IOT BASED USER COMMAND ANALYSIS

Main Project Report

Submitted by

ANGELA SONY

Reg No : FIT20MCA-2024

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Computer Applications
Of
A P J Abdul Kalam Technological University

Focus on Excellence

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®
ANGAMALY-683577, ERNAKULAM(DIST)
JULY 2022

DECLARATION

I, **ANGELA SONY** hereby declare that the report of this project work, submitted to the Department of Computer Applications, Federal Institute of Science and Technology (**FISAT**), Angamaly in partial fulfillment of the award of the degree of Master of Computer Application is an authentic record of my original work.

The report has not been submitted for the award of any degree of this university or any other university.

Date : **Signature :**

Place: Angamaly **Name :**

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®
ANGAMALY, ERNAKULAM-683577

DEPARTMENT OF COMPUTER APPLICATIONS

Focus on Excellence

CERTIFICATE

This is to certify that the project report titled **’IoT based User Command Analysis’** submitted by **Angela Sony [Reg No: FIT20MCA-2024]** towards partial fulfillment of the requirements for the award of the degree of Master of Computer Applications is a record of bonafide work carried out by her during the year 2022.

Project Guide **Head of the Department**

Submitted for viva-voice held on.......................................at................................

ACKNOWLEDGEMENT

I am deeply grateful to **Dr. Manoj George** , Principal, FISAT, Angamaly for providing me with adequate facilities, way and means by which I was able to complete my main project work and I express my sincere thanks to **Dr. C Sheela** ,Vice Principal FISAT, Angamaly.

My sincere thanks to **Dr. Deepa Mary Mathews**, Head of the department of MCA, FISAT, who had been a source of inspiration. I express heartiest thanks to **Dr.Shahna K U** my project guide for the encouragement and valuable suggestion.I express my heartiest gratitude to my scrum master **Dr.Rakhi Venugopal** and all faculty members in my department for their constant encouragement and never ending support throughout the project.I would also like to express my sincere gratitude to the lab faculty members for their guidance.

Finally I express my thanks to all my friends who gave me wealth of suggestions for successful completion of this project.

ABSTRACT

As the IoT (Internet of Things) has been commercialized recently, studies are underway for user-customized services. Accordingly, the service should be changed according to the characteristics of the user rather than the unified service. However, when existing systems operate automatically, there is a problem of providing a uniform service to all users without providing a customized service.

Internet of things (IoT)-based home automation system which can be regulated and accessed to by the cell phones. This system can perform various activities of home from anyplace of the world. IoT has been marketed as of late; studies are in progress for user-customized services. Appropriately, the services ought to be changed by the attributes of the user instead of the brought together services. Nonetheless, when existing system work naturally, there is an issue of giving a uniform service to all clients without giving customized services. To handle this issue, we propose an IoT-based regulator system for breaking down user directions. For performing such functionality, a functional programming is done through the DTMF Signal is a standard, which permits organizing the information edge originating from the sensors and in this manner regulator of the data.

The principle capacity of this undertaking is finished using capacity programming with DTMF signal which permits structuring the data frame originating from the sensors and in this manner the regulate of the data. Through this, it is possible to provide services with improved user convenience and system precision device as a UI.Nonetheless, when existing system work naturally, there is an issue of giving a uniform service to all clients without giving customized services. They can speak with home automation system 124 through an Internet gateway, by methods for low power communication protocol like Zigbee, Wi-Fi, and so on. In this propose system, we mainly work on temperature sensor and it is analyzed that our system improves switching time and the operating accuracy of work up to 30.

Introducing an IoT automatic control system for analyzing user commands. The system collects the user’s remote control commands to organize the user data sets,classify the collected user’s commands according to the devices, and classify the user’s work orders in time to derive the average operation of the devices. Through this, it is possible to provide services with improved user convenience and system accuracy..

Contents

Chapter 1

INTRODUCTION

Currently, the IoT technologies being studied include user pattern analysis and inter-device collaboration to provide customized services. When IoT is used inside the home, it arranges the smart device and connects it to a single network for data communication. Applications can also be used to remotely control smart devices or monitor the internal environment measured by sensors. Accordingly, the smart device in the home is controlled to provide the service to the user.

Existing IoT systems provide services such as environment control and repeating operation. As the work progresses automatically, there is a problem with accuracy. Accordingly, the user should be able to receive the service at a desired time. Also, it is required to provide a user oriented service by changing the system according to the user.

Chapter 2

PROOF OF CONCEPT

The project titled smart IoT based user command analysis propose to develop a system that automatically analyzes the user’s remote control commands and then controls the environment inside the home. It also visualizes and displays smart device usage over time. Then, the user can operate by selecting the automatic operation mode and the remote control mode. In this way, since the operation inside home is performed by analyzing the user’s smart device usage record, it is possible to provide a user-customized service. As a result, it is possible to provide a service with higher accuracy unlike the existing system.

2.1 Objectives

Recently, researches are being conducted to build a smart environment using various sensors and devices and to provide customized services to users through IoT (Internet of Things). The existing IoT system operates the device only by using the threshold value. So, there are problems that device operates when a service that does not consider a user characteristic is provided or when user not necessary. In this paper, to solve these problems, the system collects the remote control commands and compares it with the value of device operation threshold and sets the changed threshold value. In addition, devices not registered in the server constitute the environment by linking with existing task. Therefore, it is possible to provide a customized service to the users.

2.2 Existing System

IoT is evolving into a system for intelligent services where people, processes, data and objects are interconnected. In the past, it only supported inter-object communication via M2M (Machine To Machine), but it is evolving as a step to decide between the objects. It is also a step of recognizing and controlling the situation in the surroundings where the person is not aware.To provide these services, it is necessary to use various sensors to measure environmental data. In addition, the data generated by objects and people included in the IoT network is increasing exponentially, and there are requirements to be solved by IoT systems such as real-time processing capability and accurate semantic information. Existing IoT systems are transmit data measured through a sensor to a server, load it in a database, and extract and provide data according to the user's request. Such a system operates the device through a threshold value in the server, but does not consider the characteristics of the user, and thus the device operates even when the user does not want it

2.3 Proposed System

To solve this problem, this paper proposes an IoT automatic control system for analyzing user commands. The system collects the user's remote control commands to organize the user data sets, classify the collected user's commands according to the devices, and classify the user's work orders in time to derive the average operation of the devices. Through this, it is possible to provide services with improved user convenience and system accuracy.

Chapter 3

IMPLEMENTATION

The proposed system utilizes the application to collect user commands and monitor the status of the device. It can also provide the user with visualized information or change the operating mode. The user can change the status of the device by using the remote control function and can check the status of the devices by utilizing the monitoring function. You can also check the data measured by sensors in the home.This project is to build a system that automatically analyzes the user’s remote control commands and then controls the environment inside the home. It also visualizes and displays smart device usage over time. Then, the user can operate by selecting the automatic operation mode and the remote control mode. In this way, since the operation inside home is performed by analyzing the user’s smart device usage record, it is possible to provide a user-customized service. As a result, it is possible to provide a service with higher accuracy unlike the existing system.

