

ชื่อ-นามสกุล นางสาวอชิรัชญา บุญประชม รหัสนักศึกษา 653380157-9 Section 1

Lab#8 – Software Deployment Using Docker

วัตถุประสงค์การเรียนรู้

1. ผู้เรียนสามารถอธิบายเกี่ยวกับ Software deployment ได้
2. ผู้เรียนสามารถสร้างและรัน Container จาก Docker image ได้
3. ผู้เรียนสามารถสร้าง Docker files และ Docker images ได้
4. ผู้เรียนสามารถนำซอฟต์แวร์ที่พัฒนาขึ้นให้สามารถรันบนสภาพแวดล้อมเดียวกันและทำงานร่วมกันกับสมาชิกในทีมพัฒนาซอฟต์แวร์ผ่าน Docker hub ได้
5. ผู้เรียนสามารถเริ่มต้นใช้งาน Jenkins เพื่อสร้าง Pipeline ในการ Deploy งานได้

Pre-requisite

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

แบบฝึกปฏิบัติที่ 8.1 Hello world - รัน Container จาก Docker image

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
1. เปิด Command line หรือ Terminal บน Docker Desktop จากนั้นสร้าง Directory ชื่อ Lab8_1

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

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

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

```
Terminal
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
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
busybox        latest    af4709625109   4 months ago   4.27MB
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1>
```

- (1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร
 - Image name
- (2) Tag ที่ใช้บ่งบอกถึงอะไร
 - version

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"

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

```
Terminal
v  etc  home  lib  lib64  proc  root  sys  tmp  usr  var
/ # ls -la
total 48
drwxr-xr-x 1 root root 4096 Jan 28 14:50 .
drwxr-xr-x 1 root root 4096 Jan 28 14:50 ..
drwxr-xr-x 1 root root 4096 Jan 28 14:50 bin
drwxr-xr-x 1 root root 4096 Jan 28 14:50 boot
drwxr-xr-x 1 root root 4096 Jan 28 14:50 dev
drwxr-xr-x 1 root root 4096 Jan 28 14:50 etc
drwxr-xr-x 1 root root 4096 Jan 28 14:50 home
drwxr-xr-x 1 root root 4096 Jan 28 14:50 lib
drwxr-xr-x 1 root root 4096 Jan 28 14:50 lib64
drwxr-xr-x 1 root root 4096 Jan 28 14:50 media
drwxr-xr-x 1 root root 4096 Jan 28 14:50 mnt
drwxr-xr-x 1 root root 4096 Jan 28 14:50 opt
drwxr-xr-x 1 root root 4096 Jan 28 14:50 root
drwxr-xr-x 1 root root 4096 Jan 28 14:50 run
drwxr-xr-x 1 root root 4096 Jan 28 14:50 sbin
drwxr-xr-x 1 root root 4096 Jan 28 14:50 srv
drwxr-xr-x 1 root root 4096 Jan 28 14:50 sys
drwxr-xr-x 1 root root 4096 Jan 28 14:50 tmp
drwxr-xr-x 1 root root 4096 Jan 28 14:50 usr
drwxr-xr-x 1 root root 4096 Jan 28 14:50 var

PS C:\Users\Achiruttha\desktop\Y3\Lab8_1> docker run busybox echo "Hello Achiruttha Boonprachom from busybox"
Hello Achiruttha Boonprachom from busybox
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1> docker ps -a
docker: 'ps -a' is not a docker command.
See 'docker --help'
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1> docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS   NAMES
52d29b8b4025   busybox    "echo 'Hello Achirut..." 2 minutes ago  Exited (0)   2 minutes ago  brave_kare
b5fadabe5ba0   busybox    "sh"                    4 minutes ago  Exited (0)   3 minutes ago  bold_shtern
a6e3ba9b71d0   busybox    "sh"                    6 minutes ago  Exited (0)   6 minutes ago  elastic_goldberg
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1>
```

- (1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
 - จะสามารถใช้ command ใน container ได้โดยตรง
- (2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร
 - แสดงสถานะของแต่ละ container โดยบอกถึง สถานะการทำงาน ปัจจุบันหรือที่ผ่านมา ของ container นั้น ๆ

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

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

```
a6e3ba9b71d0 busybox "sh" 6 minutes ago Exited (0) 6 minutes ago elastic_goldberg
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1> docker rm a6e3ba9b71d00112ef7157448e802c2f3d838580af7dab4b157412a4b0c36745
a6e3ba9b71d00112ef7157448e802c2f3d838580af7dab4b157412a4b0c36745
PS C:\Users\Achiruttha\desktop\Y3\Lab8_1>
```

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

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

หรือใช้คำสั่ง

\$ touch Dockerfile

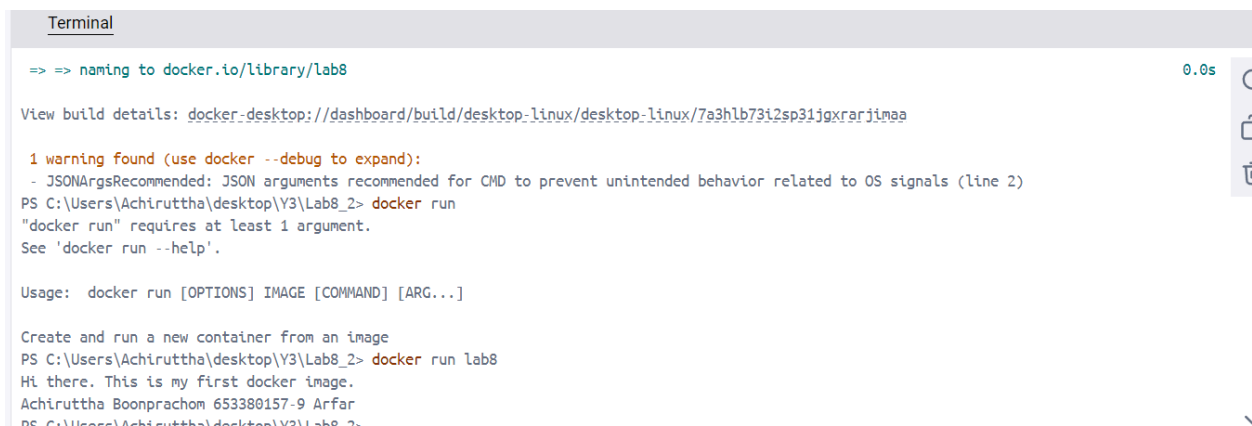
แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

