

ชื่อ-นามสกุล อรรถมาภรณ์ ถาวรพิศาลดิลก รหัสนักศึกษา 653380219-3 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

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

- ย้ายตำแหน่งปัจจุบันไปที่ Lab8_1 เพื่อใช้เป็น Working directory
- ป้อนคำสั่ง \$ docker pull busybox หรือ \$ sudo docker pull busybox สำหรับกรณีที่ติดปัญหา Permission denied (หมายเหตุ: BusyBox เป็น software suite ที่รองรับคำสั่งบางอย่างบน Unix - <https://busybox.net>)
- ป้อนคำสั่ง \$ docker images

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

```
C:\Users\Atjamaporn>cd C:\653380219-3\Lab8_1

C:\653380219-3\Lab8_1>docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
9c0abc9c5bd3: Pull complete
Digest: sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab073cd3c138bd458fc8257fbf1
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest

C:\653380219-3\Lab8_1>docker images
REPOSITORY    TAG       IMAGE ID      CREATED        SIZE
busybox        latest    af4709625109  4 months ago  4.27MB

C:\653380219-3\Lab8_1>docker run busybox
```

- สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร ชื่อ image
- Tag ที่ใช้บ่งบอกถึงอะไร เวอร์ชันของ image

- ป้อนคำสั่ง \$ docker run busybox
- ป้อนคำสั่ง \$ docker run -it busybox sh
- ป้อนคำสั่ง ls
- ป้อนคำสั่ง ls -la
- ป้อนคำสั่ง exit
- ป้อนคำสั่ง \$ docker run busybox echo "Hello ชื่อและนามสกุลของนักศึกษา from busybox"

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

11. ป้อนคำสั่ง \$ docker ps -a

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

```
C:\653380219-3\Lab8_1>docker run busybox

C:\653380219-3\Lab8_1>docker run -it busybox sh
/ # ls
bin      etc      lib      proc     sys      usr
dev      home    lib64    root     tmp      var
/ # ls -la
total 48
drwxr-xr-x  1 root    root      4096 Jan 29 02:15 .
drwxr-xr-x  1 root    root      4096 Jan 29 02:15 ..
-rwxr-xr-x  1 root    root        0 Jan 29 02:15 .dockerenv
drwxr-xr-x  2 root    root     12288 Sep 26 21:31 bin
drwxr-xr-x  5 root    root      360 Jan 29 02:15 dev
drwxr-xr-x  1 root    root      4096 Jan 29 02:15 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 237 root    root        0 Jan 29 02:15 proc
drwx----- 1 root    root      4096 Jan 29 02:15 root
dr-xr-xr-x 11 root    root        0 Jan 29 02:15 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:\653380219-3\Lab8_1>docker run busybox echo "Hello อรรถมาภรณ์ ถาวรพิ ศาลติ ลก from busybox"
"Hello อรรถมาภรณ์ ถาวรพิ ศาลติ ลก from busybox"
```

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

```
C:\653380219-3\Lab8_1>docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
PORTS         NAMES
d6b1e1e98f92   busybox    "echo "Hello อรรถมาภ..." 7 seconds ago  Exited (0
) 6 seconds ago      strange_fermat
fba2fad11fa2   busybox    "sh"                    44 seconds ago Exited (0
) 26 seconds ago      vibrant_brown
a8d7b8295ccf   busybox    "sh"                    52 seconds ago Exited (0
) 51 seconds ago      cool_colden
3b04441a283e   busybox    "echo "Hello อรรถมาภ..." 12 hours ago   Exited (0
) 12 hours ago      dreamy_ardinghelli
c5c41df745a4   busybox    "sh"                    12 hours ago   Exited (0
) 12 hours ago      nostalgic_chebyshev
b3d1cd495fe4   busybox    "sh"                    12 hours ago   Exited (0
) 12 hours ago      serene_newton
773a2d0f98b7   busybox    "sh"                    12 hours ago   Up 12 hou
rs                    lucid_albattani
f28c8d11f325   busybox    "sh"                    12 hours ago   Exited (0
) 12 hours ago      hardcore_mahavira

C:\653380219-3\Lab8_1>docker rm f28c8d11f325
f28c8d11f325

C:\653380219-3\Lab8_1>docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS
PORTS         NAMES
d6b1e1e98f92   busybox    "echo "Hello อรรถมาภ..." 6 minutes ago  Exited (0
) 6 minutes ago      strange_fermat
fba2fad11fa2   busybox    "sh"                    7 minutes ago  Exited (0
) 7 minutes ago      vibrant_brown
a8d7b8295ccf   busybox    "sh"                    7 minutes ago  Exited (0
) 7 minutes ago      cool_colden
3b04441a283e   busybox    "echo "Hello อรรถมาภ..." 12 hours ago   Exited (0
) 12 hours ago      dreamy_ardinghelli
c5c41df745a4   busybox    "sh"                    12 hours ago   Exited (0
) 12 hours ago      nostalgic_chebyshev
b3d1cd495fe4   busybox    "sh"                    12 hours ago   Exited (0
) 12 hours ago      serene_newton
773a2d0f98b7   busybox    "sh"                    12 hours ago   Up 12 hour
s                    lucid_albattani
```

