

Lab Worksheet

ชื่อ-นามสกุล อรไอลิน อิทธิศาล รหัสนักศึกษา 653380029-8 Section 2

Lab#8 – Software Deployment Using Docker

วัตถุประสงค์การเรียนรู้

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

Pre-requisite

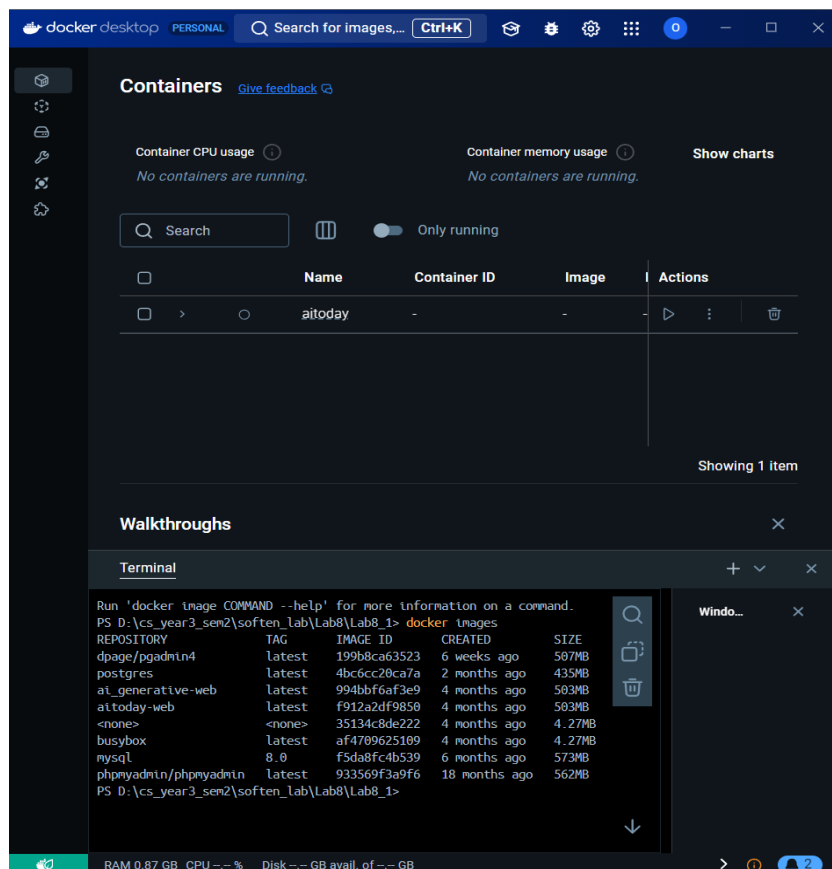
1. ติดตั้ง Docker desktop ลงบนเครื่องคอมพิวเตอร์ โดยดาวน์โหลดจาก <https://www.docker.com/get-started>
2. สร้าง Account บน Docker hub (<https://hub.docker.com/signup>)
3. กำหนดให้ \$ หมายถึง Command prompt และ <> หมายถึง ให้ป้อนค่าของพารามิเตอร์ที่กำหนด

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

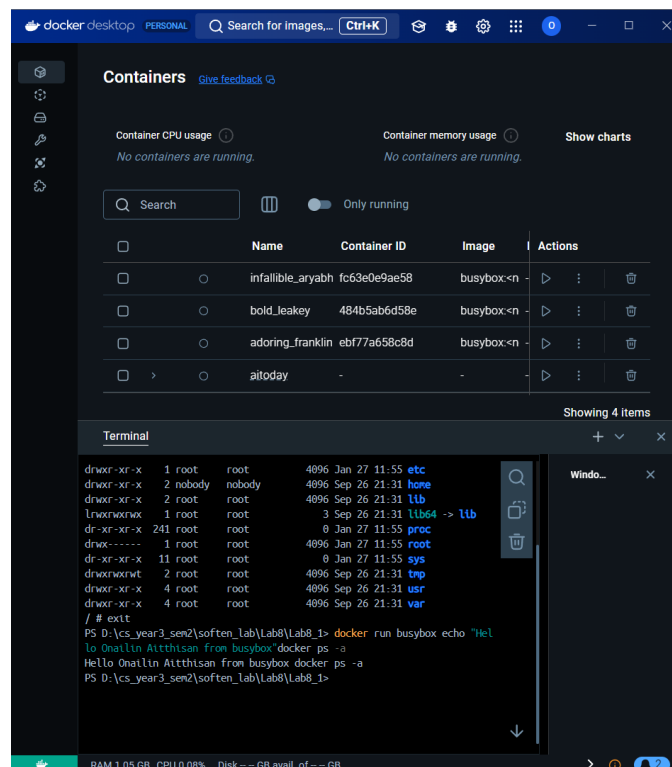


(1) สิ่งที่อยู่ภายใต้คอลัมน์ Repository คืออะไร Image name

(2) Tag ที่ใช้บ่งบอกถึงอะไร version ของ 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

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



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

```
PS D:\cs_year3_sem2\soften_lab\Lab8\Lab8_1> docker ps -a
adoring_franklin
484b5ab6d58e busybox "sh" 3 minutes ago Exited (0) 3 minutes ago bold_
leakey
fc63e0e9ae58 busybox "sh" 3 minutes ago Exited (0) 3 minutes ago infal
lible_aryabhata
488b2e77cda5 phpmyadmin/phpmyadmin "/docker-entrypoint..." 4 months ago Exited (255) 4 weeks ago 0.0.0.0:8081->80/tcp phpmy
admin
df72adcd19ab atoday-web "docker-php-entrypoint..." 4 months ago Exited (255) 4 weeks ago 0.0.0.0:8080->80/tcp php_a
pp
54e55e328c86 mysql:8.0 "docker-entrypoint.s..." 4 months ago Exited (255) 4 weeks ago 33060/tcp, 0.0.0.0:3307->3306/tcp mysql
_db
PS D:\cs_year3_sem2\soften_lab\Lab8\Lab8_1> docker rm ebf77a658c8d
ebf77a658c8d

RAM 1.05 GB CPU 0.08% Disk -- GB avail. of -- GB
```

