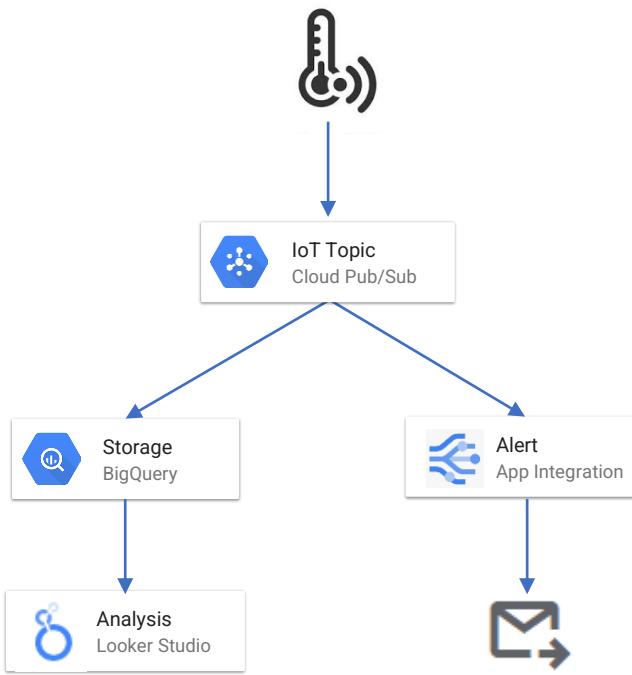


IoT & The Cloud
Free Friday Workshop
February 9, 2024

IoT & The Cloud



DOWNLOAD THE SLIDES HERE TO FOLLOW ALONG:

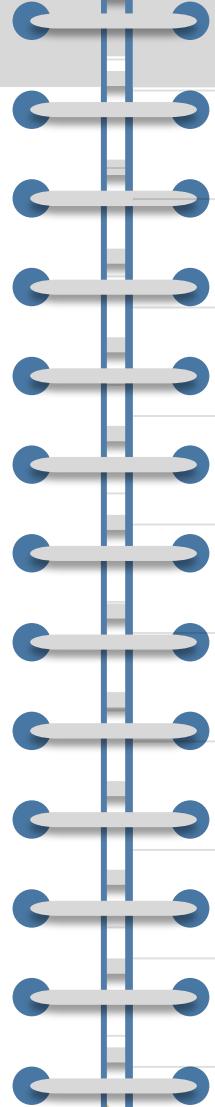
<http://tinyurl.com/48khunrk>

AGENDA

- Welcome & Intros
- Goals for the Day
- IoT Overview
- Raspberry Pi Hardware Setup
- Google Cloud Platform (GCP)
- Ingesting Telemetry Data
- Storing the Data
- Alerting & Reporting on the Data
- Futures



CHECKPOINT



Logistics Discussion

Everyone on the same
page?

Help each other out!



Links exist throughout
the deck to dive more
deeply into some of the
topics we cover

*Great for followup after
the workshop!*

Internet of Things Overview



Number of humans? – **8.1B**



Number of cell phones? – **7B**



Number of IoT devices? – **24.4 B**

Examples

Consumer – locks, doorbells, lightbulbs, cars

Commercial – healthcare, monitoring, security

Industrial – digital control sys, agribiz

Infrastructure – smart cities



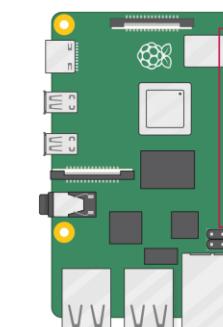
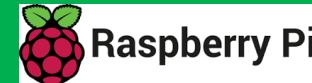
Challenges:

1. Updating & Managing Devices
2. Managing Telemetry
3. Managing Data

Raspberry Pi Overview

Pi 3 Model B (circa 2016)

- Single Board Computer (SBC)
- 1.2 GHz 64-bit quad core ARM Cortex-A53 processor
- 1 GB RAM
- HDMI output
- 802.11n Wi-Fi & Bluetooth
- 8G SD card w/64-bit Debian (bookworm)
- GPIO – General Purpose I/O



3V3 power	5V power
GPIO 2 (SDA)	Ground
GPIO 3 (SCL)	GPIO 14 (TXD)
GPIO 4 (PCCLK0)	GPIO 15 (RXD)
Ground	Ground
GPIO 17	GPIO 18 (PCM_CLK)
GPIO 27	Ground
GPIO 22	GPIO 23
3V3 power	GPIO 24
GPIO 10 (MOSI)	Ground
GPIO 9 (MISO)	GPIO 25
GPIO 11 (SCLK)	GPIO 8 (CEO)
Ground	GPIO 7 (CE1)
GPIO 0 (ID_SD)	GPIO 1 (ID_SC)
GPIO 5	Ground
GPIO 6	GPIO 12 (PWM0)
GPIO 13 (PWM1)	Ground
GPIO 19 (PCM_FS)	GPIO 16
GPIO 26	GPIO 20 (PCM_DIN)
Ground	GPIO 21 (PCM_DOUT)

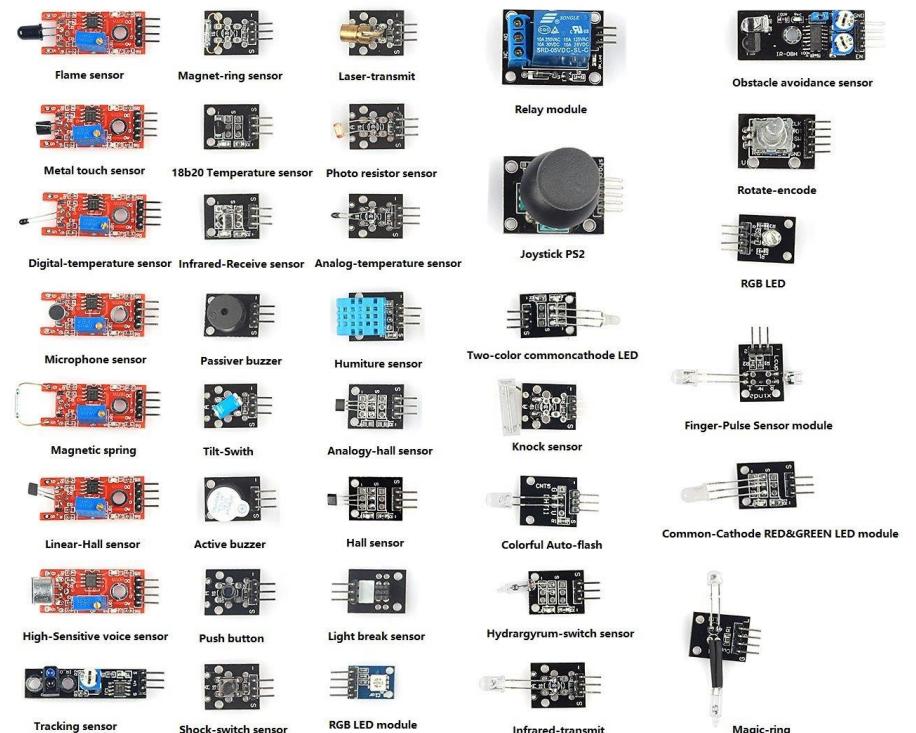
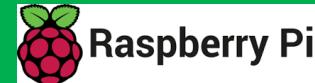
I WANT TO
KNOW MORE
History of Raspberry Pi

I WANT TO
KNOW MORE
Raspberry Pi Model Comparison

Raspberry Pi Overview

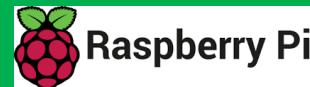
*Many popular IoT type sensors can
be connected to GPIO:*

- Temp
- Humidity
- Light
- Fire
- Sound
- Motion
- IR
- Joystick, etc.

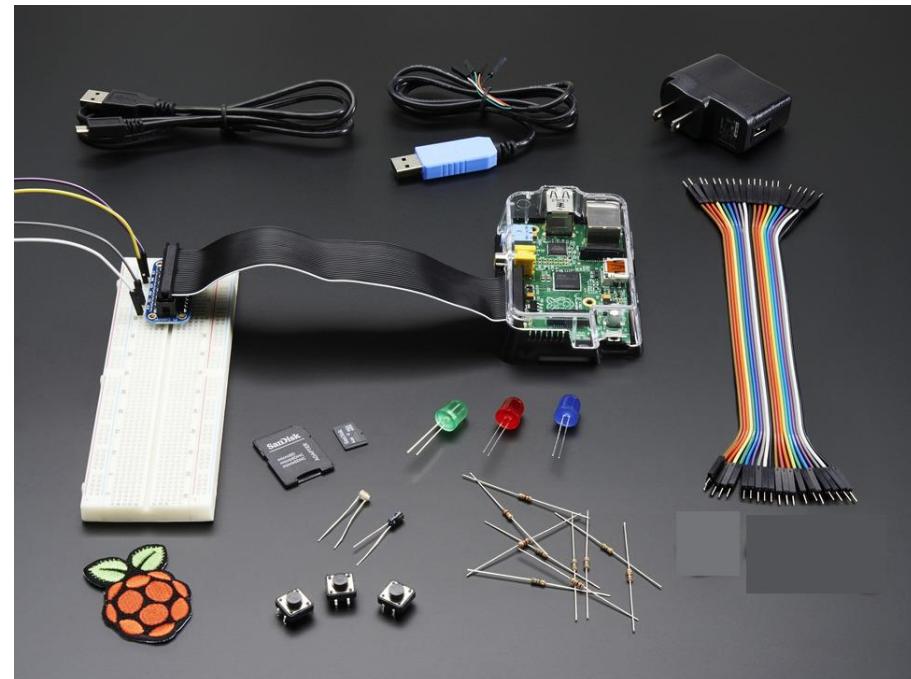


Raspberry Pi Setup

- Raspberry Pi 3 Model B
- SD Card with Raspbian OS
- Pi Case
- 5v Power Supply
- Breadboard and Jumper wires
- Leds & Resistors
- GPIO breakout ribbon cable
- USB jumper (not used)
- Raspberry Pi patch
- DS18B20 Temp Sensor



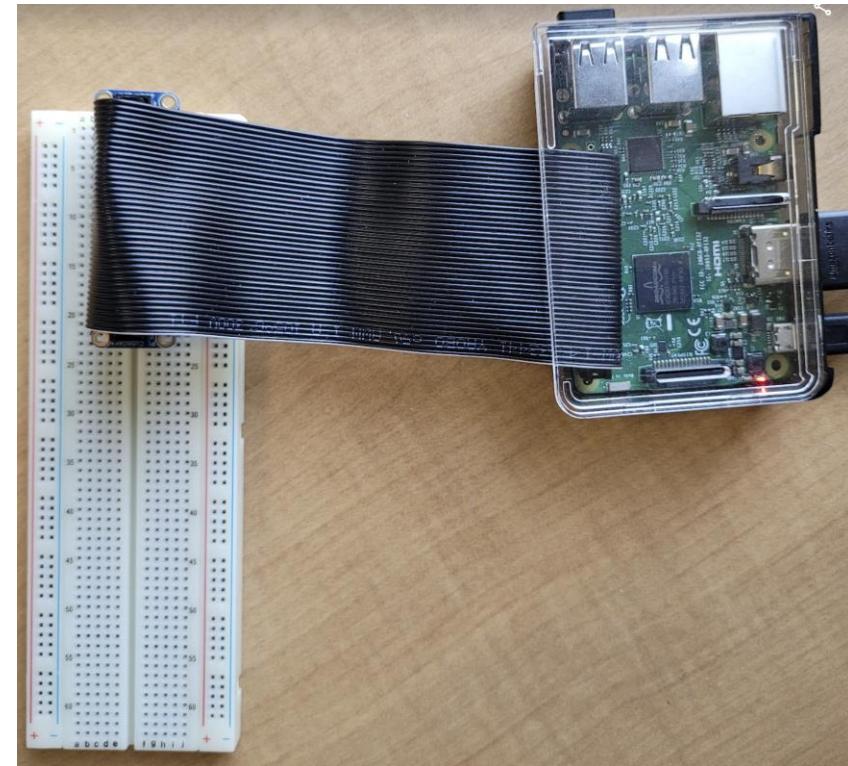
What's in the box??



Raspberry Pi Setup

Cable it up:

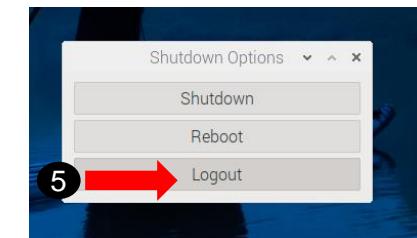
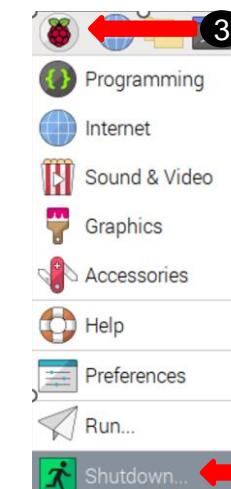
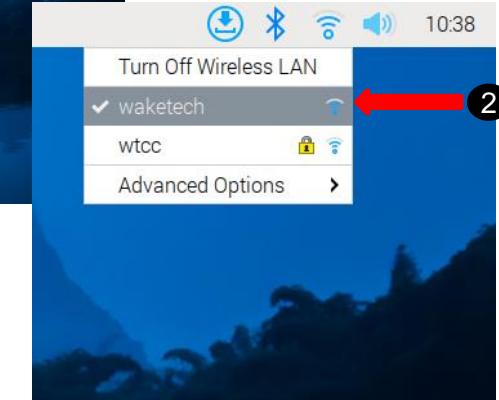
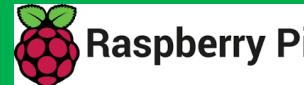
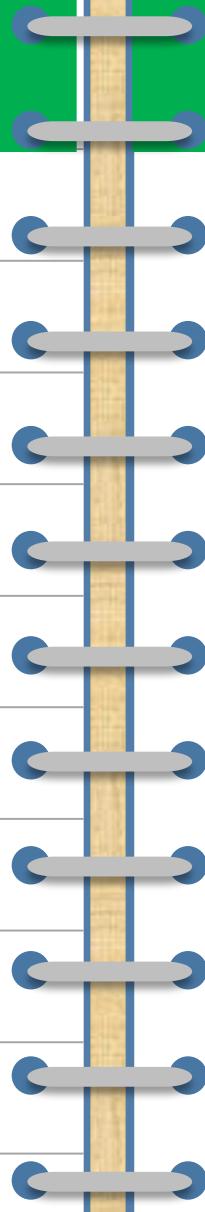
1. Insert pi board into case
2. Insert SD Card
3. Connect HDMI monitor
4. Connect Keyboard
5. Connect Mouse
6. Cable GPIO to Breadboard
7. 5v Power supply (***do this last!***)



Raspberry Pi Setup

Initial Startup – Be patient – it's a bit slow!

1. Check that your machine has joined the wireless network (waketech)
 - Click the wireless icon in upper right
2. Test the logoff / logon process
 - Click the raspberry icon in upper left & select the 'Shutdown' menu choice
 - username = pi
 - password = raspberry





CHECKPOINT

Raspberry Pi rebooted,
connected to the network and
showing the desktop



Raspberry Pi Setup

Config tweak needed for our project:

Launch terminal and run this command

```
sudo nano /boot/config.txt
```

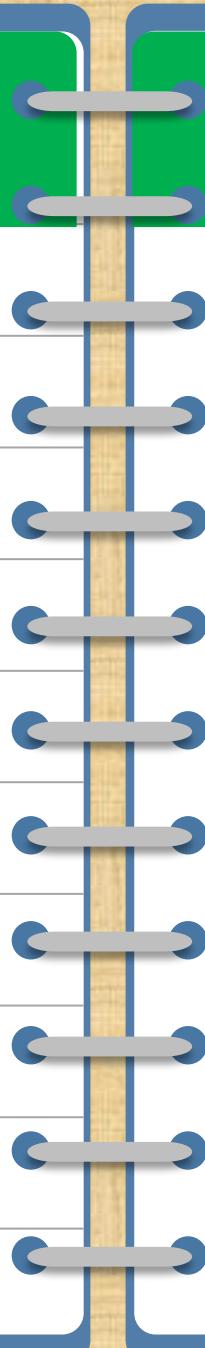
Then add these lines at bottom:

```
# Add dtoverlay setting for thermensor on GPIO
```

```
dtoverlay=w1-gpio,gpiopin=4
```

Type CTRL+O to write then CTRL+X to exit

Then reboot (pi menu – choose Shutdown)



Raspberry Pi®

```
File Edit Tabs Help  
pi@raspberrypi:~$ sudo nano /boot/config.txt  
  
File Edit Tabs Help  
GNU nano 7.2  
# Enable audio (loads snd_bcm2835)  
dtoverlay=audio=on  
  
# Additional overlays and parameters are documented  
# /boot/firmware/overlays/README  
  
# Automatically load overlays for detected cameras  
camera_auto_detect=1  
  
# Automatically load overlays for detected DSI displays  
display_auto_detect=1  
  
# Automatically load initramfs files, if found  
auto_initramfs=1  
  
# Enable DRM VC4 V3D driver  
dtoverlay=vc4-kms-v3d  
max_framebuffers=2  
  
# Don't have the firmware create an initial video= setting in cmdline.txt.  
# Use the kernel's default instead.  
disable_fw_kms_setup=1  
  
# Run in 64-bit mode  
arm_64bit=1  
  
# Disable compensation for displays with overscan  
disable_overscan=1  
  
# Run as fast as firmware / board allows  
arm_boost=1  
  
# Add dtoverlay setting for thermensor on GPIO  
dtoverlay=w1-gpio,gpiopin=4  
[cm4]  
# Enable host mode on the 2711 built-in XHCI USB controller.  
# This line should be removed if the legacy DWC2 controller is required  
# (e.g. for USB device mode) or if USB support is not required.  
otg_mode=1  
  
[all]  
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location ^U Undo ^A Set Mark  
^X Exit ^R Read File ^W Replace ^P Paste ^J Justify ^V Go To Line ^L Redo ^M-Copy
```

Raspberry Pi Code

1. Launch terminal and run this command:

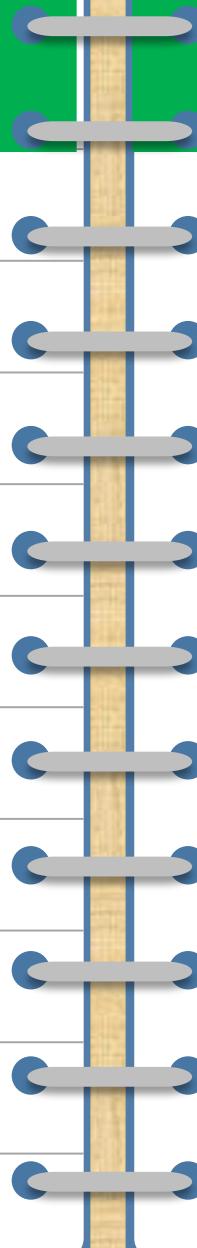
```
git clone http://github.com/jkbanham/iot-cloud-workshop.git
```

2. Note that you should now have a new directory with files we use for this project:

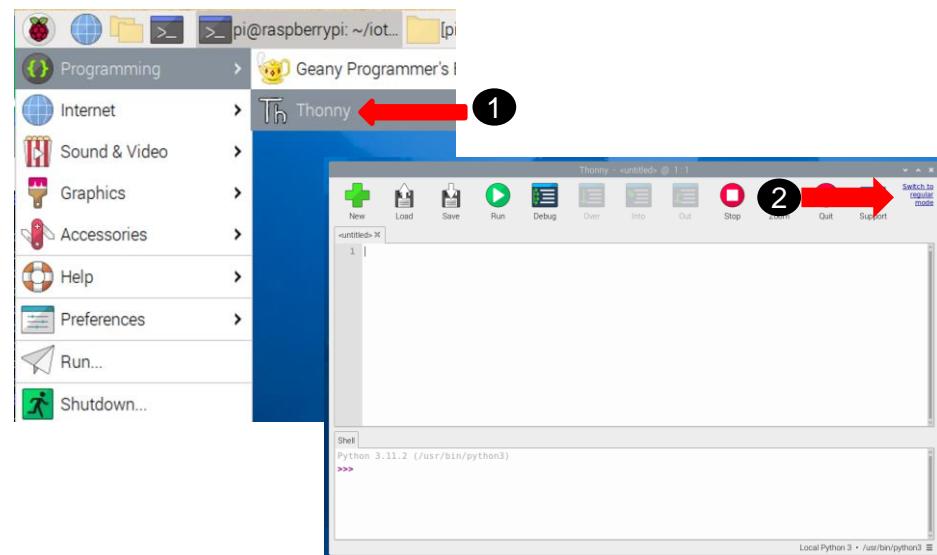
```
/home/pi/iot-cloud-workshop/
```

3. Launch the Thonny IDE from the Pi menu under Programming. Click on the 'Switch to regular mode' link in the upper right

4. Close Thonny and then re-launch it



```
pi@raspberrypi:~$ ls -l
total 36
drwxr-xr-x 2 pi pi 4096 Dec 4 23:47 Bookshelf
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Desktop
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Documents
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Downloads
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Music
drwxr-xr-x 2 pi pi 4096 Feb 5 10:46 Pictures
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Public
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Templates
drwxr-xr-x 2 pi pi 4096 Dec 5 00:06 Videos
pi@raspberrypi:~$ git clone http://github.com/jkbanham/iot-cloud-workshop.git
Cloning into 'iot-cloud-workshop'...
Warning: redirecting to https://github.com/jkbanham/iot-cloud-workshop.git/
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 13 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (13/13), 6.08 MiB | 1.66 MiB/s, done.
Resolving deltas: 100% (2/2), done.
pi@raspberrypi:~$
```



