README.md

CPSC 449 - Fall 2019 - Project 3

Names

Oscar Olazabal
Shivam Shah
Kevin Chen

Dependencies

This project depends on Foreman and all of the Python modules listed in `requirements.txt`. To install foreman and the modules and run:

```
sudo apt install ruby-foreman
pip3 install -r requirements.txt
```

You will also need to install Kong and MinIO.

Follow Step 1 of the Ubuntu Installation instructions to install Kong on Tuffix using the .deb package for Ubuntu 18.04 Bionic. Then use the following commands to install PostgreSQL and create a database and configuration file for Kong:

```
sudo apt install --yes postgresql
sudo -u postgres psql -c "CREATE USER kong WITH ENCRYPTED PASSWORD 'kong'"
sudo -u postgres psql -c 'CREATE DATABASE kong OWNER kong'
sudo cp /etc/kong/kong.conf.default /etc/kong/kong.conf
```

Now edit /etc/kong/kong.conf to ucomment pg_password and change it to kong, then use the following commands to start Kong:

```
sudo kong migrations bootstrap
ulimit -n 4096 && sudo kong start
```

Configure Kong by running

```
sh kong.conf
```

You can then proceed with the curl command in Step 4 of the Ubuntu Installation instructions to verify that Kong is running.

For Minio, create a new directory to store the data for the server, and follow the directions on their website to start the MinIO server. Be sure to start the server with the command `./minio server buckets/`

When the MinIO server starts, it will print an endpoint, access key, and secret key. Make a note of these values so that you can access the server later.

Use the information provided to log into the MinIO Browser and create a bucket named tracks. Edit the policy for this bucket to add the prefix * as Read Only. Upload some music files and verify that you can access them via URLs such as http://localhost:9000/tracks/music-file.mp3

Also install docker

```
sudo apt update
sudo apt install --yes docker.io
sudo usermod -aG docker $USER
```

And the memcache tools

```
sudo apt install --yes memcached libmemcached-tools
```

If you run into dependency errors, ensure all of the modules in `requirements.txt` are install properly. You may have to use `apt install` to install some.

Running the Application

To run the application, you have to initialize the database and run each service on their own port. To do so, run the following:

```
./start
```

Testing the Application

To test the application you can use the provided `test` executable.

```
./test
```

The executable creates, retrieves, updates, and deletes from the database using the endpoints of the application

