

Lab Worksheet

ชื่อ-นามสกุล พันณกร ภูงามเงิน รหัสนักศึกษา 653380022-2 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 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 พร้อมกับตอบคำถามต่อไปนี้

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker image ls
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
gradle	jdk17-alpine	bfff7e33f4a62	5 days ago	568MB
alpine/curl	latest	168480f55d3b	7 days ago	12.8MB
itzg/minecraft-server	java8-graalvm-ce	19b7529969dc	2 weeks ago	2.43GB
my_php	latest	b2c92511bc01	2 weeks ago	123MB
oven/bun	latest	bcd47cd7a9	2 weeks ago	222MB
jenkins/jenkins	lts-jdk17	44c1caefd796	2 weeks ago	468MB
node	alpine	31d93c95bd30	2 weeks ago	161MB
certbot/certbot	latest	fee8d78fa115	2 weeks ago	104MB
gcr.io/google.com/cloudsdktool/google-cloud-cli	stable	72a0a67b2973	2 weeks ago	571MB
redis	alpine	ee33180a8437	2 weeks ago	41.4MB
alpine/psql	latest	104df41c969a	3 weeks ago	12.7MB
gradle	jdk21-alpine	ee5bc57e6c75	5 weeks ago	592MB
postgres	alpine	3fe19d60f9ca	7 weeks ago	278MB
python	alpine	0a5bfb768070	7 weeks ago	44.9MB
node	18-alpine	87a8a7369c75	7 weeks ago	127MB
golang	alpine	27643ed1a129	7 weeks ago	246MB
nginx	latest	f876bfc1cc63	2 months ago	192MB
ppodgorsek/robot-framework	latest	03a01c8b5944	2 months ago	4.15GB
busybox	latest	af4709625109	4 months ago	4.27MB
migrate/migrate	latest	9ad5c7b72b43	4 months ago	61.1MB
maven	3-eclipse-temurin-21-alpine	ccc47f314a5e	5 months ago	378MB
maven	3-eclipse-temurin-17-alpine	d41c12640871	5 months ago	349MB
goodsmileduck/redis-cli	latest	0ed5b1e16de5	4 years ago	52.6MB
alpine/make	latest	63e39824de13	6 years ago	4.64MB

(1) สิ่งที่อยู่ภายใต้คอนเทนเนอร์ Repository คืออะไร ชื่อ image

(2) Tag ที่ใช้บ่งบอกถึงอะไร version ของ image

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

Lab Worksheet

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

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker run busybox
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker run -it busybox sh
/ # ls
bin      dev      etc      home    lib      lib64    proc    root    sys      tmp      usr      var
/ # ls -la
.         .         .dockerenv  dev      home    lib64    proc    root    sys      tmp      usr      var
..        bin      etc      lib      proc    sys      usr
/ # exit
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker run busybox echo "Hello Phannakon Phungamngoen from busybox"
Hello Phannakon Phungamngoen from busybox
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORT
a06b02a63fa7	busybox	"echo 'Hello Phannak..."	11 seconds ago	Exited (0) 8 seconds ago	
545aa5eaa93d	busybox	"sh"	About a minute ago	Exited (0) 56 seconds ago	
c3122c1df9c4	busybox	"sh"	About a minute ago	Exited (0) About a minute ago	
5ff2925ad60e	gradle:jdk17-alpine	"/_cacert_entrypoint..."	2 days ago	Up 2 days	0.0.
0.0:8080->8080/tcp	sweet_feynman				
9d9d45511a9e	redis:alpine	"docker-entrypoint.s..."	2 weeks ago	Exited (0) 2 weeks ago	
75c1eed67dfa	postgres:alpine	"docker-entrypoint.s..."	2 weeks ago	Exited (255) 5 days ago	5432
/tcp	postgresDB				

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
เปิดใช้งานหน้าต่างให้ User มี interaction กับ container ได้

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร
สถานะของ Container ว่ามีการรันอยู่หรือปิดไปแล้ว

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

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

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
a06b02a63fa7	busybox	"echo 'Hello Phannak..."	4 minutes ago	Exited (0) 4 minutes ago		jolly_jones
545aa5eaa93d	busybox	"sh"	5 minutes ago	Exited (0) 5 minutes ago		vigilant_raman
c3122c1df9c4	busybox	"sh"	5 minutes ago	Exited (0) 5 minutes ago		goofy_burnell
9d9d45511a9e	redis:alpine	"docker-entrypoint.s..."	2 weeks ago	Exited (0) 2 weeks ago		redis-server
75c1eed67dfa	postgres:alpine	"docker-entrypoint.s..."	2 weeks ago	Exited (255) 5 days ago	5432/tcp	postgresDB

