Assignment 1, Cloud Computing

Exercise 1: Understanding Cloud Computing Models

Questions:

- *What are the main differences between IaaS, PaaS, and SaaS?*

Infrastructure as a Service (IaaS)

IaaS gives users access to computing resources like servers, storage, and virtualization. The provider maintains the hardware, while customers are responsible for managing their software, such as OS, databases and apps.

Platform as a Service (PaaS)

PaaS provides the tools and environment needed to develop, test, and run applications. The cloud provider takes care of the infrastructure, including hardware and software.

Software as a Service (SaaS)

SaaS offers complete software applications that customers can access through the internet. These applications are fully managed by the cloud provider.

	Infrastructure as a Service (IaaS)	**Platform as a Service (PaaS)**	**Software as a Service (SaaS)**
Control	High control over Infrastructure. Customers manage OS, and networking infrastructure.	Less control over infrastructure, focus on development.	Minimal control over instrastrucure. Provider manages everything.
Flexibility	High flexibility to configure network settings, storage and so on	Average flexibility, more focus on development	Low flexibility, customization is limited
Use Cases	Virtual machines	Testing and Deployment of Apps, apps hosting	CRM systems

IaaS provides the most control, taking care of things like the OS and apps, while the cloud provider manages the physical servers and hardware. **PaaS** simplifies things by handling most of the infrastructure, so customers can focus on building and running their apps without worrying about the backend. In **SaaS** the provider takes care of everything from infrastructure to software.

- *Which GCP services fall under each of these models?*

IaaS - Cloud Storage, Compute Engine

PaaS - Cloud SQL, App Engine

SaaS - BigQuery, Workspace

- *Provide a real-world example where each cloud service model might be the most appropriate choice.*

IaaS - Google Cloud Storage

For example, we have a media company. So we need to store and retrieve a large amount of unstructured data, such as photos and videos.

And Google Cloud Storage is a fine scalable decision, as the data grows everyday, and would be good for media files, backups, and logs. It gives users control over how data is stored and organized, can be accessed from everywhere and supports various storage classes.

PaaS - Google Cloud SQL

For instance, we have an e-commerce app and we need a relational db to store data on orders, user information and so on.

Google Cloud SQL is a satisfactory option for managing databases in the cloud because it's easy to use, can grow with your needs, and works well with other Google Cloud services. Also, it supports databases like MySQL, PostgreSQL, making it flexible for different apps. It means businesses can move their current databases to the cloud or create new ones using the database they prefer.

SaaS - Google BigQuery

As an example, a healthcare company wants to analyze patient data from multiple sources, such as electronic devices, lab results, and insurance claims. And they need to identify trends in patient outcomes and improve treatment.

BigQuery can be a good decision because of serverless architecture, so the team can be more focused on analysis rather than system maintenance. Also, it is a good option for running complex queries in order to analyze data. And can be easily integrated with Visualization tools.

Exercise 2: Exploring Google Cloud Platform's Core Services

Questions:

- *What is the primary use case of Compute Engine?*

Google Compute Engine is mostly used for running VMs in the cloud. It is good for hosting websites, running apps, or performing computing tasks like machine learning.

- *How does Google Kubernetes Engine (GKE) simplify the management of containerized applications?*

For example, It automatically scales an app when traffic increases. So briefly it does lots of work on its own, like monitoring, setting up, and scaling.

- *What advantages does Cloud Storage offer for data management?*

Diversity of data types, starting from photos ending with datasets. Also it is scalable and reliable. Moreover, replication of data is available.

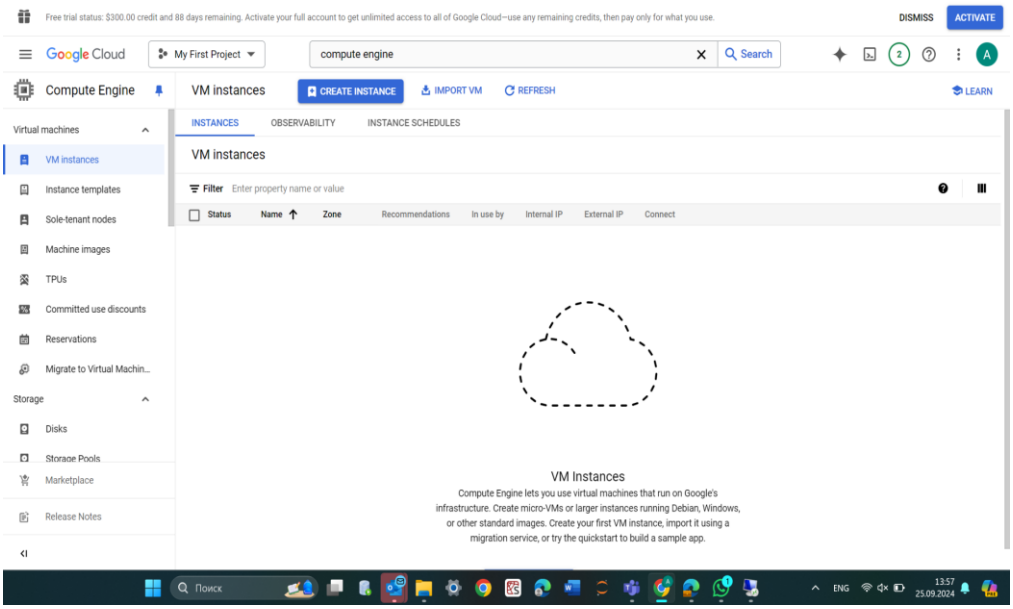
- *Why would a business choose BigQuery for their data analysis needs?*

It is a good option for the business when a large amount of data is needed to be analyzed. Complicated queries are run quickly, so it can be good for building reports.

