컴퓨터 네트워크

-12주차(Multimedia Streaming)-

학번: 201404377
이름: 진승언

과제목표

멀티미디어 스트리밍 프로토콜을 이용한 실시간 라디오 녹음 및 저장을 한다. RTMP를 이용해서 EBS 라디오를 녹음하고 HLS를 이용하여 KBS 라디오를 녹음한다. 그리고 crontab을 이용한 주기적으로 실행한다. 이렇게 녹음된 파일들을 웹서버에 올리고 실행 및 다운을 할 수 있게 한다.

과제해결방법

1. RTMP를 이용한 EBS 라디오 녹음 (EBSdownload.py)

```python
import signal
import os
import time
import subprocess
from datetime import datetime
import mutagen
import mutagen.mp3
from mutagen.easyid3 import EasyID3


def __get_current_time(is_month):
    today_time = datetime.today()
    if is_month == 'month':
        return today_time.strftime("%Y%n%d")
    return today_time.strftime("%Y%m%d%H%M%S")

def rtmp():
    file_name = __get_current_time("day") + "FILE_EBS"
    absolute_path = "/home/u201404377/u201404377/network_week12/EBS/" + file_name
    try:
        subprocess.run(("rtmpdump -r rtmp://new_tradio.ebs.co.kr/tradio/tradiolive_m4a -B 15 -o " + absolute_path+".flv"), stdout=subprocess.PIPE, shell=True)
        __flv_to_mp3(absolute_path)
        __change_meta_data(absolute_path, file_name)
    except subprocess.CalledProcessError as sub_exception:
        print("ERROR With RTMP:" , sub_exception)

def __flv_to_mp3(absolute_path):
    try:
        subprocess.run("ffmpeg -i "+ str(absolute_path)+".flv -acodec mp3 " + absolute_path + ".mp3", stdout=subprocess.PIPE, shell = True)
    except subprocess.CalledProcessError as sub_exception:
        print("ERROR With RTMP:" , sub_exception)


def __change_meta_data(file_path, file_name):
    file_path = "/home/u201404377/u201404377/network_week12/EBS/" + file_name + ".mp3"
    try:
        meta = EasyID3(file_path)
    except mutagen.id3.ID3NoHeaderError:
        meta = mutagen.File(file_path, easy = True)
        meta.add_tags()
    meta['title'] = __get_current_time("month") + "_jinSeungEon"
    meta["artist"] = "201404377"
    meta["genre"] =   "201404377_RADIO"
    meta["album"] = "201404377_album"
    meta.save()
    print("META", str(meta))

if __name__ == "__main__":
    #download ebs
    rtmp()
```

RTMP는 Real Time Messaging Protocol의 약자로서 인터넷 상에서 멀티미디어 스트리밍 서비스를 이용하기 위해서 사용하는 프로토콜중 하나이다.

먼저 EBS와 KBS 둘 다 다운 받는 디렉토리를 .../EBS 로 하였다.
rtmp()는 rtmp 프로토콜을 사용하여 EBS 라디오를 다운 받는 코드이다. subprocess를 사용하여 rtmpdump 명령어를 실행하고 15초 길이만큼 .flv 포맷으로 다운받는다.

그 다음 __flv_to_mp3()는 다운 받은 EBS 라디오 .flv 파일을 mp3로 변환해주는 코드이다. 이 또한 subprocess를 사용하여 ffmpeg 명령어를 실행한다.

마지막으로 __change_meta_data()는 mp3 파일의 메타데이터를 변경해주는 코드이다. 내 학번과 이름이 들어가게끔 변경하였다.

