Cloud Computing Experiment-2

Details :

- Name : P K Navin Shrinivas
- SRN : PES2UG20CS237
- Section : D

Install docker :

- Install using pacman

```
sudo pacman -S docker
```

- Making it sudoless :

```
sudo groupadd docker
sudo usermod -aG docker $USER
# Exit and login as same user.
```

- Install docker-compose :

```
sudo pacman -S docker-compose
```

The lab

Task 1 : Install check

- Check if docker compose and docker are installed

```
docker run hello-world #1a.png
docker compose version
```

```
➜  ~ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Task-2 : C program in ubutu image

- First pull all the needed images needed for this lab, this is so that weee cann cruisseeeee on later :

```
docker pull ubuntu && docker pull nginx && docker pull python &&
docker pull mongo
```

- Create a folder name anything
- Save the following files with respective file names :
  Dockerfile :

```
FROM ubuntu:latest
RUN apt-get update
RUN apt-get isntall gcc -y
COPY prog1.c ~/prog1.c
RUN gcc prog1.c
CMD ["./a.out"] # Will be executed on startup `run`
```

prog1.c :

```
#include<stdio.h>
```

```c
int main(){
   printf("Running this inside a container \n");
   printf("My SRN is PES2UG20CS237\n");
}
```

- Move into the folder containing both the above files and :

```
docker build -t <image-name> .
```

- We can now run the image we built like so :

```
docker run <image-name> #2a.png
```

```
➜  dockersetup1 git:(main) ✗ docker build -t simpleubuntu-c-container .
Sending build context to Docker daemon  3.072kB
Step 1/6 : FROM ubuntu:latest
 ---> 58db3edaf2be
Step 2/6 : RUN apt-get update
 ---> Using cache
 ---> f1705e8cc6cd
Step 3/6 : RUN apt-get install gcc -y
 ---> Using cache
 ---> e2e94226f446
Step 4/6 : COPY prog1.c prog1.c
 ---> 3d2fecc9a54f
Step 5/6 : RUN gcc prog1.c
 ---> Running in 3c302bbf58d7
Removing intermediate container 3c302bbf58d7
 ---> f0433cb4c060
Step 6/6 : CMD ["./a.out"] # Will be executed on startup `run`
 ---> Running in e9d7cd8e326d
Removing intermediate container e9d7cd8e326d
 ---> 1635bbfbdb87
Successfully built 1635bbfbdb87
Successfully tagged simpleubuntu-c-container:latest
➜  dockersetup1 git:(main) ✗ docker run simpleubuntu-c-container
Running this inside a container
My SRN is PES2UG20CS237
➜  dockersetup1 git:(main) ✗ _
```

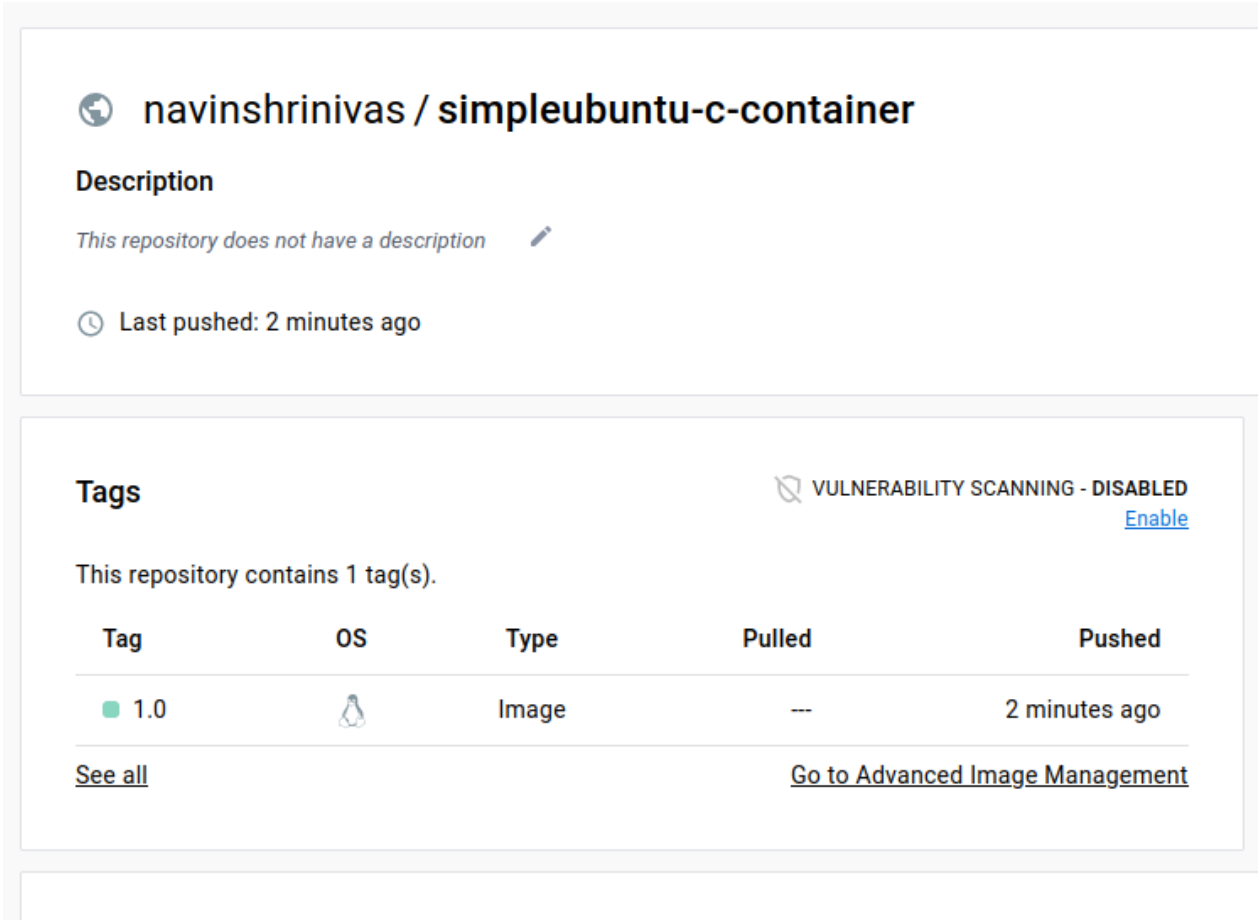
- Upload the image to docker hub :

