

**PRAKTIKUM TEKNOLOGI CLOUD COMPUTING  
LAPORAN PROYEK AKHIR**

**SISTEM MANAJEMEN PASSWORD MENGGUNAKAN UBUNTU LAMPP DAN  
PROSES PEMBUATAN DOCKERFILENYA**



**DISUSUN OLEH:**

**NAMA ANGGOTA : ALIVI MILOVA 123170062  
REFANDA SETYAGUNA S 123170093  
KELAS : D  
ASISTEN PRAKTIKUM : JALUANDA PARAMA, S.KOM.  
WAHYU AJI NUGROHO, S.KOM.**

**PROGRAM STUDI INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS TEKNIK INDUSTRI  
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"  
YOGYAKARTA  
2020**

## HALAMAN PENGESAHAN

### SISTEM MANAJEMEN PASSWORD MENGGUNAKAN UBUNTU LAMPP DAN PROSES PEMBUATAN DOCKERFILENYA

Disusun oleh :

Alivi Milova

123170062

Refanda Setyaguna Sutrisno

123170093

Telah diperiksa dan disetujui oleh Asisten Praktikum Teknologi Cloud Computing  
pada tanggal : .....

Menyetujui,

Asisten Praktikum

Asisten Praktikum

Jaluanda Parama, S.Kom.

Wahyu Aji Nugroho, S.Kom.

Mengetahui,

Ka. Lab. Sistem Digital

Mangaras Yanu Florestiyanto, S.T., M.Eng.

NIK. 2 8201 13 0425 1

## KATA PENGANTAR

Assalamualaikum warohmatullohi wabarokatuh

Segala puji bagi Allah SWT yang telah memberikan kami rahmat, pertolongan, kesehatan dan ilmu sehingga kami dapat dengan lancar mengerjakan laporan proyek akhir untuk mata kuliah Praktikum Teknologi Cloud Computing Tahun Ajaran 2020.

Laporan ini dibuat untuk menerangkan perihal proses penerapan cloud dengan judul “Sistem Manajemen Password Menggunakan Ubuntu LAMPP dan Pembuatan Dockerfilenya” dan juga guna memenuhi tugas akhir dari mata kuliah Praktikum Teknologi Cloud Computing yang dibimbing oleh saudara Jaluanda Parama, S.Kom dan juga saudara Wahyu Aji Nugroho, S.Kom.

Pada laporan ini terdapat penjelasan mengenai kenapa perlunya sebuah program itu dibuat dan bagaimana proses pengerjaan program tersebut. Sehingga laporan ini juga bisa menjadi tambahan ilmu pengetahuan bagi mahasiswa maupun kalangan umum mengenai program yang dibuat .

Kami menyadari bahwa laporan ini masih jauh dari kata sempurna dan juga masih terdapat banyak kekurangan, oleh karena itu kami mengharapakan kepada pembaca untuk kritik dan saran yang berguna untuk meningkatkan program yang sedang dikembangkan sehingga bisa lebih berkualitas.

Tidak lupa kami mengucapkan terimakasih kepada semua pihak yang membantu dalam pengerjaan laporan ini terutama pembimbing praktikum saudara Jaluanda Parama, S.Kom dan juga saudara Wahyu Aji Nugroho, S.Kom.

Demikian kata pengantar ini kami buat, jika terdapat kesalahan kami mohon maaf.

Wassalamualaikum warohmatullohi wabarokatuh

Yogyakarta, 02 April 2020

Penyusun

## DAFTAR ISI

HALAMAN PENGESAHAN.....	ii
KATA PENGANTAR.....	iii
DAFTAR ISI.....	iv
<b>BAB I PENDAHULUAN.....</b>	<b>1</b>
1.1. Latar Belakang Masalah.....	1
1.2. Tujuan Proyek Akhir.....	2
1.3. Manfaat Proyek Akhir.....	3
1.4. Tahap Penyelesaian Proyek Akhir.....	3
<b>BAB II ISI DAN PEMBAHASAN.....</b>	<b>4</b>
2.1. Komponen yang Digunakan.....	4
2.2. Rancangan Arsitektur <i>Cloud Computing</i> .....	5
2.3. Parameter dan Konfigurasi.....	7
2.4. Tahap Implementasi.....	10
2.5. Hasil Implementasi.....	16
2.6. Pengujian Singkat.....	20
<b>BAB III JADWAL Pengerjaan dan Pembagian Tugas.....</b>	<b>21</b>
3.1. Agenda Pengerjaan.....	22
3.2. Keterangan Pembagian Tugas.....	22
<b>BAB IV KESIMPULAN DAN SARAN.....</b>	<b>23</b>
4.1. Kesimpulan.....	23
4.2. Saran.....	23
<b>DAFTAR PUSTAKA.....</b>	<b>24</b>
<b>LAMPIRAN.....</b>	<b>25</b>

# BAB I

## PENDAHULUAN

### 1.1. Latar Belakang Proyek Akhir

Saat ini, *cloud computing* merupakan salah satu teknologi yang banyak sekali digunakan. Mulai dari skala personal hingga skala *enterprise* menggunakan teknologi ini, terlebih dalam menunjang kebutuhan alur bisnis yang dimiliki.

Di dalam bukunya, (Erl, Puttini, & Mahmood, 2013) menjelaskan bahwa pada awalnya konsep dari teknologi *cloud* diusulkan oleh John McCarthy pada tahun 1961. Komputer berbasis internet mulai dimanfaatkan pada pertengahan 1990-an, dengan implementasi berupa *search engine*, layanan email dan beberapa platform media sosial. Pada akhir 1990-an Salesforce memelopori layanan *remote* ke beberapa perusahaan. Kemudian pada tahun 2002 Amazon meluncurkan platform Amazon Web Service (AWS) yang menyediakan penyimpanan, sumber daya komputasi, dan fungsionalitas bisnis.

*Cloud computing* adalah bentuk khusus dari komputasi terdistribusi yang memperkenalkan model pemanfaatan untuk penyediaan sumber daya yang terukur dari jarak jauh (Erl, Puttini, & Mahmood, 2013). Secara lebih detail, *cloud computing* merupakan sebuah konsep pemahaman dalam rangka pembuatan kerangka kerja komputasi secara online lokal (LAN) maupun global (internet) dimana terdapat beragam aplikasi maupun data dan media penyimpanan yang dapat diakses dan digunakan secara berbagi (*shared device*) dan bersamaan (*simultaneous access*) oleh para pengguna yang beragam, mulai dari perseorangan sampai kepada kelas pengguna korporasi atau perusahaan (Datacomm, 2016).

Pada dasarnya *cloud computing* memiliki beberapa layanan model, yaitu: *Software as a Service* (SaaS), *Platform as a Service* (PaaS), *Infrastructure as a Service* (IaaS). *Software as a service* merupakan layanan dimana suatu aplikasi didistribusikan kepada pelanggan dan diakses melalui internet. Dengan model ini pelanggan tidak perlu memikirkan infrastruktur yang mendasari suatu aplikasi berjalan, hanya saja pelanggan akan memiliki batasan pada konfigurasi aplikasi yang digunakan terhadap infrastruktur yang mendasarinya. Model lain dari *cloud computing* yaitu *platform as a service*, maksud dari model ini yaitu penyedia layanan menawarkan lingkungan pengembangan suatu aplikasi, sehingga pengguna dapat mengembangkan suatu aplikasi tanpa perlu khawatir untuk membangun infrastruktur yang diperlukan. Kemudian terdapat model *infrastructure as a service* dimana layanan ini mencakup mesin secara fisik maupun virtual, jaringan,

penyimpanan, atau kombinasi dari semuanya. Dengan layanan ini pengguna dapat membangun dengan menggunakan apa saja yang dibutuhkan.

Pada era saat ini, banyak sekali layanan aplikasi yang memerlukan autentikasi untuk menggunakannya. Banyaknya layanan yang digunakan oleh pengguna memungkinkan satu orang memiliki banyak sekali akun yang berbeda-beda untuk mengakses beberapa aplikasi. Melihat fenomena tersebut, kini sudah banyak teknologi yang bertemakan *password* manajer yang berfungsi untuk menyimpan informasi autentikasi sebuah akun. Layanan *password* manajer tersebut mengimplementasikan *cloud computin* dengan menawarkan aplikasi sebagai sebuah *service* dan juga dapat diakses dan diintegrasikan dengan berbagai perangkat yang berbeda.

Memiliki banyak akun bukan berarti tidak memiliki resiko. Karena keterbatasan ingatan seseorang, tidak jarang terdapat pengguna yang lupa dengan akun yang dimiliki. Alhasil mereka harus melakukan reset password atau bahkan harus merelakan akun yang dimilikinya hilang karena tidak dapat dikembalikan.

Disinilah layanan *password* manajer berperan. Dengan menggunakan layanan tersebut pengguna tidak perlu melakukan serangkaian prosedur untuk melakukan validasi kepemilikan akun dan kemudian mengganti *password* dengan *password* yang baru. Apabila pada suatu saat pengguna lupa dengan autentikasi akun yang ia miliki, maka mereka cukup membuka aplikasi *password* manajer dan melihat apa password dari akun yang ingin ia buka. Hal tersebut tentu saja lebih praktis daripada melakukan serentetan prosesur dalam lupa password. Selain itu, penerapan *cloud computing* dalam proyek ini juga memudahkan dalam melakukan pengaksesan layanan dari *password* manajer, karena memungkinkan diakses melalui perangkat manapun.

Dari beberapa aplikasi bertemakan *password* manajer yang ada, banyak sekali yang bersifat langganan atau dengan kata lain aplikasi tersebut berbayar. Oleh karenanya kami mencoba membuat apliasi *password* manajemen sendiri dengan memanfaatkan algoritma enkripsi AES-256 untuk mengamankan data yang disimpan didalam *database*. Aplikasi tersebut akan mengimplementasikan layanan *hosting local* dengan menggunakan Ubuntu dan LAMPP sebagai *platform service*-nya.

## 1.2 Tujuan Proyek Akhir

Berdasarkan latar belakang proyek akhir yang telah dijelaskan sebelumnya, mengenai tujuan dari pembuatan proyek akhir ini adalah sebagai berikut:

1. Mengimplementasikan arsitektur *cloud computing* untuk Sistem Manajemen Password yang telah dibuat sebelumnya dengan menggunakan Ubuntu 18.04 dan LAMPP (Apache 8, PHP 7.1, MySQL 5.2).

