International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 09 | Sep 2020 www.irjet.net p-ISSN: 2395-0072

Integration of 'Google Assistant' and 'AdafruitIO' via 'IFTTT' to Implement an IoT based Wireless Multi-Sensor Network Surveillance and Multi-Appliance Control System

Ms. Divya Agarwal, Mr. Santosh Kumar Srivastava[2]

[1]M.Tech. Scholar, Department of Computer Science Engineering, B.R.C.M.C.E.T., Bahal, Haryana, India
[2]Assistant Professor, Department of Computer Science Engineering, B.R.C.M.C.E.T., Bahal, Haryana, India

---

Abstract - *Here in this work efforts were made to implement an IoT (Internet-of-Things) based multi-purpose system. One purpose was the controlling of multiple electrical appliances remotely via a VPA (Virtual Personal Assistant) and the other purpose was the remote monitoring of a multi-sensor network via a MQTT broker. Here, the VPA used was 'Google Assistant' by Google and MQTT broker used was 'AdafruitIO' by Adafruit. Another purpose was to integrate and connect these two different services via an 'IFTTT' Applet so that they could be able to communicate with each other. So, here in the IFTTT applet created, 'Google Assistant' was used as a Trigger service whereas 'AdafruitIO' was used as an Action service for this system. This IoT based system was designed developed and implemented as a hardware prototype around a high performance and low cost NodeMCU development board. The multi-sensor network consists of sensors like PIR motion Sensor, LDR light sensor, DHT11 Temperature and Humidity Sensor, MQ135 Gas Sensor. All the sensor's data and status of relays could be observed over the AdafruitIO dashboard on a laptop or mobile phone from anywhere in the world.*

Key Words: MQTT, IFTTT, IoT, VPA, Google Assistant, AdafruitIO, WiFi, NodeMCU, etc.

1. INTRODUCTION

Human's communication with the machines is evolving day by day. This human-machine interaction process started with the buttons has evolved to the touch-pads and now humans can give command to the intelligent machines by just talking to them. One of the goals of Artificial intelligence (AI) is the realization of natural dialogue between humans and machines. In recent years, the voice systems, also known as interactive conversational systems are the fastest growing area in AI. Voice Assistant (VA) is defined as a "voice-based interface that relies on voice commands supported by artificial intelligence to have verbal interaction with the end users for performing a variety of tasks". The VA's are gaining in popularity and getting integrated into our daily lives at a rapid pace. Besides, the progress of technology has made it easier to integrate them into devices, which greatly improves user experience and makes everyday life much easier, e.g. make a voice command and smart devices execute it. Virtual Voice Assistants already allow for many tasks like asking for information, turning off the lights, playing music and the like and are still learning with every interaction the users make. This intuitive way of interacting with technical devices without the need for haptic contact makes verbal communication the new interface to technology. Voice commands on IoT-based Smart Home are simpler to apply, without typing text messages. Users get convenience compared to using text. Dialogue systems or conversational systems can support a wide range of applications. Spoken dialogue systems or voice-controlled assistants are devices that can respond to multiple voices, regardless of accent, can execute several commands or can provide an answer, thus imitating a natural conversation. Speech recognition also called speech to text translation. It transforms human voice into a string of data than can be interpreted by the computer. However, in recent years cloud-based speech recognition systems have been developed a lot. In this way, all elements of a voice-controlled assistant are placed in cloud. "Siri" from Apple, "Google Assistant" from Google, "Cortana" from Microsoft, "Alexa" from Amazon, and "Bixby" from Samsung are the five major VA's commercially available in the market at this point of time. They are present in most smartphones and are based on artificial intelligence elements such as deep learning and neural networks.

2. MOTIVATION

The motivation behind this work came from the consistently rising demand of Artificial Intelligence and IoT devices in our day to day life. These cutting edge technologies have the power to strongly influence the working methodologies of humans as well as machines. The Human Machine Interface (HMI) can do a lot in some specific areas like robotics, smart homes, smart cars, etc. Further, for developers, the easy access to some of the open source platforms like Google Assistant, Alexa, Adafruit IO, IFTTT, etc. has increased the innovation spectrum worldwide. Even hobbyists can now use these platforms for developing their freaking gadgets. Although these platforms are limited in terms of their free usage but provides an option to enhance the knowledge levels about the practicality of these applications.