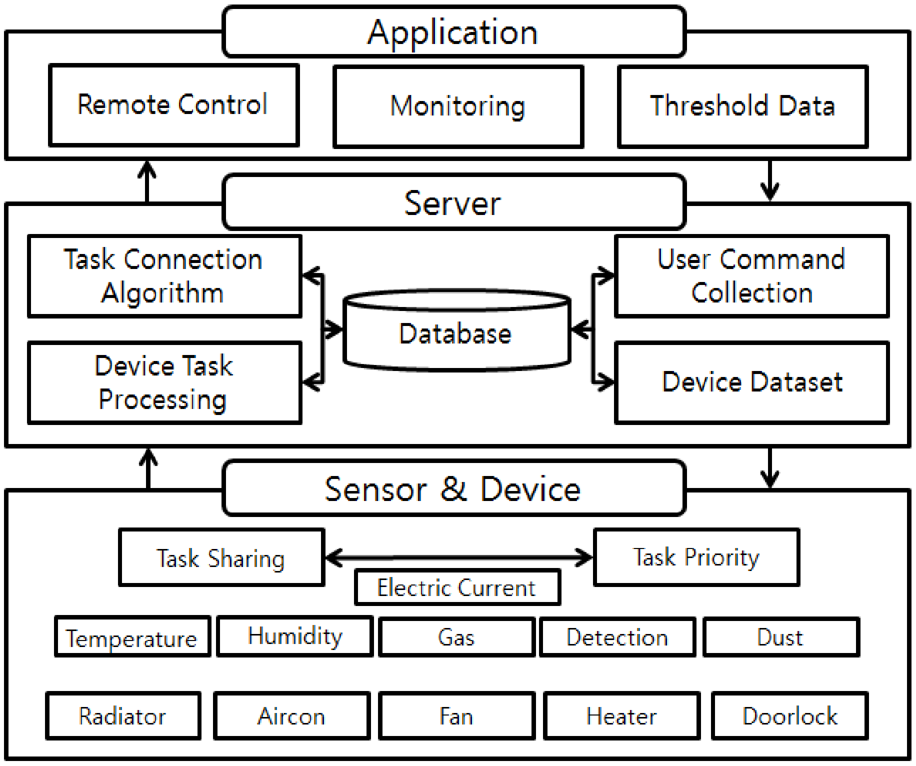

Figure 3.1: System architecture

3.1 Architecture

The system is an automatic control system and is a system in which the threshold changes when a user operates the device through remote control. Arduino is equipped with sensors and devices and communicates with the server via Wi-Fi. The server compares the sensor data with the threshold value to determine whether a task has occurred or not, and transmits a control command to the device. And the server loads the results into the database and provides them to the user application. Figure 3.1 shows the system architecture.

The sensor and the device measure the surrounding environment data in the home and transmit it to the server. When the environment change, the device operates by the threshold value and keeps it in a pleasant state. The server consists of a database and modules for data processing. The database stores sensor data for monitoring, device status, and operation thresholds. The application can monitor and control the sensor data and device status by the user. You can also check device-specific thresholds and power usage.Figure 3.2 shows a flow diagram of the algorithm for associating heterogeneous devices with existing tasks.

Data about sensor and device connected to Arduino is transmitted to server. The server judges whether or not the corresponding data is registered in the server. If it is registered, it does not need to associate with the existing task and the normal operation proceeds. If not registered, a dataset for the device is created and loaded into the database. Confirm that the added heterogeneous device is operating. When an existing operation is in operation, it is compared whether the device changes environment analyze the operation of the device. If the task is not running, it operates as an automatic control task. Therefore, the user advances the connected each time the remote control of heterogeneous devices. Figure 3.3shows the task processing priority in the server.

The first priority task is fire caused by temperature or gas sensor. If the temperature data does not exceed the threshold value and the gas threshold value is exceeded, it sends a message to the user that the gas has leaked and operates the window device to ventilate. If the threshold of the temperature is exceeded and the threshold of the gas is exceeded, it is identified as a fire, and after the message of fire occurrence is sent to the manager, all devices stop. If the temperature only exceeds the threshold, the device for temperature control operates.

