**Classes:**

Artist:

Band Name

Email

Password

Twitter url

Insta url

Fb url

Questions

Question

Asker Username

Answer

User:

Name

Username

Email

Password

Radio id

Liked

Song name

Disliked

Song name

Songs:

Song name

Artist name

Song url

Genre

Radios:

# Radio id

User:

Genres

Genre name

Songs played

Song name

time played

**Endpoints:**

/users/create

Needs:

Name

Username

Email

Password

Returns:

Successful or not

/users/login

Needs:

Email

Password

Returns:

Successful or not

/artists/create

Needs:

Band Name

Email

Password

Twitter url

Insta url

Fb url

Returns:

Successful or not

/artists/login

Needs:

Email

Password

Returns:

Successful or not

/artists/add\_song

/songs/get\_next\_song

Needs:

Radio id

User id

Returns:

Song id

/songs/like

Needs:

Song\_id

User\_id

Returns:

Successful or not

/songs/dislike

Needs:

Song\_id

User\_id

Returns:

Successful or not

Songs:

Songs1

# @app.route('/users/new\_radio', methods=['POST'])

# def create\_radio():

# json = request.json

# user\_id = json["user\_id"] # or could use username

# genre\_names = json["genre\_names"]

# genres = genre\_names.split(', ') # how to give genres "Pop, Rock, Hip Hop"

# potential\_songs = []

# potential\_songs\_weights = []

# # Get all potential songs from given genres

# for genre in genres:

# snapshot = SONGS.order\_by\_child('genre').equal\_to(genre).get()

# for key, val in snapshot.items():

# potential\_songs.append(val['song\_name'])

# potential\_songs\_weightst.append(1)

# # Get all the songs the user liked

# snapshot = USERS.order\_by\_child('user\_id').equal\_to(user\_id).child('liked\_songs').get()

# for key, val in snapshot.items():

# liked\_songs.append(val['song\_name'])

# # Get all the songs liked similar to the songs the user liked

# for song in liked\_songs:

# similar\_songs = []

# snapshot = SONGS.order\_by\_child('song\_name').equal\_to(song).child('similar\_songs').get()

# for key,val in snapshot.items():

# similar\_songs.append(val['song\_name'])

# for similar\_song in similar\_songs:

# try:

# pass

# except Exception as e:

# raise

# else:

# pass

# finally:

# pass

# @app.route('/play', methods=['POST'])

# def get\_song\_url():

# json = req.json

# song\_id = req["song\_id"]

# snapshot = SONGS.order\_by\_child('song\_id').equal\_to(song\_id).get()

# if not snapshot:

# return jsonify({'message': 'Song not found'}), 400

# for key, val in snapshot.items():

# song\_url = val['song\_url']

# return jsonify({'song\_url': song\_url}), 201