

Lab Worksheet

ชื่อ-นามสกุล น.ส.พทษรีวิภา มีระจุ Jinan

รหัสนักศึกษา 653380275-3 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. ผู้เรียนสามารถนำซอฟต์แวร์ที่พัฒนาขึ้นให้สามารถรันบนสภาพแวดล้อมเดียวกันและทำงานร่วมกันกับสมาชิกในทีมพัฒนาซอฟต์แวร์ฝ่าย 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 เครื่อง
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

[Check point#1] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

```

Terminal
miw@PathM Documents % cd Soft\ En
miw@PathM Soft En % mkdir Lab8_1
miw@PathM Soft En % cd Lab8_1
miw@PathM 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 quickview busybox
miw@PathM Lab8_1 % docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
mynode-express--frontend    latest   bc3a4d6217db  13 days ago   762MB
mynode-express--backend    latest   bccdd83486ef  13 days ago   277MB
nginx                alpine   984fda50be27  8 weeks ago   50.9MB
supernode-mongo6         latest   b4590432594f3  2 months ago  683MB
mynode-express--mongo6    latest   8ae292155194  2 months ago  683MB
prisma-seminar-mongo6    latest   2bd9d76983c5b  2 months ago  683MB
prisma-dataserial-mongo6 latest   682589d19378  2 months ago  683MB
prisma-login-mongo6     latest   b0bc5f53b2a8  2 months ago  683MB
mysql                latest   22aaacaaafc0e  3 months ago  624MB
busybox               latest   fc0179a204e2  3 months ago  4.04MB
postgres              latest   6c9aaa6ecd71d  3 months ago  456MB
dpage/pgadmin4          latest   69eb871cd151  4 months ago  497MB
phpmyadmin/phpmyadmin    latest   933569f3a9f6  18 months ago  562MB
miw@PathM Lab8_1 %

```

RAM 0.48 GB CPU 0.38% Disk: 7.70 GB used (limit 58.37 GB)

(1) ลิสต์ที่อยู่ภายในได้คอลัมน์ Repository คืออะไร

ตอบ ชื่อของแอปพลิเคชัน/โปรแกรมที่ image นั้นถูกสร้างขึ้นเพื่อใช้งาน เช่น busybox, nginx, mysql

(2) Tag ที่ใช้บ่งบอกถึงอะไร

ตอบ เวอร์ชันหรือสถานะของ Docker image ใน Repository เพื่อให้ผู้ใช้งานทราบว่าเป็น image รุ่นไหน

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) และแสดงผลลัพธ์ที่ได้ตั้งแต่ขั้นตอนที่ 6-12 พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

```

miw0PathM Lab8_1 % docker run busybox
miw0PathM 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 22 09:18 .
drwxr-xr-x 1 root root 4096 Jan 22 09:18 ..
-rwxr-xr-x 1 root root 0 Jan 22 09:18 .dockerenv
drwxr-xr-x 2 root root 12288 Sep 26 21:31 bin
drwxr-xr-x 5 root root 368 Jan 22 09:18 dev
drwxr-xr-x 1 root root 4096 Jan 22 09:18 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 221 root root 0 Jan 22 09:18 proc
drw-r----- 1 root root 4096 Jan 22 09:18 root
dr-xr-xr-x 11 root root 0 Jan 22 09:18 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
miw0PathM Lab8_1 % docker run busybox echo "Hello พี่น้อง ลีรุ่งเรือง 653380275-3 from busybox"
"Hello พี่น้อง ลีรุ่งเรือง 653380275-3 from busybox"
miw0PathM Lab8_1 % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
e21232cd0259 busybox "echo \"Hello พี่น้อง...\" 6 seconds ago Exited (0) 6 seconds ago eloquent_williamson
0f9572d25d46 busybox "sh" About a minute ago Exited (0) 50 seconds ago clever_gauss
023f8db0810e busybox "sh" About a minute ago Exited (0) About a minute ago stoic_mclean
dffc249b1d6c prisma-dataserial-mongo6 "/bin/sh -c 'mongod ...'" 13 days ago Exited (137) 13 days ago prisma-dataserial-mongo6
-1
da000a89ac3a prisma-seminar-mongo6 "/bin/sh -c 'mongod ...'" 2 months ago Exited (137) 13 days ago prisma-seminar-mongo6-1
c61bfff8583d2 dpage/pgadmin4 "/entrypoint.sh" 2 months ago Exited (0) 13 days ago my-nextauth-app-pgadmin-
1
ff805e388569 postgres:latest "docker-entrypoint.s..." 2 months ago Exited (0) 13 days ago my-nextauth-app-postgres
-1
a9234264666c prisma-login-mongo6 "/bin/sh -c 'mongod ...'" 2 months ago Exited (137) 6 weeks ago prisma-login-mongo6-1
b483d7cb6a37 phpmyadmin/phpmyadmin:latest "/docker-entrypoint.s..." 2 months ago Exited (0) 2 months ago phpmyadmin
c547593e517a mysql:latest "docker-entrypoint.s..." 2 months ago Exited (0) 2 months ago mysql_db
miw0PathM Lab8_1 %

