CP353004/SC313 004 Software Engineering (2/2567)
ผศ.ดร.ชิตสุธา สุ่มเล็ก
Lab Worksheet

ชื่อ-นามสกุล นางสาวกัลยกร พงษ์มณีศิลป์ รหัสนักศึกษา 653380262-2 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

CP353004/SC313 004 Software Engineering (2/2567)
ผศ.ดร.ชิตสุธา สุ่มเล็ก
Lab Worksheet

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) แสดงผลลัพธ์ที่ได้ พร้อมกับตอบคำถามต่อไปนี้

```
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows


    Directory: C:\Users\KKU650001\Downloads\Lab8


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        28/01/2025     15:58                Lab8_1
```

```
PS C:\Users\KKU650001\Downloads\Lab8> cd Lab8_1
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
9c0abc9c5bd3: Pull complete
Digest: sha256:a5d0ce49aa801d475da48f8cb163c354ab95cab073cd3c138bd458fc8257fbf1
Status: Downloaded newer image for busybox:latest
docker.io/library/busybox:latest
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker images
REPOSITORY                                  TAG          IMAGE ID       CREATED         SIZE
git-group-repository-group-3-sec-2-v-2-php  latest       28171b494d4c   4 weeks ago     1.3GB
mysql                                       8.0          6c55ddbef969   3 months ago    591MB
busybox                                     latest       af4709625109   4 months ago    4.27MB
```

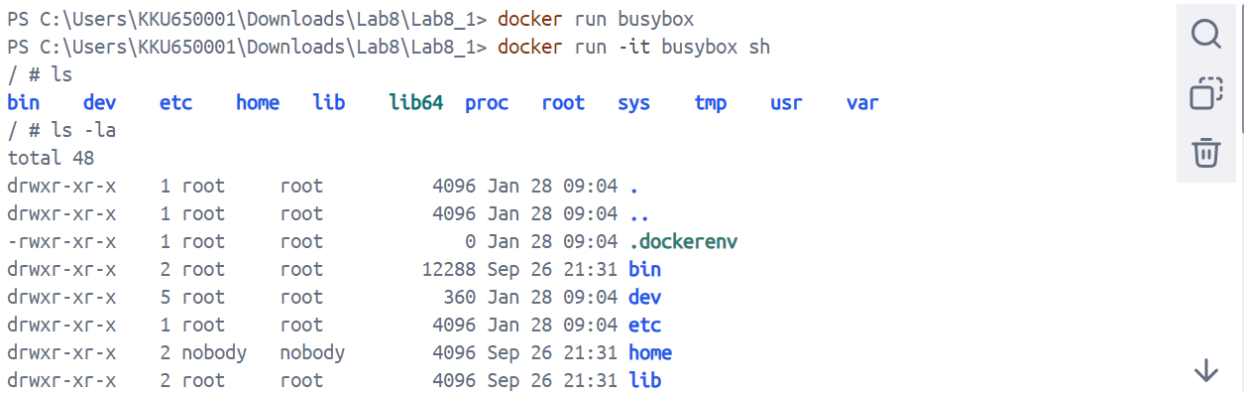
```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker images
REPOSITORY                                  TAG          IMAGE ID       CREATED         SIZE
git-group-repository-group-3-sec-2-v-2-php  latest       28171b494d4c   4 weeks ago     1.3GB
mysql                                       8.0          6c55ddbef969   3 months ago    591MB
busybox                                     latest       af4709625109   4 months ago    4.27MB
php                                         8.0-apache   3357132d6ece   14 months ago   455MB
phpmyadmin/phpmyadmin                       latest       933569f3a9f6   18 months ago   562MB
```

(1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร ชื่อ image

(2) Tag ที่ใช้บ่งบอกถึงอะไร Version ที่ใช้

5. ป้อนคำสั่ง $ docker run busybox
6. ป้อนคำสั่ง $ docker run -it busybox sh
7. ป้อนคำสั่ง ls
8. ป้อนคำสั่ง ls -la
9. ป้อนคำสั่ง exit
10. ป้อนคำสั่ง $ docker run busybox echo “Hello ชื่อและนามสกุลของนักศึกษา from busybox”
11. ป้อนคำสั่ง $ docker ps -a

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



```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker run busybox
PS C:\Users\KKU650001\Downloads\Lab8\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 28 09:04 .
drwxr-xr-x    1 root     root          4096 Jan 28 09:04 ..
-rwxr-xr-x    1 root     root             0 Jan 28 09:04 .dockerenv
drwxr-xr-x    2 root     root         12288 Sep 26 21:31 bin
drwxr-xr-x    5 root     root           360 Jan 28 09:04 dev
drwxr-xr-x    1 root     root          4096 Jan 28 09:04 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
```

```
drwxr-xr-x    5 root     root           360 Jan 28 09:04 dev
drwxr-xr-x    1 root     root          4096 Jan 28 09:04 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  247 root     root             0 Jan 28 09:04 proc
drwx------    1 root     root          4096 Jan 28 09:04 root
dr-xr-xr-x   11 root     root             0 Jan 28 09:04 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
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker run busybox echo “Hello kunyakon phongmaneesin from bu
sybox"
```

```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker run busybox echo “Hello kunyakon phongmaneesin from bu
sybox"
Hello kunyakon phongmaneesin from busybox
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker ps -a
CONTAINER ID   IMAGE                                          COMMAND                  CREATED
STATUS                          PORTS                          NAMES
95fee729f037   busybox                                        "echo 'Hello kunyako…"   15 seconds ago
Exited (0) 13 seconds ago                                     tender_spence
9f94e17b0e6a   busybox                                        "sh"                     About a minute ago
Exited (0) About a minute ago                                 quizzical_archimedes
230d802f45e0   busybox                                        "sh"                     About a minute ago
Exited (0) About a minute ago                                 stoic_bohr
468694485de7   git-group-repository-group-3-sec-2-v-2-php     "/bin/sh -c 'echo Co…"   4 weeks ago
Exited (0) 6 days ago                                         git-group-repository-group-3-sec-2-v-2
```

```
95fee729f037   busybox                                        "echo 'Hello kunyako…"   15 seconds ago
Exited (0) 13 seconds ago                                     tender_spence
9f94e17b0e6a   busybox                                        "sh"                     About a minute ago
Exited (0) About a minute ago                                 quizzical_archimedes
230d802f45e0   busybox                                        "sh"                     About a minute ago
Exited (0) About a minute ago                                 stoic_bohr
468694485de7   git-group-repository-group-3-sec-2-v-2-php     "/bin/sh -c 'echo Co…"   4 weeks ago
Exited (0) 6 days ago                                         git-group-repository-group-3-sec-2-v-2
-php-1
7d1df59856ae   phpmyadmin/phpmyadmin                          "/docker-entrypoint.…"   4 weeks ago
Up 12 minutes                   0.0.0.0:8090->80/tcp           phpmyadmin
4519f017f711   mysql:8.0                                      "docker-entrypoint.s…"   4 weeks ago
Up 12 minutes                   0.0.0.0:3306->3306/tcp, 33060/tcp   db
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1>
```

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

ทำให้ container ที่รันขึ้นมานั้นสามารถ interact ได้เหมือนกับการใช้งาน shell หรือ command line interface (CLI) โดยไม่ต้องปิด container

