Lab2 Docker

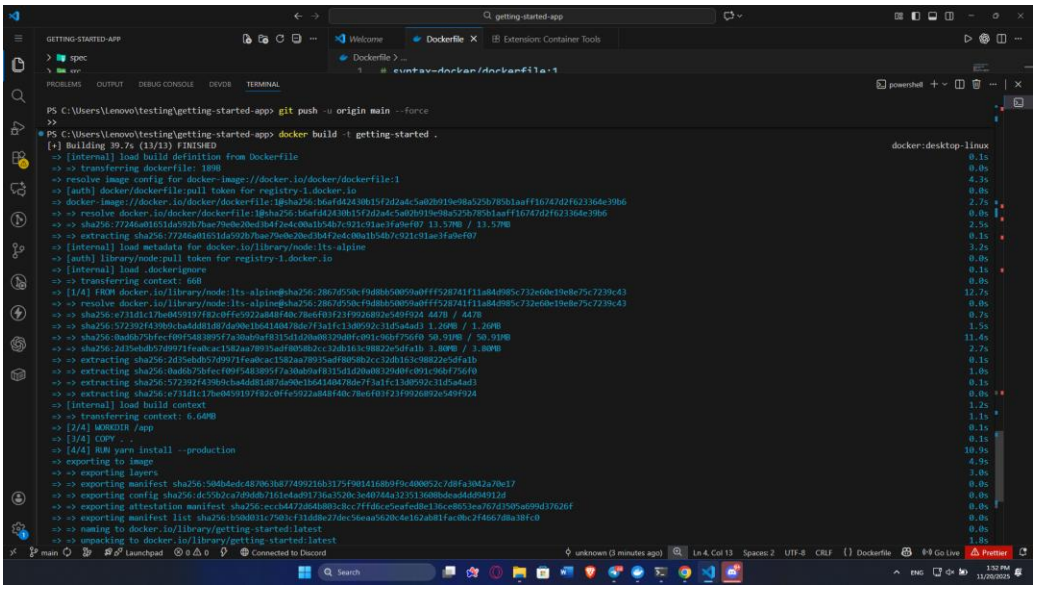
Name: Kheng Chamnan

1.Containzer Applicatons

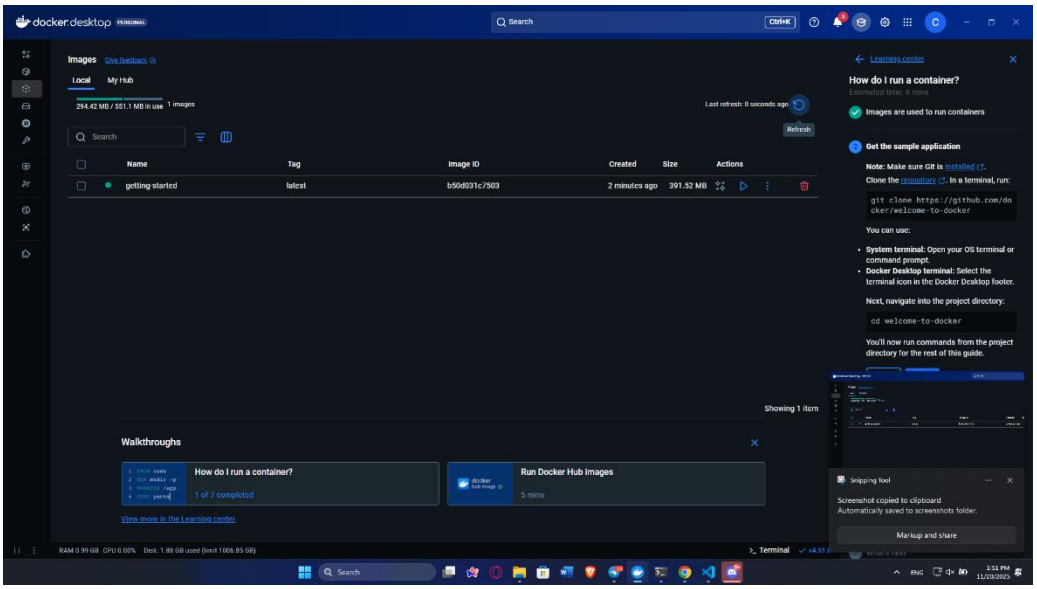
In this section I learn how to create a containzer with node:lts-alpine base image how to set working directory inside the containzer within the exisitng repository ,copy my local file into WORKDIR

Using yarn command to install image specifiy index.js as the start application then host it on my localhost

Docker Build Screenshoot



Docker Image In Docker Hub



Listing all Image

```
f8c30826f9f4cadc7b6451282a7ffbda7bda9eb23b04e9e86fa9997b6e71bfca
PS C:\Users\Lenovo\testing\getting-started-app> docker images -a
REPOSITORY        TAG       IMAGE ID       CREATED          SIZE
getting-started   latest    b50d031c7503   15 minutes ago   392MB
PS C:\Users\Lenovo\testing\getting-started-app>
```

2.Update Application

In This section I learn how to update applications in my source code and rebuild my update images

And learn that in order to build new cotainer need to stop the old one by removing it since it host on the same port .

