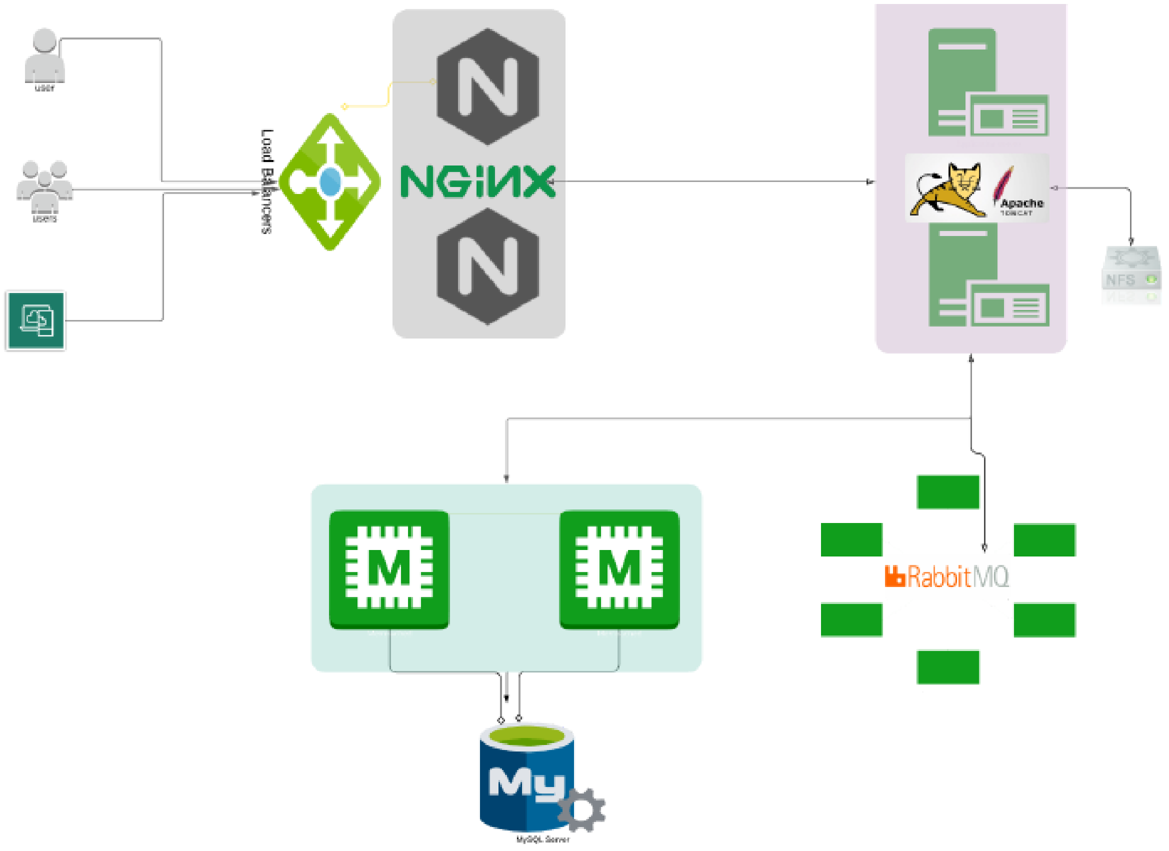
** Containerize services of a Java App architecture **

In this Project we will build a complete architecture for Java App using container technology.



- **Tools we need for this project.**

1) **VMware Workstation :** (VMware Workstation is a Desktop Hypervisor products which let users run virtual machines, containers, and Kubernetes clusters. VMware Workstation is a virtual machine software that is used for x86 and x86-64 computers to run multiple operating systems over a single physical host computer. Each virtual machine can run a single instance of any operating system (Microsoft, Linux, etc.).

2) **Docker :** (Docker is an open source platform that enables developers to build, deploy, run, update and manage containers).

    - **Docker tools:**

        I. DockerFile :- (Every Docker container starts with a simple text file containing instructions for how to build the Docker container image. DockerFile automates the process of Docker image creation. It's essentially a list of command-line interface (CLI) instructions that Docker Engine will run in order to assemble the image).

        II. Docker images :- (Docker images contain executable application source code as well as all the tools, libraries, and dependencies that the application code needs to run as a container).

III. Docker containers :- (Docker containers are the live, running instances of Docker images. While Docker images are read-only files, containers are life executable content).

IV. Docker Compose :- (Docker Compose used to manage multi-container applications, where all containers run on the same Docker host. Docker Compose creates a YAML (.YML) file that specifies which services are included in the application and can deploy and run containers with a single command).

3) **Maven :** (Maven is a popular open-source build tool developed by the Apache Group to build, publish, and deploy several projects at once for better project management. Maven is written in Java and is used to build projects written in C#, Scala, Ruby, Java, etc).

4) **Git :** (Git is a version control system used for tracking changes in computer files. It is generally used for source code management in software development).

❖ Steps:

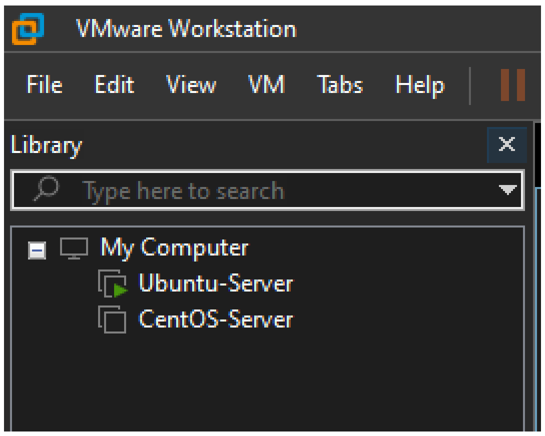
- Create a Virtual Machines using **VMware Workstation** based on any Linux OS distribution "Ubuntu, CentOS" .
- Install **Docker**, **Maven** and **Git** in the VM.
- Clone our App source code from github using **Git**, then build it using **Maven.**

- Containerize services of a complete Java App architecture:-

1. Pull the base App image for each service we need to build our App architecture (**Nginx**, **Tomcat**, **RabbitMQ**, **Memcached**, **MariaDB**).
2. Write Dockerfile to customize each App image, to be ready to create new image for each App service.
3. Test if all new images running successfully, then push them to your Docker Hub Repository.
4. Write Compose file to run and manage all services containers.

$~ Lab Setup ~$

- **Create Virtual Machine:**



➢ Installing Docker :

# Login to your Ubuntu machine & then run the following commands.

```
sudo apt update
sudo apt install curl
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/trusted.gpg.d/docker.gpg
sudo add-apt-repository "deb [arch=$(dpkg --print-architecture)]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update
sudo apt -y install lsb-release gnupg apt-transport-https ca-certificates curl software-properties-
common
sudo apt -y install  docker-ce  docker-ce-cli  containerd.io  docker-compose-plugin docker-registry
sudo usermod -aG docker $USER
newgrp docker
```

➢ Installing Maven and Git :

```
sudo apt install -y maven git
```

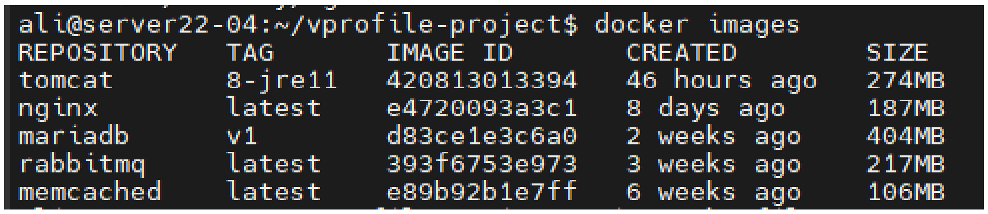
➢ Clone and build our App source code :

```
git clone -b main https://github.com/hkhcoder/vprofile-project.git
cd vprofile-project
mvn install
```

➢ Containerize services of a complete Java App architecture.

