SIMPLIFYING TEXT TO SPEECH CONVERSION

A PROJECT REPORT

Submitted by

ARUN C 92132223017
BALAVIGNESHWARAN M 92132223026

MINI-PROJECT: SIMPLIFYING TEXT TO SPEECH CONVERSION

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

PSNA COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution, Affiliated to Anna University, Chennai)

DINDIGUL - 624622

OCTOBER 2024

PSNA COLLEGE OF ENGINEERING AND TECHNOLOGY,

(Autonomous Institution Affiliated to Anna University, Chennai)

DINDIGUL – 624622

BONAFIDE CERTIFICATE

Certified that this idea report "SIMPLIFYING TEXT TO SPEECH CONVERSION

" is the bonafide work of **"ARUN C (92132223017), BALAVIGNESHWARAN M (92132223026)**" who carried out the idea work under my supervision.

SIGNATURE	SIGNATURE
Dr. A. VINCENT ANTONY KUMAR, M.E, Ph.D., **HEAD OF THE DEPARTMENT** PROFESSOR & HEAD DEPARTMENT OF IT PSNA COLLEGE OF ENGINEERING TECHNOLOGY, DINDIGUL -624622	**Dr. P. PRIYADHARSHINI M.E,** ASSISTANT PROFESSOR DEPARTMENT OF IT PSNA COLLEGE OF ENGINEERING TECHNOLOGY, DINDIGUL -624622

Submitted for the idea on_____

ABSTRACT:

text-to-Speech (TTS) systems enable machines to convert written text into spoken words, providing an essential bridge between text-based and audio-based communication. While modern TTS solutions such as Google's WaveNet and OpenAI's GPT-powered models provide high-quality and realistic voices, they often require significant computational resources and technical expertise to implement. This project proposes a simplified approach to TTS conversion, using readily available Python libraries like gTTS (Google Text-to-Speech) and pygame. The goal is to create a lightweight, easy-to-implement system that generates spoken output from text input while minimizing the need for complex setups.
The implementation focuses on providing a basic yet functional solution that can be used for small-scale applications or as a foundation for more advanced TTS models. The project handles text input, converts it into speech, saves the audio file in an MP3 format, and plays it back using pygame or other sound libraries. By leveraging a high-level API such as gTTS, this project simplifies the complexities associated with traditional TTS models, making it more accessible to beginners and developers looking for quick and reliable TTS function

INTRODUCTION:

Text-to-Speech (TTS) systems have evolved significantly in recent years, enabling machines to convert written text into natural, intelligible speech. These systems are crucial in applications such as virtual assistants, accessibility tools for the visually impaired, and interactive voice-response systems. The aim of this project is to implement a simplified TTS model, utilizing readily available Python libraries, to demonstrate a functional solution that can convert text into speech. By leveraging the gTTS (Google Text-to-Speech) API, we offer a basic but effective solution for TTS that can be expanded further into more sophisticated models like Tacotron or WaveNet**.**

PROBLEM STATEMENT:

While modern TTS systems like Google's WaveNet and Tacotron 2 produce high-quality, human-like speech, they require significant computing power and deep learning expertise to implement. For beginners and small-scale projects, integrating these models is complex. This project aims to simplify TTS by creating a lightweight, easy-to-implement solution that provides acceptable results for general text-to-speech conversion tasks.

CHALLENGES:

Audio Device Compatibility: On some Linux systems, playing audio using pygame or ALSA may result in device errors (e.g., "ALSA: Couldn't open audio device"). This required installing and configuring PulseAudio or using alternative playback libraries.

Platform Differences: Ensuring that the code runs smoothly across different operating systems (Windows, macOS, Linux) posed challenges due to the differences in audio system handling.

PROPOSED MODEL:

The proposed model is built using the gTTS library to convert input text into speech, with the speech saved as an audio file. The system includes the following steps:

1. **Text Input**: User provides a string of text that they wish to convert into speech.
2. **Text Preprocessing**: Basic formatting of the input text.
3. **Speech Synthesis**: Using gTTS, the system converts the text into an MP3 audio file.
4. **Audio Playback**: The generated MP3 file is played back using pygame or other Python-based libraries.

