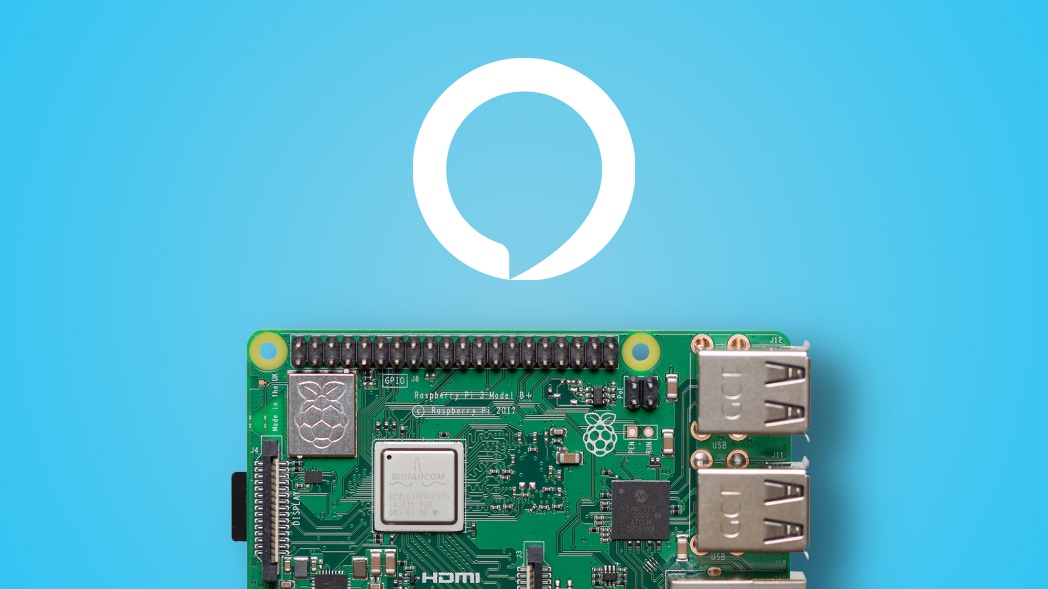
**SMART HOME ASSISTANT WITH AMAZON ALEXA**

**(AlexaPi)**



**TEAM NAME:** MINDBENDERS

**TEAM MEMBERS:**

1. CH. SANDEEP KUMAR
2. CH. SAI KRISHNA
3. D.AKSHITHA

**ABSTRACT**

Amazon Alexa, known simply as Alexa, is a virtual assistant developed by Amazon, first used in the Amazon Echo and the Amazon Echo Dot smart speakers developed by Amazon Lab126. It is capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as news. Alexa can also control several smart devices using itself as a home automation system. Users are able to extend the Alexa capabilities by installing skills. We can get most the device by downloading the alexa on a Raspberrypi board. By providing the credentials of the AVS product created in the developer.amazon.com we can give the raspberry pi board the access to alexa through our account. Here we have installed amazon Alexa on the raspberrypi 3+. And with the help of a microphone we can give the voice inputs/commands to the Alexa. The input to the board are directed to the applet created in the IFTT through the publish and subscribe keys of the pubnub. The configurations on the raspberry are altered. The audio output options are changed to 3.5mm audio jack. By connecting the daily used electronics like AC, lights, TV to the raspberrypi we can control them handsfree by just giving voice commands to alexa. Ex. “alexa turn ON lights”. The main objective of this project is to minimize the workload on the humans and make the everyday life hassle free.

