

Lab Worksheet

ชื่อ-นามสกุล _____ น.ส.สุภัทสร สุทธิพัฒน์กุล _____ รหัสนักศึกษา _____ 653380156-1 _____ Section _____ 2 _____

Lab#8 – Software Deployment Using Docker

วัตถุประสงค์การเรียนรู้

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

Pre-requisite

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

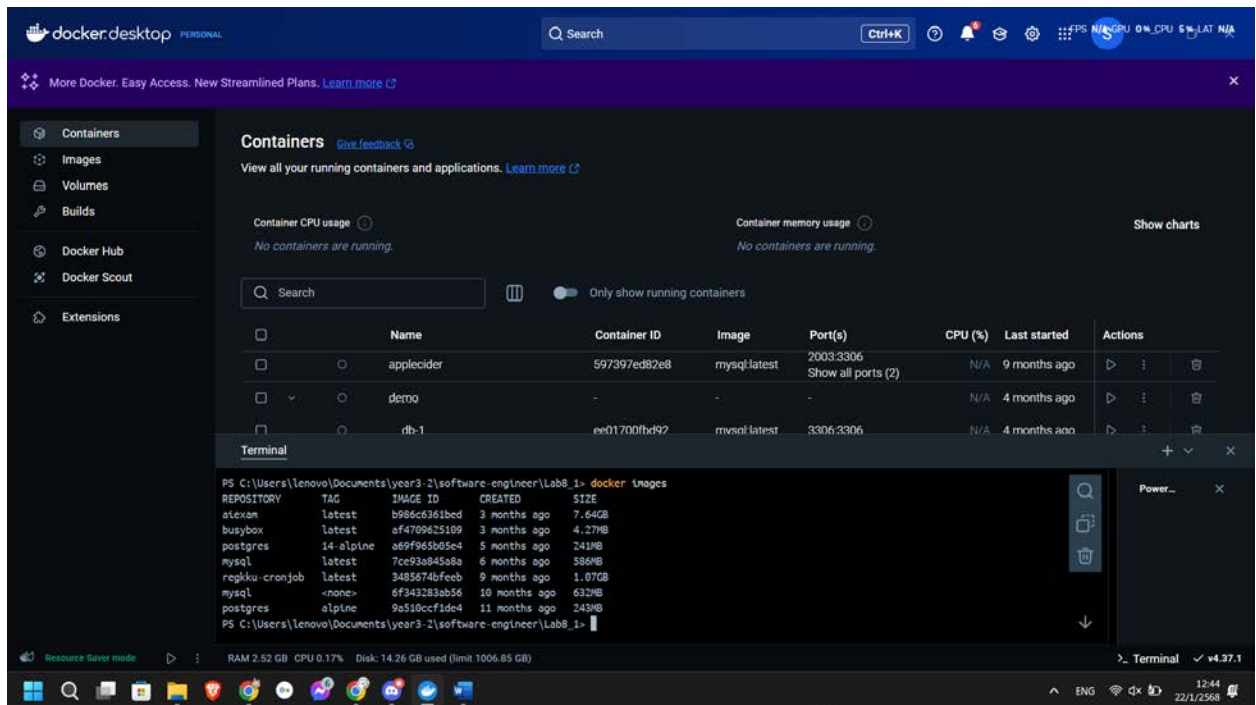
แบบฝึกปฏิบัติที่ 8.1 Hello world - รัน Container จาก Docker image

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
1. เปิด Command line หรือ Terminal บน Docker Desktop จากนั้นสร้าง Directory ชื่อ Lab8_1
2. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_1 เพื่อใช้เป็น Working directory
3. ป้อนคำสั่ง \$ docker pull busybox หรือ \$ sudo docker pull busybox สำหรับกรณีที่ติดปัญหา Permission denied
(หมายเหตุ: BusyBox เป็น software suite ที่รองรับคำสั่งบางอย่างบน Unix - <https://busybox.net>)
4. ป้อนคำสั่ง \$ docker images

Lab Worksheet

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

- (1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร ชื่อ image _____
- (2) Tag ที่ใช้บ่งบอกถึงอะไร _____ เป็น version อะไรอยู่ _____



(3)

5. ป้อนคำสั่ง \$ docker run busybox
6. ป้อนคำสั่ง \$ docker run -it busybox sh
7. ป้อนคำสั่ง ls

```

PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker run busybox
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker run -it busybox sh
/ # ls
bin    dev    etc    home  lib    lib64  proc  root  sys    tmp    usr    var
  
```

8. ป้อนคำสั่ง ls -la

```

/ # ls -la
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 295 root    root         0 Jan 22 05:45 proc
drwx----- 1 root    root       4096 Jan 22 05:45 root
dr-xr-xr-x 11 root    root         0 Jan 22 05:45 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
  
