## TABLE OF CONTENTS

**Acknowledgement…………….…………………………………………………………………..i**

**Estimated Cost..………………………………………………………………………………....,ii**

**Abstract……..……………………………………………………………….…………………...iii**

**List of Figures. .……………………………………..………………………………………….iv**

**Listof Tables …………………………………………………………………………………….v**

**Chapter 1: Introduction** **to Internet Of Things 3-6**

* 1. What is Internet Of Things ? 3
  2. How does it work ? 4
  3. The Future of IoT 5

**Chapter 2: Introduction to System On Chips 7-9**

* 1. What is a System On Chip ?7

2.2 Features Of System On Chip 8

**Chapter 3: Technologies Used 10-19**

* 1. Hardware 10

3.1.1 NODE MCU ESP8266 10

3.1.2 Four Channel Relay Board 11

3.2 Software 13

3.2.1 Arduino IDE 13

3.2.2 Adafruit IO Cloud Service 14

3.2.3 IFTT Service 16

3.2.4Google Assistant 18

**Chapter 4: Project Design 20-24**

* 1. Connection Diagram 20

4.1.1 Schmatic Diagram 20

4.1.2 General Diagram 21

4.2 Applications 24

**Conclusion 25**

**Future Scope 26**

**References**  **27**

**Appendix - A 28**

**Appendix - B 32**

**CHAPTER 1**

**INTRODUCTION TO INTERNET OF THINGS**

* 1. **WHAT IS INTERNET OF THINGS ?**

The Internet of Things (IoT) is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect and exchange data, creating opportunities for more direct integration of the physical world into computer-based systems, resulting in efficiency improvements, economic benefits, and reduced human exertions. The number of IoT devices increased 31% year-over-year to 8.4 billion in the year 2017 and it is estimated that there will be 30 billion devices by 2020. The global market value of IoT is projected to reach $7.1 trillion by 2020. IoT involves extending Internet connectivity beyond standard devices, such as desktops, laptops, smartphones and tablets, to any range of traditionally dumb or non-internet-enabled physical devices and everyday objects. Embedded with technology, these devices can communicate and interact over the Internet, and they can be remotely monitored and controlled. With the arrival of driverless vehicles, a branch of IoT, i.e. the Internet of Vehicle starts to gain more attention. The term "Internet of things" was likely coined by Kevin Ashton of Procter & Gamble, later MIT's Auto-ID Center, in 1999, though he prefers the phrase "Internet for things". At that point, he viewed Radio-frequency identification (RFID) as essential to the Internet of things, which would allow computers to manage all individual things. A research article mentioning the Internet of Things was submitted to the conference for Nordic Researchers in Logistics, Norway, in June 2002, which was preceded by an article published in Finnish in January 2002. Although many people think that IoT connectivity is or will become a commodity with little value for customers and along with the hardware will form the ugly ducklings of the value chain, IoT network operators should strive to demonstrate that IoT connectivity is vital for the global adoption of IoT and seek to increase the income derived from its connectivity se The new stream of data coming from the physical world and the billions connected things are mostly transported by IoT network operators’ networks, and once this data is captured, IoT network operators can monitor everything and feed their AI systems. It is then when IoT network operators can make a lot of money out of IoT contextual and aggregated data.rvices with aspects like security and the contextual data value that their networks transport.

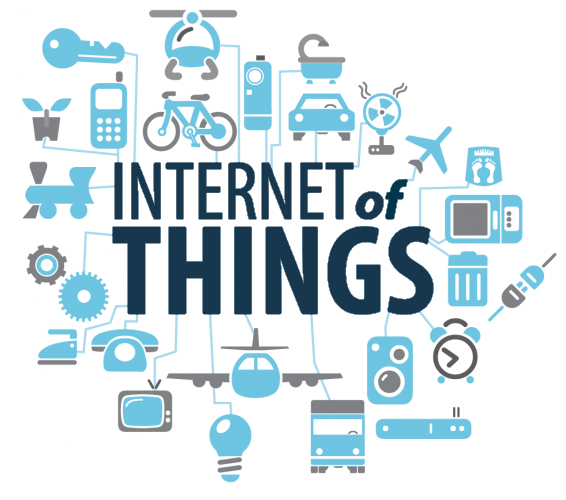


Figure 1.1 : Internet Of Things

Defining the Internet of Things as "simply the point in time when more 'things or objects' were connected to the Internet than people", Cisco Systems estimated that IoT was "born" between 2008 and 2009, with the things/people ratio growing from 0.08 in 2003 to 1.84 in 2010.

* 1. **HOW DOES IT WORK ?**

Devices and objects with built in sensors are connected to an Internet of Things platform, which integrates data from the different devices and applies analytics to share the most valuable information with applications built to address specific needs These powerful IoT platforms can pinpoint exactly what information is useful and what can safely be ignored. This information can be used to detect patterns, make recommendations, and detect possible problems before they occur.

For example, if I own a car manufacturing business, I might want to know which optional components (leather seats or alloy wheels, for example) are the most popular. Using Internet of Things technology, I can:

* Use sensors to detect which areas in a showroom are the most popular, and where customers linger longest.
* Drill down into the available sales data to identify which components are selling fastest.
* Automatically align sales data with supply, so that popular items don’t go out of stock.
* The information picked up by connected devices enables me to make smart decisions about which components to stock up on, based on real-time information, which helps me save time and money.
* With the insight provided by advanced analytics comes the power to make processes more efficient. Smart objects and systems mean you can automate certain tasks, particularly when these are repetitive, mundane, time-consuming or even dangerous.

**1.3**  **THE FUTURE OF IOT**

Internet of Things include connected security systems, thermostats, cars, electronic appliances, lights in household and commercial environments, alarm clocks, speaker systems, vending machines and more. Businesses can leverage IoT applications to automate safety tasks (for example, notify authorities when a fire extinguisher in the building is blocked) to performing real-world A/B testing using networked cameras and sensors to detect how customers engage with products. The Internet of Things extends internet connectivity beyond traditional devices like desktop and laptop computers, smartphones and tablets to a diverse range of devices and everyday things that utilize embedded technology to communicate and interact with the external environment, all via the Internet. IoT connectivity is still at the core of all IoT network operators and M2M service providers. But some of them are implementing different strategies to capture more business from the IoT value chain. The idea of IoT connectivity will become a commodity, with added value influencing the decision to invest in new IoT-enabled networks (5G, LTE-M, NB-IoT). It’s clear that there are some strong opportunities for IoT network operators and M2M service providers looking to capture the full potential of IoT, and it’s time that they open up their services to support companies from all sectors that are looking to employ not only IoT connectivity, but also machine data intelligence as part of their business models in this IoT-driven digital transformation. Telcos offering IoT connectivity should look to monetize data and offer businesses unique insights that could potentially open doors to new revenue streams or even improve operational efficiencies.As far as the reach of the Internet of Things, there are more than 12 billion devices that can currently connect to the Internet, and researchers at IDC estimate that by 2020 there will be 26 times more connected things than people.

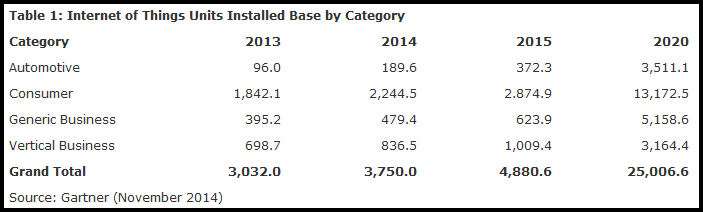


Figure 1.2 : Future Scope Of IoT

According to Gartner, consumer applications will drive the number of connected things, while enterprise will account for most of the revenue. IoT adoption is growing, with manufacturing and utilities estimated to have the largest installed base of Things by 2020.

