Aim: Containerization using Docker.

Objectives: The objective of this task is to understand and implement containerization using Docker. This includes installing Docker, containerizing an application, updating and sharing it via Docker Hub, and running multi-container applications.

Tools Used: Virtual box, Ubuntu , Docker

Concepts: Containerization is a way to run applications in separate, lightweight environments with everything they need to work. Docker makes this process easy by packaging an app and its dependencies into a container. With multi-container setups, different parts of an app, like a database and a web server, can work together smoothly.

Problem Statement:
Download the app : https://docs.docker.com/get-started/02_our_app/
Update the app : https://docs.docker.com/get-started/03_updating_app/
Share the app : https://docs.docker.com/get-started/04_sharing_app/
Download the shared app and run it again.
Multi Container App : https://docs.docker.com/get- started/07_multi_container/

Process:

1) **Installation of Docker**
Run:
sudo apt-get install

Now Install Docker Using the command:
sudo apt-get install docker.io

then to check the docker version: docker -v

```
atharva@atharva-VirtualBox:~$ docker -v
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
```

Now start the docker and enable it
sudo systemctl start docker
sudo systemctl enable docker

```
atharva@atharva-VirtualBox:~$ sudo systemctl start docker
atharva@atharva-VirtualBox:~$ sudo systemctl enable docker
```

2) Containerize an application

Make a directory and clone a repo

```
atharva@atharva-VirtualBox:~$ mkdir docker
atharva@atharva-VirtualBox:~$ cd docker/
atharva@atharva-VirtualBox:~/docker$ git clone https://github.com/docker/getting-started-app.git
Cloning into 'getting-started-app'...
remote: Enumerating objects: 79, done.
remote: Counting objects: 100% (28/28), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 79 (delta 18), reused 15 (delta 15), pack-reused 51 (from 1)
Receiving objects: 100% (79/79), 1.67 MiB | 4.22 MiB/s, done.
Resolving deltas: 100% (19/19), done.
atharva@atharva-VirtualBox:~/docker$ ls
getting-started-app
```

Now go inside the getting-started-app folder and create a Dockerfile

```
atharva@atharva-VirtualBox:~/docker$ ls
getting-started-app
atharva@atharva-VirtualBox:~/docker$ cd getting-started-app/
atharva@atharva-VirtualBox:~/docker/getting-started-app$ ls
package.json  README.md  spec  src  yarn.lock
atharva@atharva-VirtualBox:~/docker/getting-started-app$ nano Dockerfile
```

Inside Dockerfile write the below content

```
FROM node:lts-alpine
WORKDIR /app
COPY . .
RUN yarn install --production
CMD ["node", "src/index.js"]
EXPOSE 3000
```

Now Build the Image

Command: sudo docker build -t getting-started .