- (1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
-it จะช่วยให้เมื่อเปิด shell ภายใน container ทำให้ผู้ใช้สามารถรันคำสั่งภายใน container ได้เหมือนกับการใช้ terminal ปกติ
- (2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

ตรวจสอบสถานะของ container มีกำลังรื้ออยู่หรือปิดไปแล้ว

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

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

```
C:\653380219-3\Lab8_1>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
	PORTS	NAMES		
d6b1e1e98f92	busybox	"echo "Hello อรรถมาภ..."	7 seconds ago	Exited (0
) 6 seconds ago		strange_fermat		
fba2fad11fa2	busybox	"sh"	44 seconds ago	Exited (0
) 26 seconds ago		vibrant_brown		
a8d7b8295ccf	busybox	"sh"	52 seconds ago	Exited (0
) 51 seconds ago		cool_colden		
3b04441a283e	busybox	"echo "Hello อรรถมาภ..."	12 hours ago	Exited (0
) 12 hours ago		dreamy_ardinghelli		
c5c41df745a4	busybox	"sh"	12 hours ago	Exited (0
) 12 hours ago		nostalgic_chebyshev		
b3d1cd495fe4	busybox	"sh"	12 hours ago	Exited (0
) 12 hours ago		serene_newton		
773a2d0f98b7	busybox	"sh"	12 hours ago	Up 12 hou
rs		lucid_albattani		
f28c8d11f325	busybox	"sh"	12 hours ago	Exited (0
) 12 hours ago		hardcore_mahavira		

```
C:\653380219-3\Lab8_1>docker rm f28c8d11f325
f28c8d11f325
```

```
C:\653380219-3\Lab8_1>docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
	PORTS	NAMES		
d6b1e1e98f92	busybox	"echo "Hello อรรถมาภ..."	6 minutes ago	Exited (0)
6 minutes ago		strange_fermat		
fba2fad11fa2	busybox	"sh"	7 minutes ago	Exited (0)
7 minutes ago		vibrant_brown		
a8d7b8295ccf	busybox	"sh"	7 minutes ago	Exited (0)
7 minutes ago		cool_colden		
3b04441a283e	busybox	"echo "Hello อรรถมาภ..."	12 hours ago	Exited (0)
12 hours ago		dreamy_ardinghelli		
c5c41df745a4	busybox	"sh"	12 hours ago	Exited (0)
12 hours ago		nostalgic_chebyshev		
b3d1cd495fe4	busybox	"sh"	12 hours ago	Exited (0)
12 hours ago		serene_newton		
773a2d0f98b7	busybox	"sh"	12 hours ago	Up 12 hour
s		lucid_albattani		

แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_2
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_2 เพื่อใช้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ (Windows) บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

```
FROM busybox
```

```
CMD echo "Hi there. This is my first docker image."
```

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal และป้อนคำสั่งต่อไปนี้

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

```
FROM busybox
```

```
CMD echo "Hi there. This is my first docker image."
```

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

```
EOF
```

หรือใช้คำสั่ง

```
$ touch Dockerfile
```

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

```
$ docker build -t <ชื่อ Image> .
```

6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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

```
C:\653380219-3\Lab8_2>docker build -t dockerfile .
[+] Building 0.0s (5/5) FINISHED
=> [internal] load build definition from dockerfile
=> => transferring dockerfile: 335B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/busybox:latest
=> [1/1] FROM docker.io/library/busybox
=> exporting to image
=> => exporting layers
=> => writing image sha256:4a6f3b2939175c34bd04439ed11f42fcb664858550fa3dc1df7ec4e27a37101
=> => naming to docker.io/library/dockerfile
```

```
C:\653380219-3\Lab8_2>docker run dockerfile
"Hi there. My work is done. You can run them from my Docker image."
```

- (1) คำสั่งที่ใช้ในการ run คือ
docker run dockerfile
- (2) Option -t ในคำสั่ง \$ docker build ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
จะทำการตั้งชื่อและ tag ของ image ตาม parameter ที่ใส่ตามหลัง option

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."
```



## CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

### Lab Worksheet

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

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

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



## CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

### Lab Worksheet

```
Terminal
PS C:\653380219-3\Lab8_3> docker build -t atjamaporn/lab8 .
[+] Building 0.1s (5/5) FINISHED docker:desktop-linux
=> [internal] load build definition from dockerfile 0.0s
=> => transferring dockerfile: 190B 0.0s
=> WARN: JSONArgsRecommended: JSON arguments rec 0.0s
=> WARN: MultipleInstructionsDisallowed: Multipl 0.0s
=> WARN: JSONArgsRecommended: JSON arguments rec 0.0s
=> [internal] load metadata for docker.io/librar 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> CACHED [1/1] FROM docker.io/library/busybox:l 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:c94885b7c2ebfbdac20e4 0.0s
=> => naming to docker.io/atjamaporn/lab8 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/9uw9ubvjqrk4i6h3e5l7dagm

