# Infrastructure as Code (IAC)

## IP Address Management System (IPAM)

I am using Nautobot as my IPAM solution.

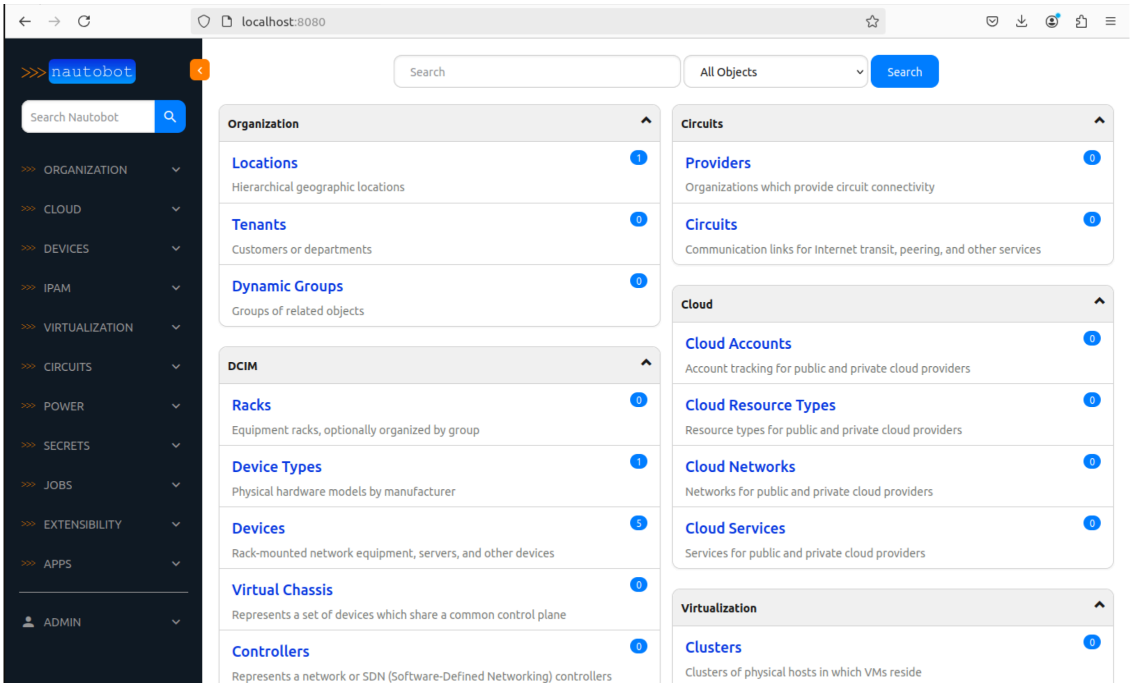
### Installation

To install Nautobot, I used this link to install Nautobot.

### Accessing the Web UI

Once the setup was complete, I accessed the Nautobot web UI by navigating to http://localhost:8080 in my browser.

Logged with the default admin credentials that I set up during the installation process.



**Organization**

| | | |
|---|---|---|
| Locations | Hierarchical geographic locations | 1 |
| Tenants | Customers or departments | 0 |
| Dynamic Groups | Groups of related objects | 0 |

**DCIM**

| | | |
|---|---|---|
| Racks | Equipment racks, optionally organized by group | 0 |
| Device Types | Physical hardware models by manufacturer | 1 |
| Devices | Rack-mounted network equipment, servers, and other devices | 5 |
| Virtual Chassis | Represents a set of devices which share a common control plane | 0 |
| Controllers | Represents a network or SDN (Software-Defined Networking) controllers | 0 |

**Circuits**

| | | |
|---|---|---|
| Providers | Organizations which provide circuit connectivity | 0 |
| Circuits | Communication links for Internet transit, peering, and other services | 0 |

**Cloud**

| | | |
|---|---|---|
| Cloud Accounts | Account tracking for public and private cloud providers | 0 |
| Cloud Resource Types | Resource types for public and private cloud providers | 0 |
| Cloud Networks | Networks for public and private cloud providers | 0 |
| Cloud Services | Services for public and private cloud providers | 0 |

