Internet Radio using IP Multicast

Group 18

Dhruv Shah	1741008
Mann Bilimoria	1741021
Dhairya Dudhatra	1741058
Param Raval	1741083

Objective:

The objective of this project is to develop an internet radio by implementing multicast such that multiple users can listen to the same live-feed and even change the radio stations as they wish. We use the Any Source Multicast (ASM) model in which the messages are identified by the multicast group address alone. Several users can reside in the same multicast group, and nodes connected over the same group receive all the messages sent to that group.

The client will have the functionality to pause, play, and change station at will without losing synchronization with the live stream. The server can add new stations, delete stations, and add songs to each station.

Concepts:

- A TCP socket is used as a control channel and receiving station info from the server.
- Server sends multimedia over a UDP socket once the client is connected
- To enable multiple users/clients, we use threading which creates a new thread each time a client connects with the server
- The multicast reception stops and the thread is deleted once the client disconnects (pauses) with the server. A new thread is formed upon restarting.

Structures used:

```
station_info_request:
uint8_t type = 1;

site_info:
uint8_t type = 10;
uint8_t site_name_size;
```

```
char site_name[ site_name_size ];
uint8_t site_desc_size;
char site_desc[ site_desc_size ];
uint8_t station_count;
station_info station_list [station_count];

station_info:
uint8_t station_number;
uint8_t station_name_size;
char station_name[ station_name_size ];
uint32_t multicast_address;
uint16_t data_port;
uint16_t info_port;
uint32_t bit_rate;

song_info:
uint8_t type = 12;
uint8_t song_name_size;
char song_name[ song_name_size ];
uint16_t remaining_time_in_sec;
uint8_t next_song_name_size;
char next_song_name[ next_song_name_size ];
```

These structures are common for all other internet radios in the class project

Design details:

- A thread for the TCP connection, which sends the station list to the client. Thus, if a client will have to wait if another client simultaneously asks for a TCP connection.
- Each multicast group will have a multicast thread (UDP) for multimedia transfer on the server side. Server will have one thread for each station running on it.
- Client has a thread setup for the TCP connection with the server. Here, the client receives the list of stations from the server which it can display on the interface.
- The other thread running on client is to handle the UDP multicast where it receives the multicast from the server.
- All the threads used in both client and server are in detach state i.e. they can run simultaneously and will not interfere with other running threads.
- When a client connects to the server, the server first sends the lists of stations to the client. The client proceeds to choose a station from the list, upon selecting it is automatically connected to the multicast of that corresponding station.

- The server maintains a table for all the stations created as well as saving their memory references.
- Compile the client side code using `gcc receivefinal4.c `pkg-config --cflags gtk+-3.0` -pthread`

Determining the Data Rate:

In ffmpeg, the streaming data rate from the server’s side is 425.2 Kbps.
From this we get the number of packets to be sent per second

$$\frac{Data\ Rate}{Buffer\ size} = \frac{425200}{16384*8} = 3.244 \text{ pps}$$

Thus, the delay between two packets is (1/(3.244)) seconds. Considering other factors for streaming, we keep the delay as 150ms (or half of 300ms).