3 warnings found (use docker --debug to expand):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)

PS C:\653380219-3\Lab8_3> docker run atjamaporn/lab8
"Atjamaporn Thawornpisandilok 653380219-3"
PS C:\653380219-3\Lab8_3> █
```

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการให้คำสั่ง

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

ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

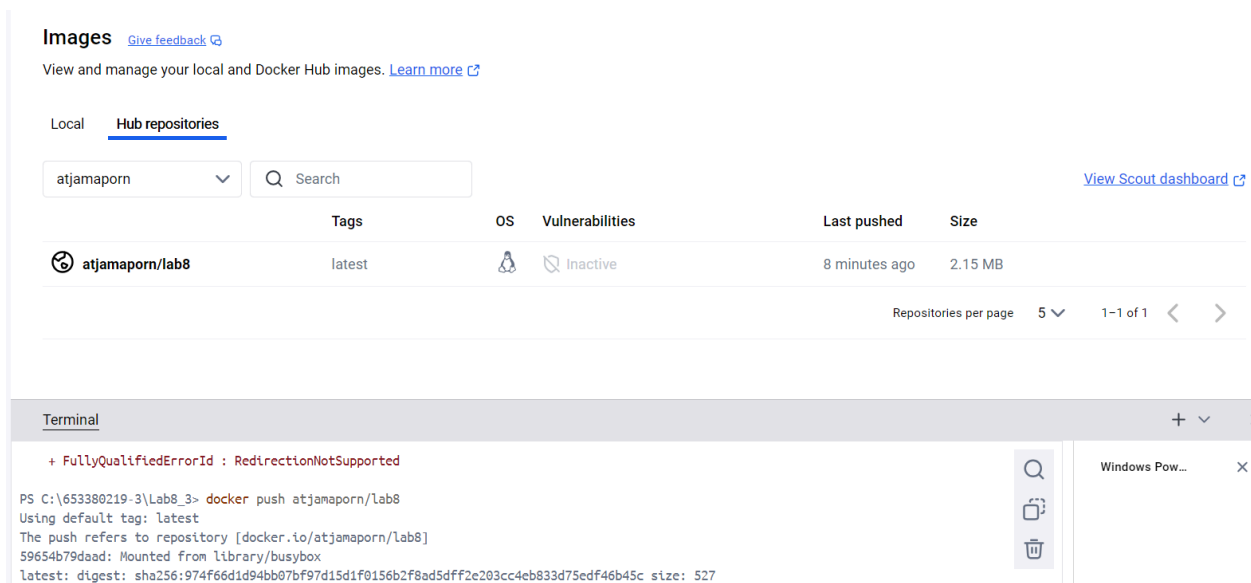
\$ docker login แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง

\$ 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



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

# CP353004/SC313 004 Software Engineering (2/2567)


ผศ.ดร.ชิตส์ฐา สุ่มเล็ก

## Lab Worksheet

### package.json

```
C:\653380219-3\Lab8_3>cd C:\653380219-3\Lab8_4

C:\653380219-3\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 | 5.77 MiB/s, done.
Resolving deltas: 100% (523/523), done.
```

```
C: > 653380219-3 > Lab8_4 > getting-started > app >  package.json > {} prettier
```

```
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 }
```

## 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**

```
C:\653380219-3\Lab8_4\getting-started\app>docker build -t myapp_6533802193 .
[+] Building 24.6s (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 4.6s
=> [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@sha256:974afb6cbc0314dc 5.8s
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc 0.0s
=> => sha256:37892ffbfcaa871a10f813803949d18c3015a48 40.01MB / 40.01MB 4.2s
=> => sha256:5650d6de56fd0bb419872b876ac1df28f577b3957 1.26MB / 1.26MB 1.8s
=> => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d97 7.67kB / 7.67kB 0.0s
=> => sha256:6e804119c3884fc5782795bf0d2adc89201c63105 1.72kB / 1.72kB 0.0s
=> => sha256:dcfb7b337595be6f4d214e4eed84f230eefe0e4ac 6.18kB / 6.18kB 0.0s
=> => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2 3.64MB / 3.64MB 1.2s
=> => extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61 0.2s
=> => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06 444B / 444B 1.6s
=> => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51 1.3s
=> => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b7 0.0s
=> => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b 0.0s
=> [internal] load build context 0.2s
=> => transferring context: 4.62MB 0.2s
=> [2/4] WORKDIR /app 0.4s
=> [3/4] COPY . . 0.0s
=> [4/4] RUN yarn install --production 12.9s
=> exporting to image 0.7s
=> => exporting layers 0.7s
=> => writing image sha256:6dac9399e796a3e02b5ba4b94012a77d2b8ea825fb1 0.0s
=> => naming to docker.io/library/myapp_6533802193 0.0s
```

View build details: [docker-desktop://dashboard/build/desktop-linux/desktop-linux/3wros6oylpdwr7q443bgkcy01](https://docker-desktop://dashboard/build/desktop-linux/desktop-linux/3wros6oylpdwr7q443bgkcy01)

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## Lab Worksheet

\$ 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