Raspberry Pi Code

1. From the ‘Run’ menu, select ‘Configure Interpreter’

and then click the ‘New virtual environment’ link in

the lower right

2. In the dialog box to “Select empty directory for new

virtual environment”, browse to the /home/pi/iot-

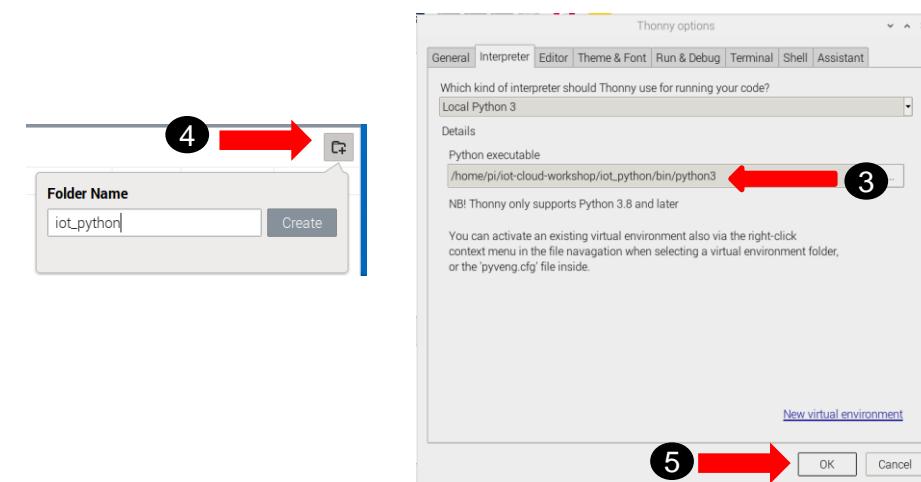
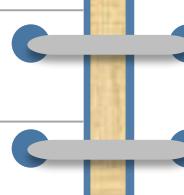
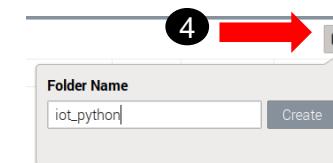
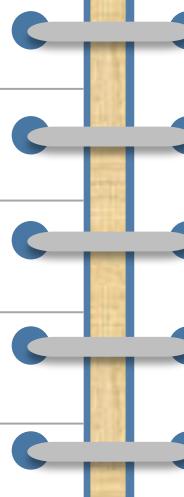
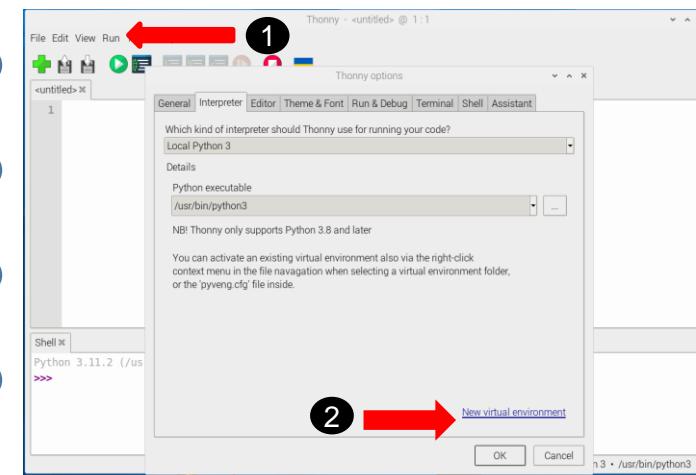
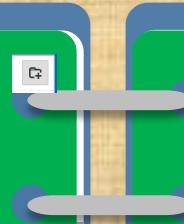
cloud-workshop folder and click the ‘Create

Folder  icon in the upper right. Type

iot_python and click ‘Create’.

3. Click ‘OK’ at the bottom and the new virtual

python environment will be created.



Raspberry Pi Code

1. Open the Tools -> Manage Packages menu

2. Type RPi.GPIO (note lowercase 'i') in the

'Search on PyPI' box and then click the

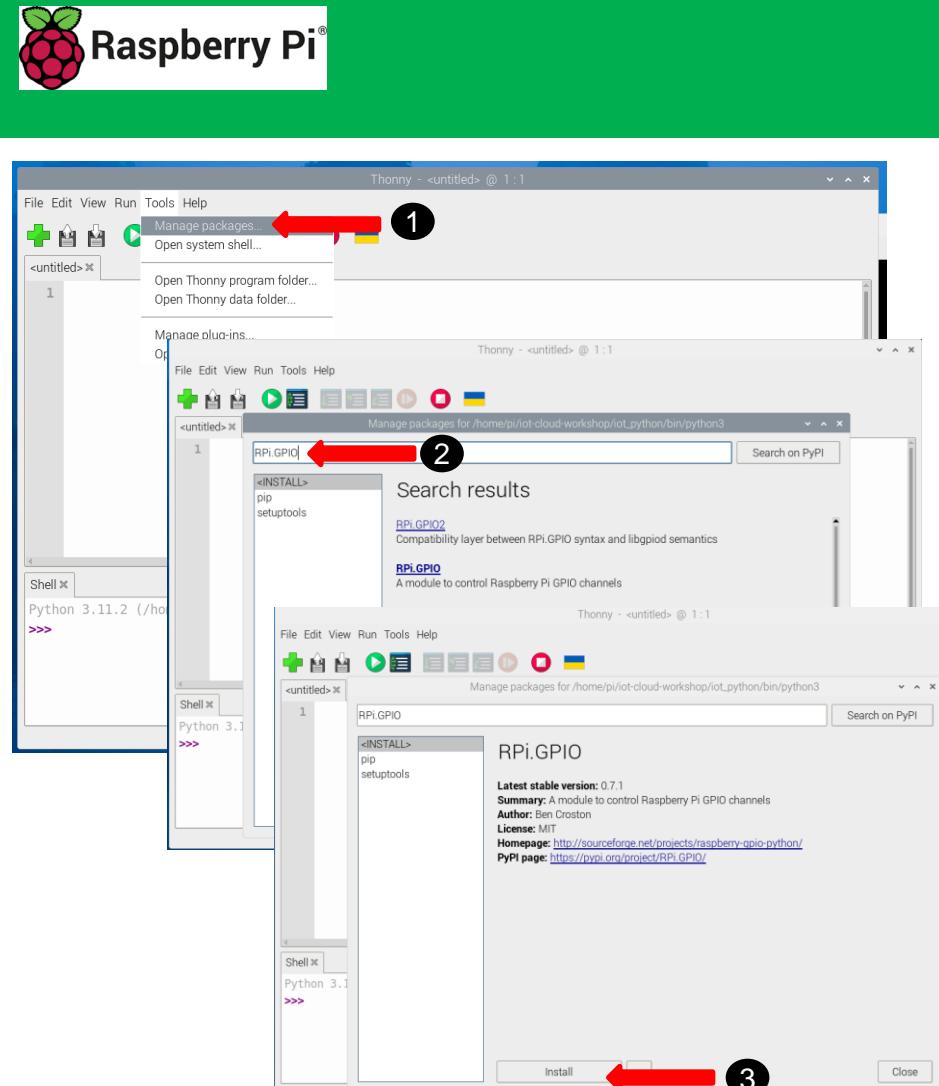
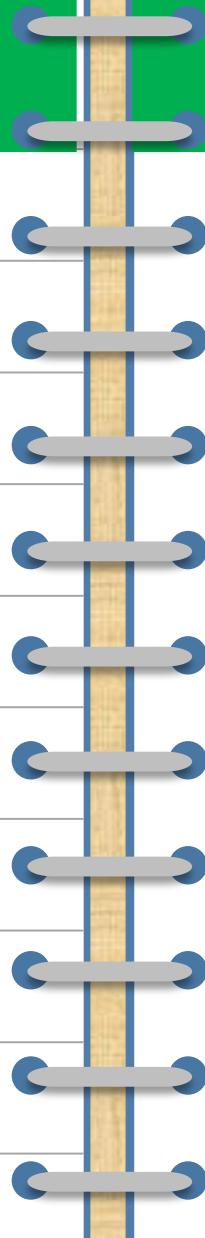
module when it appears and then click

Install button in lower left

3. Repeat the same process described above

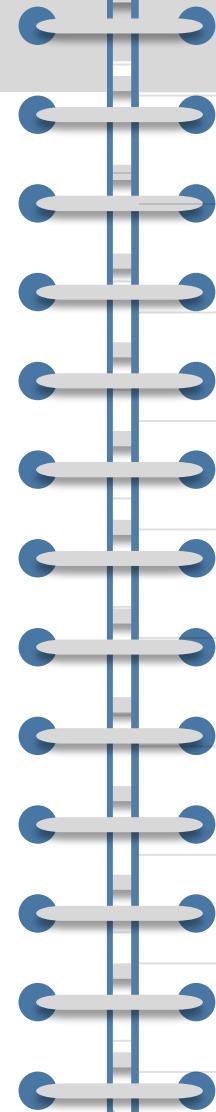
for the W1thermsensor module and the

google-cloud-pubsub modules





CHECKPOINT



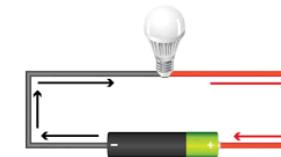
Git repo cloned, Thonny
launched, Python virtual
environment created, Python
modules installed

Check & Compare with
Instructor's Screen View

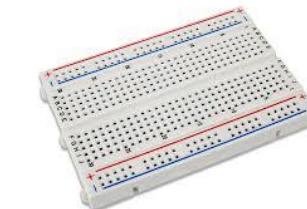
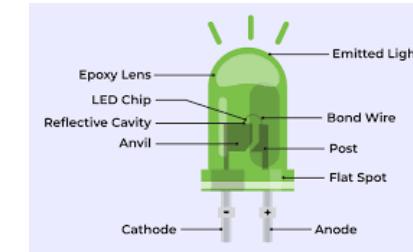
Raspberry Pi Sensor

Some key concepts:

- Voltage
- Resistors
- LEDs
- Breadboard
- Polarity



Wire Resistor Wire

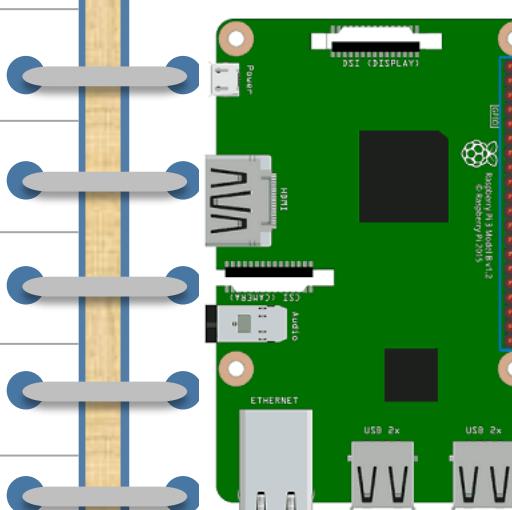
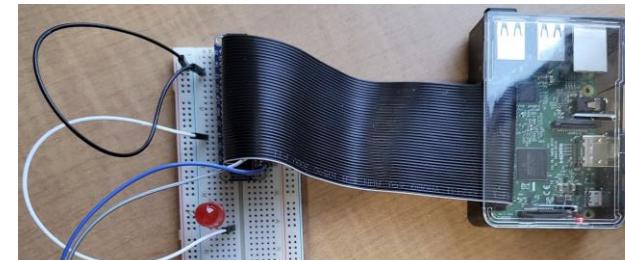
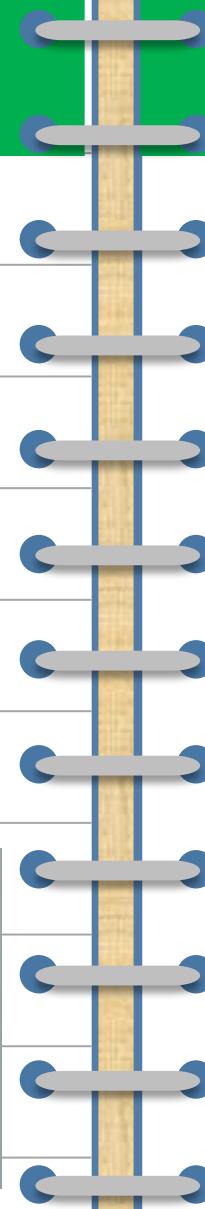
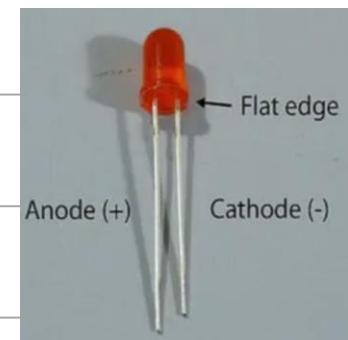


I WANT TO
KNOW MORE
 More info about circuits

Raspberry Pi Sensor

Make the following connections on your breadboard:

- Plug your red LED into the board with the anode (+) in row 30 column c and the cathode (-) in row 32 column c
- Plug a 4K resistor between row 30 column a and the negative (blue) rail
- Plug a jumper wire from pin 34 (GND) to the blue rail
- Plug a jumper wire from pin 12 (GPIO18) on the raspberry pi breakout board to row 32 column a



3.3V	1	2	5V
	3	4	5V
	5	6	GND
GPIO4 (GPIO_GCLK)	7	8	GPIO14 (UART_TXD0)
	9	10	GPIO15 (UART_RXD0)
	11	12	GPIO18 (GPIO_GEN1)
GPIO17 (GPIO_GEN0)	13	14	GND
GPIO27 (GPIO_GEN2)	15	16	GPIO23 (GPIO_GEN4)
GPIO22 (GPIO_GEN3)	17	18	GPIO24 (GPIO_GEN\$)
	19	20	GND
GPIO10 (SPI0_MOSI)	21	22	GPIO25 (GPIO_GEN6)
GPIO9 (SPI0_MISO)	23	24	GPIO8 (SPI_CE0_N)
GPIO11 (SPI0_CLK)	25	26	GPIO7 (SPI_CE1_N)
	27	28	ID_SD (I2C EEPROM)
	29	30	GND
ID_SC (I2C EEPROM)	31	32	GPIO12
GPIO5	33	34	GND
GPIO6	35	36	GPIO16
GPIO13	37	38	GPIO20
GPIO19	39	40	GPIO21
GPIO26			GND

I WANT TO
KNOW MORE
How does an LED work?

I WANT TO
KNOW MORE
Using GPIO pins with LEDs

Raspberry Pi Code

1. In the Thonny IDE, open the file named /home/pi/iot-cloud-workshop/led_blink.py
2. Read through the code and ask any questions you might have
3. Click the  button and check to see if your LED starts to blink
4. Press CTRL+C to stop the program



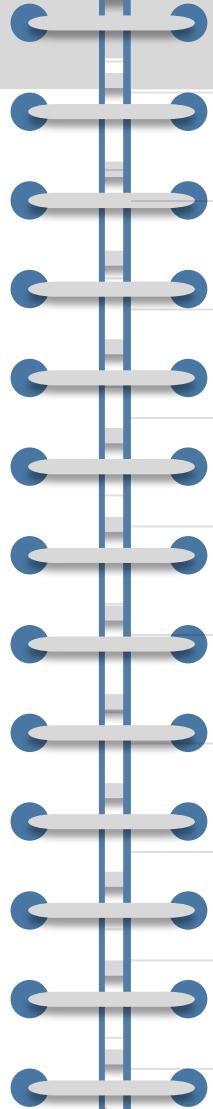
The screenshot shows the Thonny IDE interface. The top bar has tabs for File, Edit, View, Run, Tools, and Help. A red arrow points to the green 'Run' button in the toolbar. The code editor window contains the following Python script:

```
1 import RPi.GPIO as GPIO      # Import Raspberry Pi GPIO library
2 import time
3
4 # Setup the GPIO board on the Raspberry Pi...
5 GPIO.setwarnings(False)    # Ignore warnings
6 GPIO.setmode(GPIO.BOARD)   # Use physical pin numbering
7 GPIO.setup(12, GPIO.OUT, initial=GPIO.LOW)  # Set pin 12 (GPIO 18) to be an output
8
9 try:
10     while True:
11         # Turn on the LED
12         GPIO.output(12, GPIO.HIGH)
13         time.sleep(3) # Pause for 3 seconds...
14
15         # Turn off the LED
16         GPIO.output(12, GPIO.LOW)
17         time.sleep(2) # Pause for 2 seconds
18
19 except KeyboardInterrupt:
20     print("LED Blink Program Stopped!") # Enable checking for ^C sequence from
21
22
```

Below the code editor is a terminal window titled 'Shell' showing the command: >>> %Run led_blink.py and the output: LED Blink Program Stopped!



CHECKPOINT

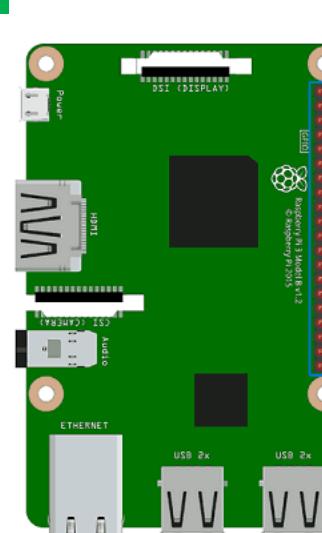
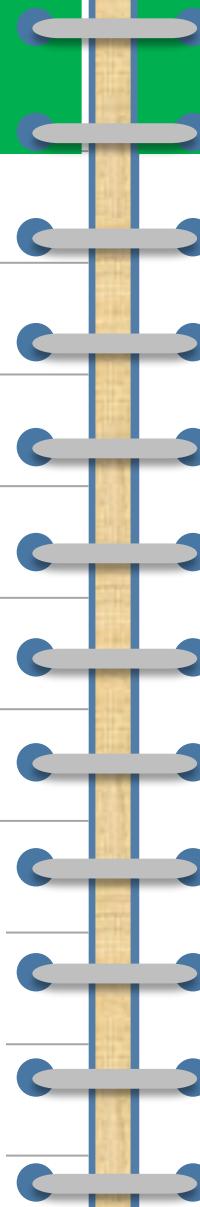


Code runs successfully to
blink the LED

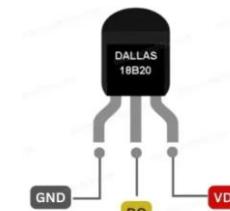
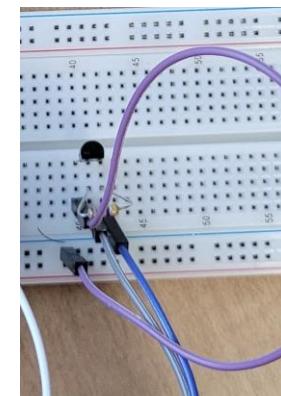
Raspberry Pi Sensor

Make the following connections on your breadboard:

- Plug your DS18B20 sensor into the board with pin 1(GND) in row 40 column e, pin 2(DATA) in row 41 column e, and pin 3 (voltage) in row 42 column e
- Plug a 4k resistor between rows 41 & 42 in column c
- Plug a jumper wire from row 40 col a to the blue (GND) rail
- Plug a jumper wire from row 41 column a to pin 7 (GPIO 4) on the Raspberry Pi breakout board
- Plug a jumper wire from row 42 column a to pin 1 (3v) on the Raspberry pi breakout board



3.3V	1	2	5V
	3	4	5V
	5	6	GND
	7	8	GPIO14 (UART_TXD0)
	9	10	GPIO15 (UART_RXD0)
	11	12	GPIO18 (GPIO_GEN1)
	13	14	GND
	15	16	GPIO23 (GPIO_GEN4)
3.3V	17	18	GPIO24 (GPIO_GEN\$)
	19	20	GND
	21	22	GPIO25 (GPIO_GEN6)
	23	24	GPIO8 (SPI_CE0_N)
	25	26	GPIO7 (SPI_CE1_N)
	27	28	ID_SC (I2C EEPROM)
	29	30	GND
	31	32	GPIO12
	33	34	GND
	35	36	GPIO16
	37	38	GPIO20
	39	40	GPIO21



I WANT TO
KNOW MORE
How does the DS18B20 work?

Raspberry Pi Sensor

Open a terminal and type:

```
ls -l /sys/bus/w1/devices
```

If you see a line starting with "28-"

then you have set things up correctly

and there is an active temp sensor

connected to your Pi!

A screenshot of a terminal window titled "pi@raspberrypi: ~". The window shows the command "ls -l /sys/bus/w1/devices" being run. The output includes a line starting with "28-3de1d44358ee" which is highlighted with a red arrow. Another line "w1_bus_master1" is also highlighted with a blue arrow. The terminal window has a dark background with white text and a grey header bar.