2. Membuat Dockerfile yang berada di Ubuntu Server (LAMPP) untuk mempermudah dalam instalasi sistem pada *host* yang berbeda.

### 1.3 Manfaat Proyek Akhir

Manfaat yang dapat diperoleh dari pembuatan proyek akhir ini adalah sebagai berikut:

1. Sistem yang telah dirancang dapat digunakan sewaktu-waktu tanpa khawatir *downtime* dikarenakan terdapatnya *primary* dan *backup server* yang menggunakan arsitektur *recovery* pada *cloud computing*.

2. Penyedia sistem tidak perlu memperlakukan *maintenance*, dikarenakan dengan menggunakan *cloud computing*, rutinitas *maintenance* akan dilakukan sepenuhnya oleh *vendor*.

3. Batasan memori penyimpanan akun autentikasi menjadi tidak terbatas dikarenakan sistem telah sepenuhnya beralih menggunakan *cloud computing*.

4. Pemasangan sistem ke *server* lain akan lebih mudah karena menggunakan Docker yang membungkus sistem beserta *dependency* yang diperlukan dalam sistem.

### 1.4 Tahap Penyelesaian Proyek Akhir

Tahapan secara singkat untuk penyelesaian proyek akhir ini adalah sebagai berikut:

1. Menganalisis kebutuhan dari sistem manajemen password untuk ditransformasikan ke dalam arsitektur *cloud computing* menggunakan basis SaaS.
2. Menentukan konfigurasi yang tepat untuk pengaturan Dockerfile sehingga dapat digunakan sesuai *requirement* yang berupa:
  - a. Dapat diinstal di mana saja tanpa mempengaruhi *environment* sistem.
  - b. Dapat melakukan migrasi database dengan mudah.
  - c. Menggunakan Ubuntu dengan versi 18.04.3 LTS.
3. Menguji konfigurasi Docker pada *host* untuk mengetahui apakah konfigurasi dapat berjalan pada *host* lain atau hanya sekedar berjalan pada *host* dimana konfigurasi Docker dibuat.

## BAB II

### ISI DAN PEMBAHASAN

#### 2.1 Komponen yang Digunakan

Untuk membangun “Sistem Manajemen Password dan Proses Pembuatan Dockerfilenya” yang berbasis konsep *cloud computing*, maka diperlukan analisis berbagai komponen. Berikut akan dijelaskan terlebih dahulu dalam bentuk poin-poin singkat:

1. Sistem yang telah dibangun menggunakan bahasa pemrograman PHP dengan versi 7.3 dan bahasa HTML dengan versi minimal 4.0.
2. Selain itu juga diperlukan penyimpanan basis data dengan arsitektur penyimpanan MySQL versi 5.2 sehingga dapat digunakan untuk menyimpan berbagai data kendaraan bermotor yang dibutuhkan oleh sistem tersebut.
3. Dalam pembuatan Dockerfile diperlukan Docker Engine dengan versi 19.03 dan Docker Compose dengan versi 1.17.
4. Untuk target pengguna dengan konsep *cloud computing*, maka penggunaanya dapat mengakses sistem dari mana saja, sehingga arsitektur dari *cloud computing* bersifat *public*.

Berdasarkan penjelasan poin-poin tersebut, untuk komponen utama penyusun *cloud computing* yang dibutuhkan dapat disimpulkan dalam bentuk tabel sebagai berikut:

**Tabel 2.1** Spesifikasi VM *cloud computing* untuk proyek pertama

No.	Nama Parameter	Nilai	Keterangan
1.	Merek Server	Virtual Machine dengan VMWare Workstation	Tidak menggunakan <i>hardware</i> fisik secara langsung, melainkan menggunakan aplikasi <i>virtual machine</i> .
2.	Prosesor	2 core @2.4Ghz	Prosesor dari <i>hypervisor</i> yang dialokasikan ke <i>guest</i> .
3.	Konfigurasi Jaringan <i>Guest OS</i>	Mode Bridge	Mode adapter jaringan VM <i>guest</i> yang digunakan.
		IP: 192.168.110.2/24	Alamat IP dan <i>network</i> yang digunakan oleh <i>guest OS</i> .
		DNS: 192.168.110.1	Alamat IP untuk DNS <i>guest OS</i> .
		GW: 192.168.110.1	Alamat untuk <i>gateway</i> atau gerbang menuju akses jaringan luar.
4.	Versi Ubuntu	Ubuntu 18.04.3 LTS	ISO Ubuntu yang digunakan untuk <i>guest OS</i> .
5.	RAM	2GB	Alokasi RAM untuk <i>guest OS</i>
6.	Disk info	20GB	Alokasi <i>storage</i> untuk <i>guess OS</i>



Selain spesifikasi mengenai VM *cloud computing* tersebut, untuk spesifikasi yang digunakan dalam Ubuntu OS yang telah dibuat dalam VM tersebut adalah sebagai berikut:

**Tabel 2.2** Spesifikasi Ubuntu OS untuk proyek pertama

No.	Nama Parameter	Nilai	Keterangan
1.	LAMPP	Apache 2.4	Preproesor bahasa pemrograman HTML, termasuk CSS dan JS.
		PHP 7.3	Dukungan bahasa pemrograman yang digunakan oleh sistem manajemen <i>password</i> .
		MySQL	Dukungan penyimpanan yang digunakan oleh sistem manajemen <i>password</i> .
2.	PuTTY	PuTTY 0.73	Dukungan file sharing yang digunakan untuk memindahkan proyek manajemen <i>password</i> .
3.	Git	Git 2.17.1	Dukungan <i>control version</i> yang digunakan untuk <i>clone</i> proyek dari GitHub <i>repository</i>
4.	SSH	OpenSSH	Dukungan untuk melakukan <i>remote server</i> .
5.	Docker Engine	Docker Server 19.03	<i>Docker engine server</i> yang digunakan dalam menjalankan container.
		Docker Client 19.03	<i>Docker engine client</i> yang digunakan untuk berkomunikasi dengan <i>docker server</i> .
6.	Docker Compose	Docker Compose 1.17	Dukungan alat untuk mendefinisikan dan menjalankan <i>multi-container</i> .

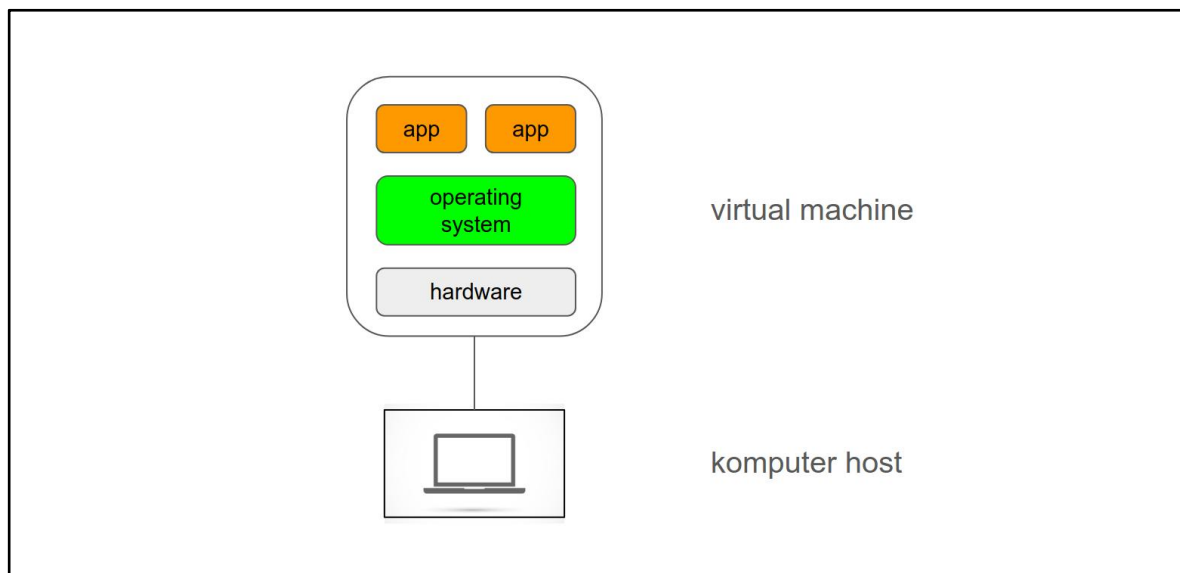
## 2.2 Rancangan Arsitektur Cloud Computing

Arsitektur dalam *cloud computing* secara umum terbagi menjadi 3 bagian, yaitu infrastruktur, platform dan aplikasi (Ernawati & Zulfiaji, 2014). Pada proyek akhir ini digunakan bentuk rancangan arsitektur SaaS untuk sistem manajemen *password*, di bawah layer SaaS terdapat layer PaaS dimana sistem operasi Ubuntu berjalan. Kemudian pada layer paling bawah atau IaaS terdapat *hardware* laptop yang melakukan virtualisasi pada *hardware* yang diperlukan. Ilustrasi mengenai rancangan arsitektur tersebut dapat dilihat pada **Gambar 2.1** berikut ini:

Model Service	Resource Manager	Layer	Example
Software as a service (SaaS)	Web application	application	Manajemen Password
Platform as a service (PaaS)	Software framework	platforms	PHP, Apache, MySQL, Docker
Infrastructure as a service (IaaS)	Computation (VM),	infrastructure	Ubuntu Server
	CPU, Memory, Disk, Processor	hardware	Komputer, Laptop

