1. PROJECT DEFINITION:

Our aim of project is to control home appliances using google assistant. It is an Internet of things based project. This project provides people a way for creating a smart home. By this one shall be able to control his/her internet connected appliances remotely from a smart phone. IoT devices are a part of the larger concept of home automation.

2. DESCRIPTION:

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 theirs 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.

Moreover, it would be better if everything such as warming bath water and adjusting the room temperature were already done before they reach their home just by giving a voice command. So, when people would arrive home, they would find the room temperature, the bath water adjusted to their suitable preferences, and they could relax right away and feel cozier and rather, feel more homely.

Human assistants like housekeepers were a way for millionaires to keep up their homes in the past. Even now when technology is handy enough only the well to do people of the society are blessed with these new smart home devices, as these devices costs are a bit high. However, not everyone is wealthy enough to be able to afford a human assistant, or some smart home kit. Hence, the need for finding an inexpensive and smart assistant for normal families keeps growing.

This paper proposes such inexpensive system. It uses the Google Assistant, the IFTTT application, the Adafruit web application and the NodeMCU microcontroller as the major components along with a relay board comprising of 4 relays. Natural language voice is used to give commands to the Google Assistant. All of the components are connected over the internet which puts this system under the IoT.

Therefore by using these above applications and components we have built a system that would be able to turn ON/OFF the lights from you smart phones.

3. ABOUT INTERNET OF THINGS

3.1 The Vision

The Internet of Things represents a vision in which the Internet extends into the real world embracing everyday objects. Physical items are no longer disconnected from the virtual world, but can be controlled remotely and can act as physical access points to Internet services. An Internet of Things makes computing truly ubiquitous a concept initially put forward by Mark Weiser in the early 1990s.This development is opening up huge opportunities for both the economy and individuals. However, it also involves risks and undoubtedly represents an immense technical and social challenge. The Internet of Things vision is grounded in the belief that the steady advances in microelectronics, communications and information technology we have witnessed in recent years will continue into the foreseeable future. In fact – due to their diminishing size, constantly falling price and declining energy consumption – processors, communications modules and other electronic components are being increasingly integrated into everyday objects today.

“Smart” objects play a key role in the Internet of Things vision, since embedded communication and information technology would have the potential to revolutionize the utility of these objects. Using sensors, they are able to perceive their context, and via built-in networking capabilities they would be able to communicate with each other, access Internet services and interact with people. “Digitally upgrading” conventional object in this way enhances their physical function by adding the capabilities of digital objects, thus generating substantial added value. Forerunners of this development are already apparent today – more and more devices such as sewing machines, exercise bikes, electric toothbrushes, washing machines, electricity meters and photocopiers are being “computerized” and equipped with network interfaces.

3.2 Facts

- By 2021, IoT is expected to be an industry worth $1.4 trillion, as companies invest in software, hardware and other services. According to General Electric, over $60 billion is expected to be invested in Industrial IoT products and devices by 2030.
- In 2020, it is expected that there will be 24 billion IoT devices, according to BI Intelligence. Most will be used by businesses and governments, but a large portion (five billion) will be used by consumers.
- Also in 2020, around a quarter of a billion vehicles are expected to be connected to the internet. Over 10 million units of clothing connected to the internet are expected to be shipped in that year as well.

4. SOFTWARE AND HARDWARE REQUIREMENTS

4.1 Software requirements

I. Google Assistant

II. Adafruit IO

III. IFTTT

IV. Arduino IDE

V. Microsoft Visio

5.2 Hardware requirements

I. Node MCU

II. Relay Board

III. Breadboard

IV. Jumper Wires

V. Battery

VI. LEDs

VII. Smart phone

5. SYSTEM DESIGN AND IMPLEMENTATION

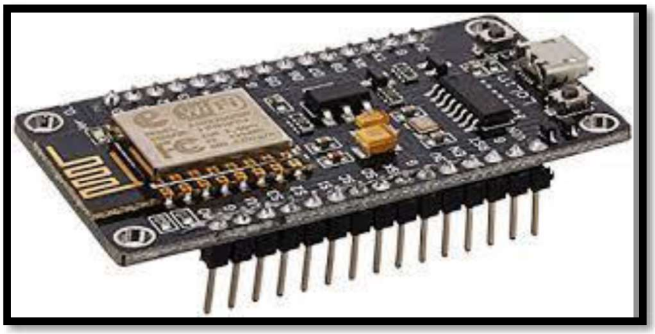
The Control Unit comprises of the microcontroller- NodeMCU and the 4 Channel Relay board. Relay board uses ULN 2803 IC to control the relays. The Adafruit web application communicates with the microcontroller and sends the desired signal via the internet. NodeMCU's digital output pins are connected to the Relay pins of the Relay board. Finally, each Relay is connected to an appliance.