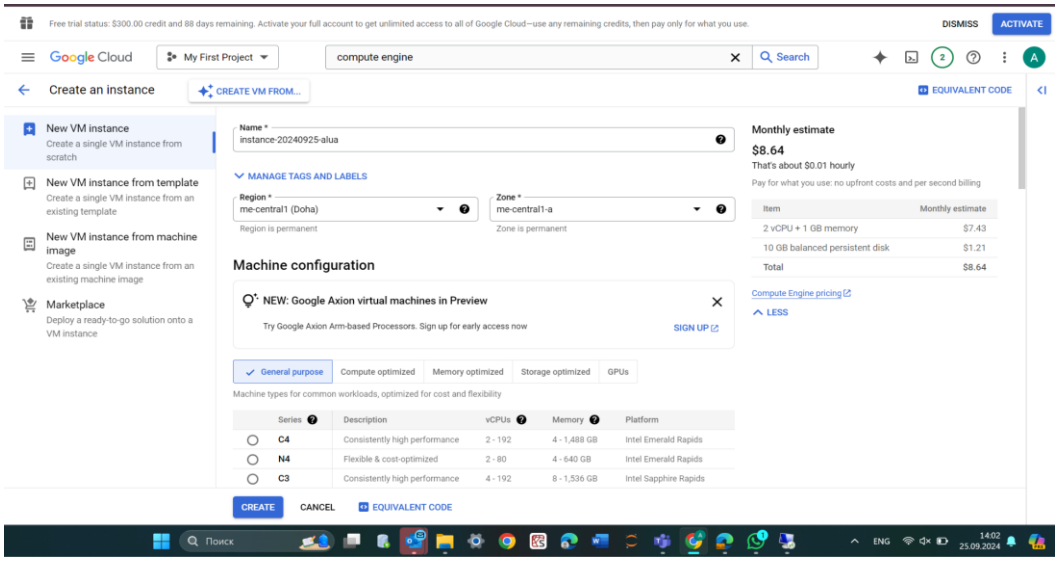
Exercise 3: Creating and Managing Virtual Machines with Compute Engine

1. **Objective**: Learn how to create, manage, and interact with virtual machines (VMs) using Compute Engine.
2. **Steps**:
   - In the Google Cloud Console, navigate to Compute Engine and create a new VM instance.
   - Configure the VM with specific parameters, such as the machine type, region, and operating system.
   - Connect to the VM using SSH and install a basic web server (e.g., Apache or Nginx).
   - Stop, start, and delete the VM through the console.
3. **Questions**:
   - What steps did you follow to create the VM?