* Note that the DS18B20 will get very hot if it is

wired incorrectly (reverse polarity)



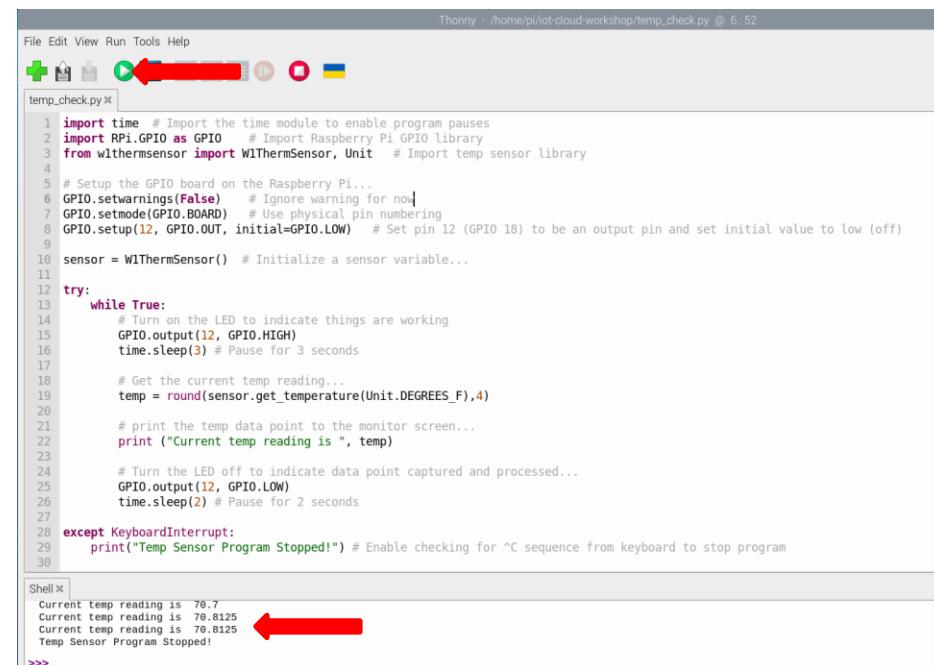
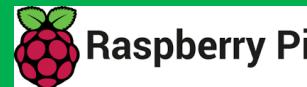
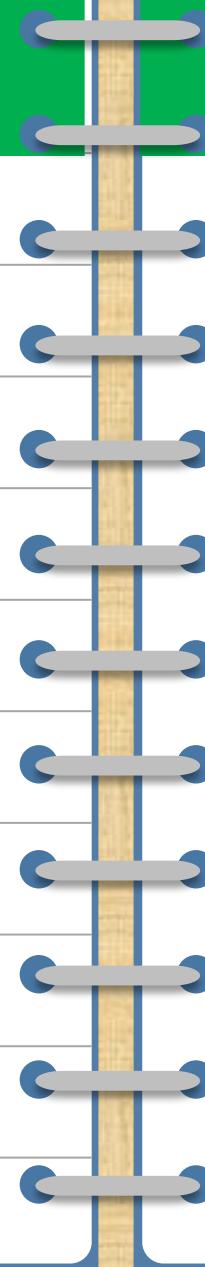
CHECKPOINT

- A vertical stack of ten white cards, likely index cards, bound together by a central vertical blue strip. Each card features a blue circular punch hole near its top and bottom edges.

Everyone can see the sensor
in the /sys/bus/w1/devices
folder

Raspberry Pi Code

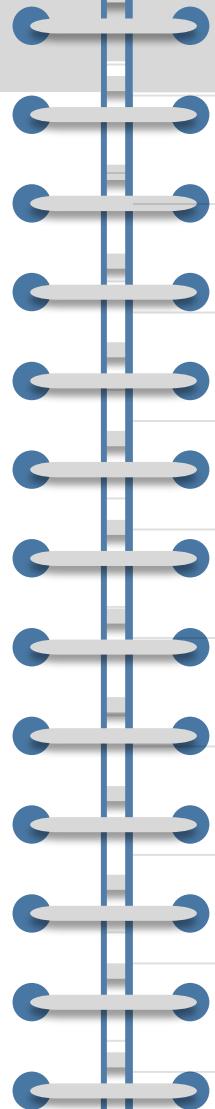
1. In the Thonny IDE, open the file named /home/pi/iot-cloud-workshop/temp_check.py
2. Read through the code and ask any questions you might have
3. Click the  button and check to see if temp readings print to the program output shell tab
4. Press CTRL+C to stop the program



```
File Edit View Run Tools Help
temp_check.py x
1 import time # Import the time module to enable program pauses
2 import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library
3 from w1thermsensor import W1ThermSensor, Unit # Import temp sensor library
4
5 # Setup the GPIO board on the Raspberry Pi...
6 GPIO.setwarnings(False) # Ignore warning for now
7 GPIO.setmode(GPIO.BCM) # Use physical pin numbering
8 GPIO.setup(12, GPIO.OUT, initial=GPIO.LOW) # Set pin 12 (GPIO 18) to be an output pin and set initial value to low (off)
9
10 sensor = W1ThermSensor() # Initialize a sensor variable...
11
12 try:
13     while True:
14         # Turn on the LED to indicate things are working
15         GPIO.output(12, GPIO.HIGH)
16         time.sleep(3) # Pause for 3 seconds
17
18         # Get the current temp reading...
19         temp = round(sensor.get_temperature(Unit.DEGREES_F),4)
20
21         # print the temp data point to the monitor screen...
22         print ("Current temp reading is ", temp)
23
24         # Turn the LED off to indicate data point captured and processed...
25         GPIO.output(12, GPIO.LOW)
26         time.sleep(2) # Pause for 2 seconds
27
28 except KeyboardInterrupt:
29     print("Temp Sensor Program Stopped!") # Enable checking for ^C sequence from keyboard to stop program
30
Shell x
Current temp reading is 70.7
Current temp reading is 70.8125
Current temp reading is 70.8125
Temp Sensor Program Stopped!
```



CHECKPOINT



Code runs successfully to
blink the LED and print the
temp reading to the screen

*Test putting your fingers on
the temp sensor to watch the
readings change*

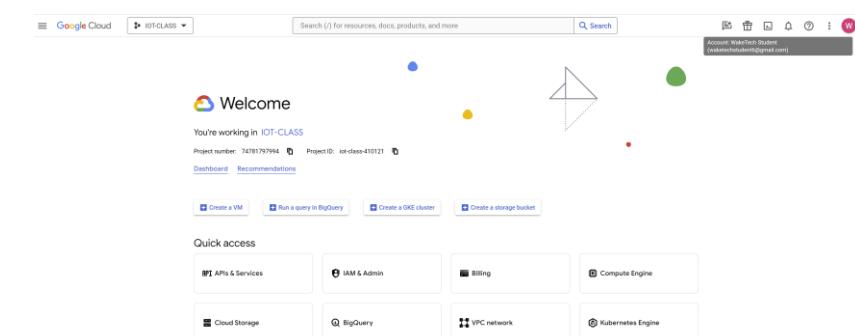
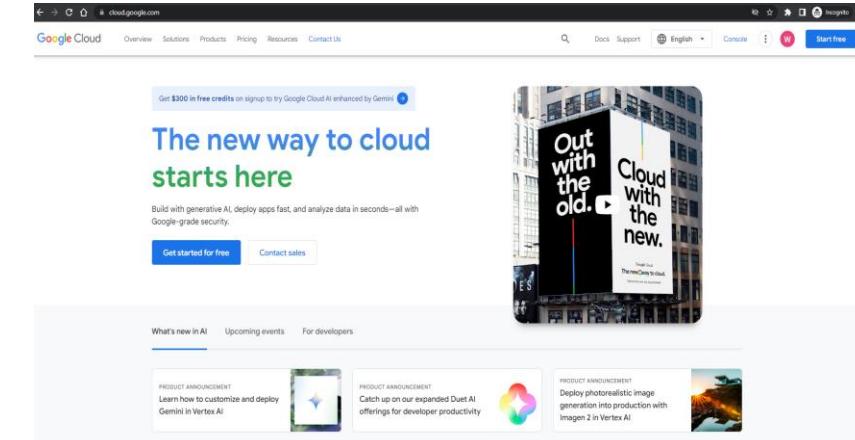


LUNCH BREAK

Google Cloud Platform (GCP)

STEP 1: Create a GCP Account

This step was completed prior to the workshop - confirm that you can log in to console.cloud.google.com in a browser window

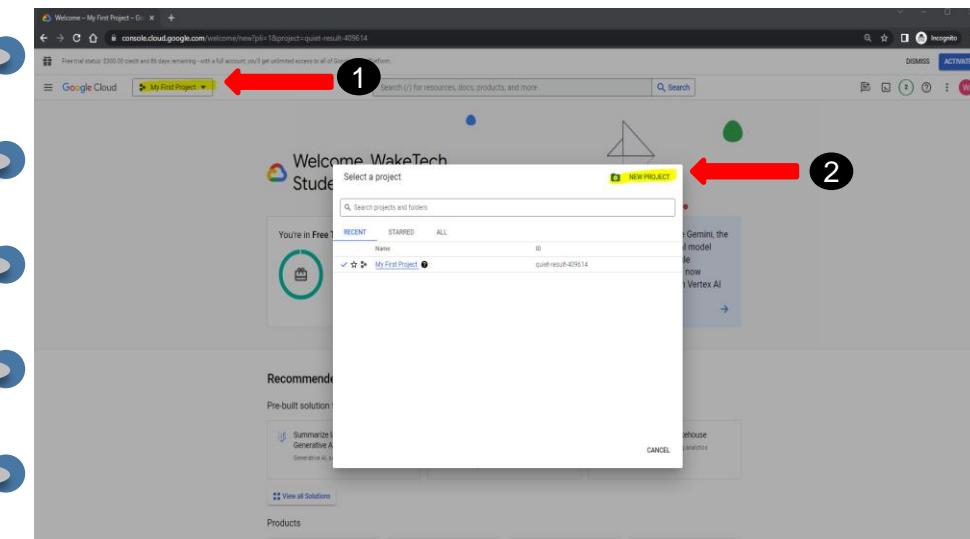


Google Cloud Platform (GCP)



STEP 2: Create a New Project

Once you are logged in to console.cloud.google.com, click on the active project name in the upper left. Then click the  NEW PROJECT button



Google Cloud Platform (GCP)

STEP 2: Create a New Project

Name the new project 'IOT-CLASS'

and then click the **CREATE** button



Google Cloud

Navigation menu

You have 23 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name *

IOT-CLASS  1

?

Project ID: iot-class-410121. It cannot be changed later. [EDIT](#)

Location *

No organization 

Parent organization or folder

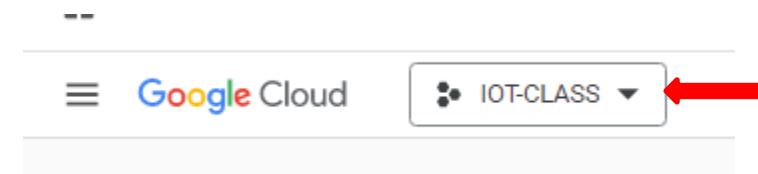
CREATE  2

Google Cloud Platform (GCP)



STEP 2: Create a New Project

Click the project list in upper left again and then select the new IOT-CLASS project – it should now show as your active project while you are working in the GCP console





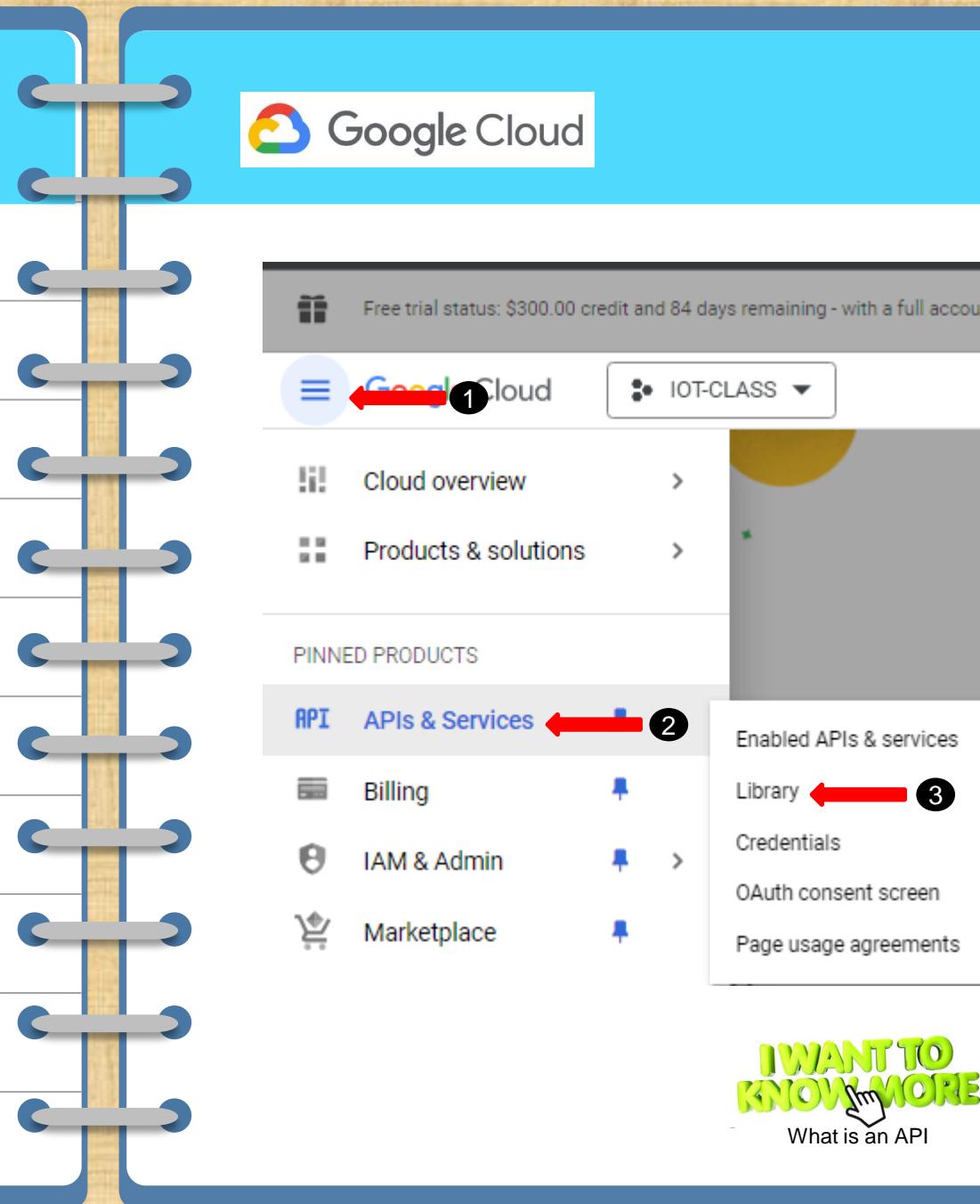
CHECKPOINT

Everyone has a new project
created named IOT-CLASS

Google Cloud Platform (GCP)

STEP 3: Enable APIs

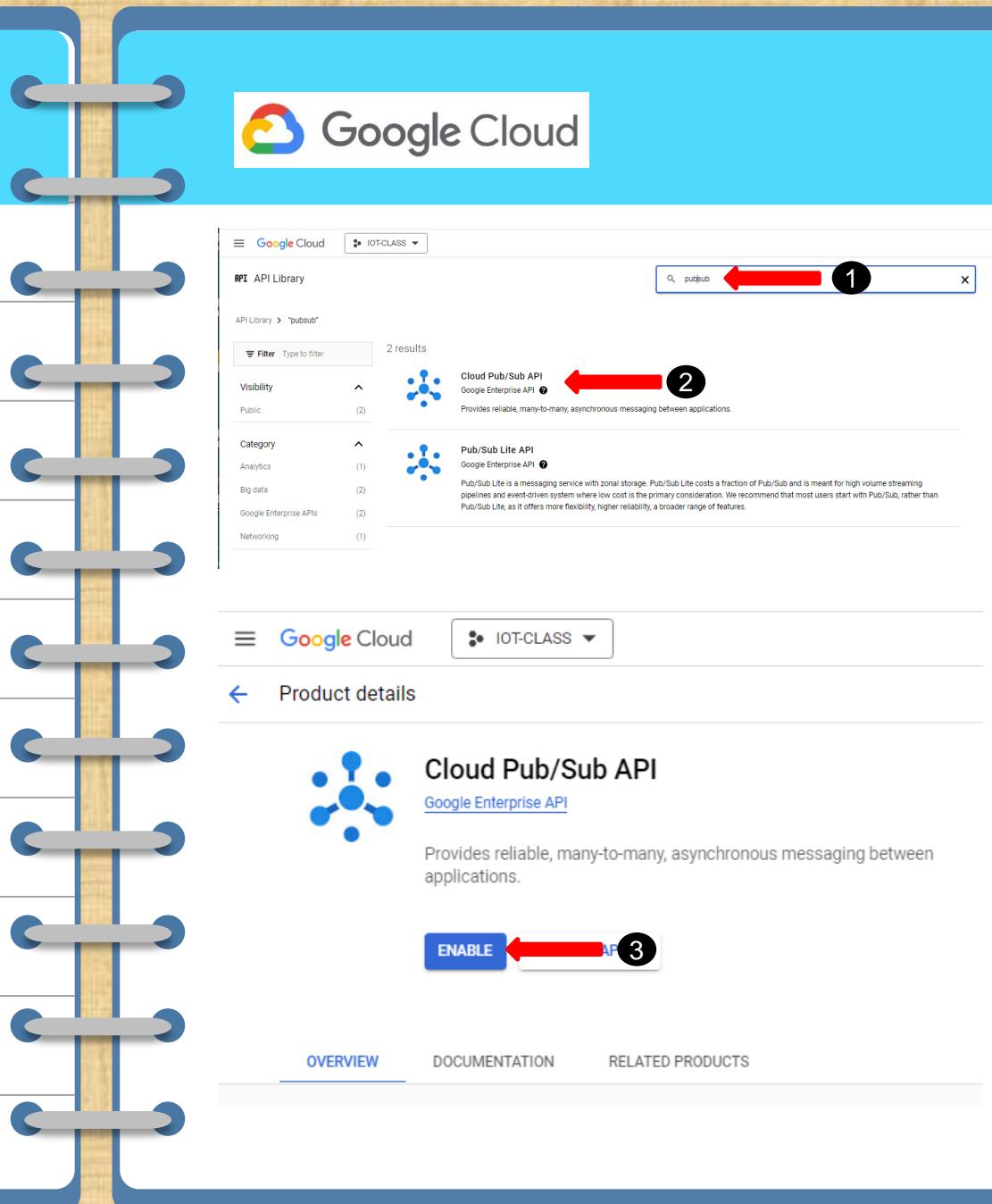
From the navigation menu on the left-hand side (often referred to as the 'hamburger' menu), select APIs & Services | Library



Google Cloud Platform (GCP)

STEP 3: Enable APIs

1. On the next screen, type 'pubsub' in the search box and then click the Cloud Pub/Sub API from the results
2. On the next screen, click the **ENABLE** button
3. Repeat this same process to enable the Application Integration API



Google Cloud Platform (GCP)

STEP 4: Create a Service Account

From the navigation menu on the left-hand side, select IAM & Admin | Service Accounts



The screenshot shows the Google Cloud navigation menu on the left. A red arrow labeled 1 points to the three-dot menu icon at the top left. Another red arrow labeled 2 points to the 'IAM & Admin' item in the list. A third red arrow labeled 3 points to the 'Service Accounts' link under the IAM & Admin section. The menu also lists other services like Cloud overview, Products & solutions, APIs & Services, Billing, Marketplace, Compute Engine, Kubernetes Engine, Cloud Storage, BigQuery, VPC network, Cloud Run, SQL, Security, and Google Maps Platform.

