

Harjoitustyö 3

Testiympäristö Vagrantilla

Timo Lehosvuori, TTV18S1

Tatu Alatalo, TTV19SMM

Tuukka Bordin, TTV18S1

Harjoitustyö

Palveluiden automatisointi, Juho Pekki

Maaliskuu 2021

Tekniikan ja liikenteen ala

Tieto- ja viestintätekniikka (AMK)

Sisältö

1	Johdanto	1
2	Tehtävänanto	1
2.1	Vaihe 1	1
2.2	Vaihe 2	2
2.3	Toimivuuden parantaminen Vaiheen 2 jälkeen	7
2.4	Environment variables.....	7
2.4.1	Vagrant reload (Halt & up) parannuksia.....	8
2.4.2	Quality of Life parannukset	10
3	Pohdinta.....	11
4	Lähteet	13

Kuviot

Kuvio 1. Vagrantfile.....	2
Kuvio 2. docker_init.sh	3
Kuvio 3. mysql_dockerize.sh.....	3
Kuvio 4. wp_dockerize.sh	4
Kuvio 5. Vagrant up virtuaalikoneille.....	4
Kuvio 6. Docker asentuu MySQL virtuaalikoneeessa	5
Kuvio 7. MySQL hakeminen MySQL virtuaalikoneessa	5
Kuvio 8. Dockerin asennus WordPress virtuaalikoneille	6
Kuvio 9. WP-imagen hakeminen WordPress virtuaalikoneessa.....	6
Kuvio 10. Kontti käyntiin WordPress virtuaalikoneessa	6
Kuvio 11. Kontit käyntiin WordPress virtuaalikoneessa	7
Kuvio 12. Uudet ympäristö muuttujat käytössä.....	7
Kuvio 13. Positionaaliset argumentit tallennetaan muuttujiin skriptissä wp_dockerize.sh.....	8
Kuvio 14. Kontin käynnistys skripti käyttäen uusia muuttujia	8

Kuvio 15. Muokattu wp_dockerize.sh	9
Kuvio 16. WordPress-kontin entrypoint skriptin wp.sh muutokset	9
Kuvio 17. Muokatut if-lauseet skriptissä wp.sh	10
Kuvio 18. Boxien käynnistyksen yhteydessä pyörivä docker_init.sh muutokset..	11
Kuvio 19. WordPress-kontin entrypoint skriptin wp.sh pieni muutos.	11

1 Johdanto

Tämä harjoitustyö on osa Jyväskylän Ammattikorkeakoulun Palveluiden Automatisointi -kurssia. Harjoituksen tarkoituksena on tutustua HashiCorpin työkaluun nimeltä Vagrant. Harjoituksessa luodaan paikallinen usean virtuaalikoneen testiympäristö Vagrantilla.

