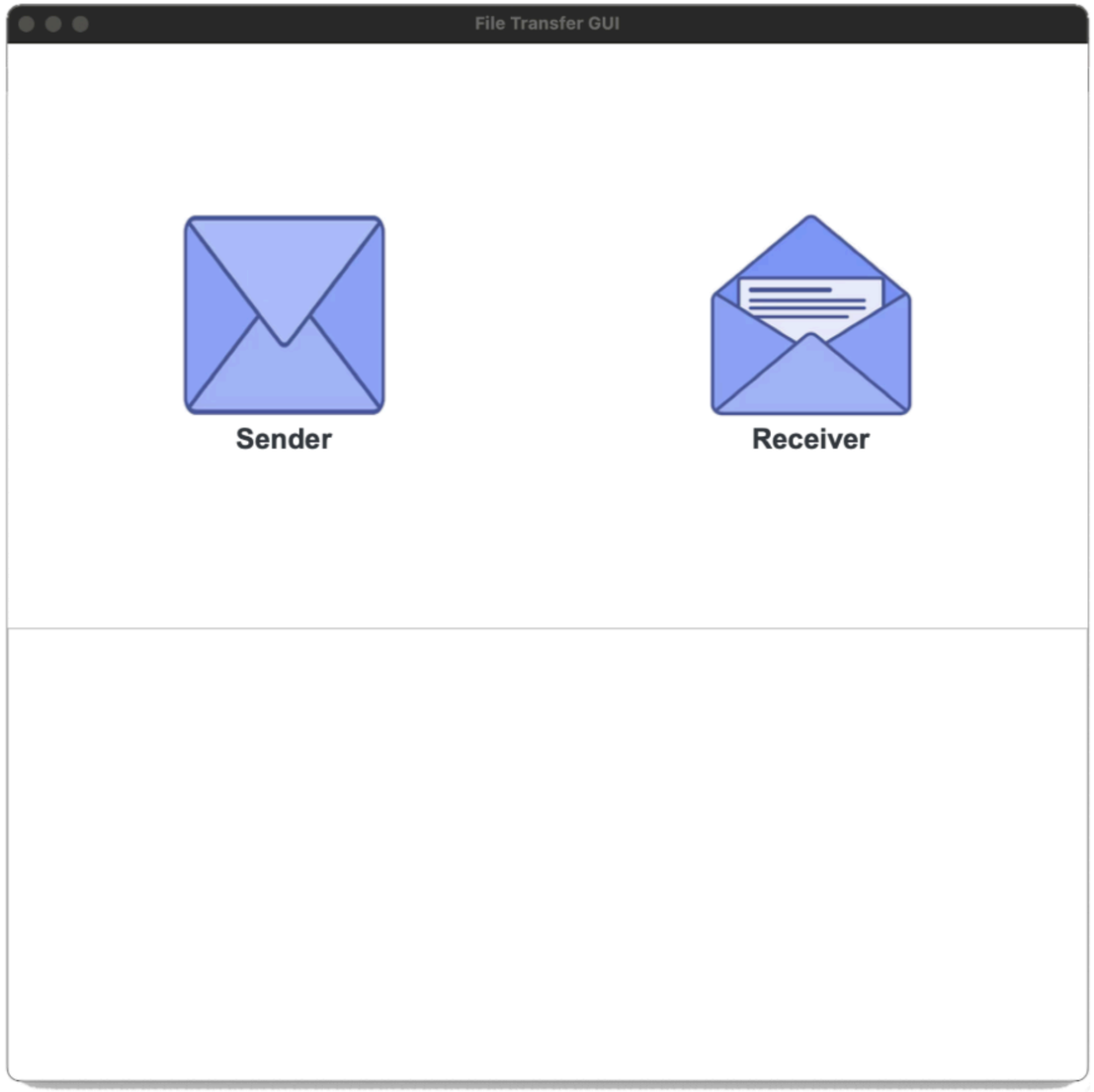
שידור מידע בין 2 מחשבים - רמקול ומיקרופון

