**ABSTRACT:**

We have long envisioned that one day computers will understand natural language and anticipate what we need, when and where we need it, and proactively complete tasks on our behalf. Physical interaction in order to provide commands or gain access to a computer system is now a history. Voice or speech simulated systems are a part of modern Smartphone culture. Automatic Speech Recognition is an important application of artificial intelligence. We believe that as computers become smaller and more ubiquitous [e.g., wearables and Internet of Things (IoT)], and the number of applications increases, both system-initiated and user-initiated task completion across various applications and web services will become indispensable for personal life management and work productivity. In this report, we give an overview of System assistant manager (SAM); describe the system architecture, Design, Implementation and technology used; and discuss its future potential to fully redefine human-computer interaction.

**INTRODUCTION:**

**Purpose:**

SAM - System Assistant Manager is an intelligent virtual assistance that is designed to help you get things done. It is a Software agent that perform tasks and services for an individual. SAM works across your devices and integrates with hundreds of applications to automate your work.

**Objective:**

The main motivation behind the SAM is to use voice queries and a natural language user interface to answer questions using information from google search engine as well as to automate your system applications.

**Scope:**

System Assistant Manager is Personal virtual assistant with voice controlled natural language interface and contextual awareness to help perform personal task for system users.

**Importance:**

Virtual assistance enabling digital experience are most promising consumer trends. As the technology improves, more advanced cognitive computing technologies are used to build virtual assistance that perform more complex task such as booking flights tickets. Thus, leveraging the importance of the virtual assistance

**Contribution:**

Our contribution to the field of virtual assistance is fundamental but provides deep insights. The Basic functionality of SAM is to minimize keyboard and mouse interaction for the users by providing voice enabled queries and searches. The most important feature of SAM is to store the user information and queries in a long file for predicting the user behaviours. It will help to understand user needs more accurately and provide recommendation on timely basis.

**Potential Customer:**

Potential customers can be anyone- Students, Professors, a general labour and so on. The system is for anyone who seek quick information, such as checking the current weather or quick task to be done or mail a friend.

**LITERATURE REVIEW:**

Virtual assistant is fundamentally transforming user behaviour by providing conversational artificial intelligent. Now, it has become more deep-rooted in our everyday user experience. It helps user with voice activated searches for finding the information and completing all manner of everyday task in more helpful way. With increase in advance Machine Learning algorithm it will becomes increasingly smarter.

In 2017, Many Tech industries such as google, Amazon, Apple and Microsoft have invested lot to improve the functionality and quality of Personal virtual assistance. It has been reported that Bank of America is developing an AI assistant allowing customer to use the bank mobile application by voice or text message. Staples plans to use AI to order office supplies by voice or text.

Below is the following prediction of improvement in the field:

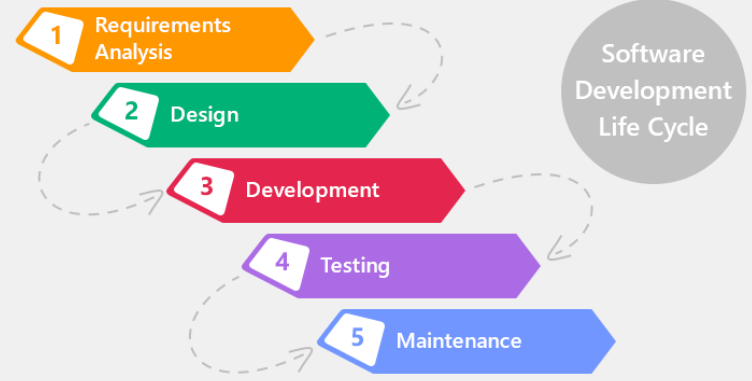
* It has been estimated that in 2017, **35.6 million** Americans will use a voice-activated assistant device at least once a month. That’s a jump of 128.9% over last year
* The number of users of virtual assistants in the U.S. will grow **23.1%** in 2017
* **60.5 million** Americans will use Siri, Cortana or another virtual assistant at least once a month in 2017
* **33 million** voice-activated devices will be installed in the U.S. by the end of 2017
* **42%** of consumers already use digital assistants, while 72% of business execs and 53% of millennials are using them
* **31%** of business executives believe “virtual personal assistants” will have the largest impact on their business, more than any other “AI-powered solution”
* By 2020, the average person will have more conversations with bots than with their spouse. **30%** of web browsing will be done by voice

From this, we can clearly say Voice enabled personal intelligent assistant is the fields that should not considered lightly.