Flask Output

```
student@tuffix-vm:~/Desktop/csuf-449-projects/project2$ ./test

INSERTING NEW USER
17:06:57 users.2        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "POST /users HTTP/1.1" 201 -
{'disp_name': 'iamuser1',
 'email': 'user1@gmail.com',
 'url_homepage': 'user1.com',
 'user_pass':
'pbkdf2:sha256:150000$WxZvh3dk$86f6e34d8c1ceab65d388ad7e6e59713589d0589785b5d7786aa75accd230774',
 'username': 'username1',
 'uuid': '4c8c604d-9887-4536-bccb-156ed1d06d3e'}

INSERTING NEW USER
17:06:57 users.2        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "POST /users HTTP/1.1" 201 -
{'disp_name': 'iamuser2',
 'email': 'user2@gmail.com',
 'url_homepage': 'user2.com',
 'user_pass':
'pbkdf2:sha256:150000$bqQYh4IQ$9d93207726ea3e5a722e26e781a3d037bbd2faa481d47906c7c1a9757cb3c69c',
 'username': 'username2',
 'uuid': '1e0fa74a-a54a-4037-9b56-6ce5370b92d0'}

INSERTING NEW USER
17:06:57 users.3        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "POST /users HTTP/1.1" 201 -
{'disp_name': 'iamuser3',
 'email': 'user3@gmail.com',
 'url_homepage': 'user3.com',
 'user_pass':
'pbkdf2:sha256:150000$UhA4YyrP$c0b3d1835d1a77d6cf92cf63a543fedbed17dc7f9d86b326eb3299510aa3fc5c',
 'username': 'username3',
 'uuid': '40000803-61a0-4860-bc21-68e5ddc9b9a9'}

GET USER
```

```
17:06:57 users.2        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "GET /users/username1 HTTP/1.1" 200 -
['username1', 'iamuser1', 'user1@gmail.com', 'user1.com']

GET USER
17:06:57 users.2        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "GET /users/username2 HTTP/1.1" 200 -
['username2', 'iamuser2', 'user2@gmail.com', 'user2.com']

GET USER
17:06:57 users.3        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "GET /users/username3 HTTP/1.1" 200 -
['username3', 'iamuser3', 'user3@gmail.com', 'user3.com']

DELETE USER
17:06:57 users.2        | 127.0.0.1 - - [18/Dec/2019 17:06:57] "DELETE /users/username3 HTTP/1.1" 200 -
{'message': 'Deleted 1 user with username username3'}

UPDATE USER PASSWORD
17:06:58 users.3        | 127.0.0.1 - - [18/Dec/2019 17:06:58] "PATCH /users/username1 HTTP/1.1" 200 -
['username1', 'iamuser1', 'user1@gmail.com', 'user1.com']

AUTHENTICATE USER
17:06:58 users.3        | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /users/authenticate HTTP/1.1" 200 -
{'result': True}


NEW TRACK
17:06:58 tracks.2       | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /tracks HTTP/1.1" 201 -
{'album_art_url': 'https://picsum.photos/id/975/200/200',
 'album_title': 'rock',
 'artist': 'Flea',
 'id': '6138f2b2-19df-4b6c-b0d2-c60700c9b078',
 'media_file_url': 'http://localhost:8000/media/bass.mp3',
 'track_length': '2.1',
 'track_title': 'bass'}

NEW TRACK
17:06:58 tracks.3       | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /tracks HTTP/1.1" 201 -
{'album_art_url': 'https://picsum.photos/id/876/200/200',
 'album_title': 'rock',
 'artist': 'Slash',
 'id': '76b35ec7-cf18-41a0-9225-151d06360f5e',
 'media_file_url': 'http://localhost:8000/media/electric-guitar.mp3',
 'track_length': '2.1',
 'track_title': 'electric guitar'}

NEW TRACK
17:06:58 tracks.2       | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /tracks HTTP/1.1" 201 -
{'album_art_url': 'https://picsum.photos/id/1062/200/200',
 'album_title': 'classy',
 'artist': 'Yoyoma',
 'id': '8df5dde3-598f-422d-91e4-db9ba7f58c95',
 'media_file_url': 'http://localhost:8000/media/harp.mp3',
 'track_length': '2.1',
 'track_title': 'harp'}

UPDATE TRACK
17:06:58 tracks.2       | 127.0.0.1 - - [18/Dec/2019 17:06:58] "PATCH /tracks/8df5dde3-598f-422d-91e4-
db9ba7f58c95 HTTP/1.1" 404 -
{'error': 'Error from server: code=2200 [Invalid query] message="Unknown '
          'identifier track_title"'}

GET TRACK
17:06:58 tracks.3       | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /tracks/8df5dde3-598f-422d-91e4-
db9ba7f58c95 HTTP/1.1" 200 -
[{'album_art_url': 'https://picsum.photos/id/1062/200/200',
  'album_title': 'classy',
  'artist': 'Yoyoma',
  'descriptor': None,
  'media_file_url': 'http://localhost:8000/media/harp.mp3',
  'title': 'harp',
  'track_description': None,
  'track_length': '2.1',
```

```
  'uuid': '8df5dde3-598f-422d-91e4-db9ba7f58c95'}]

DELETE TRACK
17:06:58 tracks.1       | 127.0.0.1 - - [18/Dec/2019 17:06:58] "DELETE /tracks/8df5dde3-598f-422d-91e4-
db9ba7f58c95 HTTP/1.1" 200 -