**Virtualization**

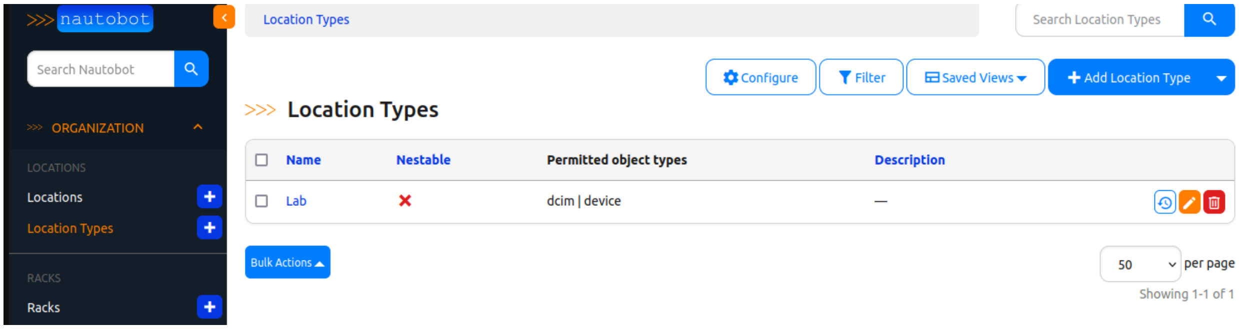
| | | |
|---|---|---|
| Clusters | Clusters of physical hosts in which VMs reside | 0 |

*Screenshot: Nautobot login page and dashboard after logging in.*

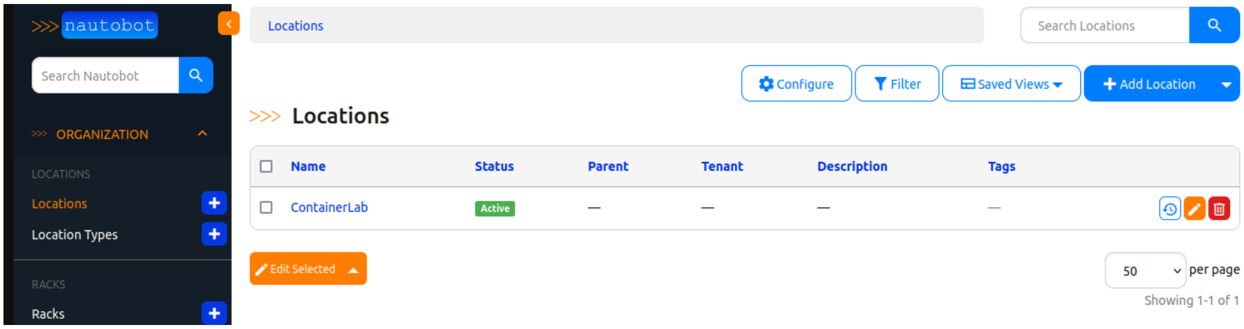
# Initial Configuration

After accessing the dashboard, I began by creating my site, device types, manufacturers, and other essential data for network management. Here's how I configured my Nautobot instance:

1. I added **Location** and **Location Type** as



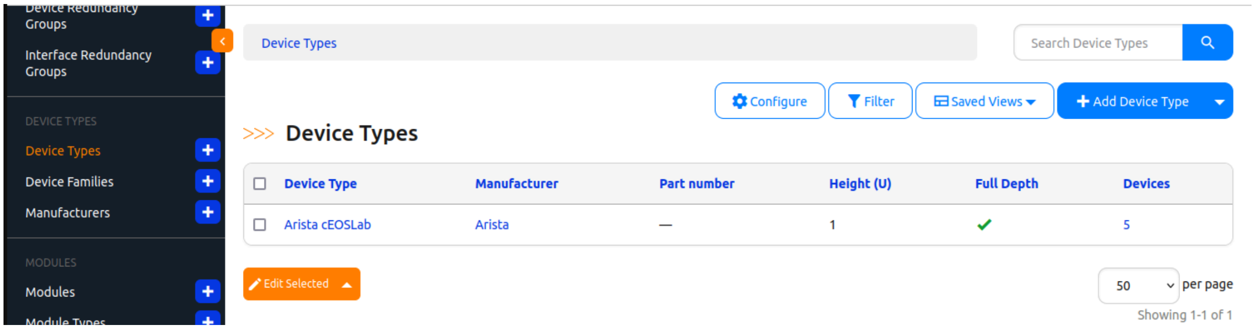
| ☐ | Name | Nestable | Permitted object types | Description |
|---|---|---|---|---|
| ☐ | Lab | ✗ | dcim \| device | — |