Change Source Code

```
49
50      if (items === null) return 'Loading...';
51
52      return (
53          <React.Fragment>
54              <AddItemForm onNewItem={onNewItem} />
55              {items.length === 0 && (
56                  <p className="text-center">You have no todo items yet! Add one above!</p>
57              )}
58              {items.map(item => (
59                  <ItemDisplay
60                      item={item}
61                      key={item.id}
62                      onItemUpdate={onItemUpdate}
63                      onItemRemoval={onItemRemoval}
```

Error From start new docker

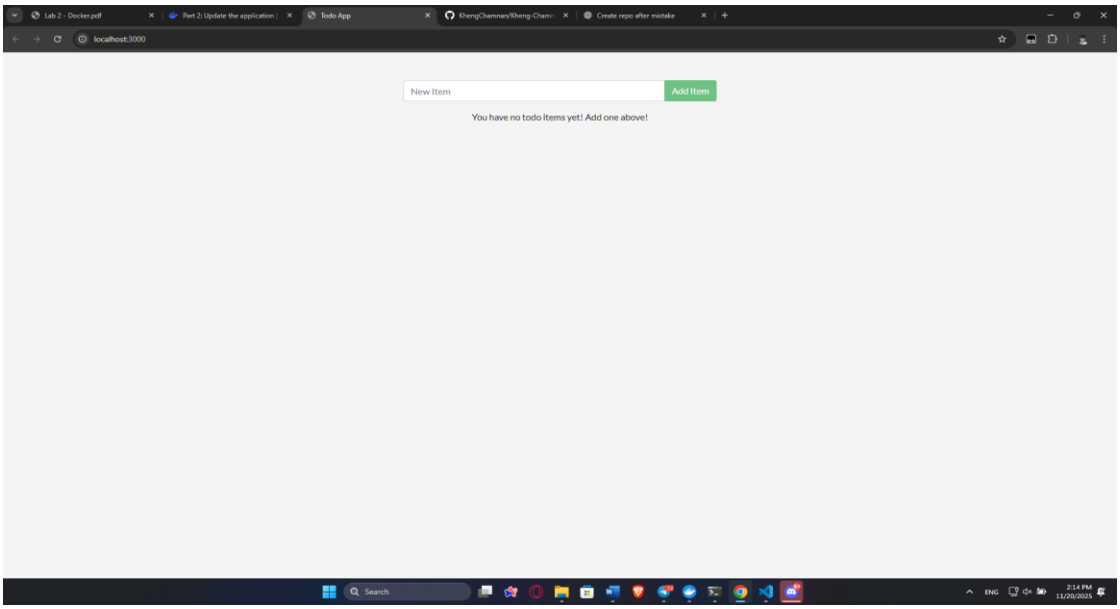
```
PS C:\Users\Lenovo\testing\getting-started-app> docker run -dp 127.0.0.1:3000:3000 getting-started
826e27f3a47dd706ec35260c3a7ba6433693dea9c2ceb552ecf88aba8597656e
docker: Error response from daemon: failed to set up container networking: driver failed programming external connectivity on endpoint reverent_kare (ed2d2d71340ea327eb28ea9b9108104ac08d9505e4616f
de18b3af4e4babf00c): Bind for 127.0.0.1:3000 failed: port is already allocated

Run 'docker run --help' for more information
PS C:\Users\Lenovo\testing\getting-started-app>
```

Remove old container and rebuild new container

```
Run 'docker --help' for more information
PS C:\Users\Lenovo\testing\getting-started-app> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                     NAMES
f8c30826f9f4   b50d031c7503   "docker-entrypoint.s…"   22 minutes ago   Up 22 minutes   127.0.0.1:3000->3000/tcp   compassionate_ramanujan
PS C:\Users\Lenovo\testing\getting-started-app> docker stop f8c30826f9f4
f8c30826f9f4
PS C:\Users\Lenovo\testing\getting-started-app> docker rm f8c30826f9f4
f8c30826f9f4
PS C:\Users\Lenovo\testing\getting-started-app>  docker run -dp 127.0.0.1:3000:3000 getting-started
>>
76718df3ff6421d5bf9e8d7c2237d09ad9c300817207c967e256a00d0243ecdb
PS C:\Users\Lenovo\testing\getting-started-app>
```

New Updated Application



3.Share The application

In this section I learning how to push my local docker to docker repo and can be share anyone for example we can share it on dockerplay

Docker push erro with wrong DockerID

```
PS C:\Users\Lenovo\testing\getting-started-app> docker push docker/getting-started
>>
Using default tag: latest
The push refers to repository [docker.io/docker/getting-started]
tag does not exist: docker/getting-started:latest
```

Login and set a right docker tag base on my created Repo

```
up chamnann1: no such host
PS C:\Users\Lenovo\testing\getting-started-app> docker login
Authenticating with existing credentials... [Username: chamnann1]

i Info → To login with a different account, run 'docker logout' followed by 'docker login'


Login Succeeded
PS C:\Users\Lenovo\testing\getting-started-app> docker tag getting-started chamnann1/getting-started
PS C:\Users\Lenovo\testing\getting-started-app>
```

