

PRAKTIKUM TEKNOLOGI CLOUD COMPUTING
LAPORAN PROYEK AKHIR

SISTEM MONITORING ACCOUNT MANAGER OS UBUNTU MENGGUNAKAN
LAMPP DAN PROSES PEMBUATAN DOCKER FILE-NYA



DISUSUN OLEH:

NAMA ANGGOTA : RAHMAT ZUMARLI 123170011
FHREZHA ZEANETH 123170044
KELAS : E
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 MONITORING ACCOUNT MANAGER OS UBUNTU MENGGUNAKAN LAMPP DAN PROSES PEMBUATAN DOCKER FILE-NYA

Disusun oleh :

Rahmat Zumarli

123170011

Fhrezha Zeaneth

123170044

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

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa yang senantiasa mencurahkan rahmat dan hidayah-Nya sehingga kami dapat menyelesaikan praktikum Teknologi *Cloud Computing* serta laporan proyek akhir praktikum yang berjudul Sistem Monitoring *Account Manager OS* Ubuntu Menggunakan *LAMPP* dan Proses Pembuatan *Docker* filenya. Adapun laporan ini berisi tentang proyek akhir yang kami pilih dari hasil proyek Kerja Praktek yang kami lakukan.

Sekilas mengenai proyek Kerja Praktek yang kami pilih merupakan proyek dari suatu divisi pada perusahaan PT. Gamatechno Indonesia. Dalam satu Divisi Marketing, Super Visi membutuhkan suatu sistem dimana dapat memantau kinerja dari staff mereka ,yaitu *Account Manager* (AM). Pemantauan ditujukan terhadap aktifitas dan perkembangan proyek setiap AM. Hak akses Super Visi dapat memantau seluruh data AM dan *Customer*, sedangkan AM hanya dapat mengakses akun mereka masing-masing. Proses CRUD akun AM dibawah kendali Super Visi, sedangkan proses CRUD *activity* dan *project* dikendalikan oleh masing-masing AM.

Tidak lupa kami ucapkan terimakasih kepada asisten dosen yang selalu membimbing dan mengajari kami dalam melaksanakan praktikum dan dalam menyusun laporan ini. Laporan ini masih sangat jauh dari kesempurnaan, oleh karena itu kritik serta saran yang membangun kami harapkan untuk menyempurnakan laporan akhir ini.

Atas perhatian dari semua pihak yang membantu penulisan ini, kami ucapkan terimakasih. Semoga laporan ini dapat dipergunakan seperlunya.

Yogyakarta, 15 April 2020

Penyusun

DAFTAR ISI

HALAMAN COVER	ii
HALAMAN PENGESAHAN	ii
KATA PENGANTAR	iii
DAFTAR ISI	iv
BAB I PENDAHULUAN	1
1.1. Latar Belakang Proyek Akhir	1
1.2. Tujuan Proyek Akhir	5
1.3. Manfaat Proyek Akhir	5
1.4. Tahap Penyelesaian Proyek Akhir	6
BAB II ISI DAN PEMBAHASAN	7
2.1. Komponen yang Digunakan	7
2.2. Rancangan Arsitektur <i>Cloud Computing</i>	8
2.3. Parameter dan Konfigurasi	10
2.4. Tahap Implementasi	19
2.5. Hasil Implementasi	23
2.6. Pengujian Singkat	28
BAB III JADWAL Pengerjaan dan Pembagian Tugas	29
3.1. Agenda Pengerjaan	33
3.2. Keterangan Pembagian Tugas	33
BAB IV KESIMPULAN DAN SARAN	34
4.1. Kesimpulan	34
4.2. Saran	34
DAFTAR PUSTAKA	35
LAMPIRAN	36

BAB I

PENDAHULUAN

1.1. Latar Belakang Proyek Akhir

Cloud computing bermula pada tahun 1960-an oleh John McCarthy seorang pakar komputasi MIT dan salah satu pionir *intelligent* buatan. Beliau menyampaikan sebuah visi mengenai komputasi bahwa “suatu hari nanti, komputasi akan menjadi infrastruktur publik seperti listrik dan telepon”. Namun di tahun 1995 Larry Ellison seorang pendiri Oracle menciptakan sebuah ide yang disebut “*Network Computing*” dimana merupakan ajang kampanye untuk menggugat dominasi Microsoft yang menjadi raja dekstop *computing* dengan Windows 95- miliknya. Larry menawarkan ide dengan prinsip *user* tidak perlu banyak menggunakan *software*, mulai dari OS dan *software* lain akan dimasukkan dalam PC Dekstop *user*. PC Dekstop dapat digantikan oleh terminal yang terhubung dengan server penyedia *enviromtment* dan berisi berbagai kebutuhan *software* siap pakai. Ide dari Larry Ellison sempat menjadi sorotan publik bersamaan dengan munculnya beberapa pabrikan seperti Sun Microsystem dan Novell Netware yang menawarkan hal serupa yaitu Network Computing *client*. Namun karena kualitas jaringan komputer yang belum memadai dan membuat akses *Network Computing* menjadi sangat lambat maka *user* kembali menggunakan PC dan *Network Computing* hilang dengan perlahan. Tahun 2000, Amazon memiliki peran penting dalam pengembangan *cloud computing* dengan melakukan modernisasi pusat data seperti jaringan komputer dengan 10% kapasitas mereka pada satu waktu. Setelah menemukan arsitektur *cloud*, perkembangan mengalami peningkatan efisiensi internal, kemudian Amazon mengembangkan produk baru sebagai penyedia *cloud computing* untuk pelanggan eksternal dengan peluncuran Amazon Web Service (AWS) tahun 2006.

Cloud computing merupakan teknologi yang menjadikan internet sebagai pusat *server* untuk mengelola data dan juga aplikasi pengguna dimana mengizinkan para pengguna untuk menjalankan program tanpa instalasi dan mengizinkan pengguna untuk mengakses data pribadi mereka melalui komputer dengan akses internet. Pada *cloud computing* sumber daya seperti *processor/computing power, storage, network, software* menjadi abstrak (*virtual*) dan diberikan sebagai layanan di jaringan/internet juga dapat menggabungkan beberapa perangkat komputer menjadi satu kesatuan (*cluster*) dan membuat membuat

banyak *server* pada satu perangkat komputer dengan virtualisasi. *Cloud computing* tercipta karena timbulnya kendala seperti keterbatasan atau pemborosan *resource* komputer yang menyebabkan terhambatnya beberapa kegiatan perkomputasian. Agar terciptanya efisiensi, perusahaan-perusahaan besar di bidang TI (Teknologi Informasi) pun sekarang beralih menggunakan teknologi *cloud computing*. Terdapat beberapa manfaat *cloud computing* yaitu :

1. Menyimpan semua data pada *server* yang terpusat sehingga pengguna tak perlu repot menyediakan infrastruktur seperti *data center*, media penyimpanan */storage* dll karena semua telah tersedia secara virtual.
2. Keamanan data pengguna disimpan dengan aman melalui *server* yang disediakan penyedia layanan *Cloud Computing* seperti jaminan platform teknologi, jaminan ISO, data pribadi, dll.
3. Fleksibilitas dengan kemudahan data akses, kapan dan dimanapun kita berada dengan catatan bahwa pengguna (*user*) terkoneksi dengan internet. Pengguna juga dapat dengan mudah meningkatkan atau mengurangi kapasitas penyimpanan data tanpa perlu membeli peralatan tambahan seperti hardisk.
4. Menghemat biaya pembelian inventaris karena pengguna akan dikenakan biaya kompensasi rutin per bulan sesuai dengan paket layanan yang telah disepakati dengan penyedia layanan *Cloud Computing*.

Dilihat dari segi fisik, cloud merupakan sebuah kumpulan komputer dan server yang dapat diakses secara publik menggunakan internet. Salah satu hal yang penting dalam cloud computing adalah arsitekturnya, dimana merupakan kemampuan pemrosesan dari sistem yang bukan hanya tampilan dekstop, dan semua user dapat mengakses komputasi yang telah disediakan oleh server cloud selama menggunakan jaringan internet. Arsitektur cloud computing dapat dijelaskan melalui **gambar 1.1**. Arsitektur cloud computing sangat sederhana dan tidak membutuhkan manajemen khusus untuk menghubungkan semua komputer dan melakukan alokasi pemrosesan task ke user. User hanya perlu mengakses menggunakan user interface di komputer, kemudian akan otomatis dapat mengakses task. Request dari user dapat dilewatkan ke manajemen sistem. Layanan ini akan mengambil sumber daya yang ada di cloud kemudian menjalankan aplikasi web tertentu dan aplikasi web lainnya. Setelah aplikasi web dijalankan, sistem akan memonitor dan mengontrol fungsi penggunaan cloud sehingga sumber daya dapat terbagi berdasar kebutuhan.

Dalam kehidupan nyata, penggunaan *cloud computing* dapat dilihat pada penggunaan *Amazon Web Services* (AWS) yang disediakan oleh *Amazon* dan dapat diakses di

<https://aws.amazon.com/id/about-aws/> . *Amazon Web Services* memiliki lebih banyak layanan, dan fitur dalam layanannya dibandingkan penyedia *cloud* lainnya, mulai dari teknologi infrastruktur seperti penghitungan, penyimpanan, dan *database* hingga teknologi yang berkembang, seperti dan kecerdasan buatan, data *lake* dan analitik, dan *Internet of Things*. Hal ini membuat seluruh proses komputasi data menjadi lebih cepat, lebih mudah, dan lebih hemat biaya. AWS juga memiliki fungsionalitas detail dalam semua layananannya. Misalnya, AWS menawarkan ragam database paling luas yang dibangun untuk beragam jenis aplikasi sehingga dapat memilih *tools* yang tepat untuk pekerjaan tersebut dengan mendapatkan biaya dan kinerja terbaik. Biaya yang dikenakan juga mengikuti penggunaan member. AWS menyediakan dukungan untuk kedua solusi *open-source* dan komersial *Docker*. Ada sejumlah cara untuk menjalankan kontainer di AWS, termasuk Amazon Elastic Container Service (ECS) yang merupakan layanan pengelolaan kontainer berkinerja tinggi dan sangat mudah diskalakan. AWS Fargate adalah teknologi untuk Amazon ECS yang memungkinkan untuk menjalankan kontainer di produksi tanpa menerapkan atau mengelola infrastruktur. Amazon Elastic Container Service for Kubernetes (EKS) memudahkan *user* dalam menjalankan Kubernetes di AWS. AWS Fargate adalah teknologi untuk Amazon ECS yang memungkinkan *user* menjalankan kontainer tanpa menyediakan atau mengelola server. Amazon Elastic Container Registry (ECR) adalah repositori kontainer pribadi yang tersedia dengan sangat baik dan aman sehingga memudahkan menyimpan dan mengelola gambar kontainer Docker *user*, mengenkripsi dan mengompresi gambar saat istirahat sehingga cepat ditarik dan aman. AWS Batch memungkinkan *user* dalam menjalankan beban kerja pemrosesan *batch* yang sangat mudah diskalakan menggunakan kontainer *Docker*. Terdapat tiga penawaran dari *Amazon Web Services* yang disebut solusi ,yaitu berdasar kasus penggunaan, industri, dan jenis organisasi. Contoh pada kasus penggunaan yaitu pengarsipan data, pencadangan dan pemulihan, *blockchain*, dan lain sebagainya. Contoh pada industri yaitu periklanan dan pemasaran, media dan hiburan, telekomunikasi dan lain-lain. Sedangkan pada jenis organisasi yaitu seperti *start-up*.

Projek pada penggunaan cloud computing ini berjudul Sistem Monitoring *Account Manager OS Ubuntu Menggunakan LAMP dan Proses Pembuatan Docker* filenya. dimana merupakan sebuah sistem yang dapat memantau kinerja staff dari satu divisi. Pemantauan kinerja dalam *activity* dan *project*. Setiap staff memiliki akun masing-masing dan begitu juga dengan kepala divisi tersebut. Hak akses staff hanya pada akun masing-masing, staff dapat membuat, mengubah, menghapus, dan melihat *activity* serta *project* mereka masing-

masing, namun tidak dapat melakukan modifikasi terhadap kinerja staff lain. Kepala divisi berhak dalam pembuatan atau penghapusan akun staff. Namun hanya dapat melihat (*monitoring*) *activity* serta *project* seluruh staff. Sistem ini hanya untuk satu divisi yaitu marketing. Dengan menggunakan *cloud computing*, sistem Monitoring Account Manager OS Ubuntu Menggunakan LAMPP dan Proses Pembuatan *Docker* filenya. dapat digunakan dimana saja. *Cloud computing* membantu sistem monitoring agar dapat bekerja lebih cepat dengan tingkat produktivitas yang tinggi. Disisi lain, dengan dibentuknya sistem monitoring dengan cloud computing maka akan memberikan kelebihan seperti *reability* dan *cost*. *Reability* dalam artian *cloud computing* dapat membuat cadangan data, dan sinkronisasi penggunaan untuk user lebih mudah dan lebih murah karena data dapat dicerminkan di beberapa situs berlebihan di jaringan penyedia *cloud*. Sistem Monitoring Account Manager OS Ubuntu Menggunakan LAMPP dan Proses Pembuatan *Docker* filenya dengan *cloud computing* juga memberikan kemudahan bagi *user* yang menggunakan seperti jika mendadak akan menggunakan aplikasi maka aplikasi dapat diakses dimana saja, dengan syarat sinyal memadai, maka dengan ini kita tidak memerlukan satu komputer atau satu *handphone* untuk mengaplikasikan Sistem Monitoring Account Manager OS Ubuntu Menggunakan LAMPP dan Proses Pembuatan *Docker* filenya melainkan dapat digunakan dimana saja. Dengan penggunaan *Docker* juga semakin mempermudah pengembangan terhadap kode, penyaluran pada pipa, sampai menyediakan berbagai lingkungan yang lebih konsisten dalam aplikasi pengembangan hingga ke tahan produksi. Kelebihan *Docker* seperti penempatan lingkungan serta konfigurasi menuju kode dan menyebarkannya menjadikan *Docker* dapat digunakan juga dalam banyak lingkungan, dengan memisahkan kebutuhan infrastruktur pada lingkungan aplikasi tersebut karena konfigurasi yang sama. Adanya *Docker* juga semakin membuat *developer* bebas dalam menjalankan berbagai aplikasi dalam beberapa IaaS/PaaS dengan tidak menggunakan *tweak* tambahan. *Docker* dapat mempersingkat waktu pemrosesan sehingga dapat memberikan kepastian kepada *client* dengan hasil yang sesuai keinginan mereka. Hal menarik lainnya yang dapat dirasakan yaitu penggunaan multi-tenance yang mana dapat menghindarkan penulisan ulang pada aplikasi yang utama. Kemampuan yang dimiliki oleh aplikasi isolasi *Docker* dapat menggabungkan banyak server, dengan demikian dapat menghemat pengeluaran. Adanya *Docker* juga memberikan konsolidasi server yang lebih padat dibandingkan apa yang bisa dilakukan di *virtual machine*.

