**SPEECH RECOGNITION PROJECT USING ARDUINO**

**A MINI PROJECT REPORT**

***Submitted by***

**VARUN PANDI R (922117104056)**

**VASANTHA KUMAR S (922117104057)**

**YASHWANTH BALAN A (922117104062)**

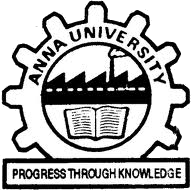
**BACHELOR OF ENGINEERING**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

**SSM INSTITUTE OF ENGINEERING AND TECHNOLOGY**

**DINDIGUL-624002**

****

**ANNA UNIVERSITY :: CHENNAI 600 025**

**APRIL /MAY 2020**

**ANNA UNIVERSITY :: CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report “**SPEECH RECOGNITION PROJECT USING ARDUINO”** is the bonafide work of**, VARUN PANDI.R(922117104056), VASANTHA KUMAR.S(922117104057), YASHWANTH BALAN.A(922117104062)** who carried out the project work under my provision.

SIGNATURE SIGNATURE

**Dr.V.SHUNMUGHAVEL M.E.,Ph.D., G.MANJARI M.TECH..,**

**HEAD OF THE DEPARTMENT AP/CSE**

Computer Science and Engineering Computer Science and Engineering

SSM Institute of Engg. & Tech., SSM Institute of Engg. & Tech.,

Dindigul-624002. Dindigul-624002.

Submitted for Viva-Voice Examination on ……………………….

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

First and foremost, we would like to express our deep sense of gratitude to most honorable Chairman and Management Trustee **Mr.P.S.Velusamy,** our honorable Secretary and Management Trustees **Mr.C.Kandaswamy,** our beloved Executive Directors and Management Trustees **Mr.K.Shanmugavel** and **Mr.K.Sivaraj** of the **SSM Institute of Engineering and Technology** for providing us with necessary facilities during the course of study.

We extend our solemn gratitude to our Principal **Dr D. Senthil Kumaran** for encouragement and support. We feel immensely pleased to express our sincere thanks to Head of the Department **Dr V. Shunmughavel,** Professor Department of Computer Science and Engineering, for this timely support to our activities.

We express our sincere thanks to our guide **Ms G.MANJARI, AP/CSE** for her guidance throughout our project work. Also, we express our thanks to the Faculty member of our Department, Non-teaching staff members and my dear friends for their moral support, help and encouragement towards the successful completion of the project.

We are most indebted to our parents, with whose support, resulted in making our dreams of becoming successful graduates, a reality. We are quite confident that our project works stands testimony to fact that hard work will bear enjoyable fruits not only to the individuals concerned of the scientists have made the lives of the brethren more comfortable.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** |  |
| **1.** | **INTRODUCTION**  **(Domain ,Problem and Objective of your Project)** |  |
|  | **1.1** **OVERVIEW OF THE PROJECT**  **1.2 AIM**   * 1. **1.3 PROBLEM DESCRIPTION** |  |
| **2.** | **SYSTEM ANALYSIS**  **2.1 Existing system**  **2.2 Proposed system**  **SYSTEM DESIGN**  **2.3 Flow Chart Diagram**  **2.4 Block Diagram**  **2.5 Circuit Diagram** |  |
| **3.** | **Requirements**  **Description of the requirements for the project**  **3.1 Software Requirement**  **3.2 Hardware Requirement** |  |
| **4.** | **Implementation**  **4.1 Module Description**  **4.2 Sample Code**  **4.3 Module Screen Shots** |  |
| **5.** | **Testing**  **(The Appropriate testing done in your project)** |  |
| **6.** | [**CONCLUTION AND FUTURE ENHANCEMENT**](#_Toc3864845) |  |
| **7.** | **REFERENCES** |  |

**ABSTRAC-T**

To Control any devices through other devices are very challenging for today’s Technology. The Following system is based on a Speech Recognition project using Arduino.

By this module any device can be controlled by using our Human vocal communication language. As a part of this module few LED’s are to be controlled by using Speech Recognition Technique using Arduino.

**CHAPTER 1**

**INTRODUCTION**

* 1. **OVERVIEW OF THE PROJECT**

Speech recognition is an interdisciplinary subfield of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language into text by computers.

In this Project Speech Recognition is implemented with a Microprocessor called Arduino DUE Board and to control LED’s by using human vocal communication attested with a Speech Software.

The main goal of this project is to control the LED’s using a person’s Speech command. This Module initially works with the Microprocessor Module called Aduino Board and a Special Software which is an Application of Speech Recognition called BitVoicer. BitVoicer is a speech recognition application that enables simple devices, with low processing power, to become voice-operated. To do that, BitVoicer uses the PC processing power to analyze audio streams, identify the sentences present in these streams and send commands to a microcontroller connected to it These Modules helps to implement a Person’s Speech and to Recognize it. Initially an Arduino DUE Board (ESP8266) is initiated and Connected to a System. Then the BitVoicer is Installed in that System and Connected to the Arduino module. These operations are done using a Software called Arduino IDE. The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. ... The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. The Arduino board is connected to a computer via USB, where it connects with the Arduino development environment (IDE). The user writes the Arduino code in the IDE, then uploads it to the microcontroller which executes the code, interacting with inputs and outputs such as sensors, motors, and lights. In this module the LED’s are controlled by the users Voice Commands and it acts based on the Command. The user will give his command to Arduino and it will operate by using the Bitvoicer and hence finally the LED’s performs according the user’s commands.

