Palestine Polytechnic University

College of Information Technology and Computer Engineering

Department of Computer System Engineering

Title:

IoT Smart Home Controllers

Team names:

Sojood Othman Sweiti.

Samah Yacoub Fannoun.

Supervisor

Dr. Radwan Tahboub.

إهداء

إلى أمي الحبيبة منبع المحبة والحنان، إلى من بذل الغالي والنفيس لأكون كما أنا عليه إلى والدي العزيز.

إلى من ذقت معهم طعم المحبة والأخوة إلى أخوتي وأخواتي وأسرتي.

إلى كل من علمني حرفا معلمينا العلماء.

إلى من ضاقت السطور ذكرهم، إلى من كانوا قوت قلوبنا إلى أحبائنا وأصدقائنا الأعزاء.

إلى من كان سندا لنا بجهده ووقته ونصحه وعلمه إلى مشرفنا القدير الدكتور رضوان طهبوب.

وإلى أرواح شهدائنا الأبرار نهدي هذا العمل.

Acknowledgment

A part of our effort and the successes of any project depends largely on the encouragements of others.

We would like to start with a special gratitude to our university for the help and support provided in its facilities for the past five years, and for our computer engineering college for the large effort had been spent during our study to reinforce us with all knowledge needed.

And no thanks could be enough for our supervisor Dr. Radwan Tahboub, for his support, effort and guidelines he gave us during the course, special thanks to Eng. Wael Takroury for his support and help.

Abstract

Smart home is a term that is commonly used to refer to homes where the appliances, light system, air-conditioning, TVs etc. are capable of communicating with each other and can be controlled remotely according to a predefined schedule or via some kind of interface.

This project presents a design and implements a smart controller compatible with IoT technology and standards to control home devices remotely such as (light system, water motor and curtain motor, etc.), connect the controller with smartphone via Ubidots standard application in IOS and Android compatible with IFTTT and Google Assistant. This controller can deal with multiple sensors and take one or more actions.

The proposed system consists of two main components; the first part is the smart controller which can communicate with the server (Ubidots MQTT broker), which presents a system core that manages, controls, and monitors users home.

Second part is the hardware interface module (application), which provides an appropriate interface to sensors and actuators of home systems. The application is used to communicate with the Ubidots database and update its values, this in turn enables us to control the various sensors and electrical appliances in the home.

Unlike most of the available smart home systems in the market, the proposed system is scalable and one server can manage many hardware devices as long as it exists on WiFi network coverage.

The proposed system is better from the scalability and flexibility point of view than the commercially available home automation systems.

Table of Contents

Abbreviations table

Abbreviation	Meaning
IoT	**Internet of Things**
ESP	**Espressif Systems Programmable**
IFTTT	**If This Then That**
MQTT	**Message Queuing Telemetry Transport**
DIY	**Do It Yourself**
IOS	**iPhone operating system**
IDE	**Integrated Development Environment**

1

Chapter one:

Introduction

1.1Overview

1.2 Motivation and importance

1.3 Objectives

1.4 Short description of the system

1.5 Problem statement

1.6 List of requirements

1.7 Expected results

Introduction

1.1 Overview

Internet of Things (IoT) can be described as connecting everyday objects like smart phones, internet televisions, sensors and actuators to the internet where the devices are intelligently linked together to enable new forms of communication amongst people.

A smart home is a home that uses internet-based devices to enable home monitoring, remote control of devices and systems such as lighting and heating. Smart home technology provides homeowners with security, comfort and energy savings by allowing them to control home appliances and control how they work.

In this project we designed and implemented a smart controller compatible with IoT technology and standards to control home devices remotely such as (light system, water motor and curtain motor, etc.), connect the controller with smartphone via Ubidots standard application available in IOS and Android compatible with IFTTT standard to create triggers and MQTT protocol to connect to the cloud and transfer data, Amazon Alexa and Google Assistant as a control way remotely.

This smart controller enables users to DIY it with devices they want, the user can connect the smart controller with anything at home.

1.2 Motivation and importance

Technology aims to make people's lives easier. When a person is outside his home, the technologies like IoT based devices will show the status of the home appliances by application on smart phone, it's easy to download that deals with smart home at an inexpensive price. Being connect to such smart controllers will let people control these devices remotely and make efficient use of time.

Existing devices and student’s projects that use microcontrollers are not making use of the international state of the art standards and protocols in the IoT era.

1.3 Objectives

- Implement a general purpose controller that can control a wide range of devices over the Internet.
- Implement a general purpose controller that can track the status of home appliances with smart phones.
- Integrated IoT state of the art standards, protocols, and application into the designed controller.
- Finishing our prototype in a professional way so it would be ready for marketing.

1.4 Short description of the system

The general-purpose smart controller is to provide the user comfort and luxury that contemporary people will enjoy in the near future based on smartphone applications, tablets and some protocols and standards. The concept of a smart home that we have heard in recent years is a home completely controlled by technology for the purpose of comfort, safety and smoothness of the owner of the home, is managed and controlled by a range of buttons, remote controls and sensors, the home is fully controlled and monitored through the Internet, this is what achieves the Internet of Things concept, through a certain application the individual can turn on/off and control various electrical devices before and after back home.

1.5 Problem statement

Smart Home is a new trend that became more important day by day because of the increase in importance of technology, and the families with disability members and busy parents need such systems, so in near future Smart Home will be a must. However, current available industry is expensive and inflexible because each manufacturer focuses on certain electrical appliances over others to control, and most of these systems depend on replacing the traditional electrical units with smart ones which increase the total system cost and make these systems hardly implemented in our market. So our project will solve this problem by implementing the general purpose smart controller that can connect with any type of electrical units and devices.

Our project is meant to provide a complete and finished solution prototype, this prototype intends to be cheap, flexible, scalable, and practical so it won't replace any traditional or old electrical units in the home.

1.6 Project requirements

The following subsections have the list of requirements of the project.

1.6.1 Functional requirements:

- The smart controller has been designed to suit various electrical powers.
- The smart controller is able to control the largest possible number of household appliances.
- The smart controller can control the devices by Ubidots mobile application and manual switches.
- The smart controller can control the devices by voice using Google Assistant.

1.6.2 Non-Functional requirements:

- Simplicity: The system is simple for any one to use, whether they are expert in technology or not.
- Ease of access: the access of the system will be ease from mobile application which is ease to use.

1.7 Expected results

The results which we achieved are the following:

- The goal of controlling the app on the smartphone and the voice control of the home appliances has been achieved.
- The system can deal with many events and triggers that make it smartest, such as water motor event which is turn off after a certain time of operation.
- The user can use DIY technique to assemble the controller with home appliances.