Dalam pembuatan projek akhir Sistem Monitoring Account Manager OS Ubuntu Menggunakan LAMPP dan Proses Pembuatan *Docker* filenya ini, beberapa tahapan yang perlu dilakukan yaitu :

1. Menentukan judul proyek akhir dan rancangan sistem proyek akhir yang akan Dibuat.
2. Mempersiapkan komponen software yang dibutuhkan seperti *VMWare Workstation* atau *Virtual Box*, *Docker*, *ISO Ubuntu*, *LAMPP* dll.
3. Melakukan instalasi *Virtual Machine* yang akan digunakan sebagai *Virtual Machine* Sistem Operasi Ubuntu.
4. Melakukan instalasi *ubuntu server* yang akan digunakan sebagai sistem operasi penunjang *cloud computing*.
5. Menginstall *software* yang akan digunakan sebagai media instalasi dan konfigurasi
6. Melakukan installasi *PHP*, *Mysql*, dan *Apache* yang akan digunakan sebagai media instalasi dan software pada ubuntu
7. Melakukan konfigurasi *PHP*, *Mysql*, dan *Apache* dll sesuai dengan kebutuhan.
8. *Testing software* sesuai kebutuhan

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 Monitoring Account Manager OS Ubuntu Menggunakan LAMPP dan Proses Pembuatan *Docker* filenya yang telah dibuat sebelumnya dengan menggunakan Ubuntu 18.04.3 LTS dan LAMPP (Apache 2.4, PHP 7.3, MySQL).
2. Mengintegrasikan Sistem *Monitoring Account Manager* yang berada di Ubuntu Serve LAMPP dengan platform produk layanan yang menggunakan virtualisasi tingkat OS *Docker* untuk digunakan sebagai media pengiriman perangkat lunak dalam paket yang disebut wadah.

1.3 Manfaat Proyek Akhir

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

1. Sistem Monitoring yang telah dirancang dapat digunakan sewaktu-waktu tanpa khawatir *downtime* dikarenakan terdapatnya *primary* dan *backup server* yang menggunakan arsitektur *recovery* pada *cloud computing*.
2. Perusahaan tidak perlu memperlakukan *maintenance*, dikarenakan dengan menggunakan *cloud computing*, rutinitas *maintenance* akan dilakukan sepenuhnya oleh *vendor*.

3. Batasan memori penyimpanan multimedia menjadi tidak terbatas dikarenakan sistem telah sepenuhnya beralih menggunakan *cloud computing*.
4. Permintaan data secara *realtime* dapat dilakukan secara terpusat maupun secara terpilih.
5. Biaya dalam perawatan sistem juga tidak mahal, hanya biaya rutin sesuai penggunaan, dan mengurangi pengeluaran perusahaan karena tidak ada lagi biaya tambahan untuk pembelian inventaris seperti infrastruktur, hardisk, dll.
6. Dengan menggunakan *Docker* juga dapat memberi manfaat berupa kemudahan dalam pengoperasian sistem, dan dapat sebagai acuan atau contoh dalam pembuatan sistem menggunakan *Docker*.

1.4 Tahap Penyelesaian Proyek Akhir

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

1. Menganalisis kebutuhan dari Sistem Monitoring Account Manager untuk ditransformasikan ke dalam arsitektur cloud computing menggunakan basis IaaS/SaaS/PaaS/DBaaS dan XaaS/WaaS.
2. Mengintegrasikan penyimpanan data record pada *Docker* dengan Sistem Monitoring Account Manager yang berada di Ubuntu Server.
3. Menentukan konfigurasi yang tepat untuk pengaturan *Docker* sehingga dapat digunakan sesuai *requirement* yang berupa:
 - a. Dapat diakses oleh jaringan/IP tertentu saja.
 - b. Terdapat divisi pusat yang dapat masuk ke semua akun kecuali akun Z.
 - c. Menggunakan Ubuntu dengan versi 18.04.3 LTS.
 - d. Menggunakan PHP 7.3
 - e. Virtualisasi menggunakan *docker*
4. Merancang topologi cloud computing untuk mengintegrasikan dua sub sistem yang berbeda sehingga dapat digunakan secara terintegrasi.
5. Melakukan konfigurasi Ubuntu Server sebagai *primary* dan *backup* untuk dapat digunakan sebagai *recovery* sehingga Sistem Monitoring Account Manager ketersediaannya/availability-nya maksimal.
6. Menguji keandalan arsitektur cloud computing yang dibangun dengan beberapa pengujian yaitu kecepatan waktu akses, batasan akses sesuai konfigurasi, dsb.

BAB II

ISI DAN PEMBAHASAN

2.1 Komponen yang Digunakan

Untuk membangun “Sistem *Monitoring Account Manager OS Ubuntu* Menggunakan LAMPP dan Proses Pembuatan Docker filenya” yang berbasiskan 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 penyimpanan MySQL versi 5.2 sehingga dapat digunakan untuk menyimpan berbagai data yang dibutuhkan oleh sistem tersebut.
3. Untuk virtualisasi menggunakan *Virtual Box Graphical User Interface* versi 6.0.18 r136238 (Qt5.6.2) dimana *virtual machine* ini digunakan untuk melakukan instalasi ubuntu server.
4. Apache sebagai web server
5. Untuk membangun aplikasi, mengemas dan menjalankan aplikasi pada project ini adalah Docker dengan versi 2.2.0.5 (44384) *stable*.
6. Untuk target pengguna dengan konsep *cloud computing*, maka penggunaanya ialah seluruh karyawan divisi marketing di PT.Gamatechno Indonesia yang ber-kantor di Yogyakarta. Tidak ada yang dapat mengakses sistem tersebut kecuali harus terhubung melalui jaringan intranet PT.Gamatechno Indonesia. Sehingga diperlukan arsitektur *cloud computing* yang bersifat *private*, tidak dapat diakses secara bebas oleh semua orang kecuali orang yang termasuk dalam divisi marketing.

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	VirtualBox Graphical User Interface	Tidak menggunakan <i>hardware</i> fisik secara langsung, melainkan menggunakan aplikasi <i>virtual machine</i> .
2.	Prosesor	Intel(R) Core(TM) @ 3.60GHz	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: 192.168.43.164/24	Alamat IP dan <i>network</i> yang digunakan oleh <i>guest OS</i> .
		DNS: 127.0.0.53	Alamat IP untuk DNS <i>guest OS</i> .
		GW: 192.168.8.1	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.	Penyimpanan Data	15GB	Penyimpanan 15 GB digunakan untuk root

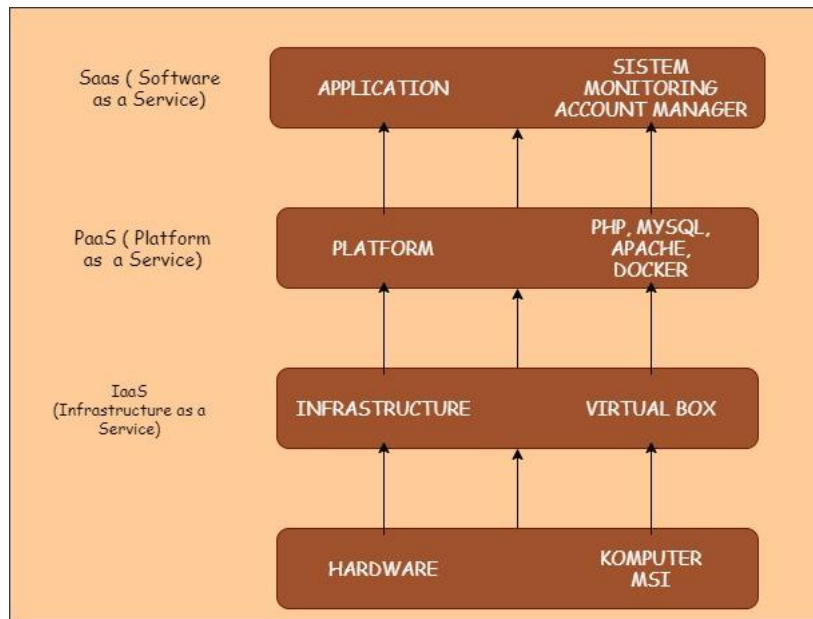
Selain spesifikasi mengenai VM *cloud computing* tersebut, untuk spesifikasi yang digunakan dalam Ubuntu OS yang telah dibuat dalam VM tersebut adalah sebagai berikut:

Tabel 2.2 Spesifikasi Ubuntu OS untuk proyek pertama

No.	Nama Parameter	Nilai	Keterangan
1.	LAMP	Apache 2.4	Preprosesor bahasa pemrograman HTML, termasuk CSS dan JS.
		PHP 7.3	bahasa pemrograman <i>script server-side</i> yang didesain untuk pengembangan web. Selain itu, PHP juga bisa digunakan sebagai bahasa pemrograman umum
		MySQL	sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS (<i>Database Management System</i>) yang multialur dan multipengguna.
2.	Docker	Versi 2.2.0.5 (44383)	aplikasi yang bersifat <i>open source</i> yang berfungsi sebagai wadah/ <i>container</i> untuk mengepak/memasukkan sebuah <i>software</i> secara lengkap beserta semua hal lainnya yang dibutuhkan oleh <i>software</i> tersebut dapat berfungsi

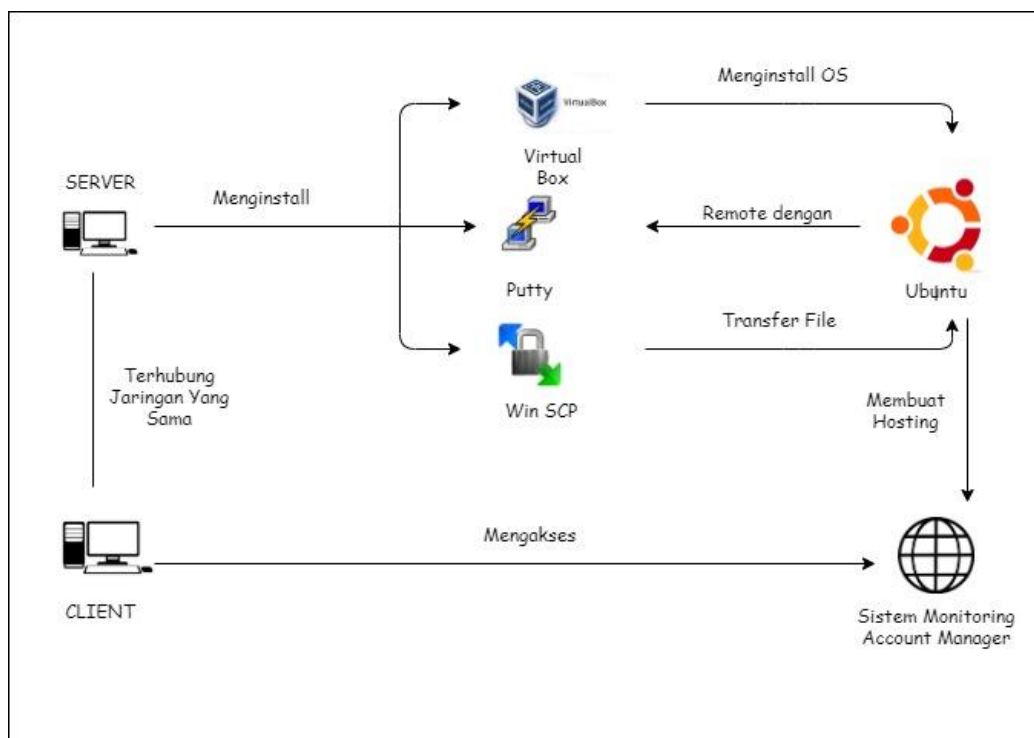
2.2 Rancangan Arsitektur Cloud Computing

