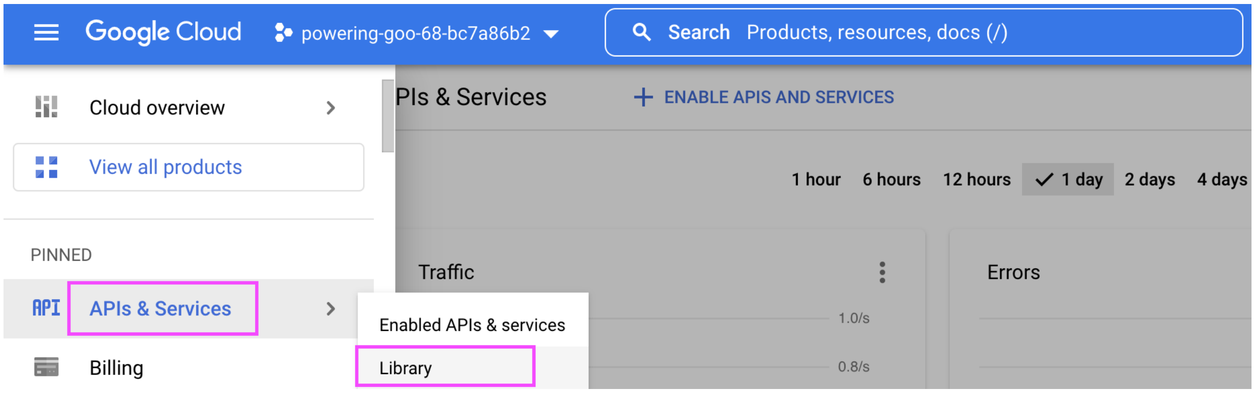
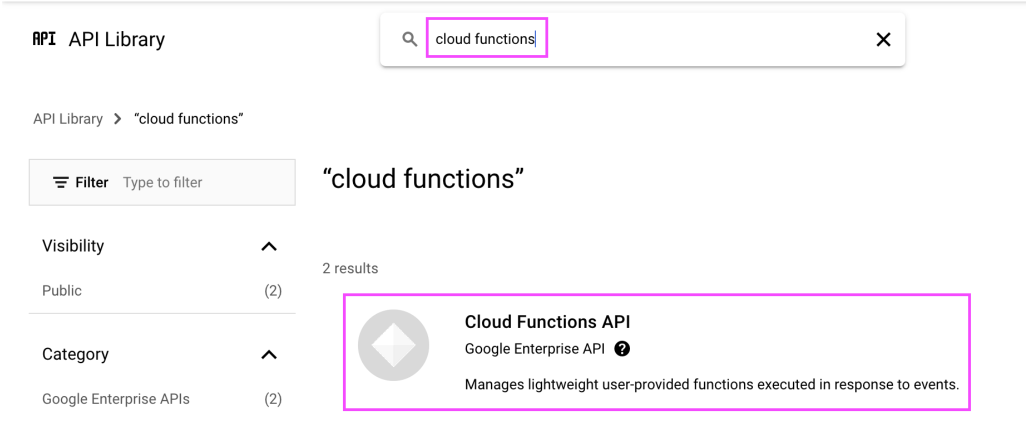
1. Enable the needed APIs: Cloud Functions, Cloud Vision, & Cloud Translation, Cloud Build
    a. “Navigation menu”; “APIs and Services”; “Library”



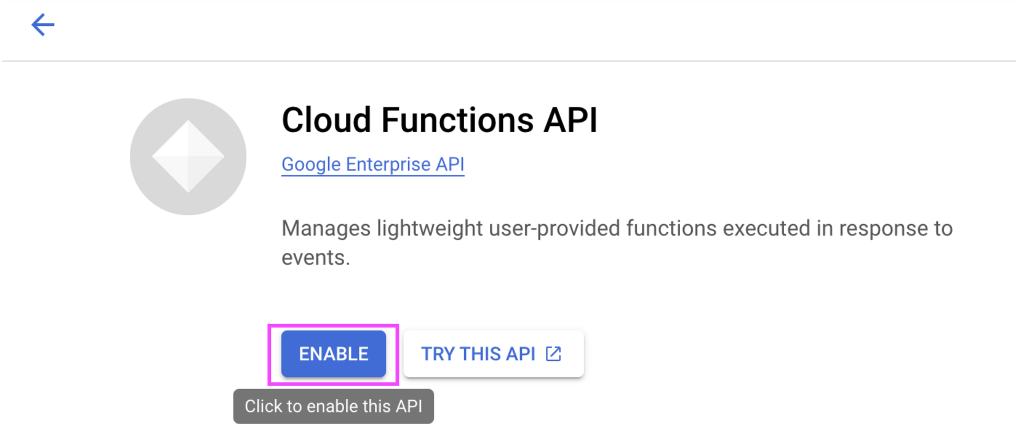
    b. Type each API name into the search bar individually, and “enable” the APIs



“cloud functions”

2 results

Cloud Functions API
Google Enterprise API
Manages lightweight user-provided functions executed in response to events.



