

Lab Worksheet

ชื่อ-นามสกุล _____ พัชรดา 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 ป้อนค่าของพารามิเตอร์ที่กำหนด

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 สำหรับกรณีที่เกิดปัญหา Permission denied (หมายเหตุ: BusyBox เป็น software suite ที่รองรับคำสั่งบางอย่างบน Unix - <https://busybox.net>)
5. ป้อนคำสั่ง \$ docker images

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

Lab Worksheet

The screenshot shows the Docker Desktop 'Images' tab. It displays a table with one image: 'busybox' with tag 'latest', image ID 'b3255e7dfbcd', created '1 year ago', and size '6.77 MB'. Below the table is a 'Terminal' window showing a PowerShell session. The terminal output includes commands like 'mkdir Lab8_1', 'cd .\Lab8_1', 'docker pull busybox', and 'docker images', along with their respective outputs and status messages.

Name	Tag	Image ID	Created	Size	Actions
busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	[Play] [More] [Delete]

```

PS D:\663380608-3\3-2\SE\Lab8> mkdir Lab8_1

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

Mode                LastWriteTime         Length Name
----                -
d-----          2/4/2026   7:03 PM             Lab8_1

PS D:\663380608-3\3-2\SE\Lab8> cd .\Lab8_1\
PS D:\663380608-3\3-2\SE\Lab8\Lab8_1> docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:b3255e7dfbcd0c367af0d40974d511ab66dfac9bcf30e97e7e4207d476f
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
PS D:\663380608-3\3-2\SE\Lab8\Lab8_1> docker images

IMAGE          ID                DISK USAGE    CONTENT SIZE    EXTRA
-----
busybox:latest  b3255e7dfbcd      6.77MB        2.22MB
PS D:\663380608-3\3-2\SE\Lab8\Lab8_1>

```

- (1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร ____ ชื่อของ Docker images ที่ชื่อว่า 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

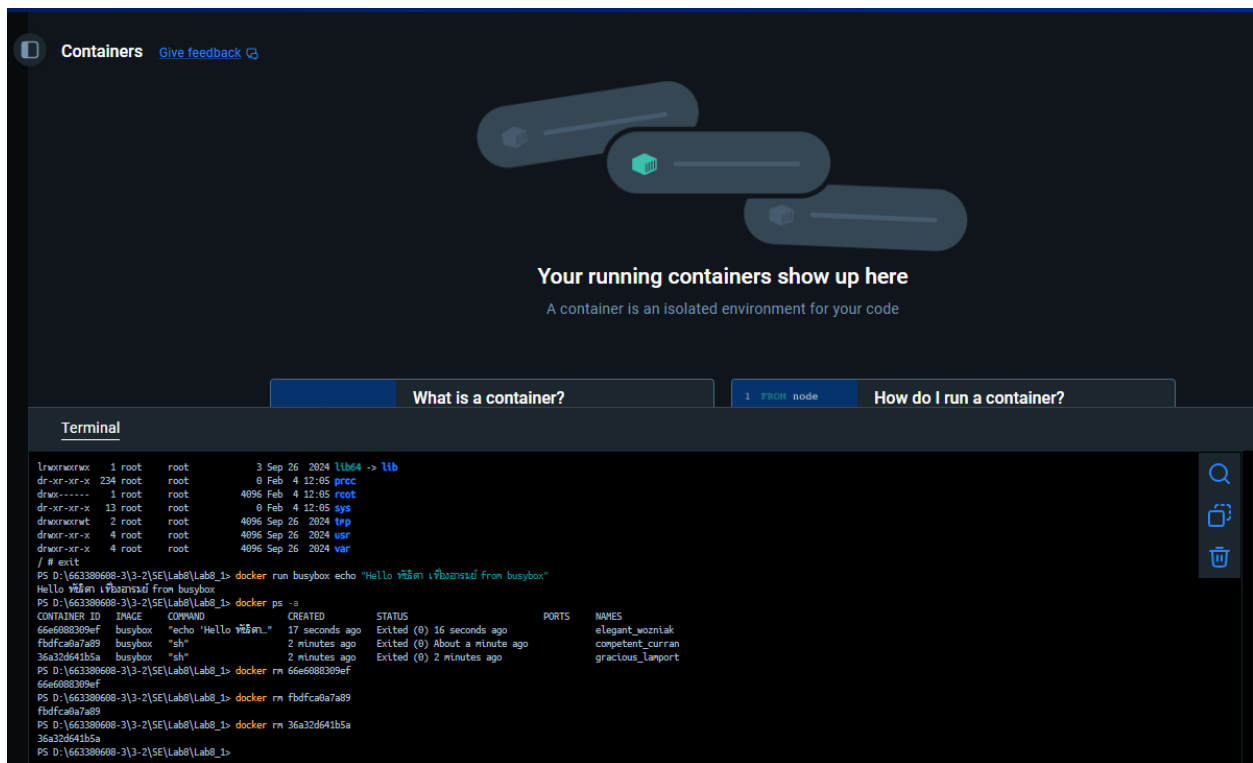
The screenshot shows the Docker Desktop interface. The top section is 'Images', showing a list of images with columns: Name, Tag, Image ID, Created, Size, and Actions. One image, 'busybox', is listed with tag 'latest' and image ID 'b3255e7dfbcd'. Below this is the 'Terminal' section, which displays a series of commands and their outputs, including 'docker run busybox', 'docker ps -a', and 'docker run busybox echo "Hello พิธีกร เป็นภรรยา from busybox"'. The bottom section is 'Containers', showing a table of running containers with columns: Name, Container ID, Image, Port(s), CPU (%), Memory usage..., Memory (%), Disk read/write, and Actions. Three containers are listed: 'gracious_lampo', 'competent_curr', and 'elegant_wozniak'. The 'elegant_wozniak' container is selected, and its details are shown below the table, including its status 'Exited (0) (2 minutes ago)' and various control buttons.

- (1) เมื่อใช้ option `-it` ในคำสั่ง `run` ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
`-it` จะทำให้สามารถพิมพ์คำสั่งและตอบโต้กันได้แบบ real time มาจาก `-i` (interactive) ที่ทำให้ container เปิดตลอดเวลา ทำให้สามารถส่ง input เข้าไปได้ และ `-t` (`-tty`) สร้าง terminal จำลองมาเชื่อมต่อกับ input/output ของ container กับ terminal สามารถพิมพ์คำสั่งและเห็นผลลัพธ์ได้ทันที
- (2) คอลัมน์ STATUS จากการรันคำสั่ง `docker ps -a` แสดงถึงข้อมูลอะไร
 สถานะการทำงานของ container ในระบบ โดยผลลัพธ์คอลัมน์ 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

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

Lab Worksheet

- ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้
\$ docker build -t <ชื่อ Image> .
- เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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

Terminal

```

PS D:\663380608-3\3-2\SE\Lab8> mkdir Lab8_2

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

Mode                LastWriteTime         Length Name
----                -
d-----          2/4/2026   7:19 PM             Lab8_2

PS D:\663380608-3\3-2\SE\Lab8> cd .\Lab8_2\
PS D:\663380608-3\3-2\SE\Lab8\Lab8_2> dir

Directory: D:\663380608-3\3-2\SE\Lab8\Lab8_2

Mode                LastWriteTime         Length Name
----                -
-a-----          2/4/2026   7:20 PM             171 Dockerfile

```

Containers

Container CPU usage ⓘ

No containers are running.

Container memory usage ⓘ

No containers are running.

Show...

Q Search

☰

● Only show running containers

	Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk read/write	Net	Actions
<input type="checkbox"/>	musings_tesla	be53795d992b	firstimage_		N/A	N/A	N/A	N/A	N/A	▶ ⋮

Terminal

```

PS D:\663380608-3\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:b3255e7dfbcd10cb367af0d409747d511ab666dfac98cf30e97a874207dd76f
=> resolve docker.io/library/busybox:latest@sha256:b3255e7dfbcd10cb367af0d409747d511ab666dfac98cf30e97a874207dd76f
=> exporting to image
=> exporting layers
=> exporting manifest sha256:b6d3b7680a3e5b75cc7c9fe165cf9afbf3de0850d20d1e3acd6c2f16251a83
=> exporting config sha256:1c34b613ffcd2cc5de8439ab4d91cc96a835a0bd80fe842fac13088871d2
=> exporting attestation manifest sha256:a05939cc23808b134013b3529af0b9dacc520a0007c4ee080714e4021ab
=> exporting manifest list sha256:f4a06b0946f249f40264f197a0c34e297031af0b0dc636f3d6c15187a278ae
=> naming to docker.io/library/firstimage_lab8_2:latest
=> unpacking to docker.io/library/firstimage_lab8_2:latest

3 warnings found (use docker --debug to expand):
- MultipleInstructionsNotAllowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
- 350MArgsRecommended: 350M arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
- 350MArgsRecommended: 350M arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
PS D:\663380608-3\3-2\SE\Lab8\Lab8_2> docker run firstimage_lab8_2
~/firstimage_lab8_2
PS D:\663380608-3\3-2\SE\Lab8\Lab8_2>

```

Images

Local

My Hub

4.55 MB / 0 Bytes in use

2 images

Last refresh: 2 hours ago ↻

Q Search

☰ ☰

	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	▶ ⋮ 🗑
<input type="checkbox"/>	firstimage_lab8_2	latest	f4a06b0946f2	1 year ago	6.77 MB	▶ ⋮ 🗑

Lab Worksheet

- (1) คำสั่งที่ใช้ในการ run คือ

_____ docker run <ชื่อ image> _____ ในที่นี้ใช้คำสั่ง docker run firstimage_lab8_2 _____

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

_____ -t มาจากคำว่า Tag ใช้กำหนดชื่อหรือเวอร์ชันให้กับ Docker ที่เราสร้างขึ้น _____

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

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้
\$ docker build -t <username ที่ลงทะเบียนกับ Docker Hub>/lab8 .
6. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง
\$ docker run <username ที่ลงทะเบียนกับ Docker Hub>/lab8

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

Lab Worksheet

Images [Give feedback](#)

Local

My Hub

4.55 MB / 0 Bytes in use 3 images

Last refresh: 2 hours ago

<input type="checkbox"/>	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	▶ ⋮ 🗑
<input type="checkbox"/>	firstimage_lab8_2	latest	f4a06b0946f2	1 year ago	6.77 MB	▶ ⋮ 🗑
<input type="checkbox"/>	phatcharida21/lab8	latest	10a8db4da449	1 year ago	6.77 MB	▶ ⋮ 🗑

Terminal

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

Mode

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

183

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: J50NArgsRecommended: J50N 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: J50NArgsRecommended: J50N 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:b3255e7dfbcd10cb367af0d409747d511aeb66dfac98cf30e97e87e4207dd76f

=> resolve docker.io/library/busybox:latest@sha256:b3255e7dfbcd10cb367af0d409747d511aeb66dfac98cf30e97e87e4207dd76f

=> exporting to image

=> exporting layers

=> exporting manifest sha256:4c62a6589c6958b2f05a3186b39ff1753cF41ecaf96ff192b9d321fc016be723

=> exporting config sha256:8aeb66674a76364760a6f32907e9631197d33360249a966205Scc4a5ba8e919

=> exporting attestation manifest sha256:0fc89eb7b07c3ab999da7305a977a0eeec96d155b18932aa652b77776ee072aa

=> exporting manifest list sha256:10a8db4da449747d1d15c5be748a4e00a4739a8e54ccfb08de33da49b378769a

=> naming to docker.io/phatcharida21/lab8:latest

=> unpacking to docker.io/phatcharida21/lab8:latest

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

- J50NArgsRecommended: J50N 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)

- J50NArgsRecommended: J50N 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

^C

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

docker:desktop-linux

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

0.0s

Containers [Give feedback](#)

Container CPU usage

Container memory usage

Show charts

No containers are running.

No containers are running.

Only show running containers

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Disk	Actions
<input type="checkbox"/>	musings_tesla	be53795d992b	firstimage_lab8_2		N/A	N/A	N/A	N/A	▶ ⋮ 🗑
<input type="checkbox"/>	eager_lumiere	11d2f37d997c	phatcharida21/lab8		N/A	N/A	N/A	N/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 แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง
```

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

7

Lab Worksheet

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

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

The screenshot displays two parts of the Docker ecosystem. The top part is a screenshot of a Docker Hub repository page for 'phatcharida21/lab8'. The bottom part is a screenshot of the Docker Desktop 'Images' and 'Containers' tabs.

Docker Hub Repository Page:

- Repositories:** All repositories within the phatcharida21 namespace.
- Search:** Search by repository name. Filter: All content.
- Repository List:**

Name	Last Pushed	Contains	Visibility	Scout
phatcharida21/lab8	3 minutes ago	IMAGE	Public	Inactive

Docker Desktop Interface:

- Images Tab:**
 - Local: 4.55 MB / 0 Bytes in use. 3 Images.
 - Table:

Name	Tag	Image ID	Created	Size	Actions
busybox	latest	b3255e7dfbcd	1 year ago	6.77 MB	[Play] [List] [Delete]
firstimage_lab8_2	latest	f4a06b0946f2	1 year ago	6.77 MB	[Play] [List] [Delete]
phatcharida21/lab8	latest	10a8db4da449	1 year ago	6.77 MB	[Play] [List] [Delete]
- Terminal:**

```

- 350MargsRecommended: 350M 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 3)
- 350MargsRecommended: 350M arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
PS D:\663380608-3\3-2\5E1\Lab8> docker run phatcharida21/lab8
docker: unknown command: docker phatcharida21/lab8

Run 'docker --help' for more information
PS D:\663380608-3\3-2\5E1\Lab8_2> docker run phatcharida21/lab8
^C
PS D:\663380608-3\3-2\5E1\Lab8_2> docker push phatcharida21/lab8
Using default tag: latest
The push refers to repository [docker.io/phatcharida21/lab8]
61dfb56712f5: Layer already exists
7073c61e8ab1: Pushed
latest: digest: sha256:10a8db4da4497471d159c5be748a4e00a4739a8e54ccfb08de33da49b378769a size: 855
PS D:\663380608-3\3-2\5E1\Lab8_2>

```
- Containers Tab:**
 - Container CPU usage: No containers are running.
 - Container memory usage: No containers are running.
 - Search: [Search]. Filter: Only show running containers.
 - Table:

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	[Play] [List] [Delete]
eager_lumiere	11d2f37d997c	phatcharida21/lab8		N/A	N/A	N/A	N/A	[Play] [List] [Delete]

แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4

Lab Worksheet

- 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
```

- 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

The screenshot shows a terminal window at the top with the following output:

```
PS D:\663380608-3\3-2\SE\Lab8> cd .\Lab8_4\
PS D:\663380608-3\3-2\SE\Lab8_4> git clone https://github.com/docker/getting-started.git
Cloning into 'getting-started'...
remote: Enumerating objects: 987, done.
remote: Counting objects: 100% (274/274), done.
remote: Compressing objects: 100% (44/44), done.
remote: Total 987 (delta 247), reused 230 (delta 230), pack-reused 713 (from 1)
Receiving objects: 100% (987/987), 5.28 MiB | 3.99 MiB/s, done.
Resolving deltas: 100% (539/539), done.
PS D:\663380608-3\3-2\SE\Lab8_4>
```

Below the terminal is a VS Code editor window with the file explorer on the left showing the project structure. The main editor area displays the contents of the `package.json` file:

```
1  {
2    "name": "101-app",
3    "version": "1.0.0",
4    "main": "index.js",
5    "license": "MIT",
6    "scripts": {
7      "prettify": "prettier -l --write \"**/*.js\"",
8      "test": "jest",
9      "dev": "nodemon src/index.js"
10   },
11   "dependencies": {
12     "express": "^4.18.2",
13     "mysql2": "^2.3.3",
14     "sqlite3": "^5.1.2",
15     "uuid": "^9.0.0",
16     "wait-port": "^1.0.4"
17   },
18   "resolutions": {
19     "ansi-regex": "5.0.1"
20   },
21   "prettier": {
22     "trailingComma": "all",
23     "tabWidth": 4,
24     "useTabs": false,
25     "semi": true,
26     "singleQuote": true
27   },
28   "devDependencies": {
29     "jest": "^29.3.1",
30     "nodemon": "^2.0.20",
31     "prettier": "^2.7.1"
32   }
33 }
```

- ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไปไฟล์

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

```
WORKDIR /app
```

```
COPY . .
```

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

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

```
EXPOSE 3000
```

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

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

Lab Worksheet

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

```

PS D:\663380608-3\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
=> transferring context: 2B
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca09d9e
=> resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca09d9e
=> [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:bc47845db128fcd3ff124792397f9668b556e5f6f76381aff72a9374dc8
=> exporting config sha256:f64f4c76a3f9325fe98bcb7630affb644dd1d2daa6d1288977ea7f739204d
=> exporting attestation manifest sha256:cfecc2f3f8c782cfc75742568991a4fab0abbf0a7c234ae4ff7c3d38d4a7ea
=> exporting manifest list sha256:92df1dbce12a04bb4a24deeb4a797408104c1f5affe72a2266f6485838da2304
=> naming to docker.io/library/myapp_6633806083:latest
=> unpacking to docker.io/library/myapp_6633806083:latest
PS D:\663380608-3\3-2\SE\Lab8\Lab8_4\getting-started\app>
  
```

File Edit Selection View Go Run Terminal Help

EXPLORER

- GETTING-STARTED
 - .github\workflows
 - ! build.yml
- app
 - spec
 - persistence
 - routes
 - src
- Dockerfile
- package.json
- yarn.lock
- docs
 - css
 - fonts
 - images
 - tutorial
- index.md
- .dockerignore
- .gitignore
- build.sh
- docker-compose.yml
- Dockerfile
- LICENSE
- mkdocs.yml
- README.md
- requirements.txt

package.json Dockerfile X

app > Dockerfile > ...

```

1 FROM node:18-alpine
2 WORKDIR /app
3 COPY . .
4 RUN yarn install --production
5 CMD ["node", "src/index.js"]
6 EXPOSE 3000
7
  
```

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

Lab Worksheet

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

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

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

The screenshot displays the Docker Desktop interface. The top section shows 'Images' with a list of four images: 'firstimage_lab8_2', 'phatcharida21/lab8', 'busybox', and 'myapp_6633806083'. Below this is the 'Terminal' section showing the command 'docker run -dp 3000:3000 myapp_6633806083' and its output. The bottom section shows 'Containers' with a list of three containers: 'eager_lumiere', 'musing_tesla', and 'gifted_moser'. The browser window at the bottom shows a web application with a search bar and a list of items: 'Book' and 'Pen'.

หมายเหตุ: นศ.สามารถทดลองเล่น Web application ที่ทำงานอยู่ได้

8. ทำการแก้ไข Source code ของ Web application ดังนี้
- เปิดไฟล์ src/static/js/app.js ด้วย Editor และแก้ไขบรรทัดที่ 56 จาก

<p className="text-center">No items yet! Add one above!</p> 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 (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ พร้อมกับตอบคำถามต่อไปนี้

```

Terminal
PS D:\663380688-3\3-2\SE\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6633806883 .
[+] Building 4.6s (9/10) 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:bc47045db128fcd3df124793397f9668b556ed5fa6fb76301af72a93744ccc8
[+] Building 28.3s (10/10) 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: 2B
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3bc498332f249011d1189455880a35cb9bc4b8ca09d9e
=> => resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3bc498332f249011d1189455880a35cb9bc4b8ca09d9e
=> [internal] load build context
=> => transferring context: 8.13kB
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:11b28c5b91e0aed7720c35a5e053b558ca496e08328a722109efae7a8834599
=> => exporting config sha256:62bc3dec4ff8035a5a1d160d5e85b8bb9b63d0bb3b6f54da6f704798c49f7d
=> => exporting attestation manifest sha256:9b70e294ae39f3378888694209a2561a935e2ee91d76099a23a595642bf56fb

PS D:\663380688-3\3-2\SE\Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6633806883
327d96153b0e0df01bb59b76deed0b1a2baa73fa365c8543e3c3f7b155ae637
docker: Error response from daemon: failed to set up container networking: driver failed programming external connectivity on endpoint jovial_euclid (415c96fbad62b9e6ad0eb7f0f44a638aa877bd88fc07295804edd06974cdeb): Bind
d for 0.0.0.0:3000 failed: port is already allocated

Run 'docker run --help' for more information

```

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

พอร์ต 3000 กำลังถูกใช้งานอยู่ เกิดจากการใช้ พอร์ต 3000 โดยยังไม่ได้หยุดการทำงานของโปรแกรมที่เคยใช้ก่อนหน้านี้ _____

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
- ใช้คำสั่ง \$ docker stop <Container ID ที่ต้องการจะลบ> เพื่อหยุดการทำงานของ Container ดังกล่าว
- ใช้คำสั่ง \$ docker rm <Container ID ที่ต้องการจะลบ> เพื่อทำการลบ

b. ผ่าน Docker desktop

- ไปที่หน้าต่าง Containers
- เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
- ยืนยันโดยการกด Delete forever

12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6

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

Lab Worksheet

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

The screenshot shows the Docker Desktop interface. At the top, there's a terminal window with commands like `docker ps`, `docker stop`, `docker rm`, and `docker run`. Below the terminal, the 'Containers' section displays a table of running containers:

Name	Container ID	Image	Port(s)	CPU (%)	Memory usage...	Memory (%)	Dis	Actions
eager_lumiere	11d2f37d997c	phatcharida21/lab8		0%	0B / 0B	0%	0B	[Stop] [Refresh] [Delete]
musing_tesla	be53795d992b	firstimage_lab8_2		0%	0B / 0B	0%	0B	[Stop] [Refresh] [Delete]
jovial_euclid	327d96153b0e	myapp_6633806083	3000:3000	0%	0B / 0B	0%	0B	[Stop] [Refresh] [Delete]
condescending_h	d79d67e5a7dd	myapp_6633806083	3000:3000	0%	18.68MB / 3.73G	0.49%	6.7	[Stop] [Refresh] [Delete]

Below the containers table, there's a section for 'localhost:3000' showing a web application. It has a 'New Item' button and an 'Add Item' button. The message says 'There is no TODO item. Please add one to the list. By พัชรดา เพ็งอามรย์'.

Below that, there's another section for 'localhost:3000' showing a list of items: 'ข้าว', 'น้ำ', and 'ปลา'. Each item has a checkbox and a delete icon.

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 ที่ใส่คำสั่งที่เกี่ยวข้องไว้

```
FROM jenkins/jenkins:its
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:\66338668-3\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/jenkins:its
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/3] FROM docker.io/jenkins/jenkins:its@sha256:d1ea795c6facd7f549a21c40e5e43ffcc5fbc5f48683d9b24750f26e8079d772
=> => resolve docker.io/jenkins/jenkins:its@sha256:d1ea795c6facd7f549a21c40e5e43ffcc5fbc5f48683d9b24750f26e8079d772
=> CACHED [2/3] RUN apt-get update && apt-get install -y python3 python3-pip python3-venv
=> [3/3] RUN pip install robotframework --break-system-packages
=> => exporting to image
=> => exporting layers
=> => exporting manifest sha256:2f8b0f83b7295345c1536d417dcb16fd7744977c8767a1937a1d4ae394b7aacf
=> => exporting config sha256:f09fb9fa32b161a5775835ad488a29a823f8fadc476c6d44dad4207bb0dd58
=> => exporting attestation manifest sha256:cdc4b7052f9ea31cac1928f5db738965958d057855e0f9e581bb21633a2c5d7d
=> => exporting manifest list sha256:24907dd9dae176d129abeb1f1b471af27df2ab7cb2135ea93e8da9c78d61f96
=> => naming to docker.io/library/jenkins-robot-local:latest
=> => unpacking to docker.io/library/jenkins-robot-local:latest
PS D:\66338668-3\3-2\SE\Lab8>

```

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

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

- 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

```

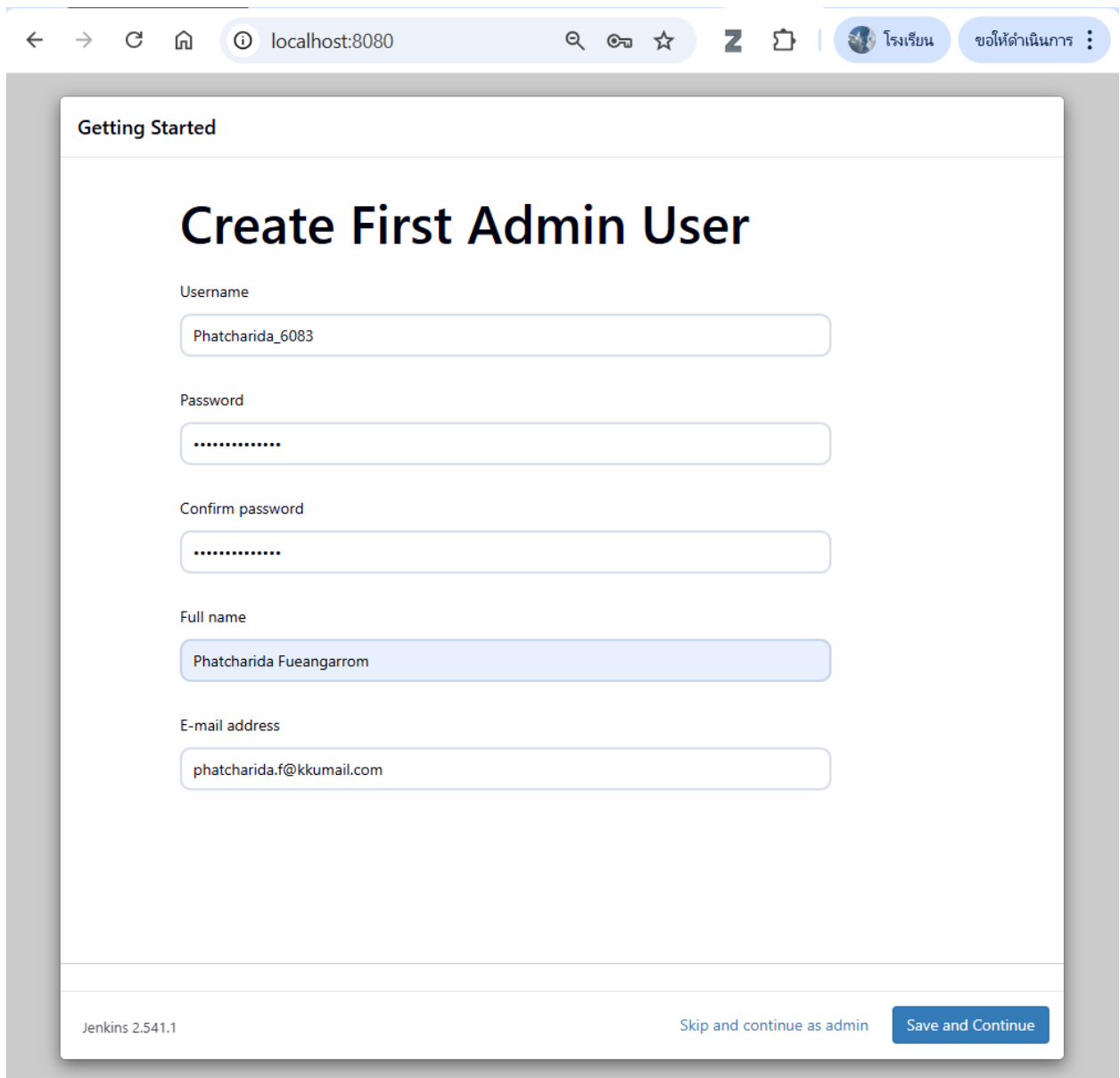
Terminal
2026-02-04 14:44:03.065+0000 [id=33] INFO jenkins.model.Jenkins#<init>: Starting version 2.541.1
2026-02-04 14:44:03.224+0000 [id=42] INFO jenkins.InitReactorRunner$1#<onAttained>: Started initialization
2026-02-04 14:44:03.252+0000 [id=43] INFO jenkins.InitReactorRunner$1#<onAttained>: Listed 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=43] 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 14:44:04.689+0000 [id=41] INFO jenkins.InitReactorRunner$1#<onAttained>: Configuration for all jobs updated
2026-02-04 14:44:04.845+0000 [id=61] INFO hudson.util.Retrier#start: 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]> *****
[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]>
[LF]> 73f8e5774c5d4b919b0d6984b15ff6e6f
[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$1#<onAttained>: Completed initialization
2026-02-04 14:44:15.629+0000 [id=33] INFO hudson.lifecycle.Lifecycle#<onReady>: Jenkins is fully up and running
2026-02-04 14:44:17.743+0000 [id=61] INFO h.n.DownloadService$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#start: Performed the action check updates server successfully at the attempt #1

```

- บันทึกรหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก
- เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น http://localhost:8080
- ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3
- สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

[Check point#14] Capture หน้าจอที่แสดงผลการตั้งค่า

Lab Worksheet



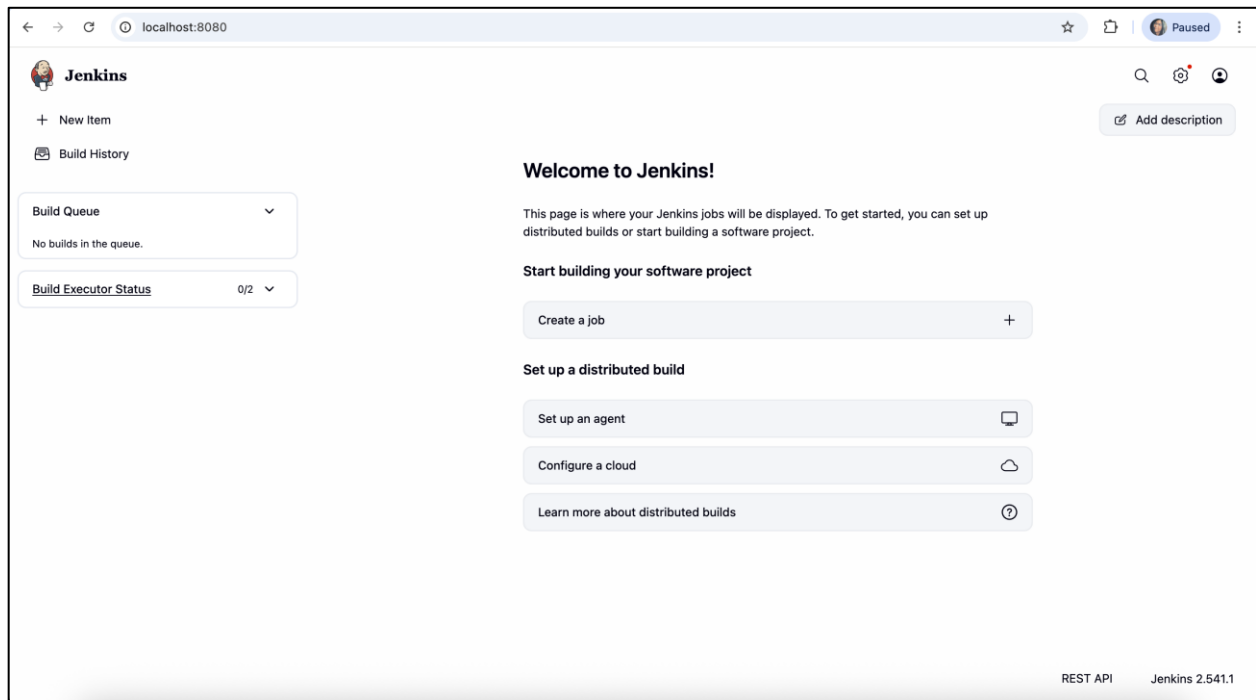
The screenshot shows a web browser window with the address bar displaying 'localhost:8080'. The page title is 'Getting Started'. The main heading is 'Create First Admin User'. The form contains the following fields:

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

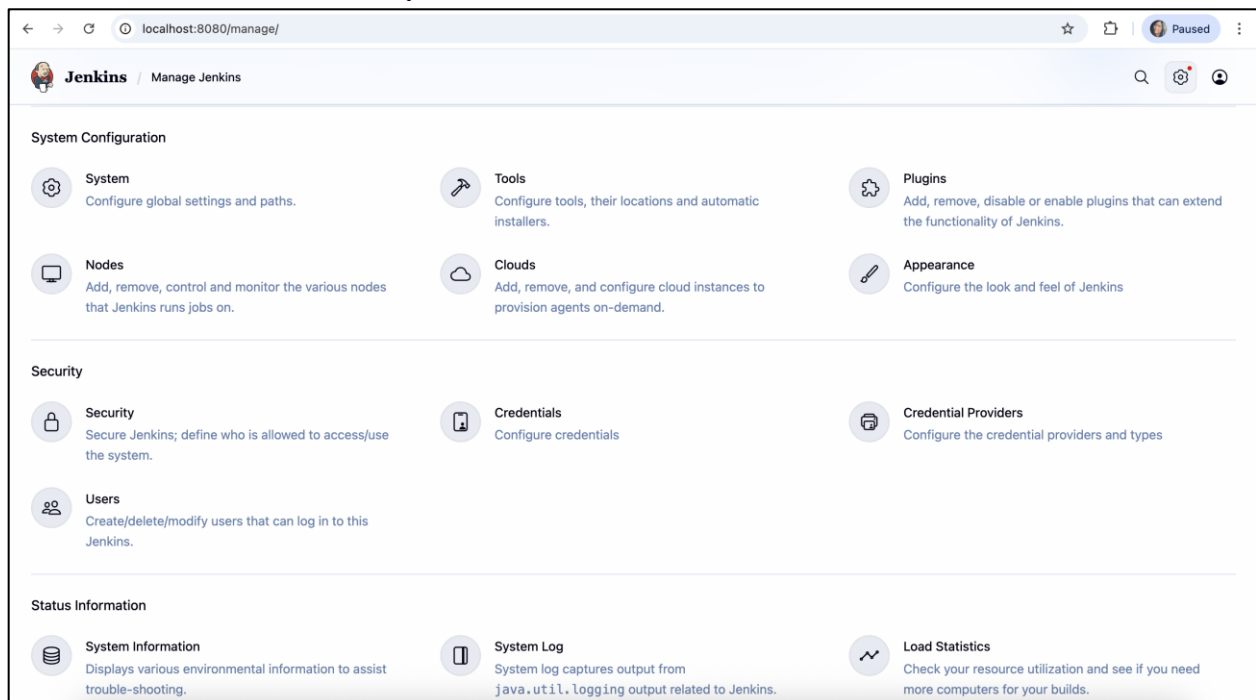
At the bottom left, it says 'Jenkins 2.541.1'. At the bottom right, there are two buttons: 'Skip and continue as admin' and 'Save and Continue'.