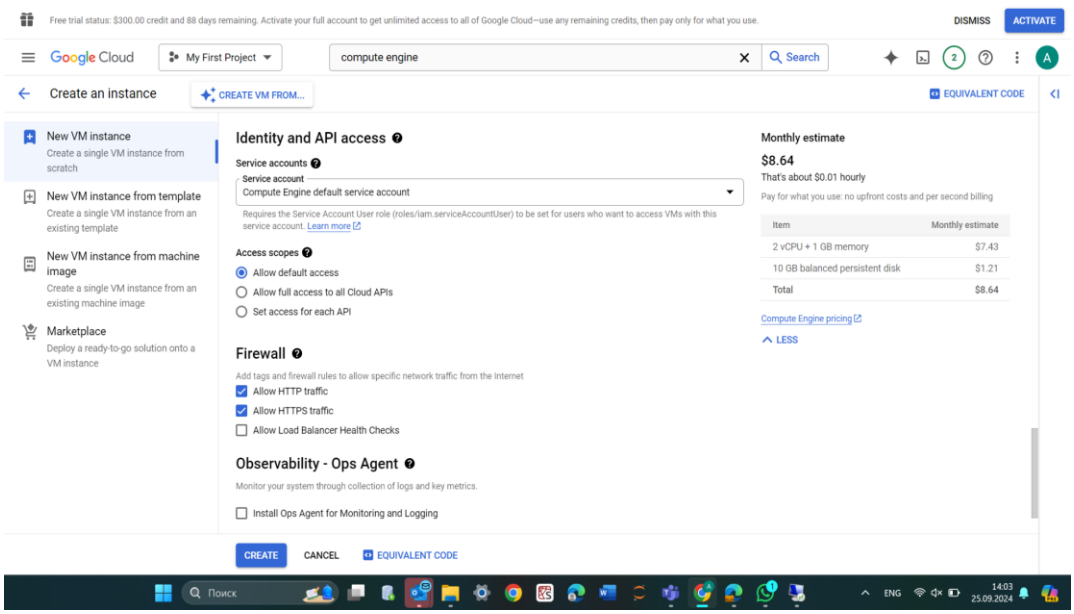
1) Press ***'Create instance'*** button



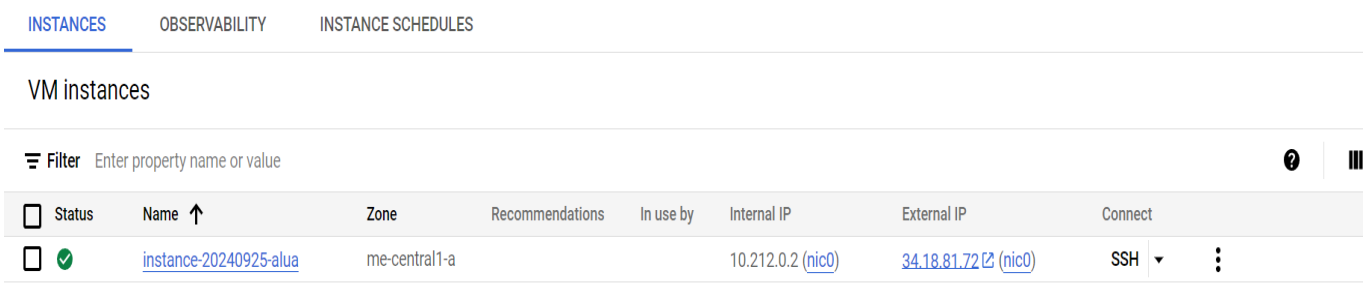
2) Edit configurations



3) Allow HTTP, HTTPS traffic

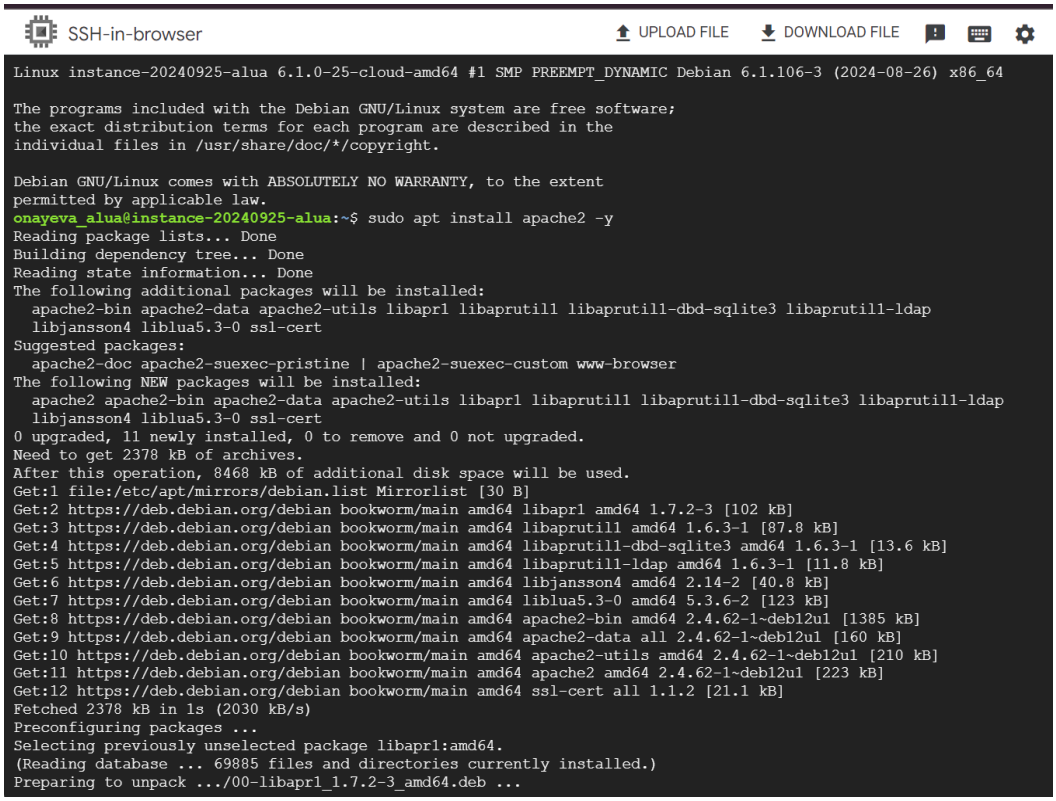


4) Instance is created, and I'm pressing SSH

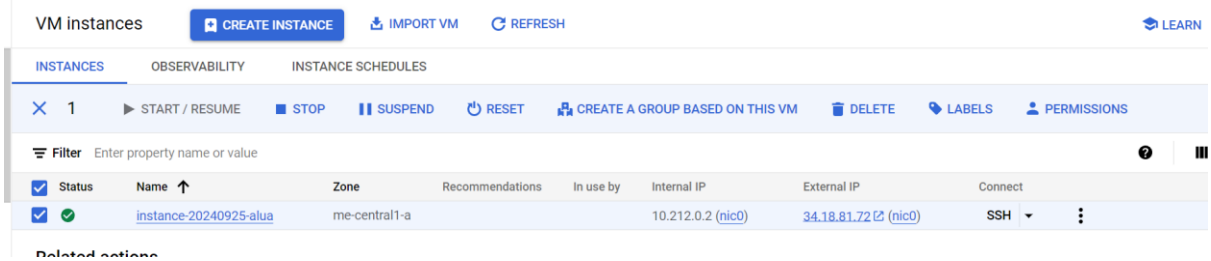


- How did you connect to the VM, and what commands did you use to install the web server?

After pressing to SSH, I install apache



So here I can stop, start and delete VM



- What happens to the VM and its data when it is stopped versus when it is deleted?

So, when VM is stopped, we are not billed for CPU usage, but will be billed for boot disk. And resources will be available. When the VM is deleted, boot disk and all resources will be deleted and no longer available unless we have created a snapshot.

Exercise 4: Deploying a Containerized Application on Google Kubernetes Engine (GKE)

1. **Objective**: Understand how to deploy and manage containerized applications using Google Kubernetes Engine.
2. **Steps**:
   - Create a simple Docker container for a web application.
   - Push the container image to Google Container Registry (GCR).
   - Create a GKE cluster in Google Cloud Console.
   - Deploy the containerized application to the GKE cluster.
   - Expose the application to the internet and verify its accessibility.

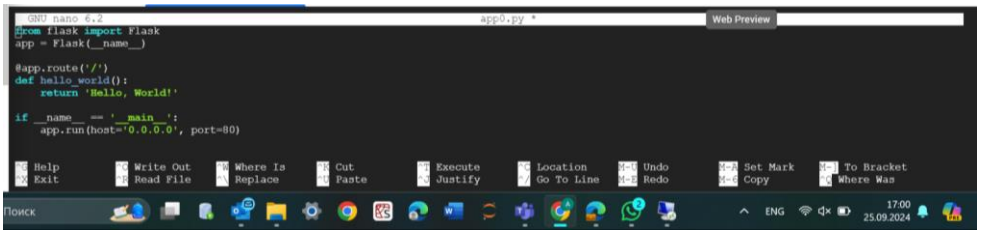
3. **Questions**:

   - How did you create and push the Docker container to GCR?

