

ASSIGNMENT – 4

DOCKER AND KUBERNETES

Student Name	Edwin Rajan G
Student Roll Number	921319104041
Maximum Marks	2 marks

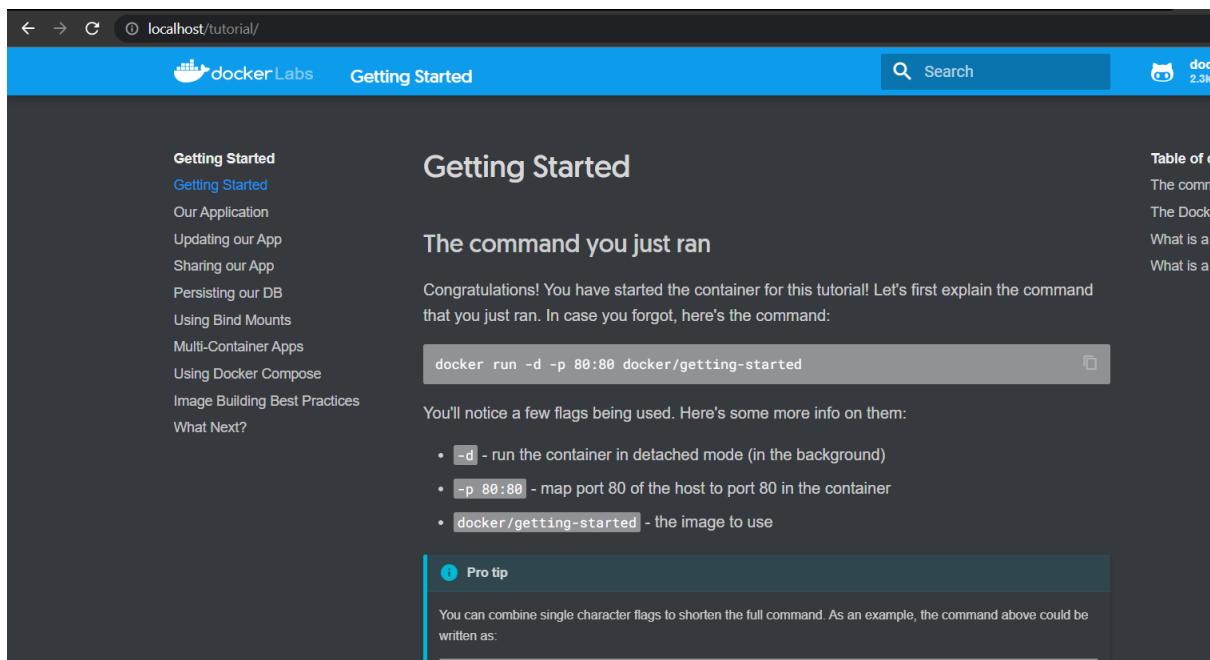
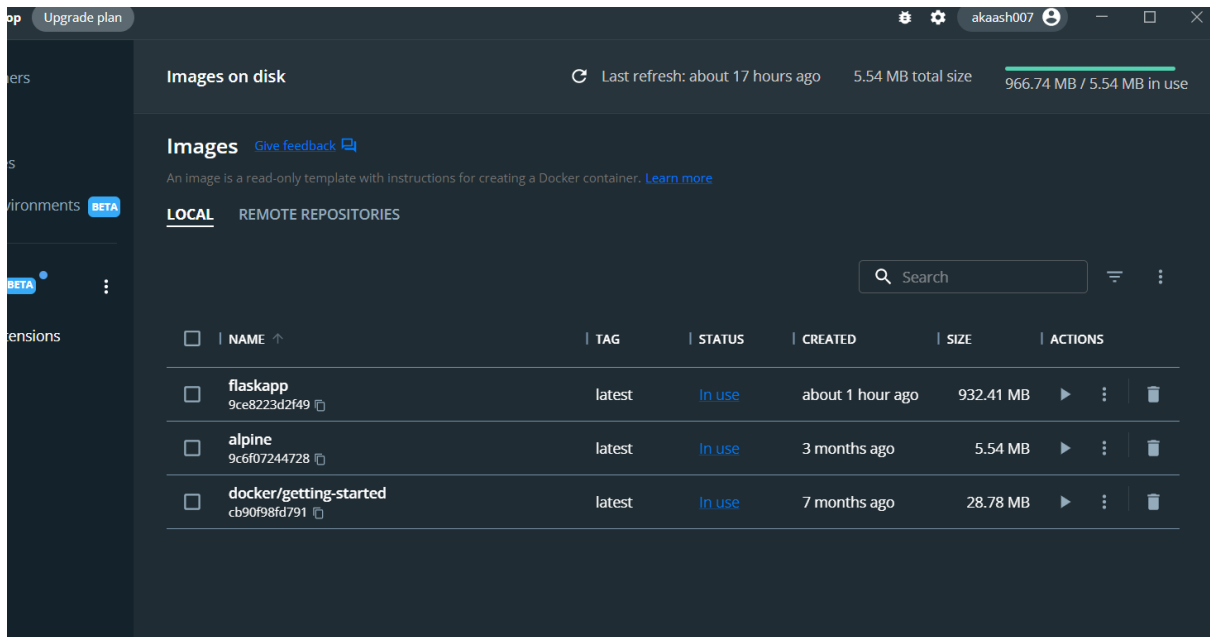
Question-1: pull an image from docker hub and run it in docker playground.