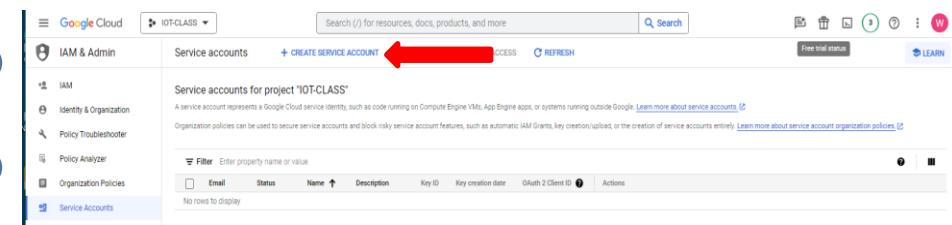


What is a Google Service Account

Google Cloud Platform (GCP)

STEP 4: Create a Service Account

On the next screen, click the
[+ CREATE SERVICE ACCOUNT](#) button near the
top left of the screen



Google Cloud Platform (GCP)

STEP 4: Create a Service Account

Give the new service account a name of “IOT-PROJECT” and then click

CREATE AND CONTINUE



Google Cloud IOT-CLASS < Create service account

Service account details

Service account name: IOT-PROJECT 1

Display name for this service account

Service account ID *: iot-project-svc-acct 2

Email address: iot-project-svc-acct@iot-class-410121.iam.gserviceaccount.com

Service account description

Describe what this service account will do

CREATE AND CONTINUE 2

Grant this service account access to project (optional)

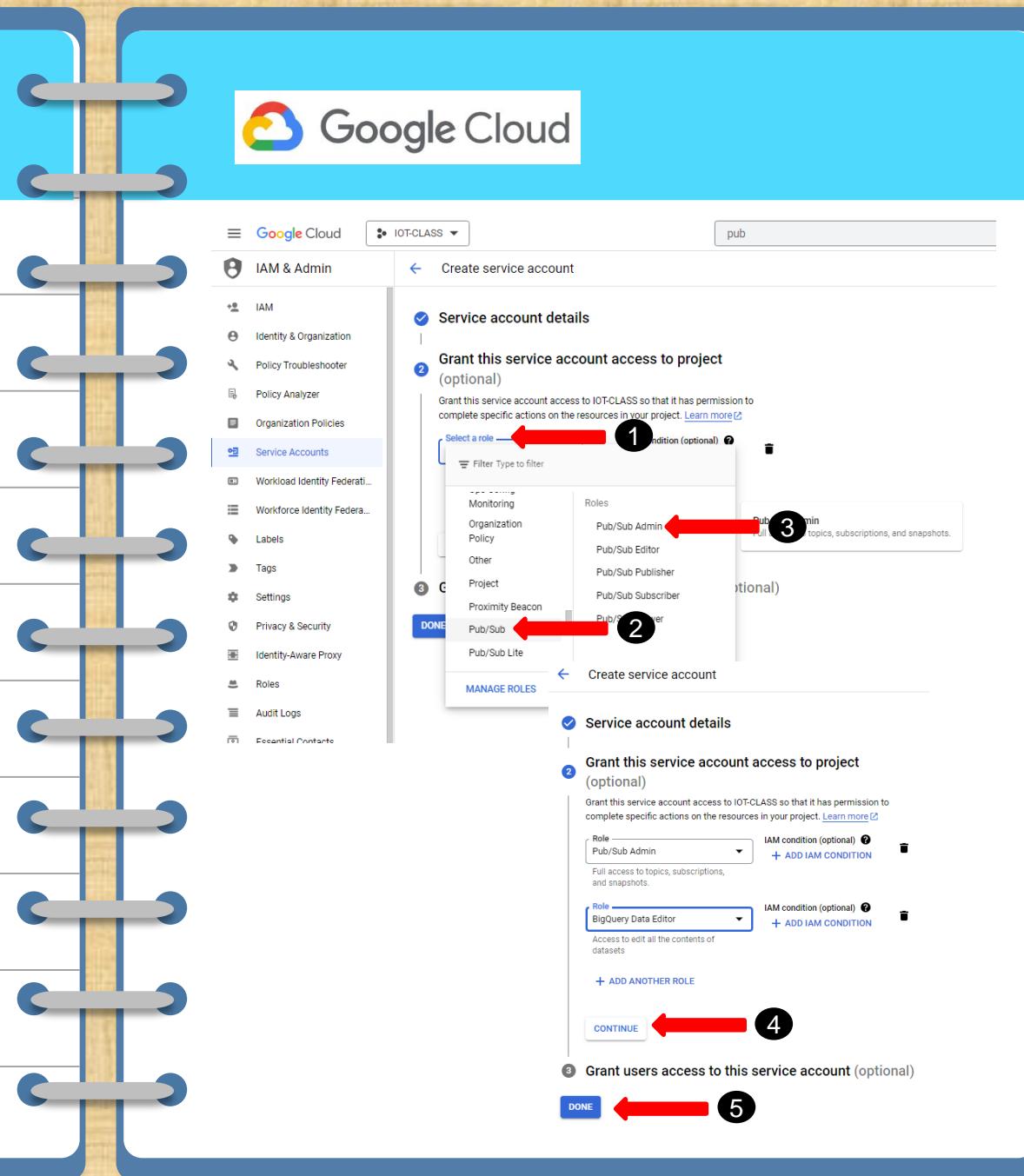
Grant users access to this service account (optional)

DONE CANCEL

Google Cloud Platform (GCP)

STEP 4: Create a Service Account

1. In the 'Select a role' drop-down list, scroll down and select the 'Pub/Sub Admin' role
2. Repeat this process to also add the 'BigQuery Data Editor' role
3. Click **CONTINUE** and then the **DONE** button at the bottom.



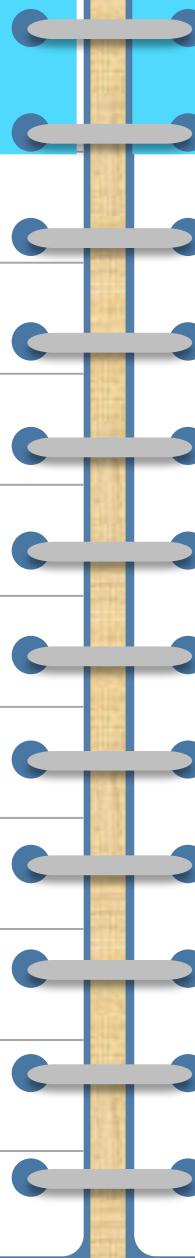


CHECKPOINT

Google Cloud Platform (GCP)

STEP 5: GET API CREDENTIALS

1. While on the service accounts page,
Click your newly created service account
to open it's details page:



The screenshots illustrate the steps to generate API credentials. Step 1 shows the creation of a service account. Step 2 shows navigating to the 'Keys' tab where new keys can be generated.

2. On the Service Account details page,
click the 'KEYS' tab, and then click the
ADD KEY button, and select the 'Create
new key' option.

Google Cloud Platform (GCP)

STEP 5: GET API CREDENTIALS

1. Select JSON for the key type and then click the **CREATE** button
2. A new public/private key pair is generated and the private key file is downloaded to your computer. Save it as

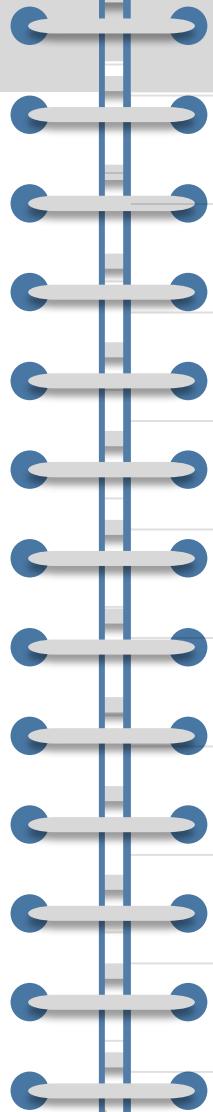
/home/pi/iot-cloud-workshop/credentials.json



I WANT TO
KNOW MORE
4
What is a JSON file



CHECKPOINT

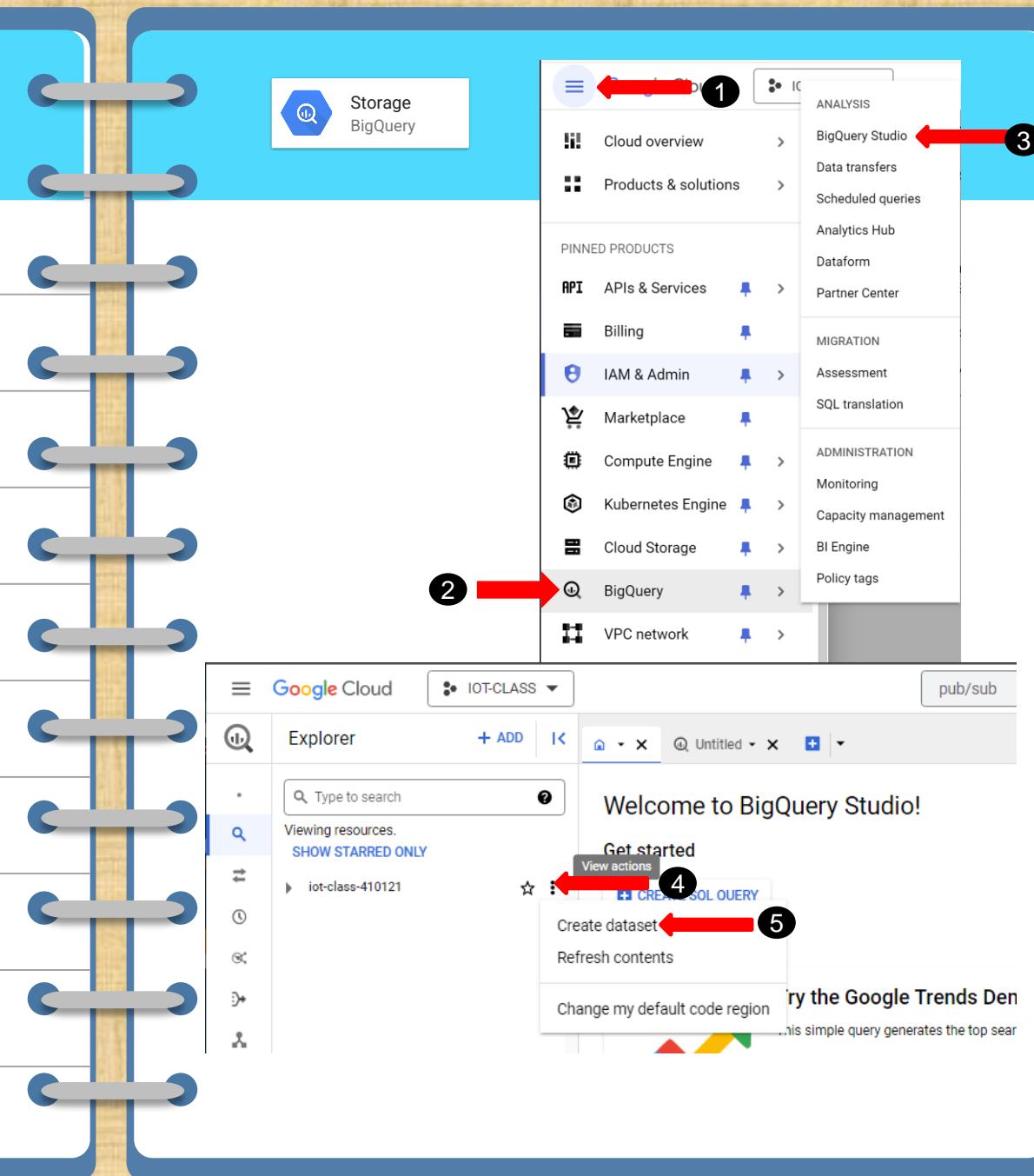


Everyone has the
credentials.json file
saved locally on their
Raspberry Pi

Google Cloud Platform (GCP)

STEP 6: BIGQUERY TABLE

From the service navigation menu,
choose BigQuery | BigQuery Studio –
click the 3 dots to the left of your
project name in the explorer pane and
select ‘Create dataset’

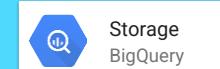


Google Cloud Platform (GCP)

STEP 6: BIGQUERY TABLE

Type 'RPITempSensor' in the Dataset

ID field and click the **CREATE DATASET** button
at the bottom



Create dataset

Project ID
iot-class-410121

CHANGE

Dataset ID *
RPITempSensor 1

Letters, numbers, and underscores allowed

Location type ?

- Region
Specify a region to colocate your datasets with other Google Cloud services.
- Multi-region
Allow BigQuery to select a region within a group to achieve higher quota limits.

Multi-region *
US (multiple regions in United States) 2

Default table expiration

Enable table expiration ?

Default maximum table age Days

Advanced options

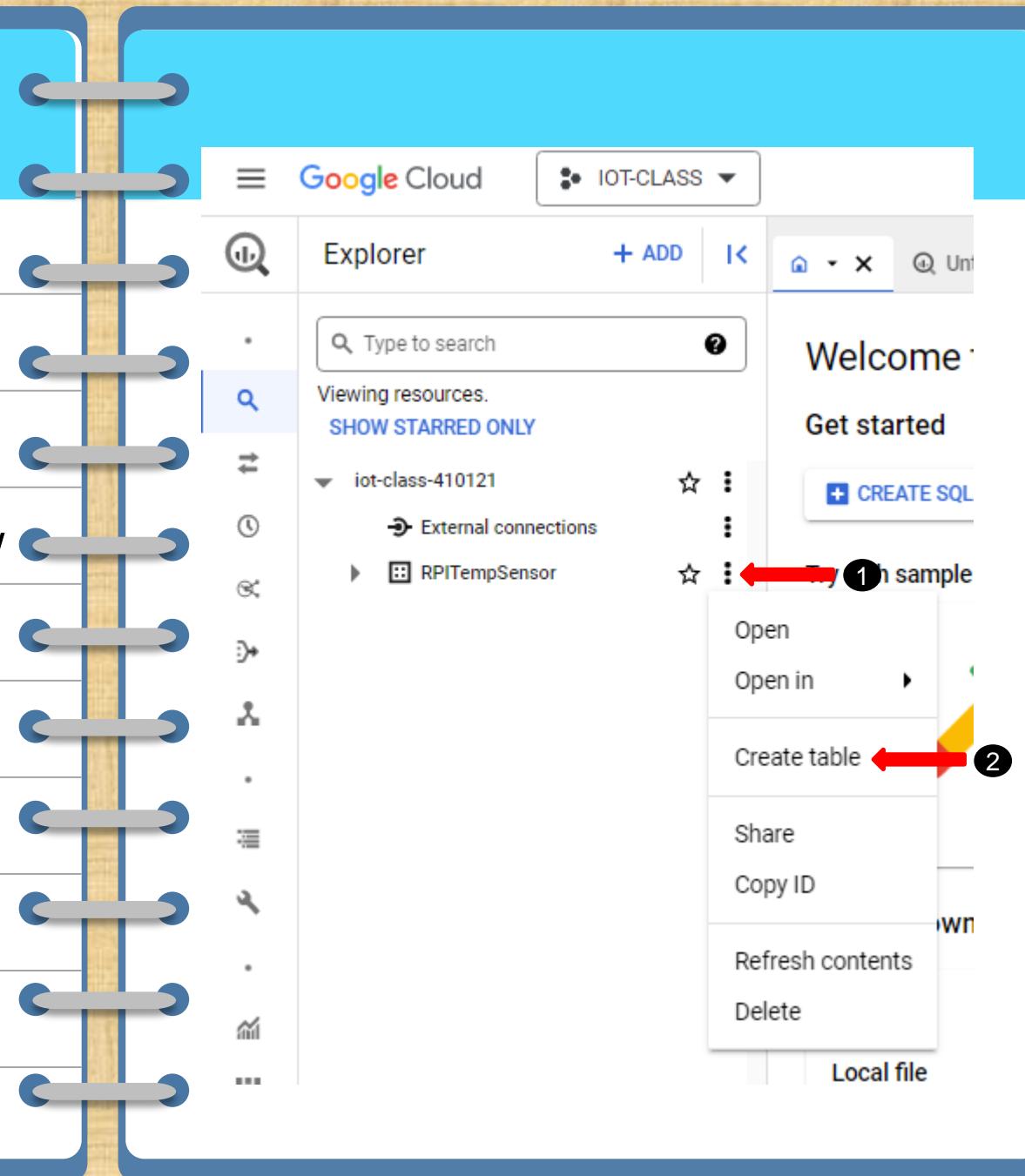
CREATE DATASET

CANCEL 2

Google Cloud Platform (GCP)

STEP 6: BIGQUERY TABLE

You'll see the new data set appear at
the bottom of your BQ Explorer window
– click the 3 dots to the right of the
dataset name and select 'Create table'



Google Cloud Platform (GCP)

STEP 6: BIGQUERY TABLE

In the 'Create table' window, set the

Table name to TEMP and then click the

plus sign under the Schema section to

add fields. Add the following fields:

time – TIMESTAMP

temp – FLOAT

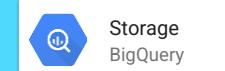
Then click the **CREATE TABLE** button at the
bottom



Google Cloud Platform (GCP)

STEP 6: BIGQUERY TABLE

You should now see the new table at the bottom of your BQ Explorer window.



A screenshot of the Google Cloud BigQuery Explorer interface. On the left, the 'Explorer' sidebar shows a tree view with 'Viewing resources. SHOW STARRED ONLY' and a node 'iot-class-410121' expanded to show 'RPTempSensor'. The main panel shows a table named 'TEMP' with two columns: 'time' (TIMESTAMP, NULLABLE) and 'temp' (FLOAT, NULLABLE). The interface includes tabs for 'SCHEMA', 'DETAILS', 'PREVIEW', 'LINEAGE', 'DATA PROFILE', and 'DATA QUALITY'. A 'SEARCH' bar at the top right contains the text 'pub/sub'.



CHECKPOINT

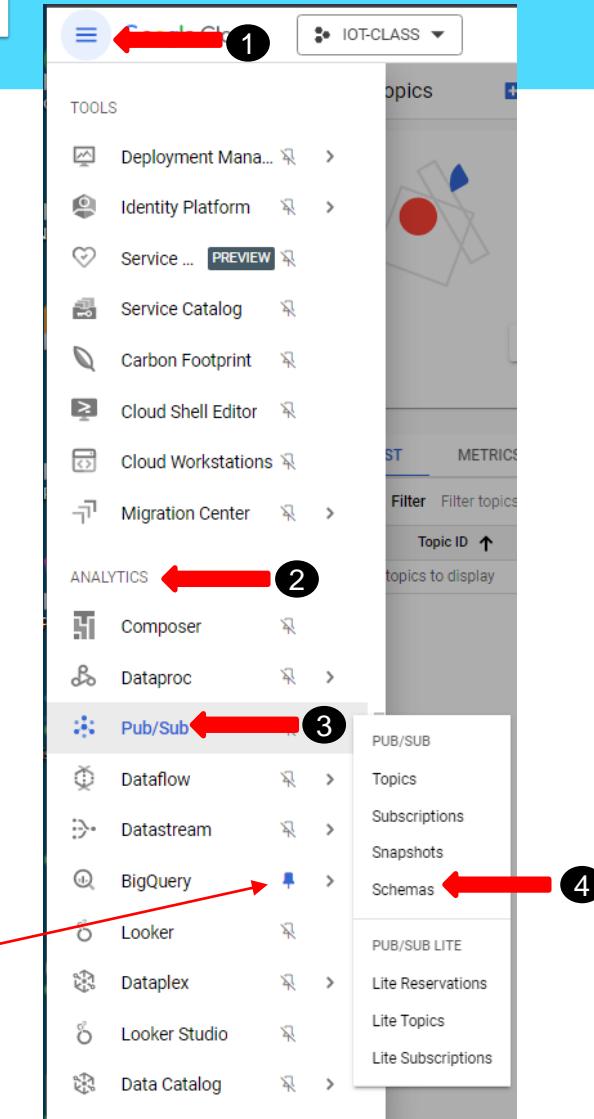
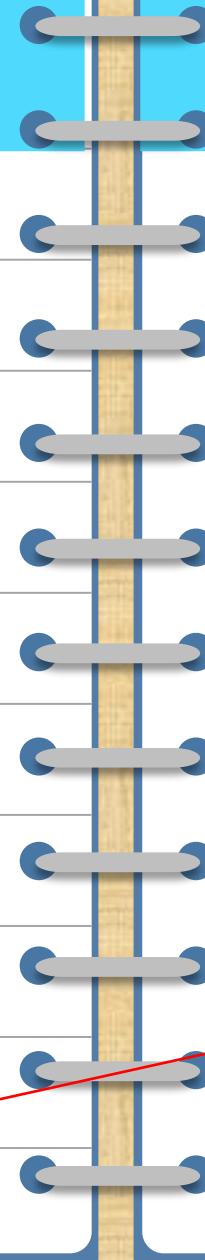
- Everyone has a new BigQuery dataset and table created

Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

From the navigation menu on the left-hand side, expand the 'More Products' section and scroll down to 'ANALYTICS' and then select Pub/Sub | Schemas as shown (or you can just type pub/sub in the top search menu and select the Pub/Sub product)

TIP: Click the  icon next to a service name in the nav menu to keep it 'pinned' to the top of the menu

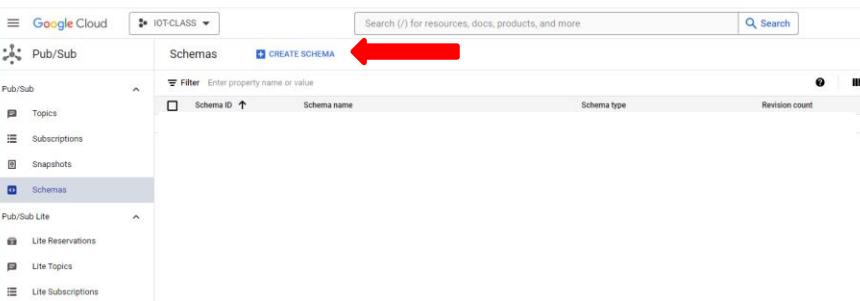


Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

On the 'Schemas' page, click the

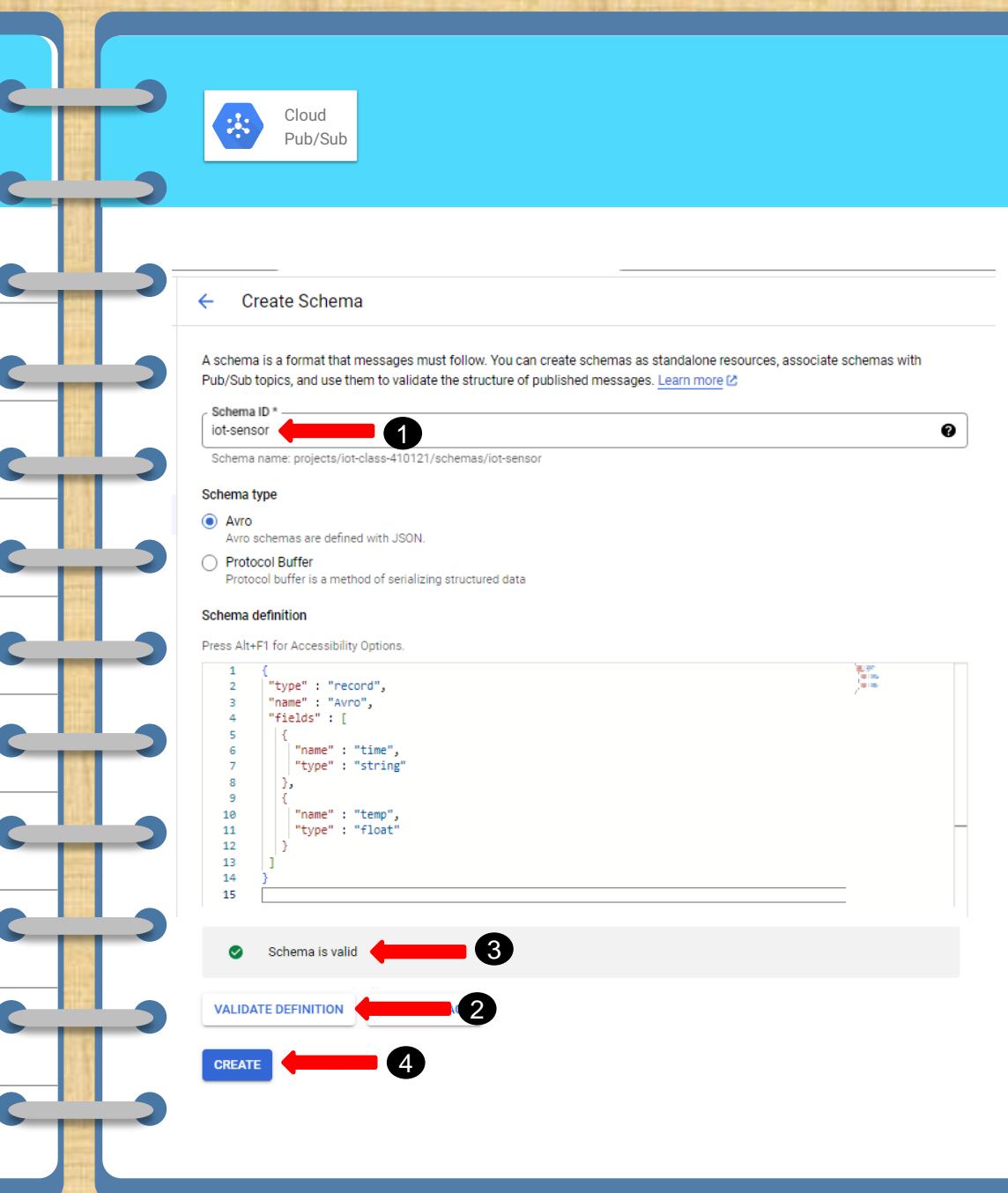
+ CREATE SCHEMA button



Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

1. Type 'iot-sensor' in the Schema ID field
2. Click the **VALIDATE DEFINITION** button at the bottom left
3. Make sure you see  Schema is valid
4. Click **CREATE**



The screenshot shows the 'Create Schema' page in the Google Cloud Platform. At the top, there's a 'Cloud Pub/Sub' icon. Below it, the title 'Create Schema' is displayed with a back arrow. A descriptive text explains that a schema defines the format of messages: 'A schema is a format that messages must follow. You can create schemas as standalone resources, associate schemas with Pub/Sub topics, and use them to validate the structure of published messages.' A link to 'Learn more' is provided.

