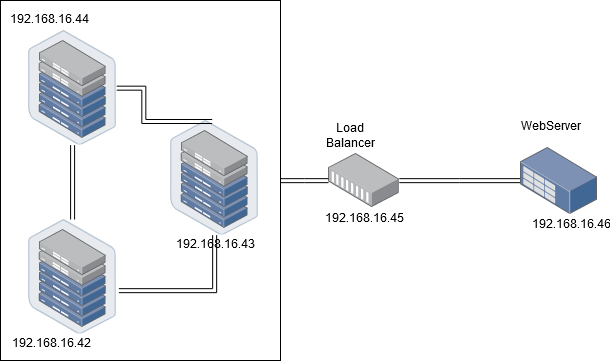
Infrastruktur Basis Data Terdistribusi



Spesifikasi server

* Server Database - 3 unit
* (192.168.16.42, 192.168.16.43, 192.168.16.44)
* OS: Ubuntu 16.04
* RAM 512 MB
* Database: MySQL
* Load-balancer - 1 unit
  + 192.168.16.45
  + OS: Ubuntu 16.04
  + RAM 512 MB
  + Database: MySQL with ProxySQL
* Webserver - 1 unit
  + 192.168.16.46
  + OS: Ubuntu 16.04
  + RAM 512 MB
  + Webserver: NGINX

**Proses Konfigurasi Server beserta MySQL**

* + 1. Konfigurasi Vagrantfile untuk membuat server dan juga proxy server.  
       Setting IP dimulai dari 192.168.16.42 - 192.168.16.45, dimana 45 merupakan proxy server dan lainnya merupakan tempat basis data terkait.

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure

# configures the configuration version (we support older styles for

# backwards compatibility). Please don't change it unless you know what

# you're doing.

Vagrant.configure("2") do |config|

# MySQL Cluster dengan 3 node

(1..3).each do |i|

config.vm.define "db#{i}" do |node|

node.vm.hostname = "db#{i}"

node.vm.box = "ubuntu/xenial64"

node.vm.network "private\_network", ip: "192.168.16.#{i+41}"

node.vm.boot\_timeout = 1200

# Opsional. Edit sesuai dengan nama network adapter di komputer

#node.vm.network "public\_network", bridge: "Qualcomm Atheros QCA9377 Wireless Network Adapter"

node.vm.provider "virtualbox" do |vb|

vb.name = "db#{i}"

vb.gui = false

vb.memory = "512"

end

node.vm.provision "shell", path: "bash/deployMySQL1#{i}.sh", privileged: false

end

end

config.vm.define "proxy" do |proxy|

proxy.vm.hostname = "proxy"

proxy.vm.box = "ubuntu/xenial64"

proxy.vm.network "private\_network", ip: "192.168.16.45"

#proxy.vm.network "public\_network", bridge: "Qualcomm Atheros QCA9377 Wireless Network Adapter"

proxy.vm.provider "virtualbox" do |vb|

vb.name = "proxy"

vb.gui = false

vb.memory = "512"

end

proxy.vm.provision "shell", path: "bash/deployProxySQL.sh", privileged: false

end

end

* + 1. Membuat skrip Provition untuk masing – masing server.

DeployMySQL 11 :

# Changing the APT sources.list to kambing.ui.ac.id

sudo cp '/vagrant/config/sources.list' '/etc/apt/sources.list'

# Updating the repo with the new sources

sudo apt-get update -y

# Install required library

sudo apt-get install libaio1

sudo apt-get install libmecab2

# Get MySQL binaries

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-server\_5.7.23-1ubuntu16.04\_amd64.deb

# Setting input for installation

sudo debconf-set-selections <<< 'mysql-community-server mysql-community-server/root-pass password admin'

sudo debconf-set-selections <<< 'mysql-community-server mysql-community-server/re-root-pass password admin'

# Install MySQL Community Server

sudo dpkg -i mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-server\_5.7.23-1ubuntu16.04\_amd64.deb

# Allow port on firewall

sudo ufw allow 33061

sudo ufw allow 3306

# Copy MySQL configurations

sudo cp /vagrant/config/my11.cnf /etc/mysql/my.cnf

# Restart MySQL services

sudo service mysql restart

# Cluster bootstrapping

sudo mysql -u root -padmin < /vagrant/sql/cluster\_bootstrap.sql

sudo mysql -u root -padmin < /vagrant/sql/addition\_to\_sys.sql

sudo mysql -u root -padmin < /vagrant/sql/create\_proxysql\_user.sql

DeployMySQL 12 :

# Changing the APT sources.list to kambing.ui.ac.id

sudo cp '/vagrant/config/sources.list' '/etc/apt/sources.list'

# Updating the repo with the new sources

sudo apt-get update -y

# Install required library

sudo apt-get install libaio1

sudo apt-get install libmecab2

# Get MySQL binaries

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-server\_5.7.23-1ubuntu16.04\_amd64.deb

# Setting input for installation

sudo debconf-set-selections <<< 'mysql-community-server mysql-community-server/root-pass password admin'

sudo debconf-set-selections <<< 'mysql-community-server mysql-community-server/re-root-pass password admin'

# Install MySQL Community Server

sudo dpkg -i mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-server\_5.7.23-1ubuntu16.04\_amd64.deb

# Allow port on firewall

sudo ufw allow 33061

sudo ufw allow 3306

# Copy MySQL configurations

sudo cp /vagrant/config/my12.cnf /etc/mysql/my.cnf

# Restart MySQL services

sudo service mysql restart

# Cluster bootstrapping

sudo mysql -u root -padmin < /vagrant/sql/cluster\_member.sql

DeployMySQL 13 :

# Changing the APT sources.list to kambing.ui.ac.id

sudo cp '/vagrant/config/sources.list' '/etc/apt/sources.list'

# Updating the repo with the new sources

sudo apt-get update -y

# Install required library

sudo apt-get install libaio1

sudo apt-get install libmecab2

# Get MySQL binaries

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-server\_5.7.23-1ubuntu16.04\_amd64.deb