```
docker login -u "username" -p "password" docker.io
```

- Create tag and upload :

```
docker tag <image-name> <username>/<image-name>:1.0
docker push <username>/<image-name>:1.0
```

- Go to docker hub and get screenshot of the uploaded image (2b.png)



Task 3 : Exposing ports and docker networks

- Create another directory named anything and create the following files with respective file names :

my.html :

```
<html>
        <body>
                <h1>My SRN is PES2UG20CS237</h1>
                <h2>I am running a nginx container!</h2>
        </body>
</html>
```

Dockerfile :

```
FROM nginx
COPY my.html /usr/share/nginx/html
```

docker-compose.yml :

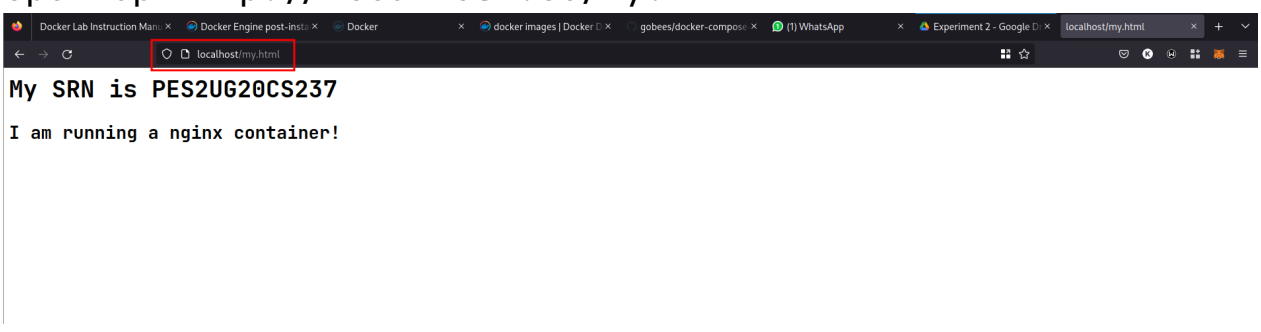
```
version: "3"
services:
  nginxserver:
    image: "nginxhelloworld"
    ports:
      - "80:80"
```

- Execute the following commands in the new folder with the above files:

```
docker build -t nginxhelloworld .
docker compose up #3a.png
```

```
➜  dockercompose2 git:(main) ✗ docker compose up
[+] Running 1/0
 ⠿ Container dockercompose2-nginxserver-1  Created                                                0.0s
Attaching to dockercompose2-nginxserver-1
dockercompose2-nginxserver-1  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configurat
ion
dockercompose2-nginxserver-1  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
dockercompose2-nginxserver-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
dockercompose2-nginxserver-1  | 10-listen-on-ipv6-by-default.sh: info: IPv6 listen already enabled
dockercompose2-nginxserver-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
dockercompose2-nginxserver-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
dockercompose2-nginxserver-1  | /docker-entrypoint.sh: Configuration complete; ready for start up
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: using the "epoll" event method
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: nginx/1.23.3
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: OS: Linux 6.1.8-arch1-1
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker processes
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 22
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 23
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 24
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 25
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 26
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 27
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 28
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 29
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 30
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 31
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 32
dockercompose2-nginxserver-1  | 2023/02/07 05:02:07 [notice] 1#1: start worker process 33
```

- open up http://localhost:80/my.html



Mongodb and python without network (Let's skip out on docker compose for this part)

- First start up the monogo container :

```
docker run -dp 27017:27017 mongo # d is daemon p is port
```

- do the following to get ip address of the mongo container :

```
docker ps -a # copy id of the mongo container :
docker inspect <container_id> # Note the ip address
```

- Create a new folder and put the following files :
  Dockerfile :

```
FROM python
RUN apt-get update
RUN pip install pymong
COPY test.py test.py
CMD ["python","test.py"]
```

test.py :

```
from pymongo import MongoClient
host = MongoClient("<ip address noted before>")

db = host["sample_db"]
collection = db["sample_collection"]

sample_data = {"Name:":"P K Navin Shrinivas","SRN":"PES2UG20CS237"}
collection.insert_one(sample_data)
print('Inserted into the MongoDB database!')

rec_data = collection.find_one({"SRN":"<YOUR_SRN>"})
print("Fecthed from MongoDB: ",rec_data)
```

- build and run the pytho file :

```
docker build -t simplepython .
docker run simplepython #3c.png
```

```
➜  dockercompose3 git:(main) ✗ docker run simpleypython
Inserted into the MongoDB database!
Fecthed from MongoDB:  {'_id': ObjectId('63e1e3d91e9f051b9f53b35b'), 'Name:': 'P K Navin Shrinivas', 'SRN': '
PES2UG20CS237'}
➜  dockercompose3 git:(main) ✗
```

Mongo python using brigde networks :

- First up create a network

```
docker network create internalnet
```

- Start the mongo docker container :

```
docker run -dp 27017:27017 --network=internalnet --name=mongodb
mongo
```

- go back to the python file and convert the ip address to the "mongodb" (Which was the name of the docker container before)

```
docker ps -a #3d.png
docker build -t simplepython
docker run --network=internalnet simplepython #3e.png
```

```
➜  dockercompose3 git:(main) ✗ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS                     PORTS
                  NAMES
cc081cef0a36   simplepython   "python test.py"         3 minutes ago   Exited (0) 3 minutes ago
                  naughty_sutherland
7528e208a584   mongo          "docker-entrypoint.s…"   4 minutes ago   Up 4 minutes               0.0.0.0:27017→27017/tcp, ::
:27017→27017/tcp   mongodb
➜  dockercompose3 git:(main) ✗
```