**Components Required:**

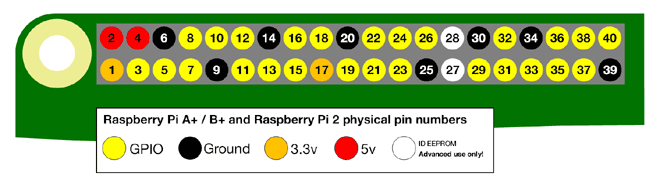
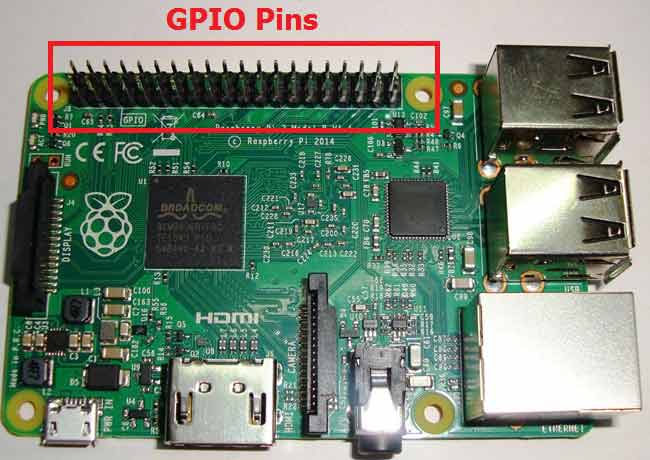
1. RaspberryPi 3+
2. External Speaker with 3.5mm AUX cable
3. USB Microphone
4. LED light / Light bulb

**Software/Websites Used:**

1. IFTT Applet
2. Publish and subscribe key from Pubnub
3. Pubnub library in Raspberry
4. Amazon AVS

**Raspberrypi 3**

The RaspberryPi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries. The original model became far more popular than anticipated, selling outside its target market for uses such as robotics. It does not include peripherals (such as keyboards and mice) and cases. However, some accessories have been included in several official and unofficial bundles.



As shown in above figure, there are 40output pins for the PI. But when you look at the second figure, you can see not all 40 pin out can be programmed to our use. These are only 26 GPIO pins which can be programmed. These pins go from **GPIO2 to GPIO27**.These **26 GPIO pins can be programmed** as per need. Some of these pins also perform some special functions, we will discuss about that later. With special GPIO put aside, we have 17 GPIO remaining.Each of these 17 GPIO pins can deliver a maximum of **15mA current**. And the sum of currents from all GPIO cannot exceed 50mA. So we can draw a maximum of 3mA in average from each of these GPIO pins. So one should not tamper with these things unless you know what you are doing.

The Raspberry Pi Foundation provides Raspbian, a Debian-based Linux distribution for download, as well as third-party Ubuntu, Windows 10 IoT Core, RISC OS, and specialised media centre distributions. It promotes Python and Scratch as the main programming languages, with support for many other languages The default firmware is closed source, while an unofficial open source is available. Many other operating systems can also run on the Raspberry Pi, including the formally verified microkernel, seL4. Other third-party operating systems available via the official website include Ubuntu MATE, Windows 10 IoT Core, RISC OS and specialised distributions for the Kodi media centre and classroom management.

**Speaker with 3.5mm AUX cable**

Raspberry pi gives us the option of connecting an external speaker through 3.5mm AUX cable. This speaker acts as an interface between the Alexa and the user. To make the 3.5mm jack of the raspberry as active we need to configure the Audio options to Force 3.5mm jack. Then the speaker will be ready to use.

**USB Microphone**

Raspberry pi has 4 USB ports. And the only way to connect the microphone to the board is through a USB cable. So we require a USB microphone. We can also use webcams which have built-in microphone. This microphone acts as the interface between the user and the Alexa.

**LED light/AC bulb**

We require a led light or a AC(alternating current) bulb to check the functionality of the commands. The GPIO pin of the Raspberry pi is connected to the light bulb.

