DOCKER

LAB ASSIGNMENTS

Pre-Requisitions:

1. Create one account with **GitHub** (https://github.com/)
2. JDK **1.6 or higher**
3. Install **Spring Tool Suite 3.6** or higher version
4. Install Maven in your Machine. Please refer below link for to download Maven.

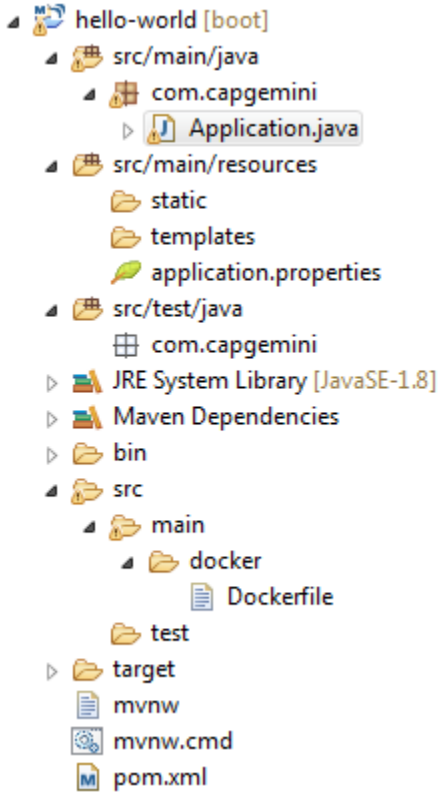
    a. http://maven.apache.org/download.cgi
5. Install **"Docker Toolbox-1.12.5"** , use the below URL to install docker:
    a. https://github.com/docker/toolbox/releases/tag/v1.12.5
6. Create one account with https://hub.docker.com .
    a. You will have one user id and password after successfully created the account in docker hub.

Lab 1:

Write simple hello world application in Spring Boot and dockerize your Hello World application.

Steps:

1. Open STS and create new Spring Start project. Name your project as **hello-world** as mentioned below:



2. Add the following code in Application.java

<u>Application.java</u>

```
package com.capgemini;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.bind.RelaxedPropertyResolver;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
```

```
@RestController
public class Application {



        @RequestMapping("/")
  public String home() {
    return "Hello Docker World";
  }
        public static void main(String[] args) {
                SpringApplication.run(Application.class, args);
        }

}
```

3. Add the following plugin in pom.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
        <modelVersion>4.0.0</modelVersion>

        <groupId>com.capgemini</groupId>
        <artifactId>hello-world</artifactId>
        <version>0.0.1-SNAPSHOT</version>
        <packaging>jar</packaging>