* 1. **Aim**

Our aim is to Control the LED’s connected to a microprocessor Arduino DUE board in which the LED’s will be controlled and react according to the users command.

* 1. **PROBLEM DESCRIPTION**

Controlling a device using other device is very easy and efficiently useful in today’s technology but controlling a device using the human vocal communication language is not efficiently added in today’s technology this was the main cause of this project which helps us to develop a small module by using which we can control some small LED’s through our speech.

This is our major idea to control the LED’s using the Speech Recognition technique and also to control the Home and Outdoor Automation using Speech Recognition in Future.

This Project definitely gives us a solution that all devices can be controlled through our vocal communication and gives a tremendous growth in Information Technology and Other Related Fields.

**CHAPTER 2**

**SYSTEM ANALYSIS**

System Analysis is a process by which we attribute process or goals to a human activity, determine how well those purposes are being achieved and specify the requirements of the various tools and techniques that are to be used within the system if the system performances are to be achieved.

**2.1 EXISTING SYSTEM**

The current Speech Recognition Techniques are not able to match and fulfil the requirements of the user/developer. The Main Dis-Advantage of the Existing System is it does not support all the connectable devices and customization was difficult. Hence, those projects are developed to its next state.

**2.2 PROPOSED SYSTEM**

The development of the new system contains the following activities, which able any user to control the Microprocessor or any other device through his/her voice and speech. This project will overcome all the previous faults and it will fulfil the user’s requirements.

This project will be initialized through various new technologies and software’s which bind to its new features and will be very useful to the Technical world.

**SYSTEM DESIGN**

System design is the process of defining the elements of a system such as the architecture, modules and components the different interfaces of those components and the data that goes through that system. It is meant to satisfy specific needs and requirements of a business or organization through the engineering of a coherent and well-running system.

**2.3 List of Figures**

**2.3.1 Flow Chart Diagram**

**Speech Command**

**Arduino Memory**

**Microphone**

**BitVoicer**

**Microprocessor-Arduino DUE Board**

**Control LED’s**

**2.3.2 Circuit Diagram**

Bread Board

OhmR3

OhmR2

OhmR1

**-**

Microphone

Arduino Board

LED1 LED2 LED3

Bit Voicer

System

**CHAPTER 3**

**3.1 SYSTEM REQUIREMENTS**

**3.1.1 Hardware Requirement**

**A System with..,**

* Processor : Intel Pentium IV
* RAM : 4-6 GB
* Monitor : 17 inch Color Monitor
* Keyboard : 108 Keys
* Mouse : Optical Mouse
* Hard Disk : 1 TB

**Other Hardware Requirements**

* Microprocessor : Arduino DUE(ESP8266)
* Microphone : Spark-fun Electret Breakout Microphone
* Wires (both male and female joints)
* Colour LED’s
* 330 OHM Resistors
* Bread Board (Mini)

**3.1.2 Software Requirement**

* Front End/Language : Arduino Sketch-C/C++
* Back End/Speech Software : BitVoicer
* Operating System : Windows 7/10

**3.1.3 Software and Language Used**

**3.1.3.1 Arduino Sketch/Code**

Since the launch of the Arduino open-source platform, the brand has established themselves at the center of an expansive open-source community. The Arduino ecosystem is comprised of a diverse combination of hardware and software. The versatility of Arduino and its simple interface makes it a leading choice for a wide range of users around the world from hobbyists, designers, and artists to product prototypes. The Arduino board is connected to a computer via USB, where it connects with the Arduino development environment (IDE). The user writes the Arduino code in the IDE, then uploads it to the microcontroller which executes the code, interacting with inputs and outputs such as [sensors](https://https/www.circuito.io/blog/arduino-sensors-explained/), motors, and lights.

Arduino code is written in C++ with an addition of special methods and functions, which we’ll mention later on. C++ is a human-readable programming language. When you create a ‘sketch’ (the name given to Arduino code files), it is processed and compiled to machine language.

The Arduino Integrated Development Environment (IDE) is the main text editing program used for Arduino programming. It is where you’ll be typing up your code before uploading it to the board you want to program. Arduino code is referred to as **sketches**.

Essentially, the IDE translates and compiles your sketches into code that Arduino can understand. Once your Arduino code is compiled it's then uploaded to the board's memory

**3.1.3.2 LANGUAGE USED IN ARDUINO IDE**

The Arduino has it's own IDE where you write the program. In case you're interested in knowing what language is used to program the Arduino Microcontroller, it is programmed using C or C++ language. However, the C or C++ code does not drive the Microcontroller.

**3.1.3.2.1 C/C++**

C and C++ are among the most powerful languages you can find. Those are quite fast and stable, and thus are a good choice for microcontrollers. The compiler used to transform code into object files is avr-g++. Then, a program called avrdude is used to upload the program into the Arduino microcontroller.