The form fields include:

- Schema ID ***: The value 'iot-sensor' is highlighted with a red arrow and circled with number 1.
- Schema type**: The 'Avro' option is selected, with a brief description below: 'Avro schemas are defined with JSON.'
- Protocol Buffer**: This option is also listed but not selected.
- Schema definition**: A code editor displays an Avro schema definition:

```
1 {  
2   "type" : "record",  
3   "name" : "Avro",  
4   "fields" : [  
5     {  
6       "name" : "time",  
7       "type" : "string"  
8     },  
9     {  
10      "name" : "temp",  
11      "type" : "float"  
12    }  
13  ]  
14}  
15
```

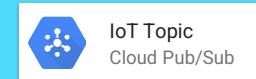
At the bottom of the page, there are two buttons: **VALIDATE DEFINITION** (highlighted with a red arrow and circled with number 2) and **CREATE** (highlighted with a red arrow and circled with number 4). A status message 'Schema is valid' with a green checkmark is shown above the 'VALIDATE DEFINITION' button.

Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

On the next screen (Schema details)

click the **+ CREATE TOPIC** button



IoT Topic
Cloud Pub/Sub

The screenshot shows the 'Schema details' page for a schema named 'iot-sensor'. At the top, there are buttons for 'CREATE TOPIC' (highlighted with a red arrow), 'ROLL BACK', and 'DELETE SCHEMA'. Below this, the 'Schema details' section shows the schema name, type (AVRO), and revision count. The 'Revisions' section lists a single revision (ID: 177e2689, Creation Time: 1/7/24, 8:58 AM). The 'Details' section displays the AVRO schema definition:

```
{ "type": "record", "name": "sensor", "fields": [ { "name": "time", "type": "string" }, { "name": "temp", "type": "float" } ] }
```

At the bottom right of the 'Details' section is a 'TEST MESSAGE' button.

Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

We'll now create a new Pub/Sub Topic which will be used to collect the telemetry data coming in from the temp sensors.

Type 'iot-sensor' for the Topic ID and then click the **CREATE** button at the bottom of the screen

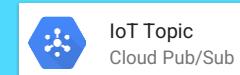
The screenshot shows the 'Create topic' dialog box. At the top, there's a header with a back arrow and the text 'Create topic'. Below it is a 'Topic ID *' field containing 'iot-sensor', with a red arrow pointing to it labeled '1'. Underneath the field, the full topic name 'projects/iot-class-410121/topics/iot-sensor' is shown. There are two checked checkboxes: 'Add a default subscription' and 'Use a schema'. The 'Schema' section is expanded, showing a 'Schema name' field with 'projects/iot-class-410121/schemas/iot-sensor'. A dropdown menu for 'Select a message encoding *' is set to 'JSON'. The 'Revision range' section is also expanded, showing dropdown menus for 'First revision allowed' (set to 'None - default to the schema's oldest revision at time of publish') and 'Last revision allowed' (set to 'None - default to the schema's newest revision at time of publish'). A checkbox for 'Enable message retention' is unchecked. The 'Encryption' section shows a selected radio button for 'Google-managed encryption key' with the note 'No configuration required'. A second radio button for 'Customer-managed encryption key (CMEK)' is present with the note 'Manage via [Google Cloud Key Management Service](#)'. At the bottom right is a large blue 'CREATE' button, with a red arrow pointing to it labeled '2'.

Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

You will now see the details screen for the newly created `iot-sensor` Topic.

Click the [CREATE SUBSCRIPTION](#) link.



The screenshot shows the Google Cloud Pub/Sub Topics page. The left sidebar has options like Pub/Sub, Topics, Subscriptions, Snapshots, Schemas, Pub/Sub Lite, Lite Reservations, Lite Topics, and Lite Subscriptions. The main area shows a topic named `iot-sensor`. Below it, there are sections for Export to BigQuery and Export to Cloud Storage. At the bottom, there's a **SUBSCRIPTIONS** section with a **CREATE SUBSCRIPTION** button, which is highlighted with a red arrow. A table below lists a single subscription: `iot-sensor_sub` with `projects/iot-class-410121/subscriptions/iot-sensor-sub` and `iot-class-410121`.

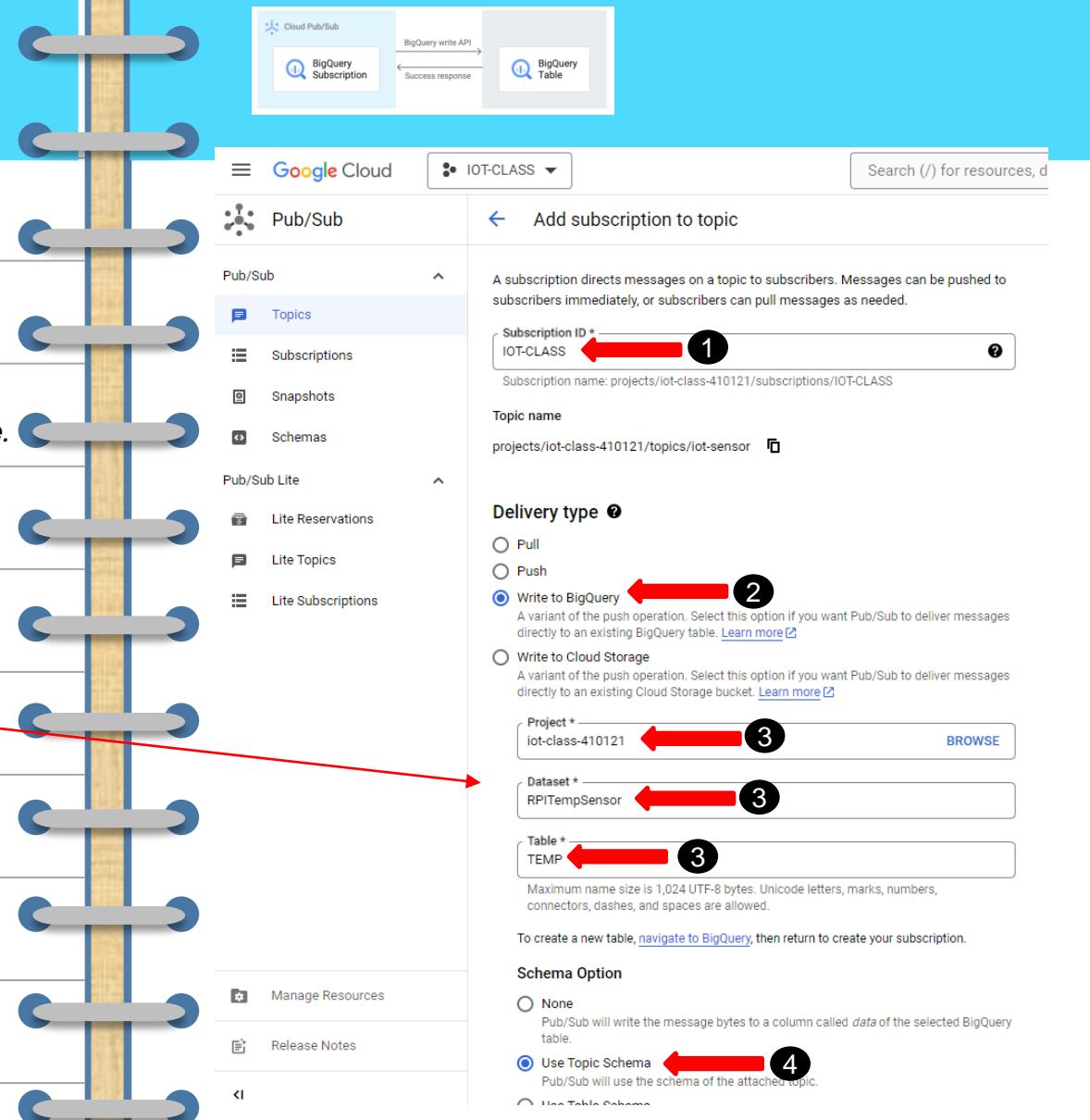
Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

We'll now create a Pub/Sub 'subscription' to enable real-time

ingestion of the IoT temp sensor data into the BigQuery database.

1. Type 'IOT-CLASS' for the Subscription ID
2. Select 'Write to BigQuery' for the Delivery type
3. Select the Project, Dataset, and Table as shown
4. Select the 'Use Topic Schema' choice for Schema
5. Scroll to the bottom of the page and click the
Option
CREATE button (not shown in screenshot)



Raspberry Pi Code

1. In the Thonny IDE, open the file named

/home/pi/iot-cloud-workshop/sensor_cloud.py

2. Read through the code and ask questions

3. Edit the values of the

GOOGLE_APPLICATION_CREDENTIALS & project_id

variables to match your environment

4. Click the button and check to see if temp

readings print to the screen

5. Press CTRL+C to stop the program

The diagram shows a vertical wooden lanyard with ten circular holes. It is positioned vertically along the right side of the slide, corresponding to the numbered steps listed on the left.

Raspberry Pi®

Thonny - /home/pi/iot-cloud-workshop/sensor_cloud.py @ 16:2

File Edit View Run Tools Help

sensor_cloud.py *x led_blink.py x temp_check.py x

1 import os # Import the os module to support accessing locally saved credential/key files
2 import time # Import the time module to enable program pauses
3 import datetime # Import the datetime module to support adding timestamp details to the sensor data
4 import RPi.GPIO as GPIO # Import the Raspberry Pi GPIO library
5 from w1thermsensor import W1ThermSensor, Unit # Import the temp sensor library
6 from google.cloud import pubsub_v1 # Import the pubsub module to support publishing data into GCP cloud
7 import json # Import the json module to simplify publishing sensor data in a json format
8
9 # Set some static GCP connection variables
10 os.environ["GOOGLE_APPLICATION_CREDENTIALS"]="/home/pi/iot-cloud-workshop/<REPLACE WITH YOUR KEY FILE NAME>.json" #
11 project_id = "<REPLACE WITH YOUR PROJECT ID>" #
12 topic_name = 'iot-sensor'
13
14 # Next two lines have test values for code checking (will not work on your Pi and can be ignored)
15 os.environ["GOOGLE_APPLICATION_CREDENTIALS"]="/home/pi/iot-cloud-workshop/iot-class-410121-29be3e3bc63c.json"
16 project_id = "iot-class-410121"
17
18 # Create a new pubsub object that will let us use the GCP PubSub API to publish our sensor data
19 publisher = pubsub_v1.PublisherClient()
20 topic_path = publisher.topic_path(project_id, topic_name)
21
22 # Set the timezone as Eastern Standard
23 os.environ['TZ'] = 'EST+05EDT,M3.1.0,M11.1.0'
24 time.tzset()
25
26 # Setup the GPIO board on the Raspberry Pi...
27 GPIO.setwarnings(False) # Ignore warnings
28 GPIO.setmode(GPIO.BCM) # Use physical pin numbering
29 GPIO.setup(12, GPIO.OUT, initial=GPIO.LOW) # Set pin 12 (GPIO 18) to be an output pin and set initial value to low (off)

Shell x

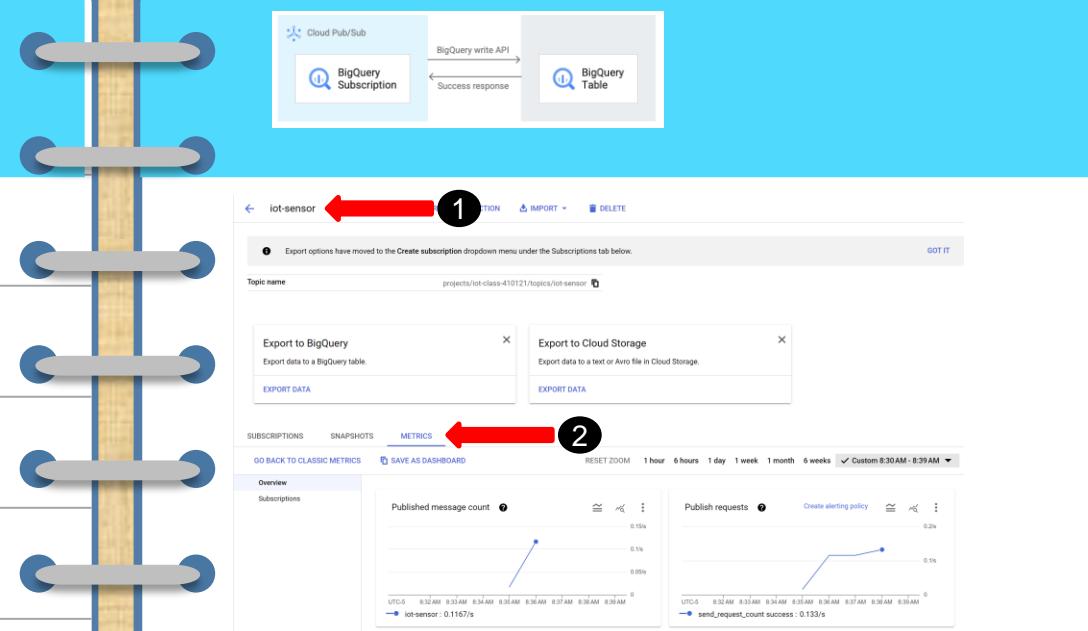
Program Run Time: Current Local Time: 09:21:22 02/08/24 EST
time: 2024-02-08 09:21:24 temp: 70.925
time: 2024-02-08 09:21:30 temp: 71.0375
time: 2024-02-08 09:21:36 temp: 71.0375

1. Red arrow points to the line: `os.environ["GOOGLE_APPLICATION_CREDENTIALS"]="/home/pi/iot-cloud-workshop/<REPLACE WITH YOUR KEY FILE NAME>.json"`. A red circle with the number 1 is placed near the end of the arrow.
2. Red arrow points to the line: `project_id = "<REPLACE WITH YOUR PROJECT ID>"`. A red circle with the number 2 is placed near the end of the arrow.
3. Red arrow points to the green play button icon in the Thonny IDE toolbar. A red circle with the number 3 is placed near the end of the arrow.
4. Red arrow points to the output window showing temperature readings. A red circle with the number 4 is placed near the end of the arrow.

Google Cloud Platform (GCP)

STEP 7: CREATE PUB/SUB TOPIC

1. Open your `iot-sensor` Pub/Sub topic and click the **METRICS** tab to ensure messages are being published to the topic



2. Open BigQuery Studio and verify that data is being collected in your TEMP table via the subscription (click the **PREVIEW** tab)

The screenshot shows the BigQuery Studio interface with the `TEMP` table selected. A red arrow labeled 3 points to the `RPITempSensor` row under the `External connections` section. Another red arrow labeled 4 points to the **PREVIEW** tab at the bottom of the table view. The table data shows rows of timestamped temperature values.

Row	time	temp
27	2024-01-07 20:54:00 UTC	66.0875015...
28	2024-01-07 20:54:05 UTC	66.0875015...
29	2024-01-07 09:09:20 UTC	68.4499969...
30	2024-01-07 15:35:57 UTC	69.012469...
31	2024-01-07 15:36:05 UTC	69.012469...
32	2024-01-07 15:36:05 UTC	69.1999969...
33	2024-01-07 20:54:52 UTC	66.1999969...
34	2024-01-07 20:35:01 UTC	66.1999969...
35	2024-01-07 20:35:10 UTC	66.1999969...
36	2024-01-07 20:40:55 UTC	66.1999969...
37	2024-01-11 15:20:31 UTC	70.5875015...
38	2024-01-11 15:20:40 UTC	70.6999969...
39	2024-01-11 15:20:49 UTC	70.6999969...
40	2024-01-11 15:20:57 UTC	70.6999969...
41	2024-01-11 15:21:06 UTC	83.0749969...
42	2024-01-11 15:21:15 UTC	86.249984...
43	2024-01-29 12:59:17 UTC	73.512469...
44	2024-01-29 12:59:25 UTC	74.3000030...
45	2024-01-29 12:59:35 UTC	84.7624969...
46	2024-01-30 10:17:50 UTC	74.6374969...
47	2024-02-05 14:37:09 UTC	71.599984...



CHECKPOINT

- Everyone has successfully sent temp data from the Raspberry Pi into GCP (Check Pub/Sub Stats & BigQuery table contents)

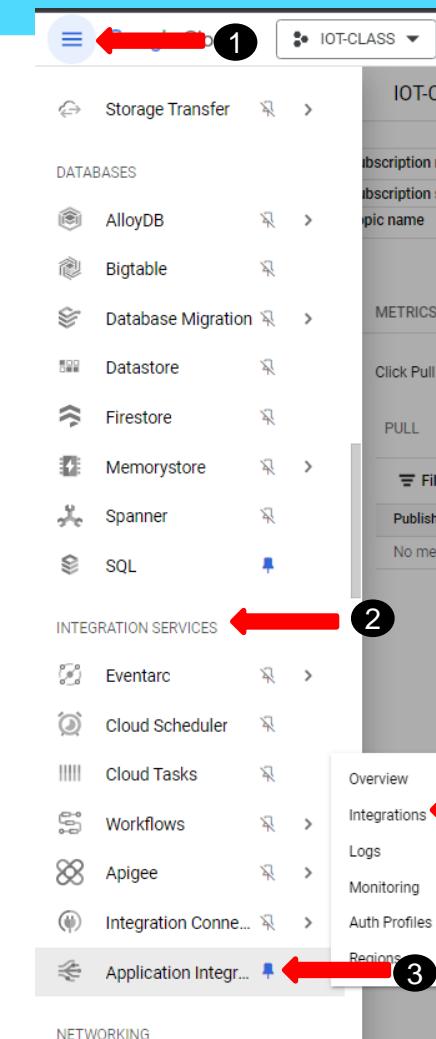
Google Cloud Platform (GCP)

STEP 8: Alerting

We will use the GCP Application Integration tool to create an email alert when the temperature from our IoT sensor exceeds a certain value.

From the nav menu, expand 'MORE PRODUCTS' at the bottom and scroll down to 'INTEGRATION SERVICES'.

Select 'Application Integration' and then 'Integrations'.



Google Cloud Platform (GCP)

STEP 8: ALERTING

You will see the App Integrations overview page. Select the 'us-east1' region from the pick-list and then click the 'QUICK SETUP' link.

The screenshot shows the Google Cloud Application Integration Overview page. At the top left, there's a navigation bar with 'Google Cloud' and 'IOT-CLASS'. On the right, there's a search bar with 'applic' and a magnifying glass icon. Below the navigation, there's a sidebar with 'Overview', 'Integrations', 'Logs', 'Monitoring', 'Auth', and 'Regions'. The 'Regions' section has a dropdown menu with 'us-east1' selected. To the right of the sidebar, there's a main content area with a heading 'Get started with Application Integration' and a sub-section 'Quick setup'. A red arrow labeled '1' points to the 'Region' dropdown. Another red arrow labeled '2' points to the 'QUICK SETUP' button. The main content area also features a preview of an integration flow diagram with various nodes and connections. On the far right, there's a sidebar titled 'Tips and guidance' with links like 'set up Application Integration', 'Application Integration overview', etc., and a 'Resources' section with links to 'Pricing' and 'Quotas and limits'.

