

Lab Worksheet

ชื่อ-นามสกุล น.ส. พลอเรนซ์ แพร์ราร่า รหัสนักศึกษา 653380210-1 Section 3

Lab#8 – Software Deployment Using Docker

วัตถุประสงค์การเรียนรู้

1. ผู้เรียนสามารถอธิบายเกี่ยวกับ Software deployment ได้
2. ผู้เรียนสามารถสร้างและรัน Container จาก Docker image ได้
3. ผู้เรียนสามารถสร้าง Docker files และ Docker images ได้
4. ผู้เรียนสามารถนำซอฟต์แวร์ที่พัฒนาขึ้นให้สามารถรันบนสภาพแวดล้อมเดียวกันและทำงานร่วมกันกับสมาชิกในทีมพัฒนาซอฟต์แวร์ผ่าน Docker hub ได้
5. ผู้เรียนสามารถเริ่มต้นใช้งาน Jenkins เพื่อสร้าง Pipeline ในการ Deploy งานได้

Pre-requisite

1. ติดตั้ง Docker desktop ลงบนเครื่องคอมพิวเตอร์ โดยดาวน์โหลดจาก <https://www.docker.com/get-started>
2. สร้าง Account บน Docker hub (<https://hub.docker.com/signup>)
3. กำหนดให้ \$ หมายถึง Command prompt และ <> หมายถึง ให้ป้อนค่าของพารามิเตอร์ที่กำหนด

แบบฝึกปฏิบัติที่ 8.1 Hello world - รัน Container จาก Docker image

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
1. เปิด Command line หรือ Terminal บน Docker Desktop จากนั้นสร้าง Directory ชื่อ Lab8_1
2. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_1 เพื่อใช้เป็น Working directory
3. ป้อนคำสั่ง \$ docker pull busybox หรือ \$ sudo docker pull busybox สำหรับกรณีที่ติดปัญหา Permission denied
(หมายเหตุ: BusyBox เป็น software suite ที่รองรับคำสั่งบางอย่างบน Unix - <https://busybox.net>)
4. ป้อนคำสั่ง \$ docker images

[Check point#1] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

```
C:\Users\Lenovo\Lab8_1>docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
jenkins/jenkins      lts-jdk17          44c1caefd796       2 weeks ago        468MB
jupyterlab_image     latest             8039bc6e557c       3 months ago       313MB
docker               latest             113ff025ce2c       3 months ago       145MB
<none>              <none>            2492a96d1ec8       3 months ago       145MB
busybox              latest             af4709625109       3 months ago       4.27MB
synthesizedio/whalesay latest             07da125a0bc8       6 months ago       45.2MB

C:\Users\Lenovo\Lab8_1>
```

(1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร

ชื่อของ Docker Image ที่ระบุภาพนั้นมาจากแหล่งไหน

(2) Tag ที่ใช้บ่งบอกถึงอะไร

ใช้ดูเวอร์ชัน หรือ ลักษณะเฉพาะ ของ Docker Image นั้นๆ

5. ป้อนคำสั่ง \$ docker run busybox

6. ป้อนคำสั่ง \$ docker run -it busybox sh

7. ป้อนคำสั่ง ls

8. ป้อนคำสั่ง ls -la

9. ป้อนคำสั่ง exit

10. ป้อนคำสั่ง \$ docker run busybox echo "Hello ชื่อและนามสกุลของนักศึกษา from busybox"

11. ป้อนคำสั่ง \$ docker ps -a

[Check point#2] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ตั้งแต่ขั้นตอนที่ 6-12 พร้อมกับตอบคำถามต่อไปนี้

```
C:\Users\Lenovo\Lab8_1>docker run busybox
C:\Users\Lenovo\Lab8_1>docker run -it busybox sh
/ # ls
bin  dev  etc  home  lib  lib64  proc  root  sys  tmp  usr  var
/ # ls -la
total 48
drwxr-xr-x  1 root    root          4096 Jan 23 02:53 .
drwxr-xr-x  1 root    root          4096 Jan 23 02:53 ..
-rwxr-xr-x  1 root    root           0 Jan 23 02:53 .dockerenv
drwxr-xr-x  2 root    root        12288 Sep 26 21:31 bin
drwxr-xr-x  5 root    root          360 Jan 23 02:53 dev
drwxr-xr-x  1 root    root          4096 Jan 23 02:53 etc
drwxr-xr-x  2 nobody  nobody        4096 Sep 26 21:31 home
drwxr-xr-x  2 root    root          4096 Sep 26 21:31 lib
lrwxrwxrwx  1 root    root           3 Sep 26 21:31 lib64 -> lib
dr-xr-xr-x 270 root    root           0 Jan 23 02:53 proc
drwx----- 1 root    root          4096 Jan 23 02:53 root
dr-xr-xr-x 11 root    root           0 Jan 23 02:53 sys
drwxrwxrwt  2 root    root          4096 Sep 26 21:31 tmp
drwxr-xr-x  4 root    root          4096 Sep 26 21:31 usr
drwxr-xr-x  4 root    root          4096 Sep 26 21:31 var
/ # exit
C:\Users\Lenovo\Lab8_1>docker run busybox echo "Hello Florence Ferrara from busybox"
```

Lab Worksheet

```
C:\Users\Lenovo\Lab8_1>docker run busybox echo "Hello Florence Ferrara from busybox"
"Hello Florence Ferrara from busybox"

C:\Users\Lenovo\Lab8_1>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
a5951ac8a9be	busybox	echo "Hello Florenc..."	12 seconds ago	Exited (0) 11 seconds ago	
f97aa9c7c661	busybox	sh	About a minute ago	Exited (0) 54 seconds ago	
32d0ee5ca69e	busybox	sh	2 minutes ago	Exited (0) 2 minutes ago	
b45899fb0012	jenkins/jenkins:lts-jdk17	/usr/bin/tini -- /u...	15 minutes ago	Up 15 minutes	0.0.0.0:8080->8080
p, 0.0.0.0:50000->50000/tcp	reverent hofstadter				
3b72f763c854	jenkins/jenkins:lts-jdk17	/usr/bin/tini -- /u...	18 minutes ago	Exited (130) 17 minutes ago	
823a417dcb74	synthesizedio/whalesay:latest	/usr/local/bin/cows...	42 minutes ago	Exited (0) 42 minutes ago	
31fd74bffc8	synthesizedio/whalesay:latest	/usr/local/bin/cows...	42 minutes ago	Exited (0) 42 minutes ago	
154cc3778958	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (0) 3 months ago	
1709fb209750	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (255) 3 months ago	0.0.0.0:8888->8888
p	funny_bhaskara				
154cc3778958	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (0) 3 months ago	
1709fb209750	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (255) 3 months ago	0.0.0.0:8888->8888/tc
p	funny_bhaskara				
a17800bb359e	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (0) 3 months ago	
ebd0534d21e8	jupyterlab_image:latest	jupyter lab --ip=0...	3 months ago	Exited (0) 3 months ago	
46727ac6a984	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (255) 3 months ago	0.0.0.0:8888->8888/tc
p	tender_joliot				
a8625d2884f8	jupyterlab_image	jupyter lab --ip=0...	3 months ago	Exited (0) 3 months ago	
1758774b72d3	docker	uvicorn main:app --...	3 months ago	Exited (0) 3 months ago	
6e1e22790ed9	2492a96d1ec8	uvicorn main:app --...	3 months ago	Exited (1) 3 months ago	
803f8cf841de	2492a96d1ec8	uvicorn main:app --...	3 months ago	Exited (1) 3 months ago	
faacef52f11d	2492a96d1ec8	uvicorn main:app --...	3 months ago	Exited (1) 3 months ago	
74fc017fcd11	2492a96d1ec8	uvicorn main:app --...	3 months ago	Exited (1) 3 months ago	

```
C:\Users\Lenovo\Lab8_1>
```

- เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป (interactive mode) ทำให้คอนเทนเนอร์แสดงผลเหมือนเทอร์มินัลจริง เช่นจากคำสั่ง docker run -it busybox sh จะทำการเปิด shell
- คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร
แสดงสถานะของคอนเทนเนอร์ในปัจจุบัน เช่น Up คือทำงานอยู่ Exited หยุดทำงานแล้ว พร้อมบอกเวลา Created คอนเทนเนอร์ถูกสร้างขึ้นแล้ว

12. ป้อนคำสั่ง \$ docker rm <container ID ที่ต้องการลบ>

[Check point#3] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ในขั้นตอนที่ 13

