# CLOUD COMPUTING

# LAB – 2

Name : Sanmat Sanjayakumar Payagoudar

SRN : PES1UG20CS385

Section : G

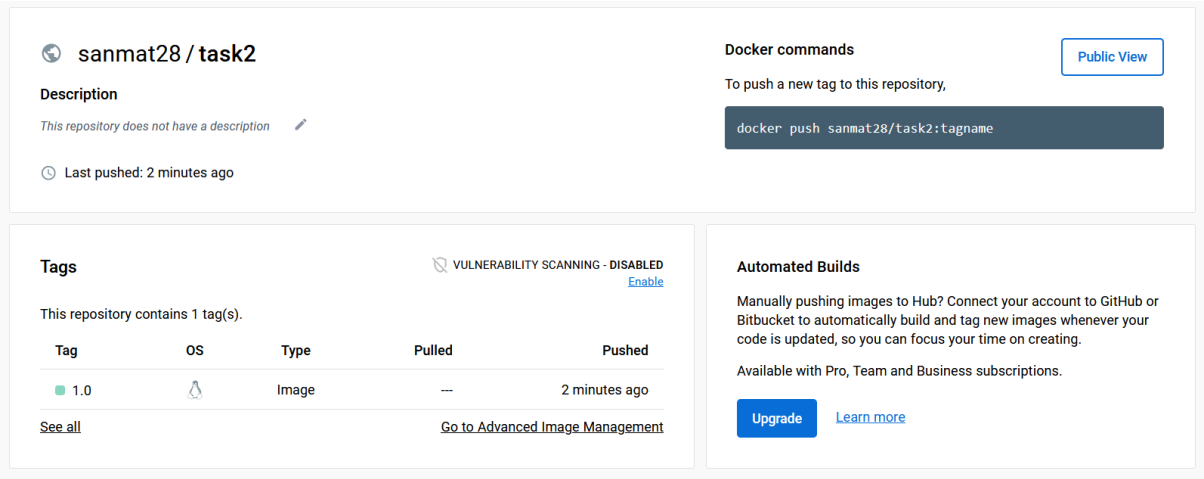
**1a)** Screenshot of running docker hello-world.

```
C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2>docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:aa0cc8055b82dc2509bed2e19b275c8f463506616377219d9642221ab53cf9fe
Status: Downloaded newer image for hello-world:latest

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/
```

## **2a)** Screenshot of C Program successfully run inside the container.

```
C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2\task2>docker run task2
Running this inside a container !
My SRN is <PES1UG20CS385>
```

## **2b)** Screenshot of the image pushed to Dockerhub.




## **3a)** Screenshot of docker container running nginx.

```
C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2\task3\task3-nginx>docker run -p 80:80 task3
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/02/09 14:38:31 [notice] 1#1: using the "epoll" event method
2023/02/09 14:38:31 [notice] 1#1: nginx/1.23.3
2023/02/09 14:38:31 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/02/09 14:38:31 [notice] 1#1: OS: Linux 5.15.79.1-microsoft-standard-WSL2
2023/02/09 14:38:31 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/02/09 14:38:31 [notice] 1#1: start worker processes
2023/02/09 14:38:31 [notice] 1#1: start worker process 29
2023/02/09 14:38:31 [notice] 1#1: start worker process 30
2023/02/09 14:38:31 [notice] 1#1: start worker process 31
2023/02/09 14:38:31 [notice] 1#1: start worker process 32
2023/02/09 14:38:31 [notice] 1#1: start worker process 33
2023/02/09 14:38:31 [notice] 1#1: start worker process 34
2023/02/09 14:38:31 [notice] 1#1: start worker process 35
2023/02/09 14:38:31 [notice] 1#1: start worker process 36
2023/02/09 14:38:31 [notice] 1#1: start worker process 37
2023/02/09 14:38:31 [notice] 1#1: start worker process 38
2023/02/09 14:38:31 [notice] 1#1: start worker process 39
2023/02/09 14:38:31 [notice] 1#1: start worker process 40
```

