```
C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud auth login
Your browser has been opened to visit:

    https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=32555940559.apps.googleuserconte
A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fappengine.admin
ogleapis.com%2Fauth%2Faccounts.reauth&state=oq0Yvh4YYOSg6BJuKXzg4iOQ6ErqLF&access_type=offline&code_challen


You are now logged in as [s222610226@deakin.edu.au].
Your current project is [sit313-frontend].  You can change this setting by running:
  $ gcloud config set project PROJECT_ID

C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud config get-value project
sit313-frontend

C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud projects list
PROJECT_ID                NAME                       PROJECT_NUMBER
des-webana-datadictionary  Des WebAna DataDictionary  1038693078675
sit323-25t1-duong-102550c  sit323-25t1-duong-102550c  87055954794

C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud config set project sit323-25t1-duong-102550c
Updated property [core/project].

C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud config get-value project
sit323-25t1-duong-102550c
```

Accessing the assigned Google Cloud Project by the following steps:

1. Log in Google Cloud via Google Cloud SDK Shell through command “gcloud auth login”.
2. Enter Deakin Email account in the pop-up Google login page.
3. Enter command “gcloud project list” to check which projects are assigned to the login accound.
4. Type in “gcloud config set project <project_id>”, in this case, “sit323-25t1-duong-10255c” to switch to that project.
5. (Optional) Re-check the current project by “gcloud config get-value project”.

Using Artifact Registry

1. Create a repository:
    - Type for “Artifact Repository” on Search box and click enable API.
    - Click on the Pin icon right-next to “Artifact Registry” upon the page as shown in the image below (Image 1).

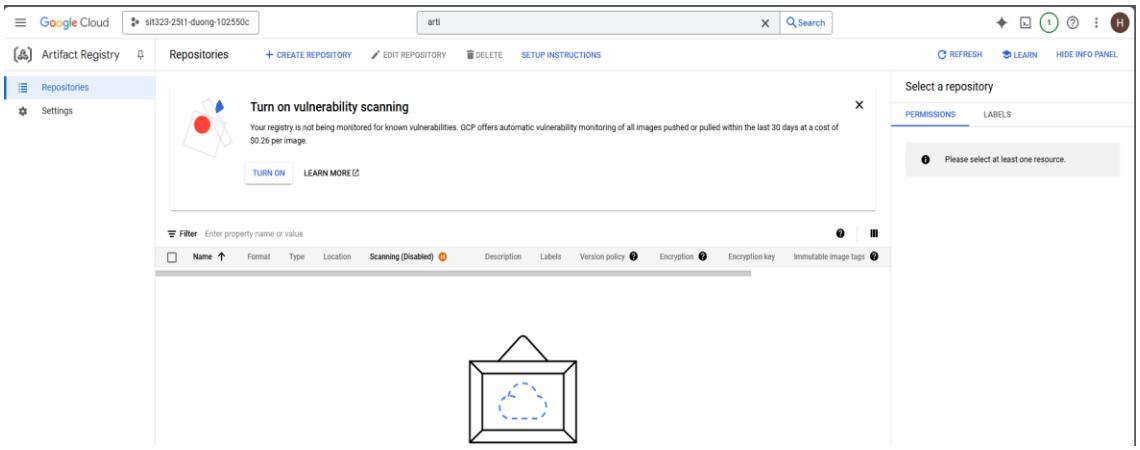

Image 1 – "Artifact Registry" main page

- Choose "Create repository" with the following selections:
  i. Name: "calculator-repo".
  ii. Format: "Docker".
  iii. Mode: "Standard".
  iv. Location Type: "Region".
    1. Region: "australia-southeast2 (Melbourne)"
    2. Description: "Repository for Calculator application (Task 4.2C).".
  v. Leave other selections as their initial state and click "Create" button.

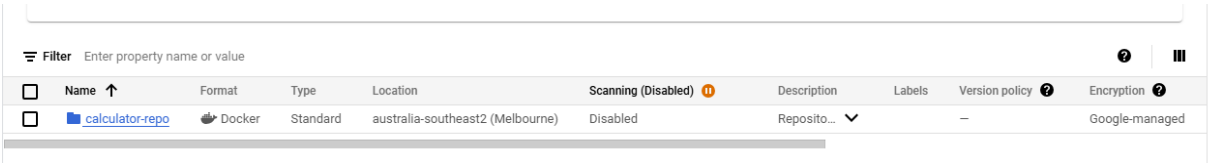

Image 2 – calculator-repo is successfully created.