Create **alua-project** directory:

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to psyched-bonfire-436413-e2.
Use “gcloud config set project [PROJECT_ID]” to change to a different project.
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ mkdir alua-project
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ cd alua-project
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$ nano app0.py
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$
```

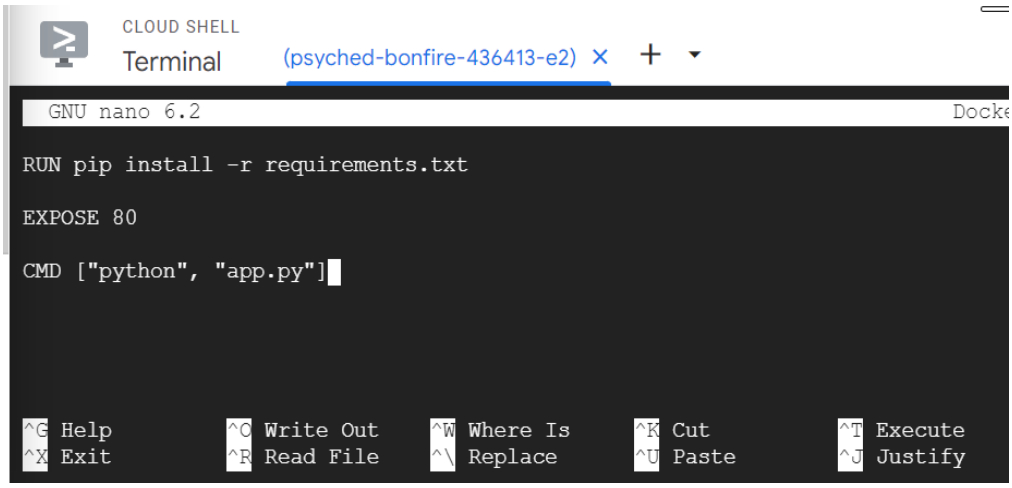
Simple Hello World Flask program:




```
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$ nano app0.py
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$ nano requirements.txt
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$
```

Dockerfile

```
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$ nano Dockerfile
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$
```



Structure of alua-project:

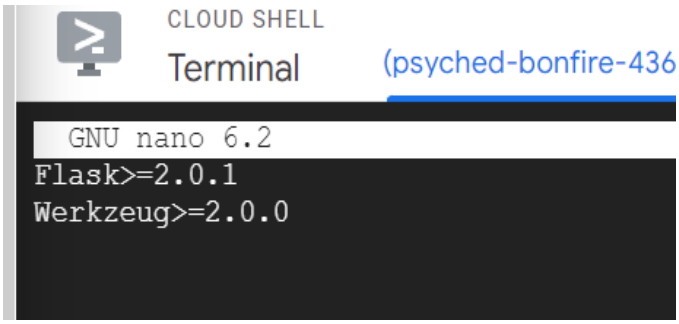
- app.py
- requirements.txt
- dockerfile

Building docker image

docker build -t alua-project .

```
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$ docker build -t alua-project .
[+] Building 13.1s (9/9) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 159B
 => [internal] load metadata for docker.io/library/python:3.8-slim
 => [internal] load .dockerignore
 => => transferring context: 2B
```

Requirements.txt



Docker container is running the app, but it's serving on port inside the container

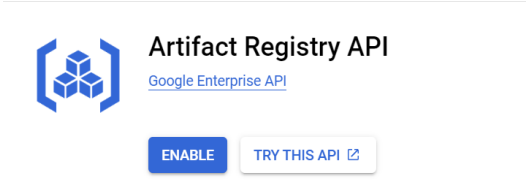
docker run -p 8080:80 alua-project

```
onayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$ docker run -p 8080:80 alua-project
 * Serving Flask app 'app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:80
 * Running on http://172.17.0.2:80
Press CTRL+C to quit
^Conayeva_alua@cloudshell:~/alua-project (psyched-bonfire-436413-e2)$
```

project number (later i don’t need it)