Push image to repo

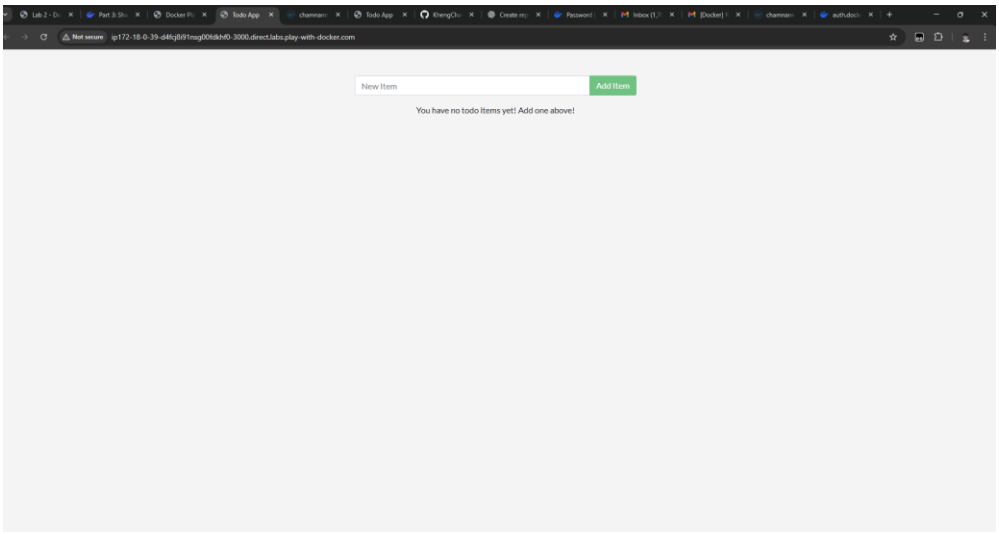
```
i Info → To login with a different account, run 'docker logout' followed by 'docker login'


Login Succeeded
PS C:\Users\Lenovo\testing\getting-started-app>
PS C:\Users\Lenovo\testing\getting-started-app> docker push chamnann1/getting-started:latest
The push refers to repository [docker.io/chamnann1/getting-started]
2d35ebdb57d9: Pushed
e731d1c17be0: Pushed
b5c085f6a80a: Pushed
0ad6b75bfecf: Pushed
28b3fff5fc34: Pushed
1cf8ffe47b87: Pushed
c381d9bdb5ee: Pushed
572392f439b9: Pushed
latest: digest: sha256:3b91ba959384a952851f1444dbbf088281874a7c2c382b7870e85ffcb5a8b54e size: 856
PS C:\Users\Lenovo\testing\getting-started-app>
```

Push image to dockerplay

```
################################################################
#                          WARNING!!!!                         #
# This is a sandbox environment. Using personal credentials    #
# is HIGHLY! discouraged. Any consequences of doing so are     #
# completely the user's responsibilites.                       #
#                                                              #
# The PWD team.                                                #
################################################################
[node1] (local) root@192.168.0.29 ~
$ docker run -dp 0.0.0.0:3000 chamnann1/getting-started
docker: invalid hostPort: 0.0.0.0.
See 'docker run --help'.
[node1] (local) root@192.168.0.29 ~
$ docker run -dp 0.0.0.0:3000:000  chamnann1/getting-started
Unable to find image 'chamnann1/getting-started:latest' locally
latest: Pulling from chamnann1/getting-started
2d35ebdb57d9: Pull complete
0ad6b75bfecf: Pull complete
572392f439b9: Pull complete
e731d1c17be0: Pull complete
b5c085f6a80a: Pull complete
28b3fff5fc34: Pull complete
c381d9bdb5ee: Pull complete
Digest: sha256:3b91ba959384a952851f1444dbbf088281874a7c2c382b7870e85ffcb5a8b54e
Status: Downloaded newer image for chamnann1/getting-started:latest
93fa1cd5f45371996d702d463a205d42a4a3497449ff3cf13b4645546f4f68b5
[node1] (local) root@192.168.0.29 ~
$
```

Application Run on dockerplay



4.Persist the DB

In this section I learn that data are not persist if we run new container using same image unless we mount it to out volume

