

## ACKNOWLEDGEMENT

We thank our principal **Thiru. S. Senthil Kumaran MSc, M. Phil.**, for the kind advice for this given to us in reference to the project.

We are extremely indebted to the Head of the Department **Mr. P. Sridhar M.E.**, who provided us great advice, timely help, acknowledgeable assistance and useful recommendations.

We extend our thanks to **Tmt. D. Juliet Thessalonica M.E.**, who guided us to finish our project in successful manner in the specified span of time.

We own our gratitude to ours Lecturers **Tmt. B. Kavitha M.E., Tmt. V. Manga M.E., Miss. V. Priya M.E.**, for their kind attention and valuable suggestions during the project.

We are also in great regard to our fellow students for their enthusiasm, during to the course of this project. Our special thanks to **Mr. M. Vairam (skilled Assistant)** for making this project a successful by providing excellent guidance.

## **ABSTRACT**

Home automation is one of the major growing industries that can change the way people live. Some of these home automation systems target those seeking luxury and sophisticated home automation platforms; others target those with special needs like the elderly and the disabled people. Typical wireless home automation system allows to control house hold appliances from a centralized control unit which is wireless. These appliances usually have to be specially designed to be compatible with each other and with the control unit for most commercially available home automation systems. The developed system can be integrated as a single portable unit and allows to control lights, fans, and turn ON or OFF any appliance that is plugged into a wall outlet. The system is portable and constructed in a way that is easy to install, configure, run, and maintain. The perfect user interface still does not exist at present and to build a good interface requires knowledge of both sociology and technology fields.

The problem lies with the situation of the elderly or disabled people, who cannot usually help themselves to move around, and might require external assistance. People who live alone might also need a helping hand at home. Therefore, a voice activated smart home appliance system is designed, so that the users can perform certain tasks by just the use of their voices, moreover, the system is designed to have a hand-held device so that the user can easily speak their commands. Having a Hand-Held device will make the system more user-friendly and portable.

# **CHAPTER 1**

## **INTRODUCTION**

Voice activated smart home appliance systems are somewhat different from ordinary homes, where the different smart devices in the presence of communications network being installed that allows the devices to communicate with each other. Integrated communication systems provide the facility for monitoring and managing the performance of the home, and offer the choice to support to the occupants for available facilities. The varieties of systems are installed in today's modern home such as central air conditioned and heating, fire and security alarms, and different other devices, such as televisions, lights, fans etc. These systems and devices usually exist in total isolation from each other. Such facility and control not only provide better control locally but also supports special needs, particularly services that support the elderly. Voice activated Smart home appliance technology also greatly improves the usability and functionality of any home. A Voice activated smart home appliance system allows saving money and the environment.

## **CHAPTER 2**

### **EXISTING SYSTEM**

The existing home automation system was developed by using Global system for mobile communication and ARM based technology. The home appliances are controlled using simple GSM based phone by sending SMS. ARM based architecture has lack of wireless function. The research provided here aims at studying the feasibility of implementing an SMS- based control of home appliances using the GSM technology without trying to access other local networks.

## **CHAPTER 3**

### **LITERATURE REVIEW**

Voice Controlled Home Automation Systems for Disabled people home automation is one of the major growing industries that can change the way people live. Typical wireless home automation system allows one to control house hold appliances from a centralized control unit which is wireless. The developed system can be integrated as a single portable unit and allows one to wirelessly control lights, fans etc. and turn ON or OFF any appliance that is plugged into a wall outlet. According to major companies that are involved in speech recognition researches, voice will be the primary interface between humans and machines in the near future.

Voice recognition based Wireless Home Automation Systems Home Automation industry is growing rapidly; this is fuelled by the need to provide supporting systems for the elderly and the disabled, especially those who live alone. This paper details the overall design of a wireless home automation system (WHAS) which has been built and implemented. The automation centres on recognition of voice commands. The home automation system is intended to control all lights and electrical appliances in a home or office using voice commands.

Android and Bluetooth Based Voice Controlled Wireless Smart Home System Now a days we use many electrical devices at homes, industries, offices, institutions that are controlled manually. To control all electrical devices, we need a lot of MAN POWER. If manpower increases maintenance cost also rises. This causes a disbenefit to the industry. So, to avoid these kinds of drawbacks we need some wireless controlling systems. One such wireless communication system to be used is Bluetooth communication system.

Arduino Based Voice Controlled Home Appliances Using Bluetooth In this paper a low cost and user friendly remote controlled home automation system is presented using Arduino board, Bluetooth module, smartphone. A smartphone application is used in the suggested system which allows the users to control up to 18 devices including home appliances and sensors using Bluetooth technology.