```
C:\Users\Lenovo\Lab8_1>docker rm 154cc3778958
154cc3778958
```

Lab Worksheet

แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_2
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_2 เพื่อใช้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ (Windows) บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

FROM busybox

CMD echo "Hi there. This is my first docker image."

CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา ชื่อเล่น"

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal และป้อนคำสั่งต่อไปนี้

\$ cat > Dockerfile << EOF

FROM busybox

CMD echo "Hi there. This is my first docker image."

CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา ชื่อเล่น"

EOF

หรือใช้คำสั่ง

\$ touch Dockerfile

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

\$ docker build -t <ชื่อ Image> .

6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

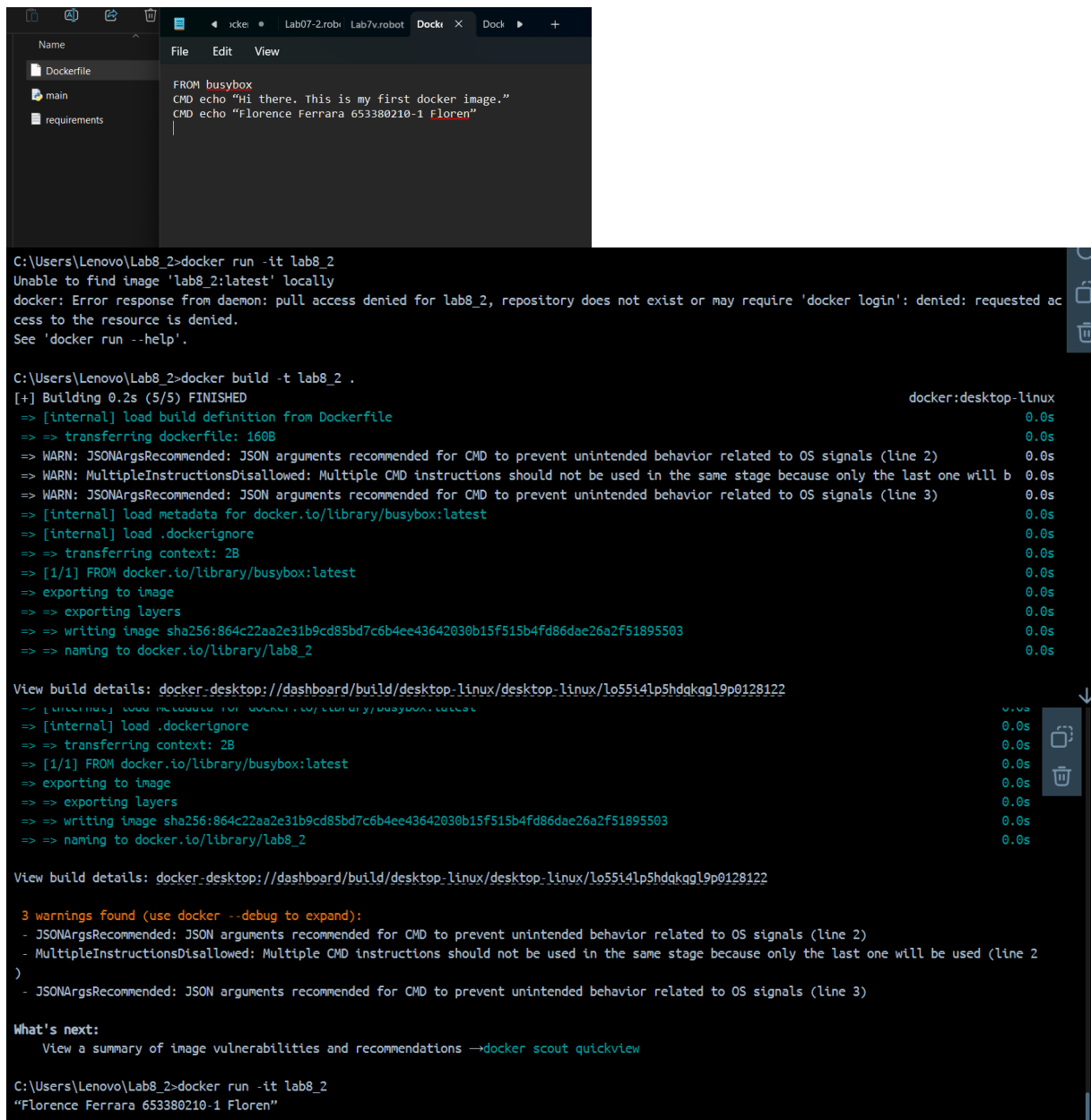
[Check point#4] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ในขั้นตอนที่ 5 พร้อมกับตอบคำถามต่อไปนี้

```
C:\Users\Lenovo>mkdir Lab8_2
```

```
C:\Users\Lenovo>cd Lab8_2
```

```
C:\Users\Lenovo\Lab8_2>echo. > Dockerfile.swp
```

Lab Worksheet



The screenshot shows a Dockerfile in a text editor and a terminal window. The Dockerfile contains the following content:

```
FROM busybox
CMD echo "Hi there. This is my first docker image."
CMD echo "Florence Ferrara 653380210-1 Floren"
```

The terminal shows the following commands and outputs:

```
C:\Users\Lenovo\Lab8_2>docker run -it lab8_2
Unable to find image 'lab8_2:latest' locally
docker: Error response from daemon: pull access denied for lab8_2, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.
```

```
C:\Users\Lenovo\Lab8_2>docker build -t lab8_2 .
[+] Building 0.2s (5/5) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 160B 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2) 0.0s
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will b 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3) 0.0s
=> [internal] load metadata for docker.io/library/busybox:latest 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [1/1] FROM docker.io/library/busybox:latest 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:864c22aa2e31b9cd85bd7c6b4ee43642030b15f515b4fd86dae26a2f51895503 0.0s
=> => naming to docker.io/library/lab8_2 0.0s
```

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/1o55i4lp5hdqkqgl9p0128122

```
-> [internal] load metadata for docker.io/library/busybox:latest 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [1/1] FROM docker.io/library/busybox:latest 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:864c22aa2e31b9cd85bd7c6b4ee43642030b15f515b4fd86dae26a2f51895503 0.0s
=> => naming to docker.io/library/lab8_2 0.0s
```

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/1o55i4lp5hdqkqgl9p0128122

3 warnings found (use docker --debug to expand):

- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
- MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)

What's next:
View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

```
C:\Users\Lenovo\Lab8_2>docker run -it lab8_2
"Florence Ferrara 653380210-1 Floren"
```

(1) คำสั่งที่ใช้ในการ run คือ

`docker run -it lab8_2`

(2) Option -t ในคำสั่ง `$ docker build` ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

ใช้สำหรับการตั้งชื่อ (tag) ให้กับ Docker image เช่น `docker build -t lab8_2:v1` . ชื่อ tag จะเป็น v1 แต่ถ้าหากไม่ตั้งค่าในส่วนนี้ ชื่อ tag จะมี default เป็น latest

Lab Worksheet

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_3
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_3 เพื่อใช้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

```
FROM busybox
```

```
CMD echo "Hi there. My work is done. You can run them from my Docker image."
```

```
CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา"
```

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal และป้อนคำสั่งต่อไปนี้

```
$ cat > Dockerfile << EOF
```

```
FROM busybox
```

```
CMD echo "Hi there. My work is done. You can run them from my Docker image."
```

```
CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา"
```

```
EOF
```

หรือใช้คำสั่ง

```
$ touch Dockerfile
```

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

7. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

```
$ docker build -t <username ที่ลงทะเบียนกับ Docker Hub>/lab8
```

5. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง

```
$ docker run <username ที่ลงทะเบียนกับ Docker Hub>/lab8
```

[Check point#5] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ในขั้นตอนที่ 5

Lab Worksheet

```

PS C:\Users\Lenovo> mkdir Lab8_3

Directory: C:\Users\Lenovo

Mode                LastWriteTime         Length Name
----                -
d-----          1/28/2025   9:18 PM             Lab8_3

PS C:\Users\Lenovo> cd Lab8_3

PS C:\Users\Lenovo\Lab8_3> touch Dockerfile

Directory: C:\Users\Lenovo\Lab8_3

Mode                LastWriteTime         Length Name
----                -
-a-----          1/28/2025   9:22 PM             0 Dockerfile

PS C:\Users\Lenovo\Lab8_3>

FROM busybox
CMD echo "Hi there. My work is done. You can run them from my Docker image."
CMD echo "Florence Ferrara 653380210-1 Floren"

PS C:\Users\Lenovo\Lab8_3> docker build -t florence04/lab8 .
[+] Building 0.2s (5/5) FINISHED
=> [internal] load metadata for docker.io/library/busybox:latest 0.0s
=> [internal] load .dockerignore 0.0s
=> transferring context: 2B 0.0s
=> CACHED [1/1] FROM docker.io/library/busybox:latest 0.0s
=> exporting to image 0.0s
=> exporting layers 0.0s
=> writing image sha256:e166555cf0aa461f71e9f7cbf31cc36c6ff9a51bcaad263954ff8ebc13f80335 0.0s
=> naming to docker.io/florence04/lab8 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/iztraxtmyp6sbns0jies67tne

