HOW TO RUN DHT11 APP INTEGRATED WITH GOOGLE SHEETS API AND SQLITE DATABASE

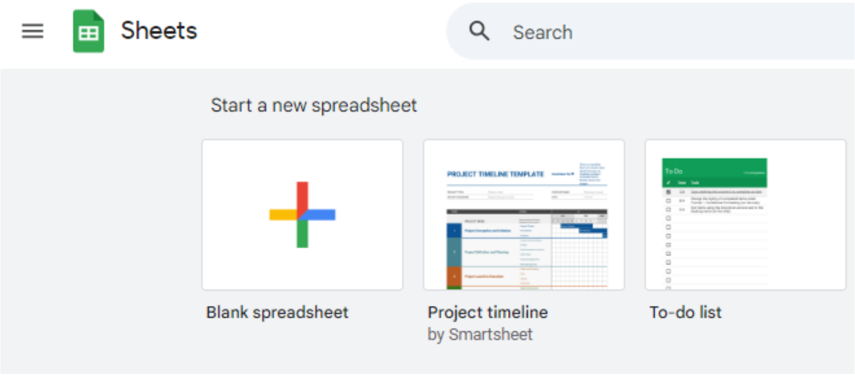
Detail Contents

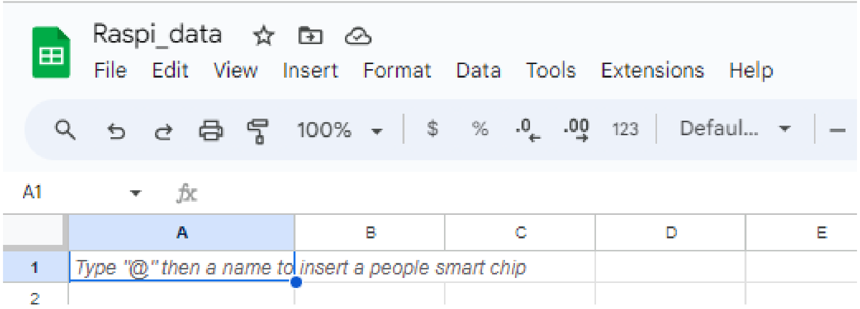
1 INTRODUCTION

In this section, we will prepare Raspberry Pi 5 to run the code of the DHT11 APP that has google sheets API and SQLite database functionality.

2 SETTING UP GOOGLE SHEETS API TO GET JSON CREDENTIALS FILE.

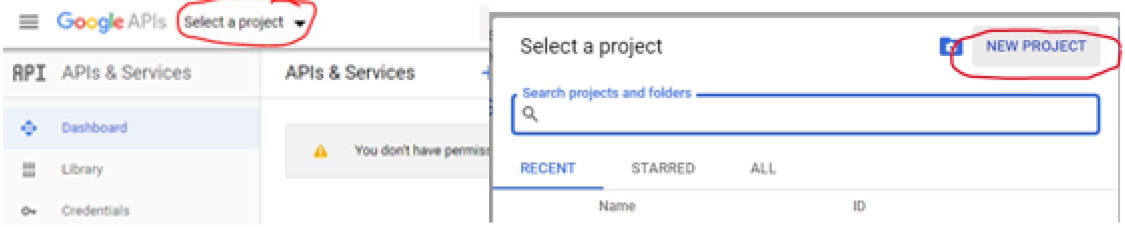
1) Create a new google sheet
    a. Go to Google Sheets website https://docs.google.com/spreadsheets/u/0/
    b. Create a new sheet. We will name the sheet 'Raspi_data'.





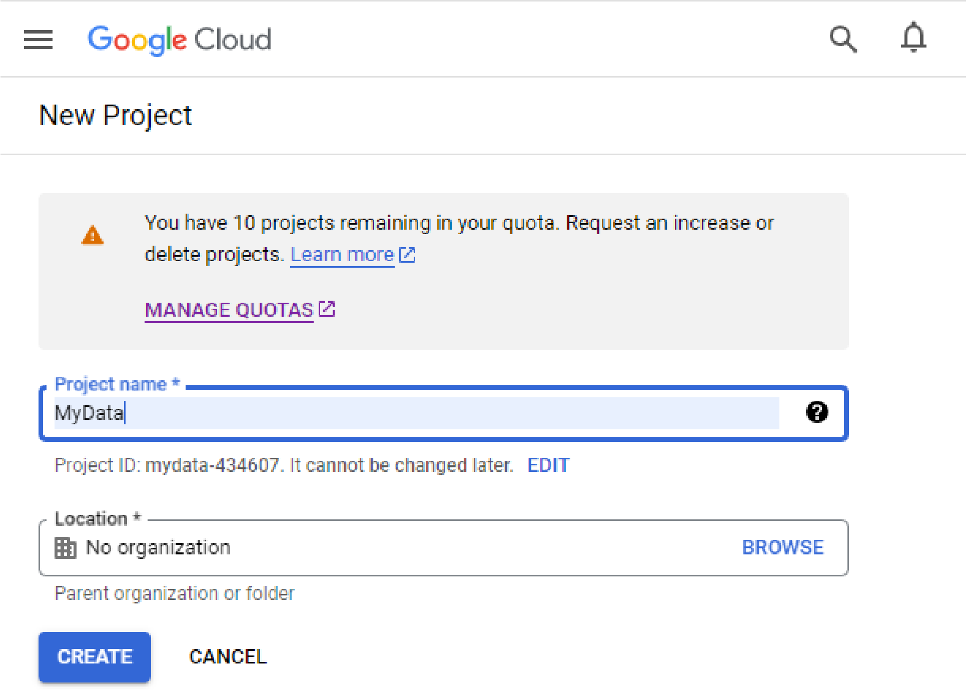
Type "@" then a name to insert a people smart chip

2) Create new Google Cloud project
    a. Open a new tab and go to the Google API console: http://console.developers.google.com
    b. You will be creating a new project.



.

c. In the New project window, give it the name ‘MyData’. (Notice that you have a project ID that is automatically generated. Write this project ID down. You don’t need to change the location.). Then, click ‘CREATE’. You will see something indicating your project is being created:



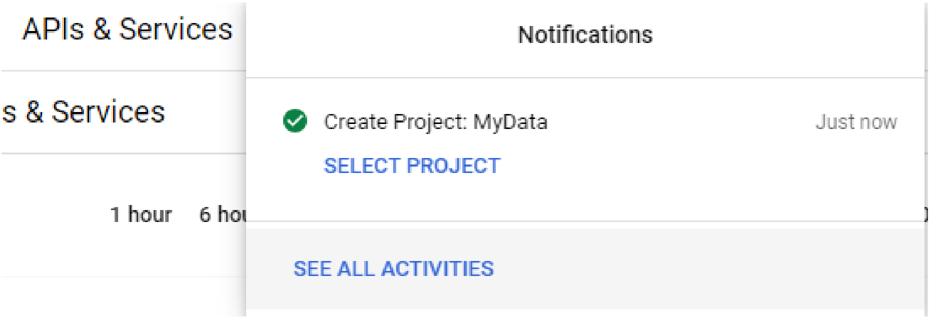
You have 10 projects remaining in your quota. Request an increase or delete projects. Learn more

Project name * MyData

Project ID: mydata-434607. It cannot be changed later. EDIT

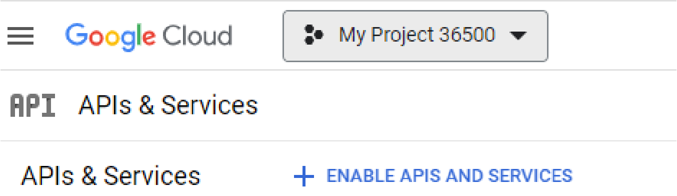
Location * No organization

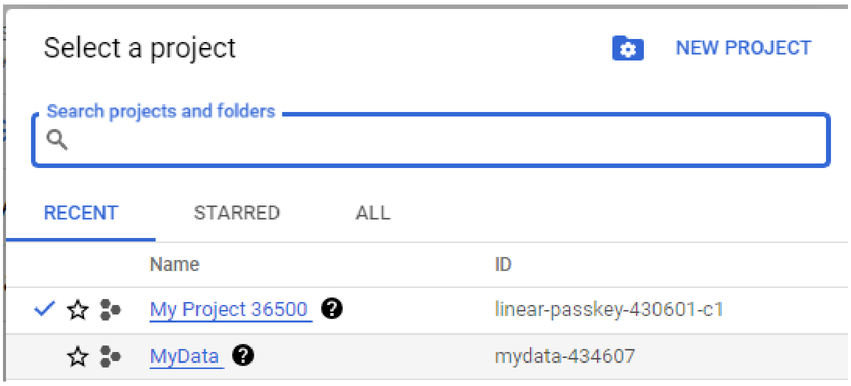
Parent organization or folder



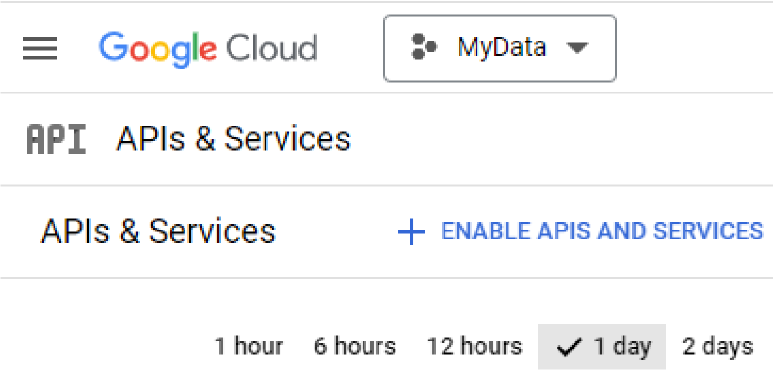
Create Project: MyData Just now

d. Next to the Google Cloud header, select myData from the dropdown menu to select this project.



