

ASSIGNMENT – 4

DOCKER AND KUBERNETES

Date	04 November 2022
Team ID	PNT2022TNID27727
Project Name	Inventory Management System For Retailers

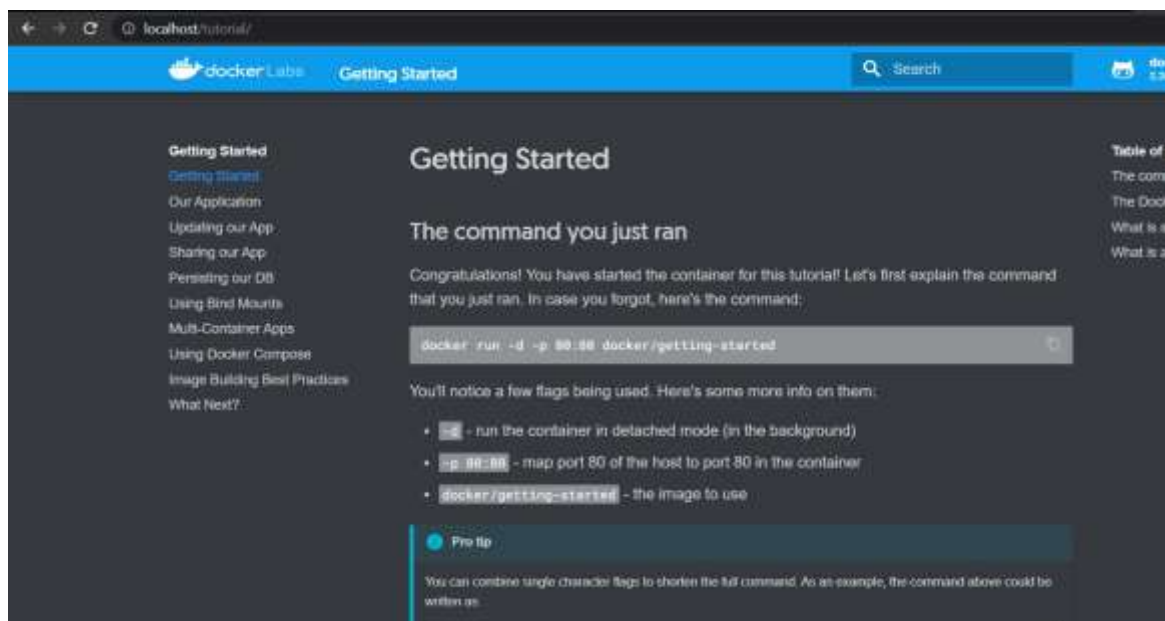
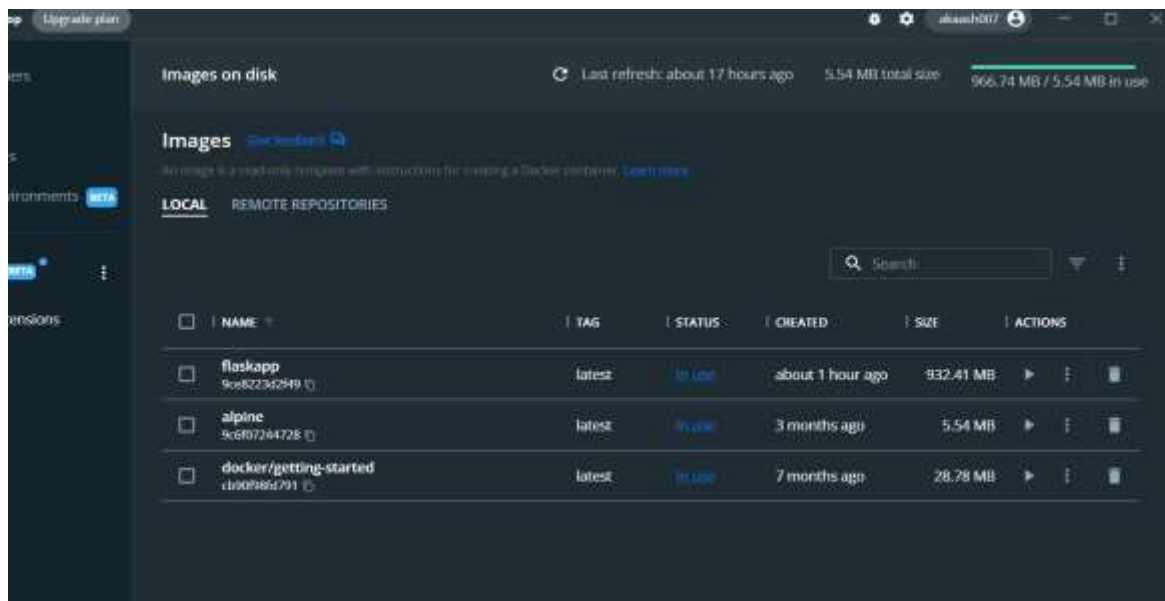
Question-1: pull an image from docker hub and run it in docker playground.

