This document explains how I set up Home Assistant with Google API SDK. There may be variation with the actual procedure because this document was written after implementation.

## Project Purpose:

The purpose of this project was to provide a button to the home user to turn on/off devices linked to Google Assistant. In my case, I have a Google Home that has devices such as a lamp that can be turned on/off by a Wi-Fi smart plug. The physical switch on the lamp is always in the “on” position as it is plugged into the Wi-Fi smart plug that is controlled by Google Assistant and Google Home.

Once you integrate a smart device, this restricts the home user to using voice commands or the Google Assistant Mobile app to control the device. **However**, what if a guest comes to stay at your house and is unfamiliar with voice commands for Google Home, they may not have a way to turn the device on/off.

## Solution:

1) Install home-assistant.io on a local computer (in my case Raspberry PI 4) running Ubuntu
2) Enable API calls from your Google API SDK on your Google account
3) Enable API calls to your Home Assistant app
4) Setup a Raspberry PI computer, in my case a Raspberry Pi 3 with a breadboard, a tactile button and wires to the GPIO of Raspberry PI and a Python script running on it.

Yes, that is right I am using a Raspberry Pi 4 and a Raspberry Pi3, two separate Raspberry Pi’s. You could have this setup on one Raspberry Pi, but I chose to separate my Home Assistant server from the Raspberry Pi that was accepting input from the tactile button.

### Procedure:

1) Install home-assistant on a Raspberry Pi running Ubuntu. Here is good Youtube site: https://youtu.be/Cvjc66-mkFo?si=NGMtMpalOc7q_UUv

   Here is an official docker site info on installing Docker on Ubuntu
   https://docs.docker.com/engine/install/linux-postinstall/
   https://docs.docker.com/engine/install/ubuntu/

   Create the `docker` group.

   ```
   sudo groupadd docker
   ```

   Add your user to the `docker` group.

   ```
   sudo usermod -aG docker $USER
   ```

   ```
   Sudo apt-get update
   Sudo apt-get install ca-certificates curl gnupg

   Sudo install -m 0755 -d /etc/apt/keyrings
   ```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg
--dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

echo \

 "deb [arch=$(dpkg --print-architecture)
signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/ubuntu \
 $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
 sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
```

Create a Docker compose file in your directory of choice on your

**docker-compose.yaml**