**CHAPTER 2**

**INTRODUCTION TO SYSTEM ON CHIPS**

* 1. **WHAT IS A SYSTEM ON CHIP ?**

A system on a chip or system on chip (SoC) is an integrated circuit that integrates all components of a computer or other electronic system. These components typically include a central processing unit (CPU), memory, input/output ports and secondary storage – all on a single substrate. It may contain digital, analog, mixed-signal, and often radio frequency signal processing functions, depending on the application. As they are integrated on a single electronic substrate, SoCs consume much less power and take up much less area than multi-chip designs with equivalent functionality. Because of this, SoCs are very common in the mobile computing and edge computing markets. Systems on chip are commonly used in embedded systems and the Internet of Things.

Systems on Chip are in contrast to the common traditional motherboard-based PC architecture, which separates components based on function and connects them through a central interfacing circuit board. Whereas a motherboard houses and connects detachable or replaceable components, SoCs integrate all of these components into a single integrated circuit, as if all these functions were built into the motherboard. An SoC will typically integrate a CPU, graphics and memory interfaces, hard-disk and USB connectivity, random-access and read-only memories and secondary storage on a single circuit die, whereas a motherboard would connect these modules as discrete components or expansion cards. More tightly integrated computer system designs improve performance and reduce power consumption as well as semiconductor die area needed for an equivalent design composed of discrete modules, at the cost of reduced replaceability of components. By definition, SoC designs are fully or nearly fully integrated across different component modules. For these reasons, there has been a general trend towards tighter integration of components in the computer hardware industry, in part due to the influence of SoCs and lessons learned from the mobile and embedded computing markets. Systems-on-Chip can be viewed as part of a larger trend towards embedded computing and hardware acceleration.

An SoC integrates a microcontroller or microprocessor with advanced peripherals like graphics processing unit (GPU), Wi-Fi module, or one or more coprocessors. Similar to how a microcontroller integrates a microprocessor with peripheral circuits and memory, an SoC can be seen as integrating a microcontroller with even more advanced peripherals. For an overview of integrating system components, see system integration.



Figure 2.1 : System On Chip

* 1. **FEATURES OF SYSTEM ON CHIPS :**

System-on-a-chip technology is used in small, increasingly complex consumer electronic devices. Some such devices have more processing power and memory than a typical 10-year-old desktop computer. A SoC contains the following in their internal architecture :

CPU – the central processing unit, whether it’s single- or multiple-core, this is what makes everything possible on your smartphone. Most processors found inside the SoCs that we’re going to look at will be based on ARM technology, but more on that later

Memory – just like in a computer, memory is required to perform the various tasks smartphone and tablets are capable of, and therefore SoCs come with various memory architectures on board

GPU – the graphic processing unit is also an important component on the SoC, and it’s responsible for handling those complex 3D games on the smartphone or tablets. As you can expect, there are various GPU architectures available out there, and we’re going to further detail them in what follows.

Northbridge – this is a component that handles communications between the CPU and other components of the SoC including the southbridge

Southbrige – a second chipset usually found on computers that handles various I/O functions. In some cases the southbridge can be found on the SoC

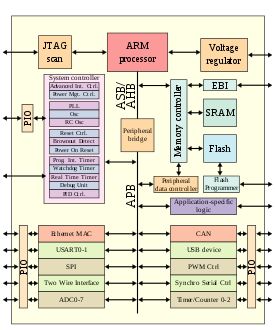


Figure 2.2 : Features Of SoCs

Cellular radios – some SoCs also come with certain modems on board that are needed by mobile operators. Such is the case with the Snapdragon S4 from Qualcomm, which has an embedded LTE modem on board responsible for 4G LTE connectivity

Other radios – some SoCs may also have other components responsible for other types of connectivity, including Wi-Fi, GPS/GLONASS or Bluetooth. Again, the S4 is a good example in this regard.

Other circuitry

**CHAPTER 3**

**TECHNOLOGIES USED**

**3.1 HARDWARE**

There are several different kinds of programmable microcontrollers. In this project many of the most common types categorized by several parameters including Bits, Flash size, RAM size, number of input/output lines, packaging type, supply voltage and speed. Our parametric filters will allow you to refine your search results according to the required specifications Programmable microcontrollers contain general purpose input/output pins.

**3.1.1 NODE MCU ESP 8266**

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson, and spiffs. NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub.

It is based ESP8266 -12E WiFi module. It is an highly integrated chip designed to provide full internet connectivity in a small package. It can be programmed directly through USB port using LUA programming or Arduino IDE. By simple programming we can establish a WiFi connection and define input/output pins according to your needs exactly like arduino, turning into a web server and a lot more. NodeMCU is the WiFi equivalent of ethernet module. It combines the features of WiFi access point and station + microcontroller. These features make the NodeMCU extremely powerful tool for WiFi networking. It can be used as access point and/or station, host a web server or connect to internet to fetch or upload data. As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate tool chains to allow Arduino C/C++ to be compiled down to these new processors. They did this with the introduction of the Board Manager and the SAM Core.

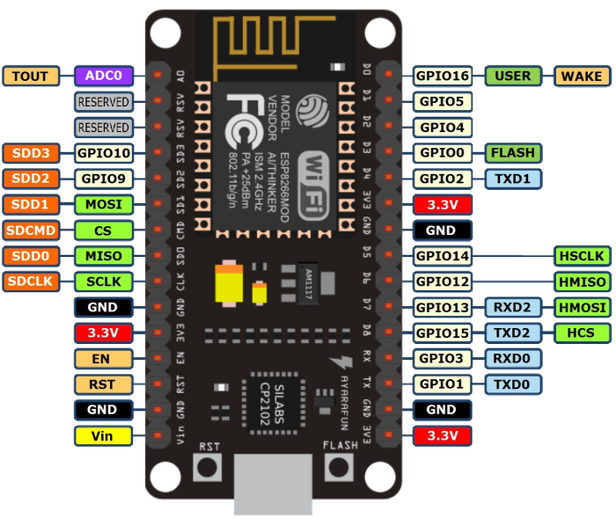


Figure 3.1 : NODE MCU ESP 8266 Pin Config

A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file down to the target MCU's machine language. Some creative ESP8266 enthusiasts have developed an Arduino core for the ESP8266 WiFi SoC that is available at the GitHub ESP8266 Core webpage. This is what is popularly called the "ESP8266 Core for the Arduino IDE" and it has become one of the leading software development platforms for the various ESP8266 based modules and development boards, including NodeMCUs[4].

**3.1.2 FOUR CHANNEL RELAY BOARD**

This is a 5V 4-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 equiped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. When the signal port is at low level, the signal light will light up and the optocoupler 817c (it transforms electrical signals by light and can isolate input and output electrical signals) will conduct, and then the transistor will conduct, the relay coil will be electrified, and the normally open contact of the relay will be closed. When the signal port is at high level, the normally closed contact of the relay will be closed. So you can connect and disconnect the load by controlling the level of the control signal port. A Four Channel Relay Board has following advantages over other relay modules and they are as follows :

* Module can be controlled directly by Microcontroller (Raspberry Pi, Arduino, 8051, AVR, PIC, DSP, ARM, ARM, MSP430, TTL logic)
* Easy to install and fix
* Optically Isolated relays to protect your microcontroller from damage if the equipment being controlled fails four screw holes, hole diameter 3.1mm
* Relay status indicator light, release status LED is off
* Relay Maximum output: DC 30V/10A, AC 250V/10A
* Size: 75mm (l) x 55mm (b) x 19.3mm (h)
* Weight: 58g

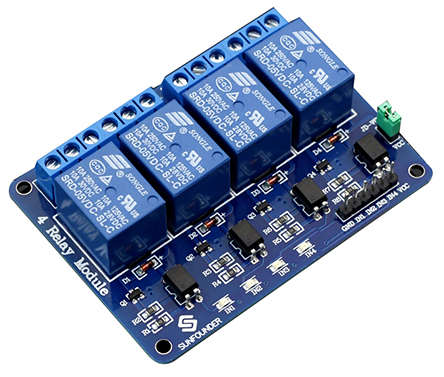


Figure 3.2 : 4 Channel Relay

**3.2 SOFTWARE**

* **ARDUINO IDE :** The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino board. The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides

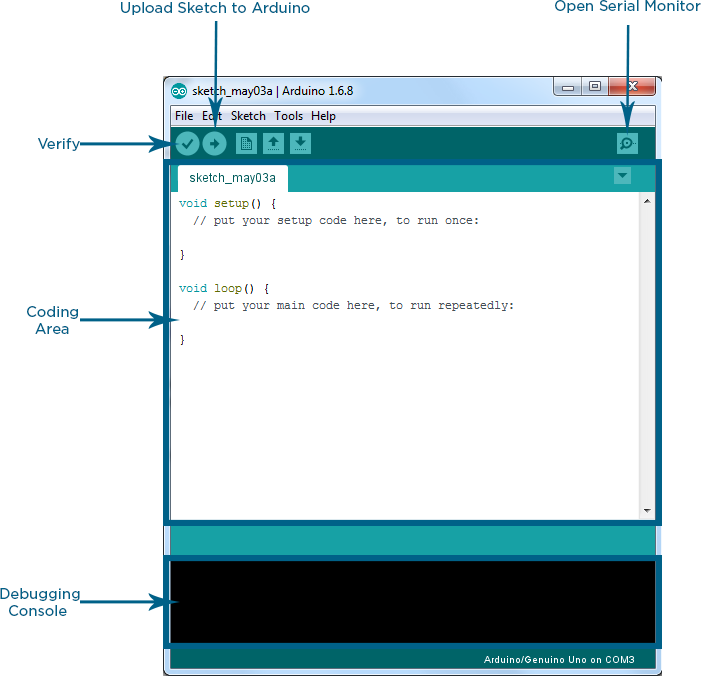


Figure 3.3 : Arduino IDE

many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware[4]. The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino board. The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. Usually, the Arduino IDE is the first IDE we use when approaching the first time to Arduino. This happens for several reasons: it is easy to use, it supports all the Arduino boards, and it has a built-in library manager that is also easy to use. Moreover, the Arduino IDE is very user-friendly without too many options, menus, and so on that could scare an inexperienced user. It is so easy that we do not have to worry about how it works, we can focus only on the development process. We write the Arduino code, and the Arduino IDE compiles it and uploads the compiled code into the Arduino board. That’s it!. Moreover, it is open source and runs on several OS, such as Windows, Mac OS X, and Linux. Anyway, if you do not feel comfortable with Arduino IDE, there are several alternatives you can use when developing your Arduino project. There are some missing features in Arduino IDE that could bother you, especially if you are used to IDE, like Eclipse, Visual Studio, or IntelliJ, just to name a few.

* **ADAFRUIT IO CLOUD SERVICE**

Adafruit.io is a cloud service - that just means we run it for you and you don't have to manage it. You can connect to it over the Internet. It's meant primarily for storing and then retrieving data but it can do a lot more than just that! The AIO Key is used to restrict or grant access to your data. The key is unique, and you can generate a key per feed, and control it in many different ways. The easiest process is to just use your automatically generated master key. You can access this key right from the right-hand side of your dashboard or from an individual feed page. The AIO Key is used to restrict or grant access to your data. The key is unique, and you can generate a key per feed, and control it in many different ways. The easiest process is to just use your automatically generated master key. You can access this key right from the right-hand side of your dashboard or from an individual feed page.

Adafruit has following features :

* Display your data in real-time, online
* Make your project internet-connected: Control motors, read sensor data, and more!
* Connect projects to web services like Twitter, RSS feeds, weather services, etc.
* Connect your project to other internet-enabled devices
* Adafruit IO also allows integration with IFTTT and Zapier.



Figure 3.4 : Adafruit IO

**DASHBOARDS**

Adafruit.io can handle and visualize multiple feeds of data. It can display data from a temperature-humidity sensor alongside data from an air quality sensor and add a button to turn on the air-conditioner in your room. Dashboards are a feature integrated into Adafruit IO which allow you to chart, graph, gauge, log, and display your data. You can view your dashboards from anywhere in the world.

**TRIGGERS**

Use triggers in Adafruit IO to control and react to your data. Configure triggers to email you when your system goes offline, react to a temperature sensor getting too hot, and publish a message to a new feed. Moreover, the Platform IO supports multi-projects, themes, and library management. The interface is very attractive and easy to use. However, it is very different from the standard

* **IFTT**

IFTTT is called as ‘If This Then That’. IFTTT is a free platform that helps you do more with all your apps and devices. We can create our own applets and connect them with the available services. We will create a trigger and then connect it to an action to perform our specific task. In our case, the trigger service will be ‘Google Assistant’ and the Action will be ‘WebHooks’.

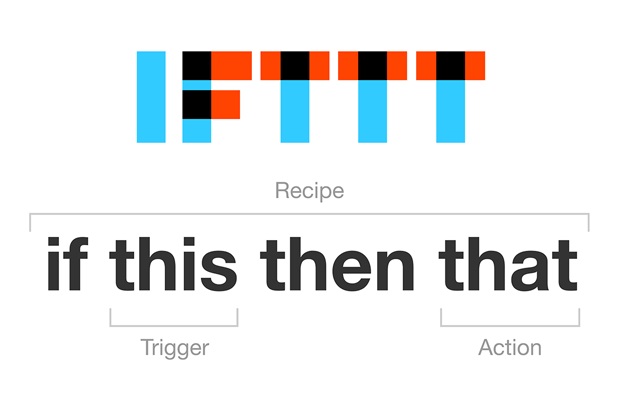


Figure 3.5 : IFTT

If This Then That, also known as IFTTT [6] is a free web-based service to create chains of simple conditional statements, called applets. An applet is triggered by changes that occur within other web services such as Gmail, Facebook, Telegram, Instagram, or Pinterest. For example, an applet may send an e-mail message if the user tweets using a hashtag, or copy a photo on Facebook to a user's archive if someone tags a user in a photo. In addition to the web-based application, the service runs on iOS and Android. In February 2015, IFTTT renamed its original application to IF, and released a new suite of apps called Do, which lets users create shortcut applications and actions. As of 2015, IFTTT users created about 20 million recipes each day. All of the functionalities of the Do suite of apps have since been integrated into a redesigned IFTTT app. IFTTT employs the following concepts:

* Services (formerly known as channels) are the basic building blocks of IFTTT. They mainly describe a series of data from a certain web service such as YouTube or eBay.
* Services can also describe actions controlled with certain APIs, like SMS. Sometimes, they can represent information in terms of weather or stocks.[20] Each service has a particular set of triggers and actions.
* Triggers are the "this" part of an applet. They are the items that trigger the action. For example, from an RSS feed, you can receive a notification based on a keyword or phrase.
* Actions are the "that" part of an applet. They are the output that results from the input of the trigger.
* Applets (formerly known as recipes) are the predicates made from Triggers and Actions. For example, if you like a picture on Instagram (trigger), an IFTTT app can send the photo to your Dropbox account (action).
* Ingredients are basic data available from a trigger—from the email trigger, for example; subject, body, attachment, received date, and sender’s address.

IFTTT can automate web-application tasks, such as posting the same content on several social networks. Marketing professionals can use IFTTT to track mentions of companies in RSS feeds. IFTTT also is used in home automation, for instance switching on a light when detecting motion in a room (with associated compliant devices).

* **GOOGLE ASSISTANT**

The Google Assistant is an artificial intelligence-powered virtual assistant developed by Google that is primarily available on mobile and smart home devices. Unlike the company's previous virtual assistant, Google Now, the Google Assistant can engage in two-way conversations. Assistant initially debuted in May 2016 as part of Google's messaging app Allo, and its voice-activated speaker Google Home. After a period of exclusivity on the Pixel and Pixel XL smartphones, it began to be deployed on other Android devices in February 2017, including third-party smartphones and Android Wear (now Wear OS), and was released as a standalone app on the iOS operating system in May.