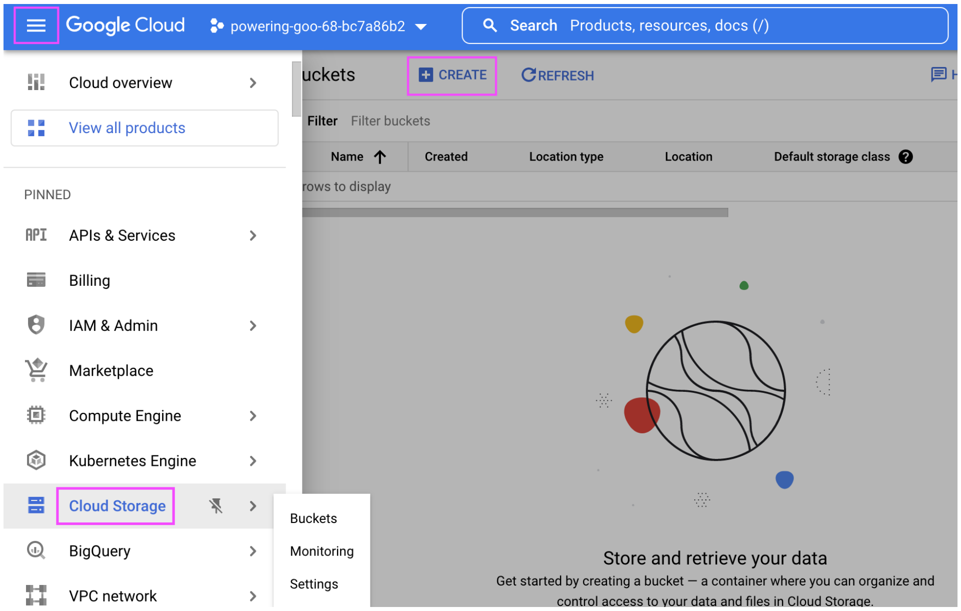
Cloud Functions API

Google Enterprise API

Manages lightweight user-provided functions executed in response to events.

2. Create two Cloud Storage buckets- one to hold the images that we upload into our workflow that will trigger the cascade of functionalities, and another to hold the end results of our work

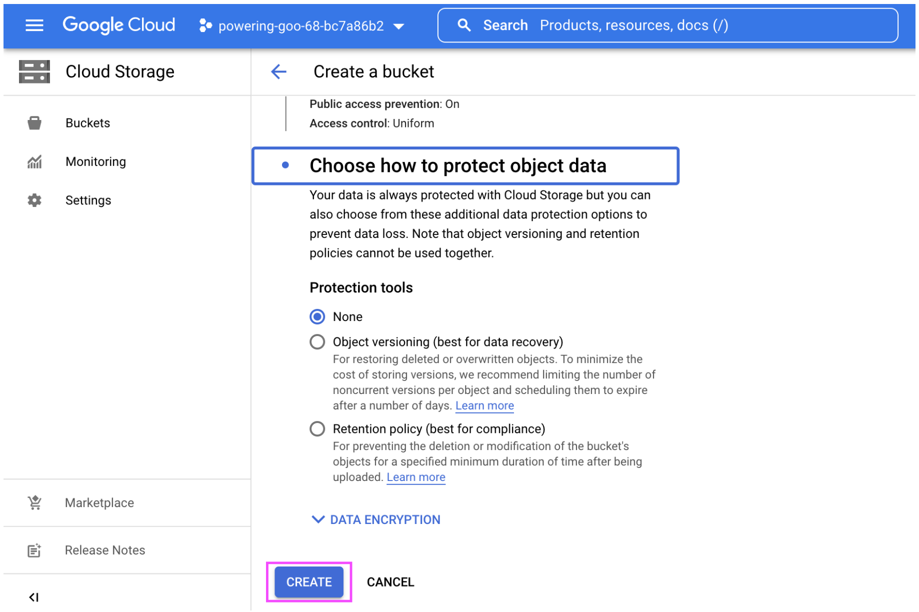
a. “Navigation Menu”; “Cloud Storage”; “Create”



Store and retrieve your data

Get started by creating a bucket — a container where you can organize and control access to your data and files in Cloud Storage.

b. Assign the first bucket a “name” and “region”, and click “Create”



Public access prevention: On
Access control: Uniform

Choose how to protect object data

Your data is always protected with Cloud Storage but you can also choose from these additional data protection options to prevent data loss. Note that object versioning and retention policies cannot be used together.

Protection tools

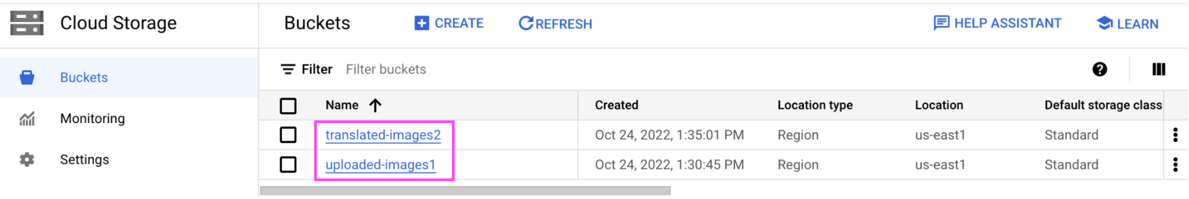
● None

○ Object versioning (best for data recovery)
For restoring deleted or overwritten objects. To minimize the cost of storing versions, we recommend limiting the number of noncurrent versions per object and scheduling them to expire after a number of days. Learn more

○ Retention policy (best for compliance)
For preventing the deletion or modification of the bucket's objects for a specified minimum duration of time after being uploaded. Learn more

