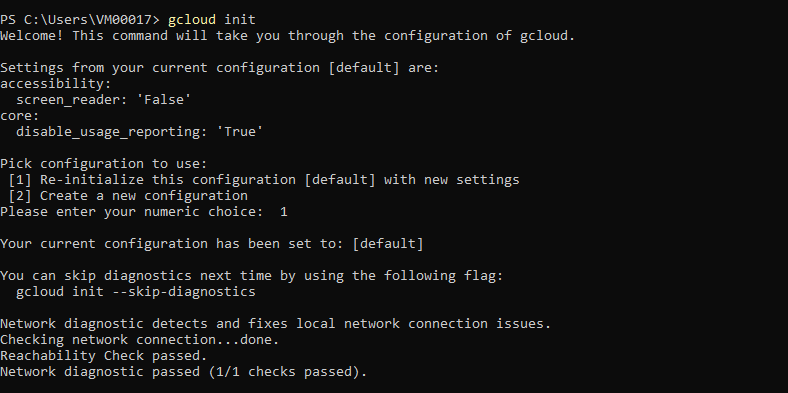
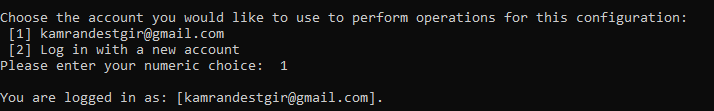
# Setup:

* Initialized gcloud cli using command

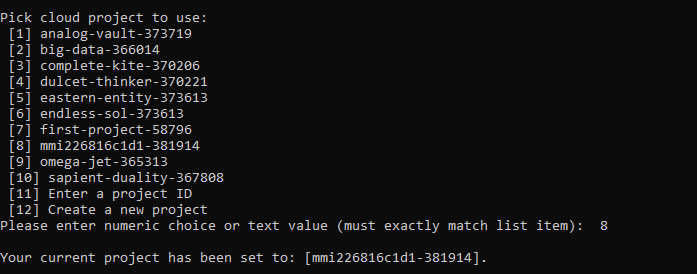
**gcloud init**



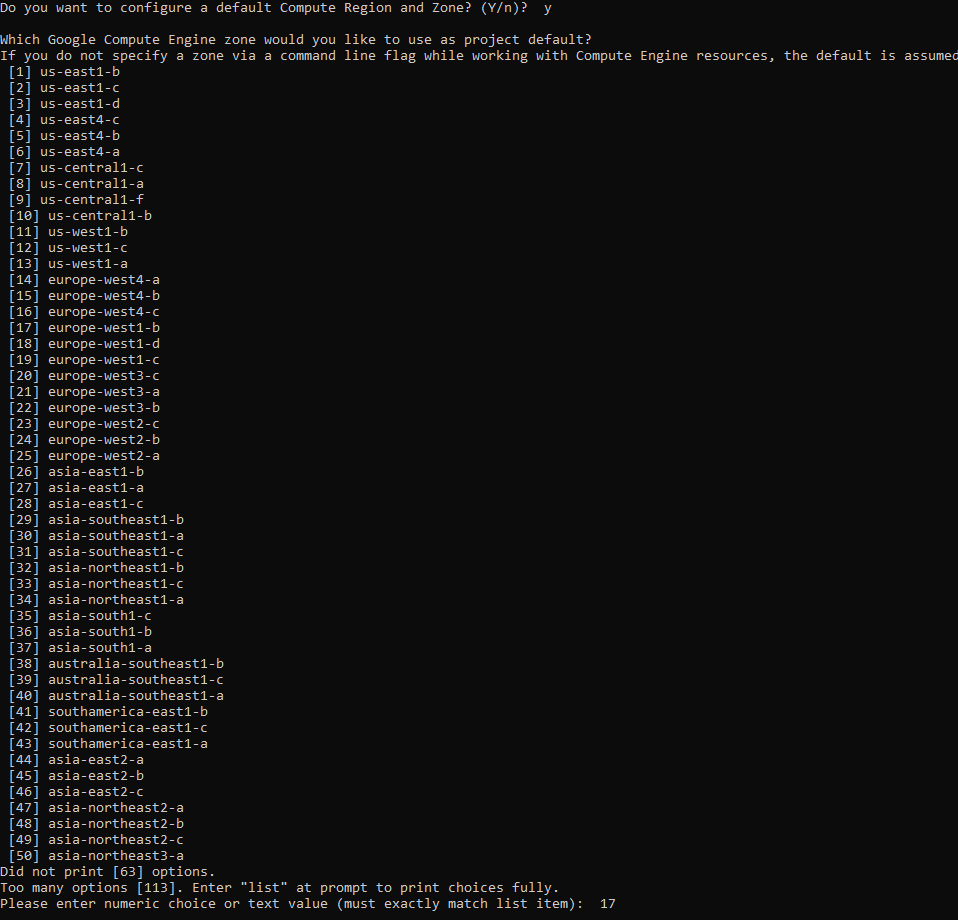
* Choose my account [kamrandestgir@gmail.com](mailto:kamrandestgir@gmail.com) as my default account



* Selected mmi226816c1d1 as project to use



* Selected **europe-west1-b** as default Compute zone



# Task 1

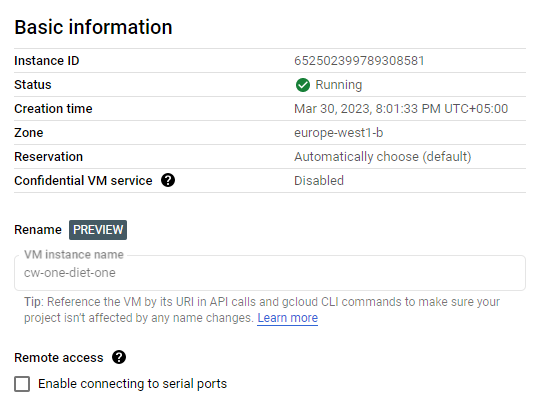
## A: Created a low cost Compute Engine VM Instance suitable for web serving

* To create VM instance executed following command.

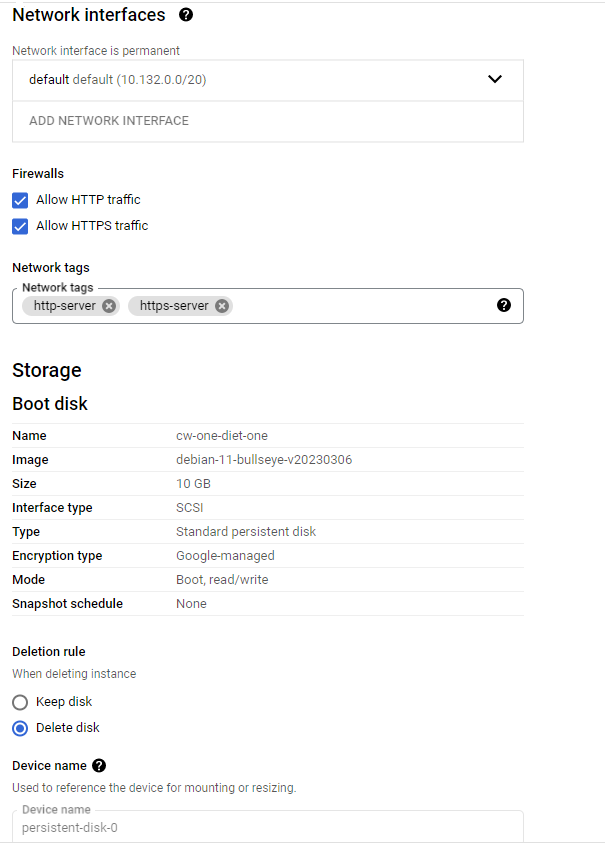
****

* This command creates a VM Instance with following configurations
  + VM instance named as **cw-one-diet-one**
  + Selected **europe-west1-b** as my selected zone
  + Selected **e2-micro** as it is the most economical machine type.
  + Selected Ubuntu 18.04 as boot image
  + Added support for both **HTTP** and **HTTPS**
* Following step have been carried out to complete this assignment
  + First of all, create project
  + Go to Virtual machine and Enable it
  + Create a new instance
  + Name Instance
  + Select Region
  + Select Zone (Both should be economical as instructed by instructor)
  + Select Machine Type
  + Added Tags for HTTP and HTTS

All the necessary information about the instance is given below in figures

From figure below, it can be observed that my created instance is in running condition. It has the zone of Europe-west1-b.

Similarly, HTTP and HTTPs traffic has been enabled for this instance. Moreover, boot disk size is set to 10 GB with default settings.



Overall, of instance status can be observed in the next figure.

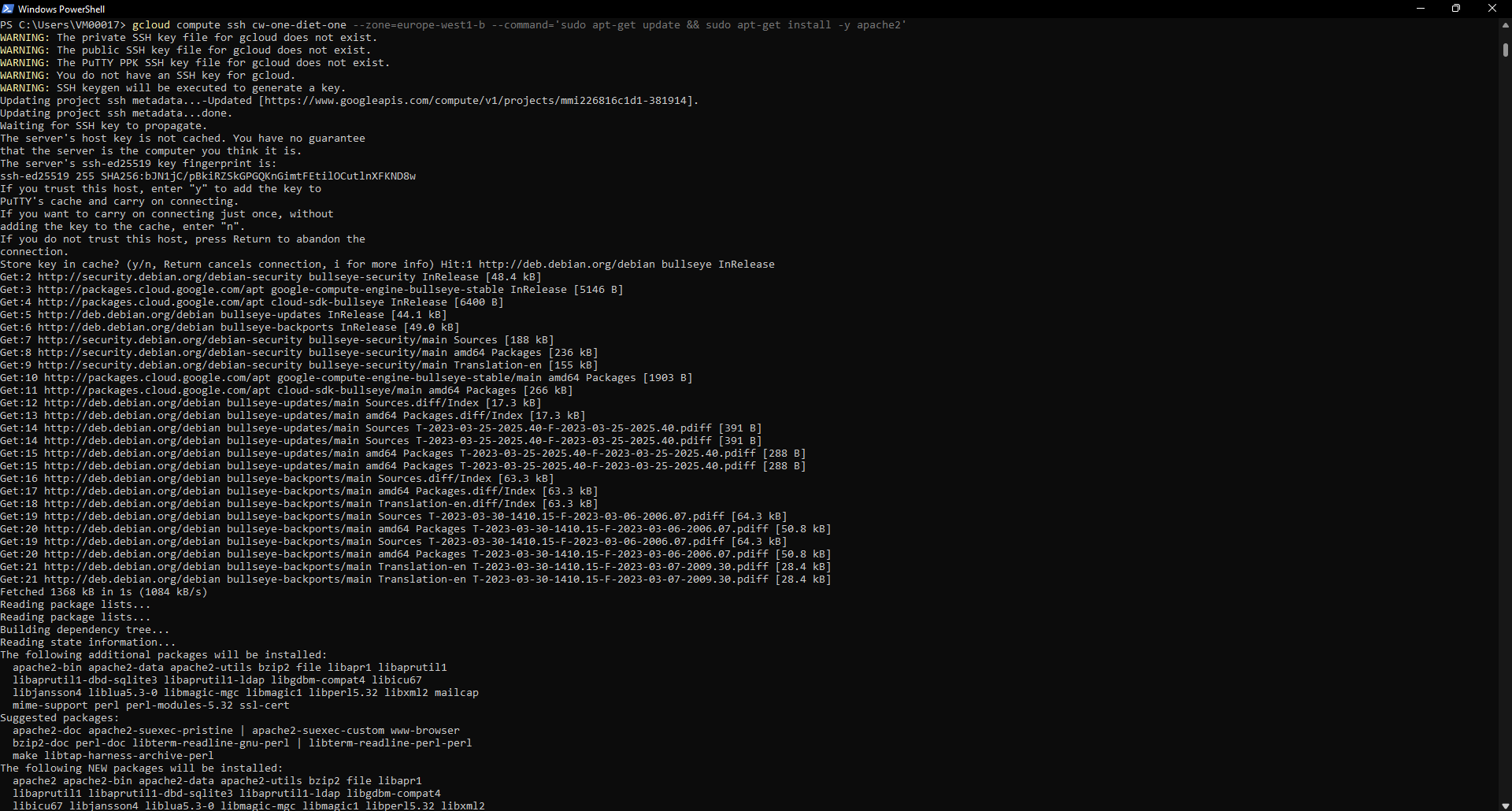


## B: Installed and tested **apache2 web server** on above created VM instance.

* Executed following command to install the apache2 web server

gcloud compute ssh cw-one-diet-one --zone=europe-west1-b --command='sudo apt-get update && sudo apt-get install -y apache2'

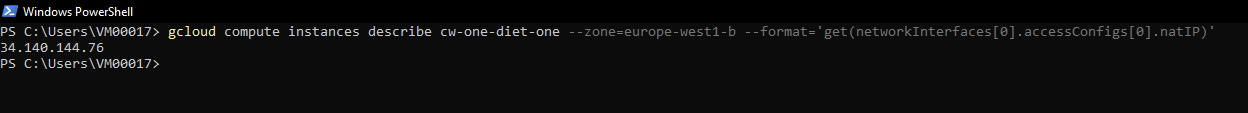
* Above command installs the apache2 server with following configurations
  + Selected **cw-one-diet-one**  VM instance that created in Step a.
  + Selected **europe-west1-b** as zone.
  + Installed **apache2** web server as it is the most basic.