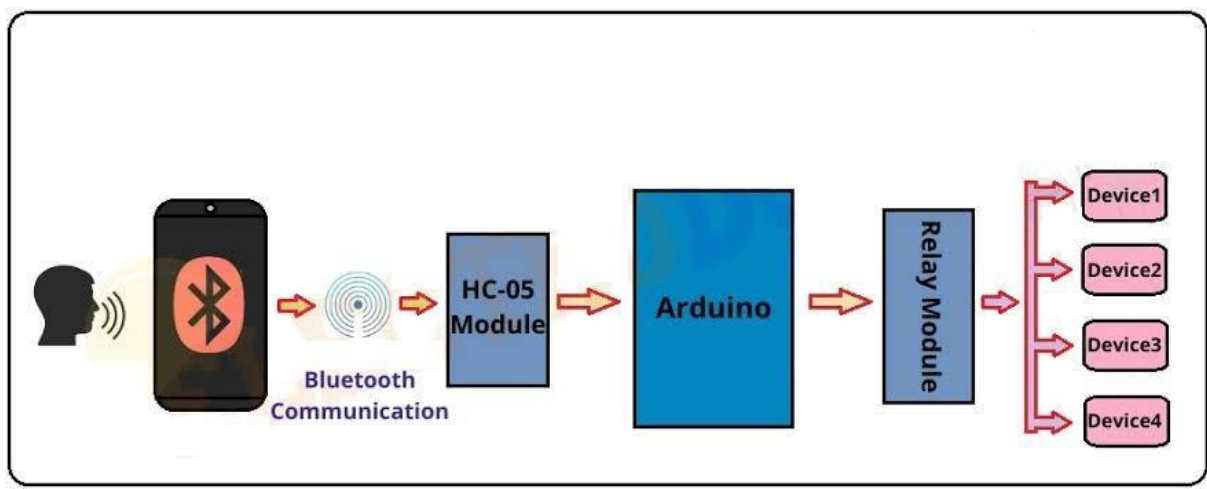
## CHAPTER 4

### PROPOSED SYSTEM

The core component of this system is the Arduino Uno which has a microcontroller i.e Atmega 328. The Atmega has a 32KB flash, it needed to burn a boot loader and download Arduino sketches. The boot loader is programmed under ISP program controller. The voice recognizer which is an inbuilt feature of android phones is used to build an application which the user can operate the appliances at his house. For wireless communication system a HC-05 Bluetooth Module is connected to the control unit for sensing the signals sent by the android voice application. The microcontroller device with the Bluetooth module and relay circuit needs to be connected to the board. Then we need to launch the android-based application on our smart phone. Through the application we can instruct the microcontroller to switch ON/OFF an appliance. After getting the instruction through the Bluetooth module, the microcontroller gives the signal to the relay board.

The application first searches for the Bluetooth device. If it is available then it launches the voice recognizer. It reads the voice and converts the audio signal into string. It provides a value for each appliance which will be fed to the microcontroller device. The microcontroller uses the port in the serial mode. After reading the data it decodes the input value and sends a signal to the parallel port through which the relay circuit will be activated.

#### 4.1 FRAMEWORK OF VOICE ACTIVATED SMART HOME APPLIANCES SYSTEM



**Figure 4.1:** Framework of Voice activated smart home appliance system

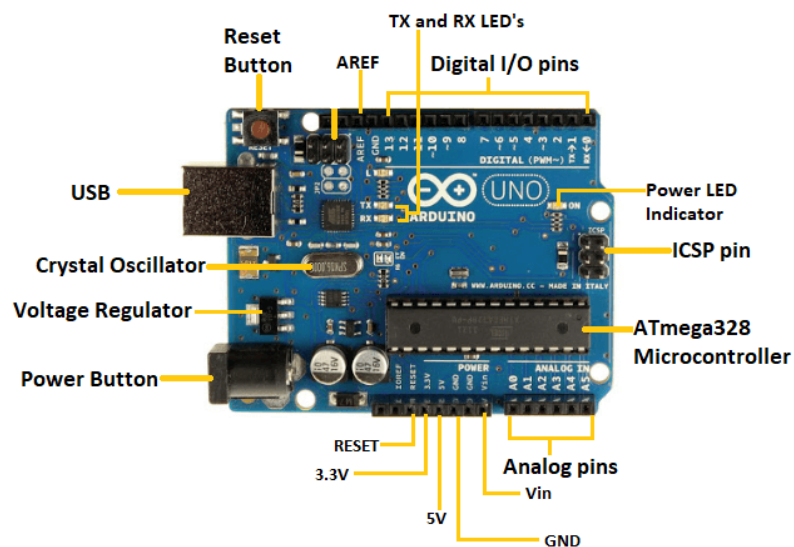
# CHAPTER 5

## HARDWARE REQUIREMENTS AND EXPLANATION

### 5.1 ARDUINO UNO

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc and initially released in 2010. The board is equipped with sets of digital and analog input/output pins that may be interfaced to various expansion boards and other circuit.