2 Tehtävänanto

2.1 Vaihe 1

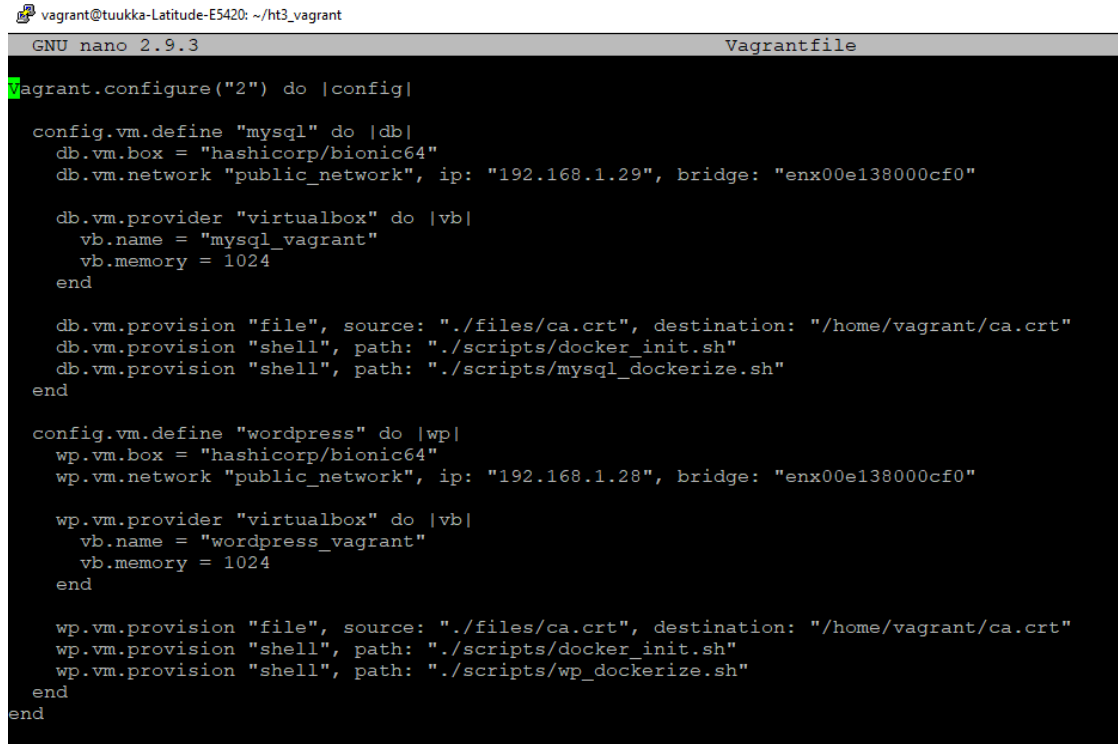
Vagrant on Hashicorpin vuonna 2010 kehittämä avoimen lähdekoodin ohjelmisto, joka on suunniteltu palveluiden automatisointiin. Sovelluksen avulla pystytään asentamaan virtuaalikäyttöjärjestelmä ja siihen halutut paketit/sovellukset. Vagrant tarvitsee virtualisointialustan toimiakseen (esim. Virtualbox, VMWare, AWS). Vagrant on kirjoitettu Ruby-ohjelmointikielellä, mutta sitä voidaan käyttää projekteissa eri ohjelmointikielien kanssa. (Wikipedia 2021)

Vagrant projektin aloittaminen on tehty helpoksi ja virtuaalikoneen saa lyötyä käyntiin kahdella komennolla (Vagrant: Quick Start N.d.). Käytettävissä olevia "Boxeja" (käyttöjärjestelmien imageja) on tarjolla Vagrantin omassa Vagrant Cloudissa. Halutessaan voi luoda itse omia Vagrant Boxeja valmiista Boxeista tai tehdä kokonaan uusia, ja sen jälkeen ladata ne Vagrant Cloudiin, tai johonkin omaan Vagrant Box repositorioon. (Vagrant: Creating a Base Box N.d; Vagrant Cloud N.d)

Vagrantilla voi käyttää eri provisioneja mm. asentamaan ohjelmia vagrant up komennon aikana. (Vagrant: Provisioning N.d) Harjoitustyön vaihe 2:ssa käytämme mm. File provisionia jolla siirsimme host koneelta cert-tiedoston luotuihin virtuaalikoneisiin sekä shell provisionia jolla ajoimme eri skriptejä. Lisäksi Dockerille, Ansiblelle löytyy myös omia provisionejaan.

2.2 Vaihe 2

Aloitimme tehtävän tekemällä Vagrantfilen, jossa määritellään asetukset MySQL:lle, WordPressille, virtuaalikoneille ja polut skripteille:



```
vagrant@tuukka-Latitude-E5420: ~/ht3_vagrant
GNU nano 2.9.3 Vagrantfile

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

  config.vm.define "mysql" do |db|
    db.vm.box = "hashicorp/bionic64"
    db.vm.network "public_network", ip: "192.168.1.29", bridge: "enx00e138000cf0"

    db.vm.provider "virtualbox" do |vb|
      vb.name = "mysql_vagrant"
      vb.memory = 1024
    end

    db.vm.provision "file", source: "./files/ca.crt", destination: "/home/vagrant/ca.crt"
    db.vm.provision "shell", path: "./scripts/docker_init.sh"
    db.vm.provision "shell", path: "./scripts/mysql_dockerize.sh"
  end

  config.vm.define "wordpress" do |wp|
    wp.vm.box = "hashicorp/bionic64"
    wp.vm.network "public_network", ip: "192.168.1.28", bridge: "enx00e138000cf0"

    wp.vm.provider "virtualbox" do |vb|
      vb.name = "wordpress_vagrant"
      vb.memory = 1024
    end

    wp.vm.provision "file", source: "./files/ca.crt", destination: "/home/vagrant/ca.crt"
    wp.vm.provision "shell", path: "./scripts/docker_init.sh"
    wp.vm.provision "shell", path: "./scripts/wp_dockerize.sh"
  end
end
```

Kuvio 1. Vagrantfile

Sertifikaatin siirron toteutimme Vagrantfile *file*-provisionilla. <https://www.vagrantup.com/docs/provisioning/file>. Asetimme Vagrantfilessä molemmille virtuaalikoneille IP-osoitteen bridged-tilaan. Myös kontit tarvitsevat tietoonsa nämä IP-osoitteet, joten loimme skriptit molemmille virtuaalikoneille, jotka hoitavat homman: *mysql_dockerize.sh* & *wp_dockerize.sh*. Selostamme nämä skriptit tarkemmin myöhemmin.

