

# EE 5103 Engineering Programming Project Final Report

## Title

C++ HTTP client interfaced with Google Assistant on Raspberry Pi

## Abstract

This project accepts request from Google assistant connected with users Gmail account and switches ON appropriate home appliances connected with Raspberry pi board. C++ code will be developed and deployed on Raspberry pi which contains Debian Linux OS.

## 1. Introduction:

### **1.1 Problem Spaces:**

- Currently Google assistant, Amazon Alexa, Apples Siri is becoming very famous and dominating market. User has to pay significant amount of money for use of these devices. This project gives facility of accepting voice commands from google assistant (i.e. Google server) connected with User's Gmail account. The C++ code will accept web request from Google and get data. User can connect two home appliances such as light, coffee machine etc.

This project gives end to end smart home IoT project with only approx. \$40 in which raspberry costs around just \$35 and relays with \$5.

### **1.2 Goal of this project:**

- This is voice operated 'IoT project' which will able to connect and receive web request from Google assistant server on raspberry Pi and operate home appliances connected to raspberry pi.

## 2. Devices used:

- 2 Raspberry Pi
- 2 Relay 240v 50-60 Hz 10A
- USB Microphone
- Study lamp

## 3. Block Diagram:

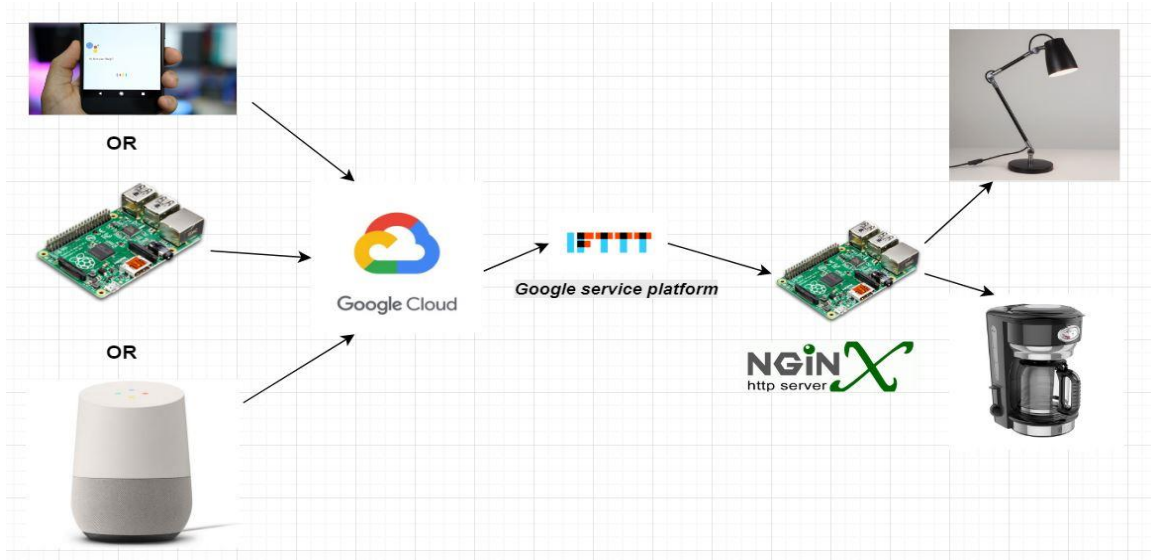


Fig.1 Block diagram of project

#### 4. Solution from Programming:

The C++ code will be running on raspberry pi which accepts web request from Google server and operate table lamp or coffee machine connected with raspberry pi.

An open source server i.e. NGINX is installed on raspberry pi to make board online.

