

# **Tugas MK. Jarkom**

## ***“Instalasi Mesin Virtualisasi Docker dan Web Ecommerce”***

Dosen Pengampu : R. Dimas Adityo



Disusun Oleh :

Oktaviana Isbirotin (1914311009)

M. Faizal Nazili (1914311022)

Aditya Deddy Ruspratama (1914311032)

Farah Distya Elvariza (1914311042)

Nur Lailatul Mukaromah (1914311053)

Bayu Pratama (1914311058)

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNIK  
UNIVERSITAS BHAYANGKARA SURABAYA**

**2020**

# KATA PENGANTAR

Puji syukur saya ucapkan kehadirat Allah SWT yang telah memberikan rahmat serta hidayah kepada kita semua, sehingga berkat Karunia-Nya penulis dapat menyelesaikan makalah ini guna memenuhi Tugas Akhir UAS untuk mata kuliah Jaringan Komputer, dengan judul *“Instalasi Mesin Virtualisasi Docker dan Web Ecommerce”*.

Saya menyadari bahwa dalam penulisan makalah ini tidak terlepas dari bantuan banyak pihak yang dengan tulus memberikan doa, saran dan kritik sehingga makalah ini dapat terselesaikan.

Saya menyadari sepenuhnya bahwa makalah ini masih jauh dari sempurna dikarenakan terbatasnya pengetahuan yang saya miliki. Oleh karena itu, kami mengharapkan segala bentuk saran serta masukan bahkan kritik yang membangun dari berbagai pihak. Dalam penyusunan makalah ini penulis berharap semoga makalah ini dapat bermanfaat bagi penulis sendiri maupun kepada pembaca umumnya.

Penyusun

# Daftar Isi

KATA PENGANTAR .....	2
BAB I.....	4
PENDAHULUAN .....	4
1. 1 Latar Belakang.....	4
1.2 Rumusan Masalah.....	4
1.3 Tujuan Masalah .....	4
BAB II.....	5
PEMBAHASAN.....	5
2.1. Instalasi Docker di Ubuntu .....	5
2.2 Penggunaan Container dalam Docker .....	6
2.3 Instalasi WordPress di Ubuntu Docker .....	8
BAB III .....	18
PENUTUP .....	18
3.1. Kesimpulan .....	18
3.2. Saran .....	18

# BAB I

## PENDAHULUAN

### ***1.1 Latar Belakang***

Aplikasi berbasis web merupakan aplikasi yang mempermudah seseorang untuk mencari informasi, bertransaksi, atau melakukan hal lainnya melalui layanan internet. Aplikasi berbasis web sangat mudah untuk digunakan karena dapat diakses di berbagai platform computer hanya dengan menjalankan web browser.

Dengan kemajuan pengetahuan dalam ilmu komputer, berbagai macam bahasa pemrograman pun telah banyak dibuat untuk menjalankan sebuah aplikasi berbasis web. Bahasa pemrograman yang digunakan antara lain yaitu *php*.

Dengan perkembangan teknologi yang begitu pesat ini, pembuatan aplikasi berbasis web pun ikut mengalami perkembangan dalam hal arsitektur. Arsitektur yang digunakan dalam pembuatan web bernama *Microservice*. *Microservice* adalah sebuah arsitektur dimana aplikasi dipecah-pecah menjadi service-service kecil dengan menggunakan bahasa pemrograman yang sama atau berbeda. Tujuan dari *Microservice* adalah untuk membuat sebuah aplikasi yang mampu berjalan secara independent.

*Docker* sangat mendukung untuk pembuatan *Microservice*, dengan menggunakan *docker* pembuatan aplikasi dan memasukkannya menjadi sebuah image yang nanti dijalankan menjadi sebuah *container*. Adanya *Microservice* membuat aplikasi *website* lebih mudah karena dapat membagi service menjadi kecil-kecil. Akan tetapi dengan adanya hal itu akan mempengaruhi *server* yang bekerja sebagai penyedia layanan dari aplikasi web tersebut serta bahasa pemrograman yang digunakan dalam pembuatan *website* akan mempengaruhi juga.