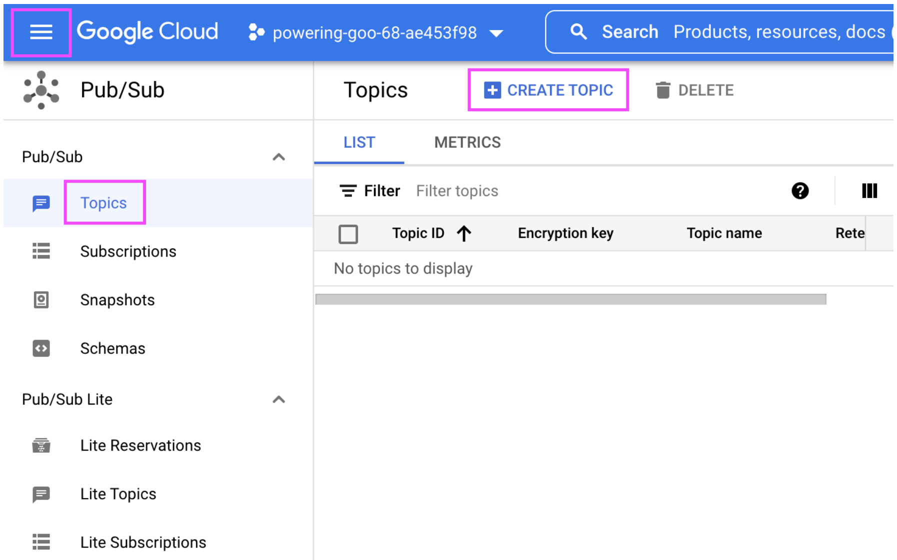
c. Go back to the Cloud Storage bucket page again to create the second bucket. Click “Create”

d. Assign the second bucket a “name” and “region”, and click “Create”



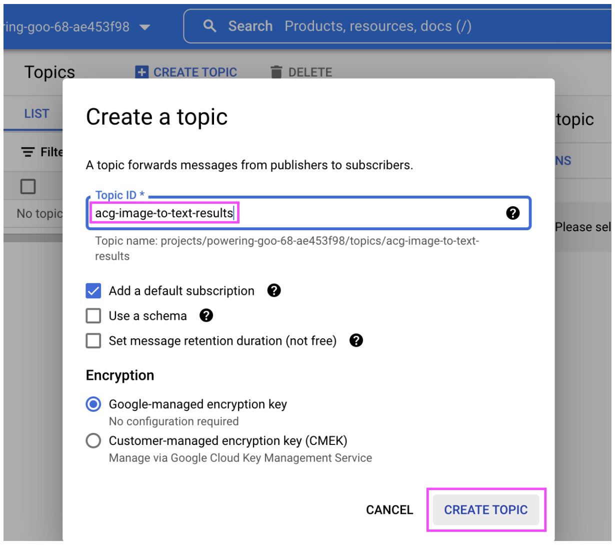
Name	Created	Location type	Location	Default storage class
translated-images2	Oct 24, 2022, 1:35:01 PM	Region	us-east1	Standard
uploaded-images1	Oct 24, 2022, 1:30:45 PM	Region	us-east1	Standard

3. Create two Pub/Sub topics- one to hold the image to text results, and another to hold the translations created by the Translations API
    a. “Navigation menu”; “Pub/Sub”; “Topics”; “Create Topic”



No topics to display

    b. Give each topic a “Topic ID” and click “Create Topic”



Create a topic

A topic forwards messages from publishers to subscribers.

Topic ID *: acg-image-to-text-results

Topic name: projects/powering-goo-68-ae453f98/topics/acg-image-to-text-results

☑ Add a default subscription

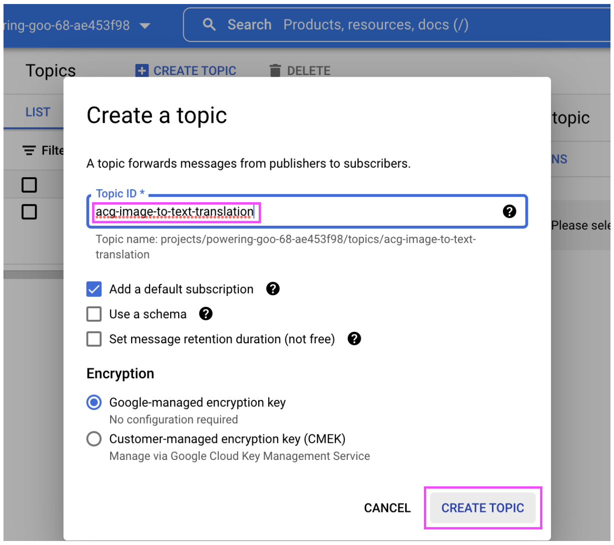
☐ Use a schema

☐ Set message retention duration (not free)

Encryption

● Google-managed encryption key
No configuration required

○ Customer-managed encryption key (CMEK)
Manage via Google Cloud Key Management Service



Create a topic

A topic forwards messages from publishers to subscribers.

