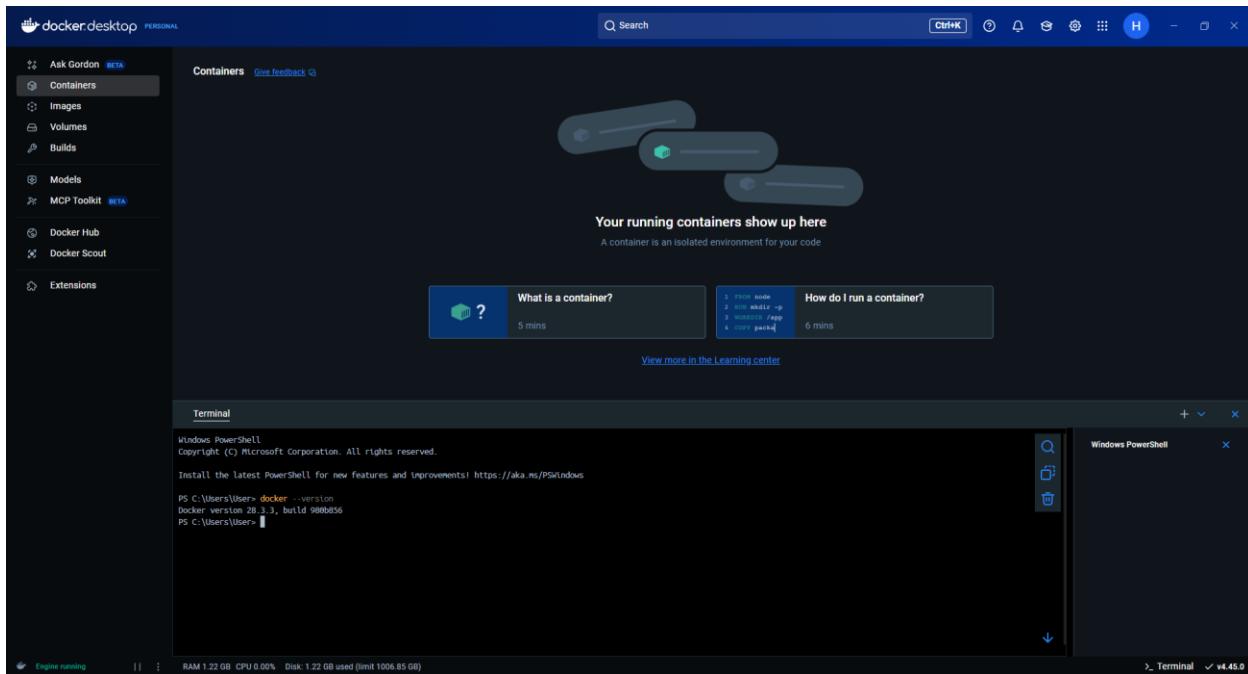


Nguyễn Việt Hoàng

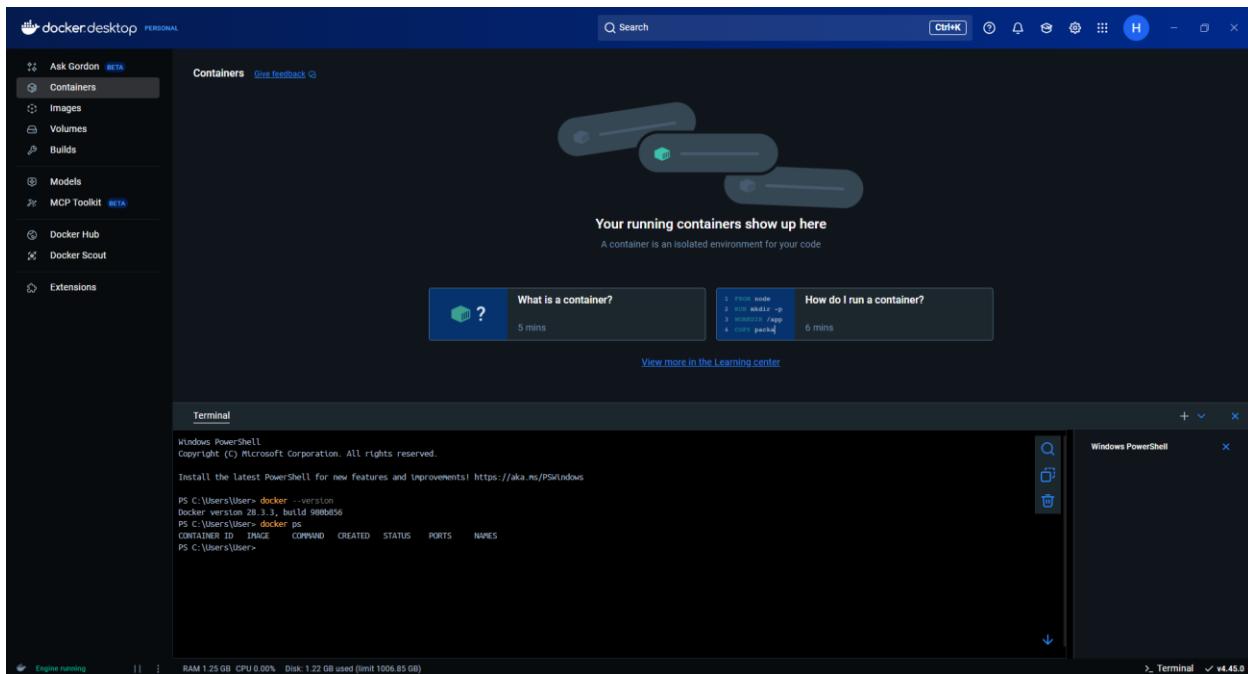
B22DCVT214

1. Docker Basic Commands

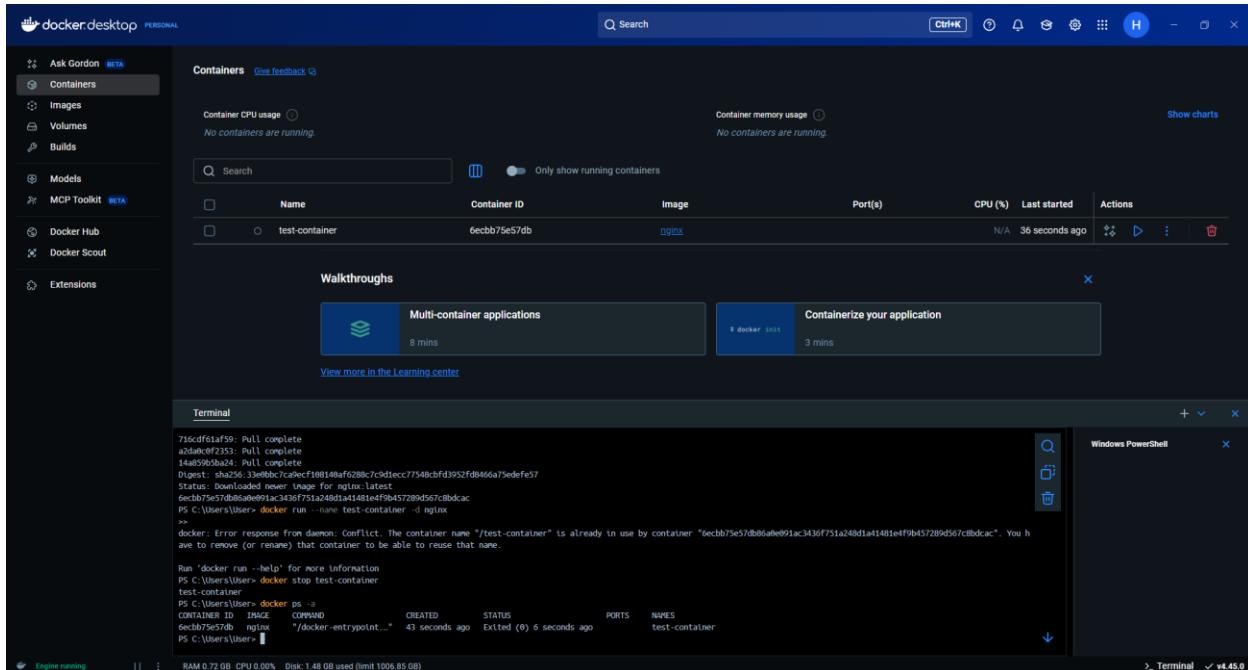
Check Docker version with docker --version



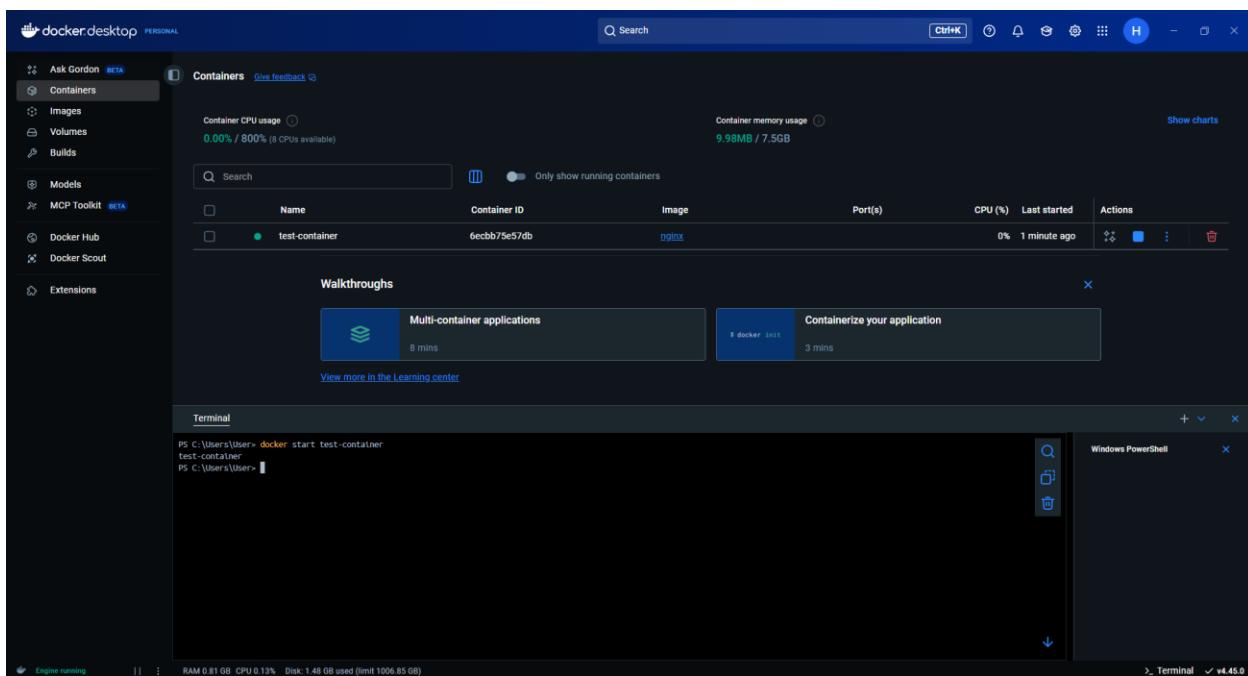
List running containers with docker ps



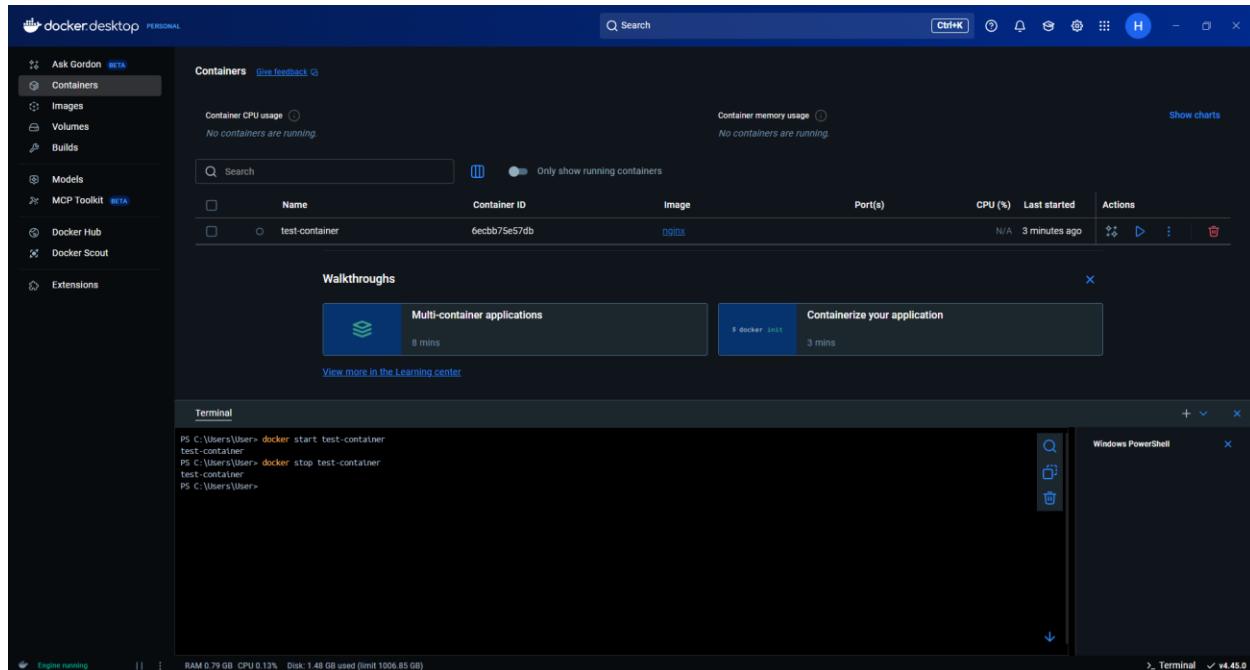
List all containers (including stopped) with docker ps -a