# Setting input for installation

sudo debconf-set-selections <<< 'mysql-community-server mysql-community-server/root-pass password admin'

sudo debconf-set-selections <<< 'mysql-community-server mysql-community-server/re-root-pass password admin'

# Install MySQL Community Server

sudo dpkg -i mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-server\_5.7.23-1ubuntu16.04\_amd64.deb

# Allow port on firewall

sudo ufw allow 33061

sudo ufw allow 3306

# Copy MySQL configurations

sudo cp /vagrant/config/my13.cnf /etc/mysql/my.cnf

# Restart MySQL services

sudo service mysql restart

# Cluster bootstrapping

sudo mysql -u root -padmin < /vagrant/sql/cluster\_member.sql

Deploy ProxySQL:

# Changing the APT sources.list to kambing.ui.ac.id

sudo cp '/vagrant/config/sources.list' '/etc/apt/sources.list'

# Updating the repo with the new sources

sudo apt-get update -y

cd /tmp

curl -OL https://github.com/sysown/proxysql/releases/download/v1.4.4/proxysql\_1.4.4-ubuntu16\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

curl -OL https://dev.mysql.com/get/Downloads/MySQL-5.7/mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo apt-get install libaio1

sudo apt-get install libmecab2

sudo dpkg -i proxysql\_1.4.4-ubuntu16\_amd64.deb

sudo dpkg -i mysql-common\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-community-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo dpkg -i mysql-client\_5.7.23-1ubuntu16.04\_amd64.deb

sudo ufw allow 33061

sudo ufw allow 3306

sudo systemctl start proxysql

#mysql -u admin -padmin -h 127.0.0.1 -P 6032 < /vagrant/proxysql.sql

* + 1. Melakukan Konfigurasi SQL untuk masing – masing Server

SQL untuk Server 1 :

# The MySQL database server configuration file.

!includedir /etc/mysql/conf.d/

!includedir /etc/mysql/mysql.conf.d/

[mysqld]

# General replication settings

gtid\_mode = ON

enforce\_gtid\_consistency = ON

master\_info\_repository = TABLE

relay\_log\_info\_repository = TABLE

binlog\_checksum = NONE

log\_slave\_updates = ON

log\_bin = binlog

binlog\_format = ROW

transaction\_write\_set\_extraction = XXHASH64

loose-group\_replication\_bootstrap\_group = OFF

loose-group\_replication\_start\_on\_boot = ON

loose-group\_replication\_ssl\_mode = REQUIRED

loose-group\_replication\_recovery\_use\_ssl = 1

# Shared replication group configuration

loose-group\_replication\_group\_name = "8f22f846-9922-4139-b2b7-097d185a93cb"

loose-group\_replication\_ip\_whitelist = "192.168.16.42, 192.168.16.43, 192.168.16.44"

loose-group\_replication\_group\_seeds = "192.168.16.42:33061, 192.168.16.43:33061, 192.168.16.44:33061"

# Single or Multi-primary mode? Uncomment these two lines

# for multi-primary mode, where any host can accept writes

loose-group\_replication\_single\_primary\_mode = OFF

loose-group\_replication\_enforce\_update\_everywhere\_checks = ON

# Host specific replication configuration

server\_id = 1

bind-address = "192.168.16.42"

report\_host = "192.168.16.42"

loose-group\_replication\_local\_address = "192.168.16.42:33061"

SQL untuk Server 2 :

….

# Host specific replication configuration

server\_id = 2

bind-address = "192.168.16.43"

report\_host = "192.168.16.43"

loose-group\_replication\_local\_address = "192.168.16.43:33061"

SQL untuk Server 3 :

….

# Host specific replication configuration

server\_id = 3

bind-address = "192.168.16.44"

report\_host = "192.168.16.44"

loose-group\_replication\_local\_address = "192.168.16.44:33061"

* + 1. Membuat File SQL Tambahan.

Addition\_to\_sys.sql :

USE sys;

DELIMITER $$

CREATE FUNCTION IFZERO(a INT, b INT)

RETURNS INT

DETERMINISTIC

RETURN IF(a = 0, b, a)$$

CREATE FUNCTION LOCATE2(needle TEXT(10000), haystack TEXT(10000), offset INT)

RETURNS INT

DETERMINISTIC

RETURN IFZERO(LOCATE(needle, haystack, offset), LENGTH(haystack) + 1)$$

CREATE FUNCTION GTID\_NORMALIZE(g TEXT(10000))

RETURNS TEXT(10000)

DETERMINISTIC

RETURN GTID\_SUBTRACT(g, '')$$

CREATE FUNCTION GTID\_COUNT(gtid\_set TEXT(10000))

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE result BIGINT DEFAULT 0;

DECLARE colon\_pos INT;

DECLARE next\_dash\_pos INT;

DECLARE next\_colon\_pos INT;

DECLARE next\_comma\_pos INT;

SET gtid\_set = GTID\_NORMALIZE(gtid\_set);

SET colon\_pos = LOCATE2(':', gtid\_set, 1);

WHILE colon\_pos != LENGTH(gtid\_set) + 1 DO

SET next\_dash\_pos = LOCATE2('-', gtid\_set, colon\_pos + 1);

SET next\_colon\_pos = LOCATE2(':', gtid\_set, colon\_pos + 1);

SET next\_comma\_pos = LOCATE2(',', gtid\_set, colon\_pos + 1);

IF next\_dash\_pos < next\_colon\_pos AND next\_dash\_pos < next\_comma\_pos THEN

SET result = result +

SUBSTR(gtid\_set, next\_dash\_pos + 1,

LEAST(next\_colon\_pos, next\_comma\_pos) - (next\_dash\_pos + 1)) -

SUBSTR(gtid\_set, colon\_pos + 1, next\_dash\_pos - (colon\_pos + 1)) + 1;

ELSE

SET result = result + 1;

END IF;

SET colon\_pos = next\_colon\_pos;

END WHILE;

RETURN result;

END$$

CREATE FUNCTION gr\_applier\_queue\_length()

RETURNS INT

DETERMINISTIC

BEGIN

RETURN (SELECT sys.gtid\_count( GTID\_SUBTRACT( (SELECT

Received\_transaction\_set FROM performance\_schema.replication\_connection\_status

