

Lab Worksheet

ชื่อ-นามสกุล _____ พัชริดา เพื่องอารมย์ _____ รหัสนักศึกษา _____ 663380608-3 _____ Section _____ 4 _____

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
2. เปิด Command line หรือ Terminal บน Docker Desktop จากนั้นสร้าง Directory ชื่อ Lab8_1
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_1 เพื่อใช้เป็น Working directory
4. ป้อนคำสั่ง \$ docker pull busybox หรือ \$ sudo docker pull busybox สำหรับกรณีที่ติดปัญหา Permission denied (หมายเหตุ: BusyBox เป็น software suite ที่รองรับคำสั่งบางอย่างบน Unix - <https://busybox.net>)
5. ป้อนคำสั่ง \$ docker images

[Check point#1] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

The screenshot shows the Docker Desktop interface. In the top navigation bar, 'Images' is selected. Below it, there are tabs for 'Local' and 'My Hub'. A search bar and filter icons are present. The main area displays a table with one image entry:

	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	D ⋮ trash

Below the table is a 'Terminal' section with a command-line interface. The terminal history shows:

```

PS D:\663388608-3\3-2\SE\Lab8> mkdir Lab8_1
Directory: D:\663388608-3\3-2\SE\Lab8

Mode          LastWriteTime     Length Name
----          -----        ----- 
d----
```

The user then runs the following commands:

```

PS D:\663388608-3\3-2\SE\Lab8> cd ..\Lab8_1
PS D:\663388608-3\3-2\SE\Lab8\Lab8_1> docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:b3255e7dfbcd10c367af0469747d511eb66dfac50cf30e97e7e4207dd7f6
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
PS D:\663388608-3\3-2\SE\Lab8\Lab8_1> docker images

```

IMAGE	ID	DISK USAGE	CONTENT SIZE	EXTRA
busybox:latest	b3255e7dfbcd	6.77MB	2.229B	Info → In Use

The terminal ends with:

```

PS D:\663388608-3\3-2\SE\Lab8\Lab8_1>
```

- (1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร _____ ชื่อของ Docker images ที่ใช้อ่าน busybox _____
- (2) Tag ที่ใช้บ่งบอกถึงอะไร _____ บอกถึง Version ของ Docker image ซึ่งจากผลลัพธ์ busybox:**latest** ในที่นี้คือ latest = Version ล่าสุด _____
6. ป้อนคำสั่ง \$ docker run busybox
7. ป้อนคำสั่ง \$ docker run -it busybox sh
8. ป้อนคำสั่ง ls
9. ป้อนคำสั่ง ls -la
10. ป้อนคำสั่ง exit
11. ป้อนคำสั่ง \$ docker run busybox echo "Hello ชื่อและนามสกุลของนักศึกษา from busybox"
12. ป้อนคำสั่ง \$ docker ps -a

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

Lab Worksheet

Images Give feedback [🔗](#)

Local My Hub

4.55 MB / 0 Bytes in use 1 images Last refresh: 2 hours ago [⟳](#)

Name	Tag	Image ID	Created	Size	Actions
busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	D ⋮ Delete

Terminal

```

busybox:latest b3255e7dfbcd 6.77MB 2.22MB
PS D:\663388608-3\3-2\SELab\Lab8_1> docker run busybox
root@1aef48...
drwxr-xr-x 1 root root 4096 Feb 4 12:05 .
drwxr-xr-x 1 root root 4096 Feb 4 12:05 ..
-rw-r--r-- 1 root root 0 Feb 4 12:05 .dockervis
drwxr-xr-x 2 root root 12288 Sep 26 2024 bin
drwxr-xr-x 5 root root 360 Feb 4 12:05 dev
drwxr-xr-x 1 root root 4096 Feb 4 12:05 etc
drwxr-xr-x 1 root root 4096 Sep 26 2024 home
drwxr-xr-x 2 root root 4096 Sep 26 2024 lib
drwxr-xr-x 2 root root 3 Sep 26 2024 lib64 -> lib
dr-xr-xr-x 234 root root 0 Feb 4 12:05 proc
drwxr-xr-x 1 root root 4096 Feb 4 12:05 root
dr-xr-xr-x 13 root root 0 Feb 4 12:05 sys
drwxr-xr-x 2 root root 4096 Sep 26 2024 tmp
drwxr-xr-x 4 root root 4096 Sep 26 2024 var
drwxr-xr-x 4 root root 4096 Sep 26 2024 var
/ # exit
PS D:\663388608-3\3-2\SELab\Lab8_1> docker run busybox echo Hello ที่นี่มา เฟื่องฟ้าดี from busybox
Hello ที่นี่มา เฟื่องฟ้าดี from busybox
PS D:\663388608-3\3-2\SELab\Lab8_1> docker ps
CONTAINER ID        COMMAND           CREATED          STATUS          PORTS     NAMES
66e6088309ef      busybox:echo Hello ที่นี่มา... 17 seconds ago   Exited (0) 16 seconds ago   elegant_wozniak
fbfdca0a7a89       busybox:sh      2 minutes ago    Exited (0) About a minute ago   competent_curras
36a32d641b5a      busybox:sh      2 minutes ago    Exited (0) 2 minutes ago    gracious_lamport
PS D:\663388608-3\3-2\SELab\Lab8_1>

```

Containers Give feedback [🔗](#)

Container CPU usage [ⓘ](#) Container memory usage [ⓘ](#) Show charts

No containers are running.

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Net	Actions
gracious_lamport	36a32d641b5a	busybox		N/A	N/A	N/A	N/A	N/A	D ⋮ Delete
competent_curras	fbfdca0a7a89	busybox		N/A	N/A	N/A	N/A	N/A	D ⋮ Delete
elegant_wozniak	66e6088309ef	busybox		N/A	N/A	N/A	N/A	N/A	D ⋮ Delete

Containers / elegant_wozniak

elegant_wozniak 66e6088309ef [🔗](#) busybox:latest

STATUS Exited (0) (2 minutes ago) [⟳](#) [D](#) [⋮](#) [Delete](#)