****

* Extracted default IP address for newly created VM instance and to test the browser using following command.

gcloud compute instances describe coursework-diet1 --zone= us-central1-a --format='get(networkInterfaces[0].accessConfigs[0].natIP)'

gcloud compute ssh coursework-diet1 --zone=us-central1-a --command='sudo apt-get update && sudo apt-get install -y apache2'



* Tested my webserver by accessing through extracted IP address **“34.140.144.76”**

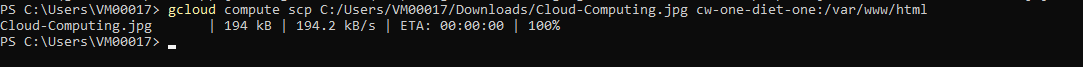


## C: Copied and served an image file to apache2 server and tested by accessing it through a URL.

* Copied photographic file named “Cloud-Computing.jpg” from local device to apache2 home directory hosted on VM instance using following command

gcloud compute scp C:/Users/VM00017/Downloads/Cloud-Computing.jpg cw-one-diet-one:/var/www/html

gcloud compute scp C:/Users/Amin/Downloads/gcu.jpg coursework-diet1:/var/www/html

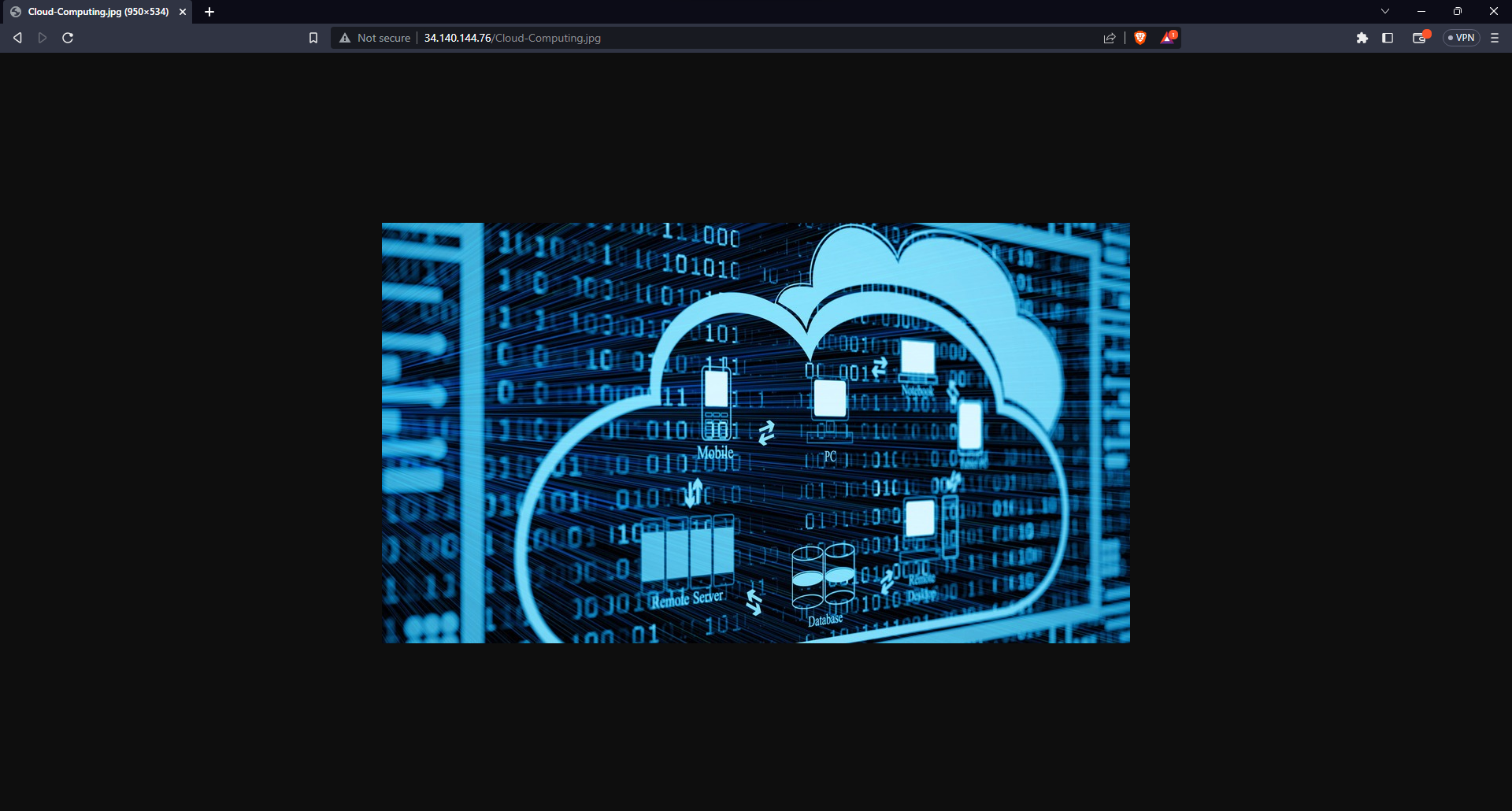


* Restarted apache2 server by following command.

gcloud compute ssh coursework-diet1 --zone= us-central1-a --command='sudo service apache2 restart'

gcloud compute scp C:/Users/Amin/Downloads/gcu.jpg coursework-diet1:/var/www/html

* Accessed newly copied photographic file using <http://34.140.144.76/Cloud-Computing.jpg>



## Part D: Developed and tested a simple App Engine app with JavaScript as my preferred language and tested it both locally and remotely.

* Created a new NodeJS App Engine app on my local device with following code:

Code:

const http = require('http');

const hostname = '0.0.0.0';

const port = process.env.PORT || 8000;

const server = http.createServer((req, res) => {

res.statusCode = 400;

res.setHeader('Content-Type', 'text/plain');

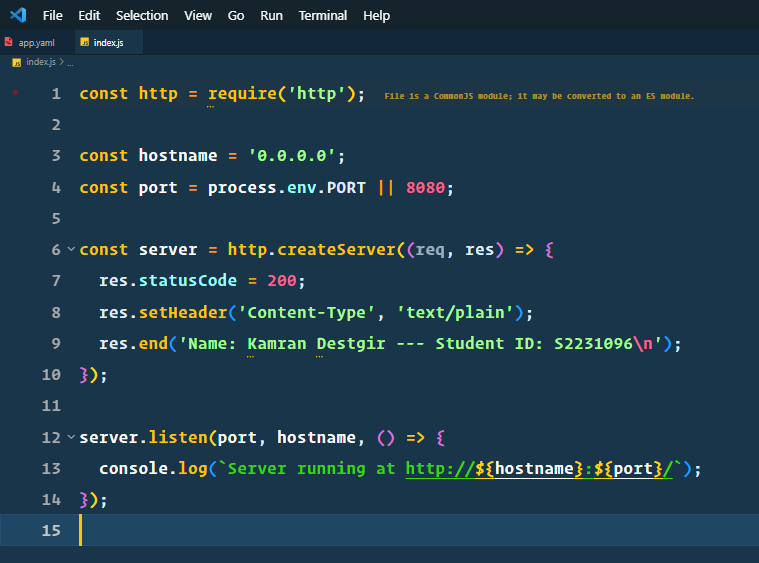
res.end('My Credentials: My Name: Muhammad Amin --- My Student ID: S2266210\n');

});

server.listen(port, hostname, () => {

console.log(`Server running at http://${hostname}:${port}/`);

});

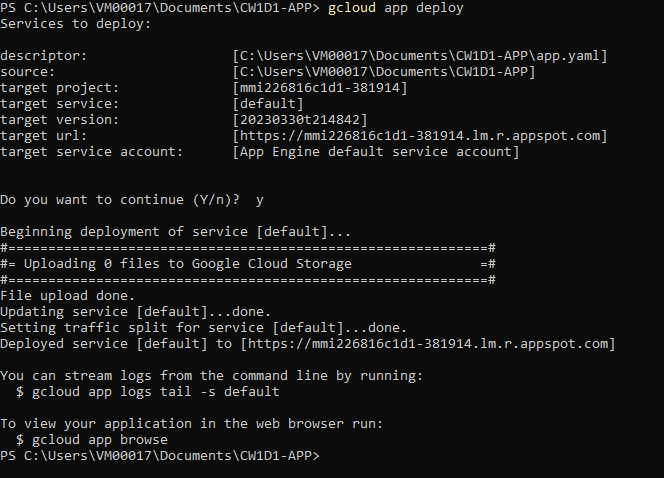


* Initialized the app using following command

gcloud app create --region=us-west1

* Assigned Storage Object Viewer permission to user as it requires to deploy the app.
* Executed following command to deploy the app

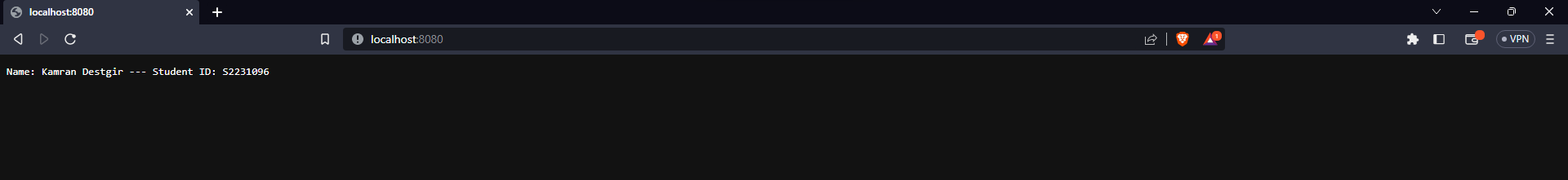
gcloud app deploy



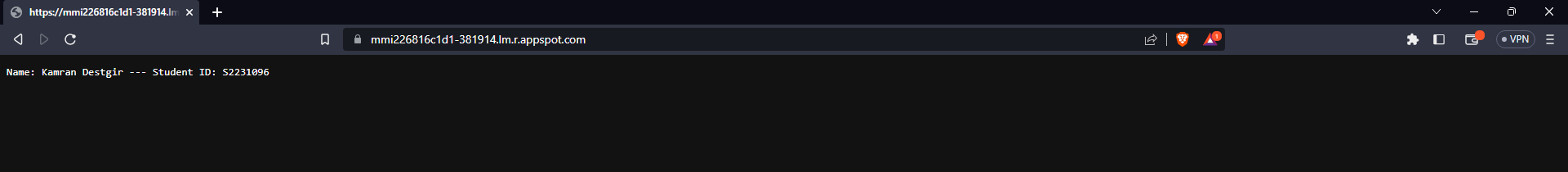
* To test the app locally executed following command

npm start

* And accessed the app on localhost



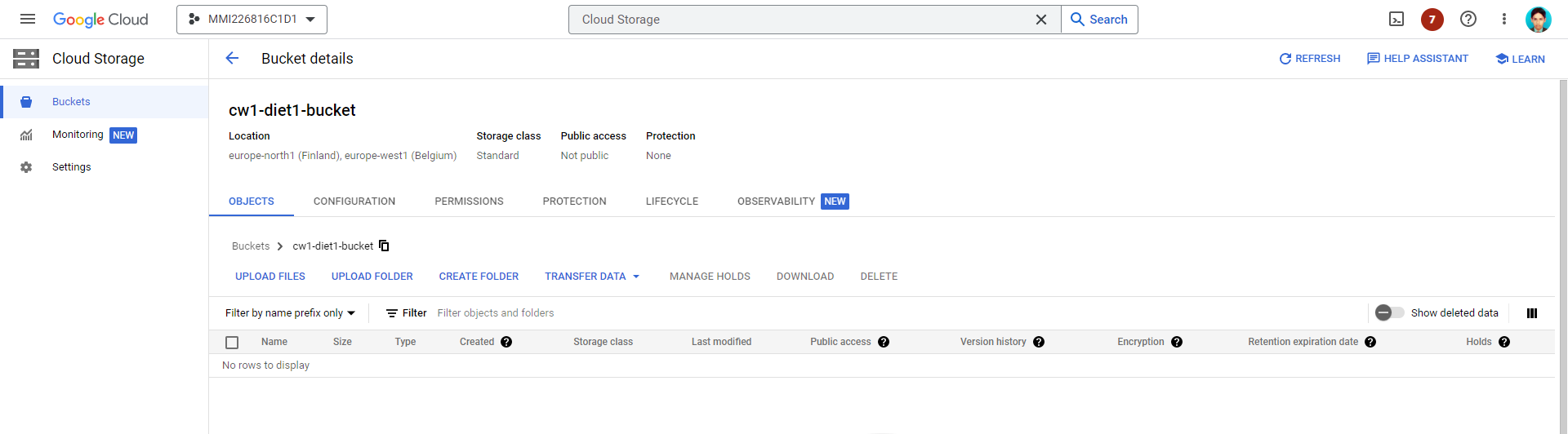
* To Test the app remotely accessed the app using target url: <https://mmi226816c1d1-381914.lm.r.appspot.com/> provided in the logs



