

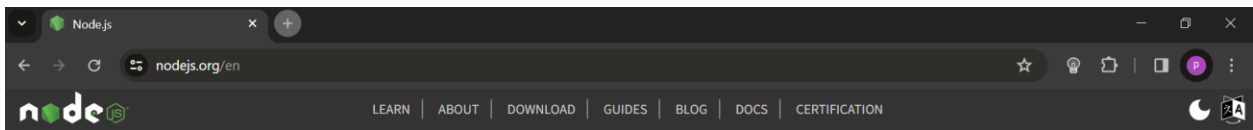
## คู่มือการใช้งาน React บน Server

### การติดตั้ง React Application

1. ติดตั้ง Node.js และ Visual studio code
2. ติดตั้ง React
3. วิธีการใช้ env file
4. การ Deploy Docker
5. การ Deploy Server

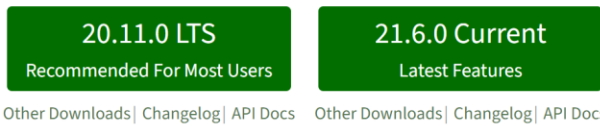
## 1. ติดตั้ง node JS และ VScode

- ดาวน์โหลดและติดตั้ง node.js <https://nodejs.org/en>

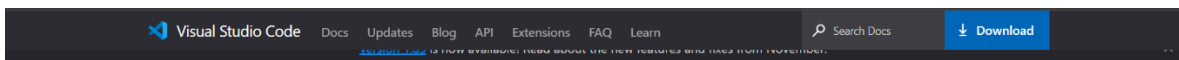


Node.js® is an open-source, cross-platform JavaScript runtime environment.

### Download Node.js®

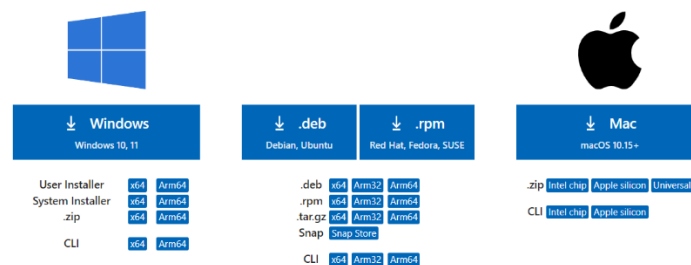


- ดาวน์โหลดและติดตั้ง VS Code <https://code.visualstudio.com/download>
- เลือก version ที่ต้องการใช้งาน



### Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



## 2. ติดตั้ง React

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\React_create_new> npm create vite@latest ← 2.1
✓ Project name: ... New_Project ← 2.2
✓ Package name: ... new-project
? Select a framework: » - Use arrow-keys. Return to submit.
  Vanilla
  Vue
>  React ← 2.3
  Preact
  Lit
  Svelte
  Solid
```

- ขั้นตอนการสร้างโปรเจกต์ด้วย React
- 2.1 พิมพ์คำสั่ง
- 2.2 ตั้งชื่อ Project name และ Package name
- 2.3 เลือก framework เป็น React

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

✓ Select a framework: » React
? Select a variant: » - Use arrow-keys. Return to submit.
  TypeScript
  TypeScript + SWC
  JavaScript
>  JavaScript + SWC ← 2.4
```

- 2.4 เลือก JavaScript + SWC

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

✓ Select a framework: » React
Scaffolding project in D:\React_create_new\New_Project...

Done. Now run:

cd New_Project      ← 2.5
npm install          ← 2.6
npm run dev          ← 2.7

PS D:\React_create_new> cd .\New_Project\
PS D:\React_create_new\New_Project> npm install
```

- 2.5 พิมพ์คำสั่งเพื่อไปยัง Folder ที่สร้างไว้
- 2.6 พิมพ์คำสั่งเพื่อติดตั้ง dependencies
- 2.7 พิมพ์คำสั่งเพื่อรันโปรแกรมขึ้น web

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

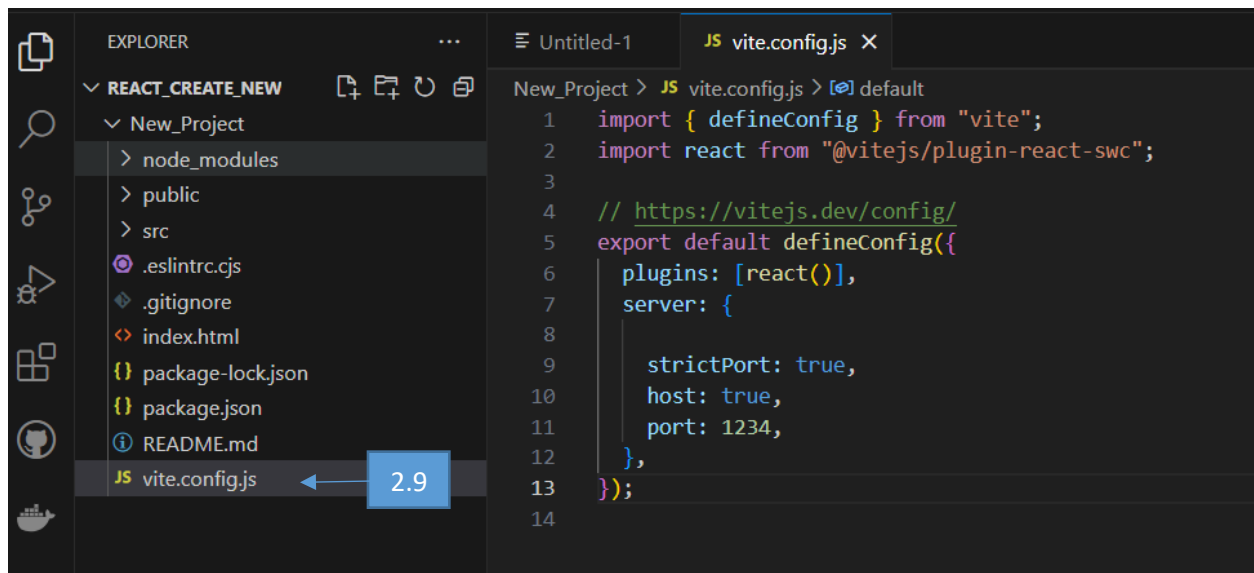
PS D:\React_create_new\New_Project> npm run dev

> new-project@0.0.0 dev
> vite

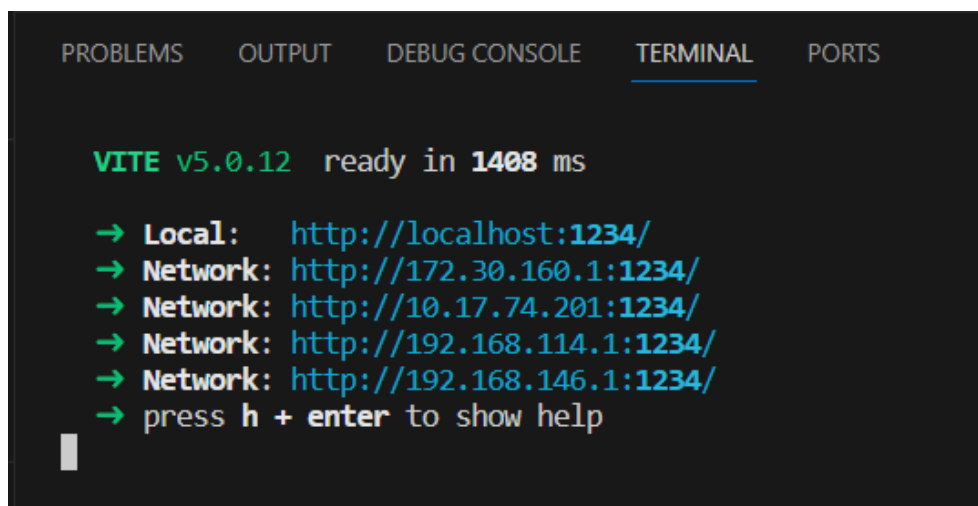
VITE v5.0.12 ready in 479 ms

→ Local:  http://localhost:5173/ ← 2.8
→ Network: use --host to expose
→ press h + enter to show help
```

