

## Lab Worksheet

ชื่อ-นามสกุล นายนิพัทธ์ ชามักดี รหัสนักศึกษา 653380332-7 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) แสดงผลลัพธ์ที่ได้ พร้อมกับตอบคำถามต่อไปนี้

```
nipatchapakdee@Macintosh-2 Lab8_1 % docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
559c60843878: Pull complete
Digest: sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab073cd3c138bd458fc8257fbf1
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest

What's next:
  View a summary of image vulnerabilities and recommendations → docker scout q
  uickview busybox
nipatchapakdee@Macintosh-2 Lab8_1 % docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
busybox        latest    fc0179a204e2   3 months ago   4.04MB
nipatchapakdee@Macintosh-2 Lab8_1 %
```

- (1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร \_\_\_\_\_ ชื่อ images ในข้อนี้ชื่อ busybox \_\_\_\_\_
- (2) Tag ที่ใช้บ่งบอกถึงอะไร \_\_\_\_\_ version ของ images นั้น ๆ ในข้อนี้จะเป็น latest \_\_\_\_\_
5. ป้อนคำสั่ง \$ docker run busybox
6. ป้อนคำสั่ง \$ docker run -it busybox sh
7. ป้อนคำสั่ง ls
8. ป้อนคำสั่ง ls -la
9. ป้อนคำสั่ง exit
10. ป้อนคำสั่ง \$ docker run busybox echo "Hello ชื่อและนามสกุลของนักศึกษา from busybox"
11. ป้อนคำสั่ง \$ docker ps -a

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

## Lab Worksheet

```
nipatchapakdee@Macintosh-2 Lab8_1 % docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
busybox        latest    fe0179a284e2   3 months ago   4.04MB
nipatchapakdee@Macintosh-2 Lab8_1 % docker run busybox
nipatchapakdee@Macintosh-2 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 23 02:47 .
drwxr-xr-x 1 root    root      4096 Jan 23 02:47 ..
-rwxr-xr-x 1 root    root        0 Jan 23 02:47 .dockerenv
drwxr-xr-x 2 root    root    12288 Sep 26 21:31 bin
drwxr-xr-x 5 root    root      368 Jan 23 02:47 dev
drwxr-xr-x 1 root    root      4096 Jan 23 02:47 etc
drwxr-xr-x 2 nobody  nobody     0 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 218 root   root        0 Jan 23 02:47 proc
drwx----- 1 root    root      4096 Jan 23 02:47 root
dr-xr-xr-x 11 root   root        0 Jan 23 02:47 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
nipatchapakdee@Macintosh-2 Lab8_1 % docker run busybox echo "Nipat Chapakdee 65380332-7 from busybox"
Nipat Chapakdee 65380332-7 from busybox
nipatchapakdee@Macintosh-2 Lab8_1 %
nipatchapakdee@Macintosh-2 Lab8_1 % docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS      PORTS      NAMES
48534d7c1dad   busybox   "echo 'Nipat Chapakd..." About a minute ago   Exited (0) About a minute ago   graceful_khorana
23f4f497ceb9   busybox   "sh"                    3 minutes ago     Exited (0) 2 minutes ago         eloquent_lamport
7fb27d36b7f9   busybox   "sh"                    3 minutes ago     Exited (0) 3 minutes ago         agitated_dhawan
nipatchapakdee@Macintosh-2 Lab8_1 %
nipatchapakdee@Macintosh-2 Lab8_1 % docker rm 7fb27d36b7f9
7fb27d36b7f9
nipatchapakdee@Macintosh-2 Lab8_1 % docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS      PORTS      NAMES
48534d7c1dad   busybox   "echo 'Nipat Chapakd..." 11 minutes ago   Exited (0) 11 minutes ago         graceful_khorana
23f4f497ceb9   busybox   "sh"                    13 minutes ago   Exited (0) 12 minutes ago         eloquent_lamport
```

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

\_\_\_\_\_ หลังจากใช้ option -it จะเป็นการเข้าไปใน images นั้น เข้าไปใช้งาน service ต่าง ๆ ใน images หากไม่ใช้ option -it images นั้นจะรันอยู่พื้นหลังไม่มีการตอบโต้กันกับผู้ใช้ \_\_\_\_\_

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

\_\_\_\_\_ คอลัมน์ STATUS แสดงถึงข้อมูลสถานะปัจจุบัน container ว่าทำงานอยู่อยู่ หรือหยุดทำงานไปแล้ว \_\_\_\_\_

