**Google Assistant**

**Home Automation**

**Home automation system to make life easier & our home smarter.**

**A**

**Project Report**

**On**

**“Google Assistant Home Automation**”

Submitted By

**Neelesh Sabhajeet Maurya**

Under the guidance of

**Prof. Unnati Dhavare**

In the partial fulfilment of

**B.Sc. Computer Science**

**Department of Computer Science**

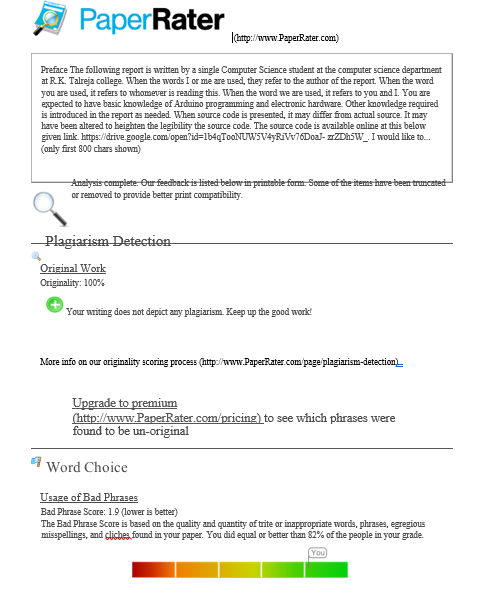
**R.K. TALREJA COLLEGE OF ARTS, SCIENCE AND COMMERCE**

**Ulhasnagar-3**

**University of Mumbai**

**2019 – 2020**

**Plagiarism Report**



**Declaration**

I hereby declare that the above shown particulars are true to be best of my knowledge and belief.

-**Neelesh Maurya**

**Prof. Kiran Gurbani Prof. Unnati Dhavare**

**H.O.D(Computer Science Department) (Project Guide)**

**Preface**

The following report is written by a single Computer Science student at the computer science department at R.K. Talreja college.

When the words I or me are used, they refer to the author of the report. When the word you are used, it refers to whomever is reading this. When the word we are used, it refers to you and I.

You are expected to have basic knowledge of Arduino programming and electronic hardware. Other knowledge required is introduced in the report as needed.  
When source code is presented, it may differ from actual source. It may have been altered to heighten the legibility the source code. The source code is available online at this below given link.

https://drive.google.com/open?id=1b4qTooNUW5V4yRiVv76DoaJ- zrZDh5W\_.

I would like to thank Prof Kiran Gurbani (H.O.D) & Prof Unnati Dhavare for her supervision during this project.

**Neelesh Maurya**

**Abstract**

This paper presents a proposal for home automation using voice via Google Assistant Home automation or domestics a term for home automation being remotely controlled by any Android OS smart phone. This paper describes how to control and monitor home appliances using android application or just giving voice commands to Google Assistant over internet.

As technology is advancing so houses are also getting smarter. Modern houses are gradually shifting from conventional switches to centralized control system, involving remote controlled switches. Presently, conventional wall switches located in different parts of the house makes it difficult for the user to go near them to operate. Even more it becomes more difficult for the elderly or physically handicapped people to do so. Remote controlled home automation system provides a most modern solution with smart phones.

There are number of commercial home automation systems available in market. But they have certain limitations to it. In this project a microcontroller named Nodemcu ESP8266 is used which includes creating Adafruit account then linking to IFTTT website then adding to Google Assistant for voice commands. In this home automation appliances like Lights, Cooling Fan, Projector, etc are used which can be controlled easily using the Google assistance from the voice control. It is also connected with some sensors like PIR motion sensor for motion detection, DHT11 sensor for monitoring temperature and a gas sensor which sends the level of co2, propane or butane gas, and sends the data to an application and also to my/your AdafruitIo dashboard. As the user give the voice command to the mike according to that the home appliances can be switched ON/OFF accordingly.

**Acknowledgement**

“Perseverance, inspiration and motivation have always played a key role in any venture. It is not just the brain that matters most, but that which guides them. The character, the heart, generous qualities and progressive forces. What was conceived just as an idea materialized slowly into a concrete fact.”

At this level of understanding it is often difficult to understand wide spectrum of knowledge without proper guidance and advice. Hence, we take this opportunity to express our heartfelt gratitude to my project guide **Prof. Unnati Dhavare** who had faith in us and allowed me to work on this project.

I would like to thanks **Prof. Kiran Gurbani (H.O.D)** for his immense interest, valuable guidance, constant inspiration and kind co-operation throughout the period of word undertaken, which has been instrumental in the success of our project.

I also acknowledge our profound sense of gratitude to all the teacher who have been instrumental for providing us the technical knowledge and moral support to complete the project with full understanding.

We thank our friends and family for their moral support to carve out this project and above all GOD for removing all hurdles in the way.

**Contents**

**Page No:**

**1. Preliminary Investigation 1**

1.1. Introduction to the system 2

1.2. Project Objectives 3

1.3. Application 4

1.4. Limitations of the system 7

1.5. Project scope 8

1.6. Technology exposure that project provides 9

1.7. Feasibility study 10

1.8. Gantt chart 11

**2. System Analysis 12**

2.1. Flowchart of the system 13

2.2. Use case Diagram 14

2.3. Sequence Diagram 15

2.1.1. Sequence diagram for login. 15

2.2.2. Sequence diagram for Controlling Devices. 16

2.3.3. Sequence diagrams for Sensor monitoring. 17

2.4 Deployment Diagram 18

2.5. Block Diagram 19

2.6. Network Architecture Diagram 20

2.7. Circuit Diagram 21

**3. Architecture of GAHA 22**

**4. List and Cost of Hardware’s 25**

**5. About Components 27**

5.1. Nodemcu Esp8266. 28

5.1.1. ESP8266 Framework. 30

5.1.2. Web Server. 31

5.2. PIR motion sensor. 32

5.3. DHT11 sensor. 34

5.4. MQ-2 gas sensor. 36

5.5. 6 - Channel Relay Board. 38

5.6. 16 Channel Analog/Digital Multiplexer. 41

**6. Platform used 42**

6.1. AdafruitIo. 43

6.2. IFTTT. 45

6.3. Google assistant. 46

**7. Implementation 47**

7.1. Getting Nodemcu ready 48

7.1.1. Installing Libraries for AdafruitIo & Dht11 sensor. 51

7.2. Setting up AdafruitIo 52

7.2.1. Creating a dashboard. 52

7.2.2. Adding Blocks and Feeds. 54

7.3. Setting up IFTTT 59

7.4. Setting up IOT MQTT Panel application and WIFI. 65

7.5. Program. 70

**8. Testing and Validation 81**

**9. Conclusion 83**

**10. Problem faced 86**

**11. Future scope 88**

**12. Bibliography 90**

**Preliminary**

**Investigation**

**1.1. Introduction to the system**

Home, it is the place where one fancies or desires to be after a long tiring day. People come home exhausted after a long hardworking day. Some are way too tired that they find it hard to move once they land on their couch, sofa or bed. So, any small device/technology that would help them switch theirs lights on or off, or play their favourite music etc. on a go with their voice with the aid of their smart phones would make their home more comfortable. Moreover, it would be better if everything such as warming bath water and adjusting the room temperature were already done before they reach their home just by giving a voice command. So, when people would arrive home, they would find the room temperature, the bath water adjusted to their suitable preferences, and they could relax right away and feel cosier and rather, feel more homely. Human assistants are like housekeepers were a way for millionaires to keep up their homes in the past. Even now when technology is handy enough only the well to do people of the society are blessed with these new smart home devices, as these devices costs are a bit high. However, not everyone is wealthy enough to be able to afford a human assistant, or some smart home kit.

Hence, the need for finding an inexpensive and smart assistant for normal families keeps growing. This paper proposes such inexpensive system. It uses the Google Assistant, the IFTTT application, the AdafruitIo, the Nodemcu microcontroller as the major components along with a relay board comprising of 6 relays. Some Sensors for temperature, humidity, Motion detection, Butane/Co2 gas detection. Natural language voice is used to give commands to the Google Assistant. All of the components are connected over the internet using Wi-Fi which puts this system under the IoT. And the devices are controlled and monitored using voice commands through google assistant and also controlled by an application named IOT MQTT panel.

**1.2. Project objective**

Google assistant home automation (GAHA) should be able to control home appliances wirelessly with effective and efficiently.

* **Controlling Home appliances via Application (switch)**

To which we can control all the devices remotely and wirelessly over the internet using our smart phone.

* **Real Time data Monitoring from sensors**

To receive the exact temperature and humidity of our room or our whole home, also we can detect the level of co2, butane and propane level of our room in case of any leakage occur in our cooking gas then this sensor informs us on IOT application.

* **Secure Connection Channels between Application and Nodemcu**

Use of secure protocols over Wi-Fi so that other devices cannot control the appliances. Options for secure connection is SSL over TCP.

* **Controlled by Voice commands through Google assistant**

To make the home appliances flexible in control, controlled by voice commands through google assistant on any smart phone or also through google home.

##### Controlled by any device capable of Wi-Fi (Mobile, PC)

To make the home appliances flexible in control, any device capable of Wi-Fi connectivity will able to control the home appliances from remote location.

**1.3. Application**

**Home Is Where the Smart Is**

Machine-to-machine communication, and you understand you’re not the most tech-savvy consumer, it’s impossible that you’ve missed the abundance of home automation products filling the shelves and ads of every home improvement store. Suddenly an ordinary errand for light bulbs will leave you wondering if your lamp could send you a message alerting you that the light bulb needs to be replaced. Furthermore, if your lamp is talking to you, could your refrigerator and sprinkler system be too? Experts say: Yes, the possibilities are endless. If that’s the case, where do you begin?

Any day-to-day, repeatable process is automatable with smart home applications. The greater the control and flexibility of these processes, the more energy and cost savings the resident experiences, which are factors anyone who pays utilities strives to moderate. The smart home revolution is likely to be more of an evolution, with the incorporation of one or two home systems at a time, gradually automating our households through smart mobile devices. However, with these elements of efficiency comes the question of ease of use. Will it bring you enjoyment or exasperation? With so many brands and models already available in an ever-growing market, how do you know which is best for you?

**Lighting Control:**

Leaving the Dark Ages and Stepping into the Light Smart lighting allows you to control wall switches, blinds, and lamps, but how intuitive is a lighting control system? It turns out, quite; its capabilities are extensive. You’re able to schedule the times lights should turn on and off, decide which specific rooms should be illuminated at certain times, select the level of light which should be emitted, and choose how particular lights react through motion sensitivity, as seen with Belkin’s WeMo Switch + Motion, which is both affordable and easy to use with its plug-and-play simplicity.

**HVAC Regulation: No Longer Burned by Your Heating Bill**

As fuel costs rise and the availability and sustainability of our resources becomes a greater concern, heating/cooling our homes efficiently is less a budgetary bonus and more of a necessity. Over the past year, smart thermostats and automated home heating systems have become more readily available and easily incorporate into any home. Heating and cooling our homes consume an average of 50% of energy costs yearly, making daily HVAC regulation progressively rewarding. Maintaining a substantial lead among the nearly non-existent competition, the Nest Learning Thermostat, learns your heating and cooling preferences over time, eliminating the need for programming and is accessible from your smartphone app. With automated HVAC you are able to reduce the heat when a room is unoccupied, and increase or decrease it at specific times based on your schedule and occupancy.

**Lawn Irrigation Systems: The Grass is Always Greener**

A lush and healthy lawn is a source of pride for most homeowners, but the weather doesn’t always cooperate and provide the adequate elements for a flourishing landscape. For decades we’ve relied on sprinkler systems to keep our yards at peak presentation, but at what cost? The average American home spends approximately 30% of their daily water usage on lawn and garden maintenance. Nearly half of that amount is wasted due to inefficiency. If you apply that statistic to the national average, up to 4.5 billion gallons of water is wasted per day through ineffective watering methods. If we reflect upon the monetary impact of this, it results in Americans spending over a thousand dollars a year in water, with a portion of that being waste. The global effects are even greater when you consider the growing concern over climate change and the dramatic decrease in agricultural natural resources. However, sprinkler control systems, like Skydrop, are providing water regulation through real-time communication with local weather data. If a rainstorm develops and deposits two inches of rainwater on your lawn, the automated sprinkler detects the saturation and disables its scheduled watering. Conversely, the system will be alerted to dry conditions and supply the necessary amount of nourishment, without over-watering.

**Security Systems: Knock, Knock...**

Who’s there? The Internet of Things. While efficiency and conservation are certainly IoT benefits, its potential to have improved control over home security is a primary focus. Smart locks, like Kwikset’s Kevo, a Bluetooth enabled electronic deadbolt, and various connected home security systems, such as iSmartAlarm, offer a variety of features including door and window sensors, motion detectors, video cameras and recording mechanisms. All of which are connected to a mobile device and accessible via the cloud, thus enabling you to access real-time information on the security status of your home. Naturally, there is a great deal of scrutiny regarding the level of trust in controlling your home’s security system via a mobile device, but it begs earnest exploration when weighing the potential benefits and peace of mind it provides homeowners.

**1.4. Limitations of the system**

**Installation**

Depending on the complexity of the system, installing a home automation device can be a significant burden on the homeowner. It can either cost you money if you hire an outside contractor or cost you time if you venture to do it yourself.

**Complex Technology**

Automating everything in life may sound extremely appealing, but sometimes a good old-fashioned flip of the switch is a lot easier than reaching for your smart phone to turn lights on and off. Before you decide which system is right for you, think about how far you really want to take home automation in your household.

**System Compatibility**

[Controlling all aspects of home automation](https://www.mythinkenergy.com/home-automation) from one centralized platform is important, but not all systems are compatible with one another. Your security system, for example, may require you to log in to one location to manage settings, while your smart thermostat may require you to log in to another platform to turn the air conditioner on and off. To truly leverage the convenience of home automation, you may need to invest in centralized platform technology to control all systems and devices from one location.

**Cost**

Even though the price of home automation systems has become much more affordable in recent years, the cost to purchase and install a device can still add up. Consumer Reports offers a wide range of information and insights – including costs – on the best home automation systems on the market

**1.5. Project Scope**

The project aims at designing a prototype for controlling the home appliances that can be controlled wirelessly via an application that provides the features of controlling devices and also monitoring the devices. An application runs on android device. The system can be used in wide range of areas. Also, the devices are controlled using voice commands through google assistant.

The system integrated with different features can be applied in the following fields.

* **The system can be used in home, small offices to the big malls**

The system can be used from home to offices to control the electrical appliances. Also, to monitor the temperature, any leakage of gas and also the motion-controlled monitoring.

* **For remote access of appliances in internet using WIFI**

The home/office appliances can be controlled in intra-network or can be accessed via internet.

* **For the development of technology friendly environment**

The system incorporated the use of technology and making smart home automation. By the use of day to day gadgets we can utilize them for different prospective.

**1.6. Technology exposure that project provides**

* Google’s Android open source technology.
* Wi-Fi technology.
* Interfacing Nodemcu with sensors.
* Interfacing relays with ac and dc power sources.
* Embedded programming.

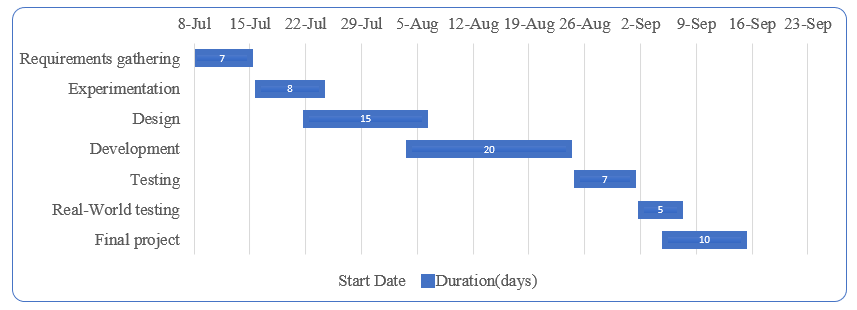
**1.7. Feasibility study**

Project Feasibility study is an activity that verifies whether a project can be started and completed successfully. The objective is to determine whether the development project has a reasonable chance of success.

Feasibility study includes consideration of all the possible ways to provide solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirement.

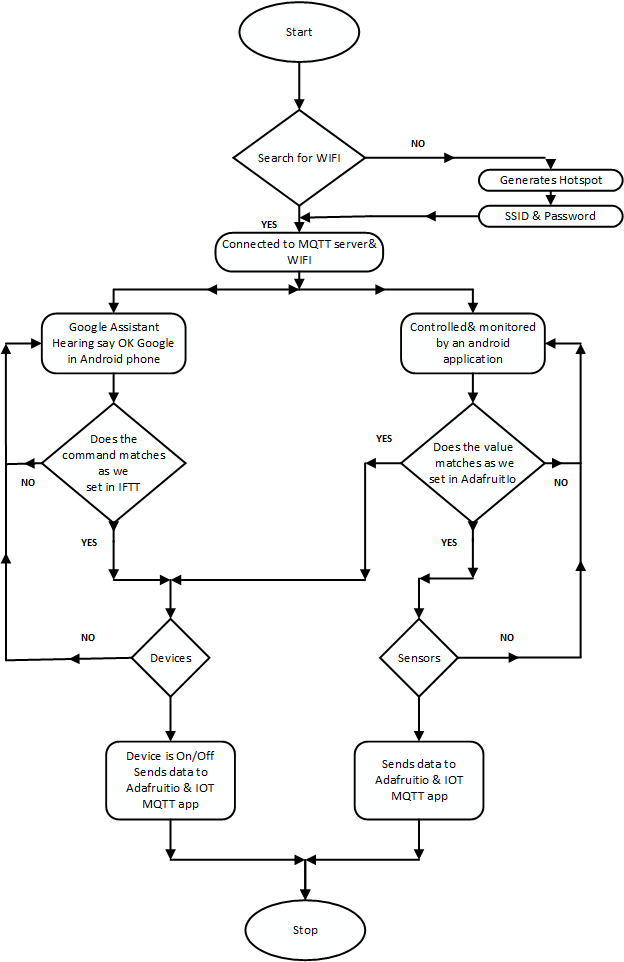
* **Economic Feasibility:** This is the very important aspect to be considered while developing project.
  + We decide the technology based on minimum possible cost factor.
  + All hardware and software cost have to be borne by the organization.
  + Overall, we have estimated that the benefits the organization is going to receive from the proposed system will surely overcome the initial cost and the later on running cost for system.
* **Technical Feasibility:** This includes the study of function performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study, we study complete functionality to be provided in the system, as described in the system requirement specification and checked if everything was possible using different types of frontend and backend platforms.