```
➜  dockercompose3 git:(main) ✗ docker run --network=internalnet simplepython
Inserted into the MongoDB database!
Fecthed from MongoDB:  {'_id': ObjectId('63e1eca2b2712941f379d30a'), 'Name:': 'P K Navin Shrinivas', 'SRN': 'PES2UG20CS237'}
➜  dockercompose3 git:(main) ✗
[0] 0:zsh- 1:zsh* 2:zsh                                                    "zsh" 11:48 07-Feb-23
```

Docker compose :

- Make a new folder and copy over the files from previous folder to this.
- create a new file :
  docker-compose.yml :

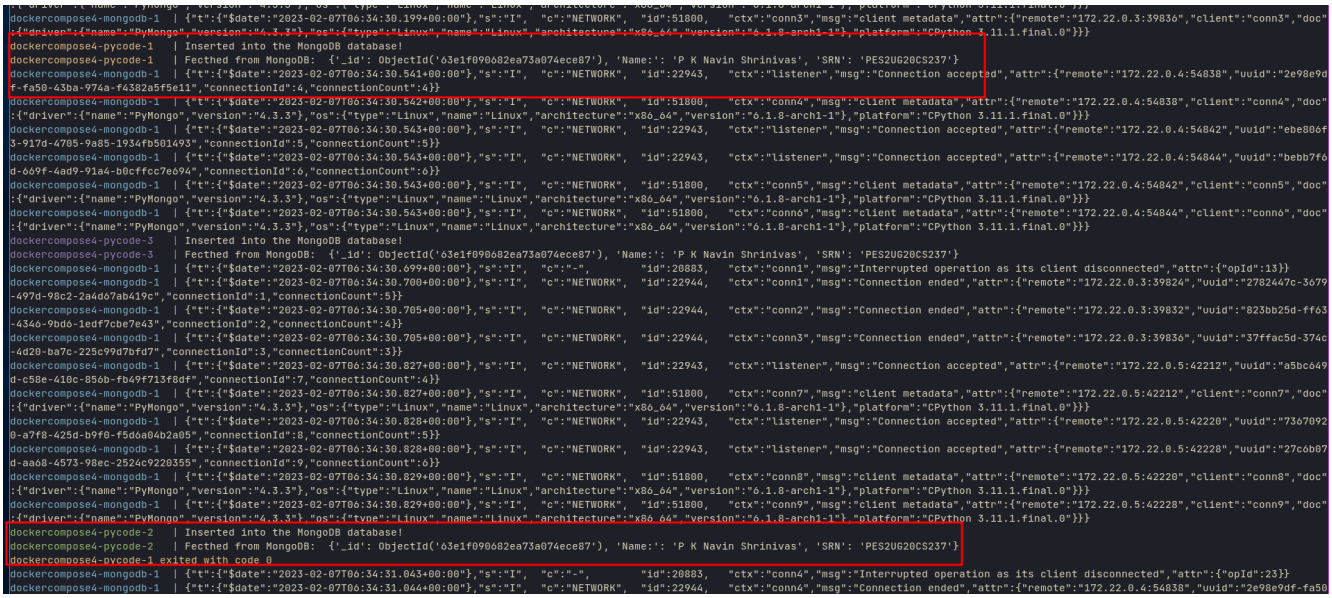
```
services:
  pycode:
    build: .
    links:
      - mongodb
  mongodb:
    image: mongo
    ports:
      - 27017
```

- Run the compose file :

```
docker compose up #4a.png
```

- docker compose with scaling (You should see read and write to mongo mutliple times)