```

Lab Worksheet

9. ป้อนคำสั่ง exit

```
/ # exit
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1>
```

10. ป้อนคำสั่ง \$ docker run busybox echo "Hello ชื่อและนามสกุลของนักศึกษา from busybox"

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker run busybox echo "Hello Supatsorn 653380156-1 from busybox"
Hello Supatsorn 653380156-1 from busybox
```

11. ป้อนคำสั่ง \$ docker ps -a

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
62e409be5756	busybox	"echo 'Hello Supatso..."	10 seconds ago	Exited (0) 9 seconds ago		kind_davinci
b1cc4182951a	busybox	"sh"	About a minute ago	Exited (0) 49 seconds ago		compassionate_kapitsa
699ee7a0dac7	busybox	"sh"	About a minute ago	Exited (0) About a minute ago		ecstatic_booth
cb824a6e97b5	aiexam	"jupyter lab --ip=0..."	3 months ago	Exited (0) 3 months ago		busy_brahmagupta
ee01700fbd92	mysql:latest	"docker-entrypoint.s..."	4 months ago	Exited (0) 4 months ago		demo-db-1
088a1c6a7f44	postgres:alpine	"docker-entrypoint.s..."	4 months ago	Exited (0) 4 months ago		rentservice-postgres-1
2b595ef4a416	mysql:latest	"docker-entrypoint.s..."	6 months ago	Exited (0) 4 months ago		lab5-db-1
597397ed82e8	6f343283ab56	"docker-entrypoint.s..."	8 months ago	Exited (0) 8 months ago		applecider
c0762c239b5a	regkku-cronjob	"cron -f"	9 months ago	Exited (137) 9 months ago		regkku-cronjob-1

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

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

_____ i : interective , t: terminal คือเปิดใช้งาน terminal ที่สามารถใส่ input ได้ _____

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

_____ ในภาพดังกล่าว หมายถึง สถานะของ container นั้น เช่น กำลังใช้งาน (Up X time) ไม่ได้ใช้งาน (Exited (code) X time ago) _____

```
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
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker run busybox echo "Hello Supatsorn 653380156-1 from busybox"
Hello Supatsorn 653380156-1 from busybox
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
62e409be5756	busybox	"echo 'Hello Supatso..."	10 seconds ago	Exited (0) 9 seconds ago		kind_davinci
b1cc4182951a	busybox	"sh"	About a minute ago	Exited (0) 49 seconds ago		compassionate_kapitsa
699ee7a0dac7	busybox	"sh"	About a minute ago	Exited (0) About a minute ago		ecstatic_booth
cb824a6e97b5	aiexam	"jupyter lab --ip=0..."	3 months ago	Exited (0) 3 months ago		busy_brahmagupta
ee01700fbd92	mysql:latest	"docker-entrypoint.s..."	4 months ago	Exited (0) 4 months ago		demo-db-1
088a1c6a7f44	postgres:alpine	"docker-entrypoint.s..."	4 months ago	Exited (0) 4 months ago		rentservice-postgres-1
2b595ef4a416	mysql:latest	"docker-entrypoint.s..."	6 months ago	Exited (0) 4 months ago		lab5-db-1
597397ed82e8	6f343283ab56	"docker-entrypoint.s..."	8 months ago	Exited (0) 8 months ago		applecider
c0762c239b5a	regkku-cronjob	"cron -f"	9 months ago	Exited (137) 9 months ago		regkku-cronjob-1

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1>
```

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

Lab Worksheet

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

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker rm 699ee7a0dac7
699ee7a0dac7
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
a52f797c11f5	lab8_2	"/bin/sh -c 'echo \"S...\""	31 minutes ago	Exited (0) 31 minutes ago		vigilant_wescoff
ea6a6862692c	lab8_2	"/bin/sh -c 'echo \"S...\""	31 minutes ago	Exited (0) 31 minutes ago		gracious_goldstine
12b2183bad8c	lab8_2	"/bin/sh -c 'echo \"S...\""	32 minutes ago	Exited (0) 32 minutes ago		lucid_swirles
62e409be5756	busybox	"echo 'Hello Supatso...'"	43 minutes ago	Exited (0) 43 minutes ago		kind_davinci
b1cc4182951a	busybox	"sh"	44 minutes ago	Exited (0) 44 minutes ago		compassionate_kapitsa
cb824a6e97b5	aiexam	"jupyter lab --ip=0.0.0.0"	3 months ago	Exited (0) 3 months ago		busy_brahmagupta
ee01700fbd92	mysql:latest	"docker-entrypoint.s..."	4 months ago	Exited (0) 4 months ago		demo-db-1
088a1c6a7f44	postgres:alpine	"docker-entrypoint.s..."	4 months ago	Exited (0) 4 months ago		rentservice-postgres-1
2b595ef4a416	mysql:latest	"docker-entrypoint.s..."	6 months ago	Exited (0) 4 months ago		lab5-db-1
597397ed82e8	6f343283ab56	"docker-entrypoint.s..."	8 months ago	Exited (0) 8 months ago		appleclider
c0762c239b5a	regkku-cronjob	"cron -f"	9 months ago	Exited (137) 9 months ago		regkku-cronjob-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 และบันทึกคำสั่งต่อไปนี้

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