- 2.8 คลิกไปยังลิ้งค์เพื่อเข้าไปหน้า web



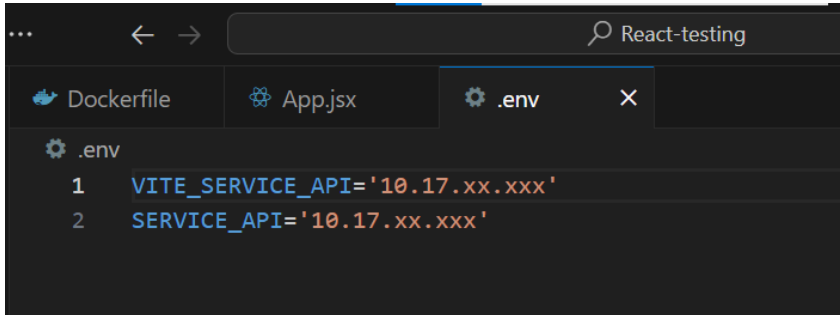
- 2.9 เข้าไปแก้ไข Code ใน vite.config.js ตามรูปด้านบน เพื่อเป็นการตั้งค่า port และ host แบบ public



- ทำการรัน npm run dev ใหม่อีกครั้ง ก็จะได้ผลลัพธ์ตามรูปด้านบน

### 3. ตั้งค่า .env ใน front-end และ back-end

#### 3.1 Front-end



- สร้างไฟล์ .env ขึ้นมา แล้วใส่ค่า IP ในส่วนของ Front-end

Note : หากใช้ vite ในการสร้าง react application จำเป็นต้องเพิ่ม VITE หน้าชื่อ เนื่องจากเป็นกฎของการใช้ Environment Variables

#### ตัวอย่าง

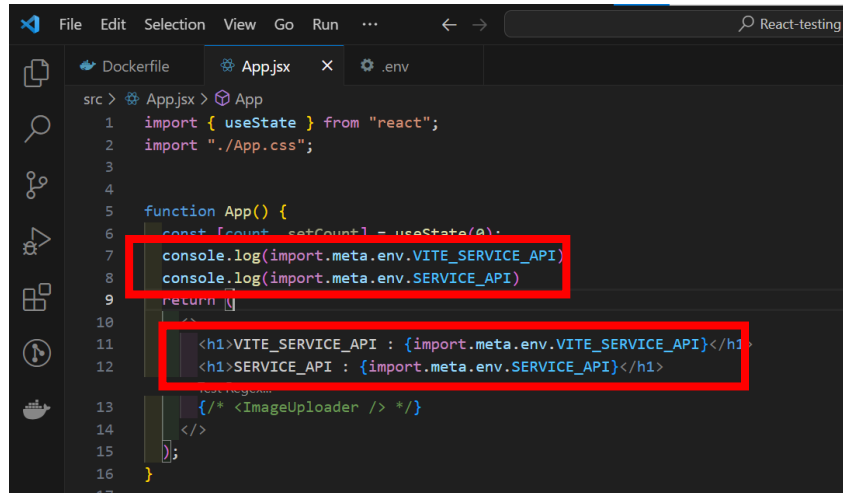
```
VITE_SERVICE_API='10.17.xx.xxx'
```

```
SERVICE_API='10.17.xx.xxx'
```

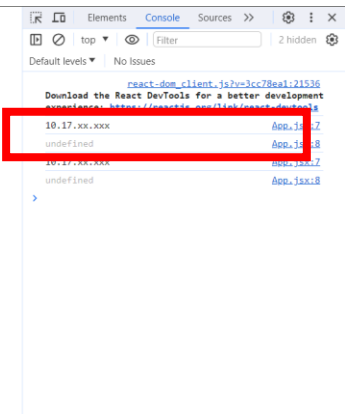
In .jsx file or front-end file

```
Console.log(import.meta.env.VITE_SERVICE_API) ผลลัพธ์ คือ 10.17.xx.xxx
```

```
Console.log(import.meta.env.SERVICE_API) // undefined
```



```
1 import { useState } from "react";
2 import "./App.css";
3
4
5 function App() {
6   const [count, setCount] = useState(0);
7   console.log(import.meta.env.VITE_SERVICE_API);
8   console.log(import.meta.env.SERVICE_API);
9   return (
10     <div>
11       <h1>VITE_SERVICE_API : {import.meta.env.VITE_SERVICE_API}</h1>
12       <h1>SERVICE_API : {import.meta.env.SERVICE_API}</h1>
13     </div>
14   );
15 }
16
```



**VITE\_SERVICE\_API : 10.17.xx.xxx**  
**SERVICE\_API :**

### 3.2 Back-end

- ในส่วนของ Back-end หากใช้ .env file จะมีความปลอดภัยและง่ายต่อการปรับปรุงแก้ไข จึงจำเป็นต้องเก็บข้อมูลเอาไว้ภายใน .env file

- การติดตั้ง dotenv ใน cmd ตัวอย่าง npm i dotenv หรือ npm install dotenv

- เราจะใช้ .env file เดียวกันกับ front-end แต่ชื่อต้องไม่ตรงกับ Variable ของ front-end

**Note :** ในส่วนของ back-end ไม่จำเป็นต้องกังวลเกี่ยวกับกฎของ VITE เราสามารถตั้งชื่ออะไรก็ได้ตามต้องการ

ตัวอย่าง

- การเชื่อมต่อ oracle database แล้วทำการสร้าง user password และ connection string ใน .env file

```

.env
1 VITE_SERVICE_API='10.17.xx.xxx'
2 SERVICE_API='10.17.xx.xxx'
3
4
5 #connect db_oracle
6
7 oracle_user= 'user1'      #user
8 oracle_pass= 'user1'     #password
9 oracle_conn= 'oracle_db' #connectionstring
10
11 |