WHERE Channel\_name = 'group\_replication\_applier' ), (SELECT

@@global.GTID\_EXECUTED) )));

END$$

CREATE FUNCTION gr\_member\_in\_primary\_partition()

RETURNS VARCHAR(3)

DETERMINISTIC

BEGIN

RETURN (SELECT IF( MEMBER\_STATE='ONLINE' AND ((SELECT COUNT(\*) FROM

performance\_schema.replication\_group\_members WHERE MEMBER\_STATE != 'ONLINE') >=

((SELECT COUNT(\*) FROM performance\_schema.replication\_group\_members)/2) = 0),

'YES', 'NO' ) FROM performance\_schema.replication\_group\_members JOIN

performance\_schema.replication\_group\_member\_stats USING(member\_id));

END$$

CREATE VIEW gr\_member\_routing\_candidate\_status AS SELECT

sys.gr\_member\_in\_primary\_partition() as viable\_candidate,

IF( (SELECT (SELECT GROUP\_CONCAT(variable\_value) FROM

performance\_schema.global\_variables WHERE variable\_name IN ('read\_only',

'super\_read\_only')) != 'OFF,OFF'), 'YES', 'NO') as read\_only,

sys.gr\_applier\_queue\_length() as transactions\_behind, Count\_Transactions\_in\_queue as 'transactions\_to\_cert' from performance\_schema.replication\_group\_member\_stats;$$

DELIMITER ;

Cluster\_bootstrap.sql :

SET SQL\_LOG\_BIN=0;

CREATE USER 'repl'@'%' IDENTIFIED BY 'password' REQUIRE SSL;

GRANT REPLICATION SLAVE ON \*.\* TO 'repl'@'%';

FLUSH PRIVILEGES;

SET SQL\_LOG\_BIN=1;

CHANGE MASTER TO MASTER\_USER='repl', MASTER\_PASSWORD='password' FOR CHANNEL 'group\_replication\_recovery';

INSTALL PLUGIN group\_replication SONAME 'group\_replication.so';

SET GLOBAL group\_replication\_bootstrap\_group=ON;

START GROUP\_REPLICATION;

SET GLOBAL group\_replication\_bootstrap\_group=OFF;

CREATE DATABASE camppkmbk;

Cluster\_member.sql :

SET SQL\_LOG\_BIN=0;

CREATE USER 'repl'@'%' IDENTIFIED BY 'password' REQUIRE SSL;

GRANT REPLICATION SLAVE ON \*.\* TO 'repl'@'%';

FLUSH PRIVILEGES;

SET SQL\_LOG\_BIN=1;

CHANGE MASTER TO MASTER\_USER='repl', MASTER\_PASSWORD='password' FOR CHANNEL 'group\_replication\_recovery';

INSTALL PLUGIN group\_replication SONAME 'group\_replication.so';

Create\_proxy\_user.sql :

CREATE USER 'monitor'@'%' IDENTIFIED BY 'monitorpassword';

GRANT SELECT on sys.\* to 'monitor'@'%';

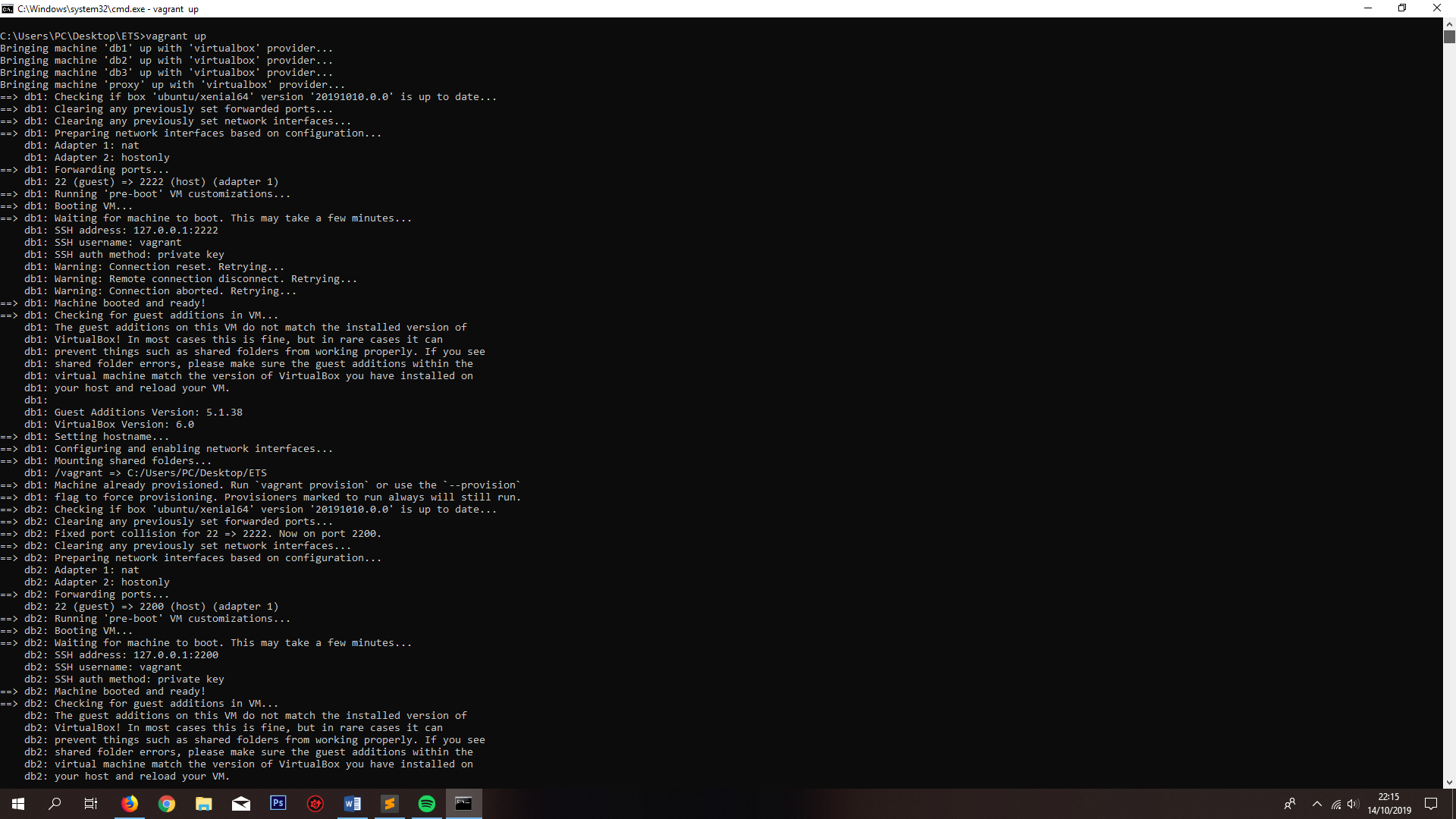
FLUSH PRIVILEGES;

CREATE USER 'camppkmbkuser'@'%' IDENTIFIED BY 'camppkmbkpassword';

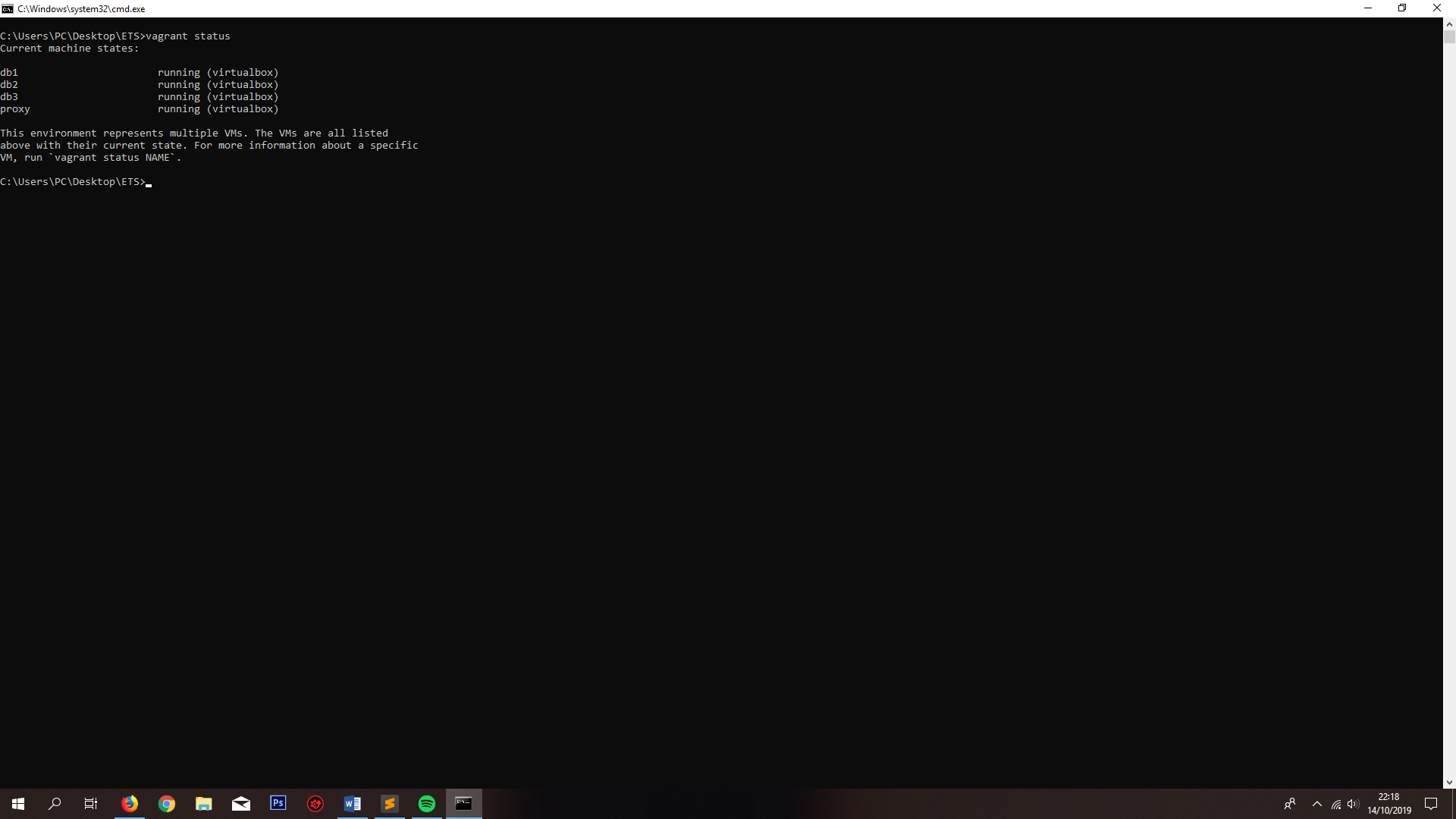
GRANT ALL PRIVILEGES on camppkmbk.\* to 'camppkmbkuser'@'%';

FLUSH PRIVILEGES;

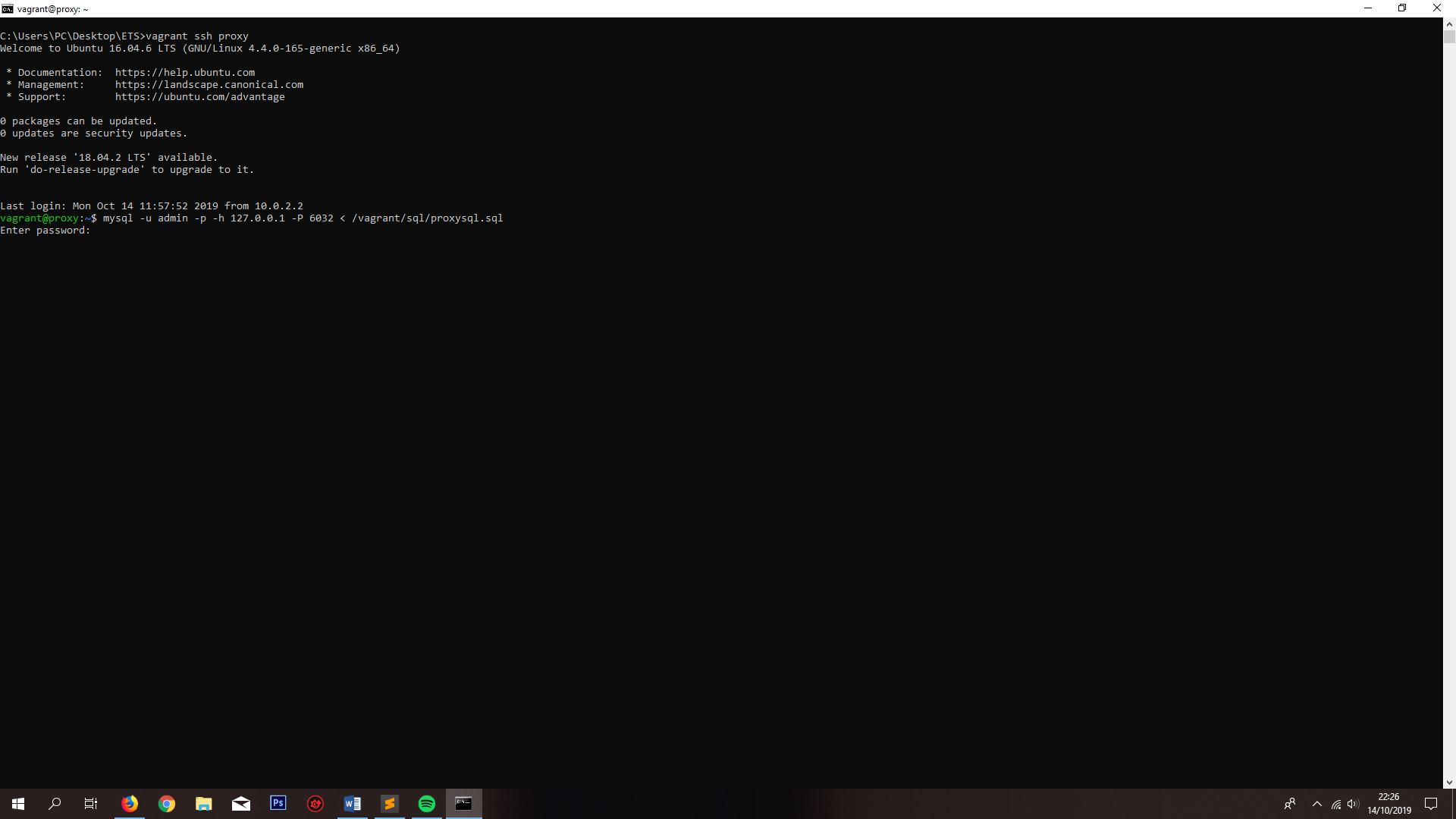
* + 1. Menjalankan Vagrant : vagrant up



* + 1. Melakukan cek terhadap status tiap server dengan perintah : vagrant status



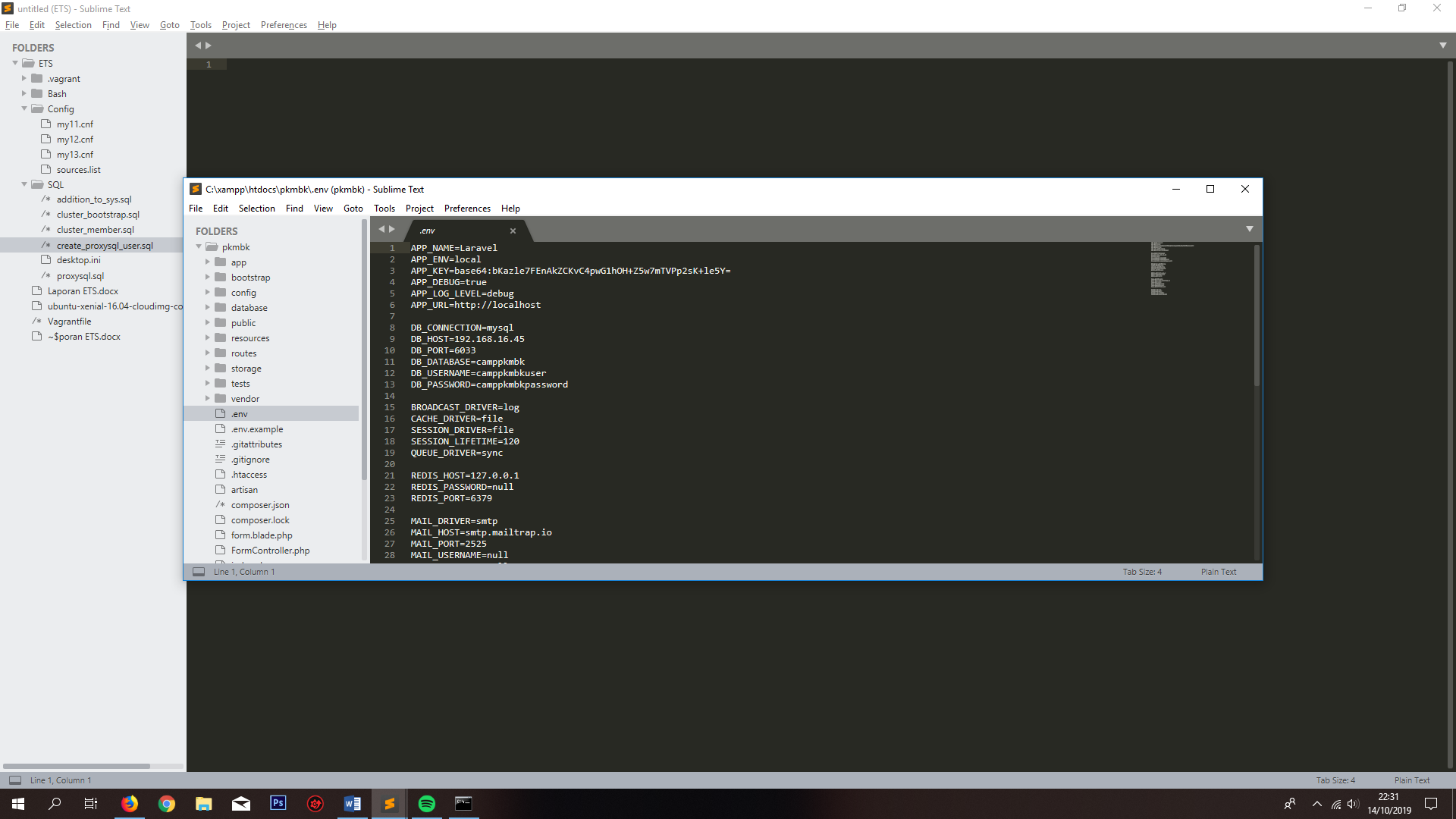
* + 1. Masuk ke ssh server Proxy dengan perintah : vagrant ssh proxy
    2. Masukkan proxysql.sql sebagai provision pada proxy :  
       mysql -u admin -p -h 127.0.0.1 -P 6032 < /vagrant/sql/proxysql.sql



**Implementasi BDT pada Web**

Penggunaan basis data terdistribusi kali ini menggunakan web yang sudah ada di komputer server. Web menggunakan Framework dan konfigurasi Laravel. Maka berikut langkah – langkah yang diperlukan untuk menaplikasikan BDT pada Web tersebut.

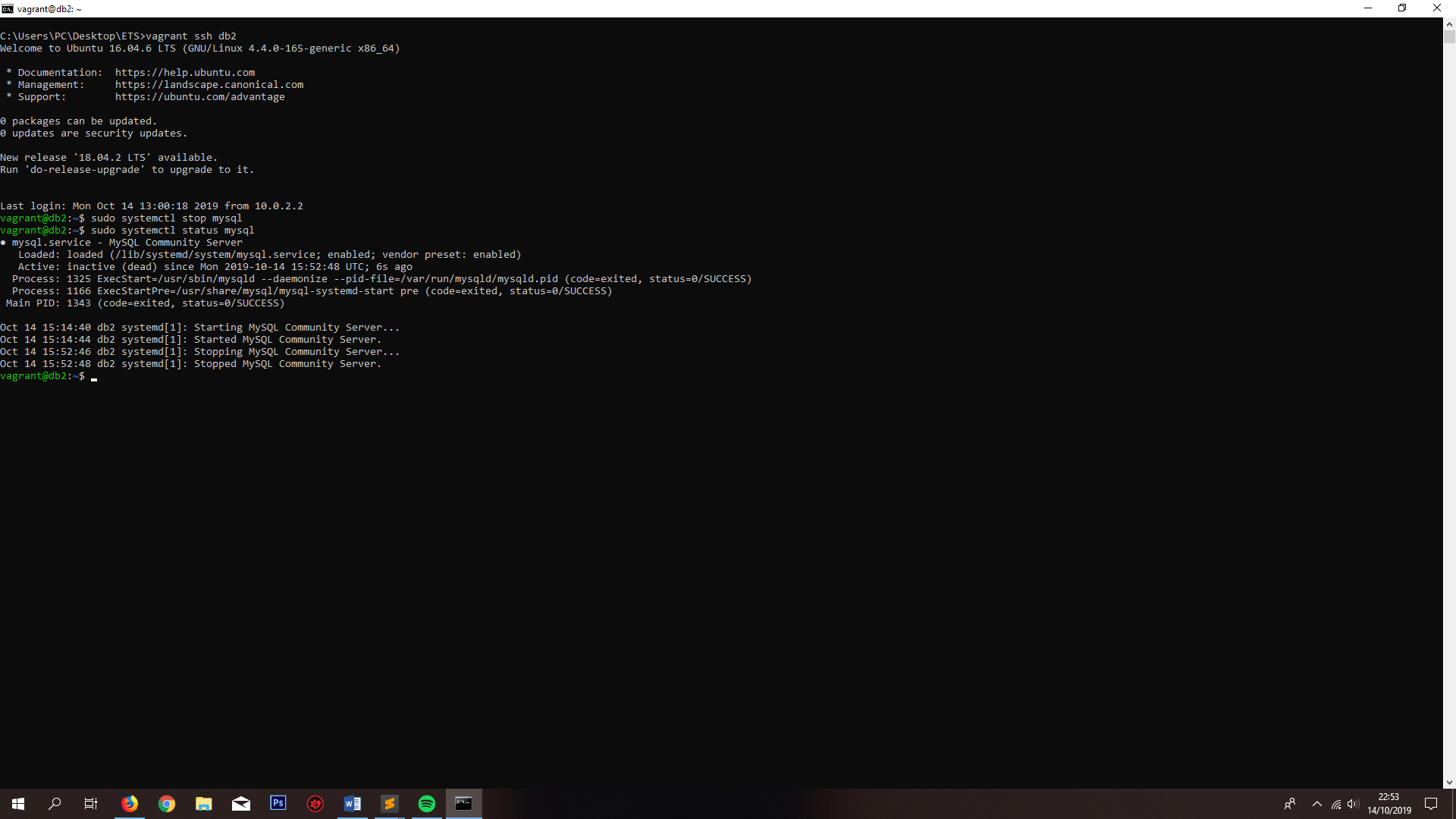
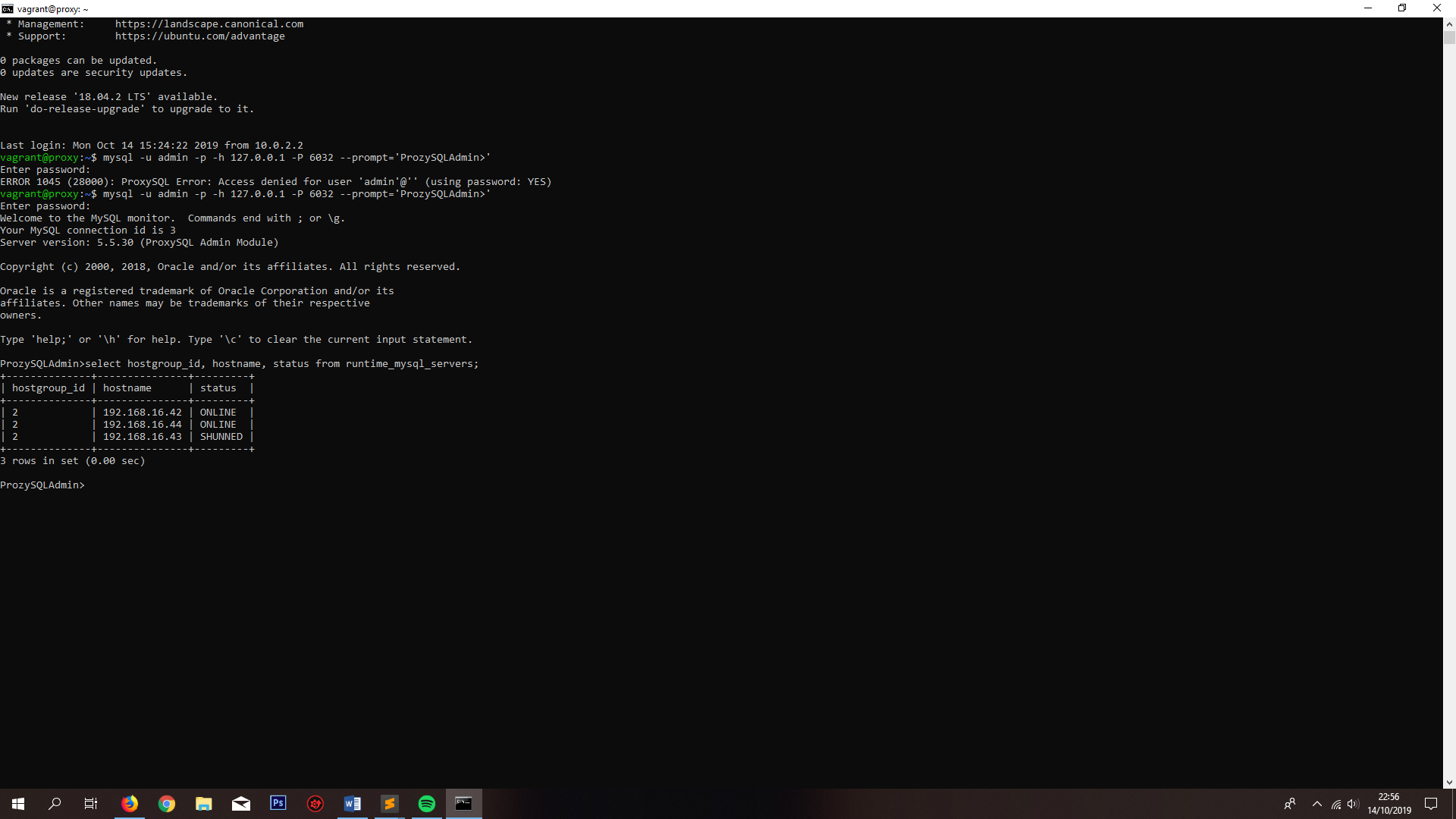
* + 1. Mengubah file .env pada laravel untuk menggunakan basis data pada server yang telah dikonfigurasi.

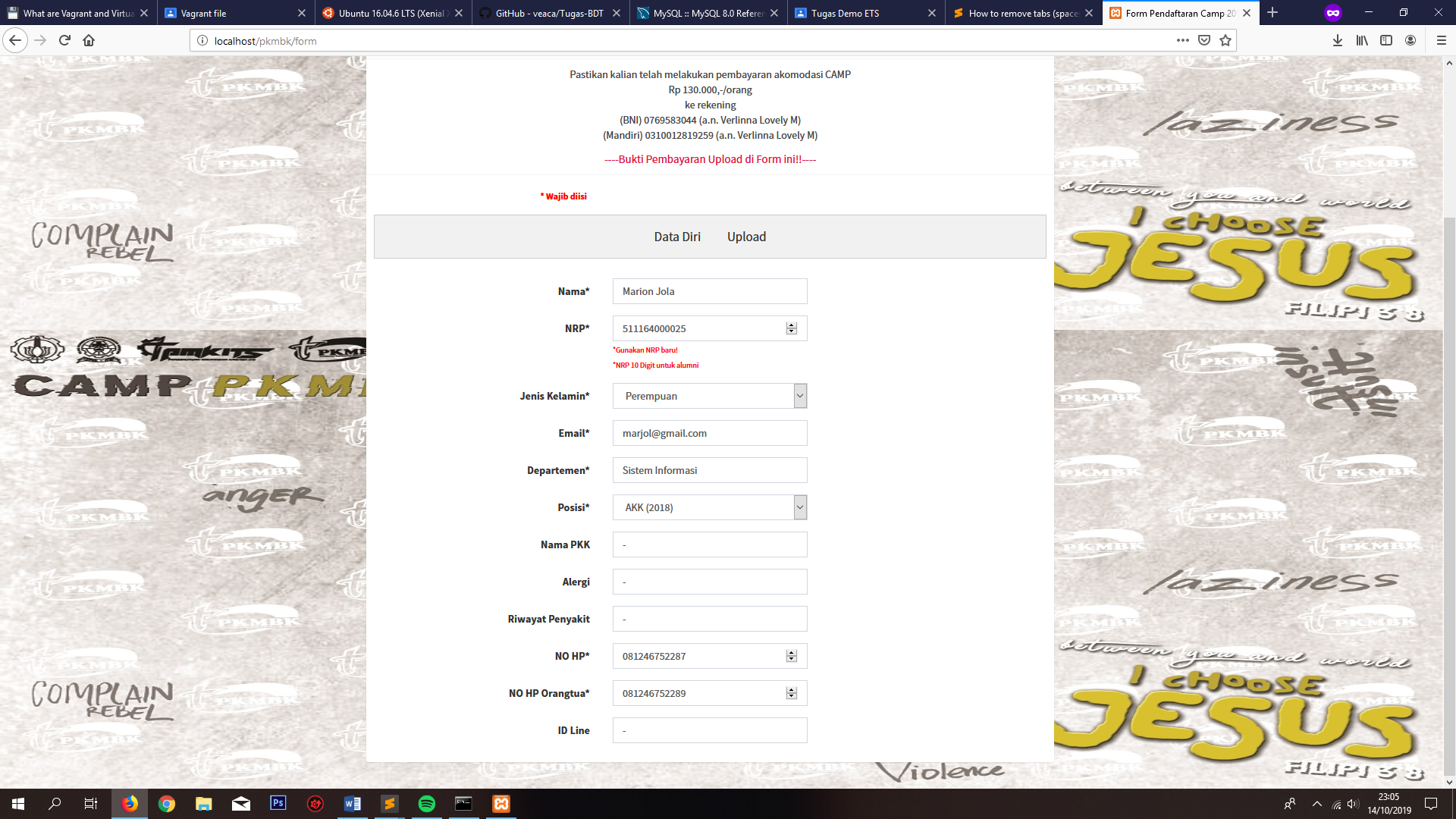


* + 1. Lalu migrate data untuk membuat tabel pada database yang telah diatur pada server.

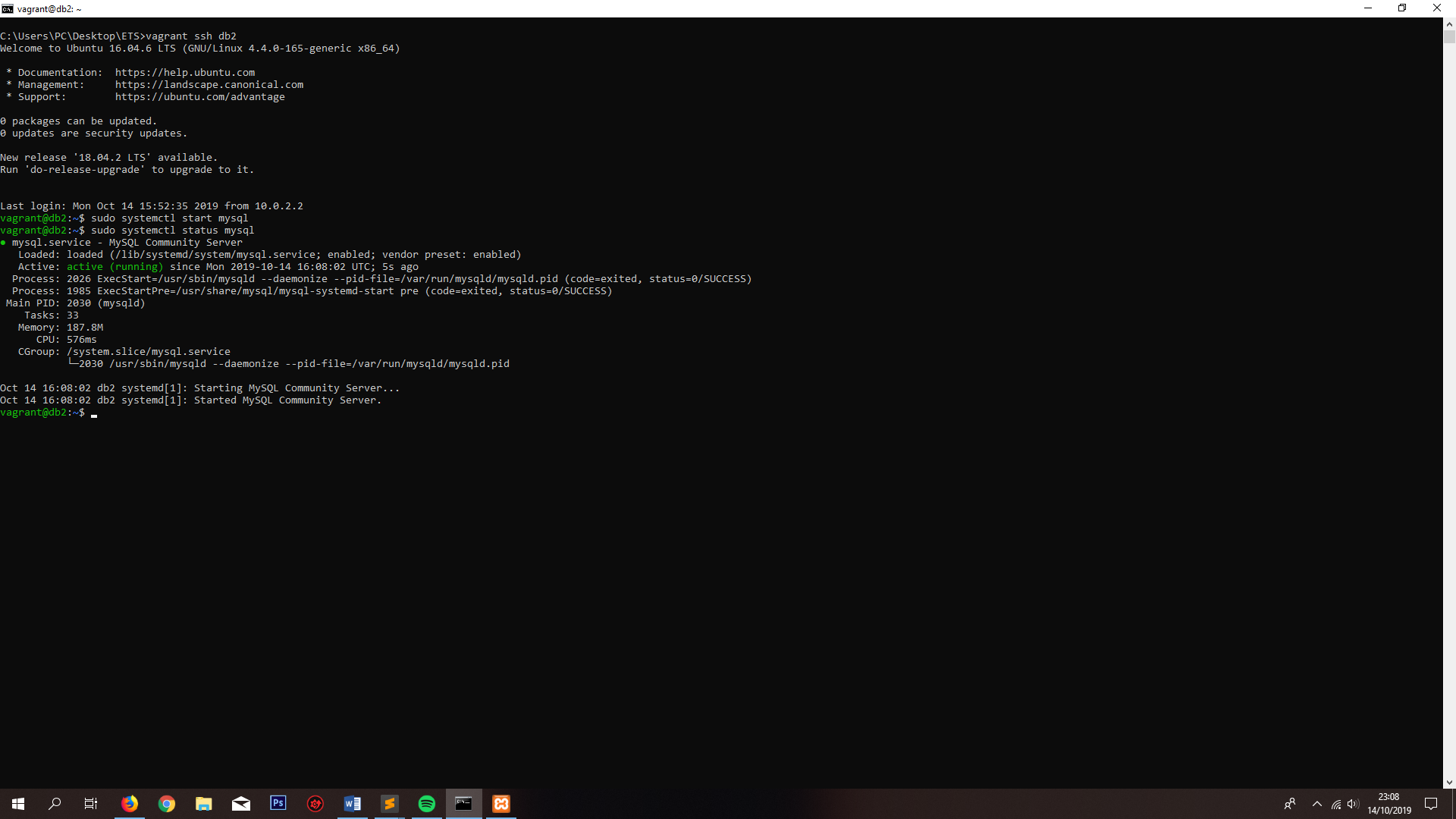
**Simulasi Fail-Over**

Langkat langkah simulasi :

1. Mematikan salah satu server, dalam hal ini yang dimatikan adalah mysql pada server db2
   1. Vagrant ssh db2
   2. Sudo systemctl stop mysql
   3. Sudo systemctl status mysql
2. Melakukan cek status server pada ProxySQL:
   1. Vagrant ssh proxy
   2. Mysql –u admin –p –h 127.0.0.1 –P 6032 –prompt=’ProxySQLAdmin>’
   3. Select hostgroup\_id, hostname, status, from runtime\_mysql\_servers;
3. Lakukan penambahan data melalui web.



1. Menyalakan kembali mysql pada db2 untuk melakukan cek apakah data terreplikasi ketika mysql kembali dihidupkan.
   1. Menyalakan kembali mysql db2
      1. Vagrant ssh db2
      2. Sudo systemctl start mysql
      3. Sudo systemctl status mysql



1. Verifikasi replikasi data dengan melakukan query.
   1. Mysql –u camppkmbk –p[password]
   2. Use camppkmbk;
   3. Select \* from pesertas