A MOBILE APPLICATION TO IMPROVE ONLINE LEARNABILITY TO VISUALLY IMPAIRED ELEMENTARYSCHOOL CHILDREN

Project Id: TMP-23-310

Project Proposal Report

Dias S.T.D

BSc (Hons) in Information Technology Specializing in InformationTechnology

Department of Information Technology

Sri Lanka Institute of Information Technology Sri Lanka

March 19th

A MOBILE APPLICATION TO IMPROVE ONLINE LEARNABILITY TO VISUALLY IMPAIRED ELEMENTARYSCHOOL CHILDREN

Project Id: TMP-23-310

Project Proposal Report

BSc (Hons) in Information Technology Specializing in InformationTechnology

Department of Information Technology

Sri Lanka Institute of Information Technology Sri Lanka

March 19th

Declaration of the Candidate and the Supervisor

I declare that this is my own work, and this dissertation does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any other university or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Name	Student ID	Signature
Dias S.T. D	IT20247836	

The supervisor/s should certify the proposal report with the following declaration.

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Sathira .   02/05/2023

……………………………………..   ………………………………

Signature of the supervisor: Date:

Abstract

Many visually impaired students today struggle to access course materials like their peers do in the digital age. Students with disabilities, particularly those who are visually impaired, can benefit greatly from online learning. We'll talk about why it's crucial to make online education accessible to students who are blind and how to do it. We want to make learning in the primary stream easier for primary students who are blind or visually impaired. They can learn new information and broaden their knowledge using this intelligent mobile device application. A tutor recommendation module based on the student's knowledge level, a voice calculator and virtual tutor, a brain-improving game module, and a helpful voice bot are all included in the application. As my part, I will be doing the Simple calculator and schedule a meeting with the virtual tutor. The calculator has two options: voice base calculator and touch base calculator. The system will provide a touch system to identify the numbers and calculations. The screen is divided into two main sections, one relating to numbers and the other with calculations. Calculator can do addition, subtraction, multiplication, and division mathematical calculations. A voice-based calculator can provide an accessible way for people with visual impairments or disabilities to perform calculations. Since they may have difficulty using traditional calculators, a voice-based calculator can serve as an alternative means of access. Further, I will create an accessibility mode and incorporate it into the user interface design.

Table of Contents

List of figures

List of Tables

1. Introduction

1.1 Background Literature

Calculator is one of the most important devices that people use in their everyday life. But now-a-days, the purpose of hand-held calculators is mostly being served by calculator applications embedded in smart-phones. Moreover, with the advancement of speech-recognition technology, many of these applications are now providing voice-command support along with auditory output. Therefore, people can use their voice for calculation instead of pressing buttons. This voice command supported calculator is helpful to people who are not able to see or use their hands. For resolving this problem in 2018 peoples are implemented Bangla Speech Recognition in Voice Input Speech Output (VISO) Calculator.

In 2019 developed calculator for blind persons called Edu Braille Edu Braille can bring up braille codes letter to a series of words. But this tool cannot make a sound. From these shortcomings, the writer wants to add sound features as a correction and additional display on an Android smartphone to make it easier for braille instructors to monitor learning. Braille keyboard with sound output as a learning tool for blind visualizes. The Braille keyboard that has been made has a Braille code so that blind people can find out the letters on the keyboard. This tool does have a lack of sound output because the software and audio amplifiers have not been done. From these shortcomings, the writer wants to add a sound output feature when the keyboard is pressed to make it easier for blind people to know that the keyboard is pressed according to what they want.

Two-language voice calculator with Braille keypad based on ATMEGA128 microcontroller is a calculator that the keypad already uses braille code and is equipped with sound output. The disadvantage of this tool is that it is not equipped with Self-Correction, a feature that can correct answers automatically. Based on these shortcomings, it can be used as a reference in making tools which is equipped with a self-correction feature so that blind people can learn to count independently without relying on teaching staff.

A simple calculator system based on ATMega328 microcontroller with a 3x2 Braille keypad is suitable for use as a counting aid for the Blind. Simple calculator system based on the braille system as a numerical input by the user.

1.2 Research Gap

As discussed above, a lot of current research and systems have been presented but it is still not possible to present a complete system with all the proposed features. But this proposed system is more effective and comprehensive.

This table shows a summary comparison of the features of the proposed system and the existing system approach.

Research	Simple voice base calculator	Virtual tutor	Accessibility modein the UI design.	Mobile application
Research 1 [1]	\	✓	✓	✓
Research 2 [2]	✓	✓	\	✓
Research 3 [3]	✕	✕	✕	✕
New Research	✓	✓	✓	✓

Table 1

2. Research Problem

One of the requirements of inclusive growth is the progressive inclusion of disabled students, a marginalized group in society, at higher levels of education. It is a crucial step on the road to stability and full participation in life. In the age of technology enhanced learning, e-learning is essential for reducing the learning differences and challenges faced by students with disabilities. Recently, several initiatives have been put into place to improve accessibility for people with disabilities, including the installation of elevators, ramps, wheelchair access, and support personnel. Even though they haven't made the digital divide disappear. For some of these students, disabilities like dyslexia and dyscalculia make higher education challenging. Similarly, those who are blind or have low vision do not currently have access to technologies like screen readers, magnifiers, and so on to access e-content.

It appears that students with visual impairments face greater disadvantages than students with other disabilities, which disadvantages them. By lowering learning complexities, it promotes the use of assistive technologies to enhance learning. Although these assistive technologies make it possible for people with disabilities to access online learning materials, they are not universally effective. Additionally, they are not always cheap. As a result, some students with disabilities might not be able to afford these. Additionally, the design of the learning system does not account for the unique needs of students with disabilities. We cannot classify them as disability-aware e-learning systems as a result. Because people with disabilities will need personalized information in particular formats, assistive and adaptive technology must be created to offer universal access to knowledge.

3. Research Objectives

3.1 Main Objective

The primary aim of a voice and touch-based calculator designed for visually impaired students is to provide them with a seamless and efficient means of conducting mathematical computations without relying on visual cues. This specialized calculator is tailored to cater to the needs of visually impaired individuals, ensuring that they can effortlessly input data and receive results using voice commands. Visual feedback is unnecessary for these students, as they can interact with the calculator through tactile inputs and audio feedback.

By offering voice-based input options, the calculator allows visually impaired students to speak their mathematical equations and operations, providing a natural and intuitive way to perform calculations. The calculator's voice recognition technology accurately interprets their spoken commands, making complex mathematical tasks more accessible. In addition, the calculator's audio feedback feature ensures that students receive immediate auditory responses, allowing them to comprehend and verify their calculations efficiently.

Furthermore, a virtual tutor designed with visually impaired students in mind serves as a valuable tool for managing academic schedules and meetings. This virtual tutor acts as a personalized assistant, aiding visually impaired students in maintaining an organized and productive academic life. Its primary objective is to enhance accessibility and convenience for these students in various aspects of time management and scheduling.

The virtual tutor assists visually impaired students by offering timely reminders and notifications about upcoming events, classes, assignments, and appointments. Through auditory alerts and voice-based interactions, it ensures that students are well-prepared and punctual. This feature is crucial for individuals with visual impairments, as it helps them stay on top of their academic responsibilities and commitments.

Additionally, the virtual tutor can facilitate the scheduling of meetings and appointments, making the process straightforward and accessible. It allows students to create, modify, and manage their schedules efficiently, ensuring that they can balance their academic and personal obligations effectively.

In summary, both the voice and touch-based calculator and the virtual tutor for meeting scheduling are essential tools tailored to the specific needs of visually impaired students. These technologies prioritize accessibility and convenience, enabling visually impaired students to excel academically and manage their time efficiently, ultimately fostering independence and empowerment within the educational context.

3.2 Sub Objective

Virtual tutor and Voice calculator.

- Scheduling Meetings with Virtual Tutor: The system will have the functionality to schedule appointments or meetings with a virtual tutor, allowing users to set up specific times for interaction or learning sessions.
- Voice-Based and Touch-Based Calculator Options: The calculator within the system will offer two modes of operation: voice-based input and traditional touch-based input. Users can either speak their calculations or manually input them by touch.
- Touch Interface for Input: The system will include a touch-based interface to recognize numbers and input calculations. This interface could involve buttons or a touchpad to input digits and mathematical operations.
- Divided Screen Layout: The screen layout of the calculator will be divided into two primary sections: one for number input and the other for displaying or showing the ongoing calculations.
- Basic Operations for Voice-Based Calculator: The voice-based calculator will perform fundamental arithmetic operations such as addition, subtraction, multiplication, and division based on voice commands.
- Review & Rating Module: A module will be developed to allow users to leave reviews and ratings, providing feedback on their experiences with the virtual tutor or the calculator, thus enabling improvement based on user feedback.
- Accessibility Mode in UI Design: An accessibility mode will be integrated into the user interface (UI) design, ensuring the system is usable by individuals with diverse needs, including those with disabilities. This mode might involve features like screen reader compatibility, larger text options, or voice commands to enhance usability for all users.

These sub-objectives outline key functionalities and design elements for the development of the virtual tutor system and the voice-based calculator, emphasizing usability, interaction options, and the user experience.