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 (ทั้งหน้าตาและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ในขั้นตอนที่ 5 พร้อมกับตอบคำถามต่อไปนี้



```
Terminal
=> => naming to docker.io/library/lab8 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/7a3hlb73i2sp31jgxrarjima

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\Achiruttha\desktop\Y3\Lab8_2> docker run
"docker run" requires at least 1 argument.
See 'docker run --help'.

Usage: docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

Create and run a new container from an image
PS C:\Users\Achiruttha\desktop\Y3\Lab8_2> docker run lab8
Hi there. This is my first docker image.
Achiruttha Boonprachom 653380157-9 Arfar
PS C:\Users\Achiruttha\desktop\Y3\Lab8_2>
```

(1) คำสั่งที่ใช้ในการ run คือ

- docker run lab8

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

- ตั้งชื่อและติด tag ให้กับ Docker image ที่สร้างขึ้นมา

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

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

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet

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

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

```
Terminal
=> [internal] load metadata for docker.io/library/busybox:latest 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> CACHED [1/1] FROM docker.io/library/busybox:latest 0.0s
=> exporting to image 0.1s
=> exporting layers 0.0s
=> writing image sha256:c7d20278201394486f2465ee50239ad82849077749c86c4225966ea077b2547a 0.0s
=> naming to docker.io/achiruttha/lab8 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/lgpuay4nbvmtkry3unk0qi7qw

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\Achiruttha\desktop\Y3\Lab8_3> docker run achiruttha/lab8
Hi there. My work is done. You can run them from my Docker image.
Achiruttha Boonprachom 653380157-9
PS C:\Users\Achiruttha\desktop\Y3\Lab8_3>
```

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>

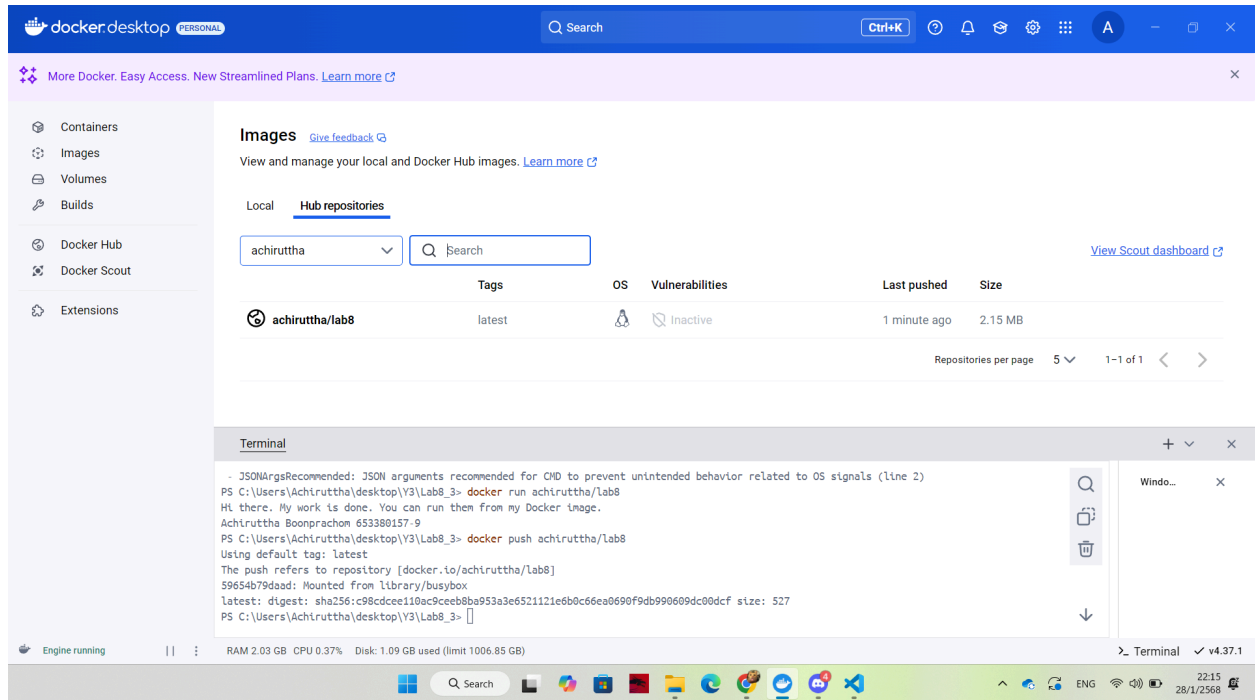
7. ไปที่ Docker Hub กด Tab ชื่อ Tags หรือไปที่ Repository ก็ได้

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

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

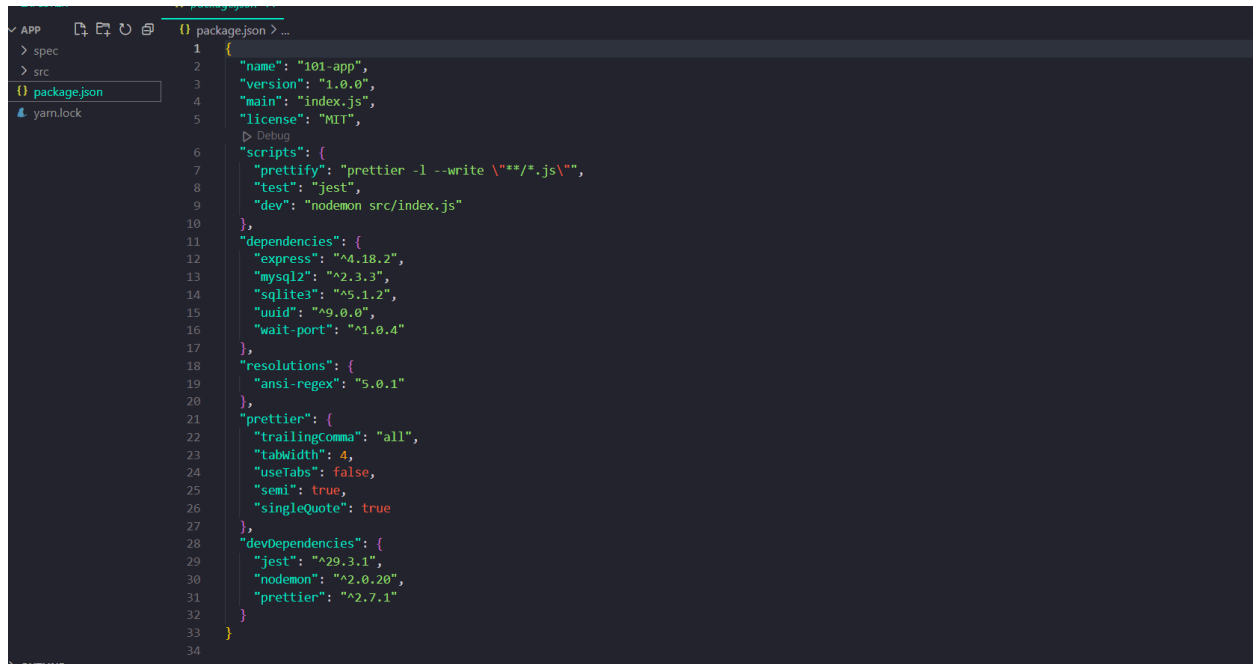
Lab Worksheet



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



```
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 }
```

4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงในไฟล์

FROM node:18-alpine

WORKDIR /app

COPY ..

