

Lab Worksheet

ชื่อ-นามสกุล กัมแพงเพชร สิงห์ขจรณ์ รหัสนักศึกษา 653380120-2 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

The screenshot shows the Docker Desktop 'Images' tab. It displays a search bar, a filter icon, and a list of local images. Below the list, there is a 'Terminal' window showing the execution of several Docker commands. The terminal output includes the command to pull the 'busybox' image, the command to list all images, and the command to run a container from the 'busybox' image. The terminal also shows the output of the 'docker images' command, which lists various images including 'base-hadoop', 'mongo', 'postgres', 'image/pgadmin4', 'mongo-express', 'docker/welcome-to-docker', 'maildev/maildev', 'jupyter/pyspark-notebook', and 'bitnami/spark'.

```

PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
Digest: sha256:a5d8ce49aa801d475da48f8cb163c354ab95cab873cd3c138bd458fc8257fbf1
Status: Image is up to date for busybox:latest
docker.io/library/busybox:latest

What's next:
View a summary of image vulnerabilities and recommendations -> docker scout quickview busybox
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
base-hadoop          1.0                e83b77bab5cd       8 weeks ago        2.34GB
<none>               <none>             fd66981134e7       8 weeks ago        2.34GB
<none>               <none>             96694baf08e0       8 weeks ago        2.34GB
<none>               <none>             22d285336226       8 weeks ago        2.34GB
<none>               <none>             be07c526dcca       8 weeks ago        2.34GB
mongo                latest             77c59b638412       3 months ago       855MB
busybox              latest             af4709625109       4 months ago       4.27MB
postgres             latest             d57ed788c154       4 months ago       434MB
image/pgadmin4       latest             a67d330eef3b       4 months ago       483MB
mongo-express        latest             870141b735e7       10 months ago      182MB
docker/welcome-to-docker latest             c1f619b6477e       14 months ago      18.6MB
maildev/maildev      latest             8a928dba53d6       18 months ago      199MB
jupyter/pyspark-notebook spark-3.3.1        c1533ed981a4       23 months ago      6.38GB
bitnami/spark        3.3.1             19d61c534ed1       23 months ago      1.24GB
  
```

(1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร ชื่อ image

(2) Tag ที่ใช้บ่งบอกถึงอะไร version of docker 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 6-11 พร้อมกับตอบคำถามต่อไปนี้

```

PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker run busybox
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker run -it busybox
/# ls
/# ls -la
total 48
drwxr-xr-x 1 root root      4096 Jan 27 13:59 .
drwxr-xr-x 1 root root      4096 Jan 27 13:59 ..
-rwxr-xr-x 1 root root         0 Jan 27 13:59 .dockerenv
drwxr-xr-x 2 root root    12288 Sep 26 21:31 bin
drwxr-xr-x 5 root root      360 Jan 27 13:59 dev
drwxr-xr-x 1 root root      4096 Jan 27 13:59 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 240 root root         0 Jan 27 13:59 proc
drwx----- 1 root root      4096 Jan 27 13:59 root
dr-xr-xr-x 11 root root         0 Jan 27 13:59 sys
drwxrwxrwt 2 root root         0 Jan 27 13:59 tmp
drwxr-xr-x 4 root root         0 Jan 27 13:59 usr
drwxr-xr-x 4 root root         0 Jan 27 13:59 var
/# exit
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker run busybox echo "Hello Kamphaengphet Singkhon"
Hello Kamphaengphet Singkhon
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker ps -a
CONTAINER ID   IMAGE      NAMES                COMMAND                CREATED        STATUS        PORTS
0996beb97ec2   busybox    focused_keller        "echo 'Hello Kamphae..." 22 seconds ago Exited (0) 20 seconds ago
279f2d992774   busybox    affectionate_carver    "sh"                    About a minute ago Exited (0) About a minute ago
476d4db19512   busybox    sad_wright            "sh"                    About a minute ago Exited (0) About a minute ago
d64e4c03604a   busybox    funny_panini          "echo 'Hello Kamphae..." 4 minutes ago Exited (0) 4 minutes ago
7913ae095ede   busybox    nostalgic_snyder      "sh"                    9 minutes ago Exited (0) 8 minutes ago
877fa93b6d71   busybox    determined_noyce      "sh"                    9 minutes ago Exited (0) 9 minutes ago
f4951e1ff951   busybox    zen_grothendieck      "sh"                    14 minutes ago Exited (0) 14 minutes ago
  
```

- (1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
ช่วยให้สามารถเข้าสู่ Terminal ของคอนเทนเนอร์ ทำให้สามารถใช้คำสั่ง shell ได้
- (2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร
เวลาที่คอนเทนเนอร์หยุดทำงาน

