Lab 07

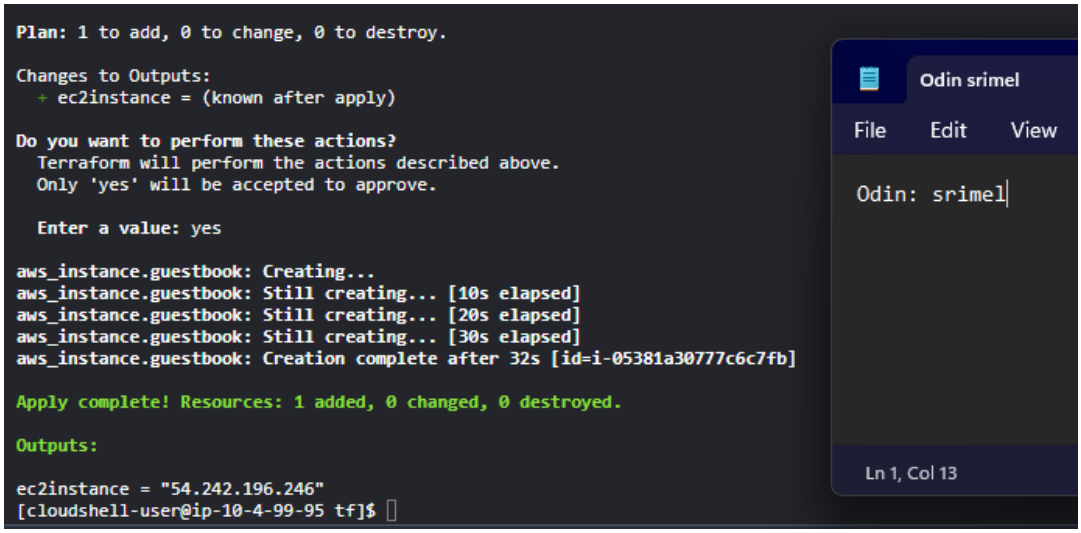
Stuart Rimel
CS530: Internet Web & Cloud, Fall 2023

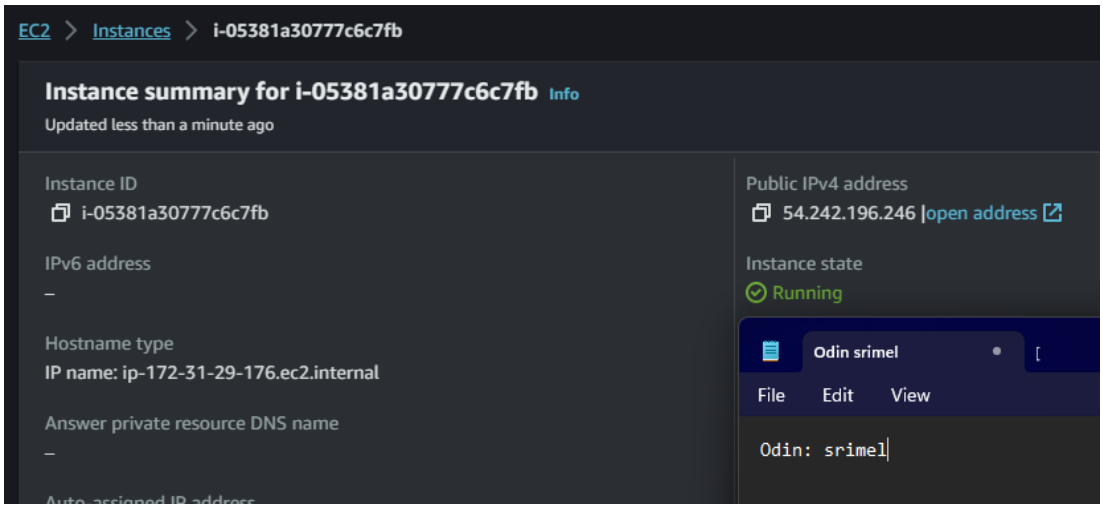
Odin: srimel
** In all the terminal screenshots my Odin name is in the terminal prompt **

Table of contents:

7.1a: Terraform AWS Guestbook

7.1a.4: Launching configuration





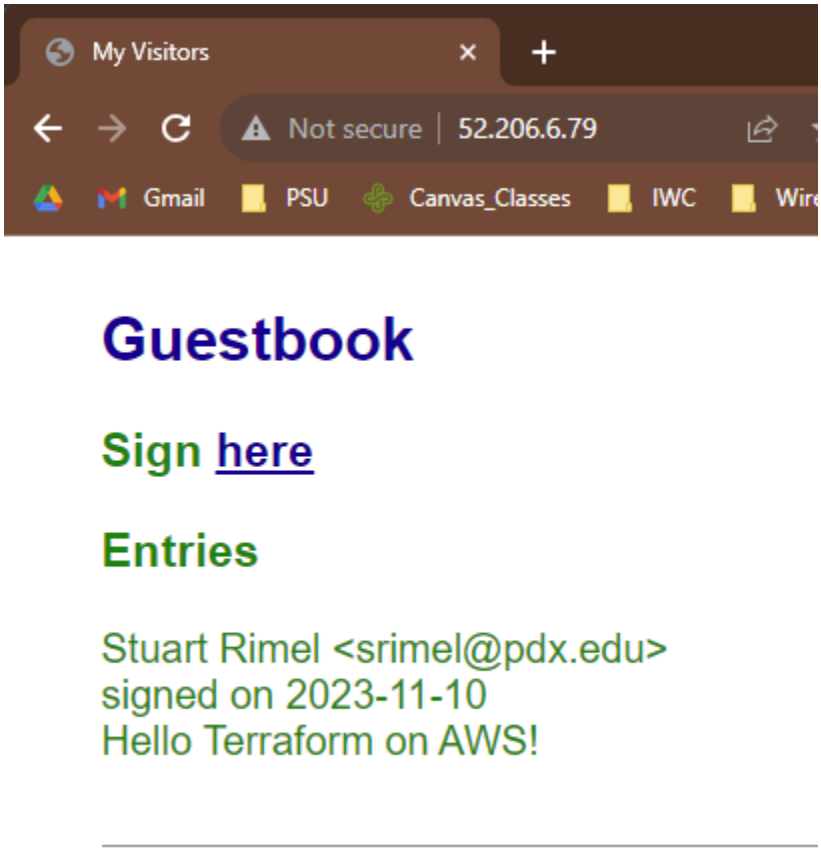
7.1a.6: Adding ssh access

7.1a.7: Adding the Guestbook application

```
  Enter a value: yes

aws_key_pair.kp: Creating...
aws_security_group.sg-guestbook: Creating...
aws_key_pair.kp: Creation complete after 0s [id=guestbook-key]
aws_security_group.sg-guestbook: Creation complete after 2s [id=sg-0ccecbdd9e3e2946c]
aws_instance.guestbook: Creating...
aws_instance.guestbook: Still creating... [10s elapsed]
aws_instance.guestbook: Still creating... [20s elapsed]
aws_instance.guestbook: Still creating... [30s elapsed]
aws_instance.guestbook: Still creating... [40s elapsed]
aws_instance.guestbook: Creation complete after 42s [id=i-022dcfddfd758c532]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

ec2instance = "52.206.6.79"
[cloudshell-user@ip-10-4-99-95 tf]$ echo "Odin: srimel"
Odin: srimel
[cloudshell-user@ip-10-4-99-95 tf]$
```