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~ | atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker build -t getting-started .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  6.451MB
Step 1/6 : FROM node:lts-alpine
lts-alpine: Pulling from library/node
f18232174bc9: Pull complete
cb2bde55f71f: Pull complete
9d0e0719fbe0: Pull complete
6f063dbd7a5d: Pull complete
Digest: sha256:9bef0ef1e268f60627da9ba7d7605e8831d5b56ad07487d24d1aa386336d1944
Status: Downloaded newer image for node:lts-alpine
 ---> 33544e83793c
Step 2/6 : WORKDIR /app
 ---> Running in b79741f7fbb1
 ---> Removed intermediate container b79741f7fbb1
 ---> 1d546f217d1f
Step 3/6 : COPY . .
 ---> c0cf53d4dba7
Step 4/6 : RUN yarn install --production
 ---> Running in 38da9ea796b7
yarn install v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
Done in 12.12s.
 ---> Removed intermediate container 38da9ea796b7
 ---> 82b4e58e50b6
Step 5/6 : CMD ["node", "src/index.js"]
 ---> Running in a38ae942bbad
 ---> Removed intermediate container a38ae942bbad
 ---> b080d69cc053
Step 6/6 : EXPOSE 3000
 ---> Running in 55e936bf302f
 ---> Removed intermediate container 55e936bf302f
 ---> bd193d3cb5ff
Successfully built bd193d3cb5ff
Successfully tagged getting-started:latest
```

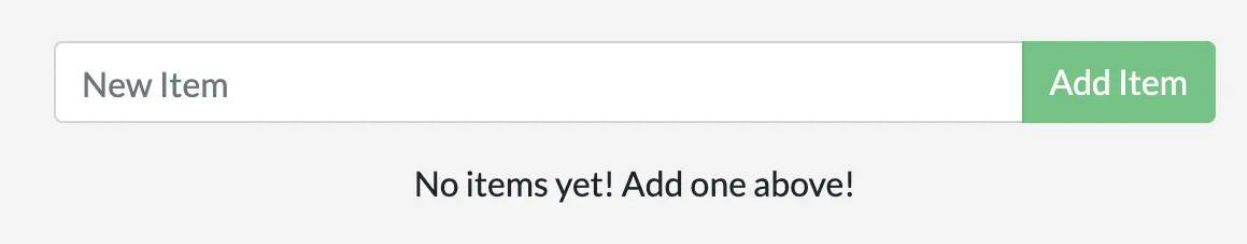
Now Start an app container

command : sudo docker run -d -p 127.0.0.1:3000:3000 getting-started

Now to check the status run: sudo docker ps -a

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~ | atharva@atharva-VirtualBox: ~/docker/getting-started-app | atharva@atharva-VirtualBox: ~/docker/getting-started-app
 ---> d452a738d505
Step 4/6 : RUN yarn install --production
 ---> Using cache
 ---> 85e4a1f655ff
Step 5/6 : CMD ["node", "src/index.js"]
 ---> Using cache
 ---> a2880ce642e4
Step 6/6 : EXPOSE 3000
 ---> Using cache
 ---> c79932de41db
Successfully built c79932de41db
Successfully tagged getting-started:latest
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker run -d -p 127.0.0.1:3000:3000 getting-started
adace5499dc37ad5c3059098f8a1f7a673bd6a9a355df9874cf8042bca9fb3eb
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED         STATUS                     PORTS                      NAMES
adace5499dc3   getting-started   "docker-entrypoint.s…"   8 seconds ago   Up 7 seconds               127.0.0.1:3000->3000/tcp   priceless_volhard
bc9aa0870352   ubuntu            "/bin/bash"              36 hours ago    Exited (0) 36 hours ago                               zen_golick
atharva@atharva-VirtualBox:~/docker/getting-started-app$
```

After a few seconds, open your web browser to http://localhost:3000. You should see your app.



3) **Update the application**

Update the source code

In the `src/static/js/app.js` file of the project make some changes.

Now, Build your updated version of the image, using the `docker build` command

->docker build -t getting-started .

Now remove the old container first

Get the ID of the container by using the `docker ps -a` command.

Use the `docker stop` command to stop the container. Replace `<the-container-id>` with the ID from `docker ps`.

```
docker stop <the-container-id>
```

Once the container has stopped, you can remove it by using the docker rm command.

