Docker module practical task 1

Practical task 1

Follow the steps below and practice with Dockerfile and docker-compose configuration.

- Fork GitHub - spring-projects/spring-petclinic: A sample Spring-based application
- Create Dockerfile for Spring-petclinic application using pre-built artifact
  - Build application outside of container
  - Copy artifact from target folder into image and make it work inside container
- Create multi-stage Dockerfile for Spring-petclinic application
  - Application should be built as part of first stage
  - Final image should contain only required files and based on minimal possible base image
- Create docker-compose configuration that will automatically start multiple containers
  - Run two containers: application + DB
  - Provide credentials as environment variables, so DB image can be configured with custom credentials and application can connect to DB automatically

1. Fork and clone github repo

```
02:32:21 adrwal@olek-desktop-pc docker-module-tasks → git clone git@github.com:adrwalGD/spring-petclinic.git
Cloning into 'spring-petclinic'...
Enter passphrase for key '/home/adrwal/.ssh/id_ed25519':
remote: Enumerating objects: 10256, done.
Receiving objects:  10% (1084/10256), 380.00 KiB | 334.00 KiB/s
```

2. Create Dockerfile for Spring-petclinic application using pre-built artifact

First we build our java project using `mvn package` locally on our system.

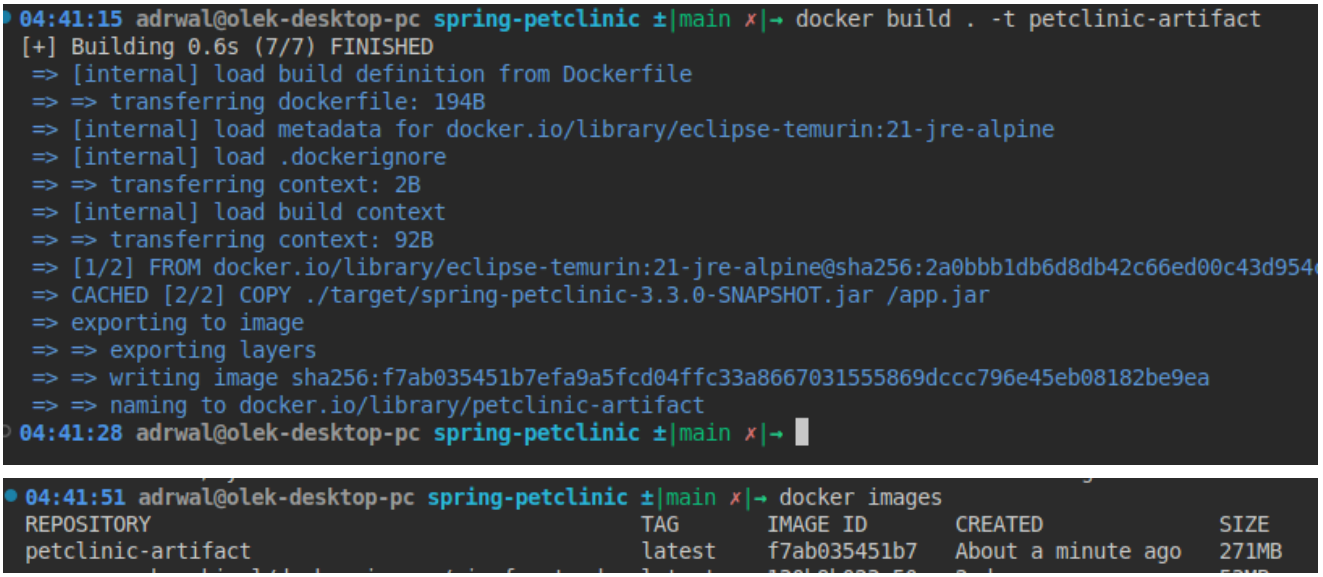
```
[INFO] Replacing main artifact /home/adrwal/praca-grid/docker-module-tasks/spring-petclinic/target/spring-petclinic-3.3.0-SNAPSHOT.jar
with repackaged archive, adding nested dependencies in BOOT-INF/.
[INFO] The original artifact has been renamed to /home/adrwal/praca-grid/docker-module-tasks/spring-petclinic/target/spring-petclinic-3
.3.0-SNAPSHOT.jar.original
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:57 min
[INFO] Finished at: 2024-11-19T14:35:57+01:00
[INFO] ------------------------------------------------------------------------
02:35:57 adrwal@olek-desktop-pc spring-petclinic ±|main ✗|→
```

