Docker Platform User Guide

Docker is the most popular software container platform that uses OS-level virtualization to deliver software in packages called containers.

Docker container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

Docker image is a lightweight, standalone, executable package of software that includes everything needed to run an application including code, runtime, system tools, system libraries and settings. Docker container images become containers at runtime when they run on Docker Engine

Docker Hub is a registry service on the cloud that allows you to download Docker images that are built by other communities

Installation of Docker on local window machine

Please follow step by step

1. Download the latest version of Docker Toolbox, go to https://github.com/docker/toolbox/releases and download the latest .exe file.
2. Install Docker Toolbox by double-clicking on the installer
3. Press Next and Next to accept all the defaults and then Install
4. When notified by Windows Security the installer will make changes, make sure you allow the installer to make the necessary changes.
5. Verify your installatio

Running spring boot application on docker machine

First step is to setup mysql environment on docker machine before deploying spring boot application container on docker engine

Please follow step by step

1. Pull mysql server image from docker hub. Following below command will pull latest mysql server from docker hub

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker pull mysql/mysql-server:latest
```

2. Verity docker image using command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mysql/mysql-server  latest              a7a39f15d42d        3 months ago        381MB
```

3. Runing docker image becomes container. Please run mysql server container using bellow command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker run --name=mysql-server -e MYSQL_ROOT_PASSWORD=password -e MYSQL_DATABASE=mydb -d mysql/mysql-server:latest
bf2b9fec90f115a8598e5947f9a74b2f6cec425f5ea178edcb2e74864246d49c
```

'--name', It gives a name of the docker container. User can choose any custom name

' -e', It specifies run time variables. User need to set the runtime variables for the mysql container. User can choose any name

MYSQL_ROOT_PASSWORD=root

MYSQL_DATABASE=mydb

4. Verify the running status of the container by issuing below command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker ps
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS                            PORTS                    NAMES
bf2b9fec90f1        mysql/mysql-server:latest      "/entrypoint.sh mysq…"   About a minute ago  Up About a minute (healthy)       3306/tcp, 33060/tcp      mysql-server
```

Now we should able to see that mysql server is running on port 3306

5. Find the IP of the container using following.

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker inspect mysql-server
    {
        "Id": "bf2b9fec90f115a8598e5947f9a74b2f6cec425f5ea178edcb2e74864246d49c",
        "IPAddress": "172.17.0.2",
```

Now we should be able to connect to mysql server using this ip address on port 3306.

6. Verifying mysql server using command line

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker exec -it mysql-server bash
bash-4.2# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 387
Server version: 8.0.19 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mydb               |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

7. We can check the logs of the running container use the following command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker logs mysql-server
[Entrypoint] MySQL Docker Image 8.0.19-1.1.15
[Entrypoint] Initializing database
```

8. We can also access mysql server using web interface (phpmyadmin or mysql workbench)

Accessing mysql database through web Interface (phpMyAdmin)

Please follow step by step

1. Pull phpmyadmin from docker hub using below command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker pull phpmyadmin/phpmyadmin
```

2. Verify phpmyadmin docker image using below command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker images
REPOSITORY                TAG                 IMAGE ID            CREATED             SIZE
springbootdeployondocker  latest              5fbd3e85c9ca        4 hours ago         692MB
phpmyadmin/phpmyadmin     latest              366fde4f732e        4 weeks ago         468MB
mysql/mysql-server        latest              a7a39f15d42d        3 months ago        381MB
```