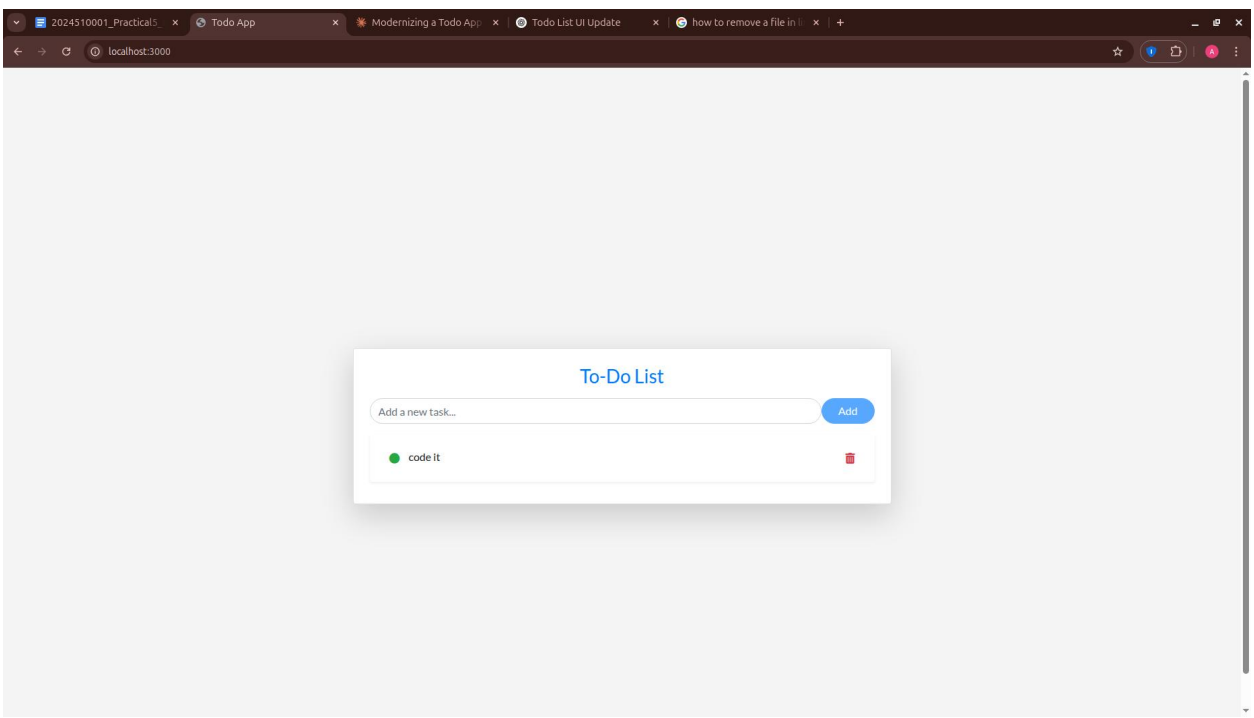
```
docker rm <the-container-id>
```

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~ | atharva@atharva-VirtualBox: ~/docker/getting-started-app | atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker run -d -p 127.0.0.1:3000:3000 getting-started
adace5499dc37ad5c3059098f8a1f7a673bd6a9a355df9874cf8042bca9fb3eb
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker ps -a
CONTAINER ID   IMAGE            COMMAND                  CREATED         STATUS                   PORTS                      NAMES
adace5499dc3   getting-started  "docker-entrypoint.s…"   8 seconds ago   Up 7 seconds             127.0.0.1:3000->3000/tcp   priceless_volhard
bc9aa0870352   ubuntu           "/bin/bash"              36 hours ago    Exited (0) 36 hours ago                             zen_golick
atharva@atharva-VirtualBox:~/docker/getting-started-app$ docker stop ^C

atharva@atharva-VirtualBox:~/docker/getting-started-app$ docker stop adace5499dc3
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.45/containers
/adace5499dc3/stop": dial unix /var/run/docker.sock: connect: permission denied
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker stop adace5499dc3
adace5499dc3
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker rm adace5499dc3
adace5499dc3
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED        STATUS                     PORTS     NAMES
bc9aa0870352   ubuntu    "/bin/bash"   36 hours ago   Exited (0) 36 hours ago              zen_golick
atharva@atharva-VirtualBox:~/docker/getting-started-app$
```

Now, start your updated app using the `docker run` command.

