

Lab Worksheet

ชื่อ-นามสกุล นางสาวเพ็ชรภา นุทอง รหัสนักศึกษา 653380144-8 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

```

Last login: Wed Jan 22 14:26:27 on ttys002
phetchapa@Pangies-MacBook softEn % mkdir Lab8_1

phetchapa@Pangies-MacBook softEn % cd Lab8_1

phetchapa@Pangies-MacBook Lab8_1 % docker pull busybox

Using default tag: latest
latest: Pulling from library/busybox
559c60843878: Download complete
Digest: sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab073cd3c138bd458fc8257fbf1
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
phetchapa@Pangies-MacBook Lab8_1 % docker images

REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
busybox        latest    a5d0ce49aa80   3 months ago   6.02MB
phetchapa@Pangies-MacBook Lab8_1 %

```

(1) ลิสต์อยู่ภายใต้คอลัมน์ Repository คืออะไร

ชื่อของ Image Repository หรือที่เก็บของ Docker Image ที่ดึงมาหรือสร้างขึ้น ซึ่งในการนี้คือ busybox หมายถึงชื่อของ Image ที่ถูกดึงมาจาก Docker Hub

(2) Tag ที่ใช้เป็นอย่างไร

ใช้ระบุเจาะชั้นหรือสถานะของ Image ใน Repository นั้น ๆ เช่น latest หมายถึง Image ล่าสุดที่ถูกดึงมาใช้ ถ้าไม่มีการกำหนด Tag ตอนดึง Image จะใช้ค่า latest เป็นค่าเริมต้น โดย Tag ช่วยในการจัดการเวอร์ชันต่าง ๆ ของ Image

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

Lab Worksheet

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

```
Last login: Mon Jan 27 07:16:39 on ttys000
phetchapa@Pangies-MacBook Lab8_1 % docker run busybox

phetchapa@Pangies-MacBook Lab8_1 % docker run -it busybox sh

/ # ls
bin    dev    etc    home   lib    lib64   proc    root    sys    tmp    usr    var
/ # ls -la
total 48
drwxr-xr-x  1 root    root        4096 Jan 27 08:22 .
drwxr-xr-x  1 root    root        4096 Jan 27 08:22 ..
-rwxr-xr-x  1 root    root        0 Jan 27 08:22 .dockerenv
drwxr-xr-x  2 root    root      12288 Sep 26 21:31 bin
drwxr-xr-x  5 root    root       360 Jan 27 08:22 dev
drwxr-xr-x  1 root    root        4096 Jan 27 08:22 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  219 root   root        0 Jan 27 08:22 proc
drwx-----  1 root    root        4096 Jan 27 08:22 root
dr-xr-xr-x  11 root   root        0 Jan 27 08:22 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
phetchapa@Pangies-MacBook Lab8_1 % docker run busybox echo "Hello Phetchapa Nuthong from busybox"
Hello Phetchapa Nuthong from busybox
phetchapa@Pangies-MacBook Lab8_1 % docker ps -a

CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
TS      NAMES
98e44542a124        busybox            "echo 'Hello Phetcha..."   12 seconds ago   Exited (0) 11 seconds ago
          competent_joliot
23981e96add6        busybox            "sh"                2 minutes ago    Exited (0) About a minute ago
          trusting_jemison
47290696f8d3        busybox            "sh"                3 minutes ago    Exited (0) 3 minutes ago
          wizardly_heisenberg
phetchapa@Pangies-MacBook Lab8_1 %
```

- (1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
 - i เปิดให้งานใน模式อินพุตแบบ Interactive ซึ่งจะทำให้ Container ยอมรับการป้อนข้อมูลจากผู้ใช้ผ่าน Standard Input (stdin) ได้
 - t เปิดให้งานการจัดการ Terminal เพื่อให้ผู้ใช้สามารถติดต่อกับ Container ได้เหมือนใช้ Terminal ปกติ เช่น การรันคำสั่ง sh เพื่อเข้าสู่ Shell

- (2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร
 - ใช้แสดงสถานะของ Container ว่ากำลังทำงานอยู่หรือหยุดทำงานไปแล้ว พร้อมกับข้อมูลเกี่ยวกับระยะเวลาที่เกิดสถานะนั้น เช่น Exited (0) 11 seconds ago Container หยุดทำงานโดยไม่มีข้อผิดพลาดเมื่อ 11 วินาทีที่แล้ว

Lab Worksheet

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

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

```

phetchapa@Pangies-MacBook Lab8_1 % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS N
AMES
98e44542a124 busybox "echo 'Hello Phetcha..." 23 hours ago Exited (0) 23 hours ago c
ompetent_joliot
23981e96add6 busybox "sh" 23 hours ago Exited (0) 23 hours ago t
rusting_jemison
47290696f8d3 busybox "sh" 23 hours ago Exited (0) 23 hours ago w
izardly_heisenberg
phetchapa@Pangies-MacBook Lab8_1 % docker rm 98e44542a124
98e44542a124
phetchapa@Pangies-MacBook Lab8_1 % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
23981e96add6 busybox "sh" 23 hours ago Exited (0) 23 hours ago trusting_jemison
47290696f8d3 busybox "sh" 23 hours ago Exited (0) 23 hours ago wizardly_heisenb
erg
phetchapa@Pangies-MacBook Lab8_1 %

```

แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

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

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

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

Lab Worksheet

\$ docker build -t <ชื่อ Image> .