```

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสั้นๆ เช่น

ตอบ

- -i ทำให้ Container รับ Input จากผู้ใช้ผ่าน Terminal ได้แบบโต้ตอบ
- -t สร้าง Terminal เสมือน เพื่อให้สามารถใช้งาน Shell หรือ Command Line ภายใน Container ได้อย่างสะดวก
- ใช้ -it ให้ผู้ใช้สามารถเข้าสู่ Shell ของ Container เพื่อทำงานแบบโต้ตอบ เช่น การเรียกใช้คำสั่งใน Container โดยตรง

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

ตอบ

- Exited (รหัส) - Container หยุดการทำงานแล้ว พิริ่มแสดงรหัส Exit Code ที่ระบุสถานะการจบการทำงาน
- Up (ระยะเวลา) - Container กำลังทำงานอยู่ และแสดงระยะเวลาที่ Container ทำงาน
- Created Container - ถูกสร้างขึ้นแต่ยังไม่เริ่มทำงาน

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

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

Lab Worksheet

```

miw0PathM Lab8_1 % docker run busybox
miw0PathM 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 22 09:18 .
drwxr-xr-x  1 root  root  4096 Jan 22 09:18 ..
-rwxr-xr-x  1 root  root  0 Jan 22 09:18 .dockercfg
drwxr-xr-x  2 root  root  12288 Sep 26 21:31 bin
drwxr-xr-x  5 root  root  360 Jan 22 09:18 dev
drwxr-xr-x  1 root  root  4096 Jan 22 09:18 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
drwxr-xr-x  221 root  root  0 Jan 22 09:18 proc
drwx-----  1 root  root  4096 Jan 22 09:18 root
dr-xr-xr-x  11 root  root  0 Jan 22 09:18 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
miw0PathM Lab8_1 % docker run busybox echo "Hello พี่แม่รีรา ยิ่งดูรุ่นใหญ่ 653380275-3 from busybox"
"Hello พี่แม่รีรา ยิ่งดูรุ่นใหญ่ 653380275-3 from busybox"
miw0PathM Lab8_1 % docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
e21232cd0259        busybox             "echo \"Hello พี่แม่รีรา...\""
6 seconds ago       Exited (0) 6 seconds ago
0f9572d25d46        busybox             "sh"
About a minute ago Exited (0) 58 seconds ago
e23f8db0810e        busybox             "sh"
About a minute ago Exited (0) About a minute ago
dffc249b1d6c        prisma-dataserial-mongo6   "/bin/sh -c 'mongod ..."
13 days ago         Exited (137) 13 days ago
-1
da00089ac3a        prisma-seminar-mongo6   "/bin/sh -c 'mongod ..."
2 months ago        Exited (137) 13 days ago
c61bf8583d2        dpage/pgadmin4      "/entrypoint.sh"
2 months ago        Exited (0) 13 days ago
1
ff8805e3885a9       postgres:latest      "docker-entrypoint.s..."
2 months ago        Exited (0) 13 days ago
-1
a9234264666c        prisma-login-mongo6   "/bin/sh -c 'mongod ..."
2 months ago        Exited (137) 6 weeks ago
b483d7cb6a37        phpmyadmin/phpmyadmin:latest "/docker-entrypoint..." 2 months ago        Exited (0) 2 months ago
c547f593e517a        mysql:latest       "docker-entrypoint.s..."
2 months ago        Exited (0) 2 months ago
miw0PathM Lab8_1 % docker rm e21232cd0259
e21232cd0259
miw0PathM Lab8_1 %

```

แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

- เปิดไฟล์ Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เครื่องเรา
- เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_2
- ย้ายตำแหน่งปัจจุบันไปที่ Lab8_2 เพื่อให้เป็น Working directory
- สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดว์ (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