Tämän jälkeen teimme *docker_init.sh* skriptin, joka asentaa dockerin, lisää openrekisterin */etc/hosts* tiedostoon, siirtää sertifikaatit oikeaan paikkaan ja käynnistää dockerin uudestaan rekisteröidäkseen muutokset:

```

vagrant@tuukka-Latitude-E5420: ~/ht3_vagrant/scripts
GNU nano 2.9.3 docker_init.sh

# install docker
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
echo \
  "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt update -y && sudo apt install docker-ce docker-ce-cli containerd.io -y

# add openrekisteri to /etc/hosts
echo "35.228.57.182 openrekisteri.com" >> /etc/hosts

# move cert file to cert-storage
mkdir -p /etc/docker/certs.d/openrekisteri.com:443/
mv /home/vagrant/ca.crt /etc/docker/certs.d/openrekisteri.com:443/ca.crt

# restart docker to register changes
# sudo systemctl restart docker

```

Kuvio 2. docker_init.sh

Seuraava skripti, jonka loimme, hakee MySQL kontin opettajan rekisteristä, luo vo-
luumit ja käynnistää kontin:

```

vagrant@tuukka-Latitude-E5420: ~/ht3_vagrant
GNU nano 2.9.3 scripts/mysql_dockerize.sh

# sudo docker image pull openrekisteri.com:443/ryhma8_mysql

sudo docker volume create mysql_data
sudo docker volume create mysql_conf

# start container
sudo docker run \
  -dp 3306:3306 \
  -v mysql_conf:/etc/mysql/conf.d/ \
  -v mysql_data:/var/lib/mysql:rw \
  --name myslikontti \
  openrekisteri.com:443/ryhma8_mysql

```

Kuvio 3. mysql_dockerize.sh

Teimme vastaavan skriptin myös WordPressille:

```

vagrant@tuukka-Latitude-E5420: ~/ht3_vagrant/scripts
GNU nano 2.9.3 wp_dockerize.sh

sudo docker image pull openrekisteri.com:443/ryhma8_wp

sudo docker volume create wp

# start container
# need to re-set some environment variables due to changed ip addresses
sudo docker run \
-dp 80:80 \
-v wp:/var/www/html:rw \
-e DB_ADDR=192.168.1.29 \
-e WP_URL=192.168.1.28 \
--name wpkonttti \
openrekisteri.com:443/ryhma8_wp

```

Kuvio 4. wp_dockerize.sh

Skriptien luonnin jälkeen ajoimme *vagrant up* -komennon:

```

==> mysql: Destroying VM and associated drives...
vagrant@tuukka-Latitude-E5420:~/ht3_vagrant$ vagrant up
Bringing machine 'mysql' up with 'virtualbox' provider...
Bringing machine 'wordpress' up with 'virtualbox' provider...
==> mysql: Importing base box 'hashicorp/bionic64'...
==> mysql: Matching MAC address for NAT networking...
==> mysql: Checking if box 'hashicorp/bionic64' version '1.0.282' is up to date.
..
==> mysql: Setting the name of the VM: mysql_vagrant
==> mysql: Clearing any previously set network interfaces...
==> mysql: Preparing network interfaces based on configuration...
mysql: Adapter 1: nat
mysql: Adapter 2: bridged
==> mysql: Forwarding ports...
mysql: 22 (guest) => 2222 (host) (adapter 1)
==> mysql: Running 'pre-boot' VM customizations...
==> mysql: Booting VM...
==> mysql: Waiting for machine to boot. This may take a few minutes...
mysql: SSH address: 127.0.0.1:2222
mysql: SSH username: vagrant
mysql: SSH auth method: private key