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไรจ๊ะสิ้นแล้ว

แสดงถึงสถานะปัจจุบันของแต่ละ container และexitedคือทำงานเสร็จ

12. ป้อนคำสั่ง $ docker rm <container ID ที่ต้องการลบ>

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

```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_1> docker rm 95fee729f037
95fee729f037
PS C:\Users\KKU650001\Downloads\Lab8\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 และป้อนคำสั่งต่อไปนี้

```
$ cat > Dockerfile << EOF
FROM busybox
CMD echo “Hi there. This is my first docker image.”
```

CMD echo “ชื่อ-นามสกุล รหัสนักศึกษา ชื่อเล่น”
EOF

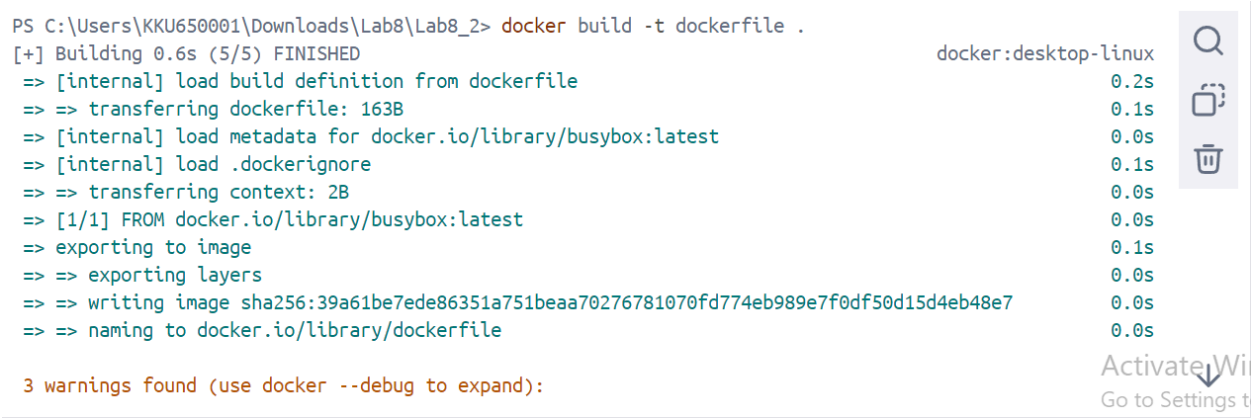
หรือใช้คำสั่ง

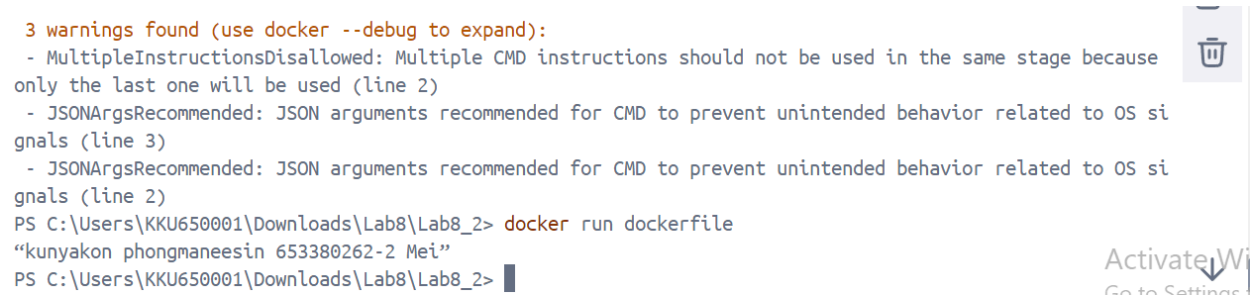
$ touch Dockerfile

แล้วใช้ Text Editor ในการใส่เนื้อหาแทน

5. ทำการ Build Docker image ที่สร้างขึ้นด้วยคำสั่งต่อไปนี้
   $ docker build -t <ชื่อ Image> .
6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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





(1) คำสั่งที่ใช้ในการ run คือ

docker run dockerfile

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

ใช้สำหรับตั้งชื่อ (tag) ให้กับ Docker image ที่สร้างขึ้นจาก Dockerfile

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 บันทึกคำสั่งต่อไปนี้ลงในไฟล์โดยใช้ 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

```
PS C:\Users\KKU650001\Downloads\Lab8> cd Lab8_3
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_3> code .
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_3> docker build -t kunyakon/lab8 .
[+] Building 0.3s (5/5) FINISHED                                  docker:desktop-linux
 => [internal] load build definition from dockerfile                              0.1s
 => => transferring dockerfile: 186B                                              0.0s
 => [internal] load metadata for docker.io/library/busybox:latest                 0.0s
 => [internal] load .dockerignore                                                 0.0s
 => => transferring context: 2B                                                   0.0s
 => CACHED [1/1] FROM docker.io/library/busybox:latest                            0.0s
 => exporting to image                                                            0.0s
 => => exporting layers                                                           0.0s
 => => writing image sha256:a139cab72893ac57a20f449ef382afdf50121de1949b4ed8938ee30f5106301d  0.0s
 => => naming to docker.io/kunyakon/lab8                                          0.0s

 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)
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_3> docker run kunyakon/lab8
“kunyakon phongmaneesin 653380262-2”
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_3>
```

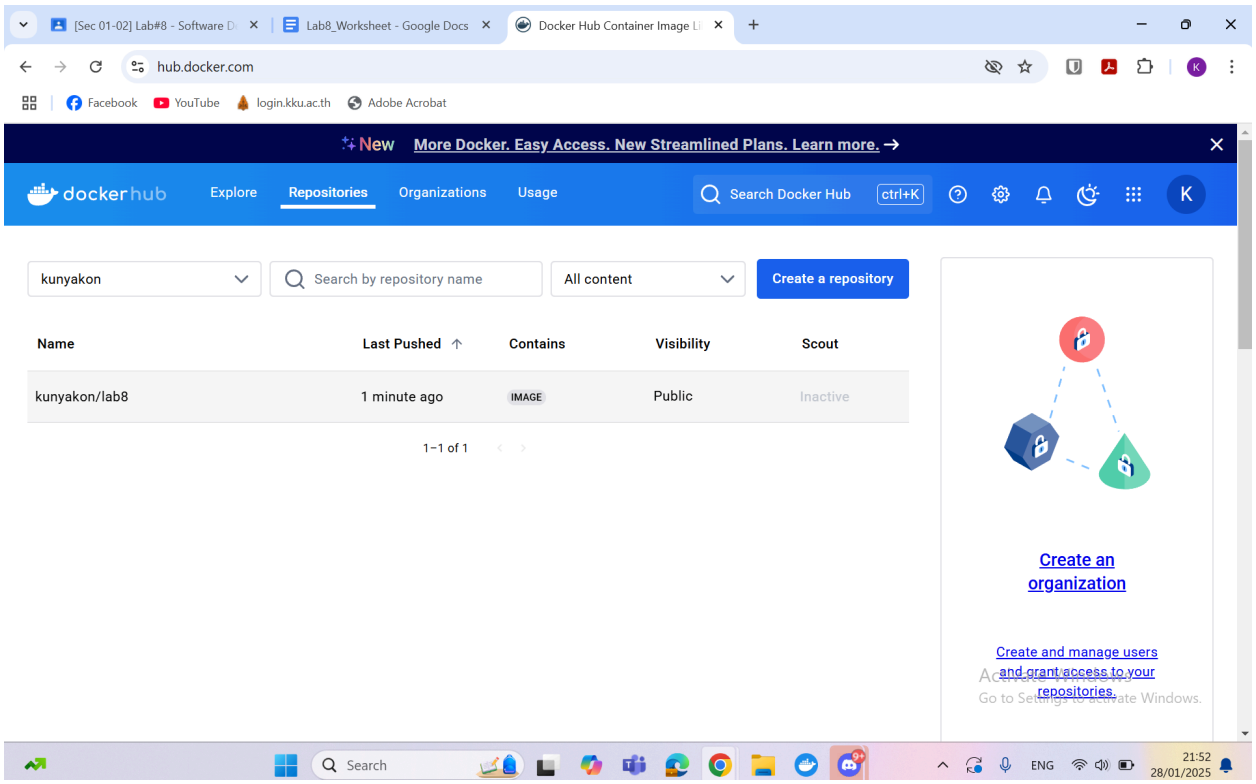
6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง
   $ docker push <username ที่ลงทะเบียนกับ Docker Hub>/lab8

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)