#### 5.1.1 ARDUINO UNO PIN DIAGRAM



*Figure 5.1.1: Arduino UNO Pin Diagram*

#### 5.1.2 ADVANTAGES

- Not much knowledge required to get started.
- Fairly low cost, depending on shields you need.
- Lots of sketches and shields available.
- No external programmer or power supply needed.

**TABLE 5.1.3: PIN CONFIGURATION OF ARDUINO UNO**

Pin Category	Pin Name	Details
Power	Vin, 3.3V, 5V, GND	Vin: input voltage to Arduino when external power source. 5V: regulated power supply used to power microcontroller and other components on the board. 3.3V: 3.3V supply generated by on board voltage regulator. Maximum current draw is 50mA. GND: ground pins.
Reset	Reset	Resets the microcontroller.
Analog pins	A0 – A5	Used to provide analog input in the range of 0-5V.
Input/output pins	Digital pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External interrupts	2,3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.
Inbuild LED	13	To turn on the inbuild LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

## 5.2 HC-05 BLUETOOTH MODULE:

The HC-05 Bluetooth Module is used as UART serial converter module and can easily transfer the UART data through the wireless Bluetooth. The Bluetooth module has a frequency:2.4GHz ISM band, PIO control and comes with an integrated antenna and edge connector. The Bluetooth module has four major components: antenna/RF component, Bluetooth hardware and firmware (baseband and link controller) and Bluetooth software protocol stack. HC-05 has red LED which indicates connection status, whether the Bluetooth is connected or not. Before connecting to HC-05 module this red LED blinks continuously in a periodic manner. When it gets connected to any other Bluetooth device, its blinking slows down to two seconds.



### 5.2.1 BLUETOOTH MODULE PIN DIAGRAM



*Figure 5.2.1: Bluetooth Module Pin Diagram*

### 5.2.2 ADVANTAGES

- It avoids interference from other devices.
- It has lower power consumption.
- It has range better than infrared.
- HC 05 Bluetooth module are available at very cheap cost

**TABLE 5.2.3: PIN CONFIGURATION OF BLUETOOTH MODULE**

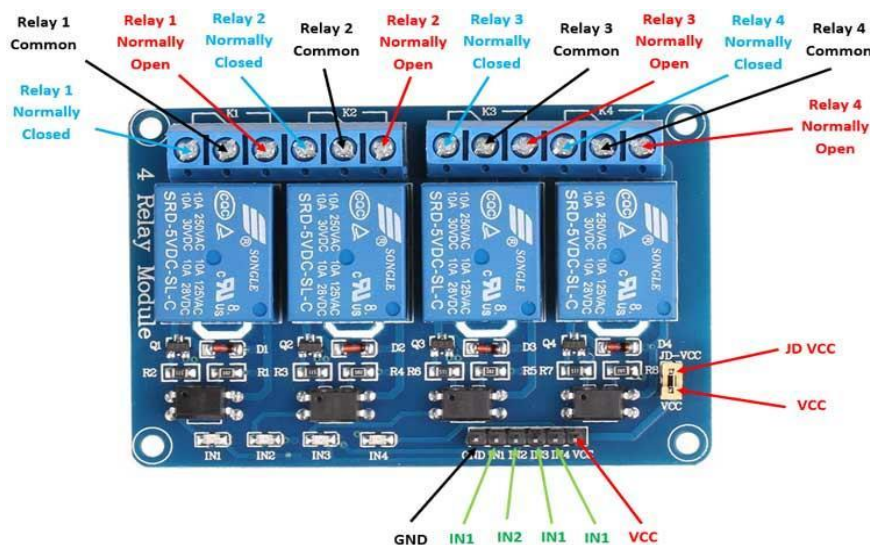
Pin Number	Pin Name	Description
1	Enable/key	This pin is used to toggle between data mode (set low) and AT command mode (set high). by default, it is in data mode.
2	Vcc	Powers the module connect to +5V supply voltage.
3	GND	Ground pin of module, connect to system ground.
4	TX – Transmitter	Transmits serial data. Everything received via Bluetooth will be given out by this pin as serial data.
5	RX – Receiver	Receive serial data. Every serial data given to this pin will be broadcasted via Bluetooth.
6	State	The state pin is connected to on board LED, it can be used as feedback to check if Bluetooth is working properly.
7	LED	Indicates the status of module <ul style="list-style-type: none"><li>● Blink once in 2 sec: module has entered command mode.</li><li>● Repeated blinking: waiting for connection in data mode</li><li>● Blink twice in 1 sec: connection successful in data mode</li></ul>
8	Button	Used to control the key/enable pin to toggle between data and command mode

## 5.3 FOUR CHANNEL RELAY MODULE

The 4 channel Relay Module is a convenient board which can be used to control high voltage, high current load such as motor, solenoid valves, lamps and AC load. It is designed to interface with microcontroller such as Arduino. The relay terminals (COM, NO and NC) is being brought out with screw terminal.

