

**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**Facultad de Ingeniería en Electricidad y Computación**

Desarrollo de una aplicación web para la automatización de  
configuraciones de administración de redes en un ISP

**PROPUESTA DE PROYECTO INTEGRADOR**

Previo la obtención del Título de:

**Ingeniero(a) en Telemática**

Presentado por:

María Carolina Aguilar Alvarado

Alex Alberto Córdova Balón

GUAYAQUIL - ECUADOR

Año: 2021

## **DEDICATORIA**

En primer lugar, dedico la culminación de mi etapa universitaria a mis padres Cesar y Graciela quienes con todo su esfuerzo y amor me han permitido llegar a cumplir hoy una meta más en mi vida, gracias por estar junto a mí en cada paso, guiándome con su ejemplo y buenos valores.

Le dedico a mis hermanos Cesar y Daniel por su apoyo incondicional, por ser mi ejemplo para seguir, cuidándome y enseñándome en los momentos más complicados que he afrontado. A toda mi familia por su cariño y apoyo en cada fase de mi vida, siempre alentándome a seguir mis sueños fomentando mi crecimiento.

Finalmente dedico este trabajo a Laura, Valeria, Alía y Gabriel porque con su amistad, amor incondicional y sus consejos pude afrontar obstáculos en mi camino y seguir adelante.

**Aguilar Alvarado María Carolina**

## **DEDICATORIA**

El presente proyecto es dedicado a mi madre, quien estuvo apoyándome en toda mi ruta de aprendizaje desde muy pequeño con su amor incondicional. A mis hermanas que, sin importar las condiciones, están dispuestas a ayudar y mejorar mediante consejos y oportunidades, mi crecimiento estudiantil. A todos mis maestros y compañeros quienes me acompañaron en mi proceso de crecimiento sin importar la dificultad y oportunidad presentada, sentando las bases de la responsabilidad y deseos de superación recibidos durante la construcción de mi vida profesional.

Finalmente, a todos mis amigos y colegas de trabajo que fui conociendo durante todo este proceso de formación profesional, quienes compartían sus conocimientos y enseñanzas en las distintas actividades que realizábamos. Esta gran familia que forme durante todos estos años son la base de todo este crecimiento.

**Córdova Balón Alex Alberto**

## **AGRADECIMIENTOS**

A mi tutora Collaguazo Jaramillo Adriana Elisa. Gracias a sus enseñanzas durante la carrera y consejos formaron parte importante de esta historia con sus aportes profesionales y creativos buscando siempre ideas innovadoras. A ustedes mis profesores, gracias por compartir sus conocimientos invaluable, por su dedicación a la enseñanza y tolerancia, por nutrirme y formarme como una profesional lista para dar lo mejor de mí poniendo en alto la institución.

A mis padres, sus consejos y valores, su guía en cada paso del camino, son quienes estuvieron siempre junto a mí día y noche durante mis horas de estudio. Inculcándome dedicación y responsabilidad que me ha llevado a culminar mi carrera con éxito. Gracias por creer en mí. A mis amigos y compañeros, quienes han recorrido esta travesía conmigo, por dar lo mejor de ustedes para triunfar en cada meta que nos hemos propuesto.

**Aguilar Alvarado María Carolina**

## **AGRADECIMIENTOS**

Mi más sincero agradecimiento a nuestros compañeros de carrera y de diferentes áreas quienes estuvieron presentes para apoyar y compartir mis logros que son resultado del trabajo constante en la formación profesional.

A mi tutora Collaguazo Jaramillo Adriana Elisa, quien nos introdujo y realizo la mentoría de nuestro trabajo paso a paso, corrigiendo e instruyéndonos para una mejora del proyecto final.

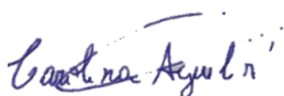
A cada uno de los Clubes, Argumentum, NIoT y aquellos otros en los que participe por distintas actividades, pues supieron enseñar y potenciar mis habilidades y destrezas durante todo este proceso estudiantil.

Finalmente, a la universidad Escuela Superior Politécnica del Litoral, quien me dio la bienvenida al mundo como tal, las oportunidades que me ha mostrado y he podido aprovechar son incomparables.

**Córdova Balón Alex Alberto**

## DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; *Aguilar Alvarado María Carolina y Córdova Balón Alex Alberto*, damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual”



Aguilar Alvarado  
María Carolina

Córdova Balón  
Alex Alberto

## **EVALUADORES**

.....  
**Vladimir Sánchez Padilla**

PROFESOR DE LA MATERIA

.....  
**Adriana Collaguazo Jaramillo**

PROFESOR TUTOR

## RESUMEN

El presente proyecto integrador muestra el análisis, desarrollo e implementación de una aplicación web que brinda el servicio de soporte y migración, entre distintos modelos de dispositivos intermediarios como lo son ruteadores, desarrollada principalmente para los ISP, empresas u organizaciones que manejan una red con múltiples dispositivos y desean hacer una actualización. Debido a que previo al desarrollo de este proyecto, se realizaban este tipo de tareas rutinarias mediante distintas aplicaciones o macros de Excel, se presenta este servicio como una mejor alternativa para la automatización de estos procesos.

La implementación de la presente propuesta fue desarrollada en Angular para Front-End y Python para Back-End haciendo uso de múltiples herramientas que permiten el manejo de configuraciones en dispositivos de red. Además, el uso de servicios en la nube permitió que la distribución o venta del servicio pueda llevarse a nivel nacional e internacional, pues este problema se presenta en múltiples proveedores de internet en distintas ciudades. La plataforma web NetMi, fue el resultado de los estudiantes de la carrera Ingeniera en Telemática en proponer una solución que permita agilizar las tareas rutinarias de este medio de trabajo, y esperando que distintas empresas la adopten para una optimización de tiempo y recursos en sus distintas áreas.

Como resultado, la implementación de esta plataforma web permitió un mejor manejo de estas tareas de rutina mediante un entorno más cómodo para los usuarios o ingenieros de los departamentos de tecnología. Dejando atrás todos esos modelos de migración antiguos que tomaban mucho tiempo, reduciendo significativamente esta variable gracias a una conexión rápida vía SSH con dispositivos intermedios y el aplicativo en cuestión, retornando un archivo con nuevas configuraciones.



Concluyendo, actualizar dispositivos intermediarios de una red de distintos modelos y sistemas operativos puede llevar un periodo de tiempo mucho más corto incluso si se considera a un profesional técnico revisando los datos retornados por la plataforma, todo esto es manejado con normas de seguridad como es un acceso restringido pues enfatizamos nuestro producto hacia un tipo de cliente como lo es un proveedor de servicio de internet.

Palabras Clave: Proveedor de servicio de internet, Migración de red, Ruteador.

## **ABSTRACT**

This capstone project shows the analysis, development, and implementation of a web application that provides the support and migration service between different models of intermediary devices, such as routers, developed mainly for ISPs, companies, or organizations that manage a network with multiple devices and want to do an update. Before this project development, these routine tasks were carried out using different Excel applications or macros. Based on the aforementioned, we present this service as an alternative for automating these types of processes.

The implementation of this proposal was based on Angular for Front-End and Python for Back-End, using multiple tools that allowed the management of configurations in network devices. In addition, the cloud services allowed the distribution or sale of the service either nationwide or worldwide, as this problem occurs in multiple Internet providers along with several scenarios. The NetMi web platform resulted as a solution for streamlining the routine tasks of this work environment, which aims to optimize assets and resources in their different areas.

As a result, the web platform deployment allowed better management of these routine tasks through a more comfortable environment for engineering and technical staff at technology departments. Moreover, this will take off old migration models that consumed a lot of time, significantly reducing this variable thanks to a fast connection via SSH with intermediate devices and the application described, returning a file with new configurations.

In conclusion, updating intermediary networking devices from different models and operating systems can take lesser periods even if a technical staff considers reviewing the data returned by the platform. Thus, this is handled through security standards, such as restricted access, because we emphasize our product towards

ISPs-type clients. Finally, we look forward to the adoption of this proposal in networking-related ventures or well-established providers.

Keywords: Internet Service Provider, Migration frameworks, Networking migration, Routing.

## ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
ISP	Internet Service Provider
TI	Tecnologías de la Información
ARP	Address Resolution Protocol
BFD	Bidirectional Forwarding Detection
BGP	Border Gateway Protocol
VRF	Virtual routing and forwarding
NAT	Network Address Translation
VLAN	Virtual LAN, Red de área local virtual
SDN	Software Designed Networking
SDH	Synchronous Digital Hierarchy
SONET	Red óptica síncrona
API	Interfaz de programación de aplicaciones
MEFE	Matriz de Evaluación de los Factores Externos
MEFI	Matriz de Evaluación de los Factores Internos
MPC	Matriz de perfil competitivo
MIE	Matriz interna y externa
vCPU	Virtual Central Processing Unit
NAPALM	Network Automation and Programmability Abstraction Layer with Multivendor support
VROOM	Virtual Routers On The Move

## ACRÓNIMOS

Hardware	Parte física que constituye un sistema informático
Software	Programas y rutinas para realizar una tarea en un sistema
Back-End	Parte lógica del desarrollo de una aplicación
Front-End	Parte visual del desarrollo de una aplicación
prefix-list	Comando para creación de listas para filtro de prefijos.
service-instance	Comando de habilitación de puerto en capa 2 y capa 3
bridge-domain	Comando de habilitación de flujo bidireccional de tráfico.
OpenFlow	Es una tecnología de switching.
Switching	El movimiento tramas de un sitio a otro dentro de una red local.

# ÍNDICE GENERAL

DEDICATORIA.....	I
DEDICATORIA.....	II
AGRADECIMIENTOS .....	III
AGRADECIMIENTOS .....	IV
DECLARACIÓN EXPRESA.....	V
EVALUADORES .....	VI
RESUMEN .....	VII
ABSTRACT .....	IX
ABREVIATURAS.....	XI
ACRÓNIMOS .....	XII
ÍNDICE GENERAL .....	XIII
ÍNDICE DE FIGURAS .....	XVI
ÍNDICE DE TABLAS .....	XVII
CAPÍTULO 1 .....	1
1. Introducción.....	1
1.1 Definición del problema.....	2
1.2 Objetivos .....	4
1.2.1 Objetivo General.....	4
1.2.2 Objetivos Específicos .....	4
1.3 Hipótesis .....	4
1.4 Descripción de escenarios .....	4
1.5 Trabajos relacionados .....	5

1.6	Marco Teórico .....	6
CAPÍTULO 2 .....		10
2.	METODOLOGÍA .....	10
2.1	Introducción al despliegue de la plataforma web .....	10
2.2	Tecnologías y herramientas usadas en la implementación.....	15
2.2.1	Entorno de desarrollo .....	15
2.2.2	Desarrollo de Front-End .....	15
2.2.3	Librerías de desarrollo de redes de datos .....	16
2.2.4	Almacenamiento de datos .....	16
2.2.5	Producción de la aplicación .....	16
2.2.6	Emulador del diseño de la red .....	17
2.3	Implementación del proyecto .....	18
CAPÍTULO 3 .....		20
3.	RESULTADOS Y ANÁLISIS .....	20
3.1	Diagnóstico estratégico externo e interno .....	20
3.2	Análisis de costos.....	27
3.3	Etapas pre-operativas.....	27
3.4	Costos de diseño y desarrollo .....	27
3.5	Costos de actualización y mantenimiento .....	30
3.6	Análisis costo/beneficio .....	30
3.7	Restricciones y limitaciones de funcionalidad en la plataforma web .....	31
CAPÍTULO 4 .....		32
4.	CONCLUSIONES Y RECOMENDACIONES .....	32
4.1	Conclusiones.....	32
4.2	Recomendaciones.....	33

4.3 Trabajo a futuro .....	34
BIBLIOGRAFÍA .....	35
ANEXO A.....	38
ANEXO B.....	53



# ÍNDICE DE FIGURAS

Ilustración 2.1. Diagrama del proyecto propuesto de plataforma web NetMi .....	10
Ilustración 2.2. Interfaz gráfica de la plataforma web. ....	11
Ilustración 2.3. Resultado de migración de configuraciones VRF. ....	14
Ilustración 2.4. Diagrama de red .....	17
Ilustración 2.5 Esquema de la plataforma web.....	19

# ÍNDICE DE TABLAS

Tabla 1 Diseño de tabla o modelo en base de datos .....	12
Tabla 2 Diseño de la solicitud.....	13
Tabla 3 Tabla del requerimiento de versiones.....	18
Tabla 4 Matriz FODA.....	21
Tabla 5 Matriz MEFE.....	22
Tabla 6 Matriz MEFI .....	24
Tabla 7 Matriz MPC.....	26
Tabla 8 Propuesta económica planteada para venta del servicio. ....	28

# CAPÍTULO 1

## 1. INTRODUCCIÓN

Redes de datos y programación de sistemas son los encargados de diseñar y desarrollar sistemas para la optimización de tareas rutinarias en la rama de telecomunicación e informática, tales como, la configuración de plantillas en los dispositivos de red y generación de archivos de configuración de respaldo [1]. Esta automatización en las distintas áreas de tecnología de la información (TI) se ha convertido en un requerimiento para los cientos de tareas repetitivas y de rutina que solo consumen tiempo y energía cuando se lo podría invertir en la mejora y crecimiento de la misma red en la que se trabaja.

Por esto, organizaciones dedicadas a este tipo de tareas demandan que existan tiempos rápidos en la implementación, además de ser eficientes y seguros, de las soluciones brindadas en dispositivos que involucran hardware y software [2]. Dependiendo de las necesidades de estas, pueden encontrar diversos proveedores de servicios y dispositivos de red a nivel mundial, donde el mercado se encuentra dividido en, Cisco con 56.4%, Huawei con 9.5%, HPE con 5.2%, Juniper con 3.8% y las demás empresas con 25.1%, datos estadísticos calculados según la facturación realizada en el año 2019, donde Cisco lidero el sector seguido de Huawei. Información presentada por Statista el 31 de mayo del 2020 [3]. Debido a estos equipos, tenemos la posibilidad de conectarnos mediante los múltiples protocolos a internet.

Se considera que los proveedores de servicios de Internet son aquellos que adoptan estos dispositivos en gran cantidad, realizando diversas configuraciones en cada uno de ellos, formando una red local encargada de brindar el servicio de conectividad a diversos usuarios [4]. Entre ellos encontramos, CISCO y HUAWEI, los cuales cada cierto tiempo implementan nuevos mecanismos de ahorro de

energía en sus enrutadores con nuevas y mejores características que mejoran su funcionamiento y se adaptan a las necesidades de transmisión de datos de una nueva generación [5], ayudando a la automatización de operaciones que ocurren en la red.

La migración de enrutadores de bajo consumo genera un impacto fuerte en el ahorro de recursos económico en empresas con grandes redes de datos locales, especialmente en un proveedor de servicio de internet. El impacto de la programación se vuelve más didáctica al momento de realizar las respectivas configuraciones y capacitaciones de los dispositivos manejados, este planteamiento es la base de una gran investigación planteada y publicada en la IEEE donde incentiva a las instituciones o empresas a realizar las actualizaciones respectivas de todos o gran parte de los dispositivos intermediarios que manejan, debido a los beneficios que se presentan a los departamentos dedicados en esta área respectiva [5].

