

SMART BUILDING AND HOME AUTOMATION BASED ON VOICE AND GESTURES

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND TELECOMMUNICATION

KHAN FAISAL 15ET30

KHAN NAVAIID 15ET33

KHAN SAHEER 15ET34

SAYYED ABDUL 15ET39

UNDER THE GUIDANCE OF
PROF. RIYAZ PATHAN



Department of Electronics and Telecommunication Engineering

Anjuman-I-Islam's Kalsekar Technical Campus
Sector 16, New Panvel, Navi Mumbai
University of Mumbai

2018-19

CERTIFICATE



Department of Electronics and Telecommunication Engineering

Anjuman-I-Islam's Kalsekar Technical Campus
Sector 16, New Panvel, Navi Mumbai.
University of Mumbai

This is to certify that the project entitled **Smart Building And Home Automation Based On Voice And Gestures** is a bonafide work of **Khan Faisal (15ET30), Khan Navaid (15ET33), Khan Saheer (15ET34), Sayyed Abdul (15ET39)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of Bachelor of Engineering in Department of Electronics and Telecommunication Engineering.

Supervisor

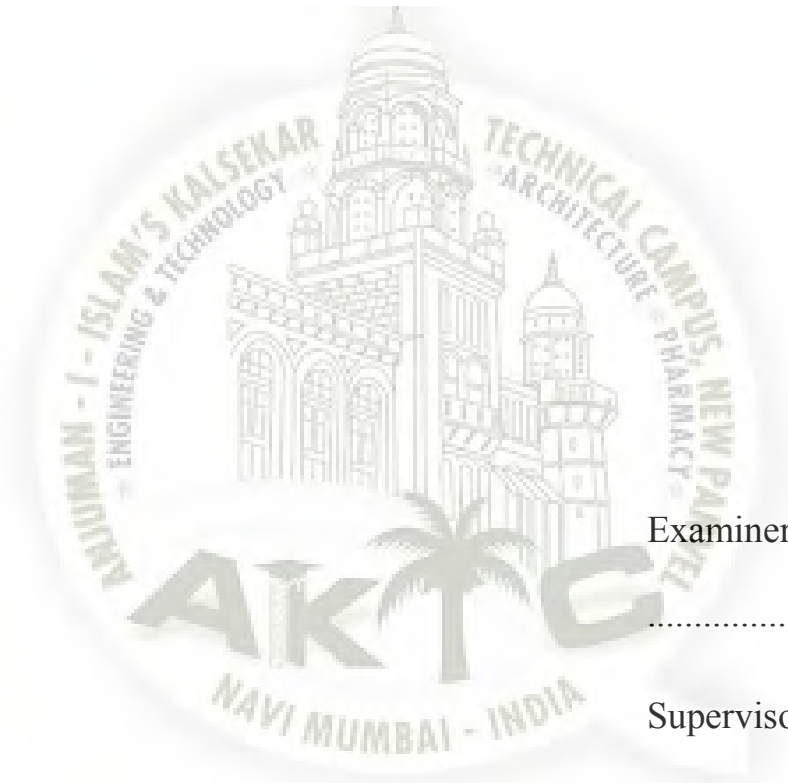
Examiner

Head of Department

Director

Project Report Approval for Bachelor of Engineering

This project entitled "**Smart Building And Home Automation Based On Voice And Gestures**" by **Khan Mohd Faisal, Khan Navaid, Khan Saheer, Sayyed Abdul** is approved for the degree of **Bachelor of Engineering in Electronics and Telecommunication** .



Examiner

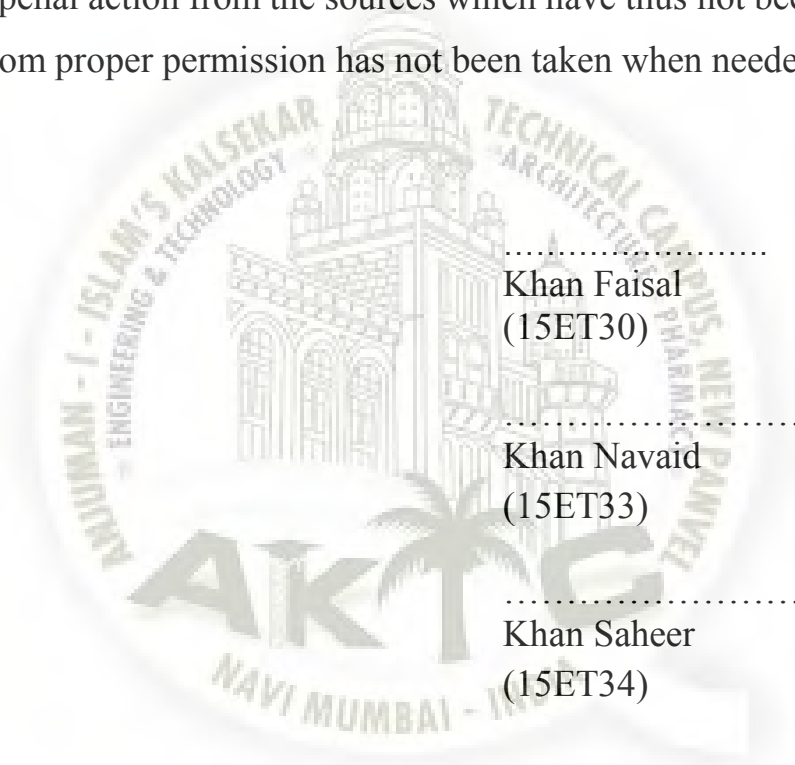
Supervisor

DATE:

PLACE:

Declaration

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.



.....
Khan Faisal
(15ET30)

.....
Khan Navaid
(15ET33)

.....
Khan Saheer
(15ET34)

.....
Sayyed Abdul
(15ET39)

Date:

Acknowledgment

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

We are highly indebted to Prof. Riyaz Pathan for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

We would like to express my gratitude towards my parents & Staff of Anjuman-I-Islam's Kalsekar Technical Campus for their kind co-operation and encouragement which help me in completion of this project.

We My thanks and appreciations also go to my colleague in developing the project and people who have willingly helped me out with their abilities.

Khan Faisal	(15ET30)
Khan Navaid	(15ET33)
Khan Saheer	(15ET34)
Sayyed Abdul	(15ET39)

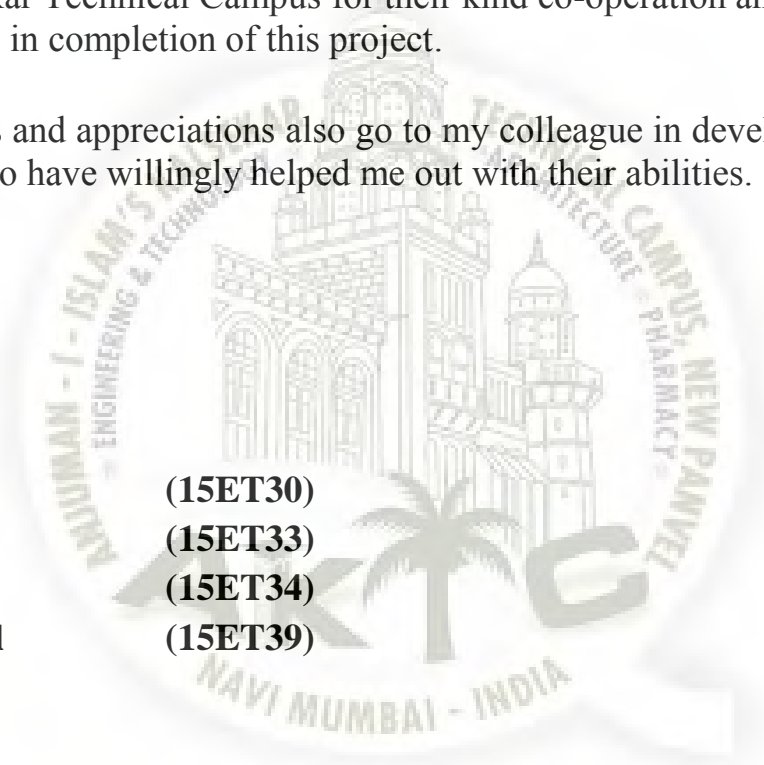
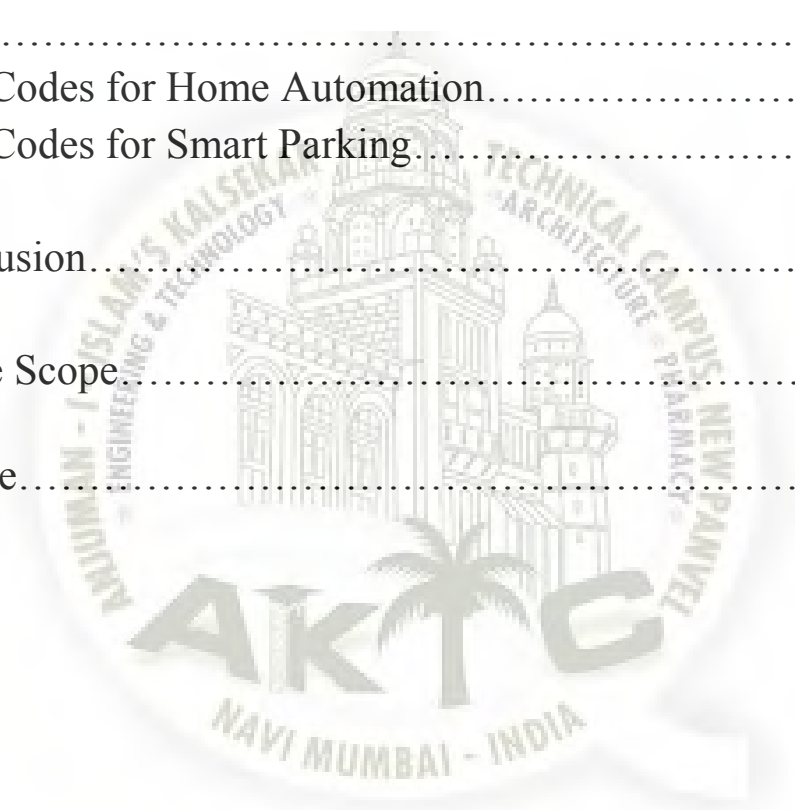