```
C:\653380219-3\Lab8_4\getting-started\app>docker build -t myapp_6533802193 .
[+] Building 24.6s (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 4.6s
=> [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@sha256:974afb6cbc0314dc 5.8s
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc 0.0s
=> => sha256:37892ffbfcaa871a10f813803949d18c3015a48 40.01MB / 40.01MB 4.2s
=> => sha256:5650d6de56fd0bb419872b876ac1df28f577b3957 1.26MB / 1.26MB 1.8s
=> => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d97 7.67kB / 7.67kB 0.0s
=> => sha256:6e804119c3884fc5782795bf0d2adc89201c63105 1.72kB / 1.72kB 0.0s
=> => sha256:dcbf7b337595be6f4d214e4eed84f230eefe0e4ac 6.18kB / 6.18kB 0.0s
=> => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2 3.64MB / 3.64MB 1.2s
=> => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06 444B / 444B 1.6s
=> => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51 1.3s
=> => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b7 0.0s
=> => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b 0.0s
=> [internal] load build context 0.2s
=> => transferring context: 4.62MB 0.2s
=> [2/4] WORKDIR /app 0.4s
=> [3/4] COPY . . 0.0s
=> [4/4] RUN yarn install --production 12.9s
=> exporting to image 0.7s
=> => exporting layers 0.7s
=> => writing image sha256:6dac9399e796a3e02b5ba4b94012a77d2b8ea825fb1 0.0s
=> => naming to docker.io/library/myapp_6533802193 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/3wros6oylpdwr7q443bgkcy0l.....

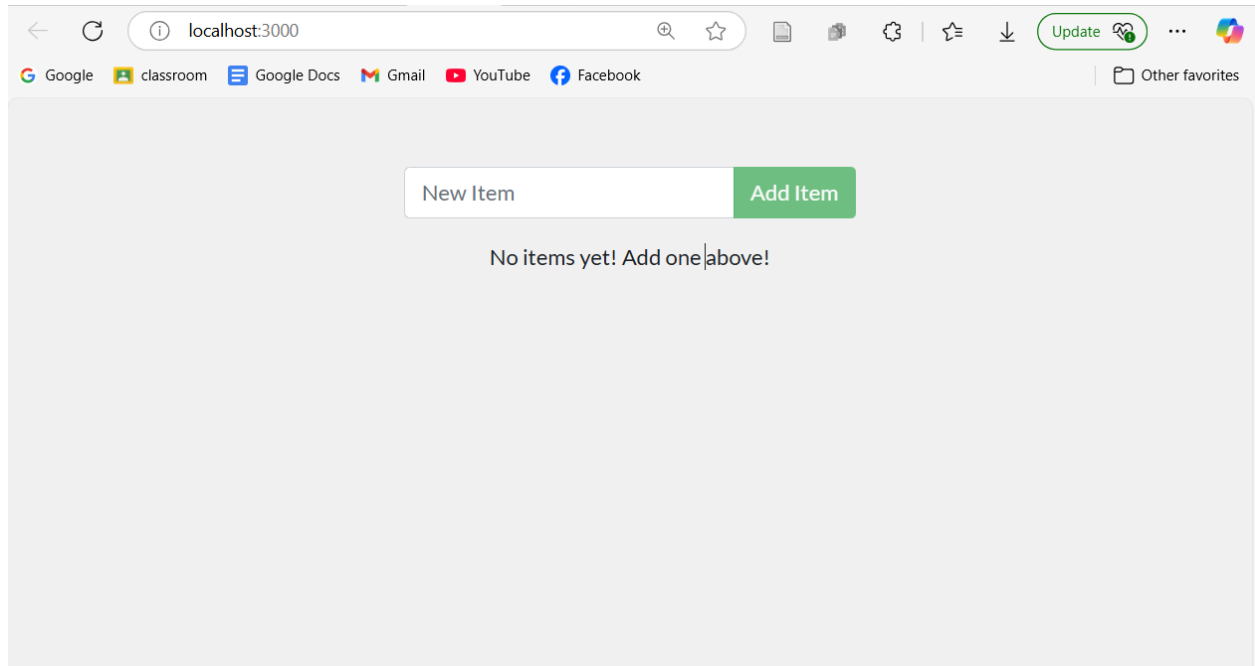
C:\653380219-3\Lab8_4\getting-started\app>
C:\653380219-3\Lab8_4\getting-started\app>docker run -dp 3000:3000 myapp_6533802193
507dbf3aa29eb5ba0c2b2a2fa1e9a047a9b299dff3f8bef235e3294e50b0d671