## **1.1 Definición del problema**

Un ruteador conocido también como encaminador, es un dispositivo intermediario que permite la interconexión de redes de datos. Desde un punto de vista funcional, se puede concebir como una computadora de propósito específico. Por consiguiente, estos equipos son considerados como dispositivos intermediarios [25].

Realizar una migración de la configuración actual de cada dispositivo intermediario, siempre fue problema y un consumo de recursos [6]. Un Proveedor de servicios de internet que mantiene dispositivos de antigua generación necesita remplazarlos por nuevos y actuales que proveen nuevas y mejores especificaciones que sus modelos anteriores como, el manejo de protocolos, velocidad de, transmisión y procesamiento de datos, cumplimiento de normas, entre otras.

Si se requiere hacer una migración de configuraciones establecidas de un Cisco Router IOS XE a un Cisco Router XR [7], al ser de diferentes modelos de dispositivos intermediarios, no se puede realizar una exportación e importación

común dado que los dispositivos no son compatibles y las capas a migrar son dependientes entre sí, lo que involucra a diversos servicios, por lo que la empresa cliente realiza este proceso mediante diversos documentos de macro en Excel.

Este procedimiento se realizaba mediante varias hojas del archivo preparadas con las configuraciones necesarias para extraer la información requerida. Por consiguiente, se realiza un análisis y configuración de protocolos de enrutamiento como VRF's, L2VPN, VFI's, enrutamiento estático y dinámico, Prefix, Route-Map, entre otro tipo de requerimientos que se desarrollan manualmente.

Cada uno de estos dispositivos intermediarios, con sus viejas configuraciones y modelos mantienen un gasto exorbitante de energía, ancho de banda, entre otras desventajas en la infraestructura de red de datos para la realización de tareas rutinarias. Asimismo, estos problemas comunes han sido planteados por la investigación *"Migration to energy efficient routers: Where to start?"* razón por la que se reemplazan los dispositivos intermediarios por nuevos modelos, requiriendo un plan de migración y todo el proceso que conlleva este tipo de actividades para las áreas de TI [4].

Este inconveniente ha sido presentado en pequeña escala, sin embargo, al tratarse de un proveedor de servicio de internet, que mantiene diversos dispositivos que requieren esta actualización, es necesario que el proceso sea automatizado y realizado por bloques, dado que cada uno de ellos posee una gran cantidad de líneas de configuración.

Así mismo, esta situación se presenta en múltiples empresas que manejan una cantidad considerable de dispositivos con diversas capas de enrutamiento y que requieren una actualización para mejorar sus sistemas y servicios. El proceso de migración mantiene los mismos pasos, sin embargo, los dispositivos intermediarios varían sus marcas, modelos y versiones de sistema operativo interfiriendo la compatibilidad entre la exportación de configuraciones y su importación [8].

## **1.2 Objetivos**

### **1.2.1 Objetivo General**

Diseñar una plataforma web con servicios en la nube que automatice las migraciones de configuraciones en dispositivos intermediarios mediante herramientas de software y servicios en la nube.

### **1.2.2 Objetivos Específicos**

1. Identificar las operaciones y configuraciones de los dispositivos intermediarios para una definición de un patrón de comportamiento.
2. Automatizar las configuraciones de los dispositivos intermediarios en un lenguaje más amigable y dinámico para la lectura y programación de una nueva configuración que será migrada.
3. Implementar una plataforma web que disminuya el proceso de la automatización de migraciones de configuraciones de dispositivos intermediarios.

## **1.3 Hipótesis**

Automatizar la migración de dispositivos intermediarios disminuye los costos operativos para la administración de redes de datos, tales como tiempo y recurso humano dedicado a este proceso.

## **1.4 Descripción de escenarios**

En primera instancia, se definirá un patrón de comportamiento de los dispositivos intermediarios utilizados, posteriormente se procederá a traducir estas configuraciones en un lenguaje que permita migrar a un dispositivo con diferente sistema operativo, por ejemplo, de Cisco IOS XE hacia Cisco IOS XR, y finalmente se actualizará la configuración requerida en el dispositivo.

La implementación está basada en un sistema de gestión de redes de datos para que el administrador pueda automatizar las migraciones en dispositivos de la red de

manera más eficiente. Por tal razón, se desplegará una plataforma web con modelo arquitectónico REST [9], que permita obtener la información de los dispositivos intermediarios y como resultado, procesarla para tener un lenguaje más amigable que pueda facilitar los parámetros importantes y modelo de configuración.

## **1.5 Trabajos relacionados**

La migración de configuraciones en dispositivos de telecomunicación son actividades que se han realizado a lo largo de los años en las empresas, por lo tanto, se han propuesto distintos sistemas o servicios como apoyo para estas tareas rutinarias. Un ejemplo de un sistema dedicado a la migración de configuraciones es Éxodus, proyecto de grado de la Universidad de Brown, implementado con Cisco IOS usando SDN y OpenFlow para la exportación de las líneas de configuración de filtrado de paquetes, listas de acceso, NAT, VLAN, enrutamiento estático y dinámico. Este servicio consume una colección de configuraciones de enrutadores haciendo el desarrollo de estas tareas más práctico mediante un controlador SDN usando programas centralizados que son verificables y evolucionales. En consecuencia, este proyecto expuso varias limitaciones tanto en los lenguajes actuales para la programación SDN como en el propio OpenFlow [10].

Por otra parte, existen múltiples organizaciones que aún manejan una comunicación SDH y SONET, que representan tecnologías antiguas provenientes de dispositivos intermediarios en sus primeros modelos que hoy ya no se compran, por lo que se están remplazando en esta década y la siguiente por infraestructuras SDN.[23] Estas migraciones llevan varios meses o incluso años, con el fin de evitar que el servicio brindado sea interrumpido, provocando que empresas opten por el uso de algoritmos o APIS que reduzcan el problema. Puesto que se desea minimizar la cantidad de interrupciones de la red que se generan a lo de una migración de red. En consecuencia, esto es considerado como un conjunto de problemas de migración que es estudiado y, planteando un cronograma técnico se puede reducir el costo general de este proceso satisfaciendo las limitaciones técnicas y operativas de los dispositivos intermedios [11].

Algunas instituciones, especialmente los proveedores de servicio de internet, cuando desean realizar la administración de una red más compleja y con múltiples dispositivos, recomiendan como nueva tecnología los VROOM que permite realizar las pruebas y los cambios necesarios en una topología lógica, además de abordar situaciones donde se realicen cambios emergentes. Varias organizaciones usan esta tecnología para el diseño, la implementación y evaluación de técnicas de migración, de esta manera al momento de ser enviado a producción ya sea en hardware o software contendrá errores mínimos y en un menor tiempo reduciendo el impacto en el tráfico de datos en la red. La implementación y pruebas realizadas en este proyecto mostró transparencia en los protocolos de enrutamiento y no generó ningún impacto en el rendimiento del tráfico de datos [12].

## **1.6 Marco Teórico**

Las múltiples empresas dedicadas a la fabricación, venta, mantenimiento y consultoría de dispositivos intermediarios, como son CISCO, HUAWEI, HPE, JUNIPER, entre otros [3]. Son aquellos que controlan el mercado donde anualmente realizan un sin número de ventas, este tipo de productos, constantemente se encuentran en la mejora de sus protocolos, interfaces, puertos, nuevas instrucciones de configuración o comandos, y otras características que facilitan y optimizan el trabajo o servicio de la comunicación en una red de datos.

Cada uno de estos dispositivos son adquiridos principalmente por los proveedores de servicio de internet que brindan una conexión o comunicación con Internet a sus clientes, principalmente por sus diferentes tecnologías manejadas (ADSL, cabledem, etc) y el usuario a través de un computador personal y su modem, proporcionando diferentes velocidades de conexión [13].

Por otra parte, las empresas han tenido que adaptarse a esta evolución esquemática de la historia tecnológica, y crear sus propias redes internas de telecomunicación mejorando así su producción y desarrollo en las distintas áreas



de su organización [14]. Esta infraestructura es desarrollada e implementada gracias a los dispositivos brindados por los distintos proveedores mencionados previamente.

La revolución industrial ha avanzado de tal modo que el Internet ofrece medios para la organización de empresas convirtiéndose en el flujo económico de estas, todas estas tareas la realizan los ordenadores encargados de almacenar, analizar y comunicar información instantáneamente, haciendo que una red de telecomunicación interna y conectividad a Internet sea cada vez más indispensable. [14]. Diversas plataformas, aplicaciones y programas desarrollados por comunidades programadoras, buscan mejorar el proceso de estas tareas rutinarias que los dispositivos intermedios manejan. Es decir, nuestro proyecto debe ser capaz de automatizar estos procesos previo a un análisis y comunicación entre librerías o dependencias.

Según CISCO, uno de los mayores problemas para la administración de redes de datos es el crecimiento de datos y dispositivos, lo que conlleva a un aumento de costos de mantenimiento, mismos que en su gran mayoría se realiza de manera manual, por lo cual la automatización es esencial para la transformación digital que afronta el mundo [15]. Uno de los principales inconvenientes de este mantenimiento es la adaptación de nuevos dispositivos, su reemplazo y reconfiguración, pues al ser más actuales y con nuevas características, la migración de su configuración varía en muchos aspectos, especialmente en los comandos manejados. Este problema es mucho más visible cuando existe un mayor lapso en la fecha de lanzamiento entre los dispositivos implicados, una versión anterior versus una nueva con características adiciones y mejoradas.

El problema definido se presenta en la compatibilidad entre dispositivos dado que se van obteniendo nuevos dispositivos intermediarios que no pueden comunicarse entre ellos por sus modelos o distintas versiones de sistema operativo, además de protocolos e interfases, lo cual provoca una configuración manual cayendo en la

misma situación de empresas que van actualizando sus sistemas o migraciones de dispositivos [16].

El cambio de configuraciones entre dispositivos intermediarios de diversos fabricantes implica un análisis en la metodología que estos implementan, operaciones de enrutamiento que mejoran la seguridad, calidad de servicio, envío de datos y funcionamiento de redes de datos con alcance empresarial.

Las nuevas librerías y tecnologías como, Cisco PyATS trabajan en conjunto con Python, con el desarrollo de software Cisco para la automatización de dispositivos como ruteador, conmutador, puntos de acceso, firewalls, entre otros. De manera que, junto a otras herramientas de integración como librerías se analiza las configuraciones de los dispositivos intermediarios, logrando tener un formato más fácil de interpretar con la ventaja de ser manipulable [17]. Cabe recalcar que este procedimiento es basado en el desarrollo de Cisco, el objetivo de NetMi es escalar con diferentes marcas del mercado que pueda realizar las mismas funciones de redes de datos dada por PyATS.

El enfoque por el cual se implementa Cisco PyATS es debido a la solución de la biblioteca Genie que contiene herramientas de automatización evitando programación funcional, cuenta con un amplio conjunto de funciones para analizar la información, de modo que se pueda obtener la configuración de un dispositivo en formato de diccionarios y establecer una relación con una traducción a otro sistema operativo.

En Genie encontramos una lista de analizadores por medio de comandos comunes de los sistemas operativos de Cisco, Junos, Linux y entre otros, la cual abarca desde listas de accesos, información de las interfaces, detalle de los protocolos tales como ARP, BFD, BGP, entre otros, e información de aplicaciones de enrutamiento.

Utilizando estas herramientas, generamos servicios en la nube que facilitan la migración de configuraciones de un dispositivo intermediario a otro, considerando sus distintas versiones de sistema operativo y modelos. Además, mediante la librería de Genie, podemos complementar esta solución adaptándola a distintos fabricantes.

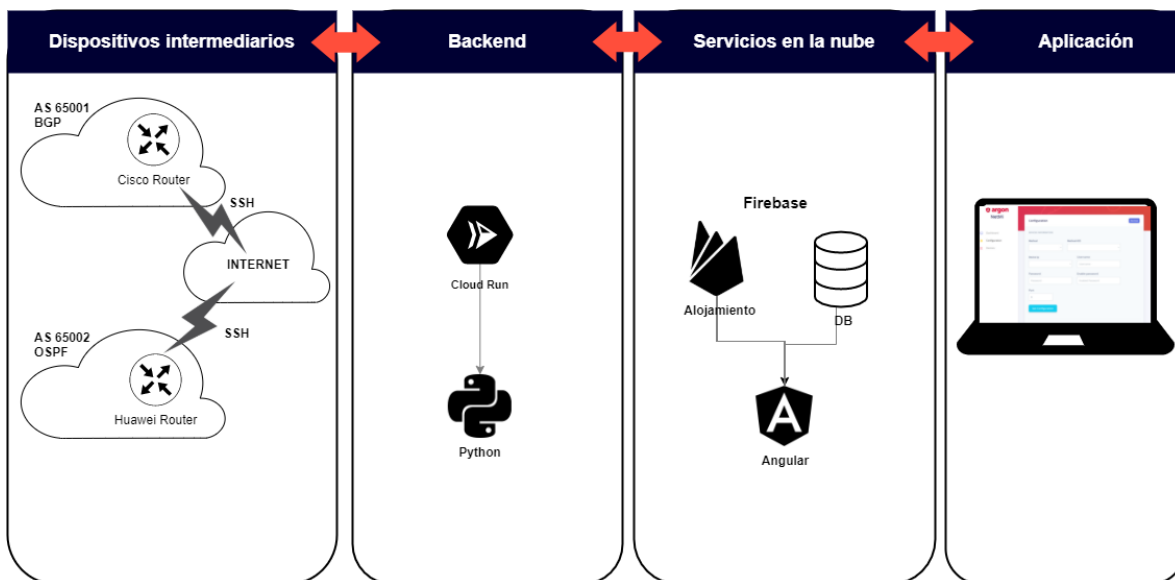
Este tipo de servicios ofertados mediante una plataforma web, con un ingreso exclusivo para hacer uso de las herramientas desarrolladas es lo que permite su comercialización, considerando medidas para el manejo, control y soporte de la plataforma [18].

# CAPÍTULO 2

## 2. METODOLOGÍA

### 2.1 Introducción al despliegue de la plataforma web

El diagrama del proyecto propuesto para la plataforma web NetMi, y el despliegue de la plataforma web está representado en la ilustración 2.1, el Front-End se caracteriza por estar basado en un entorno de trabajo para desarrollo de aplicaciones web de Angular desplegado en Firebase, por otro lado, el Back-End está desarrollado en lenguaje de Python de modo que se comunica de forma remota con la red del proveedor de servicio de internet y los dispositivos intermediarios designados para entregar la traducción de las configuraciones de redes de datos. De manera que, el proyecto se despliega en Cloud Run, mientras que, el alojamiento de Firebase permite la comunicación con su propia base de datos para alojar la información necesaria de los dispositivos de la red.