        <name>hello-world</name>
        <description>Hello World</description>

        <parent>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-parent</artifactId>
                <version>1.4.5.RELEASE</version>
                <relativePath/> <!-- lookup parent from repository -->
        </parent>
```

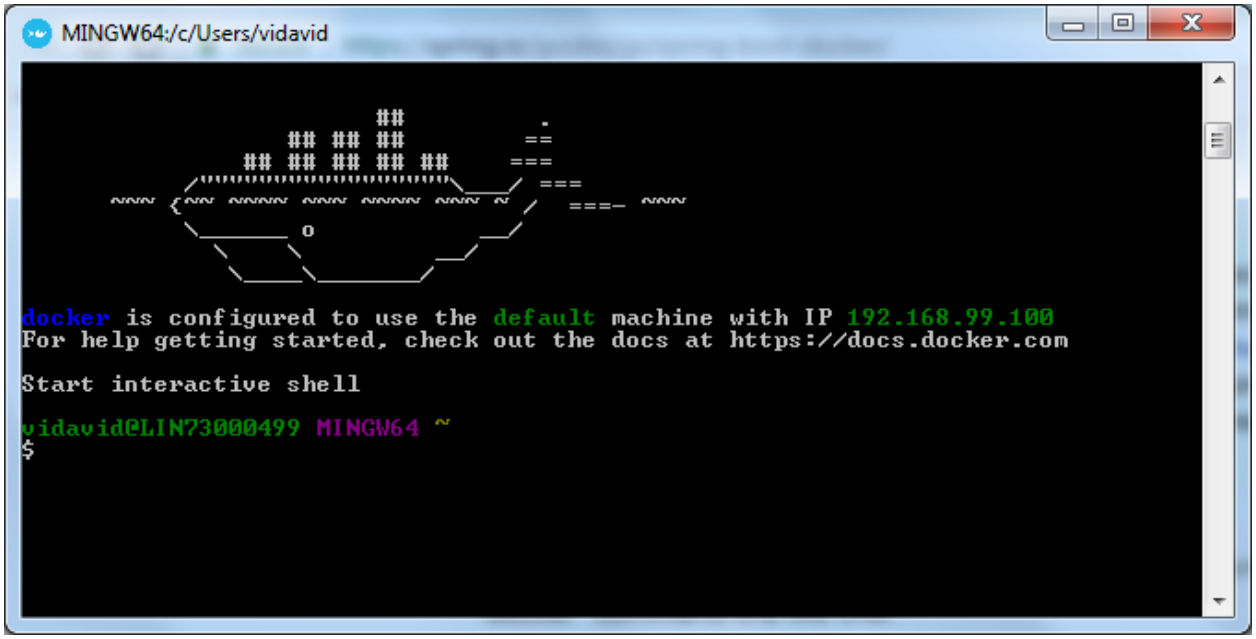
```
        <properties>
                <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
                <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
                <java.version>1.8</java.version>
<docker.image.prefix>caprepo</docker.image.prefix>
        </properties>

        <dependencies>
                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-web</artifactId>
                </dependency>

                <dependency>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-starter-test</artifactId>
                        <scope>test</scope>
                </dependency>
        </dependencies>

        <build>
                <plugins>
                        <plugin>
                                <groupId>org.springframework.boot</groupId>
                                <artifactId>spring-boot-maven-plugin</artifactId>
                        </plugin>


                        <plugin>
        <groupId>com.spotify</groupId>
        <artifactId>docker-maven-plugin</artifactId>
        <version>0.4.11</version>
        <configuration>
          <imageName>${docker.image.prefix}/${project.artifactId}</imageName>
          <dockerDirectory>src/main/docker</dockerDirectory>
          <resources>
            <resource>
```

```
                <targetPath>/</targetPath>
                <directory>${project.build.directory}</directory>
                <include>${project.build.finalName}.jar</include>
            </resource>
        </resources>
    </configuration>
  </plugin>
                </plugins>
        </build>


</project>
```

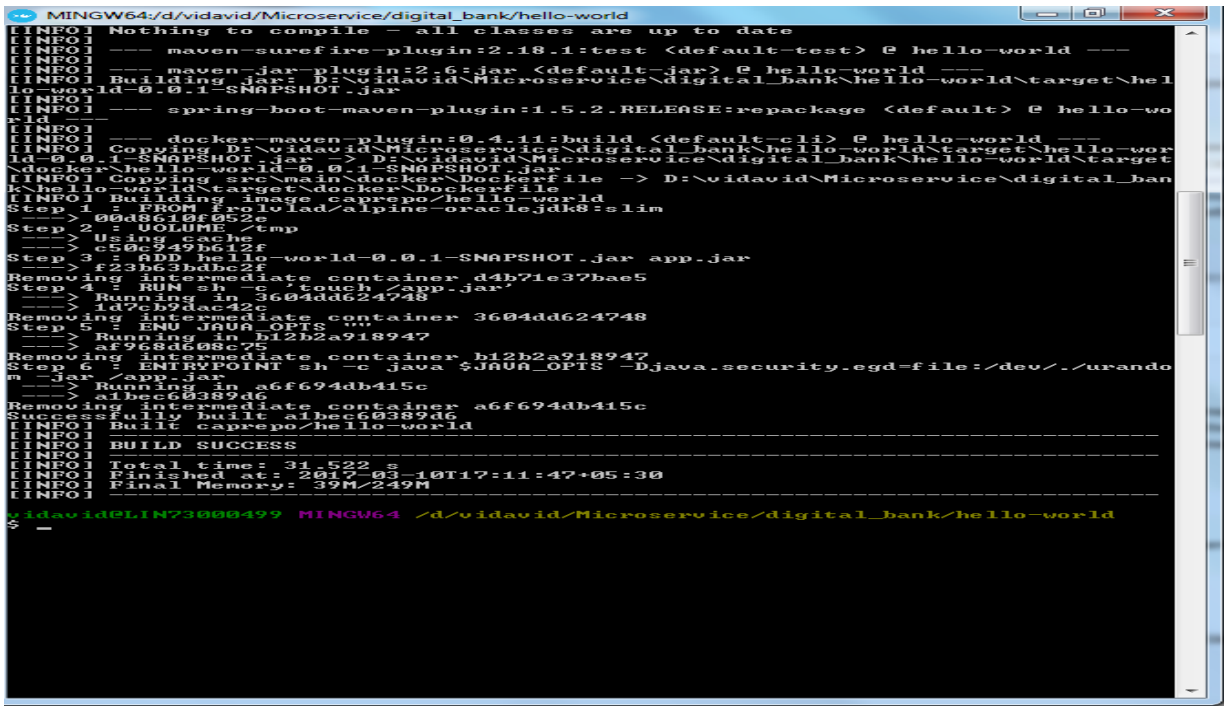
4. Add a Dockerfile under src/main/docker folder. Dockerfile should contain the following instructions:

```
FROM frolvlad/alpine-oraclejdk8:slim
VOLUME /tmp
ADD hello-world-0.0.1-SNAPSHOT.jar app.jar
RUN sh -c 'touch /app.jar'
ENV JAVA_OPTS=""
ENTRYPOINT [ "sh", "-c", "java $JAVA_OPTS -Djava.security.egd=file:/dev/./urandom -jar
/app.jar" ]
```

5. Open your docker quick start terminal window. It will open with IP Address



6. Change the current directory to your project root directory.
7. Enter the following command on the terminal:
   a. **mvn package docker:build**
8. You will be getting the command called build success:



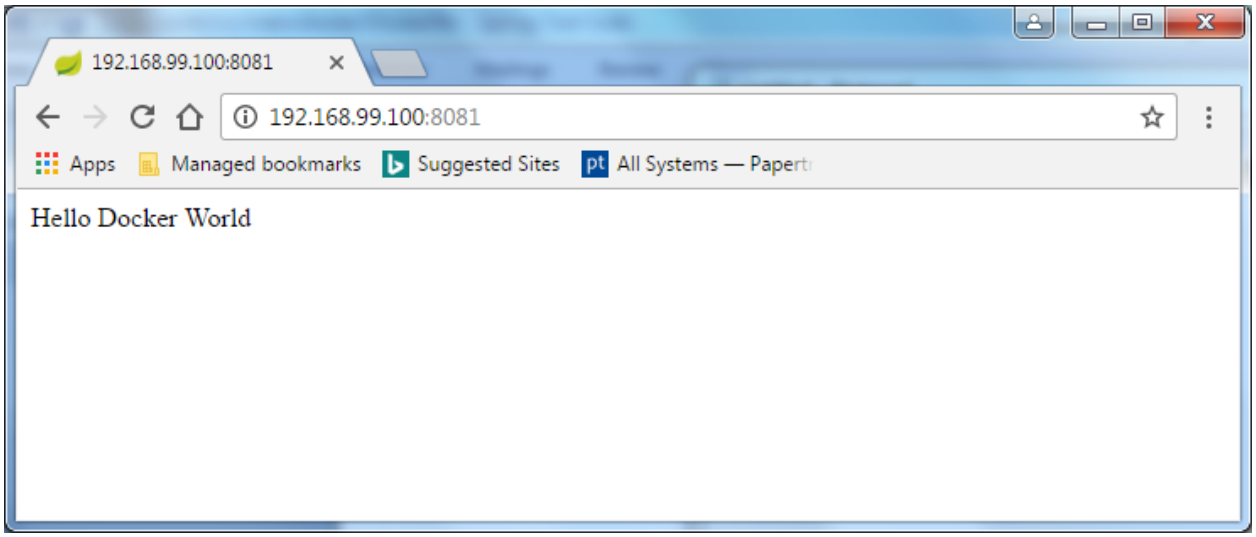
9. Now enter the images command
    a. **docker images**