Flowchart of project is as below:

- **Google Home (Rpi 1):** Google assistant SDK installed on Rpi 1.
- **IFTTT:** Google's sister company gives API service facility (This api access is given through registering with Gmail account)
- **Dataplicity:** Makes Rpi 2 online globally ( gives free service for first 10 devices)
- **Nginx Server on Rpi 2:** Used to receive HTTP web request
- **FASTCGI C++ application:** a CGI( common gateway interface) application is running which receives webrequest and configures GPIO pins accordingly on Raspberry Pi

## **5. Technical details:**

Below steps explains detailed instruction to download and install Google Assistant SDK on raspberry pi. We need to register with Gmail account to get API access.

### **5.1: Microphone usage on Raspberry pi :**

An USB microphone is being used to detect and receive voice commands from user. The price of this USB mic was \$ 5 and bought from amazon.com. Below is configuration file for setting up mic on Rpi3. We must create `.asoundrc` config file for raspberry pi platform. This file need to be saved at `/home/pi` location.

```
pcm.!default {
    type asym
    capture.pcm "mic"
    playback.pcm "speaker"
}
pcm.mic {
    type plug
    slave {
        pcm "hw:<card number>,<device number>"
    }
}
pcm.speaker {
    type plug
    slave {
        pcm "hw:<card number>,<device number>"
    }
}
```

### **5.2: Configuration of Google assistant SDK on Raspberry Pi:**

Google has given full SDK as library on the documentation website. I have followed the instructions and registered with personal Gmail account.

(Ref: <https://developers.google.com/assistant/sdk/guides/library/python/>)

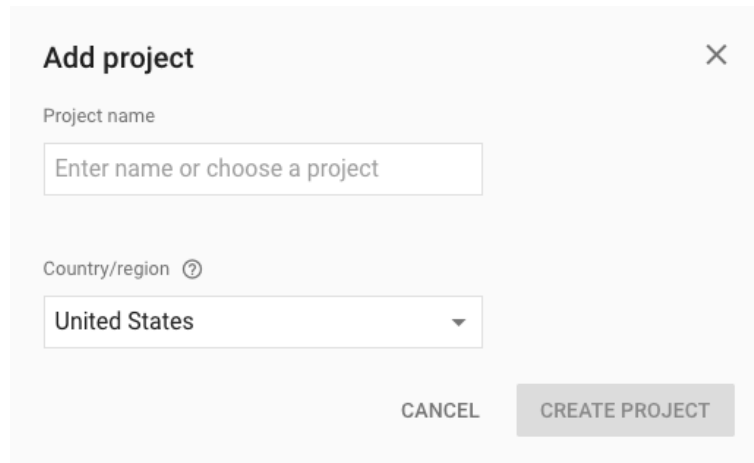
### **5.3: Configure an Actions Console project:**

A Google Cloud Platform project, managed by the Actions Console, gives your device access to the Google Assistant API. The project tracks quota usage and gives you valuable metrics for the requests made from your device.

To enable access to the Google Assistant API, need to following steps:

1. Open the Actions Console.
2. Click on Add/import project.
3. To create a new project, type a name in the Project name box and click CREATE PROJECT.

Name: GOPAL VISHWAKARMA(dde911)



The 'Add project' dialog box features a title bar with a close button (X). It contains a 'Project name' label above a text input field with the placeholder 'Enter name or choose a project'. Below this is a 'Country/region' label with a help icon (?) above a dropdown menu currently showing 'United States'. At the bottom right are two buttons: 'CANCEL' and 'CREATE PROJECT'.