```

Kuvio 5. Vagrant up virtuaalikoneille

Koneen käynnistyessä skriptit käynnistyvät. Alla MySQL-boxin skripti asentaa Dockerin.

```
mysql: (Reading database ... 65%
mysql: (Reading database ... 70%
mysql: (Reading database ... 75%
mysql: (Reading database ... 80%
mysql: (Reading database ... 85%
mysql: (Reading database ... 90%
mysql: (Reading database ... 95%
mysql: (Reading database ... 100%
mysql: (Reading database ...
mysql: 42512 files and directories currently installed.)
mysql: Preparing to unpack .../0-pigz_2.4-1_amd64.deb ...
mysql: Unpacking pigz (2.4-1) ...
mysql: Selecting previously unselected package containerd.io.
mysql: Preparing to unpack .../1-containerd.io_1.4.4-1_amd64.deb ...
mysql: Unpacking containerd.io (1.4.4-1) ...
mysql: Selecting previously unselected package docker-ce-cli.
mysql: Preparing to unpack .../2-docker-ce-cli_5%3a20.10.5~3-0~ubuntu-bionic
_amd64.deb ...
mysql: Unpacking docker-ce-cli (5:20.10.5~3-0~ubuntu-bionic) ...
mysql: Selecting previously unselected package docker-ce.
mysql: Preparing to unpack .../3-docker-ce_5%3a20.10.5~3-0~ubuntu-bionic_amd
64.deb ...
mysql: Unpacking docker-ce (5:20.10.5~3-0~ubuntu-bionic) ...
```

Kuvio 6. Docker asentuu MySQL virtuaalikoneessa

Skripti hakee MySQL kontin:

```
mysql: Verifying Checksum
mysql: f3721839f3a0:
mysql: Download complete
mysql: 0624c9c3aad2:
mysql: Verifying Checksum
mysql: 0624c9c3aad2: Download complete
mysql: 075626307450: Verifying Checksum
mysql: 075626307450: Download complete
mysql: 68bbd8eb6c49:
mysql: Verifying Checksum
mysql: 68bbd8eb6c49:
mysql: Download complete
mysql: 69098fc75611:
mysql: Verifying Checksum
mysql: 69098fc75611:
mysql: Download complete
mysql: 8634d847d477:
mysql: Verifying Checksum
mysql: 8634d847d477:
mysql: Download complete
mysql: 3d201307e5c6:
mysql: Verifying Checksum
mysql: 3d201307e5c6: Download complete
```

Kuvio 7. MySQL hakeminen MySQL virtuaalikoneessa

Skripti asentaa Dockerin WordPress-boxissa:


```

wordpress: Selecting previously unselected package libltdl7:amd64.
wordpress: Preparing to unpack .../5-libltdl7_2.4.6-2_amd64.deb ...
wordpress: Unpacking libltdl7:amd64 (2.4.6-2) ...
wordpress: Setting up containerd.io (1.4.4-1) ...
wordpress: Created symlink /etc/systemd/system/multi-user.target.wants/conta
inerd.service → /lib/systemd/system/containerd.service.
wordpress: Processing triggers for ureadahead (0.100.0-21) ...
wordpress: Processing triggers for libc-bin (2.27-3ubuntu1) ...
wordpress: Processing triggers for systemd (237-3ubuntu10.25) ...
wordpress: Setting up libltdl7:amd64 (2.4.6-2) ...
wordpress: Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
wordpress: Setting up docker-ce-cli (5:20.10.5~3-0~ubuntu-bionic) ...
wordpress: Setting up pigz (2.4-1) ...
wordpress: Setting up docker-ce (5:20.10.5~3-0~ubuntu-bionic) ...
wordpress: Created symlink /etc/systemd/system/multi-user.target.wants/docke
r.service → /lib/systemd/system/docker.service.
wordpress: Created symlink /etc/systemd/system/sockets.target.wants/docker.s
ocket → /lib/systemd/system/docker.socket.
wordpress: Setting up docker-ce-rootless-extras (5:20.10.5~3-0~ubuntu-bionic

```

Kuvio 8. Dockerin asennus WordPress virtuaalikoneille

Skripti hakee WordPress konttia:

```

wordpress: 154729f6ba86: Pulling fs layer
wordpress: 31213de2d322: Pulling fs layer
wordpress: aef10437549e: Pulling fs layer
wordpress: 3bb2e8051594:
wordpress: Waiting
wordpress: 4c761b44e2cc: Waiting
wordpress: c2199db96575: Waiting
wordpress: 1b9a9381eea8: Waiting
wordpress: fd07bbc59d34: Waiting
wordpress: 72b73ab27698: Waiting
wordpress: 983308f4f0d6: Waiting
wordpress: 6c13f026e6da: Waiting
wordpress: e5e6cd163689:
wordpress: Waiting
wordpress: 5c5516e56582: Waiting
wordpress: 154729f6ba86: Waiting
wordpress: 31213de2d322: Waiting
wordpress: aef10437549e: Waiting
wordpress: db606474d60c:
wordpress: Verifying Checksum
wordpress: db606474d60c: Download complete
wordpress: 3bb2e8051594: Verifying Checksum
wordpress: 3bb2e8051594: Download complete