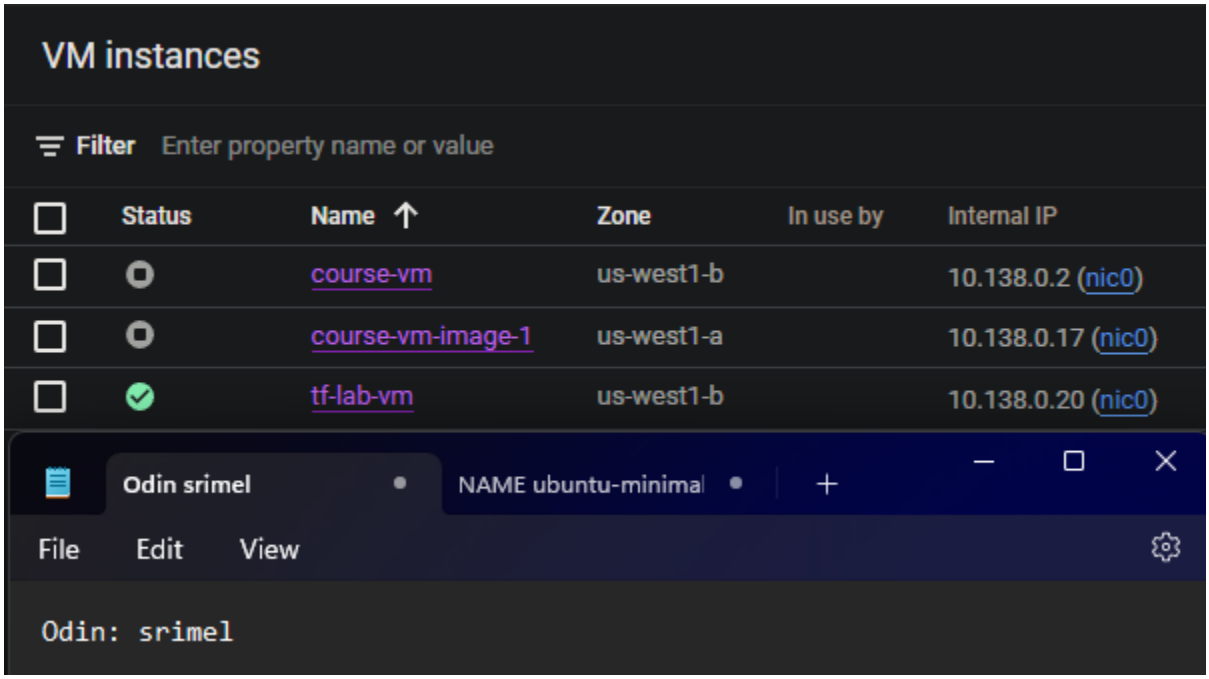
```
ubuntu@ip-172-31-29-34:~$ ps auxww | grep gunicorn
root        7445  0.4  2.2  30632 22836 ?        S    05:01   0:00 /usr/bin/python3 /usr/bin/gunicorn --bind :80 --workers 1 --threads 8 app:app
root        7447  0.2  2.9  39236 29384 ?        S    05:01   0:00 /usr/bin/python3 /usr/bin/gunicorn --bind :80 --workers 1 --threads 8 app:app
ubuntu      7455  0.0  0.0   6432   656 pts/1    S+   05:02   0:00 grep --color=auto gunicorn
ubuntu@ip-172-31-29-34:~$ echo "Odin: srimel"
Odin: srimel
ubuntu@ip-172-31-29-34:~$
```

7.1a.8: View the Guestbook

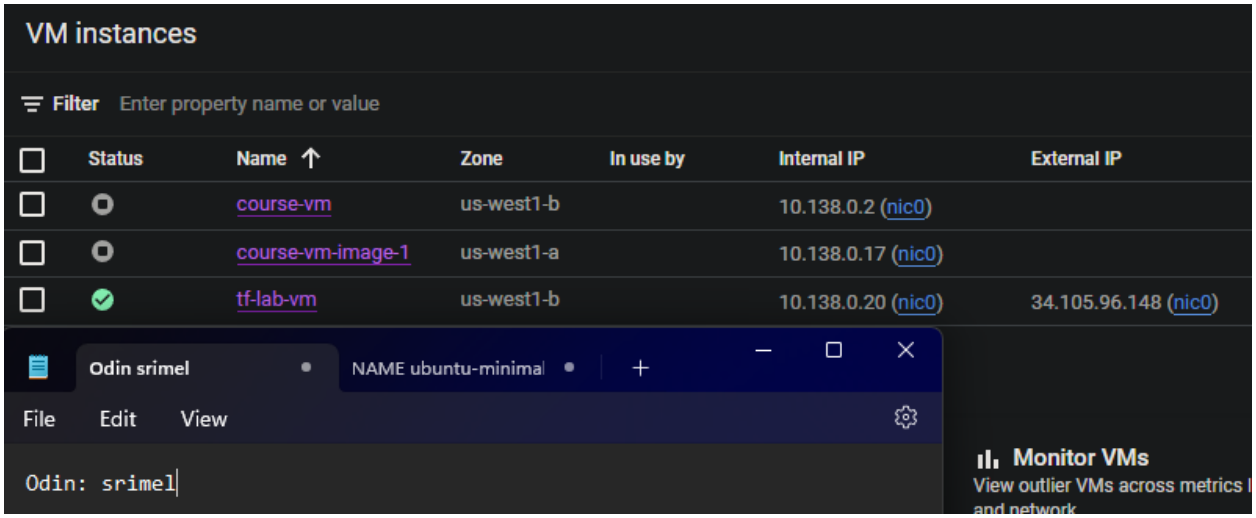


7.1g: Terraform GCP Guestbook

7.1g.4: Launching configuration



7.1g.5: Adding an external IP address



7.1g.6: Adding ssh access

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

srimel@tf-lab-vm:~$ echo "Odin: srimel"
Odin: srimel
srimel@tf-lab-vm:~$
```

7.1g.7: Adding the Guestbook application

```
Plan: 1 to add, 0 to change, 1 to destroy.
```

Resources being added:

- Resource google_compute_instance (replaced)

Resources being changed: 0

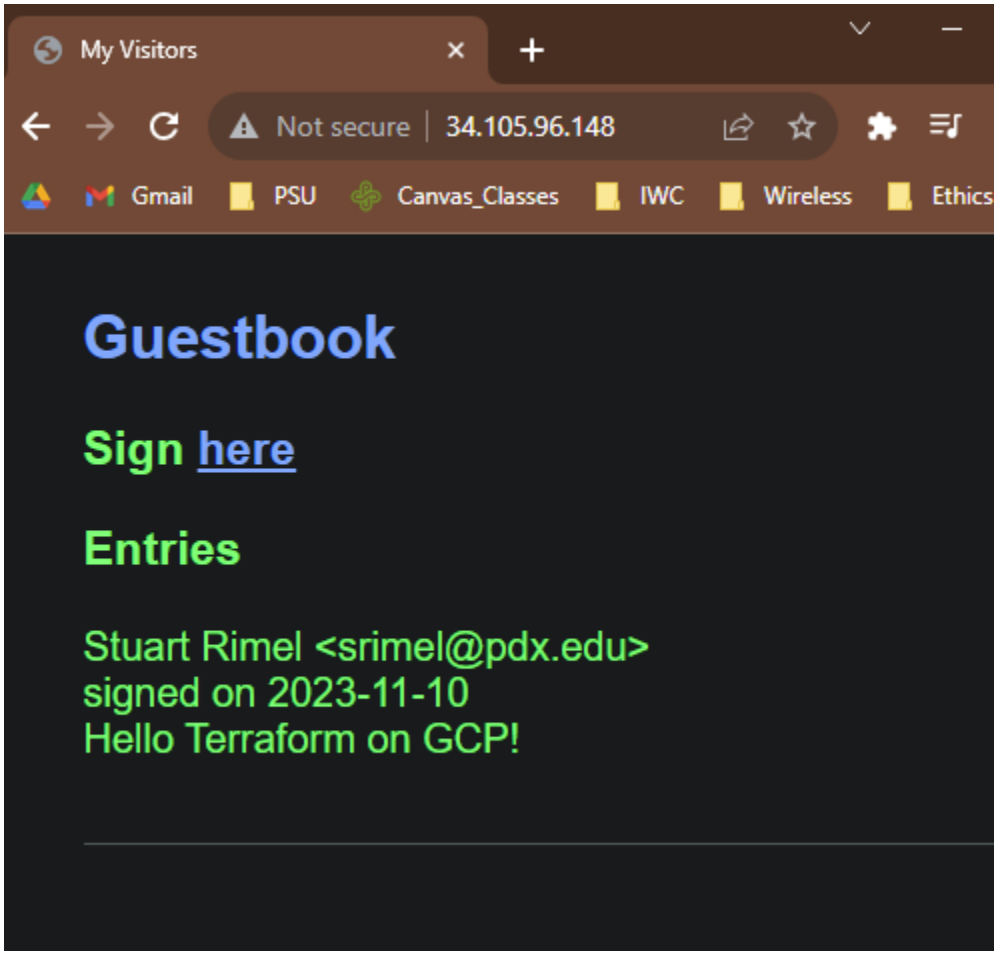
Resources being destroyed:

- Resource google_compute_instance (replaced)

Adding the metadata_startup_script to the instance is forcing replacement

