Lab : Infrastructure as Code (Dockerfile and Docker Compose)

Usa Sammapun, Kasetsart University

สำหรับนิสิตที่ใช้เครื่อง AWS ของอาจารย์ ให้ download Putty (https://www.putty.org/)

- และ log on to AWS ด้วย Putty ดังนี้
  - IP: 34.216.133.148
  - **username**: kaset
  - **password**: Sawadee35813

- จากนั้น ให้สร้าง folder ของตนเอง และเข้าไปใน Folder ของตน เช่น

$ mkdir your-name
$ cd your-name

Part 1 : Create Your Own Image

เราสามารถสร้าง image ของเราเองได้ โดยจะระบุการ set up environment ต่าง ๆ ใน Dockerfile

นิสิตสามารถใช้โปรเจค Restaurant ของตนเอง หรือใช้โปรเจค wisdom-book ของอาจารย์ก็ได้

- หากใช้ของตนเอง ให้เริ่มที่ข้อ 3)

1) fork this wisdom-book repo (หรือถ้าเคย fork มาแล้ว ใช้อันเคย fork มาได้ค่ะ)
เราจะสร้าง image ของตัวอย่างโปรแกรม Spring Boot (เป็นเว็บเพิ่มและแสดงข้อมูลหนังสือง่ายๆ) โดยนิสิตสามารถใช้ repository นี้

https://github.com/ladyusa/wisdom-book

2) clone ลงมาที่เครื่องของตนเอง หรือเครื่อง AWS ด้วยคำสั่งต่อไปนี้

$ git clone https://github.com/YOURUSERNAME/wisdom-book.git

หรือ clone จากของอาจารย์โดยตรง

$ git clone https://github.com/ladyusa/wisdom-book.git

และให้เข้าไปใน directory โค้ด wisdom-book
$ cd wisdom-book

3) สร้างไฟล์ jar ของโปรเจคด้วยคำสั่งต่อไปนี้

$ mvn package -DskipTests

(หากใช้เครื่องตนเอง ต้อง install โปรแกรม mvn จากลิงก์นี้ https://maven.apache.org/ เสียก่อน)

หลังรันคำสั่ง จะเห็นว่า ในโฟลเดอร์ /target มีการสร้างไฟล์ .jar เพิ่มขึ้นมา

4) สร้างไฟล์ชื่อ Dockerfile ในโฟลเดอร์ปัจจุบัน เพื่อสร้าง images

- ถ้าใช้ server อาจารย์ ให้ใช้โปรแกรม nano
- ถ้าใช้เครื่องตนเอง ใช้ editor ใดก็ได้
- นิสิตสามารถเปลี่ยนชื่อไฟล์ book.jar / restaurant.jar เป็นชื่ออื่นได้ แต่อย่าลืมแก้ชื่อให้ครบ

$ nano Dockerfile

Wisdom-book (ถ้า clone จาก repo อาจารย์โดยตรง)

- เปลี่ยน java version, jar, port ให้ตรงกับของนิสิต

```
FROM openjdk:11-jdk-slim
COPY target/book-1.0.jar book.jar
EXPOSE 8090
ENTRYPOINT ["java","-jar","/book.jar"]
```

Wisdom-book (ถ้าใช้โค้ดจาก repo นิสิตจากแลป selenium)

```
FROM openjdk:11-jdk-slim
COPY target/book-0.0.1-SNAPSHOT.jar book.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar","/book.jar"]
```

restaurant

- เปลี่ยน java version, jar, port ให้ตรงกับของนิสิต

```
FROM openjdk:11-jdk-slim
COPY target/restaurant-0.0.1-SNAPSHOT.jar restaurant.jar
EXPOSE 8091
ENTRYPOINT ["java","-jar","/restaurant.jar"]
```

ออกจากโปรแกรม nano โดยการพิมพ์ ^X แล้วพิมพ์ Y เพื่อ save
จากนั้น โปรแกรมจะแถมว่าจะ save file ชื่อใด ให้พิมพ์ Enter เฉยๆ

โดย Dockerfile สามารถใช้คำสั่งต่อไปนี้

- FROM - what to use as a base image/system
- COPY - copy from this directory to the container directory

- WORKDIR - define the working directory of a Docker container
- RUN - bash command that runs on system (at build time)
- ENV - create environment variables
- EXPOSE - which ports open on the container
- ENTRYPOINT - run this command in container (at run time)

5) สร้าง docker image จาก Dockerfile ใน dir นี้ (มี . ต่อท้าย command ด้วย)

$ docker build -t image-name-เช่น-book-ชื่อนิสิต .