2

Chapter two:

Background

2.1Theoretical background and literature review

2.2 Description of the parts used in the system

2.3 Specification and design constrains

Background

2.1 Theoretical background and literature review

The Internet of Things (IoT):

The Internet of Things (IoT), also called the Internet of Everything or the Industrial Internet, “is a new technology paradigm envisioned as a global network of machines and devices capable of interacting with each other. The IoT is recognized as one of the most important areas of future technology and is gaining vast attention from a wide range of industries. The true value of the IoT for enterprises can be fully realized when connected devices are able to communicate with each other and integrate with vendor-managed inventory systems, customer support systems, business intelligence applications, and business analytics” [1].

Information sharing and collaboration in the IoT can occur between people, between people and things, and between things. Sensing a predefined event is usually the first step for information sharing and collaboration.

Cloud computing: is a model for on-demand access to a shared pool of configurable resources (e.g., computers, networks, servers, storage, applications, services, software) that can be provisioned as Infrastructure as a Service (IaaS) or Software as a Service (SaaS). “One of the most important outcomes of the IoT is an enormous amount of data generated from devices connected to the Internet (Gubbi et al., 2013). Many IoT applications require massive data storage, huge processing speed to enable real time decision making, and high-speed broadband networks to stream data, audio, or video. Cloud computing provides an ideal back-end solution for handling huge data streams and processing them for the unprecedented number of IoT devices and humans in real time” [1].

RTOS: “A real time operating system is the type of system which uses maximum time and resources to output exact and on the time result. There is no difference between the results when same problem run on different occasion on same machine. There is no late or early execution on that operating system and is done on fixed time as suggested.” [2]

Free RTOS is a real-time operating system kernel for embedded devices that has been ported to 40 microcontroller platforms, one of these controllers is ESP32.

With the continuous growth of smart devices in our lives, the demand for advanced mobile applications increases everywhere in our daily life.

The use of web services and applications is the most common and interoperable with one another. For all people and especially people with physical limitations, in this case they need home automation.

IoTs can be described as connecting everyday objects like smart phones, internet televisions, sensors and actuators to the internet where the devices are intelligently linked together to enable new forms of communication amongst people and themselves.

"The significant advancement of IoTs over the last few years has created a new dimension to the world of information and communication technologies. The advancement is leading to anyone, anytime, anywhere (AAA) connectivity for things with the expectation being that this will extend and create an entirely advanced dynamic network of IoTs. The IoTs technology can be used for creating new concepts and wide development space for smart homes in order to provide intelligence, comfort and improved quality of life." [3]

In recent years, wireless systems like WLAN 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.

This study presents a design and prototype implementation of a new home automation system that uses WiFi technology as a network infrastructure connecting its parts. The proposed system consists of two main components; the first part is the server (web server), which presents a system core that manages, controls, and monitors users' home. Users and system administrators can locally (LAN) or remotely (internet) manage and control system code. Second part is hardware interface module, which provides appropriate interface to sensors and actuator of home automation system. [4]

Previous graduation project:

Home Automation Based on Raspberry Pi Single Board Computer

This project is one of the applications on the home automation system.

“The main idea of this project is to design and implement home automation application using Raspberry Pi and based on python environment. A special sensor is used to sense body motion and send a signal to Raspberry Pi if there is someone in front of the home. Consequently, the Camera captures an image for the visitor then starts recording a video. The images and videos are saved on the server. The image is sent to the homeowner by Wi-Fi. Homeowner can send feedback to the visitor as a voice message. Homeowner and police can connect by the Q-python application on a homeowner smartphone. If the visitor gets out of the range of sensor, the camera will stop recording video.” [5]

The difference between our project and previous project:

- The previous project is a home automation system the user does not make a decision, as for our project, it is a smart control system the user makes a decision and has the privileges to control the devices.
- The previous project only controlled intelligent door control Our project is more general and controls different types of devices and a wide number of them.

2.2 Description of the parts used in the system

2.2.1 Electronic parts that will use in project

The main piece to design the system is ESP32 microcontroller:

This chip was chosen because has multi type of pins, the analog and digital pins, the hardware security, the ease of programming and dealing with it, the presence of large memory, greater speed in data transfer and Internet connection compared to its peers.

And another important piece is the relay that will be the medium between the microcontroller and high-voltage devices.

2.2.2 The standards and protocols that used in project

- Ubidots platform: It has two parts one for hardware engineering and software engineering, It deals with a large number of devices, one of which is the Espressif Systems, one of the most important chips in it is ESP32 integrated with MQTT protocol, Ubidots able to interact with us through at least one of these protocols: HTTP, MQTT, TCP/UDP.

  In our project use MQTT protocol, this platform has enabled us to control the devices connected to the microcontroller by some widgets and triggers.

- IFTTT is an automation tool that lets easily script actions that link together a wide variety of devices and services. IFTTT is short for "If This Then That," a programming convention that defines in a nutshell how the service works. The "if this" part is known as the trigger, and "then that" is the action. The sum is the applet.

  - **How IFTTT works**

    IFTTT is available as a website and mobile app for both iOS and Android devices, and any automations we create are available on all three platforms. “It's compatible with more than 650 brands and services – the list is exhaustive, but includes products such as Gmail, Dropbox, Slack, Uber, GE smart appliances, Pinterest, Twitter, and Withing’s.

    By connecting these services with If This Then That statements, create Applets, which are the automations that run in the background.” [7].

2.3 Specification and design constraints and alternatives

2.3.1 ESP32 Specification

"Processors: Main processor: 32-bit microprocessor, Cores: 2 or 1 (depending on variation) all chips in the ESP32 series are dual-core except for ESP32-S0WD, which is single-core, Clock frequency: up to 240 MHz, Performance: up to 600 DMIPS.

Wireless connectivity: Wi-Fi: 802.11 b/g/n/e/i (802.11n @ 2.4 GHz up to 150 Mbit/s)
Bluetooth: v4.2 BR/EDR and Bluetooth Low Energy (BLE)

Memory: Internal memory: ROM: 448 KB/ SRAM: 520 KB for data and instruction.
Embedded flash: Flash connected internally via IO16, IO17, SD_CMD, SD_CLK, SD_DATA_0 and SD_DATA_1 on ESP32-D2WD and ESP32-PICO-D4.
External flash & SRAM: ESP32 supports up to four 16 MB external QSPI flashes and SRAMs with hardware encryption based on AES to protect developers' programs and data. ESP32 can access the external QSPI flash and SRAM through high-speed caches.
Up to 16 MB of external flash are memory-mapped onto the CPU code space, supporting 8-bit, 16-bit and 32-bit access.
Code execution is supported, Up to 8 MB of external flash/SRAM memory are mapped onto the CPU data space, supporting 8-bit, 16-bit and 32-bit access. Data-read is supported on the flash and SRAM. Data-write is supported on the SRAM.

Peripheral input/output: Rich peripheral interface with DMA that includes capacitive touch, ADCs (analog-to-digital converter), DACs (digital-to-analog converter), I²C (Inter-Integrated Circuit), UART (universal asynchronous receiver/transmitter), CAN 2.0 (Controller Area Network), SPI (Serial Peripheral Interface), I²S (Integrated Inter-IC Sound), RMII (Reduced Media-Independent Interface), PWM (pulse width modulation), and more.