Topic ID *: acg-image-to-text-translation

Topic name: projects/powering-goo-68-ae453f98/topics/acg-image-to-text-translation

☑ Add a default subscription

☐ Use a schema

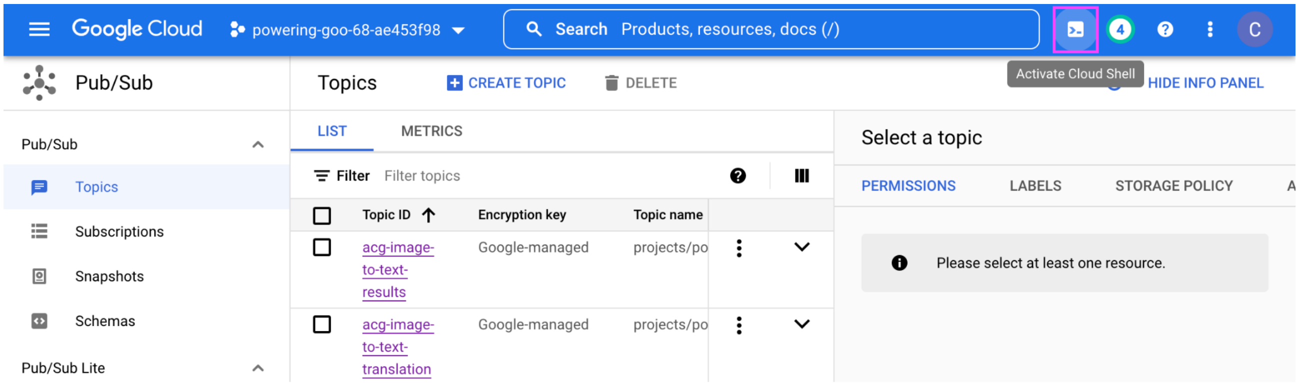
☐ Set message retention duration (not free)

Encryption

● Google-managed encryption key
No configuration required

○ Customer-managed encryption key (CMEK)
Manage via Google Cloud Key Management Service

4. Retrieve the files from a GitHub repository and configure the code using Cloud Shell
    a. “Activate Cloud Shell”



	Topic ID ↑	Encryption key	Topic name
□	acg-image-to-text-results	Google-managed	projects/po
□	acg-image-to-text-translation	Google-managed	projects/po

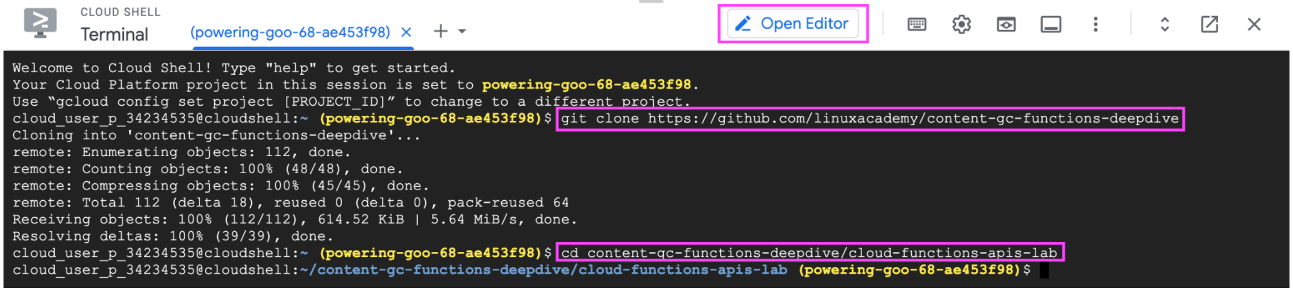
Please select at least one resource.

b. Retrieve code from the GitHub repository using the following command

   i. git clone https://github.com/linuxacademy/content-gc-functions-deepdive

c. Change the directory to our lab's folder using the following command

   i. cd content-gc-functions-deepdive/cloud-functions-apis-lab



```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to powering-goo-68-ae453f98.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
cloud_user_p_34234535@cloudshell:~ (powering-goo-68-ae453f98)$ git clone https://github.com/linuxacademy/content-gc-functions-deepdive
Cloning into 'content-gc-functions-deepdive'...
remote: Enumerating objects: 112, done.
remote: Counting objects: 100% (48/48), done.
remote: Compressing objects: 100% (45/45), done.
remote: Total 112 (delta 18), reused 0 (delta 0), pack-reused 64
Receiving objects: 100% (112/112), 614.52 KiB | 5.64 MiB/s, done.
Resolving deltas: 100% (39/39), done.
cloud_user_p_34234535@cloudshell:~ (powering-goo-68-ae453f98)$ cd content-gc-functions-deepdive/cloud-functions-apis-lab
cloud_user_p_34234535@cloudshell:~/content-gc-functions-deepdive/cloud-functions-apis-lab (powering-goo-68-ae453f98)$
```

d. Insert the project ID into the file's code

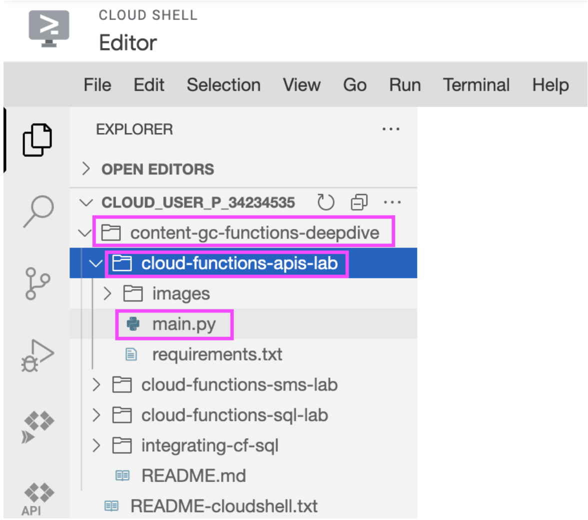
   i. Copy the project ID

   ii. "Open Editor"