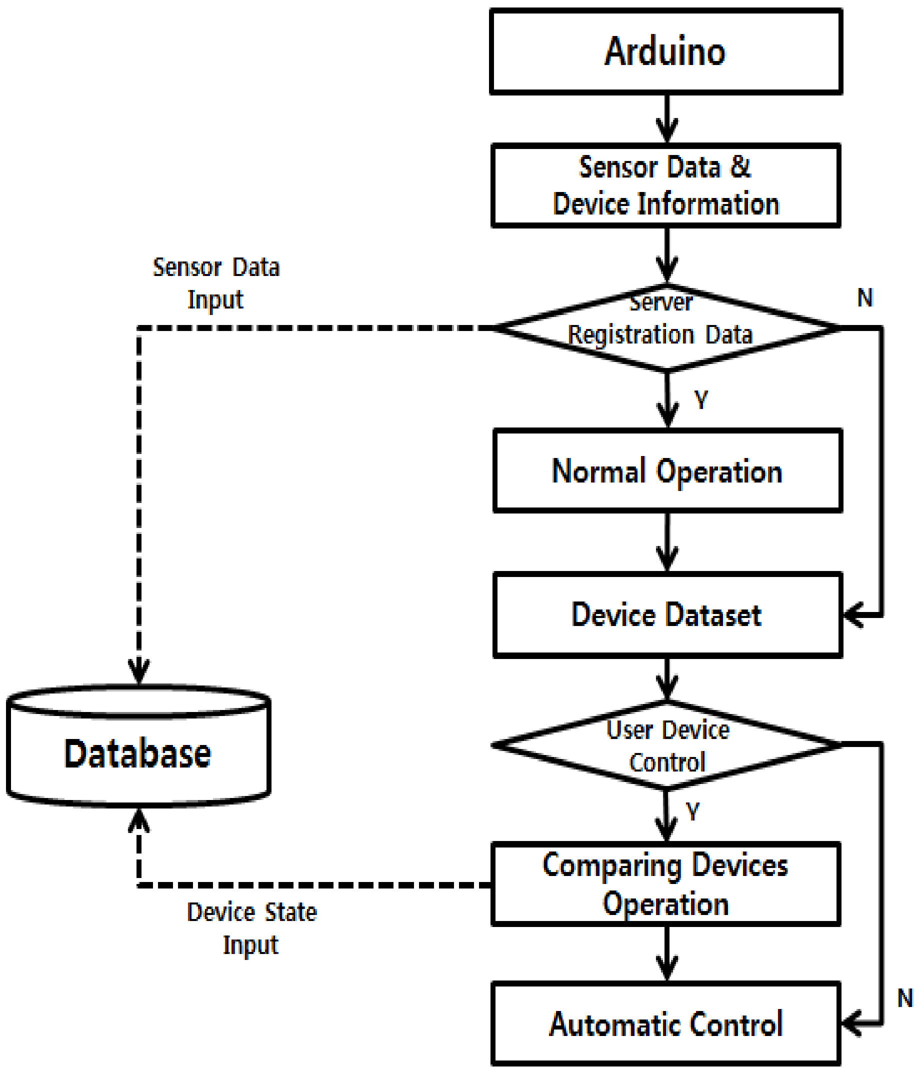

Figure 3.2: Device connection algorithm flowchart

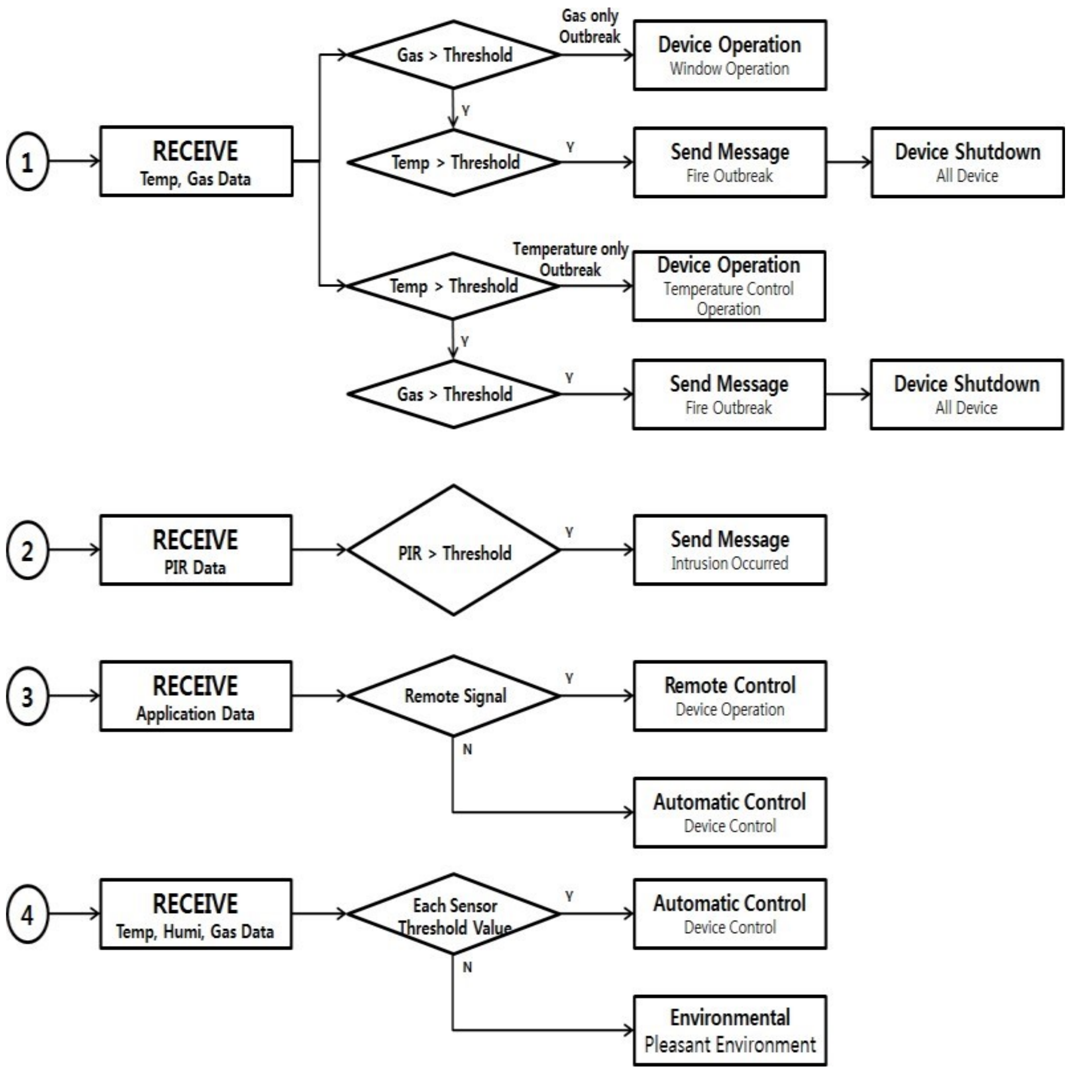

Figure 3.3: Step-by-step task priority setting

The second priority task is a door lock control by human detection sensor. If there is a user inside the home, the device for temperature control or ventilation in the server will operate. If there is no user in the home, the device does not operate.

The third priority task is remote control using user application. When a device remote control signal is generated by a user, a control command is transmitted to the corresponding device, and otherwise, the device operates by automatic control. Also, if the device is operated due to remote control, the device with lower priority stops.

The fourth priority task is an automatic control using temperature, humidity and gas sensor. The sensor data is used to determine whether an operation has occurred through the threshold value in the server, and a device for environmental control operates when temperature regulation and ventilation work occurs.

3.2 HARDWARE REQUIREMENTS

3.2.1 Arduino Uno

Arduino Uno-ATMEGA328P : Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions. You can control your board functions by sending a set of instructions to the micro-controller on the board via Arduino IDE (referred to as uploading software). Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable. Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

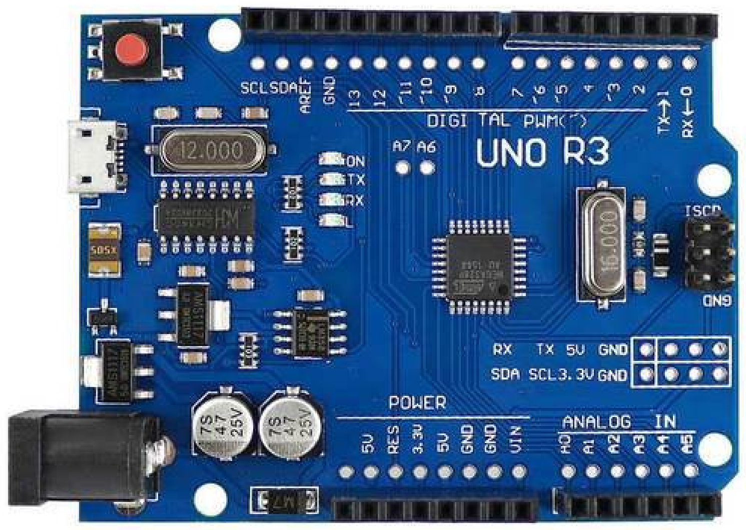

Figure 3.4: Arduino Uno

- Microcontroller : Microchip ATmega328P
- Operating Voltage : 5 Volts
- Input Voltage : 7 to 20 Volts
- Digital I/O Pins : 14 (of which 6 can provide PWM output)
- UART: 1, I2C: 1, SPPI : 1
- Analog Input Pins : 6
- DC Current per I/O Pin : 20 mA
- DC Current for 3.3V Pin : 50 mA

- Flash Memory : 32 KB of which 0.5 KB used by bootloader
- SRAM: 2 KB, EEPROM : 1 KB
- Clock Speed : 16 MHz
- Length : 68.6 mm
- Width : 53.4 mm
- Weight : 25 g

3.2.2 NodeMCU

ESP8266: NodeMCU is an open-source Lua based firmware and development board specially targeted for IoT based Applications. It includes firmware that runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.he NodeMCU ESP8266 development board comes with the ESP-12E module containing the ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. NodeMCU 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. NodeMCU can be powered using a Micro USB jack and VIN pin (External Supply Pin). It supports UART, SPI, and I2C interface.

Figure 3.5: NODEMCU

- Micro-controller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1

- Flash Memory: 4 MB
- SRAM: 64 KB
- Clock Speed: 80 MHz
- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antennaeliability of our servos have made them the favorite choice of many RC hobbyists.

3.2.3 L29N

L29: The L298N is an integrated monolithic circuit in a 15- lead Multiwatt and PowerSO20 packages. It is a high voltage , high current dual full-bridge driver de-signed to accept standard TTL logic level sand drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the in-put signals .The emitters of the lower transistors of each bridge are connected together rand the corresponding external terminal can be used for the connection of an external sensing resistor. An additional Supply input is provided so that the logic works at a lower voltage.

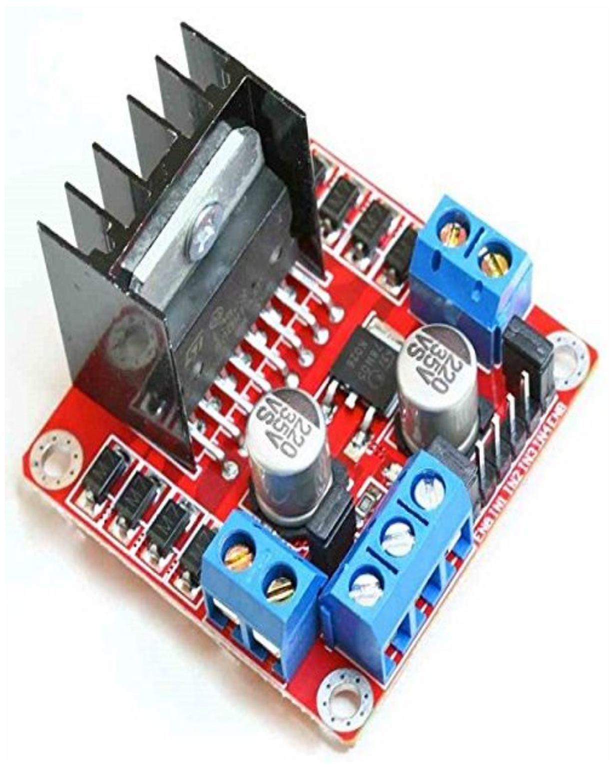

Figure 3.6: L29N

3.2.4 12V Fan

12V 8025 DC Cooling Fan – 3 inch are operating at 12V with a dimension of 80x80mmx25mm. They are typically found in ATX Computer cases, servers, and other enclosed equipment but they can also be used in a variety of other projects requiring moderate airflow.The fan spins at 2600 RPM and can move approximately 30CFM. It is fairly quiet – just 30.7dBA. To use the fan, simply connect Red to 12V, Black to Ground.

Figure 3.7: 12V Fan

3.2.5 DHT 11

The DHT11 is a commonly used Temperature and humidit sensor that comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data.