Lab Worksheet

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 พิจารณาดูว่าได้ดำเนินการตามที่ต้องการหรือไม่

```
miw0PathM Lab8_1 % cd ..
miw0PathM Soft En % mkdir Lab8_2
miw0PathM Soft En % cd Lab8_2
miw0PathM Lab8_2 % cat > Dockerfile << EOF
FROM busybox
CMD echo "Hi there. This is my first docker image."
CMD echo "ทดสอบคำสั่ง สำหรับ Docker image"
[EOF]

miw0PathM Lab8_2 % docker build -t Lab8_2_docker
ERROR: "docker buildx build" requires exactly 1 argument.
See 'docker buildx build --help'.

Usage: docker buildx build [OPTIONS] PATH | URL | -
      Start a build
miw0PathM Lab8_2 % docker build -t Lab8_2_docker .
[+] Building 0.0s (0/0)
ERROR: invalid tag "Lab8_2_docker": repository name must be lowercase
miw0PathM Lab8_2 % docker build -t lab8_2_docker .
[+] Building 0.1s (5/5) FINISHED
=> [internal] load build definition from Dockerfile
=> [internal] transfering dockerfile: 213B
=> [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 .dockerrcignore
=> [internal] transfering context: 28
=> [1/1] FROM docker.io/library/busybox:latest
=> exporting to image
=> exporting layers
=> => writing image sha256:277542b6f16f4da488b63b375425bcc7c01b5aa4ee4d21c055ad074785aca604
=> => naming to docker.io/library/lab8_2_docker
[0.0s]
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/a826z605kqd0pwvlpaoqudnq

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
miw0PathM Lab8_2 % docker run lab8_2_docker
"ทดสอบคำสั่ง สำหรับ Docker image"
miw0PathM Lab8_2 %
```

(1) คำสั่งที่ใช้ในการ run คือ

ต่อไป docker run lab8_2_docker

- (2) Option -t ในคำสั่ง \$ docker build สร้างผลลัพธ์การทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสั้นๆ
ต่อไป Option -t ในคำสั่ง docker build ใช้สำหรับตั้งชื่อ tag ให้กับ Docker image ที่สร้างขึ้น เพื่อให้ง่ายต่อการอ้างอิงและใช้งานภายหลัง

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

1. เปิดไฟล์งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เครื่อง
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_3

Lab Worksheet

3. ขยับตำแหน่งป้าจุบันไปที่ Lab8_3 เพื่อให้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการwinдовส์ 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

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

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

Lab Worksheet

```

miw@PathM_Lab8_3 % cat > Dockerfile << EOF
FROM busybox
CMD echo "Hi there. My work is done. You can run them from my Docker image."
CMD echo "ພົນເຮົາ ສະຖານທີ່ 653380275-3"
EOF

miw@PathM_Lab8_3 % docker build -t pattheerateerarujinont/lab8
ERROR: "docker buildx build" requires exactly 1 argument.
See 'docker buildx build --help'.

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

Start a build
miw@PathM_Lab8_3 % docker build -t pattheerateerarujinont/lab8 .
[+] Building 0.1s (5/5) FINISHED
--> [internal] load build definition from Dockerfile
--> => transferring dockerfile: 225B
--> 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 .dockignore
--> => transferring context: 2B
--> CACHED [1/1] FROM docker.io/library/busybox:latest
--> exporting to image
--> => exporting layers
--> => writing image sha256:91a994f6cbef5a90eb7501ba0a1a7b97e392615504e1bf205af5bfaf925cd70f
--> => naming to docker.io/pattheerateerarujinont/lab8

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/gxdh61sz64gmfforztbjzgj

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
miw@PathM_Lab8_3 % docker run pattheerateerarujinont/lab8
"ພົນເຮົາ ສະຖານທີ່ 653380275-3"
miw@PathM_Lab8_3 %

```

6. ทำการ Push ตัว Docker image ไปยังบน Docker Hub โดยการใช้คำสั่ง

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

ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

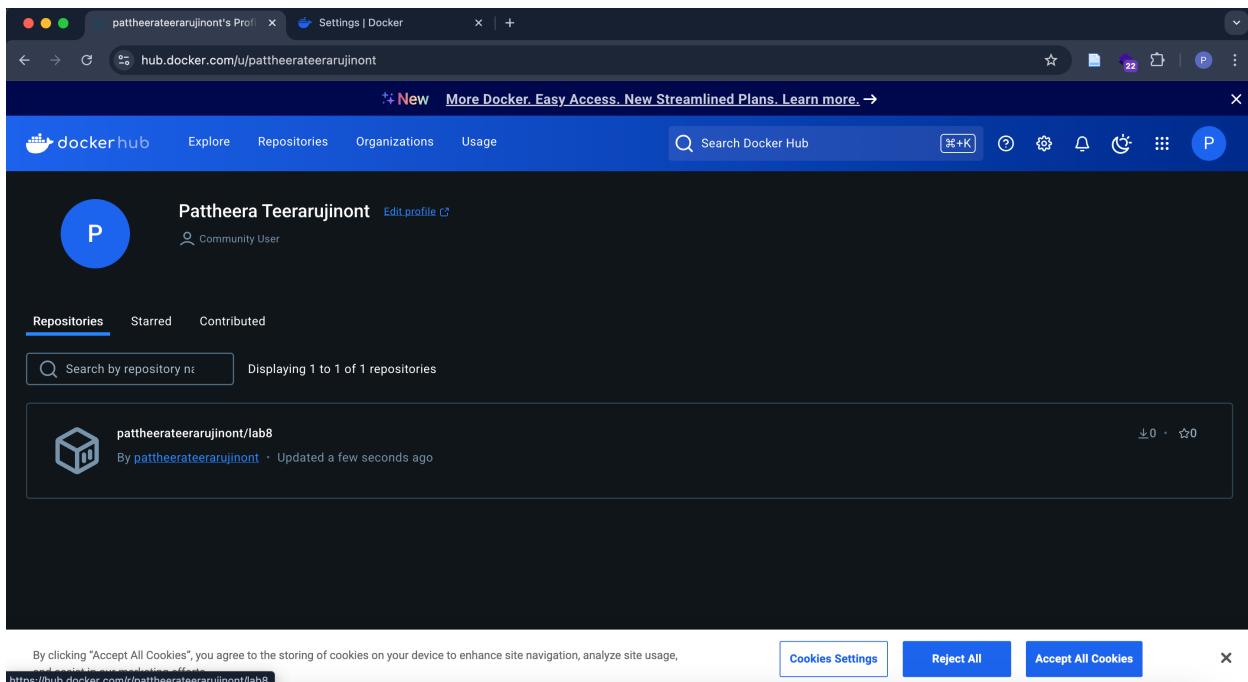
\$ docker login แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง

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

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

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

Lab Worksheet



แบบฝึกปฏิบัติที่ 8.4: การ Build และอัปโหลด Container image และการ Update และอัปโหลด

1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4
2. ทำการ Clone ซอฟต์แวร์ Docker จาก 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) และแสดงที่อยู่ของ Source code ที่ Clone มาและเนื้อหาของไฟล์ package.json

Lab Worksheet

```

{
  "name": "101-app",
  "version": "1.0.0",
  "main": "index.js",
  "license": "MIT",
  "scripts": {
    "prettify": "prettier -l --write \"**/*.js\"",
    "test": "jest",
    "dev": "nodemon src/index.js"
  },
  "dependencies": {
    "express": "^4.18.2",
    "mysql2": "^2.3.3",
    "sqlite3": "5.1.2",
    "uuid": "^9.0.0",
    "wait-port": "^1.0.4"
  },
  "resolutions": {
    "ansi-regex": "5.0.1"
  },
  "prettier": {
    "trailingComma": "all",
    "tabWidth": 4,
    "useTabs": false,
    "semi": true,
    "singleQuote": true
  },
  "devDependencies": {
    "jest": "^29.3.1",
    "nodemon": "^2.0.20",
    "prettier": "^2.7.1"
  }
}

```

4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไฟล์

FROM node:18-alpine

WORKDIR /app

COPY . .

RUN yarn install --production

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

EXPOSE 3000

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้ โดยกำหนดให้ชื่อ image เป็น

myapp_รหัสสนศ. [ไม่มีขีด]

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

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ

Lab Worksheet

The screenshot shows the VS Code interface with a Docker project named 'Lab8_4'. The Explorer sidebar shows files like package.json, Dockerfile, and image.png. The terminal window displays the build process of the Docker image, showing commands like 'FROM node:18-alpine', 'COPY . .', and 'RUN yarn install --production'. The build output includes logs such as 'Building 24.7s (10/10) FINISHED' and 'Successfully built 6533802753'. The status bar at the bottom indicates the build was successful.

```

FROM node:18-alpine
WORKDIR /app
COPY . .
RUN yarn install --production
CMD ["node", "src/index.js"]
EXPOSE 3000

```

```

mixw@PathM Lab8_4 % cd getting-started
mixw@PathM getting-started % cd app
mixw@PathM app % docker build -t myapp_6533802753 .
[+] Building 24.7s (10/10) FINISHED
--> [internal] load build definition from Dockerfile
--> [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: 150B
--> [internal] transfer dockerfile for docker.io/library/node:18-alpine
--> [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
--> [internal] transfer context: 4.60MB
--> [2/4] WORKDIR /app
--> [3/4] COPY .
--> [4/4] RUN yarn install --production
--> exporting to image
--> exporting layers
--> writing image sha256:50866ab8ca958286700388d501dc1ala85bd204aa456b0657fcebd0dd79338058
--> naming to docker.io/library/myapp_6533802753

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/hothvpy27qjqa89su6c2rl706

What's next:
View a summary of image vulnerabilities and recommendations - docker scout quickview
mixw@PathM app %

```

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

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

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

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

The screenshot shows the Docker Desktop dashboard with the build details for 'myapp_6533802753'. It includes a log of the build process, a summary of image vulnerabilities, and a link to 'docker scout quickview'. The status bar at the bottom indicates the build was successful.