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

Lab Worksheet

[Check point#3] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้ในขั้นตอนที่ 12

```

PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker run busybox echo "Hello Kamphaengphet Singkhon"
Hello Kamphaengphet Singkhon
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker ps -a
CONTAINER ID   IMAGE      NAMES                COMMAND                  CREATED          STATUS          PORTS
0996beb97ec2   busybox    focused_keller        "echo 'Hello Kamphae..." 22 seconds ago   Exited (0) 20 seconds ago
279f2d992774   busybox    affectionate_carver    "sh"                      About a minute ago Exited (0) About a minute ago
476d4db19512   busybox    sad_wright            "sh"                      About a minute ago Exited (0) About a minute ago
d64e4c03604a   busybox    funny_panini          "echo 'Hello Kamphae..." 4 minutes ago    Exited (0) 4 minutes ago
7913ae095ede   busybox    nostalgic_snyder      "sh"                      9 minutes ago    Exited (0) 8 minutes ago
877fa93b6d71   busybox    determined_noyce      "sh"                      9 minutes ago    Exited (0) 9 minutes ago
f4951e1ff951   busybox    zen_grothendieck      "sh"                      14 minutes ago   Exited (0) 14 minutes ago
cf53e7a97727   busybox    flamboyant_leakey     "sh"                      16 minutes ago   Exited (0) 16 minutes ago
6a084f498ecb   bitnami/spark:3.3.1   lab6-spark-1          "/opt/bitnami/script..." 3 weeks ago      Exited (255) 5 days ago    0.0.0.0:7077->7077/tcp,
0.0.0.0:8880->8880/tcp
df668d39576a   jupyter/pyspark-notebook:spark-3.3.1   lab6-jupyter-1        "tini -g -- start-no..." 3 weeks ago      Exited (255) 5 days ago    4040/tcp, 0.0.0.0:8889->
8889/tcp
d27e88345043   bitnami/spark:3.3.1   lab6-spark-worker-1   "/opt/bitnami/script..." 3 weeks ago      Exited (255) 5 days ago    0.0.0.0:8081->8081/tcp
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1>
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1> docker rm d64e4c03604a
d64e4c03604a
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_1>

```

แบบฝึกปฏิบัติที่ 8.2: สร้าง Docker file และ Docker image

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_2
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_2 เพื่อใช้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ (Windows) บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

FROM busybox

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

CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา ชื่อเล่น"

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal และบันทึกคำสั่งต่อไปนี้

\$ cat > Dockerfile << EOF

FROM busybox

Lab Worksheet

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

CMD echo "ชื่อ-นามสกุล รหัสนักศึกษา ชื่อเล่น"

EOF

หรือใช้คำสั่ง

\$ touch Dockerfile

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้

\$ docker build -t <ชื่อ Image> .

6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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

```

PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_2> docker build -f Dockerfile.swp -t my-image .
[+] Building 0.2s (5/5) FINISHED
=> [internal] load build definition from Dockerfile.swp
=> => transferring dockerfile: 160B
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
=> [internal] load metadata for docker.io/library/busybox:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/1] FROM docker.io/library/busybox:latest
=> exporting to image
=> => exporting layers
=> => writing image sha256:a37c32e884d74712dd0705ed4f2197348cf4d61a0ca1193ece12e445872551be
=> => naming to docker.io/library/my-image

View build details: docker-desktop://dashboard/build/default/default/gstjuc7dihp6u8do3d2ftc1zo

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
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_2> docker images
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
base-hadoop         1.0        e83b77bab5cd  8 weeks ago  2.34GB
<none>              <none>     fd66981134e7  8 weeks ago  2.34GB
<none>              <none>     96694baf08e0  8 weeks ago  2.34GB
<none>              <none>     22d205336226  8 weeks ago  2.34GB
<none>              <none>     be07c526dcca  8 weeks ago  2.34GB
mongo               latest     77c59b638412  3 months ago  859MB
my-image            latest     a37c32e884d7  4 months ago  4.27MB
busybox             latest     af4709625109  4 months ago  4.27MB

PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_2> docker run my-image
Kamphaengphet Singkhon 653380120-2 Andrew
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_2>

```

(1) คำสั่งที่ใช้ในการ run คือ `docker run my-image`

Lab Worksheet

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

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
2. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_3
3. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_3 เพื่อใช้เป็น Working directory
4. สร้าง Dockerfile.swp ไว้ใน Working directory