```
docker compose --scale pycode=3 #4b.png
```


dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.199+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn3","msg":"client metadata","attr":{"remote":"172.22.0.3:39836","client":"conn3","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-pycode-1 | Inserted into the MongoDB database!
dockercompose4-pycode-1 | Fecthed from MongoDB: {'_id': ObjectId('63e1f090682ea73a074ece87'), 'Name:': 'P K Navin Shrinivas', 'SRN': 'PES2UG20CS237'}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.541+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.22.0.4:54838","uuid":"2e98e9d
f-fa50-43ba-974a-f4382a5f5e11","connectionId":4,"connectionCount":4}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.542+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn4","msg":"client metadata","attr":{"remote":"172.22.0.4:54838","client":"conn4","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.543+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.22.0.4:54842","uuid":"ebe806f
3-917d-4705-9a85-1934fb501493","connectionId":5,"connectionCount":5}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.543+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.22.0.4:54844","uuid":"bebb7f6
d-6697-4ad9-91a4-b0cffcc7e694","connectionId":6,"connectionCount":6}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.543+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn5","msg":"client metadata","attr":{"remote":"172.22.0.4:54842","client":"conn5","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.543+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn6","msg":"client metadata","attr":{"remote":"172.22.0.4:54844","client":"conn6","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-pycode-3 | Inserted into the MongoDB database!
dockercompose4-pycode-3 | Fecthed from MongoDB: {'_id': ObjectId('63e1f090682ea73a074ece87'), 'Name:': 'P K Navin Shrinivas', 'SRN': 'PES2UG20CS237'}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.699+00:00"},"s":"I", "c":"-", "id":20883, "ctx":"conn1","msg":"Interrupted operation as its client disconnected","attr":{"opId":13}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.700+00:00"},"s":"I", "c":"NETWORK", "id":22944, "ctx":"conn1","msg":"Connection ended","attr":{"remote":"172.22.0.3:39824","uuid":"2782447c-3679
-497d-98c2-2a4d67ab419c","connectionId":1,"connectionCount":5}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.705+00:00"},"s":"I", "c":"NETWORK", "id":22944, "ctx":"conn2","msg":"Connection ended","attr":{"remote":"172.22.0.3:39832","uuid":"823bb25d-ff63
-4346-9bd6-1edf7cbe7e43","connectionId":2,"connectionCount":4}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.705+00:00"},"s":"I", "c":"NETWORK", "id":22944, "ctx":"conn3","msg":"Connection ended","attr":{"remote":"172.22.0.3:39836","uuid":"37ffac5d-374c
-4d20-ba7c-225c99d7bfd7","connectionId":3,"connectionCount":3}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.827+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.22.0.5:42212","uuid":"a5bc649
d-c58e-410c-856b-fb49f713f8df","connectionId":7,"connectionCount":4}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.827+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn7","msg":"client metadata","attr":{"remote":"172.22.0.5:42212","client":"conn7","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.828+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.22.0.5:42220","uuid":"73d7092
0-a7f8-425d-b9f0-f5daa04b2a05","connectionId":8,"connectionCount":5}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.828+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.22.0.5:42228","uuid":"27c6b07
d-aa68-4573-98ec-2524c9220355","connectionId":9,"connectionCount":6}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.829+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn8","msg":"client metadata","attr":{"remote":"172.22.0.5:42220","client":"conn8","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:30.829+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn9","msg":"client metadata","attr":{"remote":"172.22.0.5:42228","client":"conn9","doc
:{"driver":{"name":"PyMongo","version":"4.3.3"},"os":{"type":"Linux","name":"Linux","architecture":"x86_64","version":"6.1.8-arch1-1"},"platform":"CPython 3.11.1.final.0"}}}
dockercompose4-pycode-2 | Inserted into the MongoDB database!
dockercompose4-pycode-2 | Fecthed from MongoDB: {'_id': ObjectId('63e1f090682ea73a074ece87'), 'Name:': 'P K Navin Shrinivas', 'SRN': 'PES2UG20CS237'}
dockercompose4-pycode-1 exited with code 0
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:31.043+00:00"},"s":"I", "c":"-", "id":20883, "ctx":"conn4","msg":"Interrupted operation as its client disconnected","attr":{"opId":23}}
dockercompose4-mongodb-1 | {"t":{"$date":"2023-02-07T06:34:31.044+00:00"},"s":"I", "c":"NETWORK", "id":22944, "ctx":"conn4","msg":"Connection ended","attr":{"remote":"172.22.0.4:54838","uuid":"2e98e9df-fa50