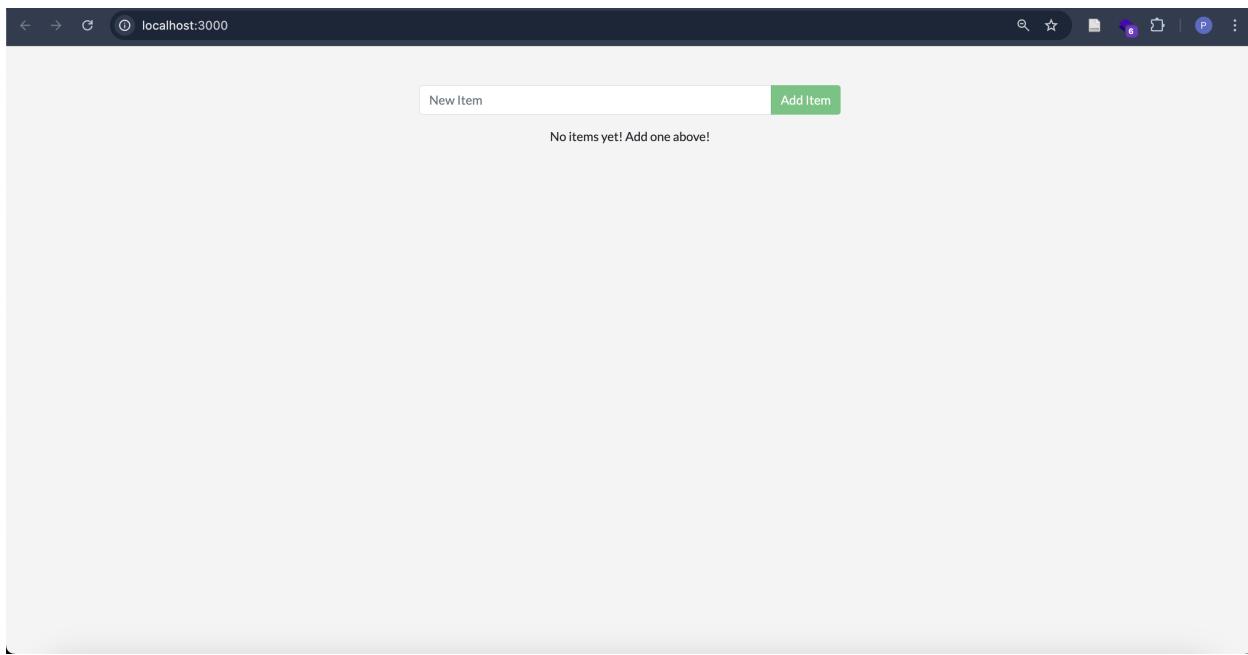
```

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/hothvpy27qjqa89su6c2rl706

What's next:
View a summary of image vulnerabilities and recommendations - docker scout quickview
mixw@PathM app % docker run -dp 3000:3000 myapp_6533802753
9b2c7e143720ae36e9b24862a9138b61a772e60908c744573c274b89a4f6db0
mixw@PathM app %

```

Lab Worksheet



หมายเหตุ: นศ.สามารถทดลองเขียน Web application ที่ทำงานอยู่ได้

8. ทำการแก้ไข Source code ของ Web application ดังนี้

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

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

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

b. Save ไฟล์ให้เรียบร้อย

9. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

10. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

[Check point#10] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

```

14 function TodoListCard() {
15   ...
16   ...
17   ...
18   ...
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   ...

```

```

min@PathM app % docker build -t myapp_6533802753 .
[+] Building 12.4s (10/10) FINISHED
--> [internal] load build definition from Dockerfile
--> transferring dockerfile: 150B
--> [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:a24108da7089c2d293ceaa61fb8969ec10821e8efe25572e5abb10b1841eb70b
--> [internal] load build context
--> transferring context: 7.99kB
--> CACHED [2/4] WORKDIR /app
--> [3/4] COPY .
--> [4/4] RUN yarn install --production
--> exporting to image
--> exporting layers
--> writing image sha256:d34cd24d9cc1c8d46daa73d66948609a0ccdd2a8a77df3f18d7f5ff2493680b1c
--> naming to docker.io/library/myapp_6533802753

```

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/mpiyxtrbu2uj62q3yp7l2qhhs

What's next:

View a summary of image vulnerabilities and recommendations → docker scout quickview

```

min@PathM app % docker run -dp 3000:3000 myapp_6533802753
c23fd3fb56383f3bdead6fdb340e35fb5a95ffff005ac0e2cf165baaff722a8cdd
docker: Error response from daemon: driver failed programming external connectivity on endpoint heuristic_parse (f2db6f0fb90beb2151b7ce8fb90d321f028ca88339b4f41c
80336e39bb3687b3): Bind for 0.0.0.0:3000 failed: port is already allocated.