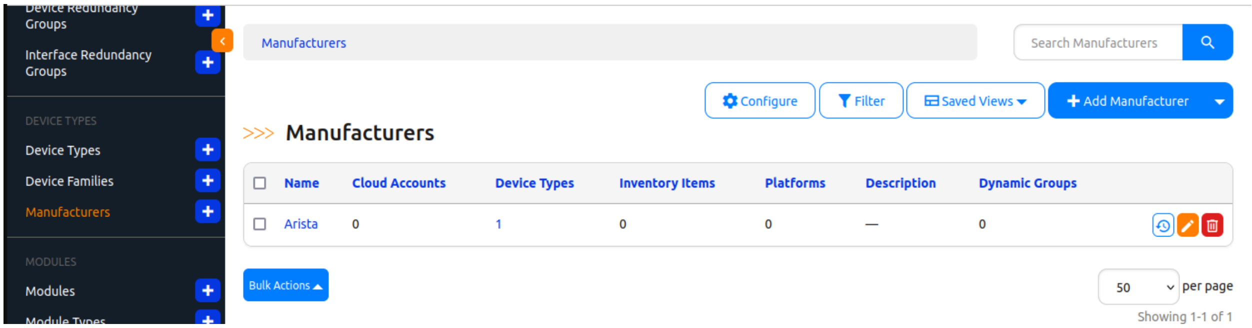
| ☐ | Name | Status | Parent | Tenant | Description | Tags |
|---|---|---|---|---|---|---|
| ☐ | ContainerLab | Active | — | — | — | — |

*Screenshot 1: Location and Location Type*

2. I added **Device Types** and **Manufacturers** relevant to my network setup.



| ☐ | Device Type | Manufacturer | Part number | Height (U) | Full Depth | Devices |
|---|---|---|---|---|---|---|
| ☐ | Arista cEOSLab | Arista | — | 1 | ✓ | 5 |

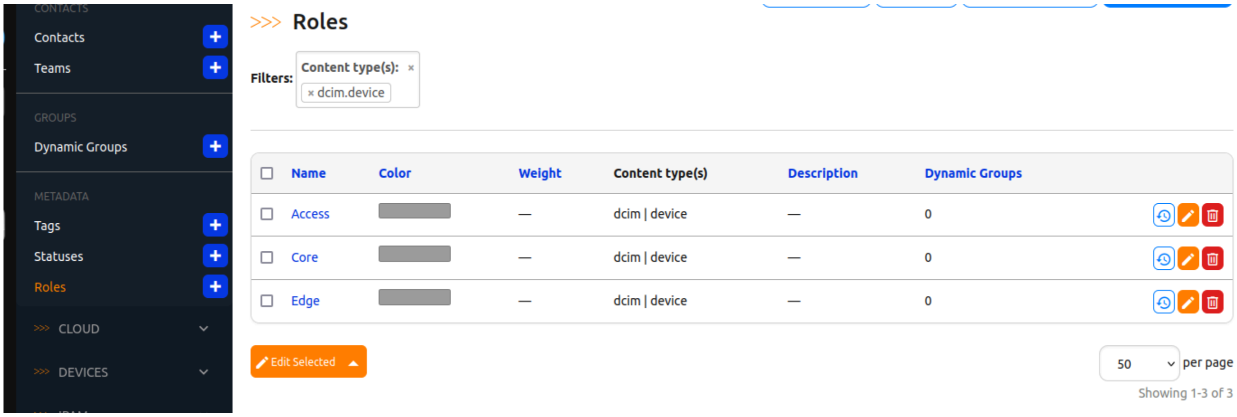


>>> **Manufacturers**

| □ | Name | Cloud Accounts | Device Types | Inventory Items | Platforms | Description | Dynamic Groups |
|---|---|---|---|---|---|---|---|
| □ | Arista | 0 | 1 | 0 | 0 | — | 0 |

*Screenshot 2: Device Types and Manufacturer*

3. I configured **Device Roles** and added 3 device types, Access, Core and Edge. I also set the Content Type to *dcim* | *device* so that the device role is visible in the Devices section.