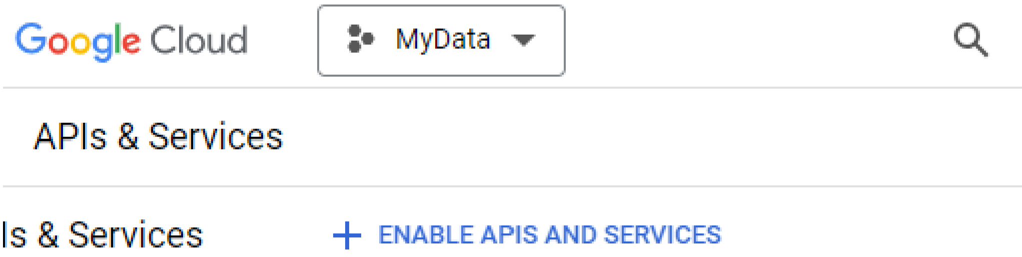


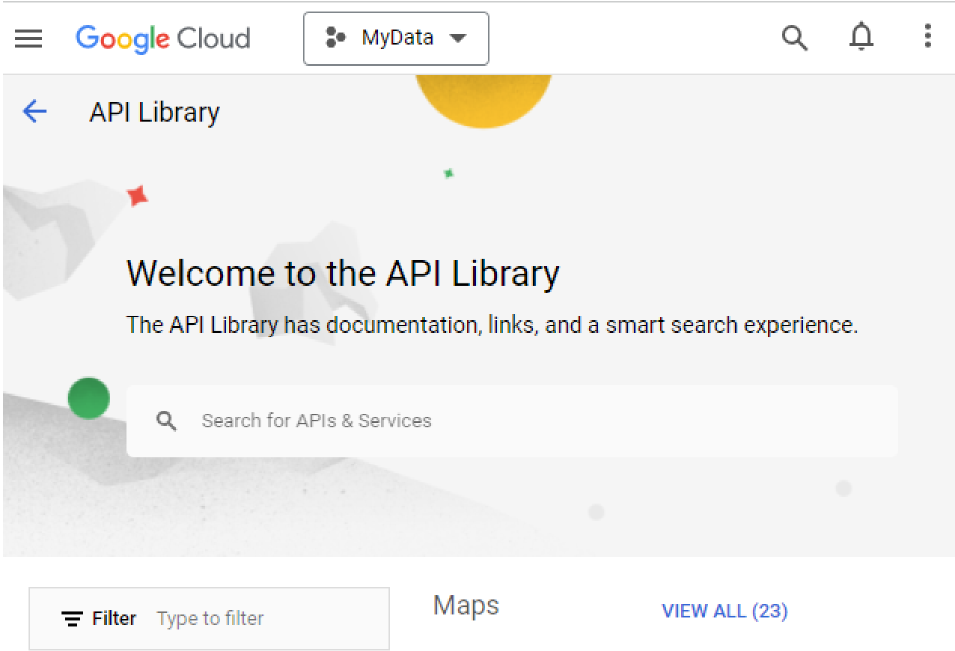
Name	ID
My Project 36500	linear-passkey-430601-c1
MyData	mydata-434607



3) Enable the Google Sheets API

a. Click on Enable APIS and SERVICES

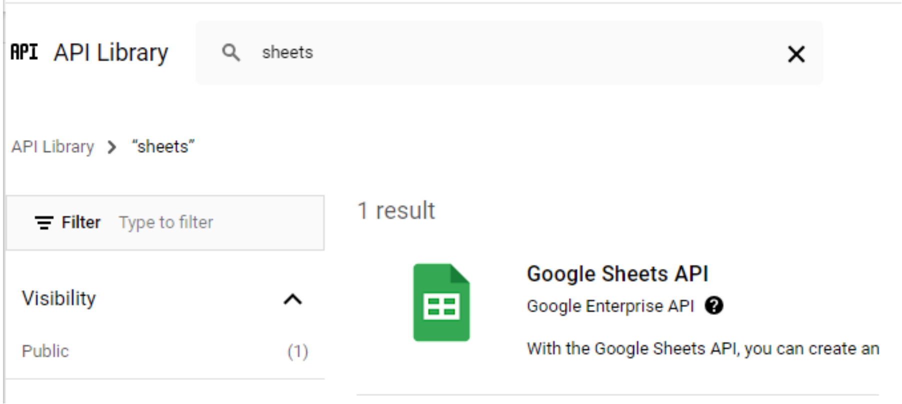




Welcome to the API Library

The API Library has documentation, links, and a smart search experience.

b. In the search window, type sheets and press enter, and you should see the following:



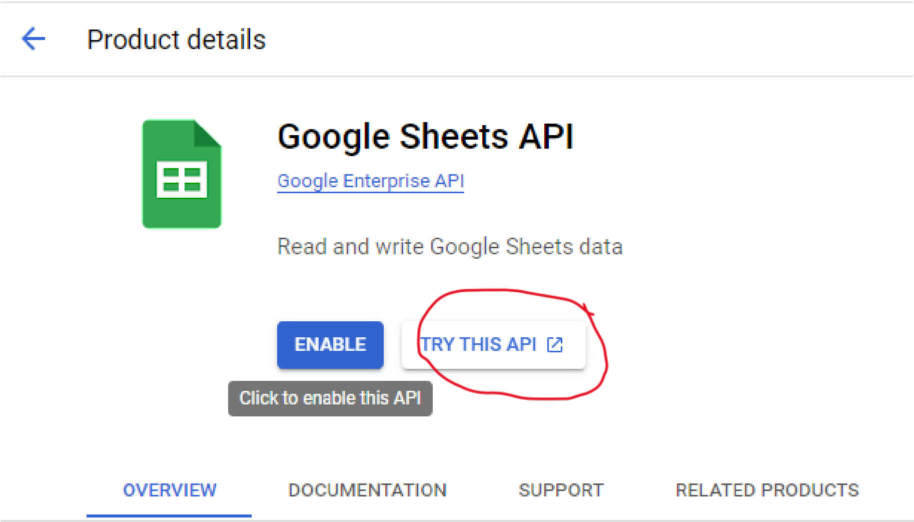
1 result

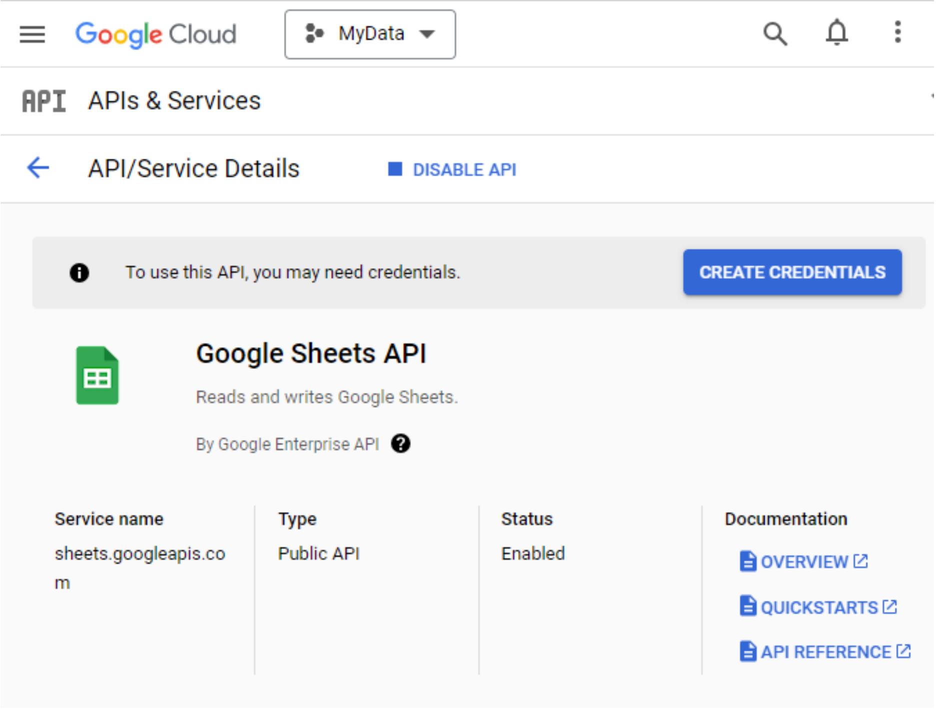
Google Sheets API

Google Enterprise API

With the Google Sheets API, you can create an

c. Click on the Google Sheets API that comes up as the result. Then click the ENABLE button.





To use this API, you may need credentials.

Google Sheets API

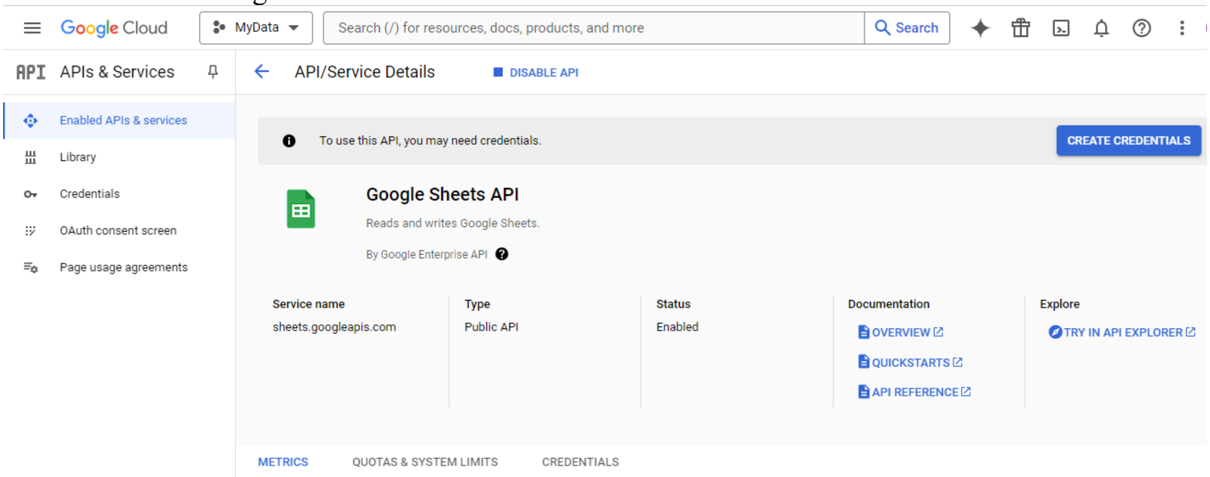
Reads and writes Google Sheets.

By Google Enterprise API

Service name	Type	Status	Documentation
sheets.googleapis.com	Public API	Enabled	OVERVIEW QUICKSTARTS API REFERENCE

4) Create a Service Account

a. Navigate to ‘API & Services > Credentials’.



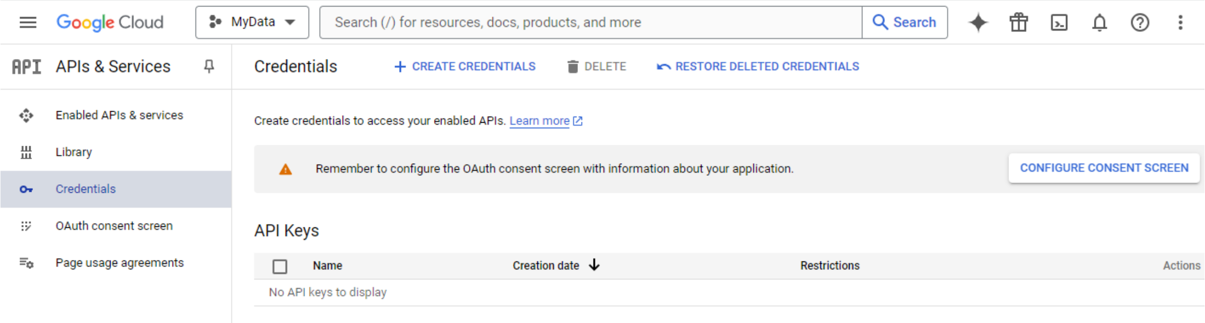
To use this API, you may need credentials.

Google Sheets API

Reads and writes Google Sheets.

