CSL7090 Assignment 3

Saurabh Shashikant Burewar (B18CSE050)

Application

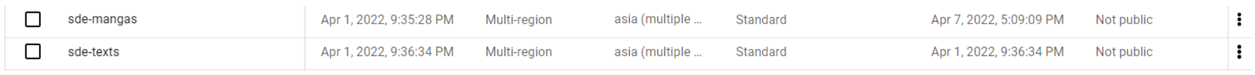
I have created a small application that translates a manga (Japanese comic) page from Japanese to english. It takes an image of a page as input, performs OCR to get the texts in that page and translates all texts from Japanese to english. It then returns the list of english texts as the output.

To create this application, the following GCP services are used -

Storage service

1. **Cloud storage**

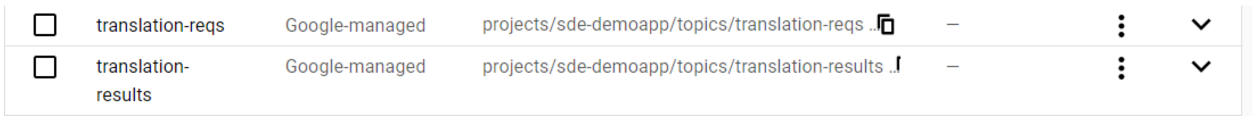
   We use cloud storage to store all our data which includes our input image and our output texts. We make two buckets for our application here -
   - sde-mangas: This bucket will contain the images that the user provides as input.
   - sde-texts: This bucket will contain the texts that the application returns.



sde-mangas	Apr 1, 2022, 9:35:28 PM	Multi-region	asia (multiple ...	Standard	Apr 7, 2022, 5:09:09 PM	Not public
sde-texts	Apr 1, 2022, 9:36:34 PM	Multi-region	asia (multiple ...	Standard	Apr 1, 2022, 9:36:34 PM	Not public

2. **Cloud Pub/Sub**

   We also create pub/sub topics to publish our translation requests and results. We make two topics for our application here -
   - translation-reqs: This topic will contain all the translation requests and will be a trigger for our translation function.
   - translation-results: This topic will contain all the translation results and will be a trigger for our result saving function.

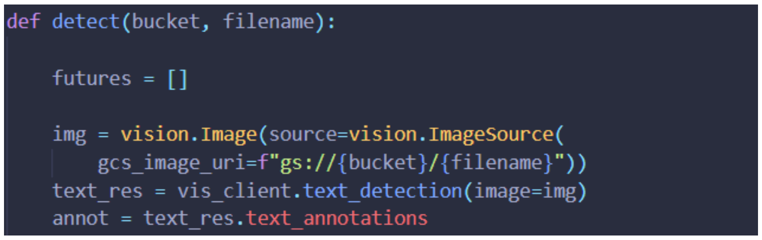


translation-reqs	Google-managed	projects/sde-demoapp/topics/translation-reqs ..	–
translation-results	Google-managed	projects/sde-demoapp/topics/translation-results ..	–

Cloud AI

1. Vision API

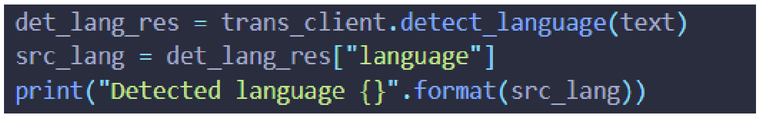
We enable the Cloud Vision API and use it through our python script. In our python script we have a detect method where we use vision API to perform OCR.



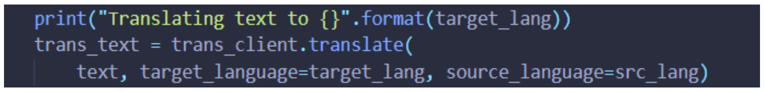
```python
def detect(bucket, filename):

    futures = []

    img = vision.Image(source=vision.ImageSource(
        gcs_image_uri=f"gs://{bucket}/{filename}"))
    text_res = vis_client.text_detection(image=img)
    annot = text_res.text_annotations
```

2. Translation AI

We use translation AI through our python script. In our python script we have a translate method where we use it to translate the detected language (japanese here) to other languages, including english.



```python
det_lang_res = trans_client.detect_language(text)
src_lang = det_lang_res["language"]
print("Detected language {}".format(src_lang))
```