```
FROM busybox
```

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

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

```
EOF
```

หรือใช้คำสั่ง

Lab Worksheet

\$ touch Dockerfile

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

\$ docker build -t <ชื่อ Image> .

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_2> docker build -t lab8_2 .
[+] Building 0.4s (5/5) FINISHED
=> [internal] load build definition from Dockerfile                                docker:desktop-linux 0.1s
=> => transferring dockerfile: 159B                                              0.0s
=> [internal] load metadata for docker.io/library/busybox:latest                 0.0s
=> [internal] load .dockerignore                                                 0.0s
=> => transferring context: 2B                                                  0.0s
=> [1/1] FROM docker.io/library/busybox:latest                                  0.0s
=> exporting to image                                                            0.0s
=> exporting layers                                                              0.0s
=> writing image sha256:ccb09c3b7453637ad04cb2a2b4a11ab2572a231dc74f06a53db9c69db6c17ef7 0.0s
=> naming to docker.io/library/lab8_2                                           0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/vdyvltwhd3kuutf9a0wmdajsz

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
```

6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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

- (1) คำสั่งที่ใช้ในการ run คือ

_____ docker run lab8_2 _____

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

_____ เรียกใช้ pseudo terminal เช่น bash หรือ sh _____

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

Lab Worksheet

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
```

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_3> docker build -t supatz3107/lab8 .
[+] Building 0.3s (5/5) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 178B
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
=> [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 to image
=> => exporting layers
=> => writing image sha256:0600fe756a1c6344ab23f2070493a53ad029abe4414f886e0485684ea9e33ecd
=> => naming to docker.io/supatz3107/lab8

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/saz49gdr-im3q3u1umkj24k39

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

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

5. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง

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

Lab Worksheet

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_3> docker run supatz3107/lab8
Hi there. My work is done. You can run them from my Docker image.
Supatsorn Sutthiphatkul 653380156-1
```

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

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

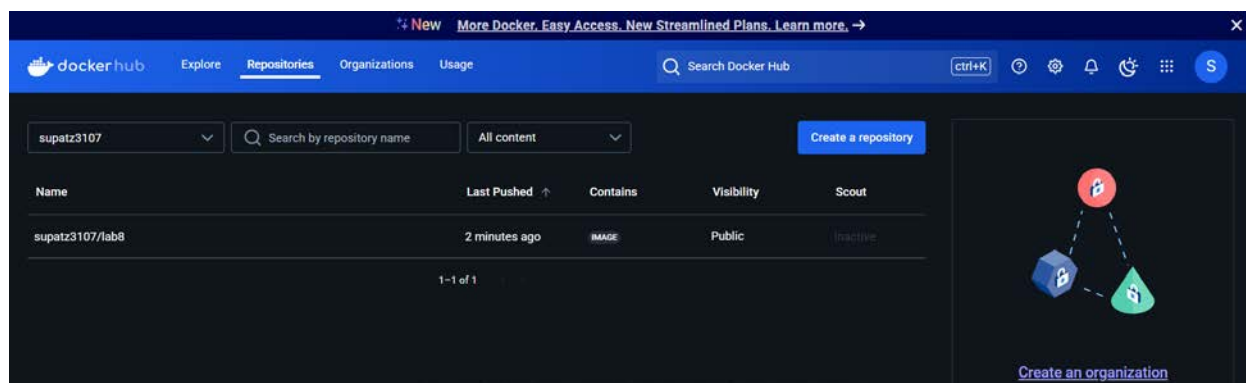
```
$ docker login แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง
```