C:\653380219-3\Lab8_4\getting-started\app>|
```

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## 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</p>`

b. Save ไฟล์ให้เรียบร้อย

9. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

10. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## Lab Worksheet

**[Check point#10]** Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ พร้อมกับตอบคำถามต่อไปนี้

```
C:\653380219-3\Lab8_4\getting-started\app>docker build -t myapp_6533802193 .
[+] Building 16.9s (10/10) FINISHED
=> [internal] load build definition from dockerfile
=> transferring dockerfile: 156B
=> [internal] load metadata for docker.io/library/node:18-alpine
=> [auth] library/node:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> transferring context: 2B
=> [internal] load build context
=> transferring context: 8.15kB
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY .
=> [4/4] RUN yarn install --production
=> exporting to image
=> exporting layers
=> writing image sha256:fb02c4a5795ead926c10cea27a7cb705ac3468780271d8b20cacabc105d53052
=> naming to docker.io/library/myapp_6533802193
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/ua0za4xzk5iito6kkmnfgwzq
C:\653380219-3\Lab8_4\getting-started\app>docker run -dp 3000:3000 myapp_6533802193
e21d9e57a88ecd1b8318e5b09e311d8160209164272a36644a76343079b4a145
docker: Error response from daemon: driver failed programming external connectivity on endpoint quizzical_jepsen (1095e8c2a3f5e07de8c60c3a20c5f81baec4e16b0d7f43f54b75bd3c059d0de9): Bind for 0.0.0.0:3000 failed: port is already allocated.
C:\653380219-3\Lab8_4\getting-started\app>
```

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร  
error นี้เกิดขึ้นเพราะว่า port 3000 บนเครื่องมีการใช้งานอยู่ เมื่อทำการรันอีกครั้งทำให้ไม่สามารถใช้งานได้ จึงต้องหยุดการใช้งานดังกล่าวก่อน จึงจะเริ่ม รันใหม่อีกครั้ง

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



## CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

### Lab Worksheet

ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ

iii. ยืนยันโดยการกด Delete forever

12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6

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

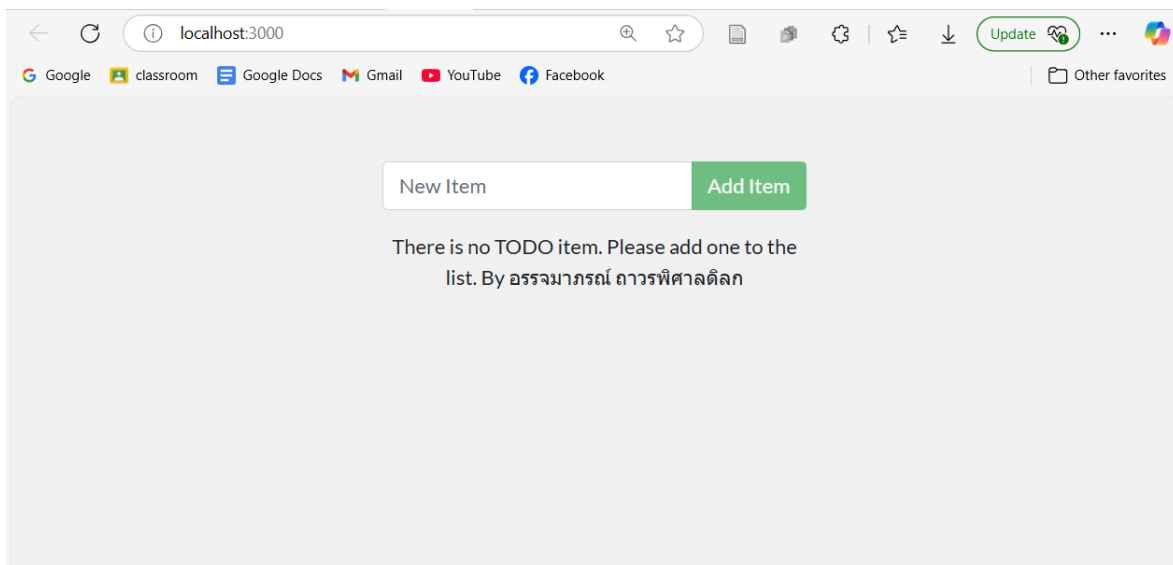
**[Check point#11]** Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้บน Browser และ Dashboard ของ Docker desktop

```
C:\653380219-3\Lab8_4\getting-started\app>docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
507dbf3aa29e 6dac9399e796 "docker-entrypoint.s..." 4 hours ago Up 4 hours 0.0.0.0
:3000->3000/tcp stupefied_wiles
773a2d0f98b7 busybox "sh" 20 hours ago Up 20 hours
lucid_albattani

C:\653380219-3\Lab8_4\getting-started\app>
C:\653380219-3\Lab8_4\getting-started\app>docker stop 507dbf3aa29e
507dbf3aa29e

C:\653380219-3\Lab8_4\getting-started\app>docker rm 507dbf3aa29e
507dbf3aa29e

C:\653380219-3\Lab8_4\getting-started\app>docker run -dp 3000:3000 myapp_6533802193
f9c2e3a1ae87bc0ecb939c0186857db52b7b78a8d6ff37705b518f0357be34b3
```



## 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**

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## Lab Worksheet