RUN yarn install --production

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

EXPOSE 3000

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

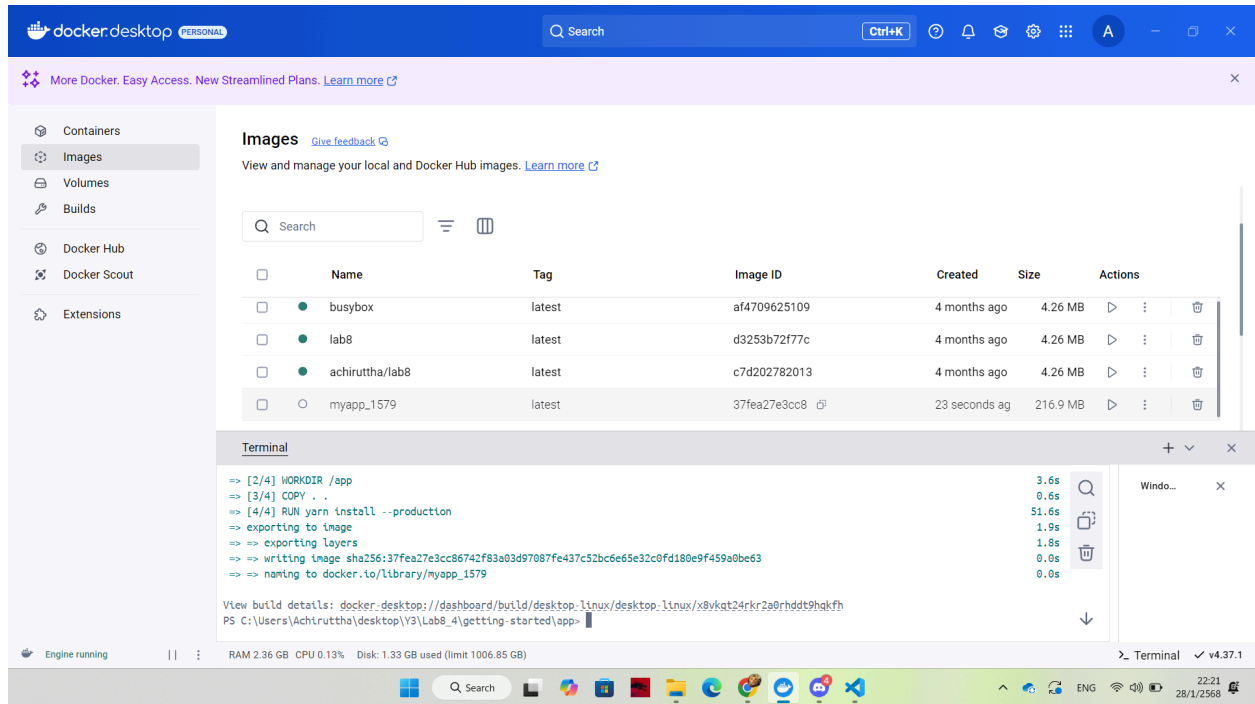
\$ docker build -t <myapp_รหัสสนศ. ไม่มีขีด> .