C++ is a [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language) created by [**Bjarne Stroustrup**](https://en.wikipedia.org/wiki/Bjarne_Stroustrup) as an extension of the [**C programming language**](https://en.wikipedia.org/wiki/C_(programming_language)), or "C with [Classes](https://en.wikipedia.org/wiki/Class_(programming))". The language has expanded significantly over time, and modern C++ now has [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [generic](https://en.wikipedia.org/wiki/Generic_programming), and [functional](https://en.wikipedia.org/wiki/Functional_programming) features in addition to facilities for [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) [memory](https://en.wikipedia.org/wiki/Memory_(computing)) manipulation. It is almost always implemented as a [compiled language](https://en.wikipedia.org/wiki/Compiled_language), and many vendors provide [C++ compilers](https://en.wikipedia.org/wiki/List_of_compilers#C.2B.2B_compilers), including the [Free Software Foundation](https://en.wikipedia.org/wiki/Free_Software_Foundation), [LLVM](https://en.wikipedia.org/wiki/LLVM), [Microsoft](https://en.wikipedia.org/wiki/Microsoft), [Intel](https://en.wikipedia.org/wiki/Intel), [Oracle](https://en.wikipedia.org/wiki/Oracle_Developer_Studio), and [IBM](https://en.wikipedia.org/wiki/IBM), so it is available on many platforms.

C++ was designed with a bias toward [system programming](https://en.wikipedia.org/wiki/System_programming) and [embedded](https://en.wikipedia.org/wiki/Embedded_software), resource-constrained software and large systems, with [performance](https://en.wikipedia.org/wiki/Performance_(software)), efficiency, and flexibility of use as its design highlights.

**3.1.3.3 Back End/Speech Software**

**3.1.3.3.1 Bit Voicer**

BitVoicer is a speech recognition application that enables simple devices, with low processing power, to become voice-operated. To do that, BitVoicer uses the PC processing power to analyze audio streams, identify the sentences present in these streams and send commands to a microcontroller connected to it.

The main BitVoicer features are--It can process audio captured by the microcontroller or by the computer's audio adapter. You write the text and BitVoicer recognizes the speech. It supports more than 17 world languages from 26 countries.

Input Devices are those capable of capturing, digitizing and sending audio streams to the server. When BitVoicer Server identifies an Input Device, it assigns one exclusive Speech Recognition Engine (SRE) to that device. SREs constantly analyze all audio streams sent to the server and when a predefined sentence is identified, BitVoicer Server performs the actions specified by the user. These actions are called commands and can be used to start other applications, synthesize speech, play audio files or send data (commands) to Output and Mixed Devices. The user can define one or more commands for each sentence. The user can also define the order in which the commands will be executed, the time interval between them, and which Output or Mixed Devices are the targets of the commands. That is, with one single Input Device license you can control multiple Output Devices. Lastly, there are the Mixed Devices that are capable of acting as Input and Output Devices. Application developers can retrieve the results of recognition operations from the server and exchange data with client devices through Windows Communication Foundation (WCF) services. For .NET development, there is also a .NET integration library available.

**CHAPTER 4**

**IMPLEMENTATION**

**4.1 MODULE DESCRIPTION**

A "module" in a open-project is a high-level description of a functional area, consisting of a group of [processes](http://www.project-open.com/en/list-processes) describing the functionality of the module and a group of [packages](http://www.project-open.com/en/list-packages) implementing the functionality.

**4.1.2 Software Module Description**

**Arduino IDE:**

1. Install and Run Arduino IDE
2. Click New Sketch in File Menu
3. Complete the required coding for LED control
4. Compile the Code
5. Set the code ready to Upload

**Bit Voicer:**

1. Install and Run Bitvoicer from BitSophia Technology
2. Click Add new Sentences
3. Give the required commands as sentences in the sentence box
4. Set the type of the inputs
5. Load the sentences
6. Start execution-
7. Give the Commands as per sentences
8. Stop

**Hardware Module Description:**

**Entire Module Description:**

1. Take the small size Bread Board and place it centre
2. Connect the Arduino DUE to the Board
3. Initialize the Arduino Kit by connecting it to the system
4. Boot the Initializing Program
5. Connect the other parts of the module like Ohm resistor, LED’s and Microphone
6. Boot the required programs for the module
7. Connect a Backup device for saving the data’s
8. Start the process
9. Stop

**4.2 Sample Code**

**Arduino Sketch Codes**

#include <BitVoicer11.h>

//Instantiates the BitVoicerSerial class

BitVoicerSerial bvSerial = BitVoicerSerial();

//Stores true if the Audio Streaming Calibration tool

//is running

boolean sampleTest = false;

//Stores the data type retrieved by getData()

byte dataType = 0;

//Sets up the pins and default variables

int pinR = 3;

int pinY = 5;

int pinG = 6;

int lightLevel = 0;

void setup()

{

//Sets the analog reference to external (AREF pin)

//WARNING!!! If anything is conected to the AREF pin,

//this function MUST be called first. Otherwise, it will

//damage the board.

bvSerial.setAnalogReference(BV\_EXTERNAL);

//Sets up the microcontroller to perform faster analog reads

//on the specified pin

bvSerial.setAudioInput(0);

//Starts serial communication at 115200 bps

Serial.begin(115200);

//Sets up the pinModes

pinMode(pinR, OUTPUT);

pinMode(pinY, OUTPUT);

pinMode(pinG, OUTPUT);

}

void loop()

{

//Captures audio and sends it to BitVoicer if the Audio

//Streaming Calibration Tool is running

if (sampleTest == true)

{

//The value passed to the function is the time

//(in microseconds) that the function has to wait before

//performing the reading. It is used to achieve about

//8000 readings per second.

bvSerial.processAudio(46);