3. AIM & OBJECTIVE

The aim of the project was to control a scrolling text message LED matrix display using Artificial Intelligence based Virtual Personal Assistants (VPAs) by Google i.e. "Google Assistant". The developed IoT system was NodeMCU centric and communication medium between the user mobile handset and the device to control was wireless only using the Wi-Fi protocol. The objective was to empower the user to assign some voice command over the mobile handset with VPA facility to control a smart device. The VPA should respond to the user by converting this voice command-to-text and revert back to the user by replying in voice. The objective of the study was the implementation of an IoT based VPA (Virtual Personal Assistant) controlled home automation system. The proposed system should be capable enough to save data to the cloud and enable the user to perform remote surveillance of multisensory environmental parameters over the phone or laptop by using internet anywhere from the world. It should also enable the user to control different appliances remotely over the internet through voice commands using the 'Google Assistant' which is a VPA by Google. The objective was to empower the user to assign some voice command over the mobile handset with VPA facility to control a smart device. The VPA should respond to the user by converting this voice command-to-text and revert back to the user by replying in voice and text both and also concurrently by performing the assigned task or transferring the commands to the device.

4. SYSTEM WORKING PRINCIPLE

This project uses two services to make it control through Google Assistant from anywhere in the world, Adafruit MQTT and IFTTT. Adafruit MQTT broker allows changing the message feed/ topic from any internet connected device globally. Here, the NodeMCU was acting as a MQTT client hence it was constantly listening to Adafruit MQTT broker. So if any changes occur in the server side, the same changes will be observed on the client side i.e. on our NodeMCU board. And to change our message on the MQTT broker side via Google assistant, we were using one service called IFTTT. In IFTTT, we were making an applet in which we could connect two services, Google Assistant and Adafruit MQTT. So by making proper applet, we could successfully update the message feed/ topic on the adafruit broker side with Google Assistant on the phone. If This Then That, also known as IFTTT is a freeware web-based service that creates 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. IFTTT pairs two different services/ devices so that they can talk to each other. In other words, IFTTT is the free way to get all the apps and devices talking to each other.

5. SYSTEM ARCHITECTURE

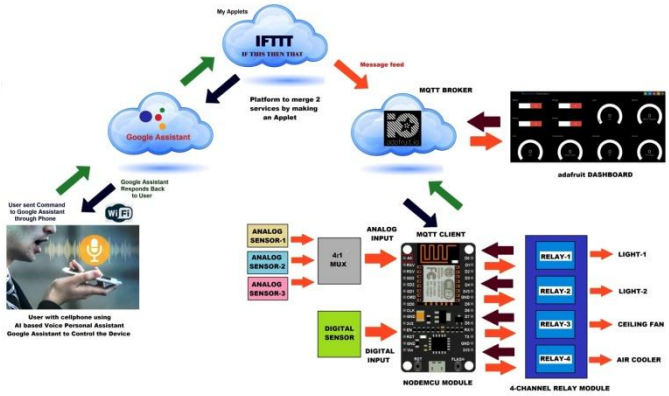

Fig.-1: System Architecture

6. METHODOLOGY ADOPTED

Here in this section the methodologies followed to design and implement this system using different hardware and software tools were discussed.

6.1 System Components

To build this application, these different hardware components were used

- NodeMCU –ESP8266 development board with Wi-Fi SoC.
- Multiple Analog and Digital Sensors including PIR, DHT-11, LDR, MQ-135
- +5V DC power supply

Also, to build this application, three different software platforms were used

- 'Google Assistant' Virtual Personal Assistant
- 'AdafruitIO' MQTT Broker
- 'IFTTT' Applet

To use above services we need to configure these software platforms individually.

6.2 Experimental Set-up

The experimental setup was developed as per the circuit design and as shown below all the sensor modules like LDR light sensor module, IR Proximity sensor module, MQ-135 gas sensor module, PIR Motion sensor module, DHT-11 temperature and humidity sensor module were deployed in the circuit along with a small buzzer and four channel relay module. The output of these relay modules were further extended to switch the ac load. For ac-load four LED bulbs were used here for the demonstration purpose only. Otherwise, any single phase ac electrical appliance could be connected with relay outputs.