```

- ในไฟล์ .js ทำการ import dotenv ด้วยคำสั่ง `require('dotenv').config()`

```

JS service.cjs > ...
1 require('dotenv').config()
2
3
4 const db_oracle_connect = {
5   user: process.env.oracle_user,
6   password: process.env.oracle_pass,
7   connectionString: process.env.oracle_conn,
8 };
9
10 console.log(db_oracle_connect)

```

- ตัวอย่างแสดงเมื่อเชื่อมต่อได้แล้ว ใน cmd

```

React-testing > node .\service.cjs
{ user: 'user1', password: 'user1', connectionString: 'oracle_db' }
Chayanon.I > React-testing

```

Note ถ้าไฟล์ api.js ไม่ได้อยู่ใน directory เดียวกันกับไฟล์ .env จำเป็นต้องกำหนดที่อยู่ไฟล์ .env ให้กับ api.js เพื่อให้สามารถค้นหาเจอได้ ตัวอย่าง

api.js file : root/src/service/api.js

.env file : root/.env

```

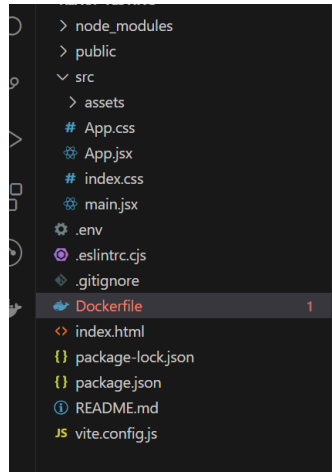
Dockerfile App.jsx .env JS service.cjs X # App.css
src > service > JS service.cjs > ...
1
2 // require('dotenv').config()
3 const path = require('path');
4 require('dotenv').config({ path: path.resolve(__dirname, '../.env') });
5
6 const db_oracle_connect = {
7   user: process.env.oracle_user,
8   password: process.env.oracle_pass,
9   connectionString: process.env.oracle_conn,
10 };
11
12 console.log(db_oracle_connect)

```



## 4. ขั้นตอนการ deployment โดยใช้ Docker

### 4.1 สร้างไฟล์ชื่อ Dockerfile ใน project folder



### 4.2 เพิ่ม promt เพื่อสร้าง docker images

```
Dockerfile X
Dockerfile > ...
1 FROM node:20
2
3 WORKDIR /app
4
5 RUN apt-get update && apt-get install -y --no-install-recommends nano vim libaio1
6
7 COPY package*.json ./
8
9 RUN npm install
10
11 COPY . .
12
13 CMD ["npm", "run", "dev"]
14
```

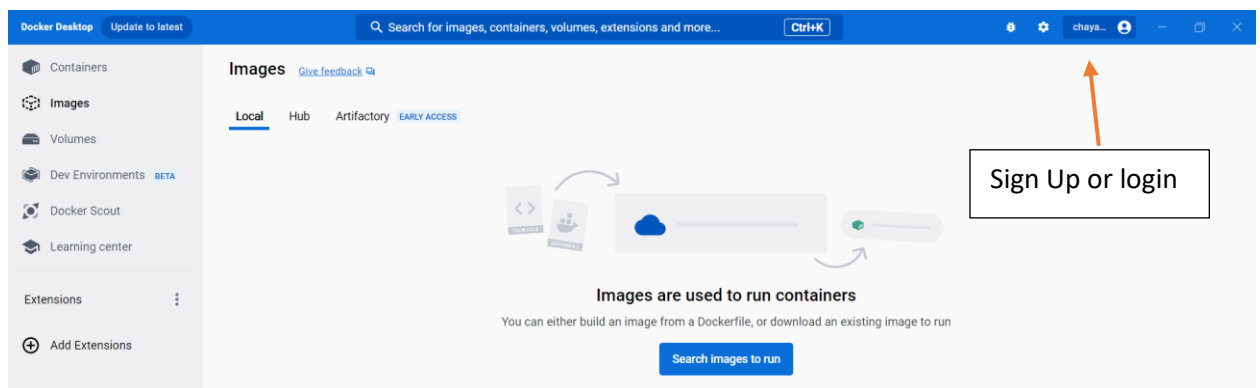
- บรรทัดที่ 1 พื้นฐานของ image จะต้องใช้ node version ตามที่ dependency ต้องการ
- บรรทัดที่ 3 กำหนดให้สร้าง directory ชื่อ app ที่จะเป็น working directory สำหรับสิ่งที่จะทำทั้งหมดใน Dockerfile ต่อจากนี้
- บรรทัดที่ 5 ติดตั้งและอัปเดตแพ็คเกจ ในระบบการทำงานของ container
- บรรทัดที่ 7 copy file package.json และ package-lock.json จาก project folder ไปใช้งานใน Container
- บรรทัดที่ 9 ติดตั้ง dependencies ในโปรเจกต์นี้
- บรรทัดที่ 11 คัดลอกไฟล์ทั้งหมดในโปรเจกต์ไปเก็บไว้ใน directory ที่สร้างขึ้น

- บรรทัดที่ 13 กำหนด prompt ในการทำงานบน container ซึ่งใน Project นี้ใช้ `npm run dev`

#### 4.3 การ deploy ไปยัง Docker

Note : ในที่นี่ใช้โปรแกรม Docker desktop เพื่อ Deploy บน Local Computer ก่อน จากนั้นจึงจะ push ไปยัง Docker hub แล้ว pull images และ build บน Linux server ในภายหลัง เพราะ internet ของ linux นั้นช้าและไม่สามารถดาวน์โหลด library บางตัวใน Docker ได้ ถ้าถึงเวลาที่ต้องการใช้งาน กรุณาดูข้อกำหนดของเซิร์ฟเวอร์ Linux และเลือกวิธีการใช้งานตามที่ต้องการ ซึ่งจะสามารถ deploy บน Linux server โดยตรง หรือ deploy บน Docker desktop ก่อนก็ได้

- เปิด Docker Desktop เพื่อ login เข้าไปใช้งาน



- เปิด Terminal หรือ CMD Administrator และ Check version แล้วตรวจสอบการติดตั้งของ Docker

```
PS C:\Users\Paiboon.Wo> docker -v
Docker version 24.0.7, build afdd53b
PS C:\Users\Paiboon.Wo> |
```

- ไปที่ react project directory

Note : ใน project directory จะต้องมีย Dockerfile สำหรับสร้าง images

```
PS D:\> cd .\00_REACT_WORK\05_TEST_DEPLOY_PON\Deploy_Test\|
```