Lab Worksheet

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
```

```
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

```
Terminal
PS D:\cs_year3_sen2\soften_lab\Lab8\Lab8_2> docker build -t dockerfile -f Dockerfile.swp .
[+] Building 0.2s (5/5) FINISHED
=> [internal] load build definition from Dockerfile.swp 0.0s
=> => transferring dockerfile: 161B 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD 0.0s
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructio 0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD 0.0s
=> [internal] load metadata for docker.io/library/busybox:latest 0.0s
=> [internal] load .dockerignore 0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/mlfvap86tp422hnah96tord3f

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:\cs_year3_sen2\soften_lab\Lab8\Lab8_2> docker run dockerfile
"O'nailin Atthitsan 653388029-8 Arty"
```

Lab Worksheet

- (1) คำสั่งที่ใช้ในการ run คือ

Docker run dockerfile

- (2) Option -t ในคำสั่ง \$ docker build ส่งผลต่อการทำงานของคำสั่งอย่างไรบ้าง อธิบายมาพอสังเขป
ใช้เพื่อตั้งชื่อและ tag ให้กับ Docker Image เพื่อให้สะดวกในการเลือกใช้หากมีมากกว่า 1 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

```

Terminal
PS D:\cs_year3_sem2\soften_lab\Lab8\Lab8_3> docker build -t onallin/lab8
-f Dockerfile.swp .
>>
[*] Building 0.2s (5/5) FINISHED          docker:desktop-linux
=> [internal] load build definition from Dockerfile.swp          0.0s
=> => transferring dockerfile: 183B                             0.0s
=> WARN: JSONArgsRecommended: JSON arguments recommended for CMD 0.0s
=> WARN: MultipleInstructionsDisallowed: Multiple CMD instructio 0.0s
=> => naming to docker.io/onallin/lab8                          0.0s

View build details: docker:desktop://dashboard/build/desktop-linux/desktop-linux/rbyx72cw9dl1i0tug5cdjzvjk

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)

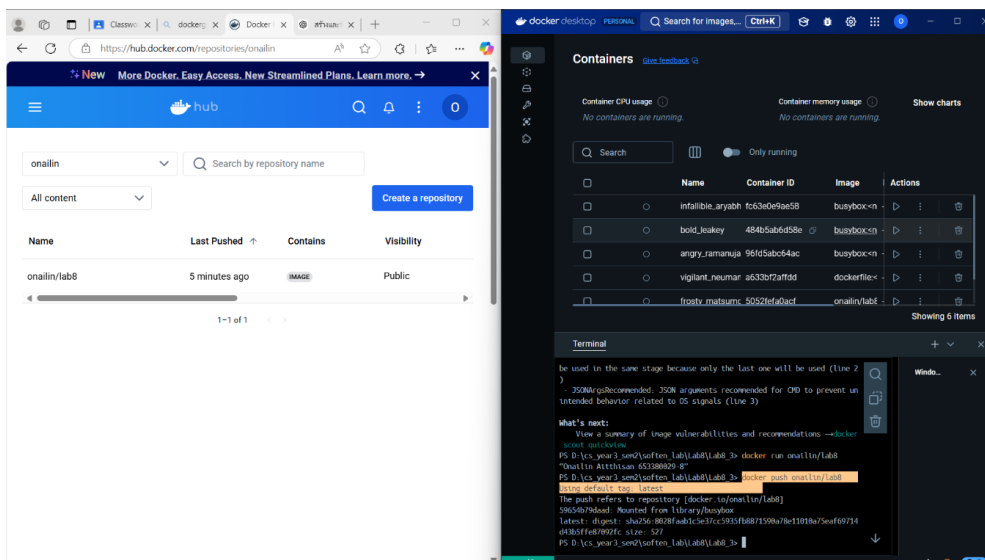
RAM 1.10 GB CPU 0.17% Disk -- GB avail. of -- GB

What's next:
View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS D:\cs_year3_sem2\soften_lab\Lab8\Lab8_3> docker run onallin/lab8
"Onallin Attttisan 653380029-8"
PS D:\cs_year3_sem2\soften_lab\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)

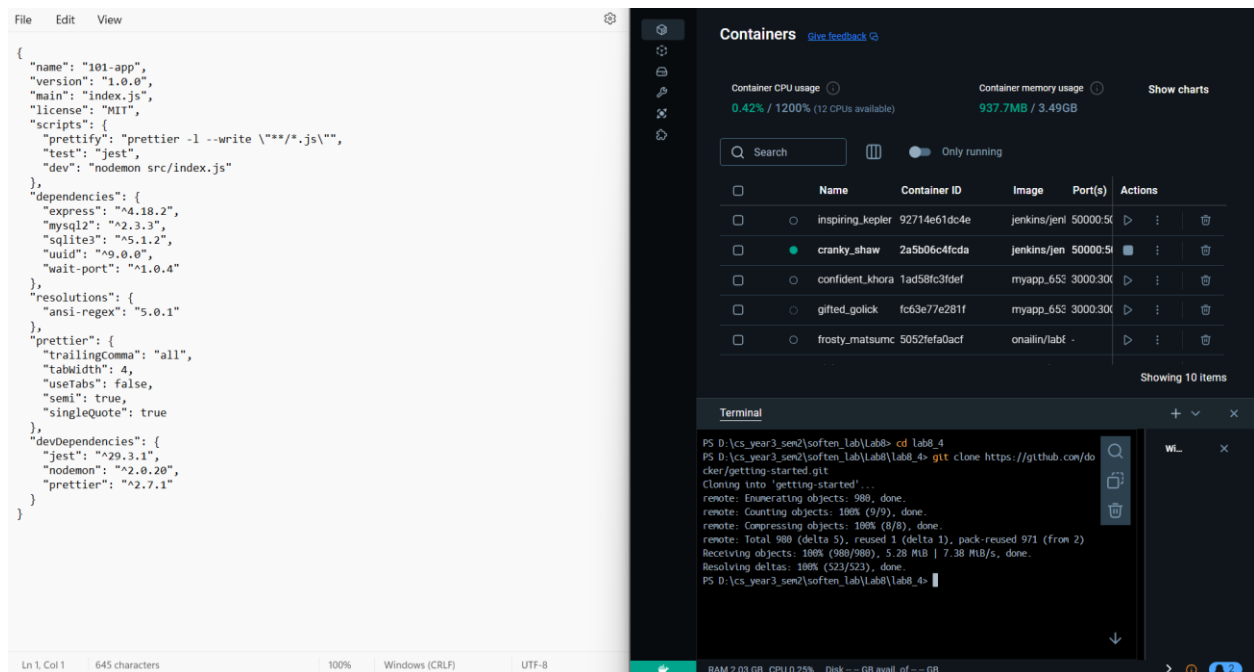


Lab Worksheet

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 พร้อมกับใส่เนื้อหาดังต่อไปนี้ลงไปไฟล์