3 warnings found (use docker --debug to expand):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
- MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)

What's next:
View a summary of image vulnerabilities and recommendations ->docker scout quickview
PS C:\Users\Lenovo\Lab8_3>

PS C:\Users\Lenovo\Lab8_3> docker run -t florence04/lab8
"Florence Ferrara 653380210-1 Floren"
PS C:\Users\Lenovo\Lab8_3>

```

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยใช้คำสั่ง

```
$ docker push <username ที่ลงทะเบียนกับ Docker Hub>/lab8
```

ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

```
$ docker login แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง
```

```
$ docker login -u <username> -p <password>
```

7. ไปที่ Docker Hub กด Tab ชื่อ Tags หรือไปที่ Repository ก็ได้

[Check point#6] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดง Repository ที่มี Docker image (<username>/lab8)

Lab Worksheet

```
PS C:\Users\Lenovo\Lab8_3> docker push florence04/lab8
Using default tag: latest
The push refers to repository [docker.io/florence04/lab8]
59654b79daad: Mounted from library/busybox
latest: digest: sha256:b68970de5be68f19ccc772e7d14203ce9b2142c9995dbf2ecb4d7d55d86263f9 size: 527
PS C:\Users\Lenovo\Lab8_3>
```

The screenshot displays the Docker Hub interface for the repository `florence04/lab8`. The top navigation bar includes links for Explore, Repositories, Organizations, and Usage, along with a search bar and a 'Create repository' button. The repository page shows the name `florence04/lab8`, the last pushed time (3 minutes ago), and the visibility (Public). The 'Tags' section lists the `latest` tag, pushed 3 minutes ago. The 'Automated builds' section provides instructions on how to connect to GitHub or Bitbucket for automated builds. The 'Image Layers' section shows the layers of the image, including the base image `BusyBox 1.37.0 (glibc, Debian 12)` and the command `CMD ["/bin/sh" "-c" "echo 'Hi'"]`.

Repository Details:

- Name: `florence04/lab8`
- Last Pushed: 3 minutes ago
- Contains: IMAGE
- Visibility: Public
- Scout: Inactive

Tags:

Tag	OS	Type	Pulled	Pushed
latest	linux/amd64	Image	3 minutes ago	3 minutes ago

Automated builds:

Manually pushing images to Docker Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#).

Image Layers:

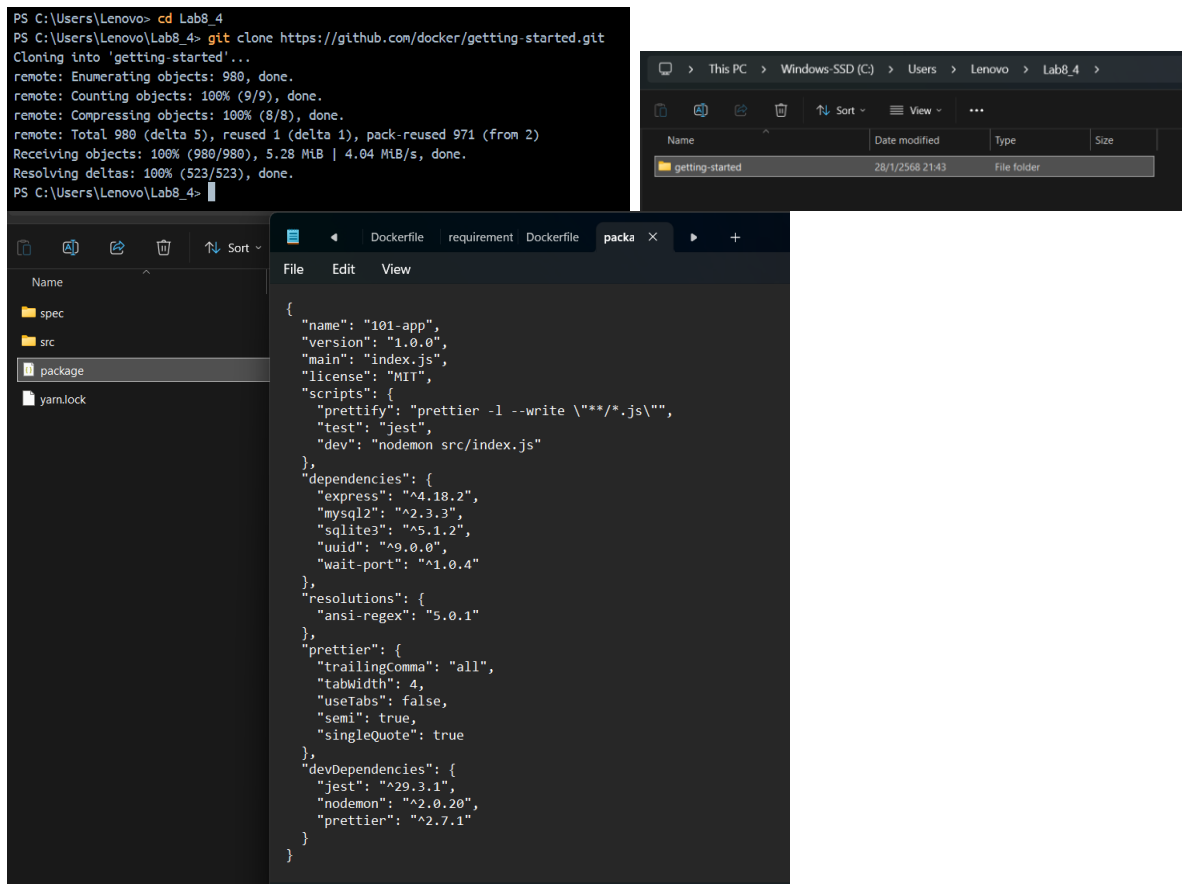
Layer	Command	Size
1	BusyBox 1.37.0 (glibc, Debian 12)	2.06 MB
2	CMD ["/bin/sh" "-c" "echo 'Hi'"]	0 B
3	CMD ["/bin/sh" "-c" "echo 'Florence'"]	0 B

Lab Worksheet

แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4
2. ทำการ Clone ซอร์สโค้ดของเว็บแอปพลิเคชันจาก GitHub repository
<https://github.com/docker/getting-started.git> ลงใน Directory ที่สร้างขึ้น โดยใช้คำสั่ง
`$ git clone https://github.com/docker/getting-started.git`
3. เปิดดูองค์ประกอบภายใน getting-started/app เมื่อพบไฟล์ package.json ให้ใช้ Text editor ในการเปิดอ่าน

[Check point#7] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงที่อยู่ของ Source code ที่ Clone มาและเนื้อหาของไฟล์ package.json



4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไปไฟล์
FROM node:18-alpine

Lab Worksheet