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

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet



6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

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

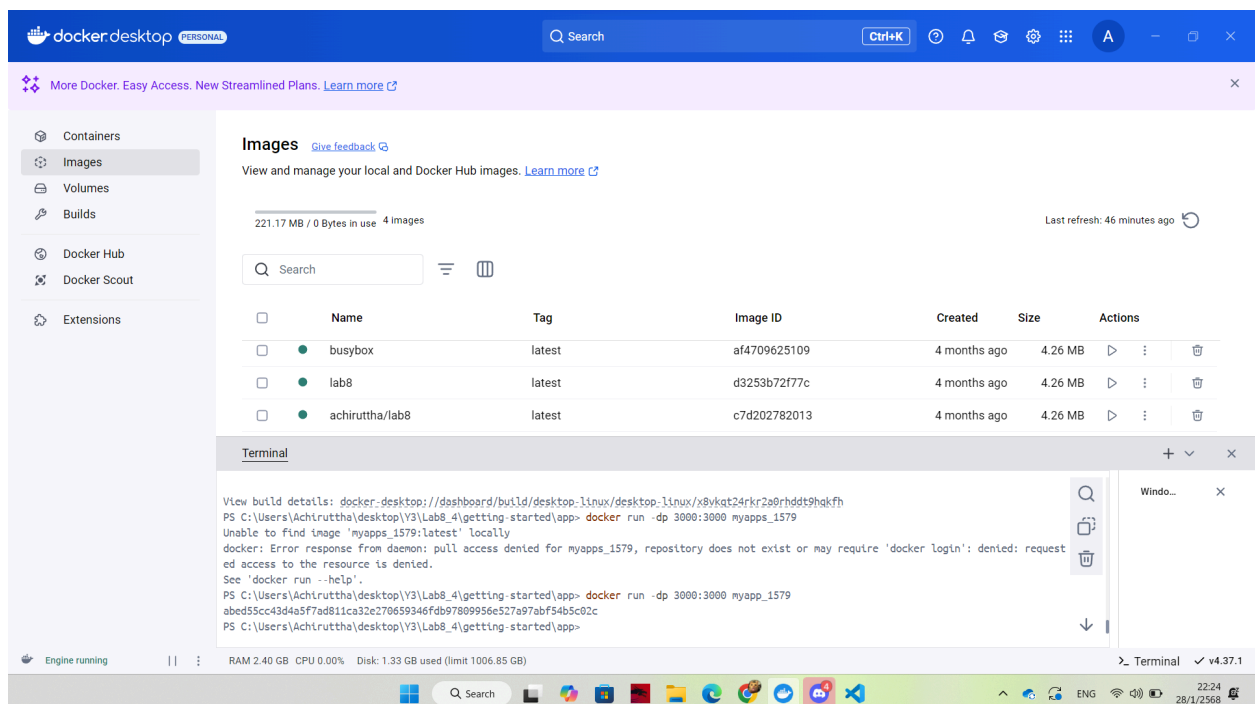
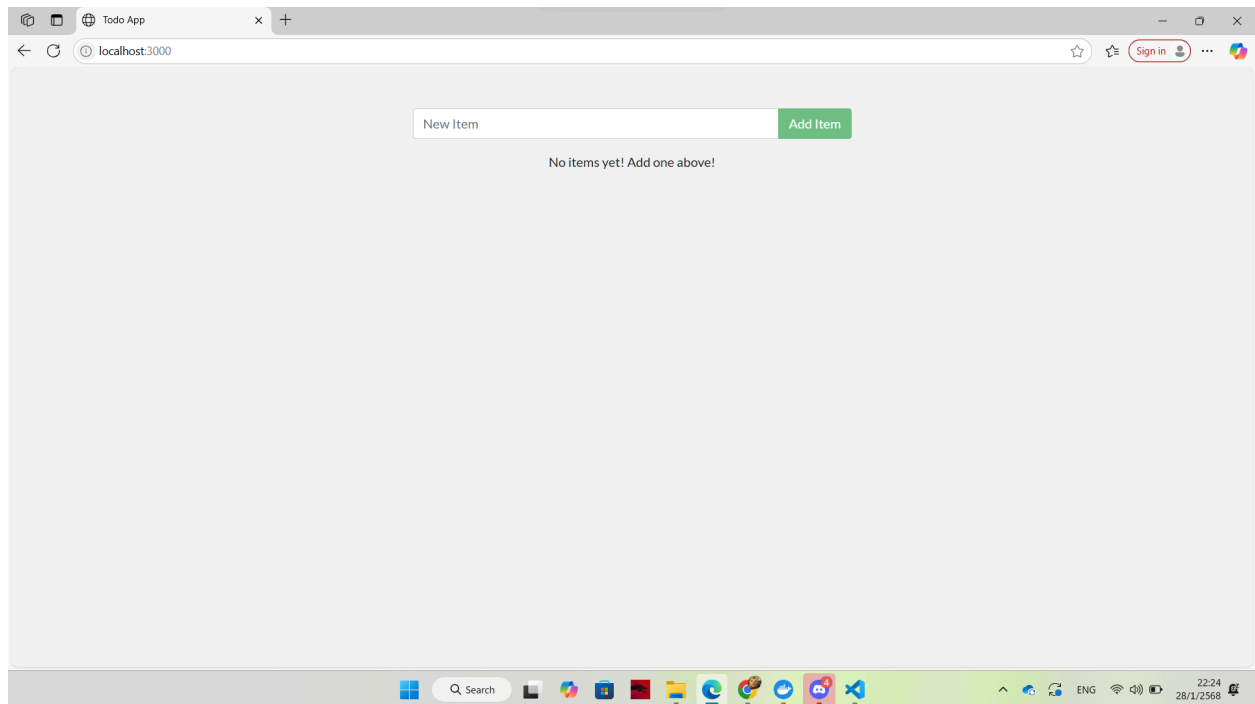
7. เปิด Browser ไปที่ URL = <http://localhost:3000>

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

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

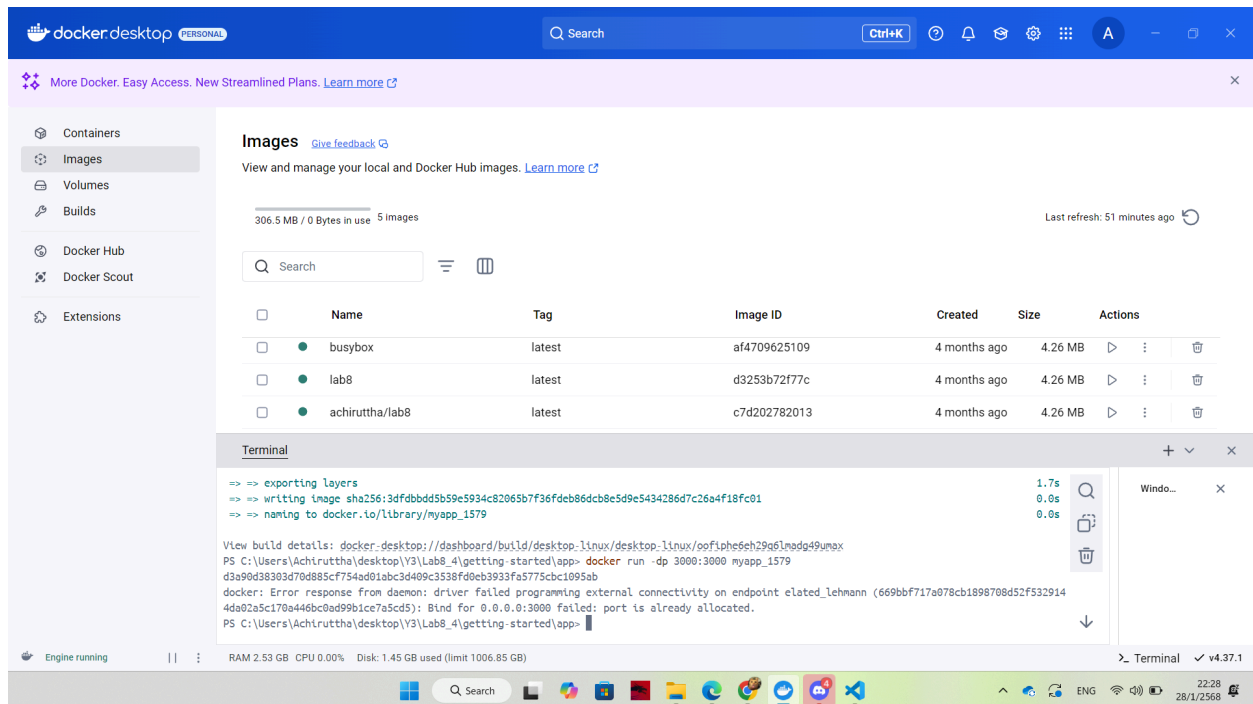
`<p className="text-center">There is no TODO item. Please add one to the list. By ชื่อและนามสกุลของนักศึกษา</p>`