9. เมื่อติดตั้งเรียบร้อยแล้วจะพบหน้าจอ Dashboard ดังแสดงในภาพ

Lab Worksheet

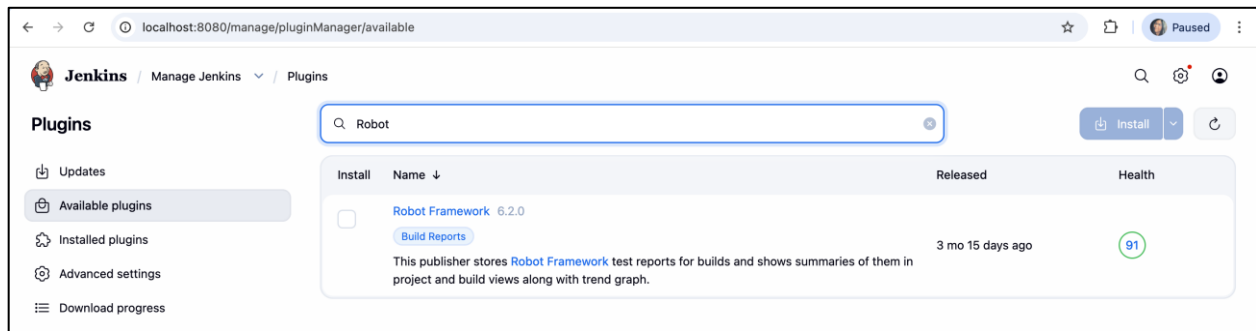


10. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

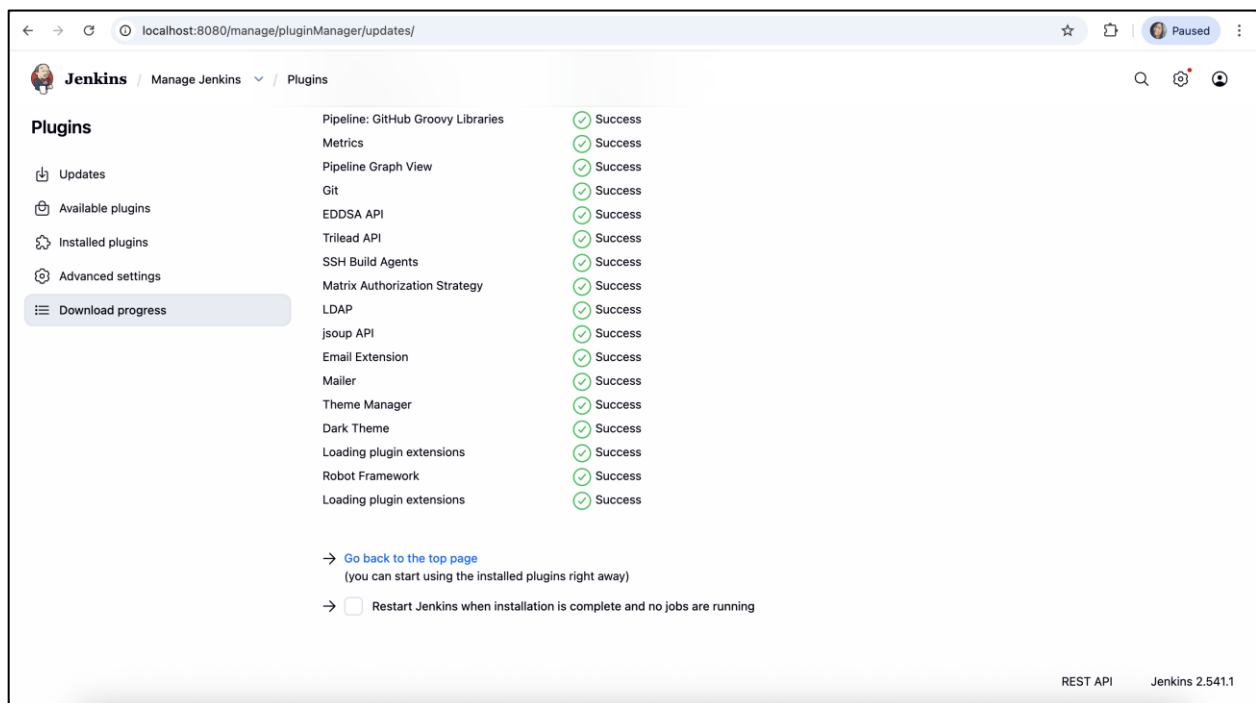


11. ไปที่เมนู Plugins > Available plugins แล้วเลือกติดตั้ง Robot Framework เพิ่มเติม

