Name: Patel Shivam S.
Enroll no: 21162101019
Sem: 5 Batch: 51 Branch: CBA

Sub: Microservices

Practical 12

AIM: Create an image that sandboxes a small Flask application. The goal of this exercise is to create a Docker image which will run a Flask app.

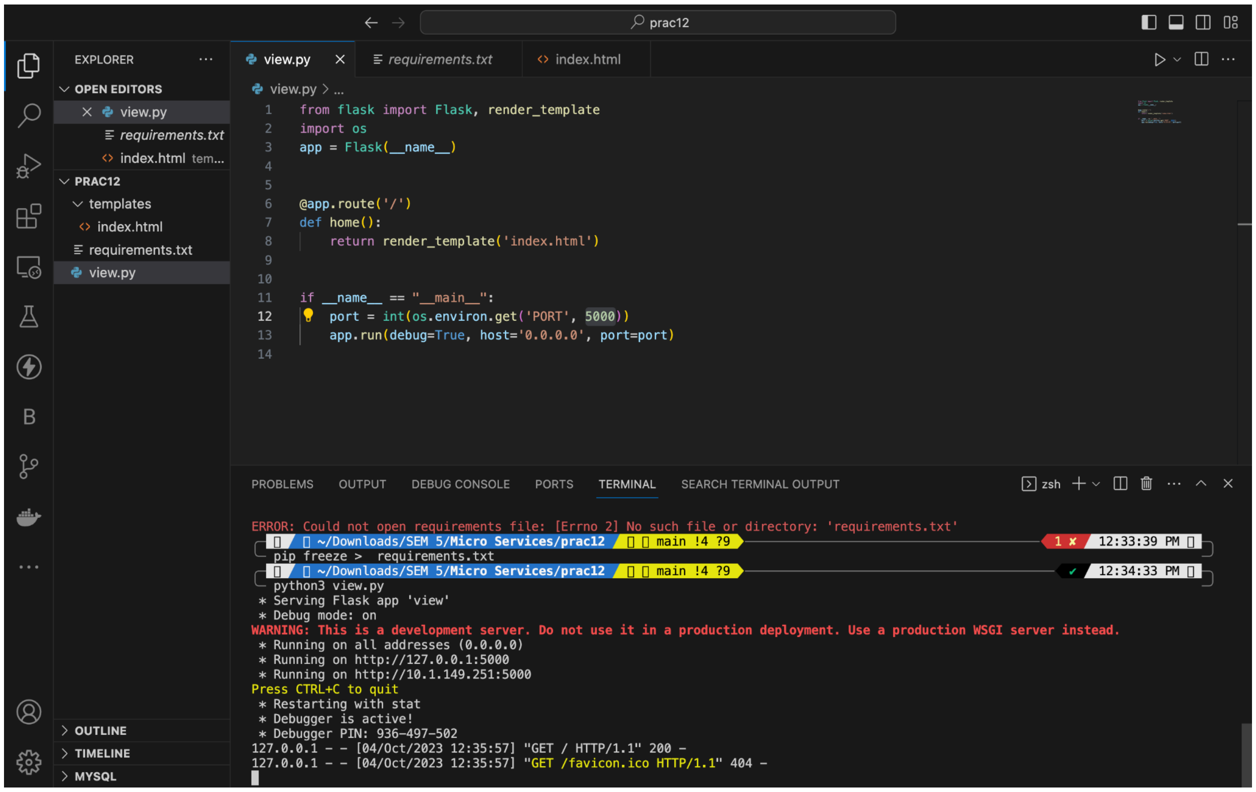
Docker, Inc. sponsors a dedicated team that is responsible for reviewing and publishing all Official Repositories content. This team works in collaboration with upstream software maintainers, security experts, and the broader Docker community. These are not prefixed by an organization or user name. In the list of images above, the python, node, alpine and nginx images are official (base) images.

User images are images created and shared by users like you. They build on base images and add additional functionality. Typically these are formatted as user/image-name. The user value in the image name is your Docker Store user or organization name. Hence,

1. Create a Python Flask app that displays random data.
2. Write a Dockerfile.
3. Build the image.
4. Run your image.
5. Push your image

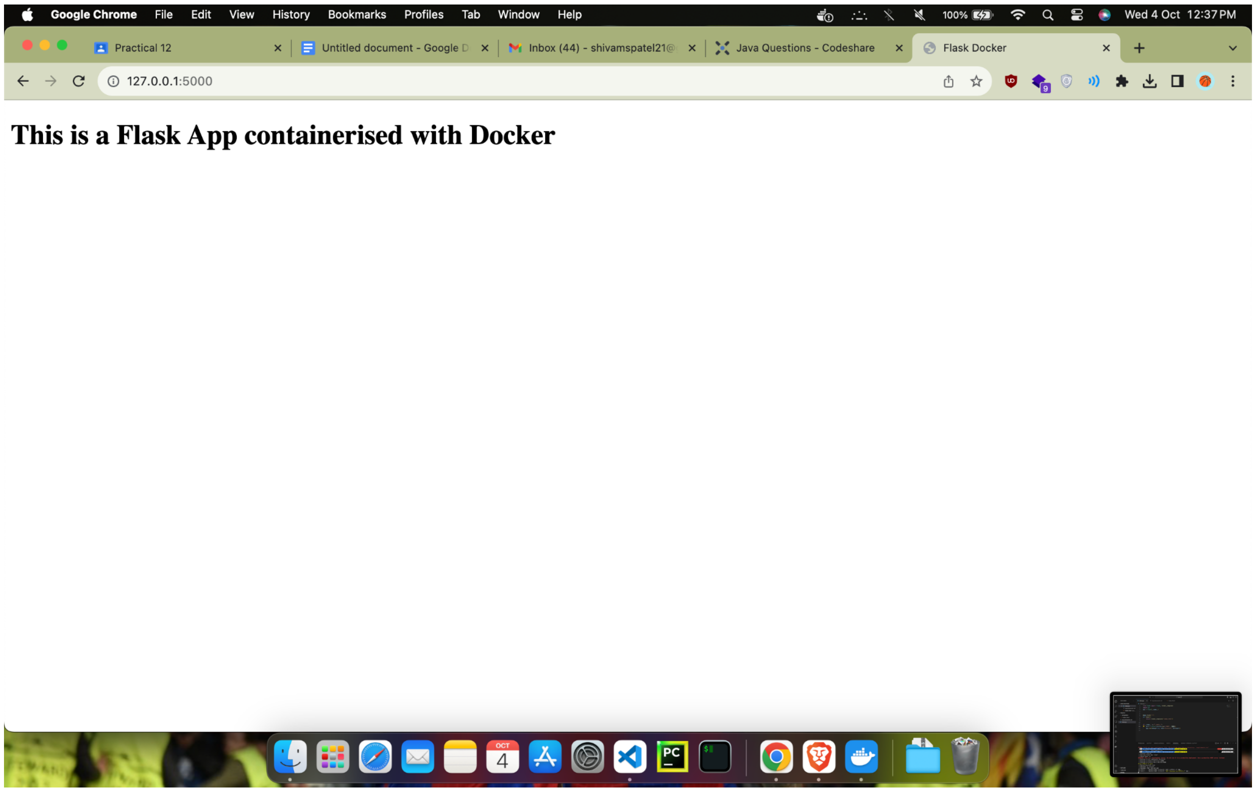
GITHUB LINK:
https://github.com/Shivam3783/microservice_practicals/tree/main/prac12