สำหรับเครื่องที่ใช้ระบบปฏิบัติการวินโดวส์ บันทึกคำสั่งต่อไปนี้ลงในไฟล์ โดยใช้ Text Editor ที่มี

```
FROM busybox
```

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

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

สำหรับเครื่องที่ใช้ระบบปฏิบัติการ MacOS หรือ Linux บนหน้าต่าง Terminal 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

Lab Worksheet

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

```
Start a build
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_3> docker build -f Dockerfile.swp -t kamphaengphet/lab8 .
[+] Building 0.2s (5/5) FINISHED
=> [internal] load build definition from Dockerfile.swp
=> => transferring dockerfile: 182B
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)
=> [internal] load metadata for docker.io/library/busybox:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> CACHED [1/1] FROM docker.io/library/busybox:latest
=> exporting to image
=> => exporting layers
=> => writing image sha256:0574f724b18f6b2c5bab59a40c3a6f5c7d93cfff37db8056af94cb77930ab1b4
=> => naming to docker.io/kamphaengphet/lab8

View build details: docker-desktop://dashboard/build/default/default/u8noh42n7tyw34i8vqmmtf9xw

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

What's next:
View a summary of image vulnerabilities and recommendations → docker scout quickview
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_3> docker run kamphaengphet/lab8
Kamphaengphet Singkhon 653380120-2
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_3> |
```

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง

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

ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

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

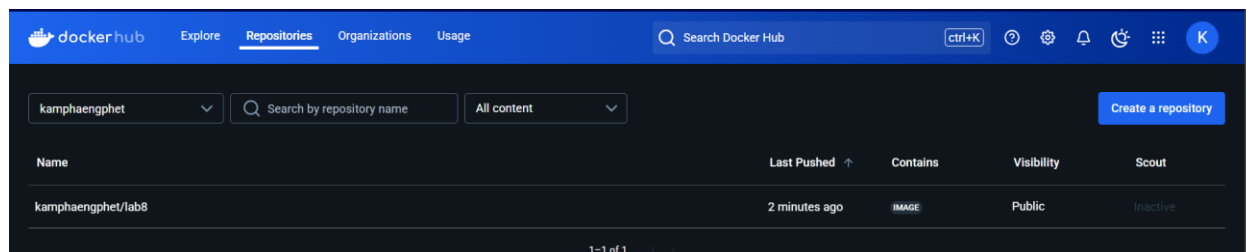
```
$ docker login -u <username> -p <password>
```

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

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

```
Windows PowerShell
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment> cd Lab8_3
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_3> docker push kamphaengphet/lab8
Using default tag: latest
The push refers to repository [docker.io/kamphaengphet/lab8]
59654b79daad: Mounted from library/busybox
latest: digest: sha256:1f7689b06b436e2bcb465a9c716e14e4a036e35c095ffea33c9d6341abb987 size: 527
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_3> |
```

Lab Worksheet



แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

1. เปิด Command line หรือ Terminal จากนั้นสร้าง Directory ชื่อ Lab8_4
2. ทำการ Clone ซอร์สโค้ดของเว็บแอปพลิเคชันจาก GitHub repository
<https://github.com/docker/getting-started.git> ลงใน Directory ที่สร้างขึ้น โดยใช้คำสั่ง
`$ git clone https://github.com/docker/getting-started.git`
3. เปิดดูองค์ประกอบภายใน getting-started/app เมื่อพบไฟล์ package.json ให้ใช้ Text editor ในการเปิดอ่าน

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

```
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment> cd Lab8_4
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\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 | 13.90 MiB/s, done.
Resolving deltas: 100% (523/523), done.
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4>
```


Lab Worksheet

**/*.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"}}. The line numbers 1 through 34 are visible on the left side of the code editor."/>

```

1  {
2    "name": "101-app",
3    "version": "1.0.0",
4    "main": "index.js",
5    "license": "MIT",
6    "scripts": {
7      "prettify": "prettier -l --write \"**/*.js\"",
8      "test": "jest",
9      "dev": "nodemon src/index.js"
10   },
11   "dependencies": {
12     "express": "^4.18.2",
13     "mysql2": "^2.3.3",
14     "sqlite3": "^5.1.2",
15     "uuid": "^9.0.0",
16     "wait-port": "^1.0.4"
17   },
18   "resolutions": {
19     "ansi-regex": "5.0.1"
20   },
21   "prettier": {
22     "trailingComma": "all",
23     "tabWidth": 4,
24     "useTabs": false,
25     "semi": true,
26     "singleQuote": true
27   },
28   "devDependencies": {
29     "jest": "^29.3.1",
30     "nodemon": "^2.0.20",
31     "prettier": "^2.7.1"
32   }
33 }
34

```

4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไปไฟล์

FROM node:18-alpine

WORKDIR /app

COPY . .