```
WORKDIR /app
```

```
COPY . .
```

```
RUN yarn install --production
```

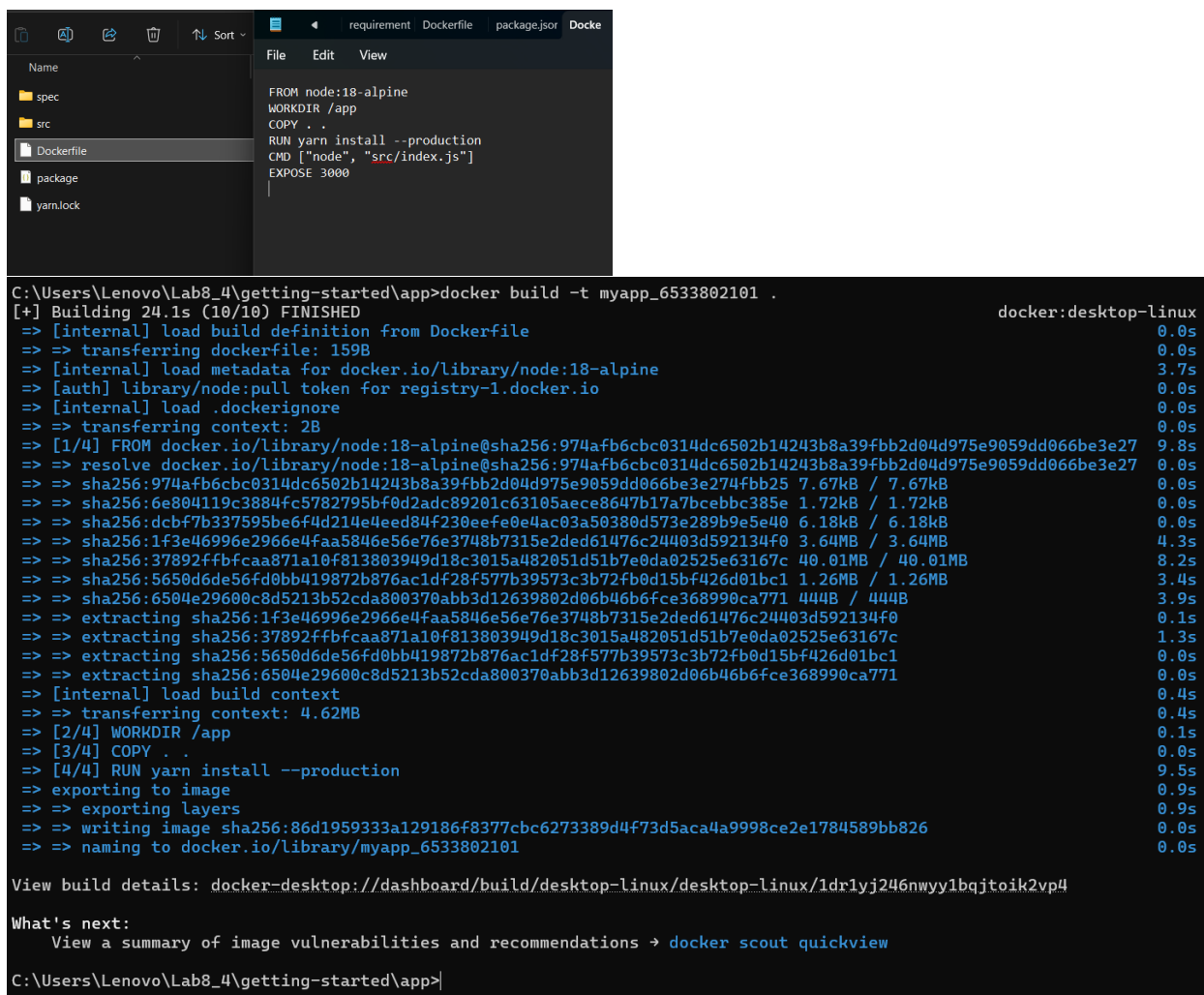
```
CMD ["node", "src/index.js"]
```

```
EXPOSE 3000
```

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดใช้ชื่อ image เป็น myapp_รหัสสนศ. ไม่มีขีด

```
$ docker build -t <myapp_รหัสสนศ. ไม่มีขีด> .
```

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ



```

C:\Users\Lenovo\Lab8_4\getting-started\app>docker build -t myapp_6533802101 .
[+] Building 24.1s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 159B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [auth] library/node:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e27
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e27
=> => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 7.67kB / 7.67kB
=> => sha256:6e804119c3884fc5782795bf0d2adc89201c63105aece8647b17a7bcebbbc385e 1.72kB / 1.72kB
=> => sha256:dcbf7b337595be6f4d214e4eed84f230ee0e4ac03a50380d573e289b9e5e40 6.18kB / 6.18kB
=> => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 3.64MB / 3.64MB
=> => sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 40.01MB / 40.01MB
=> => sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 1.26MB / 1.26MB
=> => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 444B / 444B
=> => extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0
=> => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c
=> => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1
=> => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771
=> [internal] load build context
=> => transferring context: 4.62MB
=> [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => writing image sha256:86d1959333a129186f8377cbc6273389d4f73d5aca4a9998ce2e1784589bb826
=> => naming to docker.io/library/myapp_6533802101

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/1dr1yj246nwyy1bqjtoik2vp4

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Users\Lenovo\Lab8_4\getting-started\app>

```

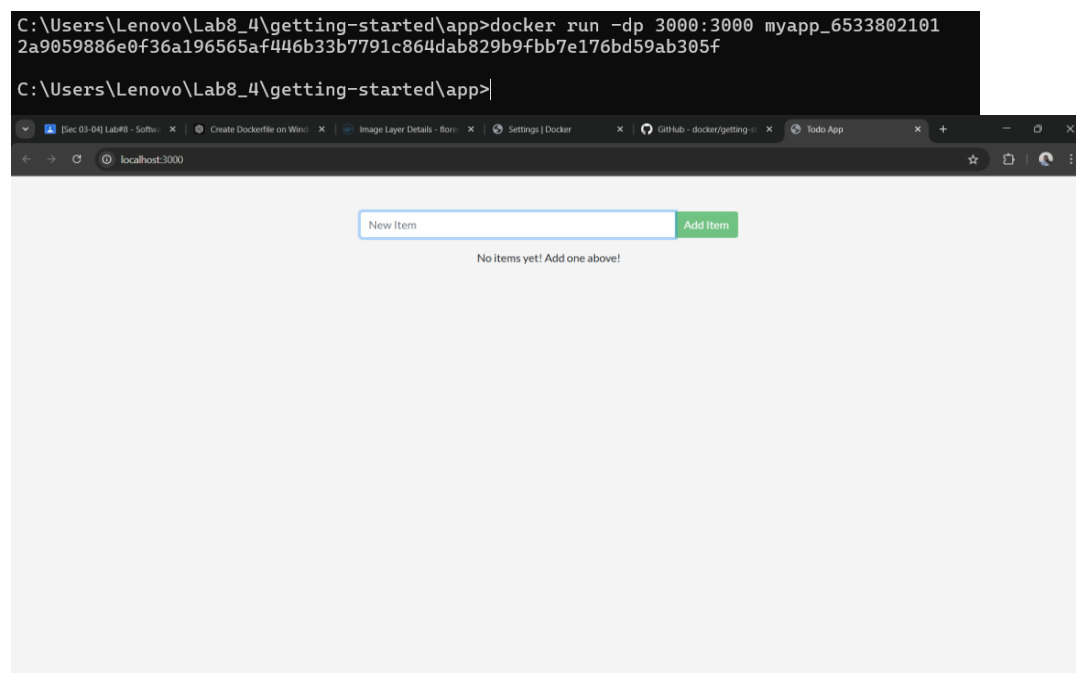
Lab Worksheet

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

```
$ docker run -dp 3000:3000 <myapp_รหัสสนศ. ไม่มีขีด>
```

7. เปิด Browser ไปที่ URL = <http://localhost:3000>

[Check point#9] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้บน Browser และ Dashboard ของ Docker desktop



หมายเหตุ: นศ.สามารถทดลองเล่น Web application ที่ทำงานอยู่ได้

8. ทำการแก้ไข Source code ของ Web application ดังนี้

- a. เปิดไฟล์ src/static/js/app.js ด้วย Editor และแก้ไขบรรทัดที่ 56 จาก

```
<p className="text-center">No items yet! Add one above!</p> เป็น
```

```
<p className="text-center">There is no TODO item. Please add one to the list.
```

```
By ชื่อและนามสกุลของนักศึกษา</p>
```

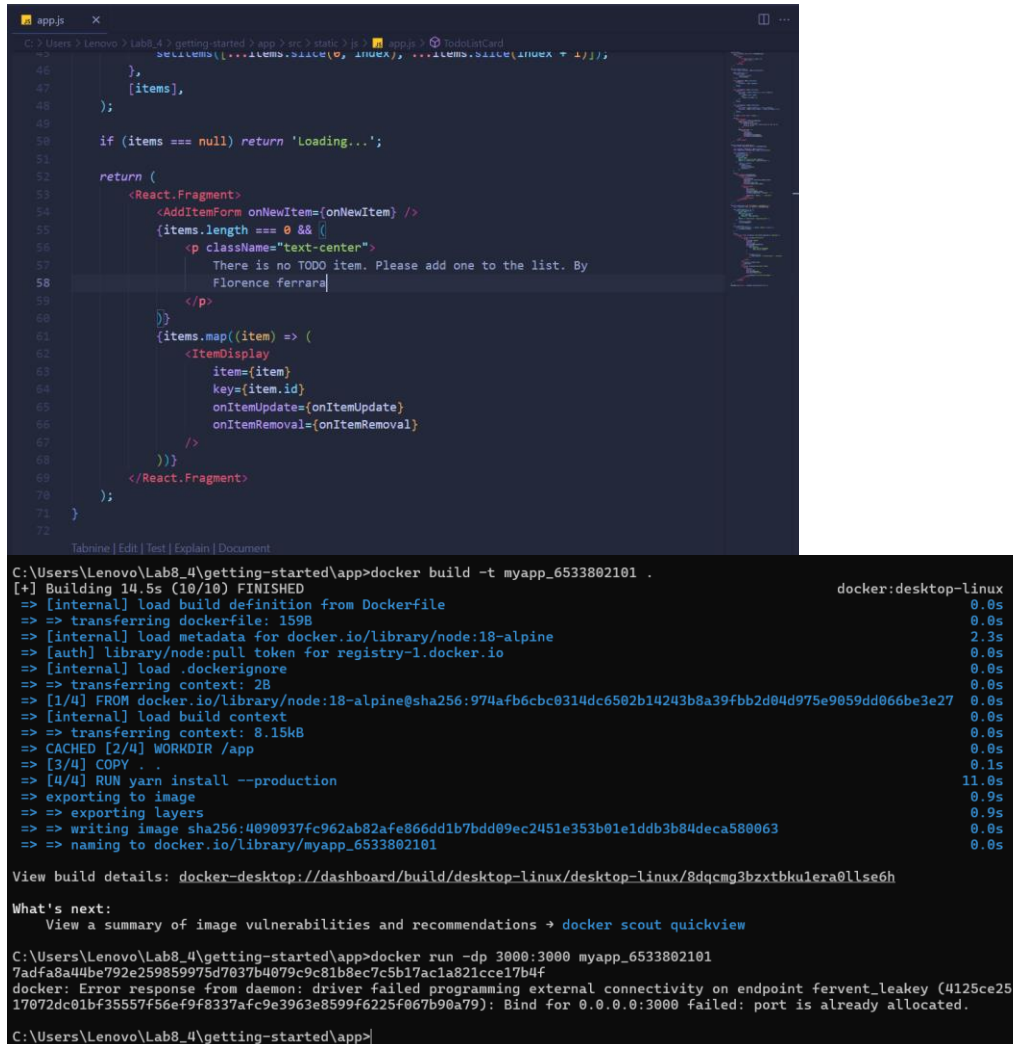
- b. Save ไฟล์ให้เรียบร้อย

9. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

10. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

Lab Worksheet

[Check point#10] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ พร้อมกับตอบคำถามต่อไปนี้