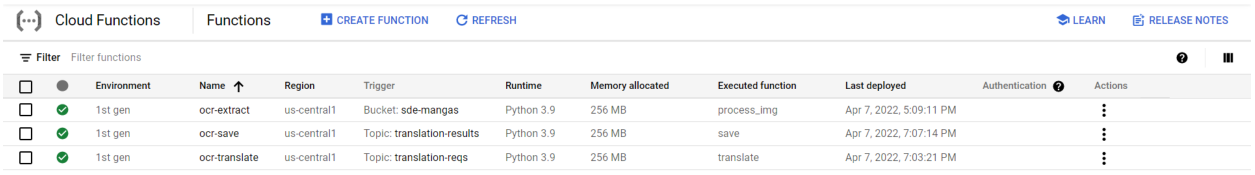
```python
print("Translating text to {}".format(target_lang))
trans_text = trans_client.translate(
    text, target_language=target_lang, source_language=src_lang)
```

Compute service

1. Cloud functions

We deploy our OCR and translation methods as cloud functions with cloud storage buckets and pub/sub topics as triggers. We create three cloud functions -

- ocr-extract: This function performs OCR, extracts text from the image and detects the language. The trigger bucket for this function is “sde-mangas”, so when there is a “create” change in this bucket, this function is called and it publishes in the translation requests topic.
- ocr-translate: This function performs the translation of texts from Japanese to other languages including english. The trigger topic for this function is “translation-reqs”, so when there is a change in this topic, this function is called and it publishes in the translation results topic.
- ocr-save: This function saves the results of the translations. The trigger topic for this function is “translation-results”, so when there is a change in this topic, this function is called and it uploads the results to the “sde-texts” bucket.



Cloud Functions | Functions | CREATE FUNCTION | REFRESH | LEARN | RELEASE NOTES

Filter | Filter functions

		Environment	Name	Region	Trigger	Runtime	Memory allocated	Executed function	Last deployed	Authentication	Actions
☐	✓	1st gen	ocr-extract	us-central1	Bucket: sde-mangas	Python 3.9	256 MB	process_img	Apr 7, 2022, 5:09:11 PM		⋮
☐	✓	1st gen	ocr-save	us-central1	Topic: translation-results	Python 3.9	256 MB	save	Apr 7, 2022, 7:07:14 PM		⋮
☐	✓	1st gen	ocr-translate	us-central1	Topic: translation-reqs	Python 3.9	256 MB	translate	Apr 7, 2022, 7:03:21 PM		⋮

We use the console to create these and upload our code to be used in these functions as zip, or we can use the inline editor and copy/paste our code.

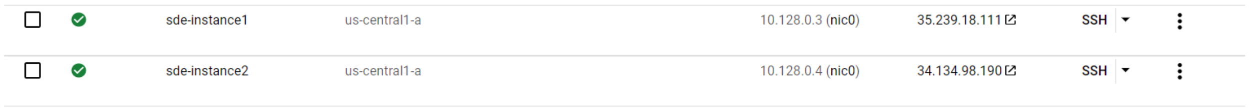
2. Compute Engine

To get our input from our user, I built a simple flask application which opens a webpage which takes file input and stores it in the cloud storage bucket (which will trigger our cloud functions). It then waits for cloud functions to finish the job and then reads the output from the cloud storage bucket to display it on the webpage.

We use VM instances in the compute engine to run our flask application. We create instances using this command -

```
gcloud compute instances create sde-instance1 \
    --image-family=debian-10 \
    --image-project=debian-cloud \
    --machine-type=g1-small \
    --scopes userinfo-email,cloud-platform \
    --metadata-from-file startup-script=startup.sh \
    --zone us-central1-a \
    --tags http-server
```

The startup script we provide downloads our flask application code and sets up everything and runs the application. The webpage can be accessed on the external IP address of the VM instance.



☐	✓	sde-instance1	us-central1-a	10.128.0.3 (nic0)	35.239.18.111	SSH	⋮
☐	✓	sde-instance2	us-central1-a	10.128.0.4 (nic0)	34.134.98.190	SSH	⋮

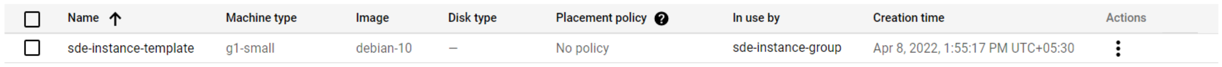
We also set up firewall rules to allow HTTP traffic from all addresses in the 8080 port.

```
gcloud compute firewall-rules create default-allow-http-8080 \
    --allow tcp:8080 \
    --source-ranges 0.0.0.0/0 \
    --target-tags http-server \
    --description "Allow port 8080 access to http-server"
```

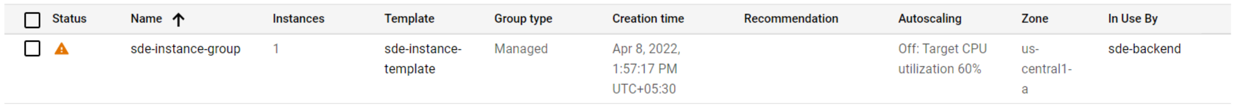
Load balancing

1. Building network

We build a VPC network and instance groups for our backend.

