

Lab Worksheet

ชื่อ-นามสกุล นายธนธรณ์ จุฬลาย รหัสนักศึกษา 653380131-7 Section 2

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

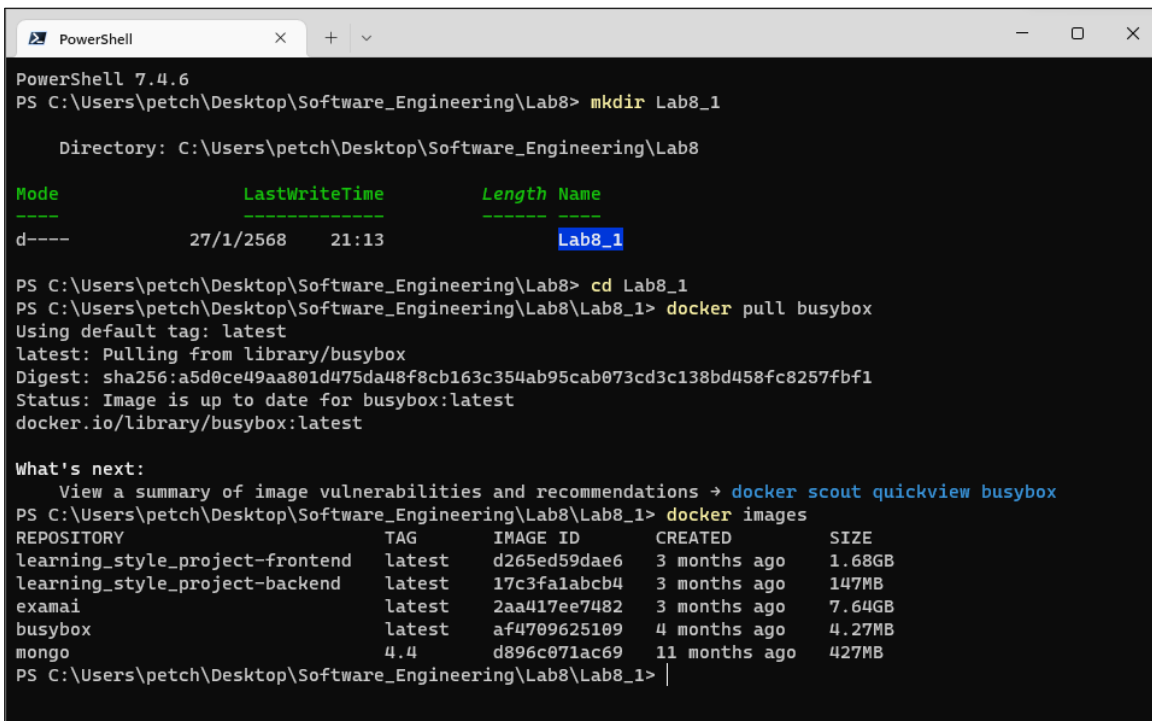
1. ติดตั้ง Docker desktop ลงบนเครื่องคอมพิวเตอร์ โดยดาวน์โหลดจาก <https://www.docker.com/get-started>
2. สร้าง Account บน Docker hub (<https://hub.docker.com/signup>)
3. กำหนดให้ \$ หมายถึง Command prompt และ <> หมายถึง ให้ป้อนค่าของพารามิเตอร์ที่กำหนด

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 สำหรับกรณีที่เกิดปัญหา 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 พร้อมกับตอบคำถามต่อไปนี้



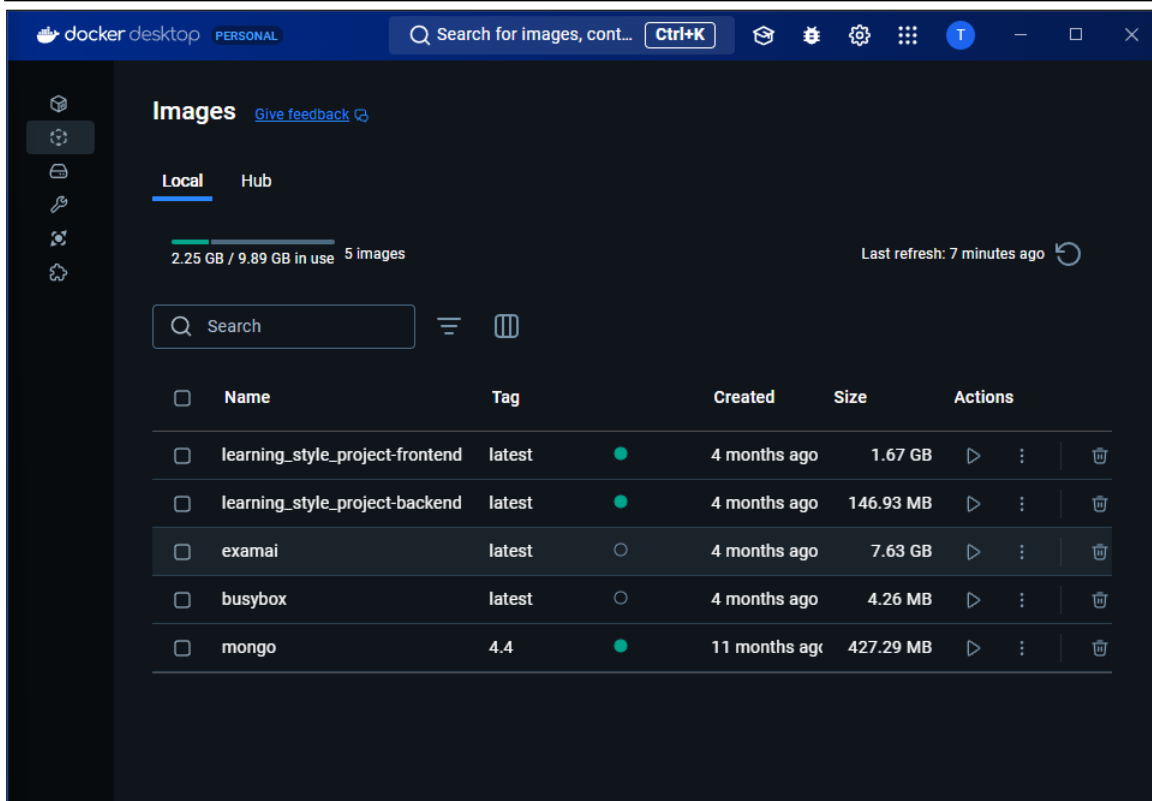
```
PowerShell 7.4.6
PS C:\Users\petch\Desktop\Software_Engineering\Lab8> mkdir Lab8_1

Directory: C:\Users\petch\Desktop\Software_Engineering\Lab8

Mode                LastWriteTime         Length Name
----                -
d-----           27/1/2568      21:13         Lab8_1

PS C:\Users\petch\Desktop\Software_Engineering\Lab8> cd Lab8_1
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab073cd3c138bd458fc8257fbf1
Status: Image is up to date for busybox:latest
docker.io/library/busybox:latest

What's next:
  View a summary of image vulnerabilities and recommendations → docker scout quickview busybox
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker images
REPOSITORY              TAG         IMAGE ID      CREATED        SIZE
learning_style_project-frontend  latest     d265ed59dae6  3 months ago  1.68GB
learning_style_project-backend  latest     17c3falabcb4  3 months ago  147MB
examai                   latest     2aa417ee7482  3 months ago  7.64GB
busybox                  latest     af4709625109  4 months ago  4.27MB
mongo                    4.4        d896c071ac69  11 months ago 427MB
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> |
```



Lab Worksheet

(1) สิ่งที่อยู่ภายใต้คอนเทนเนอร์ Repository คืออะไร

ตอบ ชื่อของอิมเมจ (Image) บน Docker ซึ่งใช้ระบุโปรเจกต์หรือแอปพลิเคชันที่อิมเมจนั้นถูกสร้างขึ้น เช่น learning_style_project-frontend, mongo หรือ busybox เป็นต้น

(2) Tag ที่ใช้บ่งบอกถึงอะไร

ตอบ Tag ใช้บ่งบอกถึงเวอร์ชันของอิมเมจ หรือสถานะของอิมเมจนั้น เช่น latest หมายถึงอิมเมจล่าสุด หรือหมายเลขเวอร์ชัน เช่น 4.4 เป็นต้น ซึ่งช่วยให้สามารถจัดการและเลือกใช้อิมเมจเวอร์ชันที่ต้องการได้ง่ายขึ้นใน Docker