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
5867cba5fcbd: 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) |
```

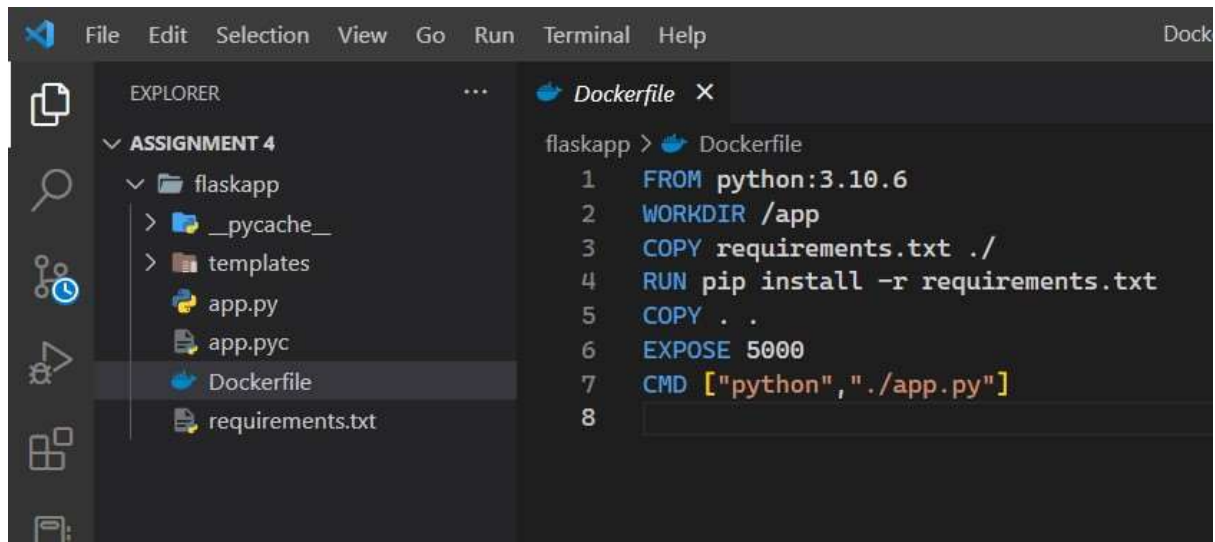
2)run it in docker playground

```
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|
```



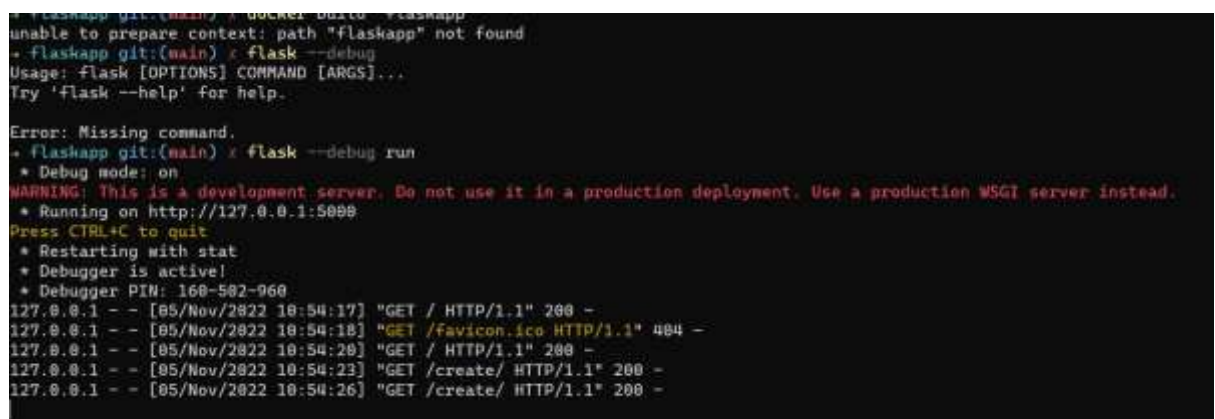
Question-2: Create a docker file for the job portal application and deploy it in docker application.

1)Creating a docker file for the job portal application



The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane shows a project named 'ASSIGNMENT 4' containing a folder 'flaskapp'. Inside 'flaskapp', there are files: '__pycache__', 'templates', 'app.py', 'app.pyc', 'Dockerfile', and 'requirements.txt'. The 'Dockerfile' file is selected. On the right, the Dockerfile content is displayed in a text editor. The Dockerfile contains the following instructions:

```
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 a terminal window with the following output:

```
flaskapp git:(main) x docker build -f Dockerfile 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.

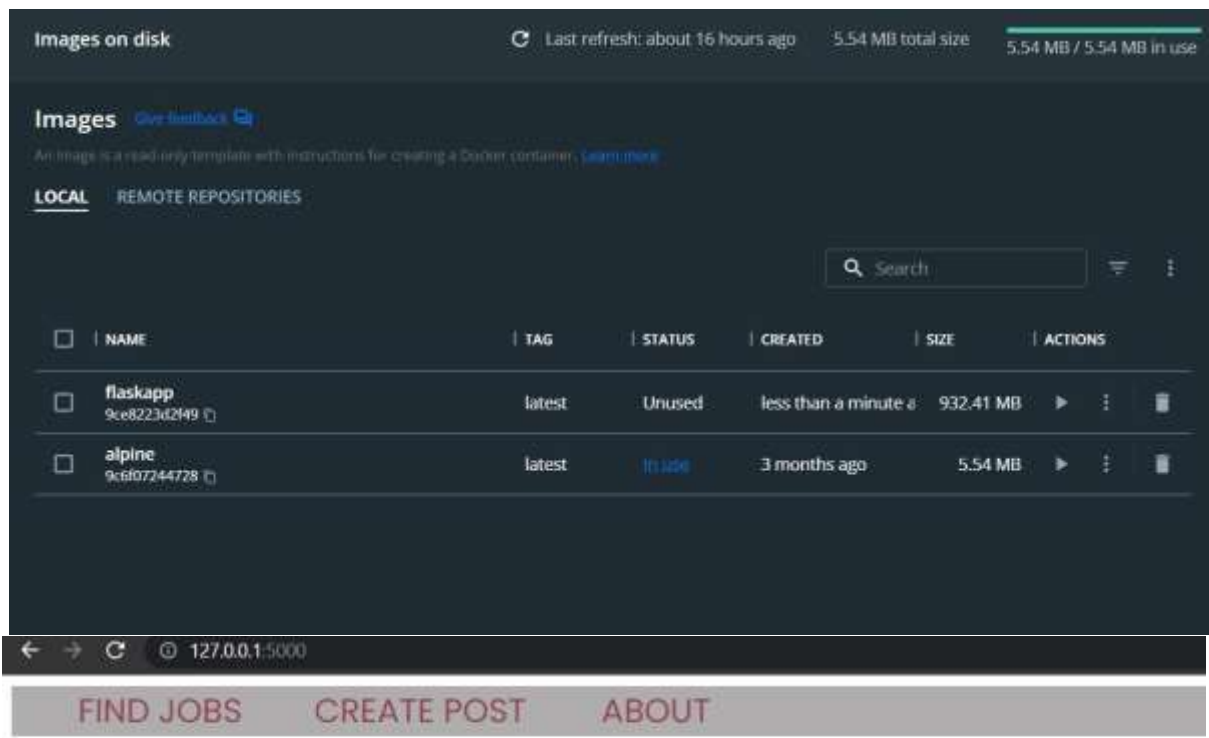
Error: Missing command.
flaskapp git:(main) x flask --debug run
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 168-582-968
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 it in docker application