**PROJECT DETAILS:**

**Project Methodology:**

* Implemented Waterfall Model, due to its simplicity.
* In Requirements phase, we first had a meeting with the professor, to jot down the software requirements.
* Designed our system architecture in the next phase for the prototype.
* In coding phase, we implemented the system using .NET 4.5 and C#.
* Tested the product with various test cases.



**REQUIREMENT ANALYSIS:**

In our Requirement Analysis Phase, we collaborated with professor in regular basis by conducting meetings to discuss about the functionality of the system and agreed to accept functionality such as Automation of the system, Web search and Logging the User behaviours.

**Functional Requirement Description:**  
Our project basically provides intelligent personal assistance services to the user. It handles functionality such as to search user queries in google search engine, logs user behaviour and automate system applications

For use case diagram refer **Appendix B**

**Non-Functional Requirement Description:**

Non-Functional requirements place constraints on how well the system or  
software will perform in certain conditions.

**Availability:**  
Availability refers to the percentage of time the system is up and running  
correctly. Since we are using Google search engine for information for the prototype. Our system shall exceed 99 Percentage uptimes to process the user request and respond to the user quickly.  
**Reliability:**  
Reliability refers to the probability that system will perform its required  
function for a specified interval under stated conditions. System should  
include resistance to failure. Our system performs the core functionality will zero failure rate and makes our application more reliable.  
**Maintainability:**  
Maintainability measures ability to make changes quickly and cost  
effectively. It should include extension with new functionality, deleting  
unwanted capabilities and restructuring. Our application uses open/close principle so that our application is open to extension and closed to modification.  
**Usability:**  
Usability refers to how the user can easily understand the system. In our  
application we included GUI user interface with ability to provide input through text and speech and provide the result in voice response in elegant way

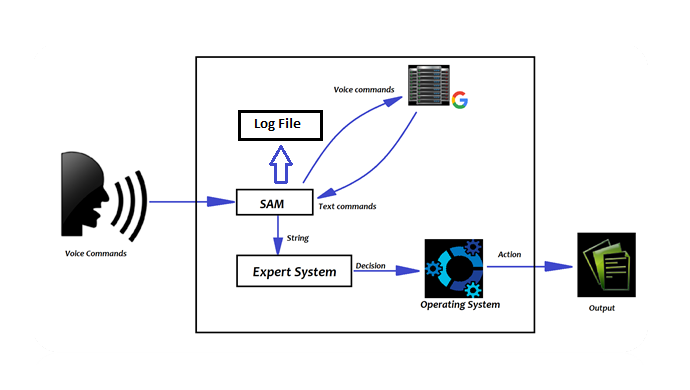
**SYSTEM DESIGN:**

**System overview:**

The application has two interconnected subsystems.

* The Conversion System: It is responsible for Speech-to-text and Text-to-speech conversion.
* The Expert system: It is responsible to make smart decision given the command by user.

The Entire System finally interacts with the OS, for kernel level processing to produce the output. Log File is used to collect user information and user behaviour.



**User activities:**

* User will use Natural language user interface to provide inputs queries.
* The Conversion System parse the input to text format using google speech API.
* The Expert system checks the user command and makes decisions based on the data present.
* If the Expert system has no information about the user commands, it will use google search to provide the result.
* The Output of the Queries will be in text or speech format.

For User interface refer **Appendix A.**

**Altamash can you choose good heading**

* The System is designed into two main component, for UML diagrams please refer to Appendix B, C and D.
* In **Appendix B**, we have displayed the process through use case diagram. The overall procedure is shown in **Appendix C**, using Sequence diagram.
* Finally, How the data flows through the system is shown in **Appendix D**, through another UML diagram, ie DataFlow Diagram.
* The first component is for Interaction with the user, that is SAM component which interacts with Text and audio.
* The decision making part is given to another component which is Expert System. Expert system takes all the decision and Requests the Operating System for the actions.

