

**PRAKTIKUM TEKNOLOGI CLOUD COMPUTING
LAPORAN PROYEK AKHIR**

**SISTEM INFORMASI PAKET WISATA MENGGUNAKAN UBUNTU LAMPP
DAN IMPLEMENTASI PENGGUNAAN DOCKER FILE PADA UBUNTU**



DISUSUN OLEH:

NAMA ANGGOTA : INDRAYANTO 123140163
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 INFORMASI PAKET WISATA MENGGUNAKAN UBUNTU LAMPP DAN IMPLEMENTASI PENGGUNAAN DOCKER FILE PADA UBUNTU

Disusun oleh :

Indrayanto

123140163

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 Warahmatullah Wabarakatuh.

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 Informasi Paket Wisata Menggunakan Ubuntu Lampp dan Implementasi Penggunaan Docker File Pada Ubuntu. Adapun laporan ini berisi tentang proyek akhir yang kami pilih dari hasil pembelajaran selama praktikum berlangsung.

Tidak lupa kami ucapkan terimakasih kepada Asisten dan 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 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. Wassalamualaikum Warahmatullah Wabarakatuh.

Yogyakarta, 3 April 2020

Penyusun

DAFTAR ISI

HALAMAN PENGESAHAN	i
KATA PENGANTAR.....	ii
DAFTAR ISI.....	iii
BAB I PENDAHULUAN.....	1
1.1. Latar Belakang Proyek Akhir	1
1.2 Tujuan Proyek Akhir	3
1.3 Manfaat Proyek Akhir	3
1.4 Tahap Penyelesaian Proyek Akhir	3
BAB II ISI DAN PEMBAHASAN	4
2.1 Komponen yang Digunakan	4
2.2 Rancangan Arsitektur <i>Cloud Computing</i>	5
2.3 Parameter dan Konfigurasi.....	6
2.4 Tahap Implementasi.....	7
2.5 Hasil Implementasi	13
2.6 Pengujian Singkat	14
BAB III JADWAL Pengerjaan dan Pembagian Tugas.....	15
3.1 Agenda Pengerjaan	15
3.2 Keterangan Pembagian Tugas	15
BAB IV KESIMPULAN DAN SARAN	16
4.1 Kesimpulan.....	16
4.2 Saran.....	16
DAFTAR PUSTAKA.....	17
LAMPIRAN	18

BAB I

PENDAHULUAN

1.1. Latar Belakang Proyek Akhir

Perkembangan aplikasi dan teknologi internet mengalami perkembangan ke arah kenyamanan dan kemudahan dalam kehidupan sehari-hari. Baik dari segi ukuran perangkat, aplikasi, space penyimpanan data, maupun kebutuhan interaksi sosial. Berkat berkembangnya teknologi internet, arsitektur komputer sekarang dapat dikembangkan menjadi *cloud computing* atau komputasi awan. Cloud computing adalah sebuah arsitektur teknologi informasi yang dimana sumber daya komputasi tersedia sebagai layanan yang dapat diakses melalui internet. Cloud computing pada dasarnya adalah menggunakan internet-based service untuk mendukung proses bisnis (Yuli Fauziah, 2013).

Cloud Computing disebut dengan Utility Computing telah mengubah pernyataan tentang penyimpanan data (informasi dan menjalankan aplikasi). Data disimpan di "Cloud" daripada di komputer individu. Cloud disebut sebagai datacenter perangkat lunak dan perangkat keras yang mendukung kebutuhan pengguna. Dengan perkembangan internet yang cepat sumber daya menjadi lebih kuat, lebih murah dan lebih banyak tersedia. Perkembangan ini menghasilkan sebuah model komputasi baru yang disebut *cloud computing*. *Cloud computing* berisi *cloud virtual* di mana pengguna informasi dan aplikasi disimpan. *Infrastructure as a Service (IaaS)*, memberikan infrastruktur komputer platform tervirtualisasi sebagai layanan tanpa membeli perangkat lunak dan server. Misalnya-IaaS penyedia termasuk Amazon EC2, GoGrid dan Flexiscale. *Platform as a Service (PaaS)*, memungkinkan pengembang aplikasi untuk menjadi tuan rumah layanan mereka. Contoh: Google Apps Engine, Amazon E2C, Microsoft Window Azure dan Force.com. *Software as a Service (SaaS)*, aplikasi itu sendiri diberikan oleh penyedia layanan. Perangkat lunak dapat digunakan sebagai layanan melalui internet internet tanpa menginstal perangkat lunak pada pengguna komputer misal Gmail, Yahoo dan Salesforce.com (Shalini Joshi, Uma Kumari, 2016).

Perkembangan aplikasi web yang sangat pesat saat ini seiring dengan perkembangan komputer dan internet. Selain itu, aplikasi berbasis web juga semakin banyak digunakan karena dapat diakses di berbagai platform komputer hanya dengan menjalankan web browser. Sehingga, kemudahan proses deployment (penyebaran) aplikasi web beserta software pendukung seperti web server, database server, dependensi dan environment lain

ke server sangat dibutuhkan. Sistem web hosting modern di dalam setiap server mengelola banyak aplikasi web. Teknologi Virtual Machine dimanfaatkan untuk menyelesaikan masalah heterogenitas (perbedaan versi library atau tools dari beberapa aplikasi web). Peningkatan jumlah aplikasi web yang harus di hosting harus diikuti dengan peningkatan kualitas ataupun kuantitas sumber daya, terlebih saat hadirnya kebutuhan high availability dari layanan web tersebut.

Teknik kontainerisasi (virtualisasi berbasis container) hadir sebagai solusi dan menjadi tren saat ini. Docker adalah salah satu software yang mengadopsi teknik kontainerisasi dan semakin banyak diterapkan di dalam lingkungan web hosting. Dalam membangun program, pengembang biasanya menjalankan virtualisasi pada server sehingga proses pembuatan program dapat berjalan pada berbagai platform maupun konfigurasi hardware. Masalah yang dihadapi dengan virtualisasi adalah perlunya menyiapkan satu sistem operasi secara utuh, termasuk berbagai aplikasi yang dibawa sistem tersebut. Bisa dibayangkan dengan banyaknya virtualisasi yang berjalan di sebuah server akan memberatkan sistem tersebut. Container digunakan untuk memberikan beberapa perubahan. Dengan container, sebuah program diikat beserta library-nya, file konfigurasi dan seluruh hal yang dibutuhkannya. Perbedaan yang sangat terlihat dibandingkan dengan virtualisasi adalah container memiliki ukuran file yang jauh lebih kecil karena tidak perlu menyiapkan sistem operasi secara penuh. Dalam hal ini, pengembang biasa menyebutnya sebagai 'lightweight' platform. Aplikasi yang berjalan menggunakan container pun jauh lebih cepat dan lebih efisien. Docker adalah salah satu platform yang dibangun berdasarkan teknologi container. Docker merupakan sebuah project open-source yang menyediakan platform terbuka untuk developer maupun sysadmin untuk dapat membangun, mengemas dan menjalankan aplikasi dimanapun sebagai sebuah wadah (container) yang ringan.

Dari beberapa acuan tersebut, maka proyek ini diharapkan mampu mengimplementasikan pengembangan aplikasi berbasis web menggunakan Docker pada sistem operasi Linux. Aplikasi web menggunakan framework codeigniter dengan database mysql. Setelah aplikasi web berfungsi dan simpan pada repository git, selanjutnya akan diimplementasikan pada Docker yang berisi paket dari Lamp.

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 Paket Wisata yang telah dibuat sebelumnya dengan menggunakan Ubuntu 16.4 dan LAMPP (Apache, PHP 7.2.1, MySQL) dan mengintegrasikannya dengan Docker.

1.3 Manfaat Proyek Akhir

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

1. Memudahkan dalam menyimpan dan mengubah data di server, dikarenakan dengan aplikasi berbasis *cloud computing* semua data termasuk file, dokumen dan aplikasi yang digunakan akan secara otomatis tersimpan di server
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 terdistribusi dikarenakan setiap sistem terhubung satu sama lain melalui *private cloud network*.

1.4 Tahap Penyelesaian Proyek Akhir

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

1. Menganalisis kebutuhan dari sistem Manajemen Paket Wisata untuk ditransformasikan ke dalam arsitektur *cloud computing* menggunakan basis IaaS/SaaS/PaaS/DBaaS dan XaaS/WaaS.
2. Mengintegrasikan penyimpanan data paket wisata dengan Sistem Manajemen Paket Wisata yang berada di Ubuntu Server.
3. Menentukan konfigurasi yang tepat untuk pengaturan Docker sehingga dapat digunakan sesuai *requirement*.
4. Merancang topologi *cloud computing* untuk mengintegrasikan dua sub sistem yang berbeda sehingga dapat digunakan secara terintegrasi.

BAB II

ISI DAN PEMBAHASAN

2.1 Komponen yang Digunakan

Untuk membangun “Sistem Manajemen Paket Wisata menggunakan Ubuntu LAMPP” 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 7.2.
2. Selain itu juga diperlukan penyimpanan basis data dengan arsitektur penyimpanan MySQL versi 5.7 sehingga dapat digunakan untuk menyimpan berbagai data tempat wisata yang dibutuhkan oleh sistem tersebut.
3. Untuk target pengguna dengan konsep *cloud computing*, maka penggunanya ialah seluruh tempat wisata di Indonesia. Tidak ada yang dapat mengakses sistem tersebut kecuali harus terhubung melalui jaringan. Sehingga diperlukan arsitektur *cloud computing* yang bersifat *private*, tidak dapat diakses secara bebas oleh semua orang kecuali orang yang berkepentingan.

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 16.04.3 LTS	ISO Ubuntu yang digunakan untuk <i>guest OS</i> .
5.	RAM	2GB	Alokasi RAM untuk <i>guest OS</i>
6.	Port	80	Port forwarding yang digunakan untuk akses aplikasi website

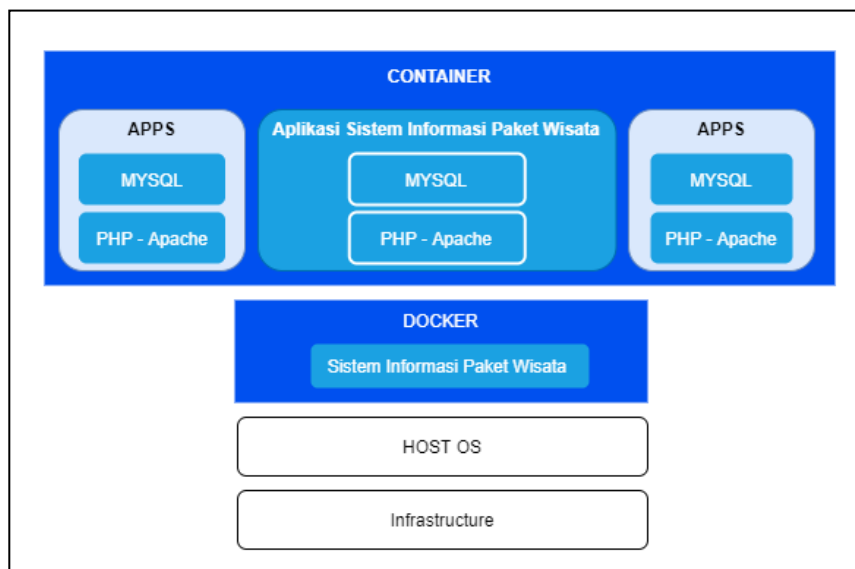
Selain spesifikasi mengenai VM *cloud computing* tersebut, untuk spesifikasi yang digunakan dalam Ubuntu OS yang telah dibuat dalam VM tersebut adalah sebagai berikut:

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 7.2	PHP: Hypertext Preprocessor digunakan untuk membangun situs web dinamis atau CMS
		Mysql 5.7	Digunakan untuk menyimpan data
2.	Git	Versi 2.7	Proyek manajemen kode perangkat lunak
3.	Docker	Versi 18.09.7	Platform yang menggunakan virtualisasi tingkat OS untuk mengirimkan perangkat lunak dalam paket

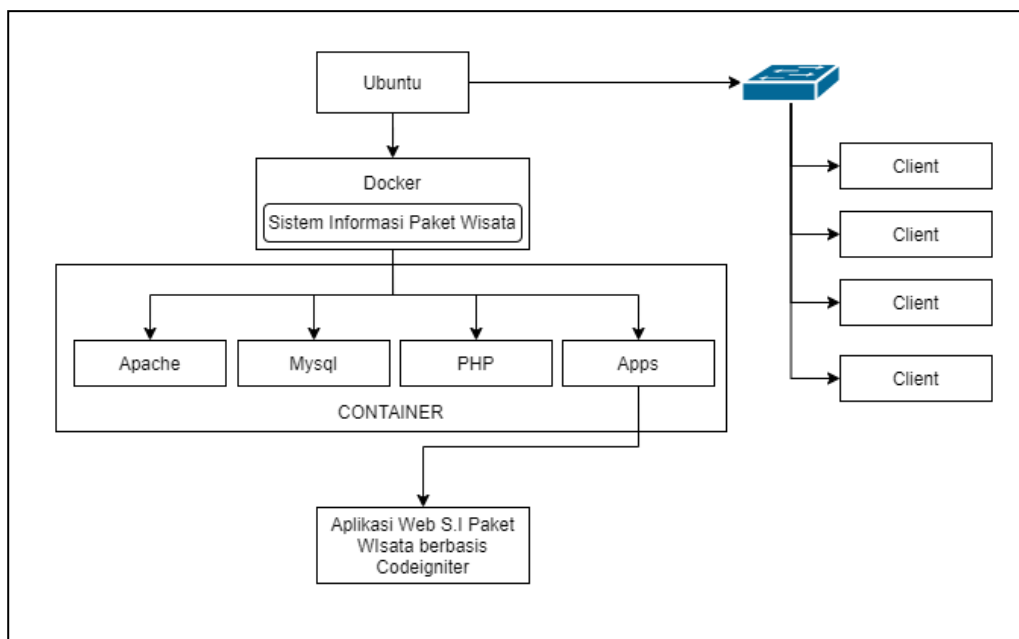
2.2 Rancangan Arsitektur *Cloud Computing*

Rancangan sistem meliputi instalasi sistem operasi Ubuntu 16.04, instalasi docker, menambahkan container dengan image aplikasi web serta melakukan konfigurasi agar virtual container bisa diakses diluar host server. Server ini nantinya akan digunakan untuk men-deploy aplikasi web yang sudah dibuat, karna proses instalasi virtualnya yang lebih efektif dan efisien.



Gambar 2.1 Docker Container

Dengan adanya docker, proses instalasi server semakin cepat karna admin hanya perlu mengambil Ubuntu yang siap pakai dari repository hub menggunakan perintah pull dan untuk pengembangannya bisa menggunakan docker compose. Compose sendiri membutuhkan dockerfile yang mana didalam file tersebut sudah ada syntax instalasi dan konfigurasi.



Gambar 2.2 Rancangan Arsitektur Cloud Computing dengan Docker

Sebagai contoh untuk instalasi webserver apache yang dibutuhkan adalah instalasi apache, php7.2, database serta konfigurasinya. Semua itu bisa digabungkan kedalam satu file, dan di eksekusi menggunakan compose. dengan menggunakan compose, proses tersebut bisa dilakukan dengan 1 baris perintah saja. Setelah docker menjalankan container aplikasi server, yang dibutuhkan selanjutnya adalah akses client menuju container karena secara default, container hanya bisa diakses oleh host saja. Agar client dapat mengakses container, ip:port yang digunakan oleh container tersebut harus di-publish.

2.3 Parameter dan Konfigurasi

Parameter yang digunakan untuk instalasi git, docker, apache dan mysql dapat dilihat pada penjelasan modul berikut ini:

```
$ sudo apt-get install git
```

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
- git : nama paket aplikasi untuk Git

Modul 2.1 Parameter instalasi Apache

```
$ sudo apt-get install docker docker-compose
```

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
- docker : nama paket aplikasi untuk Docker
- docker-compose : nama paket aplikasi untuk docker-compose

Modul 2.2 Parameter instalasi Apache

```
$ sudo apt-get 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 2.3 Parameter instalasi Apache

```
$ sudo apt-get install 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
- apache2 : nama paket aplikasi untuk mysql

Modul 2.4 Parameter instalasi Mysql

```
$ sudo docker-compose up
```

Keterangan:

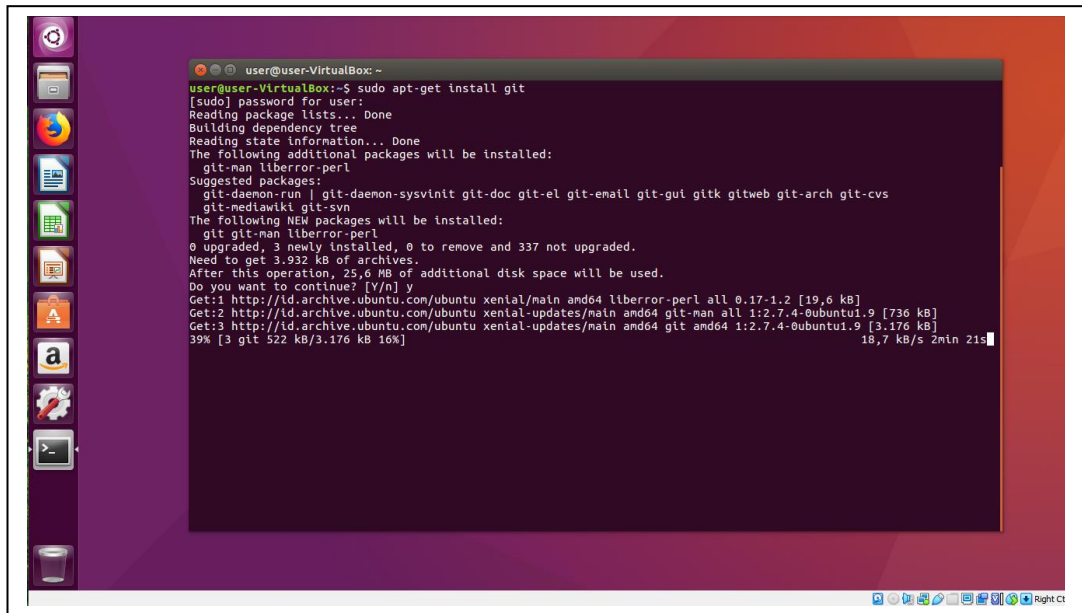
- sudo : perintah untuk eksekusi suatu command dengan hak akses tertinggi (root)
- docker-compose up : command untuk melakukan build container

Modul 2.5 Parameter build container

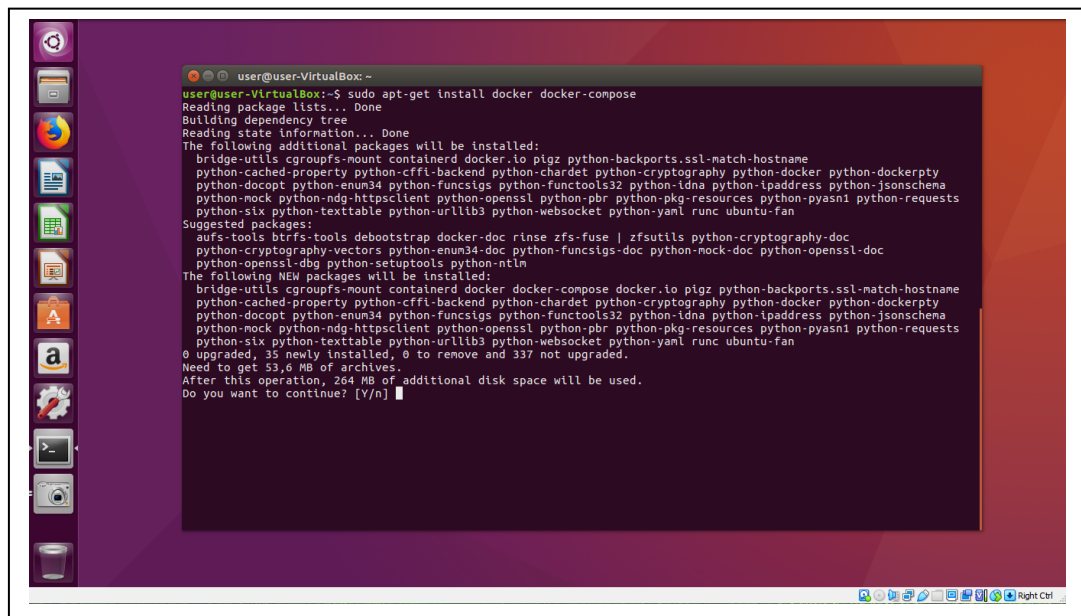
2.4 Tahap Implementasi

Pada tahap ini akan dijelaskan mengenai langkah-langkah implementasi sistem dengan menerapkan docker, tahapan tersebut di antaranya:

- a. Langkah awal adalah instalasi VM Ubuntu sebagai server dimana docker akan diimplementasikan di dalamnya.
- b. Kemudian install paket aplikasi git , docker dan docker-compose.

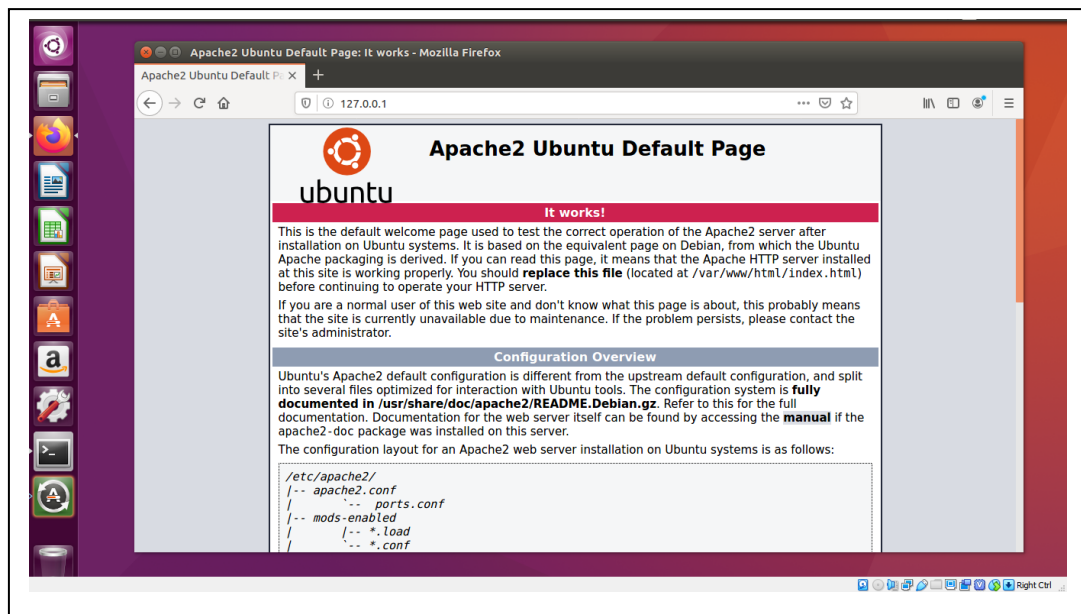


Gambar 2.3 Instalasi paket aplikasi Git



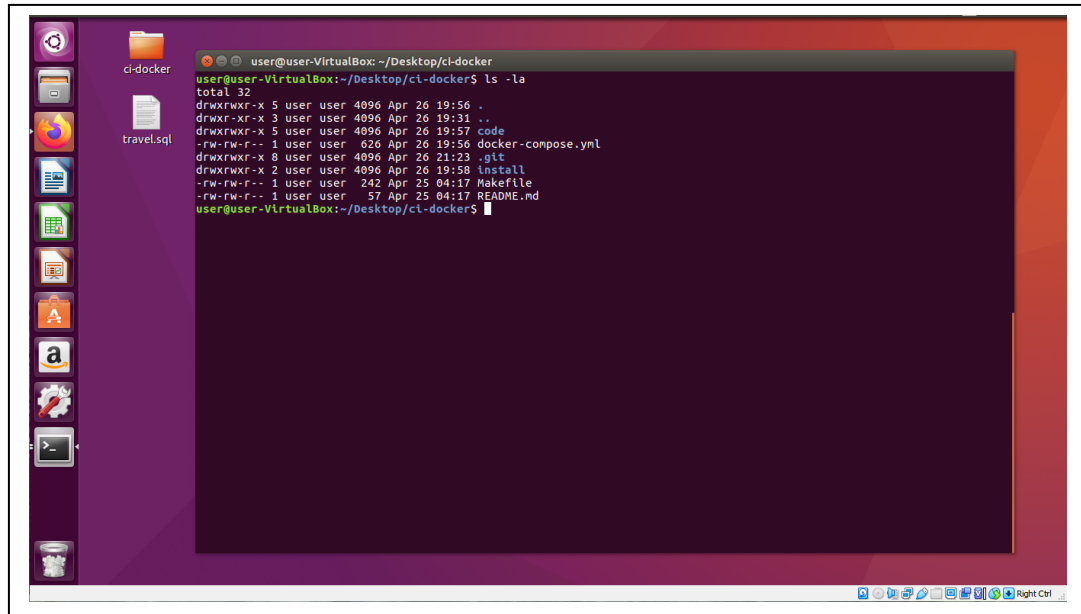
Gambar 2.4 Instalasi paket aplikasi Docker

- c. Langkah selanjutnya adalah pembuatan aplikasi web. Aplikasi web yang digunakan menggunakan framework codeigniter dengan integrasi apache dan mysql. Pada aplikasi yang sudah selesai, simpan semua source code ke dalam repository git termasuk file database yang sudah di backup. Tujuannya untuk mempermudah instalasi dan penyebaran source code.

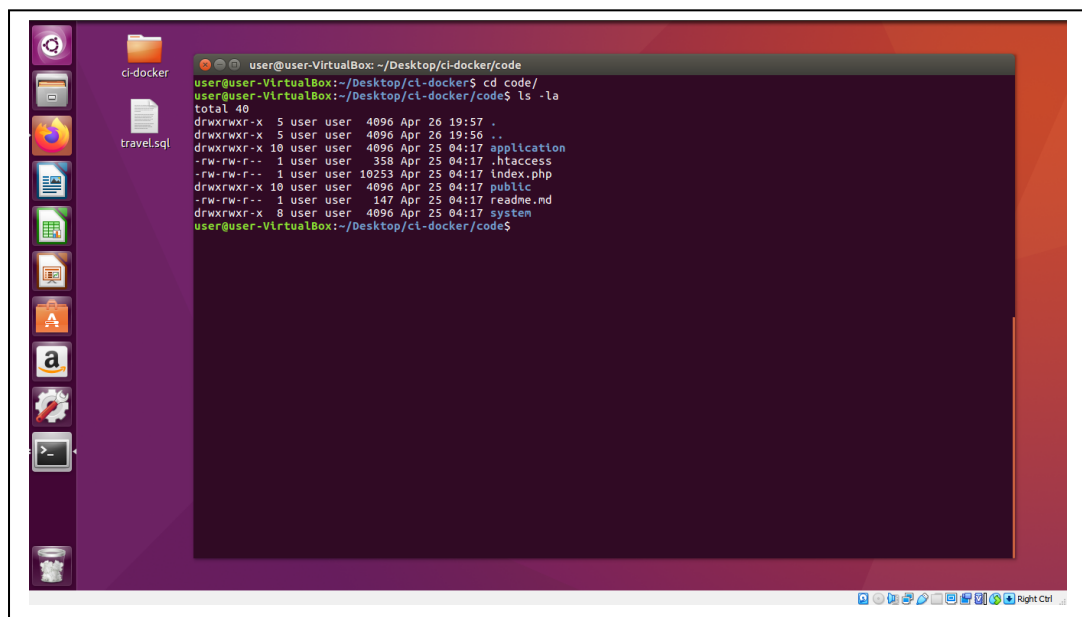


Gambar 2.5 Tampilan browser setelah layanan apache aktif

- d. Setelah aplikasi berjalan maka selanjutnya adalah membuat Dockerfile. Struktur direktori aplikasi web web selanjutnya akan kita ubah seperti gambar 2.3. Untuk source code aplikasi web yang dibuat akan diletakkan di dalam folder “code/”.

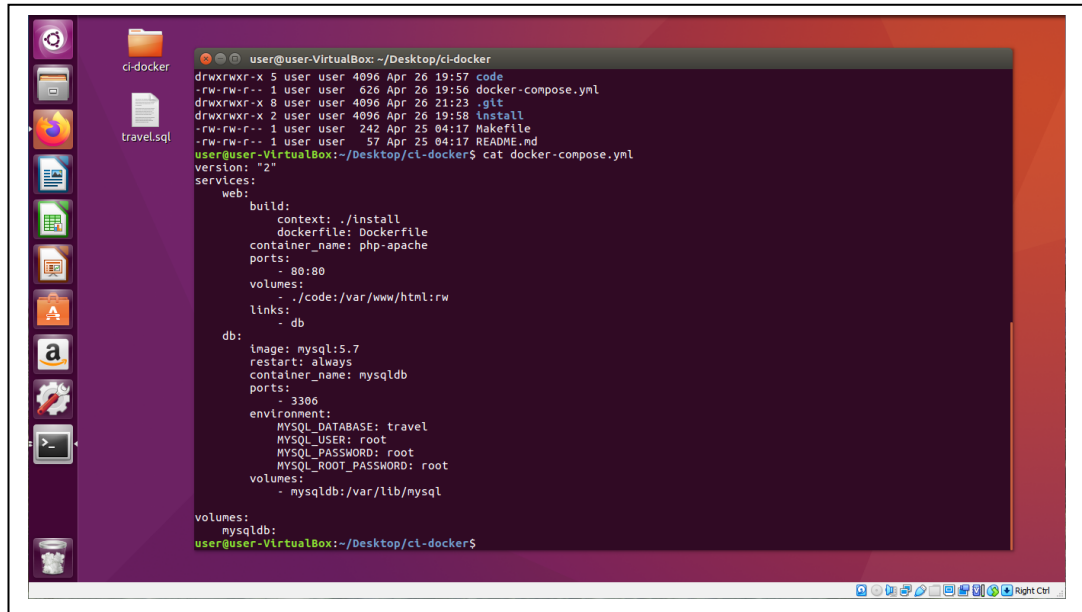


Gambar 2.6 Struktur direktori root

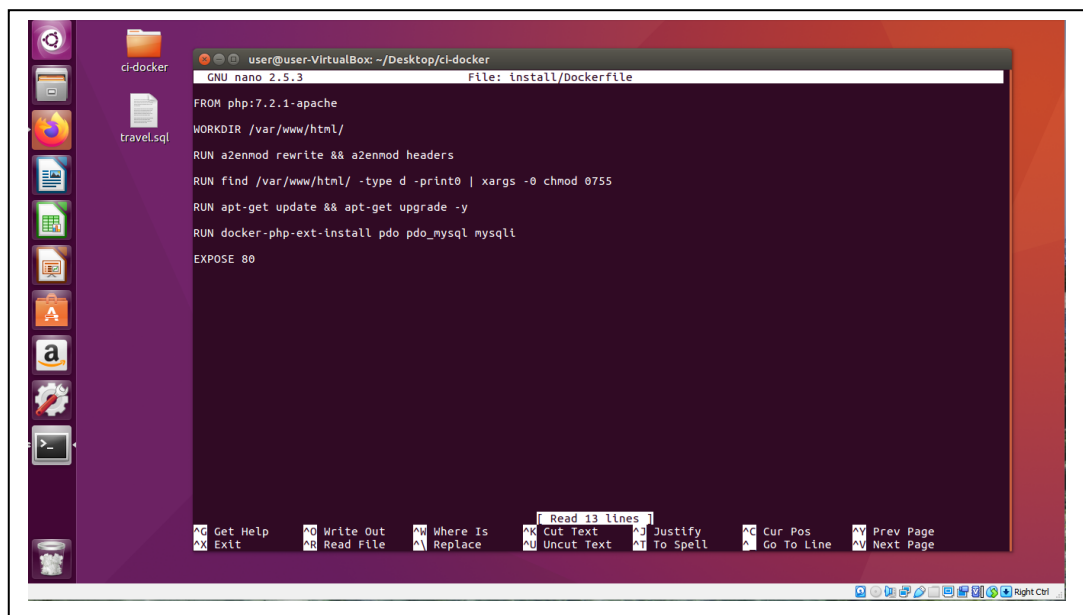


Gambar 2.7 Struktur direktori aplikasi web

- e. Setelah memindahkan aplikasi web ke folder code, langkah selanjutnya adalah membuat Dockerfile dan Docker Compose. Docker Compose diletakkan di folder root sedangkan Dockerfile diletakkan di dalam folder install.

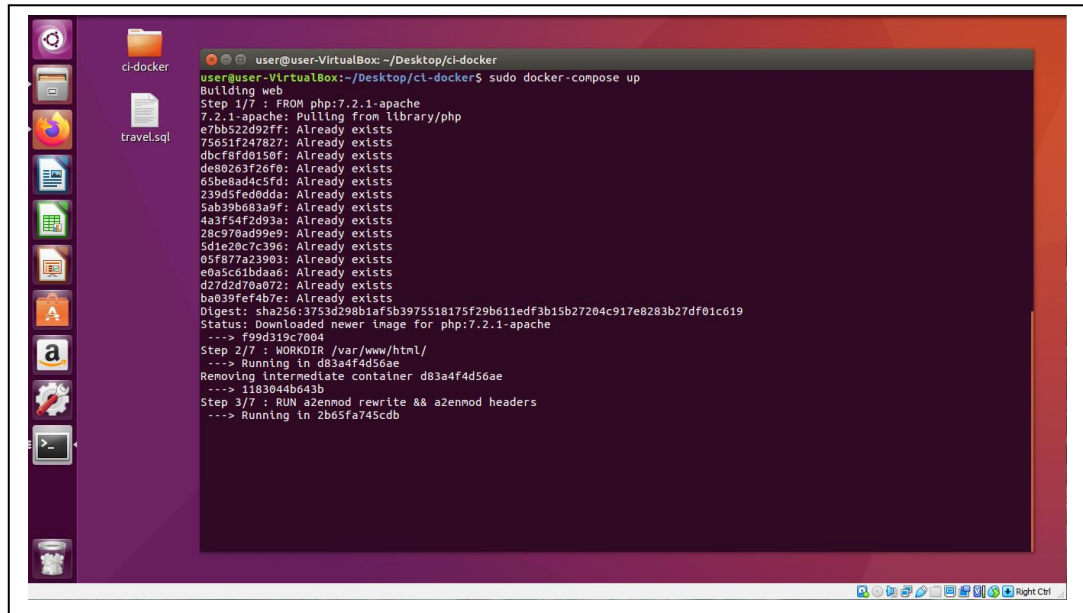


Gambar 2.8 Script docker-compose.yml



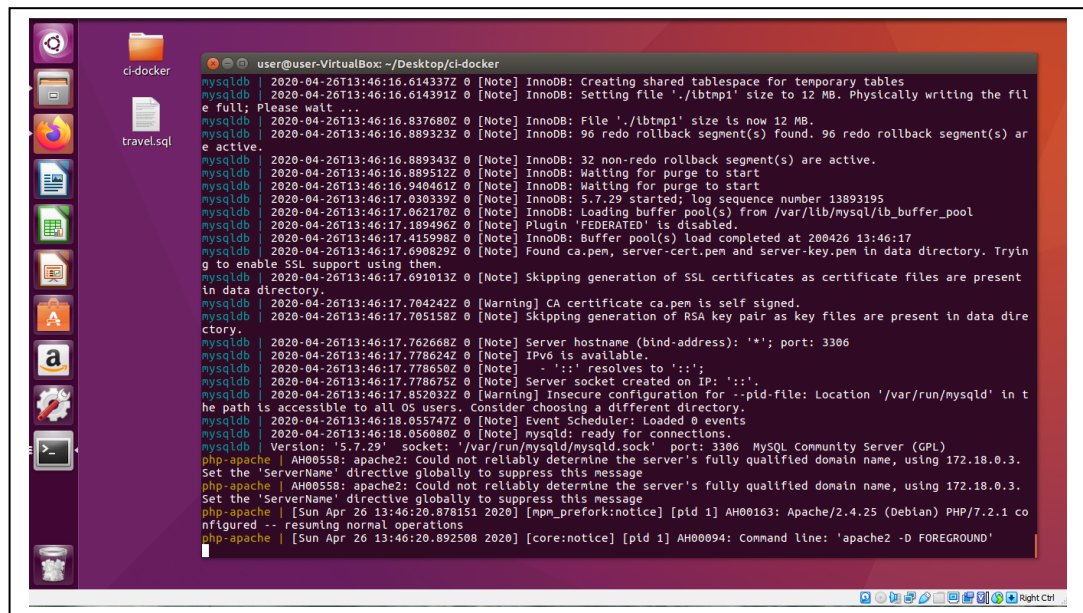
Gambar 2.9 Script Dockerfile

- f. Setelah semua konfigurasi selesai, langkah berikutnya adalah menjalankan Dockerfile. Langkah ini dapat dilakukan dengan cara mengetikkan command pada terminal sesuai modul 2.5.



Gambar 2.10 Proses build container pada docker

- g. Apabila semua paket telah berhasil di install pada container, maka aplikasi website akan langsung dapat digunakan.



Gambar 2.11 Tampilan layanan apache dan mysql

- h. Langkah terakhir adalah menyambungkan aplikasi web dengan database yang sebelumnya sudah dipersiapkan. Letakkan file database pada Desktop, kemudian jalankan perintah

“ `sudo docker exec -i mysql_container mysql -uroot -proot mysql < namadb.sql`”
 untuk mengimport file database tersebut ke database mysql yang berada di container. Apabila tidak ada pesan error yang ditampilkan, kemudian cek tabel database yang baru saja kita import ke database cloud.

```

user@user-VirtualBox: ~/Desktop
user@user-VirtualBox:~/Desktop$ sudo docker exec -i mysql_container mysql -uroot -proot mysql < travel.sql
mysql: [Warning] Using a password on the command line interface can be insecure.
user@user-VirtualBox:~/Desktop$ sudo docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
91bf67722e7   cidocker_web   "docker-php-entrypol..." 11 minutes ago Up 5 minutes   0.0.0.0:80->80
41654fb5d784   php-apache     "docker-entrypoint.s..." 11 minutes ago Up 5 minutes   33060/tcp, 0.0
0.0:32769->33060/tcp mysql:5.7
user@user-VirtualBox:~/Desktop$ sudo docker exec -it 41654fb5d784 bin/bash
sudo: exec: command not found
user@user-VirtualBox:~/Desktop$ sudo docker exec -it 41654fb5d784 bin/bash
root@41654fb5d784:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.29 MySQL Community Server (GPL)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use travel;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

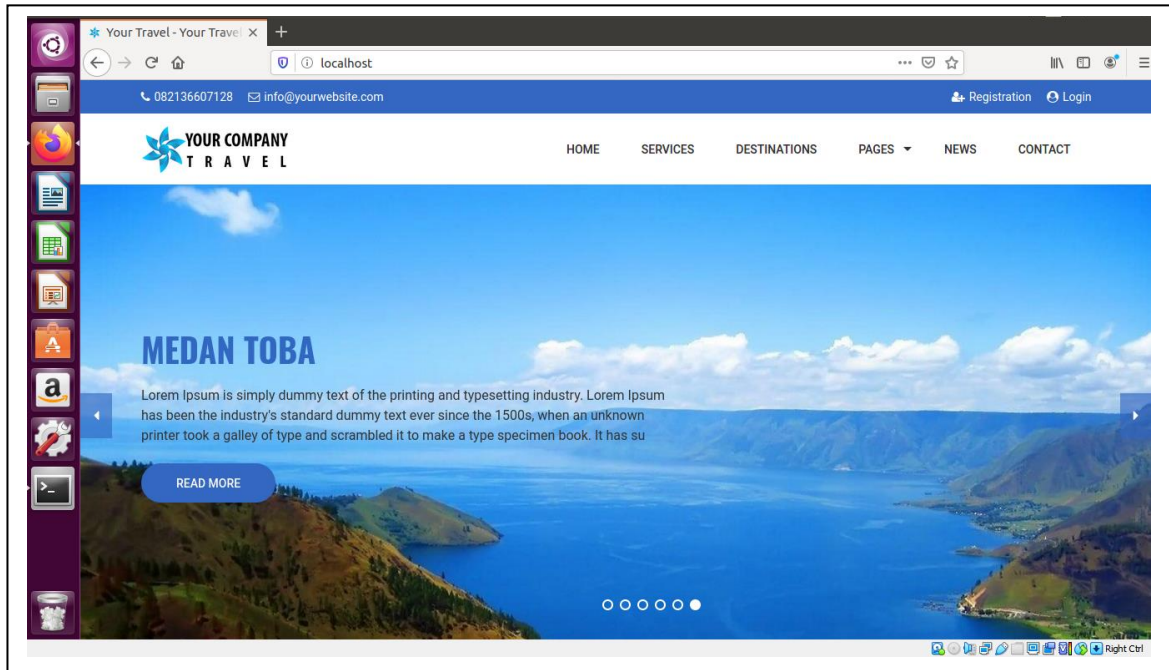
Database changed
mysql> show tables;
+-----+
| Tables_in_travel |
+-----+
| tbl_category     |
| tbl_client       |
| tbl_comment      |
| tbl_destination  |
+-----+
    
```

Gambar 2.12 Tampilan tabel database

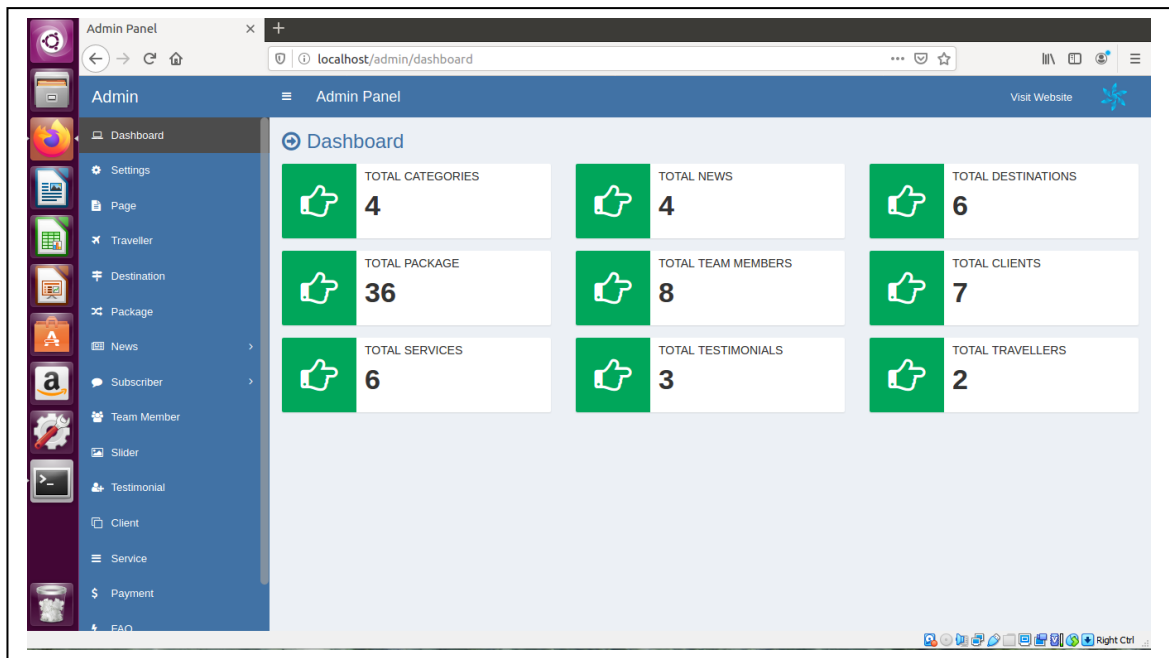
- i. Untuk pengaturan di aplikasi web pastikan hostname pada konfigurasi database adalah nama dari *mysql_container* yang didefinisikan pada docker-compose.

2.5 Hasil Implementasi

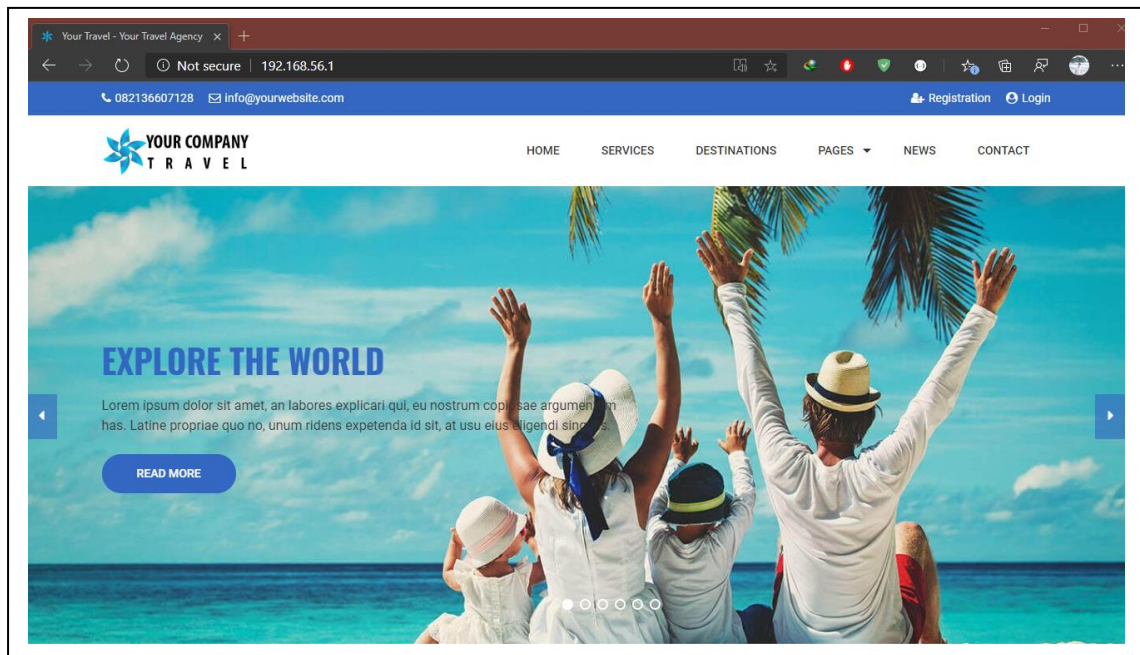
Pada tahap ini hasil dari implementasi menunjukkan aplikasi web telah berjalan sesuai rancangan dimana dapat menampilkan seluruh halaman dari aplikasi beserta panel adminnya.



Gambar 2.13 Tampilan awal/home



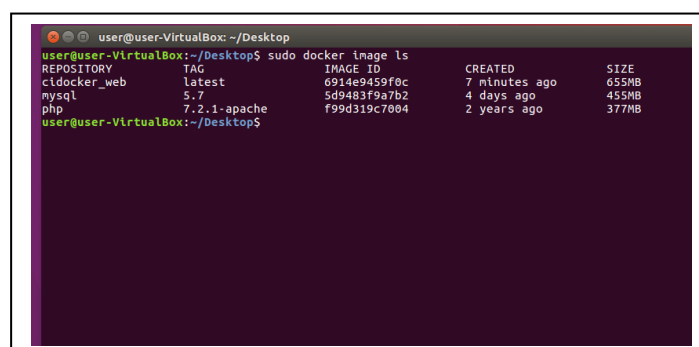
Gambar 2.14 Tampilan panel admin



Gambar 2.15 Tampilan sisi Client Windows

2.6 Pengujian Singkat

Pada tahap ini pengujian dilakukan mengatasi beberapa masalah yang dibahas. Pertama efisiensi dalam proses produksi/development, dari hasil yang dihasilkan dengan menggunakan docker pengguna hanya perlu menjalankan perintah “ sudo docker-compose up” untuk melakukan proses pengemasan paket (container) dan “sudo docker-compose up – build” apabila ada perubahan dan ingin disimpan. Selain itu struk pengkodean juga sistematis dimana semua konfigurasi terletak pada Dockerfile dan Docker Compose, seorang developer hanya perlu mengubah file tersebut untuk menambah atau mengurangi paket aplikasi. Kedua virtualisasi dengan container memiliki ukuran file yang jauh lebih kecil karena tidak perlu menyiapkan sistem operasi secara penuh dan instalasi yang lebih mudah.



Gambar 2.16 Resource Docker

BAB III

JADWAL Pengerjaan dan Pembagian Tugas

(Tuliskan pembagian tugas pembuatan proyek mulai dari perancangan hingga pembuatan laporan.)

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				Mei	
		3	4	1	2	3	4	1	2
1.	Analisa Persoalan								
2.	Pembagian Tugas								
3.	Dsb..								
4.	Pengerjaan 4								
5.	Pengerjaan 5								
6.	Pengerjaan 6								
7.	Pengerjaan 7								
8.	Pengerjaan 8								
9.	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	Agus
2.	Pengujian Singkat	Budi
3.	Latar Belakang Masalah	Candra
4.	Agenda Pengerjaan Proyek	Dedi
5.	Dsb...	Candra
6.	Tugas 6	Budi
7.	Tugas 7	Dedi
8.	Tugas 8	Candra

(Pembagian tugas proyek akan diverifikasi oleh asisten praktikum, setiap penanggung jawab wajib bertanggungjawab terhadap tugasnya. Penilaian akan berdasarkan nilai individu dan tim.)

BAB IV

KESIMPULAN DAN SARAN

4.1 Kesimpulan

Tujuan pembuatan dari aplikasi ini adalah untuk memudahkan perusahaan dalam menjalankan operasionalnya tanpa perlu memikirkan mahalanya biaya perawatan, banyaknya memori yang dibutuhkan, atau membutuhkan ruang penyimpanan yang besar. Dari pengujian sample di atas telah dibuktikan tidak perlu memakan waktu yang lama dalam mengeksekusi aplikasi tersebut, dan aplikasi berhasil menampilkan outpun mulai dari halaman web awal lalu panel admin.

4.2 Saran

Saran penulis dalam penugasan projek akhir adalah kurangnya waktu pengerjaan yang diberikan dikarenakan jadwal kuliah yang padat dan pada masa pandemi ini banyak jadwal kuliah yang tidak sesuai dengan jadwalnya mengakibatkan terganggunya penyelesaian projek akhir ini.

DAFTAR PUSTAKA

Tambahkan daftar pustaka dengan format yang digunakan di IF, yakni APA Style. Setidaknya ada 5 daftar pustaka yang Anda gunakan untuk menyelesaikan proyek ini. Spacing dalam satu paragraf single, tambahkan spasi/jarak antar paragraf

Wang, Shulong., Hou, Yibin., Gao, Fang., & Ji, Xinrong. 2016. "A Novel IoT Access Architecture for Vehicle Monitoring System". 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT).

_____, <<https://idcloudhost.com/pengertian-internet-of-things-iot/>>, (23 Jun 2016, accessed 10 Mei 2019)

Erick, Jan Solem. 2012. *Programming Computer Vision with Python*.

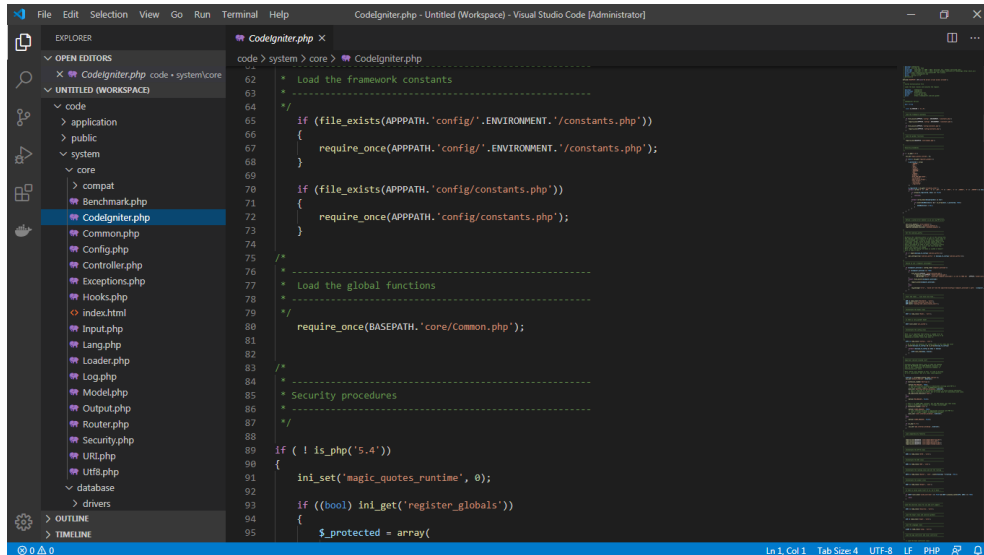
APLIKASI E-SERVICES BERBASIS CLOUD COMPUTING, Yuli Fauziah (2013). Seminar Nasional Informatika 2013 (semnasif 2013) ISSN: 1979-2328 UPN "Veteran" Yogyakarta, 18 Mei 2013

Cloud computing: Architecture & Challenges, Shalini Joshi, Uma Kumari, 2016, Mody University International Journal of Computing and Engineering Research Vol. 1 Issue 1 (December 2016), p.p.56-60

V. Spoorthy, M. Mamatha, and B. S. Kumar, "A Survey on Data Storage and Security in Cloud Computing," A Monthly Journal of Computer Science and Information Technology, vol. 3, June 2014, pp. 306-313.

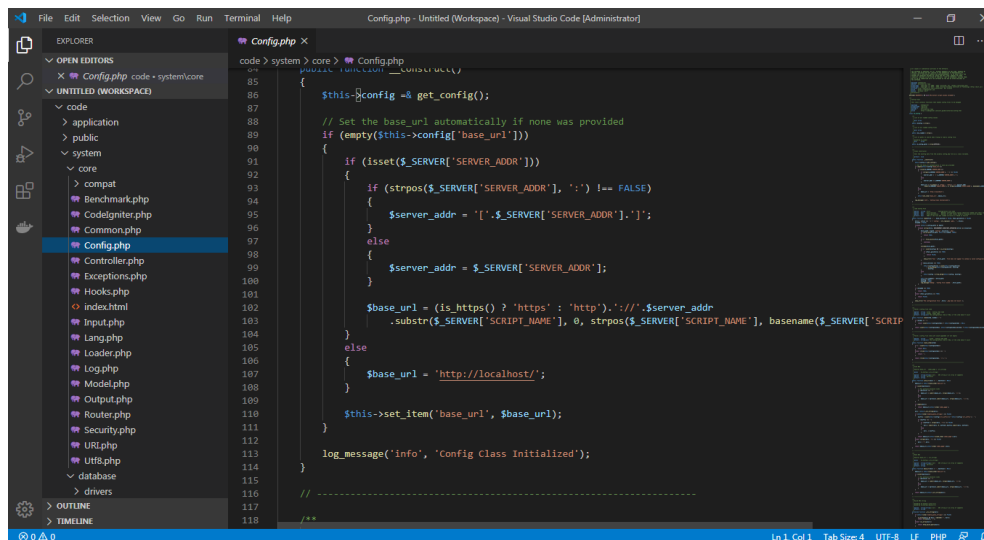
LAMPIRAN

Lampiran pada bagian ini dapat berupa screenshoot, listing program yang terlalu panjang, dan sebagainya, atau dapat juga tugas bilamana diminta oleh asisten praktikum.



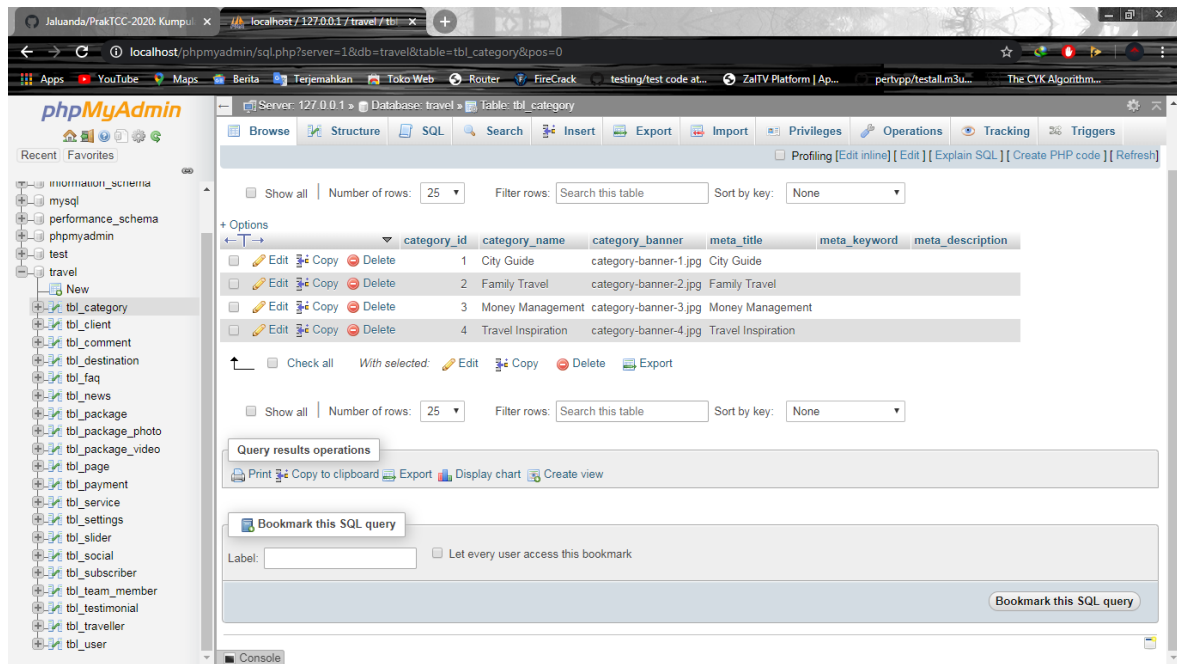
```
code > system > core > CodeIgniter.php
62 * Load the framework constants
63 *
64 */
65 if (file_exists(APPPATH.'config/'.ENVIRONMENT.'/constants.php'))
66 {
67     require_once(APPPATH.'config/'.ENVIRONMENT.'/constants.php');
68 }
69
70 if (file_exists(APPPATH.'config/constants.php'))
71 {
72     require_once(APPPATH.'config/constants.php');
73 }
74
75 /*
76 * Load the global functions
77 *
78 */
79 require_once(BASEPATH.'core/Common.php');
80
81 /*
82 * Security procedures
83 *
84 */
85 if ( ! is_php('5.4'))
86 {
87     ini_set('magic_quotes_runtime', 0);
88
89     if ((bool) ini_get('register_globals'))
90     {
91         $protected = array(
92
```

Code Igniter



```
code > system > core > Config.php
85 public function __construct()
86 {
87     $this->config =& $this->get_config();
88
89     // Set the base_url automatically if none was provided
90     if (empty($this->config['base_url']))
91     {
92         if (isset($_SERVER['SERVER_ADDR']))
93         {
94             if (strpos($_SERVER['SERVER_ADDR'], ':') !== FALSE)
95             {
96                 $server_addr = '['.$_SERVER['SERVER_ADDR'].']';
97             }
98             else
99             {
100                 $server_addr = $_SERVER['SERVER_ADDR'];
101             }
102
103             $base_url = (is_https() ? 'https' : 'http').'://'.$server_addr
104                 .substr($_SERVER['SCRIPT_NAME'], 0, strpos($_SERVER['SCRIPT_NAME'], basename($_SERVER['SCRIPT_NAME'])));
105         }
106         else
107         {
108             $base_url = 'http://localhost/';
109         }
110
111         $this->set_item('base_url', $base_url);
112     }
113
114     log_message('info', 'Config Class Initialized');
115 }
116
117 // -----
118 /**
```

Config



Database