```
version: '3'
services:
  homeassistant:
    container_name: homeassistant
    image: "ghcr.io/home-assistant/home-assistant:stable"
    volumes:
      - /home/peter/config:/config
      - /etc/localtime:/etc/localtime:ro
    restart: unless-stopped
    privileged: true
    network_mode: host
```

Run:
**docker compose up**

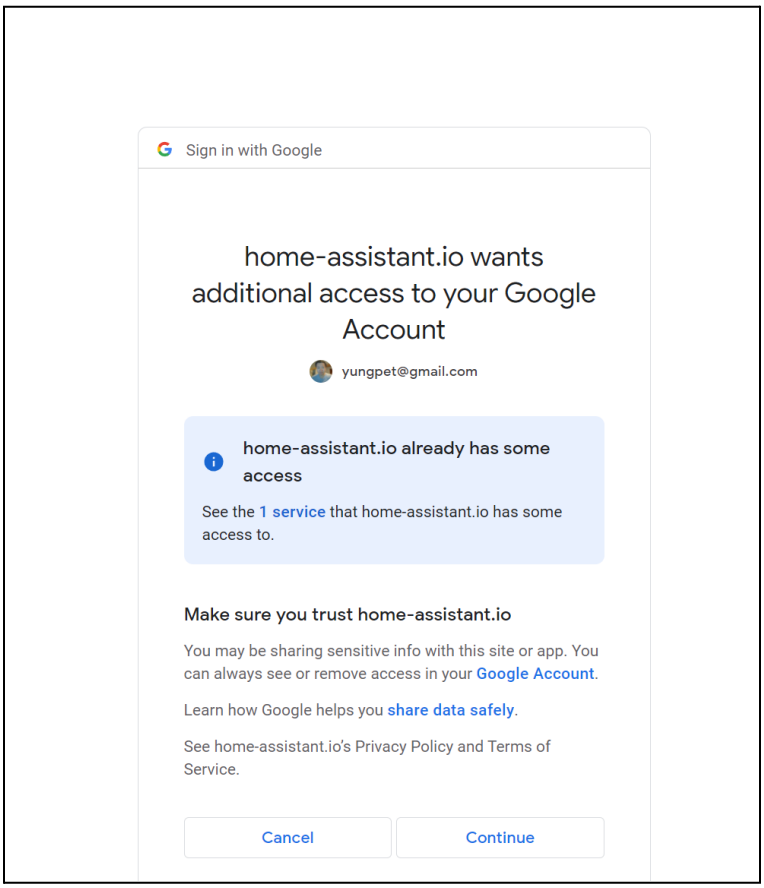
I can now access my Home Assistant locally through

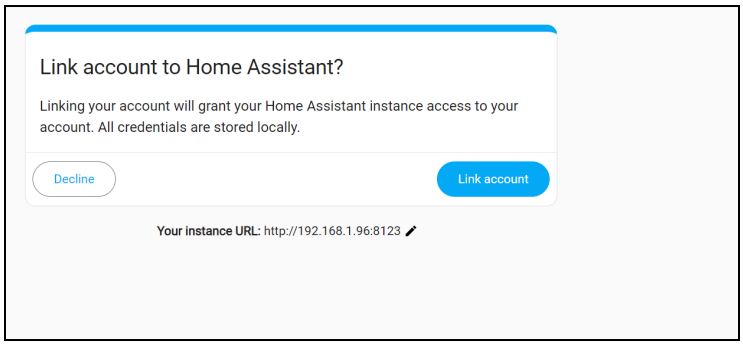
http://192.168.1.96:8123/

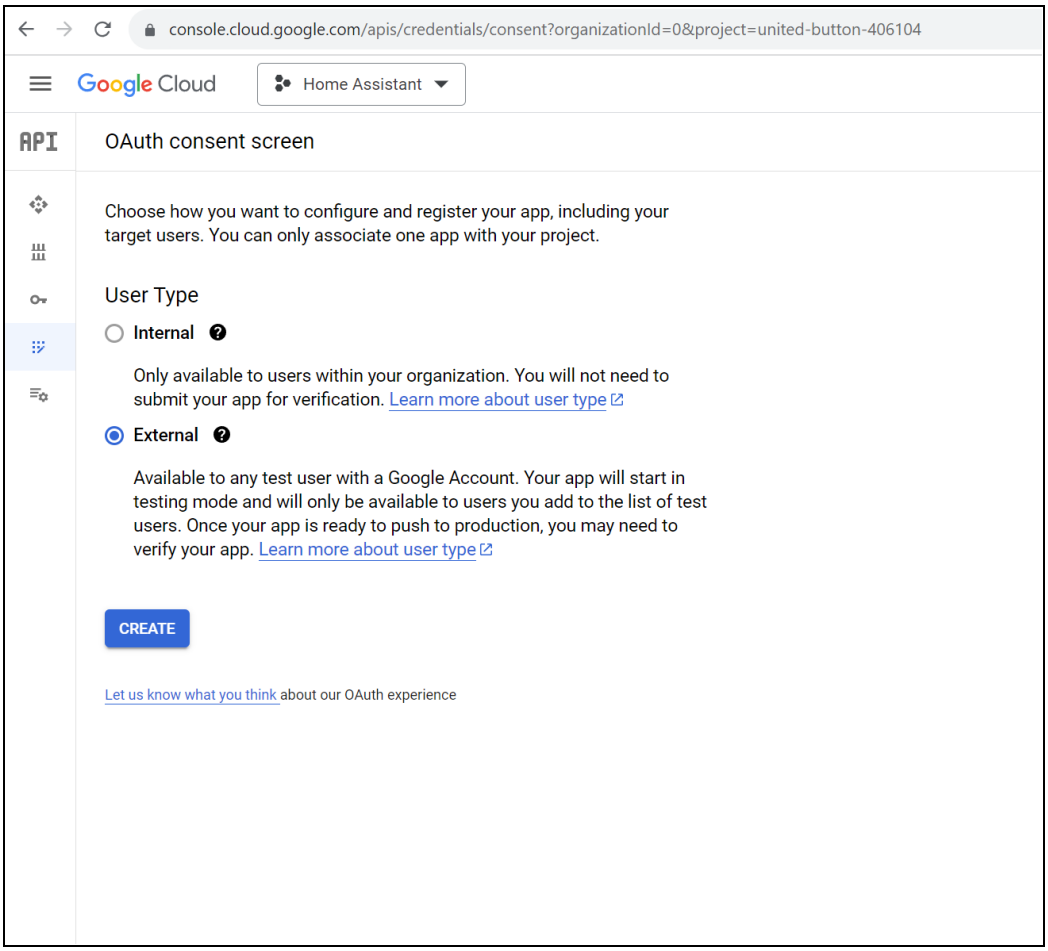
Create your admin account through your local instance of Home Assistant

2. Enable API on your Google Assistant SDK

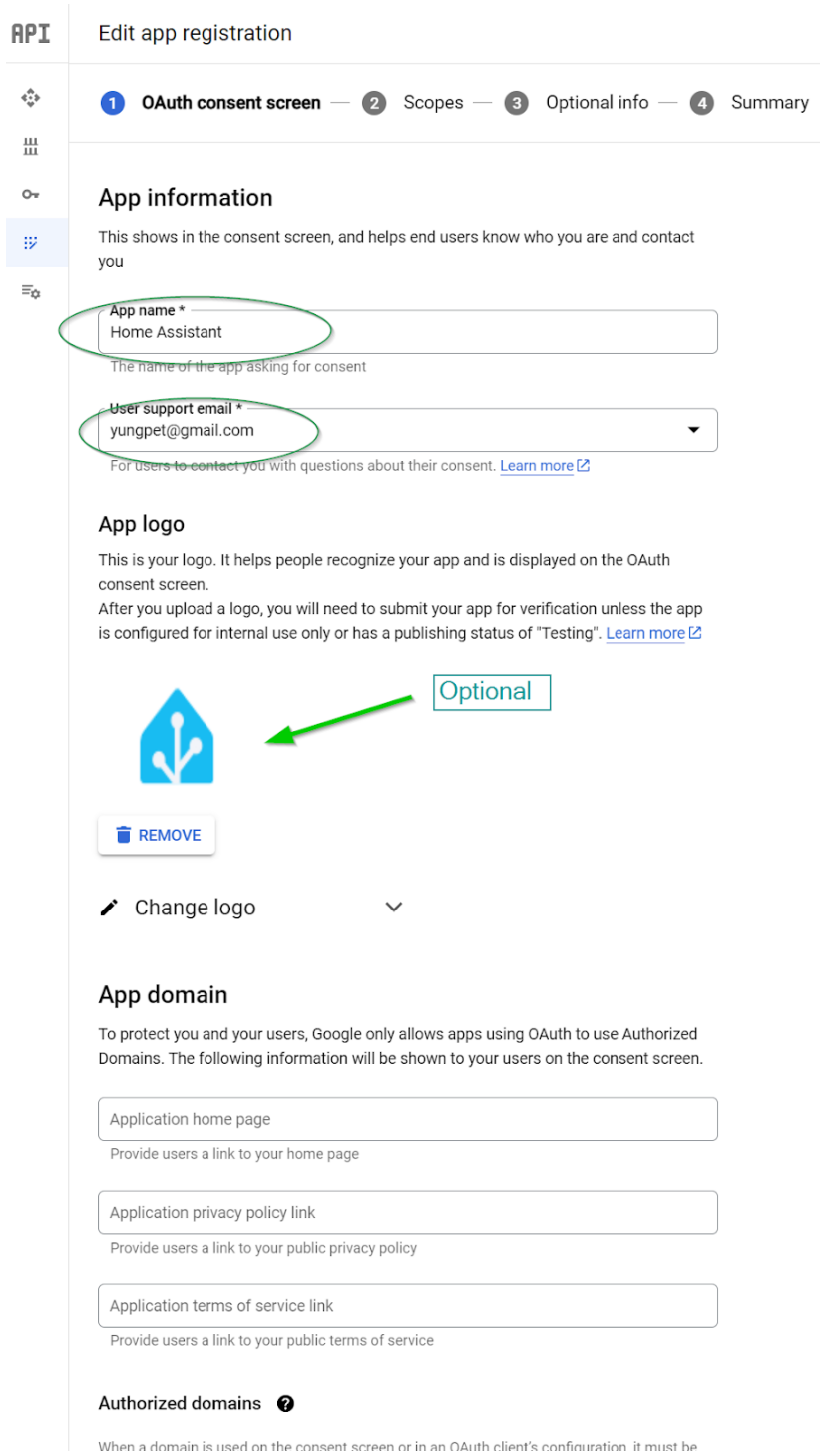
You first need to authenticate from your local Home Assistant server to the Google Assistant SDK.  Under Home Assistant > Settings > Add Integration > Google Assistant SDK
You will be prompted to use an account for OAuth authentication to connect