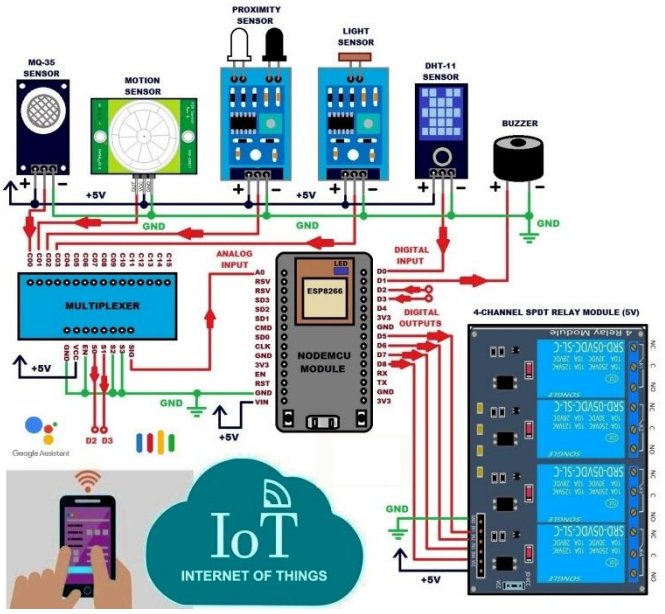

Fig.-2: Hardware Architecture

MQTT is a messaging protocol standardized by OASIS. It is integrated with IoT based systems and is referred as Message Queue Telemetry Transport which is basically a Client/ Server messaging protocol comprising publish/ subscribe approach. The protocol usually runs over TCP/IP, however, any network protocol that provides ordered, lossless, bi-directional connections can support MQTT. It is ideally used for linking of remote devices due to its small code footprint and minimum network bandwidth requirement and hence proved to be a lightweight message transport protocol. A wide range of industries including oil and gas, telecom, automotive, manufacturing, etc. today are using MQTT. Two types of network entities exist in the MQTT protocol. One is a message broker and others are number of clients. An MQTT broker is a server that receives all messages from the clients and then routes the messages to the appropriate destination clients. An MQTT client is any device (a microcontroller or a server) that runs an MQTT library and connects to an MQTT broker over a network.

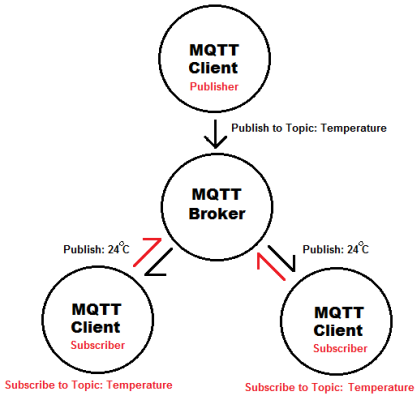

Fig.-3: MQTT Publish/ Subscribe Model

7. FIRMWARE COMPONENTS

The firmware was edited and compiled in the Arduino IDE (Integrated Development Environment). The very first step before starting writing the code is to write an algorithm. So, we referred the algorithm at each and every step of firmware writing to minimize the possibility of error and minimize the time by writing an efficient source code for the system. There are certain key features of the firmware I found worth describing those below here. As an initial and necessary step the appropriate libraries were included at the beginning of firmware. The libraries used here were as shown here.

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#include "DHT.h"
```

Then suitable names were assigned to the NodeMCU analog as well as digital pins with the help of #define function. The names were assigned to the pins just for the ease of user's understanding about each pin's functionality; otherwise it could be difficult to remember each pin function by its pin number. The pin definitions as shown below were made as per the designed schematic diagram and printed circuit board (PCB). First of all download and install all the required libraries in the Arduino IDE to run the code.

```
/******** Pin Definition ************/

//Relays for switching appliances
#define Relay1            D6
#define Relay2            D2
#define Relay3            D1
#define Relay4            D5

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

//buzzer to use it for alert
#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
```

Then the user was required to enter the Wi-Fi hotspot credentials as well as the username and a unique authorization key generated for his/ her account in the AdafruitIO MQTT platform. This key can be accessed by an authorized user only through the AdafruitIO dashboard created. So, this step was considered very important here,

as, to establish the communication between the NodeMCU and the AdafruitIO Dashboard this key must be entered in the firmware as shown below.