**IMPLEMENTATION DETAILS:**

**Technology Used:**

This section focuses on technology used to implement the application.

NET 4.5 Framework:

A programming infrastructure used for building, deploying, and running applications and services such as desktop applications and Web services. It is most widely used and easy to understand.

C# Programming Language:

We used powerful and flexible programming language C# to build our system. It provides wide collection of library, API and support.

Google Cloud Speech API:

It enables developer to convert audio to text by applying powerful neural networks model in easy to use API. You can transcribe the text of users dictating to an application’s microphone, enable command-and-control through voice, or transcribe audio files, among many other use cases.

Visual Studio 2017:

It is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web apps, web services and mobile apps.

**Problems Faced:**

1. Designing powerful and elegant UI for customer interaction.
2. Using Accurate and multi-language supported Speech Recognition API.
3. For Using Free Google speech Recognition API, we need to create account and authorize credit card for fundamentals features.

**TESTING DETAILS:**

Unit Testing:  
In the first round of testing, each module is submitted to assessments that  
focus on specific units or components of the software to determine each  
module is working as planned. Each unit is referred as function.

For Unit Testing refer **Appendix E**  
Integration Testing:  
Each function is combined into single unit and tests them as a group. Our  
application has been tested as whole and founded out the integration is  
working perfectly.

For Integration Testing refer **Appendix F**

System Testing:  
Here, complete application is tested as whole. We have tested our application, No abnormal behaviour has been found.

For System Testing refer **Appendix G**

**CONCLUSION:**

**\*\*\*\*\*\*\*NEED TO COMPLETE BY SMIRITI\*\*\*\*\*\*\*\*\*\*\*\*\***

**Existing System:**

In existing system, Voice Assistant applications performs web based actions. Our application SAM not only takes web based actions but also performs system based actions such as running a program, playing music etc. It also maintains a Log file of all user activities, that can be used for Data analysis. Unlike Google Voice, SAM warns us if we are not connected to the internet**.**

**Future Work:**

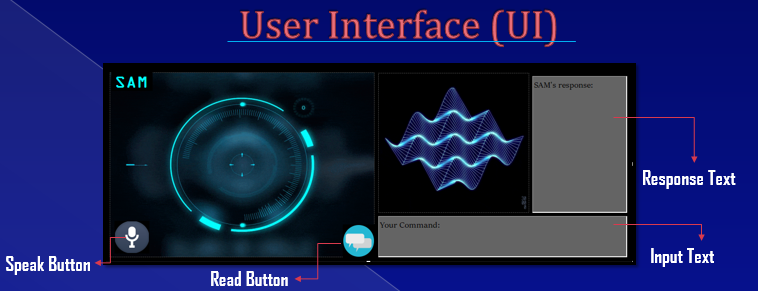
We can extend our project by implementing Multiple Language support such as Hindi, Chinese, and French, Full Automation of the system through voice commands, System response in multiple language. Data Analysis on the log files for recommendation.

**REFERENCE:**

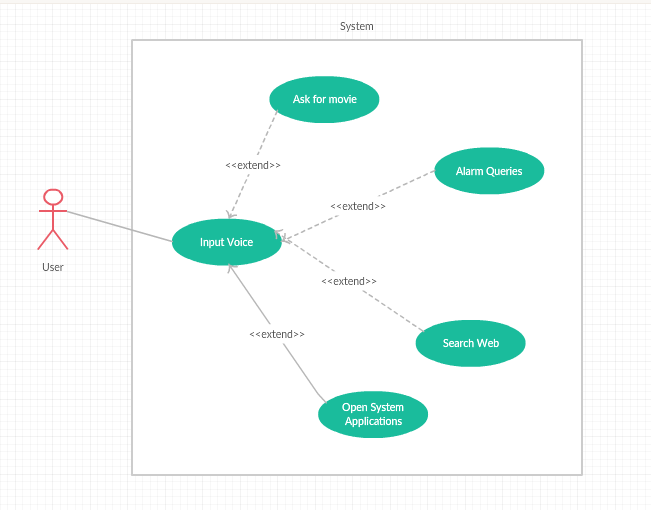
<https://www.pcmag.com/article/354371/what-are-virtual-assistants-and-what-can-you-do-with-them>

https://www.forbes.com/sites/gilpress/2017/05/15/ai-by-the-numbers-33-facts-and-forecasts-about-chatbots-and-voice-assistants/