The NO (Normally Open) contacts connect the circuit when relay is activated; the circuit is disconnected when the relay is inactive. Normally closed (NC) contacts disconnected the circuit when the relay is activated; the circuit is connected when the relay is in active. The relay module has 4 components present on a four-channel relay module 5V relay, terminal block, male headers, transistors, optocouplers, diodes, and LEDs.

### 5.3.1 4 CHANNEL RELAY MODULE PIN DIAGRAM



*Figure 5.3.1: 4 Channel Relay Module Pin Diagram*

### 5.3.2 ADVANTAGES

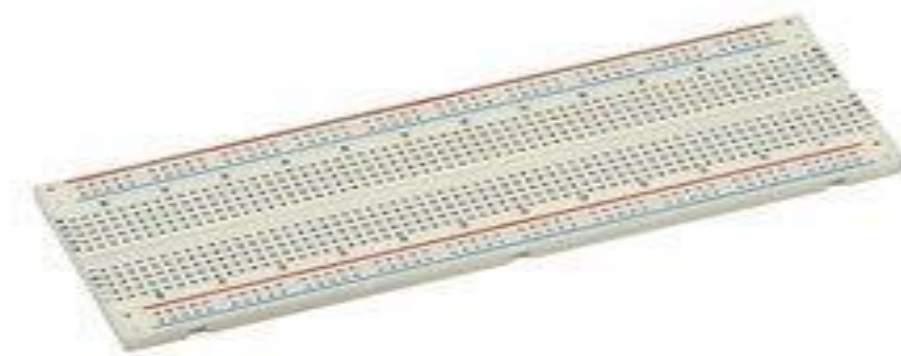
- Relays can switch AC and DC.
- Relays can switch high voltage
- Relay are a better choice for switching large currents (more than 5A).
- Relays can switch many contacts at once

**TABLE 5.3.3: PIN CONFIGURATION OF 4 CHANNEL RELAY MODULE**

Pin Number	Pin Name	Description
1	GND	Ground reference for the module
2	IN1	Input to activated relay 1
3	IN2	Input to activated relay 2
4	IN3	Input to activated relay 3
5	IN4	Input to activated relay 4
6	Vcc	Power supply for the relay module
7	Vcc	Power supply for selection jumper
8	JD-Vcc	Alternate power pin for relay module

## 5.4 BREADBOARD

A breadboard (sometimes called a plug block) is used for building temporary circuits. It is useful to designers because it allows components to be removed and replaced easily. It is useful to the person who wants to build a circuit to demonstrate its action, then to reuse the components in another circuit.



*Figure 5.4: Bread Board Diagram*

## 5.5 DC FAN

Dc fans are fans that are powered by the direct current. Like AC Axial fans, there are DC axial fans and DC cross flow fans too. DC axial fans are widely used in the air-vent exhaust, electronic device cooling, computer ventilation, etc.



**Figure 5.5: DC Fan Diagram**

## 5.6 LED BULB

LED stands for light emitting diode is a semiconductor device, which can emit light when an electric current passes through it. To do this, holes from p-type semiconductor recombine with electrons from n-type semiconductors to produce light.



**Figure 5.6: LED Diagram**

## 5.7 BUZZER

Buzzer can be categorized into two different types Active and Passive buzzer. An active buzzer has built-in oscillator so it can produce sound with only a DC power supply. A passive buzzer does not have a built-in oscillator, so it needs an AC audio signal to produce sound.



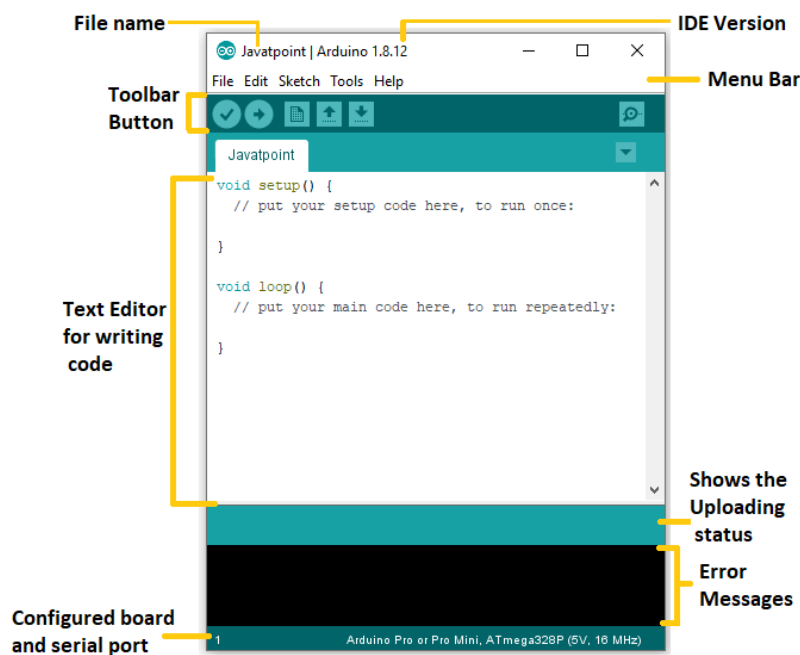
**Figure 5.7:** Buzzer Diagram

## CHAPTER 6

### SOFTWARE REQUIREMENTS AND EXPANATION

#### 6.1 ARDUINO IDE:







The Arduino Integrated Development Environment or Arduino software (IDE) contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.



