Lab Worksheet

ชื่อ-นามสกุล_____น.ส.สุภัสสร สุทธิพัฒน์กุล_____รหัสนักศึกษา_____653380156-1_____Section___2___

Lab#8 – Software Deployment Using Docker

วัตถุประสงค์การเรียนรู้

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

Pre-requisite

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

แบบฝึกปฏิบัติที่ 8.1 Hello world - รัน Container จาก Docker image

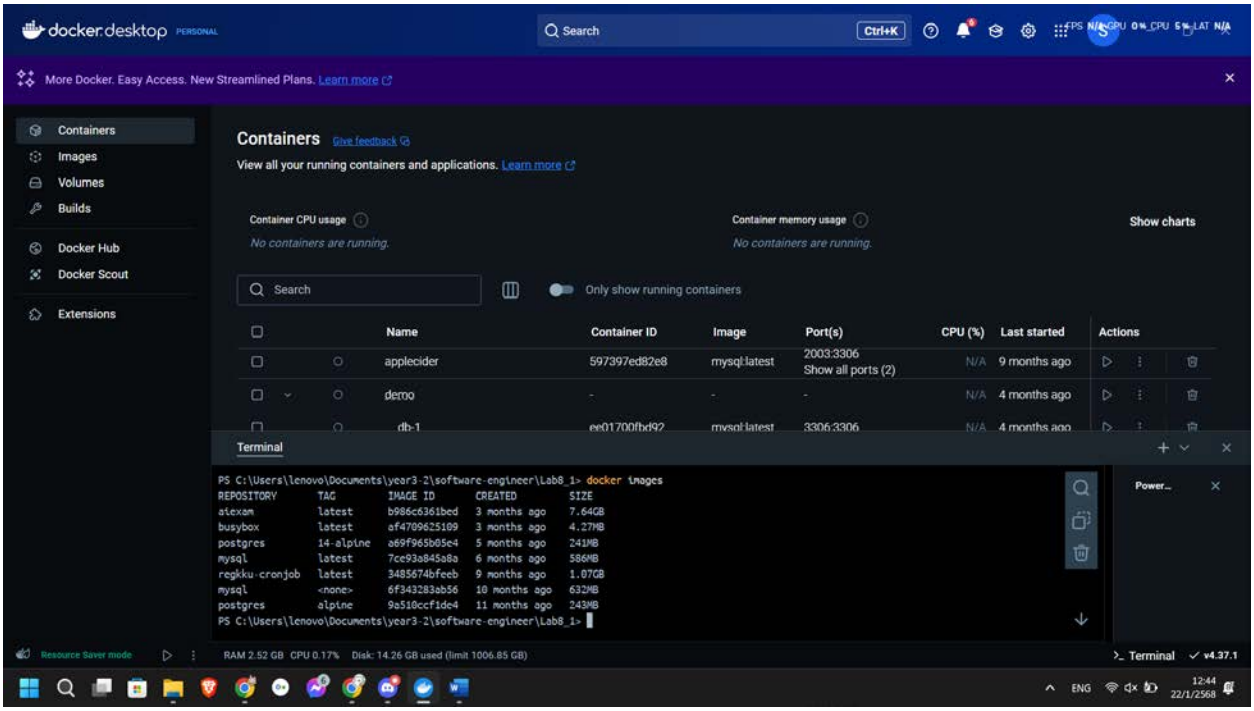
1. เปิดใช้งาน Docker desktop และ Login ด้วย Username และ Password ที่ลงทะเบียนกับ Docker Hub เอาไว้
1. เปิด Command line หรือ Terminal บน Docker Desktop จากนั้นสร้าง Directory ชื่อ Lab8_1
2. ย้ายตำแหน่งปัจจุบันไปที่ Lab8_1 เพื่อใช้เป็น Working directory
3. ป้อนคำสั่ง $ docker pull busybox หรือ $ sudo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

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

(1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร____ชื่อ image_____

(2) Tag ที่ใช้บ่งบอกถึงอะไร_____เป็น version อะไรอยู่_____

(3)



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

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

8. ป้อนคำสั่ง ls -la

```
/ # ls -la
drwxr-xr-x    2 nobody   nobody        4096 Sep 26 21:31 home
drwxr-xr-x    2 root     root          4096 Sep 26 21:31 lib
lrwxrwxrwx    1 root     root             3 Sep 26 21:31 lib64 -> lib
dr-xr-xr-x  295 root     root             0 Jan 22 05:45 proc
drwx------    1 root     root          4096 Jan 22 05:45 root
dr-xr-xr-x   11 root     root             0 Jan 22 05:45 sys
drwxrwxrwt    2 root     root          4096 Sep 26 21:31 tmp
drwxr-xr-x    4 root     root          4096 Sep 26 21:31 usr
drwxr-xr-x    4 root     root          4096 Sep 26 21:31 var
```

9. ป้อนคำสั่ง exit

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

10. ป้อนคำสั่ง $ docker run busybox echo “Hello ชื่อและนามสกุลของนักศึกษา from busybox”