```
Terraform will perform the following actions:

  # google_compute_instance.default must be replaced
-/+ resource "google_compute_instance" "default" {
      ~ cpu_platform            = "Intel Broadwell" -> (known after apply)
      ~ current_status          = "RUNNING" -> (known after apply)
      ~ effective_labels        = {} -> (known after apply)
      - enable_display          = false -> null
      ~ guest_accelerator       = [] -> (known after apply)
      ~ id                      = "projects/cloud-rimel-srimel/zones/us-west1-b/instances/tf-lab-vm" -> (known after apply)
      ~ instance_id             = "6138379567676542386" -> (known after apply)
      ~ label_fingerprint       = "42WmSpB8rSM=" -> (known after apply)
      - labels                  = {} -> null
      ~ metadata_fingerprint    = "PF7LBA5zslk=" -> (known after apply)
      + metadata_startup_script = <<-EOT # forces replacement
            apt update
            apt install -yq build-essential python3-pip gunicorn git
            pip install flask
            git clone https://github.com/wu4f/cs430-src /root/cs430-src
            cd /root/cs430-src/03_nginx_gunicorn_certbot
            gunicorn --bind :80 --workers 1 --threads 8 app:app
        EOT
      + min_cpu_platform        = (known after apply)
        name                    = "tf-lab-vm"
      - resource_policies       = [] -> null
      ~ self_link               = "https://www.googleapis.com/compute/v1/projects/cloud-rimel-srimel/zones/us-west1-b/instances/tf-lab-vm" -> (known after apply)
      ~ tags                    = [
          + "http-server",
        ]
      ~ tags_fingerprint        = "42WmSpB8rSM=" -> (known after apply)
      ~ terraform_labels        = {} -> (known after apply)
        # (6 unchanged attributes hidden)

      ~ boot_disk {
          ~ device_name                = "persistent-disk-0" -> (known after apply)
          + disk_encryption_key_sha256 = (known after apply)
          + kms_key_self_link          = (known after apply)
          ~ source                     = "https://www.googleapis.com/compute/v1/projects/cloud-rimel-srimel/zones/us-west1-b/disks/tf-lab-vm" -> (known after apply)
            # (2 unchanged attributes hidden)

