**LIRI Bot**

**Overview**

In this assignment, you will make LIRI. LIRI is like iPhone's SIRI. However, while SIRI is a Speech Interpretation and Recognition Interface, LIRI is a *Language* Interpretation and Recognition Interface. LIRI will be a command line node app that takes in parameters and gives you back data.

**Before You Begin**

1. LIRI will search Spotify for songs, Bands in Town for concerts, and OMDB for movies.
2. Make a new GitHub repository called liri-node-app and clone it to your computer.
3. To retrieve the data that will power this app, you'll need to send requests to the Bands in Town, Spotify and OMDB APIs. You'll find these Node packages crucial for your assignment.dd
   * [Node-Spotify-API](https://www.npmjs.com/package/node-spotify-api)
   * [Request](https://www.npmjs.com/package/request)
     + You'll use Request to grab data from the [OMDB API](http://www.omdbapi.com/) and the [Bands In Town API](http://www.artists.bandsintown.com/bandsintown-api)
   * [Moment](https://www.npmjs.com/package/moment)
   * [DotEnv](https://www.npmjs.com/package/dotenv)

**Submission Guide**

Make sure you use the normal GitHub. Because this is a CLI App, there will be no need to deploy it to Heroku. This time, though, you need to include screenshots, a gif, and/or a video showing us that you got the app working with no bugs. You can include these screenshots or a link to a video in a README.md file.

* Include screenshots (or a video) of typical user flows through your application (for the customer and if relevant the manager/supervisor). This includes views of the prompts and the responses after their selection (for the different selection options).
* Include any other screenshots you deem necessary to help someone who has never been introduced to your application understand the purpose and function of it. This is how you will communicate to potential employers/other developers in the future what you built and why, and to show how it works.
* Because screenshots (and well-written READMEs) are extremely important in the context of GitHub, this will be part of the grading.

If you haven't written a markdown file yet, [click herqe for a rundown](https://guides.github.com/features/mastering-markdown/), or just take a look at the raw file of these instructions.

**Submission on BCS**

* Please submit the link to the Github Repository!

**Instructions**

1. Navigate to the root of your project and run npm init -y — this will initialize a package.json file for your project. The package.json file is required for installing third party npm packages and saving their version numbers. If you fail to initialize a package.json file, it will be troublesome, and at times almost impossible for anyone else to run your code after cloning your project.
2. Make a .gitignore file and add the following lines to it. This will tell git not to track these files, and thus they won't be committed to Github.

node\_modules  
.DS\_Store  
.env

1. Make a JavaScript file named keys.js.

* Inside keys.js your file will look like this:

console.log('this is loaded');  
  
exports.spotify = {  
  id: process.env.SPOTIFY\_ID,  
  secret: process.env.SPOTIFY\_SECRET  
};

1. Next, create a file named .env, add the following to it, replacing the values with your API keys (no quotes) once you have them:

# Spotify API keys  
  
SPOTIFY\_ID=your-spotify-id  
SPOTIFY\_SECRET=your-spotify-secret

* This file will be used by the dotenv package to set what are known as environment variables to the global process.env object in node. These are values that are meant to be specific to the computer that node is running on, and since we are gitignoring this file, they won't be pushed to github — keeping our API key information private.
* If someone wanted to clone your app from github and run it themselves, they would need to supply their own .envfile for it to work.

1. Make a file called random.txt.
   * Inside of random.txt put the following in with no extra characters or white space:
     + spotify-this-song,"I Want it That Way"
2. Make a JavaScript file named liri.js.
3. At the top of the liri.js file, add code to read and set any environment variables with the dotenv package:

require("dotenv").config();

1. Add the code required to import the keys.js file and store it in a variable.

* You should then be able to access your keys information like so
* var spotify = new Spotify(keys.spotify);

1. Make it so liri.js can take in one of the following commands:
   * concert-this
   * spotify-this-song
   * movie-this
   * do-what-it-says