```
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ gcloud projects list
PROJECT_ID: psyched-bonfire-436413-e2
NAME: My First Project
PROJECT_NUMBER: 545668553047
```

Need to enable this API, to create artifacts like images and etc.



We need project number not project id.

docker tag alua-project gcr.io/psyched-bonfire-436413-e2/alua-project:v1 (Tagging Docker image with the Google Container Registry (GCR) format)

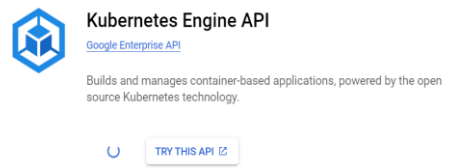
gcloud auth configure-docker (configure Docker to authenticate with GCR)

docker push gcr.io/psyched-bonfire-436413-e2/alua-project:v1 (pushing Docker)

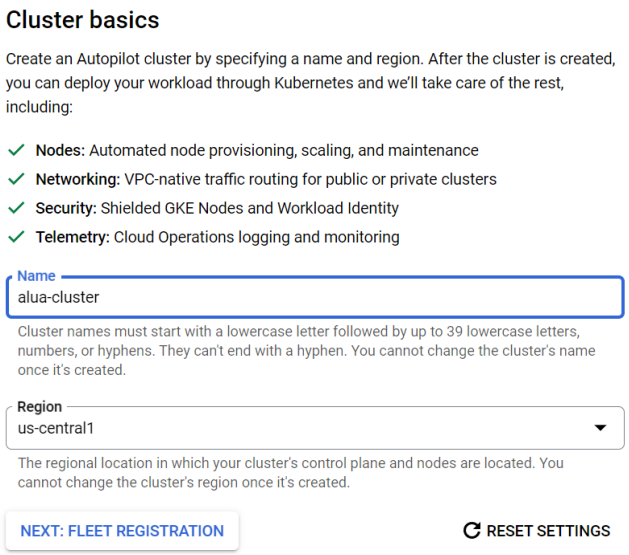
```
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ docker tag alua-project gcr.io/psyched-bonfire-436413-e2/alua-project:v1
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ docker push gcr.io/psyched-bonfire-436413-e2/alua-project:v1
The push refers to repository [gcr.io/psyched-bonfire-436413-e2/alua-project]
596575d77cb5: Pushed
aa77a101547b: Pushed
ec6b39214c20: Pushed
de06647e8a8c: Layer already exists
837964438a9e: Layer already exists
8ffef0ea5e5c: Layer already exists
8e2ab394fabf: Layer already exists
```

- What steps were involved in setting up the GKE cluster?

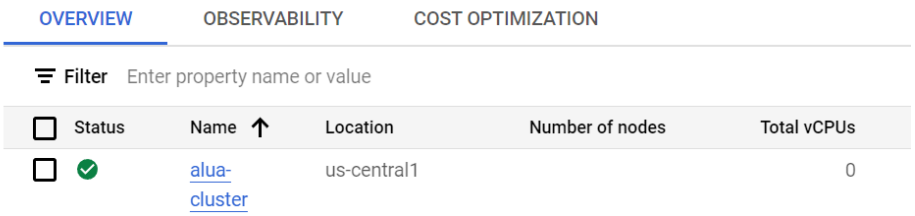
Enable Kubernetes Engine API.



We need to create a cluster



Cluster is created.



gcloud container clusters get-credentials alua-cluster --zone us-central1 --project psyched-bonfire-436413-e2

Set up VM to use the GKE cluster

```
ire-436413-e2cloudshell:~ (psyched-bonfire-436413-e2)$ gcloud container clusters get-credentials alua-cluster --zone us-central1 --project psyched-bon
Fetching cluster endpoint and auth data.
kubeconfig entry generated for alua-cluster.
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$
```

Deployment configuration

```
Once finished, enter the verification code provided in your browser: 4/0AQlEd8zqE_57H8CzDoBJP07Ik0WLokE8FZf48QWtzNZ3hFo0IByuXVttcRF8LrT

  GNU nano 6.2                        deployment.yaml *
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: web-app
  template:
    metadata:
      labels:
        app: web-app
    spec:
      containers:
      - name: web-app
        image: gcr.io/[YOUR_PROJECT_ID]/my-web-app:v1
        ports:
        - containerPort: 80

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

Firstly, I got an error that I have insufficient privileges, while deploying.
But when I changed my role to Kubernetes Engine Admin, everything worked.

Filter Enter property name or value

Type	Principal	Name	Role	Security insights
	545668553047-compute@developer.gserviceaccount.com	Compute Engine default service account	Editor	
	onayeva.alua@gmail.com	Alua Onayeva	Editor Kubernetes Engine Admin Owner	

kubectl apply -f deployment.yaml (App deploy)

```
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ nano deployment.yaml
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ kubectl apply -f deployment.yaml
        - containerPort: 80

                                                kubectl apply -f deployment.yaml
Warning: autopilot-default-resources-mutator:Autopilot updated Deployment default/web-app: defaulted unspecified 'cpu' resource for containers [web-app
] (see http://g.co/gke/autopilot-defaults).
deployment.apps/web-app created
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$
```

- How did you verify that your application was successfully deployed and accessible?

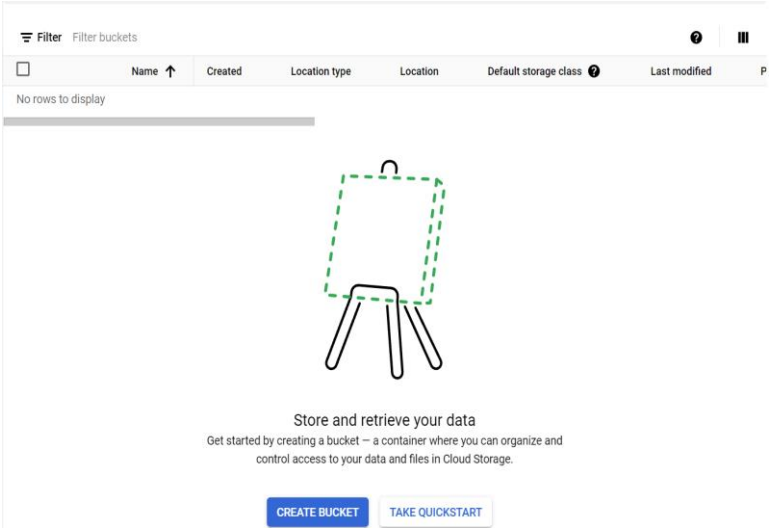
kubectl get deployments

```
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$ kubectl get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
web-app   0/3     3            0           55s
onayeva_alua@cloudshell:~ (psyched-bonfire-436413-e2)$
```

Exercise 5: Storing and Accessing Data in Google Cloud Storage

1. **Objective**: Learn how to store, manage, and access data using Google Cloud Storage.
2. **Steps**:
    - Create a new Cloud Storage bucket in the Google Cloud Console.
    - Upload various types of files (e.g., text, images, videos) to the bucket.
    - Set access permissions for the bucket and test public and private access to the files.
    - Use the Cloud Console to download, move, and delete files in the bucket.
3. **Questions**:
    - *How do you create a Cloud Storage bucket, and what options are available during setup?*

Press **‘Create bucket’** button



So, we can choose the name, region, and other parameters for the bucket.