Security: IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI, Secure boot**,** Flash encryption, 1024-bit OTP, up to 768-bit for customers.
Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG)" [8].

3

Chapter three:

Design

3.1 Detailed conceptual description of the system

3.2 Block diagrams

3.3 Schematic diagram

Design

3.1 Detailed conceptual description of the system

Our project begins that home appliances controlled remotely using a smartphone, where the turn on or turn off command is sent from the phone to the cloud via the MQTT communication protocol connected to the Ubidots platform. The -parts in the platform (widget) are connected directly via the Internet to the hardware parts such as the relay. Which enables us to control these devices connected to the relay where the relay works as a switch when an operating order comes, it moves to operation and non-operation in the event of a non-operation order and the relay is one of the most important parts in this project because it is the mediator between the microcontroller and the high current devices as it takes as an input of 5 volts and it connects with 220-volt devices.

This project relied on global standards in sending data and triggers, as we used the IFTTT system to build these triggers to deliver voice such as Google Assistant, send an email and other triggers.

The most important parts in our project

- ESP32: is a microcontroller and it is the primary part that is used to load our project code on it and connected with other hardware components relays, sensors, LEDs...etc...

Figure 1: ESP32.

- Relays: The relay module is an electrically operated switch that allows us 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.

Figure 2: Relay module.

- Temperature sensor: LM35 is a temperature measuring device having an analog output voltage /proportional to the temperature. It provides output voltage in Centigrade (Celsius). It does not require any external calibration circuitry. The temperature sensor in our project measures room temperature every period of time and gives the user a notification on his smartphone.

Figure 3: Temperature sensor.

- Logic Level Converter: it is integrated circuit that convert the voltage from 3.3 volt to 5 volts, connected as intermediary with ESP32 and relay module.

Figure 4: Logic Level Converter.

- Ubidots platform: This platform we used to create a dashboard that the user will deal with it directly to facilitate control of the system through some widgets represented by virtual switches that show on app in the smartphone screen, a scale that shows the temperature in the room and other forms of widgets that we find in the Ubidots platform, this platform deals with standards such as IFTTT to create triggers enables user to control by voice, such as Google Assistant, send notifications and email to the user.

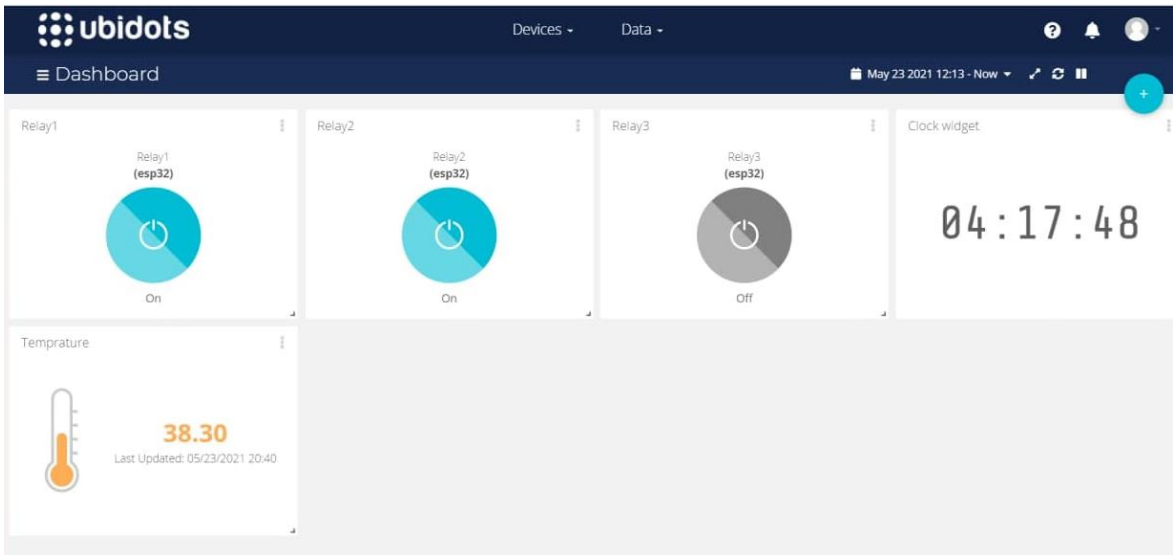


Figure 5: Ubidots Dashboard.

- IFTTT standard platform: is a free online service that integrates your favorite products and online services together to establish conditions among apps and other devices. We used this platform in our project, to create Applets to enable us to handle the controller via standards according to the device that will be connected to the controller.

3.2 Block diagrams

This block diagram is a description of what the project is doing, as the command starts from the application loaded on the smartphone and sends the data to the cloud, from which it connects to the platform and through the platform it connects to the microcontroller and creates an order to connect or not to the relay.

We note in the diagram the presence of the sensors, the function of the sensors, reading the temperature in the room, sending it to the platform and its appearance on the smartphone, and based on the reading, the IFTTT trigger is created by sending an email to the user alerting the status.