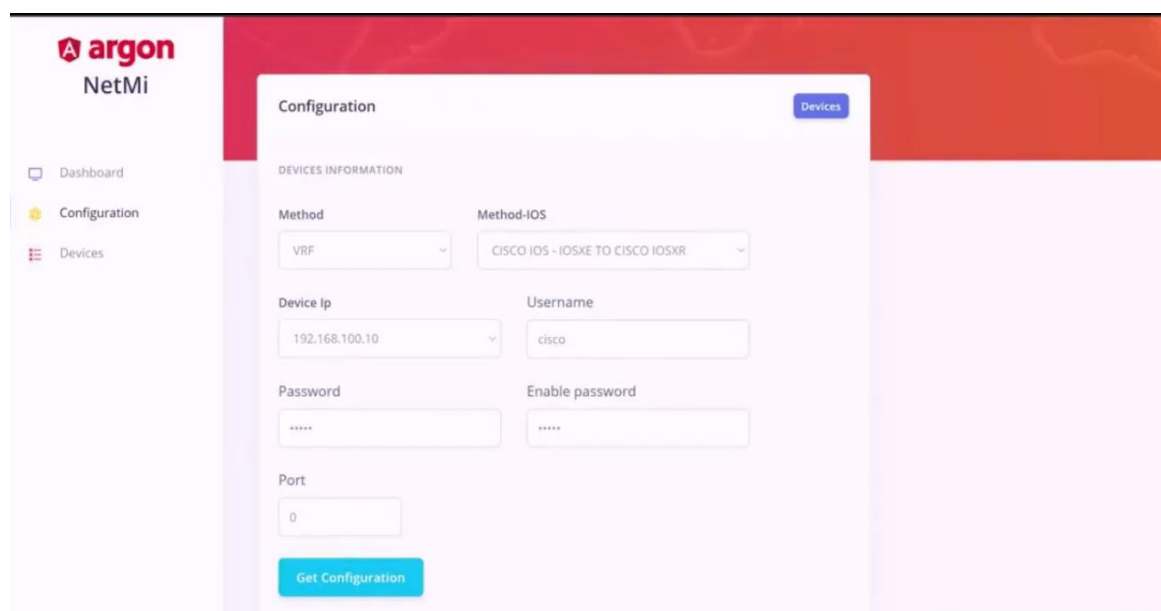


*Ilustración 2.1. Diagrama del proyecto propuesto de plataforma web NetMi*

*Fuente: Elaboración propia con referencia [26]*

Los dispositivos intermediarios planteados, representan aquellos equipos como ruteadores, sobre los cuales va a trabajar esta importación y exportación de líneas de configuración durante todo el proceso de la migración. Estos se encuentran en una red local ya sea de la empresa u organización como el proveedor de servicio de internet, proporcionando una conexión SSH para proceder a realizar las respectivas consultas de datos al denominado dispositivo antiguo de la red, de igual forma se debe proporcionar una conexión segura al equipo a migrar para poder exportar las líneas de configuración actualizadas y modificadas por el sistema.

El usuario podrá encontrarse con un entorno amigable desde su equipo personal mediante la plataforma web desplegada previamente. Este entorno es denominado el panel de trabajo, desarrollado en Angular, mejorando el proceso de migración y proporcionando los datos o líneas de configuración de dispositivos intermediarios sobre los que se trabaja, exportando un archivo con las nuevas directrices del nuevo modelo a migrar. De esta forma se mantiene al usuario retroalimentado del procedimiento, permitiéndole manipular algún cambio extra que desee realizar, para más detalle de las funcionalidades de la plataforma web revisar el manual de interfaz del Anexo B.

The image shows a web application interface for 'argon NetMi'. On the left is a sidebar with a menu containing 'Dashboard', 'Configuration', and 'Devices'. The main content area is titled 'Configuration' and includes a 'DEVICES INFORMATION' section. This section contains several form fields: 'Method' (a dropdown menu with 'VRF' selected), 'Method-IOS' (a dropdown menu with 'CISCO IOS - IOSXE TO CISCO IOSXR' selected), 'Device Ip' (a dropdown menu with '192.168.100.10' selected), 'Username' (a text input field with 'cisco'), 'Password' (a text input field with masked characters), 'Enable password' (a text input field with masked characters), and 'Port' (a text input field with '0'). At the bottom of the form is a blue button labeled 'Get Configuration'. A 'Devices' button is also visible in the top right corner of the configuration panel.

*Ilustración 2.2. Interfaz gráfica de la plataforma web.*

Cada uno de estos dispositivos que se van ingresando en la plataforma son registrados en una base de datos proporcionada por el servicio en la nube de Firebase. Esta base de datos contiene una tabla o modelo con la siguiente estructura, detallando tanto nombre del elemento, tipo de dato, descripción y tipo de clave como se muestra a continuación.

Nombre	Tipo	Nombre descriptivo	Clave
<b>id</b>	Var char	identificador	PK – Primaria
<b>ip</b>	Var char	Dirección ip	
<b>name</b>	Var char	Nombre	
<b>encryp</b>	Var char	Encriptación	
<b>model</b>	Var char	Modelo	
<b>protocol</b>	Var char	Protocolo	
<b>os</b>	Var char	Sistema Operativo	

*Tabla 1 Diseño de tabla o modelo en base de datos*

En primer lugar, el ID será un identificador en la tabla de cada dispositivo intermediario en nuestra base de datos, conocido como clave principal. La dirección IP del equipo, su nombre, el tipo de encriptación con el que trabaja, el modelo como control de equipos manejados, el protocolo usado durante la conexión y el sistema operativo del dispositivo. Todos estos atributos mantienen un tipo de dato CHAR.

Se implementará una arquitectura de microservicios por medio del entorno de trabajo de Flask en el cual se realizará la conexión con los dispositivos de red usando las funciones de las librerías de automatización de redes de datos. Con esto, se obtiene la información que se requiere configurar, transformándola en lenguaje del sistema operativo de Cisco o de Huawei. Por último, se procede a entregarla en el servicio como un archivo de configuración en la interfaz del usuario.

La librería Flask permite comunicar nuestro servicio Back-End con el Front-End mediante una solicitud de método POST, que al ser una solicitud compatible con HTTP permite realizarlas por el navegador o un aplicativo similar como Postman. Esta solicitud conlleva un identificador de la sesión y los múltiples parámetros que representan al dispositivo con el que se está trabajando. Su detalle en la tabla 2 y en manual para administración de microservicios se encuentra la información que contiene cada uno de estos parámetros por solicitudes realizadas en el Front-End.

Información de la solicitud		Descripción
<b>Método</b>	POST	Método http
<b>Identificador</b>	path	Ruta de la URL
<b>Parámetros</b>	name	Nombre del dispositivo
	username	Nombre de usuario
	password	contraseña
	enable	Contraseña de CLI
	protocol	Protocolo de comunicación SSH/telnet
	encryp	Tipo de encriptación SSH
	os	Sistema operativo
	port	Puerto
	ip	Dirección IP
	show	Comando para mostrar
	show2	Comando para mostrar
	plantilla	Plantilla de configuración

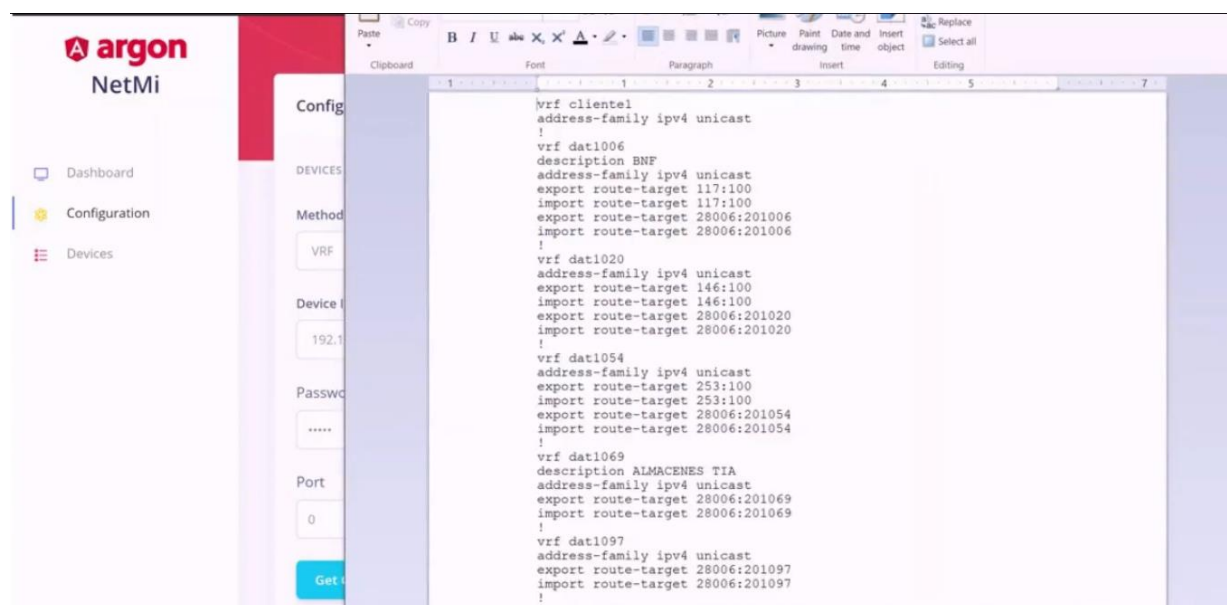
*Tabla 2 Diseño de la solicitud*

La solicitud se realiza con una ruta URL y a continuación los parámetros que recibirá nuestro Back-End para realizar la comunicación con el dispositivo intermedio y generar su respuesta, que son, el nombre del dispositivo, nombre de usuario y contraseña que permite la comunicación vía SSH, aviso de la contraseña CLI, el

protocolo a trabajar en este proceso de migración, el tipo de encriptación, el sistema operativo del modelo del equipo junto con su puerto y dirección IP. Varios de estos parámetros son los mencionados en el proceso de base de datos en la tabla 1. Por consiguiente, se especifica los comandos para mostrar y la plantilla con la que trabajaremos para esta migración.

Cuando se realiza la conexión remota vía SSH, y se envía los datos necesarios para autenticación y consulta de información, el servicio de Back-End que trabaja con las distintas librerías de Genie y PyATS en Python, configura las nuevas plantillas que serán enviadas como respuesta de la solicitud, dependiendo del dispositivo, protocolo y demás especificaciones realizadas previo a dicha consulta.

Este archivo es entregado por la plataforma web con las nuevas configuraciones del dispositivo intermediario, en donde su formato depende del protocolo seleccionado. Por consiguiente, el archivo llamado config.cfg se muestra al cliente y/o usuario para que puedan comprobar la veracidad de las nuevas configuraciones, por ejemplo, declaraciones vrf, comando de importación y exportación, y de ser necesario modificarlas.



*Ilustración 2.3. Resultado de migración de configuraciones VRF.*



Una vez finalizada la descarga y verificación del nuevo archivo configurable, se procede a enviar al nuevo dispositivo intermediario para su implementación y funcionamiento en producción.

Este tipo de procesos está diseñado para ser un procedimiento óptimo, mucho más amigable con el cliente y como un servicio que se podría prestar a diversas instituciones que presenten este inconveniente como previamente se ha planteado.

## **2.2 Tecnologías y herramientas usadas en la implementación**

### **2.2.1 Entorno de desarrollo**

Docker permite la creación de contenedores para aislar cada aplicación en un entorno de trabajo, de esta forma, podemos trabajar con diferentes tecnologías y a su vez evitar sobrecarga de la virtualización de cada entorno de trabajo [19]. Debido a su excelente manejo, entorno de pruebas local y facilidad de creación de los contenedores a diferencia de las máquinas virtuales, este sistema ha sido acordado como medio presentación y entrega al cliente una vez finalizado el proyecto, por su facilidad que muestra el aplicativo. Presentando dos contenedores para su despliegue con las versiones que se encuentran en la tabla 3.

### **2.2.2 Desarrollo de Front-End**

El Front-End, se implementa Angular y Node.js, el cual es un entorno de trabajo basado en TypeScript, que permite la creación de aplicaciones web de una sola página. Consecuentemente, maneja múltiples diseños en sus plantillas y permite su ejecución como aplicación de Node, permitiendo una administración del proyecto como las dependencias mediante su configuración en un archivo denominado package.json. Por tal razón, esta herramienta, al ser manejada mediante TypeScript, un subconjunto de JavaScript es una tecnología adecuada para su implementación y desarrollo.

Después, se despliega en un servicio de alojamiento de Firebase para que el usuario pueda acceder a la plataforma web, además ayuda en la escalabilidad que puede llegar a presentar la API e implementar un servicio de dominio por parte de la empresa [20].

### **2.2.3 Librerías de desarrollo de redes de datos**

El Back-End se encuentra implementado en un lenguaje de programación basado en objetos de Python dado a que encontramos recursos de librerías especializados en redes de datos como PyATS, que es un desarrollo de Cisco para la automatización de sus productos con la colaboración de Genie para automatizar y estructurar las configuraciones en interfaces, adicionalmente, se requiere la librería de Jinja2 para implementar plantillas de las configuraciones [17]. Se considera que cada una de estas librerías usadas mantiene unas dependencias con otras que se deben especificar en un archivo al momento de la instalación.

### **2.2.4 Almacenamiento de datos**

Es una plataforma en la nube, que permite el desarrollo, implementación y entorno de producción de aplicaciones web y móviles. Maneja varias herramientas que simplifica las tareas de gestión de un servicio web [21]. Por esta razón, ha sido usada como entorno de producción de la plataforma Web, almacenando datos durante sus ejecuciones, como lo son los especificados en la tabla 1.

### **2.2.5 Producción de la aplicación**

Cloud Run permite el desarrollo e implementación de contenedores o aplicativos que prometen una buena escalabilidad. La ejecución de estos contenedores permite realizar solicitudes HTTP hacia la implementación Back-End, permitiendo la respuesta y constante comunicación con Front-End.

### 2.2.6 Emulador del diseño de la red

Para realizar las pruebas de estudio se implementó un entorno de simulación con el software GNS3, donde se construirá una red con los dispositivos intermediarios con las configuraciones requeridas mediante conexión a un servidor local [22]. Esto permitirá conectarse mediante SSH desde la plataforma web para realizar todo el procedimiento requerido.