Pada tugas ini akan dilakukan terhadap bahasa pemrograman *php* dengan menggunakan *docker container*.

### ***1.2 Rumusan Masalah***

1. Bagaimana cara instalasi *Docker* di Ubuntu 18.04?
2. Bagaimana cara menjalankan *Container* di *Docker* ?
3. Bagaimana cara instalasi *WordPress* di Ubuntu 18.04 LTS?

### ***1.3 Tujuan Masalah***

1. Untuk mengetahui cara instalasi *Docker* di Ubuntu 18.04
2. Untuk mengetahui bagaimana menjalankan *container* di *docker*.
3. Untuk membangun aplikasi webcommerce menggunakan *WordPress*.

## BAB II

### PEMBAHASAN

#### 2.1. Instalasi Docker di Ubuntu

Jika menyangkut penyebaran Docker, Linux merupakan sistem operasi yang direkomendasikan. Hal ini disebabkan suatu fakta bahwa awalnya Docker dirilis di Linux pada tahun 2013. Instalasi Docker di distribusi Linux yang berbeda akan memberi hasil yang berbeda pula. Docker tidak ada dalam repositori resmi Ubuntu 18.04. Namun, hal tersebut bukan menjadi halangan dalam proses install Docker di Ubuntu. Dibawah ini adalah langkah-langkahnya:

##### 1. Tambahkan repositori docker

Anda harus menambahkan repositori Docker. Langkah ini akan membuat proses instalasi lebih mudah. Pertama-tama tambahkan GPG key dari official repository Docker ke dalam system:

```
curl -fsSL https://download.Docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

Untuk menambahkan repositori maka commandnya seperti ini:

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

Lalu, update informasi repositori:

```
sudo apt-get update
```

##### 2. Install Docker Ubuntu

Setelah semua selesai di update maka langkah terakhir adalah menginstal docker dengan command:

```
sudo apt-get install -y docker-ce
```

setelah penginstalan sudah selesai, kita bisa periksa status dockernya dengan command:

```
sudo systemctl status docker
```

jika ingin mengetahui apa saja command di dalam docker, bisa di tulis command berikut:

```
sudo docker-subcommand --help
```

## 2.2 Penggunaan Container dalam Docker

Sebuah Docker container berjalan menggunakan Docker image. Secara default, Docker images ditarik dari Docker Hub, sebuah Docker registry official dari Docker. Siapapun bisa membuat dan menaruh Docker images mereka dalam Docker Hub.

### 1. Membuat Container Image dari Docker Hub

Untuk memastikan apakah Anda bisa mengakses dan mendownload image dari docker hub, maka jalankan command berikut:

```
docker run hello-world
```

Jika berhasil maka tampilannya akan seperti ini:

```
ubuntu@server:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:f9dfddf63636d84ef479d645ab5885156ae030f611a56f3a7ac7f2fdd86d7e4e
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash
```

Jika ingin mencari image apa saja yang tersedia di Docker Hub, Anda bisa menggunakan command:

```
docker search [image yang ingin di cari]
```

lalu download imagenya:

```
docker pull ubuntu
```

ketika sudah di download dan menjalankan image, Anda bisa mengecek image dengan command:

```
docker images
```

dan outputnya akan seperti dibawah ini:

```
ubuntu@server:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu              latest             72300a873c2c      3 weeks ago       64.2MB
hello-world         latest             fce289e99eb9      14 months ago     1.84kB
ubuntu@server:~$
```

Untuk membuat Container di image yang sudah di download maka tulis command:

```
docker run ubuntu
```

## 2. Menjalankan Container Docker

Sekarang jalankan command berikut untuk menjalankan sebuah container dengan Ubuntu image :