Logs Inspect Bind mounts Exec Files Stats

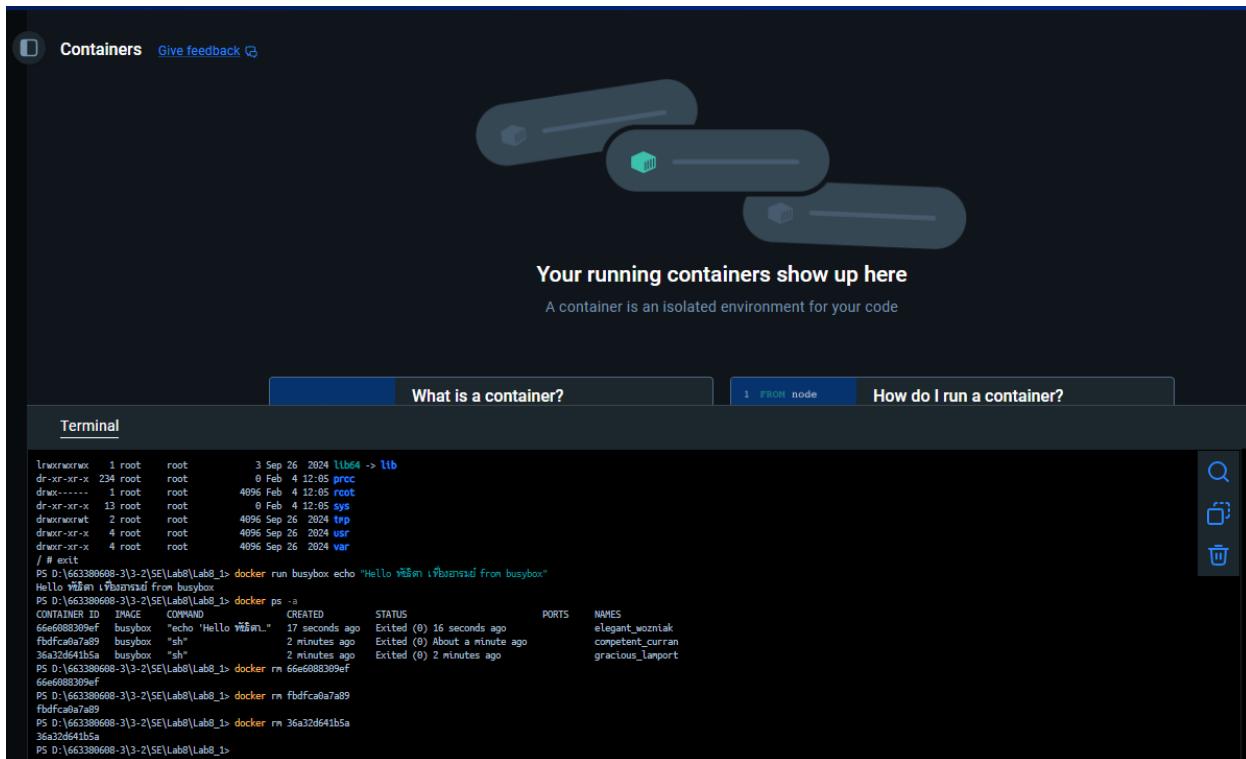
Hello ที่นี่มา เฟื่องฟ้าดี from busybox

- (1) เมื่อใช้ option -it ในคำสั่ง run สำหรับการทำงานของคำสั่งอย่างไรบ้าง จึงสามารถสั่งเช่น `_it` จะทำให้สามารถพิมพ์คำสั่งและตอบโต้กันได้แบบ real time มาจาก -i (interactive) ที่ทำให้ contrainer เปิดตลอดเวลา ทำให้สามารถส่ง input เข้าไปได้ และ -t (-tty) สร้าง terminal จำลองมาเข้ามือกับ input/output ของcontrainer กับ terminal สามารถพิมคำสั่งและเห็นผลลัพธ์ได้ทันที
 - (2) คลิ้ม STATUS จากการรันคำสั่ง `docker ps -a` และดูเงื่อนไขมูล lokale
- สถานะการทำงานของ contrainer ในระบบ โดยผลลัพธ์คือล้ม `STATUS` ที่ได้แสดงว่า `Exited`

13. ป้อนคำสั่ง `$ docker rm <container ID` ที่ต้องการลบ

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

Lab Worksheet



แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดว์ (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

Lab Worksheet

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 พิรบอมกับตอบค่าตามต่อไปนี้

The screenshot shows the Docker desktop application interface. It includes three main sections:

- Terminal:** Shows command-line interactions for building a Docker image. The terminal output is as follows:


```
PS D:\663388608-3\3-2\SE\Lab8> mkdir Lab8_2
Directory: D:\663388608-3\3-2\SE\Lab8

Mode          LastWriteTime     Length Name
----          -----        171 Dockerfile

PS D:\663388608-3\3-2\SE\Lab8> cd ..\Lab8_2
PS D:\663388608-3\3-2\SE\Lab8> dir

Directory: D:\663388608-3\3-2\SE\Lab8\Lab8_2

Mode          LastWriteTime     Length Name
----          -----        171 Dockerfile

PS D:\663388608-3\3-2\SE\Lab8\Lab8_2> docker build -t firstimage_lab8_2 .
[+] Building 0.4s (5/5) FINISHED
 => [internal] load build definition from Dockerfile
 => transferring dockerfile: 210B
 => [internal] load metadata for docker.io/library/busybox:latest
 => [internal] load .dockerignore
 => transferring context: 2B
 => CACHED [1/1] FROM docker.io/library/busybox:latest@sha256:b3255e7dfbcd10cb367af0d409747d511aebe66d4fac98cf30e97e87e4207dd76f
 => exporting to image
 => exporting layers
 => exporting manifest sha256:bb6d3b768033e5b75cc7c9fe165cf9afbfde0850d0b1e3acd5cf2f16251a83
 => exporting config sha256:1c34b613ff0c2cc5d8439a9ab491c496a35a5a0b80ffeb42fcac13088871d2
 => exporting annotation manifest sha256:ad56939c2338365134013a3529fa0d69a0c5c520a0097cae068714ae4021ab
 => exporting manifest list sha256:f4a06b0946f249f40264f197a0c4e297051af0b8cd368f3d6c15187a278aa
 => naming to docker://library/firstimage_lab8_2:latest
 => inspecting to docker://library/firstimage_lab8_2:latest

3 warnings found (use docker --debug to expand):
 - MultipleInstructionNotAllowed: 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)
 - JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)

PS D:\663388608-3\3-2\SE\Lab8> docker run firstimage_lab8_2
Hello from my first Docker image!
```
- Containers:** Shows a single running container named "musing_tesla" with ID "be53795d992b".
- Images:** Shows two images: "busybox" (latest tag, ID: b3255e7dfbcd) and "firstimage_lab8_2" (latest tag, ID: f4a06b0946f2).

Lab Worksheet

(1) คำสั่งที่ใช้ในการ run คือ

docker run <ชื่อ image> _____ ในที่นี่ใช้คำสั่ง docker run firstimage_lab8_2_____

(2) Option -t ในคำสั่ง \$ docker build ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสักเช่น

-t มาจากคำว่า Tag ใช้กำหนดชื่อหรือเวอร์ชันให้กับ Docker ที่เราสร้างขึ้น_____**แบบฝึกปฏิบัติที่ 8.3: การแขร์ 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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการwinโดว์ 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
```

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้


```
$ docker build -t <username>/lab8 .
```
6. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง


```
$ docker run <username>/lab8
```

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

Lab Worksheet

Images

Last refresh: 2 hours ago

Name	Tag	Image ID	Created	Size	Actions
busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	D ⋮ U
firstimage_lab8_2	latest	f4a06b0946f2	1 year ago	6.77 MB	D ⋮ U
phatcharida21/lab8	latest	10a8db4da449	1 year ago	6.77 MB	D ⋮ U

Terminal

```

Directory: D:\663380608-3\3-2\SE\Lab8

Node      LastWriteTime      Length Name
----      -----          ----- 
d-----  2/4/2026  7:24 PM        Lab8_3

PS D:\663380608-3\3-2\SE\Lab8> cd ..\Lab8_3\

PS D:\663380608-3\3-2\SE\Lab8\Lab8_3> dir

Directory: D:\663380608-3\3-2\SE\Lab8\Lab8_3

Mode      LastWriteTime      Length Name
----      -----          ----- 
-a---  2/4/2026  7:25 PM        103 Dockerfile

PS D:\663380608-3\3-2\SE\Lab8\Lab8_3> docker build -t phatcharida21/lab8 .

