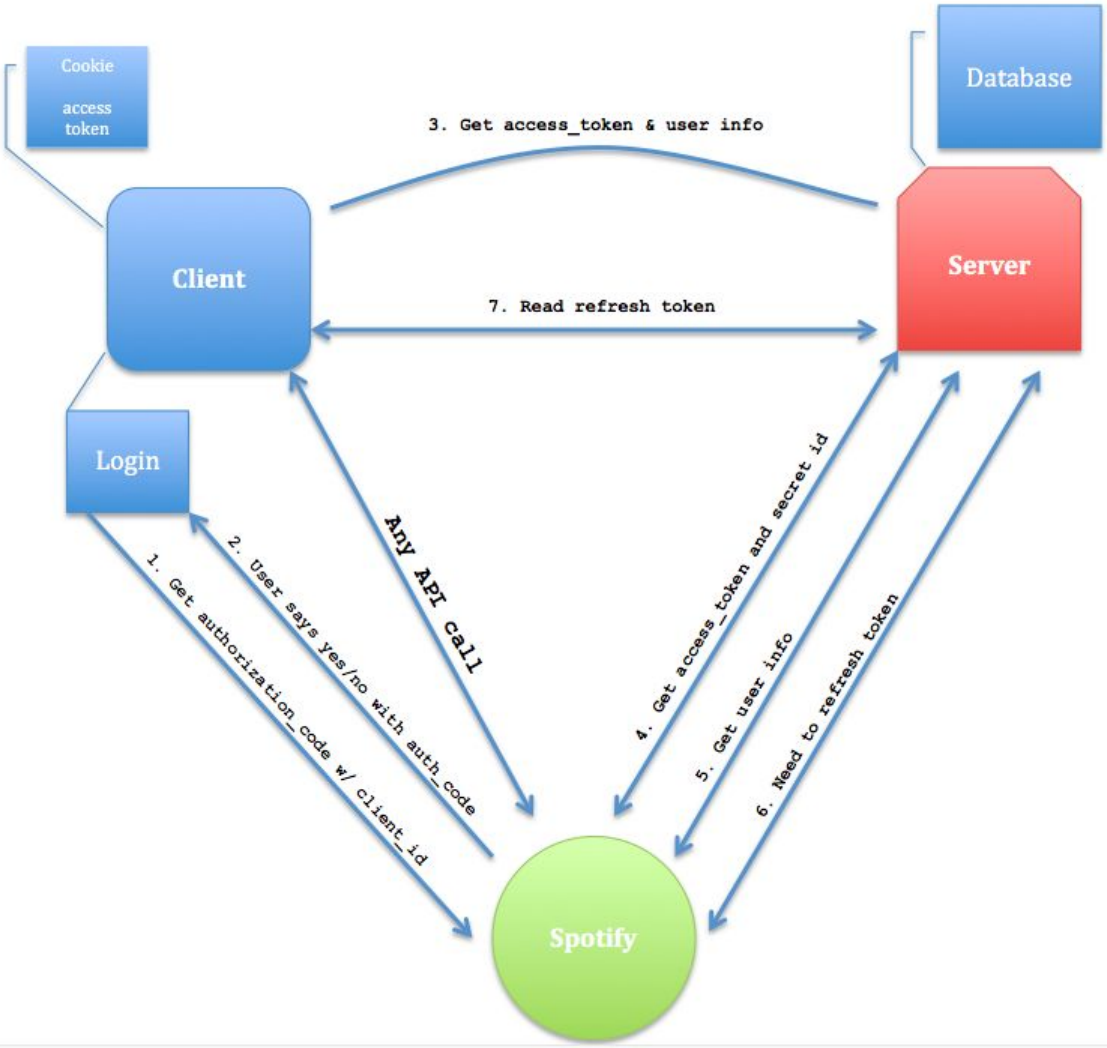
CS 408
Team #17

Ben Maxfield
Terry Lam
Christian Lock
Tom Fanella
Austin Miller

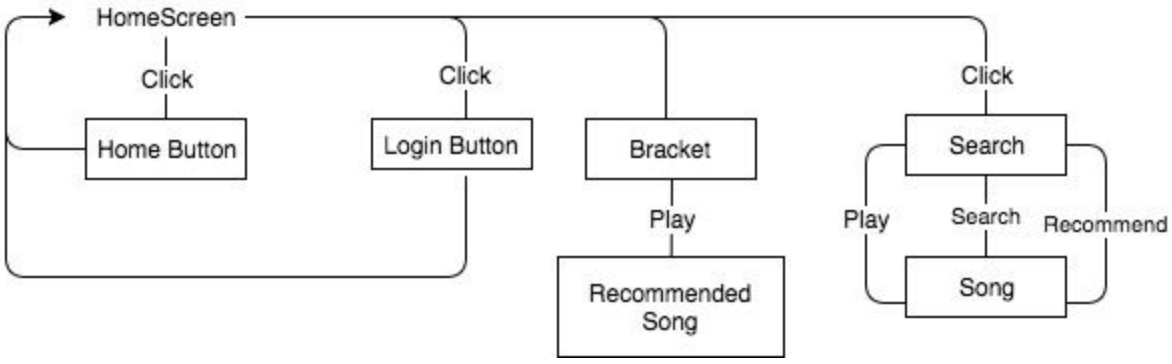
Incremental and Regression Testing Log

Classification:

Back-End Diagram



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 A	Display
Component B	Login
Component C	Search
Component D	Bracket
Component E	Recommend Song

Incremental Testing

Defect #	Description	Severity	How to correct

Regression Testing

Module		Component (ex. Authorization or search display)	
Defect #	Description	Severity	How to correct

Component A - Display

Incremental Testing

Defect #	Description	Severity	How to correct
1	Sidebar should toggle out when user clicks menu	2	Check state of sidebar visibility and modify behavior of sidebar prop accordingly
2	Link for search should bring user to search screen	1	Check and compare snapshot of expected search screen and throw searchScreenExpected exception
3	Link for homescreen should bring user to homescreen	1	Check and compare snapshot of expected homescreen display and throw homeScreenExpected exception

Regression Testing

Defect #	Description	Severity	How to correct
1	If user is logged in, clicking on homescreen or search buttons will not logout user. The user will remain logged in	1	Store access token inside cookie
2	If user clicks search, it should take them to the search screen and not a break the app	1	Catch searchScreenExpected exception and load the path to the searchscreen on the display
3	If user clicks homescreen, it should take them to the homescreen and not break the app	1	Catch homeScreenExpected exception and load the path to the homescreen on the display

Component B- Login

Incremental Testing

Defect #	Description	Severity	How to correct
1	Application should not crash when clicking login button	1	Create a login button exception
2	CORS should not cause failure on the user/validate API endpoint.	3	Enable CORS on API Gateway.
3	User must allow song wars to access their spotify credentials	2	Throw userNotLoggedIn exception
4	A logger should be set in Lambda function.	1	Default logger is set on Lambda startup.

Regression Testing

Defect #	Description	Severity	How to correct
1	Fixing user permissions given to Component B to access their spotify credentials	2	Catch exception created when user denies access to spotify credentials and print ERROR - please login to spotify
