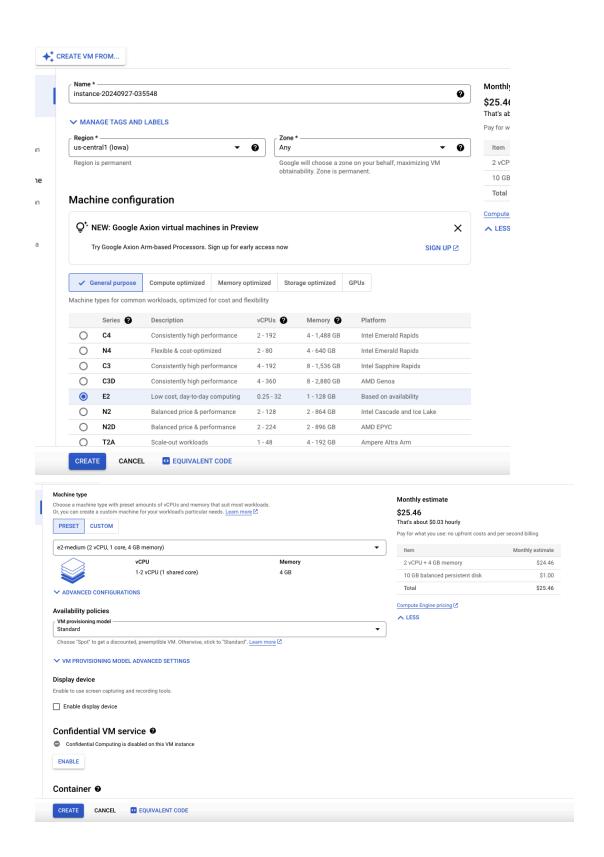
Ex 1.

Models	laaS	PaaS	SaaS
Control	Greatest control over infrastructure, storage, networks etc.	Moderate control on deployment but infrastructure is managed by service provider	Least control. Developers can only adjust configuration of the software
Flexibility	Highly flexible. Deployment environment is fully controlled	Moderately flexible	Least flexible. Software is not managed by user
Use	Used by enterprises that need virtual machine-like infrastructure without need to manage the hardware.	Microservice architecture, web app development.	Ready to use software solutions. Ex: any Google generic app like Google Photos etc.
GCP services	Google Compute Engine	Google App Engine, Google Cloud Functions	Google Drive
Real word examples	Enterprise that needs to run multiple VMs for their services.	Team of developers that want to deploy multi-platform application	Tiny businesses that need gmail and other services for rapid and smooth delivery of their services.

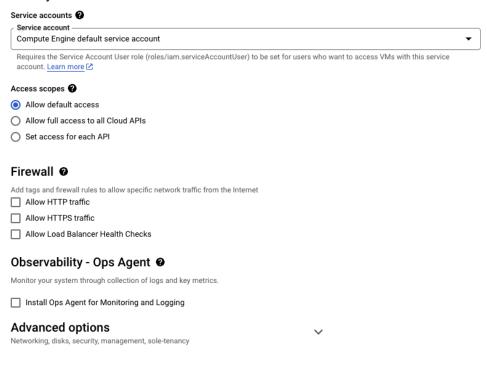
Ex 2.

- a) Primary use case of Compute Engine For enterprises that need to create and run multiple virtual machines for scalability and performance of their applications.
- b) It provides power of Kubernetes without the hassle of managing underlying infrastructure. GKE automatically manages clusters and offers both vertical and horizontal scaling. You don't need to manage nodes and control panes. It has CI/CD that allows for updating the code base while making the current version available. GKE also makes Kubernetes up to date. Aside from that it balances workload and runs in multiple cloud environments.
- c) Cloud storage can be cost effective. Businesses can scale their data on demand. Also cloud storage offers high reliability and security of data.
- d) BigQuery can lead to cost savings when it comes to processing large datasets. It is highly performant. It offers real time analytics and is compatible with vast data sources.

