# SISTEM MANAJEMEN LAYANAN WEB BERBASIS PLATFORM AS A SERVICE (PAAS) DENGAN API OPENSTACK

**SKRIPSI**



**IDA BAGUS RATHU EKA SURYA WIBAWA**

**1308605045**

**PROGRAM STUDI INFORMATIKA**

**JURUSAN ILMU KOMPUTER**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**JIMBARAN**

**2021**

# SURAT PERNYATAAN KEASLIAN KARYA ILMIAH

Yang bertanda tangan di bawah ini menyatakan bahwa naskah Skripsi dengan judul:

**“SISTEM MANAJEMEN LAYANAN WEB BERBASIS PLATFORM AS A SERVICE (PAAS) DENGAN API OPENSTACK”**

Nama : Ida Bagus Rathu Eka Surya Wibawa

NIM : 1308605045

Program Studi : Teknik Informatika

E-mail : guzekawibawa@gmail.com

Nomor telp/HP : 081337482800

Alamat : Jalan Pratu Made Rambug, Br. Sasih No. 49 Batubulan

Belum pernah dipublikasikan dalam dokumen skripsi, jurnal nasional maupun internasional atau dalam prosiding manapun, dan tidak sedang atau akan diajukan untuk publikasi di jurnal atau prosiding manapun. Apabila di kemudian hari terbukti terdapat pelanggaran kaidah-kaidah akademik pada karya ilmiah saya, maka saya bersedia menanggung sanksi-sanksi yang dijatuhkan karena kesalahan tersebut, sebagaimana diatur oleh Peraturan Menteri Pendidikan Nasional Nomor 17 Tahun 2010 tentang Pencegahan dan Penanggulangan Plagiat di Perguruan Tinggi.

Demikian Surat Pernyataan ini saya buat dengan sesungguhnya untuk dapat dipergunakan bilamana diperlukan.

Badung, 29 Januari 2021

Yang membuat pernyataan,

Ida Bagus Rathu Eka Surya Wibawa

NIM. 1308605045

# LEMBAR PENGESAHAN TUGAS AKHIR

Judul : Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) dengan API Openstack

Kompetensi : Jaringan Komputer

Nama : Ida Bagus Rathu Eka Surya Wibawa

NIM : 1308605045

Tanggal Seminar : 29 Januari 2021

Disetujui Oleh :

|  |  |
| --- | --- |
| Pembimbing I | Penguji I |
|  |  |
| I Dewa Made Bayu Atmaja Darmawan, S.Kom., M.Cs.  NIP. 198901272012121001 | I Gede Santi Astawa, S.T., M.Cs.  NIP. 198012062006041003 |
| Pembimbing II | Penguji II |
|  |  |
| I Komang Ari Mogi, S.Kom., M.Kom.  NIP. 198409242008011007 | Gst. Ayu Vida Mastrika Giri, S.Kom., M.Cs.  NIP. 1990060620181123001 |
|  | Penguji III |
|  |  |
|  | Made Agung Raharja,S.Si.,M.Cs  NIP. 1985091920181113001 |
| Mengetahui,  Ketua Jurusan Ilmu Komputer  FMIPA Universitas Udayana  Ketua,  Dr. I Ketut Gede Suhartana, S.Kom., M.Kom  NIP. 197201102008121001 | |

Judul : Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) dengan API Openstack

Nama : Ida Bagus Rathu Eka Surya Wibawa (NIM : 1308605045)

Pembimbing : 1. I Dewa Made Bayu Atmaja Darmawan, S.Kom., M.Cs.

2. I Komang Ari Mogi, S.Kom., M.Kom.

# ABSTRAK

Cloud computing merupakan teknologi yang saat ini mulai berkembang dalam banyak aktivitas teknologi informasi. Cloud computing merupakan model komputasi yang semua sumber daya yang ada dalam layanan cloud dijalankan dengan media jaringan internet. Dengan adanya cloud computing memudahkan para pengguna dalam melakukan komputasi tanpa harus melakukan instalasi aplikasi pada komputer, pengguna hanya perlu mengaksesnya melalui internet. Cloud computing memiliki beberapa fasilitas yang dapat dipilih oleh pengguna sesuai kebutuhan pengguna seperti Infrastructure as a Service(IaaS), Platform as a Service(PaaS). Serta Software as a Service(SaaS).

Pengembangan Platform as a Service sebagai salah satu teknologi cloud computing yang dapat digunakan oleh pengembang aplikasi untuk mengembangkan aplikasi yang akan dibuat tanpa perlu menyediakan infrastruktur, database, framework aplikasi dan lain sebagainya serta bersifat dinamis. Dalam pengembangan layanan cloud yang dikelola oleh seorang sistem administrator atau sysadmin, tugas menginstalasi dan menkonfigurasi sistem pada server maupun software aplikasi dilakukan dengan otomatis dengan menggunakan platform otomatisasi sistem linux untuk efisiensi waktu dan memanagemen penggunanya dengan lebih mudah.

**Kata kunci: *Cloud Computing, Platform as a Service, automation, ansible, API, Openstack***

Judul : Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) dengan API Openstack

Nama : Ida Bagus Rathu Eka Surya Wibawa (NIM : 1308605045)

Pembimbing : 1. I Dewa Made Bayu Atmaja Darmawan, S.Kom., M.Cs.

2. I Komang Ari Mogi, S.Kom., M.Kom.

**ABSTRACT**

Cloud computing is a technology that is currently starting to develop in many information technology activities. Cloud computing is a computing model in which all resources in cloud services run on the internet network media. With cloud computing, it makes it easier for users to perform computations without having to install applications on computers, users only need to access them via the internet. Cloud computing has several facilities that can be selected by users according to user needs, such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS). As well as Software as a Service (SaaS).

Development of Platform as a Service as a cloud computing technology that can be used by application developers to develop applications that will be created without the need to provide infrastructure, databases, application frameworks and so on and are dynamic. In developing cloud services that are managed by a system administrator or sysadmin, the task of installing and configuring systems on servers and application software is carried out automatically using the Linux system automation platform for time efficiency and easier management of users.

***Key Words* : *Cloud Computing, Platform as a Service, automation, ansible, API, Openstack.***

# KATA PENGANTAR

Penelitian dengan judul " Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) dengan API Openstack " ini disusun dalam rangkaian kegiatan pelaksanaan Tugas Akhir di Jurusan Ilmu Komputer Universitas Udayana.

Selama menyelesaikan penelitian tugas akhir ini, penulis telah banyak menerima bimbingan, pengarahan, petunjuk dan saran. Untuk itu penulis mengucapkan terima kasih yang sebesar-besarnya kepada yang terhormat:

1. Bapak I Dewa Made Bayu Atmaja Darmawan, S.Kom., M.Cs. selaku Pembimbing I dan Bapak I Komang Ari Mogi, S.Kom., M.Kom. selaku Pembimbing II yang telah banyak membimbing dan membantu menyempurnakan penelitian ini.
2. Bapak Dr. I Ketut Gede Suhartana, S.Kom., M.Kom. selaku Ketua Jurusan Ilmu Komputer Universitas Udayana, Bapak I Gusti Ngurah Anom Cahyadi Putra, S.T., M.Cs. selaku Ketua Komisi Tugas Akhir Jurusan Ilmu Komputer Universitas Udayana yang telah banyak memberikan masukan dan motivasi sehingga memperlancar dalam proses pelaksanaan penelitian ini.
3. Bapak dan ibu dosen di Jurusan Ilmu Komputer Universitas Udayana yang bersedia meluangkan waktunya untuk memberikan masukan dalam penyempurnaan penelitian ini.
4. Rekan – rekan mahasiswa Jurusan Ilmu Komputer khususnya angkatan 2013 yang telah memberi dukungan, motivasi, semangat dan kerja sama dalam penelitian ini.

Penulis menyadari tugas akhir ini masih jauh dari kesempurnaan. Untuk itu dengan segala kerendahan hati, penulis mengharapkan kritik dan saran yang bermanfaat dari pembaca demi kesempurnaan tugas akhir ini.

Jimbaran, Januari 2021

Penulis

# DAFTAR ISI

[SISTEM MANAJEMEN LAYANAN WEB BERBASIS PLATFORM AS A SERVICE (PAAS) DENGAN API OPENSTACK i](#_Toc64919282)

[SURAT PERNYATAAN KEASLIAN KARYA ILMIAH i](#_Toc64919283)

[LEMBAR PENGESAHAN TUGAS AKHIR ii](#_Toc64919284)

[ABSTRAK iii](#_Toc64919285)

[KATA PENGANTAR v](#_Toc64919286)

[DAFTAR ISI vi](#_Toc64919287)

[DAFTAR TABEL ix](#_Toc64919288)

[DAFTAR GAMBAR x](#_Toc64919289)

[DAFTAR LAMPIRAN xi](#_Toc64919290)

[BAB I PENDAHULUAN 2](#_Toc64919291)

[1.1. Latar Belakang 2](#_Toc64919292)

[1.2. Rumusan Masalah 2](#_Toc64919293)

[1.3. Batasan Masalah 2](#_Toc64919294)

[1.4. Tujuan Penelitian 2](#_Toc64919295)

[1.5. Manfaat Penelitian 3](#_Toc64919296)

[1.6. Metodelogi Penelitian 3](#_Toc64919297)

[BAB II TINJAUAN PUSTAKA 11](#_Toc64919298)

[2.2. Tinjauan Empiris 11](#_Toc64919299)

[2.3. *Cloud Computing* 12](#_Toc64919300)

[1. IaaS (Infrastructure as a Service) 13](#_Toc64919301)

[2. Platform as a Service (PaaS) 13](#_Toc64919302)

[3. Software as a Service (SaaS) 14](#_Toc64919303)

[2.4. Openstack 14](#_Toc64919304)

[2.5. REST API 15](#_Toc64919305)

[2.6. ANSIBLE 15](#_Toc64919306)

[BAB III ANALISIS DAN PERANCANGAN SISTEM 17](#_Toc64919307)

[3.1. Analisis Kebutuhan 17](#_Toc64919308)

[3.2. Penentuan Proses – proses dalam Operasi Sistem 18](#_Toc64919309)

[3.3. Perancangan Antarmuka Sistem 18](#_Toc64919310)

[3.4. Skenario Pengujian Sistem 18](#_Toc64919311)

[1. BlackBox Testing 19](#_Toc64919312)

[2. Performance Testing 20](#_Toc64919313)

[BAB IV HASIL DAN PEMBAHASAN 21](#_Toc64919314)

[4.1. Gambaran Umum Sistem 21](#_Toc64919315)

[4.2. Lingkungan Perancangan dan Implementasi Sistem 21](#_Toc64919316)

[4.3. Implementasi Basis Data 22](#_Toc64919317)

[4.4. Implementasi Antarmuka 22](#_Toc64919318)

[4.5. Implementasi Program 29](#_Toc64919319)

[4.6. Pengujian Sistem 42](#_Toc64919320)

[4.7. Analisa 48](#_Toc64919321)

[BABI V SIMPULAN DAN SARAN 50](#_Toc64919322)

[5.1. Simpulan 50](#_Toc64919323)

[5.2. Saran 50](#_Toc64919324)

[DAFTAR PUSTAKA 51](#_Toc64919325)

[LAMPIRAN 52](#_Toc64919326)

# DAFTAR TABEL

[Tabel 3.1 Tabel Fitur Penelitian 15](#_Toc27638313)