}

//Captures audio and sends it to BitVoicer if the Speech

//Recognition Engine is running

if (bvSerial.engineRunning)

{

//The value passed to the function is the time

//(in microseconds) that the function has to wait before

//performing the reading. It is used to achieve about

//8000 readings per second.

bvSerial.processAudio(46);

}

}

//This function runs every time serial data is available

//in the serial buffer after a loop

void serialEvent()

{

//Reads the serial buffer and stores the received data type

dataType = bvSerial.getData();

//Changes the value of sampleTest if the received data was

//the start/stop sampling command

if (dataType == BV\_COMMAND)

sampleTest = bvSerial.cmdData;

//Signals BitVoicer's Speech Recognition Engine to start

//listening to audio streams after the engineRunning status

//was received

if (dataType == BV\_STATUS && bvSerial.engineRunning == true)

bvSerial.startStopListening();

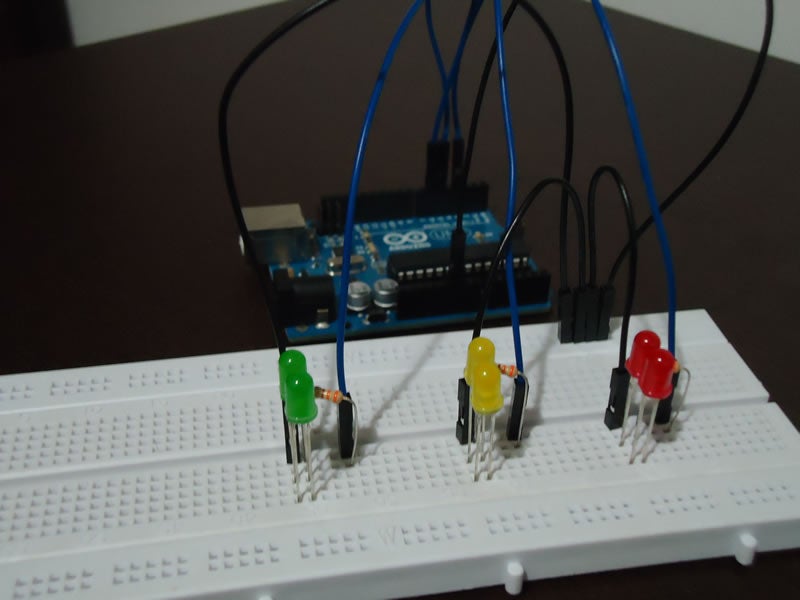
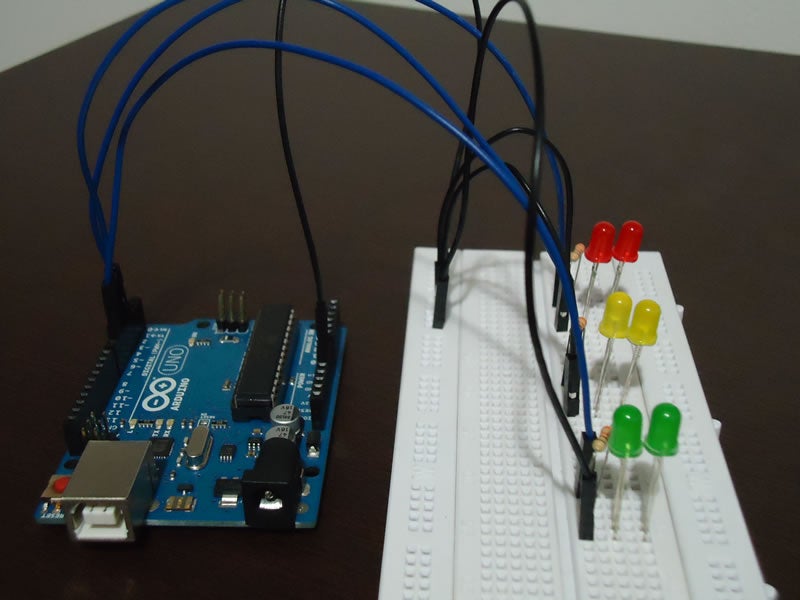
//Checks if the data type is the same as the one in the

//Voice Schema

if (dataType == BV\_STR)

setLEDs();

**4.3 Module Screenshot**

****

**CHAPTER 5**

**TESTING**

Testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to the process of executing a program or application with the intent of finding software bugs (errors or other defects). The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of executing software with the intent of ensuring it.

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement. Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during development process. In other words software testing is a verification and validation.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

* meets the requirements that guided its design and development,
* responds correctly to all kinds of inputs,
* performs its functions within an acceptable time, is sufficiently usable,
* can be installed and run in its intended environments, and
* achieves the general result its stakeholders desire.

**Speech Recognition Testing:**

The speech recognition test determines your ability to both hear and understand normal conversations. They measure the softest sound that your ear can pick up and how clearly you comprehend spoken words. Normally, the speech recognition test will let the specialist know the quietest speech that you can discern at least half of the time. The speech recognition test may be performed in either silent or noisy environments, depending on the specialist and their analysis of your hearing needs. Normally, one ear is tested at a time – not both at once.