#1 Pull the base App image for each service.

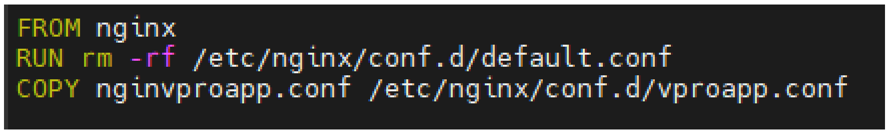
- Nginx image >	docker pull nginx
- Tomcat image >	docker pull tomcat: 8-jre11
- RabbitMQ image >	docker pull rabbitmq
- Memcached image >	docker pull memcached
- MariaDB image >	docker pull mariadb



```
ali@server22-04:~/vprofile-project$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
tomcat       8-jre11   420813013394   46 hours ago    274MB
nginx        latest    e4720093a3c1   8 days ago      187MB
mariadb      v1        d83ce1e3c6a0   2 weeks ago     404MB
rabbitmq     latest    393f6753e973   3 weeks ago     217MB
memcached    latest    e89b92b1e7ff   6 weeks ago     106MB
```

#2 Write Dockerfile to customize each App image to build new image for each App service.

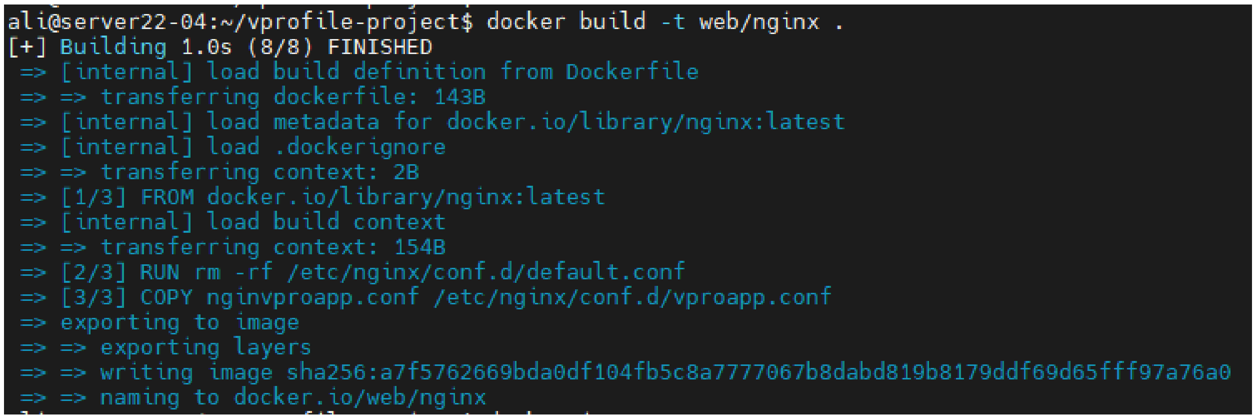
- Nginx Dockerfile :>



```
FROM nginx
RUN rm -rf /etc/nginx/conf.d/default.conf
COPY nginvproapp.conf /etc/nginx/conf.d/vproapp.conf
```

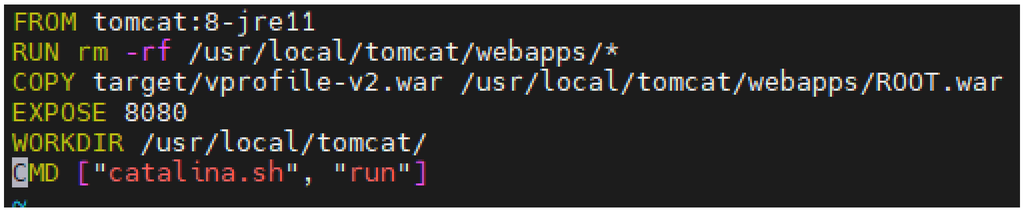
# Now run this command to build new image with this setting .

```
docker build -t  web/nginx  .
```



```
ali@server22-04:~/vprofile-project$ docker build -t web/nginx .
[+] Building 1.0s (8/8) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 143B
 => [internal] load metadata for docker.io/library/nginx:latest
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/3] FROM docker.io/library/nginx:latest
 => [internal] load build context
 => => transferring context: 154B
 => [2/3] RUN rm -rf /etc/nginx/conf.d/default.conf
 => [3/3] COPY nginvproapp.conf /etc/nginx/conf.d/vproapp.conf
 => exporting to image
 => => exporting layers
 => => writing image sha256:a7f5762669bda0df104fb5c8a7777067b8dabd819b8179ddf69d65fff97a76a0
 => => naming to docker.io/web/nginx
```

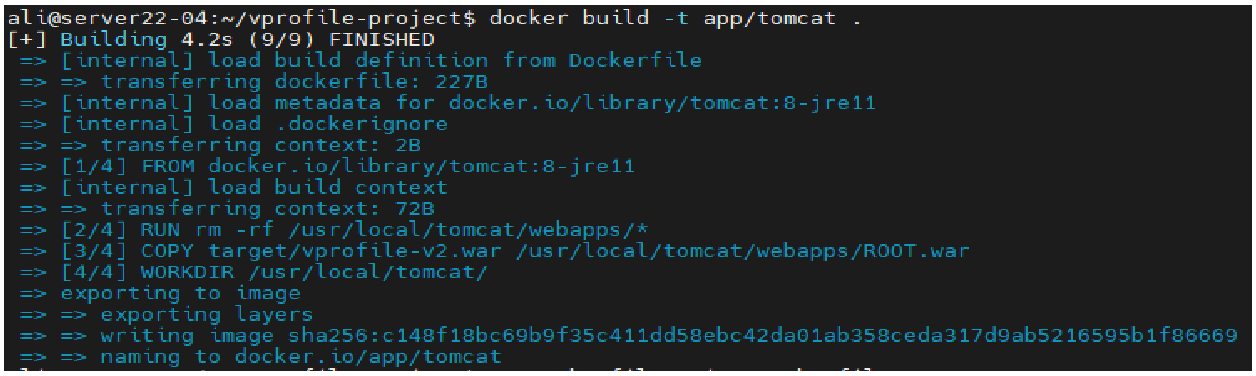
- Tomcat Dockerfile :>



```
FROM tomcat:8-jre11
RUN rm -rf /usr/local/tomcat/webapps/*
COPY target/vprofile-v2.war /usr/local/tomcat/webapps/ROOT.war
EXPOSE 8080
WORKDIR /usr/local/tomcat/
CMD ["catalina.sh", "run"]
```

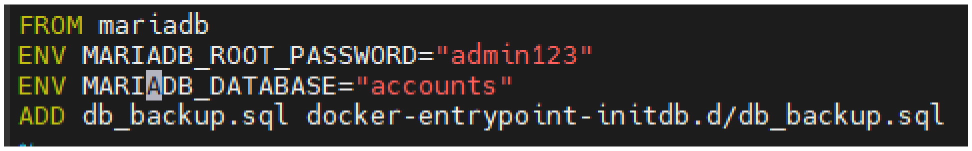
# Now run this command to build new image with this setting .

```
docker build -t  app/tomcat  .
```



```
ali@server22-04:~/vprofile-project$ docker build -t app/tomcat .
[+] Building 4.2s (9/9) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 227B
 => [internal] load metadata for docker.io/library/tomcat:8-jre11
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/4] FROM docker.io/library/tomcat:8-jre11
 => [internal] load build context
 => => transferring context: 72B
 => [2/4] RUN rm -rf /usr/local/tomcat/webapps/*
 => [3/4] COPY target/vprofile-v2.war /usr/local/tomcat/webapps/ROOT.war
 => [4/4] WORKDIR /usr/local/tomcat/
 => exporting to image
 => => exporting layers
 => => writing image sha256:c148f18bc69b9f35c411dd58ebc42da01ab358ceda317d9ab5216595b1f86669
 => => naming to docker.io/app/tomcat
```