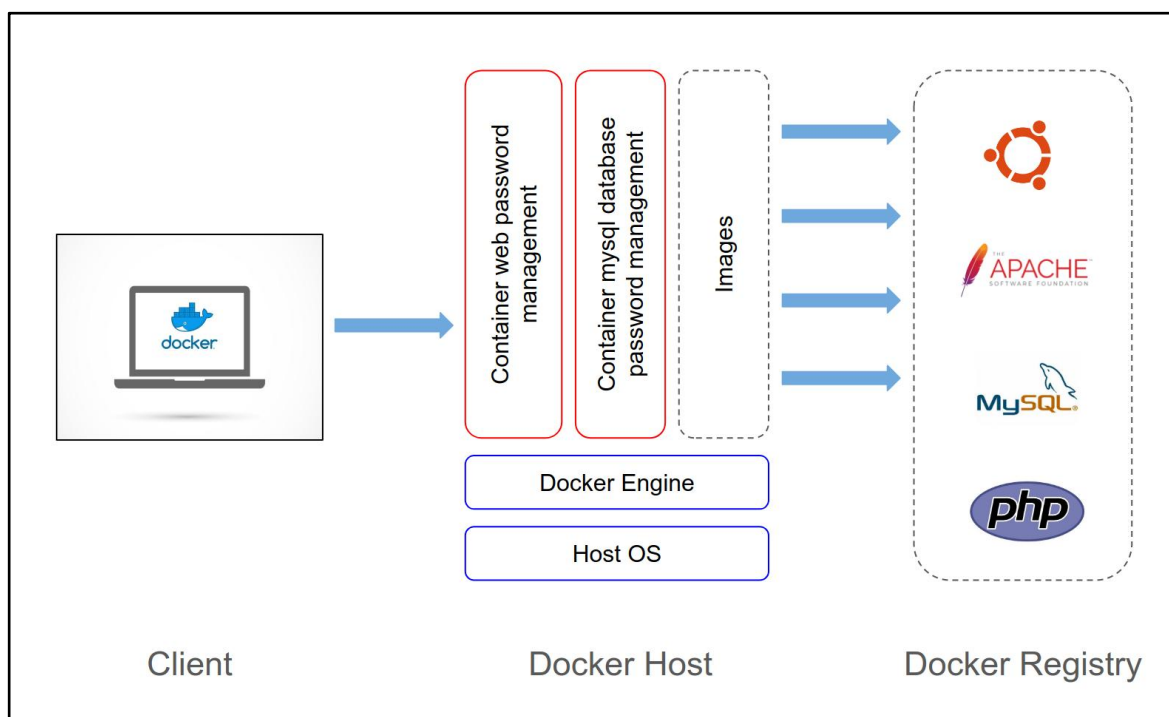
**Gambar 2.1** Penjelasan layer arsitektur terhadap komponen penyusunnya

Dalam penyelesaian proyek akhir ini, *virtual machine* yang berjalan pada komputer atau laptop digunakan sebagai layanan *hosting local*. Pada dasarnya penggunaan *virtual machine* secara *local* ini digunakan sebagai simulasi yang dapat merepresentasikan kondisi aslinya. Ilustrasi mengenai rancangan arsitektur *virtual machine* dapat dilihat pada **Gambar 2.2** berikut ini:



**Gambar 2.2** Penjelasan arsitektur *virtual machine*

Selain melakukan implementasi *hosting local*, pada proyek akhir ini juga menerapkan teknologi Docker yang mana membungkus aplikasi beserta *dependency* yang diperlukan dalam proyek tersebut. Ilustrasi mengenai rancangan arsitektur Docker pada proyek ini dapat dilihat pada **Gambar 2.3** berikut ini:



**Gambar 2.3** Penjelasan arsitektur Docker terhadap komponen penyusunnya

## 2.3 Parameter dan Konfigurasi

Dalam pembuatan proyek akhir ini, pertama yang perlu dilakukan adalah memasang *web server* Apache. Parameter yang digunakan untuk instalasi Apache dapat dilihat pada penjelasan **Modul 2.1** berikut ini:

```
$ sudo apt install apache2
```

Keterangan:

- `sudo` : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- `apt` : merupakan package manager pada Ubuntu
- `install` : parameter tambahan pada apt untuk mengeksekusi perintah instalasi paket aplikasi
- `apache2` : nama paket aplikasi untuk Apache

### Modul 2.1 Parameter instalasi Apache

Selain *web server*, dalam segi penyimpanan data diperlukan suatu *database*. Dalam proyek ini *database* yang digunakan adalah MySQL. Parameter yang digunakan untuk instalasi MySQL dapat dilihat pada penjelasan **Modul 2.2** berikut ini:

```
$ sudo apt install mysql-server
$ sudo mysql_secure_installation
```

Keterangan:

- `mysql-server` : nama paket aplikasi untuk MySQL

- `mysql_secure_installation` : command untuk menjalankan proses instalasi MySQL

### **Modul 2.2** Parameter instalasi MySQL

Agar proyek Manajemen Password ini dapat berjalan, maka diperlukan suatu bahasa pemrograman agar aplikasi *web* ini dapat berjalan. Bahasa PHP dipilih dalam pengembangan proyek Manajemen Password ini. Parameter yang digunakan untuk instalasi PHP dapat dilihat pada penjelasan **Modul 2.3** berikut ini

```
$ sudo apt install php libapache2-mod-php php-mysql
```

Keterangan:

- `php, libapache2-mod-php, php-mysql` : nama paket aplikasi

### **Modul 2.3** Parameter instalasi bahasa pemrograman PHP

Untuk mempermudah dalam melakukan manajemen *database*, maka diperlukan instalasi PHP MyAdmin. Parameter yang digunakan untuk instalasi PHP MyAdmin dapat dilihat pada penjelasan **Modul 2.4** berikut ini:

```
$ sudo apt install phpmyadmin php-mbstring php-gettext
```

Keterangan:

- `phpmyadmin, php-mbstring, php-gettext` : nama paket aplikasi

### **Modul 2.4** Parameter instalasi PHP MyAdmin

Sebelum melakukan instalasi Docker Engine pada mesin untuk pertama kalinya, maka diperlukan *set up repository* Docker, setelah itu instalasi Docker dapat dilakukan melalui *repository* tersebut. Parameter yang digunakan untuk melakukan *set up repository* dapat dilihat pada penjelasan **Modul 2.5** berikut ini:

```
$ sudo apt update
$ sudo apt install \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg-agent \
    software-properties-common
```

Keterangan:

- `update` : parameter tambahan pada apt yang berfungsi untuk memperbarui dependency pada repository system
- `apt-transport-https, ca-certificates, curl, gnupg-agent, software-properties-common` : nama paket aplikasi

### **Modul 2.5** Parameter *set up repository* Docker

Setelah *package index* terbaru, maka dilanjut dengan menambahkan *key* GPG Docker. Parameter yang digunakan dalam penambahan *key* GPG dapat dilihat pada penjelasan **Modul 2.6** berikut ini:

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

Keterangan:

- `curl` : command line tool untuk mentransfer data menggunakan protokol jaringan.
- `apt-key` : digunakan untuk manajemen daftar *key* yang digunakan oleh `apt` untuk melakukan autentikasi package.
- `add -` : parameter tambahan pada *key* package manager untuk menambahkan *key* ke daftar trusted *key*.

### Modul 2.6 Parameter penambahan *key* GPG Docker

Tahap terakhir dalam *set up repository* Docker adalah dengan mengatur *repository* mana yang ingin digunakan. Docker memiliki tiga jenis versi *repository* yaitu: *stable*, *nightly*, dan *test*. Dalam implementasi proyek ini, *repository* yang digunakan yaitu versi *stable*. Parameter yang digunakan untuk memilih *repository* dapat dilihat pada penjelasan **Modul 2.7** berikut ini:

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

Keterangan:

- `add-apt-repository` : menambahkan PPA ke daftar sumber, sehingga Ubuntu tahu untuk mencari pembaruan dari PPA tersebut berasal dari sumber yang resmi.
- `lsb_release` : command untuk menampilkan LSB (Linux Standard Base) dan Distribution Information
- `stable` : versi *repository* yang akan diinstall.

### Modul 2.7 Parameter pemilihan versi *repository* Docker

Apabila *set up repository* sudah selesai, maka dilanjutkan dengan instalasi Docker Engine. Parameter instalasi Docker Engine dapat dilihat pada penjelasan **Modul 2.8** berikut ini:

```
$ sudo apt update
$ sudo apt install docker-ce docker-ce-cli containerd.io
```

Keterangan:

- `docker-ce` : paket aplikasi untuk docker engine community
- `docker-ce-cli` : paket aplikasi untuk docker engine command line
- `containerd.io` : paket aplikasi untuk container