- 1) pull an image form docker hub

```
PowerShell
Loading personal and system profiles took 541ms.
→ assignment 4 git:(main) docker pull docker/getting-started
Using default tag: latest
latest: Pulling from docker/getting-started
df9b9388f04a: Pull complete
5867cba5fcdb: Pull complete
4b639e65cb3b: Pull complete
061ed9e2b976: Pull complete
bc19f3e8eeb1: Pull complete
4071be97c256: Pull complete
79b586f1a54b: Pull complete
0c9732f525d6: Pull complete
Digest: sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aae
Status: Downloaded newer image for docker/getting-started:latest
docker.io/docker/getting-started:latest
→ assignment 4 git:(main) |
```

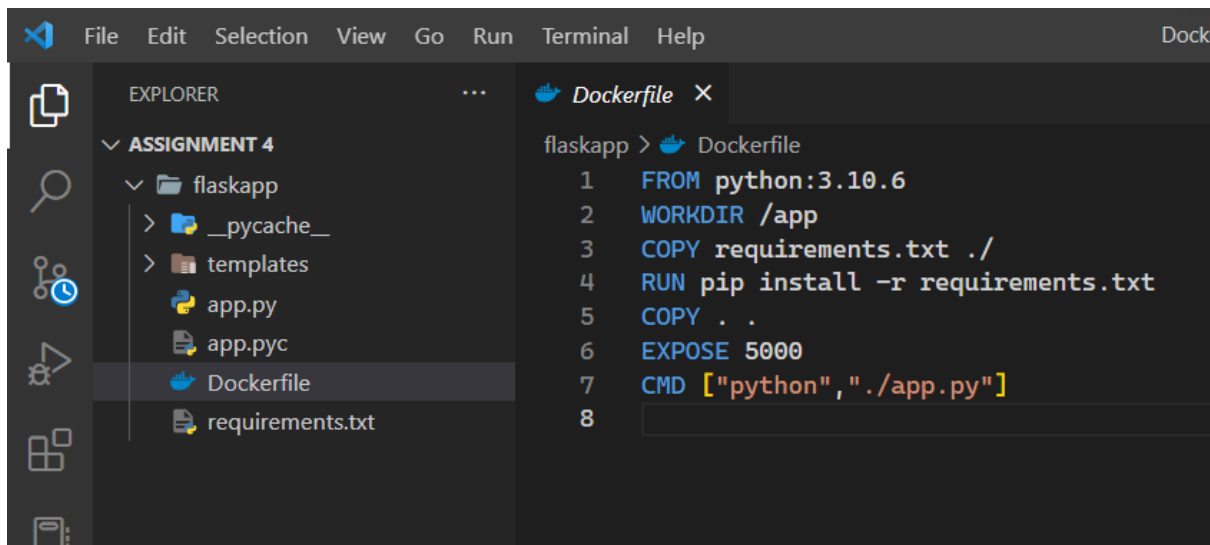
- 2) run it in docker playground

```
Digest: sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aae
Status: Downloaded newer image for docker/getting-started:latest
docker.io/docker/getting-started:latest
→ assignment 4 git:(main) docker run -d -p 80:80 docker/getting-started
ee6d34bd49e20106c8d3a3cc85bab0bde9c96a667bb3112bc896358efd6d2f68
→ assignment 4 git:(main) D|
```