   iii. Expand the "content-gc-functions-deepdive" folder

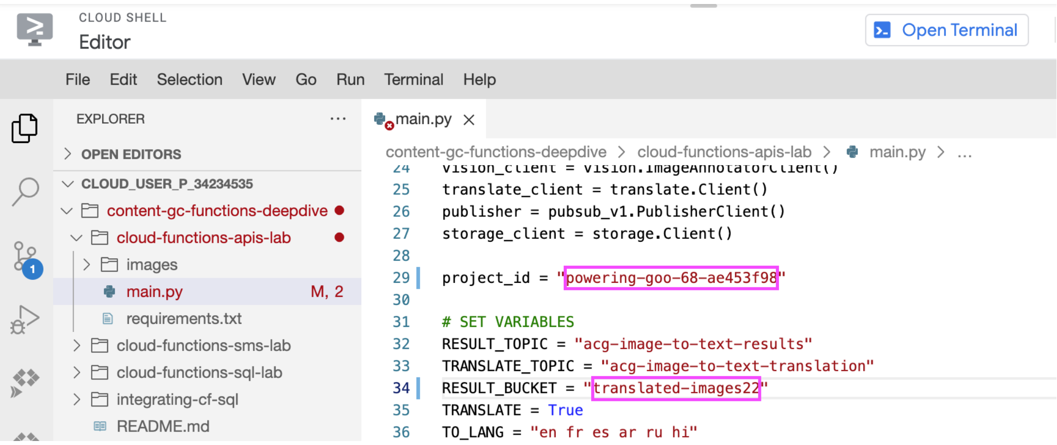
   iv. Expand the "cloud-functions-apis-lab" folder

   v. Open the main.py file



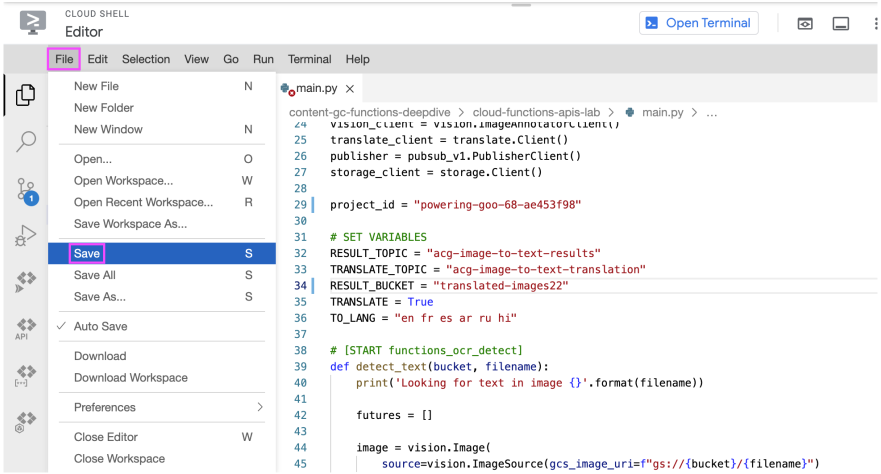
vi. Replace "[PROJECT ID]" on line 29, with the project's true ID

vii. Replace "[RESULT_BUCKET]" on line 34 with the result bucket's true name



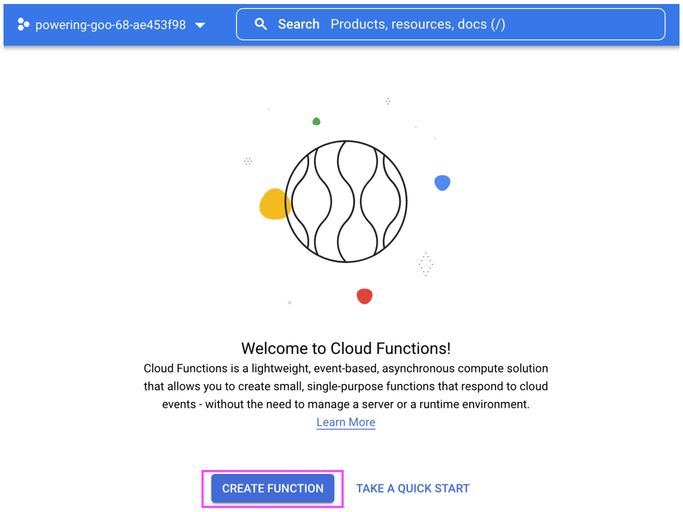
```
24  vision_client = vision.ImageAnnotatorClient()
25  translate_client = translate.Client()
26  publisher = pubsub_v1.PublisherClient()
27  storage_client = storage.Client()
28
29  project_id = "powering-goo-68-ae453f98"
30
31  # SET VARIABLES
32  RESULT_TOPIC = "acg-image-to-text-results"
33  TRANSLATE_TOPIC = "acg-image-to-text-translation"
34  RESULT_BUCKET = "translated-images22"
35  TRANSLATE = True
36  TO_LANG = "en fr es ar ru hi"
```

viii. Save the file



```
24  vision_client = vision.ImageAnnotatorClient()
25  translate_client = translate.Client()
26  publisher = pubsub_v1.PublisherClient()
27  storage_client = storage.Client()
28
29  project_id = "powering-goo-68-ae453f98"
30
31  # SET VARIABLES
32  RESULT_TOPIC = "acg-image-to-text-results"
33  TRANSLATE_TOPIC = "acg-image-to-text-translation"
34  RESULT_BUCKET = "translated-images22"
35  TRANSLATE = True
36  TO_LANG = "en fr es ar ru hi"
37
38  # [START functions_ocr_detect]
39  def detect_text(bucket, filename):
40      print('Looking for text in image {}'.format(filename))
41
42      futures = []
43
44      image = vision.Image(
45          source=vision.ImageSource(gcs_image_uri=f"gs://{bucket}/{filename}")
```

