CMPE 281 - LAB #4 - Docker Starbucks API

Introduction:

In this Lab, you will be working with a sample implementation of the Restbucks API implemented in Java with Restlet. You will test the API locally with Postman, Deploy and test using a Local Docker Host, and finally deploy to Docker on AWS.

Key Steps:

- Install Docker Desktop or Native Docker
- Setup Docker Cloud with your AWS Account
- Install Postman REST API Testing Tool
- Build the Restbucks API Project (will need Gradle)
- Test/Deploy Restbucks

Lab Files:

- https://github.com/paulnguyen/cmpe281/tree/master/labs/lab4

PART I - Setup

- **Install Docker Toolbox:**
  - https://www.docker.com/products/docker-toolbox
  - https://www.docker.com/docker-mac
  - https://www.docker.com/docker-windows
  - https://www.docker.com/docker-ubuntu
  - ***NOTE:*** *This lab works best on Mac or Linux. If you are using a Windows Machine, it is best to use the Docker Toolbox Option to run Docker in a Linux VM.*

- **Register for Docker Hub Account:**
  - https://hub.docker.com/
- **Use the Docker Hosts to your AWS Account Created in Lab #3**
  - https://github.com/paulnguyen/cmpe281/blob/master/aws/dockerhost/dockerhost.md
  - **Notes:** Micro Free-Tier Instance in any AZ
- **Download and Install Postman Desktop App:**
  - https://www.getpostman.com/

PART II - LOCAL HOST

SOURCE https://github.com/paulnguyen/cmpe281/blob/master/labs/lab4/starbucks.zip

Compile, Build JAR and Run Local Instance on port 9090 using Makefile Targets:

```
compile:
    gradle build

jar: compile
    gradle shadowJar

run:
    echo Starting Service at:  http://localhost:9090
    java -cp build/libs/starbucks-all.jar api.StarbucksServer
```

Using Postman's Starbucks Test Collection, run the "Post Starbucks API \| Localhost" Test with Postman Console View Enable.Expand the Request and Response Headers and Bodies, take a Screenshot.

PART III - LOCAL DOCKER

SOURCE https://github.com/paulnguyen/cmpe281/blob/master/labs/lab4/starbucks.zip

Compile, Build JAR and Run Local Docker Instance on port 90 using Makefile Targets:

```
 docker-build:
    docker build -t starbucks .
    docker images

docker-run-bridge:
    docker run --name starbucks -td -p 90:9090 starbucks
    docker ps
```

Using Postman's Starbucks Test Collection, run the "Post Starbucks API \| Docker Local" Test with Postman Console View Enable. Expand the Request and Response Headers and Bodies, take a Screenshot.

PART IV - DOCKER ON AWS

SOURCE https://github.com/paulnguyen/cmpe281/blob/master/labs/lab4/starbucks.zip

Using the **docker.sh** script, build and push a release to Docker Hub and then Deploy to Docker on AWS.

Note: When deploying to Docker, map the Container's Exported Port 9090 to Docker **Port 90.**

Using Postman's **Starbucks Test Collection**, run the "**Post Starbucks API | Docker Cloud (Host IP)**" Test with Postman Console View Enable. You will need to update the Docker Host IP to use your setup.

REF: https://github.com/paulnguyen/cmpe281/blob/master/labs/lab4/RESTLET.json

- Import **RESTLET.json** into Postman

Expand the Request and Response Headers and Bodies, take a Screenshot.

In your Docker Host Environment, use Docker Logs to capture the Output after running the above Postman Test.

REF: https://docs.docker.com/engine/reference/commandline/logs/