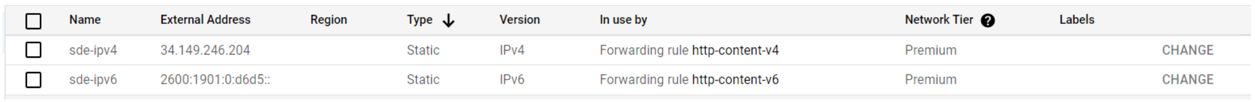


☐	Name ↑	Machine type	Image	Disk type	Placement policy	In use by	Creation time	Actions
☐	sde-instance-template	g1-small	debian-10	–	No policy	sde-instance-group	Apr 8, 2022, 1:55:17 PM UTC+05:30	⋮



☐	Status	Name ↑	Instances	Template	Group type	Creation time	Recommendation	Autoscaling	Zone	In Use By
☐	⚠	sde-instance-group	1	sde-instance-template	Managed	Apr 8, 2022, 1:57:17 PM UTC+05:30		Off: Target CPU utilization 60%	us-central1-a	sde-backend

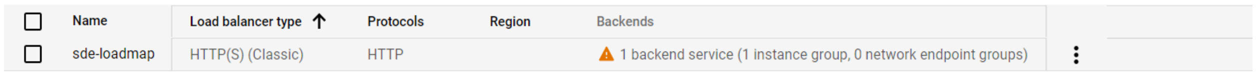
We add a named port to our instance group. We have firewall rules to allow traffic from all addresses. We also reserve external IP addresses for our load balancer.



☐	Name	External Address	Region	Type ↓	Version	In use by	Network Tier	Labels	
☐	sde-ipv4	34.149.246.204		Static	IPv4	Forwarding rule http-content-v4	Premium		CHANGE
☐	sde-ipv6	2600:1901:0:d6d5::		Static	IPv6	Forwarding rule http-content-v6	Premium		CHANGE

2. Load balancer

We make a HTTP load balancer by configuring the backend for health checks and the frontend for the forwarding rules accordingly.



☐	Name	Load balancer type ↑	Protocols	Region	Backends	
☐	sde-loadmap	HTTP(S) (Classic)	HTTP		⚠ 1 backend service (1 instance group, 0 network endpoint groups)	⋮

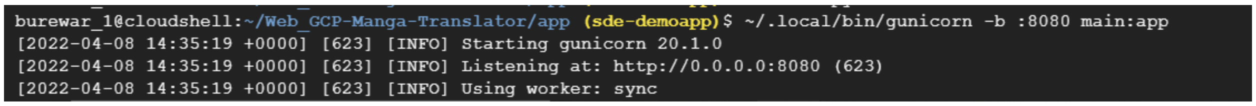
Then, we connect the domain to load balancer and set up to send traffic to our instances.

Demo

1. User input into VM instance

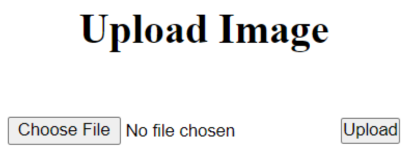
We first run the VM instance "sde-instance1". The flask application is hosted on the external IP address of the instance. We upload the image on the webpage. Once uploaded, the application stores the image into the "sde-mangas" bucket. This triggers the cloud functions and the flask application waits for cloud functions to complete the process.

This is screenshot of how the server would look like -

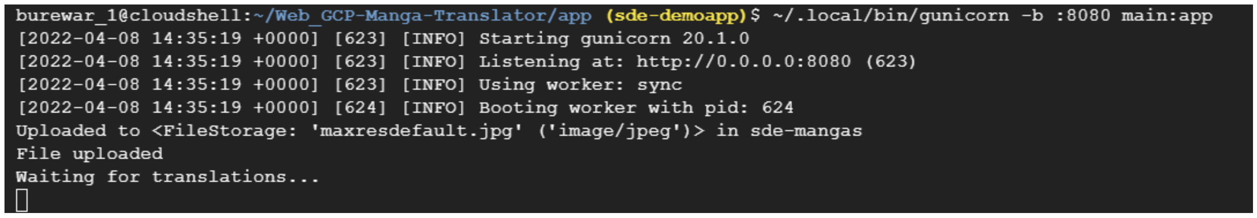


```
burewar_1@cloudshell:~/Web_GCP-Manga-Translator/app (sde-demoapp)$ ~/.local/bin/gunicorn -b :8080 main:app
[2022-04-08 14:35:19 +0000] [623] [INFO] Starting gunicorn 20.1.0
[2022-04-08 14:35:19 +0000] [623] [INFO] Listening at: http://0.0.0.0:8080 (623)
[2022-04-08 14:35:19 +0000] [623] [INFO] Using worker: sync
```