- **Choose where to store your data**

  Location: us (multiple regions in United States)
  Location type: Multi-region

- **Choose a storage class for your data**

  Default storage class: Standard

- **Choose how to control access to objects**

  Public access prevention: On
  Access control: Uniform

- **Choose how to protect object data**

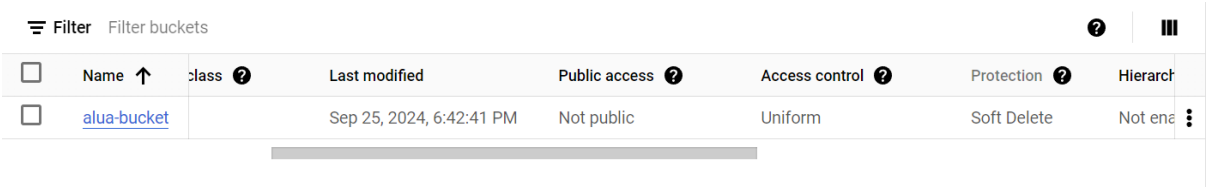
  Soft delete policy: Default
  Object versioning: Disabled
  Bucket retention policy: Disabled
  Object retention: Disabled
  Encryption type: Google-managed

- *What are the differences between setting a bucket to public versus private?*

In public anyone with the link can access files. In private only users with permissions can access files.

- *How can you manage access permissions for individual files in a bucket?*

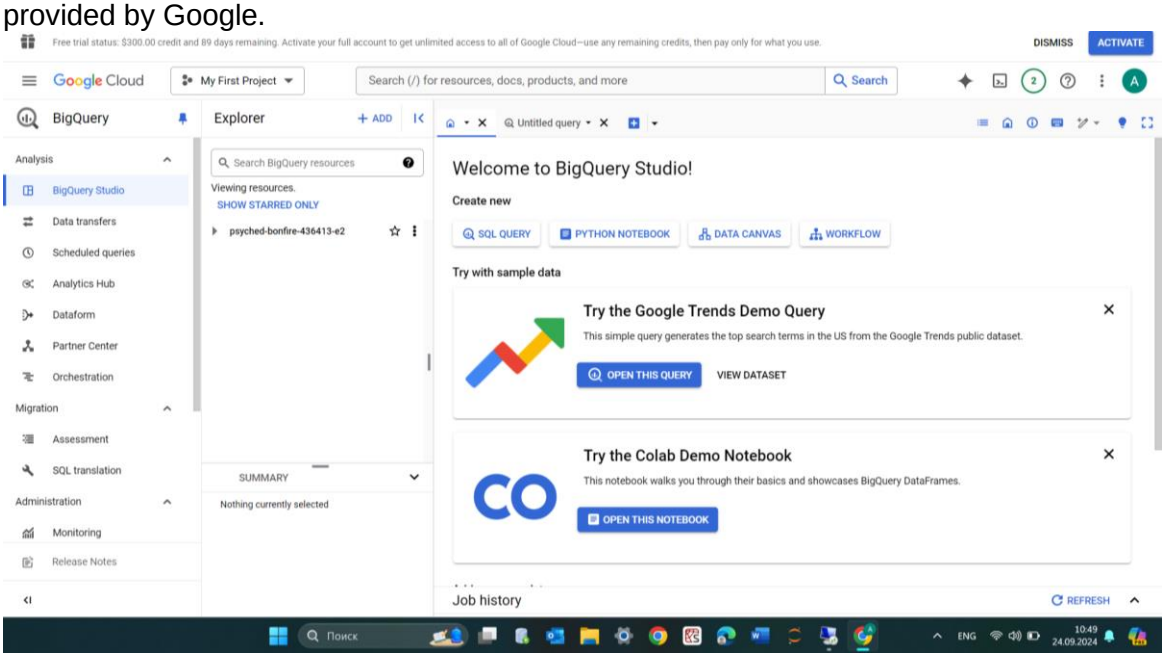
So here is my bucket, and in access control I got ‘Uniform’. If I edit this part I can set ACL (Access Control List), and grant specific users with permissions.



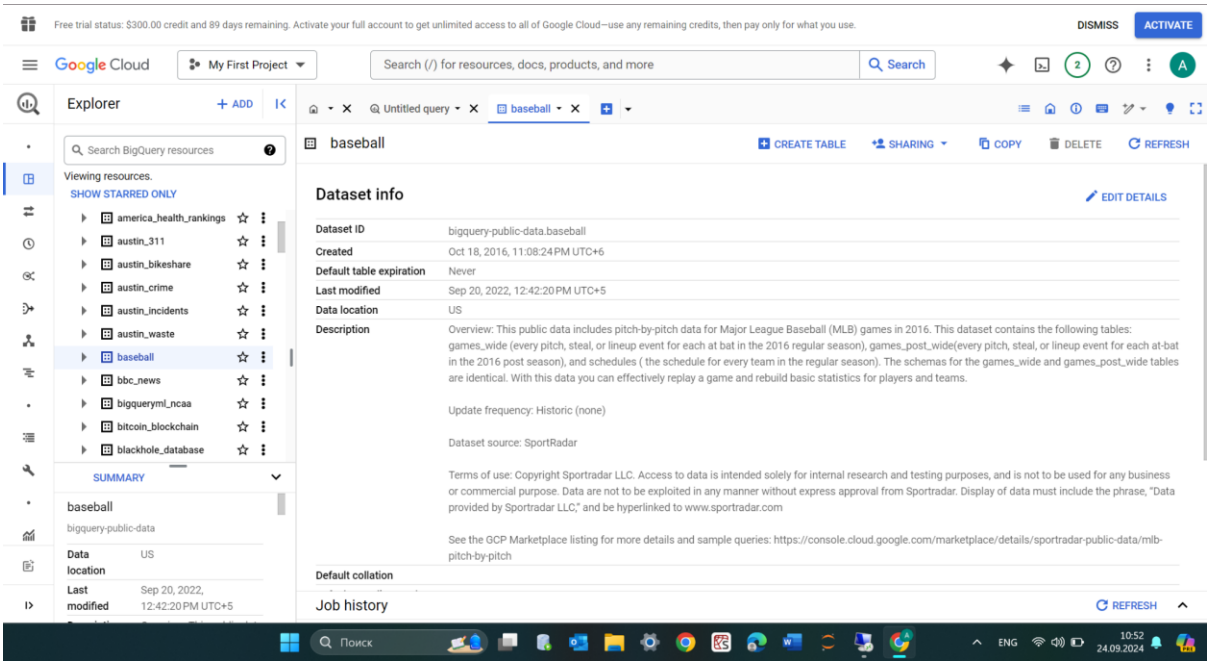
Exercise 6: Analyzing Data with BigQuery