RUN yarn install --production

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

Lab Worksheet

EXPOSE 3000

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

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

[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ

```
View build details: docker-desktop://dashboard/build/default/default/5gnlyo8ms70nddxo2aynn8a0
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker build -f Dockerfile.swp -t myapp_6533801202 .
[+] Building 21.0s (18/10) FINISHED
=> [internal] load build definition from Dockerfile.swp
=> => transferring dockerfile: 158B
=> [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@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25
=> resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25
=> sha256:dc5f7b337959b664d214e4ed84f230eeef0e4dc03a50388d573e289b9e5e40 6.18kB / 6.18kB
=> sha256:1f3e46996e2966e4faa5846e56e76e3748b7215e2ded61476c24403d592134f0 3.64MB / 3.64MB
=> sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 40.01MB / 40.01MB
=> sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 1.26MB / 1.26MB
=> sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 7.67kB / 7.67kB
=> sha256:6e804119c3884fc5782795bf0d2adc89201c63105aece8647b17a7bcebb385e 1.72kB / 1.72kB
=> sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 444B / 444B
=> extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0
=> extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c
=> extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1
=> extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771
=> [internal] load build context
=> => transferring context: 4.62MB
=> [2/4] WORKDIR /app
=> [3/4] COPY . .
=> [4/4] RUN yarn install --production
=> exporting image
=> writing image sha256:668b39637c1505960237ebff067ec1750ab6357468abf57ce29e9b78e69c67bd
=> naming to docker.io/library/myapp_6533801202
View build details: docker-desktop://dashboard/build/default/default/7mf2h5rowlv2ojgymborztcw

What's next:
View a summary of image vulnerabilities and recommendations + docker scout quickview
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> |
```

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

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

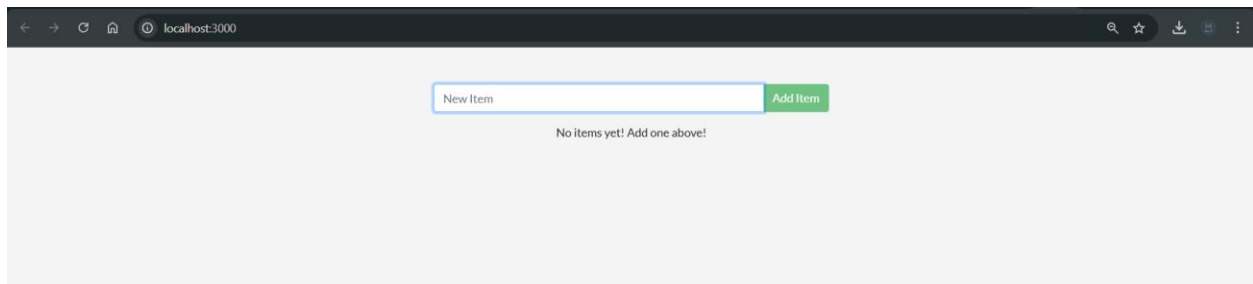
7. เปิด Browser ไปที่ URL = <http://localhost:3000>

[Check point#9] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้บน Browser

และ Dashboard ของ Docker desktop

```
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801202
704634598cebbad629f2d495cc6d7d08fba56f472aebb38024918650c4117f
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> |
```

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

```
>4 | | | | <div id="form" onmouseover={onmouseover} />
55 | | | | {items.length === 0 && (
56 | | | |   <p className="text-center">There is no TODO item. Please add one to the list. By Kamphaengphet Singkhon</p>
57 | | | | )}
```

```
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp-6533801202
12f23cc1b9b3f35b9c7f292799e85f2ead9442b3b6f1c41d3a9414263da0b61
docker: Error response from daemon: driver failed programming external connectivity on endpoint recurring_morse (3ca93c73571d3f8944446b1cef0bbb3dd07edce130c
938ea4f7c05814d3fbb6f): Bind for 0.0.0.0:3000 failed: port is already allocated.
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> |
```