```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_3> docker push kunyakon/lab8
Using default tag: latest
The push refers to repository [docker.io/kunyakon/lab8]
59654b79daad: Mounted from library/busybox
latest: digest: sha256:0b6bdcce6d9d5aa3dcdf188cf5f00bbc756924c737a7e8e22609b235009a7091 size: 527
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_3>
```



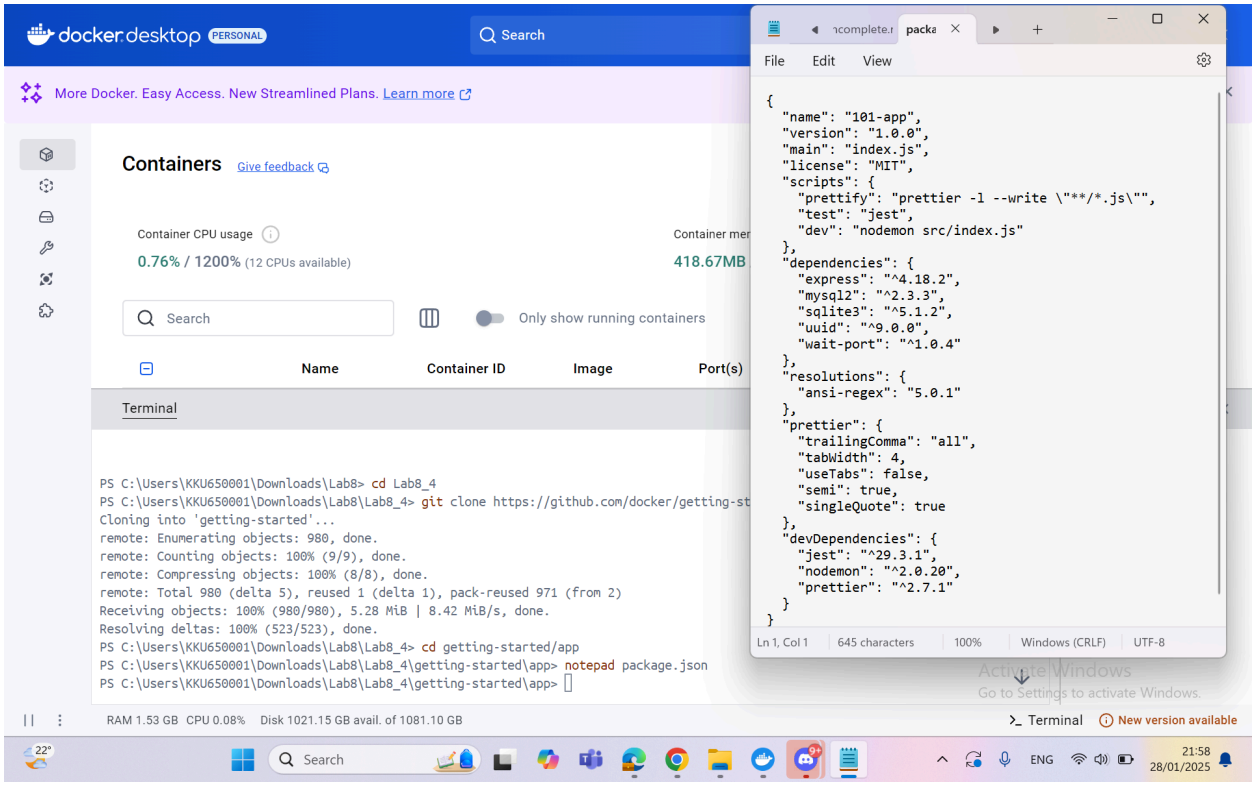
CP353004/SC313 004 Software Engineering (2/2567)
ผศ.ดร.ชิตสุธา สุ่มเล็ก
Lab Worksheet

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



4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไปในไฟล์
   FROM node:18-alpine
   WORKDIR /app
   COPY . .
   RUN yarn install --production
   CMD ["node", "src/index.js"]
   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

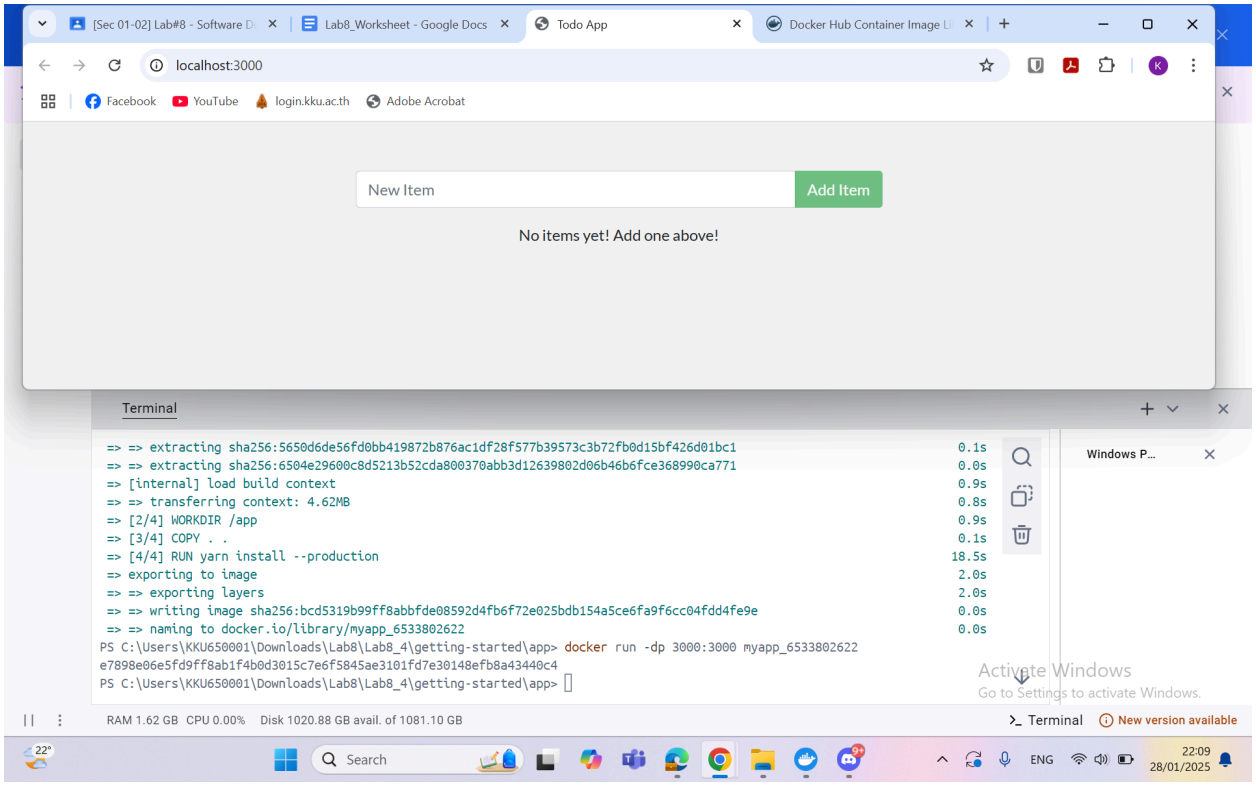
```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533802622 .
[+] Building 35.7s (10/10) FINISHED                                                                    docker:desktop-linux
 => [internal] load build definition from dockerfile                                                                0.1s
 => => transferring dockerfile: 156B                                                                                0.0s
 => [internal] load metadata for docker.io/library/node:18-alpine                                                   5.0s
 => [auth] library/node:pull token for registry-1.docker.io                                                         0.0s
 => [internal] load .dockerignore                                                                                   0.0s
 => => transferring context: 2B                                                                                     0.0s
 => [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb2  8.9s
 => => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb2  0.0s
 => => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 3.64MB / 3.64MB                      2.8s
 => => sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 40.01MB / 40.01MB                    4.3s
 => => sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 1.26MB / 1.26MB                      2.8s
 => => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 7.67kB / 7.67kB                      0.0s
```