3. Run phpmyadmin container using below command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker run --name myphpadmin -d --link mysql-server:db -p 8080:80 phpmyadmin/phpmyadmin
b7eb9ac1c2bee6f3c29de1c2b839748d8c658f731eee87fd22bca1ce66fe5f12
```

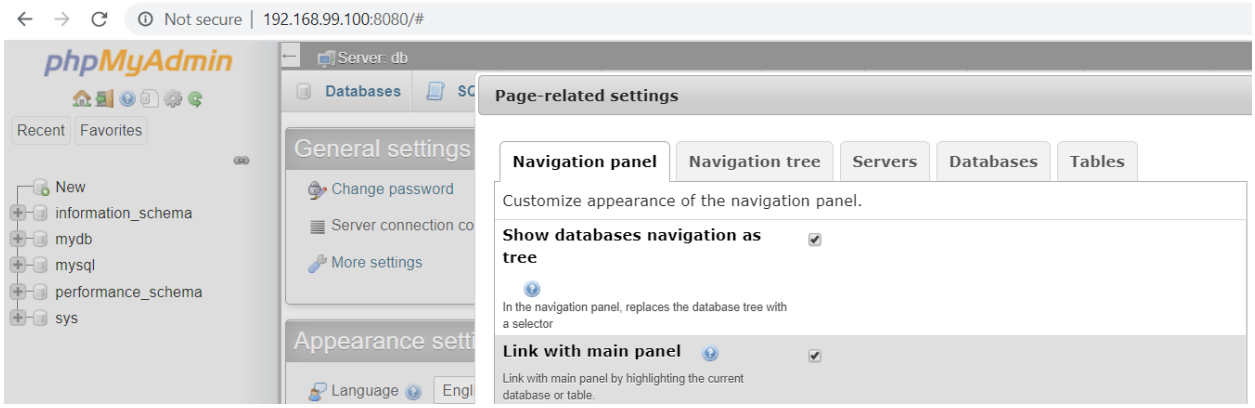
   First, this command will pull the phpmyadmin image and second will create the phpmyadmin container named 'myphpadmin' and link it to 'mysql-server' container which we created earlier

4. Checking default IP of docker machine using below command

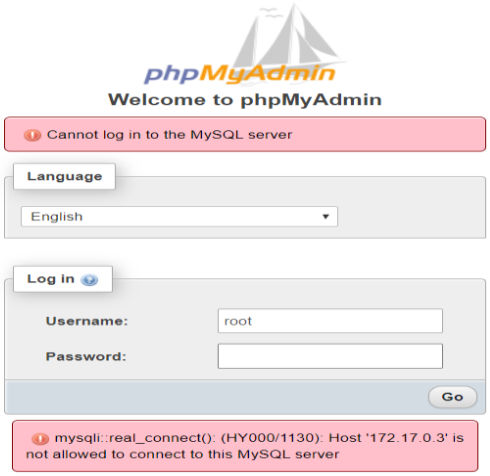
```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker-machine ip
192.168.99.100
```

5. Now you can connect to mysql server database from phpmyadmin by using url http://192.168.99.100:8080. Below page will show once we hit this url. If error please follow step 7

6. Now we can access the **mydb** database which we have created using mysql docker image



7. If phpmyadmin by using url http://192.168.99.100:8080 through error and solving this describes down



- Login mysql using command

```
khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker exec -it mysql-server bash
bash-4.2# mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 563
Server version: 8.0.19 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

- Execute these two command
  CREATE USER 'root'@'172.17.0.3' IDENTIFIED BY 'password';
  GRANT ALL PRIVILEGES ON *.* TO 'root'@'172.17.0.3' WITH GRANT OPTION;

```
mysql> CREATE USER 'root'@'172.17.0.3' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'172.17.0.3' WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT host, user FROM mysql.user;
+-----------+-----------------+
| host      | user            |
+-----------+-----------------+
| 172.17.0.3 | root            |
| localhost | healthchecker   |
| localhost | mysql.infoschema |
| localhost | mysql.session   |
| localhost | mysql.sys       |
| localhost | root            |
+-----------+-----------------+
6 rows in set (0.00 sec)
```

- Stop and start mysql container

```
khan_@LAPTOP-O2QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker ps -a
CONTAINER ID        IMAGE                      COMMAND                  CREATED             STATUS                    PORTS                    NAMES
b7eb9ac1c2be        phpmyadmin/phpmyadmin      "/docker-entrypoint.…"   About an hour ago   Up 6 minutes              0.0.0.0:8080->80/tcp     myphpadmin
bf2b9fec90f1        mysql/mysql-server:latest  "/entrypoint.sh mysq…"   About an hour ago   Up 6 minutes (healthy)    3306/tcp, 33060/tcp      mysql-server

khan_@LAPTOP-O2QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker stop bf2b9fec90f1
bf2b9fec90f1

khan_@LAPTOP-O2QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker start bf2b9fec90f1
bf2b9fec90f1
```

