Controlling LEDs via voice commands and web server

The aim of this project is to give us the ability to control any electronic device (ex LEDs) us via voice commands OR a remote web server using a raspberry Pi.

Hardware required:

1- Raspberry Pi
2- SD card (8 Gb minimum)
3- USB microphone
4- Speaker
5- Breadboard
6- LEDs
7- Resistors
8- Jumper wires

we first need to setup the Raspberry Pi by installing the latest version of Raspbian operating system.

A- Controlling LEDs using voice commands

1- Converting the Raspberry Pi to a google assistant:

We can use the google assistant SDK library to convert the Raspberry Pi to a google assistant device that enables us to control any device using voice commands.

To do that we must have a google account and follow the steps from the google developers

https://developers.google.com/assistant/sdk/guides/library/python/.

now the google assistant should be running on our Raspberry Pi and ready to be used.

We can ask the google assistant any question we like (what time is it? How will the weather be in Waterford tomorrow? Who is Irish prime minister?) or ask the google assistant to sing a song (Can you play a song?), and all of that should be answered.

2- Extend the google assistant to include device actions:

To enable the Raspberry Pi to control devices using the google assistant SDK we need to associate a query with a command to be sent to the device, to do that, we need what is called TRAITS which need to be declared within our device model.
Google has some traits but can still make our own, in this project we will use the onOFF trait.

To add a trait we can follow the steps in the google developers https://developers.google.com/assistant/sdk/guides/library/python/extend/register-device-traits

After we have added our traits, we need the google assistant to handle the commands to be converted form a voice commands to the action required, for that we will follow the steps from the google developers https://developers.google.com/assistant/sdk/guides/service/python/extend/handle-device-commands

After install the RPi.GPIO package in the virtual environment by running the command:

```
pi@raspberrypi ~ $ pip install RPi.GPIO
```

We just need to modify the sample code so that we can be able to control the devices (LEDs) by modifying the code by running the command:

```
pi@raspberrypi ~ $ cd assistant-sdk-python/google-assistant-sdk/googlesamples/assistant/grpc
```

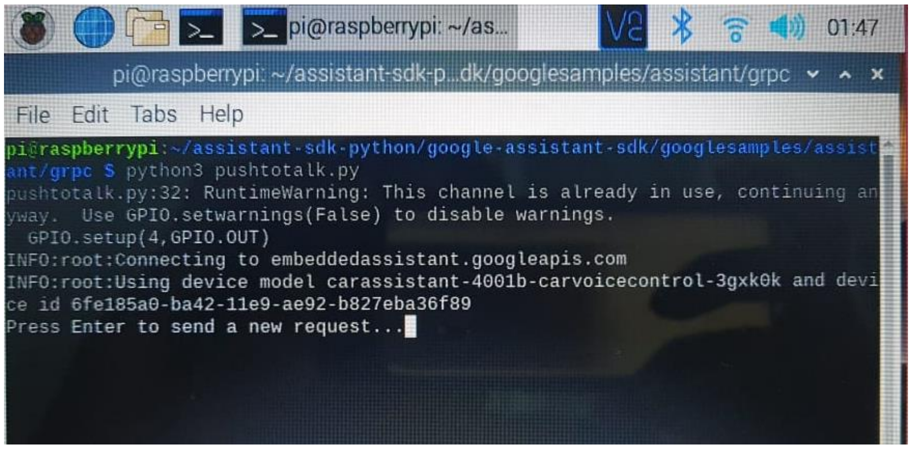
```
pi@raspberrypi ~ $ nano pushtotalk.py
```

then we have to add the following lines to code:

```
device_handler = device_helpers.DeviceRequestHandler(device_id)
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT, initial=GPIO.LOW)

@device_handler.command('action.devices.commands.OnOff')
def onoff(on):
    if on:
        logging.info('Turning device on')
        GPIO.output(4, 1)
    else:
        logging.info('Turning device off')
        GPIO.output(4, 0)
```

After running the command *python pushtotalk.py,* the terminal should look like this:


pi@raspberrypi: ~/as...
01:47
pi@raspberrypi: ~/assistant-sdk-p...dk/googlesamples/assistant/grpc
File Edit Tabs Help

```
pi@raspberrypi:~/assistant-sdk-python/google-assistant-sdk/googlesamples/assist
ant/grpc $ python3 pushtotalk.py
pushtotalk.py:32: RuntimeWarning: This channel is already in use, continuing an
yway.  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(4,GPIO.OUT)
INFO:root:Connecting to embeddedassistant.googleapis.com
INFO:root:Using device model carassistant-4001b-carvoicecontrol-3gxk0k and devi
ce id 6fe185a0-ba42-11e9-ae92-b827eba36f89
Press Enter to send a new request...
```

To toggle the LED connected to the GPIO 4, all we need is to press “enter” and say TRUN ON and the LED should turn on, to turn it off we press the “enter” and say TURN OFF, and the LED should turn off.