[Tabel 3.2 Tabel Kebutuhan Fungsional 17](#_Toc27638314)

[Tabel 3.3 Tabel Kebutuhan Non Fungsional 18](#_Toc27638315)

[Tabel 4.1 Kode Input File 37](#_Toc27638316)

[Tabel 4.2 Kode Result dari Input File 37](#_Toc27638317)

[Tabel 4.3 Kode Mengirim File 38](#_Toc27638318)

[Tabel 4.4 Kode Menerima File 40](#_Toc27638319)

[Tabel 4.5 Kode Ekstraksi Fitur 41](#_Toc27638320)

[Tabel 4.6 Kode Menerima Hasil Ekstraksi Fitur 44](#_Toc27638321)

[Tabel 4.7 Kode Jaccard Distance 46](#_Toc27638322)

[Tabel 4.8 Kode KNN 48](#_Toc27638323)

[Tabel 4.9 Hasil Pengujian ke-1 50](#_Toc27638324)

[Tabel 4.10 Hasil Pengujian ke-2 51](#_Toc27638325)

[Tabel 4.11 Hasil Pengujian ke-3 52](#_Toc27638326)

[Tabel 4.12 Hasil Pengujian ke-4 54](#_Toc27638327)

[Tabel 4.13 Hasil Pengujian ke-5 55](#_Toc27638328)

# DAFTAR GAMBAR

[Gambar 1.1. Kerangka kerja penelitian 4](#_Toc66013869)

[Gambar 1.2. Desain Kerja Sistem 5](#_Toc66013870)

[Gambar 1.3. *Flowchart* Konfigurasi Otimatis 7](#_Toc66013871)

[Gambar 1.4. *Flowchart Front-end* Sistem 8](#_Toc66013872)

[Gambar 1.5. Rancangan Entity Relationship Diagram 9](#_Toc66013873)

[Gambar 4.1. Tabel basis data *user* 22](#_Toc66013874)

[Gambar 4.2. Tabel basis data *project* 22](#_Toc66013875)

[Gambar 4.0.3 Tampilan Menu Awal 23](#_Toc66013876)

[Gambar 4.0.4 Tampilan Hasil Uji Dataset 23](#_Toc66013877)

[Gambar 4.0.5 Tampilan Menu Detail 24](#_Toc66013878)

[Gambar 4.0.6 Tampilan Menu Deteksi 25](#_Toc66013879)

[Gambar 4.0.7 Tampilan Pilih File 25](#_Toc66013880)

[Gambar 4.0.8 Tampilan Server 26](#_Toc66013881)

[Gambar 4.0.9 Hasil Ekstraksi Fitur pada Server 27](#_Toc66013882)

[Gambar 4.0.10 Tampilan Hasil Deteksi 28](#_Toc66013883)

[Gambar 4.0.11 Tampilan Hasil Deteksi 29](#_Toc66013884)

[Gambar 4.0.12 Tampilan Hasil Deteksi 29](#_Toc66013885)

[Gambar 4.0.13 Tampilan Hasil Deteksi 30](#_Toc66013886)

[Gambar 4.0.14 Tampilan Hasil Deteksi 30](#_Toc66013887)

[Gambar 4.0.15 Tampilan Hasil Deteksi 31](#_Toc66013888)

[Gambar 4.0.16 Tampilan Hasil Deteksi 31](#_Toc66013889)

[Gambar 4.0.17 Tampilan Hasil Deteksi 32](#_Toc66013890)

[Gambar 4.0.18 Tampilan Hasil Deteksi 32](#_Toc66013891)

[Gambar 4.0.11 Grafik Hasil Pengujian ke-1 50](#_Toc66013892)

[Gambar 4.0.12 Grafik Hasil Pengujian ke-2 50](#_Toc66013893)

[Gambar 4.0.13 Grafik Hasil Pengujian ke-3 50](#_Toc66013894)

[Gambar 4.0.14 Grafik Hasil Pengujian ke-4 50](#_Toc66013895)

[Gambar 4.0.15 Grafik Hasil Pengujian ke-5 50](#_Toc66013896)

[Gambar 4.0.16 Grafik Akurasi 50](#_Toc66013897)

# DAFTAR LAMPIRAN

Data Permissions 60

Data API Calls 67

# BAB I PENDAHULUAN

## Latar Belakang

Pada era teknologi seperti sekarang ini,

## Rumusan Masalah

Berdasarkan latar belakang masalah dapat dirumuskan rumusan masalah sebagai berikut :

* 1. Bagaimana otomatisasi dengan *Ansible* berkerja pada ruang lingkup komputasi awan.
  2. Bagaimana membangun sistem yang memfasilitasi developer dalam mengembangkan *website* ke dalam komputasi awan.

## Batasan Masalah

Agar penelitian lebih fokus, permasalah yang dicakup tidak akan terlalu luas dan akan sesuai dengan maksud dan tujuan yang ingin dicapai. Ruang lingkup yang dibahas dalam penelitian ini akan dibatasi hanya pada :

1. Sistem dibuat dalam bentuk purwa rupa.
2. Layanan komputasi awan menggunakan *Openstack.*
3. Sistem meyalani *virtual server* untuk membangun layanan aplikasi *web*.
4. Sistem dibangun menggunakan Bahasa permrograman *python.*
5. Sistem tidak menyediakan *domain* untuk layanan yang diterima *developer website*

## Tujuan Penelitian

Berdasarkan rumusan masalah, tujuan dari penelitian ini adalah sebagai berikut :

* + 1. Membantu *developer website* mengurangi beban resource komputer dalam mengembangkan aplikasi yang dibuat.

1. Membantu *developer website* membuat sebuah *server website* dan melakukan instalasi *web server* secara otomatis*.*

## Manfaat Penelitian

Dengan dilakukannya penelitian ini, diharapkan mampu mengurangi beban kerja komputer *developer website* dengan komputasi awan yang menyediakan *virtual server web* dan sebuah *platform digital* dalam mengembangkan *website*.

## Metodelogi Penelitian

Bagian ini akan menjelaskan mengenai langkah – langkah yang akan dilakukan dalam merancang sistem manajemen layanan web berbasis Platform as a Service (PaaS) dengan API openstack.

1. **Metode Pengumpulan Data**

Pada tahap ini dilakukan analisis kebutuhan sistem, analisis kebutuhan sistem meliputi data yang digunakan, pembelajaran dari referensi yang sudah ada dan perangkat yang digunakan baik perangkat lunak maupun perangkat keras:

Tahapan pertama yang dilakukan dalam penelitian ini adalah mengidentifikasi permasalahan. Tahap ini merupakan tahap yang paling penting dalam penelitian karena jalannya penelitian didasarkan atas permasalahan yang terjadi. Setelah menentukan masalah yang terjadi, tahapan yang diperlukan selanjutnya adalah menentukan rumusan masalah dan tujuan yang ingin dicapai dalam penenlitian. Pada penelitian ini identifikasi permasalahan dilakukan dengan menggunakan teknik observasi, dari teknik ini maka akan dapat diketahui mengenai keluhan – keluhan yang ada di lapangan.

Tahap kedua yang dilakukan dalam metodelogi penelitian ini adalah studi literatur. Studi literatur dilakukan dengan mengambil literatur – literatur pendukung dari jurnal – jurnal ilmiah, baik jurnal dalam negeri ataupun jurnal luar negeri dan dari beberapa buku. Dalam studi literatur ini, penulis mencari sumber terkait permasalahan – permasalahan yang perlu menjadi perbaikan dalam penelitian selanjutnya..

1. **Kerangka Kerja Penelitian**

Bagian ini menjelaskan tentang bagaimana penelitian ini dilakukan. Berikut adalah beberapa proses penting yang dilakukan :



Gambar 1.. Kerangka kerja penelitian

Instalasi kebutuhan konfigurasi *VPS(Virtual Private Server)*

Tahap pertama dalam penelitian ini dengan mempersiapkan sistem dan software yang dibutuhkan dalam mengkonfigurasi kebutuhan *web server* dalam menginstalasi modul-modul kebutuhan sistem seperti sistem operasi, *database, framework*, *web server .*

Menentukan proses konfigurasi kebutuhan aplikasi

Pada tahap ini mempersiapkan kebutuhan konfigurasi untuk membangun sebuah *web server*, dari *web service* sampai *database* yang diperlukan untuk menampung setiap *developer*.

Melakukaan proses otomatisasi konfigurasi modul aplikasi

Pada tahap ini melakukan otomatisasi dalam mengkonfigurasi sebuah *web server*. Proses otomatisasi menggunakan Ansible.

Merancang *web front-end* untuk *developer website.*

Pada tahap ini bila semua konfigurasi sudah berjalan dengan baik, dibuat *web* yang dapat digunakan oleh *developer* mengatur kebutuhan *web* yang dibuat serta dapat melihat informasi dari setiap *VPS* yang dimiliki.

Melakukan test pada hasil konfigurasi

Pada tahap ini menguji hasil konfigurasi dari *web* *interface*. Untuk menguji konfigurasi berjalan dengan baik.

Melakukan uji beban *server*

beban ini dilakukan untuk menguji tingkat kemampuan *server* dalam memberikan pelayanan pada developer aplikasi.

1. **Perancangan Sistem**

Pada tahap Perancangan sistem, dilakukan berdasarkan hasil analisa kebutuhan sistem yang sudah dilakukan sebelumnya. Hal ini dilakukan agar, perancangan tidak keluar dari tujuan sistem yang dikembangkan. Berikut merupakan desain arsitektur kerja sistem :



Gambar 1.. Desain Kerja Sistem

Pada gambar 1.2 merupakan hubungan antara setiap perangkat lunak yang ada pada server utama. Berikut ini merupakan penjelasannya :

Web dan database

Database digunakan untuk menyimpan informasi pengguna serta informasi pada VPS yang dimiliki oleh setiap pengguna.

1. Web dan framework flask REST API

Flask REST API digunakan untuk menghubungkan web front-end dengan back-end. Dimana back-end dibuat dengan menggunakan framework flask dari python, selanjutnya dibuat API tersendiri agar web front-end dapat mengirim dan menerima informasi serta konfigurasi yang dilakukan yang nantinya akan diproses oleh back-end server.

1. Database dan ansible API

Hubungan database dengan ansible API akan mengambil informasi pengguna berupa username, email dan password yang nantinya akan dimasukan pada konfigurasi dalam VPS yang selanjutnya sebagai super amin pada VPS yang dibuat.

1. framework flask REST API dan asible API

Hubungan framework flask REST API dengan asible API. Ketika user melakukan konfigurasi pada web utama akan dikirim melalui REST API flask selanjutnya konfigurasi tersebut akan digunakan oleh API ansible untuk mengkonfigurasi VPS yang dibuat.

1. framework flask dan openstack API

Hubungan framework flask dan openstack API adalah pada flask dikonfigurasi untuk dapat terhubung dengan server devstack, dimana server openstack yang mengelola dalam membuat sebuah VPS.

1. openstack API dan server

API openstack digunakan untuk terhubung dengan server back-end yang dibuat dengan framework flask. Untuk dapat mengembangkan aplikasi openstack dari bahasa pemrograman yang berbeda.

1. ansible API dan Virtual Private Server

hubungan ansible API dengan VPS adalah ansible akan mengirimkan konfigurasi pada VPS melalui SSH yang yang dibuat. Selanjutnya pada vps akan melakukan konfigurasi yang diperlukan untuk membangun sebuah web aplikasi secara otomatis. Pengguna hanya perlu memilih keperluan yang ada menu web front-end.

