# DevOps - Build, Test, Deploy, Monitor

Containerize a Basic Express JS Application and Deploy it on a Cloud Platform.

https://github.com/sherenaLIM

# Build

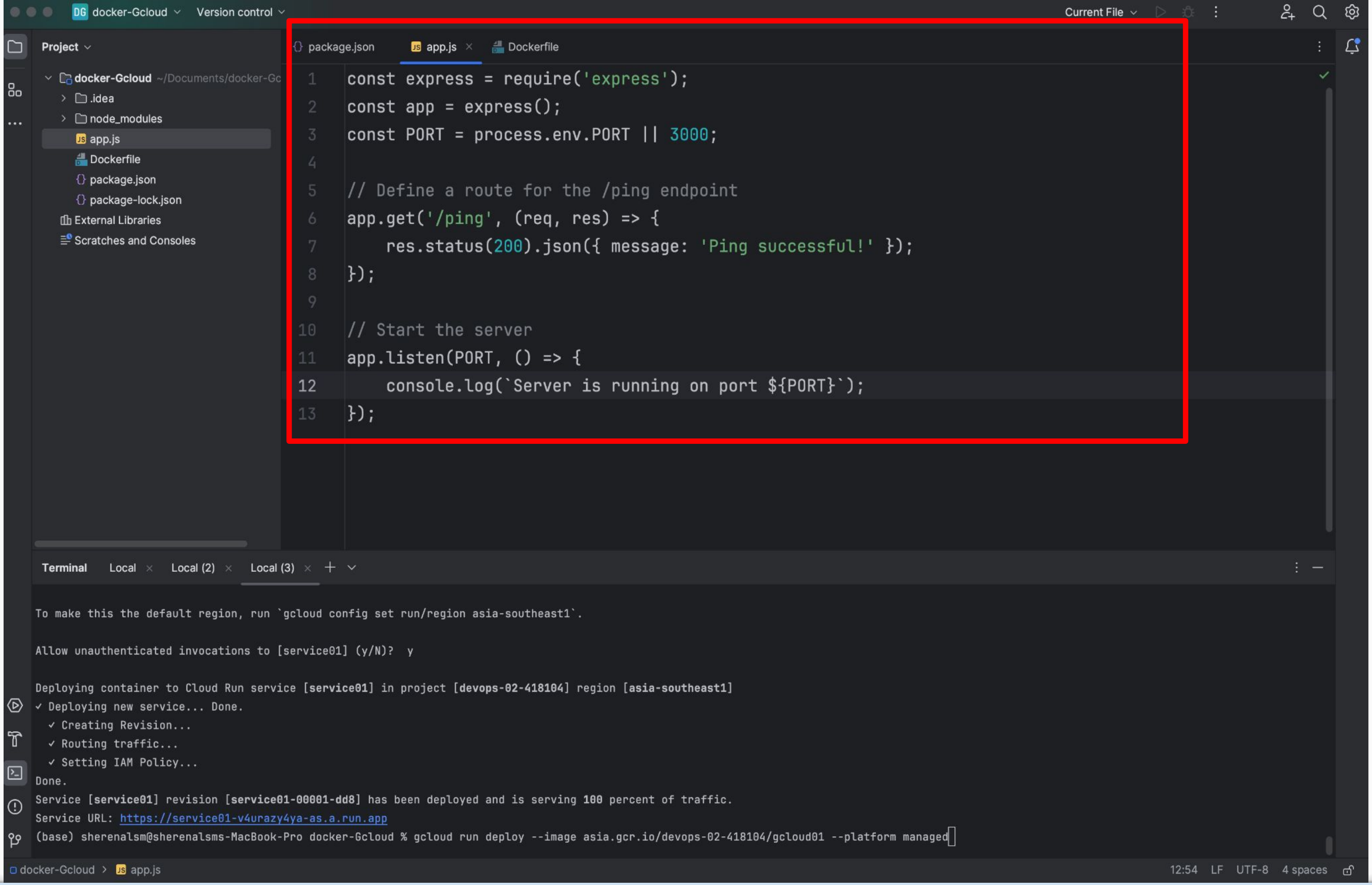
Develop a Simple Express JS Application.

https://github.com/sherenaLIM

## 1: Generate Basic Node JS Application

**Why Node JS instead of popular alternatives (e.g., Angular/React JS + Spring Boot application using Jhipster)?**

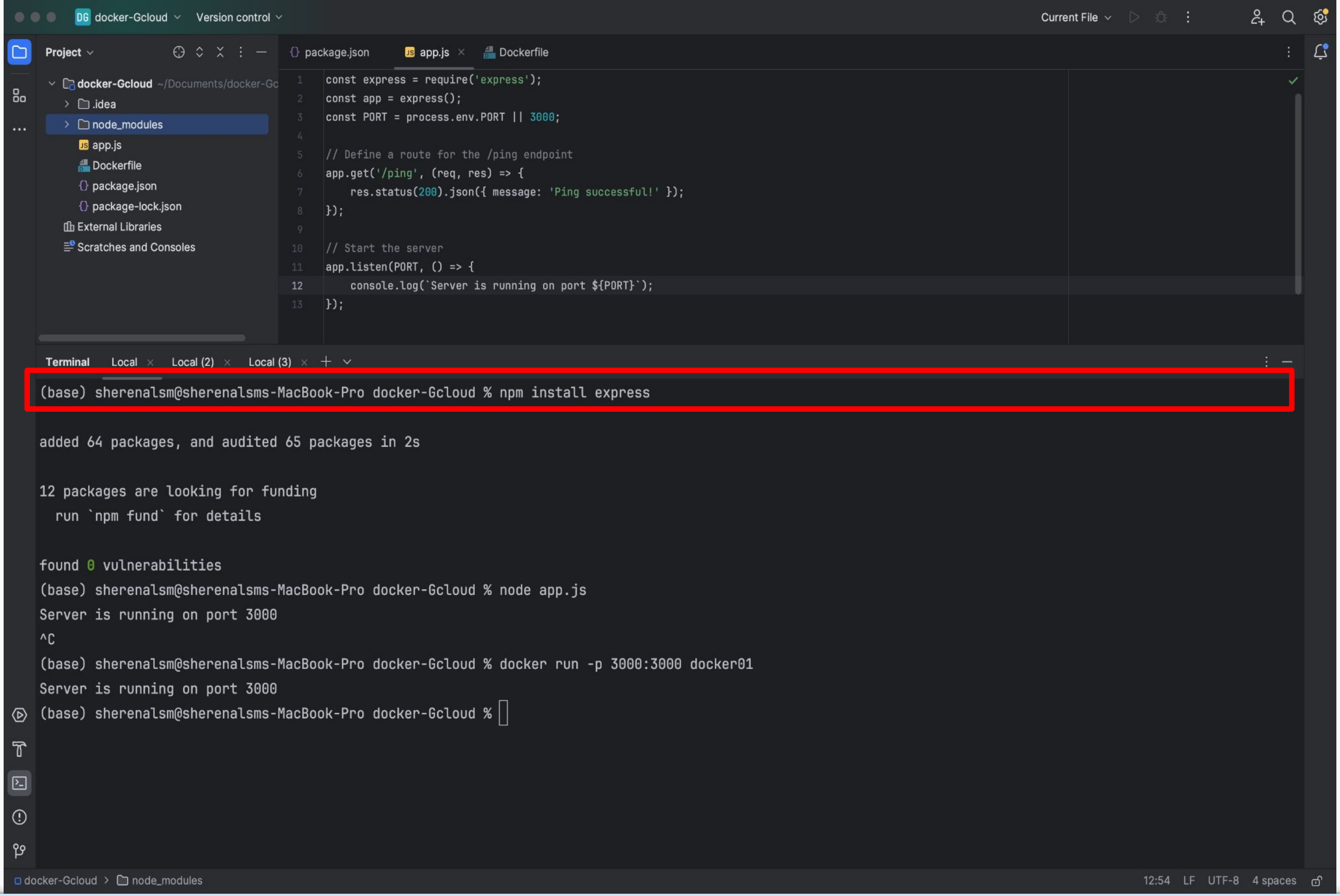
- lightweight, provides minimalistic framework for building web applications and APIs in node.js
- suitable for small to medium-sized projects where simplicity and flexibility are prioritized
- objective : **demonstrate process of dockerizing a basic web application and deploying it on google cloud**



```
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;

// Define a route for the /ping endpoint
app.get('/ping', (req, res) => {
    res.status(200).json({ message: 'Ping successful!' });
});

// Start the server
app.listen(PORT, () => {
    console.log(`Server is running on port ${PORT}`);
});
```

```
To make this the default region, run `gcloud config set run/region asia-southeast1`.

Allow unauthenticated invocations to [service01] (y/N)?  y

Deploying container to Cloud Run service [service01] in project [devops-02-418104] region [asia-southeast1]
✓ Deploying new service... Done.
  ✓ Creating Revision...
  ✓ Routing traffic...
  ✓ Setting IAM Policy...
Done.
Service [service01] revision [service01-00001-dd8] has been deployed and is serving 100 percent of traffic.
Service URL: https://service01-v4urazy4ya-as.a.run.app
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % gcloud run deploy --image asia.gcr.io/devops-02-418104/gcloud01 --platform managed
```

1a. Generate a basic express.js project with a single endpoint /ping that returns a successful response.



```
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;

// Define a route for the /ping endpoint
app.get('/ping', (req, res) => {
    res.status(200).json({ message: 'Ping successful!' });
});

// Start the server
app.listen(PORT, () => {
    console.log(`Server is running on port ${PORT}`);
});
```

```
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % npm install express

added 64 packages, and audited 65 packages in 2s

12 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % node app.js
Server is running on port 3000
^C
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker run -p 3000:3000 docker01
Server is running on port 3000
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud %
```

1b. Generate node_modules folder (installation of dependencies) using command `npm install`.