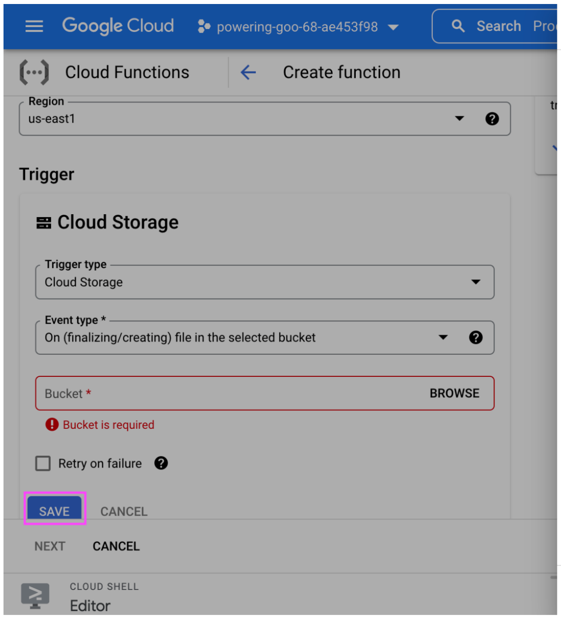
5. Set up the three Cloud Functions
    a. Open Cloud Functions by typing it in the search bar
    b. “Create Function”



Welcome to Cloud Functions!

Cloud Functions is a lightweight, event-based, asynchronous compute solution that allows you to create small, single-purpose functions that respond to cloud events - without the need to manage a server or a runtime environment.

    c. Assign the first function a “name”; assign it the same “region” we’ve been using for our project; change “trigger type” to “Cloud Storage”; change “event type” to “finalizing/creating”; assign the first bucket for images created; “Save”



Region: us-east1

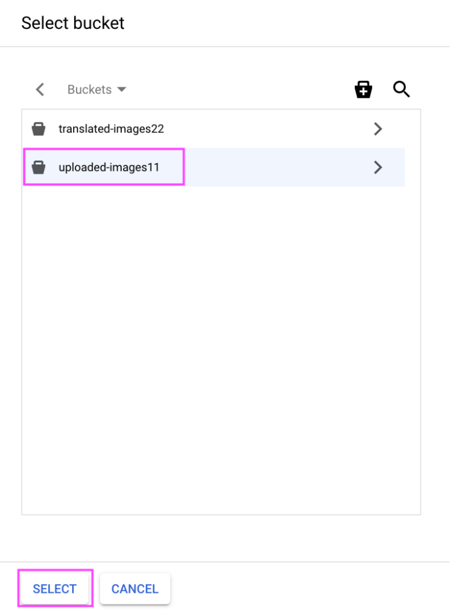
Trigger type: Cloud Storage

Event type *: On (finalizing/creating) file in the selected bucket

Bucket *

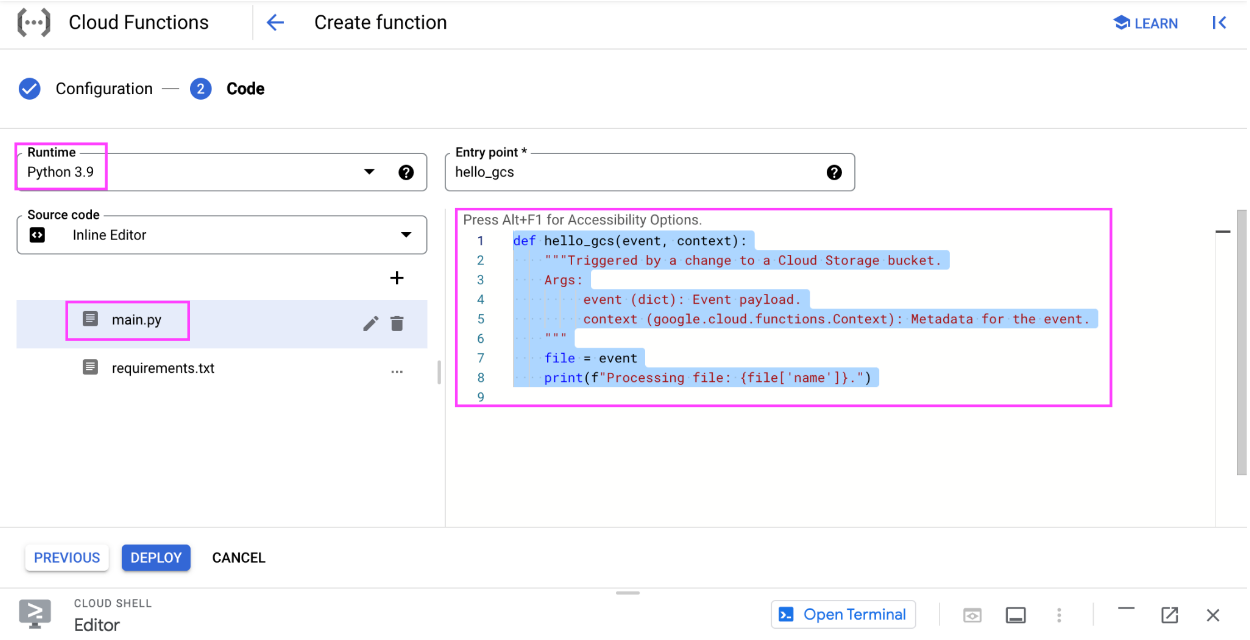
Bucket is required

□ Retry on failure



    d. “Next”