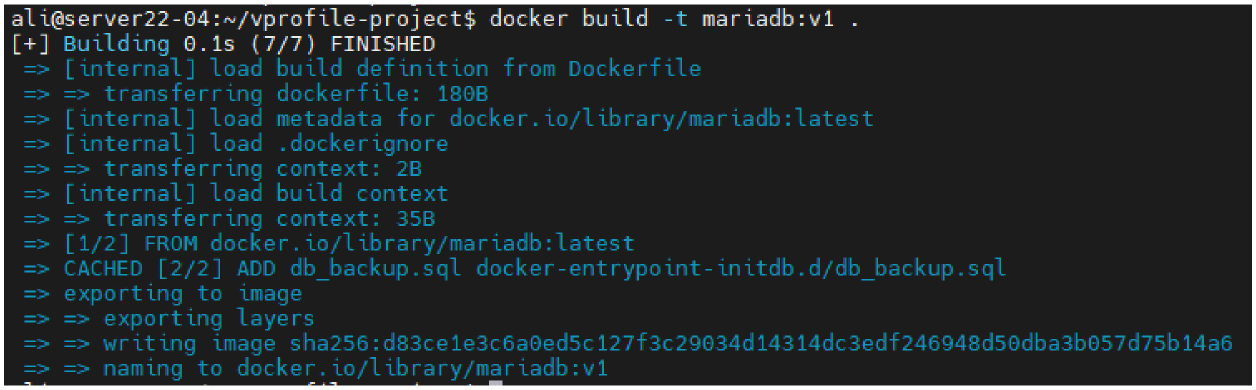
- RabbitMQ Dockerfile :> we will use RabbitMQ base image without do any edit on it.
- Memcached Dockerfile :> we will also use RabbitMQ base image without do any edit on it.
- MariaDB Dockerfile :>



```
FROM mariadb
ENV MARIADB_ROOT_PASSWORD="admin123"
ENV MARIADB_DATABASE="accounts"
ADD db_backup.sql docker-entrypoint-initdb.d/db_backup.sql
```

# Now run this command to build new image with this setting .

docker build -t mariadb:v1 .



```
ali@server22-04:~/vprofile-project$ docker build -t mariadb:v1 .
[+] Building 0.1s (7/7) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 180B
 => [internal] load metadata for docker.io/library/mariadb:latest
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build context
 => => transferring context: 35B
 => [1/2] FROM docker.io/library/mariadb:latest
 => CACHED [2/2] ADD db_backup.sql docker-entrypoint-initdb.d/db_backup.sql
 => exporting to image
 => => exporting layers
 => => writing image sha256:d83ce1e3c6a0ed5c127f3c29034d14314dc3edf246948d50dba3b057d75b14a6
 => => naming to docker.io/library/mariadb:v1
```

#3 Test if all ne images running successfully, then push them to your Docker Hub Repository.

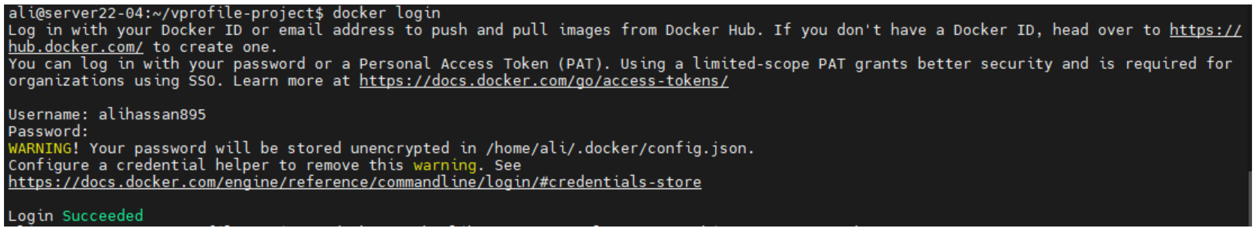
- Nginx new image “web/nginx” >

# we will create a container using this new image.

docker run -d --name web_test -p 8080:80 web/nginx

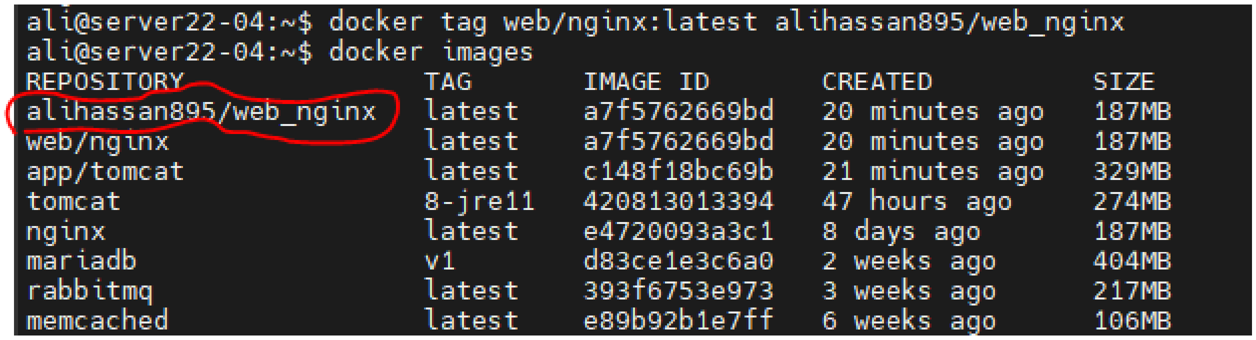
# To push an image to your Docker Hub account follow this steps :-

I. Login to your Docker Hub account using this command :
 `docker login`



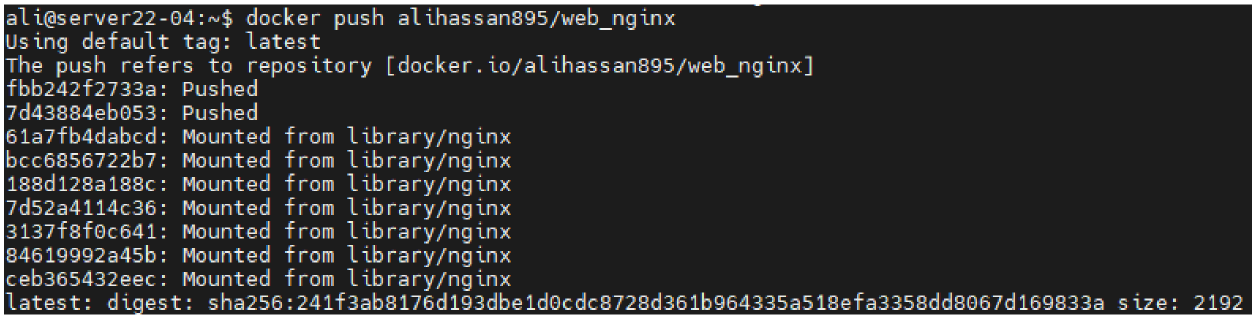
```
ali@server22-04:~/vprofile-project$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://
hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and is required for
organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/

Username: alihassan895
Password:
WARNING! Your password will be stored unencrypted in /home/ali/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

II. Use the `docker tag` command to give your image a new name in this form "Docker hub username/image name :tag" .