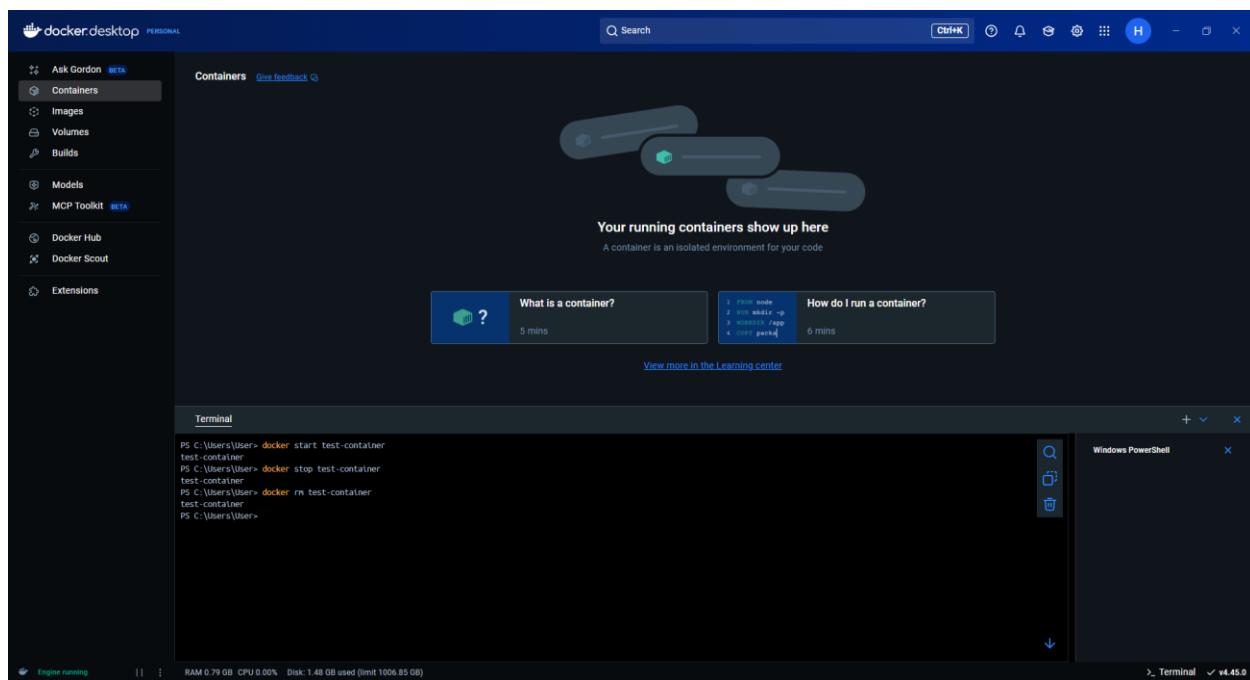
Start a container



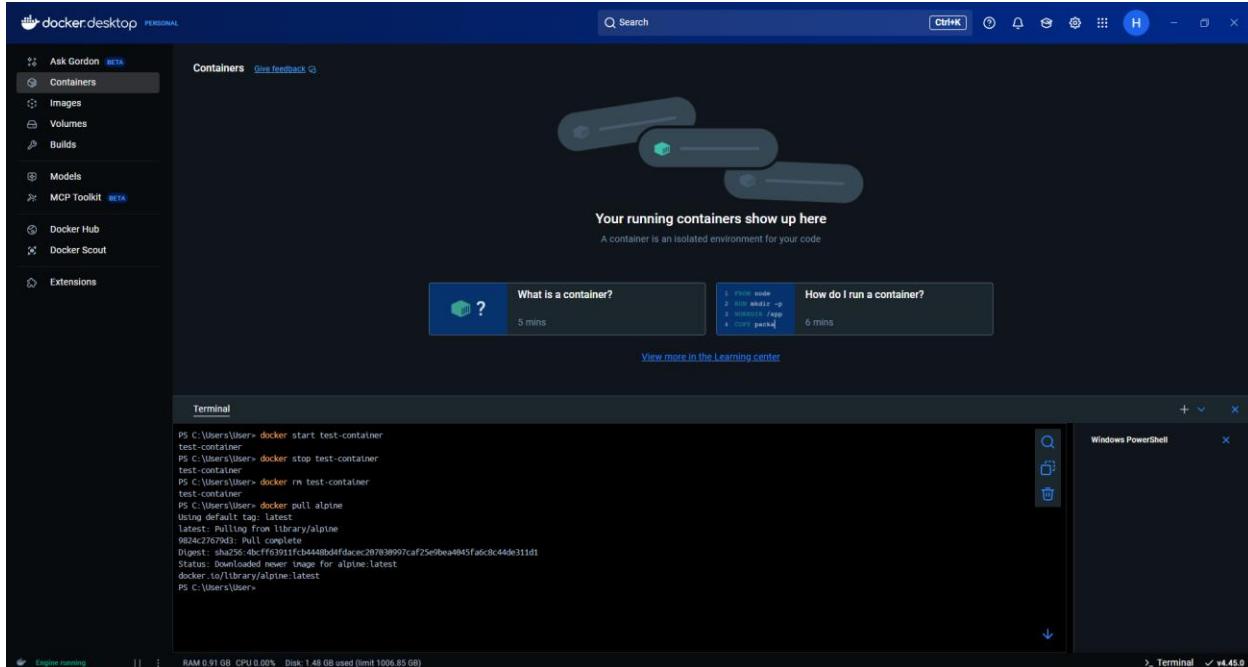
Stop a container



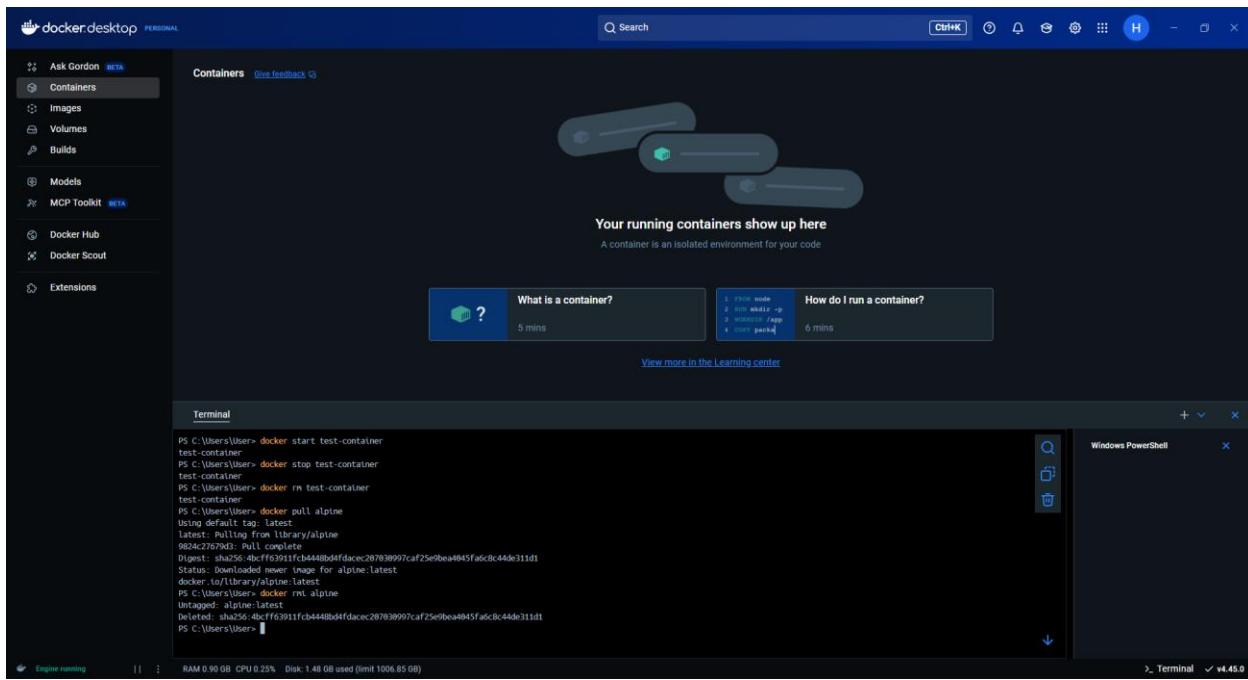
Remove a container



Pull an image

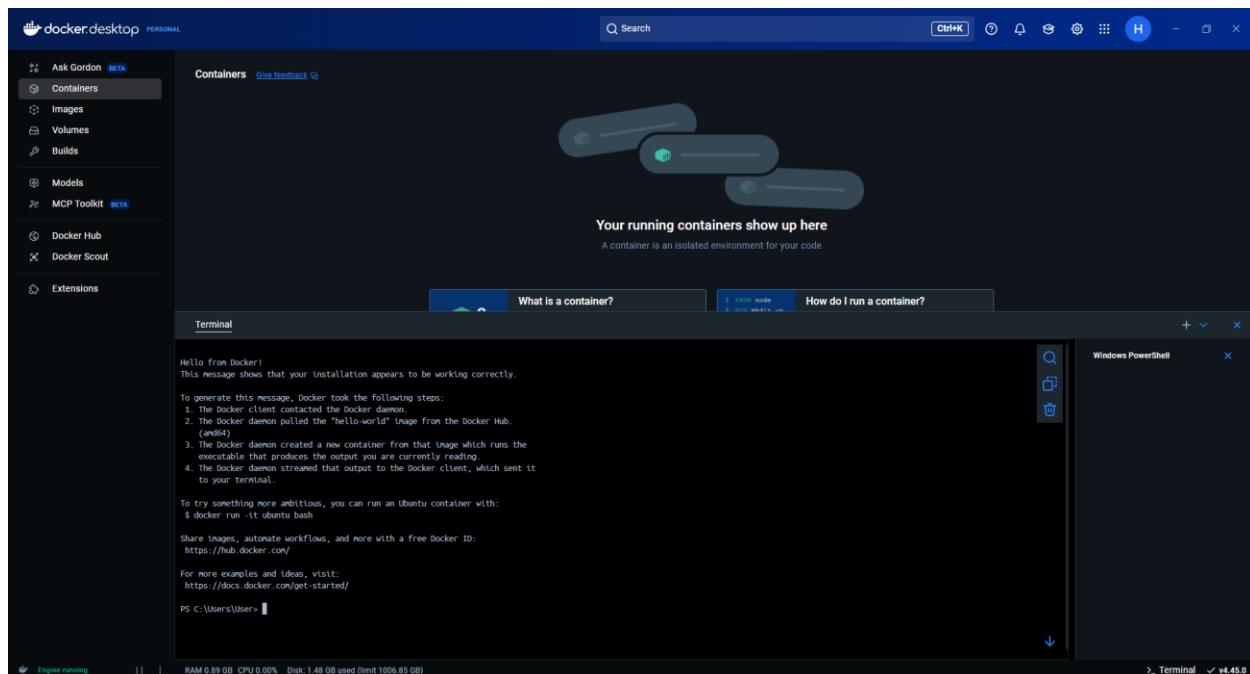


Remove an image with docker rmi

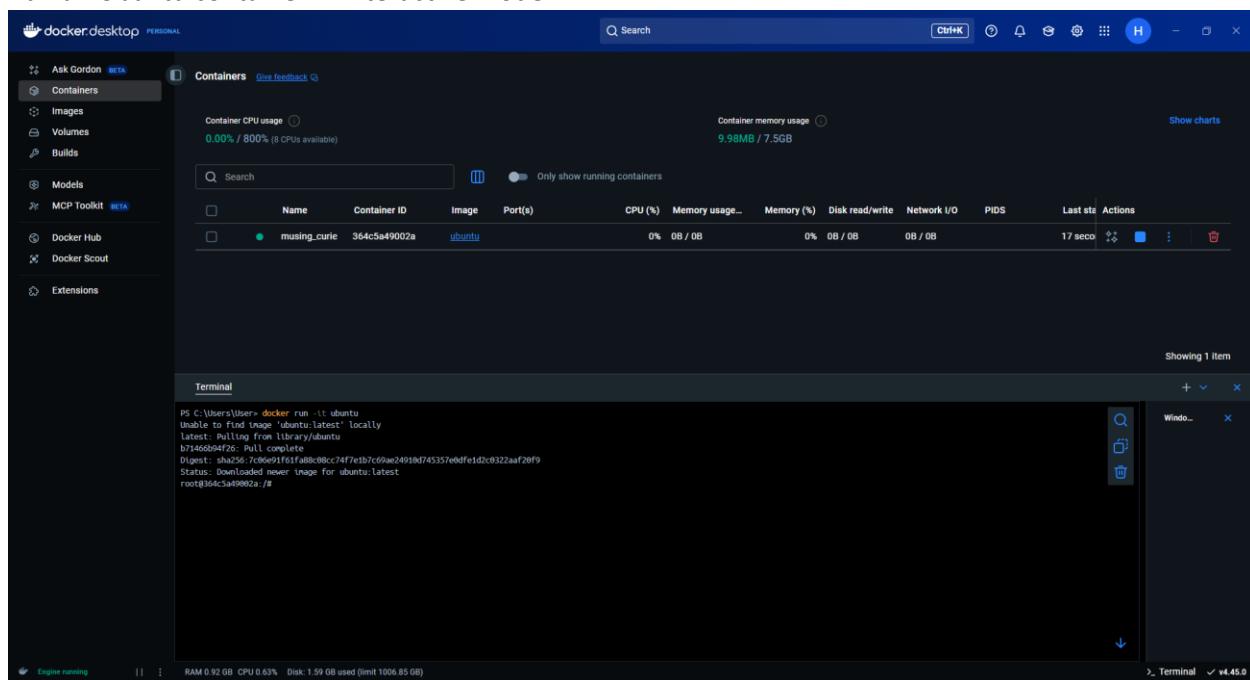


2. Docker Run

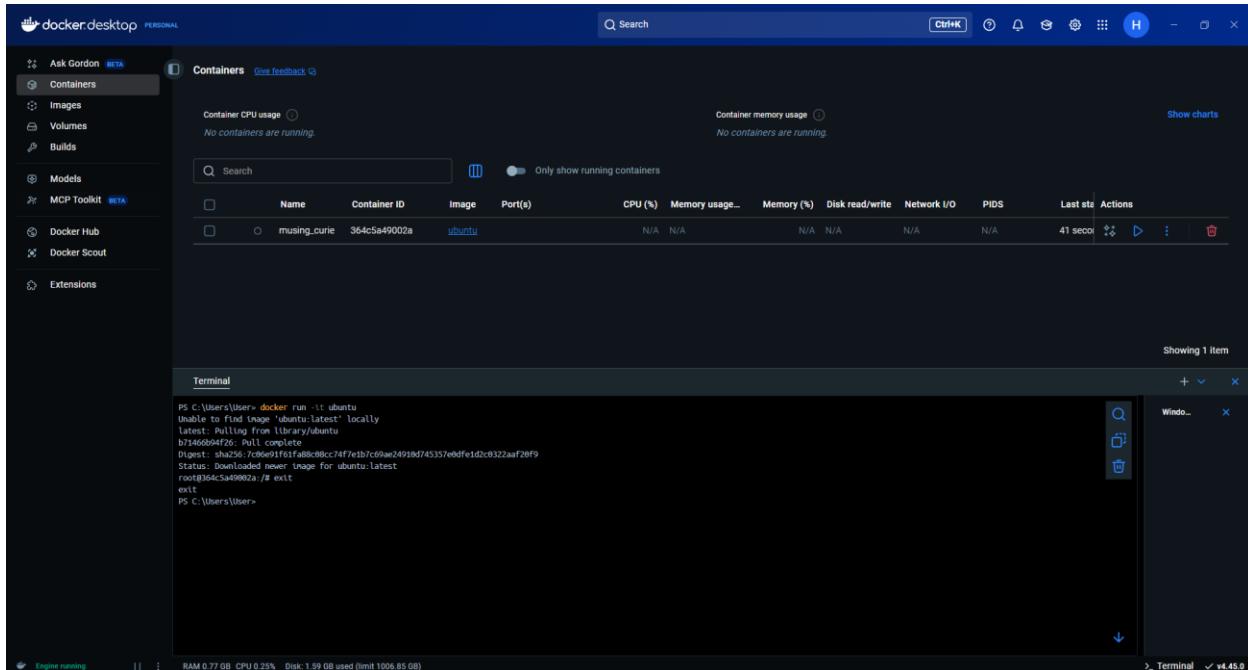
Run the hello-world container



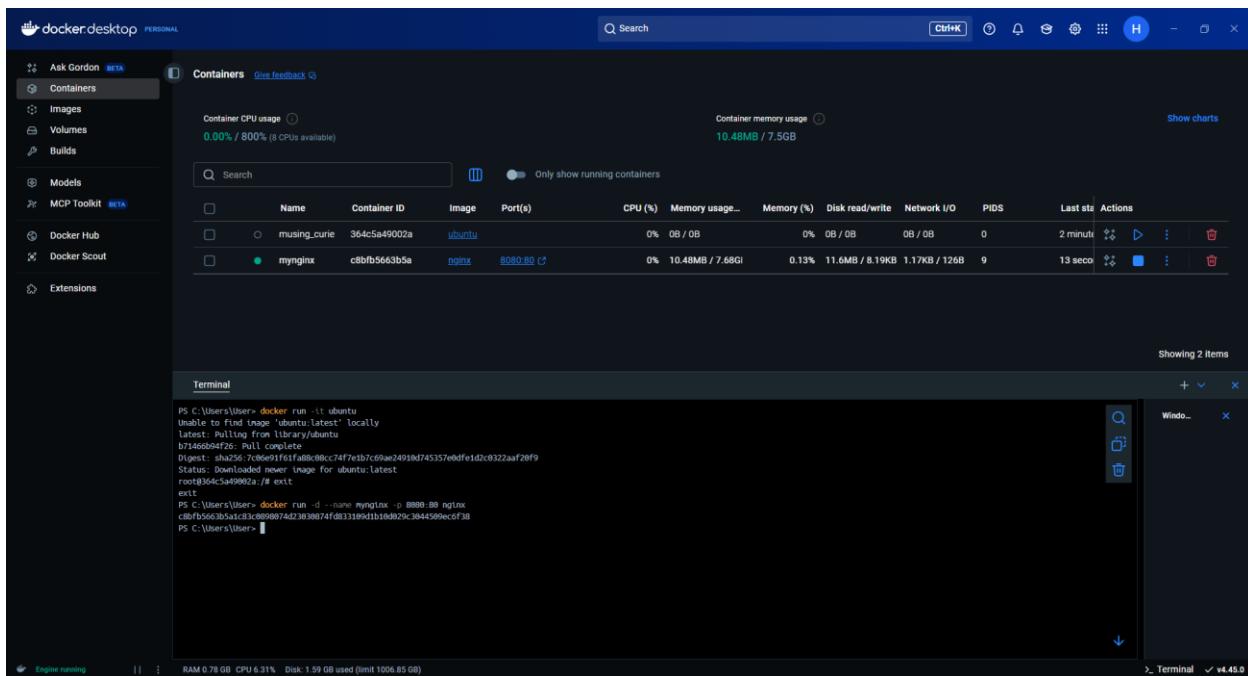
Run an Ubuntu container in interactive mode



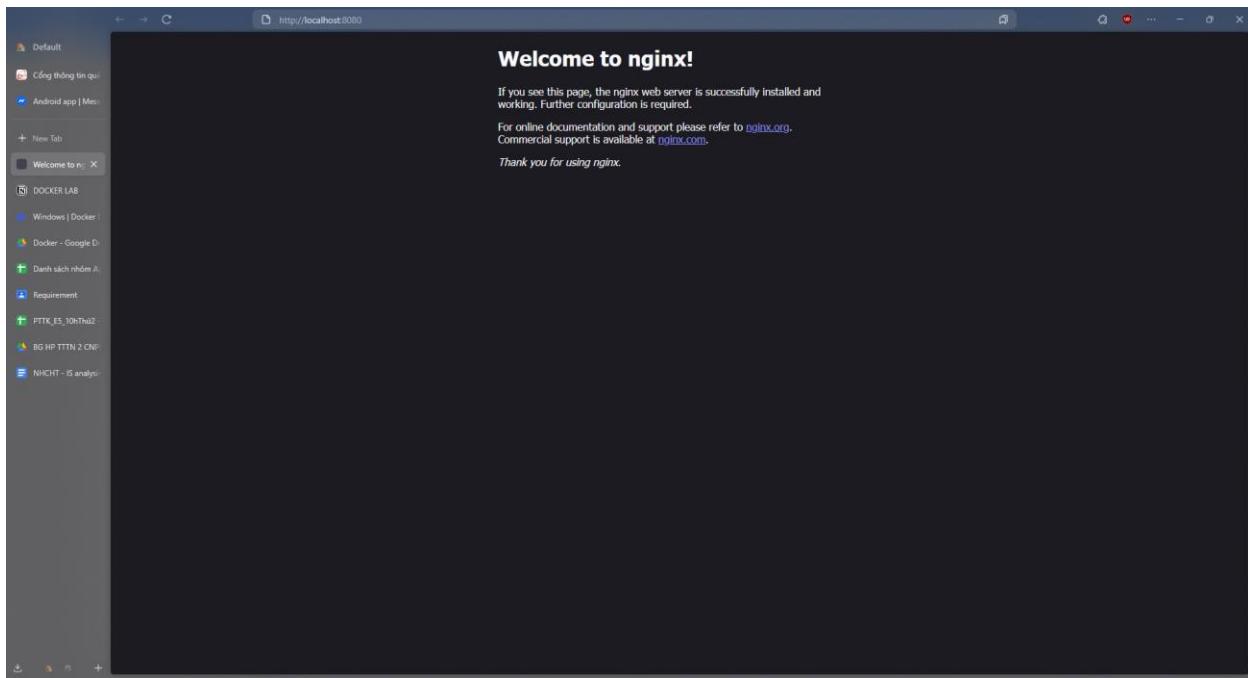
Exit Ubuntu container



Run an Nginx container mapping host port 8080→ container port 80



Access Nginx via browser at <http://localhost:8080>



3. Docker Environment Variables

Run MySQL container with `e MYSQL_ROOT_PASSWORD=my-secret-pw`

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last std	Actions
musing.curie	364c5a49002a	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	9 minutes	[More]
mynginx	c8bf85663b5a	nginx	8080:80	0%	9.07MB / 7.68GB	0.12%	11.6MB / 12.3KB	3.14KB / 2.15KB	9	7 minutes	[More]
mymysql	10f57185d7ca	mysql		1.9%	428.5MB / 7.68Gi	5.45%	65.6MB / 284MB	872B / 126B	35	2 minutes	[More]

```
PS C:\Users\User> docker run -d --name mymysql -e MYSQL_ROOT_PASSWORD=my-secret-pw mysql
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
4eaefabace0c: Pull complete
59867b754dc4: Pull complete
f1c14c7ff76f: Pull complete
3090a2a22c2c: Pull complete
9efffcfd8d91a3: Pull complete
5525b1bd205d: Pull complete
fc513be88017: Pull complete
a1bccead18c7: Pull complete
33233a3a2a20: Pull complete
50786f9a80d5: Pull complete
Digest: sha256:439fb3b944dc559a0e76c4e5c4865c97e5ba4d4007db32c48ac58d55c3869916
Status: Downloaded newer image for mysql:latest
10f57185d7ca: Pull complete
10f57185d7ca: Digest: sha256:439fb3b944dc559a0e76c4e5c4865c97e5ba4d4007db32c48ac58d55c3869916
Status: Downloaded newer image for mysql:latest
PS C:\Users\User>
```

Verify variables with `docker inspect <container_id>`

Docker Desktop interface showing three running containers:

- Containers**:
 - Container CPU usage: 1.40% / 800% (8 CPUs available)
 - Container memory usage: 437.57MB / 7.5GB
- Search**:
 - Only show running containers
- Table**:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last stat	Actions
musning_curiel	364c5a49002a	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	14 minu	[refresh] [stop] [inspect] [remove]
mynginx	c8fbfb5663b5a	nginx	8080:80	0%	9.07MB / 7.68GB	0.12%	11.6MB / 12.3KB	3.14KB / 2.15KB	9	12 minu	[refresh] [stop] [inspect] [remove]
