# Audio Streaming Platform

## Group Members :

**Aditi Dattatray Survase (as14152)**

**Kewin Venugopal (ksv2013)**

## INTRODUCTION

Audio Streaming Platform can be used to play different songs and podcasts of the user's choice.

## DESCRIPTION

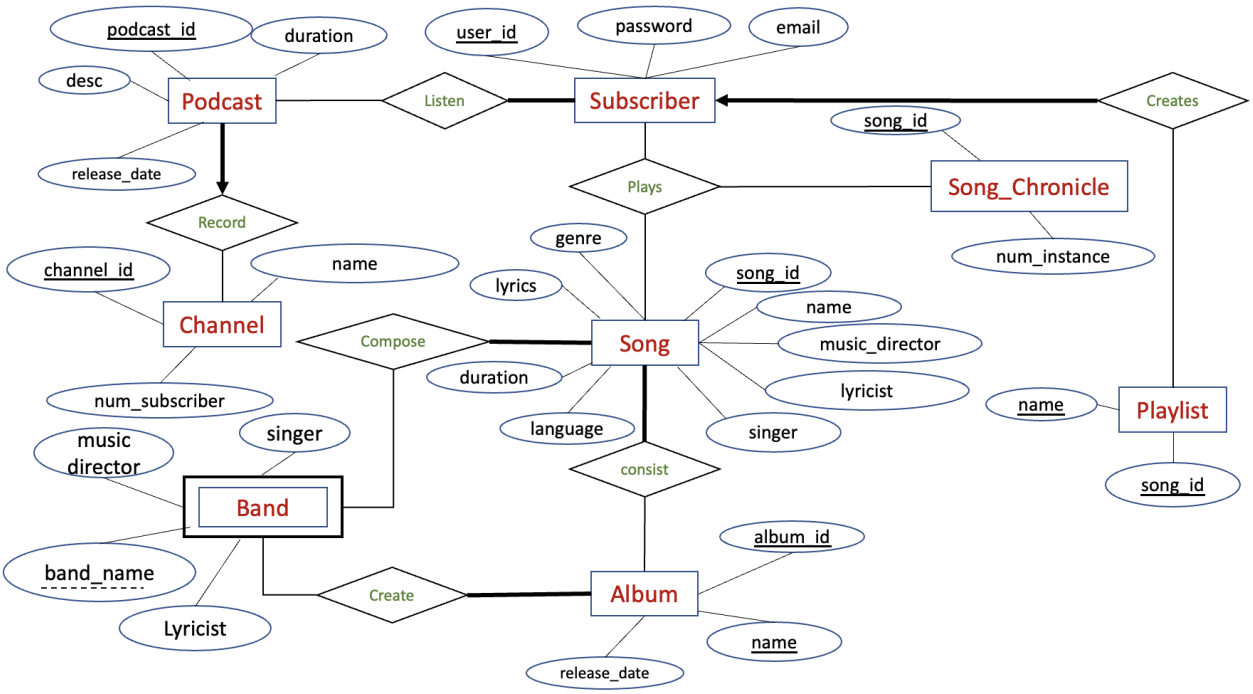
This platform will let the users play the audio files of their choice. It is split into two components, songs and podcasts -

- Songs
  - Users can play songs from different genre
  - Songs are categorized into albums that are composed by different artists
  - Users can add their favorite songs to the playlist
- Podcast
  - User can listen to episodes recorded by different podcast channel
  - They can also subscribe to channels of their choice

## BUSINESS RULES

- Subscribers listen to songs.
- Playlist is a collection of different songs and is created by exactly one subscriber
- Every song has a name, language, lyrics, genre and duration. Each song is uniquely identified by song id. All instances of songs played by subscribers are stored.
- Every song is a part of an album. Each album contains at least one song. An album is uniquely identified by their name and id. Every album is released by a music band.
- Music bands compose at least one song. Each band consists of a music director, singer and lyricists.
- Subscribers can subscribe to podcast channels. Channel is identified by their name and id. Podcast is recorded by exactly one channel. Podcasts have a name , duration and a description.

## E- R DIAGRAM




## DATA ACQUISITION

- Audio Files - Used PySpotify package to download mp4 files and converted to wav using online converter and ffmpeg package in Python
- Images - Google Images, all files are placed un
- Schema tables - generated random data with the help of dataset available in Datasets

## USER INTERACTION

- User can give text input in the search box to search for songs and podcasts available
- They can also listen to their frequently played songs under “Your Top Picks” section
- Users can listen to episodes recorded by podcast channels.
- Users can subscribe to podcast channels and view subscribed channels on the home page.
- They can also play the songs composed by the artist of their choice
- Insight section in the screen will show a quick summary of the user’s favorite genre and the number of instances all songs played by the user

–SCHEMA–

```
DROP TABLE user_subscribe_channel;

DROP TABLE channel_record_podcast;

DROP TABLE user_create_playlist;

DROP TABLE band_create_album;

DROP TABLE song_chronicle;

DROP TABLE album_has_songs;

DROP TABLE album;

DROP TABLE channel;

DROP TABLE subscriber;

DROP TABLE song;
```

```
create table Song(
  song_id serial primary key,
  music_director varchar(32),
  singer varchar(32),
  lyricist varchar(32),
  name varchar(32),
  lyrics text,
  duration float,
  genre varchar(32),
  language varchar(32)
);
create table Subscriber(
  id varchar(32) primary key,
  password varchar(32) not null,
  email varchar(32)
);
create table Album(
 name varchar(128),
 album_id integer primary key,
 release_date date
);
```

```
create table Channel(
  channel_id integer primary key,
  channel_name varchar(128) ,
  num_subscribers integer
);
create table Album_has_songs(
 album_id integer,
 song_id integer,
 primary key(album_id, song_id),
 foreign key(album_id) references Album(album_id),
 foreign key(song_id) references Song(song_id)
);
create table Band_create_Album(
album_id integer,
band_id integer,
album_name varchar(128) not null,
band_name varchar(128) not null,
music_director varchar(128),
singer varchar(128),
lyricist varchar(128),
release_date date,
Primary key (album_id,band_id)
);
```

```
create table Song_chronicle(
  song_id integer,
  id varchar(32),
  num_instance integer,
  primary key(song_id, id),
  foreign key(song_id) references Song(song_id),
  foreign key(id) references Subscriber(id)
);

create table user_create_playlist(
 name varchar(32),
 user_id varchar(32),
 song_id integer not null,
 primary key(name, user_id, song_id),
 foreign key(user_id) references Subscriber(id),
 foreign key(song_id) references Song(song_id)
);
```

```
create table channel_record_podcast(
  podcast_name varchar(32),
  podcast_id integer not null,
  description text,
  release_date date,
  duration float,
  channel_id integer not null,
  primary key(channel_id,podcast_id),
  foreign key(channel_id) references Channel(channel_id)
);
create table User_Subscribe_Channel(
  user_id varchar(32),
  channel_id integer,
  primary key(user_id, channel_id),
  foreign key(channel_id) references Channel(channel_id),
  foreign key(user_id) references Subscriber(id)
);
```