### Modul 2.8 Parameter instalasi Docker Engine

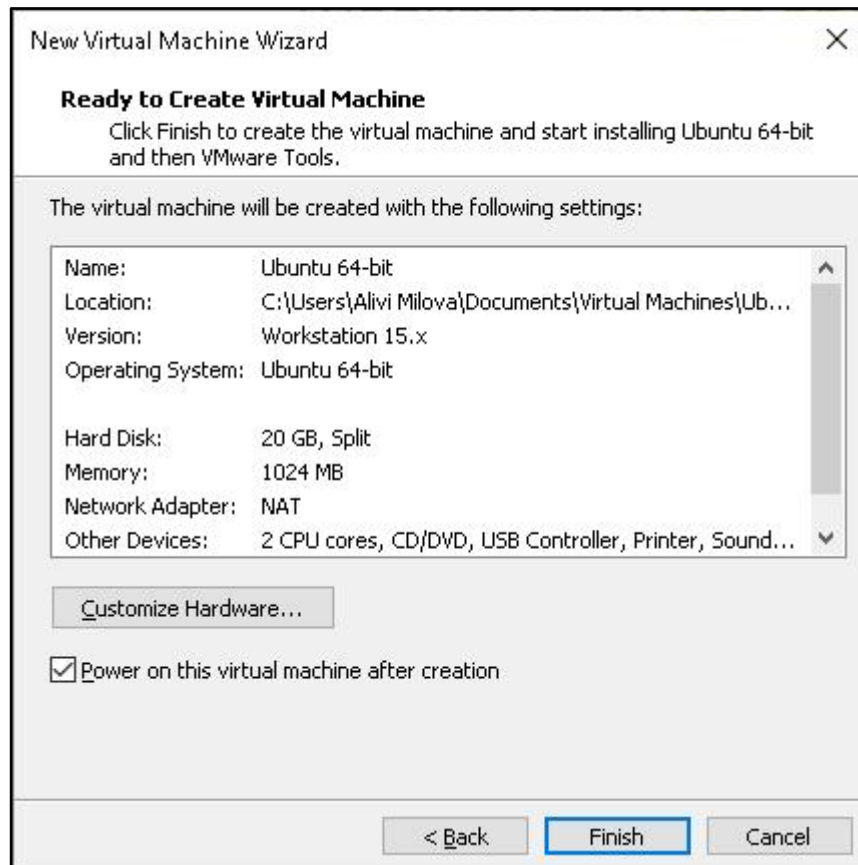
Selanjutnya agar dapat menjalankan *multi-container* secara bersamaan, maka diperlukan instalasi Docker Compose. Parameter instalasi Docker Compose dapat dilihat pada penjelasan **Modul 2.9** berikut ini:

```
$ sudo curl -L \
  "https://github.com/docker/compose/releases/download/1.25.5/docker-
  compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

### Modul 2.9 Parameter instalasi Docker Compose

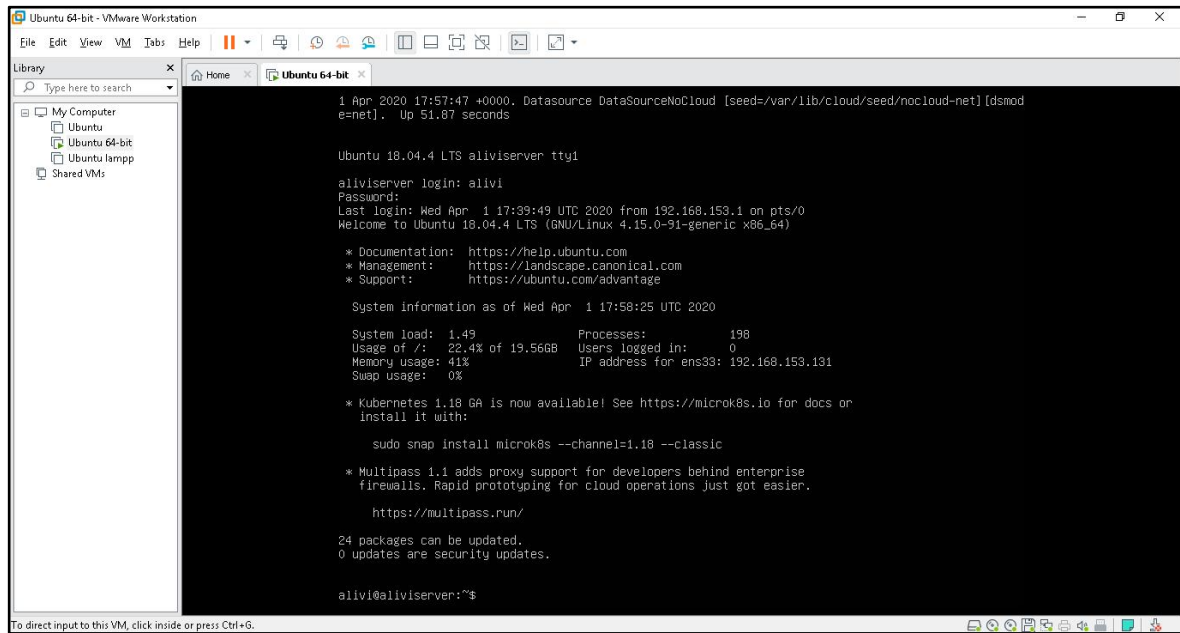
## 2.4 Tahap Implementasi

Terdapat beberapa tahapan yang perlu dilakukan dalam penerapan *hosting* dengan Ubuntu LAMPP pada proyek Manajemen Password. Pertama yaitu membuat *virtual machine* terlebih dahulu. Dalam pembuatan *virtual machine* dengan VMWare Workstation dilakukan opsi konfigurasi seperti pada **Gambar 2.4** berikut ini:



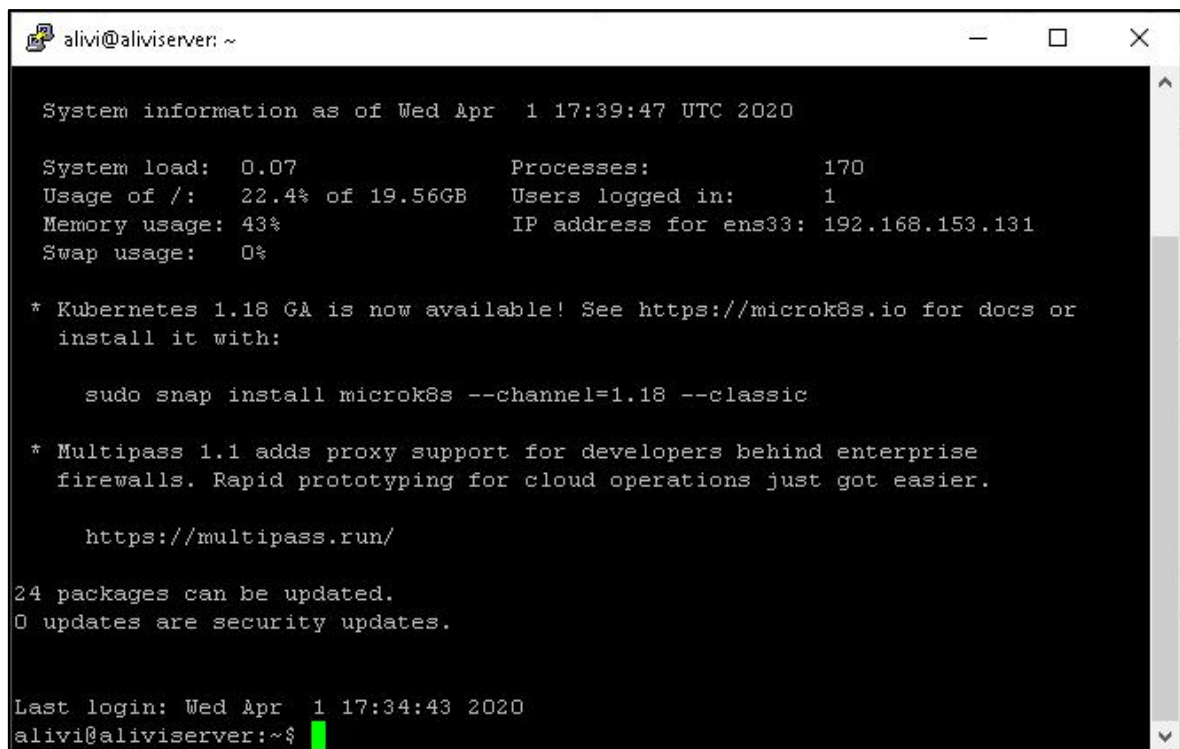
**Gambar 2.4** Tampilan opsi pemilihan *mode wizard* pembuatan VM

Jika sudah maka tinggal menunggu proses instalasi selesai. Apabila proses instalasi sudah selesai maka dilanjutkan dengan *login* pada *server* yang telah dibuat. Tampilan halaman login dapat dilihat pada **Gambar 2.5** berikut ini:



**Gambar 2.5** Tampilan *login* pada *server* Ubuntu

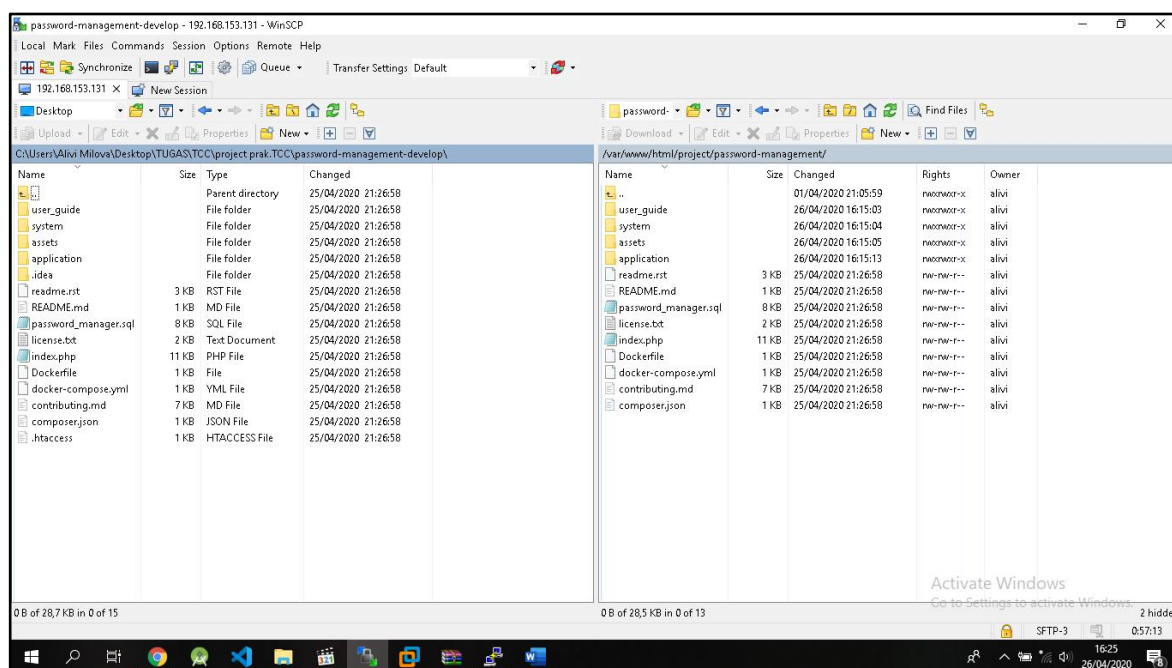
Tahap selanjutnya yaitu melakukan *remote server* dengan menggunakan PuTTY. Ketika pertama kali membuka PuTTY dibutuhkan beberapa konfigurasi, isikan IP pada PuTTY dengan IP pada *server* Ubuntu. Apabila sudah, tampilan PuTTY dapat dilihat pada **Gambar 2.6** berikut ini:



**Gambar 2.6** Tampilan awal PuTTY setelah masuk ke *server*

Tahapan selanjutnya yaitu melakukan instalasi pada Apache, MySQL, PHP, dan PHP MyAdmin untuk mempermudah dalam manajemen *database*. Adapun proses instalasi dari Apache, MySQL, PHP, dan PHP MyAdmin dapat dilihat pada **Modul 2.1**, **Modul 2.2**, **Modul 2.3** dan **Modul 2.4**.

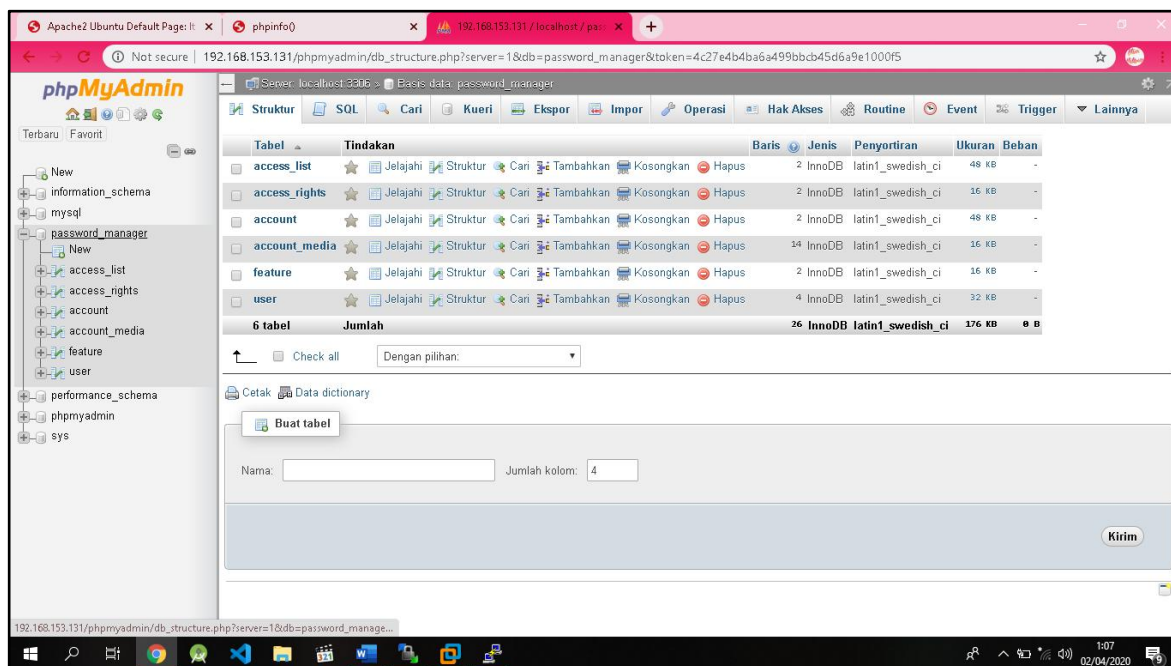
Apabila komponen LAMPP sudah terinstal, selanjutnya memindahkan file proyek kedalam *server*. Dalam pemindahan ini kami memanfaatkan aplikasi WinSCP. Adapun file proyek akan dipindah ke direktori `/var/www/html/project/` pada *server*. **Gambar 2.7** berikut ini menampilkan hasil dari pemindahan proyek dari komputer lokal ke *server* Ubuntu.



**Gambar 2.7** Hasil pemindahan file proyek ke *server* Ubuntu

Apabila pemindahan file sudah berhasil, selanjutnya yaitu melakukan *import database* yang dapat dilakukan melalui PHP MyAdmin. Cara untuk *import* yaitu dengan memilih terlebih dahulu database mana yang ingin dijadikan tempat menyimpan kemudian masuk ke fitur *import* dan memasukkan file `password_manager.sql` yang berada dalam direktori *root* proyek. **Gambar 2.8** berikut ini menampilkan hasil dari *import database*.





**Gambar 2.8** Tampilan *database* setelah di-import

Apabila proses *import* database sudah berhasil, maka proyek Manajemen Password dapat digunakan dengan mengakses IP dari VM dan masuk ke direktori `project/password-management`.

Selanjutnya dalam pembuatan Dockerfile dilakukan dengan konfigurasi VM yang sama akan tetapi berada pada *host* yang berbeda. Pertama yang harus dilakukan adalah melakukan proses instalasi Docker seperti yang tertera pada **Modul 2.5** sampai dengan **Modul 2.9**.

Dalam implementasi pembuatan Dockerfile dilakukan secara *remote* dengan memanfaatkan terminal dan SSH. **Gambar 2.9** berikut ini menampilkan terminal dengan kondisi sudah terhubung dengan *server*.