6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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

```
phetchapa@Pangies-MacBook Lab8_1 % cd ..
phetchapa@Pangies-MacBook softEn % mkdir Lab8_2
phetchapa@Pangies-MacBook softEn % cd Lab8_2
phetchapa@Pangies-MacBook Lab8_2 % cat > Dockerfile << EOF
FROM busybox
CMD echo "Hi there. This is my first docker image."
CMD echo "Phetchapa Nuthong 653380144-8 Pond"
[EOF
phetchapa@Pangies-MacBook Lab8_2 % docker build -t my-first-image .

[+] Building 5.8s (6/6) FINISHED                                            docker:desktop-linux
=> [internal] load build definition from Dockerfile                      0.0s
=> => transferring dockerfile: 148B                                         0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavio 0.0s
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the s 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavio 0.0s
=> [internal] load metadata for docker.io/library/busybox:latest          0.0s
=> [internal] load .dockerrcignore                                         0.0s
=> => transferring context: 2B                                           0.0s
=> [1/1] FROM docker.io/library/busybox:latest@sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab0 5.7s
=> => resolve docker.io/library/busybox:latest@sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab0 5.7s
=> [auth] library/busybox:pull token for registry-1.docker.io               0.0s
=> exporting to image                                                     0.0s
=> => exporting layers                                                    0.0s
=> => exporting manifest sha256:34868c3def8000d48a55ce04906e3942e4c306d9b6354482f0822e86f06185 0.0s
=> => exporting config sha256:4801a34e9c99b6cc74f5af07a4418f6dbbd168b740c7af6f326cb75b60a44b4f 0.0s
=> => exporting attestation manifest sha256:ee34fa6f48dfb43383e606dc47781ba3da02d7dc937300698f 0.0s
=> => exporting manifest list sha256:5272cf8489ae1938533eff1c579e063e04d37a7e6e5dfc09b2a7cc636 0.0s
=> => naming to docker.io/library/my-first-image:latest                   0.0s
=> => unpacking to docker.io/library/my-first-image:latest                 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/a2zqeugh2s7je70kr7krj
mq6b

3 warnings found (use docker --debug to expand):
- 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 0 signals (line 3)
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to 0 signals (line 2)
phetchapa@Pangies-MacBook Lab8_2 % docker run my-first-image

Phetchapa Nuthong 653380144-8 Pond
phetchapa@Pangies-MacBook Lab8_2 %
```

- (1) คำสั่งที่ใช้ในการ run คือ

docker run <ชื่อ Image>

ในที่นี้คือ docker run my-first-image

Lab Worksheet

- (2) Option -t ในคำสั่ง \$ docker build ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสั้นๆ เช่น ถ้าใช้ -t ในคำสั่ง docker build เราสามารถกำหนดชื่อให้กับ Docker Image ได้เอง โดยไม่ต้องใช้ IMAGE ID ในการรันหรือจัดการ Image ภายหลัง

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เครื่องเรา
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_3
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_3 เพื่อให้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory
สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดว์ 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> /lab8
5. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง
\$ docker run <username> /lab8

Lab Worksheet

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

```
phetchapa@Pangies-MacBook Lab8_2 % cd ..
phetchapa@Pangies-MacBook softEn % mkdir Lab8_3
phetchapa@Pangies-MacBook softEn % cd Lab8_3

phetchapa@Pangies-MacBook Lab8_3 % touch Dockerfile

phetchapa@Pangies-MacBook Lab8_3 % nano Dockerfile
```

```
FROM busybox
CMD echo "Hi there. My work is done. You can run them from my Docker image."
CMD echo "Phetchapa Nuthong 653380144-8"
```

```
[phetchapa@Pangies-MacBook Lab8_3 % docker build -t phetchapa/lab8 .
[+] Building 3.2s (6/6) FINISHED
  => [internal] load build definition from Dockerfile
  => => transferring dockerfile: 171B
  => WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavio 0.0s
  => WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the s 0.0s
  => WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavio 0.0s
  => [internal] load metadata for docker.io/library/busybox:latest 0.0s
  => [internal] load .dockerrcignore 0.0s
  => => transferring context: 2B
  => CACHED [1/1] FROM docker.io/library/busybox:latest@sha256:a5d0ce49aa801d475da48f8cb163c354a 3.1s
  => => resolve docker.io/library/busybox:latest@sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab0 3.1s
  => [auth] library/busybox:pull token for registry-1.docker.io 0.0s
  => exporting to image 0.0s
  => => exporting layers 0.0s
  => => exporting manifest sha256:9cb096be5d10cc12402a244e869b9212eddfa8bad251563836fb3432920fd7 0.0s
  => => exporting config sha256:0cb491613be01f96c99eb699a384f448de9c89b247bab9935ba0dbe1a396740d 0.0s
  => => exporting attestation manifest sha256:55a8f6d27dead6ca8068be983b1b35edc42145e2cee9416289 0.0s
  => => exporting manifest list sha256:809c2b175dc5c51a848be53fb78f798c4528f6e5de0f1c09eae416919 0.0s
  => => naming to docker.io/phetchapa/lab8:latest 0.0s
  => => unpacking to docker.io/phetchapa/lab8:latest 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/x34x0y479vqrubjqsozt49407

3 warnings found (use docker --debug to expand):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to 0 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 0 signals (line 3)

[phetchapa@Pangies-MacBook Lab8_3 % docker run phetchapa/lab8
Phetchapa Nuthong 653380144-8
phetchapa@Pangies-MacBook 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

Lab Worksheet

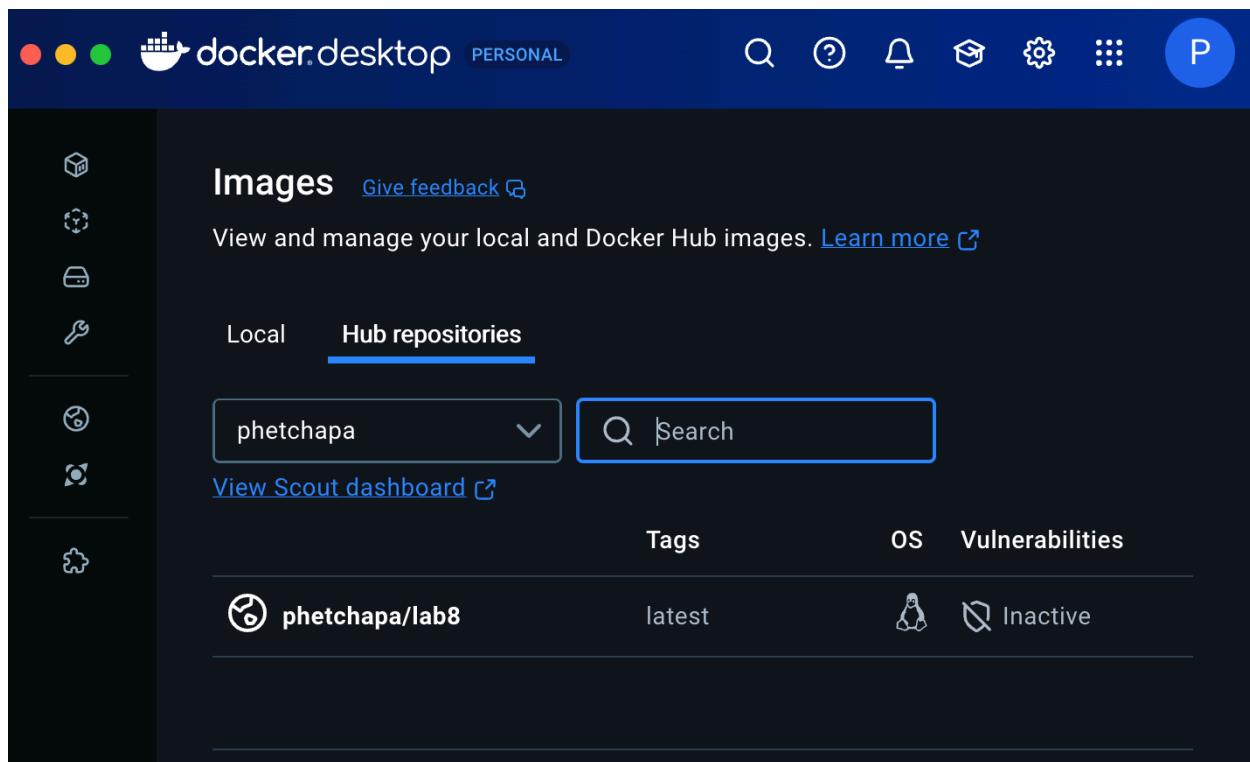
\$ docker login -u <username> -p <password>

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

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