{'message': 'Deleted 1 track with id 8df5dde3-598f-422d-91e4-db9ba7f58c95'}

NEW DESCRIPTION
17:06:58 descriptions.1 | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /descriptions HTTP/1.1" 201 -
{'description': 'this is a track description',
 'id': '6138f2b2-19df-4b6c-b0d2-c60700c9b078',
 'user_name': 'username1'}

NEW DESCRIPTION
17:06:58 descriptions.2 | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /descriptions HTTP/1.1" 201 -
{'description': 'this is a track description again',
 'id': '76b35ec7-cf18-41a0-9225-151d06360f5e',
 'user_name': 'username2'}

GET DESCRIPTION
17:06:58 descriptions.1 | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /descriptions/6138f2b2-19df-4b6c-
b0d2-c60700c9b078 HTTP/1.1" 200 -
[{'descriptor': 'username1',
  'track_description': 'this is a track description',
  'uuid': '6138f2b2-19df-4b6c-b0d2-c60700c9b078'}]

INSERTING NEW PLAYLIST
17:06:58 playlists.3    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /playlists HTTP/1.1" 201 -
{'creator': 'username1',
 'description': 'this is playlist1',
 'id': '768dc0a2-b3c0-4150-8241-2db9212720ad',
 'title': 'playlist1',
 'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078']}

INSERTING NEW PLAYLIST
17:06:58 playlists.3    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /playlists HTTP/1.1" 201 -
{'creator': 'username2',
 'description': 'this is playlist2',
 'id': '76cb58d7-a429-4f6f-b91d-49f7085b7529',
 'title': 'playlist2',
 'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078',
            '76b35ec7-cf18-41a0-9225-151d06360f5e']}

INSERTING NEW PLAYLIST
17:06:58 playlists.1    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "POST /playlists HTTP/1.1" 201 -
{'creator': 'username2',
 'description': 'this is playlist3 made by username2',
 'id': '8b9c0d4d-86a1-4e68-a194-47e908f1f8e7',
 'title': 'playlist3',
 'tracks': ['76b35ec7-cf18-41a0-9225-151d06360f5e']}

GET ALL PLAYLISTS
17:06:58 playlists.1    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /playlists/all HTTP/1.1" 200 -
[{'creator': 'username2',
  'description': 'this is playlist2',
  'title': 'playlist2',
  'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078',
             '76b35ec7-cf18-41a0-9225-151d06360f5e'],
  'uuid': '76cb58d7-a429-4f6f-b91d-49f7085b7529'},
 {'creator': 'username1',
  'description': 'this is playlist1',
  'title': 'playlist1',
  'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078'],
  'uuid': '768dc0a2-b3c0-4150-8241-2db9212720ad'},
 {'creator': 'username2',
  'description': 'this is playlist3 made by username2',
  'title': 'playlist3',
  'tracks': ['76b35ec7-cf18-41a0-9225-151d06360f5e'],
  'uuid': '8b9c0d4d-86a1-4e68-a194-47e908f1f8e7'}]

GET PLAYLIST
```

```
17:06:58 playlists.1    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /playlists/768dc0a2-b3c0-4150-8241-
2db9212720ad HTTP/1.1" 200 -
[{'creator': 'username1',
  'description': 'this is playlist1',
  'title': 'playlist1',
  'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078'],
  'uuid': '768dc0a2-b3c0-4150-8241-2db9212720ad'}]

GET PLAYLIST
17:06:58 playlists.3    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /playlists/76cb58d7-a429-4f6f-b91d-
49f7085b7529 HTTP/1.1" 200 -
[{'creator': 'username2',
  'description': 'this is playlist2',
  'title': 'playlist2',
  'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078',
             '76b35ec7-cf18-41a0-9225-151d06360f5e'],
  'uuid': '76cb58d7-a429-4f6f-b91d-49f7085b7529'}]

GET PLAYLIST
17:06:58 playlists.2    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /playlists/8b9c0d4d-86a1-4e68-a194-
47e908f1f8e7 HTTP/1.1" 200 -
[{'creator': 'username2',
  'description': 'this is playlist3 made by username2',
  'title': 'playlist3',
  'tracks': ['76b35ec7-cf18-41a0-9225-151d06360f5e'],
  'uuid': '8b9c0d4d-86a1-4e68-a194-47e908f1f8e7'}]

GET PLAYLIST BY CREATOR
17:06:58 playlists.1    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "GET /playlists?creator=username1
HTTP/1.1" 200 -
[{'creator': 'username1',
  'description': 'this is playlist1',
  'title': 'playlist1',
  'tracks': ['6138f2b2-19df-4b6c-b0d2-c60700c9b078'],
  'uuid': '768dc0a2-b3c0-4150-8241-2db9212720ad'}]

DELETE PLAYLIST
17:06:58 playlists.1    | 127.0.0.1 - - [18/Dec/2019 17:06:58] "DELETE /playlists/8b9c0d4d-86a1-4e68-
a194-47e908f1f8e7 HTTP/1.1" 200 -
{'message': 'Deleted 1 playlist with id 8b9c0d4d-86a1-4e68-a194-47e908f1f8e7'}
```