```
 => [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb2  8.9s
 => => resolve docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb2  0.0s
 => => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 3.64MB / 3.64MB                      2.8s
 => => sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c 40.01MB / 40.01MB                    4.3s
 => => sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1 1.26MB / 1.26MB                      2.8s
 => => sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb25 7.67kB / 7.67kB                      0.0s
 => => sha256:6e804119c3884fc5782795bf0d2adc89201c63105aece8647b17a7bcebbc385e 1.72kB / 1.72kB                      0.0s
 => => sha256:dcbf7b337595be6f4d214e4eed84f230eefe0e4ac03a50380d573e289b9e5e40 6.18kB / 6.18kB                      0.0s
 => => extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0                           0.9s
 => => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 444B / 444B                          3.5s
 => => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c                           3.8s
 => => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1                           0.1s
 => => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771                           0.0s
 => [internal] load build context                                                                                   0.9s
```

```
 => => sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771 444B / 444B                          3.5s
 => => extracting sha256:37892ffbfcaa871a10f813803949d18c3015a482051d51b7e0da02525e63167c                           3.8s
 => => extracting sha256:5650d6de56fd0bb419872b876ac1df28f577b39573c3b72fb0d15bf426d01bc1                           0.1s
 => => extracting sha256:6504e29600c8d5213b52cda800370abb3d12639802d06b46b6fce368990ca771                           0.0s
 => [internal] load build context                                                                                   0.9s
 => => transferring context: 4.62MB                                                                                 0.8s
 => [2/4] WORKDIR /app                                                                                              0.9s
 => [3/4] COPY . .                                                                                                  0.1s
 => [4/4] RUN yarn install --production                                                                            18.5s
 => exporting to image                                                                                              2.0s
 => => exporting layers                                                                                             2.0s
 => => writing image sha256:bcd5319b99ff8abbfde08592d4fb6f72e025bdb154a5ce6fa9f6cc04fdd4fe9e                        0.0s
 => => naming to docker.io/library/myapp_6533802622                                                                 0.0s
PS C:\Users\KKU650001\Downloads\Lab8\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



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
   No items yet! Add one above! เป็น
   There is no TODO item. Please add one to the list. By ชื่อและนามสกุลของนักศึกษา
   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

(1) Error ที่เกิดขึ้นหมายความอย่างไร และเกิดขึ้นเพราะอะไร
<u>ไม่สามารถเปิดพอร์ต 3000 ได้ เนื่องจากพอร์ตนั้นถูกใช้งานโดยโปรแกรมหรือคอนเทนเนอร์อื่นอยู่แล้ว</u>

11. ลบ Container ของ Web application เวอร์ชันก่อนแก้ไขออกจากระบบ โดยใช้วิธีใดวิธีหนึ่งดังต่อไปนี้
    a. ผ่าน Command line interface
        i. ใช้คำสั่ง $ docker ps เพื่อดู Container ID ที่ต้องการจะลบ
        ii. Copy หรือบันทึก Container ID ไว้
        iii. ใช้คำสั่ง $ docker stop <Container ID ที่ต้องการจะลบ> เพื่อหยุดการทำงานของ Container ดังกล่าว
        iv. ใช้คำสั่ง $ docker rm <Container ID ที่ต้องการจะลบ> เพื่อทำการลบ

    b. ผ่าน Docker desktop
        i. ไปที่หน้าต่าง Containers
        ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ

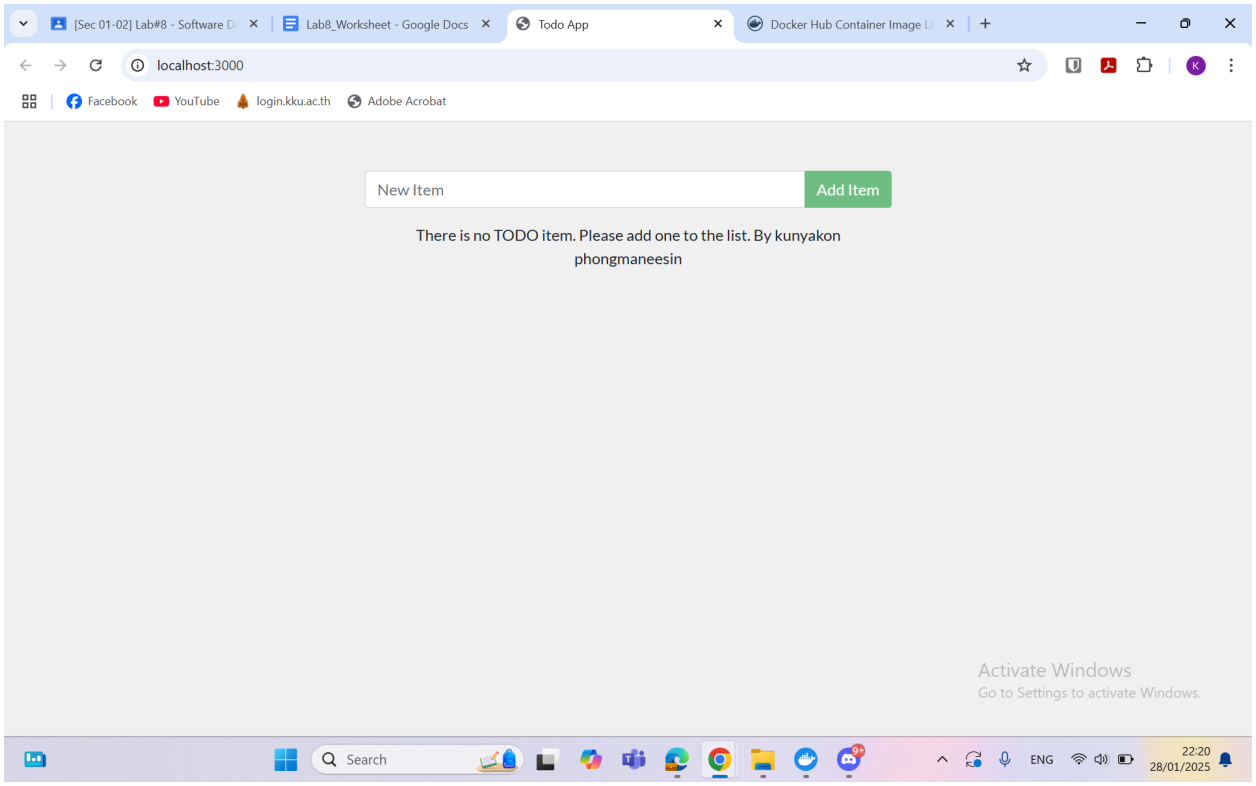
iii. ยืนยันโดยการกด Delete forever

12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6
13. เปิด Browser ไปที่ URL = http://localhost:3000