4. Methodology

4.1 Methodology including the system diagram.

React Native Calculator Application

This part of the code is a React Native application for a calculator. It allows users to perform basic arithmetic operations both by pressing buttons on the screen and by using voice commands for input. Here's a breakdown of the key components and functionality:

React Native Calculator Application:

1. **State Variables:**

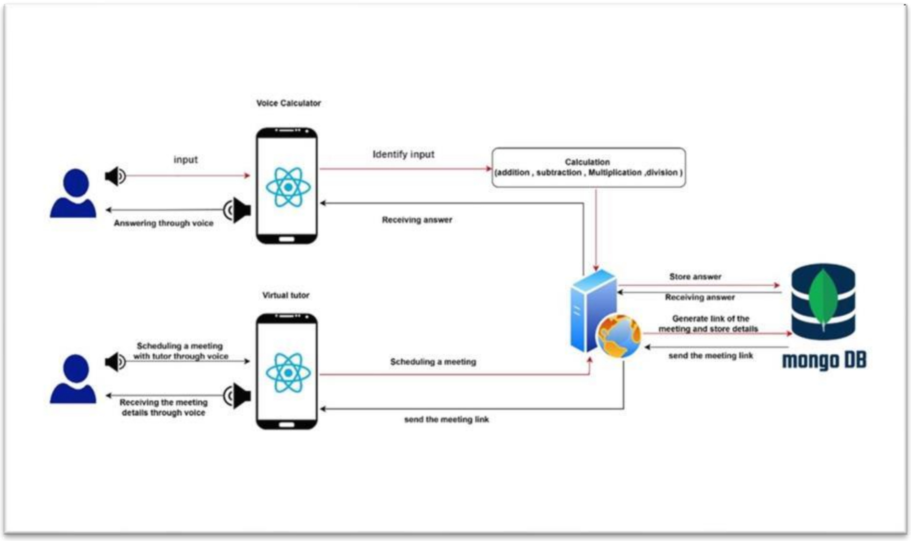
- result: Stores the current calculation result as a string.
- voiceInput: Holds the voice input provided by the user.
- isListening: Tracks whether the voice recognition system is actively listening**.**

2. **useEffect Hook:**
    - Sets up event listeners for voice recognition using the Voice library.
    - Cleans up these event listeners when the component unmounts.
3. **Button Handlers:**
    - handleButtonPress(value): Appends the pressed button's value to the result state.
    - handleCalculate(): Evaluates the mathematical expression stored in result using JavaScript's eval function.
    - handleClear(): Clears the result state.
4. **Voice Recognition:**
    - handleSpeechResults(event): Handles recognized speech results.
    - interpretVoiceInput(transcript): Converts recognized voice commands into calculator input (e.g., "add" becomes "+").
5. **Voice Recognition Control:**
    - startVoiceRecognition(): Activates voice recognition by setting isListening to true and calling the Voice.start function.
    - stopVoiceRecognition(): Stops voice recognition by setting isListening to false and calling the Voice.stop function.
6. **Render Method:**
    - Renders a user interface with buttons for digits, arithmetic operators, equals, and a voice recognition control button.
    - Displays the current calculation result and recognized voice input.

Python Speech Recognition and Calculation:

1. **Speech Recognition:**
    - Utilizes the speech_recognition library to capture audio input from the user's microphone.
    - get_voice_input() function captures and transcribes the user's spoken input using Google's speech recognition service.
2. **Mathematical Calculation:**
    - evaluate_math_expression(expression): Takes a mathematical expression as input, processes it, and calculates the result using the eval function.
    - Handles potential errors during the calculation process.
3. **Text-to-Speech Conversion:**
    - set_female_voice(): Initializes a text-to-speech engine with a female voice.
    - speak_text(text): Converts text into speech and plays it using the text-to-speech engine.
4. **Main Loop:**
    - Enters a loop that continuously captures voice input, processes it, and provides a response.
    - Checks for a "stop" command to exit the loop and end the script.
    - Detects the language based on the input text and evaluates it as a mathematical expression.
5. **gTTS (Google Text-to-Speech):**
    - **The gTTS library is likely used to convert text responses into speech.**

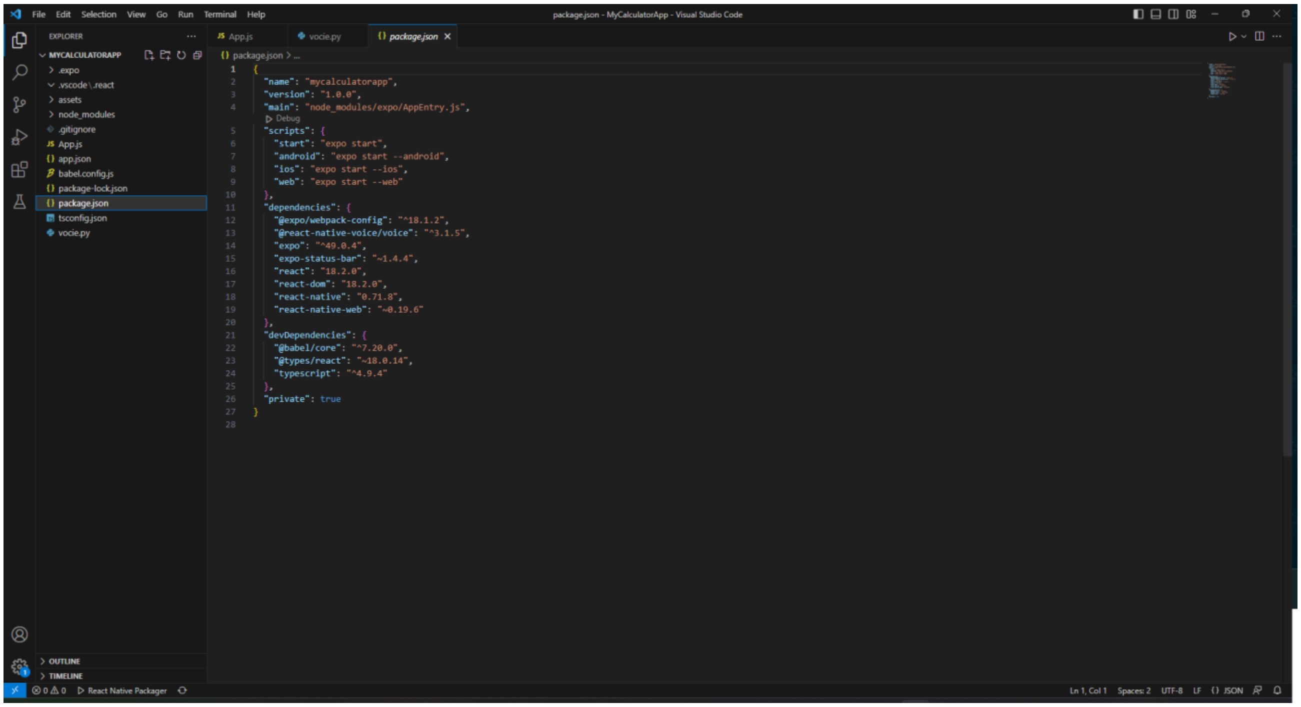
This combined system allows users to perform calculations using voice input through the React Native app, with the Python script managing the speech recognition, mathematical calculations, and providing spoken responses to the user. The React Native app captures user input sends it to the Python script for processing, and then presents the result back to the user via the app interface.


Voice Calculator
input
Answering through voice
Identify input
Calculation
(addition , subtraction , Multiplication ,division)
Receiving answer
Store answer
Receiving answer
Generate link of the
meeting and store details
send the meeting link
mongo DB
Virtual tutor
Scheduling a meeting
with tutor through voice
Receiving the meeting
details through voice
Scheduling a meeting
send the meeting link

Develop Software

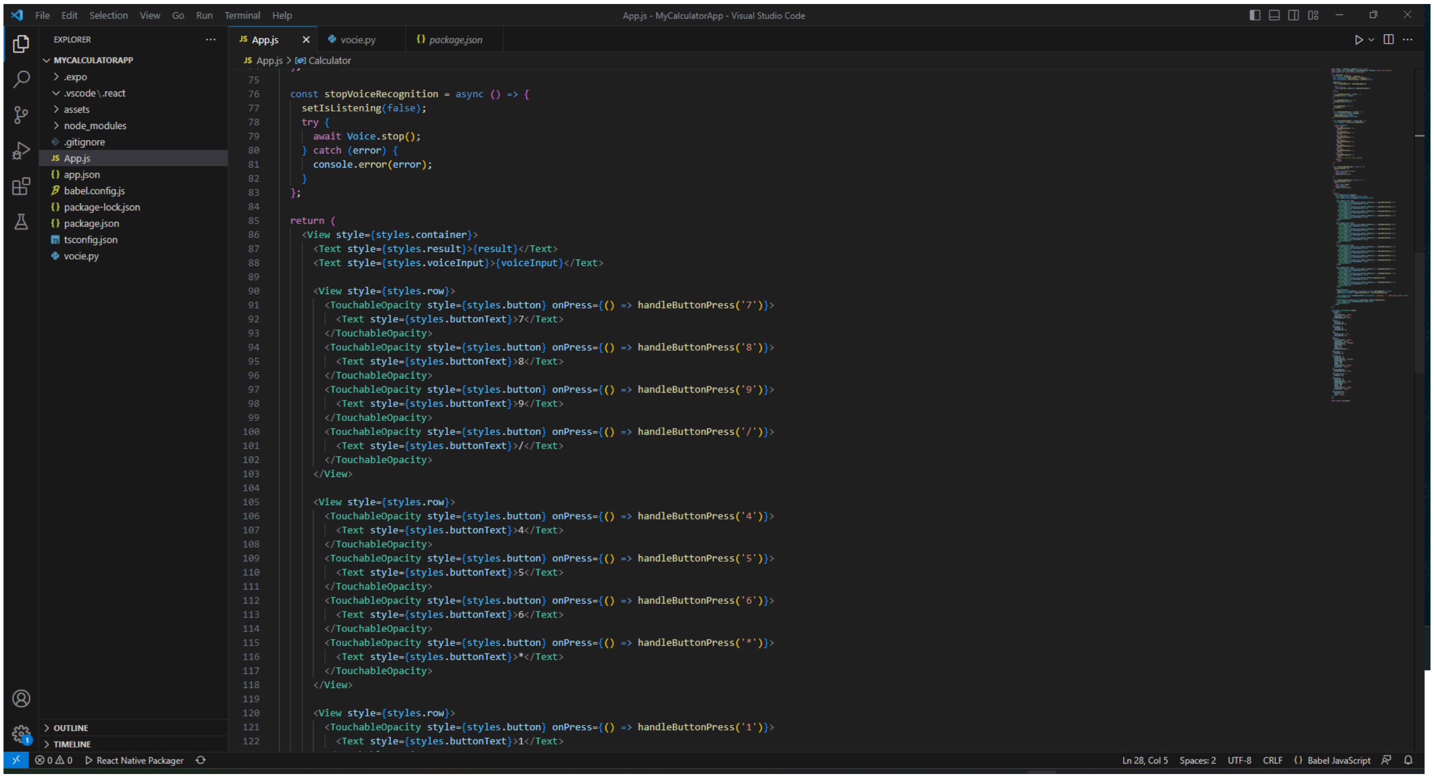
React Native Calculator App:

1. **Set Up the Development Environment:**
   - Install Node.js, npm, and the React Native CLI.
   - Create a new React Native project using the CLI.



```json
{
  "name": "mycalculatorapp",
  "version": "1.0.0",
  "main": "node_modules/expo/AppEntry.js",
  "scripts": {
    "start": "expo start",
    "android": "expo start --android",
    "ios": "expo start --ios",
    "web": "expo start --web"
  },
  "dependencies": {
    "@expo/webpack-config": "^18.1.2",
    "@react-native-voice/voice": "^3.1.5",
    "expo": "^49.0.4",
    "expo-status-bar": "~1.4.4",
    "react": "18.2.0",
    "react-dom": "18.2.0",
    "react-native": "0.71.8",
    "react-native-web": "~0.19.6"
  },
  "devDependencies": {
    "@babel/core": "^7.20.0",
    "@types/react": "~18.0.14",
    "typescript": "^4.9.4"
  },
  "private": true
}
```

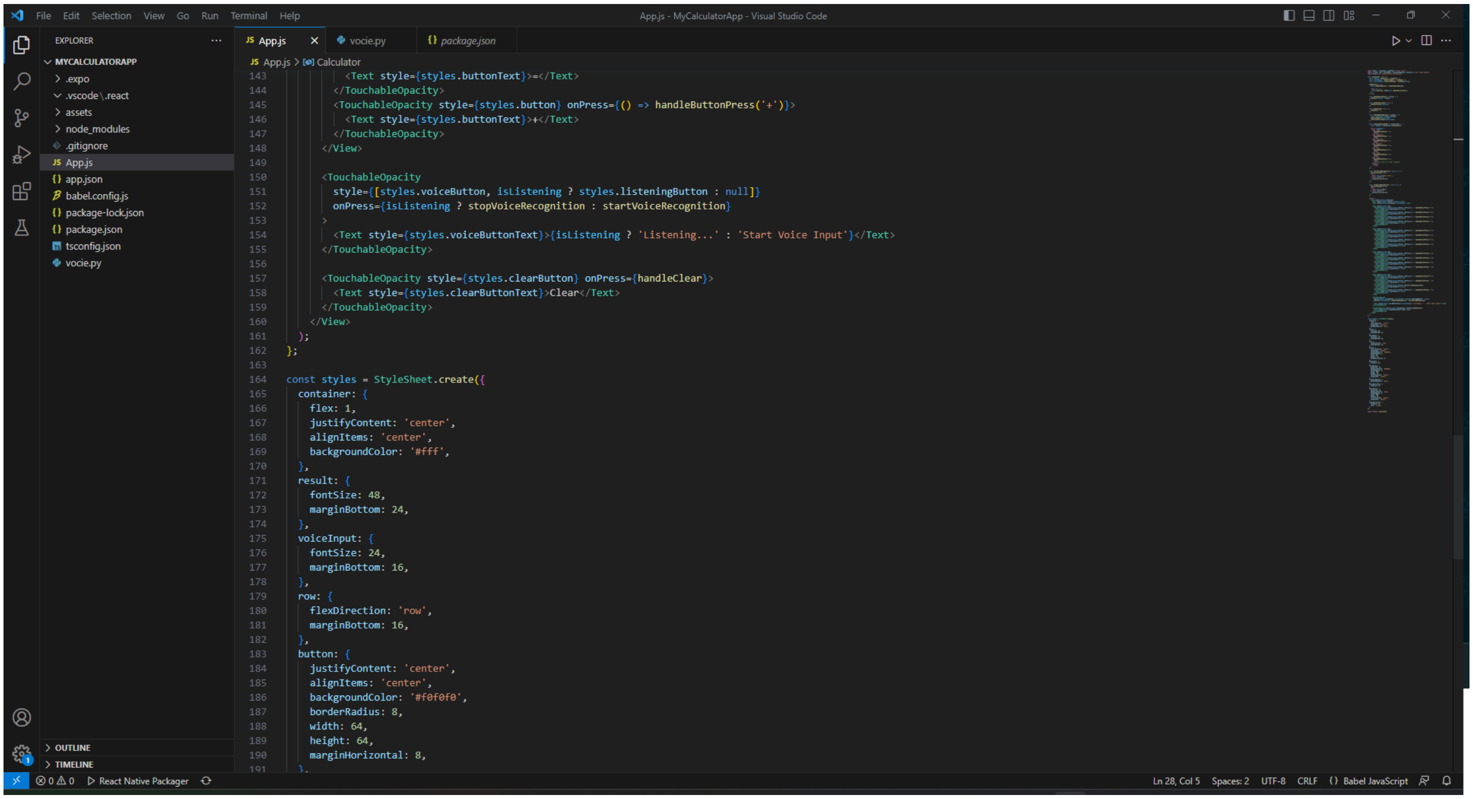
2. **User Interface (UI) Development:**
   - Design and implement the calculator's user interface using React Native components such as View, Text, and TouchableOpacity.
   - Create buttons for digits (0-9), arithmetic operators (+, -, *, /), equals (=), and voice recognition control.



```
  const stopVoiceRecognition = async () => {
    setIsListening(false);
    try {
      await Voice.stop();
    } catch (error) {
      console.error(error);
    }
  };

  return (
    <View style={styles.container}>
      <Text style={styles.result}>{result}</Text>
      <Text style={styles.voiceInput}>{voiceInput}</Text>

      <View style={styles.row}>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('7')}>
          <Text style={styles.buttonText}>7</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('8')}>
          <Text style={styles.buttonText}>8</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('9')}>
          <Text style={styles.buttonText}>9</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('/')}>
          <Text style={styles.buttonText}>/</Text>
        </TouchableOpacity>
      </View>

      <View style={styles.row}>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('4')}>
          <Text style={styles.buttonText}>4</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('5')}>
          <Text style={styles.buttonText}>5</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('6')}>
          <Text style={styles.buttonText}>6</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('*')}>
          <Text style={styles.buttonText}>*</Text>
        </TouchableOpacity>
      </View>

      <View style={styles.row}>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('1')}>
          <Text style={styles.buttonText}>1</Text>
```

3. State Management:

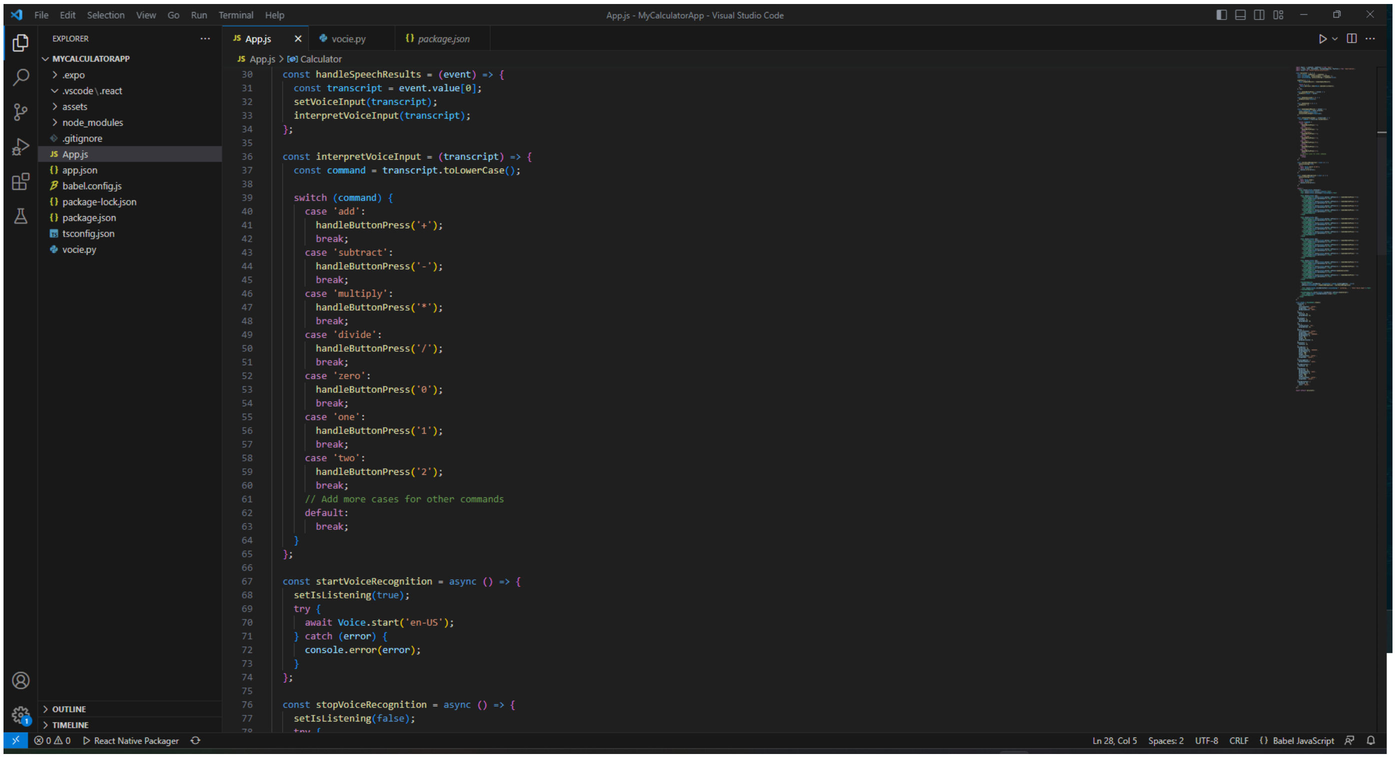
- Use React hooks like useState and useEffect to manage the app's state.
- Maintain state variables for the current calculation result, voice input, and whether voice recognition is active (isListening).



```
          <Text style={styles.buttonText}>=</Text>
        </TouchableOpacity>
        <TouchableOpacity style={styles.button} onPress={() => handleButtonPress('+')}>
          <Text style={styles.buttonText}>+</Text>
        </TouchableOpacity>
      </View>

      <TouchableOpacity
        style={[styles.voiceButton, isListening ? styles.listeningButton : null]}
        onPress={isListening ? stopVoiceRecognition : startVoiceRecognition}
      >
        <Text style={styles.voiceButtonText}>{isListening ? 'Listening...' : 'Start Voice Input'}</Text>
      </TouchableOpacity>

      <TouchableOpacity style={styles.clearButton} onPress={handleClear}>
        <Text style={styles.clearButtonText}>Clear</Text>
      </TouchableOpacity>
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#fff',
  },
  result: {
    fontSize: 48,
    marginBottom: 24,
  },
  voiceInput: {
    fontSize: 24,
    marginBottom: 16,
  },
  row: {
    flexDirection: 'row',
    marginBottom: 16,
  },
  button: {
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#f0f0f0',
    borderRadius: 8,
    width: 64,
    height: 64,
    marginHorizontal: 8,
  },
```

4. Button Handlers:

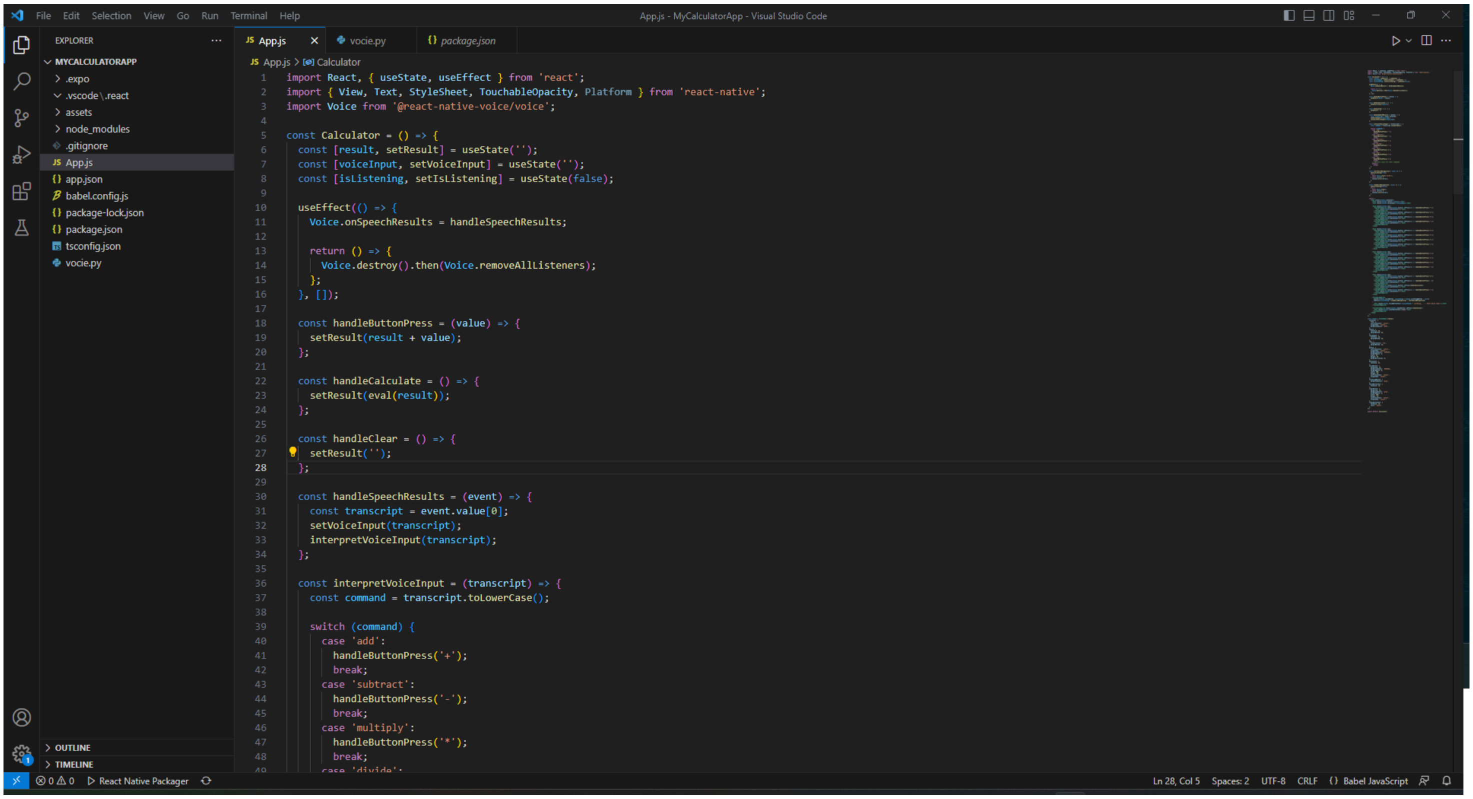
- Implement functions to handle button presses for digits, operators, and calculation.
- Update the result state accordingly.



```javascript
const handleSpeechResults = (event) => {
  const transcript = event.value[0];
  setVoiceInput(transcript);
  interpretVoiceInput(transcript);
};

const interpretVoiceInput = (transcript) => {
  const command = transcript.toLowerCase();

  switch (command) {
    case 'add':
      handleButtonPress('+');
      break;
    case 'subtract':
      handleButtonPress('-');
      break;
    case 'multiply':
      handleButtonPress('*');
      break;
    case 'divide':
      handleButtonPress('/');
      break;
    case 'zero':
      handleButtonPress('0');
      break;
    case 'one':
      handleButtonPress('1');
      break;
    case 'two':
      handleButtonPress('2');
      break;
    // Add more cases for other commands
    default:
      break;
  }
};

const startVoiceRecognition = async () => {
  setIsListening(true);
  try {
    await Voice.start('en-US');
  } catch (error) {
    console.error(error);
  }
};

const stopVoiceRecognition = async () => {
  setIsListening(false);
  try {
```

5. Voice Recognition Integration:

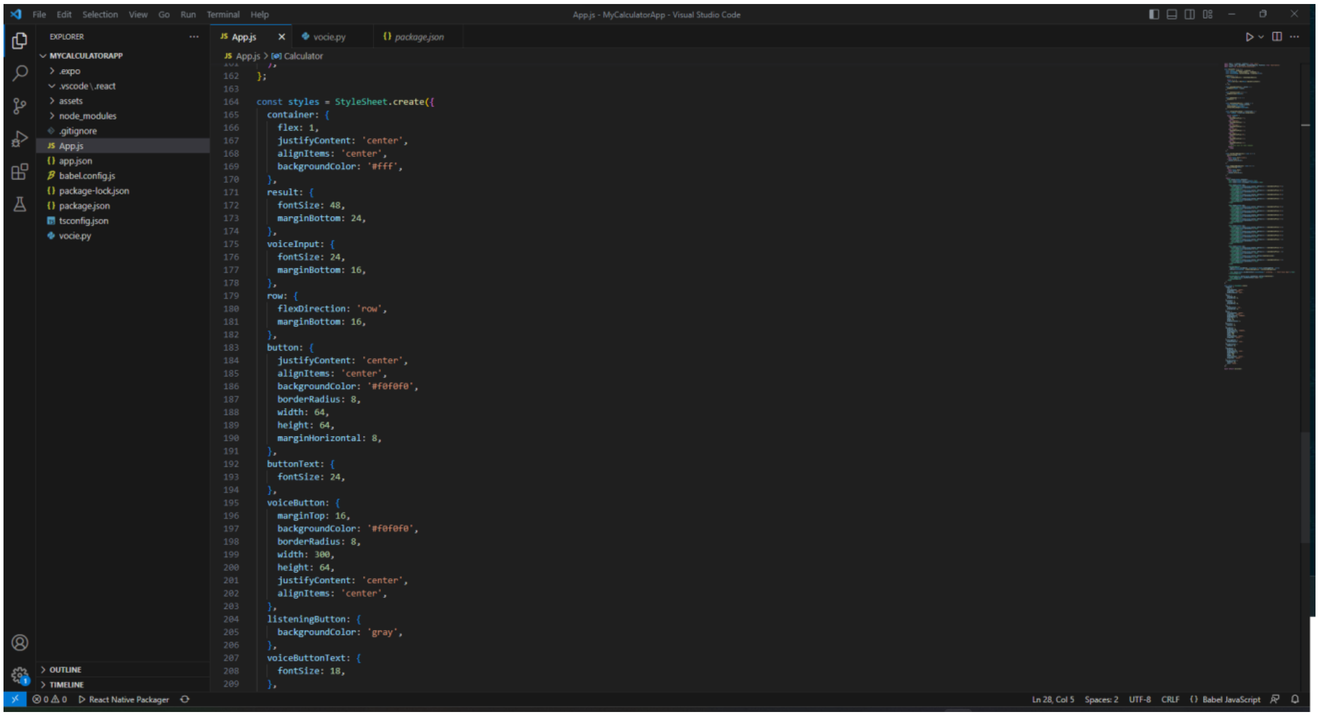
- Utilize the '@react-native-voice/voice' library for voice recognition.
- Implement functions to start and stop voice recognition.
- Handle recognized speech results and convert them into calculator input.



```
import React, { useState, useEffect } from 'react';
import { View, Text, StyleSheet, TouchableOpacity, Platform } from 'react-native';
import Voice from '@react-native-voice/voice';

const Calculator = () => {
  const [result, setResult] = useState('');
  const [voiceInput, setVoiceInput] = useState('');
  const [isListening, setIsListening] = useState(false);

  useEffect(() => {
    Voice.onSpeechResults = handleSpeechResults;

    return () => {
      Voice.destroy().then(Voice.removeAllListeners);
    };
  }, []);

  const handleButtonPress = (value) => {
    setResult(result + value);
  };

  const handleCalculate = () => {
    setResult(eval(result));
  };

  const handleClear = () => {
    setResult('');
  };

  const handleSpeechResults = (event) => {
    const transcript = event.value[0];
    setVoiceInput(transcript);
    interpretVoiceInput(transcript);
  };

  const interpretVoiceInput = (transcript) => {
    const command = transcript.toLowerCase();

    switch (command) {
      case 'add':
        handleButtonPress('+');
        break;
      case 'subtract':
        handleButtonPress('-');
        break;
      case 'multiply':
        handleButtonPress('*');
        break;
      case 'divide':
```

6. Styling and Layout:

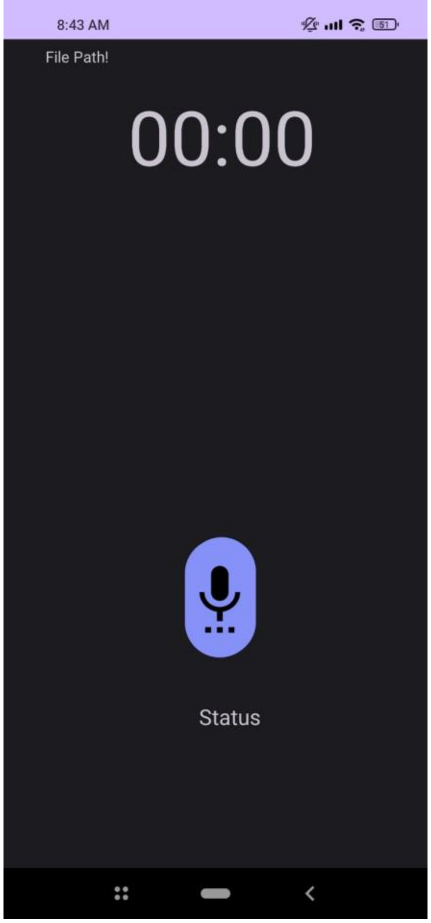
- Apply styles using StyleSheet.create to create a visually appealing and user-friendly UI.
- Organize buttons and elements in rows and columns.



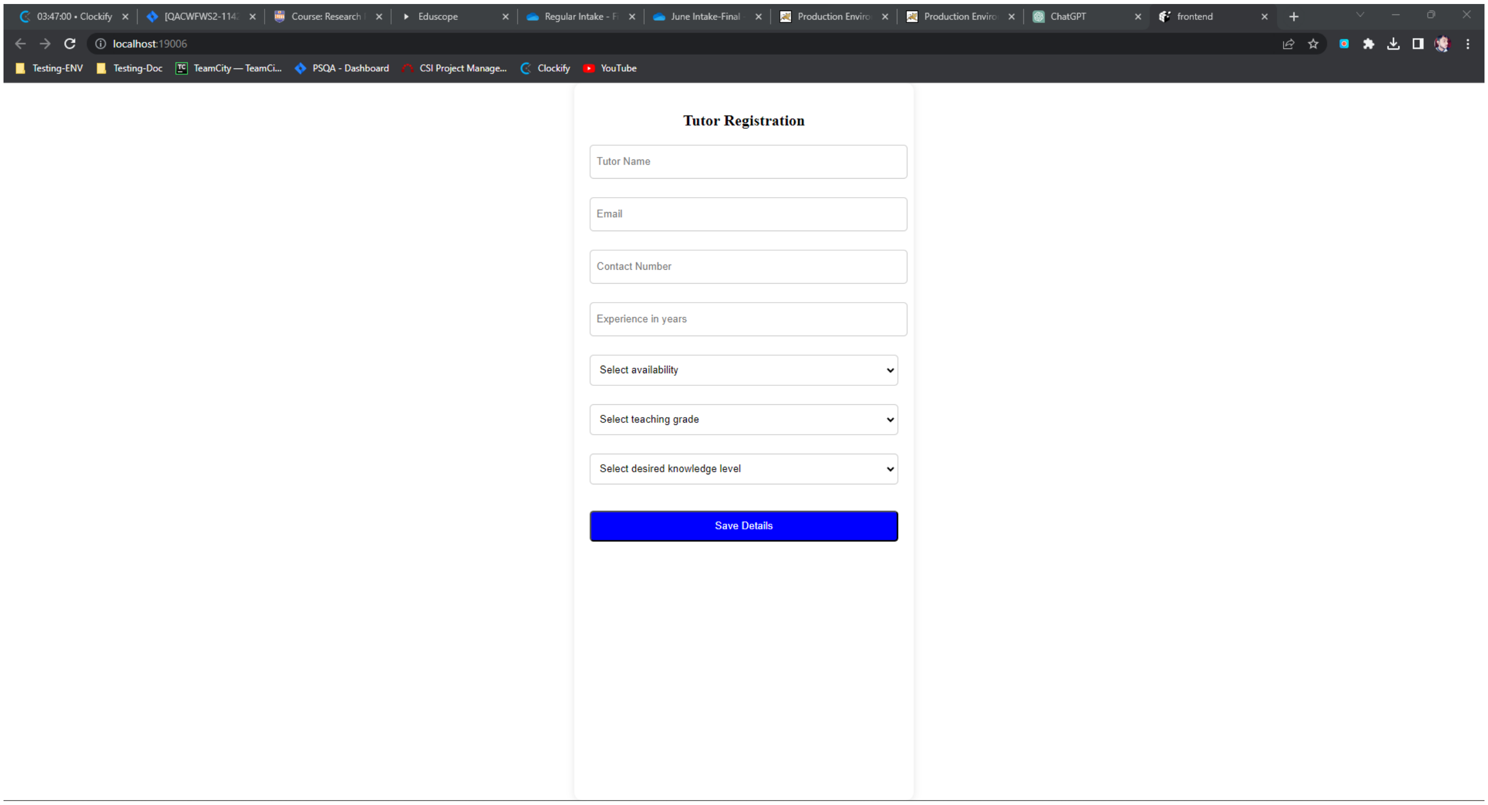
```
  };

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#fff',
  },
  result: {
    fontSize: 48,
    marginBottom: 24,
  },
  voiceInput: {
    fontSize: 24,
    marginBottom: 16,
  },
  row: {
    flexDirection: 'row',
    marginBottom: 16,
  },
  button: {
    justifyContent: 'center',
    alignItems: 'center',
    backgroundColor: '#f0f0f0',
    borderRadius: 8,
    width: 64,
    height: 64,
    marginHorizontal: 8,
  },
  buttonText: {
    fontSize: 24,
  },
  voiceButton: {
    marginTop: 16,
    backgroundColor: '#f0f0f0',
    borderRadius: 8,
    width: 300,
    height: 64,
    justifyContent: 'center',
    alignItems: 'center',
  },
  listeningButton: {
    backgroundColor: 'gray',
  },
  voiceButtonText: {
    fontSize: 18,
  },
```

7. **Testing and Debugging:**
    - Test the app on emulators or physical devices for both Android and iOS.
    - Debug and resolve any issues related to UI

Calculator Updated UI:



Tutor Profile registration:



Tutor Registration

Tutor Name

Email

Contact Number

Experience in years

Select availability

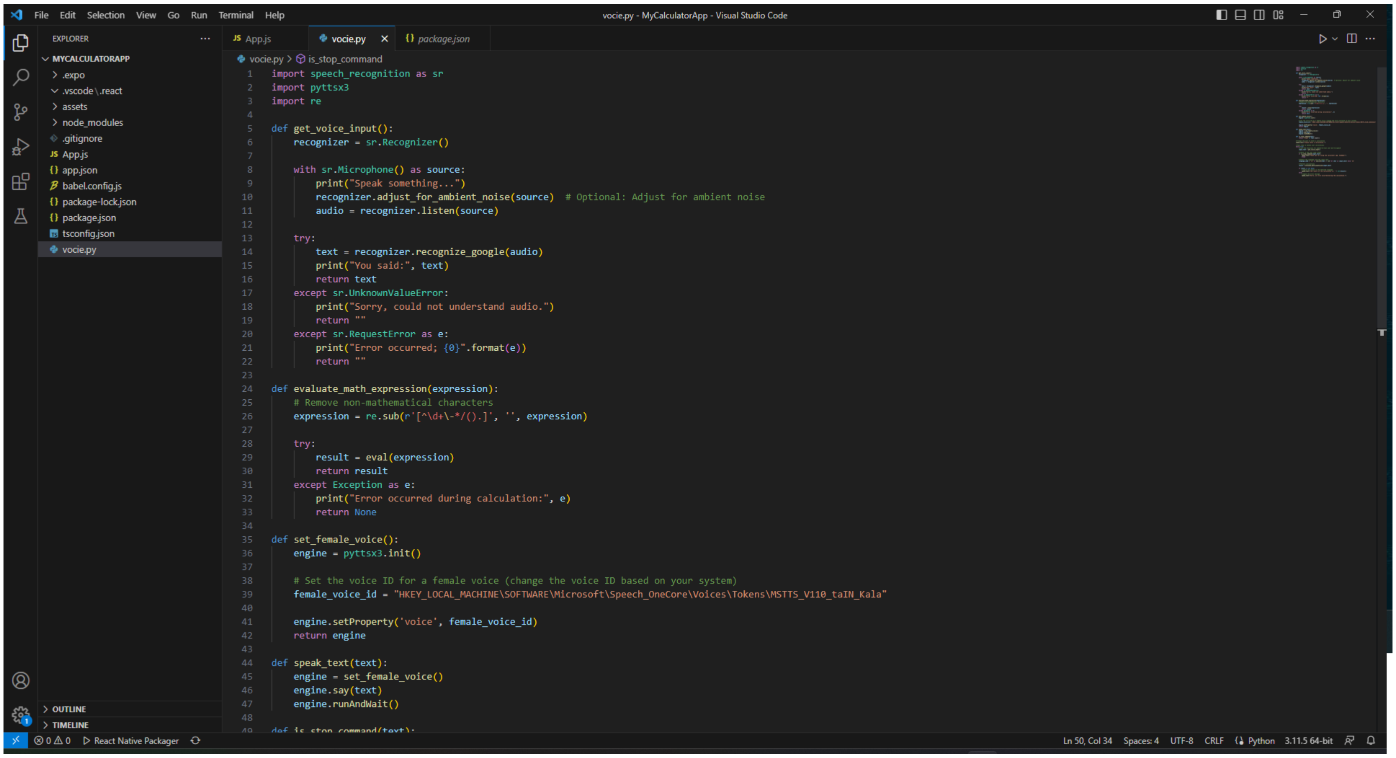
Select teaching grade

Select desired knowledge level

Python Script for Speech Recognition and Calculation:

1. Set Up the Python Environment:
   - Install Python if not already installed.
   - Use pip to install required Python libraries: speech_recognition, pyttsx3, and gTTS

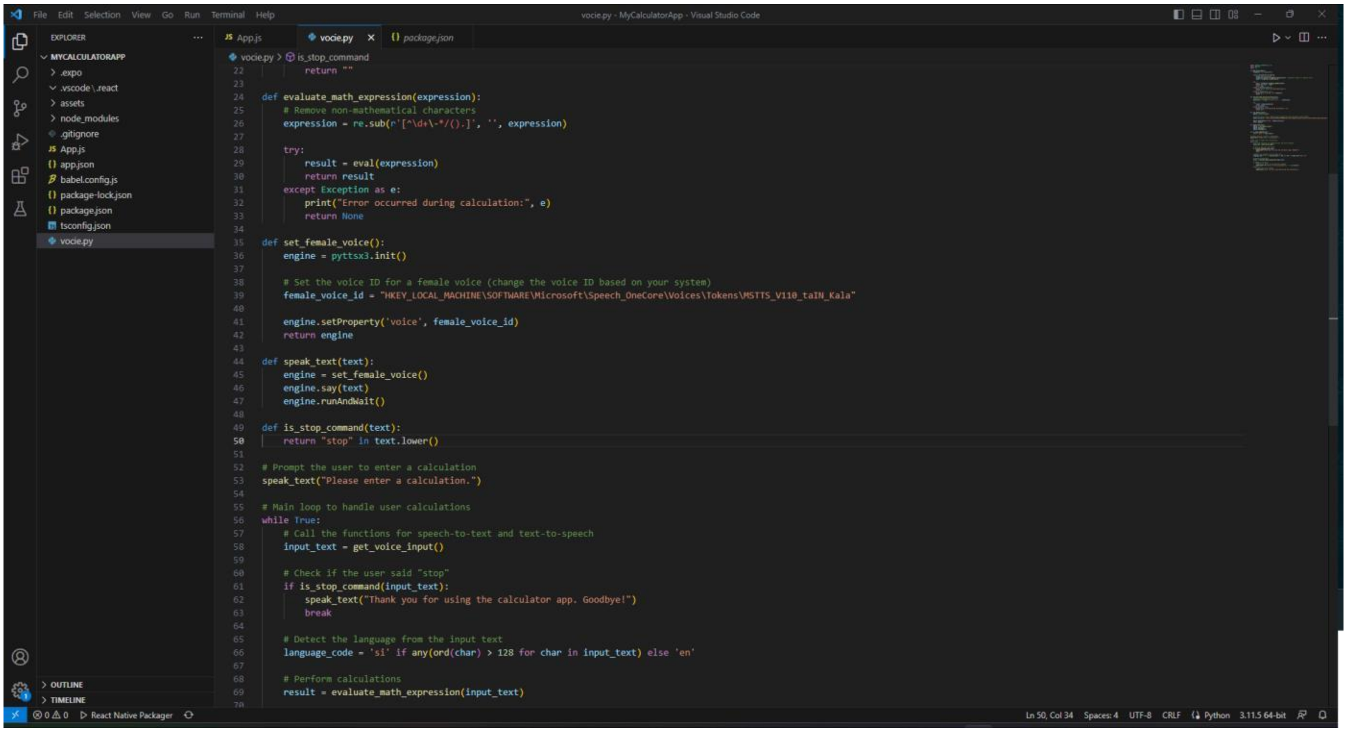
2. Speech Recognition:
   - Use the speech_recognition library to capture audio input from the microphone.
   - Create a function to transcribe the user's spoken input using Google's speech recognition service.



```python
import speech_recognition as sr
import pyttsx3
import re

def get_voice_input():
    recognizer = sr.Recognizer()

    with sr.Microphone() as source:
        print("Speak something...")
        recognizer.adjust_for_ambient_noise(source)  # Optional: Adjust for ambient noise
        audio = recognizer.listen(source)

    try:
        text = recognizer.recognize_google(audio)
        print("You said:", text)
        return text
    except sr.UnknownValueError:
        print("Sorry, could not understand audio.")
        return ""
    except sr.RequestError as e:
        print("Error occurred; {0}".format(e))
        return ""

def evaluate_math_expression(expression):
    # Remove non-mathematical characters
    expression = re.sub(r'[^\d+\-*/().]', '', expression)

    try:
        result = eval(expression)
        return result
    except Exception as e:
        print("Error occurred during calculation:", e)
        return None

def set_female_voice():
    engine = pyttsx3.init()

    # Set the voice ID for a female voice (change the voice ID based on your system)
    female_voice_id = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech_OneCore\Voices\Tokens\MSTTS_V110_taIN_Kala"

    engine.setProperty('voice', female_voice_id)
    return engine

def speak_text(text):
    engine = set_female_voice()
    engine.say(text)
    engine.runAndWait()

def is_stop_command(text):
```

3. Mathematical Calculation:

- Develop a function to evaluate mathematical expressions provided as text input.
- Handle potential errors during calculation and return the result or an error message.

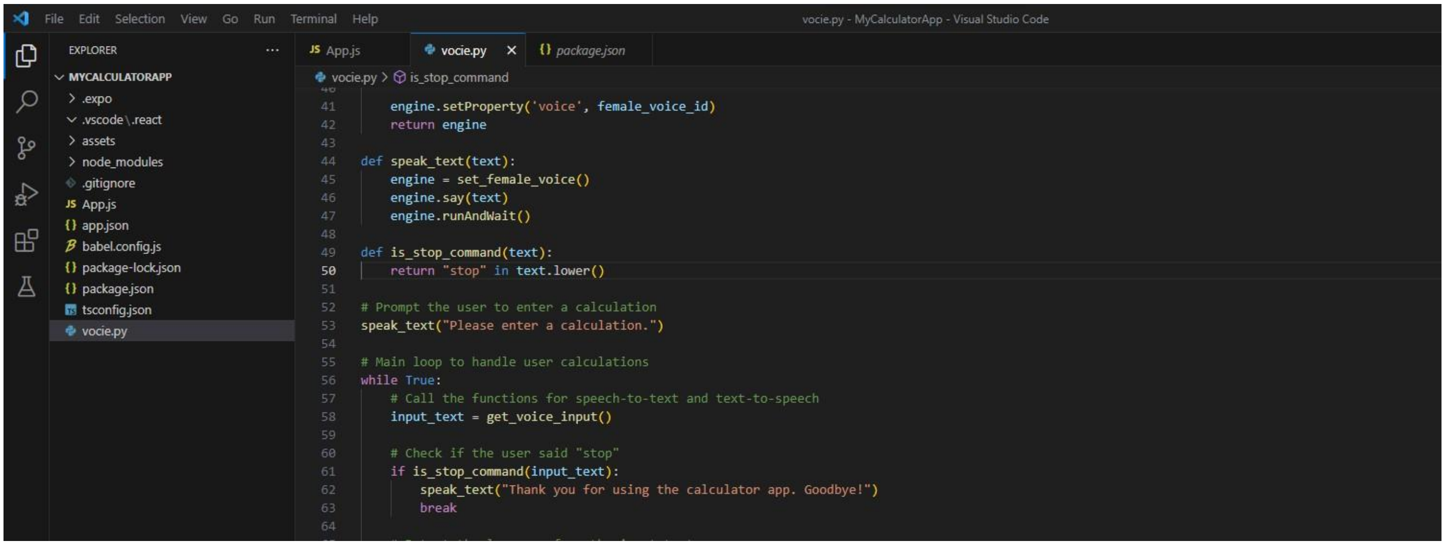


```python
        return ""

def evaluate_math_expression(expression):
    # Remove non-mathematical characters
    expression = re.sub(r'[^\d+\-*/().]', '', expression)

    try:
        result = eval(expression)
        return result
    except Exception as e:
        print("Error occurred during calculation:", e)
        return None

def set_female_voice():
    engine = pyttsx3.init()

    # Set the voice ID for a female voice (change the voice ID based on your system)
    female_voice_id = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech_OneCore\Voices\Tokens\MSTTS_V110_taIN_Kala"

    engine.setProperty('voice', female_voice_id)
    return engine

def speak_text(text):
    engine = set_female_voice()
    engine.say(text)
    engine.runAndWait()

```

```python
def is_stop_command(text):
    return "stop" in text.lower()

# Prompt the user to enter a calculation
speak_text("Please enter a calculation.")

# Main loop to handle user calculations
while True:
    # Call the functions for speech-to-text and text-to-speech
    input_text = get_voice_input()

    # Check if the user said "stop"
    if is_stop_command(input_text):
        speak_text("Thank you for using the calculator app. Goodbye!")
        break

    # Detect the language from the input text
    language_code = 'si' if any(ord(char) > 128 for char in input_text) else 'en'

    # Perform calculations
    result = evaluate_math_expression(input_text)
```

4. Text-to-Speech Conversion:
    - Implement a function to convert text responses into speech using the 'pyttsx3' library.

5. Main Loop:
    - Create a loop to continuously capture voice input from the user.
    - Check for a "stop" command to exit the loop and end the script.
    - Detect the language from the input text.



```python
    engine.setProperty('voice', female_voice_id)
    return engine

def speak_text(text):
    engine = set_female_voice()
    engine.say(text)
    engine.runAndWait()

def is_stop_command(text):
    return "stop" in text.lower()

# Prompt the user to enter a calculation
speak_text("Please enter a calculation.")

# Main loop to handle user calculations
while True:
    # Call the functions for speech-to-text and text-to-speech
    input_text = get_voice_input()

    # Check if the user said "stop"
    if is_stop_command(input_text):
        speak_text("Thank you for using the calculator app. Goodbye!")
        break

```

    - Perform calculations and provide spoken responses based on the input.

6. Testing and Debugging:
    - Test the Python script independently to ensure accurate speech recognition, calculations, and text-to-speech responses.
    - Debug and resolve any issues in the script**.**

Figure 1

Methodology of Calculator and Tutor profile

Voice and Touch-Based Calculator:

The voice and touch-based calculator is designed to cater to visually impaired students by offering a straightforward and accessible method for performing mathematical calculations. To use the calculator, the user can either speak or input numbers and mathematical operations via tactile touch inputs. The system employs advanced recognition technology to understand and interpret the user's input accurately. Once the input is recognized, the calculator proceeds to execute the mathematical operations as specified by the user, following standard mathematical rules (e.g., order of operations). Importantly, it stores the correct results in a database for future reference or verification. The calculator provides immediate feedback by vocalizing the calculated answer to the user, ensuring that visually impaired students can independently and confidently carry out mathematical tasks without the need for visual cues.

Virtual Tutor for Meeting Scheduling Methodology Explanation: The virtual tutor for meeting scheduling is a specialized tool designed to assist visually impaired students in managing their academic schedules and appointments with ease. The user interacts with the virtual tutor by verbally providing information about upcoming meetings, including details like the date and time. Using natural language processing and voice recognition capabilities, the system comprehends the user's input and assesses their scheduling needs. Once it understands the context, it generates the necessary meeting link for the specified event. Depending on the user's preferences and requirements, the virtual tutor may also include additional information, such as meeting agendas or participant lists. Crucially, the virtual tutor delivers this generated information back to the user through audio feedback, ensuring that visually impaired students have equal access to essential scheduling tools and can efficiently manage their academic commitments.

4.2 Commercialization of the Product

E-learning systems for elementary kids are in demand from educational institutions such primary schools, special education institutions, and online learning platforms that cater to students who are blind or visually impaired. The audience that is addressed should primarily consist of teachers, parents, and guardians of visually impaired pupils.

The e-learning platform should be designed with accessibility features, such as audio explanations, high Contrast Mode, text-to-Speech Mode, magnified Mode, and other elements that are particular to the needs of visually impaired students.

The primary kids should not only be the focus of the e-learning system, but also their instructors and parents/guardians, who play a significant role in fostering their academic growth.

The visually impaired society as a whole, including primary schools, visually impaired children, educators, parents, guardians, and organizations that serve the visually impaired society, would be the target market and audience for an e-learning system for visually impaired elementary students.

Commercialization also involves generating revenue from the mobile application. This could involve charging for the application itself, offering premium features for a fee, or monetizing through in-app advertising. Further, use social media to promote.

5. Software Specifications

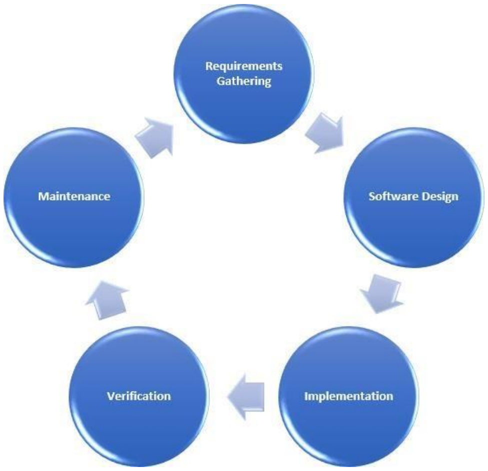


Figure 2

5.2 Tool and Technology

Tools

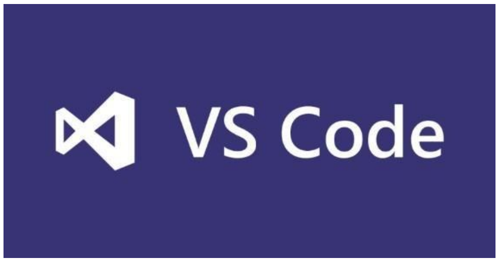
Mongo DB

Figure 3

MongoDB is an open-source, document-oriented database system. A NoSQL database program called MongoDB uses documents with schema that mimic JSON. For usage with MongoDB, developer MongoDB

Inc. provides the Server-Side Public License.

VS Code



Figure

Microsoft created Visual Studio Code, popularly known as VS Code, a source-code editor for Windows, Linux, and macOS that makes use of the Electron Framework. Among the features are debugging assistance, syntax highlighting, intelligent code completion, snippets, code refactoring, and integrated Git.

Technology

React Native

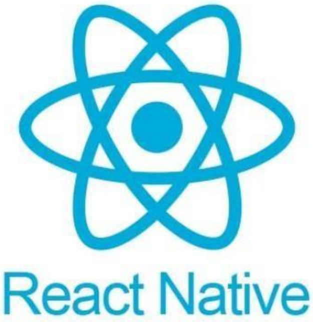


Figure 5

Facebook developed the well-known open source React Native framework for building mobile applications. It enables developers to use a single codebase to create cross-platform mobile applications for iOS, Android, and the web. The ReactJS library, which is used to create web apps, is the foundation around which React Native is constructed.

Python

5.3 Functional Requirements

1. **Accurate Arithmetic Calculations:** The calculator should accurately perform basic arithmetic operations, ensuring precise results for addition, subtraction, multiplication, and division.
2. **Accuracy in Touch System:** In line with the touch system, the calculator should provide precise and correct results when users input calculations via touch, ensuring reliability in touch-based interactions.
3. **Clear Audible Output:** The calculator should audibly and clearly voice the numbers entered and the results of calculations. This feature is particularly crucial for visually impaired users who rely on auditory feedback.
4. **Braille Symbols and Sounds:** To accommodate users who are visually impaired, the calculator interface should be designed with sections featuring Braille symbols and associated sounds to ensure accessibility.
5. **User-Friendly Navigation and Controls:** The calculator should have intuitive and easy-to-use navigation elements, such as buttons or gestures, to facilitate efficient and accurate calculations, catering to users with varying needs and abilities.
6. **Support for Multiple Input Methods:** The calculator should accommodate diverse input methods, including voice input, keypad input, or a combination of both, providing flexibility for users with different preferences or accessibility needs.
7. **Virtual Tutor Scheduling in Mobile App:** The mobile application should offer a feature enabling students to check the availability of virtual tutors and conveniently schedule appointments at their preferred time and date.
8. **Personalization of Virtual Meetings:** The system should allow students to tailor virtual meetings to their specific needs and preferred learning style, ensuring a more effective and personalized learning experience.
9. **Voice-Based Commands for Visually Impaired Users:** To assist visually impaired students, the application should support voice-based commands and prompts that aid in navigating the scheduling process for virtual meetings.
10. **Feedback Mechanism:** The system should provide a mechanism for students to offer feedback and evaluation on the virtual tutor's performance, meeting quality, and the effectiveness of teaching methods. This feedback loop enables continuous improvement in the tutoring experience.

5.4 Non -Functional Requirement

1.1.1. Usability

- The application should be able to achieve user's required goals efficiently and effectively.
- The application should have a user-friendly interface that is easy to navigate and understand.

1.1.2. Reliability

- The application output accuracy should be error free, and the code should be bug free.so the translation output could be reliable.

1.1.3. Performance

- The application should be able to respond to user requests quickly and efficiently, without lag or delays.

1.1.4. Security and privacy

- The administrative application should be protected from unauthorized access.
- The database should be protected from attacks and unauthorized access.
- The interface should be protected from attacks.

- All passwords should be stored as a secure hash of the administrator password.

1.1.5. Scalability

- The application should be able to handle a large number of users and requests without crashing or experiencing downtime.

1.1.6. Maintainability

- The application should be designed in a way that makes it easy to maintain, update, and fix bugs over time.

1.1.7. Availability

- The system should be available to users 24/7, with minimal downtime for maintenance or other issues.

1.1.8. Portability

- The application should be able to run on multiple platforms and devices, without requiring major modifications or changes.

1. Conclusions

The suggested e-learning system for visually impaired primary students presents a progressive solution addressing the educational challenges faced by these students. Comprising four fundamental components tailored to meet the unique needs of visually impaired pupils, the system utilizes React Native as its technology and MongoDB as the database.

The primary objective involves the development of a virtual tutor and a straightforward voice-based calculator. Through these functions, students can engage in personalized learning experiences with a virtual instructor at their own pace and convenience. The voice-based calculator serves as a valuable tool enabling these students to perform basic arithmetic operations effortlessly.

The overall proposed system has a multifaceted impact. It not only enhances the acquisition of knowledge and academic skills but also significantly eases the daily study life of visually impaired primary students. By facilitating personalized learning and providing tools for independent calculation, it contributes to social equality by enabling these students to access education in a manner that aligns with their needs.

Ultimately, this comprehensive system stands to revolutionize the educational experience for visually impaired primary students, fostering independence, enhancing their learning journey, and contributing to their overall academic and social well-being.

References

[1] Tasnim Ahmed , Implementation of Bangla Speech Recognition in Voice Input Speech Output (VISO) Calculator, Bangladesh, 2018.

[2] A. Sudaryanto, Calculator for Blind with Self Correction Feature, 2018.

[3] G. H. B. A. D. Silva, An investigation of visually impaired learners, Dalugama,srilanka: International Conference on Research and Academic Community Services (ICRACOS), 2021.

6. APPENDICES

6.1 Work Breakdown Structure

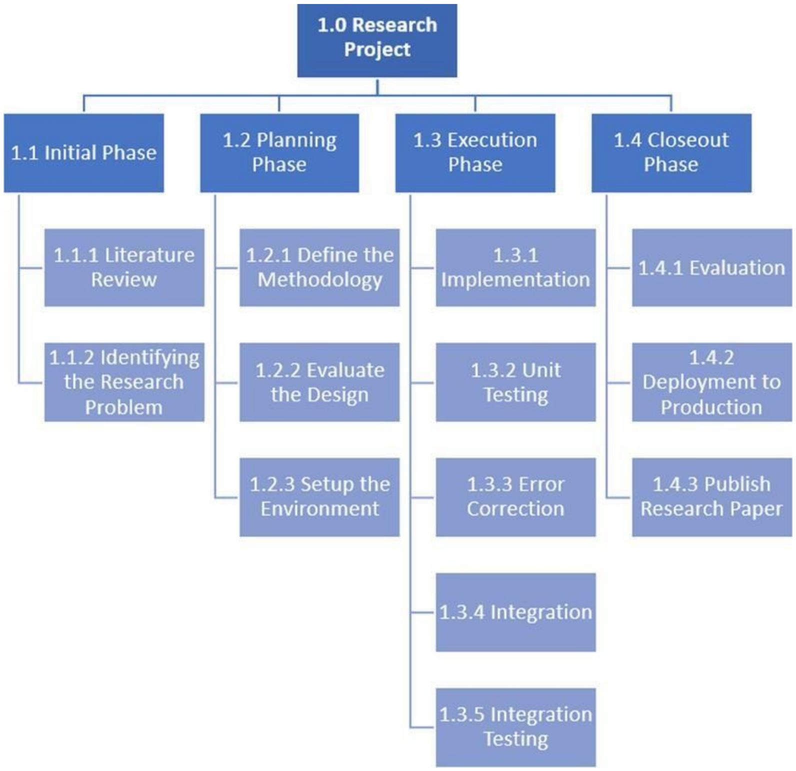


Figure 6

6.2 Gantt Chart

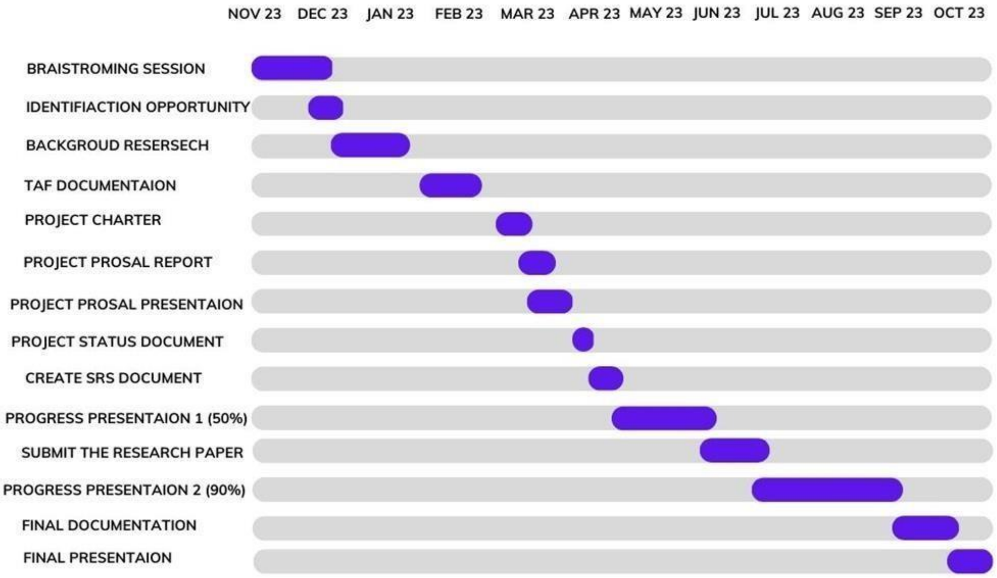


Figure 7

6.3 System Overview

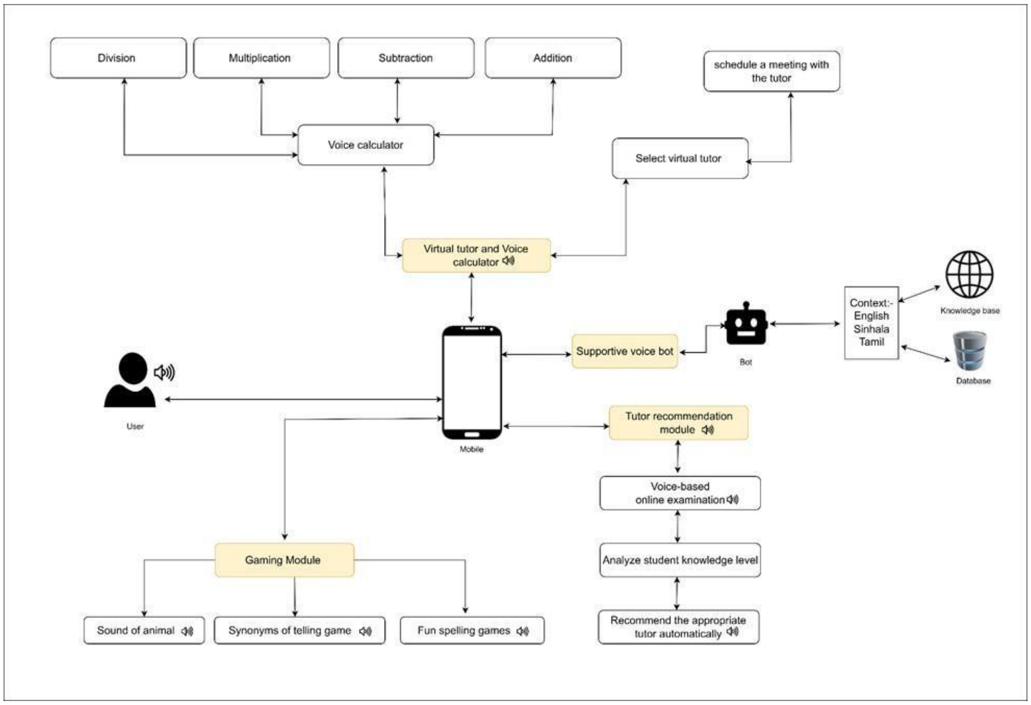

Division
Multiplication
Subtraction
Addition
schedule a meeting with the tutor
Voice calculator
Select virtual tutor
Virtual tutor and Voice calculator
Context:- English Sinhala Tamil
Knowledge base
Database
Bot
Supportive voice bot
User
Mobile
Tutor recommendation module
Voice-based online examination
Analyze student knowledge level
Recommend the appropriate tutor automatically
Gaming Module
Sound of animal
Synonyms of telling game
Fun spelling games

Figure 8

6.4 Blind school images

We visited a Ratmalana blind school to speak with teachers and students about their experiences, problems, and expectations for mobile applications, among other things.

Figure 9

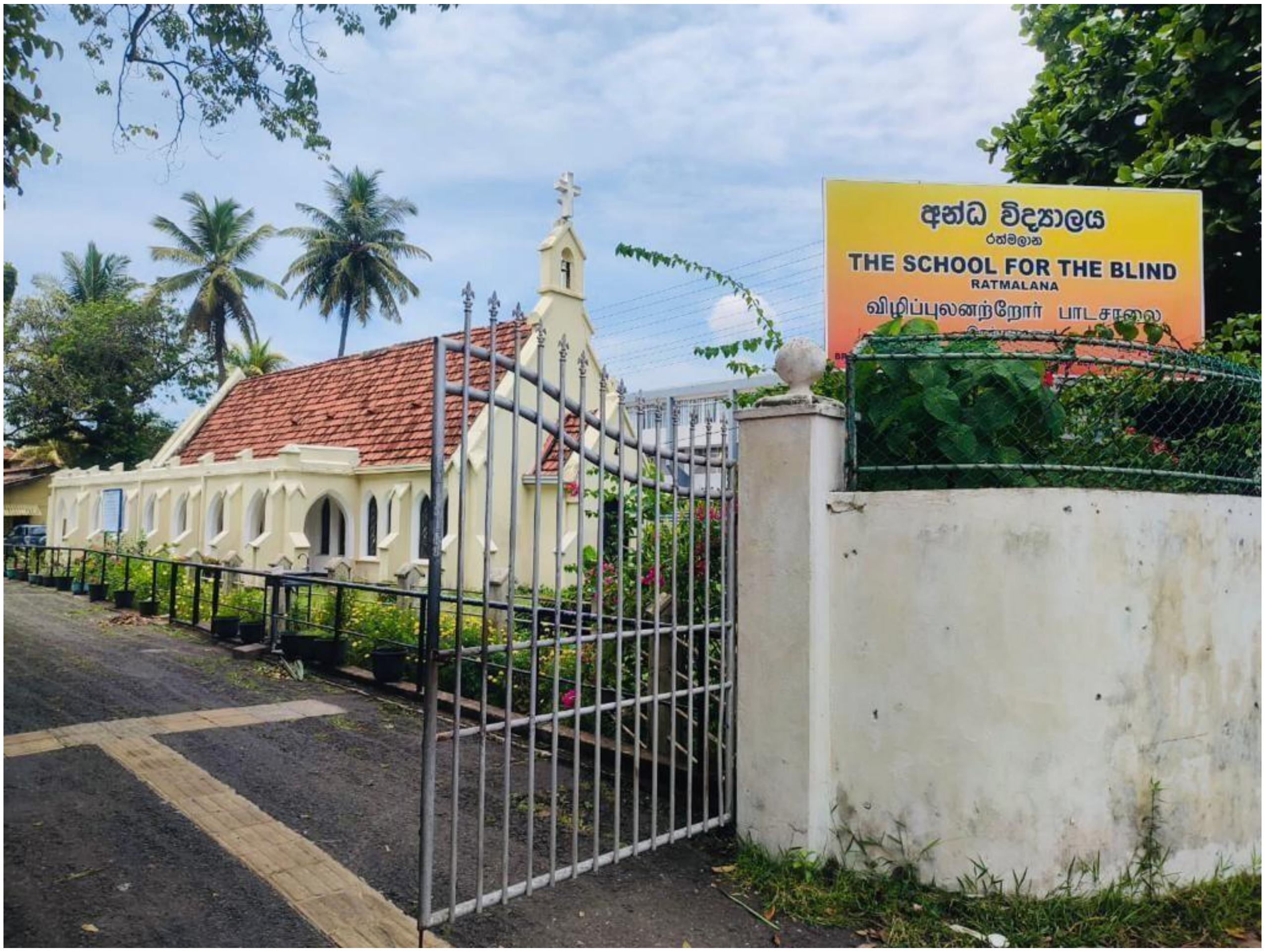

අන්ධ විද්‍යාලය
රත්මලාන
THE SCHOOL FOR THE BLIND
RATMALANA
விழிப்புலனற்றோர் பாடசாலை

Figure 10