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.

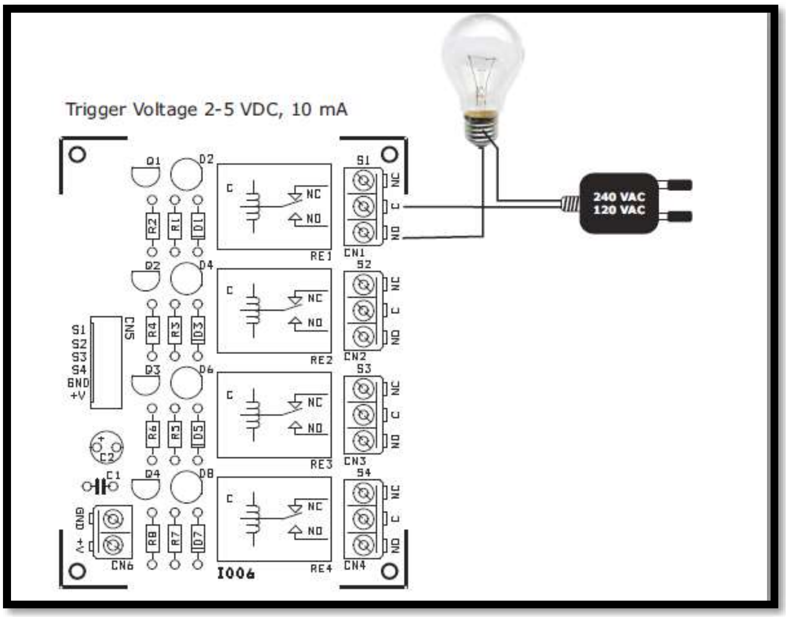
The NodeMCU aims to simplify ESP8266 development. It has two key components.

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

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



A relay is an electromagnetic switch. It is activated when a small current of some microampere is applied to it. 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.



Adafruit IO is a system that makes data useful. Our focus is on ease of use, and allowing simple data connections with little programming required.IO includes client libraries that wrap our REST and MQTT APIs. IO is built on Ruby on Rails, and Node.js

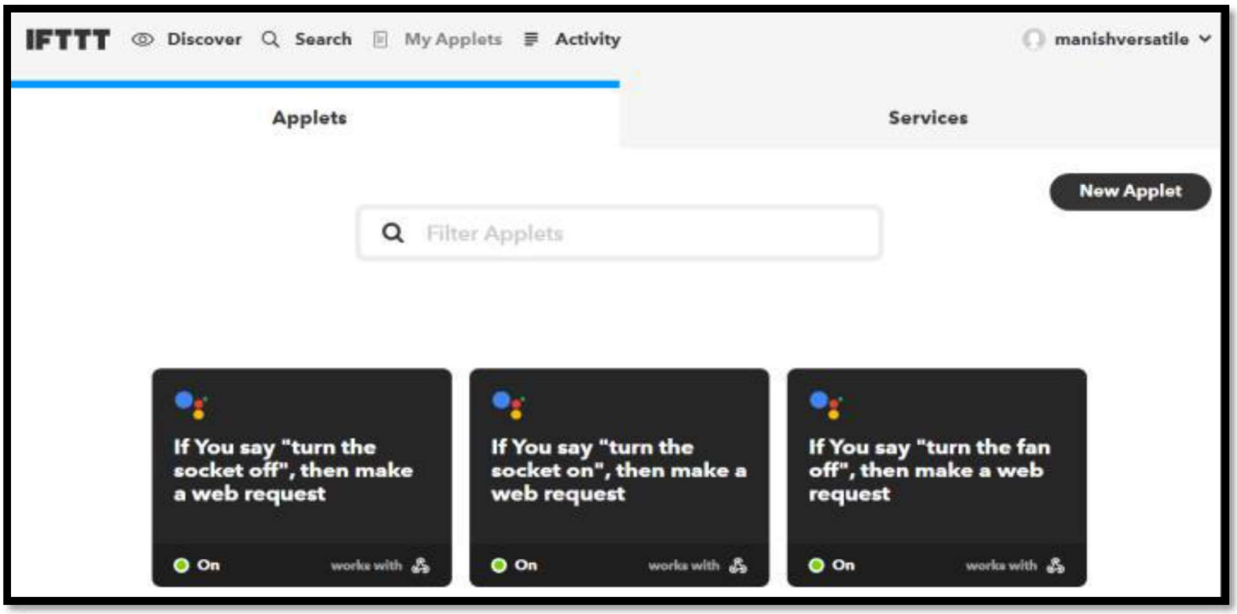
IFTTT [1] derives its name from the programming conditional statement “if this, then that.” IFTTT is both a website and a mobile app that launched in 2010 and has the slogan "Put the Internet to work for you". The idea is that you use IFTTT to automate everything from your favourite apps and websites to app-enabled accessories and smart devices. What the company provides is a software platform that connects apps, devices and services from different developers in order to trigger one or more automations involving those apps, devices and services. Here, IFTTT application is used to bridge the gap between the Google Assistant commands and the Adafruit web application.