5. ป้อนคำสั่ง \$ docker run busybox

6. ป้อนคำสั่ง \$ docker run -it busybox sh

7. ป้อนคำสั่ง ls

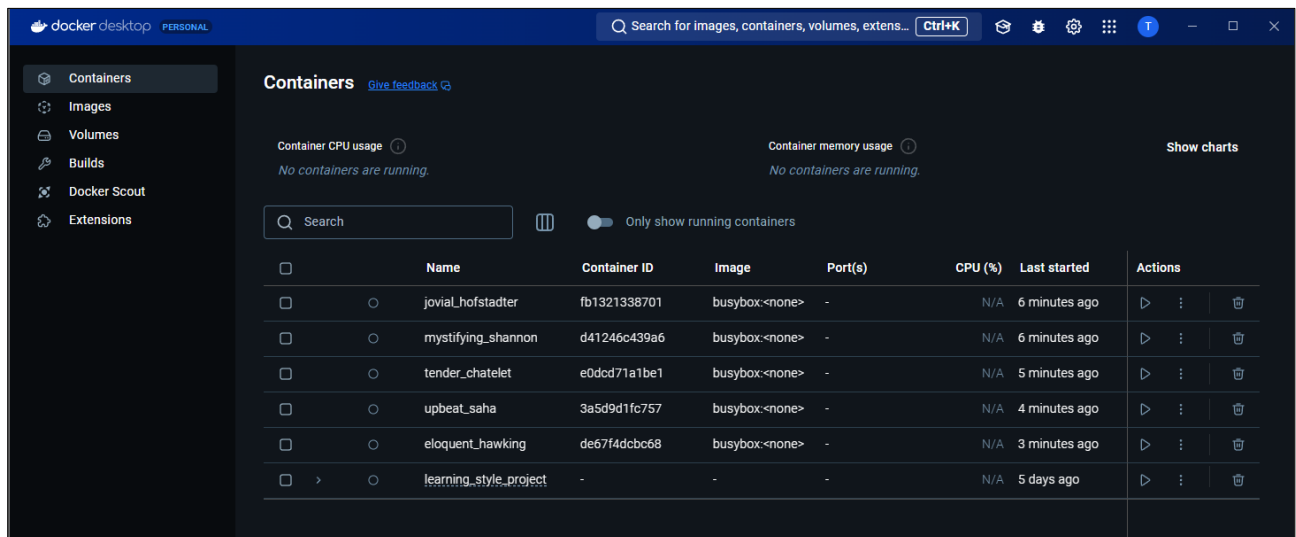
8. ป้อนคำสั่ง ls -la

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



Lab Worksheet

```

PowerShell
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker run busybox
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker run -it busybox sh
/ # ls
bin      dev      etc      home     lib      lib64    proc     root     sys      tmp      usr      var
/ # ls -la
total 48
drwxr-xr-x  1 root    root      4096 Jan 27 14:27 .
drwxr-xr-x  1 root    root      4096 Jan 27 14:27 ..
-rwxr-xr-x  1 root    root      0 Jan 27 14:27 .dockerenv
drwxr-xr-x  2 root    root     12288 Sep 26 21:31 bin
drwxr-xr-x  5 root    root      360 Jan 27 14:27 dev
drwxr-xr-x  1 root    root      4096 Jan 27 14:27 etc
drwxr-xr-x  2 nobody  nobody    4096 Sep 26 21:31 home
drwxr-xr-x  2 root    root      4096 Sep 26 21:31 lib
lrwxrwxrwx  1 root    root        3 Sep 26 21:31 lib64 -> lib
dr-xr-xr-x 244 root    root        0 Jan 27 14:27 proc
drwx----- 1 root    root      4096 Jan 27 14:27 root
dr-xr-xr-x 11 root    root        0 Jan 27 14:27 sys
drwxrwxrwt  2 root    root      4096 Sep 26 21:31 tmp
drwxr-xr-x  4 root    root      4096 Sep 26 21:31 usr
drwxr-xr-x  4 root    root      4096 Sep 26 21:31 var
/ # exit
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker run busybox echo "Hello Thanathorn Chulay from busybox"
Hello Thanathorn Chulay from busybox
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker ps -a
CONTAINER ID   IMAGE      PORTS      NAMES                                COMMAND                                CREATED        STATUS
de67f4dc6c68   busybox    -          eloquent_hawking                    "echo 'Hello Thanath..."           19 seconds ago Exited (0)
3a5d9d1fc757   busybox    -          upbeat_saha                          "sh"                                  2 minutes ago  Exited (0)
About a minute ago                                tender_chatelet
e0dcd71a1be1   busybox    -          mystifying_shannon                  "sh"                                  2 minutes ago  Exited (0)
2 minutes ago                                jovial_hofstadter
d41246c439a6   busybox    -          learning_style_project-frontend-1   "docker-entrypoint.s..."           3 months ago  Exited (1)
2 minutes ago                                learning_style_project-backend
fb1321338701   busybox    -          learning_style_project-backend-1    "uvicorn app.main:ap..."           3 months ago  Exited (0)
3 minutes ago                                learning_style_project-mongo-1
4c7343e0d128   learning_style_project-frontend-1
9346ebdf76cc   learning_style_project-backend-1
41f183a78ca8   mongo:4.4  -          learning_style_project-mongo-1      "docker-entrypoint.s..."           3 months ago  Exited (0)
5 days ago
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> |

```

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

ตอบ ทำให้คอนเทนเนอร์ที่ถูกสร้างขึ้นสามารถโต้ตอบกับผู้ใช้ได้ในลักษณะ Interactive และเปิด Terminal ภายในคอนเทนเนอร์ (TTY) เพื่อให้สามารถพิมพ์คำสั่งต่าง ๆ ได้เหมือนกับการใช้งานในระบบปฏิบัติการจริง ตัวอย่างเช่น การเรียกใช้งาน sh หรือ bash เพื่อสำรวจโครงสร้างไฟล์ในคอนเทนเนอร์หรือทดสอบการทำงานบางอย่างในสภาพแวดล้อมนั้น

Lab Worksheet

(2) คอด้มน์ STATUS จากกการรันค้ำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

ตอบ แสดงสถานะของคอนเทนเนอร์แต่ละตัว โดยข้อมูลในคอด้มน์นี้บอด้ว่าคอนเทนเนอร์กำลังทำงานอยู่ (Up) หรือหยุดทำงานแล้ว (Exited) พร้อมด้้วยระยะเวลาที่คอนเทนเนอร์หยุดหรือเริ่มทำงาน เช่น Exited (0) หมายถึงคอนเทนเนอร์หยุดทำงานเรียบร้อยโดยไม่มีข้อผิดพลาด (exit code = 0) และบอกเวลาที่เกิดการหยุดนั้น

12. ป้อนค้ำสั่ง \$ docker rm <container ID ที่ด้ต้องการลบ>

[Check point#3] Capture หน้าจอ (ทั้งหน้าด้่างและทุกหน้าด้่างที่เกี่ยวข้อง) แสดงผลลั้พธ์ที่ได้ในขันตอนที 13

The screenshot displays two windows. The top window is a PowerShell terminal showing a series of commands to remove Docker containers. The bottom window is the Docker Desktop application, showing the 'Containers' tab with a table of running containers.

PowerShell Terminal Output:

```
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker rm fb1321338701f36f780b6a833e21849
97f88a3306c46c1fad4d26f30638de404
fb1321338701f36f780b6a833e2184997f88a3306c46c1fad4d26f30638de404
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker rm d41246c439a6707cec78f29608e728f
354fc3527f696b071489b15801c80ec2b
d41246c439a6707cec78f29608e728f354fc3527f696b071489b15801c80ec2b
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker rm e0dcd71a1be151ee9a5506e2f1dd3cb
2cfbc89a43aa089d8e13259dd9d14d7e7
e0dcd71a1be151ee9a5506e2f1dd3cb2cfbc89a43aa089d8e13259dd9d14d7e7
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker rm 3a5d9d1fc757064df3c201690433719
6f9c9b4de38299676c3aacf1375bca457
3a5d9d1fc757064df3c2016904337196f9c9b4de38299676c3aacf1375bca457
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> docker rm de67f4dcbc687a0742e9fe247295f41
9119982e8da2104c1fb9f8c2846b558fe
de67f4dcbc687a0742e9fe247295f419119982e8da2104c1fb9f8c2846b558fe
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_1> |
```

Docker Desktop Interface:

The Docker Desktop window shows the 'Containers' tab. It includes a search bar, a toggle for 'Only running' containers, and a table of containers. The table has columns for Name, Container ID, Image, Port(s), and Actions. A single container named 'learning_style_project' is listed.

Name	Container ID	Image	Port(s)	Actions
learning_style_project	-	-	-	Play, Stop, Delete

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 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

Lab Worksheet

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

```

PowerShell
PS C:\Users\petch\Desktop\Software_Engineering\Lab8> cd Lab8_2
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_2> docker build -t lab8_2 .
[+] Building 0.1s (5/5) FINISHED                                docker:desktop-linu
X
=> [internal] load build definition from Dockerfile                                0.0
S
=> == transferring dockerfile: 264B                                                0.0
S
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior relat 0.0
S
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same sta 0.0
S
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior relat 0.0
S
=> [internal] load metadata for docker.io/library/busybox:latest                    0.0
S
=> [internal] load .dockerignore                                                    0.0
S
=> == transferring context: 2B                                                      0.0
S
=> [1/1] FROM docker.io/library/busybox:latest                                    0.0
S
=> exporting to image                                                                0.0
S
=> == exporting layers                                                              0.0
S
=> == writing image sha256:e2bfff7f38aead78a85eb202bc4a3d8df6c89478b368d6baa5c44753ea4c0ce35 0.0
S
=> == naming to docker.io/library/lab8_2                                            0.0
S

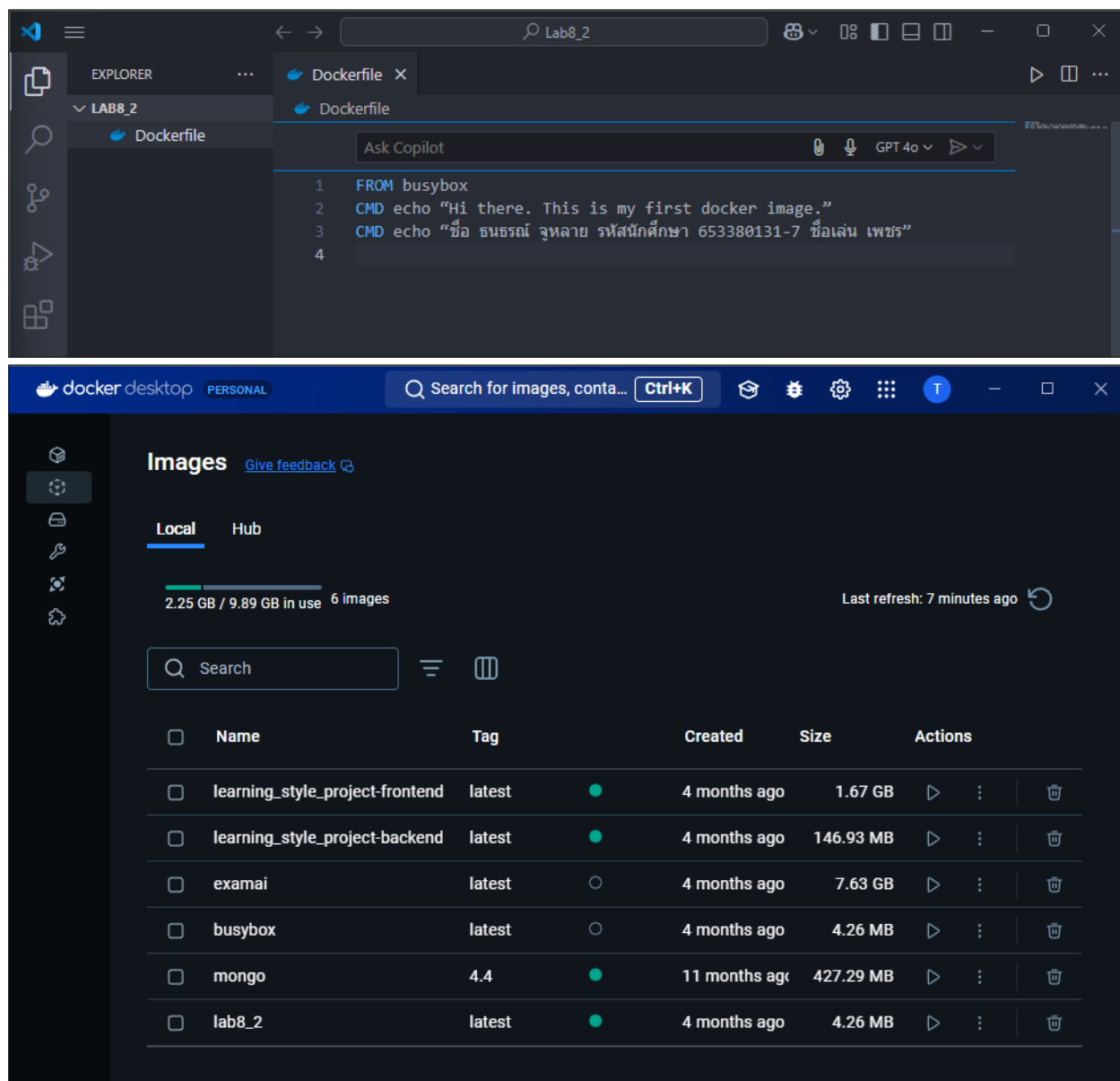
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/s210kvcjlbmy9w0h3tdn9pxr5

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

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_2> docker run lab8_2
“ ชื่อ นามสกุล จุฬาลงกรณ์มหาวิทยาลัย รหัสนักศึกษา 653380131-7 ชื่อเล่น เพชร ”
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_2> |

```

Lab Worksheet



(1) คำสั่งที่ใช้ในการ run คือ

ตอบ `docker run lab8_2`

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

ตอบ Option -t (หรือ --tag) ในคำสั่ง `docker build` ใช้สำหรับตั้งชื่อ (Name) และแท็ก (Tag) ให้กับ Docker Image ที่สร้างขึ้น เพื่อให้ง่ายต่อการระบุและเรียกใช้งานในภายหลัง ตัวอย่างเช่น การใช้ `-t lab8_2` จะตั้งชื่อ Image ว่า lab8_2 หากไม่ใช้ -t ระบบจะสร้าง Image แต่ไม่ได้ตั้งชื่อ ซึ่งจะทำให้ระบุ Image ได้ยาก (เพราะต้องใช้ Image ID ที่สร้างแบบสุ่ม)