```

46  },
47  [items],
48  );
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">
57          There is no TODO item. Please add one to the list. By
58          Florence ferrara
59        </p>
60      )}
61      {items.map((item) => (
62        <ItemDisplay
63          item={item}
64          key={item.id}
65          onItemUpdate={onItemUpdate}
66          onItemRemoval={onItemRemoval}
67        </ItemDisplay>
68      ))}
69    </React.Fragment>
70  );
71 }
72

```

```

C:\Users\Lenovo\Lab8_4\getting-started\app>docker build -t myapp_6533802101 .
[*] Building 14.5s (10/10) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 159B                                              0.0s
=> [internal] load metadata for docker.io/library/node:18-alpine                2.3s
=> [auth] library/node:pull token for registry-1.docker.io                      0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                    0.0s
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e27 0.0s
=> [internal] load build context                                                0.0s
=> => transferring context: 8.15kB                                                0.0s
=> CACHED [2/4] WORKDIR /app                                                    0.0s
=> [3/4] COPY . .                                                                0.1s
=> [4/4] RUN yarn install --production                                         11.0s
=> exporting to image                                                            0.9s
=> => exporting layers                                                            0.9s
=> writing image sha256:4090937fc962ab82afe866dd1b7bdd09ec2451e353b01e1ddb3b84deca580063 0.0s
=> naming to docker.io/library/myapp_6533802101                                0.0s

View build details: docker-desktop:///dashboard/build/desktop-linux/desktop-linux/8dqcmg3bzxrbkulera0llse6h

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Users\Lenovo\Lab8_4\getting-started\app>docker run -dp 3000:3000 myapp_6533802101
7adfa8a44bbe792e259859975d7037b4079c9c81b8ec7c5b17ac1a821cce17b4f
docker: Error response from daemon: driver failed programming external connectivity on endpoint fervent_leakey (4125ce2517072dc01bf35557f56ef9f8337afc9e3963e8599f6225f067b90a79): Bind for 0.0.0.0:3000 failed: port is already allocated.

C:\Users\Lenovo\Lab8_4\getting-started\app>

```

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

พอร์ต 3000 มันชนหรือว่าถูกใช้งานแล้ว นั่นคือใช้ในข้อ 5-6 ไปแล้ว Docker ไม่สามารถผูกพอร์ต 3000 นี้กับคอนเทนเนอร์ใหม่ได้

11. ลบ Container ของ Web application เวอร์ชันก่อนแก้ไขออกจากระบบ โดยใช้วิธีใดวิธีหนึ่งดังต่อไปนี้

a. ผ่าน Command line interface

- ใช้คำสั่ง `$ docker ps` เพื่อดู Container ID ที่ต้องการจะลบ
- Copy หรือบันทึก Container ID ไว้

Lab Worksheet

- iii. ใช้คำสั่ง `$ docker stop <Container ID ที่ต้องการจะลบ>` เพื่อหยุดการทำงานของ Container ดังกล่าว
 - iv. ใช้คำสั่ง `$ docker rm <Container ID ที่ต้องการจะลบ>` เพื่อทำการลบ
- b. ผ่าน Docker desktop
 - i. ไปที่หน้าต่าง Containers
 - ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
 - iii. ยืนยันโดยการกด Delete forever
12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6
 13. เปิด Browser ไปที่ URL = <http://localhost:3000>

[Check point#11] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้บน Browser และ Dashboard ของ Docker desktop

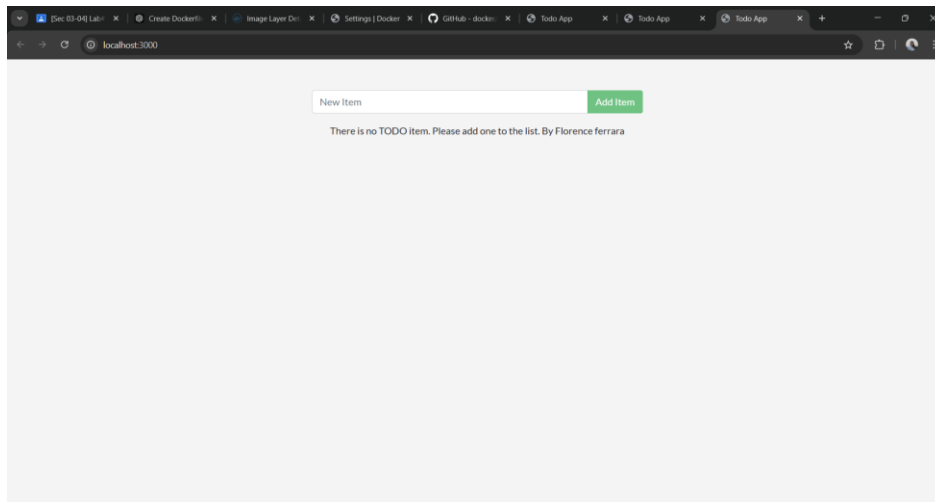
```
C:\Users\Lenovo\Lab8_4\getting-started\app>docker ps
CONTAINER ID   IMAGE                                NAMES      COMMAND
2a9059886e0f   86d1959333a1                       "docker-ent...
etent_brahmagupta
3b72f763c854   jenkins/jenkins:lts-jdk17         "/usr/bin/tini -- /u...
uent_jennings

PS C:\Users\Lenovo\Lab8_4\getting-started\app>docker stop 2a9059886e0f
2a9059886e0f
PS C:\Users\Lenovo\Lab8_4\getting-started\app>docker rm 2a9059886e0f
2a9059886e0f
PS C:\Users\Lenovo\Lab8_4\getting-started\app>docker build -t myapp_6533802101 .
[+] Building 1.1s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 159B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:974af6b6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25
=> [internal] load build context
=> => transferring context: 2.49kB
=> CACHED [2/4] WORKDIR /app
=> CACHED [3/4] COPY . .
=> CACHED [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => writing image sha256:4090937fc962ab82afe866dd1b7bdd09ec2451e353b01e1ddb3b84deca580063
=> => naming to docker.io/library/myapp_6533802101

View build details: docker_desktop://dashboard/build/desktop.linux/desktop.linux/lokargo6rwykn710y0rp2usgf

What's next:
  View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS C:\Users\Lenovo\Lab8_4\getting-started\app>docker run -dp 3000:3000 myapp_6533802101
9e7185350225a77878c1d1a6e4e6e73ab0cba572ac23d198d46eb5617a7169f4
```

Lab Worksheet



แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

1. เปิด Command line หรือ Terminal บน Docker Desktop

2. ป้อนคำสั่งและทำการรัน container โดยผูกพอร์ต

```
$ docker run -p 8080:8080 -p 50000:50000 --restart=on-failure jenkins/jenkins:lts-jdk17
```

หรือ