Google Cloud Platform (GCP)

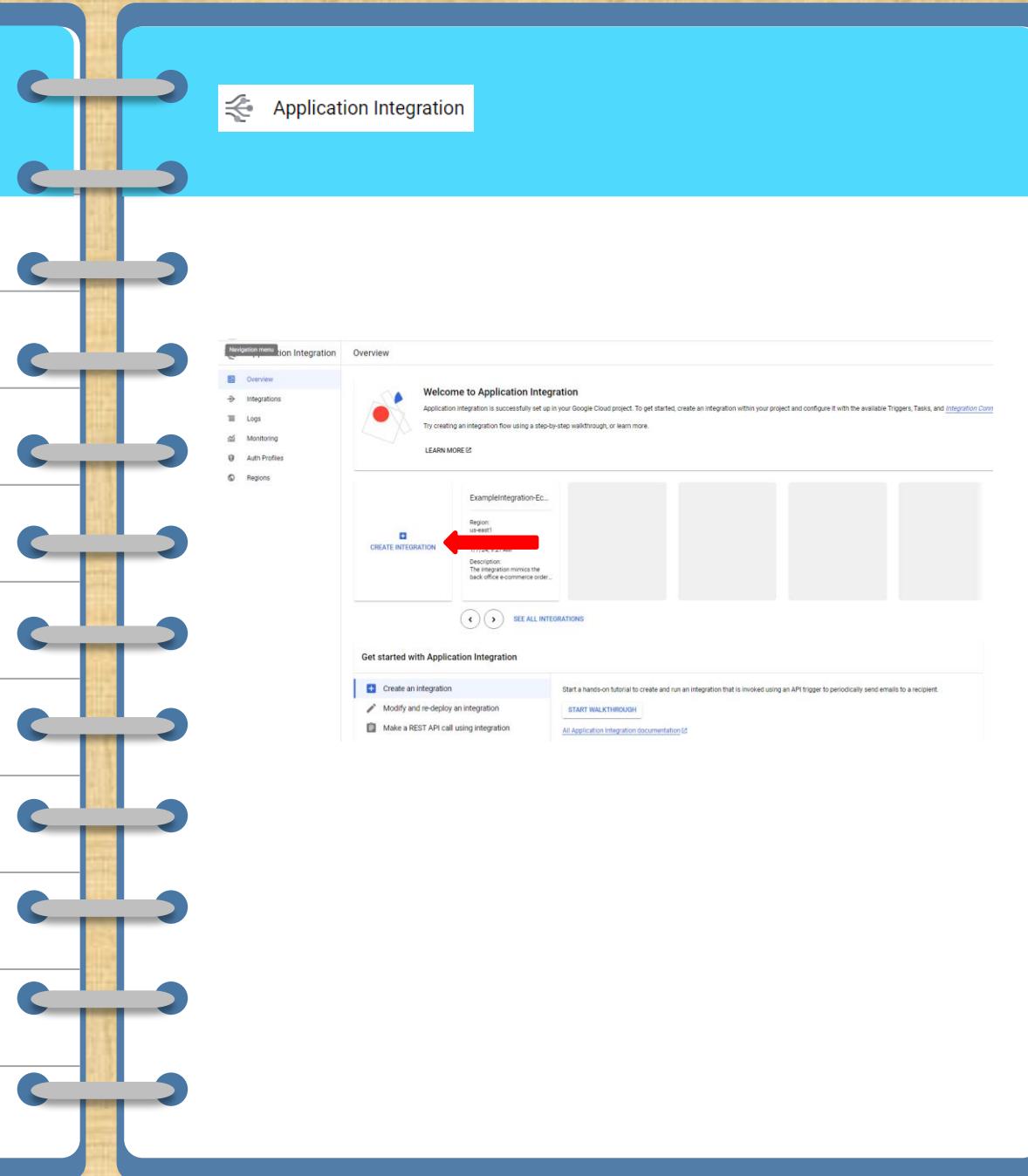
STEP 8: ALERTING

It will take a few moments while the App

Integration service gets deployed and then

you will see this screen.

Click the **+ CREATE INTEGRATION** button



Google Cloud Platform (GCP)

STEP 8: ALERTING

1. Type 'IOT-CLASS' for the Integration name
2. Enter a description like the one in the screen shot
3. Ensure that the Region is set to 'us-east1'
4. Click the **CREATE** button

Application Integration

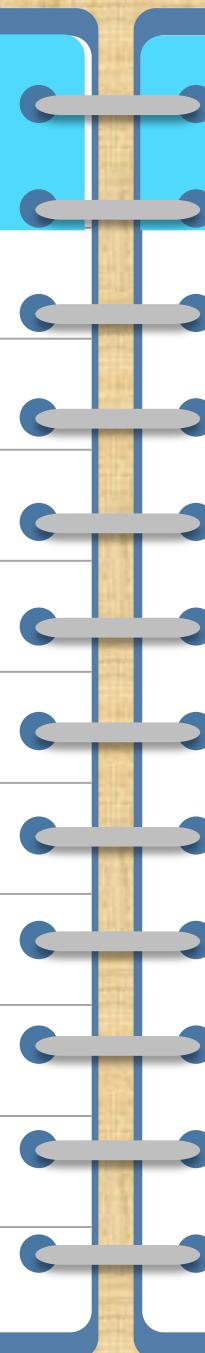
← Create Integration

Integration name (e.g. SendEmail) * IOT-CLASS 1

Description (e.g. When X happens, do Y) Send an email alert when IoT temp sensor is out of range 2

Region * us-east1 3

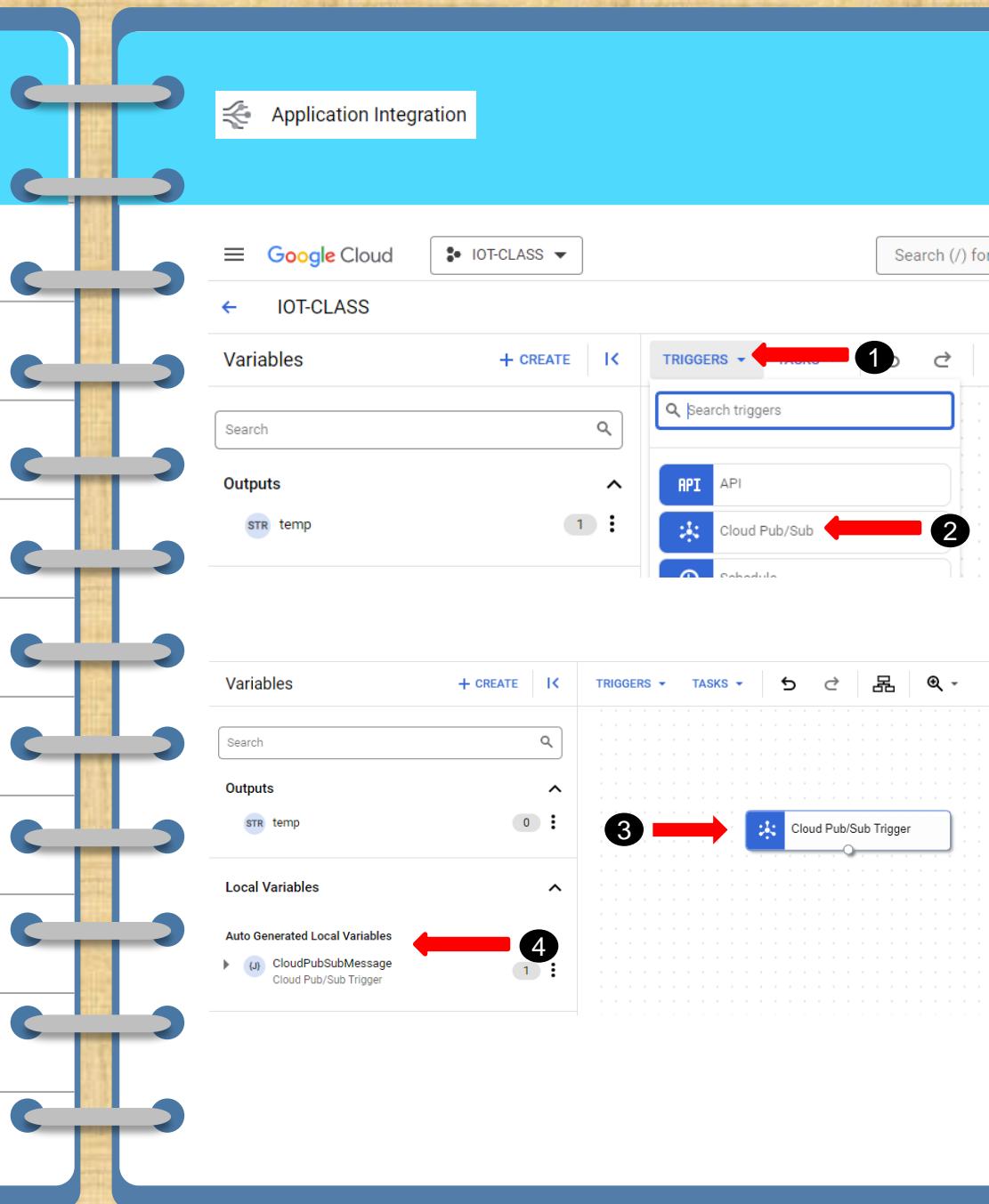
CREATE 4



Google Cloud Platform (GCP)

STEP 8: ALERTING

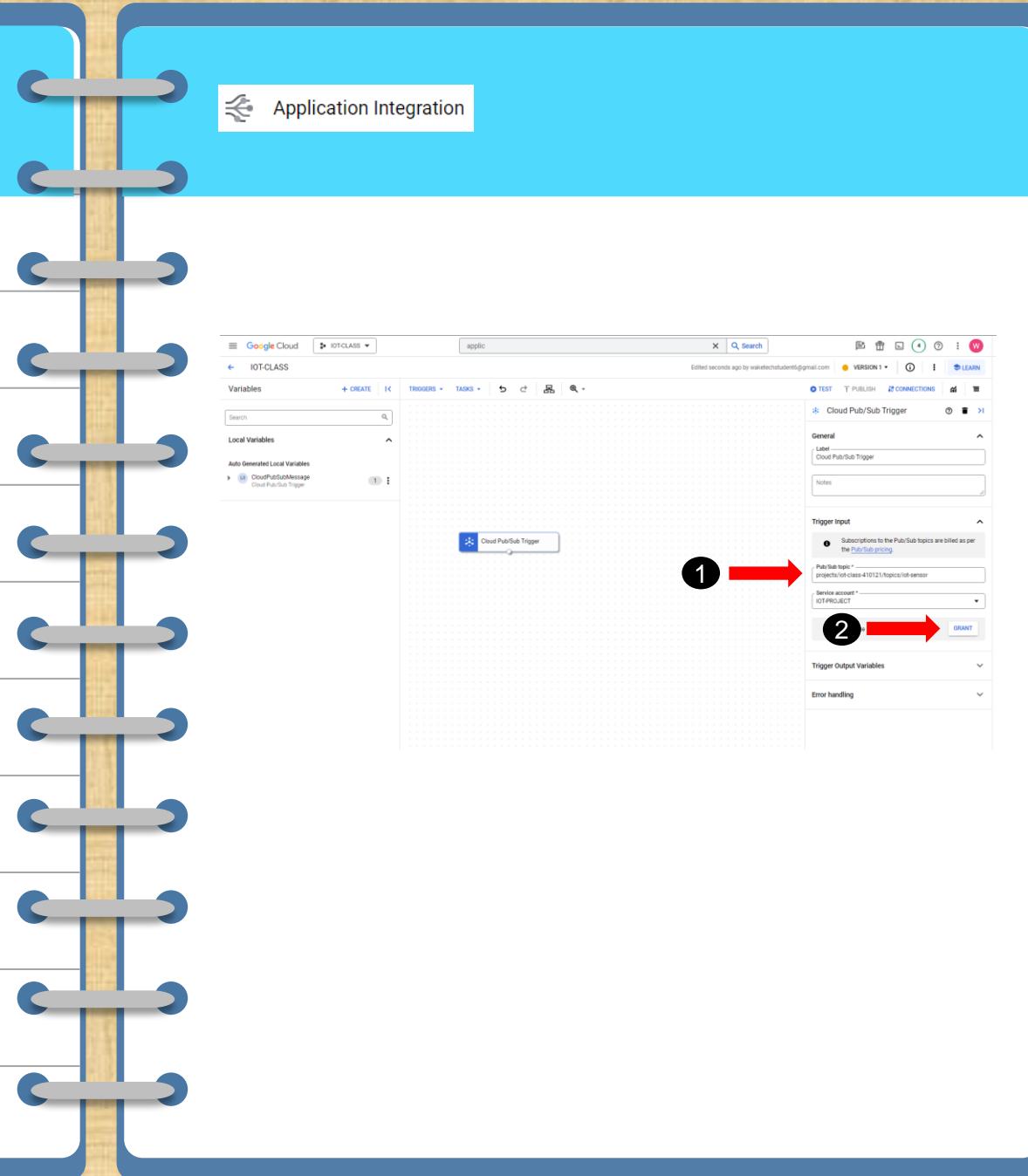
1. Select the **TRIGGERS** pick-list and choose Cloud Pub/Sub
2. Drag and drop the  trigger widget onto the App Integration workflow editor screen. You will see new Auto-Generated Local Variables appear in the left-hand pane



Google Cloud Platform (GCP)

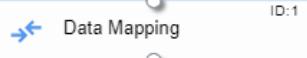
STEP 8: ALERTING

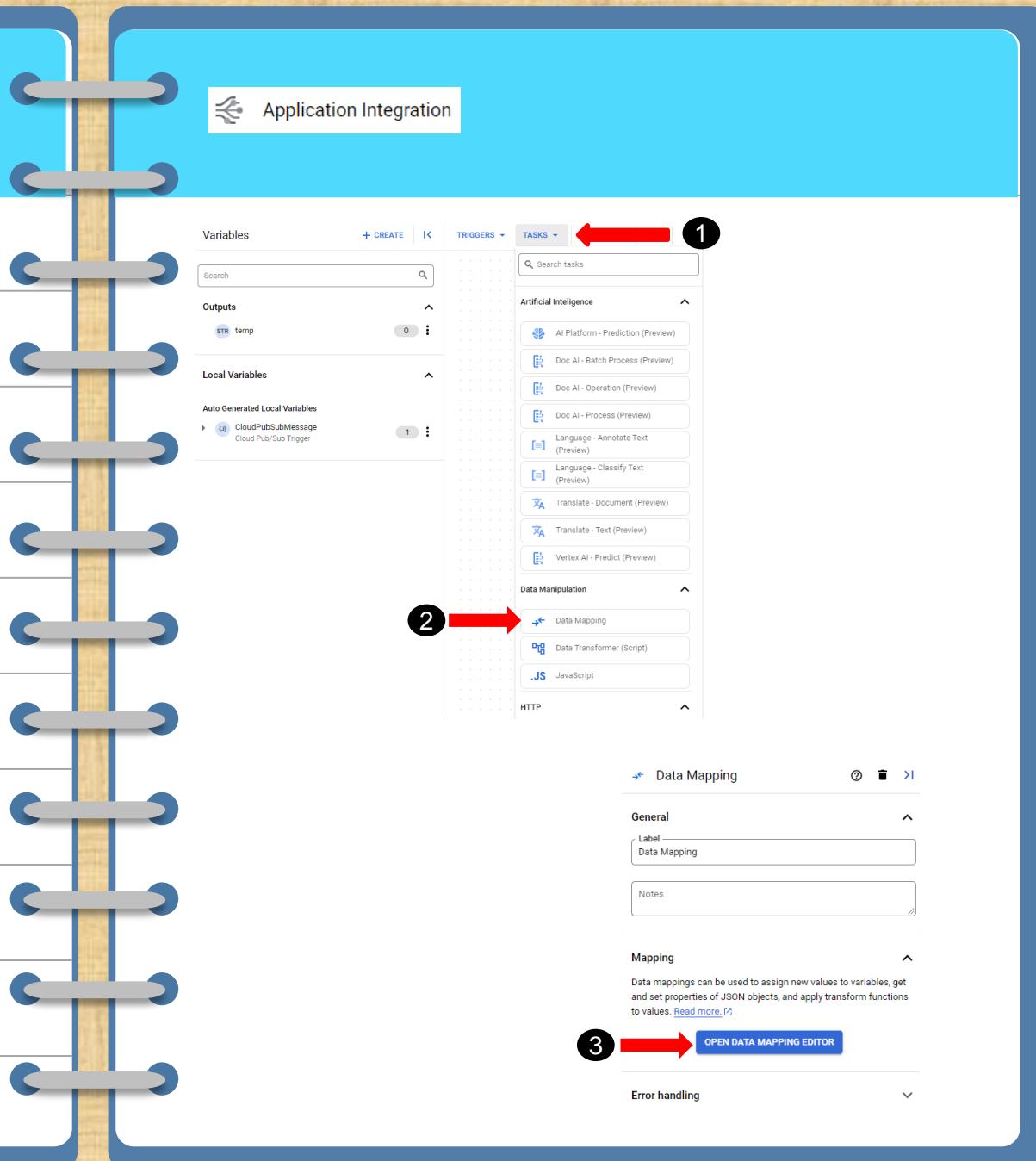
1. In the right-hand details pane, type the Pub/Sub topic name that you created in a prior step into the Trigger input field. *(It will look something like projects/iot-class-410121/topics/iot-sensor you can open another browser tab and bring up the pub/sub topic details screen easily copy the name value)*
2. Select the 'IOT-PROJECT' service account and click the **GRANT** button to assign the relevant IAM role.



Google Cloud Platform (GCP)

STEP 8: ALERTING

1. Select the **TASKS** pick-list, scroll to the 'Data Manipulation' section and choose Data Mapping
2. Drop the  widget onto the App Integration editor screen.
3. The Data Mapping details pane will appear on the right - Click on the **OPEN DATA MAPPING EDITOR** button



Google Cloud Platform (GCP)

STEP 8: ALERTING

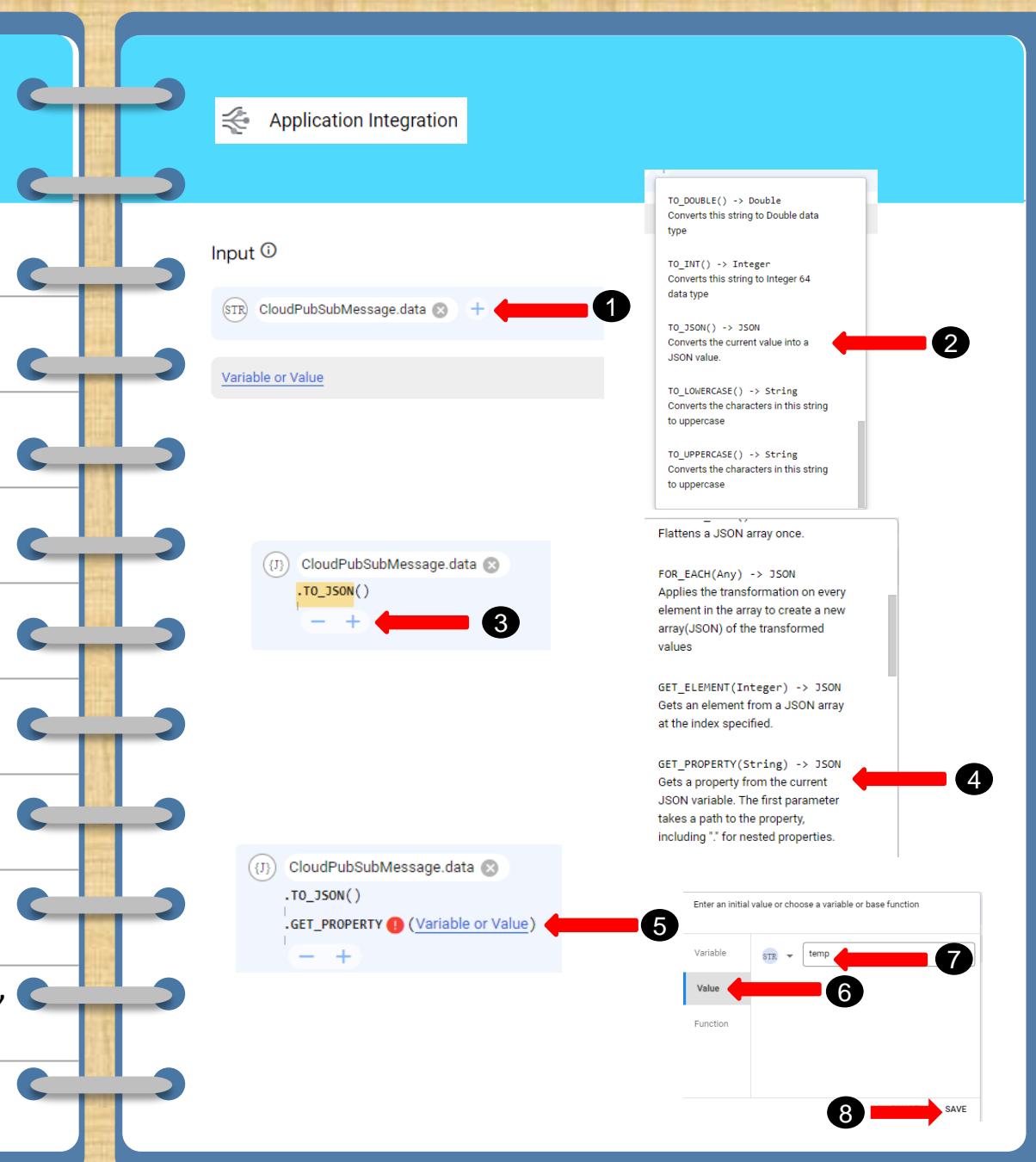
1. In the left panel under 'Local Variables' click the down-arrow to expand the CloudPubSubMessage variable and then click and drag the 'data' variable into the 'Input' section of the Data Mapping Task Editor

The screenshot shows the 'Data Mapping Task Editor' interface in GCP. On the left, under 'Local Variables', the 'CloudPubSubMessage' variable is expanded, revealing its properties: 'data', 'attributes', 'messageId', 'publishTime', and 'orderingKey'. A red arrow points from the 'data' entry in this list to its corresponding entry in the 'Input' section on the right. The 'Input' section contains a single item: 'CloudPubSubMessage.data'. The 'Variables' section at the top has a '+ CREATE' button and a search bar.

