Below is the code sample I wrote for basic audio mixing

mix_backtrack.py

```
/* Get a track from remote database, do a simple offset check, and mix the track*/
from pymongo import MongoClient
from bson.objectid import ObjectId
from pydub import AudioSegment
import os,sys
import mixing
import s3_download
import json
import math
import librosa

def get_first_non_zero(filename):
    audio_data, sr = librosa.load(filename, sr=None, mono=False)
    channel_data = audio_data[0]
    first_non_zero = 0
    for i in range(len(channel_data)):
        if channel_data[i] != 0:
            first_non_zero = float(i) / sr
            break
    return first_non_zero

if __name__ == '__main__':
    track_ids = sys.argv[1].split(',')
    download_path = sys.argv[2]
    output_filename = sys.argv[3]
    if os.getenv('MONGO_URI', 'mongodb://localhost/qa') == 'mongodb://localhost/qa':
        client = MongoClient() #default port : 27017
        db = client.qa
    else:
        client = MongoClient(os.environ['MONGO_URI'])
        db = client.prod

    sounds = []
    for track_id in track_ids:
        sound = {}
        sound['gain'] = 0
        sound['pan'] = 0
        track_db = db.track.find({'_id': ObjectId(track_id)})
        for track_doc in track_db:
            filename = s3_download.download_track(track_doc, download_path)
            if filename:
```

```
            sound['filename'] = filename
            sound['audio'] = AudioSegment.from_file(sound['filename'])

            encoding_offset = 0
            if track_doc.get('first_non_zero', 0) > 0:
               first_non_zero = get_first_non_zero(filename)
               encoding_offset = track_doc.get('first_non_zero', 0) - first_non_zero

            sound['offset'] = (track_doc.get('offset', 0) + track_doc.get('extra_offset', 0) +
encoding_offset) * 1000
            sounds.append(sound)
         break
   mixdown = mixing.mixing_audio(sounds)
   if not isinstance(mixdown, str):
      mixdown.export(os.path.join(download_path, output_filename), format="mp3",
bitrate="128k")
   for sound in sounds:
      os.remove(sound['filename'])
   client.close()
```

mixing.py

```
   /* basic audio mixing */
   from pydub import AudioSegment

   def mixing_audio(sounds):
      # first version of audio mixing
      # parse in a sound object and paramter
      # a list of dictionary with paramters
      # pan, gain, offset
      maxlen = max([len(s['audio']) for s in sounds])
      sound = 'empty'
      for s in sounds:
         audio = s['audio']
         if len(audio) < maxlen:
            audio = audio + AudioSegment.silent(duration=float(maxlen - len(audio)))
         if 'gain' in s:
            audio = audio.apply_gain(s['gain'])
         if 'pan' in s:
            audio = audio.pan(s['pan'])
         if isinstance(sound, str):
            offset = s.get('offset', 0)
            if offset == 0:
               sound = audio
            elif offset > 0:
```

```
            sound = AudioSegment.silent(duration=float(offset)) + audio
        else:
            sound = audio[-(len(audio)+float(offset)):]
    else:
        offset = s.get('offset', 0)
        if offset >= 0:
            sound = sound.overlay(audio, position=offset)
        else:
            sound = sound.overlay(audio[-(len(audio)+float(offset)):])
return sound
```