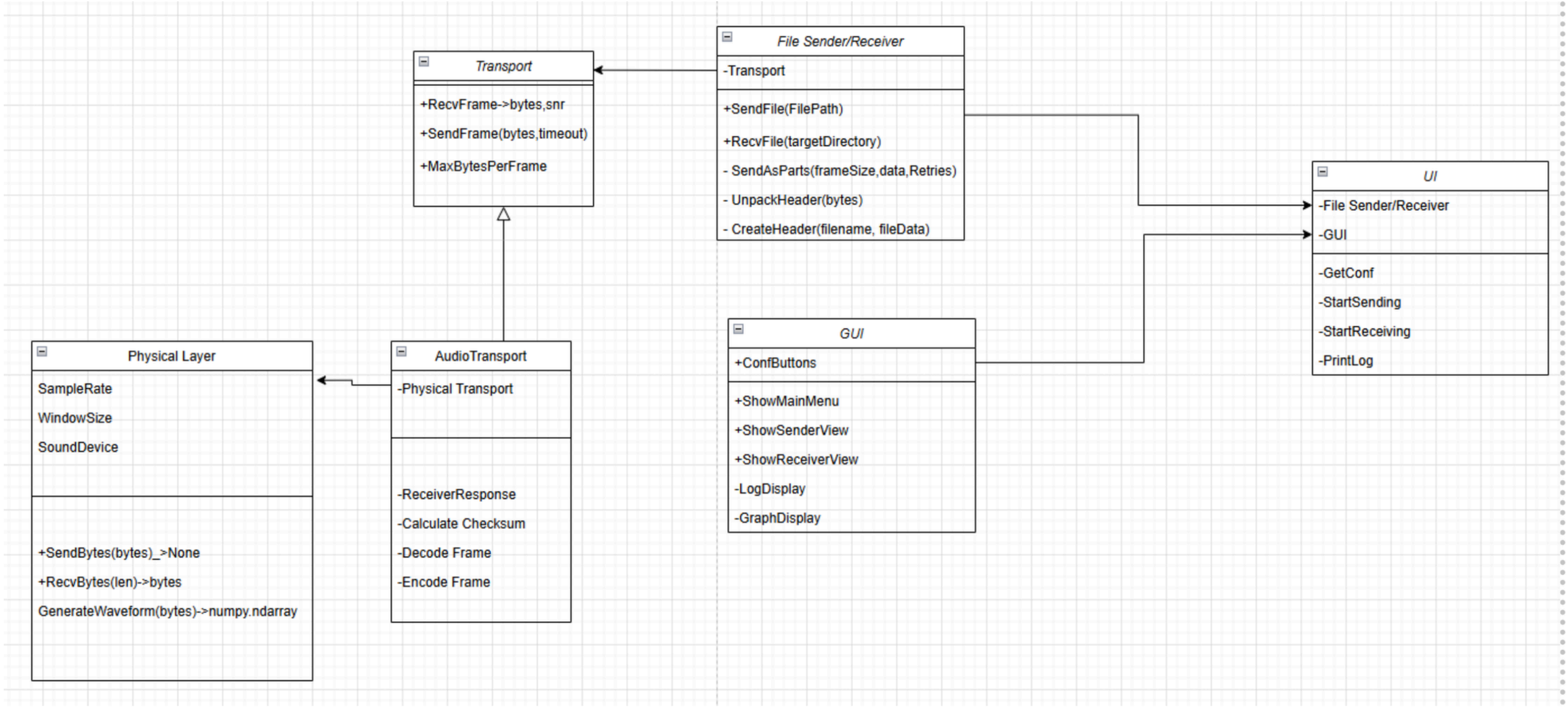
Gil Etzioni

Zak Portnoy

Final Project



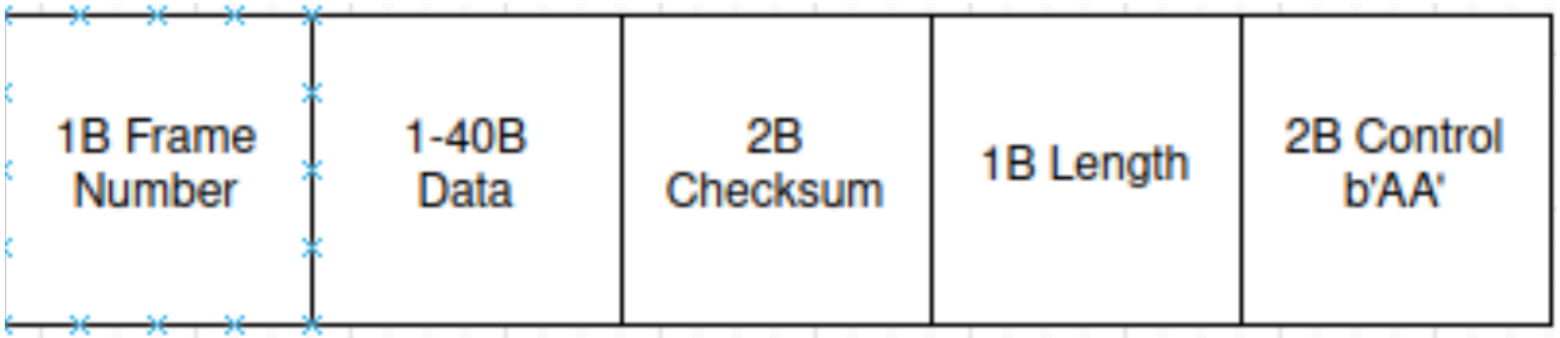
Class Diagram



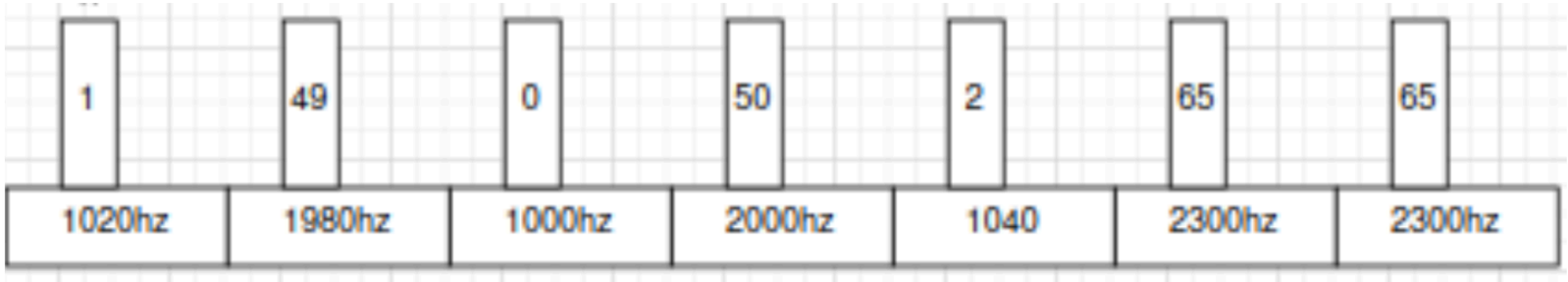
Protocols

File Header

Frame Format

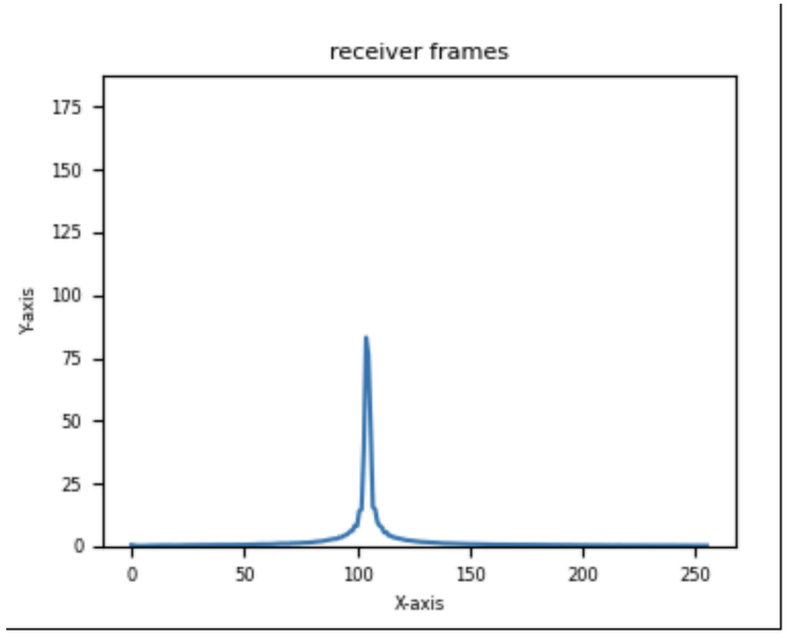


Audio Transmission

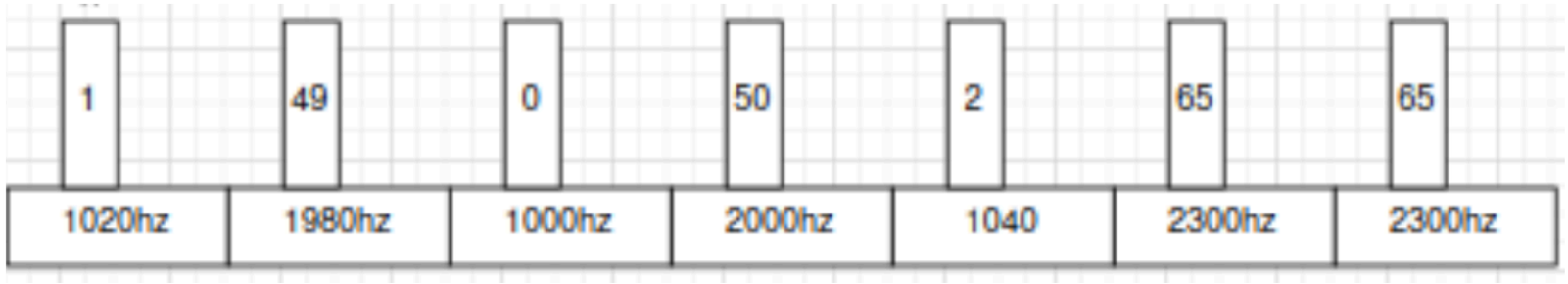


Byte encoding

Sampling frequency to bytes with fft



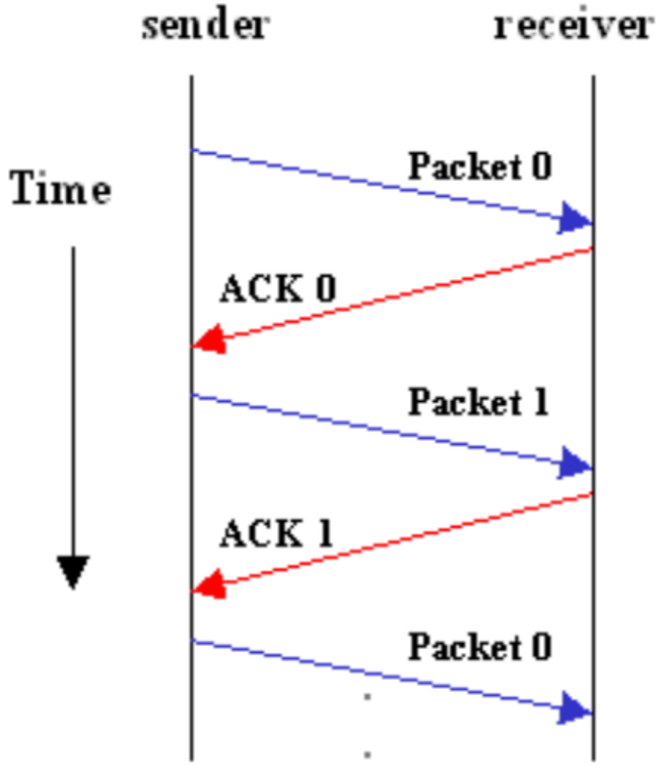
Translate frequencies and bytes