# Task 2:

## A: Created a cloud storage bucket that is configured to replicate its content to two regions with a storage class appropriate for the frequent access using Google Cloud Console with following attributes

* Named bucket **cw1-diet1-bucket**
* Selected **Dual-region** under Location type and selected **Europe** as location and selected **europe-west1** and **europe-north1** as regions.
* Selected **Standard** under storage as it is suitable for frequent access and short-term storage
* Unchecked “**Enforce public access prevention on this bucket**” option and selected **Uniform** Access control

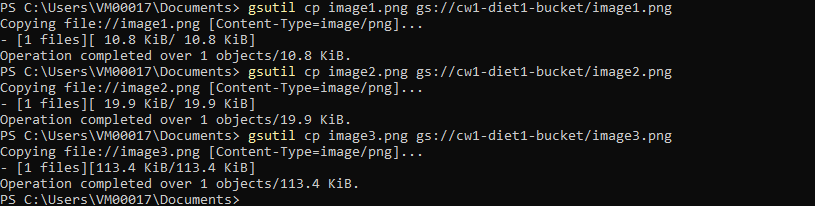


* To upload three images to cw1-diet1-bucket executed following commands

gsutil cp image1.png gs://cw1-diet1-bucket/image1.png

gsutil cp image2.png gs://cw1-diet1-bucket/image2.png

gsutil cp image3.png gs://cw1-diet1-bucket/image3.png



* To make these images accessible publically executed following command

gsutil acl ch -u AllUsers:R gs://cw1-diet1-bucket/image1.png

gsutil acl ch -u AllUsers:R gs://cw1-diet1-bucket/image2.png

gsutil acl ch -u AllUsers:R gs://cw1-diet1-bucket/image3.png

## C: Created and Copied html file to display all three images with captions and hosted it on our previously created web server.

* Created images.html file

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Diet-1 - Images</title>

</head>

<body>

<h1>Diet-1 - Images</h1>

<div>

<img

src="https://pixabay.com/photos/paper-heart-symbol-romance-1100254/"

alt="Picture 1"

/>

<p>Caption</p>

</div>

<div>

<img

src="https://pixabay.com/illustrations/book-old-surreal-fantasy-pages-863418/"

alt="Picture 2"

/>

<p>Caption</p>

</div>

<div>

<img

src=" https://storage.googleapis.com/diet1-bucket/picture4.jpg "

alt="Picture 3"

/>

<p>Caption</p>

</div>

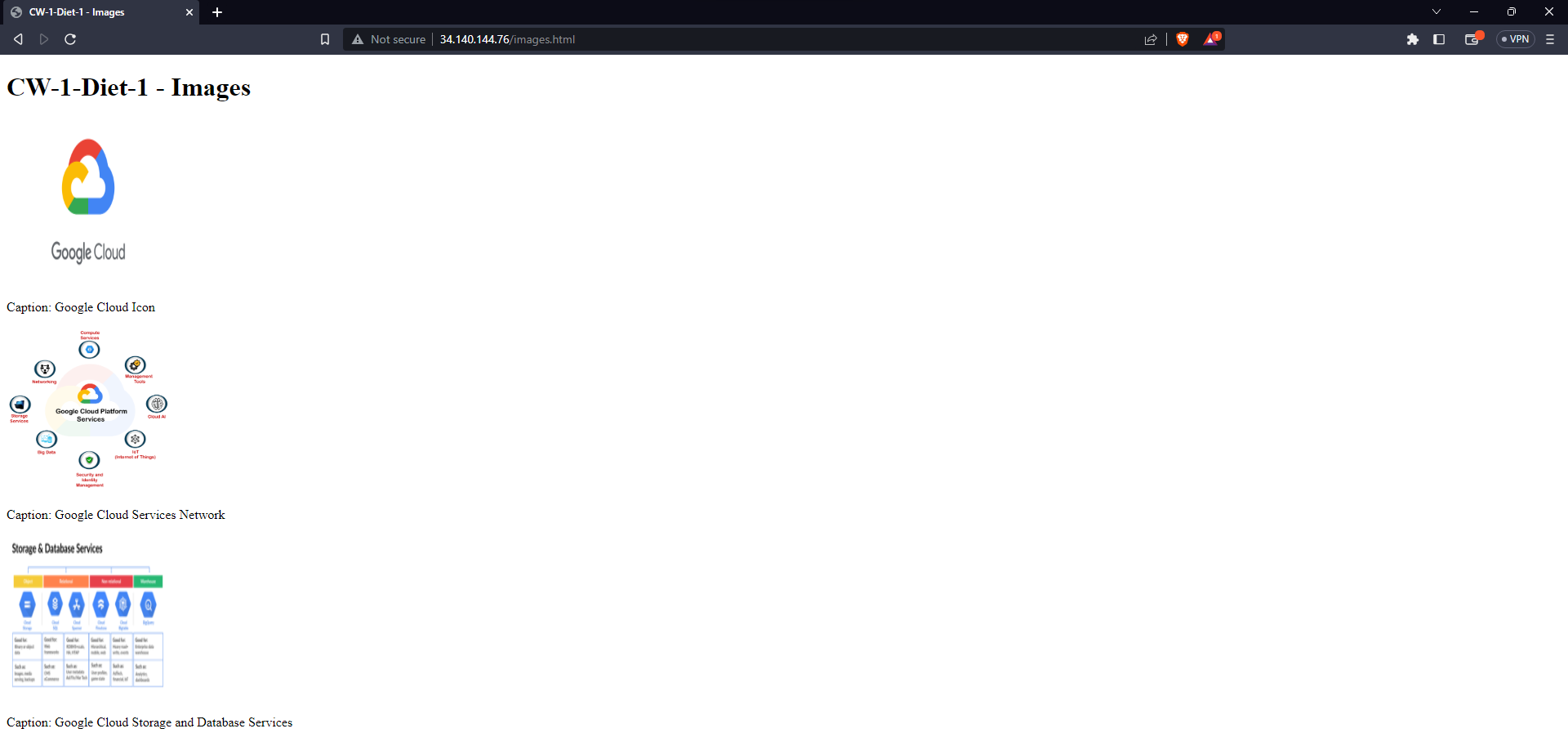
</body>

</html>

* Copied images.html file to apache2 default directory using following command

gcloud compute scp C:/Users/VM00017/Documents/images.html cw-one-diet-one:/var/www/html

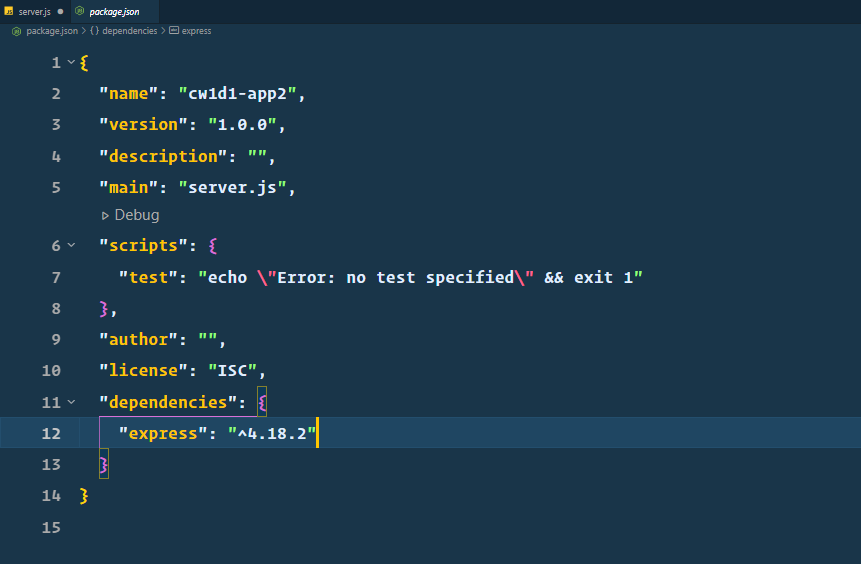
* Accessed the uploaded file



## D: Developed and tested a second App Engine app to view images stored in dual-region storage bucket created previously.

* Created a new App Engine app using NodeJS as preferred development environment.
* Initialized project following command

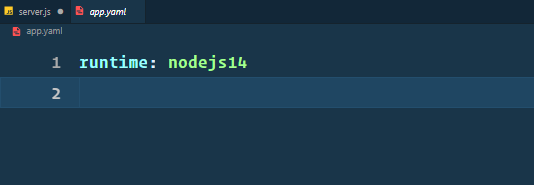
npm init



* Installed npm package **“express”** to create a server inside our newly created app using following command.

npm install express

* Create **“app.yaml”** file with following content



* Created a new file **“images.html”** inside **“views”** directory with following content

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title> Diet-1 - Images</title>

</head>

<body>

<h1> Diet-1 - Images</h1>

<div>

<img

src="https://pixabay.com/photos/paper-heart-symbol-romance-1100254/"

alt="Picture 1"

/>

<p>Caption </p>

</div>

<div>

<img

src="https://pixabay.com/illustrations/book-old-surreal-fantasy-pages-863418/"

alt="Picture 2"

/>

<p>Caption: Null</p>

</div>

<div>

<img

src="https://cdn.pixabay.com/photo/2015/12/19/20/32/paper-1100254\_960\_720.jpg"

alt="Picture 3"

/>

<p>Caption: Null</p>

</div>

</body>

</html>

* Create a new **“server.js”** file to serve our app with following content.

const express = require('express');

const app = express();

app.get('/images', (req, res) => {

res.sendFile(\_\_dirname + '/views/images.html');

});

app.get('/images/:id', (req, res) => {

const imageId = parseInt(req.params.id);

function getCaption() {

switch (imageId) {

case 1:

return 'Caption: ';

case 2:

return 'Caption: Null’;

case 3:

return 'Caption: Null’;

default:

return 'Image Not Found';

}

}

let html = `<html><head><title>Image ${imageId}</title></head><body><h1>Image ${imageId}</h1><div>

<img

src="https://storage.googleapis.com/ diet1-bucket/image${imageId}.png"

alt="Picture 1"

/>

<p>${getCaption()}</p>

</div></body></html>`;

res.send(html);