- Now we can used url http://192.168.99.100:8080. Below page will show once we hit this url.
  Username: root
  Password: password

Deploy springboot application with MySQL database on docker platform

Now we have container database server is up and running, we will create separate container for our spring boot web application. Please follow below steps –

1. Clone and copy springboot-deploy-on-docker application at local machine from github repository
2. Used below command to go inside project directory

```
D:\workspace> cd springboot-deploy-on-docker
```

3. Open project inside eclipse or used maven command line to clean and build project

   mvn clean install

4. Change database configuration inside application.properties

```
server.port = 8088

spring.datasource.url=jdbc:mysql://mysql-server:3306/mydb?createDatabaseIfNotExist=true&allowPublicKeyRetrieval=true&useSSL=false
spring.datasource.username=root
spring.datasource.password=password
spring.jpa.generate-ddl=true
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.format_sql=true
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
```

5. Modify server port as per your choice in application.properties and Dockerfile.yml file

```
FROM java:8
EXPOSE 8088
ADD /target/springbootdeployondocker.jar springbootdeployondocker.jar
ENTRYPOINT ["java", "-jar", "springbootdeployondocker.jar"]
```

6. Now we can create docker images using below command
   $ docker build –t springbootdeployondocker .

```
C:\workspace\springboot-deploy-on-docker> docker build -t springbootdeployondocker
Sending build context to Docker daemon  49.24MB
Step 1/4 : FROM java:8
8: Pulling from library/java
5040bd298390: Pull complete
fce5728aad85: Pull complete
76610ec20bf5: Pull complete
60170fec2151: Pull complete
e98f73de8f0d: Pull complete
11f7af24ed9c: Pull complete
49e2d6393f32: Pull complete
bb9cdec9c7f3: Pull complete
Digest: sha256:c1ff613e8ba25833d2e1940da0940c3824f03f802c449f3d1815a66b7f8c0e9d
Status: Downloaded newer image for java:8
 ---> d23bdf5b1b1b
Step 2/4 : EXPOSE 8088
 ---> Running in 749018cf6dda
Removing intermediate container 749018cf6dda
 ---> f8851ff59142
Step 3/4 : ADD /target/springbootdeployondocker.jar springbootdeployondocker.jar
 ---> 73e2f58069fd
Step 4/4 : ENTRYPOINT ["java", "-jar", "springbootdeployondocker.jar"]
 ---> Running in 28e093fbf6bd
Removing intermediate container 28e093fbf6bd
 ---> ec26ef48350d
Successfully built ec26ef48350d
Successfully tagged springbootdeployondocker:latest
```

7. Verify docker images

```
C:\workspace\springboot-deploy-on-docker> docker images
REPOSITORY                  TAG                 IMAGE ID            CREATED             SIZE
springbootdeployondocker    latest              ec26ef48350d        2 minutes ago       692MB
phpmyadmin/phpmyadmin       latest              366fde4f732e        4 weeks ago         468MB
mysql/mysql-server          latest              a7a39f15d42d        3 months ago        381MB
java                        8                   d23bdf5b1b1b        3 years ago         643MB
```