2. HLS를 이용한 KBS 라디오 녹음

HLS는 HTTP Live Streaming의 약자로서 http 기반 미디어 스트리밍, apple에서 만든 프로토콜이며 웹서버에서 비디오와 오디오를 전송한다.

```
from datetime import datetime
import time
import mutagen
import mutagen.mp3
from mutagen.easyid3 import EasyID3

def __get_current_time(is_month):
    today_time = datetime.today()
    if is_month == 'month':
        return today_time.strftime("%Y%n%d")
    return today_time.strftime("%Y%m%d%H%M%S")

def __change_meta_data(file_path, file_name):
    file_path = "/home/u201404377/u201404377/network_week12/KBS/" + file_name + ".mp3"
    try:
        meta = EasyID3(file_path)
    except mutagen.id3.ID3NoHeaderError:
        meta = mutagen.File(file_path, easy = True)
        meta.add_tags()
    meta['title'] = __get_current_time("month") + "_jlnSeungEon"
    meta["artist"] = "201404377_KBS"
    meta["genre"] =   "201404377_RADIO"
    meta["album"] = "201404377_album"
    meta.save()
    print("META", str(meta))

def __flv_to_mp3(absolute_path):
    try:
        subprocess.run("ffmpeg -i "+ str(absolute_path)+".flv -acodec mp3 " + absolute_path + ".mp3", stdout=subprocess.PIPE, shell = True)
    except subprocess.CalledProcessError as sub_exception:
        print("ERROR With RTMP:" , sub_exception)

def his():
    file_name = file_name = __get_current_time("day") + "FILE_KBS"
    absolute_path = "/home/u201404377/u201404377/network_week12/EBS/" + file_name
    print("CHECK", absolute_path)
    url_KBS= ""
    #curl_KBS = "ffmpeg -i $(curl -s 'http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList' | grep 'service_url' | cut -d\" -f 16 | cut -d\\ -f
1 | tail -1) -acodec mp3 " + absolute_path + ".mp3"

    curl_KBS= """mplayer $(curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList" | grep service_url | tail -1 | cut -d\\" -f16 | cut -d
\\\\ -f1) -ao pcm:file="""  + absolute_path +  '.flv -vc dummy -vo null'
    process = subprocess.Popen(curl_KBS,shell=True, preexec_fn=os.setsid)
    time.sleep(15)
    os.killpg(os.getpgid(process.pid), signal.SIGTERM)
    __flv_to_mp3(absolute_path)
    __change_meta_data(absolute_path, file_name)

if __name__ == "__main__":
    #download ebs
     his()
                                                                                53,0-1        Bot
```

mplayer는 미디어 플레이어 소프트웨어이다. mplayer $(curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"| grep "service_url" | cut -d\" -f 16 | cut -d\\ -f 1) -ao pcm:file= your/path/file_name.flv -vc dummy -vo null
를 subprocess로 실행한다. (웹 크롤링 + 파싱 + 미디어 URL을 HLS로 요청)

