

Lab Worksheet

ชื่อ-นามสกุล

ภัทรพล ต้นสวรรค์

รหัสนักศึกษา 653380338-5 Section 3

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) แสดงผลลัพธ์ที่ได้พร้อมกับตอบคำถามต่อไปนี้

```
C:\Users\First\Lab8_1>docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
jenkins/jenkins	lts-jdk17	44c1caefd796	2 weeks ago	468MB
busybox	latest	af4709625109	3 months ago	4.27MB
synthesizedio/whalesay	latest	07da125a0bc8	6 months ago	45.2MB

(1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร

Ans. Docker images ต่างๆ ที่มีในเครื่อง

(2) Tag ที่ใช้บ่งบอกถึงอะไร

Ans. ระบุถึงเวอร์ชันที่ใช้ของ Docker images นั้นๆ

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

```
C:\Users\First\Lab8_1>docker run busybox
C:\Users\First\Lab8_1>docker run -it busybox sh
/ # ls
bin  dev  etc  home  lib  lib64  proc  root  sys  tmp  usr  var
```

Lab Worksheet

```

/ # ls -la
total 48
drwxr-xr-x 1 root root 4096 Jan 23 02:56 .
drwxr-xr-x 1 root root 4096 Jan 23 02:56 ..
-rwxr-xr-x 1 root root 0 Jan 23 02:56 .dockerenv
drwxr-xr-x 2 root root 12288 Sep 26 21:31 bin
drwxr-xr-x 5 root root 360 Jan 23 02:56 dev
drwxr-xr-x 1 root root 4096 Jan 23 02:56 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 269 root root 0 Jan 23 02:56 proc
drwx----- 1 root root 4096 Jan 23 02:56 root
dr-xr-xr-x 11 root root 0 Jan 23 02:56 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

```

```

C:\Users\First\Lab8_1>docker run busybox echo "Hello Phattaraphon Tonsawan from busybox"
C:\Users\First\Lab8_1>docker ps -a

```

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
9694cb43c359	busybox	vigorous_austin	"echo 'Hello Phattar..."	11 seconds ago	Exited (0) 10 seconds ago	
b257737ea7f6	busybox	recurring_matsumoto	"sh"	About a minute ago	Exited (0) About a minute ago	
a548936b0bbb	busybox	determined_black	"sh"	2 minutes ago	Exited (0) 2 minutes ago	
5f8431dac755	jenkins/jenkins:lts-jdk17	unruffled_aryabhata	"/usr/bin/tini -- /u..."	22 minutes ago	Up 22 minutes	0.0.0.0:8080->8080
9d6e9b210b66	synthesizedio/whalesay:latest	dreamy_nightingale	"/usr/local/bin/cows..."	45 minutes ago	Exited (0) 45 minutes ago	
c3d5b284fe09	synthesizedio/whalesay:latest	frosty_stonebraker	"/usr/local/bin/cows..."	45 minutes ago	Exited (0) 45 minutes ago	

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

Ans. เปิดให้คอนเทนเนอร์สามารถรับ input จากผู้ใช้ได้และจำลอง terminal ให้กับคอนเทนเนอร์เพื่อให้การทำงานเหมือนกับการใช้งาน terminal ปกติ

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

Ans. แสดงถึงสถานะของคอนเทนเนอร์ ว่ากำลังทำงานอยู่หรือหยุดทำงานไปแล้ว

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

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

```

C:\Users\First\Lab8_1>docker rm 9694cb43c359
9694cb43c359
C:\Users\First\Lab8_1>docker ps -a

```

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
b257737ea7f6	busybox	recurring_matsumoto	"sh"	15 minutes ago	Exited (0) 15 minutes ago	
a548936b0bbb	busybox	determined_black	"sh"	16 minutes ago	Exited (0) 16 minutes ago	
5f8431dac755	jenkins/jenkins:lts-jdk17	unruffled_aryabhata	"/usr/bin/tini -- /u..."	36 minutes ago	Up 36 minutes	0.0.0.0:8080->8080/tc
9d6e9b210b66	synthesizedio/whalesay:latest	dreamy_nightingale	"/usr/local/bin/cows..."	59 minutes ago	Exited (0) 59 minutes ago	
c3d5b284fe09	synthesizedio/whalesay:latest	frosty_stonebraker	"/usr/local/bin/cows..."	About an hour ago	Exited (0) About an hour ago	

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