TABLE OF CONTENTS

Abstract.....	1
1. Introduction.....	2
1.1 what is IOT???	3
2. Hardware Specification.....	6
2.1 NodeMCU.....	6
2.2 Analog Multiplexer.....	7
2.3 Relay.....	8
2.4 Interfacing of relay with NodeMCU.....	9
2.5 Sensor Used.....	10
2.5.1 Ultrasonic Sensor.....	10
2.5.2 Temperature Sensor.....	11
2.5.3 PIR Sensor.....	12
2.5.4 IR Sensor.....	13
3. Software Specification.....	14
3.1 Arduino.....	14
3.1.1 Why Arduino???	15
3.1.2 How do we use Arduino???	17
3.2 IFTTT.....	18
3.2.1 How it Works???	19
3.2.2 Google Assistant.....	19
3.2.3 Process.....	20
3.3 Adafruit IO.....	22
4. Smart parking System Using AWS.....	24
4.1 What is AWS???	24
4.2 What is Cloud Computing???	24
4.3 Features of Amazon EC2.....	25

5. Working of AWS.....	27
5.1 Cloud Products of AWS.....	27
5.2 Node Red.....	28
5.3 Process.....	30
5.4 QR Code.....	31
6. Proposed System.....	32
6.1 Flow Chart.....	33
7. Codes.....	34
7.1 Codes for Home Automation.....	34
7.2 Codes for Smart Parking.....	41
8. Conclusion.....	46
9. Future Scope.....	47
Reference.....	48



ABSTRACT

Voice Based Home Automation System using ESP8266 NodeMCU. It is the idea which corresponds to the new era of automation and technology. The main aim of the home automation system is to make life easier. Mobile devices are very common among everyone due to its user friendly interface and portability features. In this project we aim to control electrical home appliances by android voice commands using Wi-Fi as communication protocol and Android device and to add some other application to make a smart building.

This system is basically designed for the visually challenged people to aid them in operating the home appliances individually. In this system MEMS accelerometer is used to detect hand motion and transmitted using Radio Frequency (RF). This proposed system used to control the function of various home appliances. This paper presents a low cost and 3 axis wireless accelerometers based system to control the Home Electronic devices. Advancement in industrialization leaves the parking management system outdated. Traffic condition in foreign countries are homogeneous whereas India's traffic condition is heterogeneous which makes real-time management at parking lot difficult. Traffic congestion at parking lot has become often, that people cannot even find the place to park their vehicle especially in large companies where first come first serve method is used for parking vehicle. In nutshell, parking condition needs to be improved by introducing Internet of Things (IoT) to parking lots. With the help of Microcontrollers and Image processing.

The physically challenged people have many difficulties in their day to day life. Mostly to control the home appliances is very difficult task for them and some of the people are still illiterate. In order to aid them, gesture based control of home appliances without the sign language is proposed in this report. The various components used are Flex sensor, XBee Transceiver and DC motor. The flex sensor is calibrated initially for every position using microcontroller. The data from flex sensor are transferred to wireless transmitter through microcontroller. It is received by wireless receiver and send to microcontroller which controls home appliances. The gesture made by the physically challenged people is also displayed in LCD.

1. INTRODUCTION

In today's day to day life automation can play a major role. Automation makes things simple. The main attraction of any automated system is reducing human labor, efforts, time and errors due to human negligence. This project is based on Internet of Things (IoT). Internet of Things is a network of devices such as electrical appliances for connectivity which enables these devices to connect and exchange data. This project represents a flexible way to control devices. In this project we are working on an android application where a user will provide voice commands for controlling devices such as "Turn light on" which will be connected to Wifi module. This automation can be used majorly not only in home but offices and hospitals also user can register and authenticate himself/herself in android device and after successful login can give the input commands and operate the devices. It also provides security from third party users. It allows controlling number of home appliances simultaneously. Python is used as the main programming language which is default, provided by Raspberry Pi.

Traffic is one of the banes of urban life in India. While in developed countries, even the richest of the rich use public transport, in India there is a class divide when it comes to using public transport. This has led to an exorbitant increase in the number of private vehicles leading to an increasing number of traffic snarls [1]. One of the major problems due to increase of vehicles are the problems with parking lots. Vehicle management at the parking lots have become an important aspect for the best utilization of existing parking area capacity. Wastage of time and fuel at the parking area due to increase in vehicle count have become a major issue for big companies like Infosys, Accenture etc who are making multilevel parking lot facilities for their employees.

The Flex sensor is used by the physically challenged people for gesture purpose. The Flex sensor is similar to a variable resistor but it is a flexible sensor. As we bend the Flex sensor, the resistance value changes. It is a resistive sensor and in order to obtain output voltage, a voltage divider circuit is used. The wireless transmitter and receiver used is XBee transceiver. The XBee provides wireless communication at the speed of up to 250 kbps. The main purpose of this work is to help the illiterate and physically challenged people.

Basic purpose of developing a new system of hand gesture remote control is to remove the need of the hand held remote. Gesture is an action of arms or any other body part which are made to emphasize speech. Basically Gestures include motion of the hands and face. A gesture can be divided into different categories: dynamic gesture and static gesture. Gesture recognition is movement of human action by a computing device. Gestures can be obtained from any bodily motion but commonly obtained from the face or hand.

1.1 WHAT IS IOT???

The Internet of things (IoT) is the inter-networking of physical devices, vehicles (also referred to as "connected devices" and "smart devices"), buildings, and other items embedded with electronics, software, sensors, actuators, and network connectivity that enable these objects to collect and exchange data. In 2013 the Global Standards Initiative on Internet of Things (IoT-GSI) defined the IoT as "the infrastructure of the information society. The IoT allows objects to be sensed or controlled remotely across existing network infrastructure,[4] creating opportunities for more direct integration of the physical world into computer-based systems, and resulting in improved efficiency, accuracy and economic benefit in addition to reduced human intervention. When IoT is augmented with sensors and actuators, the technology becomes an instance of the more general class of cyber-physical systems, which also encompasses technologies such as smart grids, virtual power plants, smart homes, intelligent transportation and smart cities. Each thing is uniquely identifiable through its embedded computing system but is able to interoperate within the existing Internet infrastructure. Experts estimate that the IoT will consist of almost 50 billion objects by 2020.

Typically, IoT is expected to offer advanced connectivity of devices, systems, and services that goes beyond machine-to-machine(M2M) communication and covers a variety of protocols, domains, and applications. The interconnection of these embedded devices (including smart objects), is expected to usher in automation in nearly all fields, while also enabling advanced applications like a smart grid, and expanding to areas such as smart cities.

"Things," in the IoT sense, can refer to a wide variety of devices such as heart monitoring implants, biochip transponders on farm animals, electric clams in coastal waters, automobiles with built-in sensors, DNA analysis devices for environmental/food/pathogen monitoring or field operation devices that assist firefighters in search and rescue operations. Legal scholars suggest to look at "Things" as an "inextricable mixture of hardware, software, data and service". These devices collect useful data with the help of various existing technologies and then autonomously flow the data between other devices. Current market examples include home automation (also known as smart home devices) such as the control and automation of lighting, heating (like smart thermostat), ventilation, air conditioning (HVAC) systems, and appliances such as washer/dryers, robotic vacuums, air purifiers, ovens or refrigerators/freezers that use Wi-Fi for remote monitoring.

As well as the expansion of Internet-connected automation into a plethora of new application areas, IoT is also expected to generate large amounts of data from diverse locations, with the consequent necessity for quick aggregation of the data, and an increase in the need to index, store, and process such data more effectively. IoT is one of the platforms of today's Smart City, and Smart Energy Management Systems. The term "the Internet of Things" was coined by Kevin Ashton of Procter & Gamble, later MIT's Auto-ID Center, in 1999.

The Internet of Things is an emerging topic of technical, social, and economic significance. Consumer products, durable goods, cars and trucks, industrial and utility components, sensors, and other everyday objects are being combined with Internet connectivity and powerful data analytic capabilities that promise to transform the way we work, live, and play. Projections for the impact of IoT on the Internet and economy are impressive, with some anticipating as many as 100 billion connected IoT devices and a global economic impact of more than \$11 trillion by 2025.