# Build

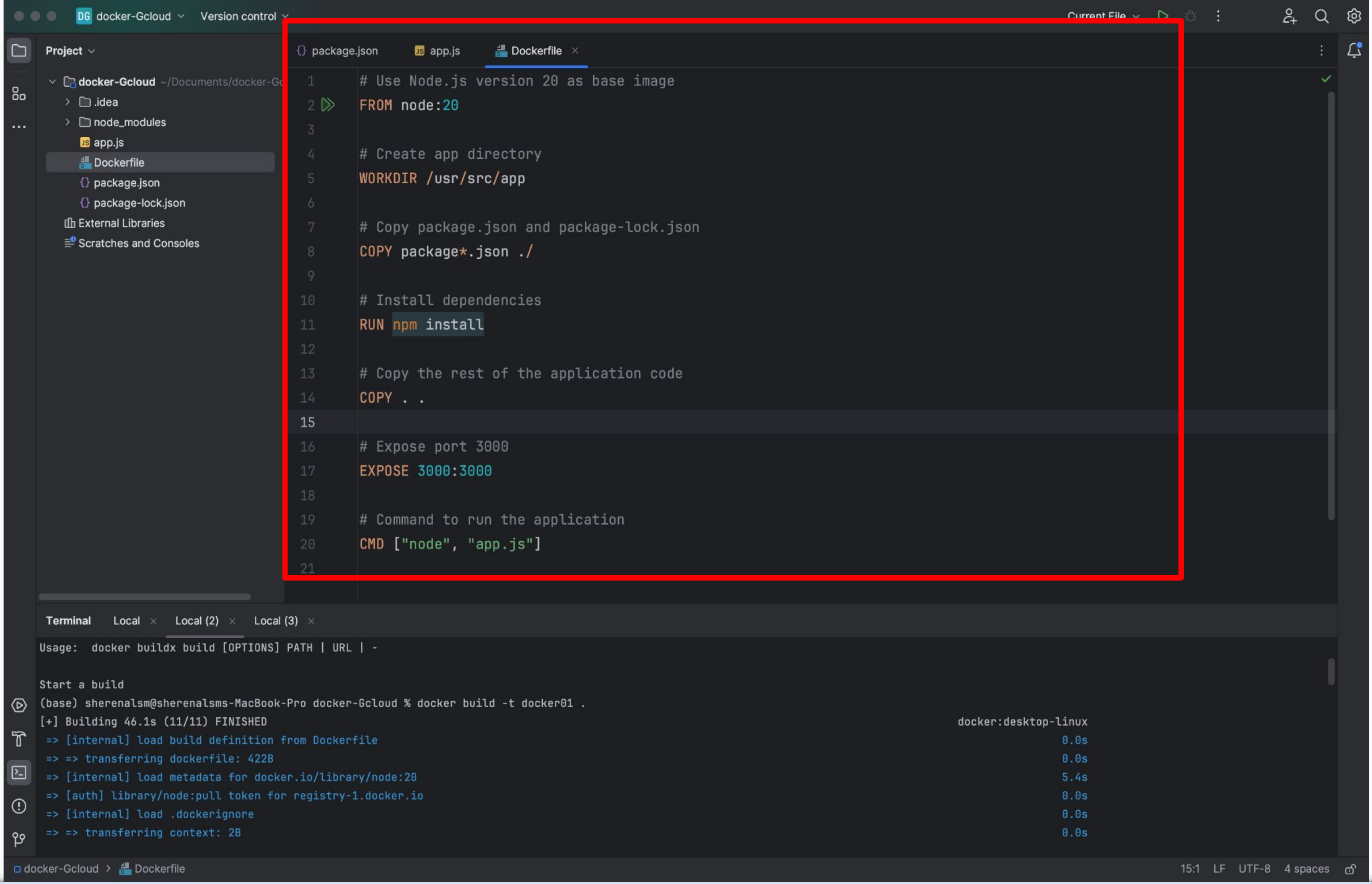
Containerize your Application. Build a Docker Image Locally.

https://github.com/sherenaLIM

## 2: Containerize the Application using Docker

**How does Containerization work?**

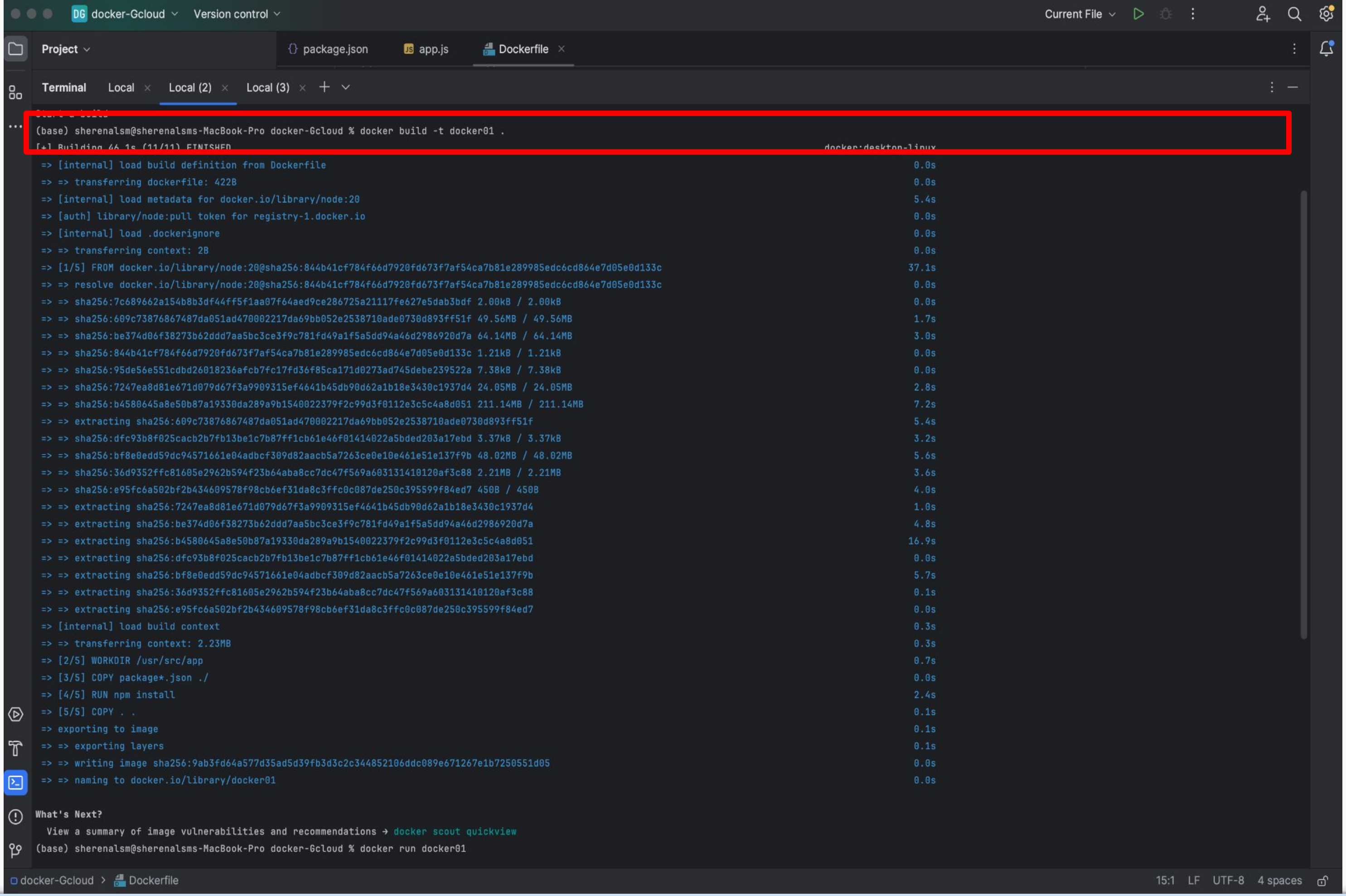
- **dockerfile** : specifies instructions to assemble the image
- encapsulates application code, runtime, and dependencies into a portable unit
- **docker image** : a static, lightweight, standalone, executable software package that you can run using the command `docker run` .
- **docker instance** : a runtime instance of an image
    - a.k.a. container or a lightweight virtual machine



```dockerfile
# Use Node.js version 20 as base image
FROM node:20

# Create app directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .

# Expose port 3000
EXPOSE 3000:3000

# Command to run the application
CMD ["node", "app.js"]
```

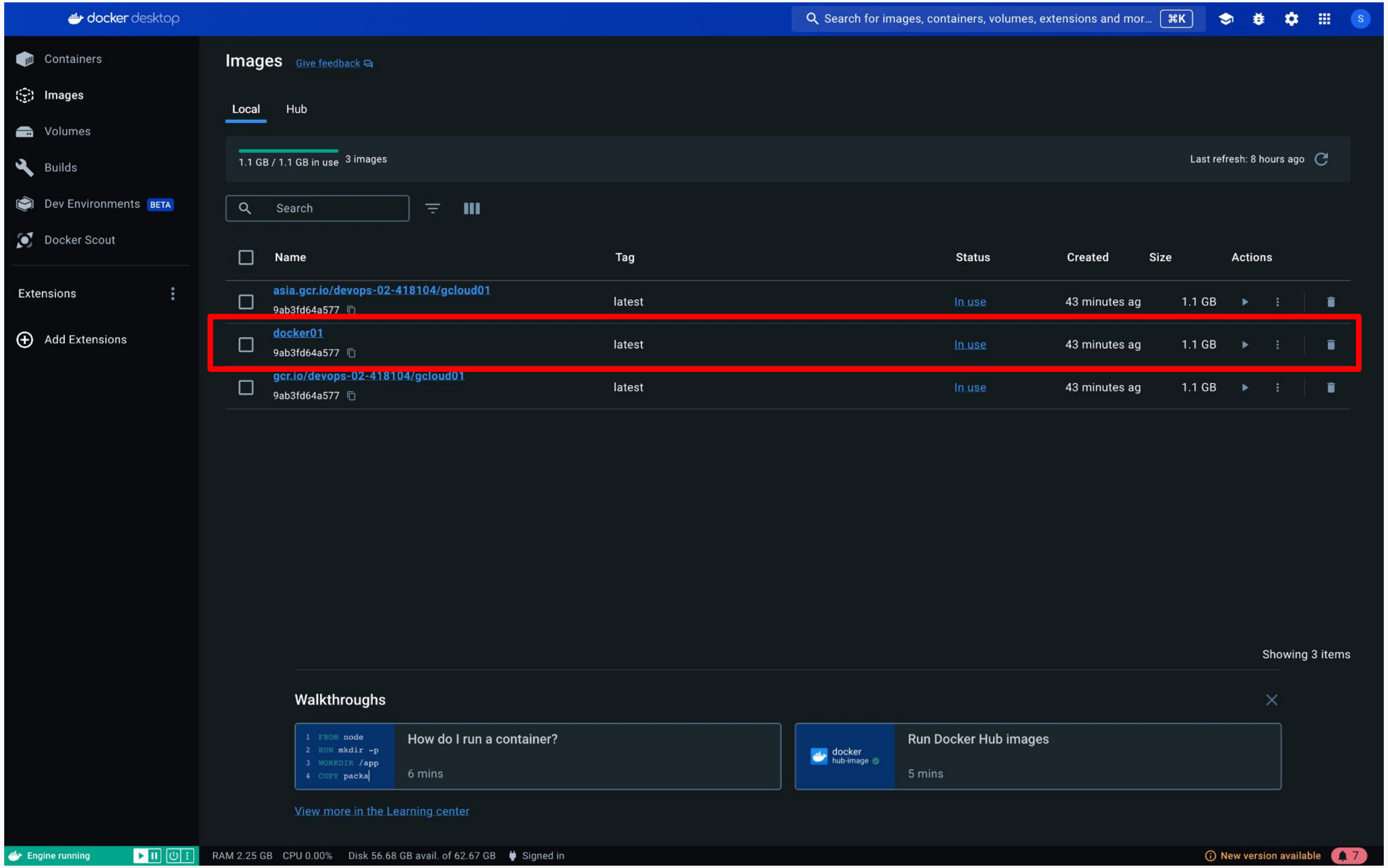
```
Usage:  docker buildx build [OPTIONS] PATH | URL | -

Start a build
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker build -t docker01 .
[+] Building 46.1s (11/11) FINISHED                                          docker:desktop-linux
 => [internal] load build definition from Dockerfile                                         0.0s
 => => transferring dockerfile: 422B                                                         0.0s
 => [internal] load metadata for docker.io/library/node:20                                   5.4s
 => [auth] library/node:pull token for registry-1.docker.io                                  0.0s
 => [internal] load .dockerignore                                                            0.0s
 => => transferring context: 2B                                                              0.0s
```

