Cloud Computing
(SOFE 4630U)

Project Milestone 4: Microservices

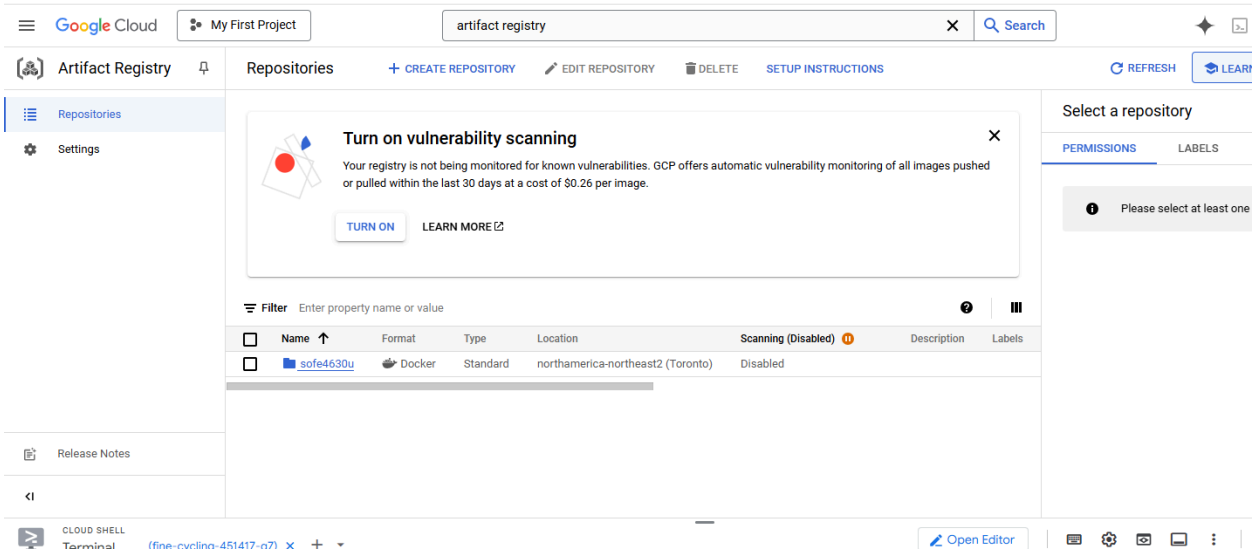
Name	Student ID
Rohan Radadiya	100704614

Due Date: March 14, 2025

GitHub Link:
https://github.com/rohanradadiya/Cloud-Computing-Project-MS-4
Google Drive for Videos:
https://drive.google.com/drive/folders/16M---vBvUWhzOViHeLaZrygLHYa5MgHI?usp=sharing

Screenshots:

Setting up the GCP Project:



Created a repository within the Artifact Registry page, called “sofe4630u” (After creating a topic and service account key):

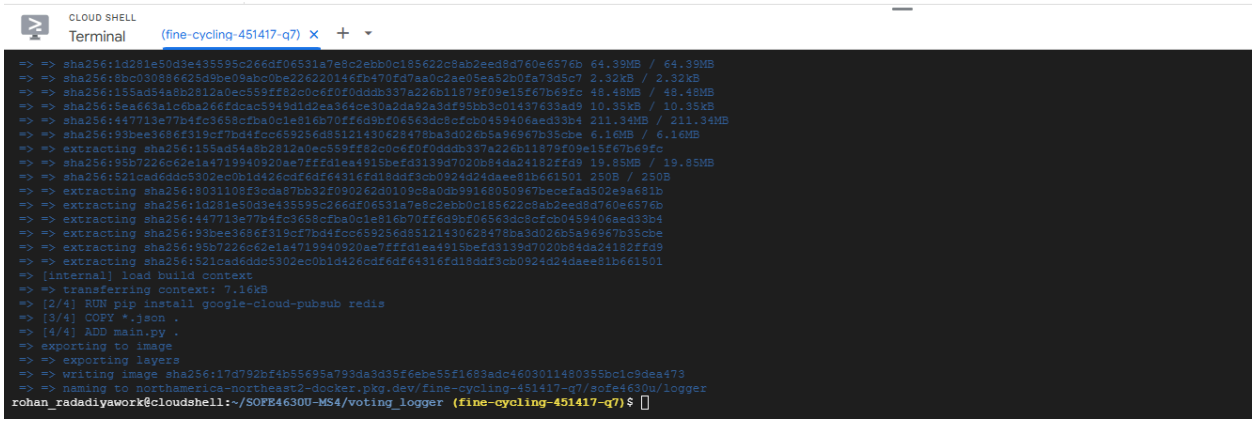
Copied the repo path and saved it in an environment variable:

The Logger Service:

Cloned the GitHub repository in the GCP console:

```
rohan_radadiyawork@cloudshell:~ (fine-cycling-451417-q7)$ mv ~/fine-cycling-451417-q7-b520b47cb17f.json ~/SOFE4630U-MS4/voting_logger/
rohan_radadiyawork@cloudshell:~ (fine-cycling-451417-q7)$ ls ~/SOFE4630U-MS4/voting_logger/
docker-compose.yaml  Dockerfile  fine-cycling-451417-q7-b520b47cb17f.json  logger.yaml  main.py
rohan_radadiyawork@cloudshell:~ (fine-cycling-451417-q7)$
```

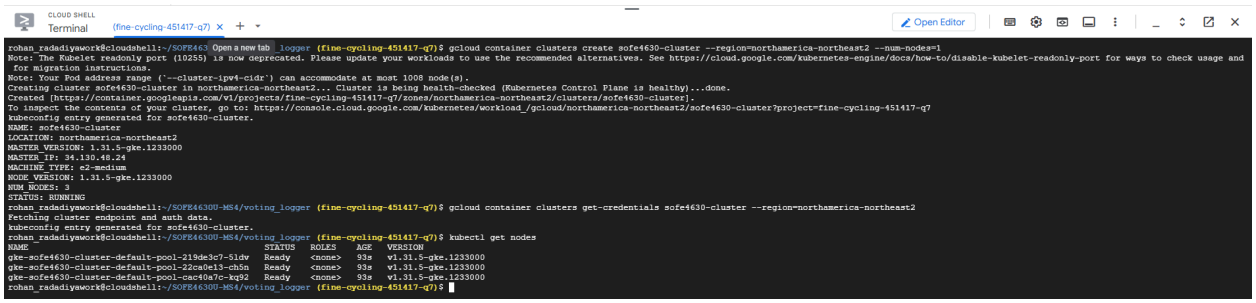
Uploaded the previously created JSON file and moved it to the correct directory:



Executed the instruction in the Dockerfile and generated the image:

```
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ docker push $LOGGER_IMAGE
Using default tag: latest
The push refers to repository [northamerica-northeast2-docker.pkg.dev/fine-cycling-451417-q7/sofe4630u/logger]
e0ed0d4be92d: Pushed
b83814db9497: Pushed
ff9c79e0a683: Pushed
01db3e67097a: Pushed
e49d0c94aa2a: Pushed
1c86760c5c93: Pushed
4b017a36fd9c: Pushed
20a9b386e10e: Pushed
f8217d7865d2: Pushed
01c9a2a5f237: Pushed
latest: digest: sha256:d557f3a1224099cc309468b9f068387370374689cee7fc9fe5783e6f83a98663 size: 2424
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$
```

Pushed the Docker image into the artifact repository for use in a Kubernetes deployment:



Had to create a new cluster and connected kubectl to the newly created cluster:

```
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ PROJECT=$PROJECT LOGGER_IMAGE=$LOGGER_IMAGE envsubst < logger.yaml | kubectl apply -f -
deployment.apps/logger-deployment created
service/redis created
deployment.apps/redis-deployment created
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$
```

Service and Redis server deployed:

```
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ kubectl get pods
kubectl logs <pod-name>
NAME                                 READY   STATUS    RESTARTS   AGE
logger-deployment-5c697ccf6d-nv69p   1/1     Running   0          77s
logger-deployment-5c697ccf6d-q96bv   1/1     Running   0          77s
logger-deployment-5c697ccf6d-rrsbc   1/1     Running   0          77s
redis-deployment-7b7f55cb55-2k94f    1/1     Running   0          77s
-bash: syntax error near unexpected token `newline'
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ kubectl logs logger-deployment-5c697ccf6d-q96bv
Connected to Redis
Listening for messages on projects/fine-cycling-451417-q7/subscriptions/election-submit-sub..

rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$
```

Checking the deployment:

The Voting Recorder Service:

```
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ cp ~/fine-cycling-451417-q7-b520b47cb17f.json ~/SOFE4630U-MS4/voting_record/
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ ls ~/SOFE4630U-MS4/voting_record/
docker-compose.yaml  Dockerfile  fine-cycling-451417-q7-b520b47cb17f.json  main.py  postgres  recorder.yaml
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$
```

Copied JSON file to the path “~/SOFE4630U-MS4/voting_record”:

```
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ REPO=northamerica-northeast2-docker.pkg.dev/fine-cycling-451417-q7/sofe4630u
RECORDER_IMAGE=$REPO/recorder
echo $RECORDER_IMAGE
northamerica-northeast2-docker.pkg.dev/fine-cycling-451417-q7/sofe4630u/recorder
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$
```

Generated full name of the service image:

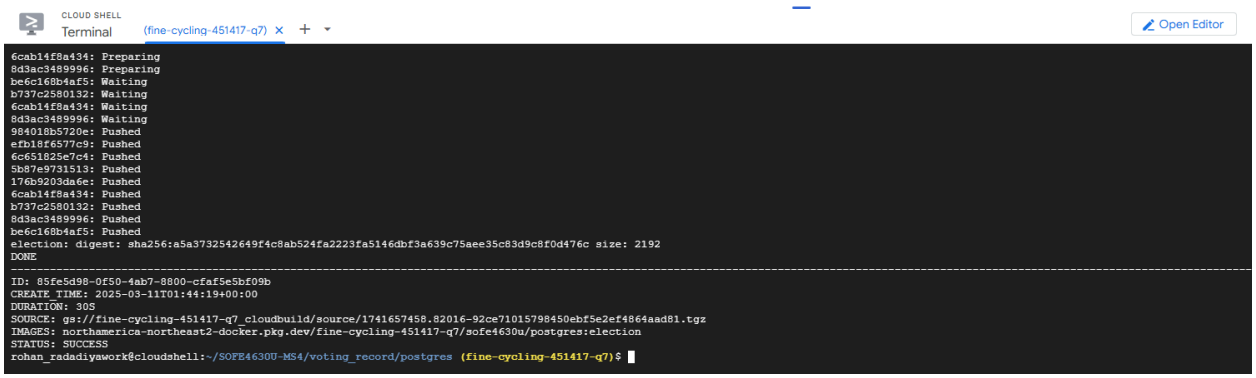
```
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_logger (fine-cycling-451417-q7)$ cd ~/SOFE4630U-MS4/voting_record
ls
docker-compose.yaml  Dockerfile  fine-cycling-451417-q7-b520b47cb17f.json  main.py  postgres  recorder.yaml
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_record (fine-cycling-451417-q7)$
```

Making sure that the path contains all of the necessary files:

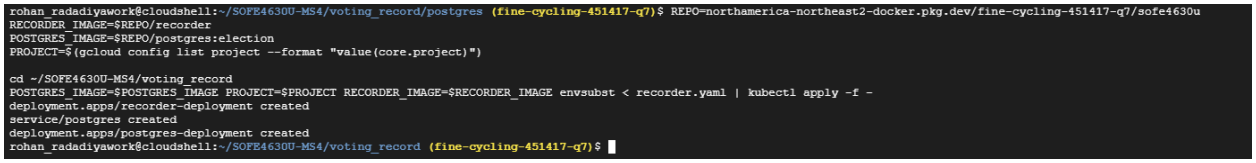
```
CLOUD SHELL
Terminal   (fine-cycling-451417-q7)                                                                                    Open Editor

1c86760c5c93: Waiting
4b017a36fd9c: Waiting
20a9b386e10e: Waiting
f8217d7865d2: Waiting
01c9a2a5f237: Waiting
e49d0c94aa2a: Layer already exists
01db3e67097a: Layer already exists
1c86760c5c93: Layer already exists
4b017a36fd9c: Layer already exists
f8217d7865d2: Layer already exists
20a9b386e10e: Layer already exists
3d4a3fd9ce77: Pushed
78fbdde96cd3: Pushed
89ad3882febc: Pushed
01c9a2a5f237: Pushed
latest: digest: sha256:d18d4728712479a3e8806361bd219a17c9fc9a2804fd7139ecbaa56f83b04397 size: 2424
DONE
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
ID: ef667b1d-45b7-471a-b280-11224d96a108
CREATE_TIME: 2025-03-11T01:41:29+00:00
DURATION: 51S
SOURCE: gs://fine-cycling-451417-q7_cloudbuild/source/1741657289.337474-82299e450eb545d7b6b82bcc047c966c.tgz
IMAGES: northamerica-northeast2-docker.pkg.dev/fine-cycling-451417-q7/sofe4630u/recorder (+1 more)
STATUS: SUCCESS
rohan_radadiyawork@cloudshell:~/SOFE4630U-MS4/voting_record (fine-cycling-451417-q7)$
```

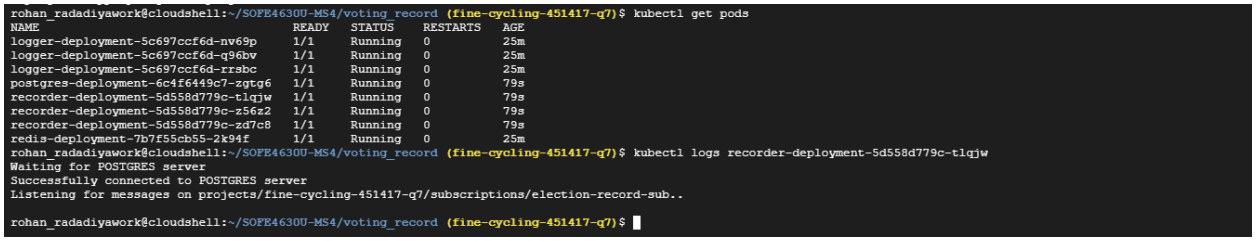
Executed the instruction in the Dockerfile to generate the image and push it to the artifact repository:



Created and pushed the Docker image to the repository:

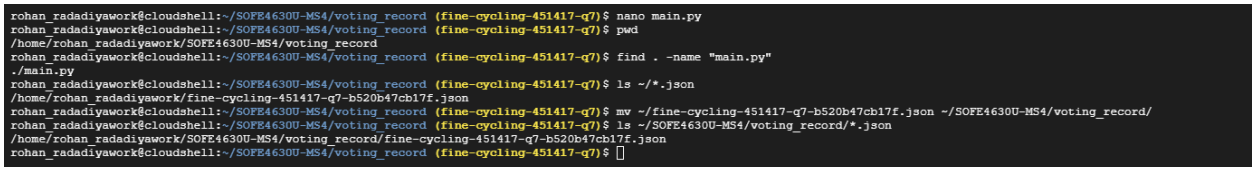


Deployed the service and the PostgreSQL server:

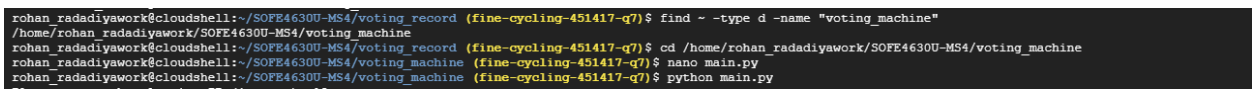


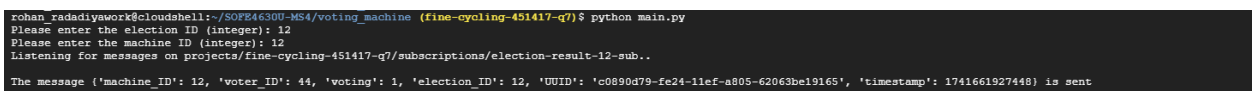
Ensured that pods are available and printed a pod's logs:

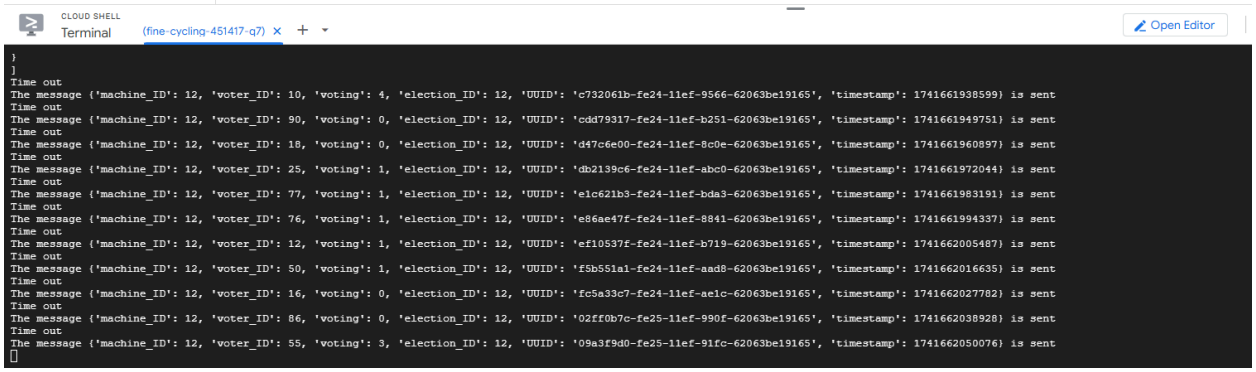
The Voting Machine:



Edited the main.py file and moved the JSON key into the same directory as the main.py file:







Receiving all of the message notifications in the output now:

Discussion:

Compare the advantages and disadvantages of using Dataflow vs microservices in preprocessing the smart reading.

There are various aspects to consider when discussing Dataflow and microservices in preprocessing the smart reading. For example, dataflow offers the advantage of simplifying the processing of large-scale data in real time by leveraging fully managed services. It allows for easy handling of stream processing tasks like data ingestion, transformation, and analytics without the need for manual scaling or infrastructure management. Also, Dataflow integrates with other cloud services. However, some disadvantages are less flexibility in certain cases, potential latency, and even the complexity of managing costs depending on usage.

On the other hand, microservices provide a modular approach to application design because of breaking down complex systems into smaller and independently deployed components. This allows for better scalability, fault tolerance, and ease of updates since each microservice can evolve separately. Some disadvantages of this can be challenges in service communication, orchestration, and monitoring. There may also be challenges in deploying and maintaining each microservice, which can also increase the complexity of the overall architecture.