Built `.jar` artifact is placed inside `./target` directory. We can run it directly using `java`:

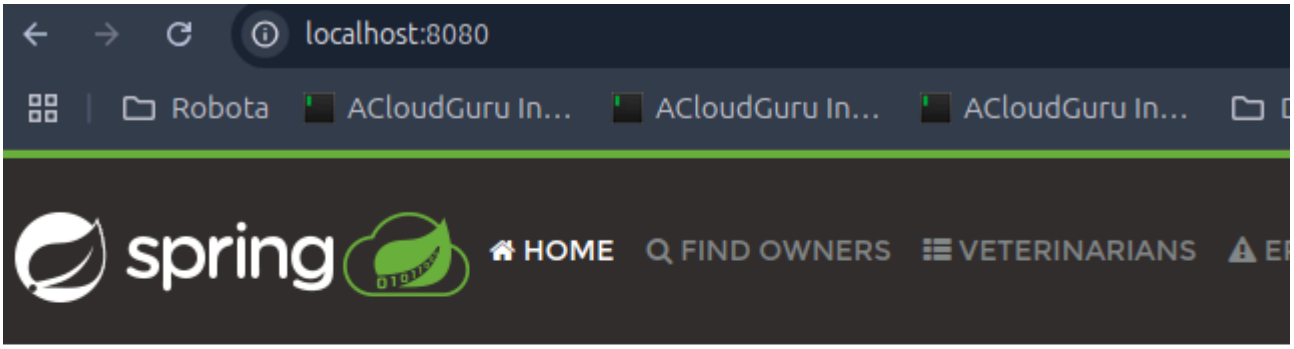
Now we create Dockerfile to copy and run artifact. We have already built our application so we only need JRE.

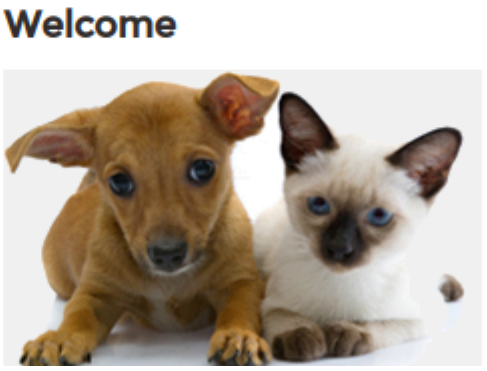
```
FROM eclipse-temurin:21-jre-alpine

COPY ./target/spring-petclinic-3.3.0-SNAPSHOT.jar /app.jar
CMD ["java", "-jar", "app.jar"]

```

Now we build this image.



Finally we can run the container and map our local port 8080 to container port 8080 to access it on local network, outside of the container.





