

Lab Worksheet

ชื่อ-นามสกุล น.ส.วรัญญา ชุ่มเมือง รหัสนักศึกษา 653380281-8 Section 1

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

Lab Worksheet

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

1.

```
PS D:\> mkdir Lab8_1
```

Directory: D:\

Mode	LastWriteTime	Length	Name
d----	1/27/2025 11:15 PM		Lab8_1

2.

```
PS D:\> cd Lab8_1
```

```
PS D:\Lab8_1>
```

3.

Images

View and manage your local and Docker Hub images. [Learn more](#)

0 Bytes / 0 Bytes in use 1 images Last refresh: 3 hours ago

Name	Tag	Image ID	Created	Size	Actions
busybox	latest	af4709625109	4 months ago	4.26 MB	

Terminal

```
PS D:\> cd Lab8_1
PS D:\Lab8_1> docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
9c0abc9c5bd3: Pull complete
Digest: sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab073cd3c138bd458fc8257fbf1
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
```

4.

```
PS D:\Lab8_1> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
busybox	latest	af4709625109	4 months ago	4.27MB

Lab Worksheet

- (1) สิ่งที่อยู่ภายใต้คอนเทนเนอร์ Repository คืออะไร คอนเทนเนอร์ Repository แสดงชื่อของภาพ ที่อยู่ใน Docker Hub หรือที่สร้างขึ้นในเครื่อง เช่น busybox
- (2) Tag ที่ใช้บ่งบอกถึงอะไร ใช้บ่งบอกถึงเวอร์ชันเฉพาะของภาพนั้น เช่น latest หมายถึงเวอร์ชันล่าสุด หากไม่ระบุ Tag โดยเฉพาะ ระบบจะดาวน์โหลด Tag ค่าเริ่มต้นคือ latest

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

6.

```
PS D:\Lab8_1> docker run busybox
PS D:\Lab8_1> docker run -it busybox sh
/ #
```

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

Containers							
<input type="text" value="Search"/>		<input type="checkbox"/> Only show running containers					
<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last star	Actions
<input type="checkbox"/>	tender_greider	c62fa41b4909	busybox		0%	1 minute	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
<input type="checkbox"/>	elastic_almeida	dee374f5383c	busybox		0%	33 second	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

7.

```
/ # ls
bin  dev  etc  home  lib  lib64  proc  root  sys  tmp  usr  var
/ #
```

Lab Worksheet

8.

```
/ # ls -la
total 48
drwxr-xr-x 1 root root 4096 Jan 27 16:22 .
drwxr-xr-x 1 root root 4096 Jan 27 16:22 ..
-rwxr-xr-x 1 root root 0 Jan 27 16:22 .dockerenv
drwxr-xr-x 2 root root 12288 Sep 26 21:31 bin
drwxr-xr-x 5 root root 360 Jan 27 16:22 dev
drwxr-xr-x 1 root root 4096 Jan 27 16:22 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
```

9.

```
/ # exit
PS D:\Lab8_1>
```

Containers [Give feedback](#)View all your running containers and applications. [Learn more](#)

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last star	Actions
<input type="checkbox"/>	tender_greider	c62fa41b4909	busybox		N/A	3 minute	▶ ⋮ 🗑
<input type="checkbox"/>	elastic_almeida	dee374f5383c	busybox		N/A	3 minute	▶ ⋮ 🗑

10.

```
PS D:\Lab8_1> docker run busybox echo Hello Waranya Hommuang from busybox
Hello Waranya Hommuang from busybox
PS D:\Lab8_1>
```

Containers [Give feedback](#)View all your running containers and applications. [Learn more](#)

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last star	Actions
<input type="checkbox"/>	tender_greider	c62fa41b4909	busybox		N/A	5 minute	▶ ⋮ 🗑
<input type="checkbox"/>	elastic_almeida	dee374f5383c	busybox		N/A	4 minute	▶ ⋮ 🗑
<input type="checkbox"/>	xenodochial_lov	12d13051a2f8	busybox		N/A	1 second	▶ ⋮ 🗑

11.

```
PS D:\Lab8_1> docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS                    PORTS     NAMES
12d13051a2f8   busybox   "echo Hello Waranya ..." 52 seconds ago   Exited (0) 51 seconds ago           xenodochial_lovelace
dee374f5383c   busybox   "sh"                      5 minutes ago    Exited (0) 2 minutes ago           elastic_almeida
c62fa41b4909   busybox   "sh"                      6 minutes ago    Exited (0) 6 minutes ago           tender_greider
```

Lab Worksheet

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

-i (interactive) เปิดการใช้งานอินพุตแบบโต้ตอบระหว่างผู้ใช้กับคอนเทนเนอร์

-t (tty) สร้าง pseudo-TTY (terminal) เพื่อจำลอง terminal สำหรับการใช้งานภายในคอนเทนเนอร์

ดังนั้น -it ช่วยให้ผู้ใช้สามารถเข้าสู่ shell ภายในคอนเทนเนอร์และทำงานในโหมดโต้ตอบ เช่น ใช้คำสั่ง sh, bash หรือคำสั่งอื่นๆ ได้โดยตรง

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

แสดงสถานะของคอนเทนเนอร์ในปัจจุบันหรือสถานะล่าสุดของคอนเทนเนอร์ เช่น

Up คือคอนเทนเนอร์กำลังทำงานอยู่ในขณะนั้น

Exited คือคอนเทนเนอร์หยุดทำงานพร้อมตัวเลขรหัสสถานะ (Exit Code)

Created คือคอนเทนเนอร์ถูกสร้างขึ้นแต่ยังไม่ได้เริ่มทำงาน

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

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

```
PS D:\Lab8_1> docker rm 12d13051a2f8
12d13051a2f8
PS D:\Lab8_1>
```

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	tender_greider	c62fa41b4909	busybox		N/A	10 minutes ago	
<input type="checkbox"/>	elastic_almeida	dee374f5383c	busybox		N/A	10 minutes ago	

แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

- เปิดใช้งาน 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

Lab Worksheet

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 และบันทึกคำสั่งต่อไปนี้

```
$ 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

```
PS D:\SE_Lab8\Lab8_2> docker build -t my-first-image .
[+] Building 0.1s (5/5) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 224B 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior rel 0.0s
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same s 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior rel 0.0s
=> [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
```

Lab Worksheet

Images [Give feedback](#)View and manage your local and Docker Hub images. [Learn more](#)

<input type="checkbox"/>	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	busybox	latest	af4709625109	4 months ago	4.26 MB	
<input type="checkbox"/>	my-first-image	latest	c77e405eaabe	4 months ago	4.26 MB	

(1) คำสั่งที่ใช้ในการ run คือ

`docker run my-first-image`

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

-t ใช้สำหรับตั้งชื่อ (tag) ให้กับ Docker Image ที่สร้างขึ้น

การตั้งชื่อช่วยให้จัดการกับภาพ (Image) ได้ง่ายขึ้น โดยผู้ใช้สามารถเรียกใช้ภาพผ่านชื่อที่กำหนดได้แทนการอ้างอิงด้วย IMAGE ID

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`

Lab Worksheet

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

```
PS D:\SE_Lab8\Lab8_3> docker run waranyaho/lab8
"วรัญญา ชื่นเมือง 653380281-8"
```

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

<input type="checkbox"/>	<input type="radio"/>	wonderful_shannon	c0fd5ad61038	waranyaho/lab8	N/A	49 seconds ago			
--------------------------	-----------------------	-------------------	--------------	----------------	-----	----------------	--	--	--

Images [Give feedback](#)

View and manage your local and Docker Hub images. [Learn more](#)

<input type="checkbox"/>	<input checked="" type="radio"/>	waranyaho/lab8	latest	da3a79cb051e	4 months ago	4.26 MB			
--------------------------	----------------------------------	----------------	--------	--------------	--------------	---------	--	--	--

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง

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

ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

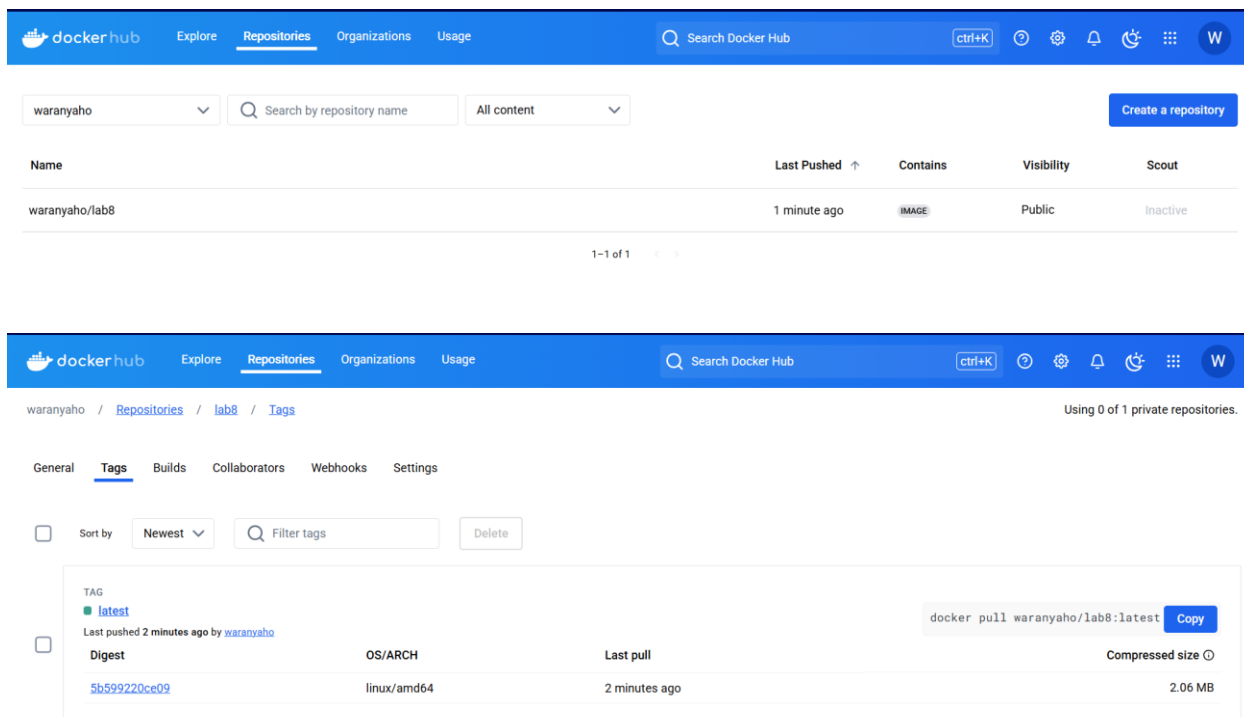
\$ docker login แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง

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

Lab Worksheet

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

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



แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

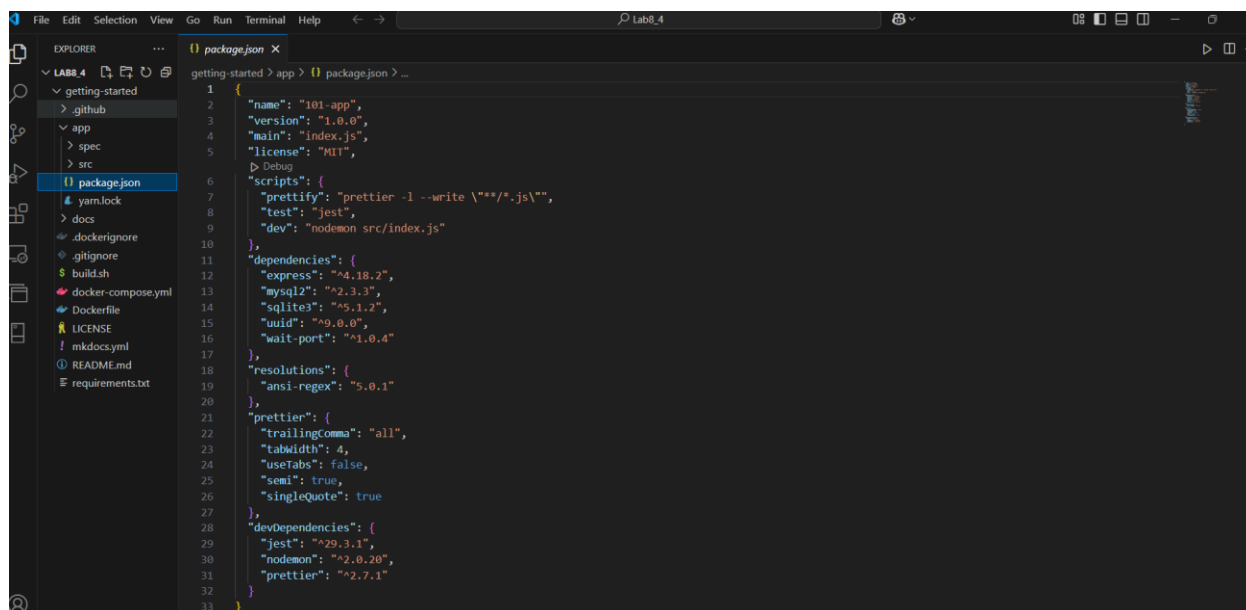
1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4
2. ทำการ Clone ซอร์สโค้ดของเว็บแอปพลิเคชันจาก GitHub repository
<https://github.com/docker/getting-started.git> ลงใน Directory ที่สร้างขึ้น โดยใช้คำสั่ง

```
$ 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

Lab Worksheet

```
PS D:\SE_Lab8\Lab8_4> git clone https://github.com/docker/getting-started.git
Cloning into 'getting-started'...
remote: Enumerating objects: 980, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 980 (delta 5), reused 1 (delta 1), pack-reused 971 (from 2)Receiving objects: 100% (980/980), 3.00 MiB | 2.99 MiB/s
Receiving objects: 100% (980/980), 5.28 MiB | 4.10 MiB/s, done.
Resolving deltas: 100% (523/523), done.
```



4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงในไฟล์

FROM node:18-alpine

WORKDIR /app

COPY . .

RUN yarn install --production

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

EXPOSE 3000

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดใช้ชื่อ image เป็น

myapp_รหัสศ. ไม่มีขีด

\$ docker build -t <myapp_รหัสศ. ไม่มีขีด> .

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ

Lab Worksheet

```

PS D:\SE_Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533802818 .
[+] Building 24.3s (10/10) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 154B                                0.0s
=> [internal] load metadata for docker.io/library/node:18-alpine    4.8s
=> [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@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 6.7s
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 0.0s
=> => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 7.67kB / 7.67kB 0.0s
=> => sha256:6e804119c3884fc5782795bf0d2adc89201c63105aece8647b17a7bcebbcb385e 1.72kB / 1.72kB 0.0s
=> => sha256:dcfb7b337595be6f4d214e4eed84f230eeef0e4ac03a50380d573e289b9e5e40 6.18kB / 6.18kB 0.0s
=> => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 3.64MB / 3.64MB 2.0s
=> => sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 40.01MB / 40.01MB 5.5s
=> => sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 1.26MB / 1.26MB 1.4s
=> => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 444B / 444B 1.8s
=> => extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 0.1s
=> => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 1.0s
=> => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 0.0s
=> => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 0.0s
=> [internal] load build context                                    0.3s
=> => transferring context: 4.62MB                                    0.2s
=> [2/4] WORKDIR /app                                              0.1s
=> [3/4] COPY . .                                                  0.0s
=> [4/4] RUN yarn install --production                             11.8s
=> exporting to image                                              0.7s
=> => exporting layers                                              0.7s
=> => writing image sha256:48a5b7ec988fa12a6ae71b2829762351136cae0ee356fce80b67efe2d6f2337a 0.0s
=> => naming to docker.io/library/myapp_6533802818                 0.0s

```

View build details: <docker-desktop://dashboard/build/desktop-linux/desktop-linux/c7wdveyao0uvszt7ef334oz01>

Images [Give feedback](#)

View and manage your local and Docker Hub images. [Learn more](#)

<input type="checkbox"/>	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	busybox	latest	af4709625109	4 months ago	4.26 MB	
<input type="checkbox"/>	my-first-image	latest	c77e405eaabe	4 months ago	4.26 MB	
<input type="checkbox"/>	waranyaho/lab8	latest	da3a79cb051e	4 months ago	4.26 MB	
<input type="checkbox"/>	myapp_6533802818	latest	48a5b7ec988f	3 minutes ago	216.9 MB	

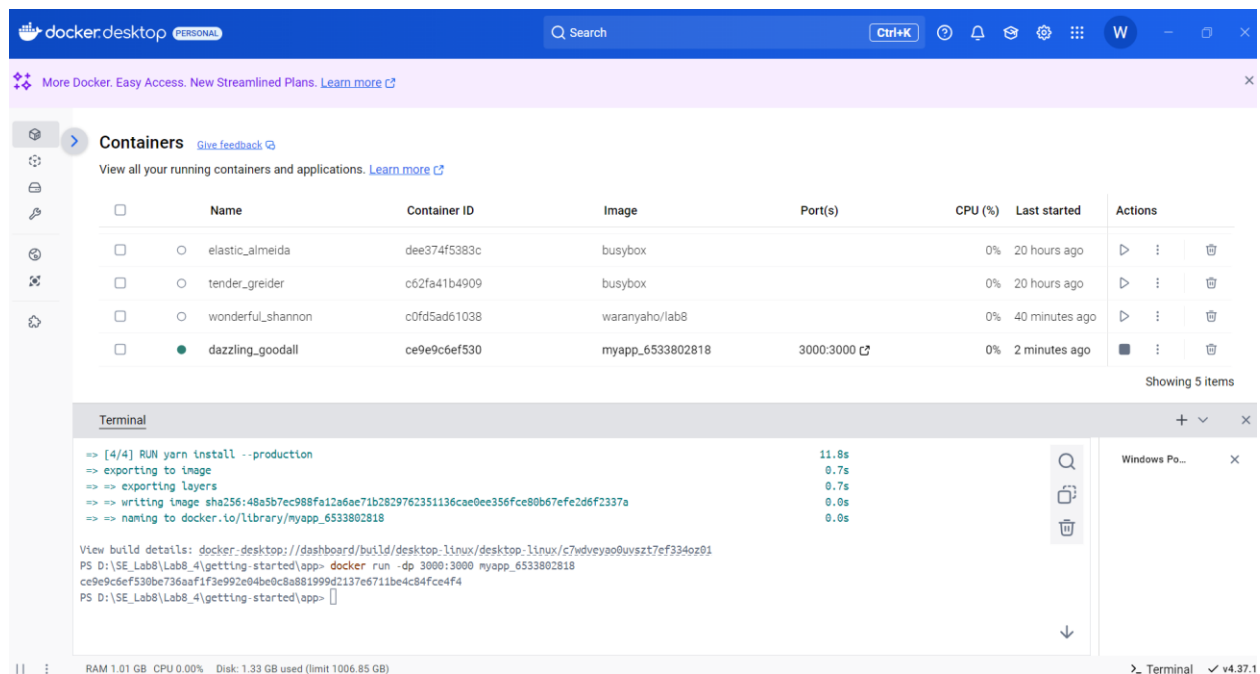
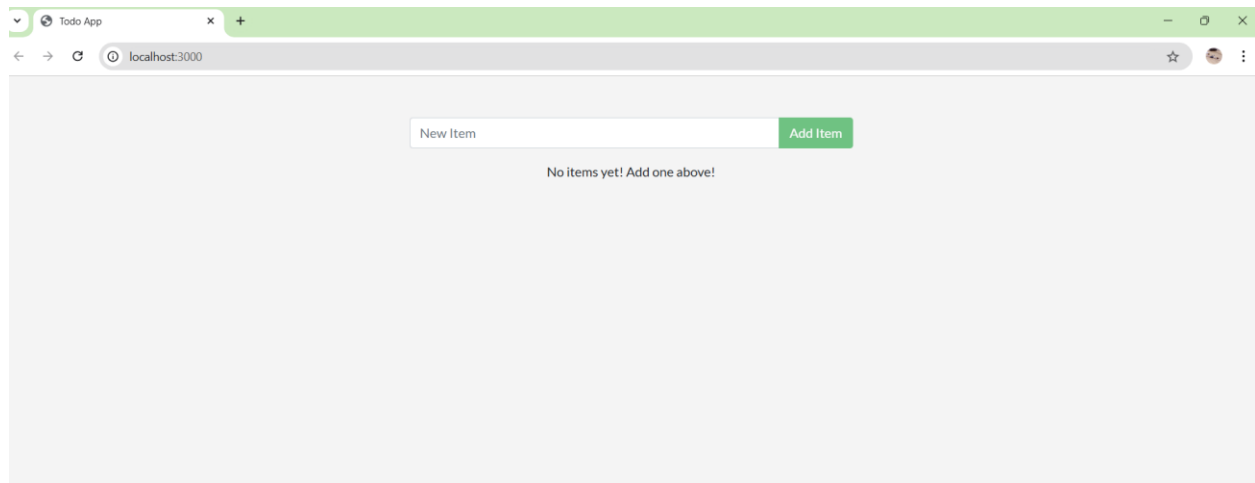
6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

```
$ 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

Lab Worksheet



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

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

```

Dockerfile JS app.js M X
getting-started > app > src > static > js > JS app.js > TodoListCard
14 doListCard() {
30   nItemUpdate = React.useCallback(
31     n => {
37     });
38
39     ams],
40
41
42   nItemRemoval = React.useCallback(
43     n => {
44       const index = items.findIndex(i => i.id === item.id);
45       setItems([...items.slice(0, index), ...items.slice(index + 1)]);
46
47       ams],
48
49
50       ns === null) return 'Loading...';
51
52   (
53     act.Fragment>
54     <AddItemForm onNewItem={onNewItem} />
55     {items.length === 0 && (
56       <p className="text-center">There is no TODO item. Please add one to the list. By Waranya Hommuang</p>
57     )}
58     {items.map(item => (
59       <ItemDisplay

```

PS D:\SE_Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533802818 .

```

[+] Building 24.3s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 154B
=> [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@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25

```

PS D:\SE_Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533802818

```

eb77eef922f81c6f8a30815a04b7684398be332def4b18375424d99b8667e737
docker: Error response from daemon: driver failed programming external connectivity on endpoint competent_lamport (2a4c9d70478b2bdb509bc8ce91635b05eafa4902c673af7f5352
48ccaa27b067): Bind for 0.0.0.0:3000 failed: port is already allocated.

```

Lab Worksheet

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

พอร์ต 3000 บนเครื่องโฮสต์ ถูกใช้งานโดยโปรเซสหรือ Container อื่นอยู่แล้ว
สาเหตุคือมี Container ที่รันอยู่ในระบบซึ่งแมปไปยังพอร์ต 3000 บนโฮสต์แล้ว

11. ลบ Container ของ Web application เวอร์ชันก่อนแก้ไขออกจากระบบ โดยใช้วิธีใดวิธีหนึ่งดังต่อไปนี้

a. ผ่าน Command line interface

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

Lab Worksheet

The screenshot shows the Docker Desktop interface. The top bar includes the Docker logo, 'docker.desktop PERSONAL', a search bar, and various icons. Below the bar, there's a navigation sidebar with icons for Containers, Images, Volumes, Settings, and Docker Hub. The main area is titled 'Containers' and shows a list of running containers. Below this, a terminal window is open, displaying build details and commands.

Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
tender_greider	c62fa41b4909	busybox		0%	20 hours ago	[Play] [Stop] [Delete]
wonderful_shannon	c0fd5ad61038	waranyaho/lab8		0%	1 hour ago	[Play] [Stop] [Delete]
competent_lamport	eb77eef922f8	myapp_6533802818	3000:3000	0%		[Play] [Stop] [Delete]
inspiring_dirac	f0763f1905d6	myapp_6533802818	3000:3000	0%	1 minute ago	[Play] [Stop] [Delete]

Showing 6 items

Terminal

```
View build details: docker-desktop://dashboard/build/desktop:linux/desktop:linux/843ry4vo9a32t42hrs90a7zsa
PS D:\SE_Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533802818
eb77eef922f81c6f8a30815a04b7684398be332def4b18375424d99b8667e737
docker: Error response from daemon: driver failed programming external connectivity on endpoint competent_lamport (2a4c9d70478b2bdb509bc8ce91635b05eafa4902c673af77f535248ccaa27b067): Bind for 0.0.0.0:3000 failed: port is already allocated.
PS D:\SE_Lab8\Lab8_4\getting-started\app> ^C
PS D:\SE_Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533802818
f0763f1905d62957c31e3881f2d7a63a8e0f99c976cc276fb8334e461b10157
PS D:\SE_Lab8\Lab8_4\getting-started\app> []
```

Below the terminal, a web browser window is open showing a 'Todo App' at localhost:3000. The app has a text input 'New Item' and an 'Add Item' button. Below the input, it says 'There is no TODO item. Please add one to the list. By Waranya Hommuang'.

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:its-jdk17
```

 หรือ


```
$ docker run -p 8080:8080 -p 50000:50000 --restart=on-failure -v
jenkins_home:/var/jenkins_home jenkins/jenkins:its-jdk17
```



Lab Worksheet


3. บันทึกการรหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก

[Check point#12] Capture หน้าจอที่แสดงผล Admin password

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

Search   Only show running containers





<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started
<input type="checkbox"/>	priceless_boyd	c3d7d135dc51	my-first-image		0%	19 hours ago
<input type="checkbox"/>	elastic_almeida	dee374f5383c	busybox		0%	21 hours ago
<input type="checkbox"/>	tender_greider	c62fa41b4909	busybox		0%	21 hours ago
<input type="checkbox"/>	romantic_hodgkin	64ee82abfc2e	jenkins/jenkins:its-jdk17	50000:50000  Show all ports (2)	0.13%	4 minutes ago

Terminal

```
*****
Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

4888e9be5f7a4d969a51422d0fe8cc3a

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
*****
```

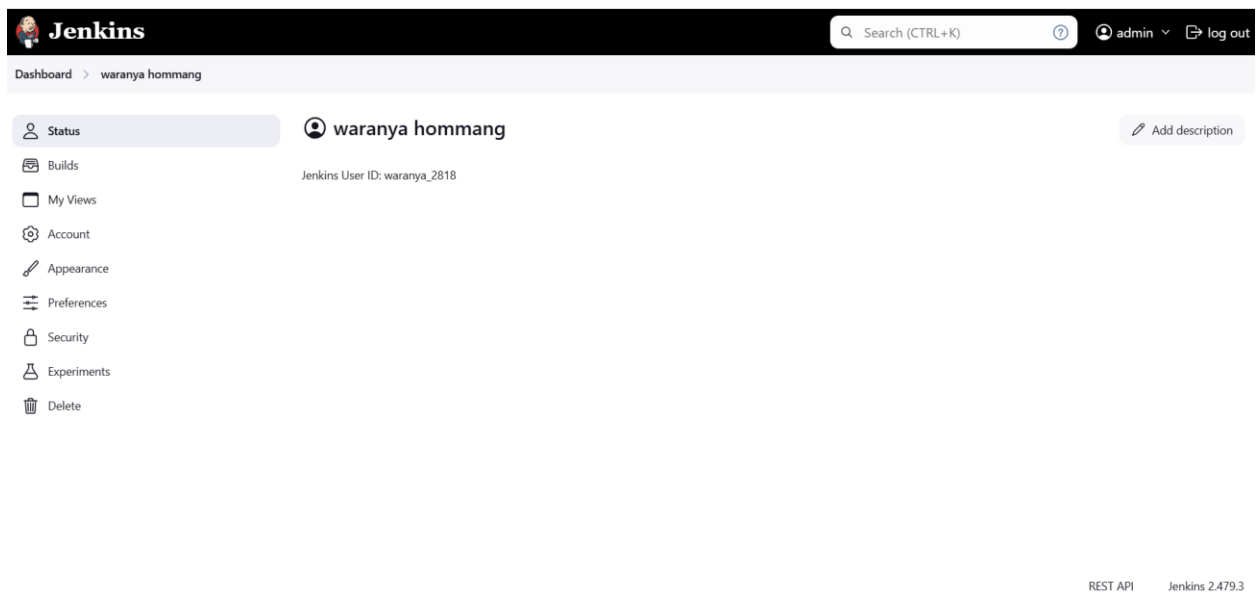
4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น localhost:8080

5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3

6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

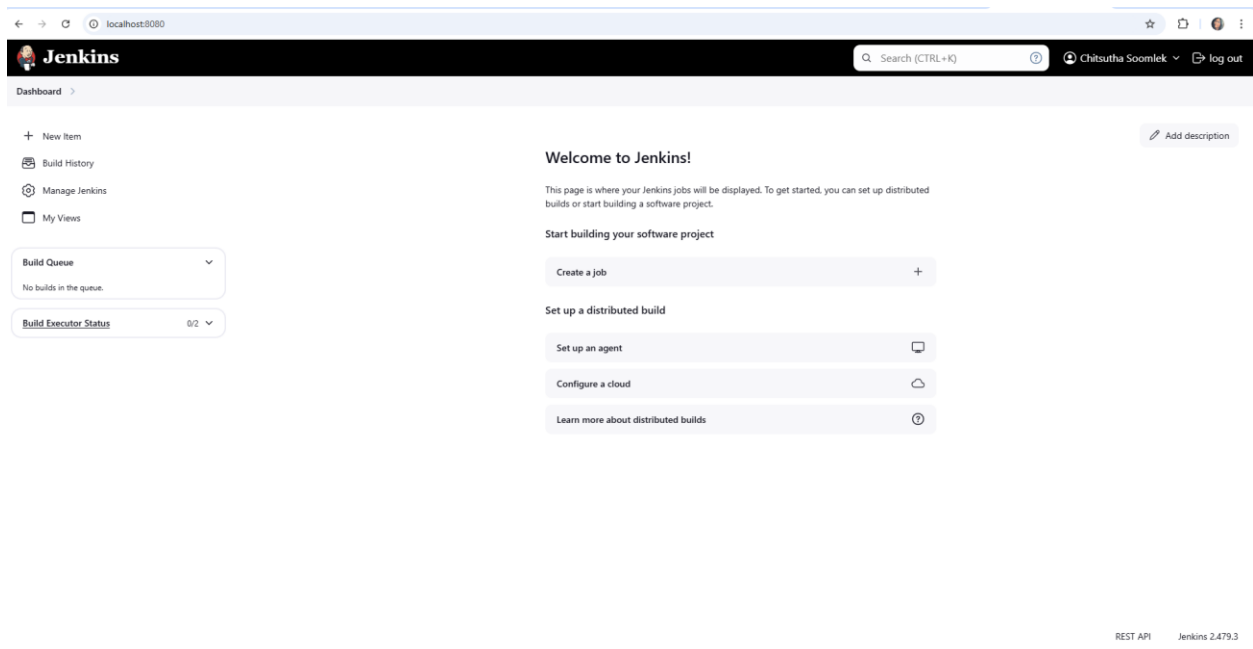
[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

Lab Worksheet



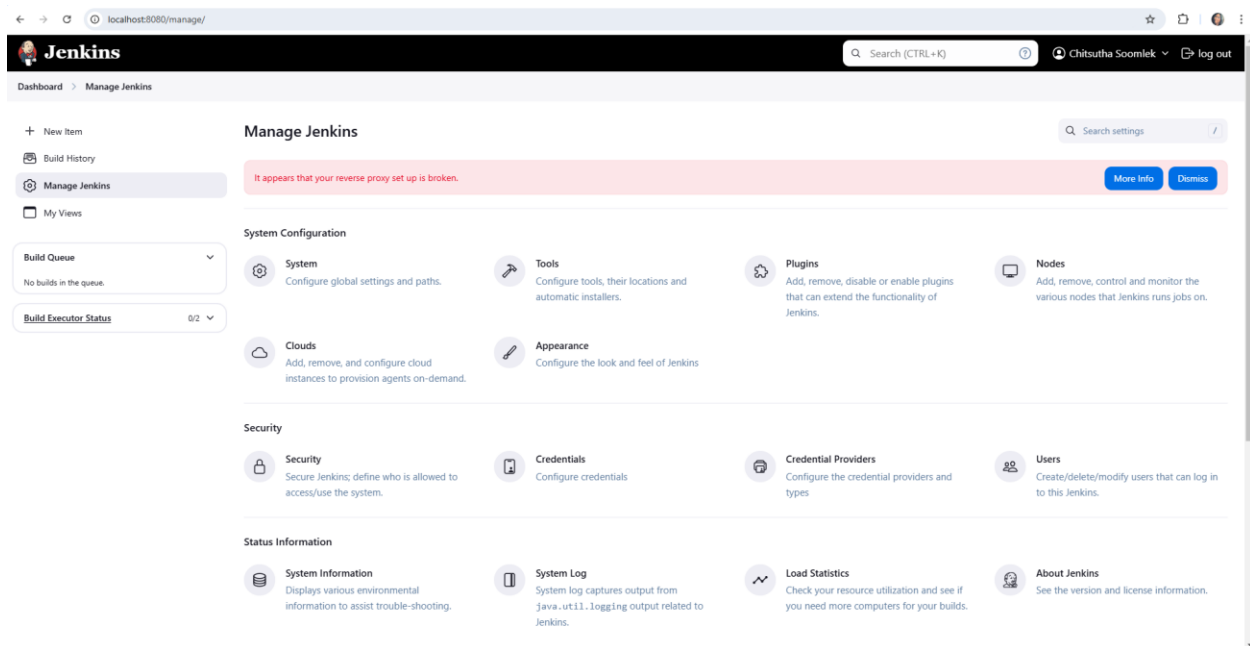
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

Lab Worksheet

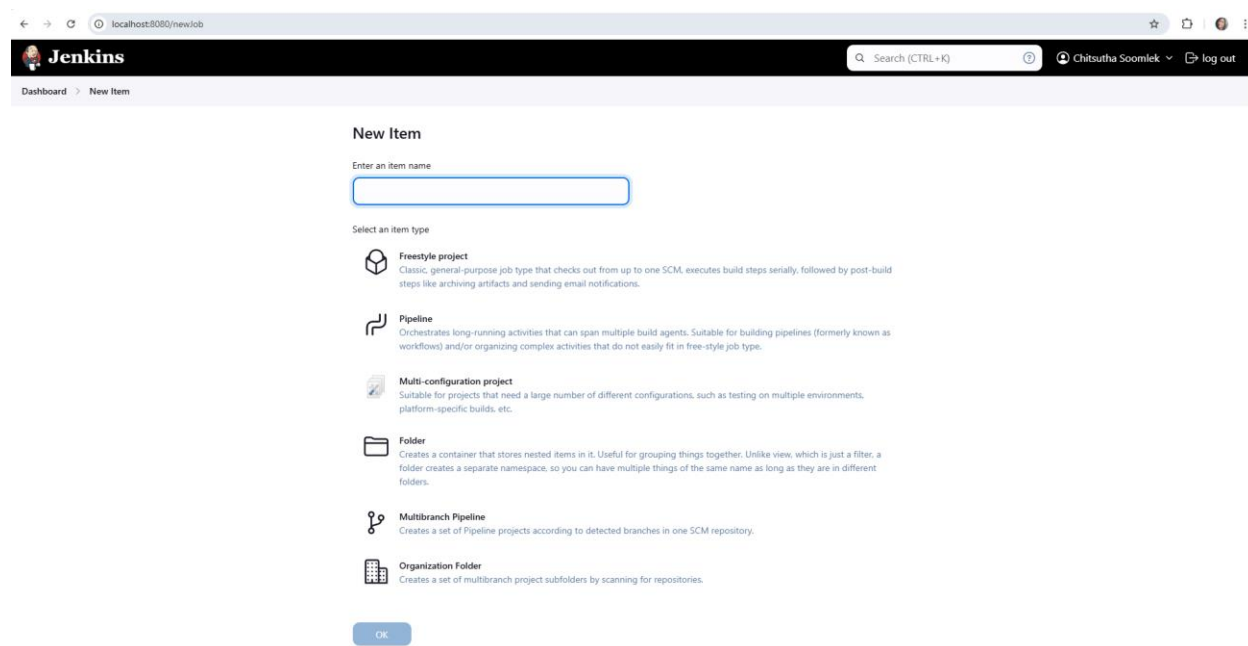


10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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

Lab Worksheet



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

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

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

 **Jenkins**

Search (CTRL+K) ?

waranya hommang ▾

Dashboard > UAT > Configuration

Configure

General

Source Code Management


Build Triggers

Build Environment

Build Steps

Post-build Actions

General

Enabled 

Description

Lab 8.5

Plain text [Preview](#)

☐ Discard old builds ?

☒ GitHub project

Project url ?

https://github.com/waranya-hommuang/SE_lab7.git

Advanced ▾

☐ This project is parameterized ?

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/waranya-hommuang/SE_lab7.git

Credentials ?

- none - ▾

+ Add

Advanced ▾

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

Lab Worksheet

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 2:05:19 PM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 2:20:19 PM Coordinated Universal Time.

☐ GitHub hook trigger for GITScm polling ?☐ Poll SCM ?

Build Steps

≡ Execute shell ?

Command

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

robot test_invalid_form.robot

Advanced ▾

Add build step ▾

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

robot test_invalid_form.robot

Post-build action: เพิ่ม Publish Robot Framework test results ->

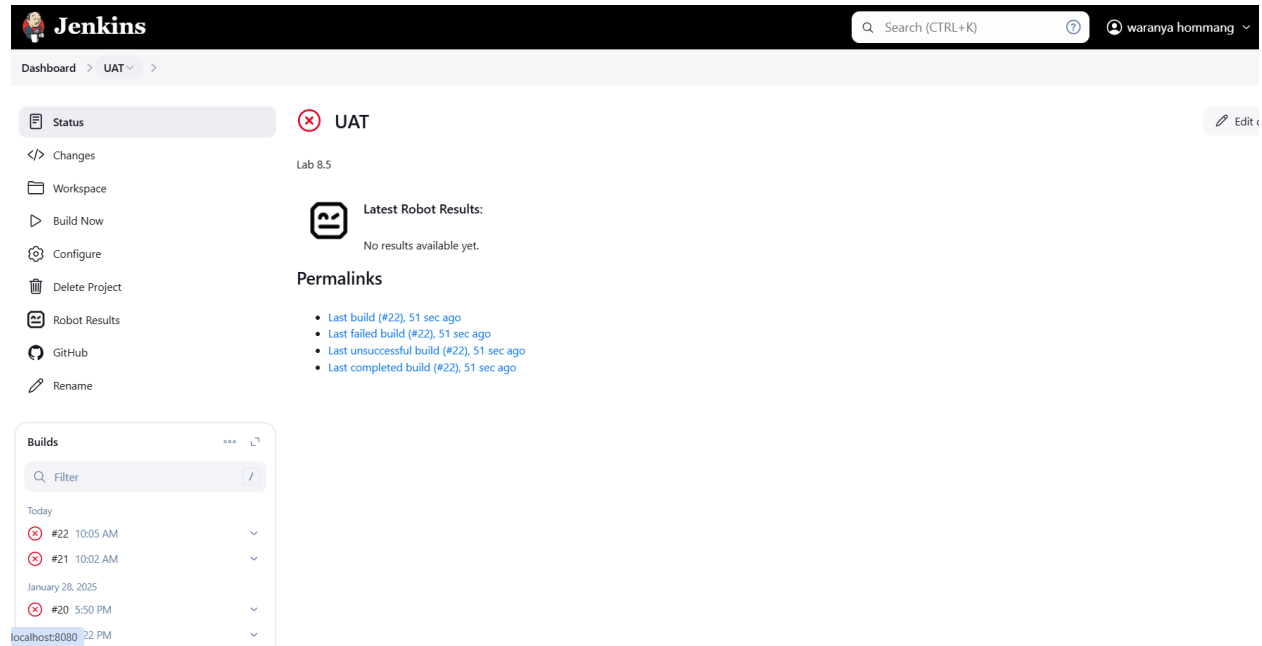
ระบุไดเรกทอรีที่เก็บไฟล์ผลการทดสอบโดย Robot framework ในรูป xml และ html -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ไม่ผ่านแล้วนับว่าซอฟต์แวร์มีปัญหา -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ผ่านแล้วนับว่าซอฟต์แวร์มีอยู่ในสถานะที่สามารถนำไปใช้งานได้ (เช่น 20, 80)

13. กด Apply และ Save

14. คลิก Build Now

Lab Worksheet

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



Jenkins Search (CTRL+K) waranya hommang

Dashboard > UAT >

Status **UAT** Edit

</> Changes
Workspace
Build Now
Configure
Delete Project
Robot Results
GitHub
Rename

Lab 8.5

Latest Robot Results:
No results available yet.

Permalinks

- Last build (#22), 51 sec ago
- Last failed build (#22), 51 sec ago
- Last unsuccessful build (#22), 51 sec ago
- Last completed build (#22), 51 sec ago

Builds

Filter /

Today

- #22 10:05 AM
- #21 10:02 AM

January 28, 2025

- #20 5:50 PM

localhost:8080 22 PM

Console Output Download Copy

```

Started by user waranya hommang
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/waranya-hommuang/SE_lab7.git # timeout=10
Fetching upstream changes from https://github.com/waranya-hommuang/SE_lab7.git
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/waranya-hommuang/SE_lab7.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision b85a2e54d33f1719121a711d9296e0d506cfd868 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f b85a2e54d33f1719121a711d9296e0d506cfd868 # timeout=10
Commit message: "update lab7"
> git rev-list --no-walk b85a2e54d33f1719121a711d9296e0d506cfd868 # timeout=10
[UAT] $ /bin/sh -xe /tmp/jenkins12931071149174380563.sh
+ robot test_invalid_form.robot
=====
Test Invalid Form
=====
Verify Form Page Display :: Test Case 1: Verify that the form page... | FAIL |
Parent suite setup failed:
WebDriverException: Message: 'chromedriver.exe' executable may have wrong permissions.
-----
Test Empty Destination :: Test Case: Verify error when Destination... | FAIL |
Parent suite setup failed:
WebDriverException: Message: 'chromedriver.exe' executable may have wrong permissions.
  
```

Lab Worksheet

```

6 tests, 0 passed, 6 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
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:
ERROR: Build step failed with exception
/var/jenkins_home/workspace/UAT/output.xml is not a directory.
    at org.apache.tools.ant.types.AbstractFileSet.getDirectoryScanner(AbstractFileSet.java:518)
    at org.apache.tools.ant.types.AbstractFileSet.getDirectoryScanner(AbstractFileSet.java:489)
    at PluginClassLoader for robot//hudson.plugins.robot.RobotParser$RobotParserCallable.invoke(RobotParser.java:76)
    at PluginClassLoader for robot//hudson.plugins.robot.RobotParser$RobotParserCallable.invoke(RobotParser.java:52)
    at hudson.FilePath.act(FilePath.java:1234)
    at hudson.FilePath.act(FilePath.java:1217)
    at PluginClassLoader for robot//hudson.plugins.robot.RobotParser.parse(RobotParser.java:48)
    at PluginClassLoader for robot//hudson.plugins.robot.RobotPublisher.parse(RobotPublisher.java:262)
    at PluginClassLoader for robot//hudson.plugins.robot.RobotPublisher.perform(RobotPublisher.java:286)
    at hudson.tasks.BuildStepCompatibilityLayer.perform(BuildStepCompatibilityLayer.java:80)
    at hudson.tasks.BuildStepMonitor$1.perform(BuildStepMonitor.java:20)
    at hudson.model.AbstractBuild$AbstractBuildExecution.perform(AbstractBuild.java:818)
    at hudson.model.AbstractBuild$AbstractBuildExecution.performAllBuildSteps(AbstractBuild.java:767)
    at hudson.model.Build$BuildExecution.post2(Build.java:179)
    at hudson.model.AbstractBuild$AbstractBuildExecution.post(AbstractBuild.java:711)
    at hudson.model.Run.execute(Run.java:1854)
    at hudson.model.FreeStyleBuild.run(FreeStyleBuild.java:44)
    at hudson.model.ResourceController.execute(ResourceController.java:101)
    at hudson.model.Executor.run(Executor.java:445)
Build step 'Publish Robot Framework test results' marked build as failure
Finished: FAILURE

```