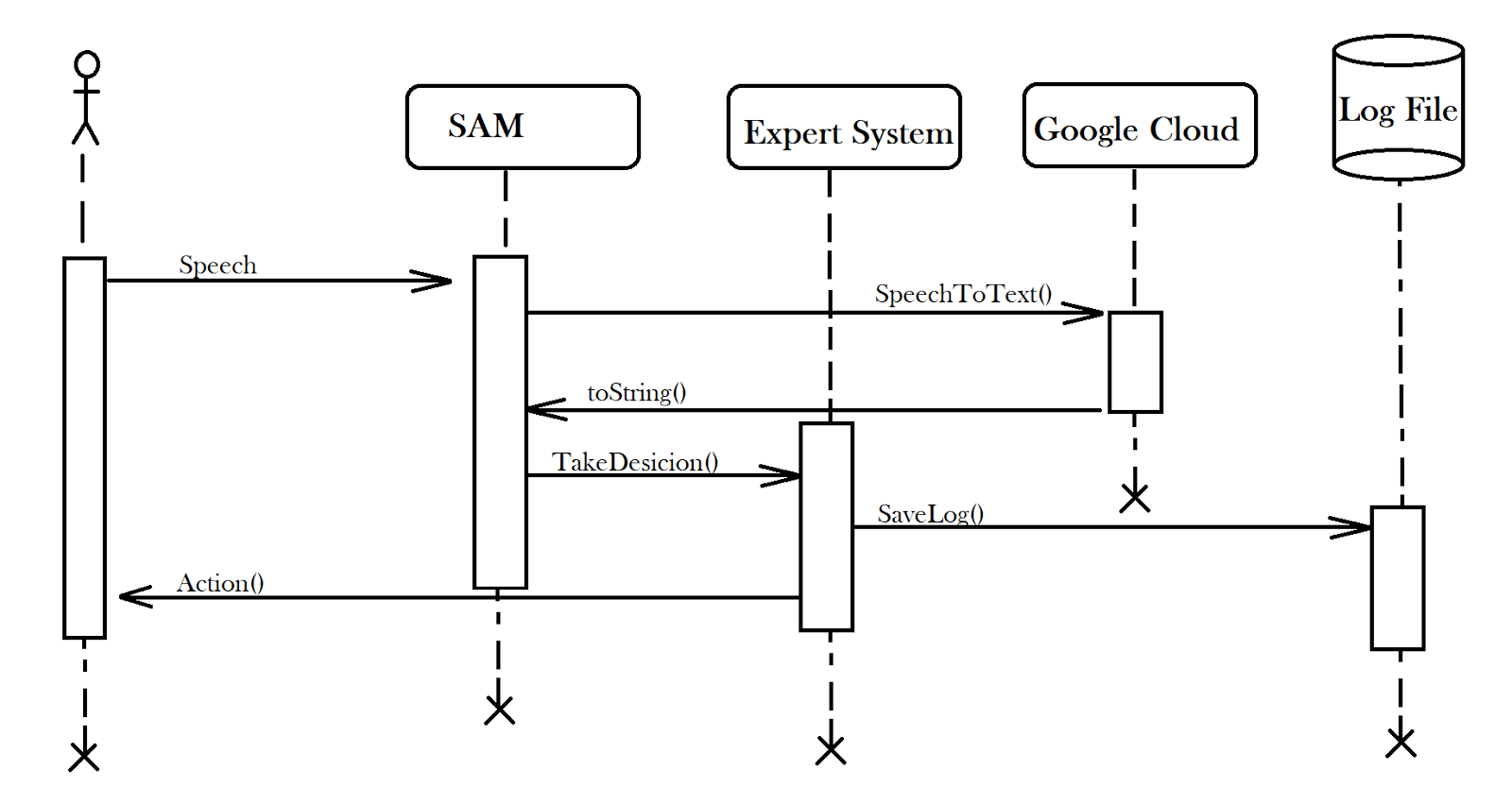
**APPENDIX A: USER INTERFACE**

****