SOURCE CODE:

MOBILE APPLICATION (CODE)

```
 import 'package:flutter/material.dart';
import 'package:flutter_tts/flutter_tts.dart';

void main() {
 runApp(const MyApp());
}

class MyApp extends StatefulWidget {
 const MyApp({super.key});

 @override
 _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
 FlutterTts flutterTts = FlutterTts();
 final TextEditingController _controller = TextEditingController();

 Future<void> _speak() async {
  String text = _controller.text;
  if (text.isNotEmpty) {
   await flutterTts.setLanguage("en-US");
   await flutterTts.setPitch(1.0);
```

```
    await flutterTts.setSpeechRate(0.5);
    await flutterTts.speak(text);
  }
}

@override
Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    home: Scaffold(
      appBar: AppBar(
        title: const Text("Text to Speech"),
        centerTitle: true,
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            TextField(
              controller: _controller,
              decoration: const InputDecoration(
                labelText: "Enter text",
                enabledBorder: OutlineInputBorder(
                  borderSide: BorderSide(
                    color: Colors.blueAccent,
                    width: 2.0,
                  ),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: BorderSide(
                    color: Colors.redAccent,
                    width: 3.0,
                  ),
                ),
              ),
            ),
            const SizedBox(height: 20),
            ElevatedButton(
              onPressed: _speak,
              child: const Text("Convert to Speech"),
            ),
          ],
        ),
      ),
```

