Python with GCP

Submitted by:
Aditya K Kasturi
Aditya Rohan Singh
Naveen Kumar Gorantla
Pradipkumar Rajasekaran

Webpage Static URL for Cloud function implementation

`https://34.102.205.200`

Webpage Static URL for Cloud run implementation

`https://34.110.171.39`

Cloud Function URL

`https://us-central1-gold-episode-347200.cloudfunctions.net/getfractal1`

Cloud Run URL

`https://getfractal1-47ntsdicza-uc.a.run.app/`

Screenshots from Tashfeen's Website

`https://tashfeen.org/fractalsetc/build/index.html`

Files Used

```
cloud/python/main.py
cloud/python/templates/getfractal.html
cloud/python/cert
```

Cloud Functions

Steps Executed to create cloud function

- Cloned the repo provided for reference to local machine.
- Removed unused files.
- Selected cloud function as method of implementation.
- Edited main.py and added function getfractal1 to create a cloud function called getfractal1 which triggers flask framework to run a html page.
- Created a storage bucket and stored screenshots from Tashfeen's website. Made it accessible to public.
- Enable Cloud function API.

Cloud Functions

Steps Executed to create cloud function (continous)

- Created a html page under templates called getfractal.html and added code to display all the images stored in the storage bucket.
- Deployed the cloud function via command line from GCP instance.
  `gcloud functions deploy getfractal1 --runtime python39 --trigger-http --allow-unauthenticated`
- Added allUsers under permission via console with the role "Cloud Function Invoker" to the function getfractal1
- Verified deployment of getfractal1 by opening the url found under trigger.
  `https://us-central1-gold-episode-347200.cloudfunctions.net/getfractal1`
- Cloud function deployment of html page complete.

Cloud Run

Steps Executed to create cloud Run

- Same steps from 1 to 5 from cloud function.
- Enabled Cloud Run API and artifact registry API.
- Same as step 7 from above.
- Edit Dockerfile and replace FROM python:2-slim with FROM python:3-slim
- Uncomment gunicorn from requirements.txt

Cloud Run

Steps Executed to create cloud Run (continous)

- Uncomment the ENTRYPOINT entry for gunicorn replacing the default entry in Dockerfile
- Deployed the cloud run via command line from GCP instance

```
gcloud run deploy getfractal1 --source . --allow-unauthenticated --platform managed
```

- Added allUsers under permission via console with the role "Cloud Run Invoker" to the run `getfractal1/`
- Verified Deployment of cloud run gerfractal1 by opening the url found under trigger.

```
https://getfractal1-47ntsdicza-uc.a.run.app
```

- Cloud Run deployement of html page complete.

Steps Executed to access the webpage via static IP

- Same steps will be executed for both Cloud function and cloud run
- except where we link the function/run to the load balancer.

Load Balancer with Cloud Function

`https://cloud.google.com/load-balancing/docs/https/setting-up-https-serverless#gcloud_1`

Self Signed SSL Certificate

`https://cloud.google.com/load-balancing/docs/ssl-certificates/self-managed-certs`

Reserve IP Address and Endpoint

- Reserve an external IP address for Cloud Function and Cloud Run
  - **Cloud Function**
    `gcloud compute addresses create fractaldisplay --network-tier=PREMIUM --ip-version=IPV4 --global`
  - **Cloud Run**
    `gcloud compute addresses create fractaldisplay1 --network-tier=PREMIUM --ip-version=IPV4 --global`
- Create a serverless NEG for your serverless app with cloud function and CLoud Run
  - **Cloud Function**
    `gcloud compute network-endpoint-groups create fractal --region=us-central1 --network-endpoint-type=serverless --cloud-function-name=getfractal1`
  - **Cloud Run**
    `gcloud compute network-endpoint-groups create fractal1 --region=us-central1 --network-endpoint-type=serverless --cloud-run-service=getfractal1`

Self Signed SSL Certificate

Create an SSL certificate resource

- Create a private key and certificate in PEM format. - `Private Key file name: privatekey` `Certificate file name: certificate`
  `openssl genrsa -out /home/adirohan95/Software/cloudtest/cloud-nebulous-serverless/cloud/python/cert/privatekey 2048`
- Create a certificate signing request (CSR) in the PEM format
- Create an OpenSSL configuration file called `CONFIG_File`
- Execute the blow command to generate the CSR
  `openssl req -new -key privatekey -out CSR -config CONFIG_FILE`
- Sign the Certificate using the below command
  `openssl x509 -req -signkey privatekey -in CSR -out certificate -extfile CONFIG_FILE -extensions extension_requirements -days 100`

Self Signed SSL Certificate

Create SSL certificate in cloud via command line

- Use the following command
  `gcloud compute ssl-certificates create fractal-certificate --certificate=certificate --private-key=privatekey --global`

Load Balancer

- Create a backend service for Cloud Function and Cloud Run
  - **Cloud Funtion**

    `gcloud compute backend-services create get-fractal --load-balancing-scheme=EXTERNAL --global`
  - **Cloud Run**

    `Cloud Run gcloud compute backend-services create get-fractal1 --load-balancing-scheme=EXTERNAL --global`

Load Balancer (continous)

- Add the serverless NEG as a backend to the backend service
  - **Cloud Funtion**
    ```
    gcloud compute backend-services add-backend get-fractal --global --network-endpoint-group=fractal --network-endpoint-group-region=us-central1
    ```
  - **Cloud Run**
    ```
    gcloud compute backend-services add-backend get-fractal1 --global --network-endpoint-group=fractal1 --network-endpoint-group-region=us-central1
    ```
- Create a URL map to route incoming requests to the backend service
  - **Cloud Funtion**
    ```
    gcloud compute url-maps create fractal-map --default-service get-fractal
    ```
  - **Cloud Run**
    ```
    gcloud compute url-maps create fractal-map1 --default-service get-fractal1
    ```

Load Balancer (continous)

- Create a target HTTP(S) proxy to route requests to your URL map
  - **Cloud Funtion**
    ```
    gcloud compute target-https-proxies create getfractalgcp --global --url-map fractal-map --global-url-map --ssl-certificates fractal-certificate --global-ssl-certificates
    ```
  - **Cloud Run**
    ```
    gcloud compute target-https-proxies create getfractalgcp1 --global --url-map fractal-map1 --global-url-map --ssl-certificates fractal-certificate --global-ssl-certificates
    ```
- Create a forwarding rule to route incoming requests to the proxy
  - **Cloud Function**
    ```
    gcloud compute forwarding-rules create fractalforwarding --load-balancing-scheme=EXTERNAL --network-tier=PREMIUM --address=fractaldisplay --target-https-proxy=getfractalgcp --global --ports=443
    ```
  - **Cloud Run**
    ```
    gcloud compute forwarding-rules create fractalforwarding1 --load-balancing-scheme=EXTERNAL --network-tier=PREMIUM --address=fractaldisplay1 --target-https-proxy=getfractalgcp1 --global --ports=443
    ```

Verify the load balancer is working by opening the below IP address:

- Cloud Function
  `https://34.102.205.200`
- Cloud Run
  `https://34.110.171.39/`

Thank you