```
2025-01-29 10:12:59.190+0000 [id=45] INFO jenkins.InitReactorRunner$1onAttained: Started all plugins
2025-01-29 10:12:59.191+0000 [id=50] INFO jenkins.InitReactorRunner$1onAttained: Augmented all extensions
2025-01-29 10:12:59.316+0000 [id=35] INFO jenkins.InitReactorRunner$1onAttained: System config loaded
2025-01-29 10:12:59.317+0000 [id=44] INFO jenkins.InitReactorRunner$1onAttained: System config adapted
2025-01-29 10:12:59.317+0000 [id=45] INFO jenkins.InitReactorRunner$1onAttained: Loaded all jobs
2025-01-29 10:12:59.319+0000 [id=54] INFO jenkins.InitReactorRunner$1onAttained: Configuration for all jobs updated
2025-01-29 10:12:59.343+0000 [id=68] INFO hudson.util.Retrier#start: Attempt #1 to do the action check updates server
2025-01-29 10:12:59.643+0000 [id=50] INFO jenkins.install.SetupWizard#init:

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

32bb9121fd724d33bc1203978dfa086a

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

2025-01-29 10:13:06.894+0000 [id=48] INFO jenkins.InitReactorRunner$1onAttained: Completed initialization
2025-01-29 10:13:06.908+0000 [id=25] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
2025-01-29 10:13:09.004+0000 [id=68] INFO h.m.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2025-01-29 10:13:09.004+0000 [id=68] INFO hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1
2025-01-29 10:15:42.413+0000 [id=98] INFO hudson.PluginManager#install: Starting installation of a batch of 20 plugins plus their dependencies
2025-01-29 10:15:42.415+0000 [id=98] INFO hudson.model.UpdateSite$Plugin#deploy: Adding dependent installation of ionicons-api for plugin cloudbees-folder
2025-01-29 10:15:42.417+0000 [id=109] INFO h.m.model.UpdateCenter$DownloadJob#run: Starting the installation of ionicons-api on behalf of admin
2025-01-29 10:15:42.417+0000 [id=98] INFO hudson.model.UpdateSite$Plugin#deploy: Adding dependent installation of json-path-api for plugin build-timeout
2025-01-29 10:15:42.418+0000 [id=98] INFO hudson.model.UpdateSite$Plugin#deploy: Adding dependent installation of asm-api for plugin json-path-api
2025-01-29 10:15:42.418+0000 [id=98] INFO hudson.model.UpdateSite$Plugin#deploy: Adding dependent installation of token-macro for plugin build-timeout
```

