CHAPTER 1

INTRODUCTION

Homes of the 21st century will become more and more self-controlled and automated due to the comfort it provides, especially when employed in a private home. A home automation system is a means that allow users to control electric appliances of varying kind. Many existing, well-established home automation systems are based on wired communication. This does not pose a problem until the system is planned well in advance and installed during the physical construction of the building. But for already existing buildings the implementation cost goes very high.

In contrast, Wireless systems can be of great help for automation systems. With the advancement of wireless technologies such as Wi-Fi, cloud networks in the recent past, wireless systems are used every day and everywhere.

In recent years, wireless systems like Wi-Fi have become more and more common in-home networking. Also, in home and building automation systems, the use of wireless technologies gives several advantages that could not be achieved using a wired network only.

1) Reduced installation costs: First and foremost, installation costs are significantly reduced since no cabling is necessary. Wired solutions require cabling, where material as well as the professional laying of cables (e.g., into walls) is expensive.

2) System scalability and easy extension: Deploying a wireless network is especially advantageous when, due to new or changed requirements, extension of the network is necessary. In contrast to wired installations, in which cabling extension is tedious. This makes wireless installations a seminal investment.

3) Aesthetical benefits: Apart from covering a larger area, this attribute helps to full aesthetical requirements as well. Examples include representative buildings with all-glass architecture and historical buildings where design or conservatory reasons do not allow laying of cables.

4) Integration of mobile devices: With wireless networks, associating mobile devices such as PDAs and Smartphones with the automation system becomes possible everywhere and at any time, as a device's exact physical location is no longer crucial for a connection (as long as the device is in reach of the network). For all these reasons, wireless technology is not only an attractive choice in renovation and refurbishment, but also for new installations.

Unfortunately, smart home technology is very few adopted in India. Most are adopted in apartments, residential, and luxury homes. This is because it is very expensive to adopt smart home i.e., for complex telecommunications and control systems and some equipment that are not available in the market. This paper offers a small smart home system to address problems above. The small smart home systems is designed with the Arduino microcontroller-based with WLAN systems, which is able to monitor and control lights, room temperature, alarms for detecting suspicious movements, and other household appliances. After briefly describing the smart home, the next section of this paper is to discuss design process and parts of the system designed. Several experiments were conducted to test the performance of the system. Finally, this paper will be closed with the conclusion

An application is anything that has a connection to the internet and wants to interact with data from devices and/or control the behavior pf those devices in some manner. Applications identify themselves to the IoT Foundation with an API key and a unique application id. Applications do not need to be registered before they can connect to the IoT Foundation.

- Health
- Home and Building
- Transportation
- Logistics
- Smart Industry
- Smart Retail
- Smart Environment

The foremost aim of technology has been to increase efficiency and decrease effort. With the advent of „Internet of Things" in the last decade, we have been pushing for ubiquitous

computing in all spheres of life. It thus is of extreme importance to simplify human interfacing with technology. Automation is one such area that aims that achieves simplicity whilst increasing efficiency. Voice controlled House Automation System aims to further the cause of automation so as to achieve the goal of simplicity.

1.1 PROBLEM STATEMENT

Home automation systems face four main challenges, these are high cost of ownership, inflexibility, poor manageability, and difficulty in achieving security. The main objectives of this research are to design and implement a home automation system using Arduino that is capable of controlling and automating most of the house appliances. and monitor temperature, humidity, lighting, fire and burglar alarms, gas density from the house and has an infrared sensor to guarantee the security of the family. This system also has an internet connection to monitor and control the home appliances from anywhere in the world. This will decrease the deployment cost and will increase the ability of upgrading, and system reconfiguration

1.2 EXISTING SYSTEM:

- The Existing system based on with the GSM Module only. switches, switch boards which are connected through wires to each electronic device

Disadvantages of Existing system

- Sometimes forget turn off the switches.
- Handicapped people's difficulty turn on/off switches.

1.3 PROPOSED SYSTEM:

Home automation system using Arduino means monitoring and controlling of home appliances remotely using the concept of internet of things (IOT). In this method we use

mobiles or computers to control the basic home appliance and make it function through the designed internet connection/local area network (LAN) servers. This type of home is also known as smart home.

The temperature sensor read temperature values and the smoke sensor detects the smoke to send SMS alarm and to ring the buzzer figure shown in 4.13. The automatic switching on and off of the light is controlled by to incorporate security in this design, a motion detector is integrated using Passive Infrared Sensor (PIR) to detect movement figure shown in 4.17.

The temperature and the motion detection is stored in cloud for analysis. If the temperature exceeds the threshold level, then the cooler will turn on automatically and it will off when the temperature comes to control figure shown in 4.12. Similarly, when there is a leakage of gas in the house alarm is raised giving the alert sound. The required lights are turned on/off automatically by detecting the light outside the house figure shown in 4.20. The user can also monitor the electric appliances through the internet via web server. If the lights or any electrical appliances are left on in hurry can be seen and turned off remotely through simply typing the IP address of the web server. To develop the Face ID controlled Digital Door lock system using ESP32-CAM figure shown in4.5 to 4.10**.** To use Blynk app and ESP8266 in order to control lamp (any other home appliances will be just fine), the combination will be through the internet. The purpose of this instructible is to show in 4.18 the simplest solution remotely-controlling your Arduino or compatible hardware ESP8266 over the Internet and to explore the world of Internet of Things (IoT).

CHAPTER 2

SYSTEM ANALYSIS

2.1 SYSTEM REQUIREMENTS SPECIFICATION

2.1.1 HARDWARE REQUIREMENTS:

- Arduino uno -2
- ESP8266 WIFI (Module and WIFI Board)
- PIR Sensor
- DTH11 Sensor
- MQ5 gas Sensor
- Electric Lock Solenoid Cabinet
- 16x2 LCD Displays – 2
- LM2596S DC-DC Buck Converter
- ESP32-CAM
- 4-channel Relay Module
- Android Phone
- Buzzer
- Push Buttons
- Some LED Lights
- Battery s
- Breadboard
- Connecting Wires

2.1.2 SOFTWARE REQUIREMENTS:

- Arduino IDE
- Blynk app
- Esp32-cam web interface

2.2 Block Diagram

The block diagram of smart home automation system is shown in fig.2.1 below. All components are connected by two Arduino. This system can run with AC and DC as the whole system use hybrid power supply. The Arduino are used to obtain values of physical conditions through sensors connected to it. The temperature sensor read temperature values and the smoke sensor detects the smoke to ring the buzzer. The automatic switching on and off of the light is controlled by the Light Dependent Resistor (LDR) which determines the day light intensity. Also, to incorporate security in this design, a motion detector is integrated using Passive Infrared Sensor (PIR) to detect movement.

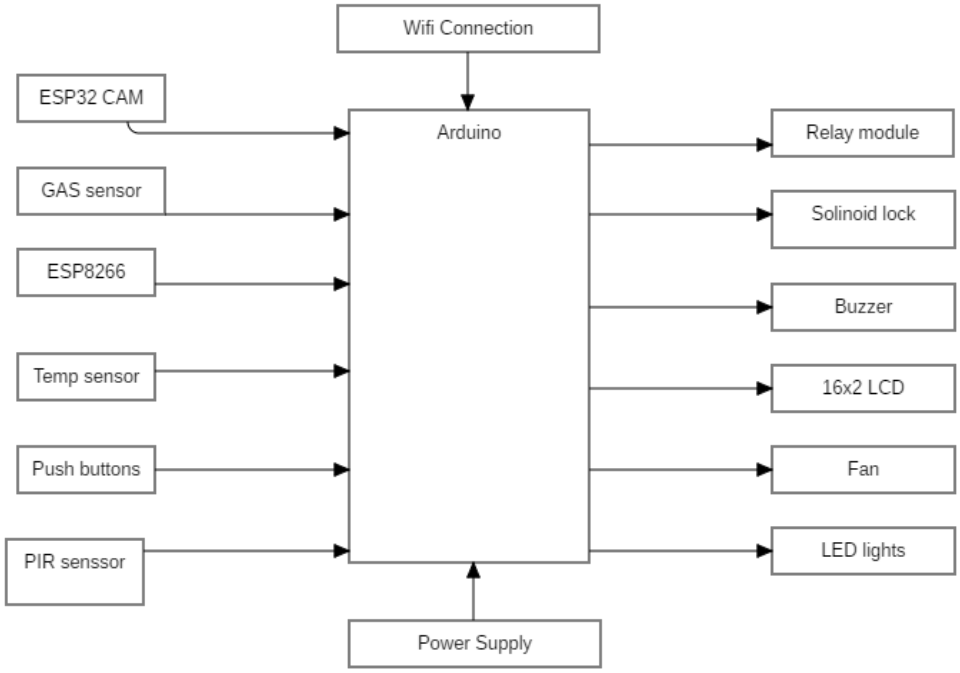

Fig 2.1 Block Diagram

2.3 USE CASE VIEW:

This kind of diagram basically explains the collaboration of the user with the system. We will discuss the system design and architecture phase with the help of Unified Modelling Language (UML). User interacts only system installation then after Smart door lock enroll the faces and recognize the faces to unlock or lock the door. The temperature sensor read temperature values and the smoke sensor detects the smoke to alarm and

to ring the buzzer. The automatic switching on and off of the light is controlled by the Light Dependent Resistor (LDR) which determines the day light intensity. Also, to incorporate security in this design, a motion detector is integrated using Passive Infrared Sensor (PIR) to detect movement. Wi-Fi module (ESP8266) is used to do such task. It allows internet access to the whole system by assigning an internet protocol (IP) address when connected to the network. When user checks the usage of electrical appliances, he has in the room via his mobile phones by implementing Blynk App, he also can decide to turn on or off the electrical appliances instantly. Through his mobile phones, he is now capable in observing the electrical status of his room remotely on condition that the internet is accessible during the practice. Instead of conventional method, the switching operation and equipment in the room can be done by touching on mobile phone's screen. Furthermore, risk in handling electrical appliances can be reduced compared to the usual method.