```
docker run -dp 127.0.0.1:3000:3000 getting-started
```

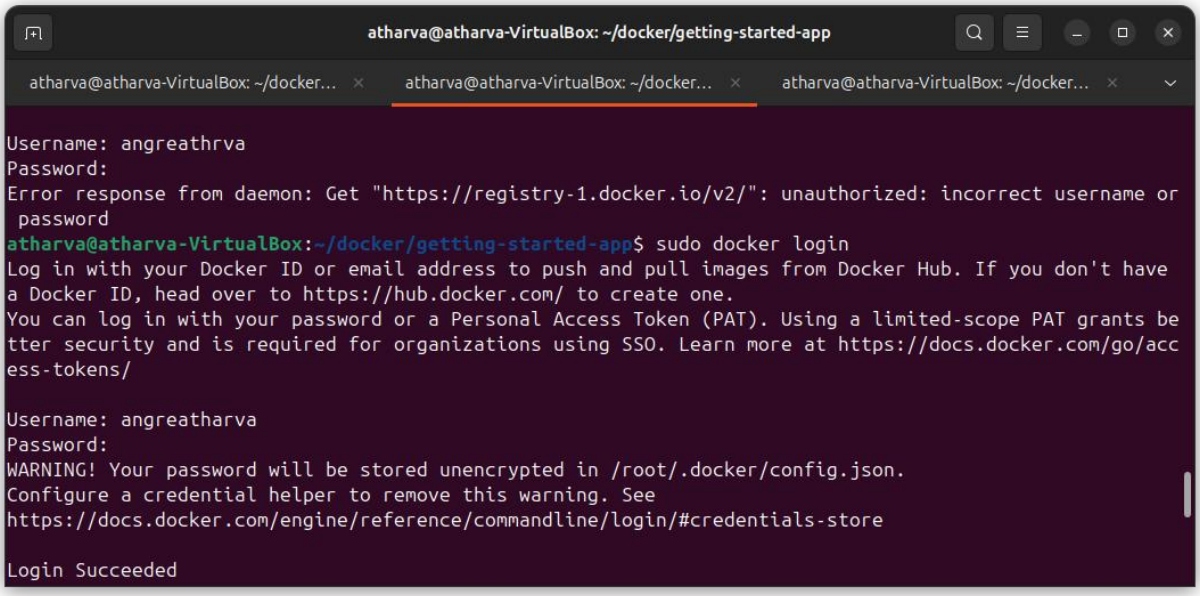


4) Share the application

Create a repository

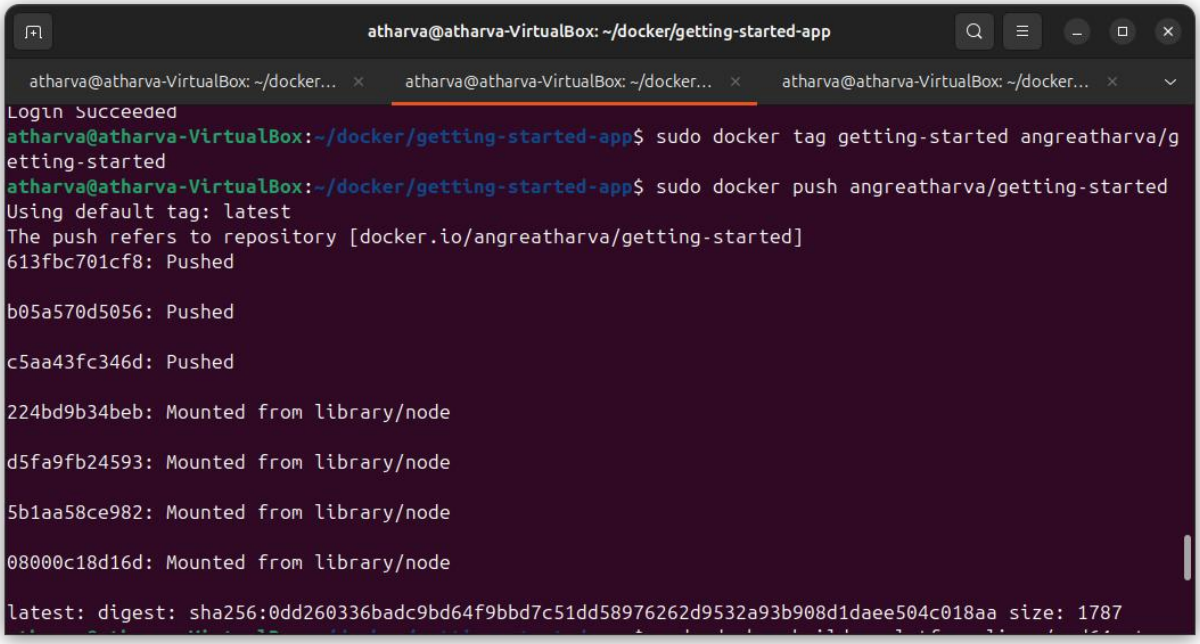
a) Sign up or Sign in to Docker Hub.
b) Select the Create Repository button.
c) For the repository name, use getting-started. Make sure the Visibility is Public.
d) Select Create.

Push the image
First Login to docker in CLI
command : sudo docker login



Now first
Use the `docker tag` command to give the `getting-started` image a new name. Replace `YOUR-USER-NAME` with your Docker ID.
Command: docker tag getting-started YOUR-USER-NAME/getting-started

And then
In the command line, run the `docker push` command that you see on Docker Hub. Note that your command will have your Docker ID.
For example, `docker push YOUR-USER-NAME/getting-started`.

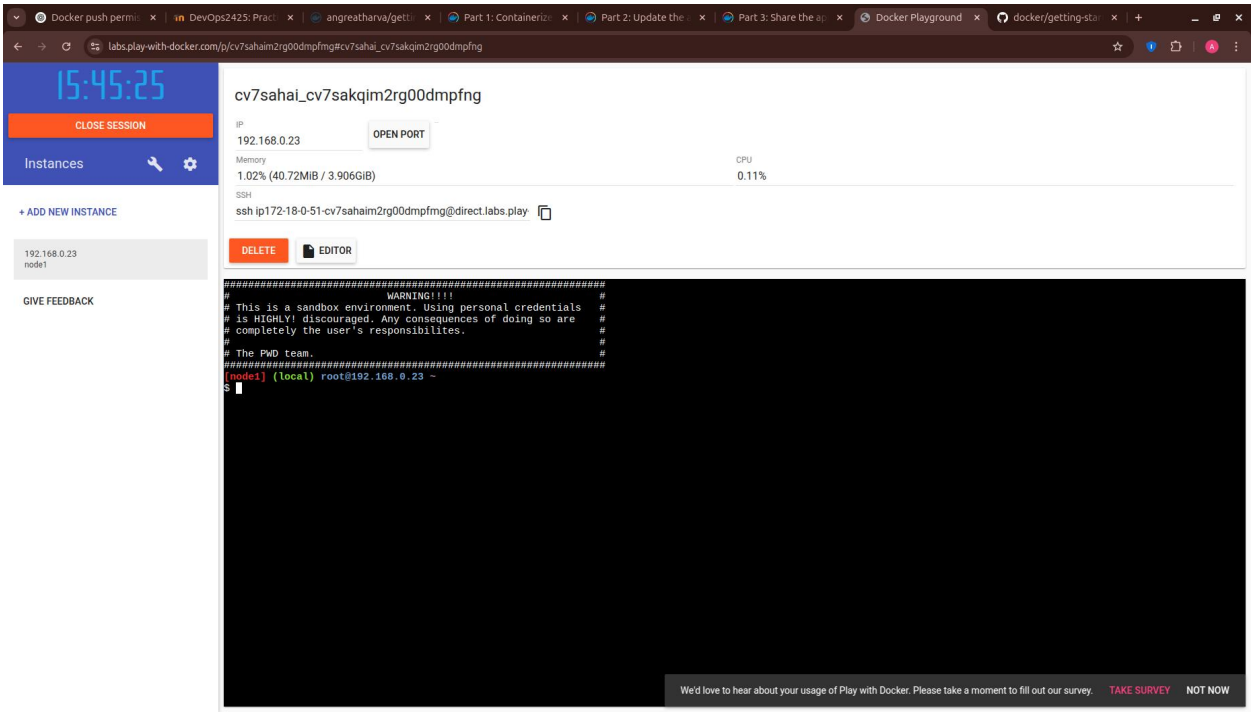


Now Run the image on a new instance

To build an image for the amd64 platform, use the `--platform` flag.
Command: docker build --platform linux/amd64 -t YOUR-USER-NAME/getting-started .

Now

a) Open your browser to Play with Docker.
b) Select Login and then select docker from the drop-down list.
c) Sign in with your Docker Hub account and then select Start.
d) Select the ADD NEW INSTANCE option on the left side bar. If you don't see it, make your browser a little wider. After a few seconds, a terminal window opens in your browser.



e) In the terminal, start your freshly pushed app.
-> In the terminal, start your freshly pushed app.

5) Multi container apps

Create the network.

command : sudo docker network create todo-app

Start a MySQL container and attach it to the network.
Command:

```
docker run -d \
  --network todo-app --network-alias mysql \
  -v todo-mysql-data:/var/lib/mysql \
  -e MYSQL_ROOT_PASSWORD=secret \
  -e MYSQL_DATABASE=todos \
  mysql:8.0
```

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started...
atharva@atharva-VirtualBox:~$ sudo docker run -d \
    --network todo-app --network-alias mysql \
    -v todo-mysql-data:/var/lib/mysql \
    -e MYSQL_ROOT_PASSWORD=secret \
    -e MYSQL_DATABASE=todos \
    mysql:8.0
Unable to find image 'mysql:8.0' locally
8.0: Pulling from library/mysql
804bb8ae89de: Pull complete
34a45e327c17: Pull complete
5b7dcae372f2: Pull complete
77895c94e6d5: Pull complete
ba9f25a041ba: Pull complete
841b743c6edc: Pull complete
f11e07506c9a: Pull complete
8615fa21e026: Pull complete
0b80abbc4aed: Pull complete
6594c9f77fef: Pull complete
cbcad5d15a99: Pull complete
Digest: sha256:59cc3d706de79eb34945e2c49c8727fd399a1e97dfc222269131b846ca7047da
Status: Downloaded newer image for mysql:8.0
008c361218b57bee8df4134ea29bd77bf6a006b5b0e8508ba2eb1903d81f5647
atharva@atharva-VirtualBox:~$ sudo docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED         STATUS                  PORTS                       NAMES
008c361218b5   mysql:8.0         "docker-entrypoint.s…"   8 seconds ago   Up 6 seconds            3306/tcp, 33060/tcp         zealous_go
lick
```