By Google Enterprise API

Service name	Type	Status	Documentation	Explore
sheets.googleapis.com	Public API	Enabled	OVERVIEW QUICKSTARTS API REFERENCE	TRY IN API EXPLORER



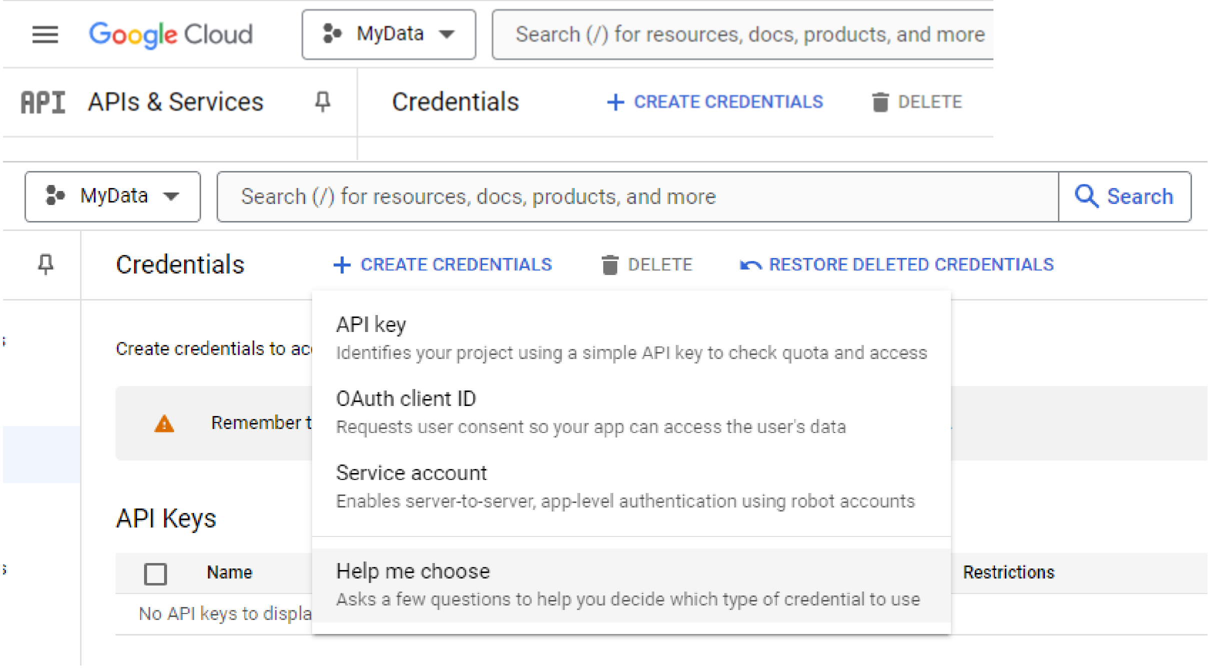
Create credentials to access your enabled APIs. Learn more

Remember to configure the OAuth consent screen with information about your application.

API Keys

□	Name	Creation date ↓	Restrictions	Actions
	No API keys to display			

b. Click on ‘create credentials’, then choose “Help me choose” option



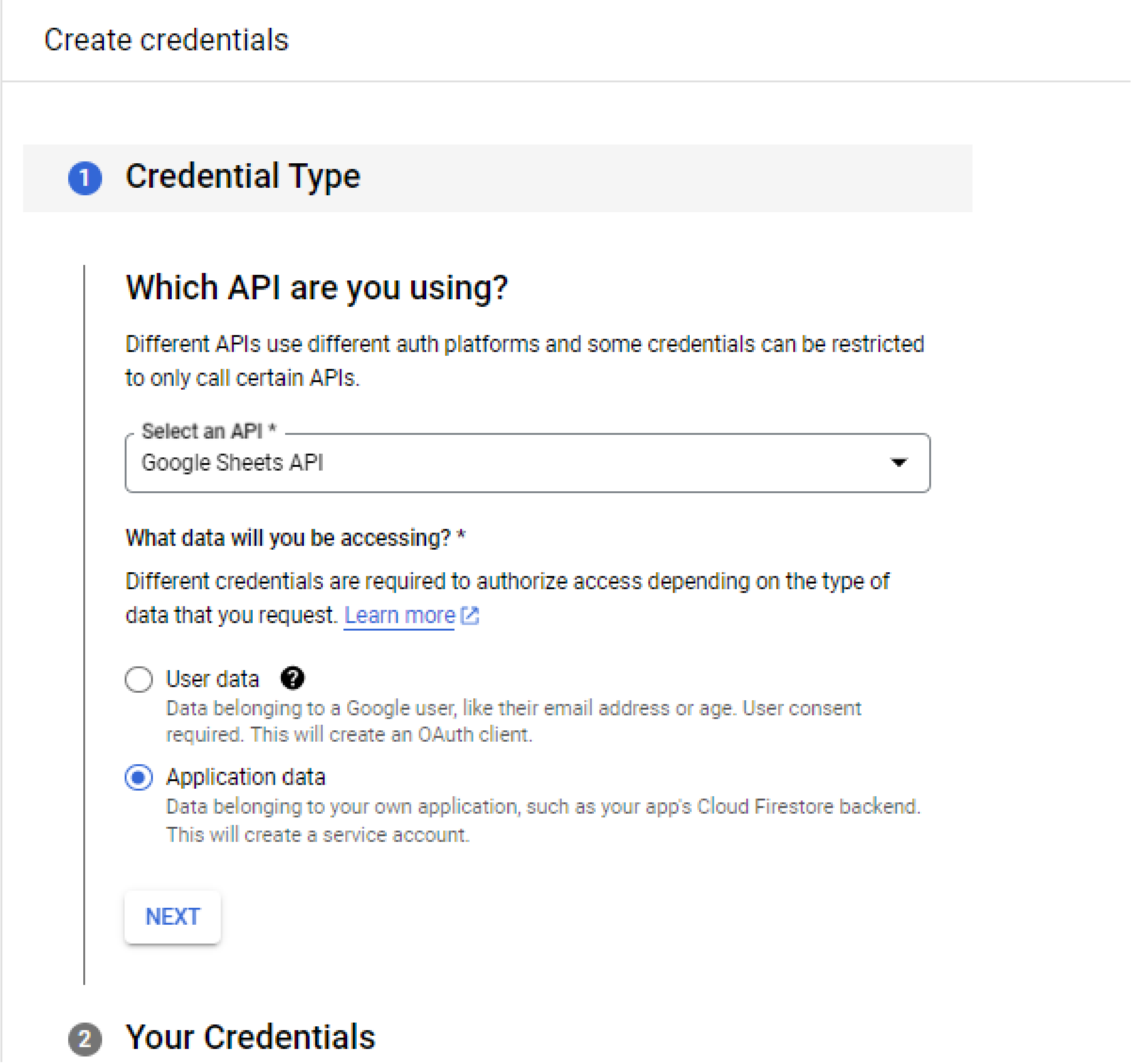
API key
Identifies your project using a simple API key to check quota and access

OAuth client ID
Requests user consent so your app can access the user's data

Service account
Enables server-to-server, app-level authentication using robot accounts

Help me choose
Asks a few questions to help you decide which type of credential to use

c. Fill in the required details and create the account.



Create credentials

1 Credential Type

Which API are you using?

Different APIs use different auth platforms and some credentials can be restricted to only call certain APIs.

Select an API *

Google Sheets API

What data will you be accessing? *

Different credentials are required to authorize access depending on the type of data that you request. Learn more

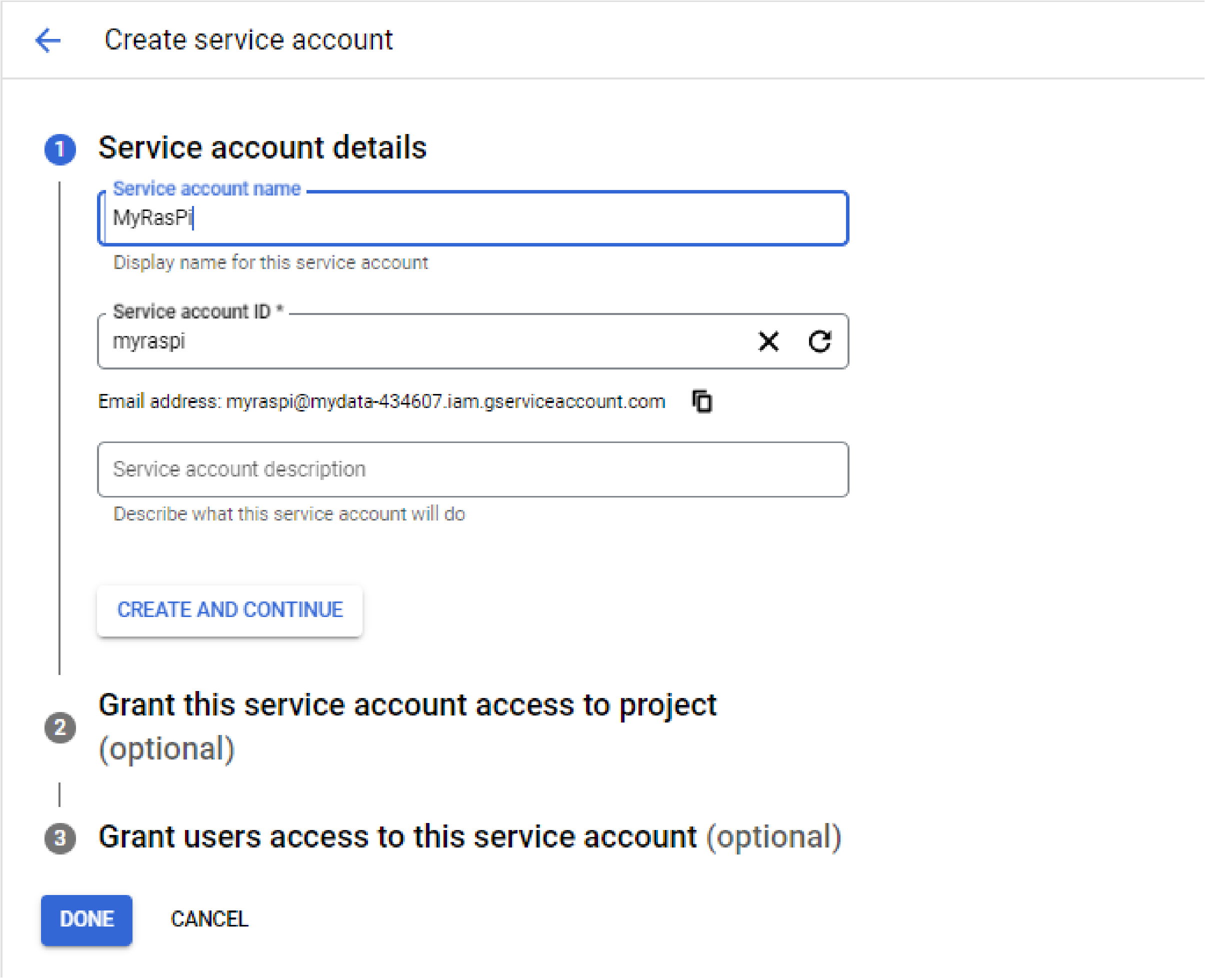
○ User data

Data belonging to a Google user, like their email address or age. User consent required. This will create an OAuth client.

● Application data

Data belonging to your own application, such as your app's Cloud Firestore backend. This will create a service account.

