MINI PROJECT – TEXT TO SPEECH CONVERSION

AIM:

To implement text-to-speech (TTS) conversion in an Android application, using java in Android studio Text To Speech API.

STEPS:

Step 1: Create a New Project

Step 2: Working with activity_main.xml file

Go to the app -> res -> layout -> activity_main.xml section and set the layout for the app. In this file add an Edit Text to input the text from the user, a Button, so whenever the user clicks on the Button then it's converted to speech and a Text View to display.

Step 3: Working with MainActivity.java file

Go to the app -> java -> com.example.GFG (Package Name) -> MainActivity.java section. Now join the Button and Edit text to Java code and comments are added inside code to understand the code easily.

PROBLEM STATEMENT:

Overview:

You are tasked with developing an Android application that converts text input into speech output. The application should utilize the Text-to-Speech (TTS) functionality available in Android devices to provide a seamless and intuitive user experience.

Features to Implement:

Text Input: The application should allow users to input text through a user interface component, such as an EditText view.

Text-to-Speech Conversion: Upon user input, the application should convert the entered text into speech using the Text-to-Speech engine available on the device.

Playback Controls: Users should have basic playback controls, such as play, pause, stop, and adjust volume, to control the speech output.

Language Support: The application should support multiple languages for text-to-speech conversion. Users should be able to select the desired language from a list of available options.

Pitch and Speed Adjustment: Provide options for users to adjust the pitch and speed of the speech output according to their preferences.

Error Handling: Implement robust error handling to manage scenarios such as initialization failures, unsupported languages, or errors during speech playback.

Accessibility: Ensure that the application adheres to accessibility standards, making it usable by individuals with disabilities, including support for screen readers and alternative input methods.

Additional Requirements:

User Interface Design: Design a user-friendly interface that facilitates easy text input and provides intuitive playback controls.

Performance Optimization: Optimize the application's performance to ensure smooth text-to-speech conversion and playback, even on low-end devices.

Testing and Debugging: Thoroughly test the application on various Android devices and screen sizes to identify and resolve any issues. Implement logging and debugging tools to facilitate troubleshooting during development.

Documentation: Provide clear documentation, including user guides and code comments, to assist users and developers in understanding the application's functionality and implementation details.

IMPLEMENTATION:

MAIN XML CODE:

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

        xmlns:android="http://schemas.android.com/apk/res/android"

        xmlns:app="http://schemas.android.com/apk/res-auto"

        xmlns:tools="http://schemas.android.com/tools"

        android:layout_width="match_parent"

        android:layout_height="match_parent"

        android:orientation="vertical"

        android:layout_margin="30dp"

        tools:context=".MainActivity">
```

```
<!--To add text in the app-->
<EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/Text"
        android:layout_marginBottom="20dp"
        android:hint="Enter Any Sentence"
        android:gravity="center"
        android:textSize="16dp"/>

<!--when you press this button it will
        convert text into speech-->
<Button
        android:layout_width="wrap_content"
        android:id="@+id/btnText"
        android:layout_height="wrap_content"
        android:text="Click Here"
        android:layout_gravity="center"/>

<!--To display the name of GeeksForGeeks -->
<TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="70dp"
        android:gravity="center_horizontal"
        android:text="GEEKSFORGEEKS"
        android:textColor="@android:color/holo_green_dark"
        android:textSize="36sp" />
```

```
</LinearLayout>
```

JAVA CODE:

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.speech.tts.TextToSpeech;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import java.util.Locale;

public class MainActivity extends AppCompatActivity {

    EditText Text;
    Button btnText;
    TextToSpeech textToSpeech;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Text = findViewById(R.id.Text);
        btnText = findViewById(R.id.btnText);

        // create an object textToSpeech and adding features into it
        textToSpeech = new TextToSpeech(getApplicationContext(), new
TextToSpeech.OnInitListener() {
            @Override
            public void onInit(int i) {
```

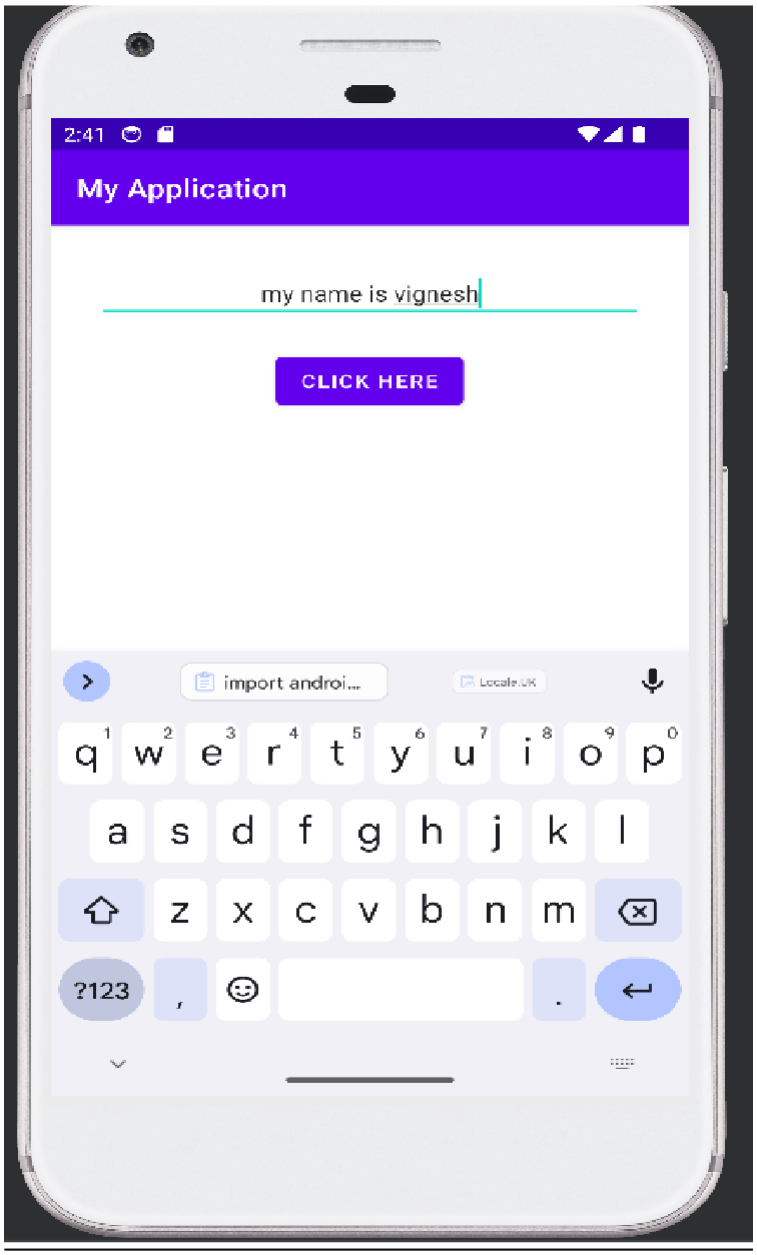
```
                    // if No error is found then only it will  run
                    if(i!=TextToSpeech.ERROR){
                        // To Choose language of speech
                        textToSpeech.setLanguage(Locale.UK);
                    }
                }
            });

            // Adding OnClickListener
            btnText.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {

        textToSpeech.speak(Text.getText().toString(),TextToSpeech.QUEUE_FLUSH,null);
                }
            });

        }
}
```

OUTPUT:



RESULT:

Hence an application with text to speech conversion executed successfully.