          ~ initialize_params {
              ~ image                 = "https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/images/ubuntu-2004-focal-v20231101" -> "ubuntu-os-cloud/ubuntu-2004-focal-v20231101"
              ~ labels                = {} -> (known after apply)
              - resource_manager_tags = {} -> null
              ~ size                  = 10 -> (known after apply)
              ~ type                  = "pd-standard" -> (known after apply)
            }
```

7.1g.8: View the Guestbook

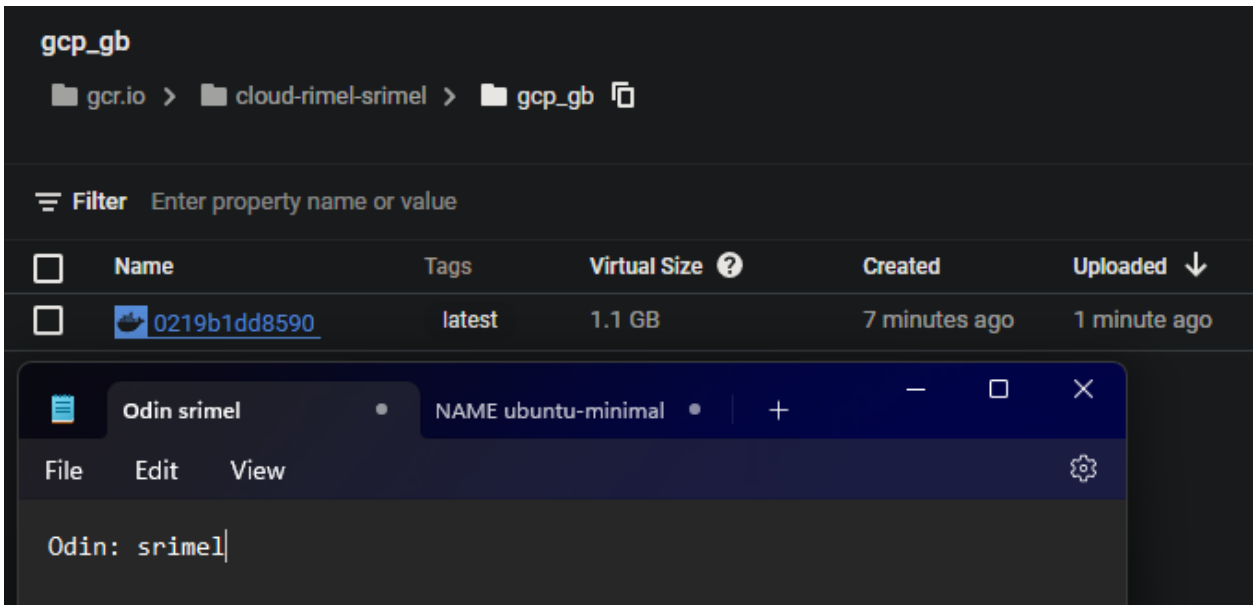


7.2g: Kubernetes Guestbook

7.2g.4: Create Kubernetes cluster

1. Instance template: gke-guestbook-default-pool-0a9eb3d6
2. Instance group: gke-guestbook-default-pool-0a9eb3d6-grp
3. Name of two nodes:
    a. gke-guestbook-default-pool-0a9eb3d6-f0mf
    b. gke-guestbook-default-pool-0a9eb3d6-rlzz

7.2g.5: Prepare a container image

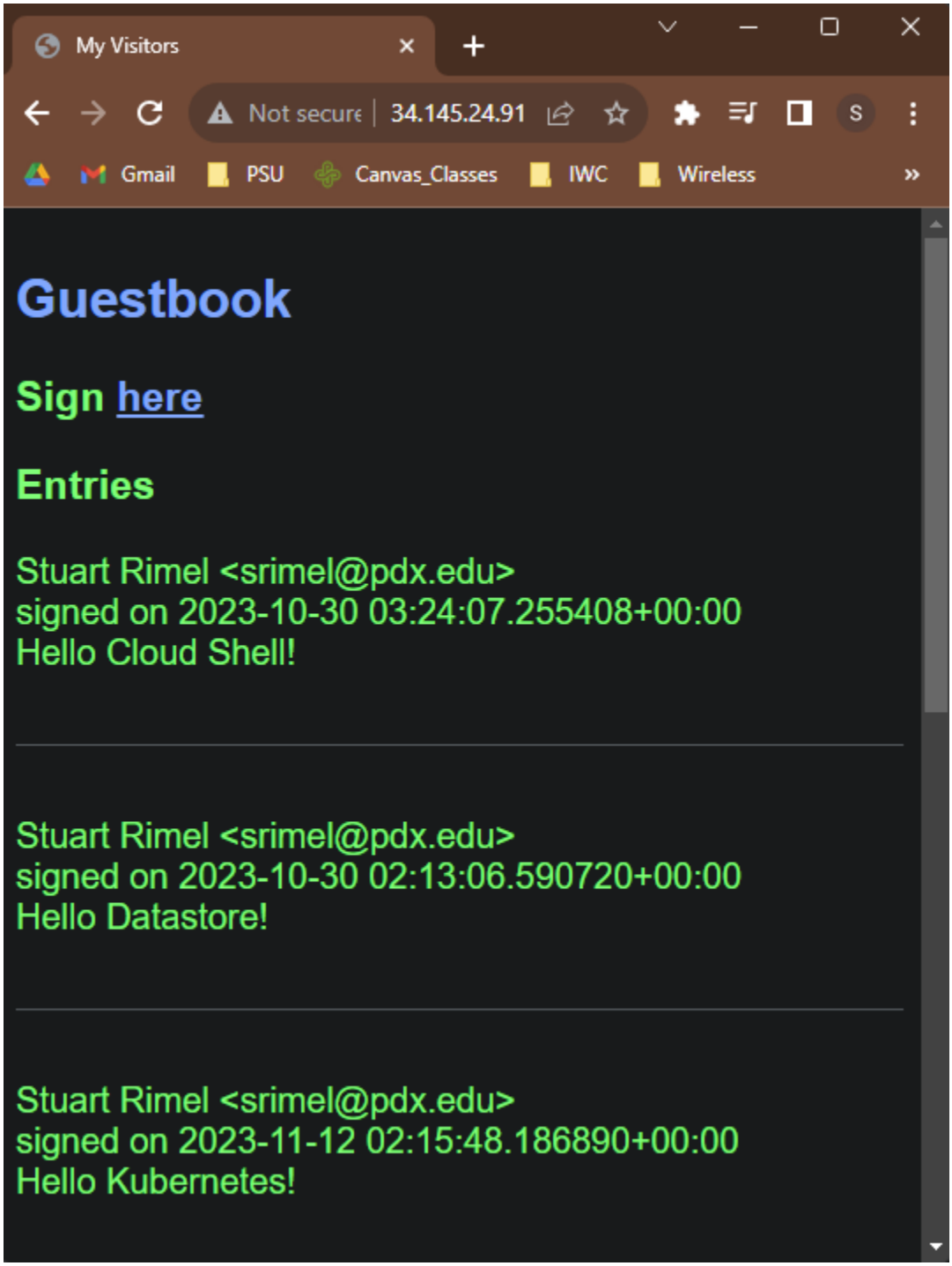


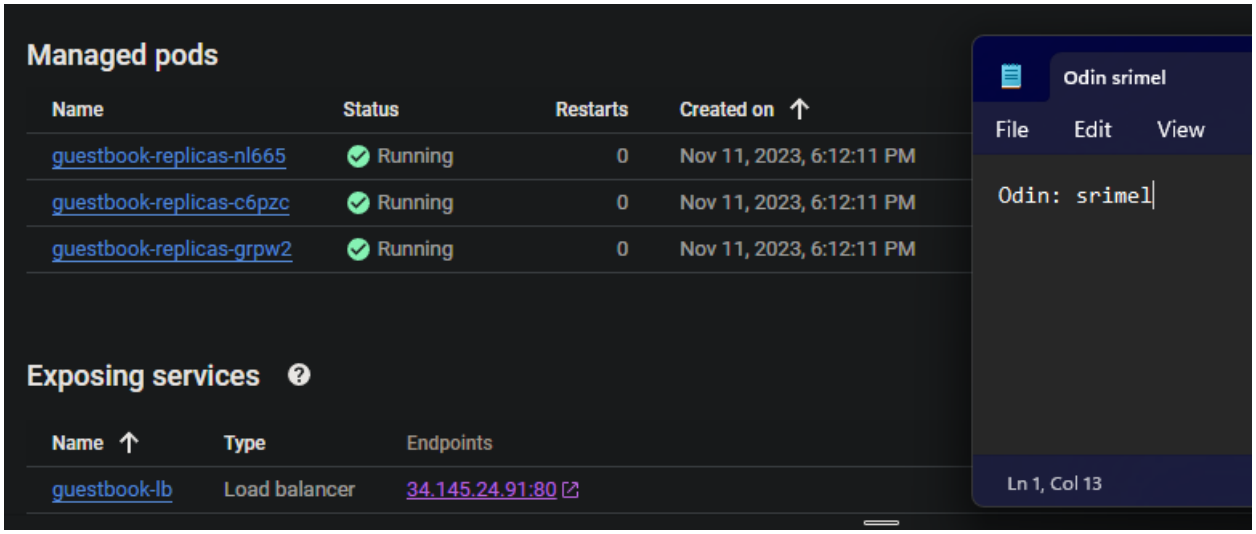
7.2g.7: Deploy the configuration

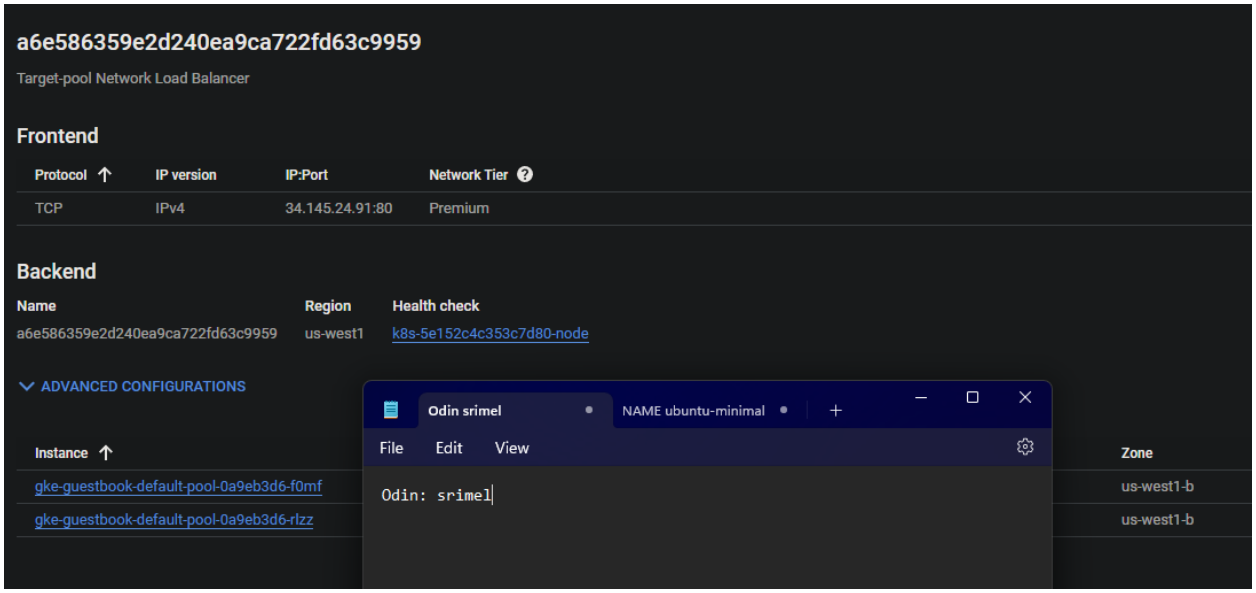
```
srimel@cloudshell:~/cs430-src/05_gcp_datastore (cloud-rimel-srimel)$ kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
guestbook-replicas-c6pzc    1/1     Running   0          100s
guestbook-replicas-grpw2    1/1     Running   0          100s
guestbook-replicas-nl665    1/1     Running   0          100s
srimel@cloudshell:~/cs430-src/05_gcp_datastore (cloud-rimel-srimel)$
```

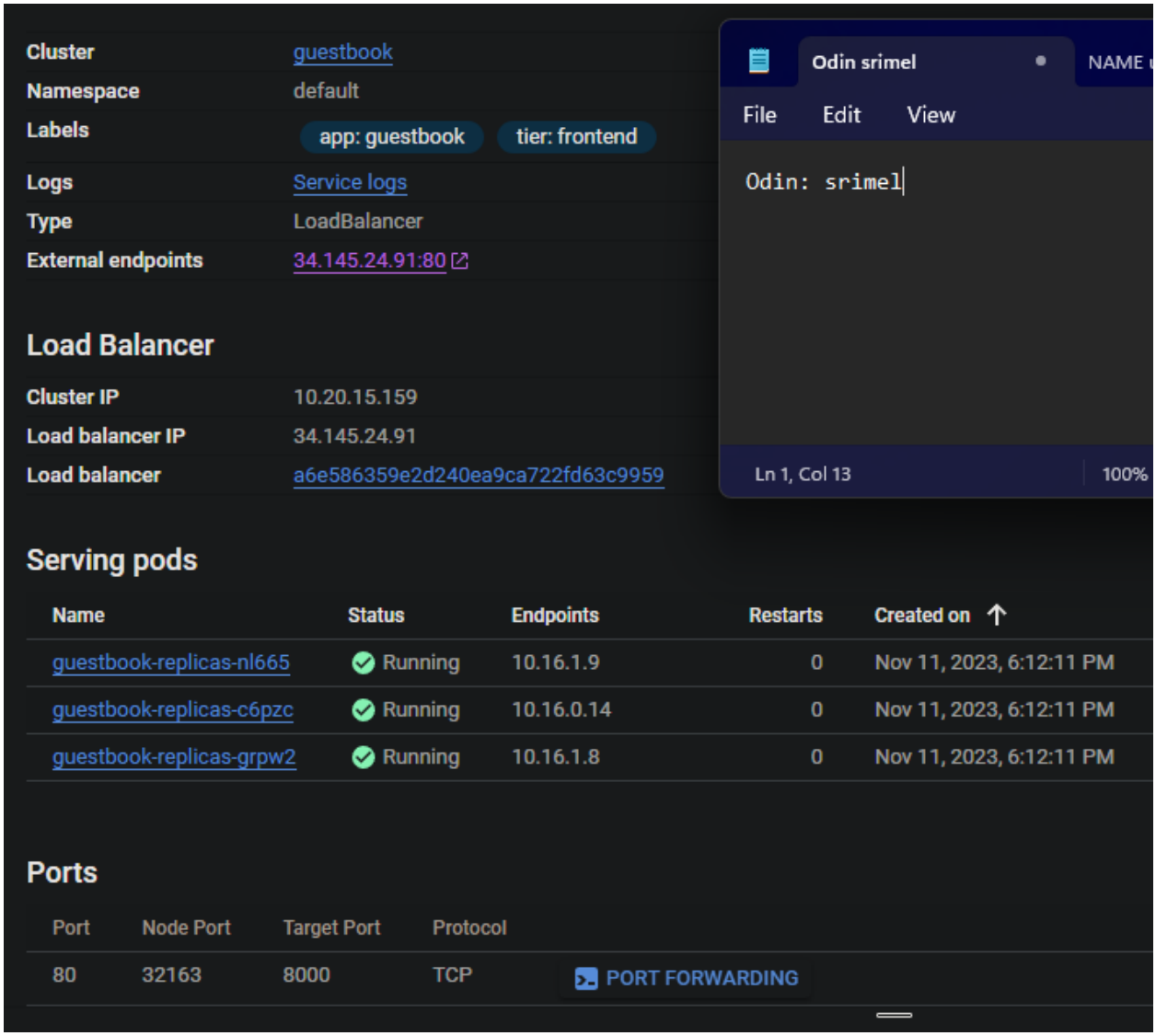
```
srimel@cloudshell:~/cs430-src/05_gcp_datastore (cloud-rimel-srimel)$ kubectl get services
NAME           TYPE           CLUSTER-IP     EXTERNAL-IP    PORT(S)        AGE