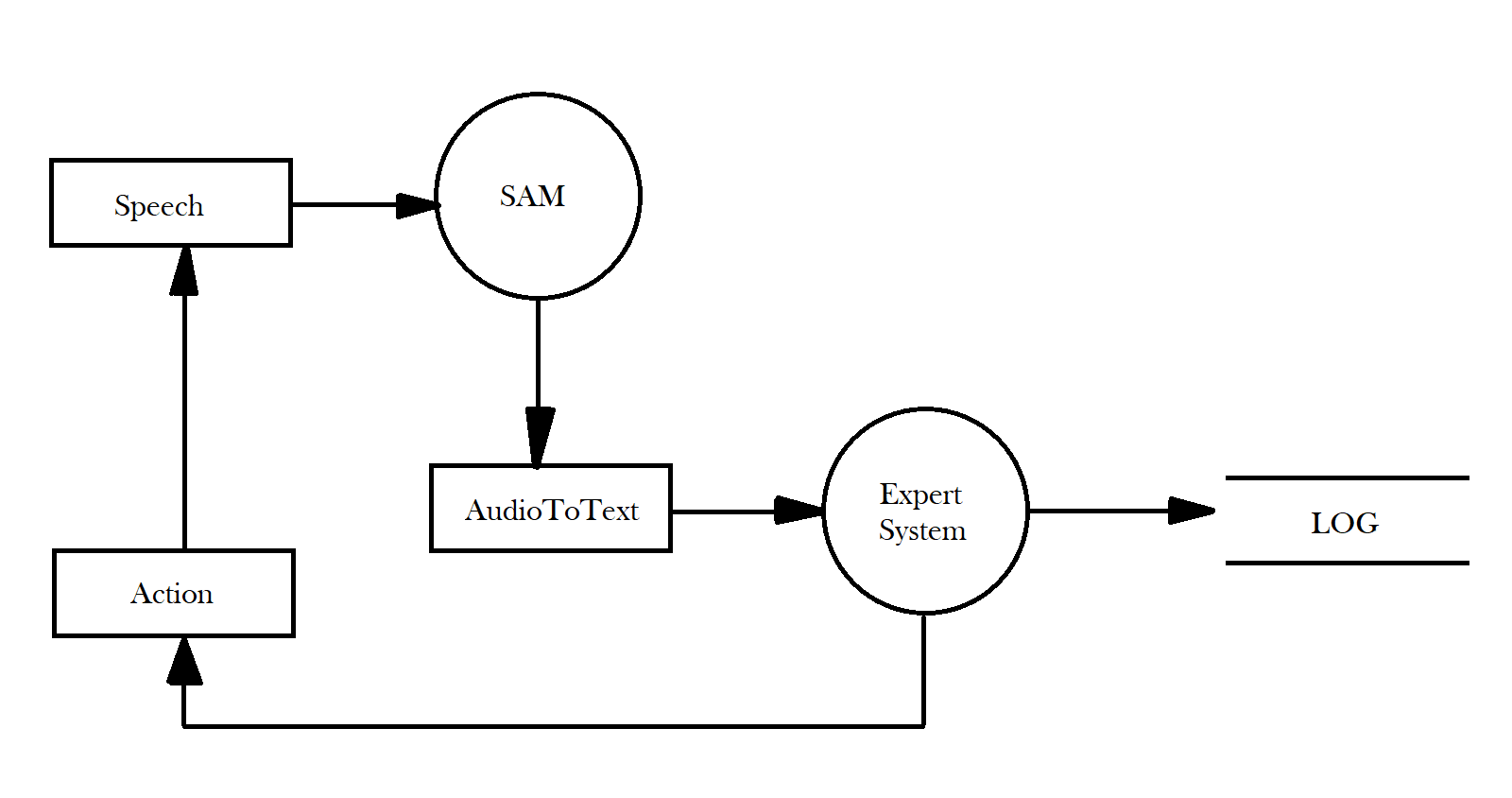
**APPENDIX B: USE CASE DIAGRAM FOR THE APPLICATION**