3. Create multi-stage Dockerfile for Spring-petclinic application

First we need to update our `Dockerfile`

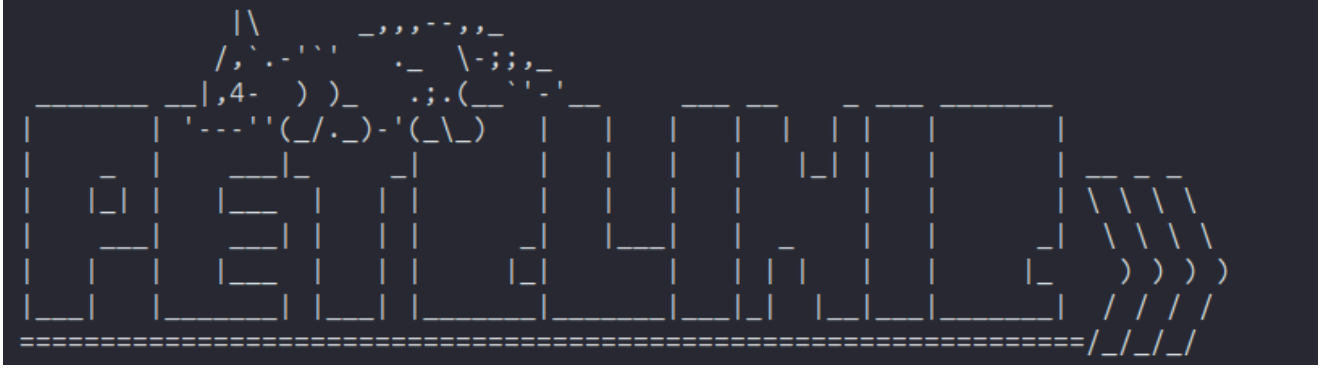
```
FROM maven:3.8.7-openjdk-18-slim AS build

RUN mkdir /app
COPY . /app
WORKDIR /app
RUN mvn package

# Minimal rintime image - only JRE
FROM gcr.io/distroless/java21-debian12 AS runtime
COPY --from=build /app/target/*.jar /app.jar
ENTRYPOINT [ "java" ]
CMD [ "-jar", "/app.jar" ]
```

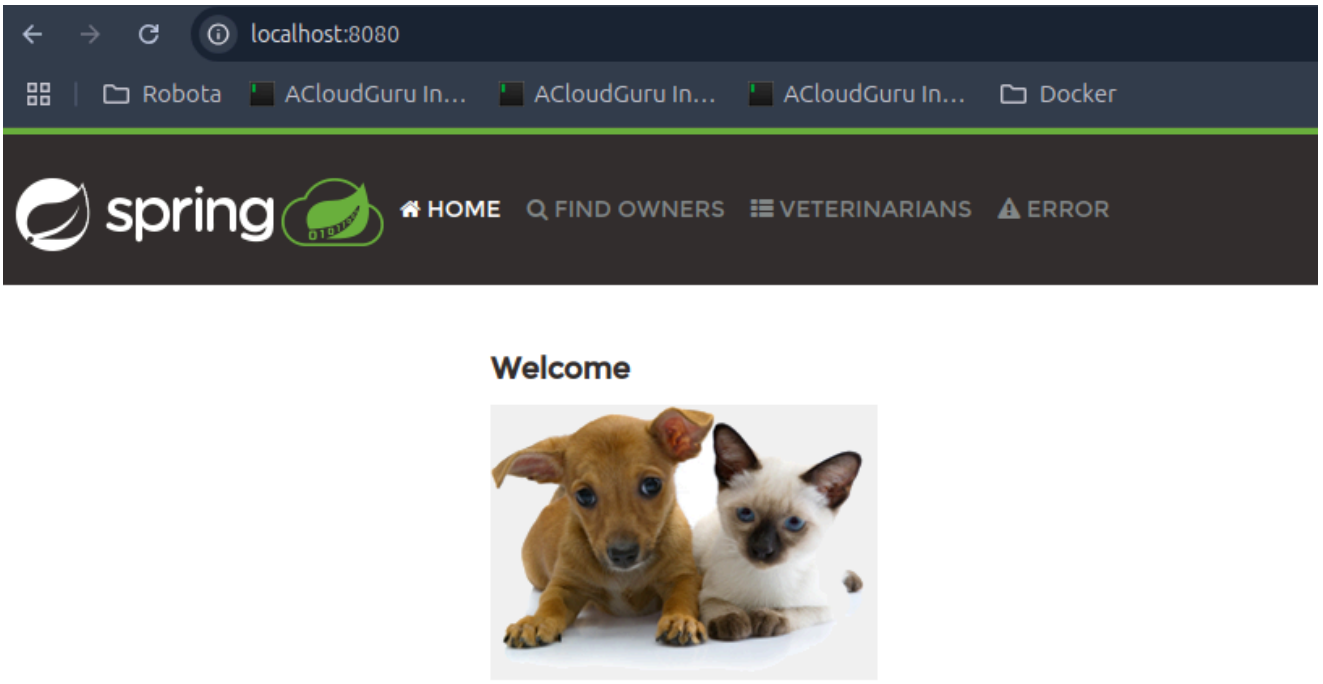
We have added two build stages. First `build` builds the package with maven, and second `runtime` copies only `.jar` artifact from previous stage and runs it in minimal environment.