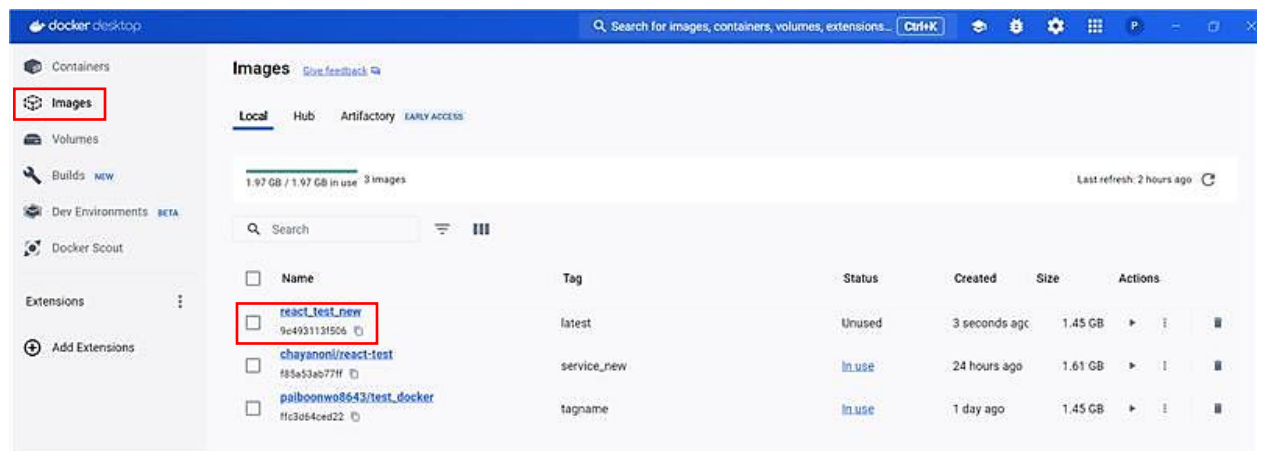
## การสร้าง images

Command : `docker build -t <images_names> .` (images\_names ตั้งชื่ออะไรก็ได้และต้องเป็นตัวพิมพ์เล็กเท่านั้น)

```
Paiboon.Wo on D:/React_create_new/New_Project
# docker build -t react_test_new .
[+] Building 441.3s (12/12) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 236B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/node:20
=> [1/6] FROM docker.io/library/node:20@sha256:ffebb4405810c92d267a764b21975fb2d96772e41877248a37bf3abaa0d3b590
=> => resolve docker.io/library/node:20@sha256:ffebb4405810c92d267a764b21975fb2d96772e41877248a37bf3abaa0d3b590
=> => sha256:1b13d4e1a46e5e969702ec92b7c787c1b6891b7f7c21ad378ff6dbc9e751d5d4 49.56MB / 49.56MB
docker:default 0.1s
0.0s
0.1s
0.0s
4.6s
361.7s
0.0s
137.8s
```

ถ้าสร้าง docker images เสร็จแล้ว แล้วตรวจสอบใน cmd ด้วยคำสั่ง `docker images` หรือ เปิด docker desktop

```
view a summary of image vulnerabilities and recommendations / docker scout quickvi
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-[#)
>> docker images
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
react_test_new      latest      9c493113f506  2 minutes ago  1.46GB
chayanoni/react-test service_new  f85a53ab77ff  24 hours ago  1.62GB
paiboonwo8643/test_docker tagname     ffc3d64ced22  24 hours ago  1.45GB
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-[#)
>> |
```



## ทดสอบ run container ในเครื่องคอม

- ทำการพิมพ์คำสั่งเพื่อสร้าง container ใน cmd

`docker run --name=<container_name> -p <public_port>:<self_port>-d <docker_images_id>`

```
>> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
react_test_new      latest             9c493113f506       2 minutes ago      1.46GB
chayanoni/react-test service_new        f85a53ab77ff       24 hours ago       1.62GB
paiboonwo8643/test_docker tagname           ffc3d64ced22       24 hours ago       1.45GB
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
>> docker run --name=name_test -p 1234:1234 -d 9c493113f506
de110d594469b2c1aae4d37139b67165054366869b01c5fbb4c01d01cf6fff77
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
```

- ถ้าต้องการตรวจสอบสถานะการ deploy docker container สามารถพิมพ์คำสั่ง `docker ps` ใน cmd

```
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
>> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
de110d594469  9c493113f506  "docker-entrypoint.s..." 15 seconds ago Up 13 seconds 0.0.0.0:1234->1234/tcp  name_test
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
>> |
```

- ถ้าต้องการดู ip ที่เครื่องเรา สามารถดูได้ที่ cmd โดยพิมพ์คำสั่ง `ipconfig`

```
[Paiboon.Wo@TN2N191001]-[~]-(#)
>> ipconfig

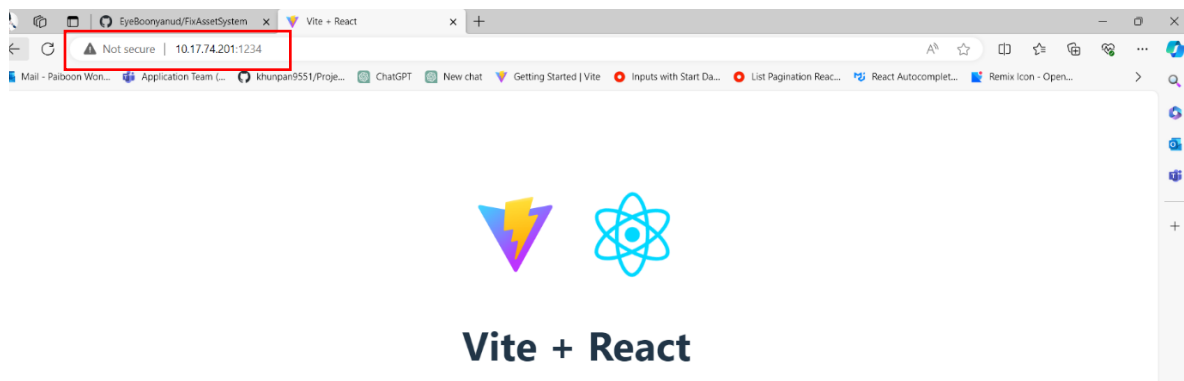
Windows IP Configuration

Ethernet adapter vEthernet (WSL):

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::c1cf:b37c:e812:3239%55
    IPv4 Address. . . . . : 172.30.160.1
    Subnet Mask . . . . . : 255.255.240.0
    Default Gateway . . . . . : 

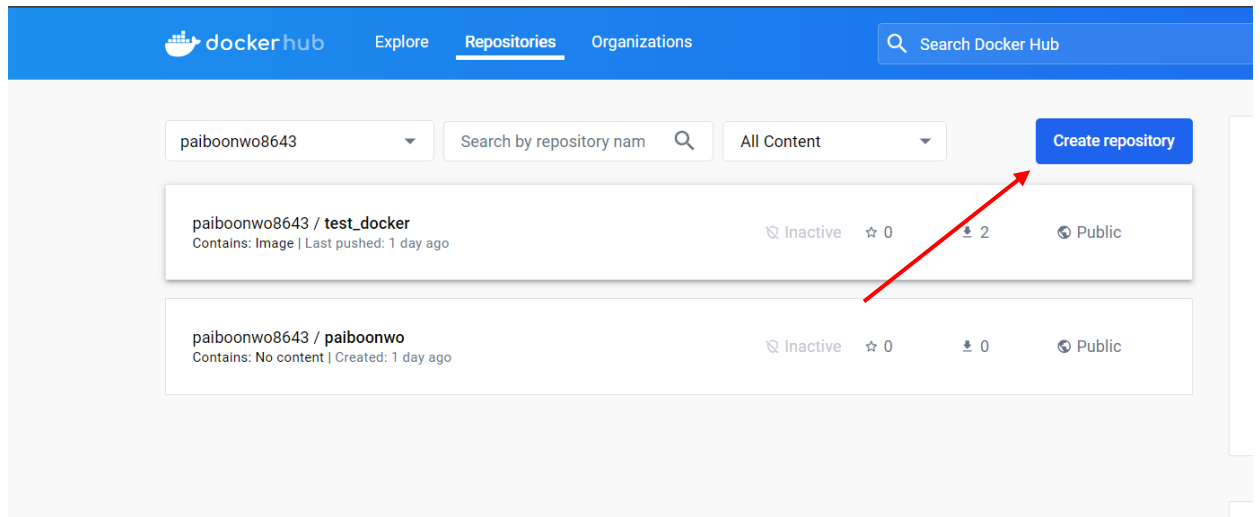
Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::1957:618b:195c:e8ce%20
    IPv4 Address. . . . . : 10.17.74.201
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 10.17.74.1
```