Lab Worksheet

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

```
FROM busybox
```

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

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal และป้อนคำสั่งต่อไปนี้

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

```
FROM busybox
```

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

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

```
EOF
```

หรือใช้คำสั่ง

```
$ touch Dockerfile
```

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

7. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้


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


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

Lab Worksheet

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

The image shows two screenshots. The top screenshot is a PowerShell terminal window with the following content:

```

PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_2> cd ..
PS C:\Users\petch\Desktop\Software_Engineering\Lab8> mkdir Lab8_3

Directory: C:\Users\petch\Desktop\Software_Engineering\Lab8

Mode                LastWriteTime         Length Name
----                -
d-----          27/1/2568      22:09         Lab8_3

PS C:\Users\petch\Desktop\Software_Engineering\Lab8> cd Lab8_3
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_3> docker build -t thanathornkku/lab8 .
[+] Building 0.1s (5/5) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile             0.0s
=> => transferring dockerfile: 251B                             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
=> => exporting layers                                           0.0s
=> => writing image sha256:b16d268ef310de00a5f64f0ed87cedf1cd6288037ca74be2420a4368e3c0b621 0.0s
=> => naming to docker.io/thanathornkku/lab8                    0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/pidwaqfvbovp1bwkj34pqk2rt

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

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_3> docker run thanathornkku/lab8
“ชื่อ ธนธรณ์ จุฬาลาย รหัสนักศึกษา 653380131-7”
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_3>

```

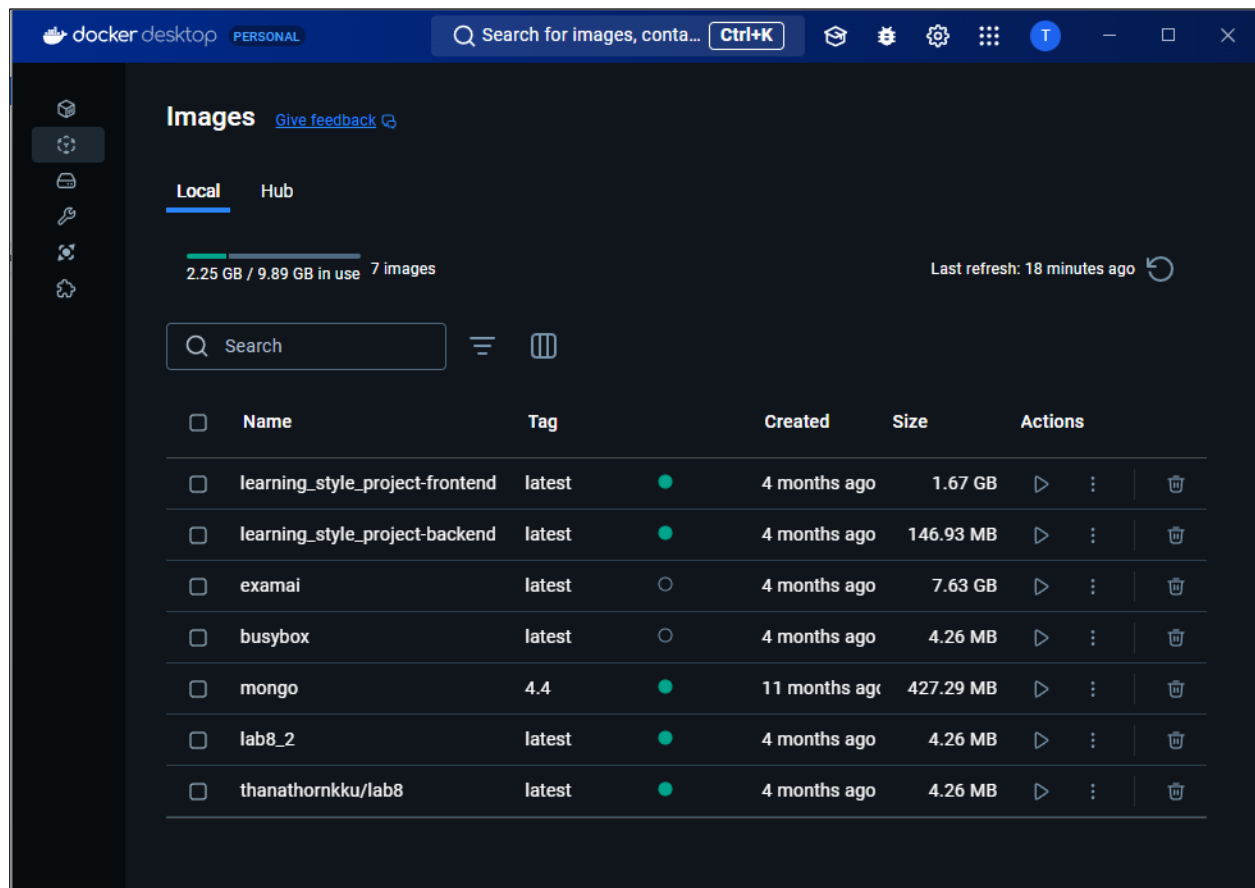
The bottom screenshot is a Visual Studio Code editor window showing the Dockerfile for Lab8_3:

```

Dockerfile
1 FROM busybox
2 CMD echo "Hi there. My work is done. You can run them from my Docker image."
3 CMD echo "ชื่อ ธนธรณ์ จุฬาลาย รหัสนักศึกษา 653380131-7"
4

```

Lab Worksheet



9. 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>
```

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

Lab Worksheet

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

The image shows three screenshots related to Docker image management:

PowerShell Terminal: A terminal window showing the command `docker push thanathornkku/lab8` being executed. The output indicates the image was pushed to the repository `docker.io/thanathornkku/lab8` with the tag `latest`. The image is based on `library/busybox` and has a size of 527 MB.

Docker Hub: A screenshot of the Docker Hub repository page for `thanathornkku/lab8`. The page shows the repository name, last pushed time (6 minutes ago), visibility (Public), and a status of Inactive.

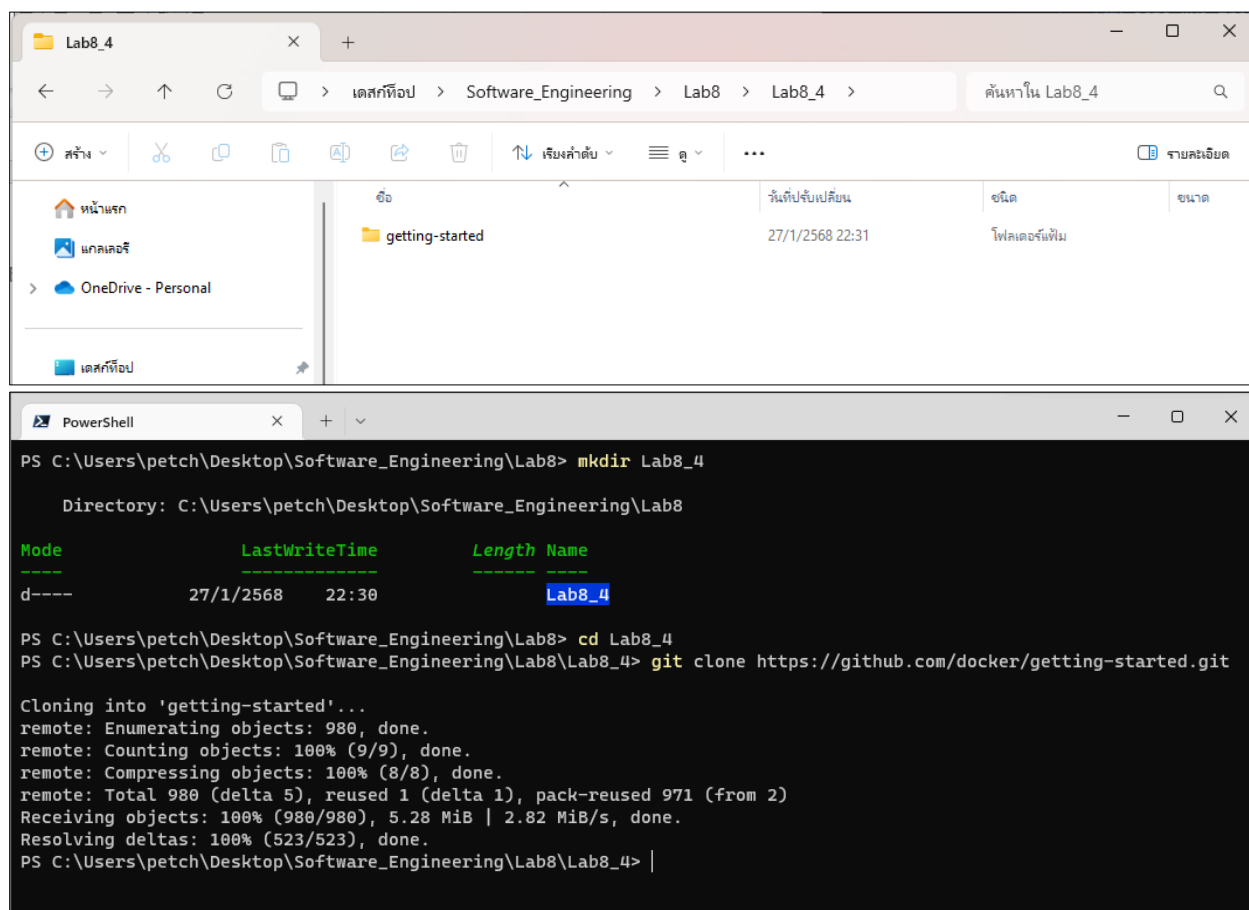
Docker Desktop: A screenshot of the Docker Desktop interface showing the `Images section. The Hub tab is selected, and the repository thanathornkku/lab8 is listed with the tag latest, OS linux, and a size of 2.15 MB.`

Lab Worksheet

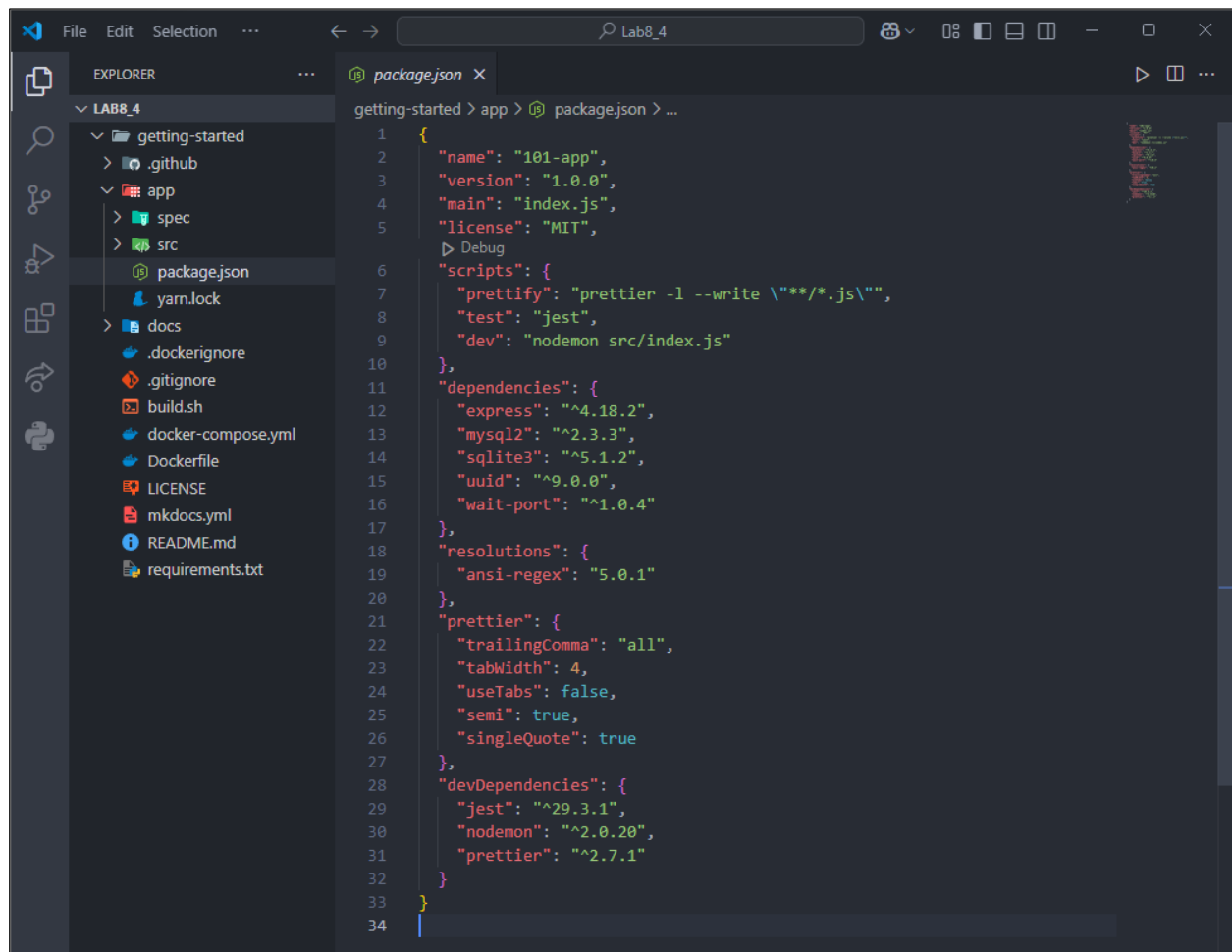
แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

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

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



Lab Worksheet



4. 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`
5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดใช้ชื่อ image เป็น `myapp_รหัสนศ. ไม่มีขีด`
`$ docker build -t <myapp_รหัสนศ. ไม่มีขีด> .`

Lab Worksheet

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ

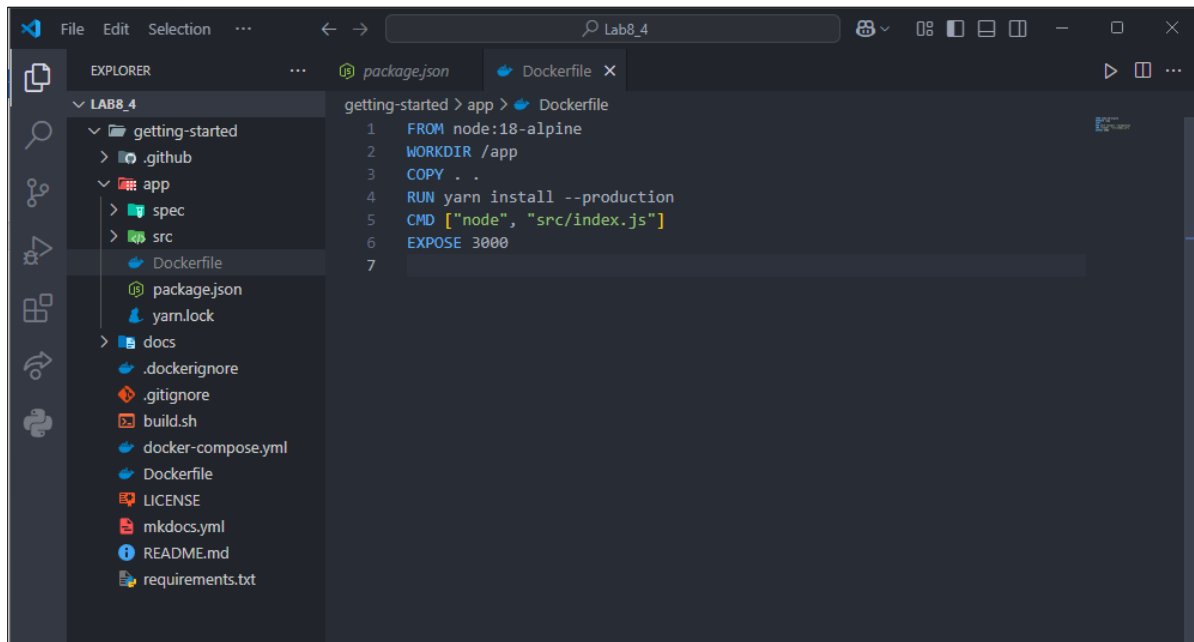
```