Use case diagram:

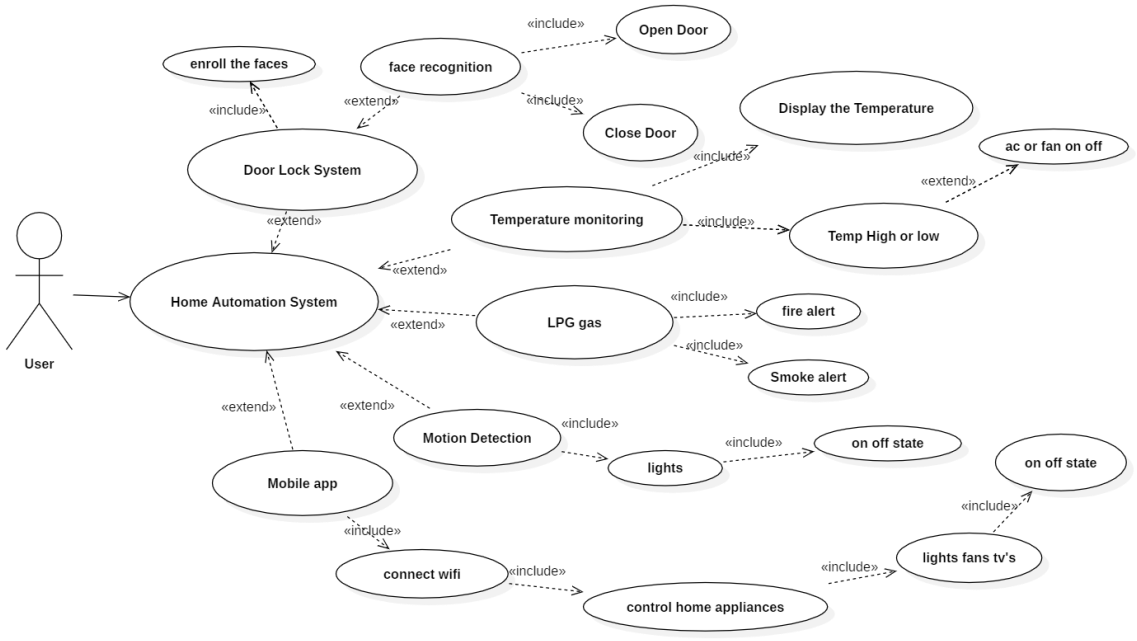

Fig 2.2 Use case diagram

2.4 SEQUENCE DIAGRAM:

This kind of diagram explains, the interaction between the system and the user by sequencing order how the relations take place between. It also turns up with the identification process of the code. In case if the code is not identified by Arduino or application, it follows the sequence as shown below in the diagram. The sequence of activities in the HAS. When the connection is established, it will start reading the parameters of sensors like pir, dth11, esp32cam, etc. The threshold levels for the required sensors are set as relays, solenoid lock, LED lights. The sensor data are sent to the web server and stored in the cloud. The data can be analyzed anywhere any time. If the sensor parameters are greater than the threshold level then the respective alarm. will be raised and the required actuation is done for the controlling of the parameters. At the door of the house a motion sensor is fixed to detect any movement near the door. Light 1 will turn on automatically when light sensor detects the darkness. A cooler/Fan will turn on when the room temperature exceeds the set threshold and in turn reduces the room temperature. The gas sensor MQ-5 is placed in the kitchen to detect any gas leakage, if any leakage is detected the alarm in the hall is raised. Relay is used to switch the electrical appliances like light, fan etc.

Sequence diagram:

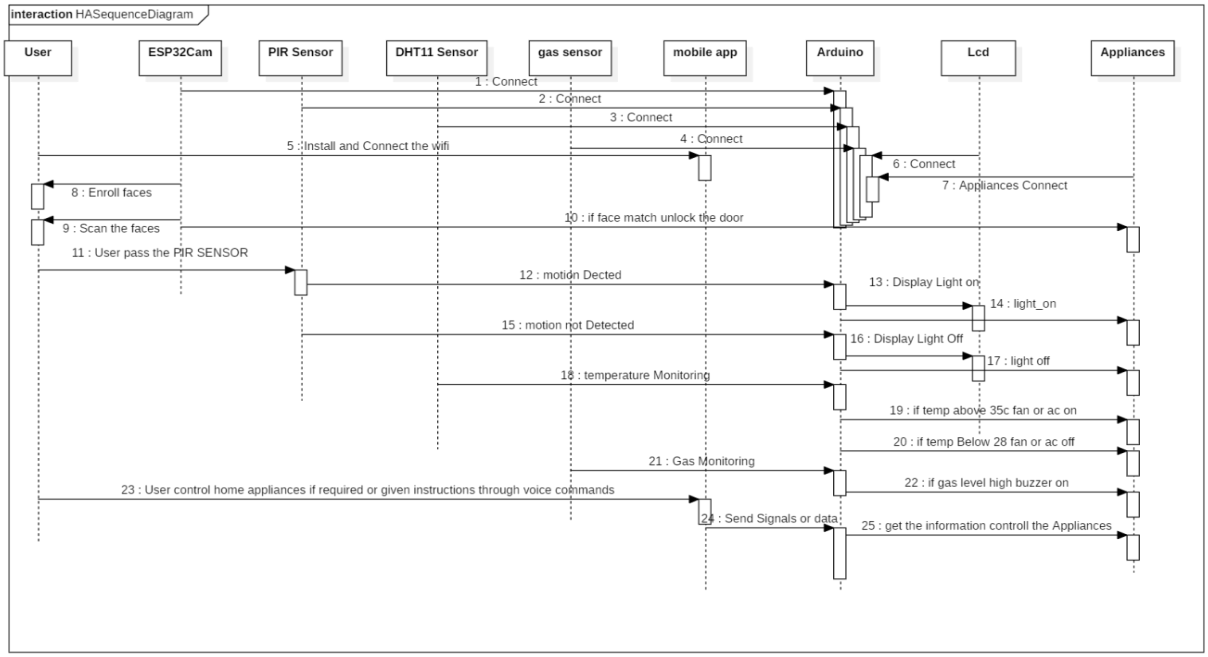

Fig 2.3 Sequence Diagram.

2.5 ACTIVITY DIAGRAM:

In Unified Modelling language, activity diagram shows the workflow with the steps of activities and actions from beginning to the end of the process in a system. According to this project, each step across is followed by the other one, so for that, every condition is satisfied in each step so that the next step is to peruse. Below activity diagram of this project illustrates all the steps that are involved and the decision that is followed corresponding to the output from each step.

Activity diagram:

Fig 2.4 Activity Diagram

CHAPTER 3

SYSTEM DESIGN

3.1 Class Diagram

Class diagrams are the main building block of any object-oriented solution. It shows the classes in a system, attributes, and operations of each class and the relationship between each class. The data can be analyzed anywhere any time. If the sensor parameters are greater than the threshold level then the respective alarm. will be raised and the required actuation is done for the controlling of the parameters. At the door of the house a motion sensor is fixed to detect any movement near the door. Light 1 will turn on automatically when light sensor detects the darkness. A cooler/Fan will turn on when the room temperature exceeds the set threshold and in turn reduces the room temperature. The gas sensor MQ-5 is placed in the kitchen to detect any gas leakage, if any leakage is detected the alarm in the hall is raised. Relay is used to switch the electrical appliances like light, fan etc.

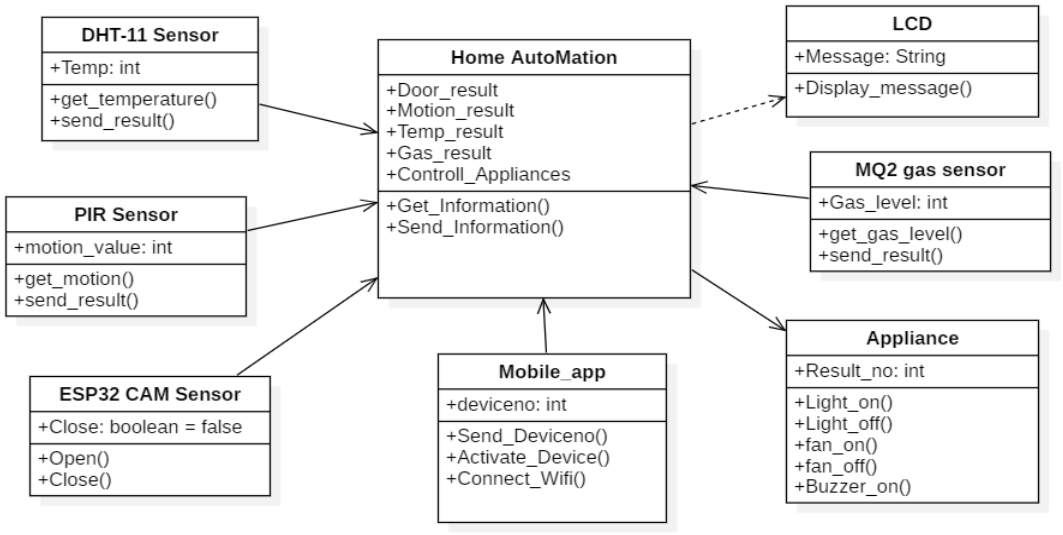

Fig 3.1 Class Diagram

3.2 Modularization Details

- Automatic door lock module
- Temperature module
- Motion detected module
- Smoke/gas detected module
- Controlled home appliances module

→**Automatic door lock module:** The ESP32 cam face recognition door lock to enroll multiple faces. if it detects any enrolled face, the door will unlock automatically.