Endpoints

Tracks `PORT 8000`

- `/tracks/<uuid:id>`
  - `GET` Retrieves a track given its UUID
  - `DELETE` Deletes a track given its UUID
  - `PATCH` Updates a track given its UUID and the JSON data of the fields to be modified
- `/tracks`
  - `POST` Inserts a track given the JSON data of a new track

Playlists `PORT 8000`

- `/playlists/<int:id>`
  - `GET` Retrieves a playlist given its ID
  - `DELETE` Deletes a playlist given its ID
- `/playlists`
  - `GET` Retrieves a playlist of a user given the argument `creator=`
  - `POST` Inserts a playlist given the JSON data of a new playlist
- `/playlists/all`

- `GET` Retrieves all playlists

Users `PORT 8000`

- `/users/<username>`
  - `GET` Retrieves user's profile (does not include password) given their username
  - `PATCH` Updates a user's password, given an updated password in JSON format
  - `DELETE` Deletes a user given their username
- `/users`
  - `POST` Inserts a new user given their data in JSON format
- `/users/authenticate`
  - `GET` Validates a user and password given JSON data containing a username and password

Descriptions `PORT 8000`

- `/descriptions/<uuid:id>`
  - `GET` Retrieves user descriptions given the UUID of a track
- `/descriptions`
  - `POST` Inserts a user description of a track given a username, a track UUID, and a description

XSPF `PORT 8000`

- `/playlist/<int:id>.xspf`
  - `GET` Retrieves playlist given its ID and renders a XSPF of it's information. If you have the correct software installed (such as Parole Media Player), you will be prompted to open the playlist and stream the music

Data Structure

Below are sample JSONs of the entities.

Tracks

```
{
    "id" : UUID
    "track_title": "Track Title",
    "album_title": "Album Title",
    "artist": "Artist's Name",
    "track_length": 2.6,
    "media_file_url": "file:///home/student/Music/mediafile1",
    "album_art_url": "file:///home/student/Music/albumart1"
}
```

Playlists

```
{
    "title": "Playlist Title",
    "creator": "Username",
    "tracks" : [
        "file:///home/student/Music/media_file1",
        "file:///home/student/Music/media_file2"
    ],
    "description": "Description of playlist"
}
```

Users

```
{
    "username" : "User name",
    "user_pass" : "Password of user",
    "disp_name" : "Display name",
    "email" : "useremail@email.com",
    "url_homepage" : "http://userhomepage.com"
}
```

Descriptions

```
{
    "track_id": UUID,
    "user_name": "User name",
    "description": "This is a track description"
}
```