guestbook-lb   LoadBalancer   10.20.15.159   34.145.24.91   80:32163/TCP   2m7s
kubernetes     ClusterIP      10.20.0.1      <none>         443/TCP        24m
srimel@cloudshell:~/cs430-src/05_gcp_datastore (cloud-rimel-srimel)$
```

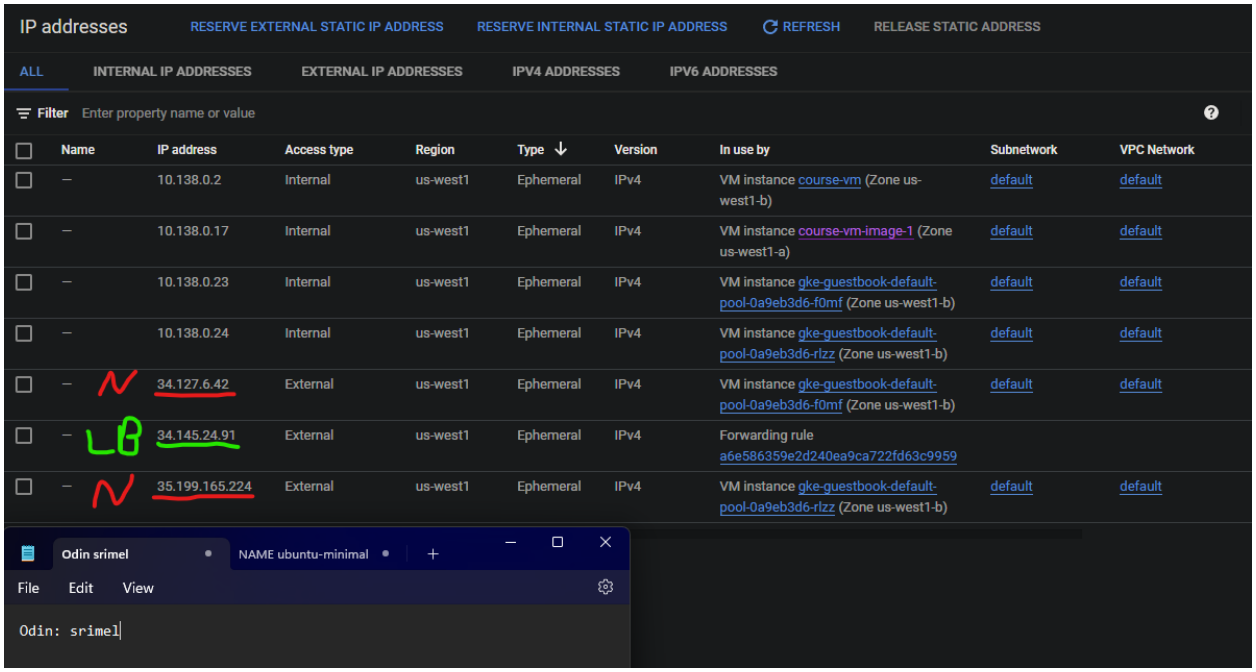
7.2g.8: View the Guestbook



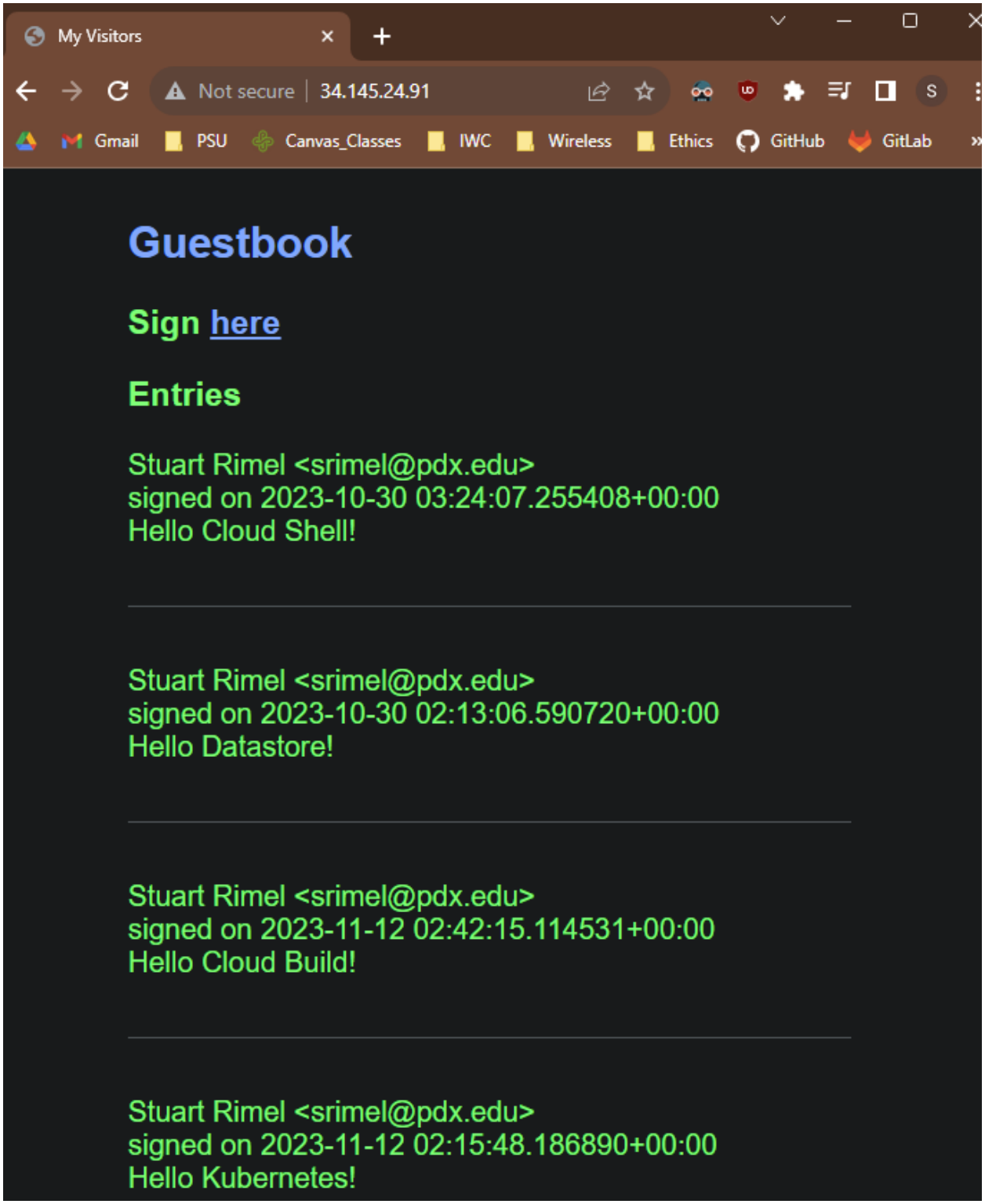

Managed pods
Name
Status
Restarts
Created on
guestbook-replicas-nl665
Running
0
Nov 11, 2023, 6:12:11 PM
guestbook-replicas-c6pzc
Running
0
Nov 11, 2023, 6:12:11 PM
guestbook-replicas-grpw2
Running
0
Nov 11, 2023, 6:12:11 PM
Exposing services
Name
Type
Endpoints
guestbook-lb
Load balancer
34.145.24.91:80
Odin srimel
File
Edit
View
Odin: srimel
Ln 1, Col 13


a6e586359e2d240ea9ca722fd63c9959
Target-pool Network Load Balancer
Frontend
Protocol
IP version
IP:Port
Network Tier
TCP
IPv4
34.145.24.91:80
Premium
Backend
Name
Region
Health check
a6e586359e2d240ea9ca722fd63c9959
us-west1
k8s-5e152c4c353c7d80-node
ADVANCED CONFIGURATIONS
Instance
Zone
gke-guestbook-default-pool-0a9eb3d6-f0mf
us-west1-b
gke-guestbook-default-pool-0a9eb3d6-rlzz
us-west1-b
Odin srimel
NAME ubuntu-minimal
File
Edit
View
Odin: srimel


Cluster guestbook
Namespace default
Labels app: guestbook tier: frontend
Logs Service logs
Type LoadBalancer
External endpoints 34.145.24.91:80
Load Balancer
Cluster IP 10.20.15.159
Load balancer IP 34.145.24.91
Load balancer a6e586359e2d240ea9ca722fd63c9959
Odin srimel
File Edit View
Odin: srimel
Ln 1, Col 13
100%
Serving pods
Name Status Endpoints Restarts Created on
guestbook-replicas-nl665 Running 10.16.1.9 0 Nov 11, 2023, 6:12:11 PM
guestbook-replicas-c6pzc Running 10.16.0.14 0 Nov 11, 2023, 6:12:11 PM