IoT Definitions: The term Internet of Things generally refers to scenarios where network connectivity and computing capability extends to objects, sensors and everyday items not normally considered computers, allowing these devices to generate, exchange and consume data with minimal human intervention. There is, however, no single, universal definition.

Enabling Technologies: The concept of combining computers, sensors, and networks to monitor and control devices has existed for decades. The recent confluence of several technology market trends, however, is bringing the Internet of Things closer to widespread reality. These include Ubiquitous Connectivity, Widespread Adoption of IP-based Networking, Computing Economics, Miniaturization, Advances in Data Analytics, and the Rise of Cloud Computing.

Connectivity Models: IoT implementations use different technical communications models, each with its own characteristics. Four common communications models described by the Internet Architecture Board include: *Device-to-Device*, *Device-to-Cloud*, *Device-to-Gateway*, and *Back-End Data-Sharing*. These models highlight the flexibility in the ways that IoT devices can connect and provide value to the user.

Transformational Potential: If the projections and trends towards IoT become reality, it may force a shift in thinking about the implications and issues in a world where the most common interaction with the Internet comes from passive engagement with connected objects rather than active engagement with content. The potential realization of this outcome – a “hyperconnected world” — is testament to the general-purpose nature of the Internet architecture itself, which does not place inherent limitations on the applications or services that can make use of the technology.



2. HARDWARE SPECIFICATION

2.1 NODEMCU

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 Lua 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. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project,^[15] enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

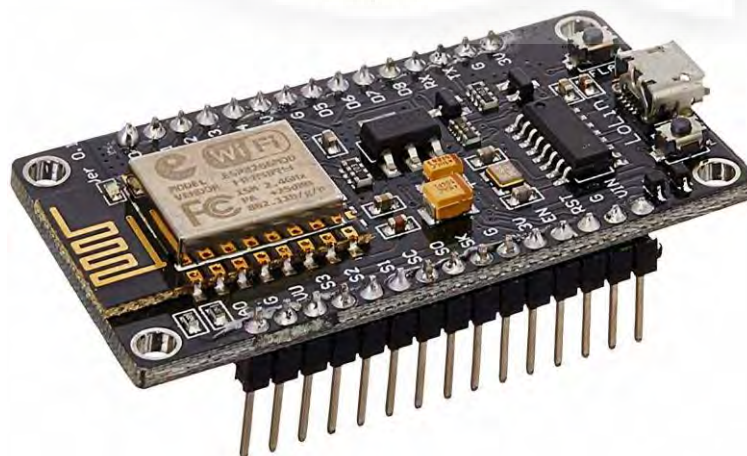


Fig. 2.1 NodeMCU

2.2 ANALOG MULTIPLEXER

In electronics, a multiplexer (or mux) is a device that selects between several analog or digital input signals and forwards it to a single output line. A multiplexer of inputs has select lines, which are used to select which input line to send to the output. Multiplexers are mainly used to increase the amount of data that can be sent over the network within a certain amount of time and bandwidth. A multiplexer is also called a data selector. Multiplexers can also be used to implement Boolean functions of multiple variables.

An electronic multiplexer makes it possible for several signals to share one device or resource, for example, one A/D converter or one communication line, instead of having one device per input signal.

Conversely, a demultiplexer (or demux) is a device taking a single input signal and selecting one of many data-output-lines, which is connected to the single input. A multiplexer is often used with a complementary demultiplexer on the receiving end.

An electronic multiplexer can be considered as a multiple-input, single-output switch, and a demultiplexer as a single-input, multiple-output switch. The schematic symbol for a multiplexer is an isosceles trapezoid with the longer parallel side containing the input pins and the short parallel side containing the output pin. The schematic on the right shows a 2-to-1 multiplexer on the left and an equivalent switch on the right. The wire connects the desired input to the output.



Fig. 2.2 Analog Multiplexer

2.3 RELAY

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

Relays are used to control high voltage circuits with the help of low voltage signals. Similarly they are used to control high current circuits with the help of low current signals. They are also used as protective relays. By this function all the faults during transmission and reception can be detected and isolated.

The control circuit functions as the coupling between the input and output circuits. In electromechanical relays, the coil accomplishes this function. A relay's Output Circuit is the portion of the relay that switches on the load and performs the same function as the mechanical contacts of electromechanical relays.



Fig.2.3 Relay

2.4 INTERFACING OF RELAY WITH NODEMCU

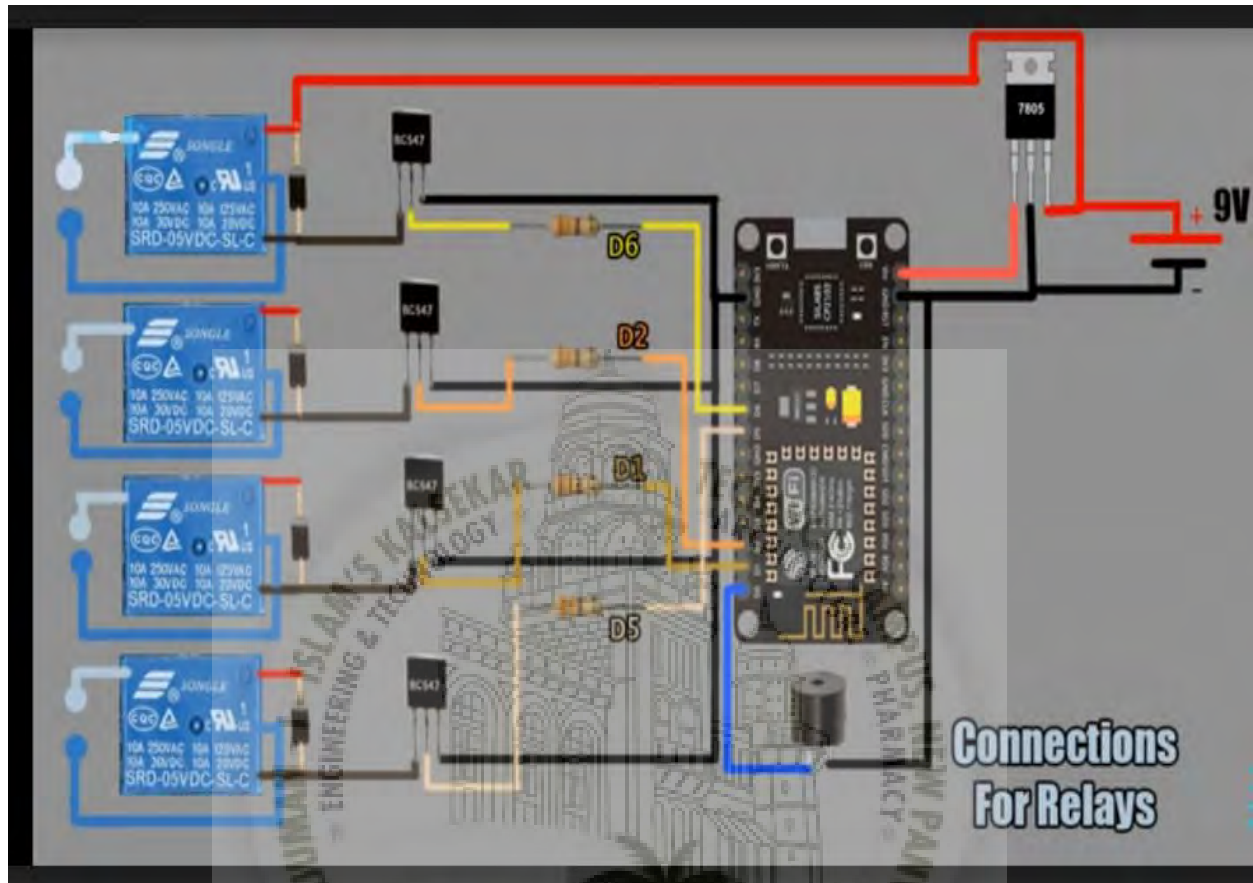


Fig. 2.4 Interfacing of Relay With NodeMCU

2.5 SENSORS USED

2.5.1 ULTRASONIC SENSOR

Ultrasonic sound vibrates at a frequency above the range of human hearing. Our ultrasonic sensors, like many others, use a single transducer to send a pulse and to receive the echo. The sensor determines the distance to a target by measuring time lapses between the sending and receiving of the ultrasonic pulse.

The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

Description. The HC-SR04 ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. From 2cm to 400 cm or 1” to 13 feet. Ultrasonic sensors are used around the world, indoors and outdoors in the harshest conditions, for a variety of applications. Our ultrasonic sensors, made with piezoelectric crystals, use high frequency sound waves to resonate a desired frequency and convert electric energy into acoustic energy, and vice versa. Ultrasonic Transmitter and Receiver. Ultrasonic sound is a cyclic sound pressure with a high frequency than the upper limit of human hearing equal to 20KHz. Some animals like dolphins, mice, dogs, and bats have a high- frequency limit that is larger than that of the human ear & thus can hear ultrasound.