[*] Building 0.3s (5/5) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 222B
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
=> [internal] load metadata for docker.io/library/busybox:latest
=> [internal] load .dockerignore
=> CACHED [1/1] FROM docker.io/library/busybox:latest@sha256:b3255e7dfbcd10cb367af0d409747d511ae66dfac98cf30e97e87e4207dd76f
=> => resolve docker.io/library/busybox:latest@sha256:b3255e7dfbcd10cb367af0d409747d511ae66dfac90cf30e97e87e4207dd76f
=> => exporting to image
=> => exporting layers
=> => exporting manifest sha256:4c62a6599-6958bf2f05a3186b39ff1753cf41aca9fffb94321fc016be723
=> => exporting config sha256:8ae66674478364760a0ef32907e9831197d33389249a9662055cc45bbae919
=> => exporting attestation manifest sha256:10a8db4da49747d1d159c5be748a8e54ccfb5de33d496378769a
=> => naming to docker.io/phatcharida21/lab8:latest
=> => unpacking to docker.io/phatcharida21/lab8:latest

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)

PS D:\663380608-3\3-2\SE\Lab8\Lab8_3> docker run phatcharida21/lab8

"ต้องการเข้าสู่ระบบ 663380608-3"
PS D:\663380608-3\3-2\SE\Lab8\Lab8_3>

Containers

Container CPU usage ⓘ Container memory usage ⓘ Show charts



No containers are running.



| Name          | Container ID | Image              | Port(s) | CPU (%) | Memory usage...                                       | Memory (%) | Disk | Actions |
|---------------|--------------|--------------------|---------|---------|-------------------------------------------------------|------------|------|---------|
| musing_tesla  | be53795d992b | firstimage_lab8_2  | N/A N/A | N/A N/A | <a href="#">D</a> <a href="#">⋮</a> <a href="#">U</a> |            |      |         |
| eager_lumiere | 11d2f37d997c | phatcharida21/lab8 | N/A N/A | N/A N/A | <a href="#">D</a> <a href="#">⋮</a> <a href="#">U</a> |            |      |         |


```

7. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง

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

ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

\$ docker login และป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง

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

Lab Worksheet

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

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

The screenshot shows the Docker Hub dashboard for the user 'phatcharida21'. The left sidebar includes options like Repositories, Hardened Images, Collaborations, Settings, Default privacy, Notifications, Billing, Usage, Pulls, and Storage. The main area has tabs for 'Repositories' and 'Images'.

Repositories:

- Search bar: Search by repository name
- Filter: All content
- Create a repository button
- Table headers: Name, Last Pushed, Contains, Visibility, Scout
- Row: phatcharida21/lab8, 3 minutes ago, IMAGE, Public, Inactive
- Pagination: 1-1 of 1

Images:

- Local tab is selected, My Hub tab is available.
- 4.55 MB / 0 Bytes in use, 3 images.
- Last refresh: 2 hours ago.
- Search bar.
- Table headers: Name, Tag, Image ID, Created, Size, Actions.
- Rows:
 - busybox, latest, b3255e7dfbcd, 1 year ago, 6.77 MB, actions
 - firstimage_lab8_2, latest, f4a06b0946f2, 1 year ago, 6.77 MB, actions
 - phatcharida21/lab8, latest, 10a8db4da449, 1 year ago, 6.77 MB, actions

Terminal:

```

Run `docker -help` for more information.
PS D:\663380698-3\3-2\SEL\Lab8\Lab8_3> docker run phatcharida21/lab8
"忙しい" 7b3b794d-e633-3868-8-3
PS D:\663380698-3\3-2\SEL\Lab8\Lab8_3> docker push phatcharida21/lab8
Using default tag: latest
The push refers to repository [docker.io/phatcharida21/lab8]
61df5b572fc5: Layer already exists
7072c3eab1b1: Pushed
latest: digest: sha256:9b8d4da4497471d159c5be748a4e00a4739a8e54ccfbddde33da49b378769a size: 855
PS D:\663380698-3\3-2\SEL\Lab8\Lab8_3>
  
```

Containers:

- Container CPU usage: No containers are running.
- Container memory usage: No containers are running.
- Show charts button.
- Search bar.
- Only show running containers checkbox.
- Table headers: Name, Container ID, Image, Port(s), CPU (%), Memory usage..., Memory (%), Disk, Actions.
- Rows:
 - musing_tesla, be53795d992b, firstimage_lab8_2, N/A, N/A, N/A, N/A, actions
 - eager_lumiere, 11d2f37d997c, phatcharida21/lab8, N/A, N/A, N/A, N/A, actions

แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4

Lab Worksheet

2. ทำการ Clone ซอฟต์แวร์โค้ดของเว็บแอปพลิเคชันจาก 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 เมื่อพื้นที่ package.json ให้ใช้ Text editor ในการเปิดอ่าน

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

The terminal window shows the command \$ git clone https://github.com/docker/getting-started.git being run and completed successfully.

The code editor window shows the package.json file content:

```

{
  "name": "101-app",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "scripts": {
    "prettify": "prettier -l --write \"**/*.js\"",
    "test": "jest",
    "dev": "nodemon src/index.js"
  },
  "dependencies": {
    "express": "^4.18.2",
    "mysql2": "^2.3.3",
    "sqlite": "^5.1.2",
    "uuid": "9.0.0",
    "wait-port": "^1.0.4"
  },
  "resolutions": {
    "ansi-regex": "5.0.1"
  },
  "prettier": {
    "trailingComma": "all",
    "tabWidth": 4,
    "useTabs": false,
    "semi": true,
    "singleQuote": true
  },
  "devDependencies": {
    "jest": "^29.3.1",
    "nodemon": "^2.0.20",
    "prettier": "^2.7.1"
  }
}

```

4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไว้ในไฟล์

```
FROM node:18-alpine
```

```
WORKDIR /app
```

```
COPY . .
```

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

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

```
EXPOSE 3000
```

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดให้ชื่อ image เป็น myapp_รหัสศ. ไม่มีชีด

```
$ docker build -t <myapp_รหัสศ. ไม่มีชีด> .
```

Lab Worksheet

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