*Figure 6.1: Arduino IDE Software*

#### 6.1.1 WRITING SKETCH

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. `.ino`. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

-  - Verify Checks your code for errors compiling it.
-  - Upload Compiles your code and uploads it to the configured board.
-  - New Creates a new sketch.
-  - Open Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.
-  - Save Saves your sketch.
-  - Serial Monitor Opens the serial monitor.

Additional commands are found within the five menus: **File**, **Edit**, **Sketch**, **Tools**, **Help**. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

### 6.1.2 SKETCHBOOK

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the **File > Sketchbook** menu or from the **Open** button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the **Preferences** dialog.

### 6.1.3 TABS, MULTIPLE FILES AND COMPIATION

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Before compiling the sketch, all the normal Arduino code files of the sketch (.ino, .pde) are concatenated into a single file following the order the tabs are shown in. The other file types are left as is.

### 6.1.4 UPLOADING

Before uploading your sketch, you need to select the correct items from the **Tools > Board** and **Tools > Port** menus. The boards are described below. On the Mac, the serial port is probably something like **/dev/tty.usbmodem241** (for an UNO or Mega2560 or Leonardo) or **/dev/tty.usbserial-1B1** (for a Duemilanove or earlier USB board), or **/dev/tty.USA19QW1b1P1.1** (for a serial board connected with a Keyspan USB-to-Serial

adapter). On Windows, it's probably **COM1** or **COM2** (for a serial board) or **COM4**, **COM5**, **COM7**, or higher (for a USB board) - to find out, you look for USB serial device in the port section of the Windows Device Manager. On Linux, it should be **/dev/ttyACMx** , **/dev/ttyUSBx** or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the **Upload** item from the **Sketch** menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino **bootloader**, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

### 6.1.5 LIBRARIES

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the **Sketch > Import Library** menu. This will insert one or more **#include** statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its **#include** statements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch.

### 6.1.6 THIRD-PARTY HARDWARE

Support for third-party hardware can be added to the **hardware** directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create



the **hardware** directory, then unzip the third-party platform into its own sub-directory. (Don't use "Arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

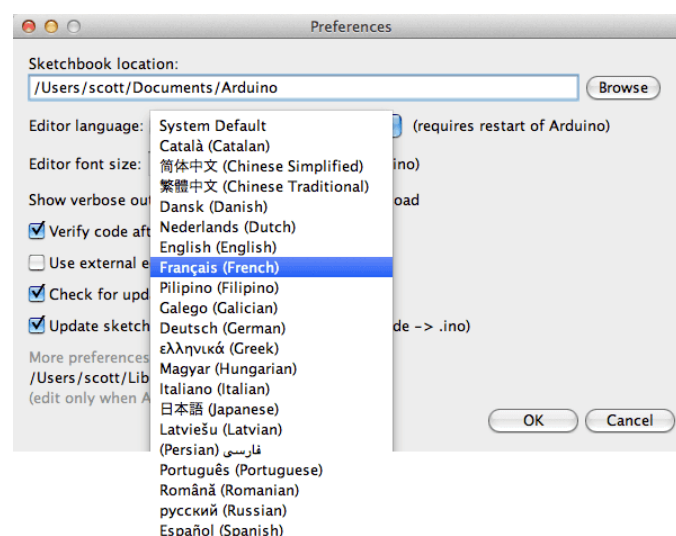
### 6.1.7 SERIAL MONITOR

This displays serial sent from the Arduino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to **Serial.begin** in your sketch. Note that on Windows, Mac or Linux the board will reset (it will rerun your sketch) when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board.

### 6.1.8 PREFERENCES

Some preferences can be set in the preferences dialog (found under the **Arduino** menu on the Mac, or **File** on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.