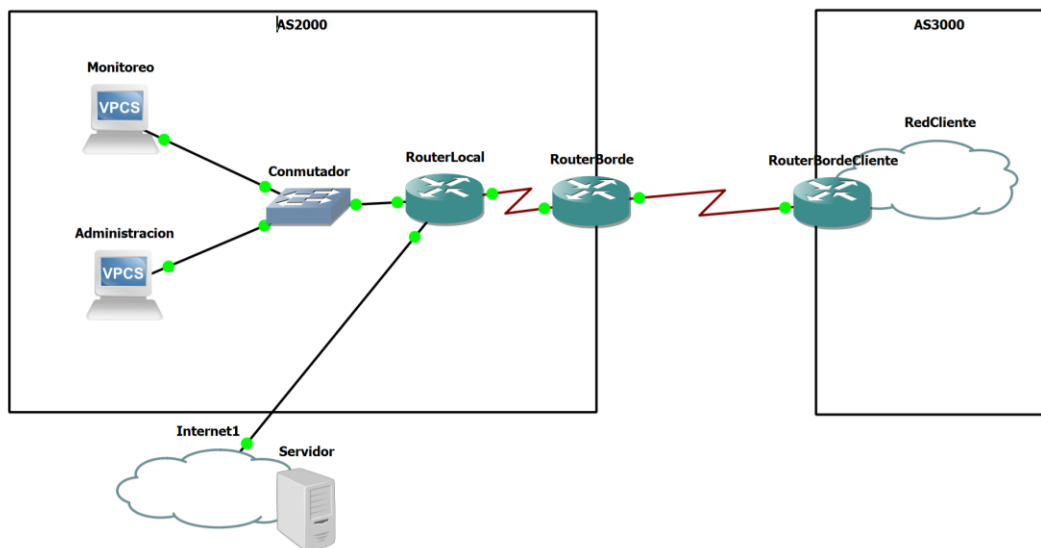


Ilustración 2.4. Diagrama de red

La Ilustración 2.4 visualiza una arquitectura simplificada de una red de proveedor de servicios de internet la cual maneja protocolos de enrutamiento para poder entregar un camino para que el enrutamiento de los datos acceda a internet, uno de estos es el protocolo BGP. En una red empresarial como la de un proveedor de servicio de internet, se cuenta con funcionarios dedicados a monitoreo y administración de los equipos, es en esa instancia en que los ingenieros de telecomunicaciones pueden realizar sus tareas para dar un servicio de calidad a los clientes.

Cada una de estas herramientas ha sido seleccionada por los beneficios, y facilidades que otorgaban durante la instalación y pruebas de entorno generados durante el desarrollo del servicio web. Las versiones y nombres de cada una de las

herramientas de software para la programación tales como librerías, entornos de trabajo y servicios en la nube usados en este proyecto se muestran en la Tabla 3.

Nombre	Tipo	Versión
<b>Docker: Nginx</b>	Contenedor: Servidor Web	1.21.0 – Alpine
<b>Docker: Python</b>	Contenedor: Lenguaje de programación	3.9.5 - Alpine3.13
<b>Docker</b>	Entorno de virtualización	20.10.6
<b>Angular</b>	Entorno de trabajo	12.0.3
<b>NodeJs</b>	Entorno de ejecución	14 – alpine
<b>Firebase</b>	Servicio en la nube	-
<b>Google Cloud Run</b>	Servicio en la nube	-
<b>GNS3</b>	Simulador Grafico de Red	
<b>Python</b>	Lenguaje de programación	3.6
<b>PyATS</b>	Librería de Python	21.6
<b>Genie</b>	Librería de Python	21.6
<b>PyYaml</b>	Librería de Python	5.4.1
<b>Jinja2</b>	Librería de Python	3.0.1
<b>Flask</b>	Librería de Python	2.0.1
<b>Flask-Cors</b>	Librería de Python	3.0.10
<b>Requests</b>	Librería de Python	2.25.1

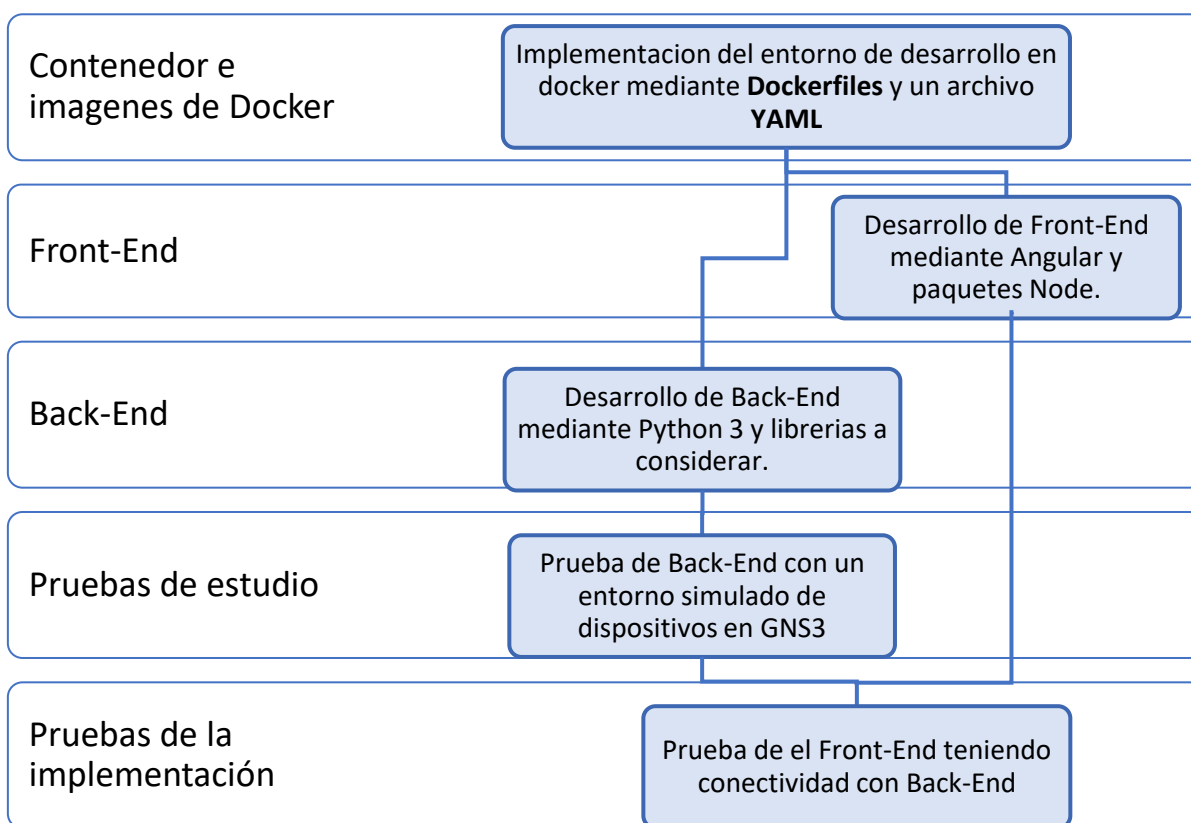
*Tabla 3 Tabla del requerimiento de versiones*

## 2.3 Implementación del proyecto

La entrega del producto desarrollado en acuerdo con el cliente se realizará mediante contenedores de Docker. En los párrafos posteriores se describe el esquema de implementación y presentación.

Acorde con las herramientas y tecnologías presentadas para la implementación de la plataforma web, se preparan los archivos YAML y Dockerfiles para el entorno de

trabajo o contenedores. Después, se lleva a cabo el desarrollo del Back-End y Front-End con las librerías y dependencias mencionadas previamente en la tabla 3. Por último, se realizan pruebas de simulación mediante una red en GNS3 con un dispositivo intermediario como es el ruteador, su correcta conexión vía SSH y exportación de las configuraciones mediante el comando show run, para posterior usar las respectivas plantillas desarrolladas en el Back-End y presentarlas al usuario final.



*Ilustración 2.5 Esquema de la plataforma web.*

Por otra parte, se ha descrito a detalle el proceso de desarrollo Back-End y Front-End, con sus plantillas respectivas y despliegue en plataformas de la nube para su respectiva replicación y entendimiento de la plataforma web. Como complemento, se recomienda ver el Anexo A, nota técnica. Además, el código desarrollado y entregado se encuentra en una organización privada en GitHub. Ver Anexo A, almacenamiento de código fuente.

# CAPÍTULO 3

## 3. RESULTADOS Y ANÁLISIS

### 3.1 Diagnóstico estratégico externo e interno

Esta sección presenta un diagnóstico para la plataforma web NetMi para conocer de una forma más clara los factores para evaluar el estado actual de la solución implementada. Se define como cliente a un proveedor de servicios de internet y como los usuarios a los ingenieros de telecomunicaciones de este.

#### 3.1.1. Matriz FODA

El análisis FODA consiste en la identificación de fortalezas, debilidades, oportunidades y amenazas para evaluar su situación interna y externa de la organización, la cual será evaluada la plataforma web NetMi

Las fortalezas y oportunidades de la plataforma web NetMi se caracterizan por la innovación de automatiza migraciones entre diferentes sistemas operativos, pero a su vez, cuenta con debilidades y amenazas dado que, a pesar de que no existen en el mercado este tipo de aplicación, Cisco entrega este servicio por medio de sus ingenieros y otras fuentes de librerías enfocadas en redes de datos se encuentra desarrollando soluciones similares.

Fortalezas	Debilidades
<ul style="list-style-type: none"> <li>• Implementación con software de Cisco.</li> <li>• Migración de configuraciones entre sistemas operativos de Cisco IOS y Cisco IOS XE hacia Cisco IOS XR y Huawei AR.</li> <li>• Aplicación de múltiples fabricantes.</li> <li>• Implementación del Back-End y Front-End con servicios en la Nube.</li> </ul>	<ul style="list-style-type: none"> <li>• Limitación de sistemas operativos de Cisco a Cisco o Huawei.</li> <li>• La ingeniería de Cisco no cuenta con desarrollo avanzado.</li> <li>• Configuración automática no es implementada por los ISP.</li> </ul>
Oportunidades	Amenazas
<ul style="list-style-type: none"> <li>• Incremento de soluciones de programabilidad de dispositivos intermediarios.</li> <li>• Mantener los dispositivos configurados es esencial para la calidad de servicio.</li> <li>• El proyecto puede escalar internacionalmente.</li> </ul>	<ul style="list-style-type: none"> <li>• Productos de Cisco u otros fabricantes con funcionamiento similar</li> <li>• Múltiples librerías similares a la ingeniería de Cisco.</li> <li>• La herramienta eNMs, software diseñado para crear soluciones de automatización de redes de datos, podría escalar para realizar migraciones.</li> </ul>

Tabla 4 Matriz FODA

### 3.1.2. Análisis de MEFE

La matriz de evaluación de los factores externos toma en cuenta las oportunidades y amenazas para la plataforma web NetMi, se procederá a asignar un peso para completar la suma total de 1 entre todos los factores, en lo que 0.01 es de menor importancia y 0.2 es de mayor importancia. Además, se clasificarán como oportunidad y amenazas entre 4 a 1, considerando 4 con mayor ponderación. El peso ponderado será la multiplicación entre el peso por la calificación, obteniendo como total la suma del peso ponderado.

En la matriz MEFE de la tabla 5, se puede observar que existe mayor peso de oportunidades dado a que la automatización de dispositivos intermediarios se encuentra en su auge por parte de estas organizaciones tal como Cisco que cuenta con librerías que están siendo desarrolladas y actualizadas constantemente. El total

de 2.65 establece que tiene una posición extrema fuerte dado a que se encuentra entre el rango de 2.5 a 3 de evaluación de factores externos. Este resultado se debe a que existen herramientas internas de estos proveedores de dispositivos intermediarios como Cisco que realizan la función de migrar configuraciones, pero se encuentra disponible para exclusividad de ingenieros de Cisco. Así mismo, otras organizaciones se encuentran en constante desarrollo de librerías de redes de datos para abastecer a múltiples proveedores de dispositivos intermediarios como NAPALM.

	<b>Factores críticos de éxito</b>	<b>Peso</b>	<b>Calificación</b>	<b>Peso ponderado</b>
<b>Oportunidades</b>	Incremento de soluciones de programabilidad de dispositivos intermediarios.	0.2	3	0.6
	Mantener los dispositivos configurados es esencial para la calidad de servicio.	0.2	4	0.8
	El proyecto puede escalar internacionalmente.	0.15	2	0.30
<b>Amenazas</b>	Productos de Cisco u otros fabricantes con funcionamiento similar.	0.15	3	0.45
	Múltiples librerías similares a la ingeniería de Cisco.	0.2	2	0.4
	La herramienta eNMs, software diseñado para crear soluciones de automatización de redes de datos, podría escalar para realizar migraciones.	0.1	1	0.1
<b>TOTAL</b>				<b>2.65</b>

Tabla 5 Matriz MEFE



### **3.1.3. Análisis de MEFI**

La matriz de evaluación de los factores internos toma en cuenta las fortalezas y debilidades para la plataforma web NetMi se procederá a asignar un peso para completar la suma total de 1 entre todos los factores, en lo que 0.01 es de menor importancia y 0.2 es de mayor importancia. Además, se clasificarán como fortaleza y debilidad entre 4 a 1 considerando 4 con mayor ponderación. El peso ponderado será la multiplicación entre el peso por la calificación, obteniendo como total la suma del peso ponderado.

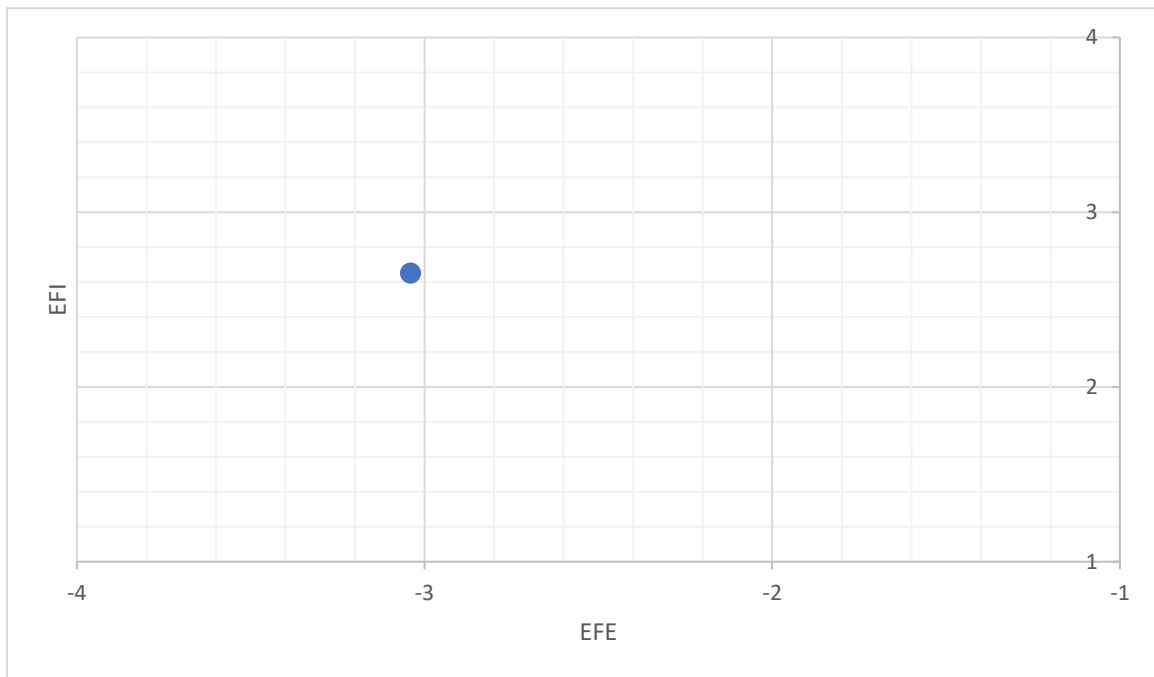
En la matriz MEFI de la tabla 6, se puede observar que existe mayor peso de fortalezas dado a que el desarrollo de la plataforma web NetMi se ha dado con la ingeniería de Cisco, cubriendo otro potente proveedor de dispositivos como Huawei, ambos de sistemas operativos actuales, además se encuentra en la Nube, lo cual evita procesamiento de información en recursos de los clientes. El total de 3.04 establece que tiene una posición de Fortaleza inicial debido que se encuentra entre el rango de 3 a 4 de evaluación de factores internos. Este resultado se debe a que al ser la primera fase de la plataforma web NetMi, abarca dos proveedores, de los múltiples que se encuentra en el mercado, por otro lado, la ingeniería de las librerías de Python utilizadas, se encuentran igualmente en primera fase, por otro lado, los proveedores de servicio de internet realizan estas migraciones por medio de sus funcionarios, evitando herramientas de automatización.