Fig 2.5 Ultrasonic Sensor

2.5.2 TEMPERATURE SENSOR

Contact sensors include thermocouples and thermistors that touch the object they are to measure, and noncontact sensors measure the thermal radiation a heat source releases to determine its temperature. The latter group measures temperature from a distance and often are used in hazardous environments.

DHT11 is a Humidity and Temperature Sensor, which generates calibrated digital output. DHT11 can be interface with any microcontroller like Arduino, Raspberry Pi, etc. and get instantaneous results. DHT11 is a low-cost humidity and temperature sensor which provides high reliability and long term stability.

All the DHT11 Sensors are accurately calibrated in the laboratory and the results are stored in the memory. A single wire communication can be established between any microcontroller like Arduino and the DHT11 Sensor. Also, the length of the cable can be as long as 20 meters. The data from the sensor consists of integral and decimal parts for both Relative Humidity (RH) and temperature. The data from the DHT11 sensor consists of 40 – bits and the format is as follows: 8 – Bit data for integral RH value, 8 – Bit data for decimal RH value, 8 – Bit data for integral Temperature value, 8 – Bit data for integral Temperature value and 8 – Bit data for checksum.



Fig. 2.6 Temperature Sensor

2.5.3 PIR SENSOR

When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls or outdoors. When a warm body like a human or animal passes by, it first intercepts one half of the PIR sensor, which causes a positive differential change between the two halves. 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.

The motion sensors work in the dark. The way that a motion sensor detects movement has nothing to do with how light or dark it is in the immediate area. Since infrared energy is present and can be detected regardless of the amount of light in the environment, a PIR motion sensor will work just fine in the dark. Typically, motion detectors use infra-red rays to detect motion and they work just fine in complete darkness. Motion detectors do not need to emit any light or infra-red of their own, as they rely on picking up infra red radiation given off by people.



Fig. 2.7 PIR Sensor

2.5.4 IR SENSOR

Infrared Obstacle Sensor Module has built in IR transmitter and IR receiver that sends out IR energy and looks for reflected IR energy to detect presence of any obstacle in front of the sensor module. The sensor has very good and stable response even in ambient light or in complete darkness.

Infrared sensors work on the principle of reflected light waves. ... Infrared sensors can't work in dark environments while Ultrasonic Sensors can. Brighter surfaces are easier to detect for Infrared than dark surfaces, as the sensor doesn't detect darker surfaces.

Infrared proximity sensor made by Sharp. Part # GP2Y0A02YK0F has an analog output that varies from 2.8V at 15cm to 0.4V at 150cm with a supply voltage between 4.5 and 5.5VDC. The sensor has a Japanese Solderless Terminal (JST) Connector. An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. These types of sensors measures only infrared radiation, rather than emitting it that is called as a passive IR sensor.



Fig. 2.8 IR Sensor

3. SOFTWARE SPECIFICATION

3.1 ARDUINO

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike.

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

3.1.1 WHY ARDUINO???

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.

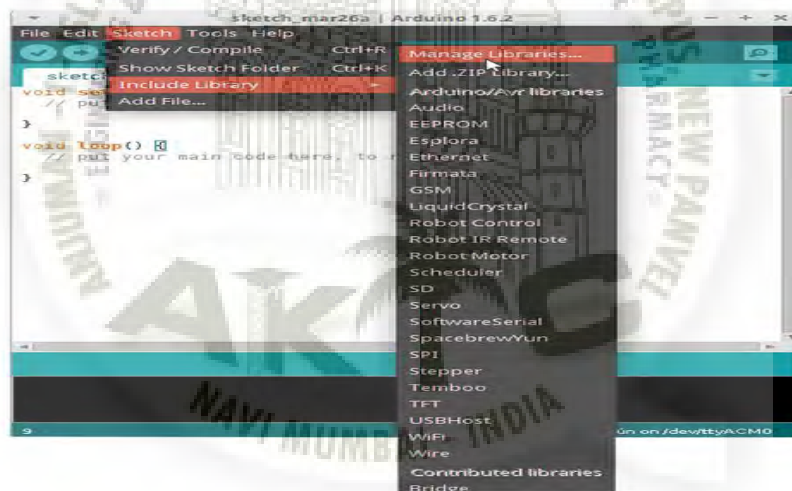


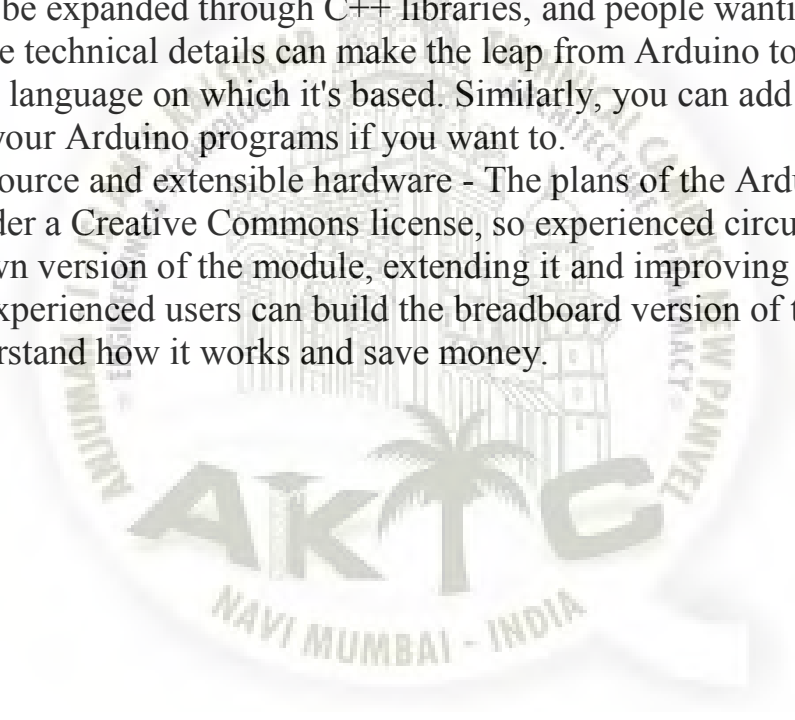
Fig. 3.1 Arduino IDE

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems.

Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50. **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows. **Simple, clear programming environment** - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.

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

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



3.1.2 HOW DO WE USE ARDUINO???

See the getting started guide. If you are looking for inspiration you can find a great variety of Tutorials on Arduino Project Hub. The text of the Arduino getting started guide is licensed under a Creative Commons Attribution-Share like 3.0 License. Code samples in the guide are released into the public domain. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. This is a screenshot of the Arduino IDE.

Arduino software has been used as the interface between software and hardware of this project. Microcontroller needs a program to operate and execute the process associated with proposed design. It is easy to verify and compile after writing the code. The complete flowchart which gives the whole operation of system is shown in figure-6. The complete flowchart which indicates the whole operation of the system and controlled by a mobile app. The focus of this project is to bring automation in home or industries Firstly, the Wi-Fi shield (Wi-Fi hotspot) connect to the existing network infrastructure& it initializing blynk server which is of open source server ,The Wi-Fi module send single to app that provide for the client(operator) indicating system is in online or offline then it check the input-output pins i.e.switch-1,2,3,4,5,6,7,&8.If the client (operator) switches any of the switch the data will be received by blynk server and give status return to the user by display it on LCD provided in app. Lastly, this process is in continues operation the system will loop to the initial condition.

NODEMCU (esp8266) has been selected as the controller for this system due to its compact size, compatibility, easy interfacing over several other type of controller including Programmable Integrated Circuit (PIC), Programmable Logic Controller (PLC) and others. ESP8266 is an open source firmware that is built on top of the chip manufacturer's proprietary SDK. The firmware provides a simple programming environment, which is a very simple and fast scripting language The ESP8266 chip incorporates 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-3 shows the diagram of NODEMCU (ESP8266).It has Processor called L106 32bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 MHz and has a memory of 32 Kbit instruction RAM ,32 Kbit instruction cache RAM, 80 Kbit user data RAM&16 Kbit ETS system data RAM. It has inbuilt Wi-Fi module of IEEE 802.11.

3.2 IFTTT

If This Then That, 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, Instagram, Telegram. 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.^[8] All of the functionalities of the Do suite of apps have since been integrated into a redesigned IFTTT app.

Home, it is the place where one fancies or desires to be after a long tiring day. People come home exhausted after a long hard working 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 their lights on or off, or play their favorite music etc. on a go with their voice with the aid of their smart phones would make their home more comfortable.

3.2.1 HOW IT WORKS

Create a free account. Browse the IFTTT website or app to find an Applet that interests you. Click into the Applet and turn it on. Connect the services that are involved in the Applet — this is only so we can use them to run Applets on your behalf. Find more Applets, and repeat. Over 600 apps work with IFTTT including Twitter, Telegram, Google Drive, Twitch, Weather Underground, Instagram, Gmail, and devices like Google Home, Amazon Alexa, Nest, Philips Hue, and your iPhone. The IFTTT app also integrates with the Health app, so you can easily track and maintain your habits.

3.2.2 GOOGLE ASSISTANT

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, Google Assistant can engage in two-way conversations.