```
docker run -it ubuntu
```

Command run berarti membuat sebuah container baru dengan image ubuntu. Option -it (gabungan dari -i dan -t) akan memberikan akses shell ke container baru yang akan kamu buat. Dan akan menampilkan DI container dan image yang telah dijalankan.

Lalu masuk ke shell docker ubuntu yang sudah di jalankan tadi dan install package Nodejs dengan commend dibawah ini:

```
apt-get update
apt-get install nodejs -y
```

setelah itu cek nodejs dengan command:

```
node -v
```

lalu ketik exit untuk keluar dari command

## 3. Push perubahan ke Docker Hub

Langkah ini adalah bagaimana kamu bisa menaruh image baru kamu ke Docker Hub, sehingga orang lain bisa menggunakan image yang sudah kamu buat. Sebelum lanjut, pastikan kamu sudah membuat akun Docker Hub.

Pertama login ke akun Docker Hub kamu dengan command :

```
docker login
```

Pastikan username pada nama image kamu sama dengan username Docker Hub.

Lalu jalankan docker commit untuk mengubah container menjadi images baru dengan command

```
docker commit <idcontainer> <username>/<image_name>:tag
```

Lalu, push image ke Docker Hub dengan command :

```
docker push <username>/<image_name>:tag
```

Jika berhasil maka akan muncul gambar Image docker yang sudah Anda Push.

### **2.3 Instalasi WordPress di Ubuntu Docker**

WordPress adalah perangkat lunak sumber terbuka yang dapat digunakan siapa saja untuk membuat situs web yang kuat dan dinamis berdasarkan PHP. Banyak situs web besar dan populer yang menjalankan WordPress secara online.

Tutorial singkat ini akan menunjukkan kepada siswa dan pengguna baru cara menginstal WordPress di Ubuntu 18.04 LTS dengan dukungan Nginx, MariaDB dan PHP-FPM. Berikut langkah langkahnya:

#### **1. Siapkan dan perbarui UBUNTU**

Sebelum menjalankan paket ada baiknya untuk selalu memperbarui server Ubuntu. Untuk memperbarui Ubuntu jalankan perintah dibawah ini:

```
apt update && apt dist-upgrade && apt autoremove
```

Setelah memperbarui Ubuntu, lanjutkan dibawah ini dengan menginstal paket paket yang diperlukan agar Wordpress berfungsi.

#### **2. Instal beberapa utility**

Jalankan perintah di bawah ini untuk menginstal beberapa utility yang nantinya digunakan untuk membantu kita mengkonfigurasi

```
Apt install net-tools  
Apt install dnsutils  
Apt install inetutils-ping  
Apt install nano  
Apt install wget  
Apt install vim
```



### 3. Instal Web Server Nginx

Setelah memperbarui Ubuntu, jalankan perintah dibawah ini untuk menginstal Nginx HTTP server.