(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

พอร์ต 3000 ของคอนเทนเนอร์ ถูกใช้ไปแล้ว

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

Lab Worksheet

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

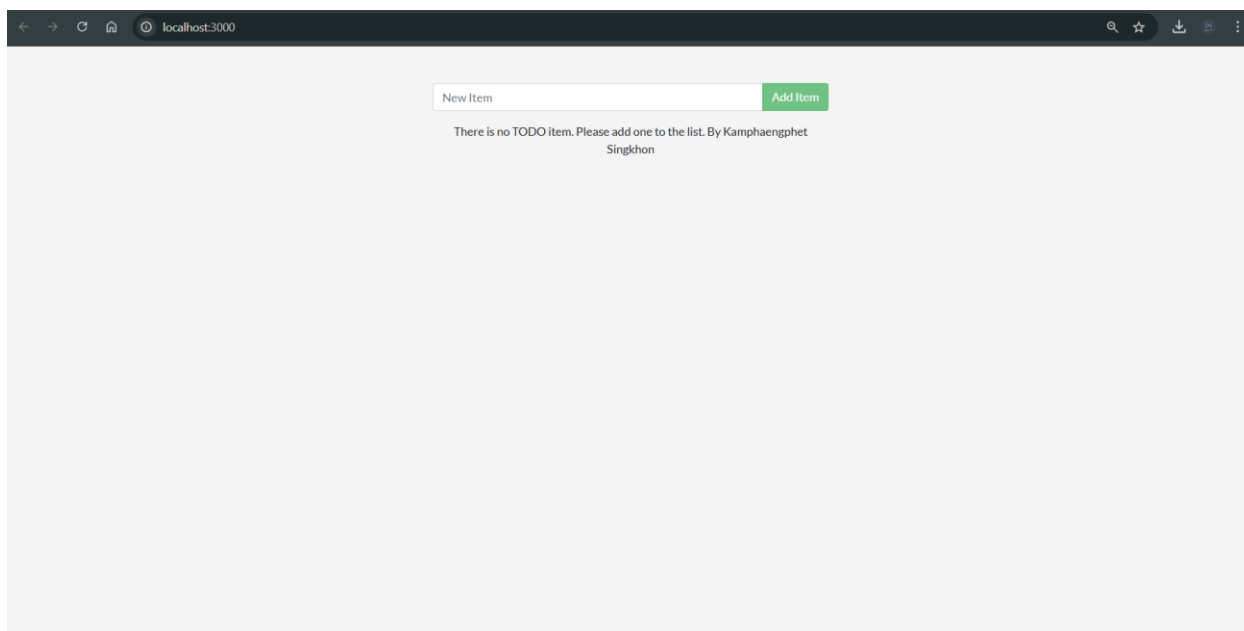
iii. ยืนยันโดยการกด Delete forever

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

```
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
704634598ceb   668b39637c15   "docker-entrypoint.s..." 46 minutes ago Up 46 minutes   0.0.0.0:3000->3000/tcp   crazy_nash
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker rm 704634598ceb
Error response from daemon: cannot remove container "/crazy_nash": container is running: stop the container before removing or force remove
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker rm -f 704634598ceb
704634598ceb
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533801202
2fe4bfe2d8f9ef40b7d499b86d85da497c25f870528cc792956c57f3b62b363
PS D:\University_Repository\Year3\Semester2\SoftwareEngineer\LabAssignment\Lab8_4\getting-started\app> |
```



Lab Worksheet

แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

1. เปิด Command line หรือ Terminal บน Docker Desktop

2. บ้อนคำสั่งและทำการรัน 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

```
*****
*****
*****

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

4406886759fc4d6e8ab991d3ba547a59

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*****
*****
*****
```