1. server dan Virtual Private Server

hubungan antara server dan VPS(Virtual Private Server) adalah server mengelola dan memberikan resource pada VPS serta mengatur konektivitas setiap VPS yang dimiliki.

1. **Metode Pengembangan Perangkat Lunak**

Pada bagian ini akan menjelaskan proses yang dilakukan oleh perangkat lunak, bagaimana proses tersebut berjalan yang akan dijelaskan pada setiap flowchart berikut ini:

* + - 1. Flowchart Konfigurasi Otomatis



Gambar 1.. *Flowchart* Konfigurasi Otimatis

Pada desain flowchart diatas merupakan garis besar gambaran system yang akan dikerjakan, bagaimana alur platform yang dibuat dapat mengotomatitasi dalam instalasi serta konfigurasi sebuah layanan pada virtual machine. Mulai dari memasukkan aplikasi pendukung yang ingin digunakan seperti database, web service,usernme dan e-mail administrator layanan aplikasi. Kemudian dari hasil inputan tersebut diolah oleh flask API yang dibuat untuk dimasukkan ke dalam konfigurasi yang terdapat ada Ansible API, selanjutnya dari ansible API dengan menggunakan SSH akan melakukan instalasi konfigurasi pada virtual server yang didapat setiap user. Bila konfigurasi telah selesai user akan menerima sebuah ip public untuk dapat mengkakses layanan aplikasi.

* + - 1. Flowchart Front-end Web



Gambar 1.. *Flowchart Front-end* Sistem

Pada desain flowchart Front-end Web merupakan alur kerja web dari sisi front-side atau pada sisi user. Setelah user melakukan registrasi pada website, selanjutnya user akan diarahkan ke menu instalasi dan konfigurasi dalam membangun layanan VPS. Menu akan dibuat secara user friendly sehingga user hanya perlu memilih menu konfigurasi system. Ketika user sudah selesai memilih menu konfigurasi, system akan memvalidasi informasi yang dimasukkan oleh user, bila sudah benar system akan mengirimkan hasil instalasi dan konfigurasi dalam sebuah data berrntuk JSON ke dalam system yang nantinya akan dikirim ke virtual sever untuk melakukan konfigurasi dan instalasi modul kebutuhan aplikasi selanjutnya sistem akan melakukan backup berkala untuk menjaga keamanan data pengguna bila tejadi satu kesalahan dapat dilakukan restore data.

1. **Rancangan *ERD (Entity Relationship Diagram)***

Pada desain *Entity Relationship Diagram (ERD)* diatas menjelaskan hubungan user dengan VPS yang dimiliki user memiliki sebuah vps dengan satu ip public. Serta pada vps akan menyimpan log mulai dari kinerja dan resource yang dimiliki.



Gambar 1.. Rancangan Entity Relationship Diagram

1. **Evaluasi dan Validasi Hasil**

Pada tahap ini dilakukan evaluasi terhadap desain perancangan sistem. Bila sistem telah sesuai dengan kebutuhan awal yang didefinisikan akan dilanjutkan ke tahap implementasi. Namun apabila desain sistem belum memenuhi kebutuhan awal yang didefinisikan, maka akan dilakukan perancangan ulang desain sistem

# BAB II TINJAUAN PUSTAKA

## Tinjauan Empiris

1. **Analisis Implementasi Metode Klasifikasi Bayes Untuk Deteksi *Malware* *Android* (Ridlo & Karima, 2016)**

Pada penelitian ini, penulis mendeteksi *malware* dengan menerapkan metode klasifikasi bayes. Penulis menggunakan dua buah model properti *feature*. Model yang pertama adalah model properti *permissions* dan model yang kedua adalah model properti berbasis *code*. Peneliti melakukan tiga kali pengujian klasifikasi, pertama hanya dengan model properti berbasis *permissions* dan menunjukan akurasi *classifier* sebesar 54.0%, pengujian yang kedua dilakukan dengan model properti berbasis *code* dan menunjukan akurasi *classifier* sebesar 89.5%, dan yang terakhir peneliti melakukan pengujian dengan menggabungkan kedua properti tersebut dan menunjukan akurasi *classifier* sebesar 88.0%.

1. ***Evaluation* *of* *machine* *learning* *classifiers* *for* *mobile* *malware* *detection* (Narudin, Feizollah, Anuar, & Gani, 2014)**

Pada penelitian ini, penulis melakukan perbandingan lima buah metode untuk mendeksi *malware*. Metode yang digunakan adalah *bayes* *network*, *multi-layer* *perception*, knn, j48, dan *random* *forest*. Penulis mendeteksi *malware* dengan menggunakan pendekatan berbasis anomali, dengan duah buah dataset yaitu *public* *dataset* yang diambil dari MalGenome dan *private* *dataset* berupa *malware* yang dikumpulkan sendiri oleh penulis. *Feature* yang digunakan oleh penulis berupa *feature* lalu lintas data dalam sebuah jaringan. Dari pengujian didapatkan hasil akurasi *true* *positive* untuk dataset pertama dari metode BN sebesar 99.97%, metode MLP sebesar 91.73%, metode J48 sebesar 99.98%, metode KNN sebesar 99.65%, metode random forest sebesar 99.98%. Untuk dataset kedua, didapatkan hasil akurasi *true* *positive* dari metode BN sebesar 99.98%, metode MLP sebesar 88.25%, metode J48 sebesar 99.98%, metode KNN sebesar 99.65%, metode random forest sebesar 99.99%.

1. ***Merging* *Permission* *and* *API* *Features* *for Android Malware Detection* (Qiao, Sung, & Liu, 2016)**

Pada penelitian ini, penulis menggunakan algoritma SVM, ANN, RF untuk mendeteksi *malware*. Fitur yang digunakan penulis dalam penelitian ini berupa *permission* dan API. Pada penelitian ini, didapatkan hasil yaitu tingkat ketepatan SVM, ANN, dan RF dalam mengklasifikasi *malware* menggunakan fitur *Binary* *permission* sebesar 87.72%, 92.12%, 87.84%. tingkat ketepatan SVM, ANN, dan RF dalam mengklasifikasi *malware* menggunakan fitur *Binary* API sebesar 88.40%, 93.84%, 90.10%. tingkat ketepatan SVM, ANN, dan RF dalam mengklasifikasi *malware* menggunakan fitur gabungan *Binary* *permission* dengan *Binary* API sebesar 91.48%, 96.00%, 94.08%. Tingkat ketepatan SVM, ANN, dan RF dalam mengklasifikasi *malware* menggunakan fitur *Numerical* *permission* sebesar 79.68%, 92.52%, 92.52%. tingkat ketepatan SVM, ANN, dan RF dalam mengklasifikasi *malware* menggunakan fitur *Numerical* API sebesar 78.40%, 94.18%, 94.34%. tingkat ketepatan SVM, ANN, dan RF dalam mengklasifikasi *malware* menggunakan fitur gabungan *Numerical* *permission* dengan *Numerical* API sebesar 81.68%, 94.66%, 94.98%.

## *Cloud Computing*

Cloud computing mengacu pada aplikasi dan service yang berjalan dalam jaringan data terdistribusi dengan menggunakan sumberdaya virtual dan internet akses protokol pada umumnya. Hal ini dibedakan pada gagasan sumberr daya virtual dan detail dari mesin fisik sistem dalam software yang berjalan secara abtraksi dari user. (Sosinsky, 2011). Cloud Computing secara sederhana adalah “layanan teknologi informasi yang bisa dimanfaatkan atau diakses oleh pelanggannya melalui jaringan internet”. Komputasi awan adalah suatu konsep umum yang mencakup SaaS, Web 2.0, dan tren teknologi terbaru lain yang dikenal luas, dengan tema umum berupa ketergantungan terhadap Internet untuk memberikan kebutuhan komputasi pengguna. Sebagai contoh, Google Apps menyediakan aplikasi bisnis umum secara sharing yang diakses melalui suatu penjelajah web dengan perangkat lunak dan data yang tersimpan di server. Jenis-jenis dari Cloud Computing dapat dijabarkan sebagai berikut:

### IaaS (Infrastructure as a Service)

Terletak satu level lebih rendah dibanding PaaS. Ini adalah sebuah layanan yang “menyewakan” sumberdaya teknologi informasi dasar, yang meliputi media penyimpanan, processing power, memory, sistem operasi, kapasitas jaringan dan lainlain, yang dapat digunakan oleh penyewa untuk menjalankan aplikasi yang dimilikinya. Model bisnisnya mirip dengan penyedia data center yang menyewakan ruangan untuk co-location, tapi ini lebih ke level mikronya. Penyewa tidak perlu tahu, dengan mesin apa dan bagaimana caranya penyedia layanan menyediakan layanan IaaS. Yang penting, permintaan mereka atas sumberdaya dasar teknologi informasi itu dapat dipenuhi.

### Platform as a Service (PaaS)

Konsepnya hampir serupa dengan IaaS. Namun Platform disini adalah penggunaan operating system dan infrastruktur pendukungnya. Yang cukup terkenal adalah layanan dari situs Force.Com serta layanan dari para vendor server.Seperti namanya, PaaS adalah layanan yang menyediakan modul-modul siap pakai yang dapat digunakan untuk mengembangkan sebuah aplikasi, yang tentu saja hanya bisa berjalan diatas platform tersebut. Seperti juga layanan SaaS, pengguna PaaS tidak memiliki kendali terhadap sumber daya komputasi dasar seperti memory, media penyimpanan, processing power dan lain-lain, yang semuanya diatur oleh provider layanan ini. Pionir di area ini adalah Google AppEngine, yang menyediakan berbagai tools untuk mengembangkan aplikasi di atas platform Google, dengan menggunakan bahasa pemrograman Phyton dan Django. Kemudian Salesforce juga menyediakan layanan PaaS melalui Force.com, menyediakan modul-modul untuk mengembangkan aplikasi diatas platform Salesforce yang menggunakan bahasa Apex. Dan mungkin yang jarang sekali kita ketahui, bahwa Facebook juga bisa dianggap menyediakan layanan PaaS, yang memungkinkan kita untuk membuat aplikasi diatasnya.Salah satu yang berhasil menangguk untung besar dari layanan PaaS

### Software as a Service (SaaS)

Berada satu tingkat diatas PaaS dan IaaS, dimana disini yang ditawarkan adalah software atau suatu aplikasi bisnis tertentu. Contoh yang paling mutakhir adalah SalesForce.Com, Service-Now.Com, Google Apps, dsb. SaaS ini merupakan layanan Cloud Computing yang paling dahulu populer.Software as a Service ini merupakan evolusi lebih lanjut dari konsep ASP (Application ServiceProvider). Sesuai namanya, SaaS memberikan kemudahan bagi pengguna untuk bisa memanfaatkan sumberdaya perangkat lunak dengan cara berlangganan. Sehingga tidak perlu mengeluarkan investasi baik untuk in house development ataupun pembelian lisensi. Dengan cara berlangganan via web, pengguna dapat langsung menggunakan berbagai fitur yang disediakan oleh penyedia layanan. Hanya saja dengan konsep SaaS ini, pelanggan tidak memiliki kendali penuh atas aplikasi yang mereka sewa.Hanya fiturfitur aplikasi yang telah disediakan oleh penyedia saja yang dapat disewa oleh pelanggan

## Openstack

OpenStack adalah sistem aplikasi cloud yang mengelola sumberdaya seperti komputasi, penyimpan dan jaringan, yg tersedia pada infrastruktur fisik seperti dalam sebuah fasilitas pusat-data (data center). Admin atau pengguna dapat mengendalikan dan melakukan provisioning atas sumber-daya ini melalui dashboard / antar-muka web. Developer dapat mengakses sumber-daya tersebut melalui sejumlah API standar (Mulyana, 2017).