**Block Diagram**

**USB Microphone**

**Human Voice**

**SPEAKER**

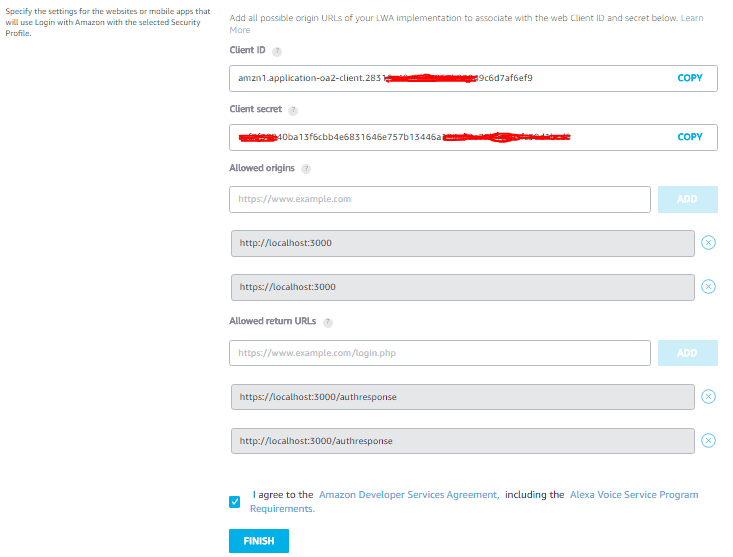
**LIGHT IS ON/OFF**

**IFTT**

**Raspberry Pi**

**Creating AVS product**

* Before beginning the setup on the Raspberry we have to create an Amazon account and has to login to developer.amazon.com. we have to create a product with the help of the Alexa voice service (AVS).
* We must create a new device and have to specify the following details: Device ID, Device Type, Security profile description, security profile ID, Amazon ID, Client ID, and Client Secret.
* After creating the device we have to save the above credentials in a notepad or a word document these credentials are key while accessing the Alexa on the raspberry pi.



* The allowed origins and allowed URLs should be mentioned as http://localhost:5050, https://localhost:5050,http://your.raspberry.pi.ip:5050 and https://your.raspberry.pi.ip:5050

**Setting up Raspberry pi**

The first step will be to make sure your Raspberry Pi is up and running and open up the terminal.

First of all we need to make sure we are in the /opt directory by typing in

cd /opt

Since AlexaPi is hosted on GitHub, we’ll also now need to make sure that git is installed

sudo apt-get install git

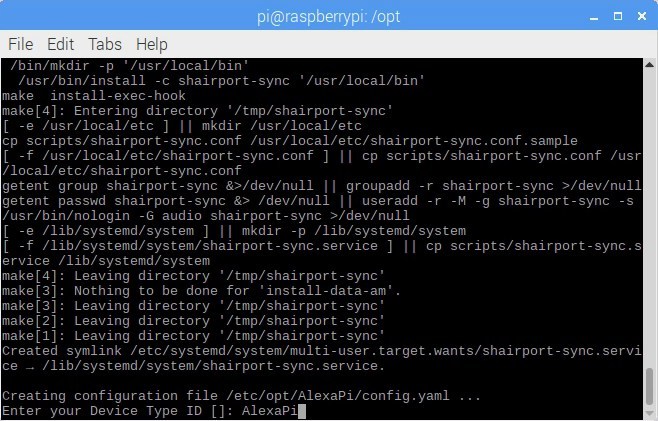
Now we can clone the AlexaPi repository to the storage on the Raspberry Pi

sudo git clone https://github.com/alexa-pi/AlexaPi.git

From here, you’ve got to follow the prompts that show up in the terminal, many of them will be filled in but you will be asked whether you want AirPlay functionality, Alexa to automatically run on startup and also to enter your Amazon product credentials that we created earlier.

When asked to enter your Device Type ID, enter the Product ID that you created.

Once you’ve done that you’ll need to copy and paste your Client ID, Client secret and Security Profile ID into the terminal.



Once you’ve entered those, you can browse to the URL displayed in the terminal of the Raspberry Pi and log into your Amazon account.Now you can press Ctrl+C to close the Window and AlexaPi is set up!