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

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

```
nipatchapakdee@Macintosh-2 Lab8_1 % docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS      PORTS      NAMES
48534d7c1dad   busybox   "echo 'Nipat Chapakd..." About a minute ago   Exited (0) About a minute ago   graceful_khorana
23f4f497ceb9   busybox   "sh"                    3 minutes ago     Exited (0) 2 minutes ago         eloquent_lamport
7fb27d36b7f9   busybox   "sh"                    3 minutes ago     Exited (0) 3 minutes ago         agitated_dhawan
nipatchapakdee@Macintosh-2 Lab8_1 % docker rm 7fb27d36b7f9
7fb27d36b7f9
nipatchapakdee@Macintosh-2 Lab8_1 % docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS      PORTS      NAMES
48534d7c1dad   busybox   "echo 'Nipat Chapakd..." 11 minutes ago   Exited (0) 11 minutes ago         graceful_khorana
23f4f497ceb9   busybox   "sh"                    13 minutes ago   Exited (0) 12 minutes ago         eloquent_lamport
```

## 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

## Lab Worksheet

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 << EOFdocker

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**

```

nipatchapakdee@Macintosh-2 Lab8_2 % docker build -t first_docker .
[*] Building 0.0s (5/5) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 422B
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
=> [internal] load metadata for docker.io/library/busybox:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/1] FROM docker.io/library/busybox:latest
=> exporting to image
=> => exporting layers
=> => writing image sha256:c5df64fc8d22dd798d9e0482eb5664d853132c95386f339cab785923bbdc8f
=> => naming to docker.io/library/first_docker

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
nipatchapakdee@Macintosh-2 Lab8_2 % docker run first_docker
Nipat Chapakdee 653388332-7 Pee
  
```

- (1) คำสั่งที่ใช้ในการ run คือ

รัน container จาก image first\_docker

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

Option -t เมื่อใช้จะเป็นการตั้งชื่อให้ image และสามารถเรียกใช้และลบผ่านชื่อ image ได้เลย หากไม่ใช่ docker จะสร้าง image ขึ้นมาและเมื่อเราจะใช้งานจะต้องเรียกใช้ผ่าน image ID

## Lab Worksheet

## แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8\_3
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8\_3 เพื่อใช้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

```
FROM busybox
```

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

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal 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

## Lab Worksheet

```
nipatchapakdee@Macintosh-2 Lab8_3 % docker build -t peenipat/lab8 .
[+] Building 0.1s (9/5) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 172B
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
=> [internal] load metadata for docker.io/library/busybox:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> CACHED [1/1] FROM docker.io/library/busybox:latest
=> exporting image
=> => exporting layers
=> => writing image sha256:77315ae57f142df3994848d1ba783a6dd0872d943f5aaa0fe943cf1a45fd9ad0
=> => naming to docker.io/peenipat/lab8

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
nipatchapakdee@Macintosh-2 Lab8_3 % docker run peenipat/lab8
Nipat Chapakdee 653388932-7
```

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยใช้คำสั่ง

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

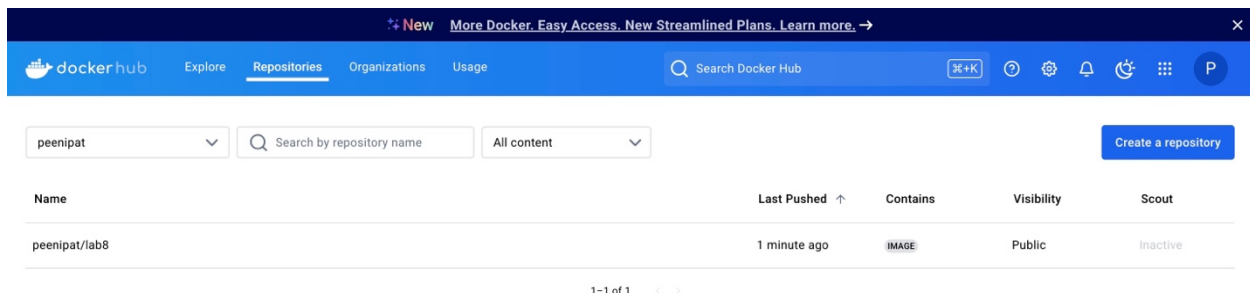
ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

```
$ 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)



Login Succeeded

```
nipatchapakdee@Macintosh-2 Lab8_3 %
```

```
nipatchapakdee@Macintosh-2 Lab8_3 % docker push peenipat/lab8
```

```
Using default tag: latest
```

```
The push refers to repository [docker.io/peenipat/lab8]
```

```
613e5fc506b9: Mounted from library/busybox
```

```
latest: digest: sha256:0d9ade47c1781aaa5ef10db5fccbdb4218b7f452ed6f3b6a3428175ba0cb5061 size: 527
```

```
nipatchapakdee@Macintosh-2 Lab8_3 %
```

## 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

```

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

```

A git repository was found in the parent folders of the workspace or the open file(s). Would you like to open the repository?

Source: Git

Yes Always Never

4.
 