mymysql	10f57185d7ca	mysql		1.47%	428.7MB / 7.68GB	5.45%	65.6MB / 285MB	8728 / 1268	35	6 minut	[refresh] [stop] [inspect] [remove]
- Terminal**:

```
PS C:\Users\user> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
10f57185d7ca mysql "docker-entrypoint.s..." 5 minutes ago Up 5 minutes 3386/tcp, 33860/tcp 0.0.0.0:8880->88/tcp, [::]:8880->88/tcp mynginx
c8fbfb5663b5a nginx "/docker-entrypoint...." 11 minutes ago Up 11 minutes 8080/tcp 0.0.0.0:80->80/tcp, [::]:80->80 mynginx
364c5a49002a ubuntu "/bin/bash" 13 minutes ago Exited (0) 12 minutes ago
PS C:\Users\user> docker inspect 10f57185d7ca
{
  "Id": "10f57185d7caed49ed9ba1ab5b5a4da540d154ff5e887db43d49b20c35cd8c85",
  "Created": "2025-08-30T06:42:37.757Z",
  "Path": "docker-entrypoint.sh",
  "Args": [
    "mysql"
  ],
  "State": {
    "Status": "running",
    "Running": true,
    "Paused": false,
    "Restarting": false,
    "Health": {
      "Status": "healthy"
    }
  },
  "Config": {
    "Image": "mysql:latest",
    "Cmd": [
      "mysqld"
    ],
    "ExposedPorts": {
      "3386/tcp": {}
    },
    "Env": [
      "MYSQL_ROOT_PASSWORD=mypassword",
      "MYSQL_DATABASE=sample",
      "MYSQL_USER=root",
      "MYSQL_PASSWORD=root"
    ],
    "Labels": {}
  },
  "HostConfig": {
    "Binds": [],
    "Mounts": []
  }
}
```

Run a custom container with user-defined env vars.

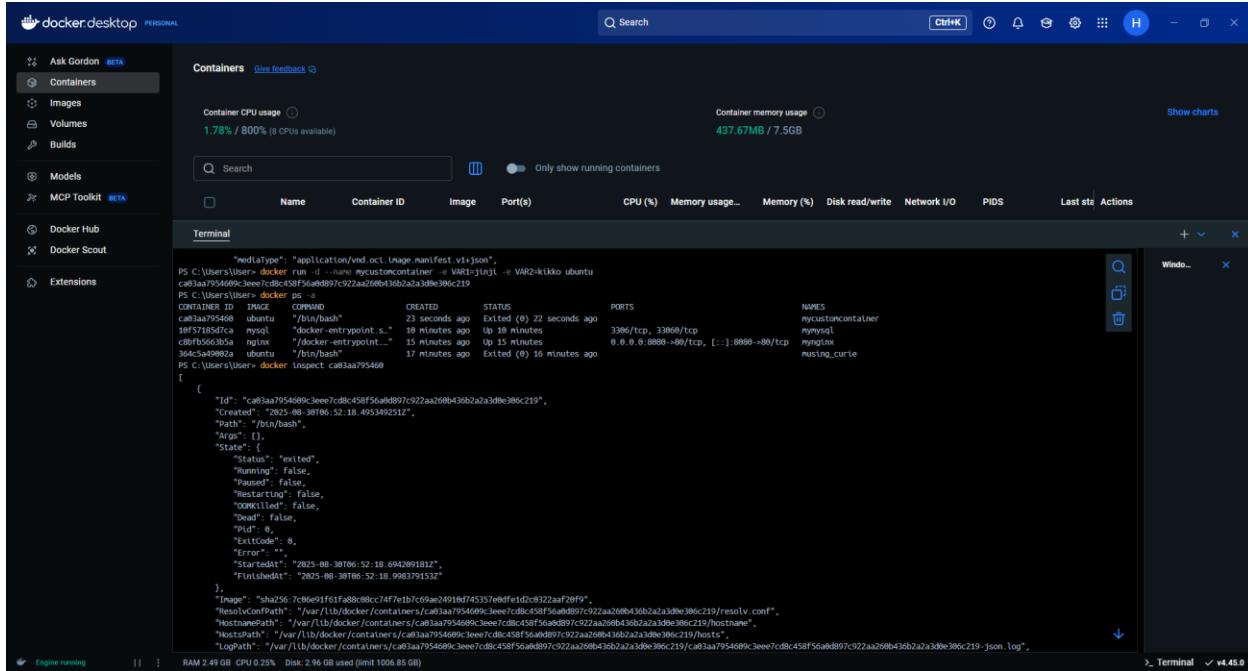
Docker Desktop interface showing four containers:

- Containers**:
 - Container CPU usage: 1.76% / 800% (8 CPUs available)
 - Container memory usage: 437.77MB / 7.5GB
- Search**:
 - Only show running containers
- Table**:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last stat	Actions
musning_curiel	364c5a49002a	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	17 minu	[refresh] [stop] [inspect] [remove]
mynginx	c8fbfb5663b5a	nginx	8080:80	0%	9.07MB / 7.68GB	0.12%	11.6MB / 12.3KB	3.14KB / 2.15KB	9	16 minu	[refresh] [stop] [inspect] [remove]
mymysql	10f57185d7ca	mysql		1.76%	428.7MB / 7.68GB	5.45%	65.6MB / 285MB	8728 / 1268	35	10 minu	[refresh] [stop] [inspect] [remove]
mycustomconta	ca03aa795469c3eeec7d8c458f56a8d897c922aa264b436b2a3d8e386c219	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	1 secon	[refresh] [stop] [inspect] [remove]	
- Terminal**:

```
PS C:\Users\user> docker run -d --name mycustomconta -e VAR1=junju -e VAR2=klikko ubuntu
ca03aa795469c3eeec7d8c458f56a8d897c922aa264b436b2a3d8e386c219
PS C:\Users\user>
```