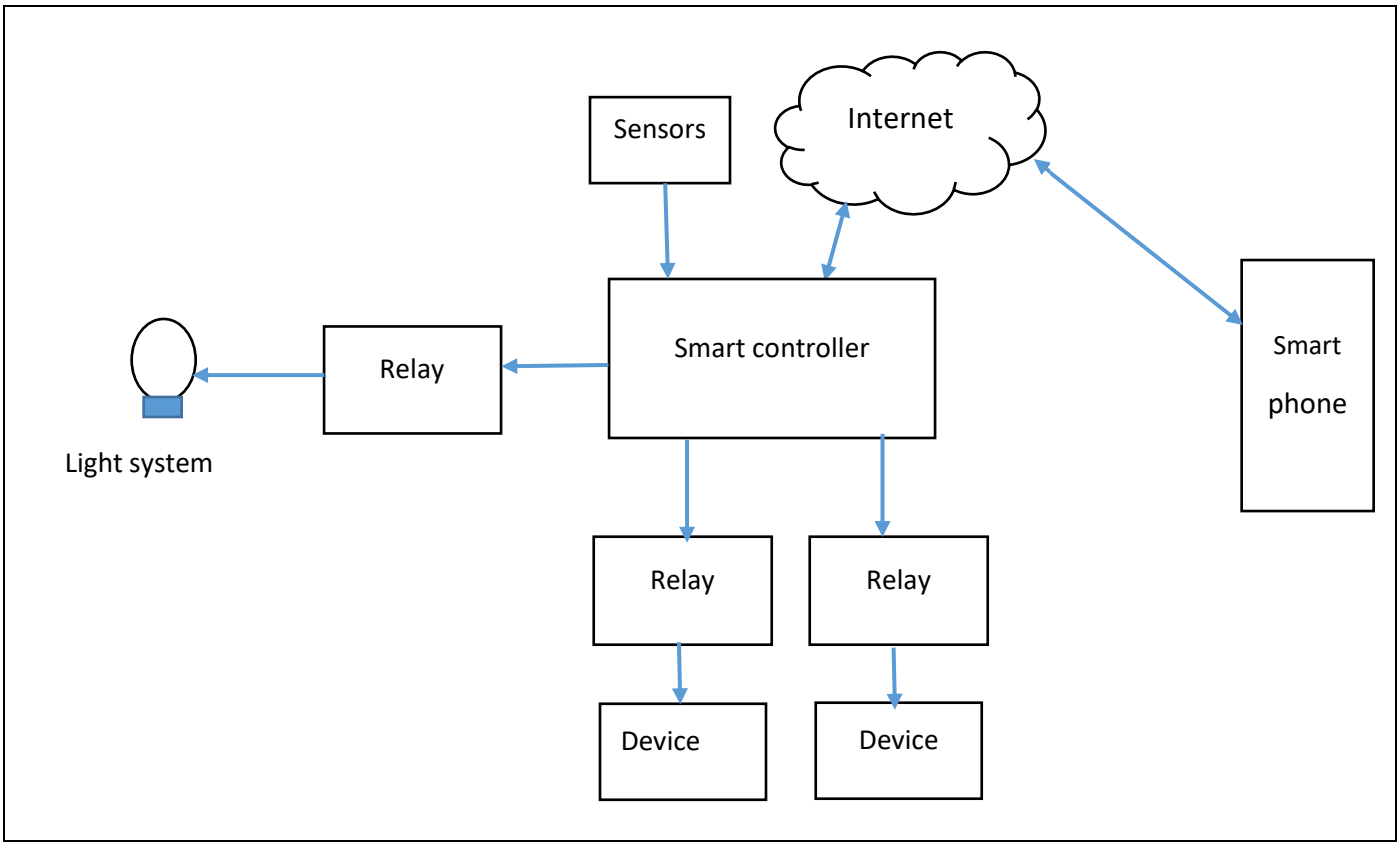


Figure 6: Block diagram

3.3 Schematic diagram

We connected the relays to the digital pins 25, 26 and 27, respectively, and in proportion to the voltage difference between them and the smart controller. We connected the Logical Level Convertor. We connected the sensor to the analog pin 35 to read the temperature. We connected the LED on digital pin 5 to show us the state of the connection in the Internet. If there is no connection, the LED blinks for a period of time every half a second. If it connects to the internet, it lights up.

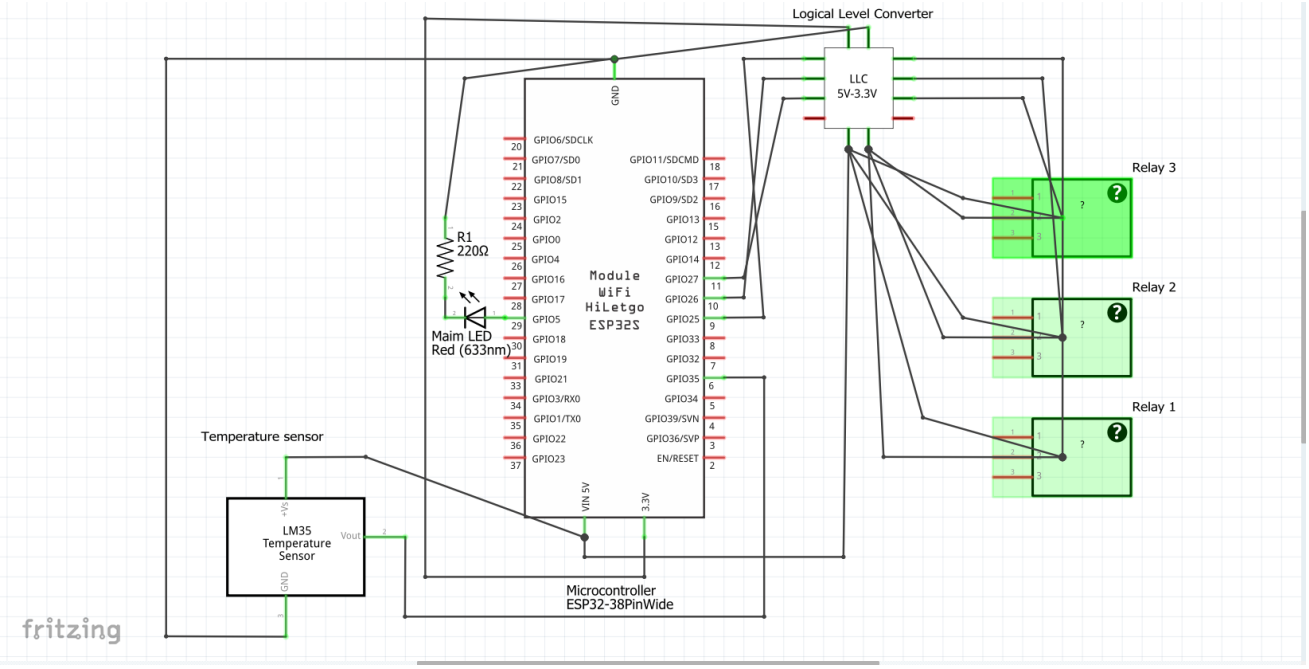


Figure 7: Schematic diagram

4

Chapter four:

Software

4.1 Description of the software

4.2 Flow chart

4.3 Pseudocode

Software

4.1 Description of the software

The system depends on two active points, smartphone application in the user hand, and the ESP32 microcontroller connected with devices. The two points communicate with each other using WIFI.

In communication between the user with an application and the microcontroller there are two directions, one is that when the user sends the controlling command for the microcontroller, it can turn on or off some device. The other direction is that when the microcontroller sends a configuration command which informs that the action has been done.

The developer implements the dashboard on web Ubidots platform, then the user opens the Ubidots application on his smartphone, login to the application so he can further access the system and control the devices, or he can control the devices by web dashboard on the platform.

When the user chooses the switch button on the dashboard, it sends a command to the ESP32 to open/close the relay that is connected with the device.

An appropriate sensor which is in our case a temperature sensor (LM35), it has widgets on the dashboard that display temperature degrees and it is linked with Ubidots platform events. It senses the temperature and based on the current value, if the temperature increases the threshold value that sends an email on Google Gmail that temperature is high in the room. The user will decide if he wants to turn on the fan or not.

