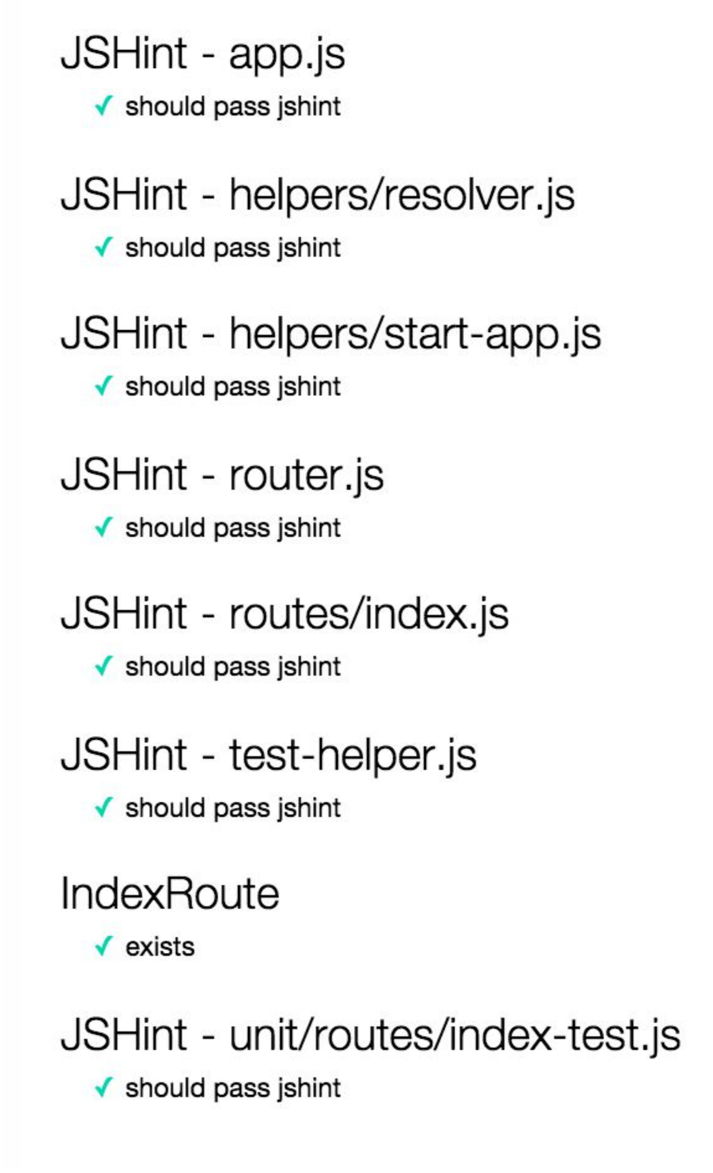
CSCI 3308 Project Part 3

Who: Michael Guida, Nicole Woytarowicz, Ben Williams, and Kylie Dale
Title: GitSound
Vision: "Our vision is to allow all music listeners to version control their playlists."
Automated Tests: https://mochajs.org/, http://chaijs.com/, and http://www.ember-cli.com/#testing-using-the-cli



```
JSHint - app.js
  ✓ should pass jshint

JSHint - helpers/resolver.js
  ✓ should pass jshint

JSHint - helpers/start-app.js
  ✓ should pass jshint

JSHint - router.js
  ✓ should pass jshint

JSHint - routes/index.js
  ✓ should pass jshint

JSHint - test-helper.js
  ✓ should pass jshint

IndexRoute
  ✓ exists

JSHint - unit/routes/index-test.js
  ✓ should pass jshint
```

The automated user testing tests the front end of our application. We have a couple of functions that we run tests against whenever we make changes to GitSound. Specifically, our automated tests check whether pages on the website are up and whether the code for a page passes JSHint (a static analysis tool for JavaScript).

User Acceptance Testing:

Use Case ID:	UC-01
Use Case Name:	Show Remote
Description:	User can view all their remote Spotify playlists with GitSound

Users:	Spotify users		
Pre-conditions:	User is logged in to GitSound with Spotify ID. User has playlists on Spotify. Command line interface needs to recognize “show remote” command.		
Post-conditions:	User can see all their Spotify playlists.		
Frequency of Use:	Whenever someone would like to see their current Spotify playlists		
Flow of Events:	**Actor Action**	**System Response**	**Comments**
	1. Open terminal window		
	2. Login to GitSound with Spotify ID.	Login successful message printed to terminal.	
	3. Type “python3 cli.py show remote”.	All Spotify playlist IDs displayed.	
Test Pass?:	Pass / Fail		
Notes and Issues:			

Use Case ID:	UC-02

Use Case Name:	Add track to playlist
Description:	Users can add a track to a playlist through GitSound.

Users:	Spotify users		
Pre-conditions:	User is logged in with Spotify ID. Current branch is desired playlist. The track ID is known and not already added to the playlist. CLI needs to recognize the “add” command.		
Post-conditions:	Track is added to playlist, and user gets feedback that addition is successful.		
Frequency of Use:	Whenever addition of song is needed		
Flow of Events:	**Actor Action**	**System Response**	**Comments**
	1. Open terminal window		
	2. Login to GitSound with Spotify ID	Login successful message printed to terminal.	
	3. Select playlist to be current branch.	Active branch is shown in green text.	
	4. Type “python3 cli.py add *tracknumber*”	Confirmation message is printed to terminal to say song addition was successful.	
	5. Type “git status” to see if index.txt has been modified	index.txt verifies that the new song has been added to the current playlist.	
Test Pass?:	Pass / Fail		
Notes and Issues:			

Use Case ID:	UC-03

Use Case Name:	Remove Track
Description:	User can remove a song from the current playlist

Users:	Spotify users		
Pre-conditions:	User is logged in with Spotify ID. Current branch is desired playlist. The track ID is known and song is on the playlist. CLI needs to recognize the “remove” command.		
Post-conditions:	Track is removed from playlist, and user gets feedback that removal is successful.		
Frequency of Use:	Whenever someone would like to remove a song from a playlist		
Flow of Events:	**Actor Action**	**System Response**	**Comments**
	1. Open terminal window		
	2. Login to GitSound with Spotify ID	Login successful message displayed.	
	3. Select playlist to be current branch.	Active branch is shown in green text.	
	4. Type “python3 cli.py remove *tracknumber*”	Confirmation message is printed to terminal to say song removal was successful.	
	5. Type “git status” to see if index.txt has been modified	index.txt verifies that the song has been removed from the current playlist.	
Test Pass?:	Pass / Fail		
Notes and Issues:			

VCS: GitHub (https://github.com/GitSound/GitSound)