그 후는 EBS에서 한거와 동일하게 .flv를 mp3로 변환하고 메타데이터를 변경해준다.

```
u201404377@ubuntu:~/u201404377/network_week12$ mplayer $( curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList "| grep "service_url" |
cut -d\" -f 16 | cut -d\\ -f 1 ) -ao pcm:file=/home/u201404377/u201404377/network_week12/TEST12.flv -vc dummy -vo null
MPlayer 1.3.0 (Debian), built with gcc-7 (C) 2000-2016 MPlayer Team
do_connect: could not connect to socket
connect: No such file or directory
Failed to open LIRC support. You will not be able to use your remote control.

Playing item.
File not found: 'item'
Failed to open item.


Playing http://1fm.gscdn.kbs.co.kr/1fm_192_1.m3u8?Expires=1577734872&Policy=eyJTdGF0ZW1lbnQiOlt7IlJlc291cmNlIjoiaHR0cDovLzFmbS5nc2Nkbi5rYnMuY28ua3IvMWZtXzESMl8xLm0zdTgiLCJDb25kaXRpb24iOnsiRGF0ZUxlc3NUaGFuIjp7IkFXUzpFcG9jaFRpbWUiOjE1Nzc3MzQ4NzJ9fX1dfQ__&Signature=i87AOrUiMP7Pzzk3aAlXnXSccP09mKdTcsW0wj8Pqf9fMFkW0MO41jx0MWT2YclYcBg~JbvkxLIvSuQ5wkCrUJe1~KlEA8vquaDsIP2piC2M7gg8ldvHFTvltNwvKTZAohkP09ctfdqNnpCoUEAhEsyuaHw7sbXhNldYYGaX7-qviABNe~l52CczjBTT2B5uec9wSxlCL8vmJqrd4I4Pf0nmpEnq7DfpYqVxGqDMu42m~hKUlkOefMPUJFkEephemA0t2jSMG3EwGZmYc9I7Pnxlmdll0Tqemv3JJ7VenJuNzj2IfHU5Pf1xkd9JjrwlsgG4vJLDTalHyqqwortgw__&Key-Pair-Id=APKAICDSGT3Y7IXGJ3TA.
Resolving 1fm.gscdn.kbs.co.kr for AF_INET6...

Couldn't resolve name for AF_INET6: 1fm.gscdn.kbs.co.kr
Resolving 1fm.gscdn.kbs.co.kr for AF_INET...
Connecting to server 1fm.gscdn.kbs.co.kr[99.86.144.8]: 80...

Cache size set to 320 KBytes
Cache fill:  0.10% (329 bytes)

libavformat version 57.83.100 (external)
libavformat file format detected.
[hls,applehttp @ 0x7f19634dd2a0]Opening 'http://1fm.gscdn.kbs.co.kr/1fm_192_1_3768293.aac?m=1563518190' for reading
[lavf] stream 0: audio (aac), -aid 0
LAVF: Program 0
==========================================================================
Opening audio decoder: [ffmpeg] FFmpeg/libavcodec audio decoders
libavcodec version 57.107.100 (external)
AUDIO: 48000 Hz, 2 ch, floatle, 195.4 kbit/6.36% (ratio: 24421->384000)
Selected audio codec: [ffaac] afm: ffmpeg (FFmpeg AAC (MPEG-2/MPEG-4 Audio))
==========================================================================
[AO PCM] File: /home/u201404377/u201404377/network_week12/TEST12.flv (WAVE)
PCM: Samplerate: 48000Hz Channels: Stereo Format floatle
[AO PCM] Info: Faster dumping is achieved with -benchmark -vc null -vo null -ao pcm:fast
[AO PCM] Info: To write WAVE files use -ao pcm:waveheader (default).
AO: [pcm] 48000Hz 2ch floatle (4 bytes per sample)
Video: no video
Starting playback...
A:55885.1 (15:31:25.1) of 0.0 (00.0)  0.6% 0%
[hls,applehttp @ 0x7f19634dd2a0]Opening 'http://1fm.gscdn.kbs.co.kr/1fm_192_1_3768294.aac?m=1563518190' for reading
A:55889.1 (15:31:29.0) of 0.0 (00.0)  3.5% 0%
[hls,applehttp @ 0x7f19634dd2a0]Opening 'http://1fm.gscdn.kbs.co.kr/1fm_192_1_3768295.aac?m=1563518190' for reading
A:55893.2 (15:31:33.1) of 0.0 (00.0)  3.3% 0%
[http @ 0x7f19634e4180]HTTP error 403 Forbidden
[hls,applehttp @ 0x7f19634dd2a0]Failed to reload playlist 0
[http @ 0x7f19634e4180]HTTP error 403 Forbidden
[hls,applehttp @ 0x7f19634dd2a0]Failed to reload playlist 0
A:55893.2 (15:31:33.2) of 0.0 (00.0) 31.6% 0%


Exiting... (End of file)
u201404377@ubuntu:~/u201404377/network_week12$ ls
aa.mp3  EBSdownload.py  ebsJln.mp3  ex.html      MP3_PATH      server.py   test.mp3
EBS     ebsJIN.flv      exex.html   KBSdownload.py  noIndex.html  TEST12.flv
```

KBS radio를 다운 받는 명령어를 콘솔에서 실행해본 결과이다. TEST12.flv 파일이 받아진 것을 볼 수 있다.

밑은 이 외에 curl을 실행해본 결과들이다.