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

11. ป้อนคำสั่ง $ docker ps -a

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker ps -a
CONTAINER ID   IMAGE            COMMAND                  CREATED             STATUS                       PORTS     NAMES
62e409be5756   busybox          "echo 'Hello Supatso…"   10 seconds ago      Exited (0) 9 seconds ago               kind_davinci
b1cc4182951a   busybox          "sh"                     About a minute ago  Exited (0) 49 seconds ago              compassionate_kapitsa
699ee7a0dac7   busybox          "sh"                     About a minute ago  Exited (0) About a minute ago          ecstatic_booth
cb824a6e97b5   aiexam           "jupyter lab --ip=0.…"   3 months ago        Exited (0) 3 months ago                busy_brahmagupta
ee01700fbd92   mysql:latest     "docker-entrypoint.s…"   4 months ago        Exited (0) 4 months ago                demo-db-1
088a1c6a7f44   postgres:alpine  "docker-entrypoint.s…"   4 months ago        Exited (0) 4 months ago                rentservice-postgres-1
2b595ef4a416   mysql:latest     "docker-entrypoint.s…"   6 months ago        Exited (0) 4 months ago                lab5-db-1
597397ed82e8   6f343283ab56     "docker-entrypoint.s…"   8 months ago        Exited (0) 8 months ago                applecider
c0762c239b5a   regkku-cronjob   "cron -f"                9 months ago        Exited (137) 9 months ago              regkku-cronjob-1
```

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

(1) เมื่อใช้ option -it ในคำสั่ง run ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป

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

(2) คอลัมน์ STATUS จากการรันคำสั่ง docker ps -a แสดงถึงข้อมูลอะไร

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

```
drwxr-xr-x    4 root     root          4096 Sep 26 21:31 usr
drwxr-xr-x    4 root     root          4096 Sep 26 21:31 var
/ # exit
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker run busybox echo "Hello Supatsorn 653380156-1 from busybox"
Hello Supatsorn 653380156-1 from busybox
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker ps -a
CONTAINER ID   IMAGE            COMMAND                  CREATED             STATUS                       PORTS     NAMES
62e409be5756   busybox          "echo 'Hello Supatso…"   10 seconds ago      Exited (0) 9 seconds ago               kind_davinci
b1cc4182951a   busybox          "sh"                     About a minute ago  Exited (0) 49 seconds ago              compassionate_kapitsa
699ee7a0dac7   busybox          "sh"                     About a minute ago  Exited (0) About a minute ago          ecstatic_booth
cb824a6e97b5   aiexam           "jupyter lab --ip=0.…"   3 months ago        Exited (0) 3 months ago                busy_brahmagupta
ee01700fbd92   mysql:latest     "docker-entrypoint.s…"   4 months ago        Exited (0) 4 months ago                demo-db-1
088a1c6a7f44   postgres:alpine  "docker-entrypoint.s…"   4 months ago        Exited (0) 4 months ago                rentservice-postgres-1
2b595ef4a416   mysql:latest     "docker-entrypoint.s…"   6 months ago        Exited (0) 4 months ago                lab5-db-1
597397ed82e8   6f343283ab56     "docker-entrypoint.s…"   8 months ago        Exited (0) 8 months ago                applecider
c0762c239b5a   regkku-cronjob   "cron -f"                9 months ago        Exited (137) 9 months ago              regkku-cronjob-1
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1>
```

12. ป้อนคำสั่ง $ docker rm <container ID ที่ต้องการลบ>

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

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker rm 699ee7a0dac7
699ee7a0dac7
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_1> docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED          STATUS                      PORTS     NAMES
a52f797c11f5   lab8_2            "/bin/sh -c 'echo \"S…"   31 minutes ago   Exited (0) 31 minutes ago             vigilant_wescoff
ea6a6862692c   lab8_2            "/bin/sh -c 'echo \"S…"   31 minutes ago   Exited (0) 31 minutes ago             gracious_goldstine
12b2183bad8c   lab8_2            "/bin/sh -c 'echo \"S…"   32 minutes ago   Exited (0) 32 minutes ago             lucid_swirles
62e409be5756   busybox           "echo 'Hello Supatso…"   43 minutes ago   Exited (0) 43 minutes ago             kind_davinci
b1cc4182951a   busybox           "sh"                     44 minutes ago   Exited (0) 44 minutes ago             compassionate_kapitsa
cb824a6e97b5   aiexam            "jupyter lab --ip=0.…"   3 months ago     Exited (0) 3 months ago               busy_brahmagupta
ee01700fbd92   mysql:latest      "docker-entrypoint.s…"   4 months ago     Exited (0) 4 months ago               demo-db-1
088a1c6a7f44   postgres:alpine   "docker-entrypoint.s…"   4 months ago     Exited (0) 4 months ago               rentservice-postgres-1
2b595ef4a416   mysql:latest      "docker-entrypoint.s…"   6 months ago     Exited (0) 4 months ago               lab5-db-1
597397ed82e8   6f343283ab56      "docker-entrypoint.s…"   8 months ago     Exited (0) 8 months ago               applecider
c0762c239b5a   regkku-cronjob    "cron -f"                9 months ago     Exited (137) 9 months ago             regkku-cronjob-1
```

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) บันทึกคำสั่งต่อไปนี้ลงในไฟล์โดยใช้ 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> .

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

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

 3 warnings found (use docker --debug to expand):
 - JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)
 - MultipleInstructionsDisallowed: Multiple CMD instructions should not be used in the same stage because only the last one will be used (line 2)
 - JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 3)