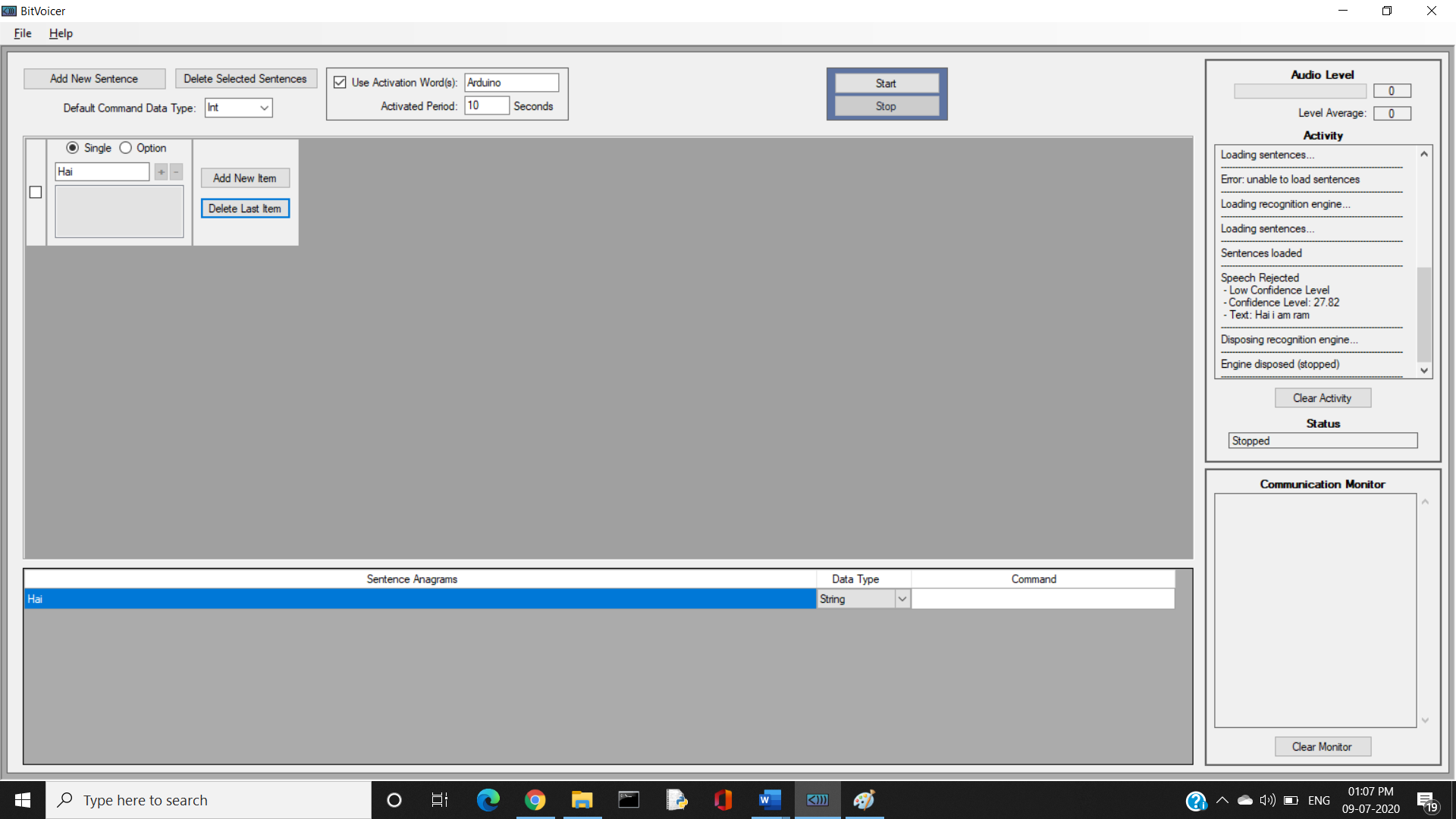
The test itself is very simple. The specialist will probably give you headphones and present a series of words to you, each time varying the loudness or intensity of the speech. They will ask you to repeat back the words. Generally, a list of pre-selected words are used for the test. Once you are hearing correctly only 50 percent of the time, the test will end and results will be determined.

In this way, they will be able to determine at what threshold of sound you can no longer understand speech very well. Though simple, the test can be frustrating for some if they experience a more severe hearing loss, as they will not be able to hear or repeat most of the words.