guestbook-replicas-grpw2 Running 10.16.1.8 0 Nov 11, 2023, 6:12:11 PM
Ports
Port Node Port Target Port Protocol
80 32163 8000 TCP
PORT FORWARDING


IP addresses
RESERVE EXTERNAL STATIC IP ADDRESS
RESERVE INTERNAL STATIC IP ADDRESS
REFRESH
RELEASE STATIC ADDRESS
ALL
INTERNAL IP ADDRESSES
EXTERNAL IP ADDRESSES
IPV4 ADDRESSES
IPV6 ADDRESSES
Filter
Enter property name or value
Name
IP address
Access type
Region
Type
Version
In use by
Subnetwork
VPC Network
– 10.138.0.2 Internal us-west1 Ephemeral IPv4 VM instance course-vm (Zone us-west1-b) default default
– 10.138.0.17 Internal us-west1 Ephemeral IPv4 VM instance course-vm-image-1 (Zone us-west1-a) default default
– 10.138.0.23 Internal us-west1 Ephemeral IPv4 VM instance gke-guestbook-default-pool-0a9eb3d6-f0mf (Zone us-west1-b) default default
– 10.138.0.24 Internal us-west1 Ephemeral IPv4 VM instance gke-guestbook-default-pool-0a9eb3d6-rlzz (Zone us-west1-b) default default
– 34.127.6.42 External us-west1 Ephemeral IPv4 VM instance gke-guestbook-default-pool-0a9eb3d6-f0mf (Zone us-west1-b) default default
– LB 34.145.24.91 External us-west1 Ephemeral IPv4 Forwarding rule a6e586359e2d240ea9ca722fd63c9959
– 35.199.165.224 External us-west1 Ephemeral IPv4 VM instance gke-guestbook-default-pool-0a9eb3d6-rlzz (Zone us-west1-b) default default
Odin srimel
NAME ubuntu-minimal
File
Edit
View
Odin: srimel

7.2g.12: Deploy and view the application



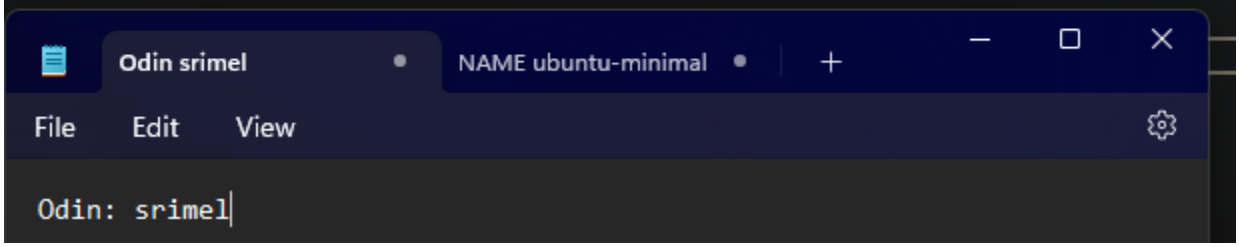
7.3g: APIs (Slack, Knowledge Graph)

7.3g.2: Code

1. Google does not provide a python package specifically for accessing the knowledge graph API

7.3g.3: Code