**What Each Command Should Do**

1. node liri.js concert-this <artist/band name here>
   * This will search the Bands in Town Artist Events API ("https://rest.bandsintown.com/artists/" + artist + "/events?app\_id=codingbootcamp") for an artist and render the following information about each event to the terminal:
     + Name of the venue
     + Venue location
     + Date of the Event (use moment to format this as "MM/DD/YYYY")
2. node liri.js spotify-this-song '<song name here>'
   * This will show the following information about the song in your terminal/bash window
     + Artist(s)
     + The song's name
     + A preview link of the song from Spotify
     + The album that the song is from
   * If no song is provided then your program will default to "The Sign" by Ace of Base.
   * You will utilize the [node-spotify-api](https://www.npmjs.com/package/node-spotify-api) package in order to retrieve song information from the Spotify API.
   * The Spotify API requires you sign up as a developer to generate the necessary credentials. You can follow these steps in order to generate a client id and client secret:
   * Step One: Visit <https://developer.spotify.com/my-applications/#!/>
   * Step Two: Either login to your existing Spotify account or create a new one (a free account is fine) and log in.
   * Step Three: Once logged in, navigate to <https://developer.spotify.com/my-applications/#!/applications/create> to register a new application to be used with the Spotify API. You can fill in whatever you'd like for these fields. When finished, click the "complete" button.
   * Step Four: On the next screen, scroll down to where you see your client id and client secret. Copy these values down somewhere, you'll need them to use the Spotify API and the [node-spotify-api package](https://www.npmjs.com/package/node-spotify-api).
3. node liri.js movie-this '<movie name here>'
   * This will output the following information to your terminal/bash window:
   * \* Title of the movie.  
       \* Year the movie came out.  
       \* IMDB Rating of the movie.  
       \* Rotten Tomatoes Rating of the movie.  
       \* Country where the movie was produced.  
       \* Language of the movie.  
       \* Plot of the movie.  
       \* Actors in the movie.
   * If the user doesn't type a movie in, the program will output data for the movie 'Mr. Nobody.'
     + If you haven't watched "Mr. Nobody," then you should: <http://www.imdb.com/title/tt0485947/>
     + It's on Netflix!
   * You'll use the request package to retrieve data from the OMDB API. Like all of the in-class activities, the OMDB API requires an API key. You may use trilogy.
4. node liri.js do-what-it-says
   * Using the fs Node package, LIRI will take the text inside of random.txt and then use it to call one of LIRI's commands.
     + It should run spotify-this-song for "I Want it That Way," as follows the text in random.txt.
     + Edit the text in random.txt to test out the feature for movie-this and my-tweets

**BONUS**

* In addition to logging the data to your terminal/bash window, output the data to a .txt file called log.txt.
* Make sure you append each command you run to the log.txt file.
* Do not overwrite your file each time you run a command.

**Twitter BONUS**

1. Get your Twitter API keys by following these steps:

* Step One: Visit <https://apps.twitter.com/app/new>
* Step Two: Fill out the form with dummy data. Type http://google.com in the Website input. Don't fill out the Callback URL input. Then submit the form.
* Step Three: On the next screen, click the Keys and Access Tokens tab to get your consume key and secret.
  + Copy and paste them into your .env file, replacing the your-twitter-consumer-key and your-twitter-consumer-secret placeholders.
* Step Four: At the bottom of the page, click the Create my access token button to get your access token key and secret.
  + Copy the access token key and secret displayed at the bottom of the next screen. Paste them into your .env file, replacing the placeholders for your-twitter-access-token-key and your-twitter-access-token-secret.

1. To retrieve the data that will power the twitter portion, you'll need this crucial Node package.

* [Twitter](https://www.npmjs.com/package/twitter)

1. Inside your keys.js file add:

exports.twitter = {  
  consumer\_key: process.env.TWITTER\_CONSUMER\_KEY,  
  consumer\_secret: process.env.TWITTER\_CONSUMER\_SECRET,  
  access\_token\_key: process.env.TWITTER\_ACCESS\_TOKEN\_KEY,  
  access\_token\_secret: process.env.TWITTER\_ACCESS\_TOKEN\_SECRET  
};

1. Inside your .env file, add the following to it, replacing the values with your API keys (no quotes) once you have them:

# Twitter API keys  
  
TWITTER\_CONSUMER\_KEY=your-twitter-consumer-key  
TWITTER\_CONSUMER\_SECRET=your-twitter-consumer-secret  
TWITTER\_ACCESS\_TOKEN\_KEY=your-access-token-key  
TWITTER\_ACCESS\_TOKEN\_SECRET=your-twitter-access-token-secret

* To access your twitter key information from the keys.js file add in the line below to your liri.js file.

var twitter = new Twitter(keys.twitter);

1. node liri my-tweets or node liri my-tweets <twitter handle here>
   * This command will show the last 10 tweets from a twitter account and when they were created at in your terminal/bash window.
   * You can either hard code in a twitter handle or make it so that we can add a handle of our choice to the command.

**Reminder: Submission on BCS**

* Please submit the link to the Github Repository!

**Minimum Requirements**

Attempt to complete homework assignment as described in instructions. If unable to complete certain portions, please pseudocode these portions to describe what remains to be completed. Adding a README.md as well as adding this homework to your portfolio are required as well and more information can be found below.

**Create a README.md**

Add a README.md to your repository describing the project. Here are some resources for creating your README.md. Here are some resources to help you along the way:

* [About READMEs](https://help.github.com/articles/about-readmes/)
* [Mastering Markdown](https://guides.github.com/features/mastering-markdown/)

**Add To Your Portfolio**

After completing the homework please add the piece to your portfolio. Make sure to add a link to your updated portfolio in the comments section of your homework so the TAs can easily ensure you completed this step when they are grading the assignment. To receive an 'A' on any assignment, you must link to it from your portfolio.

**One More Thing**

If you have any questions about this project or the material we have covered, please post them in the community channels in slack so that your fellow developers can help you! If you're still having trouble, you can come to office hours for assistance from your instructor and TAs.

Good Luck!