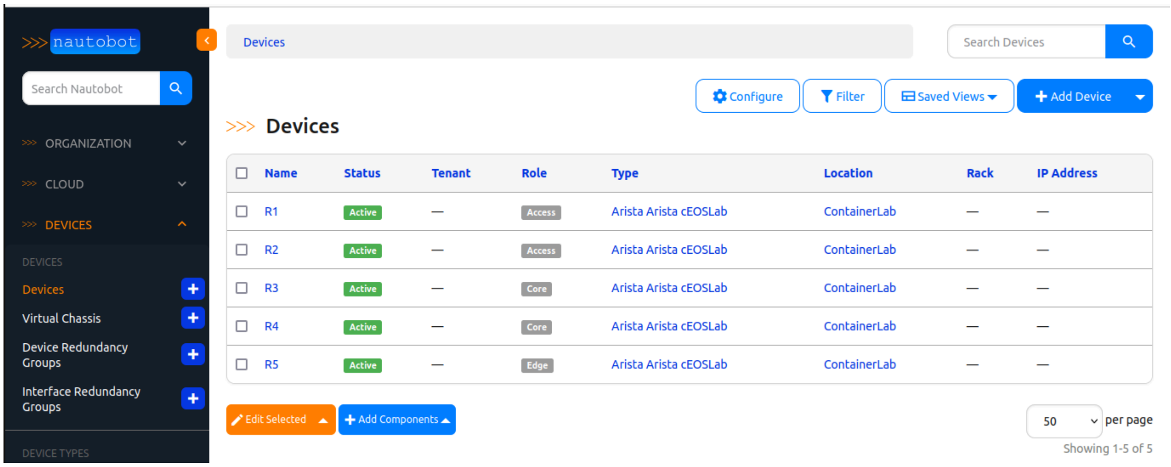
>>> **Roles**

Filters: Content type(s): × dcim.device

| □ | Name | Color | Weight | Content type(s) | Description | Dynamic Groups |
|---|---|---|---|---|---|---|
| □ | Access | ▬ | — | dcim \| device | — | 0 |
| □ | Core | ▬ | — | dcim \| device | — | 0 |
| □ | Edge | ▬ | — | dcim \| device | — | 0 |