```
ali@server22-04:~$ docker tag web/nginx:latest alihassan895/web_nginx
ali@server22-04:~$ docker images
REPOSITORY               TAG       IMAGE ID       CREATED          SIZE
alihassan895/web_nginx   latest    a7f5762669bd   20 minutes ago   187MB
web/nginx                latest    a7f5762669bd   20 minutes ago   187MB
app/tomcat               latest    c148f18bc69b   21 minutes ago   329MB
tomcat                   8-jre11   420813013394   47 hours ago     274MB
nginx                    latest    e4720093a3c1   8 days ago       187MB
mariadb                  v1        d83ce1e3c6a0   2 weeks ago      404MB
rabbitmq                 latest    393f6753e973   3 weeks ago      217MB
memcached                latest    e89b92b1e7ff   6 weeks ago      106MB
```

III. Use the `docker push` command to push the image to your Docker Hub account.

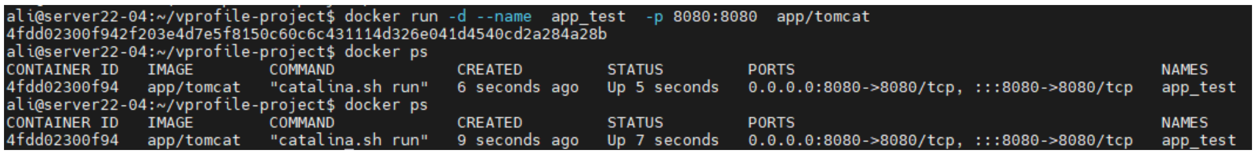


```
ali@server22-04:~$ docker push alihassan895/web_nginx
Using default tag: latest
The push refers to repository [docker.io/alihassan895/web_nginx]
fbb242f2733a: Pushed
7d43884eb053: Pushed
61a7fb4dabcd: Mounted from library/nginx
bcc6856722b7: Mounted from library/nginx
188d128a188c: Mounted from library/nginx
7d52a4114c36: Mounted from library/nginx
3137f8f0c641: Mounted from library/nginx
84619992a45b: Mounted from library/nginx
ceb365432eec: Mounted from library/nginx
latest: digest: sha256:241f3ab8176d193dbe1d0cdc8728d361b964335a518efa3358dd8067d169833a size: 2192
```

- Tomcat new image "app/tomcat" >

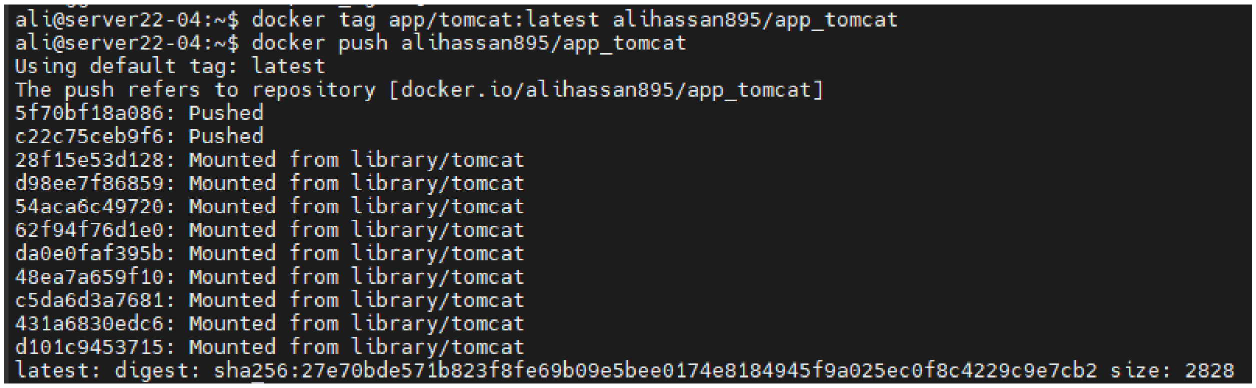
# we will create a container using this new image.

`docker run -d --name  app_test  -p 8080:8080  app/tomcat`



```
ali@server22-04:~/vprofile-project$ docker run -d --name  app_test  -p 8080:8080  app/tomcat
4fdd02300f942f203e4d7e5f8150c60c6c431114d326e041d4540cd2a284a28b
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE        COMMAND             CREATED         STATUS         PORTS                                       NAMES
4fdd02300f94   app/tomcat   "catalina.sh run"   6 seconds ago   Up 5 seconds   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   app_test
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE        COMMAND             CREATED         STATUS         PORTS                                       NAMES
4fdd02300f94   app/tomcat   "catalina.sh run"   9 seconds ago   Up 7 seconds   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   app_test
```

# Push the new image to your Docker Hub account :-

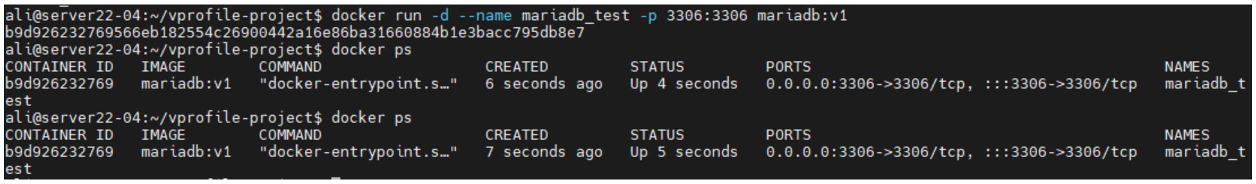


```
ali@server22-04:~$ docker tag app/tomcat:latest alihassan895/app_tomcat
ali@server22-04:~$ docker push alihassan895/app_tomcat
Using default tag: latest
The push refers to repository [docker.io/alihassan895/app_tomcat]
5f70bf18a086: Pushed
c22c75ceb9f6: Pushed
28f15e53d128: Mounted from library/tomcat
d98ee7f86859: Mounted from library/tomcat
54aca6c49720: Mounted from library/tomcat
62f94f76d1e0: Mounted from library/tomcat
da0e0faf395b: Mounted from library/tomcat
48ea7a659f10: Mounted from library/tomcat
c5da6d3a7681: Mounted from library/tomcat
431a6830edc6: Mounted from library/tomcat
d101c9453715: Mounted from library/tomcat
latest: digest: sha256:27e70bde571b823f8fe69b09e5bee0174e8184945f9a025ec0f8c4229c9e7cb2 size: 2828
```

- Mariadb new image “mariadb:v1” >

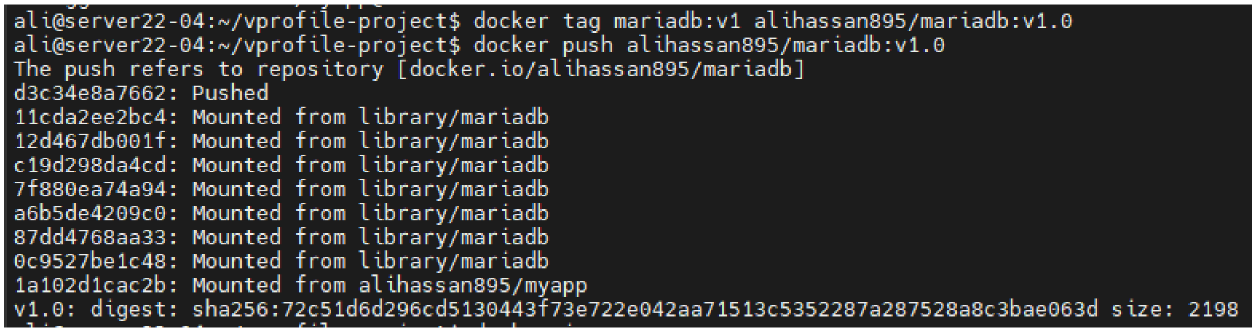
# we will create a container using this new image.

docker run -d --name mariadb_test -p 3306:3306 mariadb:v1