Practical 12.1: Create a Python Flask app that displays random data.



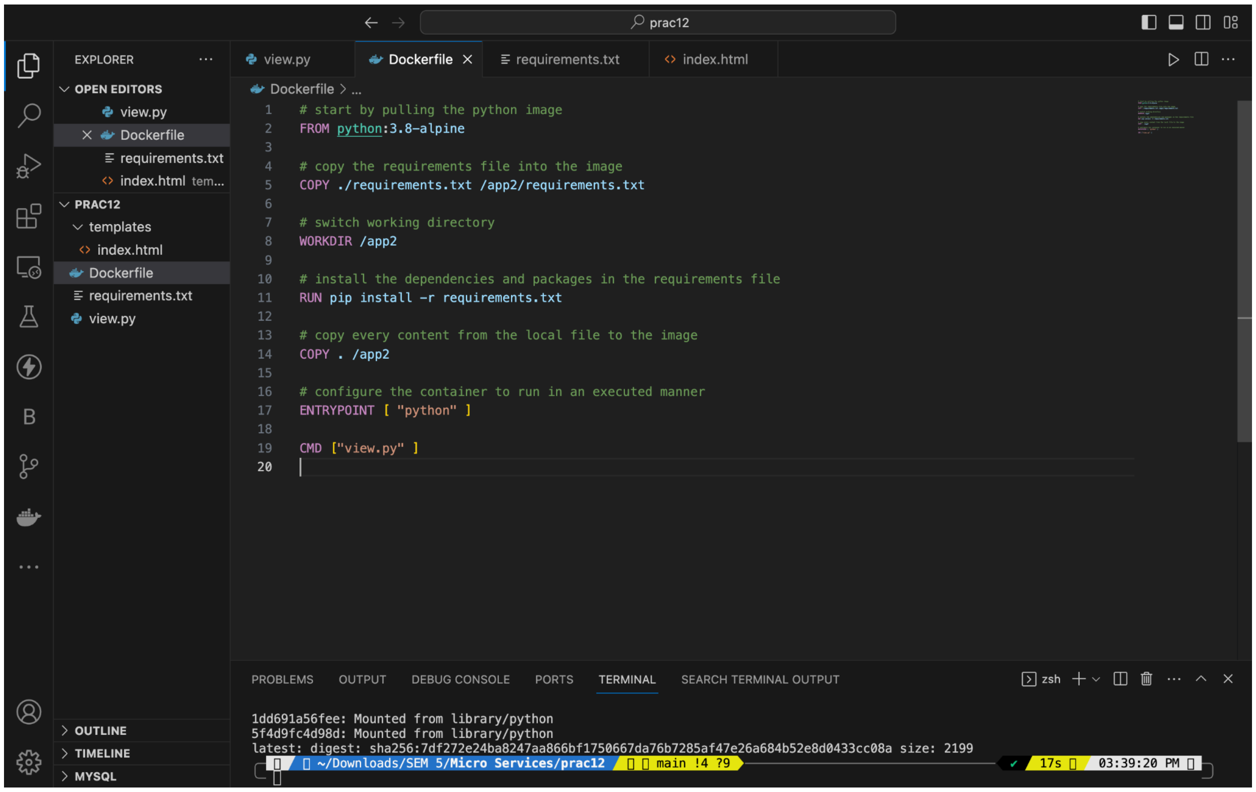
```python
from flask import Flask, render_template
import os
app = Flask(__name__)


@app.route('/')
def home():
    return render_template('index.html')


if __name__ == "__main__":
    port = int(os.environ.get('PORT', 5000))
    app.run(debug=True, host='0.0.0.0', port=port)
```

```
ERROR: Could not open requirements file: [Errno 2] No such file or directory: 'requirements.txt'
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9                      1 ✘  12:33:39 PM
 pip freeze >  requirements.txt
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9                        ✔  12:34:33 PM
 python3 view.py
 * Serving Flask app 'view'
 * Debug mode: on
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://127.0.0.1:5000
 * Running on http://10.1.149.251:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 936-497-502
127.0.0.1 - - [04/Oct/2023 12:35:57] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [04/Oct/2023 12:35:57] "GET /favicon.ico HTTP/1.1" 404 -
```