});

app.listen(process.env.PORT || 8090, () => {

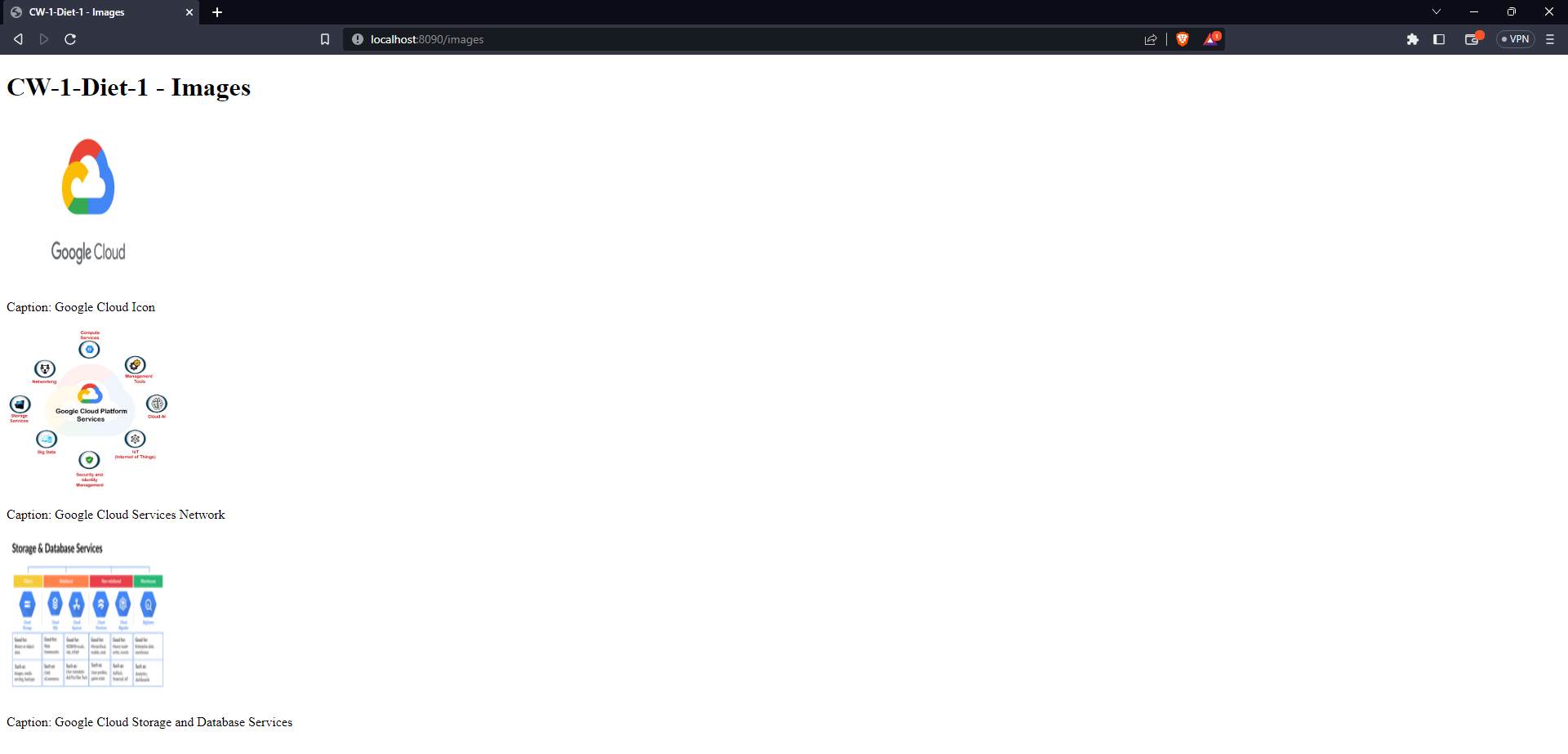
console.log('App listening on port 8090');

});

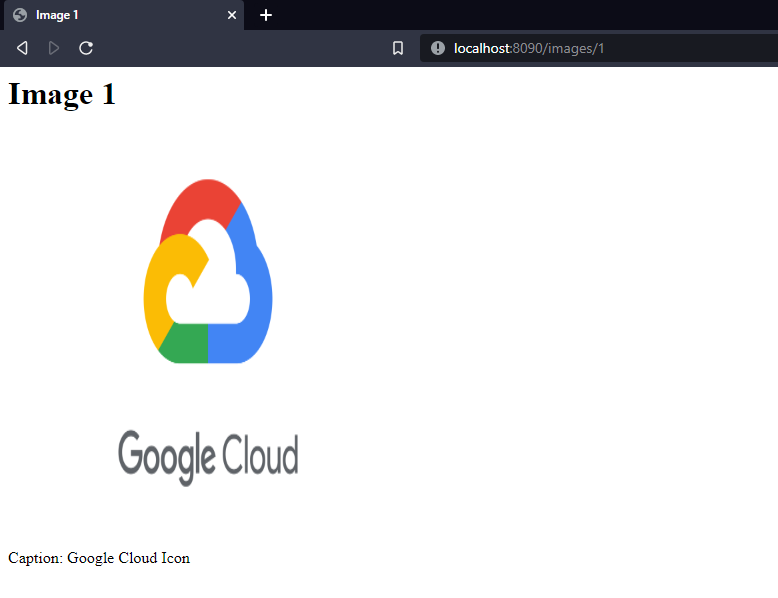


npm start

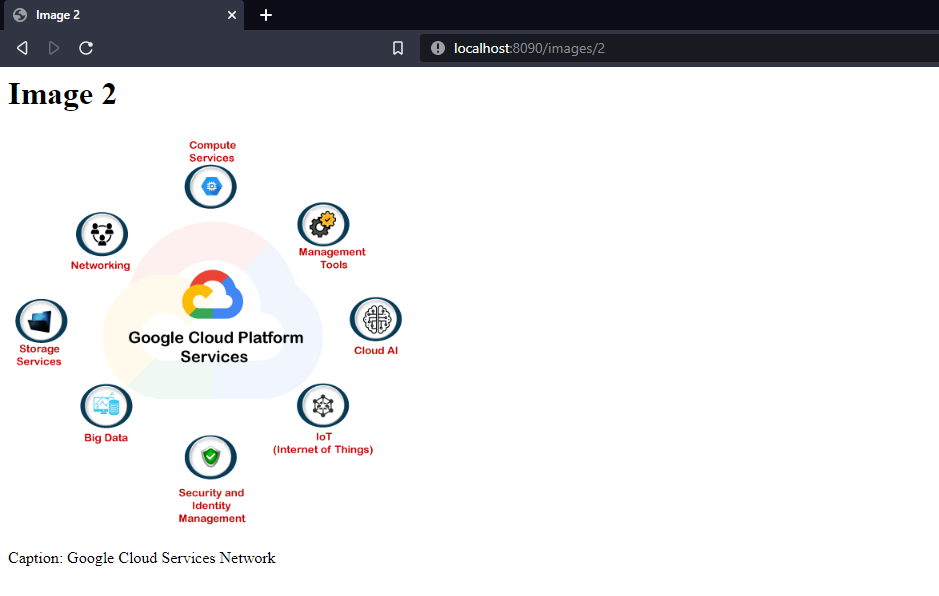
* Accessed all images by [**http://localhost:8090/images**](http://localhost:8090/images)and got following result



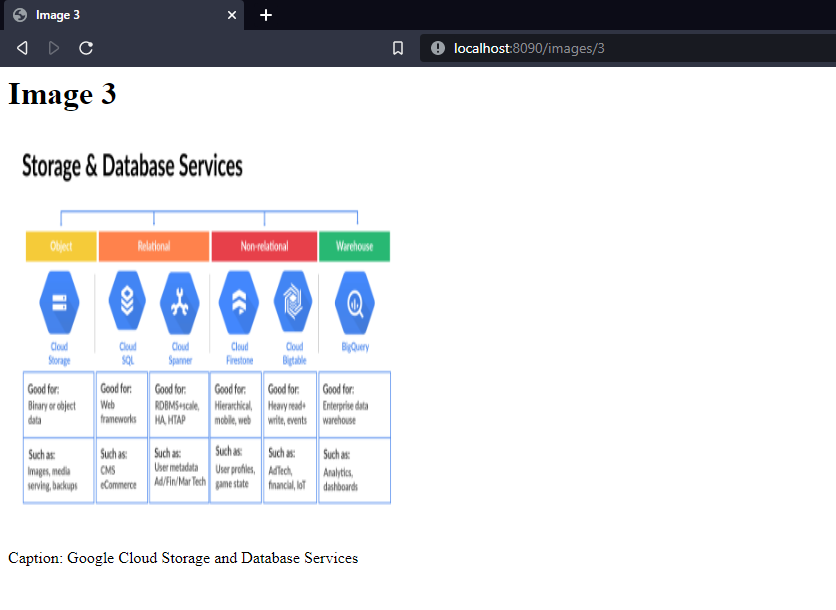
* Accessed **“First Image”** by [**http://localhost:8090/images/1**](http://localhost:8090/images/1)and got following result.



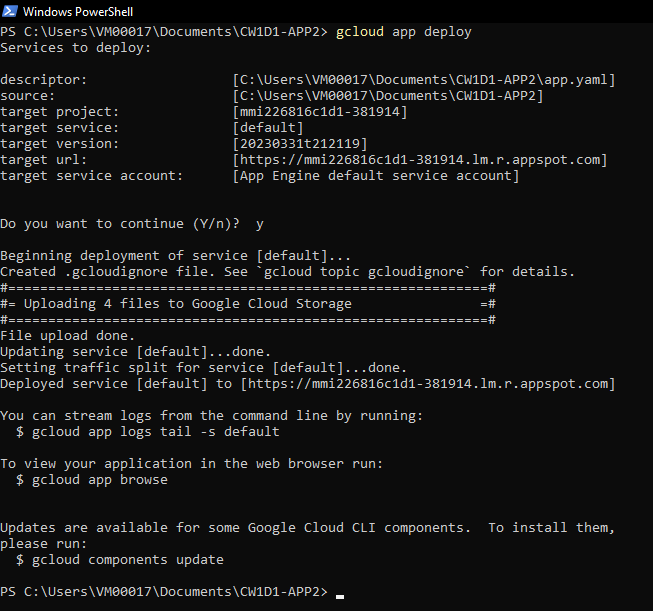
* Accessed **“Second Image”** by [**http://localhost:8090/images/2**](http://localhost:8090/images/2)and got following result.



* Accessed **“First Image”** by [**http://localhost:8090/images/3**](http://localhost:8090/images/3)and got following result.



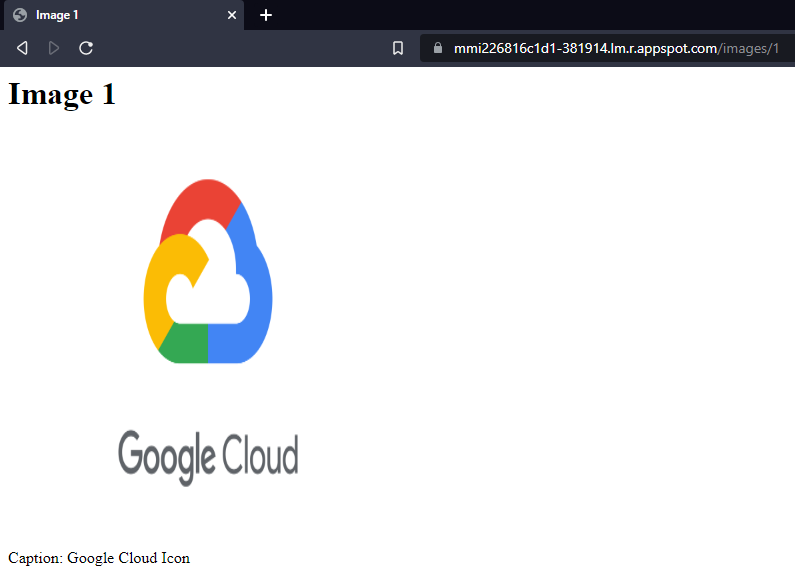
* Deployed this app to Google Cloud by **“gcloud app deploy”**



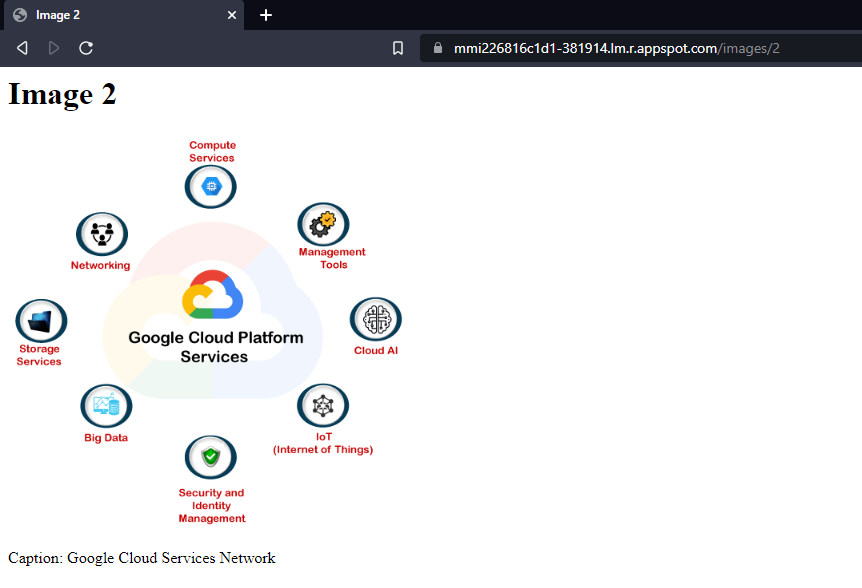
* Accessed All images by remote path [**https://mmi226816c1d1-381914.lm.r.appspot.com/images**](https://mmi226816c1d1-381914.lm.r.appspot.com/images)