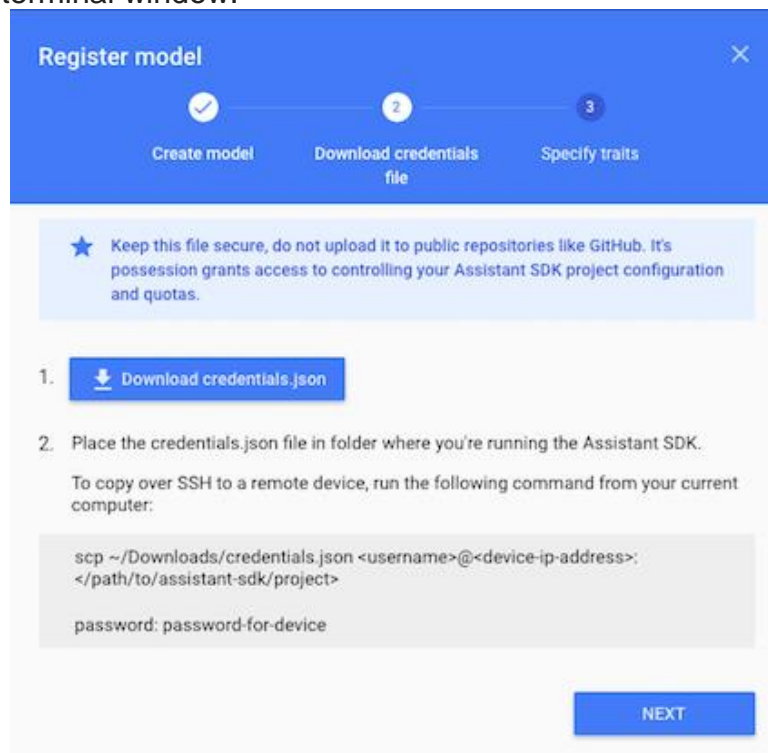
3.Enable the Google Assistant API on the project you selected. You need to do this in the Cloud Platform Console.

#### 5.4: Register the Device Model:

1. Open a new terminal window. Run the following command:

```
scp ~/Downloads/credentials.json pi@raspberrypi-ip-ddress:/home/pi/  
password: password-for-device
```

2. Close this terminal window.



The 'Register model' dialog box has a blue header with a close button (X) and a progress indicator showing three steps: '1. Create model' (checked), '2. Download credentials file' (active), and '3. Specify traits'. A warning message with a star icon states: 'Keep this file secure, do not upload it to public repositories like GitHub. It's possession grants access to controlling your Assistant SDK project configuration and quotas.' Below this, step 1 is 'Download credentials.json' with a download icon. Step 2 is 'Place the credentials.json file in folder where you're running the Assistant SDK.' followed by the instruction 'To copy over SSH to a remote device, run the following command from your current computer:' and a code block containing:  

```
scp ~/Downloads/credentials.json <username>@<device-ip-address>:  
</path/to/assistant-sdk/project>  
  
password: password-for-device
```

  
A 'NEXT' button is at the bottom right.

Name: GOPAL VISHWAKARMA(dde911)

3. Save and proceed for next step.

### 5.5: Install the SDK:

To Install SDK, we need to set up python virtual environment. Below are series of commands that need to be executed to set up environment.

```
sudo apt-get update

sudo apt-get install python3-dev python3-venv # Use python3.4-venv
if the package cannot be found.

python3 -m venv env

env/bin/python -m pip install --upgrade pip setuptools wheel

source env/bin/activate
```

### 5.6: Get the Packages:

The Google Assistant SDK package contains all the code required to get the Google Assistant running on the device, including the sample code.

```
sudo apt-get install portaudio19-dev libffi-dev libssl-dev
```

### 5.7: Generate credentials:

1. Install or update the authorization tool:

```
python -m pip install --upgrade google-auth-oauthlib[tool]
```

2. Generate credentials to be able to run the sample code and tools. Reference the JSON file you downloaded in a previous step; you may need to copy it to the device. Do not rename this file.

```
google-oauthlib-tool -scope
https://www.googleapis.com/auth/assistant-sdk-prototype --save --
headless --client-secrets /path/to/credentials.json
```

Name: GOPAL VISHWAKARMA(dde911)

3. Copy the URL and paste it into a browser (this can be done on any machine). The page will ask you to sign in to your Google account. Sign into the Google account that created the developer project in the previous step.