2 Your Credentials



Create service account

1 Service account details

Service account name: MyRasPi

Display name for this service account

Service account ID *: myraspi

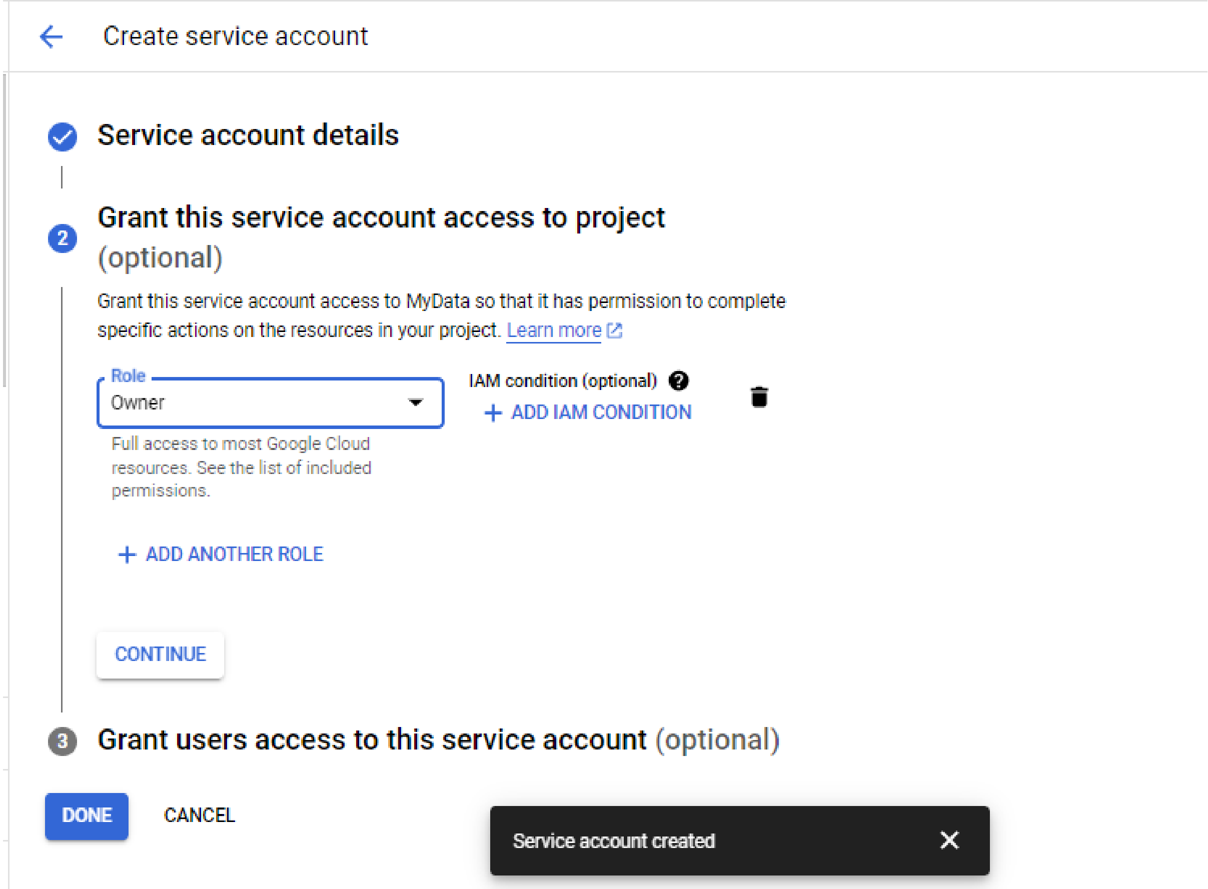
Email address: myraspi@mydata-434607.iam.gserviceaccount.com

Service account description

Describe what this service account will do

2 Grant this service account access to project (optional)

3 Grant users access to this service account (optional)



Create service account

✓ Service account details

2 Grant this service account access to project (optional)

Grant this service account access to MyData so that it has permission to complete specific actions on the resources in your project. Learn more

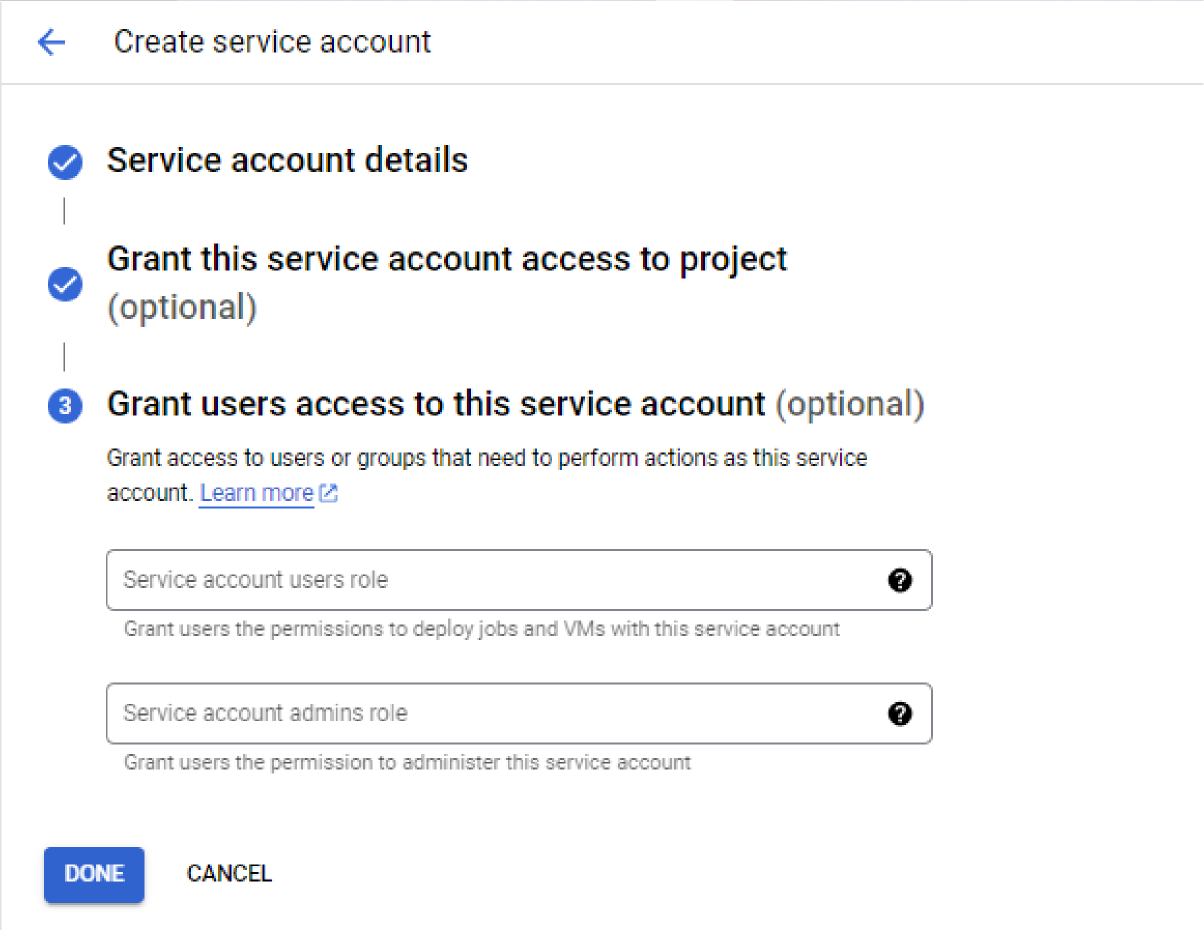
Role: Owner

Full access to most Google Cloud resources. See the list of included permissions.

IAM condition (optional)

3 Grant users access to this service account (optional)

Service account created ✕



Create service account

✓ Service account details

✓ Grant this service account access to project (optional)

3 Grant users access to this service account (optional)

Grant access to users or groups that need to perform actions as this service account. Learn more

Service account users role

Grant users the permissions to deploy jobs and VMs with this service account

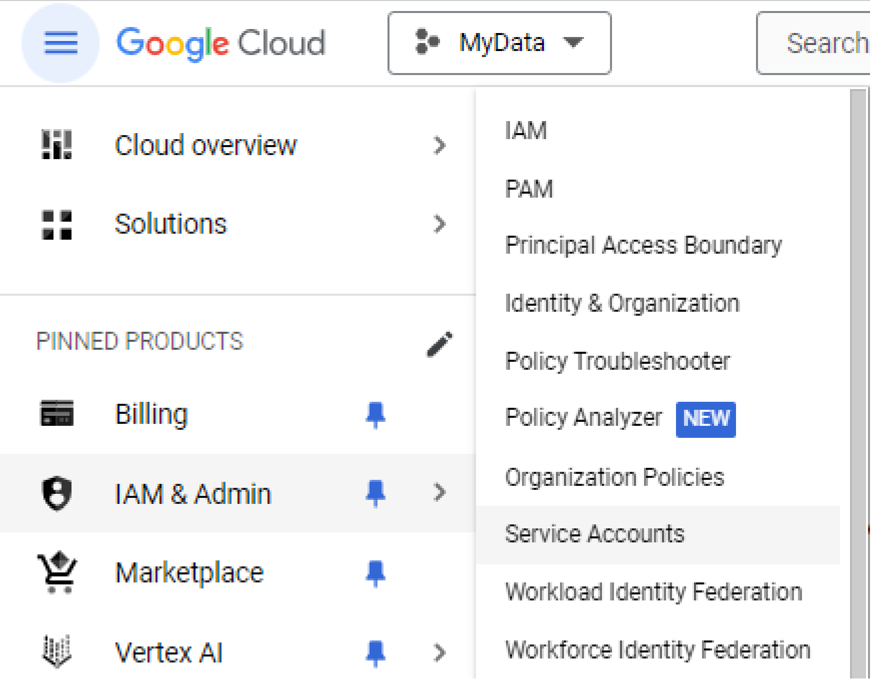
Service account admins role

Grant users the permission to administer this service account

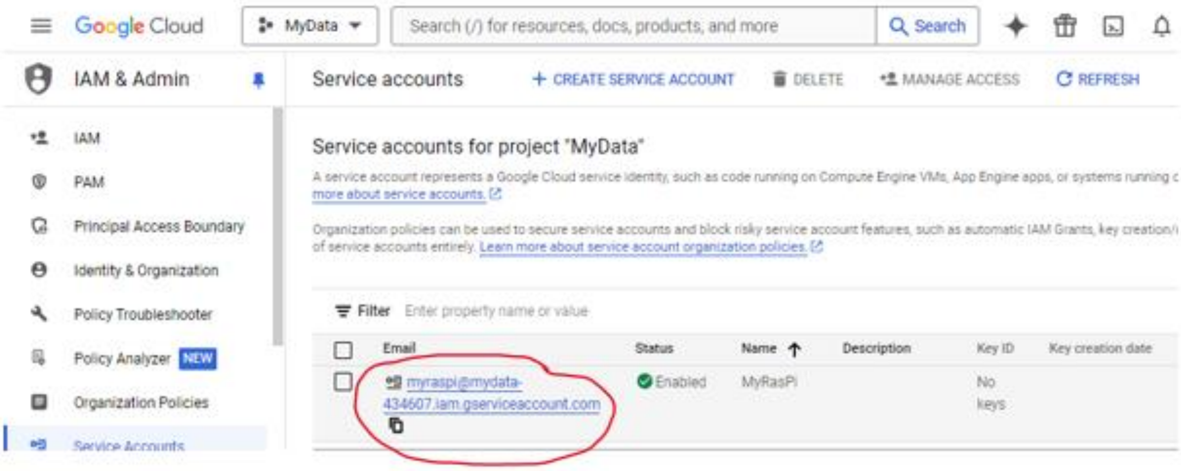
You now have set up the credentials for this account.