```
PowerShell
- flaskapp git:(main) - 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: 1B 0.0s
=> [internal] load metadata for docker.io/library/python:3.10.6 0.3s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [internal] load build context 0.1s
=> => transferring context: 8.56kB 0.0s
=> [1/8] FROM docker.io/library/python:3.10.6@sha256:748efdfb7e4aac9a8022bd8c63d8bc3da693e8979a296c19677cb83e6ae 100.5s
=> => resolve docker.io/library/python:3.10.6@sha256:748efdfb7e4aac9a8022bd8c63d8bc3da693e8979a296c19677cb83e6ae 0.0s
=> => sha256:d2ba06308618281661ff496d7777bba5c0b1b01264908e70118e90740940ecf00 8.53kB / 8.53kB 0.0s
=> => sha256:3e90d13e667a9e174f21376f37f05c7e1706931f8704a99230966091f9e6 5.16MB / 5.16MB 10.5s
=> => sha256:fa9c7528c685216129e8e97bf362a7782e7b1daa838ab33548a91588838657d5 10.83MB / 10.83MB 23.4s
=> => sha256:748efdfb7e4aac9a8022bd8c63d8bc3da693e8979a296c19677cb83e6ae91052 2.35kB / 2.35kB 0.0s
=> => sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 196.79MB / 196.79MB 0.0s
=> => sha256:0d1f565cc8f8c365c9661d3fbc164e73d81f180438c6179580428f99a9da2e 2.35kB / 2.35kB 0.0s
=> => sha256:1671565cc8f8c365c9661d3fbc164e73d81f180438c6179580428f99a9da2e 55.81MB / 55.81MB 70.5s
=> => sha256:53ad872f9cd16cf8eb93b182b28e7b362a7782e7b1daa838ab33548a91588838657d5 54.53MB / 54.53MB 78.0s
=> => sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 196.79MB / 196.79MB 100.0s
=> => extracting sha256:1671565cc8f8c365c9661d3fbc164e73d81f180438c6179580428f99a9da2e 2.3s
=> => sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 6.29MB / 6.29MB 81.7s
=> => extracting sha256:3e90d13e667a9e174f21376f37f05c7e1706931f8704a99230966091f9e6 0.2s
=> => extracting sha256:fa9c7528c685216129e8e97bf362a7782e7b1daa838ab33548a91588838657d5 0.3s
=> => extracting sha256:53ad872f9cd16cf8eb93b182b28e7b362a7782e7b1daa838ab33548a91588838657d5 3.0s
=> => sha256:c71afc637d99adc04c56d3c348584d4f7e3b356bb204f0657ea22c6ac8a1d385a5 20.82MB / 20.82MB 100.1s
=> => sha256:08041865c704553e08cb3fcd12f8ee1c87048f6335f8fa35e84a285413da408b 234B / 234B 82.1s
=> => sha256:4334b3fa8283d19ddc1a3559093aaa88f21801a7c85a31cdda8c0ac40f66ed3c 3.80MB / 3.80MB 86.6s
=> => extracting sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 7.0s
=> => extracting sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 0.3s
```