Setting up the IFTTT application first requires logging in after which we need to create an applet and then “This”, i.e. the trigger, here we select Google Assistant and then we will type in the commands to which the Google Assistant should respond and to this command it should control the appliance/relay associated with it. The response command from the Goggle Assistant can also be typed in as desired.

After configuring the trigger, i.e. “This” of the application we need to configure the “That”. What should be done once the Google Assistant hears the command which we just configured? This is decided by setting “That” of the Adafriut. We click “That” and then select webhooks and click connect. Webhooks will allow us to send commands to the Adafriut. Now, in the URL we type the IP address of the Adafriut followed by the Authentation token sent by the Adafriut and then the pin number of the microcontroller to which the device to be controlled is connected.

Then in the method we select ‘PUT’ and the content type is ‘Application/JSON’ and in the body we write [“1”] to turn ON and [“0”] to turn OFF. This creates the action for the trigger i.e. the Google Assistant command. The action taken by it is simply sending a message to the Adafriut to either turn ON or OFF the concerned connected device.

Finally, the microcontroller is programmed with the actions it needs to do once it receives the signal from the Adafriut . Before that, the Adafriut and the microcontroller should communicate and the communication is done via the internet and since the microcontroller, NodeMCU comes with inbuilt Wi-Fi module, it is programmed to connect to the desired network once plugged in. ‘C’ language is used to program the microcontroller and is programmed in the Arduino IDE.