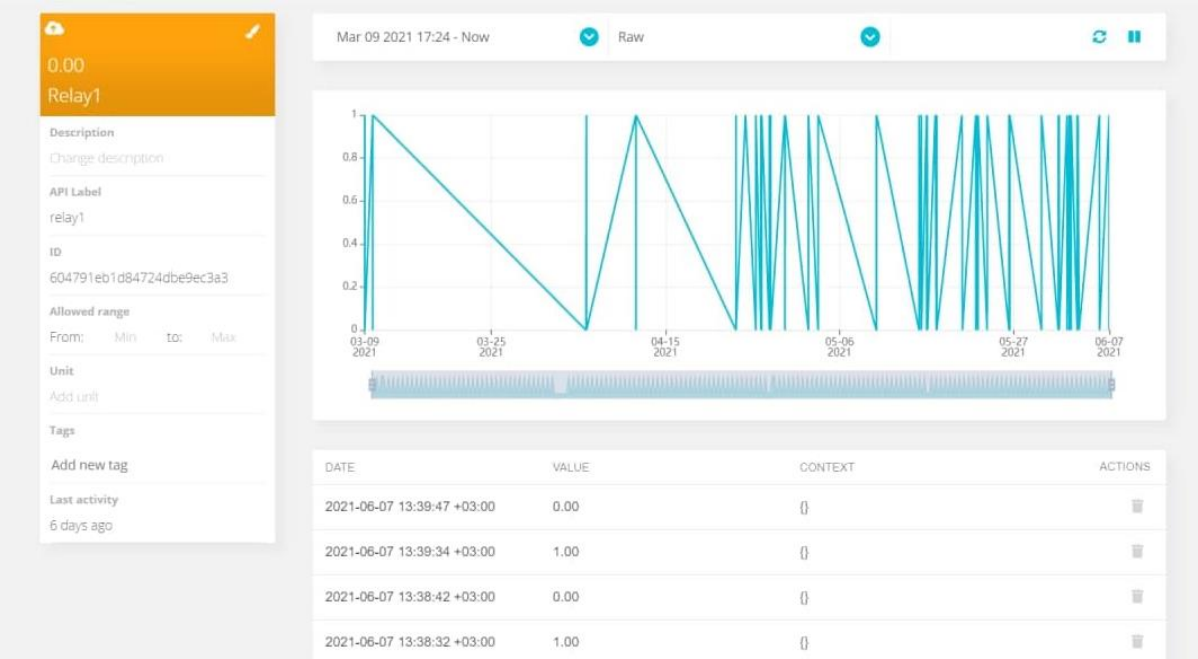


Figure 8: Relay log file.

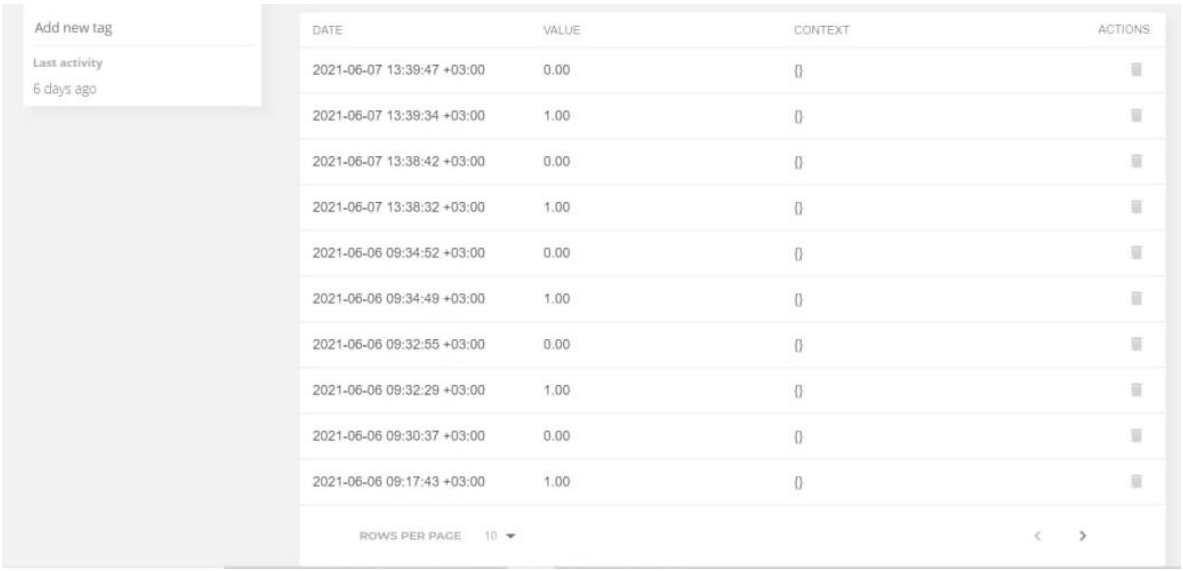


Figure 9: Relay log file.

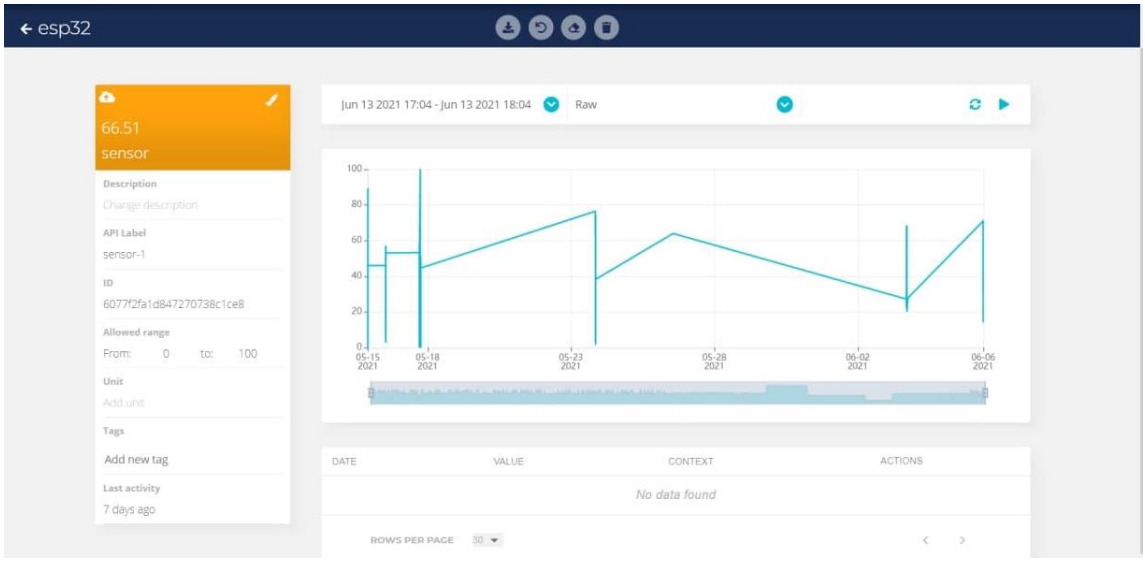


Figure 10: Sensor log file.