This is a Flask App containerised with Docker

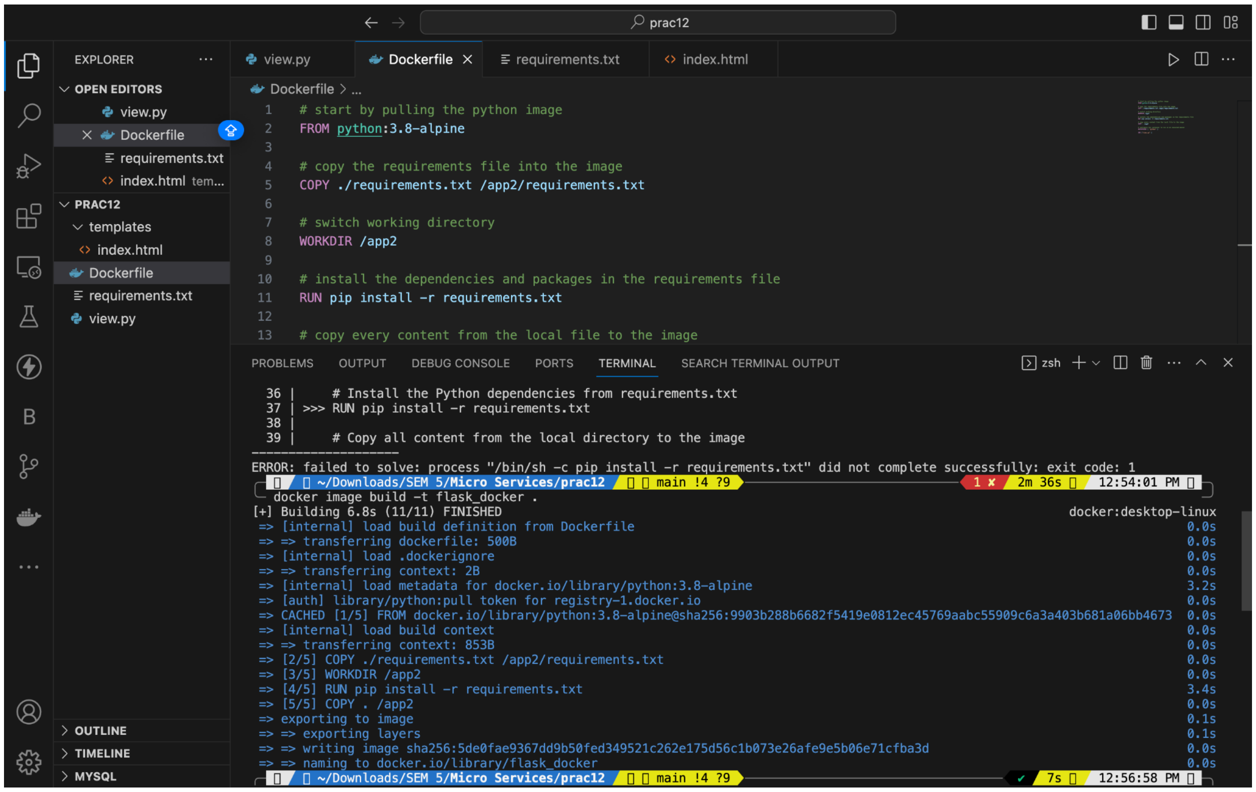
Practical 12.2. Write a Dockerfile.



```
# start by pulling the python image
FROM python:3.8-alpine

# copy the requirements file into the image
COPY ./requirements.txt /app2/requirements.txt

# switch working directory
WORKDIR /app2

# install the dependencies and packages in the requirements file
RUN pip install -r requirements.txt

# copy every content from the local file to the image
COPY . /app2

# configure the container to run in an executed manner
ENTRYPOINT [ "python" ]

CMD ["view.py" ]

```

```
1dd691a56fee: Mounted from library/python
5f4d9fc4d98d: Mounted from library/python
latest: digest: sha256:7df272e24ba8247aa866bf1750667da76b7285af47e26a684b52e8d0433cc08a size: 2199
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9                    ✓ 17s  03:39:20 PM
```

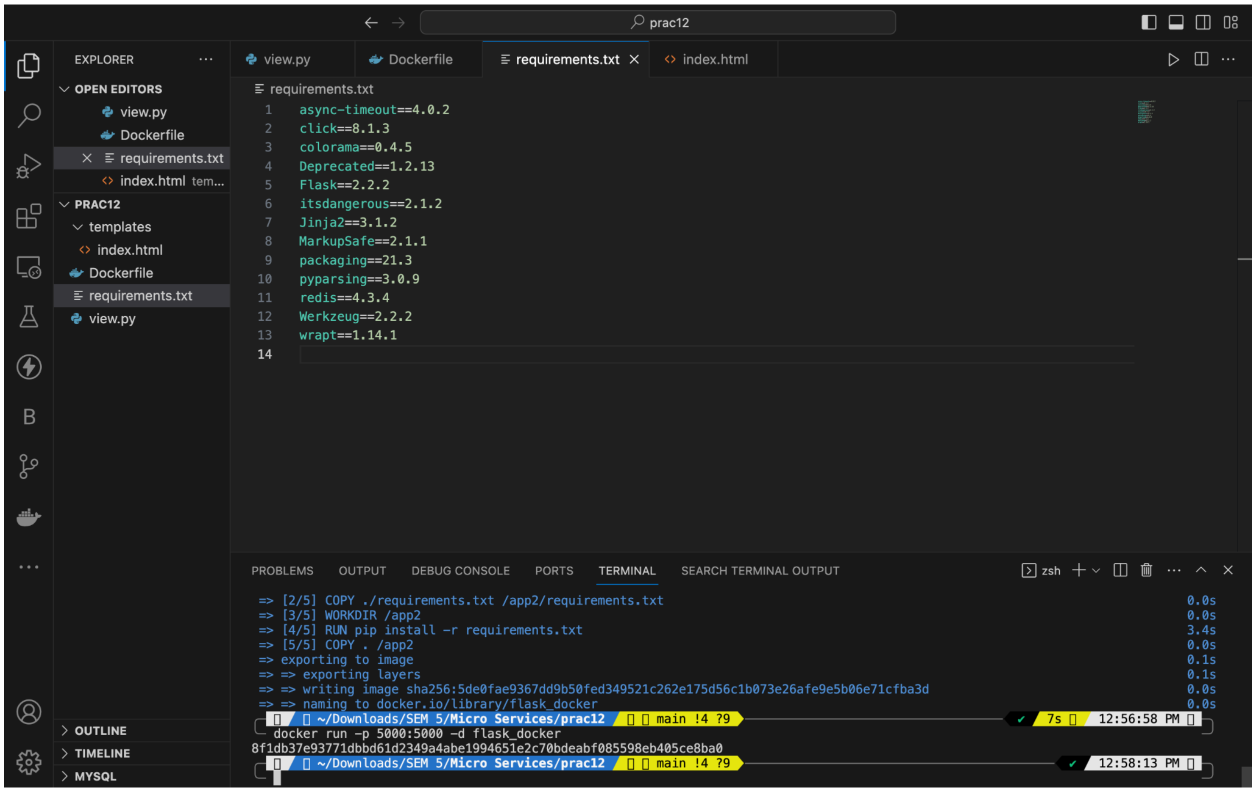
Practical 12.3. Build the image.



```
# start by pulling the python image
FROM python:3.8-alpine

# copy the requirements file into the image
COPY ./requirements.txt /app2/requirements.txt

# switch working directory
WORKDIR /app2

# install the dependencies and packages in the requirements file
RUN pip install -r requirements.txt

# copy every content from the local file to the image
```

```
  36 |     # Install the Python dependencies from requirements.txt
  37 | >>> RUN pip install -r requirements.txt
  38 |
  39 |     # Copy all content from the local directory to the image
--------------------
ERROR: failed to solve: process "/bin/sh -c pip install -r requirements.txt" did not complete successfully: exit code: 1
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9          1 ✘  2m 36s  12:54:01 PM
 docker image build -t flask_docker .
[+] Building 6.8s (11/11) FINISHED                                                       docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                    0.0s
 => => transferring dockerfile: 500B                                                                    0.0s
 => [internal] load .dockerignore                                                                       0.0s
 => => transferring context: 2B                                                                         0.0s
 => [internal] load metadata for docker.io/library/python:3.8-alpine                                    3.2s
 => [auth] library/python:pull token for registry-1.docker.io                                           0.0s
 => CACHED [1/5] FROM docker.io/library/python:3.8-alpine@sha256:9903b288b6682f5419e0812ec45769aabc55909c6a3a403b681a06bb4673  0.0s
 => [internal] load build context                                                                       0.0s
 => => transferring context: 853B                                                                       0.0s
 => [2/5] COPY ./requirements.txt /app2/requirements.txt                                                0.0s
 => [3/5] WORKDIR /app2                                                                                 0.0s
 => [4/5] RUN pip install -r requirements.txt                                                           3.4s
 => [5/5] COPY . /app2                                                                                  0.0s
 => exporting to image                                                                                  0.1s
 => => exporting layers                                                                                 0.1s
 => => writing image sha256:5de0fae9367dd9b50fed349521c262e175d56c1b073e26afe9e5b06e71cfba3d            0.0s
 => => naming to docker.io/library/flask_docker                                                         0.0s
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9                    ✓ 7s  12:56:58 PM
```

