PERANCANGAN APLIKASI PENGELOLAAN PERANGKAT JARINGAN DENGAN PEMROGRAMAN PYTHON BERBASIS WEB (STUDI KASUS: SMKN 3 KOTA BEKASI)

Jadid Alif Ramadhan, Agung Susilo Yuda Irawan, Arip Solehudin
Program Studi Informatika, Fakultas Ilmu Komputer
Universitas Singaperbangsa Karawang, Jl. HS. Ronggo Waluyo, Karawang, Indonesia
jadid.alif19025@student.unsika.ac.id

ABSTRAK

Perkembangan teknologi informasi dan komunikasi kini sudah semakin maju dan pesat, Salah satunya yakni jaringan komputer. Dibalik rumitnya jaringan komputer ada sebuah profesi yaitu administrator jaringan yang memiliki pekerjaan penting seperti melakukan konfigurasi perangkat jaringan, pemeliharaan jaringan, memonitoring serta pemecahan masalah pada jaringan. Biasanya administrator jaringan mengkonfigurasi perangkat jaringan secara manual yaitu melakukan satu persatu konfigurasi pada setiap perangkat, dan pastinya membutuhkan waktu banyak jika harus menyelesaikan konfigurasi pada seluruh perangkat jaringan yang ada. Selain itu, administrator jaringan yang melakukan konfigurasi banyak perangkat jaringan juga akan memberi peluang adanya kesalahan konfigurasi. Untuk mengatasi permasalahan tersebut, pada penelitian ini dirancang sebuah sistem yang bertujuan untuk mempermudah admin jaringan dalam mengelola beberapa perangkat jaringan melalui satu website. Metode yang digunakan dalam penelitian ini yaitu *Software Development Life Cycle* (SDLC) model *waterfall*. Hasil penelitian ini telah dibangun sistem pengelolaan perangkat jaringan berbasis web menggunakan pemrograman Python dengan *library* Paramiko dan *framework web* Django. Aplikasi telah diuji coba dan berhasil melakukan konfigurasi di perangkat jaringan secara bersamaan pada aplikasi simulasi jaringan GNS3.

Kata kunci: *Django, Paramiko, Python, Pengelolaan Perangkat Jaringan*

1. PENDAHULUAN

Perkembangan teknologi informasi serta komunikasi saat ini sudah semakin maju dan pesat, Salah satu bidang teknologi yang berkembang pesat yakni jaringan komputer. hampir semua lembaga, sekolah maupun perusahaan mempergunakan jaringan komputer untuk dapat saling berkomunikasi. Dibalik rumitnya jaringan komputer ada sebuah profesi yaitu administrator jaringan yang memiliki tanggung jawab untuk melakukan konfigurasi terhadap perangkat jaringan, pemeliharaan jaringan secara rutin, memonitoring jaringan, serta berperan dalam analisis jaringan atau pemecahan masalah pada jaringan.

Administrator jaringan memiliki pekerjaan yang cukup penting dan beresiko, Kesalahan pada konfigurasi jaringan dapat berakibat gangguan pada jaringan bahkan dapat gagal terhubungnya semua jaringan, perangkat tidak dapat terhubung ke internet, perangkat tidak dapat terhubung satu sama lain.

Administrator jaringan biasanya mengkonfigurasi perangkat jaringan secara manual yaitu melakukan satu persatu konfigurasi pada setiap perangkat menggunakan *remote SSH* [1]. Namun, akan melelahkan jika harus konfigurasi seluruh perangkat jaringan pada skala yang besar seperti pada perusahaan, kantor, perguruan tinggi, bahkan sekolah. Hal ini dinilai tidak efisien, karena konfigurasi manual dapat menimbulkan kesalahan dalam konfigurasi. Dan ketika perangkat jaringan yang ingin dikonfigurasi terdiri dari ratusan bahkan ribuan perangkat, sehingga administrator jaringan membutuhkan waktu yang sangat lama.

Berdasarkan penjelasan diatas maka diperlukan sebuah sistem yang bertujuan untuk mempermudah pengelolaan perangkat jaringan serta konfigurasi secara terpusat melalui sebuah website. Dibuatnya sistem pengelolaan perangkat jaringan berbasis website pada SMK Negeri 3 Bekasi ini diharapkan mampu membantu admin jaringan dalam hal efisiensi waktu dan tenaga untuk mengelola perangkat jaringan yang ada di sekolah.

2. TINJAUAN PUSTAKA

2.1. Jaringan Komputer

Jaringan komputer atau network adalah sekumpulan perangkat jaringan (*network devices*) yang saling terhubung satu sama lain untuk melaksanakan proses komunikasi dan pertukaran data menggunakan media kabel maupun nirkabel sehingga dapat berjalan dengan baik [2].

2.2. Python

Python adalah salah satu bahasa pemrograman yang sangat populer di dunia. Dirilis pertama kali pada 1990-an dan sekarang digunakan untuk membuat jutaan aplikasi, permainan, hingga situs web. Python merupakan bahasa pemrograman komputer berbasis teks. Dengan penulisan menggunakan kata-kata dalam bahasa Inggris, tanda baca, angka dan simbol. Ini membuat kode Python lebih mudah ditulis, dibaca, dan dipahami [3].

Python termasuk dalam bahasa pemrograman tingkat tinggi berorientasi objek yang bersifat dinamis. Struktur data tingkat tinggi bawaan dikombinasikan dengan pengetikan dinamis membuat python sangat

cocok untuk pengembangan aplikasi yang cepat. Python umumnya digunakan sebagai bahasa skrip untuk menghubungkan komponen yang ada [4].

2.3. Django

Django merupakan *framework* web yang memakai bahasa pemrograman Python, framework Django menunjang untuk perancangan *website* dengan metode rapid *development*. Django digunakan untuk menyederhanakan pengembangan situs web dan *database* yang rumit. Salah satu kelebihan *Django* yakni memperkenalkan *Object Relational Mapper* (ORM) sehingga kueri tidak perlu disesuaikan saat *database* yang digunakan terjadi perubahan [5].

Framework Django merupakan salah satu framework yang banyak diminati oleh *developer* terutama untuk merancang sebuah aplikasi berbasis *website*. Django adalah sebuah kerangka kerja yang menggunakan bahasa pemrograman tingkat tinggi Python, yang memudahkan pekerjaan *developer* karena bahasa yang dipakai mudah dimengerti dibandingkan dengan bahasa pemrograman lainnya. Salah satu keunggulan dari *framework* django yakni proses pengembangan aplikasi lebih cepat dari *framework* lain dan merupakan salah satu *framework* paling multifungsi yang dipergunakan oleh banyak sekali organisasi [6].