PowerShell
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4> cd getting-started/app
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533801317 .
[+] Building 44.0s (10/10) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 156B                                0.0s
=> [internal] load metadata for docker.io/library/node:18-alpine  4.2s
=> [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@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066b 21.8s
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066b 0.0s
=> => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066b3e274fbb25 7.67kB / 7.67kB 0.0s
=> => sha256:6e804119c3884fc5782795bf0d2adc89201c63105aeece8647b17a7bcebbc385e 1.72kB / 1.72kB 0.0s
=> => sha256:dcbf7b337595be6f4d214e4eed84f230eefe0e4ac03a50380d573e289b9e5e40 6.18kB / 6.18kB 0.0s
=> => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 3.64MB / 3.64MB 2.0s
=> => sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 40.01MB / 40.01MB 20.2s
=> => sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 1.26MB / 1.26MB 2.2s
=> => extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 0.1s
=> => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 444B / 444B 2.6s
=> => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 1.3s
=> => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 0.0s
=> => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 0.0s
=> [internal] load build context                                  0.4s
=> => transferring context: 4.62MB                                  0.4s
=> [2/4] WORKDIR /app                                           0.4s
=> [3/4] COPY . .                                              0.0s
=> [4/4] RUN yarn install --production                          16.6s
=> exporting to image                                           0.9s
=> => exporting layers                                           0.9s
=> => writing image sha256:3a413f4e0b8dc6a98a29beb1c6cb8f4eb06b8de9c554e731aaa121386a0b0760 0.0s
=> => naming to docker.io/library/myapp_6533801317              0.0s

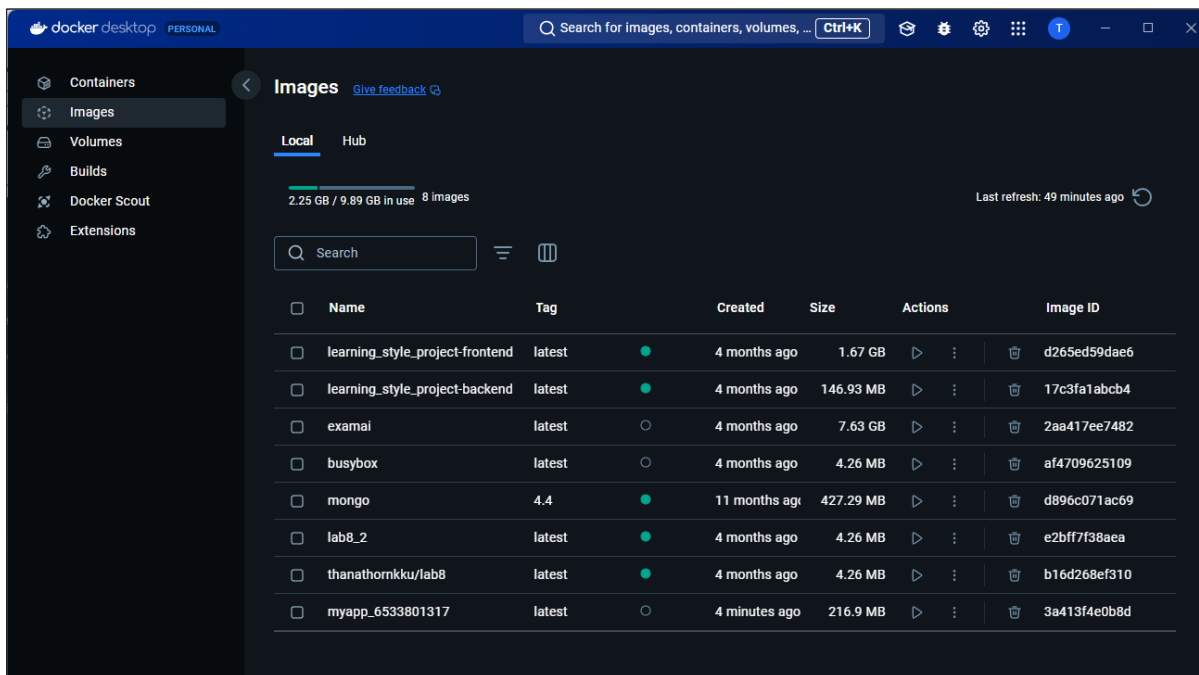
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/yfm24i3qbun9f4n235byyqzj1

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> |

```



Lab Worksheet

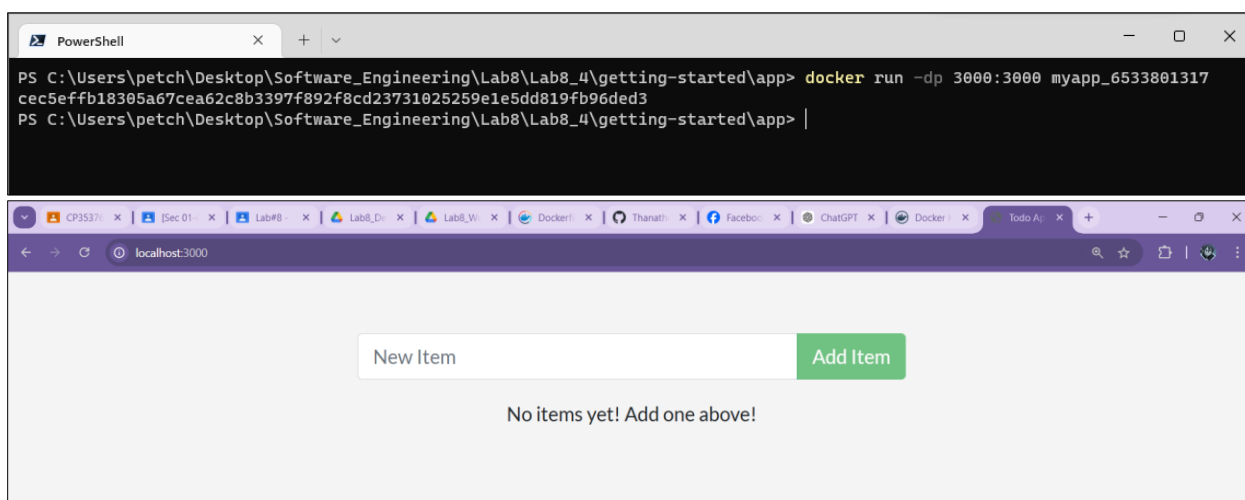


6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

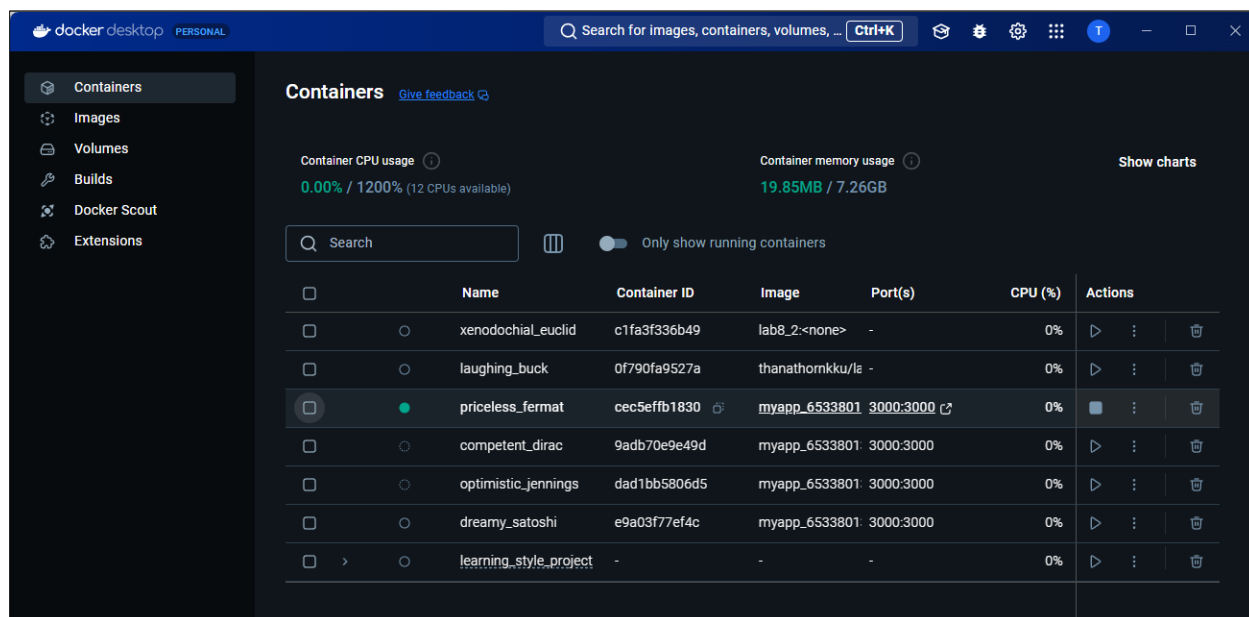
\$ docker run -dp 3000:3000 <myapp_รหัสศ. ไม่มีขีด>

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

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



Lab Worksheet



เนื่องจากก่อนหน้านี้ได้ลบบันทึกภาพใน Docker Desktop จึงติด Container เวอร์ชันที่แก้ไขแล้วมาด้วย

หมายเหตุ: นศ.สามารถทดลองเล่น Web application ที่ทำงานอยู่ได้

8. ทำการแก้ไข Source code ของ Web application ดังนี้

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

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

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

`ชื่อและนามสกุลของนักศึกษา</p>`

b. Save ไฟล์ให้เรียบร้อย

9. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

10. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

[Check point#10] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

The image shows a VS Code editor with a file explorer on the left displaying the project structure for 'Lab8_4'. The main editor shows the 'app.js' file with a React component 'TodoListCard' and a function 'AddItemForm'. The code includes state management for items and a form to add new items.

Below the editor is a PowerShell terminal window showing the execution of Docker commands. The first command is a 'docker run' command that fails due to port 3000 being occupied. The second command is a 'docker build' command that successfully builds the image 'myapp_6533801317'.