```

nipatchapakdee@Macintosh-2 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 | 6.44 MiB/s, done.
Resolving deltas: 100% (523/523), done.
nipatchapakdee@Macintosh-2 Lab8_4 %
      
```



## Lab Worksheet

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

5. 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
```

6. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดใช้ชื่อ image เป็น myapp\_รหัสสนศ. ไม่มีขีด

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

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

```
nipatchapakdee@Macintosh-2 app % docker build -t myapp_6533803327 .
[+] Building 13.6s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 446B
=> [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
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:77e3b76b47148e28acc84f2e903f34bedc21385b3644c9967aa25b324bb0e6b4
=> [internal] load build context
=> => transferring context: 4.68MB
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => writing image sha256:17cd394aef48d78691c83f912ca864da261b152c2d173b895627977e54fcee23
=> => naming to docker.io/library/myapp_6533803327

What's next:
  View a summary of image vulnerabilities and recommendations → docker scout quickview
nipatchapakdee@Macintosh-2 app % docker run -dp 3000:3000 myapp_6533803327
f477abb83c4d71cee4abab78b8cfc3dade5b151eb678119e587d86cbb8aa6331
```

7. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

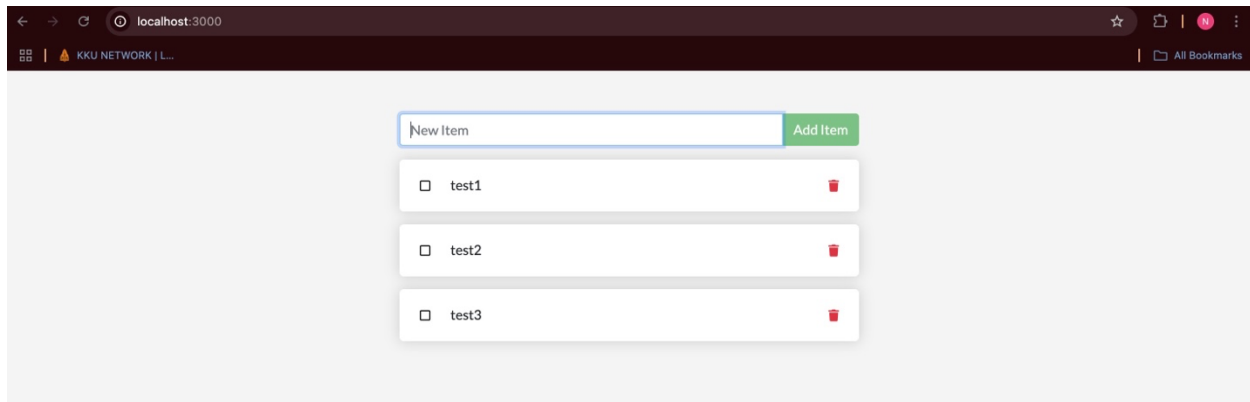
```
$ docker run -dp 3000:3000 <myapp_รหัสสนศ. ไม่มีขีด>
```