```
MINGW64:/d/vidavid/Microservice/digital_bank/hello-world
vidavid@LIN73000499 MINGW64 /d/vidavid/Microservice/digital_bank/hello-world
$ docker images
REPOSITORY                  TAG                 IMAGE ID            CREATED
        SIZE
caprepo/hello-world         latest              a1bec60389d6        2 minutes a
go      195.3 MB
caprepo/duckairlines        latest              8ffed0f86cf6        2 days ago
        716.4 MB
<none>                      <none>              34de2747dec6        2 days ago
        716 MB
<none>                      <none>              dee213f35fb0        3 days ago
        716 MB
frolvlad/alpine-oraclejdk8  slim                00d8610f052e        6 days ago
        166.6 MB
maven                       latest              b5ab9b7ecf5a        8 days ago
        651.5 MB
mysql                       5.6                 673eeb95a1a4        10 days ago
        327.5 MB
mysql                       5.5                 0697aa95507c        10 days ago
        255 MB
mysql                       latest              22be5748ecbe        10 days ago
        405.6 MB
mysql/mysql-server          latest              65f6760a2eef        3 weeks ago
        214.7 MB
```

10. You can find **caprepo/hello-world** image in the list
11. Now enter the below command:
    a. **docker run -p 8081:8081 -t caprepo/hello-world**

```
MINGW64:/d/vidavid/Microservice/digital_bank/hello-world
HandlerMapping : Mapped "{[/error],produces=[text/html]}" onto public org.spring
framework.web.servlet.ModelAndView org.springframework.boot.autoconfigure.web.Ba
sicErrorController.errorHtml(javax.servlet.http.HttpServletRequest,javax.servlet
.http.HttpServletResponse)
2017-03-10 11:47:08.509  INFO 5 --- [           main] s.w.s.m.m.a.RequestMapping
HandlerMapping : Mapped "{[/error]}" onto public org.springframework.http.Respon
seEntity<java.util.Map<java.lang.String, java.lang.Object>> org.springframework.
boot.autoconfigure.web.BasicErrorController.error(javax.servlet.http.HttpServlet
Request)
2017-03-10 11:47:08.591  INFO 5 --- [           main] o.s.w.s.handler.SimpleUrlH
andlerMapping  : Mapped URL path [/webjars/**] onto handler of type [class org.s
pringframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-03-10 11:47:08.598  INFO 5 --- [           main] o.s.w.s.handler.SimpleUrlH
andlerMapping  : Mapped URL path [/**] onto handler of type [class org.springfra
mework.web.servlet.resource.ResourceHttpRequestHandler]
2017-03-10 11:47:08.713  INFO 5 --- [           main] o.s.w.s.handler.SimpleUrlH
andlerMapping  : Mapped URL path [/**/favicon.ico] onto handler of type [class o
rg.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2017-03-10 11:47:09.104  INFO 5 --- [           main] o.s.j.e.a.AnnotationMBeanE
xporter        : Registering beans for JMX exposure on startup
2017-03-10 11:47:09.285  INFO 5 --- [           main] s.b.c.e.t.TomcatEmbeddedSe
rvletContainer : Tomcat started on port(s): 8081 (http)
2017-03-10 11:47:09.305  INFO 5 --- [           main] com.capgemini.Application
               : Started Application in 8.851 seconds (JVM running for 10.929)
```

12. You will be getting the command that application started. Now open your application in the browser with the ip:



13. Now our docker image is up and running well.

Conclusion:

From the above example, we learnt how to create docker container with image for simple hello-world application.

Lab 2:

Push the hello-world docker image which has been created in the last lab assignment.

Steps:

1. List all images with "docker images" command. You can see caprepo/hello-world is one of the docker image.