Ex 3. a) For creating choose your machine type to adjust cost, region, OS, traffic settings etc.



Identity and API access @



- b) Connected via SSH button on VM instance. Commands to install web server (apache):
 - sudo apt-get update
 - sudo apt-get install apache2 php7.0
 - echo '<!doctype html><html><body><h1>Hello World!</h1></body></html>' | sudo tee
 /var/www/html/index.html

View the main web page via external IP.

For instance: 34.72.100...

c) When stopped data is persisted, and charged only for storage. When deleted data is also deleted.

Ex 4.

- 1. npm init npm install express
- 2. docker build -t ass1-kubernetes . docker run -p 8080:8080 ass1-kubernetes
- 3. gcloud init gcloud auth configure-docker gcr.io
- 4. docker images

docker tag ass1-kubernetes:latest gcr.io/integral-server-383514/ass1-kubernetes:latest

- 5. docker push gcr.io/integral-server-383514/ass1-kubernetes:latest
- 6. gcloud container clusters get-credentials autopilot-cluster-1 --zone us-central1
 - Message, otherwise install auth plugin:

Fetching cluster endpoint and auth data. kubeconfig entry generated for autopilot-cluster-1.

(Optional). gcloud components install gke-gcloud-auth-plugin

- Update Your kubectl Configuration: Once the plugin is installed, you can update your kubectl config to use it.

export USE_GKE_GCLOUD_AUTH_PLUGIN=True

- 7. kubectl create deployment ass1-kubernetes
- --image=gcr.io/integral-server-383514/ass1-kubernetes:latest kubectl expose deployment ass1-kubernetes --type=LoadBalancer --port 80 --target-port 8080

kubectl get services ass1-kubernetes

kubectl get deployments

a) You build via "docker build -t"

To push to GCR you need to have gcloud cli installed

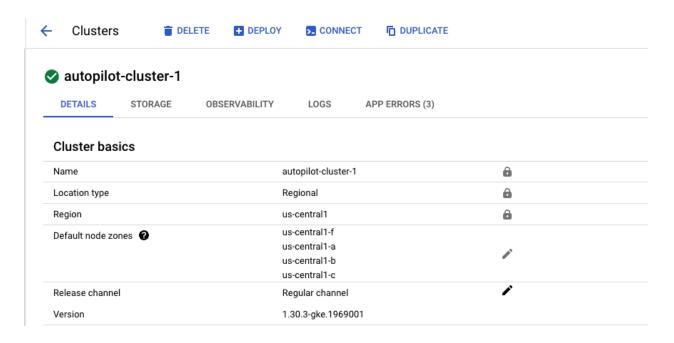
Once installed you log in, you retag your docker image to be compatible with google instructions.

"docker tag [image]:[tag] gcr.io/[project-id]/[image]:[tag]"

Then push via "docker push gcr.io/[project-id]/[image]:[tag]"

b) Installation is straightforward like in VM creation.

You go to Kubernetes Engine, go to tab Clusters and create new one.



c)

You expose with "kubectl expose deployment [deployment-name] --type=LoadBalancer --port 80 --target-port 8080"

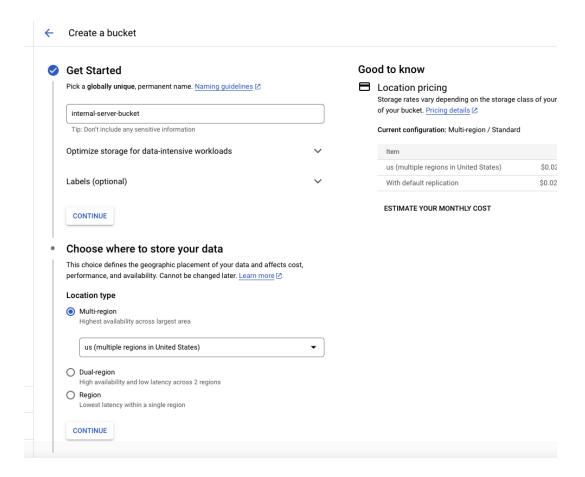
Then via

"kubectl get services"

You should see your deployment with an external ip. When you proceed to ip you should see your web app.