2. Access permission to created Artifact Registry
   - (Optional) Checking the current active account with "gcloud auth list". (Image 3)

```
C:\Users\Lac T. Duong\AppData\Local\Google\Cloud SDK>gcloud auth list
     Credentialed Accounts
ACTIVE  ACCOUNT
*       s222610226@deakin.edu.au
        tamlac20121996@gmail.com

To set the active account, run:
    $ gcloud config set account `ACCOUNT`
```

Image 3 – Checking the current active account.

- Artifact Registry is a private registry which will requiring permission-granted users and systems to be able to push and pull image from repositories created by this method.
- To grant that permission to client application, Docker, enter the following command: “gcloud auth configure-docker <Region_selected>-docker.pkg.dev”, which is **australia-southeast2** in this case.
    - This command will using the login account, from “gcloud auth login” to create a credential helper and store the credential info into configuration file of Docker (~/.docker/config.json) so that whenever Docker application need to interact with Google Cloud like push and pull images, Docker will “contact” with this platform via Credential Helper “gcloud” to get the access token. (Image 4)
    - Note: this permission is only granted to the input region in the command, and will be required to re-enter it again with new <Region_selected> input that matching the target repository in Artifact Registry.

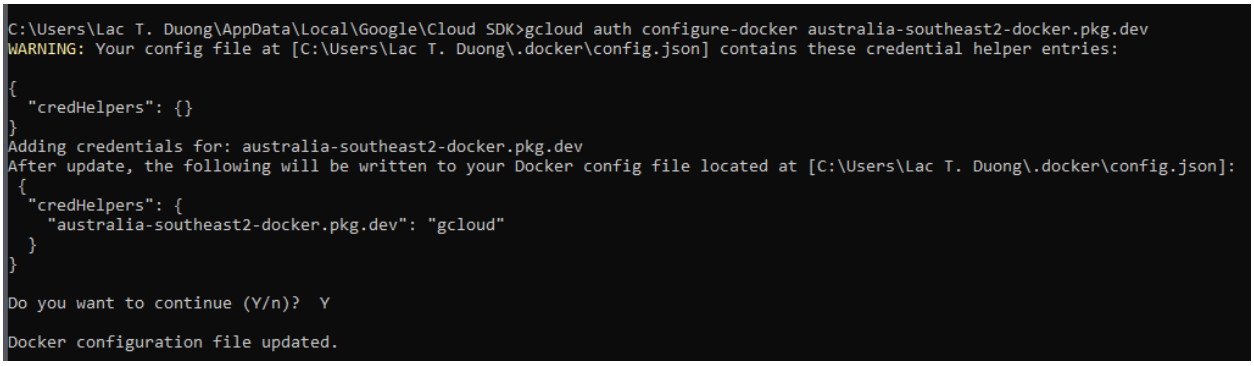

Image 4 – Authorize the accessibility of local Docker to Artifact Registry repository.

3. Pushing Docker Image to Artifact Registry (AR) repository
    - (Optional) Re-checking if the target image is still present in the local host by starting up Docker Desktop & run command “docker images” in the terminal. (Image 5)