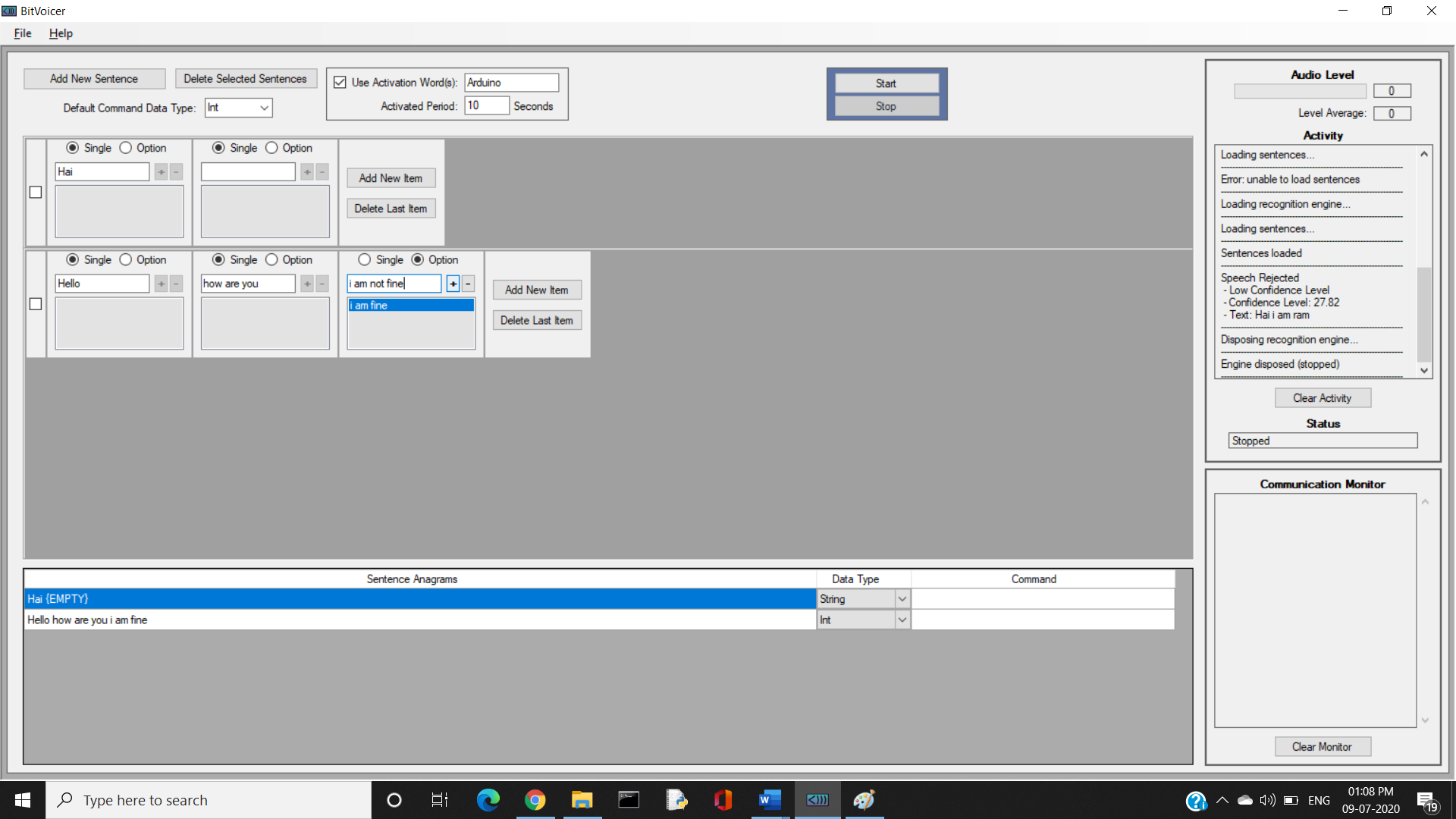
In this testing process the user’s speech is tested using the software called BitSophia. In this module various user’s voices are collected and saved and further tested foe the required stages.