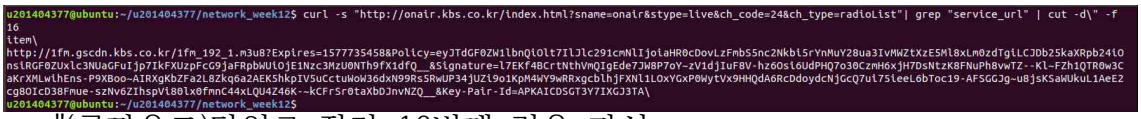
```
u201404377@ubuntu:~/u201404377/network_week12$ curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"

<!DOCTYPE html>
<html lang="ko">

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no, target-densityDpi=medium-dpi"
    <meta name="apple-mobile-web-app-capable" content="yes" />
    <meta name="apple-mobile-web-app-status-bar-style" content="black" />
    <link rel="shortcut icon" href="/resources/images/favicon/favicon.ico" /><!-- [C] 추 후 절대경로로 변경 예정 -->
    <link rel="shortcut icon" href="/resources/images/favicon/favicon.png" /><!-- [C] 추 후 절대경로로 변경 예정 -->
    <link rel="apple-touch-icon" href="/resources/images/favorite/favorite.png" /><!-- [C] 추 후 절대경로로 변경 예정 -->
    <meta name="description" content="KBS Onair" />
    <meta name="keywords" content="KBS,kbs,한국공영방송,Onair" />
    <meta property="og:title"                content="KBS 온에어"/>
    <meta property="og:description"          content="1FM 명연주 명음반 03:00~05:00"/>
    <meta property="og:site_name"            content="KBS" />
    <meta property="og:Type"                 content="video" />
    <meta property="og:url"                      content="http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"/>
    <meta property="og:image"                content="http://img.kbs.co.kr/pgm/resource/program/R2002/R2002-0286/R2002-0286_00_04.png"/>
    <meta name="twitter:card"                content="summary">
    <meta name="twitter:title"               content="KBS 온에어">
    <meta name="twitter:site"                content="KBS">
    <meta name="twitter:url"                 content="http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList">
    <meta name="twitter:image"               content="http://img.kbs.co.kr/pgm/resource/program/R2002/R2002-0286/R2002-0286_00_04.png">
    <meta name="twitter:description"    content="1FM 명연주 명음반 03:00~05:00">
    <meta name="mobile-web-app-capable" content="yes">
    <meta name="apple-mobile-web-app-capable" content="yes">
    <title>LIVE | 디지털 KBS</title>

    <link rel="stylesheet" href="/resources/css/style.min.css" media="all">

    <link rel="stylesheet" type="text/css" href="/resources/js/jwplayer/skins/kbslive.css" media="all" />
</head>
<body>
<!--
<div class="loading">
    <div class="loading-box">
        <span class="blind">로딩중 입니다.</span>
    </div>
</div>
-->
<div id="skipNav" class="skip_navi">
    <div id="skipNav" class="skip_navi">
                <ul>
                <li><a href="#content-skiplink">본문 바로가기</a></li>
```

=> 라디오 생방송 페이지를 가져온다.

curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList“

=> 라디오 생방송 페이지에서 service_url만 뽑아낸다.

curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"| grep "service_url"



=> "(큰따옴표)단위로 잘라 16번째 것을 파싱

curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"| grep "service_url" | cut -d\" -f 16

=> \단위로 잘라 첫 번째 것을 파싱

curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"| grep "service_url" | cut -d\" -f 16 | cut -d\\ -f 1

이런 식으로 크롤링 후 파싱을 하여 미디어 URL을 HLS로 요청하면 2번 가장 상단과 같다.

=> $(curl -s "http://onair.kbs.co.kr/index.html?sname=onair&stype=live&ch_code=24&ch_type=radioList"| grep "service_url" | cut -d\" -f 16 | cut -d\\ -f 1) -ao pcm:file= your/path/file_name.flv -vc dummy -vo null

3. crontab을 이용한 주기적 실행

3.1. 매분 실행

```
# 매분 test.sh 실행
* * * * * /home/script/test.sh
```

3.2. 특정 시간 실행

```
# 매주 금요일 오전 5시 45분에 test.sh 를 실행
45 5 * * 5 /home/script/test.sh
```

3.3. 반복 실행

```
# 매일 매시간 0분, 20분, 40분에 test.sh 를 실행
0,20,40 * * * * /home/script/test.sh
```

3.4. 범위 실행

```
# 매일 1시 0분부터 30분까지 매분 tesh.sh 를 실행
0-30 1 * * * /home/script/test.sh
```