	Factores críticos de éxito	Peso	Calificación	Peso ponderado
<b>Fortalezas</b>	Implementación con software de Cisco.	0.13	3	0.39
	Migración de configuraciones entre sistemas operativos de Cisco IOS y Cisco IOS XE hacia Cisco IOS XR y Huawei AR.	0.16	4	0.64
	Aplicación de múltiples fabricantes.	0.18	4	0.72
	Implementación del Back-End y Front-End con servicios en la Nube.	0.17	4	0.68
<b>Debilidades</b>	Limitación de sistemas operativos de Cisco a Cisco o Huawei.	0.13	2	0.26
	La ingeniería de Cisco no cuenta con desarrollo avanzado.	0.12	2	0.24
	Configuración automática no es implementada por los ISP.	0.11	1	0.11
<b>TOTAL</b>				<b>3.04</b>

Tabla 6 Matriz MEFI

### 3.1.4. Análisis de MIE

La matriz interna externa se basa en los totales obtenidos en las matrices MEFI y MEFE para determinar estrategias o recomendaciones para la organización, obteniendo la coordenada (3.04, 2.65). La matriz ubica el punto en el cuarto cuadrante, por lo que considera que la plataforma web NetMi debería crecer y construir.



*Ilustración 3.1 Ubicación dentro de la matriz interna y externa de los resultado de MEFE y MEFI*

### 3.1.5. Análisis de MPC

La matriz del perfil competitivo se enfoca en determinar las fortalezas y debilidades de competidores directos. Se procederá a asignar un peso en el cual la suma de los totales será 1. La calificación de los factores se determina entre fuerza mayor como 4, fuerza menor como 3, debilidad menor como 2 y debilidad mayor como 1. No obstante, se multiplica el peso y su calificación para obtener el peso ponderado. Por último, se suma el peso ponderado para su respectivo análisis.

La matriz de perfil competitivo de la Tabla 7 muestra la comparación de la plataforma web Netmi con la herramienta de migración de los ingenieros de Cisco. Se puede observar que existe una ventaja competitiva de la plataforma web NetMi debido a que entrega servicio de migración en más de un proveedor de dispositivos y a su vez la disponibilidad de esta en la nube marca la diferencia. Por otro lado, esta

herramienta de Cisco es especialista en realizar estas migraciones de estos dispositivos, por lo que su impacto se encuentra con posición extrema fuerte.

Factores críticos para el éxito	Peso	Netmi		Cisco	
		Calificación	Peso ponderado	Calificación	Peso ponderado
Implementación con software de Cisco.	0.13	3	0.39	4	0.52
Migración de configuraciones entre sistemas operativos de Cisco IOS y Cisco IOS XE hacia Cisco IOS XR y Huawei AR.	0.16	4	0.64	3	0.48
Aplicación de múltiples fabricantes.	0.18	4	0.72	2	0.36
Implementación del Back-End y Front-End con servicios en la Nube.	0.17	4	0.68	2	0.34
Limitación de sistemas operativos de Cisco a Cisco o Huawei.	0.13	2	0.26	2	0.26
La ingeniería de Cisco no cuenta con desarrollo avanzado.	0.12	2	0.24	4	0.48
Configuración automática no es implementada por los ISP.	0.11	1	0.11	1	0.11
		<b>TOTAL</b>	<b>3.04</b>	<b>TOTAL</b>	<b>2.55</b>

Tabla 7 Matriz MPC

### **3.2 Análisis de costos**

El desarrollo de esta API se llevó a cabo con las múltiples tecnologías presentadas en este proyecto. Estas fueron usadas como base de la configuración de toda la interfaz de usuario y de los procesos que se realizan durante cada operación o petición generada por un cliente en el Back-End. La implementación y entorno de producción del proyecto se realizó en los servicios de la nube que permitieron un mejor control de fallos, configuraciones y análisis de los servicios implementados, por ejemplo, las bases de datos.

### **3.3 Etapa pre-operativa**

Se plantea una cotización del proyecto para la venta hacia un cliente u organización que requiera el servicio. La plataforma web manejará datos de los dispositivos intermediarios de los proveedores de servicio de internet que cuenten con acceso, este puede brindar un servicio a empleados u organizaciones asociadas mediante el uso de credenciales proporcionadas por el administrador de la página ya que esta se encuentra restringida por un inicio de sesión debido a que almacena información crítica de las redes de datos que se administran.

### **3.4 Costos de diseño y desarrollo**

Durante esta fase se busca optimizar el desarrollo del proyecto a implementar, esta es la razón por la que se manejan plataformas en la Nube como Firebase y Cloud Run, ambas pertenecientes a los servicios brindados por Google, pues se mantiene una mejor experiencia de trabajo. De esta forma, se espera reducir los tiempos de desarrollo y los costos que genera.

Como etapa inicial de la plataforma web NetMi se definieron los siguientes costos de implementación, programación y desarrollo. Además de considerar la etapa operativa y sus gastos, los rubros se detallan en la Tabla 8:

Ítem	Descripción	Cantidad	Valor
1	Análisis, desarrollo e implementación (programación) del servicio web dedicado a la migración de dispositivos intermediarios. (Back-End y Front-End)	1	\$800,00
2	Análisis, implementación y configuración de los servicios en la nube y sus bases de datos en tiempo real.	1	\$300,00
4	Inducción en el manejo del sistema al personal encargado del área o departamento de TI	1	*\$70,00
5	Soporte adicional del servicio implementado. (Control de cambios requeridos por los distintos fabricantes de dispositivos intermediarios o modelos más comunes del mercado.)	-	**\$30,00
6	Corrección de errores y ajustes en producción.	-	-
<b>TOTAL</b>			<b>\$1,200.00</b>

*Tabla 8 Propuesta económica planteada para venta del servicio.*

\*\* El valor de la inducción al encargado del área o departamento de TI es un pago mensual, considerando que se necesitan más de una de estas actividades. La primera inducción es gratis.

\*\*\* El soporte adicional del servicio implementado es gratuito durante los primeros 15 días, a partir de las siguientes fechas se cobrará el valor establecido en la propuesta por cada evento.

Además, se deben cotizar los valores mensuales generados por las plataformas en la nube de las que se hará uso. Estas decisiones varían dependiendo de la cantidad de dispositivos que la empresa maneje y del personal de trabajo que pertenece al departamento de TI.

<b>Propuesta Económica de servicios en la nube (Valores mensuales)</b>			
<b>Servicio en la Nube</b>	<b>Detalle</b>	<b>Versión Gratuita</b>	<b>Versión Pagada</b>
Google Firebase	Alojamiento	10 GB	\$0.026/GB
	Alojamiento – transferencia de datos	360MB/día	\$0.15/GB
	Base de datos	100	200.000 por base de datos
	Conexiones simultaneas		
	Base de datos - almacenamiento	1 GB	5 GB
	Base de datos- transferencia	10 GB	\$1/GB
Google Cloud Run	CPU	180000 vCPU/segundo	\$0.000024 vCPU/segundo
	Memoria	360000 GB/segundo	\$0.0000025 GB/segundo
	Solicitudes	2 millones solicitudes	\$0.40 por millón de solicutdes
	Redes	1 GB	\$0.085 de 0-10 TB

*Tabla 9 Propuesta Económica contratación de servicios en la nube*

Estas capacidades y recursos permitirán a la plataforma web NeMi funcionar en óptimas condiciones ya que los servicios en la nube ofrecen en primer lugar durabilidad para almacenar información de manera redundante evitando pérdidas de información por desastres naturales, errores humanos o fallos en los dispositivos,

por segundo, disponibilidad en cualquier instante de tiempo y, en tercer lugar, seguridad usando cifrado de datos y controles de acceso.

En la Tabla 9 se detallan los costos sin cargo de los servicios en la nube de Firebase y Cloud Run, los cuales pueden ser utilizados de manera gratuita con limitantes de almacenamiento en GB o por recursos computacionales virtuales por segundo o solicitudes, cuando sobrepasa esos límites. Google factura por la utilización de estos servicios.

### **3.5 Costos de actualización y mantenimiento**

Las plantillas de configuración de los dispositivos intermediarios deberán ser implementadas y/o actualizadas dependiendo de los modelos que maneja la empresa. Este control deberá tener una revisión mensual de todos los componentes, librerías y plantillas para configuración que existen y permiten el correcto funcionamiento del sistema.

### **3.6 Análisis costo/beneficio**

El producto final, presentado como el resultado de esta investigación y desarrollo tiene un valor aproximado de \$1000,00 considerando ciertos recursos como gratuitos. Esto representa la inversión realizada para el desarrollo de la plataforma web, sin embargo, al simbolizar un único pago, las próximas facturas a generar serán las presentadas en la Tabla 9, valores mensuales propuestos por los servicios alojados en la nube. Además, al poder tener una implementación con inicio de sesión autorizado, será suficiente una implementación para que las distintas sucursales puedan aprovechar el producto contratado, generando un beneficio en un largo plazo y evitando la necesidad de contratar a distintas personas por cada tarea rutinaria de migraciones en dispositivos de telecomunicación.



Los servicios de Google mantienen un recurso sin costo con un límite. Sin embargo, la plataforma web está desarrollada de tal forma que este consumo sea mínimo y pueda realizar tu trabajo esperado como las modificaciones en las configuraciones de los dispositivos intermediarios.

### **3.7 Restricciones y limitaciones de funcionalidad en la plataforma web**

El requerimiento de funcionalidades de la aplicación por parte de los proveedores de servicio de internet se enfocaba en dispositivos de Cisco, por lo que se determinó que el recurso de librerías existentes de redes de datos para Python sea PyATS con genie, las cuales han sido desarrolladas por Cisco, por lo tanto, los sistemas operativos disponibles de migración son de IOS/IOS XE de Cisco hacia IOS RX de Cisco o AR de Huawei.

Con todo esto, estas librerías de PyATS y Genie lleva desarrollándose desde el 2017. No obstante, a pesar de que la documentación de estas, por parte de Cisco, es la más completa, existen aún carencia de metodología para obtener la información de los dispositivos intermediarios de distintos modelos. Sin embargo, las aplicaciones que se han enfocado para el desarrollo de la plataforma web son la configuración de dispositivos de proveedores de servicio de internet, tal como *prefix-list*, *service-instance*, *bridge-domain* y entre otros.

La plataforma web depende de la estabilidad de los enlaces o conexión con los dispositivos intermediarios pues estos adquieren la configuración actual directamente de los mismos mediante un enlace vía SSH, funcionando como una base de datos, los tiempos de respuesta de la aplicación dependerá de la conexión, tamaño de la información almacenada, método y plantilla de estos dispositivos.

# CAPITULO 4

## 4. CONCLUSIONES Y RECOMENDACIONES

### 4.1 Conclusiones

La aplicación NetMi permitió actualizar los dispositivos intermediarios de una red empresarial verificando las configuraciones del nuevo dispositivo, como es el caso de los sistemas operativo IOS XE a IOS XR, quienes el fabricante ha optimizado los comandos de configuración. Realizar este proceso de migración de manera manual conlleva un tiempo aproximado de una semana por cada protocolo. Mientras que, automatizar este proceso con NetMi, la configuración se obtiene en un tiempo aproximado de 30 minutos, considerando que el profesional técnico de telecomunicaciones requiere verificar los datos obtenidos de la aplicación por norma de seguridad.

Las herramientas propuestas como, Python, lenguaje de programación, usado para la lógica del Back-End; y Angular, entorno de trabajo, usado para el desarrollo del Front-End, la plataforma web de NetMi permitió la implementación de un control de acceso restringido para mayor seguridad de los dispositivos intermediarios registrados en su base de datos. Para ello, se definió información que permite autenticarse con estos dispositivos y a su vez entregar configuraciones de políticas, interfaces, listas de acceso y protocolos de enrutamiento.

NetMi mostró que el análisis estratégico en su plataforma web, cuenta con una gran ventaja, dado que permite abarcar dispositivos de fabricantes como Cisco y Huawei, que son los más utilizados en el mercado, ubicando a la aplicación en el cuadrante de construcción y crecimiento de la matriz interna y externa. Por lo que, se obtuvo este resultado debido a que, en las distintas investigaciones presentadas en el problema, en el mercado existen soluciones similares de automatización, pero no logran migrar configuraciones de dispositivos a nivel empresarial.

## 4.2 Recomendaciones

En el momento en que el Back-End se comunica con una conexión remota vía SSH con los dispositivos intermediarios, los datos son enviados por una solicitud que se genera en el navegador de la plataforma web, sin embargo, no se ha desarrollado una capa de seguridad para el acceso a una red privada para el transporte de los datos entre los microservicios que solicita el Front-End, se recomienda que el proveedor de servicio de internet ingrese en su lista de acceso la dirección IP del servicio de Back-End para mayor seguridad de sus dispositivos intermediarios.

Establecer la plataforma web NetMi en una red local de un proveedor de servicio de internet será la mejor opción para implementación a comparación de un servicio en la nube por seguridad y por motivo de que múltiples dispositivos intermediarios no se les configura una salida a internet o la conexión SSH no es permitida lejos de la red en la que se encuentran. Desde este punto el aplicativo podría brindar servicio a los demás dispositivos que se encuentren asociados a la organización.

Como norma de seguridad de la empresa se debería verificar las configuraciones que entrega la plataforma web antes de enviarlas a producción, dado que existen datos que las librerías que fueron usadas aún se encuentran en etapa de desarrollo y no entregan información completa para que las plantillas tengan mayor fiabilidad de información.

### **4.3 Trabajo a futuro**

Actualmente, la ingeniería de Cisco se encuentra en constante desarrollo de librerías en Python, así como otras organizaciones para la automatización en redes de datos. A medida que se van desarrollando más funcionalidades, se puede expandir la plataforma web NetMi para migrar configuraciones de distintos sistemas operativos. Un trabajo a corto plazo podría ser enfocado en migrar configuraciones de Cisco hacia otro tipo de proveedor tal como Huawei, HP, Juniper o Arista, incluso se puede migrar configuraciones de Juniper hacia los otros mencionados debido a que se encuentra implementado funciones básicas para obtener información de dispositivos de Juniper.

Un trabajo adicional que se puede aplicar a la plataforma web NetMi, es la programación de actividades tal como configuraciones o migraciones en horarios con menor tráfico de las redes, ya que esto evitaría que el ingeniero realice este proceso manualmente debido a que son realizadas en horarios nocturnos y pueden existir errores humanos en el momento de realizar esta actividad.