```
/************************ WiFi Access Point *********************************/

#define WLAN_SSID       "Divya"
#define WLAN_PASS       "pass@123"

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

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "DIVYA_IOT"
#define AIO_KEY         "aio_DbNi35gBVsydyWFePVRgBPjwEptA"
```

Next important step was to set the serial baud rate and configure the digital pins of the NodeMCU board as input and output and initializing the relays to a default state.

```
void MQTT_connect();
//Servo myservo;
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);
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
digitalWrite(Relay1, HIGH);
digitalWrite(Relay2, HIGH);
digitalWrite(Relay3, HIGH);
digitalWrite(Relay4, HIGH);
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++//
  Serial.println(F("Adafruit MQTT demo"));
```

As per the 16:1 multiplexer's select line status of S0 and S1, each of the connected analog output sensor got NodeMCU's service. The other two status lines of the multiplexer i.e. S2 and S3 were grounded as we were demonstrating the system application with just four analog output sensors. The source code was as shown below.

```
    // Address 00
    digitalWrite(S0, LOW);
    digitalWrite(S1, LOW);
    gas = analogRead(analogpin);
    Serial.print("Gas - "); Serial.println(gas);

    // Address 11
      digitalWrite(S0, HIGH);
      digitalWrite(S1, HIGH);
      int raw_light = analogRead(analogpin);
      light = map(raw_light, 1024, 0, 0, 100);
      Serial.print("Light - "); Serial.println(light);

      Blynk.run();
      timer.run();

    // Address 10
      digitalWrite(S0, HIGH);
      digitalWrite(S1, LOW);
      motion = analogRead(analogpin);
      if (motion > 512)
      {
        motion = 1;

      }
      else
      {
        motion = 0;