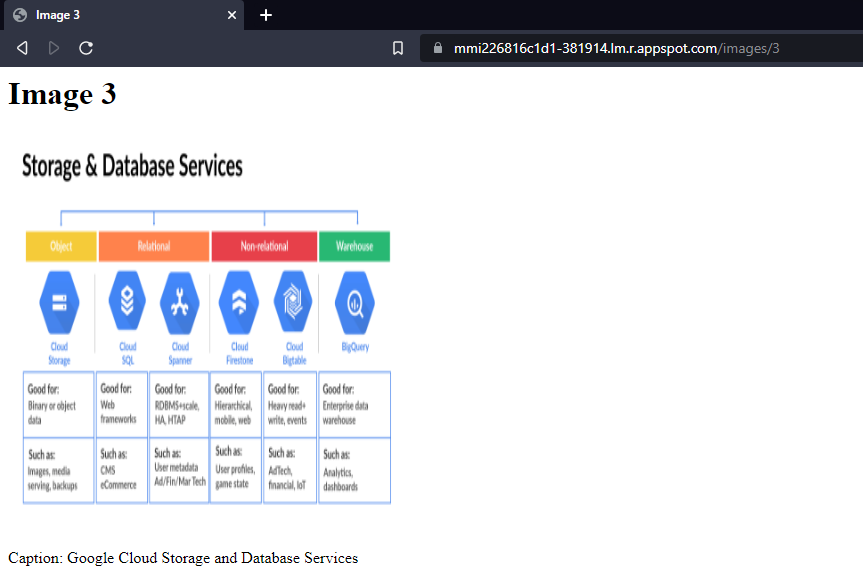
* Accessed **First Image** on remote by [**https://mmi226816c1d1-381914.lm.r.appspot.com/images/1**](https://mmi226816c1d1-381914.lm.r.appspot.com/images/1)

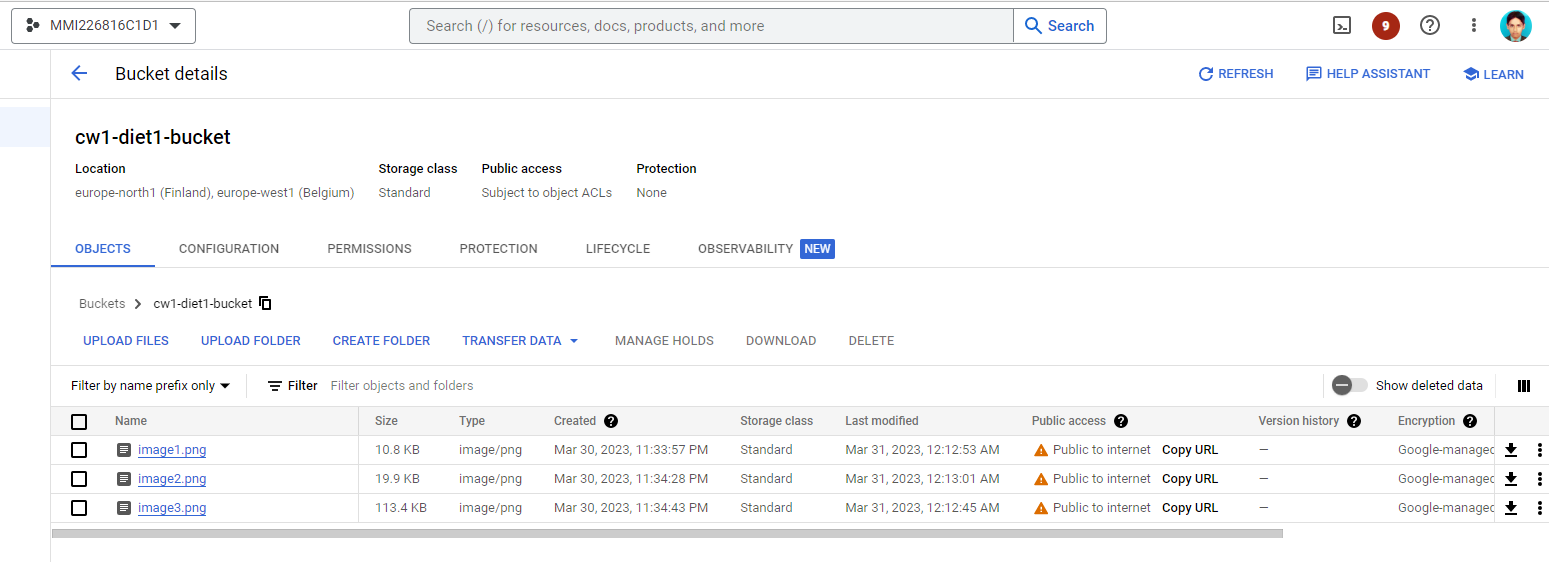


* Accessed **Second Image** on remote by [**https://mmi226816c1d1-381914.lm.r.appspot.com/images/2**](https://mmi226816c1d1-381914.lm.r.appspot.com/images/2)

****

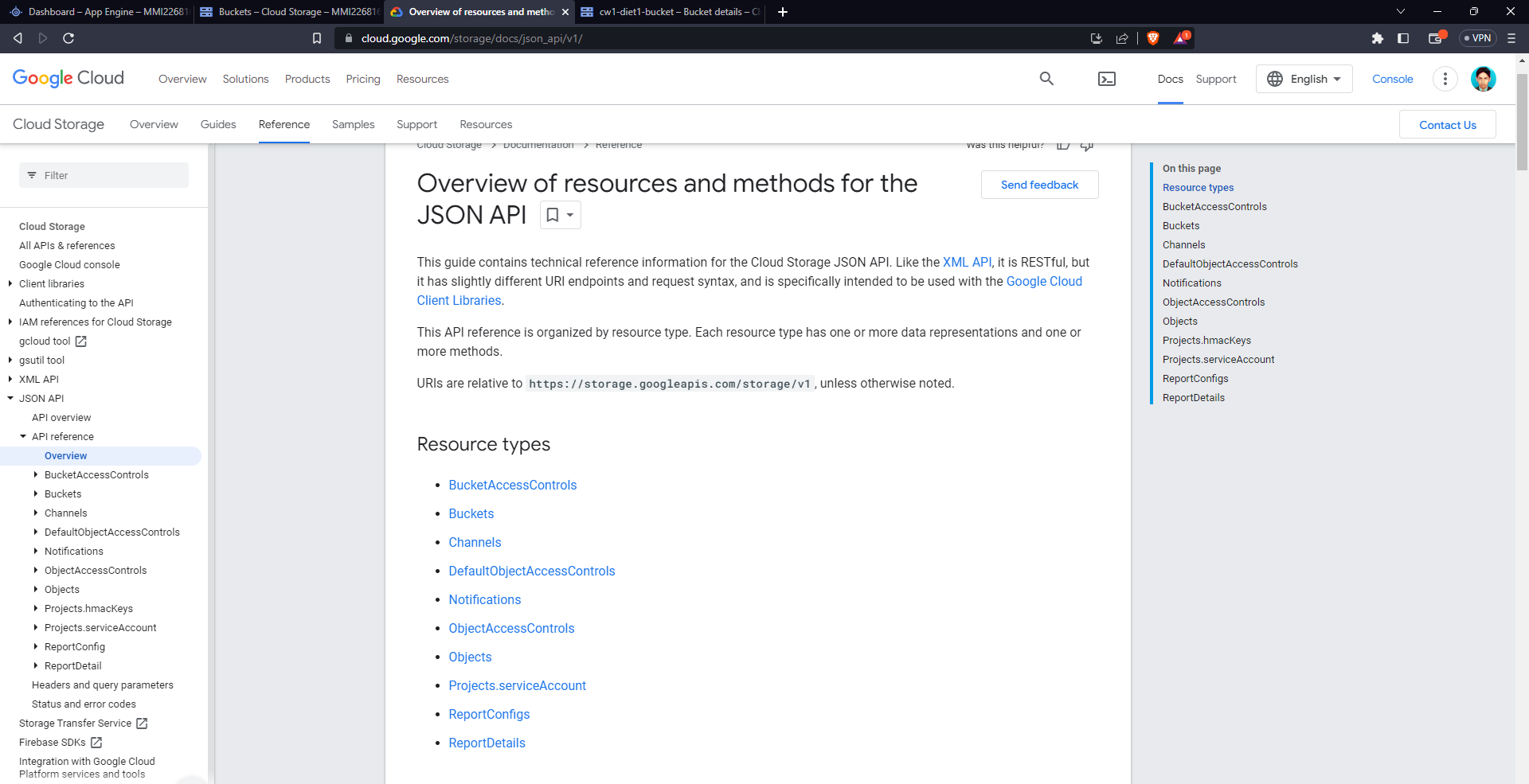
* Accessed **Third Image** on remote by [**https://mmi226816c1d1-381914.lm.r.appspot.com/images/3**](https://mmi226816c1d1-381914.lm.r.appspot.com/images/3)