$ docker images

6) ทดลองใช้งาน โดยใช้ random port

$ docker run --name container-name -P -d image-name

7) see logs of running containers ว่ากำลังทำอะไรอยู่

$ docker logs container-name

8) ดูว่า random port คือ port อะไร

$ docker ps

ลองเข้าไปดูเว็บของเราผ่าน browser โดยใช้ random port ที่ได้ เช่น
http://34.216.133.148:49153/

9) สามารถ build image โดยตรงจาก Dockerfile บน GitHub ได้ด้วย

Add and commit Dockerfile เข้าไปใน repo แล้วใช้คำสั่งต่อไปนี้ (ข้างล่างเครื่องหมาย : จะเป็นหมายเลขเวอร์ชัน)

$ docker build https://github.com/<username>/<repo-name>.git -t <image-name>:latest
$ docker build https://github.com/<username>/<repo-name>.git -t <image-name>:<version>

10) สร้างบัญชีที่ Docker Hub ตามลิงก์ต่อไปนี้ และ tag ด้วยชื่อบัญชีของเรา
https://hub.docker.com/

$ docker tag image-name เว้นวรรค ชื่อบัญชีบน-docker-hub/image-name

เช่น
$ docker tag book ladyusa/book

11) push image ขึ้น Docker Hub (might take some time)
$ docker login
$ docker push ชื่อบัญชีบน-docker-hub/image-name

เช่น
$ docker push ladyusa/book

12) pull image ของเราลงมา หรือ รันเลยก็ได้
$ docker image pull <ชื่อบัญชีบน docker hub>/<image-name>

$ docker run --name <container-name> -P -d <ชื่อบัญชีบน docker hub>/<image-name>

Part 2: Docker Compose

“Compose is a tool for defining and running multi-container Docker applications”
Ref: https://docs.docker.com/compose/

นิสิตสามารถใช้โปรเจค restaurant ของตนเอง หรือใช้โปรเจค bankaccount-api ของอาจารย์ในการเรียนรู้การสร้าง docker compose

- หากใช้ของตนเอง ให้เริ่มที่ข้อ 3)

1) fork this repo
https://github.com/ladyusa/bankaccount-api-docker

2) clone your repo into your local host
$ git clone https://github.com/USERNAME/bankaccount-api-docker.git
$ cd bankaccount-api-docker

3) สร้างไฟล์ jar ของโปรเจคด้วยคำสั่งต่อไปนี้
$ mvn package -DskipTests

4) look at files
- bankaccount-api.dockerfile
- docker-compose.yml

หากนิสิตใช้โปรเจค restaurant api ของตนเอง ให้สร้างไฟล์ต่อไปนี้ และปรับโค้ดจากของ bankaccount-api
- restaurant-api.dockerfile

- docker-compose.yml

bankaccount-api.dockerfile

```
FROM openjdk:8-jdk-slim
COPY target/bankaccount-1.0.0.jar bankaccount-api.jar
EXPOSE 8091
ENTRYPOINT ["java","-jar","/bankaccount-api.jar"]
```

docker-compose.yml

- ให้นิสิตที่ใช้

```yaml
version: '3'

services:

 docker-mysql:
   container_name: docker-mysql
   image: mysql:latest
   environment:
     MYSQL_DATABASE: bank
     MYSQL_ROOT_PASSWORD: abc123
   expose:
     - 3306
   ports:
     - 3307:3306
   networks:
     - bank-network
   restart: always

 bankaccount-api:
   restart: on-failure
   container_name: bankaccount-api
   build:
     context: .
     dockerfile: bankaccount-api.dockerfile
   image: bankaccount-api:latest
   expose:
     - 8091
   ports:
     - 8091:8091
   networks:
     - bank-network
   environment:
     SPRING_DATASOURCE_URL: jdbc:mysql://docker-mysql:3306/bank
   depends_on:
     - docker-mysql

networks:
 bank-network:
   driver: bridge
```

5) build all images in .yml file and also run them
$ docker-compose up -d

6) check to see our running containers
$ docker-compose ps

7) see logs of running containers
$ docker logs <container-name>

Open web browser to port 8091 หรือ port ที่ได้แก้ไป
http://localhost:8091/api/bankaccount

http://34.216.133.148:8091/api/bankaccount

http://localhost:8090/restaurant

8) stop all containers
$ docker-compose down

ตัวอย่าง docker compose สำหรับ wisdom-book

https://github.com/ladyusa/wisdom-book-docker-compose