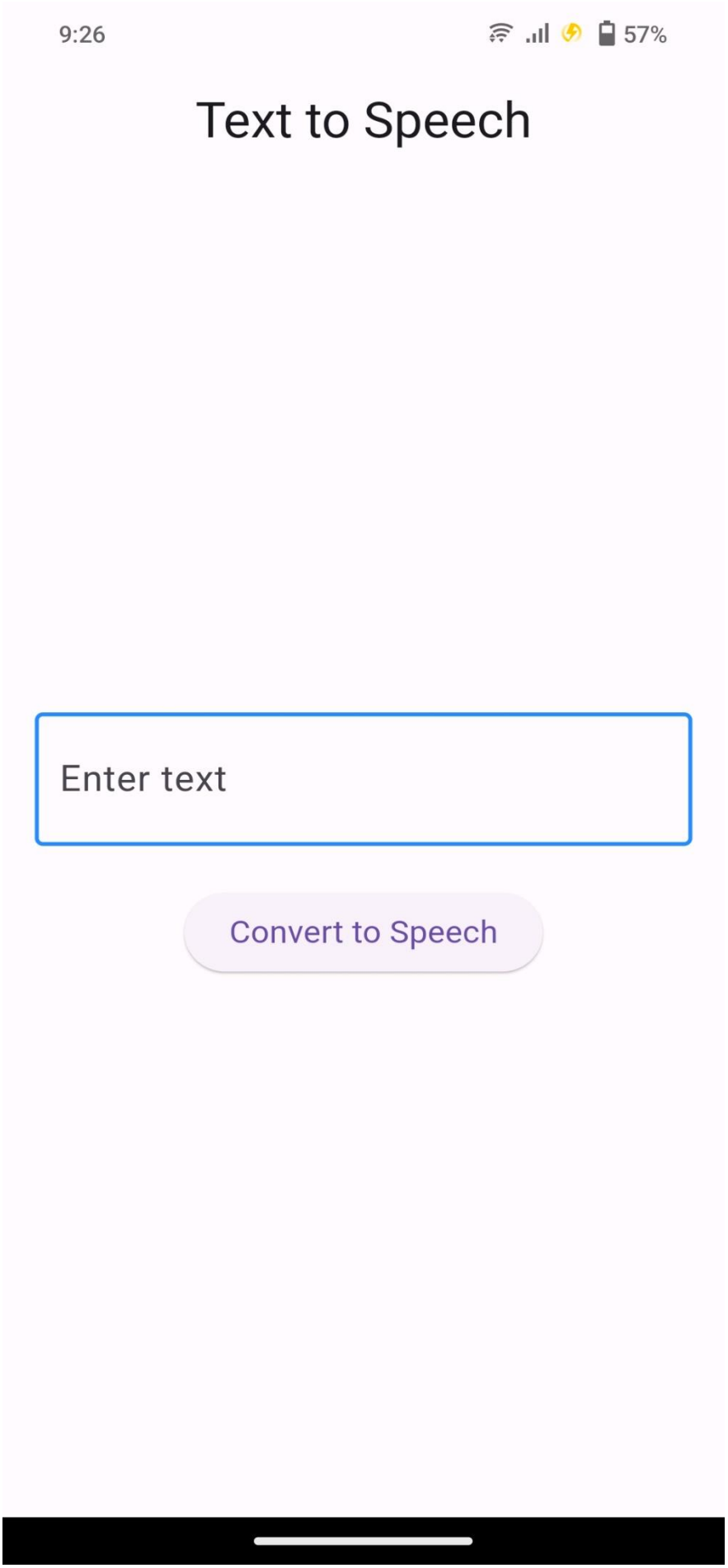
```
      ),
    );
  }
}
```

BACKEND CODE:

```
from gtts import gTTS
import pygame
import os

def text_to_speech(text, lang='en', slow=False):
    """
    Convert text to speech using Google Text-to-Speech (gTTS)
    and play the sound using pygame.

    Parameters:
    text (str): The text to convert to speech.
    lang (str): The language for the speech (default is English).
    slow (bool): If True, the speech will be slower.
    """
    try:
        # Create gTTS object
        tts = gTTS(text=text, lang=lang, slow=slow)

        # Save the speech to an MP3 file
        filename = "speech_output.mp3"
        tts.save(filename)
        print(f"Speech saved to {filename}")

        # Initialize pygame mixer
        pygame.mixer.init()

        # Load and play the saved MP3 file
        pygame.mixer.music.load(filename)
        pygame.mixer.music.play()

        # Wait for the speech to finish playing
        while pygame.mixer.music.get_busy():
            continue
```

```
        # Optional: Remove the MP3 file after playing
        os.remove(filename)

    except Exception as e:
        print(f"An error occurred: {e}")

# Example usage
if __name__ == "__main__":
    text = "Hello, this is a test for playing audio using pygame."
    text_to_speech(text)
```

OUTPUT:


9:26
57%
Text to Speech
Enter text
Convert to Speech

CONCLUSION:

This project successfully demonstrates a simplified approach to TTS conversion using Python and the gTTS library. Although it is a lightweight solution compared to advanced models like Tacotron or WaveNet, it offers a practical and accessible implementation for general use. The system works well across platforms and can be expanded further with features such as real-time processing and advanced voice customization. Through this project, we provide a foundation for building more sophisticated TTS systems in the future.

Simplifying text-to-speech conversion.

by

ARUN C

BALAVIGNESHVARAN M

ABSTRACT:

- Text-to-Speech (TTS) technology converts written text into spoken words using voice synthesis.
- The project aims to simplify TTS implementation while enhancing the naturalness and accuracy of speech.
- Advanced models like WaveNet and Tacotron are used for more human-like voice output.
- The simplified TTS system can benefit applications like virtual assistants, assistive technologies, and automated content creation.
- The goal is to make TTS more accessible, efficient, and user-friendly for broader use cases.

PROBLEM STATEMENT:

- Traditional Text-to-Speech (TTS) systems are complex and difficult to implement.
- Synthesized speech often lacks naturalness and expressiveness.
- Existing TTS systems may struggle to support multiple languages and diverse voice outputs.

high resource consumption and latency in real-time applications limit user experience

.