→**Temperature module:** This module helps if temperature exceeds the threshold level, then the AC or fan will turn on automatically and it will off when temperature comes to control**.**

→**Motion detected module:** The required lights are turned on/off automatically by detecting the motion in the house.

→**Smoke/gas detected module**: if there is a leakage of gas in the house the alarm is raised giving the alert sound**.**

→**Controlled home appliances module:** This module helps to connect mobile app and Wi-Fi to control the Home appliances and also controlled the home appliances through voice commands.

3.3 Circuit Diagram

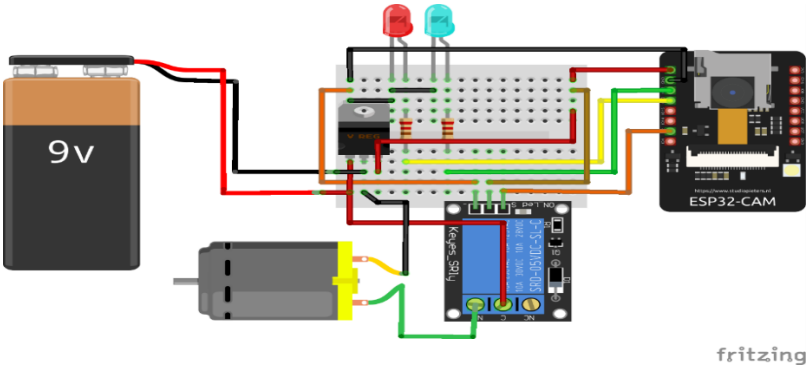

Fig 3.2 Door lock Circuit Diagram

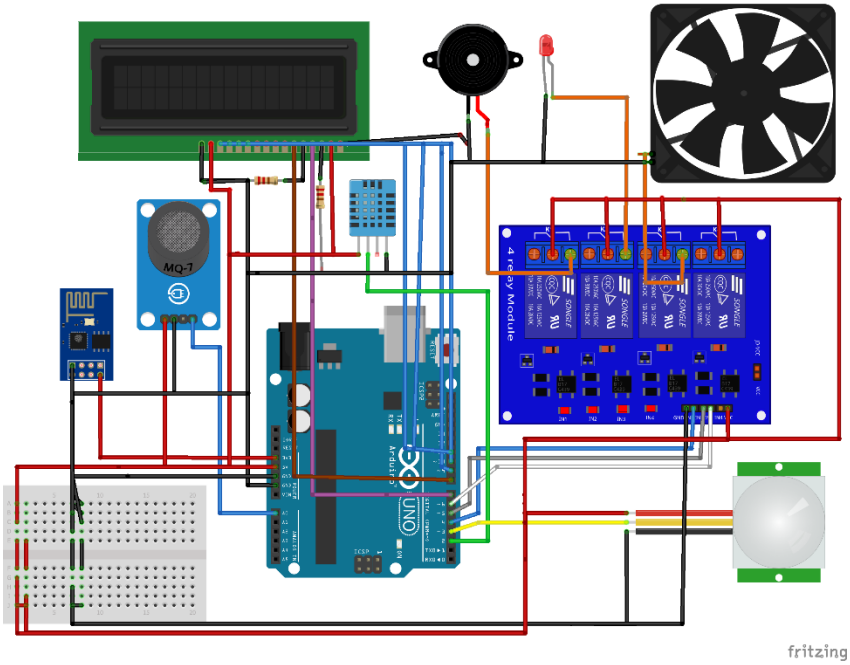

Fig 3.3 Home Automation circuit

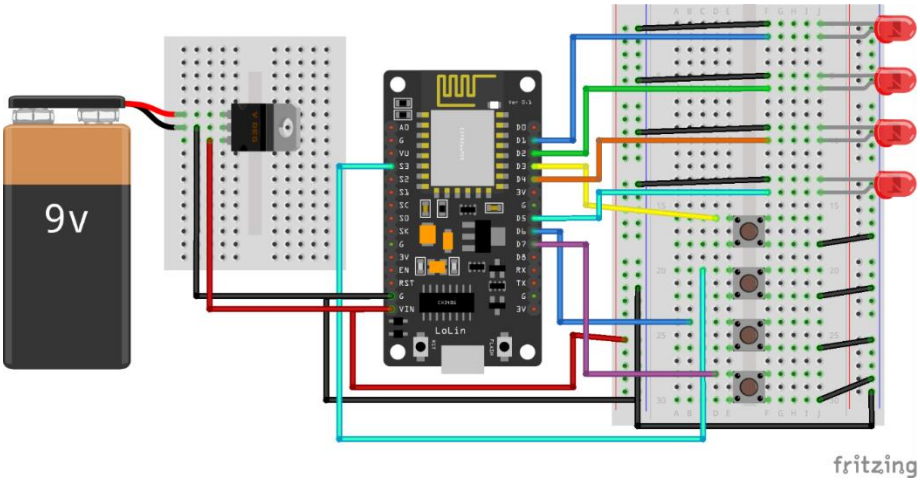

Fig 3.4 Control Home appliances circuit

3.4 COMPONENTS DETAILS

3.4.1 Hardware Modules

3.4.1.1 Arduino:

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The software, too, is open-source, and it is growing through the contributions of users worldwide.

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low-cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and

to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire.

- **Inexpensive** –Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50

- **Cross-platform** –The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

- **Simple, clear programming environment –**The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

- **Open source and extensible software** –The Arduino software is published as open-source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.

- **Open source and extensible hardware** –The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money

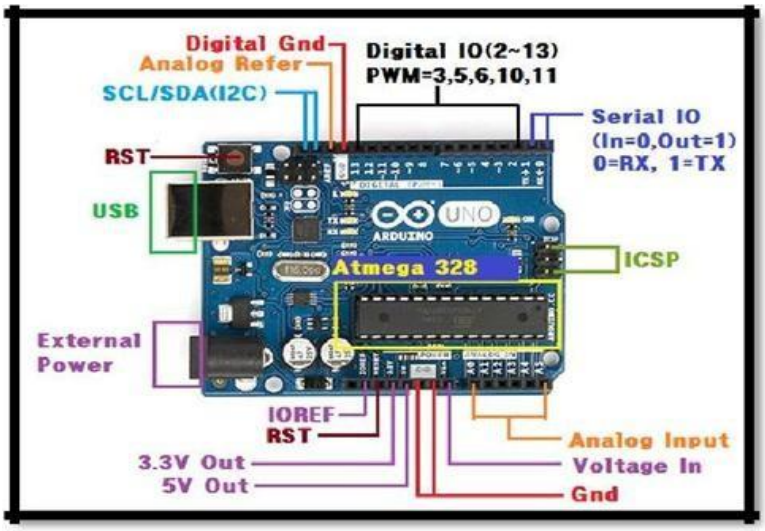

Fig 3.5 Arduino

3.4.1.2 DHT11:

DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal-acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness. Page | 3 Each DHT11 element is strictly calibrated in the laboratory that is extremely accurate on humidity calibration. The calibration coefficients are stored as programmes in the OTP memory, which are used by the sensor's internal signal detecting process. The single-wire serial interface makes system integration quick and easy. Its small size, low power consumption and up-to-20-meter signal transmission making it the best choice for various applications, including those most demanding ones. The component is 4-pin single row pin package.

It is convenient to connect and special packages can be provided according to users' request.

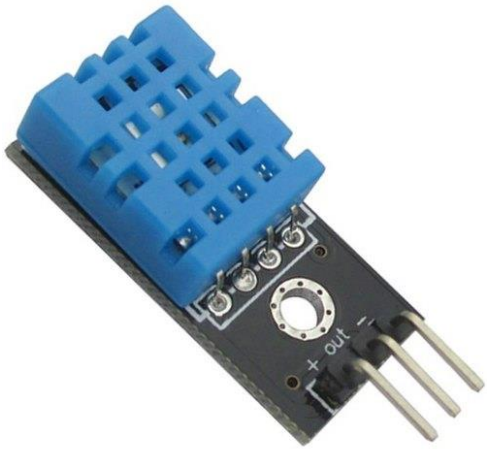

Fig 3.6 DHT11 Sensor

3.4.1.3 Esp8266:

The ESP**8266** development board comes with the ESP-12E module containing ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. ESP8266 has 128 KB RAM and 4MB of Flash memory to store data and programs. Its high processing power with in-built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IoT projects.