e. Change “Runtime” to “Python 3.9”; click “main.py” and delete all of it’s code



Runtime: Python 3.9

Entry point *: hello_gcs

Source code: Inline Editor

Press Alt+F1 for Accessibility Options.

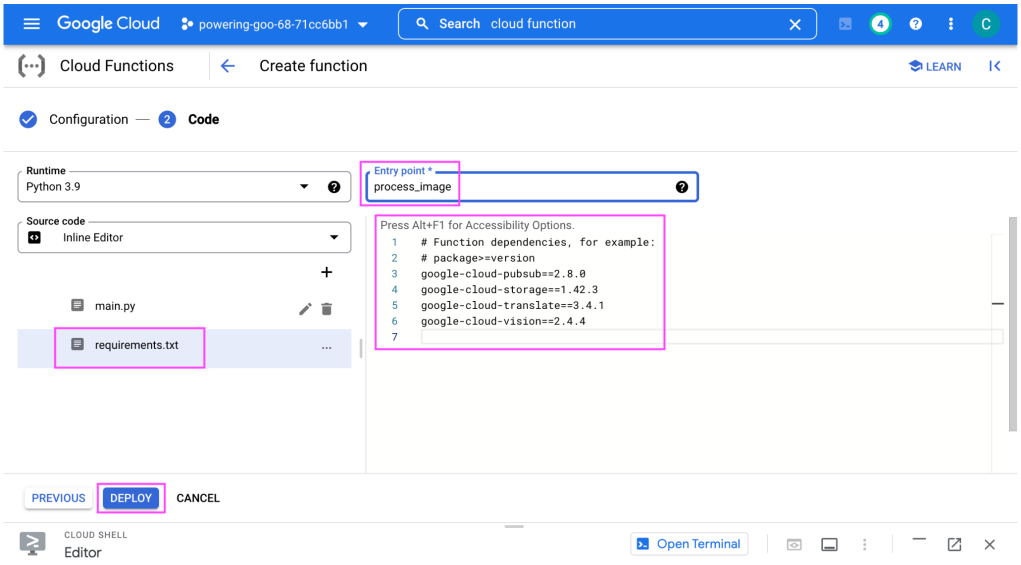
```
def hello_gcs(event, context):
    """Triggered by a change to a Cloud Storage bucket.
    Args:
         event (dict): Event payload.
         context (google.cloud.functions.Context): Metadata for the event.
    """
    file = event
    print(f"Processing file: {file['name']}.")
```

f. Copy all the code from our Cloud Shell Editor, and paste it into the “main.py” Cloud Functions section

g. Copy the “requirements.txt” section of Cloud Shell Editor, and paste it underneath the “requirements.txt” section on our Cloud Functions page

h. Change the “Entry point” function name to our function name listed on line 94 of our main.py script

i. “Deploy”



Runtime: Python 3.9

Entry point *: process_image

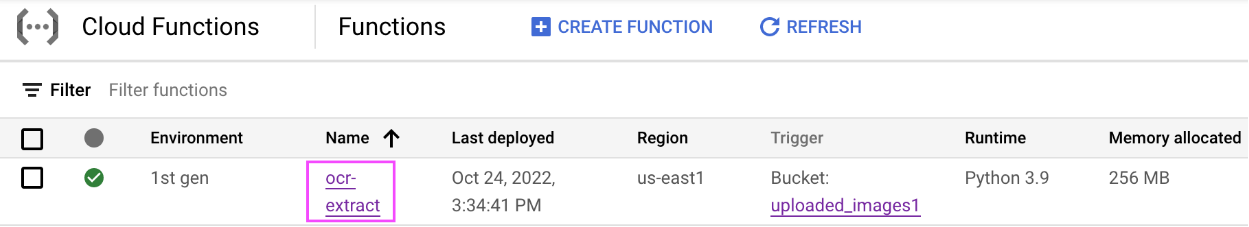
Source code: Inline Editor

Press Alt+F1 for Accessibility Options.

```
# Function dependencies, for example:
# package>=version
google-cloud-pubsub==2.8.0
google-cloud-storage==1.42.3
google-cloud-translate==3.4.1
google-cloud-vision==2.4.4
```