```
ali@server22-04:~/vprofile-project$ docker run -d --name mariadb_test -p 3306:3306 mariadb:v1
b9d926232769566eb182554c26900442a16e86ba31660884b1e3bacc795db8e7
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE        COMMAND                  CREATED         STATUS         PORTS                                       NAMES
b9d926232769   mariadb:v1   "docker-entrypoint.s…"   6 seconds ago   Up 4 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp   mariadb_t
est
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE        COMMAND                  CREATED         STATUS         PORTS                                       NAMES
b9d926232769   mariadb:v1   "docker-entrypoint.s…"   7 seconds ago   Up 5 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp   mariadb_t
est
```

# Push the new image to your Docker Hub account :-

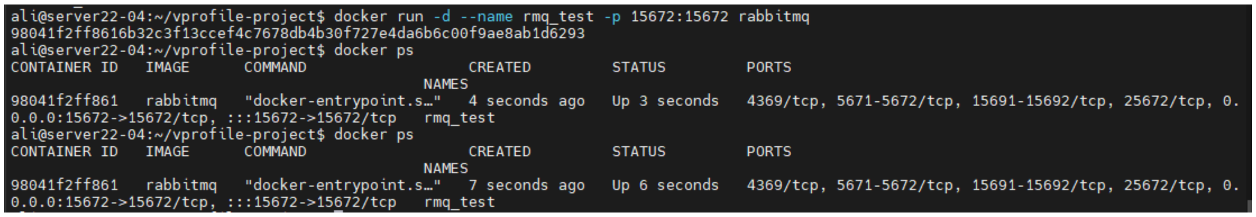


```
ali@server22-04:~/vprofile-project$ docker tag mariadb:v1 alihassan895/mariadb:v1.0
ali@server22-04:~/vprofile-project$ docker push alihassan895/mariadb:v1.0
The push refers to repository [docker.io/alihassan895/mariadb]
d3c34e8a7662: Pushed
11cda2ee2bc4: Mounted from library/mariadb
12d467db001f: Mounted from library/mariadb
c19d298da4cd: Mounted from library/mariadb
7f880ea74a94: Mounted from library/mariadb
a6b5de4209c0: Mounted from library/mariadb
87dd4768aa33: Mounted from library/mariadb
0c9527be1c48: Mounted from library/mariadb
1a102d1cac2b: Mounted from alihassan895/myapp
v1.0: digest: sha256:72c51d6d296cd5130443f73e722e042aa71513c5352287a287528a8c3bae063d size: 2198
```

- RabbitMQ base image “rabbitmq” >

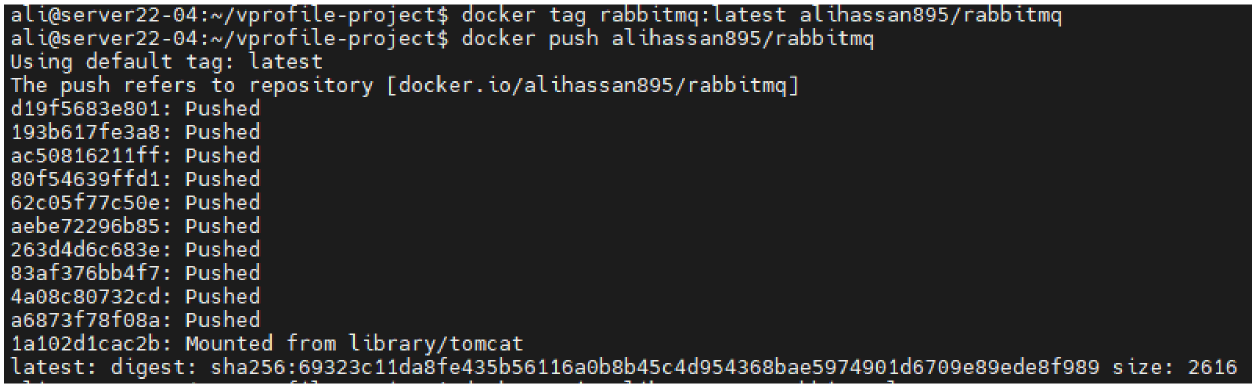
# we will create a container using this image.

docker run -d --name rmq_test -p 15672:15672 rabbitmq



```
ali@server22-04:~/vprofile-project$ docker run -d --name rmq_test -p 15672:15672 rabbitmq
98041f2ff8616b32c3f13ccef4c7678db4b30f727e4da6b6c00f9ae8ab1d6293
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS         PORTS
                                   NAMES
98041f2ff861   rabbitmq   "docker-entrypoint.s…"   4 seconds ago   Up 3 seconds   4369/tcp, 5671-5672/tcp, 15691-15692/tcp, 25672/tcp, 0.
0.0.0:15672->15672/tcp, :::15672->15672/tcp   rmq_test
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS         PORTS
                                   NAMES
98041f2ff861   rabbitmq   "docker-entrypoint.s…"   7 seconds ago   Up 6 seconds   4369/tcp, 5671-5672/tcp, 15691-15692/tcp, 25672/tcp, 0.
0.0.0:15672->15672/tcp, :::15672->15672/tcp   rmq_test
```

# Push the new image to your Docker Hub account :-

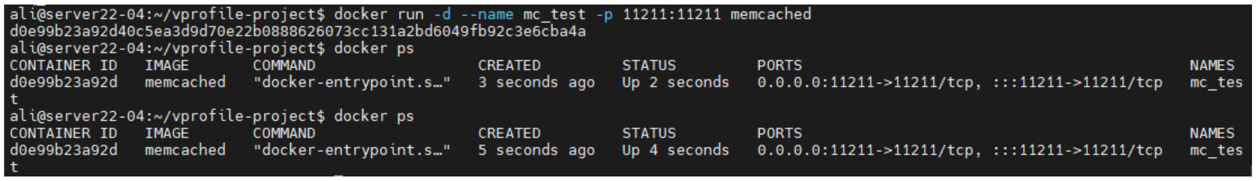


