Start

Idle

User closes program → Exit

Streaming / mic ← Choose mode → Media player

↓

File transfer

*Mode details on separate diagrams

Each mode is a module. Different team members will be responsible for implementing each.

Client

Server

scan all .wav &
.mp3 file names
in current
directory

populate
dropdown

listen on port 6000

connect to given
ip & port 6000

recv

send file names (&
sizes) back

populate
dropdown

recv

select song

send file name

recv

send file size as
long

parse sizeof(long)
bytes for file size

recv

send file size long
back

recv

If file sizes are same
(security check),
send file as bytes
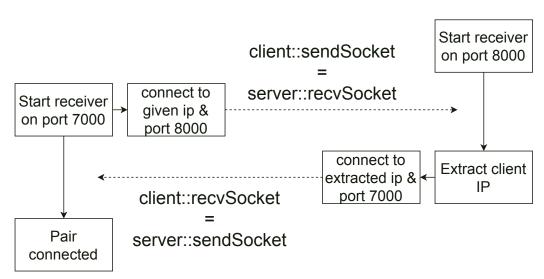
stop recv if:
- file size bytes recv'd
- socket disconnected

```
- disconnect called by
         user
```

```
If disconnect called by
user: warn that audio
file may be unplayable
```

Unlike other 2 modes, doesnt need to choose mode, just need to follow steps

Pick file in local directory

Mode enabled by default since doesnt need to connect

Play

Pause

file open error

User interact with music

fast forward

slow down

song ended

song ended

Stop

Return to idle

Client

Server

client::sendSocket
=
server::recvSocket

Start receiver
on port 8000

Start receiver
on port 7000

connect to
given ip &
port 8000

connect to
extracted ip &
port 7000

Extract client
IP

Pair
connected

client::recvSocket
=
server::sendSocket

A single connected pair

Streaming /
mic
mode chosen

Client
mode

Start thread

Server mode

Start
listening

Disconnect
client & remove
pair from map

Start
listening

Initiate connection

For every client that disconnects

For every client that connects

Use map to link pairs of sockets

add pair to map

Connect & make pair with above method

Client side

Server side

Transfer mode?

Transfer mode?

*Stream & multicast same on client

Stream / Multicast

Mic

Multicast

Stream

Run loop

Wait if mic /socket no data

Data available to read?

Data available to read?

wait until multicast button pressed

For client that just connected

Yes

Yes

Yes

For every connected client

Read bytes from file

Pull recvSocket data

Pull recvSocket data

Pull mic data

Push to speaker

Push to speaker

Push to sendSocket

Push to sendSocket

User disconnects

Disconnect any connected sockets

Return to idle

Connection refused