We can build the image and run the container.

```
10:48:01 adrwal@olek-desktop-pc spring-petclinic ±|main ✗|→ docker build . -t petclinic-better
[+] Building 294.2s (13/13) FINISHED                                docker:rootless
 => [internal] load build definition from Dockerfile                           0.0s
 => => transferring dockerfile: 283B                                           0.0s
 => [internal] load metadata for gcr.io/distroless/java21-debian12:late        0.3s
 => [internal] load metadata for docker.io/library/maven:3.8.7-openjdk-        0.6s
 => [internal] load .dockerignore                                              0.0s
 => => transferring context: 2B                                                0.0s
 => [internal] load build context                                              0.2s
 => => transferring context: 64.58MB                                           0.2s
 => [build 1/5] FROM docker.io/library/maven:3.8.7-openjdk-18-slim@sha2        0.0s
 => CACHED [runtime 1/2] FROM gcr.io/distroless/java21-debian12:latest@        0.0s
 => CACHED [build 2/5] RUN mkdir /app                                          0.0s
 => [build 3/5] COPY . /app                                                    0.4s
 => [build 4/5] WORKDIR /app                                                   0.0s
 => [build 5/5] RUN mvn package                                              292.2s
 => [runtime 2/2] COPY --from=build /app/target/*.jar /app.jar                 0.2s
 => exporting to image                                                         0.2s
 => => exporting layers                                                        0.1s
 => => writing image sha256:bae0222a7f28f84d27f647e591e2780bd60bad291f5        0.0s
 => => naming to docker.io/library/petclinic-better                            0.0s
10:52:56 adrwal@olek-desktop-pc spring-petclinic ±|main ✗|→
```

```
10:59:27 adrwal@olek-desktop-pc ~ → docker run -p 8080:8080 -it petclinic-better
```





Verify that our container only contains JRE:

```
11:01:41 adrwal@olek-desktop-pc ~ → docker exec -it eager_archimedes java --version
openjdk 21.0.5 2024-10-15 LTS
OpenJDK Runtime Environment Temurin-21.0.5+11 (build 21.0.5+11-LTS)
OpenJDK 64-Bit Server VM Temurin-21.0.5+11 (build 21.0.5+11-LTS, mixed mode, sharing)
11:05:51 adrwal@olek-desktop-pc ~ → docker exec -it eager_archimedes sh
OCI runtime exec failed: exec failed: unable to start container process: exec: "sh": executable file not found in $PATH: unknown
11:05:58 adrwal@olek-desktop-pc ~ → docker exec -it eager_archimedes bash
OCI runtime exec failed: exec failed: unable to start container process: exec: "bash": executable file not found in $PATH: unknown
11:06:01 adrwal@olek-desktop-pc ~ → docker exec -it eager_archimedes 'echo $PATH'
OCI runtime exec failed: exec failed: unable to start container process: exec: "echo $PATH": executable file not found in $PATH: unkno
11:06:46 adrwal@olek-desktop-pc ~ → docker exec -it eager_archimedes maven --version
OCI runtime exec failed: exec failed: unable to start container process: exec: "maven": executable file not found in $PATH: unknown
11:07:40 adrwal@olek-desktop-pc ~ →
```