Openstack merupakan platform cloud-computing open source yang memungkinkan pengguna untuk membangun sebuah "IAAS" Infrastruktur sebagai service cloud yang bergerak secara massal pada komoditas hardware dan skala. Openstack mengontrol kolam besar komponen komputasi awan di seluruh datacenter, semua dikelola melalui dashboard yang menyediakan administrator kontrol penuh sambil memberikan pengguna kemampuan untuk sumber penyediaan melalui antarmuka web.

## REST API

REST (REpresentational State Transfer) merupakan standar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. Umumnya menggunakan HTTP (Hypertext Transfer Protocol) sebagai protocol untuk komunikasi data. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000.

REST API merupakan web service yang bertujuan untuk mendukung kebutuhan server web pada suatu kebutuhan situs atau aplikasi lainnya. program client mengunakan Application Programming Interface (API) untuk berkomunikasi dengan layanan web. Secara umum, API mengekspos seperangkat data dan fungsi untuk memfasilitasi interaksi antara proram komputer dan memungkinkan merekas saling bertukar informasi (Masse, 2012).

Dalam pengaplikasiannya, REST lebih banyak digunakan untuk web service yang berorientasi pada resource. Maksud orientasi pada resource adalah orientasi yang menyediakan resource-resource sebagai layanannya dan bukan kumpulan-kumpulan dari aktifitas yang mengolah resource tersebut

## ANSIBLE

Ansible merupakan sebuah softaware yang bisa membantu seorang sistem administrator untuk melakukan otomasi pada server. ansible merupakan teknologi yang digunakan untuk melakukan otomasi, memudahkan dalam melakukan konfigurasi server, tujuan dibuat ansible membuat hal tersebut menjadi sederhana dan mudah.namun tetap fokus pada keamana dan keandalan dalam melakukan otomasi. ansible menggunakan OpenSSH untuk transportasi ( dengan mode socket yang cepat).

Dengan ansible sysadmin dapat melakukan instalasi, deployment hingga melakukan update server. Sistem kerja yang dimiliki oleh ansible membutuhkan koneksi khusus berupa SSH. Ansible bekerja di koneksi SSH remote client yang ingin di deploy atau dilakukan otomasi. Pada ansible memerlukan inventory atau data server tujuan untuk dapat dilakukan otomasi. Pada penerapannya, ansible menggunakan playbook dan roles, dimana konfigurasi tersebut dalam format markup YAML dan environment variabel dapat ditulis dalam bentuk JSON. .

Ansible dirancang untuk memudahkan para sysadmin dan para pakar IT mengelola lingkungan server dengan mudah. ansible mengelola mesin dengan cara yang tidak biasa, tidak pernah bertanya cara melakukan upgrade daemon jarak jauh atau masalah karena tidak dapat mengelola sistem karena daemon sistem terhapus.

Ansible merupakan salah satu jenis Configuration Management Tools ayan dapat digunakan merubah proses infrastruktur manajemen dari program manual menjadi otomatis. Dalam zaman cloud kehadiran ansible membantu para sysadmin atau para devops dalam instalasi dan konfigurasi server dengan otomatis, oleh karena itu ansible menjadi satu platform yang digunakan untuk mengelola server – server.

# BAB III ANALISIS DAN PERANCANGAN SISTEM

## Analisis Kebutuhan

Pada tahap ini dilakukan analisis kebutuhan sistem, analisis kebutuhan sistem meliputi data yang digunakan, pembelajaran dari referensi yang sudah ada dan perangkat yang digunakan baik perangkat lunak maupun perangkat keras:

Tahapan pertama yang dilakukan dalam penelitian ini adalah mengidentifikasi permasalahan. Tahap ini merupakan tahap yang paling penting dalam penelitian karena jalannya penelitian didasarkan atas permasalahan yang terjadi. Setelah menentukan masalah yang terjadi, tahapan yang diperlukan selanjutnya adalah menentukan rumusan masalah dan tujuan yang ingin dicapai dalam penelitian. Pada penelitian ini identifikasi permasalahan dilakukan dengan menggunakan teknik observasi, dari teknik ini maka akan dapat diketahui mengenai keluhan – keluhan yang ada di lapangan.

Tahap kedua yang dilakukan dalam metodelogi penelitian ini adalah studi literatur. Studi literatur dilakukan dengan mengambil literatur – literatur pendukung dari jurnal – jurnal ilmiah, baik jurnal dalam negeri ataupun jurnal luar negeri dan dari beberapa buku. Dalam studi literatur ini, penulis mencari sumber terkait permasalahan – permasalahan yang perlu menjadi perbaikan dalam penelitian selanjutnya.

1. **Kebutuhan Fungsional Sistem**

Kebutuhan fungsional sistem mencakup fungsi – fungsi yang mampu dilakukan oleh sistem. Kebutuhan fungsional sistem pada penelitian ini ditunjukan pada tabel 3.2.

Tabel 3.1 Tabel Kebutuhan Fungsional

|  |  |  |
| --- | --- | --- |
| No. | Kebutuhan Fungsional | User |
| 1 | Melihat akurasi hasil klasifikasi | √ |
| 2 | Melihat detail hasil klasifikasi | √ |

1. **Kebutuhan Non Fungsional Sistem**

Adapun kebutuhan non fungsional yang harus dipenuh oleh sistem untuk melengkapi sistem secara keseluruhan, diantaranya sebagai berikut :

Tabel 3.2 Tabel Kebutuhan Non Fungsional

|  |  |  |
| --- | --- | --- |
| No. | Kebutuhan Non Fungsional | Keterangan |
| 1 | *Reliability* | Sistem mampu memberikan informasi yang sesuai, akurat, dan terpercaya |
| 2 | *User* *Friendly* | Sistem yang didukung dengan tampilan yang menarik dan mudah digunakan |

## Penentuan Proses – proses dalam Operasi Sistem

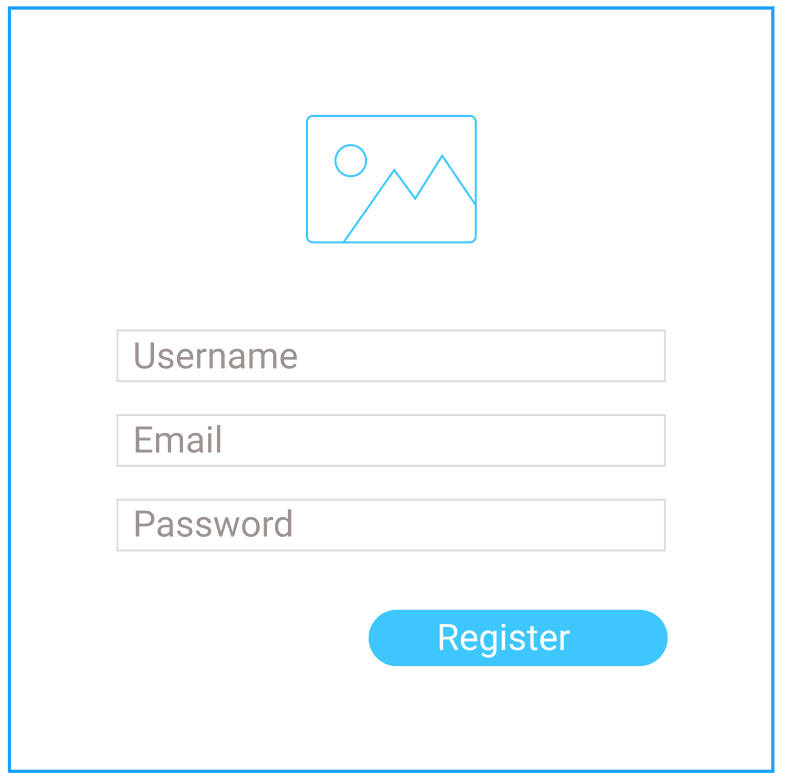
Setelah melakukan Teknik kajian pustaka pada tahap sebelumnya, secara garis besar proses-proses yang ada pada sistem klasifikasi *malware* adalah :

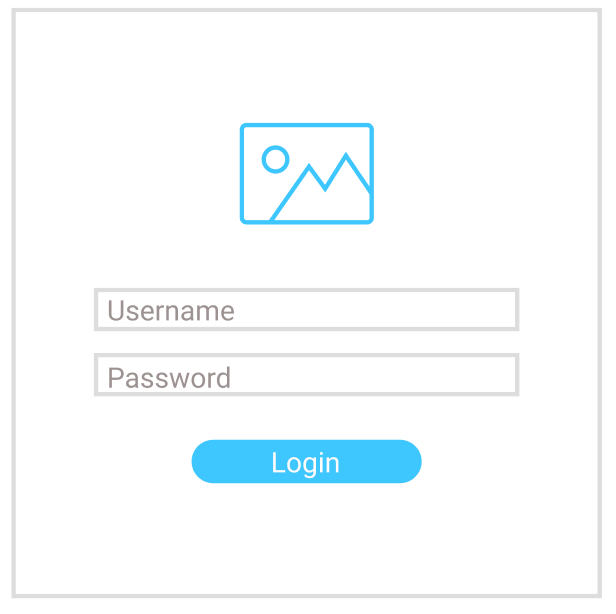
1. Proses *reverse* *engineering* sebagai *preprocessing* untuk mencari fitur yang akan digunakan dalam klasifikasi
2. Proses *input* dataset ke dalam sistem
3. Proses menghitung jarak dataset dengan menggunakan *jackard* *distance*
4. Proses klasifikasi dengan menggunakan algoritma KNN

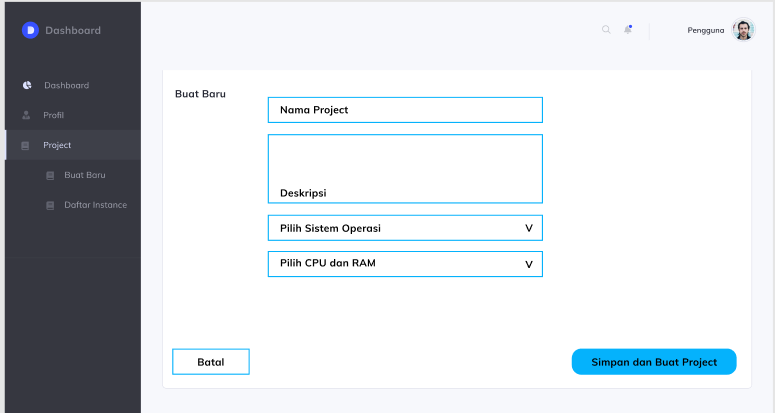
## Perancangan Antarmuka Sistem

Pada tahap ini dijelaskan perancangan antarmuka sistem dalam memanajemen layanan web berasis Platform as a Service (PaaS), pada sistem perancangan ini Sistem administrator dapat membuat dan mengembangkan websitenya pada sistem cloud computing.