4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และบ้อนที่อยู่เป็น

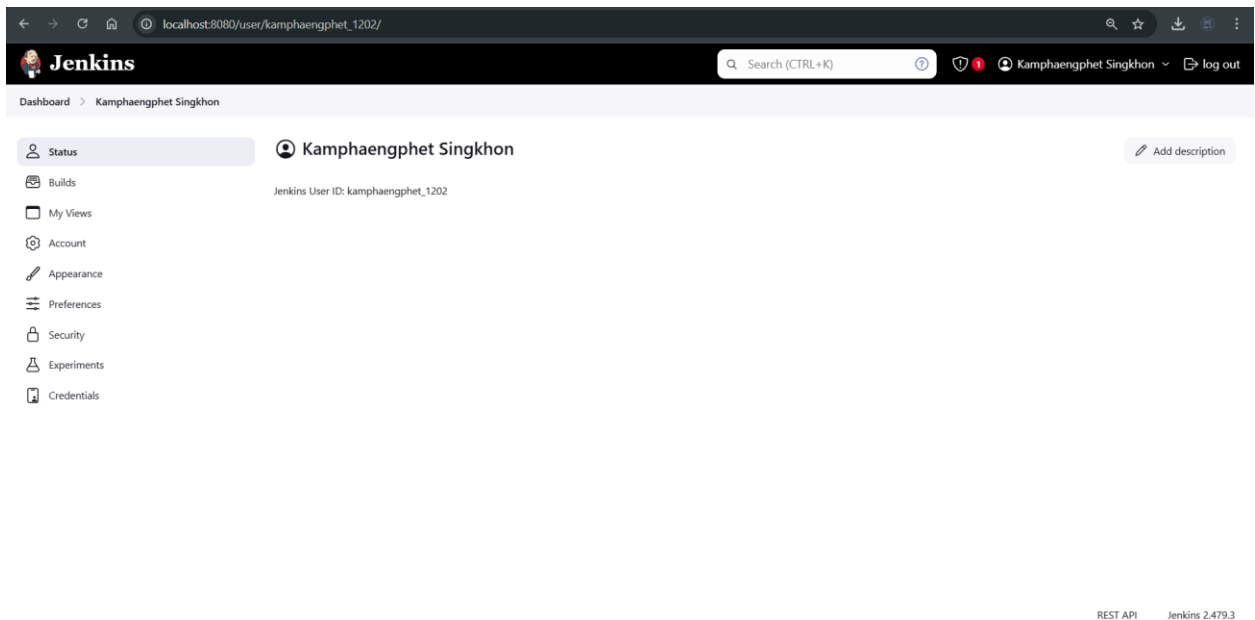
localhost:8080

5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3

6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

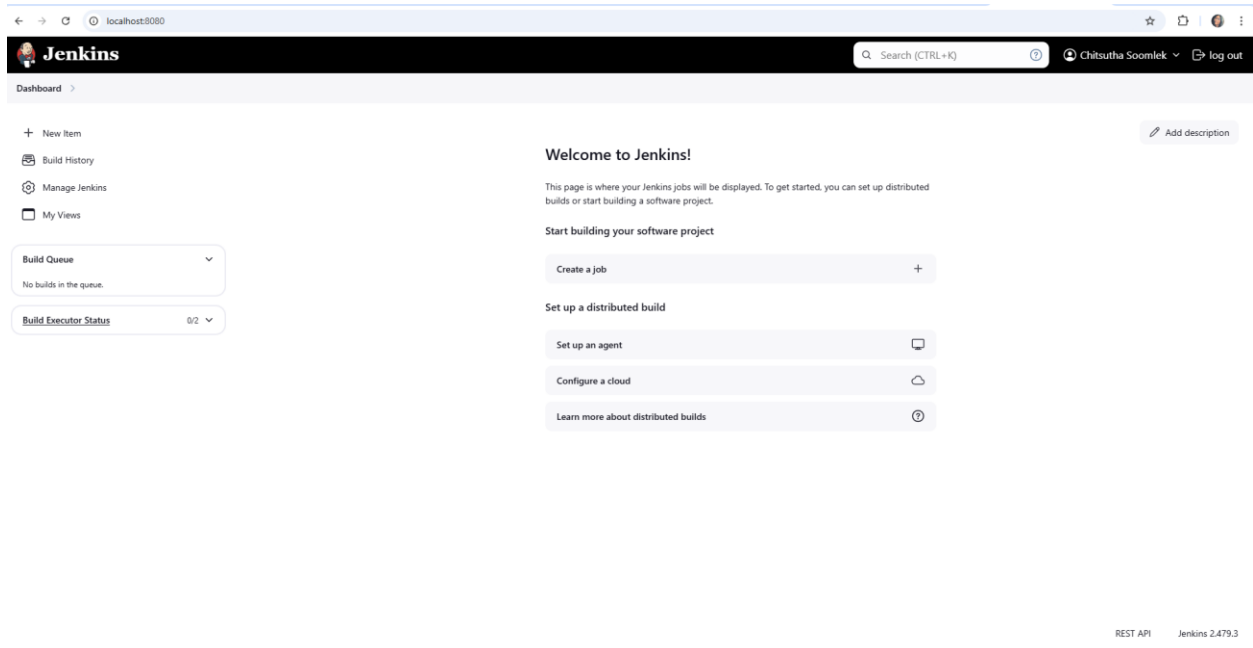
Lab Worksheet

[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า



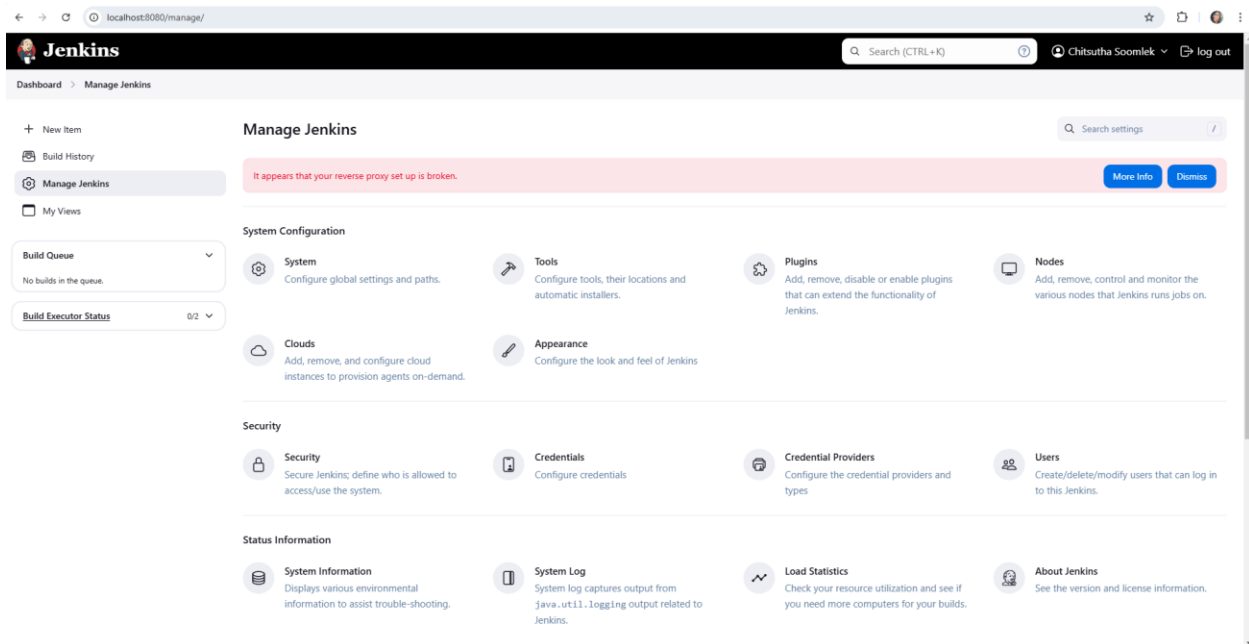
7. กำหนด Jenkins URL เป็น <http://localhost:8080/lab8>

8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

Lab Worksheet

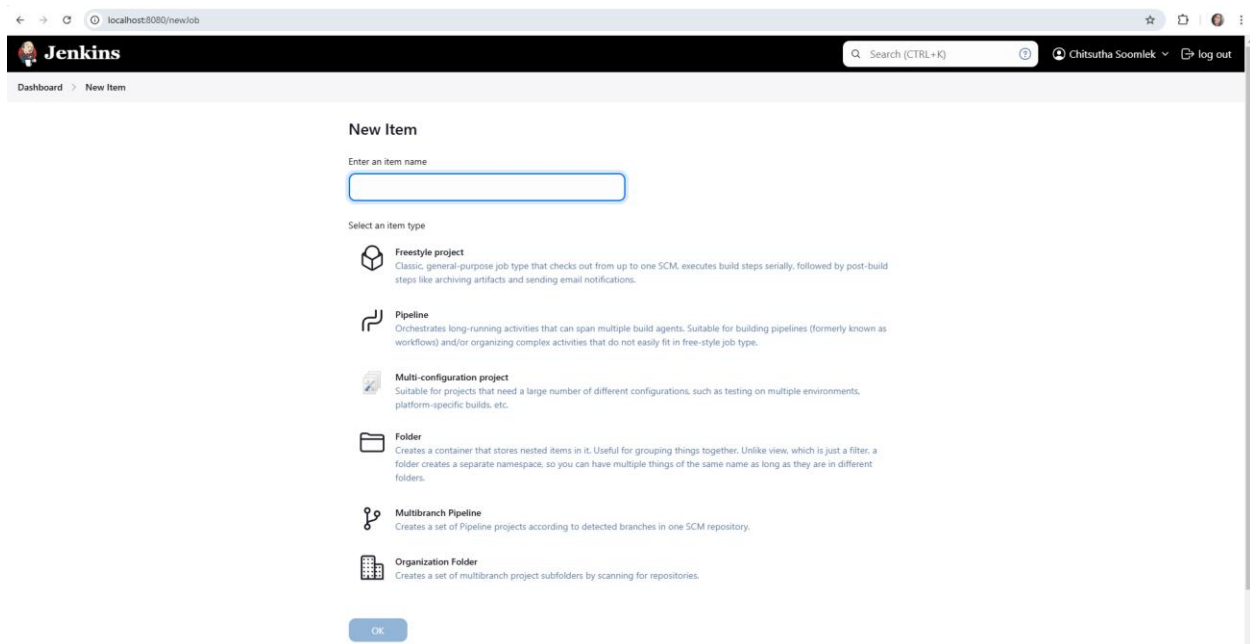


10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม



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

Lab Worksheet



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

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

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Lab Worksheet

Jenkins Search (CTRL+K) Kamphaengphet Singkhon log out

Dashboard > UAT > Configuration

Configure

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

General

Enabled ☒

Description

Lab 8.5

Plain text [Preview](#)

☐ Discard old builds ?