```

PS D:\663380608-3\3-2\SE\Lab8_4\getting-started\app> docker build -t myapp_663380608 .
[*] Building 4.6s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 156B
=> [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@sha256:8d6421d663b4c28fd3ebc498332f249011d1189455880a35cb9bc4b8ca09d9e
=> => resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d1189455880a35cb9bc4b8ca09d9e
=> [internal] load build context
=> => transferring context: 2.49B
=> CACHED [2/4] WORKDIR /app
=> CACHED [3/4] COPY . .
=> CACHED [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:bc47045db128fcad3df124793397f9660b556ed5fa6fb76301af72a937d4cc8
=> => exporting config sha256:f64f4c76a3f9325fe9dbc87620af8364dd1d2daaa6d1285897ea7f739204d
=> => exporting attestation manifest sha256:cfe4d2f3fc782fcf757d256991a4feb0eabbfb0a7c23a4ae4f7c3d38d4a7ea
=> => exporting manifest list sha256:92fd1dcbe12a0bb4a424deeb4797400104cf5affef72a2366f6485830a2304
=> => naming to docker.io/myapp_myapp_663380608:latest
=> => unpacking to docker.io/library/myapp_myapp_663380608:latest
PS D:\663380608-3\3-2\SE\Lab8_4\getting-started\app>
  
```

The screenshot shows a Windows terminal window at the bottom displaying the command `docker build -t myapp_663380608 .` and its output. Above the terminal is a code editor window showing the file structure of a Node.js application. The `Dockerfile` tab is active, containing the following content:

```

FROM node:18-alpine
WORKDIR /app
COPY . .
RUN yarn install --production
CMD ["node", "src/index.js"]
EXPOSE 3000
  
```

The left sidebar of the code editor shows the project structure with files like `package.json`, `yarn.lock`, and `Dockerfile` highlighted.

- ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

Lab Worksheet

\$ docker run -dp 3000:3000 <myapp_รหัสนศ. ไม่มีชีด>

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

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

The screenshot shows the Docker Desktop application window. The top navigation bar has tabs for 'Images' (selected), 'Give feedback', 'Local' (selected), and 'My Hub'. Below the navigation is a search bar and a refresh button. The main area displays a table of images:

	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	firstimage_lab8_2	latest	f4a06b0946f2	1 year ago	6.77 MB	<input type="button"/> <input type="button"/> <input type="button"/>
<input type="checkbox"/>	phatcharida21/lab8	latest	10a8db4da449	1 year ago	6.77 MB	<input type="button"/> <input type="button"/> <input type="button"/>
<input type="checkbox"/>	busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	<input type="button"/> <input type="button"/> <input type="button"/>
<input type="checkbox"/>	myapp_6633806083	latest	92df1dbce12a	5 minutes ago	342.03 MB	<input type="button"/> <input type="button"/> <input type="button"/>

Below the table is a terminal window showing the build process of the 'myapp_6633806083' image:

```
= [Internal] load build context
=> transferring context: 2.49kB
=> COHED [3/4] COPY . .
=> COHED [3/4] RUN yarn install --production
=> exporting layers
=> exporting manifest sha256:fc64fc7c6a3f9325fc9d8d3f24793397f9660b556ed5faefb7630af772a037d4cc8
=> exporting config sha256:f64fc7c6a3f9325fc9d8d3f24793397f9660b556ed5faefb7630af772a037d4cc8
=> exporting attestation manifest sha256:cfdcf73fd782fcf757d25699191f4fb6ebabf0a7c23a4aeaf7c3d38d4a7ea
=> exporting manifest list sha256:92fd1dbce12a04b4a24deebfa977458194cf5afffe72a2366f485838da7364
=> naming to docker.io/library/myapp_6633806083:latest
=> pushing to docker.io/library/myapp_6633806083:latest
PS D:\[6633806083]-[3]-[2]\SE\Lab8\Lab8_4\getting-started\apps> docker run -dp 3000:3000 myapp_6633806083
049c84b13f786aecf748a8f499318c98d3e3ef40f1c6690ca32372f5b329318
PS D:\[6633806083]-[3]-[2]\SE\Lab8\Lab8_4\getting-started\apps> [
```

The 'Containers' section shows the running containers:

	Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Dis	Actions
<input type="checkbox"/>	eager_lumiere	11d2f37d997c	phatcharida21/lab8		0%	0B / 0B	0%	0B	<input type="button"/> <input type="button"/> <input type="button"/>
<input type="checkbox"/>	musing_tesla	be53795d992b	firstimage_lab8_2		0%	0B / 0B	0%	0B	<input type="button"/> <input type="button"/> <input type="button"/>
<input checked="" type="checkbox"/>	gifted_moser	049c84b13f57	myapp_6633806083	3000:3000	0%	19.91MB / 3.73G	0.52%	0B	<input type="button"/> <input type="button"/> <input type="button"/>

At the bottom, there is a browser window showing a simple web application with a list of items: 'Book' and 'Pen'.

หมายเหตุ: นศ.สามารถทดลองเล่น Web application ที่ทำงานอยู่ได้

- ทำการแก้ไข Source code ของ Web application ดังนี้

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

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

Lab Worksheet

<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

[Check point#10] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เปิดอยู่) และคงไว้ในรูปภาพ

```
Terminal

PS D:\gg63380608-3\3-2\SE\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6633806083 .

[*] Building 4.6s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 156B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [internal] load .dockerignore
=> [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
=> exporting manifest sha256:b47045bd120fcad3df124793397f9660b556ed5fa6fb76301af72a937d4cc8
[*] Building 28.3s (19/19) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 156B
=> [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: 28
=> [4/4] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9cb4b8ca09d9e
=> => resolving docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9cb4b8ca09d9e
=> [internal] load build context
=> => transferring context: 8.1KB
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY .
=> [4/4] RUN yarn install --production
=> exporting to image
=> exporting layers
=> exporting manifest sha256:11b28c5591eaed7720c35a5e053b550ca9f6e0820a722109efae7ad834599
=> => exporting config sha256:62bc3dec4ff803e5551d16bd8e855bb963d6bb3bf54dad6704798c49f7d
=> => exporting attestation manifest sha256:9b70e294aae39f37868069420942561a935ee91d76099a2d3695642bf56fb
```

```
PS D:\663380608-3\3-2\SE\Lab0_4\getting-started\app> docker run -dp 3000:3000 myapp_6633806083  
32796153b6fe0fb1bb59f76deed0b1aa9a7fa5cc85434e3c7fb155aa637  
docker: Error response from daemon: failed to set up container networking: driver failed programming external connectivity on endpoint jovial_euclid (415c96fbad62d2b9e6a0ae8bf0f4a4a638aa877bd88fc07295884edd06974c1bf): Bind for 0.0.0.0:3000 failed: port is already allocated
```

- (1) Error ที่เกิดขึ้นหมายความอย่างไร และเกิดขึ้นเพราะอะไร

พอร์ต 3000 กำลังถูกใช้งานอยู่ เมื่อจากการไข้ พอร์ต 3000 โดยยังไม่ได้หยุดการทำงานของโปรแกรมที่เคยใช้ก่อนหน้า

11. ลง Container ของ Web application เว็บร์ชันก่อนแก้ไขออกจากระบบ โดยใช้วิธีไดร์ฟหนึ่งดังต่อไปนี้

- ### 3.1.1 Command Line interface

- i. ใช้คำสั่ง `$ docker ps` เพื่อดู Container ID ที่ต้องการจะลบ
 - ii. Copy หรือบันทึก Container ID ไว้
 - iii. ใช้คำสั่ง `$ docker stop <Container ID ที่ต้องการจะลบ>` เพื่อยุดการทำงานของ Container ดังกล่าว
 - iv. ใช้คำสั่ง `$ docker rm <Container ID ที่ต้องการจะลบ>` เพื่อทำการลบ

- b. ផ្ទាល់នា Docker desktop

- i. ไปที่หน้าต่าง Containers
 - ii. เลือกไอคอนถังขยะในแท็บของ Container ที่ต้องการจะลบ
 - iii. ปุ่มยืนยันการลบ Delete forever

- 12 Start และรัน Container ตัวใหม่คือครั้ง ໂດຍให้คำสั่งเดิมกันก้าวข้าม

13. ໃນ Browser 1 ທີ່ URL = <http://localhost:3000>

Lab Worksheet

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