```
Apt install nginx
```

Setelah menginstal Nginx, perintah dibawah ini dapat digunakan untuk berhenti, memulai dan mengaktifkan layanan Nginx.

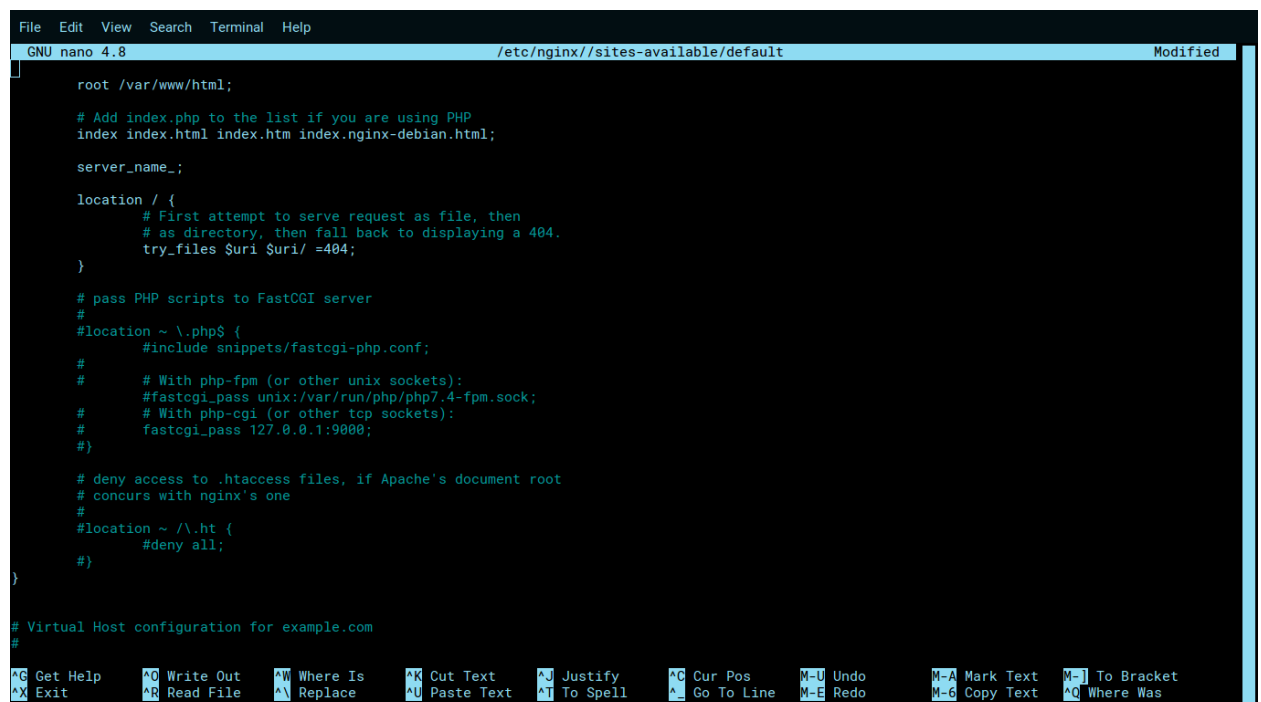
```
Service nginx start  
Service nginx stop  
Service nginx restart  
Service nginx status
```

### 4. Konfigurasi Nginx

Selanjutnya, konfigurasi file konfigurasi situs Wordpress di server. Selanjutnya, konfigurasi file konfigurasi situs Wordpress di server. Jalankan perintah di bawah ini untuk membuka dan mengedit file konfigurasi Nginx

```
nano /etc/nginx/sites-available/default
```

nanti akan muncul tampilan sebagai berikut :



```
File Edit View Search Terminal Help  
GNU nano 4.8 /etc/nginx/sites-available/default Modified  
root /var/www/html;  
  
# Add index.php to the list if you are using PHP  
index index.html index.htm index.nginx-debian.html;  
  
server_name _;  
  
location / {  
    # First attempt to serve request as file, then  
    # as directory, then fall back to displaying a 404.  
    try_files $uri $uri/ =404;  
}  
  
# pass PHP scripts to FastCGI server  
#  
#location ~ /\.php$ {  
#include snippets/fastcgi-php.conf;  
#  
# With php-fpm (or other unix sockets):  
#fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;  
# With php-cgi (or other tcp sockets):  
#fastcgi_pass 127.0.0.1:9000;  
#}  
  
# deny access to .htaccess files, if Apache's document root  
# concurs with nginx's one  
#  
#location ~ /\.ht {  
#deny all;  
#}  
}  
  
# Virtual Host configuration for example.com  
#  
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos ^M-U Undo ^M-A Mark Text ^M-] To Bracket  
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line ^M-E Redo ^M-G Copy Text ^Q Where Was
```

hapus beberapa tagar untuk mengaktifkan konfigurasi. Samakan dengan tampilan dibawah ini.

