Operating Systems – Dockers and Cloud Exercise

- Amiram Abergel - 204147219

Segment 1 - Containers and Docker (70 points)

Part 0 – Install Docker on your Ubuntu machine (0 points):

** Note: when showing terminal commands, the character "\" means breaking a line into two lines. Feel free to press Enter after this character or just ignore it and write the whole command in one line. **

1. Open terminal.
2. Run the following commands:
   - Update the apt package index:
     sudo apt-get update
   - Install packages to allow apt to use a repository over HTTPS:
     *sudo apt-get install *
     *apt-transport-https *
     *ca-certificates *
     *curl *
     software-properties-common
   - Add Docker's official GPG key:
     curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

- Use the following command to set up the stable repository:

```
sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"
```

- Install Docker:

```
sudo apt-get install docker.io
```

Part 1 – Docker basic commands and "Hello World" (0 points):

1. Run *sudo docker info* to view details about your docker installation.
2. Run *sudo docker container ls* to see a list of your running containers. The output should be empty since you don't have any running containers yet.
3. Run *sudo docker image ls* to see a list of your images.
   The output should be empty since you don't have any images yet.
4. Run *sudo docker pull hello-world* to pull the "Hello World" image from the Docker Hub.
5. Run again *sudo docker image ls*. Now the list is not empty and hello-world image is there.
6. Run *sudo docker run hello-world* to start a container running the hello-world image. The hello-world container displays a message and then exits.
7. Run again *sudo docker container ls --all*. Now the list is not empty and hello-world container is there.
   Note: If the hello-world container was still running you would not need the --all option.

Part 2 – Creating your first Docker image (40 points):

In Part 1 you have used an existing image named *hello-world*. In this part you are required to create an image by yourself and run it.
The image will run a Java application which creates a simple web server that prints your full name as a response message.

1. Editing the application source code:
   a. Create a new empty directory.
   b. Change directory into the new directory.
   c. Edit the file named **WebServer.java** (supplied with this pdf).
      Change the variable 'name' to hold your full name.
2. Compiling and running the java application:

First, let's make sure the app works.

a. Compile the java application (use *javac* command line, or your IDE).
b. Run the java application (use *java* command line or your IDE).
c. Open a web browser and browse to http://localhost:8000

   As a result, your full name should be printed on the browser page.

   If it does – everything works as expected. Let's use Docker.

3. First, you need to get back to the starting state. Therefore, delete the file **WebServer.class** that was created after the compilation. **The directory should include only WebServer.java file**.
4. Create a new file named **Dockerfile** (yes, with no extension), and write its content.
   a. When writing the script, think about the flow of things that should happen when the image will be built:
      i. The Docker should import an image from the remote Docker library. This image should be a JVM image.
      ii. Then, you need to copy WebServer.java to the machine, compile it, open port 8000.
   b. When the container runs, it should execute the compiled file.
   c. Use the commands shown in the recitation: RUN, WORKDIR, COPY, FROM, EXPOSE, CMD…
5. Now that the Dockerfile is ready:
   a. Build a new image which includes the java application source code. In the folder where the Dockerfile and WebServer.java are located run the following command: *sudo docker build -t javawebserver .* (mind the space and the dot after "javawebserver").
   b. Run *sudo docker image ls* to view current images. Note that your new image *javawebserver* is included.
   c. Run the image using the command line: *sudo docker run -p 4000:8000 javawebserver*
   d. Open a web browser and browse to http://localhost:4000

      As a result, your full name should be printed in the browser.
      Take a screenshot. Make sure the IP (in the address line) and your name are visible.

Part 3 (30 points, 6 points each):

For each of the statements below, state whether **it is true or false and explain concisely**.

1. Each container requires its own underlying OS, and the hardware is virtualized.

שקר,
container משתמש ב Kernal של ה Host OS ויש לו תלות במערכת ההפעלה, לכן גם container אינו צריך מערכת הפעלה משלו וכן גם חומרה.

2. Containers can be used only on Linux environment (use Google, if needed)

שקר,
"Docker runs natively on both operating systems" windows and linux.

3. The Dockerfile file contains instruction commands to build and run the image.

אמת,
DOCKER יכול לבנות images אוטומטית ע"י קריאת הפקודות הנמצאות ב Dockerfile ואז גם להריץ את ה image.

4. Docker can pull images only from Docker Hub registry.

שקר,
ישנה אפשרות נוספת לבצע pull ידנית ע"י איזכור ה path של ה registry .

5. If your highest priority is maximizing the number of applications running on a minimal number of servers, you will prefer using containers and not VMs.

אמת, משום ש containers דורשים מעט מאוד גודל זיכרון לעומת vm , ובנוסף זולים לעומת ה vm.

From the Docker segment you will need to submit:

- The Dockerfile (from Part 2).
- The answers to Part 3 as a PDF.
- The screenshot you took, as an Image file.

Segment 2 – Cloud (30 points)

In this segment we'll learn about Cloud services.
Cloud services play an important role in every engineer's life, whether they work in a big corporate or doing their first steps in their new venture.
We'll do a Microsoft Azure tutorial, which will list the different types of services and ways they can be utilized for our benefit. Note that this tutorial is general and relevant to the other cloud providers (AWS, Google…).
Following, we'll create our own VM on Azure and play with it for a bit.
Enjoy…

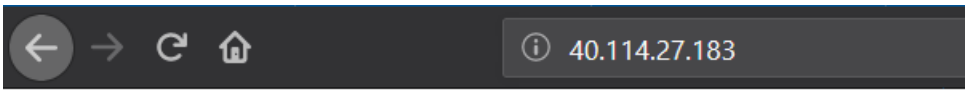
Go to https://docs.microsoft.com/en-us/learn/paths/azure-fundamentals/

Part 1 – Cloud Concepts (0 points)

- Go over the first module: "Cloud Concepts - Principles of cloud computing".
- That's it.

Part 2 - Core Cloud Services - Introduction to Azure (30 points)

- This module is located right below the first one.
- In it, you will play around with Azure, create a VM and a very simple web server.
    - **IMPORTANT:** When creating the VM – do not name it myVM (which is the default name), but rather by your first and last name. For example, our good friend and colleague Tony will name his VM: "TonySoprano".
- When asked to navigate to your VM's IP, take a screenshot of the tab. **Make sure the IP (in the address line) and your name are visible.** For example, my man Tony will get this message, once he's done with his career change:



Welcome to Azure! My name is tonySoprano.

From this segment you will submit the screenshot, as an Image file.

Submit, as ONE ZIP FILE:

- One PDF file for the Docker questions.
- One Dockerfile file.
- One screenshot from the Docker part.
- One screenshot from the Cloud part.
- **ALSO: on Moodle, unit 12, there's a very short survey that will be used to evaluate this exercise. Please complete it until submission date (MANDATORY)**

  Submissions that will be missing files won't be graded.

Enjoy!