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

```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app> docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED         STATUS         PORTS
      NAMES
e7898e06e5fd   bcd5319b99ff           "docker-entrypoint.s…"   9 minutes ago   Up 9 minutes   0.0.0.0:3000->3000/tcp
      vibrant_allen
7d1df59856ae   phpmyadmin/phpmyadmin  "/docker-entrypoint.…"   4 weeks ago     Up 6 hours     0.0.0.0:8090->80/tcp
      phpmyadmin
4519f017f711   mysql:8.0              "docker-entrypoint.s…"   4 weeks ago     Up 6 hours     0.0.0.0:3306->3306/tcp, 33060/
tcp   db
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app> docker stop e7898e06e5fd
e7898e06e5fd
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app> docker rm e7898e06e5fd
e7898e06e5fd
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app>
```

```
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533802622 .
[+] Building 2.9s (10/10) FINISHED                                                          docker:desktop-linux
 => [internal] load build definition from dockerfile                                                    0.0s
 => => transferring dockerfile: 156B                                                                    0.0s
 => [internal] load metadata for docker.io/library/node:18-alpine                                       2.6s
 => [auth] library/node:pull token for registry-1.docker.io                                             0.0s
 => [internal] load .dockerignore                                                                       0.0s
 => => transferring context: 2B                                                                         0.0s
 => [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb2  0.0s
 => [internal] load build context                                                                       0.0s
 => => transferring context: 2.49kB                                                                     0.0s
 => CACHED [2/4] WORKDIR /app                                                                           0.0s
 => CACHED [3/4] COPY . .                                                                               0.0s
 => CACHED [4/4] RUN yarn install --production                                                          0.0s

 => => transferring context: 2B                                                                         0.0s
 => [1/4] FROM docker.io/library/node:18-alpine@sha256:974afb6cbc0314dc6502b14243b8a39fbb2d04d975e9059dd066be3e274fbb2  0.0s
 => [internal] load build context                                                                       0.0s
 => => transferring context: 2.49kB                                                                     0.0s
 => CACHED [2/4] WORKDIR /app                                                                           0.0s
 => CACHED [3/4] COPY . .                                                                               0.0s
 => CACHED [4/4] RUN yarn install --production                                                          0.0s
 => exporting to image                                                                                  0.0s
 => => exporting layers                                                                                 0.0s
 => => writing image sha256:a54ba51b57678fd0a6143dae1710d4ecf357259ffc05c2320732da10106be4a9            0.0s
 => => naming to docker.io/library/myapp_6533802622                                                     0.0s
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app> docker run -dp 3000:3000 myapp_6533802622
f78e0419f38932ea43476919eccc431ba837fdccbc44a0f62f0e17445eb198c7
PS C:\Users\KKU650001\Downloads\Lab8\Lab8_4\getting-started\app>
```