```
File Edit View Search Terminal Help
GNU nano 4.8 /etc/nginx/sites-available/default Modified
# include snippets/snakeoil.conf;

root /var/www/html;

# Add index.php to the list if you are using PHP
index index.php index.html index.htm index.nginx-debian.html;

server_name <your_ip_address>;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}

# pass PHP scripts to FastCGI server
#
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    #
    # With php-fpm (or other unix sockets):
    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
    #
    # With php-cgi (or other tcp sockets):
    fastcgi_pass 127.0.0.1:9000;
}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
location ~ /\.ht {
    deny all;
}

# Virtual Host configuration for example.com

#G Get Help #O Write Out #W Where Is #K Cut Text #J Justify #C Cur Pos #U Undo #A Mark Text #] To Bracket
#X Exit #R Read File #V Replace #U Paste Text #T To Spell #G Go To Line #E Redo #G Copy Text #_ Where Was
```

Jika sudah selesai, simpan perubahan dengan cara tekan CTRL + X lalu tekan Y lalu Enter

## 5. Konfigurasi File Hosts

File hosts adalah salah satu dari beberapa fasilitas sistem yang membantu dalam menangani node jaringan dalam jaringan computer. Disini saya menggunakan system operasi linux Ubuntu 16.04,

- buka terminal baru ( CTRL + ALT + T ) dan jalankan perintah dibawah ini untuk melihat ip address kalian yang nantinya akan di tambahkan di file hosts PC kalian

```
ifconfig
```

- setelah itu jalankan perintah dibawah ini untuk membuka dan mengedit file Hosts kalian

```
nano /etc/hosts
```

tambahkan ip address kalian ke dalam file hosts tersebut dan simpan perubahannya dengan cara tekan CTRL + X, lalu tekan Y lalu enter

## 6. Install MARIADB database

MariaDB adalah server default pada sebagian besar distribusi linux dan wordpress memerlukan server database, jalankan perintah dibawah ini untuk menginstal MariaDB

```
Apt install mariadb-client mariadb-server
```

Setelah menginstal, perintah dibawah ini dapat digunakan untuk berhenti, memulai dan mengaktifkan layanan MariaDB.

```
Service mysql start  
Service mysql stop  
Service mysql restart  
Service mysql status
```

## 7. Instal PHP dan Modul terkait

Setelah Nginx dan MariaDb diinstal, jalankan perintah dibawah ini untuk menginstal PHP dan modul PHP terkait di server

```
apt install php-fpm php-mysql php-cli php-common php-xml php-curl php-intl php-gd  
php-ldap php-mbstring php-xmldrpc
```

Setelah menginstal PHP, jalankan perintah dibawah ini untuk membuka file konfigurasi default PHP.

```
Nano /etc/php/7.4/fpm/php.ini
```

Kemudian gulir kebawah baris dalam file dan ubah baris berikut dibawah ini dan simpan

NB : gunakan fitur search ( CTRL + W ) untuk memudahkan mencari baris

```
Post_max_size = 100M  
Memory_limit = 256M  
Max_execution_time = 360  
Upload_max_filesize = 100M  
Date.timezone = Asia/Jakarta
```

simpan perubahannya dengan cara tekan CTRL + X, lalu tekan Y lalu enter.

## 8. Buat Database Wordpress

Pada titik ini, semua paket dan server wordpress yang diperlukan telah diinstal. Server baru sekarang siap untuk meng-host Wordpress. Di server baru, buat database Wordpress kosong. Wordpress akan menggunakan database kosong ini untuk menyimpan kontennya.

Sebelum itu restart dulu mysql dengan menjalankan perintah dibawah ini

```
Service mysql restart
```

Jalankan perintah di bawah ini untuk masuk ke server database.

```
Mysql -u root -p
```

Kemudian buat database kosong bernama WP\_database anda bias menggunakan nama database yang sama dari server lama.

```
Create database wpdb;
```

Buat pengguna basis data bernama wp\_user dengan kata sandi baru. Anda dapat menggunakan nama pengguna dan kata sandi yang sama dari server lama.