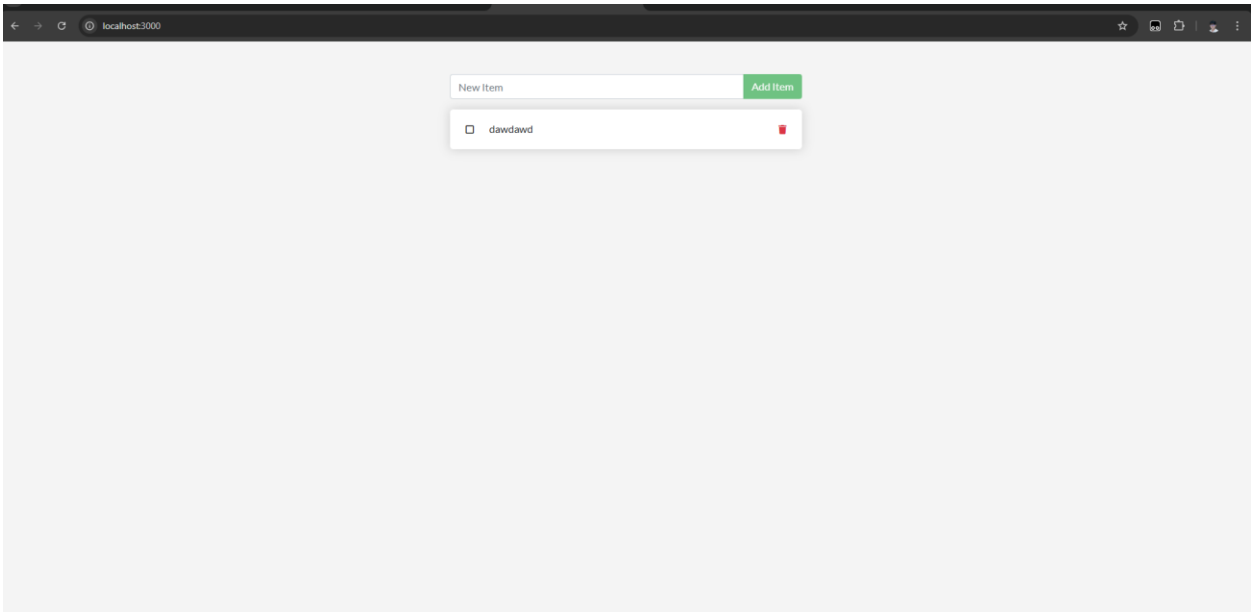
Data is not persist

```
>> C:\Users\Lenovo\testing\getting-started-app>
stat: can't stat 'greeting.txt': No such file or directory
PS C:\Users\Lenovo\testing\getting-started-app>  docker run --rm alpine stat greeting.txt
>>
stat: can't stat 'greeting.txt': No such file or directory
PS C:\Users\Lenovo\testing\getting-started-app>  docker run --rm alpine touch greeting.txt
```

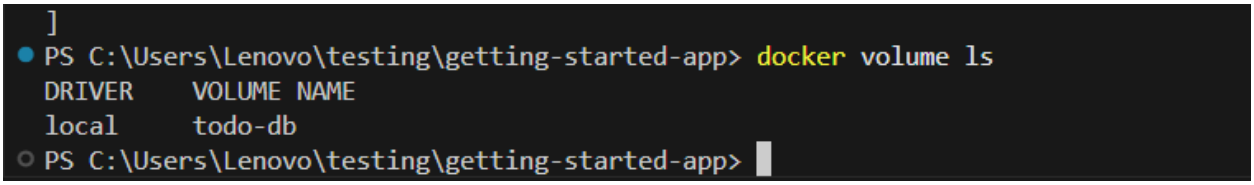
Create volume and mount to todos on my host machine

```
todo-db
PS C:\Users\Lenovo\testing\getting-started-app> docker rm -f 66efcd40e5cd
66efcd40e5cd
PS C:\Users\Lenovo\testing\getting-started-app> docker run -dp 127.0.0.1:3000:3000 --mount type=volume,src=todo-db,target=/etc/todos getting-started
02222b5752f0e7590f683754a8ff8460530edd95f3d2c688947d4322b33d4101
PS C:\Users\Lenovo\testing\getting-started-app>
```