Google Cloud Platform (GCP)

STEP 8: ALERTING

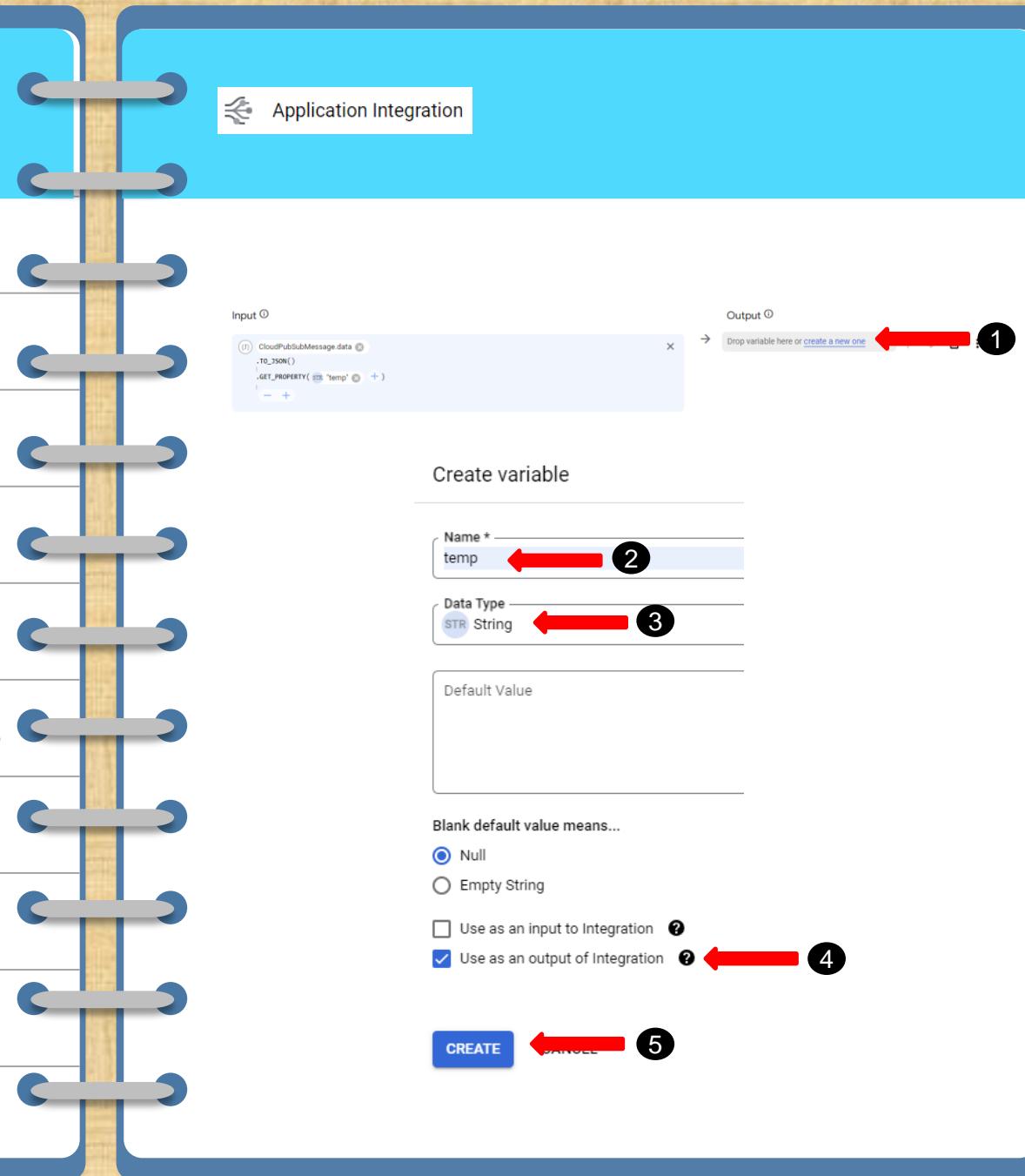
1. Click the button to the right of the CloudPubSubMessage.data variable to add a function.
2. Scroll down and choose the “.TO_JSON” function.
3. Click the button again to add another function and choose “.GET_PROPERTY(string)”.
4. Click the “(Variable or Value) link and select ‘Value’ in the left column. Name the value ‘temp’ and click ‘Save’



Google Cloud Platform (GCP)

STEP 8: ALERTING

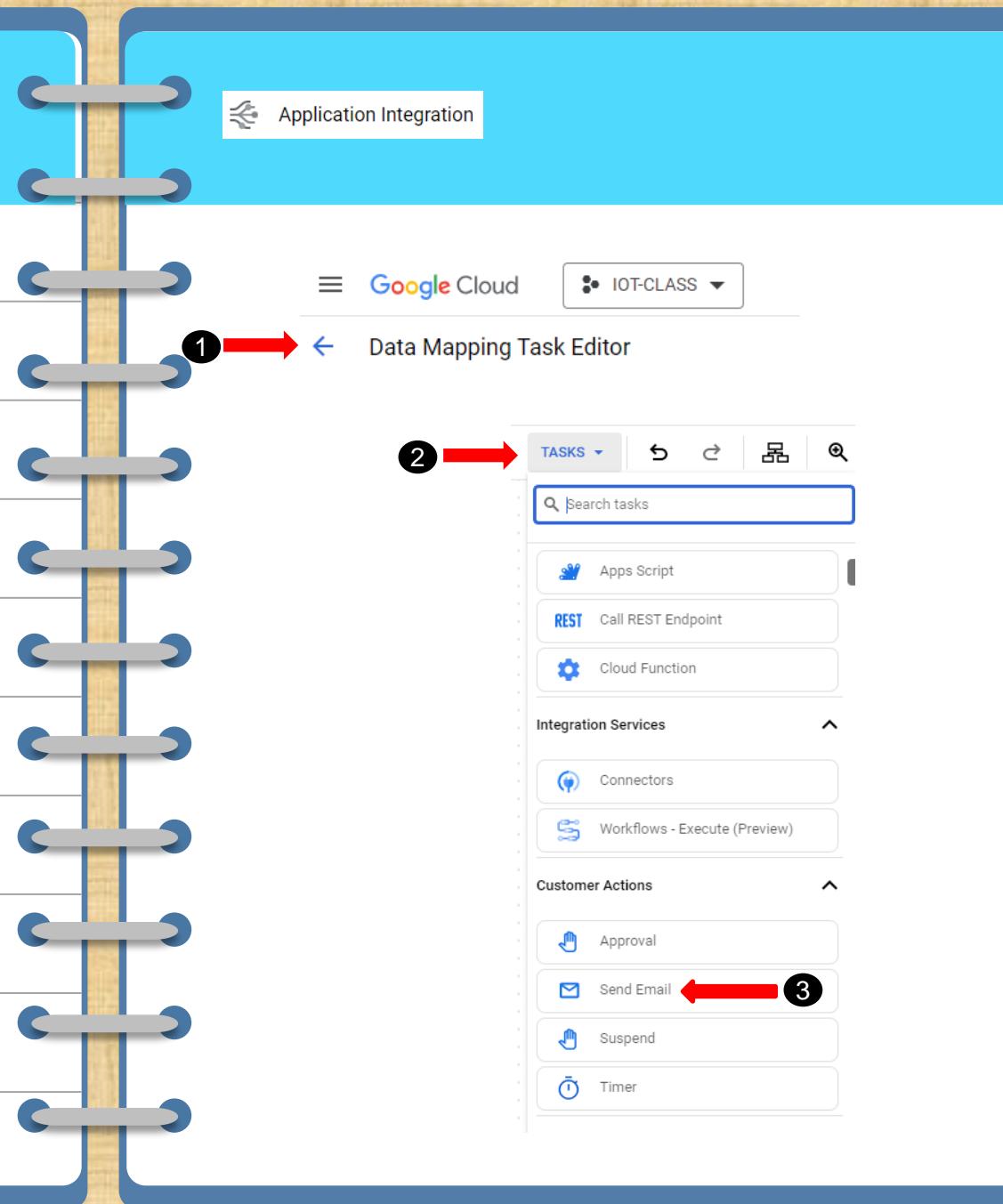
1. Click on the 'create a new one' link under the Output section of the data mapping editor
2. In the 'Create variable' window that opens, name it 'temp', set the Data Type to 'String', and check the 'Use as an ouput of integration' box
3. Click the **CREATE** button



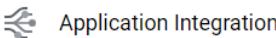
Google Cloud Platform (GCP)

STEP 8: ALERTING

1. Click the left-pointing arrow in the upper left of the screen to exit the Data Mapping Task Editor back to the main Application Integration Workflow Editor screen
2. Click the **TASKS ▾** widget once more, scroll down to the 'Customer Actions' section and select the **✉️ Send Email** task and drop it into the workflow.



Google Cloud Platform (GCP)



STEP 8: ALERTING

1. In the Send Email detail panel that appears on the right – complete the fields as shown in the screenshot. Note the following:
 - Use your own email address as the recipient
 - Be sure to include the variable name in the Body variable enclosed in dollar sign characters: \$temp\$

Send Email

General

Label: Send Email

Notes:

Task input

To Recipient(s) * 1 VARIABLE

Use a comma to separate multiple email To addresses

[s] Cc Recipient(s) VARIABLE

Use a comma to separate multiple email Cc addresses

[s] Bcc Recipient(s) VARIABLE

Use a comma to separate multiple email Bcc addresses

Subject * 2

Body Format * 3

Body in Plain Text

Message in plain text/HTML format

Google Cloud Platform (GCP)

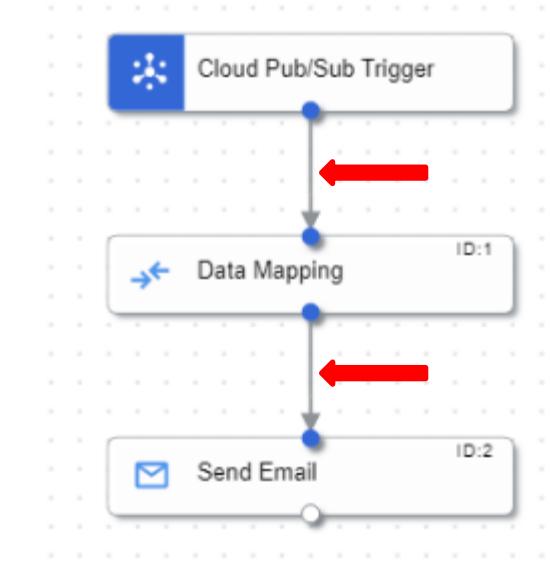
Application Integration

STEP 8: ALERTING

Now link the tasks to build the workflow by clicking on the circle at the bottom of the Cloud Pub/Sub Trigger task and drag the connector down to the Data Mapping Task. Do the same thing to connect the Data Mapping task to the Send Email task.

These arrows are referred to as 'Edge

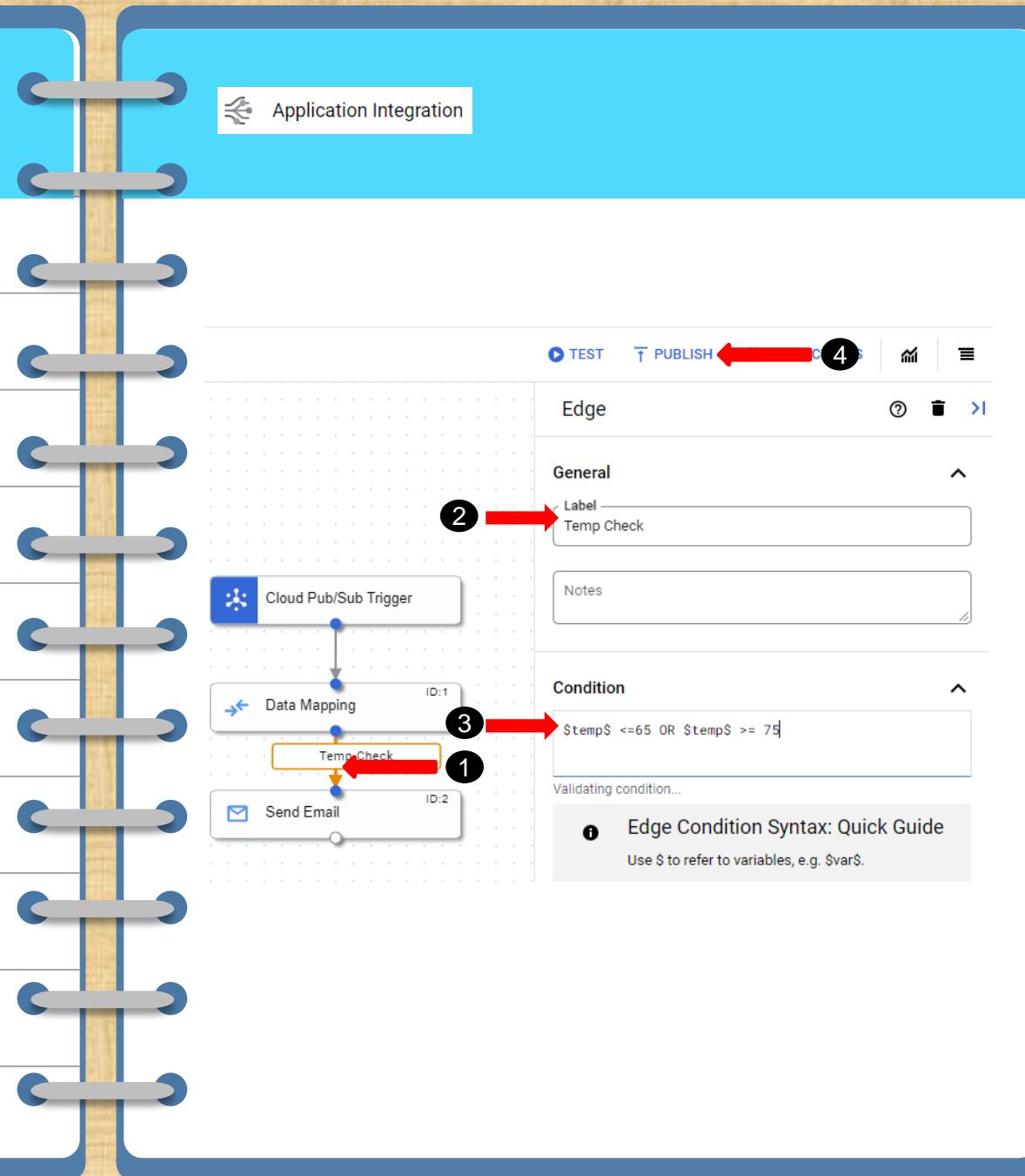
Connectors' between the workflow elements



Google Cloud Platform (GCP)

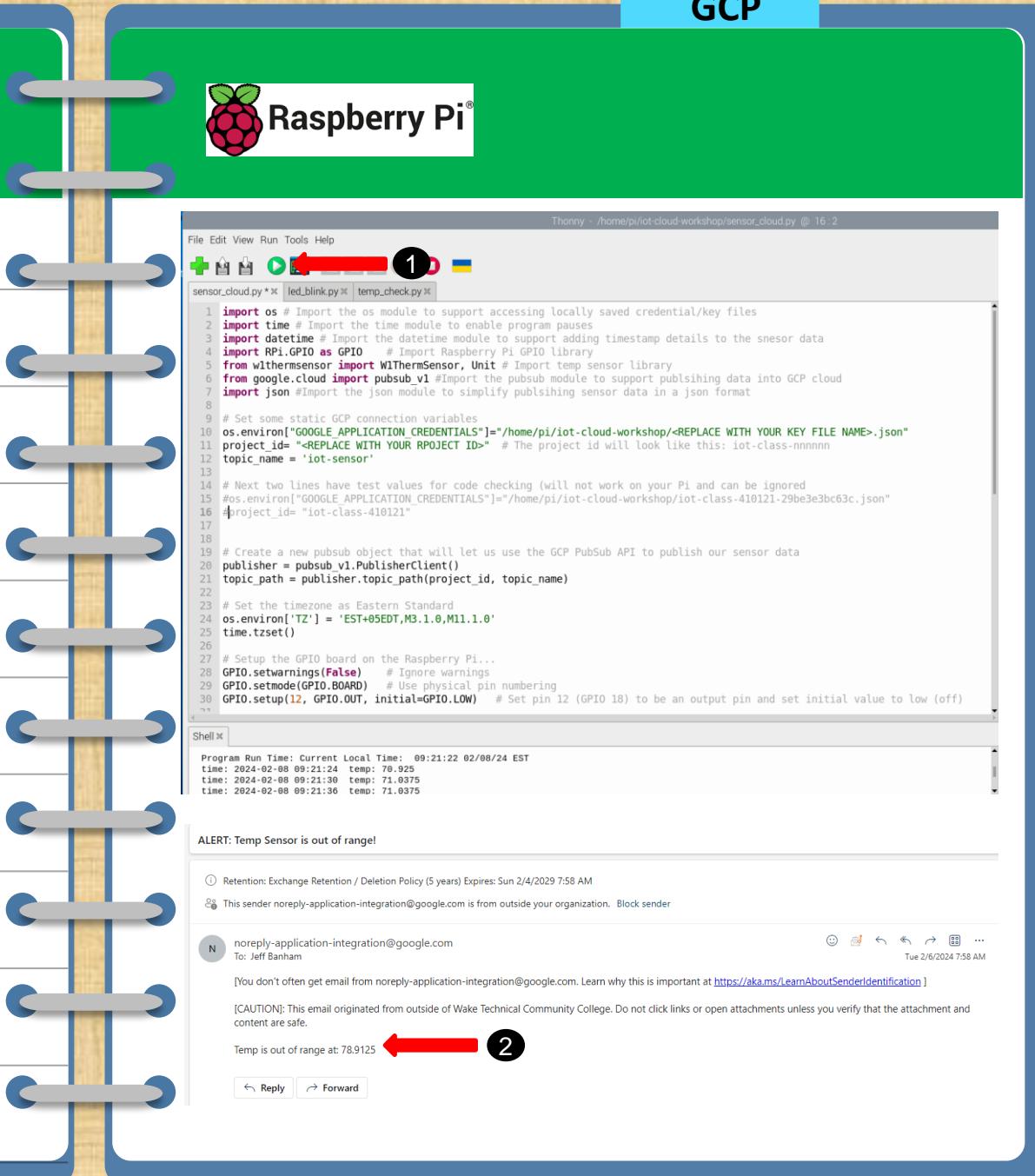
STEP 8: ALERTING

1. Click on the Edge connector line between the Data Mapping and Send Email widgets and its properties pane will appear on the right
2. Set the Label value to 'Temp Check'
3. Set the Condition value to:
 $\$temp\$ \leq 65 \text{ OR } \$temp\$ \geq 75$
4. Click the **PUBLISH** button to activate the workflow.



Raspberry Pi Code

1. In the Thonny IDE, if it's not already open, open the file named /home/pi/iot-cloud-workshop/sensor_cloud
 2. Click the  button
 3. Try raising the temp (by holding your fingers on the sensor)
 4. Check if you are receiving the email alerts when temp exceeds limit
 5. Press CTRL+C to stop the program





CHECKPOINT

- A vertical stack of ten white rectangular cards. Each card features a blue circular hole near its top and bottom edges, and a central vertical slot. The cards are arranged one on top of the other, creating a pattern of alternating blue circles and central slots.