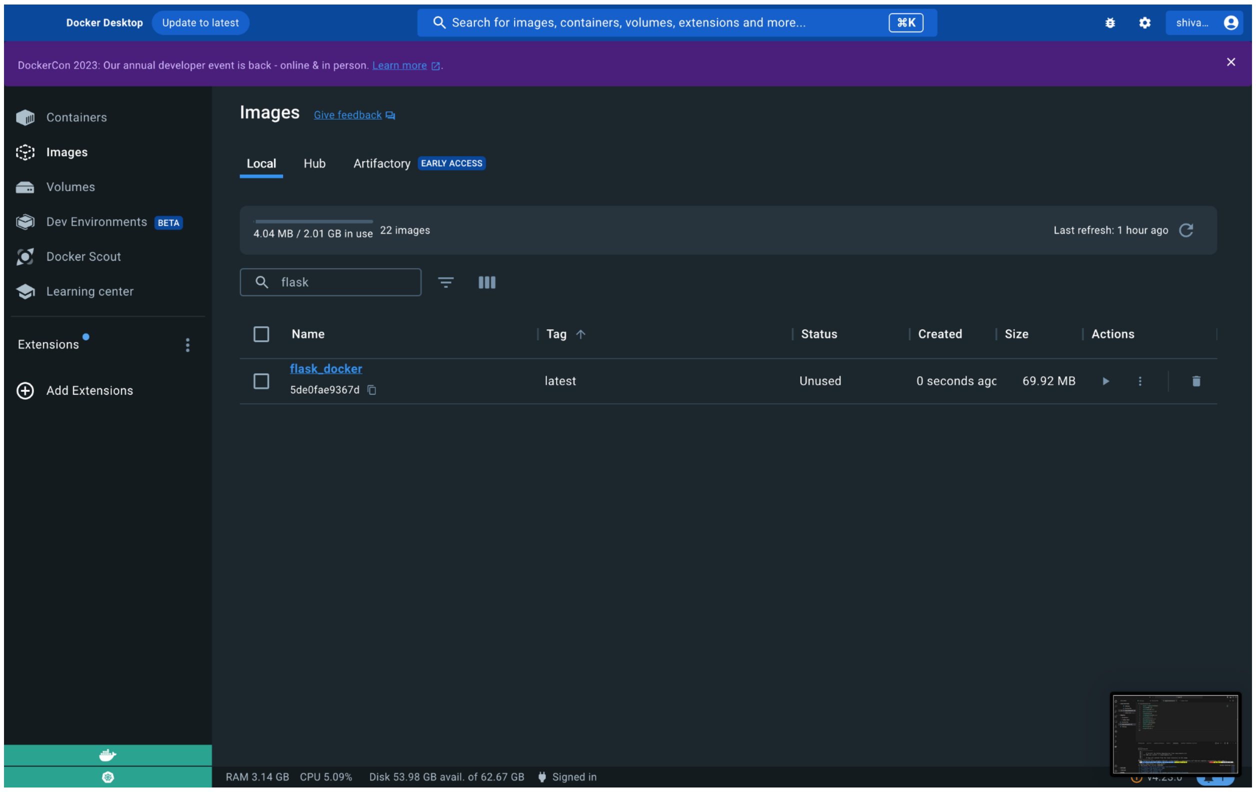
Practical 12.4. Run your image.



requirements.txt

```
async-timeout==4.0.2
click==8.1.3
colorama==0.4.5
Deprecated==1.2.13
Flask==2.2.2
itsdangerous==2.1.2
Jinja2==3.1.2
MarkupSafe==2.1.1
packaging==21.3
pyparsing==3.0.9
redis==4.3.4
Werkzeug==2.2.2
wrapt==1.14.1
```

```
 => [2/5] COPY ./requirements.txt /app2/requirements.txt                    0.0s
 => [3/5] WORKDIR /app2                                                      0.0s
 => [4/5] RUN pip install -r requirements.txt                                3.4s
 => [5/5] COPY . /app2                                                       0.0s
 => exporting to image                                                       0.1s
 => => exporting layers                                                      0.1s
 => => writing image sha256:5de0fae9367dd9b50fed349521c262e175d56c1b073e26afe9e5b06e71cfba3d  0.0s
 => => naming to docker.io/library/flask_docker                              0.0s
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9          ✓ 7s  12:56:58 PM
 docker run -p 5000:5000 -d flask_docker
8f1db37e93771dbbd61d2349a4abe1994651e2c70bdeabf085598eb405ce8ba0
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9               ✓  12:58:13 PM
```