```
Create user wpuser@localhost identified by 'passwordmu';
```

Kemudian beri pengguna akses penuh ke database.

```
Grant all on wpdb.* to wpuser@localhost identified by 'passwordmu';
```

Simpan perubahan anda dan keluar.

```
Flush privileges;
```

```
Exit;
```

## 9. Instal Wordpress terbaru

Selanjutnya, kunjungi situs Wordpress dan unduh yang terbaru atau jalankan perintah di bawah untuk melakukannya

```
Cd /var/www/html && wget https://wordpress.org/latest.tar.gz
```

```
Tar -xvzf latest.tar.gz
```

Kemudian jalankan perintah di bawah ini untuk mengatur izin yang benar untuk direktori root Wordpress

```
Chown -R www-data:www-data /var/www/html/wordpress/
```

```
Chmod -R 755 /var/www/html/wordpress/
```

## 10. Aktifkan situs Wordpress

Setelah semua step sudah selesai restart Nginx, php-fpm dan mysql dengan menjalankan perintah di bawah ini

```
Service nginx restart
```

```
Service php7.4-fpm restart
Service mysql restart
```

Setelah memulai ulang Nginx, buka browser anda dan akses menggunakan IP server atau nama host. Jika semuanya sudah diatur dengan benar, anda akan melihat panduan konfigurasi default wordpress.

```
Localhost/wordpress
IpAddress/wordpress
```

Ikuti petunjuk di layar sampai anda berhasil mengkonfigurasi Wordpress. Setelah selesai, masuk ke dasbor admin akan konfigurasi pengaturan Wordpress.

Selamat Anda baru saja menginstall Wordpress

## 11. Konfigurasi SFTP Remote Path

SFTP atau secure file transfer protocol merupakan salah satu protocol internet yang berfungsi untuk melakukan proses upload dan download file website dengan lebih aman. Mengapa lebih aman ? karena SFTP merupakan kombinasi dari FTP dan SCP. Hal ini akan mempermudah kita untuk mengesdit file sourcecode wordpress kita.

Jalankan perintah berikut ini untuk menginstall modul yang di butuhkan :

```
Apt update
Apt install openssh-server openssh-client
```

Jalankan perintah berikut untuk mengganti password root di docker container

```
Echo root:password | chpasswd
```

Jalankan perintah berikut ini untuk membuka file konfigurasi ssh.

```
Nano /etc/ssh/sshd_config
```

Pada file sshd\_config ubah nilai empat konfigurasi diatas menjadi no.

NB: jika anda tetap ingin menggunakan "root" untuk login maka jangan disable PermitRootLogin biarkantetap "yes".

- ChallengeResponseAuthentication
- PasswordAuthentication
- UsePAM
- PermitRootLogit

Jalankan perintah di bawah ini untuk melakukan reload ssh

```
Service ssh reload  
Service ssh restart
```

Silahkan anda keluar dari ubuntu docker container.

Disini saya menggunakan system operasi linux

Buka terminal ( CTRL + SHIFT + T ) dan jalankan perintah berikut ini untuk mencoba sshnya