# BIBLIOGRAFÍA

- [1] Jaramillo, A. C., Villavicencio, M., Moreira, V., & Lara, P. (2019). Ingeniería de Software Aplicada a Telemática: Rediseñando el Curso de Conmutación y Enrutamiento. In ClbSE. Escuela Superior Politécnica del Litoral, Facultad de ingeniería en Electricidad y Computación (pp. 197-210).
- [2] Suzanne M. Mannes, Walter Kintsch,. (1991). Routine Computing Tasks: Planning as Understanding. Cognitive Science, Volume 15, Issue 3, Pages 305-342. [https://doi.org/10.1016/0364-0213\(91\)80001-L](https://doi.org/10.1016/0364-0213(91)80001-L).
- [3] Fernández, R. (2020, 31 may). Statista. Cuota de mercado de los principales proveedores de servicios de red (network services) para empresas a nivel mundial en 2019. [En línea]. Accedido el 2 de julio desde: <https://es.statista.com/estadisticas/580535/cuota-de-mercado-mundial-de-los-proveedores-de-redes-informaticas-empresariales/>
- [4] VELTE, Toby J.; VELTE, Anthony T. Cisco: a beginner's guide. McGraw-Hill/Osborne, 2007.
- [5] M. Caria, F. Carpio, A. Jukan and M. Hoffmann, "Migration to energy efficient routers: Where to start?," 2014 IEEE International Conference on Communications (ICC), 2014, pp. 4300-4306, doi: 10.1109/ICC.2014.6883996.
- [6] T. Das, M. Caria, A. Jukan and M. Hoffmann, "Insights on SDN migration trajectory," 2015 IEEE International Conference on Communications (ICC), 2015, pp. 5348-5353, doi: 10.1109/ICC.2015.7249174.
- [7] B. Edgeworth, A. Foss and R. Rios, IP routing on Cisco IOS, IOS XE, and IOS XR. Indianapolis, Ind.: Cisco Press, 2015. ISBN-10:1-58714-423-9 ISBN-13:978-1-58714-423-3
- [8] M. H. Perez and A. M. Santiago, "Process of Migration of Routing Protocols in a LAN: RIP to OSPF," 2006 3rd International Conference on Electrical and Electronics Engineering, 2006, pp. 1-5, doi: 10.1109/ICEEE.2006.251919.
- [9] W. Zhou, L. Li, M. Luo y W. Chou, "REST API Design Patterns for SDN Northbound API", 2014 28th International Conference on Advanced Information

Networking and Applications Workshops, 2014, págs. 358-365, doi: 10.1109/WAINA.2014.153.

[10]" Nelson, T., Ferguson, A.D., Yu, D., Fonseca, R., & Krishnamurthi, S. "Exodus: toward automatic migration of enterprise network configurations to SDNs., 2015. In Brown University. doi: 10.1145/2774993.2774997

[11] H. Pouya, B. Jaumard and C. Preston-Thomas, "Minimum network migration cost and duration," 2017 IEEE 38th Sarnoff Symposium, 2017, pp. 1-6, doi: 10.1109/SARNOF.2017.8080395.

[12] Yi Wang, Eric Keller, Brian Biskeborn, Jacobus van der Merwe, and Jennifer Rexford. 2008. Virtual routers on the move: live router migration as a network-management primitive. SIGCOMM Comput. Commun. Rev. 38, 4 (October 2008), 231–242. DOI: 10.1145/1402946.1402985

[13] Cosoi, E. (2000). Elección de empresas de acceso a Internet (ISP: Internet Service Provider). Revista chilena de pediatría, 71(5), 451-452. doi:10.4067/S0370-41062000000500015

[14] Vera, A., Badia, A., & Barberá, M. P. (2003). La adopción de internet en la red de empresas y la percepción de la nueva economía en comarcas semi-remotas de tradición industrial de Catalunya. Boletín de la Asociación de Geógrafos Españoles.

[15]"Cisco", What is Network Automation? 2021. [En línea]. Accedido el 25 de mayo, 2021, desde: <https://www.cisco.com/c/en/us/solutions/automation/network-automation.html>.

[16] R. Ansible, "Network Automation with Ansible", Ansible.com, 2021. [En línea]. Accedido el 2 de junio, 2021, desde: <https://www.ansible.com/integrations/networks>.

[17]"Cisco DevNet", Cisco DevNet, 2021. [En línea]. Accedido el 25 de mayo, 2021, desde: <https://developer.cisco.com/docs/pyats/#!introduction/cisco-pyats-network-test--automation-solution>.

[18] Sánchez Vega, Marcelo Dario (2018). Diseño e implementación de una aplicación web para la venta de artesanías en el programa Ciudades Creativas de la UNESCO. Trabajo final para la obtención del título: Ingeniero en Computación. Espol.Fiec, Guayaquil. 55p.

- [19] Docker, "Developing with docker", Docker.com, 2021. [En línea]. Accedido el 8 de junio, 2021, desde: <https://www.docker.com/why-docker>.
- [20] E. Qin, Y. Wang, L. Yuan and Y. Zhong, "Research on Nginx Dynamic Load Balancing Algorithm," 2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), 2020, pp. 620-624, doi: 10.1109/ICMTMA50254.2020.00138.
- [21] Moroney, L., Moroney, & Anglin. (2017). Definitive Guide to Firebase (pp. 51-71). Apress. doi: 10.1007/978-1-4842-2943-9. ISBN: 978-1-4842-2943-9
- [22] GNS3, "Docker support in GNS3" docs.gns3.com , 2021. [En línea]. Accedido el 8 de junio, 2021, desde: <https://docs.gns3.com/docs/emulators/docker-support-in-gns3>
- [23] M. Caria, A. Jukan and M. Hoffmann, "A performance study of network migration to SDN-enabled Traffic Engineering," 2013 IEEE Global Communications Conference (GLOBECOM), 2013, pp. 1391-1396, doi: 10.1109/GLOCOM.2013.6831268.
- [24] Napalm, "Welcome to NAPALM's documentation!", napalm.readthedocs.io, 2021. [En línea]. Accedido el 8 de junio, 2021, desde: <https://napalm.readthedocs.io/en/latest/>.
- [25] Matturro, G. (2007). "Introducción a la configuración de routers cisco". Universidad ORT Uruguay, Facultad de ingeniería. Cisco Certified Network Associate. pp. 172-179,
- [26] A. Collaguazo, M. Villavicencio and A. Abran, "Education Model for Developing IoT and Cloud Mobile Applications," 2020 IEEE World Congress on Services (SERVICES), 2020, pp. 251-258, doi: 10.1109/SERVICES48979.2020.00057.

## **ANEXO A**



# Nota Técnica

## Diseña de una aplicación web para la automatización de configuraciones de administración de redes de datos en un ISP

### Introducción

El objetivo de esta aplicación es migrar configuraciones entre dispositivos intermediarios de un proveedor de servicio de internet con sistemas operativos Cisco IOS/IOS XE a Cisco IOS XR o Huawei AR debido a que los protocolos de enrutamiento de los sistemas operativos más actuales han sido optimizados por los fabricantes.

Se construye una aplicación web basada en microservicios, el Front-End está basado en el entorno de trabajo de Angular con alojamiento y base de datos en el servicio en la nube de Firebase, por otro lado, el Back-End está desarrollado en Python, el cual puede alojarse localmente con Docker o con el servicio en la nube de Cloud Run.

### Materiales

- Sistema operativo de Linux / Sistema operativo virtual de Linux
- Visual Studio Code
- GNS3
- Docker
- Repositorio de proyecto: <https://github.com/ESPOL-NETMI>

## Procedimiento

### Paso 1: Creación de programa base de Back-End

- a) Instalación de WSL2 para sistemas operativo de Windows 10, omitir este ítem en caso de tener sistema operativo Linux o MacOS para el desarrollo del proyecto.

cmd	dism.exe /online/enable-feature/featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
	dism.exe /online/enable-feature/featurename:VirtualMachinePlatform/all /norestart
	<i>wsl –set-default-version 2.</i>

Tabla 1 comandos de instalación de wsl

Instalar versión de Linux de la tienda de Microsoft, se recomienda instalar Ubuntu 18.04 LTS.



Ilustración 1 instalación de versión de WSL

- b) Habilitar WSL2 en Visual Studio Code para Windows 10 instalando la extensión Remote – WSL

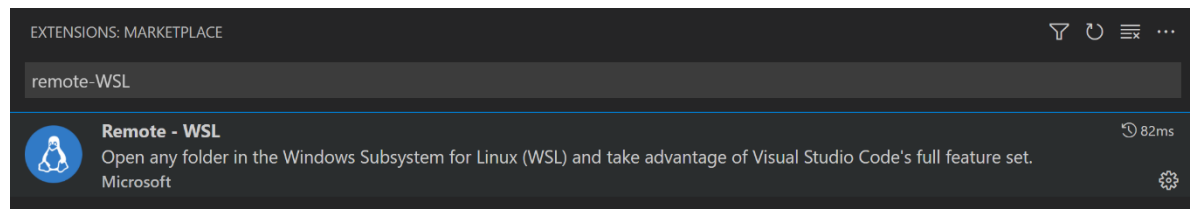



Ilustración 2 Instalación de WSL remoto para VSC

- c) Crear la carpeta base del proyecto de manera local, luego abrir la carpeta en VS, por siguiente dar clic en el ícono  de abrir una ventana remota y por último reabrir la carpeta en modo WSL.

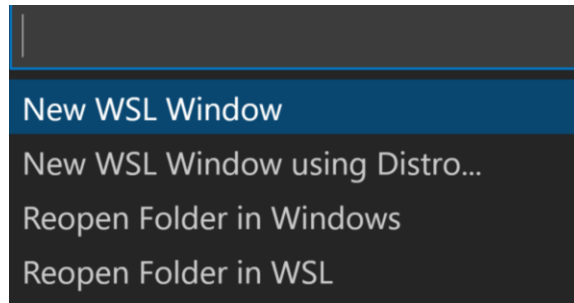


Ilustración 3 Acceso a ambiente de WSL en VSC

- d) En la carpeta crear los siguientes archivos y carpeta:
- El archivo `main.py` donde se declararán los microservicios.
  - El archivo `gpapi.py` donde se declara funciones para conexión con los dispositivos.
  - El archivo `genipyats.txt` donde se declararán las librerías de cisco con sus versiones.
  - El archivo `microservice.txt` donde se declararán las librerías de microservicios y de plantillas con sus versiones.
  - La carpeta `plantillas` contiene los archivos base para cada método de migración.
- e) Declarar en `genipyats.txt` las siguientes librerías.
- `pyats[full] == 21.8`
  - `genie == 21.8`
- f) Declarar en `microservice.txt` las siguientes librerías.
- `pyyaml == 5.4.1`
  - `Jinja2==3.0.1`
  - `requests==2.26`
  - `flask == 2.0.1`
  - `flask-cors == 3.0.10`

g) Crear un entorno virtual de Python e instalar las librerías.

Bash	apt-get install python3-venv
	pip install --upgrade pip
	python3 -m venv envnetmi
	source envnetmi/bin/actívale
	python3 -m pip install --upgrade setuptools
	python3 -m pip install -r geniepyats.txt
	python3 -m pip install -r microservice.txt

Tabla 2 Instalación de ambiente de trabajo y librerías

## Paso 2: Desarrollo de Back-End

Se realiza la creación de funciones en gpapi.py tal para recibir los parámetros de autenticación con dispositivos intermediarios y construcción de un diccionario para la conexión remota, así mismo, obtener las configuraciones y cerrar la conexión con el dispositivo.

Los parámetros para autenticación con los dispositivos intermediarios son los siguientes:

- Nombre del equipo.
- Dirección IP.
- nombre de usuario.
- Contraseña.
- sistema operativo.
- contraseña de privilegios.
- puerto (si es requerido).
- Encriptación.

```

gapi.py
1  from genie import testbed
2
3  class gpapi(object):
4      def __init__(self, json_data=None):
5
6          name = json_data["name"]
7          ip = json_data["ip"]
8          protocol = json_data["protocol"]
9          password = json_data["password"]
10         username = json_data["username"]
11         os = json_data["os"]
12         enable = json_data["enable"]
13         port = json_data["port"]
14         encriptacion = json_data["encryp"]
15         if(protocol=="ssh"):
16             protocol=protocol+" -c "+encriptacion
17         testbeddict={"devices":{
18             name:{
19                 "connections":{
20                     "cli":{
21                         "ip":ip,
22                         "protocol": protocol,
23                         "port":port
24                     }
25                 },
26                 "credentials":{
27                     "default":{
28                         "username": username,
29                         "password": password
30                     }, "enable":{
31                         "password": enable
32                     }
33                 },
34                 "os": os
35             }
36         }}
37         test = testbed.load(testbeddict)
38         self.device = test.devices[name]
39
40     def connect(self):
41         self.device.connect(init_exec_commands=[], init_config_commands=[], log_stdout=False)
42
43     def disconnect(self):
44         self.device.disconnect()
45
46     def showconfig(self, show=None):
47         self.connect()
48         parse = self.device.parse(show)
49         return parse

```

Ilustración 4 Funciones para acceso a dispositivos intermediarios

El formato del diccionario de autenticación es representado como en el siguiente ejemplo:

```
! testbed.yaml
1  devices:
2    GYE_PRIMAX:
3      connections:
4        cli:
5          ip: 172.25.192.101
6          port: 443
7          protocol: ssh -c aes128-cbc
8      credentials:
9        default:
10         password: cisco
11         username: admin
12         enable:
13           password: cisco123
14     os: iosxe
```

Ilustración 5 Ejemplo de formato de diccionario creado en código

La creación de microservicios en main.py conlleva un conjunto de funciones con método POST para recibir datos de autenticación y entregar la configuración de los dispositivos, para declarar los servicios se requiere el uso de Flask, Json, Request, Cors, Environment, YAML y entre otras librerías que se detalla en la ilustración 6, para mayor detalle de las funciones de microservicios acceder al repositorio del proyecto.

```
main.py
1  from flask import Flask,json,request
2  import logging
3  from flask_cors import CORS, cross_origin
4  from jinja2 import Environment, FileSystemLoader
5  import yaml
6  from gpapi import gpapi
7  app = Flask(__name__)
8  cors = CORS(app)
9  app.config['CORS_HEADERS'] = 'Content-Type'
10
11  @app.route('/show',methods=['POST'])
12  @cross_origin()
13  > def show(): ...
```

Ilustración 6 Declaración de librerías y ejemplo de microservicio "show" para pruebas

```

60 @app.route('/getparsingcfg',methods=['POST'])
61 @cross_origin()
62 def getparsingcfg():
63     try:
64         json_data = request.get_json()
65         show = json_data["show"]
66         plantilla = json_data["plantilla"]
67         connection = gpapi(json_data)
68         parse = connection.showconfig(show)
69         data = {"data":parse}
70         connection.disconnect()
71     try:
72         show2 = json_data["show2"]
73         parse2 = connection.showconfig(show2)
74         data = {"data":parse,"data2":parse2}
75     except:
76         data = {"data":parse}
77         connection.disconnect()
78         env = Environment(loader = FileSystemLoader('.'), trim_blocks=True, lstrip_blocks=True)
79         template = env.get_template(plantilla)
80         doc = template.render(data)
81         s=200
82     except:
83         doc="No data"
84         s=400
85     response = app.response_class(response=doc,
86                                   status=s,
87                                   mimetype='text/cfg')

```

Ilustración 7 Ejemplo de función para obtener plantilla de configuración

Para poder agregar más funciones de microservicio, la documentación de los diccionarios con la información de los dispositivos intermediarios se puede acceder a <https://pubhub.devnetcloud.com/media/genie-feature-browser/docs/#/parsers>.

### Paso 3: Creación de programa base de Front-End

La base del proyecto pertenece al panel de trabajo Argon de Creative Tim, para más información visitar <https://www.creative-tim.com/product/argon-dashboard-angular>. Se procede a construir el Front-End con interfaces para obtener información de dispositivos y obtener las plantillas de configuración, así mismo, el repositorio del proyecto se puede utilizar como base del Front-End.

### Paso 4: Desarrollo de Front-End

En esta sección se explicará información necesaria que involucra las funcionalidades del proyecto.

- a) Se establece una estructura que pertenece a información esencial de los dispositivos.

```
1  export class Device {  
2    $key: string;  
3    name: string;  
4    os: string;  
5    ip: string;  
6    model: string;  
7    protocol: string;  
8    encryp:string;  
9  }
```

Ilustración 8 Estructura de base de datos

- b) Se utiliza la librería file-saver para poder descargar los archivos de configuración.

```
npm install file-saver --save
```

- c) Se utiliza la librería AngularFireAuth como método de autenticación para acceder a la plataforma web.
- d) Se utiliza las librerías AngularFireDatabase y AngularFireList para las funciones de base de datos en la nube, tal como, obtener, insertar, actualizar y eliminar datos.

```
npm install --save firebase
```

- e) Los archivos que definen la estructura se detallan a continuación:



El archivo `app.module.ts` define el archivo principal de la plataforma web indicando estructura, enrutamiento y conexión con los servicios.

```
18 @NgModule({
19   imports: [
20     BrowserModule,
21     FormsModule,
22     HttpClientModule,
23     ComponentsModule,
24     NgbModule,
25     RouterModule,
26     AppRoutingModule,
27     AngularFireModule.initializeApp(environment.firebase),
28     AngularFireDatabaseModule,
29   ],
30   declarations: [
31     AppComponent,
32     AdminLayoutComponent,
33     AuthLayoutComponent
34   ],
35   providers: [AngularFireAuth, AuthGuard, AuthService],
36   bootstrap: [AppComponent]
37 })
38 export class AppModule { }
```

Ilustración 9 archivo `app.module.ts`

El archivo `environment.ts` define la estructura con la información de autenticación con el servicio en la nube de Firebase.

```
5 export const environment = {
6   production: false,
7   firebase :{
8     apiKey: "xxxxxxxxxxxxx",
9     authDomain: "netmi-frontend.firebaseio.com",
10    projectId: "netmi-frontend",
11    storageBucket: "netmi-frontend.appspot.com",
12    messagingSenderId: "xxxxxxxxx",
13    appId: "xxxxxxxxxxxxx",
14    measurementId: "G-xxxxxxxxx"
15  }
16 };
```

Ilustración 10 archivo `environment.ts`

El archivo `app.routing.ts` define el enrutamiento general de la plataforma web, de modo que divide la aplicación en `AdminLayoutComponent` y `AuthLayoutComponent`.

```
8  const routes: Routes = [
9    {
10     path: '',
11     redirectTo: 'dashboard',
12     pathMatch: 'full',
13   }, {
14     path: '',
15     component: AdminLayoutComponent,
16     children: [
17       {
18         path: '',
19         loadChildren: './layouts/admin-layout/admin-layout.module#AdminLayoutModule'
20       }
21     ],
22   }, {
23     path: '',
24     component: AuthLayoutComponent,
25     children: [
26       {
27         path: '',
28         loadChildren: './layouts/auth-layout/auth-layout.module#AuthLayoutModule'
29       }
30     ]
31   }, {
32     path: '**',
33     redirectTo: 'dashboard'
34   }
35 ];
```

Ilustración 11 `app.routing.ts`

El archivo `auth-layout.routing.ts` define el enrutamiento de la plataforma web cuando se requiere autenticación de usuario, por lo cual, solo se puede acceder a la ruta de 'login'.

```
5  export const AuthLayoutRoutes: Routes = [
6    { path: 'login', component: LoginComponent }
7  ];
```

Ilustración 12 `auth-layout.routing.ts`

El archivo `admin-layout.routing.ts` define el enrutamiento de la plataforma web cuando existe autenticación de usuario.

```
7  export const AdminLayoutRoutes: Routes = [
8    { path: 'dashboard', component: DashboardComponent, canActivate : [AuthGuard] },
9    { path: 'configuration', component: DevicesComponent, canActivate : [AuthGuard] },
10   { path: 'tables', component: TablesComponent, canActivate : [AuthGuard] },
11 ];
```

Ilustración 13 `admin-layout.routing.ts`

El archivo global.ts define la ruta que conecta el Front-End con los microservicios.

```
1 export const ruta = "http://xxxxxxxxxx:5000/";
```

Ilustración 14 global.ts

## Paso 5: Despliegue de aplicación

### a) Despliegue local y simulación de GNS3

Despliegue de Back-End

ng serve

Despliegue de Front-End

python3 main.py

### b) Despliegue en la nube

Despliegue de Back-End en Cloud Run

o	Abrir SDK de Gcloud, clonar repositorio
< / >	cd src
	gcloud app deploy
¿?	Desplegar: yes
< / >	gcloud app browse

Tabla 3 Pasos para despliegue en Cloud Run

Despliegue de Front-End en Firebase

< / >	firebase login
¿?	Compartir datos: y/n [opcional]
< / >	ng build
	Firebase init
Opción	Hosting
¿?	Elegir directorio público: yes
	Sobreescribir index: no
< / >	Firebase deploy

c) Despliegue en contenedores.

Se recomienda crear una carpeta destinada a la virtualización de contenedores, en ella se definirá un archivo docker-compose.yml y dos carpetas para división del Back-End y Front-End con archivos Dockerfile.

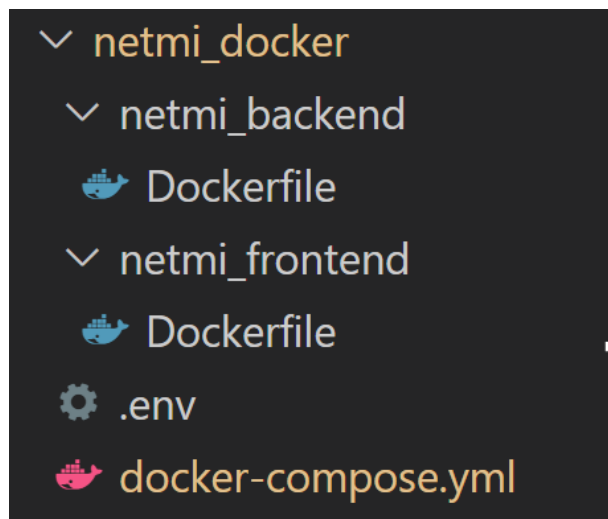


Ilustración 15 Estructura de carpeta para docker

En docker-compose.yml se definirá como servicio al Back-End y Front-End definiendo una red, nombre de contenedor, puertos de acceso, ubicación de Dockerfile y entre otras configuraciones. En cada Dockerfile se indicará el tipo de contenedor con sus respectivas instrucciones y ubicación de archivos para crear la virtualización. Un ejemplo de estas configuraciones personalizadas de contenedores se encuentra en la ruta netmi\_docker del repositorio del proyecto. Para crear los contenedores, ubicarse en la carpeta principal donde se encuentran estos y proceder con el comando:

```
docker-compose up --build -d
```

## Almacenamiento de código fuente

El código desarrollado de todo el servicio web será colocado en una organización privada de Github creada por nosotros. En él se crearán los repositorios destinados a cada parte del proyecto realizado, además se agregará al cliente y los respectivos tutores para su constancia.

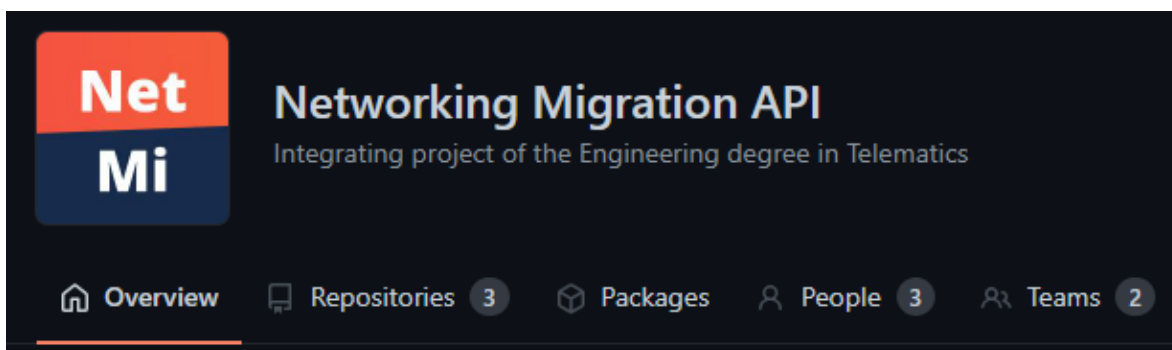


Ilustración 16 Organización en Github de la plataforma web NetMi

En esta organización se han creado tres repositorios donde se manejarán individualmente los cambios e interacciones del Back-End y Front-End, el ultimo repositorio contendrá los archivos Docker que realizaran la respectiva instalación de todos los contenedores necesarios para su ejecución.



Ilustración 17 Estructura del proyecto con Back-End, Front-End y Docker

Dentro de este último repositorio se encuentra un instructivo de instalación desarrollado en un archivo README.MD que se puede visualizar de la siguiente forma.

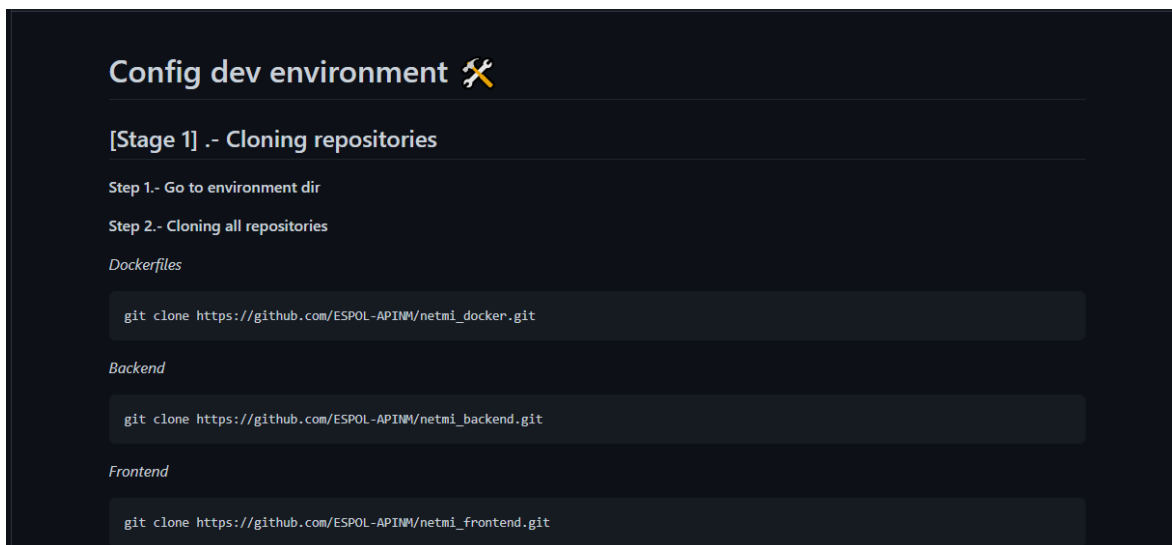


Ilustración 18 archivo readme.md para clonar el proyecto

## **ANEXO B**

# Manual de Usuario



**Net**  
**Mi**

María Carolina Aguilar Alvarado, Alex  
Alberto Córdova Balón



# ¿Qué es NetMi?

NetMi es una aplicación web diseñada para automatizar la migración de configuraciones de dispositivos intermediarios basada en sistemas operativos de los fabricantes Cisco y Huawei.

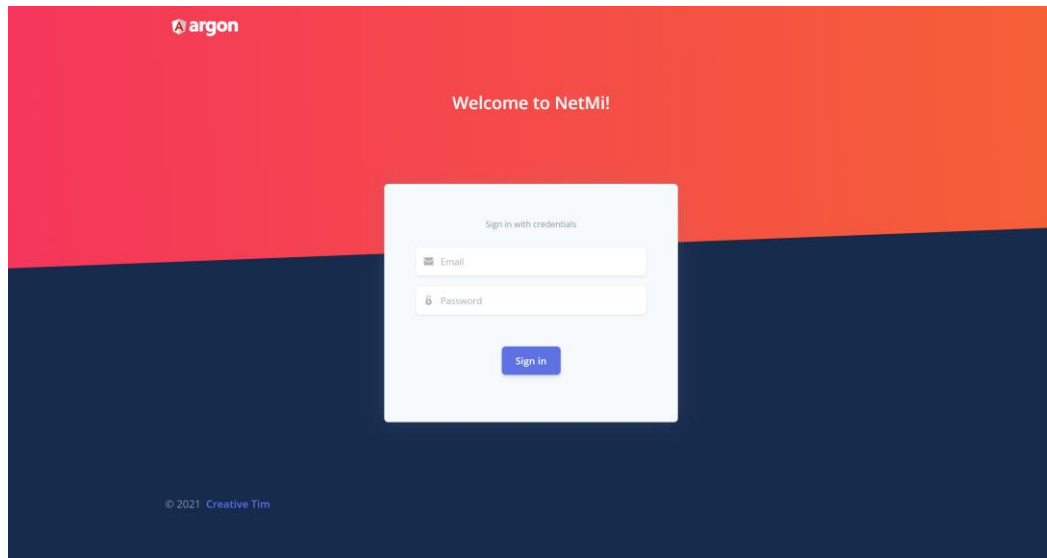
Abarca los siguientes métodos habilitados para migración de configuraciones:

- Route-Map entre Cisco IOS a Cisco IOS XR
- Route-Map entre Cisco IOS XE a Cisco IOS XR
- Route-Map entre Cisco IOS a HUAWEI AR
- Route-Map entre Cisco IOS XE a HUAWEI AR
- VRF entre Cisco IOS a Cisco IOS XR
- VRF entre Cisco IOS XE a Cisco IOS XR
- VRF entre Cisco IOS a HUAWEI AR
- VRF entre Cisco IOS XE a HUAWEI AR
- ACL entre Cisco IOS a Cisco IOS XR
- ACL entre Cisco IOS XE a Cisco IOS XR
- ACL entre Cisco IOS a HUAWEI AR
- ACL entre Cisco IOS XE a HUAWEI AR
- Prefix-List entre Cisco IOS a Cisco IOS XR
- Prefix-List entre Cisco IOS XE a Cisco IOS XR
- Prefix-List entre Cisco IOS a HUAWEI AR
- Prefix-List entre Cisco IOS XE a HUAWEI AR
- Interfaces entre Cisco IOS a Cisco IOS XR
- Interfaces entre Cisco IOS XE a Cisco IOS XR
- Interfaces entre Cisco IOS a HUAWEI AR
- Interfaces entre Cisco IOS XE a HUAWEI AR
- Bridge-Domain entre Cisco IOS a Cisco IOS XR
- Bridge-Domain entre Cisco IOS XE a Cisco IOS XR
- Static-Routing entre Cisco IOS a Cisco IOS XR