## Skenario Pengujian Sistem

Bagian ini menjelaskan mengenai skenario pengujian sistem yang akan dilakukan pada Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) dengan API Openstack. Pengujian sistem ini dilakukan dengan menguji black box dan *Performance Testing.*

### BlackBox Testing

Black Box Testing atau dikenal sebagai “Behaviour Testing” merupakan suatu metode pengujian yang digunakan untuk menguji executable code dari suatu perangkat lunak terhadap perilakunya. Pendekatan Black Box Testing dapat dilakukan jika kita sudah memiliki executable code. Orang-orang yang terlibat dalam Black Box Testing adalah tester, end-user, dan developer.

Fokus dari pengujian ini ialah pada kebutuhan fungsional perangkat lunak, sehingga memungkinkan tester mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu untuk program. Kesalahan yang ditemukan dalam pengujian, nantinya dapat disimpulkan apakah kesalahan tersebut murni dikarenakan kesalahan dari aplikasi atau kesalahan implementasi dari tester.

Tabel .1. Tabel Pengujian Black Box

|  |  |  |
| --- | --- | --- |
| *Identifikasi* |  | |
| *Nama Kasus Uji* |  | |
| *Deskripsi* |  | |
| *Kondisi Awal* |  | |
| *Tanggal Pengujian* |  | |
| *Penguji* |  | |
| *Skenario* | | |
| *1.* |  | |
| *2.* |  | |
| *(Dst...)* |  | |
| ***Hasil Yang Diharapkan*** | ***Hasil Yang Didapatkan*** | ***Kesimpulan*** |
|  |  |  |

Untuk pengujian antarmuka pengguna atau rancangan skenario pengujian balck box dari sistem ini, dilakukan dua jenis pengujian yaitu pengujian secara happy path yaitu pengujian yang dilakukan dengan cara yang benar, serta pengujian secara alternative path yaitu mencoba segala kemungkinan yang mungkin terjadi pada sistem.

### Performance Testing

Teknik pengujian memvalidasi perilaku perangkat lunak terhadap teknik pengujian software dari sisi kecepatan. Kecepatan ini dalam konteks pengujian dalam mengukur waktu respon perangkat lunak ketika berada jumlah kerja yang berlebih yang biasa dikenal dengan beban kerja. Untuk memperlihatkan kecepatan sebenarnya sebuah perangkat lunak harus dilakukan pengujian performance testing.

Tujuan dari performance testing untuk memvalidasi kecepatan sebuah perangkat lunak terhadap kebutuhan sistem yang cepat. Secara umum harus mendefisinikan kombinasi waktu respons dan beban kerja.

# BAB IV HASIL DAN PEMBAHASAN

## Gambaran Umum Sistem

Sistem yang dibuat bertujuan membantu para pengembang aplikasi berbasis web, dengan meneriakan platform web server berbasis cloud computing, dengan menyediakan virtual machine berbasis cloud/ instance. Pengembang web dapat dengan mudah menjalankan sebuah web yang dibuat dengan mengakses alamat IP yang di berikan. Karena berbasis cloud, pengembang web dapat mengakses virtual machine yang di buat dimana saja secara online. Namun pada penetilian ini pengembang hanya dapat menggunakan jaringan Local untuk mengkases server cloud / private cloud.

Dalam implementasinya, server virtual yang diberikan pada pengembang web, secara otomatis sudah terinstall LAMP package. Namun, bila pengembang web ingin menggunakan package web server yang berbeda dapat melakukan instalasi manual dengan mengakses virtual machine menggunakan SSH.

## Lingkungan Perancangan dan Implementasi Sistem

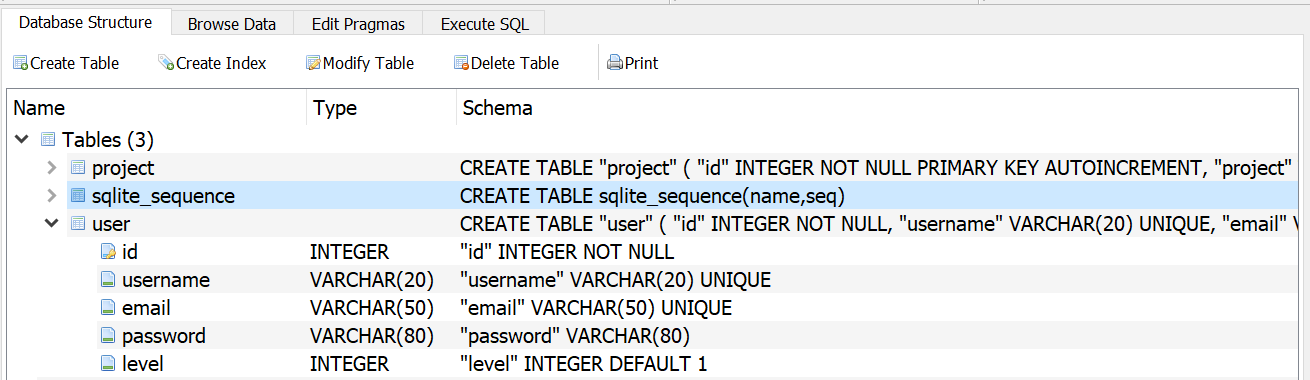
Sistem dirancang dan diimplementasikan pada sistem operasi Ubuntu dalam virtualisasi menggunakan Vmware, server dan sistem berjalan pada Laptop dengan spesifikasi i7, RAM 16 GB, dengan sistem operasi windows 10 64bit dan dengan support virtualisasi.

Sistem diimplementasikan menggunakan bahasa pemrograman python dengan *frameworks* *Flask*. Dalam perancangan dan implementasi sistem digunakan beberapa perangkat lunak untuk memenuhi semua kebutuhan sistem. Berikut adalah perangkat lunak yang digunakan :

1. *VMware*
2. *Visual Studio Codes*
3. *Putty*
4. *Jmeters*

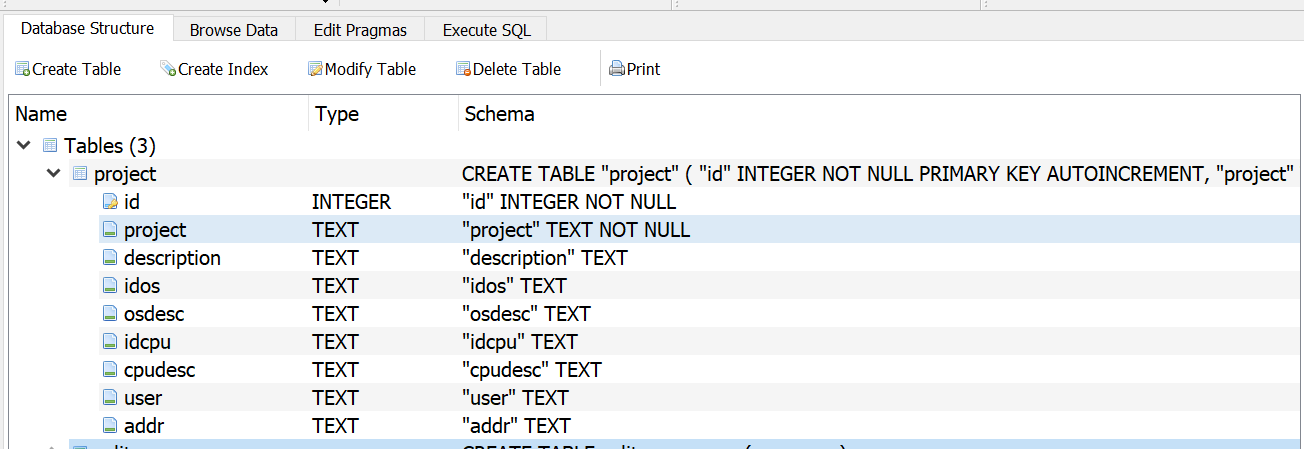
## Implementasi Basis Data

Pada sub bab ini akan dibahas mengenai tahap implementasi basis data pada sistem menginisiasi basis data dibutuhkan dua table, yaitu user atau developer aplikasi dan project.



Gambar 4.. Tabel basis data *user*

Pada tabel user berisi infromasi id yang digenerate secara otomatis, username, email, password dalam bentuk hash dan level pengguna.



Gambar 4.. Tabel basis data *project*

Selanjutnya pada tabel project berisi ID project, nama project, ID sistem operasi ID CPU, ID penyimpanan, User, dan IP Address server.

## Implementasi Antarmuka



Gambar 4.0.3 Tampilan Menu Awal

Gambar 4.7 merupakan tampilan menu awal dimana terdapat 2 pilihan yaitu Uji Dataset dan Deteksi. Menu Uji Dataset berfungsi untuk pengujian dataset sedangkan menu deteksi berfungsi untuk melakukan deteksi pada suatu *file*. Jika *user* memilih menu uji dataset, maka akan terlihat tampilan seperti pada Gambar 4.8. sedangkan jika *user* memilih menu deteksi maka akan terlihat tampilan seperti pada Gambar 4.10.



Gambar 4.0.4 Tampilan Hasil Uji Dataset

Gambar 4.8 merupakan tampilan hasil uji dataset. Dalam tampilan tersebut terdapat empat kolom yang berisikan jumlah *file* yang terdeteksi aplikasi *malware* atau aplikasi aman (*benign*). Kolom kiri atas merupakan kolom *True Benign*, dimana pada kolom ini ditampilkan jumlah dataset dengan kelas aman dan dideteksi sebagai aman. Kolom kiri bawah merupakan kolom *Benign as* *Malware*, dimana pada kolom ini ditampilkan jumlah dataset aman yang dideteksi sebagai *malware*. Kolom kanan bawah merupakan kolom *True Malware*, dimana pada kolom ini ditampilkan jumlah dataset *malware* yang dideteksi sebagai *malware*. Kolom kanan atas merupakan kolom *Malware as Benign*, dimana pada pada kolom ini ditampilkan jumlah dataset *malware* yang dideteksi aman. Pada bagian bawah kolom, ditampilkan akurasi dari klasifikasi yang telah dilakukan oleh sistem. Dibawah akurasi terdapat tombol detail. Tombol detail ini akan menampilkan detail dari klasifikasi seperti yang ditunjukkan pada gambar 4.9.



Gambar 4.0.5 Tampilan Menu Detail

Gambar 4.9. merupakan tampilan dari menu detail. Menu detail ini, berisikan informasi detail dari hasil klasifikasi pada menu uji dataset. Imformasi yang ada pada menu ini adalah nama *package*, kelas asli dari *package* yang akan diklasifikasikan, dan kelas prediksi yang merupakan hasil dari klasifikasi pada sistem.



Gambar 4.0.6 Tampilan Menu Deteksi

Gambar 4.10. merupakan tampilan menu deteksi. Terlihat pada gambar menu deteksi terdapat tombol “+” dan tombol proses. Tombol “+” berfungsi untuk *menginput* *file* yang akan dideteksi seperti yang ditunjukkan pada gambar 4.11. *File* yang dugunakan berupa *file* *Android Package* (APK), sedangkan tombol proses digunakan untuk melakukan proses klasifikasi. Ketika *user* menekan tombol proses, APK yang dipilih akan dikirimkan ke *server* untuk dilakukan proses ekstraksi fitur seperti pada gambar 4.12. sehingga menhasilkan fitur yang nantinya dapat digunakan untuk proses klasifikasi.