```
Ssh root@172.17.0.2 -p 22
```

Silahkan masukkan password jika disuruh menginputkan password, Jika sudah berhasil maka anda akan masuk ke ubuntu yang ada di docker container kalian.

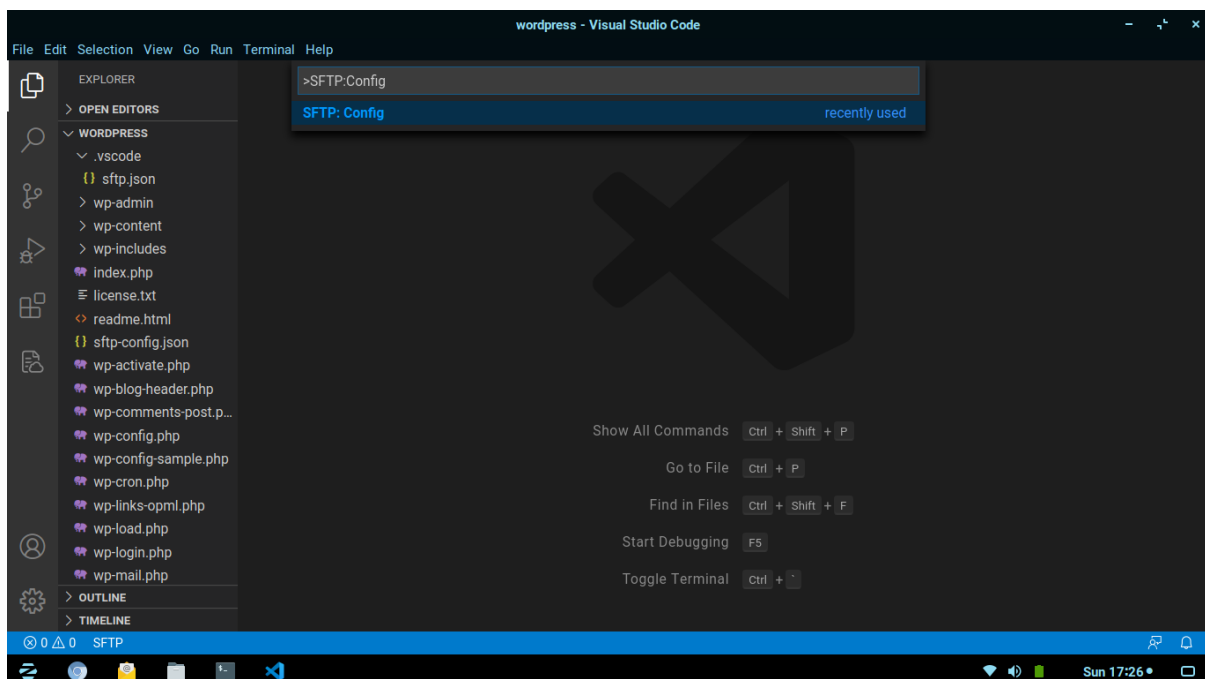
Sebelumnya copy kan dulu file wordpress yang ada di dalam docker container kalian dengan menjalankan perintah berikut .

```
Docker cp <namacontainer>:/var/www/html/wordpress /<folder kalian>
```

Silahkan install visual studio code dan jalankan di system operasi kalian. Lalu Buka menu extension dan download sftp.

Lalu buka folder wordpress kalian di visual studio code.

Lalu tekan ( CTRL + SHIFT + P ) untuk membuka command pallet dan ketikan sftp dan pilih menu sftp:config.



Lalu akan muncul di folder project kita yaitu .vscode berupa file json. File ini yang akan kita konfigurasi sesuai dengan server yang dituju kemudian. Isi konfigurasi sebagai berikut :

```
{
  "name": "localhost",
  "host": "172.17.0.2", //ip server sftp
  "protocol": "sftp",
  "port": 22,
  "username": "user",
  "password": "password",
  "remotePath": "/var/www/html/wordpress",
  "uploadOnSave": true
}
```

Kemudian simpan

Sekarang kalian tinggal mengedit source code yang ada di folder kalian maka source code tersebut akan otomatis terupload ke file yang ada di docker container kalian.

## 12. Membuat Dockerfile

Dockerfile memudahkan kita untuk menjalankan Modul Modul yang dibutuhkan oleh Wordpress seperti Nginx ataupun Mysql tanpa harus mengeksekusi ataupun mengakses Ubuntu terlebih dahulu

Buat file dengan nama Dockerfile di system operasi kalian ( windows = Notepad, Linux = Nano ) dan isi dengan perintah berikut:

*NB: jika extension file berupa \*.txt maka hapus extension \*.txt nya*