2a. Create a Dockerfile which specifies the instructions for assembling the image.



```
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker build -t docker01 .
[+] Building 46.1s (11/11) FINISHED                                                                   docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                  0.0s
 => => transferring dockerfile: 422B                                                                                  0.0s
 => [internal] load metadata for docker.io/library/node:20                                                            5.4s
 => [auth] library/node:pull token for registry-1.docker.io                                                           0.0s
 => [internal] load .dockerignore                                                                                     0.0s
 => => transferring context: 2B                                                                                       0.0s
 => [1/5] FROM docker.io/library/node:20@sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c     37.1s
 => => resolve docker.io/library/node:20@sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c      0.0s
 => => sha256:7c689662a154b8b3df44ff5f1aa07f64aed9ce286725a21117fe627e5dab3bdf 2.00kB / 2.00kB                        0.0s
 => => sha256:609c73876867487da051ad470002217da69bb052e2538710ade0730d893ff51f 49.56MB / 49.56MB                      1.7s
 => => sha256:be374d06f38273b62ddd7aa5bc3ce3f9c781fd49a1f5a5dd94a46d2986920d7a 64.14MB / 64.14MB                      3.0s
 => => sha256:844b41cf784f66d7920fd673f7af54ca7b81e289985edc6cd864e7d05e0d133c 1.21kB / 1.21kB                        0.0s
 => => sha256:95de56e551cdbd26018236afcb7fc17fd36f85ca171d0273ad745debe239522a 7.38kB / 7.38kB                        0.0s
 => => sha256:7247ea8d81e671d079d67f3a9909315ef4641b45db90d62a1b18e3430c1937d4 24.05MB / 24.05MB                      2.8s
 => => sha256:b4580645a8e50b87a19330da289a9b1540022379f2c99d3f0112e3c5c4a8d051 211.14MB / 211.14MB                    7.2s
 => => extracting sha256:609c73876867487da051ad470002217da69bb052e2538710ade0730d893ff51f                             5.4s
 => => sha256:dfc93b8f025cacb2b7fb13be1c7b87ff1cb61e46f01414022a5bded203a17ebd 3.37kB / 3.37kB                        3.2s
 => => sha256:bf8e0edd59dc94571661e04adbcf309d82aacb5a7263ce0e10e461e51e137f9b 48.02MB / 48.02MB                      5.6s
 => => sha256:36d9352ffc81605e2962b594f23b64aba8cc7dc47f569a603131410120af3c88 2.21MB / 2.21MB                        3.6s
 => => sha256:e95fc6a502bf2b434609578f98cb6ef31da8c3ffc0c087de250c395599f84ed7 450B / 450B                            4.0s
 => => extracting sha256:7247ea8d81e671d079d67f3a9909315ef4641b45db90d62a1b18e3430c1937d4                             1.0s
 => => extracting sha256:be374d06f38273b62ddd7aa5bc3ce3f9c781fd49a1f5a5dd94a46d2986920d7a                             4.8s
 => => extracting sha256:b4580645a8e50b87a19330da289a9b1540022379f2c99d3f0112e3c5c4a8d051                            16.9s
 => => extracting sha256:dfc93b8f025cacb2b7fb13be1c7b87ff1cb61e46f01414022a5bded203a17ebd                             0.0s
 => => extracting sha256:bf8e0edd59dc94571661e04adbcf309d82aacb5a7263ce0e10e461e51e137f9b                             5.7s
 => => extracting sha256:36d9352ffc81605e2962b594f23b64aba8cc7dc47f569a603131410120af3c88                             0.1s
 => => extracting sha256:e95fc6a502bf2b434609578f98cb6ef31da8c3ffc0c087de250c395599f84ed7                             0.0s
 => [internal] load build context                                                                                     0.3s
 => => transferring context: 2.23MB                                                                                   0.3s
 => [2/5] WORKDIR /usr/src/app                                                                                        0.7s
 => [3/5] COPY package*.json ./                                                                                       0.0s
 => [4/5] RUN npm install                                                                                             2.4s
 => [5/5] COPY . .                                                                                                    0.1s
 => exporting to image                                                                                                0.1s
 => => exporting layers                                                                                               0.1s
 => => writing image sha256:9ab3fd64a577d35ad5d39fb3d3c2c344852106ddc089e671267e1b7250551d05                          0.0s
 => => naming to docker.io/library/docker01                                                                           0.0s

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker run docker01
```

2b. Navigate to the directory containing your Dockerfile and run the command `docker build`.



Images

| Name | Tag | Status | Created | Size |
|---|---|---|---|---|
| asia.gcr.io/devops-02-418104/gcloud01<br>9ab3fd64a577 | latest | In use | 43 minutes ag | 1.1 GB |
| docker01<br>9ab3fd64a577 | latest | In use | 43 minutes ag | 1.1 GB |
| gcr.io/devops-02-418104/gcloud01<br>9ab3fd64a577 | latest | In use | 43 minutes ag | 1.1 GB |

1.1 GB / 1.1 GB in use 3 images

Last refresh: 8 hours ago

Showing 3 items

2c. Docker image ‘docker01’ successfully generated !

## Portability + Isolation = Replicability

**What problem does Containerisation solve?**

- image can be deployed and run consistently across various environments
- application remains unaffected by changes to the underlying infrastructure

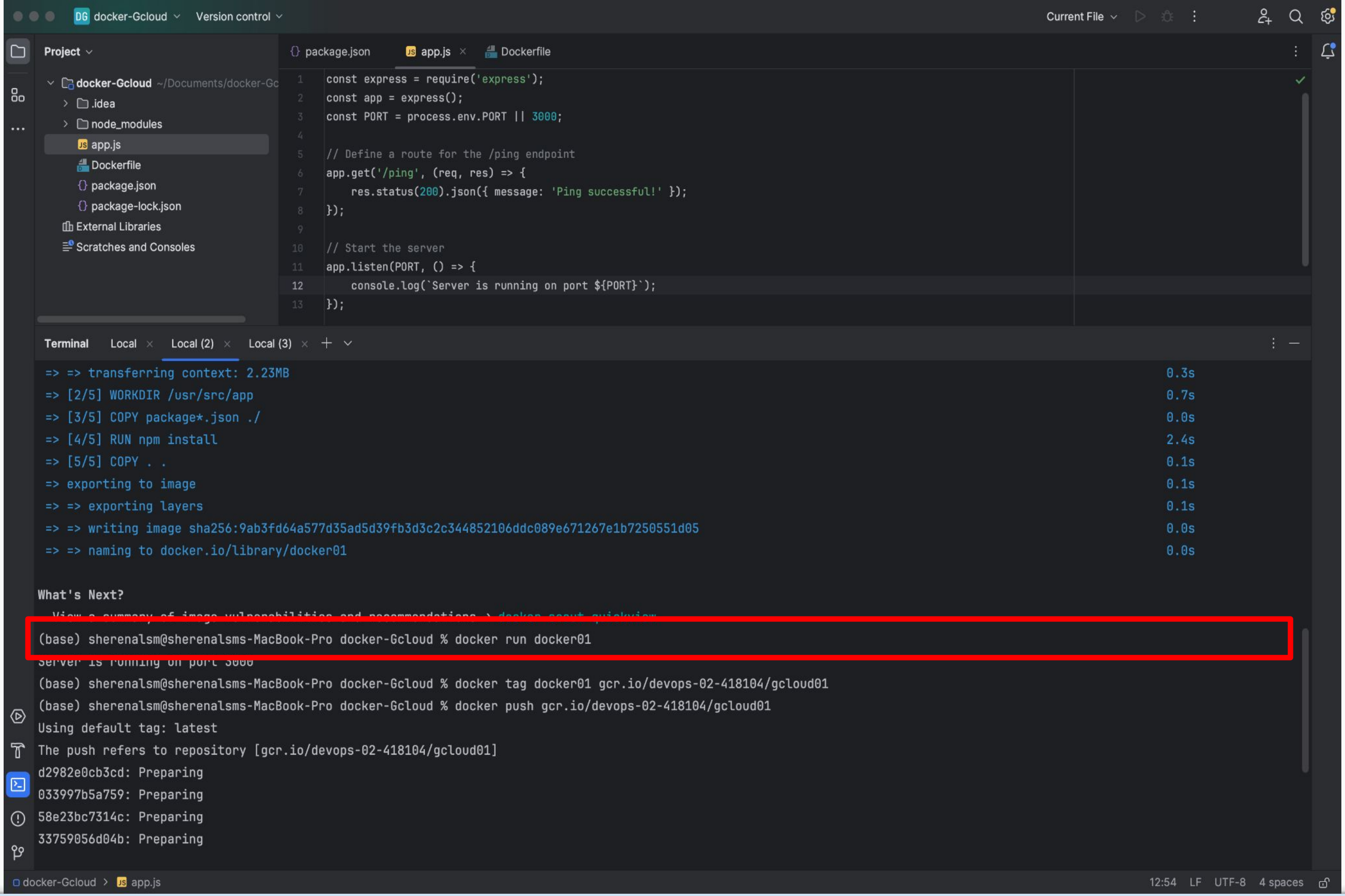
**Limitation of Docker :**

- Docker Desktop is not free for organizations with >250 employees or more than $10mil in annual revenue
- more suitable for medium-sized or small pet projects

## Test

Run your Dockerized Application Locally.