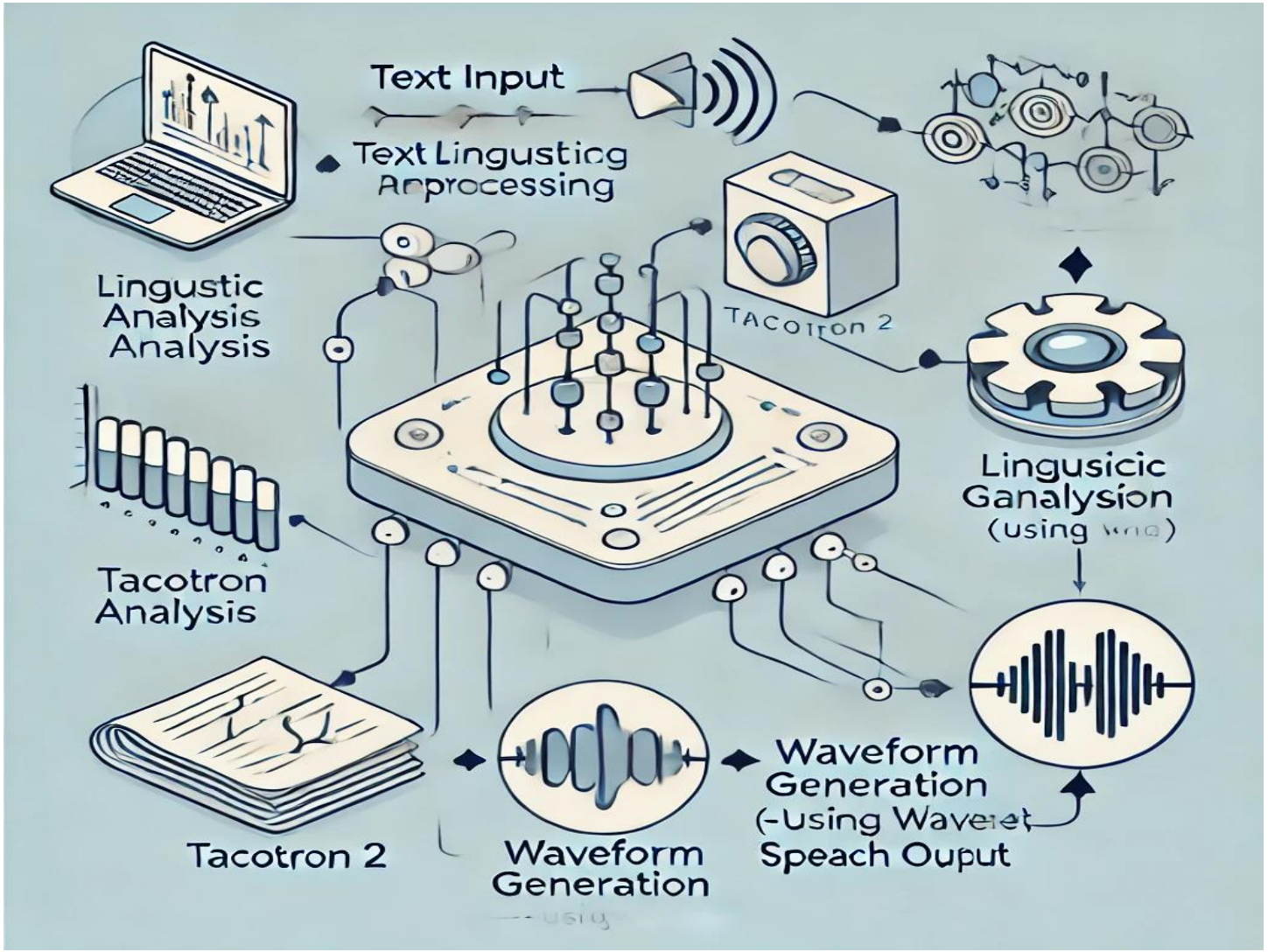
CHALLENGES:

- **Naturalness of Speech**: Achieving human-like, expressive, and natural-sounding synthesized voices.
- **Language Diversity**: Handling multiple languages, accents, and dialects effectively in a single TTS system.
- **Resource Efficiency**: Reducing computational complexity, memory usage, and latency for real-time applications.
- **User Accessibility**: Creating a user-friendly and easily integrable TTS system for developers and end-users.

PROPOSED MODEL:

- **Lightweight and Optimized**:
  Optimize the model to reduce memory usage and computation time, enabling real-time performance even on low-resource devices.
- **User-Friendly Interface**:
  Design a simple API or interface that allows developers to integrate the TTS system with minimal effort.
- **Cloud Integration**:
  Provide optional cloud-based processing for resource-intensive tasks, making it accessible across multiple devices and platforms.

DIAGRAM:


Text Input
Text Lingustic
Anprocessing
Lingustic
Analysis
Analysis
TACOTron 2
Tacotron
Analysis
Lingusicic
Ganalysion
(using
Tacotron 2
Waveform
Generation
Waveform
Generation
(-using Wavelet
Speach Ouput

APPLICATION (CODE)

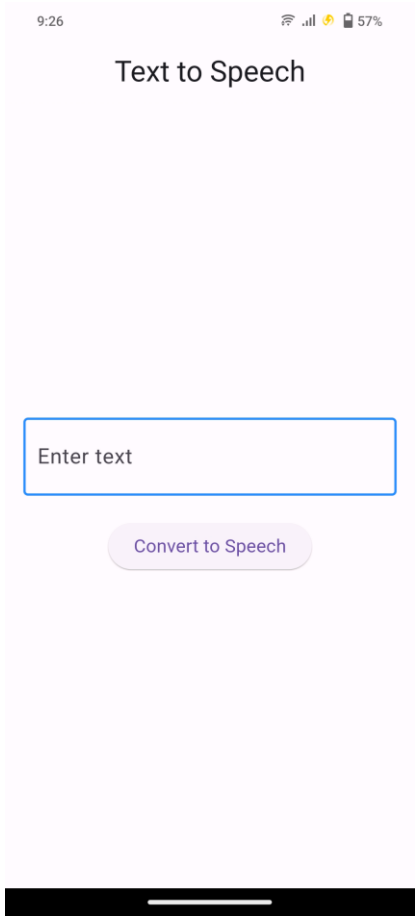
```dart
import 'package:flutter/material.dart';
import 'package:flutter_tts/flutter_tts.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatefulWidget {
  const MyApp({super.key});

  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  FlutterTts flutterTts = FlutterTts();
  final TextEditingController _controller = TextEditingController();

  Future<void> _speak() async {
    String text = _controller.text;
    if (text.isNotEmpty) {
      await flutterTts.setLanguage("en-US");
      await flutterTts.setPitch(1.0);
      await flutterTts.setSpeechRate(0.5);
      await flutterTts.speak(text);
    }
  }
```

```dart
@override
Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    home: Scaffold(
      appBar: AppBar(
        title: const Text("Text to Speech"),
        centerTitle: true,
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            TextField(
              controller: _controller,
              decoration: const InputDecoration(
                labelText: "Enter text",
                enabledBorder: OutlineInputBorder(
                  borderSide: BorderSide(
                    color: Colors.blueAccent,
                    width: 2.0,
                  ),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: BorderSide(
                    color: Colors.redAccent,
                    width: 3.0,
                  ),
                ),
```

```python
from gtts import gTTS
import pygame
import os

def text_to_speech(text, lang='en', slow=False):
    """
    Convert text to speech using Google Text-to-Speech (gTTS)
    and play the sound using pygame.

    Parameters:
    text (str): The text to convert to speech.
    lang (str): The language for the speech (default is English).
    slow (bool): If True, the speech will be slower.
    """
    try:
        # Create gTTS object
        tts = gTTS(text=text, lang=lang, slow=slow)

        # Save the speech to an MP3 file
        filename = "speech_output.mp3"
        tts.save(filename)
        print(f"Speech saved to {filename}")
```

```
            # Initialize pygame mixer
            pygame.mixer.init()

            # Load and play the saved MP3 file
            pygame.mixer.music.load(filename)
            pygame.mixer.music.play()

            # Wait for the speech to finish playing
            while pygame.mixer.music.get_busy():
                continue

            # Optional: Remove the MP3 file after playing
            os.remove(filename)

        except Exception as e:
            print(f"An error occurred: {e}")

# Example usage
if __name__ == "__main__":
    text = "Hello, this is a test for playing audio using pygame."
    text_to_speech(text)
```

OUTPUT



CONCLUSION:

- The project successfully demonstrates a simplified approach to Text-to-Speech (TTS) conversion by leveraging advanced models like Tacotron 2 and WaveNet. This approach significantly improves the naturalness, expressiveness, and accuracy of synthesized speech. The proposed system is more efficient and user-friendly, enabling seamless integration into various applications like virtual assistants, assistive technologies, and content creation platforms. Future enhancements could focus on supporting more languages, improving real-time performance, and expanding customization options, making TTS technology more accessible and impactful across industries.