Flow control

Stop and wait with frame numbering

- The sender waits for receiver to respond between each frame.
- If there is no response it will resend up to three times
- The receiver will ignore repeated frames with the same number



Transmission speed

Issues encountered

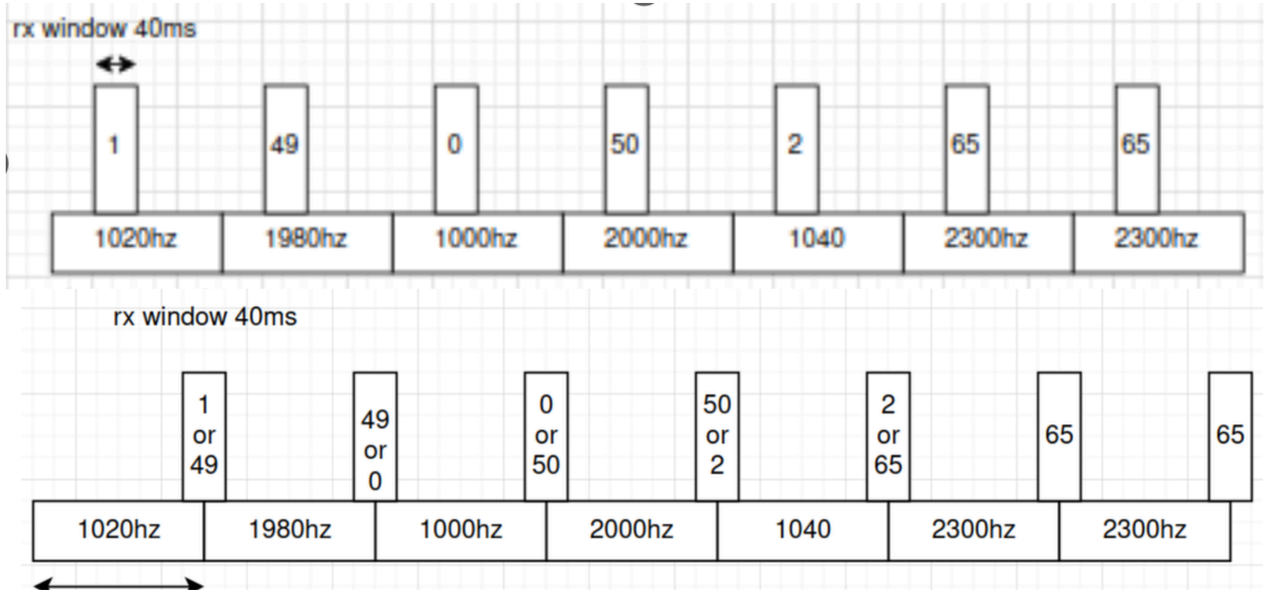
- Some Frequencies are received much stronger than others
- Signals last longer than expected and bytes overlap
- It takes time for a frequency to stabilize on the microphone end

Solutions

- Keep signal per byte at least 100ms
- Identify entire frame at once on the receivers end
- Sample the receiver at a higher rate with a smaller window size

Identifying a full frame at once

We store enough samples to identify the entire frame and return once one is found
Len, checksum and frame header helps us avoid invalid frames



install

- המערכת מותאמת לעבודה בסביבת Windows, Linux ו- mac.