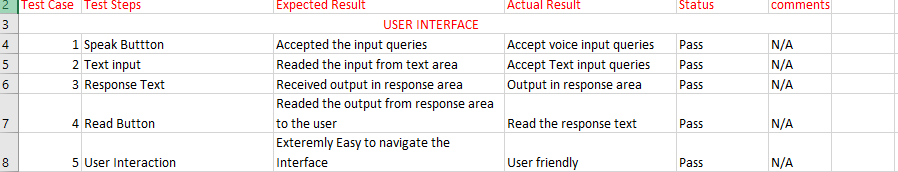
****

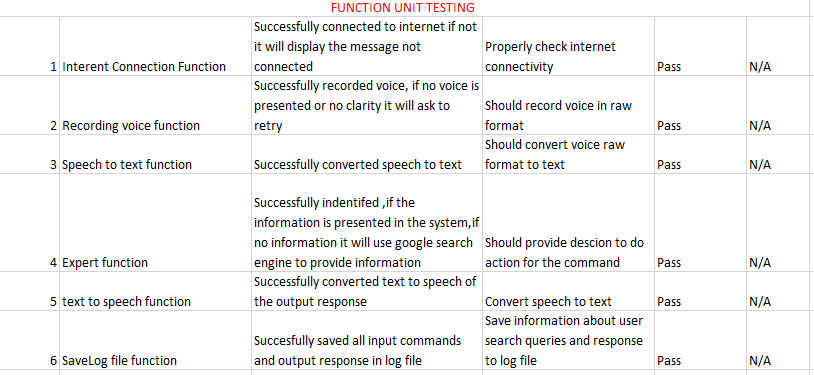
**APPENDIX C:  SEQUENCE DIAGRAM FOR THE APPLICATION**

****

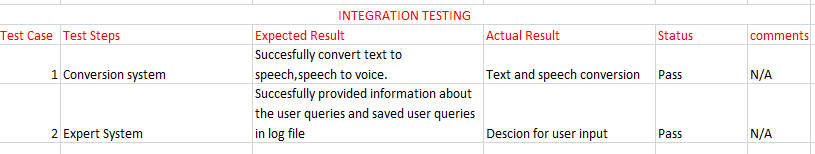
**APPENDIX D: DATAFLOW DIAGRAM FOR THE APPLICATION**

****

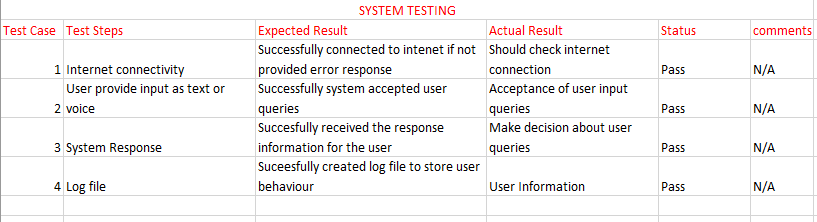
**APPENDIX E: TEST CASE: UNIT TESTING**



**APPENDIX F: TEST CASE: INTEGRATION TESTING**



**APPENDIX G: TEST CASE: SYSTEM TESTING**

****

**APPENDIX H: SNAPSHOT OF THE PROJECT**

**Check internet connection:**

           if (CheckForInternetConnection())

           {

               //Internet works. SAM greets.

               samText.Text = "";

               Say(greetings[randomChoice(greetings)]);

           }

           else

               Say("Your internet is not connected. I may not fully be able to help you!");

           theSpeaker.Volume = 100;

           theSpeaker.Rate = -1;

**Event handler for recording voice**

//Event handler for recording voice

       void waveIn\_DataAvailable(object sender, WaveInEventArgs e)

       {

           bwp.AddSamples(e.Buffer, 0, e.BytesRecorded);

       }

       private void pictureBox1\_Click(object sender, EventArgs e)

       {

       }

       private void button2\_Click(object sender, EventArgs e)

       {

           Expert(); //Speaks

       }

       private void richTextBox1\_TextChanged(object sender, EventArgs e)

       {

       }

       private void mouseDownEvent(object sender, MouseEventArgs e)

       {

           //When the mouse is pressed... Listen to the audio

           waveIn.StartRecording();

       }

**Saving speech into raw file:**

   //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Saving the speech into a raw file \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

           //When the mouse is released

           //stop recodring and save the file (Overwrtie if exist)

           waveIn.StopRecording();

           if (File.Exists("audio.raw"))