**1.8. Gantt Chart**

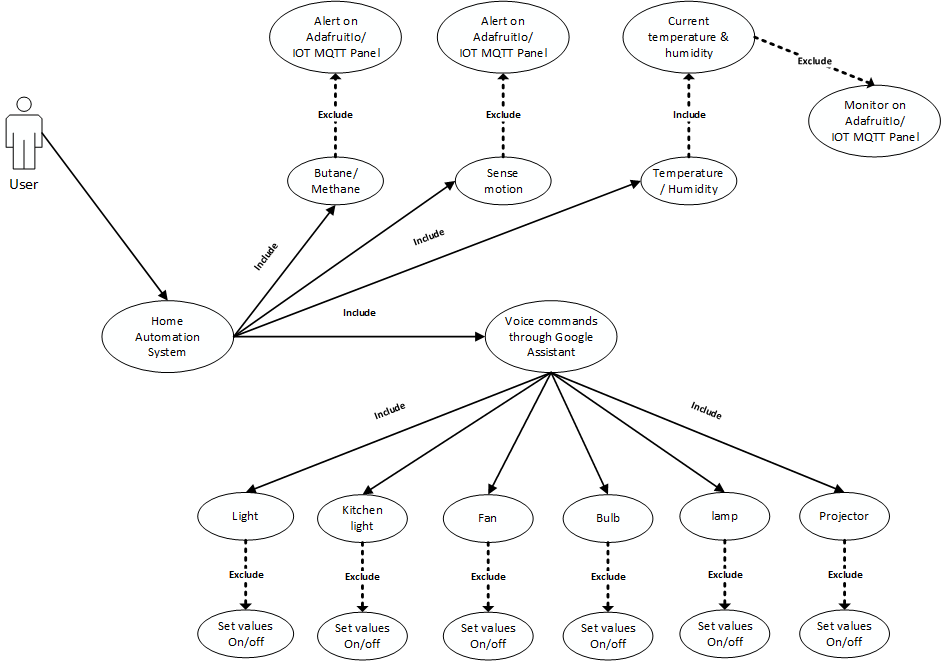


**System Analysis**

**2.1. Flowchart of the system**

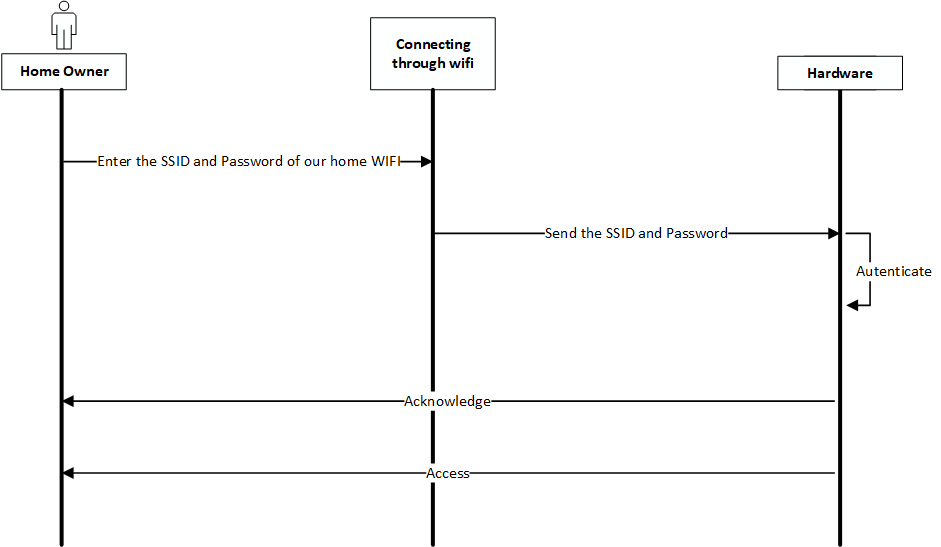


**2.2. Use Case Diagram**

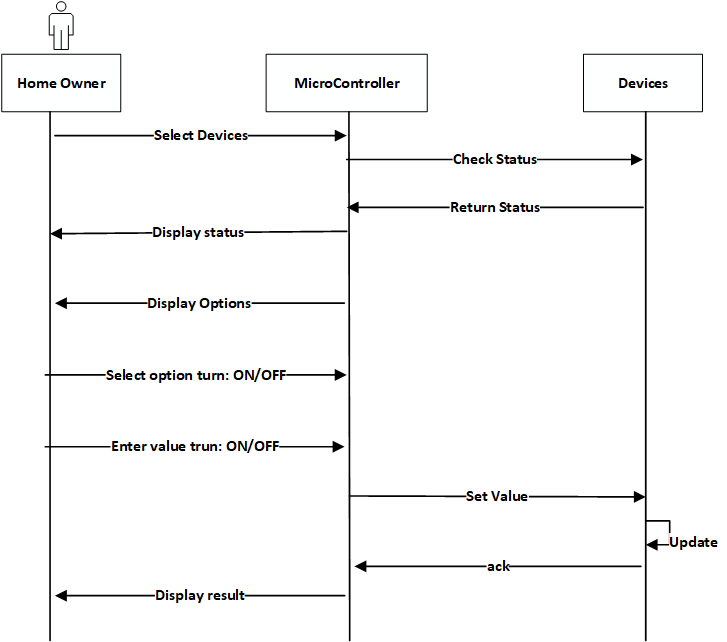


**2.3. Sequence Diagram**

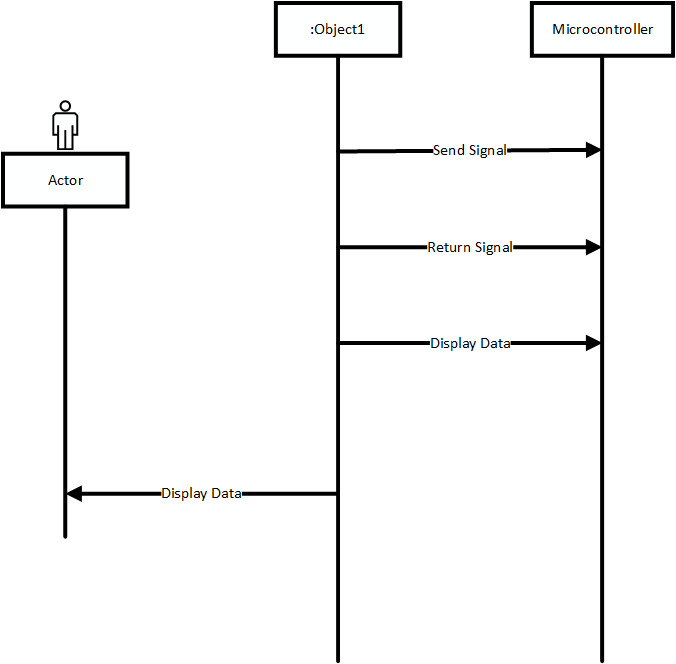
**2.3.1. Sequence diagram for login.**



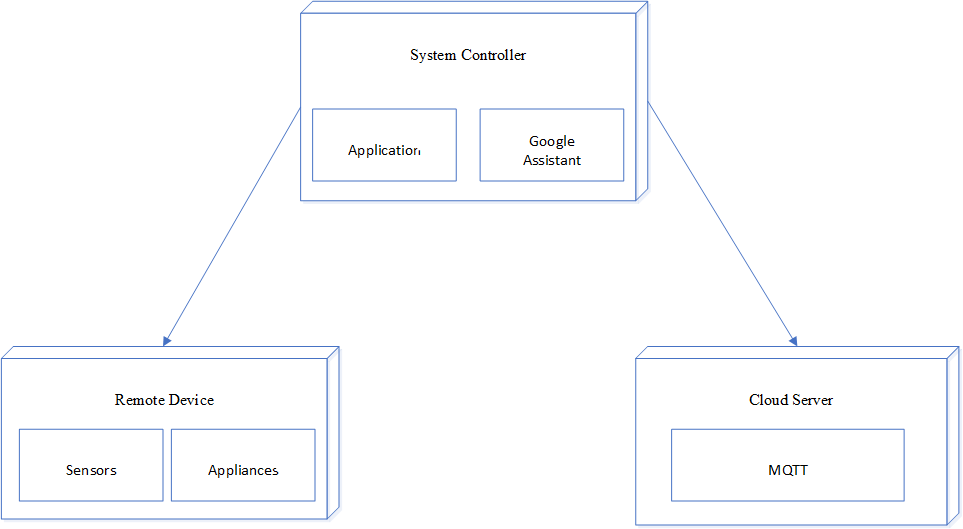
**2.3.2. Sequence diagram for Controlling Devices.**



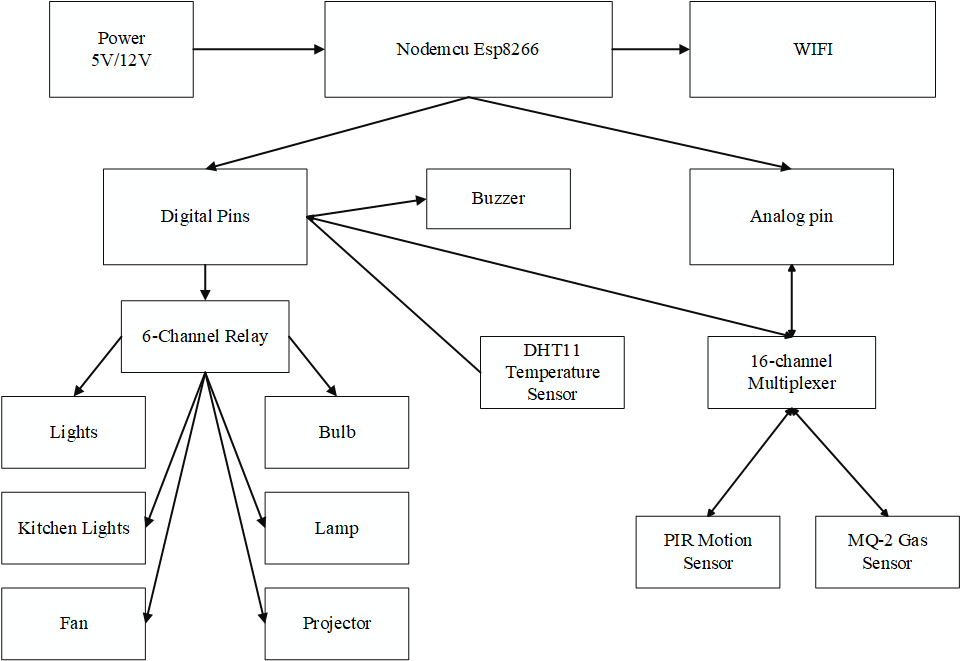
**2.3.3. Sequence diagrams for Sensor Monitoring.**



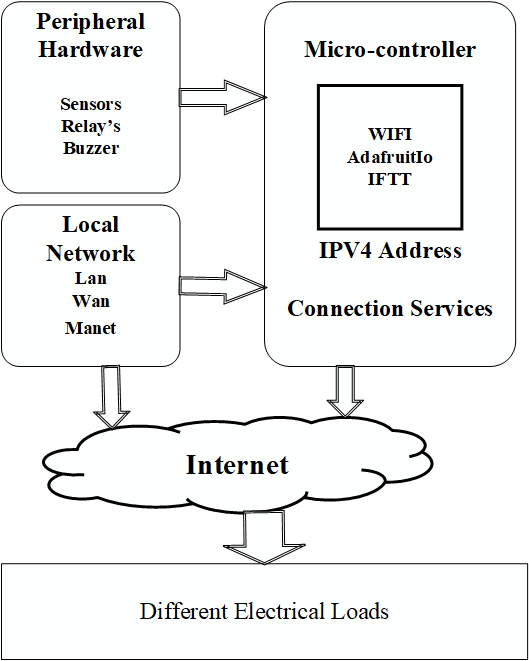
**2.4. Deployment Diagram**



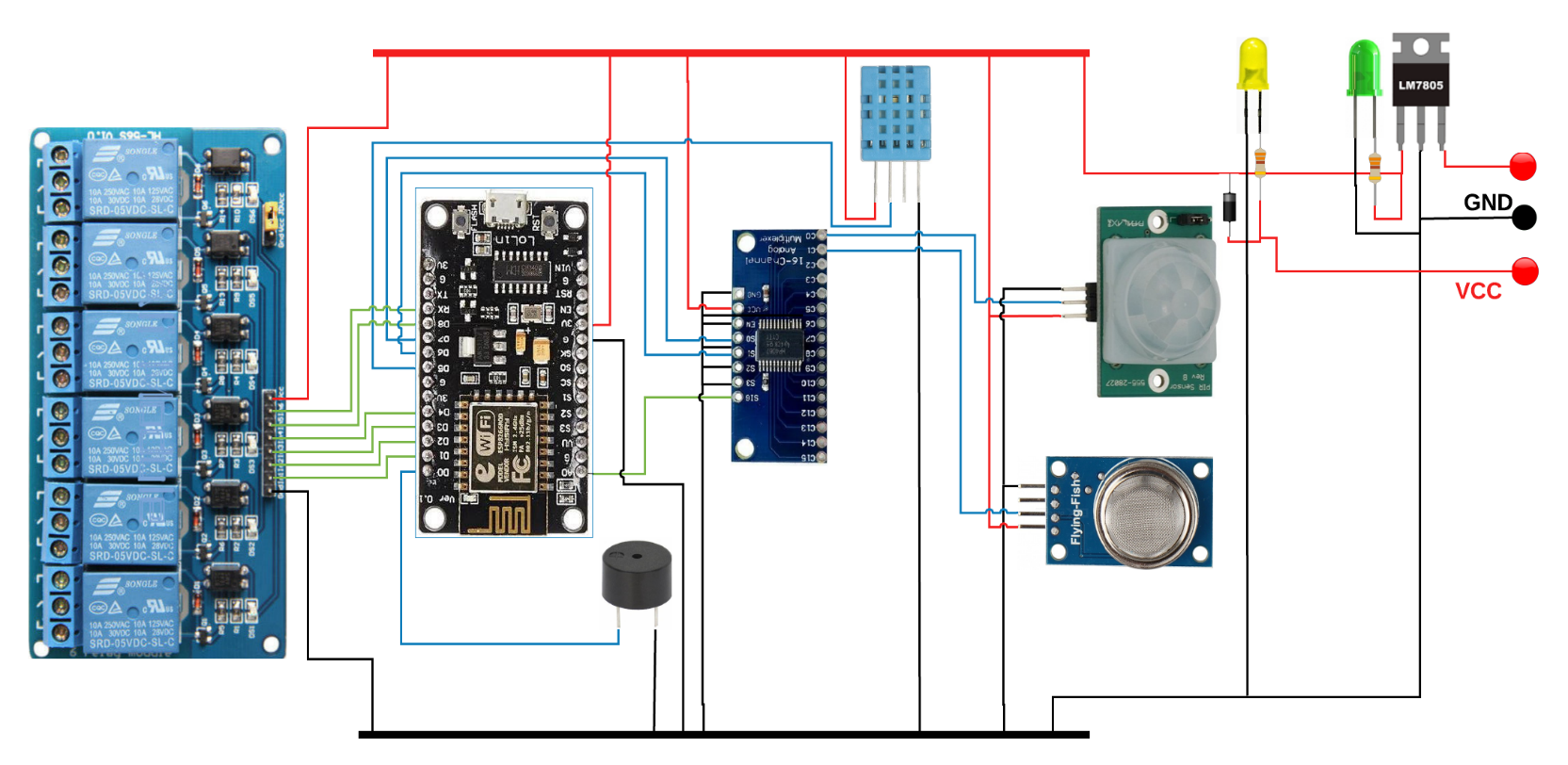
**2.5. Block Diagram**



**2.6. Network Architecture Diagram**



**2.7. Circuit Diagram**



**+5V**

**+12V**

**Architecture of GAHA**

**3. Architecture of GAHA**

The whole system is broken down into two main categories, areas follows

i. The hardware It has the capability to connect to the router. It would also be able to turn on/off specified devices, such as lights and fans. It is called the Control Unit. And The sensors and, It is also show the data of the sensors to an application.

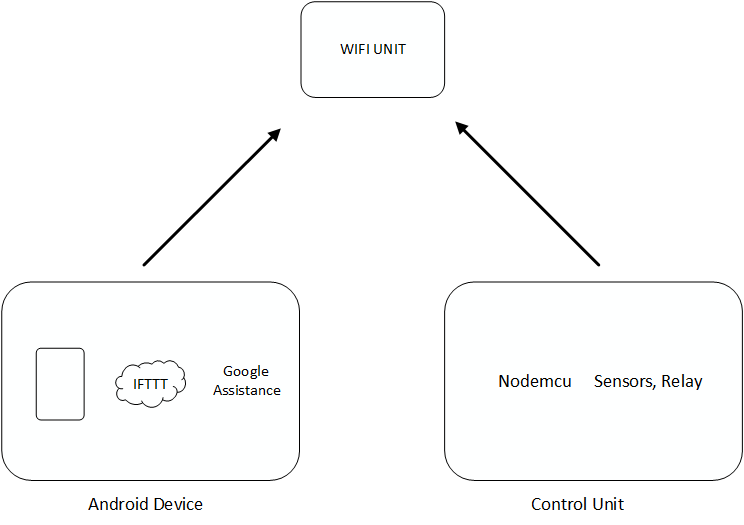
ii. The Software-the IFTTT app and the Google Assistant constitute the software of the design and these applications would be integrated in the Android device.