1. **Objective**: Perform data analysis tasks using BigQuery.
2. **Steps**:
   - Access BigQuery in the Google Cloud Console.
   - Create a dataset and table by importing a sample dataset provided by Google.
   - Write and execute SQL queries to perform basic data analysis, such as filtering, aggregation, and sorting.
   - Visualize the results using Google Data Studio or another visualization tool.
3. **Questions**:
   - *What steps did you take to create a dataset and table in BigQuery?*

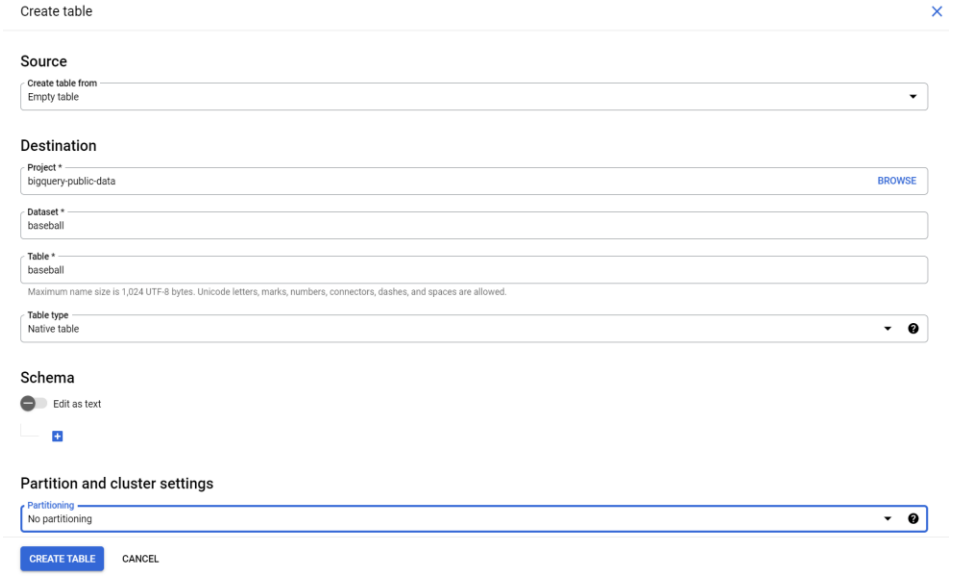
1. Open BigQuery and choose *‘View Dataset’* option, in order to work with datasets provided by Google.