```
PowerShell
=> => sha256:fa9c7528c685216129e8e97bf362a7782e7b1daa838ab33548a91588838657d5 10.83MB / 10.83MB 23.4s
=> => sha256:748efdfb7e4aac9a8022bd8c63d8bc3da693e8979a296c19677cb83e6ae91052 2.35kB / 2.35kB 0.0s
=> => sha256:b01f493ceaf7b31cc85df5c8926e7958836b040b706178b79c5608f37a13fca 2.22kB / 2.22kB 0.0s
=> => sha256:1671565cc8f8c365c9661d3fbc164e73d81f180438c6179580428f99a9da2e 55.81MB / 55.81MB 70.5s
=> => sha256:53ad872f9cd16cf8eb93b182b28e7b362a7782e7b1daa838ab33548a91588838657d5 54.53MB / 54.53MB 78.0s
=> => sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 196.79MB / 196.79MB 100.0s
=> => extracting sha256:1671565cc8f8c365c9661d3fbc164e73d81f180438c6179580428f99a9da2e 2.3s
=> => sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 6.29MB / 6.29MB 81.7s
=> => extracting sha256:3e90d13e667a9e174f21376f37f05c7e1706931f8704a99230966091f9e6 0.2s
=> => extracting sha256:fa9c7528c685216129e8e97bf362a7782e7b1daa838ab33548a91588838657d5 0.3s
=> => extracting sha256:53ad872f9cd16cf8eb93b182b28e7b362a7782e7b1daa838ab33548a91588838657d5 3.0s
=> => sha256:c71afc637d99adc04c56d3c348584d4f7e3b356bb204f0657ea22c6ac8a1d385a5 20.82MB / 20.82MB 100.1s
=> => sha256:08041865c704553e08cb3fcd12f8ee1c87048f6335f8fa35e84a285413da408b 234B / 234B 82.1s
=> => sha256:4334b3fa8283d19ddc1a3559093aaa88f21801a7c85a31cdda8c0ac40f66ed3c 3.80MB / 3.80MB 86.6s
=> => extracting sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 7.0s
=> => extracting sha256:d80983117533b718370f1781e593bd2a4a6613c7998c6583be8a2a158e6408a 0.3s
=> => extracting sha256:c71afc637d99adc04c56d3c348584d4f7e3b356bb204f0657ea22c6ac8a1d385a5 0.8s
=> => extracting sha256:08041865c704553e08cb3fcd12f8ee1c87048f6335f8fa35e84a285413da408b 0.0s
=> => extracting sha256:4334b3fa8283d19ddc1a3559093aaa88f21801a7c85a31cdda8c0ac40f66ed3c 0.3s
[2/5] WORKDIR /app 1.1s
[3/5] COPY requirements.txt ./ 0.0s
[4/5] RUN pip install -r requirements.txt 0.9s
[5/5] COPY 0.3s
=> exporting to image 0.2s
=> exporting layers 0.3s
=> writing image sha256:9ce8273d2f49cc126a779c9b82e656fccc1f408a34c6e97fa97b58be3710911e 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) |
```



Blog Page

Messages

SDE Job in OHO

perks: unlimited snacks and drinks

Message Two

Message Two Content

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

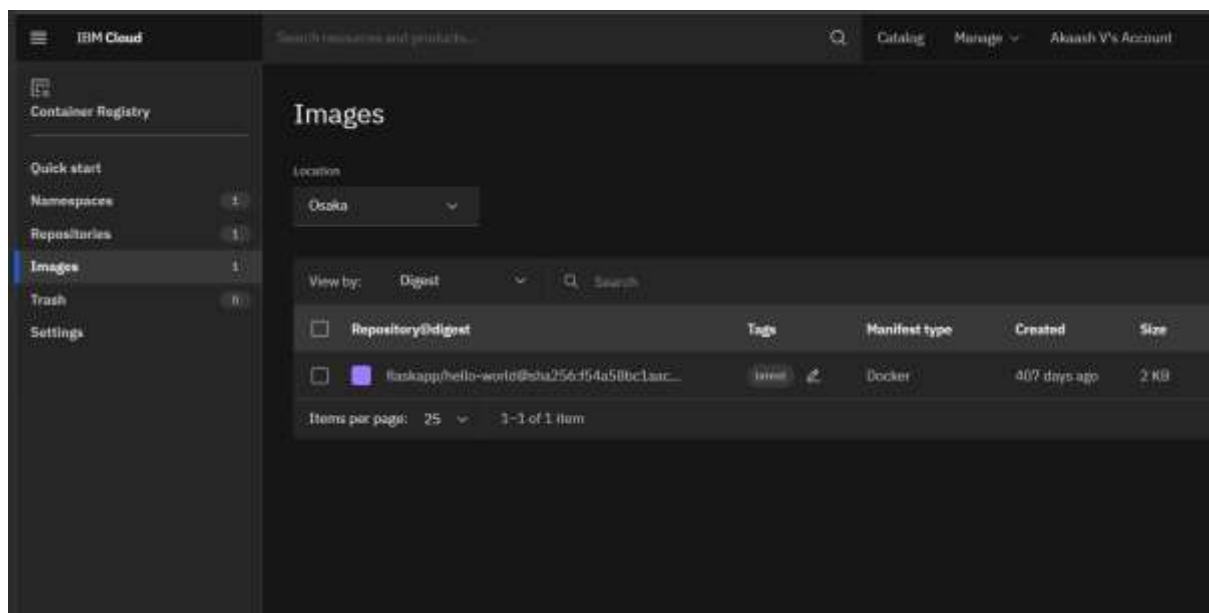
USAGE:
  [environment variables] C:\Program Files\IBM\Cloud\bin\ibmcloud.exe [global options] command [arguments.
  ptions]

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