When the user sends command through google assistant or IOT MQTT panel application it first goes to the IFTTT there we given set statements that states IF THIS THEN THAT you can understand this when you see the architecture of our system. From there it goes to the Adafruit is cloud service it is like MQTT broker which interacts with the NodeMCU The Control Unit comprises of the microcontroller-NodeMCU and the 4Channel Relay board. Android device communicates with the microcontroller and sends the desired signal via the internet. Figure 1below shows the basic system design architecture.

The Control Unit comprises of the microcontroller-NodeMCU and the 8 Channel Relay board. The Android device communicates with the microcontroller and sends the desired signal via the internet. Figure 1 below shows the basic system design architecture.



The hardware also called the Control Unit comprises of the NodeMCU microcontroller and the Relay board. NodeMCU digital output pins are connected to the Relay pins of the Relay board.



**List and Cost of**

**Hardware’s**

**4.** **List and Cost of Hardware’s**

|  |  |  |
| --- | --- | --- |
| **Components** | **Piece’s** | **Prices** |
| NodeMCU - 32-bit ESP8266 development board. | 1 | 400 |
| 6 channel Relay Module. | 1 | 400 |
| MQ-2 gas sensor. | 1 | 135 |
| DHT11 temperature &humidity sensor. | 1 | 105 |
| PIR Motion sensor. | 1 | 18 |
| 16 channel analog multiplexer. | 1 | 130 |
| 7805 IC | 1 | 10 |
| IN4007 diode. | 2 | 2 |
| 100 Ohm register. | 2 | 2 |
| Raw PCB or Zero PCB. | 1 | 20 |
| LED’s | 2 | 5 |
| Connecting cables(wires). | 1 mtr | 30 |
| Buzzer. | 1 | 30 |
| 3mm insulated copper wire. | 5 mtr | 100 |
| 12 v or 5 v power supply. | 1 | 150 |
| 6 channel Spy Card. | 1 | 100 |
| **Total** | | 1637 |

**About Components**

**5.** **About Components**

**5.1. Nodemcu Esp8266**

The NodeMCU (Node Microcontroller Unit) is an open source software and hardware development environment that is built around a very inexpensive System-on-a-Chip (SoC) called theESP8266. The ESP8266 is designed and manufactured by Express, contains all crucial elements of the modern computer: CPU, RAM, networking (wi-fi), and even a modern operating system and SDK. When purchased a single Nodemcu, the ESP8266 chip costs only ₹ 400 INR a piece. That makes it an excellent choice for this system design.

The NodeMCU aims to simplify ESP8266 development. It has two key components.

I. An open source ESP8266 firmware that is built on top of the chip manufacturer's proprietary SDK. The firmware provides a simple programming environment based oneLua (embedded Lua), which is a very simple and fast scripting language with an established developer community. For new comers, the Lua scripting language is easy to learn. And to add on NodeMCU can be programmed with the Android IDE too.

II. A development kit board, that incorporates the ESP8266 chip on a standard circuit board. The board has a built-in USB port that is already wired up with the chip, a hardware reset button, Wi-Fi antenna, LED lights, and standard-sized GPIO (General Purpose Input Output) pins that can plug into a bread board. Figure 2below shows the NodeMCU development board.

III. **Internal Specification**

* The ESP8266 is a Wi-Fi SoC integrated with a [Tensilica](https://en.wikipedia.org/wiki/Tensilica) Xtensa LX106 core.
* It has in all total 16 GPIO pins and also one analog pin.
* To compile an Arduino C/C++ source file for the target MCU's machine language.
* There are two significant sizes of flash -- the 512K and 4Mb (or bigger).
* MicrocontrollerATmega2560
* OperatingVoltage5V
* Input Voltage(recommended)7-12V
* Serial Peripheral Interface Bus.

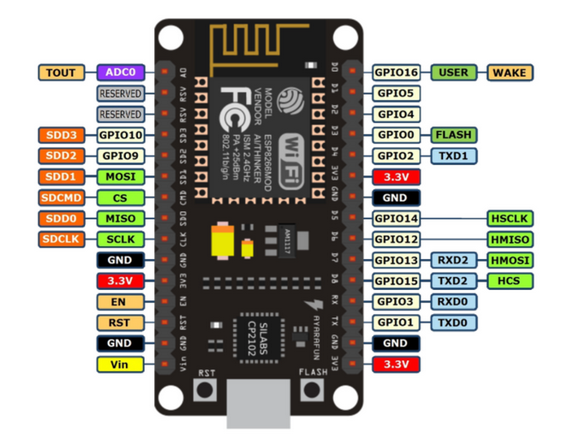
**IV. Programming**

* The Arduino Mega can be programmed with the Arduino software.
* The ESP8266 on the Nodemcu comes pre-burned with a boot loader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500protocol. You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming.



**NodeMCU**

**Pinout of NodeMCU ESP8266**



**5.1.1.** **ESP8266 Framework**

ESP modules are available from a variety of sources, and the firmware contained in the ESP8266 chips on the modules is almost always outdated and often of questionable origin. The firmware "updates" and tools that are available from these same sources are also sometimes suspect. Consequently, it is the aim of this article to document a procedure for downloading the latest available firmware directly from Espresso if and installing it on an ESP8266 using the flash tool provided by Espresso. In order to update the firmware on any ESP8266, it is necessary to have it properly powered and connected to a PC. In addition, a means of resetting the IC and putting it in the download mode must be provided. The schematic diagram and photograph below show the recommended setup; note that the wire colours in the schematic correspond to the wire colours

**5.1.2. Webserver**

A web server (sometimes called an HTTP server or application server) is a program that serves content using the HTTP protocol. This content is frequently in the form of HTML documents, images, and other web resources, but can include any type of file. The content served by the web server can be pre-existing (static content) or generated on the fly (dynamic content). In order to be considered a web server, an application must implement the HTTP protocol. Applications built on top of web servers (such as blogging software, forums, or wikis) belong in the separate web software category. A user agent, commonly a web browser or web crawler, initiates communication by making a request for a specific resource using HTTP and the server responds with the content of that resource or an error message if unable to do so. The resource is typically areal file on the server's secondary storage, but this is not necessarily the case and depends on how the web server is implemented. While, the primary function is to serve content, a full implementation of HTTP also includes ways of receiving content from clients. This feature is used for submitting web forms, including uploading of files. Web servers are not only used for serving the World Wide Web. They can also be found embedded in devices such as printers, routers, webcam sand serving only a local network. The web server may then be used as a part of a system for monitoring or administering the device in question. This usually means that no additional software has to be installed on the client computer, since only a web browser is required (which now is included with most operating systems)

**5.2. PIR motion sensor.**

PIR sensor detects a human being moving around within approximately 10m from the sensor. This is an average value, as the actual detection range is between 5m and 12m.PIR are fundamentally made of a pyro electric sensor, which can detect levels of infrared radiation. For [numerous essential projects](http://www.efxkits.com/) or items that need to discover when an individual has left or entered the area. PIR sensors are incredible, they are flat control and minimal effort, have a wide lens range, and are simple to interface with.

### **Motion Detection using PIR Sensor**

A PIR or a Passive Infrared Sensor can be used to detect presence of human beings in its proximity. The output can be used to control the motion of door.

Basically, motion detection uses light sensors to detect either the presence of infrared light emitted from a warm object or absence of infrared light when a object interrupts a beam emitted by another part of the device.

A PIR sensor detects the infrared light radiated by a warm object. It consists of pyro electric sensors which introduce changes in their temperature (due to incident infrared radiation) into electric signal. When infrared light strikes a crystal, it generates an electrical charge.

Thus, a PIR sensor can be used to detect presence of human beings within a detection area of approximately 14 meters.

In presence of human IR radiations, the sensor detects the radiations and converts it directly to electrical pulses, which is fed to the inverter circuit. The inverter circuit consists of a transistor, which gets into saturation with application of high base current and eventually develops a low collector voltage. Thus, the transistor output is low.

### **Areas of Applications of PIR Sensors**

* All outdoor Lights
* Lift Lobby
* Multi Apartment Complexes
* Common staircases
* For Basement or Covered Parking Area
* Shopping Malls
* For garden lights

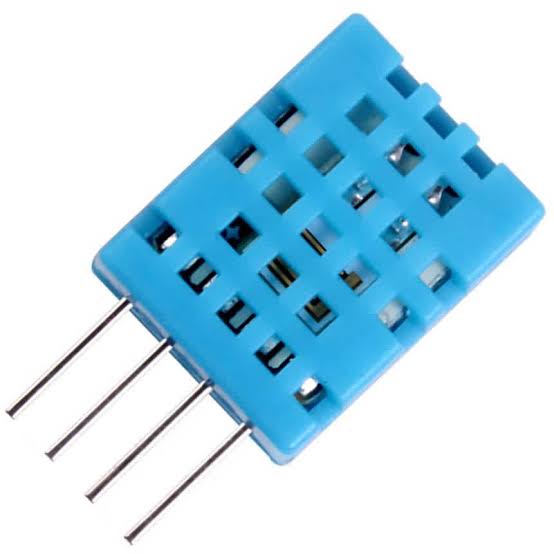
### **Features**

1. Complete with PIR, Motion Detection.
2. Dual Element Sensor with Low Noise and High Sensitivity.
3. Supply Voltage – 5V.
4. Delay Time Adjustable.
5. Standard TTL Output.

**5.3. DHT11 Sensors**

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 excel lent quality, fast response, anti-interference ability and cost-effectiveness.

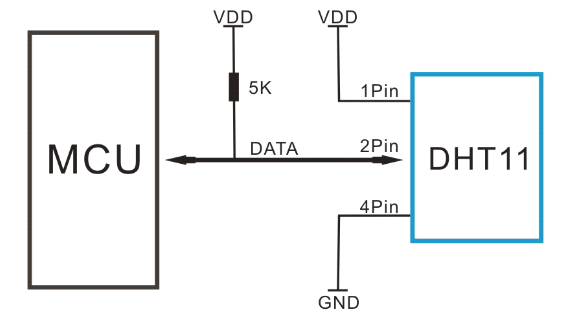
`

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-20meter 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 user’s request.

**Specification**

* Supply Voltage: +5 V
* Temperature range :0-50 °C error of ± 2 °C
* Humidity :20-90% RH ± 5% RH error
* Interface: Digital

**Application**



Note: Pin –Null; MCU = Micro-computer Unite or single chip Computer

When the connecting cable is shorter than 20 metres, a 5K pull-up resistor is recommended; when the connecting cable is longer than 20 metres, choose a appropriate pull-up resistor as needed.

**5.4. MQ-2 gas sensor**

Sensitive material of MQ-2 gas sensor is SnO2, which with lower conductivity in clean air. When the target combustible gas exists, the sensor’s conductivity is higher along with the gas concentration rising. Please use simple electronic circuit, convert change of conductivity to correspond output signal of gas concentration.

MQ-2 gas sensor has highly sensitive to LPG, Propane and Hydrogen, also could be used to Methane and other combustible steam, it is with low cost and suitable for different application.

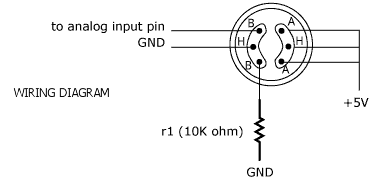
### **Features:**

* Operating Voltage is +5V
* Can be used to Measure or detect LPG, Alcohol, Propane, Hydrogen, CO and even methane
* Analog output voltage: 0V to 5V
* Digital Output Voltage: 0V or 5V (TTL Logic)
* Preheat duration 20 seconds
* Can be used as a Digital or analog sensor
* The Sensitivity of Digital pin can be varied using the potentiometer.



### **How to use the MQ-2 sensor to measure PPM:**

If you are looking for some accuracy with your readings then measuring the PPM would be the best way to go with it. It can also help you to distinguish one gas from another. So, to measure PPM you can directly use a module. A basic wiring for the sensor from datasheet is shown below.



**5.5. 6 - Channel Relay Board.**

This is a LOW Level 5V 6-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller.

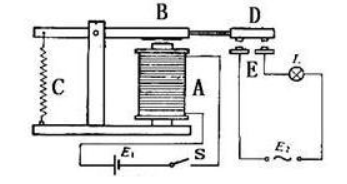


**Specification**

* Relay Maximum output: DC 30V/10A, AC 250V/10A
* .6Channel Relay Module with Opto-coupler. LOW Level Trigger expansion board, which is compatible with Arduino control board.
* Standard interface that can be controlled directly by microcontroller (Nodemcu, 8051, AVR, \*PIC, DSP, ARM, ARM, MSP430, TTL logic, etc.).
* Relay of high-quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.
* Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller.

**Operating Principle**

See the picture below: A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is on.



Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So, the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts result in power on and off of the circuit.

**Input**

**VCC:** Connected to positive supply voltage (supply power according to relay voltage).

**GND:** Connected to supply ground.

**IN1:** Signal triggering terminal 1 of relay module.

**IN2:** Signal triggering terminal 2 of relay module.

**IN3:** Signal triggering terminal 3of relay module.

**IN4:** Signal triggering terminal 4of relay module.

**IN5:** Signal triggering terminal 5of relay module.

**IN6:** Signal triggering terminal6of relay module.

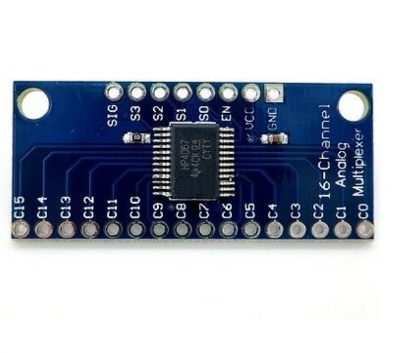
**Output:**

Each module of the relay has one NC (normally close), one NO (normally open) and one COM(Common) terminal. So, there are 8NC, 8NO and 8COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

**5.6. 16 Channel Analog/Digital Multiplexer.**

The CD74HC4067 and CD74HCT4067 devices are digitally controlled analog switches that utilize silicon-gate CMOS technology to achieve operating speeds similar to LSTTL, with the low power consumption of standard CMOS integrated circuits.

These analog multiplexers/demultiplexers control analog voltages that may vary across the voltage supply range. They are bidirectional switches thus allowing any analog input to be used as an output and vice-versa. The switches have low "on" resistance and low "off" leakages. In addition, these devices have an enable control which when high will disable all switches to their "off" state.



## **Features**

* Wide Analog Input Voltage Range
* Low "ON" Resistance
* Fast Switching and Propagation Speeds
* Wide Operating Temperature Range . . . -55°C to 125°C
* Balanced Propagation Delay and Transition Times

**Applications**

* Analog multiplexing and demultiplexing
* Digital multiplexing and demultiplexing
* Connect 16 sensors or analog devices to a microcontroller which has a single analog pin only.

**Platform Used**

**6. Platform Used**

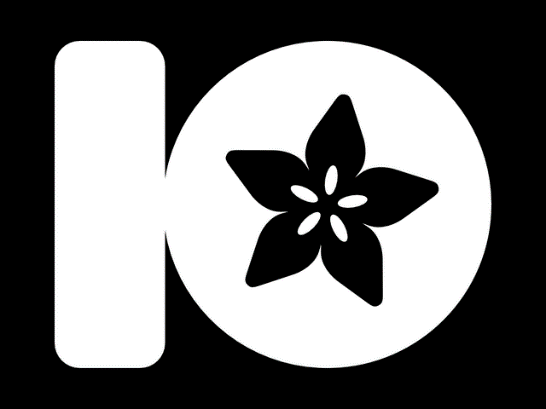
**6.1. AdafruitIo**

Adafruit Industries is an open-source hardware company based in New York City. It was founded by Limor Fried in 2005. The company designs, manufactures and sells a number of electronics products, electronics components, tools and accessories. It also produces a number of learning resources, including live and recorded videos related to electronics, technology, and programming MQTT, or message queue telemetry transport, is a protocol for device communication that Adafruit IO supports. Using a MQTT library or client you can publish and subscribe to a feed to send and receive feed data.

Adafruit IO is a system that makes data useful. Our focus is on ease of use, and allowing simple data connections with little programming required.

IO includes client libraries that wrap our REST and MQTT APIs. IO is built on Ruby on Rails, and Node.js.

[Adafruit IO](https://io.adafruit.com/welcome) is available and **free** for anyone with an Adafruit account. For the vast number of people starting out with IoT, the basic account will do everything they need. Adafruit IO [plus](https://io.adafruit.com/plus) members get some nifty extras and more bandwidth.