```
# [START functions_slack_search]
@functions_framework.http
def kg_search(request):
    if request.method != "POST":
        return "Only POST requests are accepted", 405

    verify_signature(request)
    kg_search_response = make_search_request(request.form["text"])
    return jsonify(kg_search_response)
```




```
# [START functions_slack_request]
def make_search_request(query):
    req = kgsearch.entities().search(query=query, limit=1)
    res = req.execute()
    return format_slack_message(query, res)
```

Name of method that sends the query to Knowledge Graph API: make_search_request

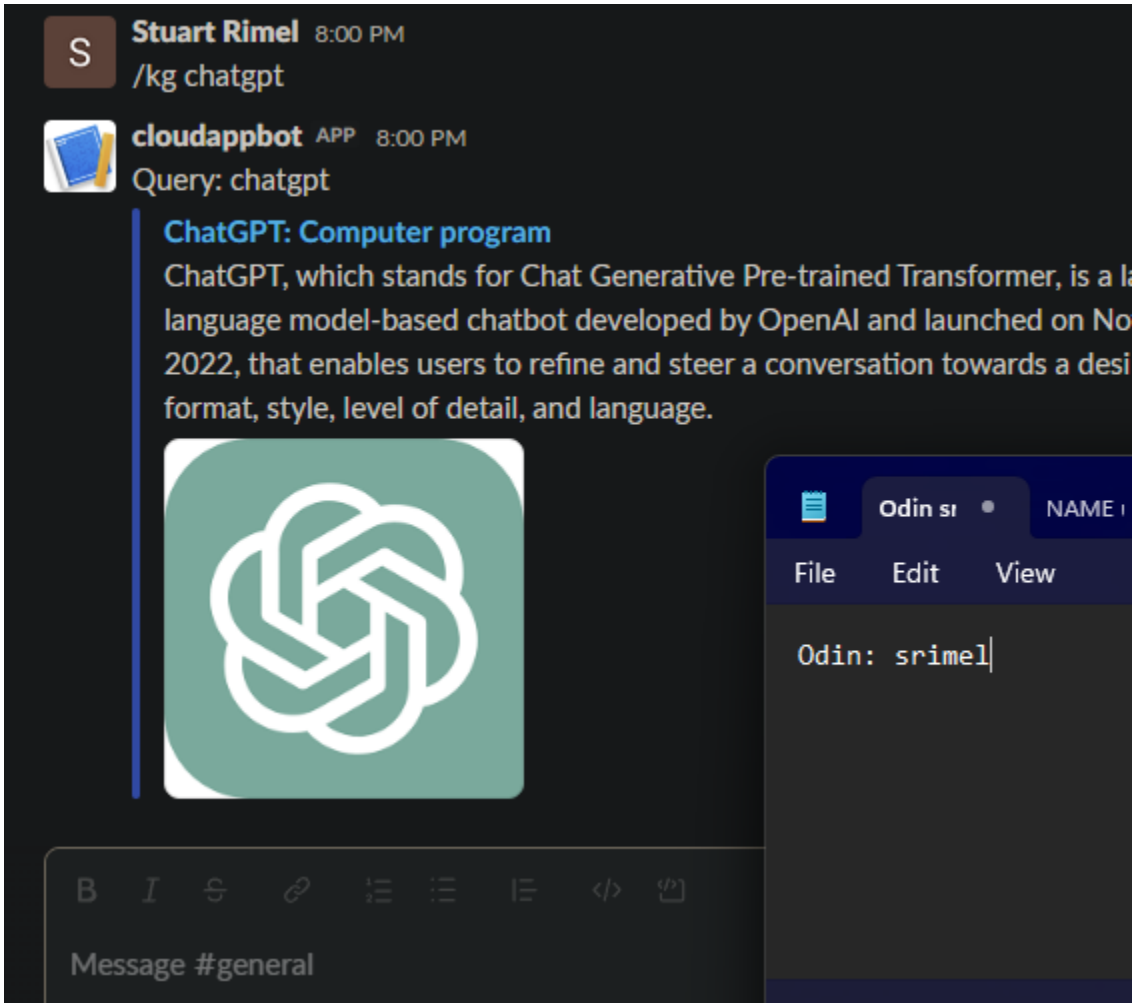
- Uses the kgsearch client object created from discovery package.

Python data type for slack message: Dict

Three main attributes of the formatted message passed back to Slack:

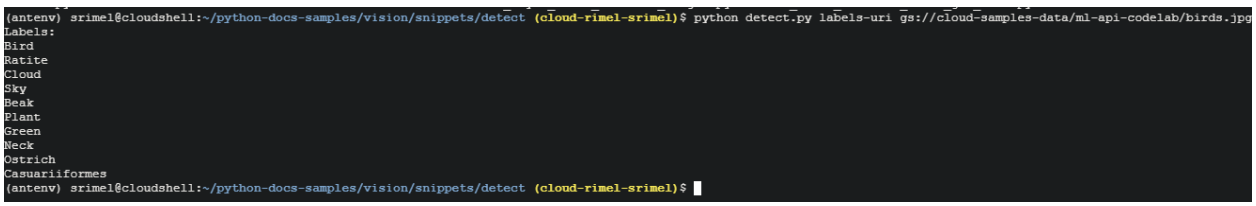
- response_type, text, attachments

7.3g.8: Test the command



7.4g: ML APIs

7.4g.3: Vision

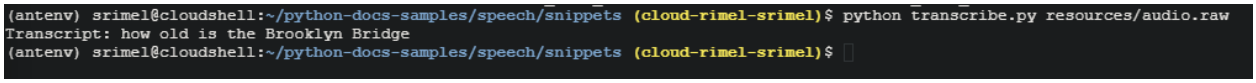


1. Name of function: detect_labels_uri

2. ImageAnnotatorClient
3. Function: label_detection
4. Attribute: label_annotations

5. What method for logos: detect_logos

7.4g.4: Speech

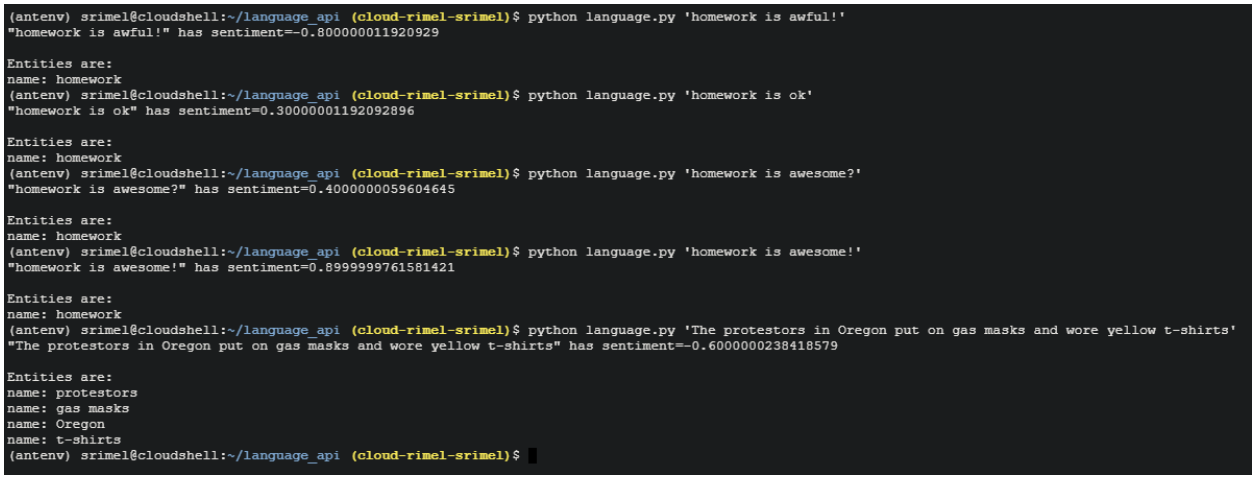


1. Function name: transcribe_file
2. Method to perform detection: recognize
3. Attribute in response object: results

7.4g.5: Translate

1. Name of function: translate_text
2. Name of method invoked for translation: translate
3. Attribute in response contains results: tranlsatedText

7.4g.6: Natural Language



7.4g.8: Code

1. Function that does transcription: transcribe_gcs
2. Function that performs translation: translate_text
3. Function that performs entity analysis on the translation: entities_text