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

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_3> docker push supatz3107/lab8
Using default tag: latest
The push refers to repository [docker.io/supatz3107/lab8]
59654b79daad: Mounted from library/busybox
latest: digest: sha256:4c3cf7c680b4d0a16db8f114b96d76c396e5b9ab3781a569df35f077701413a4 size: 527
```

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

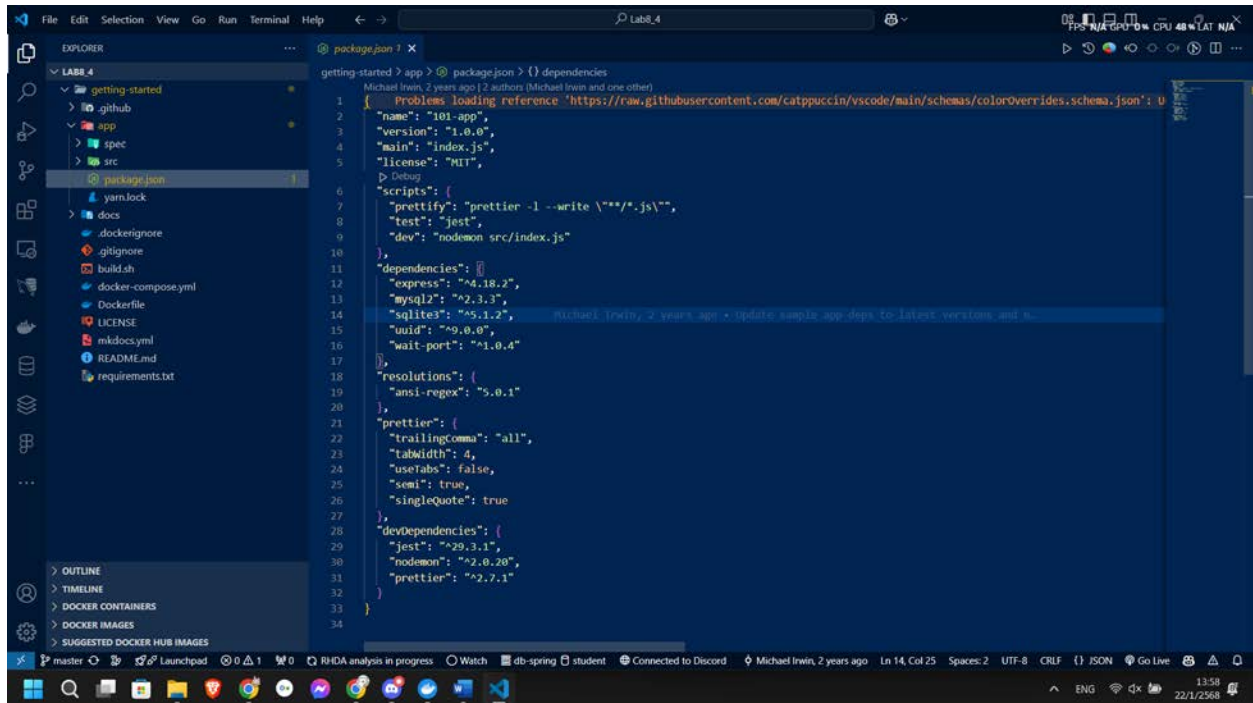
1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4
2. ทำการ Clone ซอร์สโค้ดของเว็บแอปพลิเคชันจาก GitHub repository
<https://github.com/docker/getting-started.git> ลงใน Directory ที่สร้างขึ้น โดยใช้คำสั่ง

```
$ git clone https://github.com/docker/getting-started.git
```


Lab Worksheet

- 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



- 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
- ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดใช้ชื่อ image เป็น myapp_รหัสสนศ. ไม่มีขีด
 \$ docker build -t <myapp_รหัสสนศ. ไม่มีขีด> .

Lab Worksheet

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ

```

getting-started > app > Dockerfile > ...
1 FROM node:18-alpine
2 WORKDIR /app
3 COPY . .

ERROR: "docker build build" requires exactly 1 argument.
See "docker build build --help".