ESP8266 can be powered using Micro USB jack and VIN pin (External Supply Pin). It supports UART, SPI, and I2C interface.

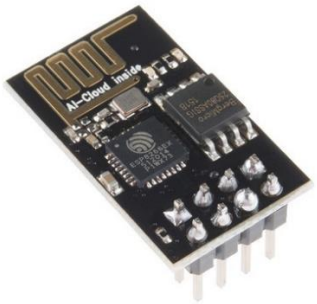

Fig 3.7 ESP8266(WIFI)

Once Arduino IDE is installed on the computer, connect the board with the computer using the USB cable. Now open the Arduino IDE and choose the correct board by selecting Tools>Boards>NodeMCU1.0 (ESP-12E Module), and choose the correct Port by selecting Tools>Port. To get it started with the Node MCU board and blink the built-in LED, load the example code by selecting Files>Examples>Basics>Blink. Once the example code is loaded into your IDE, click on the 'upload' button given on the top bar. Once the upload is finished, you should see the built-in LED of the board blinking.

3.4.1.4 PIR SENSOR:

PIR sensors are more complicated than many of the other sensors explained in these tutorials (like photocells, FSRs and tilt switches) because there are multiple variables that affect the sensors input and output. To begin explaining how a basic sensor works, we'll use this rather nice diagram

The PIR sensor itself has two slots in it, each slot is made of a special material that is sensitive to IR. The lens used here is not really doing much and so we see that the two slots can 'see' out past some distance (basically the sensitivity of the sensor). When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls or outdoors. When a warm body like a human or animal passes by, it first intercepts one half of the PIR sensor, which causes a *positive differential* change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a negative differential change. These change pulses are what is detected.

Fig 3.8 PIR Sensor

The IR sensor itself is housed in a hermetically sealed metal can to improve noise/temperature/humidity immunity. There is a window made of IR-transmissive material (typically coated silicon since that is very easy to come by) that protects the sensing element. Behind the window are the two balanced sensors.

3.4.1.5 Esp32cam

The ESP32-CAM board is a $7 device that combines an ESP32-S chip and an OV2640 camera. It allows you to set up a video streaming web server, build a surveillance camera to integrate with your home automation system, do face recognition and detection, and much more.

Besides the OV2640 camera and several GPIOs to connect peripherals, the ESP32-CAM also features a microSD card slot that can be useful to store images taken with the camera or to store files to serve to clients.

Fig 3.9 ESP32 Cam

3.4.1.6 MQ 5 gas Sensor

A Typical human nose has 400 types of scent receptors enabling us to smell about 1 trillion different odours. But still, many of us do not have the capacity to identify the

type or concentration of gas present in our atmosphere. This is where Sensors comes in, there are many types of sensors to measure different parameters and a Gas sensor is one which comes handy in applications where we have to detect the variation in the concentration of toxic gases in order to maintain the system safe and avoid/caution any unexpected threats. There are various gas sensors to detect gases like oxygen, Carbon Dioxide, Nitrogen, methane etc. They can also be commonly found in devices that are used to detect the leakage of the harmful gases, monitor the air quality in industries and offices etc.

In this article, we will learn more about gas sensors, their construction, types, working and how they can be used to measure the required type and concentration of Gas in our atmosphere. There are many types of Gas sensors but the MQ type gas sensors are commonly used and widely popular so will focus more on these types of sensors for this article.

A **gas sensor** is a device which detects the presence or concentration of gases in the atmosphere. Based on the concentration of the gas the sensor produces a corresponding potential difference by changing the resistance of the material inside the sensor, which can be measured as output voltage. Based on this voltage value the type and concentration of the gas can be estimated.

Fig 3.10 MQ5 Sensor

The type of gas the sensor could detect depends on the sensing material present inside the sensor. Normally these sensors are available as modules with comparators as shown above. These comparators can be set for a particular threshold value of gas

concentration. When the concentration of the gas exceeds this threshold, the digital pin goes high. The analog pin can be used to measure the concentration of the gas.
Gas sensors are typically classified into various types based on the type of the sensing element it is built with. Below is the classification of the various types of gas sensors based on the sensing element that are generally used in various applications:

- Metal Oxide based gas Sensor.
- Optical gas Sensor.
- Electrochemical gas Sensor.
- Capacitance-based gas Sensor.
- Calorimetric gas Sensor.
- Acoustic based gas Sensor.

These sensors are normally available as modules (shown right), these modules consist of the gas sensor and a comparator IC. Now let's see the pin description of the gas sensor module which we will generally use with an Arduino. The gas sensor module basically consists of 4 terminals

- **Vcc** – Power supply
- **GND** – Power supply
- **Digital output** – This pin gives an output either in logical high or logical low (0 or 1) that means it displays the presence of any toxic or combustible gases near the sensor.
- **Analog output** – This pin gives an output continuous in voltage which varies based on the concentration of gas that is applied to the gas sensor.

3.4.1.7 LCD

An LCD (Liquid Crystal Display) screen is an electronic display module and has a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. The 16 x 2 intelligent alphanumeric dot matrix display is capable

of displaying 224 different characters and symbols. This LCD has two registers, namely, Command and Data.

Command registers stores various commands given to the display. Data register stores data to be displayed. The process of controlling the display involves putting the data that form the image of what you want to display into the data registers, then putting instructions in the instruction register. In your Arduino project Liquid Crystal Library simplifies this for you so you don't need to know the low-level instructions. Contrast of the display can be adjusted by adjusting the potentiometer to be connected across VEE pin.

Fig 3.11 LCD

3.4.1.8 Buzzer

A **buzzer** or **beeper** is an audio signaling device,[1] which may be mechanical, electromechanical, or piezoelectric (*piezo* for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

Fig 3.12 Buzzer

3.4.1.9 Push buttons:

A push-button (also spelled pushbutton) or simply button is a simple switch mechanism to control some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, although many un-biased buttons (due to their physical nature) still require a spring to return to their un-pushed state. Terms for the "pushing" of a button include pressing, depressing, mashing, slapping, hitting, and punching.

Fig 3.13 push button

3.4.1.10 Solenoid lock:

12V Solenoid lock has a slug with a slanted cut and a good mounting bracket. It's basically an electronic lock, designed for a basic cabinet, safe or door. When 9-12VDC is applied, the slug pulls in so it doesn't stick out and the door can be opened. It does not use any power in this state. It is very easy to install for automatic door lock systems like electric door lock with the mounting board. This solenoid in particular is nice and strong.

Fig 3.14 Solenoid lock

3.4.1.11 Relay:

The relay module is an electrically operated switch that allows you to turn on or off a circuit using voltage and/or current much higher than a microcontroller could handle. There is no connection between the low voltage circuit operated by the microcontroller and the high-power circuit. The relay protects each circuit from each other. Each channel in the module has three connections named NC, COM, and NO. Depending on the input signal trigger mode, the jumper cap can be placed at high level effective mode which 'closes' the normally open (NO) switch at high level input and at low level effective mode which operates the same but at low level input. Specifications

- On-board EL817 photoelectric coupler with photoelectric isolating anti interference ability strong
- On-board 5V, 10A / 250VAC, 10A / 30VDC relays
- Relay long life can absorb 100000 times in a row
- Module can be directly and MCU I/O link, with the output signal indicator
- Module with diode current protection, short response time

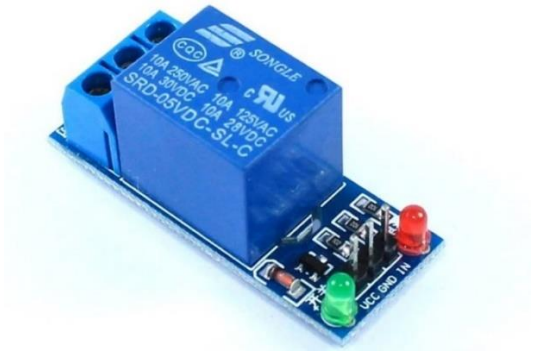

Fig 3.15 Relay

3.4.1.12 Fan:

The direct current fans, or DC fans, are powered with a potential of fixed value such as the voltage of a battery. Typical voltage values for DC fans are, 5V, 12V, 24V and 48V.

In contrast, the alternating current fans, or AC fans, are powered with a changing voltage of positive and of equal negative value. In general, this changing voltage has

sinusoidal shape. Worldwide, the usual value of this sinusoidal voltage may vary in size and in frequency, such as 100VAC, 120VAC, 200VAC, 220VAC, 230VAC or 240VAC, and with frequency (cycles per second) of 50Hz or 60Hz.

In the past, big AC fans were typically less expensive compared to big DC fans. Today however, their price difference is negligible due to their payback advantage. We will try to point out differences between the above fan types, to help you choose and purchase the correct fan type for your application.

Fig 3.16 Fan

DC technology has become much more sophisticated in recent years, and it can now be applied to both residential and industrial ceiling fans. DC fans have motors that rely on permanent magnets in order to attract and repel a rotor around the axis using electronic switching. DC technology is much newer than AC technology, which means there are fewer options available.