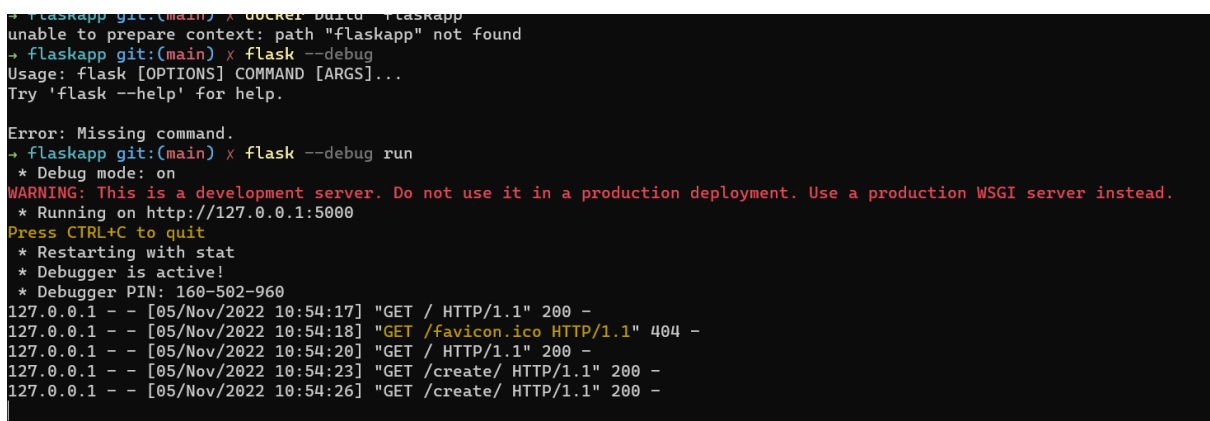
Question-2: Create a docker file for the job portal application and deploy it in docker application.

1)Creating a docker file for the job portal application



The screenshot shows the Visual Studio Code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'ASSIGNMENT 4' containing a folder 'flaskapp' and files 'app.py', 'app.pyc', 'Dockerfile', and 'requirements.txt'. The code editor shows the content of the 'Dockerfile' file, which is as follows:

```
flaskapp > Dockerfile
1 FROM python:3.10.6
2 WORKDIR /app
3 COPY requirements.txt ./
4 RUN pip install -r requirements.txt
5 COPY . .
6 EXPOSE 5000
7 CMD ["python", "./app.py"]
8
```