4. Function that performs entity analysis on the image: detect_labels_uri

7.4g.9: Test integration

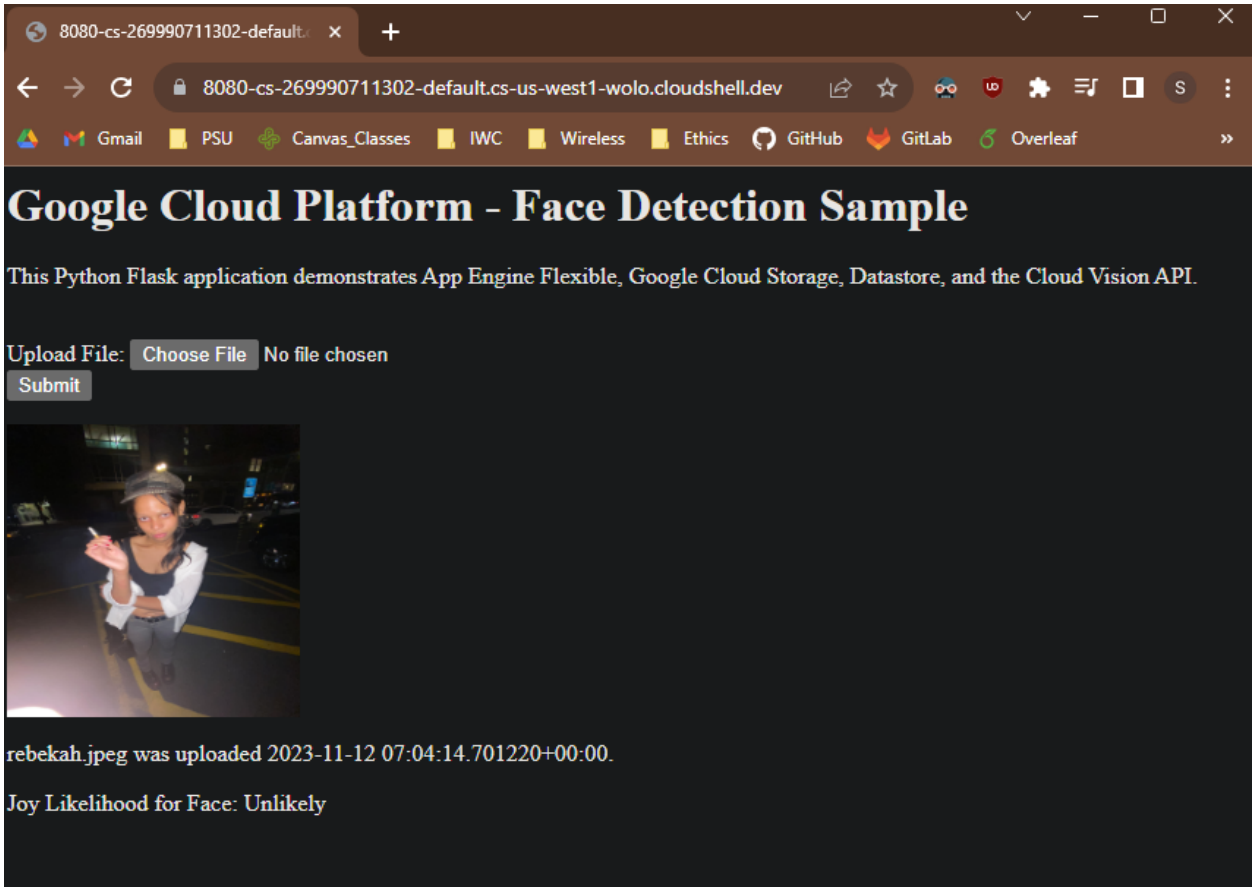
1. There seems to be an issues within the “naive check”. This is coming from an issue with the case of strings, the transcription/translation creates a lowercase ‘football’, but within the image analysis it is listed at ‘Football’. This is fixed by making the naive check case insensitive.
2. The problem seems to be that the audio transcription is translated to the short slang of bicycle ‘bike’, but that entity doesn’t exist within the image analysis. To change this you would have to do something within the image analysis to include ‘bike’ somehow.
3. The transcription returns ostriches, but the image analysis gives ostrich. The naive check once again fails and must be updated to account for pluralities.

7.4g.13: Video Intelligence

1. Three highest labels
    a. Sports: confidence = 0.9218811392784119
    b. Basketball: confidence = 0.9137870669364929
    c. Player: confidence = 0.8446521162986755

2. VideoIntelligenceServiceClient
3. annotate_video

7.4g.16: Application



7.4g.17: Code

1.
```
query = datastore_client.query(kind="Faces")
```
2.
```
image_entities = list(query.fetch())
```
3.
```
CLOUD_STORAGE_BUCKET = os.environ.get("CLOUD_STORAGE_BUCKET")
```
4. File?
```
photo = request.files["file"]
```

5.
```
# Create a new blob and upload the file's content.
blob = bucket.blob(photo.filename)
blob.upload_from_string(photo.read(), content_type=photo.content_type)
```
6.
```
faces = vision_client.face_detection(image=image).face_annotations
```
7. blob_name, image_public_url, timestamp, joy
8. Redirects to /