```

PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801317
ce5e5fffb18305a67cea62c8b3397f892f8cd23731025259e1e5dd819fb96ded3
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533801317 .
[+] Building 23.4s (10/10) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 156B                                              0.0s
=> [internal] load metadata for docker.io/library/node:18-alpine                  2.5s
=> [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@sha256:974afb6c0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fb 0.0s
=> [internal] load build context                                                  0.0s
=> => transferring context: 8.19kB                                                0.0s
=> CACHED [2/4] WORKDIR /app                                                      0.0s
=> [3/4] COPY . .                                                                0.1s
=> [4/4] RUN yarn install --production                                           20.0s
=> exporting to image                                                            0.9s
=> => exporting layers                                                            0.9s
=> => writing image sha256:3a97ad809589441167af87eebe7d9d4085c3d6b7db502c71ac852e30b1ededc6 0.0s
=> => naming to docker.io/library/myapp_6533801317                             0.0s

View build details: docker-desktop:///dashboard/build/desktop-linux/desktop-linux/ijzonki6v64zp3zdrwf7b5ehg

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801317
9adb70e9e49d1af99e3d610bdfaf3cbaaafd9a196ec9412f659ac9d78a8cbded
docker: Error response from daemon: driver failed programming external connectivity on endpoint competent_dirac (0747a1d32d1e4baaf9c6fbd3a76dbe4615b12b7a47772915d28b937d43cee4eb): Bind for 0.0.0.0:3000 failed: port is already allocated.
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801317
dad1b5806d5d4b85f10c6b0526c3dd4ed8fc5e30f0fa7d3316a632fbc3895ce
docker: Error response from daemon: driver failed programming external connectivity on endpoint optimistic_jennings (b257767dd6dc09ddfd5d647a09e0d7748148c6add098abc5723db30675ada35cd): Bind for 0.0.0.0:3000 failed: port is already allocated.
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801317
e9a03f77ef4c24b6c9e8b5641b13b3456cfa335596138748a33a8049ed78c508
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> |

```

Lab Worksheet

The screenshot shows the Docker Desktop interface. On the left, the 'Containers' tab is selected. The main area displays a list of containers with the following details:

Name	Container ID	Image	Port(s)	CPU (%)	Actions
xenodochiaLeucid	c1fa3f336b49	lab8_2:<none>	-	0%	[Stop] [Refresh] [Delete]
laughing_buck	0f790fa9527a	thanathornkku/la	-	0%	[Stop] [Refresh] [Delete]
priceless_fermat	cec5effb1830	myapp_6533801	3000:3000	0%	[Stop] [Refresh] [Delete]
competent_dirac	9adb70e9e49d	myapp_6533801	3000:3000	0%	[Stop] [Refresh] [Delete]
optimistic_jennings	dad1bb5806d5	myapp_6533801	3000:3000	0%	[Stop] [Refresh] [Delete]
dreamy_satoshi	e9a03f77ef4c	myapp_6533801	3000:3000	0.02%	[Stop] [Refresh] [Delete]
learning_style_project	-	-	-	0%	[Stop] [Refresh] [Delete]

Below the container list, a web browser window is open at localhost:3000. It shows a 'New Item' input field and an 'Add Item' button. Below these, a message states: 'There is no TODO item. Please add one to the list. By ชนธรณ์ จุฬหลาย'.

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

ตอบ หมายความว่า พอร์ต 3000 บนเครื่อง Host (0.0.0.0:3000) ถูกใช้งานอยู่แล้วโดย Container อื่น หรือแอปพลิเคชันอื่น ทำให้ Docker ไม่สามารถผูก (bind) พอร์ตนี้กับ Container ใหม่ได้ และเกิดขึ้นเพราะ Container ชื่อ priceless_fermat กำลังทำงานอยู่และใช้พอร์ต 3000:3000 ดังนั้นจำเป็นต้องหยุดการทำงานของ Container ชื่อ priceless_fermat ทำให้หลังจากนั้นสามารถใช้คำสั่ง run และแสดง Web application ที่แก้ไขแล้วได้

Lab Worksheet

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>

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

```

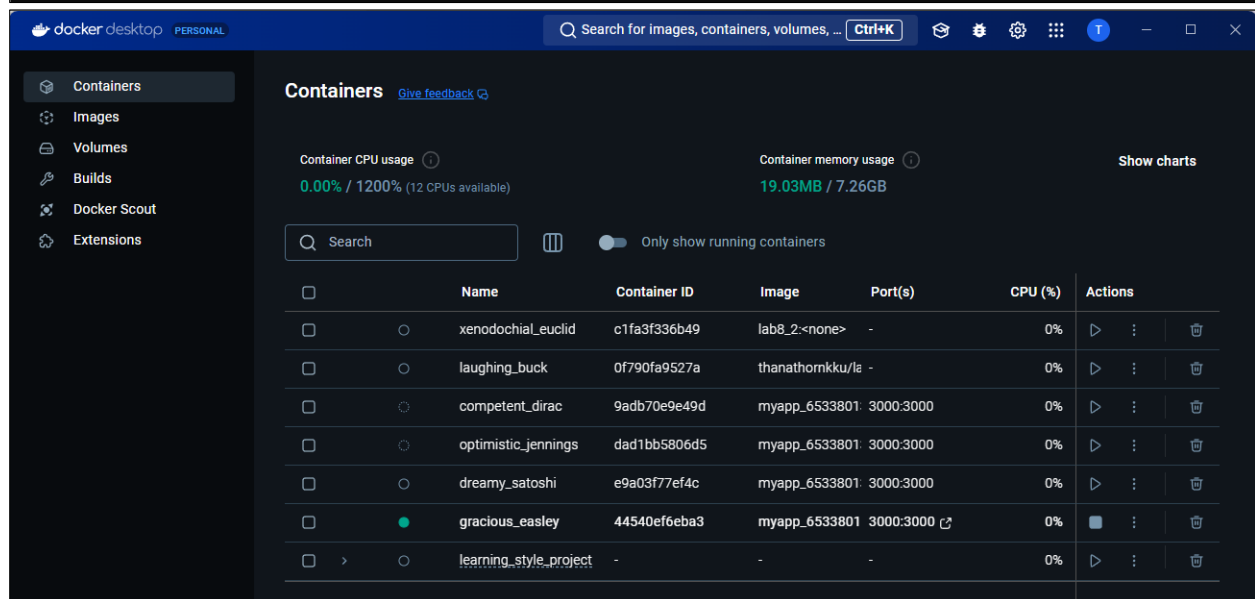
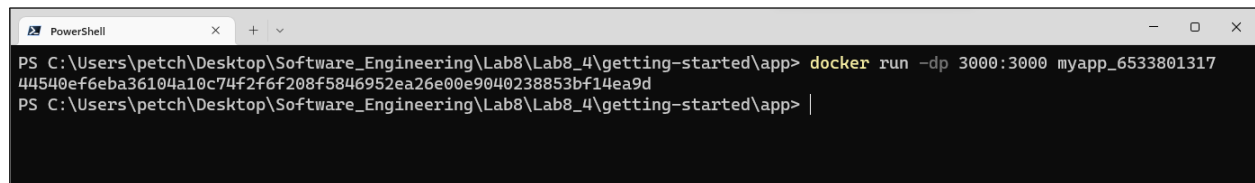
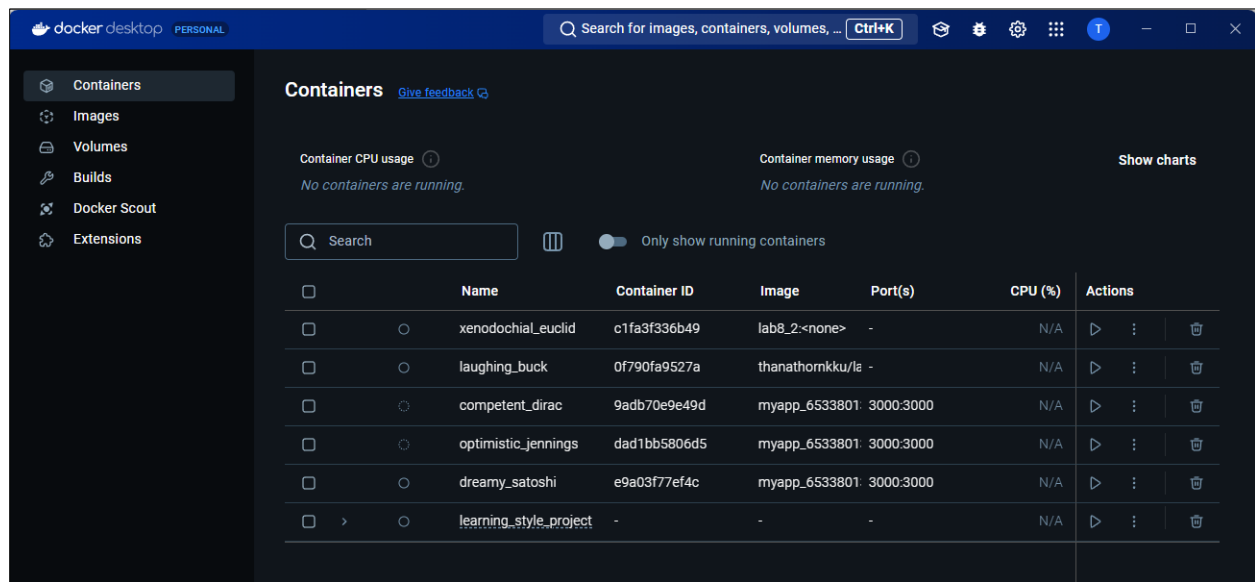
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
9a03777ef4c    myapp_6533801317  "docker-entrypoint.s..." 24 minutes ago Exited (0) 14 seconds ago
dad1bb5806d5    myapp_6533801317  "docker-entrypoint.s..." 25 minutes ago Created
9adb70e9e49d    myapp_6533801317  "docker-entrypoint.s..." 25 minutes ago Created
cec5effb18385a67cea62c8b3397f892f8cd23731025259e1e5dd819fb96ded3  "docker-entrypoint.s..." 33 minutes ago Exited (0) 8 minutes ago priceless-fermat
0f790fa9527a    thanathornkku/Lab8  "/bin/sh -c 'echo <u...  About an hour ago Exited (0) About an hour ago laughing_buck
c1fa3f336b49    lab8_2          "/bin/sh -c 'echo <u...  About an hour ago Exited (0) About an hour ago xenodochial_euclid
4c7343e0d128    learning_style_project-frontend  "docker-entrypoint.s..." 3 months ago Exited (1) 5 days ago learning_style_project-frontend-1
9346ebdf76cc    learning_style_project-backend  "uvicorn app.main:ap..." 3 months ago Exited (0) 5 days ago learning_style_project-backend-1
41f183a78ca8    mongo:4.4       "docker-entrypoint.s..." 3 months ago Exited (0) 5 days ago learning_style_project-mongo-1

PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker stop cec5effb18385a67cea62c8b3397f892f8cd23731025259e1e5dd819fb96ded3
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker rm cec5effb18385a67cea62c8b3397f892f8cd23731025259e1e5dd819fb96ded3
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app> docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
9a03777ef4c    myapp_6533801317  "docker-entrypoint.s..." 27 minutes ago Exited (0) 2 minutes ago
dad1bb5806d5    myapp_6533801317  "docker-entrypoint.s..." 27 minutes ago Created
9adb70e9e49d    myapp_6533801317  "docker-entrypoint.s..." 28 minutes ago Created
0f790fa9527a    thanathornkku/Lab8  "/bin/sh -c 'echo <u...  About an hour ago Exited (0) About an hour ago laughing_buck
c1fa3f336b49    lab8_2          "/bin/sh -c 'echo <u...  About an hour ago Exited (0) About an hour ago xenodochial_euclid
4c7343e0d128    learning_style_project-frontend  "docker-entrypoint.s..." 3 months ago Exited (1) 5 days ago learning_style_project-frontend-1
9346ebdf76cc    learning_style_project-backend  "uvicorn app.main:ap..." 3 months ago Exited (0) 5 days ago learning_style_project-backend-1
41f183a78ca8    mongo:4.4       "docker-entrypoint.s..." 3 months ago Exited (0) 5 days ago learning_style_project-mongo-1