```

Kuvio 9. WP-imagen hakeminen WordPress virtuaalikoneessa

Skripti käynnistää kontin:

```

wordpress: Digest: sha256:acee1c2b61e29713333ef62190d76b780b7bd20d25e7bfda5b
2b1ce7fce9c1e1
wordpress: Status: Downloaded newer image for openrekisteri.com:443/ryhma8_w
p:latest
wordpress: openrekisteri.com:443/ryhma8_wp:latest
wordpress: wp
wordpress: 6c8e55ac11ab5959c3e2fb8e6433949caaf1fad49cdd28a4dceda4c45bfe9b94
vagrant@tuukka-Latitude-E5420:~/ht3_vagrant$

```

Kuvio 10. Kontti käyntiin WordPress virtuaalikoneessa

2.3 Toimivuuden parantaminen Vaiheen 2 jälkeen

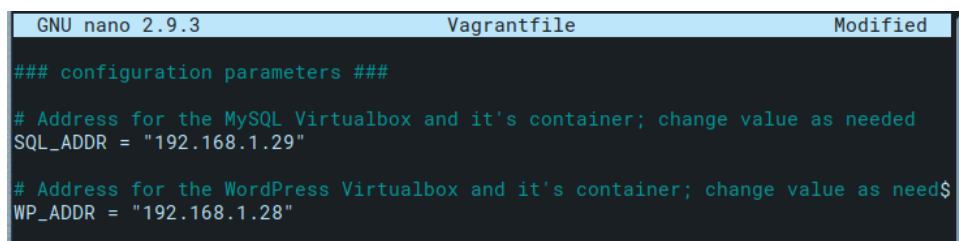
Tässä kappaleessa on tehty ns. Extrajuttuja, joita ei ole vaadittu tehtävänannossa, mutta jotka ovat tarpeellisia tiettyjen toiminnallisuuksien toimivuuden vuoksi.

Ideamme oli seuraava:

- Saada kontit pystyyn myös siinä tapauksessa, että virtuaalikoneet sammutettaisiin esim. yön yli. Eli että ei vaadittaisi *vagrant destroy* komentoa, että kontit menisivät käyntiin
- Määritellä environment variablet, joita kontti käyttää, suoraan Vagrantfilessä. Näin ne olisi määritelty yhdessä paikassa monen paikan sijaan.
- Tehdä tarvittavat muutokset näitä varten tarvittavissa paikoissa.

Lisäksi pyrimme tekemään joitain pieniä muutoksia. Esimerkiksi sen, että skripti ei yritä asentaa Dockeria aina joka käynnistyksen yhteydessä ja muita Quality-of-Life parannuksia.

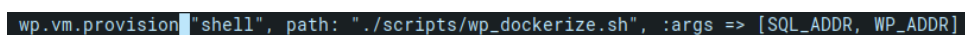
2.4 Environment variables



```
GNU nano 2.9.3 Vagrantfile Modified
### configuration parameters ###
# Address for the MySQL Virtualbox and it's container; change value as needed
SQL_ADDR = "192.168.1.29"
# Address for the WordPress Virtualbox and it's container; change value as need$
WP_ADDR = "192.168.1.28"
```

Kuvio 11. Kontit käyntiin WordPress virtuaalikoneessa

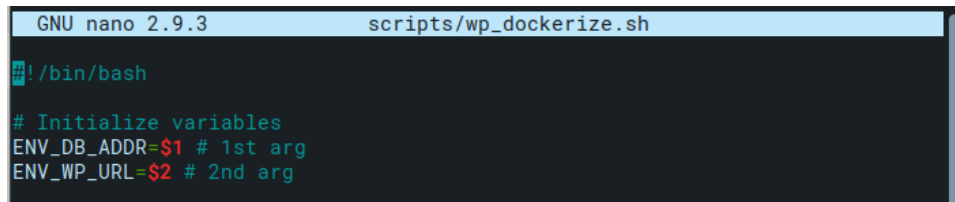
Kuvassa määritelty 2 environment variablea Dockeria varten, joita käytetään myös Vagrantfilessä itsessään IP-osoitteiden asettamiseen.



```
wp.vm.provision "shell", path: "./scripts/wp-dockerize.sh", :args => [SQL_ADDR, WP_ADDR]
```

Kuvio 12. Uudet ympäristö muuttujat käytössä

Vagrantfile syöttää määritellyt variabellet skriptille. Skripti tallentaa sen jälkeen variabellet.