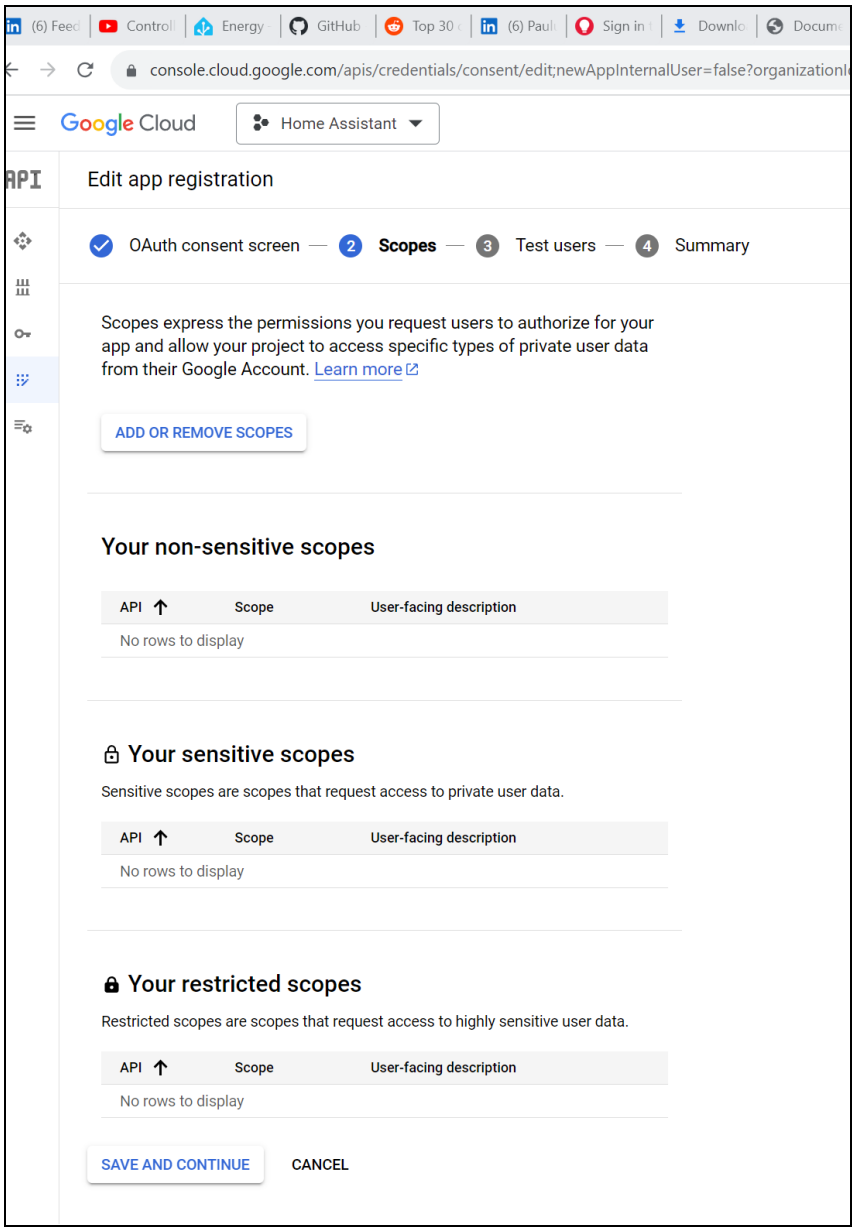


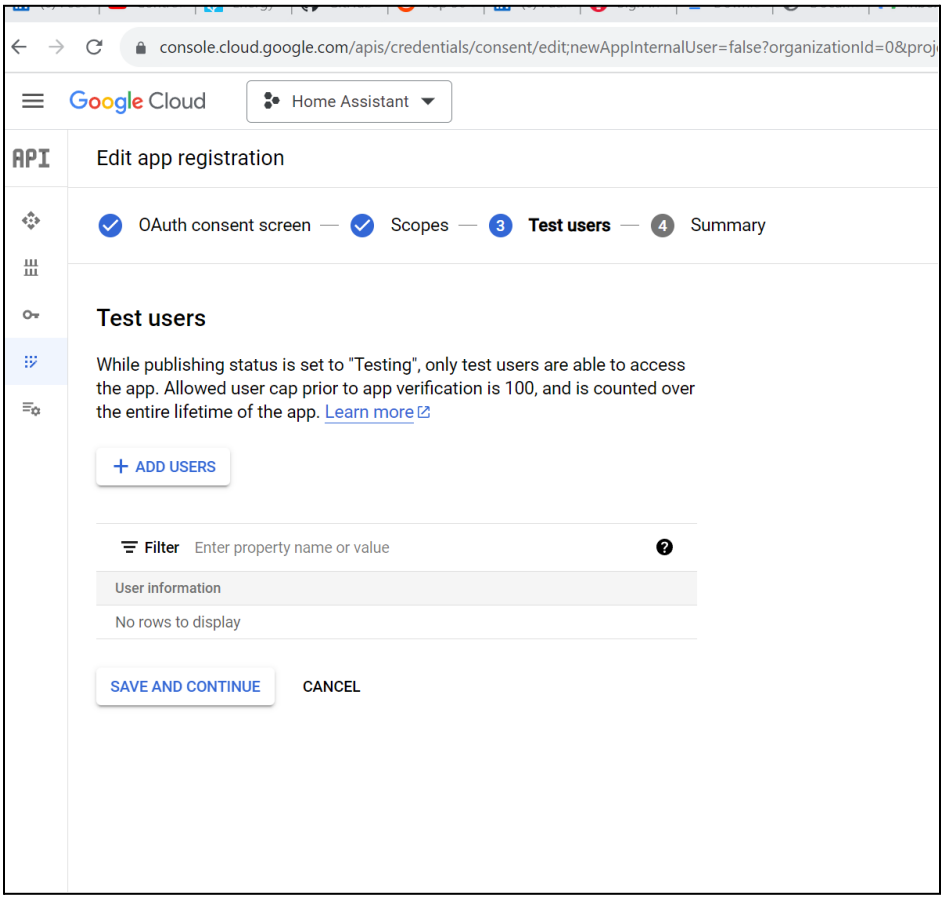

console.cloud.google.com/apis/credentials/consent?organizationId=0&project=united-button-406104
Google Cloud
Home Assistant
API
OAuth consent screen
Choose how you want to configure and register your app, including your target users. You can only associate one app with your project.
User Type
Internal
Only available to users within your organization. You will not need to submit your app for verification. Learn more about user type
External
Available to any test user with a Google Account. Your app will start in testing mode and will only be available to users you add to the list of test users. Once your app is ready to push to production, you may need to verify your app. Learn more about user type
CREATE
Let us know what you think about our OAuth experience