Everyone can successfully generate an email alert when temp exceeds coded limit

Google Cloud Platform (GCP)

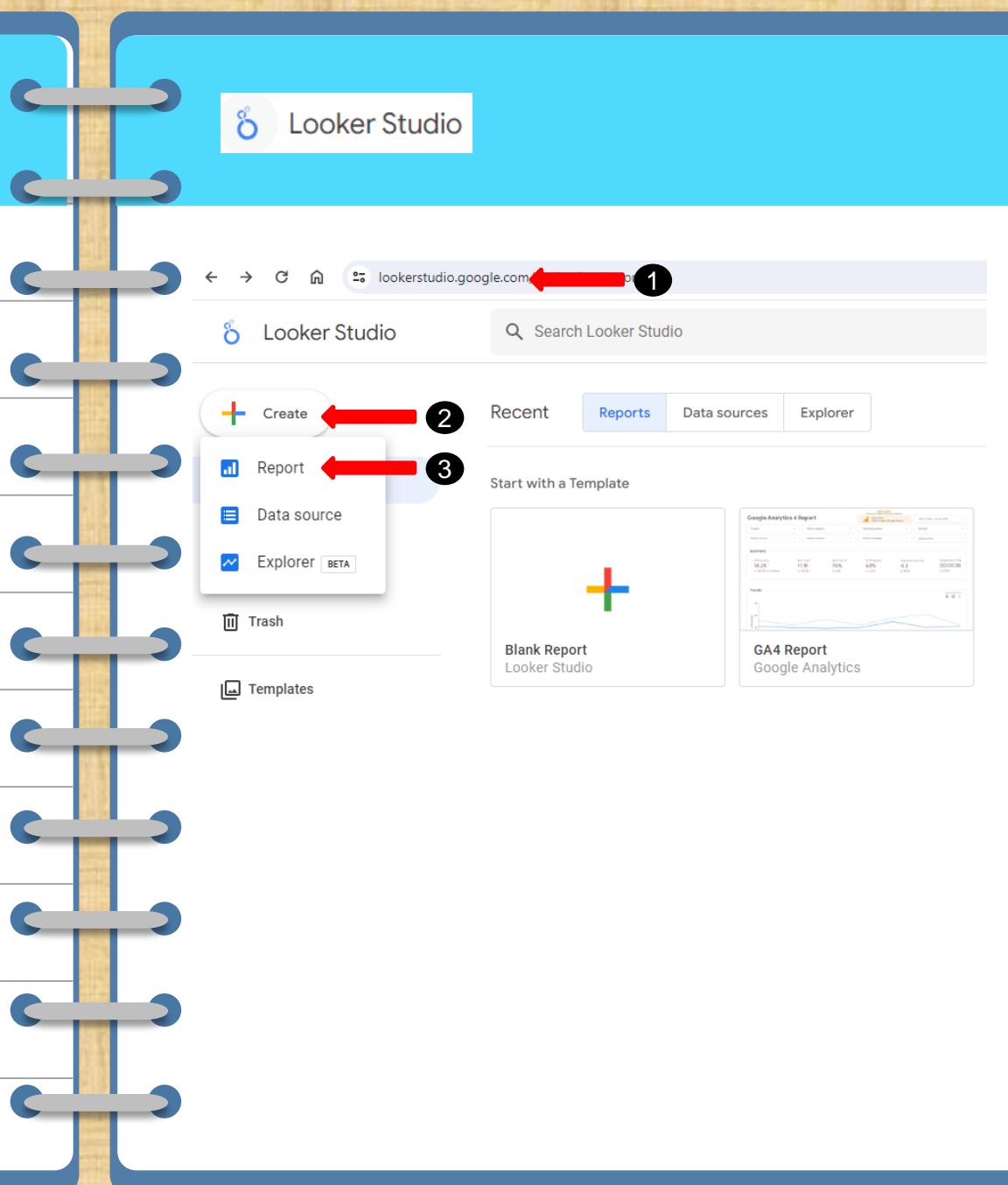
STEP 10: REPORTING

1. In a new browser tab, open

<https://lookerstudio.google.com>

2. Click the  button and

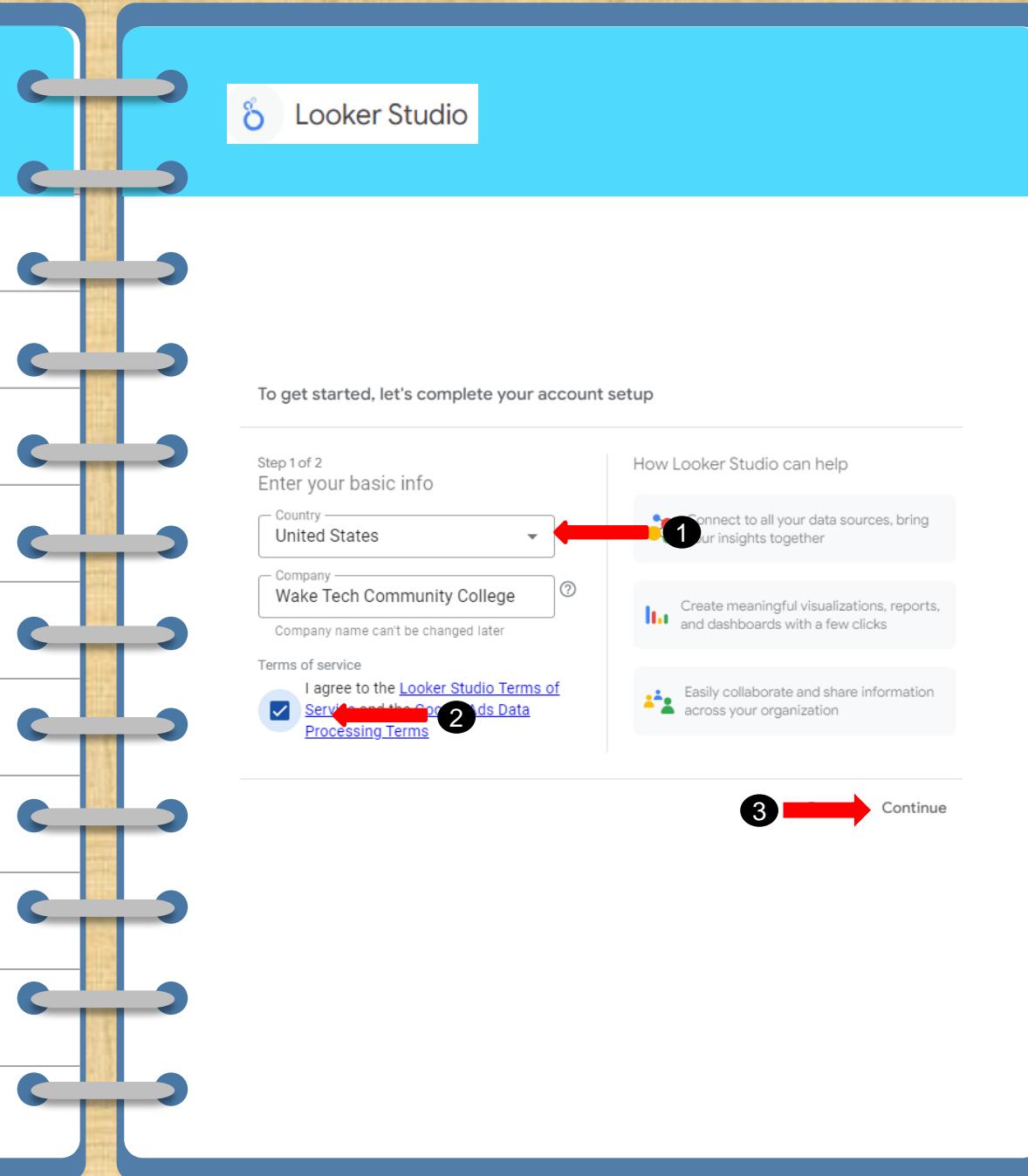
then select  Report



Google Cloud Platform (GCP)

STEP 10: REPORTING

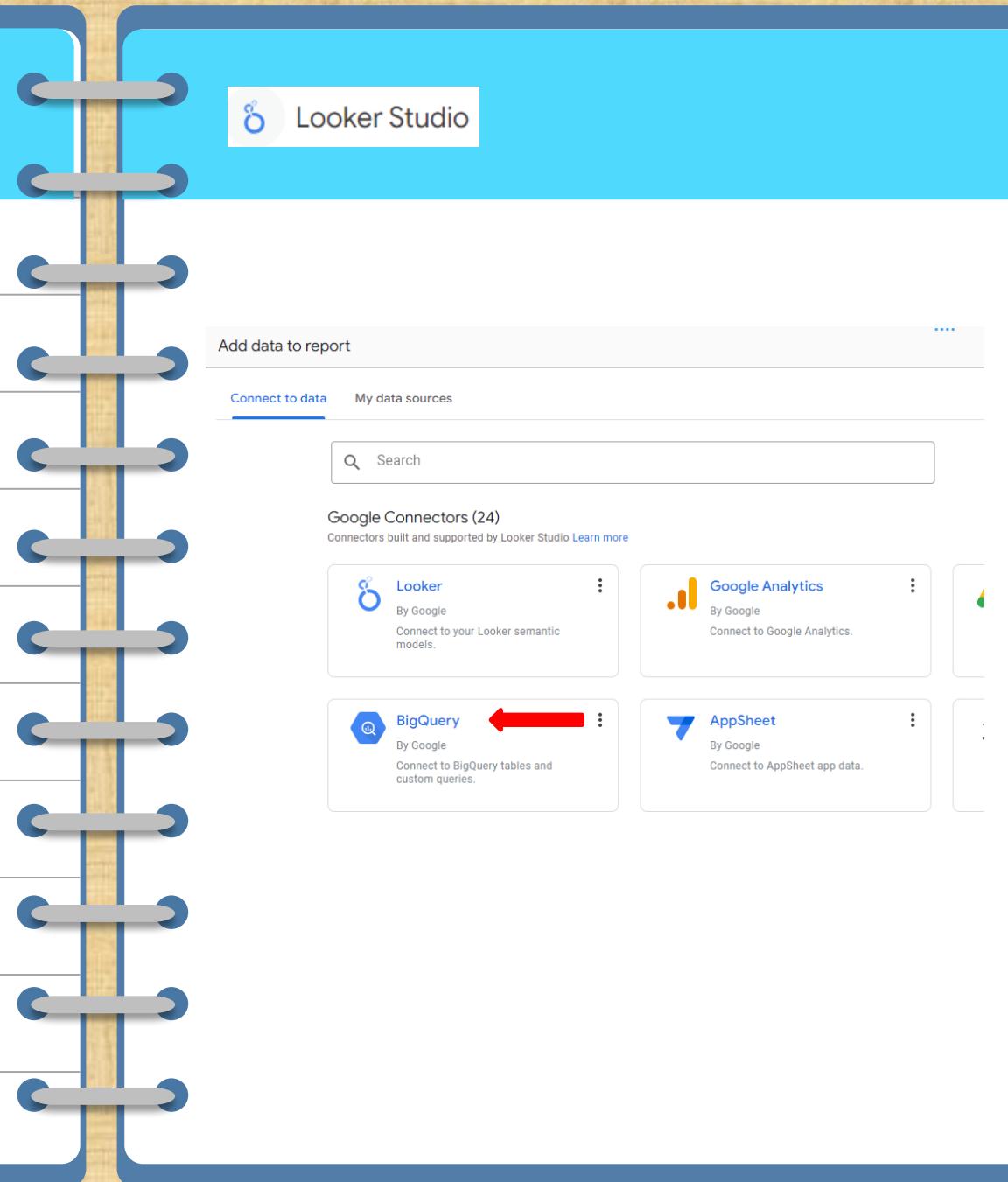
You'll be prompted to complete setting up your Looker Studio account. Fill in the form and agree to the service terms, and click 'Continue' – On the following screen (not shown), answer the 3 questions and click 'Continue' again



Google Cloud Platform (GCP)

STEP 10: REPORTING

On the 'Add data to report' screen,
select the BigQuery connector



Google Cloud Platform (GCP)

STEP 10: REPORTING

Click the 'Authorize' button to allow Looker Studio to connect to BQ and then select your account



Add data to report

Make your BigQuery reports load even faster with BigQuery BI Engine. [Learn More](#)

BigQuery

By Google

BigQuery is Google's fully managed, petabyte scale, low-cost analytics data warehouse. BigQu... of data. Those queries are charged to the credit card of the billing project.

[LEARN MORE](#) [REPORT AN ISSUE](#)

Authorization

Looker Studio requires authorization to connect to your BigQuery projects.

AUTHORIZE



1

Sign in with Google



Choose an account

to continue to [Looker Studio](#)

WakeTech Student
waketechstudent6@gmail.com

Use another account



2

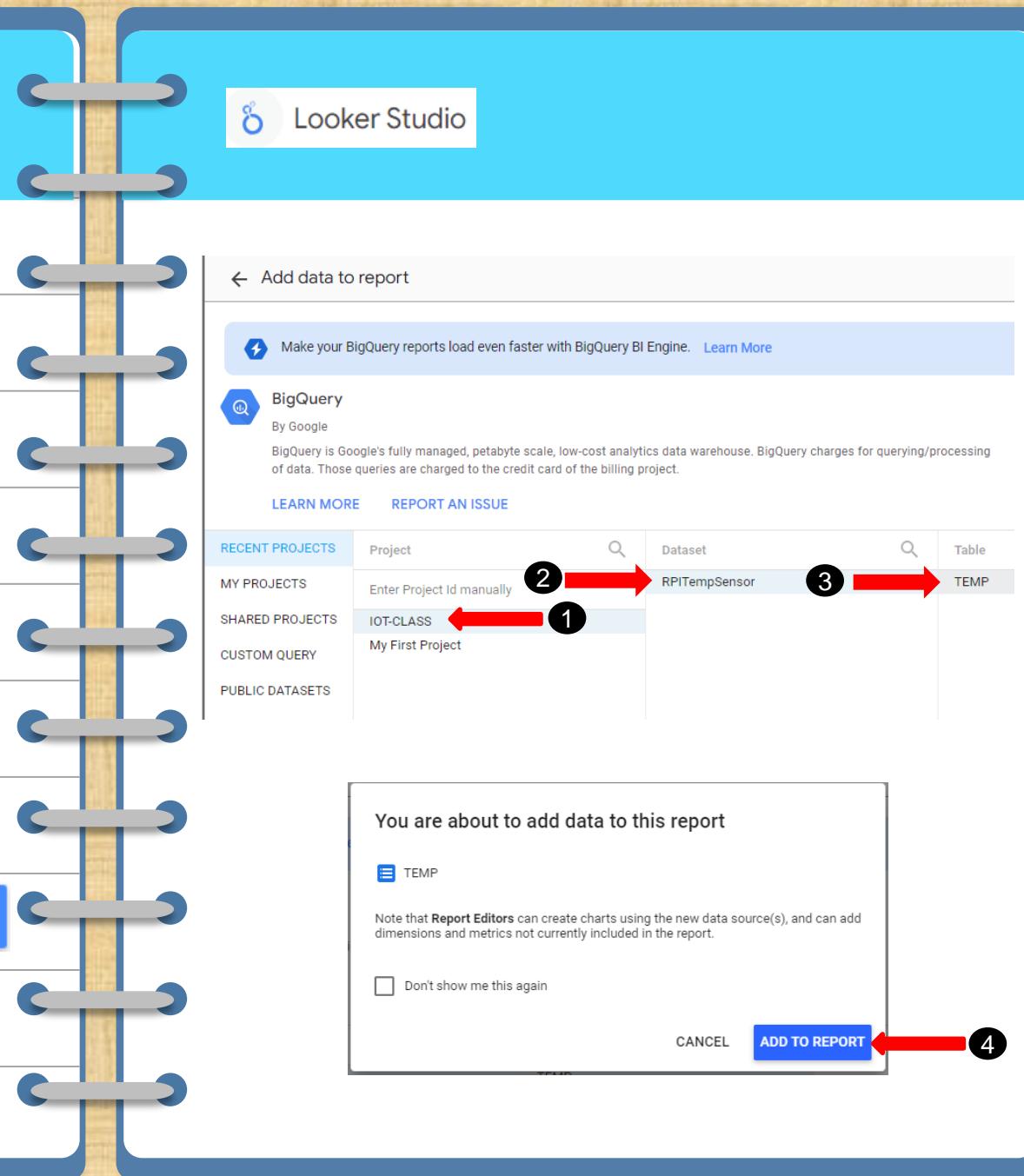
Google Cloud Platform (GCP)

STEP 10: REPORTING

Select your 'IOT-CLASS' project,
RPITempSensor Dataset, and TEMP
table and then click the **Add**

button in the lower right (not shown in
screen shot)

On the next screen, click the **ADD TO REPORT**
button



Google Cloud Platform (GCP)

STEP 10: REPORTING

Click the 'Add a chart' button and then
select the first time series chart type



Untitled Report

File Edit View Insert Page Arrange Resource Help

← → 🔍 Add page Add data **Add a chart** ①

+ Add quick filter

The screenshot shows a table with 10 rows of data and a sidebar with chart selection options. The table has columns for 'time' and 'Record Count'. The sidebar includes sections for 'Scorecard', 'Time series', 'Bar', 'Pie', and 'Google Maps'. A red arrow points to the 'Add a chart' button in the top navigation bar, and another red arrow points to the 'Time series' icon in the sidebar.

	time	Record Count
1.	Jan 7, 2024, 5:11:06PM	1
2.	Jan 7, 2024, 5:11:32PM	1
3.	Jan 7, 2024, 5:11:41PM	1
4.	Jan 7, 2024, 5:11:50PM	1
5.	Jan 8, 2024, 4:57:36PM	1
6.	Jan 8, 2024, 4:57:45PM	1
7.	Jan 7, 2024, 3:29:27PM	1
8.	Jan 7, 2024, 3:29:45PM	1
9.	Jan 7, 2024, 3:51:17PM	1

Total: 1,168 Sessions: 69.3K

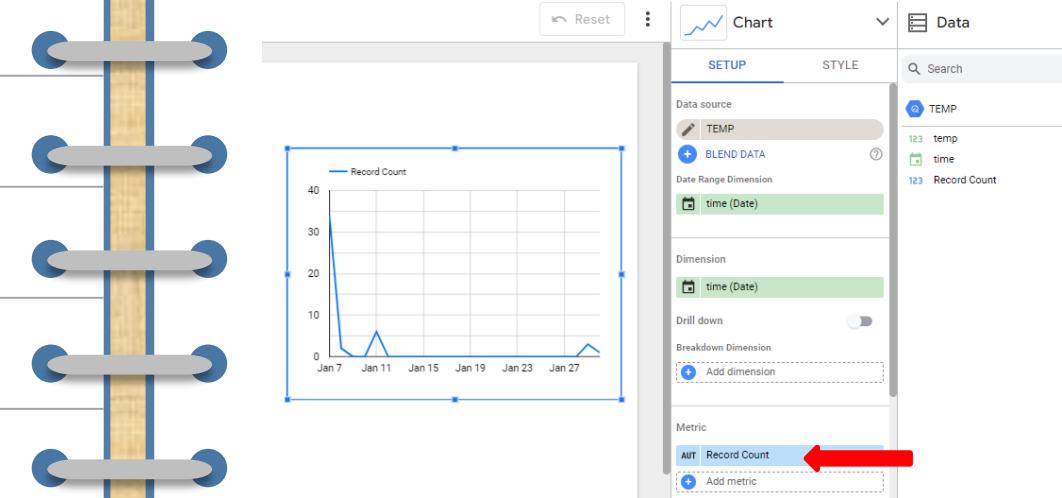
Time series Bar Pie Google Maps

Google Cloud Platform (GCP)



STEP 10: REPORTING

Note that the default chart settings just map the record count metric over time.



Google Cloud Platform (GCP)

STEP 10: REPORTING

Drag the 'temp' field from the 'Data' column into the 'Metric' section of the Chart column. Click on the metric type field and change the value from 'Sum' to 'Max'. Click the 'X' next to the 'Record Count' field under Metric to remove it (not shown in screen shot)

The screenshot shows the Looker Studio interface with the following configuration:

- Chart:** The main view where the chart is being built.
- Setup:** Tab selected in the top navigation bar.
- Data:** Tab in the top navigation bar.
- Data source:** TEMP
- Date Range Dimension:** time (Date)
- Dimension:** time (Date)
- Metric:** MAX temp
- Additional Metrics:** (None listed)
- Metric Sliders:** (None listed)

A red arrow points from the text in the adjacent box to the 'temp' field in the Metric section. A red circle with the number 1 points to the 'temp' field. A red circle with the number 2 points to the 'Record Count' field, which is not visible in the screenshot but mentioned in the instructions.

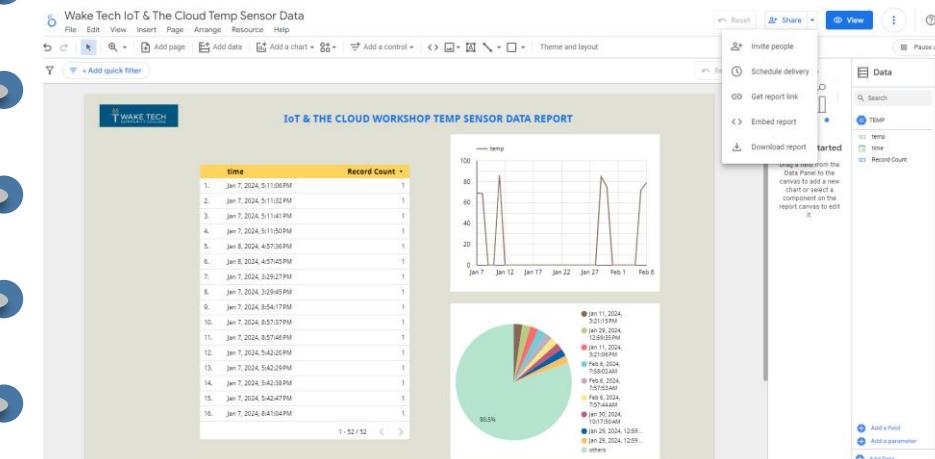
Google Cloud Platform (GCP)



STEP 10: REPORTING

Depending on time, play around with the various settings and see what you can come up with for a report.

Note the options for sharing it also!



WHAT'S NEXT?

- Take your kit home!
- Try other sensors
- Check out AWS & Azure
- Check out sample projects on the internet:
 - [Tom's Hardware](#)
 - [PC Magazine](#)
 - [Raspberry Pi Forums](#)

KEEP LEARNING

COLLABORATE

STAY IN TOUCH!

Please take a moment
to complete this short
feedback form – it will
help us to improve the
workshop for future
students!

[https://forms.office.com
m/r/UKRPBbb81X](https://forms.office.com/r/UKRPBbb81X)





This material is based upon work supported by the National Science Foundation Advanced Technical Education grant program, Enhancing Preparation of Students for Technical Careers in Cloud Computing Technologies, DUE-2055288.

The opinions, findings, and conclusions or recommendations expressed are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.



WAKE TECH
COMMUNITY COLLEGE

This material is the property of Wake Tech Community College and should not be reproduced or shared without prior written permission.

Please contact Jeff Banham
jkbanham@waketech.edu for more info.