               File.Delete("audio.raw");

           //Create the raw audio file, with the given hz

           writer = new WaveFileWriter(output, waveIn.WaveFormat);

           byte[] buffer = new byte[bwp.BufferLength];

           int offset = 0;

           int count = bwp.BufferLength;

           var read = bwp.Read(buffer, offset, count);

           if (count > 0)

           {

               writer.Write(buffer, offset, read);

           }

           waveIn.Dispose();

           waveIn = null;

           writer.Close();

           writer = null;

           // \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* RETRY

           waveOut = new WaveOut();

           waveIn = new WaveIn();

           //configuring the settings for the audio file

           waveIn.DataAvailable += new EventHandler<WaveInEventArgs>(waveIn\_DataAvailable);

           waveIn.WaveFormat = new NAudio.Wave.WaveFormat(16000, 1);           //16000hz

           bwp = new BufferedWaveProvider(waveIn.WaveFormat);

           bwp.DiscardOnBufferOverflow = true;                         //Discard data if overflowed.

**Speech to text conversion:**

   // \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* speech to text conversion\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

           if (File.Exists("audio.raw"))

           {

               //create a speech client, (Google cloud speech API)

               var speech = SpeechClient.Create();

               //Start recognizing the audio

               var response = speech.Recognize(new RecognitionConfig()

               {

                   Encoding = RecognitionConfig.Types.AudioEncoding.Linear16,

                   SampleRateHertz = 16000,

                   LanguageCode = "en",

               }, RecognitionAudio.FromFile("audio.raw"));

               //Initializing the text

               userText.Text = "";

               //Get each response in text

               foreach (var result in response.Results)

               {

                   foreach (var alternative in result.Alternatives)

                   {

                       userText.Text = alternative.Transcript + userText.Text;

                   }

               }

               //if nothing was said in the audio

               if (userText.Text.Length == 0)

                   userText.Text = "No Data ";

           }

           else

           {

               //if the audio file was not found

               userText.Text = "Audio File Missing ";

           }

           this.Expert();

}

**Expert system to make decision:**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* EXPERT SYSTEM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

           // NOTE : Treat all string as a lower case.

       private void Expert()

       {

           query = userText.Text.ToLower();

           //Facebook

           if (query == "hi" || query == "hey" || query ==  "hello" || query == "how are you" || query == "what is up" || query == "how you doing")

           {

               Say(greetings[randomChoice(greetings)]);

           }

           else if (query.Contains("who are you") || query == "who is this" || query.Contains("what can you do") || query == "what are you" || query.Contains("how can you help me") || query.Contains("how can i use you") || query.Contains("what can you do for me") || query.Contains("introduce yourself") || query.Contains("who is sam"))

           {

               Say("Hello, My developers named me Sam, a System Assisted Manager. I can help you with anything. I can automate your system, open programs, tell you weather, time, news and much more. I am here to help, Just ask!");

           }

           else if (query.Contains("open word") || query == "microsoft word" || query.Contains("open ms word") || query.Contains("write a word document") || query.Contains("open microsoft word"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start(@"C:\Program Files\Microsoft Office\Office16\WINWORD.exe");

           }

           else if (query.Contains("open music player") || query == "music player" || query.Contains("open media player") || query.Contains("play a music") || query.Contains("play a song") || query.Contains("play music"))

           {

               Say(approves[randomChoice(approves)]);

               Say("Please choose a song.");

               Process.Start(@"C:\Program Files (x86)\Windows Media Player\wmplayer.exe");

           }

           else if (query.Contains("open browser") || query.Contains("browser") || query.Contains("open internet") || query.Contains("internet") || query.Contains("open web") || query.Contains("web"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://www.google.com");

           }

           else if (query.Contains("what is the time")|| query.Contains("what time is it") || query.Contains("time now"))

           {

               Say("The time is " + DateTime.Now.ToString("h:mm tt"));

           }

           else if (query.Contains("what is the day") || query.Contains("what day is it") || query.Contains("date") || query.Contains("what is today"))

           {

               Say("Today is " + DateTime.Now.ToString("M/d/yyyy"));

           }

           else if (query.Contains("open facebook") || query.Contains("facebook") || query.Contains("fb") || query.Contains("face book"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://www.facebook.com/");