```
$ docker run -p 8080:8080 -p 50000:50000 --restart=on-failure -v
```

```
jenkins_home:/var/jenkins_home jenkins/jenkins:lts-jdk17
```

3. บันทึกที่รหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก

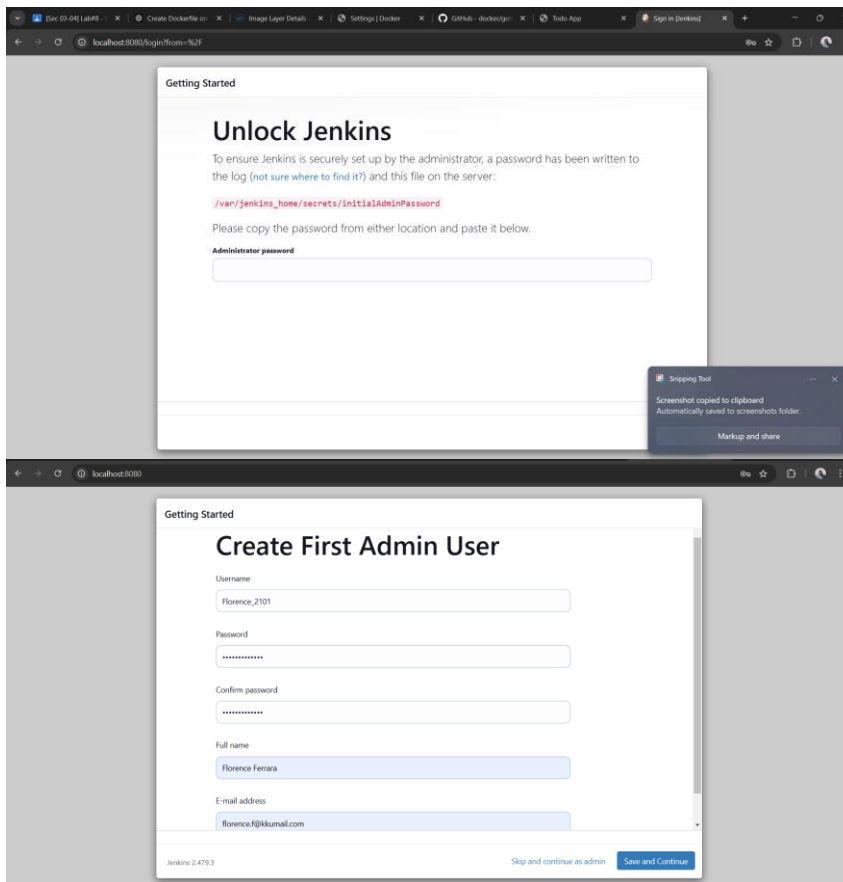
[Check point#12] Capture หน้าจอที่แสดงผล Admin password