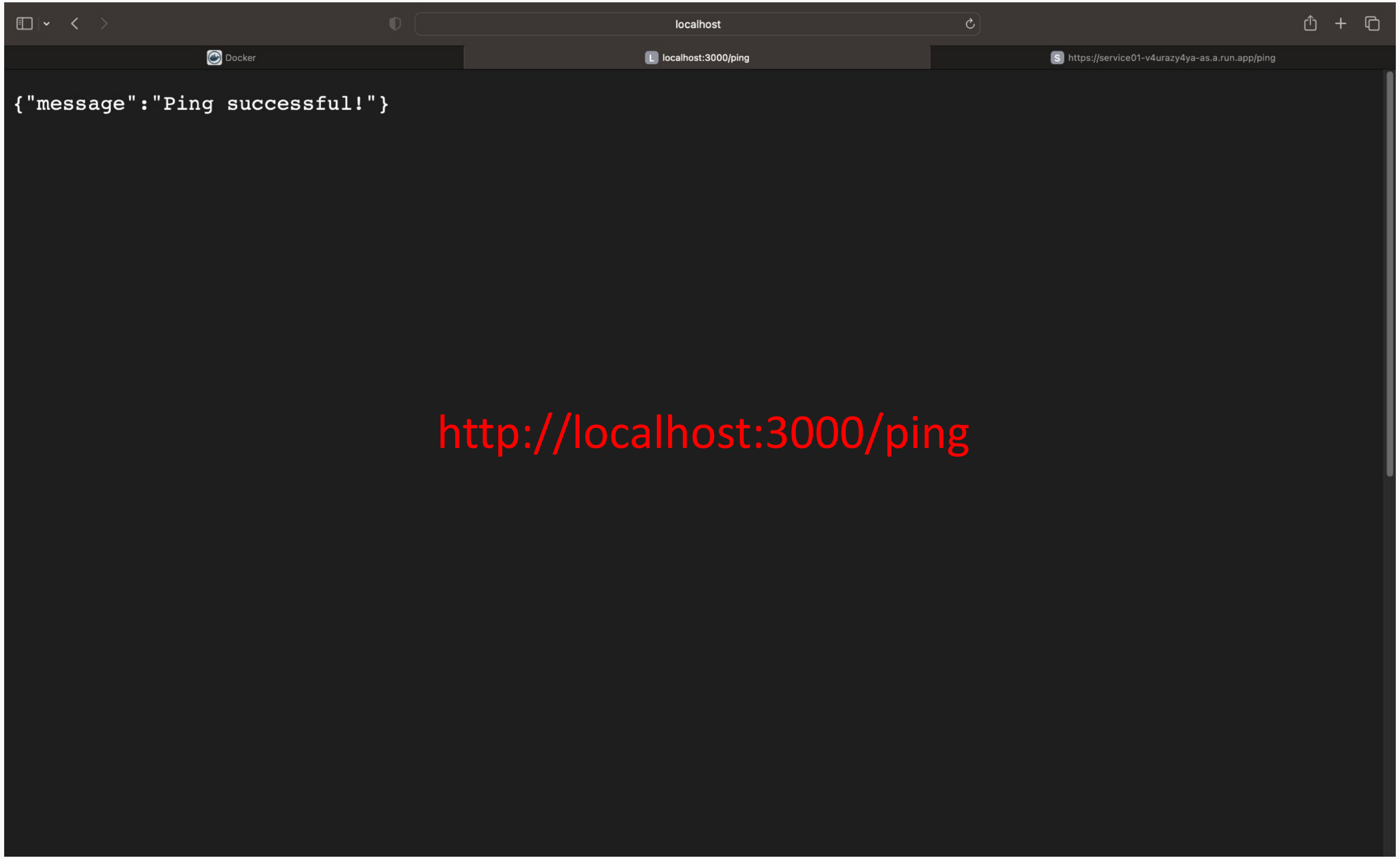
https://github.com/sherenaLIM



```javascript
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;

// Define a route for the /ping endpoint
app.get('/ping', (req, res) => {
    res.status(200).json({ message: 'Ping successful!' });
});

// Start the server
app.listen(PORT, () => {
    console.log(`Server is running on port ${PORT}`);
});
```

```
=> => transferring context: 2.23MB                                                               0.3s
=> [2/5] WORKDIR /usr/src/app                                                                    0.7s
=> [3/5] COPY package*.json ./                                                                   0.0s
=> [4/5] RUN npm install                                                                         2.4s
=> [5/5] COPY . .                                                                                0.1s
=> exporting to image                                                                            0.1s
=> => exporting layers                                                                           0.1s
=> => writing image sha256:9ab3fd64a577d35ad5d39fb3d3c2c344852106ddc089e671267e1b7250551d05      0.0s
=> => naming to docker.io/library/docker01                                                       0.0s

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker run docker01
Server is running on port 3000
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker tag docker01 gcr.io/devops-02-418104/gcloud01
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker push gcr.io/devops-02-418104/gcloud01
Using default tag: latest
The push refers to repository [gcr.io/devops-02-418104/gcloud01]
d2982e0cb3cd: Preparing
033997b5a759: Preparing
58e23bc7314c: Preparing
33759056d04b: Preparing
```

3a. Deploy docker image locally. Run it as a container using the command `docker run`.



```
{"message":"Ping successful!"}
```

http://localhost:3000/ping

3b. Access the application on a web browser or use tools like curl or Postman.

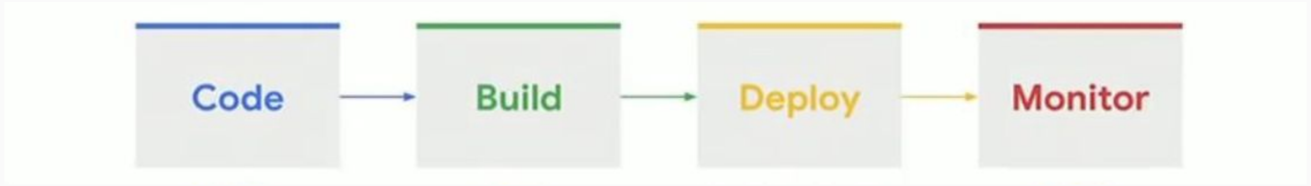
# Deploy

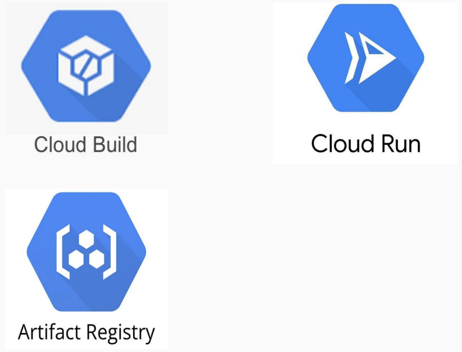
Deploy Docker Image to Google Cloud Run.

https://github.com/sherenaLIM

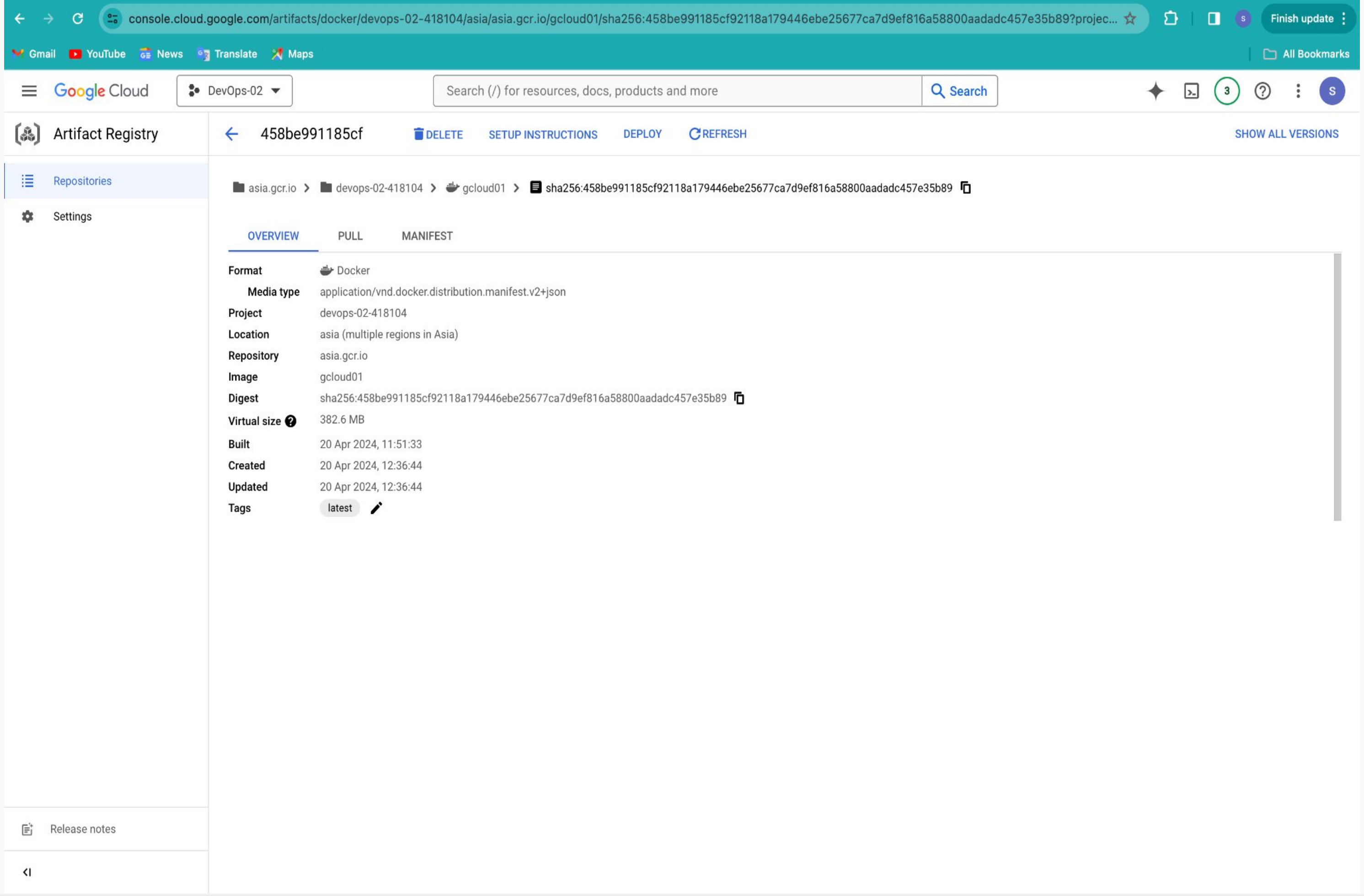
# 4: Set Up Required Google Cloud Services

Install Google Cloud CLI command-line tools to manage resources hosted on the cloud.





Google
Kubernetes Engine



console.cloud.google.com/artifacts/docker/devops-02-418104/asia/asia.gcr.io/gcloud01/sha256:458be991185cf92118a179446ebe25677ca7d9ef816a58800aadadc457e35b89?projec...

