# **Project Statement: TechR Project 5: Deepfake detection**

# **Objectives:** The main objective of this project is to enable voice cloning using the Tortoise TTS library. Specifically, it aims to take a few short audio samples of a person's voice and use those samples to generate new speech audio in that cloned voice for any given text input.

**Scope:**

• April 3, 2024 - Start Date.

• April 5, 2024 – Research on various methods for project completion.

• April 9, 2024 – Actual coding complete.

• April 12, 2024 – Documentation and submission of project complete.  
  
**Methods:**

Here are the key objectives and steps involved:

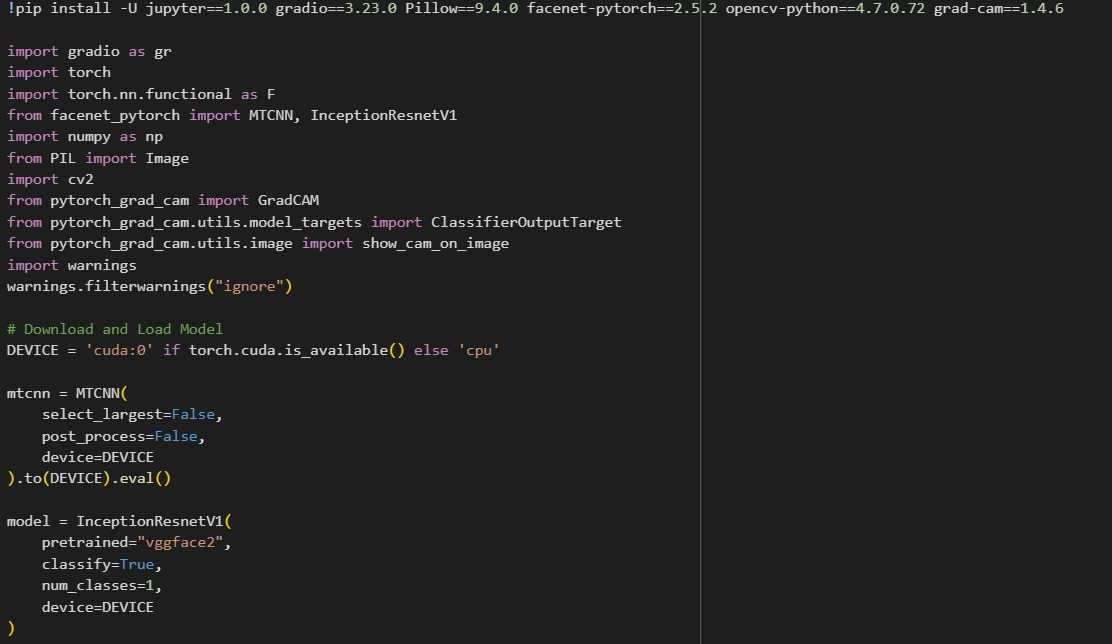
1. **Install dependencies:** The code starts by installing required Python packages like SciPy, Transformers, einops, rotary\_embedding\_torch, unidecode, and the Tortoise TTS library itself.
2. **Import necessary libraries:** It then imports various Python libraries and modules required for text-to-speech generation, audio processing, and file handling.
3. **Initialize TextToSpeech object:** An instance of the TextToSpeech class from the Tortoise API is created, which will be used for generating speech.
4. **Define input text:** The text to be spoken is specified in the text variable.
5. **Set speech quality preset**: The preset variable is set to "high\_quality", which determines the quality level of the generated speech (other options are "ultra\_fast", "fast", and "standard").
6. **Upload audio samples**: The user is prompted to upload at least two audio clips in WAV format, each 6-10 seconds long. These audio samples will be used to clone the custom voice.
7. **Load custom voice**: The uploaded audio samples are loaded into the   
   voice\_samples and conditioning\_latents variables using the load\_voice function from tortoise.
8. **Generate speech**: The tts\_with\_preset method from the TextToSpeech instance is called with the input text, loaded voice samples, conditioning latents, and the specified quality preset. This generates the speech audio in the cloned voice.
9. 6. **Save and play generated audio**: The generated speech audio is saved as a WAV file (generated-{CUSTOM\_VOICE\_NAME}.wav) and then played back using the IPython.display.Audio function.  
     