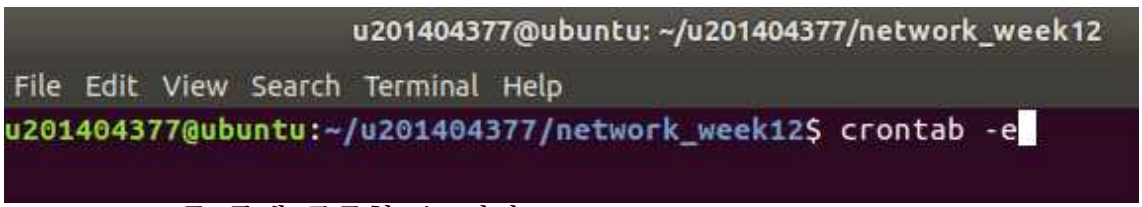
3.5. 간격 실행

```
# 매 10분마다 test.sh 를 실행
*/10 * * * * /home/script/test.sh
```

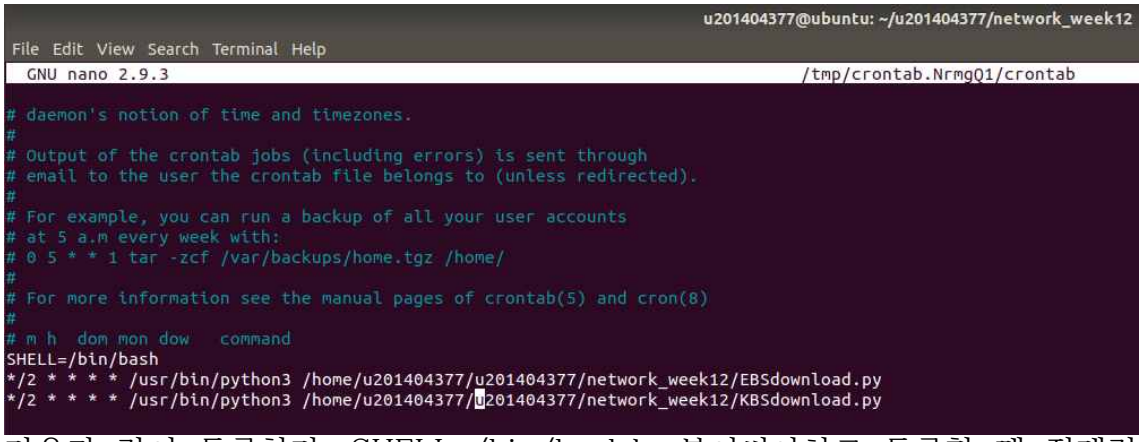
3.6. 조금 복잡하게 실행

```
# 5일에서 6일까지 2시,3시,4시에 매 10분마다 test.sh 를 실행
*/10 2,3,4 5-6 * * /home/script/test.sh
```

crontab을 사용해서 프로그램 실행을 스케줄링(예약) 할 수 있다. crontab을 통해 내가 만든 EBS, KBS download.py를 주기적으로 실행시키게 하여 mp3 파일을 다운 받게 해보았다.



crontab -e를 통해 등록할 수 있다.



다음과 같이 등록한다. SHELL=/bin/bash는 붙여써야하고 등록할 때 절대경로를 사용해야한다. 2분 간격으로 실행되게 설정하였다.

```
u201404377@ubuntu:~/u201404377/network_week12$ crontab -e
crontab: installing new crontab
u201404377@ubuntu:~/u201404377/network_week12$ crontab -e
crontab: installing new crontab
u201404377@ubuntu:~/u201404377/network_week12$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
SHELL=/bin/bash
*/2 * * * * /usr/bin/python3 /home/u201404377/u201404377/network_week12/EBSdownload.py
*/2 * * * * /usr/bin/python3 /home/u201404377/u201404377/network_week12/KBSdownload.py
u201404377@ubuntu:~/u201404377/network_week12$
```

crontab -l을 통해 등록한 것을 확인 할 수 있다.

4. 웹서버에 녹음된 파일 올리기

소켓을 이용하여 서버를 열고 클라이언트에게 다운받은 mp3을 다운 및 재생할 수 있는 웹페이지를 응답해준다.