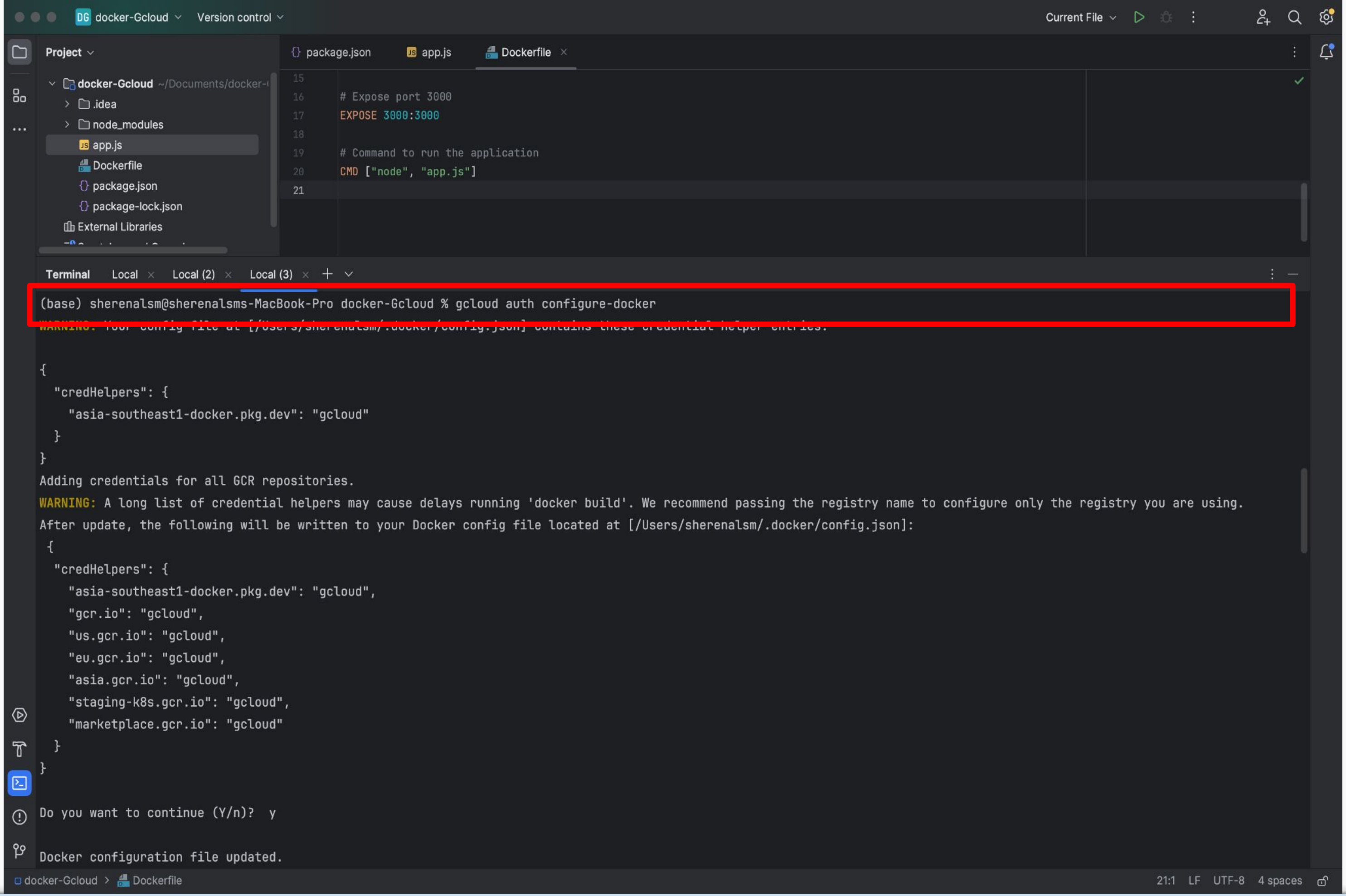
Artifact Registry

458be991185cf

asia.gcr.io > devops-02-418104 > gcloud01 > sha256:458be991185cf92118a179446ebe25677ca7d9ef816a58800aadadc457e35b89

| | |
|---|---|
| Format | Docker |
| Media type | application/vnd.docker.distribution.manifest.v2+json |
| Project | devops-02-418104 |
| Location | asia (multiple regions in Asia) |
| Repository | asia.gcr.io |
| Image | gcloud01 |
| Digest | sha256:458be991185cf92118a179446ebe25677ca7d9ef816a58800aadadc457e35b89 |
| Virtual size | 382.6 MB |
| Built | 20 Apr 2024, 11:51:33 |
| Created | 20 Apr 2024, 12:36:44 |
| Updated | 20 Apr 2024, 12:36:44 |
| Tags | latest |

4a. Set-up Google Artifact Repository.

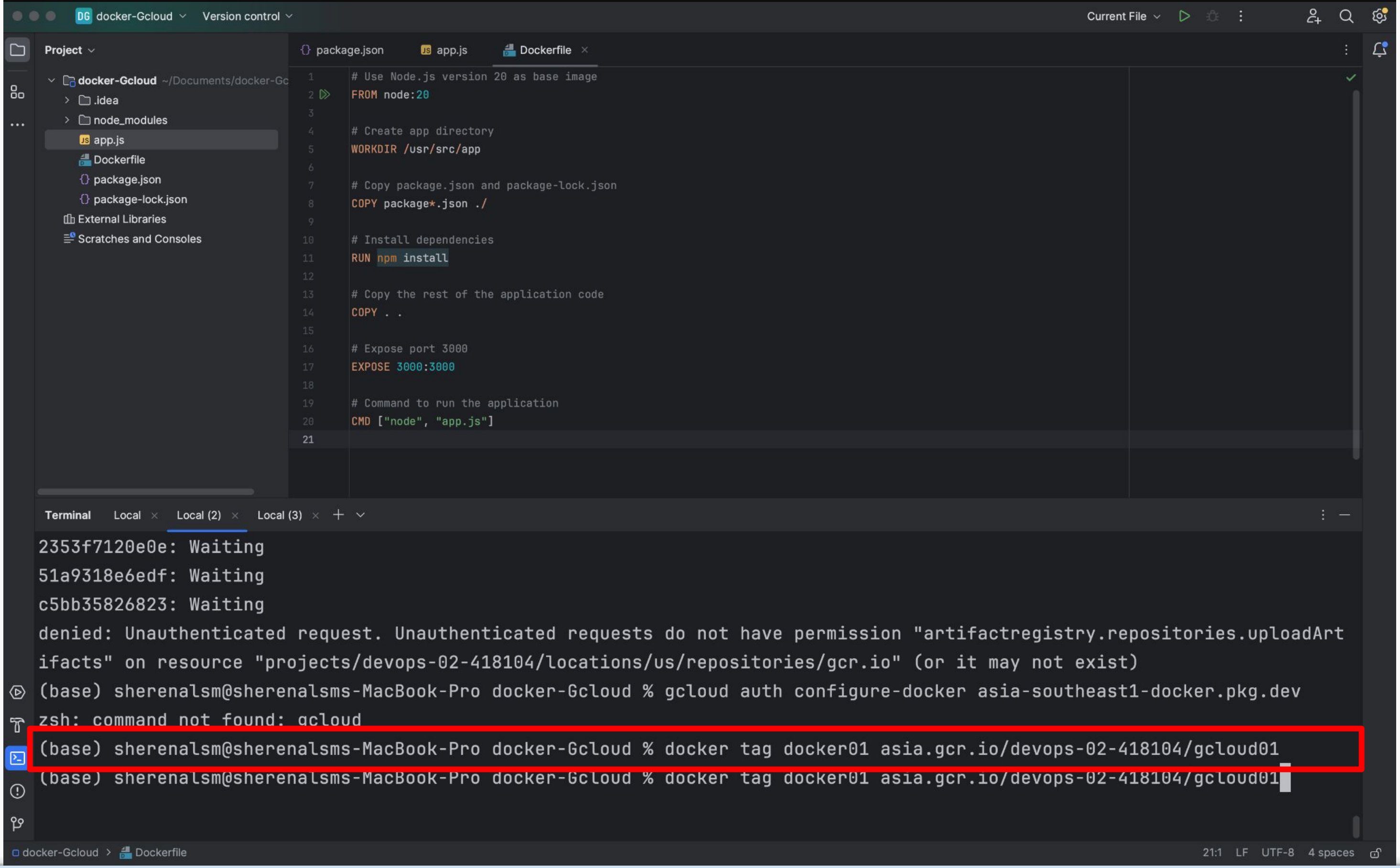


```
# Expose port 3000
EXPOSE 3000:3000

# Command to run the application
CMD ["node", "app.js"]
```

```
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % gcloud auth configure-docker
WARNING: Your config file at [/Users/sherenalsm/.docker/config.json] contains these credential helper entries:

{
  "credHelpers": {
    "asia-southeast1-docker.pkg.dev": "gcloud"
  }
}
Adding credentials for all GCR repositories.
WARNING: A long list of credential helpers may cause delays running 'docker build'. We recommend passing the registry name to configure only the registry you are using.
After update, the following will be written to your Docker config file located at [/Users/sherenalsm/.docker/config.json]:
 {
  "credHelpers": {
    "asia-southeast1-docker.pkg.dev": "gcloud",
    "gcr.io": "gcloud",
    "us.gcr.io": "gcloud",
    "eu.gcr.io": "gcloud",
    "asia.gcr.io": "gcloud",
    "staging-k8s.gcr.io": "gcloud",
    "marketplace.gcr.io": "gcloud"
  }
}

Do you want to continue (Y/n)?  y

Docker configuration file updated.
```

4b. Configure Docker to authenticate with Google Cloud Registry.

```
docker tag myimage gcr.io/myproject/myimage:latest
```

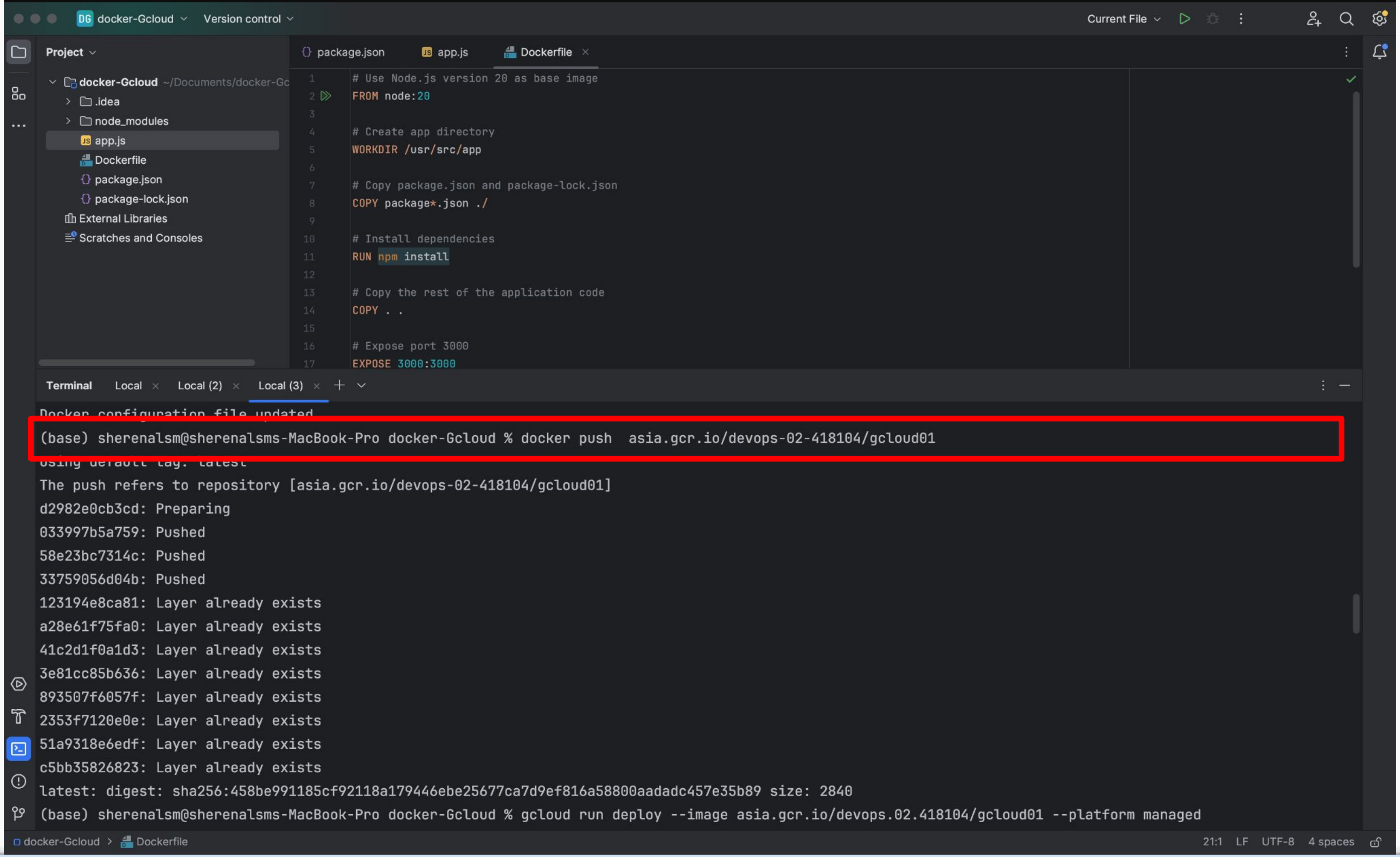


```dockerfile
# Use Node.js version 20 as base image
FROM node:20

# Create app directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .

# Expose port 3000
EXPOSE 3000:3000

# Command to run the application
CMD ["node", "app.js"]