```
ali@server22-04:~/vprofile-project$ docker tag rabbitmq:latest alihassan895/rabbitmq
ali@server22-04:~/vprofile-project$ docker push alihassan895/rabbitmq
Using default tag: latest
The push refers to repository [docker.io/alihassan895/rabbitmq]
d19f5683e801: Pushed
193b617fe3a8: Pushed
ac50816211ff: Pushed
80f54639ffd1: Pushed
62c05f77c50e: Pushed
aebe72296b85: Pushed
263d4d6c683e: Pushed
83af376bb4f7: Pushed
4a08c80732cd: Pushed
a6873f78f08a: Pushed
1a102d1cac2b: Mounted from library/tomcat
latest: digest: sha256:69323c11da8fe435b56116a0b8b45c4d954368bae5974901d6709e89ede8f989 size: 2616
```

- Memcached base image “memcached” >

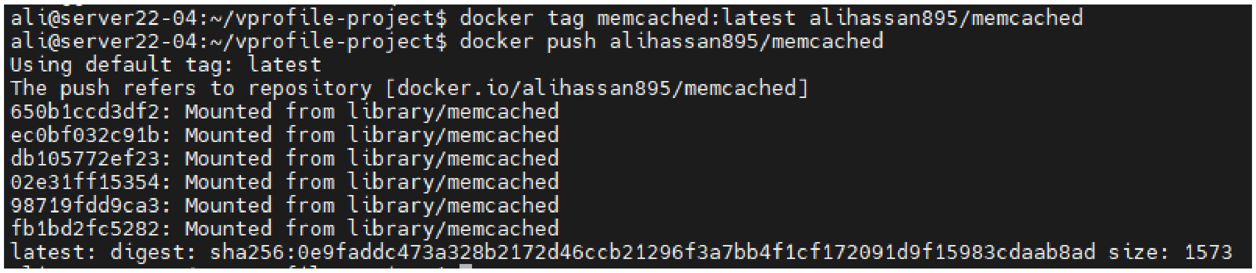
# we will create a container using this image.

docker run -d --name mc_test -p 11211:11211 memcached



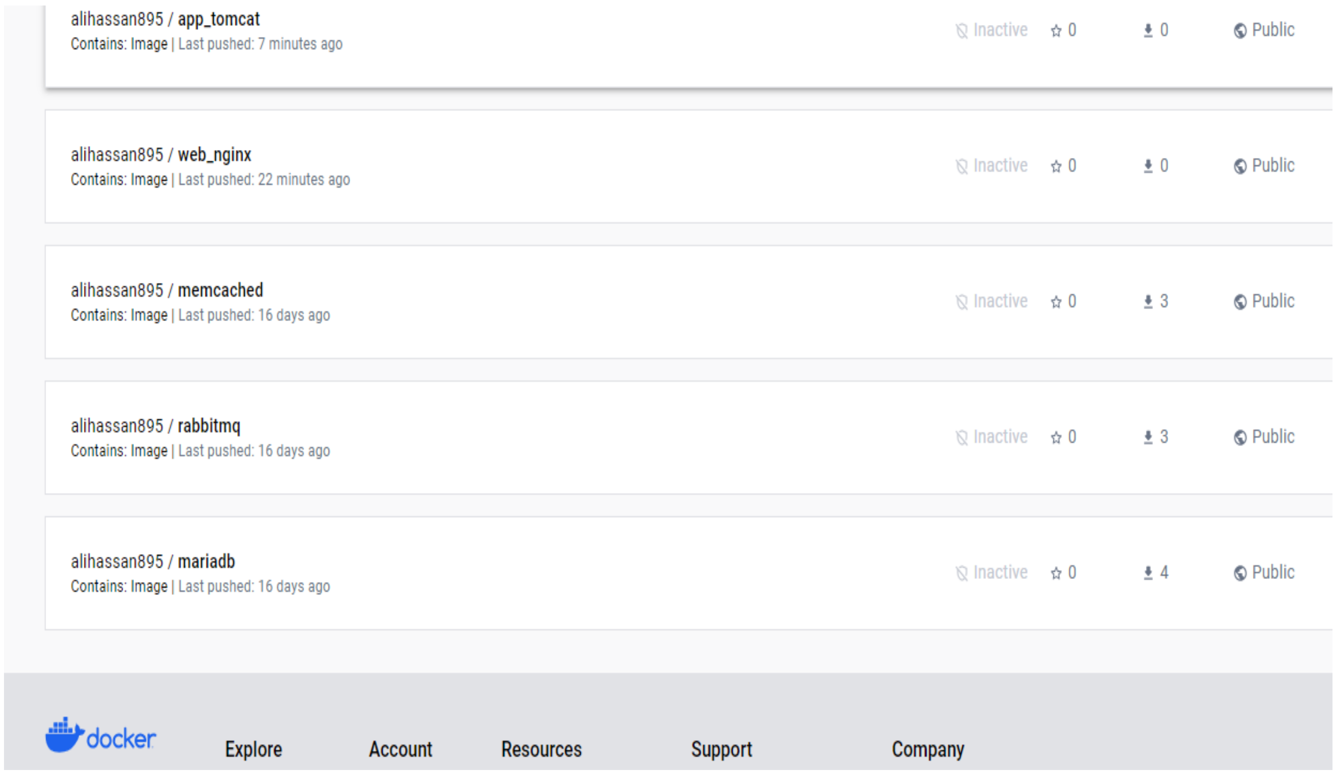
```
ali@server22-04:~/vprofile-project$ docker run -d --name mc_test -p 11211:11211 memcached
d0e99b23a92d40c5ea3d9d70e22b0888626073cc131a2bd6049fb92c3e6cba4a
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED         STATUS         PORTS                                           NAMES
d0e99b23a92d   memcached   "docker-entrypoint.s…"   3 seconds ago   Up 2 seconds   0.0.0.0:11211->11211/tcp, :::11211->11211/tcp   mc_tes
t
ali@server22-04:~/vprofile-project$ docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED         STATUS         PORTS                                           NAMES
d0e99b23a92d   memcached   "docker-entrypoint.s…"   5 seconds ago   Up 4 seconds   0.0.0.0:11211->11211/tcp, :::11211->11211/tcp   mc_tes
t
```

# Push the new image to your Docker Hub account :-



```
ali@server22-04:~/vprofile-project$ docker tag memcached:latest alihassan895/memcached
ali@server22-04:~/vprofile-project$ docker push alihassan895/memcached
Using default tag: latest
The push refers to repository [docker.io/alihassan895/memcached]
650b1ccd3df2: Mounted from library/memcached
ec0bf032c91b: Mounted from library/memcached
db105772ef23: Mounted from library/memcached
02e31ff15354: Mounted from library/memcached
98719fdd9ca3: Mounted from library/memcached
fb1bd2fc5282: Mounted from library/memcached
latest: digest: sha256:0e9faddc473a328b2172d46ccb21296f3a7bb4f1cf172091d9f15983cdaab8ad size: 1573
```

# verify that all images successfully uploaded to Docker Hub.



alihassan895 / app_tomcat
Contains: Image | Last pushed: 7 minutes ago
Inactive 0 0 Public

alihassan895 / web_nginx
Contains: Image | Last pushed: 22 minutes ago
Inactive 0 0 Public

alihassan895 / memcached
Contains: Image | Last pushed: 16 days ago
Inactive 0 3 Public

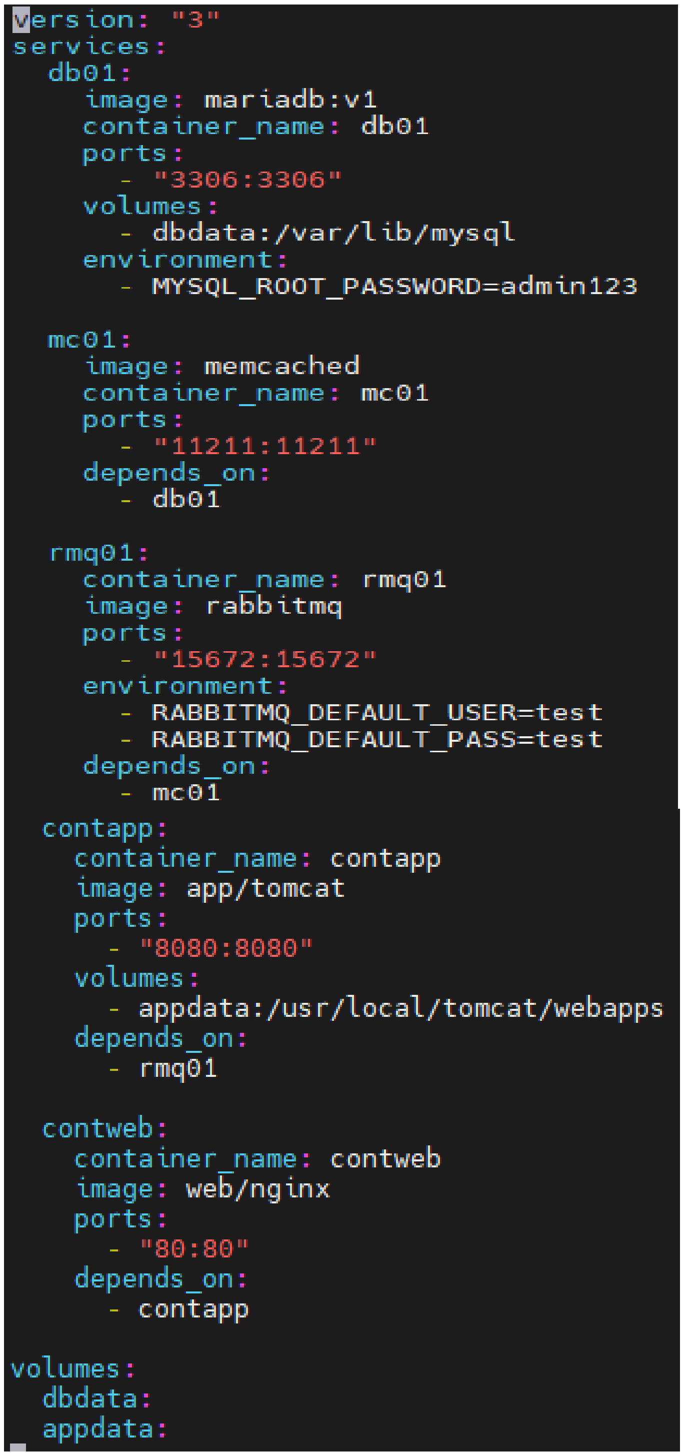
alihassan895 / rabbitmq
Contains: Image | Last pushed: 16 days ago
Inactive 0 3 Public

alihassan895 / mariadb
Contains: Image | Last pushed: 16 days ago
Inactive 0 4 Public

#4 Write Compose file to run and manage all services containers.

# create new file.

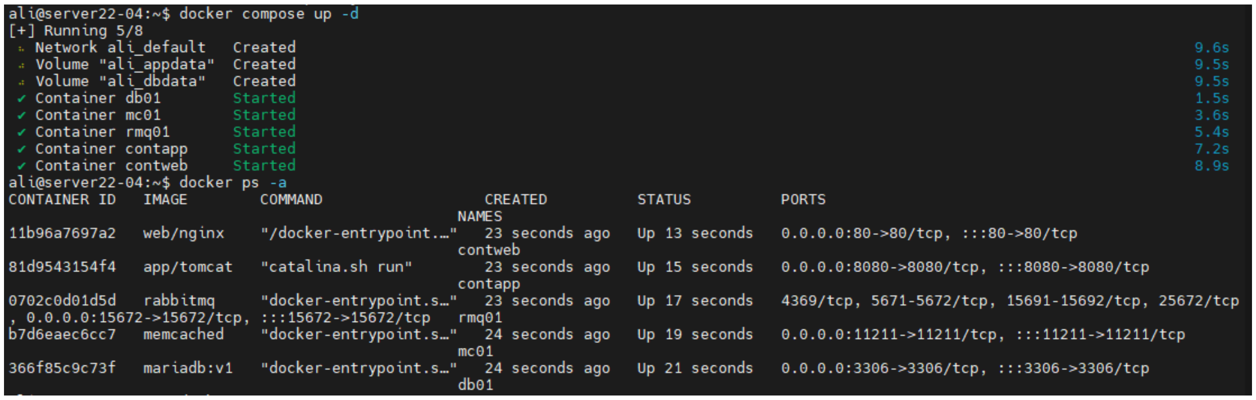
vim docker-compose.yml