5) Get the JSON key

a. Navigate to ‘IAM & Admin > Service Accounts’



b. Click on the email of your service account.

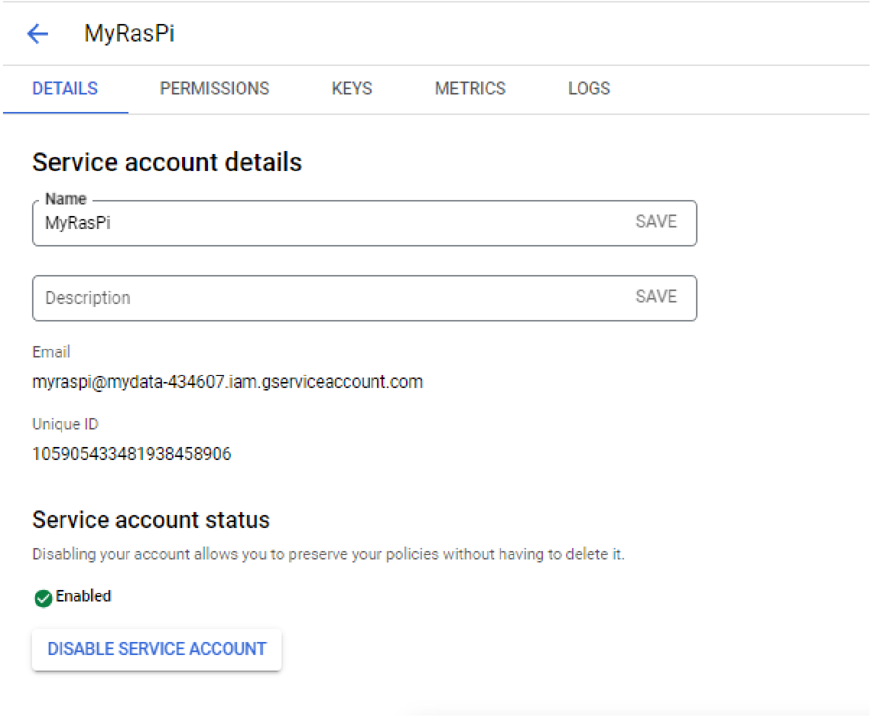


Service accounts for project "MyData"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running c
more about service accounts.

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/i
of service accounts entirely. Learn more about service account organization policies.

□	Email	Status	Name ↑	Description	Key ID	Key creation date
□	myraspi@mydata-434607.iam.gserviceaccount.com	✓ Enabled	MyRasPi		No keys	



Service account details

Name
MyRasPi

Email
myraspi@mydata-434607.iam.gserviceaccount.com

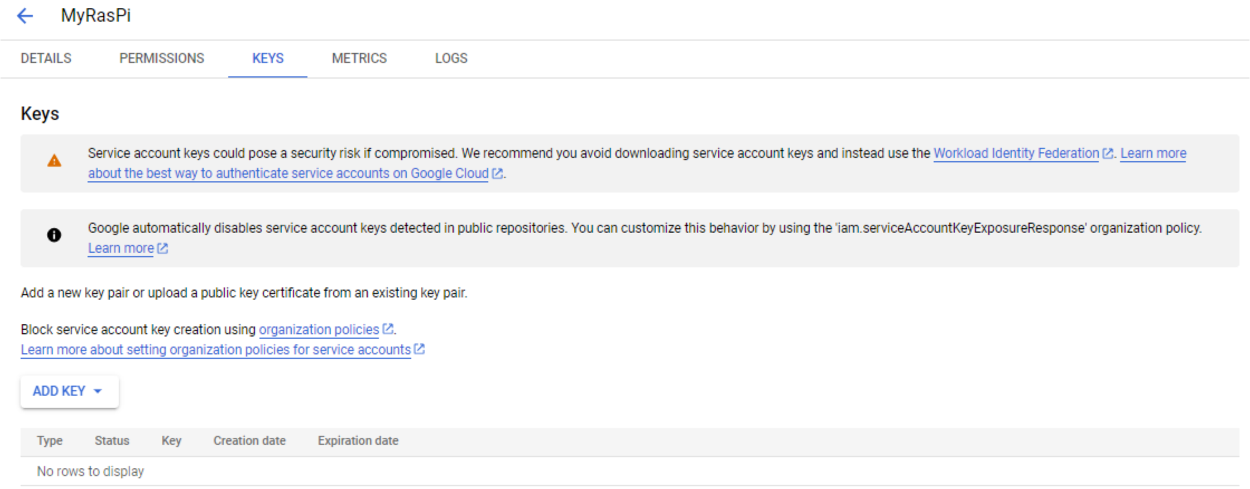
Unique ID
105905433481938458906

Service account status

Disabling your account allows you to preserve your policies without having to delete it.

✓ Enabled

c. Go to the “KEYS” tab.



Keys

⚠ Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the Workload Identity Federation. Learn more about the best way to authenticate service accounts on Google Cloud.

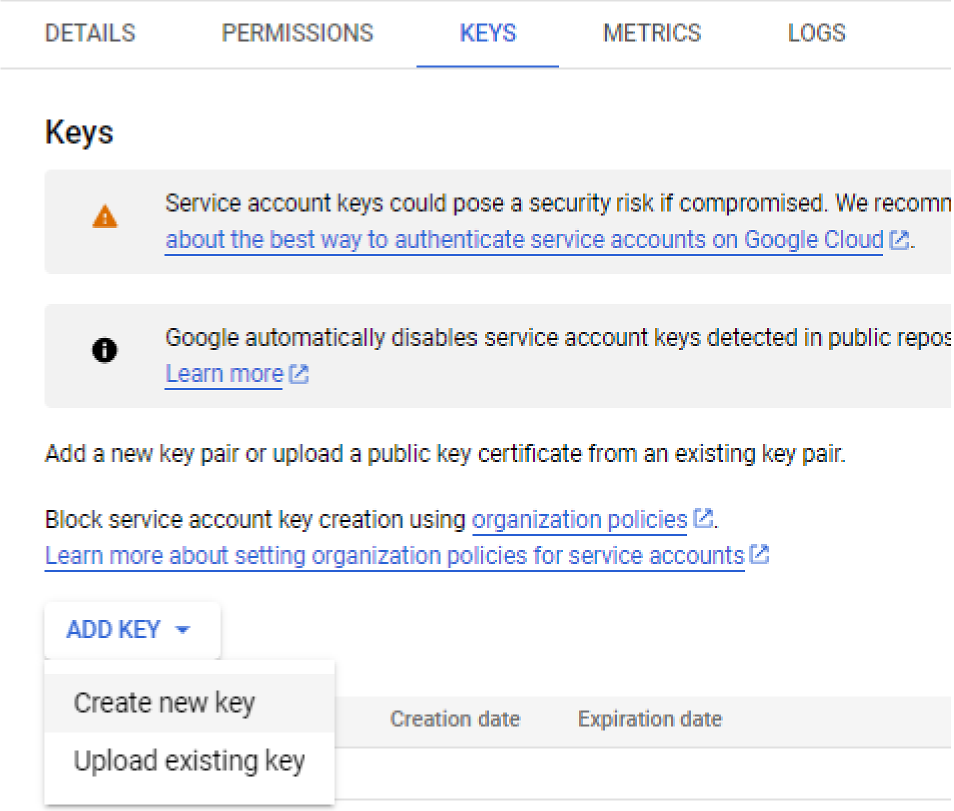
Google automatically disables service account keys detected in public repositories. You can customize this behavior by using the 'iam.serviceAccountKeyExposureResponse' organization policy. Learn more

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using organization policies.
Learn more about setting organization policies for service accounts

Type	Status	Key	Creation date	Expiration date
No rows to display				

d. Click on "ADD KEY", then "CREATE NEW KEY". After that, choose "JSON" key type.



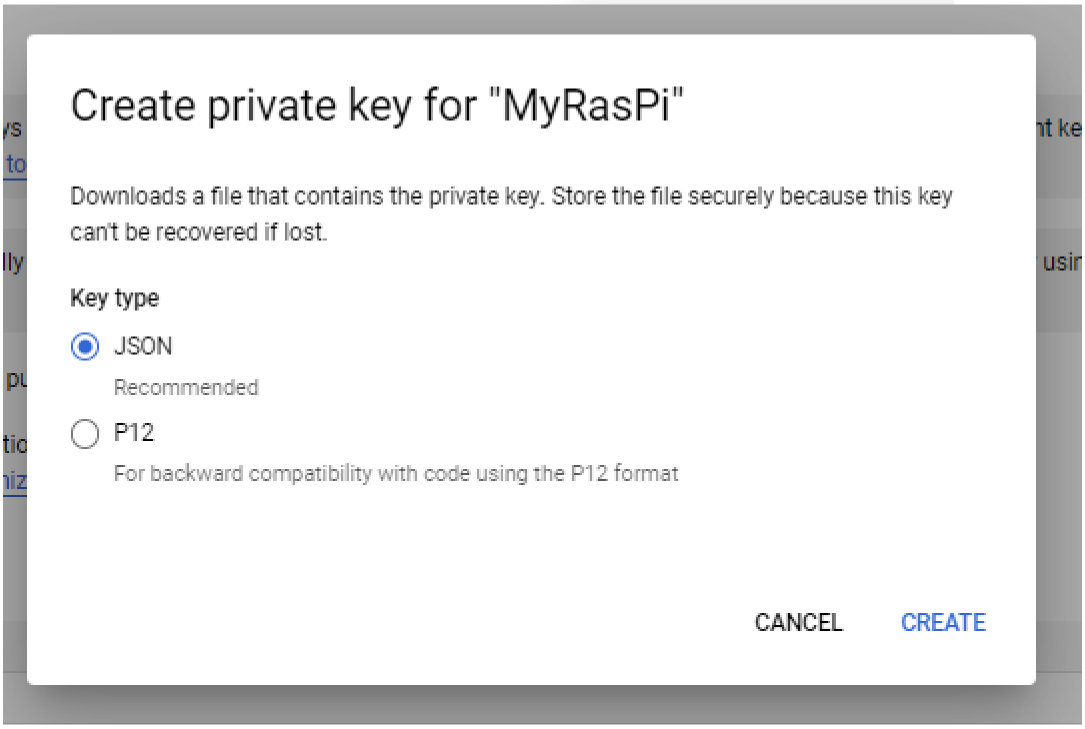
Keys

Service account keys could pose a security risk if compromised. We recomn
about the best way to authenticate service accounts on Google Cloud.