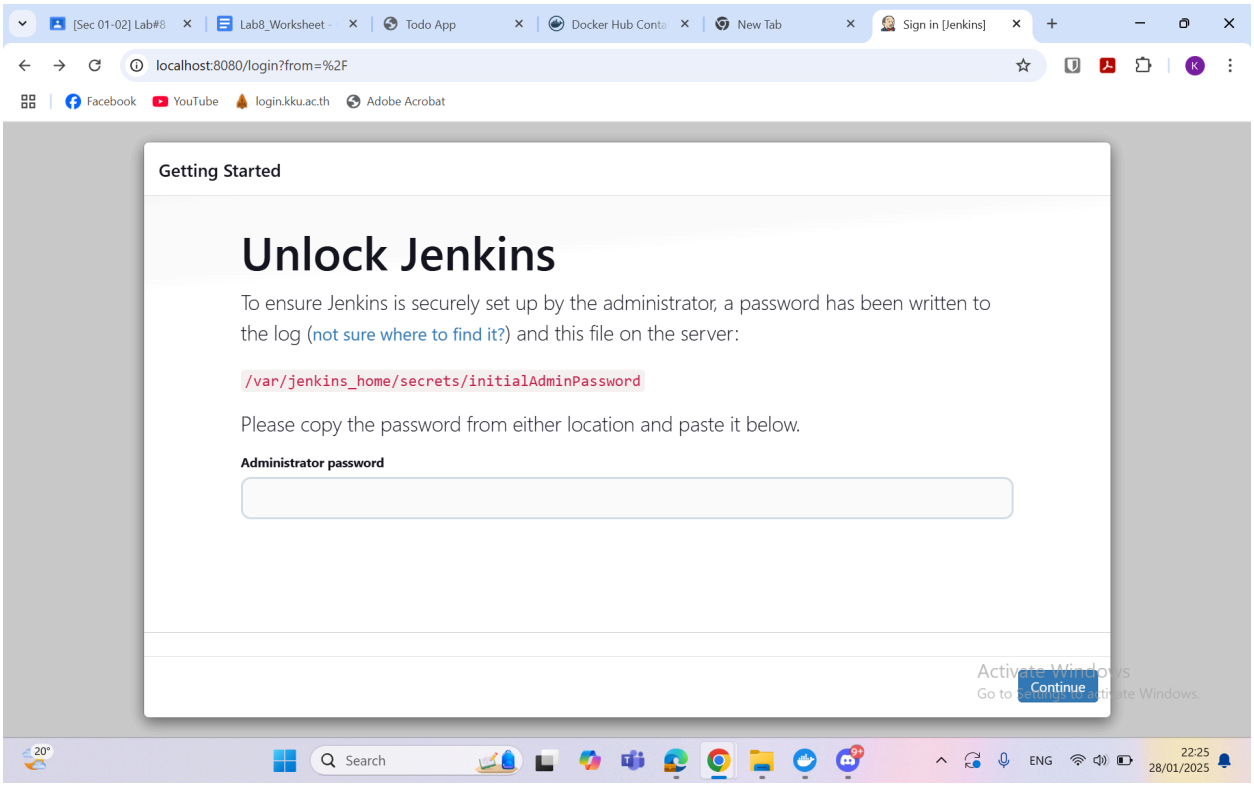


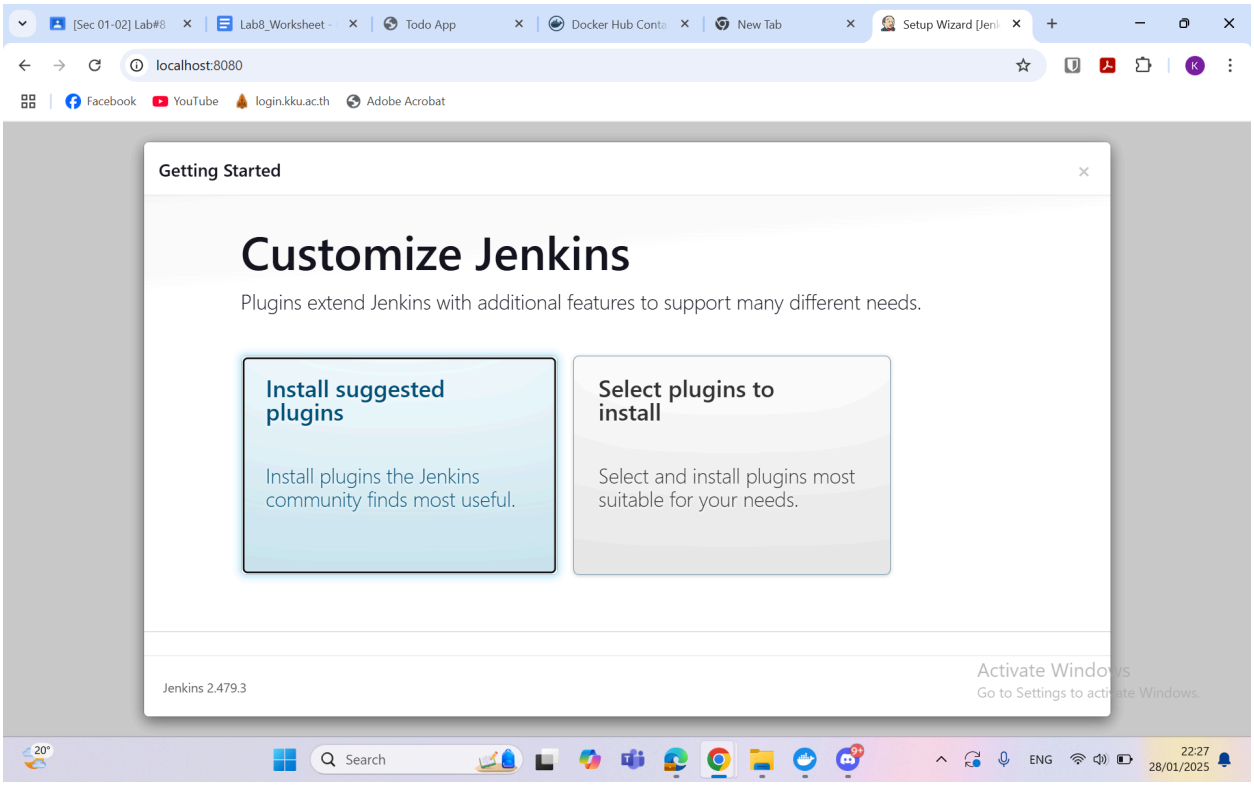
แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

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

[Check point#12] Capture หน้าจอที่แสดงผล Admin password

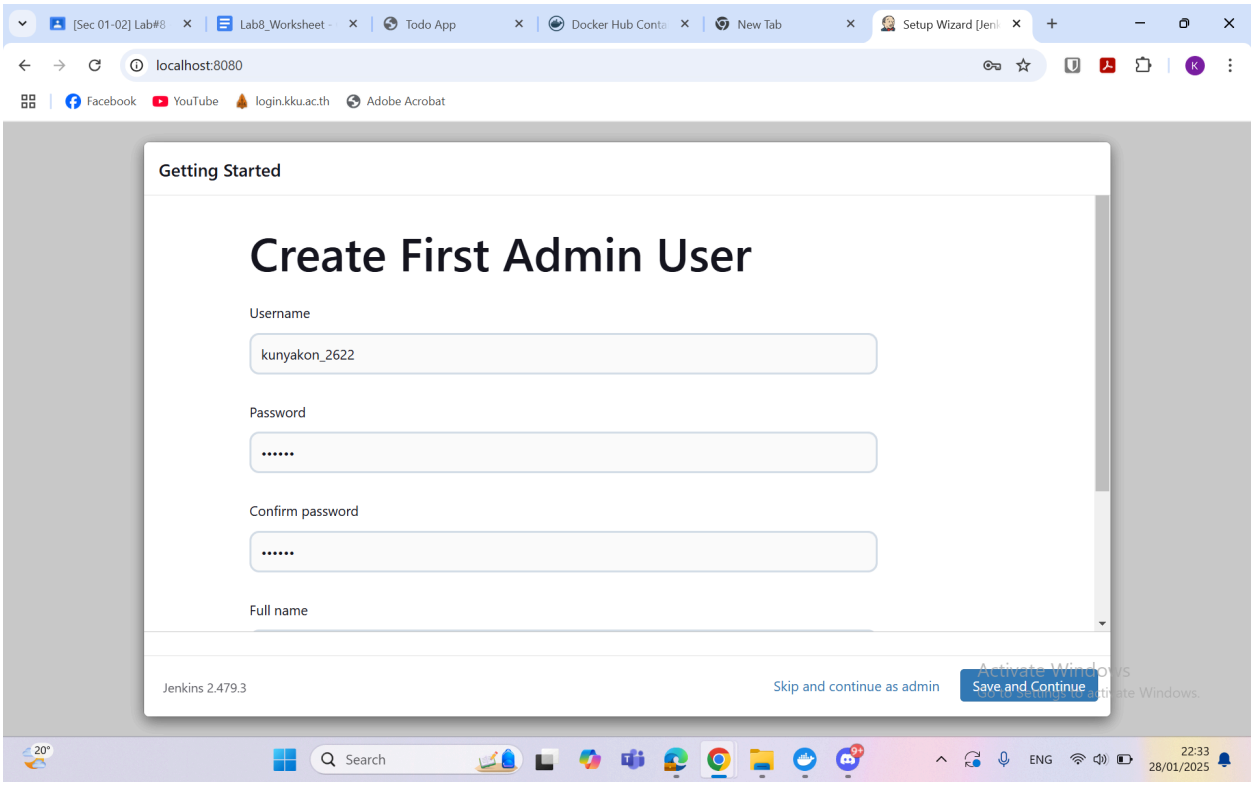
```
Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

52cc5a70e34a407e97ed0b41dd694482

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

*************************************************************
*************************************************************
*************************************************************

2025-01-28 15:22:37.194+0000 [id=38]    INFO    jenkins.InitReactorRunner$1#onAttained: Completed initialization
2025-01-28 15:22:37.225+0000 [id=25]    INFO    hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
2025-01-28 15:22:39.945+0000 [id=68]    INFO    h.m.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tas
```

