

Assignment 4

Name:	Udhayakumaran H
Roll No:	718020Z435
Project:	Inventory Management System for Retailers
Team ID:	PNT2022TMID12716

Dockerfile:

```
Dockerfile
1 FROM python:3.10.4
2 WORKDIR /app
3 ADD . /app
4 COPY requirements.txt /app
5 RUN python3 -m pip install -r requirements.txt
6 RUN python3 -m pip install ibm_db
7 EXPOSE 5000
8 ENTRYPOINT ["python"]
9 CMD ["app.py"]
```

Building Image:

docker build -t <image_name> .

Execute the above command in the working directory where Dockerfile is present

```
(myApp) D:\Inventory_Management_System_for_Retailers>docker build -t inventory-mgmt .
[+] Building 217.2s (12/12) FINISHED
=> [internal] load build definition from Dockerfile                                0.1s
=> => transferring dockerfile: 252B                                              0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/python:3.10.4                2.6s
=> [3/6] ADD . /app                                                             0.5s
=> [4/6] COPY requirements.txt /app                                              0.1s
=> [5/6] RUN python3 -m pip install -r requirements.txt                        208.1s
=> [6/6] RUN python3 -m pip install ibm_db                                     1.0s
=> exporting to image                                                            1.1s
=> => exporting layers                                                            1.0s
=> => writing image sha256:82d0d6d55d7e358b0b69a8f2a38ea24804dcc39a81c5d3a6554bd1a1dcf8dc78 0.0s
=> => naming to docker.io/library/inventory-mgmt                               0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

(myApp) D:\Inventory_Management_System_for_Retailers>docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
inventory-mgmt      latest      82d0d6d55d7e  About a minute  1.21GB
```

Run the container:

Docker run -d -p <port>:<port> <image_name>

Execute the command to run the container in the mentioned port number

```
(myApp) D:\Inventory_Management_System_for_Retailers>docker run -d -p 5000:5000 inventory-mgmt
25eb65cdf5cc7db4171174c02a8a73024a9228d72d440fce1c0556eb764d686f

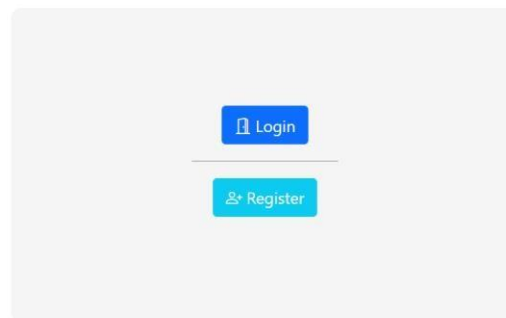
(myApp) D:\Inventory_Management_System_for_Retailers>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
25eb65cdf5cc   inventory-mgmt "python app.py"         7 seconds ago Up 6 seconds  0.0.0.0:5000->5000/tcp            affectionate_chatelet

(myApp) D:\Inventory_Management_System_for_Retailers>
```

Open localhost:5000 or 127.0.0.1:5000 to view the running container



Inventory Management System for Retailers



Tagging and pushing the image to the IBM Container Registry:

```
(myApp) D:\Inventory_Management_System_for_Retailers>ibmcloud cr login
Logging 'docker' in to 'us.icr.io'...
Logged in to 'us.icr.io'.

OK

(myApp) D:\Inventory_Management_System_for_Retailers>docker tag inventory-mgmt us.icr.io/udhayakumaran/inventory-mgmt:1.1

(myApp) D:\Inventory_Management_System_for_Retailers>docker push us.icr.io/udhayakumaran/inventory_mgmt:1.1
The push refers to repository [us.icr.io/udhayakumaran/inventory_mgmt]
b90e32367499: Layer already exists
2f535d5fc5c0: Layer already exists
0d2f265ad776: Layer already exists
dc00774c89ec: Layer already exists
fc9886f4d896: Layer already exists
9fda40ddc568: Layer already exists
428e1f341db7: Layer already exists
9ea8d200cd5d: Layer already exists
13b045a1dfd2: Layer already exists
2fbabeba902e: Layer already exists
ee509ed6e976: Layer already exists
9177197c67d0: Layer already exists
7dbadf2b9bd8: Layer already exists
e7597c345c2e: Layer already exists
1.1: digest: sha256:384ea4d17e3c6ca5099c39156b144c16e3e76065fcc77f0f9120878ba3d02152 size: 3263
```



IBM Container Registry UI:

Namespaces

Location

Dallas

Resource group: Filter... Search Create +

	Name	Resource group	Repository count	Image count	Retention policy
^	 udhayakumaran	Default	1	1	Retain all images
	Repository		Image count		Last updated
		us.icr.io/udhayakumaran/inventory_mgmt		1	4 hours ago

Items per page: 25 1-1 of 1 item 1 1 of 1 page

deployment.yaml:

```
! deployment.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: inventory-mgmt
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: flasknode
10   template:
11     metadata:
12       labels:
13         app: flasknode
14     spec:
15       containers:
16       - name: flasknode
17         image: us.icr.io/udhayakumaran/inventory_mgmt:1.1
18         imagePullPolicy: Always
19         ports:
20         - containerPort: 5000
```

Enter `ibmcloud ks cluster config -c <cluster_id>` to connect with the kubernetes cluster created in the IBM Cloud

```
(myApp) D:\Inventory_Management_System_for_Retailers>ibmcloud ks cluster config -c [REDACTED]
OK
The configuration for cdticv7f05afjutac0hg was downloaded successfully.

Added context for cdticv7f05afjutac0hg to the current kubeconfig file.
You can now execute 'kubectl' commands against your cluster. For example, run 'kubectl get nodes'.
If you are accessing the cluster for the first time, 'kubectl' commands might fail for a few seconds while
RBAC synchronizes.

(myApp) D:\Inventory_Management_System_for_Retailers>kubectl create -f deployment.yaml
deployment.apps/inventory-mgmt created

(myApp) D:\Inventory_Management_System_for_Retailers>kubectl get deployment
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
inventory-mgmt      1/1      1             1            38s
```

After deploying, expose the application to public using the command; `kubectl expose deployment inventory-mgmt --type=NodePort --name=inventory-mgmt`

Kubernetes Cluster Worker Node

Pool: Filter...

Search

Settings

Add

<input type="checkbox"/>	Name	Status	Worker pool	Zone	Private IP	Public IP	Version
<input checked="" type="checkbox"/>	0000002b	Normal	default	Milan 01	10.144.222.89	169.51.203.120	1.24.7_1543

Items per page: 251-1 of 1 item

11 of 1 page

Kubernetes Dashboard

kubernetes

default

Search

+🔔👤

Workloads

Workload Status

Running: 1

Deployments

Running: 1

Pods

Running: 1

Replica Sets

Deployments

Name	Images	Labels	Pods	Created
● inventory-mgmt	Show all	-	1 / 1	13 minutes ago

Pods

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
● inventory-mgmt-79c857c8bf-mbd87	Show all	Show all	10.144.222.89	Running	0	4.00m	59.40Mi	13 minutes ago

Replica Sets

Name	Images	Labels	Pods	Created
● inventory-mgmt-79c857c8bf	Show all	Show all	1 / 1	13 minutes ago

Application running on IBM Kubernetes Cluster



Inventory Management System for Retailers

LOGIN

Username

Password

LOGIN

Did not have an account? [SignUp](#)