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker rm a0
a0
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_1> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
545aa5eaa93d	busybox	"sh"	6 minutes ago	Exited (0) 5 minutes ago		vigilant_raman
c3122c1df9c4	busybox	"sh"	6 minutes ago	Exited (0) 6 minutes ago		goofy_burnell
9d9d45511a9e	redis:alpine	"docker-entrypoint.s..."	2 weeks ago	Exited (0) 2 weeks ago		redis-server
75c1eed67dfa	postgres:alpine	"docker-entrypoint.s..."	2 weeks ago	Exited (255) 5 days ago	5432/tcp	postgresDB

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

Lab Worksheet

- ย้ายตำแหน่งปัจจุบันไปที่ Lab8_2 เพื่อใช้เป็น Working directory
- 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

- 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

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_2> docker build -t 8_2_bb .
[*] Building 1.8s (5/5) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 151B
=> [internal] load metadata for docker.io/library/busybox:latest
=> [internal] load .dockerignore
=> => transferring context: 671B
=> [1/1] FROM docker.io/library/busybox:latest
=> exporting to image
=> => exporting layers
=> => writing image sha256:f1809fe5475416d2b1928d9ce3c0fe5880990ecad98da9efd6599d7ca41bc0e0
=> naming to docker.io/library/8_2_bb
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/03mkxgdh25obw8y2m65mwv8on
3 warnings found (use docker --debug to expand):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
- MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
What's next:
View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_2> docker run 8_2_bb
Phannakon Phungamngoen 653380022-2 Art
```

Lab Worksheet

(1) คำสั่งที่ใช้ในการ run คือ

_ docker run 8_2_bb

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

__ทำการตั้งชื่อและtag ของ image ตาม parameter ที่ใส่มาตามหลัง Option นี้__

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

FROM busybox

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

CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา"

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal และป้อนคำสั่งต่อไปนี้

\$ cat > Dockerfile << EOF

FROM busybox

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

CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา"

EOF

หรือใช้คำสั่ง

\$ touch Dockerfile

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

7. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

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

5. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง

Lab Worksheet

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

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

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_3> mv\m dockerfile
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_3> ls

Directory: D:\somethingIdroppedin\Software Engineer\lab8\lab8_3

Mode                LastWriteTime         Length Name
----                -
-a----             1/27/2025 10:27 AM             629 .dockerignore
-a----             1/27/2025 10:27 AM            1993 compose.yaml
-a----             1/27/2025 10:28 AM             136 Dockerfile
-a----             1/27/2025 10:27 AM             680 README.Docker.md

PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_3> docker build -t merlince4/lab8 .
[*] Building 1.3s (5/5) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.2s
=> => transferring dockerfile: 175B                               0.0s
=> [internal] load metadata for docker.io/library/busybox:latest  0.0s
=> [internal] load .dockerignore                                  0.2s
=> => transferring context: 671B                                   0.0s
=> CACHED [1/1] FROM docker.io/library/busybox:latest            0.0s
=> exporting to image                                             0.2s
=> => exporting layers                                           0.0s
=> => writing image sha256:b2977bdc659e1cc7b114035869e12007a368af193fa42451c8dd7e9ffd4bfe8a 0.0s
=> => naming to docker.io/merlince4/lab8                         0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/3md92lnr74p92qx4rt6Ujmc7

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

What's next:
View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_3> docker run merlince4/lab8
Phannakon Phungampon 653380922-2
```

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง

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

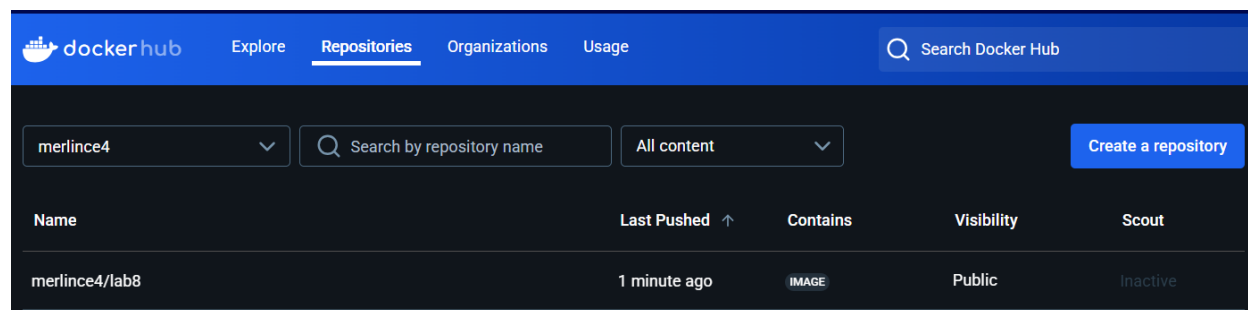
ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