```
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_รหัสสนศ. ไม่มีขีด
`$ docker build -t <myapp_รหัสสนศ. ไม่มีขีด> .`

Lab Worksheet

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

```

PS D:\cs_year3_sen2\soften_lab\Lab8\Lab8_4\getting-started\app> docker build -t myapp_6533880298 -f Dockerfile.swp .
[+] Building 34.7s (10/10) FINISHED
=> [internal] load build definition from Dockerfile.swp
=> => transferring dockerfile: 188B
=> [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
=> => exporting to image
=> => exporting layers
=> => writing image sha256:a8d5a7b6ba49d68a615d9cc32e25f31663d9ef251d80654794610241fd1b5400
=> => naming to docker.io/library/myapp_6533880298

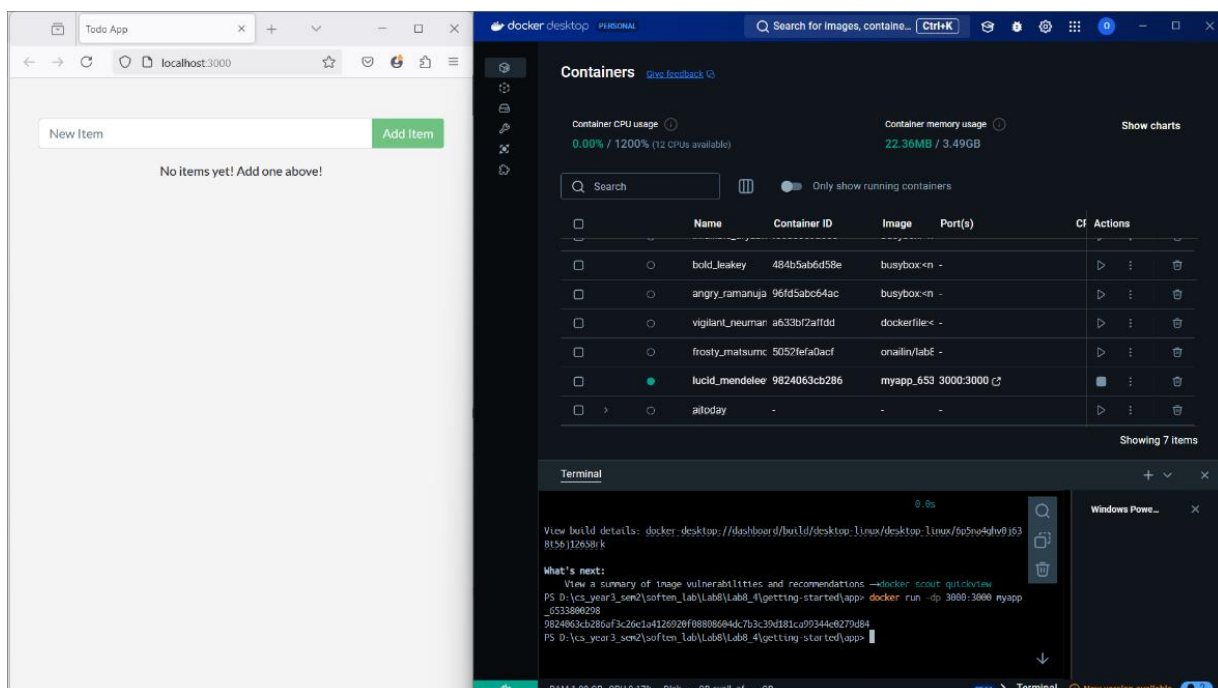
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/otuln73dlws5d1socc987ztzd

1 warning found (use docker --debug to expand):
- JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 5)

What's next:
View a summary of image vulnerabilities and recommendations -> docker scout quickview
PS D:\cs_year3_sen2\soften_lab\Lab8\Lab8_4\getting-started\app>
  
```

6. ทำการ Start ตัว Container ของแอปพลิเคชันที่สร้างขึ้น โดยใช้คำสั่ง
\$ docker run -dp 3000:3000 <myapp_รหัสสนศ. ไม่มีขีด>
7. เปิด Browser ไปที่ URL = <http://localhost:3000>
- 8.

