Speech Recognition and Synthesis

Implementation using namespace System.Speech with WinForms in C#

Name: Rakshita Singh

Roll no:21549

FY MSc Computer Science Sem 2

❖ Objective

In this project we will be learning how to synthesize a user’s audio and text inputs to perform speech recognition, text to speech and speech to text conversion using WinForms C#.

❖ Introduction

Speech recognition: It is the process of converting a voice into digital data. Speech recognition's purpose is to identify the words of the speaker. Here, the program needs a huge dictionary to identify the speaker's words.

Speech synthesis: Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech computer or speech synthesizer, and can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech.

This project aims to create voice recognition and synthesis WinForm applications that can recognize speech, convert text to speech and speech to text. To achieve this the two primary requirements are to know about WinForms and their event handling mechanisms along with the namespace System.Speech.Recognition and System.Speech.Synthesis.

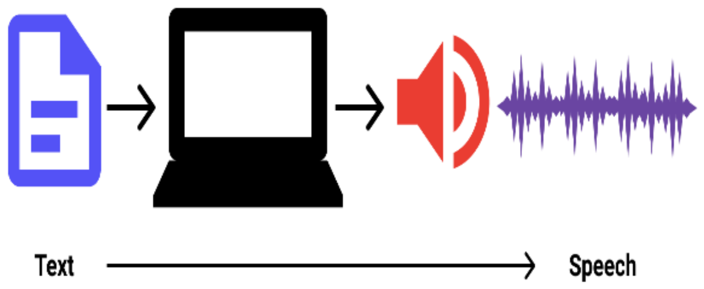
❖ Modules in the project:

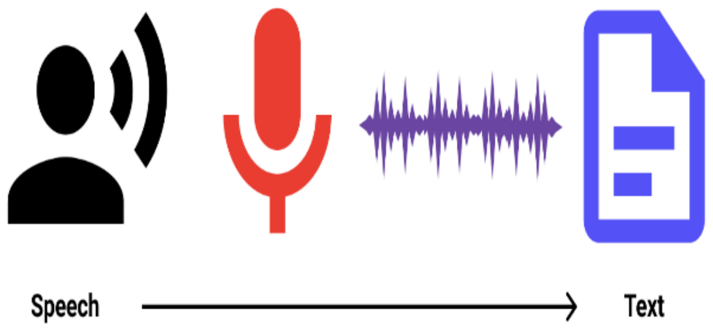
The project is divided into 3 parts with each having a WinForm application that performs a different task as follows:

1.Speech recognition

2.Text to Speech

3.Speech to text

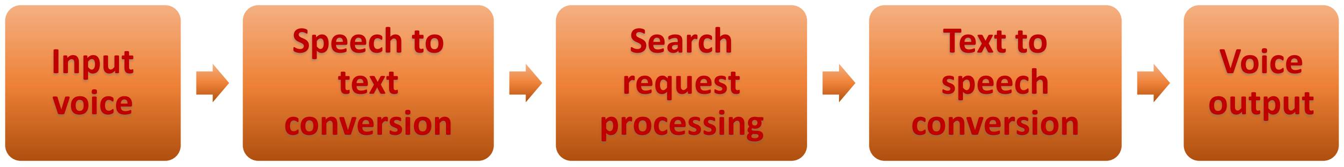




❖ Functionalities of Project and work flow of applications (Module wise):

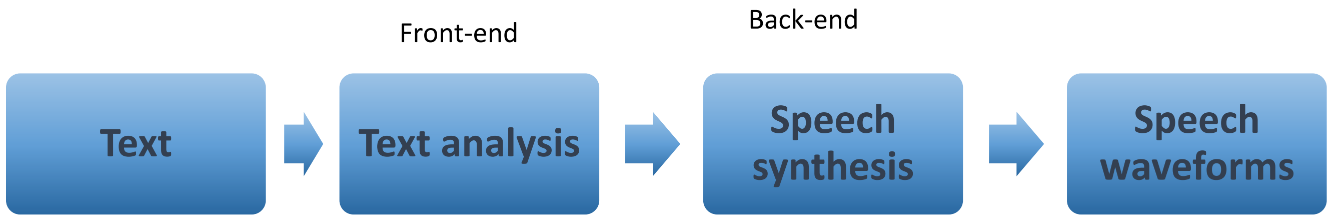
1. Speech recognition

In this application, as soon as it is launched the command recognition is available and the user may give several commands to the form and the application replies accordingly. It makes use of a pre-existing set of commands and its pre-defined replies.