```
PS C:\Users\Lenovo> docker run -p 8080:8080 -p 50000:50000 --restart=on-failure jenkins/jenkins:lts-jdk17
Running from: /usr/share/jenkins/jenkins.war
webroot: /var/jenkins_home/war
2025-01-28 16:31:53.303+0000 [id=1] INFO winstone.Logger#logInternal: Beginning extraction from war file
2025-01-28 16:31:53.929+0000 [id=1] WARNING o.e.j.ee9.nested.ContextHandler#setContextPath: Empty contextPath
2025-01-28 16:31:53.977+0000 [id=1] INFO org.eclipse.jetty.server.Server#doStart: jetty-12.0.16; built: 2024-12-09T21:02:54.535Z; git: c3f88bafb4e393f23204dc
14dc57b042e84debc7; jvm 17.0.13+11
2025-01-28 16:31:54.289+0000 [id=1] INFO o.e.j.e.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.ee9.jsp.Jet
tyJspServlet
2025-01-28 16:31:54.332+0000 [id=1] INFO o.e.j.s.DefaultSessionIdManager#doStart: Session workerName=node0
2025-01-28 16:31:54.614+0000 [id=1] INFO hudson.WebAppMain#contextInitialized: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get(
"JENKINS_HOME")
2025-01-28 16:31:54.734+0000 [id=1] INFO o.e.j.s.handler.ContextHandler#doStart: Started oeje9n.ContextHandler$CoreContextHandler@772861aa[Jenkins v2.479.3,/
_b=file:///var/jenkins_home/war/,a=AVAILABLE,h=oeje9n.ContextHandler$CoreContextHandler$CoreToNestedHandler@6631cb64[STARTED]]
2025-01-28 16:31:54.747+0000 [id=1] INFO o.e.j.server.AbstractConnector#doStart: Started ServerConnector@44550792(HTTP/1.1, (http/1.1)){0.0.0.0:8080}
2025-01-28 16:31:54.758+0000 [id=1] INFO org.eclipse.jetty.server.Server#doStart: Started oejs.Server@1c7fd41f[STARTING][12.0.16,sto=0] @1910ms
2025-01-28 16:31:54.759+0000 [id=35] INFO winstone.Logger#logInternal: Winstone Servlet Engine running: controlPort=disabled
2025-01-28 16:31:54.904+0000 [id=43] INFO jenkins.InitReactorRunner$1onAttained: Started initialization
2025-01-28 16:31:54.918+0000 [id=63] INFO jenkins.InitReactorRunner$1onAttained: Listed all plugins
2025-01-28 16:31:55.457+0000 [id=72] INFO jenkins.InitReactorRunner$1onAttained: Prepared all plugins
2025-01-28 16:31:55.460+0000 [id=43] INFO jenkins.InitReactorRunner$1onAttained: Started all plugins
2025-01-28 16:31:55.461+0000 [id=52] INFO jenkins.InitReactorRunner$1onAttained: Augmented all extensions
2025-01-28 16:31:55.586+0000 [id=66] INFO jenkins.InitReactorRunner$1onAttained: System config loaded
2025-01-28 16:31:55.586+0000 [id=66] INFO jenkins.InitReactorRunner$1onAttained: System config adapted
```

Lab Worksheet

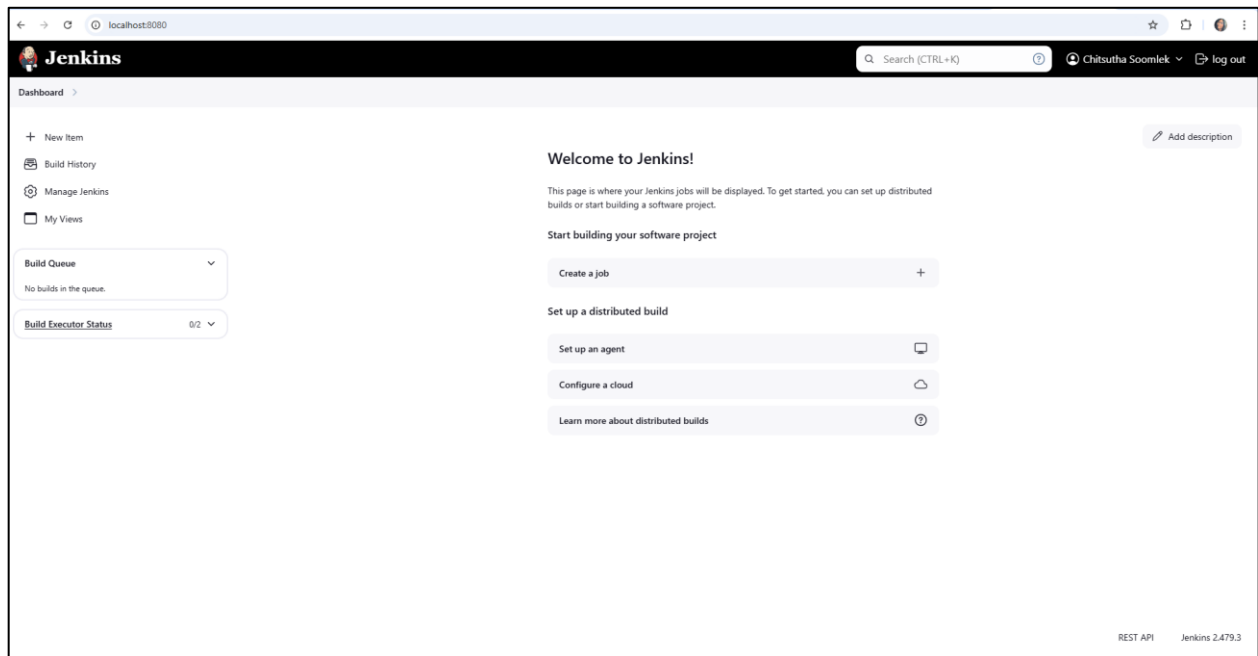
- เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น localhost:8080
- ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3
- สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

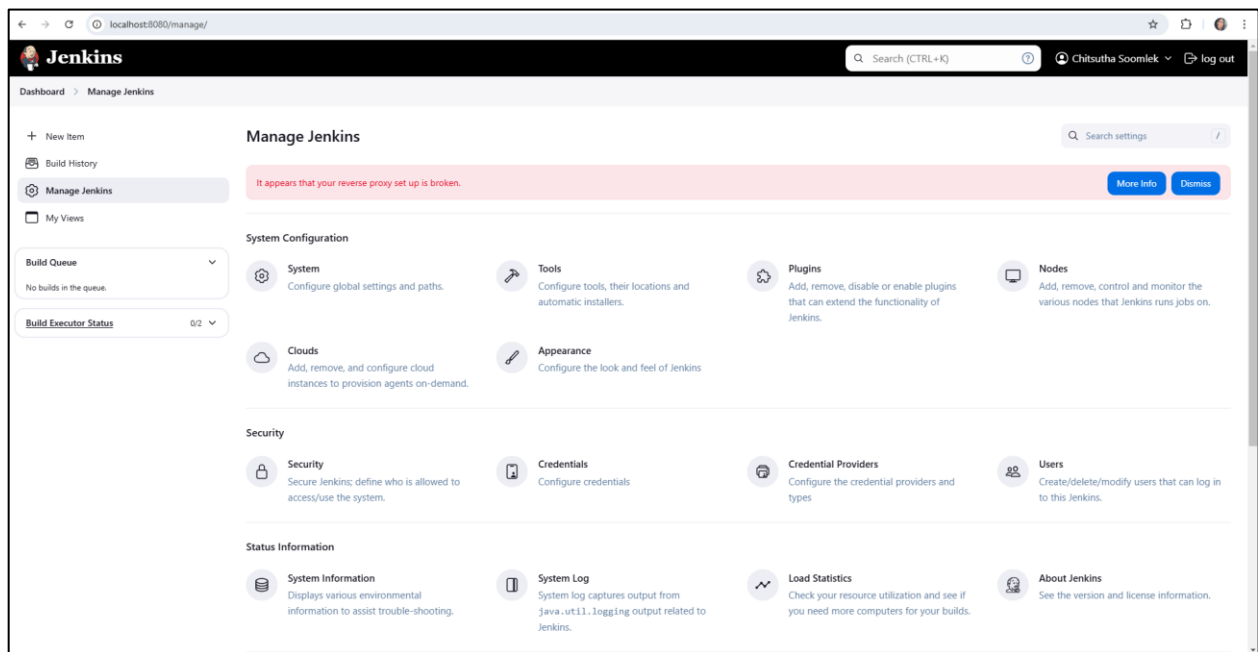


- กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>
- เมื่อติดตั้งเรียบร้อยแล้วจะพบหน้าจอ Dashboard ดังแสดงในภาพ

Lab Worksheet

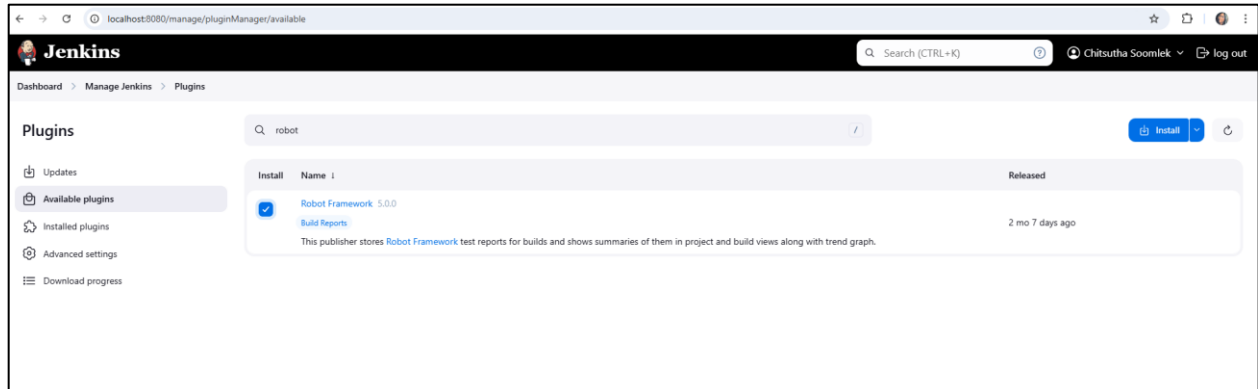


9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

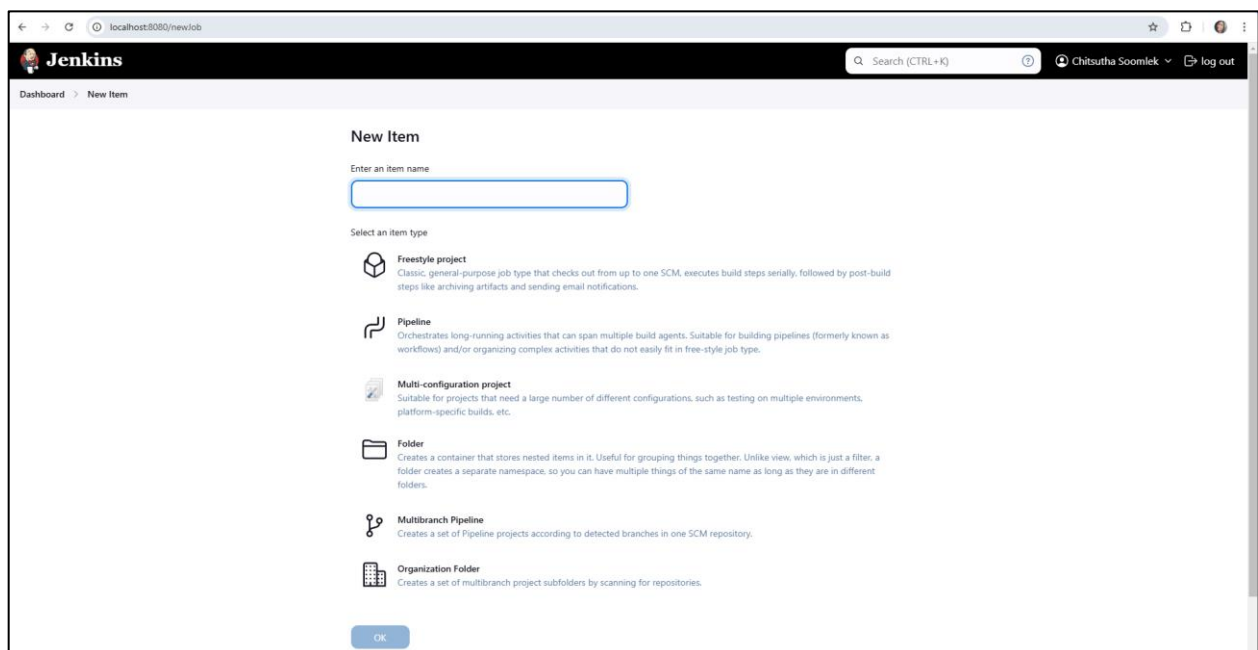


Lab Worksheet

10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



11. กลับไปที่หน้า Dashboard แล้วสร้าง Pipeline อย่างง่าย โดยกำหนด New item เป็น Freestyle project และตั้งชื่อเป็น UAT



12. นำไฟล์ .robot ที่ทำให้แบบฝึกปฏิบัติที่ 7 (Lab#7) ไปไว้บน Repository ของนักศึกษา จากนั้นตั้งค่าที่จำเป็นในหน้านี้ทั้งหมด ดังนี้

Description: Lab 8.5

GitHub project: กดเลือก แล้วใส่ Project URL เป็น repository ที่เก็บโค้ด .robot (ดูขั้นตอนที่ 12)

Build Trigger: เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

Lab Worksheet

Build Steps: เลือก Execute shell แล้วใส่คำสั่งในการรันไฟล์ .robot (หากไฟล์ไม่ได้อยู่ในหน้าแรกของ repository ให้ใส่ Path ไปถึงไฟล์ให้เรียบร้อยด้วย)

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Configure

General Enabled

Description: Lab 8.5

Plain text [Preview](#)

☐ Discard old builds ?

☒ GitHub project

Project url ?
https://github.com/TheTable1/Soften_Lab8.git

[Advanced](#)

☐ This project is parameterized ?

[Save](#) [Apply](#)

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☐ Build after other projects are built ?

☒ Build periodically ?

Schedule ?

H/15 * * * *

Would last have run at Tuesday, January 28, 2025 at 4:50:24 PM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 5:05:24 PM Coordinated Universal Time.

Build Steps

Execute shell ?

Command

See [the list of available environment variables](#)

robot Lab07-2.robot

Lab Worksheet

- (1) คำสั่งที่ใช้ในการ Execute ไฟล์ .robot ใน Build Steps คือ
robot Lab07-2.robot

Post-build action: เพิ่ม Publish Robot Framework test results -> ระบุไดเรกทอรีที่เก็บไฟล์ผลการทดสอบโดย Robot framework ในรูป xml และ html -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ไม่ผ่าน แล้วนับว่าซอฟต์แวร์มีปัญหา -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ผ่านแล้วนับว่าซอฟต์แวร์มีอยู่ในสถานะที่สามารถนำไปใช้งานได้ (เช่น 20, 80)

13. กด Apply และ Save

14. สั่ง Build Now

[Check point#15] Capture หน้าจอแสดงหน้าหลักของ Pipeline และ Console Output

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions**

Publish Robot Framework test results

Directory of Robot output
Path to directory containing robot xml and html files (relative to build workspace)

Soften_Lab8/results

Advanced Edited

Thresholds for build result

20.0%

80.0%

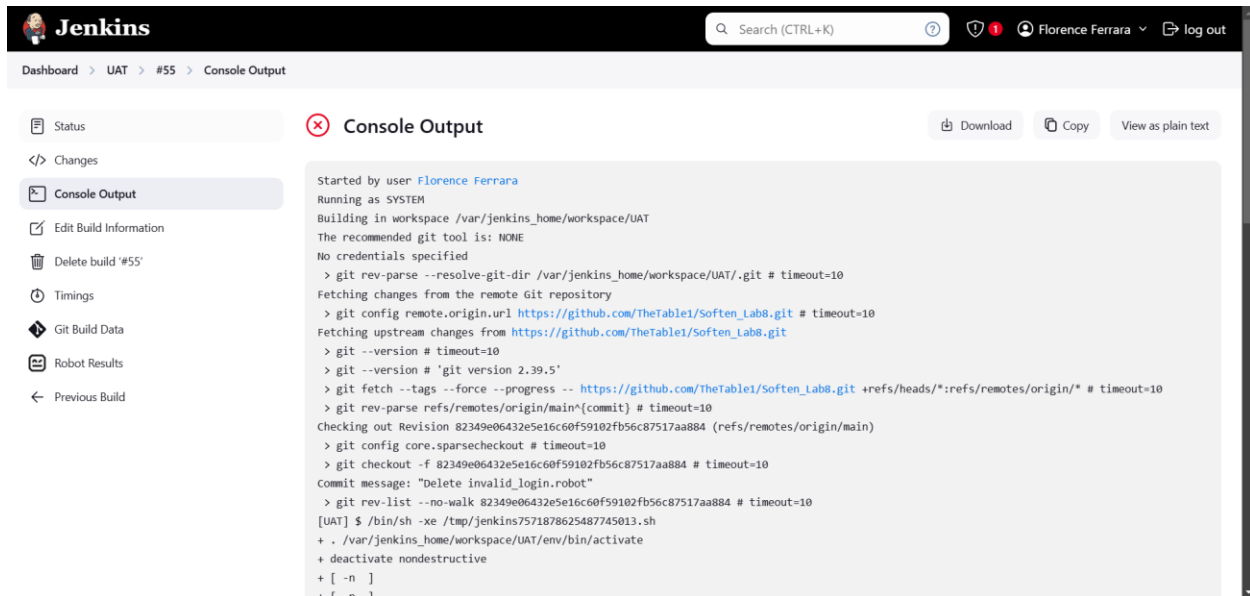
☒ DEPRECATED! THIS FLAG DOES NOTHING! - Use thresholds for critical tests only

☐ Include skipped tests in total count for thresholds

Add post-build action

Save Apply

Lab Worksheet



Jenkins Search (CTRL+K) Florence Ferrara log out

Dashboard > UAT > #55 > Console Output

Status
Changes
Console Output
Edit Build Information
Delete build #55
Timings
Git Build Data
Robot Results
Previous Build

Console Output Download Copy View as plain text