### 6.1.9 LANGUAGE SUPPORT



*Figure 6.1.9: Supported Language in Arduino IDE*

Since version 1.0.1, the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

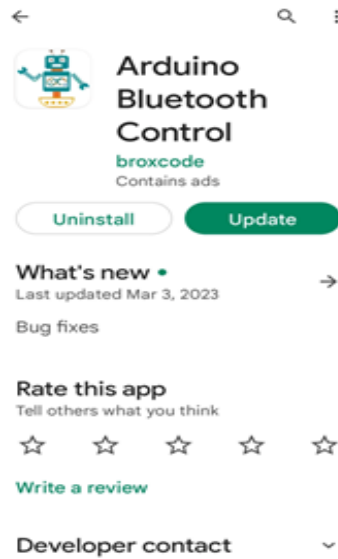
If you would like to change the language manually, start the Arduino Software (IDE) and open the **Preferences** window. Next to the **Editor Language** there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting **System Default** from the **Editor Language** drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

## 6.2 ARDUINO BLUETOOTH CONTROL

Arduino Bluetooth Control is an application that allows you to control your Arduino board (and similar boards) via Bluetooth, and so to create awesome and fully customized projects. The settings section allows you to adapt the application to your needs, through a very simple and intuitive interface.

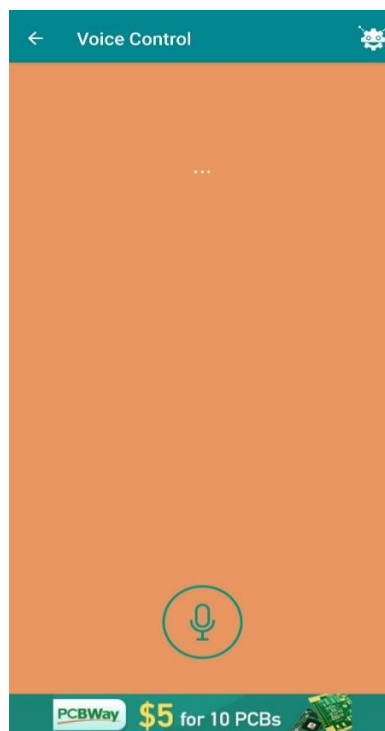
The application also smartly remembers your Bluetooth module and tries to connect automatically to the latest one you have used, so you won't have to select it every time you use it.



*Figure 6.2: Arduino Bluetooth Control Software*

### 6.2.1 VOICE CONTROL TOOL

Have you ever dreamed of talking to you robot? well now your dream is becoming true! With the Arduino Bluetooth control, you can customize your own vocal commands and use them to control all your microcontroller-based boards.



*Figure 6.2.1: Voice Control Tool*

### **6.2.2 ARDUINO BLUETOOTH CONTROLLER ADVANTAGES**

1. Community, which willingly shares their knowledge and complete projects on the web, so it is easy for us to start.
2. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board – you can simply use a USB cable.
3. Additionally, Arduino IDE uses a simplified version of C++, making it easier to learn how to program.
4. Many programmers give up their interests in microcontrollers due to fear of electronics. At the beginning, electronics is not a big deal. The Arduino community likes to share their projects on the web. They also share simplified electronic schematics like in the picture below. To find schematics like this, you only need to tap in the name of the sensor or module plus Arduino and then go to Google Graphics, and the proper schema should be there. If you would like to develop more complicated devices and create your own printed circuits (PCB), then you should dig into electronics.

### **6.2.3 ARDUINO CODE SYNTAX**

This is a good occasion to briefly explain Arduino's code syntax.

As you can see, we have two main methods here. First `setup()`: in this method, we initialize serial transmission at baud rate 9600. Baud rate is the maximum number of chars that we can send in one second. The higher the baud rate, the more sensitive the transmission becomes to noise. We set it to that specific number because our HC-06 Bluetooth module is configured to that number by default.

The only thing that is important from the start is that we should set the same baud rate on Arduino and on the Bluetooth module's side.

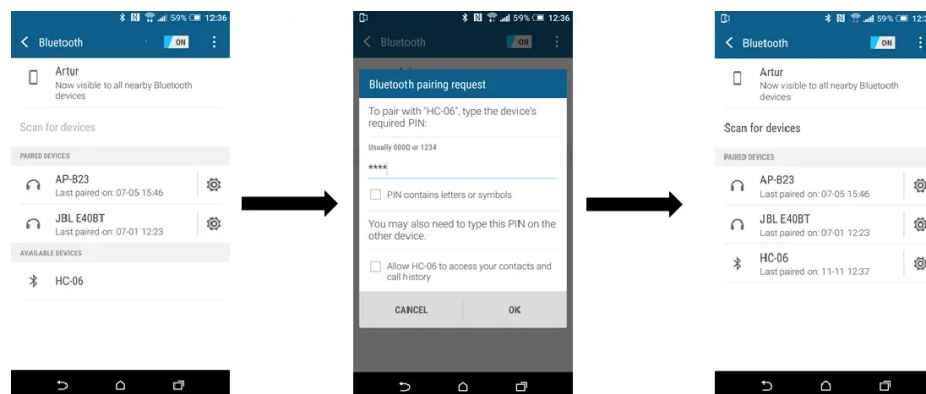
The second method, `loop()`, is an infinite loop of a program. So here we use the serial transmission to send text Software Hut and add a one second delay. And that's it! Nothing more. For me personally, it's like magic. You can achieve such a great thing like the wireless transmission with almost no effort.