3.4.1.13 Power Supply:

A **power supply** is an electrical device that supplies electric power to an electrical load. The primary function of a power supply is to convert electric current from a source to the correct voltage, current, and frequency to power the load. As a result, power supplies are sometimes referred to as electric power converters. Some power supplies are separate standalone pieces of equipment, while others are built into the load appliances that they power. Examples of the latter include power supplies found in desktop computers and consumer electronics devices. Other functions that power supplies may perform include limiting the current drawn by the load to safe levels,

shutting off the current in the event of an electrical fault, power conditioning to prevent electronic noise or voltage surges on the input from reaching the load, power-factor correction, and storing energy so it can continue to power the load in the event of a temporary interruption in the source power (uninterruptible power supply).

3.17 DC Bug

3.4.1.14 LED lights:

A **light-emitting diode** (**LED**) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor. White light is obtained by using multiple semiconductors or a layer of light-emitting phosphor on the semiconductor device.

Fig 3.18 LED's

3.4.1.15 Connecting wires

Connecting wires allows an electrical current to travel from one point on a circuit to another, because electricity needs a medium through which to move. In the case of computers, wires are embedded into circuit boards, carrying pulses of electricity that are interpreted as binary signals of zeros and ones.

In a basic circuit, the wire comes from one terminal of a power source, such as a battery. It then connects to a switch that determines whether the circuit is open or closed. The wire then connects to the device that is drawing power, allowing it to draw electricity and perform its task. Finally, the wire connects the load back to the opposite terminal of the power source.

.

Fig 3.19 Wires

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 Arduino Software IDE:

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the GPL (General Public License), version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain , also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text.

file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware

Fig 4.1: Arduino IDE.

4.1.2 Open Arduino Tool

Open Arduino tool from start menu, display the new project Arduino window.

Fig 4.2: Open Arduino Window

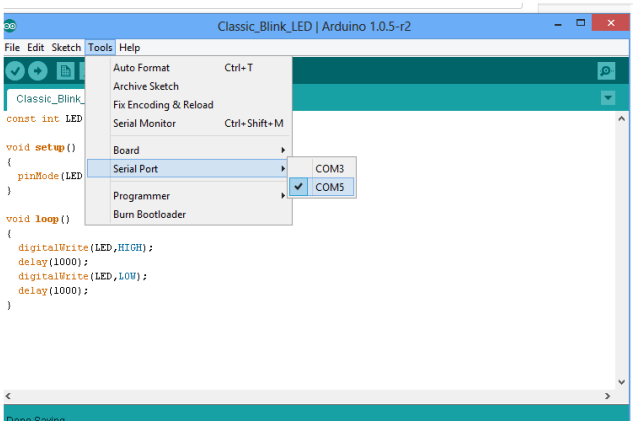

Fig 4.3: serial port selects

4.2 Writing Source Code

4.2.1 Code

```
//Home Automation using Arduino

#include "DHT.h"

#include <LiquidCrystal.h>

#include <SoftwareSerial.h>

#define DHTPIN 2
```

```
#define DHTTYPE DHT11
#define fan 3

LiquidCrystal lcd(12, 11, 7, 6, 5, 4);
DHT dht(DHTPIN, DHTTYPE);
String apiKey = "EN622Q7I9JWWJW86";    // Edit this API key according to your Account

String Host_Name = "DataMining";         // Edit Host_Name
String Password = "123456789";          // Edit Password

SoftwareSerial ser(9, 10);              // RX, TX
int i = 1;

int buzzer = 8;
int smokeA0 = A0;
int sensorThres = 200;
void setup()
{
  pinMode(fan, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(smokeA0, INPUT);
  lcd.begin(16, 2);
  dht.begin();
  Serial.begin(9600);                 // enable software serial
  ser.begin(115200);                    // reset ESP8266
  ser.println("AT+RST");               // Resetting ESP8266
dht.begin();                        // Enabling DHT11
char inv = "";
```

```
  //"AT+CWJAP"="ZTE-adde78","885dfbad"
  String cmd = "AT+CWJAP";
  cmd += "=";
  cmd += inv;
  cmd += Host_Name;
  cmd += inv;
  cmd += ",";
  cmd += inv;
  cmd += Password;
  cmd += inv;
  ser.println(cmd);
}

void loop()
{
  float Temp= dht.readTemperature();
  int analogSensor = analogRead(smokeA0);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("TEMP:");
  lcd.print(Temp);
  if (Temp >= 35.00) {
    digitalWrite(fan, HIGH);
    lcd.setCursor(10, 0);
    lcd.print("Fan on");
  }
  else if (Temp <= 32.00) {
    digitalWrite(fan, LOW);
    lcd.setCursor(10, 0);
```

```
  lcd.print("FanOff");
}
else{
  lcd.setCursor(10,0);
  lcd.print("Normal");
  }
Serial.print("Temperature: ");
Serial.print(Temp);
Serial.println();
lcd.setCursor(0, 1);
Serial.print("Pin A0: ");
Serial.println(analogSensor);
lcd.print("Smoke Lev:");
lcd.print(analogSensor);
// Checks if it has reached the threshold value
if (analogSensor > sensorThres)
{

  tone(buzzer, 1000, 200);
}
else
{
  noTone(buzzer);
}
delay(1000);
String state1 = String(Temp);              // Converting them to string
String state2 = String(analogSensor);          // as to send it through URL
String cmd = "AT+CIPSTART=\"TCP\",\"";         // Establishing TCP connection
cmd += "184.106.153.149";                    // api.thingspeak.com
```

```
cmd += "\",80";                          // port 80
ser.println(cmd);
Serial.println(cmd);
if (ser.find("Error"))
{
  Serial.println("AT+CIPSTART error");
  return;
}
String getStr = "GET /update?api_key=";        // prepare GET string
getStr += apiKey;
getStr += "&field1=";
getStr += String(state1);                  // Humidity Data
getStr += "&field2=";
getStr += String(state2);                  // Temperature Data
getStr += "\r\n\r\n";
cmd = "AT+CIPSEND=";
cmd += String(getStr.length());              // Total Length of data
ser.println(cmd);
Serial.println(cmd);

if (ser.find(">"))
{
  ser.print(getStr);
  Serial.print(getStr);
}
else
{
  ser.println("AT+CIPCLOSE");               // closing connection
  Serial.println("AT+CIPCLOSE");
```

```
  }
  delay(1000);

}

//
#include <LiquidCrystal.h>
int led = 7;                // the pin that the LED is atteched to
int sensor = 6;              // the pin that the sensor is atteched to
int state = LOW;             // by default, no motion detected
int val = 0;                 // variable to store the sensor status (value)

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
char * largeText=" Welcome To Home Automation System Using Arduino. ";
int icursor=0;
void setup() {
  pinMode(led, OUTPUT);      // initalize LED as an output
  pinMode(sensor, INPUT);    // initialize sensor as an input
  Serial.begin(9600);        // initialize serial
  lcd.begin(16,2);
  lcd.setCursor(2,1);
  lcd.print("NAGARJUNA.T");
```

```
}

void loop() {
  UpdateLCDDisplay();
  delay(800);
  val = digitalRead(sensor);   // read sensor value
  if (val == HIGH) {           // check if the sensor is HIGH
    digitalWrite(led, HIGH);   // turn LED ON
    delay(100);                // delay 100 milliseconds

    if (state == LOW) {
      Serial.println("Motion detected!");
      state = HIGH;       // update variable state to HIGH
    }
  }
  else {
      digitalWrite(led, LOW); // turn LED OFF
      delay(100);             // delay 200 milliseconds

      if (state == HIGH){
        Serial.println("Motion stopped!");
        state = LOW;       // update variable state to LOW
```

```
    }
  }
}

void UpdateLCDDisplay(){
  int ilenoflargetext=strlen(largeText);
  if (icursor==(ilenoflargetext-1)){
       icursor=0;
   }
   lcd.setCursor(0,0);
   if (icursor<ilenoflargetext-16){
    for (int ichar = icursor;ichar<icursor+16;ichar++){
      lcd.print(largeText[ichar]);
      }
    }
    else{
      for (int ichar=icursor;ichar<(ilenoflargetext-1);ichar++){
        lcd.print(largeText[ichar]);
        }
        for(int ichar=0;ichar<=16-(ilenoflargetext-icursor);ichar++){
          lcd.print(largeText[ichar]);
          }
```

```
        }

        icursor++;

    }
```

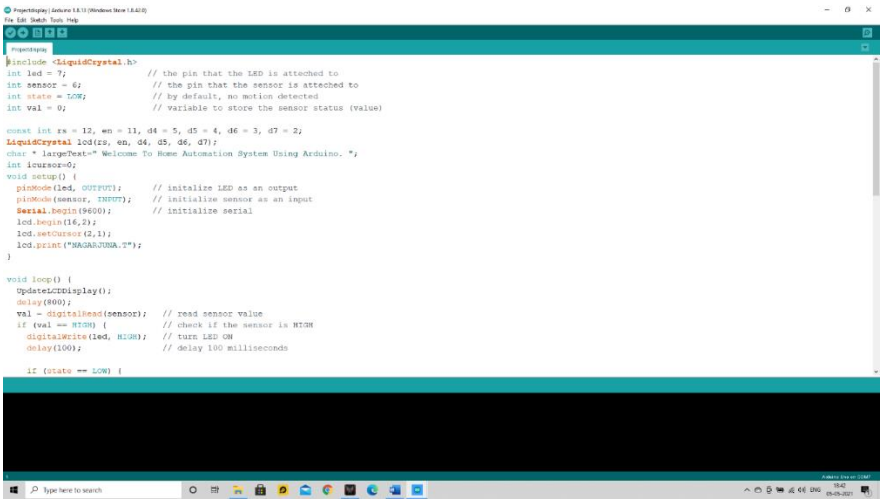

Fig.4.4 code Window

4.2.2 Compile and Upload

After written source code next compile the source code. Click on button. If the no errors are occur in compile time, upload the code into Arduino by click on button.