           }

           //Youtube

           else if (query.Contains("open youtube") || query.Contains("youtube") || query.Contains("yt") || query.Contains("you tube"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://www.youtube.com/");

           }

           //Twitter

           else if (query.Contains("open twitter") || query.Contains("twitter"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://twitter.com/");

           }

           //IG

           else if (query.Contains("open instagram") || query.Contains("instagram") || query.Contains("ig"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://www.instagram.com/");

           }

           //Google

           else if (query.Contains("open google") || query.Contains("google"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://www.google.com");

           }

           else if (query.Contains("hide") || query.Contains("minimize"))

           {

               if (this.WindowState == FormWindowState.Normal)

               {

                   Say(approves[randomChoice(approves)]);

                   this.WindowState = FormWindowState.Minimized;

               }

           }

           else if (query.Contains("maximize") || query.Contains("show up") || query.Contains("where are you"))

           {

               if (this.WindowState == FormWindowState.Minimized)

               {

                   Say("Here I am.");

                   this.WindowState = FormWindowState.Normal;

               }

           }

           else if (query.Contains("thank you"))

           {

               Say("My Pleasure!");

           }

           else if (query.Contains("who is pooya"))

           {

               Say("He is a Professor at Windsor University. Very generous at grading students. Does not likes to be called professor or doctor, Just Pooya!");

           }

           //Less imp

           else if (query.Contains("what") || query.Contains("who") || query.Contains("which") || query.Contains("how") || query.Contains("why") || query.Contains("when"))

           {

               Say(approves[randomChoice(approves)]);

               Process.Start("https://www.google.ca/search?q=" + query);

           }

           // Least important

           else if (query.Contains("sam"))

           {

               if (query.Contains("turn off") || query.Contains("bye") || query.Contains("shutdown") || query.Contains("shut down") || query.Contains("goodbye"))

               {

                   Say(farewell[randomChoice(farewell)]);

                   Application.Exit();

               }

           }

           //When all else fails

           else

           {

               Say("let me find it for you");

               Process.Start("https://www.google.ca/search?q=" + query);

           }

**SAM Output response to Speech:**

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* SAM SPeaks \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

       private void Say(string speakString)

       {

           waveBox.Image = thePic;

           textPhrase = "";

           textPhrase = speakString;

           pb.AppendText(textPhrase, PromptRate.Medium);

           theSpeaker.SpeakAsync(pb);

           samText.Text += textPhrase + "\n";

           pb.ClearContent();

           theSpeaker.SpeakCompleted += new EventHandler<SpeakCompletedEventArgs>(voiceComplete);

       }

       private void voiceComplete(object sender, EventArgs e)

       {

           //When the speak is complete

           waveBox.Image = null;

       }

       //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Random string generator \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

       private int randomChoice(String[] theList)

       {

           Random r = new Random();

           //Random number from 0 to the length of the list

           return (r.Next(0, theList.Length));

       }

       public static bool CheckForInternetConnection()

       {

           try

           {

               using (var client = new System.Net.WebClient())

               {

                   using (client.OpenRead("http://clients3.google.com/generate\_204"))

                   {

                       return true;

                   }

               }

           }

           catch

           {

               return false;

           }

       }

**Logging user information and behaviour**

private void saveLogFile()

       {

           string path = @"C:\ProgramData\SAM\_logFile.txt";

           string userName = System.Security.Principal.WindowsIdentity.GetCurrent().Name;  //gets the user name

           //if the file does not exists.

           if (!File.Exists(path))

           {

               File.Create(path).Dispose();

               using (TextWriter tw = new StreamWriter(path))

               {

                   tw.WriteLine("\*\*\* Log File \*\*\*\*");

                   tw.Close();

               }

           }

           //keep writing

           else if (File.Exists(path))

           {

               using (TextWriter tw = new StreamWriter(path,true)) //Parameter: path, append : bool

               {

                   tw.WriteLine("Name: "+userName+"\t"+ "Command: " + query +"\t"+ "Time: " + DateTime.Now.ToString("h:mm tt")+"\t"+ "Day: " + DateTime.Now.ToString("M/d/yyyy"));

                   tw.Close();

               }

           }

       }