How to test

- start a local server in some way (in my case I used xampp)
- start a mySQL server in some way (used xampp)
- you need to dirct to http://localhost:port/project1/ASE-230-Project-1/code/
- then the index should load and you can mess with it

Get Artist by ID explanation

What it does

- The way it works is taking in a ID from the get request url and then quering a mySQL database for a ID that matches that
- You would use this for
    - searching for a artist
    - directing someone to a specific artist
    - more specific search than by name

Get Artist by ID examples

Example Usage

- in the html testing you would input the artist id in a form and below would show if it was successful or no row was found
- so you would input something like 2312537

Returns this

- if successful: {
  "success": true,
  "name": "Eve"
  }
- if failure: "error no row found"

Get Song by ID explanation

What it does

- The way it works is taking in a ID from the get request url for songs and then quering a mySQL database for a ID that matches that
- You would use this for
    - searching for a song
    - directing someone to a specific song
    - more specific search than by name or by any other attribue
    - see if a song exits in a database

Get Song by ID examples

Example Usage

- in the html testing you would input the song id in a form and below would show if it was successful or no row was found
- so you would input something like 205638407

Returns this

- {
  "success": true,
  "name": "dragon force"
  }
- if failure: "error no row found"

Get User by ID explanation

What it does

- The way it works is taking in a ID from the get request url for user and then quering a mySQL database for a ID that matches it and returns the row
- You would use this for
  - searching for a user
  - directing someone to a specific user
  - more specific search
  - to check if a user exists in the DB by that id

Get User by ID examples

Example Usage

- in the html testing you would input the user id in a form and below would show if it was successful or no row was found
- so you would input something like 644939995

Returns this

- if successful: {
  "success": true,
  "name": "newUser"
  }
    - of course it doesn't have to look like this since it responds with the row
- if failure: "error no row found"

Get playlist by ID explanation

What it does

- The way it works is taking in a ID from the get request url and then quering a mySQL database for a ID that matches that
- You would use this for
  - searching for a playlist
  - want to find playlist associated with a user and vise versa
  - you can also use it to find songs that are matched with this playlist by using it in conjuction with somehting like a seach songs with playlist id api

Get playlist by ID examples

Example Usage

- in the html testing you would input the playlist id in a form and below would show if it was successful or no row was found
- so you would input something like 1900738092

Returns this

- if successful: {
  "success": true,
  "associatedUSer": 644939995
  }
- if failure: "error no row found"

Get Song by Name explanation

What it does

- The way it works is taking in a string from the get request url for search and then quering a mySQL database for a ID that matches that
- so something like search/string
- You would use this for
  - getting all songs by that name
  - you could use this to seach for songs with close names by using a algorithm to generlize the name

Get Song by Name examples

Example Usage

- in the html testing you would input the artist id in a form and below would show if it was successful or no row was found
- so you would input something like dramatergy

Returns this

- if successful: {
  "Song_ID": 23044146,
  "Artist_ID": 0,
  "Name": "eve",
  "Length_in_sec": 160
  }
- if failure: "error no row found"

Post new user explanation

What it does

- Works by
  - inputing string for username and string for a password
  - making a POST request for users like POST user/
  - adding a row to the sql database with the data
- You would use this for
  - creating a new user for a website

Post new user examples

Example Usage

- sending a POST request like {
  "username": someName,
  "hashed_password": hashedPassword
  }

Returns this

- if successful: {
  "success": true,
  }
- if failure: {
  "success": false,
  }

login explanation

What it does

- Works by
  - Taking in a username and password
  - Making a post request to the /login api
  - if successful make a bearer token and give it to the user in some way
- You would use this for
  - logging in a user

login examples

Example Usage

- you would make a POST request to /login with something lie
  {
  "username": username,
  "password": password
  }

Returns this

- if successful: {
  "success": true,
  "token": bearer token}
- if failure: {
  "success": false,
  "message": "could not log user in"}

create new song explanation

What it does

- Works by
  - taking in from a POST request the name of the song and artist, the length of the song in seconds, and the token from thr user
  - verifying the user
  - updating the sql database
- You would use this for
  - creating a new song in the database
  - could be used by users and by the developers to fill the database

create new song examples

Example Usage

- sending a POST request to /song with something like {
  songName: "name",
  songArtist: "artist name",
  songLength: 160,
  songAuth: token}

Returns this

- if successful: {
  "success": true,
  "songID": 47982357
  }
- if failure: {
  "success": false,

create new playlist explanation

What it does

- Works by
  - taking in from a POST request the user id and auth token
  - verifying the user
  - updating the sql database
- You would use this for
  - creating a new playlist in the database
  - could be used by users and by the developers to fill the database

create new playlist examples

Example Usage

- sending a POST request to /playlist with something like {
  userID: 4298423,
  playlistAuth: token}

Returns this

- if successful: {
  "success": true,
  "plalistID": 572935
  }
- if failure: {
  "success": false,
  }

add song to playlist explanation

What it does

- Works by
  - taking the POST request and getting a playlist ID, song ID, and a token
  - verify the token
  - update the DB
- You would use this for
  - creating a link between a playlist and a song
  - seeing what songs are in a playlist
  - seeing what playlists or how many playlists a song is in

add song to playlist examples

Example Usage

- sending a POST request to /playlist with something like {
  playlistID: 4298423,
  songID: 936195,
  playlistAuth: token}

Returns this

- if successful: {
  "success": true,
  "message": "Added song by the ID songID to the playlist with the ID playlistID"
  }
- if failure: {
  "success": false,
  }