   **Screenshots of Work Done:**

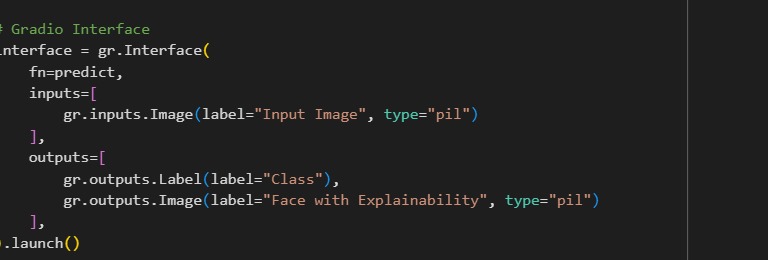
Fig 1: Installing all the dependencies required.

Fig 2: Code snippet of Gradio Interface

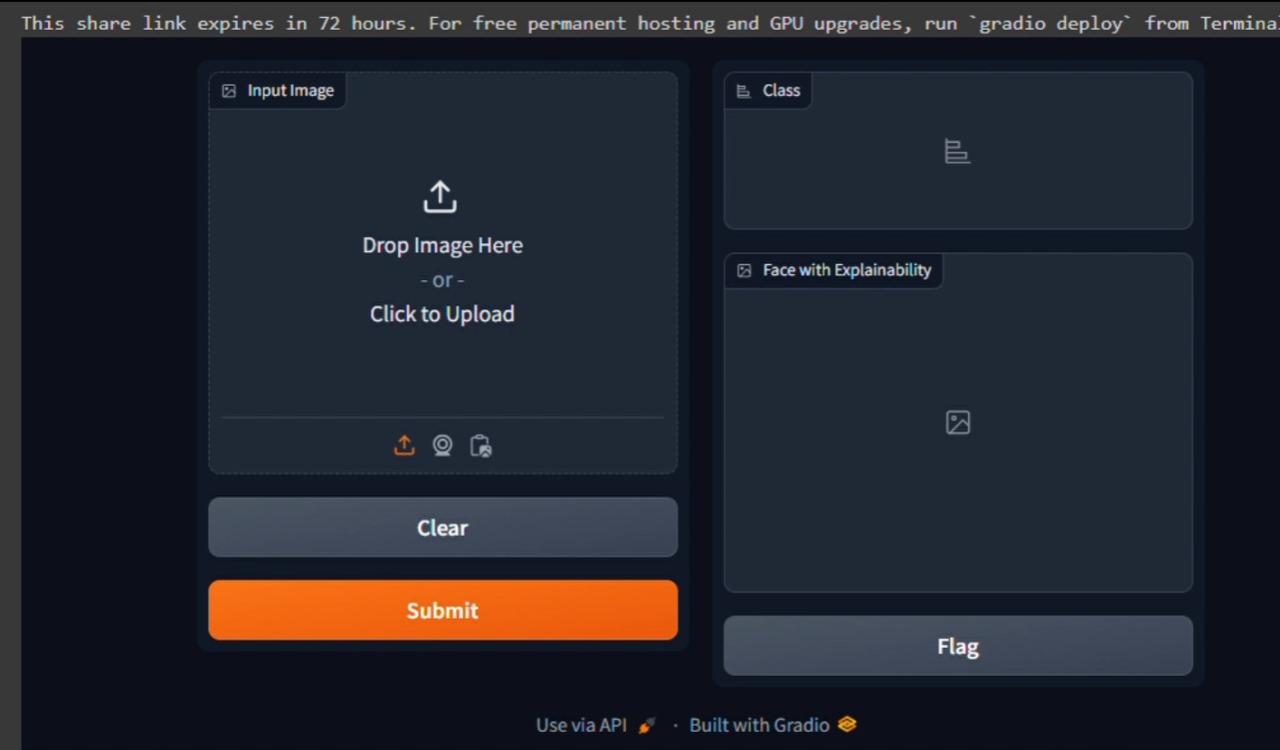


Fig 3: Interface developed using Gradio, where you can insert image and detect fake images.