- DHT11 Specifications
- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C

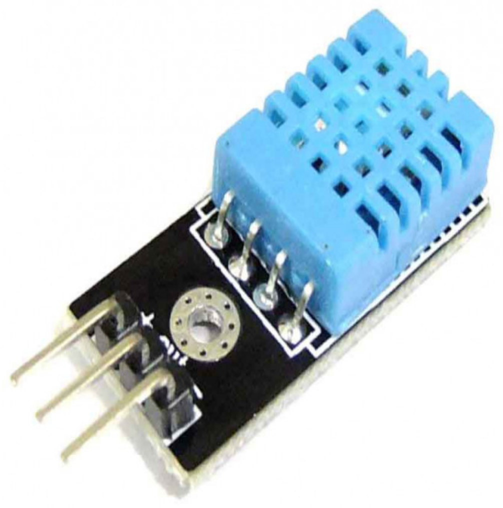

Figure 3.8: DHT11

- Humidity Range: 20 to 90
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1

3.2.6 ds18b20

The DS18B20 is one type of temperature sensor and it supplies 9-bit to 12-bit readings of temperature. These values show the temperature of a particular device. The communication of this sensor can be done through a one-wire bus protocol which uses one data line to communicate with an inner microprocessor. The digital temperature sensor like

DS18B20 follows single wire protocol and it can be used to measure temperature in the range of -67oF to +257oF or -55oC to +125oC with +-5 accuracy. The range of received data from the 1-wire can range from 9-bit to 12-bit. Because, this sensor follows the single wire protocol, and the controlling of this can be done through an only pin of Microcontroller. This is an advanced level protocol, where each sensor can be set with a 64-bit serial code which aids to control numerous sensors using a single pin of the microcontroller. The specifications of this sensor include the following.

Figure 3.9: ds18b20

- This sensor is a programmable and digital temperature sensor
- The communication of this sensor can be done with the help of a 1-Wire method

- The range of power supply is 3.0V – 5.5V
- Fahrenheit equal s to -67°F to +257°F
- The accuracy of this sensor is ±0.5°C
- The o/p resolution will range from 9-bit to 12-bit
- It changes the 12-bit temperature to digital word within 750 ms time
- This sensor can be power-driven from the data line
- Alarm options are programmable
- The multiplexing can be enabled by Unique 64-bit address
- The temperature can be calculated from -55°C to +125°C.
- These are obtainable like SOP, To-92, and also as a waterproof sensor

3.3 SOFTWARE REQUIREMENTS

3.3.1 Arduino 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 GNU 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 tool chain, 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.

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extensionino. 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 right hand corner of the window displays the configured board and serial port.

3.3.2 Features of Arduino IDE

- Multi-Platform Application Arduino IDE works on the three most popular operating systems: Windows, Mac OS, and Linux. Aside from that, the application is also accessible from the cloud. These options provide programmers with the choice of creating and saving their sketches on the cloud or building their programs locally and upload it directly to the board.
- Board Management Arduino IDE comes with a board management module, where users can select the board they want to work with at the moment. If they wish to change it, they can do so easily from the drop down menu. Modifying their selection also automatically updates the PORT infos with the data they need in relation to the new board.
- Straightforward Sketching With Arduino IDE, users can create programs called sketches that are built with a text editor. The process is a straightforward one though it has several bells
- The tool is armed with a board management module, wherein users can choose which board they want to use. If another board is needed, they can seamlessly select another option from the dropdown menu. PORT data is updated automatically whenever modifications are made on the board or if a new board is chosen.

3.3.3 Blynk IoT

At our core there is suite of powerful software products to build IoT apps, connect things, people, and data.Blynk is a new platform that allows you to quickly build interfaces for controlling and monitoring your hardware projects from your iOS and Android device. After downloading the Blynk app, you can create a project dashboard and arrange buttons, sliders, graphs, and other widgets onto the screen. Using the widgets, you can turn pins on and off or display data from sensors.

Currently, Blynk supports most Arduino boards, Raspberry Pi models, the ESP8266, Particle Core, and a handful of other common microcontrollers and single-board computers, and more are being added over time. Arduino Wi-Fi and Ethernet shields are supported, though you can also control devices plugged into a computer's USB port as well.While there are other platforms for controlling hardware over the internet(Particle, ThingSpeak, Temboo, IFTTT), Blynk is one of the most user-friendly I've seen yet, and it's also free and open-source under an MIT license.

3.3.4 C (Programming language)

C was originally developed at Bell Labs by Dennis Ritchie, between 1972 and 1973.It was created to make utilities running on Unix. Later, it was applied to reimplementing the kernel of the Unix operating system.During the 1980s, C gradually gained popularity. Nowadays, it is one of the most widely used programminglanguages.

With C compilers from various vendors available for the majority of existing computer architectures and operating systems. C has been standardized by the American National Standards Institute (ANSI) since 1989 (see ANSI C) and subsequently by the International Organization for Standardization (ISO).

C is an imperative procedural language. It was designed to be compiled using a relatively straightforward compiler, to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal run time support. Despite its low-level capabilities, the language was designed to encourage cross-platform programming.

A standards compliant C program that is written with portability in mind can be compiled for a wide variety of computer platforms and operating systems with few changes to its source code; the language has become available on various platforms, from embedded micro-controllers to supercomputers.

3.3.5 Features of C

- It is a robust language with rich set of built-in functions and operators that can be used to write any complex program.
- The C compiler combines the capabilities of an assembly language with features of a high-level language.
- It is many time faster than BASIC.
- C is highly portable this means that programs once written can be run on another machines with little or no modification.
- Another important feature of C program, is its ability to extend itself.
- A C program is basically a collection of functions that are supported by C library. We can also create our own function and add it to C library.
- C language is the most widely used language in operating systems and embedded system development today.
- Unlike assembly language, c programs can be executed on different machines with some machine specific changes. Therefore, C is a machine independent language.

3.4 Modules

In this project,it divided into three modules.

- Owner authentication
- Temperature Detection
- Timer based detection

3.4.1 Owner authentication

Owner authentication: Your very first step will be to create an account in BLYNK app.

- In the Blynk app or in Blynk.Console
- Navigate to My Profile / User profile in the left menu
- Check that the Developer Mode switch is set to ON

Developer is a special user who has access to all of the functionality required to configure the platform for use by end-users.Once you are in Developer mode, you can start working on your first Device Template. Device Template is a set of configurations inherited by devices of a similar type.

- Activate your first device: To start using Blynk. Cloud you should assign a unique AuthToken to each device. AuthToken aims to identify the device in the Blynk Cloud. There are a few ways of getting Auth Tokens for your device and they may vary depending on the hardware, connectivity, and the IoT use-case you are working on.
- Activating Device with a Static Auth Token (for Ethernet, cellular, and other connection types):This method is recommended for devices that can connect to the Internet using Ethernet, Cellular (2G, 3G, 4G,

LTE), or other types of connection (that don't require custom WiFi credentials for example).

– Send Device Data to Blynk:To send messages from the app to the code that's running on your board (via the Blynk server) you will use virtual pins.

3.4.2 Temperature Detection

Temperature Detection: The prototype of the proposed system is composed of four kinds of sensors: temperature sensor, humidity sensor, gas sensor and human detection sensor. The devices that operate are air conditioners, heaters, windows and door locks. The data measured through the sensor is initiated through the thresholds in the device and the system proceeds through sharing and comparison. The temperature and humidity sensor operate the fan and the heater related to the temperature control, and the gas sensor performs the ventilation according to the gas generation. The human detection sensor confirms whether there is a user in the management place and controls the door lock.The working of the DHT sensor is pretty simple. DHT11 sensor consists of a capacitive humidity sensing element and a thermistor for sensing temperature. The humidity sensing capacitor has two electrodes with a moisture-holding substrate as a dielectric between them. Change in the capacitance value occurs with the change in humidity levels. The IC measure, process this changed resistance values and change them into digital form.