**Testing Modes in BitVoicer:**

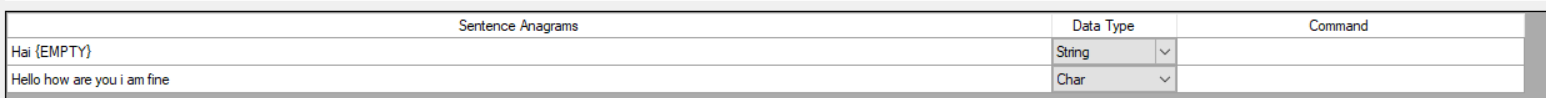
**Step 1:** Click Add new Sentences



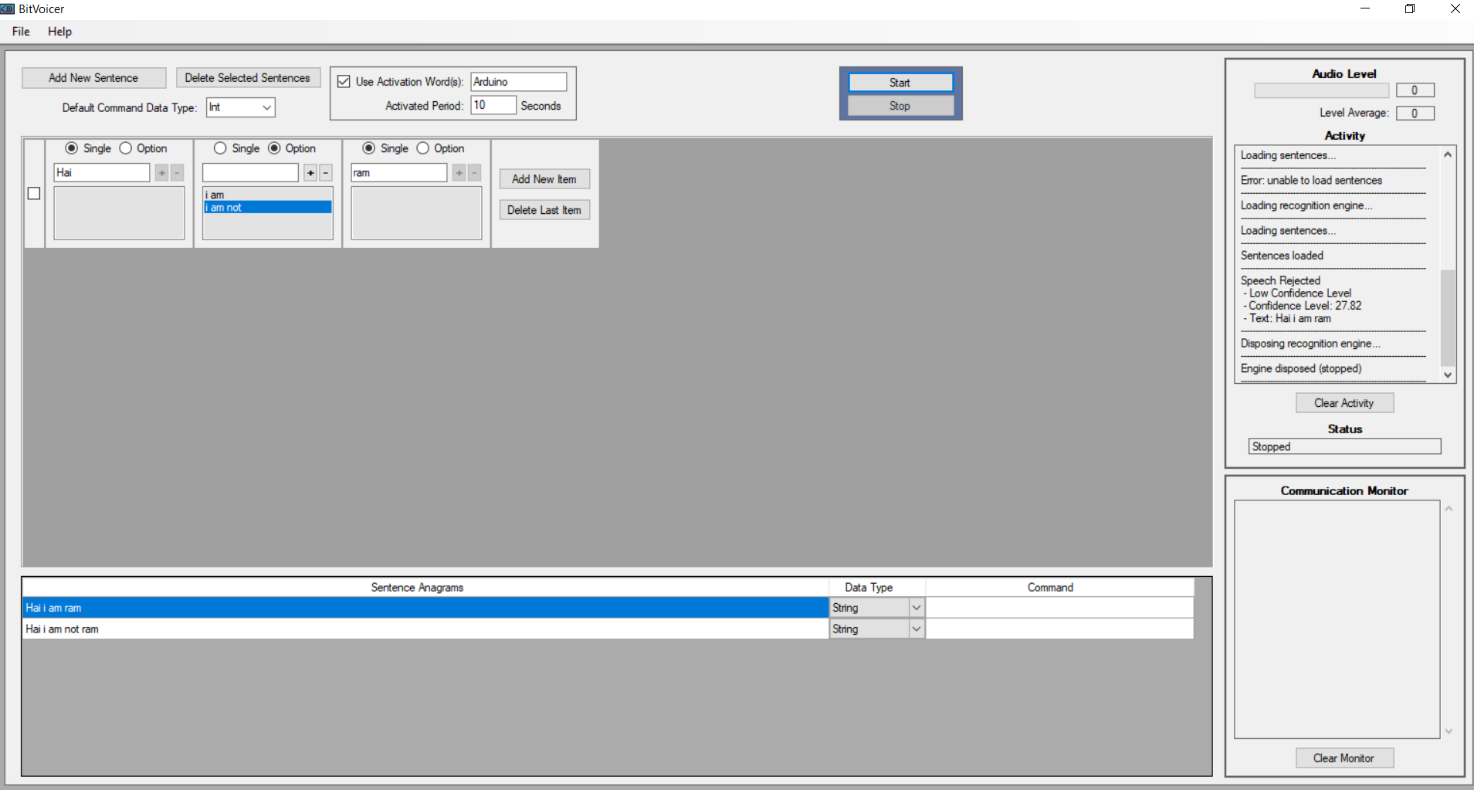
**Step 2:** Give the required commands as sentences in the sentence box



**Step 3:** Set the type of the inputs

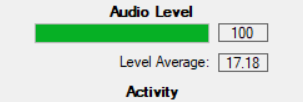
****

**Step 4:** Give complete sentences as per the Project and start testing

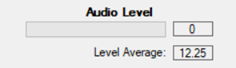
****

**Voice Level Testing:**

**Audio at Level Peak: Range 1:**

****

**Audio at Level Dull: Range 0:**

****

**CHAPTER 6**

**CONCLUSION AND FUTURE ENHANCEMENT**

**6.1 CONCLUSION**

Thus, the Speech Recognition project using Arduino has proved to be more effective and efficient in which it fulfilled all the user requirements which helps us to develop more in future works.

This project proved good for me as it provided practical knowledge of not only programming in HTML , C’C++ skills and now some extent knowledge about Arduino and Speech Recognition system, but also about all handling procedure related with “**SPEECH RECOGNITION PROJECT USING ARDUINO**”. It also provides knowledge about the latest technology used in developing Speech enabled application and Speech based technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently. This Project helps various domains of the latest technologies which are ready to build up the current fields to higher range.

This easily interrupts to the Current field of technology to its next level. Even more the main domain of this module is “Artificial Intelligence”, Because speech is the important aspect of AI.

Easier and faster Speech technologies could be developed and make it possible for all fields. This gives us better opportunities in the Internet Technology fields and give a guidance in future in developing projects independently.

**6.2 FUTURE WORKS**

The Analysis on Speech Recognition using Arduino can be even more developed by using the following techniques,

**Speaker Identification Robot using Artificial Intelligence:**

In this system, we are going to implement even more technologies to build up its state by using various domains like Artificial Intelligence where in this module the Speaker of the Module is identified and verified even more better by a Robot Module. This will improve the Security level of technologies to the next step.

Hence, these are our Future Enhancements which will lead to a Major Improvement in Current Technological World.

**APPENDICES**

**Back End Arduino Sketch Codes in C Programming:**

#include <BitVoicer11.h>

//Instantiates the BitVoicerSerial class

BitVoicerSerial bvSerial = BitVoicerSerial();

//Stores true if the Audio Streaming Calibration tool

//is running

boolean sampleTest = false;

//Stores the data type retrieved by getData()

byte dataType = 0;

//Sets up the pins and default variables

int pinR = 3;

int pinY = 5;

int pinG = 6;

int lightLevel = 0;

void setup()

{

//Sets the analog reference to external (AREF pin)

//WARNING!!! If anything is conected to the AREF pin,

//this function MUST be called first. Otherwise, it will

//damage the board.

bvSerial.setAnalogReference(BV\_EXTERNAL);

//Sets up the microcontroller to perform faster analog reads

//on the specified pin

bvSerial.setAudioInput(0);

//Starts serial communication at 115200 bps

Serial.begin(115200);

//Sets up the pinModes

pinMode(pinR, OUTPUT);

pinMode(pinY, OUTPUT);

pinMode(pinG, OUTPUT);

}

void loop()

{

//Captures audio and sends it to BitVoicer if the Audio

//Streaming Calibration Tool is running

if (sampleTest == true)

{

//The value passed to the function is the time

//(in microseconds) that the function has to wait before

//performing the reading. It is used to achieve about

//8000 readings per second.

bvSerial.processAudio(46);