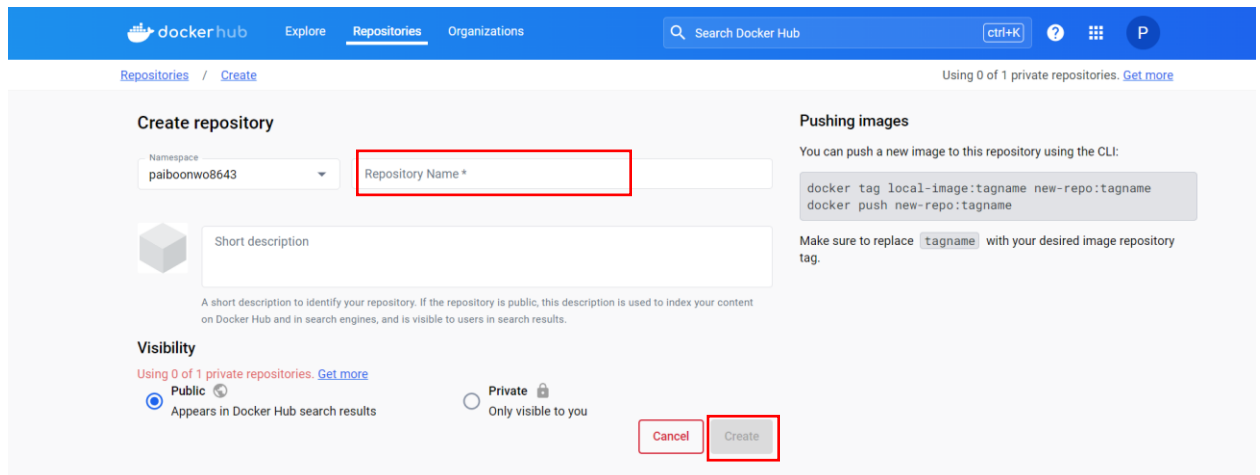


## Push ไปยัง Docker hub

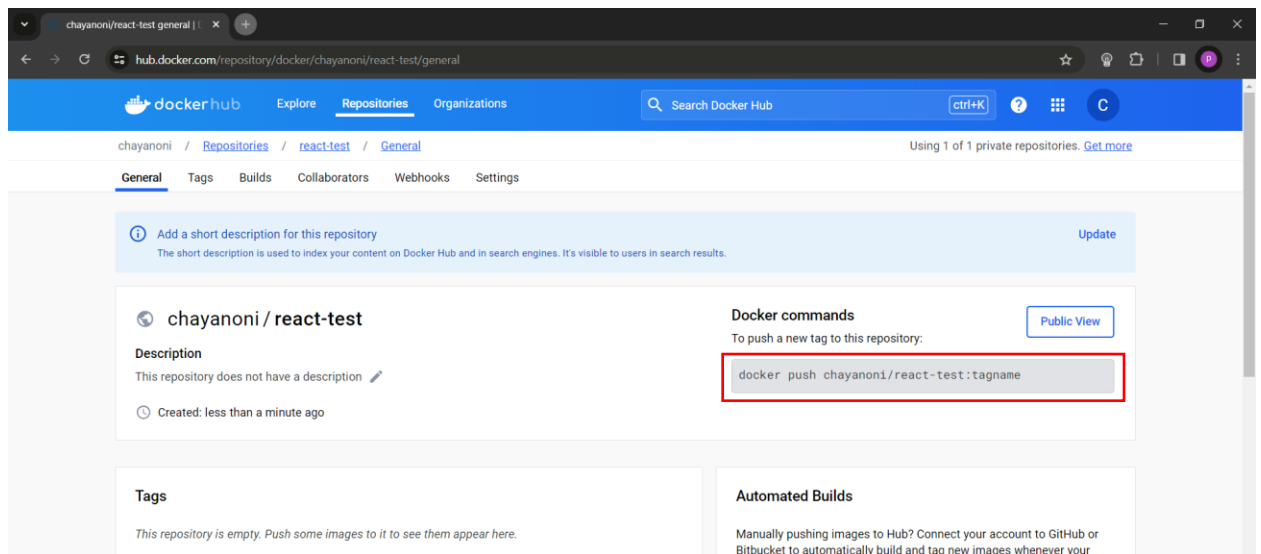
- ไปที่ <https://hub.docker.com> แล้ว sign up หรือ login
- สร้าง repository



- กำหนดชื่อ repository name แล้วคลิก create เพื่อสร้าง



- สามารถ push images จากเครื่องคอมไปยัง docker hub ได้แล้ว โดยนำ path ที่ได้มาเอาไปใช้



- ไปที่ cmd เพื่อสร้าง tag ใน docker images

พิมพ์คำสั่ง `docker tag <image_id> <username_dockerhub>/<name_repository>:tagname`

```
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
>> docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
react_test_new      latest       9c493113f506     About an hour ago 1.46GB
chayanoni/react-test service_new  f85a53ab77ff     25 hours ago     1.62GB
paiboonwo8643/test_docker tagname      ffc3d64ced22     25 hours ago     1.45GB
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
>> docker tag 9c493113f506 paiboonwo8643/tag_test:tagname
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
```