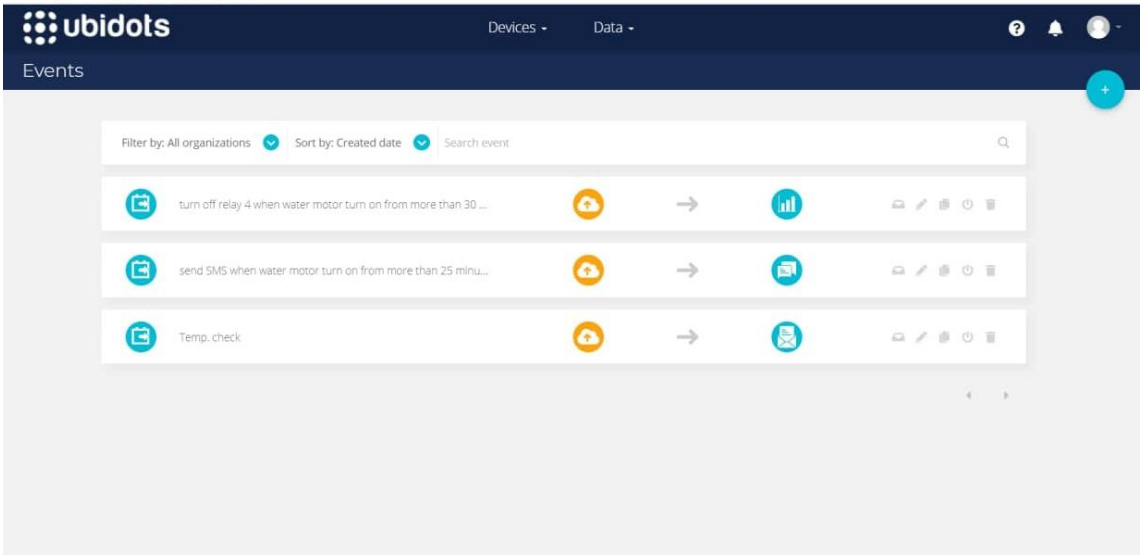


Figure 11: Some events

Figure 12: IFTTT Applets.

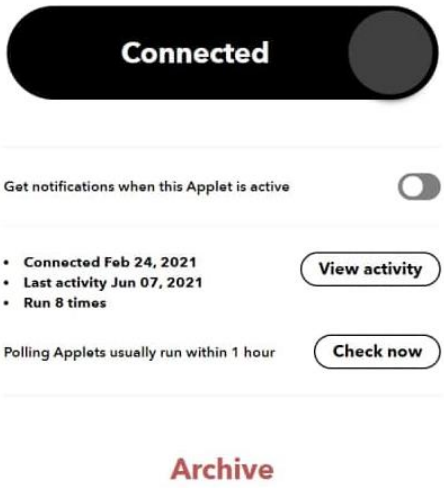


Figure 13: IFTTT log file.

4.2 Flow chart

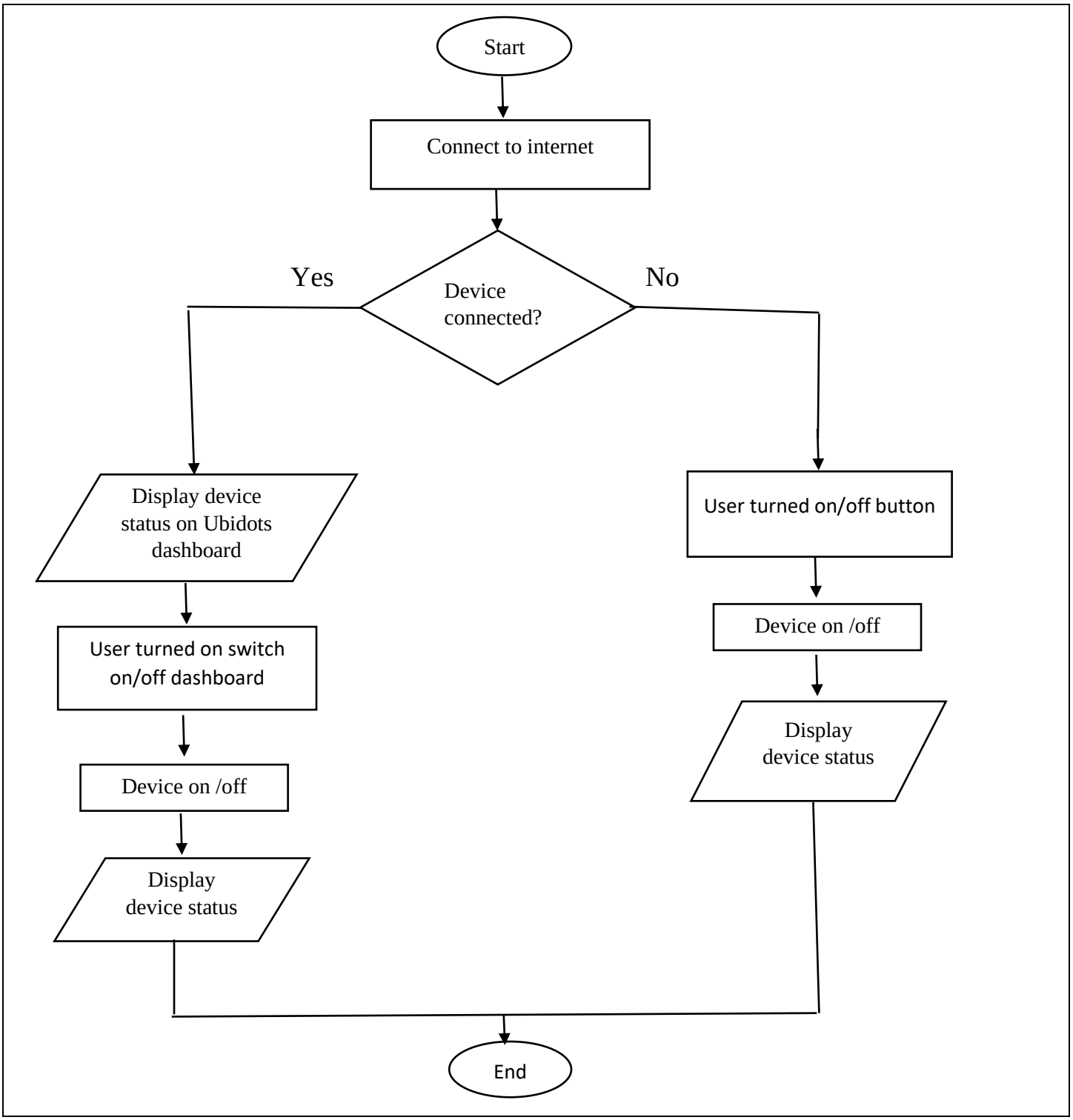


Figure 14: Flowchart

4.3 Pseudocode

We will write a pseudocode to explain the function that control relays part of our code:

```
switch (state) {
  case 1:
    first control
    Write value on relay
    Print first control
    Break
  case 2:
    second control
    Write value on relay
    Print second control
    Break
  case 3:
    third control
    Write value on relay
    Print third control
    break
  case ERROR_VALUE:
    print error when error occurred
    break
    default:
    print default
```

5

Chapter five:

Validation and Discussion

5.1 Description of the implementation

5.2 Implementation issues

5.3 Implementation challenges

Validation and Discussion

5.1 Description of the implementation

At the beginning, we wrote the code that links between the parts of the system hardware and software. The code contains the libraries for the controller and the MQTT broker, and it also contains the Ubidots token and many functions to run the relays that are connected to the controller and connect to the Internet and the dashboard in the Ubidots platform.