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

5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3

6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษา พร้อมรหัสสี่ตัวท้าย เช่น somsri\_3062

**[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า**

---

Getting Started

---

## Create First Admin User

Username

atjamaporn\_2193

Password

.....

Confirm password

.....

Full name

Atjamaporn Thawompisandilok

E-mail address

Atjamaporn.t@kkumail.com

---

Jenkins 2.479.3

[Skip and continue as admin](#)

[Save and Continue](#)

---

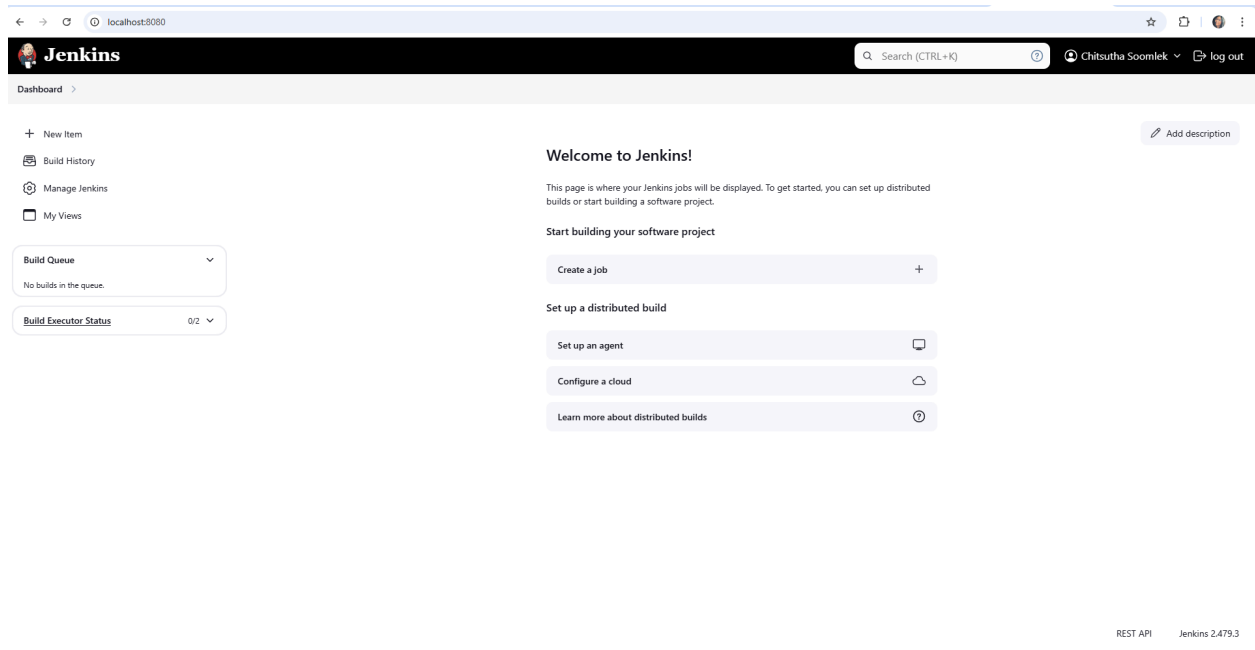
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ

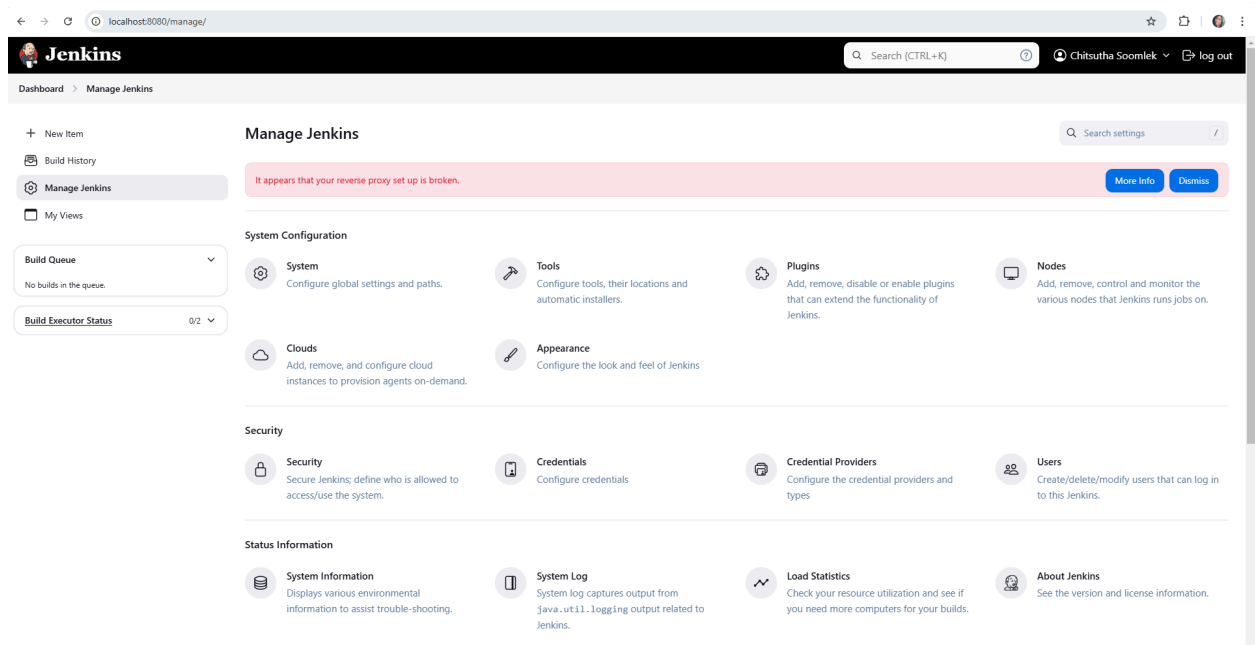
# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## Lab Worksheet



## 9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins



# CP353004/SC313 004 Software Engineering (2/2567)

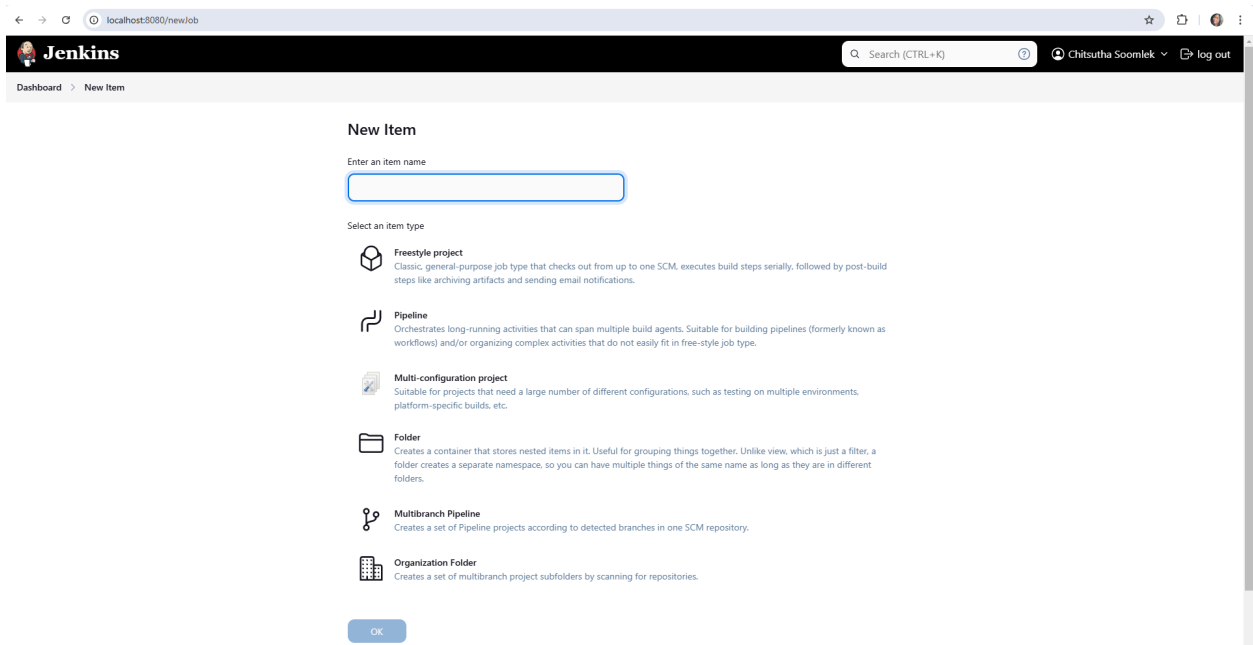
ผศ.ดร.ชิตสุธา สุ่มเล็ก

## 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

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## Lab Worksheet

**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

The screenshot shows the GitHub Actions configuration page for a workflow named "Lab 8.5". The "Description" field contains the text "Lab 8.5". Below this, there are several configuration options:

- ☐ Discard old builds ?
- ☒ GitHub project
  - Project url ?
  - Advanced ▾
- ☐ This project is parameterized ?
- ☐ Throttle builds ?
- ☐ Execute concurrent builds if necessary ?
- Advanced ▾

Under the "Source Code Management" section, the "None" option is selected with a radio button, and the "Git" option is also visible with a question mark icon.



## CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

### Lab Worksheet

#### Build Triggers

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

Schedule ?

H/15 \* \* \* \*

Would last have run at Wednesday, January 29, 2025 at 3:59:49 PM  
Coordinated Universal Time; would next run at Wednesday, January 29, 2025  
at 4:14:49 PM Coordinated Universal Time.

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

#### 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

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## Lab Worksheet

Command

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