4.3 Working

The working in the HAS. When the connection is established, it will start reading the parameters of sensors like p1, p2, p3 etc. The threshold levels for the required sensors are set as t1, t2, t3 etc. The sensor data are sent to the web server and stored in the cloud. The data can be analyzed anywhere any time. If the sensor parameters are greater than the threshold level then the respective alarm a1, a2, a3 etc. will be raised

and the required actuation is done for the controlling of the parameters. At the door of the house a motion sensor is fixed to detect any movement near the door. Light 1 will turn on automatically when light sensor detects the darkness. A cooler/Fan will turn on when the room temperature exceeds the set threshold and in turn reduces the room temperature. The gas sensor MQ-5 is placed in the kitchen to detect any gas leakage, if any leakage is detected the alarm in the hall is raised. Relay is used to switch the electrical appliances like light, fan etc.

4.4 Test Cases

A **test case** is a specification of the inputs, execution conditions, testing procedure, and expected results that define a single test to be executed to achieve a particular software testing objective, such as to exercise a particular program path or to verify compliance with a specific requirement. Test cases underlie testing that is methodical rather than haphazard. A battery of test cases can be built to produce the desired coverage of the software being tested. Formally defined test cases allow the same tests to be run repeatedly against successive versions of the software, allowing for effective and consistent regression testing.

4.4.1 Formal Test Cases

In order to fully test that all the requirements of an application are met, there must be at least two test cases for each requirement: one positive test and one negative test. If a requirement has sub-requirements, each sub-requirement must have at least two test cases. Keeping track of the link between the requirement and the test is frequently done using a traceability matrix. Written test cases should include a description of the functionality to be tested, and the preparation required to ensure that the test can be conducted. A formal written test-case is characterized by a known input and by an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a postcondition.

4.4.2 Informal Test Cases

For applications or systems without formal requirements, test cases can be written based on the accepted normal operation of programs of a similar class. In some schools of testing, test cases are not written at all but the activities and results are reported after the tests have been run. In scenario testing, hypothetical stories are used to help the tester think through a complex problem or system. These scenarios are usually not written down in any detail. They can be as simple as a diagram for a testing environment or they could be a description written in prose. The ideal scenario test is a story that is motivating, credible, complex, and easy to evaluate. They are usually different from test cases in that test cases are single steps while scenarios cover a number of steps of the key.

Testing Levels

S.No	Testcase Description	Actual value	Expected Value	Result
1	Connections	Correct	Correct	Pass
2	Power supply	5v	5v	Pass
3	Code	Correct logic	Correct logic	Pass
4	DHT11 Sensor	Fan on	Read room Temperature fan on	pass
5	PIR Sensor	Light on	Lamp on while any move detected light on	pass
6	MQ7 Sensor	Buzzer on	Smoke/gas Detected buzzer rings	pass
7	Mobile app	connected	To connect the WIFI	pass

Table 4.1 Testcases

S.no	Testcase Description	Actual value	Expected Value	Result
1	Connections	Incorrect	Correct	Fail
2	Power supply	3v	5v	Fail
3	Code	incorrect logic	Correct logic	Fail
4	DHT11 Sensor	Not reading room temperature	Read room Temperature fan on	Fail
5	PIR Sensor	If move detected Light off	Lamp on while any move detected light on	Fail
6	MQ5 Sensor	Gas detected but not ring alarm	Smoke/gas Detected buzzer rings	Fail
7	Mobile app	Not detected	To connect the WIFI	Fail