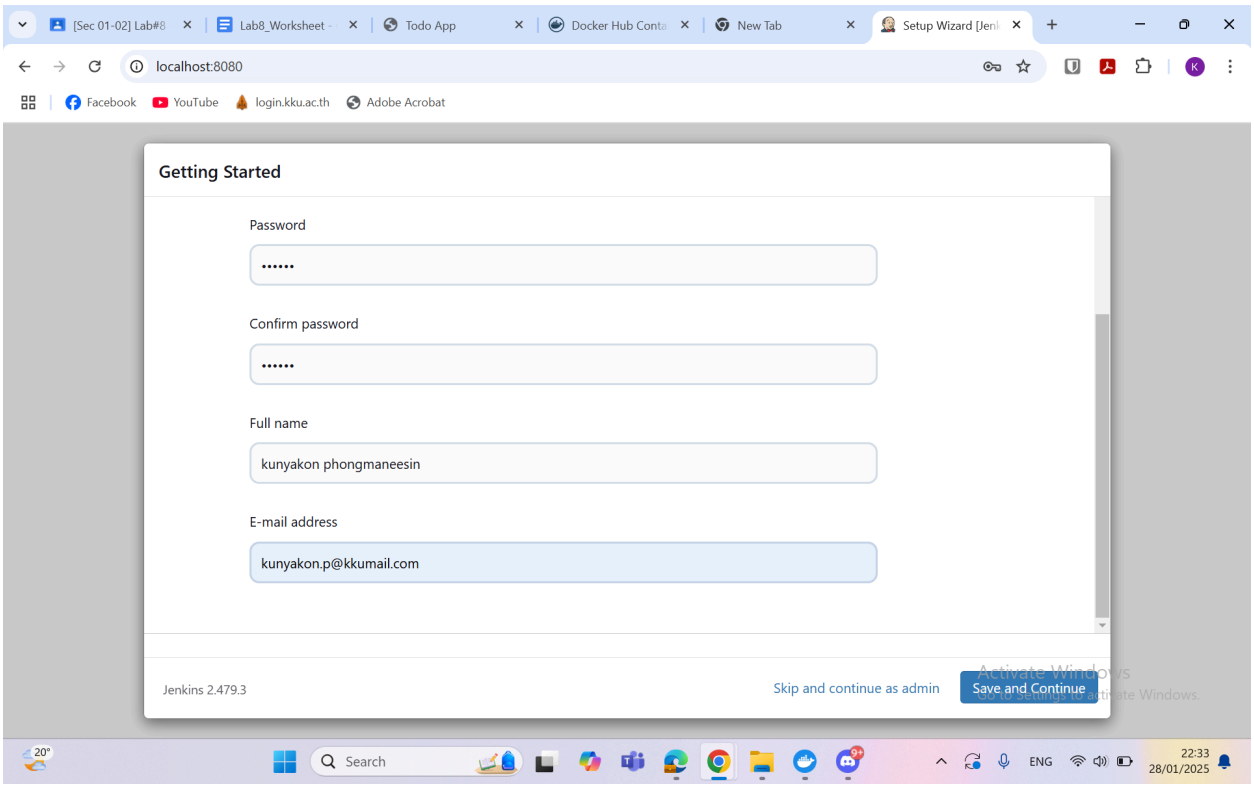




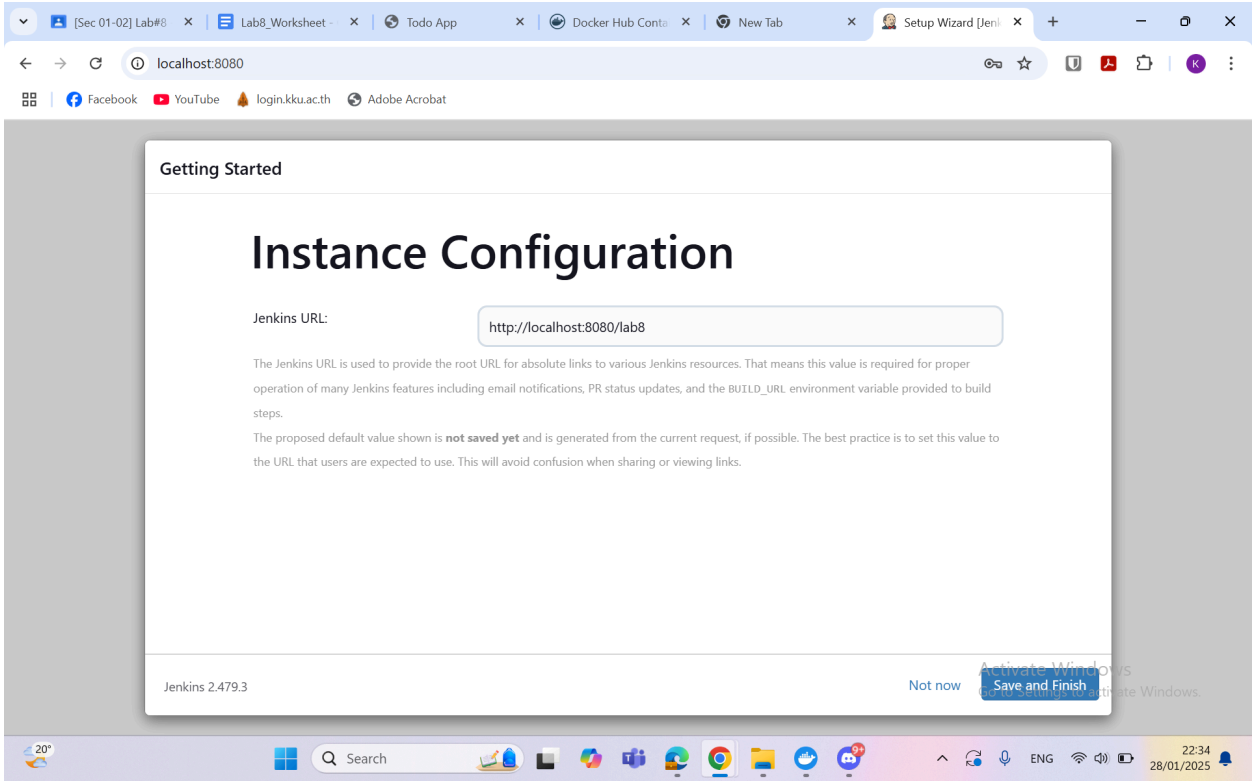
4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น localhost:8080
5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3
6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษา พร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