Usage: docker build build [OPTIONS] PATH | URL | -

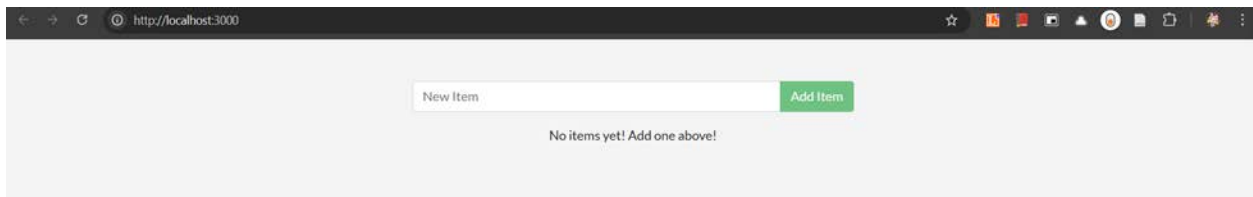
Start a build
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab6_4\getting-started\app> docker build -t myapp.6538801561 .
[+] building 41.3s (10/10) FINISHED
-> [internal] load build definition from Dockerfile
-> => transferring dockerfile: 156B
-> [internal] load metadata for docker.io/node:18-alpine
-> [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:a24108da7089c2d293cea61fb8969ec10821e0efc2572e5abb10b1841eb70b
-> resolve docker.io/library/node:18-alpine@sha256:a24108da7089c2d293cea61fb8969ec10821e0efc2572e5abb10b1841eb70b
-> sha256:4e8e5b9bf1c9496e6ef0f37cd8eb5ab9e0cdd0f5b0d2894931f5617adda3a5 40.00kB / 40.00kB
-> sha256:56147f690f6e6b9f54d1d1d27cad26d9d5be909049c5f705dcb9684488f2d27 1.26kB / 1.26kB
-> sha256:a24108da7089c2d293cea61fb8969ec10821e0efc2572e5abb10b1841eb70b 7.67kB / 7.67kB
-> sha256:ce927bbeecbd91a41d3bb64014bf44796eb06a35a4798f0ebbf3725024c6 1.72kB / 1.72kB
-> sha256:87aba7369c7592b4ab4d132ba570e2b56c11c0fa289ff02594c91e51c98ad96d 6.18kB / 6.18kB
-> sha256:1f3e46996e2966e4fa5846e56e70e3748b7315e2ded61476c24403d592134f0 3.64kB / 3.64kB
-> extracting sha256:1f3e46996e2966e4fa5846e56e70e3748b7315e2ded61476c24403d592134f0
-> sha256:be1495f7193c512f5cb09ecbd736ff146c06df9e0db4691eaeaa887c8dbf0b 445B / 445B
-> extracting sha256:4e8e5b9bf1c9496e6ef0f37cd8eb5ab9e0cdd0f5b0d2894931f5617adda3a5
-> extracting sha256:56147f690f6e6b9f54d1d1d27cad26d9d5be909049c5f705dcb9684488f2d27
-> extracting sha256:be1495f7193c512f5cb09ecbd736ff146c06df9e0db4691eaeaa887c8dbf0b
-> [internal] load build context
-> => transferring context: 4.82kB
-> [2/4] WORKDIR /app
-> [3/4] COPY . .
-> [4/4] RUN yarn install --production
-> exporting to image

```

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

\$ docker run -dp 3000:3000 <myapp_รหัสศ. ไม่มีชื่อ>

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



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

Lab Worksheet

หมายเหตุ: นศ.สามารถทดลองเล่น Web application ที่ทำงานอยู่ได้

8. ทำการแก้ไข Source code ของ Web application ดังนี้

a. เปิดไฟล์ src/static/js/app.js ด้วย Editor และแก้ไขบรรทัดที่ 56 จาก

<p className="text-center">No items yet! Add one above!</p> เป็น

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

```

Dockerfile
14 function TodoListCard() {
15   // ...
16 }
17
18 if (items === null) return 'Loading...';
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

```

View a summary of image vulnerabilities and recommendations + docker scout quickview
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801561
bfa5428462962e38f6ac3b0b65ffabab5c4b52e8d689466b123c967c6274b8d
docker: Error response from daemon: driver failed programming external connectivity on endpoint competent_banza1 (b7f313e22ee9346b9e9f693fb48cbcf113ff4015f7cbba
5126d978345e4b887b): Bind for 0.0.0.0:3000 failed: port is already allocated.
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_4\getting-started\app>