To confirm you have the database up and running, connect to the database and verify that it connects.

Command : docker exec -it <mysql-container-id> mysql -u root -p

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started...
atharva@atharva-VirtualBox:~$ sudo docker exec -it 008c361218b5 mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.41 MySQL Community Server - GPL

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| todos              |
+--------------------+
```

Start a new container using the nicolaka/netshoot image. Make sure to connect it to the same network.

Command: docker run -it --network todo-app nicolaka/netshoot

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started...
atharva@atharva-VirtualBox:~$ sudo docker run -it --network todo-app nicolaka/netshoot
Unable to find image 'nicolaka/netshoot:latest' locally
latest: Pulling from nicolaka/netshoot
4abcf2066143: Pull complete
f72249ed6705: Pull complete
d21093198226: Pull complete
ff793c57efef: Pull complete
b8cdfec6d24e: Pull complete
b6621d484422: Pull complete
452eb7889eb5: Pull complete
4f4fb700ef54: Pull complete
89065cf5c037: Pull complete
a4b421d4901a: Pull complete
d5c3ad7ea15a: Pull complete
ab073295bbd0: Pull complete
737c1bf9f2ef: Pull complete
097ac21093f8: Pull complete
59e353e0ee74: Pull complete
Digest: sha256:a20c2531bf35436ed3766cd6cfe89d352b050ccc4d7005ce6400adf97503da1b
Status: Downloaded newer image for nicolaka/netshoot:latest
                    dP            dP                           dP
                    88            88                           88
88d888b. .d8888b. d8888P .d8888b. 88d888b. .d8888b. .d8888b. d8888P
88'  `88 88ooood8   88   Y8ooooo. 88'  `88 88'  `88 88'  `88   88
88    88 88.  ...   88         88 88    88 88.  .88 88.  .88   88
dP    dP `88888P'   dP   `88888P' dP    dP `88888P' `88888P'   dP

Welcome to Netshoot! (github.com/nicolaka/netshoot)
Version: 0.13
```

Inside the container, you're going to use the `dig` command, which is a useful DNS tool. You're going to look up the IP address for the hostname `mysql`.

Command : dig mysql

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started... atharva@atharva-VirtualBox: ~/docker/getting-started...

ba201d83a380 ~ dig mysql

; <<>> DiG 9.18.25 <<>> mysql
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37308
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;mysql.                         IN      A

;; ANSWER SECTION:
mysql.                  600     IN      A       172.18.0.2

;; Query time: 22 msec
;; SERVER: 127.0.0.11#53(127.0.0.11) (UDP)
;; WHEN: Mon Mar 17 14:13:39 UTC 2025
;; MSG SIZE  rcvd: 44