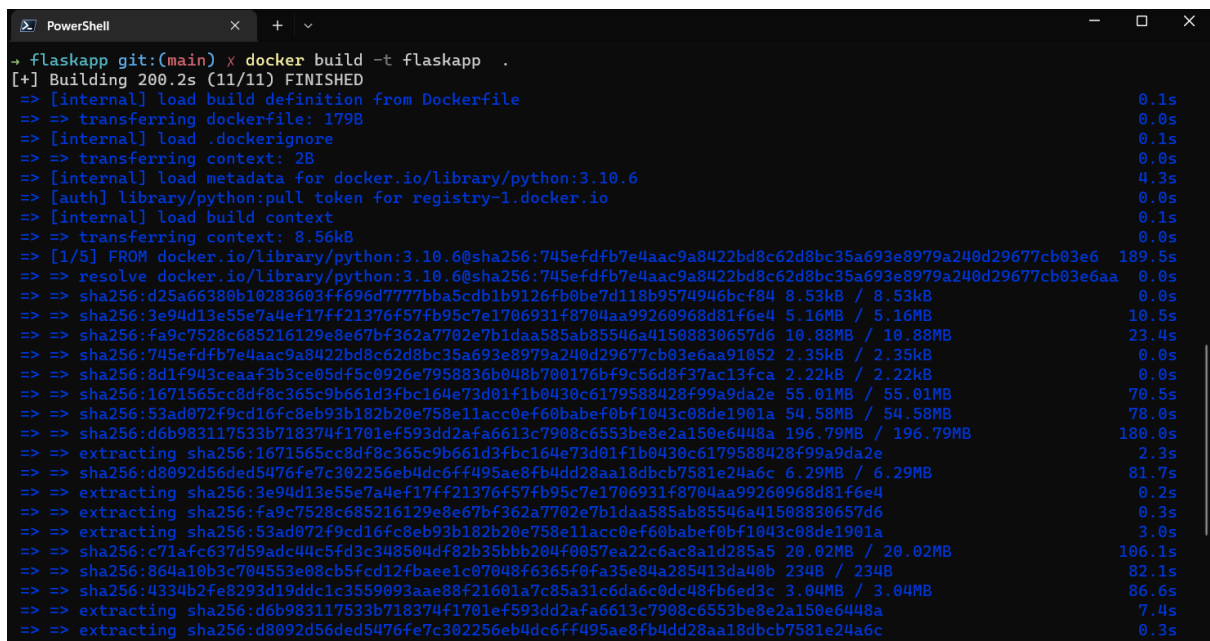


The screenshot shows the terminal output of the commands executed to build and run the Flask application. The output is as follows:

```
+ flaskapp git:(main) x docker build -t flaskapp
unable to prepare context: path "flaskapp" not found
+ flaskapp git:(main) x flask --debug
Usage: flask [OPTIONS] COMMAND [ARGS]...
Try 'flask --help' for help.

Error: Missing command.
+ flaskapp git:(main) x flask --debug run
* 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 http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 160-502-960
127.0.0.1 - - [05/Nov/2022 10:54:17] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Nov/2022 10:54:18] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [05/Nov/2022 10:54:20] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Nov/2022 10:54:23] "GET /create/ HTTP/1.1" 200 -
127.0.0.1 - - [05/Nov/2022 10:54:26] "GET /create/ HTTP/1.1" 200 -
```