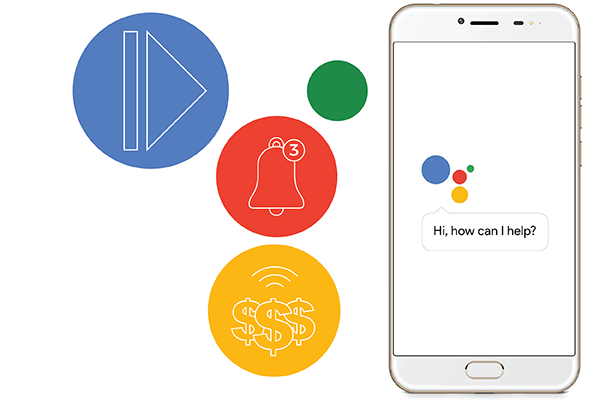


Figure 3.6 : Google Assistant

Alongside the announcement of a software development kit in April 2017, the Assistant has been, and is being, further extended to support a large variety of devices, including cars and smart home appliances. The functionality of the Assistant can also be enhanced by third-party developers. Users primarily interact with the Google Assistant through natural voice, though keyboard input is also supported. In the same nature and manner as Google Now, the Assistant is able to search the Internet, schedule events and alarms, adjust hardware settings on the user's device, and show information from the user's Google account. Google has also announced that the Assistant will be able to identify objects and gather visual information through the device's camera, and support purchasing products and sending money, as well as identifying songs.

The Google Assistant was unveiled during Google's developer conference on May 18, 2016, as part of the unveiling of the Google Home smart speaker and new messaging app Allo; Google CEO Sundar Pichai explained that the Assistant was designed to be a conversational and two-way experience, and "an ambient experience that extends across devices". Later that month, Google assigned Google Doodle leader Ryan Germick and hired former Pixar animator Emma Coats to develop "a little more of a personality." The Google Assistant, in the nature and manner of Google Now, can search the Internet, schedule events and alarms, adjust hardware settings on the user's device, and show information from the user's Google account. Unlike Google Now, however, the Assistant can engage in a two-way conversation, using Google's natural language processing algorithm. Search results are presented in a card format that users can tap to open the page. In February 2017, Google announced that users of Google Home would be able to shop entirely by voice for products through its Google Express shopping service, with products available from Whole Foods Market, Costco, Walgreens, PetSmart, and Bed Bath & Beyond at launch, and other retailers added in the following months as new partnerships were formed. The Google Assistant can maintain a shopping list; this was previously done within the notetaking service Google Keep, but the feature was moved to Google Express and the Google Home app in April 2017, resulting in a severe loss of functionality. In May 2017, Google announced that the Assistant would support a keyboard for typed input and visual responses, support identifying objects and gather visual information through the device's camera, and support purchasing products and sending money. Through the use of the keyboard, users can see a history of queries made to the Google Assistant, and edit or delete previous inputs. The Assistant warns against deleting, however, due to its use of previous inputs to generate better answers in the future. In November 2017, it became possible to identify songs currently playing by asking the Assistant. Google Assistant allows users to activate and modify vocal shortcut commands in order to perform actions on their device -both Android and Ipad/IPhone- or configuring it as an hub for the home automation. This feature of the speech recognition is available in English, among other languages.

**CHAPTER 4**

**PROJECT DESIGN**

**4.1 CONNECTION DIAGRAMS**

**4.1.2 SCHEMETIC DIAGRAM**

In this system we have directly connected Relay Board on to the NODE MCU ESP 8266’s pins and powered the with thePower Adapter.

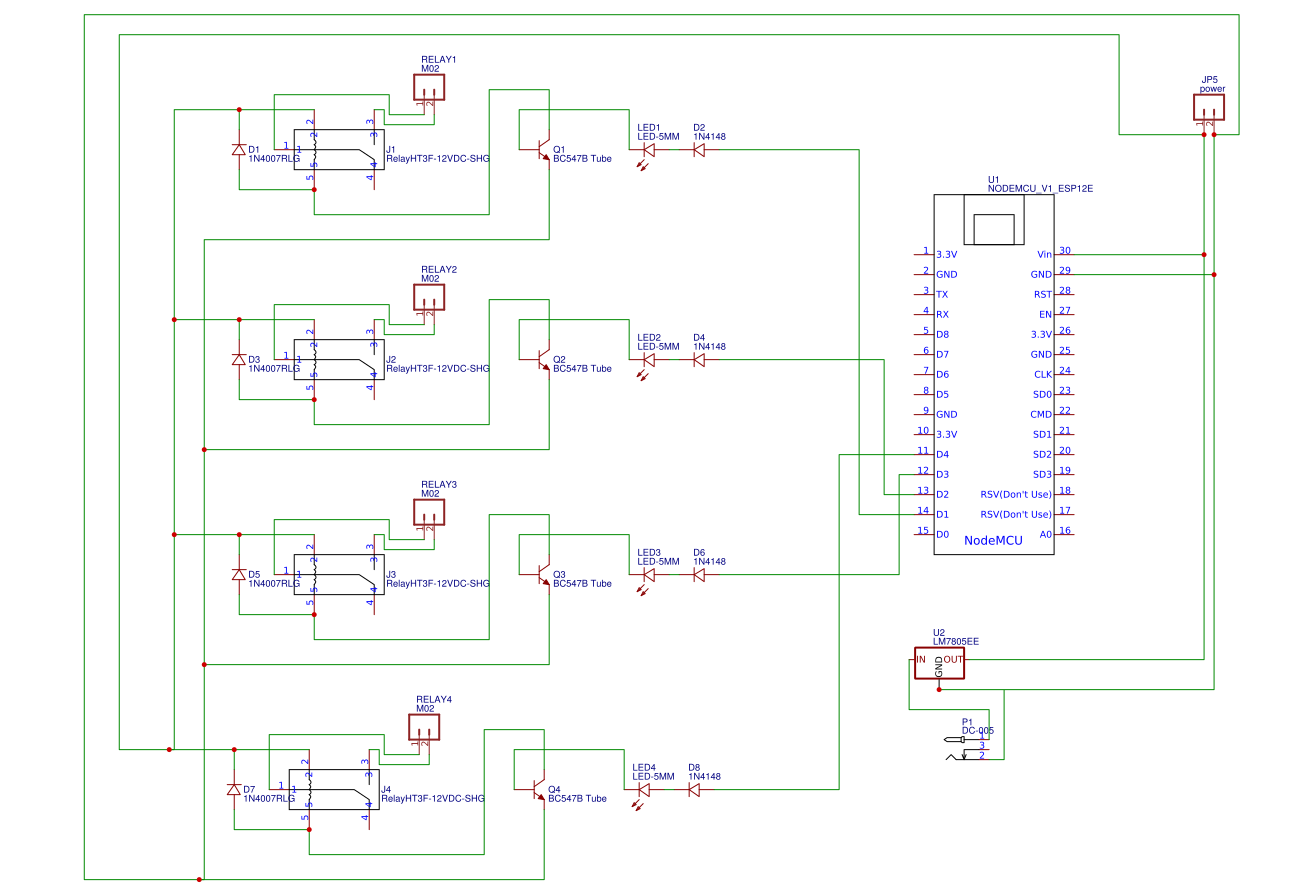


Figure 4.1 : Circuit Diagram

**4.1.2 GENERAL DIAGRAM**

This is a general diagram of the NODE MCU ESP 8266 which is connected with a single Relay Board with is furthur connected with the load :