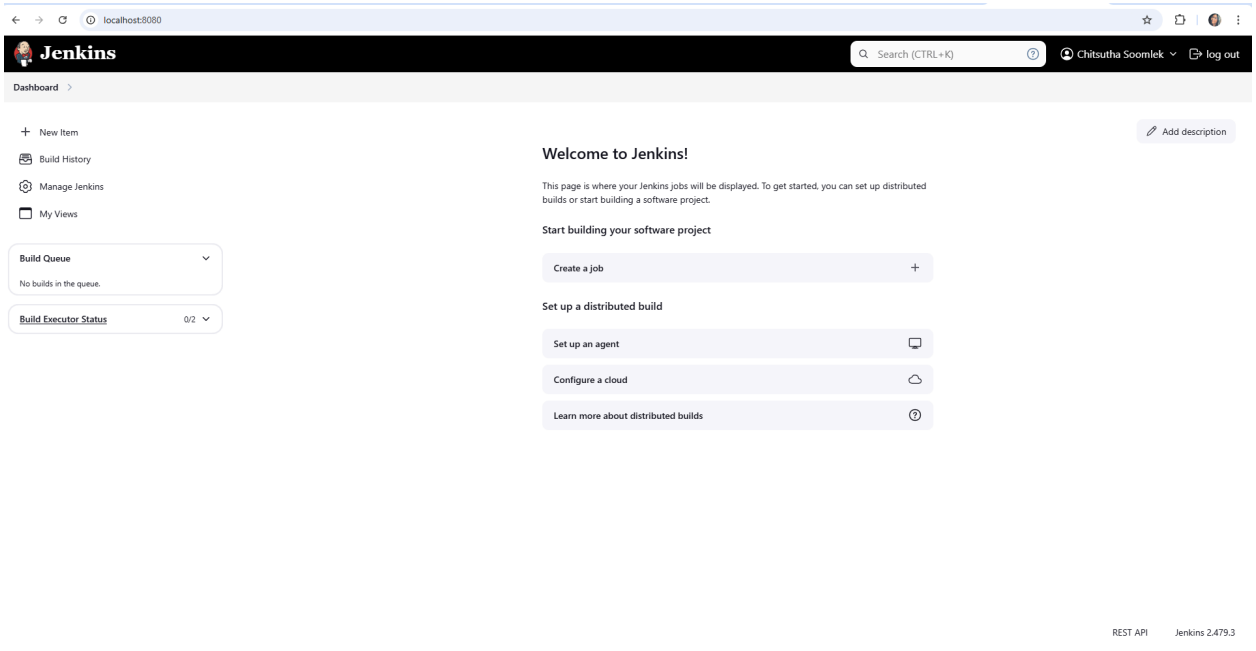

[Sec 01-02] Lab#8
Lab8_Worksheet
Todo App
Docker Hub Conta
New Tab
Setup Wizard [Jenk
localhost:8080
Facebook
YouTube
login.kku.ac.th
Adobe Acrobat
Getting Started
Create First Admin User
Username
kunyakon_2622
Password
Confirm password
Full name
Jenkins 2.479.3
Skip and continue as admin
Save and Continue
Search
ENG
22:33
28/01/2025


[Sec 01-02] Lab#8
Lab8_Worksheet
Todo App
Docker Hub Conta
New Tab
Setup Wizard [Jenk
localhost:8080
Facebook
YouTube
login.kku.ac.th
Adobe Acrobat
Getting Started
Password
Confirm password
Full name
kunyakon phongmaneesin
E-mail address
kunyakon.p@kkumail.com
Jenkins 2.479.3
Skip and continue as admin
Save and Continue
Search
ENG
22:33
28/01/2025

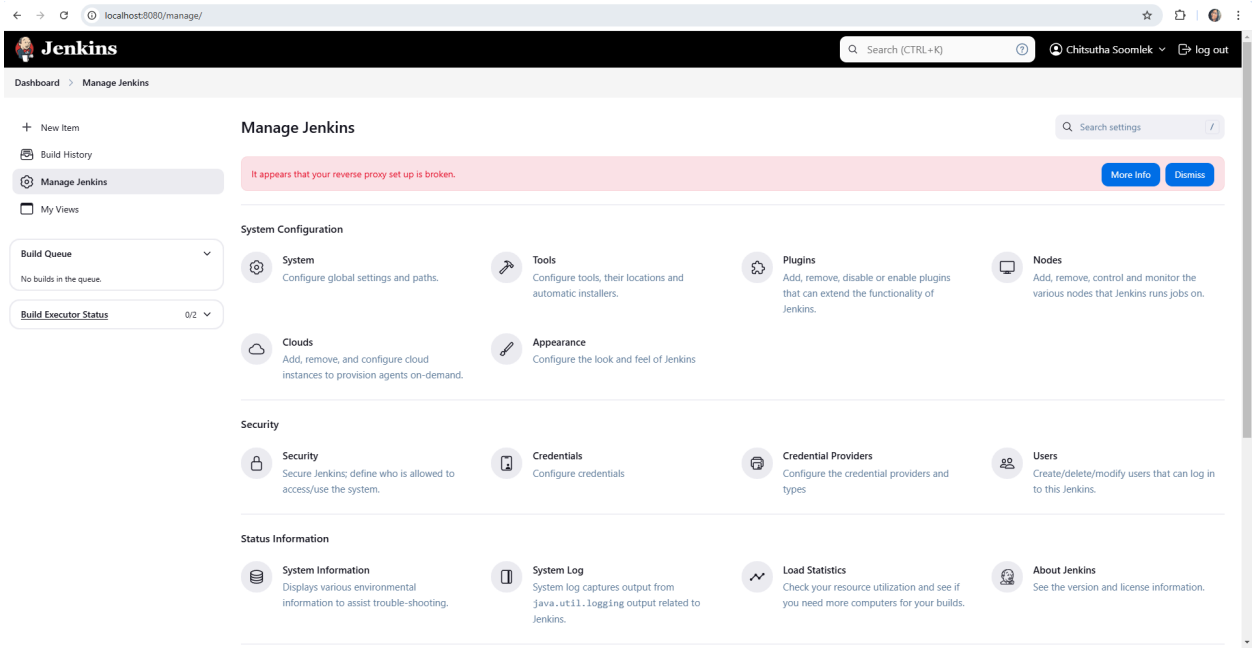
7. กำหนด Jenkins URL เป็น http://localhost:8080/lab8



8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ

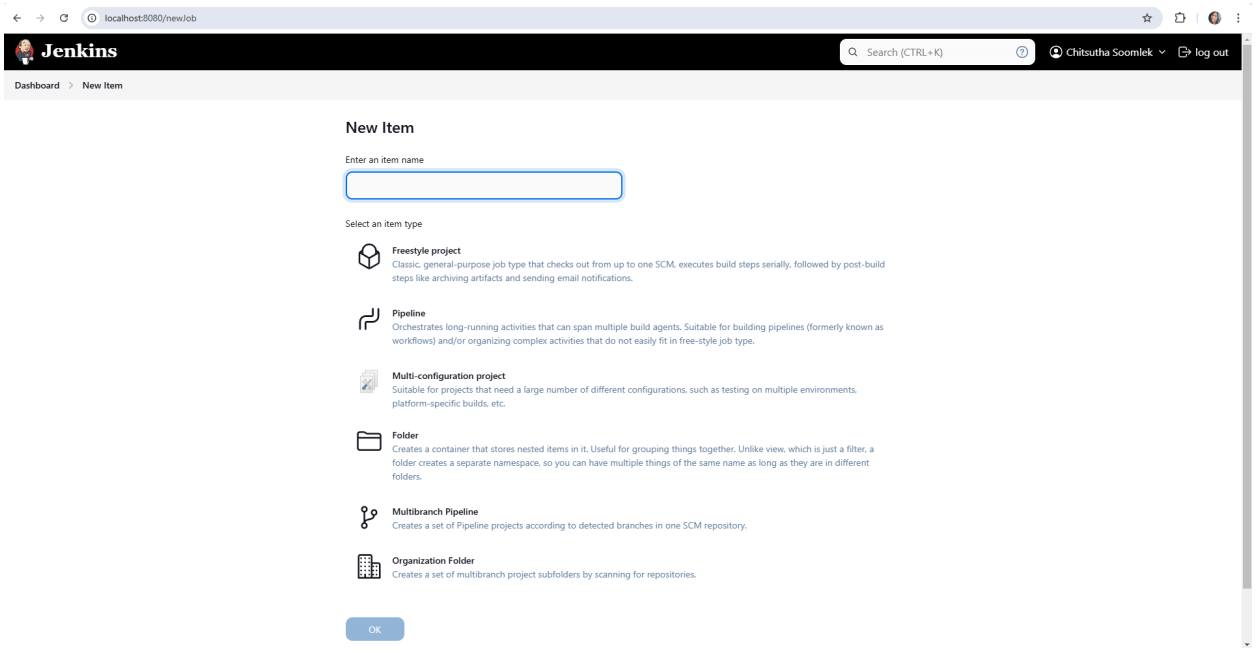


9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins



10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม

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



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

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 ไปถึงไฟล์ให้เรียบร้อยด้วย)

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

(1) คำสั่งที่ใช้ในการ Execute ไฟล์ .robot ใน Build Steps คือ robot test001.robot

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