2.4. Paramiko

Paramiko adalah *library* python yang merupakan implementasi dari protokol SSH versi 2 yang mempersediakan fungsionalitas *client* dan *server*. Semua perangkat jaringan yang bisa dikonfigurasi dengan SSH juga dapat menggunakan *library* ini [7]. *Library* Paramiko juga dapat langsung menjalankan *script* yang berisi konfigurasi perangkat, lalu selanjutnya akan dikirimkan pada perangkat yang akan dikonfigurasi dengan menggunakan protokol SSH [8].

2.5. SSH *Remote Access*

Secure Shell (SSH) merupakan protokol jaringan yang memungkinkan transmisi data melalui saluran yang aman antara dua perangkat jaringan. SSH juga mendukung fungsi pengiriman file melalui *Secure File Transfer Protocol* (SFTP) atau *Secure Copy* (SCP). SSH menggunakan model komunikasi client-server. *Port default* untuk ssh yaitu *port* 22, yang sudah ditetapkan sebagai jalur untuk *server* SSH. Klien SSH biasanya dipergunakan untuk membuat koneksi antara klien dan *server* SSH sehingga dapat dikontrol dari jarak jauh [9].

SSH merupakan protokol administratif yang memungkinkan pengguna untuk mengakses dan mengubah berbagai pengaturan dan file di *server*. koneksi SSH terenkripsi menggunakan beberapa teknik termasuk enkripsi simetris, enkripsi asimetris, dan hashing. Ketiganya adalah teknologi enkripsi yang menjamin koneksi terenkripsi.

Cara kerja SSH yakni, Klien SSH menangani koneksi dan menggunakan kunci enkripsi untuk mengautentikasi dan mengidentifikasi server SSH. Kemudian koneksi yang tertaut dienkripsi menggunakan *symmetric encryption* dan hashing algorithm. Tujuan dari metode enkripsi ini adalah untuk memastikan kerahasiaan dan integritas pertukaran data antara klien dan *server*.

2.6. GNS3

GNS3 merupakan aplikasi simulasi jaringan komputer berbasis *Graphical User Interface* (GUI). GNS3 dapat mensimulasikan topologi jaringan yang kompleks dan rumit karena memakai sistem operasi dari perangkat jaringan, tetapi untuk menggunakannya harus menyediakan sistem operasinya sendiri [10].

Aplikasi GNS3 ini dapat mensimulasikan berbagai jenis perangkat jaringan seperti *router*, *switch*, dan *server*. Ada beberapa *vendor* perangkat jaringan seperti Mikrotik dan Cisco, Bahkan terdapat sistem operasi seperti Windows, Linux dan lain-lain .

2.7. Software Development Life Cycle

Software Development Life Cycle (SDLC) adalah sebuah metodologi yang umum untuk mengembangkan sebuah sistem informasi. SDLC mencakup beberapa tahapan mulai dari perencanaan, analisis, desain, implementasi sampai dengan pemeliharaan. Konsep SDLC ini menjadi dasar dari berbagai model pengembangan perangkat lunak sebagai bentuk kerangka kerja untuk pengendalian dan perencanaan perancangan sistem [11].

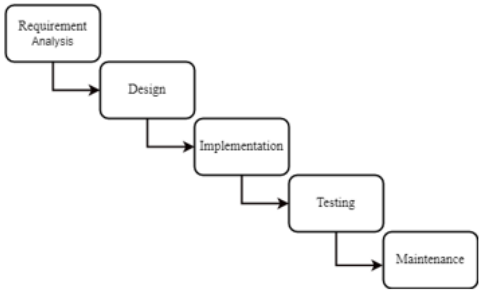
Model SDLC yang umum dipergunakan antara lain *waterfall, agile, prototyping*, dan spiral. Model pada SDLC memiliki kelebihan dan kekurangannya tersendiri, serta memiliki karakteristik masing masing dalam membangun sebuah sistem atau perangkat lunak.

2.8. *Waterfall*

Model *waterfall* adalah model SDLC tertua yang digunakan untuk proses pengembangan perangkat lunak. *Waterfall* merupakan jenis model pengembangan *classic life cycle* yang menekankan langkah-langkah berurutan dan sistematis. Model ini menyerupai air terjun yang dimana setiap tahapan dilaksanakan secara berurutan dari atas ke bawah, model ini menggambarkan proses "*Linear Sequential Model*" yang berarti bahwa setiap tahapan proses pengembangan tidak akan dimulai sampai tahap sebelumnya selesai. Selain itu, proses tidak dapat mengulangi langkah sebelumnya [12].

Model *waterfall* cocok untuk mengembangkan sistem atau *software* yang memiliki sifat generik, yang berarti bahwa sistem dapat mengidentifikasi semua persyaratannya terlebih dahulu dengan spesifikasi umum, dan sesuai untuk proyek akhir atau tesis yang tujuannya adalah membangun sistem yang dari awal mengumpulkan persyaratan sistem yang akan dimulai sesuai dengan topik penelitian yang dipilih hingga sistem tersebut diuji [11].

Model *waterfall* terdiri dari 5 tahapan, yaitu *Requirements analysis, Design, Implementation, Testing, dan Maintenance*. Berikut adalah tahapan dari model *waterfall* menurut [13].



Gambar 1. Alur *Waterfall*

2.9. *Use Case Diagram*

Use case diagram adalah salah satu bagian dari diagram *Unified Modeling Language* (UML) dan dapat didefinisikan sebagai sebuah diagram proses yang menunjukkan setiap hal atau setiap aktivitas yang bisa dilakukan oleh aktor dalam menyelesaikan pekerjaan. *use case diagram* biasanya dibuat pada tahapan analisis kebutuhan sistem [14].

2.10. *Activity Diagram*

Activity diagram juga merupakan salah satu bagian dari diagram *Unified Modeling Language* (UML). *Activity diagram* dapat didefinisikan sebagai suatu diagram yang menggambarkan alur kerja atau *workflow* pada sebuah sistem ataupun proses bisnis. *Activity diagram* tidak hanya menggambarkan aktivitas dari aktor saja, tetapi juga menggambarkan aktivitas yang dilakukan oleh sistem [14].

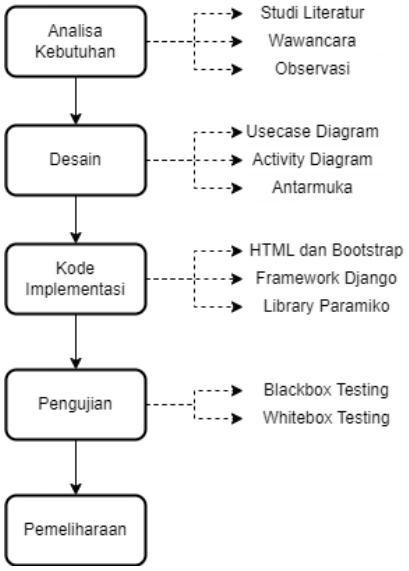
2.11. *Black-box Testing*

Pengujian *black-box* adalah sebuah metode pengujian perangkat lunak yang berfokus pada kebutuhan fungsional dari sebuah perangkat lunak atau sistem yang dibangun. Tujuan dari dilakukannya pengujian *black-box* ini yaitu untuk menunjukan cara kerja fungsi dari sistem ketika diberikan inputan, apakah hasil yang diharapkan sudah sesuai atau belum. *Black-box* testing ini dapat dipergunakan sebagai pelengkap untuk menguji hal – hal diluar cakupan *white-box* testing [15].