```

Lab Worksheet

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

_____port 3000 กำลังใช้งานอยู่ โดยงานอื่น_____

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

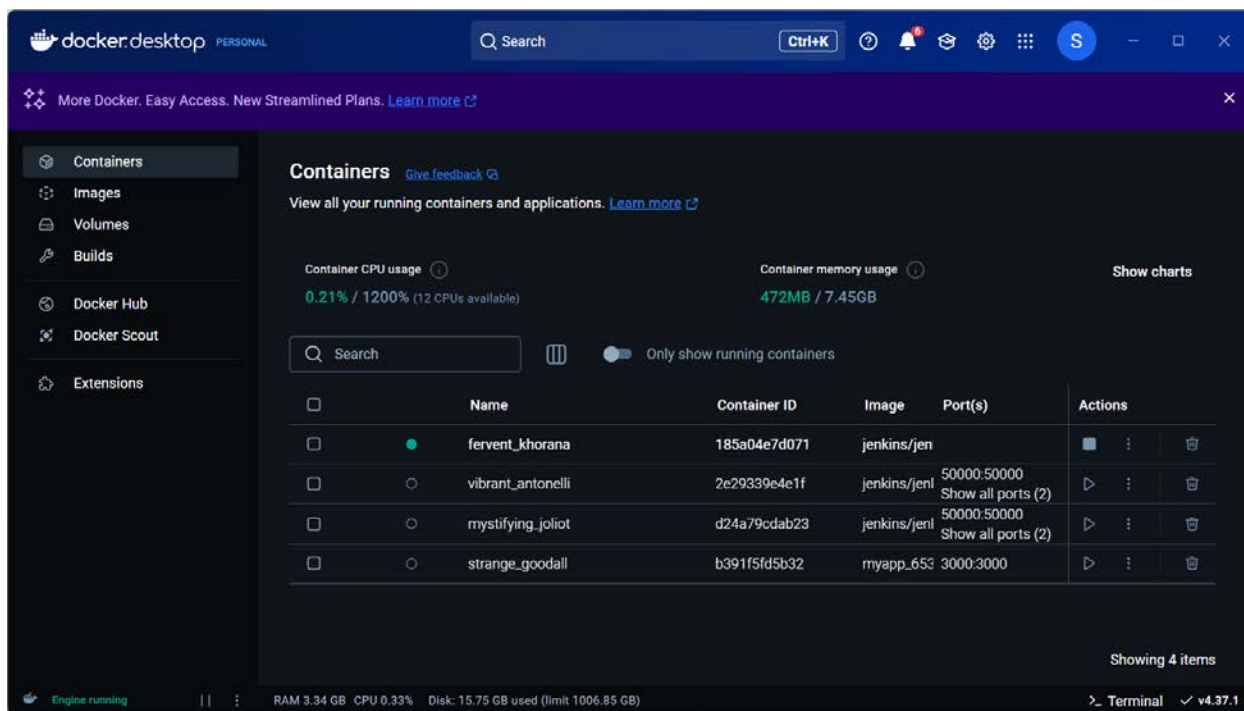
- i. ไปที่หน้าต่าง Containers
- ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
- iii. ยืนยันโดยการกด Delete forever

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

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

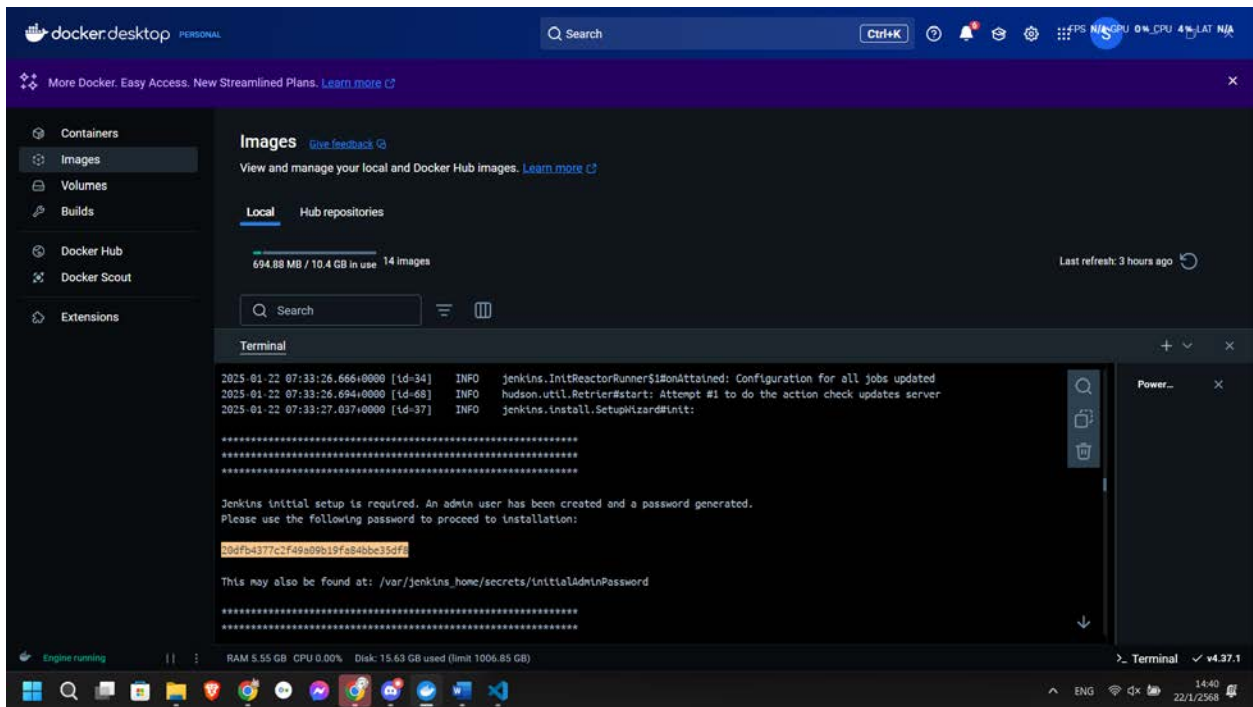
```
$ docker run -p 8080:8080 -p 50000:50000 --restart=on-failure -v
```