b. Save ไฟล์ให้เรียบร้อย

9. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

10. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

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



(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

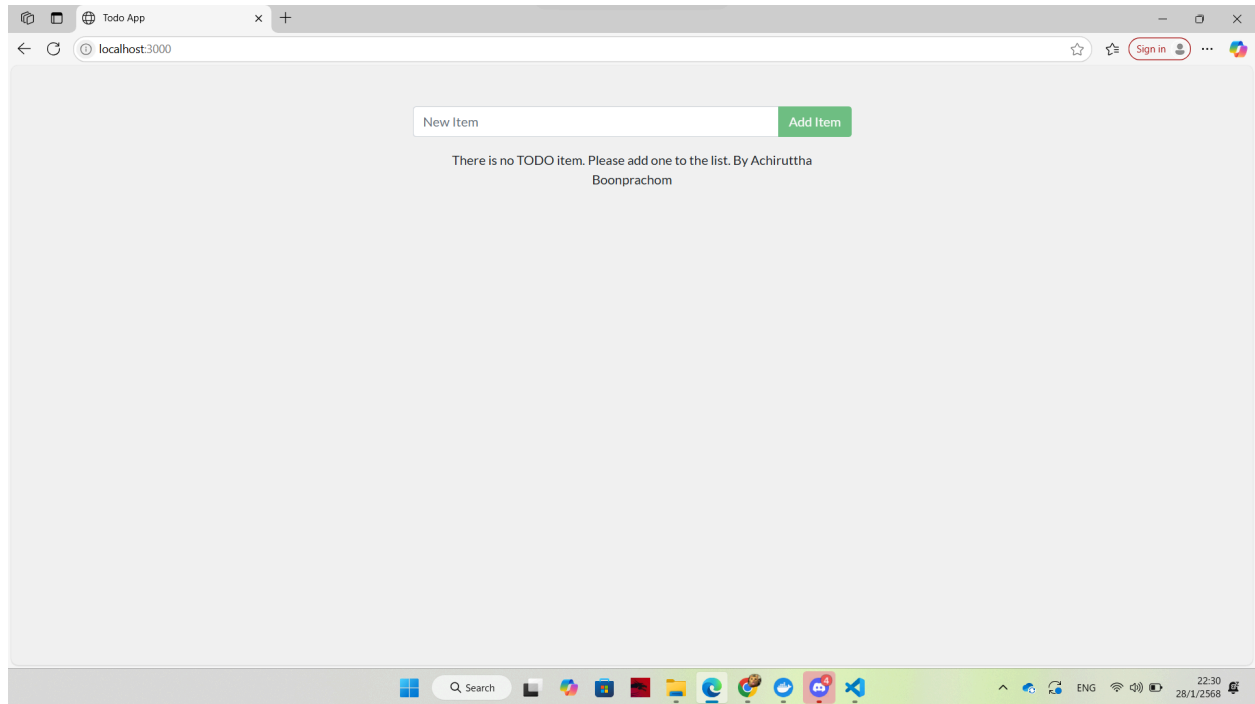
Lab Worksheet

- มีการใช้ port 3000 อยู่แล้ว เพราะมีการ run แต่ไม่ได้ทำการ stop จึงเกิด error ขึ้น
 - 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

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet



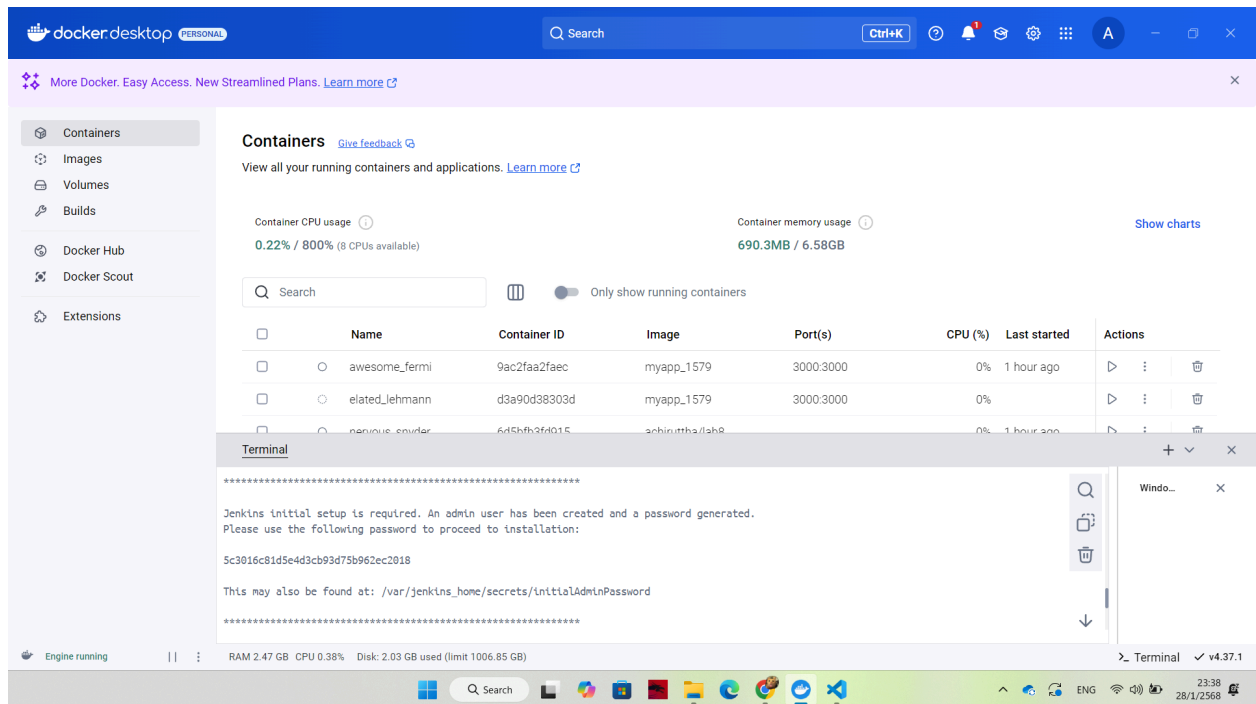
แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