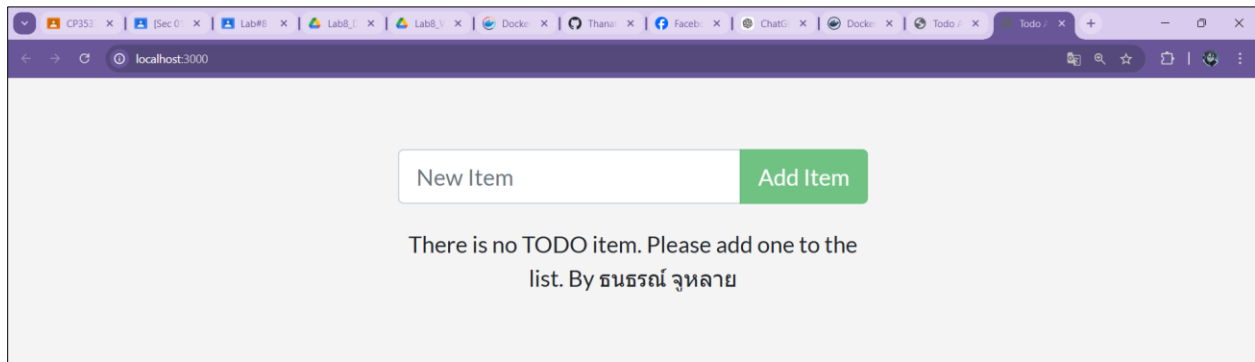
PS C:\Users\petch\Desktop\Software_Engineering\Lab8\Lab8_4\getting-started\app>

```

Lab Worksheet



Lab Worksheet



Lab Worksheet

แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

1. เปิด Command line หรือ Terminal บน Docker Desktop

2. ป้อนคำสั่งและทำการรัน container โดยผูกพอร์ต

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

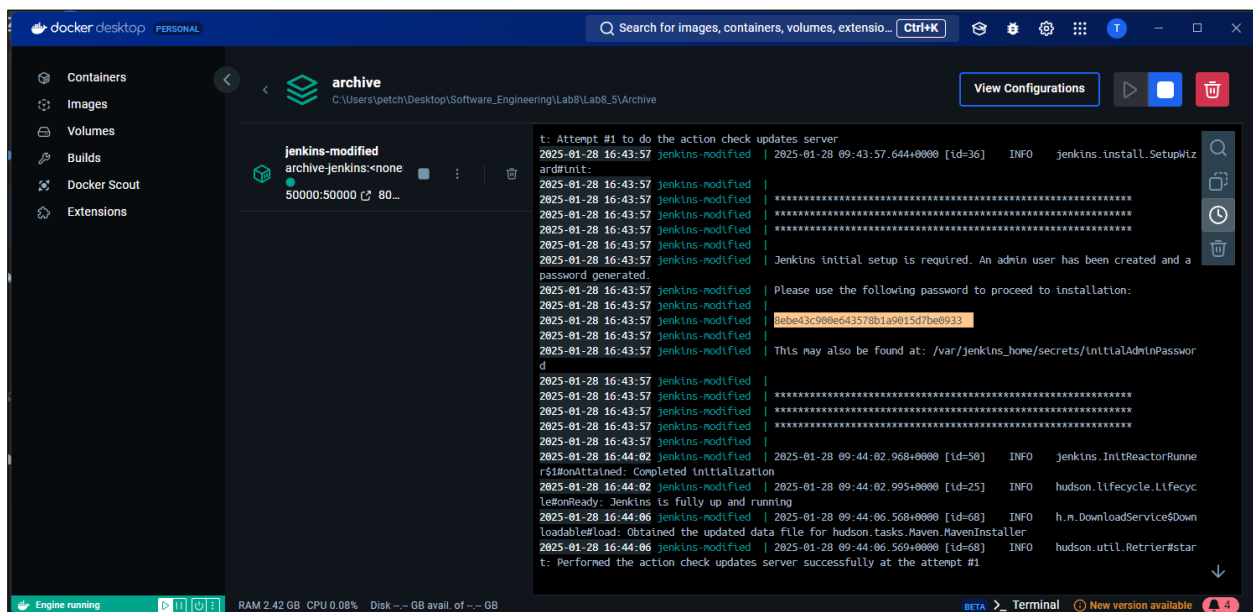
หรือ

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

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

3. บันทึกรหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก

[Check point#12] Capture หน้าจอที่แสดงผล Admin password



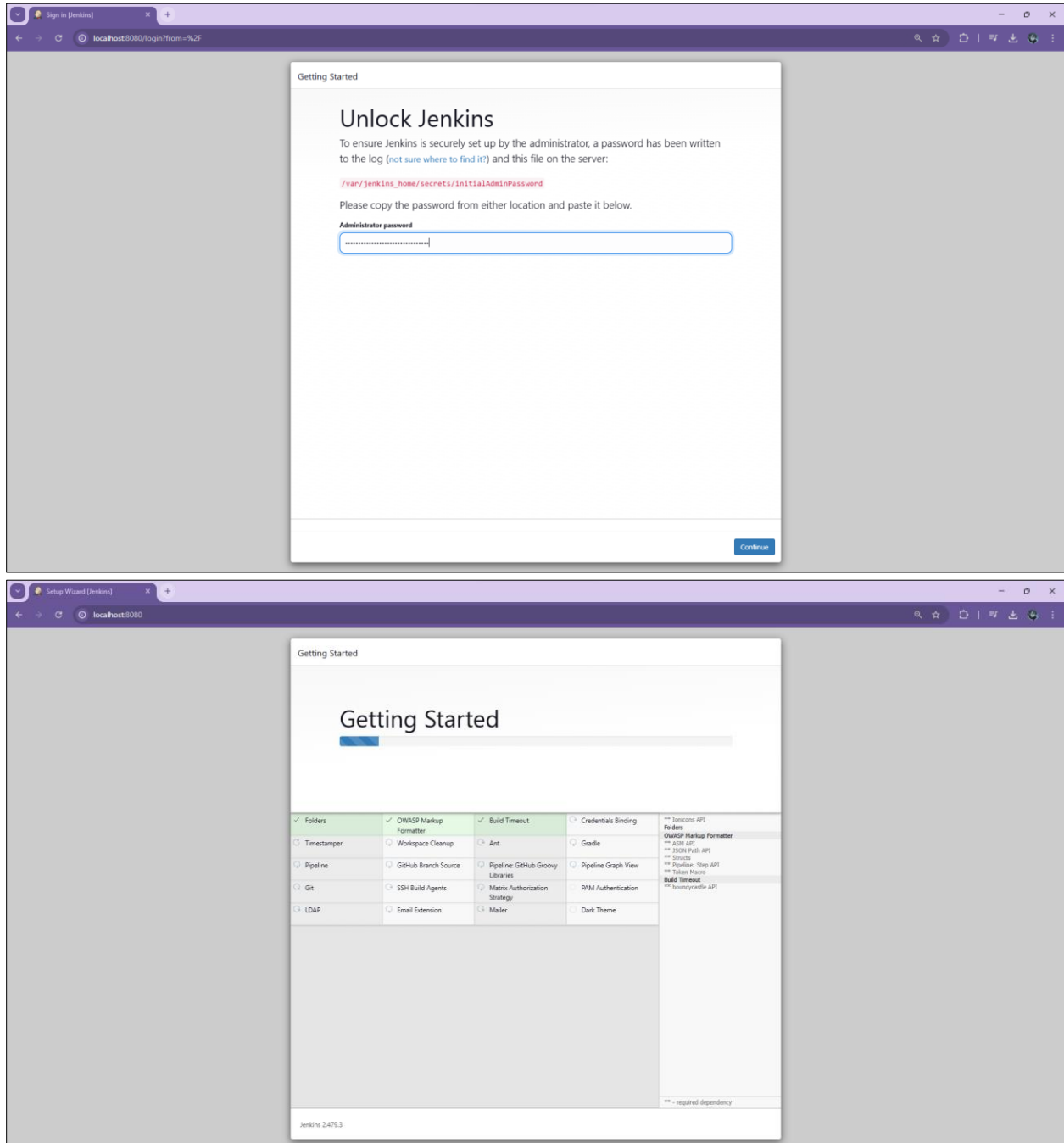
4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น localhost:8080

5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3

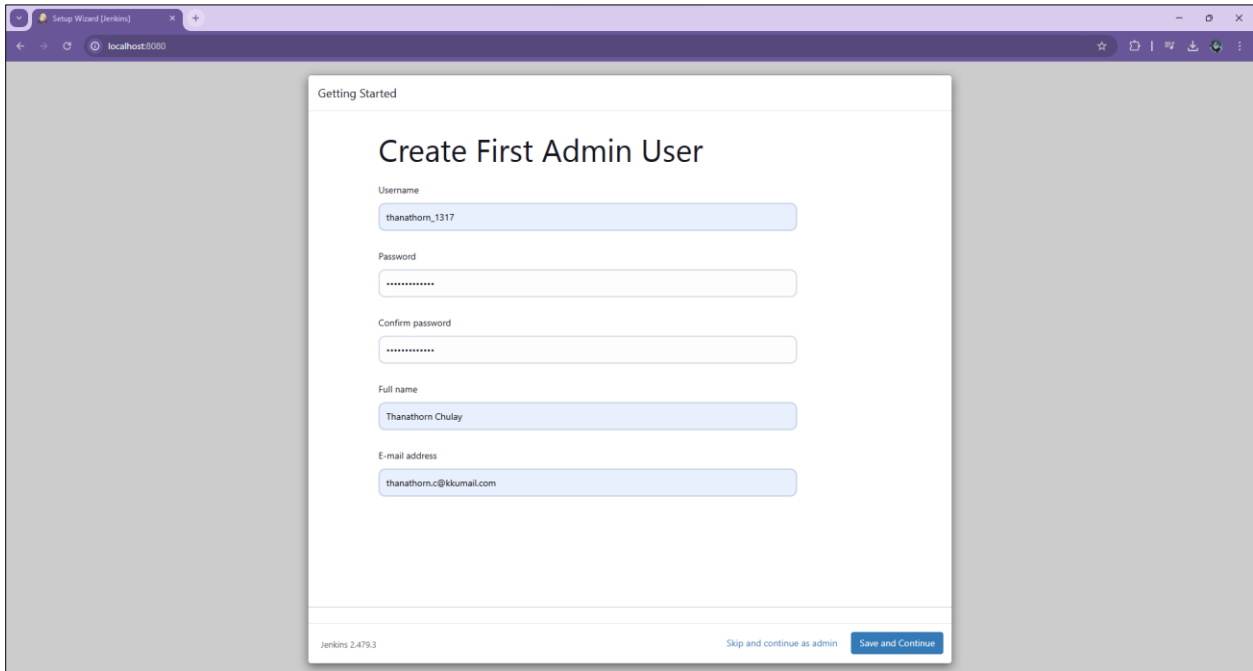
6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

Lab Worksheet

[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

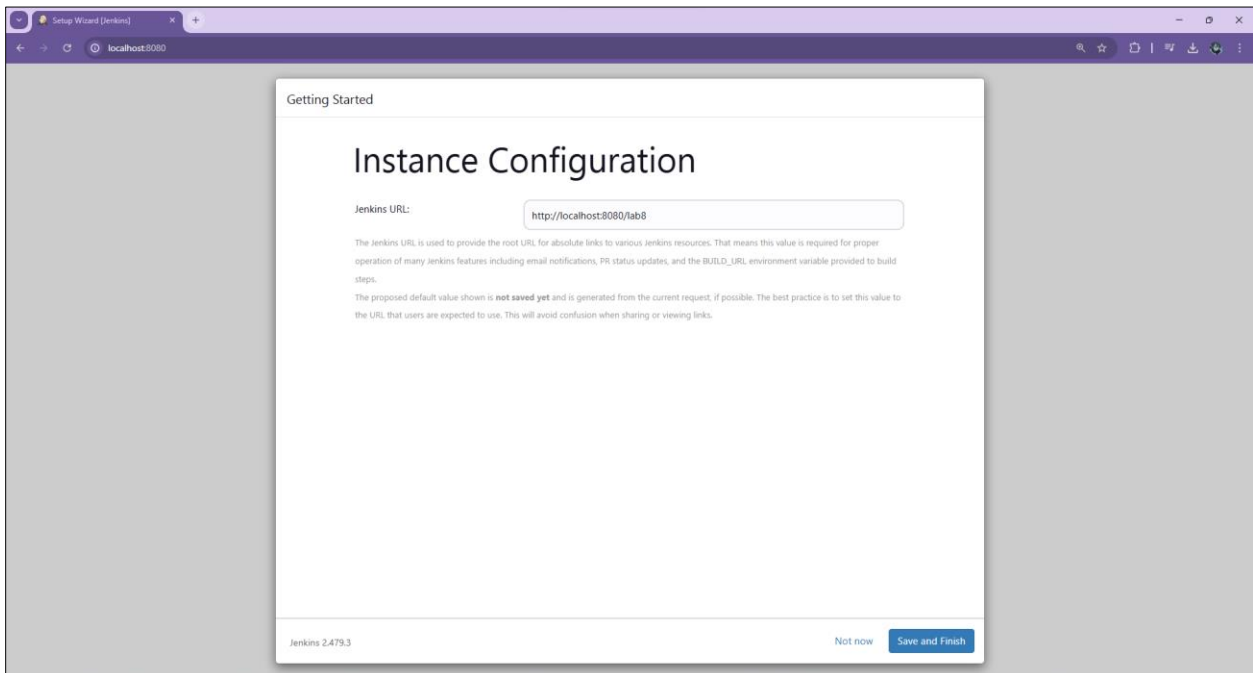


Lab Worksheet



The screenshot shows the 'Getting Started' section of the Jenkins Setup Wizard. The main heading is 'Create First Admin User'. Below this, there are five input fields: 'Username' (filled with 'thanathorn_1317'), 'Password' (masked with dots), 'Confirm password' (masked with dots), 'Full name' (filled with 'Thanathorn Chulay'), and 'E-mail address' (filled with 'thanathorn.c@kkumail.com'). At the bottom right, there are two buttons: 'Skip and continue as admin' and 'Save and Continue'.

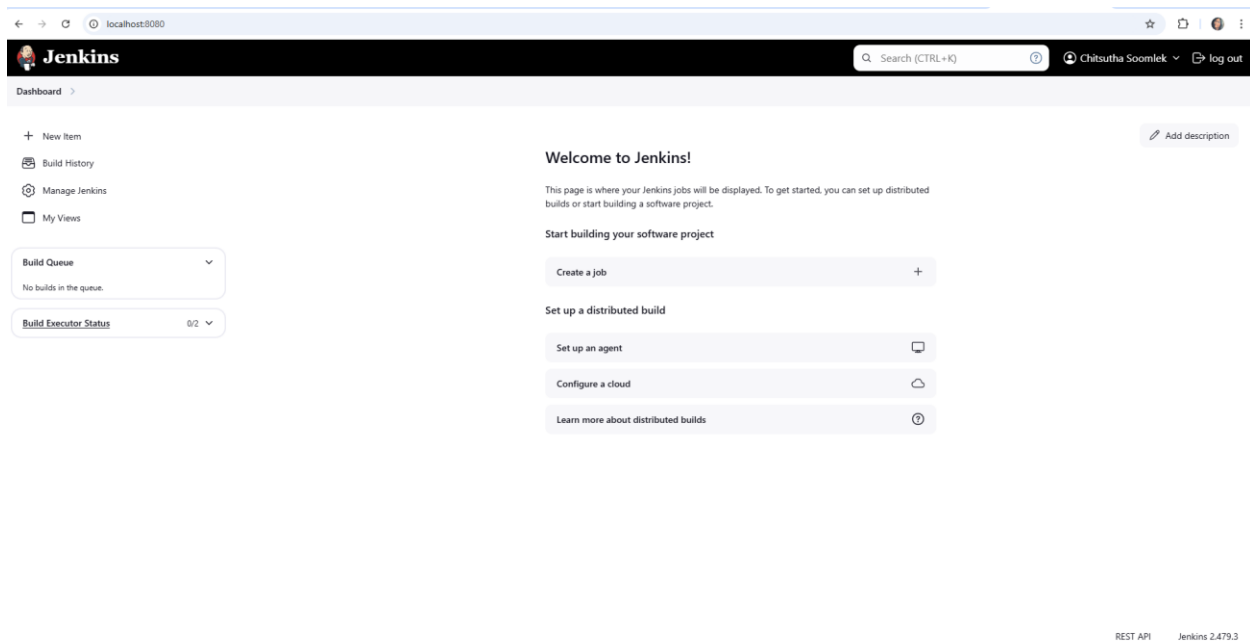
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>



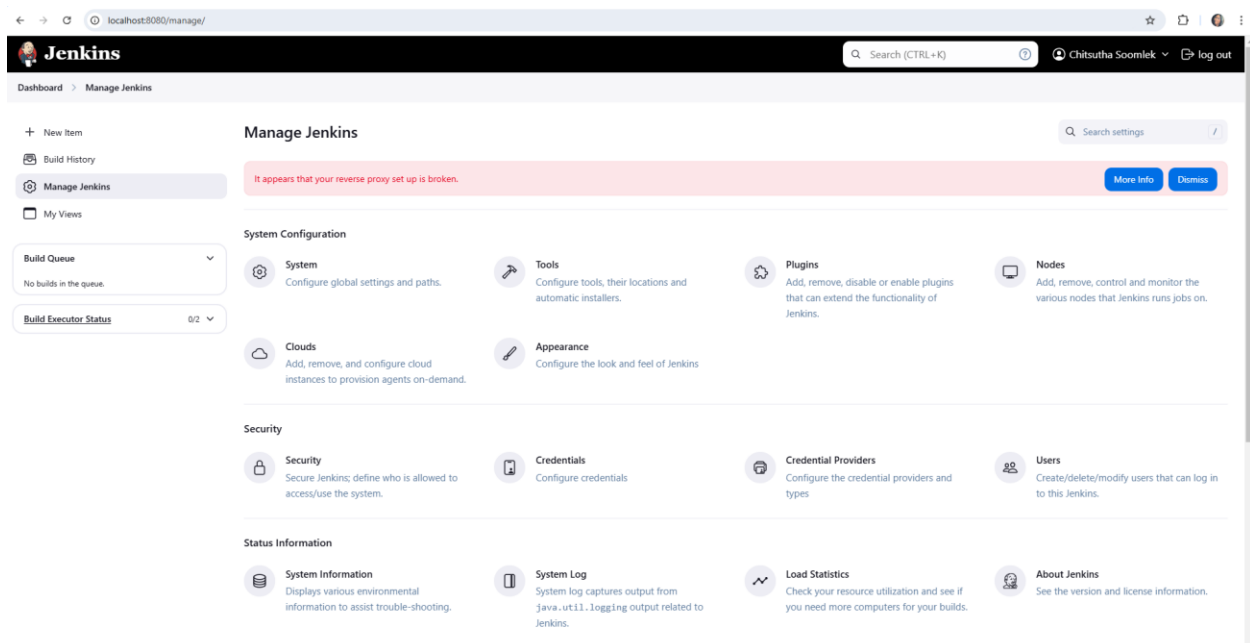
The screenshot shows the 'Getting Started' section of the Jenkins Setup Wizard. The main heading is 'Instance Configuration'. Below this, there is a 'Jenkins URL:' label followed by an input field containing 'http://localhost:8080/lab8'. Below the input field, there is a paragraph of text explaining the purpose of the Jenkins URL. At the bottom right, there are two buttons: 'Not now' and 'Save and Finish'.

Lab Worksheet

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

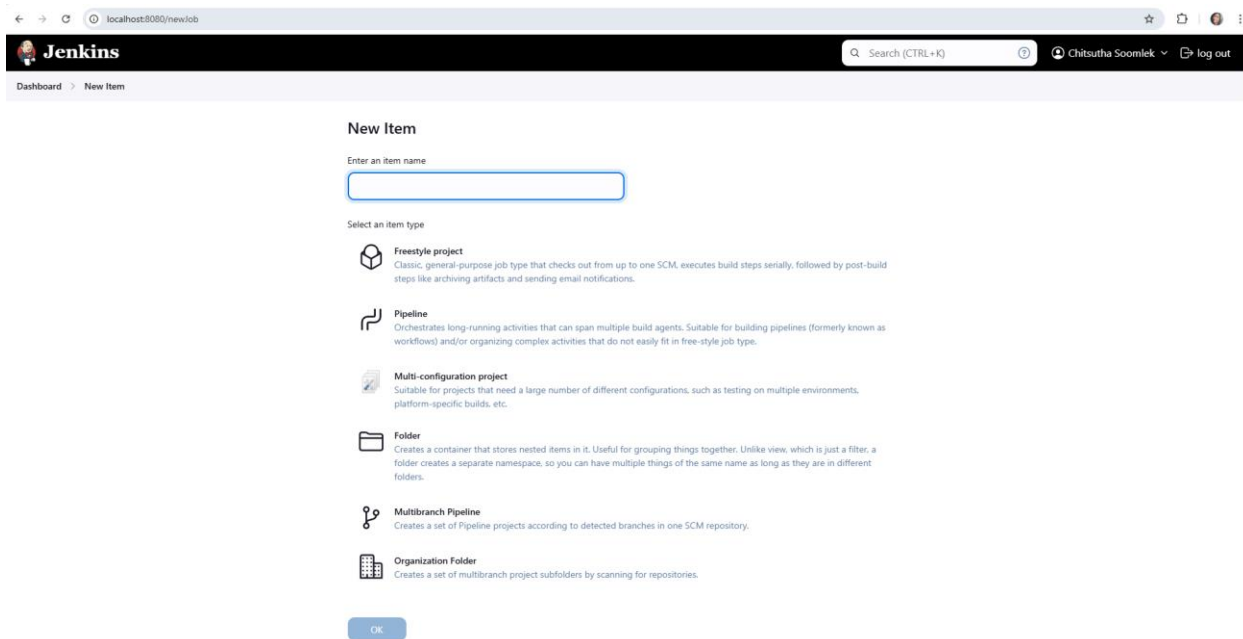


Lab Worksheet

10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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



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

Description: Lab 8.5

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

Build Trigger: เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

Lab Worksheet

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

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

The image displays two screenshots of the Jenkins web interface, specifically the configuration page for a job named 'UAT Config'.

Top Screenshot: General Tab

- Configuration:** General (Selected), Source Code Management, Build Triggers, Build Environment, Build Steps, Post-build Actions.
- General Tab:**
 - Description:** Lab 8.5
 - Plain text:** Preview
 - ☐ Discard old builds
 - ☒ GitHub project
 - Project url:** https://github.com/ThanathornKKU/Lab8_Jenkins
 - Advanced:**
 - ☐ This project is parameterized
 - ☐ Throttle builds
 - ☐ Execute concurrent builds if necessary
 - Buttons:** Save, Apply

Bottom Screenshot: Source Code Management Tab

- Configuration:** General, Source Code Management (Selected), Build Triggers, Build Environment, Build Steps, Post-build Actions.
- Source Code Management Tab:**
 - ☐ None
 - ☒ Git
 - Repositories:**
 - Repository URL:** https://github.com/ThanathornKKU/Lab8_Jenkins.git
 - Credentials:** - none -
 - Buttons:** + Add, Advanced
 - Add Repository**
 - Branches to build:**
 - Branch Specifier (blank for 'any'):** */main
 - Buttons:** Save, Apply

Lab Worksheet

The image displays two screenshots of the UAT Config (Jenkins) web interface, showing the configuration for a Jenkins job.

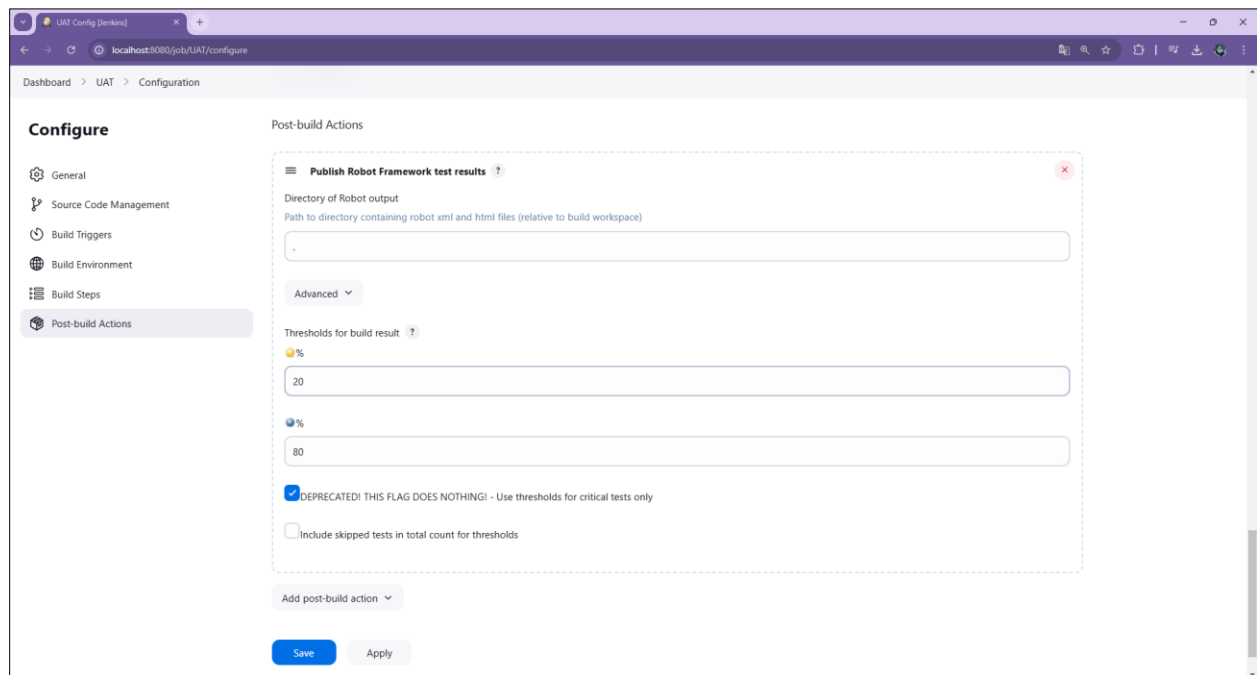
Top Screenshot: Build Triggers Configuration

- Repository browser:** (Auto)
- Additional Behaviours:** Add
- 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 10:05:14 AM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 10:20:14 AM Coordinated Universal Time.
 - ☐ GitHub hook trigger for GITScm polling
 - ☐ Poll SCM

Bottom Screenshot: Build Environment and Build Steps Configuration

- Build Environment:**
 - ☐ Delete workspace before build starts
 - ☐ Use secret text(s) or file(s)
 - ☐ Add timestamps to the Console Output
 - ☐ Inspect build log for published build scans
 - ☐ Terminate a build if it's stuck
 - ☐ With Ant
- Build Steps:**
 - Execute shell**
 - Command:** robot test_001_1.robot

Lab Worksheet



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

ตอบ `robot test_001_1.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. สั่ง Build Now

Lab Worksheet

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

The image displays two screenshots of the Jenkins web interface. The top screenshot shows the Jenkins dashboard for a pipeline named 'UAT'. The dashboard includes a sidebar with navigation options like Status, Changes, Workspace, Build Now, Configure, Delete Project, Robot Results, GitHub, and Rename. The main content area shows the 'UAT' pipeline status as 'Lab 8.5' with 'Latest Robot Results' indicating 'No results available yet'. A 'Builds' section lists two builds: '28 ม.ค. 2568' and '#1 10:12'. The bottom screenshot shows the details for build '#1 (28 ม.ค. 2568 10:12:26)'. It includes a 'Console Output' section, a 'Robot Test Summary' table, and a 'Changes' section. The 'Robot Test Summary' table shows 2 failed tests and 0 passed tests. The 'Changes' section indicates 'No changes'.

UAT Pipeline Dashboard

Dashboard > UAT

Status

UAT

Lab 8.5

Latest Robot Results:

No results available yet.

Permalinks

Builds

28 ม.ค. 2568

#1 10:12

REST API Jenkins 2.479.3

Build #1 Details

Dashboard > UAT > #1

Status

#1 (28 ม.ค. 2568 10:12:26)

Started by user Thanathorn Chulay

Started 33 sec ago
Took 11 sec

This run spent:

- 69 ms waiting;
- 11 sec build duration;
- 11 sec total from scheduled to completion.

Revision: 5888b1cc7d49f723b61b90afc702bd5aedebf7b

Repository: https://github.com/ThanathornKKU/Lab8_Jenkins.git

- refs/remotes/origin/main

Robot Test Summary:

Total	Failed	Passed	Skipped	Pass %
All tests	2	2	0	0.0

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

Changes

No changes.

REST API Jenkins 2.479.3

Lab Worksheet

Console Output