Data still exist when we run our container mount to todo volume



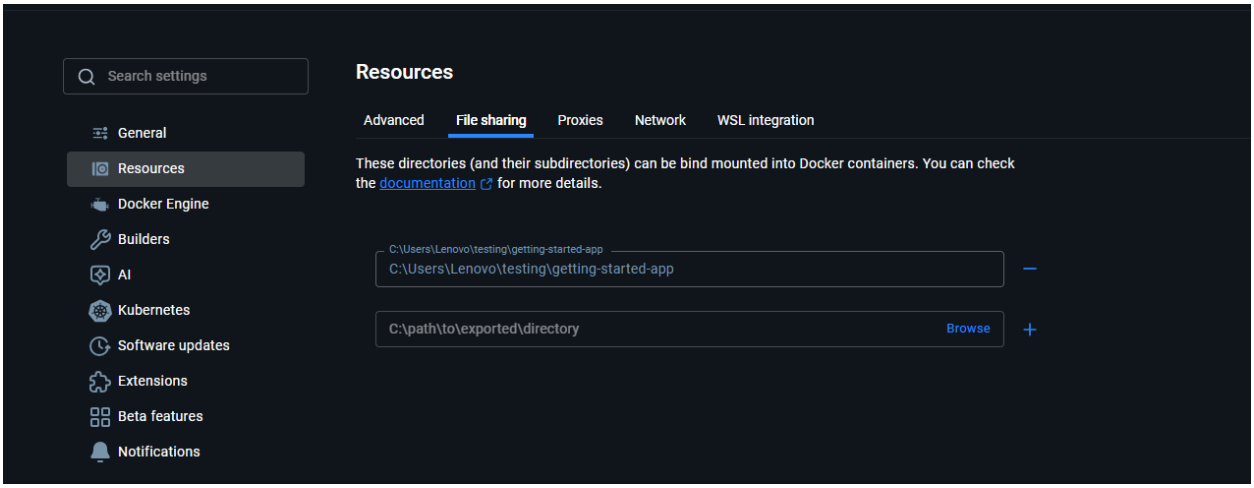
Proof of mount to local machine



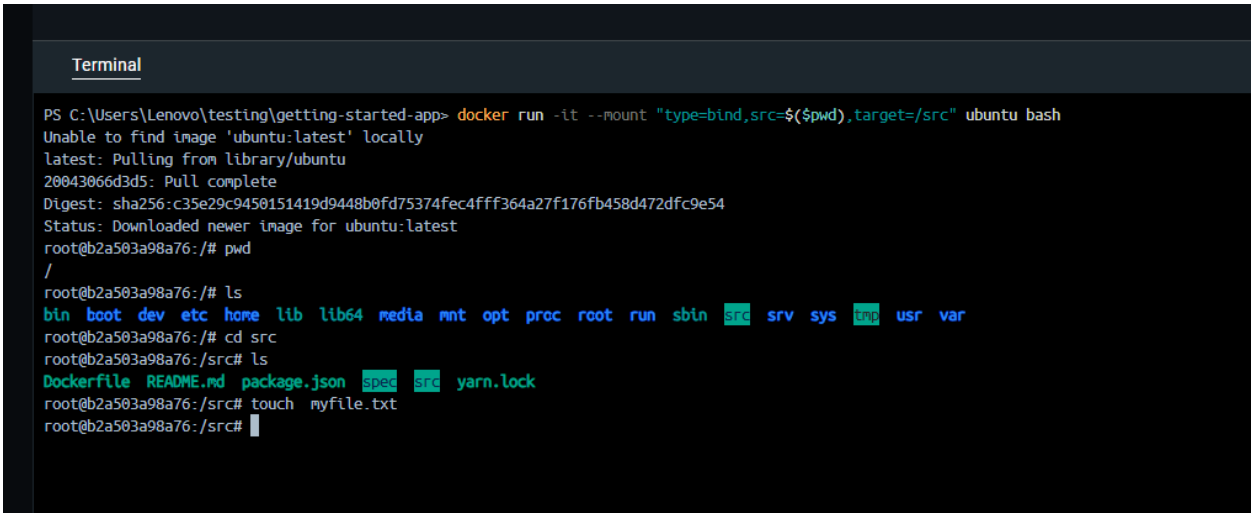
5.Use Bind mount

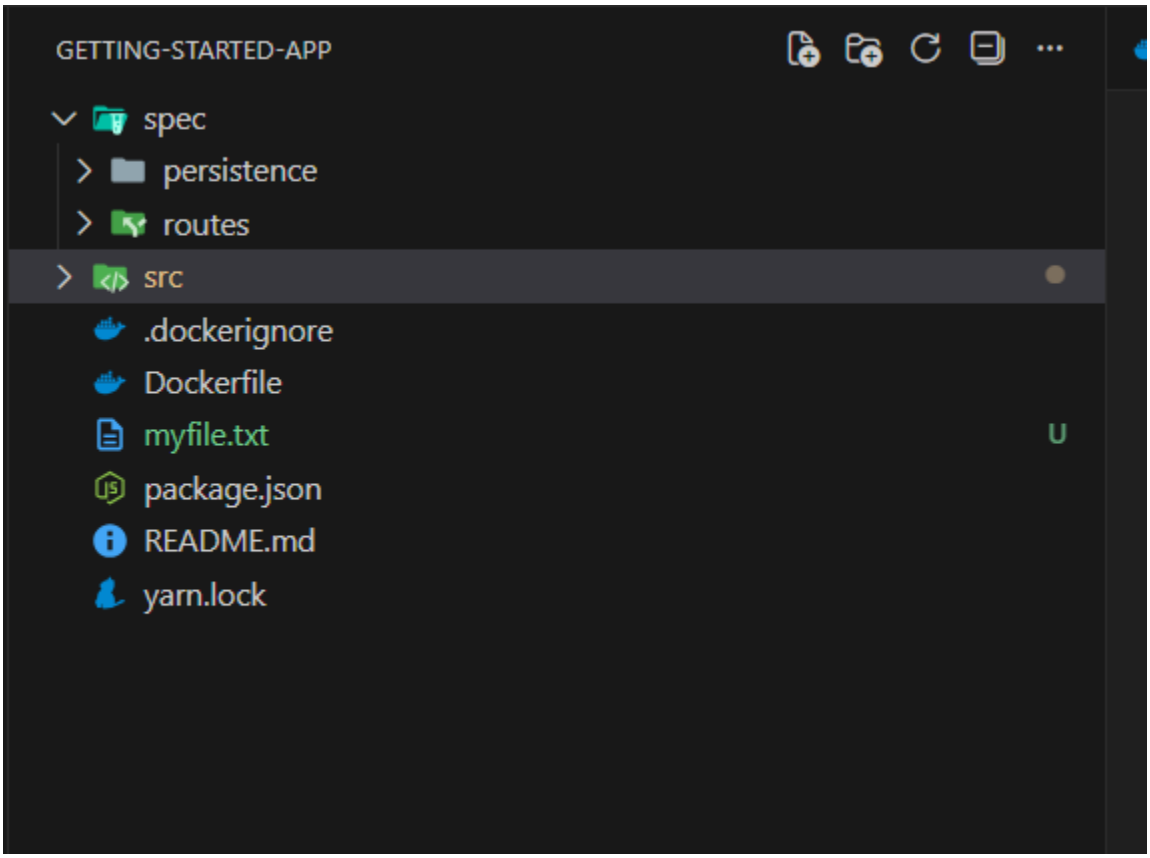
In this section

Define directory

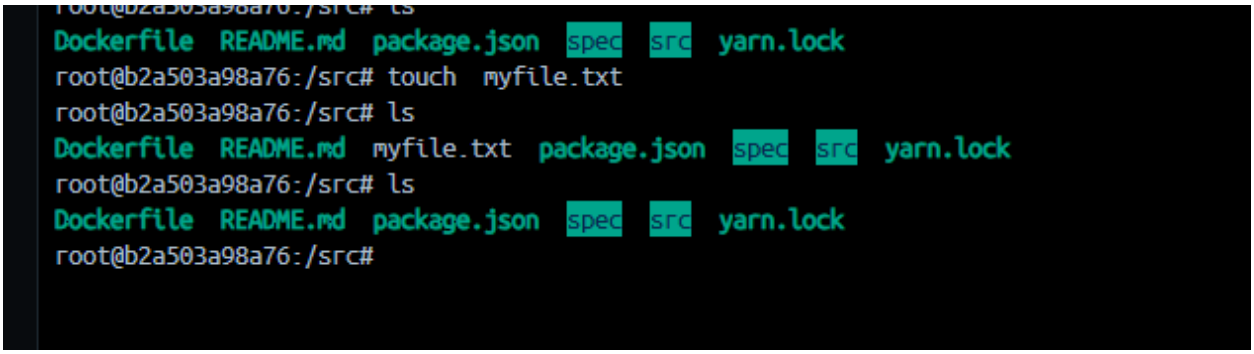


Mount to my local machine and verify





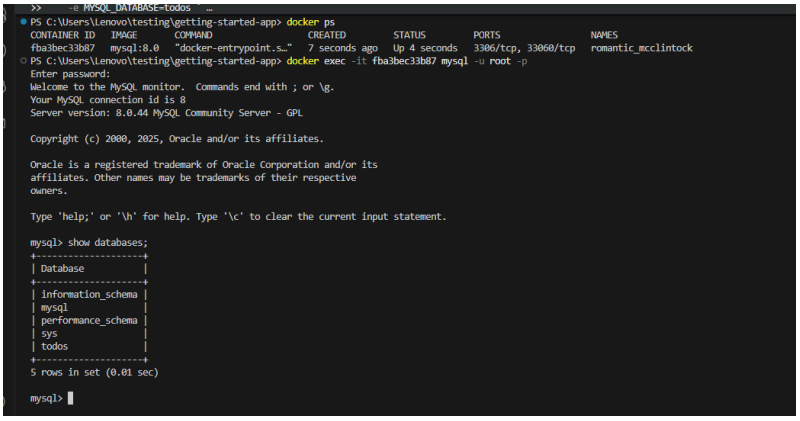
Verify delete in host and check in the container session



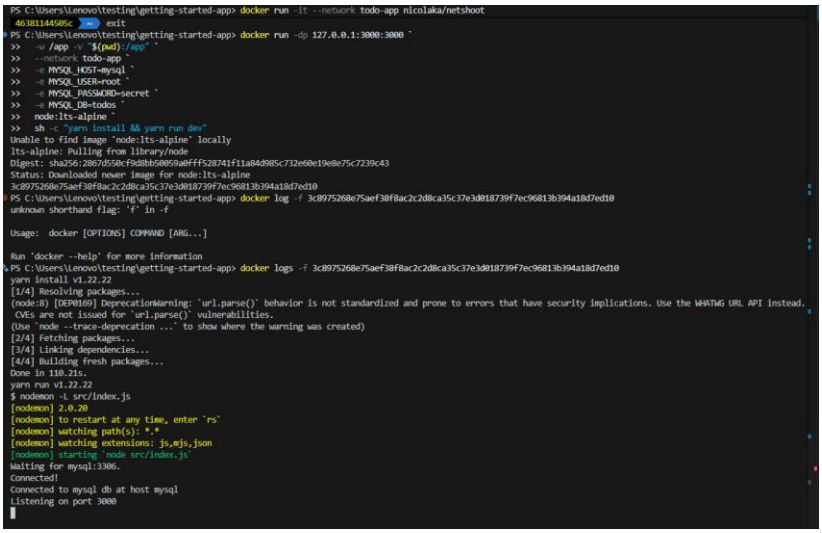
6.Mutiple container

In this section I learn how to run other container such as mysql and know how to let each container talk to each other by staying in the same network by mount the volume to the mysql our data now are store in mysql

Mysql container

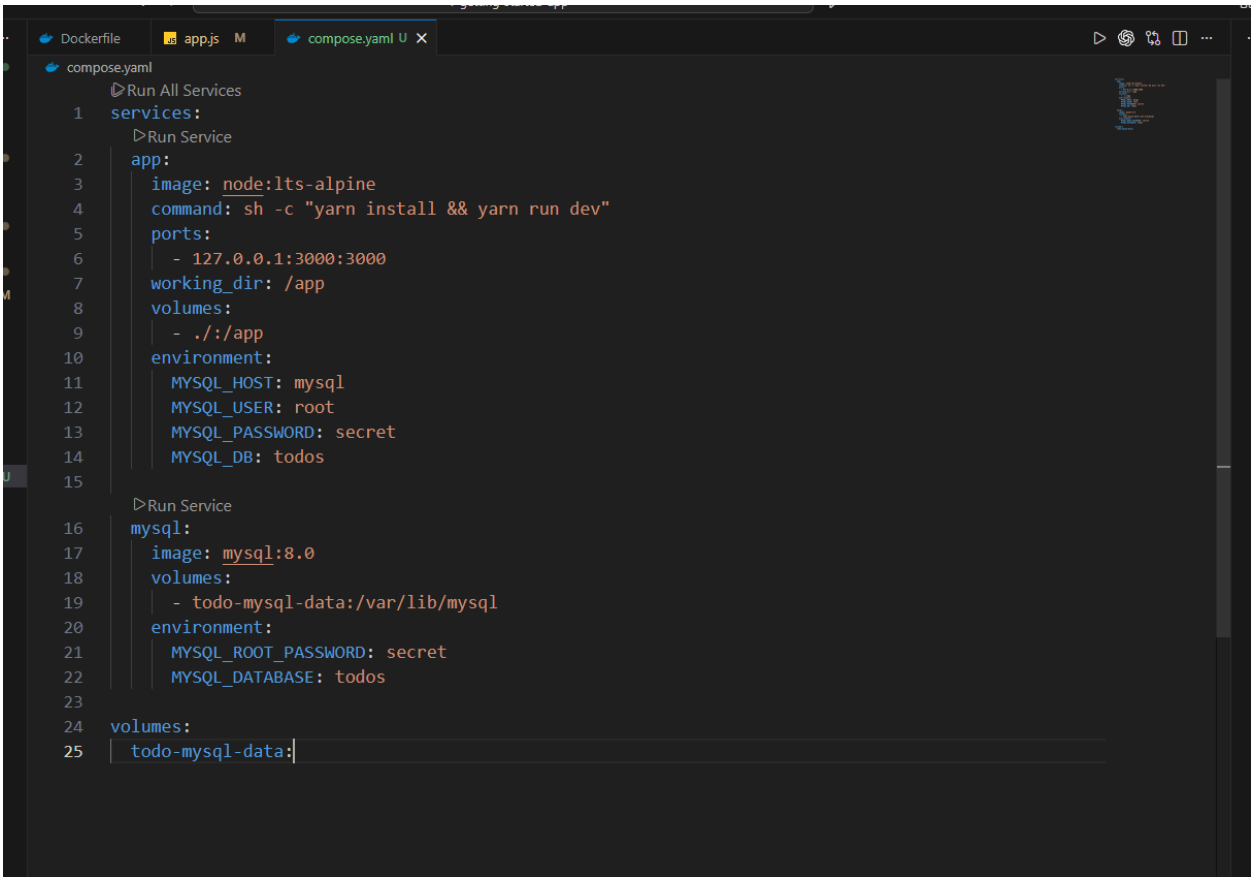


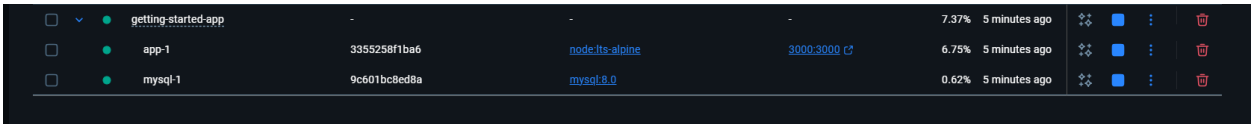
Connect app to mysql



7.Docker Compose

in this section I learn how to define all container using compose yaml which is easy instead of typing and run each container and benefit from compose yaml is that when we run stack of application it create a network for us automatically too.

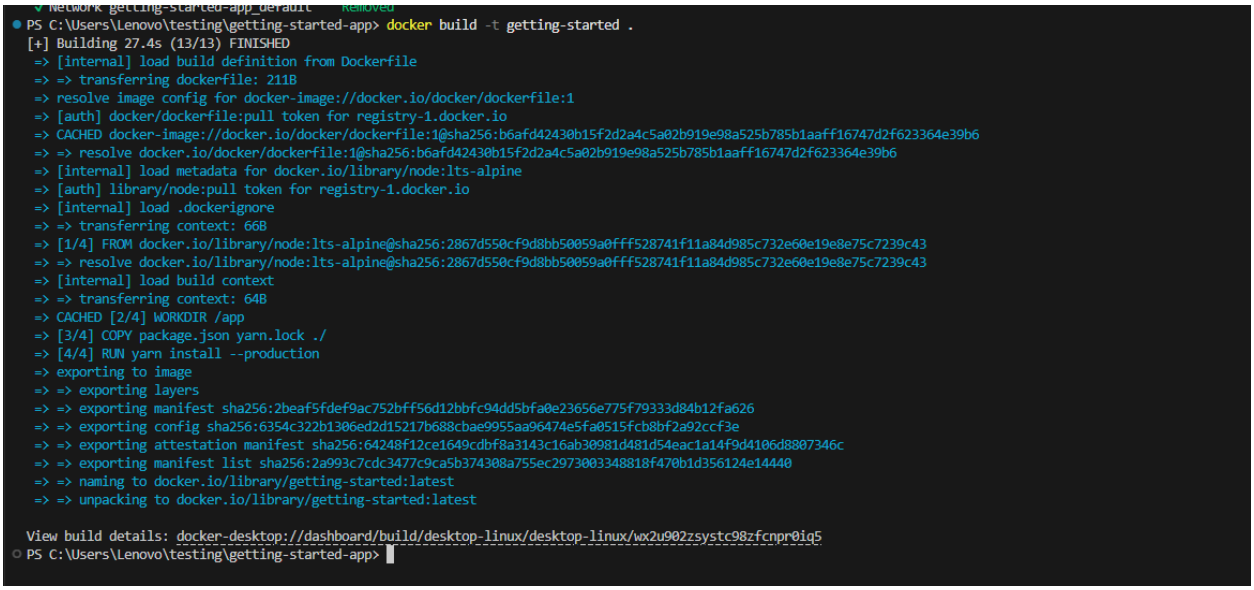




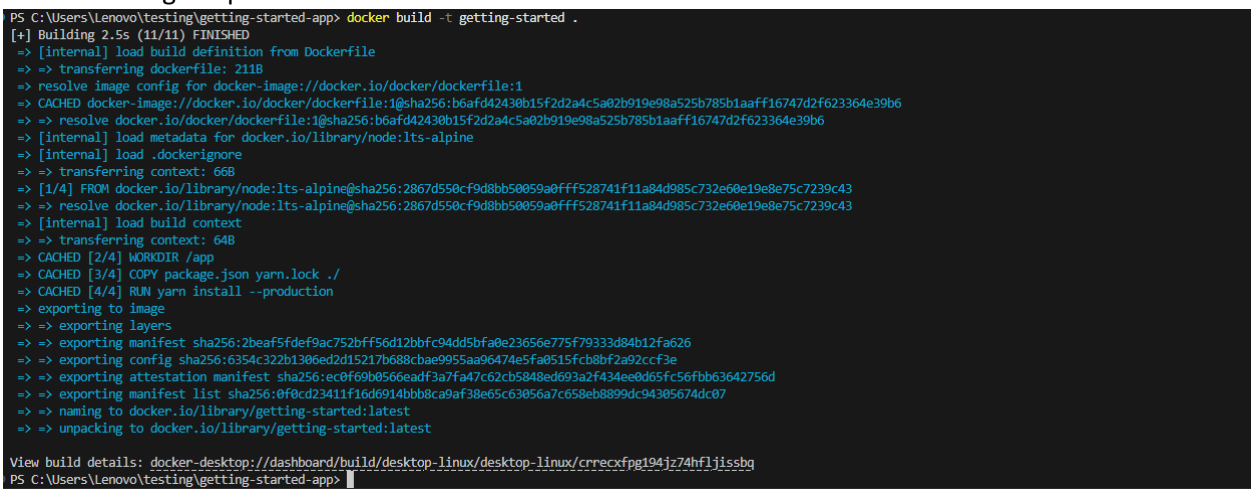
8. Image Building best practice

In this section, I learned about caching layers in Docker image building, which decreases the build time because unchanged layers (like dependencies) don't need to be rebuilt every time I make changes to my code. By copying package.json and yarn.lock first and installing dependencies before copying the rest of the code, I can reuse the cached layers for faster builds

Build image with cache



With no change dependencies



Check to see if data are add to the database

```
mysql> select *from todo_items
    -> ;
ERROR 1046 (3D000): No database selected
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| todos              |
+--------------------+
5 rows in set (0.00 sec)

mysql> use todos;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from todo_items;
+--------------------------------------+-------+-----------+
| id                                   | name  | completed |
+--------------------------------------+-------+-----------+
| 272efa7f-68e3-4633-87c4-b739ab3ded43 | hi    |         0 |
| df2fb5c8-7dd8-4810-b1e4-89f1e60ebfff | hello |         0 |
+--------------------------------------+-------+-----------+
2 rows in set (0.01 sec)

mysql>
```

App running on compose