8. 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

### 9. 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

### 10. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

### 11. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

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

```

nipatchapakdee@Macintosh-2 app % docker run -dp 3000:3000 myapp_6533803327
f477ab0b3c4d71ce4abab70b0cf3dadedb151eb678119e587d86c0b08ae0351
nipatchapakdee@Macintosh-2 app % docker build -t myapp_6533803327 .
[+] Building 14.3s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 448B
=> [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
=> [1/4] FROM docker.io/library/node:18-alpine@sha256:77e3b76b47148e28acc84f2e903f34bedc21385b3644c9967aa25b324bb0e6b4
=> [internal] load build context
=> => transferring context: 10.06kB
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting image
=> writing image sha256:c85702bbad1ca23692915d6531bda7859f39709e85cb07b144c673d893d3dd22
=> naming to docker.io/library/myapp_6533803327

What's next:
  View a summary of image vulnerabilities and recommendations → docker scout quickview
nipatchapakdee@Macintosh-2 app % docker run -dp 3000:3000 myapp_6533803327
6254d0bb3b7f0e6cac8fbb9a49bcfe1c14654bb1351fd25b35a0f8a08ff4ca
docker: Error response from daemon: driver failed programming external connectivity on endpoint sharp_ptolemy (304288b9546a750cdbe943de66f94b108bf0e48c3432b5382fdf631ca69fa0): Bind for 0.0.0.0:3000 failed:
port is already allocated.
nipatchapakdee@Macintosh-2 app % docker run -dp 3000:3000 myapp_6533803327
e35cc47d3d7ab339aac078da08c87b211f1f5704f087c25ed334d01ecb08e095
docker: Error response from daemon: driver failed programming external connectivity on endpoint gracious_hasslett (3b635e7761770287f7a7d6ab3482450ce2573eb5573dec84db5e25c1eca347bc): Bind for 0.0.0.0:3000 failed:
d: port is already allocated.
nipatchapakdee@Macintosh-2 app % docker run -dp 3000:3000 myapp_6533803327
f1eecd468a46231ffffe56de3109e97d094335e7b3697b62e861f9c02b72da3

```

### (1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

ไม่สามารถเชื่อมกับ port 3000 ในเครื่องได้เพราะว่าในเครื่องมีการใช้งาน port 3000 อยู่

## Lab Worksheet

12. 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

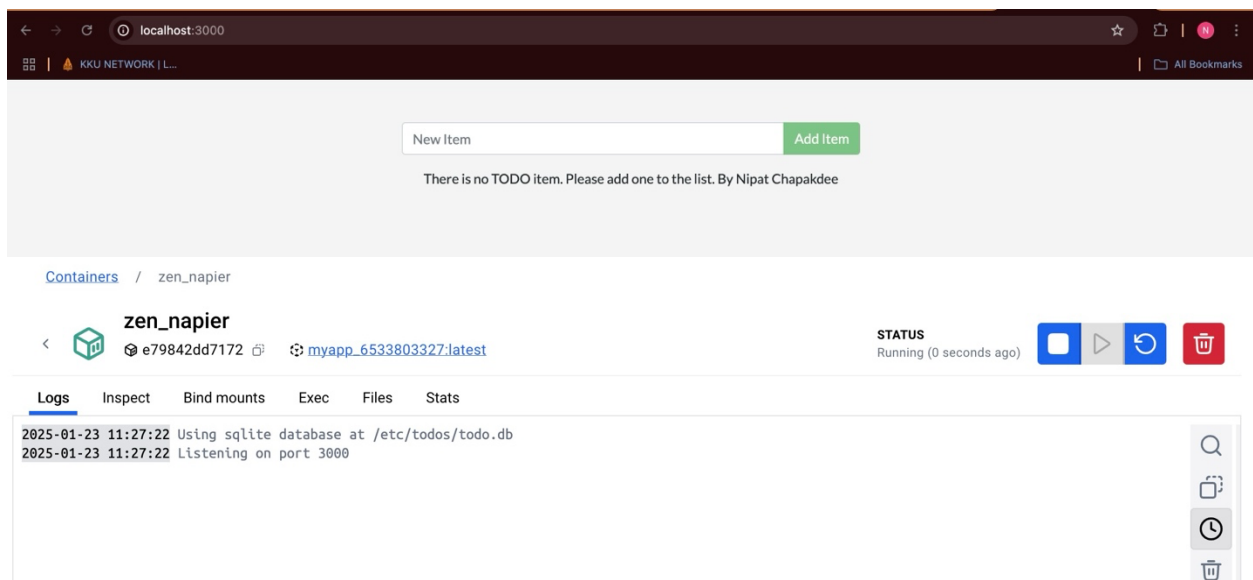
b. ผ่าน Docker desktop

- i. ไปที่หน้าต่าง Containers
- ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
- iii. ยืนยันโดยการกด Delete forever

13. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6

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

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



## Lab Worksheet

Containers [Give feedback](#)

Container CPU usage ⓘ

0.00% / 800% (8 CPUs available)

Container memory usage ⓘ

20.9MB / 3.74GB

[Show charts](#)