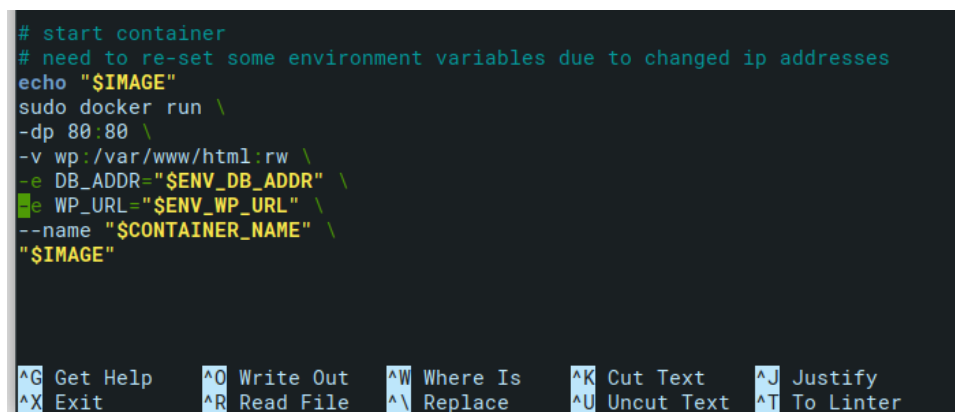
```
GNU nano 2.9.3 scripts/wp_dockerize.sh

#!/bin/bash

# Initialize variables
ENV_DB_ADDR=$1 # 1st arg
ENV_WP_URL=$2 # 2nd arg
```

Kuvio 13. Positionaaliset argumentit tallennetaan muuttujiin skriptissä *wp_dockerize.sh*

Viimeiseksi skripti syöttää variabellet kontin luonnin yhteydessä environment variabelleina.



```
# start container
# need to re-set some environment variables due to changed ip addresses
echo "$IMAGE"
sudo docker run \
  -dp 80:80 \
  -v wp:/var/www/html:rw \
  -e DB_ADDR="$ENV_DB_ADDR" \
  -e WP_URL="$ENV_WP_URL" \
  --name "$CONTAINER_NAME" \
  "$IMAGE"
```

Kuvio 14. Kontin käynnistys skripti käyttäen uusia muuttujia

2.4.1 Vagrant reload (Halt & up) parannuksia

Jos esimerkiksi tietokanta on siirretty toiseen osoitteeseen ja uusi osoite halutaan siirtää WordPressin kontin tietoon ilman *vagrant destroy* -komentoa, niin seuraavat skriptin pätkät ovat hyödyllisiä:

```

GNU nano 2.9.3          scripts/wp_dockerize.sh

# Checks to see if container is running
echo "Checking if container is running..."
if [ $(sudo docker ps -q -l -f name="$CONTAINER_NAME") != "" ]; then
    # The old volume will remount. Possible re-creation of the wp-config.php
    # file will be done by the image itself via the updated wp.sh (24.3.2021)
    echo "Found container. Stopping and deleting $CONTAINER_NAME..."
    sudo docker container stop "$CONTAINER_NAME"
    sudo docker container prune --force
else
    echo "No containers found. Downloading a new image \"$IMAGE\"..."
    # Assuming the container has not ever been created, if it was not running
    sudo docker image pull openrekisteri.com:443/ryhma8_wp
    sudo docker volume create wp
fi

```

Kuvio 15. Muokattu wp_dockerize.sh

wp_dockerize.sh-skripti, jonka Vagrant pyörittää aina WordPress-virtuaalikoneen käynnistyksen yhteydessä, tarkistaa vanhan kontin olemassaolon ja poistaa sen tarvittaessa. Näin uusi kontti voidaan käynnistää uusilla arvoilla. Uusilla arvoilla käynnistäminen tapahtuu saman skriptin lopussa kuviossa 13 kuvattuna.