**Configuring Raspbian**

The final stage is to make sure Raspbian has the correct configuration so that AlexaPi will work without any issues.

First of all, let’s open up the Raspberry Pi configuration screen

sudo raspi-config

The next step will be changing the audio output.

Go down to the “Advanced Options” tab and select “Audio”. Here you can override the automatic settings and force the Raspberry Pi to use the 3.5mm output by selecting “Force 3.5mm (‘headphone’) jack”.

Now that you’ve done that, you can go across to “Finish” at the bottom and reboot!

When your Raspberry Pi boots back up, you should hear Alexa say “Hello”. Now you have Alexa running on your Raspberry Pi.

By following the above steps, we have successfully installed Amazon Alexa on our raspberry pi.

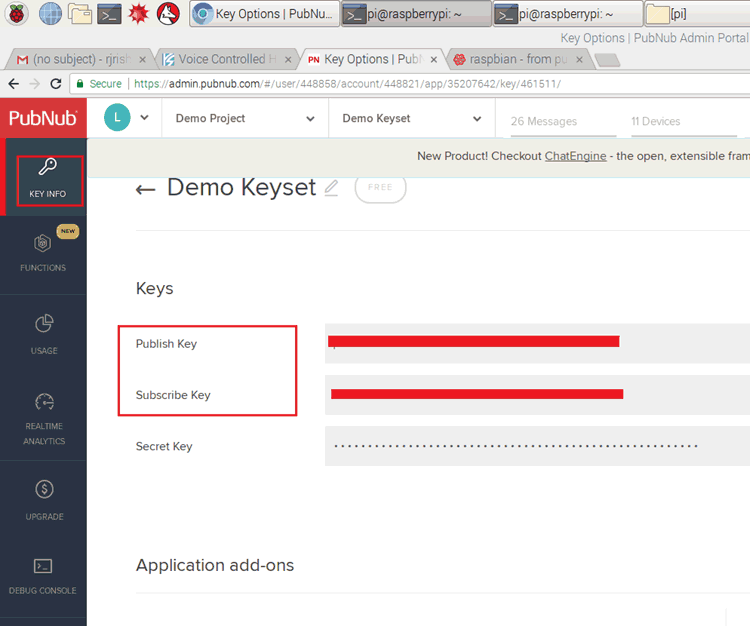
Now we have to code the raspberry pi in such a way that the voice commands from the user are turned into actions. Then we can say that we have created a handsfree assistant. For that we have to create accounts in IFTT and Pubnub

More over we have to install Pubnub in the raspberry pi.

**Creating pubnub keys**

**PubNub** is a global Data Stream Network (DSN) and real-time infrastructure-as-a-service (IaaS) company based in San Francisco, California. The company makes products for software and hardware developers to build Realtime web, mobile, and Internet of Things (IoT) applications.

By logging in to pubnub account we will get publish, subscribe key. Save those keys for the future uses.



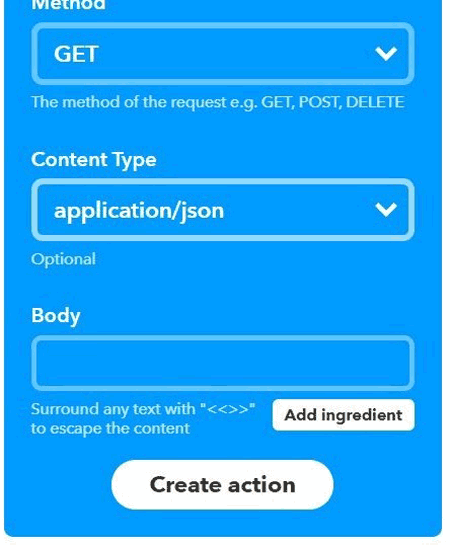
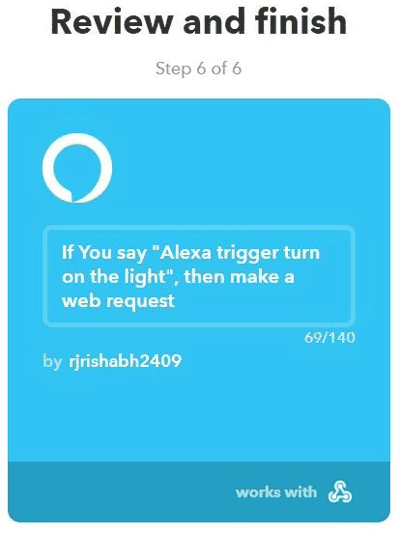
**Creating IFTTT applets**