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



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
//ลิม capture หน้าจอ

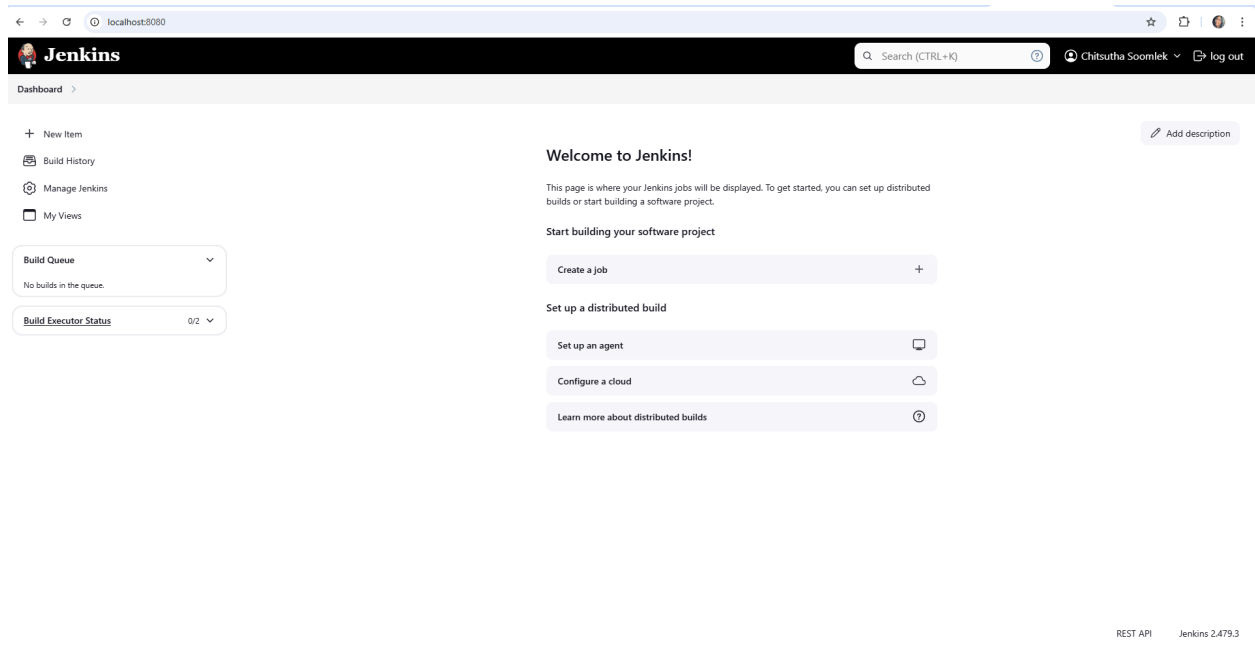
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ

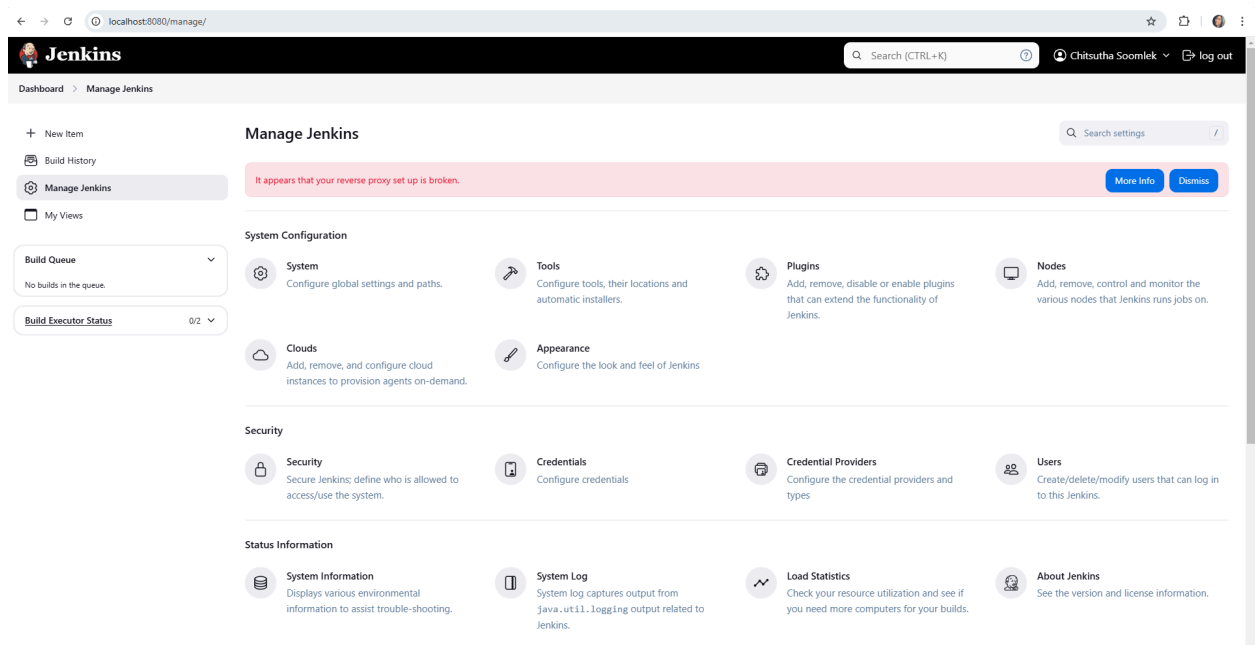
CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins



CP353004/SC313 004 Software Engineering (2/2567)