```
version: "3"
services:
  db01:
    image: mariadb:v1
    container_name: db01
    ports:
      - "3306:3306"
    volumes:
      - dbdata:/var/lib/mysql
    environment:
      - MYSQL_ROOT_PASSWORD=admin123

  mc01:
    image: memcached
    container_name: mc01
    ports:
      - "11211:11211"
    depends_on:
      - db01

  rmq01:
    container_name: rmq01
    image: rabbitmq
    ports:
      - "15672:15672"
    environment:
      - RABBITMQ_DEFAULT_USER=test
      - RABBITMQ_DEFAULT_PASS=test
    depends_on:
      - mc01
  contapp:
    container_name: contapp
    image: app/tomcat
    ports:
      - "8080:8080"
    volumes:
      - appdata:/usr/local/tomcat/webapps
    depends_on:
      - rmq01

  contweb:
    container_name: contweb
    image: web/nginx
    ports:
      - "80:80"
    depends_on:
      - contapp

volumes:
  dbdata:
  appdata:
```

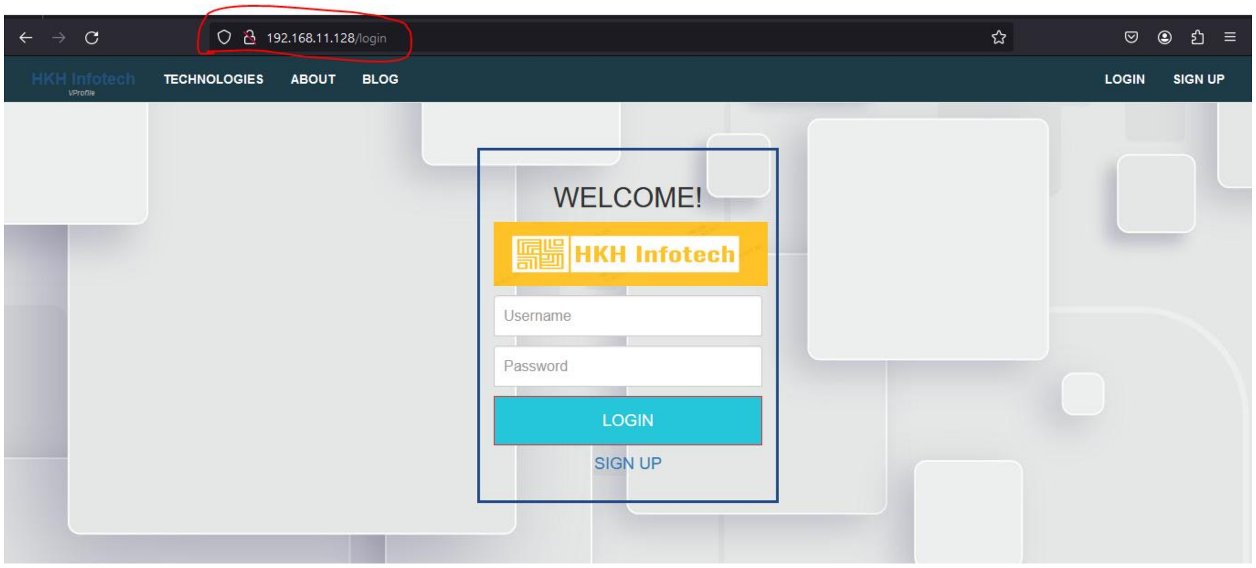
Now run the Compose file using this command.

docker compose up -d

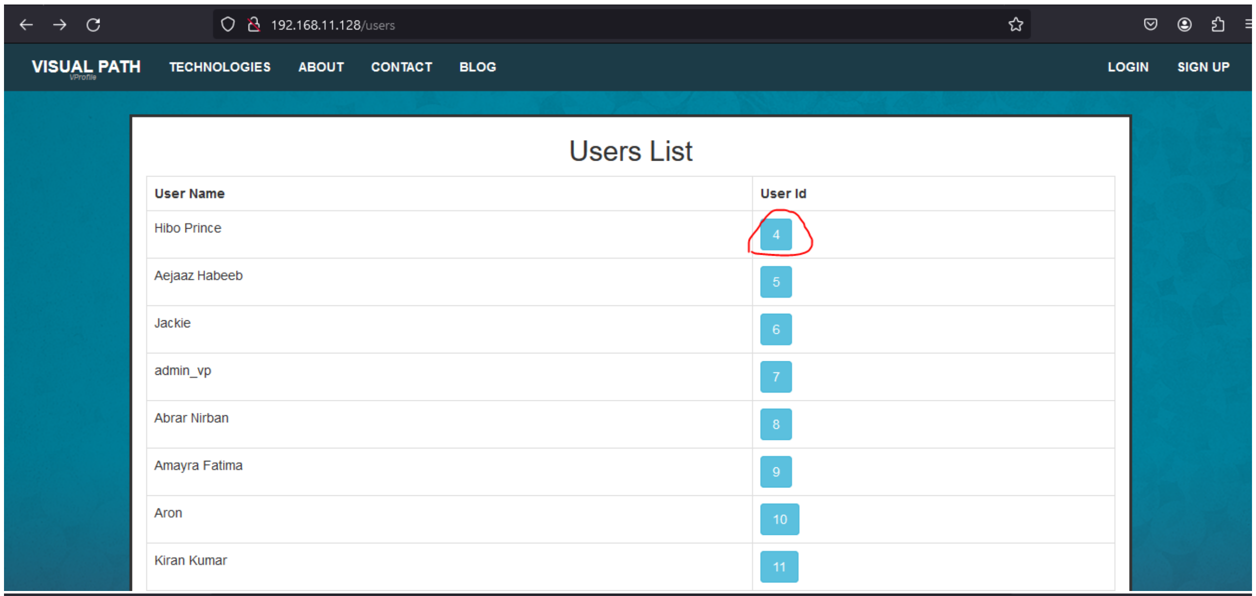