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

6. เมื่อ Build สำเร็จแล้ว ให้ทำการรัน Docker image ที่สร้างขึ้นในขั้นตอนที่ 5

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

(1) คำสั่งที่ใช้ในการ run คือ

_____ docker run lab8_2_____

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

_____เรียกใช้ pseudo terminal เช่น bash หรือ sh _____

แบบฝึกปฏิบัติที่ 8.3: การแชร์ Docker image ผ่าน Docker Hub

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

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

```
PS C:\Users\lenovo\Documents\year3-2\software-engineer\Lab8_3> docker build -t supatz3107/lab8 .
[+] Building 0.3s (5/5) FINISHED                                                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                              0.1s
 => => transferring dockerfile: 178B                                                                                              0.0s
 => WARN: JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 2)  0.1s
 => [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                                                                                                            0.1s
 => => exporting layers                                                                                                           0.0s
 => => writing image sha256:0600fe756a1c6344ab23f2070493a53ad029abe4414f886e0485684ea9e33ecd                                      0.0s
 => => naming to docker.io/supatz3107/lab8                                                                                        0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/saz49g9drim3q3u1umkj24k39

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

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

5. ทำการรัน Docker image บน Container ในเครื่องของตัวเองเพื่อทดสอบผลลัพธ์ ด้วยคำสั่ง

$ docker run <username ที่ลงทะเบียนกับ Docker Hub>/lab8

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

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

6. ทำการ Push ตัว Docker image ไปไว้บน Docker Hub โดยการใช้คำสั่ง

   $ docker push <username ที่ลงทะเบียนกับ Docker Hub>/lab8

   ในกรณีที่ติดปัญหาไม่ได้ Login ไว้ก่อน ให้ใช้คำสั่งต่อไปนี้ เพื่อ Login ก่อนทำการ Push

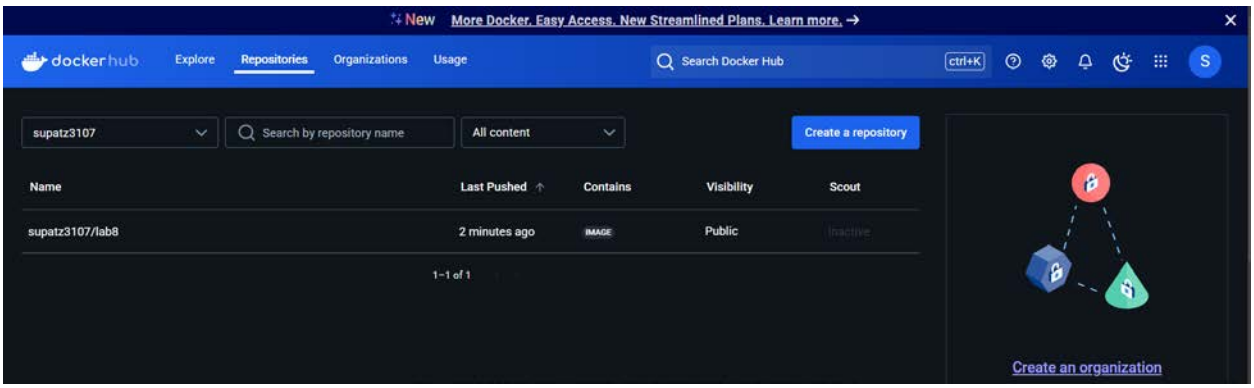
   $ docker login แล้วป้อน Username และ Password ตามที่ระบุใน Command prompt หรือใช้คำสั่ง

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

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

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

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



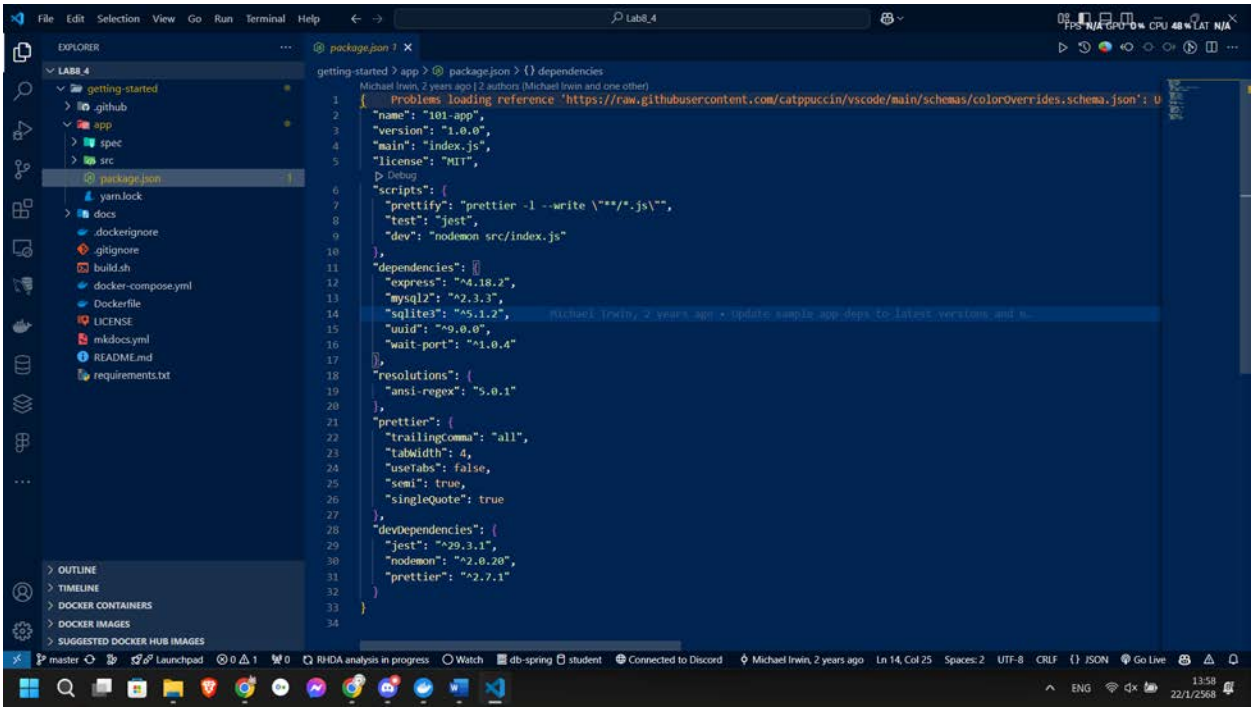
แบบฝึกปฏิบัติที่ 8.4: การ Build แอปพลิเคชันจาก Container image และการ Update แอปพลิเคชัน

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

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

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



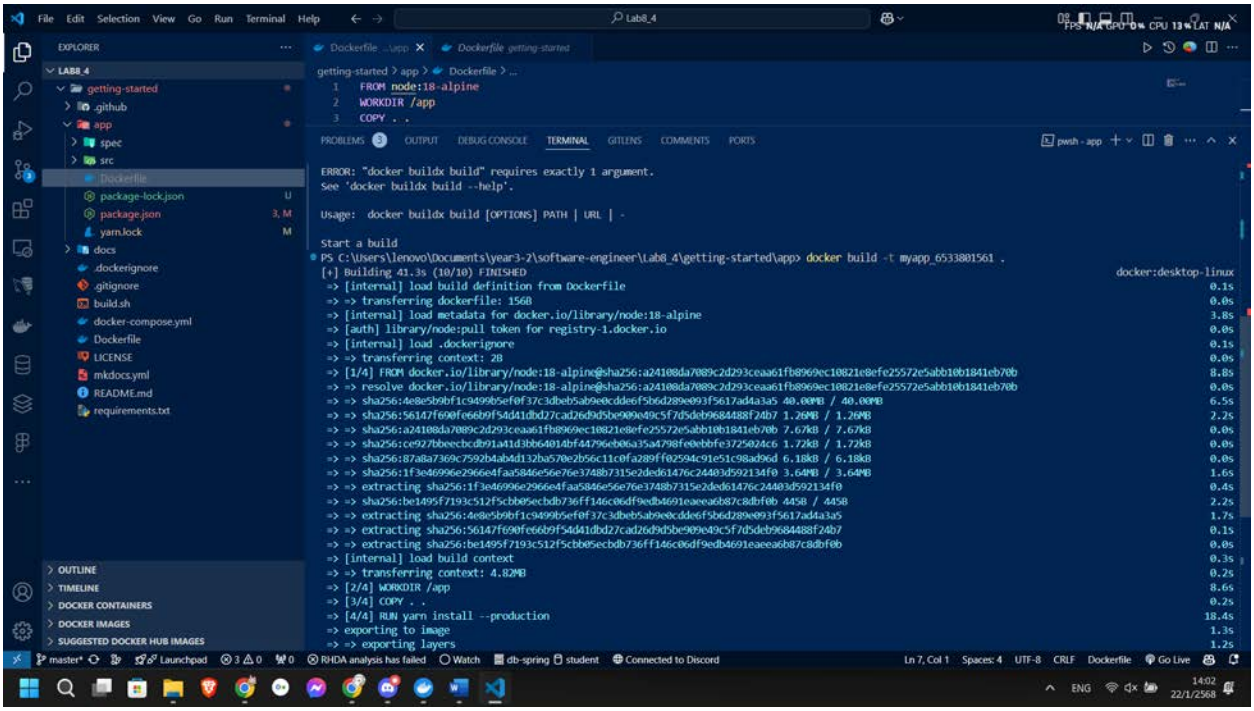
4. ภายใต้ getting-started/app ให้สร้าง Dockerfile พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไปในไฟล์