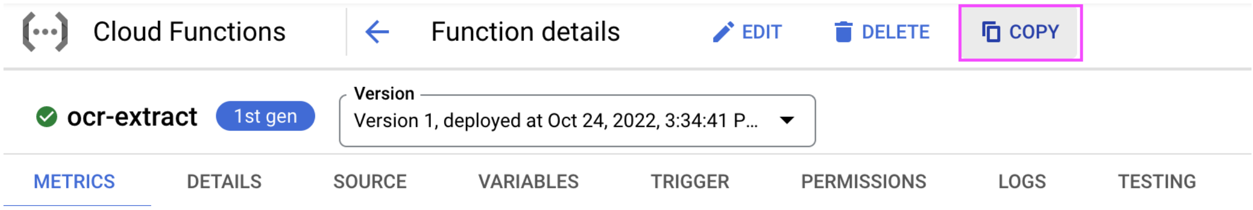
j. Create the next two functions

i. Click on the first function’s name once it is created



□	●	Environment	Name ↑	Last deployed	Region	Trigger	Runtime	Memory allocated
□	✓	1st gen	ocr-extract	Oct 24, 2022, 3:34:41 PM	us-east1	Bucket: uploaded_images1	Python 3.9	256 MB

ii. “Copy”

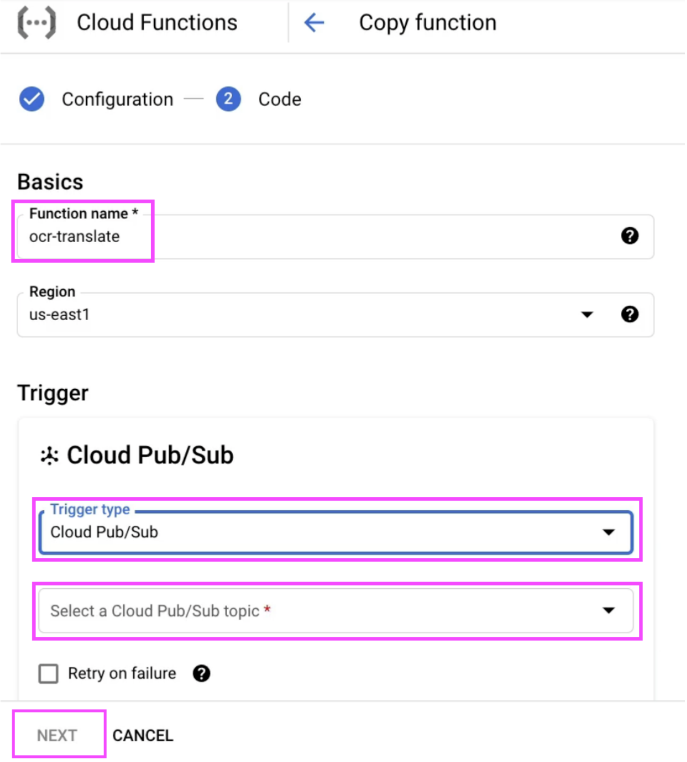


✓ ocr-extract 1st gen

Version: Version 1, deployed at Oct 24, 2022, 3:34:41 P...

iii. Give the new function a different “Function name”; change “Trigger type” to “Cloud Pub/Sub”; under “Select a Cloud Pub/Sub topic” to “projects/powering-goo-68-71cc6bb1/topics/acg-image-to-text-translation; “Save”

iv. “Next”



Basics

Function name *: ocr-translate

Region: us-east1

Trigger

Cloud Pub/Sub

Trigger type: Cloud Pub/Sub

Select a Cloud Pub/Sub topic *

□ Retry on failure

v. Change the “Entry point” to “translate_text

vi. “Deploy”

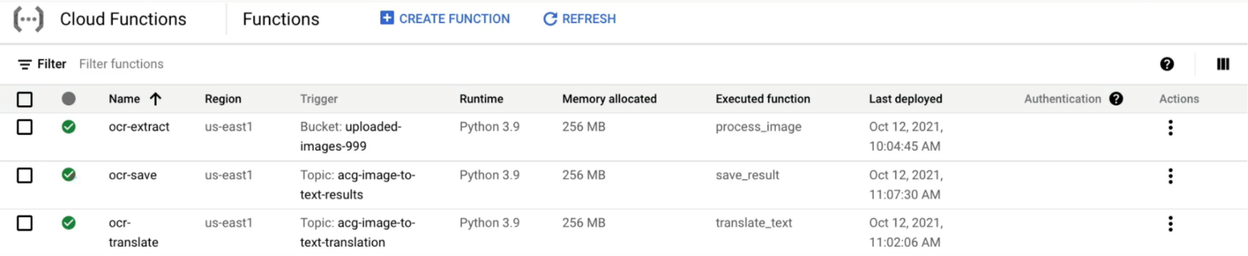
vii. “Copy” the newly created function

viii. Change the “Cloud Pub/Sub topic” to “acg-image-to-text-results”

ix. “Next”

x. Change the “Entry point” to “save_result”

xi. “Deploy”



□	●	Name ↑	Region	Trigger	Runtime	Memory allocated	Executed function	Last deployed	Authentication	Actions
□	✓	ocr-extract	us-east1	Bucket: uploaded-images-999	Python 3.9	256 MB	process_image	Oct 12, 2021, 10:04:45 AM		⋮
□	✓	ocr-save	us-east1	Topic: acg-image-to-text-results	Python 3.9	256 MB	save_result	Oct 12, 2021, 11:07:30 AM		⋮
□	✓	ocr-translate	us-east1	Topic: acg-image-to-text-translation	Python 3.9	256 MB	translate_text	Oct 12, 2021, 11:02:06 AM		⋮

6. Test the entire project’s workflow
    a. “Navigation Menu”; “Cloud Storage”; “Open Terminal”
    b. Use the following commands to reach the images through Cloud Shell, in order to upload them
    c. Upload the images to the Cloud Shell
        i. gsutil cp sign.png gs://uploaded_images1