}

//Captures audio and sends it to BitVoicer if the Speech

//Recognition Engine is running

if (bvSerial.engineRunning)

{

//The value passed to the function is the time

//(in microseconds) that the function has to wait before

//performing the reading. It is used to achieve about

//8000 readings per second.

bvSerial.processAudio(46);

}

}

//This function runs every time serial data is available

//in the serial buffer after a loop

void serialEvent()

{

//Reads the serial buffer and stores the received data type

dataType = bvSerial.getData();

//Changes the value of sampleTest if the received data was

//the start/stop sampling command

if (dataType == BV\_COMMAND)

sampleTest = bvSerial.cmdData;

//Signals BitVoicer's Speech Recognition Engine to start

//listening to audio streams after the engineRunning status

//was received

if (dataType == BV\_STATUS && bvSerial.engineRunning == true)

bvSerial.startStopListening();

//Checks if the data type is the same as the one in the

//Voice Schema

if (dataType == BV\_STR)

setLEDs();

}

//Performs the LED changes according to the value in

//bvSerial.strData

void setLEDs()

{

if (bvSerial.strData == "wake")

{

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

digitalWrite(pinR, HIGH);

digitalWrite(pinY, HIGH);

digitalWrite(pinG, HIGH);

delay(200);

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

delay(200);

digitalWrite(pinR, HIGH);

digitalWrite(pinY, HIGH);

digitalWrite(pinG, HIGH);

delay(200);

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

delay(200);

digitalWrite(pinR, HIGH);

digitalWrite(pinY, HIGH);

digitalWrite(pinG, HIGH);

delay(200);

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

lightLevel = 0;

}

else if (bvSerial.strData == "sleep")

{

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

digitalWrite(pinR, HIGH);

digitalWrite(pinY, HIGH);

digitalWrite(pinG, HIGH);

delay(200);

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

delay(200);

digitalWrite(pinR, HIGH);

digitalWrite(pinY, HIGH);

digitalWrite(pinG, HIGH);

delay(200);

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

lightLevel = 0;

}

else if (bvSerial.strData == "RH")

{

digitalWrite(pinR, HIGH);

lightLevel = 255;

}

else if (bvSerial.strData == "RL")

{

digitalWrite(pinR, LOW);

lightLevel = 0;

}

else if (bvSerial.strData == "YH")

{

digitalWrite(pinY, HIGH);

lightLevel = 255;

}

else if (bvSerial.strData == "YL")

{

digitalWrite(pinY, LOW);

lightLevel = 0;

}

else if (bvSerial.strData == "GH")

{

digitalWrite(pinG, HIGH);

lightLevel = 255;

}

else if (bvSerial.strData == "GL")

{

digitalWrite(pinG, LOW);

lightLevel = 0;

}

else if (bvSerial.strData == "ALLON")

{

digitalWrite(pinR, HIGH);

digitalWrite(pinY, HIGH);

digitalWrite(pinG, HIGH);

lightLevel = 255;

}

else if (bvSerial.strData == "ALLOFF")

{

digitalWrite(pinR, LOW);

digitalWrite(pinY, LOW);

digitalWrite(pinG, LOW);

lightLevel = 0;

}

else if (bvSerial.strData == "brighter")

{

if (lightLevel < 255)

{

lightLevel += 85;

analogWrite(pinR, lightLevel);

analogWrite(pinY, lightLevel);

analogWrite(pinG, lightLevel);

}

}

else if (bvSerial.strData == "darker")

{

if (lightLevel > 0)

{

lightLevel -= 85;

analogWrite(pinR, lightLevel);

analogWrite(pinY, lightLevel);

analogWrite(pinG, lightLevel);

}

}

else

{

bvSerial.startStopListening();

bvSerial.sendToBV("ERROR:" + bvSerial.strData);

bvSerial.startStopListening();

}

}

**CHAPTER 7**

**REFERENCES**

1. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P. and Silovsky, J., 2011. The Kaldi speech recognition toolkit. In IEEE 2011 workshop on automatic speech recognition and understanding (No. CONF). IEEE Signal Processing Society.
2. Juang, Biing Hwang, and Laurence R. Rabiner. "Hidden Markov models for speech recognition." Technometrics 33, no. 3 (1991): 251-272.
3. Bahl, Lalit R., et al. "Performance of the IBM large vocabulary continuous speech recognition system on the ARPA Wall Street Journal task." 1995 International Conference on Acoustics, Speech, and Signal Processing. Vol. 1. IEEE, 1995.
4. Aubert, Xavier, et al. "Large vocabulary continuous speech recognition of Wall Street Journal data." Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing. Vol. 2. IEEE, 1994.
5. Base paper: <https://www.engpaper.com/speech-recognition-2019.htm>
6. <https://www.instructables.com/id/Speech-Recognition-with-BitVoicer-and-Arduino/>
7. <https://medium.com/electronza/speech-recognition-with-arduino-hardware-considerations-17a6c7b55c27>\*
8. Alhawiti, Khaled M. "Advances in artificial intelligence using speech recognition." World Academy of Science, Engineering and Technology, International
9. Journal of Computer, Electrical, Automation, Control and Information Engineering 9.6 (2015): 1351-1354.
10. Arduino Sketch and C: Arduino, Store Arduino. "Arduino." Arduino LLC (2015) and Johnson, Stephen C. Lint, a C program checker. Murray Hill: Bell Telephone Laboratories, 1977.