## 6.2.4 PAIRING DEVICES

To establish Bluetooth communication between two devices, we need to pair them. Device pairing is needed only once. The process looks like this:

One device must enable Bluetooth discoverability (HC-06 enables discoverability implicitly when it powers up). So, this device will be visible to its surroundings.

Afterwards, we choose the option to scan for other devices on the second device. When we detect the Bluetooth module, we can send a pairing request to this module. Next, we get a pop-up dialog window on our smartphone, and here we must type the password. Naturally, its 1234, and on the HC-06, we can change it via AT commands. Then, if the password is correct, we will see that our Bluetooth device is in the paired devices section.



*Figure 6.2.4: Pairing Bluetooth Devices*

## 6.2.5 ANDROID PART

Now we smoothly move on to the Android part. Establishing a connection is based on the client-server model. The server creates a server socket which waits for incoming connection requests, then the client creates a client socket which sends a connection request to the server.

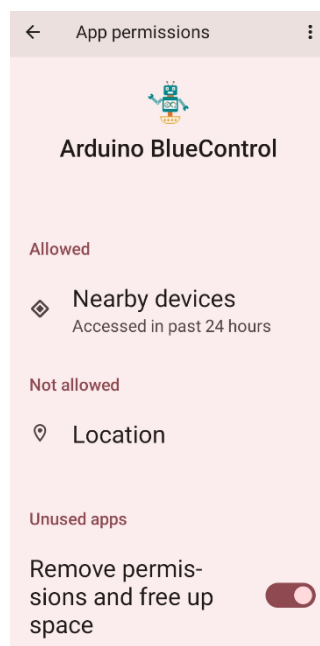
The server accepts this request and generates a second socket which is used to communicate with this client. Now it is possible for us to send and receive data both ways.



**Figure 6.2.5:** *Android Part*

### 6.2.5.1 SET PERMISSIONS

Let's begin with an example of the Android code. First, we should set the permissions in the Android Manifest file. Bluetooth Admin and Bluetooth are needed.

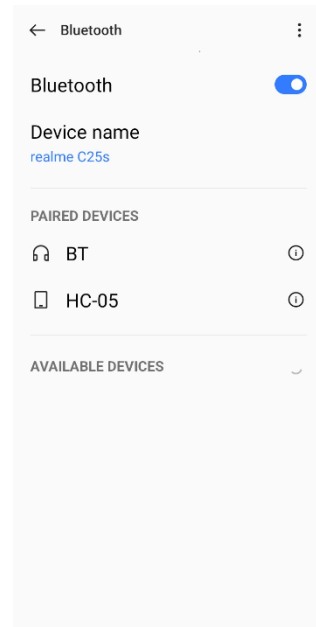


**Figure 6.2.5.1:** *Allowing Permission*

### 6.2.5.2 CHECK BLUETOOTH'S AVAILABILITY

Then, we begin checking Bluetooth's state, and to do so, we use the Bluetooth Adapter class. It represents the phone's built-in Bluetooth module. We will use this class to check if our device even has Bluetooth to begin with.

If not, sorry, we can't do much here, so we just can finish our Activity. But we rather expect that we have Bluetooth onboard, so then we can check if this Bluetooth is enabled. If not, we can call up the system's dialog window asking the user if they would like to turn Bluetooth on.



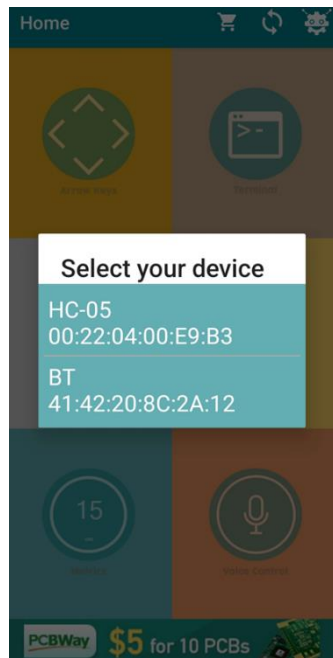
*Figure 6.2.5.1: Checking Bluetooth Availability*

### 6.2.5.3 DISPLAY THE LIST OF PAIRED DEVICES

All right, we have checked Bluetooth's availability, we have it turned on, so now we can display the list of paired devices. For this purpose, we use the `getBondedDevices` method. Then you can just add each bonded device to the list and display the most useful data, like name and MAC address.

If we don't have any paired devices, then we display a short message to inform the user.