```
[phetchapa@Pangies-MacBook Lab8_3 % docker push phetchapa/lab8
Using default tag: latest
The push refers to repository [docker.io/phetchapa/lab8]
983a814b35cb: Pushed
559c60843878: Mounted from library/busybox
latest: digest: sha256:806c2b175dc5c51a848be53fb78f798c4528f6e5de0f1c09eae41691991eddaf size: 855
phetchapa@Pangies-MacBook Lab8_3 % ]
```

Name	Last Pushed	Contains	Visibility
phetchapa/lab8	less than a minute ago	IMAGE	Public



แบบฝึกปฏิบัติที่ 8.4: การ Build และ Update Container image และการ Update และ Open Container

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

```
[phetchapa@Pangies-MacBook Lab8_3 % cd ..
[phetchapa@Pangies-MacBook softEn % mkdir Lab8_4
phetchapa@Pangies-MacBook softEn % cd Lab8_4

phetchapa@Pangies-MacBook 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 | 7.12 MiB/s, done.
Resolving deltas: 100% (523/523), done.
[phetchapa@Pangies-MacBook Lab8_4 % cd getting-started/app

[phetchapa@Pangies-MacBook app % ls -l

total 296
-rw-r--r-- 1 phetchapa staff 645 Jan 28 14:31 package.json
drwxr-xr-x 4 phetchapa staff 128 Jan 28 14:31 spec
drwxr-xr-x 6 phetchapa staff 192 Jan 28 14:31 src
-rw-r--r-- 1 phetchapa staff 147266 Jan 28 14:31 yarn.lock
[phetchapa@Pangies-MacBook app % nano package.json

phetchapa@Pangies-MacBook app % ]
```

```
UW PICO 5.09 File: package.json

{
  "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"
  }
}
```

Lab Worksheet

4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไว้ในไฟล์

```
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

```
[phetchapa@Pangies-MacBook app % touch Dockerfile
[phetchapa@Pangies-MacBook app % nano Dockerfile
```

```
UW PICO 5.09 File: Dockerfile
FROM node:18-alpine
WORKDIR /app
COPY ..
RUN yarn install --production
CMD ["node", "src/index.js"]
EXPOSE 3000
```