```
robot invalid_login.robot
```

Advanced ▼

Add build step ▼

### 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)

Advanced ▼

Edited

Thresholds for build result ?

🟡 %

# CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

## 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)

Advanced ▾

Edited

Thresholds for build result ?

🟡 %

🟢 %

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

☐ Include skipped tests in total count for thresholds

Add post-build action ▾

Save

Apply

## CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

### Lab Worksheet

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

**Post-build action:** เพิ่ม Publish Robot Framework test results -> ระบุไดเรกทอรีที่เก็บไฟล์ผลการทดสอบโดย Robot framework ในรูป xml และ html -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ไม่ผ่านแล้ว นับว่าซอฟต์แวร์มีปัญหา -> 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**

#### ⊗ Console Output

[Download](#)[Copy](#)[View as plain text](#)

```
Started by user Atjamaporn Thawornpisandilok
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT2
[UAT2] $ /bin/sh -xe /tmp/jenkins6766084038224990764.sh
+ robot invalid_login.robot
/tmp/jenkins6766084038224990764.sh: 2: robot: not found
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:
Failed!
hudson.AbortException: No files found in path /var/jenkins_home/workspace/UAT2 with configured filemask:
output.xml
 at PluginClassLoader for
robot//hudson.plugins.robot.RobotParser$RobotParserCallable.invoke(RobotParser.java:81)
 at PluginClassLoader for
robot//hudson.plugins.robot.RobotParser$RobotParserCallable.invoke(RobotParser.java:52)
 at hudson.FilePath.act(FilePath.java:1234)
 at hudson.FilePath.act(FilePath.java:1217)
 at PluginClassLoader for robot//hudson.plugins.robot.RobotParser.parse(RobotParser.java:48)
 at PluginClassLoader for
robot//hudson.plugins.robot.RobotPublisher.parse(RobotPublisher.java:262)
 at PluginClassLoader for
robot//hudson.plugins.robot.RobotPublisher.perform(RobotPublisher.java:286)
 at hudson.tasks.BuildStepCompatibilityLayer.perform(BuildStepCompatibilityLayer.java:80)
 at hudson.tasks.BuildStepMonitor$1.perform(BuildStepMonitor.java:20)
 at hudson.model.AbstractBuild$AbstractBuildExecution.perform(AbstractBuild.java:818)
 at
hudson.model.AbstractBuild$AbstractBuildExecution.performAllBuildSteps(AbstractBuild.java:767)
 at hudson.model.Build$BuildExecution.post2(Build.java:179)
```