차례대로 보면 다음과 같다.

```
# search mp3 file dir
def search(dirname):
    result = []
    filenames = os.listdir(dirname)
    for filename in filenames:
        full_filename = os.path.join(dirname, filename)
        ext = os.path.splitext(full_filename)[-1]
        if ext == '.mp3':
            result.append(full_filename)
            print("SEARCH => " , full_filename)

    return result
```

os.listdir()을 통해 mp3 파일의 절대경로들을 리스트에 저장한다.

```
# add audio html related mp3 (size : mp3 count)
def __html_templates(srces, download_name, file_name):
    tr_templates ="""<tr><td>
        <audio controls>
        <source src ='""" + srces + """' type = 'audio/mpeg'>
        </audio>
        </td>
        <td>
        <a href = '""" + srces + """' download ='""" + download_name + """'>
        """+file_name+"""</a>
        </td>
        </tr>
        """
    return tr_templates
```

mp3를 재생할 수 있는 <audio>와 다운 받을 수 있는 <a>를 html에 추가하는 코드이다. 위에서 찾은 mp3 파일만큼 반복문을 돌려 이 메소드를 호출해 추가해준다.

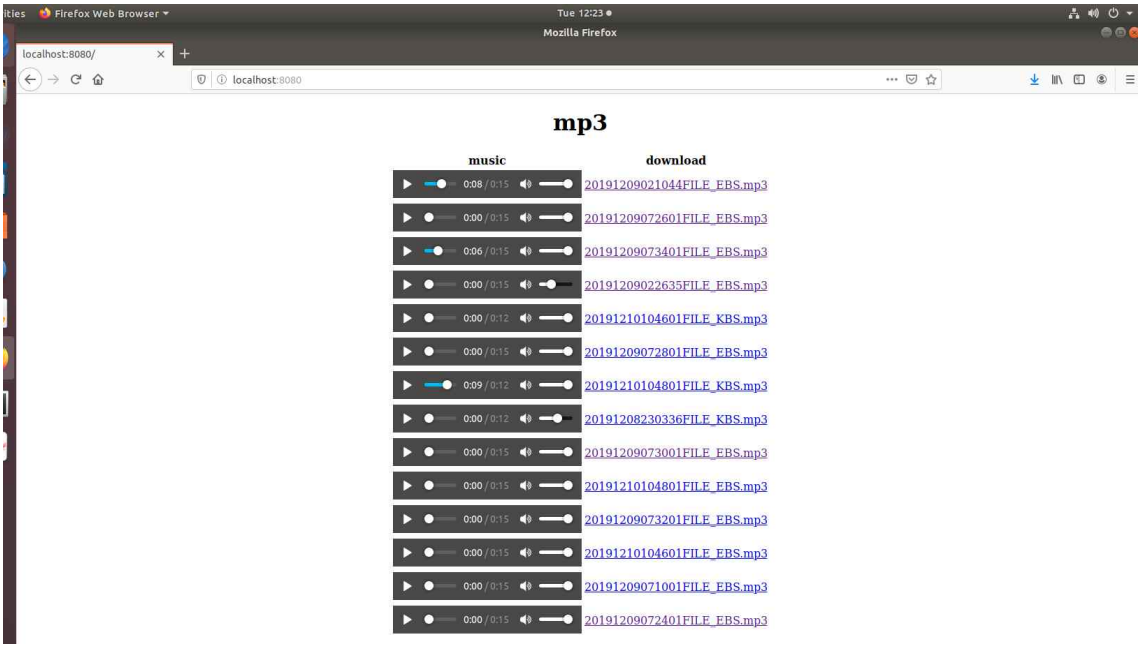
```
while True:
    print("wait ... ")
    conn, addr = s_socket.accept()
    data = conn.recv(1024).decode('utf-8')
    #http_method is POST or GET
    http_method = data.split(" ")[0]

    #response mp3 byte data
    if http_method == "GET":
        data2 = data.split(" ")[1]
        print("data ===> ", data2)
        if ".mp3" in data2:
            conn.send('HTTP/1.1 200 OK\r\n'.encode('utf-8'))
            conn.send('Content-Type: text/html\n\n'.encode('utf-8'))
            # read mp3 byte
            with open(data2, "rb") as byteData:
                conn.send(byteData.read())
            conn.close()
            continue
```

mp3 파일을 get요청 했을 경우 해당 파일을 바이트로 읽어 send 해준다.