```
PS D:\663380608-3\3-2\SE\Lab8\Lab8_4> docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
049c84b13f57 92af1dbce12a "docker-entrypoint.s..." 40 minutes ago Up 18 minutes 0.0.0.0:3000->3000/tcp, [:]:3000->3000/tcp gifted_moser
PS D:\663380608-3\3-2\SE\Lab8\Lab8_4> docker stop 049c84b13f57
049c84b13f57
PS D:\663380608-3\3-2\SE\Lab8\Lab8_4> docker rm 049c84b13f57
049c84b13f57
PS D:\663380608-3\3-2\SE\Lab8\Lab8_4> docker run -dp 3000:3000 myapp_6633806083 d79d7e5a7ddad28698a47fbab5d586b38c208381bc6f69dd93ea20b2cf18c92
PS D:\663380608-3\3-2\SE\Lab8\Lab8_4> []

Containers Give feedback ⓘ
```

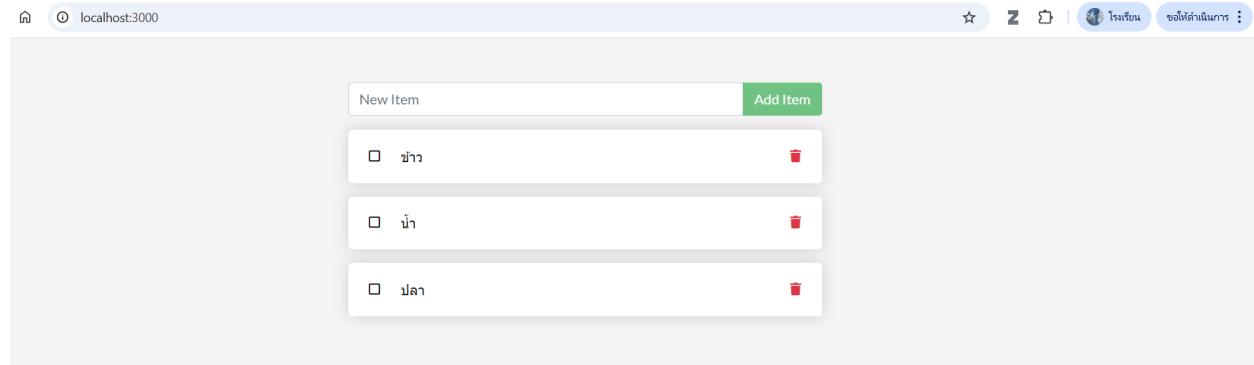
Container CPU usage ⓘ 0.00% / 400% (4 CPUs available)

Container memory usage ⓘ 18.68MB / 3.64GB

Show charts

	Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Dis	Actions
<input type="checkbox"/>	eager_lumiere	11d2f37d997c	phatcharida21/lab8		0%	0B / 0B	0%	0B	▶ ⋮ trash
<input type="checkbox"/>	musing_tesla	be53795d992b	firstimage_lab8_2		0%	0B / 0B	0%	0B	▶ ⋮ trash
<input type="checkbox"/>	jovial_euclid	327d96153b0e	myapp_6633806083	3000:3000	0%	0B / 0B	0%	0B	▶ ⋮ trash
<input checked="" type="checkbox"/>	condescending_h	d79d67e5a7dd	myapp_6633806083	3000:3000	0%	18.68MB / 3.73G	0.49%	6.7	stop ⋮ trash

localhost:3000 [▶](#) [Z](#) [refresh](#) [ໂຄສະນາ](#) [ຂອ້ເດືອນການ](#) :



แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

1. สร้าง Dockerfile เพื่อสร้าง Jenkins และ Environment ที่เหมาะสมกับการรัน Robot framework ใน Container

[Check point#12] ส่ง Dockerfile ที่ได้คำสั่งที่เกี่ยวข้องไว้

```
FROM jenkins/jenkins:lts
USER root
RUN apt-get update && apt-get install -y python3 python3-pip python3-venv
RUN pip3 install robotframework --break-system-packages
USER jenkins
```

Lab Worksheet

```
PS D:\6633890688-3\3-2\SE\Lab8> docker build -t jenkins-robot-local .
[+] Building 26.5s (7/7) FINISHED
--> [internal] load build definition from Dockerfile
--> transferring dockerfile: 221B
--> [internal] load metadata for docker.io/jenkins:lts
--> [internal] load .dockerignore
--> transferring context: 2B
--> [1/1] FROM docker.io/jenkins:lts@sha256:d1ea795c5facd7f549a21c40e5e43ffcc5fb5f48683d9b24750f26e8079d772
--> resolving docker.io/jenkins:lts@sha256:d1ea795c5facd7f549a21c40e5e43ffcc5fb5f48683d9b24750f26e8079d772
--> CACHED [2/3] RUN apt-get update && apt-get install -y python3 python3-pip python3-vn
--> [3/3] RUN pip install robotframework --break-system-packages
--> exporting to image
--> exporting layers
--> exporting manifest sha256:2fdb0f83b7395345c1536bd17dcb16fd274497fc8767a1937a1d4ae394b7aacf
--> exporting config sha256:f09f9fa32b161a57783532ad488ae29a23f8fad47e5cd44da4207bb0d58
--> exporting attestation manifest sha256:cdd4b7052f9ea31ac1929f5db738965958d05785e0f9a581b21633a2c5d7d
--> exporting manifest list sha256:24907dd9da176d129abe1f1b471af27df2ab7cb2135eaa93e8da9c78b61f96
--> naming to docker.io/library/jenkins-robot-local:latest
--> unpacking to docker.io/library/jenkins-robot-local:latest
PS D:\6633890688-3\3-2\SE\Lab8>
```

2. เปิด Command line หรือ Terminal บน Docker Desktop
3. Build image ที่สร้างขึ้นในข้อที่ 1 พร้อมกับตั้งชื่อของ image เป็น jenkins-robot-local

\$ docker build -t jenkins-robot-local .

4. รัน container โดยผูกพอร์ตให้เรียบร้อย เช่น