3.4.3 Timer based Detection

Timer based Detection: Allows time-controlled or temperature-controlled switching, or both combined. In Time mode there are three possible On/Off time pairs in a 24 hour cycle. In Temperature mode, the device switches itself on/off when the value falls below a freely-set temperature

Chapter 4

RESULT ANALYSIS

Existing systems have been designed to utilize various sensors to configure a smart environment and provide automation services. However, there is a problem that it operates with a fixed threshold value without considering a command or characteristic of a user. In addition, when a heterogeneous device is added and associated with existing work, there is a disadvantage that the server must be reconfigured.In order to solve the problems of the system which does not take into account the characteristics of the user in the IoT environment, this study collects the sensor data at the time of command when the control command of the user occurs. The threshold value to be operated by the corresponding device is analyzed and the threshold value is converted. In addition, it collects sensor data and device information of devices not registered inthe server, and works with existing registered devices.

4.1 Connection Diagram of System

The Circuit Diagram is the Graphical representation of the electrical Circuit. A pictorial circuit diagram uses simple images of

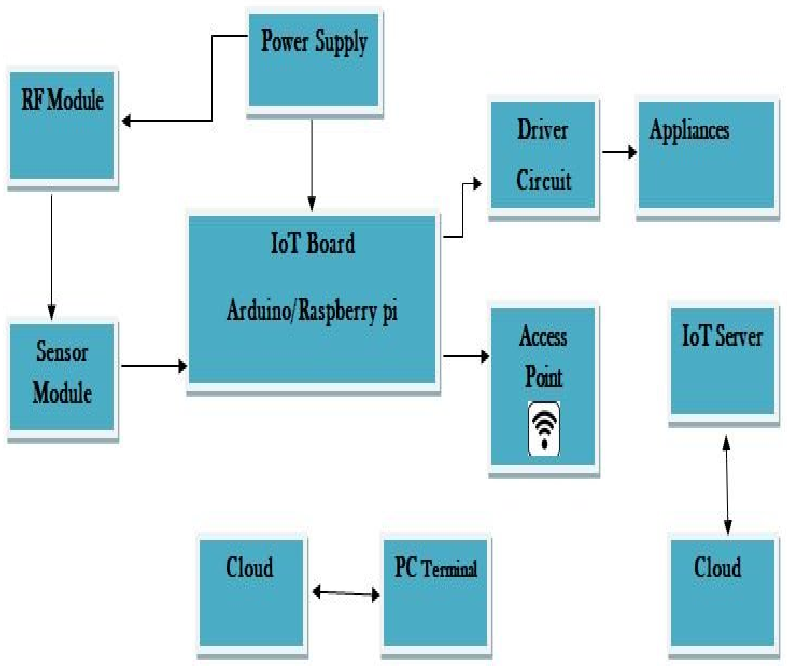

Figure 4.1: Connection Layout

components, while a schematic diagram shows the components and interconnections of the circuit using standardized symbolic representations. The presentation of the interconnections between circuit components in the schematic diagram does not necessarily correspond to the physical arrangements in the finished device. Unlike a block diagram or layout diagram, a circuit diagram shows the actual electrical connections. A drawing meant to depict the physical arrangement of the wires and the components they connect is called artwork or layout, physical design, or wiring diagram. Circuit diagrams are used for the design (circuit design), construction (such as PCB layout), and maintenance of electrical and electronic equipment.

4.2 Testing

Hardware testing is the process of evaluating and verifying that a hardware product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

4.2.1 Test Case

Testing is based on test cases. It describes which feature or service test attempts to cover. In test cases specify what you are testing and which particular feature it tests.

- Test the normal use of system
- Test the abnormal, but reasonable use of system
- Test the abnormal and reasonable use of system
- Test the boundary Conditions

4.2.2 Functionality Testing

an be performed on hardware or software products to verify that your product functions exactly as designed. The general purpose of hardware and software functionality testing is to verify if the product performs as expected and documented, typically in technical or functional specifications. Developers creating a new product start from a functional specification, which describes the product's capabilities and limitations.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

Recently IoT has been used in various fields. The smart device can be connected to the network and communication can be performed to operate the device or to provide the sensor data to the user. Accordingly, methods of providing services in various ways are being studied. Existing systems provide only the same service on the basis of the set threshold value, and operation of the device operation judgment is performed according to the environment inside the home rather than the user. Also, the user can not control the device.

To solve this problem, this paper collects and analyzes user 's remote control command. Based on this, we propose an automatic control system. When a user performs remote control using an application, the system collects the commands, loads them in the database, and sorts the data according to devices and time. Once the analysis is complete, device averaging is derived and the device is automatically controlled based on the analysis results. As a result, a customized service can be provided for the characteristics of the user. Also, device usage can be derived and provided to the user as a graph. Future research should be based on complex data based on subdivision of scope and contents of analysis.

The analysis learns thresholds and proceeds by device. Accordingly, the analysis is performed every time a control command of a user is generated, and a service considering characteristics of each user can be provided. Also, when heterogeneous devices not registered in the server occur, sensor data and device information are loaded in the database. Each time the device operation proceeds, it is linked with the device operation that is in the existing operation or in the standby state. The user can monitor and control the threshold, power consumption, sensor data, and device status through the application. We have verified that efficiency and accuracy are increased compared to existing systems. Through this, the threshold value of the device changes according to the characteristics of the user, and a new device can be learned and a service with increased convenience can be provided. Future research should be carried out by applying the proposed system to various environments.

5.2 Future Scope

As discussed customized automation is an needed in several areas. User command analysis system will be a great innovation and it would be easier for the user for assessing the cooling system . Personalized sercices can be prodvided in home itself.This can be used not only in house but also in factories, organization etc.In future it can be develop as an online cloud scheme with reduction in all commodities

Chapter 6

CODING

6.1 Arduino.ino

```
include <SoftwareSerial.h>
SoftwareSerial espSerial(9, 10);//(rx,tx)
Fan_PWM =6;
int Fan_IN3 =5;
int Fan_IN4 =4;
include <DHT.h>
define DHTPIN 7
define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE);
float Air_Temp;
float Air_Humidity;
include <OneWire.h>
include <DallasTemperature.h>
const int oneWireBus = 3;
OneWire oneWire(oneWireBus);
DallasTemperature sensors(oneWire);
```

```
float Body_Temp ;
int Fan_Speed;
int Mode_Data_Pin=2;
int Fan_Data_Pin=8;
ngo.shortcuts import render
void setup() {
Serial.begin(115200);
espSerial.begin(115200);
delay(100);
dht.begin();
delay(100);
sensors.begin();
pinMode(Mode_DataPin,INPUT);
pinMode(Fan_DataPin,INPUT);
pinMode(Fan_PWM,OUTPUT);
pinMode(Fan_IN3,OUTPUT);
pinMode(Fan_IN4,OUTPUT);
}
voidloop(){
Air_Temp = dht.readTemperature();
Air_Humidity = dht.readHumidity();
sensors.requestTemperatures();
Body_Temp = sensors.getTempCByIndex(0);
if(digitalRead(Mode_Data_Pin) == HIGH){
Serial.print("AUTOMATICMODE");
if(digitalRead(Fan_Datapin) == HIGH){
Fan_Speed = map(Body_Temp,25,40,0,255);
if(Fan_Speed < 40){
Fan_Speed = 40;
}
if(Fan_Speed > 255){
Fan_Speed = 255;
}
analogWrite(Fan_PWM,FanSpeed);
digitalWrite(Fan_IN3,LOW);
digitalWrite(Fan_IN4,HIGH);
}
if(digitalRead(Fan_Data_Pin) == LOW){
analogWrite(Fan_PWM,0);
digitalWrite(Fan_IN3,LOW);
digitalWrite(Fan_IN4,LOW);
}
}
if(digitalRead(Mode_Data_Pin) == LOW){
Serial.print(”REMOTEMODE”);
if(digitalRead(Fan_DataPin) == HIGH){
Serial.print(”FANON : 255”);
analogWrite(Fan_PWM,255);
digitalWrite(Fan_IN3,LOW);
digitalWrite(Fan_IN4,HIGH);
}
if(digitalRead(Fan_Data_Pin) == LOW){
Serial.print(”FANOFF : 0”);
analogWrite(Fan_PWM,0);
digitalWrite(Fan_IN3,LOW);
digitalWrite(Fan_IN4,LOW);
}
}
Serial.println(””);
Print_Data();
```

```
Data_Esp();
delay(1000);
}
voidPrint_Data(){
Serial.print("Air_Temp : ");
Serial.print(Air_Temp);
Serial.print("Air_Humidity : ");
Serial.print(Air_Humidity);
Serial.print("Fan Speed PWM : ");
Serial.print(Fan_Speed);
Serial.print("DS18B20Temp : ");
Serial.println(Body_Temp);
}
voidData_Esp(){
Serial.print("DataTOESP : ");
Serial.println(String(Air_Temp) + "," + String(Air_Humidity) +
"," + String(Fan_Speed + "," + String(Body_Temp));
espSerial.println(String(Air_Temp) + "," + String(Air_Humidity) +
"," + String(Fan_Speed) + "," + String(Body_Temp));
}
```