To build the hardware, we connected the ESB32 which carries the code to the Relay via Logic Level Converter, and a Relay that communicates directly with the electrical appliances in the home.

To build the dashboard on the Ubidots platform, we created a device (ESP32) which contains variables (relays, sensors, and others). These variables connect to the code using MQTT Publish and Subscribe.

The Ubidots platform gives us the ability to create events. This event contains an if this then that, where the event executes the condition command if the condition is fulfilled according to the inputs that we specified.

As for the IFTTT platform, we can create triggers linked with Standard's such as Google Assistant, the principle of Trigger work is the same principle for Events in the Ubidots platform, where these Triggers execute the condition command if the condition is fulfilled according to the inputs that we specified.

5.2Implementation issues

5.2.1 Hardware issues

- Relay test and connection

  The relay connected to the ESP32 via 3V to 5V Logic Level Converter because the GPIOs voltage is less than 5 volts, after connecting ESP32 with the electricity and running the code the relay worked properly, the relay worked as an electrical switch turn on or off the device based on the switch widget status on Ubidots dashboard and on serial monitor of Arduino IDE.

- sensor test and connection

  The sensor connected directly to the ESP32 via GPIO pin and the sensor use MQTT protocol to send data to the Ubidots dashboard.

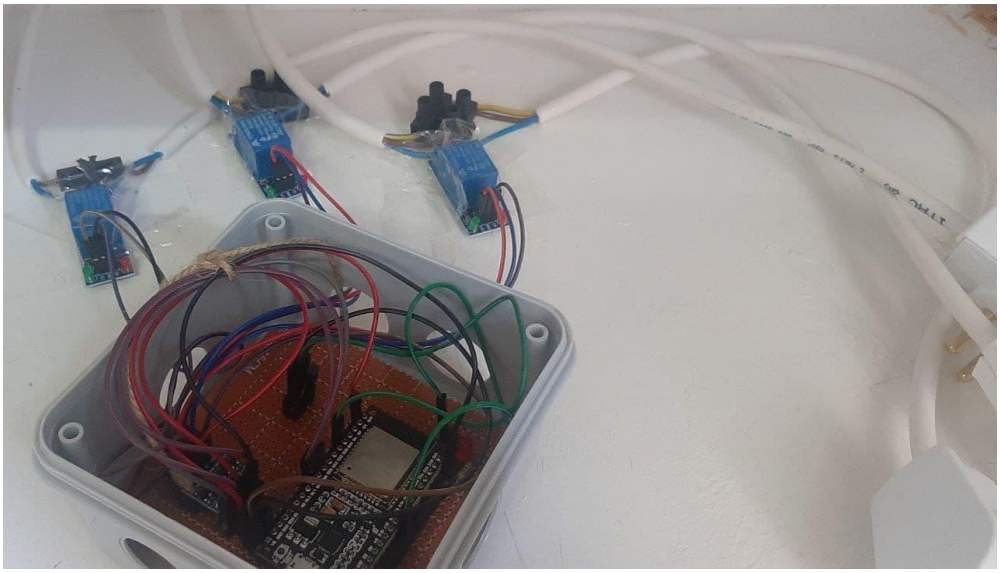


Figure 15: Project circuit

5.2.2 Software issues

- Ubidots dashboard and events

  The dashboard in the Ubidots platform contains variables by which we control the devices or variables for the display. As for the event in the platform, it depends if this then that system.

- IFTTT Standard

  The IFTTT platform is a standard platform for the if this then that system, as it enables us to create an applet to facilitate the control of home appliances.

5.3 Implementation challenges

1. At first we went to use the eWelink app but for the privacy of the application and our inability to deal with its own libraries we went to use the Ubidots application due to its high rating by previous users.
2. We faced a problem accessing the MQTT server, as the port was not open. We solved this problem by downloading the server to the local device (laptop) and opening the port dedicated to MQTT.
3. The voltage specified for the pins in the ESP32 is 3.3 volt, and the relay deals with 5-volt DC input, which is not compatible. To solve this problem, we needed an intermediary between the two modules, which is a logic level converter.

6

Chapter six:

Conclusion

6.1 Conclusion

Our project aims to enable the home owner to control the devices in his home with his mobile phone, as he can know the status of each device and verify whether it is operational or not, and the home owner can interact with his home appliances, whether inside or outside the home, and if he is inside the home, he has a choice Voice control of the devices or via the Ubidots app by pressing the switches on the dashboard on his phone.

As a future update, these devices will generate data and communicate with each other. So we will need a common language to deal with this data, and also to improve the workings of these devices in the smart home.

That is, people who want to come after us in the future can create a database. Later, the relevant persons will be able to use this database to find solutions to potential problems related to specifications or equipment work. Thus, manufacturers will be provided with potential future solutions and new ways to improve their products.

References

[1] Kyoochun Lee, title: *"The Internet of Things (IoT): Applications, investments, and challenges for enterprises",* April (2015), pp.1-4, Available at: https://www.sciencedirect.com/science/article/pii/S0007681315000373

[2] Title: "*RTOS*", last access to web page at November 2020, Available at: https://icircuit.net/esp32-introduction-freertos/1297

[3] Title: "*Ubiquitous Smart Home System Using Android Application*", last access to web page at April 2021, Available at: https://arxiv.org/ftp/arxiv/papers/1402/1402.2114.pdf

[4] Title: "*Design and Implementation of a WiFi Based Home Automation System*", last access to web page at April 2021, Available at: https://www.researchgate.net/publication/230883124_Design_and_Implementation_of_a_WiFi_Based_Home_Automation_System