      }
// Address 01
  digitalWrite(S0, LOW);
  digitalWrite(S1, HIGH);
  proximity = analogRead(analogpin);
  Serial.print("Proximity - "); Serial.println(proximity);
  if (proximity < 512)
  {
    proximity = 1;
    digitalWrite(buzzer, HIGH);
  }
  else
  {
    proximity = 0;
    digitalWrite(buzzer, LOW);
  }
```

8. EXPERIMENTAL RESULTS

The results obtained from the final prototype board were observed on an adafruitio dashboard as well as over the hardware prototype board. Google Assistant and IFTTT was downloaded and installed into the user's handset and accounts were created there by entering the required user credentials. From the adafruit account, the user was assigned an authorization token ID which was required to be added in the firmware before uploading it to the NodeMCU. A new project was created in the graphical-user-interface dashboard window of the AdafruitIO by selecting some widgets/ blocks from its wide library and configured those widgets/ blocks as per the system requirement. All the sensors and the relay board along with the NodeMCU were carefully connected and powered up by a DC power source. The NodeMCU based hardware and the Adafruit dashboard over the mobile handset/ laptop got connected with the help of Wi-Fi hotspot network. The individual outputs measured from each sensor could be easily monitored on the Adafruit dashboard screen and that too in real-time. Also the relays were triggered remotely by using Google Assistant over the phone. A four channel relay module was interfaced

here with the NodeMCU which operated on 5V dc only. All the four relay outputs were connected to the ac-load and for that purpose LED bulbs were used for the demonstration purpose. Otherwise, any single phase ac load could be connected with the individual relay outputs. One thing to note down here by the user was that the wireless communication network established here for the system was a local network only and thus the system operation was limited within the confines of the local Wi-Fi hotspot used by the user to establish system connectivity. Hence the work was tested and validated by the developed hardware, IFTTT, AdafruitIO Dashboard and Google Assistant over the mobile phone.

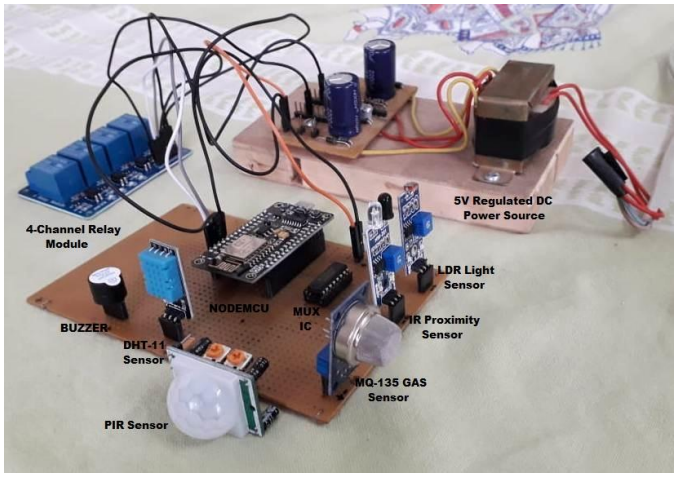

Fig -4: Final Experimental Setup

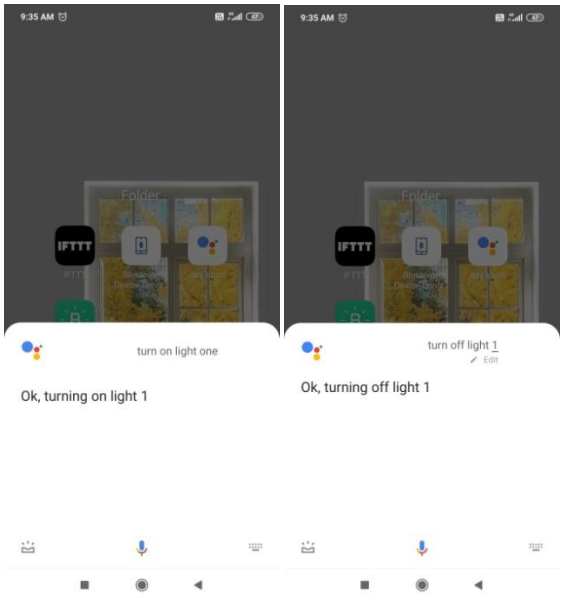

Fig-5: Turning Light-1 'On' & 'Off' using 'Google Assistant'

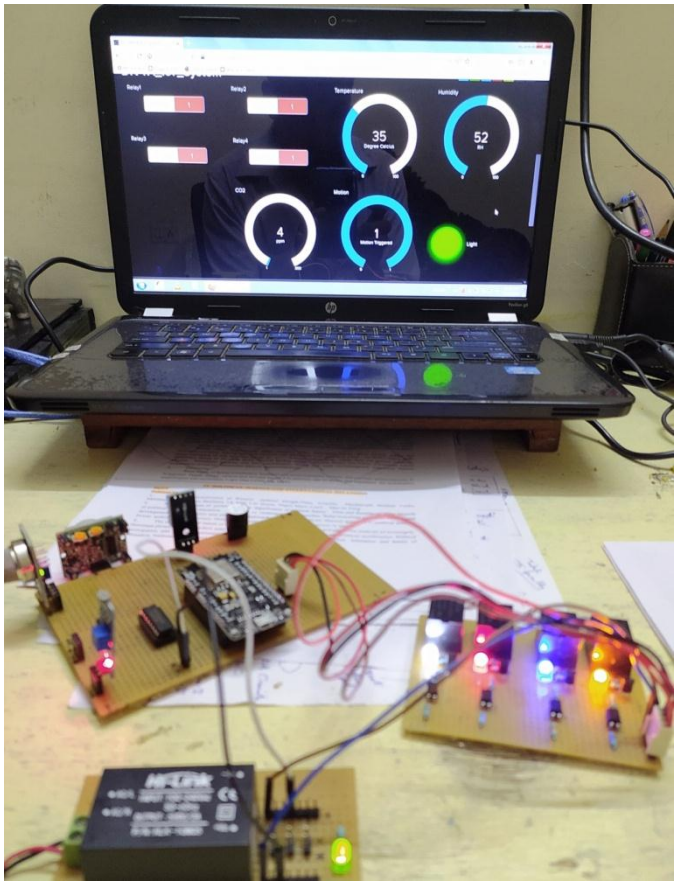

Fig.-6: Experimental Set-up during Testing Phase

Fig -7: System Status-1 on AdafruitIO Dashboard

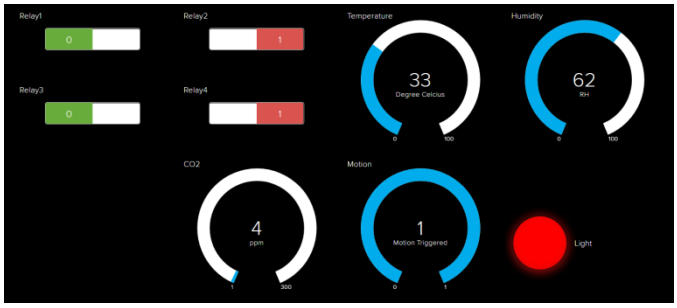

Fig.-8: System Status-2 on AdafruitIO Dashboard

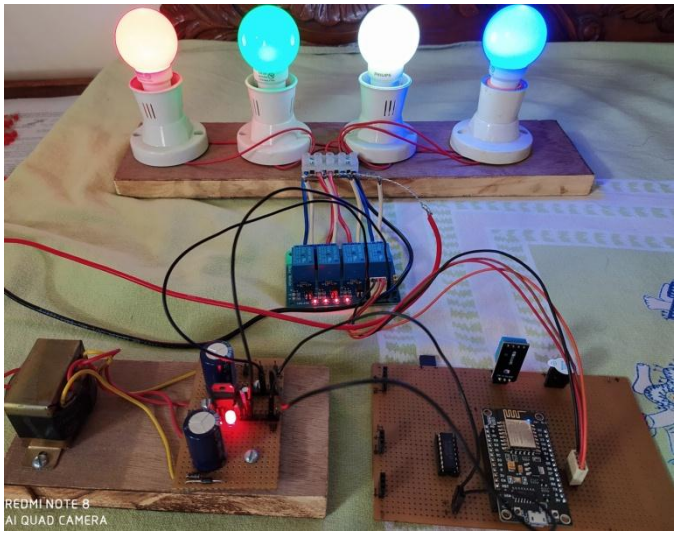

Fig.-9: All Appliances in 'On' State

8. CONCLUSION DRAWN

Finally an application has been implemented which included technologies like Artificial Intelligence based Voice Personal Assistant (VPA)

1. Transferring the data over to the Cloud via Google Assistant
2. Let the two apps connect via an IFTTT's Real-time API. Here, the created Applet connects Google Assistant to the Adafruit MQTT broker. It involved user-oriented triggers from the service that runs near-instantly