Gambar 4.0.7 Tampilan Pilih File

Setelah *user* menekan tombol “+” pada menu deteksi, *user* akan melihat tampilan seperti pada gambar 4.11. Pada Gambar 4.11. terlihat *user* mampu memilih berbagai jenis *file*, seperti *file* csv, txt, jpg, mp4, dan lain – lain. Namun *file* yang dapat digunakan untuk proses selanjutnya adalah *file* *Android Package* (APK).



Gambar 4.0.8 Tampilan Server

Gambar 4.12. merupakan tampilan dari *server* untuk menerima *file* APK yang dikirim oleh *client*. Setelah *file* diterima, *server* akan menjalankan proses ekstraksi fitur seperti yang ditunjukan pada gambar 4.13. *Server* dijalankan pada ip 0.0.0.0 dengan port 5000. Pada gambar 4.12. tampak *server* sedang *standby listening client*. Ketika *server* menerima APK dari *client*, maka akan terlihat tampilan seperti pada Gambar 4.13.



Gambar 4.0.9 Hasil Ekstraksi Fitur pada Server

Gambar 4.13. menunjukkan proses ekstraksi fitur yang dilakukan oleh *server*. Ketika *file* diterima, maka akan muncul pemberitahuan “file ditemukan” kemudian proses ekstraksi fitur akan dilakukan. Pada Gambar 4.13. terlihat beberapa *permissions* yang berhasil di ekstrak dari APK yang dikirim oleh *client*. Terlihat juga hasil ekstraksi fitur berupa *array* berisi 20 biner yang merupakan fitur APK dan sebuah string yang merupakan *package* *name* dari APK yang dikirim. Sama seperti pada *file* csv sebelumnya, 10 dijit biner pertama merupakan fitur *permission* dan 10 dijit biner selanjutnya merupakan fitur *API* *Calls*. *Array* tersebut selanjutnya akan dikirim kembali ke *client* guna dilakukannya proses klasifikasi.



Gambar 4.0.10 Tampilan Hasil Deteksi

Selanjutnya, *client* akan menerima data berupa 20 fitur yaitu 10 *permissions* dan 10 *API Calls* serta nama dari *package* *android* yang dikirim. 20 fitur tersebut selanjutnya akan diklasifikasikan dengan menggunakan KNN sehingga menghasilkan hasil seperti pada Gambar 4.14. Gambar 4.14. merupakan tampilan dari hasil deteksi. Hasil deteksi akan ditampilkan sebagai sebuah teks “Terdeteksi *malware*” jika hasil deteksinya adalah *malware* dan “Terdeteksi aman” jika hasil deteksinya adalah aman.



Gambar 4.0.11 Tampilan Hasil Deteksi



Gambar 4.0.12 Tampilan Hasil Deteksi



Gambar 4.0.13 Tampilan Hasil Deteksi



Gambar 4.0.14 Tampilan Hasil Deteksi



Gambar 4.0.15 Tampilan Hasil Deteksi



Gambar 4.0.16 Tampilan Hasil Deteksi



Gambar 4.0.17 Tampilan Hasil Deteksi



Gambar 4.0.18 Tampilan Hasil Deteksi









## Implementasi Program

Pada sub bab ini akan dibahas mengenai tahap implementasi dari rancangan yang sebelumnya telah dibuat ke dalam bahasa pemrograman.

1. **Proses Input File**

Tabel 4.1 Kode Input File

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7 | private void performFileSearch() {  Intent intent = new Intent(Intent.*ACTION\_OPEN\_DOCUMENT*); intent.addCategory(Intent.*CATEGORY\_OPENABLE*);  intent.setType("text/\*");   startActivityForResult(intent, *READ\_EXTERNAL\_DATA*); } |

Kutipan kode diatas merupakan fungsi dari performFileSearch dimana dalam fungsi tersebut digunakan Intent dalam membuka *file* manager dan memilih *file* bertipe teks.

Tabel 4.2 Kode Result dari Input File

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | @Override protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  if (requestCode == *READ\_EXTERNAL\_DATA* && resultCode == Activity.*RESULT\_OK*) {  if (data != null) {  Uri uri = data.getData();  if (uri != null){  String path = uri.getPath();  int sub = path.indexOf(":");  String simplePath = path.substring(sub + 1);  Toast.*makeText*(this, ""+simplePath, Toast.*LENGTH\_SHORT*).show();  klasifikasi(simplePath, uri);  }  }  } } |

Pada kutipan kode diatas merupakan result dari fungsi performFileSearch. Jika data yang dipilih dalam *file* manager tidak null, maka Uri pada data akan diambil, uri ini kemudian dirubah menjadi path untuk menunjukkan lokasi *file*. Selanjutnya dijalankan fungsi klasifikasi dengan *inputan* simplepath.

Tabel 4.3 Kode Mengirim File

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | RequestBody requestFile = RequestBody.*create*(MediaType.*parse*(getContentResolver().getType(uri)), file); RequestBody descBody = RequestBody.*create*(MediaType.*parse*("text/plain"), "Test"); Gson gson = new GsonBuilder()  .setLenient()  .create(); Retrofit retrofit = new Retrofit.Builder()  .baseUrl(Api.*BASE\_URL*)  .addConverterFactory(GsonConverterFactory.*create*(gson))  .build(); Api api = retrofit.create(Api.class); Call<MyResponse> call = api.uploadImage(requestFile, descBody);  public interface Api { String *BASE\_URL* = "http://192.168.1.7:5000/"; @Multipart  @POST("upload")  Call<MyResponse> uploadImage(@Part("file\"; filename=\"test\" ") RequestBody file, @Part("desc") RequestBody desc);  } |

Kutipan kode diatas merupakan kode untuk mengirim *file* dari *client* (*Android*) ke *server* pada linux. Pada kutipan kode diatas fungsi untuk mengirim *file* adalah “Call<MyResponse> call = api.uploadImage(requestFile, descBody);” dimana untuk mengirim *file* dibutuhkan RequestBody yang berisikan *Uniform Resource Identifier* (Uri) dari *file* dan *file* APK itu sendiri, serta descBody yang berisi deskripsi dari RequestBody. Selanjutnya *file* akan dikirim ke alamat *server* http://192.168.1.7 pada port 5000 dengan metode POST(“upload”).

1. **Proses Pada *Server***

Tabel 4.4 Kode Menerima File

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | @app.route('/upload', methods=['POST'])  def upload\_file():  print request.files  # checking if the file is present or not.  if 'file' not in request.files:  return "No file found"  file = request.files['file']  file.save("static/test")  rtn = androgrd1.main()  print rtn  resp = jsonify(  {  "data\_array" : rtn,  'message' : 'File successfully uploaded'  }  )  resp.status\_code = 200  return resp |

Kutipan kode diatas merupakan kutipan kode untuk menerima *file* pada *server*. Pada kutipan kode diatas, data yang diterima dengan menggunakan metode post “upload” akan disimpan pada direktori /static dengan nama *file* “test”. Selanjutnya fungsi androgrd1.main() akan dijalankan untuk melakukan ekstraksi fitur. Setelah fitur didapatkan data akan dikirim menggunakan json data yang dikirim berisikan array dan pesan.

Tabel 4.5 Kode Ekstraksi Fitur

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101 | def main():  input\_dir = './static'  file\_num = 1  tic = time.time()  user\_input = os.path.join(input\_dir, 'test')  assert os.path.exists(user\_input), "file tidak ditemukan pada, "+str(user\_input)  print("File Ditemukan !!")  a, d, dx = AnalyzeAPK(user\_input)  dx.create\_xref()  package\_name = a.get\_package()  permissions = a.get\_permissions()  val =["android.permission.READ\_PHONE\_STATE", "android.permission.RECEIVE\_BOOT\_COMPLETED", "android.permission.READ\_SMS", "android.permission.WRITE\_SMS", "android.permission.SEND\_SMS", "android.permission.RECEIVE\_SMS", "android.permission.CALL\_PHONE",  "android.permission.CHANGE\_WIFI\_STATE",  "android.permission.RESTART\_PACKAGES",  "android.permission.GET\_TASKS"]    ps = []  API\_freq = dict()  frequency(dx, API\_freq)  fs = open(os.path.join('./Documents/API-Kurang/coba',package\_name+'.txt'), "w")  datas = str(sorted(API\_freq.items(), key=lambda b:b[1], reverse=True))    print(permissions)  for i in range(0,len(val)):  if (check(permissions,val[i])):  ps.append(1)  else :  ps.append(0)  if 'Landroid/content/Intent;->getAction()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Ldalvik/system/DexClassLoader;->loadClass' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/TelephonyManager;->getDeviceId()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/TelephonyManager;->getLine1Number()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/TelephonyManager;->getNetworkOperator()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/TelephonyManager;->getSimSerialNumber()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/TelephonyManager;->getSimOperator()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/TelephonyManager;->getSubscriberId()' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/telephony/SmsManager;->sendTextMessage' in datas:  ps.append(1)  else :  ps.append(0)  if 'Landroid/location/LocationManager;->getLastKnownLocation' in datas:  ps.append(1)  else :  ps.append(0)  ps.append(package\_name)  print(ps)  print(time.time()-tic)  print("Done !!!")  return ps  file\_num += 1  def check(permissions, val):  for x in permissions :  if x == val :  return True |

Kutipan kode pada tabel 4.5. merupakan kode untuk melakukan ekstraksi fitur. Fungsi “a, d, dx = AnalyzeAPK(user\_input)” digunakan untuk menganalisa user\_input yang berupa APK. Pada fungsi “a, d, dx = AnalyzeAPK(user\_input)”, “a” digunakan untuk menganalisa *permissions* dengan fungsi “permissions = a.get\_permissions()” dan *package name* dengan fungsi “package\_name = a.get\_package()”, sedangkan “dx” untuk menganalisa API Calls dengan fungsi “frequency(dx, API\_freq)”. Selanjutnya *permission* dan *API Calls* yang telah terkumpul dibandingkan dengan list dari *permission* dan *API Calls* dan hasil perbandingannya disimpan dalam array ps[].

1. **Proses Pembacaan data**

Tabel 4.6 Kode Menerima Hasil Ekstraksi Fitur

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | @Override public void onResponse(Call<MyResponse> call, Response<MyResponse> response) {  Log.*i*("a", response.body().message);   Toast.*makeText*(getApplicationContext(), "" + response.body().message, Toast.*LENGTH\_SHORT*).show();  dialog.dismiss();   List<String> returnFromServer = response.body().data\_array;  List<String> returnAtribut = returnFromServer.subList(0,20);  List<String> returnPackageName = returnFromServer.subList(20,21);  List<Integer> ratrList = new ArrayList<>();  for(String s : returnAtribut) ratrList.add(Integer.*valueOf*(s));   Log.*e*("returnAtribut",""+returnAtribut);  Log.*e*("returnPackageName","" +returnPackageName);   fileTest[] fTest = new fileTest[1];  ArrayList<fileTest> atrFile = new ArrayList<>();  fTest[0] = new fileTest(Collections.*singletonList*(ratrList), Collections.*singletonList*( returnPackageName.get(0)));  atrFile.add(fTest[0]);  Log.*e*("atrFile",""+atrFile.get(0).atribut);   InputStream inputData = getResources().openRawResource(R.raw.*dataset1*);  readData readData = new readData(inputData);  ArrayList<dataSet> atributData = readData.readAtribut();   Log.*e*("atributData","" +atributData.get(50).atribut);   knn(atrFile, atributData); } |

Kutipan kode diatas merupakan kutipan kode untuk menerima hasil ekstraksi fitur yang berupa array. Fitur yang diterima selanjutnya disimpan dalam sebuah list string dengan menggunakan kode “List<String> returnFromServer = response.body().data\_array;”. Dari list returnFromServer selanjutnya akan dipisahkan menjadi 2 buah list dan disimpan dalam kelas fileTest. Proses selanjutnya adalah memanggil fungsi klasifikasi dengan kode “knn(atrFile, atributData);” dimana atrFile merupakan list dari kelas fileTest dan atributData merupakan list dari kelas dataSet. Kelas dataSet merupakan kelas yang berisi kumpulan data latih.