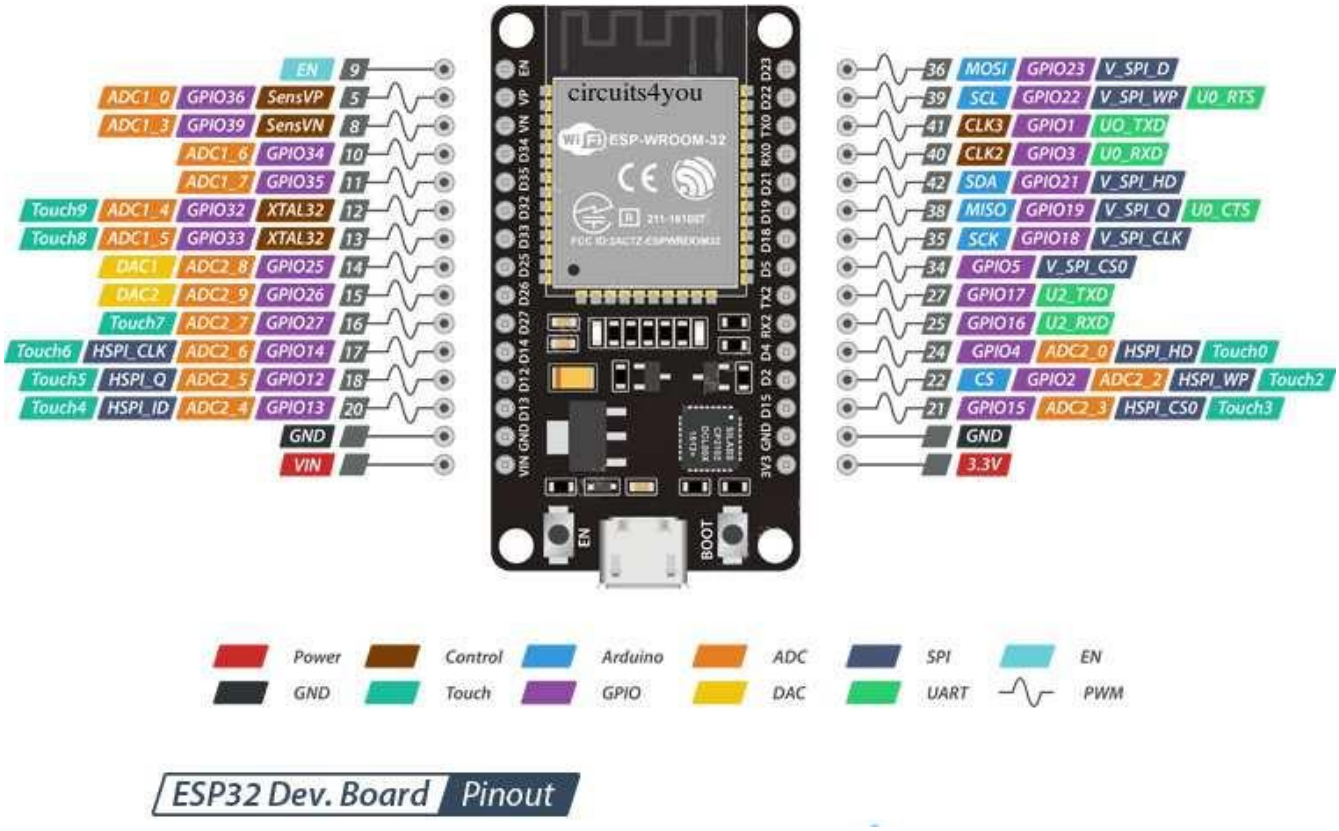
[5] Title: "*Home Automation Based on Raspberry Pi Single Board Computer*", last access to web page at April 2021, Available at: http://scholar.ppu.edu/bitstream/handle/123456789/630/Home%20Automation%20Based%20on%20Raspberry%20Pi%20Single%20Board%20Computer%20-2015.pdf?sequence=1&isAllowed=y

[6] Title: "*MQTT protocol*", last access to web page at November 2020, Available at: https://mqtt.org

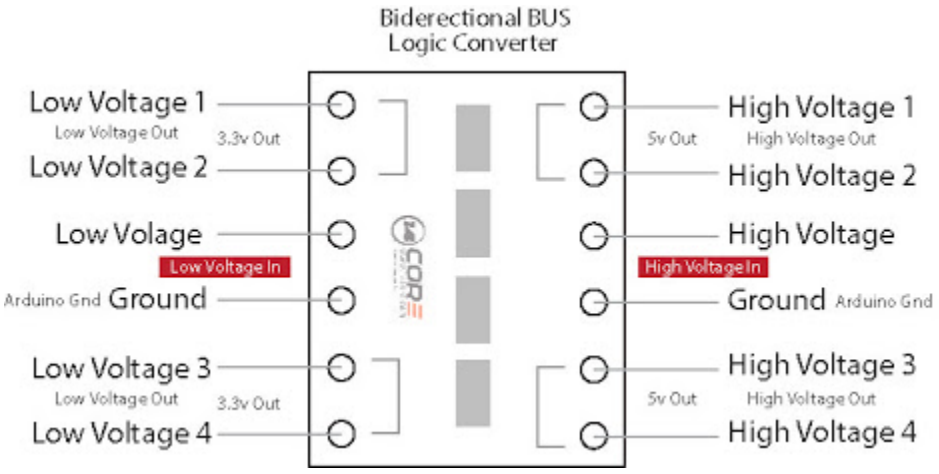
[7] Title: "*IFTTT standard*", last access to web page at April 2021, Available at: https://www.computerworld.com/article/3239304/what-is-ifttt-how-to-use-if-this-then-that-services.html

[8] Title: "*ESP32 Specification*", last access to web page at November 2020, Available at: https:// espressif.com

Appendices



ESP32 Pinout



Logical Level Converter

Auxiliary Functions from Ubidots platform

Callback Function:

```
void callback(char* topic, byte* payload, unsigned int length) {
  char* variable_label = (char *) malloc(sizeof(char) * 30);;
  get_variable_label_topic(topic, variable_label);
  value = btof(payload, length);
  set_state(variable_label);
  execute_cases();
  free(variable_label);
}
```

Get variable label topic Function

```
// Parse topic to extract the variable label which changed value
void get_variable_label_topic(char * topic, char * variable_label) {
  Serial.print("topic:");
  Serial.println(topic);
  sprintf(variable_label, "");
  for (int i = 0; i < NUMBER_OF_VARIABLES; i++) {
    char * result_lv = strstr(topic, variable_labels[i]);
    if (result_lv != NULL) {
      uint8_t len = strlen(result_lv);
      char result[100];
      uint8_t i = 0;
      for (i = 0; i < len - 3; i++) {
        result[i] = result_lv[i];
      }
```

```
      result[i] =  '\0';
      Serial.print("Label is: ");
      Serial.println(result);
      sprintf(variable_label, "%s", result);
      break; }
  }
}
// cast from an array of chars to float value.
float btof(byte * payload, unsigned int length) {
  char * demo_ = (char *) malloc(sizeof(char) * 10);
  for (int i = 0; i < length; i++) {
    demo_[i] = payload[i];
  }
  return atof(demo_);
}
```

Set state Function:

```
// State machine to use switch case
void set_state(char* variable_label) {
  variable = 0;
  for (uint8_t i = 0; i < NUMBER_OF_VARIABLES; i++) {
    if (strcmp(variable_label, variable_labels[i]) == 0) {
      break;
    }
    variable++;
  }
  if (variable >= NUMBER_OF_VARIABLES) variable = ERROR_VALUE; // Not valid }
```