**Prototype Documentation**

To implement a python script that takes audio files as input without using HTML, I would use the **argparse** module for handling command-line arguments, the **pydub** library for audio file manipulation, and the **speech\_recognization** library for speech-to-text conversion.

Below is a simple implementation of this script:

**“import argparse**

**from pydub import AudioSegment**

**import speech\_recognition as sr**

**def convert\_audio\_to\_text(audio\_file):**

**sound = AudioSegment.from\_file(audio\_file)**

**sound.export("temp.wav", format="wav")**

**r = sr.Recognizer()**

**with sr.AudioFile("temp.wav") as source:**

**audio = r.record(source)**

**try:**

**text = r.recognize\_google(audio)**

**return text**

**except sr.UnknownValueError:**

**return "Could not understand audio"**

**except sr.RequestError as e:**

**return f"Could not request results; {e}"**

**def main():**

**parser = argparse.ArgumentParser(description='Convert audio files to text')**

**parser.add\_argument('audio\_file', help='Path to the audio file to be converted')**

**args = parser.parse\_args()**

**result = convert\_audio\_to\_text(args.audio\_file)**

**print("Converted text: ", result)**

**If \_\_name\_\_ == '\_\_main\_\_':**

**main() ”**

This script takes an audio file as an argument, converts it to a temporary WAV file, and then uses Google’s speech recognition to convert the audio to text. Finally, it prints the converted text.

To use the script, the user would need to have Python installed, along with the required libraries. They would then need to run the script from the command line, providing the path to the audio file as an argument.

Accepted audio file formats include those supported by the **pydub** library such as WAV, MP3, and more. Users can print the results by running the script and viewing the converted text that is outputted.

Challenges may arise from incompatible or unsupported audio formats, large file sizes, and noisy or poor quality recordings, which can affect the accuracy of the speech-to text conversation. Error handling for these cases should be robust to ensure the script runs smoothly.