☒ GitHub project

Project url ?

<https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git/>

Advanced ▾

☐ This project is parameterized ?

☐ Throttle builds ?

☐ Force concurrent builds if necessary ?

[Save](#) [Apply](#)

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 Monday, January 27, 2025 at 7:50:44 PM Coordinated Universal Time; would next run at Monday, January 27, 2025 at 7:50:44 PM Coordinated Universal Time.

Build Steps

Execute shell ?

Command

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

```
robot -d results Lab7-001.robot
robot -d results Lab7-002.robot
```

Advanced ▾

Add build step ▾

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

`robot -d results Lab7-001.robot`

`robot -d results Lab7-002.robot`

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



The screenshot shows the Jenkins console output for the UAT pipeline. The 'Console Output' tab is selected, displaying the following text:

```

Started by user Kamphaengphet Singkhon
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/UAT
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git
> git init /var/jenkins_home/workspace/UAT # timeout=10
Fetching upstream changes from https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git
> git --version # timeout=10
> git --version # 'git version 2.39.5'
> git fetch --tags --force --progress -- https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.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/master^{commit} # timeout=10
Checking out Revision 6adfd621cb58d8a0989561fef2ceefc558b79262 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 6adfd621cb58d8a0989561fef2ceefc558b79262 # timeout=10
Commit message: "Initial commit"
First time build. Skipping changelog.
[UAT] $ /bin/sh -xe /tmp/jenkins1841057385332730349.sh
+ robot -d results Lab7-001.robot
  
```

Lab Worksheet

Started by user Kamphaengphet Singkhon

Running as SYSTEM

Building in workspace /var/jenkins_home/workspace/UAT

[WS-CLEANUP] Deleting project workspace...

[WS-CLEANUP] Deferred wipeout is used...

[WS-CLEANUP] Done

The recommended git tool is: NONE

No credentials specified

Cloning the remote Git repository

Cloning repository https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git

> git init /var/jenkins_home/workspace/UAT # timeout=10

Fetching upstream changes from https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git

> git --version # timeout=10

> git --version # 'git version 2.39.5'

> git fetch --tags --force --progress -- https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.git +refs/heads/*:refs/remotes/origin/* # timeout=10

> git config remote.origin.url https://github.com/Andrew7y/Software-Engineer-Lab-RobotTesting.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/master^{commit} # timeout=10

Checking out Revision 6adfd621cb58d8a0989561fef2ceefc558b79262 (refs/remotes/origin/master)

> git config core.sparsecheckout # timeout=10

> git checkout -f 6adfd621cb58d8a0989561fef2ceefc558b79262 # timeout=10

Commit message: "Initial commit"

First time build. Skipping changelog.

Lab Worksheet

```
[UAT] $ /bin/sh -xe /tmp/jenkins1841057385332730349.sh
```

```
+ robot -d results Lab7-001.robot
```

```
=====
=====
```

```
Lab7-001 :: this resource file contain test cases.
```

```
=====
=====
```

```
Test Case 1 | FAIL |
```

```
NoSuchDriverException: Message: Unable to obtain driver for chrome; For documentation on this
error, please visit:
```

```
https://www.selenium.dev/documentation/webdriver/troubleshooting/errors/driver\_location
```

```
-----
```

```
Test Case 2 | FAIL |
```

```
NoSuchDriverException: Message: Unable to obtain driver for chrome; For documentation on this
error, please visit:
```

```
https://www.selenium.dev/documentation/webdriver/troubleshooting/errors/driver\_location
```

```
-----
```

```
Lab7-001 :: this resource file contain test cases. | FAIL |
```

```
2 tests, 0 passed, 2 failed
```

```
=====
=====
```

```
Output: /var/jenkins_home/workspace/UAT/results/output.xml
```

```
Log: /var/jenkins_home/workspace/UAT/results/log.html
```

```
Report: /var/jenkins_home/workspace/UAT/results/report.html
```

```
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:
```

Lab Worksheet

Done!

-Copying log files to build dir:

Done!

-Assigning results to build:

Done!

-Checking thresholds:

Done!

Done publishing Robot results.

Finished: FAILURE