8. Now we can run springboot container

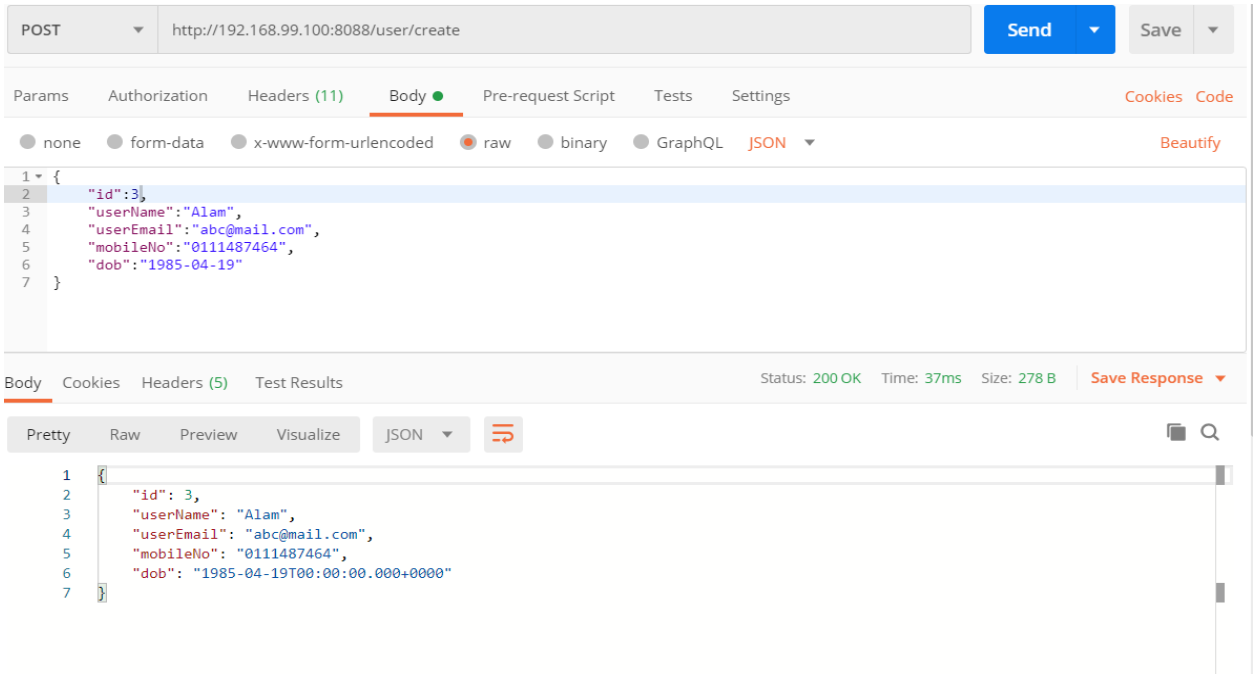
```
C:\workspace\springboot-deploy-on-docker>docker run --name springbootdeployondocker -d --link mysql-server:db -p 8088:8088 springbootdeployondocker
042581a9553a63e028f70de29befe6af0573a6046bcdd4dead8a13e3677c92d6
```