Google automatically disables service account keys detected in public repos
Learn more

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using organization policies.
Learn more about setting organization policies for service accounts



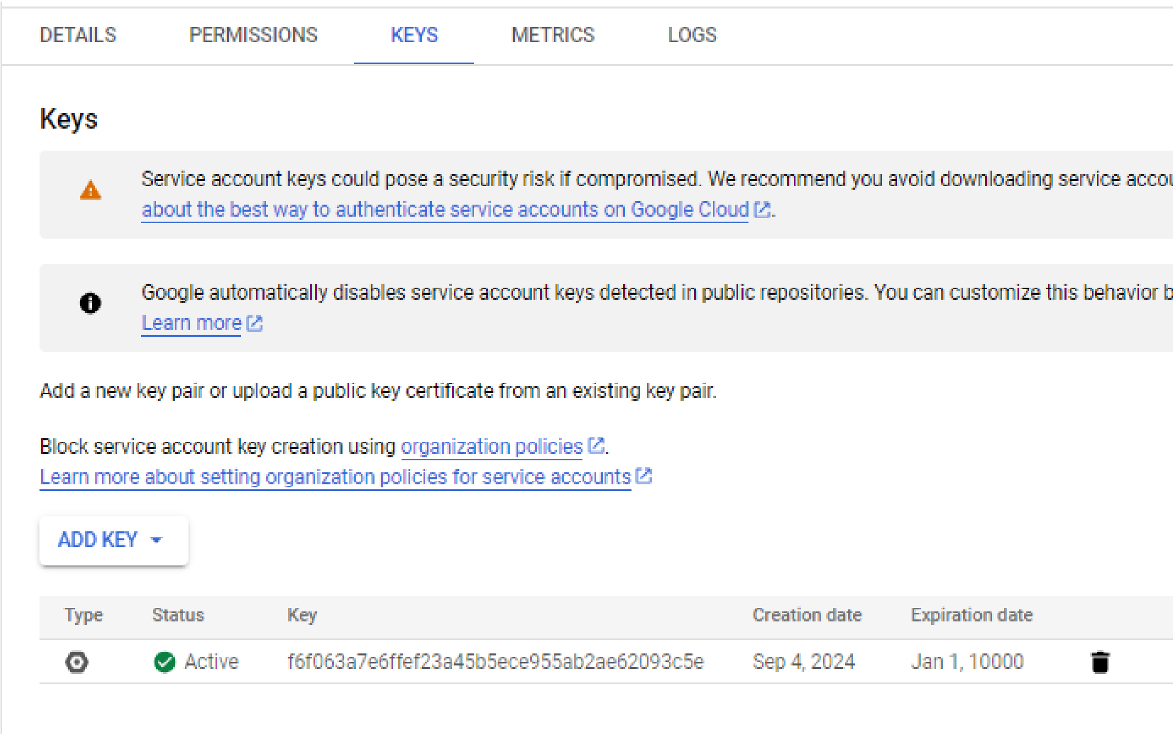
Create private key for "MyRasPi"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

● JSON
Recommended

○ P12
For backward compatibility with code using the P12 format



Keys

Service account keys could pose a security risk if compromised. We recommend you avoid downloading service accou
about the best way to authenticate service accounts on Google Cloud.

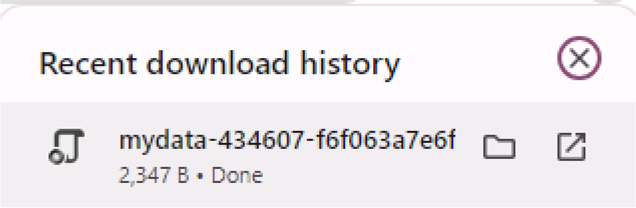
Google automatically disables service account keys detected in public repositories. You can customize this behavior b
Learn more

Add a new key pair or upload a public key certificate from an existing key pair.

Block service account key creation using organization policies.
Learn more about setting organization policies for service accounts

Type	Status	Key	Creation date	Expiration date
	Active	f6f063a7e6ffef23a45b5ece955ab2ae62093c5e	Sep 4, 2024	Jan 1, 10000

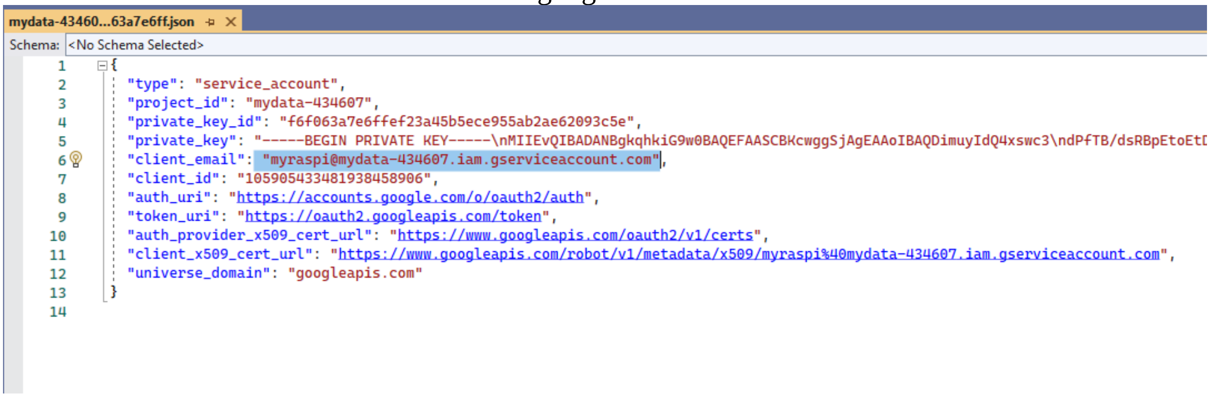
e. You will get an automatically downloaded JSON key that you need to use later. It will be in your downloaded files folder. **Do not lose this.**


Recent download history

mydata-434607-f6f063a7e6f
2,347 B • Done

6) Set up Google Sheet with Client Email from JSON

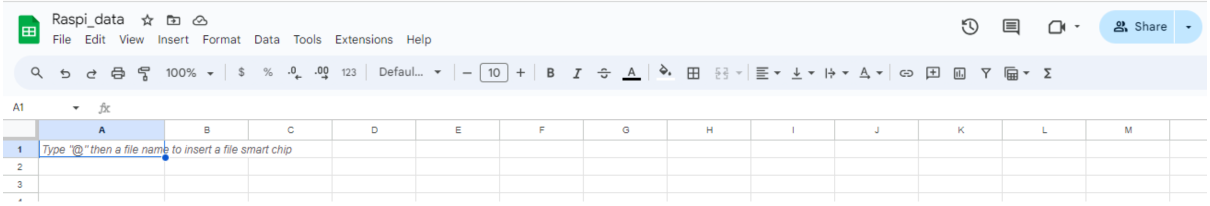
a. Using a text Editor, open the JSON downloaded file. You will need the email address called client_email.. Copy that email address into your clipboard (using Crtr-C) because we will need for the google sheet.



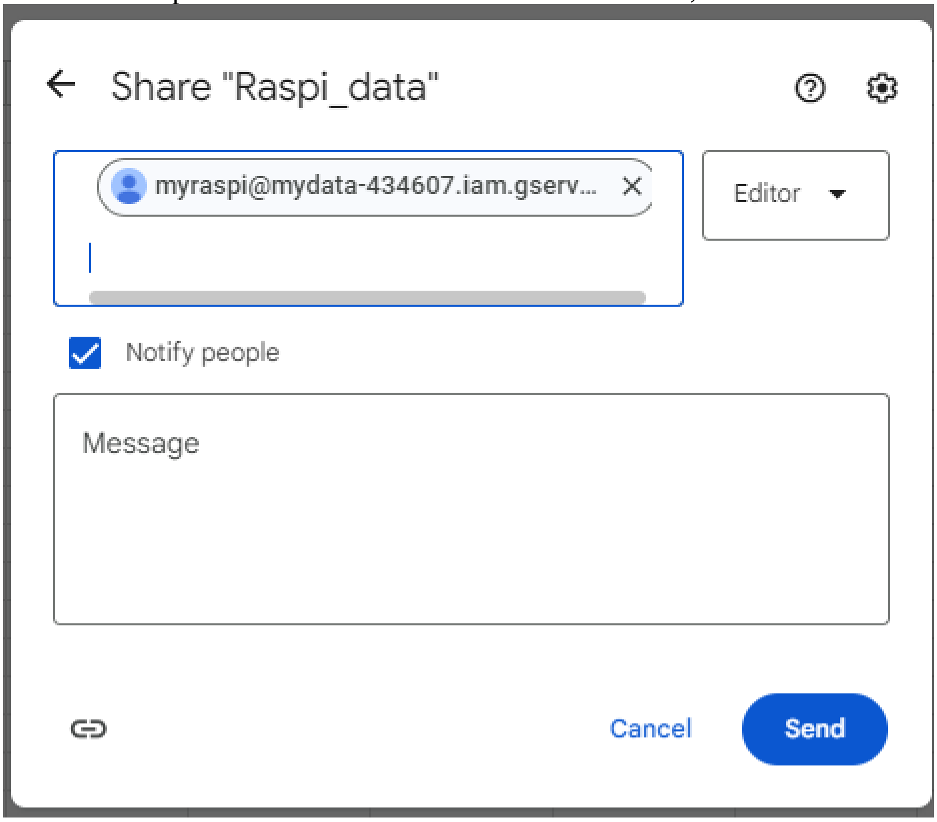
Schema: <No Schema Selected>

```
{
  "type": "service_account",
  "project_id": "mydata-434607",
  "private_key_id": "f6f063a7e6ffef23a45b5ece955ab2ae62093c5e",
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDimuyIdQ4xswc3\ndPfTB/dsRBpEtoEt
  "client_email": "myraspi@mydata-434607.iam.gserviceaccount.com",
  "client_id": "105905433481938458906",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/myraspi%40mydata-434607.iam.gserviceaccount.com",
  "universe_domain": "googleapis.com"
}
```

b. Go back to your Google sheet (Raspi_data) and click on file / share or click on the green share button in the upper right side of screen.


Raspi_data
File Edit View Insert Format Data Tools Extensions Help
Share
100%
Defaul...
10
A1
A B C D E F G H I J K L M
Type "@" then a file name to insert a file smart chip

c. You will then paste that email address into the share email, then click send.


Share "Raspi_data"

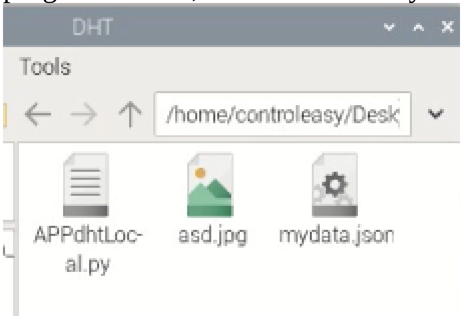
myraspi@mydata-434607.iam.gserv...

Editor

☑ Notify people

Message

7) Copy the downloaded JSON file to your Raspberry Pi. You will need to copy that JSON file into the directory where you write your python programs on your raspberry pi. You will need to have access to it from the python programs that you write that will access the google sheet. The file that I downloaded was called mydata-7ad604e1aaef.json, when I put it in my python programs folder, I renamed it to mydata.json so that I could easily reference it later.