Table 4.2 Test cases1

4.5 Output screenshots:

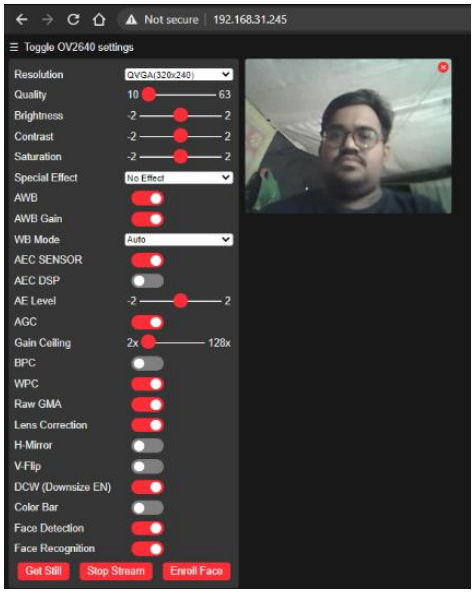

Fig 4.5 user interface of doorlock

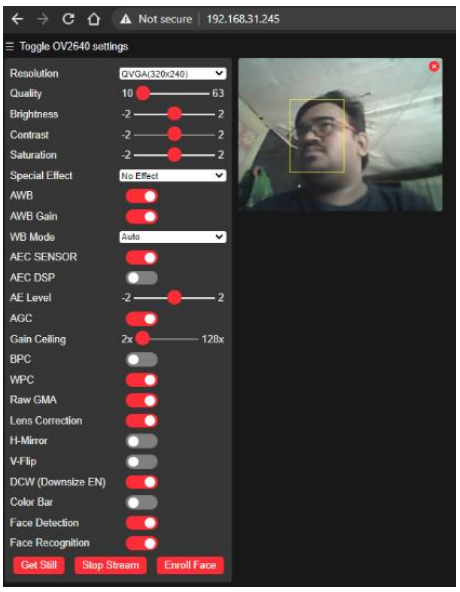

Fig 4.6 Enroll face

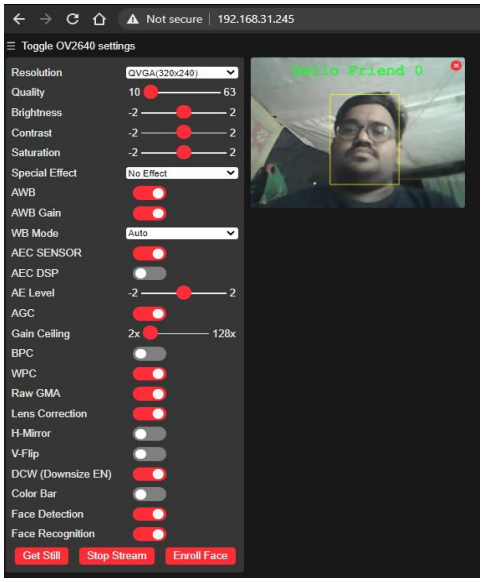

Fig 4.7 Facematched

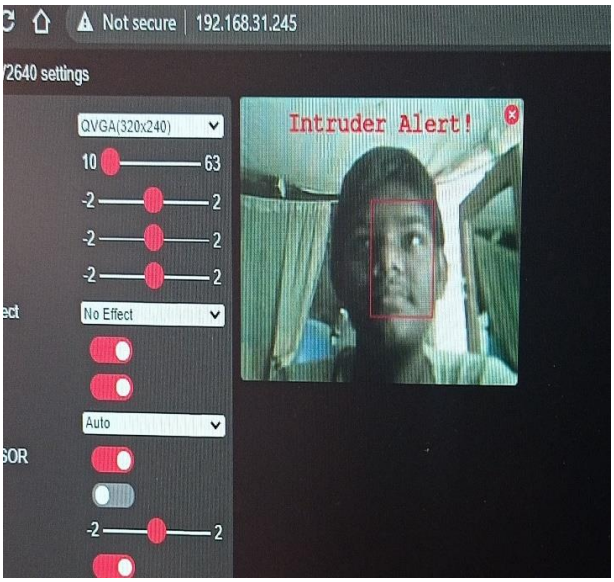

Fig 4.8 Face not matched

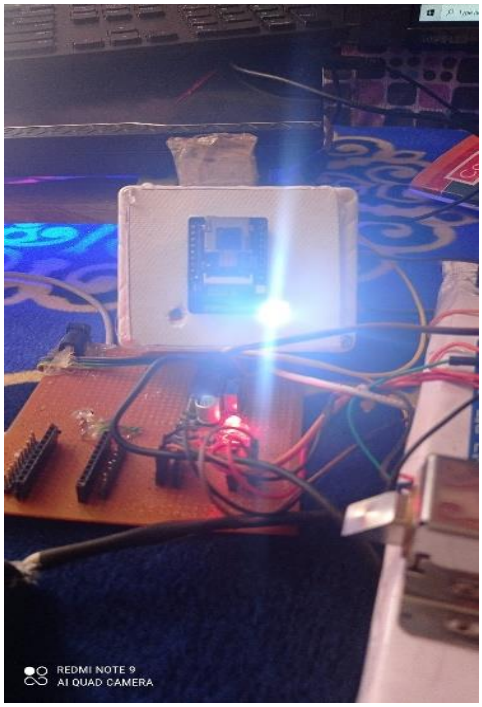

Fig 4.9 door lock is lock

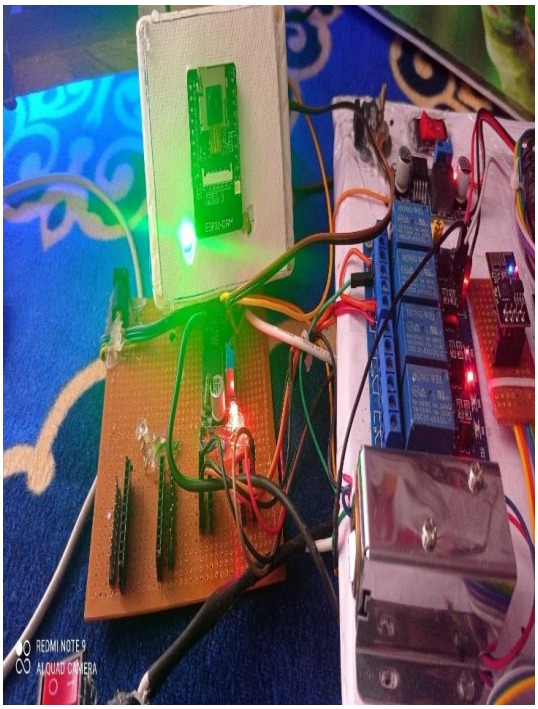

Fig 4.10 door lock is unlocke

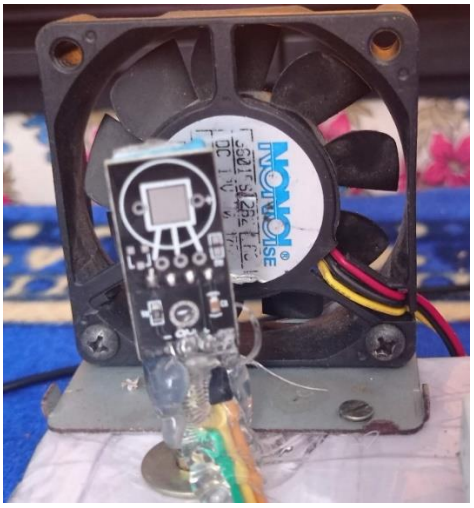

Fig 4.11 Fan off state

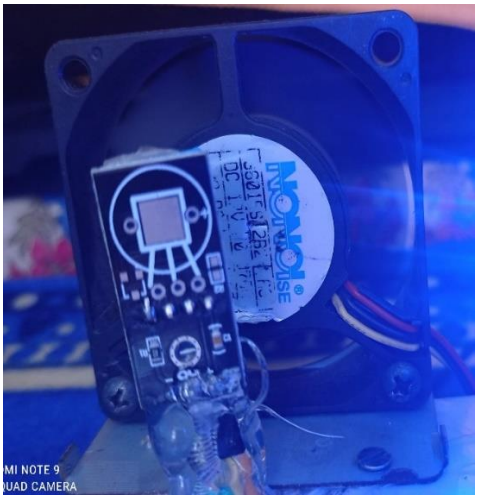

Fig 4.12 Fan on state

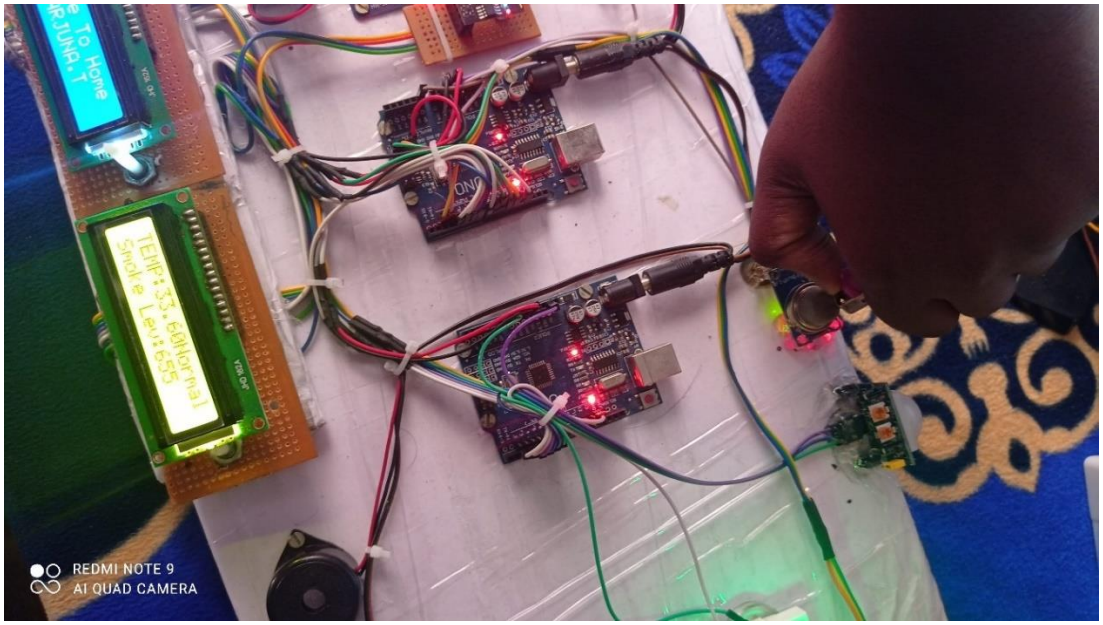

Fig 4.13 Gas detection and shown leavel

Fig 4.14,4.15 Display Temp-level and somke level

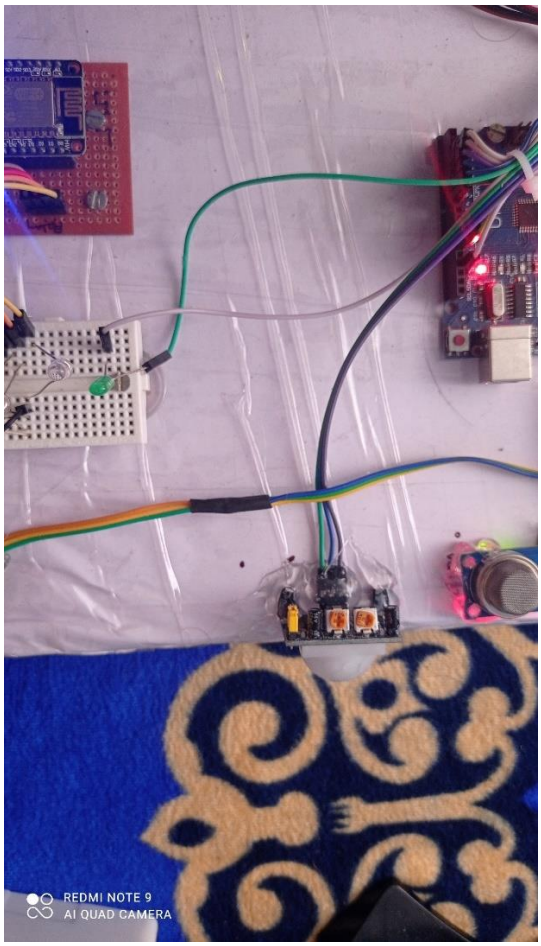

Fig 4.16 Motion not Detected

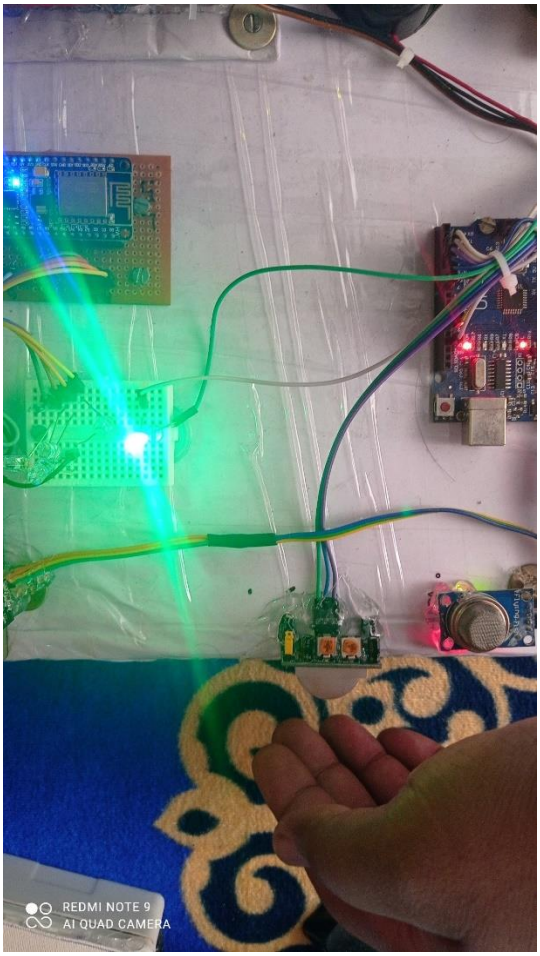

Fig 4.17 Motion Detected

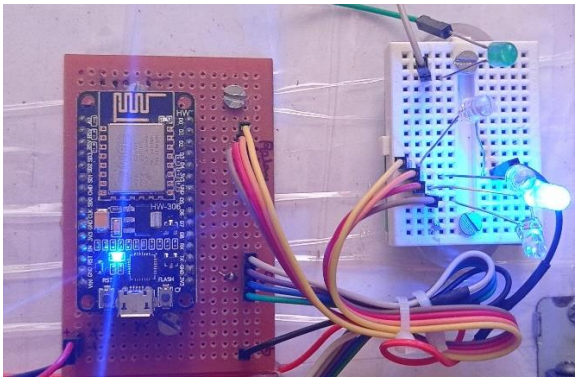

Fig 4.18 Home appliances

Fig 4.19 switch contorl

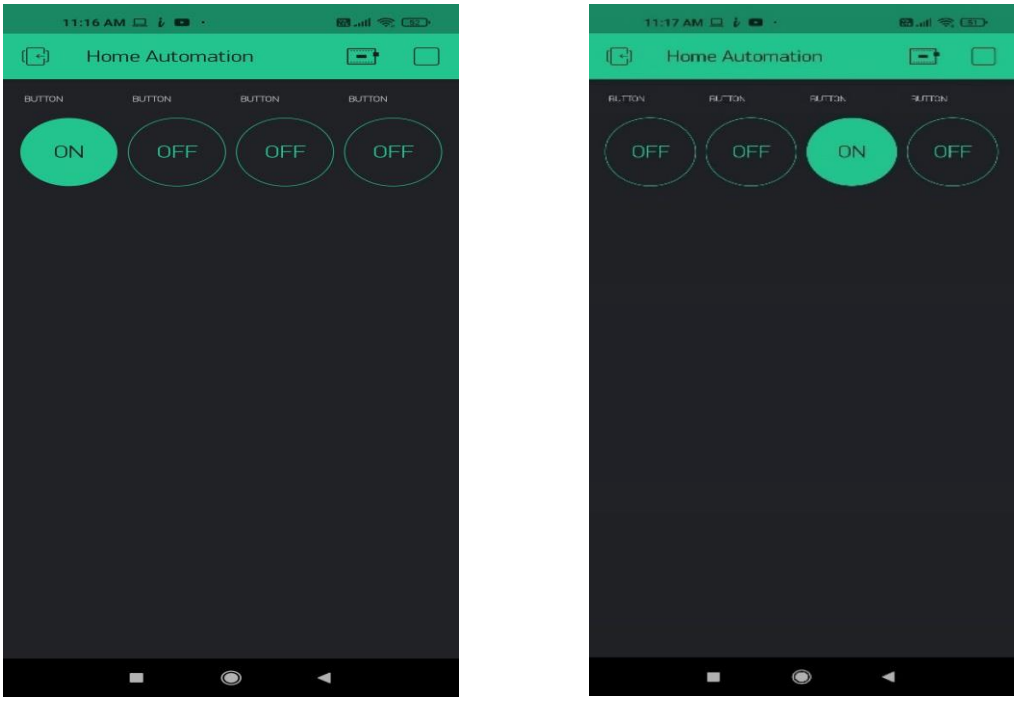

Fig 4.20,4.21 Controll Home appliance by using mobile phone

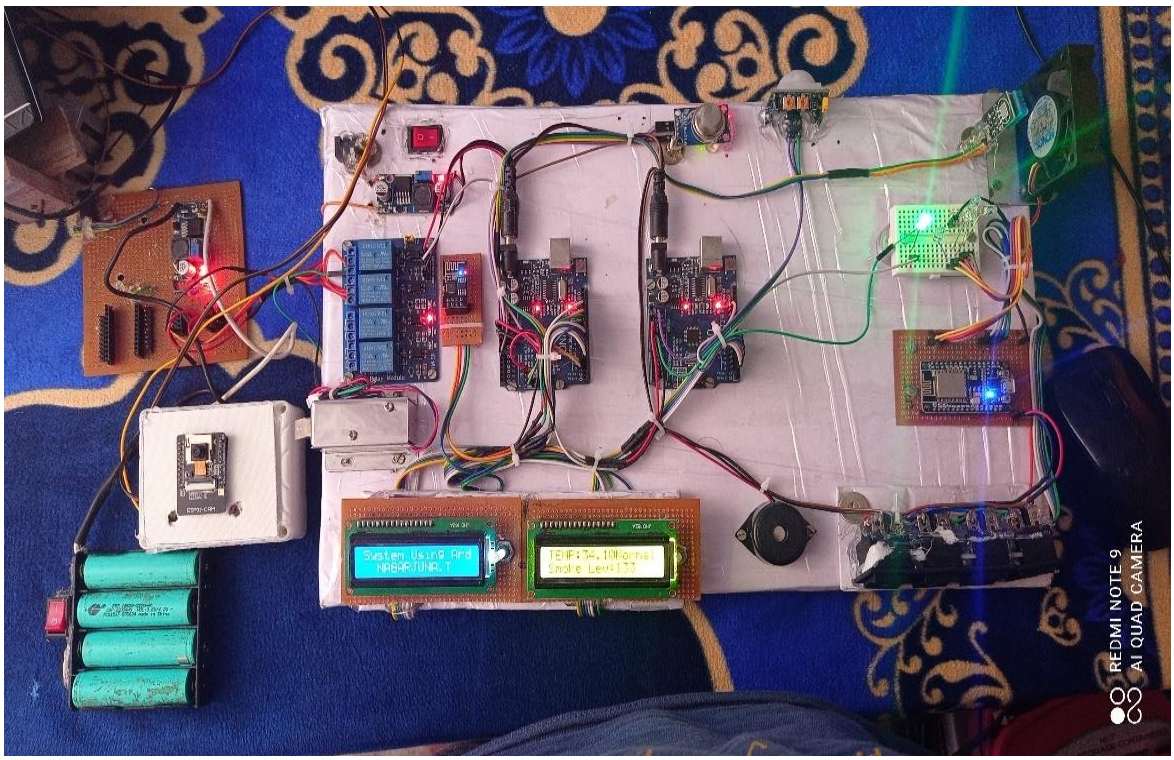

Fig 4.22 Toatal project

CHAPTER 5

CONCLUSION

Design and implementation concept for a Home automation system based on Arduino microcontroller board. Sensors are connected to the microcontroller board. The home appliances can be monitored, controlled and accessed automatically in response to any signals came from related sensors. The system relies on power supply. If power supply fails, the connection will be halted and SMS alarm functions will be stopped. The security system is powered by another power source for security safety. Without security system, it doesn't complete the whole system. The proposed system is shown to be a simple, cost effective and flexible

The home automation using Arduino has been experimentally proven to work satisfactorily by connecting simple appliances to it and the appliances were successfully controlled remotely through internet. The designed system not only monitors the sensor data, like temperature, gas, light, motion sensors, but also actuates a process according to the requirement, for example switching on the light when it gets dark. It also stores the sensor parameters in the cloud (Gmail)in a timely manner. This will help the user to analyze the condition of various parameters in the home anytime anywhere.