Input voice
Speech to text conversion
Search request processing
Text to speech conversion
Voice output

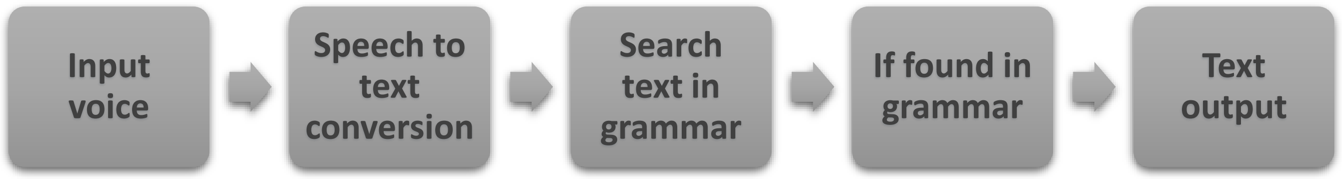
2. Text to speech

This form has a textbox with some input text lines along with button such as start, stop, resume, pause, along with a selection menu to select voice (Male/Female) and two track bars to change the volume and speed of speaker.



3. Speech to text

For this one, we will have a text box to display the output and two buttons start and stop which make the application start listening and interpreting the commands in order to give the required output. It makes use of a pre-loaded grammar to recognize the command given to it.



❖ Technology:

➢ Technical feasibility:

The system runs on multiple user environments.

- Operating System: windows/Linux/mac.
- Front end: Windows Form Application C#
- Back end: C#

The project is technically feasible since it makes use of existing System.Speech namespace to synthesize and recognize speech.

➢ Components:

➢ Windows Form Application:

Windows Forms is a UI framework for building Windows desktop apps. It provides one of the most productive ways to create desktop apps based on the visual designer provided in Visual Studio.

➢ Various classes from System.Speech.Recognition and System.Speech.Synthesis used within project:

- **SpeechRecognitionEngine**
  Namespace: System.Speech.Recognition
  Assembly: System.Speech.dll
  Provides the means to access and manage an in-process speech recognition engine.
  Initializes a new instance of the SpeechRecognitionEngineClass using the default speech recognizer for the system.

- **SpeechSynthesiser**
  Namespace: System.Speech.Synthesis
  Assembly: System.Speech.dll
  Provides access to the functionality of an installed speech synthesis engine

- **PromptBuilder**
  Namespace: System.Speech.Synthesis
  Assembly: System.Speech.dll
  Creates an empty Prompt object and provides methods for adding content, selecting voices, controlling voice attributes, and controlling the pronunciation of spoken words.

- **Grammar**
  Namespace: System.Speech.Recognition
  Assembly: System.Speech.dll
  Provides a mechanism for programmatically building the constraints for a speech recognition grammar.
  Initializes a new instance of the Grammar class.

- **Choices class**
  Namespace: System.Speech.Recognition
  Assembly: System.Speech.dll
  Represents a set of alternatives in the constraints of a speech recognition grammar.

1. SpeechRecognitionEngine

   Methods:

   1. SetInputToDefaultAudioDevice()
   2. LoadGrammarAsync(new Grammar(new GrammarBuilder(new Choices("* * * * * *"))))
   3. RecognizeAsync(RecognizeMode.Multiple)
   4. RecognizeAsyncCancel()
   5. RequestRecognizerUpdate()
   6. RecognizeAsyncStop()

   Events:

   7. <u>SpeechDetected</u>
      Raised when the SpeechRecognitionEngine detects input that it can identify as speech.

SRE_obj.SpeechDetected += new EventHandler<SpeechDetectedEventArgs>(_recognizer_SpeechRecognized);

   8. <u>SpeechRecognized</u>
      Raised when the SpeechRecognitionEngine receives input that matches any of its loaded and enabled Grammar objects.

SRE_obj.SpeechRecognized += new EventHandler<SpeechRecognizedEventArgs>(Default_SpeechRecognized);