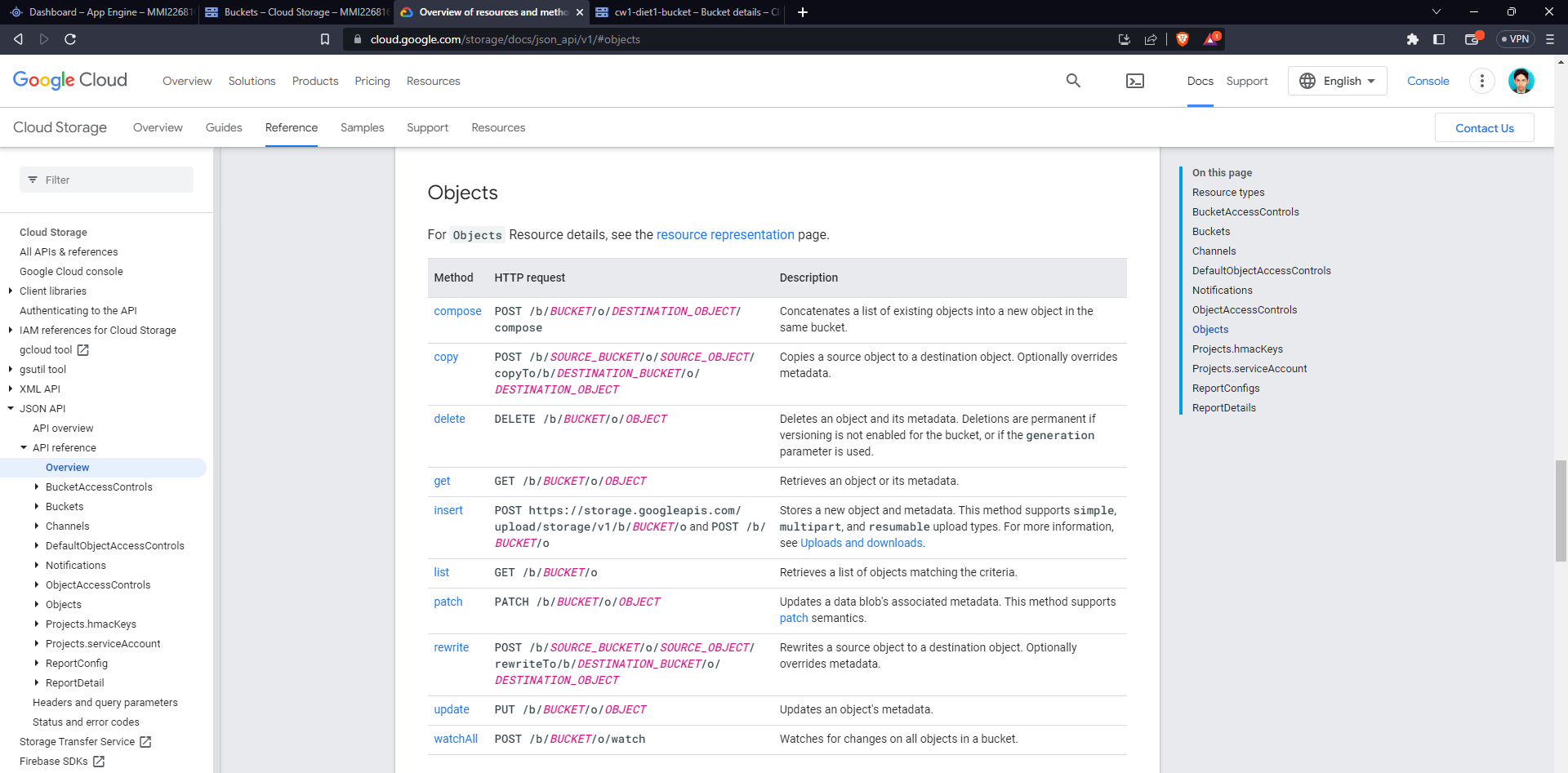


# Task 3:

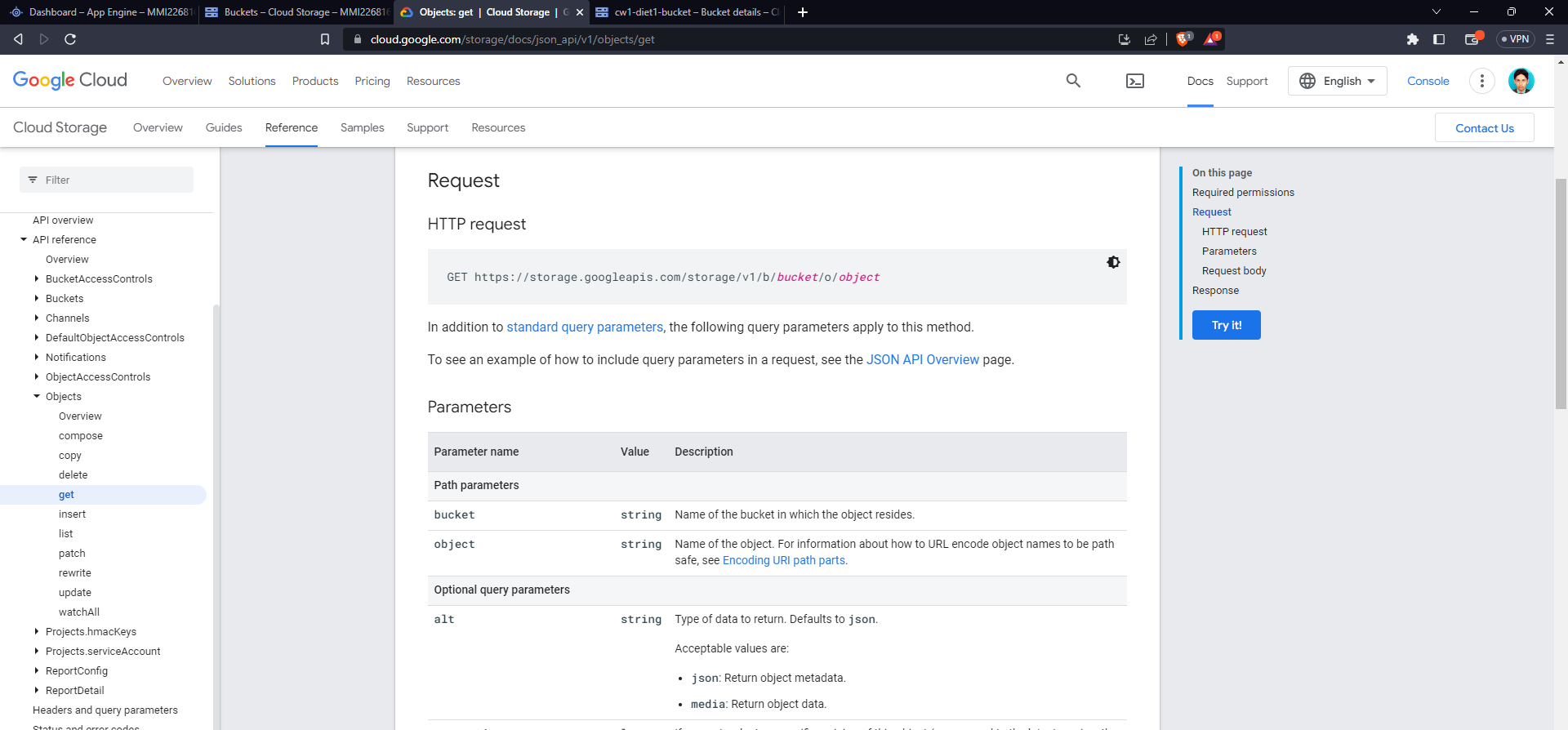
## A: Explored Google APIs Explorer and after thorough reading of multiple APIs found the **“Cloud Storage API”** most suitable for our scenario.



* **“Cloud Storage API”** is most suitable as it supports both **“Buckets”** and **“Objects”**, that is according to our scenario.
* Further explored the **“Objects”** resource and found a **“get”** method that can be used to retrieve metadata of objects.



* **“get”** method has complete endpoint [https://storage.googleapis.com/storage/v1/b/**BUCKET**/o/**OBJECT**](https://storage.googleapis.com/storage/v1/b/BUCKET/o/OBJECT).
* In given URL we can replace **“BUCKET”** with our bucket name and **“OBJECT”** with our image name.



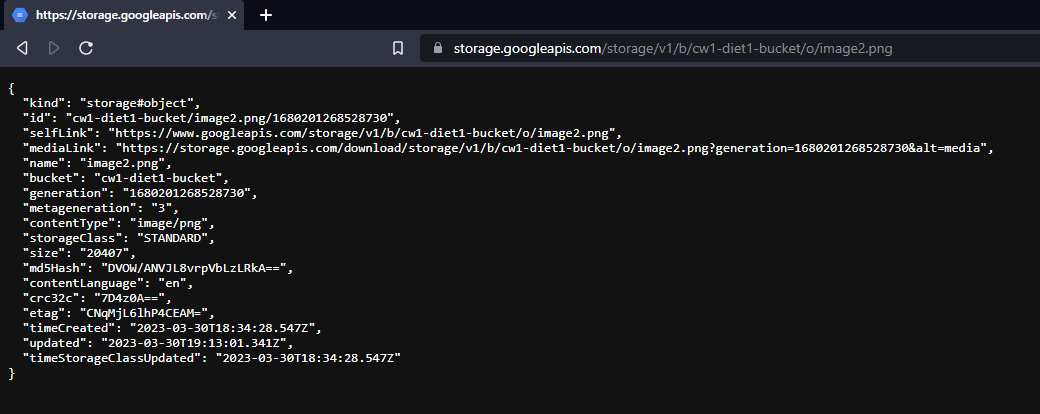
* To get the metadata for **“image1.png”** requested to following API endpoint

<https://storage.googleapis.com/storage/v1/b/cw1-diet1-bucket/o/image1.png>



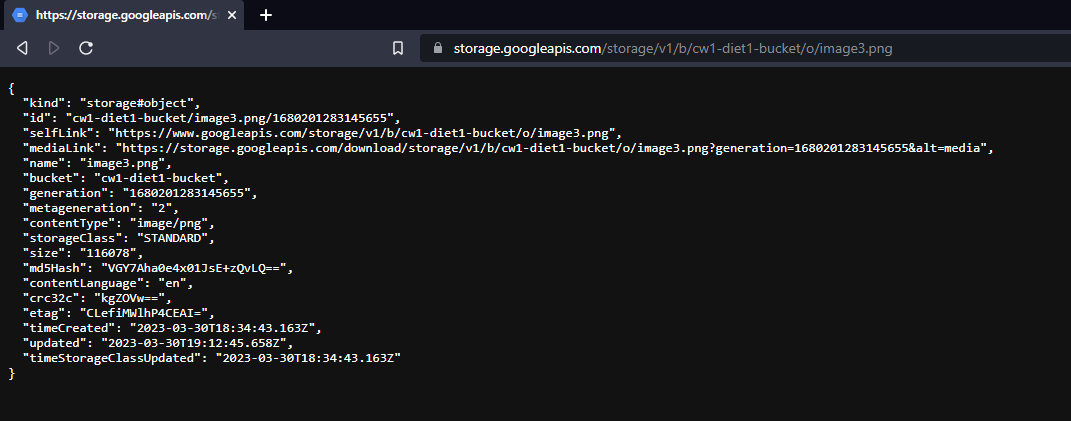
* To get the metadata for **“image1.png”** requested to following API endpoint

<https://storage.googleapis.com/storage/v1/b/cw1-diet1-bucket/o/image2.png>



* To get the metadata for **“image1.png”** requested to following API endpoint

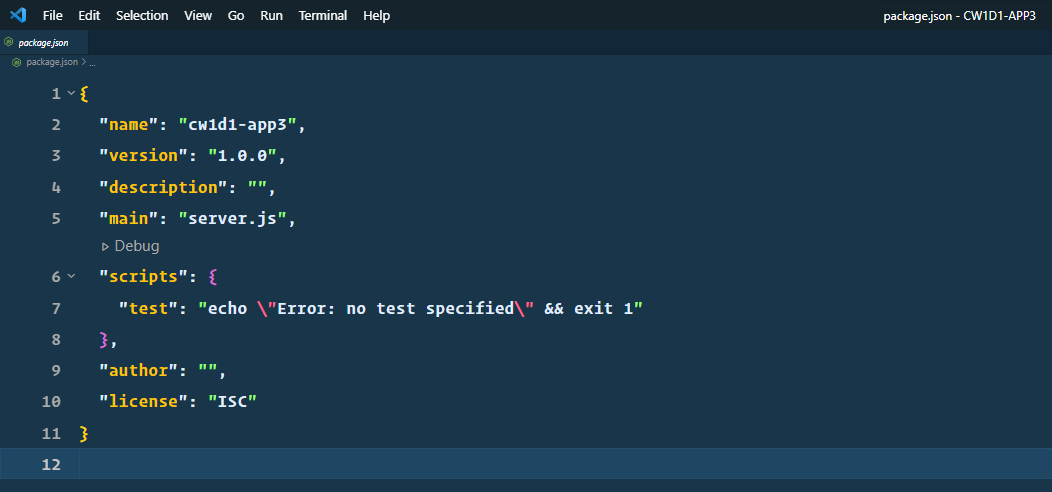
<https://storage.googleapis.com/storage/v1/b/cw1-diet1-bucket/o/image3.png>



## B: Developed and tested a third App Engine app, similar to the one created in Task 2 that serves metadata of each image by requesting the REST API explored in part A.

* Created a new App Engine app using NodeJS as preferred development environment.
* Initialized project following command

npm init



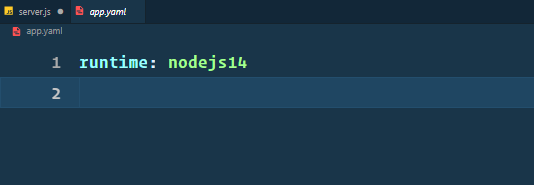
* Installed npm package **“express”** to create a server inside our newly created app using following command.

npm install express

* Installed npm package **“axios”** to request API endpoints.

npm install axios

* Create **“app.yaml”** file with following content



* Create a new **“server.js”** file to serve our app with following content.

const express = require('express');

const axios = require('axios');

const app = express();

app.get('/images', (req, res) => {

res.sendFile(\_\_dirname + '/views/images.html');

});

app.get('/images-metadata/:id', async (req, res) => {

const imageId = parseInt(req.params.id);

const url = `https://storage.googleapis.com/storage/v1/b/cw1-diet1-bucket/o/image${imageId}.png`;

const options = {

method: 'GET',

headers: {

'Content-Type': 'application/json',

},

};

const response = await axios(url, options);

if (response?.data) {

let html = `<html><head><title>Image ${imageId}</title></head><body>

<div><span>ID</span>: <span>${response?.data?.id}</span></div>

<div><span>Self Link</span>: <span>${response?.data?.selfLink}</span></div>

<div><span>Media Link</span>: <span>${response?.data?.mediaLink}</span></div>

<div><span>Name</span>: <span>${response?.data?.name}</span></div>

<div><span>Bucket</span>: <span>${response?.data?.bucket}</span></div>

<div><span>Generation</span>: <span>${response?.data?.generation}</span></div>

<div><span>Meta Generation</span>: <span>${response?.data?.metageneration}</span></div>

<div><span>Content Type</span>: <span>${response?.data?.contentType}</span></div>

<div><span>Storage Class</span>: <span>${response?.data?.storageClass}</span></div>

<div><span>Size</span>: <span>${response?.data?.size}</span></div>

<div><span>Creation Time</span>: <span>${response?.data?.timeCreated}</span></div>

<div><span>Last Updated</span>: <span>${response?.data?.updated}</span></div>

</body></html>`;

res.send(html);

} else {

let html = `<html><head><title>Image ${imageId}</title></head><body>

<h1>Image Not Found</h1>

</body></html>`;

res.send(html);

