Appendix C: Setup guide

INTRODUCTION

This guide is designed to encourage future collaborators to prepare the environment in which the solution of this project has been implemented and to build a very similar environment thus being able to contribute innovative functionalities to the system or enhance the implementation that has been proposed.

The commands are based on Centos 7, so in case the server where this solution is implemented is under another Linux distribution, you should look for the synonymous commands for Advanced Packaging Tool (APT).

The structure of the chapters of which this guide is composed is as follows:

- Master node
- Worker nodes
- Authenticate worker nodes with public key
- Master node: Web Server

MASTER NODE

First we proceed to update the system packets, change the server hostname and add local DNS for the nodes.

```
sudo yum update -y && sudo yum upgrade -y
sudo hostnamectl set-hostname master
sudo cat << EOF >> /etc/hosts
# Final Project hosts
172.26.91.18      master      www.thesisraul.mche.edu.pl
172.26.91.20      node01
172.26.91.21      node02
EOF
```

Then, install all the necessary dependencies to be able to continue.

```
sudo yum install -y epel-release yum-utils git curl php73
mysql-server python3 perl yaml php73-php-pecl-yaml.x86_64 php-yaml
php-mysqli pip3 pyyaml mod_ssl
```

Then ports 80 (HTTP) and 443 (HTTPS) are opened in the system to be able to locate the web server on these.

```
sudo firewall-cmd --list-all
sudo firewall-cmd --permanent --add-service=httpd
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-port=80/tcp
sudo firewall-cmd --permanent --add-port=443/tcp
sudo firewall-cmd --reload
```

Then, we generate a pair of asymmetric keys to be able to authenticate ourselves further on the node servers without entering the password manually. We also enter the permissions in the directory where they are stored octally to only allow access to these keys to the owner of them.

```
ssh-keygen -t rsa -b 4096
sudo chmod -R 700 .ssh/
sudo chmod -R 600 ~/.ssh/id_*
sudo chmod -R 600 ~/.ssh/authorized_keys
```

Using PIP3 install the YAML extension for Python that will be required to execute the code.

```
pip3 install --user pyyaml
```

Now we set the two global variables to .bash_profile that will be used to authenticate the Gmail SMTP server.

```
sudo cat << EOF >> ~/.bash_profile
export MAIL_USERNAME=admin@mail.com
export MAIL_TOKEN=xxxxxxxxxxxxxxxx
EOF
source ~/.bash_profile
```

Next, we add the current user to the apache group and start the httpd and mysqld services.

```
sudo usermod -aG apache $(whoami)
sudo systemctl enable httpd
sudo systemctl start httpd
sudo systemctl enable mysqld
sudo systemctl start mysqld
```

WORKER NODES

In each of the worker servers, you must first verify that virtualization is enabled on all CPUs, this is a requirement because the services are mounted on top of virtualization platforms.

```
lscpu | grep Virtualization
- Virtualization: VT-x
```

Now update the system, change the hostname and put the local DNS for the master node and the current node.

```
sudo yum update -y && sudo yum upgrade -y
sudo hostnamectl set-hostname node01
sudo cat << EOF >> /etc/hosts
# Final Project hosts
172.26.91.18      master      www.thesisraul.mche.edu.pl
172.26.91.20      node01 # This is the same worker host where we are
configuring
EOF
```

Next, all the packages that are required to continue must be installed.

```
sudo yum install -y epel-release yum-utils git curl python3 gcc dkms
make qt libgomp patch kernel-devel-$(uname -r) kernel-headers
kernel-headers-$(uname -r kernel-devel binutils glibc-headers
glibc-devel font-forge dkms qemu-kvm libvirt libvirt-python
libguestfs-tools virt-install
```

This step must be performed if Virtualbox is used because it requires having some GUI to be able to execute the virtual machines.

```
sudo yum groupinstall "Cinnamon Desktop"
systemctl set-default graphical
```

```
reboot
```

Then install Docker, which is necessary to start the service The Littlest Jupytherhub.

```
sudo yum install -y yum-utils
sudo yum-config-manager --add-repo
https://download.docker.com/linux/centos/docker-ce.repo
Centos 7
    sudo yum install docker-ce docker-ce-cli containerd.io
Centos 8
    sudo dnf install docker-ce-3:18.09.1-3.el7
sudo systemctl start docker
sudo systemctl enable docker
sudo usermod -aG docker $(whoami)
```

On the other hand, Vagrant is installed, which is used with the Antidote Selfmedicate service.

```
sudo wget
https://releases.hashicorp.com/vagrant/2.2.2/vagrant_2.2.2_x86_64.rp
m
sudo rpm -i vagrant_2.2.2_x86_64.rpm
```

And together, Virtualbox is installed.

```
sudo wget
https://download.virtualbox.org/virtualbox/rpm/el/virtualbox.repo -P
/etc/yum.repos.d
sudo yum install VirtualBox-6.1
```

Then the current user is added to the group docker and vboxusers to have absolute control over these platforms. Also create the ssh directory, if it does not exist, and the permissions are changed to only have access to the same user and no one else.

```
sudo usermod -aG docker $(whoami)
sudo usermod -aG vboxusers $(whoami)
mkdir -p .ssh/
sudo chmod -R 700 .ssh/
sudo chmod -R 600 ~/.ssh/authorized_keys
```

AUTHENTICATE WORKERS WITH PUBLIC KEY

After finishing the individual configuration, it logs back into the master server and the following command is executed for each node that is available. This command is in charge of copying the public key of the master node to the authorized_keys files which will allow accessing the node via SSH without putting a password.

```
master:~ $ ssh-copy-id -i ~/.ssh/id_rsa.pub $(whoami)@nodeXX
```

MASTER NODE - WEB SERVER

The idea of this web server is to serve as a basis for teachers to request the service that best suits their needs through a simple guided web form. The web application is built on LAMP (Linux, Apache, MySQL, PHP).