- בכדי להריץ את התוכנית נדרש לבצע את ההתקנות הבאות:
  - נדרש לוודא שהמחשב מכיל גרסה עדכנית של שפת python (מומלץ לוודא שהגרסה היא לפחות 3.13.0).

- כדי לבצע את כל ההתקנות, ולוודא שהמחשב מכיל את כל הסיפריות המעודכנות, נדרש לכתוב את הפקודות הבאות ל-terminal:

```
python --version # ensure you have python 3.13.0 installed
pip install tk ttkbootstrap Pillow matplotlib numpy sounddevice # install the required libraries
```

Run

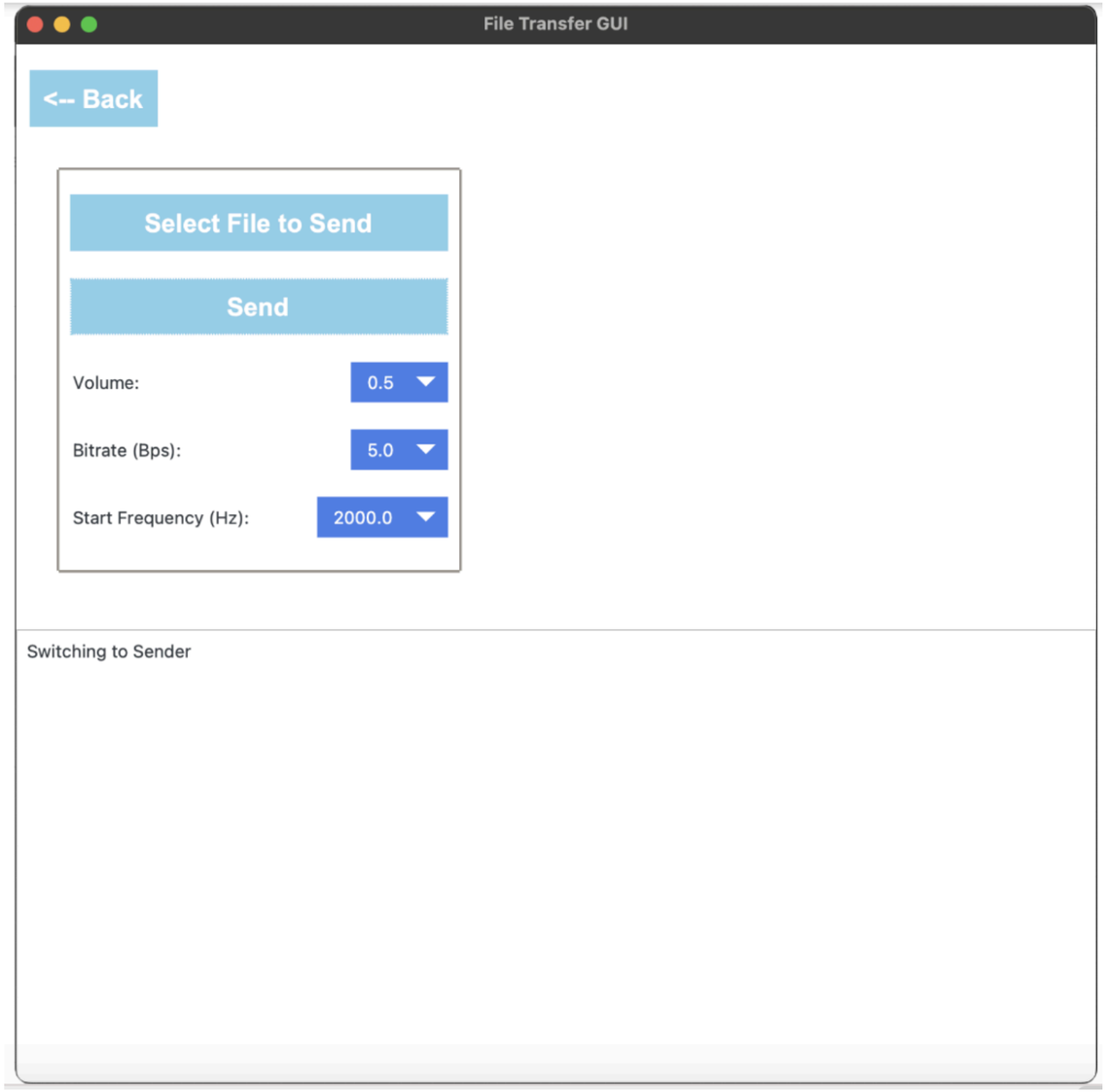
- מומלץ להריץ את התוכנית על 2 מחשבים שונים (sender and receiver), אבל התוכנה עובדת גם על מחשב אחד.