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

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

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

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



Lab Worksheet

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

```
{
  "name": "101-app",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "scripts": {
    "prettify": "prettier -l --write \"**/*.js\"",
    "test": "jest",
    "dev": "nodemon src/index.js"
  },
  "dependencies": {
    "express": "^4.18.2",
    "mysql2": "^2.3.3",
    "sqlite3": "^5.1.2",
    "uuid": "^9.0.0",
    "wait-port": "^1.0.4"
  },
  "resolutions": {
    "ansi-regex": "5.0.1"
  },
  "prettier": {
    "trailingComma": "all",
    "tabWidth": 4,
    "useTabs": false,
    "semi": true,
    "singleQuote": true
  },
  "devDependencies": {
    "jest": "^29.3.1",
    "nodemon": "^2.0.20",
    "prettier": "^2.7.1"
  }
}
```

4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงในไฟล์

FROM node:18-alpine

WORKDIR /app

COPY . .

RUN yarn install --production

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

EXPOSE 3000

Lab Worksheet

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

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

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

```
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_4\getting-started\app> nvim dockerfile
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_4\getting-started\app> docker build -t myapp_6533800222 .
[+] Building 27.9s (9/9) FINISHED
=> [internal] load build definition from dockerfile                                docker:desktop-linux 0.3s
=> => transferring dockerfile: 156B                                              0.1s
=> [internal] load metadata for docker.io/library/node:18-alpine                 0.0s
=> [internal] load .dockerignore                                                 0.2s
=> => transferring context: 2B                                                  0.0s
=> [1/4] FROM docker.io/library/node:18-alpine                                0.4s
=> [internal] load build context                                                 2.0s
=> => transferring context: 4.62MB                                              1.7s
=> [2/4] WORKDIR /app                                                         0.4s
=> [3/4] COPY . .                                                             0.6s
=> [4/4] RUN yarn install --production                                         21.1s
=> exporting to image                                                         2.7s
=> => exporting layers                                                         2.6s
=> => writing image sha256:96eb096fec2f39504c5c7bd6c97f908066620295b5305c66bc5cfa7594993d6 0.0s
=> => naming to docker.io/library/myapp_6533800222                          0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/q17stmhmj99b1n0q88fetc0s5

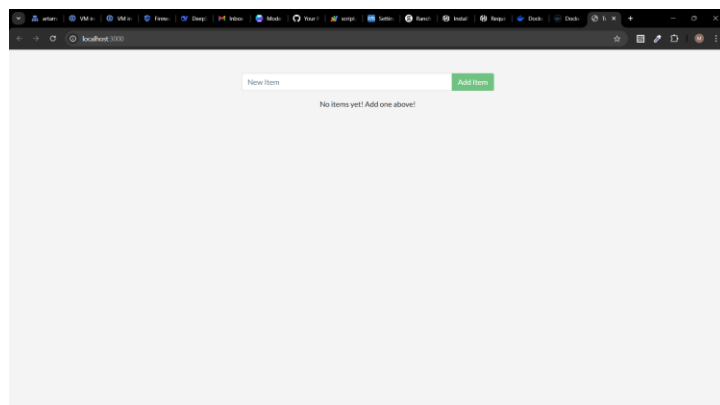
What's next:
View a summary of image vulnerabilities and recommendations -> docker scout quickview
```