}

});

app.listen(process.env.PORT || 8070, () => {

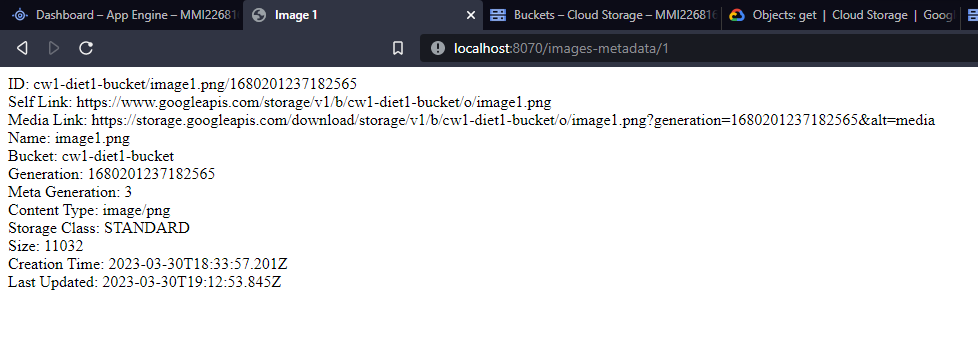
console.log('App listening on port 8090');

});

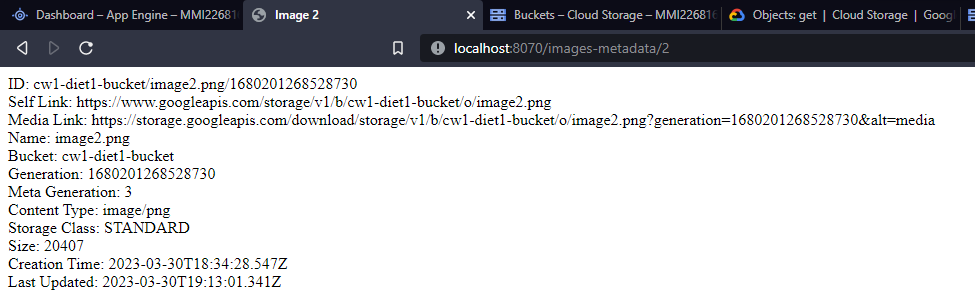
* Started app on **“localhost:8070”** by following command

npm start

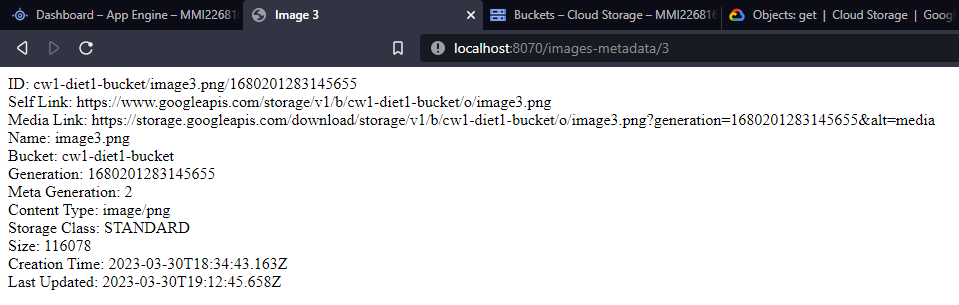
* Accessed metadata for **“image1”** on localhost by [**http://localhost:8070/images-metadata/1**](http://localhost:8070/images-metadata/1)



* Accessed metadata for **“image2”** on localhost by [**http://localhost:8070/images-metadata/2**](http://localhost:8070/images-metadata/2)

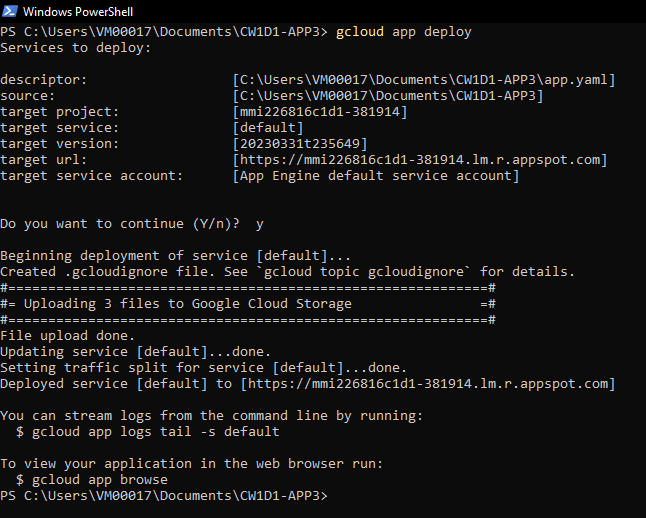


* Accessed metadata for **“image3”** on localhost by [**http://localhost:8070/images-metadata/3**](http://localhost:8070/images-metadata/3)

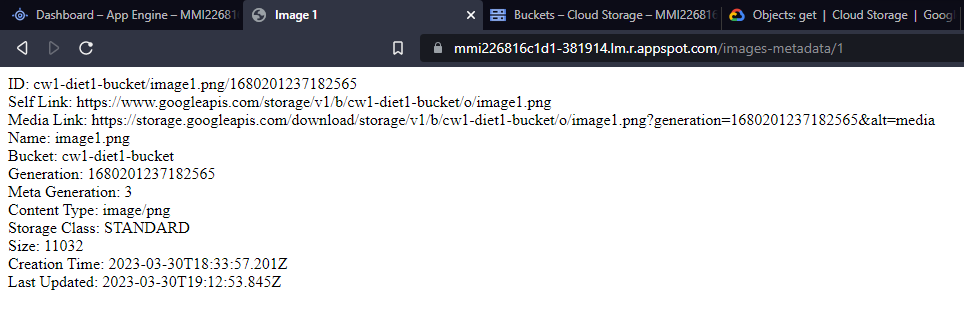


* To deploy app on Google Cloud executed following command

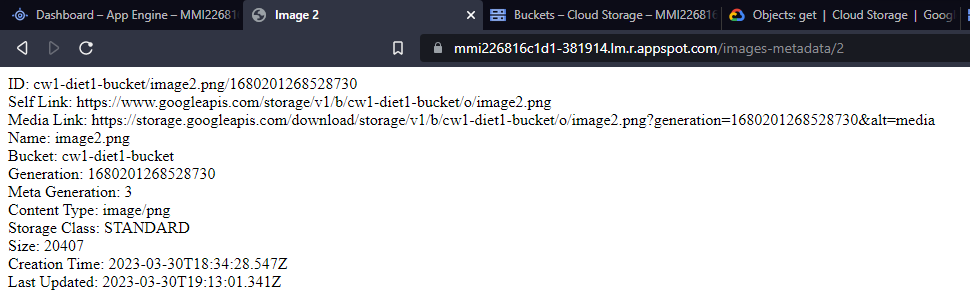
gcloud app deploy



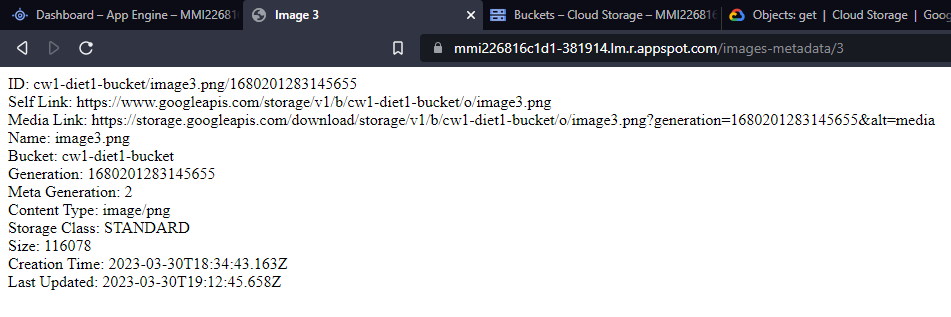
* Accessed metadata for **“image1”** on remote by [**https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/1**](https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/1)



* Accessed metadata for **“image2”** on remote by [**https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/2**](https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/2)

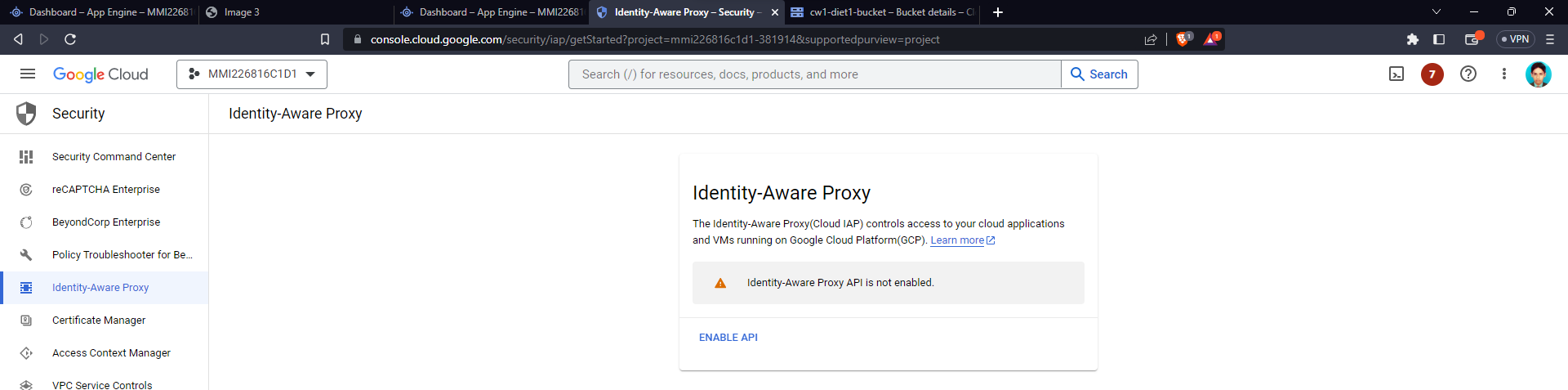


* Accessed metadata for **“image3”** on remote by [**https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/3**](https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/3)

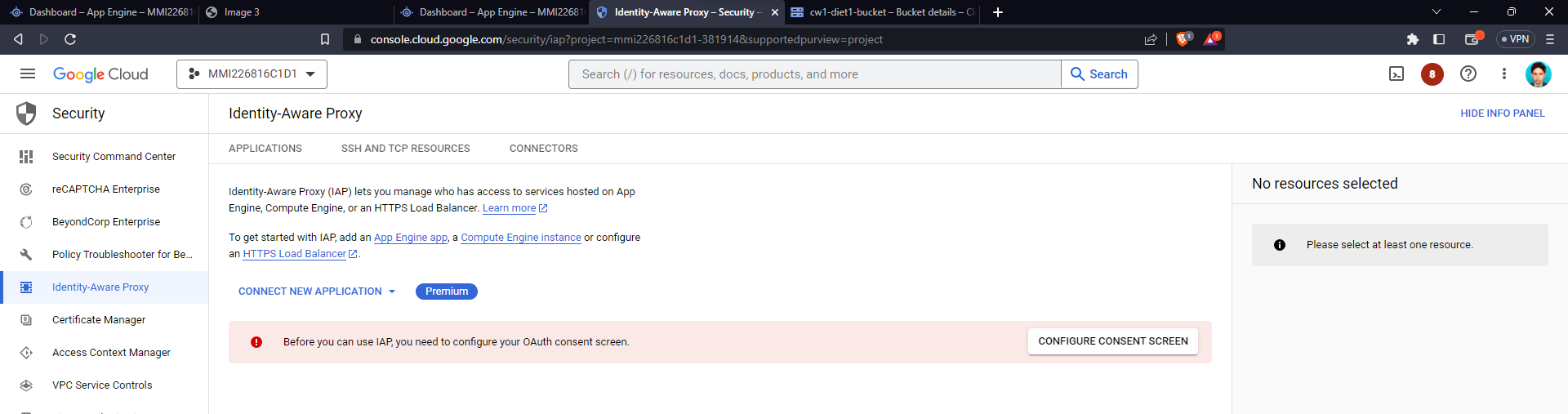


## C: Secured App Engine using Google Identity-Aware Proxy (IAP) by following steps:

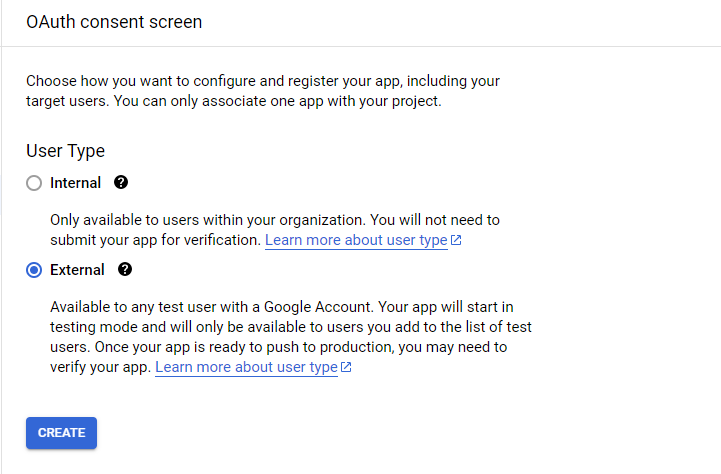
* Visited Identity-Aware Proxy page
* Selected my project **“mmi226816c1d1”**
* Enabled IAP