4. After you approve the permission request from the API, a code will appear in your browser, such as "4/XXXX". Copy and paste this code into the terminal:

```
Enter the authorization code:
```

If authorization was successful, you will see a response similar to the following:

```
credentials saved: /path/to/.config/google-oauthlib-  
tool/credentials.json
```

### 5.8: Command to run Google assistant:

After above all steps our Google assistant is ready. It can be executed with following commands:

```
env/bin/python3 -u /root/env/bin/googlesamples-assistant-hotword --  
project_id ga-pi-194622 --device_model_id ga-pi-194622
```

## **6. NGINX server and FastCGI setup on 2nd raspberry pi:**

We installed NGINX a open source server which receives web request from Google server and forward that request to CGI interface which parses data accordingly. FCGI (FastCGI) is a protocol in which web applications can talk to a web server to serve web requests. Basically, FCGI spawns static number of processes to handle web requests.

### 6.1: Installation:

Below are required libraries that need to be installed on Rpi:

```
sudo apt-get install libfcgi-dev  
sudo apt-get install spawn-fcgi  
sudo apt-get install nginx  
sudo apt-get install curl
```

### 6.2: NGINX config file:

NGINX server config file location is 'nano /etc/nginx/sites-enabled/default' and configured as below. It listens on port 80 and forwards all request to 8000 port.

```
server{
listen 80 default_server;
listen [::]:80 default_server;
server_tokens off;

root /var/www/html;

# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;

server_name _;

    location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ =404;
    }

    location /incoming {
        fastcgi_pass 127.0.0.1:8000;
    }
}
```

### 6.3: Running C++ application code:

```
# compile code
g++ mainV21.cpp GPIO_class.cpp GPIO_class.h -lfcgi++ -lfcgi
-o mainexe

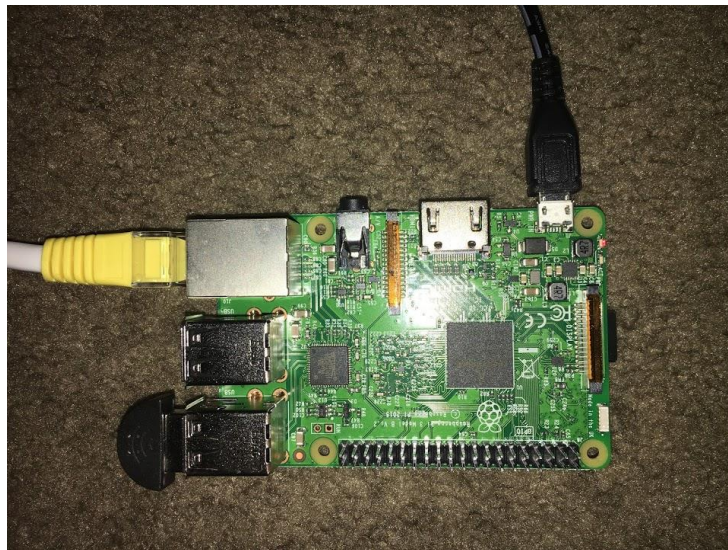
# spawn the fcgi app on port 8000 with no fork
spawn-fcgi -p 8000 -n mainexe
```



Name: GOPAL VISHWAKARMA(dde911)

## **7. Setup:**

### **1<sup>st</sup> Raspberry Pi (Google Assistant):**



### **2<sup>nd</sup> Raspberry Pi (Relay and NGINX server setup):**





Name: GOPAL VISHWAKARMA(dde911)



### **Conclusion:**

This project successfully installed Google assistant SDK on raspberry platform and created virtual project on Google developers platform and accessed services through Google Api's. Moreover, this project successfully installed an open source server called NGINX server and processed incoming web request through FastCGI interface on 2<sup>nd</sup> raspberry Pi.

Table lamp is connected to raspberry pi GPIO through 120V DC relay. Basically, First raspberry Pi gets voice command and push to Google server. At 2<sup>nd</sup> raspberry pi, Incoming web request from Google server is processed through C++ application to turn ON/OFF table lamp.