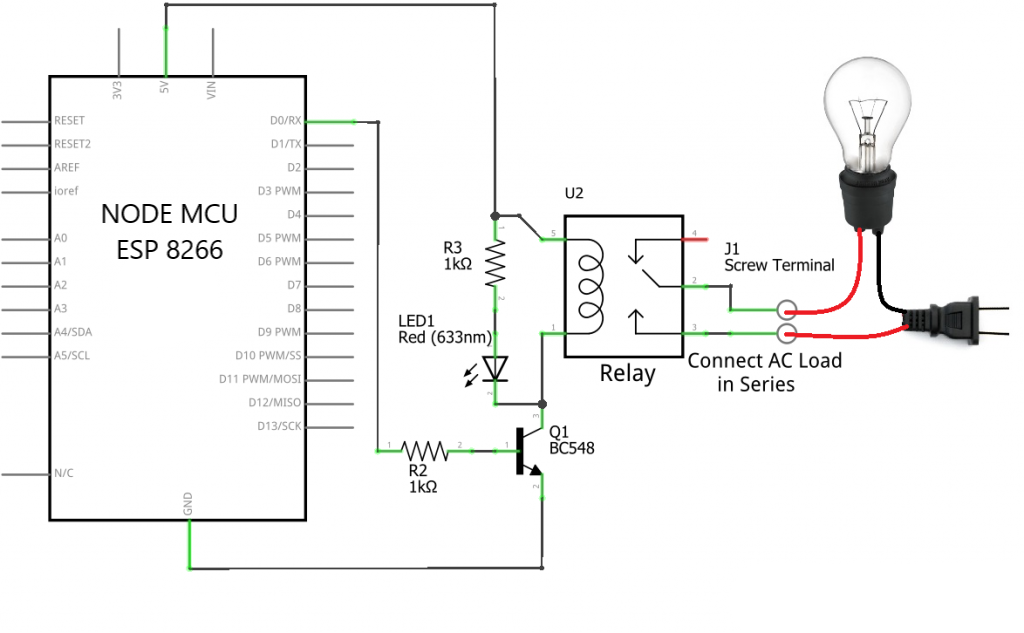
****

Figure 4.2 Circuit Diagram with Load

This is a general diagram of the NODE MCU ESP 8266 which is connected with the 4 Channel Relay Board with is furthur connected with the load :

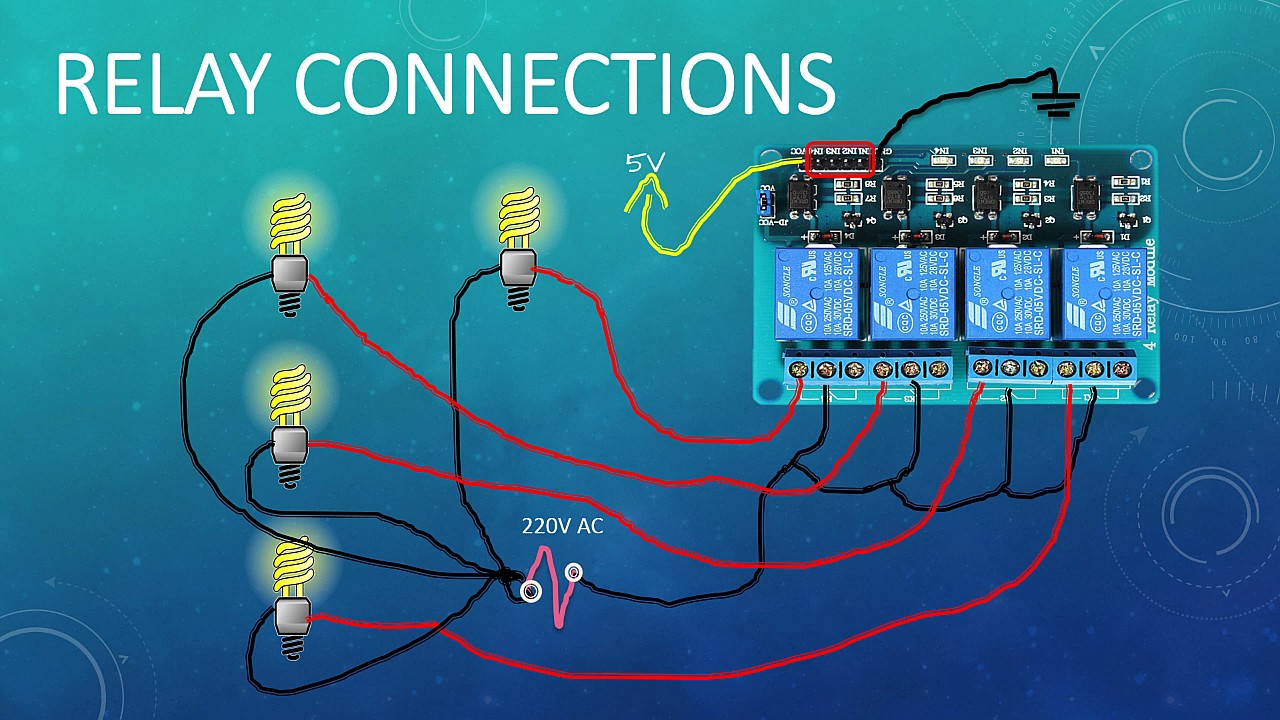


Figure 4.3: Connection Diagram with Loads

Here is the picture of the actual sytem with relay board connected on a PCB board design :

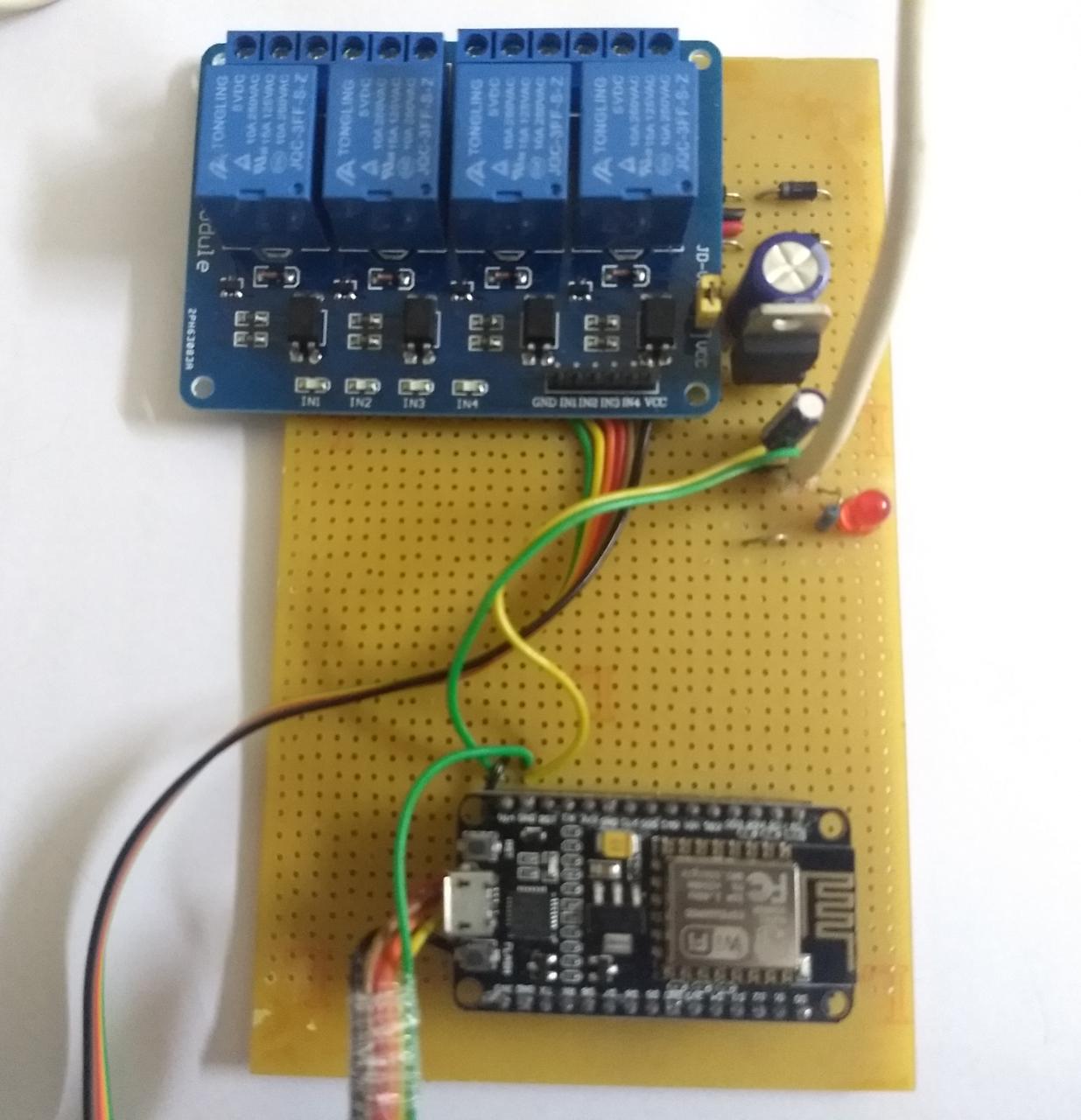


Figure 4.4 : Actual Image of the System

Here are some of the screenshots of the programs programed on Arduino IDE :