Assistant initially debuted in May 2016 as part of Google's messaging app Allow, 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 2017. 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 third party smart home appliances. The functionality of the Assistant can also be enhanced by third-party developers.

In 2017, Google Assistant was installed on more than 400 million devices. Users primarily interact with 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.

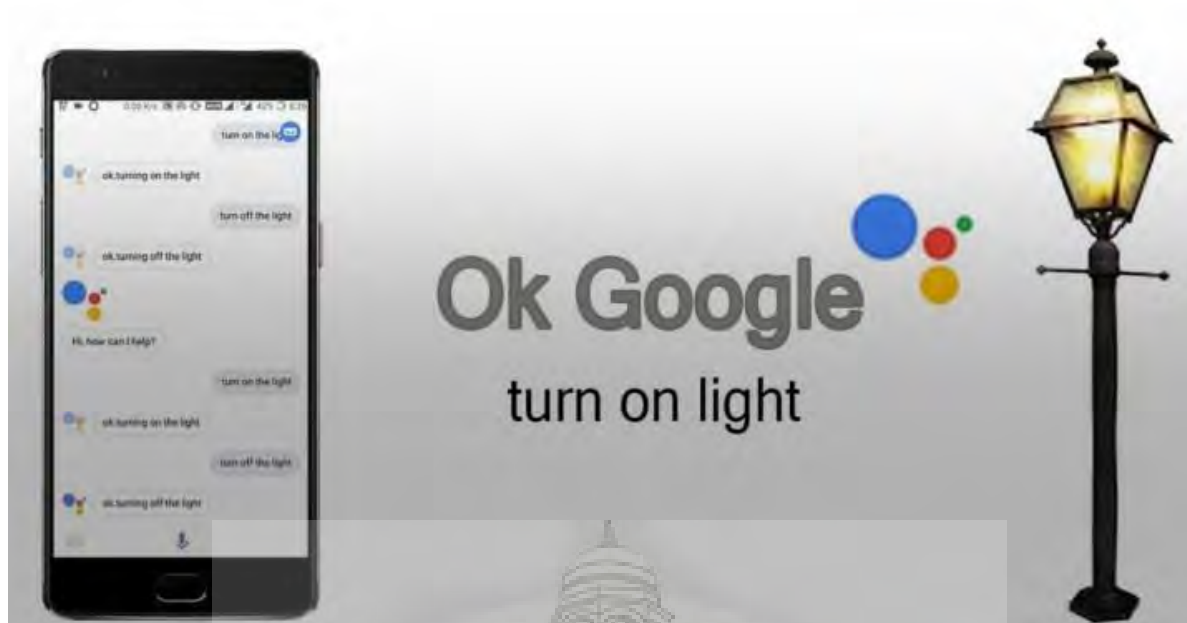


Fig. 3.2 Working of IFTTT With Google Assistant

3.2.3 PROCESS

For the hardware we need to have a relay to switch AC appliances which is operated through WiFi. For coding the ESP8266 we are going to use Adafruit MQTT Library which you can download from my Github account. In that library, we are just going to modify the example code called “mqtt_esp8266”.

IFTTT stands for If This Then That which basically provides a platform on which we can merge two different services.

Like for our project, we are going to use Google assistant and Adafruit MQTT. So any instruction coming from Google assistant will be processed by IFTTT and accordingly the actions will be performed on Adafruit MQTT server side.

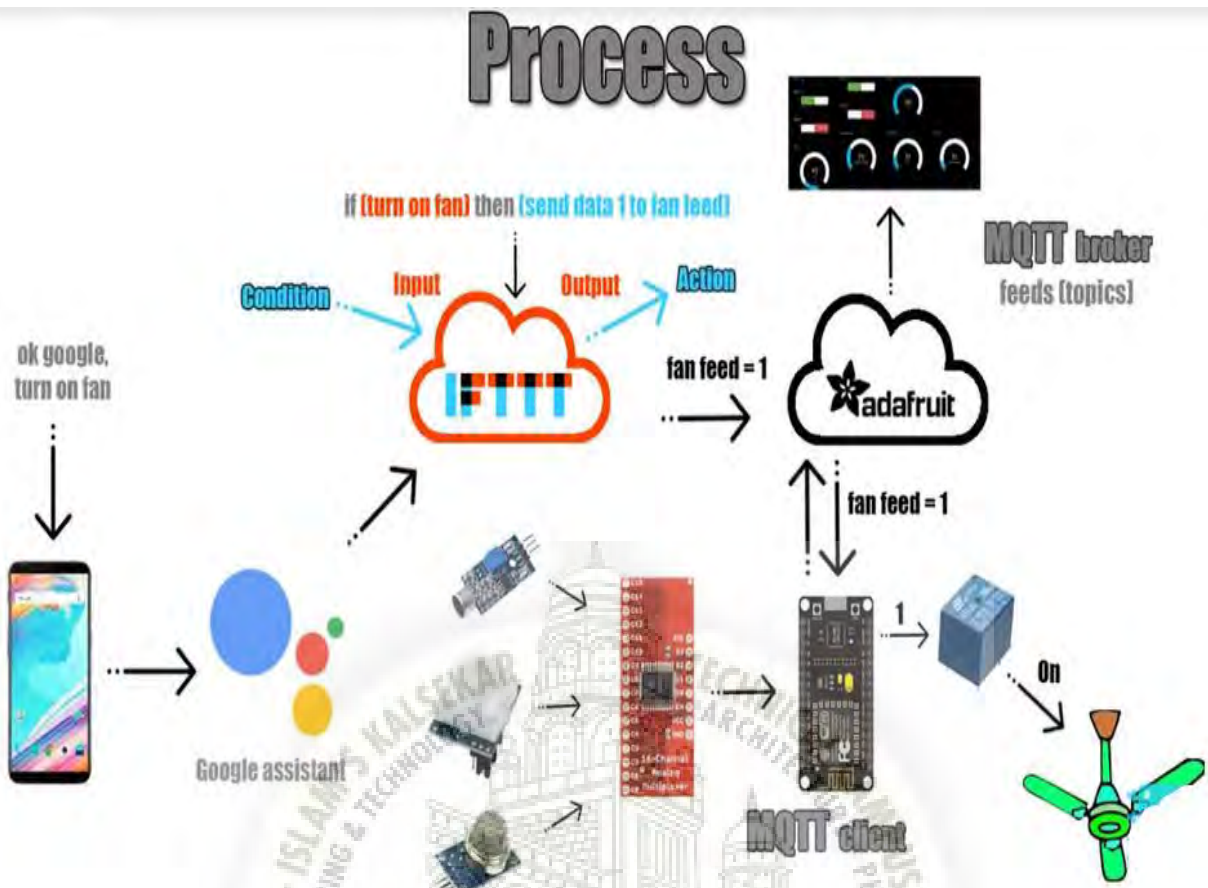


Fig. 3.3 Process

3.3 ADAFRUIT IO

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.

The Feather development boards constitute Adafruit's broadest platform of "Arduino-like" boards. The boards all share similarities in that they have the same form factor, same pinout, similar microcontrollers, feature lithium polymer battery charging. Each board has a special feature in addition to the microcontroller breakout, such as Bluetooth, Wi-Fi or cellular network connectivity or built-in prototyping space or SD card communication. The name "Feather" comes from the fact that the boards are small, thin, light and easily powered from a battery. In addition to the boards themselves, Adafruit engineers and manufactures "Feather Wings", which are expansion cards allowing the addition of features such as an LCD, a Neo Pixel array or DC motor drivers.

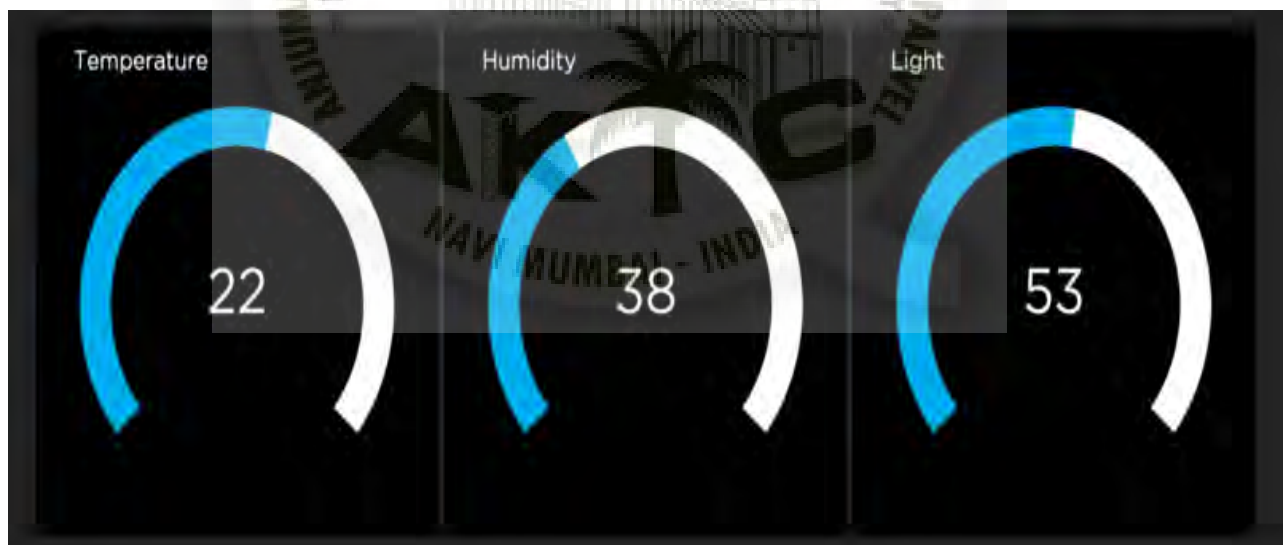


Fig. 3.4 Adafruit Dashboard

Once you have located the right folder you can go ahead and right-click and select Copy from the drop down menu. Then to install the library you navigate to the Library folder located inside your Arduino IDE install location. By default it should be located at: Program Files (x86) -> Arduino -> libraries.