**3b)** Sample.html showing the web page on the browser.

---

**My SRN is PES1UG20CS385**

**I am running a nginx container!**

**3c)** Screenshot of python application successfully writing and reading from the MongoDB database

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

C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2\task3\task3-pymongo>docker run task3
Inserted into the MongoDB database!
Fecthed from MongoDB:  {'_id': ObjectId('63e5084093959a2190922275'), 'Name:': '<Sanmat>', 'SRN': '<PES1UG20CS385>'}
```

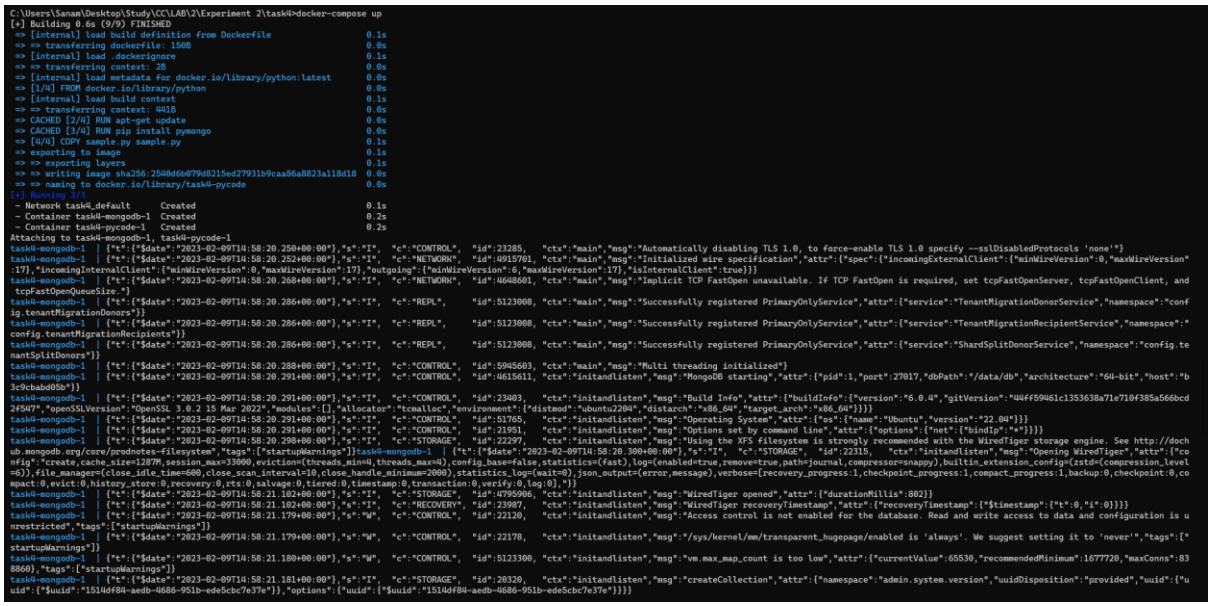
**3d)** Screenshot showing mongodb being run within the network(docker command has to be clearly highlighted)

```
C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2\task3\task3-pymongo>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS         PORTS                      NAMES
fb1a5b2363d9   mongo:latest   "docker-entrypoint.s…"   11 seconds ago   Up 8 seconds   0.0.0.0:27017->27017/tcp   mongodb
```

**3e)** Screenshot showing python file being run within the network and successfully writing and reading from MongoDB (docker command has to be clearly highlighted)

```
C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2\task3\task3-pymongo>docker build -t task3 .
[+] Building 0.4s (9/9) FINISHED
 => [internal] load build definition from Dockerfile                    0.0s
 => => transferring dockerfile: 31B                                     0.0s
 => [internal] load .dockerignore                                       0.0s
 => => transferring context: 2B                                         0.0s
 => [internal] load metadata for docker.io/library/python:latest        0.0s
 => [1/4] FROM docker.io/library/python                                 0.0s
 => [internal] load build context                                       0.0s
 => => transferring context: 453B                                       0.0s
 => CACHED [2/4] RUN apt-get update                                     0.0s
 => CACHED [3/4] RUN pip install pymongo                                0.0s
 => [4/4] COPY sample.py sample.py                                      0.0s
 => exporting to image                                                  0.1s
 => => exporting layers                                                 0.1s
 => => writing image sha256:8a6adf705c7e54c62475bf30281eafb62ad2a0dc5    0.0s
 => => naming to docker.io/library/task3                                0.0s

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

C:\Users\Sanam\Desktop\Study\CC\LAB\2\Experiment 2\task3\task3-pymongo>docker run --network=my-bridge-network task3
Inserted into the MongoDB database!
Fecthed from MongoDB:  {'_id': ObjectId('63e5098ab7d07ab077162c7e'), 'Name:': '<Sanmat>', 'SRN': '<PES1UG20CS385>'}
```

**4a)** Screenshot of python-mongodb application running as a docker-compose application(logs of the application)



**4b)** Screenshot of 3 python application writes and reads from MongoDB after scaling the python application.