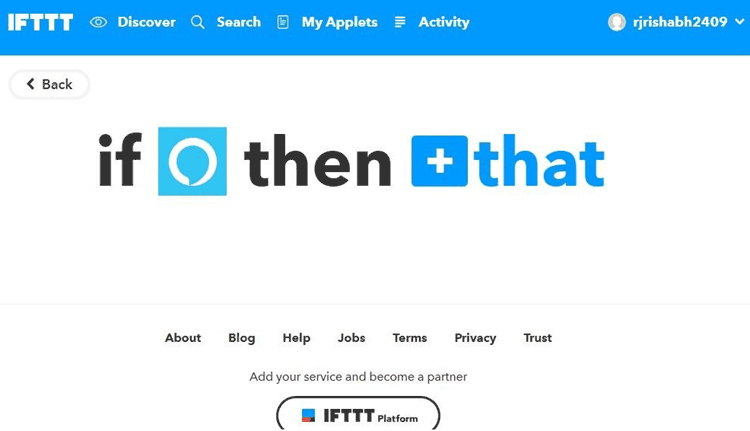
**If This Then That**, also known as **IFTTT** is a free web-based service to create chains of simple conditional statements, called *applets*.

An applet is triggered by changes that occur within other web services such as Gmail, Facebook, Telegram, Instagram, or Pinterest.

For example, an applet may send an e-mail message if the user tweets using a hashtag, or copy a photo on Facebook to a user's archive if someone tags a user in a photo.

We should login to IFTTT with our credentials and create new applet by selecting the web service as alexa and the we have to provide phrases like turn on light (or) turn off light. At the end we will get a URL.



**Program**

#Import all the libraries

import RPi.GPIO as GPIO

import time

from pubnub import Pubnub

# Initialize the Pubnub Keys

pub\_key = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

sub\_key = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

LIGHT = 18           #define pin of RPi on which you want to take output

def init():          #initalize the pubnub keys and start subscribing

 global pubnub    #Pubnub Initialization

 GPIO.setmode(GPIO.BCM)

 GPIO.setwarnings(False)

 GPIO.setup(LIGHT,GPIO.OUT)

 GPIO.output(LIGHT, False)

 pubnub = Pubnub(publish\_key=pub\_key,subscribe\_key=sub\_key)

 pubnub.subscribe(channels='alexaTrigger', callback=callback, error=callback, reconnect=reconnect, disconnect=disconnect)

def control\_alexa(controlCommand):          #this function control Aalexa, commands received and action performed

 if(controlCommand.has\_key("trigger")):

  if(controlCommand["trigger"] == "light" and controlCommand["status"] == 1):

   GPIO.output(LIGHT, True)

   print "light is on"

  else:

   GPIO.output(LIGHT, False)

   print "light is off"

 else:

  pass

def callback(message, channel):        #this function waits for the message from the aleatrigger channel

 if(message.has\_key("requester")):

  control\_alexa(message)

 else:

  pass

def error(message):                    #if there is error in the channel,print the  error

 print("ERROR : " + str(message))

def reconnect(message):                #responds if server connects with pubnub

 print("RECONNECTED")

