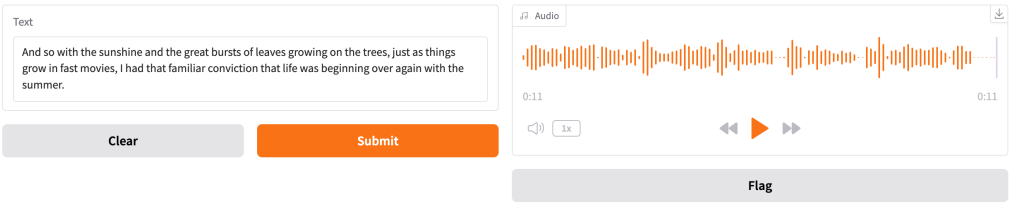
Project - Jingquan Li

Text-to-speech implementation

1. create virtual environment: `python -m venv coqui`
2. activate virtual environment: `source coqui/bin/activate`
3. install text-to-speech package: `pip install coqui-tts`
    a. reference: https://github.com/idiap/coqui-ai-TTS
    b. a deep learning toolkit for Text-to-Speech
    c. This repo is the forked version which fixed some bugs. The original version repo (https://github.com/coqui-ai/TTS) earns 34.9k stars in GitHub.
4. install gradio library: `pip install gradio`
    a. an open-source Python library that allows people to quickly create customizable user interfaces for machine learning models, APIs, or any Python function
    b. especially useful for showcasing machine learning models by providing a simple interface where users can input data (like text, images, or audio) and get the model's predictions as output, all within a web-based GUI
5. two forms of interaction: locally saved or graphic user interface (GUI)
    a. Locally saved
        i. run my provided code: `python tts.py`
        ii. the output audio will be saved in the "output" folder
        iii. I have included three example audios generated by the code in the "output" foler
    b. GUI form
        i. run my provided code: `python tts_GUI.py`
        ii. Here is one example quote from the book "The Great Gatsby": I have downloaded the audio file as "gatsby.wav". You can check out it directly in the "GUI" folder and listen to the audio. Or you can run

the code as instructed above and get the result on this graph user interface as shown below.



6. About the TTS model
    a. The model I use now is “tts_models/en/ljspeech/fast_pitch”
    b. If you want to try other models, just use command tts —list_models, then you will see the list of all models. There are different language modes to choose from, and they also include multilingual mode.
    c. Choose the model you prefer, and modify the corresponding model name in the code

```
tts = TTS(model_name='tts_models/en/ljspeech/fast_pitch').to(device)
```

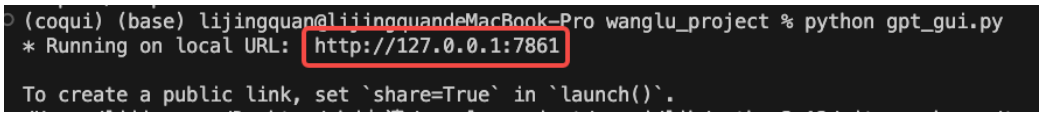
```
(coqui) (base) lijingquan@lijingquandeMacBook-Pro wanglu_project % tts --list_models

 Name format: type/language/dataset/model
   1: tts_models/multilingual/multi-dataset/xtts_v2 [already downloaded]
   2: tts_models/multilingual/multi-dataset/xtts_v1.1
   3: tts_models/multilingual/multi-dataset/your_tts
   4: tts_models/multilingual/multi-dataset/bark
   5: tts_models/bg/cv/vits
   6: tts_models/cs/cv/vits
   7: tts_models/da/cv/vits
   8: tts_models/et/cv/vits
   9: tts_models/ga/cv/vits
```

More exploration: Integrate LLM with TTS

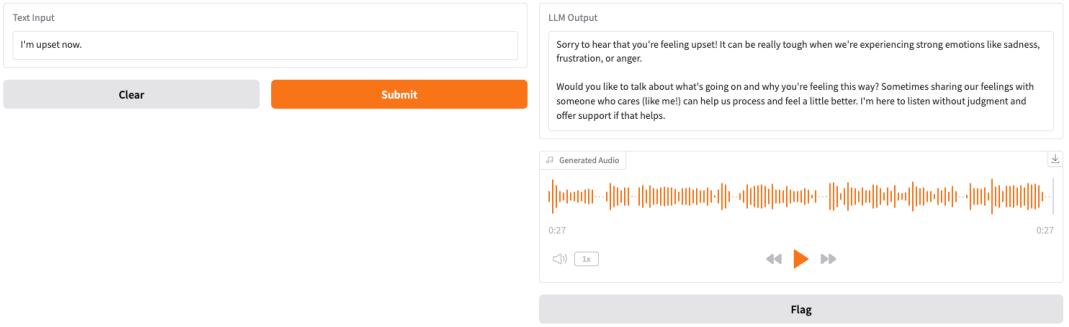
1. Furthermore, I tried to integrate LLM with TTS as the aim of the project is to use this system for helping depressed people communicate and work as a chatbot to deal with their frustration.
    a. The model used is an open source local LLM in github: https://github.com/nomic-ai/gpt4all
    b. install the package using `pip install gpt4all`
2. There are also two versions to choose from, GUI version (gpt_gui.py) or locally saved version (gpt.py).

a. locally saved version:
    i. modify text input in the function parameter text_input
    ii. modify output file path
    iii. use command `python gpt.py` to run the code
    iv. The text response generated by LLM will be converted to audio and saved in the LLM output folder
    v. I have included three generated example audio files in the LLM output folder.

b. GUI version:
    i. use command `python gpt_gui.py` to run the code
    ii. There will be a link shown in the terminal as below



    iii. Use cmd+click to open the link, the user interface is as below. You can input your ideas and the LLM output response in both text and audio forms will display on the right. You can listen to the audio online or save it locally.



Code file and audio file structure

```
code_Jingquan Li
├── GUI
```

```
│   ├── gatsby.wav
├── LLM output
│   │   ├── output1.wav
│   │   ├── output2.wav
│   │   └── output3.wav
├── outputs
│   │   ├── output.wav
│   │   ├── output2.wav
│   │   └── output3.wav
├── gpt_gui.py
├── gpt.py
├── tts_gui.py
└── tts.py
```