Lab Worksheet

```
phetchapa@Pangies-MacBook app % docker build -t myapp_6533801448 .

[+] Building 27.4s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 151B
=> [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@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d9
=> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d9
=> => sha256:e668eba0f82bcfec9fb0cd787bd7f3d013a1266567b092e90ff8b3d3be41807c 443B / 443B 0.0s
=> => sha256:fc4eb59aab89271acc5a8bd8e5e96fc17af489976964461471ea28fc8e0be459 1.26MB / 1.26MB 0.0s
=> => sha256:4fe16fa8f46966191d59cffcabfff137a623b3cdda747d387bd85dcbf0feff3d 39.66MB / 39.66MB 0.0s
=> => sha256:52f827f723504aa3325bb5a54247f0dc4b92bb72569525bc951532c4ef679bd4 3.99MB / 3.99MB 0.0s
=> => extracting sha256:52f827f723504aa3325bb5a54247f0dc4b92bb72569525bc951532c4ef679bd4
=> => extracting sha256:4fe16fa8f46966191d59cffcabfff137a623b3cdda747d387bd85dcbf0feff3dd
=> => extracting sha256:fc4eb59aab89271acc5a8bd8e5e96fc17af489976964461471ea28fc8e0be459
=> => extracting sha256:e668eba0f82bcfec9fb0cd787bd7f3d013a1266567b092e90ff8b3d3be41807c
=> [internal] load build context
=> => transferring context: 4.59MB
=> [2/4] WORKDIR /app
=> [3/4] COPY .
=> [4/4] RUN yarn install --production
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:64498b4dbfa4f5ab5b2447e75545efeecc7c0d451c3f3ba2a7c421b69c7c6ff
=> => exporting config sha256:f9b3c22e637c0121ab08cc6807c2bcb5b47ded3e4664234d64df21d17ad75ac8
=> => exporting attestation manifest sha256:4c98fb599f15993bcf516d247dfb64e2b41b1e4c8dd568cd3
=> => exporting manifest list sha256:f2ff3b920dd71cbc8b70afa62582a8d3afa79123be1b0c2d06c0c923d
=> => naming to docker.io/library/myapp_6533801448:latest
=> => unpacking to docker.io/library/myapp_6533801448:latest

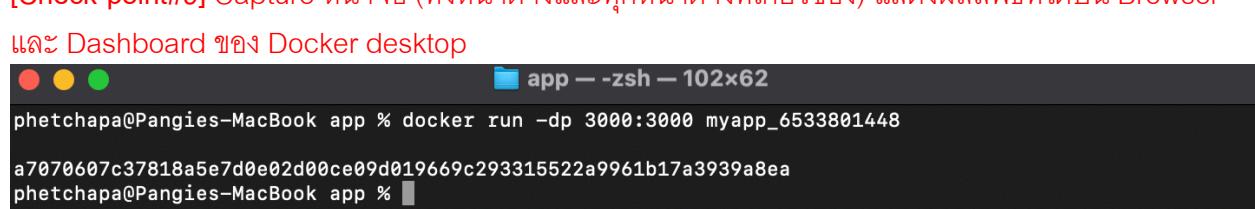
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/maumdbx3cqpelgf31576w
ytzd
phetchapa@Pangies-MacBook app %
```

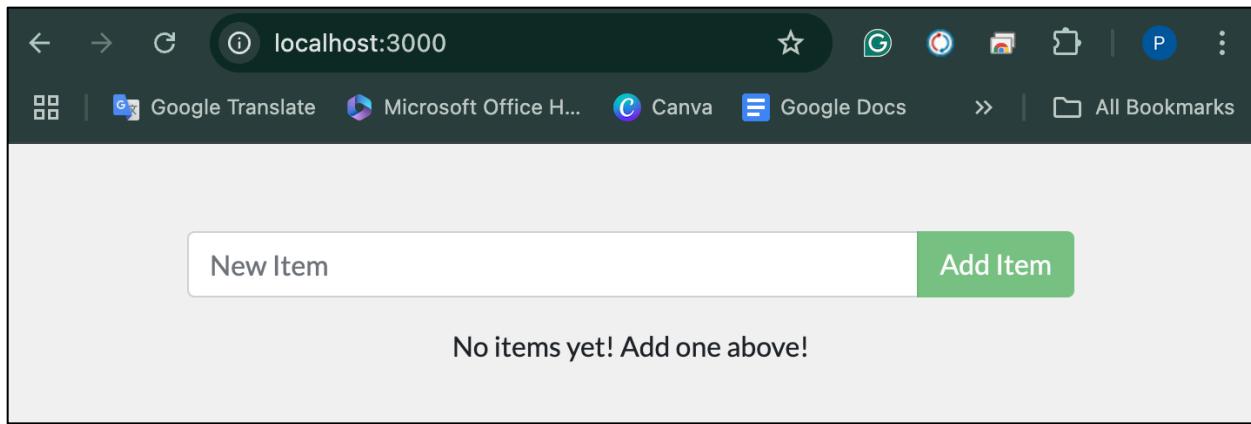
6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

\$ docker run -dp 3000:3000 <myapp_รหัส> [ไม่มีชื่อ]

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

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





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

Lab Worksheet

```

14 function TodoListCard() {
15   ...
16   const onItemRemoval = React.useCallback(
17     item => {
18       const index = items.findIndex(i => i.id === item.id);
19       setItems([...items.slice(0, index), ...items.slice(index + 1)]);
20     },
21     [items],
22   );
23 
24   if (items === null) return 'Loading...';
25 
26   return (
27     <React.Fragment>
28       <AddItemForm onNewItem={onNewItem} />
29       {items.length === 0 && (
30         <p className="text-center">There is no TODO item. Please add one to the list. By Phetchapa Nuthong</p>
31       )}
32       {items.map(item => (
33         <ItemDisplay
34           item={item}
35           key={item.id}
36           onItemUpdate={onItemUpdate}
37           onItemRemoval={onItemRemoval}
38         />
39       ))}
40     </React.Fragment>
41   );
42 }
43 
44 function AddItemForm({ onNewItem }) {
45   const { Form, InputGroup, Button } = ReactBootstrap;
46 
47   const [newItem, setNewItem] = React.useState('');
48   const [submitting, setSubmitting] = React.useState(false);
49 
50   const submitNewItem = e => {
51     e.preventDefault();
52     setSubmitting(true);
53     fetch('/items', {
54       method: 'POST',
55       body: JSON.stringify({ name: newItem }),
56       headers: { 'Content-Type': 'application/json' },
57     })
58   }
59 }
60 
```