```
FROM <namaimages:tag>
EXPOSE 22 80 443
COPY supervisord.conf /etc/supervisor/conf.d/supervisord.conf
CMD ["/usr/bin/supervisord"]
```

Buat file dengan nama supervisord.conf di system operasi kalian dan isi dengan perintah berikut:

```
[supervisord]
Nodaemon=true
```

```

[program:nginx]
Command=nginx -c /etc/nginx/nginx.conf -g 'daemon off;'
Stdout_logfile=/dev/stdout
Stdout_logfile_maxbytes=0
Stderr_logfile=/dev/stderr
Stderr_logfile_maxbytes=0

[program=php-fpm]
Command=php-fpm7.4 -R -F -c /etc/php/7.4/fpm/php-fpm.conf
Stdout_logfile=/dev/stdout
Stdout_logfile_maxbites=0
Stderr_logfile=/dev/stderr
Stderr_logfile_maxbites=0

[program:mysql]
Command=/usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql
Stdout_logfile=/dev/stdout
Stdout_logfile_maxbites=0
Stdout_events_enabled=true
Stderr_logfile=/dev/stderr
Stderr_logfile_maxbites=0
Stderr_events_enabled=true
Autorestart=true

[program: sshd]
Command=/usr/sbin/sshd -D

```

Simpan kedua file di folder yang sama. Kemudian jalankan perintah berikut:

```
Docker build -t <namaimagebaru:tag> .
```

Maka image baru akan terbentuk dan untuk menjalankan image baru tersebut, bisa menggunakan perintah dibawah ini :



```
Docker run -tdi -name <namacontainer> -p 80:80 <namaimagebaru:tag>
```

Cek apakah container sudah berjalan dengan menggunakan perintah

```
Docker ps
```

Jika sudah berjalan, buka browser dan akses wordpress anda dengan memasukkan ipadress/wordpress atau localhost/wordpress, dan anda tidak perlu lagi mengakses docker container anda untuk menjalankan modul modul yang dibutuhkan.

### 13. Konfigurasi docker container autostart

Konfigurasi ini berguna ketika kita booting ke pc. Kita tidak perlu memulai kembali docker container, kita bisa langsung mengakses localhost ataupun vscode nya. Disini saya menggunakan system operasi linux, jadi saya akan menggunakan systemctl

Jalankan perintah berikut untuk membuat file systemctl

```
Sudo nano /etc/systemd/system/docker-wpserver.service
```

Kemudian isi dengan perintah berikut:

```
[Unit]
Description=wpserver container
Requires=docker.service
After=docker.service

[service]
Restart=always
ExecStart=/usr/bin/docker start -a wpserver
ExecStop=/usr/bin/docker stop -t wpserver

[install]
WantedBy=multi-user.target
```

Kemudia simpan

Jalankan perintah dibawah ini untuk memuat ulang systemctl

```
Sudo systemctl daemon-reload
```

Untuk mengaktifkan service, jalankan perintah di bawah ini:

```
Sudo systemctl start docker-wpserver.service  
Sudo systemctl enable docker-wpserver.service
```

Untuk menonaktifkan service, jalankan perintah di bawah ini:

```
Sudo systemctl stop docker-wpserver.service  
Sudo systemctl disable docker-wpserver.service
```

jika sudah silahkan reboot system operasi kalian dan docker container akan berjalan dengan dengan otomatis

## **BAB III**

### **PENUTUP**

#### ***3.1. Kesimpulan***

Setelah kita mengimplementasikan *docker* dengan menggunakan Linux dalam membuat aplikasi webcommerce, kita dapat mengetahui bagaimana cara instalasi dan cara penggunaannya.

#### ***3.2. Saran***

Bagi pembaca makalah ini, semoga makalah ini dapat menjadi bermanfaat bagi semua. Kepada generasi muda untuk lebih bijak dalam penggunaan perkembangan teknologi saat ini.