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

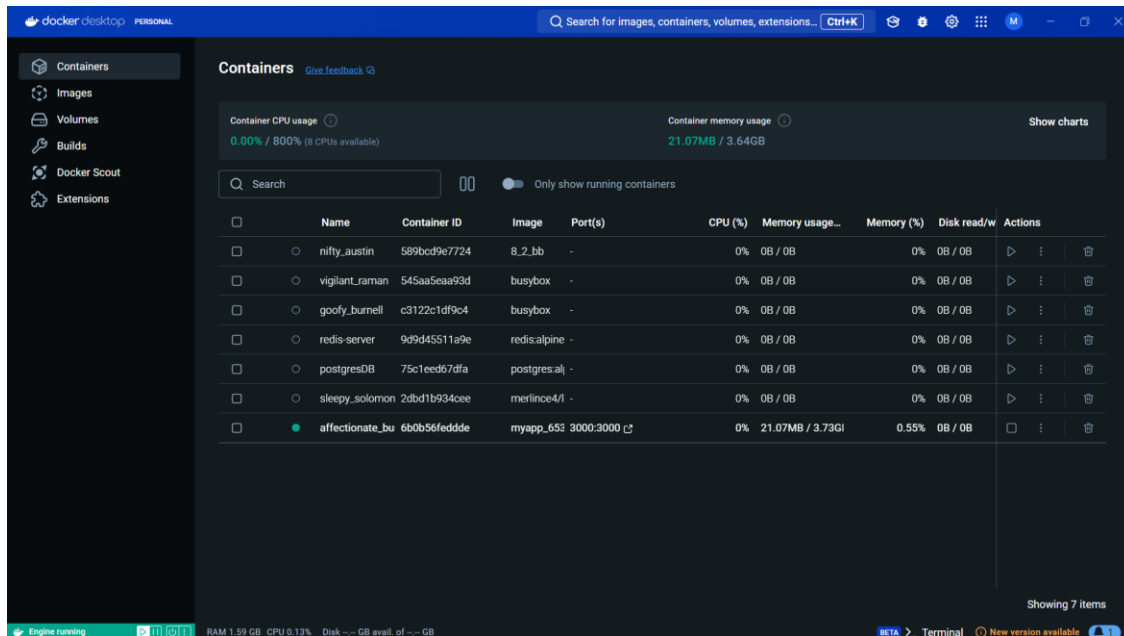
\$ docker run -dp 3000:3000 <myapp_รหัสสนศ. ไม่มีขีด>

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

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

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

```

PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_4\getting-started\app> nvim .
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_4\getting-started\app> docker build -t myapp_6533800222 .
[+] Building 28.0s (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
=> [internal] load build context
=> => transferring context: 8.07kB
=> [1/4] FROM docker.io/library/node:18-alpine
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => writing image sha256:87413dbf920871fca70e36c6dee9c268666f72e8f7a47f45b4010026728fedc5
=> => naming to docker.io/library/myapp_6533800222

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/wgu6ts62lr1ppmxdl1tblgzpqd

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS D:\somethingIdroppedin\Software Engineer\lab8\lab8_4\getting-started\app> docker run -d -p 3000:3000 myapp_6533800222
cc15f573019887be2511a033be5327e02651dac7714cff68fc0afec883f75439
docker: Error response from daemon: driver failed programming external connectivity on endpoint friendly_thompson (6281a9e9da3e7aebd6e96fe882c1a84aea4fd4a2ffc076d103ec8dc7fa735149): Bind for 0.0.0.0:3000 failed: port is already allocated.

```

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

___ Host หรือ เครื่องที่รัน Docker อยู่ นั้น port 3000 ไม่ว่าง ถูกใช้จาก Process อื่นอยู่ จึงไม่สามารถสร้าง Container ที่ต้องใช้ port นั้นได้