2)deploy in docker application



The screenshot shows the PowerShell terminal output of the command to build the Docker image. The output is as follows:

```
+ flaskapp git:(main) x docker build -t flaskapp .
[+] Building 200.2s (11/11) FINISHED
=> [internal] load build definition from Dockerfile                                0.1s
=> => transferring dockerfile: 179B                                              0.0s
=> [internal] load .dockerignore                                                  0.1s
=> => transferring context: 2B                                                    0.0s
=> [internal] load metadata for docker.io/library/python:3.10.6                 4.3s
=> [auth] library/python:pull token for registry-1.docker.io                    0.0s
=> [internal] load build context                                                  0.1s
=> => transferring context: 8.56kB                                                0.0s
=> [1/5] FROM docker.io/library/python:3.10.6@sha256:745efdfb7e4aac9a8422bd8c62d8bc35a693e8979a240d29677cb03e6 189.5s
=> => resolve docker.io/library/python:3.10.6@sha256:745efdfb7e4aac9a8422bd8c62d8bc35a693e8979a240d29677cb03e6aa 0.0s
=> => sha256:d25a66380b10283603ff696d7777bba5c5db1b9126fb0be7d118b9574946bcf84 8.53kB / 8.53kB 0.0s
=> => sha256:3e94d13e55e7a4ef17ff21376f57fb95c7e1706931f8704aa99260968d81f6e4 5.16MB / 5.16MB 10.5s
=> => sha256:fa9c7528c685216129e8e67bf362a7702e7b1daa585ab85546a41508830657d6 10.88MB / 10.88MB 23.4s
=> => sha256:745efdfb7e4aac9a8422bd8c62d8bc35a693e8979a240d29677cb03e6aa91052 2.35kB / 2.35kB 0.0s
=> => sha256:8d1f943ceaf3b3ce05df5c0926e7958836b048b700176bf9c56d8f37ac13fca 2.22kB / 2.22kB 0.0s
=> => sha256:1671565cc8df8c365c9b661d3fbc164e73d01f1b0430c6179588428f99a9da2e 55.01MB / 55.01MB 70.5s
=> => sha256:53ad072f9cd16fc8eb93b182b20e758e11acc0ef60babe0bf1043c08de1901a 54.58MB / 54.58MB 78.0s
=> => sha256:d6b983117533b718374f1701ef593dd2afa6613c7908c6553be8e2a150e6448a 196.79MB / 196.79MB 180.0s
=> => extracting sha256:1671565cc8df8c365c9b661d3fbc164e73d01f1b0430c6179588428f99a9da2e 2.3s
=> => sha256:d8092d56ded5476fe7c30225eb4dc6ff495ae8fb4dd28aa18dbcb7581e24a6c 6.29MB / 6.29MB 81.7s
=> => extracting sha256:3e94d13e55e7a4ef17ff21376f57fb95c7e1706931f8704aa99260968d81f6e4 0.2s
=> => extracting sha256:fa9c7528c685216129e8e67bf362a7702e7b1daa585ab85546a41508830657d6 0.3s
=> => extracting sha256:53ad072f9cd16fc8eb93b182b20e758e11acc0ef60babe0bf1043c08de1901a 3.0s
=> => sha256:c71afc637d59adc44c5fd3c348504df82b35bbb204f0057ea22c6ac8a1d285a5 20.02MB / 20.02MB 106.1s
=> => sha256:864a10b3c704553e08cb5fcd12fbaee1c07048f6365f0fa35e84a285413da40b 234B / 234B 82.1s
=> => sha256:4334b2fe8293d19ddc1c3559093aae88f21601a7c85a31c6da6c0dc48fb6ed3c 3.04MB / 3.04MB 86.6s
=> => extracting sha256:d6b983117533b718374f1701ef593dd2afa6613c7908c6553be8e2a150e6448a 7.4s
=> => extracting sha256:d8092d56ded5476fe7c30225eb4dc6ff495ae8fb4dd28aa18dbcb7581e24a6c 0.3s
```

```
PowerShell
=> => sha256:fa9c7528c685216129e8e67bf362a7702e7b1daa585ab85546a41508830657d6 10.88MB / 10.88MB 23.4s
=> => sha256:745efd7b7e4aac9a8422bd8c62d8bc35a693e8979a240d29677cb03e6aa91052 2.35kB / 2.35kB 0.0s
=> => sha256:8d1f943ceaa3b3ce05df5c0926e7958836b048b700176bf9c56d8f37ac13fca 2.22kB / 2.22kB 0.0s
=> => sha256:1671565cc8df8c365c9b661d3fbc164e73d01f1b0430c6179588428f99a9da2e 55.01MB / 55.01MB 70.5s
=> => sha256:53ad072f9cd16fc8eb93b182b20e758e11acc0ef60babe0bf1043c08de1901a 54.58MB / 54.58MB 78.0s
=> => sha256:d6b983117533b718374f1701ef593dd2afa6613c7908c6553be8e2a150e6448a 196.79MB / 196.79MB 180.0s
=> => extracting sha256:1671565cc8df8c365c9b661d3fbc164e73d01f1b0430c6179588428f99a9da2e 2.3s
=> => sha256:d8092d56ded5476fe7c302256eb4dc6ff495ae8fb4dd28aa18dbcb7581e24a6c 6.29MB / 6.29MB 81.7s
=> => extracting sha256:3e94d13e55e7a4ef17ff21376f57fb95c7e1706931f8704aa99260968d81f6e4 0.2s
=> => extracting sha256:fa9c7528c685216129e8e67bf362a7702e7b1daa585ab85546a41508830657d6 0.3s
=> => extracting sha256:53ad072f9cd16fc8eb93b182b20e758e11acc0ef60babe0bf1043c08de1901a 3.0s
=> => sha256:c71afc637d59adc44c5fd3c348504df82b35bbb204f0057ea22c6ac8a1d285a5 20.02MB / 20.02MB 106.1s
=> => sha256:864a10b3c704553e08cb5fcd12fbaee1c07048f6365f0fa35e84a285413da40b 234B / 234B 82.1s
=> => sha256:4334b2fe8293d19ddc1c3559093aae88f21601a7c85a31c6da6c0dc48fb6ed3c 3.04MB / 3.04MB 86.6s
=> => extracting sha256:d6b983117533b718374f1701ef593dd2afa6613c7908c6553be8e2a150e6448a 7.4s
=> => extracting sha256:d8092d56ded5476fe7c302256eb4dc6ff495ae8fb4dd28aa18dbcb7581e24a6c 0.3s
=> => extracting sha256:c71afc637d59adc44c5fd3c348504df82b35bbb204f0057ea22c6ac8a1d285a5 0.8s
=> => extracting sha256:864a10b3c704553e08cb5fcd12fbaee1c07048f6365f0fa35e84a285413da40b 0.0s
=> => extracting sha256:4334b2fe8293d19ddc1c3559093aae88f21601a7c85a31c6da6c0dc48fb6ed3c 0.3s
=> [2/5] WORKDIR /app 1.1s
=> [3/5] COPY requirements.txt ./ 0.0s
=> [4/5] RUN pip install -r requirements.txt 4.9s
=> [5/5] COPY . . 0.1s
=> exporting to image 0.2s
=> => exporting layers 0.2s
=> => writing image sha256:9ce8223d2f49cc126af77c5b826056fccf1f446e84e6e67fa97b58be5718931e 0.0s
=> => naming to docker.io/library/flaskapp 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
-> flaskapp git:(main) |
```