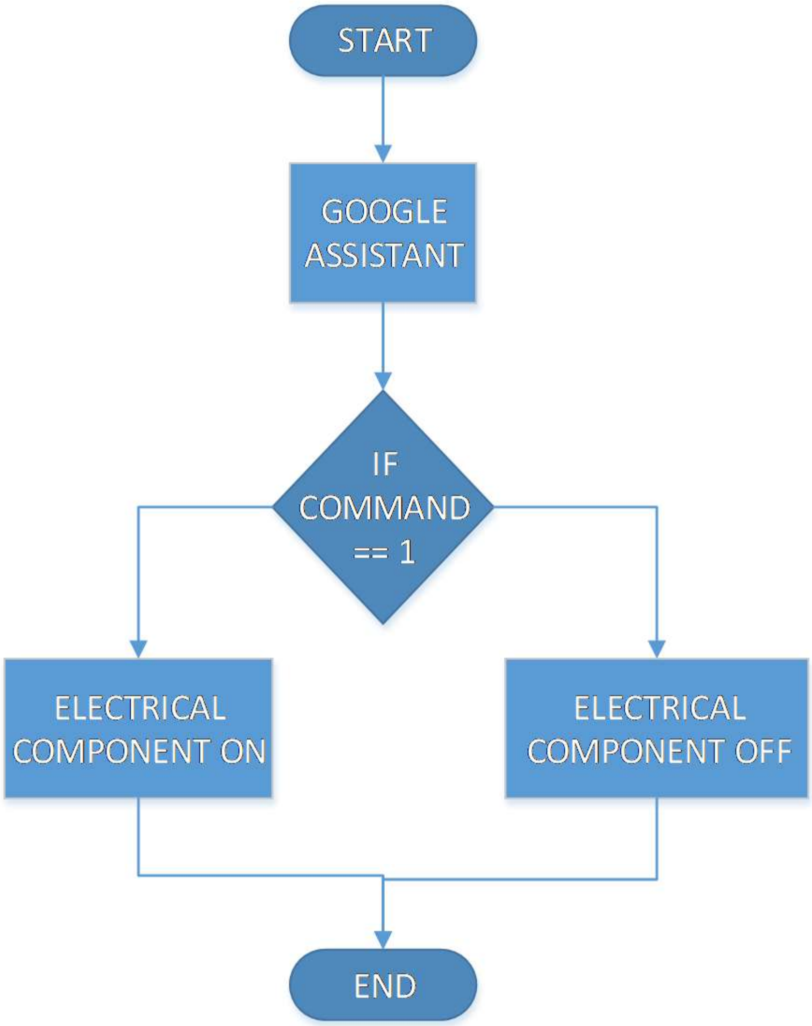


If You say "turn the socket off", then make a web request

If You say "turn the socket on", then make a web request

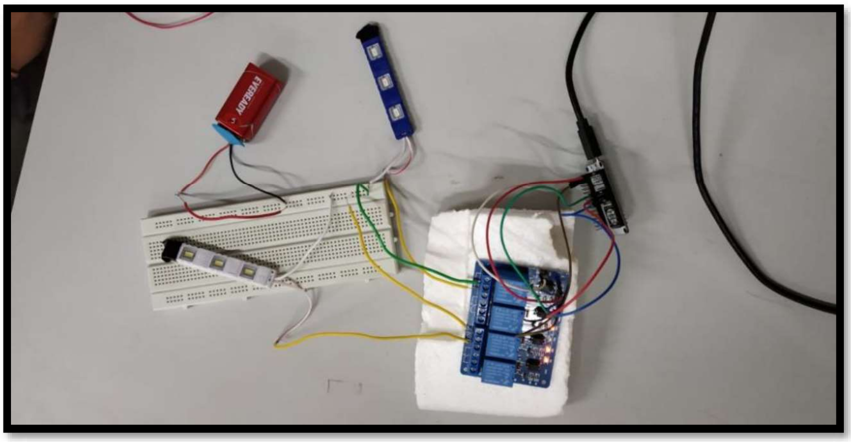
If You say "turn the fan off", then make a web request