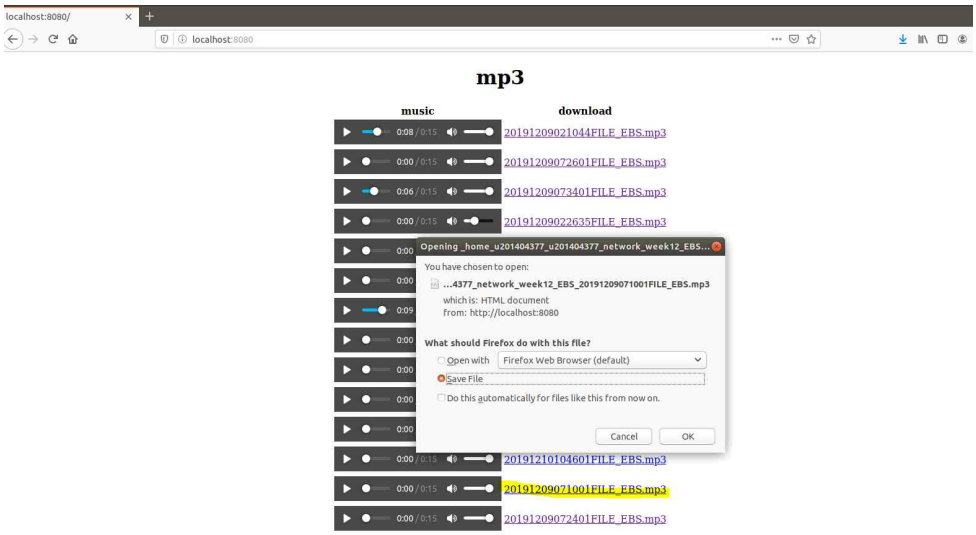
```
        try:
            # response audio html
            if http_method == "GET":
                #HTML
                outputdata = """<!DOCTYPE html>
                            <html>
                             <head>
                                </head>
                        <body>
                        <h1 style = 'text-align:center'>
                           mp3
    </h1>
    <table style='margin-left: auto; margin-right: auto;'>
    <tr>
    <th> music </th>
    <th> download </th>
    </tr>
"""
                outputdata += mp3_html
                outputdata += """</table></body></html>"""
                print(outputdata)
                #SEND
                conn.send('HTTP/1.1 200 OK\r\n'.encode('utf-8'))
                conn.send('Content-Type: text/html\n\n'.encode('utf-8'))
                conn.send(outputdata.encode('utf-8'))
                conn.close()
```

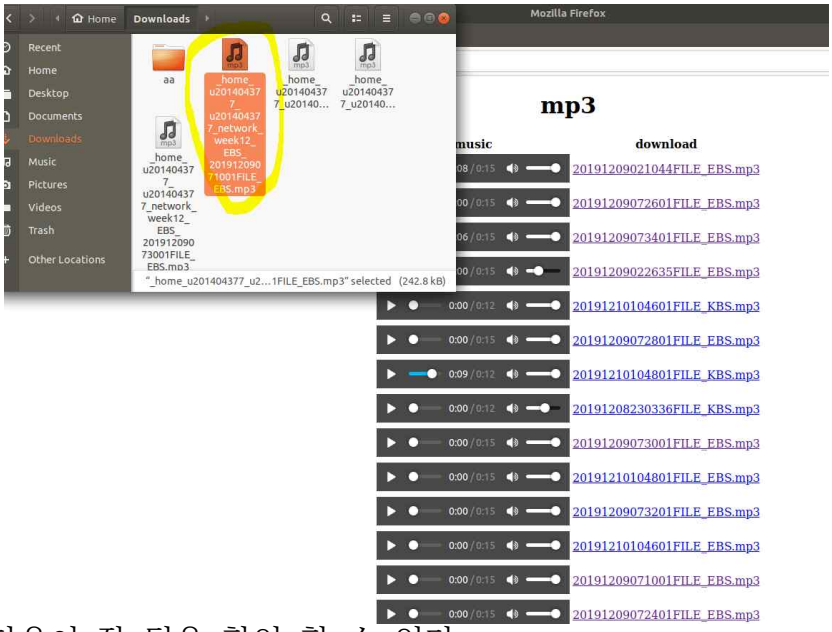
처음 클라이언트가 웹페이지를 요청할 때 띄울 html을 보내준다. mp3_html은 __html_templates() 함수로 저장된 mp3 재생 및 다운로드 관련 html이 저장되어있다.

