TTS Module – Documentation

Student: Fatima Zahid
Roll Number: BSCS51F22S010
Project: TTS Module – Google TTS + Flask

1. Introduction

The goal of this module is to convert textual data into speech using Python, specifically using Google Text-to-Speech (gTTS) library, and make it accessible through a Flask-based API. This allows the module to be integrated into other applications where dynamic text needs to be converted into audio.

2. Tools and Libraries Used

Tool / Library	Version / Notes	Purpose
Python	3.x	Main programming language
Flask	Latest	Lightweight web framework for creating API endpoints
gTTS (Google Text-to-Speech)	Latest	Converts text input into audio (MP3)
playsound	Latest	Optional library to play generated audio locally
os	Built-in	For file handling and path management

3. Google TTS (gTTS)

Google Text-to-Speech (gTTS) is a Python library that interfaces with the Google Translate TTS API.

Why use Google TTS?

1. Ease of use: Very simple syntax to convert text to audio.
2. Multiple languages: Supports dozens of languages including Urdu, English, Arabic, etc.
3. High-quality speech output: Natural-sounding voice.
4. Integration-friendly: Can generate audio files on-the-fly for web applications.

Basic Usage Example

```
from gtts import gTTS

text = "Hello, welcome to the TTS module"
tts = gTTS(text=text, lang='en')
tts.save("output.mp3")
```

- `text`: The text to convert.
- `lang`: Language code (e.g., 'en' for English, 'ur' for Urdu).
- `save()`: Saves the audio file locally.

---

4. Flask API

Flask is used to expose the TTS functionality as a REST API. Other applications can send text to your Flask server and get back an audio file without needing direct access to your TTS code.

Why Flask is used?

1. Lightweight: No heavy setup, suitable for small APIs.
2. Rapid development: Easy to define endpoints for different services.
3. Integration: Can be easily integrated with web apps, mobile apps, or other APIs.

Basic Flask API Example

```
from flask import Flask, request, send_file
from gtts import gTTS
```

```
import os

app = Flask(__name__)

@app.route('/tts', methods=['POST'])
def tts_service():
    data = request.get_json()
    text = data.get('text', '')

    if not text:
        return {"error": "No text provided"}, 400

    # Generate TTS
    tts = gTTS(text=text, lang='en')
    filename = "output.mp3"
    tts.save(filename)

    return send_file(filename, mimetype="audio/mpeg")

if __name__ == "__main__":
    app.run(debug=True)
```

Explanation:

1. `@app.route('/tts', methods=['POST'])`: Creates an endpoint `/tts` that accepts POST requests.
2. `request.get_json()`: Reads JSON data sent by the client.
3. `gTTS(text, lang='en')`: Converts received text into speech.
4. `send_file()`: Sends the generated MP3 file back to the client.

---

5. Project Workflow

1. **Input:** Text is provided either manually or through JSON requests.
2. **Processing:**

    - Flask receives the text via POST request.
    - gTTS converts the text to audio (MP3 format).
3. **Output:** Audio file is returned to the client.
4. **Optional Playback:** Locally using `playsound` or web app can play audio directly.

Flow Diagram:

Client (JSON text) ---> Flask API ---> gTTS processes ---> MP3 file ---> Client receives audio

---

6. JSON Request Example

Request Body (JSON):

```
{
   "text": "Hello, this is Fatima Zahid's TTS module."
}
```

Response:

- An MP3 audio file named `output.mp3` containing the spoken version of the text.

7. Advantages of This Module

1. Language Flexibility: Supports multiple languages.
2. Scalable: Can be integrated into larger projects or web apps.
3. Ease of Use: Simple API endpoint for dynamic text-to-speech conversion.
4. Automation-Friendly: Can be automated for batch processing or real-time applications.

8. Future Improvements

- **KG & NLU Integration**: Enable context-aware text generation via TTS.
- **Output Management**: Store and log audio outputs for easy retrieval.
- **Automation & Deployment**: Run server reliably with error handling and logging.
- **Enhancements**: Connect TTS with front-end or chatbot interfaces for seamless interaction.