1. **Proses Perhitungan Jarak**

Tabel 4.7 Kode Jaccard Distance

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | public static double getDistance(dataTrain tr, dataTest ts){  int p = 0;  int q = 0;  int r = 0;  int s = 0;  double distance;  for(int j = 0; j < 20; j ++){  int dtr = tr.atribut.get(0).get(j);  int dts = ts.atribut.get(0).get(j);  if (dtr == 1){  if (dts == 1){  p++;  }  else {  q++;  }  }  else if (dtr == 0){  if (dts == 0){  s++;  }  else {  r++;  }  }}  distance = (double)(q+r)/(p+q+r); return distance;  } |

Kutipan kode diatas merupakan fungsi dari Jaccard Distance. Dimana *inputannya* berupa objek pada kelas dataTrain sebagai tr dan objek pada kelas dataTest sebagai ts. Seperti pada penjelasan di Bab II mengenai jaccard distance, untuk mendapatkan distance digunakan rumus (q+r)/(p+q+r) dimana p merupakan jumlah atribut antara tr dan ts yang bernilai positif atau 1, q merupakan atribut tr yang bernilai 1 dan 0 pada ts, sedangkan r merupakan atribut tr yang bernilai 0 dan 1 pada ts.

1. **Proses Klasifikasi**

Tabel 4.8 Kode KNN

|  |  |
| --- | --- |
| Baris | Kode |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | private void knn(ArrayList<fileTest> atrFile, ArrayList<dataSet> atributData) {  */\*DISTANCE\*/* List<Double> distance = new ArrayList<>(); int nilaiK = 3;  for (int k=0; k < 1; k++) {  for (int i = 0; i < 200; i++) {  double hasilJD = jaccard2.*getDistance*(atributData.get(i), atrFile.get(k));  distance.add(hasilJD);   atributData.get(i).distance = distance.get(i);   */\*COMPARE\*/* Collections.*sort*(atributData, dataSet.*dCompare*);  for (int i = 0; i < 200; i++) {  int kelasbaik = 0;  int kelasburuk = 0;  Log.*e*(TAG, "atribut Distance" + atributData.get(i).distance);  Log.*e*(TAG,"atribut Kelas"+atributData.get(i).kelas);  for (int j = 0; j < nilaiK; j++) {  if (atributData.get(i).kelas.get(0) == 0) {  kelasbaik++;  } else if (atributData.get(i).kelas.get(0) == 1) {  kelasburuk++;  }  }  Log.*e*(TAG,"Kelas Baik"+kelasbaik);  Log.*e*(TAG,"Kelas Buruk"+kelasburuk);  */\*PREDIKSI\*/* if (kelasbaik < kelasburuk) { tvNamaData.setText("TERDETEKSI MALWARE");  }  else if (kelasbaik > kelasburuk) { tvNamaData.setText("TERDETEKSI AMAN");  }  atributData.remove(distance);  distance.clear();  }  }  } |

Kutipan kode diatas merupakan fungsi dari KNN dimana dalam fungsi tersebut dibagi menjadi tiga bagian yaitu distance, compare, dan prediksi. Pada bagian distance, fungsi jaccard distance dipanggil dengan *inputan* atrFile dan atributData sebagai data uji dan data latihnya, fungsi jaccard distance memberi kembalian berupa jarak yang disimpan pada arraylist atributData. Selanjutnya pada bagian compare, arraylist atributData di urutkan dengan menggunakan fungsi sort dan di komparasikan atau dibandingkan. Kemudian dipilih jarak terkecil sebanyak nilaiK yang diimputkan. Pada bagian prediksi kelas dari setiap distance yang dipilih tadi dibandingkan, jika kelas baik lebih besar dari kelas buruk maka *file* yang diuji diklasifikasikan sebagai kelas baik (terdeteksi aman) dan sebaliknya jika kelas buruk lebih besar dari kelas baik maka *file* yang diuji diklasifikasikan sebagai kelas buruk (terdeteksi *malware*).

## Pengujian Sistem

Pada penelitian ini pengujian sistem “Sistem Manajemen Layanan Web Berbasis Platform as a Service (PaaS) Dengan API Openstack”, pengujian akan di lakukan dengan metode black box testing dan performance testing. Pengujian dilakukan dengan tujuan aplikasi yang dibuat dapat berjalan sesuai kebutuhan pengguna.

### **Black Box Testing**

Black Box Testing merupakan pengujian fungsional dari sistem yang dibuat. Tujuan dari pengujian fungsional adalah untuk memvalidasi perilaku perangkat lunak yang dibuat terhadap fungsionalitas kebutuhan para pengguna. Berikut merupakan tabel pengujian menggunakan black box :

Tabel 4.9 Hasil Pengujian ke-1

|  |  |  |  |
| --- | --- | --- | --- |
| **Kode Uji** | **Butir Uji** | **Teknik Pengujian** | **Hasil Pengujian** |
| OS1 | Pengguna baru melakukan pendaftaran untuk dapat masuk ke dalam sistem | Black Box | Diterima |
| OS2 | Pengguna melakukan login ke dalam sistem | Black Box | Diterima |
| OS3 | Admin dan pengguna memiliki antarmuka yang berbeda. | Black Box | Diterima |
| OS4 | Pengguna dapat membuat instance (server virtual) baru untuk pertama kali, | Black Box | Diterima |
| OS5 | Pengguna dapat membuat kembali instance (server virtual) baru. | Black Box | Diterima |
| OS6 | Pengguna dapat memilih sendiri kebutuhan spesifikasi prosessor, RAM, penyimpanan dan sistem Operasi yang akan di buat | Black Box | Diterima |
| OS7 | Pengguna dapat melihat list instance yang telah dibuat. | Black Box | Diterima |
| OS8 | Pengguna dapat mendownload SSH Key, untuk mengakses instance melalui SSH. | Black Box | Diterima |
| OS9 | Pengguna dapat mengakses instance melalui SSH dengan menggunakan SSHkey. | Black Box | Diterima |
| OS10 | Pengguna dapat melihat IP publik dari instance melalui web | Black Box | Tidak diterima |
| OS11 | Pengguna dapat melakukan instalasi paket Apache, PHP dan MySQL (LAMPP) secara otomatis. | Black Box | Tidak diterima |
| OS12 | Pengguna dapat mengganti kata sandi untuk masuk ke dalam sistem | Black Box | Diterima |
|  | Penggu tidak dapat masuk ke dashboard bila tidak melakukan login terlebih dahulu atau belum terdaftar dalam web admin | Black Box | Diterima |
| OS13 | Pengguna dapat keluar atau logout dari sistem pada menu yang sudah disediakan. | Black Box | Diterima |

### **Performance Testing**

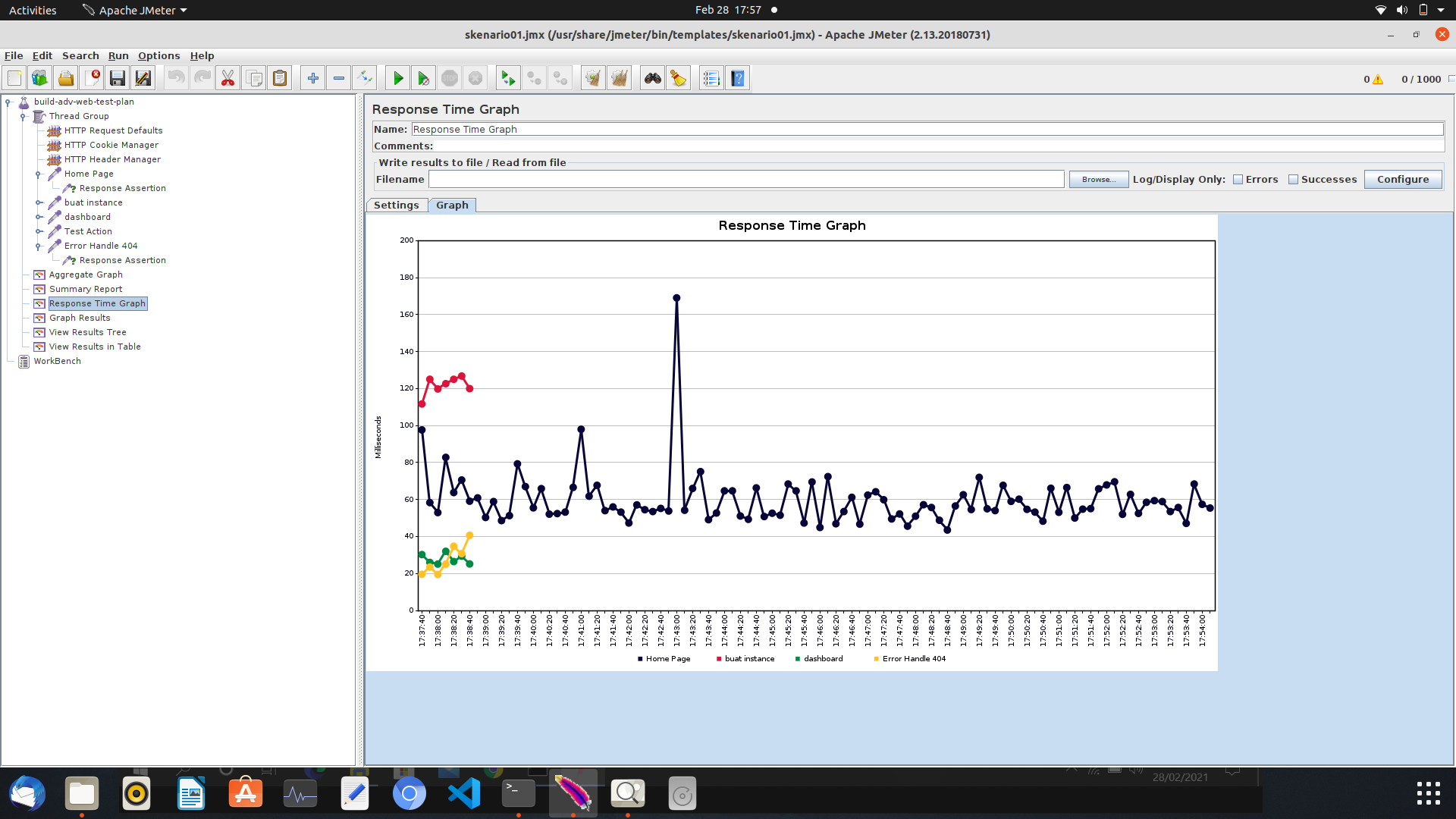
Pada pengujian *Performance Testing* merupakan lanjutan dari pengujian fungsionalitas sistem yang sudah dibuat. Performance Testing bertujuan memvalidasi “kecepatan” sistem yang dibuat. Kecepatan dalam hal ini peneliti melakukan pengujian dengan mengukur aspek waktu respon sistem. Peneliti melakukan pengujian kinerja dalam ruang lingkup yang mendekati ruang lingkup sebenarnya. Pengujian kinerja sistem ini dicapai dengan melakukan simulasi menggunakan perangkat lunak aplikasi *Jmeter*.