Lisäksi kontissa itsessään piti olla muutamia tarkistuksia siinä tapauksessa, että kontti käynnistetään uudelleen, mutta volume ei ole tyhjä. Tässä tapauksessa volumen *wp-config.php* voi sisältää vääriä arvoja:

```

GNU nano 2.9.3          wp.sh          Modified

# Taulukko environment variableiden muutosten tarkastusta varten
declare -A conf_vars=(['DB_NAME']=$DB_DATABASE ['DB_USER']=$DB_USER \
['DB_PASSWORD']=$DB_PW ['DB_HOST']=$DB_ADDR ['DB_CHARSET']=$DB_CHARSET \
['DB_COLLATE']=$DB_COLLATE )

conf_changed=0

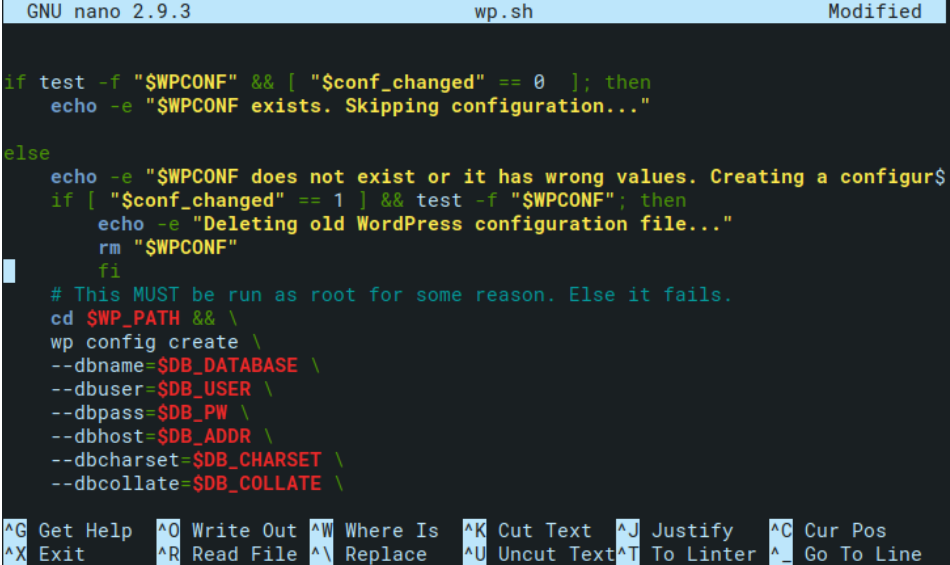
# Tarkastus Vagrantia varten. Jos palvelu käynnistetään Vagrantin skriptillä
# ja vanhaa konttia ei tuhota, tämä varmistaa, että conf luodaan uusiksi,
# jos se sisältää vanhentuneita arvoja.
# Tämä varmistaa myös sen, että conf muuttuu, vaikka volume olisikin populoitu.
# Eli voluumin siirto helpottuu.
for key in "${!conf_vars[@]}"; do
    old_val=$(wp config get "$key" --allow-root)
    new_val=${conf_vars["$key"]}
    if [ "$new_val" != "$old_val" ]; then
        conf_changed=1
        break;
    fi
done

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File ^_ Replace   ^U Uncut Text ^T To Linter ^_ Go To Line

```

Kuvio 16. WordPress-kontin entrypoint skriptin *wp.sh* muutokset

Yllä olevassa kuviossa *wp.sh* -joka pyörii kontin sisällä aina käynnistyksen yhteydessä-tarkistaa, vastaavatko environment variableiden arvot niitä arvoja, jotka ovat *wp-config.php* -tiedostossa. Jos eivät, *conf_changed* -muuttuja asetetaan arvoon 1:



```

GNU nano 2.9.3 wp.sh Modified

if test -f "$WPCONF" && [ "$conf_changed" == 0 ]; then
    echo -e "$WPCONF exists. Skipping configuration..."
else
    echo -e "$WPCONF does not exist or it has wrong values. Creating a configur$
    if [ "$conf_changed" == 1 ] && test -f "$WPCONF"; then
        echo -e "Deleting old WordPress configuration file..."
        rm "$WPCONF"
    fi
    # This MUST be run as root for some reason. Else it fails.
    cd $WP_PATH && \
    wp config create \
    --dbname=$DB_DATABASE \
    --dbuser=$DB_USER \
    --dbpass=$DB_PW \
    --dbhost=$DB_ADDR \
    --dbcharset=$DB_CHARSET \
    --dbcollate=$DB_COLLATE \
    ^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
    ^X Exit ^R Read File ^_ Replace ^U Uncut Text ^T To Linter ^_ Go To Line

```

Kuvio 17. Muokatut if-lauseet skriptissä *wp.sh*

HT-2 verrattuna *wp.sh* if-lausetta piti muuttaa siten, että se otti huomioon muuttujan *conf_changed*. Lisäksi uudessa versiossa poistetaan vanha *wp-config.php*, jotta uuden conf-tiedoston luonti on mahdollista.