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



Lab Worksheet

หมายเหตุ: นศ.สามารถทดลองเล่น Web application ที่ทำงานอยู่ได้

9. 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

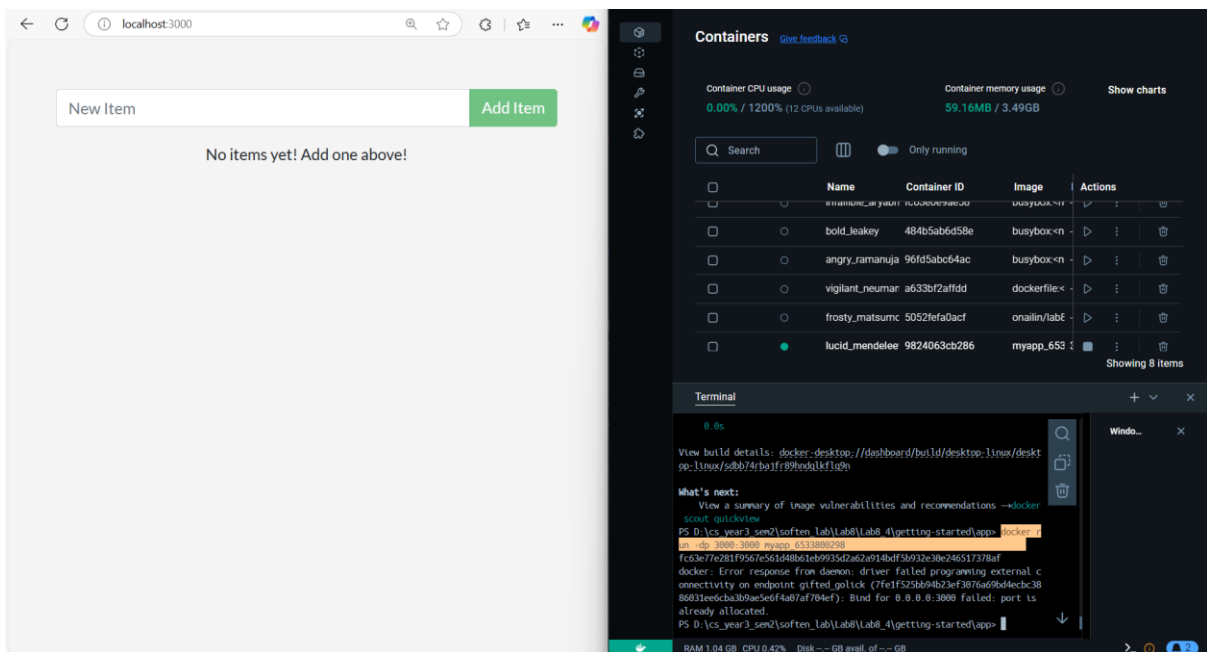
b. Save ไฟล์ให้เรียบร้อย

10. ทำการ Build Docker image โดยใช้คำสั่งเดียวกันกับข้อ 5

11. Start และรัน Container ตัวใหม่ โดยใช้คำสั่งเดียวกันกับข้อ 6

[Check point#10] Capture หน้าจอ (ทั้งหน้าต่างและทุกหน้าต่างที่เกี่ยวข้อง)

แสดงคำสั่งและผลลัพธ์ที่ได้ทางหน้าจอ พร้อมกับตอบคำถามต่อไปนี้



(1) Error ที่เกิดขึ้นหมายความว่าอย่างไร และเกิดขึ้นเพราะอะไร

Docker ไม่สามารถเชื่อมต่อหรือใช้งานพอร์ตได้ เพราะพอร์ต 3000 บนเครื่องถูกใช้งานแล้ว

12. ลบ Container ของ Web application เวอร์ชันก่อนแก้ไขออกจากระบบ โดยใช้วิธีใดวิธีหนึ่งดังต่อไปนี้