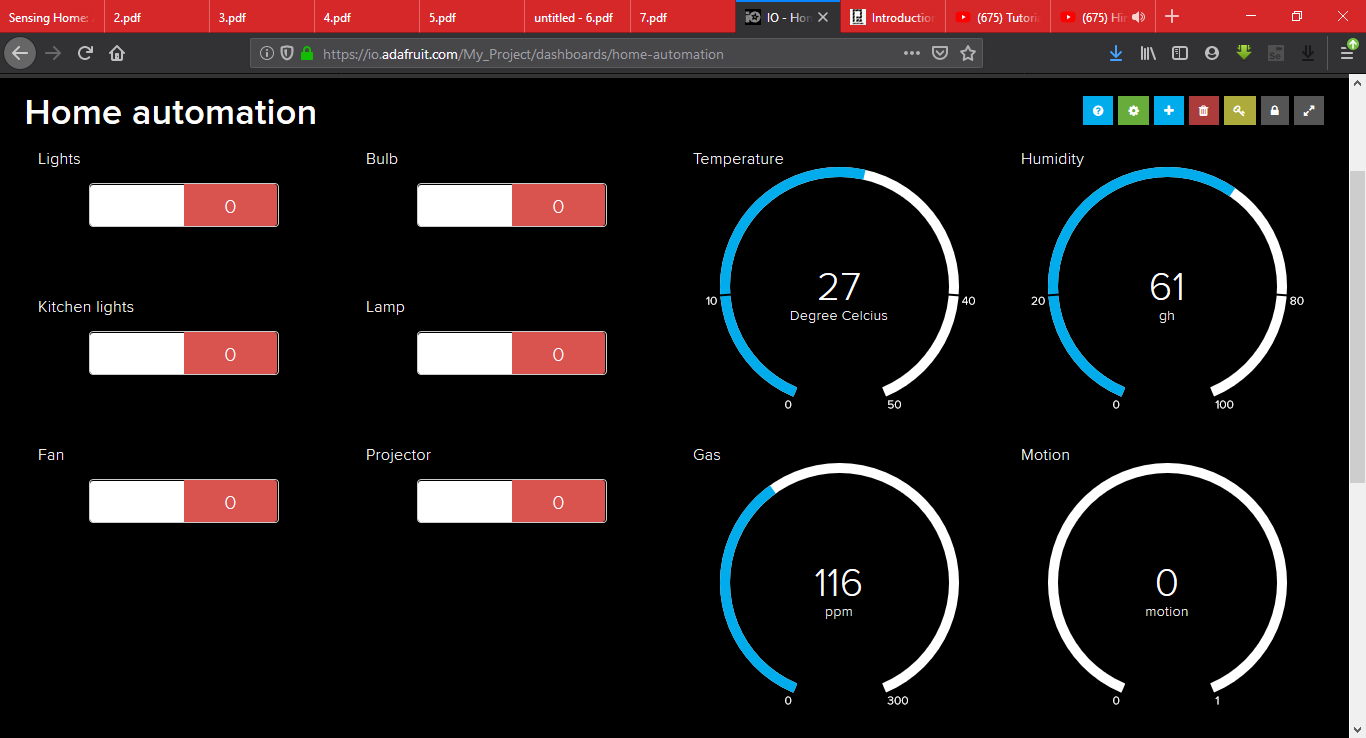


**Applications**

* Display your data in real-time, online
* Make your project internet-connected: Control relays, read sensor data, and more!
* Connect projects to web services like Google assistant, Twitter, RSS feeds, weather services, etc.
* Connect your project to other internet-enabled devices
* The best part? All of the above is do-able for **free**with Adafruit IO

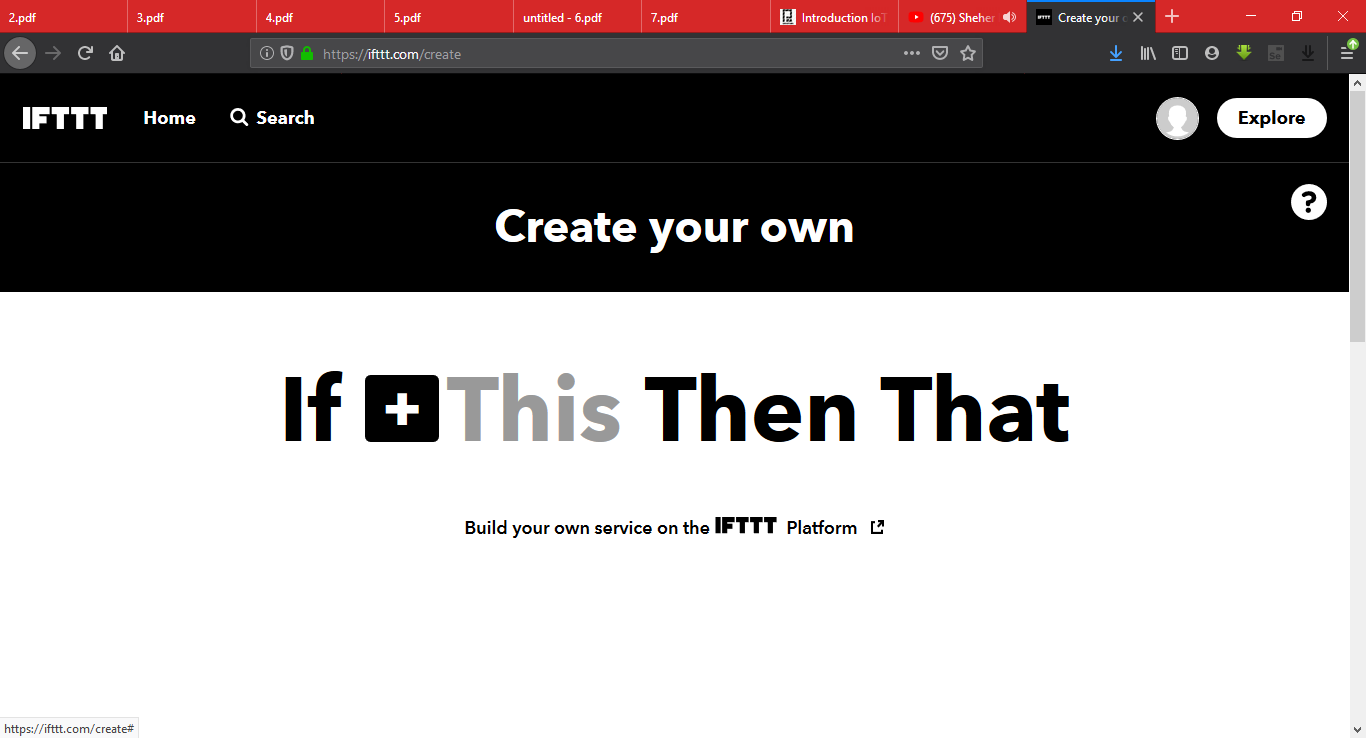
Devices can also be controlled by AdafruitIo dashboard which was created by Me.

My Adafruitio dashboard from where I can monitor the sensors and turn on and off the devices



**6.2. IFTTT**

IFTTT derives its name from the programming conditional statement if this, then that. IFTTT is both a website and a mobile app that launched in 2010 and has the slogan “Put the Internet to work for you”. The idea is that you use IFTTT to automate everything from your favourite apps and websites to app -enabled accessories and smart devices. What the company provides is a software platform that connects apps, devices and services from different developers in order to trigger one or more automations involving those apps, devices and services. Here, IFTTT application is used to bridge the gap between the Google Assistant commands and the Smart phone. Setting up the IFTTT application first requires logging in after which we need to create an applet and then This, i. e. the trigger, 16 here we select Google Assistant and then we will type in the commands to which the Google Assistant should respond and to this command it should control the appliance/relay associated with it. The response command from the Goggle Assistant can also be typed in as desired. After configuring the trigger, i.e. This of the application we need to configure the That. What should be done once the Google Assistant hears the command which we just configured? This is decided by setting That of the app. communication is done via the internet and since the microcontroller, NodeMCU comes with inbuilt Wi -Fi module, it is programmed to connect to the desired network once plugged in. C language is used to program the microcontroller and is programmed in the Arduino IDE.



**6.3. Google Assistant**

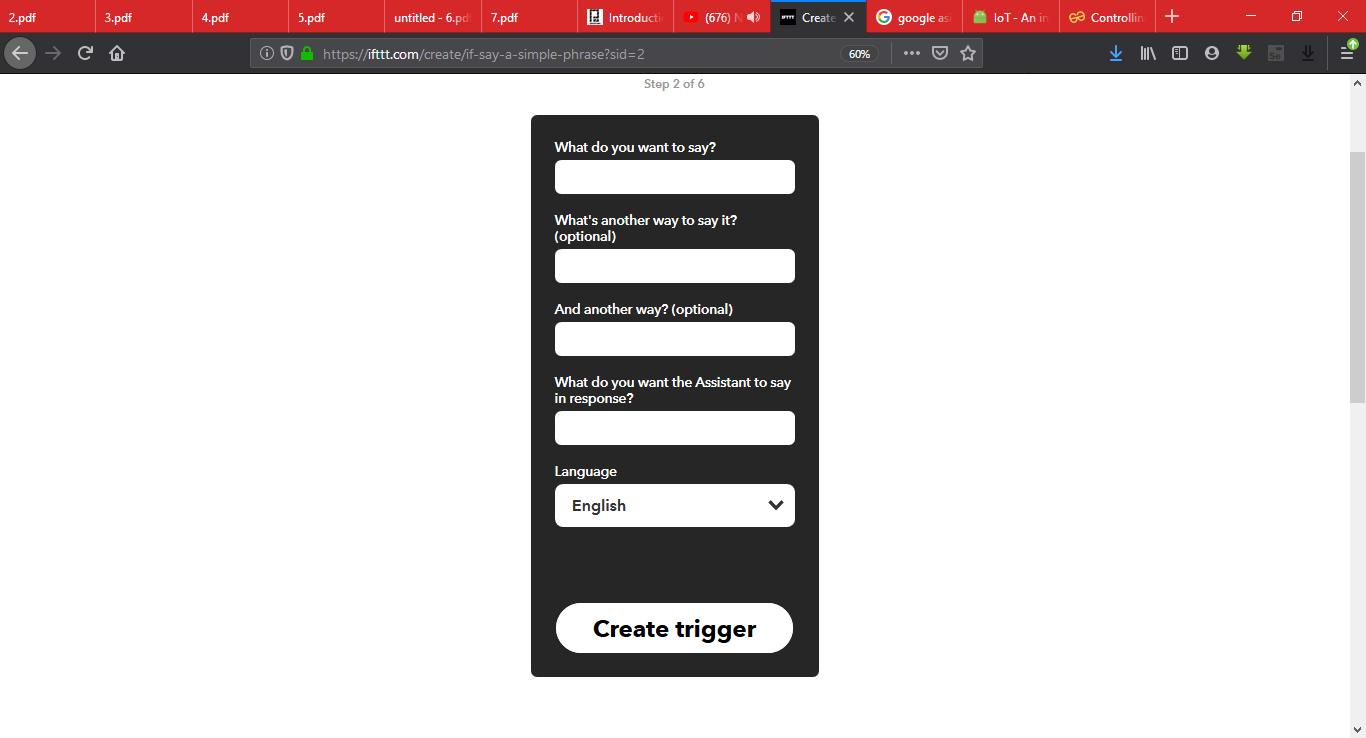
**Google Assistant** is an [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence)-powered [virtual assistant](https://en.wikipedia.org/wiki/Virtual_assistant) developed by [Google](https://en.wikipedia.org/wiki/Google) that is primarily available on [mobile](https://en.wikipedia.org/wiki/Mobile_device) and [smart home](https://en.wikipedia.org/wiki/Home_automation) devices. Unlike the company's previous virtual assistant, [Google Now](https://en.wikipedia.org/wiki/Google_Now), the Google Assistant can engage in two-way conversations.

All the Android Devices have Google Assistant in built in it. But we cannot control any devices or anything using our own phrases, to do so we have to create our own google assistant phrases from IFTT website or Application.

To create our own command, go to IFTT register our self and create applet,

Click on ‘this’**->** search for google assistance -> click on google assistance

-> Click on simple phrases -> then fill the page and completed.



**Implementation**

**7. Implementation**

For Getting started Power up the Nodemcu by just plugging in the Micro USB to the board and connect the Nodemcu with Computer or a laptop by using USB. The next step is to install the and configure the software named “Arduino IDE”

**7.1. Getting Nodemcu ready**

### **Step 1: Connect your NodeMCU to your computer**

You need a USB micro B cable to connect the board. Once you plugged it in, a blue LED will start flashing. If your computer is not able to detect the NodeMCU board, you may need to download the driver of our board.

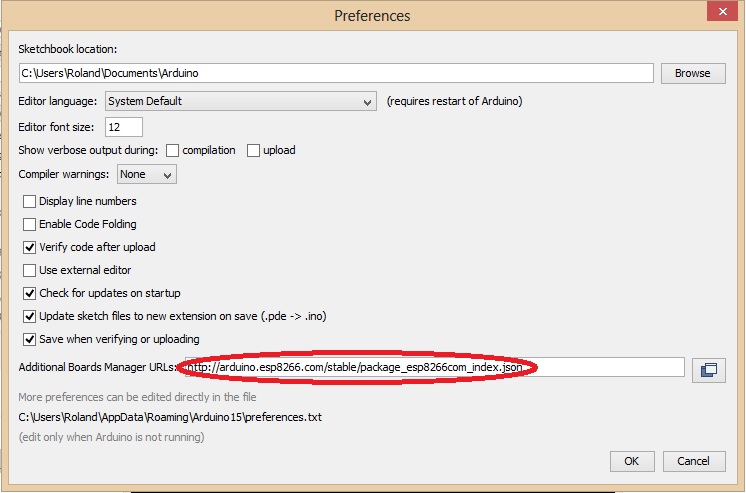
### **Step 2: Open Arduino IDE**

You need to have at least Arduino IDE version 1.6.4 to proceed with this.

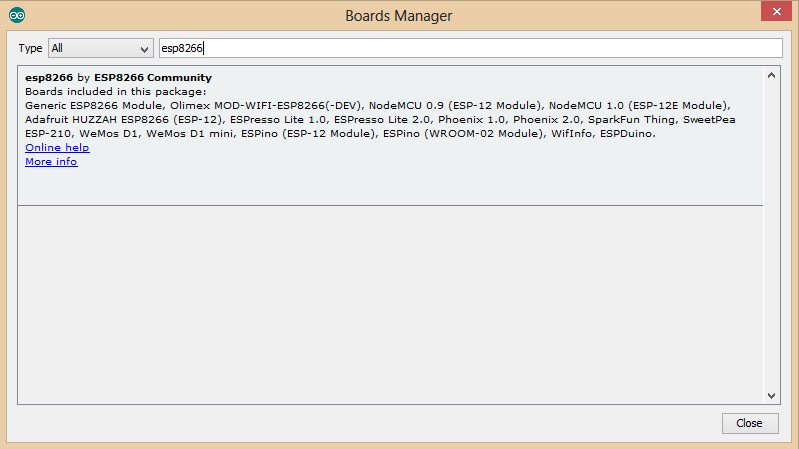
Go to File > Preferences. In the "Additional Boards Manager URLs" field, type (or copy paste) the following link:

<http://arduino.esp8266.com/stable/package_esp8266com_index.json>.

 Don't forget to click OK!



Then go to Tools > Board > Board Manager. Type "esp8266" in the search field. The entry "esp8266 by ESP8266 Community" should appear. Click that entry and look for the install button on the lower right.



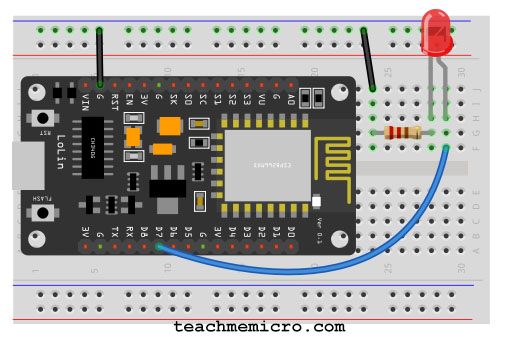
Google assistant is AI (Artificial Intelligence) based voice command service. Using voice, we can interact with google assistant and it can search on the internet, schedule events, set alarms, control appliances, etc.

Once the download is complete, you can start coding!

### **Step 3: Make a LED blink using NodeMCU**

For our first program, we'll blink a LED connected to one of the digital pins of the board. But before that, you need to know that the pin names printed on the board are not the pin names we'll be using for our program. BTW, I'm using NodeMCU V1.0.

In this example, I'll connect the LED to D7 as it is printed on the board. D7 is GPIO13 according to the image above. You can follow the wiring diagram below:



And so, here's my code (which is basically the example Blink code in Arduino):

**Code for blinking an LED to check the configuration is proper or not**

//the setup function runs once when you press reset or power the board

void setup() {

  // initialize digital pin 13 as an output.

  pinMode(13, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)

  delay(1000);              // wait for a second

  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW

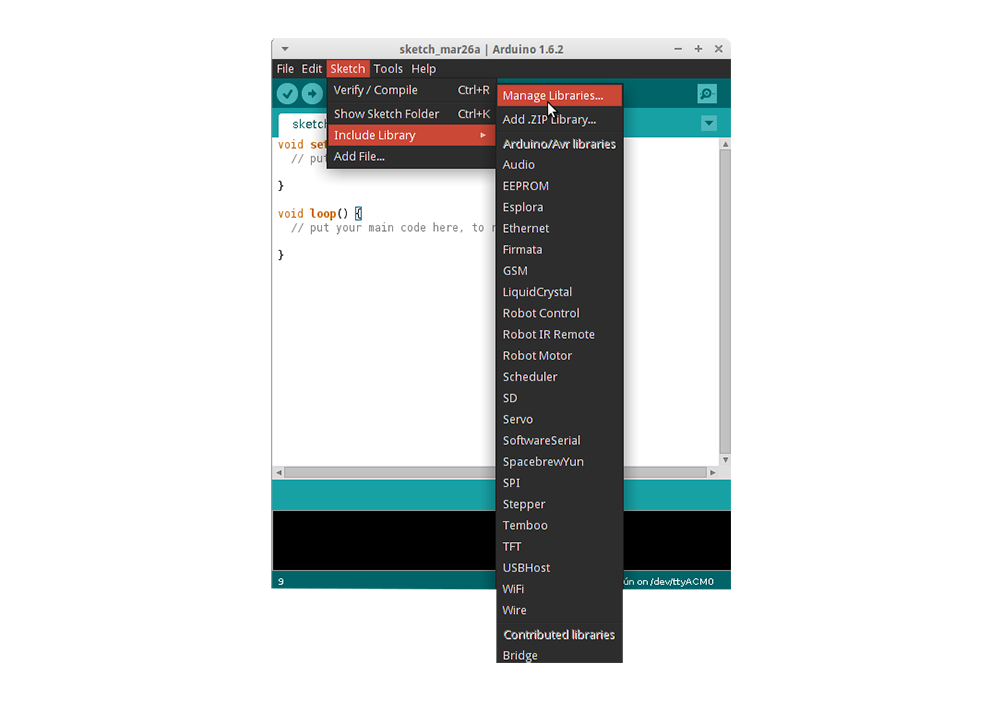
  delay(1000);              // wait for a second

}

Click the upload button and wait for the "Done Uploading" text to appea

**7.1.1. Installing Libraries for AdafruitIo & Dht11 sensor.**

To install a new library into your Arduino IDE you can use the Library Manager (available from IDE version 1.6.2). Open the IDE and click to the "Sketch" menu and then Include Library > Manage Libraries.