```
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> .
```

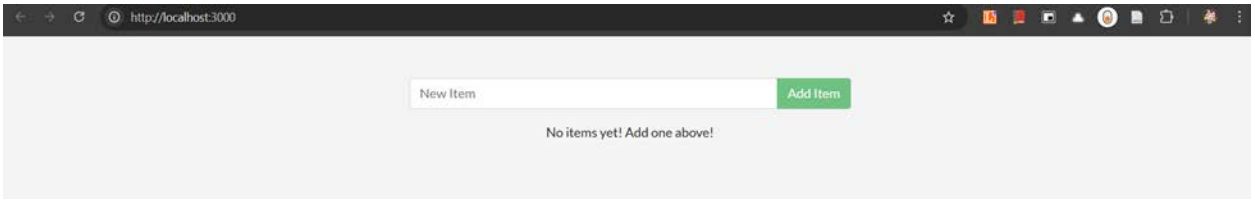
[Check point#8] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ



6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง

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

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

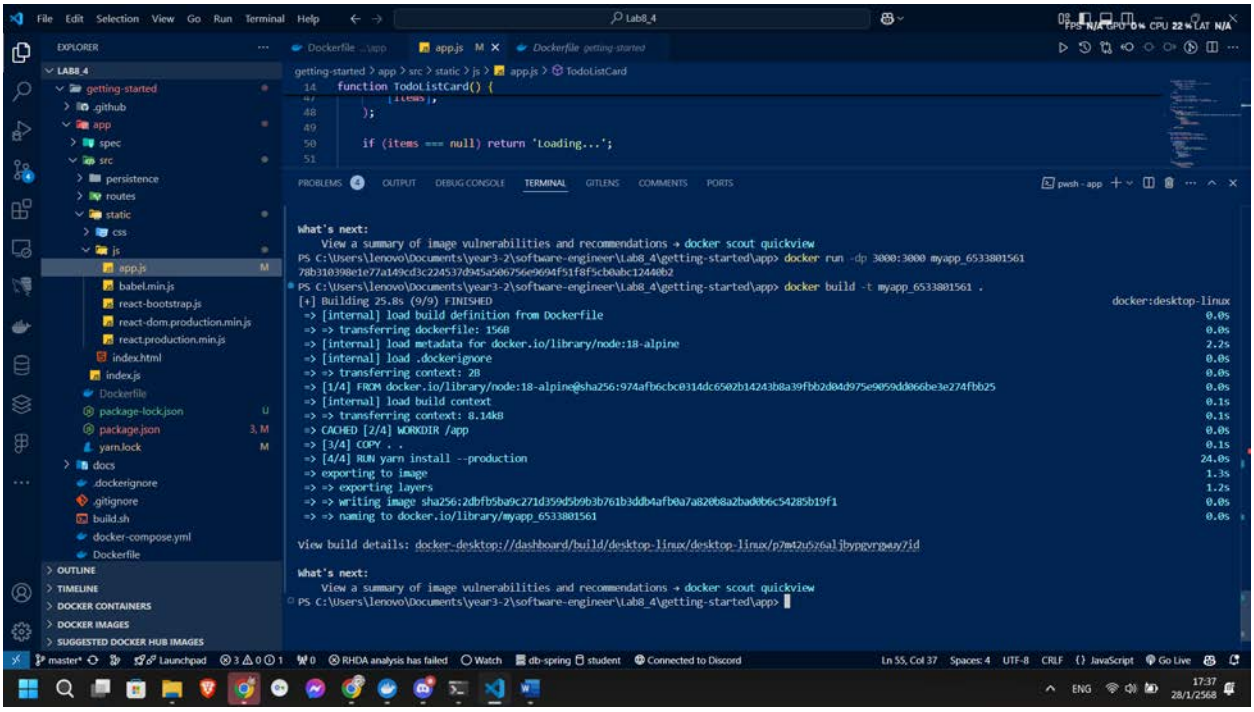


[Check point#9] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง) แสดงผลลัพธ์ที่ได้บน Browser และ Dashboard ของ Docker desktop

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



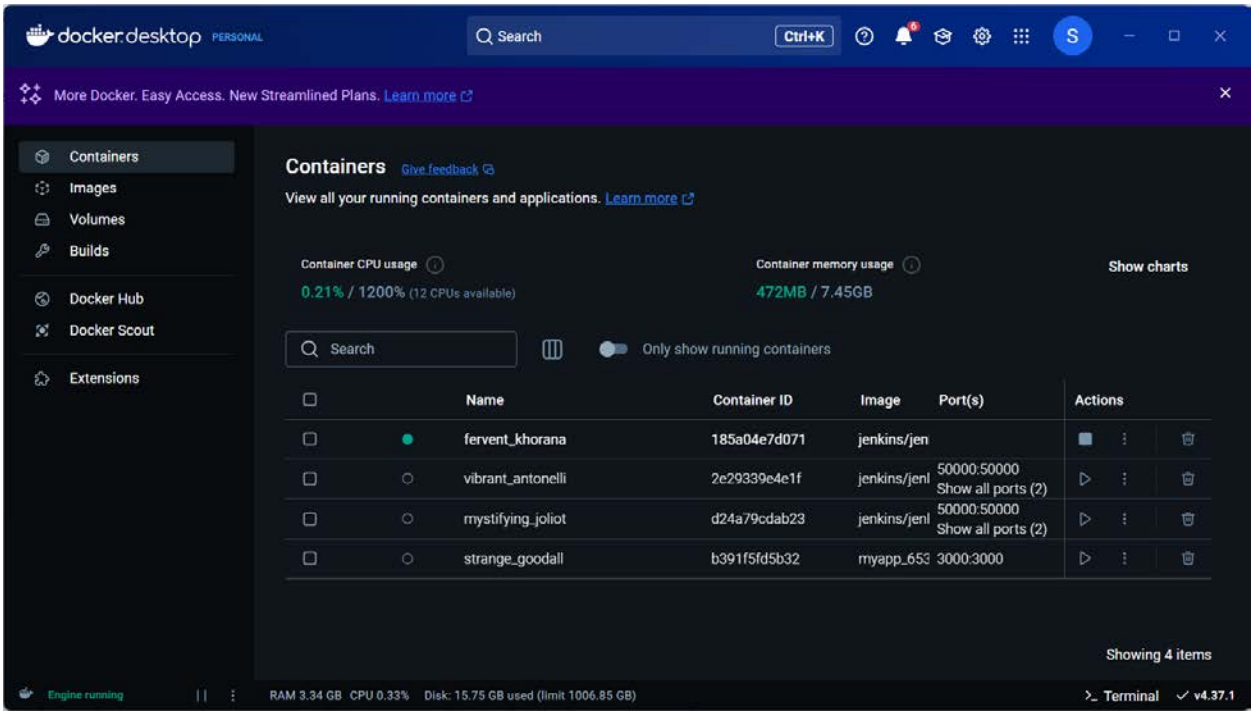
(1) Error ที่เกิดขึ้นหมายความอย่างไร และเกิดขึ้นเพราะอะไร

_____port 3000 กำลังใช้งานอยู่ โดยงานอื่น_____

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

    b. ผ่าน Docker desktop
        i. ไปที่หน้าต่าง Containers
        ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
        iii. ยืนยันโดยการกด Delete forever
12. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6
13. เปิด Browser ไปที่ URL = http://localhost:3000