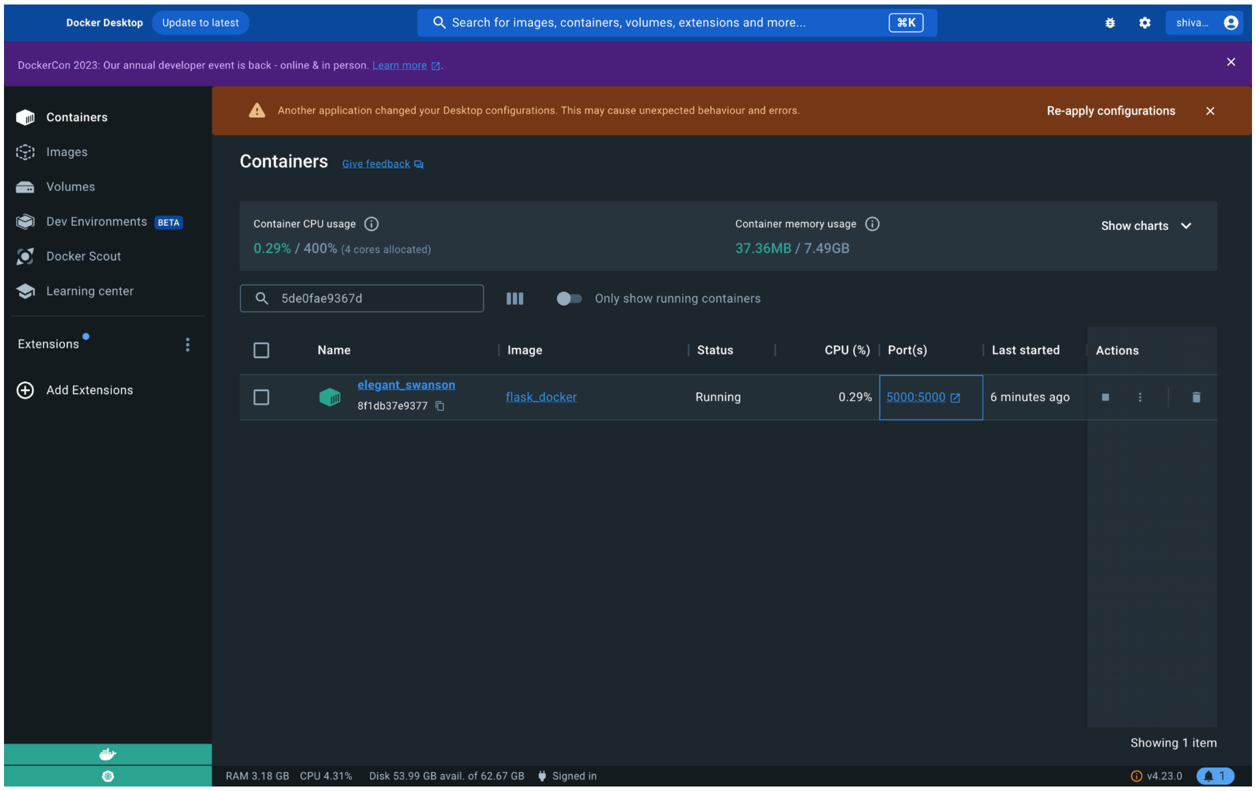
DockerCon 2023: Our annual developer event is back - online & in person. Learn more.

Images

4.04 MB / 2.01 GB in use   22 images   Last refresh: 1 hour ago

Name	Tag	Status	Created	Size	Actions
flask_docker 5de0fae9367d	latest	Unused	0 seconds ago	69.92 MB	

RAM 3.14 GB   CPU 5.09%   Disk 53.98 GB avail. of 62.67 GB   Signed in



DockerCon 2023: Our annual developer event is back - online & in person. Learn more.

Another application changed your Desktop configurations. This may cause unexpected behaviour and errors.

Containers

Container CPU usage 0.29% / 400% (4 cores allocated)