4. Create docker-compose configuration that will automatically start multiple containers

First create `docker-compose.yml`

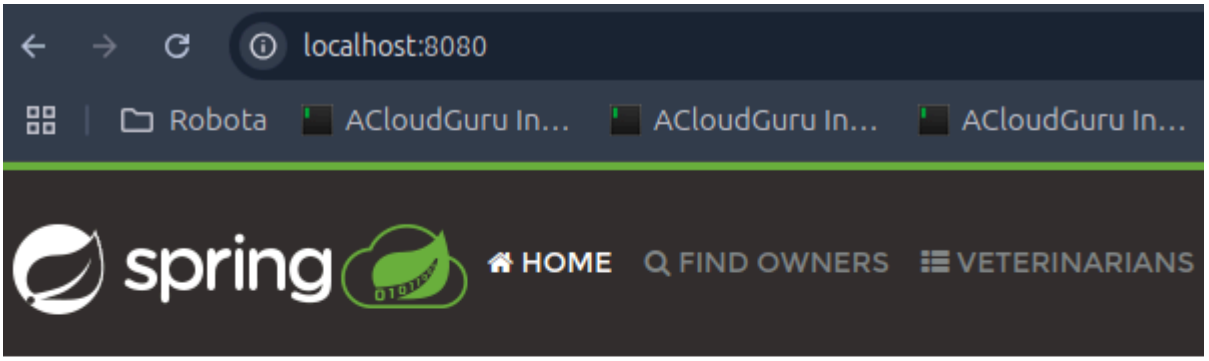
```yaml
services:
  server:
    build:
      context: .
    ports:
      - 8080:8080
    environment:
      - POSTGRES_URL=jdbc:postgresql://db:5432/petclinic
    command: ["-jar", "-Dspring.profiles.active=postgres", "/app.jar"]
  db:
    image: postgres:17.1
    restart: always
    volumes:
      - db-data:/var/lib/postgresql/data
    environment:
      - POSTGRES_DB=petclinic
      - POSTGRES_USER=petclinic
      - POSTGRES_PASSWORD=petclinic
    ports:
      - 5432:5432
volumes:
  db-data:
```

In there we run two containers. One to run postgres database based on image pulled from docker hub, and the other one to run our java application based on our own Dockerfile from earlier step. We have also changed CMD, adding one argument to tell spring to use postgres. For postgres we have set environment variables to match those in spring-petclinic default configuration.

Run our configuration:

```
⊗ 11:30:25 adrwal@olek-desktop-pc spring-petclinic ±|main ✗|→ docker compose up --build
 [+] Building 283.8s (14/14) FINISHED
  => [server internal] load build definition from Dockerfile
  => => transferring dockerfile: 355B
  => [server internal] load metadata for gcr.io/distroless/java21-debian12:latest
  => [server internal] load metadata for docker.io/library/maven:3.8.7-openjdk-18-slim
  => [server internal] load .dockerignore
  => => transferring context: 2B
  => [server runtime 1/2] FROM gcr.io/distroless/java21-debian12:latest@sha256:75bff39aa6eaaa759db7b4
  => [server build 1/5] FROM docker.io/library/maven:3.8.7-openjdk-18-slim@sha256:0ccb246803384595673
  => [server internal] load build context
  => => transferring context: 37.79kB
  => CACHED [server build 2/5] RUN mkdir /app
  => [server build 3/5] COPY . /app
  => [server build 4/5] WORKDIR /app
```

```
db-1      | PostgreSQL Database directory appears to contain a database; Skipping initialization
db-1      |
db-1      | 2024-11-20 10:35:13.484 UTC [1] LOG:  starting PostgreSQL 17.1 (Debian 17.1-1.pgdg120+1
bian 12.2.0-14) 12.2.0, 64-bit
db-1      | 2024-11-20 10:35:13.484 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
db-1      | 2024-11-20 10:35:13.484 UTC [1] LOG:  listening on IPv6 address "::", port 5432
db-1      | 2024-11-20 10:35:13.487 UTC [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.
db-1      | 2024-11-20 10:35:13.493 UTC [29] LOG:  database system was shut down at 2024-11-20 10:3
db-1      | 2024-11-20 10:35:13.500 UTC [1] LOG:  database system is ready to accept connections
server-1  |
server-1  |
server-1  |              |\      _,,,--,,_
server-1  |             /,`.-'`'    ._  \-;;,_
server-1  |  _______ __|,4-  ) )_   .;.(__`'-'__     ___ __    _ ___ _______
server-1  | |       | '---''(_/._)-'(_\_)   |   |   |   |  |  | |   |       |
```



Welcome