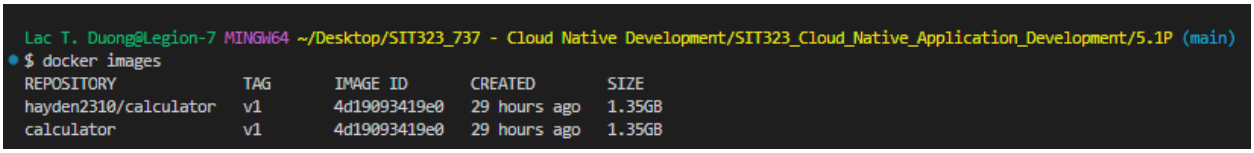

Image 5 – Images that are currently present in localhost

- Change the current tag of the target image to meet Artifact Registry standard which is “<Region_selected>-docker.pkg.dev/<project-id>/<AR-repo-name>/<image-name>:<tag>” by command “docker tag …” (Image 6)

```
Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/5.1P (main)
$ docker tag calculator:v1 australia-southeast2-docker.pkg.dev/sit323-25t1-duong-102550c/calculator-repo/calculator:v1

Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/5.1P (main)
$ docker images
REPOSITORY                                                                      TAG       IMAGE ID       CREATED         SIZE
australia-southeast2-docker.pkg.dev/sit323-25t1-duong-102550c/calculator-repo/calculator   v1        4d19093419e0   35 hours ago   1.35GB
hayden2310/calculator                                                           v1        4d19093419e0   35 hours ago   1.35GB
calculator                                                                      v1        4d19093419e0   35 hours ago   1.35GB
```

Image 6 – New image is created through “docker tag …” command.

- Push the newly created image to “calculator-repo” created on Google Cloud platform through command “docker push <image-name>”. (Image 7, 8)

```
Lac T. Duong@Legion-7 MINGW64 ~/Desktop/SIT323_737 - Cloud Native Development/SIT323_Cloud_Native_Application_Development/5.1P (main)
$ docker push australia-southeast2-docker.pkg.dev/sit323-25t1-duong-102550c/calculator-repo/calculator:v1
The push refers to repository [australia-southeast2-docker.pkg.dev/sit323-25t1-duong-102550c/calculator-repo/calculator]
0f40644d8e72: Pushed
24e7b7ec8536: Pushed
8b52281f0e68: Pushed
2e7e0de75b77: Pushed
6063f15db639: Pushed
3a335708223f: Pushed
1424ff7cd636: Pushed
28603fe995a1: Pushed
4c13ea2c0b02: Pushed
2ed6a19677f5: Pushed
d2f7abddd607: Pushed
53babe930602: Pushed
v1: digest: sha256:0f3b0761ca471aad2300579c870c7a179c58c6c125fa2dff84241dc91d36ea28 size: 2842
```

Image 7 – Image of the original “calculator” had successfully pushed to Google Cloud.

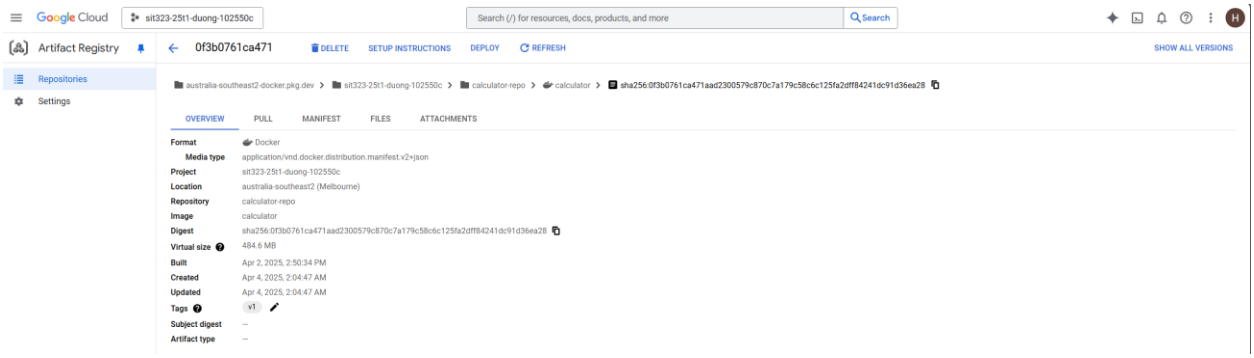

Image 8 – The image is found in “calculator-repo”.

4. (Optional) Pulling the pushed image back to local host
    - Remove the newly created (tagged) image in the localhost through command “docker rmi <image-name”.

- The pushed image in Google Cloud can be pulled by one of these two ways:

Image 9 – Ways to pull the image down to the local host.