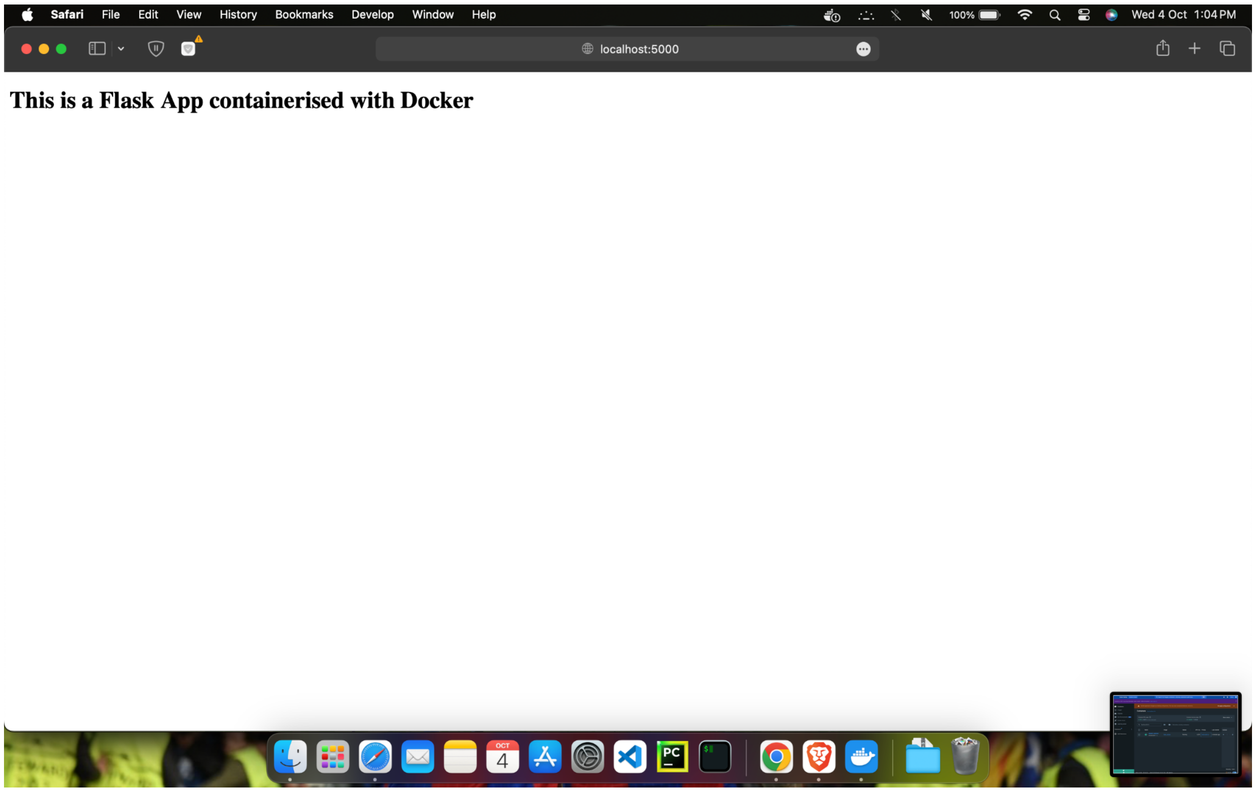
Container memory usage 37.36MB / 7.49GB

5de0fae9367d

Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
elegant_swanson 8f1db37e9377	flask_docker	Running	0.29%	5000:5000	6 minutes ago	