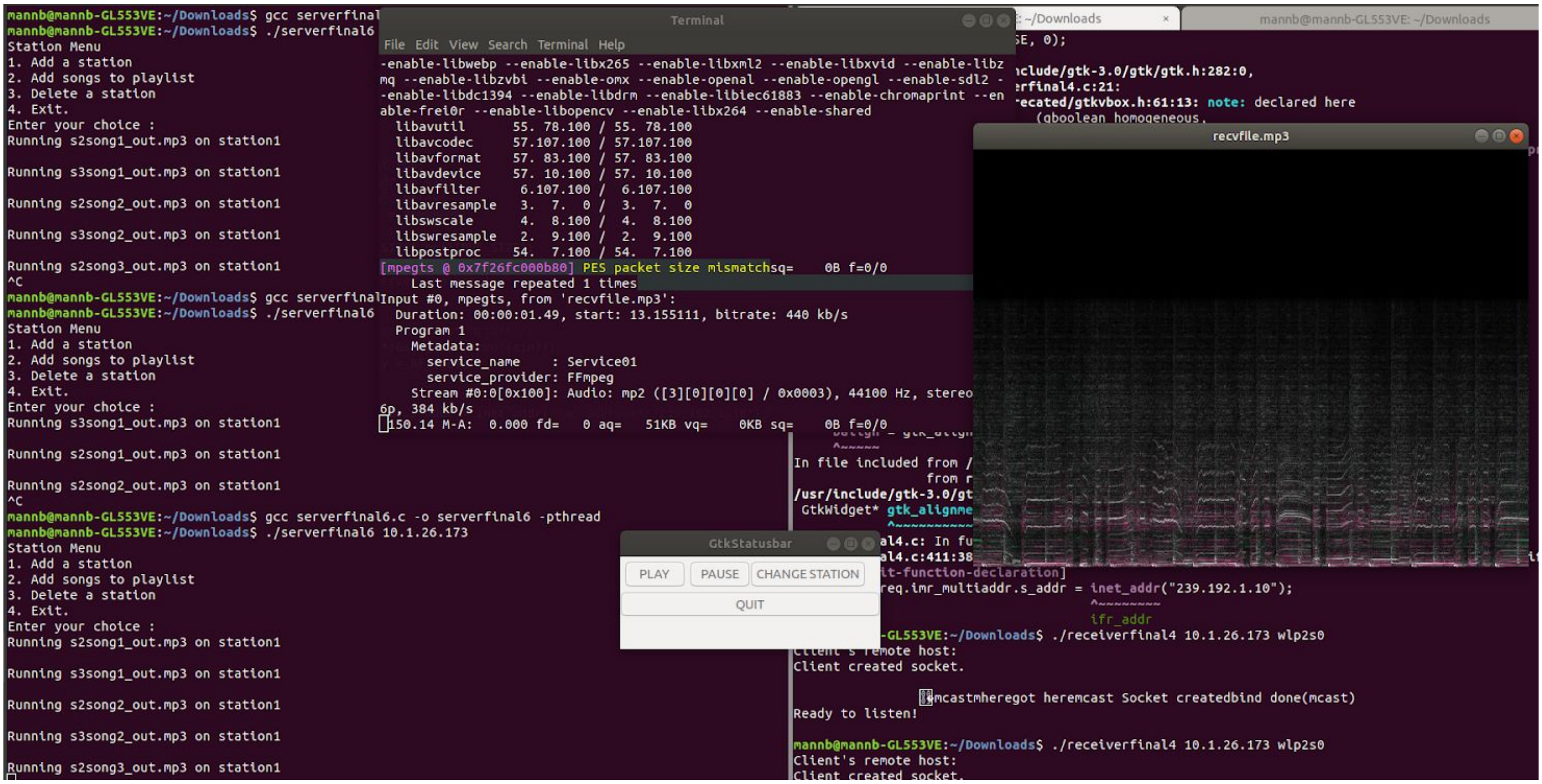
The average data rate on the receiver side is 400-430 Kbps.

Playback buffer size:
The limit for a UDP datagram is 65,535 bytes (65,507 data bytes). But keeping this buffer size caused the packets to be stored and thus, effectively getting skipped while streaming.
By experimenting some values we have kept the playback buffer size as 16,384 bytes (half of the maximum size).

Features:

- Server offers options to add a new station as well as delete a station.
- Client can switch between stations anytime it wants.
- All the threads are set in Detach state. Detached threads do not interfere with or block each other and run parallel.
- Threads on client side:
    - Thread for TCP (main)
    - Thread for multicast (UDP)
- Threads on server side:
    - Main thread
    - Thread for each station
    - Thread for each client

Screenshots of results:



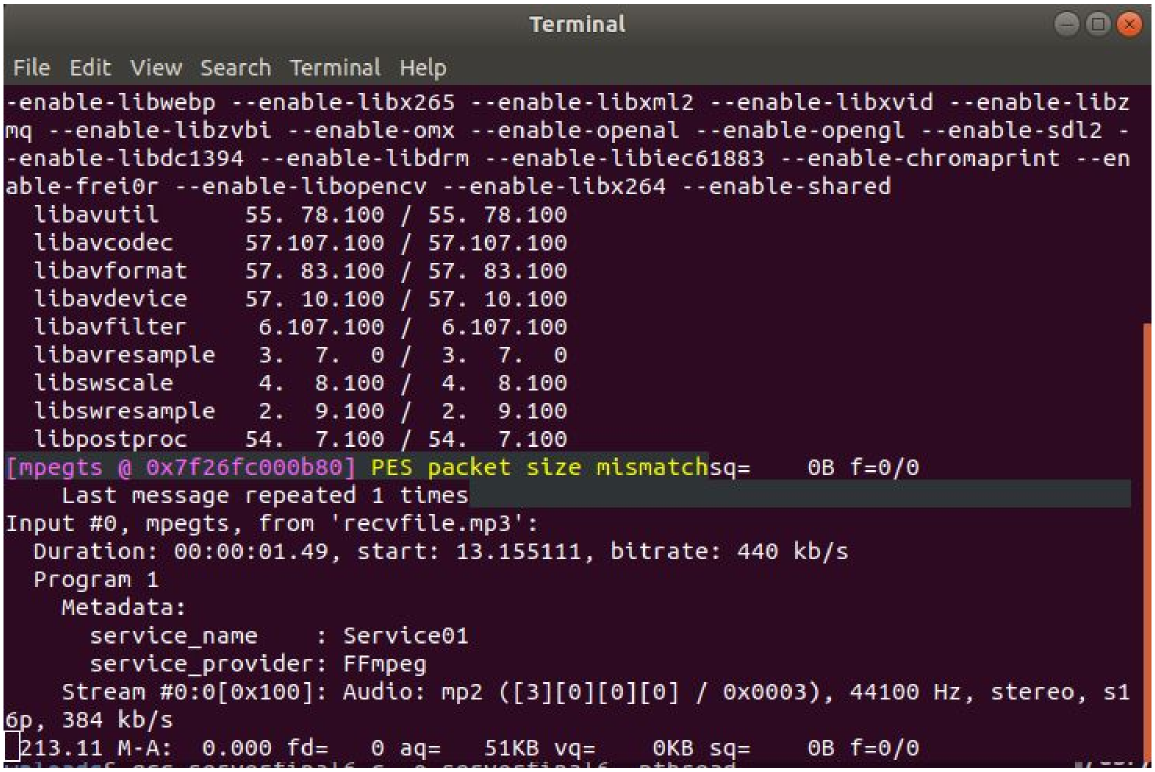
```
mannb@mannb-GL553VE:~/Downloads$ gcc serverfinal
mannb@mannb-GL553VE:~/Downloads$ ./serverfinal6
Station Menu
1. Add a station
2. Add songs to playlist
3. Delete a station
4. Exit.
Enter your choice :
Running s2song1_out.mp3 on station1

Running s3song1_out.mp3 on station1

Running s2song2_out.mp3 on station1

Running s3song2_out.mp3 on station1

Running s2song3_out.mp3 on station1
^C
mannb@mannb-GL553VE:~/Downloads$ gcc serverfinal
mannb@mannb-GL553VE:~/Downloads$ ./serverfinal6
Station Menu
1. Add a station
2. Add songs to playlist
3. Delete a station
4. Exit.
Enter your choice :
Running s3song1_out.mp3 on station1

Running s2song1_out.mp3 on station1

Running s2song2_out.mp3 on station1
^C
mannb@mannb-GL553VE:~/Downloads$ gcc serverfinal6.c -o serverfinal6 -pthread
mannb@mannb-GL553VE:~/Downloads$ ./serverfinal6 10.1.26.173
Station Menu
1. Add a station
2. Add songs to playlist
3. Delete a station
4. Exit.
Enter your choice :
Running s2song1_out.mp3 on station1

Running s3song1_out.mp3 on station1

Running s2song2_out.mp3 on station1

Running s3song2_out.mp3 on station1

Running s2song3_out.mp3 on station1
```

```
-enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libz
mq --enable-libzvbi --enable-omx --enable-openal --enable-opengl --enable-sdl2 -
-enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint --en
able-frei0r --enable-libopencv --enable-libx264 --enable-shared
  libavutil      55. 78.100 / 55. 78.100
  libavcodec     57.107.100 / 57.107.100
  libavformat    57. 83.100 / 57. 83.100
  libavdevice    57. 10.100 / 57. 10.100
  libavfilter     6.107.100 /  6.107.100
  libavresample   3.  7.  0 /  3.  7.  0
  libswscale      4.  8.100 /  4.  8.100
  libswresample   2.  9.100 /  2.  9.100
  libpostproc    54.  7.100 / 54.  7.100
[mpegts @ 0x7f26fc000b80] PES packet size mismatchsq=    0B f=0/0
    Last message repeated 1 times
Input #0, mpegts, from 'recvfile.mp3':
  Duration: 00:00:01.49, start: 13.155111, bitrate: 440 kb/s
  Program 1
    Metadata:
      service_name    : Service01
      service_provider: FFmpeg
    Stream #0:0[0x100]: Audio: mp2 ([3][0][0][0] / 0x0003), 44100 Hz, stereo
6p, 384 kb/s
 150.14 M-A:  0.000 fd=   0 aq=   51KB vq=    0KB sq=    0B f=0/0
```

```
SE, 0);

nclude/gtk-3.0/gtk/gtk.h:282:0,
erfinal4.c:21:
recated/gtkvbox.h:61:13: note: declared here
   (gboolean homogeneous,
```