```
$ docker run -d \
  --name jenkins-robot \
  -p 8080:8080 -p 50000:50000 \
  -v jenkins_home:/var/jenkins_home \
  -v /var/run/docker.sock:/var/run/docker.sock \
  jenkins-robot-local
```

หรือ

\$ docker run -it --name jenkins-debug -p 8080:8080 -v /var/run/docker.sock:/var/run/docker.sock jenkins-robot-local

[Check point#13] Capture หน้าจอที่แสดงผล Admin password

```
Terminal
2026-02-04 14:44:03.066+0000 [id=33] INFO jenkins.model.Jenkins<init>: Starting version 2.541.1
2026-02-04 14:44:03.224+0000 [id=42] INFO jenkins.InitReactorRunner$1#onAttained: Started initialization
2026-02-04 14:44:03.255+0000 [id=43] INFO jenkins.InitReactorRunner$1#onAttained: Listed all plugins
2026-02-04 14:44:04.270+0000 [id=44] INFO jenkins.InitReactorRunner$1#onAttained: Prepared all plugins
2026-02-04 14:44:04.276+0000 [id=44] INFO jenkins.InitReactorRunner$1#onAttained: Started all plugins
2026-02-04 14:44:04.285+0000 [id=44] INFO jenkins.InitReactorRunner$1#onAttained: Augmented all extensions
2026-02-04 14:44:04.682+0000 [id=41] INFO jenkins.InitReactorRunner$1#onAttained: System config loaded
2026-02-04 14:44:04.683+0000 [id=41] INFO jenkins.InitReactorRunner$1#onAttained: System config adapted
2026-02-04 14:44:04.683+0000 [id=41] INFO jenkins.InitReactorRunner$1#onAttained: Loaded all jobs
2026-02-04 14:44:04.688+0000 [id=41] INFO jenkins.InitReactorRunner$1#onAttained: Configuration for all jobs updated
2026-02-04 14:44:04.845+0000 [id=41] INFO hudson.util.Retrier$1#onRetried: Attempt #1 to do the action check updates server
2026-02-04 14:44:05.558+0000 [id=41] INFO jenkins.install.SetupWizard$init:
[LF]
[LF] ****
[LF] Jenkins initial setup is required. An admin user has been created and a password generated.
[LF] Please use the following password to proceed to installation:
[LF] 73f8e5774c5d4b919b0d6984b15ffef
[LF]
[LF] This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
[LF]
[LF] ****
[LF] ****
[LF] ****
2026-02-04 14:44:15.524+0000 [id=41] INFO jenkins.InitReactorRunner$1#onAttained: Completed initialization
2026-02-04 14:44:15.629+0000 [id=33] INFO hudson.lifecycle.LifeCycle$1#onReady: Jenkins is fully up and running
2026-02-04 14:44:17.743+0000 [id=61] INFO h.m.DownloadService$Downloadable$Load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2026-02-04 14:44:17.747+0000 [id=61] INFO hudson.util.Retrier$1#onStart: Performed the action check updates server successfully at the attempt #1
```

5. บันทึกรหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก
6. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดเบราว์เซอร์ และป้อนที่อยู่เป็น http://localhost:8080
7. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้เขียนข้อที่ 3
8. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

[Check point#14] Capture หน้าจอที่แสดงผลการตั้งค่า

Lab Worksheet

The screenshot shows a web browser window with the URL `localhost:8080` in the address bar. The page title is "Getting Started" and the main heading is "Create First Admin User". The form contains the following fields:

- Username: Phatcharida_6083
- Password: (Redacted)
- Confirm password: (Redacted)
- Full name: Phatcharida Fueangarrom
- E-mail address: phatcharida.f@kkumail.com

At the bottom left, it says "Jenkins 2.541.1". On the right, there are two buttons: "Skip and continue as admin" and "Save and Continue".

- เมื่อติดตั้งเรียบร้อยแล้วจะพบหน้า Dashboard ดังแสดงในภาพ

Lab Worksheet

The screenshot shows the Jenkins homepage at localhost:8080. The top navigation bar includes links for 'New Item', 'Build History', 'Add description', and user status. The main content area features a 'Welcome to Jenkins!' message with a brief description of its purpose. Below this are sections for 'Start building your software project' (with 'Create a job' and a '+' button), 'Set up a distributed build' (with 'Set up an agent', 'Configure a cloud', and a 'Learn more about distributed builds' link), and monitoring sections for 'Build Queue' (empty) and 'Build Executor Status' (0/2). The bottom right corner displays 'REST API' and 'Jenkins 2.541.1'.

10. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

The screenshot shows the 'Manage Jenkins' page at localhost:8080/manage/. The left sidebar has a 'System Configuration' section with links for System, Nodes, Security, and Users. The main content area is titled 'Plugins' and contains a brief description of its function. Below this are sections for 'Clouds', 'Appearance', 'Credentials', and 'Credential Providers'. The bottom section is 'Status Information' with links for System Information, System Log, and Load Statistics.

11. ไปที่เมนู Plugins > Available plugins และเลือกติดตั้ง Robot Framework เพิ่มเติม

Lab Worksheet

Jenkins / Manage Jenkins / Plugins

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Download progress

Robot

Install

Install	Name	Released	Health
<input type="checkbox"/>	Robot Framework 6.2.0	Build Reports	3 mo 15 days ago

This publisher stores Robot Framework test reports for builds and shows summaries of them in project and build views along with trend graph.

91

12. เมื่อติดตั้งสำเร็จจะพบรายการ Plugins ทั้งหมดที่ถูกติดตั้ง ถ้าติดตั้งสำเร็จให้เลือก “Restart Jenkins...” และกด Go back to the top page

Jenkins / Manage Jenkins / Plugins

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Download progress

Pipeline: GitHub Groovy Libraries	<input checked="" type="checkbox"/>	Success
Metrics	<input checked="" type="checkbox"/>	Success
Pipeline Graph View	<input checked="" type="checkbox"/>	Success
Git	<input checked="" type="checkbox"/>	Success
EDDSA API	<input checked="" type="checkbox"/>	Success
Trilead API	<input checked="" type="checkbox"/>	Success
SSH Build Agents	<input checked="" type="checkbox"/>	Success
Matrix Authorization Strategy	<input checked="" type="checkbox"/>	Success
LDAP	<input checked="" type="checkbox"/>	Success
jsoup API	<input checked="" type="checkbox"/>	Success
Email Extension	<input checked="" type="checkbox"/>	Success
Mailer	<input checked="" type="checkbox"/>	Success
Theme Manager	<input checked="" type="checkbox"/>	Success
Dark Theme	<input checked="" type="checkbox"/>	Success
Loading plugin extensions	<input checked="" type="checkbox"/>	Success
Robot Framework	<input checked="" type="checkbox"/>	Success
Loading plugin extensions	<input checked="" type="checkbox"/>	Success

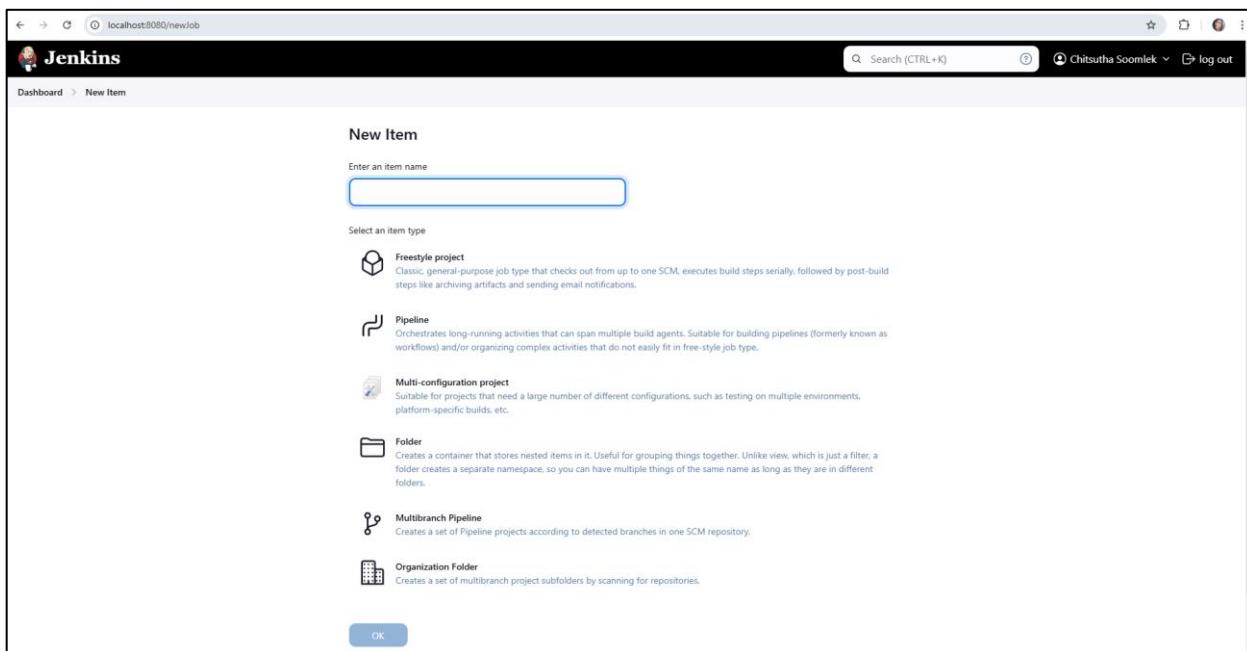
→ Go back to the top page
(you can start using the installed plugins right away)

→ Restart Jenkins when installation is complete and no jobs are running

REST API Jenkins 2.541.1

13. สร้างไฟล์ Jenkinsfile ไม่มีนามสกุล เพื่อ execute คำสั่งต่าง ๆ กับ built-in agent และเอาไฟล์ดังกล่าว เก็บไว้ที่ root ของ GitHub Repository ของนักศึกษา
14. สร้าง folder ชื่อ tests/ บน GitHub Repository ของนักศึกษา และสร้างไฟล์ Lab8.robot และนำไฟล์ไปไว้ folder ที่สร้าง
15. กลับไปที่หน้า Dashboard ของ Jenkins และสร้าง Pipeline อย่างง่าย โดยกำหนด New item เป็น Freestyle project และตั้งชื่อเป็น UAT

Lab Worksheet



16. ตั้งค่าที่จำเป็นในหน้านี้ทั้งหมด ดังนี้

Description: Lab 8.5

GitHub project: กดเดิอก แล้วใส่ Project URL เป็น repository ที่เก็บโค้ด .robot (ดูขั้นตอนที่ 14)

Build Trigger: เลือกแบบ Build periodically และกำหนดให้ build ทุก 15 นาที

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

[Check point#15] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามดังนี้

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

robot Lab8/test/Lab8.robot

Lab Worksheet

General Enabled

Description
 Lab 8.5

Plain text [Preview](#)

Discard old builds [?](#)

GitHub project

Project url [?](#)
https://github.com/Phatcharida21/CP353004-663380608-3/blob/bc69a6e0a28198e777fd12a1b54f3987525141fe/Lab8/test/Lab8.robot/

Advanced [▼](#)

This project is parameterized [?](#)

Throttle builds [?](#)

Execute concurrent builds if necessary [?](#)

Advanced [▼](#)

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None

Git [?](#)

Repositories [?](#)

Repository URL ?	<input type="text"/> https://github.com/Phatcharida21/CP353004-663380608-3/	X
Credentials ?	<input type="text"/> - none -	▼
Advanced ▼		+ Add

+ Add Repository

Branches to build [?](#)

Branch Specifier (blank for 'any') ?	<input type="text"/> */main	X
+ Add Branch		

Lab Worksheet

Repository browser ?
(Auto) ▾

Additional Behaviours
+ Add

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
Schedule ?
H/15 * * * *

Would last have run at Wednesday, February 4, 2026, 4:05:00 PM Coordinated Universal Time; would next run at Wednesday, February 4, 2026, 4:20:00 PM Coordinated Universal Time.

GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Execute shell ? ✖

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

```
robot Lab8/test/Lab8.robot
```

Advanced ▾

+ Add build step

Lab Worksheet

The screenshot shows the Jenkins configuration for publishing Robot Framework test results. It includes fields for the directory of Robot output, archive directory name, output XML name, report HTML name, log HTML name, other files to copy, disable archiving, enable cache, X-axis label, use Artifact Manager, and thresholds for build results.

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

Name of archive directory: Name of archive directory where to store builds. Set to archive to use jenkins build archive directory. Value: robot-test-SE-Lab8

Output xml name: Name of the xml file containing robot output. Value: output.xml

Report html name: Name of the html file containing robot test report. Value: report.html

Log html name: Name of the html file containing detailed robot test log. Value: log.html

Other files to copy: Comma separated list of robot related artifacts to be saved. Value: (empty)

Disable Archiving of output xml file to server: Disable archiving output xml

Enable cache for test results: Enable cache

X-axis label: Overwrite default x-axis label for publish trend. You can use \$display_name to change the label for the build display name. Value: (empty)

Use Artifact Manager to copy tests results: Use Artifact Manager

Thresholds for build result:

- % Value: 20.0
- % Value: 80.0

Include skipped tests in total count for thresholds

Post-build action: เพิ่ม Publish Robot Framework test results -> ระบุไดร์รคทอรีที่เก็บไฟล์ผลการทดสอบโดย Robot framework ในรูป xml และ html -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ไม่ผ่านแล้วนับว่าซอฟต์แวร์มีปัญหา -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ผ่านได้ แล้วบันทึกซอฟต์แวร์มีอยู่ในสถานะที่สามารถนำไปใช้งานได้ (เช่น 20, 80)

17. กด Apply และ Save

18. ล็อก Build Now

[Check point#16] Capture หน้าจอแสดงหน้าหลักของ Pipeline และ Console Output

Lab Worksheet

Jenkins / UAT / #16

#16 (4 ก.พ. 2569 16:37:45)

Started by timer

This run spent:

- 2 min 14 sec waiting;
- 10 sec build duration;
- 2 min 24 sec total from scheduled to completion.

Revision: b03980fb2cba93f945e1075aa624366e7295bfb2
Repository: <https://github.com/Phatcharida21/CP353004-663380608-3/>

refs/remotes/origin/main

Robot Test Summary:

	Total	Failed	Passed	Skipped	Pass %
All tests	2	0	2	0	100.0

- [Browse results](#)
- [Open report.html](#)
- [Open log.html](#)

</> No changes.

Jenkins / UAT / #16 / Robot result

Robot Framework Test Results

Executed: 2026-02-04T16:37:50.661349
Duration: 0:00:0.225 (+0:00:0.175)
Status: 2 test total (±0), 2 passed, 0 failed, 0 skipped
Results: [report.html](#) [log.html](#) [Original result files](#)

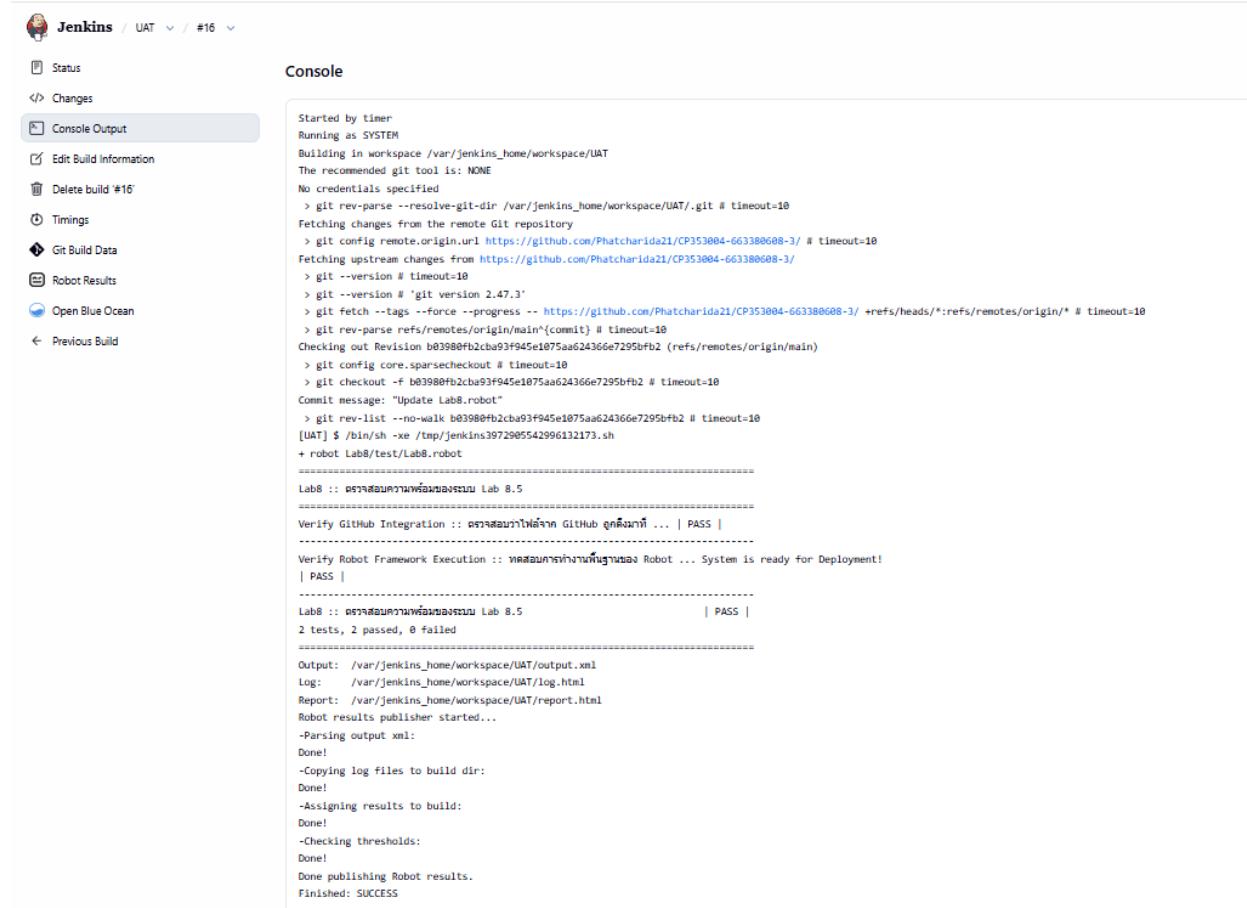
Test Result Trend

Duration Trend

Test Suites

Name	Failed tests	Total tests	Duration
Lab8	0 (±0)	2 (±0)	0:00:0.225 (+0:00:0.175)

Lab Worksheet



The screenshot shows the Jenkins interface for a build named "UAT" (Build #16). The "Console" tab is selected, displaying the command-line log of the build process. The log shows the following steps:

- Started by timer
- 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/Phatcharida21/CP353004-663380608-3/ # timeout=10
- Fetching upstream changes from https://github.com/Phatcharida21/CP353004-663380608-3/
- git --version # timeout=10
- git -version # 'git version 2.47.3'
- git fetch --tags --force --progress -- https://github.com/Phatcharida21/CP353004-663380608-3/ +refs/heads/*:refs/remotes/origin/* # timeout=10
- git rev-parse refs/remotes/origin/main^{commit} # timeout=10
- Checking out Revision b03980fb2cba93f945e1075aa624366e7295fb2 (refs/remotes/origin/main)
- git config core.sparsecheckout # timeout=10
- git checkout -f b03980fb2cba93f945e1075aa624366e7295fb2 # timeout=10
- Commit message: "Update Lab8.robot"
- git rev-list -no-walk b03980fb2cba93f945e1075aa624366e7295fb2 # timeout=10
- [UAT] \$ /bin/sh -xe /tmp/Jenkins3972985542996132173.sh
- + robot Lab8/test/Lab8.robot
- =====
 - Lab8 :: ทดสอบความพร้อมของระบบ Lab 8.5
=====
- Verify GitHub Integration :: ทดสอบ能否将 GitHub 連接到 ... | PASS |
- =====
 - Verify Robot Framework Execution :: ทดสอบการทำงานเบื้องต้นของ Robot ... System is ready for Deployment!
=====
- | PASS |
- =====
 - Lab8 :: ทดสอบความพร้อมของระบบ Lab 8.5 | PASS |
=====
- 2 tests, 2 passed, 0 failed
- =====
 - Output: /var/jenkins_home/workspace/UAT/output.xml
 - Log: /var/jenkins_home/workspace/UAT/log.html
 - Report: /var/jenkins_home/workspace/UAT/report.html
 - Robot results publisher started...
 - Parsing output xml:
 - Done!
 - Copying log files to build dir:
 - Done!
 - Assigning results to build:
 - Done!
 - Checking thresholds:
 - Done!
 - Done publishing Robot results.
 - Finished: SUCCESS
=====