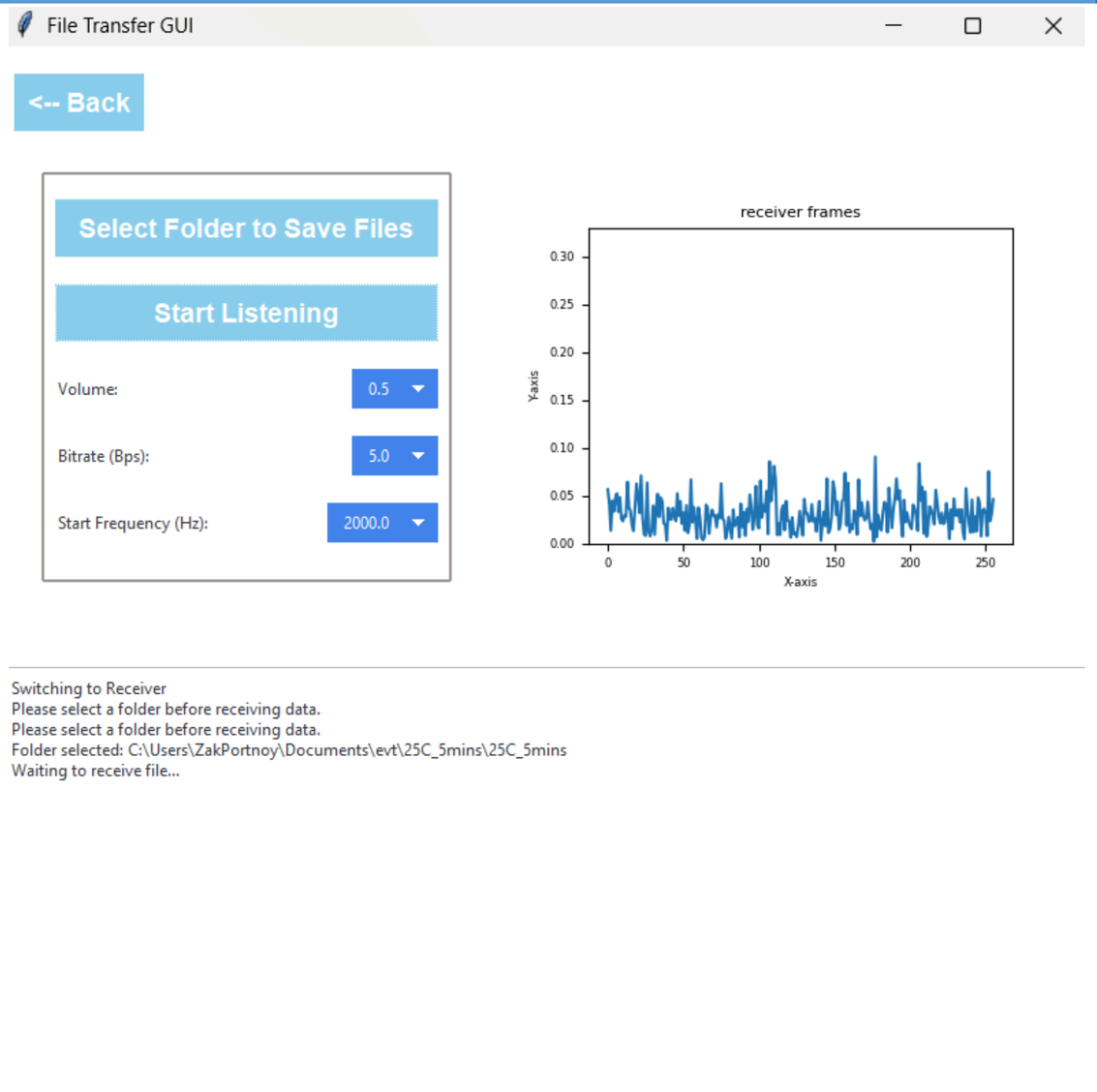
- לפני ההרצה:
  - שולח - נדרש לבחור קובץ שאותו ישלח (המערכת עובדת גם עם קבצים גדולים אבל זה מאוד איטי, ולכן בהצגה נעבוד עם קובץ txt קטן).
  - מקבל - עליו לוודא שיש לו תיקייה מוכנה לקבלת הקבצים.