Give an arbitrary name to the App Name and use your email address.

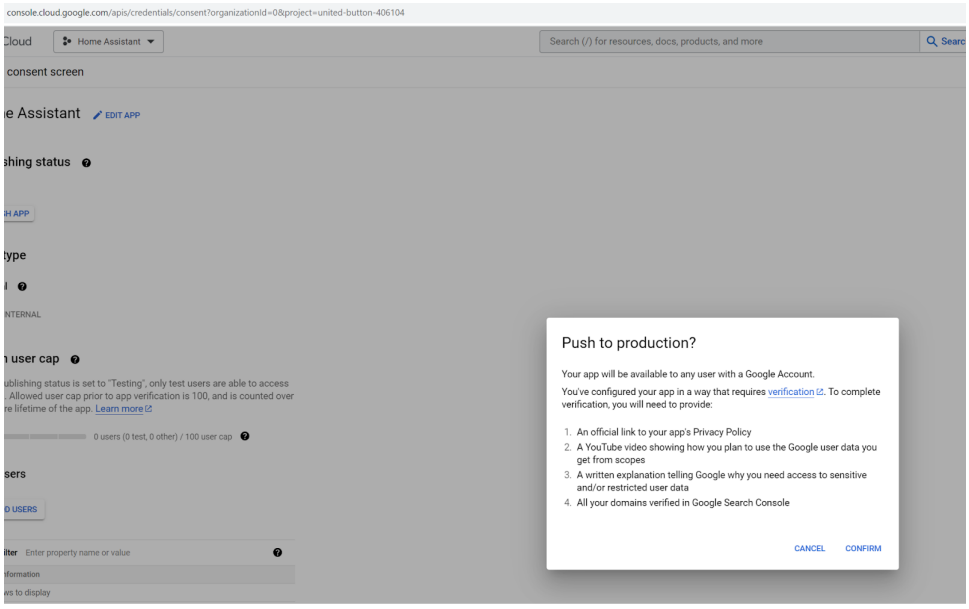



(6) Feed
Control
Energy
GitHub
Top 30
(6) Paul
Sign in
Downlo
Docum
console.cloud.google.com/apis/credentials/consent/edit;newAppInternalUser=false?organizationl
Google Cloud
Home Assistant
API
Edit app registration
OAuth consent screen
2
Scopes
3
Test users
4
Summary
Scopes express the permissions you request users to authorize for your app and allow your project to access specific types of private user data from their Google Account. Learn more
ADD OR REMOVE SCOPES
Your non-sensitive scopes
API
Scope
User-facing description
No rows to display
Your sensitive scopes
Sensitive scopes are scopes that request access to private user data.
API
Scope
User-facing description
No rows to display
Your restricted scopes
Restricted scopes are scopes that request access to highly sensitive user data.
API
Scope
User-facing description