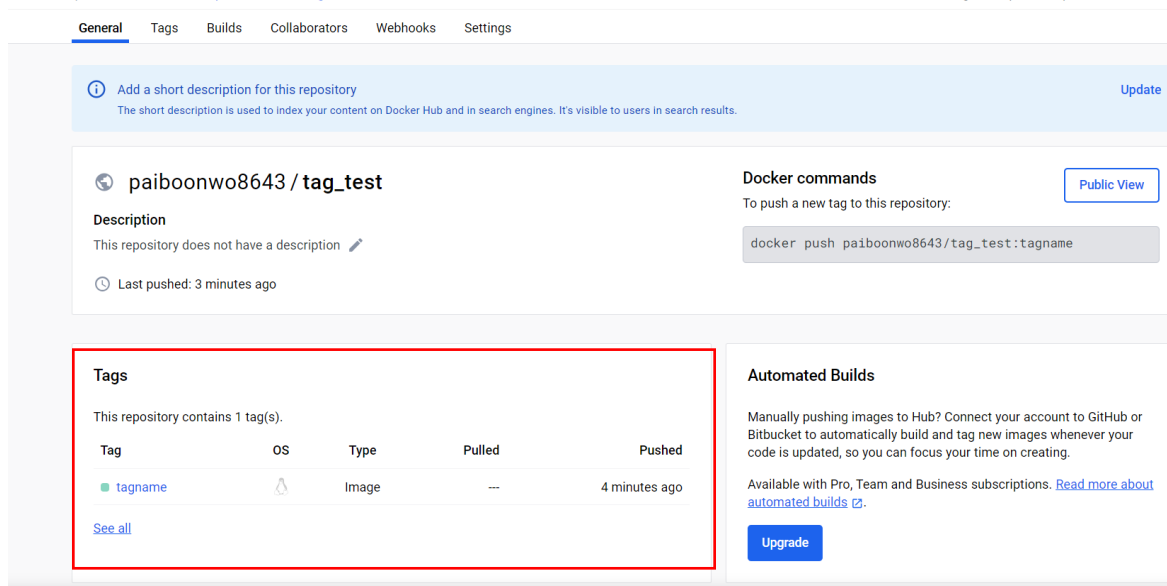
- เมื่อสร้าง tag images เสร็จแล้ว สามารถตรวจสอบได้โดยพิมพ์คำสั่ง Docker images ใน cmd

```
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
>> docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
paiboonwo8643/tag_test tagname      9c493113f506     About an hour ago 1.46GB
react_test_new      latest       9c493113f506     About an hour ago 1.46GB
chayanoni/react-test service_new  f85a53ab77ff     25 hours ago     1.62GB
paiboonwo8643/test_docker tagname      ffc3d64ced22     25 hours ago     1.45GB
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
```

- สามารถ push ไปยัง Docker hub ได้แล้ว ดังรูป

```
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)  
>> docker push paiboonwo8643/tag_test:tagname  
The push refers to repository [docker.io/paiboonwo8643/tag_test]  
ceb14acddadb: Pushed  
acc1dbec64a0: Pushed  
5871aebb8b7a: Pushed  
8721dadff1dea: Pushed  
03ead86f7976: Pushed  
272e3c178eda: Mounted from paiboonwo8643/test_docker  
ee2bb7d5222a: Mounted from paiboonwo8643/test_docker  
82215cd0dbfd: Mounted from paiboonwo8643/test_docker  
c13263a86bbe: Mounted from paiboonwo8643/test_docker  
a876dfc51caa: Mounted from paiboonwo8643/test_docker  
5bb1de08f5af: Mounted from paiboonwo8643/test_docker  
0dfa23fffa41: Mounted from paiboonwo8643/test_docker  
aa904f36746c: Mounted from paiboonwo8643/test_docker  
tagname: digest: sha256:a3aa0c1a9e3a2e94a17ee6a69752043b53db20469d5eec5e55a8064908b86c69 size: 3055  
[Paiboon.Wo@TN2N191001]-[D:\..\New_Project]-(#)
```

- เมื่อเสร็จแล้วทำการตรวจสอบ tag ใน Docker hub



The screenshot shows the Docker Hub interface for the repository `paiboonwo8643/tag_test`. The page includes a navigation bar with tabs: General, Tags, Builds, Collaborators, Webhooks, and Settings. A blue banner at the top prompts to add a short description for the repository. The repository name is `paiboonwo8643/tag_test`, and it currently has no description. The 'Last pushed' time is 3 minutes ago. The 'Tags' section is highlighted with a red box and shows a table with one tag: `tagname`, which is an Image type, pushed 4 minutes ago. The 'Automated Builds' section on the right explains how to connect GitHub or Bitbucket for automatic builds and includes an 'Upgrade' button.

Tag	OS	Type	Pulled	Pushed
tagname		Image	---	4 minutes ago

## ขั้นตอนการแก้ไขไฟล์

- พิมพ์คำสั่ง `docker ps` เพื่อตรวจสอบ Container ID ที่กำลังทำงานอยู่ แล้วพิมพ์คำสั่ง

`docker exec -it <Container_ID> /bin/bash`

```
Paiboon.Wo on D:/React_create_new/New_Project
# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                    NAMES
c17a1f630ec3   5fc57d90058f                       "docker-entrypoint.s..." 45 minutes ago Up 3 minutes    0.0.0.0:1234->9090/tcp    funny_curie
Paiboon.Wo on D:/React_create_new/New_Project
# docker exec -it c17a1f630ec3 /bin/bash
root@c17a1f630ec3:/app# ls
Dockerfile  README.md  index.html  node_modules  package-lock.json  package.json  public  src  vite.config.js
root@c17a1f630ec3:/app# cd
```

- จากนั้นพิมพ์คำสั่ง `ls` เพื่อดูข้อมูลสำหรับแก้ไข ทำการ `cd ../` เข้าไป แล้วใช้คำสั่ง `nano <file_name>`

```
root@c17a1f630ec3:/app# ls
Dockerfile  README.md  index.html  node_modules  package-lock.json  package.json  public  src  vite.config.js
root@c17a1f630ec3:/app# cd src/
root@c17a1f630ec3:/app/src# ls
App.css  App.jsx  assets  index.css  main.jsx
root@c17a1f630ec3:/app/src# nano App.jsx
```

- ตัวอย่างไฟล์ที่จะแก้ไข

```
GNU nano 7.2 App.jsx
import { useState } from 'react'
import reactLogo from './assets/react.svg'
import viteLogo from '/vite.svg'
import './App.css'

function App() {
  const [count, setCount] = useState(0)

  return (
    <div>
      <a href="https://vitejs.dev" target="_blank">
        <img src={viteLogo} className="logo" alt="Vite logo" />
      </a>
      <a href="https://react.dev" target="_blank">
        <img src={reactLogo} className="logo react" alt="React logo" />
      </a>
    </div>
    <h1>Vite + React</h1>
    <div className="card">
      <button onClick={() => setCount((count) => count + 1)}>
        count is {count}
      </button>
      <p>
        Edit <code>src/App.jsx</code> and save to test HMR
      </p>
    </div>
    <p className="read-the-docs">
      Click on the Vite and React logos to learn more
    </p>
  </div>
)
export default App
```