- הרצה - נדרש לכתוב ב- terminal:

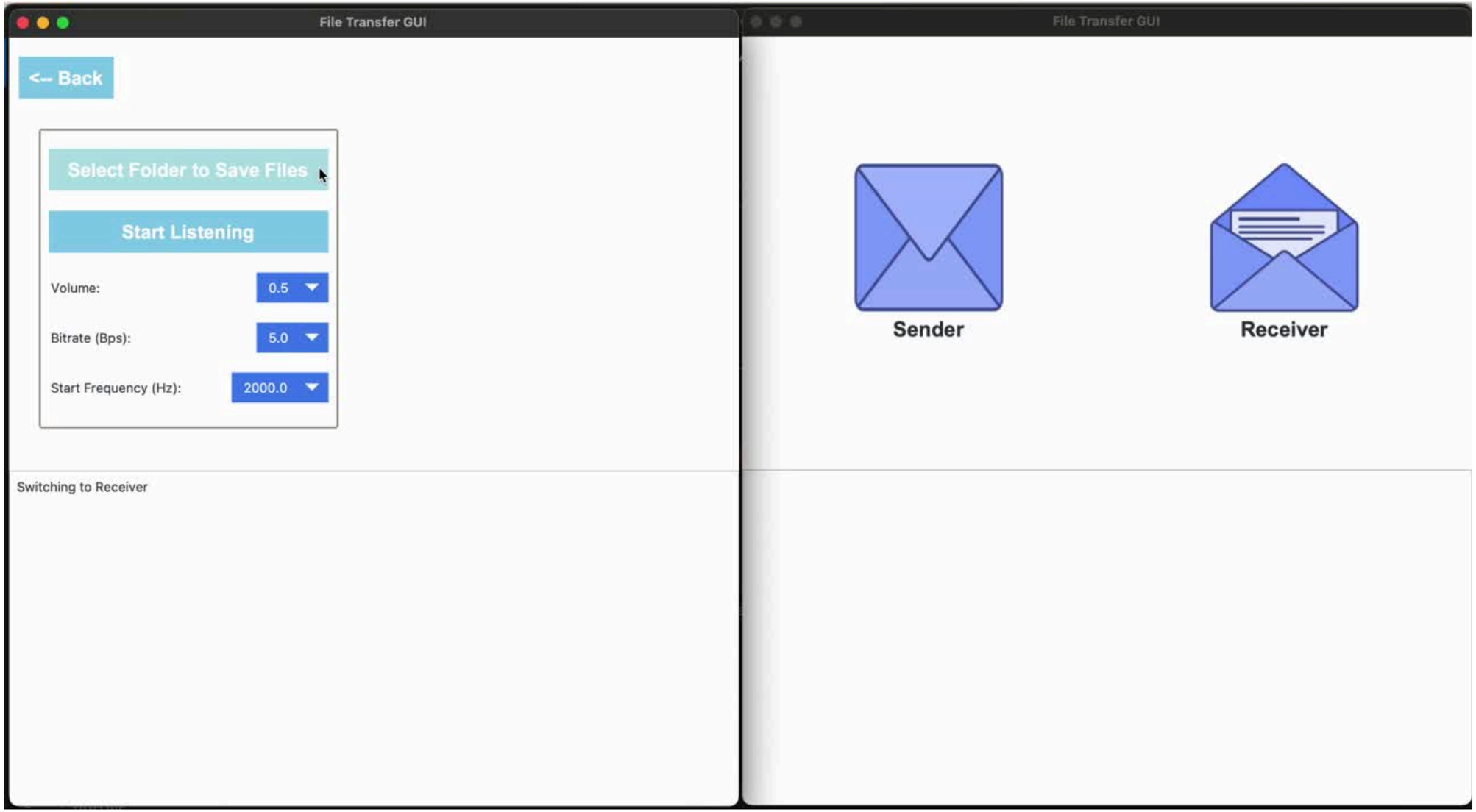
```
python3 src/main.py
```

Demo

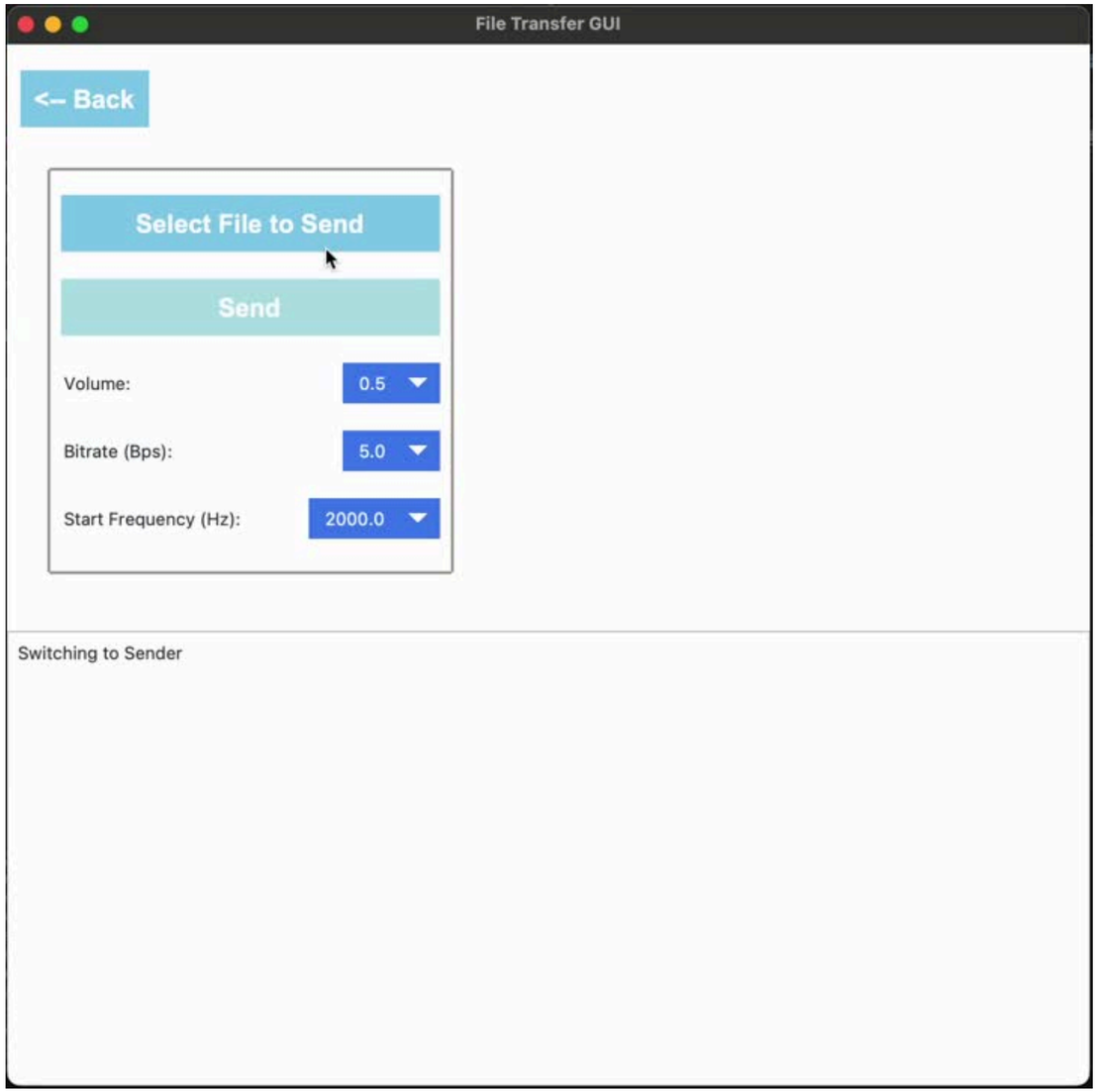

File Transfer GUI
<-- Back
Select File to Send
Send
Volume:
0.5
Bitrate (Bps):
5.0
Start Frequency (Hz):
2000.0
Switching to Sender


File Transfer GUI
<-- Back
Select Folder to Save Files
Start Listening
Volume:
0.5
Bitrate (Bps):
5.0
Start Frequency (Hz):
2000.0
receiver frames
Y-axis
X-axis
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
50
100
150
200
250
Switching to Receiver
Please select a folder before receiving data.
Please select a folder before receiving data.
Folder selected: C:\Users\ZakPortnoy\Documents\evt\25C_5mins\25C_5mins
Waiting to receive file...

Success (no sound)



Failure (no sound)



Physical layer Improvements

- We can use more than one frequency at a time to enable us to represent more than one byte per encoded signal. This will reduce the maximum amplitude per frequency but considering that stabilization time is our problem this should still give us a good signal. Using the same 256 steps and encoding 2 frequencies at once should double the transmission speed. 4 at once should quadruple it and so on.
- When processing the audio, we can look at not only the loudest frequency and try to identify the top few
- We can look into different hardware or if it is possible to disable the processing that is causing the issues.
- We can increase the bandwidth used or the sampling speed at the receivers end to allow for each signal to represent more than 8 bits. For example, using 1024 different frequencies can allow us to send 12 bits per signal increasing our speed by 50%.
- We can encode the data to allow for error recovery at the receiver