First, download the library as a ZIP, which is done by clicking the green “Clone or download” button and then clicking “Download ZIP”. Once downloaded, go to the Arduino IDE and click Sketch > Include Library > Add .zip Library. In the file dialogue windows that opens, locate your downloaded ZIP file.



Fig. 3.5 Interfacing Google Assistant with Adafruit IO

4. SMART PARKING SYSTEM USING AWS

Everyone is familiar with the chaos in the parking spaces. Imagine, when you reach a parking area and you don't find a parking space. In such a situation, the owner will park his/her vehicle in an NO PARKING zone and in some cases have to pay the fine for the same.

Now, Imagine a situation where you access the parking area from anywhere using just by typing an IP in your web browser on your mobile from your home or scanning the QR Code near the parking area. This will save a lot of valuable time and also prevent you from paying fine. Let's now get to the technical part.

For our project we are using Amazon Web Services(AWS).

4.1 WHAT IS AWS???

Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet. A cloud services platform, such as Amazon Web Services, owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

4.2 WHAT IS CLOUD COMPUTING???

Cloud computing is the on-demand delivery of compute power, database storage, applications, and other IT resources through a cloud services platform via the Internet with pay-as-you-go pricing. Whether you are running applications that share photos to millions of mobile users or you're supporting the critical operations of your business, a cloud services platform provides rapid access to flexible and low-cost IT resources. With cloud computing, you don't need to make large upfront investments in hardware and spend a lot of time on the heavy lifting of managing that hardware. Instead, you can provision exactly the right type and size of computing resources you need to power your newest bright idea or operate your IT department. You can access as many resources as you need, almost instantly, and only pay for what you use. Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet. A cloud services platform, such as Amazon Web Services, owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

Global Infrastructure AWS serves over a million active customers in more than 190 countries. We are steadily expanding global infrastructure to help our customers achieve lower latency and higher throughput, and to ensure that their data resides only in the AWS Region they specify. As our customers grow their businesses, AWS will continue to provide infrastructure that meets their global requirements.

The AWS Cloud infrastructure is built around AWS Regions and Availability Zones. An AWS Region is a physical location in the world where we have multiple Availability Zones. Availability Zones consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities. These Availability Zones offer you the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center. The AWS Cloud operates 60 Availability Zones within 20 geographic Regions around the world, with announced plans for 12 more Availability Zones and four more Regions.

4.3 AMAZON EC2 PROVIDES THE FOLLOWING FEATURES

Virtual computing environments, known as instances Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package bits you need for your server (including the operating system and additional software) Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place)

Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as instance store volumes Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as Regions and Availability Zones A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses Metadata, known as tags, that you can create and assign to your Amazon EC2 resources Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

Amazon EC2 provides a web-based user interface, the Amazon EC2 console. If you've signed up for an AWS account, you can access the Amazon EC2 console by signing into the AWS Management Console and selecting EC2 from the console home page. If you prefer to use a command line interface, you have the following options: AWS Command Line Interface (CLI) Provides commands for a broad set of AWS products, and is supported on Windows, Mac, and Linux. To get started, see AWS Command Line Interface User Guide. For more information about the commands for Amazon EC2, see `ec2` in the AWS CLI Command Reference. AWS Tools for Windows PowerShell.

Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see the AWS Tools for Windows PowerShell User Guide. For more information about the cmdlets for Amazon EC2, see the AWS Tools for PowerShell Cmdlet Reference. Amazon EC2 provides a Query API. These requests are HTTP or HTTPS requests that use the HTTP verbs GET or POST and a Query parameter named Action. For more information about the API actions for Amazon EC2, see Actions in the Amazon EC2 API Reference.



5. WORKING OF AWS

5.1 CLOUD PRODUCTS OF AWS

Amazon Web Services offers a broad set of global cloud-based products including compute, storage, databases, analytics, networking, mobile, developer tools, management tools, IoT, security and enterprise applications. These services help organizations move faster, lower IT costs, and scale. AWS is trusted by the largest enterprises and the hottest start-ups to power a wide variety of workloads including: web and mobile applications, game development, data processing and warehousing, storage, archive, and many others.



Fig. 5.1 Working of NodeRed

5.2 NODE RED

Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things.

Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions. Elements of applications can be saved or shared for re-use. The runtime is built on Node.js. The flows created in Node-RED are stored using JSON. Since version 0.14 MQTT nodes can make properly configured TLS connections.

Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flows using the wide range of nodes in the palette that can be deployed to its runtime in a single-click. Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click. JavaScript functions can be created within the editor using a rich text editor. A built-in library allows you to save useful functions, templates or flows for re-use.

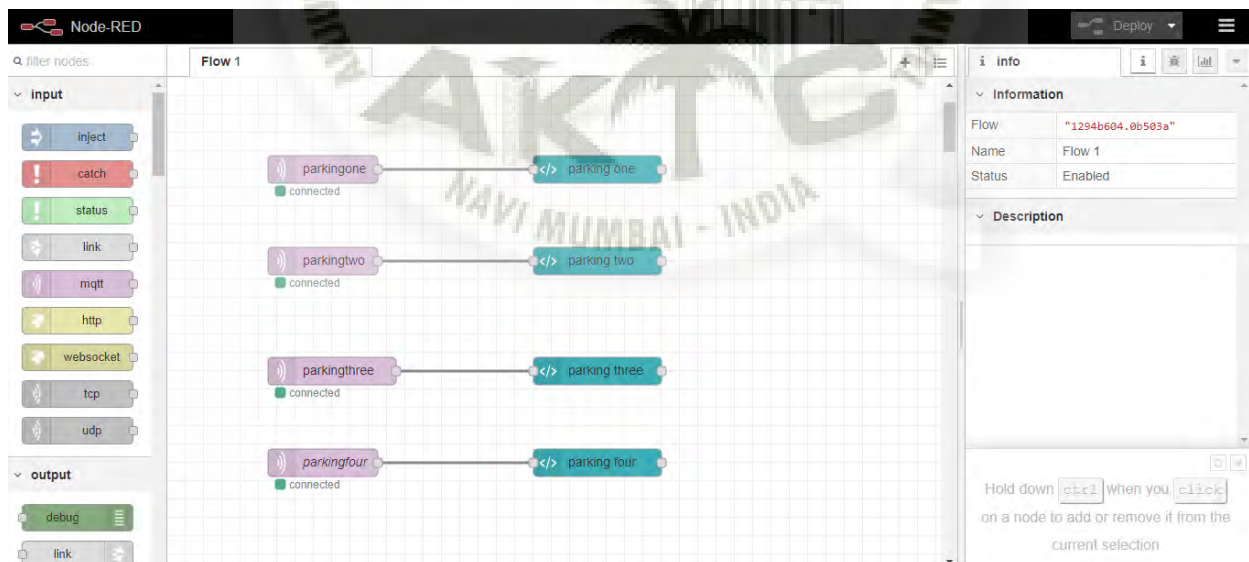
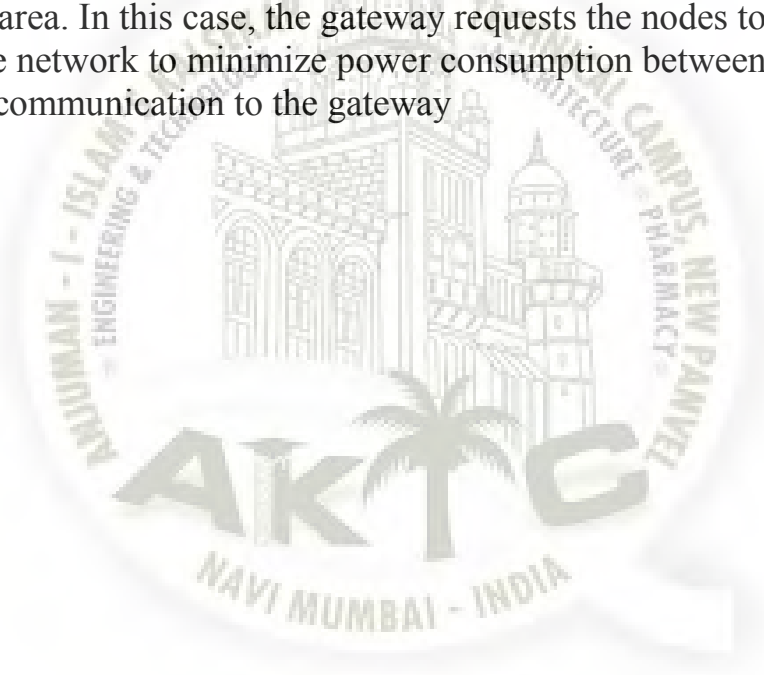


Fig. 5.2 Implementation on Node Red

The parking detection centre uses two very recent technologies, wireless sensor networks (WSN) and RFID technology. The formation of the sensor network changes according to the type of parking in the area. For linear car parks, a chain topology will be formed in the network and, on the other side, a network topology in cluster form will be created in mass car parks. The formation of different topologies is based on the execution of a hybrid self-organization algorithm that is adaptable to the type and structure of parking that allows to form either a chain or clusters, according to the distribution of the nodes and how they are scattered in the parking.