When the user clicks an item on the list, we then acquire a MAC address from this device. Then we send it via intent to the next screen and start our second Activity. If you would like to connect with only one device, then you can skip the part about getting the list of bonded devices. The main purpose was to get the MAC address identifying our Bluetooth device.



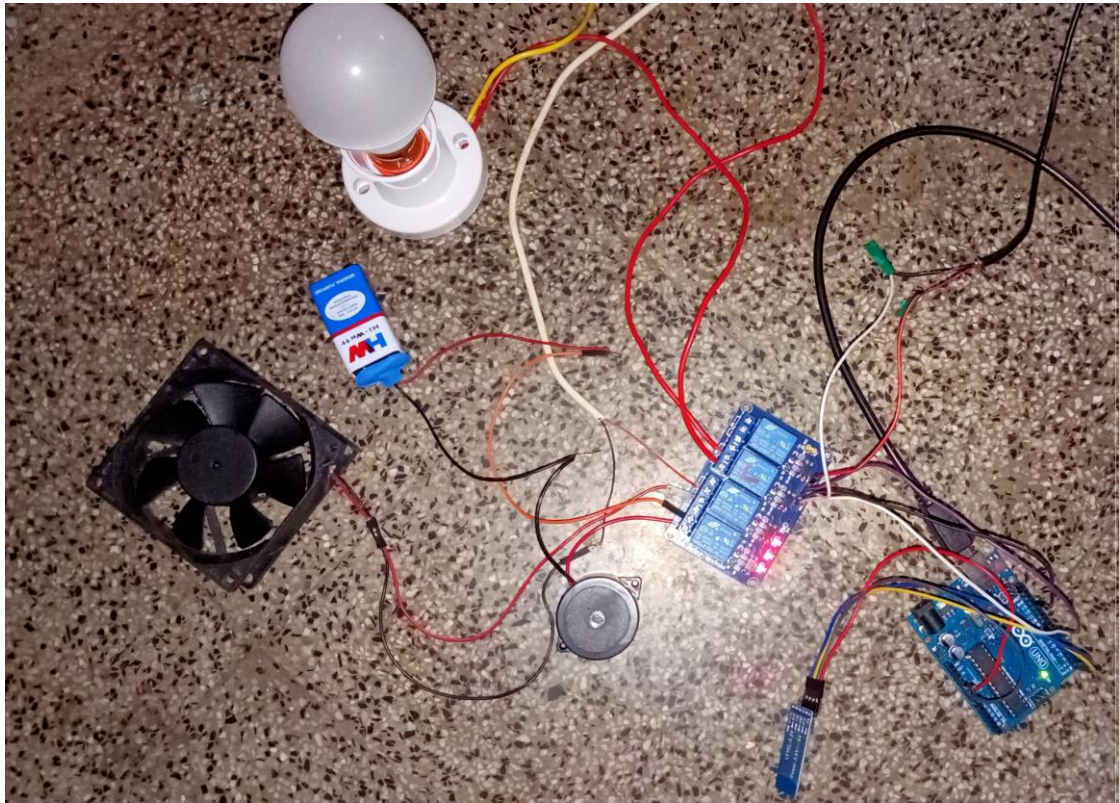
***Figure 6.2.5.3: Paired Device***



## CHAPTER 7

### RESULT

#### 7.1 OVERALL CIRCUIT



*Figure 7.1: Overall circuit*

#### 7.2 FAN ON AND OFF



*Figure 7.2: Fan ON*



*Figure 7.2: Fan OFF*

### 7.3 LIGHT ON AND OFF



*Figure 7.3: Light ON*



*Figure 7.3: Light OFF*

### 7.4 BUZZER ON AND OFF



*Figure 7.4: Buzzer ON*



*Figure 7.4: Buzzer OFF*

## **CHAPTER 8**

### **CONCLUSION AND FUTURE ENHANCEMENT**

#### **8.1 CONCLUSION**

The implementation of a system for monitoring environmental parameters using the IoT has been tentatively tested to verify weather parameters. The system provides low power consumption solution for the establishment of a station weather system. The system tested in an indoor environment and it successfully updated the environment and weather conditions from sensor data. This information will be useful for future review and tend to be shared effectively with various users. this model can be extended to the observation of contamination in new and modern urban areas. To protect the general well-being from contamination, this model provides an effective and minimal effort response for continuous observation.

#### **8.2 FUTURE ENHANCEMENT**

The future of voice activated smart home appliance and connected devices will be increasingly based on ambient sensors. This means that instead of controlling household appliances and other devices with a series of manual control pads, users could simply use their smartphones to control them with no need for wires or cables.

## **CHAPTER 9**

### **BIBLIOGRAPHY**

1. <http://www.geeksforgeeks.org/what-is-arduino/>
2. <http://www.atmel.com>
3. <http://www.beyondlogic.org>
4. <http://www.elementzonline.com>
5. <http://www.elementztechblog.wordpress.com>