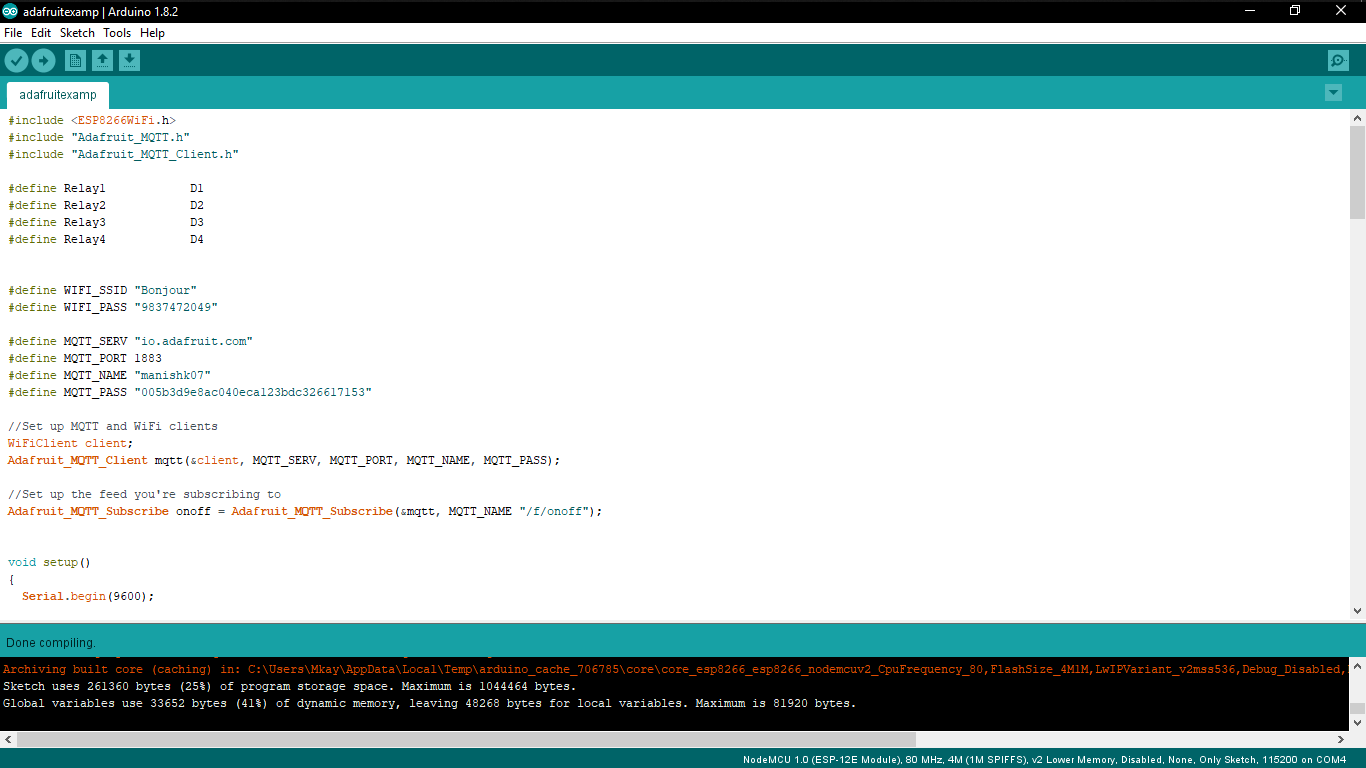


Figure 4.5 : Arduino IDE

And here is the screenshot of the Adafruit IO Cloud service used in this project :

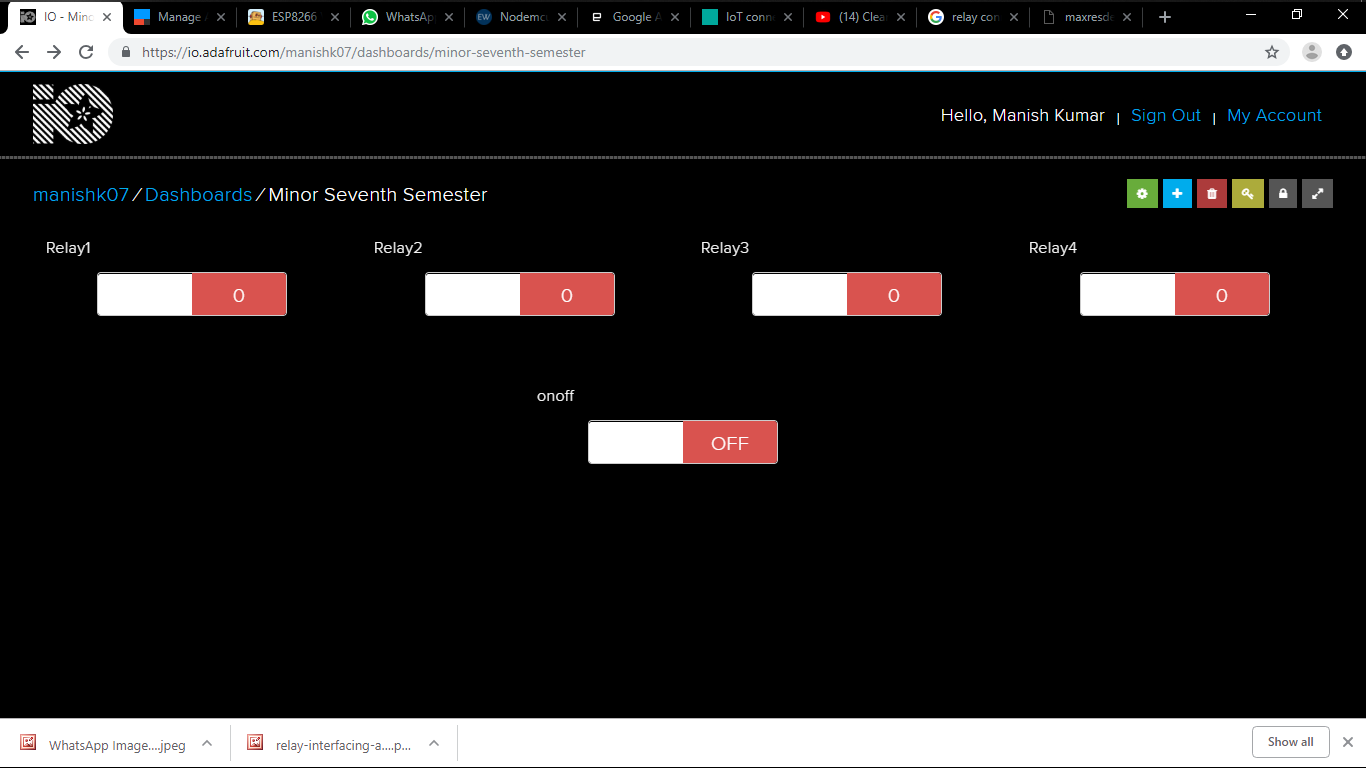


Figure 4.6 : Adafruit IO Status

And here is the screenshot of the IFTTT Service used in this project :