Ex 5

a) Simply go to Cloud storage and create bucket by giving name and setting right configuration (location, access control, storage class etc.)



Choose how to control access to objects

Prevent public access

Restrict data from being publicly accessible via the internet. Will prevent this bucket from being used for web hosting. Learn more ☑

Enforce public access prevention on this bucket

Access control

Uniform

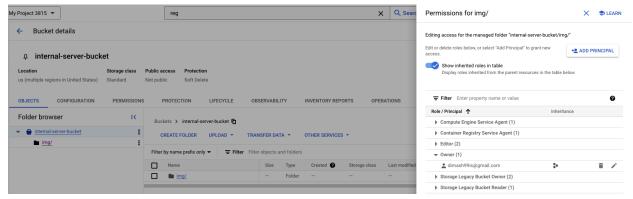
Ensure uniform access to all objects in the bucket by using only bucket-level permissions (IAM). This option becomes permanent after 90 days. Learn more 🗹

Fine-grained

Specify access to individual objects by using object-level permissions (ACLs) in addition to your bucket-level permissions (IAM). Learn more ☑

CONTINUE

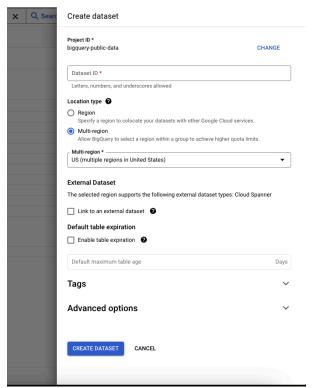
b) In public access anyone (auth and non-auth users) can access the bucket
 In private only those with specific roles and permissions can access
 Can be set via IAM roles if you are using uniform level bucket control

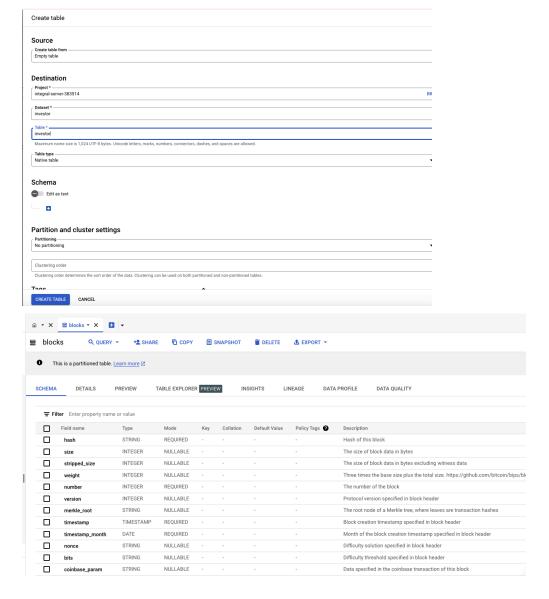


3. If you are using fine-grained control over your bucket you can use Access Control Lists (ACLs) to grant specific file permissions.

Ex6.

a) Tap create dataset, choose dataset Id and any other settings, then press create dataset. After that, create a table, choose the source. For our purpose, we will be using Google's public dataset. Later review the schema and create a table.





b) Im using "crypto blockchain" public dataset. My queries:

```
SELECT timestamp_month FROM `integral-server-383514.investor.blocks` WHERE
timestamp_month = "2011-06-01" LIMIT 1000
SELECT COUNT(*), AVG(size) FROM `integral-server-383514.investor.blocks` LIMIT 1000
SELECT * FROM `integral-server-383514.investor.blocks` ORDER BY version LIMIT 1000
```

c) By visualizing data one can analyze the trends over time, outliers of the data, aggregated information about the data, etc.