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



แบบฝึกปฏิบัติที่ 8.5: เริ่มต้นสร้าง Pipeline อย่างง่ายสำหรับการ Deploy ด้วย Jenkins

1. เปิด Command line หรือ Terminal บน Docker Desktop
2. ป้อนคำสั่งและทำการรัน container โดยผูกพอร์ต

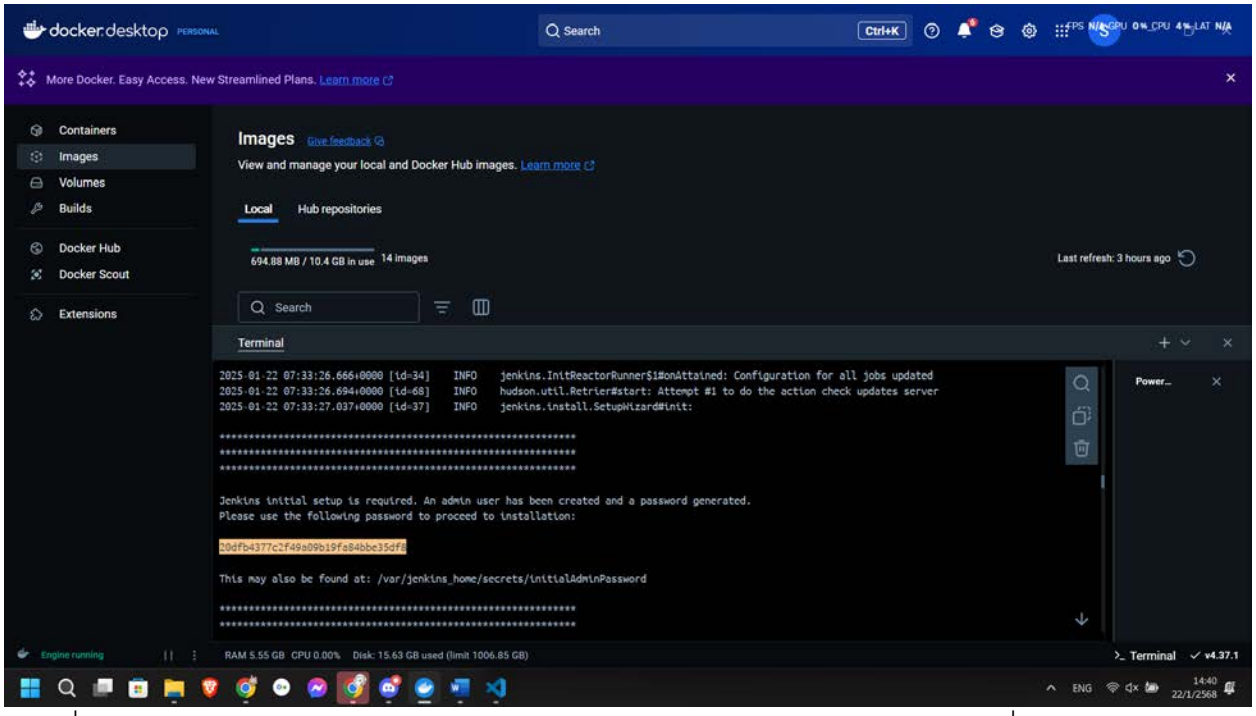
   $ docker run -p 8080:8080 -p 50000:50000 --restart=on-failure jenkins/jenkins:lts-jdk17

   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

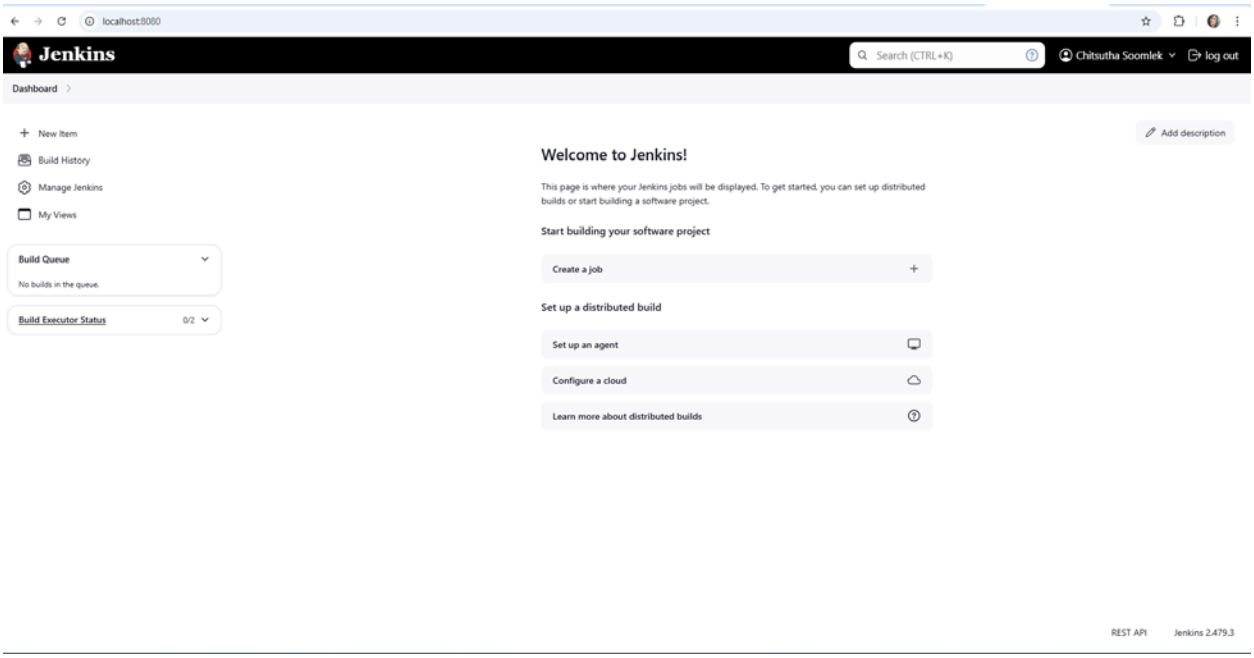
[Check point#12] Capture หน้าจอที่แสดงผล Admin password



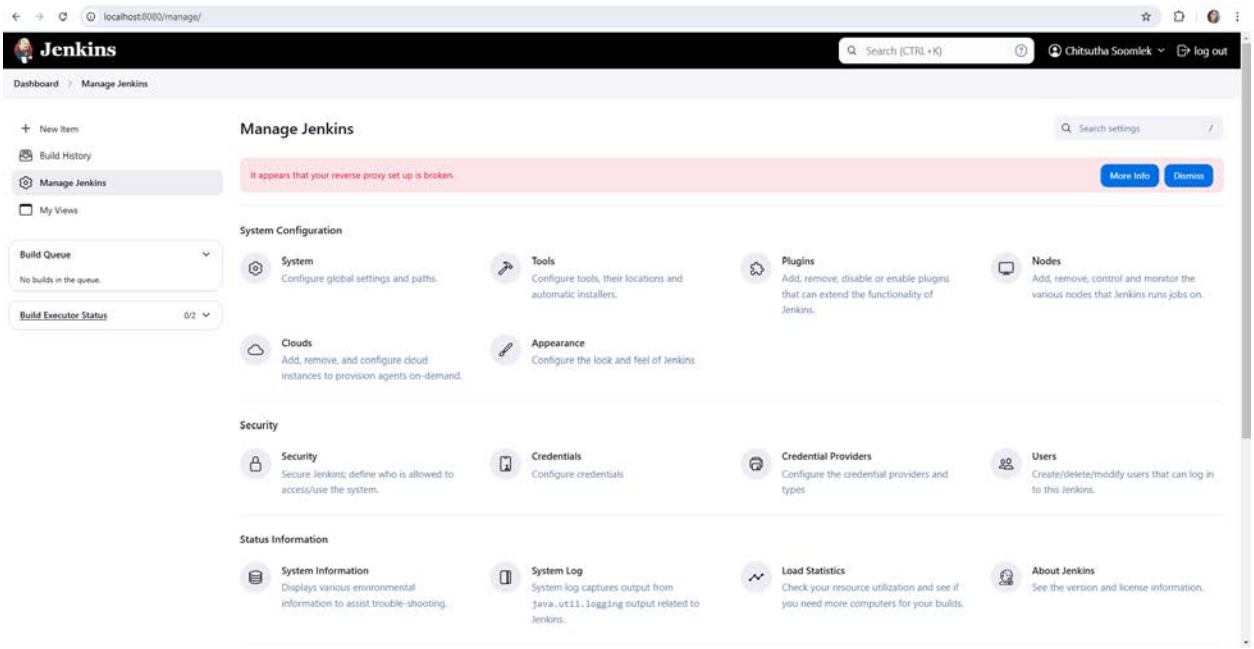
4. เมื่อได้รับการยืนยันว่า Jenkins is fully up and running ให้เปิดบราวเซอร์ และป้อนที่อยู่เป็น localhost:8080
5. ทำการ Unlock Jenkins ด้วยรหัสผ่านที่ได้ในข้อที่ 3
6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri_3062

[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า

7. กำหนด Jenkins URL เป็น http://localhost:8080/lab8
8. เมื่อติดตั้งเรียบร้อยแล้วจะพบกันหน้า Dashboard ดังแสดงในภาพ

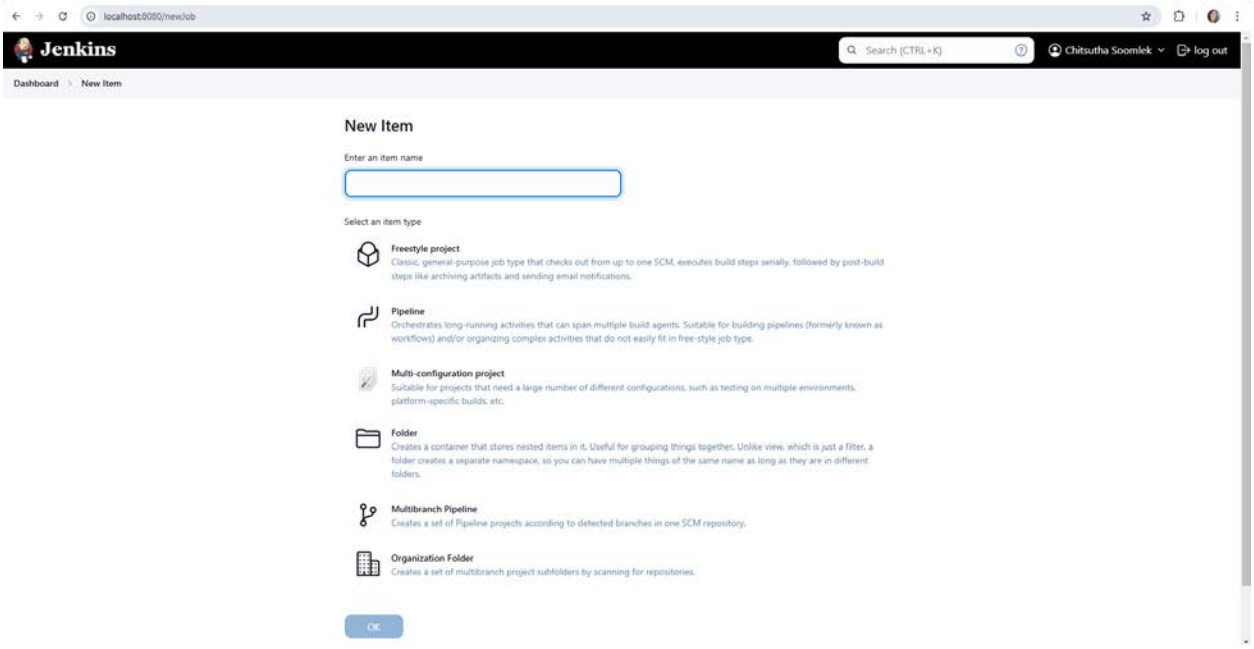