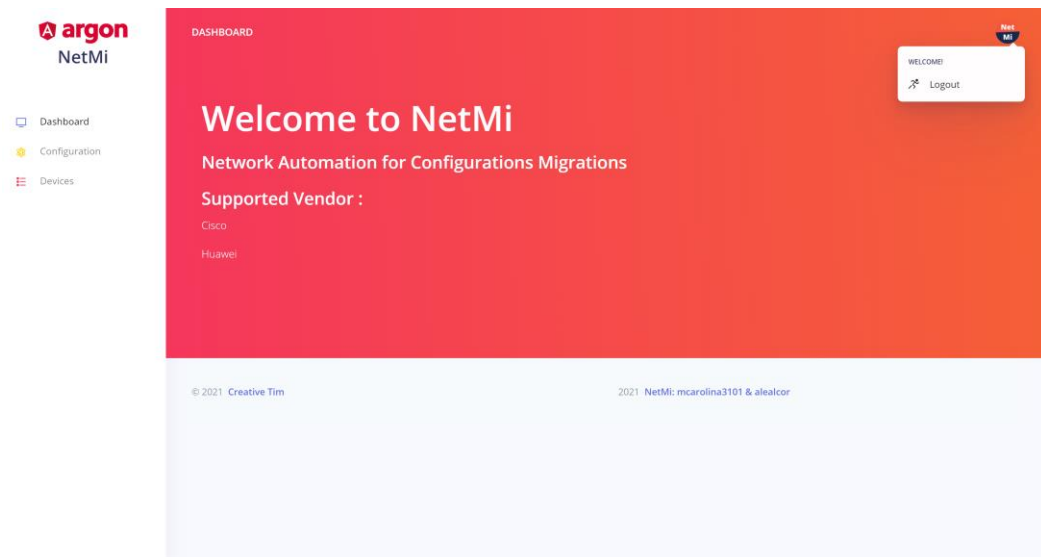
- Static-Routing entre Cisco IOS XE a Cisco IOS XR

## Manual de Interfaz

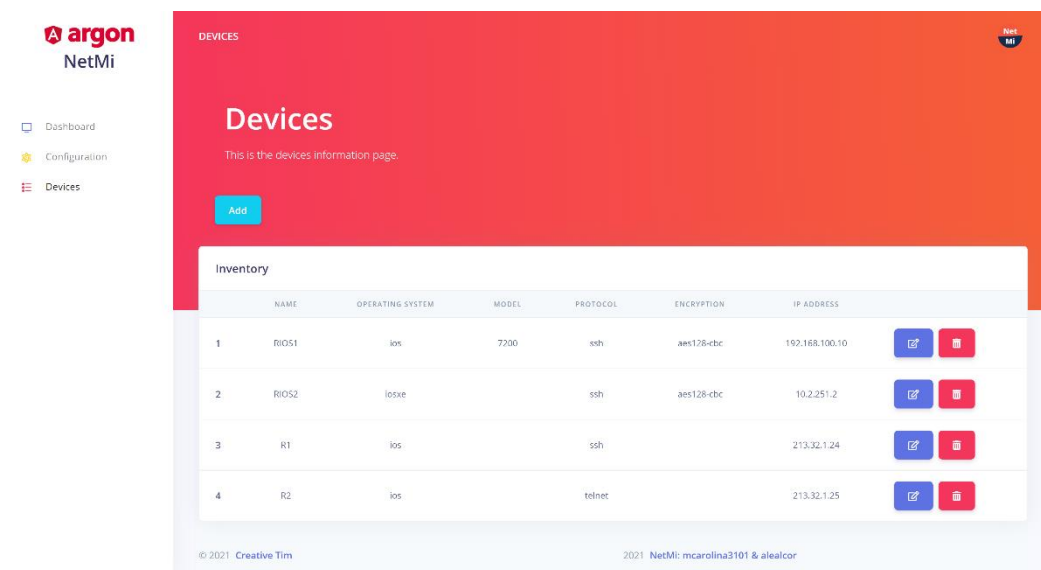
El acceso de la plataforma web es restringido para los usuarios registrados por el administrador, los datos requeridos es el correo electrónico y contraseña.



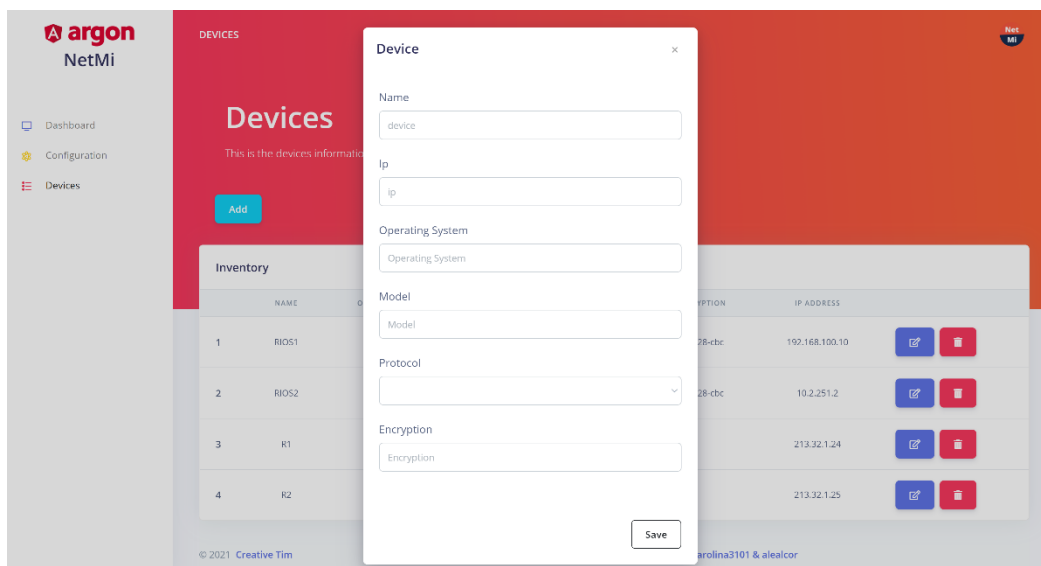
Al ingresar con credenciales, se observa la página de bienvenida y en el lateral izquierdo se accede a la lista de dispositivos registrados en la base de datos en la opción **Devices** y para obtener las migraciones se accede a **Configuration**, por último, se cuenta con un cierre de sesión en la esquina derecha.



En la página de **Devices** se accede al listado de dispositivos registrados en la base de datos, en **Add** se puede ingresar nuevos dispositivos y en el lateral derecho de ítem se puede actualizar su información y eliminar el mismo.



La información que se ingresan y actualizan en la base de datos de dispositivos son los siguientes: se tiene en modo obligatorio a los datos nombre, dirección IP, sistema operativo, protocolo y opcional a modelo y encriptación.



Se cuenta con dos tipos de protocolos de comunicación: SSH y telnet

The screenshot shows the Argon NetMi web interface. A modal window titled 'Device' is open, allowing configuration of a new device. The modal contains the following fields:

- Name:
- Ip:
- Operating System:
- Model:
- Protocol: A dropdown menu with 'ssh' and 'telnet' options. 'telnet' is currently selected.
- Encryption:

A 'Save' button is located at the bottom right of the modal. In the background, the 'Devices' page is visible, showing a table with columns for NAME, IP ADDRESS, and ACTION. The table contains four entries:

NAME	IP ADDRESS	ACTION
1 RIOS1	192.168.100.10	<input type="button" value="edit"/> <input type="button" value="delete"/>
2 RIOS2	10.2.251.2	<input type="button" value="edit"/> <input type="button" value="delete"/>
3 R1	213.32.1.24	<input type="button" value="edit"/> <input type="button" value="delete"/>
4 R2	213.32.1.25	<input type="button" value="edit"/> <input type="button" value="delete"/>

La información que se ingresa para obtener la configuración son los siguientes: método, sistema operativo de migración, dirección IP, nombre de usuario, contraseña y como opcional se tiene contraseña de privilegios y puerto.

The screenshot shows the Argon NetMi web interface with the 'Configuration' modal open. The modal is titled 'Configuration' and contains the following fields:

- Method:
- Method-IOS:
- Device Ip:
- Username:
- Password:
- Enable password:
- Port:

A 'Get Configuration' button is located at the bottom left of the modal. The background shows the 'Configuration' page with a sidebar menu containing 'Dashboard', 'Configuration', and 'Devices'.

# Manual para administración de despliegue

## 1. NetMi Front-End

### Google Firebase

Pasos para desplegar la aplicación en una cuenta de firebase

<b>&lt;/&gt;</b>	firebase login
<b>¿?</b>	Compartir datos: y/n [opcional]
<b>&lt;/&gt;</b>	ng build
	Firebase init
<b>Opción</b>	Hosting
<b>¿?</b>	Elegir directorio público: yes Sobreescribir index: no
<b>&lt;/&gt;</b>	Firebase deploy

Pasos para futuros cambios

<b>&lt;/&gt;</b>	firebase login
	ng build
	Firebase deploy

## 2. Netmi Back-End

### Google App Engine

<b>O</b>	Abrir SDK de Gcloud, clonar repositorio
<b>&lt;/&gt;</b>	cd src
	gcloud app deploy
<b>¿?</b>	Desplegar: yes
<b>&lt;/&gt;</b>	gcloud app browse

# Manual para administración de servicios

## servicio: getparsingyaml

Entrega la transformación de la información en diccionario con archivo YAML.

### Solicitud

Método	Nombre de método	
POST	http://127.0.0.1:5000/getparsingyaml	
Tipo	Parámetro	Valor
POST	Name	string
POST	Username	string
POST	Password	string
POST	Ip	string
POST	Protocol	string
POST	Encrypt	string
POST	Os	string
POST	Port	string
POST	Enable	string
POST	Show	string
POST	Show2	string
POST	plantilla	string

Ejemplo de request:

```
{
  "name": "R10S1",
  "username": "cisco",
  "password": "cisco",
  "ip": "192.168.56.10",
  "protocol": "ssh",
  "encryp": "aes128-cbc",
  "os": "ios",
  "port": 0,
  "enable": "cisco",
  "show": "show interfaces",
  "show2": "show ethernet service instance detail",
  "plantilla": "./plantillas/interfacesyam1.j2"
}
```

## 1. servicio: getparsingcfg

Entrega la transformación de la información en diccionario con archivo CFG

### Solicitud

Método	Nombre de método	
POST	http://127.0.0.1:5000/getparsingcfg	
Tipo	Parámetro	Valor
POST	name	string
POST	Username	string
POST	Password	string
POST	Ip	string
POST	Protocol	string
POST	Encrypt	string
POST	Os	string
POST	Port	string
POST	Enable	string
POST	Show	string
POST	Show2	string
POST	plantilla	string

Ejemplo de request:

```
{
  "name": "RIOS1",
  "username": "cisco",
  "password": "cisco",
  "ip": "192.168.56.10",
  "protocol": "telnet",
  "encryp": "",
  "os": "ios",
  "port": 0,
  "enable": "cisco",
  "show": "show interfaces",
  "show2": "show ethernet service instance detail",
  "plantilla": "./plantillas/interfacesyaml.j2"
}
```

## Información de solicitud

- La referencia getparsingcfg entrega un archivo de configuración para el sistema operativo de Cisco IOS-XR o para Huawei.
- La referencia getparsingyaml entrega un archivo que representa el diccionario de configuraciones del sistema operativo de Cisco IOS o de Cisco IOS-XE.

### 1. Route-Map

Tipo	Path	Parámetro	Valor
POST	getparsingcfg	Show	show route-map all
POST	getparsingcfg	Show2	Null   ""
POST	getparsingcfg	plantilla	./plantillas/routemap.j2 ./plantillas/routemaphuawei.j2
Tipo	Path	Parámetro	Valor
POST	getparsingyaml	Show	show route-map all
POST	getparsingyaml	Show2	Null   ""
POST	getparsingyaml	plantilla	./plantillas/routemapyaml.j2



## 2. Prefix-List

Tipo	Ruta	Parámetro	Valor
POST	getparsingcfg	Show	show ip prefix-list detail
POST	getparsingcfg	Show2	Null   ""
POST	getparsingcfg	plantilla	./plantillas/prefixlist.j2 ./plantillas/prefixlisthuawei.j2
Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show ip prefix-list detail
POST	getparsingyaml	Show2	Null   ""
POST	getparsingyaml	plantilla	./plantillas/prefixlistyaml.j2

## 3. Interfaces

Tipo	Ruta	Parámetro	Valor
POST	getparsingcfg	Show	show interfaces
POST	getparsingcfg	Show2	show ethernet service instance detail
POST	getparsingcfg	plantilla	./plantillas/interfaces.j2 ./plantillas/interfaceshuawei.j2
Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show interfaces
POST	getparsingyaml	Show2	show ethernet service instance detail
POST	getparsingyaml	plantilla	./plantillas/interfacesyaml.j2

## 4. Access list

Tipo	Ruta	Parámetro	Valor
POST	getparsingcfg	Show	show access-list
POST	getparsingcfg	Show2	Null   ""
POST	getparsingcfg	plantilla	./plantillas/accesslist.j2 ./plantillas/accesslisthuawei.j2
Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show access-list
POST	getparsingyaml	Show2	Null   ""
POST	getparsingyaml	plantilla	./plantillas/accesslistyaml.j2

## 5. VRF

Tipo	Ruta	Parámetro	Valor
POST	getparsingcfg	Show	show vrf detail
POST	getparsingcfg	Show2	Null   ""
POST	getparsingcfg	plantilla	./plantillas/vrfdetail.j2 ./plantillas/vrfdetailhuawei.j2
Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show vrf detail
POST	getparsingyaml	Show2	Null   ""
POST	getparsingyaml	plantilla	./plantillas/vrfdetailyaml.j2

## 6. MP BGP VRF

Tipo	Ruta	Parámetro	Valor
POST	getparsingcfg	Show	show vrf detail
POST	getparsingcfg	Show2	Null   ""
POST	getparsingcfg	plantilla	./plantillas/mpbgpvrf.j2 ./plantillas/mpbgpvrfhuawei.j2
Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show vrf detail
POST	getparsingyaml	Show2	Null   ""
POST	getparsingyaml	plantilla	./plantillas/mpbgpvrfyaml.j2

## 7. Static-Route

Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show interfaces
POST	getvrfstaticroute	Show2	Null   ""
POST	getvrfstaticroute	plantilla	./plantillas/staticroutevrfyaml.j2

## 8. Bridge-domain

Tipo	Ruta	Parámetro	Valor
POST	getparsingyaml	Show	show ethernet service instance detail
POST	getparsingyaml	Show2	Null   ""
POST	getparsingyaml	plantilla	./plantillas/bridgedomainyaml.j2