a. ผ่าน Command line interface

i. ใช้คำสั่ง `$ docker ps` เพื่อดู Container ID ที่ต้องการจะลบ

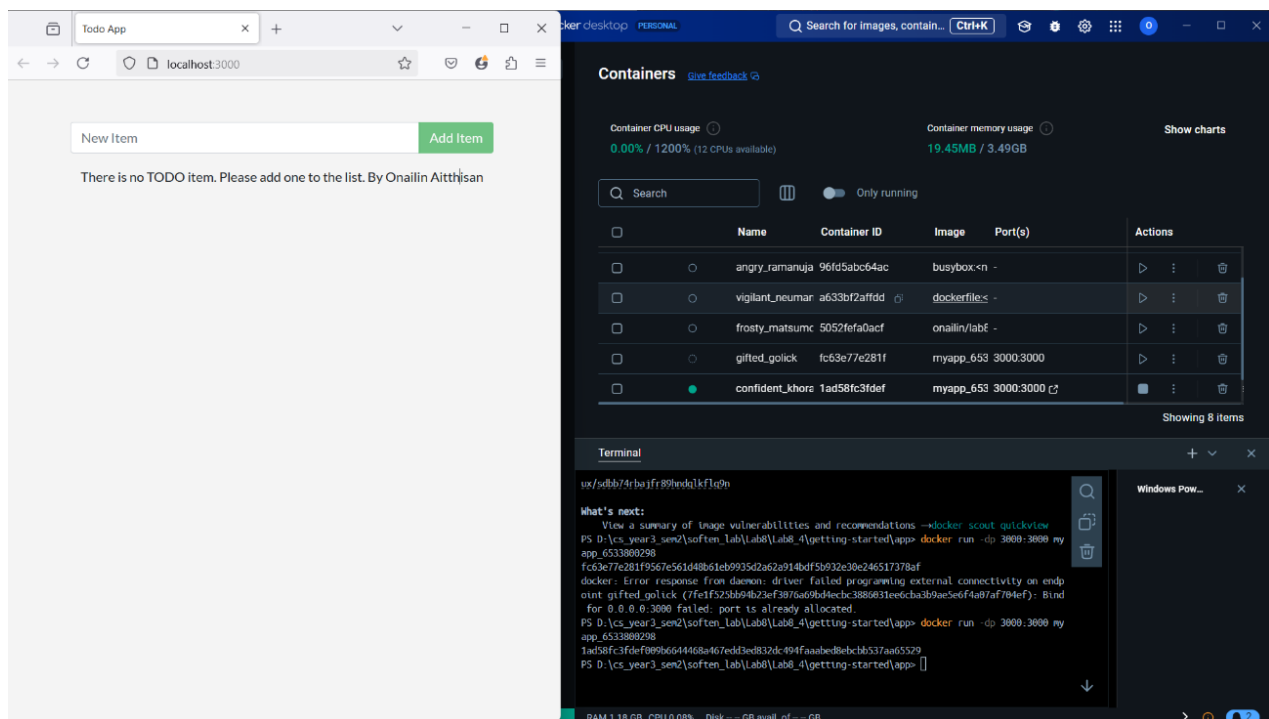
ii. Copy หรือบันทึก Container ID ไว้

Lab Worksheet

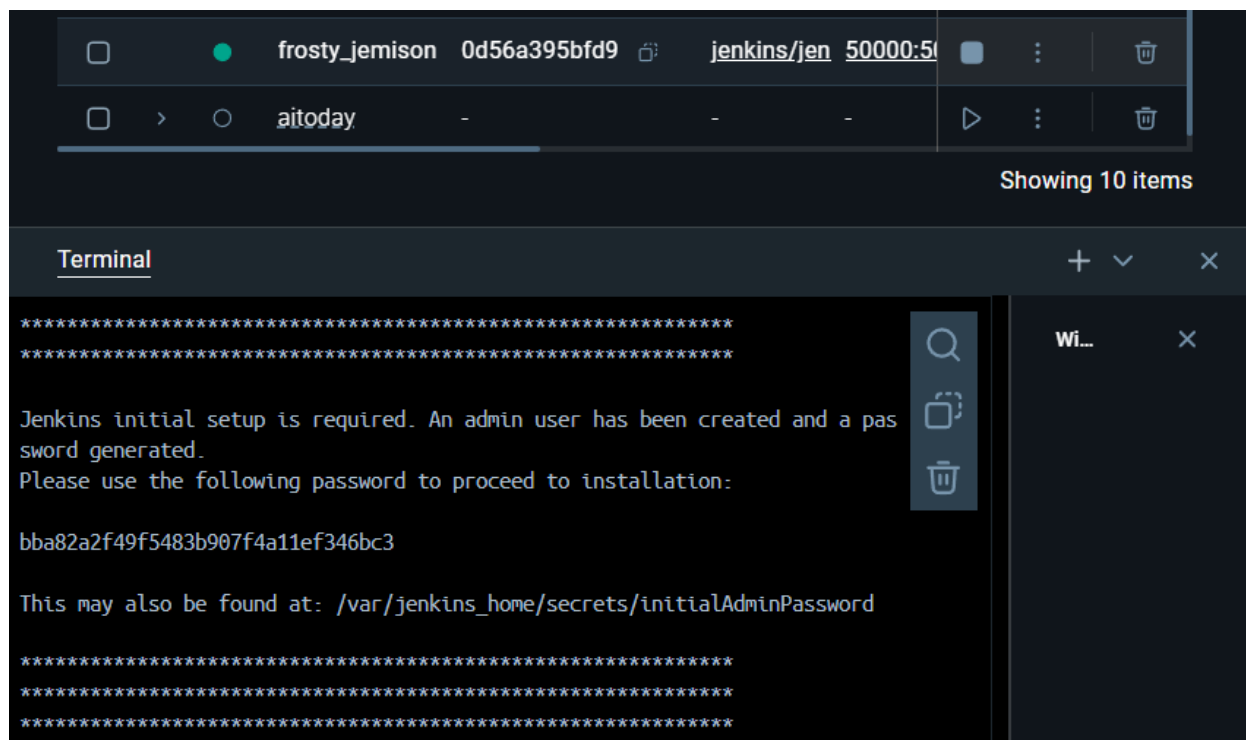
- iii. ใช้คำสั่ง `$ docker stop <Container ID ที่ต้องการจะลบ>` เพื่อหยุดการทำงานของ Container ดังกล่าว
 - iv. ใช้คำสั่ง `$ docker rm <Container ID ที่ต้องการจะลบ>` เพื่อทำการลบ
- b. ผ่าน Docker desktop
- i. ไปที่หน้าต่าง Containers
 - ii. เลือกไอคอนถังขยะในแถวของ Container ที่ต้องการจะลบ
 - iii. ยืนยันโดยการกด Delete forever
13. Start และรัน Container ตัวใหม่อีกครั้ง โดยใช้คำสั่งเดียวกันกับข้อ 6
14. เปิด Browser ไปที่ URL = <http://localhost:3000>

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

ลบโดยการคลิกไอคอน ผ่าน Docker desktop



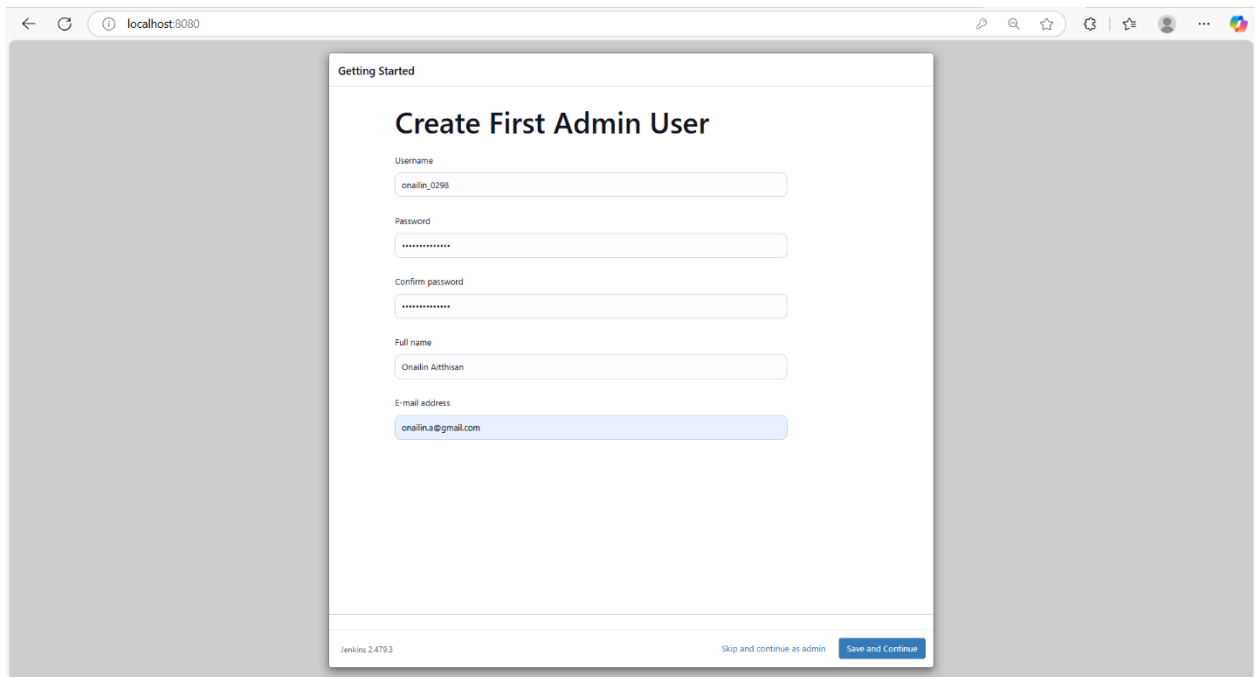
- [Check point#12] Capture หน้าจอที่แสดงผล Admin password



