**Student: Kash Pritt**

**Class: CS 490**

**Date Created: 10/15**

**Engineering Notebook**

DATE 9/16

Got app cloned and ran on device per Dr.Liu’s requests

A screenshot of a video chat

Description automatically generated

DATE 9/26

Finished up the project proposal before submission

DATE 10/6

Made the Architecture model to show the remote system and the user gui along with the database. Files are loaded into the remote as they are loaded into the host, the transcriber whisper is represented separate rather than a class in salac due to it being a program that has been created

A diagram of a remote control

Description automatically generated

DATE 10/7

Created Data flow diagram to show how files move throughout the system

A diagram of a computer system

Description automatically generated

DATE 10/9

Remade the UML class model. Removed needless callback methods and added in SALAC

A diagram of a computer

Description automatically generated with medium confidence

DATE 10/17

Created a new gui design in gimp, simple to change design pattern to make changes listed by dr.liu. The new design incorporates file selector, an editor, buttons for transcription, a change to the transcribe readout where it now aligns with the spectrogram. Annotation zones capture things of interest such as callbacks, runways etc.

*A screenshot of a computer

Description automatically generated*

DATE: 10/19

Added a timestamp code callback to be updated on user file change or selection. The timestamp does not follow playtime of audio do to pydubs not having a method for is\_playing(), have to find other way to track audio playtime.

@callback(

    Output("timestamp","children"),

    Input("global-waveform", "selectedData"),

    Input("file-selector", "value")

)

def updateTimestamp(selection, sample\_audio):

    if not sample\_audio:

        raise PreventUpdate

    audio = AudioSegment.from\_file(sample\_audio)

    start = time.time()

    if selection is not None and "range" in selection.keys():

        left\_bound = selection["range"]["x"][0] \* 0.01

    else:

        left\_bound = 0

    current = time.strftime("%H:%M:%S", time.gmtime(time.time() % audio.duration\_seconds))

    audio\_length = audio.duration\_seconds

    left\_ms = int(left\_bound % 1 \*1000)

    left\_seconds = int(left\_bound % 60)

    left\_minutes = int((left\_bound/60)%60)

    left\_hours = int((left\_bound /3600)%60)

    #this represents the right sde of the timestamp for audio play

    right\_ms = int(audio\_length % 1 \* 1000)

    right\_seconds = int(audio\_length % 60)

    right\_minutes = int((audio\_length / 60) % 60)

    right\_hours = int((audio\_length / 3600) % 60)

    return f"{left\_hours:02d}:{left\_minutes:02d}:{left\_seconds:02d}:{left\_ms:02d}/{right\_hours:02d}:{right\_minutes:02d}:{right\_seconds:02d}.{right\_ms:03d}",

DATE 10/18

Worked on section 2 of SRD, finished up until point needing a USE CASE model to write scenarios.