Lab Worksheet

```

phetchapa@Pangies-MacBook app % code src/static/js/app.js
phetchapa@Pangies-MacBook app % docker build -t myapp_6533801448 .

[+] Building 18.7s (10/10) FINISHED
--> [internal] load build definition from Dockerfile          docker:desktop-linux
--> => transferring dockerfile: 151B                         0.0s
--> [internal] load metadata for docker.io/library/node:18-alpine      0.0s
--> [auth] library/node:pull token for registry-1.docker.io      3.6s
--> [internal] load .dockerignore                                0.0s
--> => transferring context: 2B                                0.0s
--> [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d9 0.0s
--> => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d9 0.0s
--> [internal] load build context                            0.0s
--> => transferring context: 7.95kB                          0.0s
--> CACHED [2/4] WORKDIR /app                               0.0s
--> [3/4] COPY . .                                         0.0s
--> [4/4] RUN yarn install --production                   12.4s
--> exporting to image                                     2.6s
--> => exporting layers                                    1.9s
--> => exporting manifest sha256:0e1f45630b7bc3068d9b63550ce2e001368c3718925c1ac7a0bac2a25b6ad4 0.0s
--> => exporting config sha256:fec38ab1770b8aaa8382a4f8b718e3b0f7cc0a397a852986d006ccaf4e47e659 0.0s
--> => exporting attestation manifest sha256:51746c9b4e0251d6c8ab6dbbe5149b3cc2ab3223ef258dfe32 0.0s
--> => exporting manifest list sha256:4a8f5fd7d27f8aa2b390595dca7cf55e40901d8fa995b5de81da72e47 0.0s
--> => naming to docker.io/library/myapp_6533801448:latest 0.0s
--> => unpacking to docker.io/library/myapp_6533801448:latest 0.6s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/ntsy8s72bokdod598gs0j
ghc
phetchapa@Pangies-MacBook app % docker run -dp 3000:3000 myapp_6533801448
d3af24f747f9d87d47a3213eb151e90ceeaa9e67970447a6e4a06baf5183fb382
docker: Error response from daemon: driver failed programming external connectivity on endpoint pedant_ic_mendeleev (3d83022c692d4e44d057c60817b3b37404c2121334d02aa8173705c204a67d61): Bind for 0.0.0.0:3000 failed: port is already allocated.
phetchapa@Pangies-MacBook app %

```

(1) Error ที่เกิดขึ้นหมายความอย่างไร และเกิดขึ้นเพราอะไร

Error "port is already allocated" หมายถึง Port 3000 บนเครื่องถูกใช้งานอยู่แล้ว ทำให้ Docker ไม่สามารถ Bind Port 3000 ให้กับ Container ใหม่ได้

11. ลบ Container ของ Web application เวอร์ชันก่อนแก้ไขออกจากระบบ โดยใช้วิธีใดวิธีหนึ่งดังต่อไปนี้

a. ผ่าน Command line interface

- ใช้คำสั่ง \$ docker ps เพื่อดู Container ID ที่ต้องการจะลบ
- Copy หรือบันทึก Container ID ไว้
- ใช้คำสั่ง \$ docker stop <Container ID ที่ต้องการจะลบ> เพื่อหยุดการทำงานของ Container ดังกล่าว
- ใช้คำสั่ง \$ docker rm <Container ID ที่ต้องการจะลบ> เพื่อทำการลบ

b. ผ่าน Docker desktop

- ไปที่หน้าต่าง Containers
- เลือกไอคอนถังขยะในແກ່ວຂອງ Container ที่ต้องการจะลบ

Lab Worksheet

iii. ยืนยันโดยการกด Delete forever

12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6

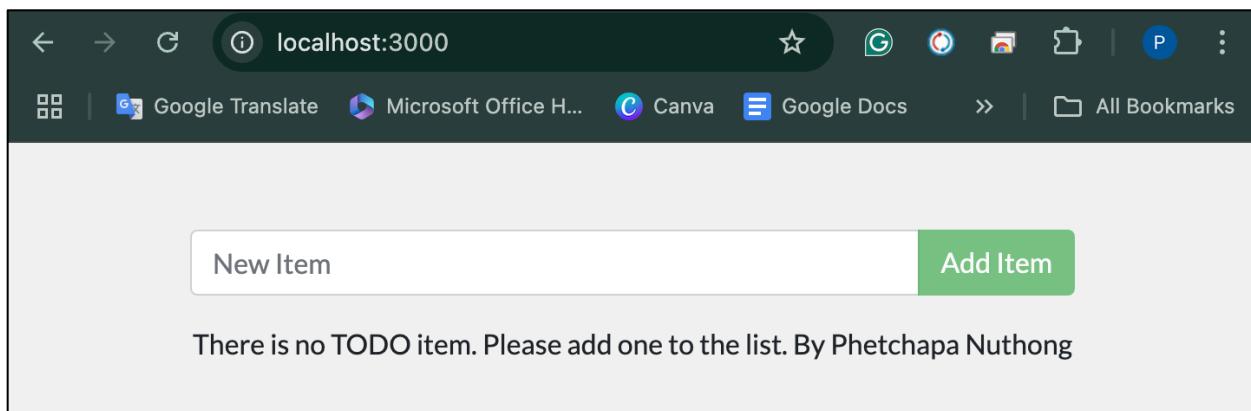
13. เปิด Browser ไปที่ URL = <http://localhost:3000>