6. สร้าง Admin User โดยใช้ username เป็นชื่อจริงของนักศึกษาพร้อมรหัสสี่ตัวท้าย เช่น somsri 3062

Lab Worksheet

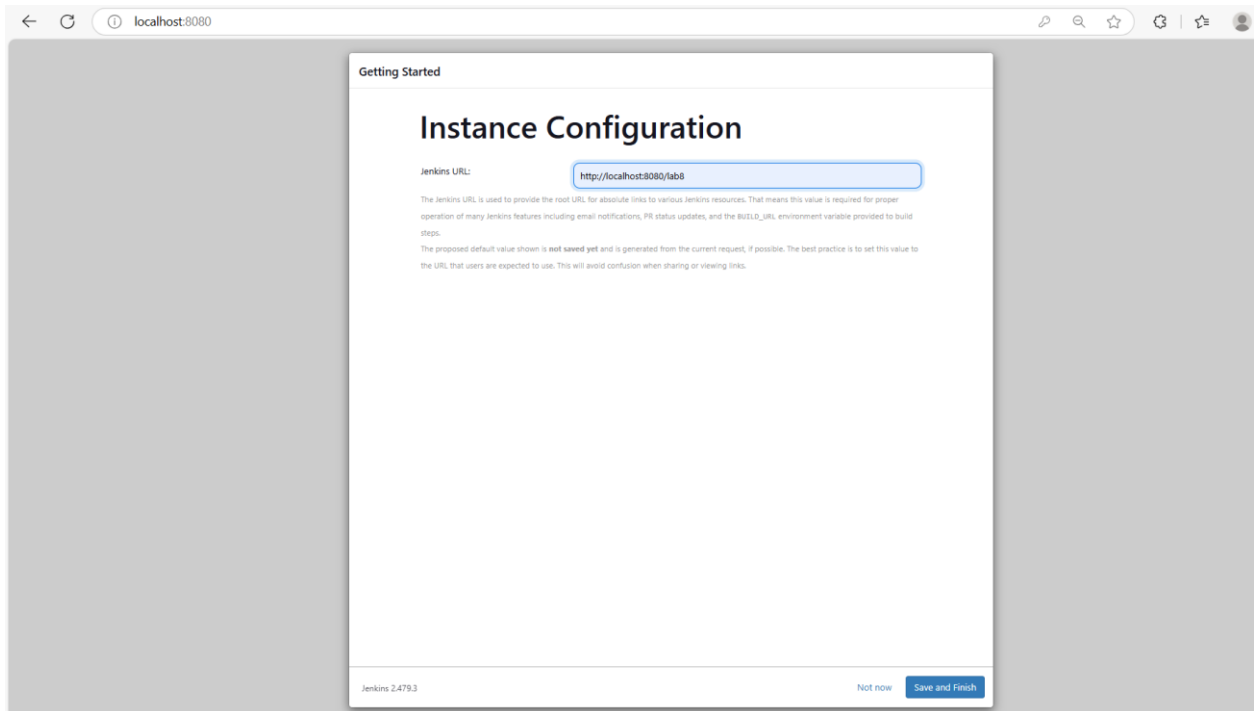
[Check point#13] Capture หน้าจอที่แสดงผลการตั้งค่า



The screenshot shows the Jenkins 'Getting Started' page with the 'Create First Admin User' form. The form contains the following fields and values:

- Username: onalin_0298
- Password: [masked]
- Confirm password: [masked]
- Full name: Onalin Atthisan
- E-mail address: onalina@gmail.com

At the bottom of the form, there is a 'Skip and continue as admin' link and a 'Save and Continue' button. The Jenkins version 2.479.3 is displayed in the bottom left corner.