```

Started by user Thanathorn Chulay
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT
The recommended git tool is: NONE
No credentials specified
Cloning the remote git repository
Cloning repository https://github.com/ThanathornKKU/Lab8_Jenkins.git
> git init /var/jenkins_home/workspace/UAT # timeout=10
Fetching upstream changes from https://github.com/ThanathornKKU/Lab8_Jenkins.git
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/ThanathornKKU/Lab8_Jenkins.git # timeout=10
> git config remote.origin.url https://github.com/ThanathornKKU/Lab8_Jenkins.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 5888b1cc7d49f7723b61b90afc702bd5aede77b (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 5888b1cc7d49f7723b61b90afc702bd5aede77b # timeout=10
Commit message: "Add Lab8_SoftwareDeployment_053300131-7.pdf"
First time build. Skipping changelog.
[UAT] $ /bin/sh -xe /tmp/jenkins17051543499583467368.sh
+ robot test_001.1.robot
[ ERROR ] Error in file '/var/jenkins_home/workspace/UAT/test_001.1.robot' on line 2: Resource file '../test_resource.robot' does not exist.
[ ERROR ] Error in file '/var/jenkins_home/workspace/UAT/test_001.1.robot' on line 3: Importing library 'SeleniumLibrary' failed: ModuleNotFoundError: No module named 'SeleniumLibrary'
Traceback (most recent call last):
  None
PYTHONPATH:
  /usr/local/bin
  /usr/lib/python3.11.zip
  /usr/lib/python3.11
  /usr/lib/python3.11/lib-dynload
  /usr/local/lib/python3.11/dist-packages
  /usr/lib/python3.11/dist-packages

Test 001.1
-----
Open Form :: เปิดฟอร์มหน้า Form.html | FAIL |
No keyword with name 'Open Browser To Form Page' found.
-----
Record Success :: บันทึกความสำเร็จ Record Success หน้า Form.html | FAIL |
No keyword with name 'Open Browser To Form Page' found.
-----
Test 001.1 | FAIL |
2 tests, 0 passed, 2 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:
Done!
-Copying log files to build dir:
Done!
-Assigning results to build:
Done!
-Checking thresholds:
Done!
Done publishing Robot results.
Finished: FAILURE
  
```

REST API Jenkins 2.479.3

Lab Worksheet