```

(1) Error ที่เกิดขึ้นหมายความอย่างไร และเกิดขึ้นเพราอะไร

ตอบ หมายความว่า Docker ไม่สามารถใช้พอร์ต 3000 บนเครื่องของคุณได้ เนื่องจากพอร์ตนี้ถูกใช้ งานอยู่แล้วโดยโปรแกรมหรือ Container อื่นในระบบ Error เกิดเพรา พอร์ต 3000 บนเครื่องถูกใช้ งานอยู่แล้ว โดยโปรแกรมหรือ Container อื่น ทำให้ Docker ไม่สามารถจองพอร์ตนี้ได้

11. ลบ Container ของ Web application เวอร์ชันก่อนแก้ไขออกจากระบบ โดยใช้รูปเดิมที่มีดังต่อไปนี้

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

Lab Worksheet

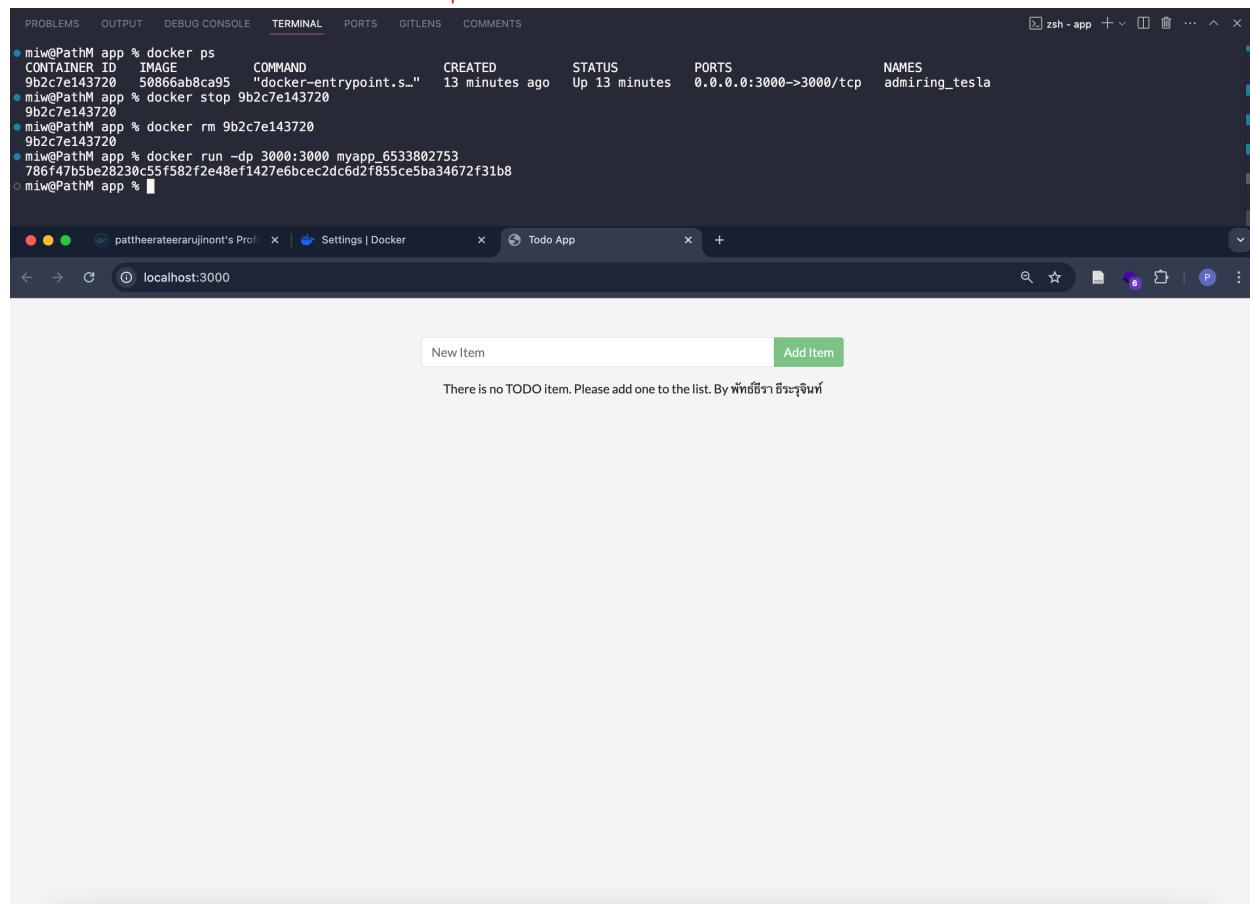
- i. ไปที่หน้าต่าง Containers
- ii. เลือกไอคอนถังขยะในแถบของ Container ที่ต้องการจะลบ
- iii. ยืนยันโดยการกด Delete forever

12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6

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

[Check point#11] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) และผลลัพธ์ที่ได้บน Browser

และ Dashboard ของ Docker desktop



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
```

Lab Worksheet

3. บันทึกรหัสผ่านของ Admin user ให้สำหรับ log-in ในครั้งแรก

[Check point#12] Capture หน้าจอที่แสดงผล Admin password