To form a network topology by the sensor nodes in an outdoor parking, all the nodes send their coordinates to the gateway which executes Algorithm 1 making it possible to calculate and detect the type of topology to be formed. In the case of linear parking, the gateway detects that all the nodes have the same X coordinate or the same Y coordinate as a function of the distribution of the nodes in the parking area. In this case, the gateway requests the nodes to create a chain topology in the network to minimize power consumption between the nodes using the multi-hop communication to the gateway



5.3 PROCESS

In this project we used IR module which is connected to NodeMCU. This IR module helps us to detect that the parking slot is vacant/filled.

The data is then sent to our web server. The web server has real time information of the parking slots. A person can use the IP from home or can scan the QR Code near parking area for information which redirects you to our website.

Users primarily interact with 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.



Fig. 5.3 Process of AWS

5.4 QR CODE

QR code (abbreviated from Quick Response Code) is the trademark for a type of matrix barcode (or two-dimensional barcode) first designed in 1994 for the automotive industry in Japan. A barcode is a machine-readable optical label that contains information about the item to which it is attached. In practice, QR codes often contain data for a locator, identifier, or tracker that points to a website or application. A QR code uses four standardized encoding modes (numeric, alphanumeric, byte/binary, and kanji) to store data efficiently; extensions may also be used.

The Quick Response system became popular outside the automotive industry due to its fast readability and greater storage capacity compared to standard UPC barcodes. Applications include product tracking, item identification, time tracking, document management, and general marketing.

A QR code consists of black squares arranged in a square grid on a white background, which can be read by an imaging device such as a camera, and processed using Reed–Solomon error correction until the image can be appropriately interpreted. The required data is then extracted from patterns that are present in both horizontal and vertical components of the image.^[2]



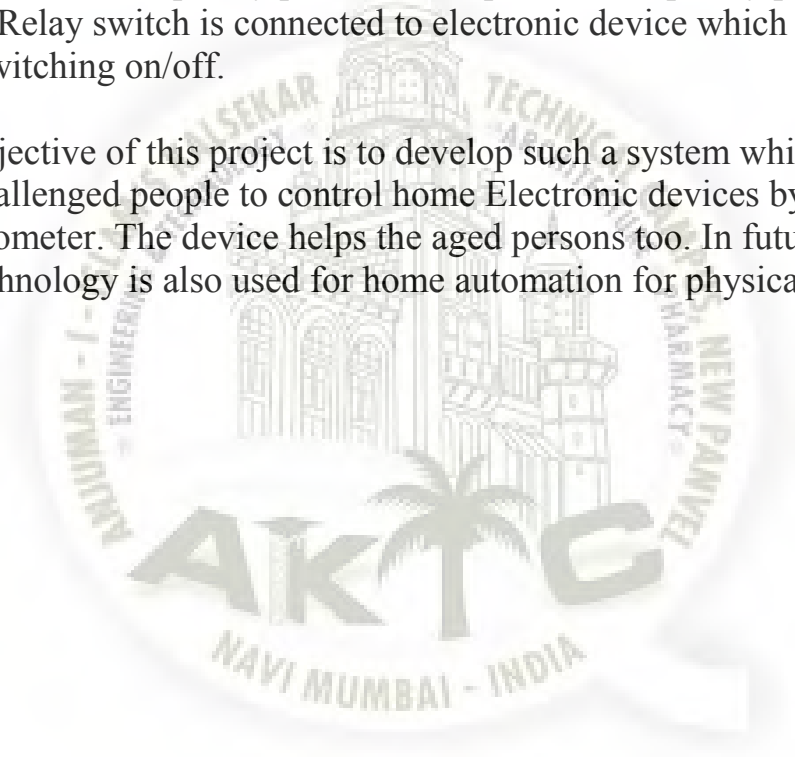
Fig. 5.4 QR code

6. PROPOSED SYSTEM

In this proposed methodology, the two flex sensors are used which can control three DC motors i.e. If 'n' flex sensors are used, $2n-1$ DC motors can be controlled. The flat position of the Flex sensor is considered as '0' and the flex sensor when bend to 90o angle is considered as '1'.

The system architecture gives overall flow of the project and how system components are connected to each other and perform there role of work in this project. Raspberry pi is main technology used in this project. A 5v power supply is provided and passed through regulator so that it can be converted to 3.3v and provided to raspberry pi. The voice command is given as input to android device which is connected to raspberry pi and the output from raspberry pi is given to relay switch. Relay switch is connected to electronic device which does the main function of switching on/off.

The objective of this project is to develop such a system which will help physically challenged people to control home Electronic devices by hand gestures using accelerometer. The device helps the aged persons too. In future Wireless Bluetooth technology is also used for home automation for physically impaired.



6.1 FLOWCHART



Fig. 6.1 Implementation

The Control Unit comprises of the microcontroller- NodeMCU and the 4/8 Channel Relay board. Relay board uses ULN 2803 IC to control the relays. The Blynk app on an Android device communicates with the microcontroller and sends the desired signal via the internet. Figure 1 below shows the basic system design architecture.

Normally a relay is used in a circuit as a type of switch, an automatic switch. There are different types of relays and they operate at different voltages. When a circuit is built the voltage that will trigger it has to be considered. In this system the relay circuit is used to turn the appliances ON/OFF. The high/low signal is supplied from the NodeMCU microcontroller. When a low voltage is given to the relay of an appliance it is turned off and when a high voltage is given it is turned on. The relay circuit to drive four appliances in the Home automation system is shown below in figure 3. The number of appliances can be modified according to the user's requirements.

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 the ESP8266. 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 at bulk, the ESP8266 chip costs only \$2 USD a piece. That makes it an excellent choice for this system design.

7. CODES

7.1 Codes for Home Automation

```

#include <Adafruit_Sensor.h>
#include <Adafruit_AM2320.h>
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#include "DHT.h"

#define Relay1      D6
#define Relay2      D2
#define Relay3      D1
#define Relay4      D5

//DHT11 for reading temperature and humidity value
#define DHTPIN      D7

//buzzer to know the status of MQTT connections and can be used for any
other purpose according to your project need.
#define buzzer      D0

//Selection pins for multiplexer module to switch between different sensors
and give data on a single analog pin.
#define S0          D3
#define S1          D4

//Analog pin to read the incoming analog value from different sensors.
#define analogpin   A0

#define WLAN_SSID   "Trojanhorse"
#define WLAN_PASS   "9324831574"

#define AIO_SERVER   "io.adafruit.com"
#define AIO_SERVERPORT 1883           // use 8883 for SSL
#define AIO_USERNAME "faisalmkhan"
#define AIO_KEY      "e8ea59d4ee9e42b993c334473530993d"

// 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 Sound = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/sound");
Adafruit_MQTT_Publish Motion = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/motion");
Adafruit_MQTT_Publish Light = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/light");

// Setup a feed called 'onoff' for subscribing to changes.
Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/relay1");
Adafruit_MQTT_Subscribe Fan1 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/relay2");
Adafruit_MQTT_Subscribe Light2 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/relay3");
Adafruit_MQTT_Subscribe Fan2 = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/relay4");

#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);
uint32_t delayMS;

// Bug workaround for Arduino 1.6.6, it seems to need a function declaration
// for some reason (only affects ESP8266, likely an arduino-builder bug).
void MQTT_connect();
```

```
void setup() {
  Serial.begin(115200);

  delay(10);

  pinMode(buzzer, OUTPUT);
  pinMode(Relay1, OUTPUT);
  pinMode(Relay2, OUTPUT);
  pinMode(Relay3, OUTPUT);
  pinMode(Relay4, OUTPUT);
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(A0, INPUT);

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

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  Serial.println("WiFi connected");
  Serial.println("IP address: "); Serial.println(WiFi.localIP());

  //Setting up DHT sensor
  dht.begin();

  // Setup MQTT subscription for onoff feed.
  mqtt.subscribe(&Light1);
  mqtt.subscribe(&Fan1);
  mqtt.subscribe(&Light2);
  mqtt.subscribe(&Fan2);
}

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 == &Light1) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light1.lastread);
      int Light1_State = atoi((char *)Light1.lastread);
      digitalWrite(Relay1, Light1_State);
    }
    if (subscription == &Light2) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light2.lastread);
      int Light2_State = atoi((char *)Light2.lastread);
      digitalWrite(Relay2, Light2_State);
    }
    if (subscription == &Fan1) {
      Serial.print(F("Got: "));
      Serial.println((char *)Fan1.lastread);
      int Fan1_State = atoi((char *)Fan1.lastread);
      digitalWrite(Relay3, Fan1_State);
    }
    if (subscription == &Fan2) {
      Serial.print(F("Got: "));
      Serial.println((char *)Fan2.lastread);
      int Fan2_State = atoi((char *)Fan2.lastread);
      digitalWrite(Relay4, Fan2_State);
    }
  }
}
```

```
// 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(S0, HIGH);  
digitalWrite(S1, LOW);  
Serial.print("CO2 "); Serial.println(analogRead(analogpin));  
Serial.print("...");  
Value = analogRead(analogpin);  
if (! CO2.publish(Value)) {  
  Serial.println(F("Failed"));  
} else {  
  Serial.println(F("OK!"));  
}  
  
digitalWrite(S0, LOW);  
digitalWrite(S1, HIGH);  
Serial.print("Sound "); Serial.println(analogRead(analogpin));  
Serial.print(" ...");  
int raw_sound = analogRead(analogpin);  
Value = map(raw_sound,0,1024,0,100);  
if (! Sound.publish(Value)) {  
  Serial.println(F("Failed"));  
} else {  
  Serial.println(F("OK!"));  
}
```

```
digitalWrite(S0, HIGH);
digitalWrite(S1, HIGH);
Serial.print("Light "); Serial.println(analogRead(analogpin));
Serial.print("...");
int raw_light = analogRead(analogpin);
Value = map(raw_light,1024,0,0,100);
if (! Light.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()) {
    mqtt.disconnect();
  }
  */
}
```



```
// 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);
}
```