FUTURE SCOPE:

Using this system as framework, the system can be expanded to include various other options which could include home security feature like capturing the photo of a person moving around the house and storing it onto the cloud. This will reduce the data storage than using the CCTV camera which will record all the time and stores it. The system can be expanded for energy monitoring, or weather stations. This kind of a system with respective changes can be implemented in the hospitals for disable people or in industries where human invasion is impossible or dangerous, and it can also be implemented for environmental monitoring

CHAPTER 6

REFERENCES

1. Chandramohan, J., Nagarajan, R., Satheeshkumar, K., Ajithkumar, N., Gopinath, P.A. and Ranjithkumar, S., 2017. Intelligent smart home automation and security system using Arduino and Wi-fi. *International Journal of Engineering And Computer Science (IJECS)*, *6*(3), pp.20694-20698.
2. Sivapriyan, R., K. Manisha Rao, and M. Harijyothi. "Literature Review of IoT based Home Automation System." In *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*, pp. 101-105. IEEE, 2020.
3. Singh, Harsh Kumar, Saurabh Verma, Shashank Pal, and Kavita Pandey. "A step towards Home Automation using IOT." In *2019 Twelfth International Conference on Contemporary Computing (IC3)*, pp. 1-5. IEEE, 2019.
4. Bhat, Abhishek, Satvik Sharma, K. R. Pranav, and H. G. Monika Rani. "Home automation using internet of things." *International Research Journal of Engineering and Technology (IRJET)* 4, no. 07 (2017).
5. Naing, Ma, and N. N. S. Hlaing. "Arduino Based Smart Home Automation System." *Int. J. Trend Sci. Res. Dev.* 3, no. 4 (2019): 276-280.
6. Sulayman, Iman IM Abu, Sami HA Almalki, Mohamed S. Soliman, and Majed O. Dwairi. "Designing and Implementation of Home Automation System Based on Remote Sensing Technique with Arduino Uno Microcontroller." In *2017 9th IEEE-GCC Conference and Exhibition (GCCCE)*, pp. 1-9. IEEE, 2017.
7. https://www.viralsciencecreativity.com/post/esp32-cam-face-detection-door-lock-system
8. https://www.pantechsolutions.net/iot-based-home-automation-system-using-android-application
9. http://www.ijirset.com/upload/2016/iccstar/37_O032_CAMERAREADY.pdf