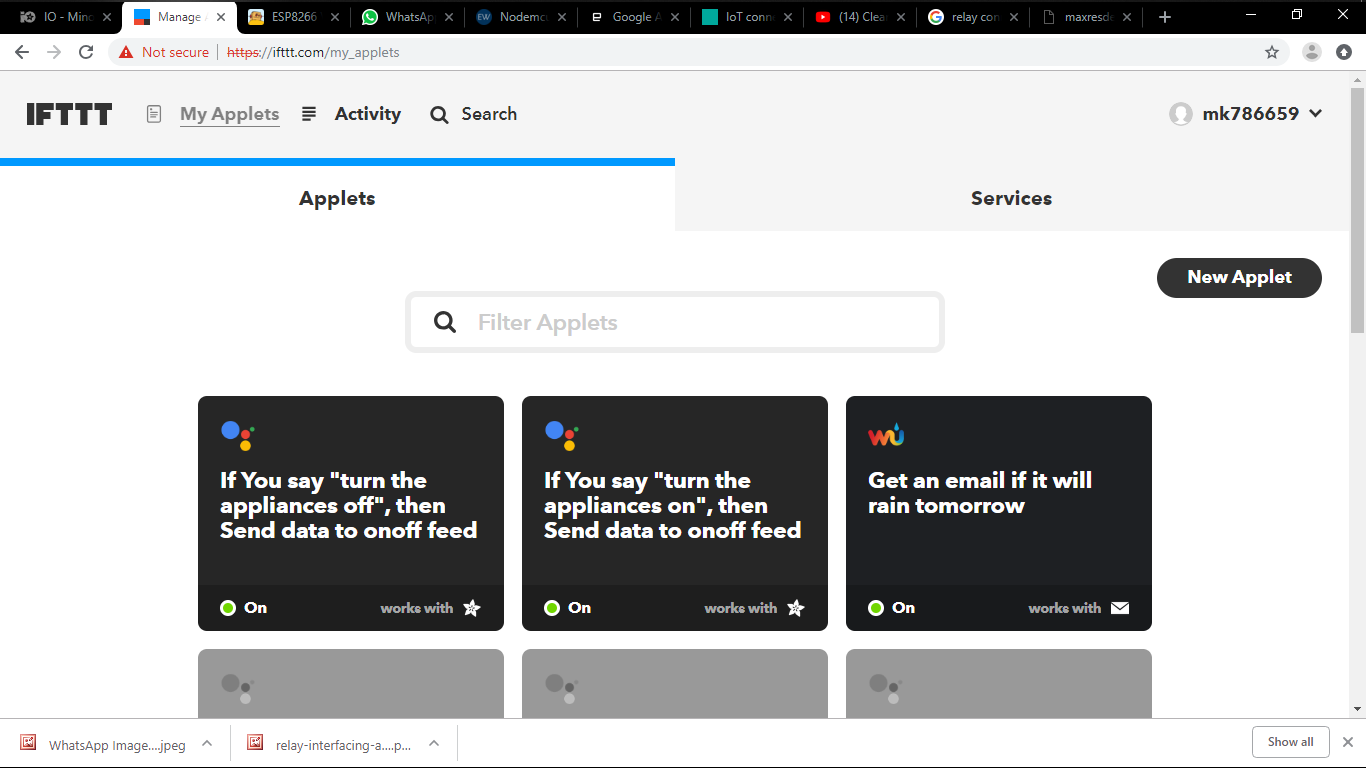


Figure 4.7 : IFTTT Status

Now, here’s a link to the github repo where everyone can acesses to the project :

<https://github.com/manishkumar1080/IOT-Based-Home-Automation-Using-Google-Assistant>

**4.2 APPLICATIONS**

* Smart Homes
* Smart Appliances
* Electronic Industries

**CONCLUSION**

The project has proposed the idea of smart homes that can support a lot of home automation systems. A smart home contains a connection between wireless communication, sensors, monitoring and tracking. Smart homes are a huge system that includes multiple technologies and applications that can be used to provide security and control of the home easily. In this project, an efficient approach for smart homes was proposed and implemented.

The future of IoT is virtually unlimited due to advances in technology and consumers desire to integrate devices such as smart phones with household machines. Wi-Fi has made it possible to connect people and machines on land, in the air and at sea. It is critical that both companies and governments keep in ethics in mind as we approach the fourth Industrial Revolution. With so much data traveling from device to device, security in technology will be required to grow just as fast as connectivity in order to keep up with demands. Governments will undoubtable face tough decisions as to how far the private the sector is allowed to go in terms of robotics and information sharing. The possibilities are exciting, productivity will increase and amazing things will come by connecting the world**.**

**FUTURE SCOPE**

It’s the technology of today which is touching and transforming the every aspect of our real life. IOT has given a concept of Machine to-Machine (M2M) communication. Companies like Microsoft and SAP are implementing strategy to capitalize on the Internet of Things so that you can just stop your business and start making it thrive. IOT is going to have huge impact on home automation and building automation system where every convenience will be taken care of by the interconnected devices on IOT. With the personal electronics good connected to Internet will enable us to “author” our lives. In medical science field, IOT has given a privilege to devices and system to sense for coming disease and to prevent it ,for eg: It can make a person healthier with wearables that can predict heart attack and cardiovascular strokes. As per a study of Consumer Electronics Association and Parks Associates found only 10% of the household in USA fully understood the usage of these products. Many interesting IoT products like automatic door locks, Wi-Fi connected ceiling fans, light swirches,LED bulbs, smart watches,3-D printers and smart clothes will be popular among consumers.MyBrain Technology in France has developed "Melomind". This EEG Headset can measure a human's brain waves and adjust music in a Smartphone app as they change. This product can be used as a digital meditation aid. A smart baby pacifier can measure the temperature of a baby and transmit the same to the Smartphone of parents. IoT is proved to be an emerging technological innovation. In the current context, it is now possible that an helmet of a two wheeler can interact with a car for avoiding collision. Connected toothbrush can now monitor and make one's experience pleasurable .A three dimensional senor of the electric brush can connect with Smartphone apps and provide real time feedback to the person . Many scopes will be created for technology companies to release offerings as per the behaviour of consumers. It may so happen that Netflix can know when a person is sad and alone by monitoring the smart watch, smart thermostat and in-home camera. Subsequently, Netflix may offer a movie to change the mood. In a consumer electronics show in Los Vegas, Samsung informed that the company would invest 100 million dollar for progress of IoT. The company will also promote an open technology ecosystem for facilitating the usage of IoT

**REFERENCES**

[ 1 ] <https://www.hackster.io/>

[ 2 ] <https://github.com/>

[ 3 ] <https://www.electronicshub.org/>

[ 4 ] <https://www.arduino.cc/>

[ 5 ] <https://www.adafruit.io/>

[ 6 ] <https://www.ifttt.com/>

[ 7 ] <https://instrustables.com/>

**APPENDIX – A**

**NODE MCU ESP 8266**

### The ESP8266 is the name of a micro controller designed by Espressif Systems. Digital I/O Just like a normal Arduino, the ESP8266 has digital input/output pins (I/O or GPIO, General Purpose Input/Output pins). As the name implies, they can be used as digital inputs to read a digital voltage, or as digital outputs to output either 0V (sink current) or 3.3V (source current).

#### VOLTAGE AND CURRENT RESTRICTIONS :

The ESP8266 is a 3.3V microcontroller, so its I/O operates at 3.3V as well. The pins are **not 5V tolerant, applying more than 3.6V on any pin will kill the chip.**The maximum current that can be drawn from a single GPIO pin is **12mA**.

