**Mini Project Report: Audio**

**Calculator in Python**

**Title Page**



* Project Title: Audio Calculator
* Your Name: [Yashik srinivas]

**Abstract**

* The Audio Calculator is a Python-based application that allows users to perform basic arithmetic operations through a graphical user interface (GUI) with audio feedback. The project leverages the GTTS (Google Text-to-Speech) library for audio output and Tkinter for GUI development, Pygame for developing multimedia applications, Os for creating the interactions with the operating system. The tool aims to provide a user-friendly and accessible experience for individuals who prefer or require auditory interaction.

**Table of Contents**

**s.no page.no**

1. Introduction 03
2. Methodology 04-05
3. System Design 05-06
4. Results and Analysis 06-07
5. Conclusion 07
6. References 07
7. Appendices 07-12

**1. Introduction** 1.1 Problem Statement

* Calculators often lack accessibility features for visually impaired individuals. This project addresses the need for an audio-enabled calculator.
* 1.2 Objectives
* Develop a calculator with a user-friendly GUI and audio feedback.
* Use Python libraries to achieve seamless audio integration.
* 1.3 Scope of the Project
* Includes basic arithmetic operations: addition, subtraction, multiplication, and division.
* Supports audio feedback for results.
* 1.4 Relevance
* Enhances accessibility in calculator tools and demonstrates the integration of audio with Python applications.

**2. Methodology** 2.1 Tools and Technologies Used

* **Programming Language:** Python
* **Libraries:**
  + **gTTS:** Converts text to speech for audio feedback.
  + **Tkinter:** Builds the graphical user interface.
  + **Os:** provides a way to interact with the operating system
  + **Pygame :** Designed for development of multimedia applications
* **Other Tools:** Python IDLE or any Python-compatible IDE for development ,Visual studio code

2.2 Implementation Steps

* Step 1: Install required libraries using pip install gtts and ensure Tkinter , pygame , os is included (default with Python).
* Step 2: Design the GUI layout with Tkinter.
* Step 3: Implement arithmetic operations as Python functions.
* Step 4: Integrate gTTS to generate audio responses for calculation results.
* Step 5: Test the application for accuracy and usability.

**3. System Design** 3.1 Architectural Diagram

* The system consists of three main components:
  1. User Input via GUI
  2. Backend Calculation Logic
  3. Audio Output through gTTS

3.2 Flowchart

**Start**

**|**

**User Inputs Values via GUI**

**|**

**Select Operation (+, -, \*, /)**

**|**

**Perform Calculation**

**|**

**Generate Audio Feedback with gTTS**

**|**

**Display Result on GUI**

**|**

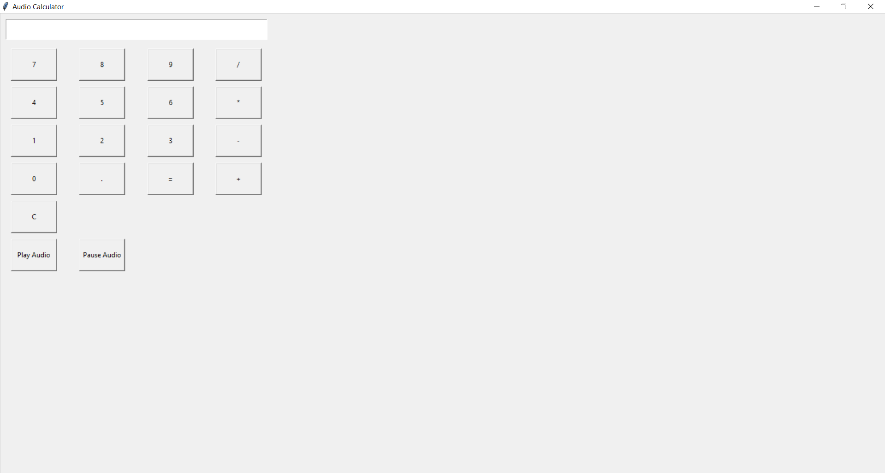
**End**

3.3 GUI Layout

* A Tkinter-based interface with entry fields for input, buttons for operations, and an output display area.