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

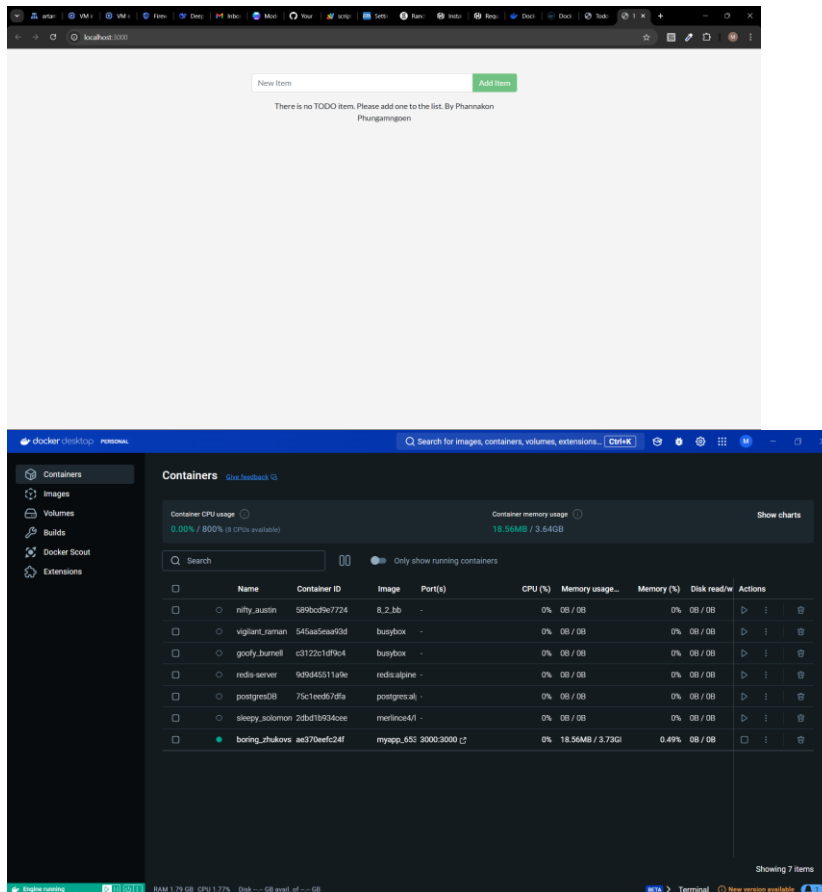
- ไปที่หน้าต่าง Containers
- เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
- ยืนยันโดยการกด Delete forever

12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6

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

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

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. บันทึกที่กรหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก

[Check point#12] Capture หน้าจอที่แสดงผล Admin password

Lab Worksheet

```

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

4b0ff83e9812423da8d0f9b70964dd03

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

*****
4b0ff83e9812423da8d0f9b70964dd03

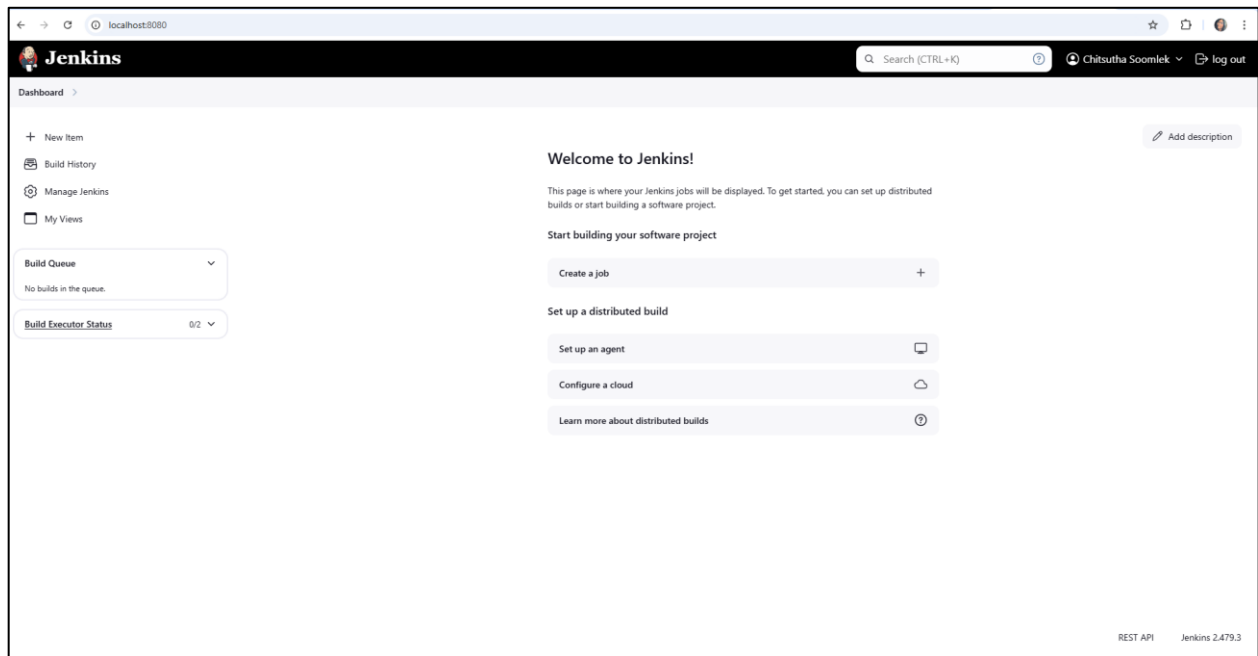
```