2.4.2 Quality of Life parannukset

Tämä kappale sisältää pieniä muutoksia, jotka nopeuttavat ohjelman toimintaa tai muuten vaan "helpottavat elämää".

```

GNU nano 2.9.3          scripts/docker_init.sh          Modified
# install docker
if [ -x "$(command -v docker)" ]; then
    echo 'Docker is installed. Skipping installation.'
else
    echo "Installing Docker..."
    curl -fsSL https://download.docker.com/linux/ubuntu/gpg \
    | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
    echo \
    "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] \
    https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
    sudo apt update -y && sudo apt install docker-ce docker-ce-cli containerd.io -y

    # add openrekisteri to /etc/hosts
    echo "35.228.57.182 openrekisteri.com" >> /etc/hosts
    # move cert file to cert-storage
    mkdir -p /etc/docker/certs.d/openrekisteri.com:443/
    mv /home/vagrant/ca.crt /etc/docker/certs.d/openrekisteri.com:443/ca.crt

    # restart docker to register changes
    sudo systemctl restart docker

```

Kuvio 18. Boxien käynnistyksen yhteydessä pyörivä *docker_init.sh* muutokset

Kuviossa yllä Dockeria ei yritetä asentaa uudelleen, jos se on jo asennettu. Tämä komento on provisioitu Vagrantfilessä pyörimään aina virtuaalikoneen käynnistyksen yhteydessä.

```

# set apache as main process to prevent exit
echo -e "Starting main process..."
/usr/sbin/apache2ctl stop
sleep 3
/usr/sbin/apache2ctl -D FOREGROUND;

```

Kuvio 19. WordPress-kontin entrypoint skriptin *wp.sh* pieni muutos.

Docker (HT-2) WordPressin imagen *wp.sh* sammuttaa nyt Apachen ennen sen uudelleen käynnistämistä. Tämä vähentää virheviestien määrää.

3 Pohdinta

Lähdimme tekemään tätä harjoitustyötä samalla periaatteella, minkä totesimme toimivaksi harjoitustyö 2:ssa, eli Bash-skriptien hyväksikäytöllä. Tämän tehtävän olisi

voinut tehdä myös Vagrantin provisioneilla, eli esimerkiksi Dockerin olisi voinut määrittää suoraan näiden avulla ilman skriptejä. Mutta koska ratkaisumme toimii emme lähteneet sitä enää muuttamaan.

Lisäksi meillä oli jostain syystä suuria ongelmia saada Word toimimaan. Word ikään kuin ”räjähti” yhdessä kohtaa ja kaikki tyylit olivat pilalla. Niiden takaisin saaminen oli melkein mahdotonta, sillä Word kieltäytyi kaikesta yhteistyöstä. Kaiken kaikkiaan Word-ongelmien selvittelyyn meni melkein yhtä paljon aikaa kuin harjoitustyön tekemiseen ylipäätään.

4 Lähteet

Vagrant Cloud. N.d. Vagrant cloud sivusto, josta voi ladata Vagrant Boxeja. Viitattu 18.3.2020. <https://app.vagrantup.com/boxes/search>.

Vagrant: Creating a Base Box. N.d. Vagrantin dokumentaatio kuinka oma Vagrant Box luodaan. Viitattu 18.3.2020. <https://www.vagrantup.com/docs/providers/virtual-box/boxes#packaging-the-box>.

Vagrant: Provisioning. N.d. Vagrantin dokumentaatio provisioneista. Viitattu 24.3.2021. <https://www.vagrantup.com/docs/provisioning#provisioning>.

Vagrant: Quick Start. N.d. Vagrantin dokumentaatio kuinka projekti saadaan nopeasti käyntiin. Viitattu 18.3.2020. <https://learn.hashicorp.com/tutorials/vagrant/getting-started-index?in=vagrant/getting-started>.

Wikipedia. 12.2.2021. Wikipedian artikkeli Vagrant-ohjelmistosta. Viitattu 18.3.2021. <https://app.vagrantup.com/boxes/search>.