서버를 실행시킨 결과는 다음과 같다.

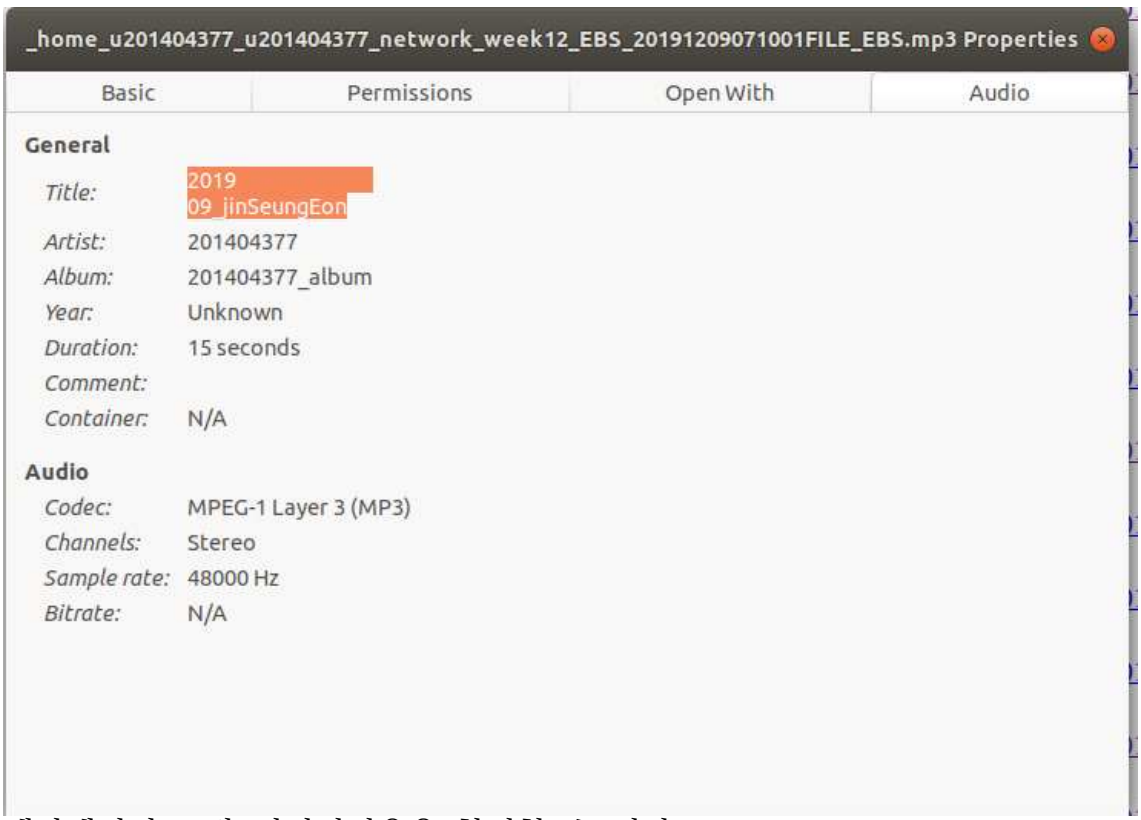


밑과 같이 mp3 파일의 재생과 다운이 가능하다.





다운이 잘 됨을 확인 할 수 있다.



메타데이터도 잘 변경되었음을 확인할 수 있다.

과제 후기

python을 네트워크를 통해 처음 접했는데 익숙하지 않아 과제가 더 힘들었던 것 같다. 그래도 잘 마무리해서 다행이다.
매번 질문 잘 받아주셔서 감사합니다.