2. There is a list of datasets

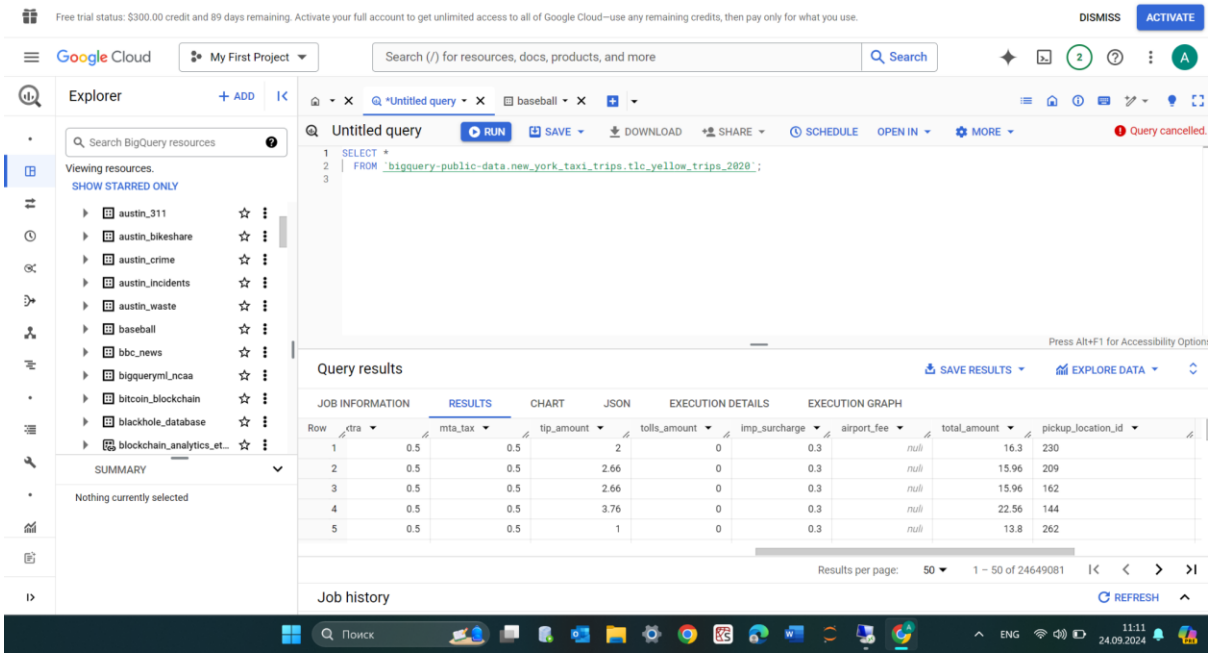


3. To create table we can select *'Create Dataset'* button



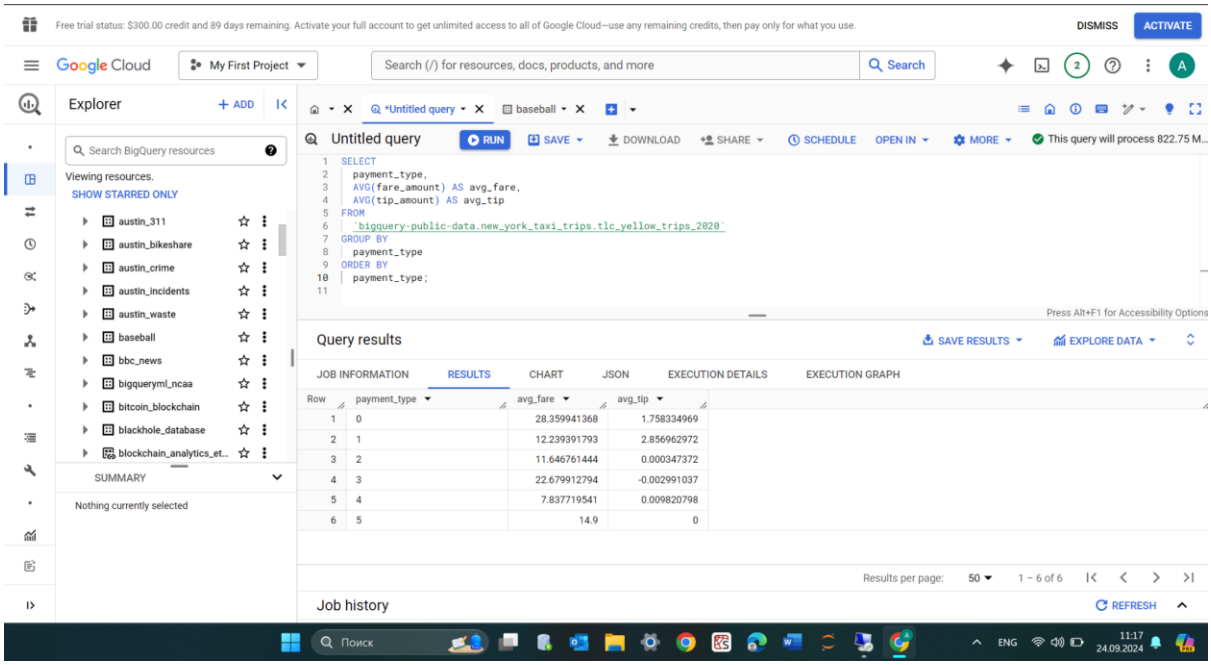
- *How did you write and execute SQL queries in BigQuery?*

Open a query, and write a select, I use a public dataset already provided by Google. Dataset about NYC taxi rides.

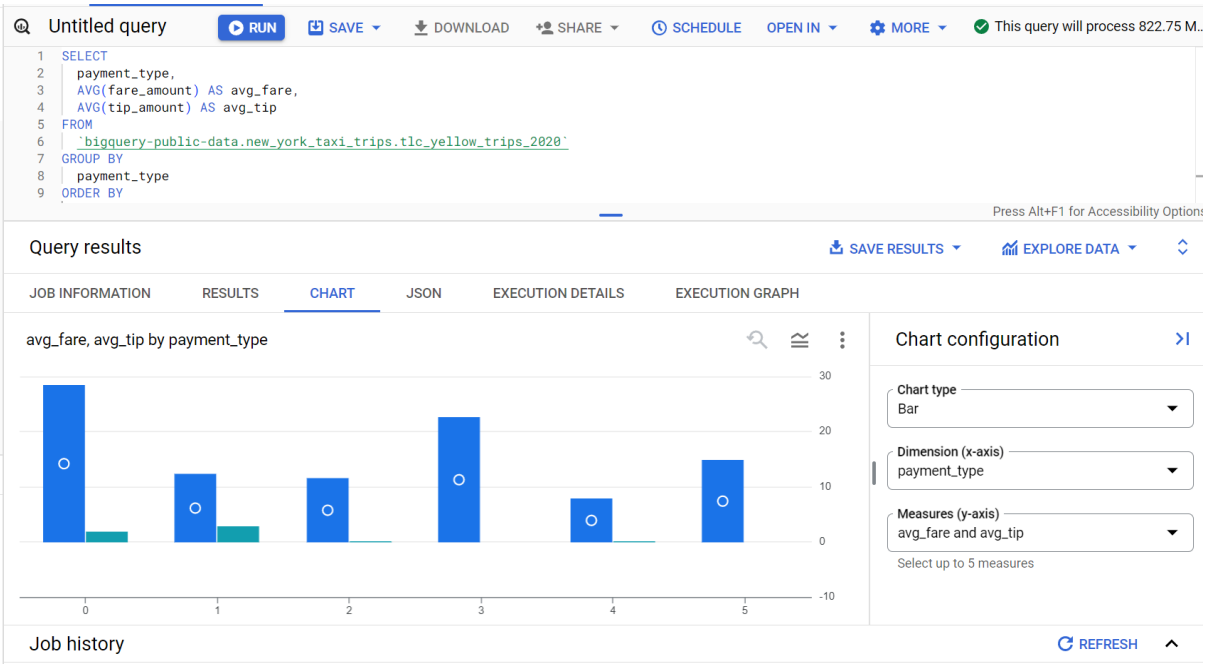


- *What insights were you able to derive from the data analysis?*

Average fare and tip amounts for different payment methods.



Also, we can visualize it.



I want to see the distribution of trips amount by each day of the week.
So as it can be seen, Friday is the day with the most taxi trips.