No rows to display
SAVE AND CONTINUE
CANCEL

No test users necessary




I just pressed “Confirm” here.

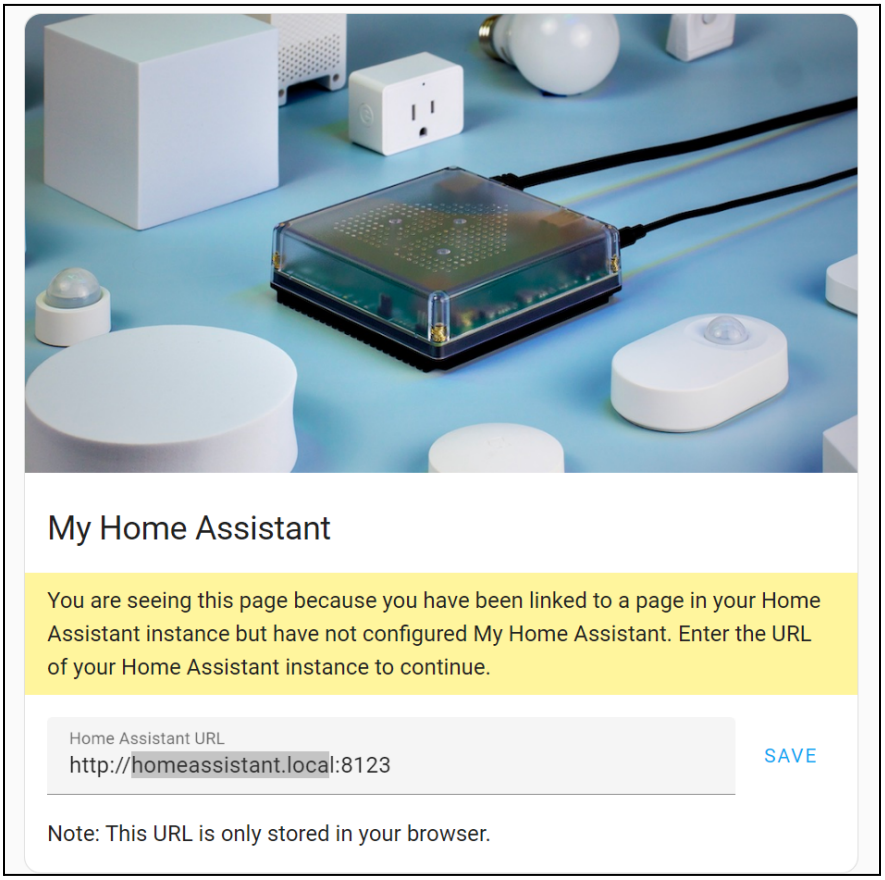


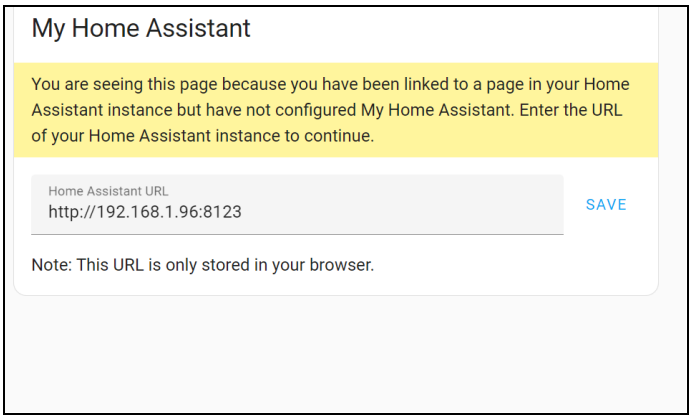

Then go back to Create New Credentials for OAuth,

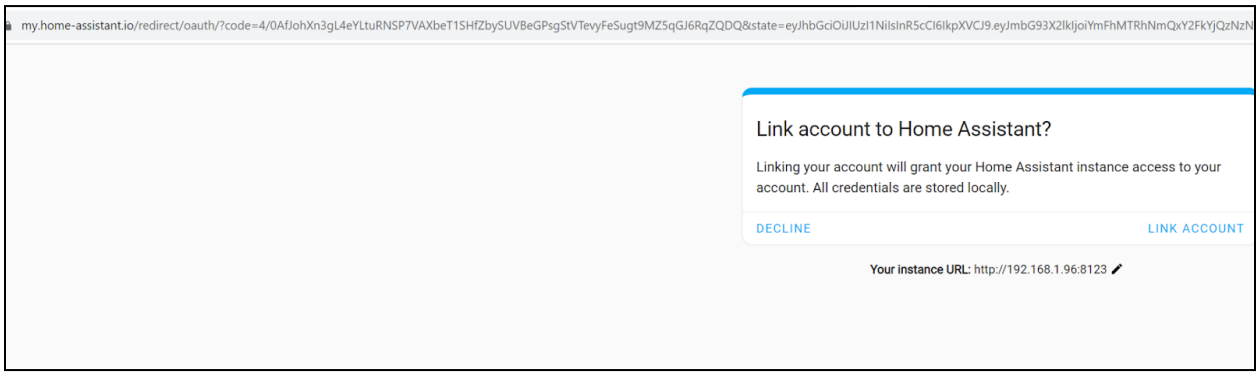
Enter
http://my.home-assistant.io/redirect/oauth
for the url

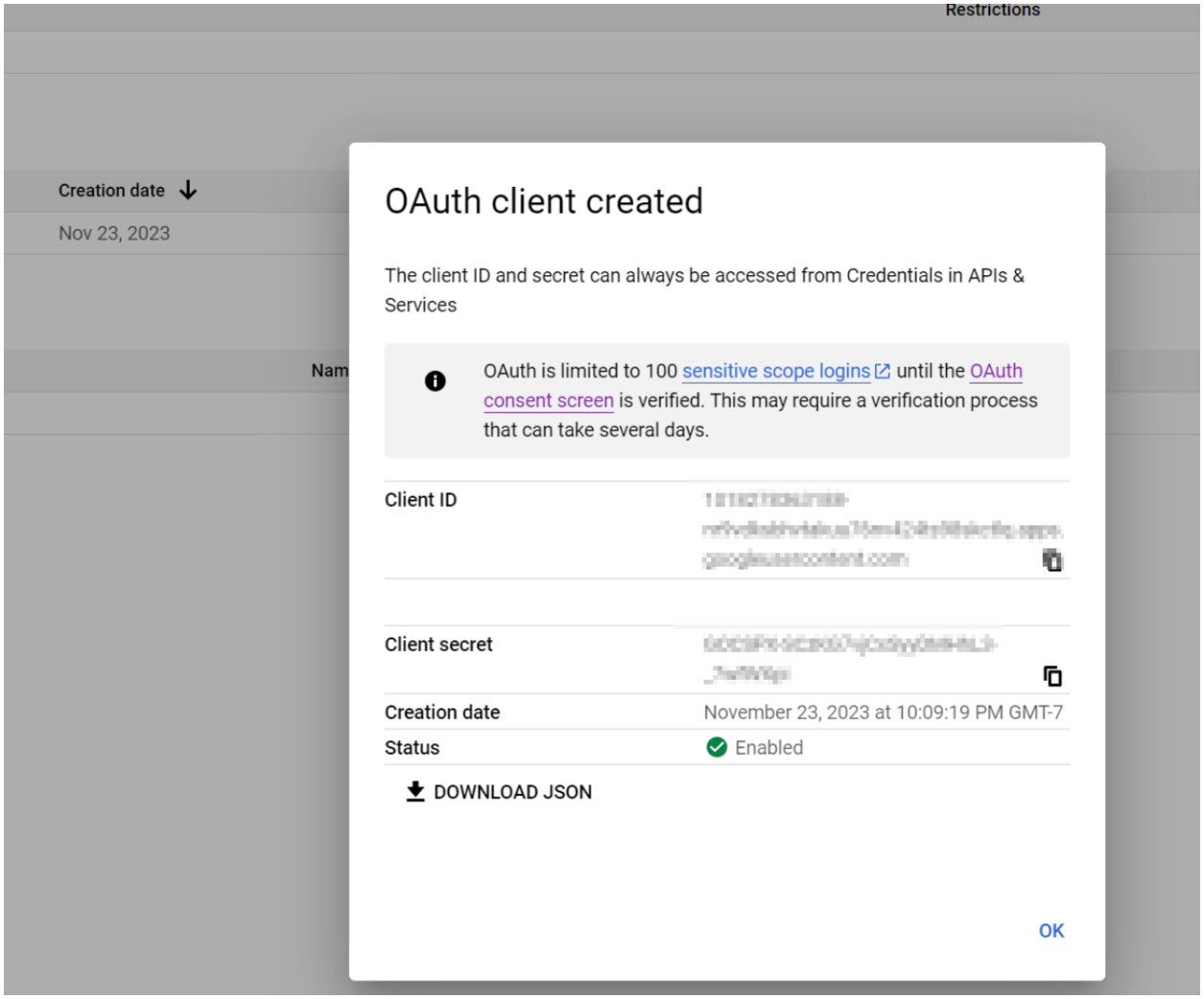
Get the client ID and client secret that it generates.

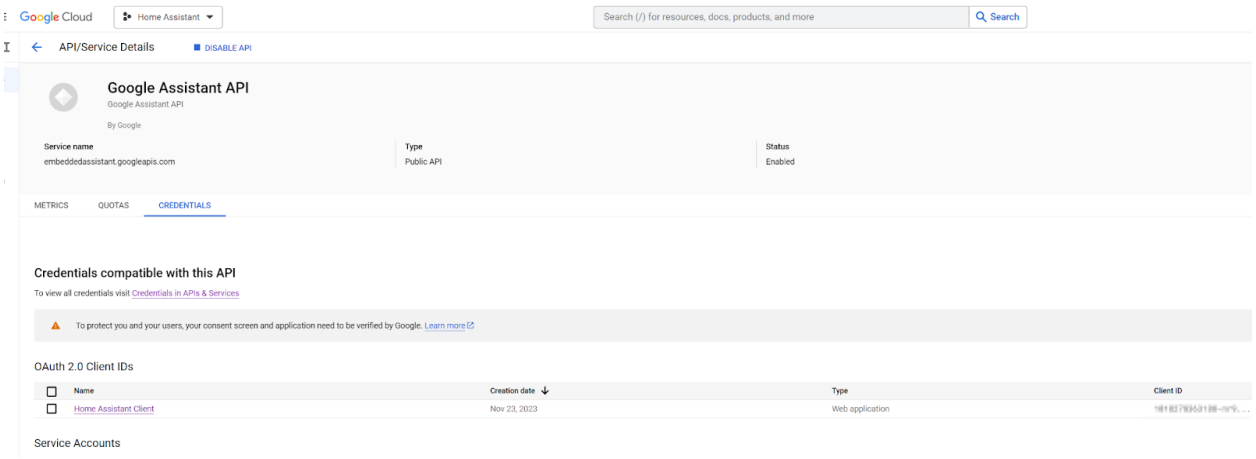
Go back to the home assistant setup screen and fill in the client ID and client secret. It will then bring you to authenticate against Google API with your credentials

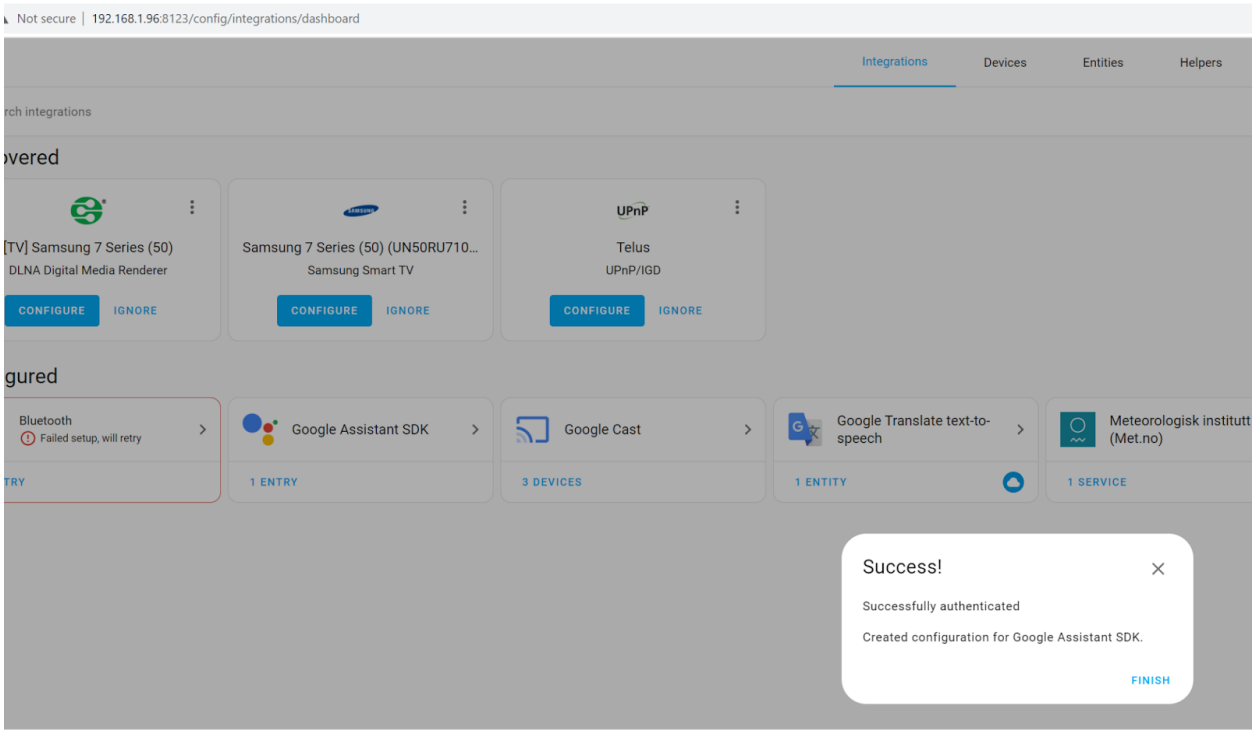



My Home Assistant
You are seeing this page because you have been linked to a page in your Home Assistant instance but have not configured My Home Assistant. Enter the URL of your Home Assistant instance to continue.