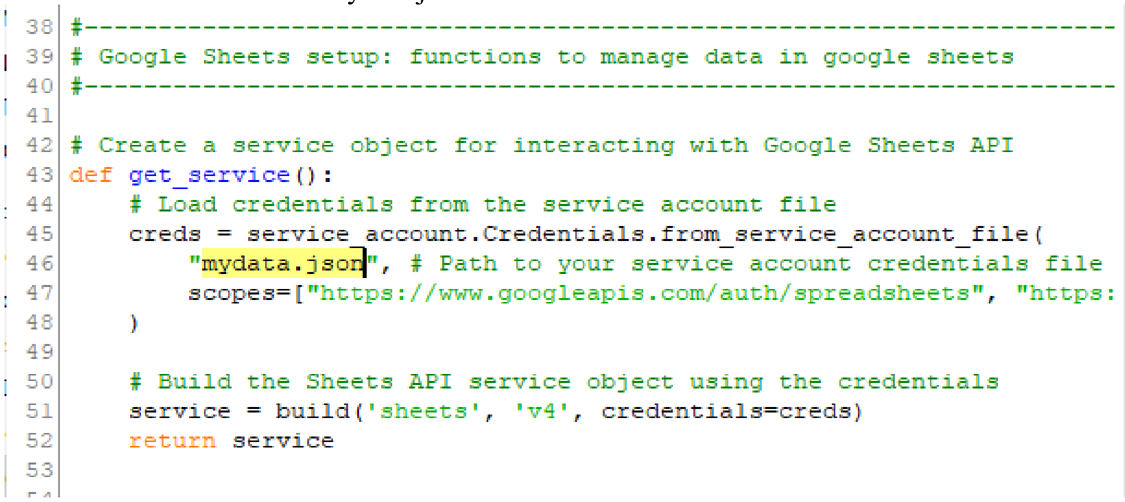
The project folder should have 3 things: the dht program (.py), the company logo for the app (asd.jpg), and the credentials file (.json)

3 EDIT THE PYTHON CODE

Now, we must insert some information into the application code.

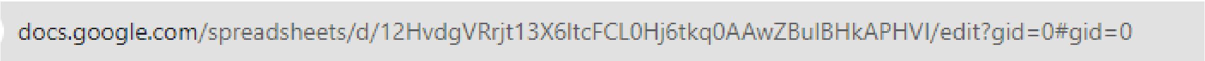
1) Open the application code “APPdhtLocal.py”

2) Search the upper part of the code for “get_service” function. Make sure to name your downloaded JSON file as “mydata.json” as in this code.



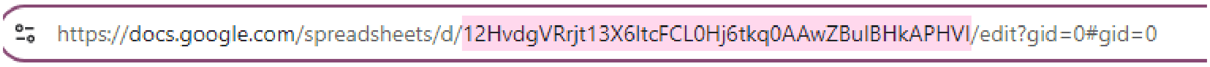
```
#-----------------------------------------------------------------------
# Google Sheets setup: functions to manage data in google sheets
#-----------------------------------------------------------------------

# Create a service object for interacting with Google Sheets API
def get_service():
    # Load credentials from the service account file
    creds = service_account.Credentials.from_service_account_file(
        "mydata.json", # Path to your service account credentials file
        scopes=["https://www.googleapis.com/auth/spreadsheets", "https:
    )

    # Build the Sheets API service object using the credentials
    service = build('sheets', 'v4', credentials=creds)
    return service

```

3) Go your Google sheet (Raspi_data). Then, look at the url.



docs.google.com/spreadsheets/d/12HvdgVRrjt13X6ltcFCL0Hj6tkq0AAwZBulBHkAPHVI/edit?gid=0#gid=0

4) Get your spreadsheet id. The spreadsheet id is written after “/d/” . Copy it.

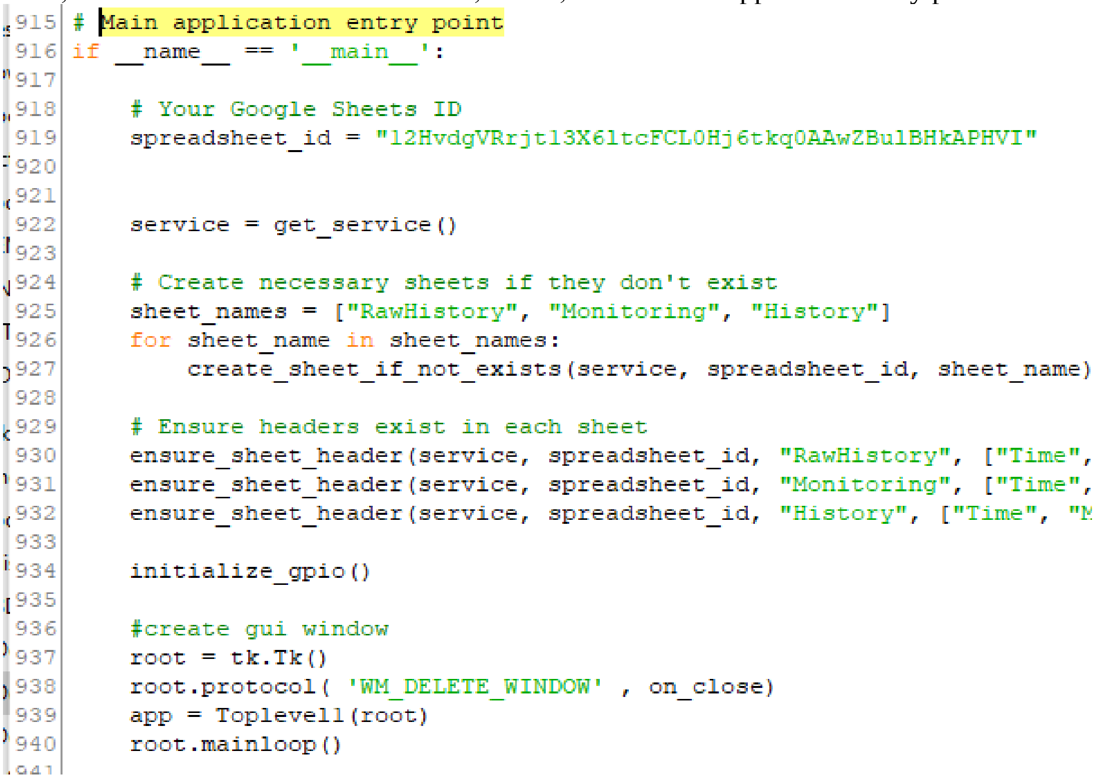


https://docs.google.com/spreadsheets/d/12HvdgVRrjt13X6ltcFCL0Hj6tkq0AAwZBulBHkAPHVI/edit?gid=0#gid=0

In this case, the spreadsheet id is:

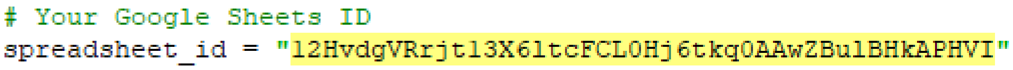
“12HvdgVRrjt13X6ltcFCL0Hj6tkq0AAwZBulBHkAPHVI”.

5) Next, scroll down to the end of the code, that is, to the “main application entry point”.



```
# Main application entry point
if __name__ == '__main__':

    # Your Google Sheets ID
    spreadsheet_id = "12HvdgVRrjt13X6ltcFCL0Hj6tkq0AAwZBulBHkAPHVI"


    service = get_service()

    # Create necessary sheets if they don't exist
    sheet_names = ["RawHistory", "Monitoring", "History"]
    for sheet_name in sheet_names:
        create_sheet_if_not_exists(service, spreadsheet_id, sheet_name)

    # Ensure headers exist in each sheet
    ensure_sheet_header(service, spreadsheet_id, "RawHistory", ["Time",
    ensure_sheet_header(service, spreadsheet_id, "Monitoring", ["Time",
    ensure_sheet_header(service, spreadsheet_id, "History", ["Time", "M

    initialize_gpio()

    #create gui window
    root = tk.Tk()
    root.protocol( 'WM_DELETE_WINDOW' , on_close)
    app = Toplevel1(root)
    root.mainloop()
```

6) Change the value of “spreadsheet_id” to your spreadsheet_id.



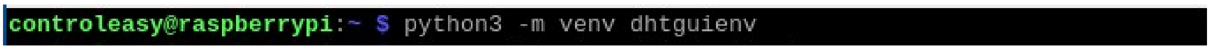
```
# Your Google Sheets ID
spreadsheet_id = "12HvdgVRrjt13X6ltcFCL0Hj6tkq0AAwZBulBHkAPHVI"
```

7) Now, you can save the changes of the code.

4 ENVIRONMENT FOR PYTHON CODE

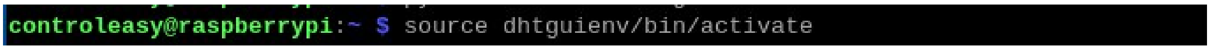
Now, we will install additional libraries that are needed for GUI, Google Sheets API, and SQLite.

1) Create a virtual environment. In this case,we named the virtual environment as “dhtguienv”



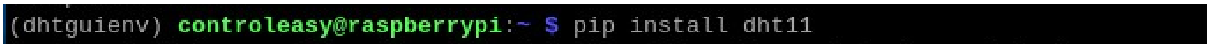
```
controleasy@raspberrypi:~ $ python3 -m venv dhtguienv
```

2) Activate the virtual environment.



```
controleasy@raspberrypi:~ $ source dhtguienv/bin/activate
```

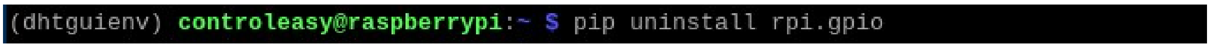
3) Install dht11 library



```
(dhtguienv) controleasy@raspberrypi:~ $ pip install dht11
```

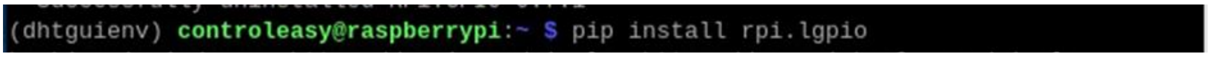
4) Install rpi.lgpio library

   a. Uninstall rpi.gpio library



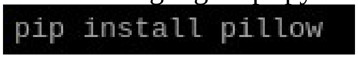
```
(dhtguienv) controleasy@raspberrypi:~ $ pip uninstall rpi.gpio
```

   b. Install rpi.lgpio library

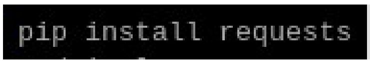


```
(dhtguienv) controleasy@raspberrypi:~ $ pip install rpi.lgpio
```

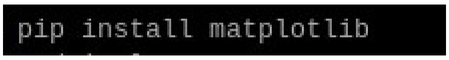
5) Install pillow, requests, matplotlib, google-auth, google-auth-oauthlib, google-auth-httplib2, and google-api-python-client inside the virtual environment.