2.12. *White-box Testing*

Pengujian *white-box* yakni merupakan suatu metode pengujian perangkat lunak yang mempergunakan penjelasan struktur *control* sebagai bagian dari *component-level design* untuk membuat uji kasus. Pengujian *white-box* ini dapat memperlihatkan kesalahan dari implementasi suatu sistem.dengan cara menganalisa kesalahan kode dari program yang telah dibuat, apabila hasil yang didapat tidak sesuai maka akan dikompilasi dan dilakukan pengecekan ulang hingga sesuai dengan yang diharapkan [16].

3. METODE PENELITIAN



Gambar 2. Tahapan Penelitian

Metodologi penelitian untuk perancangan sistem menggunakan *Software Development Life Cycle* (SDLC) model *waterfall* yang terdiri dari 5 tahap, yaitu *Requirements analysis, Design, Implementation, Testing,* dan *Maintenance*. Alur atau tahapan dari penelitian yang dilakukan dapat dilihat pada Gambar 3.1 berikut ini:

3.1. Analisis Kebutuhan

Pada langkah ini, dilakukan analisis kebutuhan untuk kebutuhan pengguna terhadap sistem yang akan dibuat dan mengidentifikasi permasalahan di SMKN 3 Kota Bekasi sehingga akan diperoleh solusinya. Metode pengumpulan data yang digunakan dalam analisis kebutuhan adalah metode studi literatur, wawancara dan observasi.

3.2. Desain

Pada tahapan berikutnya yaitu melakukan desain sistem, desain dibuat dari hasil analisis pada langkah sebelumnya, tujuannya untuk memberikan gambaran dalam implementasi sistem. Desain yang dibuat berupa UML diagram yang terdiri dari *Use case diagram* dan *Activity diagram*, serta Rancangan antarmuka atau *user interface.*

3.3. Kode/Implementasi

Melakukan pengkodean untuk membangun *website* memakai aplikasi Visual Studio Code, pembuatan *frontend* menggunakan HTML dan Bootstrap 5, serta bahasa pemrograman python untuk *framework* web Django dan *library* paramiko untuk dapat menghubungkan perangkat jaringan.

3.4. Pengujian

Tahap berikutnya adalah pengujian, tahap pengujian aplikasi menggunakan metode *white-box* dan *black-box testing* yang digunakan untuk mengamati hasil dan fungsionalitas sistem yang telah

dibuat, apakah sudah sesuai dengan kebutuhan atau kriteria yang diinginkan.

3.5. Pemeliharaan

Langkah terakhir adalah pemeliharaan atau maintenance. Setelah *website* berhasil dibuat, selanjutnya akan dilakukan pengelolaan sistem supaya tetap berjalan dengan seharusnya tanpa ada kendala.

4. HASIL DAN PEMBAHASAN

4.1. Analisis Kebutuhan

Berikut adalah hasil dari tahapan analisis kebutuhan dalam pengembangan sistem.

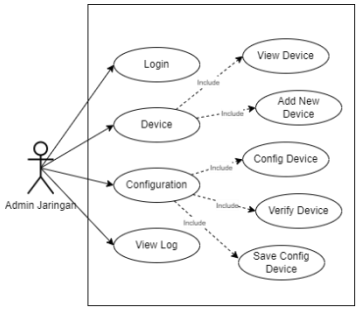
a. Studi Literatur
   Mempelajari dari buku, paper, jurnal dan artikel-artikel penelitian terdahulu. Sumber data berupa pengetahuan mengenai perancangan website dengan *framework* django dan penggunaan *library* python yaitu paramiko sebagai implementasi protokol SSHv2 untuk menyambungkan ke perangkat jaringan.
b. Wawancara
   Wawancara dilaksanakan dengan cara mendatangi sekolah dan bertemu Pihak ICT dari SMKN 3 Bekasi, wawancara dilakukan dengan berkomunikasi langsung, berdiskusi dan bertanya terkait apa saja yang menjadi kendala serta permasalahan dalam jaringan pada sekolah tersebut.
c. Observasi
   Observasi dilakukan dengan cara mengamati topologi dan perangkat jaringan yang digunakan di sekolah, kemudian informasi yang diperoleh akan dijadikan data untuk kebutuhan pada sistem.

4.2. Desain

Pada tahapan ini dilakukan desain sistem dari hasil analisis sebelumnya.

A. Use Case Diagram

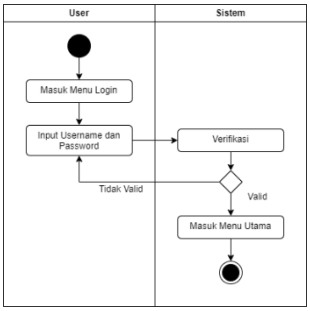
Pada Gambar 3 berikut merupakan *Use Case diagram* yang menggambarkan sistem pengelolaan perangkat jaringan pada SMKN 3 Kota Bekasi.



Gambar 3. Use Case Diagram

B. Activity Diagram

1) Activity Diagram Login

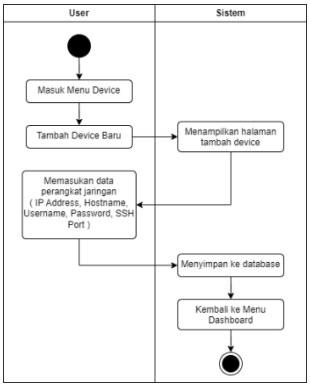


Gambar 4. Activity Diagram Login

Gambar 4 menampilkan *activity diagram* admin dalam melakukan *login* kedalam sistem. Setelah Admin memasukan *username* dan *password*, lalu sistem akan mengecek validasi dari inputan, apabila *username* atau *password* yang diinputkan benar maka akan menampilkan halaman *dashboard*. Jika *username* atau *password* yang diinputkan salah maka akan kembali ke halaman *login*.

2) Activity Diagram Add New Device

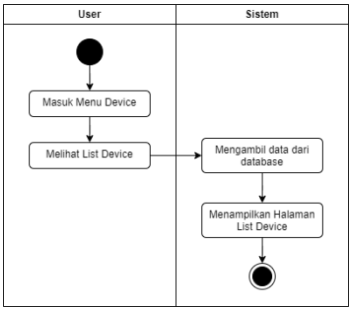
Pada Gambar 5 menampilkan *activity diagram* admin ketika menambahkan perangkat jaringan baru.



Gambar 5. Activity Diagram Add New Device

3) Activity Diagram Device List

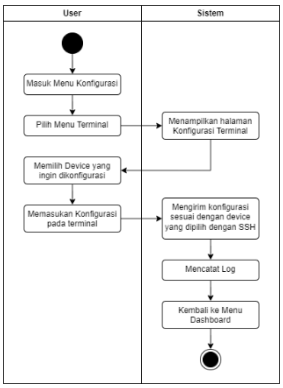
Pada Gambar 6 merupakan *activity diagram* admin dalam mengelola *perangkat* jaringan dengan menampilkan halaman *list device.*



Gambar 6. Activity Diagram Device List

4) Activity Diagram Configuration Device

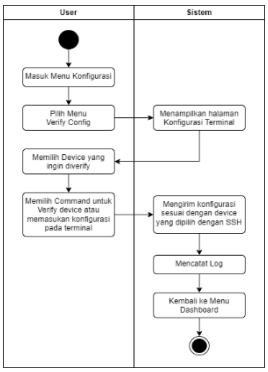
Pada Gambar 7 merupakan *activity diagram* admin ketika melakukan konfigurasi ke perangkat jaringan.



Gambar 7. Activity Diagram Configuration

5) Activity Diagram Verify

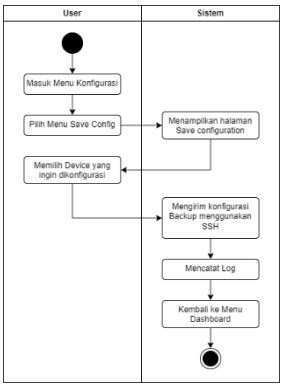
Pada Gambar 8. menampilkan activity diagram admin ketika melakukan verifikasi konfigurasi pada perangkat jaringan.



Gambar 8. Activity Diagram Verify

6) Activity Diagram Save Configuration

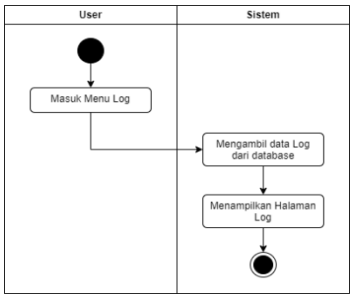
Pada Gambar 9 menampilkan *activity diagram* admin ketika ingin menyimpan konfigurasi pada perangkat jaringan.



Gambar 9. Activity Diagram Save Configuration

7) Activity Diagram Log

Pada Gambar 10 menampilkan *activity diagram* admin ketika ingin melihat log.



Gambar 10. Activity Diagram Log

C. Desain User Interface

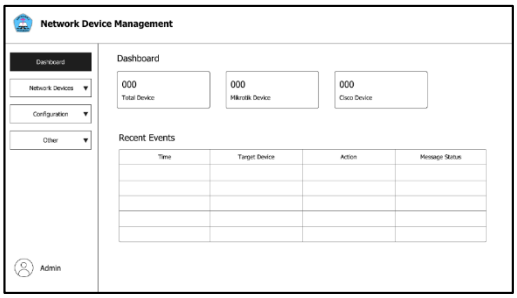
1) Desain User Interface Login

Pada Gambar 11 merupakan tampilan perancangan desain awal saat *login*, terdapat masukan untuk *username* dan juga *password*.

Gambar 11. Desain UI Login

2) Desain User Interface Dashboard

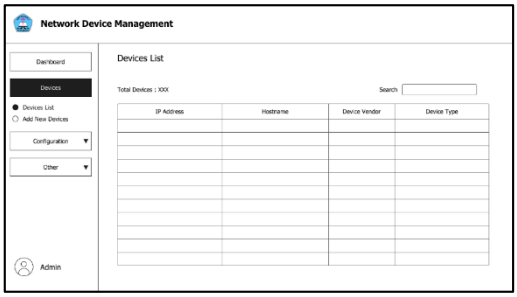
Pada Gambar 12 merupakan tampilan *perancangan* desain awal *dashboard*, *dashboard* merupakan halaman utama pada sistem, terdapat total jumlah *device* serta *recent events*, dan menu navbar dan admin pada sebelah kiri.



Gambar 12. Desain UI *Dashboard*

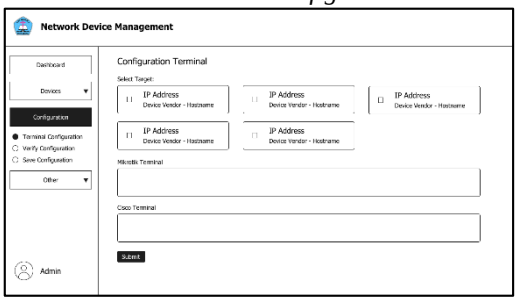
3) Desain *User Interface Device List*

Pada Gambar 13 *merupakan* tampilan perancangan desain awal *device list*, terdapat list dari perangkat jaringan yang telah dimasukan ke *database*.



Gambar 13. Desain UI *Device List*

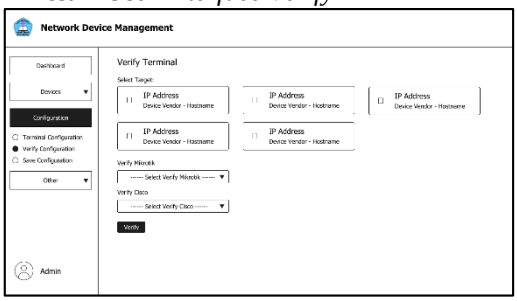
4) Desain User Interface *Configuration Device*



Gambar 14. Desain UI Configuration Device

Pada Gambar 14 merupakan tampilan perancangan desain awal *configuration*, terdapat target *device* yang ingin dikonfigurasi serta masukan sebagai terminal untuk konfigurasi.

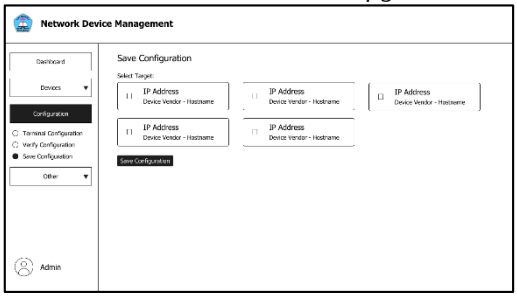
5) Desain *User Interface Verify*



Gambar 15. Verify

Pada Gambar 15 merupakan tampilan perancangan desain awal *verify*, terdapat target *device* yang ingin verifikasi konfigurasinya, serta pilihan *command* untuk *verify*.

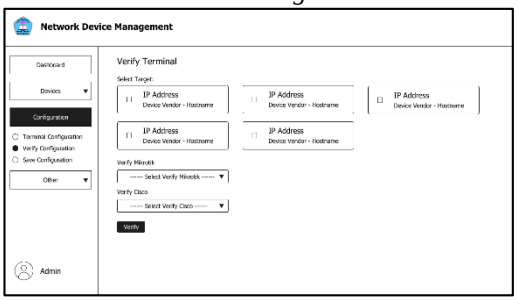
6) Desain User Interface *Save Configuration*



Gambar 16. Desain UI Save Configuration

Pada Gambar 16 merupakan tampilan perancangan *desain* awal *save configuration*, terdapat target *device* yang ingin disimpan konfigurasinya.

7) Desain User Interface *Log*



Gambar 17. Desain UI Log

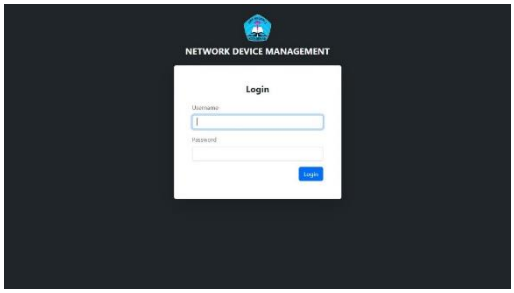
Pada Gambar 17 merupakan tampilan perancangan desain awal Log, terdapat *list* catatan dari aktivitas yang telah dilakukan oleh admin.

4.3. Implementasi

Pada tahap pengkodean sistem, peneliti mengimplementasikan tahapan desain ke pembuatan website dengan menggunakan HTML dan Bootstrap sebagai *frontend* dan Django versi 4.2.3 sebagai *framework web* serta *Library* Python Paramiko versi 3.2.0 untuk menghubungkan ke perangkat jaringan. Sistem dibuat pada Visual Studio Code, dan didapat hasil sebagai berikut.

1) Halaman Login

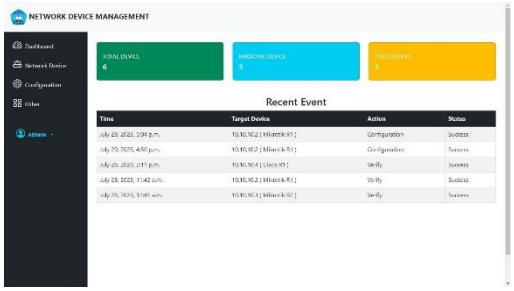
Gambar 18 Halaman *login* dari sistem pengelolaan perangkat jaringan menampilkan masukan *username* dan *password* untuk masuk ke dalam sistem.



Gambar 18. Halaman Login

2) Halaman *Dashboard*

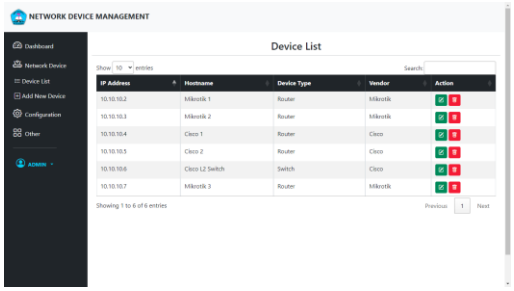
Gambar 19 Halaman utama dari sistem pengelolaan perangkat jaringan menampilkan jumlah total *device* keseluruhan dan *recent event* yang telah dilakukan oleh admin.



Gambar 19. Halaman Dashboard

3) Halaman *Device List*

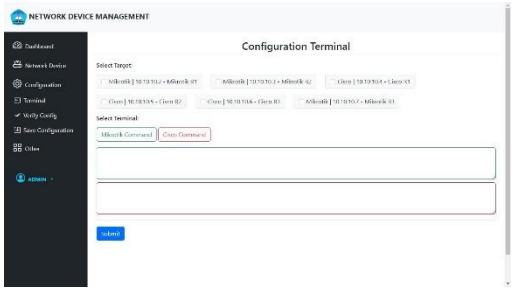
Pada Gambar 20 Halaman *Device* list menampilkan semua perangkat jaringan yang telah dimasukan admin.



Gambar 20. Halaman Device List

4) Halaman *Configuration Device*

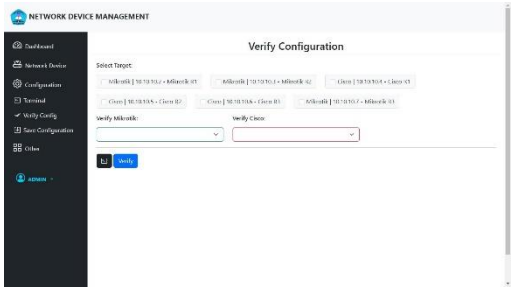
Gambar 21 *Halaman Configuration* menampilkan list *device* yang ingin dikonfigurasi, setelah mengklik button "*Command*" akan muncul *textarea* sebagai terminal untuk memasukan konfigurasi yang nantinya akan dikirim ke *device* yang telah dipilih.



Gambar 21. Halaman Configuration Device

5) Halaman *Verify*

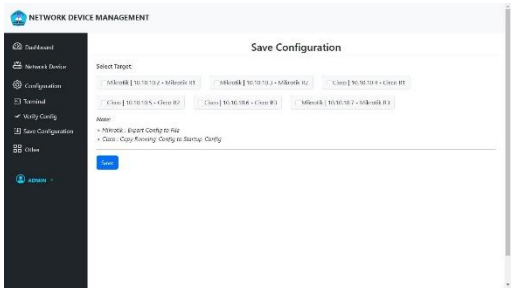
Gambar 22 Halaman *Verify Menampilkan* list *device* yang ingin di verifikasi konfigurasinya serta terdapat pilihan *select* untuk memilih *command* tertentu, dan setelah mengklik button "*Command*" akan muncul *textarea* sebagai masukan untuk konfigurasi *verify* secara manual.



Gambar 22. Halaman *Verify*

6) Halaman *Save Configuration*

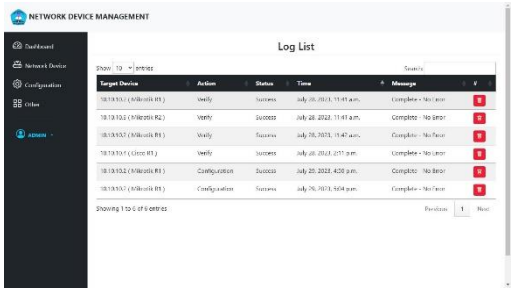
Gambar 23 Halaman *Save Configuration* menampilkan *list device* yang ingin disimpan konfigurasinya.



Gambar 23. Halaman *Save Configuration*

7) Halaman *Log*

Gambar 24 Halaman *Log menampilkan* keseluruhan catatan dari pekerjaan yang dilakukan oleh admin pada sistem pengelolaan perangkat jaringan.



Gambar 24. Halaman Log

8) Halaman *Administration*

Halaman Admin merupakan halaman yang sudah tersedia pada *framework django*, yang menampilkan manage *Groups* dan *Users* untuk sistem serta menampilkan keseluruhan data *device* dan juga *log*.

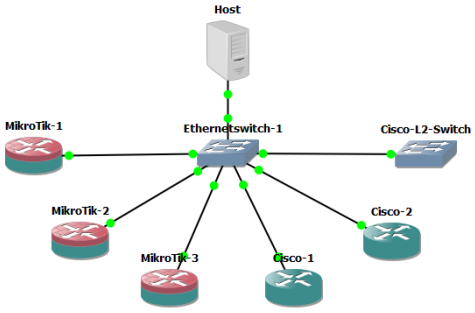
Gambar 25. Halaman Administration

4.4. Pengujian

Pada tahapan ini dilakukan sebuah pengujian dari sistem yang sudah dikerjakan dan dalam tahapan ini dapat diketahui sejauh mana fungsi-fungsi pada sistem ini dapat berjalan sesuai dengan tujuan perancangan sistem tersebut. Pengujian dilakukan dengan bantuan aplikasi GNS3 sebagai simulasi perangkat jaringan virtual yang terdiri dari 3 *Router* Mikrotik dan 2 *Router* Cisco dan 1 *Switch* Cisco.

Tabel 1. Spesifikasi Perangkat Simulasi

Hostname	Versi	Tipe Device	IP Address
Mikrotik 1	chr-6.44	Router	10.10.10.2
Mikrotik 2	chr-6.48	Router	10.10.10.3
Mikrotik 3	chr-7.10	Router	10.10.10.7
Cisco 1	C2691	Router	10.10.10.4
Cisco 2	C7200	Router	10.10.10.5
Cisco L2 Switch	vIOS L2	Switch	10.10.10.6



Gambar 26. Simulasi GNS3

Pengujian dilaksanakan dengan menggunakan dua cara yaitu *white-box* dan *black-box testing*. Berdasarkan hasil dari *black-box testing* yang telah dilaksanakan, dapat diambil sebuah kesimpulan bahwa sistem sudah sesuai dengan fungsionalitasnya dan berfungsi sesuai yang diharapkan.

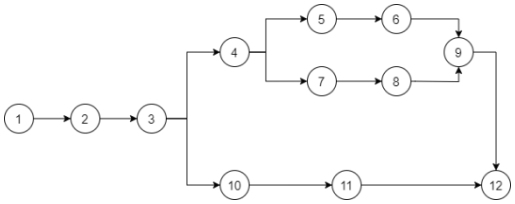
Tabel 2. Pengujian Blackbox

No	Aktivitas Pengujian	Realisasi yang diharapkan	Respon Sistem	Hasil
1	Masuk sistem	Muncul halaman *login*	Muncul halaman *login*	Valid
2	*Login user*	Masuk ke *Dashboard*	Masuk ke *Dashboard*	Valid
3	Tambah *device*	Menambahkan perangkat jaringan baru	Menambahkan perangkat jaringan baru	Valid
4	Masuk Menu *Device List*	Menampilkan halaman dan data perangkat jaringan	Menampilkan halaman dan data perangkat jaringan	Valid
5	Edit *device*	Mengubah data perangkat jaringan	Mengubah data perangkat jaringan	Valid
6	Hapus *device*	Menghapus data perangkat jaringan	Menghapus data perangkat jaringan	Valid
7	Konfigurasi *device* Mikrotik	Konfigurasi berhasil dan mencatat *Log*	Konfigurasi berhasil dan mencatat *Log*	Valid
8	Konfigurasi *device* Cisco	Konfigurasi berhasil dan mencatat *Log*	Konfigurasi berhasil dan mencatat *Log*	Valid
9	*Verify device* Mikrotik	Menampilkan hasil *verify* dan mencatat *Log*	Menampilkan hasil *verify* dan mencatat *Log*	Valid
10	*Verify device* Cisco	Menampilkan hasil *verify* dan mencatat *Log*	Menampilkan hasil *verify* dan mencatat *Log*	Valid
11	Simpan Konfigurasi Mikrotik	Berhasil menyimpan konfigurasi pada *device*	Berhasil menyimpan konfigurasi pada *device*	Valid
12	Simpan Konfigurasi Cisco	Berhasil menyimpan konfigurasi pada *device*	Berhasil menyimpan konfigurasi pada *device*	Valid
13	Masuk menu *Log*	Menampilkan halaman dan data *Log*	Menampilkan halaman dan data *Log*	Valid
14	Hapus *Log*	Hapus data *Log*	Hapus data *Log*	Valid

Selanjutnya *white-box testing* dilakukan dengan menghitung kompleksitas siklomatik pada fungsi *configuration*, *verify* dan *save configuration*.

```
1  def configuration(request):
2      if request.method == 'POST':
3          selected_device_id = request.POST.getlist('device')
           mikrotik_cmd = request.POST['mikrotik_cmd'].splitlines()
           cisco_cmd = request.POST['cisco_cmd'].splitlines()
           for x in selected_device_id:
               dev = get_object_or_404(Device, pk=x)
               ssh_client = paramiko.SSHClient()
               ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
               ssh_client.connect(hostname=dev.IP_address, username=dev.username,
   password=dev.password, port=dev.SSH_port)
4              try:
5                  if dev.vendor.lower() == 'mikrotik':
6                      for cmd in mikrotik_cmd:
                           stdin, stdout, stderr = ssh_client.exec_command(cmd)
7                  else:
8                      conn = ssh_client.invoke_shell()
                       conn.send("conf terminal\n")
                       for cmd in cisco_cmd:
                           conn.send(cmd+"\n")
                           time.sleep(1)
9              log = Log(target=dev.IP_address + " ( " +dev.hostname+" ) ",
   action="Configuration", status="Success", time=datetime.now(),
   message="Complete - No Error ")
               log.save()
10         except Exception as Exc:
11             log = Log(target=dev.IP_address + " ( " +dev.hostname+" ) ",
   action="Configuration", status="Error", time=datetime.now(), message=Exc)
               log.save()
12     return redirect('home')
```

Gambar 27. Pengujian White-box Configuration



Gambar 28. Flowgraph Configuration

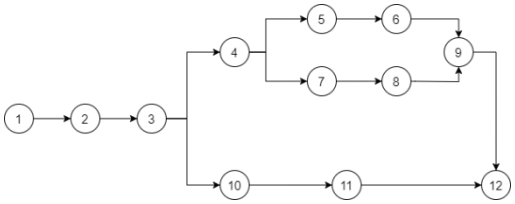
V(G) = E - N + 2
= 13 - 12 + 2
= 3

Sehingga pada skenario *configuration* terdapat 3 *path* yakni:

a) *Path* 1 = 1 - 2 - 3 - 4 - 5 - 6 - 9 - 12
b) *Path* 2 = 1 - 2 - 3 - 4 - 7 - 8 - 9 - 12
c) *Path* 3 = 1 - 2 - 3 - 10 - 11 – 12

```
1  def verify(request):
2      if request.method == 'POST':
3          result = []
           selected_device_id = request.POST.getlist('device')
           mikrotik_verify_select = request.POST['mikrotik_verify_select'].splitlines()
           mikrotik_verify_cmd = request.POST['mikrotik_verify_cmd'].splitlines()
           cisco_verify_select = request.POST['cisco_verify_select'].splitlines()
           cisco_verify_cmd = request.POST['cisco_verify_cmd'].splitlines()
           for x in selected_device_id:
               dev = get_object_or_404(Device, pk=x)
               ssh_client = paramiko.SSHClient()
               ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
               ssh_client.connect(hostname=dev.IP_address, username=dev.username, password=dev.password,
   port=dev.SSH_port)
4              try:
5                  if dev.vendor.lower() == 'mikrotik':
6                      for mikrotikselect in mikrotik_verify_select:
                           stdin, stdout, stderr = ssh_client.exec_command(mikrotikselect)
                           result.append("Result on {}".format(dev.IP_address))
                           result.append(stdout.read().decode())
                       for mikrotikcmd in mikrotik_verify_cmd:
                           stdin, stdout, stderr = ssh_client.exec_command(mikrotikcmd)
                           result.append("Result on {}".format(dev.IP_address))
                           result.append(stdout.read().decode())
7                  else:
8                      conn = ssh_client.invoke_shell()
                       conn.send("terminal length 0\n")
                       for ciscoselect in cisco_verify_select:
                           result.append("Result on {}".format(dev.IP_address))
                           conn.send(ciscoselect+"\n")
                           time.sleep(5)
                           output = conn.recv(65535)
                           result.append (output.decode())
                       for ciscocmd in cisco_verify_cmd:
                           result.append("Result on {}".format(dev.IP_address))
                           conn.send(ciscocmd+"\n")
                           time.sleep(5)
                           output = conn.recv(65535)
                           result.append (output.decode())
9              log = Log(target=dev.IP_address + " ( " +dev.hostname+" ) ", action="Verify", status="Success",
   time=datetime.now(), message="Complete - No Error")
               log.save()
10         except Exception as Exc:
11             log = Log(target=dev.IP_address + " ( " +dev.hostname+" ) " , action="Verify", status="Error",
   time=datetime.now(), message=Exc)
               log.save()
12     result = "\n".join(result)
       return render(request,'verifyResult.html',{"result":result})
```

Gambar 29. Pengujian White-box Verify



Gambar 30. Flowgraph Verify

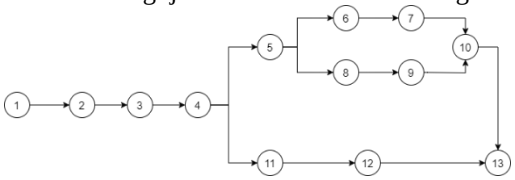
V(G) = E - N + 2
= 13 - 12 + 2
= 3

Sehingga pada skenario *verify* terdapat 3 path yakni:

a) *Path* 1 = 1 - 2 - 3 - 4 - 5 - 6 - 9 - 12
b) *Path* 2 = 1 - 2 - 3 - 4 - 7 - 8 - 9 - 12
c) *Path* 3 = 1 - 2 - 3 - 10 - 11 – 12

```
1  def saveconfig(request):
2      timenow = datetime.today().strftime("%d-%m-%Y,%H:%M:%S")
3      if request.method == 'POST':
4          result = []
           selected_device_id = request.POST.getlist('device')
           for x in selected_device_id:
               dev = get_object_or_404(Device, pk=x)
               ssh_client = paramiko.SSHClient()
               ssh_client.set_missing_host_key_policy(paramiko.AutoAddPolicy())
               ssh_client.connect(hostname=dev.IP_address, username=dev.username,
   password=dev.password, port=dev.SSH_port)
5              try:
6                  if dev.vendor.lower() == 'mikrotik':
7                      stdin, stdout, stderr = ssh_client.exec_command("export file= Backup-" +
   timenow)
                       stdin, stdout, stderr = ssh_client.exec_command("file print")
                       result.append("Result on {}".format(dev.IP_address))
                       result.append(stdout.read().decode())
8                  else:
9                      conn = ssh_client.invoke_shell()
                       conn.send("terminal length 0\n")
                       result.append("Result on {}".format(dev.IP_address))
                       conn.send("copy running-config startup-config \n\n")
                       time.sleep(5)
                       output = conn.recv(65535)
                       result.append (output.decode())
10             log = Log(target=dev.IP_address + " ( " +dev.hostname+" ) ", action="Verify",
   status="Success", time=datetime.now(), message="Complete - No Error")
               log.save()
11         except Exception as Exc:
12             log = Log(target=dev.IP_address + " ( " +dev.hostname+" ) " , action="Verify",
   status="Error", time=datetime.now(), message=Exc)
               log.save()
13         result = "\n".join(result)
           return render(request,'verifyResult.html',{"result":result})
```

Gambar 31. Pengujian White-box Save Configuration



Gambar 32. Flowgraph Save Configuration

V(G) = E - N + 2
= 14 - 13 + 2
= 3

Sehingga pada skenario *save configuration* terdapat 3 *path* yakni:

Path 1 = 1 - 2 - 3 - 4 - 5 - 6 - 7 - 10 - 13
Path 2 = 1 - 2 - 3 - 4 - 5 - 8 - 9 - 10 - 13
Path 3 = 1 - 2 - 3 - 4 -11 - 12 – 13

Berdasarkan keseluruhan skenario pengujian yang telah dilakukan, maka dapat disimpulkan bahwa hasil yang didapat pada *white-box testing* di setiap fungsi kode adalah berjalan dengan baik dan tidak ada *error* serta kesalahan logika.

4.5. Pemeliharaan

Tahap ini merupakan tahap akhir pada penelitian ini. Setelah dilakukan pengujian, sistem sudah dapat dijalankan oleh *user* dan disamping itu juga dilakukan pemeliharaan dan update agar sistem tetap berjalan lancar, jika terdapat sebuah *error* tidak terduga, atau *bug* dapat segera diperbaiki.