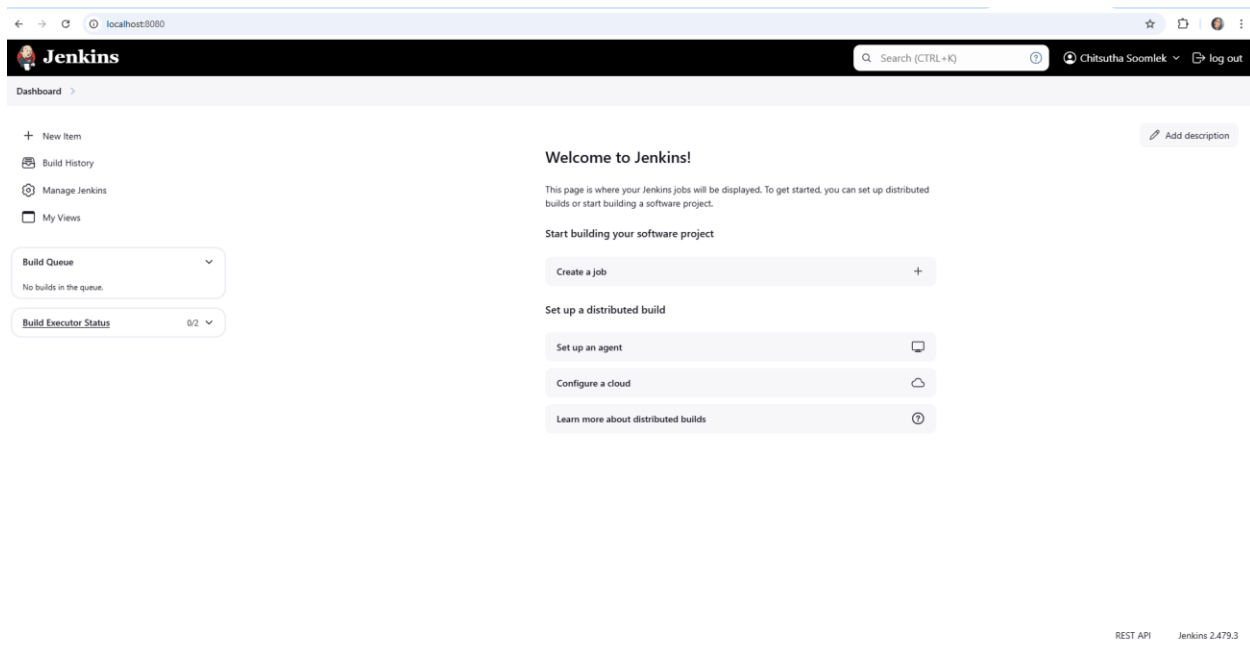
The screenshot shows the Jenkins 'Getting Started' page with the 'Instance Configuration' form. The form contains the following fields and values:

- Jenkins URL: http://localhost:8080/lab8

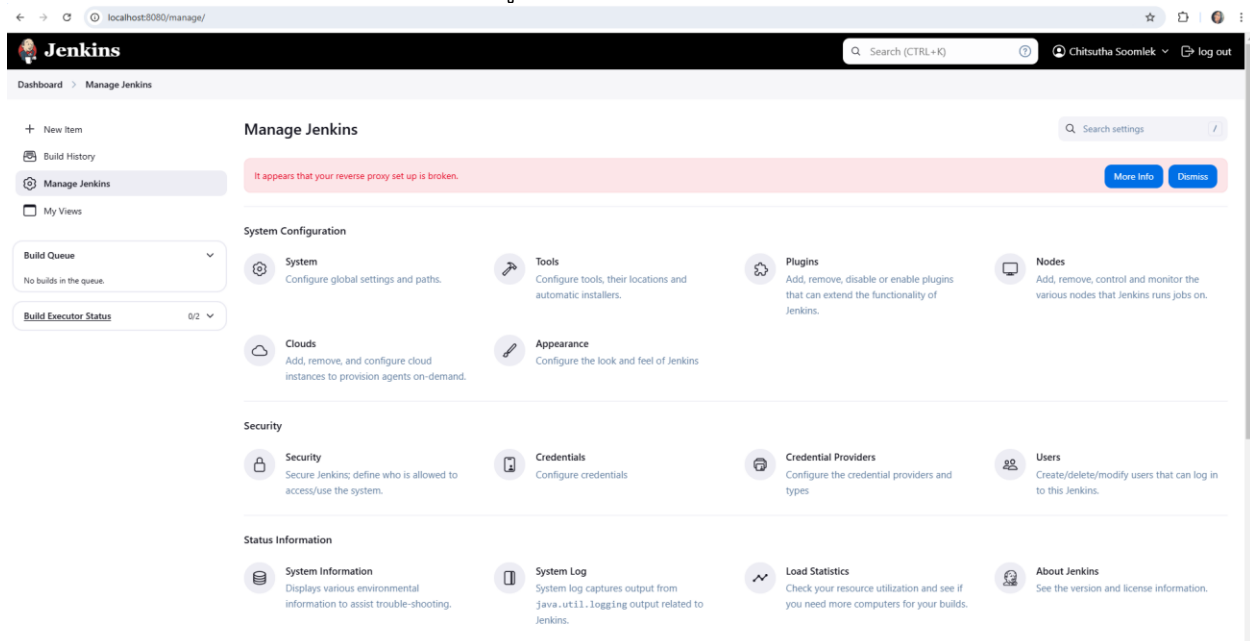
Below the URL field, there is explanatory text: 'The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps. The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.'

At the bottom of the form, there is a 'Not now' link and a 'Save and Finish' button. The Jenkins version 2.479.3 is displayed in the bottom left corner.