2. SpeechSynthesizer

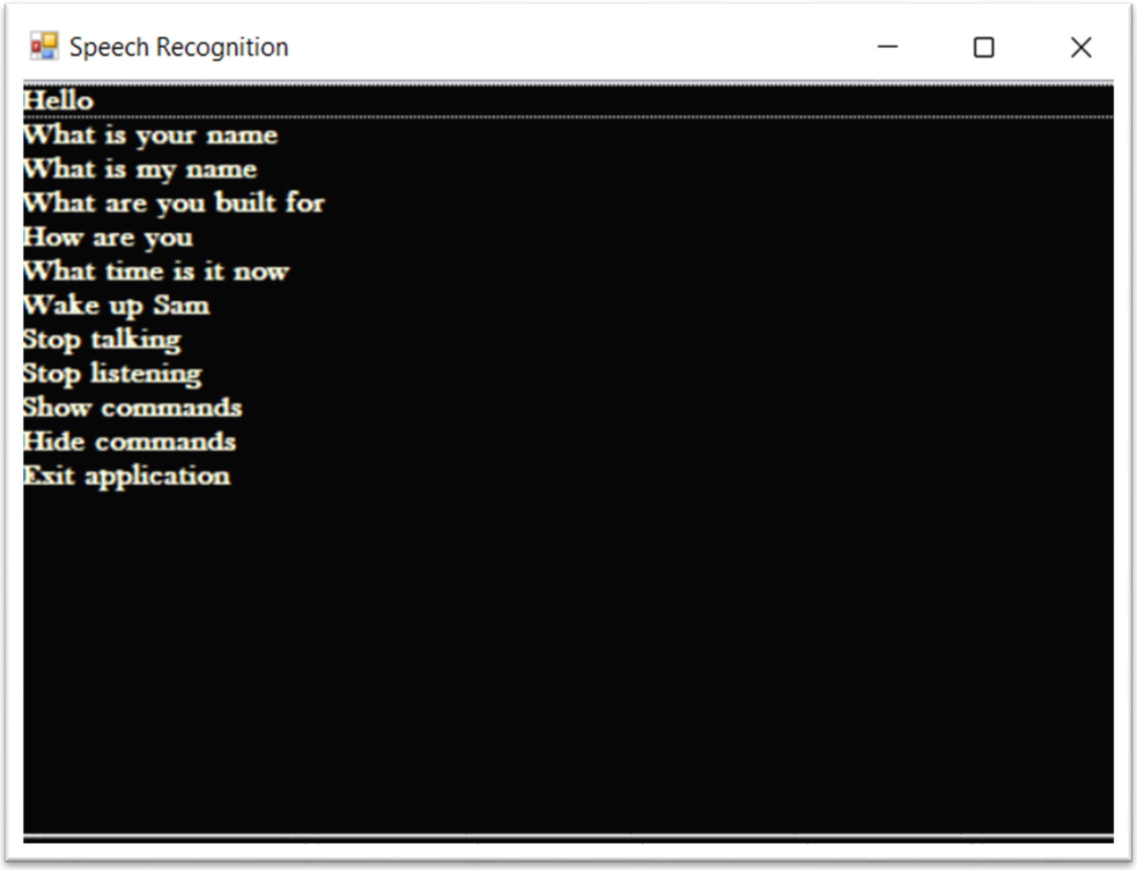
Methods:

1. SpeakAsync()
2. SpeakAsyncCancelAll()
3. SelectVoiceByHints(VoiceGender.Male)/SelectVoiceByHints(VoiceGender.Female)
4. Resume()
5. Pause()
6. Dispose()

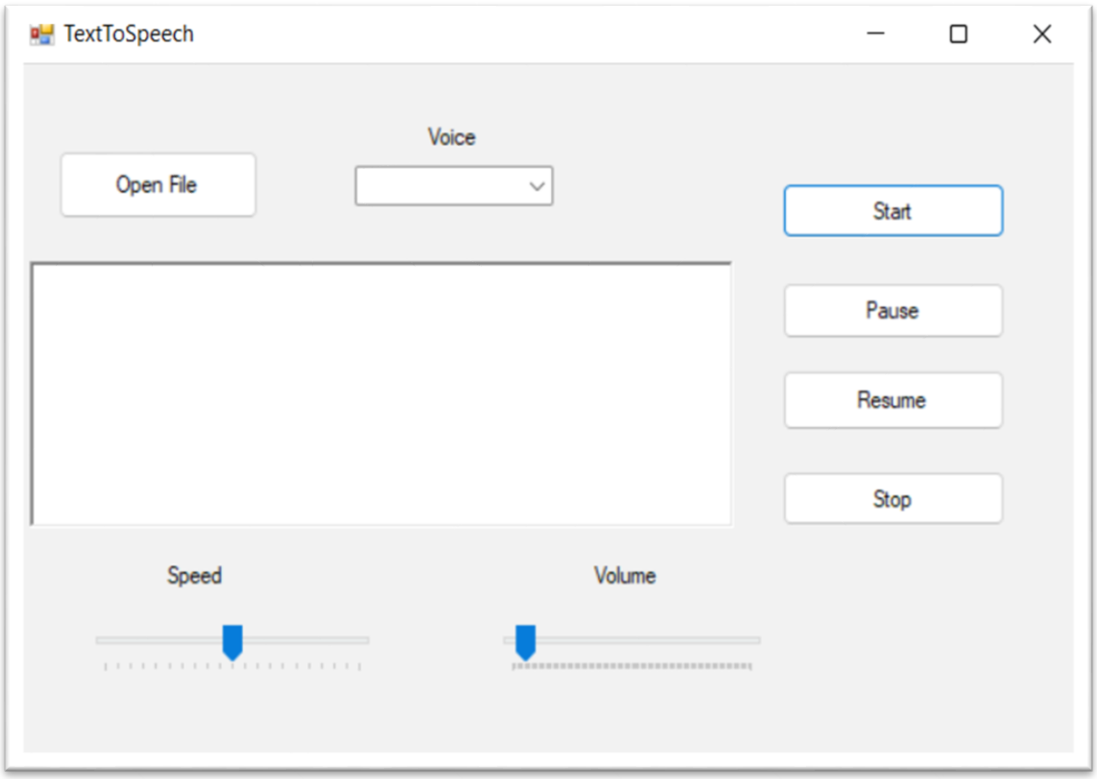
Property:

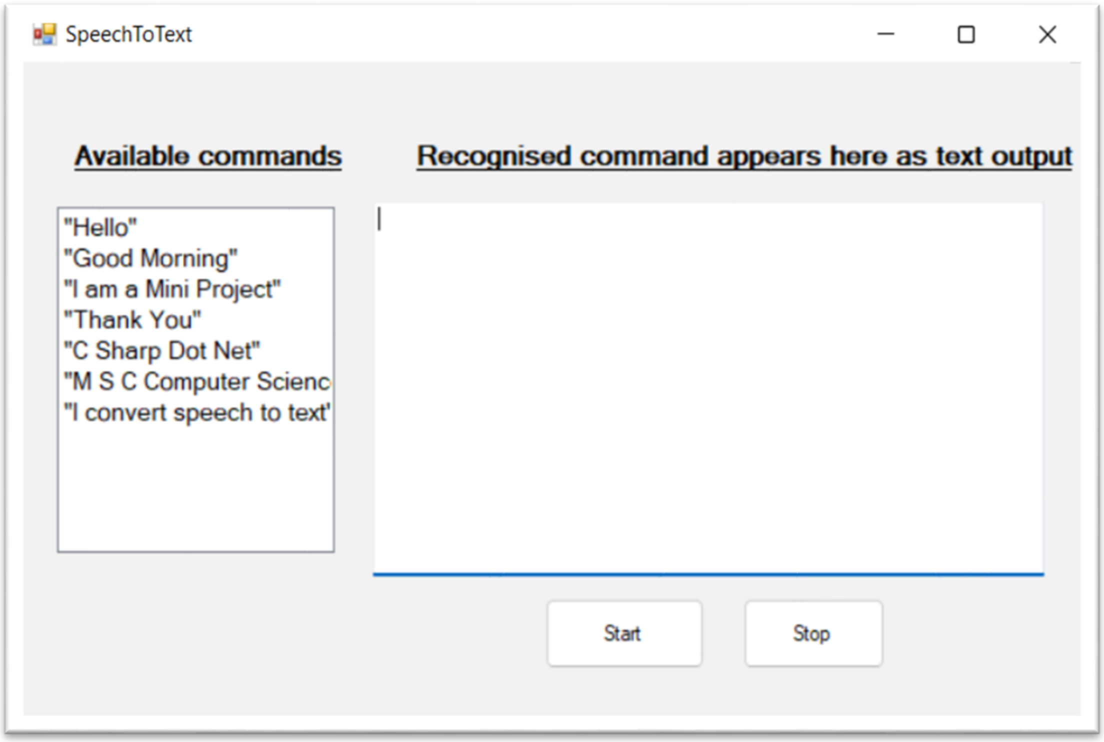
7. .State : SynthesizerState.Speaking/SynthesizerState.Paused
8. .Rate
9. .Volume

❖ Output Screens:



```
Hello
What is your name
What is my name
What are you built for
How are you
What time is it now
Wake up Sam
Stop talking
Stop listening
Show commands
Hide commands
Exit application
```





"Hello"
"Good Morning"
"I am a Mini Project"
"Thank You"
"C Sharp Dot Net"
"M S C Computer Scienc
"I convert speech to text'

❖ Future enhancement:

Real time speech synthesis using Google APIs available on Google Cloud or with the help NuGet Packages (Google.Cloud.Speech.V1).

❖ New Learnings:

1.Audio Synthesis in C#

2.Form UI designing