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

```

phetchapa@Pangies-MacBook app % docker ps
CONTAINER ID   IMAGE          COMMAND       CREATED      STATUS        PORTS
NAMES
a7070607c378   f2ff3b920dd7   "docker-entrypoint.s..."  15 minutes ago   Up 15 minutes   0.0.0.0:3000->
3000/tcp
[nostalgic_chatterjee]
[phetchapa@Pangies-MacBook app % docker stop a7070607c378
a7070607c378
[phetchapa@Pangies-MacBook app % docker rm a7070607c378
a7070607c378
[phetchapa@Pangies-MacBook app % docker run -dp 3000:3000 myapp_6533801448
5f1c6cf87ff73a33645dd1a5041051270e4a0cd7e202753b514b3b57fb74347f
phetchapa@Pangies-MacBook app %

```



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

Lab Worksheet

[Check point#12] Capture หน้าจอที่แสดงผล Admin password

```
Archive — docker run -p 8080:8080 -p 50000:50000 --restart=on-failure jenkins/jenkins:lts-jdk1...
Last login: Tue Jan 28 13:52:05 on ttys000
[phetchapa@Pangies-MacBook Archive % docker compose up --build -d
[+] Running 0/1
[+] Building 10.3s (5/6)
[+] Building 39.0s (8/8) FINISHED
    => [jenkins internal] load build definition from Dockerfile          docker:desktop-linux
    => => transferring dockerfile: 523B                                     docker:desktop-linux
    => [jenkins internal] load metadata for docker.io/jenkins/jenkins:lts      0.0s
    => [jenkins auth] jenkins/jenkins:pull token for registry-1.docker.io      0.0s
    => [jenkins internal] load .dockereignore                                0.0s
    => => transferring context: 2B                                         0.0s
    => [jenkins 1/2] FROM docker.io/jenkins/jenkins:lts@sha256:dc56634cc8fa476f36eba16d7db6c6bc1f5  0.1s
    => => resolve docker.io/jenkins/jenkins:lts@sha256:dc56634cc8fa476f36eba16d7db6c6bc1f5e3c70624  0.0s
    => [jenkins 2/2] RUN apt-get update && apt-get install -y python3 python3-pip && pip3  24.9s
    => [jenkins] exporting to image                                         10.5s
    => => exporting layers                                              8.9s
    => => exporting manifest sha256:e682bed155b5bad4bde5c65970c9620d43416d0625cb90d12ab2ab6bf9d10d  0.0s
    => => exporting config sha256:d5ac53b2b2c272fa46aa577b272cb2f84be7230e746882e4af14633a2656d43  0.0s
    => => exporting attestation manifest sha256:6d65757f3dd013c42ae1fa4fa1505aa0a847ba2495a99d83c0  0.0s
    => => exporting manifest list sha256:ed55b9023991046514fe09ce61a7a45839acd7e86a3f274b52c7bc40b  0.0s
    => => naming to docker.io/library/archive-jenkins:latest                0.0s
[+] Running 3/3g to docker.io/library/archive-jenkins:latest             1.6s
✓ Service jenkins           Built                                         39.2s
✓ Network archive_default   Created                                       0.0s
✓ Container jenkins-modified Started                                      0.5s
```

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

The image contains two side-by-side screenshots of a web browser displaying the Jenkins setup interface.

Left Screenshot:

- Header: localhost:8080/login?from=%2F
- Title: Getting Started
- Section: UNLOCK JENKINS
- Text: To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:
- Text: /var/jenkins_home/secrets/initialAdminPassword
- Text: Please copy the password from either location and paste it below.
- Text: Administrator password: (redacted)
- Button: Continue

Right Screenshot:

- Header: localhost:8080
- Title: Getting Started
- Section: Create First Admin User
- Form fields:
 - Username: phetchapa_1448
 - Password: (redacted)
 - Confirm password: (redacted)
 - Full name: Phetchapa Nuthong
 - E-mail address: phetchapa.n@kku.edu
- Text: Jenkins 2.479.3
- Buttons: Skip and continue as admin, Save and Continue

Lab Worksheet

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

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ

The screenshot shows the Jenkins dashboard at localhost:8080. The top navigation bar includes links for 'Dashboard', 'Search (CTRL+K)', 'Chitsutha Soomlek', and 'log out'. The main content area features a 'Welcome to Jenkins!' message and a 'Start building your software project' section. It includes links for 'Create a job', 'Set up a distributed build', 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'. On the left, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (0/2). A sidebar on the right has an 'Add description' button.

9. เลือก Manage Jenkins และไปที่เมนู Plugins

The screenshot shows the 'Manage Jenkins' page at localhost:8080/manage/. The top navigation bar includes links for 'Dashboard', 'Manage Jenkins', 'Search (CTRL+K)', 'Chitsutha Soomlek', and 'log out'. The main content area features a 'System Configuration' section with tabs for 'System', 'Tools', 'Plugins', and 'Nodes'. Below this are sections for 'Security', 'Status Information', and 'About Jenkins'. A prominent red banner at the top states 'It appears that your reverse proxy set up is broken.' with 'More Info' and 'Dismiss' buttons. The 'Manage Jenkins' tab is highlighted in the sidebar.

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) ไปไว้ใน Repository ของนักศึกษา จากนั้นตั้งค่าที่จำเป็นในหน้านี้ทั้งหมด ดังนี้

Description: Lab 8.5

GitHub project: กดเลือกแล้วใส่ Project URL เป็น repository ที่เก็บไฟล์ .robot (ดูขั้นตอนที่ 12)

Build Trigger: เลือกแบบ Build periodically แล้วกำหนดให้ build ทุก 15 นาที

Lab Worksheet

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

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พื้นที่กับตตอบคำถามต่อไปนี้

The screenshot shows a web browser window with the URL `localhost:8080` in the address bar. The page title is "Getting Started". The main content is titled "Instance Configuration". A "Jenkins URL:" field contains the value `http://localhost:8080/lab8`, which is highlighted with a blue border. Below the field, a descriptive text explains that the Jenkins URL is used for absolute links to various Jenkins resources, including email notifications, PR status updates, and build steps. It notes that the proposed default value is not saved yet and is generated from the current request if possible. At the bottom, there is a Jenkins version notice "Jenkins 2.479.3" and two buttons: "Not now" and "Save and Finish".