```

```
2353f7120e0e: Waiting
51a9318e6edf: Waiting
c5bb35826823: Waiting
denied: Unauthenticated request. Unauthenticated requests do not have permission "artifactregistry.repositories.uploadArt
ifacts" on resource "projects/devops-02-418104/locations/us/repositories/gcr.io" (or it may not exist)
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % gcloud auth configure-docker asia-southeast1-docker.pkg.dev
zsh: command not found: gcloud
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker tag docker01 asia.gcr.io/devops-02-418104/gcloud01
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker tag docker01 asia.gcr.io/devops-02-418104/gcloud01
```

4c. Tag docker image (docker01) with the GCR repository name (devops-02-418104) and tag (gcloud01).

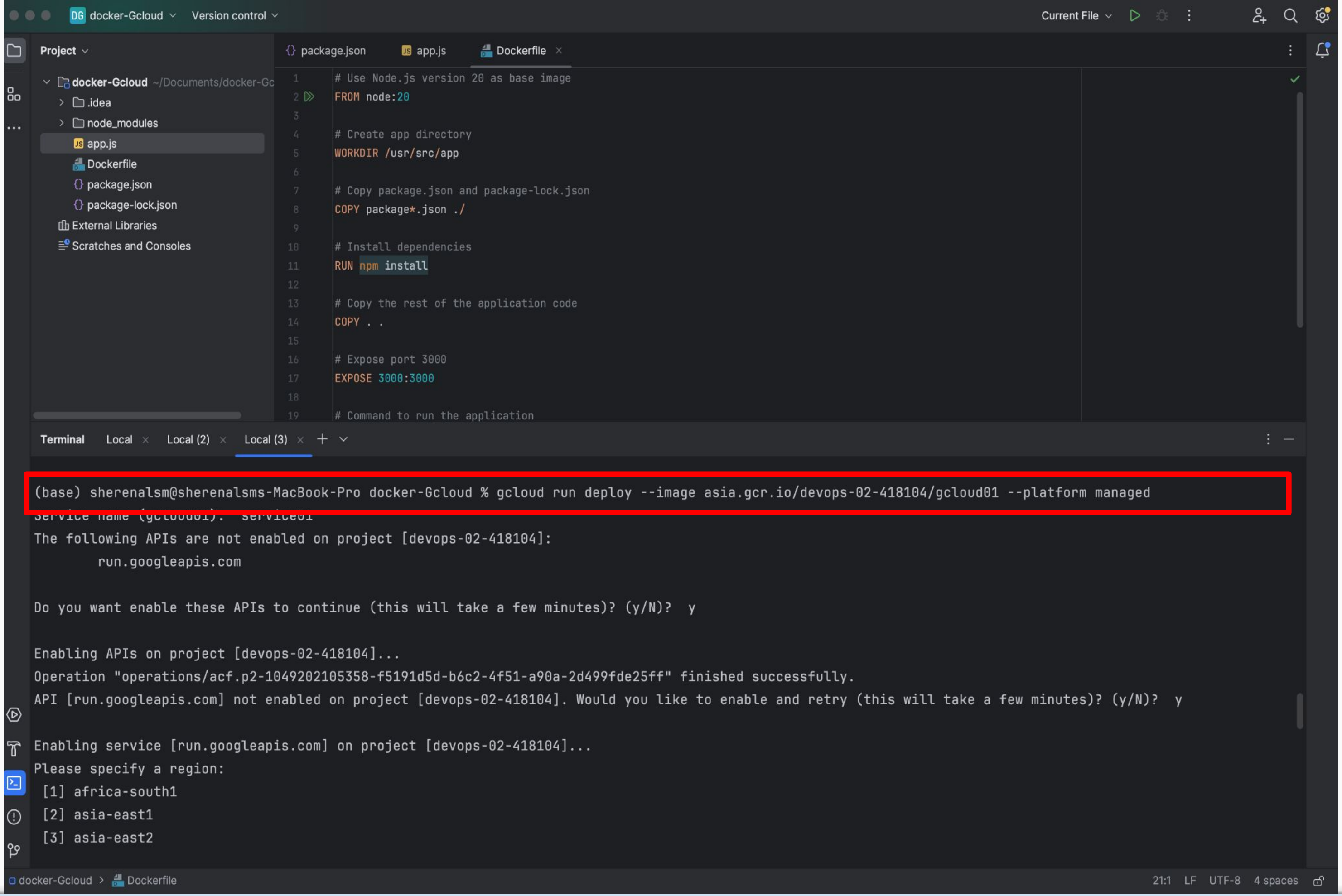
```
docker push gcr.io/myproject/myimage:latest
```



```
# Use Node.js version 20 as base image
FROM node:20

# Create app directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .

# Expose port 3000
EXPOSE 3000:3000
```

```
Docker configuration file updated.
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % docker push  asia.gcr.io/devops-02-418104/gcloud01
Using default tag: latest
The push refers to repository [asia.gcr.io/devops-02-418104/gcloud01]
d2982e0cb3cd: Preparing
033997b5a759: Pushed
58e23bc7314c: Pushed
33759056d04b: Pushed
123194e8ca81: Layer already exists
a28e61f75fa0: Layer already exists
41c2d1f0a1d3: Layer already exists
3e81cc85b636: Layer already exists
893507f6057f: Layer already exists
2353f7120e0e: Layer already exists
51a9318e6edf: Layer already exists
c5bb35826823: Layer already exists
latest: digest: sha256:458be991185cf92118a179446ebe25677ca7d9ef816a58800aadadc457e35b89 size: 2840
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % gcloud run deploy --image asia.gcr.io/devops.02.418104/gcloud01 --platform managed
```

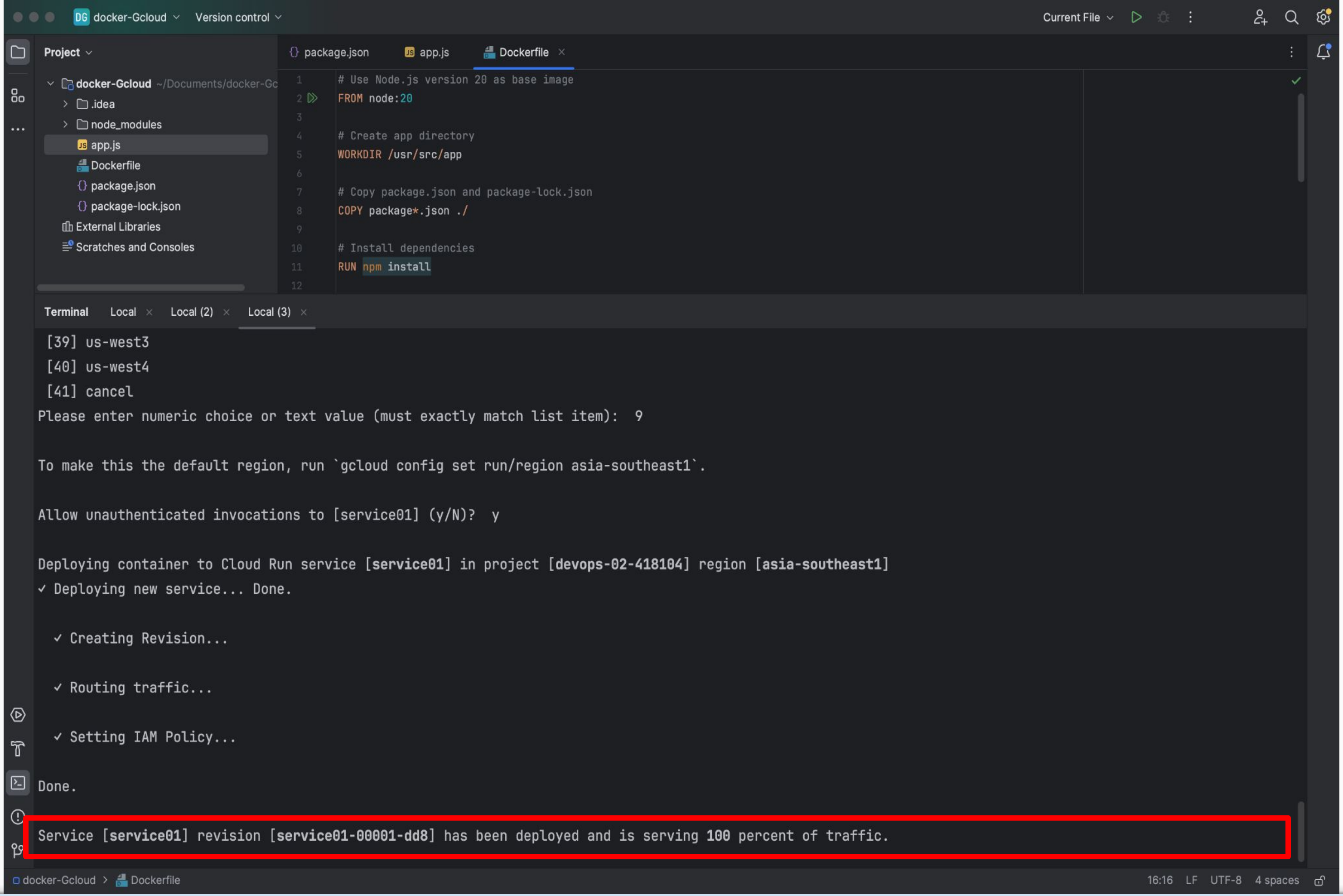
4d. Push the tagged docker image to GCR.



```dockerfile
# Use Node.js version 20 as base image
FROM node:20

# Create app directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .

# Expose port 3000
EXPOSE 3000:3000

# Command to run the application
```

```
(base) sherenalsm@sherenalsms-MacBook-Pro docker-Gcloud % gcloud run deploy --image asia.gcr.io/devops-02-418104/gcloud01 --platform managed
Service name (gcloud01):  service01
The following APIs are not enabled on project [devops-02-418104]:
        run.googleapis.com

Do you want enable these APIs to continue (this will take a few minutes)? (y/N)?  y

Enabling APIs on project [devops-02-418104]...
Operation "operations/acf.p2-1049202105358-f5191d5d-b6c2-4f51-a90a-2d499fde25ff" finished successfully.
API [run.googleapis.com] not enabled on project [devops-02-418104]. Would you like to enable and retry (this will take a few minutes)? (y/N)?  y

Enabling service [run.googleapis.com] on project [devops-02-418104]...
Please specify a region:
 [1] africa-south1
 [2] asia-east1
 [3] asia-east2
```