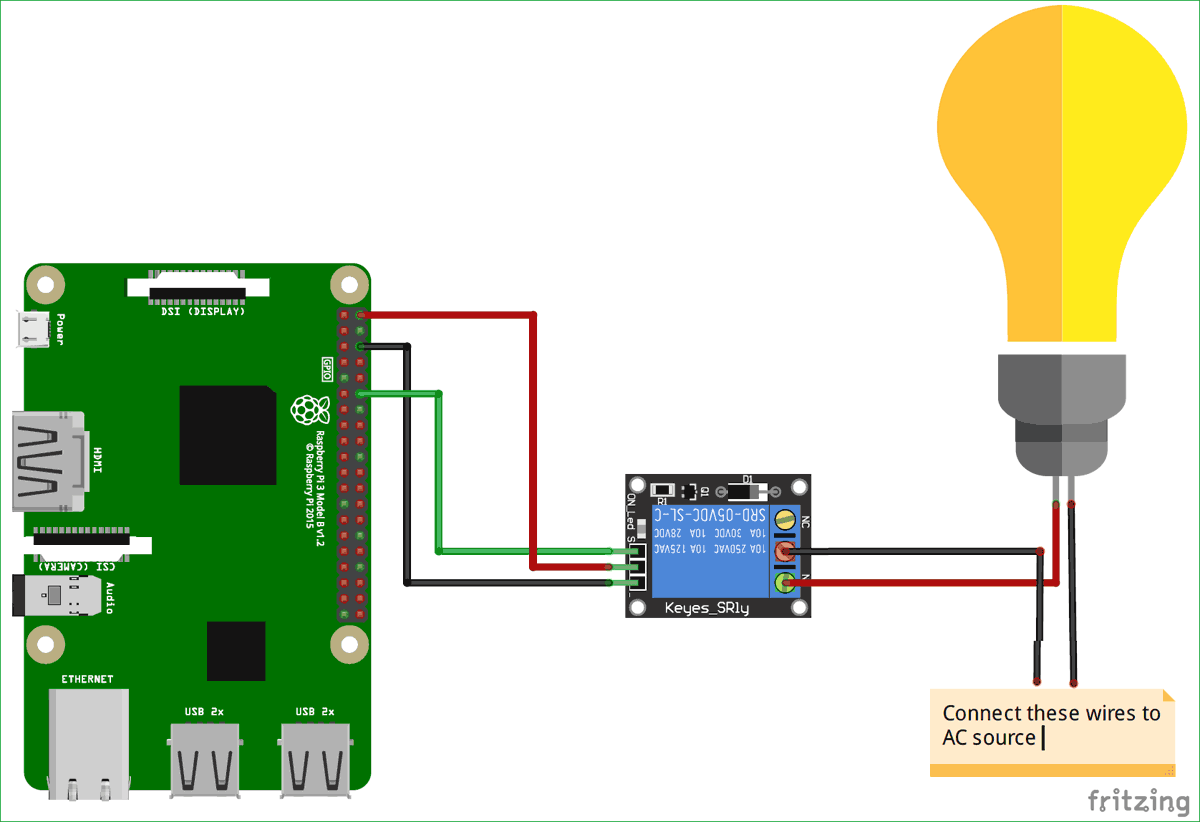
def disconnect(message):               #responds if server disconnects with pubnub

 print("DISCONNECTED")

if \_\_name\_\_ == '\_\_main\_\_':

 init()                    #Initialize the Scrip

**Hardware connections and Circuit Diagram**



Reference links:

1.<https://developer.amazon.com/docs/alexa-voice-service/set-up-raspberry-pi.html>

2. <https://admin.pubnub.com/#/login>

3.<https://circuitdigest.com/microcontroller-projects/voice-controlled-home-automation-using-amazon-alexa-and-raspberry-pi>

4. <https://ifttt.com/>

5. <https://richardtech.net/2019/03/alexa-on-raspberry-pi/>