6.2 node.ion

```
define BLYNK_PRINT Serial
include <ESP8266WiFi.h>
include <BlynkSimpleEsp8266.h>
define
BLYNK_AUTH_TOKEN"7RfrLAB9M3if74_SPHe9nOn6J7vx4RS"
char auth [ ] = BLY NK_AUT H_T OKEN;
charssid [ ] = "SUNDAY";//W IFIName
charpass[ ] = "123456789";//W IFIPassword
include < NT PClient.h >
include < WiFiUdp.h >
WiFiUDPnt PUDP;
NT PClienttimeClient(ntpUDP,"in.pool.ntp.org",19800,60000);
int time_blynk;
int HH,MM, final_time;
int Timer_status;
int Timer_Activate;
include < ESP8266HT T PClient.h >
include < WiFiClient.h >
String apiKeyValue = "tPmAT5Ab3 j7F9";
String sensorName = "";
String sensorLocation = "EKM";
de fine ON_Board_LED2//BlueLED
String myString;
String Air_Temp;
String Air_Hum;
String Body_Temp;
String Fan_Speed;
f loat Air_Temp_Blynk;
f loat Air_Hum_Blynk;
```

```
int Mode_pin=5;//D1
int Fan_pin=4;//D2
int Data;
int Fan_Data;
int Mode_data;
signed long start;
void setup() {
Serial.begin(115200);
Blynk.begin(auth, ssid, pass,”blynk.cloud”,80);
timeClient.begin();
pinMode(ONBoard_LED,OUTPUT);
WiFi.begin(ssid, pass);
Serial.println(”Connecting”);
while(WiFi.status()! = WL_CONNECTED)
{
delay(500);
Serial.print(”.”);
digitalWrite(ON_Board_LED, LOW);
delay(250);
digitalWrite(ON_Board_LED, HIGH);
delay(250);
}
Serial.println(””);
Serial.print(”Connected to WiFi network with IP Address: ”);
Serial.println(WiFi.localIP());
digitalWrite(ON_Board_LED, LOW);
pinMode(Mode_pin,OUTPUT);
pinMode(Fan_pin,OUTPUT);
}
BLYNK_WRITE(V6)
```

```
{
time_blynk = param.asInt();
}
BLYNK_WRITE(V0)
{
Data = param.asInt();
if(Data == 1)
{
Fan_Data=1;
}
if(Data == 0){
Fan_Data=0;
}
}
BLYNK_WRITE(V1) {
Data = param.asInt();
if(Data == 1)
{
Mode_data=1; //Remote control
}
if(Data == 0){
Mode_data=0; //Automatic mode
}
}
BLYNK_WRITE(V9)
{
Data = param.asInt();
if(Data == 1)
{
Timer_Activate=1; //Auto mode without Timing fuction
}
if(Data == 0){
Timer_Activate=0; ////Auto mode with Timing fuction
}
}
void loop() {
Blynk.run();
if(Mode_data==1){ ///////////////AUTOMATIC MODE
digitalWrite(Mode_pin,HIGH);
Fan_Data=0;
if(Timer_status==1)///////////With Timer
digitalWrite(Fan_pin,HIGH);
Blynk.virtualWrite(V7,1);
Blynk.virtualWrite(V0,1);
Blynk.virtualWrite(V8,Fan_Speed_Blynk);
}
if(Timer_Activate==1){///////////Without Timer
digitalWrite(Fan_pin,HIGH);
Blynk.virtualWrite(V0,1);
Blynk.virtualWrite(V8,Fan_Speed_Blynk);
}
if(Timer_status==0  Timer_Activate==0){
digitalWrite(Fan_pin,LOW);
Blynk.virtualWrite(V7,0);
Blynk.virtualWrite(V0,0);
Blynk.virtualWrite(V8,0);
}
}
if(Mode_data==0){
Blynk.virtualWrite(V9,0);
```

```
Timer_Activate=0;
digitalWrite(Mode_pin,LOW);
if(Fan_Data==1){
digitalWrite(Fan_pin,HIGH);
Blynk.virtualWrite(V8,255);
Blynk.virtualWrite(V0,1);
}
if(Fan_Data==0)
digitalWrite(Fanpin,LOW);
Blynk.virtualWrite(V8,0);
Blynk.virtualWrite(V0,0);
}
}
timeClient.update();
HH = timeClient.getHours();
MM = timeClient.getMinutes();
finaltime = 3600∗HH+60∗MM;
if(time_blynk == final_time){
Serial.println(”.............TIMERACTIVE.............”);
Timer_status = 1;
}
if(time_blynk ! = final_time){
Timer_status = 0;
}
if( Serial.available()¿0)
{
myString = Serial.readStringUntil(”);
Serial.print(”Data from Arduino: ”);
Serial.println(myString);
Air_Temp = getValue(myString, ’,’, 0);
```

```
Air_Hum = getValue(myString, ',', 1);
Fan_Speed = getValue(myString, ',', 2);
Body_Temp = getValue(myString, ',', 3);
Air_Temp Blynk = Air_Temp.toFloat();
Air_Hum Blynk = Air_Hum.toFloat();
Fan_Speed Blynk = Fan_Speed.toInt();
Body_Temp Blynk = Body_Temp.toFloat();
Data_to_Blynk();
}
if( Serial.available()==0)
{
Print_data1();
Print_data2();
}
if( millis() - start 5000 ){
Upload_data();
start = millis();
}
}
void Print_data1()
Serial.print("Real Time HH: ");
Serial.print(HH);
Serial.print(" MM: ");
Serial.print(MM);
Serial.print(" Final Time: ");
Serial.print(final_time);
Serial.print(" Time Frm Blynk: ");
Serial.println(time_blynk);
}
void Print_data2 (){
```

```
Serial.print(”Air_Temp:”);
Serial.print(Air_Temp_Blynk);
Serial.print(” Air_Hum:”);
Serial.print(Air_Hum_Blynk);
Serial.print(” Fan_Speed:”);
Serial.print(Fan_Speed_Blynk);
Serial.print(” Body_Temp:”);
Serial.println(Body_Temp_Blynk);
}
void Data_to_Blynk(){
Blynk.virtualWrite(V2,Body_TempBlynk);
Blynk.virtualWrite(V3,Air_TempBlynk);
Blynk.virtualWrite(V4,Air_HumBlynk);
}
void Upload_data(){
if(WiFi.status()== WL_CONNECTED )
{
digitalWrite(ON_Board_LED, LOW);
HTTPClient http; .begin(”http://192.168.36.125/sensordata/post-esp-data.php”);
http.addHeader(”Content-Type”, ”application/x-www-form-urlencoded”);
String httpRequestData = ”api_key=” + apiKeyValue + ”sensor=” +
sensorName+ ”location=” + sensorLocation + ”value1=” + Air_Temp +
”value2=” + Air_Hum + ”value3=” + Body_Temp + ””;
Serial.print(”Upload_Data: ”);
Serial.println(httpRequestData);
int  httpResponseCode = http.POST(httpRequestData);
Serial.print(”Upload_ResponseCode: ”);
Serial.println(httpResponseCode);
delay(1000);
}
```

```
else {
Serial.println(”WiFi Disconnected”);
digitalWrite(ON_Board_LED, HIGH);
}
}
String getValue(String data, char separator, int index)
{
int found = 0;
int strIndex[] = 0, −1;
int maxIndex = data.length() − 1;
    for (int i = 0; i ¡= maxIndex  found ¡= index; i++) {
    if (data.charAt(i) == separator —— i == maxIndex) {
    found++;
    strIndex[0] = strIndex[1] + 1;
    strIndex[1] = (i == maxIndex) ? i + 1 : i;
    }
    }
    return found  > index  ?data . substring( strIndex[0] ,strIndex[1]) : ””;
    }
```