ba201d83a380 ~ ls
motd
```

You can now start your dev-ready container.

1. Specify each of the previous environment variables, as well as connect the container to your app network. Make sure that you are in the `getting-started-app` directory when you run this command.

   Command : docker run -dp 127.0.0.1:3000:3000 \

   -w /app -v "$(pwd):/app" \

   --network todo-app \

   -e MYSQL_HOST=mysql \

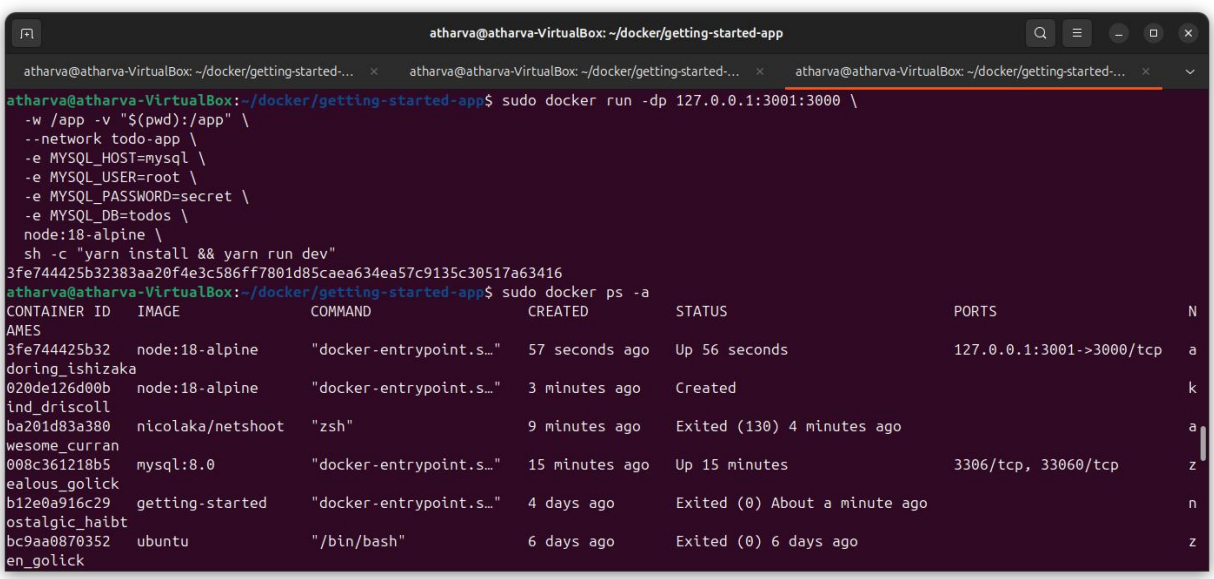
   -e MYSQL_USER=root \

   -e MYSQL_PASSWORD=secret \

   -e MYSQL_DB=todos \

   node:18-alpine \

   sh -c "yarn install && yarn run dev"



2. If you look at the logs for the container (`docker logs -f <container-id>`), you should see a message similar to the following, which indicates it's using the mysql database.

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
b12e0a916c29   getting-started     "docker-entrypoint.s…"   4 days ago       Exited (0) About a minute ago                    n
ostalgic_haibt
bc9aa0870352   ubuntu              "/bin/bash"              6 days ago       Exited (0) 6 days ago                            z
en_golick
atharva@atharva-VirtualBox:~/docker/getting-started-app$ docker logs -f 3fe744425b32
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.s
ock/v1.45/containers/3fe744425b32/json": dial unix /var/run/docker.sock: connect: permission denied
atharva@atharva-VirtualBox:~/docker/getting-started-app$ sudo docker logs -f 3fe744425b32
yarn install v1.22.22
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
[4/4] Building fresh packages...
Done in 18.80s.
yarn run v1.22.22
$ nodemon -L src/index.js
[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node src/index.js`
Waiting for mysql:3306.
Connected!
Connected to mysql db at host mysql
Listening on port 3000
```

3. Open the app in your browser and add a few items to your todo list.

4. Connect to the mysql database and prove that the items are being written to the database. Remember, the password is `secret`.

```
$ docker exec -it <mysql-container-id> mysql -p todos
```

```
atharva@atharva-VirtualBox: ~/docker/getting-started-app
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| todos              |
+--------------------+
5 rows in set (0.00 sec)

mysql> select * from todos;
ERROR 1146 (42S02): Table 'todos.todos' doesn't exist
mysql> use todos
Database changed
mysql> select * from todo_items;
+--------------------------------------+-------------+-----------+
| id                                   | name        | completed |
+--------------------------------------+-------------+-----------+
| 2b9db6b4-88d1-4f47-a21b-8e91ef02cb7c | code it out |         0 |
+--------------------------------------+-------------+-----------+
1 row in set (0.01 sec)

mysql>
```

Observation: Docker made it simple to package, run, and update the app. Sharing it through Docker Hub was easy, and using multiple containers helped organize the app better. This method makes applications more portable, scalable, and easy to manage.