Then the Library Manager will open and you will find a list of libraries that are already installed or ready for installation. In this example we will install the Bridge library. Scroll the list to find it, click on it, then select the version of the library you want to install. Sometimes only one version of the library is available. If the version selection menu does not appear, don't worry: it is normal

Search for the AdafruitIo and install the library by just clicking on install & also search for Dht11 and install the latest library.

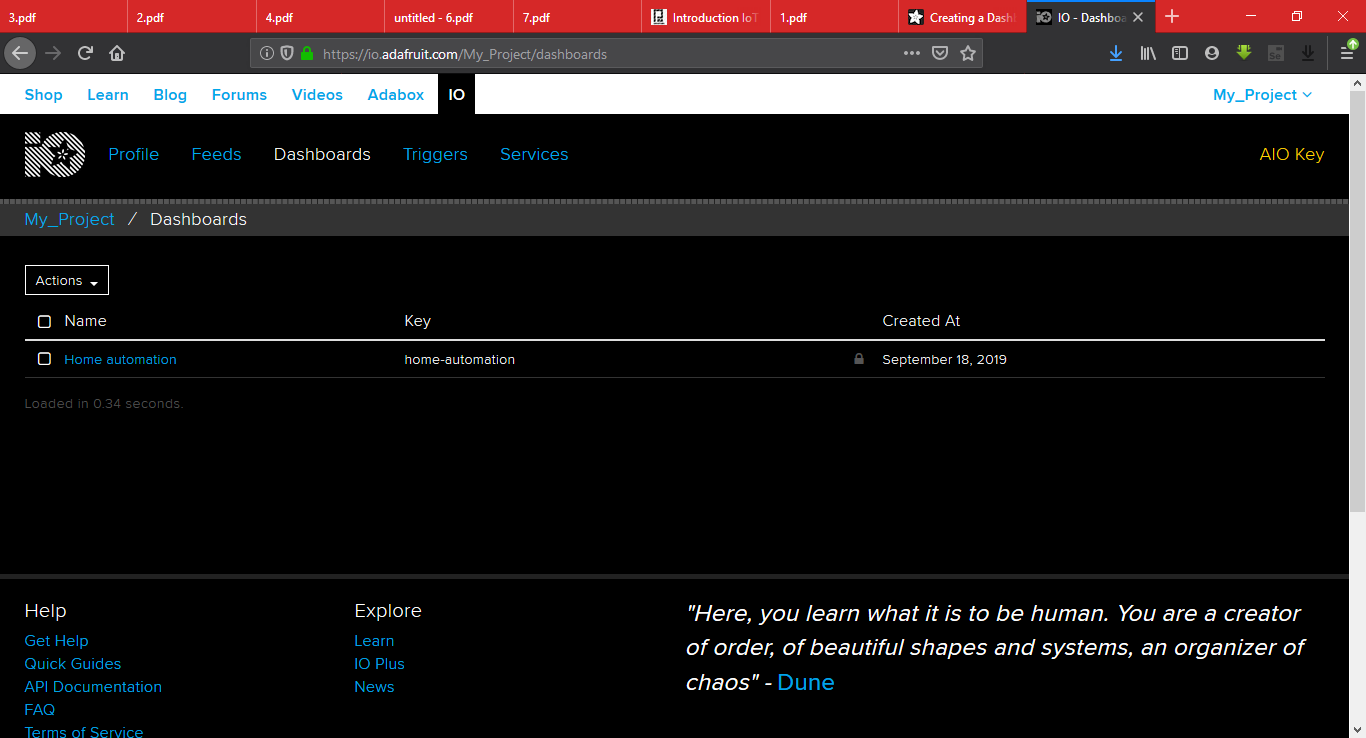
Now, Our Nodemcu is almost ready for uploading code and also for further process.

**7.2. Configure AdafruitIo**

**7.2.1. Creating a dashboard**

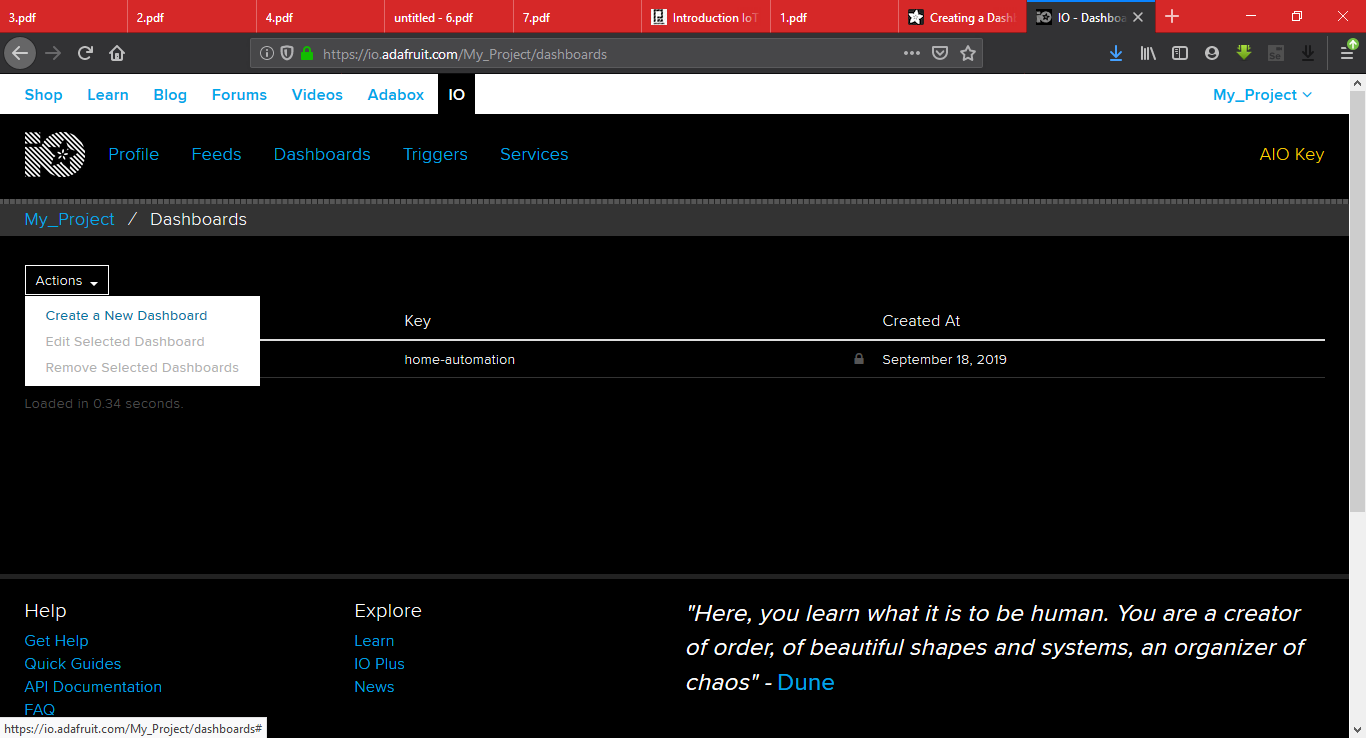
Before Creating a dashboard login into AdafruitIo website,

When you, login to your [io.adafruit.com](https://io.adafruit.com) account, you will be redirected to your list of dashboards. It will look like the page seen below.

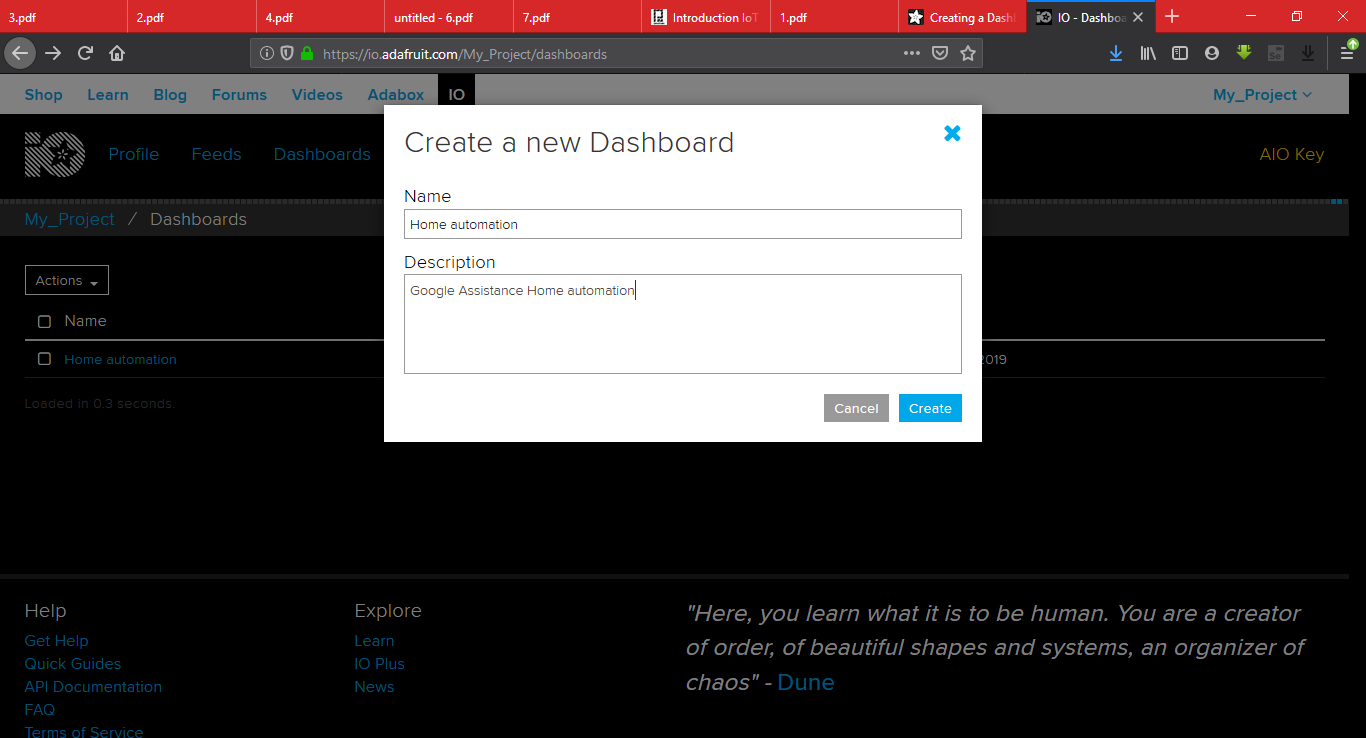


Your list of dashboards will only have the Welcome Dashboard when it is first loaded. You can start the dashboard creation process by clicking the **Actions** menu on the upper left-hand side of the screen.

Next, select **Create a New Dashboard** from the dropdown menu.

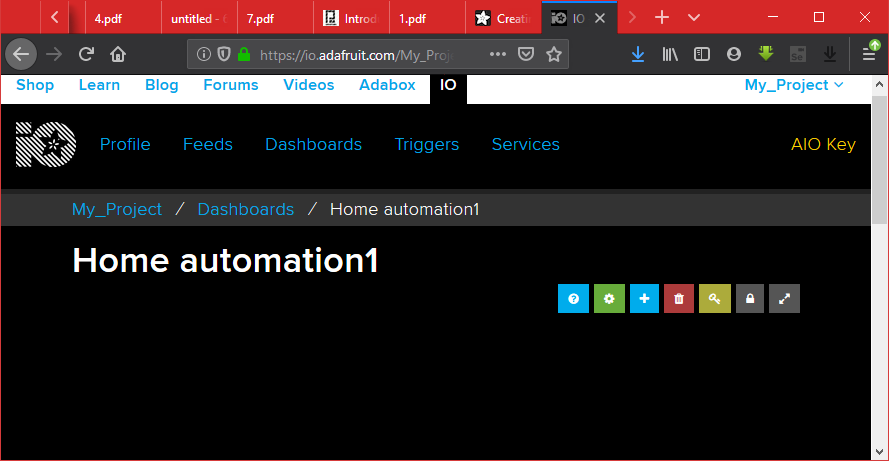


You can then enter the name and description of your new dashboard, and click the **Create** button once you are finished.



Once your dashboard has been created, click on the name of your new dashboard to load it.

You should now see your new blank dashboard.

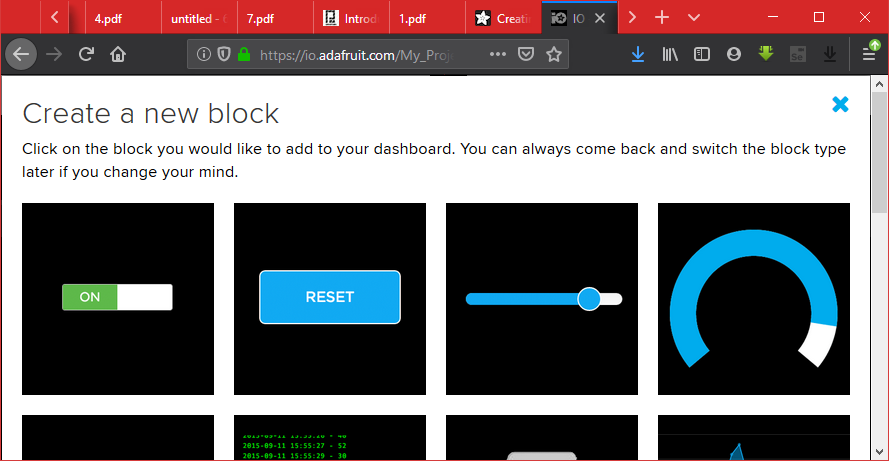


Next, we will look at adding blocks to the new dashboard.

**7.2.2. Adding Blocks and Feeds**

Blocks are widgets that you can add to your dashboard. There are some blocks that can be used as outputs, and some that can be used as inputs. To add a new block, you can click the **+** (plus) button on the upper right-hand side of the dashboard.

You will then be presented with a list of block types to choose from, like the one seen below.



Select toogle button to create one toggle button

## **Toggle Button**

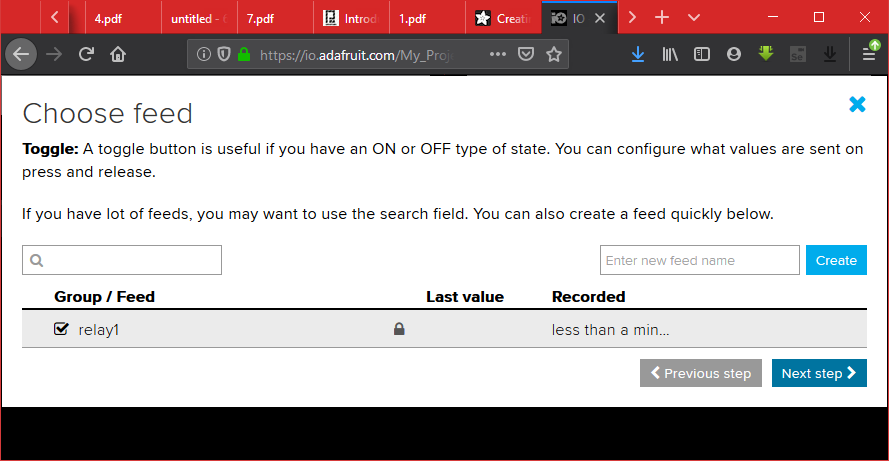
The toggle button will allow you to switch between any two text or numeric values. Unlike the momentary button, the values will stay the same until you click the button again to toggle to the second value.

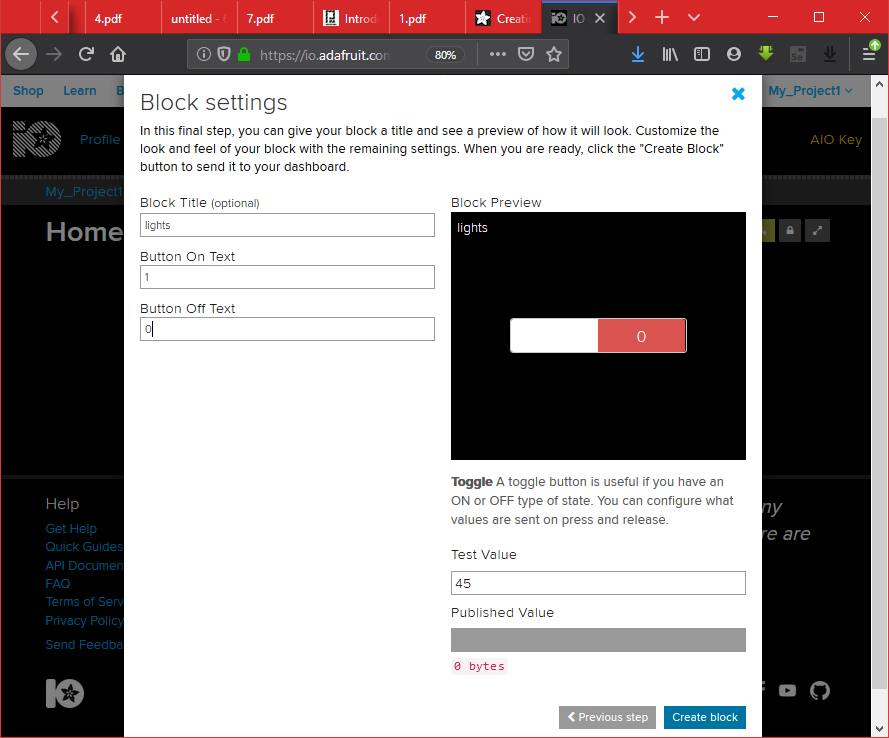
Create one feed named as relay1 or any other name you want to add it just click on create and then the feed is created



After creating the feed select the and Click on Next step.



After creating feed, then add the Block title as lights, and also change the button on text as ‘1’ and button off text as ‘0’ as shown in the picture below,



After adding the block title, button on text, button off text just click on create block to create a toggle switch.

Repeat the same steps to create 5 More toggle switch’s and their feed name as named as relay2, relay3, relay4, relay5, relay6 and their block name as same as Kitchen lights, fan, bulb, lamp, projector respectively and their button on text as 1 and button off text as 0.

Thus, we have created 6 toggle switches for light, Kitchen lights, fan, bulb, lamp, projector.