```
vidavid@LIN73000499 MINGW64 /d/vidavid/Microservice/digital_bank/hello-world
$ docker images
REPOSITORY                  TAG                 IMAGE ID            CREATED
          SIZE
caprepo/hello-world         latest              c1ae54f88644        13 seconds
ago       195.3 MB
frolvlad/alpine-oraclejdk8  slim                00d8610f052e        11 days ago
          166.6 MB
```

2. Login to docker container with docker login
   a. Note: use docker hub login details to login.

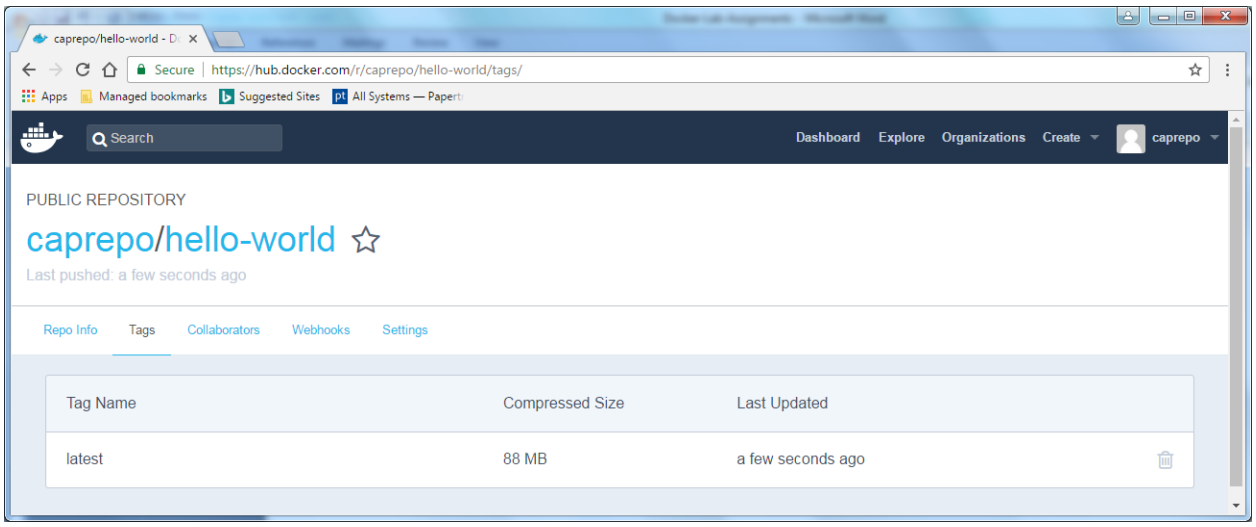
```
MINGW64:/d/vidavid/Microservice/digital_bank/hello-world

vidavid@LIN73000499 MINGW64 /d/vidavid/Microservice/digital_bank/hello-wor
$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you
have a Docker ID, head over to https://hub.docker.com to create one.
Username (caprepo): caprepo
Password:
Login Succeeded

vidavid@LIN73000499 MINGW64 /d/vidavid/Microservice/digital_bank/hello-wor
$
```

3. **$docker push caprepo/hello-world**
   a. This command will push the docker image into docker hub public repository.