**4. Results and Analysis** 4.1 Outputs

* Screenshot of the GUI.



* Example audio output: “The result is .”
* 4.2 Performance Analysis
* The application successfully integrates GUI and audio feedback, delivering accurate results.
* 4.3 Challenges Encountered
* Synchronizing GUI updates with audio output.
* Handling exceptions such as division by zero.

**5. Conclusion**

* The Audio Calculator is a functional and accessible tool that combines GUI design with audio feedback. Future improvements could include voice input capabilities and support for more complex calculations.

**6. References**

* [GTTS Documentation](https://gtts.readthedocs.io/)
* [Tkinter Documentation](https://docs.python.org/3/library/tkinter.html)
* [Python Official Documentation](https://www.python.org/doc/)

**7. Appendices**

* **Source Code:**

**import tkinter as tk**

**from tkinter import messagebox**

**from gtts import gTTS**

**import pygame**

**import os**

**class AudioCalculator:**

**def \_\_init\_\_(self, master):**

**self.master = master**

**master.title("Audio Calculator")**

**master.geometry("400x600")**

**# Display**

**self.display = tk.Entry(master, width=30, justify='right', font=('Arial', 20))**

**self.display.grid(row=0, column=0, columnspan=4, padx=10, pady=10)**

**# Buttons**

**buttons = [**

**'7', '8', '9', '/',**

**'4', '5', '6', '\*',**

**'1', '2', '3', '-',**

**'0', '.', '=', '+'**

**]**

**# Create buttons**

**row = 1**

**col = 0**

**for button in buttons:**

**cmd = lambda x=button: self.click(x)**

**tk.Button(master, text=button, width=10, height=3, command=cmd).grid(row=row, column=col, padx=5, pady=5)**

**col += 1**

**if col > 3:**

**col = 0**

**row += 1**

**# Clear Button**

**tk.Button(master, text='C', width=10, height=3, command=self.clear).grid(row=row, column=col, padx=5, pady=5)**

**# Audio Control Buttons**

**self.play\_btn = tk.Button(master, text='Play Audio', width=10, height=3, command=self.play\_audio)**

**self.play\_btn.grid(row=row+1, column=0, padx=5, pady=5)**

**self.pause\_btn = tk.Button(master, text='Pause Audio', width=10, height=3, command=self.pause\_audio)**

**self.pause\_btn.grid(row=row+1, column=1, padx=5, pady=5)**

**# Initialize pygame mixer**

**pygame.mixer.init()**

**# Audio file path**

**self.audio\_file = 'calculator\_output.mp3'**

**def click(self, key):**

**if key == '=':**

**try:**

**result = eval(self.display.get())**

**self.display.delete(0, tk.END)**

**self.display.insert(0, str(result))**

**# Generate audio for the result**

**self.generate\_audio(str(result))**

**except:**

**messagebox.showerror("Error", "Invalid Input")**

**else:**

**self.display.insert(tk.END, key)**

**def clear(self):**

**self.display.delete(0, tk.END)**

**# Stop and unload any playing audio**

**pygame.mixer.music.stop()**

**def generate\_audio(self, text):**

**try:**

**# Remove existing audio file if it exists**

**if os.path.exists(self.audio\_file):**

**os.remove(self.audio\_file)**

**# Convert text to speech**

**tts = gTTS(text=f"The result is {text}", lang='en')**

**tts.save(self.audio\_file)**

**except Exception as e:**

**messagebox.showerror("Audio Error", str(e))**

**def play\_audio(self):**

**try:**

**# Check if audio file exists**

**if os.path.exists(self.audio\_file):**

**pygame.mixer.music.load(self.audio\_file)**

**pygame.mixer.music.play()**

**else:**

**messagebox.showinfo("Audio", "No audio available. Calculate something first.")**

**except Exception as e:**

**messagebox.showerror("Play Error", str(e))**

**def pause\_audio(self):**

**pygame.mixer.music.pause()**

**def main():**

**root = tk.Tk()**

**calculator = AudioCalculator(root)**

**root.mainloop()**

**if \_\_name\_\_ == "\_\_main\_\_":**

**main()**