The screenshot shows the Jenkins configuration interface for a project named "Lab 8.5". The top navigation bar indicates the path: Dashboard > UAT > Configuration. The main section is titled "General". A blue toggle switch labeled "Enabled" is turned on. The "Description" field contains the text "Lab 8.5". Below the description, there is a "Plain text Preview" link. Under the "GitHub project" section, the "Project url" is set to "https://github.com/Phetchapa/Lab-8-Software-Deployment-653380144-8.git". There are several optional checkboxes: "Discard old builds", "GitHub project" (which is checked), "Project url", "This project is parameterized", "Throttle builds", and "Execute concurrent builds if necessary". Each checkbox has a question mark icon next to it. At the bottom of the configuration panel, there are two "Advanced" dropdown buttons.

Lab Worksheet

Build Triggers

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

Build after other projects are built ?

Build periodically ?

Schedule ?

H/15 * * * *

Would last have run at Tuesday, January 28, 2025 at 8:35:47 AM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 8:50:47 AM Coordinated Universal Time.

GitHub hook trigger for GITScm polling ?

Poll SCM ?

Build Steps

≡ Execute shell ? X

Command

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

```
robot UAT_Lab7_001.robot
```

Advanced ▾

Add build step ▾

Post-build Actions

Add post-build action ▾

Save Apply

Lab Worksheet

The screenshot shows a software interface for configuring post-build actions. The top navigation bar includes 'Dashboard', 'UAT', and 'Configuration'. The main section is titled '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)
Input field: .

Advanced Edited

Thresholds for build result

Yellow circle icon: 20.0

Blue circle icon: 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**

- (1) คำสั่งที่ใช้ในการ Execute ไฟล์ .robot ใน Build Steps คือ
robot UAT_Lab7_001.robot

Lab Worksheet

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

จะบุ๊ดเร็คทอรี่ที่เก็บไฟล์ผลการทดสอบโดย Robot framework ในรูป xml และ html -> ตั้งค่า Threshold เป็น % ของ การทดสอบที่ไม่ผ่านแล้วนับว่าซอฟต์แวร์มีปัญหา -> ตั้งค่า Threshold เป็น % ของการทดสอบที่ผ่านแล้วนับว่าซอฟต์แวร์มีอยู่ในสถานะที่สามารถนำไปใช้งานได้ (เช่น 20, 80)

13. กด Apply และ Save

14. ล้าง Build Now

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

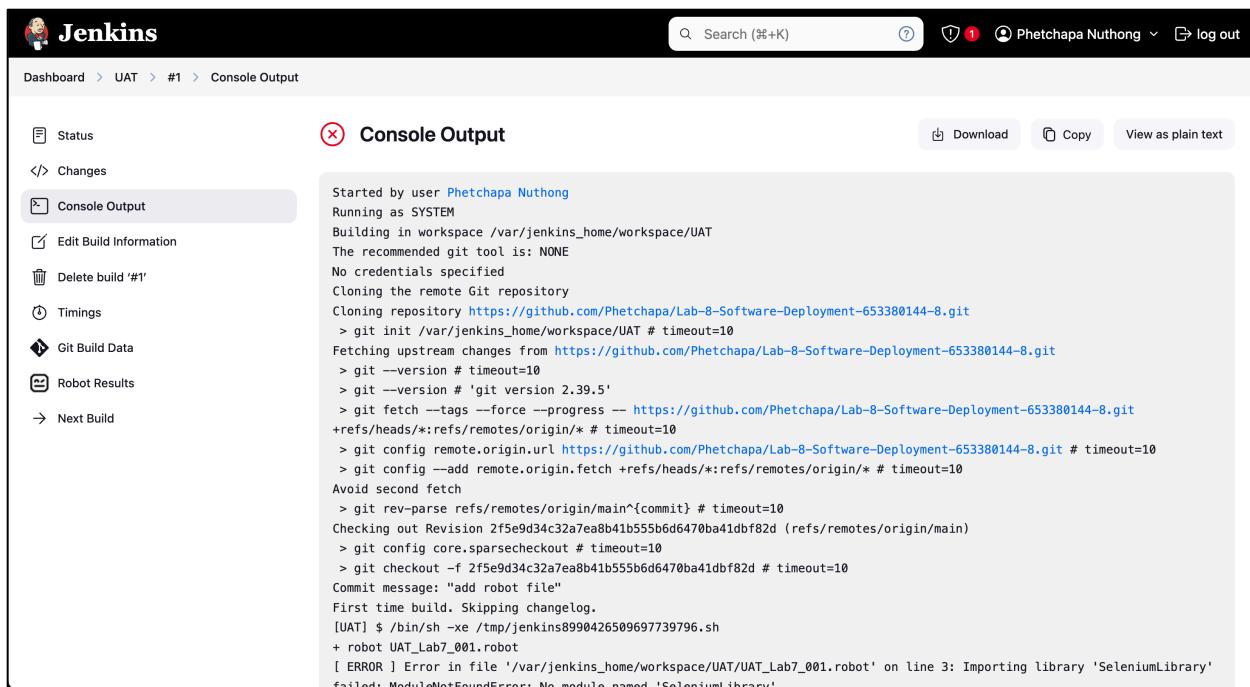
The screenshot shows the Jenkins Pipeline interface for build #1. The build status is red, indicating failure. The build was started by user Phetchapa Nuthong on Jan 28, 2025, at 11:30:58 AM. It took 1.3 seconds. The run spent details show waiting, build duration, and total time from scheduled to completion. The Git revision is 2f5e9d34c32a7ea8b41b555b6d6470ba41dbf82d, and the repository is https://github.com/Phetchapa/Lab-8-Software-Deployment-653380144-8.git. The Robot Test Summary shows 2 failed tests. The console output indicates no changes.