```
vidavid@LIN73000499 MINGW64 /d/vidavid/Microservice/digital_bank/hello-world
$ docker push caprepo/hello-world
The push refers to a repository [docker.io/caprepo/hello-world]
754d33a71ee4: Pushed
05600308ad3b: Pushed
162f935b1198: Mounted from frolvlad/alpine-oraclejdk8
dcf909146faa: Mounted from frolvlad/alpine-oraclejdk8
23b9c7b43573: Mounted from frolvlad/alpine-oraclejdk8
latest: digest: sha256:81aa2fd1587bf33ca97a6c0bdcf38bcaf6011fac79f619db45ef7c1a9
9f47e52 size: 1375
```

Output:

Once it pushes the docker image you will be getting the image in your docker hub reposity as shown below:



4. To check whether docker image works properly , you use docker run command.
   a. **docker run -p 8081:8081 -t caprepo/hello-world**

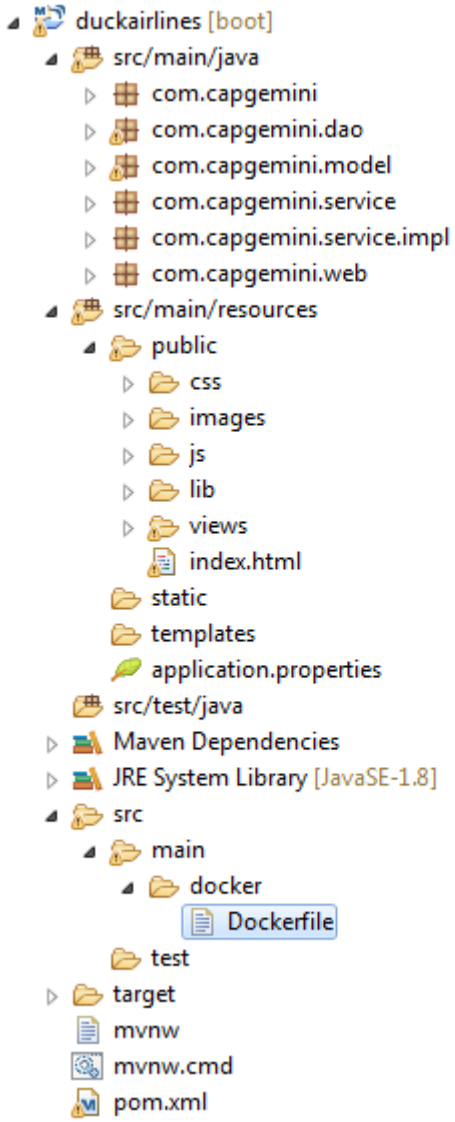
Conclusion:

We have successfully uploaded the docker image into public repository.

Lab 3:

Use your "duckairlines" project, just convert the app and database into docker image and execute the application using that database.

Steps:

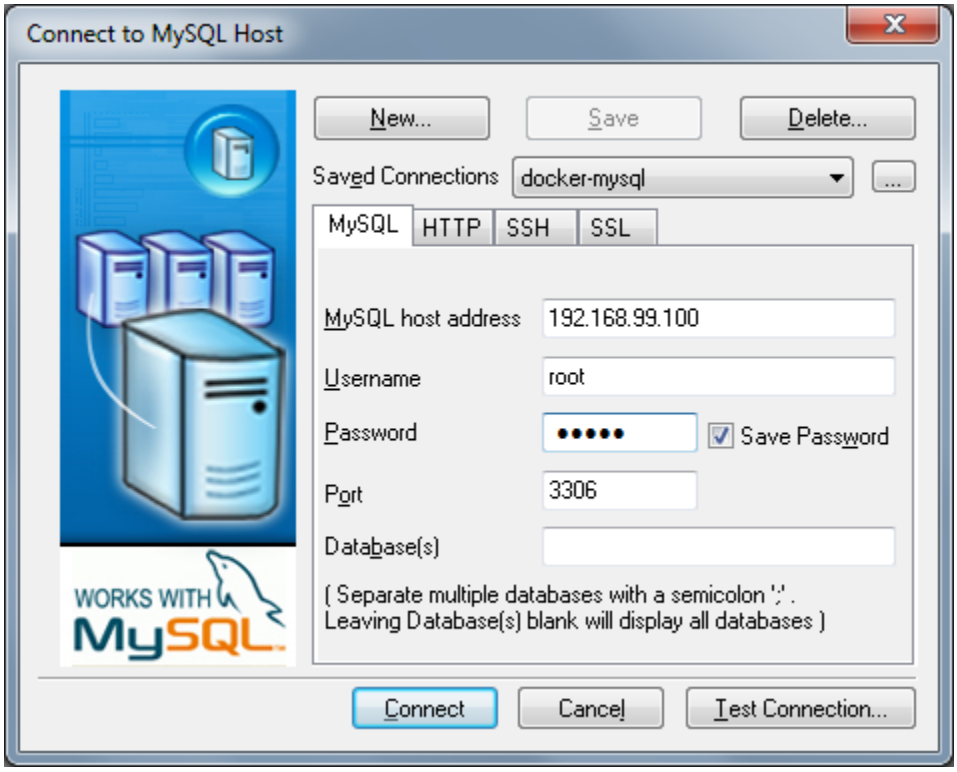


1. Your project structure should look like the above one, once you have downloaded the project into Spring Tool Suite.
2. Under src/main create new folder called docker and add Dockerfile which has the following script within it.

Dockerfile

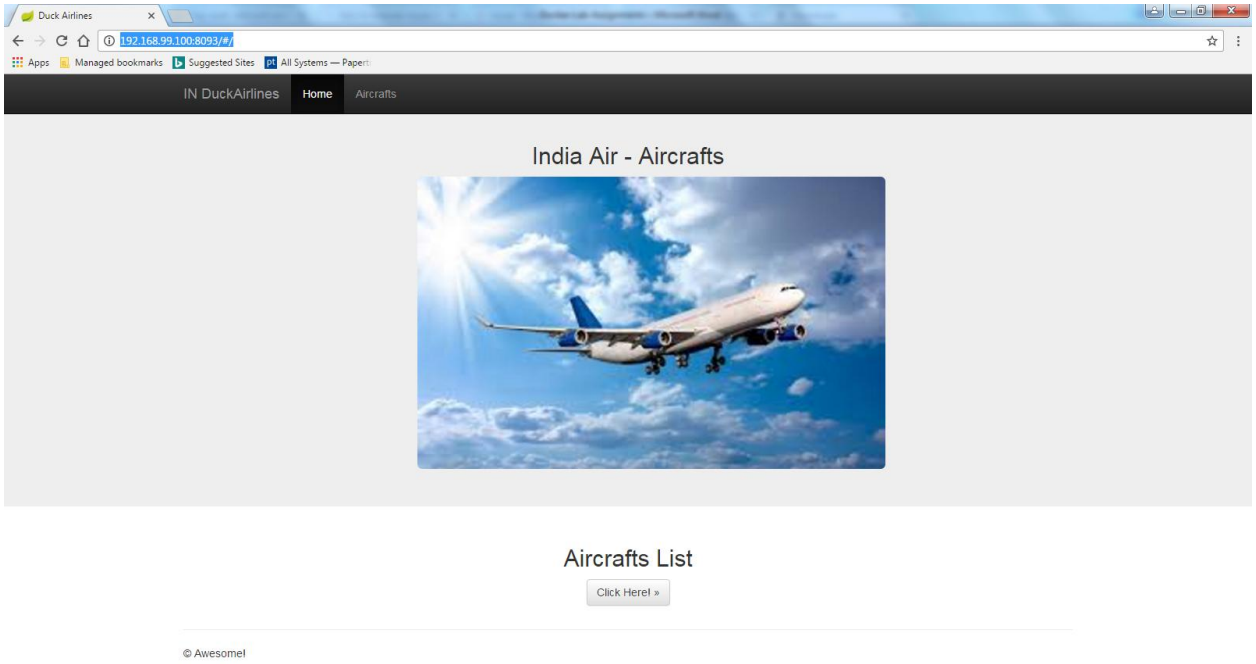
```
FROM maven

VOLUME /tmp
ADD duckairlines-0.1.0.jar app.jar
RUN sh -c 'touch /app.jar'
ENV JAVA_OPTS=""
ENTRYPOINT [ "sh", "-c", "java $JAVA_OPTS -
Djava.security.egd=file:/dev/./urandom -jar /app.jar" ]
```

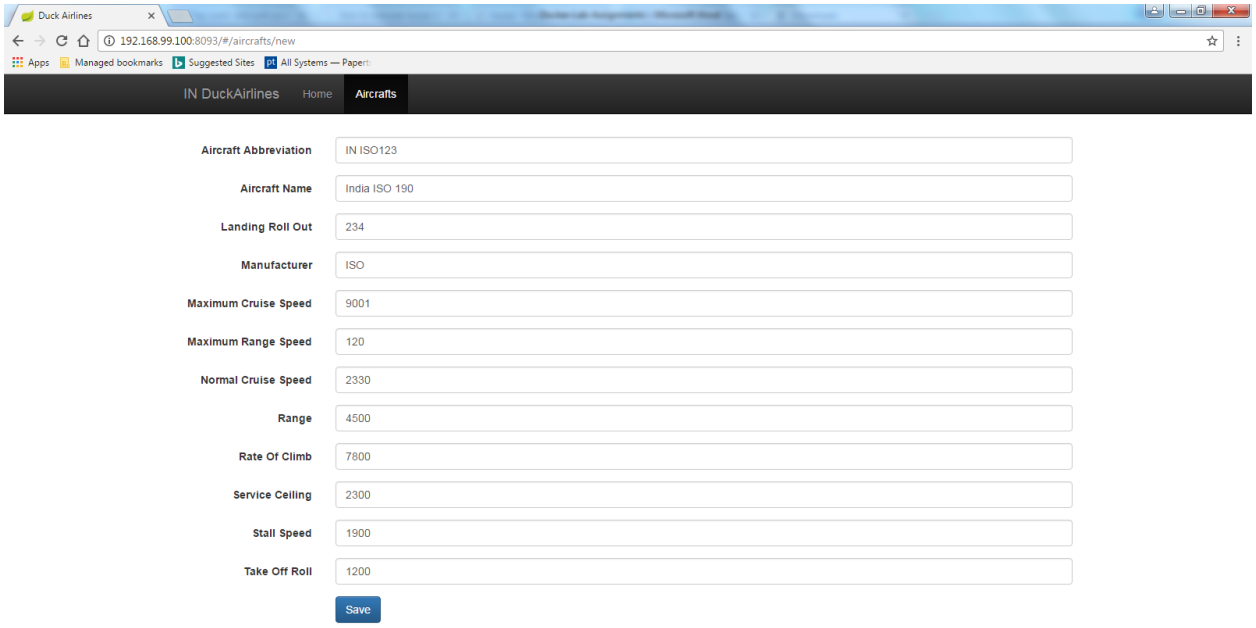
3. Open your Docker quick terminal window:
4. Navigate to the root directory of your project and enter the following command:
    a. **mvn package docker:build**
5. once the command executed successfully, you can see the docker image using
    a. **docker images**
6. Now we need mysql image, because our project uses mysql database as backend. To bring mysql in your Virtual machine use the following command.
    a. **docker run -e MYSQL_ROOT_PASSWORD=admin --name mysql -d -p=3306:3306 mysql**
    b. **$docker images**
7. You can see mysql image. To ensure that this mysql running in your machine use the below command:
    a. **$docker ps**
8. Open mySql yog window to check your connection with database show below:



9. Once its connected create your database which you want:
    a. Create database cap_duckairlines(in this project database name is duckairlines)
10. Run the docker image by using the following command:
    a. **$ docker run -p 8093:8093 -t caprepo/duckairlines**
11. once the image started you can access the duckairlines with the following link:
    a. http://192.168.99.100:8093/#/

Output:


Duck Airlines
192.168.99.100:8093/#/
Apps
Managed bookmarks
Suggested Sites
All Systems — Paper
IN DuckAirlines
Home
Aircrafts
India Air - Aircrafts
Aircrafts List
Click Here! »
© Awesome!


Duck Airlines
192.168.99.100:8093/#/aircrafts/new
Apps
Managed bookmarks
Suggested Sites
All Systems — Paper
IN DuckAirlines
Home
Aircrafts
Aircraft Abbreviation
IN ISO123
Aircraft Name
India ISO 190
Landing Roll Out
234
Manufacturer
ISO
Maximum Cruise Speed
9001
Maximum Range Speed
120
Normal Cruise Speed
2330
Range
4500
Rate Of Climb
7800
Service Ceiling
2300
Stall Speed
1900
Take Off Roll
1200
Save

Now check your database it will be filled with the details.

Conclusion:

From the above example we learnt how to create image for your application with database.