```
refanda@vo:~$ ssh ubuntu@172.16.70.132
ubuntu@172.16.70.132's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-96-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Thu Apr 30 01:48:44 UTC 2020

System load:  0.02          Processes:            166
Usage of /:   33.5% of 19.56GB Users logged in:       1
Memory usage: 28%          IP address for ens33: 172.16.70.132
Swap usage:   0%           IP address for docker0: 172.17.0.1

 * Ubuntu 20.04 LTS is out, raising the bar on performance, security,
and optimisation for Intel, AMD, Nvidia, ARM64 and Z15 as well as
AWS, Azure and Google Cloud.

https://ubuntu.com/blog/ubuntu-20-04-lts-arrives

0 packages can be updated.
0 updates are security updates.

*** System restart required ***
Last login: Thu Apr 30 01:47:30 2020 from 172.16.70.1
ubuntu@ubuntu-server:~$
```

**Gambar 2.9** Tampilan awal terminal setelah terhubung dengan *server*

Tahap selanjutnya yaitu memindahkan proyek kedalam *server*. Dikarenakan proyek sudah ter-*host* dalam GitHub *repository* maka dalam pengujian ini tinggal melakukan *clone* pada *repository* tersebut. Proses *clone* dilakukan seperti pada **Gambar 2.10** berikut ini:

```
ubuntu@ubuntu-server:~$ git clone https://github.com/refandas/password-management.git
Cloning into 'password-management'...
remote: Enumerating objects: 2798, done.
remote: Counting objects: 100% (2798/2798), done.
remote: Compressing objects: 100% (1859/1859), done.
remote: Total 2798 (delta 922), reused 2771 (delta 901), pack-reused 0
Receiving objects: 100% (2798/2798), 17.74 MiB | 2.77 MiB/s, done.
Resolving deltas: 100% (922/922), done.
ubuntu@ubuntu-server:~$
```

**Gambar 2.10** Melakukan *clone* proyek Password Management

Apabila file proyek sudah berada dalam *server* maka dilanjutkan dengan membuat Dockerfilenya. Sebelum melakukan pembuatan Dockerfile, maka sebaiknya pindah ke direktori *root* proyek agar memudahkan dalam melakukan perintah-perintah Docker. Selanjutnya penjelasan mengenai parameter pembuatan Dockerfile dapat dilihat pada **Modul 2.10** berikut ini:

```
FROM php:7.3-apache
RUN apt-get update && apt-get upgrade -y
RUN docker-php-ext-install mysqli
EXPOSE 80
```

```
RUN a2enmod rewrite
RUM chmod -R 755 /var/www/html/
COPY ./ /var/www/html
RUN service apache2 restart
```

Keterangan:

- FROM php:7.3-apache : menggunakan image php:7.3-apache
- upgrade : menginstall versi terbaru dari package yang dimiliki
- RUN : menjalankan command
- EXPOSE : meng-expose PORT yang digunakan untuk berkomunikasi
- a2enmod rewrite : command untuk mengaktifkan mod\_rewrite dalam apache
- chmod : command untuk memodifikasi permission dari suatu direktori
- COPY : command untuk menyalin sutau file atau direktori
- service apache2 restart : command untuk merestard apache server

### **Modul 2.10** Parameter pembuatan Dockerfile

Dalam pembuatan Manajemen Password ini diperlukan dua buah *service* yang harus berjalan secara bersamaan, yaitu *service web app* dan *service dataabase*. Kedua *service* tersebut akan dijalankan diatas dua buah *container*. Dikarenakan akan menjalankan dua *container* secara bersamaan, maka diperlukan Docker Compose yang diinisialisasi

dengan membuat `docker-compose.yml`. Inisialisasi dapat dilakukan pada direktori *root* proyek ataupun diluar dari direktori *root* proyek. Penjelasan mengenai parameter pembuatan `docker-compose.yml` dapat dilihat pada **Lampiran 2.1**.

Jika semua *service* yang diperlukan sudah selesai diinisialisasi pada `docker-compose.yml`, tahap selanjutnya adalah menjalankan *service* tersebut. Adapun parameter untuk menjalankan *service* melalui Docker compose dapat dilihat pada **Modul 2.11** berikut ini:

```
password-management$ docker-compose up -d
```

Keterangan:

- `docker-compose` : command untuk mengeksekusi perintah-perintah Docker Compose.
- `up` : parameter dalam `docker-compose` untuk menjalankan container.
- `-d` : parameter dalam `docker-compose` agar proses berjalan di background.

#### **Modul 2.11** Parameter eksekusi Docker compose

Diperlukan waktu agar semua *service* dapat berjalan dengan semestinya. Setelah 2 - 3 menit diperkirakan *container* sudah berjalan secara sempurna. Selanjutnya tahap terakhir yaitu melakukan eksekusi migrasi *database*. Migrasi *database* saat ini tidak dilakukan melalui PHP MySQL maupun *script* SQL akan tetapi dengan memanfaatkan fitur *migration* dalam CodeIgniter. Penjelasan parameter eksekusi *migration* dapat dilihat pada **Modul 2.12** berikut ini:

```
$ dokcer exec -it password-management php index.php migration migrate /bin/bash
```

Keterangan:

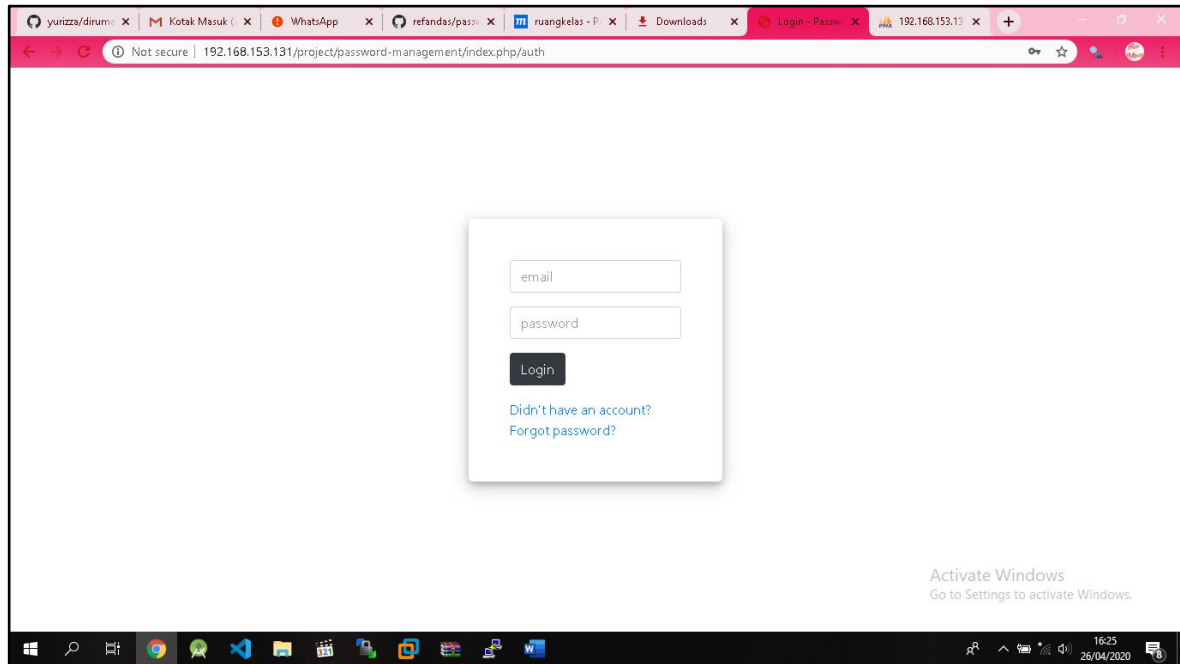
- `exec` : merupakan opsi dalam docker, berfungsi untuk menjalankan command pada suatu container yang sedang berjalan
- `-it` : merupakan opsi gabungan dari docker exec, terdiri dari `-i` dan `-t`. Fungsi dari `-i` yaitu untuk tetap mempertahankan STDIN terbuka meski tidak dilampirkan dan fungsi dari `-t` yaitu untuk mengalokasikan pseudo-TTY.
- `pass-manag-db` : merupakan nama container dimana kita ingin menjalankan command didalamnya, dapat diisi juga dengan id container
- `php` : merupakan inisialisasi penggunaan php cli
- `index.php` : berfungsi untuk mengakses file `index.php`
- `migration` : mengakses controller migration
- `migrate` : mengakses method migrate, dimana proses migration dijalankan pada method ini

#### **Modul 2.12** Parameter eksekusi *migration* dalam proyek

Apabila proses migrasi sudah selesai maka proyek Management Password ini dapat dibuka melalui browser dengan mengakses `http://[IP_SERVER]:[PORT]`.

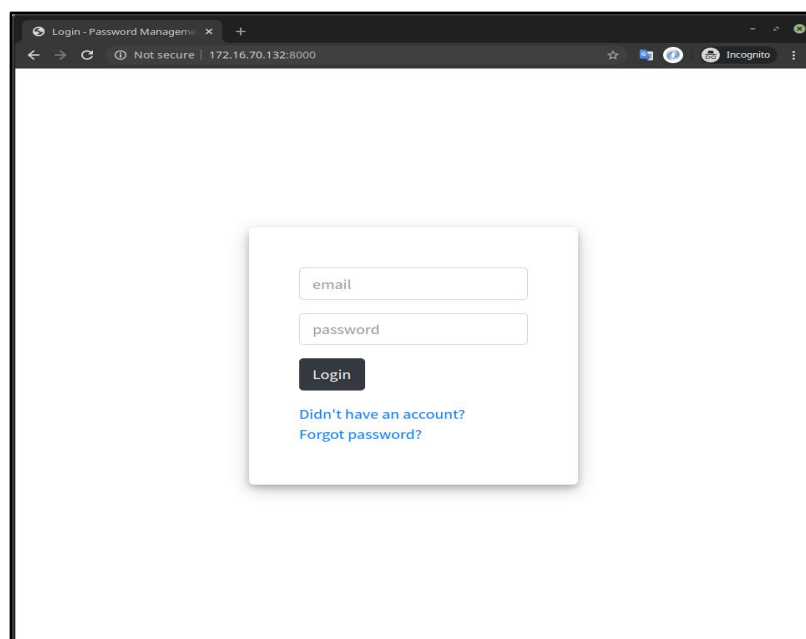
## 2.5 Hasil Implementasi

Setelah melakukan konfigurasi Ubuntu LAMPP dan integrasi sistem ke *server*, hasil proses *hosting local* dapat dilihat pada **Gambar 2.11** berikut ini:



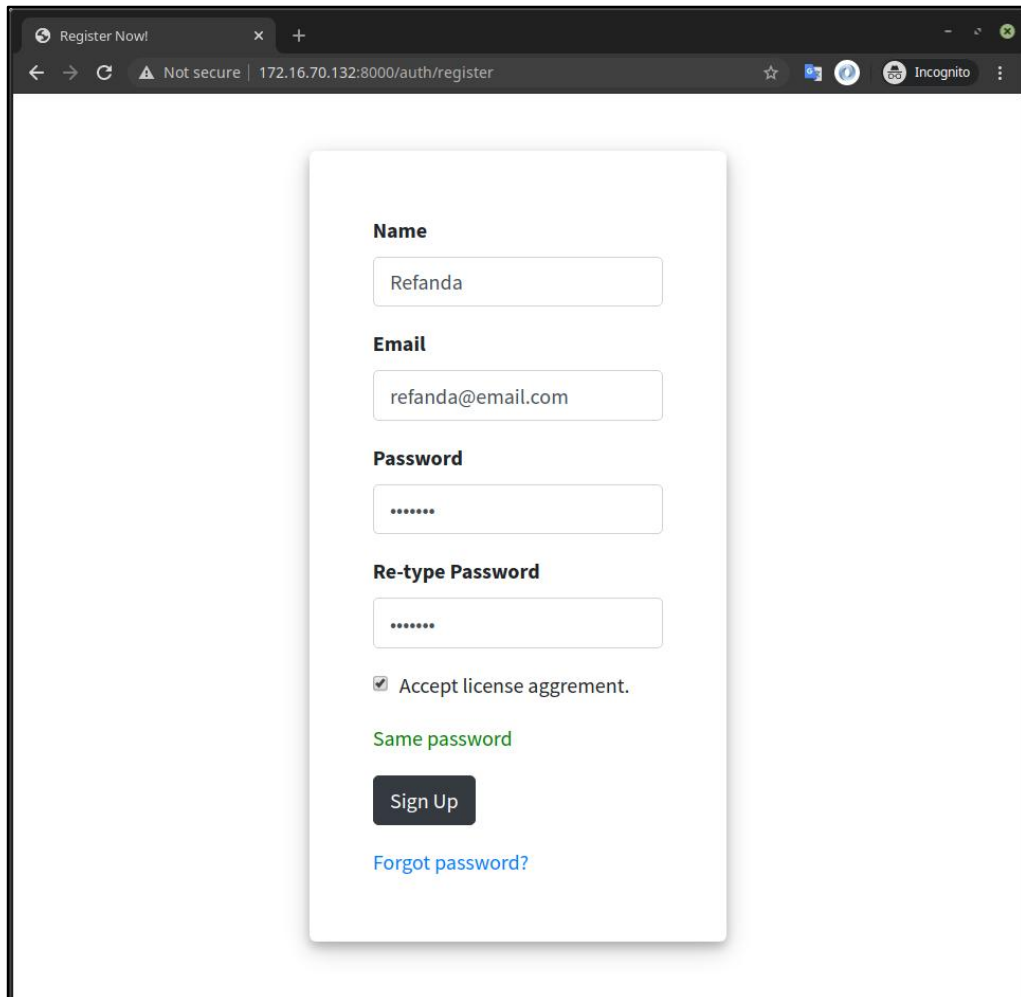
**Gambar 2.11** Tampilan proyek Manajemen Password

Kemudian untuk proyek yang berjalan diatas *container*, hasilnya dapat dilihat pada **Gambar 2.12** berikut ini:



**Gambar 2.12** Tampilan proyek yang berjalan diatas *container*

Dalam tahap implementasi ini terdapat dua skenario, yaitu membuat akun untuk masuk pada Manajemen Password dan mencoba melakukan penyimpanan pada sebuah *username* dan *password*. Proses pembuatan akun dilakukan dengan menekan link “*didn’t have an account?*”, adapun data yang diisikan tampak pada **Gambar 2.13** berikut ini:



Register Now!

← → ↻ ⚠ Not secure | 172.16.70.132:8000/auth/register ☆ 📧 🌐 Incognito

**Name**

Refanda

**Email**

refanda@email.com

**Password**

\*\*\*\*\*

**Re-type Password**

\*\*\*\*\*

☒ Accept license agreement.

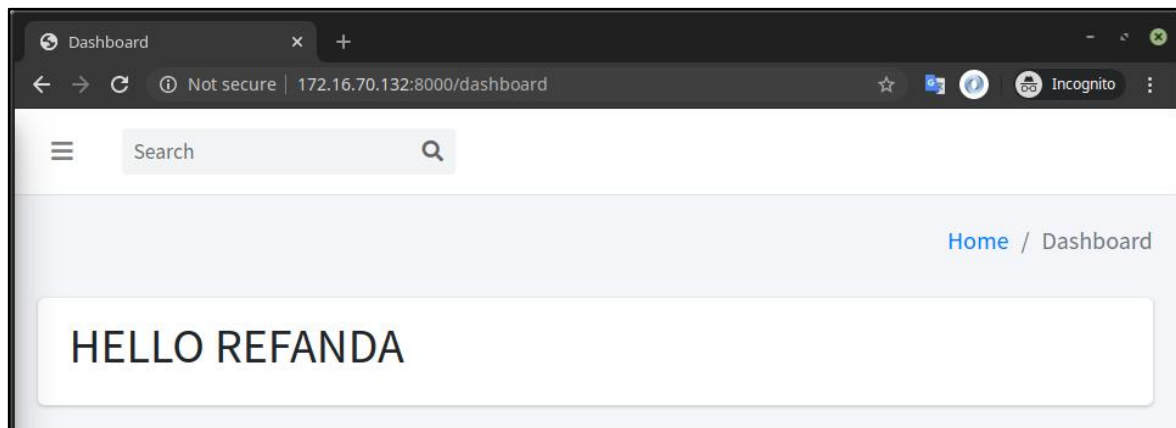
Same password

Sign Up

[Forgot password?](#)

**Gambar 2.13** Proses registrasi pada aplikasi Manajemen Password

Untuk mencoba apakah proses registrasi berhasil atau tidak, maka mencoba melakukan *login* pada aplikasi, **Gambar 2.14** menampilkan hasil setelah proses *login* menggunakan *username* dan *password* yang sama dengan yang didaftarkan.



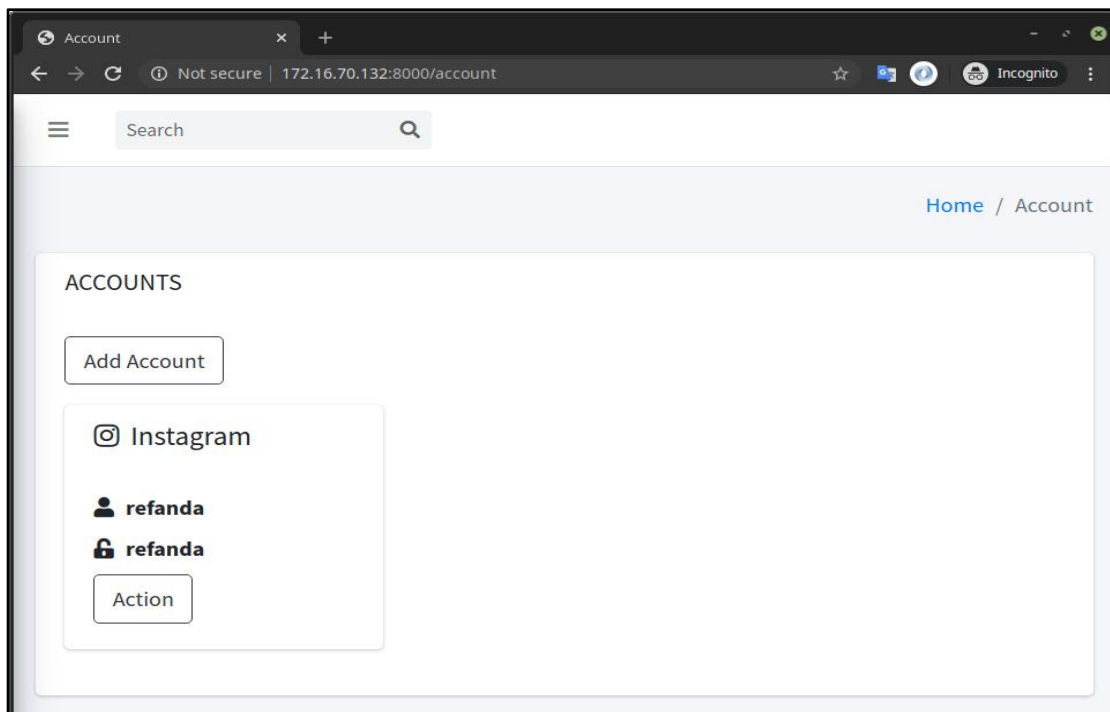
**Gambar 2.14** Hasil dari proses login berhasil

Skenario pertama sudah berhasil, kemudian dilanjutkan dengan skenario kedua. Dalam skenario kedua ini akan mencoba menyimpan satu buah *username* dan *password* dari akun sosial media. **Gambar 2.15** menampilkan isi data yang *form* yang akan disimpan pada bagian *account*.

A screenshot of a web browser displaying the "Add Account" form. The browser's address bar shows the URL "172.16.70.132:8000/account/add\_account". The form is titled "ADD ACCOUNTS" and contains three input fields: "Username" with the value "refanda", "Password" with the value "refanda", and "Account Media" with a dropdown menu showing "Instagram". Below the input fields is a button labeled "Add New". The breadcrumb navigation at the top right shows "Home / Account / Add Account". The footer of the page contains the text "Copyright © 2014-2019 AdminLTE.io. All rights reserved." and "Version 3.0.0-rc.5".

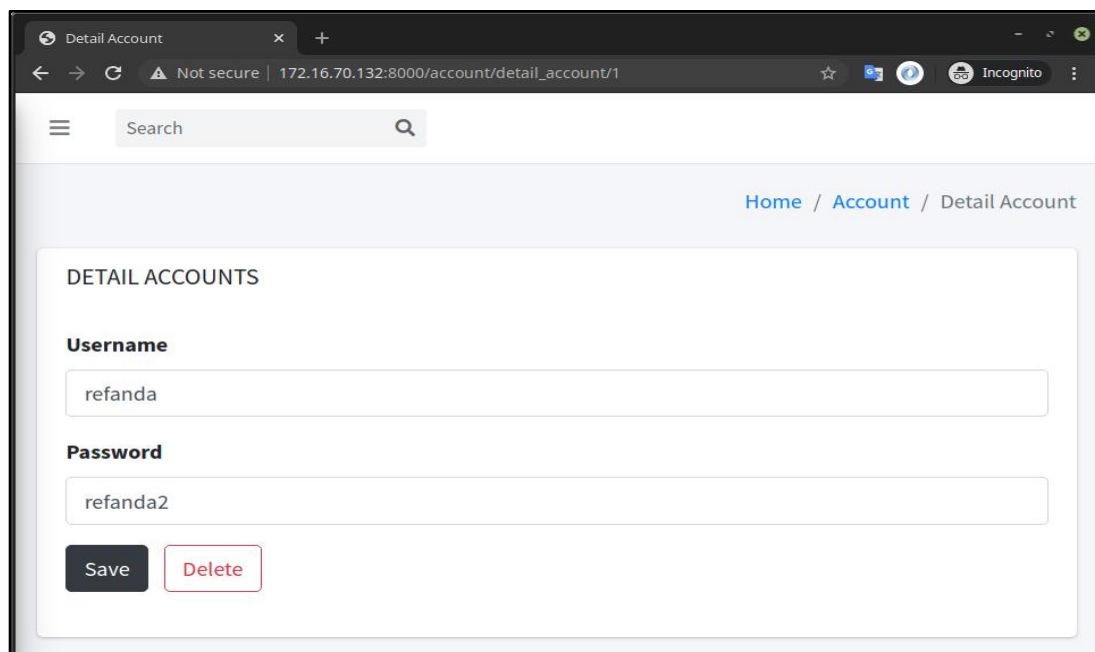
**Gambar 2.15** Proses penambahan akun yang akan disimpan

Kemudian dilanjutkan dengan menekan tombol “*Add New*”, untuk mengetahui apakah akun sudah tersimpan atau belum, maka membuka halaman *account* seperti pada **Gambar 2.16** berikut ini:



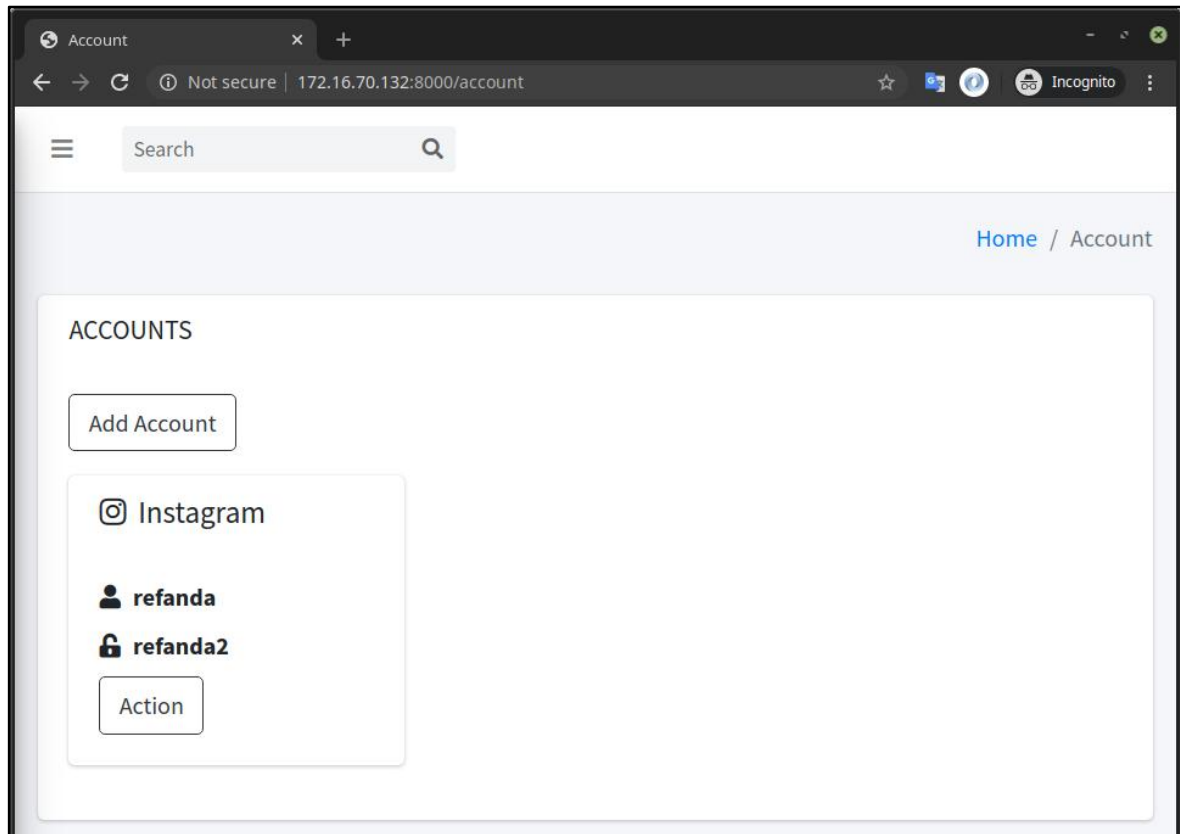
**Gambar 2.16** Tampilan halaman *account*

Selain menambah akun, pengguna juga dapat melakukan pengeditan terhadap akun yang ada. Sebagai contoh, pengguna akan memperbarui *password* akun Instagram, form edit seperti pada **Gambar 2.17** berikut ini:



**Gambar 2.17** Pengubahan data *password* pada akun Instagram

Ketika tombol “*Save*” ditekan maka program akan mengalihkan halaman edit menuju halaman *account*, dalam halaman *account* yang ditampilkan pada **Gambar 2.18**, menampilkan *password* yang berbeda dengan sebelumnya, sehingga proses edit *password* berhasil.

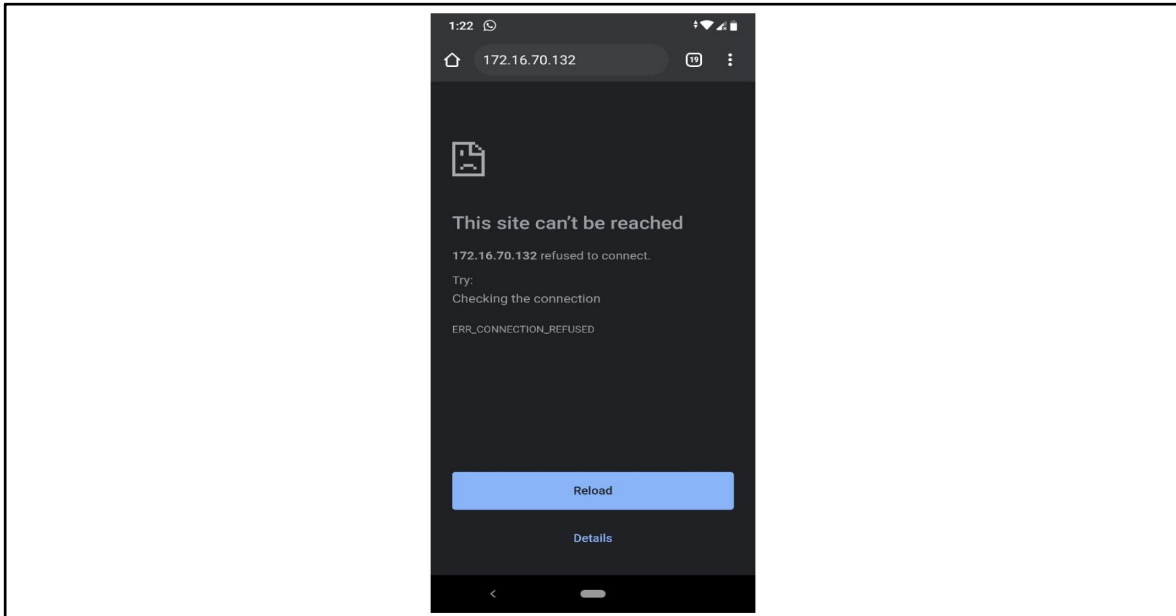


**Gambar 2.18** Tampilan halaman *account* setelah mengedit akun Instagram

## 2.6 Pengujian Singkat

Dalam pengujian singkat ini, terdapat dua hal yang akan diuji. Pertama yaitu menguji tugas pertama dimana membuat *hosting* lokal dengan menggunakan Ubuntu LAMPP, dalam pengujian ini hal yang dilakukan adalah mencoba untuk mengakses halaman *web* melalui perangkat *mobile*, hal ini bertujuan untuk mengetahui apakah pada tugas pertama memang dapat diakses dimanapun. Hasil pengujian singkat ini dapat dilihat pada **Gambar 2.19** berikut ini

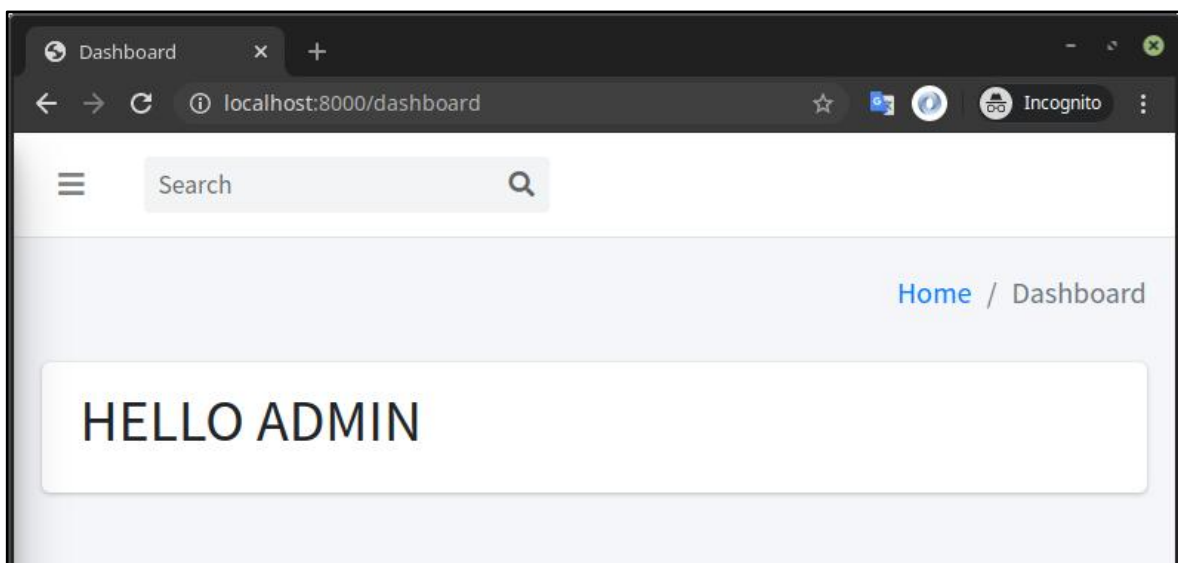




**Gambar 2.19** Hasil pengujian akses dengan perangkat *mobile*

Kegagalan dalam pengujian kemungkinan dikarenakan salah dalam melakukan konfigurasi jaringan pada *virtual machine*.

Selanjutnya untuk pengujian yang kedua, akan dilakukan pengujian berupa menjalankan proyek dengan komputer lokal. Dalam pengujian ini *operating system* yang digunakan yaitu Linux Mint 19. Tujuan dari pengujian ini yaitu untuk mengetahui apakah proyek dapat di-*install* dalam *host* lain dengan mudah atau tidak. Hasil pengujian singkat ini dapat dilihat pada **Gambar 2.20** berikut ini:



**Gambar 2.20** Tampilan pengujian halaman dashboard

Pengujian kedua berhasil dilakukan, dimana mencoba untuk melakukan instalasi pada komputer lokal dan dari hasil yang ditampilkan sistem dapat berjalan dengan baik.

### BAB III

#### JADWAL Pengerjaan dan Pembagian Tugas

##### 3.1 Agenda Pengerjaan

Berikut pada **Tabel 3.1** merupakan pembagian jenis tugas proyek akhir terhadap alokasi waktu pengerjaan pada bulan Maret dan April tahun 2020:

**Tabel 3.1** Agenda Pengerjaan Proyek

No.	Jenis Tugas	Waktu Pengerjaan							
		Maret				April			
		1	2	3	4	1	2	3	4
1.	Analisa Persoalan								
2.	Pembagian Tugas								
3.	Pengerjaan Tugas 1 (LAMPP)								
4.	Pembuatan Laporan Submisi								
5.	Pengerjaan Tugas 2 (Docker)								
6.	Melanjutkan Pembuatan Laporan								
7.	Revisi								
8.	Presentasi Proyek Akhir								

##### 3.2 Keterangan Pembagian Tugas

Berikut pada **Tabel 3.2** merupakan pembagian tugas-tugas pada proyek akhir terhadap anggota pada tim pembuatan proyek akhir:

**Tabel 3.2** Pembagian Tugas Proyek

No.	Keterangan Tugas	Penanggung Jawab
1.	Perancangan Arsitektur Cloud Computing	Alivi
2.	Pengujian Singkat	Alivi
3.	Latar Belakang Masalah	Refanda
4.	Agenda Pengerjaan Proyek	Refanda
5.	Pembuatan Tugas 1 (LAMPP)	Alivi
6.	Pembuatan Tugas 2 (Docker)	Refanda
7.	Pembuatan Laporan	Alivi dan Refanda

## **BAB IV**

### **KESIMPULAN DAN SARAN**

#### **4.1 Kesimpulan**

Berdasarkan masalah yang ada, yaitu membuat aplikasi manajemen *password* diperoleh hasil yang memuaskan. Sistem Manajemen Password dapat berjalan dengan baik entah dijalankan melalui *hosting local* maupun melalui Docker *container*. Dari pengujian sistem, semua skenario yang dibuat dapat berjalan dengan baik sesuai dengan yang diharapkan.

Pembagian tugas terlaksana dengan baik dan sesuai dengan waktu yang ditentukan, Alivi tugas *hosting local* dengan Ubuntu LAMPP dan Refanda tugas membuat Dockerfile.

#### **4.2 Saran**

Terdapat perbedaan dalam penggunaan software VMWare, dimana saat pengerjaan *hosting local* menggunakan VMWare Workstation Pro sedangkan saat pengerjaan Dockerfile menggunakan VMWare Player. Akan tetapi hal tersebut tidak menjadi masalah yang berarti karena semua berjalan dengan baik.

## DAFTAR PUSTAKA

Erl, T., Puttini, R., & Mahmood, Z. (2013). *Cloud computing: concepts, technology & architecture*. Pearson Education.

Ernawati, T., & Zulfiaji, A. H. (2014). *Analisis dan pembangunan infrastruktur Cloud Computing*. Jurnal Cybermatika, 1(2).

Datacomm. *Definisi Cloud Computing*. <https://datacommcloud.co.id/definisi-cloud-computing>. Diakses 6 Juni 2016.

Docker Docs. *Orientation and setup*. <https://docs.docker.com/get-started>. Diakses 30 April 2020.

## LAMPIRAN