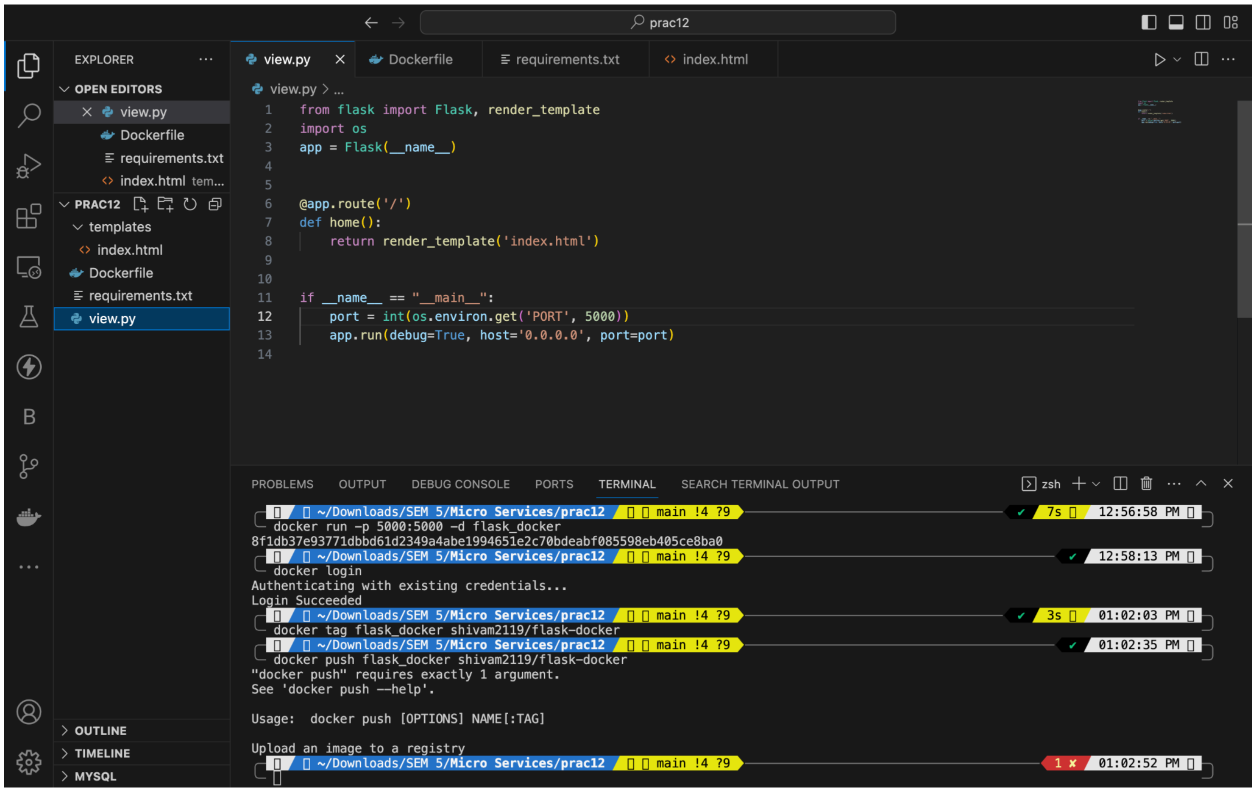
Showing 1 item

RAM 3.18 GB CPU 4.31% Disk 53.99 GB avail. of 62.67 GB Signed in v4.23.0



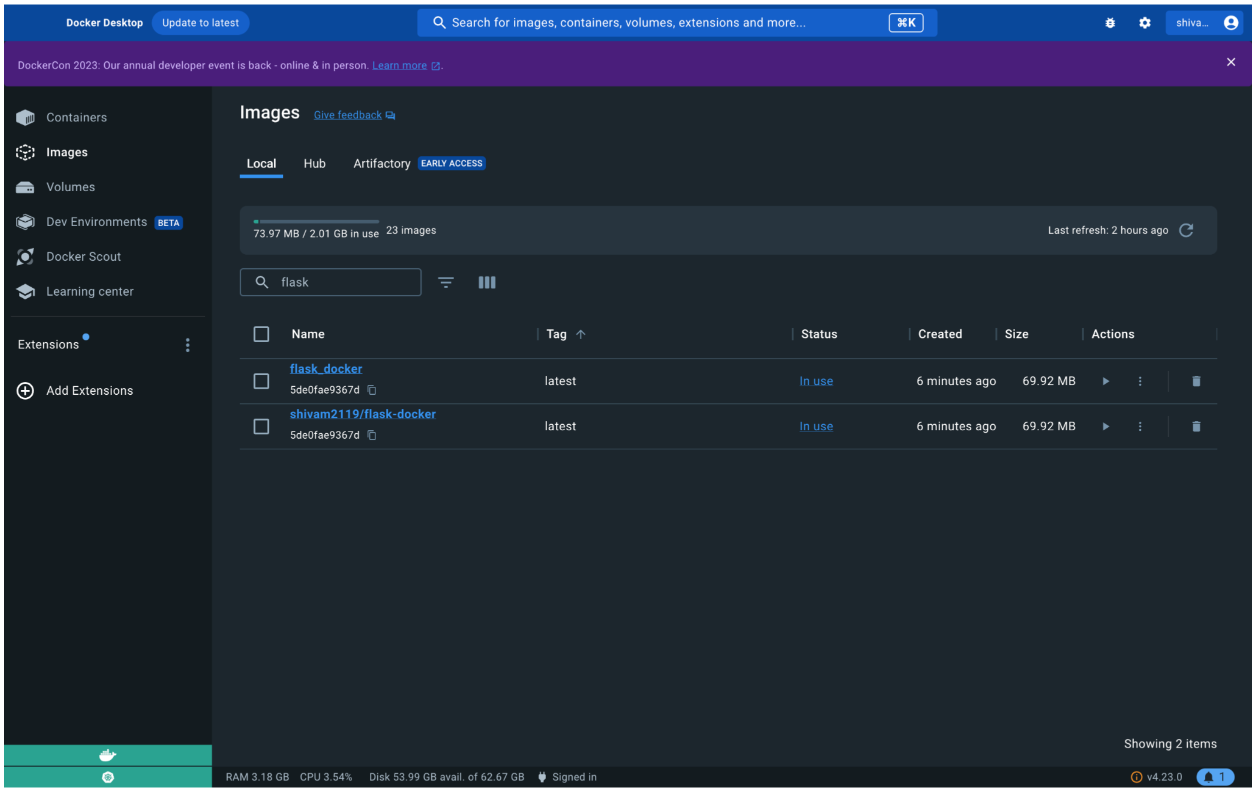
This is a Flask App containerised with Docker

Practical 12.5. Push your image



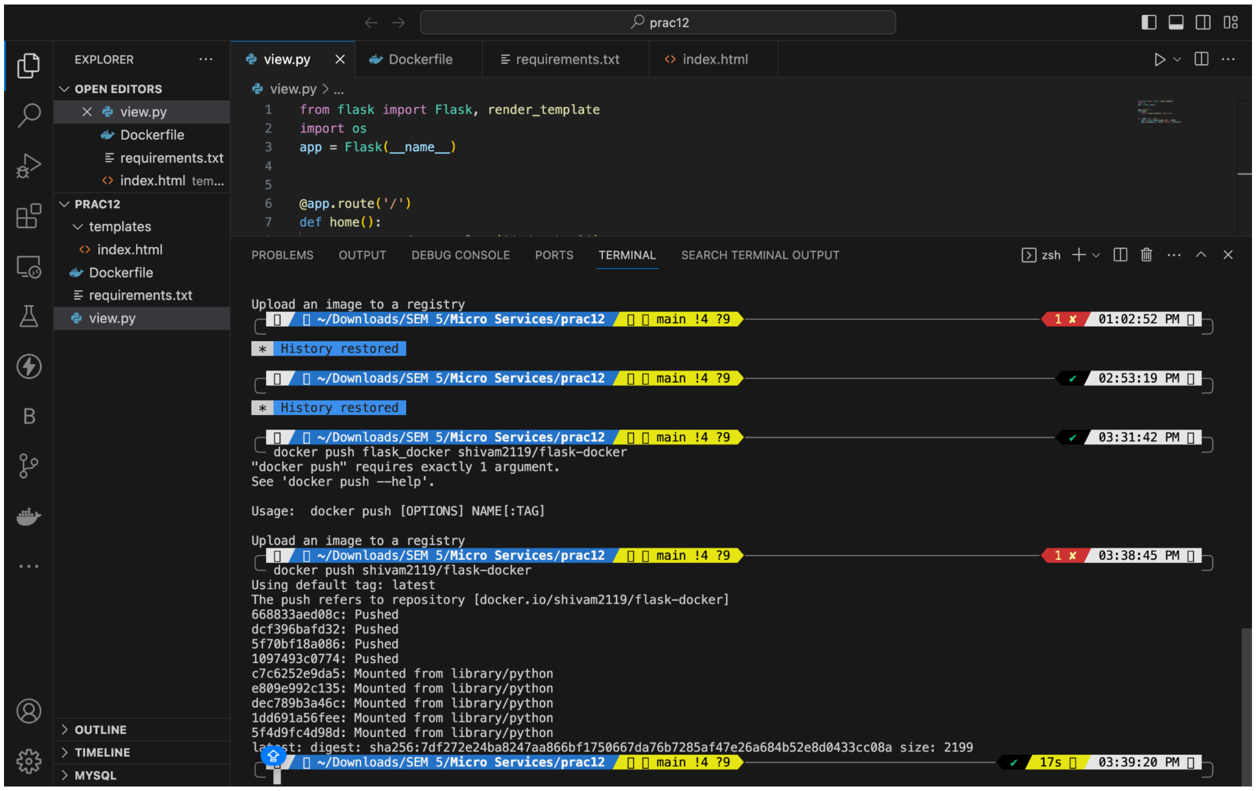
```python
from flask import Flask, render_template
import os
app = Flask(__name__)


@app.route('/')
def home():
    return render_template('index.html')


if __name__ == "__main__":
    port = int(os.environ.get('PORT', 5000))
    app.run(debug=True, host='0.0.0.0', port=port)
```

```
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9 ─────────── ✔  7s   12:56:58 PM
 docker run -p 5000:5000 -d flask_docker
8f1db37e93771dbbd61d2349a4abe1994651e2c70bdeabf085598eb405ce8ba0
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9 ─────────────────── ✔  12:58:13 PM
 docker login
Authenticating with existing credentials...
Login Succeeded
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9 ─────────── ✔  3s   01:02:03 PM
 docker tag flask_docker shivam2119/flask-docker
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9 ─────────────────── ✔  01:02:35 PM
 docker push flask_docker shivam2119/flask-docker
"docker push" requires exactly 1 argument.
See 'docker push --help'.

Usage:  docker push [OPTIONS] NAME[:TAG]

Upload an image to a registry
 ~/Downloads/SEM 5/Micro Services/prac12   main !4 ?9 ─────────────── 1 ✘  01:02:52 PM
```