Chapter 7

SCREEN SHOTS

The screenshots include:

- IoT based user command analysis
- Working Mode
- Remote Mode in UI
- Automatic Mode in UI

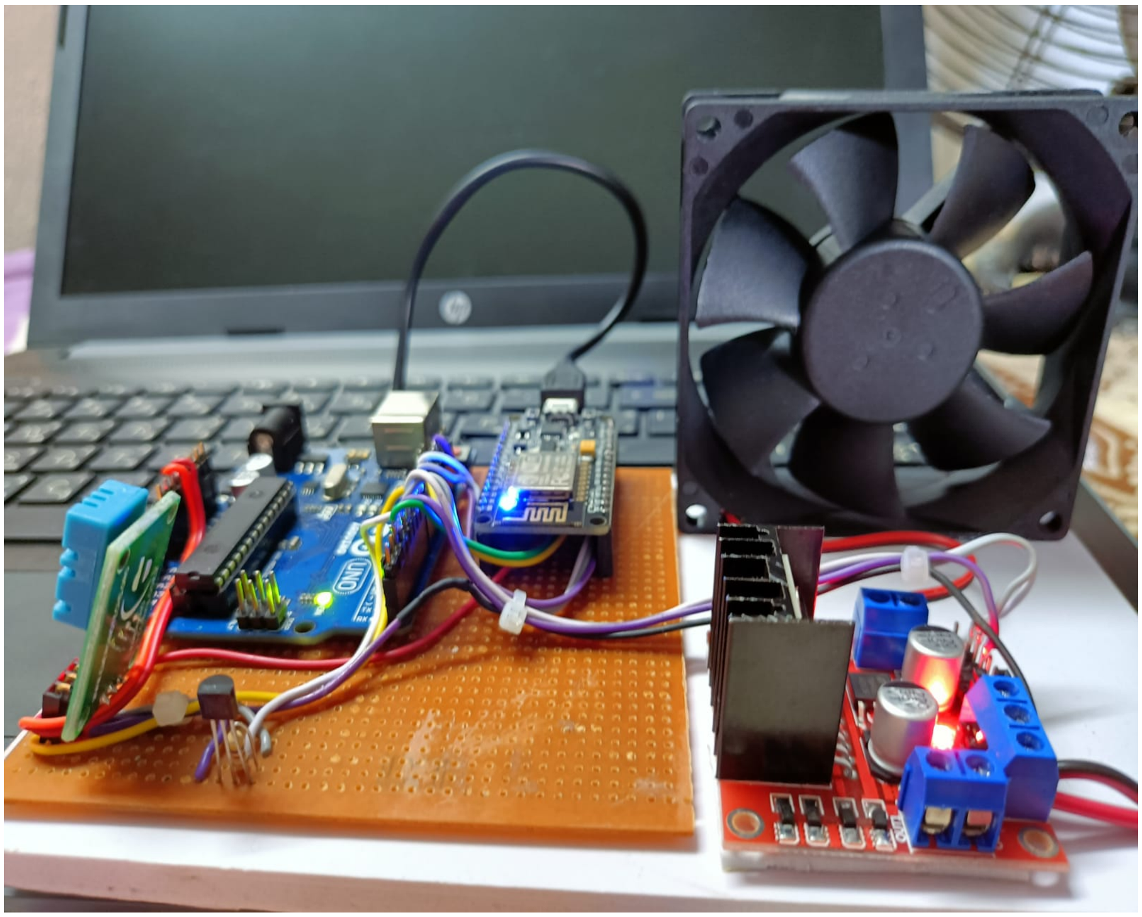

Figure 7.1: IoT based user command analysis

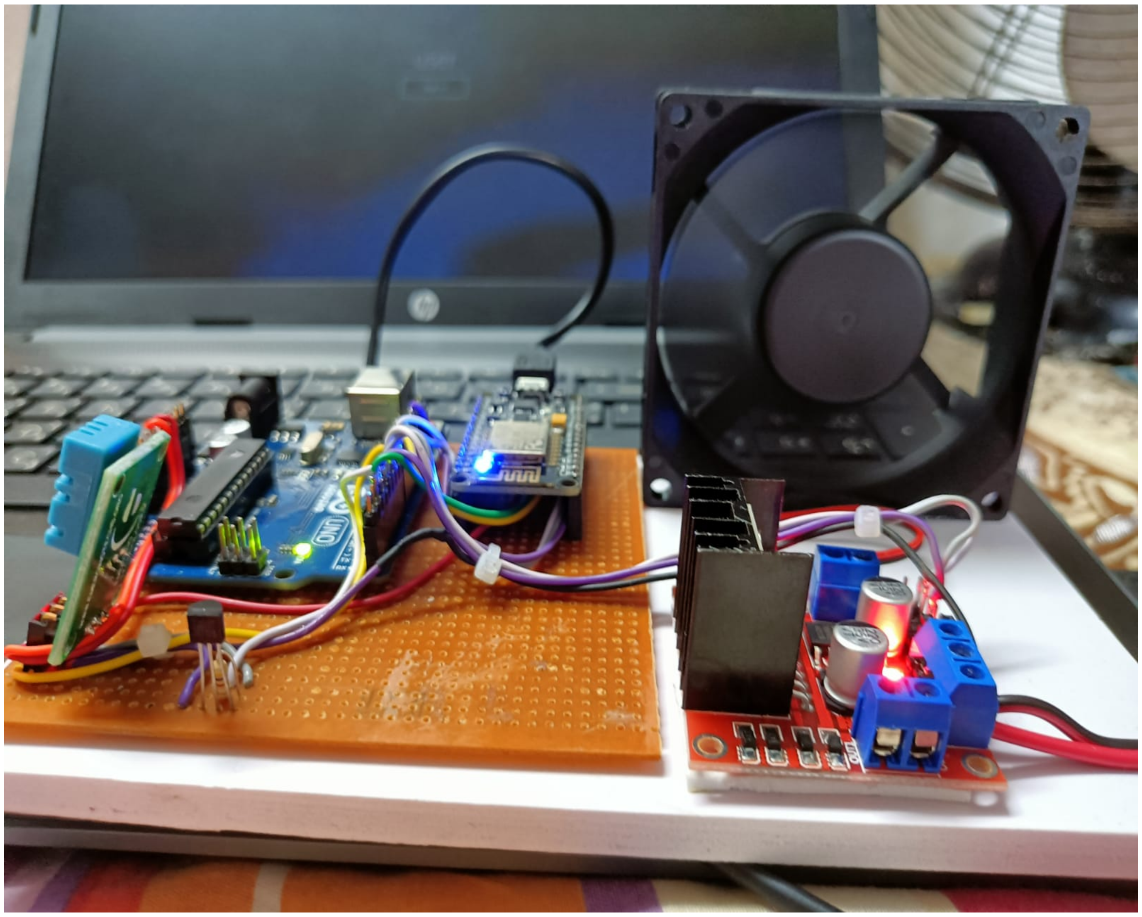

Figure 7.2: Working Mode

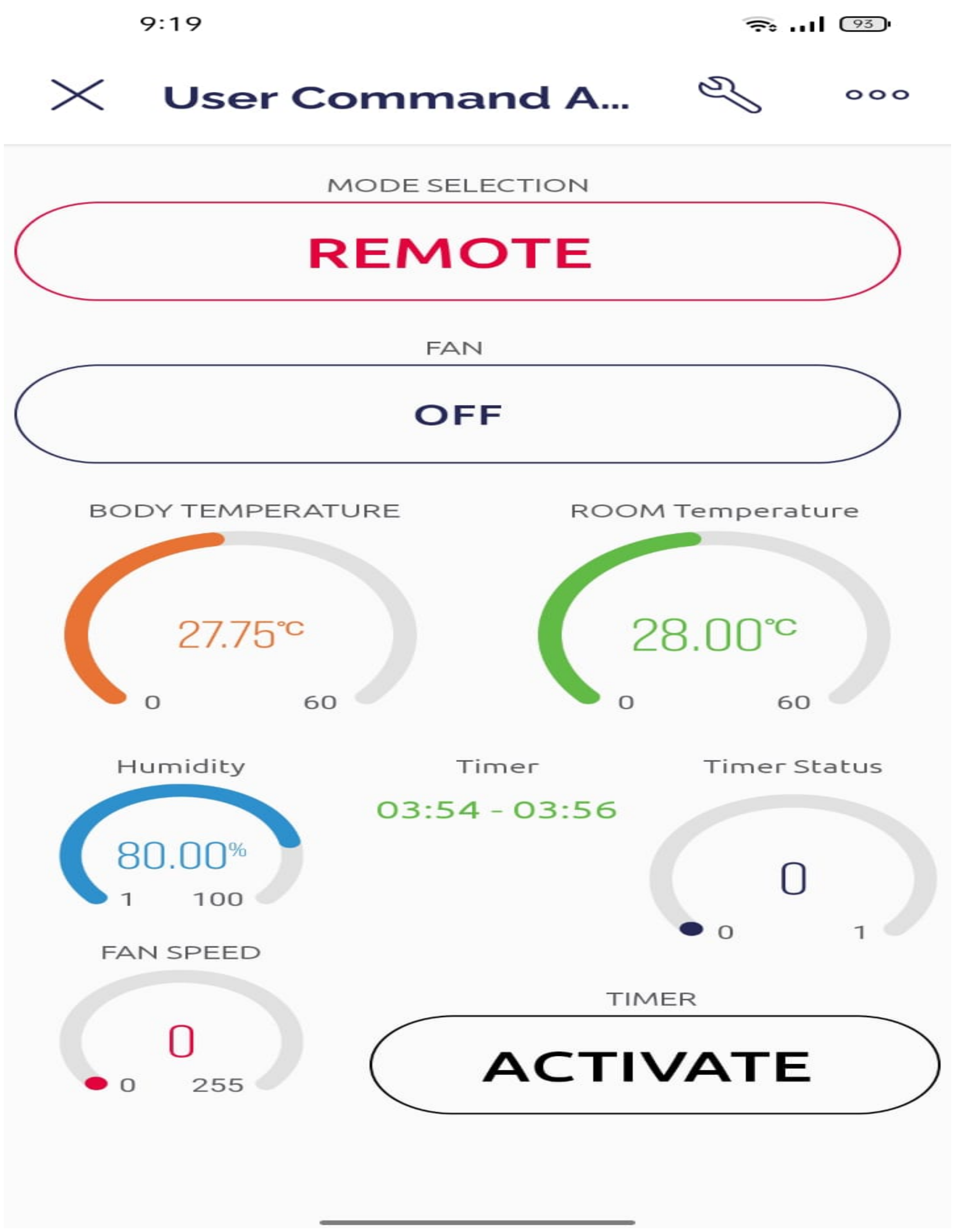

Figure 7.3: Remote Mode in UI

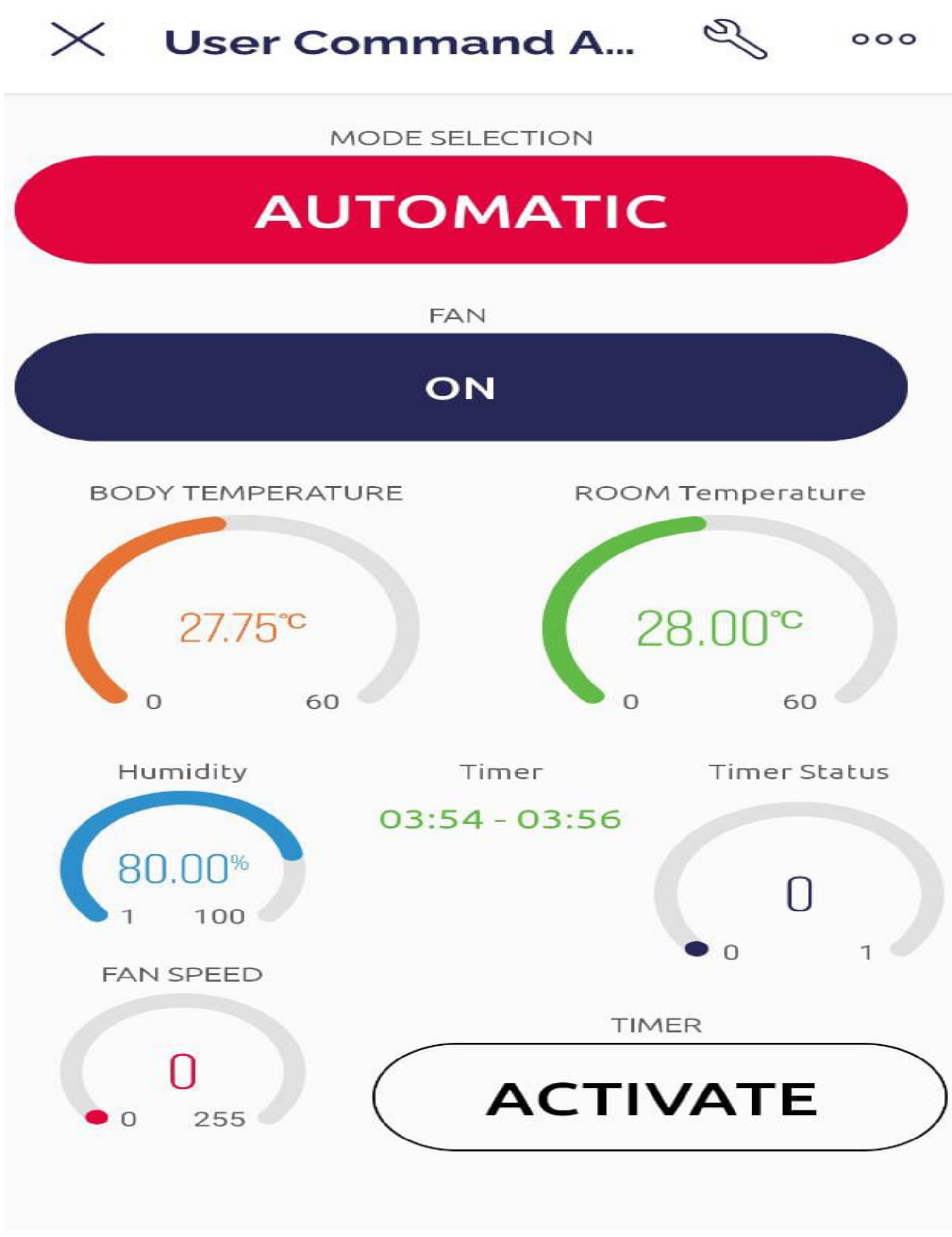

Figure 7.4: Automatic Mode in UI

Chapter 8

REFERENCES

1. https://ieeexplore.ieee.org/document/9213796
2. D. W. Choi, H. J. Cho. Machine learning based motion recognition technology for intelligent IoT service. Journal of the Korea Electromagnetic Engineering Society.2016; 27(4); 19-28
3. K. Manickavasagam. Automatic Generation Control of MultiArea Power System with Generating Rate Constraints Using Computational Intelligence Techniques. International Journal of Applied Power Engineering. 2013; 2(1): 27-38.
4. Lakafosis, A. Traille, L. Hoseon, E. Gebara, M. M. Tentzeris, G. R. DeJean, and D.Kirovski, ”RF Fingerprinting Physical Objects for Anticounterfeiting Applications,” Microwave Theory and Techniques, IEEE Transactions on, vol. 59, pp. 504-514, 2011
5. W. Y. Lee, H. M. Ko, J. H. Yu, K. B. Sim. An Implementation of Smart Dormitory System Based on Internet of Things. Journal of Korean Institute of Intelligent Systems. 2016; 26(4): 295-300.