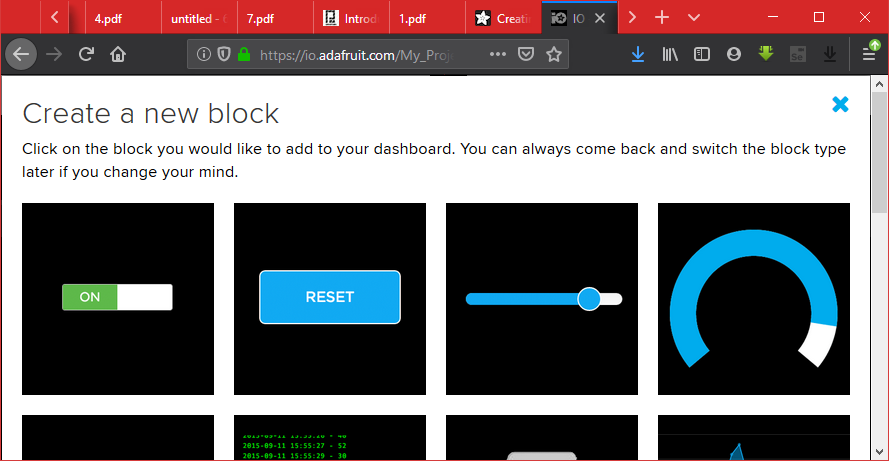
Now, we will create 4 blocks for all the three sensors.

Starting steps for creating all the block is almost same, just click on plus icon (‘+’) on right-hand side and then select a gauge instead of toggle switch

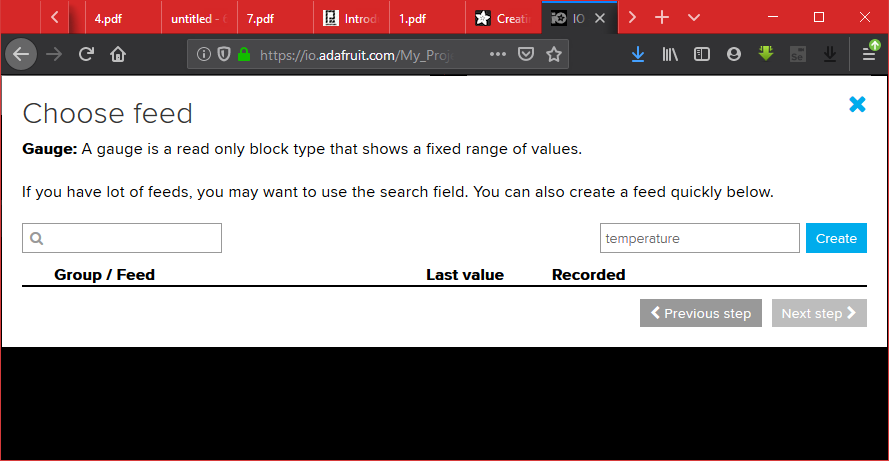
## **Gauge**

The gauge block allows you to quickly view the current value of a numeric feed. You can set a minimum and maximum value for the gauge, and it will automatically scale the value to a percentage and display it graphically. The gauge will update automatically whenever a new value is pushed to the feed.

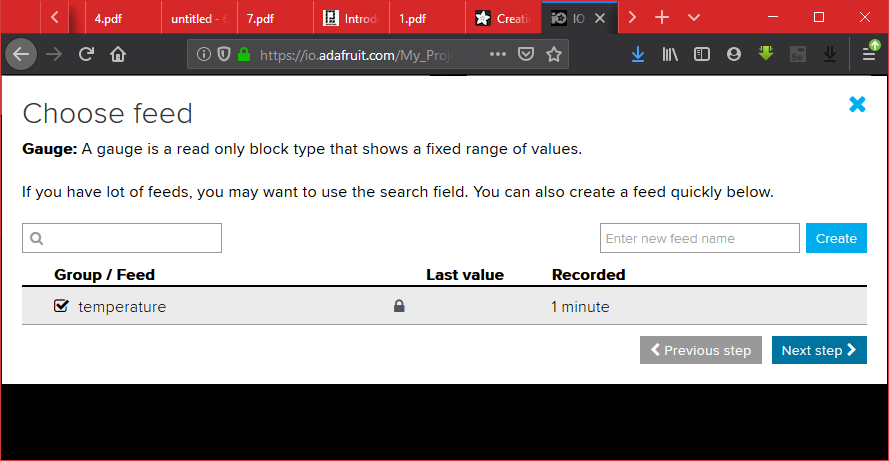
Select Gauge which is present on the righthand ,to create one Gauge indicator for Temperature indication which will be received the data from DHT11 sensor which is connected to the Nodemcu board.



Create one feed name it as temperature or any other name you want to add it just click on create and then the feed is created.

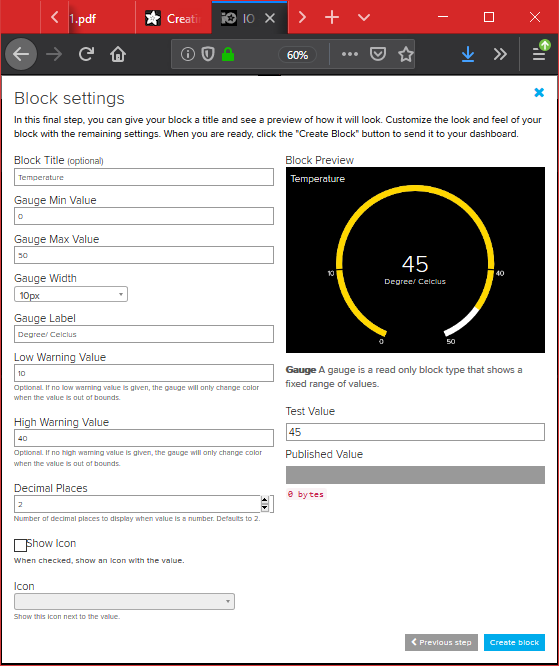


After creating the feed select the and Click on Next step.



After creating feed, then add the Block title as Temperature,change the Gauge min value as ‘0’ and Gauge max value as ‘50’, Gauge label as ‘Degree/Celcius’, low warning value as ‘10’ and High warning value as ‘40’ as shown in the picture below,

After adding all the necessary things click on Create block.



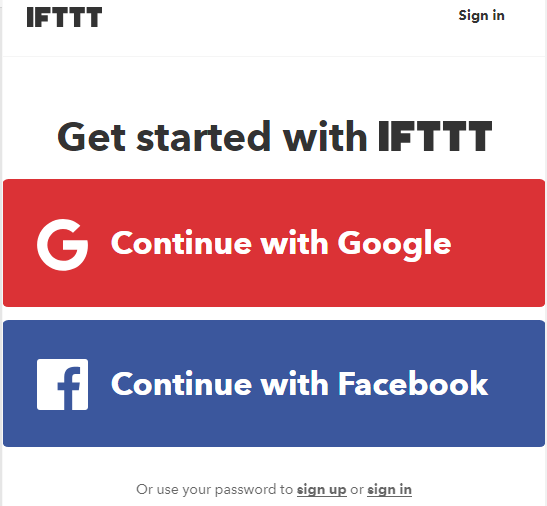
Repeat the same steps to create 3 more Gauge’s fill the data according to the table given below,

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Feed Name** | **Gauge Label** | **Gauge min value** | **Gauge max value** | **Gauge Width** | **Low Warning value** | **High Warning value** |
| Humidity | gh | 0 | 100 | 10 | 20 | 80 |
| Gas | Ppm | 0 | 300 | 10 | - | - |
| Motion | Motion | 0 | 1 | 10 | - | - |

Thus, we have created 4 gauge’s for Temperature, Humidity, Gas and Motion.

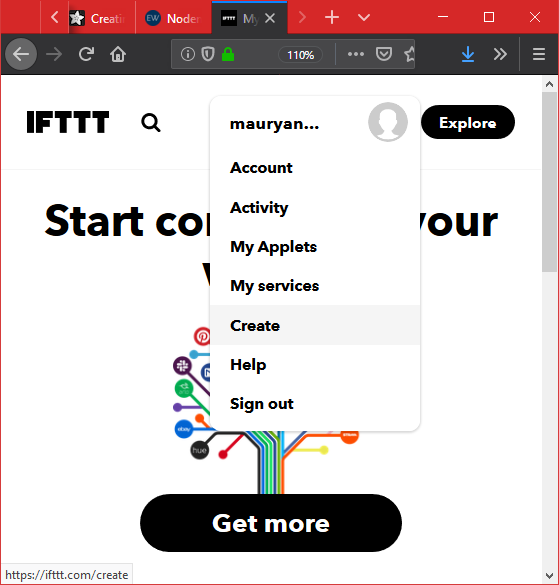
**7.3. Setting up IFTTT**

First step is creating account on IFTTT. ​​​​​

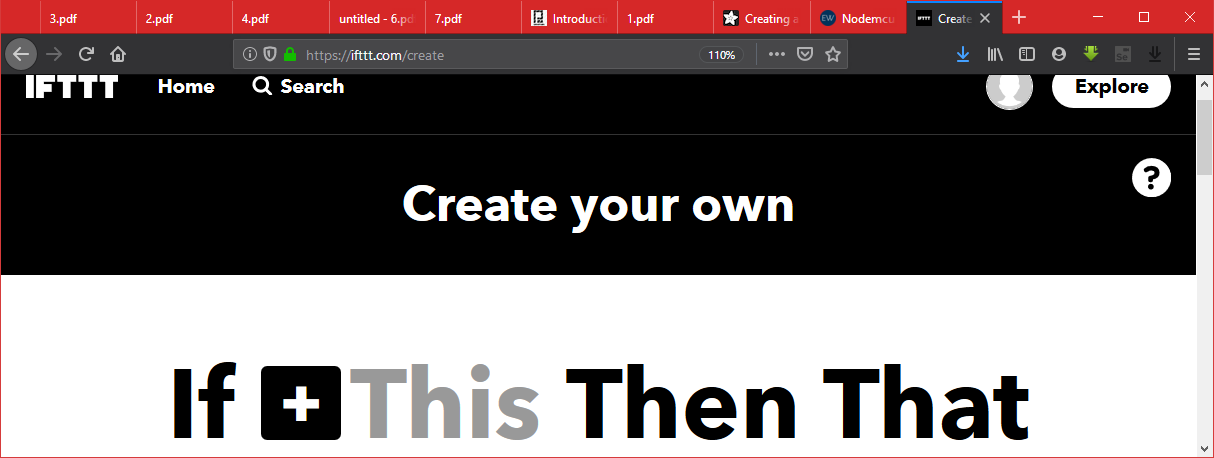


**Note:** Create account on IFTTT byusing same e-mail id which you have used for Adafruit.

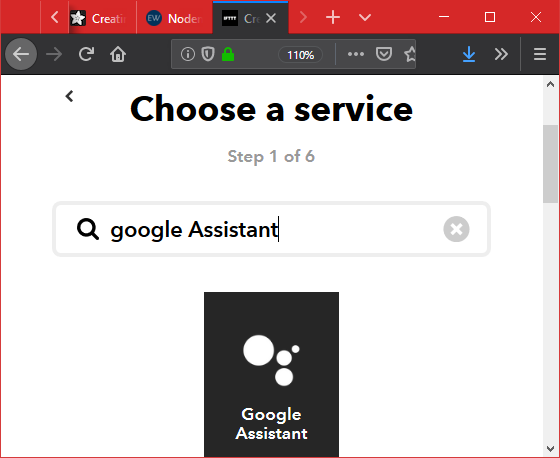
After account creation, on right-hand side there is a profile icon click on it then a drop down occurs, then click on create for creating a new applet.



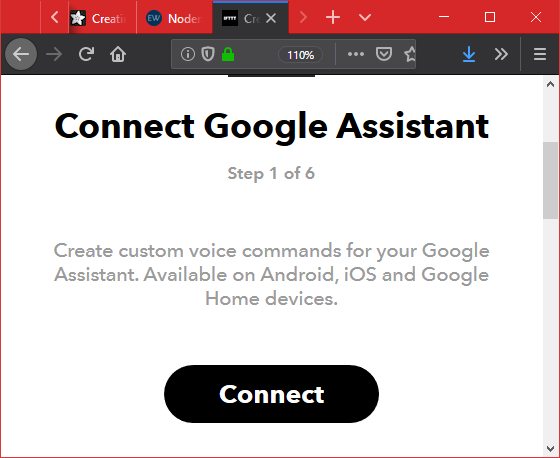
After selecting a new applet, we get a new page in which we should click on to **This** as shown in below image in the blue box.



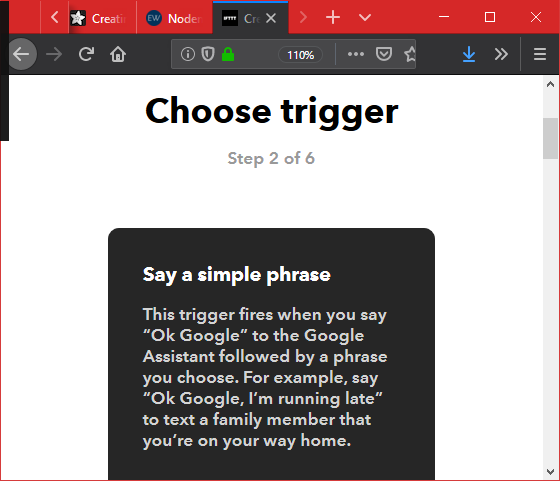
Then search for **Google Assistant** and select it.

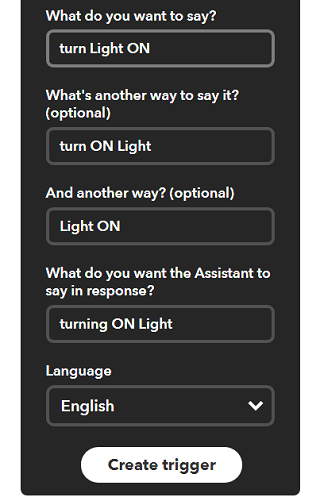


Then click on connect and select the account in which you have create the AdafruitIo blocks and also give the permission to it.



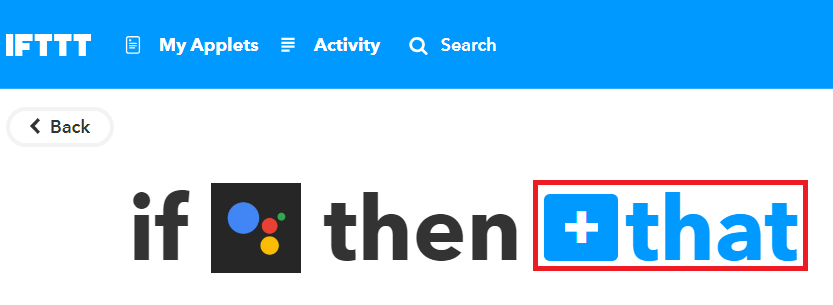
Click on say a simple phrase block.



Now, enter voice phrases which we will use as a command for google assistant. ​​​​​​​​​​​​

We can enter any phrase as per our application. As you can see, the phrases entered in the above fields is for making **Light ON.**

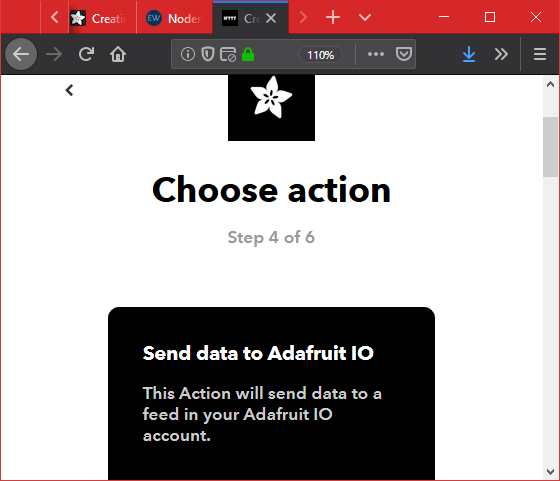
Now, we get another page on which we have to click on **that** option which is used to connect Google Assistant with AdafruitIo. ​​​​​​​​​​​​​​



Then search for **Adafruit** and select it and select it.

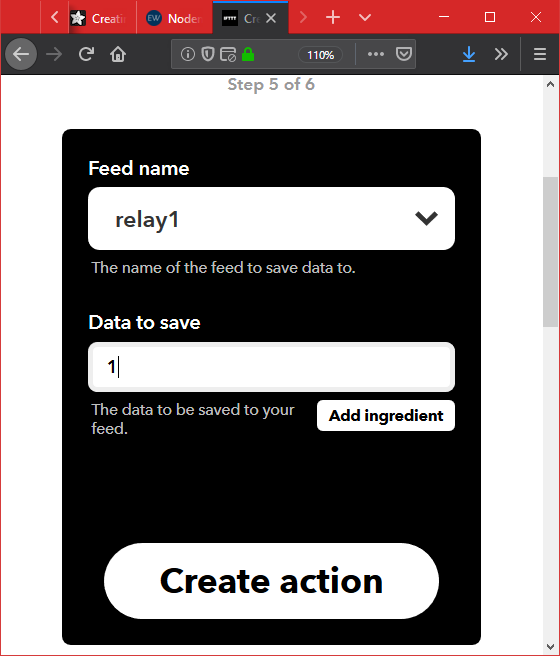


Click on send data to AdaFruitIo.



Now enter what data we need to send to which feed of Adafruit dashboard.

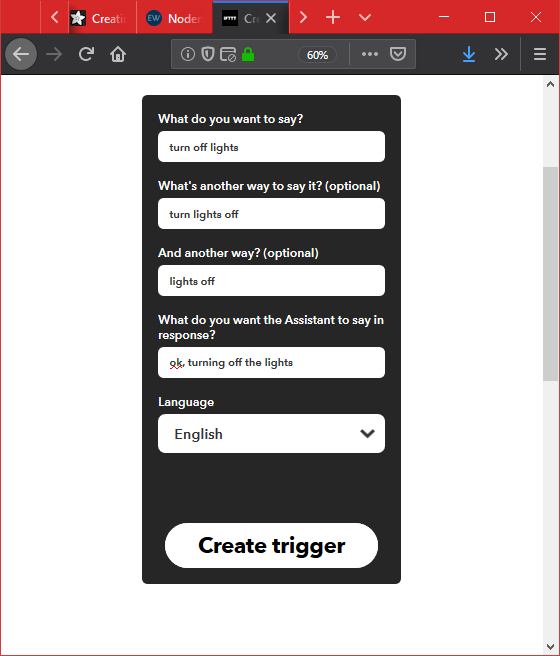
Select the feed name as relay one and enter 1 in Data to save textbox.