```
version: '3.3'
services:
  web:
    build:
      context: ./
      dockerfile: Dockerfile
    container_name: password-management
    depends_on:
      - db
    volumes:
      - ./:/var/www/html/
    ports:
      - 8000:80
  db:
    container_name: pass-manag-db
    image: mysql:8.0
    command: --default-authentication-plugin=mysql_native_password
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: password_manager
      MYSQL_USER: latihan
      MYSQL_PASSWORD: latihan
    ports:
      - 6033:3306
```

### Keterangan:

- version : merupakan versi dari docker-compose.yml.
- web : merupakan nama dari image yang akan di-build.
- build : berisi proyek yang akan di build.
- context : bagian dari build, dimana pada context berisi direktori dimana proyek berada. Karena docker-compose.yml diletakkan satu direktori dengan direktori root proyek, maka dalam kasus ini context diisi dengan ./ atau mereferensik pada folder dimana dokcer-compose.yml itu berada.
- dockerfile : nama dari file Dockerfile.
- depends\_on : kebergantungan container pada container yang lain.
- volumes : melakukan mount pada direktori ketika service dijalankan.
- ports : perintah untuk mengekspose port, dalam kasus ini diisi 8000:80 yang berarti 80 untuk port internal container dan 8000 untuk port eksternal. Sehingga jika ingin mengakses layanan secara eksternal maka menggunakan port 8000.
- db : merupakan nama dari image yang akan di-build.
- container\_name : merupakan nama container yang akan dibuat.
- image : mendefinisikan image yang akan digunakan dalam membuat container.
- command : menjalankan beberapa perintah selama container dijalankan di awal.
- restart : jika mengalami kesalahan maka container akan di-restart
- environment : merupakan variabel yang tersedia pada lingkungan container

### Lampiran 2.1 Parameter konfigurasi Docker compose