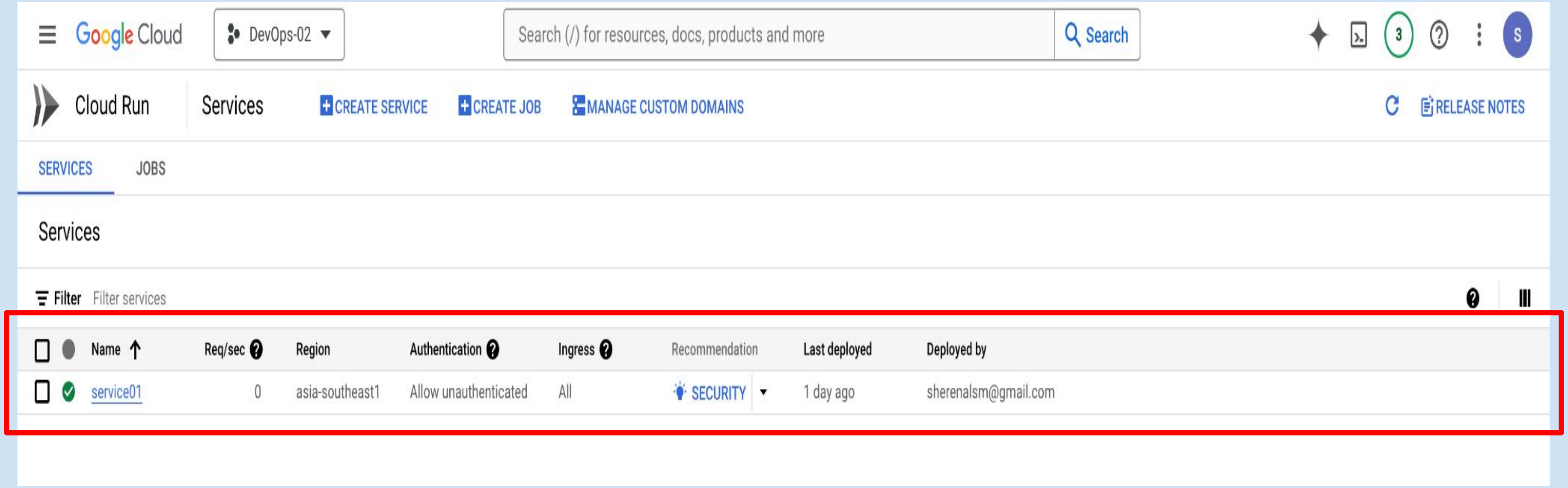
4e. Deploy docker image to Cloud Run (Part 1).



```
# Use Node.js version 20 as base image
FROM node:20

# Create app directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

```

```
 [39] us-west3
 [40] us-west4
 [41] cancel
Please enter numeric choice or text value (must exactly match list item):  9

To make this the default region, run `gcloud config set run/region asia-southeast1`.

Allow unauthenticated invocations to [service01] (y/N)?  y

Deploying container to Cloud Run service [service01] in project [devops-02-418104] region [asia-southeast1]
✓ Deploying new service... Done.

  ✓ Creating Revision...

  ✓ Routing traffic...

  ✓ Setting IAM Policy...

Done.

Service [service01] revision [service01-00001-dd8] has been deployed and is serving 100 percent of traffic.
```

4e. Container deployed successfully to Cloud Run (Part 2).



Google Cloud DevOps-02 Search (/) for resources, docs, products and more Search

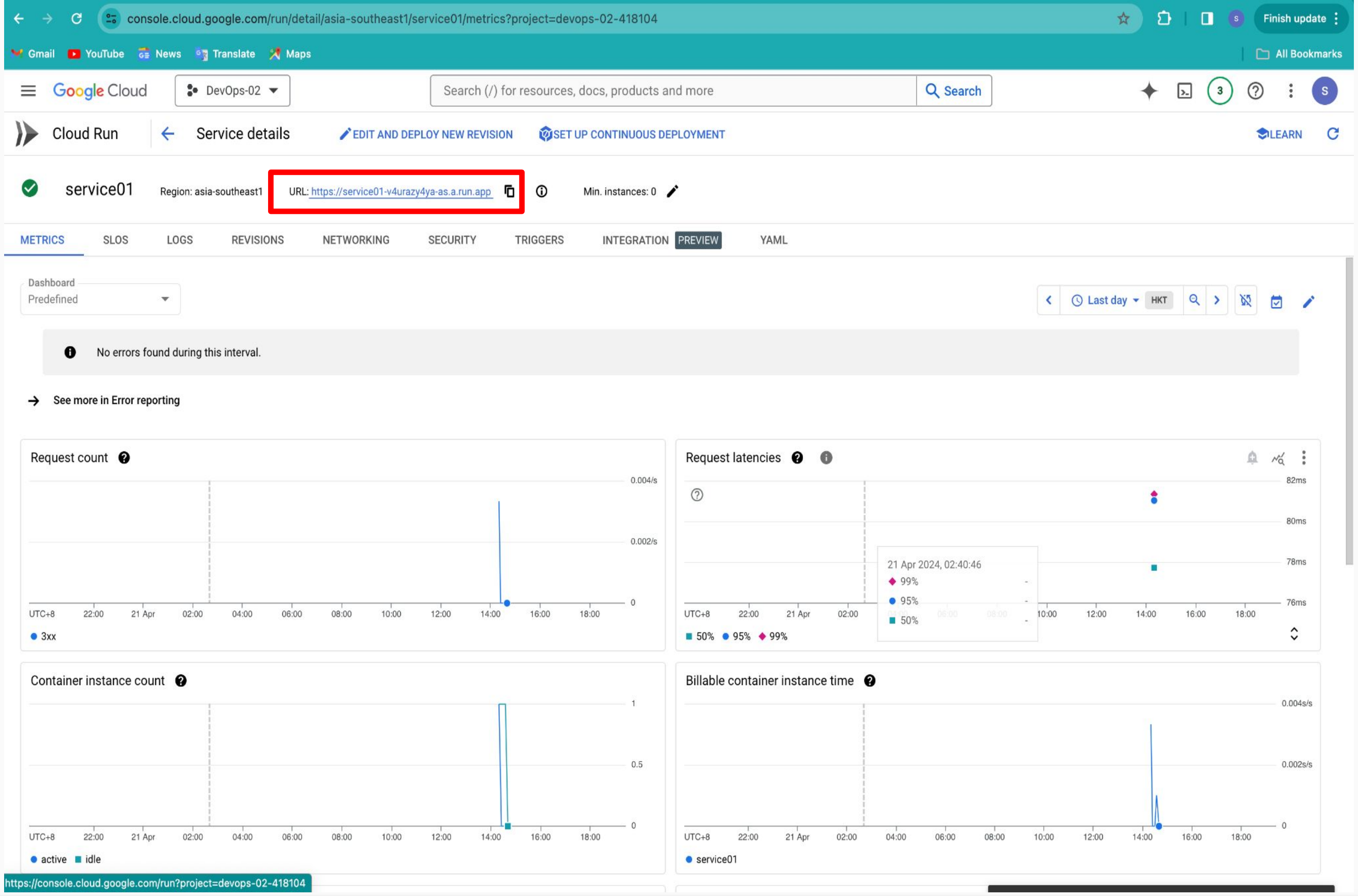
Cloud Run Services CREATE SERVICE CREATE JOB MANAGE CUSTOM DOMAINS RELEASE NOTES

SERVICES JOBS

Services

Filter Filter services

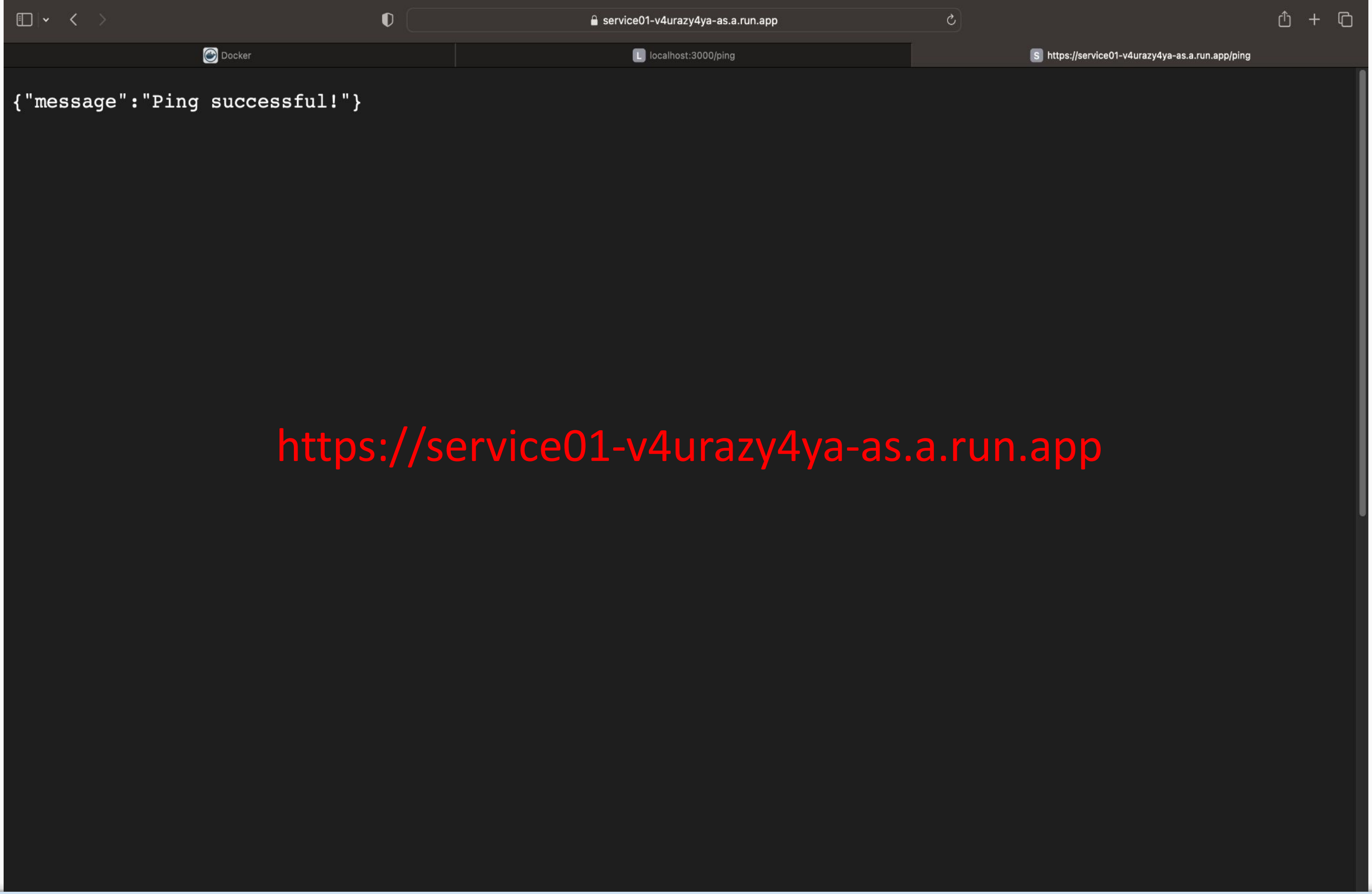
| | | Name ↑ | Req/sec | Region | Authentication | Ingress | Recommendation | Last deployed | Deployed by |
|---|---|---|---|---|---|---|---|---|---|
| ☐ | ✓ | service01 | 0 | asia-southeast1 | Allow unauthenticated | All | SECURITY | 1 day ago | sherenalsm@gmail.com |