Lab Worksheet



9. เลือก Manage Jenkins แล้วไปที่เมนู Plugins

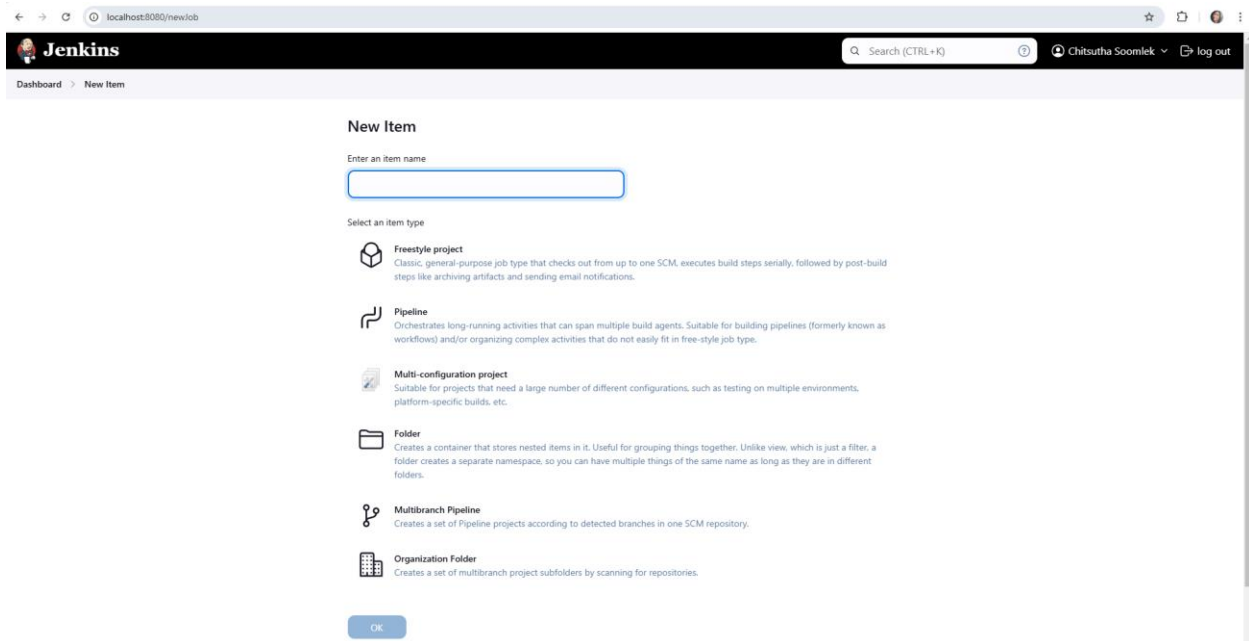


10. ไปที่เมนู Available plugins แล้วเลือกติดตั้ง Robotframework เพิ่มเติม

Lab Worksheet



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



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

Lab Worksheet

[Check point#14] Capture หน้าจอแสดงการตั้งค่า พร้อมกับตอบคำถามต่อไปนี้

Jenkins

Dashboard > All > New Item

New Item

Enter an item name

UAT

Select an item type

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

OK

REST API Jenkins 2.479.3

Jenkins

Dashboard > UAT > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

General

Enabled

Description

Lab8.5

Plain text [Preview](#)

☐ Discard old builds

☒ GitHub project

Project url

https://github.com/onainait/fab7_robotframework.git

Advanced

REST API Jenkins 2.479.3

Jenkins

Dashboard > UAT > Configuration

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Source Code Management

☐ None

☒ Git

Repositories

Repository URL

https://github.com/onainait/fab7_robotframework.git

Credentials

+ Add

Advanced

Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/main

Add Branch

Repository browser

(Auto)

Additional Behaviours

Add

REST API Jenkins 2.479.3

Lab Worksheet

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts) ?
- ☐ Build after other projects are built ?
- ☒ Build periodically ?

Schedule ?

H/15 * * * *

Would last have run at Tuesday, January 28, 2025 at 6:05:10 PM Coordinated Universal Time; would next run at Tuesday, January 28, 2025 at 6:20:10 PM Coordinated Universal Time.

- ☐ 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](#)

robot login_tests/testfrom.robot

Advanced ▾

Add build step ▾

Dashboard > UAT > Configuration

Configure

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

- ☐ 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

Filter

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Publish JUnit test result report
- Publish Robot Framework test results
- Record fingerprints of files to track usage
- Git Publisher
- E-mail Notification
- Editable Email Notification
- Set GitHub commit status (universal)
- Set build status on GitHub commit [deprecated]
- Delete workspace when build is done

Add post-build action ▾

Save

Apply

Lab Worksheet

- (1) คำสั่งที่ใช้ในการ Execute ไฟล์ .robot ใน Build Steps คือ
robot login_tests/testfrom.robot

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

Jenkins Dashboard > UAT > #1

Status ✖ #1 (Jan 28, 2025, 4:44:06 PM)

Started by user Onalín Altthian

This run spent:

- 0.1 sec waiting;
- 0.7 sec build duration;
- 9.8 sec total from scheduled to completion.

git Revision: 213d24f87b1b55c51e67131b029096fdefdbede
Repository: https://github.com/onalinait/lab7_robotframework.git
refs/remotes/origin/main

Robot Test Summary:

Total	Failed	Passed	Skipped	Pass %	
All tests	7	6	1	0	14.2

[Browse results](#)
[Open report.html](#)
[Open log.html](#)

No changes.

REST API Jenkins 2.479.3

Console Output

Started by user Onalín Altthian

Running as SYSTEM

Building in workspace /var/jenkins_home/workspace/UAT

The recommended git tool is: NONE

No credentials specified

Cloning the remote git repository

Cloning repository https://github.com/onalinait/lab7_robotframework.git

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

Fetching upstream changes from https://github.com/onalinait/lab7_robotframework.git

> git --version # timeout=10

> git fetch --tags --force --progress -- https://github.com/onalinait/lab7_robotframework.git # timeout=10

> git config remote.origin.url https://github.com/onalinait/lab7_robotframework.git # timeout=10

> git config --add remote.origin.fetch refs/heads/*:refs/remotes/origin/* # timeout=10

Avoid second fetch

> git rev-parse refs/remotes/origin/main^{commit} # timeout=10

Checking out Revision 213d24f87b1b55c51e67131b029096fdefdbede (refs/remotes/origin/main)

> git config core.sparsecheckout # timeout=10

> git checkout -f 213d24f87b1b55c51e67131b029096fdefdbede # timeout=10

Commit message: "Add files via upload"

First time build. Skipping changelog.

[UAT] \$ /bin/sh -xe /tmp/jenkins1046680823717236758.sh

+ robot login_tests/testfrom.robot

/tmp/jenkins1046680823717236758.sh 2: robot: not found

Build step 'Execute shell' marked build as failure

Robot results publisher started...

INFO: Checking test criticality is deprecated and will be dropped in a future release!

-Parsing output xml:

Done!

-Copying log files to build dir:

Done!

-Assigning results to build:

Done!

-Checking thresholds: