

Development of a Web-Based Network Device Configuration Automation Application Using Python

Willy Rivaldo Manurung^{1*}, Andy Triwinarko^{2**}, Didi Istardi^{3***}

* Teknik Multimedia Jaringan, Politeknik Negeri Batam

** Teknik Komputer, Politeknik Negeri Batam

*** Teknik Elektro, Politeknik Negeri Batam

willy.4311911008@polibatam.ac.id¹, andy@polibatam.ac.id², istardi@polibatam.ac.id³

Article Info

Article history:

Received 2024-06-25

Revised 2026-08-26

Accepted 2026-08-30

Keyword:

Configuration Management,
Django,
GNS3,
Network Automation,
Python

ABSTRACT

In today's digital era, the demand for networks and internet services continues to rise, resulting in increasingly dynamic and complex computer networks. With rapidly evolving network conditions and a growing number of connected devices, there is a critical need for automation solutions to mitigate manual configuration errors and streamline time efficiency. This research focuses on developing a web-based network device configuration automation application, named ANCO (Automated Network Configuration), using Python and the Django framework, with the Paramiko library providing SSH-based (Secure Shell) command execution to Cisco and Mikrotik devices. The application aims to reduce manual configuration errors and enhance efficiency in managing multi-vendor network devices. Employing the four phases of the Rapid Application Development (RAD) methodology-requirements planning, system design, construction, and implementation-accelerates the development process. The application facilitates dynamic routing OSPF configuration, VLAN addition, command-line-based configuration, configuration verification, manual and scheduled backup, and activity logging. Testing using the black-box method confirms that all functional features operate successfully, while an efficiency analysis on three Cisco and three Mikrotik routers demonstrates significant time savings, up to 97% compared to manual configurations, along with more consistent configuration times, while offering an intuitive interface for streamlined device and configuration management.



This is an open access article under the [CC-BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.

I. PENDAHULUAN

Di era digital saat ini, kebutuhan akan jaringan dan internet semakin meningkat pesat, menghasilkan jaringan komputer yang semakin dinamis dan kompleks. Istilah "dinamis dan kompleks" menandakan bahwa kondisi jaringan terus berubah dan semakin banyak perangkat yang terhubung ke jaringan komputer. Secara umum, jaringan komputer terdiri dari beberapa perangkat berkabel atau nirkabel yang saling terhubung, seperti *access point*, *router*, *switch*, dan sejumlah komputer. Agar jaringan dapat berfungsi secara optimal, perangkat jaringan harus di-konfigurasi terlebih dahulu oleh administrator jaringan. Tantangan dalam mengelola jaringan semakin besar ketika terdapat banyak perangkat dari vendor berbeda, sehingga konfigurasi manual satu per satu

memerlukan banyak waktu. Solusi otomatisasi jaringan dibutuhkan untuk meningkatkan efisiensi dan akurasi dalam pengelolaan jaringan, karena konfigurasi manual rentan terhadap kesalahan dan memakan waktu lebih lama [1], [2].

Otomatisasi jaringan adalah metode untuk menyelesaikan tugas lebih cepat dengan menggunakan logika pemrograman untuk mengelola layanan jaringan [3]. Melalui otomatisasi jaringan, administrator dapat menghilangkan tugas-tugas berulang seperti pencadangan (*backup*), melihat informasi perangkat, melakukan konfigurasi, dan verifikasi hanya melalui satu *platform* [4]. Konfigurasi manual tidak efisien dan rentan terhadap kesalahan. Menurut beberapa laporan industri, kesalahan manusia menyumbang setidaknya 40% dari kegagalan jaringan, dengan beberapa perkiraan mencapai 80% [5]. Dalam Laporan Cisco 2020 *Business Resilience*

Network Survey Report [6], 50% responden memprioritaskan otomatisasi jaringan untuk mengatasi gangguan yang terjadi. Pemanfaatan skrip Python memastikan penerapan perintah konfigurasi yang konsisten, sehingga menghilangkan potensi kesalahan konfigurasi manual oleh administrator jaringan.

Python menjadi salah satu bahasa pemrograman paling populer untuk pekerjaan otomatisasi jaringan. Python memiliki dukungan komunitas yang luas dan menawarkan banyak pustaka yang mendukung otomatisasi jaringan, seperti Paramiko [7]. Paramiko menyediakan fungsi untuk membuat koneksi *Secure Shell* v2 (SSHv2) ke berbagai perangkat jaringan, memungkinkan eksekusi skrip yang berisi konfigurasi dan pengiriman skrip tersebut ke berbagai perangkat melalui protokol *Secure Shell* (SSH) [8].

Pendekatan otomatisasi yang digunakan pada penelitian ini bersifat imperatif berbasis command-line interface (CLI) melalui SSH, berbeda dengan pendekatan deklaratif berbasis model seperti NETCONF/RESTCONF atau orkestrasi menggunakan Ansible. Pendekatan CLI dipilih karena kompatibel dengan perangkat Cisco IOS dan Mikrotik RouterOS yang digunakan pada penelitian ini tanpa memerlukan dukungan API tambahan pada sisi perangkat, meskipun konsekuensi-nya adalah keluaran skrip yang lebih rentan terhadap perubahan sintaks *command-line* dibandingkan pendekatan berbasis model data terstruktur seperti YANG pada NETCONF. Pengembangan lebih lanjut ke arah dukungan NETCONF/RESTCONF atau integrasi dengan Ansible dapat menjadi arah penelitian selanjutnya untuk meningkatkan portabilitas antar-vendor.

Terdapat beberapa buku dan penelitian yang membahas atau berhubungan dengan otomatisasi jaringan. Salah satu buku yang relevan adalah *Network Automation using Python* [9]. Buku ini menjelaskan implementasi otomatisasi jaringan menggunakan Python dengan memanfaatkan beberapa pustaka seperti Paramiko, Netmiko, NAPALM, PyNTC, dan Ansible. Implementasi dilakukan dengan menggunakan sistem operasi Linux Ubuntu versi 16.04, dan pengujian konfigurasi dilakukan pada router Cisco.

Penelitian lain yang membahas implementasi otomatisasi jaringan berbasis web pada perangkat jaringan Cisco dan Mikrotik [10]. Aplikasi yang dikembangkan dapat menampilkan jumlah perangkat router, menambahkan konfigurasi, memverifikasi konfigurasi, dan melihat riwayat aktivitas. Pengujian dilakukan dengan menggunakan metode *black-box* dan *white-box*, dan hasil penelitian menunjukkan bahwa aplikasi dapat melakukan konfigurasi penambahan *loopback interface* pada tiga router Cisco dan tiga router Mikrotik sekaligus.

Penelitian berikutnya membahas pengembangan aplikasi otomatisasi jaringan berbasis web untuk melakukan konfigurasi IP gateway secara otomatis [11]. Penelitian ini menggunakan metode pengujian *black-box* untuk menguji fungsionalitas aplikasi dan pengujian koneksi untuk memvalidasi konfigurasi penambahan IP gateway. Hasil pengujian menunjukkan berhasilnya penambahan IP gateway pada perangkat router Cisco dan Mikrotik, yang di-verifikasi

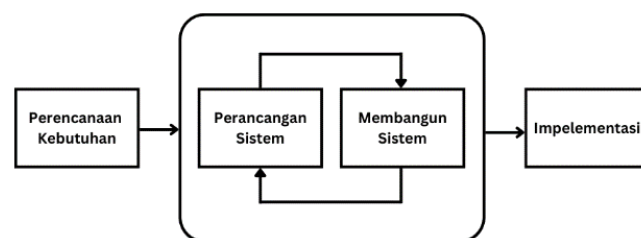
dengan berhasil melakukan ping ke alamat 8.8.8.8 atau google.com.

Selain itu juga penelitian implementasi penggunaan skrip Python dalam otomatisasi jaringan [12] untuk mengelola router Mikrotik yang dimiliki PT Link Net Tbk dalam melakukan konfigurasi awal atau provisioning untuk pengguna baru. Metode yang digunakan adalah NDLC dalam membangun jaringan. Implementasi dilakukan menggunakan sistem operasi Linux Ubuntu versi 20.04.1 LTS, Python, dan Paramiko. Pengujian dilakukan menggunakan dua skrip Python yaitu “provisioning.py” untuk melakukan konfigurasi pada pelanggan baru dan “mass-update-config1.py” untuk melakukan pembaruan konfigurasi massal radius dan SNMP (Simple Network Management Protocol) pada CPE (Customer-Premise Equipment) yang sudah ada. Hasil pengujian menunjukkan bahwa konfigurasi massal berhasil pada 10 perangkat router hanya dalam waktu sekitar 25 detik.

Berdasarkan beberapa penelitian tersebut, maka tujuan dari penelitian ini adalah mengembangkan aplikasi berbasis web untuk mengotomatisasi konfigurasi perangkat jaringan sehingga dapat mengeliminasi tugas-tugas yang bersifat berulang seperti pencadangan (backup), mengkonsolidasikan informasi perangkat, konfigurasi, dan verifikasi dalam satu platform. Pengembangan aplikasi dilakukan menggunakan bahasa pemrograman Python, pustaka Paramiko, dan diintegrasikan dengan situs web yang dibangun menggunakan framework Django.

II. METODE PENELITIAN

Penelitian dimulai dengan melakukan tinjauan literatur pada beberapa artikel dan buku yang berkaitan dengan implementasi otomatisasi jaringan. Metode yang digunakan adalah RAD (*Rapid Application Development*) sebagai metode pendekatan berorientasi objek untuk pengembangan sistem yang bertujuan untuk mempersingkat waktu antara desain dan implementasi [2], [14], [15]. Terdapat empat langkah dalam pengembangan sistem, yaitu perencanaan kebutuhan, perancangan sistem, membangun sistem, dan implementasi. Tahapan pada metode RAD dapat dilihat pada Gambar 1.



Gambar 1. Metode RAD yang memiliki 4 tahapan, yaitu perencanaan kebutuhan, perancangan sistem, membangun sistem, dan implementasi.

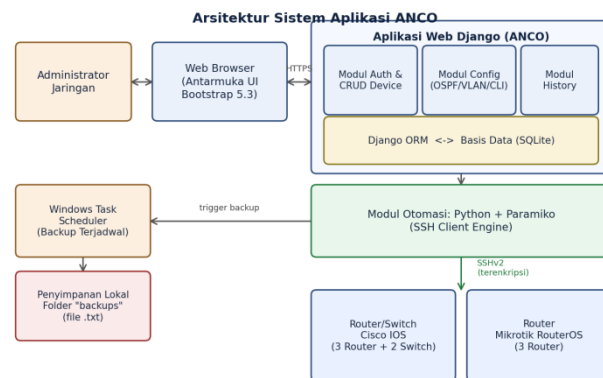
Tahapan pertama dalam pengembangan aplikasi adalah perencanaan kebutuhan. Pada tahap ini, identifikasi kebutuhan perangkat keras dan perangkat lunak dilakukan untuk memastikan semua elemen yang diperlukan tersedia

dan kompatibel. Pada penelitian ini, perangkat keras yang digunakan adalah satu unit laptop dengan spesifikasi Intel Core i5, 16GB RAM, dan 512GB SSD untuk menjalankan semua perangkat lunak dalam pengembangan aplikasi.

Perangkat lunak terdiri dari sistem operasi Windows 11 sebagai *server/controller*. Bahasa pemrograman Python 3.12.1 digunakan untuk membuat logika konfigurasi, sementara *library* Paramiko 3.4.0 digunakan untuk melakukan koneksi *secure shell* (SSH) dan mengeksekusi perintah secara aman ke perangkat jaringan. Visual Studio Code sebagai kode editor dan Django dipilih sebagai *web framework* karena menyediakan fitur *templating*, *otentikasi pengguna*, dan *Object-Relational Mapping* (ORM) untuk mendefinisikan model dan basis data sebagai tempat penyimpanan data perangkat jaringan. Dibandingkan dengan micro-framework seperti Flask atau FastAPI, Django dipilih karena bersifat "*batteries-included*", yaitu sudah menyediakan modul admin, sistem otentikasi, dan ORM secara bawaan sehingga pengembangan fitur CRUD, manajemen pengguna, dan riwayat aktivitas pada aplikasi ANCO dapat dilakukan lebih cepat tanpa perlu menambahkan banyak pustaka pihak ketiga, meskipun konsekuensinya adalah beban aplikasi yang lebih besar dibandingkan Flask atau FastAPI yang lebih ringan dan cocok untuk layanan API sederhana. GNS3 2.2.43 sebagai alat simulasi jaringan, dengan VMware Workstation 17 Pro untuk menjalankan GNS3 VM. Selain itu *image* perangkat Cisco dan Mikrotik dipasang ke dalam aplikasi GNS3 untuk simulasi.

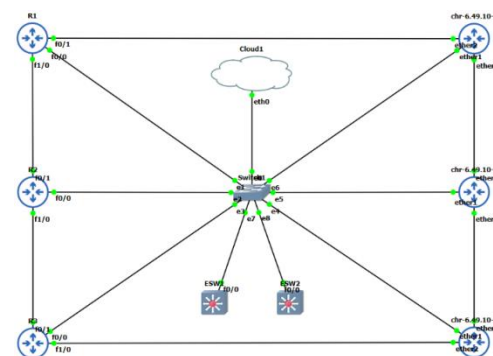
Selain kebutuhan perangkat keras dan perangkat lunak, aplikasi ini juga memiliki kebutuhan fungsional dan non-fungsional. Kebutuhan fungsional aplikasi memungkinkan pengguna untuk mendaftar, masuk, dan keluar dari aplikasi; menambah konfigurasi *dynamic routing Open Shortest Path First* (OSPF); menambahkan konfigurasi *Virtual Local Area Network* (VLAN); melakukan konfigurasi berbasis *command line*; melakukan verifikasi konfigurasi; menjalankan pencadangan (*backup*) secara manual dan terjadwal; melakukan operasi CRUD (*Create, Read, Update, Delete*) pada perangkat jaringan, dan melihat riwayat aktivitas. Kebutuhan non-fungsional aplikasi mencakup kompatibilitas terhadap berbagai browser.

Sebelum membahas topologi jaringan secara rinci, arsitektur sistem aplikasi ANCO secara umum digambarkan pada Gambar 2. Administrator berinteraksi dengan aplikasi Django (mencakup modul otentikasi dan CRUD perangkat, modul konfigurasi OSPF/VLAN/command-line, serta modul history) melalui protokol HTTP/HTTPS. Aplikasi Django menyimpan data perangkat dan riwayat aktivitas pada basis data melalui Django ORM, kemudian meneruskan perintah konfigurasi ke modul otomasi berbasis Python dan pustaka Paramiko, yang selanjutnya mengeksekusi perintah pada perangkat Cisco dan Mikrotik melalui koneksi SSHv2 yang terenkripsi. Proses pencadangan terjadwal dipicu oleh Windows Task Scheduler dan hasilnya disimpan pada penyimpanan lokal berupa berkas teks.



Gambar 2. Arsitektur sistem aplikasi ANCO, menggambarkan alur komunikasi antara web browser, aplikasi Django, modul otomasi Python-Paramiko, dan perangkat jaringan Cisco/Mikrotik melalui SSHv2.

Tahapan kedua dalam pengembangan aplikasi adalah perancangan sistem. Pada tahapan ini dirancang sebuah topologi dan desain sistem. Topologi jaringan dibuat menggunakan aplikasi GNS3 dimana terdapat satu *cloud* yang merupakan perangkat fisik laptop yang dijadikan sebagai *server/controller*, 3 *router* Cisco c3725, 3 *router* Mikrotik CHR-6.49.10 dan 2 *switch* Cisco. Desain topologi dapat dilihat pada Gambar 3.



Gambar 3. Desain topologi jaringan yang terdiri dari satu *cloud*, tiga *router* Cisco, tiga *router* Mikrotik, dan dua *switch* Cisco (EtherSwitch).

Setiap perangkat telah diberikan alokasi alamat *IP* yang harus dikonfigurasi terlebih dahulu pada masing-masing perangkat. Terdapat 9 buah perangkat yang harus dikonfigurasi alamat *IP* sesuai pada gambar topologi jaringan yang sudah dirancang sebelumnya. Alokasi alamat *IP* dapat dilihat pada Tabel 1.

TABEL I
ALOKASI ALAMAT IP

No	Perangkat	Antarmuka	Alamat IP
1	Cloud/Controller	GNS3 eth0	192.168.227.128/24
2	R1 Cisco	Fa0/0	192.168.227.11/24
		Fa0/1	10.10.11.1/30
		Fa1/0	10.10.12.1/30
3	R2 Cisco	Fa0/0	192.168.227.12/24
		Fa0/1	10.10.12.2/30
		Fa1/0	10.10.13.1/30
4	R3 Cisco	Fa0/0	192.168.227.13/24
		Fa0/1	10.10.13.2/30

		Fa1/0	10.10.14.1/30
5	CHR-6.49.10-1 Mikrotik	Ether1	192.168.227.14/24
		Ether2	10.10.11.2/30
		Ether3	10.10.16.2/30
6	CHR-6.49.10-2 Mikrotik	Ether1	192.168.227.15/24
		Ether2	10.10.15.2/30
		Ether3	10.10.16.1/30
7	CHR-6.49.10-3 Mikrotik	Ether1	192.168.227.16/24
		Ether2	10.10.14.2/30
		Ether3	10.10.15.1/30
8	ESW1 Cisco	Fa0/0	192.168.227.17/24
9	ESW2 Cisco	Fa0/0	192.168.227.18/24

Selain itu, setiap perangkat juga harus dikonfigurasi untuk mengaktifkan SSHv2 agar dapat diakses dari jarak jauh oleh *server/controller*. Konfigurasi SSHv2 dapat dilihat pada Gambar 4.

```

R1(config)#username cisco password cisco
R1(config)#username cisco privilege 15
R1(config)#ip domain-name anco.local
R1(config)#crypto key generate rsa modulus 1024
The name for the keys will be: R1.anco.local

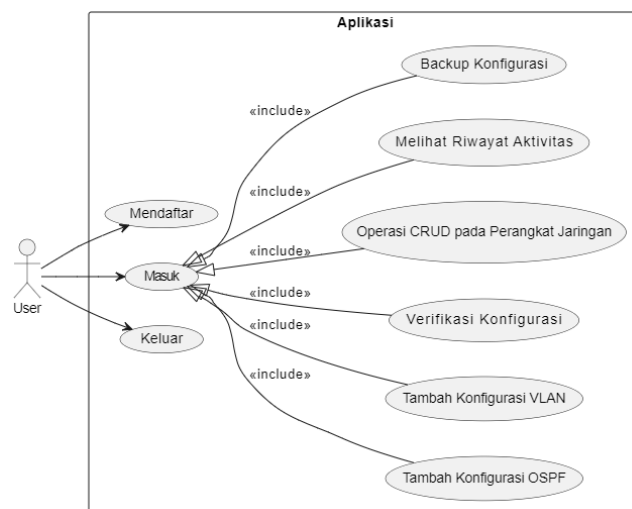
% The key modulus size is 1024 bits
% Generating 1024 bit RSA keys, keys will be non-exportable...[OK]

R1(config)#line vty 0 15
R1(config-line)#login local
R1(config-line)#end
R1#wr
Building configuration...
[OK]
R1#
*Mar  1 01:12:09.555: %SSH-5-ENABLED: SSH 1.99 has been enabled
*Mar  1 01:12:09.619: %SYS-5-CONFIG_I: Configured from console by console
R1#

```

Gambar 4. Konfigurasi SSHv2 pada terminal router Cisco

Perancangan desain sistem dibuat menggunakan diagram *Unified Modelling Language* (UML). Penelitian ini menggunakan *use case diagram* untuk mengilustrasikan interaksi fungsi utama sistem dan pengguna [16]. Pada *use case diagram*, seorang Administrator jaringan digambarkan sebagai aktor yang berinteraksi dengan sistem.



Gambar 5. *Use Case Diagram* dimana aktor digambarkan sebagai seorang administrator jaringan yang berinteraksi dengan aplikasi atau sistem.

Gambar 5 merupakan desain *use case diagram* aplikasi yang akan dikembangkan diberi nama ANCO, akronim dari *Automated Network Configuration*. Aktor dapat mendaftar dan masuk ke dalam aplikasi. Aktor dapat mengatur data perangkat jaringan melalui operasi CRUD, melakukan konfigurasi *dynamic routing* OSPF, menambah VLAN, dan konfigurasi lewat *command line* secara simultan, melihat riwayat aktivitas, melakukan *backup* dan keluar dari aplikasi.

Tahapan ketiga adalah membangun sistem. Pada tahapan ini aplikasi dibangun menggunakan menggunakan Python sebagai bahasa pemrograman dan menggunakan *library* Paramiko sebagai modul melakukan koneksi *secure shell* (SSH) ke perangkat jaringan yang terhubung dan *framework* Django untuk membangun aplikasi berbasis *website*.

Tahapan keempat adalah implementasi. Implementasi dilakukan secara simulasi dengan menggunakan aplikasi GNS3 sebelum diimplementasikan ke lingkungan nyata. Pengujian dilakukan menggunakan metode *black-box* untuk mengetahui keberhasilan fungsional aplikasi yang sudah dirancang sebelumnya.

III. HASIL DAN PEMBAHASAN

Penelitian ini menghasilkan sebuah aplikasi otomatisasi konfigurasi perangkat jaringan berbasis *website* yang diberi nama ANCO yang merupakan akronim dari *Automated Network Configuration*. Aplikasi ini hanya dijalankan oleh seorang administrator jaringan dalam melakukan konfigurasi perangkat jaringan.

A. Tampilan Antarmuka Aplikasi

Aplikasi ini memiliki tampilan antarmuka yang terdiri dari beberapa halaman berdasarkan fungsinya masing-masing, yaitu *Sign Up*, *Login*, *Dashboard*, *Network Device*, *Dropdown Config* (OSPF Config, VLAN Config, *Command Line*, *Check Configuration*), *Backup*, dan *History*. Pengembangan antarmuka aplikasi menggunakan *frontend toolkit* Bootstrap v5.3 dengan tujuan untuk membuat aplikasi responsif dan mempercepat proses pembuatannya karena

sudah tersedia kumpulan komponen *User Interface* (UI) yang siap pakai.

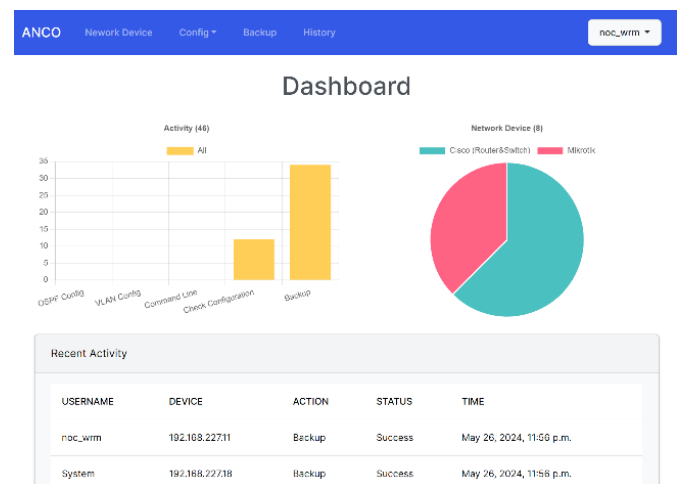
1) *Halaman Sign Up*: Halaman ini digunakan untuk mendaftarkan akun baru. Pada halaman ini, pengguna baru diminta untuk mengisi informasi berikut: username, email, password, dan confirm password. Setelah mengisi informasi tersebut, pengguna dapat klik tombol "Sign Up" untuk memproses pendaftaran. Jika data yang dimasukkan benar, pengguna baru akan dialihkan ke halaman Login. Namun, jika terdapat kesalahan dalam pengisian, data tidak akan diproses dan akan ditampilkan pesan kesalahannya dan diharuskan mengisi ulang informasi yang diminta. Pengguna juga memiliki opsi untuk langsung masuk ke aplikasi jika sudah memiliki akun sebelumnya dengan cara klik pada tautan "Already have an account? Log In here". Tampilan halaman Sign Up dapat dilihat pada Gambar 6.

Gambar 6. Halaman *Sign Up* untuk mendaftarkan akun pengguna baru.

2) *Halaman Login*: Halaman ini digunakan untuk masuk ke dalam aplikasi. Seorang administrator terlebih dahulu harus melakukan proses login sebelum dapat menggunakan aplikasi. Hal ini dilakukan untuk mencegah penggunaan aplikasi oleh pihak yang tidak berwenang. Pengguna harus memasukkan username dan password pada formulir login yang disediakan dengan benar agar dapat masuk ke dalam aplikasi. Jika data yang dimasukkan tidak benar, pengguna akan kembali diarahkan ke halaman login untuk memasukkan ulang username dan password. Pengguna juga memiliki opsi untuk mendaftar akun baru jika belum memiliki akun sebelumnya dengan cara klik pada tautan "Don't have an account? Sign Up here". Tampilan halaman Login dapat dilihat pada Gambar 7.

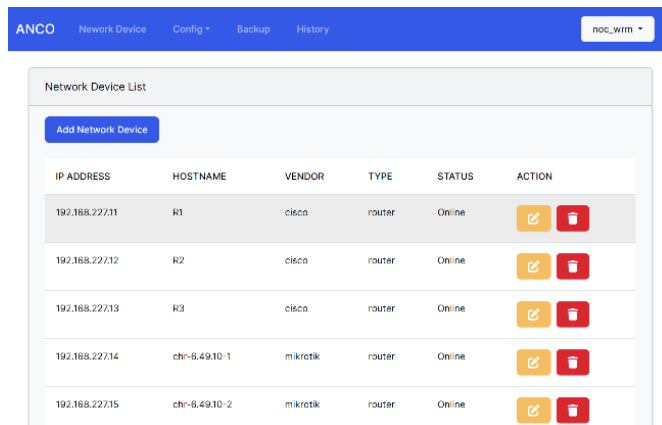
Gambar 7. Halaman *Login* untuk masuk ke dalam aplikasi oleh pengguna terdaftar.

3) *Halaman Dashboard*: Halaman dashboard menjadi halaman pertama yang diakses pengguna setelah berhasil masuk (login) ke aplikasi. Halaman ini menampilkan informasi total perangkat yang terhubung ke aplikasi, dan aktivitas terakhir dalam bentuk grafik dan tabel. Pada bagian navbar terdapat 4 menu utama, yaitu Network Device, Dropdown Config, Backup, dan History. Pada bagian ujung navbar ditampilkan juga nama pengguna beserta tombol "Logout" dalam bentuk dropdown. Tampilan halaman dashboard dapat dilihat pada Gambar 8.



Gambar 8. Halaman *dashboard* menampilkan informasi total jumlah perangkat dari kedua vendor Cisco dan Mikrotik, beserta aktivitas terakhir dalam bentuk grafik dan tabel.

4) *Halaman Network Device*: Halaman ini menampilkan daftar perangkat jaringan yang terhubung ke aplikasi. Informasi yang ditampilkan berupa IP address, hostname, vendor, type, status, dan action yang memiliki dua opsi yaitu edit dan delete. Pada halaman ini juga disediakan sebuah tombol "Add Network Device" untuk mendaftarkan perangkat jaringan ke dalam basis data aplikasi. Tampilan halaman Network Device dapat dilihat pada Gambar 9.



Gambar 9. Halaman *Network Device* digunakan untuk menambahkan perangkat, menampilkan informasi perangkat yang tersimpan, dan tindakan yang memiliki 2 opsi, yaitu *edit* dan *delete*.

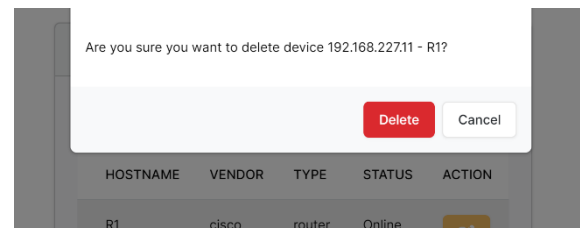
Pengguna dapat menambahkan perangkat jaringan ke dalam basis data aplikasi dengan klik tombol “Add Network Device” yang akan dialihkan ke halaman yang berisi formulir registrasi. Beberapa informasi yang perlu diisi untuk menambahkan perangkat, yaitu *IP address*, *hostname*, *username*, *password*, *SSH port*, *vendor*, dan *device type*. Tampilan formulir registrasi perangkat dapat dilihat pada Gambar 10.

Gambar 10. Tampilan formulir registrasi penambahan perangkat jaringan ke basis data sistem.

Selain fungsi menambahkan perangkat, pengguna juga dapat mengubah data perangkat dengan klik tombol “Edit” pada kolom *action*. Pengguna akan dialihkan ke halaman yang sama seperti halaman penambahan perangkat, pengguna dapat mengubah data perangkat pada formulir yang disediakan. Setelah selesai mengubah data, pengguna dapat menekan tombol “Update” untuk menyimpan perubahan ke basis data. Tampilan formulir sunting perangkat dapat dilihat pada Gambar 11.

Gambar 11. Tampilan formulir sunting data perangkat jaringan untuk memperbaharui informasi perangkat.

Selain fungsi perubahan data, pengguna juga dapat menghapus data perangkat jaringan dari basis data apabila sudah tidak digunakan dengan klik tombol “Delete” pada kolom *action*. Tampilan fungsi *delete* dapat dilihat pada Gambar 12.



Gambar 12. Fungsi *delete* untuk menghapus data perangkat jaringan dari basis data.

5) *Halaman Config*: Halaman ini digunakan untuk melakukan konfigurasi pada perangkat jaringan yang terhubung ke sistem aplikasi. Halaman ini memiliki submenu, yaitu OSPF Config, VLAN Config, Command Line, dan Check Configuration. Sebelum melakukan konfigurasi pengguna harus memilih perangkat yang akan dikonfigurasi pada daftar perangkat yang tersedia. Pada konfigurasi dynamic routing OSPF pengguna hanya perlu memasukkan beberapa parameter konfigurasi pada formulir yang sudah disediakan, yaitu process ID (Cisco), IP address, wildcard mask (Cisco), subnet mask/prefix, dan area ID. Setelah selesai pengguna dapat klik tombol “Submit” untuk mengirimkan skrip konfigurasi ke perangkat jaringan yang telah dipilih secara simultan. Tampilan halaman konfigurasi OSPF dapat dilihat pada gambar 13.

Gambar 13. Tampilan formulir konfigurasi *dynamic routing* OSPF dengan memilih daftar perangkat *Cisco*

Pada konfigurasi penambahan VLAN daftar perangkat yang disediakan hanya perangkat *switch* *Cisco*. pengguna hanya perlu memasukkan nomor VLAN, dan nama VLAN. Setelah selesai pengguna dapat klik tombol “*Submit*” untuk mengirimkan skrip konfigurasi ke perangkat jaringan yang telah dipilih secara simultan. Tampilan halaman konfigurasi VLAN dapat dilihat pada gambar 14.

Gambar 14. Tampilan formulir konfigurasi VLAN pada *switch* *Cisco*

Selain konfigurasi *dynamic routing* OSPF dan VLAN dengan menggunakan formulir, pengguna juga dapat melakukan konfigurasi lain berbasis *command line*. Fitur ini memungkinkan pengguna tidak perlu lagi masuk secara satu-per-satu ke setiap perangkat jaringan karena dengan menggunakan terminal konfigurasi yang disediakan, pengguna dapat melakukan konfigurasi secara simultan pada semua perangkat jaringan. Tampilan halaman konfigurasi berbasis *command line* dapat dilihat pada Gambar 15.

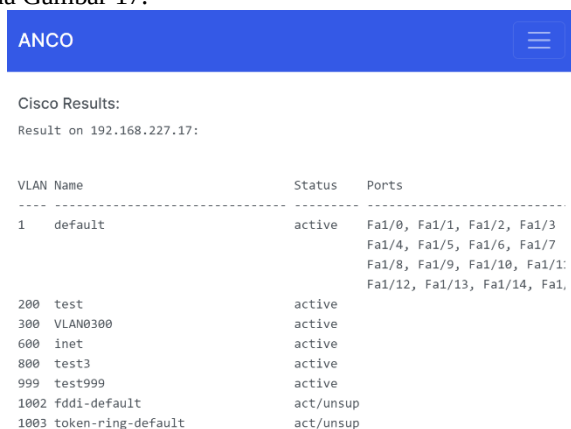
Gambar 15. Tampilan Halaman *Config* untuk melakukan konfigurasi pada perangkat jaringan berbasis *command line*.

Halaman Check Configuration: Halaman ini digunakan untuk memeriksa konfigurasi pada perangkat jaringan. Terdapat beberapa tombol yang dapat digunakan untuk memeriksa konfigurasi umum seperti “*Show IP*”, “*Show IP Route*”, dan “*Show VLAN Brief*”. Selain itu, disediakan juga fitur *command line* untuk memeriksa konfigurasi lainnya. Tampilan halaman dapat dilihat pada Gambar 16.

Gambar 16. Tampilan halaman *check configuration* untuk memeriksa hasil konfigurasi perangkat *switch* *Cisco*.

Sebelum melakukan pemeriksaan konfigurasi, pengguna harus terlebih dahulu memilih perangkat apa saja yang akan diperiksa pada daftar perangkat yang tersedia. Setelah memilih perangkat, pengguna dapat menekan tombol yang menyediakan fungsi cek konfigurasi umum atau menuliskan skrip pada *textarea* sesuai dengan *vendor* yang dipilih.

Setelah selesai, pengguna dapat melakukan klik tombol “Submit” untuk mendapatkan hasil pengecekan. skrip akan dijalankan secara simultan pada semua perangkat yang dipilih. Tampilan halaman hasil pengecekan dapat dilihat pada Gambar 17.



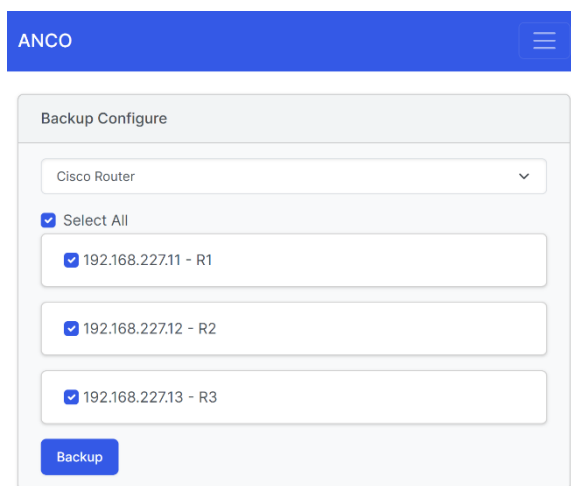
ANCO

Cisco Results:
Result on 192.168.227.17:

VLAN	Name	Status	Ports
1	default	active	Fa1/0, Fa1/1, Fa1/2, Fa1/3 Fa1/4, Fa1/5, Fa1/6, Fa1/7 Fa1/8, Fa1/9, Fa1/10, Fa1/11 Fa1/12, Fa1/13, Fa1/14, Fa1/15
200	test	active	
300	VLAN0300	active	
600	inet	active	
800	test3	active	
999	test999	active	
1002	fdi-default	act/unsup	
1003	token-ring-default	act/unsup	

Gambar 17. Tampilan hasil pengecekan konfigurasi VLAN pada perangkat switch Cisco.

6) *Halaman Backup*: Halaman ini digunakan untuk melakukan pencadangan konfigurasi perangkat. Pengguna dapat memilih perangkat yang akan dicadangkan pada daftar perangkat yang tersedia. Hasil pencadangan dalam bentuk file.txt yang disimpan pada direktori proyek dengan nama folder “backups”. Folder akan dibuat secara otomatis melalui skrip konfigurasi jika belum tersedia. Tampilan halaman backup dapat dilihat pada Gambar 18.



ANCO

Backup Configure

Cisco Router

☒ Select All

☒ 192.168.227.11 - R1

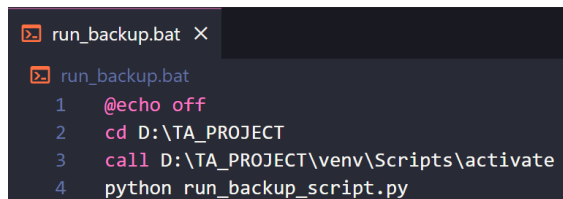
☒ 192.168.227.12 - R2

☒ 192.168.227.13 - R3

Backup

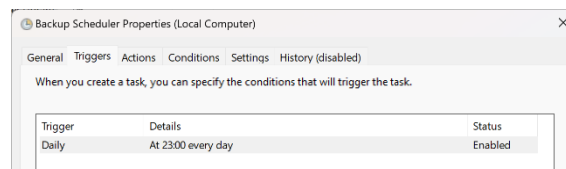
Gambar 18. Tampilan halaman backup untuk melakukan pencadangan pada perangkat router Cisco

Selain pencadangan secara manual oleh pengguna, aplikasi dapat melakukan pencadangan otomatis dengan skrip yang akan dijalankan secara terjadwal oleh sistem. Skrip ini dipanggil dengan bantuan program Windows Task Scheduler yang diatur untuk menjalankan program pada pukul 23:00 setiap hari. Tampilan skrip backup dan pengaturan program Windows Task Scheduler dapat dilihat pada Gambar 19 dan Gambar 20.



```
run_backup.bat X
run_backup.bat
1 @echo off
2 cd D:\TA_PROJECT
3 call D:\TA_PROJECT\venv\Scripts\activate
4 python run_backup_script.py
```

Gambar 19. Skrip pencadangan yang dijalankan oleh task scheduler.



Backup Scheduler Properties (Local Computer)

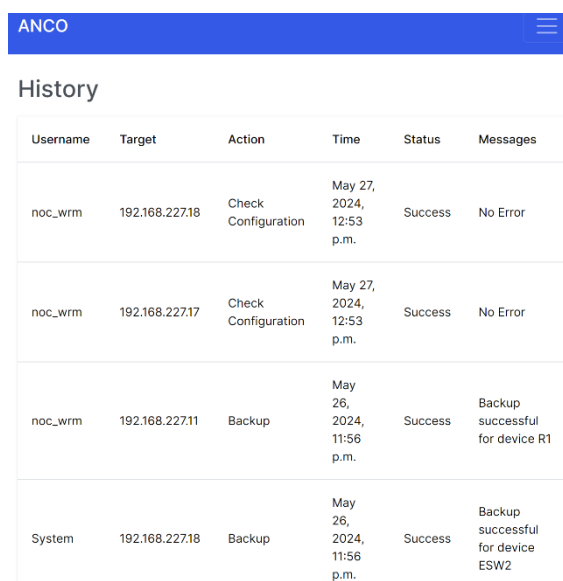
General Triggers Actions Conditions Settings History (disabled)

When you create a task, you can specify the conditions that will trigger the task.

Trigger	Details	Status
Daily	At 23:00 every day	Enabled

Gambar 20 tugas pencadangan diatur pada pukul 23:00 setiap hari menggunakan task scheduler.

7) *Halaman History*: Halaman ini menampilkan riwayat aktivitas pengguna atau administrator jaringan. Menu ini menampilkan informasi aktivitas konfigurasi, backup, dan pemeriksaan konfigurasi yang dilakukan oleh pengguna. Selain itu, juga ditampilkan informasi username, alamat IP, aksi, waktu eksekusi, status dan pesan. Tampilan Halaman history dapat dilihat pada Gambar 21.



ANCO

History

Username	Target	Action	Time	Status	Messages
noc_wrm	192.168.227.18	Check Configuration	May 27, 2024, 12:53 p.m.	Success	No Error
noc_wrm	192.168.227.17	Check Configuration	May 27, 2024, 12:53 p.m.	Success	No Error
noc_wrm	192.168.227.11	Backup	May 26, 2024, 11:56 p.m.	Success	Backup successful for device R1
System	192.168.227.18	Backup	May 26, 2024, 11:56 p.m.	Success	Backup successful for device ESW2

Gambar 21. Halaman history menampilkan aktivitas yang dilakukan oleh pengguna.

B. Black-box Testing

Pengujian *black-box* dilakukan untuk mengetahui keberhasilan fungsionalitas aplikasi yang sudah dirancang sebelumnya dengan mengamati hasil *input* dan *output* dari aplikasi. Hasil pengujian dapat dilihat pada Tabel 2.

TABEL II
HASIL PENGUJIAN BLACK BOX

No	Fitur	Skenario	Hasil
----	-------	----------	-------

1	Sign Up	Mendaftarkan pengguna baru dan memverifikasi pengguna berhasil terdaftar di basis data	berhasil
2	Login	Masuk ke dalam aplikasi menggunakan akun yang sudah didaftarkan	berhasil
3	CRUD	Menambahkan perangkat ke basis data	berhasil
		Melihat daftar dan status perangkat yang terdaftar	berhasil
		Melakukan pembaharuan data perangkat	berhasil
		Menghapus perangkat yang terdaftar	berhasil
4	Konfigurasi	Menambahkan konfigurasi <i>dynamic routing OSPF</i> pada <i>router</i>	berhasil
		Menambahkan konfigurasi VLAN pada <i>switch</i>	berhasil
		Menambahkan konfigurasi menggunakan <i>command line</i>	berhasil
		Menjalankan dan melihat hasil verifikasi konfigurasi	berhasil
6	Backup	Menjalankan <i>backup</i> oleh pengguna	berhasil
		Menjalankan <i>backup</i> oleh Sistem (otomatis)	berhasil
7	Logout	Keluar dari aplikasi	berhasil

C. Analisa Efisiensi Konfigurasi Manual vs Otomatis

Pengujian efisiensi konfigurasi manual dengan konfigurasi otomatis melibatkan pengujian konfigurasi *dynamic routing OSPF single-area*. Hal ini sesuai dengan fitur yang telah dibuat pada aplikasi. Konfigurasi diimplementasikan pada 3 *router* Cisco dan 3 *router* Mikrotik. Proses konfigurasi menggunakan aplikasi Putty dengan melakukan *remote* pada perangkat melalui protokol SSH. Perhitungan waktu konfigurasi menggunakan aplikasi Wireshark dengan merekam lalu lintas paket SSH. Perhitungan waktu dimulai sejak paket SSH pertama kali ditangkap atau proses dimulai hingga selesai melakukan konfigurasi. Dilakukan pengujian sebanyak 5 kali untuk memperoleh lebih banyak data dan membantu dalam memvalidasi konsistensi hasil. Hasil pengujian dapat dilihat pada Tabel 3 dan Tabel 4.

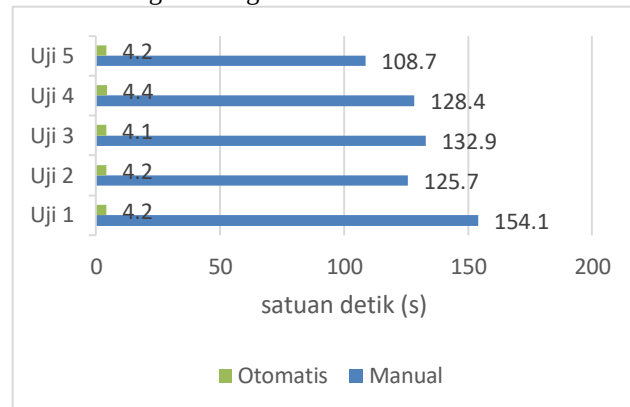
TABEL III
HASIL PENGUJIAN KONFIGURASI MANUAL ROUTING OSPF

Uji	Cisco			Mikrotik			Total	Mean
	R1	R2	R3	R1	R2	R3		
1	25.0	30.4	41.0	20.9	16.8	20.0	154.1s	25.7s
2	25.5	24.0	25.1	17.7	14.1	19.3	125.7	21.0s
3	36.5	21.2	23.1	21.4	15.4	15.3	132.9s	22.2s
4	28.7	26.0	21.9	13.8	18.8	19.2	128.4s	21.4s
5	25.9	19.0	18.6	18.4	11.4	15.4	108.7	18.1s
Rata-rata waktu konfigurasi seluruh perangkat: 130s								

TABEL IV
HASIL KONFIGURASI OTOMATIS ROUTING OSPF

Uji	Jumlah perangkat	Waktu	Mean
1	6 (Cisco & Mikrotik)	4.2s	4.2s
2		4.2s	
3		4.1s	
4		4.4s	
5		4.2s	

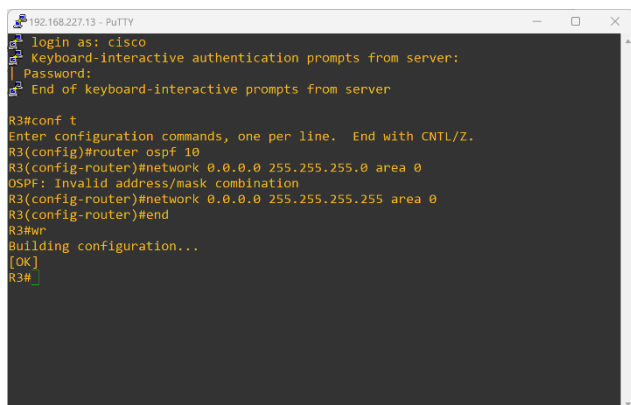
Konfigurasi otomatis hanya membutuhkan rata-rata waktu sekitar 4,2 detik untuk melakukan konfigurasi pada seluruh perangkat, sedangkan konfigurasi manual membutuhkan rata-rata waktu sekitar 130 detik. Hal ini menunjukkan konfigurasi otomatis dapat menghemat waktu 125,8 detik atau sekitar 97% dibandingkan dengan manual.



Gambar 22. Waktu rata-rata konfigurasi seluruh perangkat

Konsistensi waktu konfigurasi otomatis juga lebih tinggi dibandingkan dengan manual. Dapat dilihat pada Gambar 22 waktu konfigurasi manual bervariasi dari 108,7 detik hingga 154,1 detik, sedangkan waktu konfigurasi otomatis cukup konsisten, berkisar dari 4,1 detik hingga 4,4 detik. Selain itu ditemukan 3 kesalahan dalam penulisan konfigurasi manual terdiri dari 2 kesalahan dalam penentuan *wildcard mask* pada *router* Cisco dan 1 kesalahan penulisan *password* pada *router* Mikrotik seperti terlihat pada Gambar 23 dan 24.

Secara kuantitatif, konsistensi ini dapat diukur menggunakan standar deviasi (SD) dan koefisien variasi (coefficient of variation/CV). Total waktu konfigurasi manual pada lima kali pengujian (154,1; 125,7; 132,9; 128,4; dan 108,7 detik) memiliki SD sekitar 14,6 detik dengan CV sekitar 11,2%, sedangkan waktu konfigurasi otomatis (4,2; 4,2; 4,1; 4,4; dan 4,2 detik) memiliki SD hanya sekitar 0,1 detik dengan CV sekitar 2,3%. Nilai CV yang jauh lebih kecil pada konfigurasi otomatis menunjukkan secara statistik bahwa aplikasi ANCO tidak hanya lebih cepat, tetapi juga jauh lebih dapat diprediksi (predictable) dan konsisten dibandingkan proses manual yang dipengaruhi variasi kecepatan mengetik dan pengalaman operator. Selain itu, apabila ditinjau per jenis perangkat, rata-rata waktu konfigurasi manual pada *router* Cisco (28,3; 24,1; dan 25,9 detik untuk R1, R2, dan R3) secara konsisten lebih tinggi dibandingkan *router* Mikrotik (18,4; 15,3; dan 17,8 detik untuk R1, R2, dan R3), yang mengindikasikan sintaks *command-line* Cisco IOS yang lebih panjang dan berjenjang (mode konfigurasi global, interface, dan *router*) berkontribusi terhadap waktu pengerjaan manual yang lebih lama dibandingkan *RouterOS* Mikrotik yang cenderung lebih ringkas.



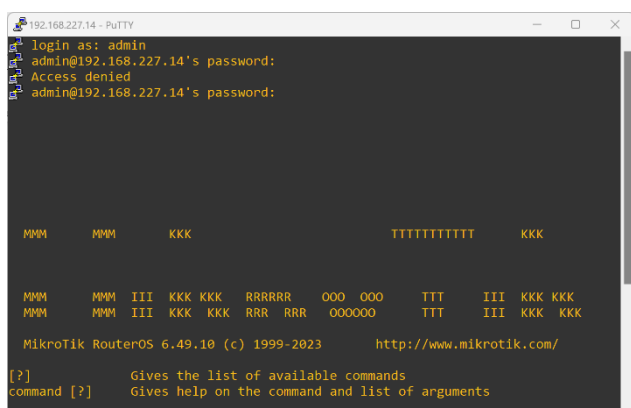
```

192.168.227.13 - PuTTY
login as: cisco
Keyboard-interactive authentication prompts from server:
| Password:
| End of keyboard-interactive prompts from server

R3#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R3(config)#router ospf 10
R3(config-router)#network 0.0.0.0 255.255.255.0 area 0
OSPF: Invalid address/mask combination
R3(config-router)#network 0.0.0.0 255.255.255.0 area 0
R3(config-router)#end
R3#wr
Building configuration...
[OK]
R3#

```

Gambar 23. Temuan kesalahan penulisan wildcard mask konfigurasi dynamic routing OSPF pada router Cisco



```

192.168.227.14 - PuTTY
login as: admin
admin@192.168.227.14's password:
Access denied
admin@192.168.227.14's password:

MMM      MMM      KKK      TTTTTTTTTT      KKK

MMM      MMM      III      KKK      KKK      RRRRRR      OOO      OOO      TTT      III      KKK      KKK
MMM      MMM      III      KKK      KKK      RRR      RRR      OOOOOO      TTT      III      KKK      KKK

MikroTik RouterOS 6.49.10 (c) 1999-2023      http://www.mikrotik.com/

[?]      Gives the list of available commands
command [?]      Gives help on the command and list of arguments

```

Gambar 24. Temuan kesalahan login password pada router Mikrotik

Berdasarkan hasil pengujian dan analisis yang sudah dilakukan dapat disimpulkan, penggunaan aplikasi otomatisasi konfigurasi perangkat jaringan berbasis web memberikan peningkatan efisiensi yang signifikan dalam hal waktu dan konsistensi konfigurasi dibandingkan dengan metode konfigurasi manual.

D. Keamanan, Reliabilitas, dan Keterbatasan Sistem

Selain efisiensi waktu, aspek keamanan menjadi pertimbangan penting mengingat aplikasi berbasis web ini mengelola kredensial dan konfigurasi perangkat jaringan yang rentan terhadap risiko akses tidak sah. Pada implementasi saat ini, autentikasi pengguna aplikasi memanfaatkan mekanisme bawaan Django (session-based authentication dengan password hashing PBKDF2), sedangkan komunikasi ke perangkat jaringan sepenuhnya melalui protokol SSHv2 yang terenkripsi menggunakan pustaka Paramiko, sehingga perintah konfigurasi tidak dikirim dalam bentuk plain-text pada jalur jaringan. Namun demikian, kredensial perangkat (username dan password) pada versi aplikasi ini masih disimpan pada basis data tanpa mekanisme enkripsi tambahan (at-rest encryption) atau vault khusus, sehingga menjadi salah satu keterbatasan yang perlu ditingkatkan pada pengembangan berikutnya, misalnya dengan menerapkan enkripsi kredensial atau integrasi dengan secret manager eksternal.

Terkait keandalan (reliability) operasional, aplikasi menyediakan fitur pencatatan aktivitas (activity logging) pada halaman History yang merekam username, target perangkat, jenis aksi, waktu eksekusi, status, dan pesan setiap operasi konfigurasi maupun backup, sehingga administrator dapat melakukan audit trail. Akan tetapi, aplikasi belum menyediakan mekanisme rollback konfigurasi otomatis maupun version control terhadap perubahan konfigurasi; berkas hasil backup yang tersimpan pada folder "backups" saat ini masih perlu dibandingkan dan dipulihkan secara manual oleh administrator apabila terjadi kesalahan konfigurasi. Penambahan mekanisme rollback otomatis berbasis perbandingan berkas backup sebelum dan sesudah konfigurasi menjadi salah satu rekomendasi pengembangan lanjutan.

Pengujian efisiensi pada penelitian ini dilakukan pada skala terbatas, yaitu 3 router Cisco dan 3 router Mikrotik yang dikonfigurasi secara simultan, dan belum mencakup pengujian skalabilitas (scalability testing) pada jumlah perangkat yang lebih besar (misalnya puluhan hingga ratusan perangkat) yang umum dijumpai pada jaringan enterprise atau kampus berskala besar. Konsistensi waktu konfigurasi otomatis yang tinggi (SD kurang dari 0,1 detik) pada skala kecil ini menjadi indikasi awal yang positif, namun perilaku sistem pada beban kerja (concurrent connection) yang lebih besar memerlukan pengujian lanjutan sebelum diterapkan pada lingkungan produksi.

Dibandingkan dengan tools otomatisasi jaringan yang telah mapan di industri seperti Ansible [4] atau pustaka NAPALM yang dibahas pada [9], yang umumnya bersifat generik dan memerlukan penulisan playbook atau driver terpisah untuk setiap vendor, kontribusi utama aplikasi ANCO terletak pada penyediaan antarmuka web terintegrasi yang menggabungkan manajemen inventaris perangkat, konfigurasi OSPF dan VLAN berbasis formulir, konfigurasi command-line, verifikasi, backup terjadwal, dan riwayat aktivitas dalam satu platform yang dirancang khusus untuk kebutuhan administrator jaringan skala kampus tanpa memerlukan penulisan playbook maupun instalasi tools tambahan pada sisi administrator. Dengan demikian, kontribusi penelitian ini bersifat implementatif dan berorientasi pada kemudahan penggunaan (usability) bagi administrator yang belum familiar dengan tools orkestrasi berbasis command-line, sekaligus membuka peluang riset lanjutan untuk memperkaya aspek metodologis, seperti pengujian skalabilitas dan mekanisme keamanan kredensial yang lebih baik. Beberapa keterbatasan lain dari penelitian ini juga perlu dicatat: (1) aplikasi bergantung pada koneksi jaringan yang stabil antara server/controller dan perangkat, karena kegagalan koneksi SSH di tengah proses konfigurasi belum ditangani dengan mekanisme retry otomatis; (2) pengujian hanya dilakukan pada perangkat Cisco c3725 dan Mikrotik CHR-6.49.10 dalam lingkungan simulasi GNS3, sehingga kompatibilitas dengan perangkat vendor lain seperti Juniper belum diverifikasi; dan (3) evaluasi belum mencakup pengujian pada lingkungan produksi nyata (real production network) di luar simulasi.

IV. KESIMPULAN

Penelitian ini menghasilkan sebuah aplikasi otomatisasi konfigurasi perangkat jaringan berbasis *website* yang dikembangkan menggunakan pemrograman Python, *library* Paramiko, dan *framework* Django. Hasilnya, aplikasi dapat melakukan konfigurasi dynamic routing OSPF (Open Shortest Path First), penambahan VLAN (Virtual Local Area Network), dan konfigurasi berbasis command line secara simultan pada beberapa perangkat jaringan. Selain itu, aplikasi juga dapat melakukan verifikasi konfigurasi, memiliki fungsi CRUD (Create, Read, Update, Delete), menjalankan pencadangan (backup) secara manual dan terjadwal, serta melihat riwayat aktivitas. Metode pengujian yang digunakan adalah black-box untuk mengetahui keberhasilan fungsional aplikasi dengan mengamati hasil input dan output dari aplikasi. Selain itu, dilakukan analisis efisiensi waktu dengan membandingkan konsumsi waktu konfigurasi otomatis dengan manual. Konfigurasi yang diuji adalah dynamic routing OSPF pada tiga router Cisco dan tiga router Mikrotik. Pengujian dilakukan secara virtual menggunakan aplikasi simulasi jaringan GNS3 sebelum diimplementasikan ke lingkungan nyata. Batasan pengembangan aplikasi berfokus pada otomatisasi pencadangan (backup) terjadwal dan konfigurasi dynamic routing OSPF pada perangkat router Cisco dan Mikrotik.

Pengujian pada aplikasi dilakukan menggunakan metode *black-box* menunjukkan keberhasilan fungsional aplikasi yang sudah dirancang sebelumnya. Selain itu dilakukan juga analisa efisiensi antara konfigurasi otomatis dan konfigurasi manual. Berdasarkan analisa yang sudah dilakukan konfigurasi otomatis hanya membutuhkan rata-rata waktu sekitar 4,2 detik untuk melakukan konfigurasi pada seluruh perangkat, sedangkan konfigurasi manual membutuhkan rata-rata waktu sekitar 130 detik. Hal ini menunjukkan konfigurasi otomatis dapat menghemat waktu 125,8 detik atau sekitar 97% dibandingkan dengan manual. Selain itu, aplikasi ini juga mengurangi kesalahan konfigurasi yang umum terjadi pada metode manual.

DAFTAR PUSTAKA

- [1] V. V. S. S. Balaram, C. Mukundha, dan S. Bhutada, "Enhancement of network administration through software defined networks," *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 18, no. 1, hlm. 30–36, 2016, doi: 10.9790/0661-18113036.
- [2] R. A. Wiryawan dan N. R. Rosyid, "Pengembangan aplikasi otomatisasi administrasi jaringan berbasis website menggunakan bahasa pemrograman python," *Jurnal SIMETRIS*, vol. 10, no. 2, hlm. 741–752, Nov 2019, doi: 10.24176/simet.v10i2.3589.
- [3] G. S. Santyadiputra, I. M. E. Listartha, dan G. A. J. Saskara, "The effectiveness of Automatic Network Administration (ANA) in network automation simulation at Universitas Pendidikan Ganesha," dalam *Journal of Physics: Conference Series*, IOP Publishing Ltd, Mar 2021. doi: 10.1088/1742-6596/1810/1/012028.
- [4] Red Hat, *Network Automation for Everyone: Modernize Your Network With Red Hat Ansible Automation*. Red Hat, Inc., 2021. Diakses: 24 Februari 2023. [Daring]. Tersedia pada: <https://www.redhat.com/en/engage/network-automation-everyone-201901070306>
- [5] K. Fultz, M. Krishnarao, dan S. Wu, *Network Automation VMware® Special Edition*. Hoboken, New Jersey: John Wiley & Sons, Inc., 2020. Diakses: 24 Februari 2023. [Daring]. Tersedia pada: https://www.vmware.com/content/microsites/learn/en/544472_REG.html
- [6] Cisco, "2021 Global Networking Trends Report," 2020. Diakses: 24 Februari 2023. [Daring]. Tersedia pada: <https://www.cisco.com/c/dam/en/us/solutions/enterprise-networks/2021-networking-report.pdf>
- [7] A. Ratan, *Practical Network Automation*, 2 ed. Birmingham, UK: Packt Publishing, 2018. Diakses: 24 Februari 2023. [Daring]. Tersedia pada: <https://learning.oreilly.com/library/view/practical-network-automation/9781789955651/>
- [8] K. Nugroho, A. D. Abrariansyah, dan S. Ikhwani, "Perbandingan kinerja library paramiko dan netmiko dalam proses otomasi jaringan," *InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan*, vol. 5, no. 1, hlm. 1–8, Sep 2020, doi: 10.30743/infotekjar.v5i1.2758.
- [9] A. R. Komarudin, *Otomatisasi Administrasi Jaringan Dengan Script Python*. Jasakom, 2018.
- [10] S. Nugroho dan B. Pujiarto, "Network automation pada beberapa perangkat router menggunakan pemrograman python," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 9, no. 1, hlm. 79–86, 2022, doi: 10.25126/jtiik.2022913947.
- [11] E. S. Ginting, Suroso, dan I. Hadi, "Pengujian Konfigurasi Otomatis Penambahan Gateway Pada Virtual Router Menggunakan Aplikasi Otomatisasi Jaringan Berbasis Web," *Jurnal Media Informatika Budidarma*, vol. 4, hlm. 1126–1131, 2020, doi: 10.30865/mib.v4i4.2485.
- [12] M. Fahmi, M. Maisyaroh, I. Komarudin, S. Faizah, dan I. Fadhillah, "Otomatisasi Jaringan Menggunakan Script Python Untuk Penyediaan Konfigurasi Internet Dan Manajemen Mikrotik," *Bina Insani ICT Journal*, vol. 8, no. 1, hlm. 53–62, Jun 2021, doi: 10.51211/biict.v8i1.1517.
- [13] Y. C. E. Paksi dan I. R. Widiarsari, "Perancangan dan implementasi website network automation pada RT RW NET Dusun Senden Magelang dengan framework django," *Jurnal Teknik Informatika (JUTIF)*, vol. 3, no. 5, hlm. 1313–1322, Okt 2022, doi: 10.20884/1.jutif.2022.3.5.350.
- [14] K. E. Kendall dan J. E. Kendall, *Systems Analysis And Design*, 8 ed. Upper Saddle River, New Jersey: Pearson Education, Inc., 2010.
- [15] M. S. J. Pradana, A. Hendrawan, B. V. Christioko, dan A. P. R. Pinem, "Implementasi sistem administrasi di Unit Pelaksana Teknis Pusat Pengembangan Publikasi Ilmiah Dosen Universitas Semarang berbasis website," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 9, no. 3, hlm. 501–506, 2022, doi: 10.25126/jtiik.2021864089.
- [16] A. Dennis, B. H. Wixom, dan D. Tegarden, *Systems Analysis & Design An Object-Oriented Approach With UML*, 5 ed. Hoboken, New Jersey: John Wiley & Sons, Inc, 2015.