4.6. Pembahasan

Dalam penelitian ini, website pengelolaan perangkat jaringan untuk SMKN 3 Kota Bekasi telah berhasil dibuat dan dibangun menggunakan metodologi *Software Development Life Cycle* (SDLC) model *waterfall*. Website ini sendiri berguna untuk memberikan kemudahan bagi network admin untuk mengelola perangkat jaringan serta konfigurasi secara terpusat, serta untuk meminimalisir kesalahan konfigurasi yang dilakukan terdapat menu *verify* untuk melihat konfigurasi yang telah dibuat.

Berdasarkan pengujian dengan *Black-box*, diketahui bahwa *website* sudah berfungsi dengan lancar dan fitur-fitur berjalan sesuai dengan semestinya. Lalu berdasarkan pengujian *White-box*, diketahui kode fungsi pada *Configuration, Verify* dan *Save Configuration* sudah sesuai dengan logika dan berjalan dengan baik.

5. KESIMPULAN DAN SARAN

Berdasarkan penelitian yang telah dilaksanakan, maka dapat diambil sebuah kesimpulan bahwa seluruh tahapan pengembangan sistem telah dilaksanakan dengan baik dan sistem yang dirancang sudah sesuai dengan kebutuhan, serta mampu mengatasi permasalahan admin jaringan sehingga pengelolaan dan konfigurasi perangkat jaringan menjadi lebih mudah, efisien dan terpusat pada satu *website*. Saran penelitian kedepannya dapat menambahkan fungsi konfigurasi untuk perangkat jaringan dengan *vendor* lain sesuai dengan kebutuhan.

DAFTAR PUSTAKA

[1] M. Fahmi, M. Maisyaroh, I. Komarudin, S. Faizah, and I. Fadhilah, "Otomatisasi Jaringan Menggunakan Script Python Untuk Penyediaan Konfigurasi Internet Dan Manajemen Mikrotik," *Bina Insa. Ict J.*, vol. 8, no. 1, pp. 53–62, 2021, doi: 10.51211/biict.v8i1.1517.

[2] S. Prahara, Martanto, and I. Ali, "Optimalisasi Jaringan Internet Dengan Optimalisasi Load Balancing Menggunakan Parameter QOS (Studi Kasus: SMK Bina Warga Lemahabang)," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 7, no. 1, pp. 211–217, 2023.

[3] J. T. Santoso, *Proyek Coding dengan Python*. Semarang: Yayasan Prima Agus Teknik, Universitas Sains & Teknologi Komputer (Universitas STEKOM).

[4] W. Prabowo, "Rancang Bangun Automasi Jaringan Komputer Dengan Python," POLITEKNIK NEGERI MALANG, 2021.

[5] B. P. Aji, A. Hernawan, A. Nurjihadi, and A. A. Rizal, "Sistem Informasi Surat Elektronik Untuk Akademik UIN Mataram (Dengan Python Django Framework)," *J. Begawe Teknol. Inf.*, vol. 3, no. 2, pp. 252–262, 2022, doi: 10.29303/jbegati.v3i2.777.

[6] G. Fauziah and A. Tholib, "Aplikasi Inventaris Sekolah Berbasis Web Menggunakan Framework Django Di Mts. Nurul Hidayah Sumberrejo Paiton," *COREAI J. Kecerdasan Buatan, Komputasi dan Teknol. Inf.*, vol. 3, no. 1, pp. 72–80, 2022, doi: 10.33650/coreai.v3i1.4211.

[7] L. G. Mauboy and T. Wellem, "Studi Perbandingan Library Untuk Implementasi Network Automation Menggunakan Paramiko Dan Netmiko Pada Router Mikrotik," *JURIKOM (Jurnal Ris. Komputer)*, vol. 9, no. 4, p. 790, 2022, doi: 10.30865/jurikom.v9i4.4420.

[8] K. Nugroho, A. D. Abrariansyah, and S. Ikhwan, "Perbandingan Kinerja Library Paramiko dan Netmiko Dalam Proses Otomasi Jaringan," *Infotekjar*, vol. 5, no. 1, pp. 1–8, 2020, doi: 10.30743/infotekjar.v5i1.2758.

[9] R. Pratama, M. Orisa, and F. Ariwibisono, "Aplikasi Monitoring Dan Controlling Server Menggunakan Protocol Icmp (Internet Control Message Protocol) Dan Ssh (Secure Shell) Berbasis Website," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 4, no. 1, pp. 397–403, 2020, doi: 10.36040/jati.v4i1.2310.

[10] H. B. Susueno, I. T. Wibowo, S. U. Masruroh, D. Khairani, and I. U. Khoiri, "Analisis Routing Protocol Is-Is dengan MPLS Traffic Engineering Menggunakan GNS3," *J. Ilm. Fifo*, vol. 13, no. 1, pp. 32–40, 2021, doi: 10.22441/fifo.2021.v13i1.004.

[11] R. Susanto and A. Dara Andriana, "Perbandingan Model Waterfall Dan Prototyping Untuk Pengembangan Sistem Informasi," *Maj. Ilm. UNIKOM*, vol. 14, no. 1, pp. 41–46, 2016, [Online]. Available: http://repository.unikom.ac.id/id/eprint/30459.

[12] A. Abdul Wahid, "Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi," *INFOMAN'S J. Ilmu-ilmu Inform. dan Manaj.*, no. November, pp. 1–5, 2020, [Online]. Available: https://www.researchgate.net/profile/Aceng-Wahid/publication/346397070_Analisis_Metode_Waterfall_Untuk_Pengembangan_Sistem_Informasi/links/5fbfa91092851c933f5d76b6/Analisis-Metode-Waterfall-Untuk-Pengembangan-Sistem-Informasi.pdf.

[13] A. Dwi Riski, "Analisis Model Waterfall : Pengertian, Tahapan, Kelebihan dan kekurangan," 2022. https://osc.medcom.id/community/analisis-model-waterfall-pengertian-tahapan-kelebihan-dan-kekurangan-4352.

[14] Y. Heriyanto, "Perancangan Sistem Informasi

Rental Mobil Berbasis Web Pada PT.APM Rent Car," *J. Intra-Tech*, vol. 2, no. 2, pp. 64–77, 2018.

[15] B. B. Sasongko, F. Malik, F. Ardiansyah, A. F. Rahmawati, F. D. Adhinata, and D. P. Rakhmadani, "Pengujian Blackbox Menggunakan Teknik Equivalence Partitions pada Aplikasi Petgram Mobile," *Fak. Inform. Inst. Teknol. Telkom Purwokerto*, vol. 2, no. 1, pp. 10–16, 2021.

[16] M. F. Londjo, "Implementasi White Box Testing Dengan Teknik Basis Path Pada Pengujian Form Login," *J. Siliwaangi*, vol. 7, no. 2, pp. 35–40, 2021.