Rancangan akhir pada proyek ini digunakan bentuk rancangan arsitektur *Infrastructure as a service* (IaaS) dimana hardware komputer MSI sebagai layer utama di bagian bawah yang melakukan virtualisasi dengan *Virtual Box*, kemudian di atasnya terdapat *Platform as a Service* (PaaS) dimana sistem operasi Ubuntu Server dijalankan. Di atasnya lagi terdapat *Software as a Service* (SaaS) dimana Sistem Monitoring Account Manager berjalan. Ilustrasi mengenai rancangan arsitektur tersebut dapat dilihat pada **Gambar 2.1** berikut ini :



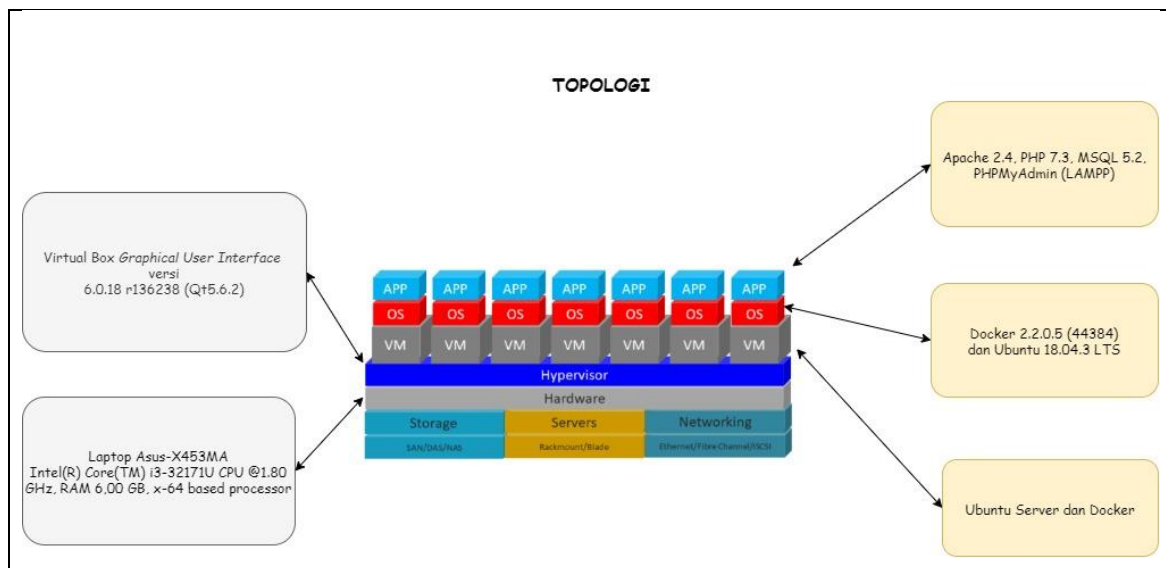
Gambar 2.1 Ilustrasi rancangan arsitektur proyek

Dalam pembuatan proyek untuk membuat *web hosting*, komputer yang digunakan sebagai *server* memakai sistem operasi *Windows 10* di mana akan dilakukan instalasi *virtual machine*, seperti *Virtual Box*, *puTTY* untuk melakukan *remote* dari *Virtual Box*, dan *WinSCP* untuk membuat transfer berkas yang berisi *source code* Sistem Monitoring Account Manager agar *website* dapat di *hosting*. Setelah semua konfigurasi dilakukan, *computer client* diharuskan terkoneksi dengan jaringan yang sama agar dapat mengakses *website* yang telah dibuat. Ilustrasi mengenai topologi yang digunakan bias dilihat pada **Gambar 2.2** berikut:



Gambar 2.2 Ilustrasi rancangan arsitektur proyek

Sedangkan untuk topologi *layer by layer cloud computing* yang digunakan dapat dilihat pada gambar berikut



Gambar 2.3 Topologi untuk proyek akhir

2.3 Parameter dan Konfigurasi

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

```
$ sudo ufw allow in "Apache Full"
```

Keterangan: Untuk mengatur Firewall agar port 80 dan port 443 diizinkan oleh sistem Ubuntu.

Modul 2.1 Parameter instalasi *Apache*

Parameter yang digunakan untuk instalasi *MySQL* dapat dilihat pada penjelasan

Modul 2.2 berikut ini:

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

Keterangan:

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- apt : merupakan package manager pada Ubuntu
- install : parameter pada apt untuk mengeksekusi perintah instalasi paket
- mysql-server : nama paket untuk MySQL

```
$ sudo mysql_secure_installation.
```

Keterangan: Untuk mengatur keamanan pada MySQL contohnya username dan password

Modul 2.2 Parameter instalasi *MySQL*

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:

- 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
- php libapache2-mod-php php-mysql : nama paket aplikasi untuk PHP

```
$sudo nano /var/www/html/info.php
```

Keterangan:

Untuk membuat file baru dengan menggunakan nano dengan nama info.php yang berada pada direktori /var/www/html

Modul 2.3 Parameter instalasi *PHP*

Parameter yang digunakan untuk instalasi *PHPMyAdmin* dapat dilihat pada penjelasan **Modul 2.4** berikut ini:

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

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
- phpMyAdmin php-mbstring php-gettext: nama paket untuk PHPMyAdmin

```
$ sudo mysql -u root
```

Keterangan:

Untuk masuk ke MySQL sebagai user root.

```
mysql> UPDATE mysql.user SET plugin = 'mysql_native_password',  
authentication_string = PASSWORD('mkfs.vfat') WHERE User = 'root';
```

Keterangan:

Untuk mengubah password dari user 'root' menjadi 'mkfs.vfat'

```
mysql> FLUSH PRIVILEGES;
```

Keterangan:

Untuk merefresh akun yang terkoneksi dengan phpmyadmin

```
$ sudo chown rahmatfhrezha /var/www/html
```

Keterangan:

Untuk memberikan akses ke user rahmatfhrezha agar dapat mengubah/menambah/menghapus berkas pada direktori var/www/html

Modul 2.4 Parameter instalasi *PHPMYAdmin*

Parameter yang digunakan untuk instalasi *PHPMYAdmin* dapat dilihat pada penjelasan **Modul 2.5** berikut ini:

```
$ sudo apt update
```

Keterangan: Untuk mengupdate package yang terinstall di Ubuntu.

```
$ sudo apt install apt-transport-https ca-certificates curl
software-properties-common
```

Keterangan: Sebelum install Docker, install package yang diperlukan untuk menginstall dan menggunakan Docker, yaitu package apt-transport-https ca-certificates, curl, dan software-properties-common.

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

Keterangan:

Untuk menambahkan kunci GPG dari repository Docker ke sistem.

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- curl -fsSL : merupakan perintah untuk mengunduh suatu file berdasarkan link yang diberikan
- apt-key add - : merupakan perintah untuk menambahkan kunci GPG ke sistem

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

Keterangan:

Untuk menambahkan repository Docker ke daftar sumber package APT (package manager Ubuntu).

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- add-apt-repository : merupakan perintah untuk menambahkan repository baru ke daftar sumber package APT (package manager Ubuntu)

- "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable" : merupakan sumber package Docker dengan acuan deb adalah jenis package, [arch=amd64] adalah jenis arsitektur, https://download.docker.com/linux/ubuntu adalah link sumber package Docker, bionic adalah versi Ubuntu yang digunakan, dan stable adalah versi rilis yang ingin digunakan.

```
$ sudo apt update
```

Keterangan: Untuk melakukan update package yang terinstall di Ubuntu.

```
$ sudo apt install docker-ce
```

Keterangan: Untuk mengunduh dan menginstall Docker.

```
$ sudo systemctl status docker
```

Keterangan:

Untuk melihat status dari Docker.

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- systemctl : merupakan system manager pada Ubuntu
- status : parameter tambahan pada systemctl untuk melihat status suatu service
- docker : nama service

Modul 2.5 Parameter instalasi *Docker*

Parameter yang digunakan untuk konfigurasi *Docker* tanpa menggunakan perintah `sudo` di awal dapat dilihat pada penjelasan **Modul 2.6** berikut ini:

```
$ sudo usermod -aG docker ${USER}
```

Keterangan:

Untuk menambahkan user aktif saat ini ke group docker

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- usermod : merupakan user manager pada Ubuntu
- -aG : parameter tambahan pada usermod untuk menambahkan user ke suatu group
- docker : nama group
- \${USER} : untuk memanggil user yang aktif saat ini

```
$ su - ${USER}
```

Keterangan:

Untuk refresh session user (agar konfigurasi di atas bisa langsung aktif tanpa harus login ulang).

Modul 2.6 Konfigurasi *Docker* tanpa perintah `sudo`

Parameter yang digunakan untuk instalasi *Docker Compose* dapat dilihat pada penjelasan **Modul 2.7** berikut ini:

```
$ sudo curl -L
https://github.com/docker/compose/releases/download/1.21.2/dockercompos
e-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose
```

Keterangan:

Untuk mengunduh Docker Compose versi 1.21.2.

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- curl -L : merupakan perintah untuk mengunduh suatu file berdasarkan link yang diberikan
- https://github.com/docker/compose/releases/download/1.21.2/docker-compose-`uname -s`-`uname -m` : link Docker Compose versi 1.21.2
- uname -s : perintah untuk mengoutputkan nama kernel
- uname -m : perintah untuk mengoutputkan nama mesin (hardware)
- -o /usr/local/bin/docker-compose : parameter tambahan pada curl untuk mengarahkan file yang diunduh ke direktori yang diinginkan

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

Keterangan:

Untuk mengubah status/permission file docker-compose menjadi executeable(bisa dieksekusi).

- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- chmod +x : perintah untuk menjadikan suatu file menjadi executeable.
- /usr/local/bin/docker-compose : lokasi file docker compose

```
$ docker-compose --version
```

Keterangan:

Untuk melihat versi docker compose yang terinstal.

- docker-compose : perintah untuk menjalankan docker-compose -
- version : parameter dari docker-compose untuk menampilkan versi docker compose yang terpasang

Modul 2.7 Parameter instalasi *Docker Compose*

Parameter yang digunakan untuk konfigurasi dari *php.Dockerfile* dapat dilihat pada penjelasan **Modul 2.8** berikut ini:

```
$ nano php.Dockerfile
```

Keterangan:

Untuk mengedit file (text editor dalam terminal).

- nano : perintah untuk menjalan text editor dalam terminal linux (bawaan dari Ubuntu)
- php.Dockerfile : nama file yang diinginkan diedit

```
# Isi dari php.Dockerfile
FROM php:7.4.3-apache
RUN docker-php-ext-install mysqli pdo pdo_mysql
Keterangan:
Untuk menginstall extensions php yang diperlukan, yaitu mysqli, pdo,
dan pdo_mysql.
- FROM php:7.4.3-apache : lokasi Dockerfile yang akan dituju /
dieksekusi
- RUN docker-php-ext-install : perintah untuk menjalankan instalasi
extensions php
- mysqli pdo pdo_mysql : nama extensions php
```

Modul 2.8 Parameter konfigurasi *php.Dockerfile*

Parameter yang digunakan untuk konfigurasi *docker-compose.yaml* dapat dilihat pada penjelasan **Modul 2.9** berikut ini:

```
$ nano docker-compose.yaml
Keterangan:
Untuk mengedit file (text editor dalam terminal).
- nano : perintah untuk menjalan text editor dalam terminal linux
(bawaan dari Ubuntu)
- docker-compose.yaml : nama file yang diingin diedit
# Isi dari docker-compose.yaml
version: "3" services:
web-server:
  build:
    dockerfile: php.Dockerfile
    context: .
  restart: always
  volumes:
    - "./aplikasimarketing/:/var/www/html/"
  ports:
    - "8080:80"
mysql-server:
  image: mysql:8.0.19
  restart: always
  environment:
    MYSQL_DATABASE: raport_online
    MYSQL_USERNAME: root
    MYSQL_PASSWORD: root
    MYSQL_ROOT_PASSWORD: root
  volumes:
```

```

- mysql-data:/var/lib/mysql
- ./aplikasimarketing/db_mysql/marketing.sql:/dockerentrypoint-
  initdb.d/marketing.sql
ports:
- "3306:3306"
phpmyadmin:
  image: phpmyadmin/phpmyadmin:5.0.1
  restart: always
  environment:
    PMA_HOST: mysql-server
    PMA_USER: root
    PMA_PASSWORD: root
  ports:
    - "5000:80"
volumes:
  mysql-data:

```

Keterangan:

Untuk mendefinisikan service yang akan diinstall pada container Docker serta konfigurasinya.

- version : versi compose file format yang akan digunakan sesuai dengan docker engine yang terinstall
- services : bagian untuk mendefinisikan service yang ingin diinstall di docker
- web-server, mysql-server, phpmyadmin : nama service yang ingin diinstall
- build : perintah bahwa service akan diinstall sesuai perintah yang ada di dalam build tersebut
- dockerfile : lokasi dockerfile yang ingin digunakan
- context : mengarahkan direktori untuk service ada di folder tersebut (. artinya direktori ada di folder sesuai lokasi dockercompose.yaml berada)
- restart : konfigurasi dari service untuk melakukan restart container ketika sesuatu hal yang tidak diinginkan terjadi
- volumes : perintah dalam service untuk mengarahkan serta mengcopy isi folder sumber ke direktori yang dituju
- ports : perintah dalam service untuk mendefinisikan port yang ingin dibuka/digunakan
- image : konfigurasi untuk memilih image / installer / package dari repository Docker yang ingin digunakan dan diinstall
- environment : perintah pada service untuk mengkonfigurasi service itu sendiri sesuai environment yang berlaku

- MYSQL_DATABASE : mendefinsikan environment mysql untuk nama database yang akan digunakan
- MYSQL_USERNAME : mendefinsikan environment mysql untuk nama username yang akan digunakan
- MYSQL_PASSWORD : mendefinsikan environment mysql untuk password mysql yang akan digunakan
- MYSQL_ROOT_PASSWORD : mendefinsikan environment mysql untuk password dari root mysql yang akan digunakan
- PMA_HOST : mendefinsikan environment phpmyadmin untuk nama host yang akan digunakan
- PMA_USER : mendefinsikan environment phpmyadmin untuk nama username yang akan digunakan
- PMA_PASSWORD : mendefinsikan environment phpmyadmin untuk password akun phpmyadmin yang akan digunakan

Modul 2.9 Parameter konfigurasi *docker-compose.yml*

Parameter yang digunakan untuk konfigurasi dari koneksi *.php* dapat dilihat pada penjelasan **Modul 2.10** berikut ini:

```
$db['default'] = array(
    'dsn' => '',
    'hostname' => 'mysql-server',
    'username' => 'root',
    'password' => 'mkfs.vfat',
    'database' => 'marketing',
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);
```

Keterangan :

Untuk menginisiasasi/mendefinisikan koneksi ke database mysql sesuai konfigurasi dari docker-compose.yaml pada Modul 2.9.

Modul 2.10 Parameter konfigurasi koneksi.php

Parameter yang digunakan untuk menjalankan *Docker Compose* dapat dilihat pada penjelasan **Modul 2.11** berikut ini:

```
$ cd docker-project
```

Keterangan :

Untuk berpindah direktori ke direktori yang terdapat dockercompose.yaml yang akan dijalankan.

- cd : perintah untuk pindah direktori
- docker-project : tujuan direktori dengan nama docker-project


```
$ docker-compose up -d
```

Keterangan :

Untuk membaca isi konfigurasi docker-compose.yaml dan membuat container docker sesuai konfigurasinya.

- docker-compose : perintah untuk menjalankan docker-compose
- up : perintah dari docker-compose untuk membaca dockercompose.yaml dan menjalankan docker
- -d : parameter dari docker-compose untuk menjalankan containers di background

Modul 2.11 Parameter konfigurasi menjalankan *Docker Compose*

Parameter yang digunakan untuk melihat *containers Docker* yang sudah terpasang sebelumnya pada **Modul 2.11** dapat dilihat pada penjelasan Modul 2.12 berikut ini:

```
$ docker ps -a
```

Keterangan :

Untuk melihat containers Docker yang terpasang.

- docker : perintah untuk menjalankan docker
- ps : perintah dari docker untuk menampilkan containers docker yang aktif
- -a : parameter dari docker untuk menampilkan seluruh containers docker baik yang aktif maupun tidak aktif

Modul 2.11 Parameter melihat *containers Docker* yang terpasang

Parameter yang digunakan untuk melihat *ip address* dapat dilihat pada penjelasan **Modul 2.13** berikut ini:

```
$ ip addr
```

Keterangan : Untuk melihat ip address.

- ip : perintah untuk menjalankan ip yang akan menampilkan seluruh interface jaringan yang ada beserta konfigurasinya seperti ip address
- addr : perintah dari ip untuk menampilkan ip address pada setiap interface jaringan yang ada

Modul 2.13 Parameter melihat *ip address*

Parameter yang digunakan untuk menghentikan *containers Docker* yang aktif sebelumnya pada **Modul 2.11** dapat dilihat pada penjelasan **Modul 2.14** berikut ini:

```
$ docker-compose down
```

Keterangan :

Untuk menghentikan containers docker yang aktif.

- docker-compose : perintah untuk menjalankan docker-compose
- up : perintah dari docker-compose untuk menghentikan containers docker yang aktif

Modul 2.14 Parameter untuk menghentikan *containers Docker* yang aktif

Parameter yang digunakan untuk meng-clone repository github dapat dilihat pada penjelasan **Modul 2.15** berikut ini:

```
$ git clone https://github.com/rzumarli/Docker-Project-TCC-E_3.git
```

Keterangan :

Untuk mengclone repository github.

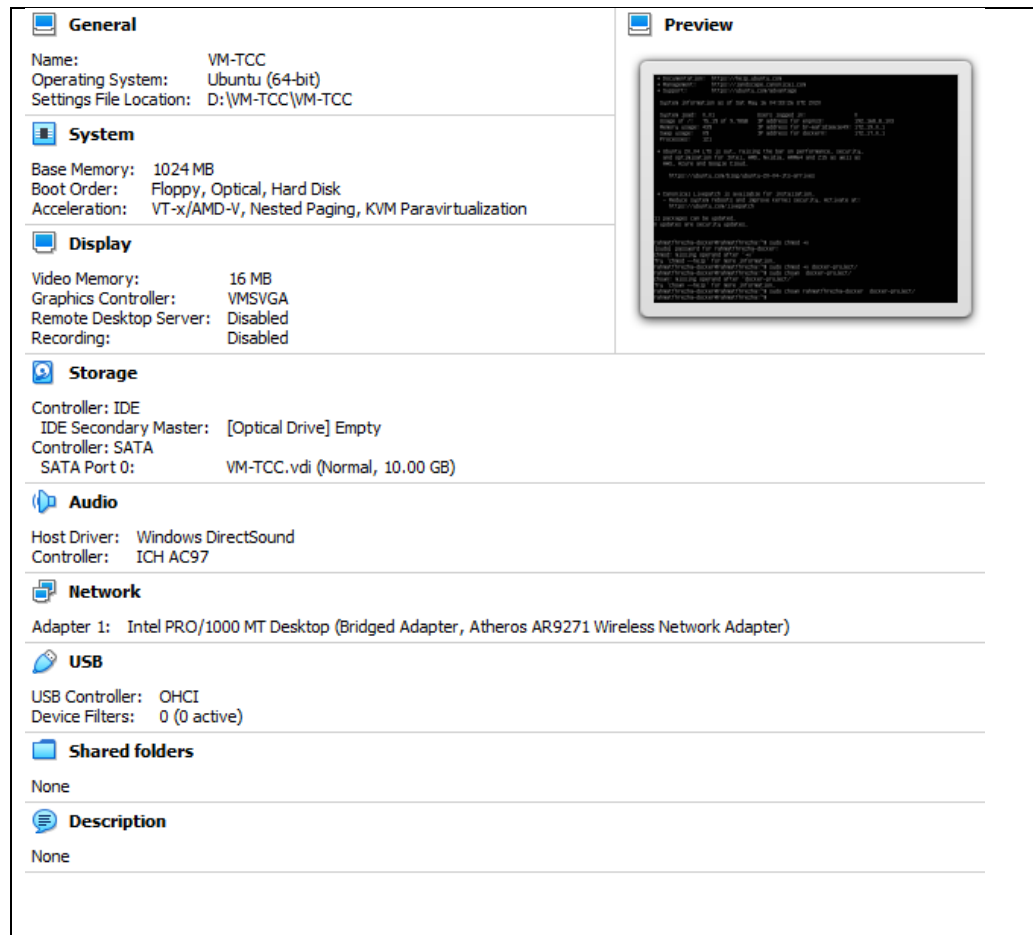
- git : perintah untuk menjalankan git
- clone : perintah dari git untuk mengunduh repository github berdasarkan link yang diberi
- https: https://github.com/rzumarli/Docker-Project-TCC-E_3.git
: link repository github

Modul 2.15 Parameter untuk menghentikan *containers Docker* yang aktif

2.4 Tahap Implementasi

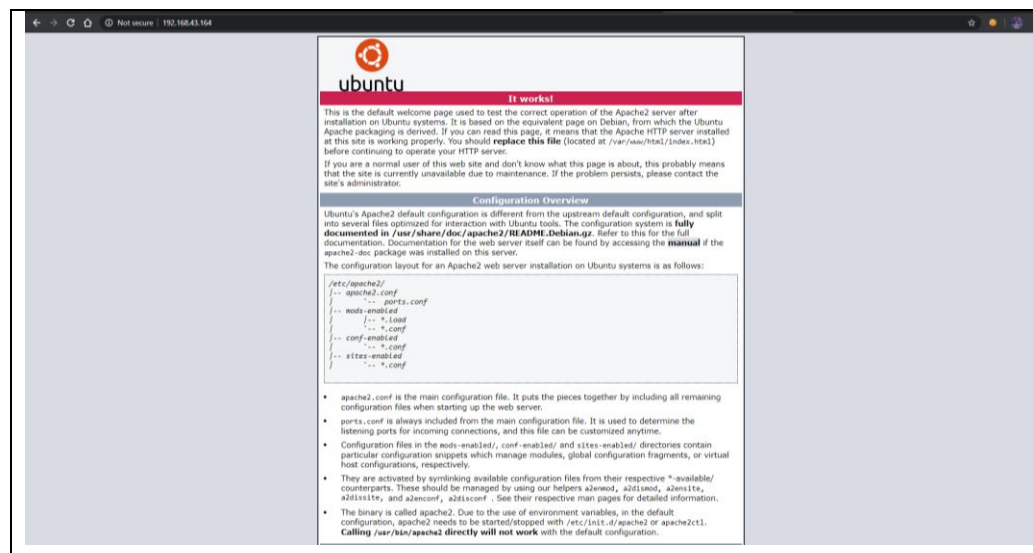
Untuk mengimplementasikan “Sistem Monitoring *Account Manager* Menggunakan Ubuntu *LAMPP* dan Proses Pembuatan *Dockerfile*-nya” yang berbasiskan konsep *cloud computing*, maka tahapan yang dilakukan adalah sebagai berikut:

- a. Tahapan awal pada pembuatan *Virtual Machine* dengan Virtual Box digunakan opsi konfigurasi *Customize* seperti pada **Gambar 2.3** berikut ini :



Gambar 2.4 Tampilan hasil pembuatan VM pada *Virtual Box*

- b. Hasil Implementasi dari **Modul 2.1** tentang cara instalasi *Apache2* dapat dilihat pada **Gambar 2.5** berikut ini:



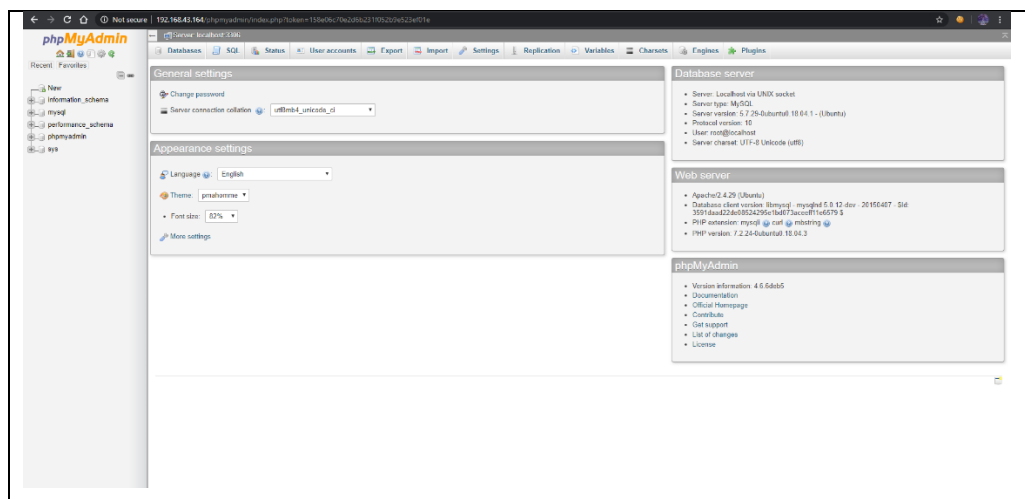
Gambar 2.5 Tampilan hasil instalasi *Apache2*

- c. Hasil implementasi dari **Modul 2.3** tentang cara instalasi *PHP* dapat dilakukan pada **Gambar 2.6** berikut ini:



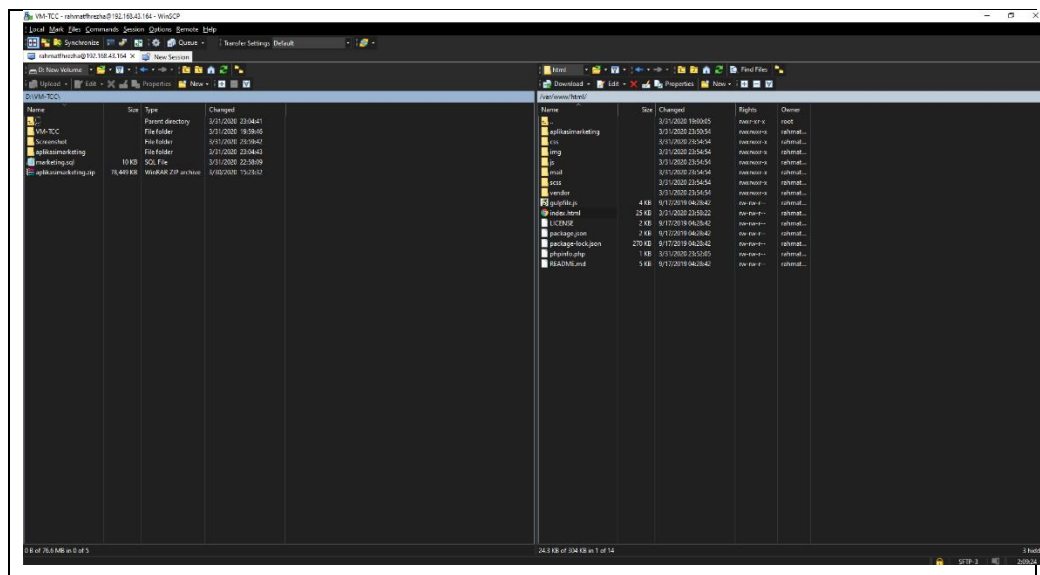
Gambar 2.6 Tampilan hasil instalasi *PHP*

- d. Hasil implementasi dari **Modul 2.4** tentang cara instalasi *PHPMysqlAdmin* dapat dilakukan pada **Gambar 2.7** berikut ini:

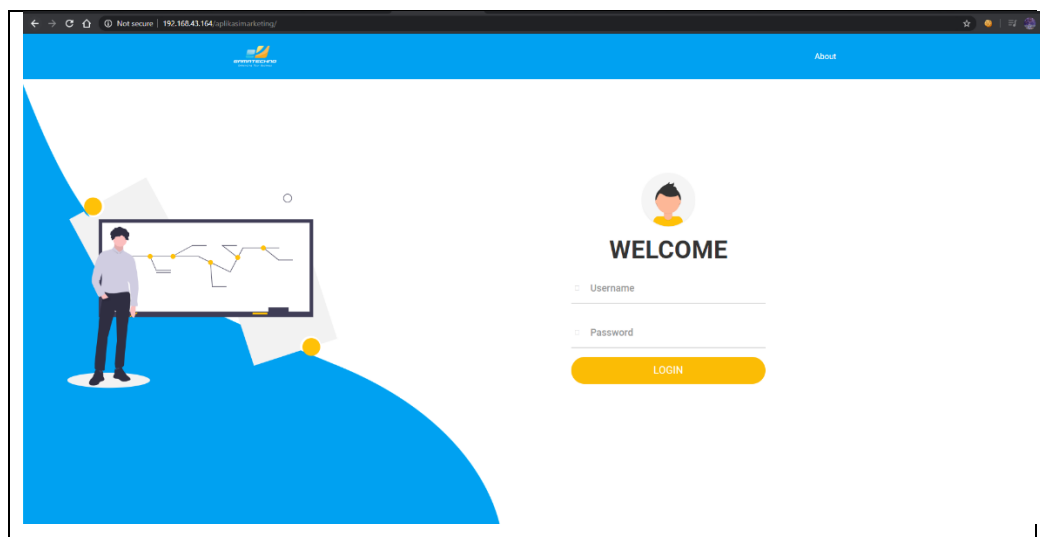


Gambar 2.7 Tampilan hasil instalasi *PHPMysqlAdmin*

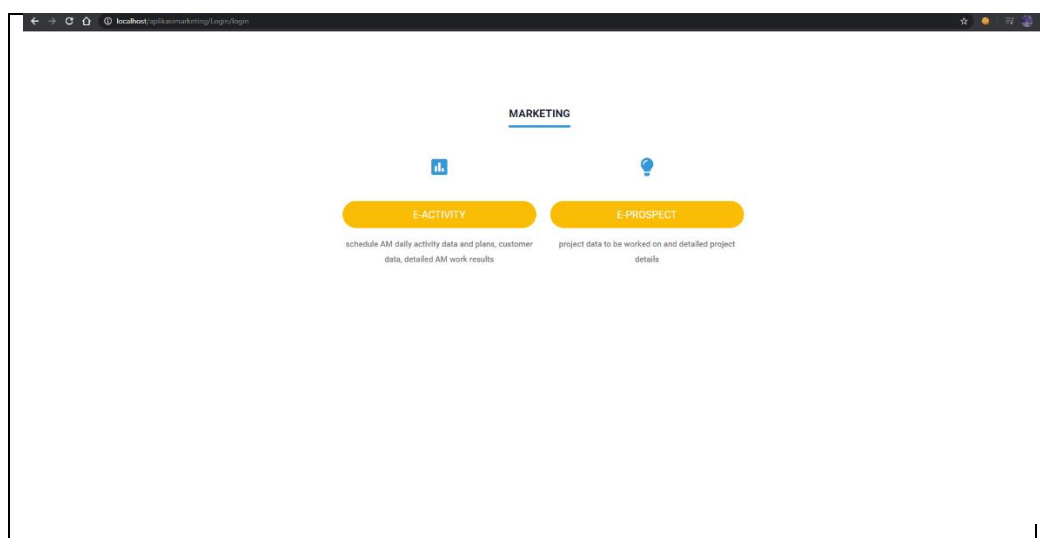
- e. Setelah semua proses instalasi LAMPP, selanjutnya adalah memindahkan *source code* ke server dengan menggunakan *WinSCP*. Pemindahan tersebut dengan cara melakukan *drag and drop* folder yang berisi *source code* ke direktori `/var/www/html`. Untuk hasil pemindahan *source code* dapat dilihat pada **Gambar 2.8** berikut ini:



Gambar 2.8 Tampilan hasil pemindaian *source code*



Gambar 2.9 Tampilan hasil *source code website*

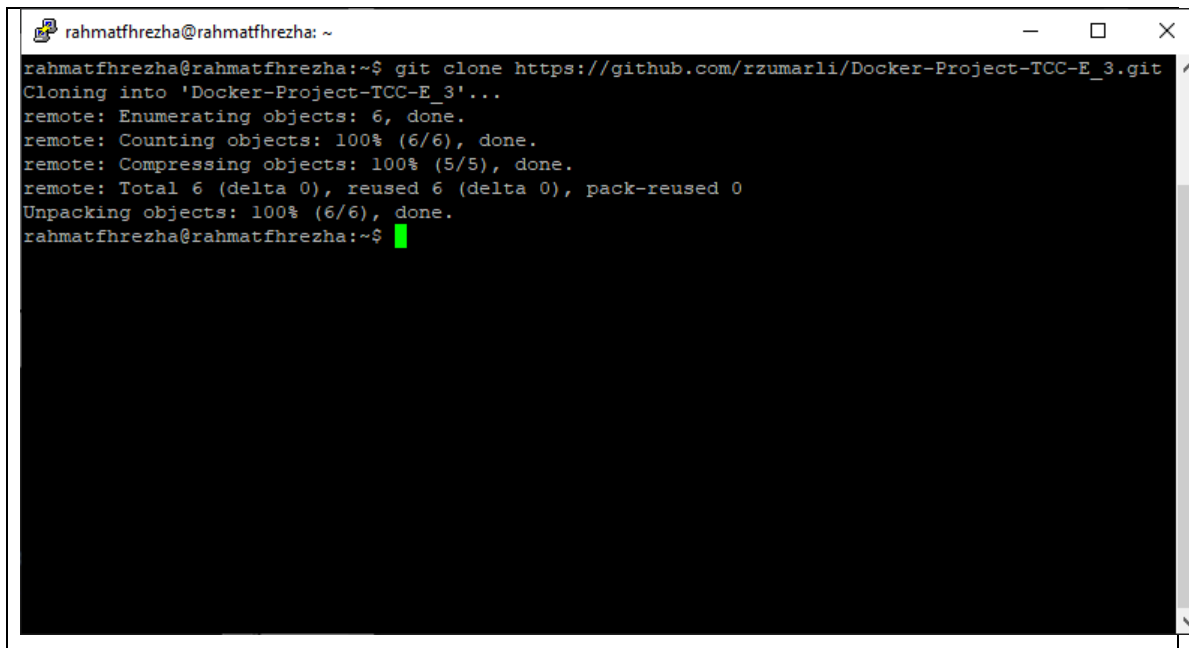


Gambar 2.10 Tampilan hasil *source code* didalam LAMPP

2.5 Hasil Implementasi

Untuk melihat hasil implementasi pembuatan *Docker* file ke dalam sistem, maka akan dilakukan *testing* dengan menggunakan *OS* Ubuntu dan *website* dari program sistem *monitoring* AM tanpa perlu konfigurasi, jadi langsung dapat diakses. Hasil dari pembuatan *Docker* file dapat memudahkan penginstallan aplikasi ke dalam sistem tanpa melakukan konfigurasi lagi. Berikut tahapan pengimplementasian *Dockerfile* :

a. Mengclone *repository* sistem monitoring AM di *github* sesuai dengan **Modul 2.15**.

A screenshot of a terminal window with a black background and white text. The window title bar shows the user 'rahmatfhrezha@rahmatfhrezha' and the home directory '~'. The terminal output shows the command 'git clone https://github.com/rzumarli/Docker-Project-TCC-E_3.git' being executed. The output lines are: 'Cloning into 'Docker-Project-TCC-E_3'...', 'remote: Enumerating objects: 6, done.', 'remote: Counting objects: 100% (6/6), done.', 'remote: Compressing objects: 100% (5/5), done.', 'remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0', and 'Unpacking objects: 100% (6/6), done.'. The prompt 'rahmatfhrezha@rahmatfhrezha:~\$' is visible at the end of the output.

```
rahmatfhrezha@rahmatfhrezha:~$ git clone https://github.com/rzumarli/Docker-Project-TCC-E_3.git
Cloning into 'Docker-Project-TCC-E_3'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 6 (delta 0), reused 6 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), done.
rahmatfhrezha@rahmatfhrezha:~$
```

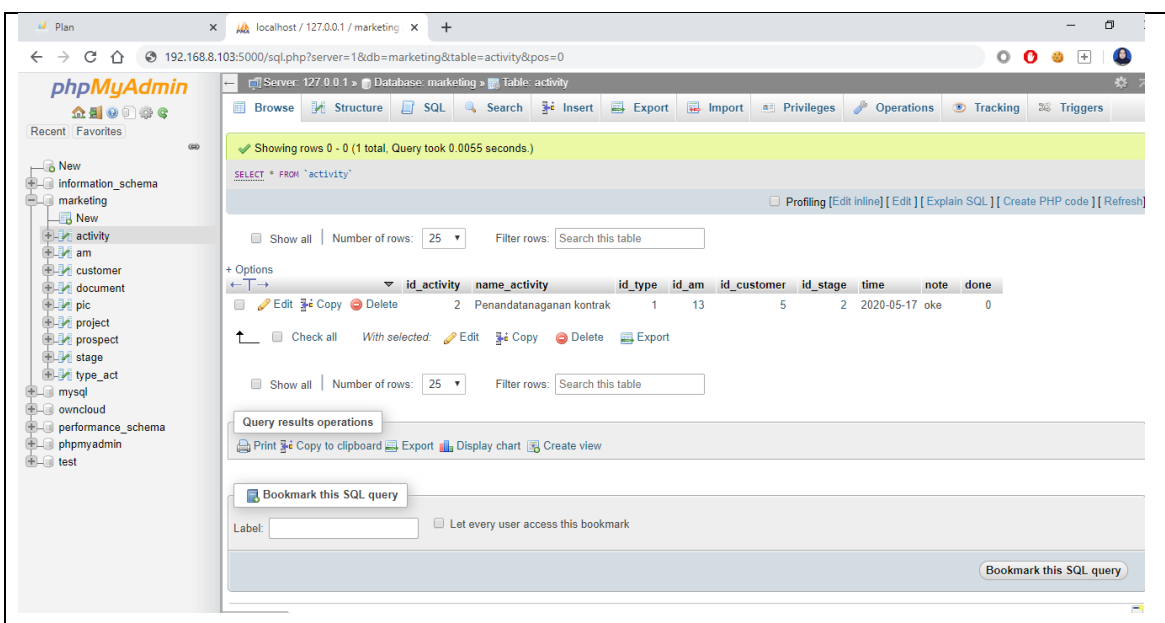
Gambar 2.11 Tampilan hasil *clone repository github*

- b. *Install docker* dan *docker-compose* sesuai dengan **Modul 2.5**, **Modul 2.6**, dan **Modul 2.7**.
- c. Pindah *direktori* dan *membuild docker* sesuai dengan **Modul 2.11**.

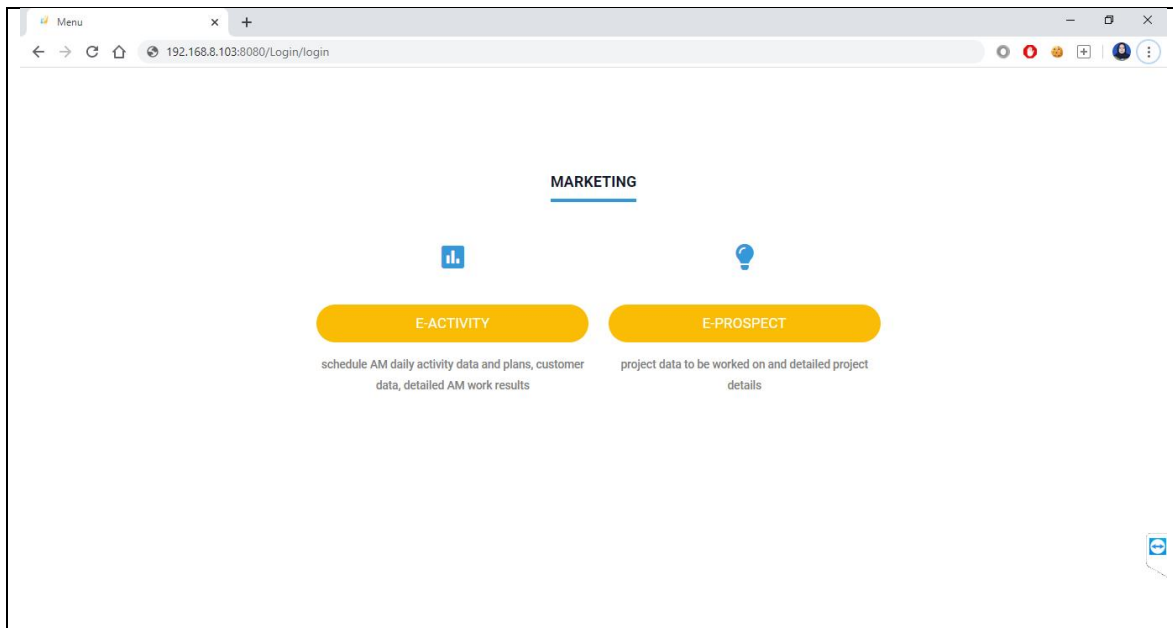
```
rahmatfhrezha@rahmatfhrezha: ~
rahmatfhrezha@rahmatfhrezha:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel stat
    1000
    link/ether 08:00:27:4a:7c:fe brd ff:ff:ff:ff:ff:ff
    inet 192.168.8.103/24 brd 192.168.8.255 scope global dynamic enp0s3
        valid_lft 85738sec preferred_lft 85738sec
    inet6 fe80::a00:27ff:fe4a:7cfe/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue st
    link/ether 02:42:fa:68:2c:31 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
4: br-6bea16ac8a75: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqu
    ult
    link/ether 02:42:ff:55:fb:79 brd ff:ff:ff:ff:ff:ff
    inet 172.24.0.1/16 brd 172.24.255.255 scope global br-6bea16ac8a75
        valid_lft forever preferred_lft forever
    inet6 fe80::42:ffff:fe55:fb79/64 scope link
```

Gambar 2.12 Tampilan hasil *build docker-compose.yaml*

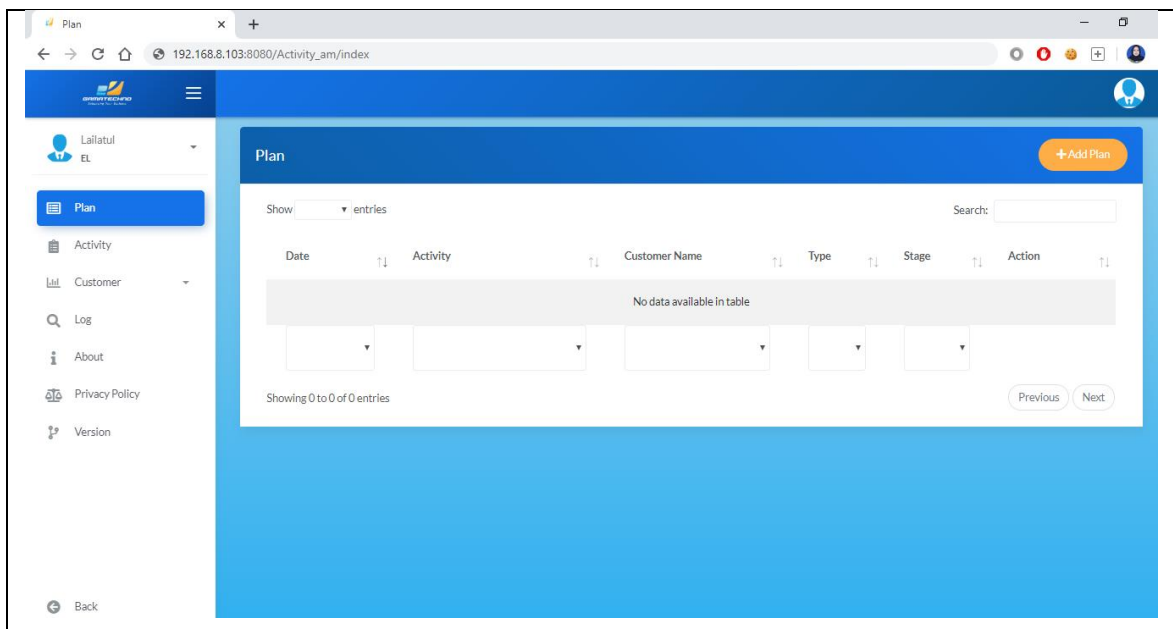
d. Cek *ip address* sesuai **Modul 2.13** dan buka *website phpmyadmin* dan *website* sistem monitoring AM



Gambar 2.13 Tampilan *website phpmyadmin*



Gambar 2.14 Tampilan awal sistem monitoring AM



Gambar 2.15 Tampilan *website* sistem monitoring AM setelah *login*

The screenshot shows the 'Add Plan' modal form. The form fields are as follows:

Field	Value
Name Activity	Penandatangan kontrak
Stage	Prospecting Progress
Customer	Diskominfo Jogja
Date	17-05-2020
Type	Call
Noted	oke

Buttons: Add (blue), Close (red).

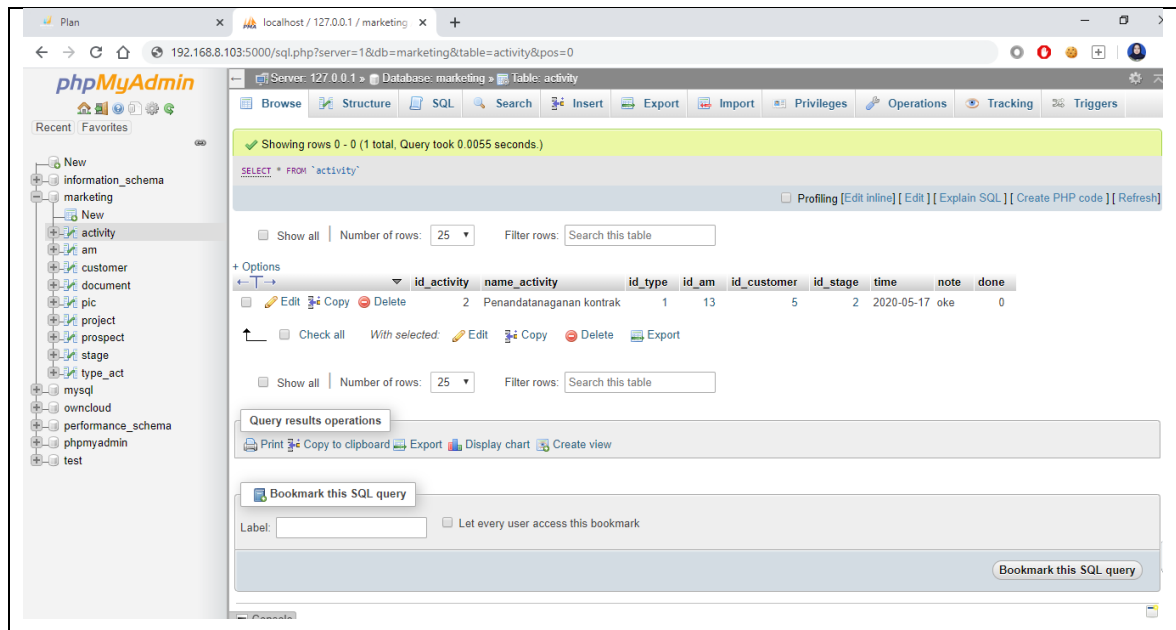
Gambar 2.16 Tampilan *website* sistem monitoring AM *input* data

The screenshot shows the 'Plan' table with the following data:

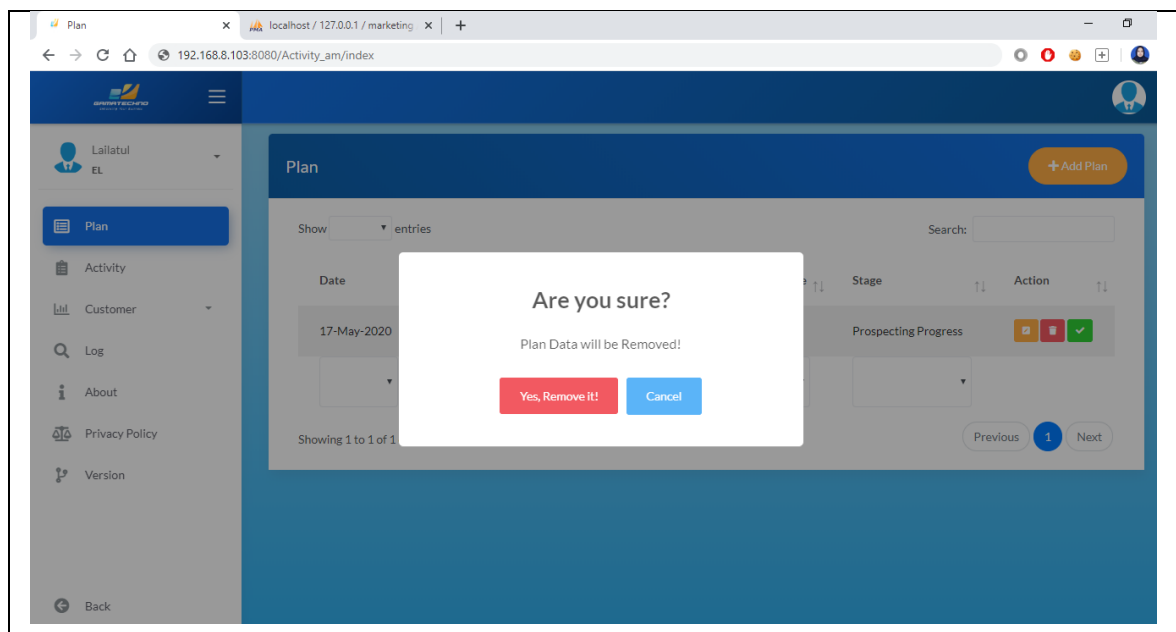
Date	Activity	Customer Name	Type	Stage	Action
17-May-2020	Penandatangan kontrak	Diskominfo Jogja	Call	Prospecting Progress	[Edit] [Delete] [Add]

Showing 1 to 1 of 1 entries. Navigation: Previous, 1, Next.

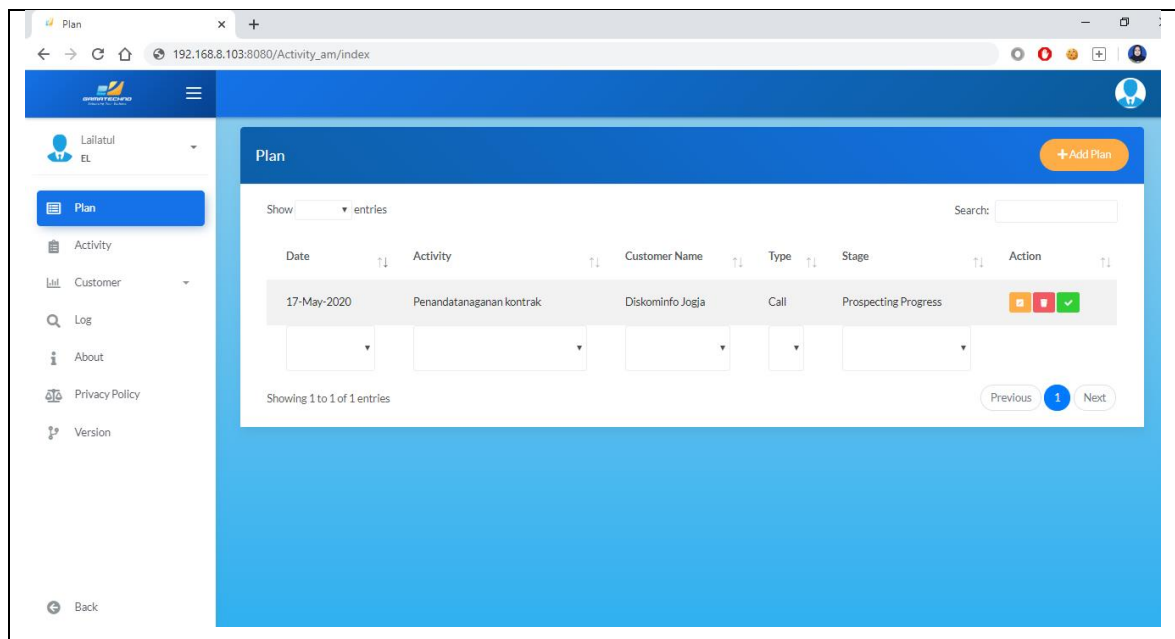
Gambar 2.17 Tampilan *website* sistem monitoring AM setelah data di *input*



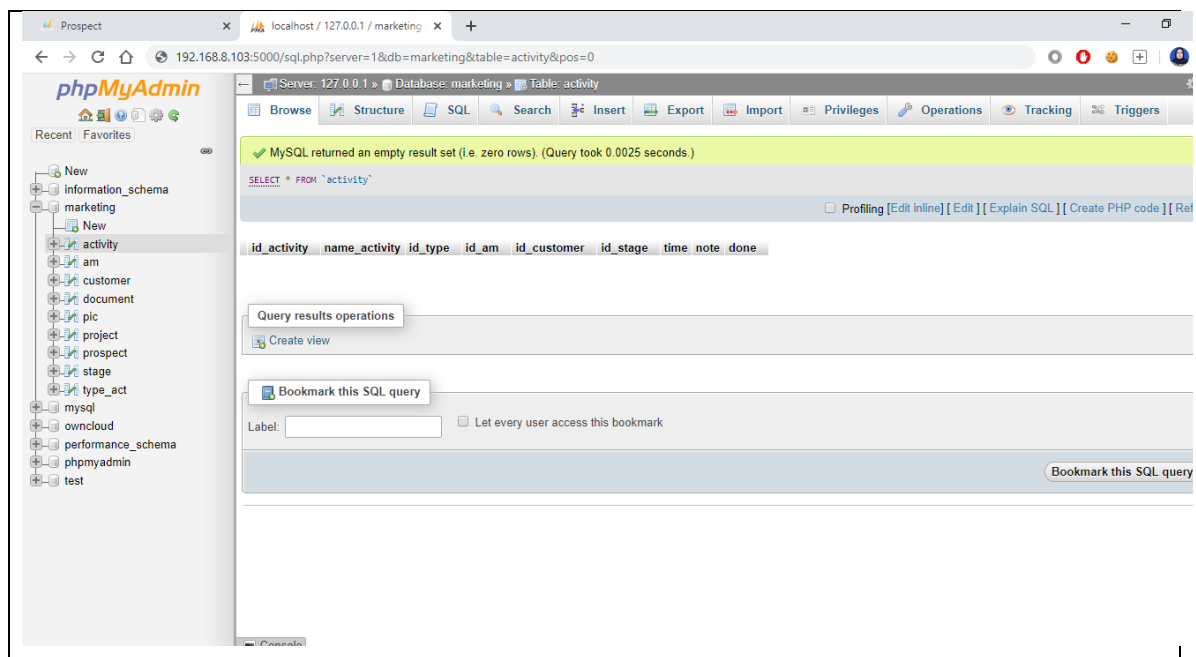
Gambar 2.18 Tampilan PHPMyAdmin sistem monitoring AM setelah data di *input*



Gambar 2.19 Tampilan *website* sistem monitoring AM konfirmasi *delete* data



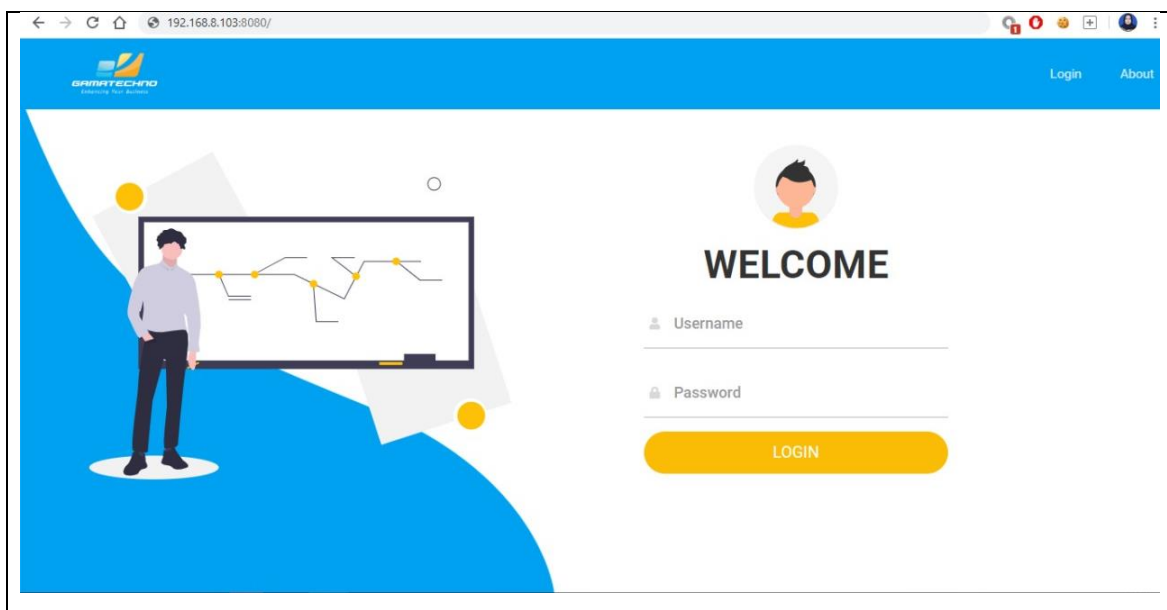
Gambar 2.20 Tampilan *website* sistem monitoring AM setelah data di *delete*



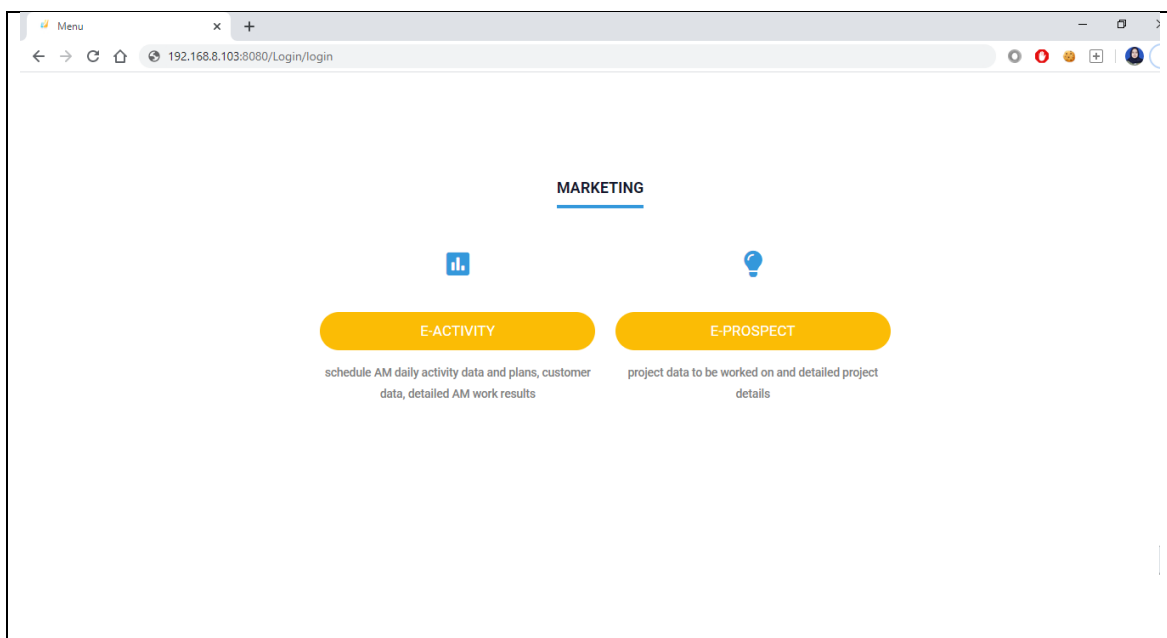
Gambar 2.21 Tampilan PHPMYAdmin sistem monitoring AM setelah data di *delete*

2.6 Pengujian Singkat

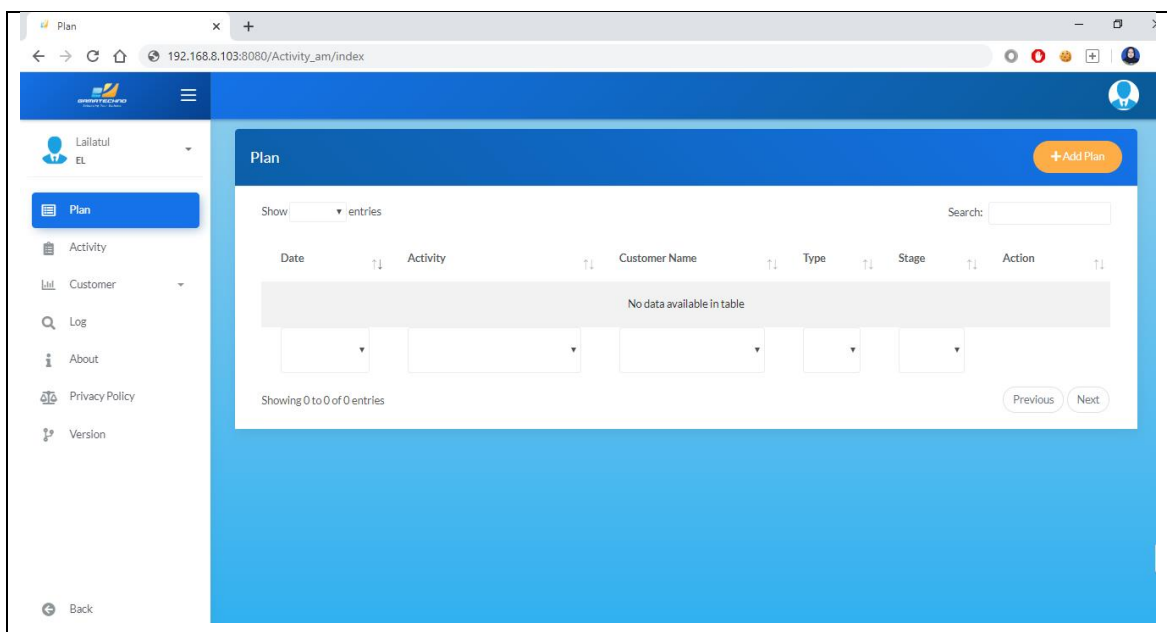
Dengan adanya Sistem Monitoring Account Manager dapat dilihat pada **Gambar 2.22**, **Gambar 2.23**, dan **Gambar 2.24** berikut



Gambar 2.22 Tampilan *login website Sistem Monitoring Account Manager*

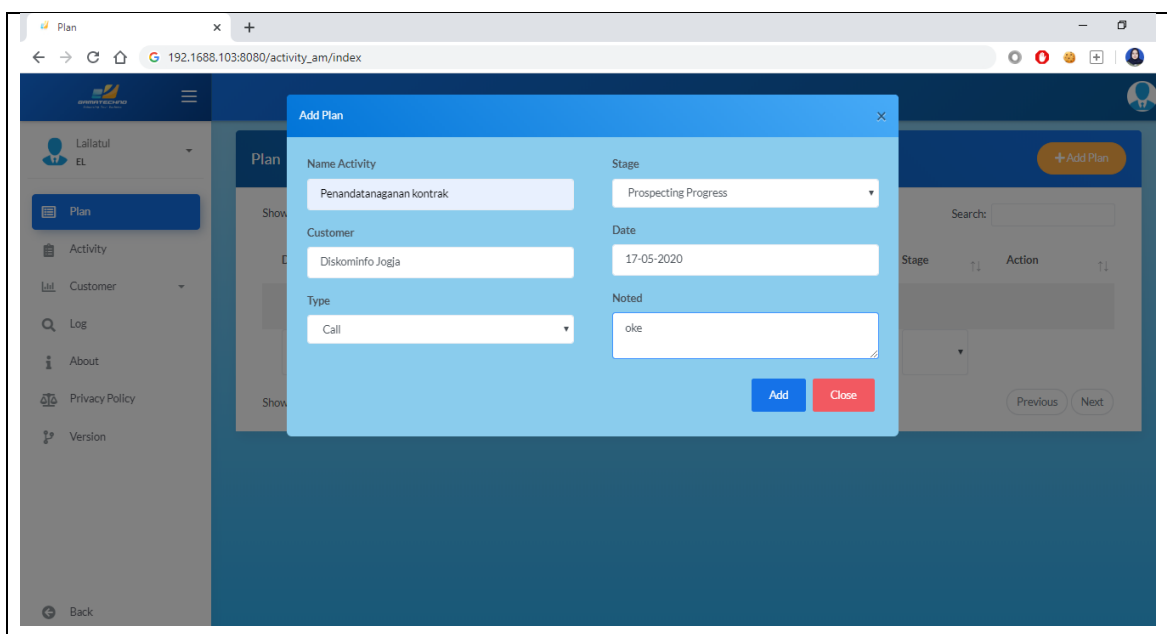


Gambar 2.23 Tampilan awal *website Sistem Monitoring Account Manager setelah login*



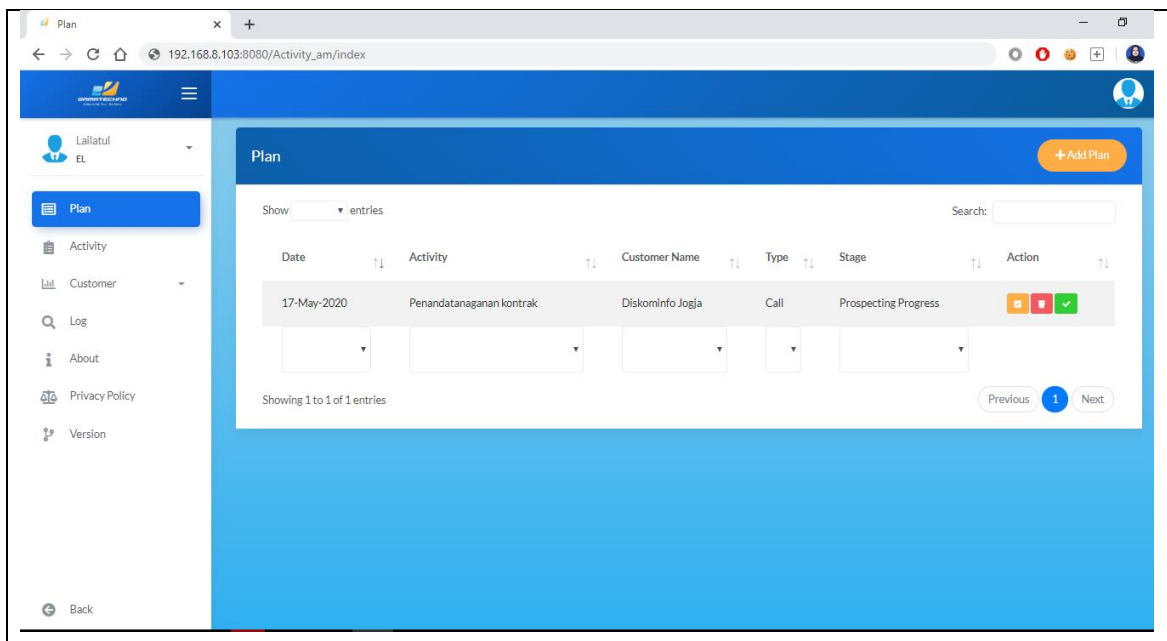
Gambar 2.24 Tampilan halaman depan *E-Activity* bagian *Plan*

Berikut ini adalah halaman awal *E-Activity* bagian *plan* yang diakses kapanpun dan dari mana saja selama ada koneksi internet. Setelah itu yang harus dilakukan adalah *input data* seperti pada **Gambar 2.25** berikut.



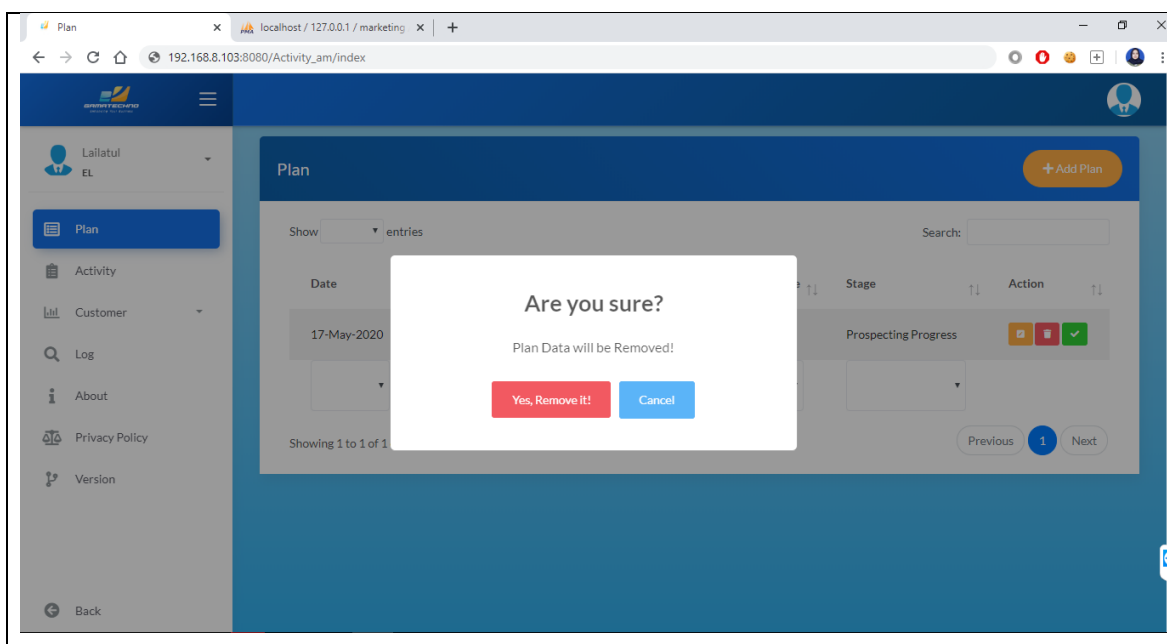
Gambar 2.25 Tampilan *input data*

Kemudian data yang telah diinput dapat adalah seperti pada **Gambar 2.26** berikut.



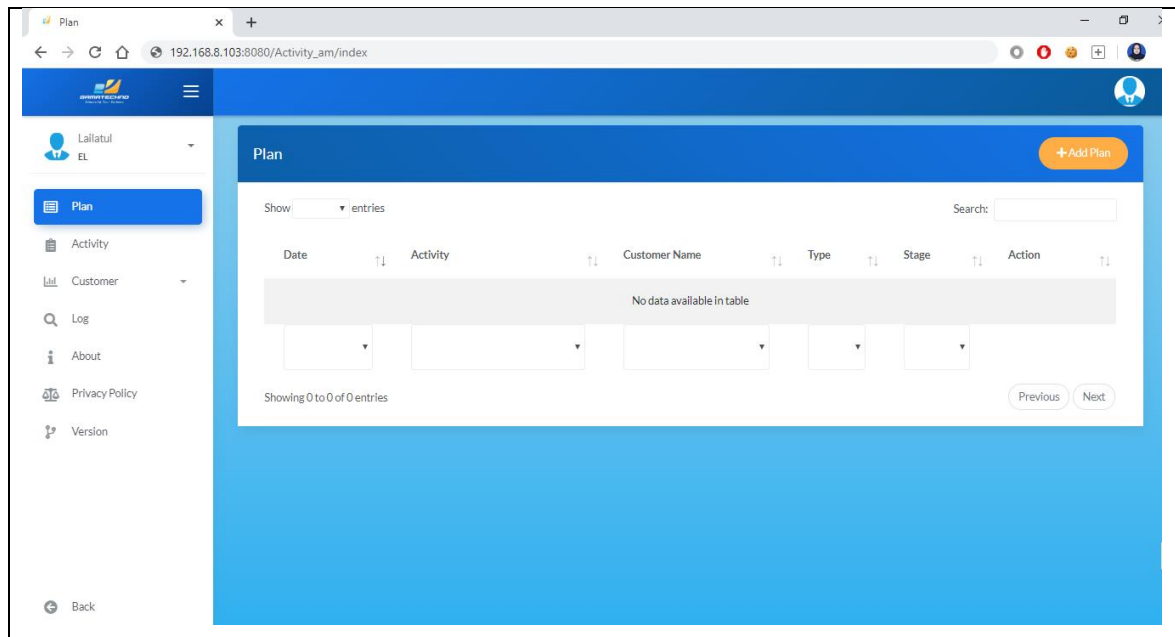
Gambar 2.26 Tampilan data setelah di *input*

Apabila ingin menghapus data maka yang dilakukan adalah dengan *click icon* yang berbentuk kotak sampah dan berwarna merah, yang dimana setelah dilakukan *click* maka akan muncul konfirmasi *delete* data seperti **Gambar 2.27** berikut.



Gambar 2.27 Tampilan konfirmasi data yang akan di *delete*

Setelah data di *delete* maka akan seperti **Gambar 2.28** berikut.



Gambar 2.28 Tampilan setelah data di *delete*

Hasil dari apa yang telah dilakukan adalah pengguna dapat melakukan *create*, *read*, *update* dan *delete* pada Sistem Monitoring Account Manager dan dapat membantu proses marketing. Kesimpulan dari percobaan tersebut bahwa Sistem Monitoring Account Manager yang dibuat mampu memberikan layanan yang sangat bermanfaat bagi perusahaan Gamatechno.

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, April, dan Mei tahun 2020:

Tabel 3.1 Agenda Pengerjaan Proyek

No.	Jenis Tugas	Waktu Pengerjaan											
		Maret				April				Mei			
		1	2	3	4	1	2	3	4	1	2	3	4
1.	Analisa Persoalan												
2.	Pembagian Tugas												
3.	Pemilihan judul proyek dan pembuatan rancangan												
4.	Install <i>Virtual Box</i> , Pembuatan Laporan Bab I												
5.	Install <i>Ubuntu</i> , <i>PHP MyAdmin</i> , <i>PHP</i> , <i>MySQL</i> , <i>Apache2</i>												
6.	Revisi Laporan Bab I												
7.	Install <i>Docker</i>												
8.	Pembuatan Laporan Bab II												
9.	Pengerjaan <i>Docker</i> dan <i>Testing</i>												
10.	Pengerjaan Laporan Bab III, IV												
11.	Pengecekan Laporan dan Proyek Akhir keseluruhan												
12.	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 <i>Cloud Computing</i>	Fhrezha Rahmat
2.	Penginstalan <i>Ubuntu</i>	Fhrezha
3.	Penginstalan <i>Package</i>	Rahmat
4.	Agenda Pengerjaan Proyek	Fhrezha
5.	Instalasi <i>LAMPP</i>	Rahmat
6.	Instalasi <i>Docker</i> dan konfigurasi	Rahmat Fhrezha
7.	Bab I	Fhrezha
8.	Bab II	Fhrezha
9.	Bab III	Rahmat
10.	Bab IV	Rahmat
11.	<i>Testing</i>	Fhrezha

BAB IV

KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan permasalahan yang telah dijabarkan dan dibahas sebelumnya pada latar belakang mengenai pembuatan layanan *cloud computing* menggunakan *OS Ubuntu* dan *Docker* telah sesuai dengan tujuan proyek akhir yang ada. Dengan demikian diperoleh hasil bahwa program dapat dibuka melalui *web browser* menggunakan jaringan internet lokal dengan lancar seperti saat ketika mengakses *web* menggunakan *hosting* lokal. Dalam pengerjaan terdapat kendala seperti kurangnya materi mengenai penggunaan *Docker* sehingga cukup menghambat proses pengerjaan proyek akhir, juga kurang efisien untuk komunikasi membahas proyek akhir karena adanya pandemik ini sehingga kurang maksimal dalam pengerjaan, dan koneksi internet yang sangat mempengaruhi proses pengerjaan proyek akhir ini. Pada bagian *testing* atau pengujian, sistem yang dijalankan dengan virtualisasi *cloud* mendapat hasil bahwa perancangan layanan *cloud computing* yang diterapkan dalam sistem monitoring AM dapat berjalan sesuai kebutuhan. Dengan rancangan virtualisasi *cloud*, sistem aplikasi tersebut dapat mengurangi beban kerja *storage* dalam kinerja proses komputasi, sehingga proses yang bekerja di dalam sistem menjadi lebih cepat. Untuk pembagian tugas proyek akhir terutama pada bab 3, dirasa sudah cukup baik karena terbantu juga dengan *timeline* yang telah ditentukan oleh asisten laboratorium. Sehingga proyek akhir *cloud computing* dapat terselesaikan dengan hasil cukup baik.

4.2 Saran

Berdasarkan hasil pengerjaan proyek dan kendala yang dialami yaitu kurangnya pemahaman lebih mengenai *Docker*, ataupun tutorial pengaplikasian *Docker* membuat pengerjaan proyek akhir ini sedikit terhambat. Namun, akan lebih baik jika terdapat tutorial mengenai materi *Docker*, khususnya dari pihak kampus (Asisten Laboratorium) mengenai pengimplementasian *Docker* untuk membantu dalam penyelesaian proyek akhir yang menggunakan *Docker*. Kemudian untuk sistem *monitoring* AM agar dikembangkan lebih baik agar fungsi nya dapat lebih bermanfaat juga menambah keamanan untuk melindungi hak akses pengguna.

DAFTAR PUSTAKA

- _____, < <https://aws.amazon.com/id/docker/> >, (____, accessed 28 April 2020)
- Development & Security, Web & Development, *Mengenal apa itu docker, definisi, fungsi, keunggulan, dan cara kerjanya*, < <https://idcloudhost.com/mengenal-apa-itu-docker-definisi-fungsi-keunggulan-dan-cara-kerjanya/> >, (6 Februari 2020, accessed 27 April 2020)
- Hogan, B. (2018, Juli 5). *How To Install and Use Docker on Ubuntu 18.04*. Diambil kembali dari DigitalOcean, LLC: <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-18-04>
- Juni Yadi, *Apa itu docker? Serta kelebihan dan kekurangan docker?*, < <https://juniyadi.id/apa-itu-docker-serta-kelebihan-dan-kekurangan-docker/> > , (28 Maret 2019, accessed 28 April 2020)
- Mobnasesemka.com, *Keunggulan docker sebagai containerized platform*, < <https://mobnasesemka.com/keunggulan-docker/> >, (24 Juli 2016, accessed 28 April 2020)
- PostMedya, *Cloud computing : pengertian, sejarah, serta manfaatnya*, < <http://www.postmedya.com/teknologi/cloud-computing-pengertian-sejarah-serta-manfaatnya/> >, (2018, accessed 29 April 2020)
- Restu Adyatma, *Memahami arsitektur cloud computing*, < <http://aboutdoublr.blogspot.com/2013/11/memahami-arsitektur-cloud-computing.html> >, (2014, accesseed 29 April 2020)

LAMPIRAN