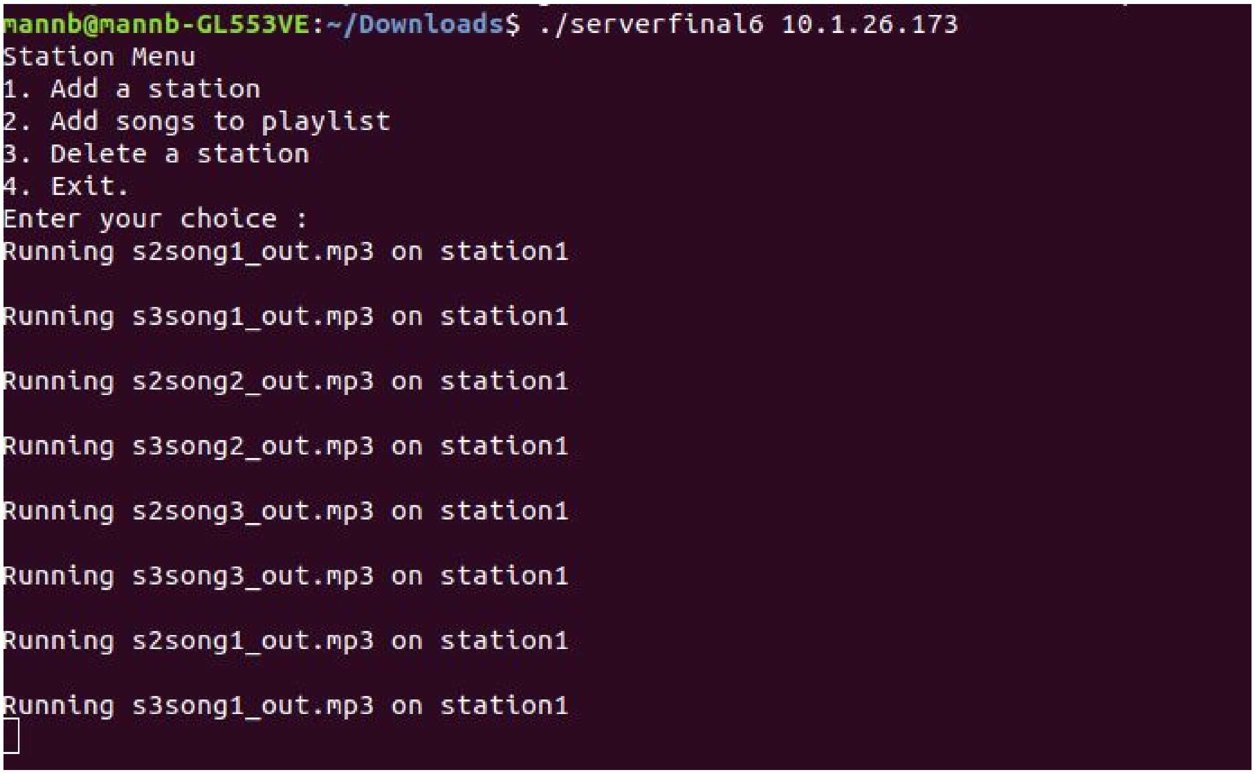
```
          ^~~~~~
In file included from /
                 from r
/usr/include/gtk-3.0/gt
 GtkWidget* gtk_alignme
            ^~~~~~~~~~~
al4.c: In fu
al4.c:411:38
it-function-declaration]
req.imr_multiaddr.s_addr = inet_addr("239.192.1.10");
                           ^~~~~~~~~
                           ifr_addr
-GL553VE:~/Downloads$ ./receiverfinal4 10.1.26.173 wlp2s0
Client's remote host:
Client created socket.

                mcastmheregot heremcast Socket createdbind done(mcast)
Ready to listen!

mannb@mannb-GL553VE:~/Downloads$ ./receiverfinal4 10.1.26.173 wlp2s0
Client's remote host:
Client created socket.
```

Here we see the server on the left side and client on the right.
FFMPEG is playing the recvfile.mp3 temp buffer file in the window on the right.
Our GUI window is in the centre with: PLAY(restart), PAUSE, CHANGE STATION and QUIT buttons.