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

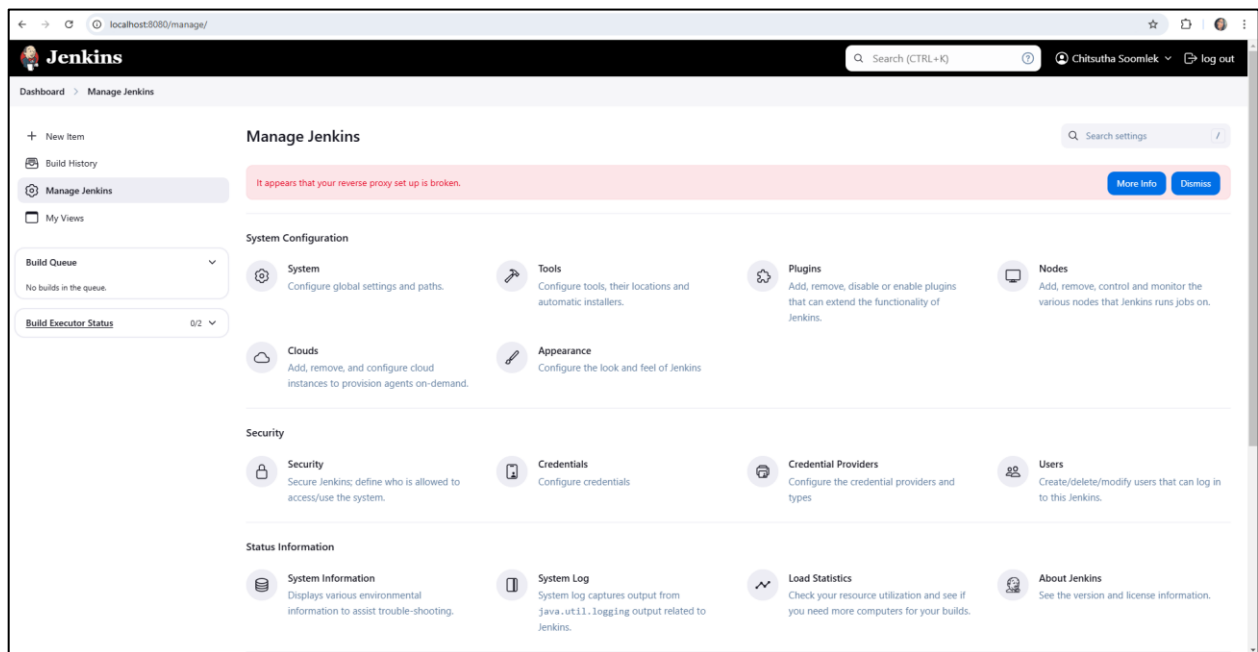
[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>
8. เมื่อติดตั้งเรียบร้อยแล้วจะพบหน้าจอ Dashboard ดังแสดงในภาพ