```
ali@server22-04:~$ docker compose up -d
[+] Running 5/8
 ∴ Network ali_default   Created                                                                                9.6s
 ∴ Volume "ali_appdata"  Created                                                                                9.5s
 ∴ Volume "ali_dbdata"   Created                                                                                9.5s
 ✓ Container db01        Started                                                                                1.5s
 ✓ Container mc01        Started                                                                                3.6s
 ✓ Container rmq01       Started                                                                                5.4s
 ✓ Container contapp     Started                                                                                7.2s
 ✓ Container contweb     Started                                                                                8.9s
ali@server22-04:~$ docker ps -a
CONTAINER ID   IMAGE        COMMAND                   CREATED          STATUS          PORTS
                                                    NAMES
11b96a7697a2   web/nginx    "/docker-entrypoint.…"   23 seconds ago   Up 13 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp
                                                    contweb
81d9543154f4   app/tomcat   "catalina.sh run"         23 seconds ago   Up 15 seconds   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp
                                                    contapp
0702c0d01d5d   rabbitmq     "docker-entrypoint.s…"   23 seconds ago   Up 17 seconds   4369/tcp, 5671-5672/tcp, 15691-15692/tcp, 25672/tcp
, 0.0.0.0:15672->15672/tcp, :::15672->15672/tcp    rmq01
b7d6eaec6cc7   memcached    "docker-entrypoint.s…"   24 seconds ago   Up 19 seconds   0.0.0.0:11211->11211/tcp, :::11211->11211/tcp
                                                    mc01
366f85c9c73f   mariadb:v1   "docker-entrypoint.s…"   24 seconds ago   Up 21 seconds   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp
                                                    db01
```

- We can access our App from any browser by typing our server IP “host VM” (192.168.11.128).

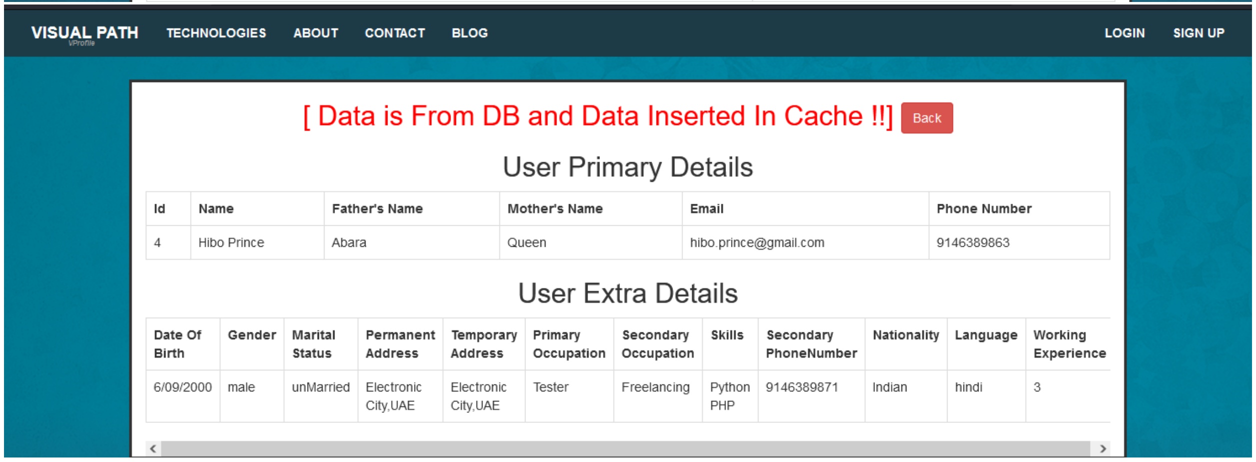


Check **Memcached** service by access any user.



Users List

User Name	User Id
Hibo Prince	4
Aejaaz Habeeb	5
Jackie	6
admin_vp	7
Abrar Nirban	8
Amayra Fatima	9
Aron	10
Kiran Kumar	11



[Data is From DB and Data Inserted In Cache !!]

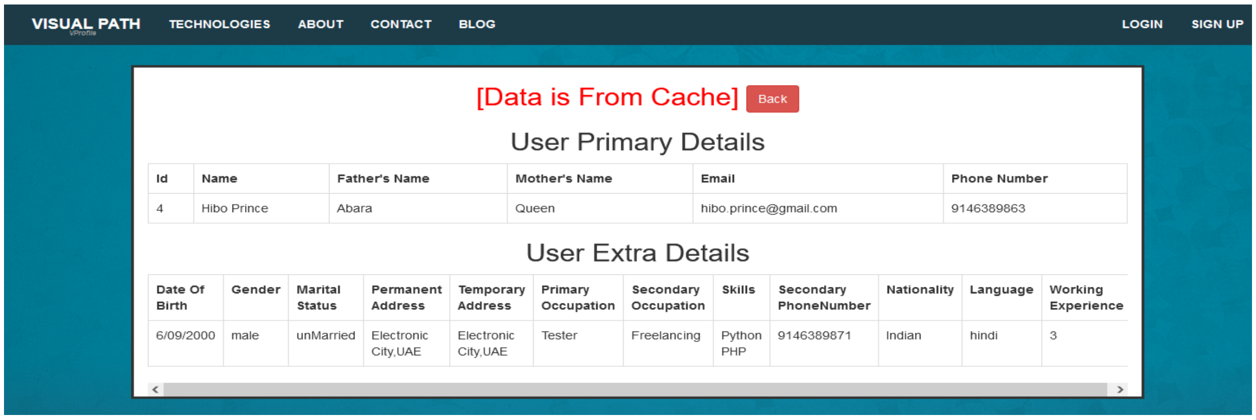
User Primary Details

Id	Name	Father's Name	Mother's Name	Email	Phone Number
4	Hibo Prince	Abara	Queen	hibo.prince@gmail.com	9146389863

User Extra Details

Date Of Birth	Gender	Marital Status	Permanent Address	Temporary Address	Primary Occupation	Secondary Occupation	Skills	Secondary PhoneNumber	Nationality	Language	Working Experience
6/09/2000	male	unMarried	Electronic City,UAE	Electronic City,UAE	Tester	Freelancing	Python PHP	9146389871	Indian	hindi	3

Click on Back, then access this user again.



[Data is From Cache]

User Primary Details

Id	Name	Father's Name	Mother's Name	Email	Phone Number
4	Hibo Prince	Abara	Queen	hibo.prince@gmail.com	9146389863

User Extra Details

Date Of Birth	Gender	Marital Status	Permanent Address	Temporary Address	Primary Occupation	Secondary Occupation	Skills	Secondary PhoneNumber	Nationality	Language	Working Experience
6/09/2000	male	unMarried	Electronic City,UAE	Electronic City,UAE	Tester	Freelancing	Python PHP	9146389871	Indian	hindi	3

➢ **Conclusion:**

# In this project we have covered:

1. What is (**Docker**, **DockerFile**, **Docker image**, **Docker container** and **Docker compose**).
2. How to write Dockerfile to create new image for any App service.
3. How to write Docker Compose file to Containerize and manage all App services.