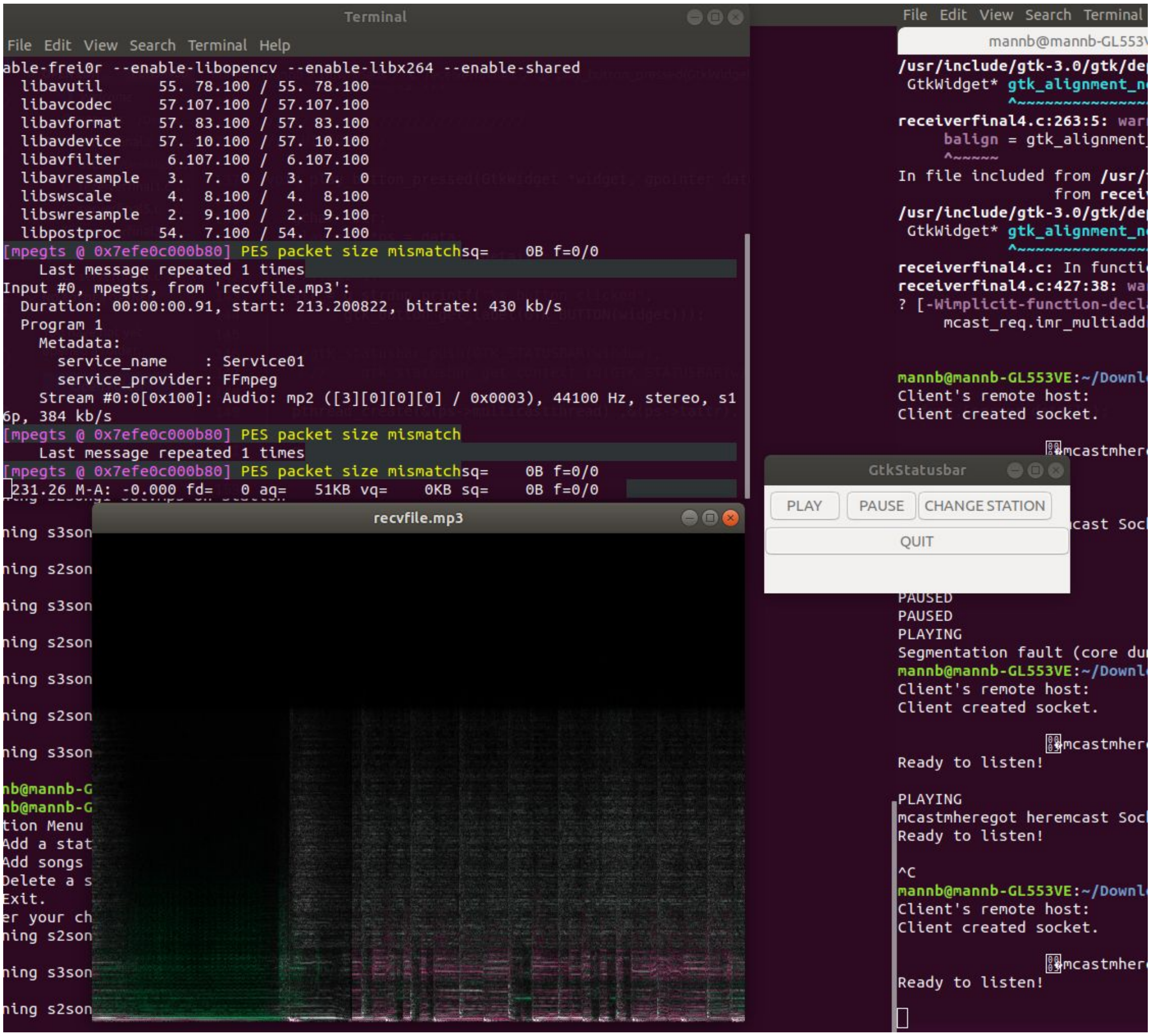
```
-enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libz
mq --enable-libzvbi --enable-omx --enable-openal --enable-opengl --enable-sdl2 -
-enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint --en
able-frei0r --enable-libopencv --enable-libx264 --enable-shared
  libavutil      55. 78.100 / 55. 78.100
  libavcodec     57.107.100 / 57.107.100
  libavformat    57. 83.100 / 57. 83.100
  libavdevice    57. 10.100 / 57. 10.100
  libavfilter     6.107.100 /  6.107.100
  libavresample   3.  7.  0 /  3.  7.  0
  libswscale      4.  8.100 /  4.  8.100
  libswresample   2.  9.100 /  2.  9.100
  libpostproc    54.  7.100 / 54.  7.100
[mpegts @ 0x7f26fc000b80] PES packet size mismatchsq=    0B f=0/0
    Last message repeated 1 times
Input #0, mpegts, from 'recvfile.mp3':
  Duration: 00:00:01.49, start: 13.155111, bitrate: 440 kb/s
  Program 1
    Metadata:
      service_name    : Service01
      service_provider: FFmpeg
    Stream #0:0[0x100]: Audio: mp2 ([3][0][0][0] / 0x0003), 44100 Hz, stereo, s1
6p, 384 kb/s
 213.11 M-A:  0.000 fd=   0 aq=   51KB vq=    0KB sq=    0B f=0/0
```