Home Assistant URL
http://192.168.1.96:8123
SAVE
Note: This URL is only stored in your browser.


my.home-assistant.io/redirect/oauth/?code=4/0AfJohXn3gL4eYLtuRNSP7VAXbeT1SHfZbySUVBeGPsgStVTevyFeSugt9MZ5qGJ6RqZQDQ&state=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJmbG93X2lkIjoiYmFhMTRhNmQxY2FkYjQzNzN
Link account to Home Assistant?
Linking your account will grant your Home Assistant instance access to your account. All credentials are stored locally.
DECLINE
LINK ACCOUNT
Your instance URL: http://192.168.1.96:8123


Restrictions
Creation date
Nov 23, 2023
Nam
OAuth client created
The client ID and secret can always be accessed from Credentials in APIs & Services
OAuth is limited to 100 sensitive scope logins until the OAuth consent screen is verified. This may require a verification process that can take several days.
Client ID
Client secret
Creation date
November 23, 2023 at 10:09:19 PM GMT-7
Status
Enabled
DOWNLOAD JSON
OK


Google Cloud
Home Assistant
Search (/) for resources, docs, products, and more
Search
API/Service Details
DISABLE API
Google Assistant API
Google Assistant API
By Google
Service name
embeddedassistant.googleapis.com
Type
Public API
Status
Enabled
METRICS
QUOTAS
CREDENTIALS
Credentials compatible with this API
To view all credentials visit Credentials in APIs & Services
To protect you and your users, your consent screen and application need to be verified by Google. Learn more
OAuth 2.0 Client IDs
Name
Creation date
Type
Client ID
Home Assistant Client
Nov 23, 2023
Web application
Service Accounts




3. Enable the Home Assistant API

Configure home-assistant api.

To enable home -assistant api, use the directory that has the volume to the Docker image /home/peter/config/configuration.yaml
Set :

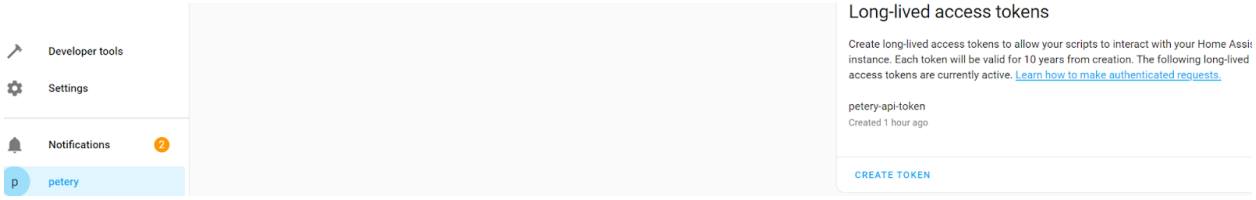
```
default_config:

# Load frontend themes from the themes folder
frontend:
  themes: !include_dir_merge_named themes

automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml

api:
```

Also need to create a token, which is the Authorization Bearer token that you will be using

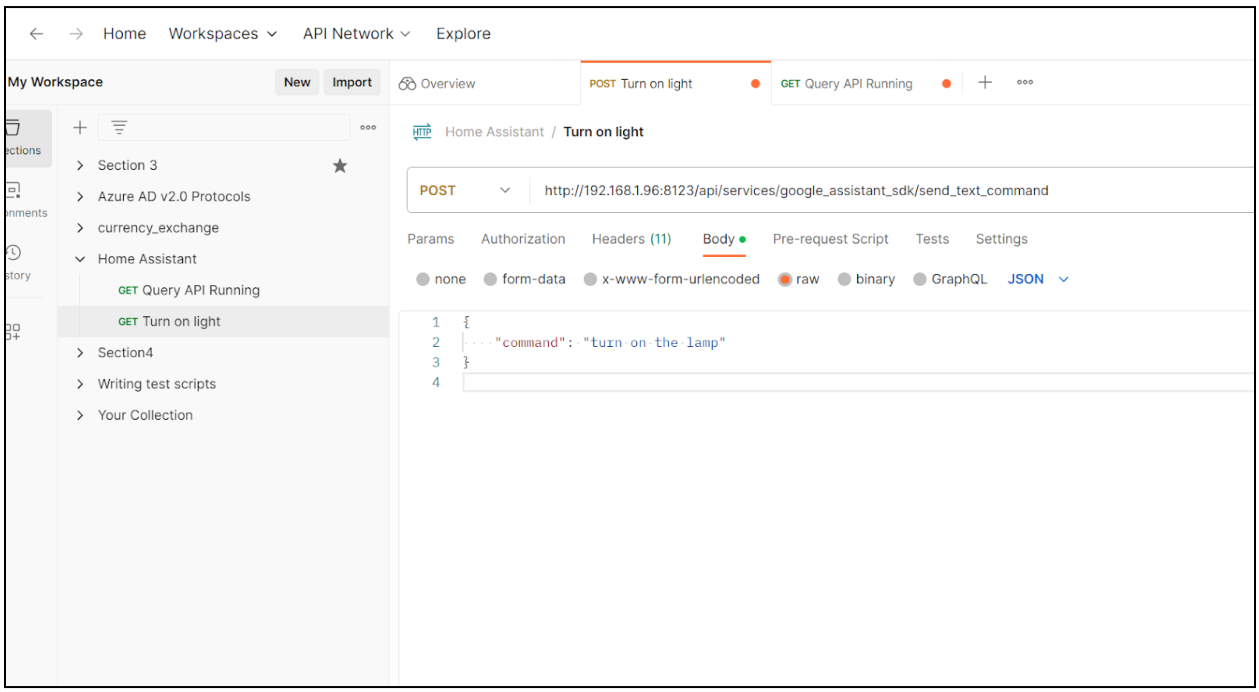


You can test the API by running:

curl -H “Authorization: Bearer <token code>” -H “Content-Type: application/json: http://localhost:8123/api

You can refer to the following URL to test your API and see the services offered https://developers.home-assistant.io/docs/api/rest/

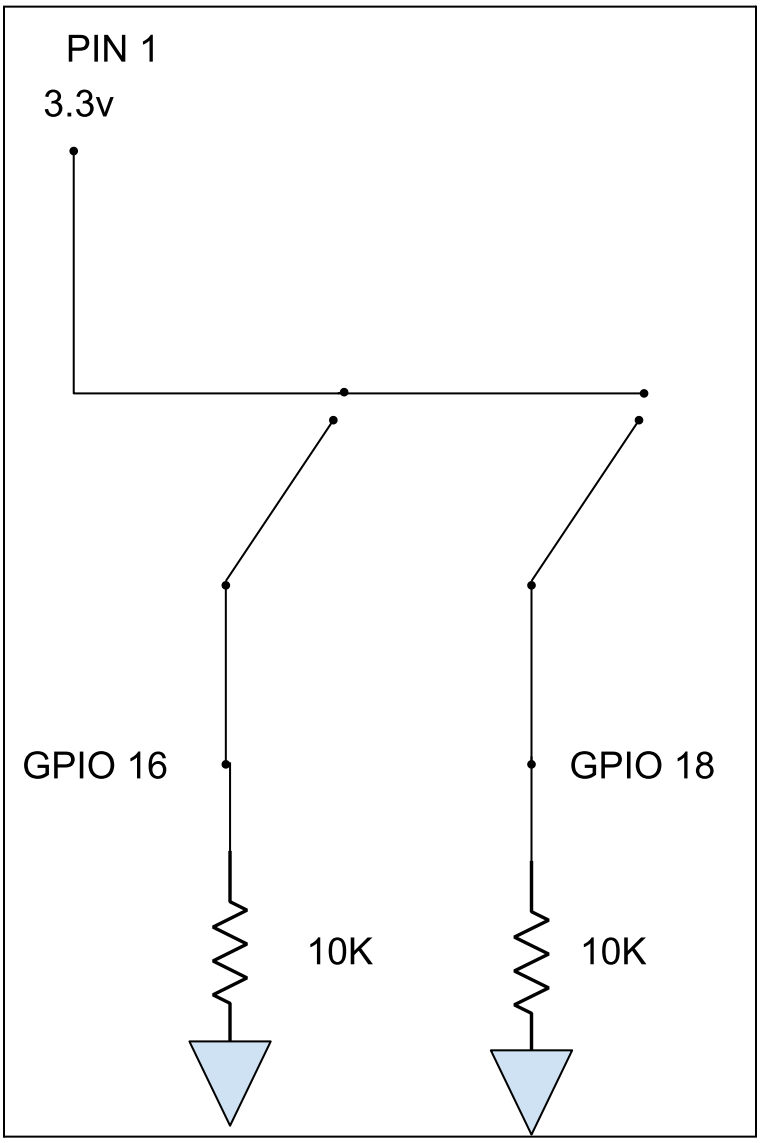
Test your Home Assistant API with all call to the Google Assistant SDK API as a test using POSTMAN

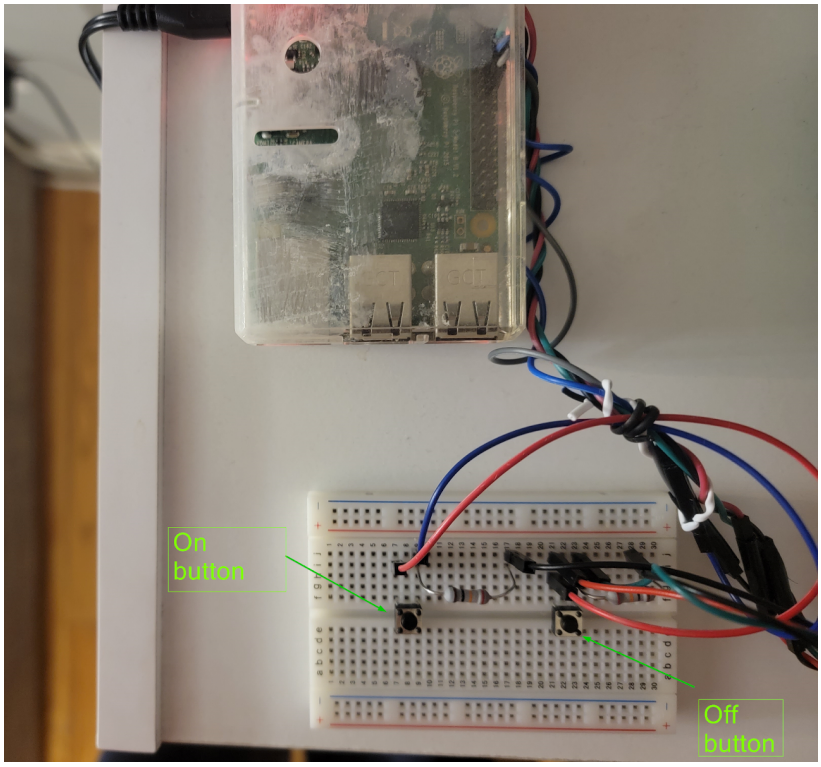



Now use the program in this Github and make REST API calls to your Home Assistant that will call the Google Assistant SDK API because you have linked your Home Assistant server to Google Assistant through OAuth.

If your API call using the POST request runs properly, it should return a null value ‘[]’. This is because of a bug on the Google Assistant SDK side, but don’t worry if it is returning HTTP response code 200, then you are ok.

4. Setup a Raspberry PI computer, my case Raspberry Pi 3 using Python, a bread board, a tactile button and wires to the GPIO of Raspberry PI






Here is a good reference from where I got the GPIO configuration from:

https://toptechboy.com/raspberry-pi-with-lesson-26-controlling-gpio-pins-in-python/

Use my script in GitHub and if everything is set right, you should be able to run

python3 lamp_control.py
On the Linux command line.

Allow your script to run 24x7 by putting it into a Systemd service using a file with contents similar to what I have below:

```
[Unit]
Description=Control Google Assistant sdk device with a button
After=network.target

[Service]
User=pi
Group=pi
WorkingDirectory=/home/pi/lamp_control_home_assistant
ExecStart=/usr/bin/python3 /home/pi/lamp_control_home_assistant/lamp_control.py
Restart=always

[Install]
WantedBy=multi-user.target
```