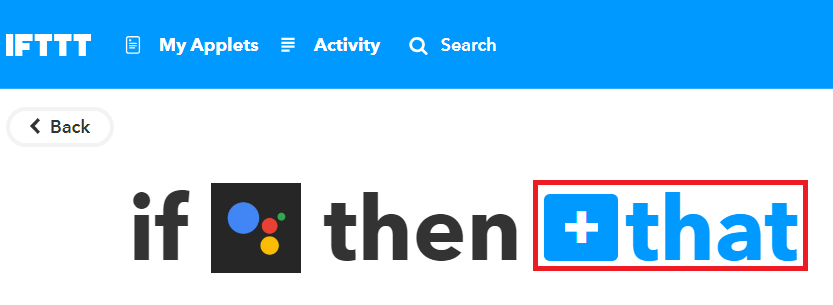
And then click on Create action and then click on Finish.

For making **Light OFF**, we have to create another applet with different phrases.

Repeat the same step from above and make changes in that,



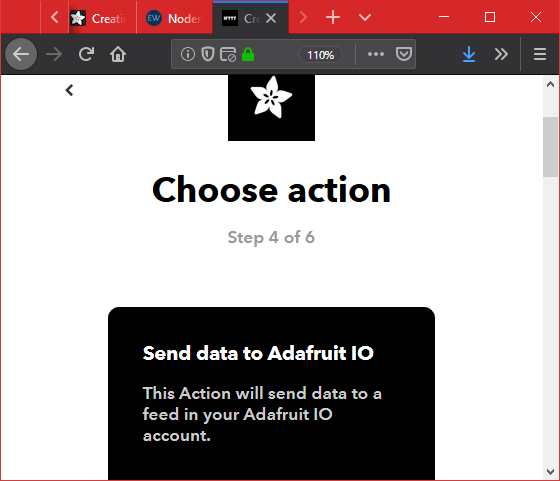
Click on Create trigger and the repeat the same step.



Then search for **Adafruit** and select it and select it.

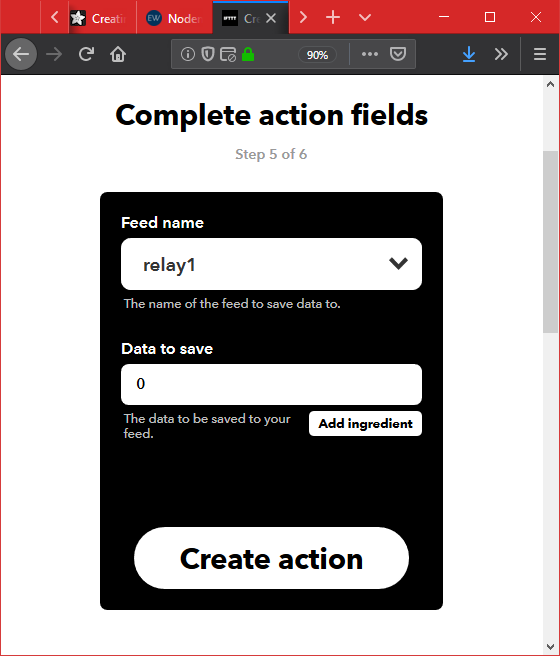


Click on send data to AdaFruitIo.



Now enter what data we need to send to which feed of Adafruit dashboard.

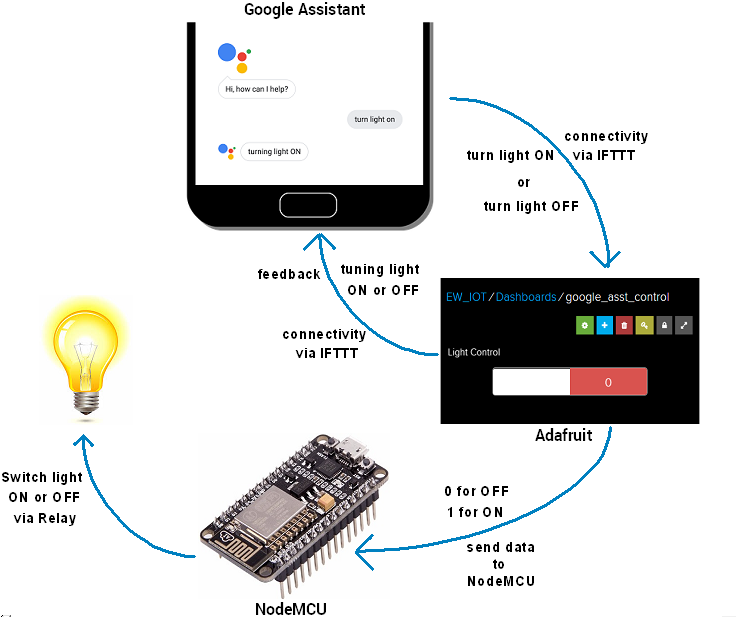
Select the feed name as relay one and enter 0 in Data to save textbox.



And then click on Create action and then click on Finish.

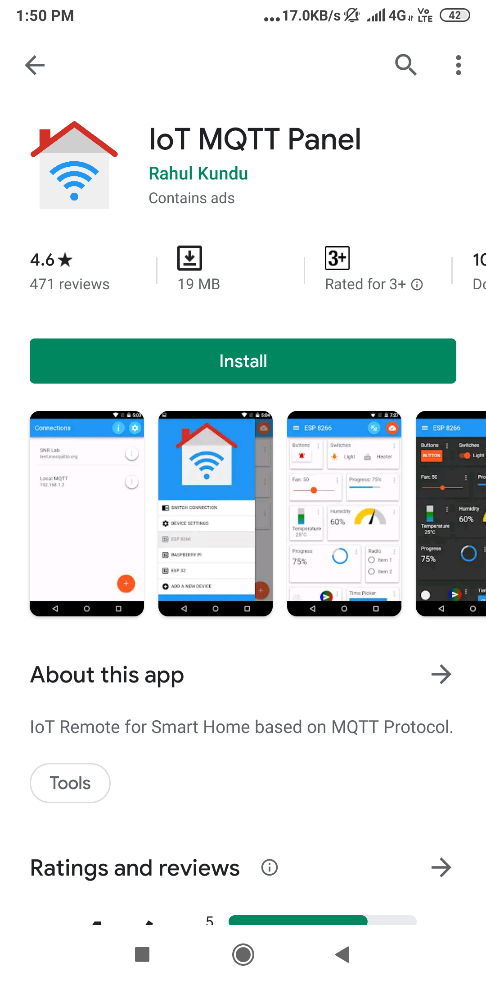
Now, repeat both the things for relay2, relay3, relay4, relay5, relay6 and make both on and off actions and applets for the devices like kitchen lights, fan, bulb, lamp, projector respectively.

After creating dashboard, feeds, blocks and also setting up IFTT with google assistance and AdafruitIo. The next step is Setting up IOT MQTT Panel application.

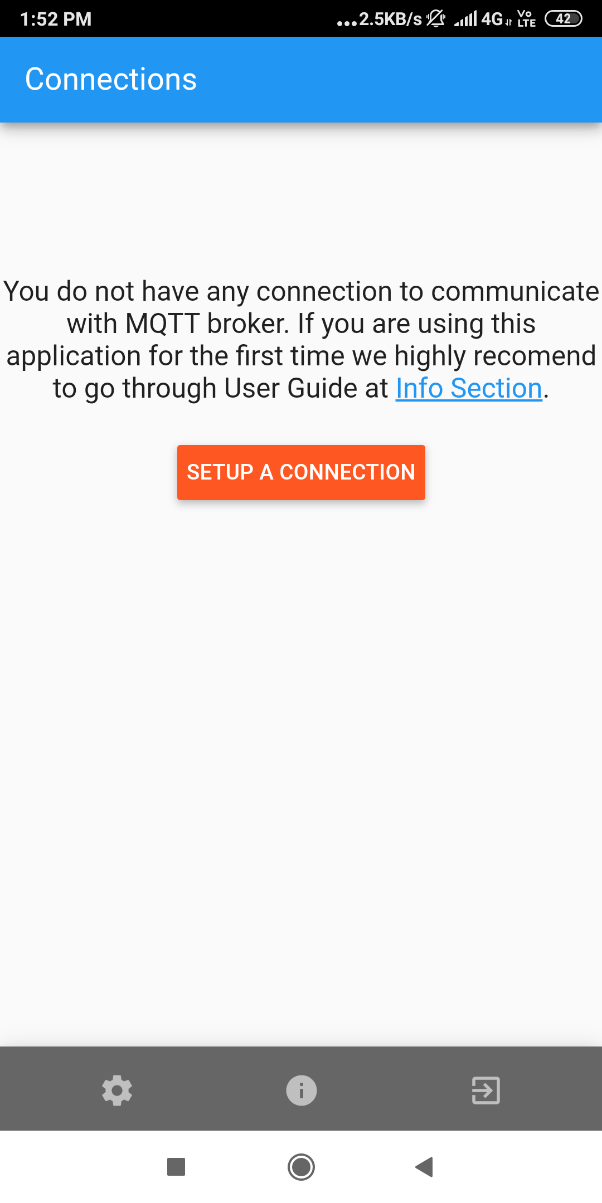


**7.4. Setting up IOT MQTT Panel application and WIFI**

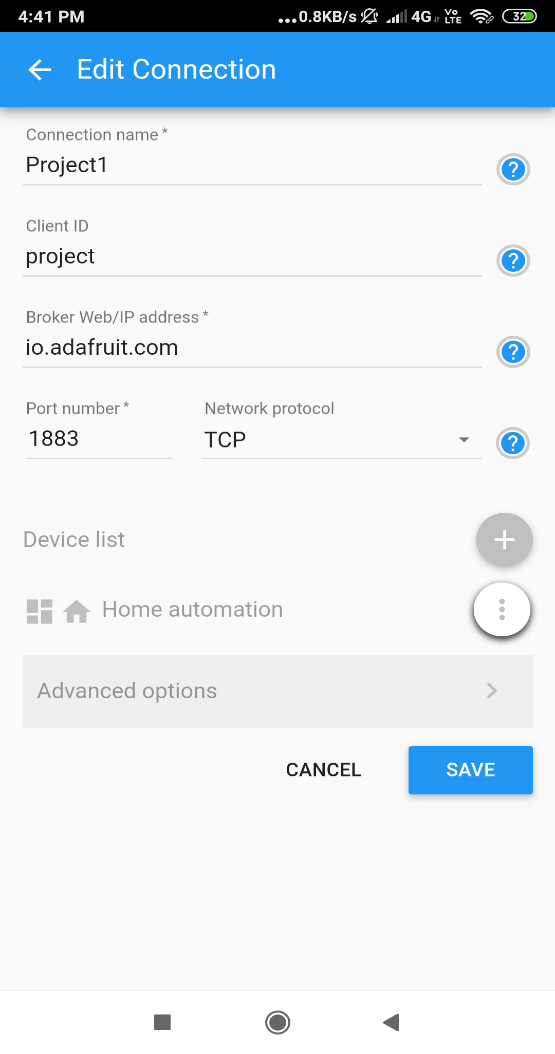
For setting up the app, go to play store on any android phone, Search for IOT MQTT panel and install the app.



Open the app and create the connection setup.

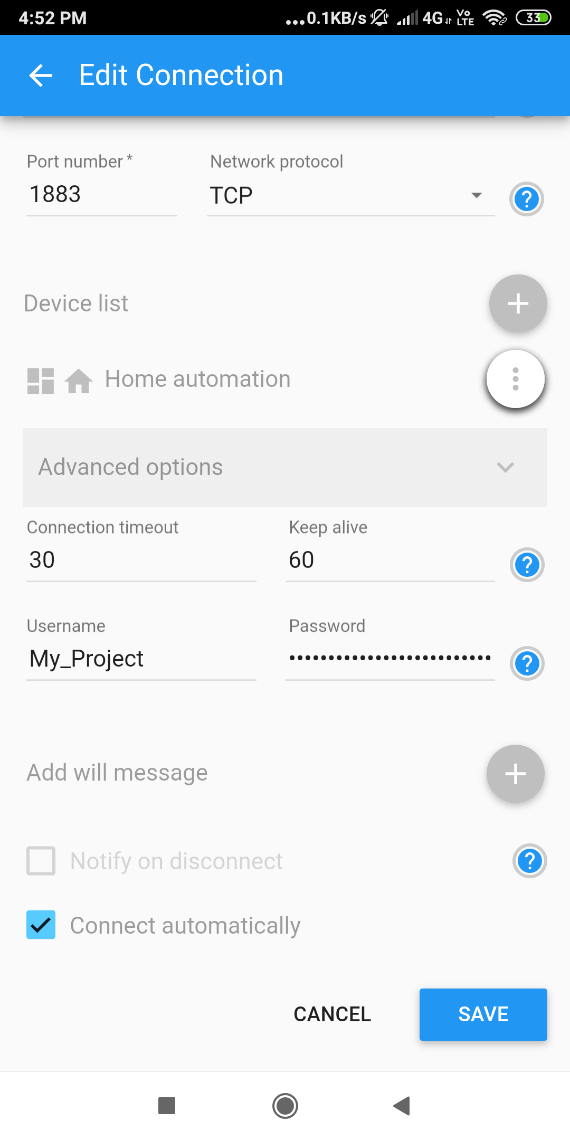


Adding a connection, and fill the data and save it and according the figure given below,

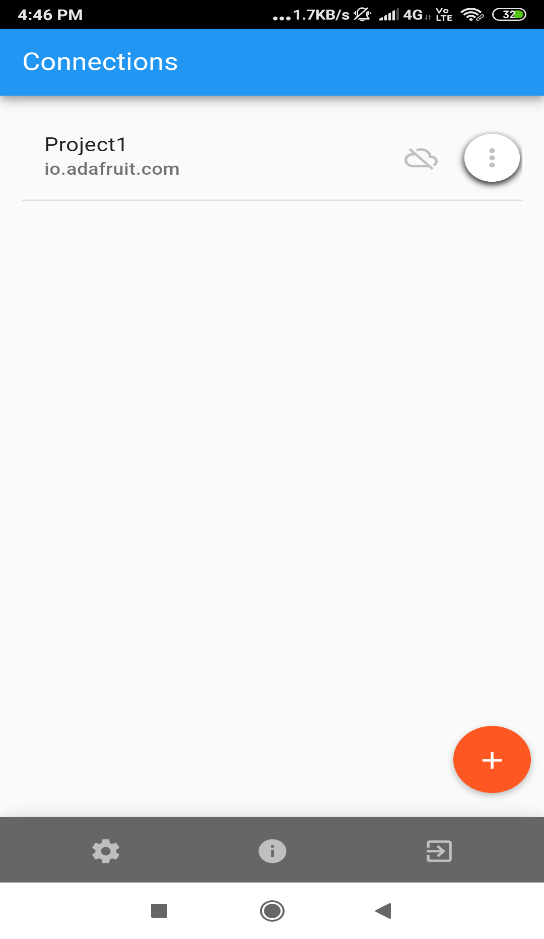


we have to setup the connection with our AdafruitIo by clicking on advance option’s & fill the form and click on save as given below

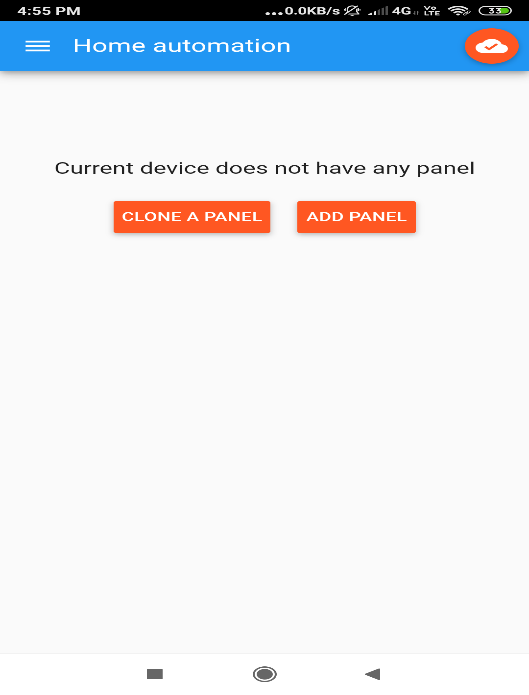
note: the username and password (auth key) is from our AdafruitIo dashboard.