Images on disk

Last refresh: about 16 hours ago5.54 MB total size5.54 MB / 5.54 MB in use

Images

Give feedback

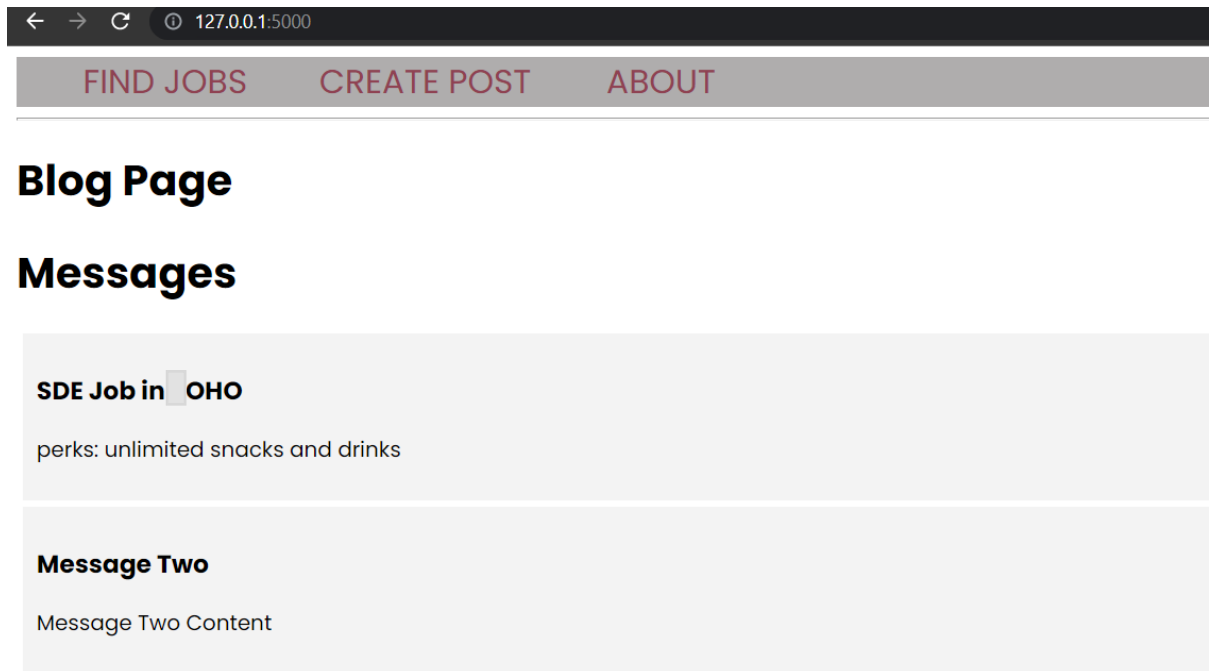
An image is a read-only template with instructions for creating a Docker container. [Learn more](#)