```

mix@PathM Lab8_5 % docker run -p 8080:8080 -p 50000:50000 lab8_jenkins
Running from: /usr/share/jenkins/jenkins.war
webroot: /var/jenkins_home/war
2025-01-22 10:27:30.383+0000 [id=1]     INFO    winstone.Logger#logInternal: Beginning extraction from war file
2025-01-22 10:27:31.344+0000 [id=1]     WARNING o.e.j.ee9.nested.ContextHandler#setContextPath: Empty contextPath
2025-01-22 10:27:31.387+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: jetty-12.0.16; built: 2024-12-09T21:02:54.535Z; git: c3f88bafb4e393f232
04dc14dc570d042e84debc7; jvm 17.0.13+11
2025-01-22 10:27:31.726+0000 [id=1]     INFO    o.e.j.e.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.ee9.jsp.JettyJspServlet
2025-01-22 10:27:31.766+0000 [id=1]     INFO    o.e.j.s.DefaultSessionIdManager#doStart: Session workerName=node0
2025-01-22 10:27:32.099+0000 [id=1]     INFO    hudson.WebAppMain#contextInitialized: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get("JENKINS_HOME")
2025-01-22 10:27:32.218+0000 [id=1]     INFO    o.e.j.s.handler.ContextHandler#doStart: Started oeje9n.ContextHandler$CoreContextHandler@772861aa{[Jenkins v2.47f3., b-file:///var/jenkins_home/war/, a=AVAILABLE,h=oeje9n.ContextHandler$CoreContextHandler$CoreToNestedHandler@6631cb64{STARTED}}
2025-01-22 10:27:32.229+0000 [id=1]     INFO    o.e.j.server.AbstractConnector#doStart: Starter ServerConnector@c35af2a{HTTP/1.1, {http/1.1}}{0.0.0.0:8080}
2025-01-22 10:27:32.243+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: Started oejs.Server@1c7fd41f{[STARTING]}[12.0.16,sto=0] @2539ms
2025-01-22 10:27:32.245+0000 [id=26]    INFO    winstone.Logger#logInternal: Winstone Servlet Engine running: controlPort=disabled
2025-01-22 10:27:32.434+0000 [id=34]    INFO    jenkins.InitReactorRunners$1#onAttained: Started initialization
2025-01-22 10:27:32.451+0000 [id=40]    INFO    jenkins.InitReactorRunners$1#onAttained: Listed all plugins
2025-01-22 10:27:33.171+0000 [id=47]    INFO    jenkins.InitReactorRunners$1#onAttained: Prepared all plugins
2025-01-22 10:27:33.177+0000 [id=43]    INFO    jenkins.InitReactorRunners$1#onAttained: Started all plugins
2025-01-22 10:27:33.178+0000 [id=40]    INFO    jenkins.InitReactorRunners$1#onAttained: Augmented all extensions
2025-01-22 10:27:33.376+0000 [id=36]    INFO    jenkins.InitReactorRunners$1#onAttained: System config loaded
2025-01-22 10:27:33.377+0000 [id=36]    INFO    jenkins.InitReactorRunners$1#onAttained: System config adapted
2025-01-22 10:27:33.378+0000 [id=36]    INFO    jenkins.InitReactorRunner$1#onAttained: Loaded all jobs
2025-01-22 10:27:33.379+0000 [id=44]    INFO    jenkins.InitReactorRunners$1#onAttained: Configuration for all jobs updated
2025-01-22 10:27:33.404+0000 [id=60]    INFO    hudson.util.Retrier#start: Attempt #1 to do the action check updates server
2025-01-22 10:27:33.887+0000 [id=45]    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:
***** 78118acc75764c6bdbf13a48b8945c530
***** This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
***** 
***** Jenkins initial setup is required. An admin user has been created and a password generated.
***** Please use the following password to proceed to installation:
***** 78118acc75764c6bdbf13a48b8945c530
***** Jenkins initial setup is required. An admin user has been created and a password generated.
***** Please use the following password to proceed to installation:
***** 78118acc75764c6bdbf13a48b8945c530
2025-01-22 10:27:39.017+0000 [id=38]    INFO    jenkins.InitReactorRunners$1#onAttained: Completed initialization
2025-01-22 10:27:39.040+0000 [id=25]    INFO    hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
2025-01-22 10:27:41.344+0000 [id=60]    INFO    h.m.DownloadServiceDownloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
2025-01-22 10:27:41.345+0000 [id=60]    INFO    hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1

```

Ln 12, Col 13 Spaces: 4 UTF-8 LF Dockerfile ⚙ Go Live 🌐 ⓘ Prettier ▾

4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น

localhost:8080

5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3

6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสีตัวท้าย เช่น somsri_3062

[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

Lab Worksheet

Getting Started

Create First Admin User

Username
pattheera_2753

Password
.....

Confirm password
.....

Full name
Pattheera Teerarujinont

E-mail address
pattheera.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

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job +

Set up a distributed build

Set up an agent ☒

Configure a cloud ☁

Learn more about distributed builds ?

Build Queue
No builds in the queue.

Build Executor Status 0/2

REST API Jenkins 2.479.3

9. เลือก Manage Jenkins และไปที่เมนู Plugins

Lab Worksheet

The screenshot shows the Jenkins Manage Jenkins interface. At the top, there's a message: "It appears that your reverse proxy set up is broken." Below this, the "System Configuration" section includes links for Build Queue (No builds in the queue), Tools (Configure tools, their locations and automatic installers), Plugins (Add, remove, disable or enable plugins that can extend the functionality of Jenkins), and Nodes (Add, remove, control and monitor the various nodes that Jenkins runs jobs on). The "Security" section includes links for Security (Secure Jenkins; define who is allowed to access/use the system), Credentials (Configure credentials), Credential Providers (Configure the credential providers and types), and Users (Create/delete/modify users that can log in to this Jenkins). The "Status Information" section includes links for System Information (Displays various environmental information to assist trouble-shooting), System Log (System log captures output from java.util.logging output related to Jenkins), Load Statistics (Check your resource utilization and see if you need more computers for your builds), and About Jenkins (See the version and license information).

10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม

The screenshot shows the Jenkins Plugin Manager. The left sidebar has links for Updates, Available plugins (selected), Installed plugins, Advanced settings, and Download progress. The main area shows a search bar with "robot" and a list of available plugins. One plugin is highlighted: "Robot Framework 5.0.0" by "Build Reports". It is described as "This publisher stores Robot Framework test reports for builds and shows summaries of them in project and build views along with trend graph." The status is "Released" and it was updated "2 mo 7 days ago". There is an "Install" button with a progress bar.

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

Lab Worksheet

New Item

Enter an item name

Select an item type

- Freestyle project**
Classic general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

OK

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

Description: Lab 8.5

GitHub project: กดเลือก แล้วใส่ Project URL เป็น repository ที่เก็บໂດຍ .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 พิริยมกับตอบคำถามต่อไปนี้

Lab Worksheet

The screenshot shows the Jenkins 'Configure' screen for a job named 'UAT'. The 'Build Triggers' section has 'Build periodically' selected with a schedule of 'H/15 * * * *'. The 'Build Environment' section contains several optional checkboxes. The 'Build Steps' section shows a single step: 'Execute shell' with the command 'robot UAT-Lab7-resource.robot'. At the bottom are 'Save' and 'Apply' buttons.

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

robot UAT-Lab7-resource.robot

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

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

ของกราฟทดสอบที่ผ่านแล้วนับว่าซอฟต์แวร์มีอยู่ในสถานะที่สามารถนำไปใช้งานได้ (เช่น 20, 80)

13. กด Apply และ Save

14. ล้าง Build Now

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

Lab Worksheet