[ Read 35 lines



- เมื่อแก้ไขเสร็จแล้ว ให้ออกจาก container ด้วยคำสั่ง exit จากนั้นให้ทำการ restart container

```

Paiboon.Wo on D:/React_create_new/New_Project
# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                   NAMES
c17a1f630ec3   5fc57d90058f   "docker-entrypoint.s..." 52 minutes ago Up 9 minutes   0.0.0.0:1234->9090/tcp   funny_curie
Paiboon.Wo on D:/React_create_new/New_Project
# docker restart c17a1f630ec3
c17a1f630ec3
Paiboon.Wo on D:/React_create_new/New_Project
# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                   NAMES
c17a1f630ec3   5fc57d90058f   "docker-entrypoint.s..." 52 minutes ago Up 6 seconds    0.0.0.0:1234->9090/tcp   funny_curie
Paiboon.Wo on D:/React_create_new/New_Project
#

```

## ขั้นตอนการลบ Container และ images

- ถ้าต้องการลบ container จะใช้คำสั่ง `docker rm < Container_id>` (จำเป็นต้องหยุดการทำงานของ container ที่ต้องการก่อน ถ้าหาก container กำลังทำงานอยู่ จะไม่สามารถลบได้)

```

Paiboon.Wo on D:/React_create_new/New_Project
# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                   NAMES
99205ae0a09b   5fc57d90058f   "docker-entrypoint.s..." About an hour ago Created                                hardcore_mayer
Paiboon.Wo on D:/React_create_new/New_Project
# docker rm 99205ae0a09b
99205ae0a09b
Paiboon.Wo on D:/React_create_new/New_Project
# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                   NAMES
Paiboon.Wo on D:/React_create_new/New_Project
#

```

- ถ้าต้องการลบ images จะใช้คำสั่ง `docker rmi < images_id>` (หาก images ที่ต้องการจะลบมี container ที่กำลังทำงานอยู่ภายใต้ images นั้น จะไม่สามารถลบได้จำเป็นต้องหยุดการทำงานของ container นั้นก่อน )

```

Paiboon.Wo on D:/React_create_new/New_Project
# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
react_test_new latest    5fc57d90058f   2 hours ago   1.46GB
Paiboon.Wo on D:/React_create_new/New_Project
# docker rmi 5fc57d90058f
Untagged: react_test_new:latest
Deleted: sha256:5fc57d90058f398f5211e793c84af19ea332d9651722306469aa866e7c8b29b7
Paiboon.Wo on D:/React_create_new/New_Project
# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
Paiboon.Wo on D:/React_create_new/New_Project
#

```

- ถ้าต้องการลบข้อมูลทั้ง images และ Container ที่ไม่ได้ทำงานอยู่นั้นจะใช้คำสั่ง docker system prune -a

```
Paiboon.Wo on D:/React_create_new/New_Project
# docker system prune -a
WARNING! This will remove:
- all stopped containers
- all networks not used by at least one container
- all images without at least one container associated to them
- all build cache

Are you sure you want to continue? [y/N] y
Deleted build cache objects:
lq95usg0al3iym3oo1cwxdhvd
kvqnipejddqel1zkx9pnl09swf
vsletw2iyayqquxal33cna0ct
ienaprhhqqnmjrwmsabbx7h1ic
qf3twp4wtkzffscqn4cgzj219
exjxhdswwc66mhmg326d7az1yr
pmqj9y8q6i16why9f7wp5pni
uenif2nu3zk6ne0dyhpppvuu3
tbuo5ctqpakhciok3bvj6mv2t
cvv3m9fe90rubq896h1wotsmq
n8x0rl1c5ac96x3ile1ufa9ju
bd2mkmr686gugmx4nhkeqebcm
khzon5qqzytuqtfzy8mz8uwwx
92jw711khuogx8cwxnxw4hw1e
lancq6darb3fxx3ntf1hh8qyg
nnb5ezjwwfrihsng3b0p0kzs2

Total reclaimed space: 464.1MB
Paiboon.Wo on D:/React_create_new/New_Project
# |
```

## 5. ขั้นตอนการ deploy application โดยใช้ Docker บน Linux

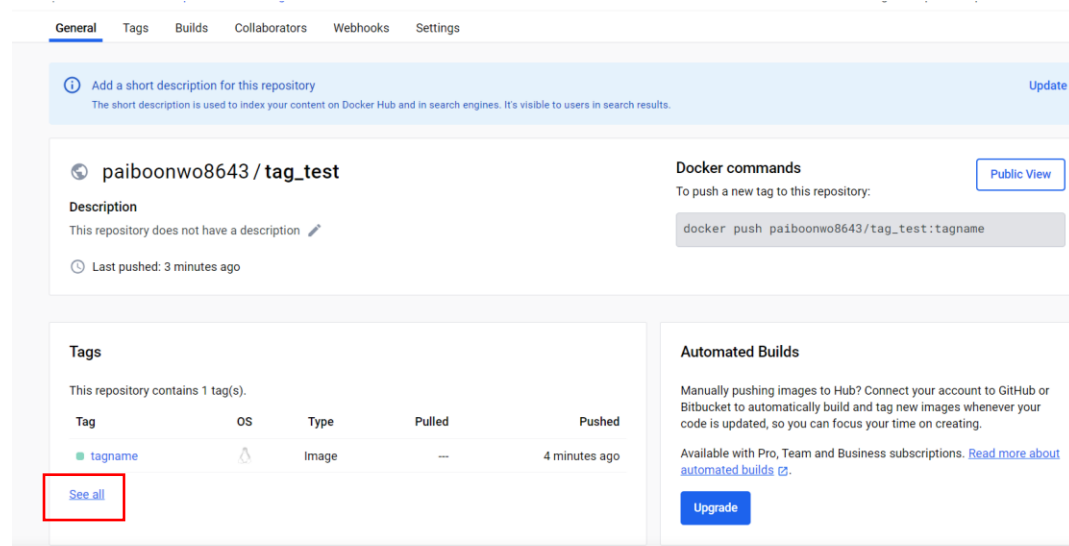
### 5.1 ทำการ remote เข้าไปยัง Linux server และเข้าระบบด้วย user ของผู้ใช้งาน

Note : ใช้คำสั่ง `sudo su` เพื่อให้สถานะเป็น root user

### 5.2 ตรวจสอบการติดตั้ง Docker

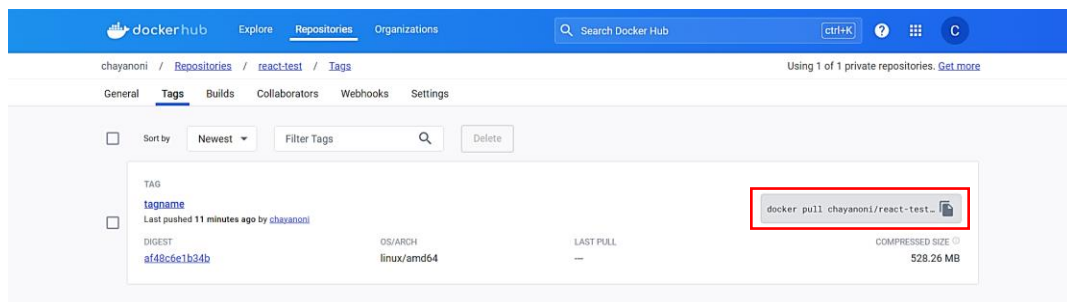
```
root@talslpgsql120:/# docker -v
Docker version 24.0.7, build afdd53b
root@talslpgsql120:/#
```