Total	Failed	Passed	Skipped	Pass %
All tests	2	0	0	0.0

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

</> No changes.

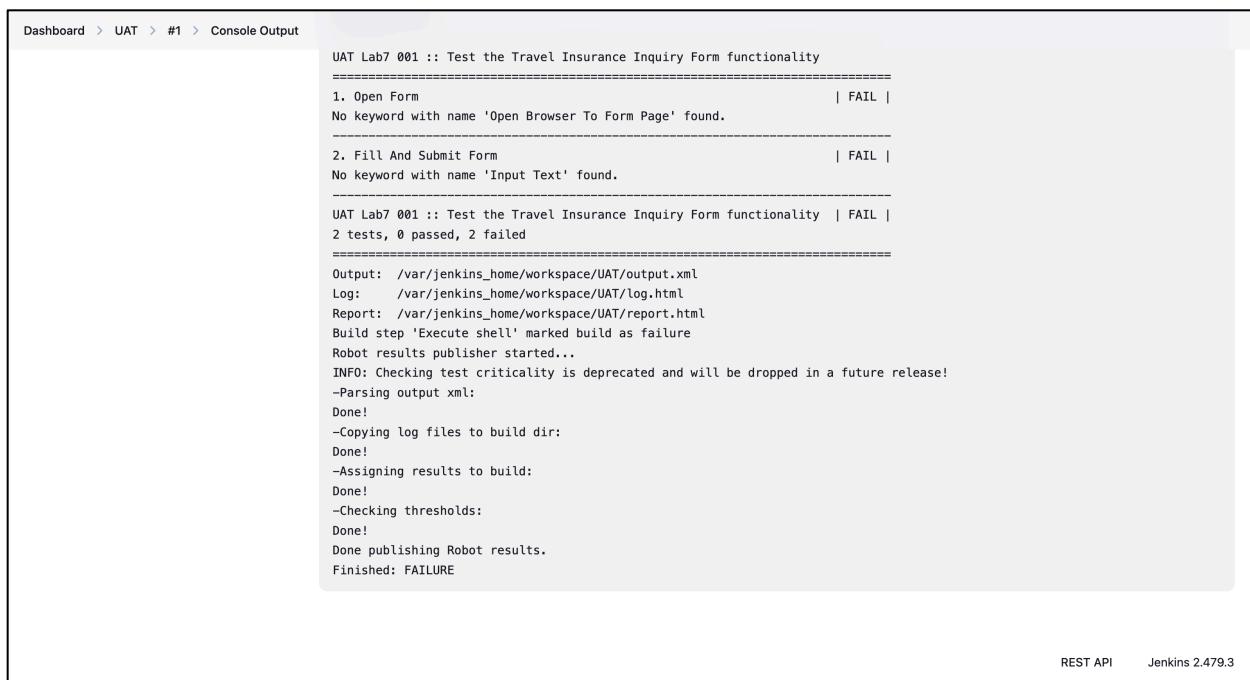
Lab Worksheet



The screenshot shows the Jenkins interface for a build named 'UAT #1'. The left sidebar has links for Status, Changes, Console Output (which is selected), Edit Build Information, Delete build '#1', Timings, Git Build Data, Robot Results, and Next Build. The main area is titled 'Console Output' and displays the following log output:

```

Started by user Phetchapa Nuthong
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Phetchapa/Lab-8-Software-Deployment-653380144-8.git
> git init /var/jenkins_home/workspace/UAT # timeout=10
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/Phetchapa/Lab-8-Software-Deployment-653380144-8.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/Phetchapa/Lab-8-Software-Deployment-653380144-8.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision 2f5e9d34c32a7ea8b41b555b6d6470ba41dbf82d (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f 2f5e9d34c32a7ea8b41b555b6d6470ba41dbf82d # timeout=10
Commit message: "add robot file"
First time build. Skipping changelog.
[UAT] $ /bin/sh -xe /tmp/jenkins8990426509697739796.sh
+ robot UAT_Lab7_001.robot
[ ERROR ] Error in file '/var/jenkins_home/workspace/UAT/UAT_Lab7_001.robot' on line 3: Importing library 'SeleniumLibrary'
failed: ModuleNotFoundError: No module named 'SeleniumLibrary'
```



The screenshot shows the Jenkins interface for a build named 'UAT #1'. The left sidebar has links for Status, Changes, Console Output (selected), Edit Build Information, Delete build '#1', Timings, Git Build Data, Robot Results, and Next Build. The main area is titled 'Console Output' and displays the following log output:

```

Dashboard > UAT > #1 > Console Output

UAT Lab7_001 :: Test the Travel Insurance Inquiry Form functionality
=====
1. Open Form | FAIL |
No keyword with name 'Open Browser To Form Page' found.
=====
2. Fill And Submit Form | FAIL |
No keyword with name 'Input Text' found.
=====
UAT Lab7_001 :: Test the Travel Insurance Inquiry Form functionality | FAIL |
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:
Done!
-Assigning results to build:
Done!
-Checking thresholds:
Done!
Done publishing Robot results.
Finished: FAILURE
```