7.2 codes for Smart Parking

```
const int a=16;
const int b=5;
const int c=4;
const int d=0;
int sensor1= 0;
int sensor2= 0;
int sensor3= 0;
int sensor4= 0;

#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Update these with values suitable for your network.

const char* ssid = "Trojanhorse";
const char* password = "9324831574";
const char* mqtt_server = "13.126.250.135";
String topic;

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;

void setup_wifi() {

  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
}
```

```
randomSeed(micros());

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");

  String strop = topic;
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();

  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(BUILTIN_LED, LOW); // Turn the LED on (Note that
LOW is the voltage level
    // but actually the LED is on; this is because
    // it is active low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, HIGH); // Turn the LED off by making the
voltage HIGH
  }

  if(strop == "bedroom"){

  if ((char)payload[0] == '1') {
    digitalWrite(5, HIGH);
    Serial.print("Message arrived [");

  }

  if ((char)payload[0] == '0') {
    digitalWrite(5, LOW);

  }

  }
}
```

```
if(strop == "livingroom"){

if ((char)payload[0] == '1') {
  digitalWrite(4, HIGH);
}

if ((char)payload[0] == '0') {
  digitalWrite(4, LOW);
}

}

}

}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("outTopic", "hello world");
      // ... and resubscribe
      client.subscribe("bedroom");
      client.subscribe("livingroom");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
}
```

```
void setup() {
  pinMode(a,INPUT);
  pinMode(b,INPUT);
  pinMode(c,INPUT);
  pinMode(d,INPUT);

  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop() {

  sensor1 = digitalRead(a);
  sensor2 = digitalRead(b);
  sensor3 = digitalRead(c);
  sensor4 = digitalRead(d);

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  long now = millis();
  if (now - lastMsg > 2000) {
    lastMsg = now;
    ++value;
    snprintf(msg, 50, "hello world #%ld", value);
    Serial.print("Publish message: ");
    Serial.println(msg);
    client.publish("outTopic", msg);
  }

  if (sensor1 == HIGH)
  {
    Serial.println("Output 1 ON");
    client.publish("parkingone", "1");
  }
  else{

    Serial.println("Output 1 OF");
    client.publish("parkingone", "0");
  }
}
```

```
if (sensor2 == HIGH)
{
    Serial.println("Output 1 ON");
    client.publish("parkingtwo", "1");
}
else{
    Serial.println("Output 1 OF");
    client.publish("parkingtwo", "0");
}

if (sensor3 == HIGH)
{
    Serial.println("Output 1 ON");
    client.publish("parkingthree", "1");
}
else{
    Serial.println("Output 1 OF");
    client.publish("parkingthree", "0");
}

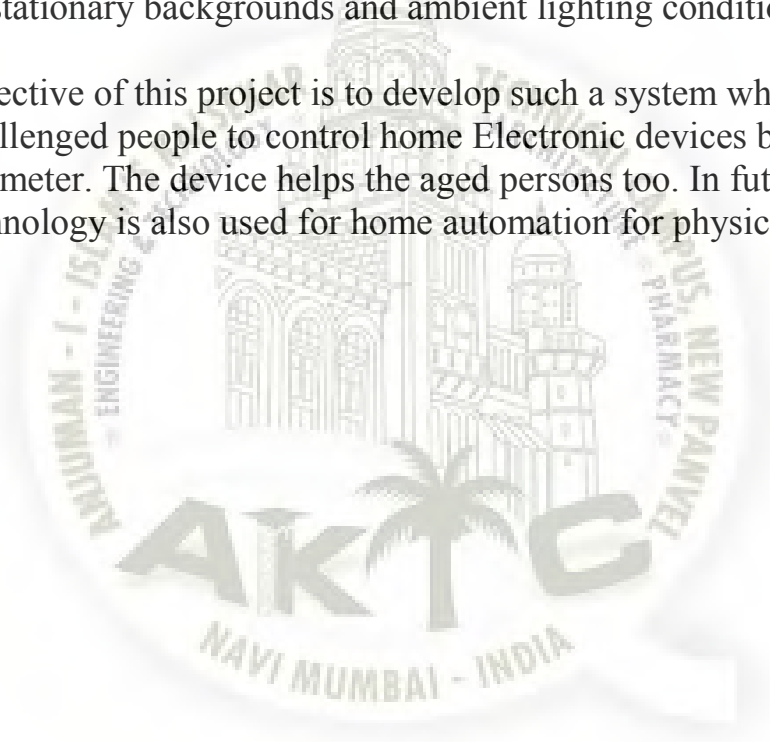
if (sensor4 == HIGH)
{
    Serial.println("Output 1 ON");
    client.publish("parkingfour", "1");
}
else{
    Serial.println("Output 1 OF");
    client.publish("parkingfour", "0");
}
}
```


8. CONCLUSION

We have developed a new system of Smart Parking to optimize parking management of the vehicle in city. We implement parking reservation technique to balance the benefit of both the service providers and requirements from the users. Moreover, we have represented the detailed design, implementation and evaluation of the system how it works. We hope that this app may help to reduce the traffic on the road in cities and also helps the drivers to park the vehicle easily.

A main problem hampering most approaches is that they rely on several underlying assumptions that may be suitable in a controlled lab setting but do not generalize to arbitrary settings. Several common assumptions include: assuming high contrast stationary backgrounds and ambient lighting conditions.

The objective of this project is to develop such a system which will help physically challenged people to control home Electronic devices by hand gestures using accelerometer. The device helps the aged persons too. In future Wireless Bluetooth technology is also used for home automation for physically impaired



9. FUTURE SCOPE

Since the size of the Arduino and XBee setup is very large further optimization can be done on the design of the gesture based gloves. If required the gesture control can also be made to use more number of flex sensor to increase the appliance control respectively. Using more number of sensors, we can also increase the resolution correspondingly.

In future use, we can give voice authentication to provide security. In this only authenticated person voice can access secured device (like locker). By using sensors we reduce the effort of declaring each and every device a particular name. Example: If a person gives a command “lights on” the sensor will sense person location and only that light will get on.

The smart Doorbell can be made by implementing voice and video calls with the person standing right outside the door and the owner remotely. Thereby increasing the safety quotient of the system.

Leading smart cities are recognizing that smart parking infrastructure (i.e., sensors and communications networks) can be leveraged to help enable cars of the future to park themselves. AV fleets are also expected to fundamentally change the way cars are used, affecting how often and where future vehicles will be parked. For more information on the smart parking industry, check out Navigant Research’s recently released report, Smart Parking Systems.

Reference

- [1] Ata-Ur-Rehman, Salman Afghani, Muhammad Akmal, Raheel Yousaf, “Microcontroller and Sensors Based Gesture Vocalizer”, ISPRA '08, 2008.
- [2] Sushmita Mitra, and Tinku Acharya, “Gesture Recognition: A Survey”, IEEE Transactions on Systems, Man and Cybernetics–Part C: Applications and Reviews, 37(3) (2007).
- [3] Anurag Pandey¹, Umesh Mishra², Akash Chaubey³,” Voice Controlled Home Automation” BE CMPN, Department of Computer Engineering, Shree L.R. Tiwari College of Engineering, Mira Road (E), Thane, Maharashtra, India . International Journal of Research in Science & Engineering Special Issue 7- ICEMTE March 2017.
- [4] kyildiz, I.F. Vuran, M.C. Wireless Sensor Networks; Wiley Publication: Hoboken, NJ, USA, 2010