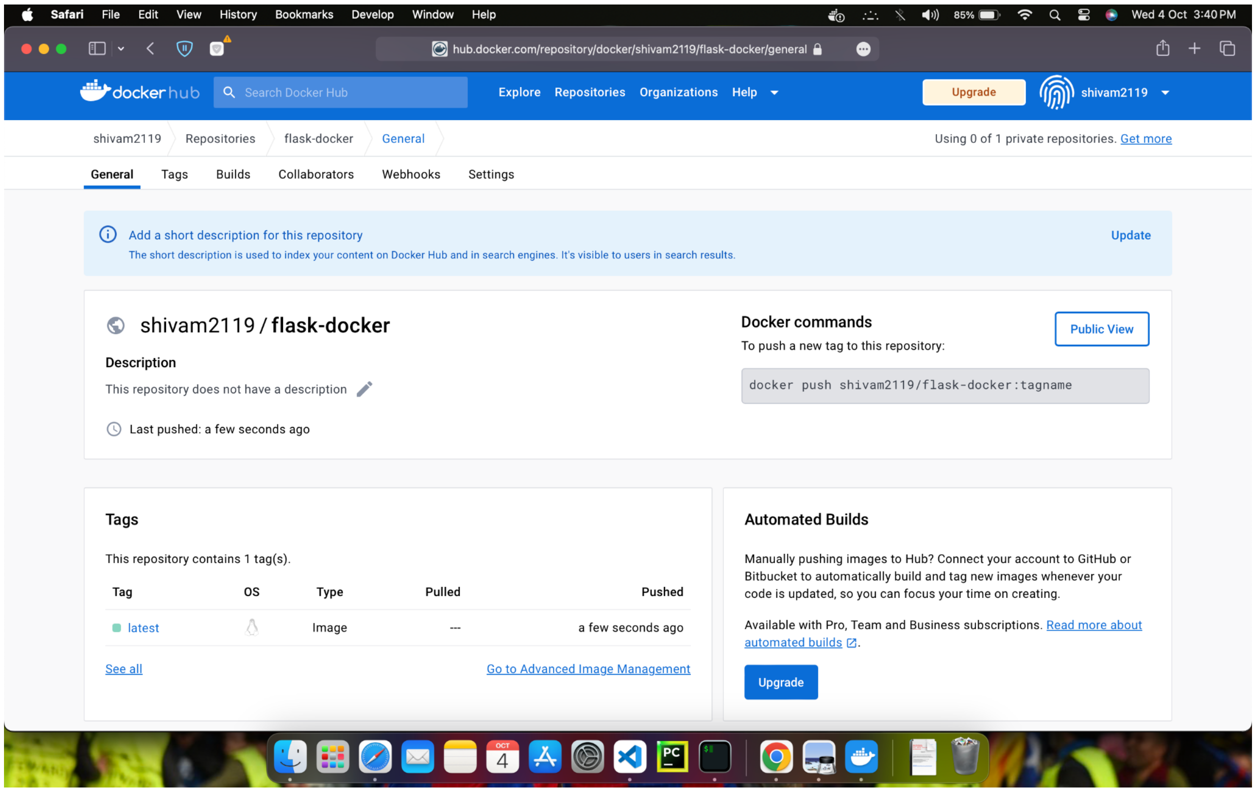
Docker Desktop
Update to latest
Search for images, containers, volumes, extensions and more...
⌘K
shiva...
DockerCon 2023: Our annual developer event is back - online & in person. Learn more
Containers
Images
Volumes
Dev Environments BETA
Docker Scout
Learning center
Extensions
Add Extensions
Images
Give feedback
Local
Hub
Artifactory EARLY ACCESS
73.97 MB / 2.01 GB in use
23 images
Last refresh: 2 hours ago
flask
Showing 2 items
RAM 3.18 GB CPU 3.54% Disk 53.99 GB avail. of 62.67 GB Signed in
v4.23.0

Name	Tag	Status	Created	Size	Actions
flask_docker 5de0fae9367d	latest	In use	6 minutes ago	69.92 MB	
shivam2119/flask-docker 5de0fae9367d	latest	In use	6 minutes ago	69.92 MB	



```python
from flask import Flask, render_template
import os
app = Flask(__name__)


@app.route('/')
def home():
```

```
Upload an image to a registry
 ~/Downloads/SEM 5/Micro Services/prac12  main !4 ?9                1 ✘  01:02:52 PM
* History restored
 ~/Downloads/SEM 5/Micro Services/prac12  main !4 ?9                  ✔  02:53:19 PM
* History restored
 ~/Downloads/SEM 5/Micro Services/prac12  main !4 ?9                  ✔  03:31:42 PM
  docker push flask_docker shivam2119/flask-docker
"docker push" requires exactly 1 argument.
See 'docker push --help'.

Usage:  docker push [OPTIONS] NAME[:TAG]

Upload an image to a registry
 ~/Downloads/SEM 5/Micro Services/prac12  main !4 ?9                1 ✘  03:38:45 PM
  docker push shivam2119/flask-docker
Using default tag: latest
The push refers to repository [docker.io/shivam2119/flask-docker]
668833aed08c: Pushed
dcf396bafd32: Pushed
5f70bf18a086: Pushed
1097493c0774: Pushed
c7c6252e9da5: Mounted from library/python
e809e992c135: Mounted from library/python
dec789b3a46c: Mounted from library/python
1dd691a56fee: Mounted from library/python
5f4d9fc4d98d: Mounted from library/python
latest: digest: sha256:7df272e24ba8247aa866bf1750667da76b7285af47e26a684b52e8d0433cc08a size: 2199
 ~/Downloads/SEM 5/Micro Services/prac12  main !4 ?9            ✔  17s  03:39:20 PM
```



Add a short description for this repository

The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results.

shivam2119 / **flask-docker**

Description

This repository does not have a description

Last pushed: a few seconds ago

Docker commands

To push a new tag to this repository:

```
docker push shivam2119/flask-docker:tagname
```

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	---	a few seconds ago

See all

Go to Advanced Image Management

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

Upgrade