Lab Worksheet

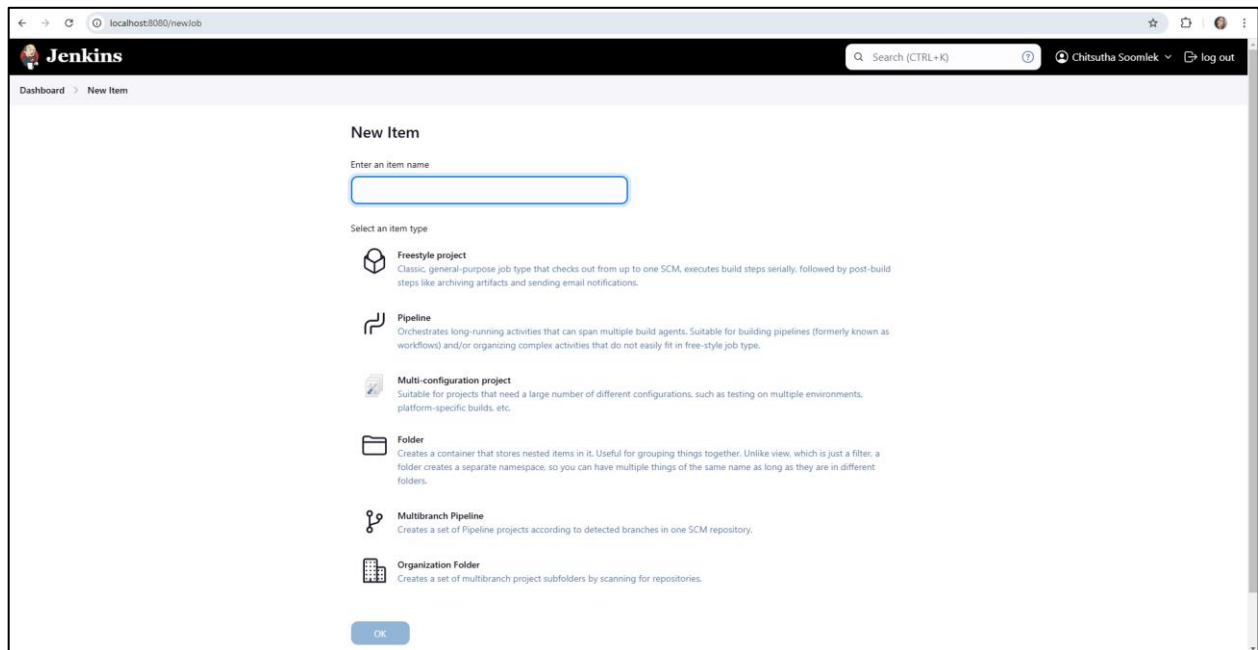


12. เมื่อติดตั้งสำเร็จจะพบกับรายการ Plugins ทั้งหมดที่ถูกติดตั้ง ถ้าติดตั้งสำเร็จให้เลือก “Restart Jenkins...” แล้วกด Go back to the top page



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

Lab Worksheet



16. ตั้งค่าที่จำเป็นในหน้านี้นี้ทั้งหมด ดังนี้

Description: Lab 8.5

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

Build Trigger: เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

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

[Check point#15] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

(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 projectProject 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 [?](#)<https://github.com/Phatcharida21/CP353004-663380608-3/>Credentials [?](#)

- none -



+ Add

Advanced 

+ Add Repository

Branches to build [?](#)Branch Specifier (blank for 'any') [?](#)

*/main

+ 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

Publish Robot Framework test results ?

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

.

Advanced ^ Edited

Name of archive directory
Name of archive directory where to store builds. Set to archive to use jenkins build archive directory.

robot-test-SE-Lab8

Output xml name ?
Name of the xml file containing robot output

output.xml

Report html name ?
Name of the html file containing robot test report

report.html

Log html name ?
Name of the html file containing detailed robot test log

log.html

Other files to copy ?
Comma separated list of robot related artifacts to be saved

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.

Use Artifact Manager to copy tests results
☐ Use Artifact Manager

Thresholds for build result ?

%
20.0

%
80.0

☐ Include skipped tests in total count for thresholds


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

17. กด Apply และ Save

18. สั่ง Build Now

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

Lab Worksheet


Jenkins / UAT / #16

Status

</> Changes

Console Output

Edit Build Information

Delete build '#16'


Timings


Git Build Data


Robot Results

Open Blue Ocean


← Previous Build


#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

Status

</> Changes

Console Output

Edit Build Information

Timings

Git Build Data

Robot Results

Open Blue Ocean

← Previous Build

Robot Framework Test Results

Executed: 2026-02-04T16:37:50.661349
Duration: 0:00:00.225 (+0:00:00.175)
Status: 2 test total (±0), 2 passed, 0 failed, 0 skipped
Results: [report.html](#)
[log.html](#)
[Original result files](#)

Test Result Trend



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

Duration Trend




[Show bigger image](#)

Test Suites

Name	Failed tests (±0)	Total tests (±0)	Duration (±0)
Lab8	0 (±0)	2 (±0)	0:00:00.225 (+0:00:00.175)

Lab Worksheet


Jenkins / UAT / #16

Status

</> Changes

Console Output

✓ Edit Build Information

🗑 Delete build #16

🕒 Timings

📊 Git Build Data

📄 Robot Results

🌊 Open Blue Ocean

← Previous Build

Console

```

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 b03980fb2cba93f945e1075aa624366e7295bfb2 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f b03980fb2cba93f945e1075aa624366e7295bfb2 # timeout=10
Commit message: "Update Lab8.robot"
> git rev-list --no-walk b03980fb2cba93f945e1075aa624366e7295bfb2 # timeout=10
[UAT] $ /bin/sh -xe /tmp/jenkins3972905542996132173.sh
+ robot Lab8/test/Lab8.robot
=====
Lab8 :: ตรวจสอบความพร้อมของระบบ Lab 8.5
=====
Verify GitHub Integration :: ตรวจสอบว่าได้จาก GitHub ถูกหรือไม่ ... | PASS |
-----
Verify Robot Framework Execution :: ทดสอบการทำงานของ Robot ... System is ready for Deployment!
| PASS |
-----
Lab8 :: ตรวจสอบความพร้อมของระบบ 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

```