3. Proper message feeds were created by the user which declares every action that triggers based on the conditions fed by the user itself in the IFTTT applet
4. When a particular condition got satisfied, the Adafruit MQTT broker received feeds/ topics from the IFTTT client and sent it to the other client i.e. NodeMCU which was subscribed to that topic/ feed.
5. As the data received from the topic/ feed is '1', the Relay connected to the NodeMCU got turned ON and the LED bulb attached to it eventually turned ON.
6. At the same time the status of Relay block in the AdafruitIO dashboard got updated. Means the toggle button block got shifted to turn on position

Adafruit IO MQTT Dashboard

1. The NodeMCU here was a client and the Adafruit IO is a MQTT broker
2. All the sensors were connected to the NodeMCU for monitoring sensors data over the internet from anywhere in the world
3. Data of all those sensors were fed to different feeds/ topics on the same MQTT broker, with an interval of 20 seconds
4. User could see all the sensor's data over the MQTT Dashboard on a laptop or a smartphone

Hence, an IoT based system was implemented that utilizes another AI powered platform for controlling. So, Google Assistant controlled LED scrolling message display was successfully implemented here in this work. The VPA is highly responsive in accepting commands and responding back with appropriate actions. This system provided user the liberty to use either a voice based command or a text command. The developed system was highly responsive and performance wise much more efficient than the conventional systems available. Controlling multiple electrical appliances over a voice command has become reality now and supervision of multi-sensor network from anywhere in the world has become more precise by implementing these techniques. Overall it's a worthy system.

9. FUTURE SCOPE

This work can be further extended to new levels as adoption rate of AI is increasing at a rapid pace and IoT has already captured the market. The combination of two would lead to the development and implementation of much more sophisticated systems. These systems would restrict human interventions as most of the tasks would be effectively and efficiently performed by these smart systems only. Google Assistant based controlling of different parameters and devices could be quite useful in the health sector, agriculture, defense sector, security, etc. One can integrate multiple sensors and devices with the help of IoT platforms and their controlling could be made easy using these AI powered VPAs like Google Assistant over the phone.

REFERENCES

[1] Haris Isyanto; AjibSetyo Arifin; Muhammad Suryanegara, "Design and Implementation of IoT-Based Smart Home Voice Commands for disabled people using Google Assistant", International Conference on Smart Technology and Applications (ICoSTA), 2020, Publisher: IEEE

[2] Tae-Kook Kim, "Short Research on Voice Control System Based on Artificial Intelligence Assistant", International Conference on Electronics, Information, and Communication (ICEIC), 2020, Publisher: IEEE