Verify variable



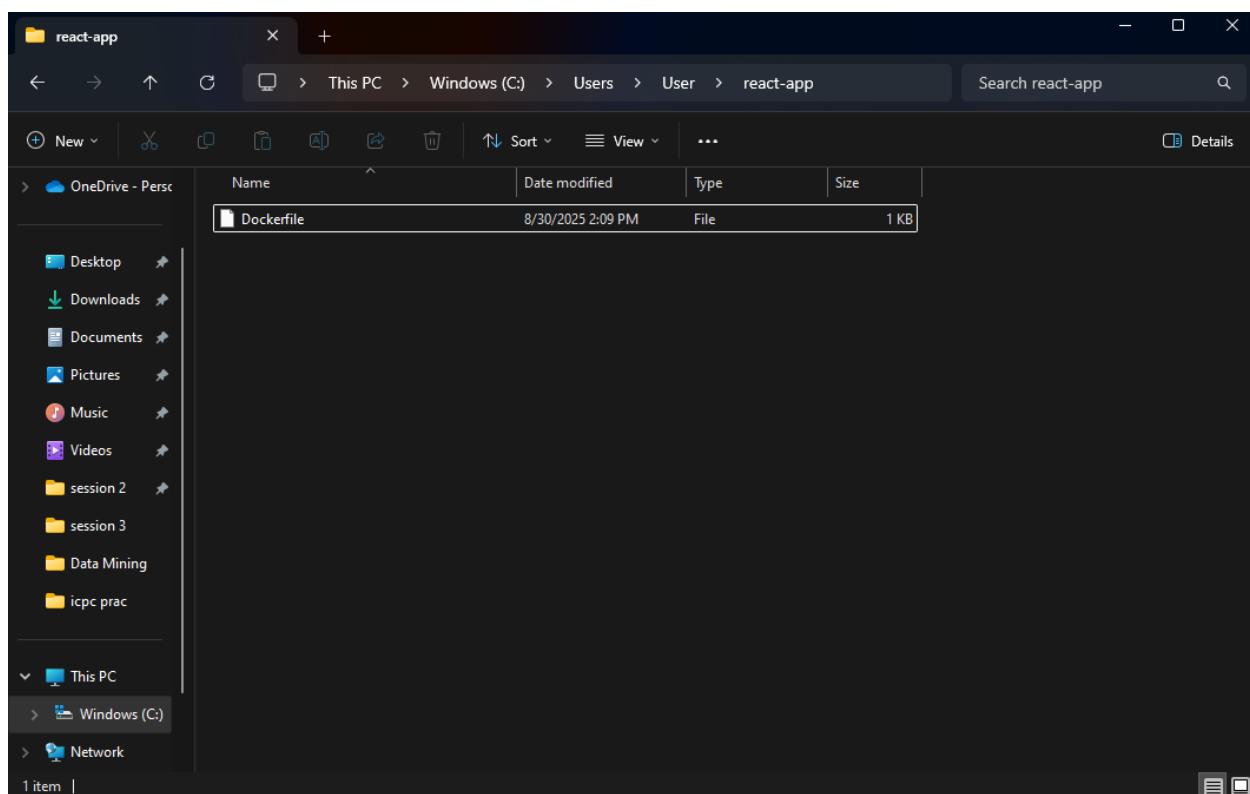
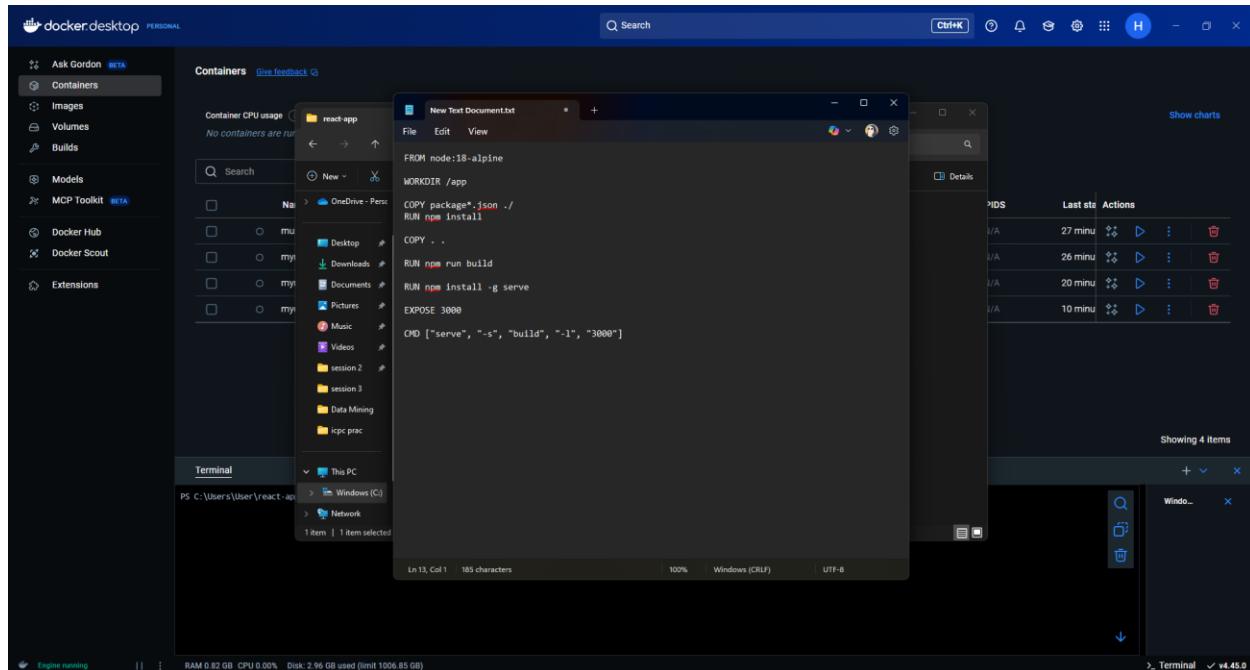
```

],
"Mounts": [],
"Config": {
  "Hostname": "ca03aa795460",
  "Domainname": "",
  "User": "",
  "AttachStdin": false,
  "AttachStdout": false,
  "AttachStderr": false,
  "Tty": false,
  "OpenStdin": false,
  "StdinOnce": false,
  "Env": [
    "VAR1=jinji",
    "VAR2=kikko",
    "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
  ],
}

```

4. Docker Images

Create a Dockerfile in the root of your app repo (React app).



Create react app

A screenshot of the Docker Desktop application interface on a Windows system. The main window shows a PowerShell session titled "Windows PowerShell" with the following command history:

```
Creating a new React app in C:\Users\User\react-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1323 packages in 2m

270 packages are looking for funding
  run 'npm fund' for details

Initialized a git repository.

Installing template dependencies using npm...
added 17 packages, and changed 1 package in 12s

270 packages are looking for funding
  run 'npm fund' for details

Removing template package using npm...

removed 1 package, and audited 1340 packages in 8s

270 packages are looking for funding
  run 'npm fund' for details

  9 vulnerabilities (3 moderate, 6 high)

To address all issues (including breaking changes), run:
  npm audit fix --force

Run 'npm audit' for details.

Created git commit.

Success! Created react-app at C:\Users\User\react-app
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

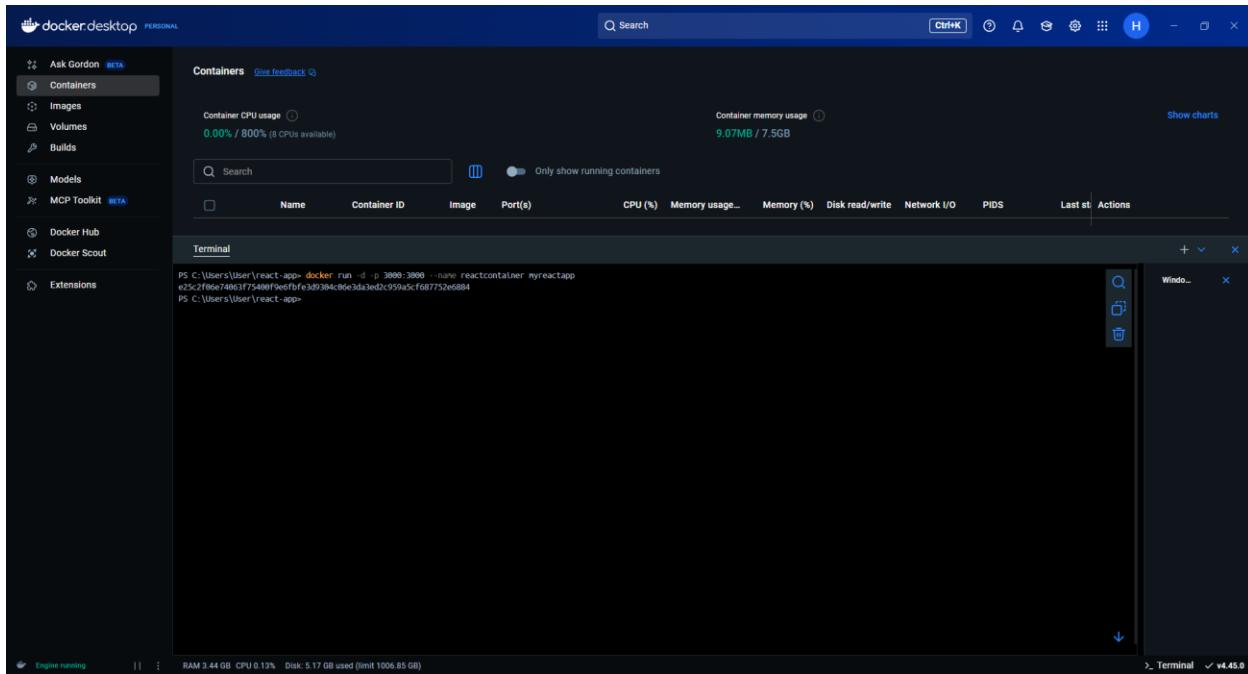
  npm test
    Starts the test runner.
```

The Docker Desktop sidebar on the left includes sections for Ask Gordon, Containers, Images, Volumes, Builds, Models, MCP Toolkit, Docker Hub, Docker Scout, and Extensions. The status bar at the bottom indicates "Engine running".

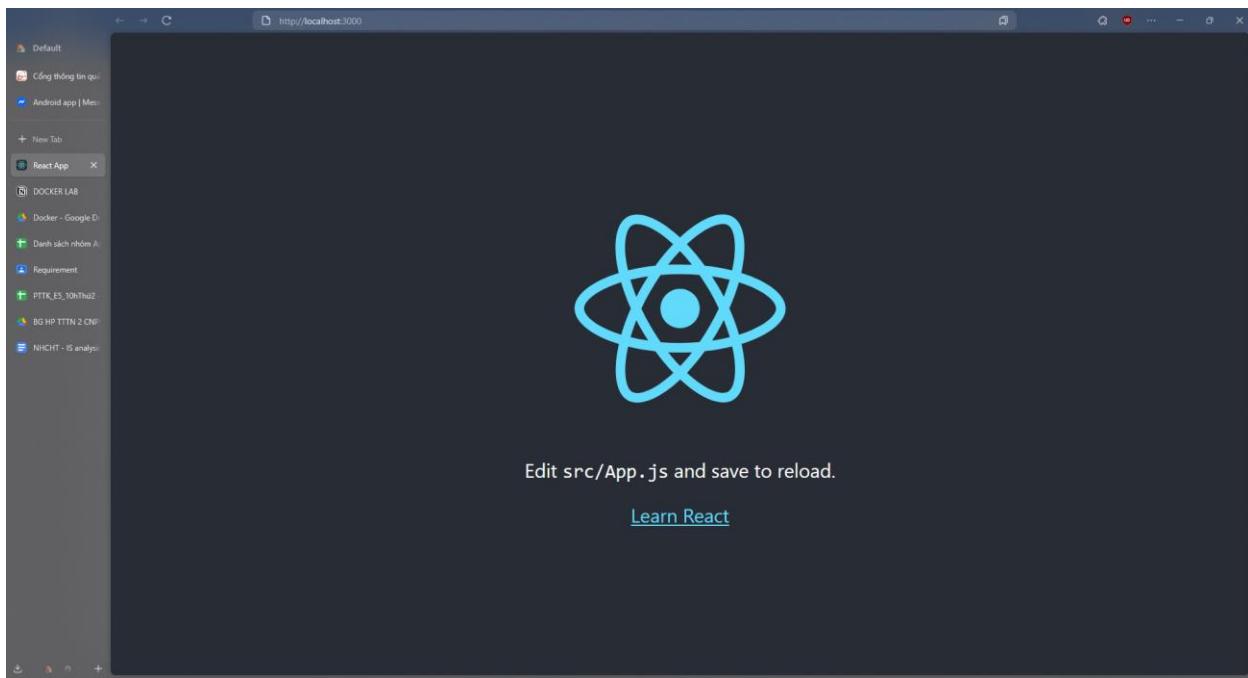
Build docker app with docker build -t myreactapp .

The screenshot shows the Docker Desktop interface. On the left, there's a sidebar with sections like Ask Gordon, Containers, Images, Volumes, Builds, Models, MCP Toolkit, Docker Hub, Docker Scout, and Extensions. The 'Containers' section is currently selected. In the main area, there are two charts: 'Container CPU usage' and 'Container memory usage', both showing 'No containers are running'. Below these is a search bar and a filter option 'Only show running containers'. A table header row includes columns for Name, Container ID, Image, Port(s), CPU (%), Memory usage..., Memory (%), Disk read/write, Network I/O, PIDS, Last stats, and Actions. Under the 'Terminal' tab, a command-line interface is displayed with several log entries. At the bottom, a status bar shows system information like RAM, CPU, and Disk usage, along with a terminal icon and a version number.