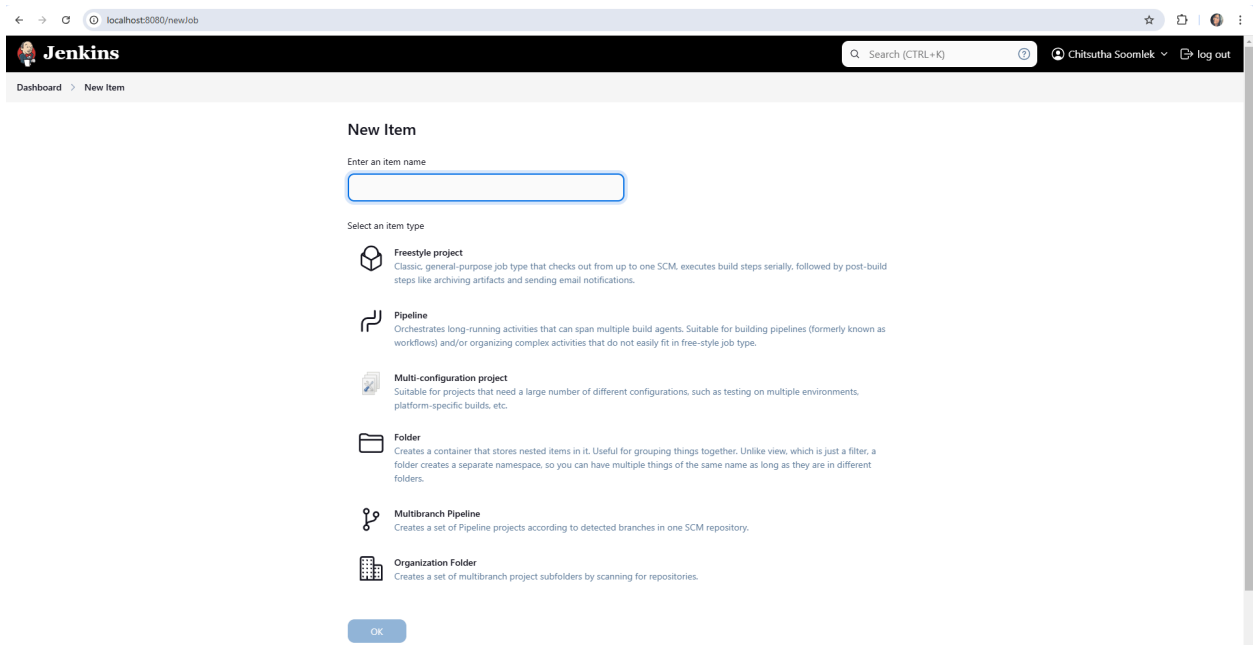
ผศ.ดร.ชิตสุธา สุ่มเล็ก

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

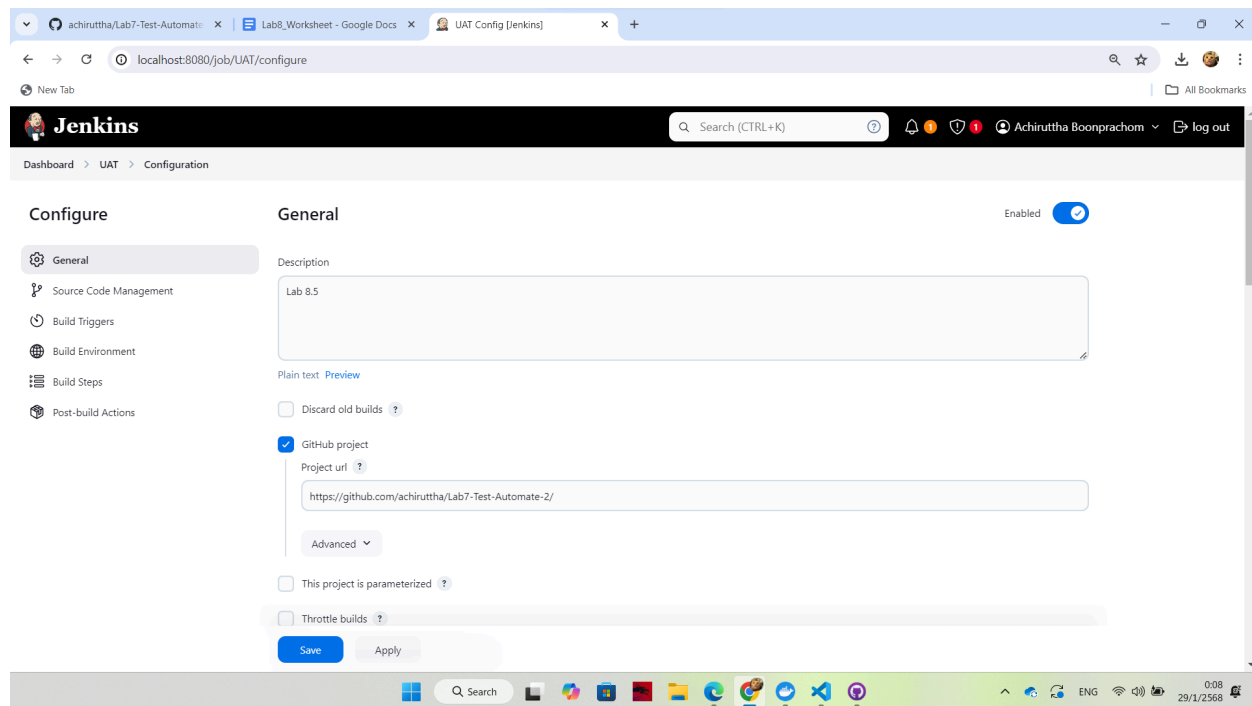
Description: Lab 8.5

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 ไปถึงไฟล์ให้เรียบร้อยแล้ว)