2	CORS should not fail on any response code from user/validate API endpoint.	2	Manually enable CORS on response mappings in API Gateway because automated process did not do it.
3	Content-Type header should always be set on any POST or PUT HTTP request.	1	Default Content-Type header is set in the makeHttpsRequest function

Component C - Search

Incremental Testing

Defect #	Description	Severity	How to correct
1	User searches for song with nontraditional ascii characters (~, %, &, #, @)	2	Show “no results found”
2	Null values returned when search query is empty	1	Throw noValueEntered exception

Regression Testing

Defect #	Description	Severity	How to correct
1	Return an error if search is not working due to user authentication	1	Catch userNotLoggedIn exception from Component B - defect 1 and prompt user to login to spotify

Component D - Bracket

Incremental Testing

Defect #	Description	Severity	How to correct
1	User should have authorization to view bracket data	2	Use authorization header to validate user’s authorization code
2	The response should contain	3	Map song_ids to a list of

	every piece of relevant data mapped to each song_id.		objects rather than compressing into a single string or object.
3	Current bracket data should be easily migratable and retrievable as historical data	2	Create separate database tables to distinguish current bracket data and historical bracket data.

Regression Testing

Defect #	Description	Severity	How to correct
1	Calls to retrieve bracket data returned a null response	3	Updated AWS environmental variables with correct JDBC driver
2	Response returns malformed output	1	Validate each piece of data that's mapped from song_id matches and is cast to the correct type from database table

Component E - Recommend Song

Incremental Testing

Defect #	Description	Severity	How to correct
1	Song popularity could be artificially boosted in api call to be above or below Spotify limit.	1	Ensure caps on popularity input before submitting to RDS.

Regression Testing

Defect #	Description	Severity	How to correct
1	Arbitrary cookie ID swapped for access token in new design. Expected cookie ID in code.	1	Change cookie ID with access token. Remove all extra occurrences of cookie ID.
2	Refresh token call no longer expected to be called in new design.	1	Remove refresh token API call from Lambda function.