### 5.3 สามารถ pull images จาก Docker hub ได้แล้ว จากนั้นไปที่ Docker hub repository



### 5.4 สามารถ copy ข้อความตามรูปข้างล่าง เพื่อ pull docker images

Note : วิธีที่ง่ายที่สุดในการ pull images คือ การตั้งค่าพื้นที่จัดเก็บข้อมูลเป็น public เมื่อ pull images แล้วไม่พบปัญหาใดๆก็สามารถเปลี่ยนเป็น private ได้ในภายหลัง



## 5.5 สามารถนำมามาใน Linux server ได้เลย ดังรูป

```
root@ta1slpgsql120: /# docker pull chayanoni/react-test:tagname
tagname: Pulling from chayanoni/react-test
1b13d4e1a46e: Already exists
1c74526957fc: Already exists
30d855997954: Already exists
ad5739181616: Already exists
b2aed33d4664: Already exists
4b0f411a48b2: Already exists
6b8a05f6378b: Already exists
cddf686f5e25: Already exists
cfff2a464b722: Pull complete
5a5a9038643c: Pull complete
3137db98c242: Pull complete
71a7ac517a66: Downloading [=====] 36.99MB/94.73MB
de1d97d68a99: Downloading [=====] 2.47MB/29.51MB
```

## 5.6 ถ้า pull images เสร็จแล้ว สามารถตรวจสอบได้โดยพิมพ์คำสั่ง docker images

```
root@ta1slpgsql120: /# docker images
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
chayanoni/react-test tagname      79f342c9d67f 49 minutes ago 1.46GB
chayanoni/secure_test tagname      dc610aa15a14 3 days ago    1.17GB
chayanoni/service    tagname      952770868a89 4 days ago    1.61GB
chayanoni/my-forntend tagname      c4a955f69dfa 4 days ago    2.58GB
root@ta1slpgsql120: /#
```

## 5.7 จากนั้นพิมพ์คำสั่งรันเดียวกันกับการรันบน local computer

docker run --name=<container\_id> -p <public\_port>:<self\_port> -d <docker\_images>

```
root@ta1slpgsql120: /# docker images
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
chayanoni/react-test tagname      79f342c9d67f 51 minutes ago 1.46GB
chayanoni/secure_test tagname      dc610aa15a14 3 days ago    1.17GB
chayanoni/service    tagname      952770868a89 4 days ago    1.61GB
chayanoni/my-forntend tagname      c4a955f69dfa 4 days ago    2.58GB
root@ta1slpgsql120: /# docker run --name=react-test -p 8080:8080 -d 79f342c9d67f
2af557c4785e1449cdfc3adce63ebfbf3ec854e8e1d93028b91a2adde2ead82b
root@ta1slpgsql120: /#
```

## 5.8 จากนั้นตรวจสอบ docker container โดยพิมพ์คำสั่ง docker ps

```
root@ta1slpgsql120: /# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
2af557c4785e   79f342c9d67f   "docker-entrypoint.s..." 2 minutes ago Up 2 minutes   0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   react-test
```

Note : หาก Run Docker Container เสร็จแล้วและต้องการกำหนดค่า memory สามารถใช้คำสั่ง ดังนี้

`docker update --memory <2GiB> --memory-swap <2GiB> --restart always < container_id >`

ก่อน

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8a3c0872a92b	secure_api_test	0.00%	12.93MiB / 1GiB	1.26%	51.3kB / 2.29kB	295kB / 24.6kB	10
ebac343176cc	service-api	0.00%	44.28MiB / 7.752GiB	0.56%	0B / 0B	1.22MB / 0B	14
04d1c7e30194	my-webapp	0.05%	97.37MiB / 7.752GiB	1.23%	172kB / 225kB	1.74MB / 8.19kB	39

หลัง

```
root@ta1slpgsql120:/# docker update --memory 2GiB --memory-swap 2GiB --restart always 8a3c0872a92b
8a3c0872a92b
root@ta1slpgsql120:/#
```

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
8a3c0872a92b	secure_api_test	0.00%	12.93MiB / 2GiB	0.63%	51.3kB / 2.29kB	295kB / 24.6kB	10
ebac343176cc	service-api	0.00%	44.28MiB / 7.752GiB	0.56%	0B / 0B	1.22MB / 0B	14
04d1c7e30194	my-webapp	0.12%	97.62MiB / 7.752GiB	1.23%	172kB / 225kB	1.74MB / 8.19kB	39



## 6. การแสดงเว็บแอปพลิเคชันและการค้นหา IP บน Linux

- ถ้าหากไม่พบปัญหาใดๆ สามารถเข้าถึงเว็บไซต์ที่ใช้งาน โดยใช้ IP และ port ของ Server

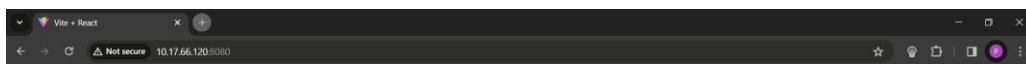
Note : หากลิ้ม IP Address ของเซิร์ฟเวอร์ สามารถตรวจสอบด้วยคำสั่ง ifconfig

```
root@talslpgsql120:/# ifconfig
br-284df2eb977c: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.18.0.1 netmask 255.255.0.0 broadcast 172.18.255.255
    inet6 fe80::42:f1ff:fe71:f47 prefixlen 64 scopeid 0x20<link>
    ether 02:42:f1:71:0f:47 txqueuelen 0 (Ethernet)
    RX packets 3727 bytes 10273975 (10.2 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4537 bytes 977256 (977.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    inet6 fe80::42:61ff:fe8b:b672 prefixlen 64 scopeid 0x20<link>
    ether 02:42:61:f8:b6:72 txqueuelen 0 (Ethernet)
    RX packets 22214 bytes 60323603 (60.3 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 21779 bytes 4338422 (4.3 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.17.66.120 netmask 255.255.255.0 broadcast 10.17.66.255
    inet6 fe80::250:56ff:febe:29d7 prefixlen 64 scopeid 0x20<link>
    ether 00:50:56:be:29:d7 txqueuelen 1000 (Ethernet)
    RX packets 112860181 bytes 45406217738 (45.4 GB)
    RX errors 0 dropped 66332 overruns 0 frame 0
    TX packets 93541142 bytes 15380786140 (15.3 GB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

- สามารถแสดงหน้า React Application Website ได้แล้ว



**Vite + React**

count is 0

Edit src/App.jsx and save to test HMR

Click on the Vite and React logos to learn more