### 

Gambar 4.0.11 Grafik Hasil Pengujian ke-1

Tabel 4.10 Hasil Pengujian ke-1

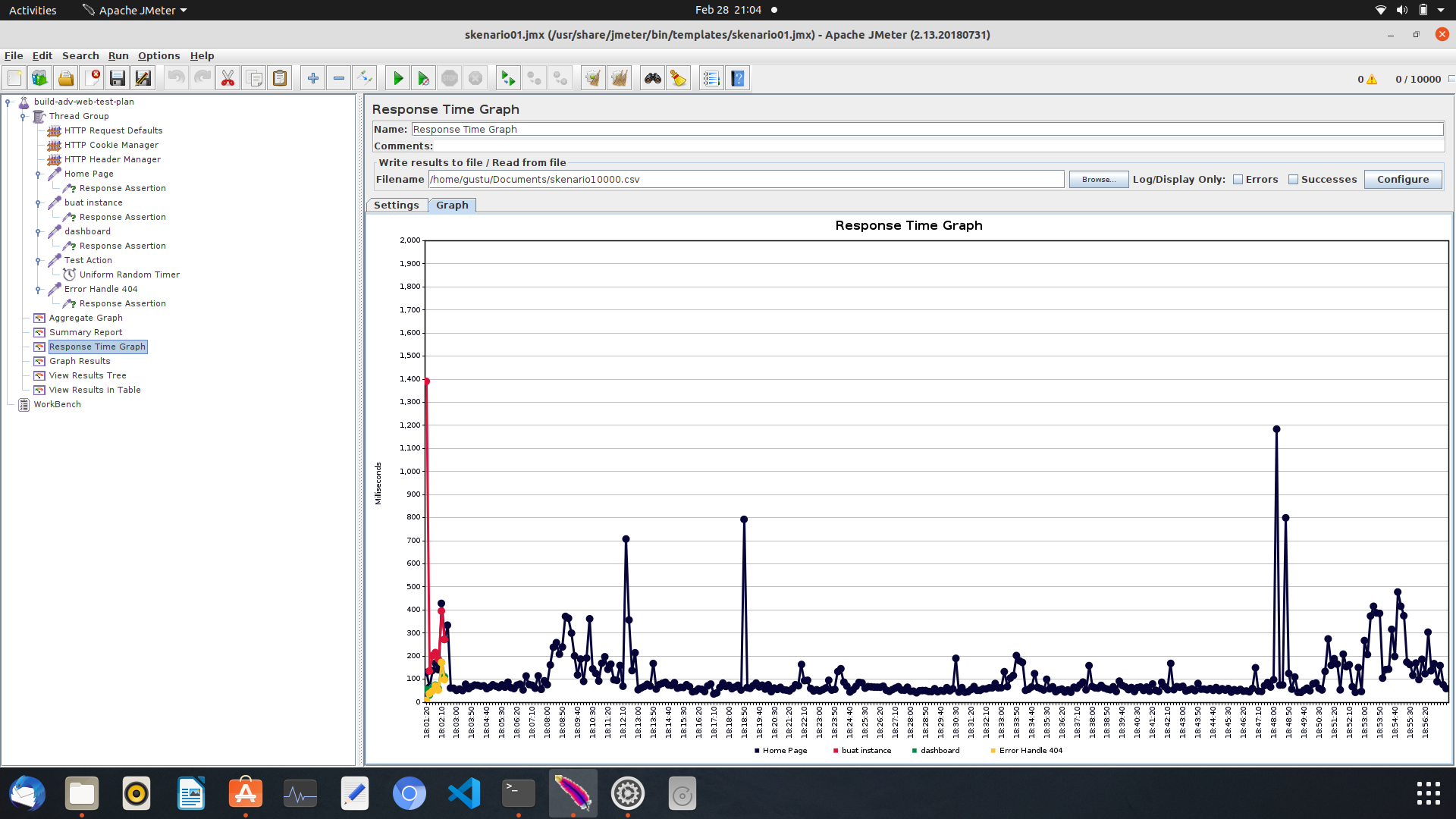
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Label | # Sample | Average | Min | Max | Std. Dev | Error % | Throughput | Received KB/sec | Send KB/sec | Avg. Bytes |
| Home Page | 314 | 463 | 49 | 2174 | 292.54 | 0.00% | 3.2/sec | 159.19 | 0.97 | 51151.0 |
| dashboard | 272 | 226 | 27 | 2971 | 262.31 | 0.00% | 4.5/sec | 54.81 | 1.42 | 12329.0 |
| Buat Instace | 268 | 539 | 241 | 1511 | 171.88 | 0.00% | 4.5/sec | 68.40 | 1.41 | 15626.0 |
| Page returning 404 | 261 | 208 | 20 | 2659 | 227.84 | 0.00% | 4.5/sec | 1.72 | 1.39 | 393.0 |
| total | 1115 | 264 | 20 | 2971 | 283.84 | 0.00% | 11.3/sec | 235.22 | 3.51 | 21284.7 |



Gambar 4.0.12 Grafik Hasil Pengujian ke-2

Tabel 4.11 Hasil Pengujian ke-2

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Label | # Sample | Average | Min | Max | Std. Dev | Error % | Throughput | Received KB/sec | Send KB/sec | Avg. Bytes |
| Home Page | 314 | 463 | 49 | 2174 | 292.54 | 0.00% | 3.2/sec | 159.19 | 0.97 | 51151.0 |
| dashboard | 272 | 226 | 27 | 2971 | 262.31 | 0.00% | 4.5/sec | 54.81 | 1.42 | 12329.0 |
| Buat Instace | 268 | 539 | 241 | 1511 | 171.88 | 0.00% | 4.5/sec | 68.40 | 1.41 | 15626.0 |
| Page returning 404 | 261 | 208 | 20 | 2659 | 227.84 | 0.00% | 4.5/sec | 1.72 | 1.39 | 393.0 |
| total | 1115 | 264 | 20 | 2971 | 283.84 | 0.00% | 11.3/sec | 235.22 | 3.51 | 21284.7 |



Gambar 4.0.13 Grafik Hasil Pengujian ke-3

Tabel 4.12 Hasil Pengujian ke-3

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Label | # Sample | Average | Min | Max | Std. Dev | Error % | Throughput | Received KB/sec | Send KB/sec | Avg. Bytes |
| Home Page | 314 | 463 | 49 | 2174 | 292.54 | 0.00% | 3.2/sec | 159.19 | 0.97 | 51151.0 |
| dashboard | 272 | 226 | 27 | 2971 | 262.31 | 0.00% | 4.5/sec | 54.81 | 1.42 | 12329.0 |
| Buat Instace | 268 | 539 | 241 | 1511 | 171.88 | 0.00% | 4.5/sec | 68.40 | 1.41 | 15626.0 |
| Page returning 404 | 261 | 208 | 20 | 2659 | 227.84 | 0.00% | 4.5/sec | 1.72 | 1.39 | 393.0 |
| total | 1115 | 264 | 20 | 2971 | 283.84 | 0.00% | 11.3/sec | 235.22 | 3.51 | 21284.7 |

Gambar 4.0.14 Grafik Hasil Pengujian ke-4

Tabel 4.13 Hasil Pengujian ke-5

Gambar 4.0.15 Grafik Hasil Pengujian ke-5

## Analisa

Gambar 4.0.16 Grafik Akurasi

.

# BABI V SIMPULAN DAN SARAN

## Simpulan

Berdasarkan penelitian yang telah dilakukan, dapat diambil beberapa kesimpulan sebagai berikut :

* 1. Untuk mendeteksi *malware* dengan menggunakan K Nearest Neighbor, tahap pertama yang dilalui adalah *preprocessing* yaitu pencarian fitur berupa *Permissions* dan *API Calls* yang nantinya akan digunakan dalam proses klasifikasi. Selanjutnya proses klasifikasi dimana pertama – tama kita harus menentukan jarak dari setiap dataset menggunakan Binary Jackard Distance karena bentuk fitur yang digunakan berupa binary 1 dan 0, kemudian menentukan kelas sesuai dengan jumlah K yang *diinputkan*.
  2. Berdasarkan hasil pengujian sebanyak lima kali, didapatkan rata – rata akurasi yaitu k = 1 sebesar 82 %, k = 3 sebesar 85,8 %, k = 5 sebesar 85,8 %, k = 7 sebesar 84,4 %, k = 9 sebesar 83,2 %, k = 11 sebesar 83,4 %, k = 13 sebesar 84,2 %, k = 15 sebesar 84,4 %, k = 17 sebesar 85,8 %, k = 19 sebesar 85 %.
  3. Dari rata – rata pengujian tertinggi sebesar 85,8 % dapat disimpulkan nilai k terbaik yang dapat digunakan dalam sistem “Pendeteksian Malware Android dengan Pendekatan Permissions dan API Call menggunakan K-Nearest Neighbor (KNN)” yaitu k = 3 dikarenakan proses klasifikasi dengan k = 3 lebih cepat dibandingkan dengan k = 5 dan k = 17.

## Saran

Untuk pengembangan sistem lebih lanjut, penulis ingin memberikan beberapa saran sebagai berikut :

* + 1. Dapat menggunakan fitur lain dalam proses klasifikasi *malware* untuk mendapatkan nilai akurasi yang lebih baik.
    2. Dapat menggunakan algoritma lain atau gabungan dari beberapa algoritma untuk mendapatkan nilai akurasi yang lebih baik.

# DAFTAR PUSTAKA

Xia, Q., Lan, Y., & Xiao, L. (2015). A Heuristic Adaptive Threshold Algorithm on IaaS Clouds.

Anggeriana, H. (2011). *Cloud Computing.*

Ansible. (t.thn.). *ansible.com*. Diambil kembali dari http://docs.ansible.com/ansible/latest/index.html: http://docs.ansible.com/ansible/latest/index.html

Collings, T., & Kurt , W. (2015). *Duties of the System Administrator. In Red Hat Linux Networking System Administrator (chap. 1).*

Corporation, Exabyte. (2004). *The Basic Backup Guide.* cororado.

Doty, S. (2008). *Python Basics.*

Gerald D. Everett, R. M. (2007). *Software Testing: Testing Across the Entire Software Development Life Cycle 1st Edition.* Canada: IEEE Press.

Lei Xiaojiang, S. Y. (2013). The Design and Implementation of Resource Monitoring for Cloud Computing Service Platform.

Liyun Zuo, L. S. (2015). A Dynamic Self-adaptive Resource-Load Evaluation Method in Cloud Computing.

Masse, M. (2012). REST API. Dalam *REST API Design Rulebook* (hal. 5). America: O’Reilly Media, Inc.

Mulyana, E. (2017, 10 05). *Pengantar Openstack*. Diambil kembali dari https://eueung.gitbooks.io: https://eueung.gitbooks.io/buku-komunitas-sdn-rg/content/index.html

Nicole Ng, H. C. (2011). An Adaptive Threshold Method to Address Routing Issues in Delay-Tolerant Networks.

Saleh, Y. W. (2013). Adaptive Resource Management for Service Workflows in Cloud Environments.

Sosinsky, B. (2011). *Cloud Computing Bible.* Canada: Wiley Publishing, Inc.

# LAMPIRAN

**Data Permissions**