Q Search



Only show running containers

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	eloquent_lamport	23f4f497ceb9	busybox		0%	2 hours ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	recursing_kowalevski	037ad5e9edbc	first_docker		0%	1 hour ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	infallible_kowalevski	588f62ac7be9	first_docker		0%	1 hour ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	determined_haslett	87a2bbae8cde	peenipat/lab8		0%	37 minutes ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	blissful_bassi	89fafc35d35d	myapp_6533803327		0%	24 minutes ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	funny_ishizaka	4b63c344a8b9	myapp_6533803327	3000:3000	0%	23 minutes ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	nice_nobel	708ce41d0429	myapp_6533803327	3000:3000	0%	16 minutes ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	upbeat_euclid	f477abb83c4d	myapp_6533803327	3000:3000	0%	12 minutes ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	sharp_ptolemy	6254dbb0ba3b	myapp_6533803327	3000:3000	0%		<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	gracious_haslett	e35cc47e3d7a	myapp_6533803327	3000:3000	0%		<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>
<input type="checkbox"/>	recursing_mclean	f1eecfd468a4	myapp_6533803327	3000:3000	0%	5 minutes ago	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">🗑</a>

Showing 12 items

## 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

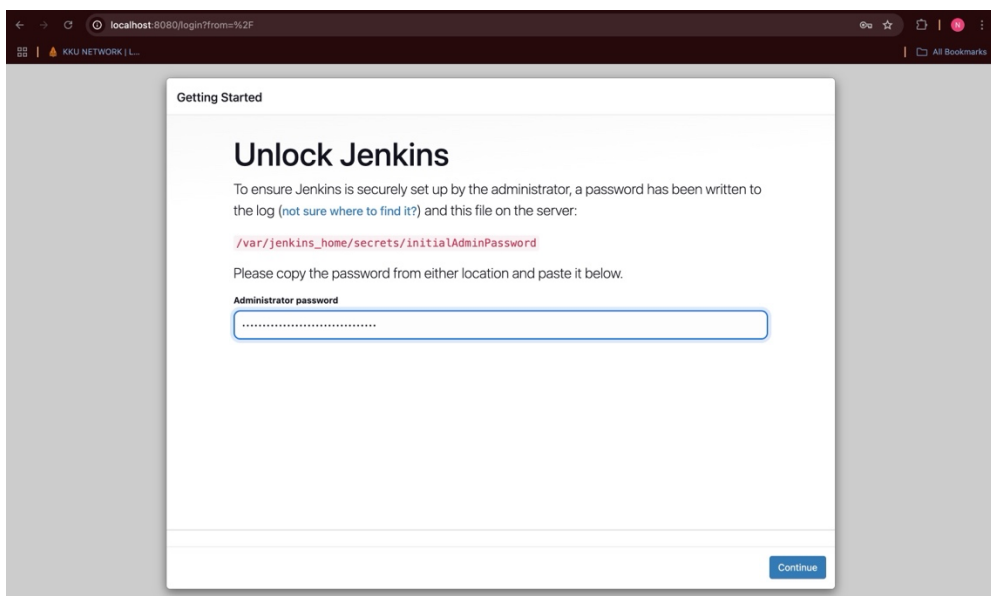
```
$ docker run -p 8080:8080 -p 50000:50000 --restart=on-failure -v
```

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

3. บันทึกการรหัสผ่านของ 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:

735f5d71130247f7949d23d9514c0025

This may also be found at: /var/jenkins\_home/secrets/initialAdminPassword

~~~~~

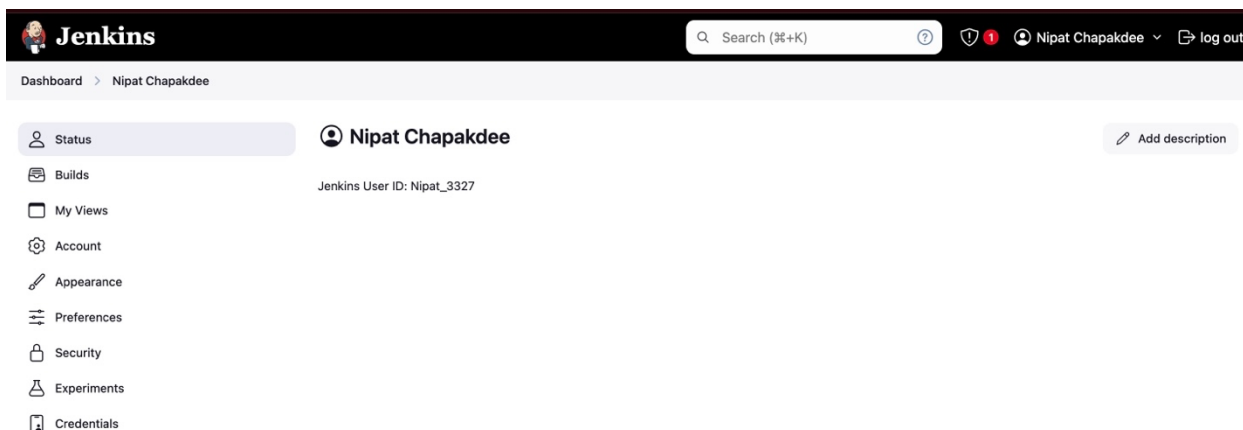
4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น

localhost:8080

5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3

6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri\_3062

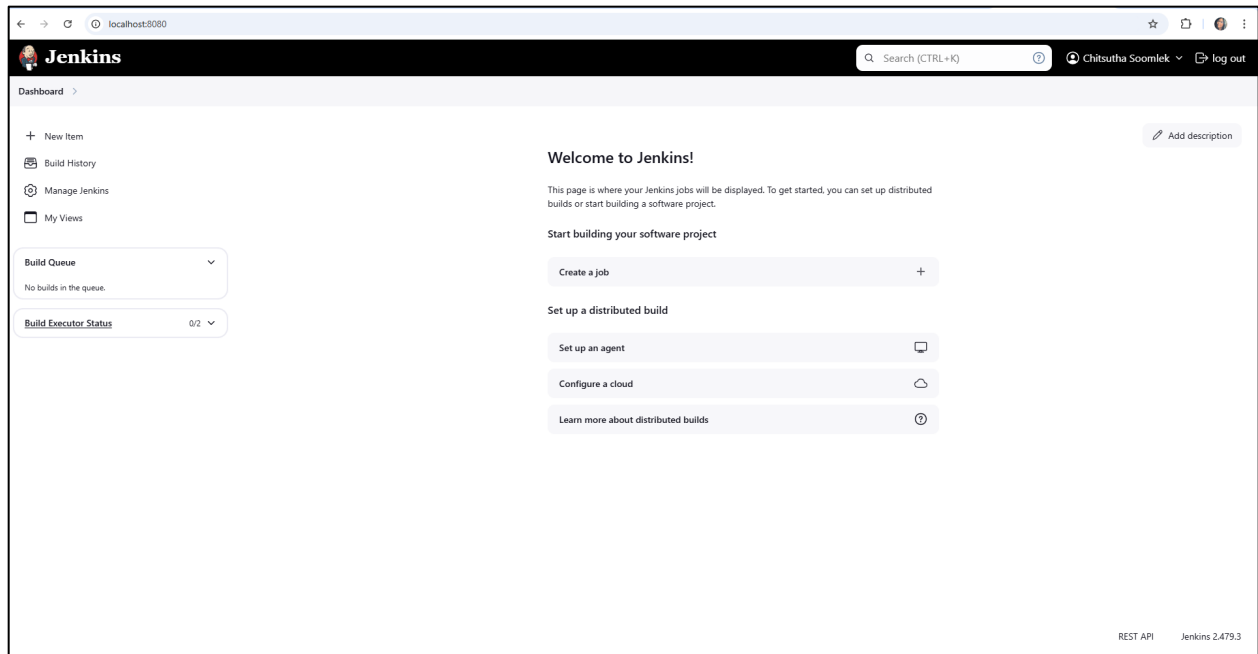
[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า



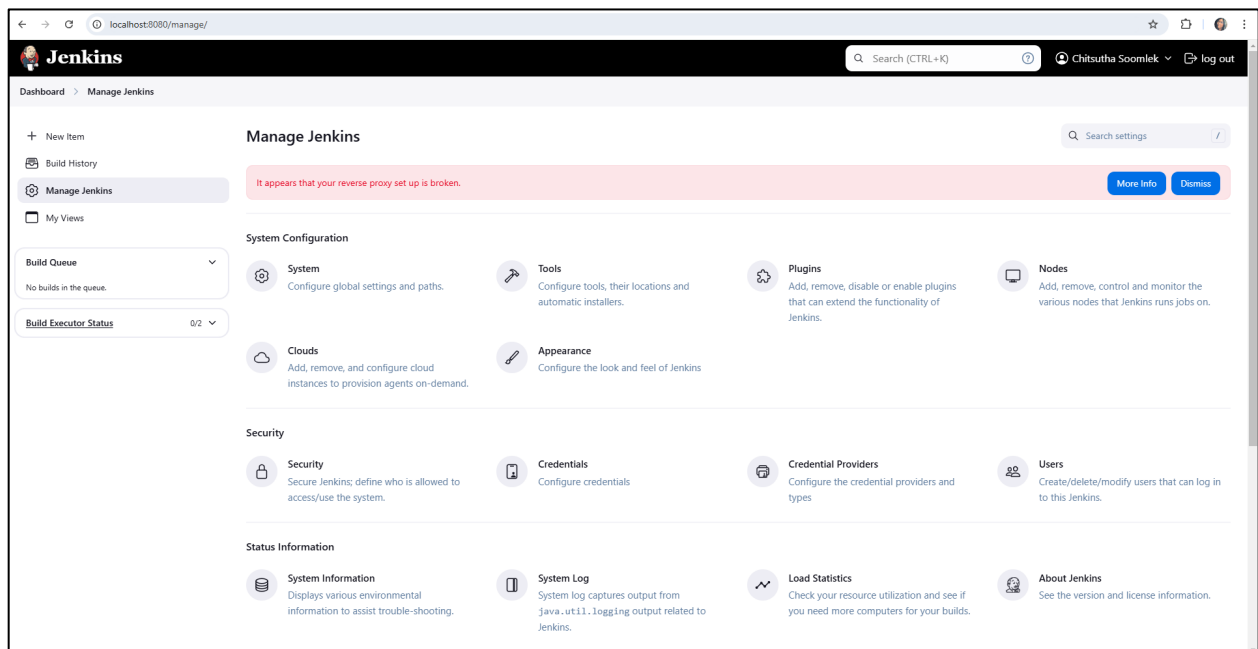
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>

## Lab Worksheet

## 8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ

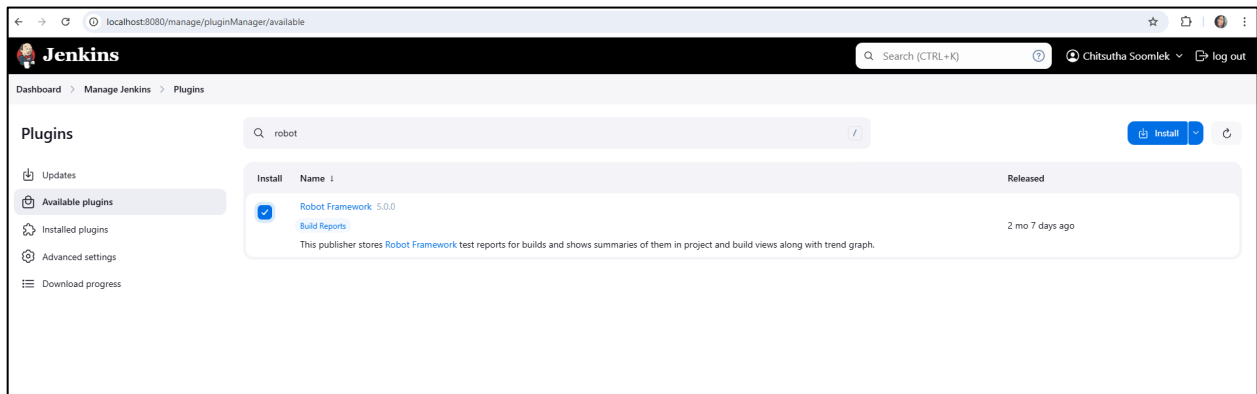


## 9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

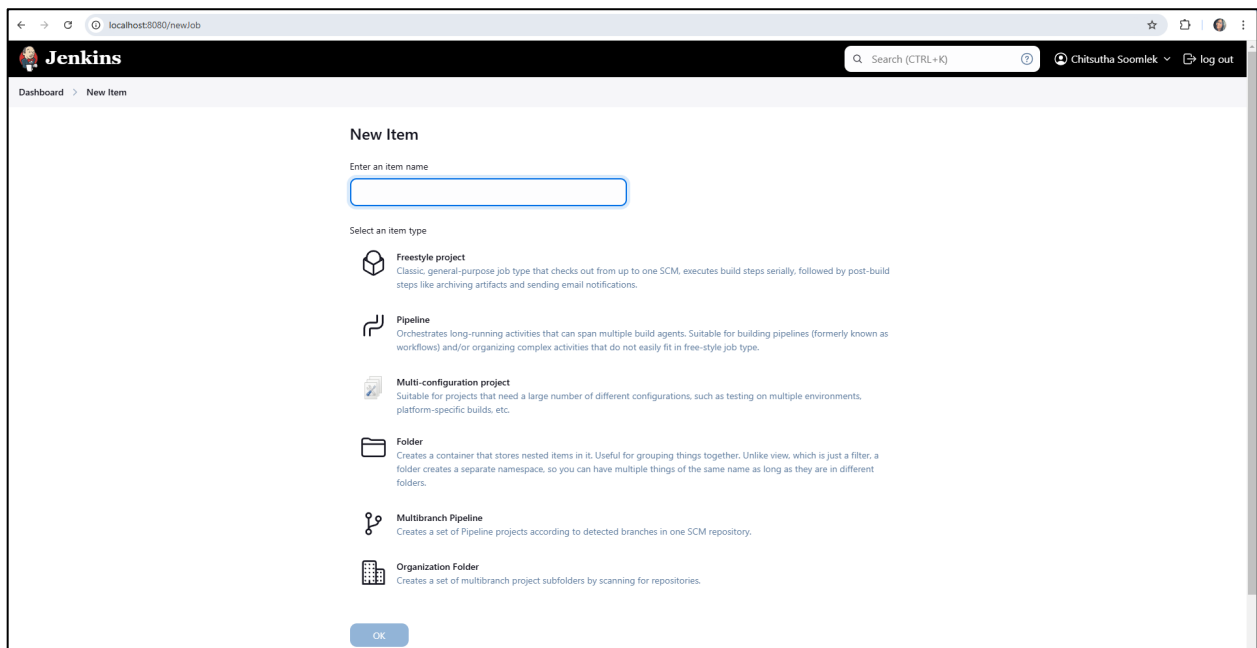


## Lab Worksheet

10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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



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

**Description:** Lab 8.5

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

**Build Trigger:** เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

## Lab Worksheet

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

**[Check point#14]** Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Dashboard > UAT > Configuration

**Configure**

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

**General** Enabled ☒

Description

Lab 8.5

Plain text [Preview](#)

☐ Discard old builds ?

☒ GitHub project

Project url ?

<https://github.com/nipat-c/UAT.git/>

Advanced ▾

☐ This project is parameterized ?

☐ Throttle builds ?

☐ Execute concurrent builds if necessary ?

Advanced ▾

Save Apply

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