LOCAL

REMOTE REPOSITORIES

Q Search

	NAME	TAG	STATUS	CREATED	SIZE	ACTIONS
<input type="checkbox"/>	<div>flaskapp</div> <div>9ce8223d2f49</div>	latest	Unused	less than a minute a	932.41 MB	<div></div> <div></div> <div></div>
<input type="checkbox"/>	<div>alpine</div> <div>9c6f07244728</div>	latest	In use	3 months ago	5.54 MB	<div></div> <div></div> <div></div>



Question-3: Create a IBM container registry and deploy hello world app or jobportalapp

- 1) create a IBM container registry

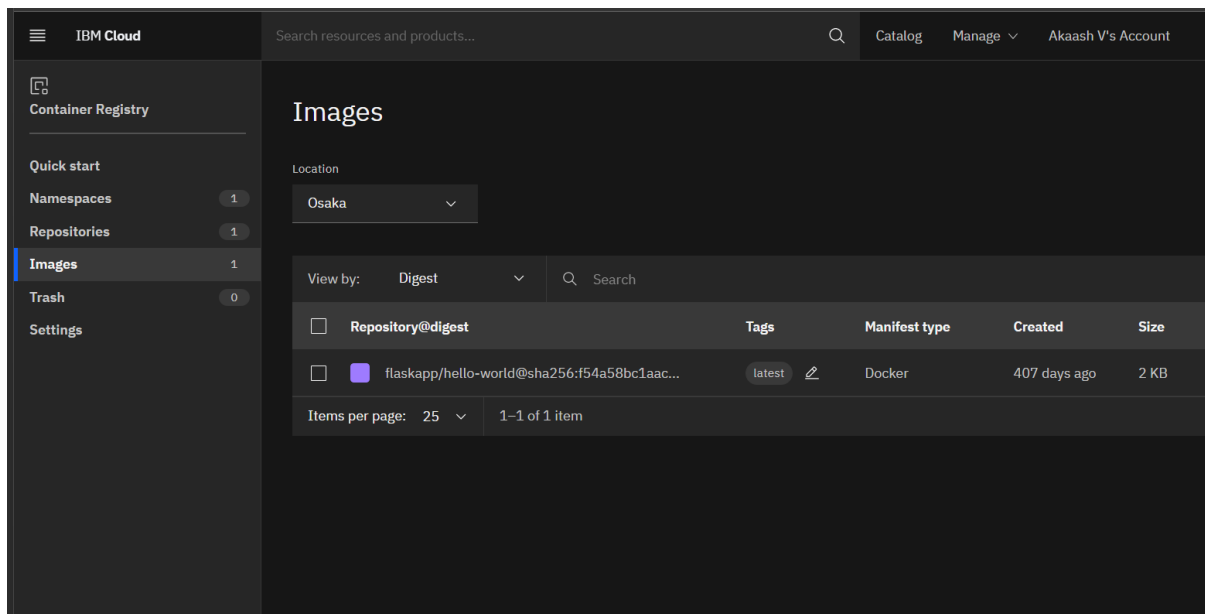
```
→ ~ git:(main) x ibmcloud
NAME:
  C:\Program Files\IBM\Cloud\bin\ibmcloud.exe - A command line tool to interact with IBM Cloud
  Find more information at: https://ibm.biz/cli-docs

USAGE:
  [environment variables] C:\Program Files\IBM\Cloud\bin\ibmcloud.exe [global options] command [arguments.
  ptions]

VERSION:
  2.12.1+b8488a1-2022-10-31T15:08:10+00:00

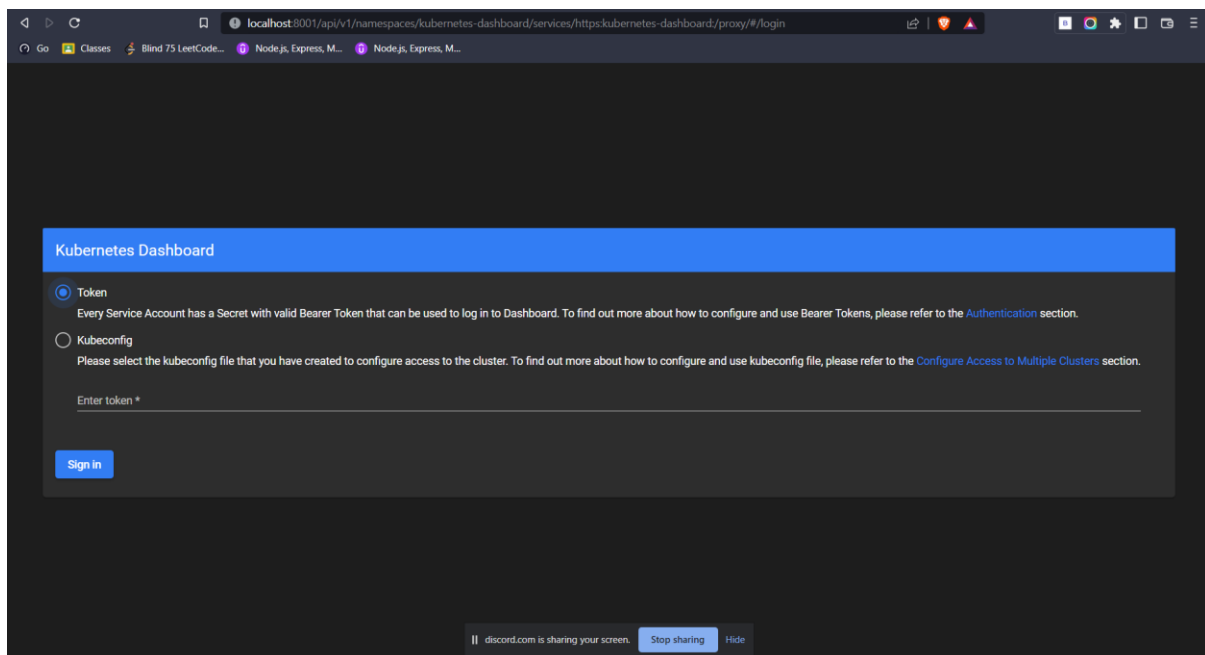
COMMANDS:
  account      Manage accounts, users, orgs and spaces
  api          Set or view target API endpoint
  billing      Retrieve usage and billing information
  catalog      Manage catalog
  cf           Run Cloud Foundry CLI with IBM Cloud CLI context
  config       Write default values to the config
  cr           Manage IBM Cloud Container Registry content and configuration.
  dev         Create, develop, deploy, and monitor applications
  enterprise   Manage enterprise, account groups and accounts.
  iam          Manage identities and access to resources
  login        Log user in
  logout       Log user out
  plugin       Manage plug-ins and plug-in repositories
  regions      List all the regions
```