9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins



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 จากนั้นตั้งค่าที่จำเป็นในหน้านี้ทั้งหมด ดังนี้

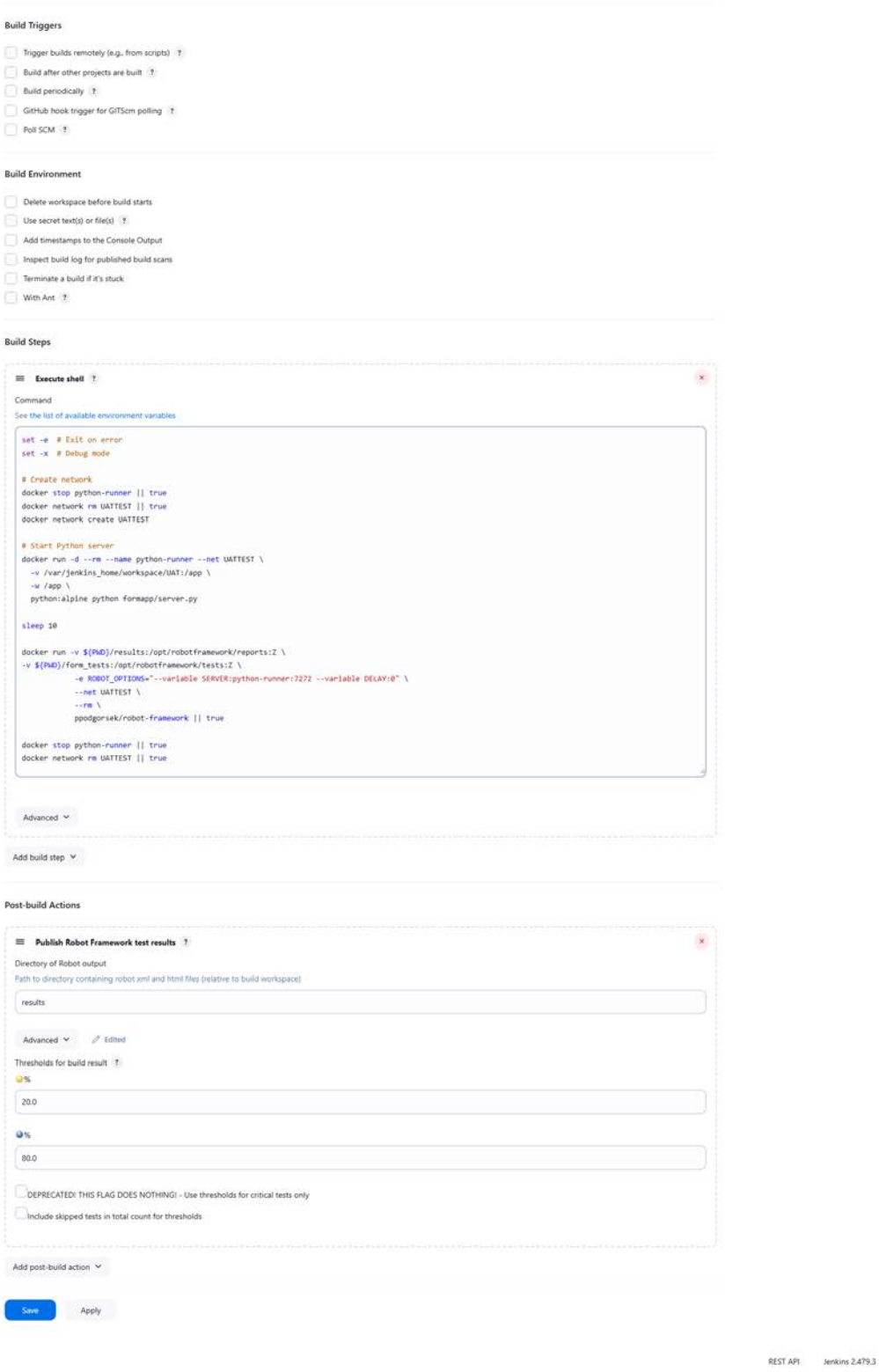
Description: Lab 8.5

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 ไปถึงไฟล์ให้เรียบร้อยด้วย)

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้


Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Add timestamps to the Console Output
Inspect build log for published build scans
Terminate a build if it's stuck
With Ant
Build Steps
Execute shell
Command
See the list of available environment variables
set -e # Exit on error
set -x # Debug mode
Create network
docker stop python-runner || true
docker network rm UATTEST || true
docker network create UATTEST
Start Python server
docker run -d --rm --name python-runner --net UATTEST \
-v /var/jenkins_home/workspace/UAT:/app \
-w /app \
python:alpine python formapp/server.py
sleep 10
docker run -v ${PWD}/results:/opt/robotframework/reports:Z \
-v ${PWD}/form_tests:/opt/robotframework/tests:Z \
-e ROBOT_OPTIONS="--variable SERVER:python-runner:7272 --variable DELAY:0" \
--net UATTEST \
--rm \
ppodgorsek/robot-framework || true
docker stop python-runner || true
docker network rm UATTEST || true
Advanced
Add build step
Post-build Actions
Publish Robot Framework test results
Directory of Robot output
Path to directory containing robot xml and html files (relative to build workspace)
results
Advanced
Edited
Thresholds for build result
%
20.0
%
80.0
DEPRECATED! THIS FLAG DOES NOTHING! - Use thresholds for critical tests only
Include skipped tests in total count for thresholds
Add post-build action
Save
Apply
REST API
Jenkins 2.479.3

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

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

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

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

sleep 10

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

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

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

Lab Worksheet