#### USABLE PINS :

The ESP8266 has 17 GPIO pins (0-16), however, you can only use 11 of them, because 6 pins (GPIO 6 - 11) are used to connect the flash memory chip GPIO 1 and 3 are used as TX and RX of the hardware Serial port (UART), so in most cases, you can’t use them as normal I/O while sending/receiving serial data.

#### BOOT MODES :

As mentioned in the previous chapter, some I/O pins have a special function during boot: They select 1 of 3 boot modes:

Table 1 : Boot Mode

| **GPIO15** | **GPIO0** | **GPIO2** | **Mode** |
| --- | --- | --- | --- |
| 0V | 0V | 3.3V | Uart Bootloader |
| 0V | 3.3V | 3.3V | Boot sketch (SPI flash) |
| 3.3V | X | x | SDIO mode (not used for Arduino) |

Note: you don’t have to add an external pull-up resistor to GPIO2, the internal one is enabled at boot. We made sure that these conditions are met by adding external resistors in the previous chapter, or the board manufacturer of your board added them for you. This has some implications, however:

* GPIO15 is always pulled low, so you can’t use the internal pull-up resistor. You have to keep this in mind when using GPIO15 as an input to read a switch or connect it to a device with an open-collector (or open-drain) output, like I²C.
* GPIO0 is pulled high during normal operation, so you can’t use it as a Hi-Z input.
* GPIO2 can’t be low at boot, so you can’t connect a switch to it.

#### INTERNAL PULL-UP/-DOWN RESISTORS :

GPIO 0-15 all have a built-in pull-up resistor, just like in an Arduino. GPIO16 has a built-in pull-down resistor.

#### PWM :

Unlike most Atmel chips (Arduino), the ESP8266 doesn’t support hardware PWM, however, software PWM is supported on all digital pins. The default PWM range is 10-bits @ 1kHz, but this can be changed (up to >14-bit@1kHz).

### ANALOG INPUT :

The ESP8266 has a single analog input, with an input range of 0 - 1.0V. If you supply 3.3V, for example, you will damage the chip. Some boards like the NodeMCU have an on-board resistive voltage divider, to get an easier 0 - 3.3V range. You could also just use a trimpot as a voltage divider.

The ADC (analog to digital converter) has a resolution of 10 bits.

### COMMUNICATION :

#### SERIAL :

The ESP8266 has two hardware UARTS (Serial ports):  
UART0 on pins 1 and 3 (TX0 and RX0 resp.), and UART1 on pins 2 and 8 (TX1 and RX1 resp.), however, GPIO8 is used to connect the flash chip. This means that UART1 can only transmit data.

UART0 also has hardware flow control on pins 15 and 13 (RTS0 and CTS0 resp.). These two pins can also be used as alternative TX0 and RX0 pins.

#### I²C :

The ESP doesn’t have a hardware TWI (Two Wire Interface), but it is implemented in software. This means that you can use pretty much any two digital pins. By default, the I²C library uses pin 4 as SDA and pin 5 as SCL. (The data sheet specifies GPIO2 as SDA and GPIO14 as SCL.) The maximum speed is approximately 450kHz.

#### SPI :

The ESP8266 has one SPI connection available to the user, referred to as HSPI. It uses GPIO14 as CLK, 12 as MISO, 13 as MOSI and 15 as Slave Select (SS). It can be used in both Slave and Master mode (in software).

### GPIO OVERVIEW :

| **GPIO** | **Function** | **State** | **Restrictions** |
| --- | --- | --- | --- |
| 0 | Boot mode select | 3.3V | No Hi-Z |
| 1 | TX0 | - | Not usable during Serial transmission |
| 2 | Boot mode select TX1 | 3.3V (boot only) | Don’t connect to ground at boot time  Sends debug data at boot time |
| 3 | RX0 | - | Not usable during Serial transmission |
| 4 | SDA (I²C) | - | - |
| 5 | SCL (I²C) | - | - |
| 6 – 11 | Flash connection | x | Not usable, and not broken out |
| 12 | MISO (SPI) | - | - |
| 13 | MOSI (SPI) | - | - |
| 14 | SCK (SPI) | - | - |
| 15 | SS (SPI) | 0V | Pull-up resistor not usable |
| 16 | Wake up from sleep | - | No pull-up resistor, but pull-down instead  Should be connected to RST to wake up |

|  |  |  |
| --- | --- | --- |
|  |  | Table 2 : GPIO Overview |

**APPENDIX – B**

**Program for the NODE MCU ESP 8266**

#include <ESP8266WiFi.h>

#include "Adafruit\_MQTT.h"

#include "Adafruit\_MQTT\_Client.h"

#define Relay1 D1

#define Relay2 D2

#define Relay3 D3

#define Relay4 D4

#define WLAN\_SSID "...your SSID..."

#define WLAN\_PASS "...your password..."

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

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

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

#define AIO\_USERNAME "...your AIO username (see https://accounts.adafruit.com)..."

#define AIO\_KEY "...your AIO key..."

//Set up MQTT and WiFi clients

WiFiClient client;

Adafruit\_MQTT\_Client mqtt(&client, MQTT\_SERV, MQTT\_PORT, MQTT\_NAME, MQTT\_PASS);

//Set up the feed you're subscribing to

Adafruit\_MQTT\_Subscribe onoff = Adafruit\_MQTT\_Subscribe(&mqtt, MQTT\_NAME "/f/onoff");

void setup()

{

Serial.begin(9600);

//Connect to WiFi

Serial.print("\n\nConnecting Wifi... ");

WiFi.begin(WIFI\_SSID, WIFI\_PASS);

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

}

Serial.println("OK!");

//Subscribe to the onoff feed

mqtt.subscribe(&onoff);

pinMode(LED\_BUILTIN, OUTPUT);

pinMode(Relay1, OUTPUT);

pinMode(Relay2, OUTPUT);

pinMode(Relay3, OUTPUT);

pinMode(Relay4, OUTPUT);

digitalWrite(LED\_BUILTIN, HIGH);

digitalWrite(Relay1, HIGH);

digitalWrite(Relay2, HIGH);

digitalWrite(Relay3, HIGH);

digitalWrite(Relay4, HIGH);

}

void loop()

{

MQTT\_connect();

//Read from our subscription queue until we run out, or

//wait up to 5 seconds for subscription to update

Adafruit\_MQTT\_Subscribe \* subscription;

while ((subscription = mqtt.readSubscription(5000)))

{

//If we're in here, a subscription updated...

if (subscription == &onoff)

{

//Print the new value to the serial monitor

Serial.print("onoff: ");

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

//If the new value is "ON", turn the light on.

//Otherwise, turn it off.

if (!strcmp((char\*) onoff.lastread, "ON"))

{

//Active low logic

digitalWrite(LED\_BUILTIN, LOW);

digitalWrite(Relay1, LOW);

delay(1000);

digitalWrite(Relay2, LOW);

delay(1000);

digitalWrite(Relay3, LOW);

delay(1000);

digitalWrite(Relay4, LOW);

delay(1000);

}

else

{

digitalWrite(LED\_BUILTIN, HIGH);

digitalWrite(Relay1, HIGH);

digitalWrite(Relay2, HIGH);

digitalWrite(Relay3, HIGH);

digitalWrite(Relay4, HIGH);

}

}

}

// ping the server to keep the mqtt connection alive

if (!mqtt.ping())

{

mqtt.disconnect();

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Adafruit MQTT Library ESP8266 Example

Must use ESP8266 Arduino from:

https://github.com/esp8266/Arduino

Works great with Adafruit's Huzzah ESP board & Feather

----> https://www.adafruit.com/product/2471

----> https://www.adafruit.com/products/2821

Adafruit invests time and resources providing this open source code,

please support Adafruit and open-source hardware by purchasing

products from Adafruit!

Written by Tony DiCola for Adafruit Industries.

MIT license, all text above must be included in any redistribution

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void MQTT\_connect()

{

int8\_t ret;

// Stop if already connected.

if (mqtt.connected())

{

return;

}

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

uint8\_t retries = 3;

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!");

}