```

PS C:\Users\First\Lab8_2> docker build -t lab8_2 .
>>
[+] Building 0.1s (5/5) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile              0.0s
=> == transferring dockerfile: 184B                             0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2) 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
=> == writing image sha256:bcee7c37aa87f53109abf3bcb191fbaf0258a0c322782b580cbb16335841a479 0.0s
=> == naming to docker.io/library/lab8_2                         0.0s

1 warning found (use docker --debug to expand):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
PS C:\Users\First\Lab8_2> docker run lab8_2
>>
Hi there. This is my first docker image.
ภัทรพล ดันสวรรค์ 653380338-5 BigDaddy
PS C:\Users\First\Lab8_2>
  
```

The screenshot also shows the Dockerfile content in the Docker Desktop interface:

```

1 FROM busybox
2 CMD echo "Hi there. This is my first docker image." && echo "ภัทรพล ดันสวรรค์ 653380338-5 BigDaddy"
3
  
```

(1) คำสั่งที่ใช้ในการ run คือ

Ans. รันคอนเทนเนอร์จาก images โดยสร้างคอนเทนเนอร์จาก Dockerfile ที่ได้สร้าง

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

Ans. ใช้สำหรับกำหนดชื่อให้กับ images lab8_2 จะช่วยให้ Run อ้างอิง image lab8_2 ได้โดยไม่ต้องใช้ 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

Lab Worksheet

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 และบันทึกคำสั่งต่อไปนี้

\$ 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

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

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

```
PS C:\Users\First\lab8_3> docker build -t phattaraphonbigdaddy/lab8_3 .
[+] Building 0.1s (5/5) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 209B                                              0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2) 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

PS C:\Users\First\lab8_3> docker run phattaraphonbigdaddy/lab8_3
Hi there. My work is done. You can run them from my Docker image.
ภัทรพล ต้นสวรรค์ 653380338-5 BigDaddy
PS C:\Users\First\lab8_3> 
```

Lab Worksheet

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง

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

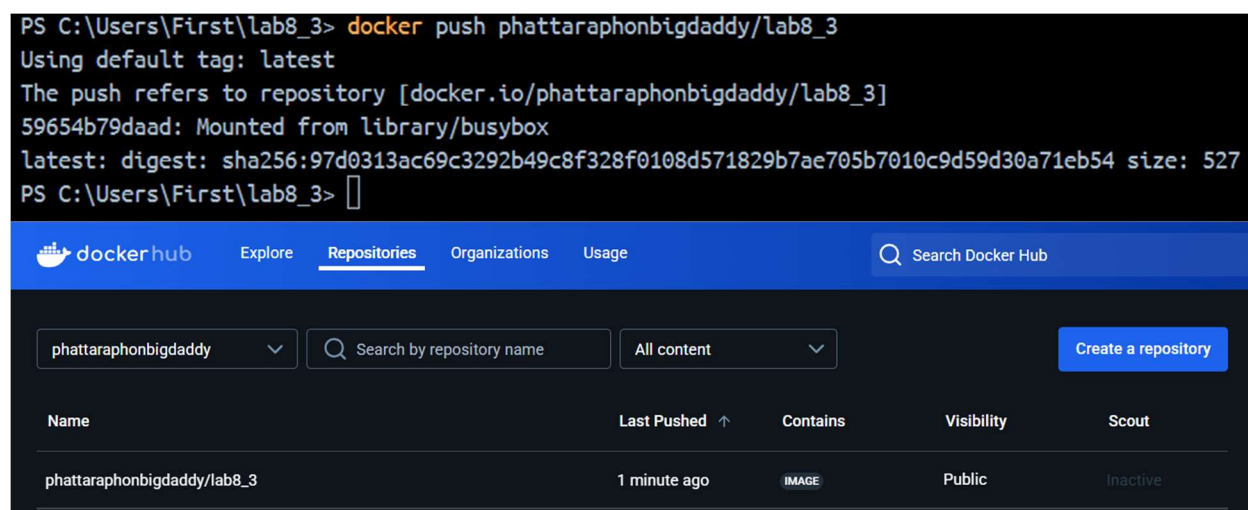
ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