Lab Worksheet

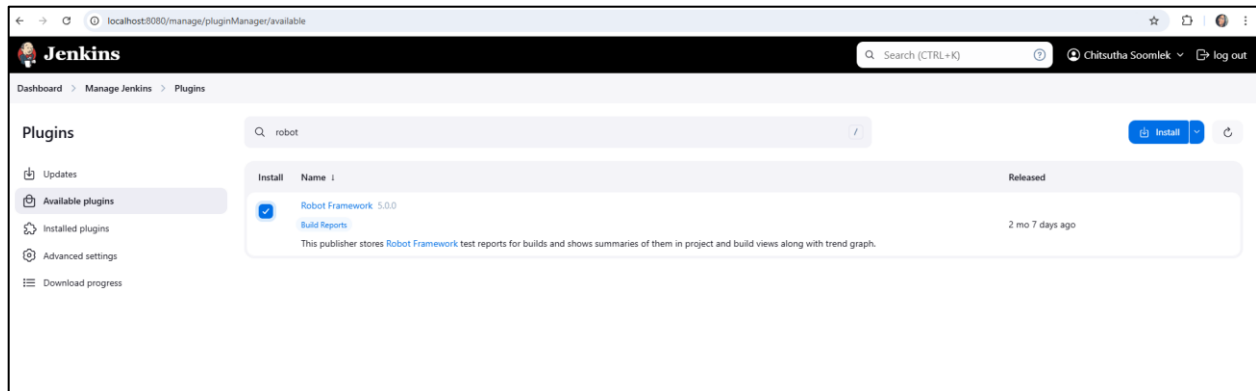


9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

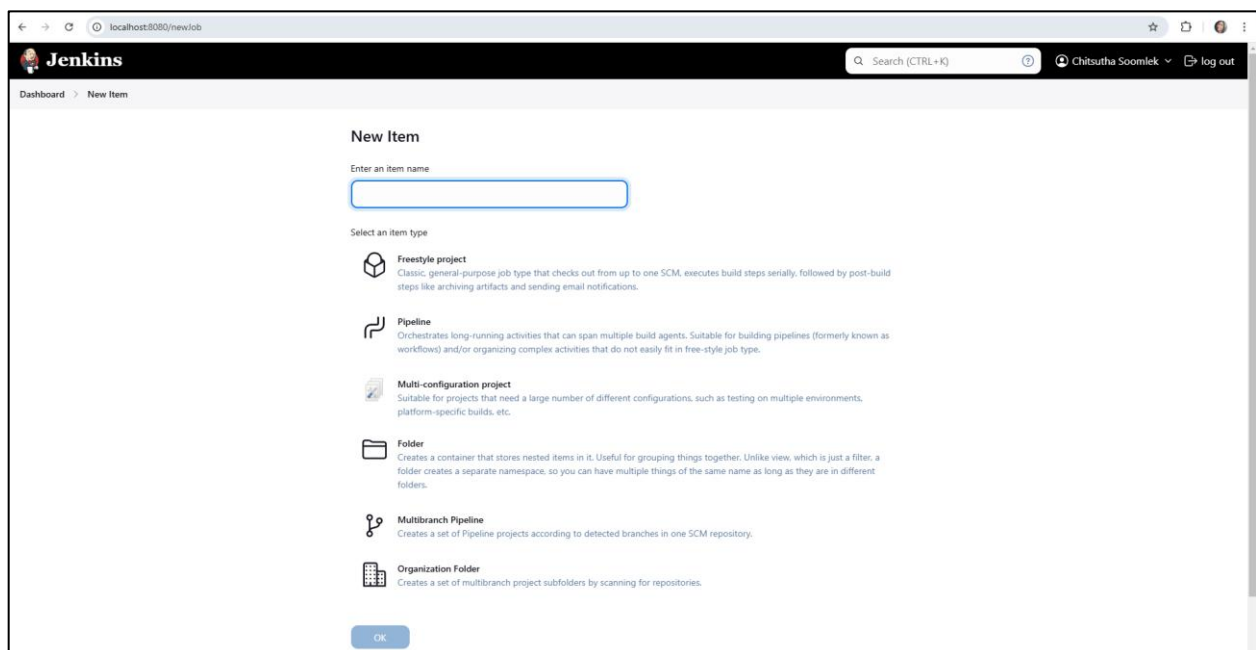


Lab Worksheet

10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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



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

Description: Lab 8.5

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

Build Trigger: เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

Lab Worksheet

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

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Plain text [Preview](#)

☐ Commit agent's Docker container ?

☐ Define a Docker template

☐ Discard old builds ?

☒ GitHub project

Project url ?

[https://github.com/Merlicne/lab8-softeng.git/](#)

Advanced ▾

Git ?

Repositories ?

Repository URL ?

[https://github.com/Merlicne/lab8-softeng.git](#)

Credentials ?

- none - ▾

+ Add

☒ Build periodically ?

Schedule ?

H/15 * * * * *

Would last have run at Tuesday, January 28, 2025 at 8:50:12 AM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 9:05:12 AM Coordinated Universal Time.

[GitHub build trigger for GitHub pull request](#) ?

```
set -e # Exit on error
set -x # Debug mode

# Create network
docker stop python-runner || true
docker network rm UATTEST || true
docker network create UATTEST

# Start Python server
docker run -d --rm --name python-runner --net UATTEST \
-v /var/jenkins_home/workspace/UAT:/app \
-w /app \
python:alpine python demoapp/server.py

sleep 10

docker run -v ${PWD}/pipe-reports:/opt/robotframework/reports:Z \
-v ${PWD}/UAT_test:/opt/robotframework/tests:Z \
-e ROBOT_OPTIONS="--variable SERVER:python-runner:7272 --variable DELAY:0" \
--net UATTEST \
--rm \
ppodgorsek/robot-framework || true

docker stop python-runner || true
docker network rm UATTEST || true
```