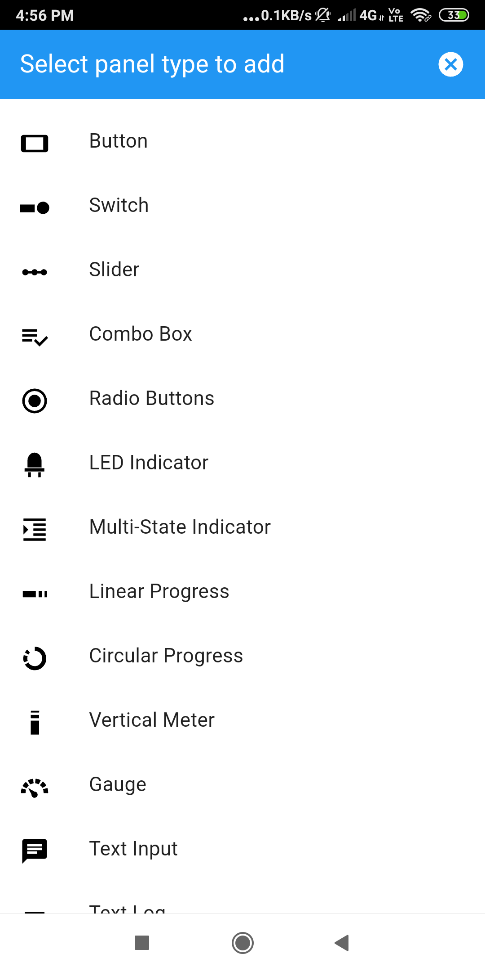
Click on the cloud symbol to connect with the MQTT server



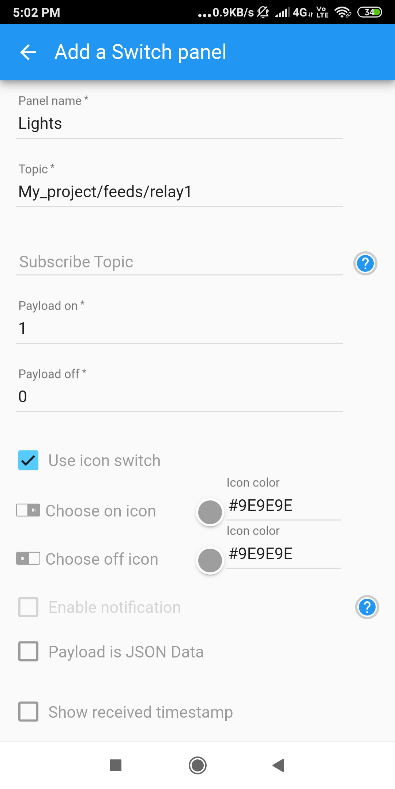
Click on add panel

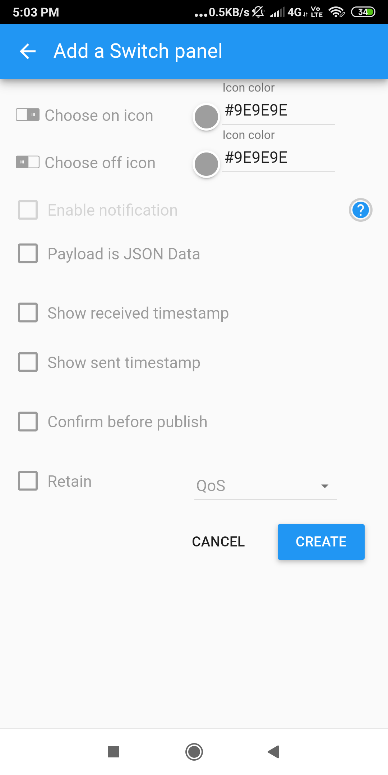


Click on Switch option to create a switch.



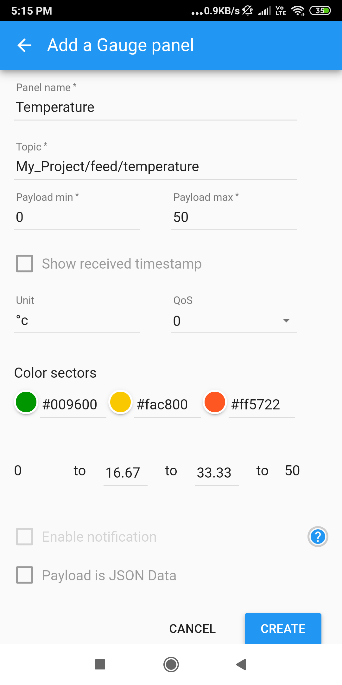
Add all the data for a swich where topic is a common name of the device, Topic is the fath of the feed of that device, add Payload On data as 1 and Payload Off data as 0 and click on create button





After creating a single switch, create 5 more switches. Panel name it as kitchen lights, fan, bulb, lamp, projector and change the feed name and ad payload on to 1 and payload off to 0

Now add a Gauge by clicking on select panel to add and fill the data as given below and click on save in the diagram.

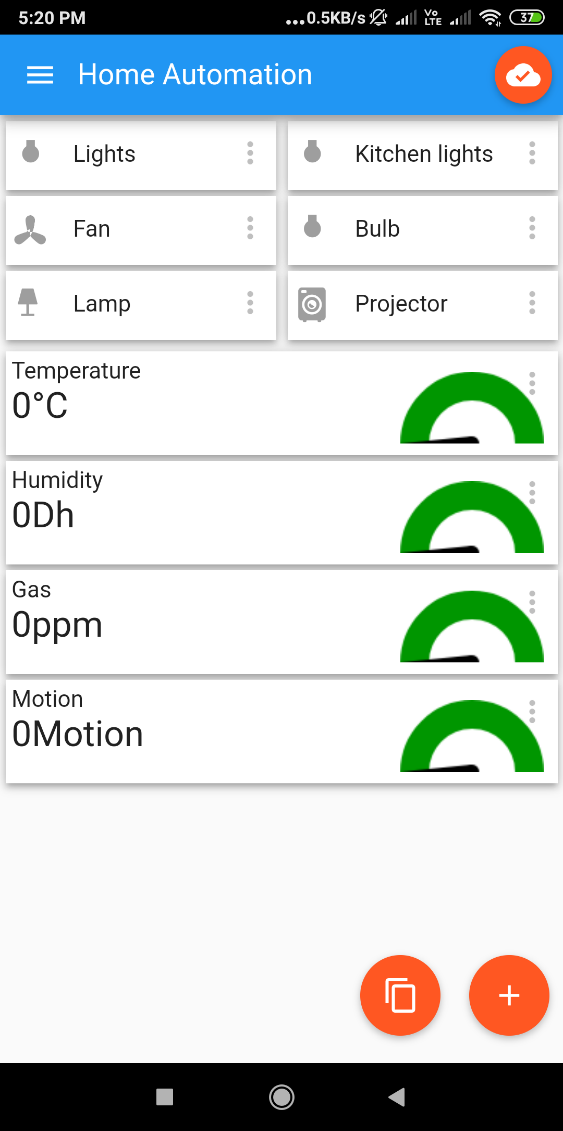


Repeat the same steps to create 3 more Gauge’s fill the data according to the table given below,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Panel name** | **Topic** | **Payload min** | **Payload max** | **Low Warning value** | **High Warning value** |
| Humidity | /humidity | 0 | 100 | 20 | 80 |
| Gas | /co2 | 0 | 300 | - | - |
| Motion | /motion | 0 | 1 | - | - |

Thus, we have created 4 gauge’s for Temperature, Humidity, Gas and Motion.

And the final dashboard looks like this,



**7.5. Program**

// including all the necessary files

#include <ESP8266WiFi.h>

#include "Adafruit\_MQTT.h"

#include "Adafruit\_MQTT\_Client.h"

#include "DHT.h"

#include <WiFiManager.h>

#include <DNSServer.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\* Pin Definition \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Relays for switching appliances

#define Relay1 D1 //GPIO 5

#define Relay2 D2 //GPIO 4

#define Relay3 D3 //GPIO 0

#define Relay4 D4 //GPIO 2

#define Relay5 D8 //GPIO 15

#define Relay6 D9 //GPIO19 / Rx pin

//DHT11 for reading temperature and humidity value

#define DHTPIN D5 //GPIO 14

//buzzer to know the status of MQTT connections and can be used for any other purpose according to your project need.

#define buzzer D0 //GPIO 16

//Selection pins for multiplexer module to switch between different sensors and give data on a single analog pin.

#define S0 D7 //GPIO 12

#define S1 D6 //GPIO 13

//Analog pin to read the incoming analog value from different sensors.

#define analogpin A0 //Analog pin

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Adafruit.io Setup \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define AIO\_SERVER "io.adafruit.com"

#define AIO\_SERVERPORT 1883 // use 8883 for SSL

#define AIO\_USERNAME "My\_Project" // Username

#define AIO\_KEY "###############################" // Auth Key

//**Note: username and AIO\_KEY are available on AdafruitIo dashboard on right-hand side there is an option with Key Symbol and**

/\*\*\*\*\*\*\*\*\*\*\*\* Global State (you don't need to change this!) \*\*\*\*\*\*\*\*\*\*\*/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.

WiFiClient client;

// or... use WiFiFlientSecure for SSL

//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.

Adafruit\_MQTT\_Client mqtt(&client, AIO\_SERVER, AIO\_SERVERPORT, AIO\_USERNAME, AIO\_KEY);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Feeds \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Notice MQTT paths for AIO follow the form: <username>/feeds/<feedname>

Adafruit\_MQTT\_Publish Humidity = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/humidity");

Adafruit\_MQTT\_Publish Temperature = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/temperature");

Adafruit\_MQTT\_Publish CO2 = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/co2");

Adafruit\_MQTT\_Publish Motion = Adafruit\_MQTT\_Publish(&mqtt, AIO\_USERNAME "/feeds/motion");

// Setup a feed called 'onoff' for subscribing to changes.

Adafruit\_MQTT\_Subscribe lights = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/relay1");

Adafruit\_MQTT\_Subscribe kitchenlights = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/relay2");

Adafruit\_MQTT\_Subscribe fan = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/relay3");

Adafruit\_MQTT\_Subscribe bulb = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/relay4");

Adafruit\_MQTT\_Subscribe lamp = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/relay5");

Adafruit\_MQTT\_Subscribe projector = Adafruit\_MQTT\_Subscribe(&mqtt, AIO\_USERNAME "/feeds/relay6");

/\*\*\*\*\*\*\*\*\*\*\*\* Necessary declaration for DHT11 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);

uint32\_t delayMS;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Sketch Code \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void MQTT\_connect();

void setup() {

Serial.begin(115200);

WiFiManager wifiManager;

wifiManager.autoConnect("Home\_Automation");

Serial.println("connection...yeey :)");

delay(10);

pinMode(buzzer, OUTPUT);

pinMode(Relay1, OUTPUT);

pinMode(Relay2, OUTPUT);

pinMode(Relay3, OUTPUT);

pinMode(Relay4, OUTPUT);

pinMode(Relay5, OUTPUT);

pinMode(Relay6, OUTPUT);

pinMode(S0, OUTPUT);

pinMode(S1, OUTPUT);

pinMode(A0, INPUT);

Serial.println(F("Adafruit MQTT demo"));

//Setting up DHT sensor

dht.begin();

// Setup MQTT subscription for onoff feed.

mqtt.subscribe(&lights);

mqtt.subscribe(&kitchenlights);

mqtt.subscribe(&fan);

mqtt.subscribe(&bulb);

mqtt.subscribe(&lamp);

mqtt.subscribe(&projector);

}

uint32\_t x = 0;

void loop() {

// Ensure the connection to the MQTT server is alive (this will make the first

// connection and automatically reconnect when disconnected). See the MQTT\_connect

// function definition further below.

MQTT\_connect();

// this is our 'wait for incoming subscription packets' busy subloop

// try to spend your time here

Adafruit\_MQTT\_Subscribe \*subscription;

while ((subscription = mqtt.readSubscription(20000))) {

if (subscription == &lights) {

Serial.print(F("Got: "));

Serial.println((char \*)lights.lastread);

int lights\_State = atoi((char \*)lights.lastread);

digitalWrite(Relay1, lights\_State);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

if (subscription == &kitchenlights) {

Serial.print(F("Got: "));

Serial.println((char \*)kitchenlights.lastread);

int kitchenlights\_State = atoi((char \*)kitchenlights.lastread);

digitalWrite(Relay2, kitchenlights\_State);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

if (subscription == &fan) {

Serial.print(F("Got: "));

Serial.println((char \*)fan.lastread);

int fan\_State = atoi((char \*)fan.lastread);

digitalWrite(Relay3, fan\_State);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

if (subscription == &bulb) {

Serial.print(F("Got: "));

Serial.println((char \*)bulb.lastread);

int bulb\_State = atoi((char \*)bulb.lastread);

digitalWrite(Relay4, bulb\_State);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

if (subscription == &lamp) {

Serial.print(F("Got: "));

Serial.println((char \*)lights.lastread);

int lamp\_State = atoi((char \*)lamp.lastread);

digitalWrite(Relay1, lamp\_State);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

if (subscription == &projector) {

Serial.print(F("Got: "));

Serial.println((char \*)projector.lastread);

int projector\_State = atoi((char \*)projector.lastread);

digitalWrite(Relay1, projector\_State);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

}

// Now we can publish stuff!

digitalWrite(S0, LOW);

digitalWrite(S1, LOW);

Serial.print("Motion ");

Serial.println(analogRead(analogpin));

Serial.print("...");

int Value = analogRead(analogpin);

if(Value>400)

Value=1;

else

Value=0;

if (! Motion.publish(Value)) {

Serial.println(F("Failed"));

} else {

Serial.println(F("OK!"));

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

}

digitalWrite(S0, HIGH);

digitalWrite(S1, LOW);

Serial.print("C02 ");

Serial.println(analogRead(analogpin));

Serial.print("...");

Value = analogRead(analogpin);

if (! CO2.publish(Value)) {

Serial.println(F("Failed"));

} else {

Serial.println(F("OK!"));

}

// Reading temperature or humidity takes about 250 milliseconds!

// Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)

float h = dht.readHumidity();

// Read temperature as Celsius (the default)

float t = dht.readTemperature();

// Read temperature as Fahrenheit (isFahrenheit = true)

float f = dht.readTemperature(true);

// Check if any reads failed and exit early (to try again).

if (isnan(h) || isnan(t) || isnan(f)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

if (! Humidity.publish(h)) {

Serial.println(F("Failed"));

} else {

Serial.println(F("OK!"));

}

if (! Temperature.publish(t)) {

Serial.println(F("Failed"));

} else {

Serial.println(F("OK!"));

}

if(! mqtt.ping()) { // ping the server to keep the mqtt connection alive

mqtt.disconnect(); // NOT required if you are publishing once every KEEPALIVE seconds

}

}

// Function to connect and reconnect as necessary to the MQTT server.

// Should be called in the loop function and it will take care if connecting.

void MQTT\_connect() {

int8\_t ret;

// Stop if already connected.

if (mqtt.connected()) {

return;

}

Serial.print("Connecting to MQTT... ");

uint8\_t retries = 3;

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

digitalWrite(buzzer, HIGH);

delay(200);

digitalWrite(buzzer, LOW);

delay(200);

while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected

Serial.println(mqtt.connectErrorString(ret));

Serial.println("Retrying MQTT connection in 5 seconds...");

mqtt.disconnect();

delay(5000); // wait 5 seconds

retries--;

if (retries == 0) {

// basically die and wait for WDT to reset me

while (1);

}

}

Serial.println("MQTT Connected!");

digitalWrite(buzzer, HIGH);

delay(2000);

digitalWrite(buzzer, LOW);

}

**Testing & Validation**

**8. Testing and Validation**

**Testing:**

* This Google assistance is tested on both WIFI & Hotspot and it is performing well.
* The Total internet it consumes in a day is 10 megabytes only.
* It has the capacity to turn off all the devices at one time.
* This test has a 100% positive working and request & response from the server.
* We can control the devices by using IOT MQTT Panel application and also from the google assistant command both at a time.

**Validation:**

* No can change the WIFI SSID and password easily.
* Password of WIFI and the AUTH key of AdafruitIo is always confidential.

**Conclusion**

**9. Conclusion**

The aim of this paper was to propose a cost-effective voice controlled (Google Assistant) home automation controlling general appliances found in one’s home. The approach discussed in the paper was successful as GAHA’s (Google Assistant Home Automation) design was successfully implemented

. The system as the name indicates, ‘Home automation’ makes the system more flexible and provides attractive user interface compared to other home automation systems. In this system we integrate mobile devices into home automation systems. A novel architecture for a home automation system is proposed using the relatively new communication technologies. The system consists of mainly three components is a Sensors, ESP8266 and relay circuits. WIFI is used as the communication channel between android phone and the ESP8266. We hide the complexity of the notions involved in the home automation system by including them into a simple, but comprehensive set of related concepts. This simplification is needed to fit as much of the functionality on the limited space offered by a mobile device’s display. This paper proposes a low cost, secure, ubiquitously accessible, auto-configurable, remotely controlled solution.

The approach discussed in the paper is novel and has achieved the target to control home appliances remotely using the WiFi technology to connects system parts, satisfying user needs and requirements. WiFi technology capable solution has proved to be controlled remotely, provide home security and is cost-effective as compared to the previously existing systems. It also sense different gases, temperature and even also motion From gas sensor we can check the level of Gas available in a room, Temperature sensor sense the temperature of the room , motion sensor detects the motion of any living object like any human being or animal and all the data of these sensors are monitored by using an application. Hence, we can conclude that the required goals and objectives of home automation system have been achieved. The system design and architecture were discussed, and prototype presents the basic level of home appliance control and remote monitoring has been implemented. Finally, the proposed system is better from the scalability and flexibility point of view than the commercially available home automation systems.

**Problem faced**

**10. Problem faced**

If the connection is not properly done then this may cause an life danger or also you can burn up our home fuse or also burn the Nodemcu so if you are doing this then please take a proper knowledge of electrical wiring , if you don’t know any thing then please take the help of the electrician.

I have burned my Nodemcu because of improper wiring

In this system Wi-Fi module is a very important part for communication between mobile phone and microcontroller. Wi-Fi module requires only 3.3v power whereas the Relay board needs more power. Thus, separate power will be given to both the relay and microcontroller. Connection between the microcontroller and Wi-Fi module and relay should not be loose if it is losing the system will not work properly.

**Future Scope**

**11. Future scope**

The future scope for GAHA can be huge. There are many factors to improve on to make GAHA more powerful, intelligent, scalable, and to become better overall for home automation. For example, controlling the speed of the fan, a greater number of devices can be integrated, like a coffee machine, air conditioner etc. Well, no system is ever perfect. It always has a scope for improvement. One just needs to put on a thinking cap and try and make the system better.

Android app will also develop for easily. For more security purpose camera module can also be implemented on the system. If any person attempts to enter in home with more than three-time wrong password then at that time camera module will be activate. And camera module will capture the image of person who trying to attack on system. It can use antivirus so that hacking of the system can be difficult.

**12. Bibliography**

* **learn.adafruit.com**

<https://learn.adafruit.com/adafruit-io-basics-dashboards/creating-a-dashboard?view=all>

* IFTTT

<https://ifttt.com>

* AdafruitIo

https://accounts.adafruit.com