2)deploy hello world or jobportal



Question-4: Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in note port

1)Creating a Kubernetes cluster in IBM cloud




Kubernetes Dashboard

IBM

CAD-B2-2M4E (Evening Session)

localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/#/pod?namespace=kuberne

 **kubernetes**

kubernetes-das...

Search

Workloads > Pods

Workloads N

Cron Jobs

Daemon Sets

Deployments

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Service

Ingresses N

Ingress Classes






Services N

Config and Storage

Config Maps N

Persistent Volume Claims N

Pods

	Name	Images	Labels	Node
	flask-app-8569bc947b-7q5l2	flask-app-testing	app: flask-app pod-template-hash: 8569bc947b	docker-desktop
	flask-app-8569bc947b-ln2xd	flask-app-testing	app: flask-app pod-template-hash: 8569bc947b	docker-desktop
	flask-app-8569bc947b-tmzp8	flask-app-testing	app: flask-app pod-template-hash: 8569bc947b	docker-desktop
	dashboard-metrics-scraper-64bcc67c9c-2zstf	kubernetesui/metrics-scraper:v1.0.8	k8s-app: dashboard-metrics-scraper pod-template-hash: 64bcc67c9c	docker-desktop
	kubernetes-dashboard-66c887f759-fp85b	kubernetesui/dashboard:v2.6.1	k8s-app: kubernetes-dashboard pod-template-hash: 66c887f759	docker-desktop