Table of Contents

python # Assume ‘gemini\_model’ is a pre-configured instance of a Gemini model client.

def get\_gemini\_response(text\_prompt: str): “““Get response from Gemini AI model”“” print(f”[Gemini] Generating response for prompt: {text\_prompt}“) try: response = gemini\_model.generate\_content(text\_prompt) print(”[Gemini] Response generated successfully”) return response.text except Exception as e: print(f”[ERROR] Gemini generation failed: {str(e)}“) return f”Error: {str(e)}”

# 1 Example usage:

prompt = “What are the main benefits of using Python for web development?” response = get\_gemini\_response(prompt)

if not response.startswith(“Error:”): print(“AI Response:”) print(response) else: print(f”Failed to get response: {response}“)

python import pyttsx3 import os

# 2 Assume generate\_tts\_audio is defined as in the snippet

def generate\_tts\_audio(text\_to\_speak, output\_file\_path): “““Generate TTS audio file using pyttsx3”“” print(f”[TTS] Generating speech for text: {text\_to\_speak}“) if not text\_to\_speak: return False

try:  
 engine = pyttsx3.init()  
 engine.setProperty('rate', 150) # Adjust speech rate  
 engine.save\_to\_file(text\_to\_speak, output\_file\_path)  
 engine.runAndWait()  
 print(f"[TTS] Audio saved at: {output\_file\_path}")  
  
 if os.path.exists(output\_file\_path) and os.path.getsize(output\_file\_path) > 0:  
 return True  
 return False  
except Exception as e:  
 print(f"[TTS ERROR] {str(e)}")  
 if os.path.exists(output\_file\_path):  
 try: os.remove(output\_file\_path)  
 except: pass  
 return False

# 3 Example usage:

text = “Hello, world! This is a test of the text to speech conversion.” output\_file = “hello\_world.mp3”

success = generate\_tts\_audio(text, output\_file)

if success: print(f”Successfully created audio file at ‘{output\_file}’“) # Clean up the created file os.remove(output\_file) else: print(”Failed to create the audio file.”)

python from flask import Flask, render\_template import os

app = Flask(**name**)

# 4 Ensure a ‘templates’ directory exists for the example

if not os.path.exists(‘templates’): os.makedirs(‘templates’)

# 5 Create a simple index.html file for the example

with open(‘templates/index.html’, ‘w’) as f: f.write(’<!DOCTYPE html>

Home

Welcome to the Index Page!

’)

@app.route(‘/’) def index(): ““” Handles requests for the index page. ““” print(“[ROUTE] Index page requested”) return render\_template(‘index.html’)

if **name** == ‘**main**’: # When you run this script and navigate to http://127.0.0.1:5000/ # the console will print “[ROUTE] Index page requested” # and the browser will display the content of index.html. app.run(debug=True)

python import requests import urllib.parse

# 6 The URL of the Flask endpoint

url = “http://127.0.0.1:5000/process-full-audio”

# 7 Path to the audio file you want to send

audio\_file\_path = ‘path/to/your/query.webm’

try: with open(audio\_file\_path, ‘rb’) as f: # The key ‘audio\_blob’ must match what the server expects files = {‘audio\_blob’: (audio\_file\_path, f, ‘audio/webm’)}

response = requests.post(url, files=files)  
  
if response.status\_code == 200:  
 # Save the returned audio response  
 with open('response.wav', 'wb') as out\_f:  
 out\_f.write(response.content)  
   
 print("Successfully received and saved response.wav")  
   
 # Access and decode the custom headers  
 transcription = urllib.parse.unquote(response.headers.get('X-Transcription', ''))  
 gemini\_response = urllib.parse.unquote(response.headers.get('X-Gemini-Response', ''))  
   
 print(f"\n--- Metadata ---")  
 print(f"Transcription: {transcription}")  
 print(f"Gemini Response: {gemini\_response}")  
   
else:  
 print(f"Error: {response.status\_code}")  
 print(response.json())

except requests.exceptions.RequestException as e: print(f”An error occurred: {e}“)

bash # Send a POST request with a JSON payload # The audio output is saved to response.wav # The response headers are printed to the console curl -X POST  
-H “Content-Type: application/json”  
-d ‘{“text”: “What is the weather like on Mars?”}’  
-i  
-o response.wav  
http://127.0.0.1:5000/process-text

# 8 Expected Console Output (Headers):

# 9 HTTP/1.1 200 OK

# 10 Server: Werkzeug/2.0.1 Python/3.9.1

# 11 Date: Mon, 01 Jan 2024 12:00:00 GMT

# 12 Content-Type: audio/wav

# 13 Content-Length: 123456

# 14 X-Gemini-Response: The%20weather%20on%20Mars%20is%20typically%20cold%20and%20dry…

# 15 …

# 16 You can now play the ‘response.wav’ file.