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

3. บันทึกรหัสผ่านของ Admin user ไว้สำหรับ log-in ในครั้งแรก

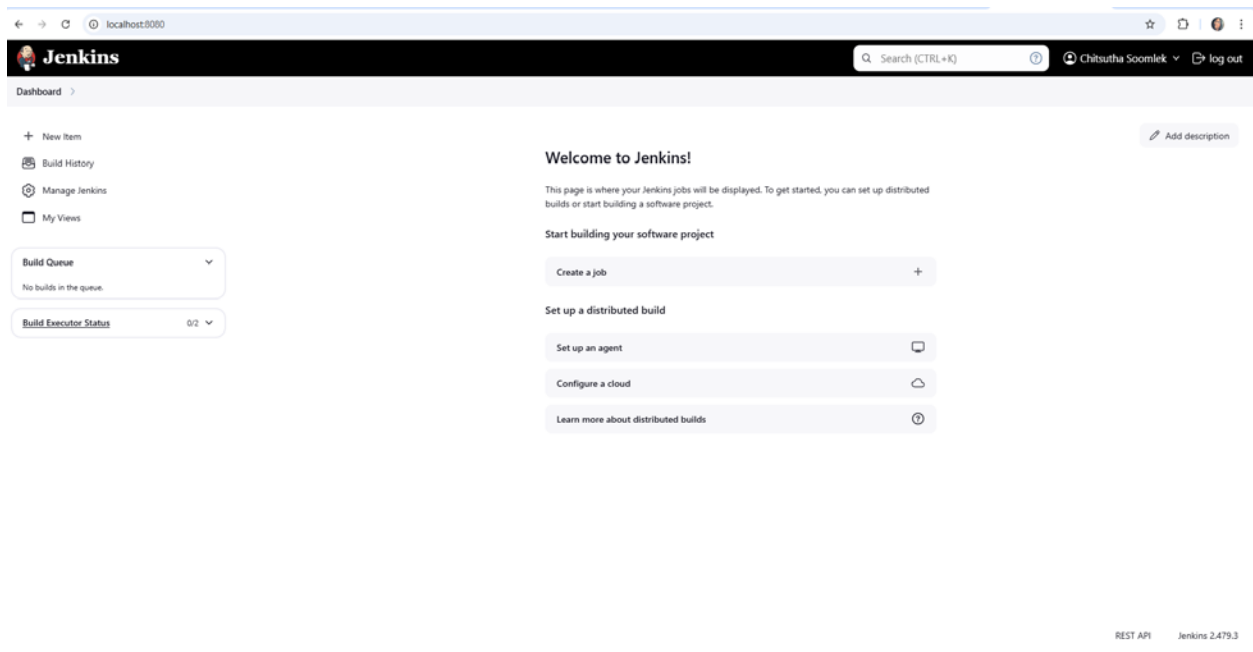
[Check point#12] Capture หน้าจอที่แสดงผล Admin password

