# Solution M1: DevOps and Containerization

This is one possible solution for the tasks included in the homework

All steps that follow assume that we decided to base our solution on **Debian 12.x**

## Virtualization

### Creating a Debian box

Define a simple VM in **VirtualBox** with the following parameters

* Name set to **debian**
* 1 CPU
* 2 GB RAM
* 32 GB HDD (Dynamic)
* No Audio
* 1 NIC in NAT mode (with a port forwarding rule **SSH**/**TCP**/**2222**/**22**)

Attach the installation media and boot the VM

Follow the installation wizard steps and

* Set the **root** user password to **vagrant**
* Create user **vagrant** with password **vagrant**
* Do not forget to mark the installation of the **OpenSSH** server

Once the installation is finished, open an SSH session to the VM with the user you created during the installation

* Switch to the **root** user

**su - root**

* Upgrade all installed packages (you may use either **apt** or **apt-get**)

**apt-get update**

**apt-get upgrade**

* Add new packages if you see fit. For example, install at least these (they are needed for building the **VirtualBox** modules) plus the **sudo** one:

**apt-get install gcc make perl sudo linux-headers-$(uname -r)**

* Insert the **VirtualBox Add-ons** media
* Mount the **VirtualBox Add-ons** media and install them

**mount /dev/sr0 /mnt**

**/mnt/VBoxLinuxAdditions.run**

* Reboot the system (even if not required)

**reboot**

* Establish an SSH session with the **vagrant** user and switch to the **root** user
* Eject the **VirtualBox Add-ons** media
* Add the **vagrant** user to the **vboxsf** group

**usermod -aG vboxsf vagrant**

* Add the **vagrant** user to the **sudoers** list and allow it to **sudo** without entering password

**echo "vagrant ALL=(ALL) NOPASSWD:ALL" | tee /etc/sudoers.d/vagrant**

* Clear the history and exit the root’s session

**history -c -w && cat /dev/null > .bash\_history && exit**

* Create a folder to store the insecure key

**mkdir -m 0700 -p /home/vagrant/.ssh**

* Install the **vagrant** insecure key

**wget --no-check-certificate \**

**https://raw.github.com/mitchellh/vagrant/master/keys/vagrant.pub \**

**-O /home/vagrant/.ssh/authorized\_keys**

* Change the permissions of the key

**chmod 0600 /home/vagrant/.ssh/authorized\_keys**

* Stop the auto update processes

**sudo systemctl disable --now apt-daily.timer**

**sudo systemctl disable --now apt-daily-upgrade.timer**

* Clean up the package cache

**sudo apt-get clean**

* Make sure that the hard disk is aligned

**sudo dd if=/dev/zero of=/EMPTY bs=1M status=progress**

**sudo rm -f /EMPTY**

* Clean the history and reboot the VM

**history -c -w && cat /dev/null > .bash\_history && sudo reboot**

Now, on the host

* Create a folder on the host to store the box, for example **debian**
* In a terminal session, navigate to the folder created earlier
* While the VM is still running, build the box by executing

**vagrant package --base debian**

We can publish our box or use it locally

For the sake of simplicity, let’s go for the second option – use it locally

We must register the box with

**vagrant box add debian-local package.box**

### Using the Debian Box

Use the practice files from the lecture

* Create folder **M1HW**
* Copy there the contents of folder **m1\2-2** from the practice files
* Open the **Vagrantfile** and change both **web.vm.box** and **db.vm.box** to the box created earlier. For example, if not published, but used locally, it may be set to **debian-local** (or the name used for box registration)
* Modify the **db.sh** file to match this

#!/bin/bash

echo "\* Add hosts ..."

echo "192.168.89.100 web.do1.lab web" >> /etc/hosts

echo "192.168.89.101 db.do1.lab db" >> /etc/hosts

echo "\* Install Software ..."

apt-get update -y && apt-get upgrade -y

apt-get install -y mariadb-server

echo "\* Adjust MariaDB connectivity ..."

sudo sed -i.bak s/127.0.0.1/0.0.0.0/g /etc/mysql/mariadb.conf.d/50-server.cnf

systemctl restart mariadb

echo "\* Create and load the database ..."

mysql -u root < /vagrant/db\_setup.sql

* Modify the **web.sh** file to match this

#!/bin/bash

echo "\* Add hosts ..."

echo "192.168.89.100 web.do1.lab web" >> /etc/hosts

echo "192.168.89.101 db.do1.lab db" >> /etc/hosts

echo "\* Install Software ..."

apt-get update -y && apt-get upgrade -y

apt-get install -y apache2 php php-mysqlnd

echo "\* Remove the default index.html file ..."

rm /var/www/html/index.html

echo "\* Copy web site files to /var/www/html/ ..."

cp /vagrant/\* /var/www/html

* Deploy the environment with

**vagrant up**

* Open browser on the host and navigate to [**http://192.168.89.100**](http://192.168.89.100)
* There should be a working web application
* Destroy the machines

**vagrant destroy --force**

## Containerization

We will assume that we have access to a Docker instance (in a VM or installed directly on the host)

* Open a terminal session
* Start the container

**docker container run --name homework -it shekeriev/animal-stories**

* Search for the requested file

**find / -type f -name animal-stories.txt**

* Either copy the path *(it will be different each time)* to reuse it for the following commands, or use command substitution
* Let’s go with the second option – command substitution
* Check the contents of the file *(they will vary each time)*

**cat $(find / -type f -name animal-stories.txt)**

* Filter all the rows about **tigers** *(the result will differ)*

**cat $(find / -type f -name animal-stories.txt) | grep tigers**

* Finally, let’s prepare a list of all the **unique colors** sorted in **reverse** (descending) **order**

**cut -d ' ' -f 1 $(find / -type f -name animal-stories.txt) | sort -u -r**

*Note that the separator (space in the above command) will differ*

* Exit the container

**exit**

* And remove it

**docker container rm homework**