MySQL

Proceed with a guided installation to guarantee the minimum security of MySQL.

```
mysql_secure_installation
   -  Change the root password? [Y/n] n
   -  Remove anonymous users? [Y/n] y
   -  Disallow root login remotely? [Y/n] y
   -  Remove test database and access to it? [Y/n] y
   -  Reload privilege tables now? [Y/n] y
```

Create database

```
mysql -u root -p
create database cloud;
use cloud;
```

Create table TMrequests and TMspecs

```
create table tmrequests(id varchar(32), tstart varchar(32), uuid
char(32), fname varchar(32), sname varchar(32), email varchar(64),
homeinst varchar(32), homedept varchar(32), trole varchar(10),
opmode char(6), cmode varchar(10), startd varchar(32), span int,
PRIMARY KEY(uuid));
```

```
create table tmspecs(id char(32), msize varchar(6), ltype
varchar(32), ltopic varchar(32), homedir varchar(32), nusers int,
coopmode varchar(10), ngroups int, addsrv varchar(10), whatpi
varchar(20), PRIMARY KEY (id), FOREIGN KEY (id) REFERENCES
tmrequests(uuid) ON UPDATE RESTRICT ON DELETE CASCADE);
```

Exit the database and we restart the MySQL service

```
exit
sudo systemctl restart mysqld
```

# INFORMATION - Commands to check the records of the tables.

```
mysql -u root -p
use cloud;
select * from tmrequests;
select * from tmspecs;
```

APACHE - CONFIGURATION

Assuming that the project source code is downloaded, we copy the Apache directory, change the permissions and the owner and restart the service.

```
sudo cp -r ~/cloud-orchestration/TM/thesisraul/ /var/www/
sudo chown -R apache:apache /var/www/
sudo chmod -R 755 /var/www/
sudo systemctl restart httpd
```

Next, we generate a self signed certificate for the implementation of the project. In a real environment this must be verified by a CA. Next we move the generated keys and change the permissions together with the owner.

```
openssl req -new -newkey rsa:4096 -days 3650 -nodes -x509 -subj
"/C=PL/ST=Malopolska/L=Krakow/O=MCHE/CN=www.thesisraul.mche.edu.pl"
-keyout thesisraul.key -out thesisraul.crt
sudo mkdir -p /etc/ssl/crt
sudo mv thesisraul.* /etc/ssl/crt/
sudo chown -R apache:apache /etc/ssl/crt
sudo chmod -R 755 /etc/ssl/crt
```

Then, configure httpd to link the web site to the virtual hosts.

```
sudo mv /etc/httpd/conf/httpd.conf /etc/httpd/conf/httpd.conf.bak
sudo cp ~/cloud-orchestration/TM/httpd/conf/httpd.conf
/etc/httpd/conf/httpd.conf
sudo mkdir -p /etc/httpd/sites-available
sudo mkdir -p /etc/httpd/sites-available
sudo cp ~/cloud-orchestration/TM/httpd/sites-available/*
/etc/httpd/sites-available
sudo ln -s ../sites-available/thesisraul.conf thesisraul.conf
sudo ln -s ../sites-available/thesisraul-ssl.conf
thesisraul-ssl.conf
sudo systemctl restart httpd
```

COMMON ISSUE

Then we modify the apache user so that he has access without a password, this is the solution to an error that occurred when calling a bash file from PHP.

```
sudo visudo
  apache  ALL=(ALL) NOPASSWD:
/var/www/thesisraul/public_html/bash_call.sh
```

PHP - Configuration

Just keep in mind that the PECL version installed must be greater than 0.5.0

```
PECL yaml >= 0.5.0
```

Then disable SElinux so that there are no problems with the process in general, since for this implementation it is not required to have its operation.

```
sudo vi etc/selinux/config
# This file controls the state of SELinux on the system.
SELINUX=disabled
```

Finally we change the context in the bash_call.sh file which will call the main codiog of the program as user the one specified. This is necessary since by default the apache user is used because the call is made from it.

```
sudo vi /var/www/thesisraul/public_html/bash_call.sh
```

```
USR_CONTEXT='raulloga' # Must be changed to the user used in the implementation
```

SOURCE CODE

Regarding the source code, you must first clone in the $HOME directory and edit the configuration variables as appropriate.

```
cd ~
git clone https://github.com/llopisga/cloud-orchestration.git
```

```
vi cloud-orchestration/code/conf/config.yaml
PROJECT_PATH: "/home/USERNAME/cloud-orchestration"
TM_PATH: "/var/www/thesisraul/public_html/requests"
SMTP_SERVER: "smtp.gmail.com"
SMTP_PORT: 587
NOTIFY_ADMIN: "raullg8@gmail.com"
```