The webpage looks like this -



After uploading the image -

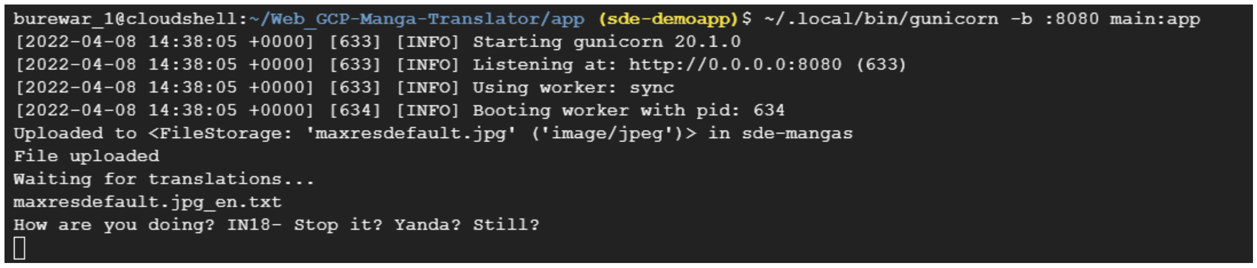


```
burewar_1@cloudshell:~/Web_GCP-Manga-Translator/app (sde-demoapp)$ ~/.local/bin/gunicorn -b :8080 main:app
[2022-04-08 14:35:19 +0000] [623] [INFO] Starting gunicorn 20.1.0
[2022-04-08 14:35:19 +0000] [623] [INFO] Listening at: http://0.0.0.0:8080 (623)
[2022-04-08 14:35:19 +0000] [623] [INFO] Using worker: sync
[2022-04-08 14:35:19 +0000] [624] [INFO] Booting worker with pid: 624
Uploaded to <FileStorage: 'maxresdefault.jpg' ('image/jpeg')> in sde-mangas
File uploaded
Waiting for translations...
```

2. Cloud functions

The cloud functions have triggers and are called automatically on change in the cloud storage bucket.

- The cloud function "ocr-extract" gets triggered when a new file is uploaded to "sde-mangas". This function performs OCR and changes "translation-reqs" topic.
- The change in "translation-reqs" topic triggers the cloud function "ocr-translate". This function performs translation and changes "translation-results" topic.
- The change in "translation-results" topic triggers the cloud function "ocr-save". This function saves the result into the "sde-texts" bucket.

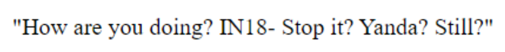


```
burewar_1@cloudshell:~/Web_GCP-Manga-Translator/app (sde-demoapp)$ ~/.local/bin/gunicorn -b :8080 main:app
[2022-04-08 14:38:05 +0000] [633] [INFO] Starting gunicorn 20.1.0
[2022-04-08 14:38:05 +0000] [633] [INFO] Listening at: http://0.0.0.0:8080 (633)
[2022-04-08 14:38:05 +0000] [633] [INFO] Using worker: sync
[2022-04-08 14:38:05 +0000] [634] [INFO] Booting worker with pid: 634
Uploaded to <FileStorage: 'maxresdefault.jpg' ('image/jpeg')> in sde-mangas
File uploaded
Waiting for translations...
maxresdefault.jpg_en.txt
How are you doing? IN18- Stop it? Yanda? Still?
```

3. Output at the VM instance

The results are ready in the "sde-texts" bucket. The flask application reads the results from the bucket and displays it on the webpage.

The output looks like this -


"How are you doing? IN18- Stop it? Yanda? Still?"

Demo video

Link to the demonstration video -
https://youtu.be/7xOWBCW6QCk

Code

All the code is available in this github repo -
https://github.com/saurabhburewar/Web_GCP-Manga-Translator

References

- Application -
  - https://cloud.google.com/vision
  - https://cloud.google.com/translate
  - https://cloud.google.com/functions/docs/tutorials/ocr#functions_ocr_setup-python
  - https://github.com/GoogleCloudPlatform/python-docs-samples/tree/cf1643c8e2818c539d0c204405018acc8c7b0994/functions/ocr
  - https://cloud.google.com/python/docs/getting-started/getting-started-on-compute-engine#windows
  - https://github.com/GoogleCloudPlatform/getting-started-python
- Load balancing -
  - https://cloud.google.com/load-balancing/docs/https/setting-up-https-serverless#console
  - https://cloud.google.com/load-balancing/docs/https/setting-up-https#console-and-web-browser