Run container

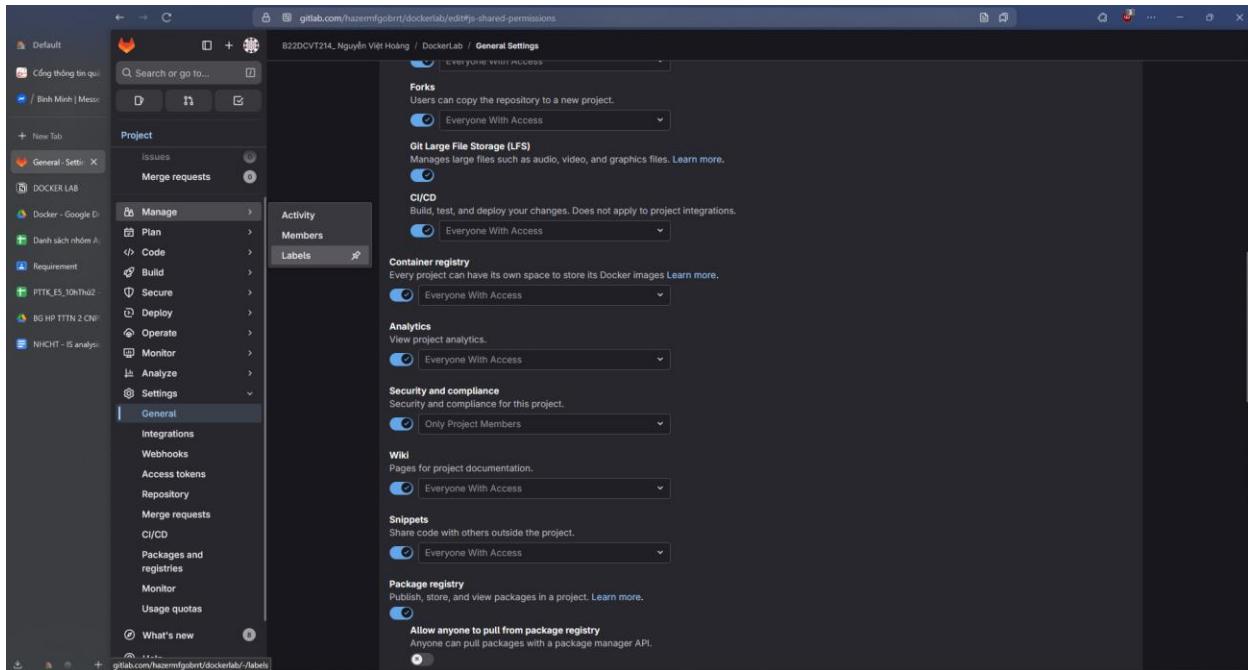


Test on browser



5. Docker Registry (GitLab Container Registry)

Enable GitLab Container Registry for your project.



Login GitLab

The screenshot shows the Docker Desktop application. On the left, there's a sidebar with 'Containers', 'Images', 'Volumes', 'Builds', 'Models', and 'MCP Toolkit'. The 'Containers' tab is active, showing a table of running containers:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last stat	Actions
musing_curi	364c5a49002a	ubuntu		N/A	N/A	N/A	N/A	N/A	N/A	1 hour ago	⋮
mynginx	c8fb5663b5a	nginx	8080:80	N/A	N/A	N/A	N/A	N/A	N/A	1 hour ago	⋮
mymysql	10f57185d7ca	mysql		N/A	N/A	N/A	N/A	N/A	N/A	1 hour ago	⋮
mycustomconta	ca03aa795460	ubuntu		N/A	N/A	N/A	N/A	N/A	N/A	58 minutes ago	⋮
reactcontainer	e25c2f06e740	myreactapp	3000:3000	N/A	N/A	N/A	N/A	N/A	N/A	20 minutes ago	⋮

Below the table is a 'Terminal' window showing a command-line session:

```

Password: Error response from daemon: Get "https://registry.gitlab.com/v2/": unauthorized: HTTP Basic: Access denied. If a password was provided for Git authentication, the password was incorrect or you're required to use a token instead of a password. If a token was provided, it was either incorrect, expired, or improperly scoped. See https://gitlab.com/help/user/profile/account/two_factor_authentication_trouble-shooting#error-http-basic-access-denied-if-a-password-was-provided-for-git-authentication
PS C:\Users\user\react-app> docker login registry.gitlab.com
Username: hazernfgobr
Password: Error response from daemon: Get "https://registry.gitlab.com/v2/": unauthorized: HTTP Basic: Access denied. If a password was provided for Git authentication, the password was incorrect or you're required to use a token instead of a password. If a token was provided, it was either incorrect, expired, or improperly scoped. See https://gitlab.com/help/user/profile/account/two_factor_authentication_trouble-shooting#error-http-basic-access-denied-if-a-password-was-provided-for-git-authentication
PS C:\Users\user\react-app> docker login registry.gitlab.com -u hazernfgobr -p "g1pat-BqHtFlzRjxZ-Y1QlR08k2869Qj0mhdJ32Cw.81.128tkoksd"
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
Login succeeded.
PS C:\Users\user\react-app>

```

Tag local image

Containers

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last stat	Actions
musing_curi	364c5a49002a	ubuntu		N/A	N/A	N/A	N/A	N/A	N/A	2 hours	⋮
mynginx	c8bfbd663b5a	nginx	8080:80	N/A	N/A	N/A	N/A	N/A	N/A	2 hours	⋮
mymysql	10f57185d7ca	mysql		N/A	N/A	N/A	N/A	N/A	N/A	1 hour	⋮
mycustomconta	ca03aa795460	ubuntu		N/A	N/A	N/A	N/A	N/A	N/A	1 hour	⋮
reactcontainer	e25cf06e740	myreactapp	3000:3000	N/A	N/A	N/A	N/A	N/A	N/A	41 minu	⋮

Showing 5 items

Terminal

```
PS C:\Users\user\react-app> docker tag myreactapp registry.gitlab.com/hazernfgobr/dockerlab/myreactapp:latest
PS C:\Users\user\react-app>
```

Push the image

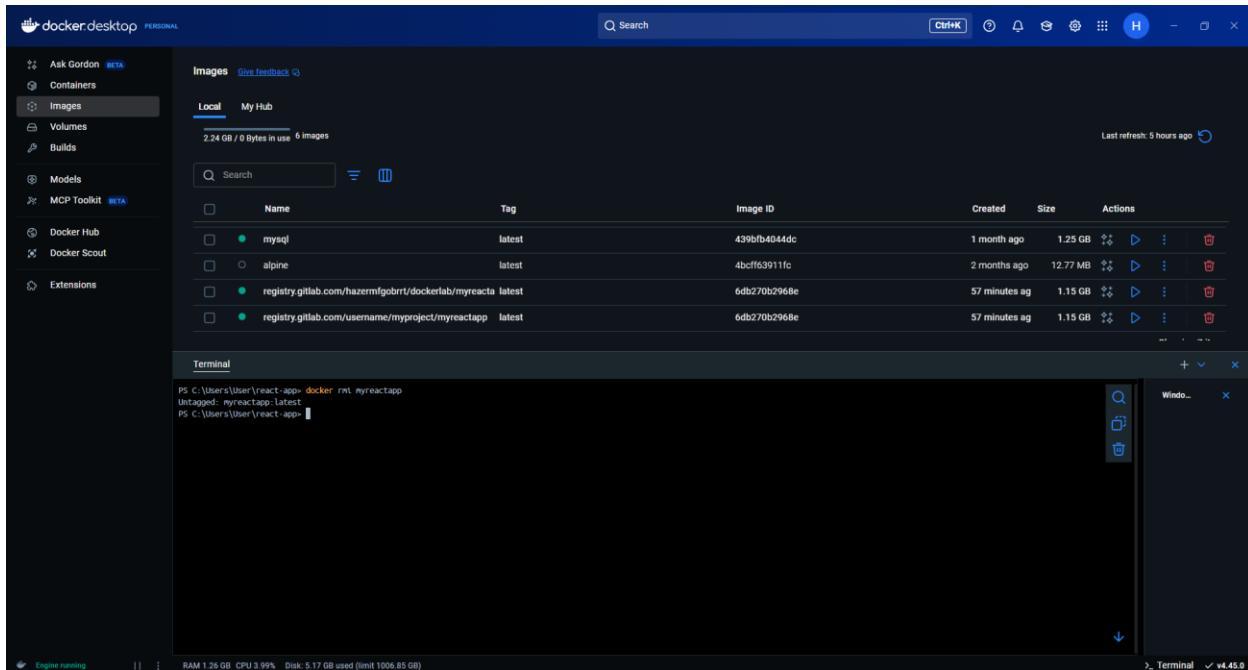
Images

Name	Tag	Image ID	Created	Size	Actions
nginx	latest	33e0bbc7ca9e	17 days ago	279.18 MB	⋮
hello-world	latest	a0dfb02aac21	22 days ago	20.34 KB	⋮
ubuntu	latest	7c06e91f61fa	1 month ago	117.31 MB	⋮
mysql	latest	439bf4044dc	1 month ago	1.25 GB	⋮

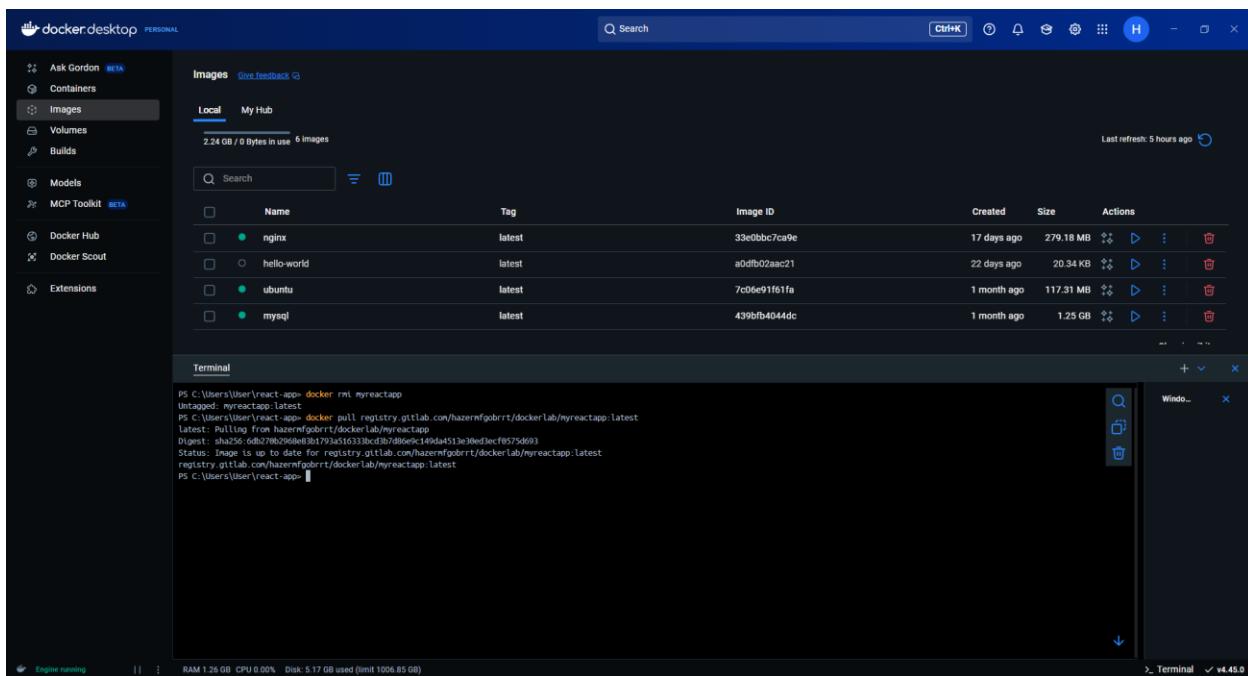
Terminal

```
PS C:\Users\user\react-app> docker push registry.gitlab.com/hazernfgobr/dockerlab/myreactapp:latest
The push refers to repository [registry.gitlab.com/hazernfgobr/dockerlab/myreactapp]
74d53f1f4728a: Pushed
b481d7f76f73b: Pushed
3a9233a33334d: Pushed
7eb337c269d4d: Pushed
45b613f7512d2: Pushed
f18232174b9: Pushed
2219a7d48d4e: Pushed
4d56b2482949: Pushed
1557f13333333: Pushed
1e54d39c9e5: Pushed
dd71dde834b5: Pushed
Latest: digest: sha256:6db27862968e83b179a51633b3bcd7d86e9c149da4513e30ed3ecf0575d693 size: 856
PS C:\Users\user\react-app>
```

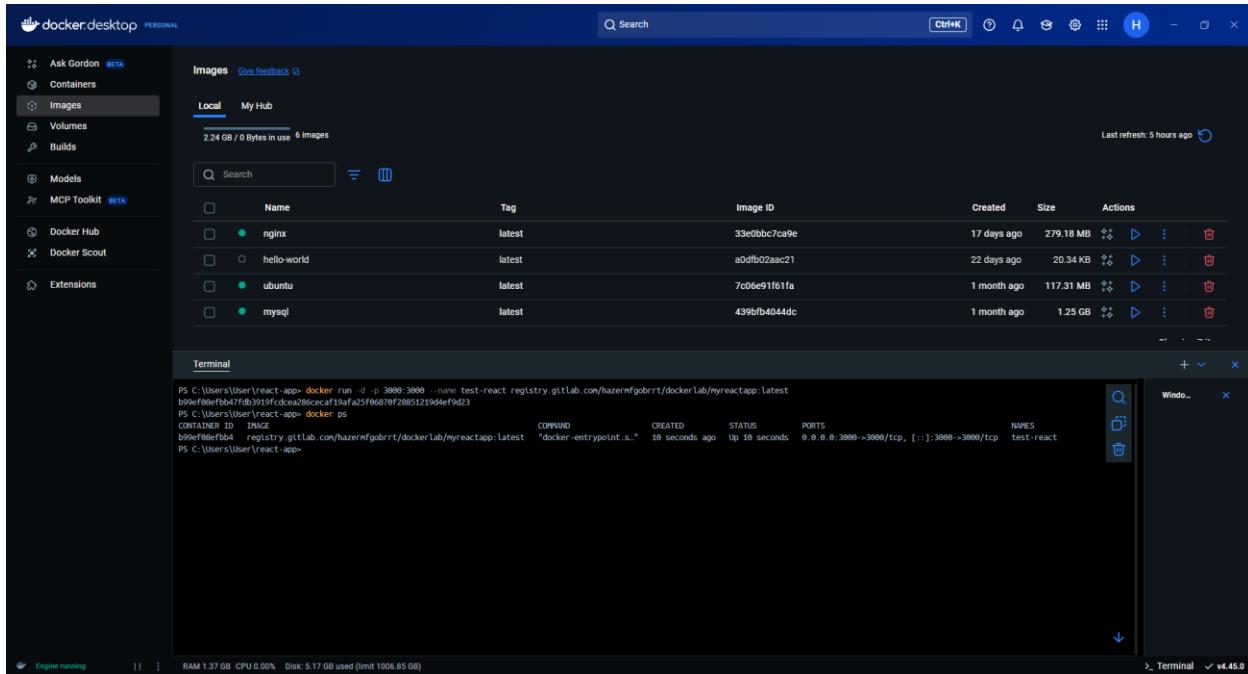
Remove local image



Pull the image back

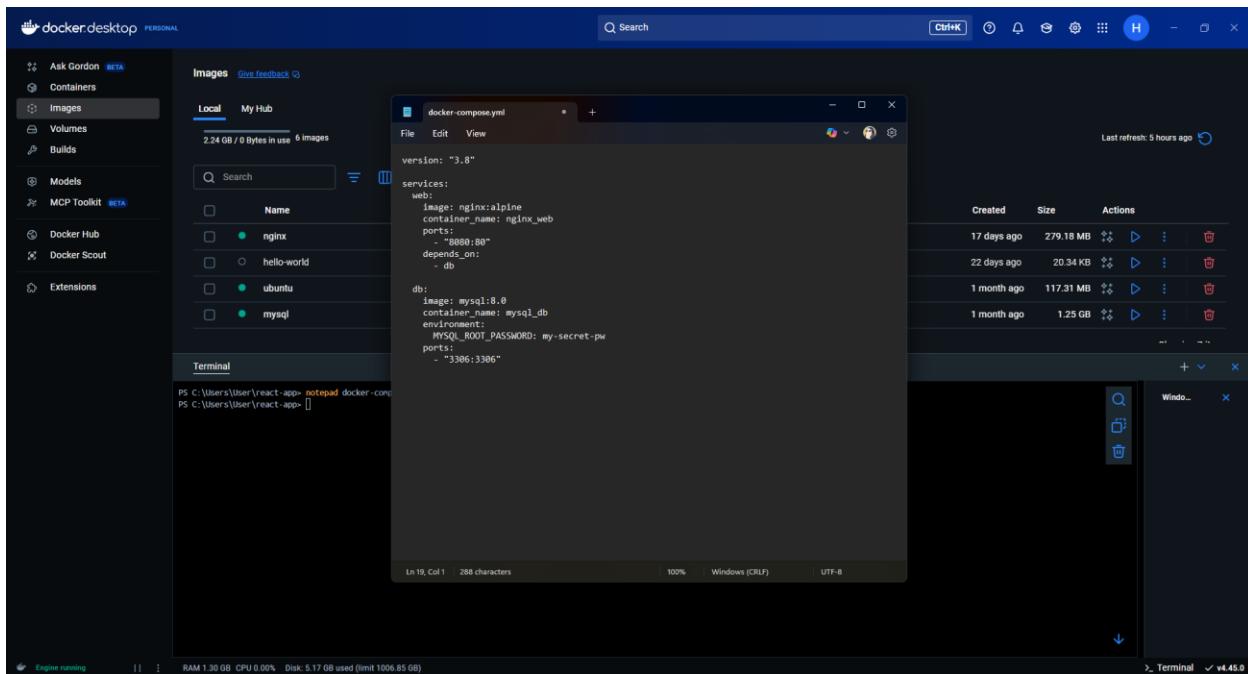


Run container

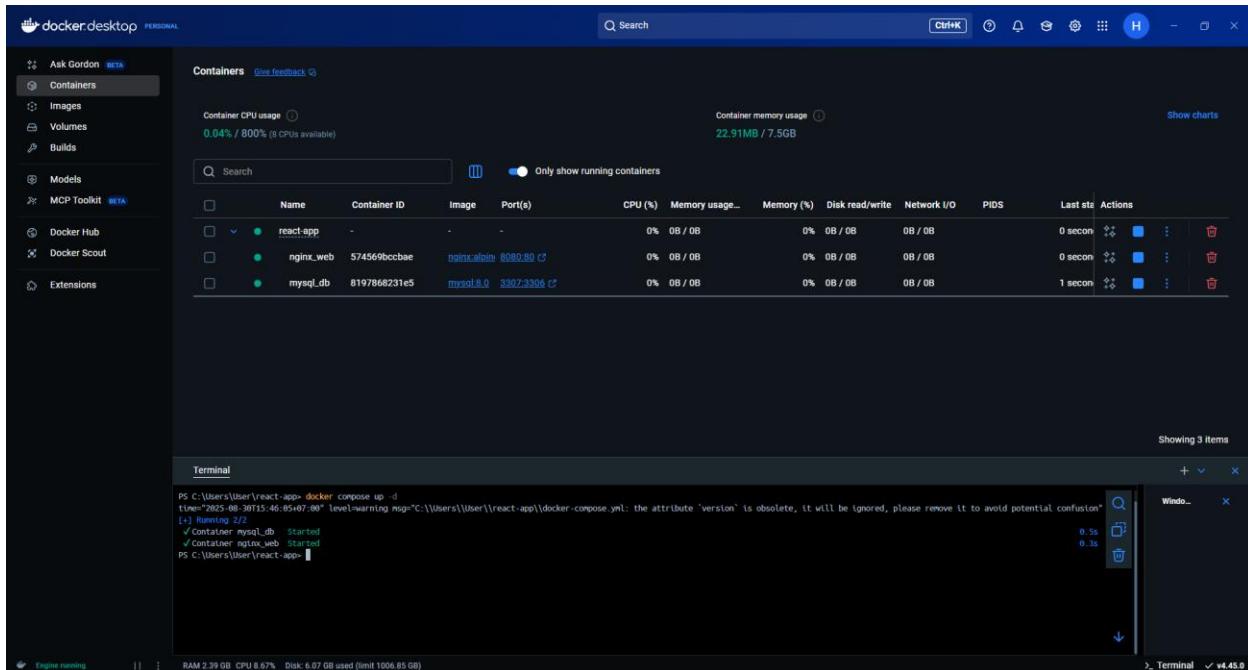


6. Docker Compose (Part 1 – Basics)

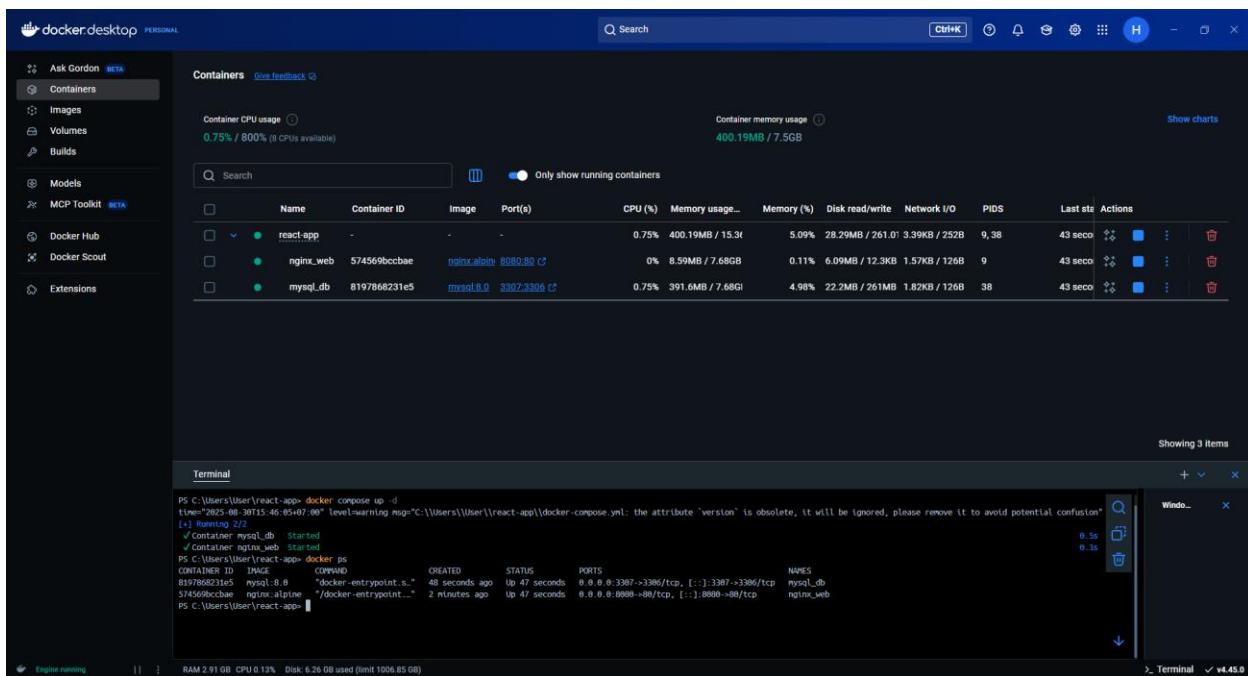
Create docker-compose.yml with services



Run docker compose up



Verify both containers are running with docker ps



7. Docker Storage

Run MySQL with a named volume

Docker Desktop interface showing the Containers tab. The sidebar includes Ask Gordon (BETA), Containers, Images, Volumes, Builds, Models, MCP Toolkit (BETA), Docker Hub, Docker Scout, and Extensions. The main area displays a table of running containers:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last st.	Actions
musing_curi	364c5a49002a	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	2 hours	[Actions]
mynginx	c8fb5663b5a	nginx	8080:80	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	2 hours	[Actions]
mycustomconta	ca03aa795460	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	2 hours	[Actions]
reactcontainer	e25c2f06e740	myreactapp	3000:3000	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	1 hour	[Actions]
test-react	b99ef08ebfb4	hazermfsql	3000:3000	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	32 min	[Actions]
test_db	19da65b6c988	mysql:8.0		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	1 sec	[Actions]
react-app	-	-	-	0.64%	368.48MB / 15.36	4.68%	28.29MB / 265.01	3.39KB / 252B	9, 37	10 min	[Actions]
nginx_web	574569bccbae	nginx:alpine	8080:80	0%	7.88MB / 7.68GB	0.1%	6.09MB / 12.3KB	1.57KB / 126B	9	10 min	[Actions]

The terminal window shows the command used to run the MySQL container:

```
PS C:\Users\user\react-app> docker run -d --name test_db -e MYSQL_ROOT_PASSWORD=my-secret-pw -v mysql_data:/var/lib/mysql mysql:8.0
19da65b6c988973e2d67368c4465554881cb69bb24dc1b55eeef9d7866d7
PS C:\Users\user\react-app>
```

System status at the bottom: RAM 1.45 GB, CPU 14.71%, Disk: 6.06 GB used (limit 1006.85 GB).

Create a table

Docker Desktop interface showing the Containers tab. The sidebar includes Ask Gordon (BETA), Containers, Images, Volumes, Builds, Models, MCP Toolkit (BETA), Docker Hub, Docker Scout, and Extensions. The main area displays a table of running containers:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last st.	Actions
musing_curi	364c5a49002a	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	2 hours	[Actions]
mynginx	c8fb5663b5a	nginx	8080:80	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	2 hours	[Actions]
mycustomconta	ca03aa795460	ubuntu		0%	0B / 0B	0%	0B / 0B	0B / 0B	0	2 hours	[Actions]
reactcontainer	e25c2f06e740	myreactapp	3000:3000	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	1 hour	[Actions]
test-react	b99ef08ebfb4	hazermfsql	3000:3000	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	34 min	[Actions]
test_db	19da65b6c988	mysql:8.0		0.77%	354.9MB / 7.68Gi	4.51%	35.9MB / 267MB	1.17KB / 126B	38	2 minut	[Actions]
react-app	-	-	-	0%	0B / 0B	0%	0B / 0B	0B / 0B	0, 0	12 min	[Actions]
nginx_web	574569bccbae	nginx:alpine	8080:80	0%	7.88MB / 7.68GB	0%	6.09MB / 12.3KB	1.57KB / 126B	9	12 min	[Actions]

The terminal window shows the MySQL command used to create a database and table:

```
PS C:\Users\user\react-app> docker exec -it test_db mysql -uroot -pmy-secret-pw -e "CREATE DATABASE IF NOT EXISTS testdb; USE testdb; CREATE TABLE users(id INT, name VARCHAR(50));"
mysql: [Warning] Using a password on the command line interface can be insecure.
PS C:\Users\user\react-app> docker exec -it test_db mysql -uroot -pmy-secret-pw -e "USE testdb; SHOW TABLES;"
mysql: [Warning] Using a password on the command line interface can be insecure.
+-----+
| Tables_in_testdb |
+-----+
| users           |
+-----+
PS C:\Users\user\react-app>
```

System status at the bottom: RAM 1.70 GB, CPU 0.13%, Disk: 6.26 GB used (limit 1006.85 GB).

Stop and remove the container

Docker Desktop interface showing the following containers:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last st.	Actions
mysql_db	8197868231e5	mysql:8.0	3307:3306	N/A	N/A	N/A	N/A	N/A	N/A	13 min.	⋮
nginx_web	574569bccbae	nginx:alpine	8080:80	N/A	N/A	N/A	N/A	N/A	N/A	13 min.	⋮
reactcontainer	e25cd2f06e740	myreactor	3000:3000	N/A	N/A	N/A	N/A	N/A	N/A	1 hour	⋮
test-react	b99ef08efbb4	hazermfgql	3000:3000	N/A	N/A	N/A	N/A	N/A	N/A	35 min.	⋮
react-app	-	-	-	N/A	N/A	N/A	N/A	N/A	N/A	13 min.	⋮
musing_carie	364c5a49002a	ubuntu	-	N/A	N/A	N/A	N/A	N/A	N/A	2 hours	⋮
mynginx	c8bfbd563b5a	nginx	8080:80	N/A	N/A	N/A	N/A	N/A	N/A	2 hours	⋮
mycustomconta	ca03aa795460	ubuntu	-	N/A	N/A	N/A	N/A	N/A	N/A	2 hours	⋮

Terminal output:

```
PS C:\Users\user\react-app> docker stop test_db
test_db
PS C:\Users\user\react-app> docker rm test_db
test_db
PS C:\Users\user\react-app>
```

Engine running | RAM 1.37 GB CPU 5.16% Disk: 6.26 GB used (limit 1006.85 GB) | > Terminal v4.45.0

Start a new MySQL container using the same volume

Docker Desktop interface showing the following containers:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Network I/O	PIDS	Last st.	Actions
reactcontainer	e25cd2f06e40	myreactor	3000:3000	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	37 min.	⋮
test-react	b99ef08efbb4	hazermfgql	3000:3000	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	23 sec.	⋮
test_db2	a1bf27b7b391	mysql:8.0	-	0.76%	349.1MB / 7.68Gi	4.44%	913KB / 12.8MB	1.17KB / 126B	38	16 min.	⋮
react-app	-	-	-	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	16 min.	⋮
nginx_web	574569bccbae	nginx:alpine	8080:80	0%	0B / 0B	0%	0B / 0B	0B / 0B	0	16 min.	⋮

Terminal output:

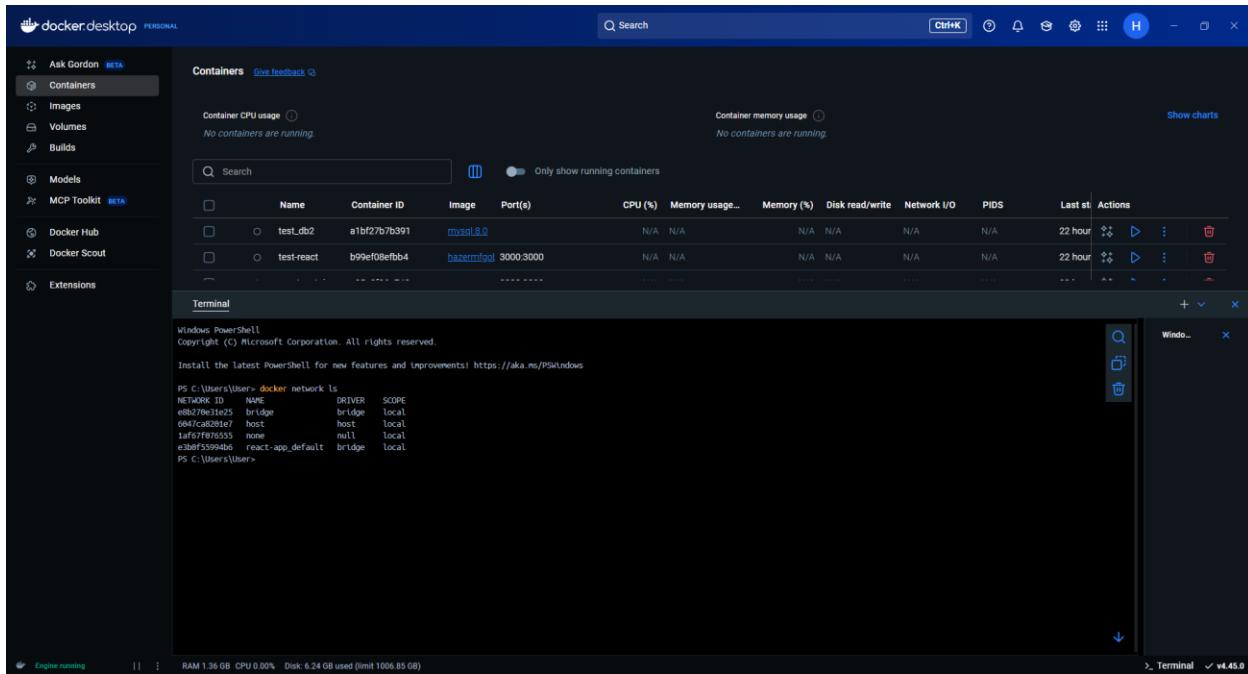
```
PS C:\Users\user\react-app> docker run -d -v ./test_db2:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=py-secret-pr -v mysql_data:/var/lib/mysql mysql:8.0
a1bf27b7b391@24ecf7c163ccb8a99e97a6833dc81c8f8ed51:86ff388
PS C:\Users\user\react-app> docker exec -it test_db2 mysql -uroot -p$py-secret-pr -e "SHOW DATABASES; USE testdb; SHOW TABLES;" 
mysql: [Warning] Using a password on the command line interface can be insecure.

+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| testdb |
+-----+
| Tables_in_testdb |
+-----+
| users |
+-----+
PS C:\Users\user\react-app>
```

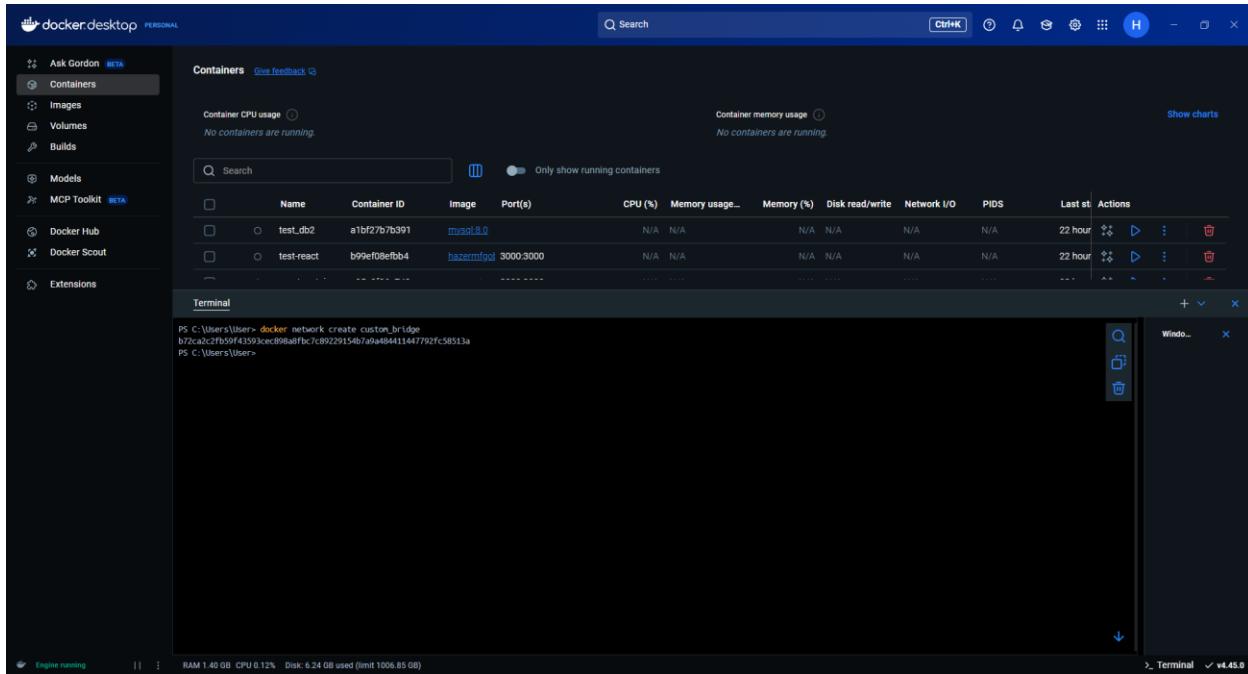
Engine running | RAM 1.84 GB CPU 0.50% Disk: 6.26 GB used (limit 1006.85 GB) | > Terminal v4.45.0

8. Docker Network

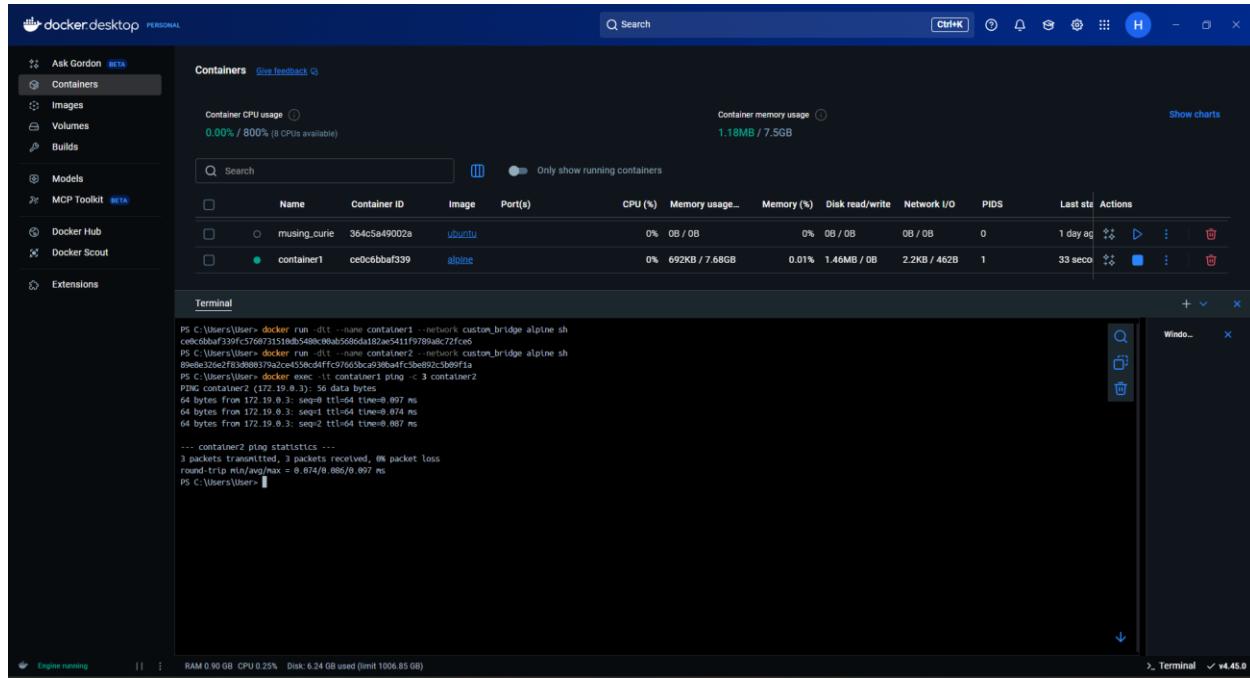
List networks with docker network ls



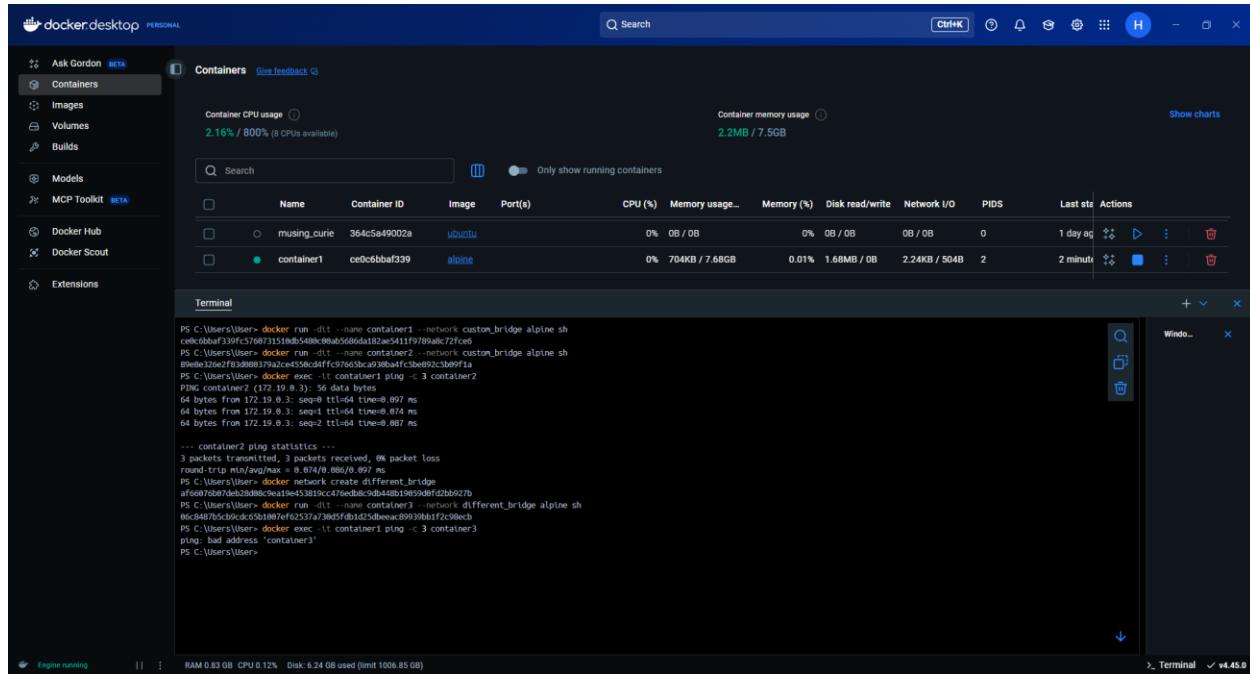
Create a custom bridge network



Run two containers in the same network and ping each other

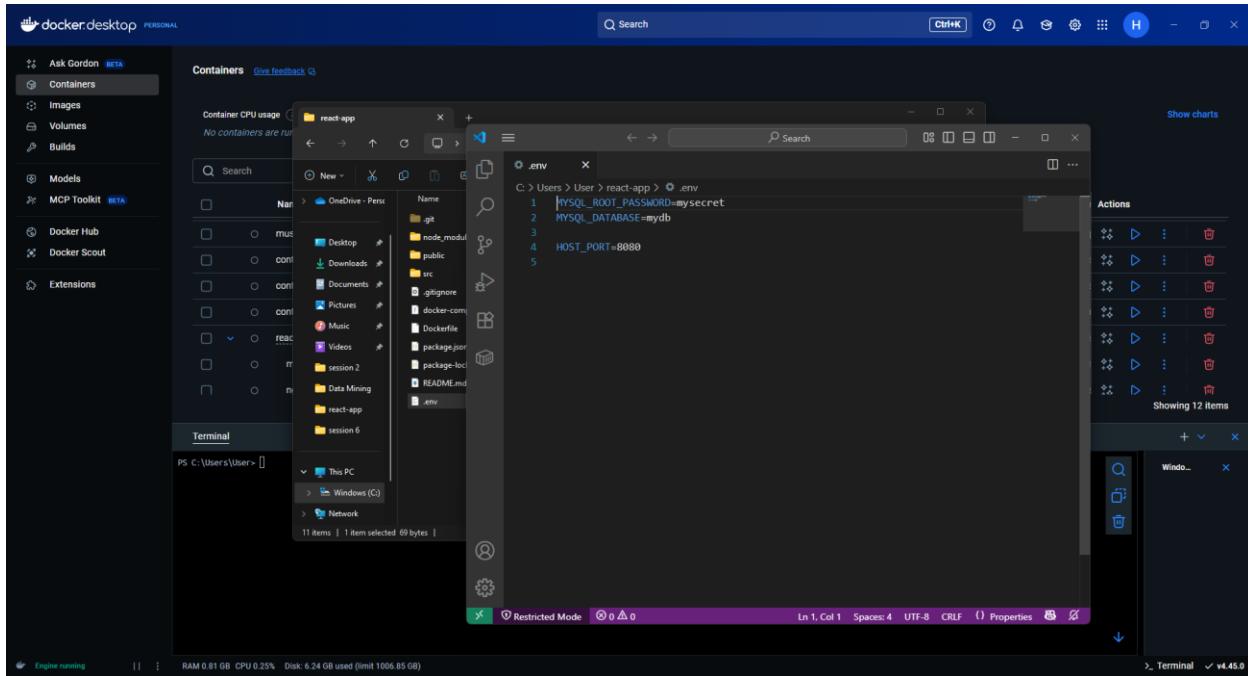


Run containers in different networks and test communication

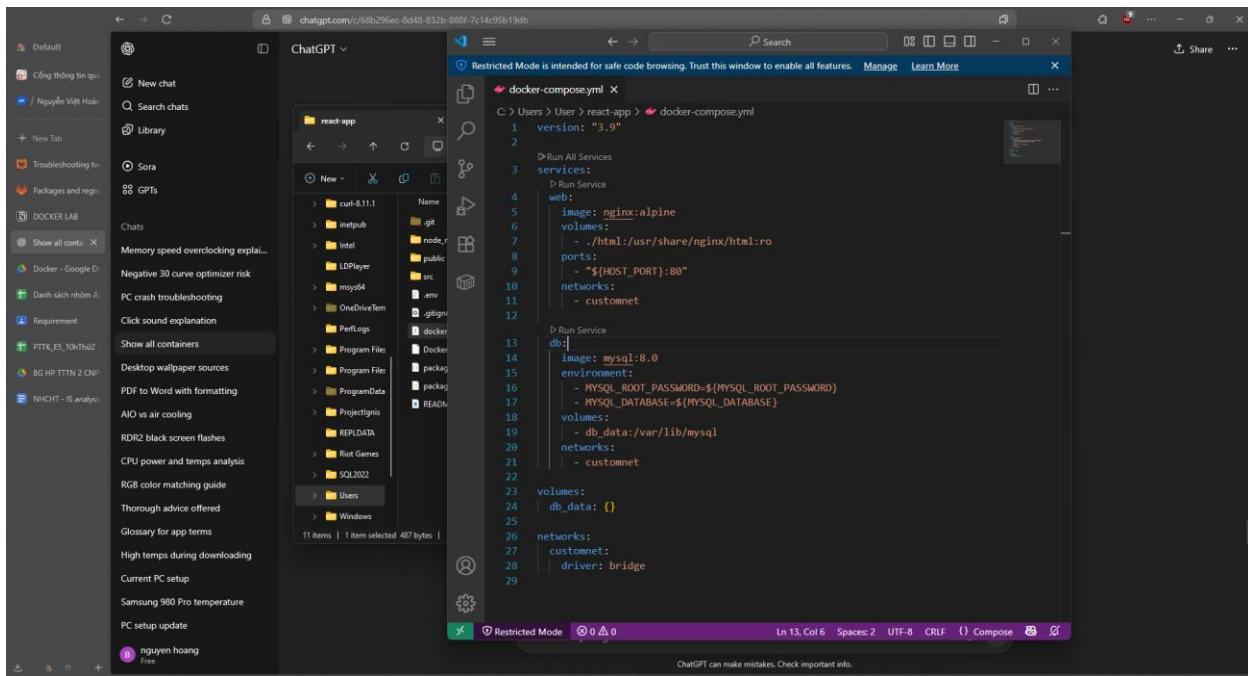


9. Docker Compose (Part 2 – Advanced)

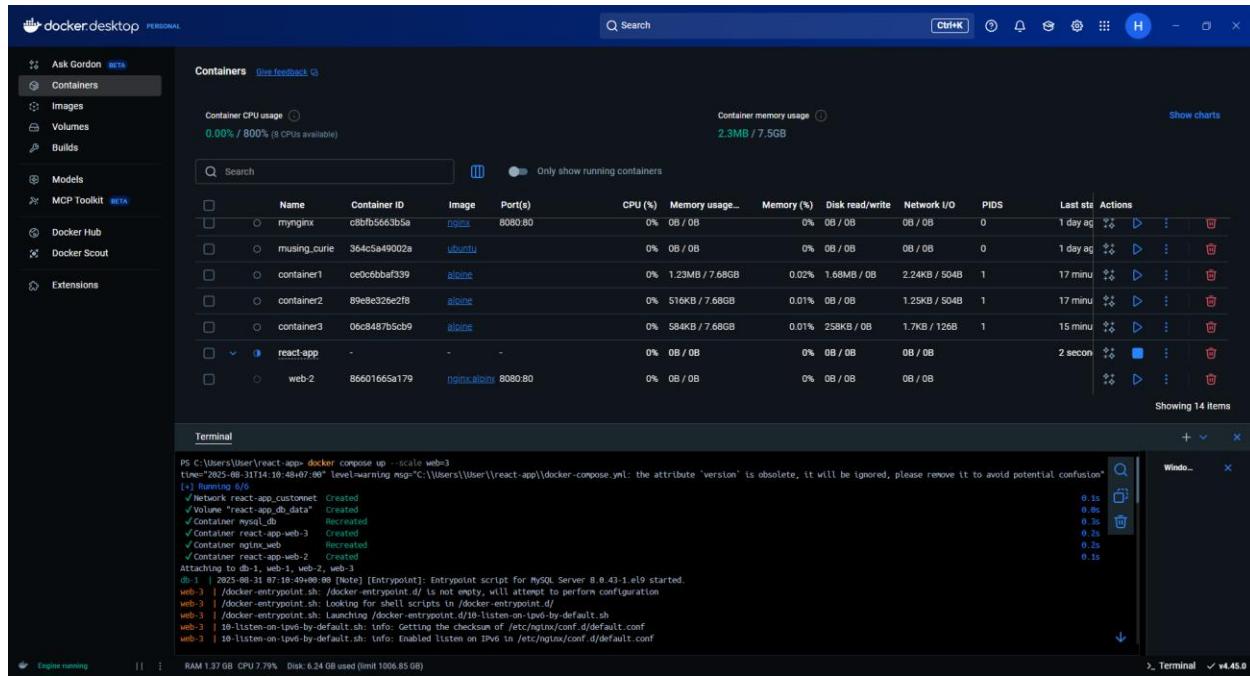
Add environment variables using a .env file



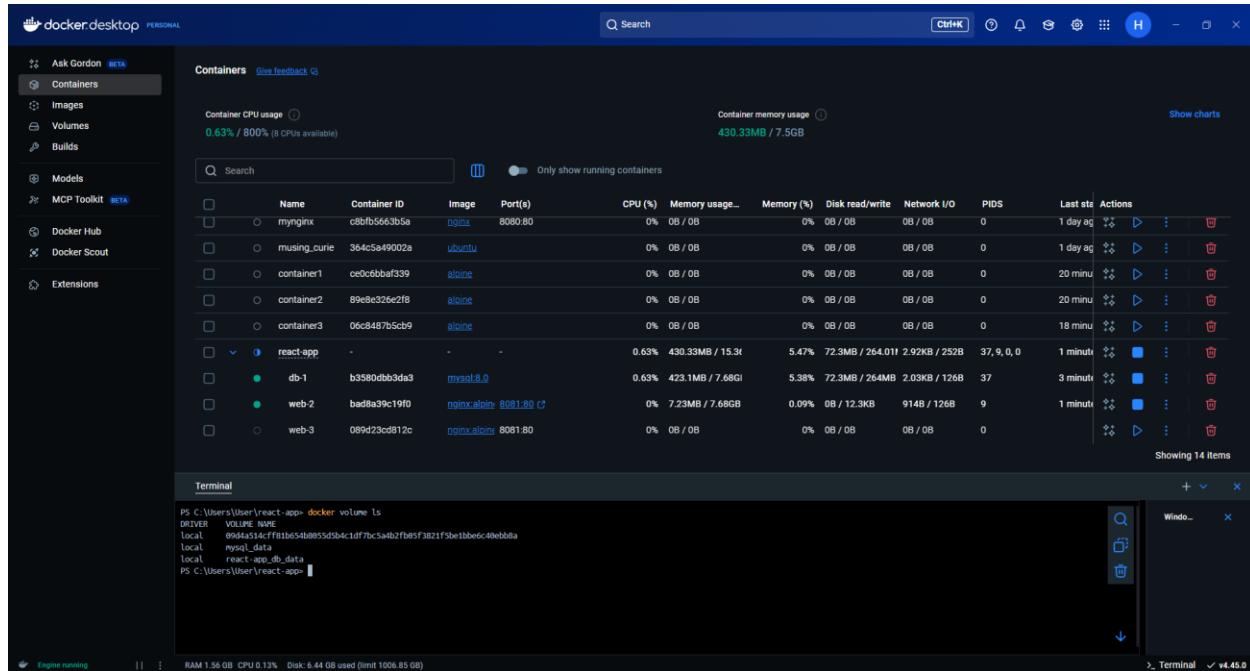
Add volume mappings for persistence and define custom networks in docker-compose.yml



Scale services with docker compose up --scale web=3



Check all the volume



10. Advanced – Build & Push with GitLab CI/CD

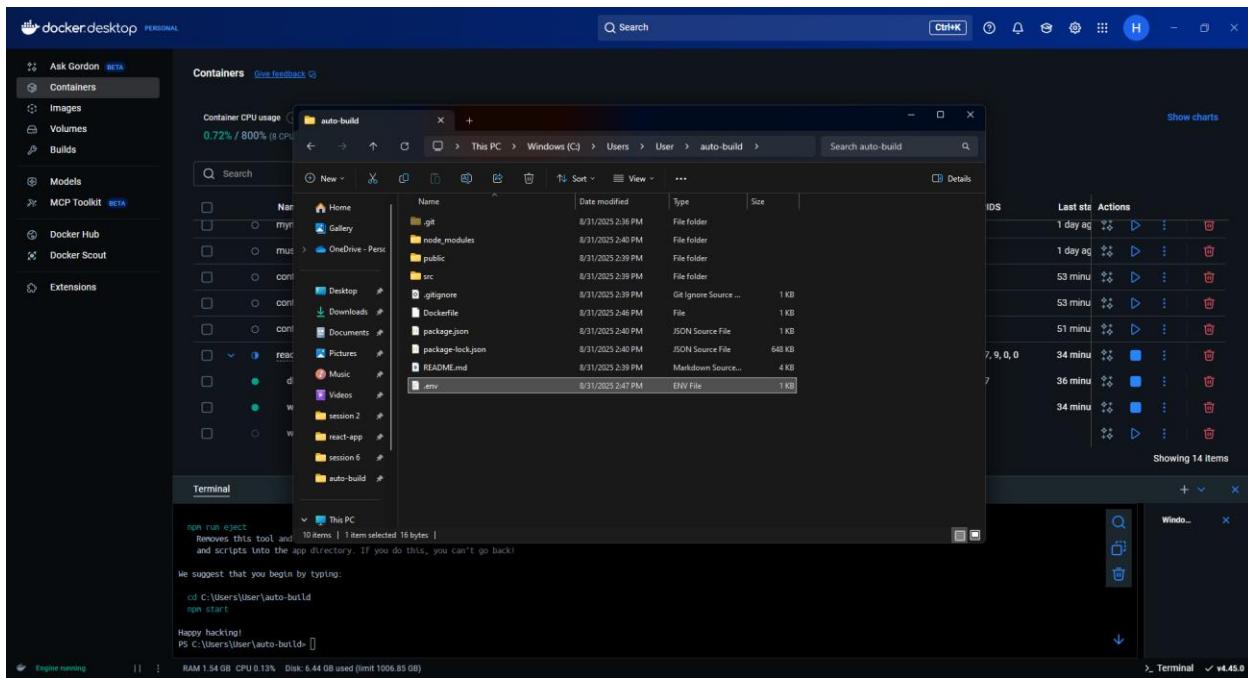
Create new GitLab repo

A screenshot of the Docker Desktop application interface. On the left, the sidebar shows 'Containers' selected. In the center, a 'Containers' tab is open, showing a list of containers including 'auto-build'. A terminal window is open at the bottom, showing a command-line session in 'Restricted Mode'. The terminal output includes:

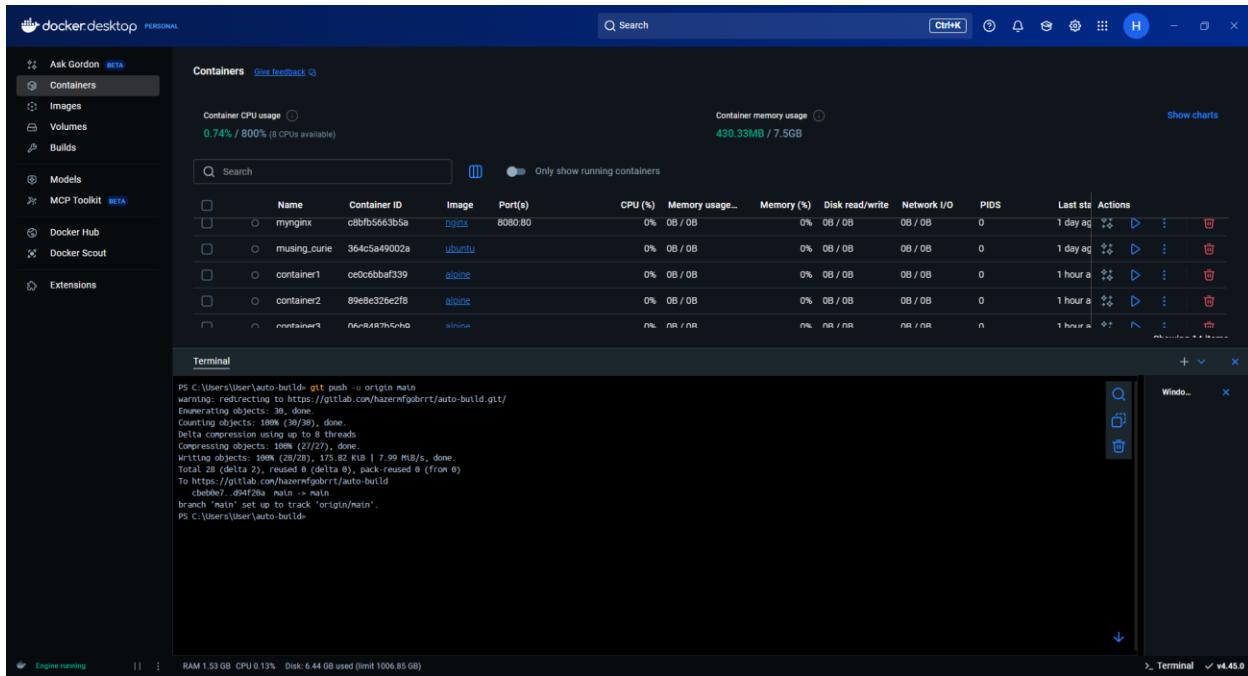
```
npm run eject
Removes this tool and
and scripts into the app directory. If you do this, you
We suggest that you begin by typing:
cd C:\Users\User\auto-build
npm start
Happy hacking!
PS C:\Users\User\auto-build>
```

The terminal also shows the current working directory as 'C:\Users\User\auto-build'. At the top right, there is a 'Restricted Mode' indicator.

Add a Dockerfile (from Step 4) to your repo.



Commit and push to GitLab



Check GitLab CI/CD pipeline