```

Started by user Pattheera Teerarujinont
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/UAT/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Pattheera-Teerarujinont/lab8_jenkins.git/ # timeout=10
Fetching upstream changes from https://github.com/Pattheera-Teerarujinont/lab8_jenkins.git/
> git -version # timeout=10
> git fetch --tags --force --progress -- https://github.com/Pattheera-Teerarujinont/lab8_jenkins.git/ +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision fd23b665f5ee28e12fcf7587c9e5e61e5823d7c (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f fd23b665f5ee28e12fcf7587c9e5e61e5823d7c # timeout=10
Commit message: "delete file UAT-Lab7-resource and add UAT-Lab7-001"
> git rev-list --no-walk a786db773312a3d57c98fb0b3b6774a8075de # timeout=10
[UAT] $ ./bin/sh -xe /tmp/jenkins15022123065182508751.sh
+ robot UAT-Lab7-001.robot
[ERROR] Error in file '/var/jenkins_home/workspace/UAT/UAT-Lab7-001.robot' on line 2: Resource file 'UAT-Lab7-resource.robot' does not exist.

=====
UAT-Lab7-001
=====
NO_1: Open Form | FAIL |
No keyword with name 'Open Browser To Form Page' found.

NO_2: Record Success | FAIL |
No keyword with name 'Input Firstname' found.

=====
UAT-Lab7-001
=====
2 tests, 0 passed, 2 failed
=====
Output: /var/jenkins_home/workspace/UAT/output.xml
Log: /var/jenkins_home/workspace/UAT/log.html
Report: /var/jenkins_home/workspace/UAT/report.html
Build step 'Execute shell' marked build as failure
Robot results publisher started...
INFO: Checking test criticality is deprecated and will be dropped in a future release!
-Parsing output xml:
Done!
-Copying log files to build dir:

```