```
Started by user Florence Ferrara
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/UAT/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/TheTable1/Soften_Lab8.git # timeout=10
Fetching upstream changes from https://github.com/TheTable1/Soften_Lab8.git
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/TheTable1/Soften_Lab8.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 82349e06432e5e16c60f59102fb56c87517aa884 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 82349e06432e5e16c60f59102fb56c87517aa884 # timeout=10
Commit message: "Delete invalid_login.robot"
> git rev-list --no-walk 82349e06432e5e16c60f59102fb56c87517aa884 # timeout=10
[UAT] $ /bin/sh -xe /tmp/jenkins7571878625487745013.sh
+ . /var/jenkins_home/workspace/UAT/env/bin/activate
+ deactivate nondestructive
+ [ -n ]
+ f -n l
```

Lab Worksheet

The screenshot shows the Jenkins console output for a build named 'UAT #55'. The output displays the execution of a shell script that sets up a virtual environment and runs a test suite. The test suite 'Valid Login' failed due to a missing resource file 'resource.robot'.

```

Commit message: "Delete invalid_login.robot"
> git rev-list --no-walk 82349e06432e5e16c60f59102fb56c87517aa884 # timeout=10
[UAT] $ /bin/sh -xe /tmp/jenkins7571878625487745013.sh
+ . /var/jenkins_home/workspace/UAT/env/bin/activate
+ deactivate nondestructive
+ [ -n ]
+ [ -n ]
+ [ -n -o -n ]
+ [ -n ]
+ unset VIRTUAL_ENV
+ unset VIRTUAL_ENV_PROMPT
+ [ ! nondestructive = nondestructive ]
+ VIRTUAL_ENV=/var/jenkins_home/workspace/UAT/env
+ export VIRTUAL_ENV
+ _OLD_VIRTUAL_PATH=C:\Users\Lenovo\AppData\Local\Programs\Python\Python312\Scripts\robot.exe
+ PATH=/var/jenkins_home/workspace/UAT/env/bin:C:\Users\Lenovo\AppData\Local\Programs\Python\Python312\Scripts\robot.exe
+ export PATH
+ [ -n ]
+ [ -z ]
+ _OLD_VIRTUAL_PS1=$
+ PS1=(env) $
+ export PS1
+ VIRTUAL_ENV_PROMPT=(env)
+ export VIRTUAL_ENV_PROMPT
+ [ -n -o -n ]
+ export PATH=/var/jenkins_home/workspace/UAT/env/bin:C:\Users\Lenovo\AppData\Local\Programs\Python\Python312\Scripts\robot.exe
+ mkdir -p results
+ robot --outputdir results valid_login.robot
[ ERROR ] Error in file '/var/jenkins_home/workspace/UAT/valid_login.robot' on line 6: Resource file 'resource.robot' does not exist.

Valid Login :: A test suite with a single test for valid login.
=====
Valid Login | FAIL |
No keyword with name 'Open Browser To Login Page' found.

Also teardown failed:
No keyword with name 'Close Browser' found.
-----
Valid Login :: A test suite with a single test for valid login. | FAIL |
1 test, 0 passed, 1 failed
-----
Output: /var/jenkins_home/workspace/UAT/results/output.xml
Log: /var/jenkins_home/workspace/UAT/results/log.html
Report: /var/jenkins_home/workspace/UAT/results/report.html
Build step 'Execute shell' marked build as failure
Robot results publisher started...
INFO: Checking test criticality is deprecated and will be dropped in a future release!
-Parsing output xml:
Done!
-Copying log files to build dir:
Done!
-Assigning results to build:
Done!
-Checking thresholds:
Done!
Done publishing Robot results.
Finished: FAILURE

```

Lab Worksheet

Jenkins

localhost:8080/job/UAT/55/robot/

Dashboard > UAT > #55 > Robot result

Status

Changes

Console Output

Edit Build Information

Timings

Git Build Data

Robot Results

Previous Build

Robot Framework Test Results

Executed: 2025-01-29T16:27:46.015298
Duration: 0:00:00.019 (+0:00:00.019)
Status: 1 critical test, 0 passed, 1 failed, 0 skipped
 1 test total (±0), 0 passed, 1 failed, 0 skipped
Results: [report.html](#)
[log.html](#)
[Original result files](#)

Test Result Trend

Number of test cases

Build

☒ Zoom to changes ☐ Show only failed ☐ Show only critical Max builds [Show bigger image](#)

Legend: Skipped (yellow), Passed (green), Failed (red)

Duration Trend

Duration (ms)

Build

[Show bigger image](#)

Failed Test Cases

Name	Crit.	Duration	Age
Valid Login.Valid Login	no	0:00:00.002	1

Test Suites

Name	Failed tests (asm)	Total tests (asm)	Duration (asm)
Valid Login	1 (+1)	1 (+1)	0:00:00.019 (+0:00:00.019)

REST API Jenkins 2.479.3

localhost:8080/job/UAT/55/robot/durationGraph?maxBuildsToShow=0&hd=true