- Results (Image 10, 11):

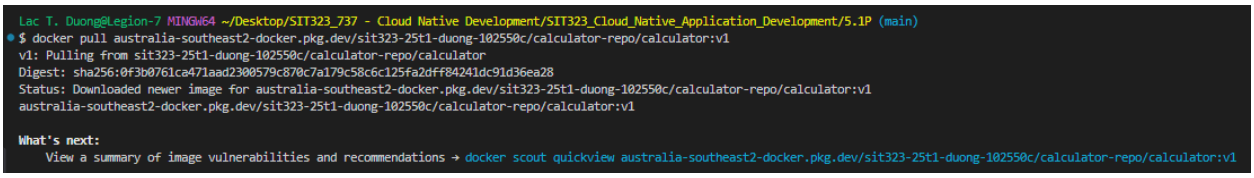

Image 10 – Image pulled by "Pull by tag" method. (Ideally)

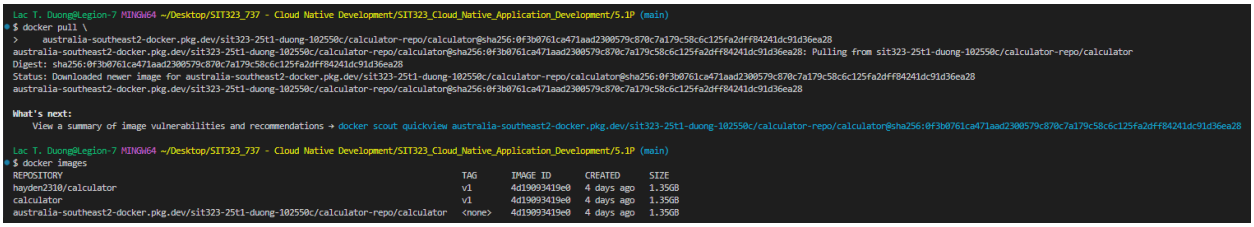

Image 11 – Image pulled by "Pull by digest" method.

- It was found that pulling the image via "Pull by tag" is more convenient because the pulled image will tagged with the designated tag represents in input docker command – make it easier to be interacted with (v1 vs @sha256:0f3b0761ca471aad2300579c870c7a179c58c6c125fa2dff84241dc91d36ea28).

5. Build a container from the pulled image
   - Use command "docker run …" upon the pulled image to generate a container from this image. (Image 12)

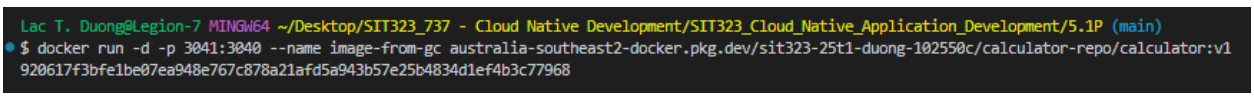

Image 12 – A container is successfully created from the pulled image.

- Check the container with Docker Desktop application to make sure it is running and be able to access to through the provided link in “Port(s)” section of this container. (Image 13).

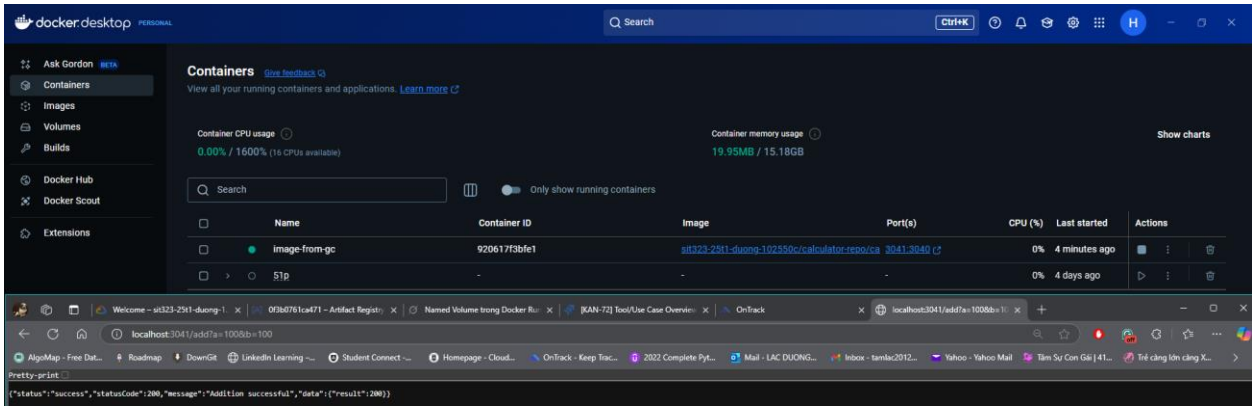

Image 13 – A container named “image-from-gc”, generated from pulled image is currently running and accessible via “https:localhost:3041”