```
./var/jenkins_home/robot_env/bin/activate
export PATH=$PATH:/usr/bin
mkdir -p results
robot --outputdir results test.robot
```

**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. สั่ง Build Now



## Lab Worksheet

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

Dashboard >

+ New Item

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Add description

| S | W | Name ↓ | Last Success     | Last Failure    | Last Duration | Robot Results + Duration Trend |
|---|---|--------|------------------|-----------------|---------------|--------------------------------|
| ✓ | ☀ | UAT    | 9 min 15 sec #74 | 1 hr 23 min #69 | 6.1 sec       | ▶                              |

Dashboard > UAT > #74 > Console Output

```

+ PATH=/var/jenkins_home/robot_env/bin:/opt/java/openjdk/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
+ export PATH
+ [ -n ]
+ [ -z ]
+ _OLD_VIRTUAL_PS1=$
+ PS1=(robot_env) $
+ export PS1
+ VIRTUAL_ENV_PROMPT=(robot_env)
+ export VIRTUAL_ENV_PROMPT
+ [ -n -o -n ]
+ export
PATH=/var/jenkins_home/robot_env/bin:/opt/java/openjdk/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
+ mkdir -p results
+ robot --outputdir results test.robot
=====
Test
=====
Open KKU Website | PASS |
=====
Test
1 test, 1 passed, 0 failed | PASS |
=====
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
Finished: SUCCESS

```

REST API Jenkins 2.479.3

✓ #76 (27 ม.ค. 2568 20:21:03)

Add description Keep this build forever

Started by user Nipat Chapakdee

Started 10 sec ago  
Took 5.2 sec

This run spent:

- 6 ms waiting;
- 5.2 sec build duration;
- 5.2 sec total from scheduled to completion.

Revision: 97b57f3d41faeffc5756e887d54b9046e2d845  
Repository: [https://nipat-c:ghp\\_onv9KbbqzDydziXXHC7kj5KUHX57R92PRkLb@github.com/nipat-c/UAT.git](https://nipat-c:ghp_onv9KbbqzDydziXXHC7kj5KUHX57R92PRkLb@github.com/nipat-c/UAT.git)

- refs/remotes/origin/main

Robot Test Summary:

|           | Total | Failed | Passed | Skipped | Pass % |
|-----------|-------|--------|--------|---------|--------|
| All tests | 1     | 0      | 1      | 0       | 100.0  |

- Browse results
- Open report.html
- Open log.html