B- Controlling the LEDs from a web server using Flask:

We will create a standalone web server that can control two LEDs. In order to create the web server, we will use a Python microframework called Flask.

1- install Flask by running the following commands to update the Raspberry Pi and install pip:

```
pi@raspberrypi ~ $ sudo apt-get update
pi@raspberrypi ~ $ sudo apt-get upgrade
pi@raspberrypi ~ $ sudo apt-get install python-pip python-flask
```

Then, use pip to install Flask and its dependencies:

```
pi@raspberrypi ~ $ sudo pip install flask
```

2- Python script:

This is the script that sets up the server and interacts with the Raspberry Pi GPIOs.
We start by creating a new folder:

```
pi@raspberrypi ~ $ mkdir web-server
pi@raspberrypi ~ $ cd web-server
pi@raspberrypi:~/web-server $
```

a- create a new file called app.py

```
pi@raspberrypi:~/web-server $ nano app.py
```

we copy and paste the code *app.py* from the repository.

b- create the HTML file

Flask uses a template engine called Jinja2 that you can use to send dynamic data from your Python script to your HTML file.

Create a new folder called templates:

```
pi@raspberrypi:~/web-server $ mkdir templates
pi@raspberrypi:~/web-server $ cd templates
pi@raspberrypi:~/web-server/templates $
```

create a new file called main.html:

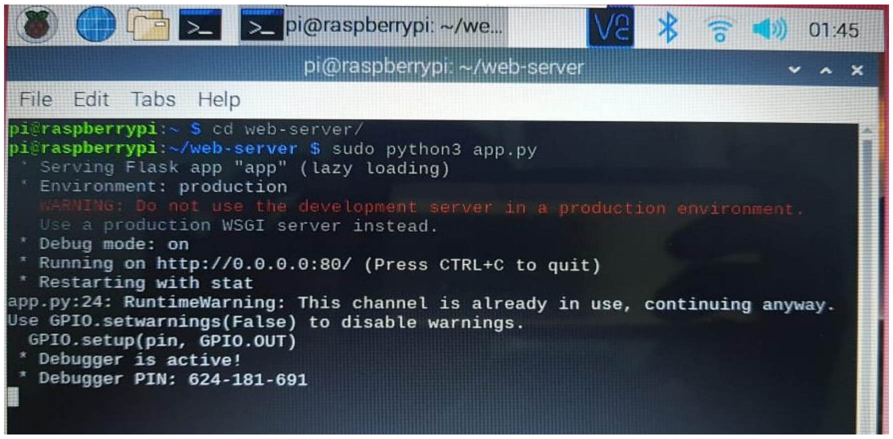
```
pi@raspberrypi:~/web-server/templates $ nano main.html
```

Then we copy and paste the code *main.html* from the repository:

To launch the web server by running the following command:

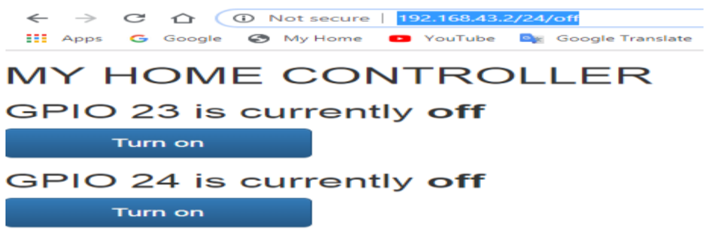
```
pi@raspberrypi:~/web-server $ sudo python app.py
```

```
The web server will start immediately
```



```
pi@raspberrypi:~ $ cd web-server/
pi@raspberrypi:~/web-server $ sudo python3 app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
app.py:24: RuntimeWarning: This channel is already in use, continuing anyway.
Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(pin, GPIO.OUT)
 * Debugger is active!
 * Debugger PIN: 624-181-691
```

Finally we demonstrate by opening the raspberry Pi address in the browser by entering its IP address



MY HOME CONTROLLER

GPIO 23 is currently **off**

GPIO 24 is currently **off**

We can now control the LEDs connected to GPIO 23 and GPIO 24 using the buttons on the web server from anywhere.