Additional information provided by FFMPEG



```
mannb@mannb-GL553VE:~/Downloads$ ./serverfinal6 10.1.26.173
Station Menu
1. Add a station
2. Add songs to playlist
3. Delete a station
4. Exit.
Enter your choice :
Running s2song1_out.mp3 on station1

Running s3song1_out.mp3 on station1

Running s2song2_out.mp3 on station1

Running s3song2_out.mp3 on station1

Running s2song3_out.mp3 on station1

Running s3song3_out.mp3 on station1

Running s2song1_out.mp3 on station1

Running s3song1_out.mp3 on station1
```

Window on server side with functionalities as listed.



```
able-frei0r --enable-libopencv --enable-libx264 --enable-shared
  libavutil      55. 78.100 / 55. 78.100
  libavcodec     57.107.100 / 57.107.100
  libavformat    57. 83.100 / 57. 83.100
  libavdevice    57. 10.100 / 57. 10.100
  libavfilter     6.107.100 /  6.107.100
  libavresample   3.  7.  0 /  3.  7.  0
  libswscale      4.  8.100 /  4.  8.100
  libswresample   2.  9.100 /  2.  9.100
  libpostproc    54.  7.100 / 54.  7.100
[mpegts @ 0x7efe0c000b80] PES packet size mismatchsq=    0B f=0/0
    Last message repeated 1 times
Input #0, mpegts, from 'recvfile.mp3':
  Duration: 00:00:00.91, start: 213.200822, bitrate: 430 kb/s
  Program 1
    Metadata:
      service_name    : Service01
      service_provider: FFmpeg
    Stream #0:0[0x100]: Audio: mp2 ([3][0][0][0] / 0x0003), 44100 Hz, stereo, s1
6p, 384 kb/s
[mpegts @ 0x7efe0c000b80] PES packet size mismatch
    Last message repeated 1 times
[mpegts @ 0x7efe0c000b80] PES packet size mismatchsq=    0B f=0/0
 231.26 M-A: -0.000 fd=   0 aq=   51KB vq=    0KB sq=    0B f=0/0

ning s3son

ning s2son

ning s3son

ning s2son

ning s3son

ning s2son

ning s3son

nb@mannb-G
nb@mannb-G
tion Menu
Add a stat
Add songs
Delete a s
Exit.
er your ch
ning s2son

ning s3son

ning s2son
```

```
/usr/include/gtk-3.0/gtk/de
 GtkWidget* gtk_alignment_n
            ^~~~~~~~~~~~~~~
receiverfinal4.c:263:5: war
     balign = gtk_alignment_
     ^~~~~~
In file included from /usr/
                 from recei
/usr/include/gtk-3.0/gtk/de
 GtkWidget* gtk_alignment_n
            ^~~~~~~~~~~~~~~
receiverfinal4.c: In functi
receiverfinal4.c:427:38: wa
? [-Wimplicit-function-decl
     mcast_req.imr_multiadd


mannb@mannb-GL553VE:~/Downl
Client's remote host:
Client created socket.

                  mcastmher

                  cast Soc


PAUSED
PAUSED
PLAYING
Segmentation fault (core du
mannb@mannb-GL553VE:~/Downl
Client's remote host:
Client created socket.

                  mcastmher
Ready to listen!

PLAYING
mcastmheregot heremcast Soc
Ready to listen!

^C
mannb@mannb-GL553VE:~/Downl
Client's remote host:
Client created socket.

                  mcastmher
Ready to listen!
```

Overall working window with GUI

Work distribution:

Dhruv Shah	1741008	Server side and overall threading
Mann Bilimoria	1741021	Server side
Dhairya Dudhatra	1741058	GUI
Param Raval	1741083	Client side and debugging