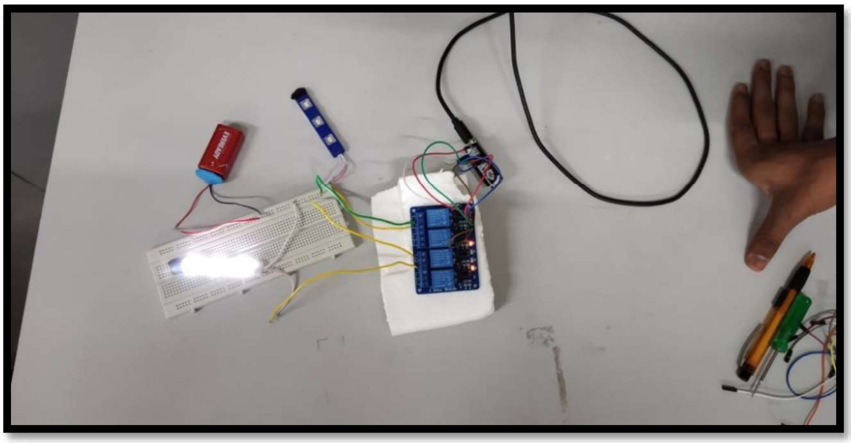
5.1 System Workflow

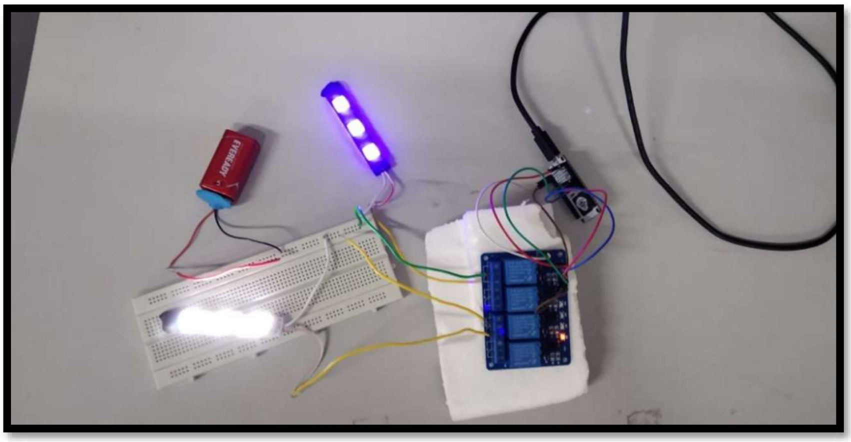


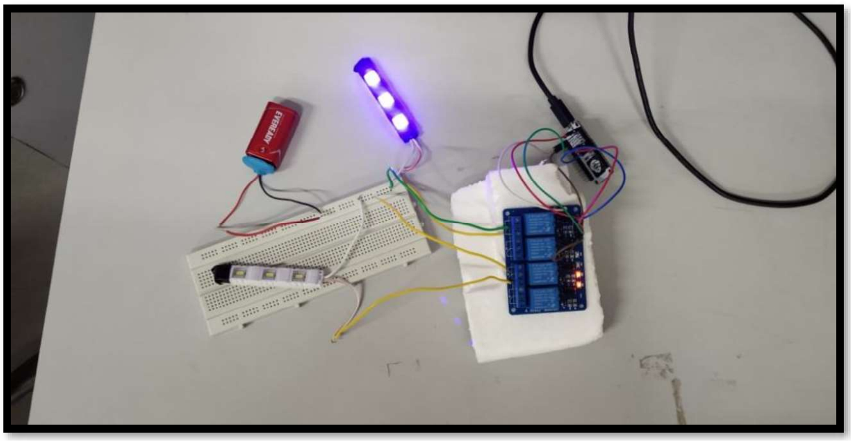
5.2 System Design

5.2.1 Screen Layout









5.2.2 Method Pseudo Code

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

#define Relay1          D7
#define Relay2          D6
#define Relay3          D5
//#define Relay4          D1

#define WLAN_SSID       "CaptainCool_7"             // Your SSID
#define WLAN_PASS       "54919703"        // Your password

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

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "ishanladva"            // Replace it with your username
#define AIO_KEY         "35c3dd026d5648e79ee835a561e2a0c7"   // Replace with your Project Auth Key

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

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME,
AIO_KEY);

/****************************** Feeds ***************************************/
// Setup a feed called 'onoff' for subscribing to changes.
Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME"/feeds/relay1");
// FeedName
Adafruit_MQTT_Subscribe Light2 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/relay2");
Adafruit_MQTT_Subscribe Light3 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/relay3");
//Adafruit_MQTT_Subscribe Light4 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME
"/feeds/Relay4");
```

```
void MQTT_connect();

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

  pinMode(Relay1, OUTPUT);
  pinMode(Relay2, OUTPUT);
  pinMode(Relay3, OUTPUT);


  // 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());

// Setup MQTT subscription for onoff feed.
  mqtt.subscribe(&Light1);
  mqtt.subscribe(&Light2);
  mqtt.subscribe(&Light3);

}
void loop() {

  MQTT_connect();


  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 == &Light3) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light3.lastread);
      int Light3_State = atoi((char *)Light3.lastread);
      digitalWrite(Relay3, Light3_State);
    }
  }
}

void MQTT_connect() {
  int8_t ret;

  // Stop if already connected.
  if (mqtt.connected()) {
    return;
  }

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

  uint8_t retries = 3;

  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Retrying MQTT connection in 5 seconds...");
    mqtt.disconnect();
    delay(5000);  // wait 5 seconds
    retries--;
    if (retries == 0) {
      // basically die and wait for WDT to reset me
      while (1);
    }
  }
  Serial.println("MQTT Connected!");

}
```

6. LIMITATIONS OF PROJECT

- It requires intense strong internet connection
- Is phrase specific (of Applets)
- Works only on two states 1(ON) and 0(OFF)

7. FUTURE ENHANCEMENT

- Remove the limitation of phrase specifies.
- Implement on more devices.
- Try to make more than only ON & OFF states.

8. BIBLIOGRAPHY

- **https://learn.adafruit.com/welcome-to-adafruit-io/what-is-adafruit-io**
- **https://ifttt.com/discover**
- **https://www.youtube.com/watch?v=369rBvlAnk0&t=561s**
- **https://goo.gl/BvNcST**
- **https://support.google.com/googlehome/answer/7194656?co=GENIE.Platform%3DDesktop&hl=en**