```
$ 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)



แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

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

```
$ git clone https://github.com/docker/getting-started.git
```
- เปิดดูองค์ประกอบภายใน getting-started/app เมื่อพบไฟล์ package.json ให้ใช้ Text editor ในการเปิดอ่าน

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

Lab Worksheet

```

PS C:\Users\First\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), 5.28 MiB | 1.27 MiB/s, done.
Resolving deltas: 100% (523/523), done.
PS C:\Users\First\Lab8_4> cd getting-started
PS C:\Users\First\Lab8_4\getting-started> ls

-a----      1/28/2025  10:32 PM             267 build.sh
-a----      1/28/2025  10:32 PM             179 docker-compose.yml
-a----      1/28/2025  10:32 PM            1223 Dockerfile
-a----      1/28/2025  10:32 PM           11556 LICENSE
-a----      1/28/2025  10:32 PM           2076 mkdocs.yml
-a----      1/28/2025  10:32 PM           1739 README.md
-a----      1/28/2025  10:32 PM             110 requirements.txt

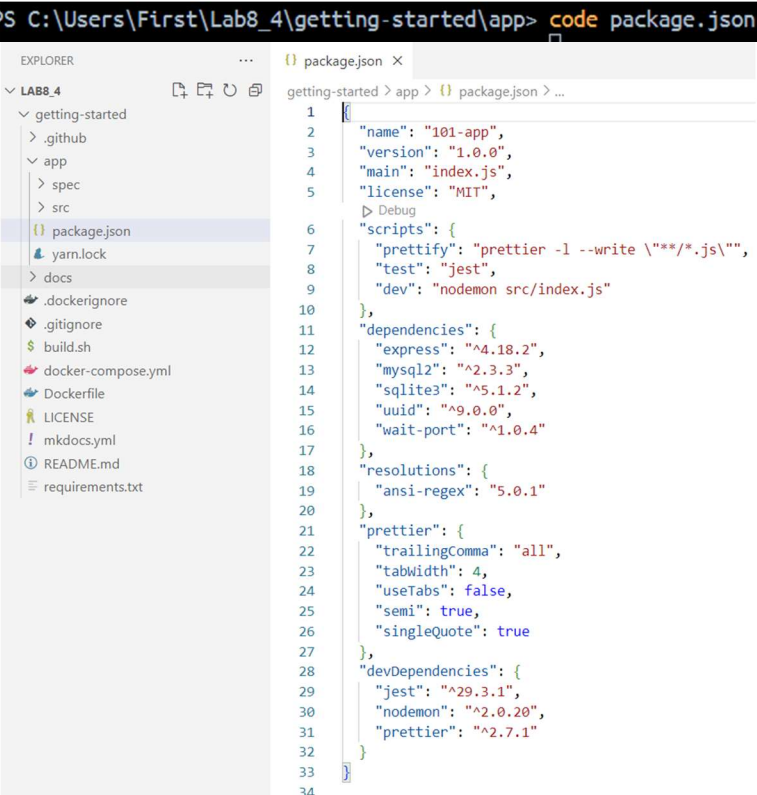
PS C:\Users\First\Lab8_4\getting-started\app> ls

Directory: C:\Users\First\Lab8_4\getting-started\app

Mode                LastWriteTime         Length Name
----                -
d-----      1/28/2025  10:32 PM              spec
d-----      1/28/2025  10:32 PM              src
-a----      1/28/2025  10:32 PM             678 package.json
-a----      1/28/2025  10:32 PM          150541 yarn.lock

PS C:\Users\First\Lab8_4\getting-started\app> code package.json

```



```

1  {
2    "name": "101-app",
3    "version": "1.0.0",
4    "main": "index.js",
5    "license": "MIT",
6    "scripts": {
7      "prettify": "prettier -l --write \"**/*.js\"",
8      "test": "jest",
9      "dev": "nodemon src/index.js"
10   },
11   "dependencies": {
12     "express": "^4.18.2",
13     "mysql2": "^2.3.3",
14     "sqlite3": "^5.1.2",
15     "uuid": "^9.0.0",
16     "wait-port": "^1.0.4"
17   },
18   "resolutions": {
19     "ansi-regex": "5.0.1"
20   },
21   "prettier": {
22     "trailingComma": "all",
23     "tabWidth": 4,
24     "useTabs": false,
25     "semi": true,
26     "singleQuote": true
27   },
28   "devDependencies": {
29     "jest": "^29.3.1",
30     "nodemon": "^2.0.20",
31     "prettier": "^2.7.1"
32   }
33 }
34

```


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_รหัสสนศ. ไม่มีขีด

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

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ

```
PS C:\Users\First\Lab8_4\getting-started\app> New-Item -ItemType File -Name Dockerfile
>>

Directory: C:\Users\First\Lab8_4\getting-started\app

Mode                LastWriteTime         Length Name
----                -
-a----             1/28/2025  10:53 PM              0 Dockerfile

PS C:\Users\First\Lab8_4\getting-started\app> code Dockerfile

PS C:\Users\First\Lab8_4\getting-started\app> docker build -t myapp_6533803385 .
[+] Building 10.6s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 154B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25
=> [internal] load build context
=> => transferring context: 2.61kB
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => writing image sha256:73794713a2b213dee0c5a4021a804543bf72b89aeeea0a20684086594d989ea6
=> => naming to docker.io/library/myapp_6533803385
```

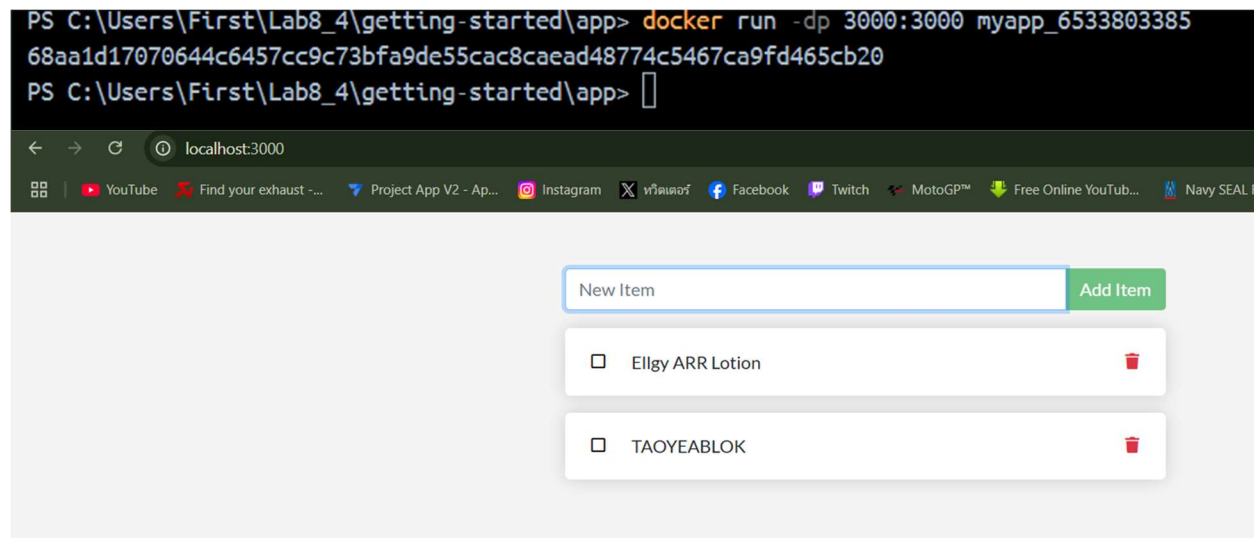
6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

Lab Worksheet

```
$ docker run -dp 3000:3000 <myapp_รหัสสนศ. ไม่มีชี้ด>
```

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

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



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

```

PS C:\Users\First\Lab8_4\getting-started\app> cd src
PS C:\Users\First\Lab8_4\getting-started\app\src> cd static
PS C:\Users\First\Lab8_4\getting-started\app\src\static> cd js
PS C:\Users\First\Lab8_4\getting-started\app\src\static\js> code app.js

PS C:\Users\First\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533803385
2004f1eb963168a4e534dabd53f748b35957373f5fb6a0af51e9b84af94107
docker: Error response from daemon: driver failed programming external connectivity on endpoint serene_robinson (64fb6be692f8d3646b2a1da6cc770797e3f0881538fa29975d40ef3e4c3b44df): Bind for 0.0.0.0:3000 failed: port is already allocated.

```

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

Ans. Port3000:3000 ได้ถูกกำหนดใช้ไปแล้วต้องกำหนดportอื่น

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

```

PS C:\Users\First\Lab8_4\getting-started\app> docker ps

```

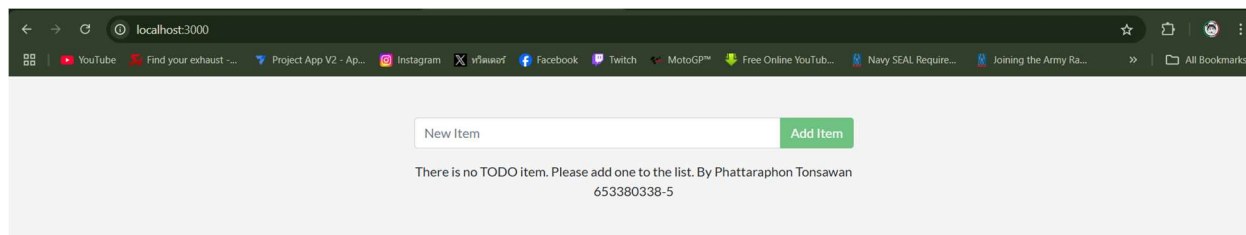
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
eaa05e1b488c	myapp_6533803385	"docker-entrypoint.s..."	2 minutes ago	Up 2 minutes	0.0.0.0:3000->3000/tcp
5f8431dac755	jenkins/jenkins:lts-jdk17	"/usr/bin/tini -- /u..."	5 days ago	Up 2 hours	0.0.0.0:8080->8080/tcp, 0.0.0.0:50000->50000/tcp

```

PS C:\Users\First\Lab8_4\getting-started\app> docker stop eaa05e1b488c
eaa05e1b488c
PS C:\Users\First\Lab8_4\getting-started\app> docker rm eaa05e1b488c
eaa05e1b488c

```

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

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

7dd70815ad0d4deabccf05da39bd6703

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

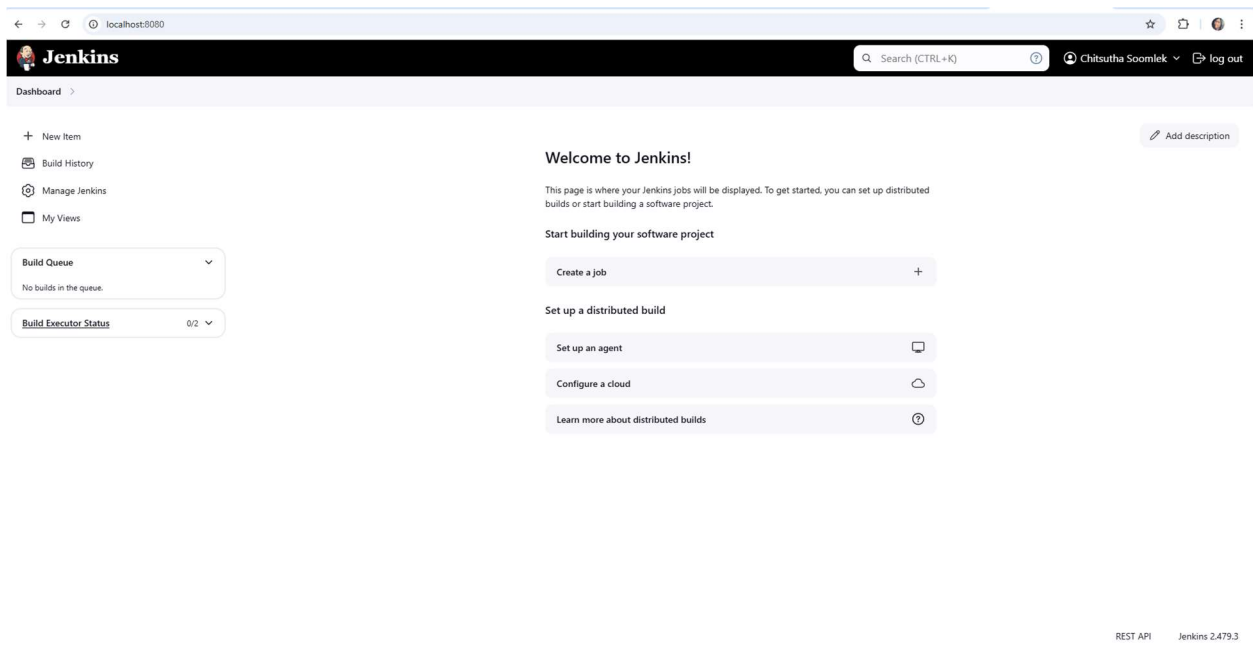
The screenshot shows the Jenkins 'Getting Started' page in a web browser at localhost:8080. The main heading is 'Create First Admin User'. The form contains the following fields:

- Username:** phattaraphon_3385
- Password:** (masked with dots)
- Confirm password:** (masked with dots)
- Full name:** Phattaraphon Tonsawan

At the bottom of the form, it says 'Jenkins 2.479.3'. There are two buttons: 'Skip and continue as admin' and 'Save and Continue'.

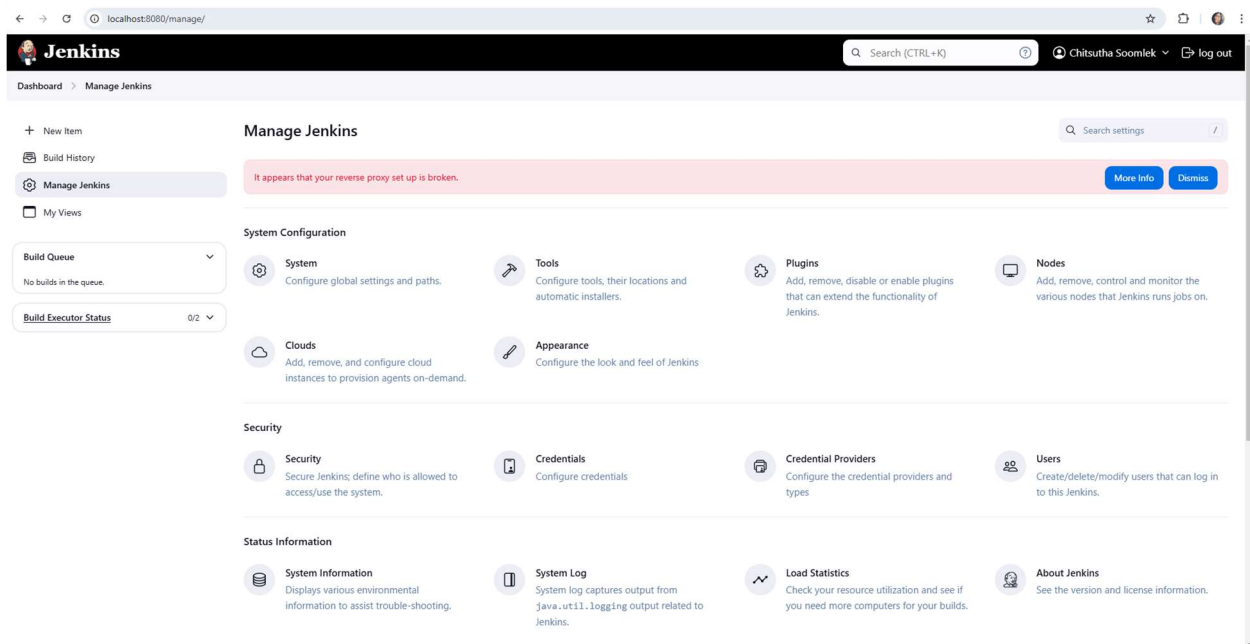
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบหน้าจอ Dashboard ดังแสดงในภาพ



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

Lab Worksheet

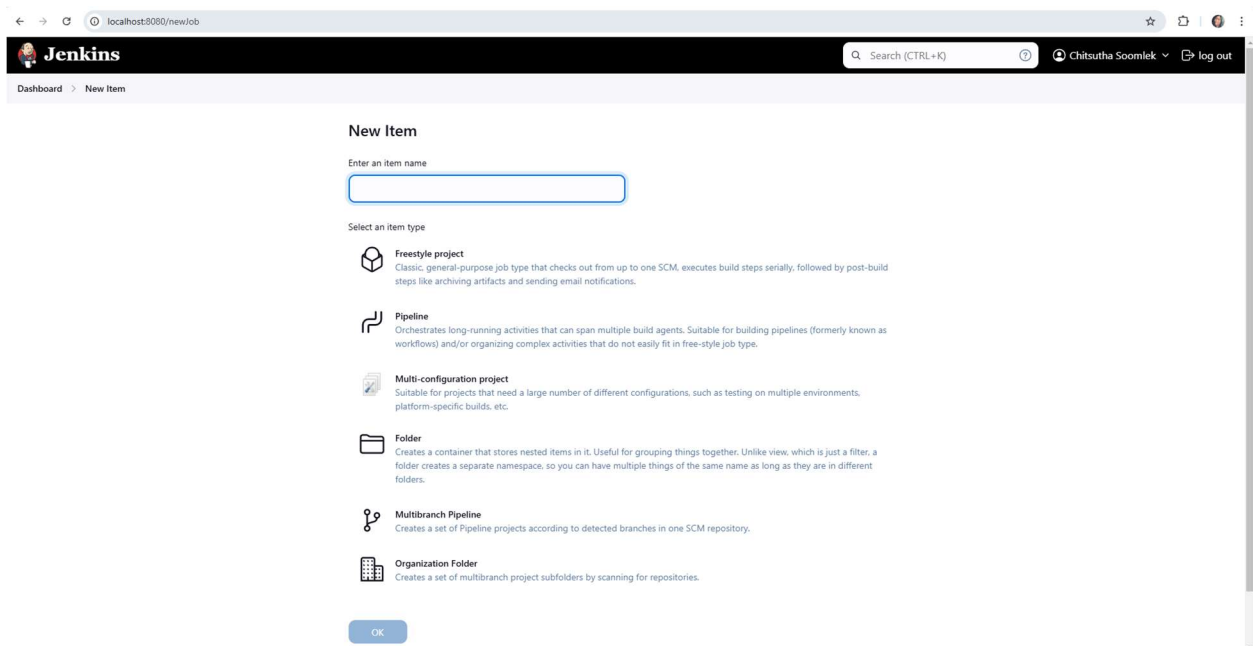


10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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 ไปถึงไฟล์ให้เรียบร้อยด้วย)

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Description

Lab 8.5

Plain text Preview

☐ Discard old builds ?

☒ GitHub project

Project url ?

https://github.com/PhattaraphonTonsawan/lab7_UAT.git

Advanced ▾

Lab Worksheet

Source Code Management

☐ None☒ Git ?

Repositories ?

Repository URL ?

Credentials ?

Advanced ▾

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?☐ Build after other projects are built ?☒ Build periodically ?

Schedule ?

Would last have run at Tuesday, January 28, 2025 at 6:05:43 PM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 6:20:43 PM Coordinated Universal Time.

Command

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

```
kill -9 -f "chrome" || true
kill -9 -f "chromedriver" || true

export PATH=$PATH:/usr/local/bin:/opt

export CHROME_BIN=/opt/google/chrome/google-chrome
export CHROMEDRIVER_BIN=/usr/local/bin/chromedriver

mkdir -p results
robot --outputdir results login_tests/valid_login.robot
```

Advanced ▾

Lab Worksheet

Post-build Actions

Publish Robot Framework test results ?

Directory of Robot output

Path to directory containing robot xml and html files (relative to build workspace)

results

Advanced ▾ Edited

Thresholds for build result ?

80.0

20.0

☒ DEPRECATED! THIS FLAG DOES NOTHING! - Use thresholds for critical tests only

☐ Include skipped tests in total count for thresholds

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

Ans.สั่งเพื่อให้ไฟล์ที่เป็น .robot ทำงานและให้ RobotFramework ทำงาน

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

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

Lab Worksheet

Status

</> Changes

Console Output

Edit Build Information

Delete build '#20'

Timings

Git Build Data

Robot Results

Previous Build

Console Output

```

Started by timer
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/UAT/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/PhattaraphonTonsawan/lab7_UAT.git # timeout=10
Fetching upstream changes from https://github.com/PhattaraphonTonsawan/lab7_UAT.git
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/PhattaraphonTonsawan/lab7_UAT.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 8b67ce44cd085ebf649674df08fd872545e6b3c0 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 8b67ce44cd085ebf649674df08fd872545e6b3c0 # timeout=10
Commit message: "Update resource.robot"
> git rev-list --no-walk 8b67ce44cd085ebf649674df08fd872545e6b3c0 # timeout=10
[UAT] $ /bin/sh -xe /tmp/jenkins5044259668752575687.sh
+ pkill -9 -f chrome

```

Dashboard > UAT > #20 > Console Output

```

+ robot --outputdir results login_tests/valid_login.robot
=====
Valid Login :: A test suite with a single test for valid login.
=====
Valid Login                                     | FAIL |
NoSuchDriverException: Message: Unable to obtain driver for chrome; For documentation on this error, please visit:
https://www.selenium.dev/documentation/webdriver/troubleshooting/errors/driver_location
-----
Valid Login :: A test suite with a single test for valid login. | FAIL |
1 test, 0 passed, 1 failed
=====
Output: /var/jenkins_home/workspace/UAT/results/output.xml
Log: /var/jenkins_home/workspace/UAT/results/log.html
Report: /var/jenkins_home/workspace/UAT/results/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

```

Jenkins

Search (CTRL+K)

Phattaraphon Tonsawan

log out

Dashboard > UAT >

Status

</> Changes

Workspace

Build Now

Configure

Delete Project

Robot Results

GitHub

Rename

UAT

Lab8.5

Latest Robot Results:

Total Failed Passed Skipped Pass %

All tests 1 1 0 0 0.0

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

Permalinks

- Last build (#21), 1 min 0 sec ago
- Last failed build (#21), 1 min 0 sec ago
- Last unsuccessful build (#21), 1 min 0 sec ago
- Last completed build (#21), 1 min 0 sec ago

Robot Framework Tests Trend (all tests)

Builds

Filter /

Today

#21 4:15 PM

#20 4:05 PM

#19 4:01 PM

#18 3:56 PM