Lab Worksheet

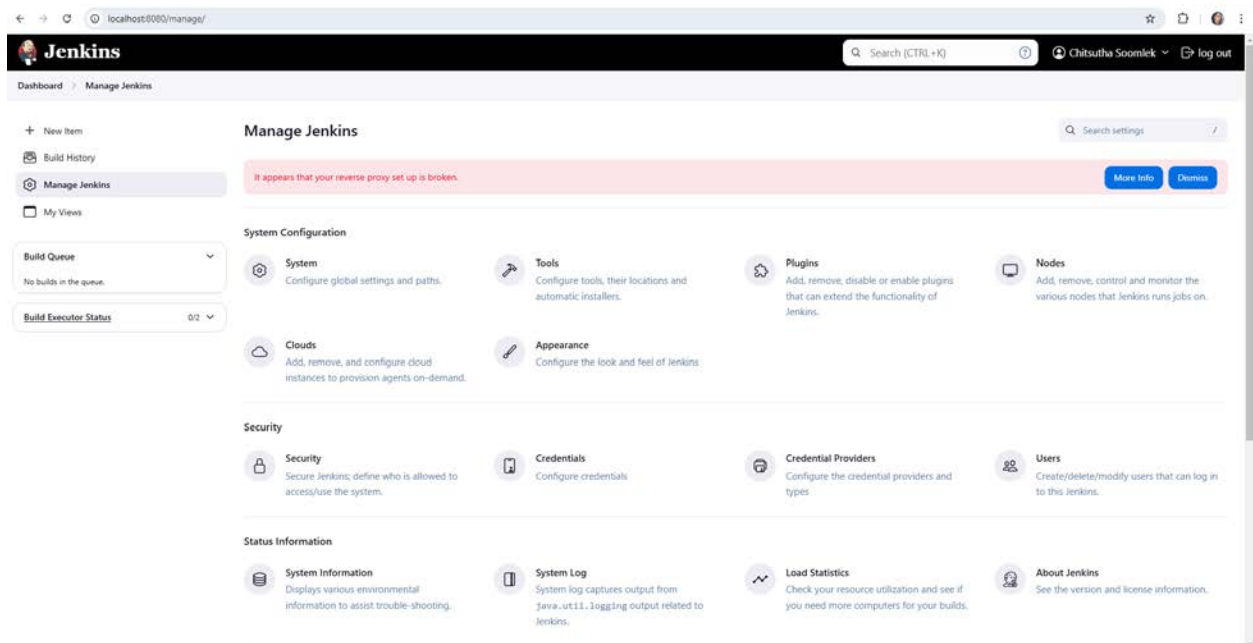


4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น localhost:8080
5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3
6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062
- [Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>
8. เมื่อติดตั้งเรียบร้อยแล้วจะพบหน้าจอ Dashboard ดังแสดงในภาพ

Lab Worksheet



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

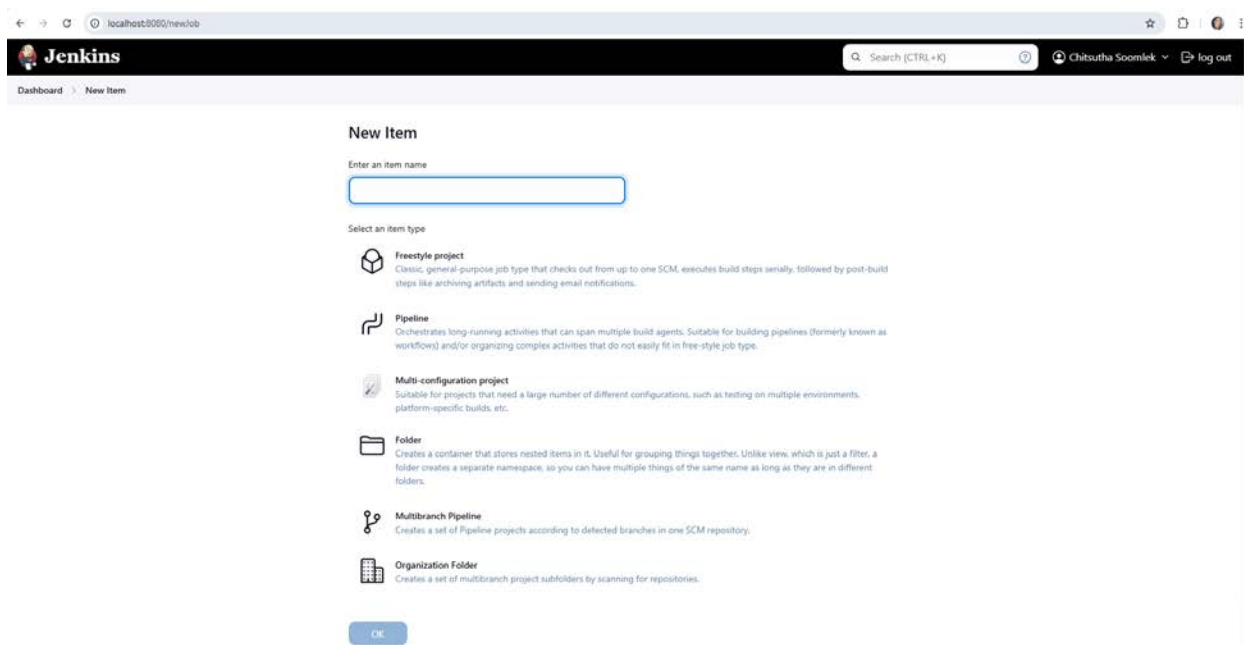


Lab Worksheet

10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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



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

Description: Lab 8.5

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

Build Trigger: เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

Lab Worksheet

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

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☐ Build after other projects are built ?

☐ Build periodically ?

☐ 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

☐ With Ant ?

Build Steps

Execute shell ?

Command

See the list of available environment variables

```

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

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

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

sleep 10

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

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

```

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)

results

Advanced ▾ Edited

Thresholds for build result ?

👍%

20.0

👎%

80.0

☐ 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

Lab Worksheet

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

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

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

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

sleep 10

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

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

Lab Worksheet

Post-build action: เพิ่ม Publish Robot Framework test results ->

ระบุไดเรกทอรีที่เก็บไฟล์ผลการทดสอบโดย Robot framework ในรูป xml และ html -> ตั้งค่า Threshold เป็น % 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

[illegible]