[3] Vanathi K. Lalitha; B. Mahalakshmi; S. Madhusudan; M. Srinivasaperumal; S. Srikanth; Sathish R. Kumar, "Smart Control Of Home Amenities Using Google Assistant And Clap Switch Circuit", 5th International Conference on

Advanced Computing & Communication Systems (ICACCS), 2019, Publisher: IEEE

[4] David Sheppard; Nick Felker; John Schmalzel, "Development of Voice Commands in Digital Signage for Improved Indoor Navigation Using Google Assistant SDK", IEEE Sensors Applications Symposium (SAS), 2019, Publisher: IEEE

[5] Mokh. SholihulHadi; Maulana Ahmad As Shidiqi; Ilham Ari ElbaithZaeni; Muhammad Alfian Mizar; Mhd Irvan, "Voice-Based Monitoring and Control System of Electronic Appliance Using Dialog Flow API Via Google Assistant", International Conference on Electrical, Electronics and Information Engineering (ICEEIE), 2019, Volume-6, Publisher: IEEE

[6] İlkan Yıldırım; Erkan Bostancı; Mehmet Serdar Güzel, "Forensic Analysis with Anti-Forensic Case Studies on Amazon Alexa and GoogleAssistant Build-In Smart Home Speakers", 4th International Conference on Computer Science and Engineering (UBMK), 2019, Publisher: IEEE

[7] M. Sundarramurthi; A. M. Anjana Sundari; Anandi Giridharan, "A Method of Designing Home Automation Control System (HACS) Using Virtual Assistant And Mobile Application", International Conference on contemporary Computing and Informatics (IC3I), 2019, Publisher: IEEE

[8] Veton Këpuska; Gamal Bohouta, "Next-generation of virtual personal assistants (Microsoft Cortana, Apple Siri, Amazon Alexa and Google Home)", IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), 2018, Publisher: IEEE

[9] Septimiu Mischie; Liliana Mâţiu-Iovan; Gabriel GăŠpăresc, "Implementation of Google Assistant on Raspberry Pi," International Symposium on Electronics and Telecommunications (ISETC), 2018, Publisher: IEEE

[10] Aleksandar Lazić; Milan Z. Bjelica; Dejan Nad; Branislav M. Todorović, "Google Assistant Integration in TV Application for Android OS", 26th Telecommunications Forum (TELFOR), 2018, Publisher: IEEE

[11] UrošVišekruna; Milan Savić, "Integration of Google Assistant in Android Application for Voice Control of Media Playback", 26th Telecommunications Forum (TELFOR), 2018, Publisher: IEEE

[12] Arsénio Reis; Dennis Paulino; Hugo Paredes; Isabel Barroso; Maria João Monteiro; Vitor Rodrigues; João Barroso, "Using intelligent personal assistants to assist the elderlies an evaluation of Amazon Alexa, Google Assistant, Microsoft Cortana, and Apple Siri", 2nd International Conference on Technology and Innovation in Sports, Health and Wellbeing (TISHW), 2018, Publisher: IEEE

[13] Sandipan Chakraborty; Shouvik Mukherjee; Suvrojit Kumar Saha; Himadri Nath Saha, "Autonomous Vehicle for Industrial Supervision Based on Google Assistant Services & IoT Analytics", IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2019, Publisher: IEEE

[14] Laura Burbach; Patrick Halbach; Nils Plettenberg; Johannes Nakayama; Martina Ziefle; André Calero Valdez, ""Hey, Siri", "Ok, Google", "Alexa". Acceptance-Relevant Factors of Virtual Voice-Assistants", IEEE International Professional Communication Conference (ProComm), 2019, Publisher: IEEE