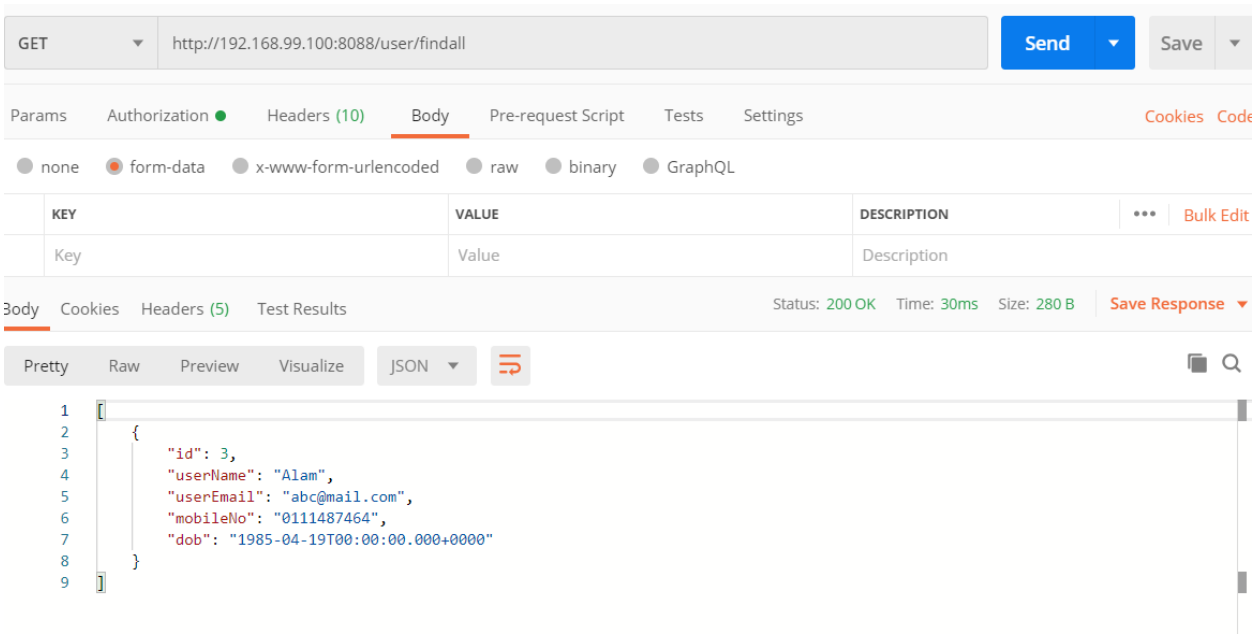
9. Checked log file for details

```
$ docker ps -a
CONTAINER ID        IMAGE                          COMMAND                  CREATED             STATUS                    PORTS                              NAMES
6947af3c6d81        springbootdeployondocker       "java -jar springboo…"   22 seconds ago      Up 21 seconds             0.0.0.0:8088->8088/tcp             springbootdeployondocker
b7eb9ac1c2be        phpmyadmin/phpmyadmin          "/docker-entrypoint.…"   23 hours ago        Up 47 minutes             0.0.0.0:8080->80/tcp               myphpadmin
bf2b9fec90f1        mysql/mysql-server:latest      "/entrypoint.sh mysq…"   24 hours ago        Up 21 minutes (healthy)   3306/tcp, 33060/tcp                mysql-server

khan_@LAPTOP-02QAHBFL MINGW64 /c/Program Files/Docker Toolbox
$ docker logs 6947af3c6d81

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.2.6.RELEASE)

2020-04-19 09:09:52.918  INFO 1 --- [           main] .d.d.SpringbootDeployOnDockerApplication : Starting SpringbootDeployOnDockerApplication v0.0.1-SNAPSHOT on 6947af3c6d8
1 with PID 1 (/springbootdeployondocker.jar started by root in /)
2020-04-19 09:09:52.929  INFO 1 --- [           main] .d.d.SpringbootDeployOnDockerApplication : No active profile set, falling back to default profiles: default
2020-04-19 09:09:55.347  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2020-04-19 09:09:55.475  INFO 1 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 114ms. Found 1 JPA repository i
nterfaces.
2020-04-19 09:09:57.930  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8088 (http)
2020-04-19 09:09:57.981  INFO 1 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2020-04-19 09:09:57.981  INFO 1 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.33]
2020-04-19 09:09:58.235  INFO 1 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2020-04-19 09:09:58.236  INFO 1 --- [           main] o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization completed in 5127 ms
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual
 loading of the driver class is generally unnecessary.
2020-04-19 09:09:59.560  INFO 1 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2020-04-19 09:09:59.769  INFO 1 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.4.12.Final
2020-04-19 09:10:00.101  INFO 1 --- [           main] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
2020-04-19 09:10:00.382  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2020-04-19 09:10:00.386  WARN 1 --- [           main] com.zaxxer.hikari.util.DriverDataSource  : Registered driver with driverClassName=com.mysql.jdbc.Driver was not found,
 trying direct instantiation.
2020-04-19 09:10:00.818  INFO 1 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2020-04-19 09:10:00.875  INFO 1 --- [           main] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.MySQL5Dialect2020-04-19 09:
10:03.310  INFO 1 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platfo
rm.internal.NoJtaPlatform]
2020-04-19 09:10:03.339  INFO 1 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2020-04-19 09:10:04.140  WARN 1 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries
may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2020-04-19 09:10:04.677  INFO 1 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2020-04-19 09:10:05.371  INFO 1 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8088 (http) with context path ''
2020-04-19 09:10:05.379  INFO 1 --- [           main] .d.d.SpringbootDeployOnDockerApplication : Started SpringbootDeployOnDockerApplication in 13.745 seconds (JVM running
for 15.384)
```

Now we can see spring boot application running on port 8088

10. Calling api using postman




GET
http://192.168.99.100:8088/user/findall
Send
Save
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
Code
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
DESCRIPTION
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 30ms
Size: 280 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 3,
"userName": "Alam",
"userEmail": "abc@mail.com",
"mobileNo": "0111487464",
"dob": "1985-04-19T00:00:00.000+0000"
}
]