Lab Worksheet

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

```
set -e # Exit on error
set -x # Debug mode

# Create network
docker stop python-runner || true
docker network rm UATTEST || true
docker network create UATTEST

# Start Python server
docker run -d --rm --name python-runner --net UATTEST \
-v /var/jenkins_home/workspace/UAT:/app \
-w /app \
python:alpine python demoapp/server.py

sleep 10

docker run -v ${PWD}/pipe-reports:/opt/robotframework/reports:Z \
-v ${PWD}/UAT_test:/opt/robotframework/tests:Z \
-e ROBOT_OPTIONS="--variable SERVER:python-runner:7272 --variable DELAY:0" \
--net UATTEST \
--rm \
ppodgorsek/robot-framework || true

docker stop python-runner || true
docker network rm UATTEST || true
```


Lab Worksheet

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

13. กด Apply และ Save

14. สิ่ง Build Now

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

```

Started by timer
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAU
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/UAU/.git # timeout=10
Fetching changes from the remote git repository
> git config core.origin.url https://github.com/Merlino/lab-softeng.git # timeout=10
Fetching upstream changes from https://github.com/Merlino/lab-softeng.git
> git --version # timeout=10
> git --version # git version 2.39.5
> git fetch --tags --force --progress -- https://github.com/Merlino/lab-softeng.git refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
> git rev-parse refs/remotes/origin/main # timeout=10
Checking out Revision 13ad30af50b4b626ce23ec1acd3eaf6 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 13ad30af50b4b626ce23ec1acd3eaf6 # timeout=10
Commit message: "git robot test"
> git rev-list --no-walk 13ad30af50b4b626ce23ec1acd3eaf6 # timeout=10
[UAU] #1 / Fail sh -v /var/jenkins_home/workspace/UAU/13ad30af50b4b626ce23ec1acd3eaf6
> git --version
Docker version 27.5.1, build 9f0a805
# docker image ls
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
python              alpine     d5c8e1ba27f7 10 days ago   44.9MB
podgorcak/robot-framework latest      03a01dc58044 2 months ago   4.15GB
# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED        STATUS        PORTS          NAMES
# set -x
# docker stop python-runner
Error response from daemon: No such container: python-runner
# true
# docker network rm UATTEST
Error response from daemon: network UATTEST not found
# true
# docker network creates UATTEST
17f72c49622f6e00d19f9e5f61621e40540b977311e95074d1350820b72
# docker run -d --rm --name python-runner --net UATTEST -v /var/jenkins_home/workspace/UAU:/app -w /app python:alpine python domapop/server.py
a511b9a0490840817a0dcb1ec5a747621a72a7773c4928c48276a91
# sleep 10
# docker run -v /var/jenkins_home/workspace/UAU:/app:ro python:alpine:reports:/opt/robotframework/reports:z -v /var/jenkins_home/workspace/UAU:/UAU:latest /opt/robotframework/robotframework
runnerr 7272 --variable DELAY=0 --net UATTEST -re podgorcak/robot-framework
=====
Tests
=====
Tests.UAT-Lab7-001 :: A test suite containing tests related to invalid login.
=====
Open Form                                     | PASS |
-----
Tests.UAT-Lab7-001 :: A test suite containing tests related to invalid login.
=====
Open Form                                     | PASS |
-----
Record Success                               | FAIL |
Location should have contained 'http://python-runner:7272/Complete.html' but it was 'http://python-runner:7272/Form.html'.
Tests.UAT-Lab7-001 :: A test suite containing tests related to inv... | FAIL |
2 tests, 1 passed, 1 failed
=====
Tests.UAT-Lab7-002 :: A test suite containing tests related to invalid login.
=====
Open Form                                     | PASS |
-----
Empty Destination                             | FAIL |
Page should have contained text 'Please enter your destination.' but did not.
Empty Email                                   | FAIL |
Page should have contained text 'Please enter a valid email address.' but did not.
Invalid Email                                | FAIL |
Page should have contained text 'Please enter a valid email address.' but did not.
Empty Phone Number                           | FAIL |
Page should have contained text 'Please enter a phone number.' but did not.
Invalid Phone Number                         | FAIL |
Page should have contained text 'Please enter a valid phone number, e.g., 081.234.5678, 081.234.5678, or 081.234.5678)' but did not.
Tests.UAT-Lab7-002 :: A test suite containing tests related to inv... | FAIL |
6 tests, 1 passed, 5 failed
=====
Tests
=====
8 tests, 2 passed, 6 failed
=====
Output: /opt/robotframework/reports/output.xml
Log: /opt/robotframework/reports/log.html
Report: /opt/robotframework/reports/report.html
# true
# docker stop python-runner
python-runner
# docker network rm UATTEST
UATTEST
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 id:
Done!
=====

```