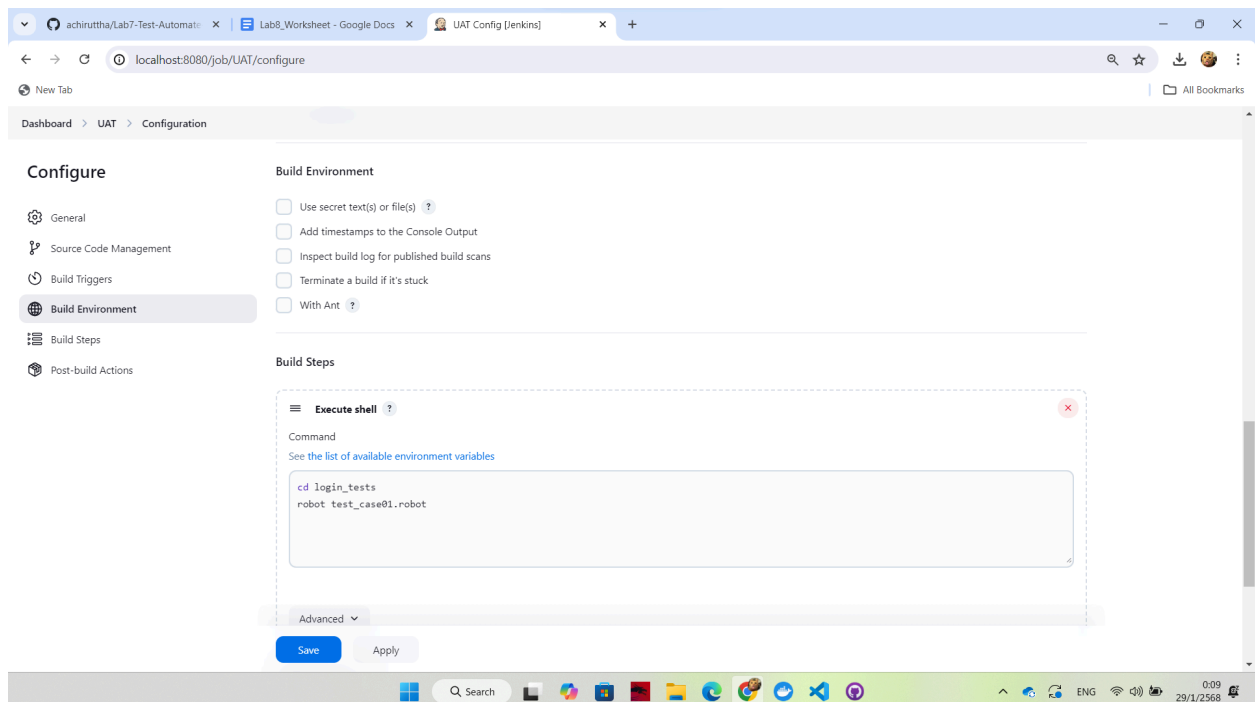
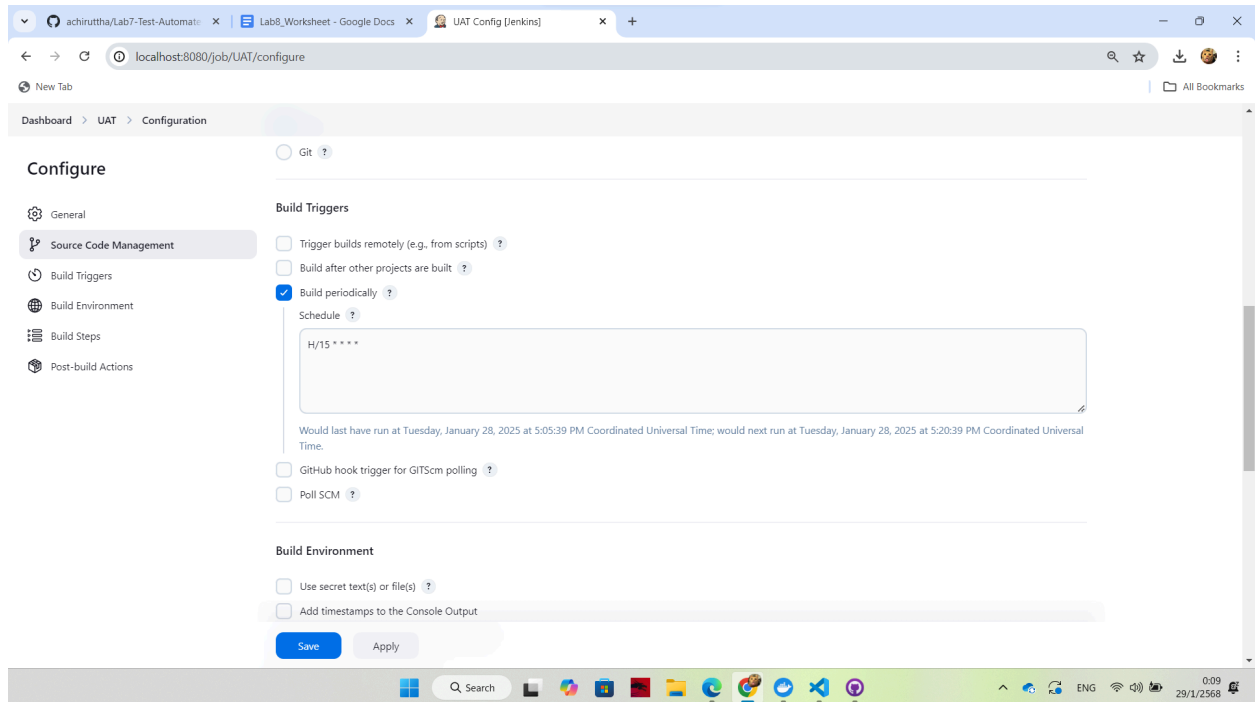
[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้



CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

Lab Worksheet



- (1) คำสั่งที่ใช้ในการ Execute ไฟล์ .robot ใน Build Steps คือ
- robot test_case01.robot

CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตสุธา สุ่มเล็ก

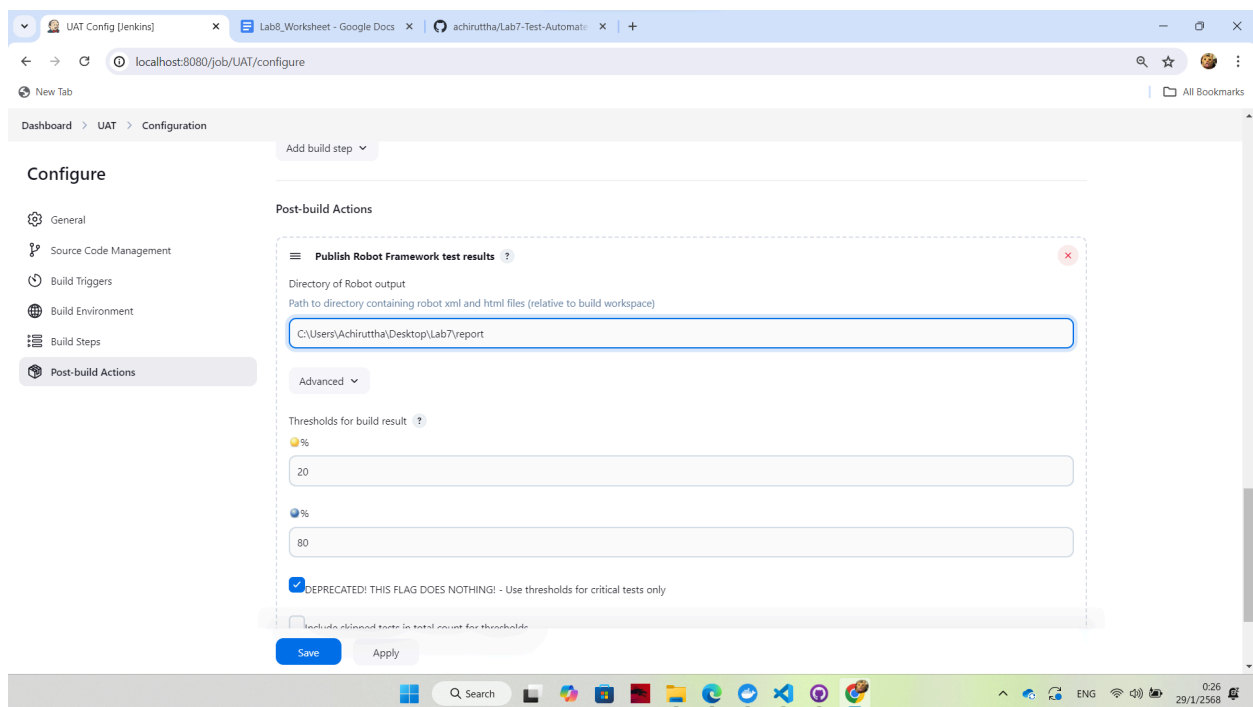
Lab Worksheet

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



CP353004/SC313 004 Software Engineering (2/2567)

ผศ.ดร.ชิตส์ฐา สุ่มเล็ก

Lab Worksheet