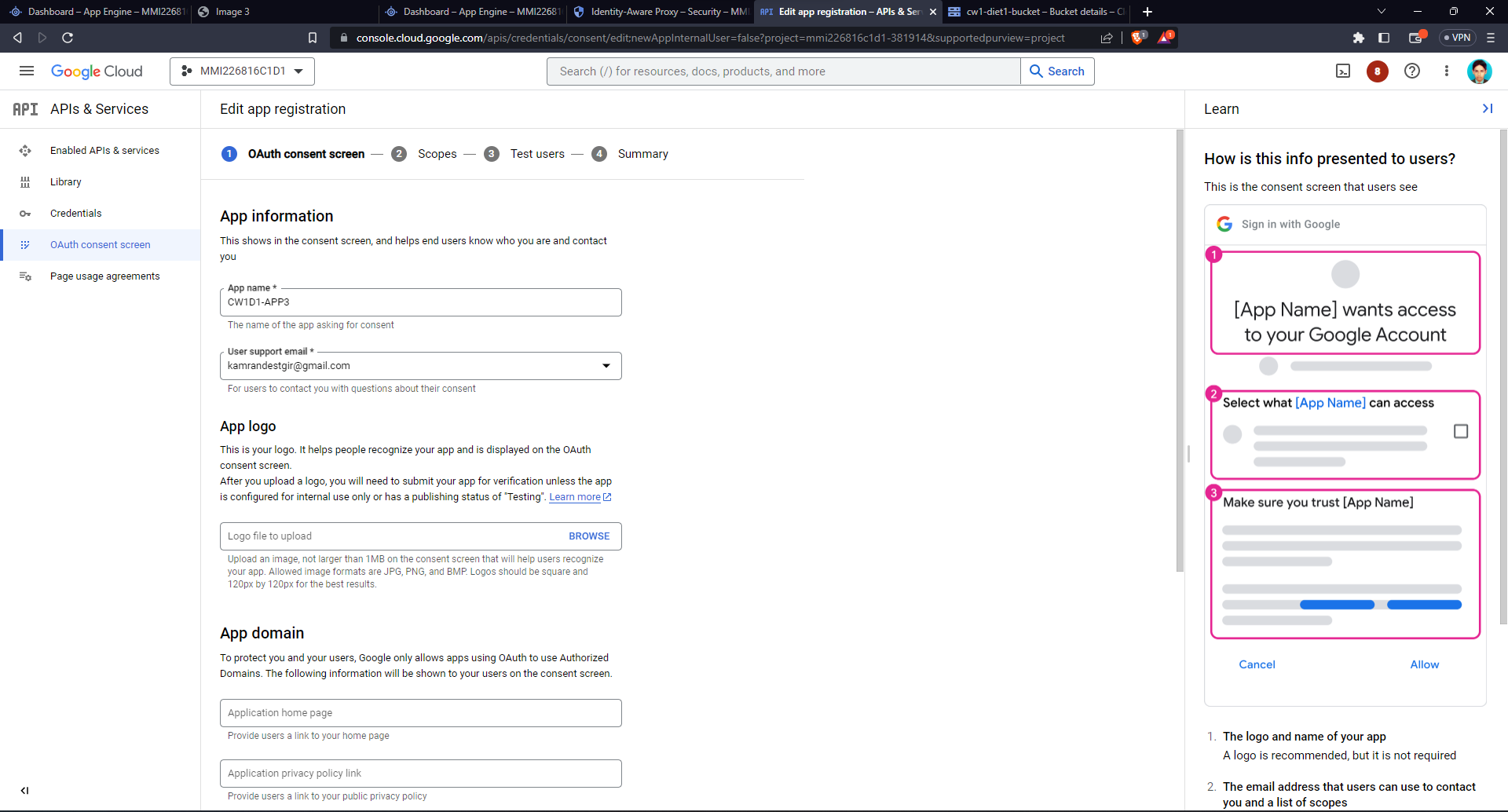
* Visited IAP page.
* To enable IAP it is required to configure **OAuth Consent Screen**



* Clicked on **Configure Consent Screen**
* Selected **External** option



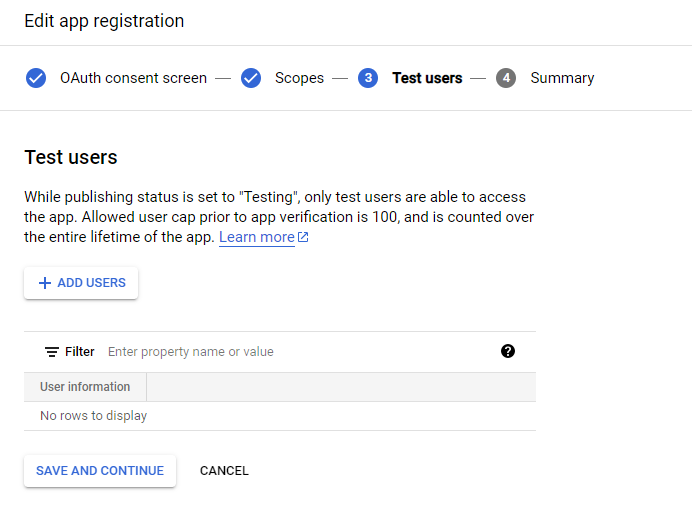
* Provided required app information



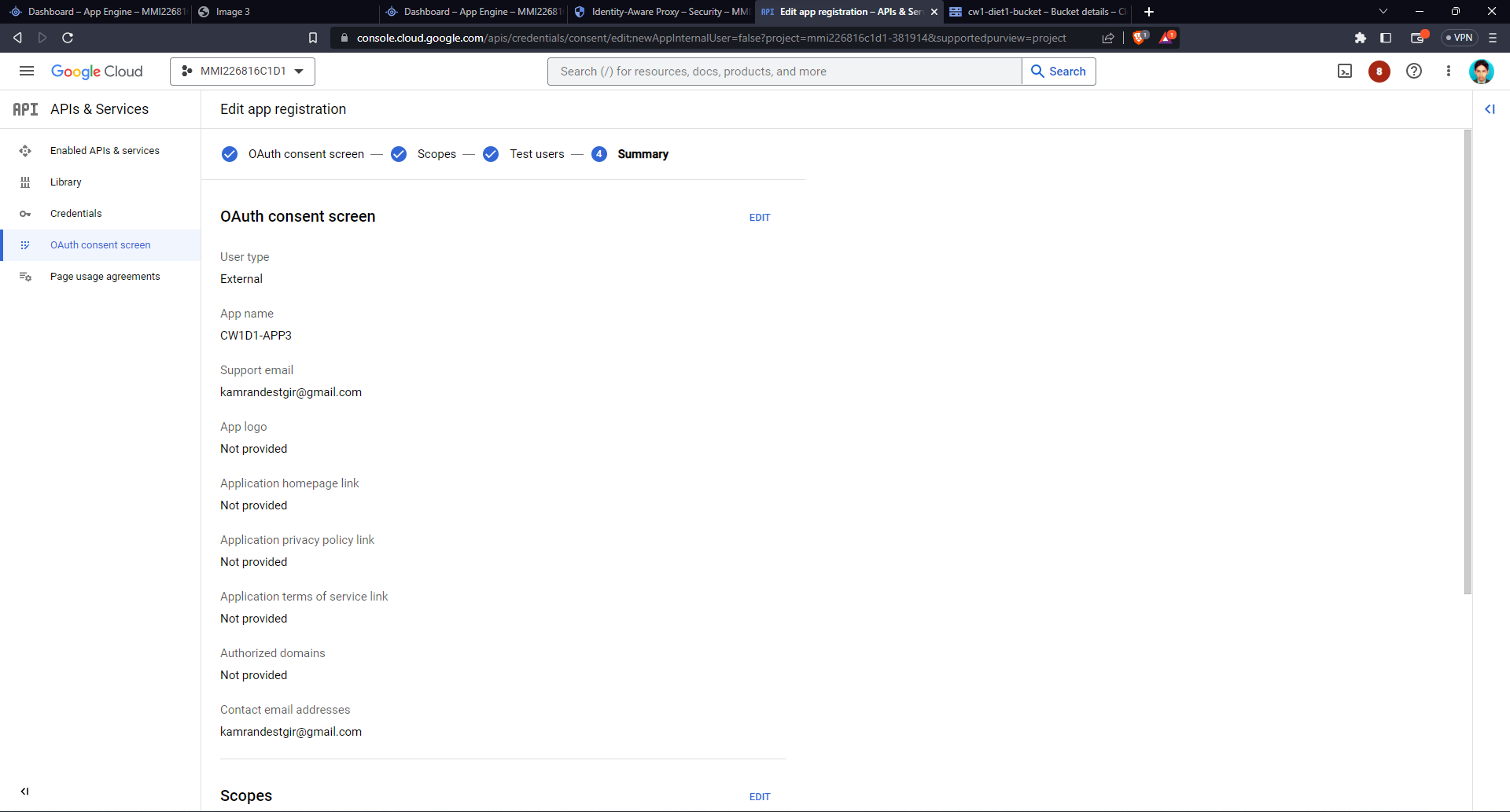
* Saved default Scopes setting



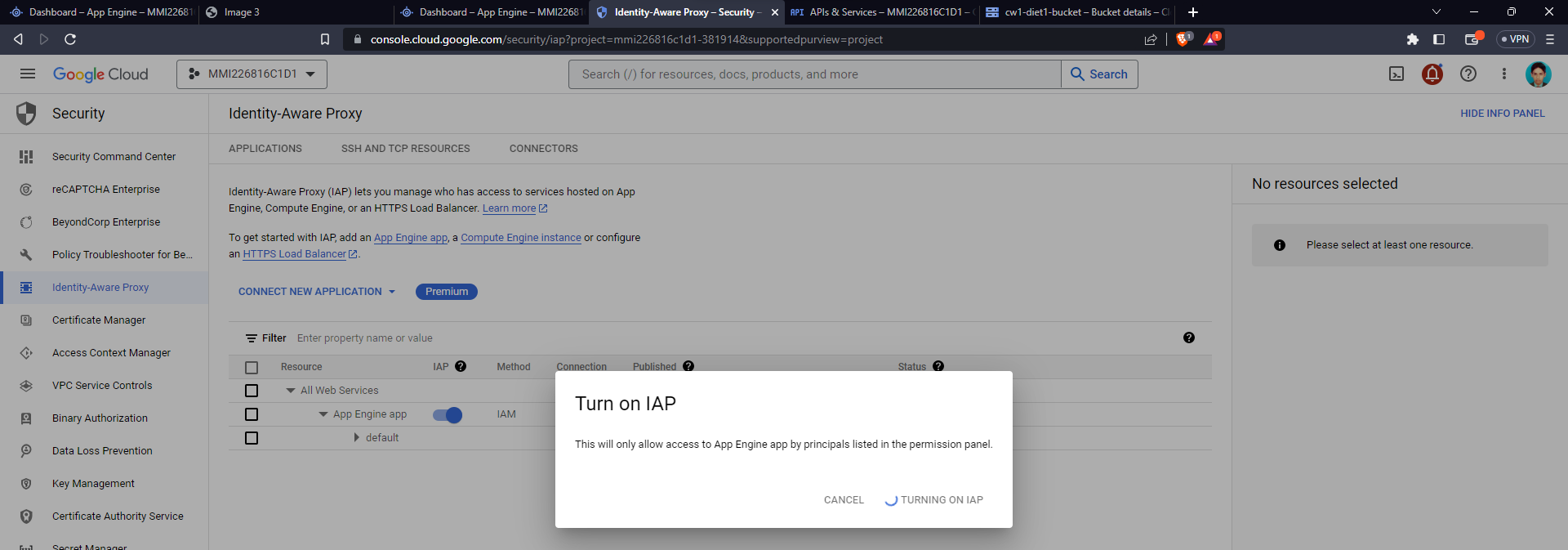
* Saved default Test Users settings



* Reviewed summary and finished process



* After completing OAuth Consent Screen setup revisited IAP page and turned on IAP for app by toggling the IAP option.



* To test IAP working requested [**https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/1**](https://mmi226816c1d1-381914.lm.r.appspot.com/images-metadata/1)
* On visit it asks for Google Sign-In which means it is secured by IAP 