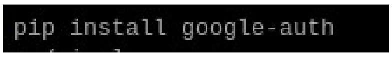
```
pip install pillow
```



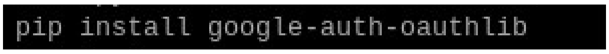
```
pip install requests
```



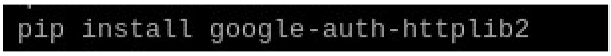
```
pip install matplotlib
```



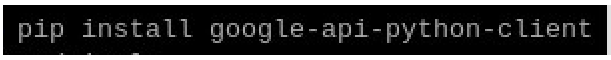
```
pip install google-auth
```



```
pip install google-auth-oauthlib
```

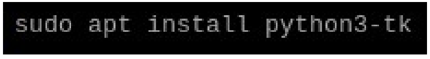


```
pip install google-auth-httplib2
```



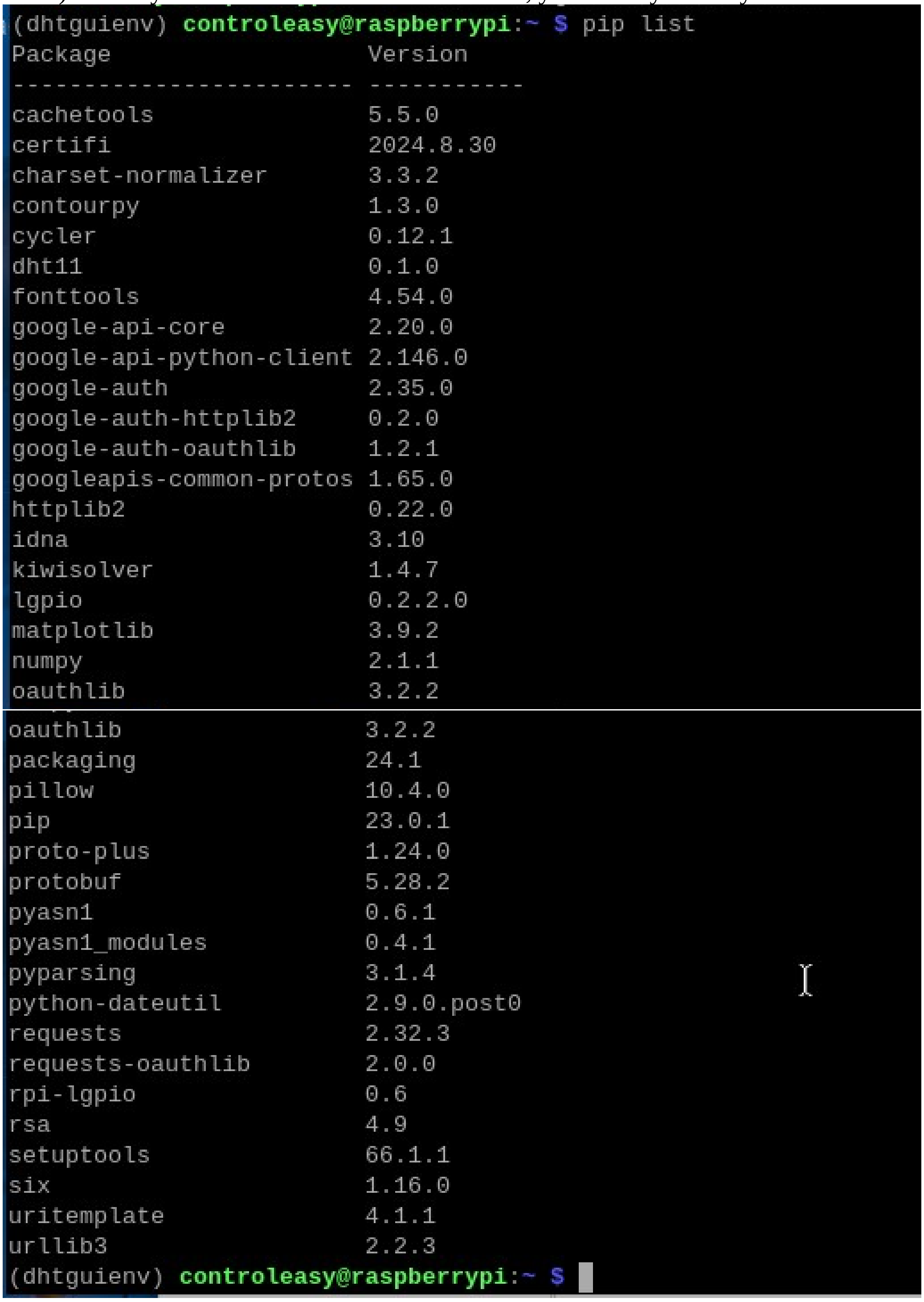
```
pip install google-api-python-client
```

6) Install python3-tk



```
sudo apt install python3-tk
```

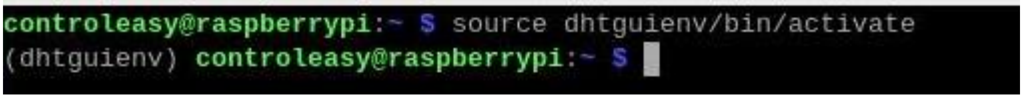
7) After you have installed all these libraries, your library list may look as follows:



```
(dhtguienv) controleasy@raspberrypi:~ $ pip list
Package                   Version
------------------------- -----------
cachetools                5.5.0
certifi                   2024.8.30
charset-normalizer        3.3.2
contourpy                 1.3.0
cycler                    0.12.1
dht11                     0.1.0
fonttools                 4.54.0
google-api-core           2.20.0
google-api-python-client  2.146.0
google-auth               2.35.0
google-auth-httplib2      0.2.0
google-auth-oauthlib      1.2.1
googleapis-common-protos  1.65.0
httplib2                  0.22.0
idna                      3.10
kiwisolver                1.4.7
lgpio                     0.2.2.0
matplotlib                3.9.2
numpy                     2.1.1
oauthlib                  3.2.2
oauthlib                  3.2.2
packaging                 24.1
pillow                    10.4.0
pip                       23.0.1
proto-plus                1.24.0
protobuf                  5.28.2
pyasn1                    0.6.1
pyasn1_modules            0.4.1
pyparsing                 3.1.4
python-dateutil           2.9.0.post0
requests                  2.32.3
requests-oauthlib         2.0.0
rpi-lgpio                 0.6
rsa                       4.9
setuptools                66.1.1
six                       1.16.0
uritemplate               4.1.1
urllib3                   2.2.3
(dhtguienv) controleasy@raspberrypi:~ $
```

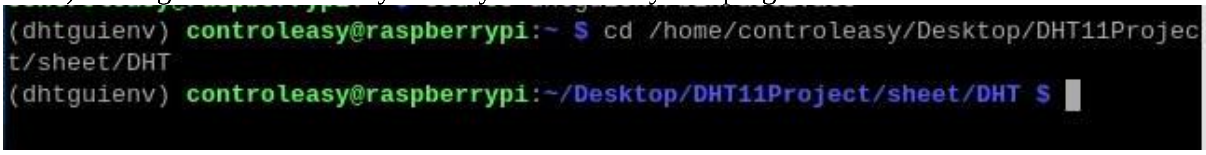
5 RUNNING THE PROGRAM

1) Open your virtual environment



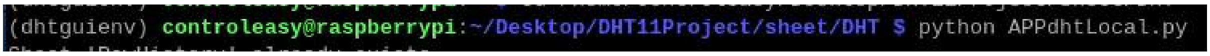
```
controleasy@raspberrypi:~ $ source dhtguienv/bin/activate
(dhtguienv) controleasy@raspberrypi:~ $ 
```

2) Navigate to the directory where you save the Python program

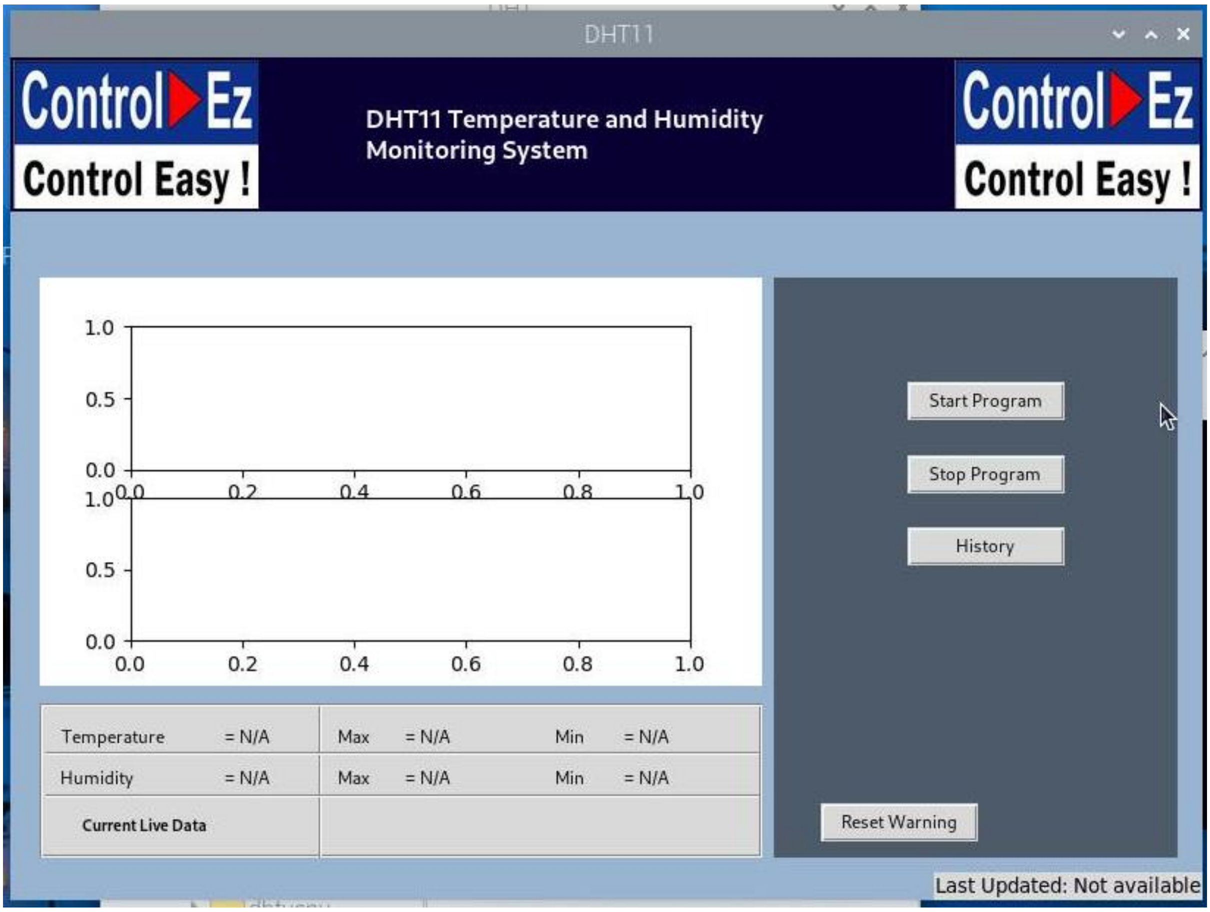


```
(dhtguienv) controleasy@raspberrypi:~ $ cd /home/controleasy/Desktop/DHT11Projec
t/sheet/DHT
(dhtguienv) controleasy@raspberrypi:~/Desktop/DHT11Project/sheet/DHT $ 
```

3) Run the program



```
(dhtguienv) controleasy@raspberrypi:~/Desktop/DHT11Project/sheet/DHT $ python APPdhtLocal.py
```



Temperature = N/A	Max = N/A	Min = N/A	
Humidity = N/A	Max = N/A	Min = N/A	
Current Live Data			

Last Updated: Not available