console.cloud.google.com/run/detail/asia-southeast1/service01/metrics?project=devops-02-418104
Google Cloud
DevOps-02
Search (/) for resources, docs, products and more
Cloud Run
Service details
EDIT AND DEPLOY NEW REVISION
SET UP CONTINUOUS DEPLOYMENT
service01
Region: asia-southeast1
URL: https://service01-v4urazy4ya-as.a.run.app
Min. instances: 0
METRICS
SLOS
LOGS
REVISIONS
NETWORKING
SECURITY
TRIGGERS
INTEGRATION PREVIEW
YAML
Dashboard Predefined
Last day HKT
Request count
Request latencies
21 Apr 2024, 02:40:46
Container instance count
Billable container instance time

No errors found during this interval.

See more in Error reporting

4f. Cloud Run service runs instance of Docker image pushed into Google Artifact Registry. Copy unique URL provided.


service01-v4urazy4ya-as.a.run.app
Docker
localhost:3000/ping
https://service01-v4urazy4ya-as.a.run.app/ping

```
{"message":"Ping successful!"}
```

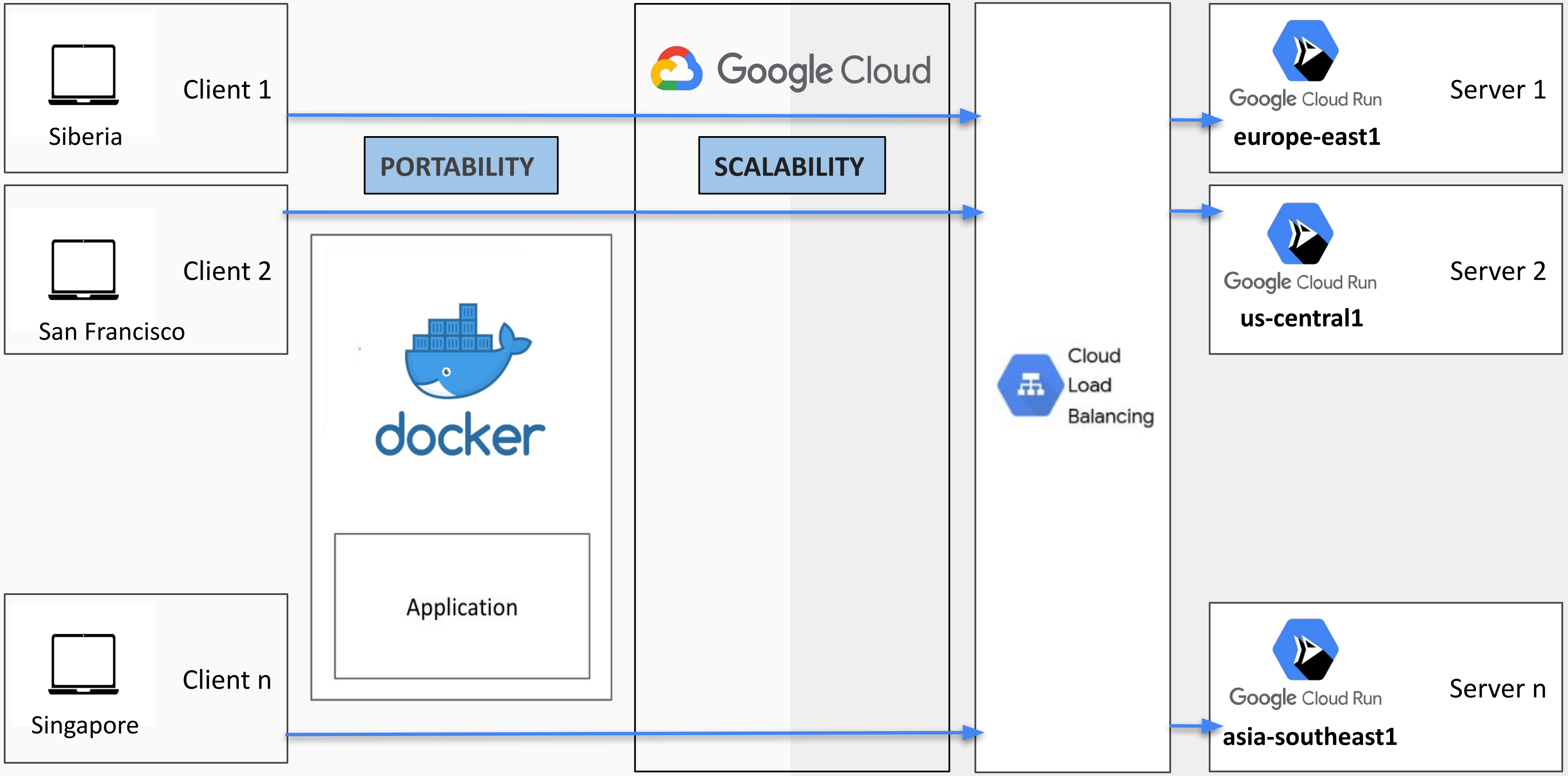
https://service01-v4urazy4ya-as.a.run.app

4f. Apply unique URL that serves as the endpoint for accessing service01 over the internet.

# Software Architecture

Client-Server Architecture Diagram.

https://github.com/sherenaLIM

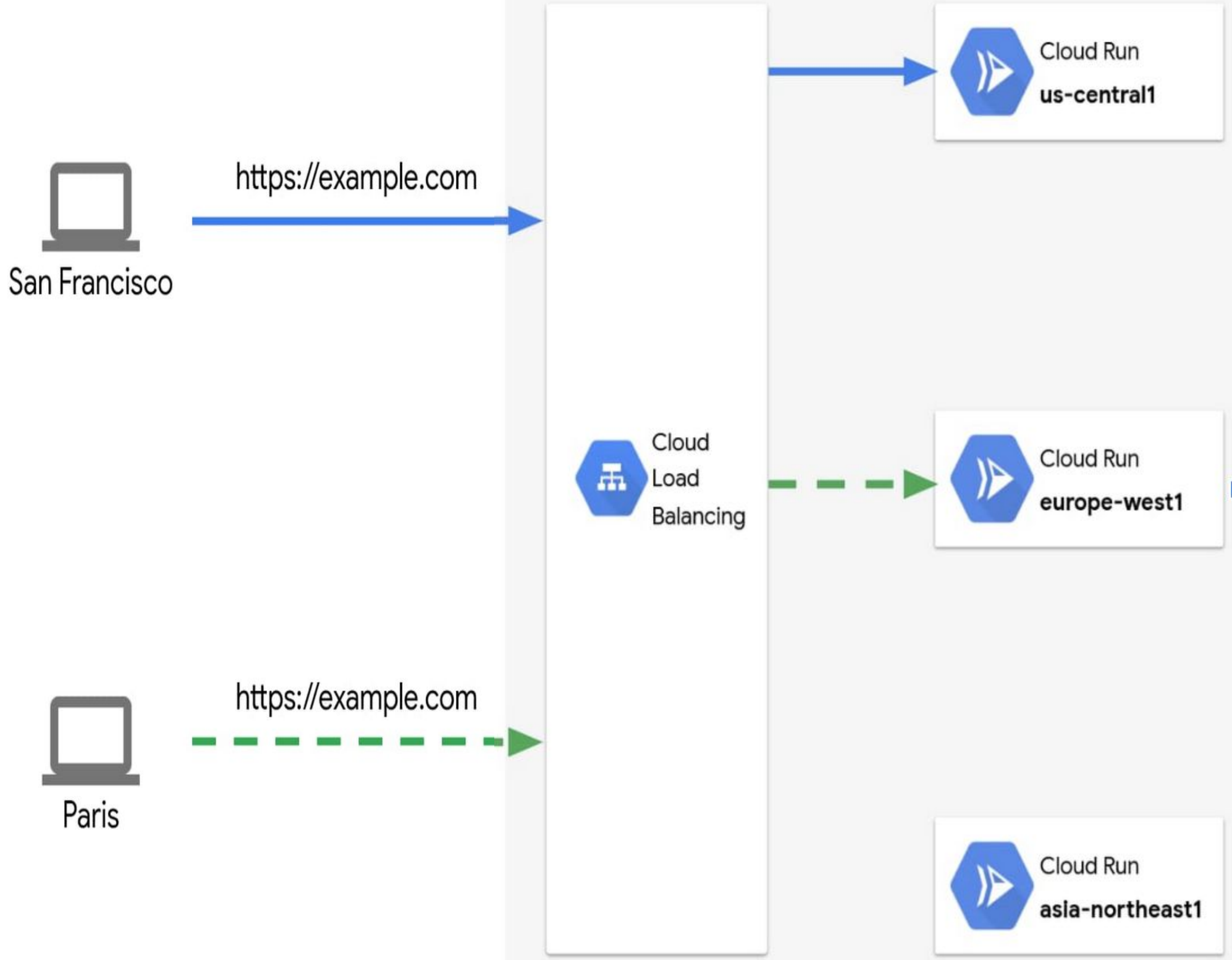


Deployment of Containerized Monolithic Application on Google Cloud.

## Optimal Resource Utilization

A load balancer distributes incoming network traffic across multiple servers or instances.

https://github.com/sherenaLIM



**Load Balancing**
mitigates
**Denial-of-Service (DoS) Attacks**