*Screenshot 3: Device Roles configuration*

4. Post the device roles configuration, I configured the Devices.



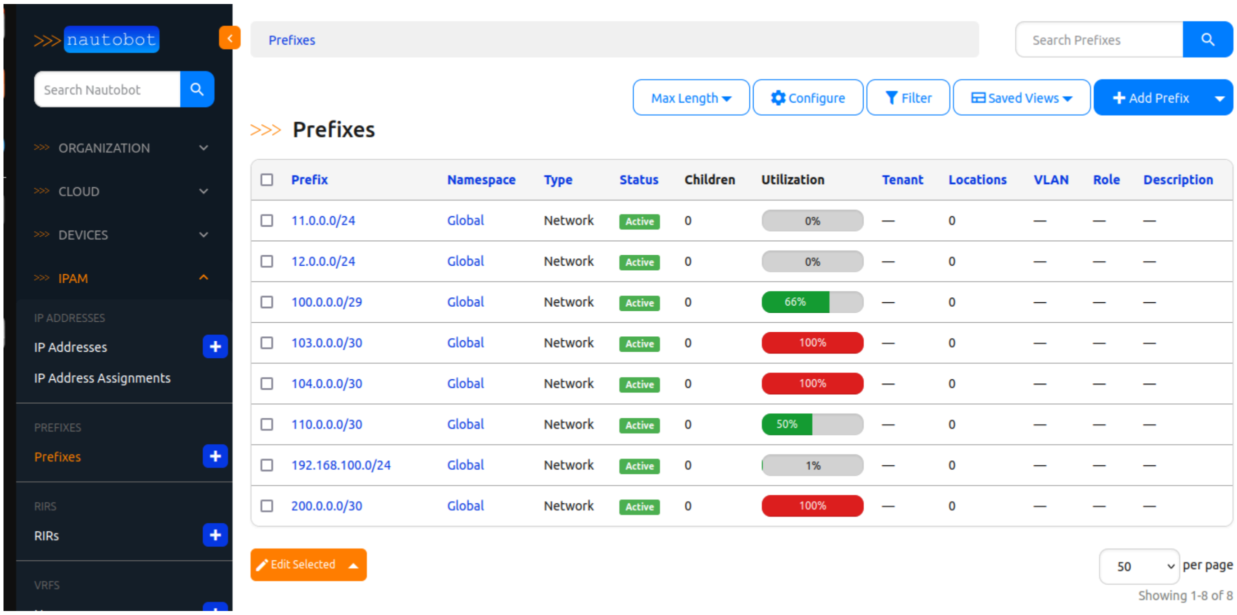
>>> **Devices**

| □ | Name | Status | Tenant | Role | Type | Location | Rack | IP Address |
|---|---|---|---|---|---|---|---|---|
| □ | R1 | Active | — | Access | Arista Arista cEOSLab | ContainerLab | — | — |
| □ | R2 | Active | — | Access | Arista Arista cEOSLab | ContainerLab | — | — |
| □ | R3 | Active | — | Core | Arista Arista cEOSLab | ContainerLab | — | — |
| □ | R4 | Active | — | Core | Arista Arista cEOSLab | ContainerLab | — | — |
| □ | R5 | Active | — | Edge | Arista Arista cEOSLab | ContainerLab | — | — |

*Screenshot 4: Devices configuration*

# Using Nautobot as Source of Truth

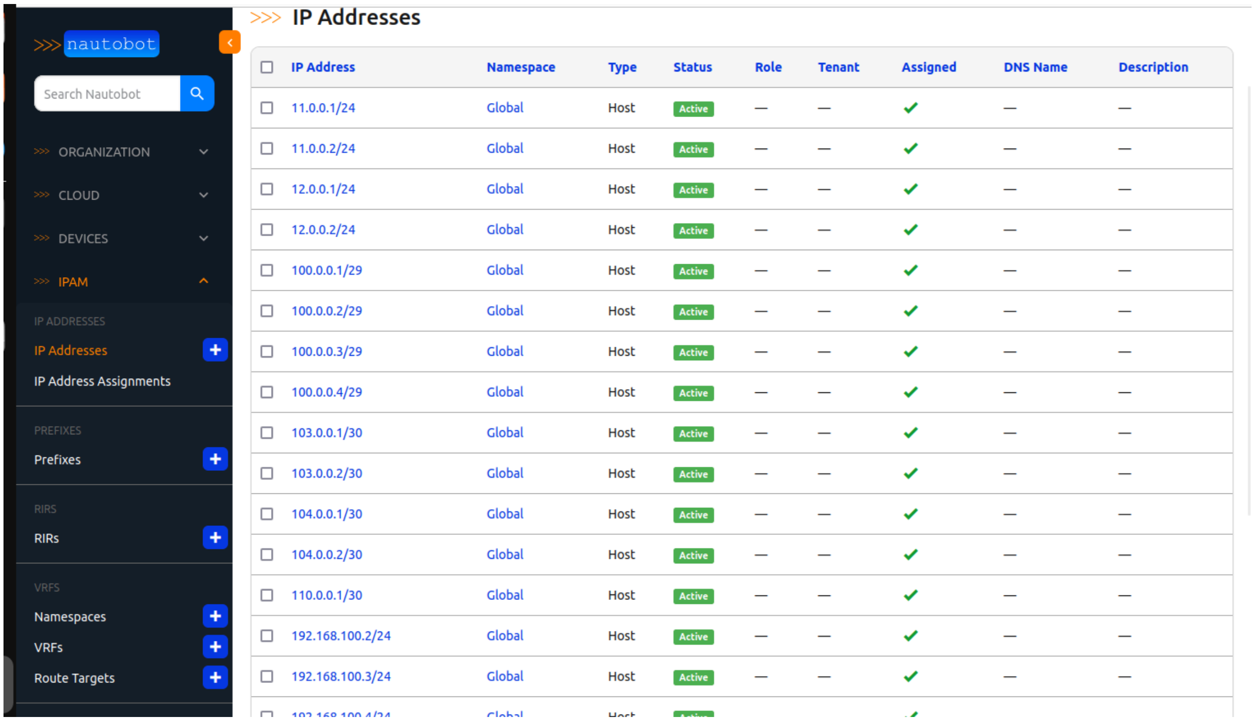
With the initial setup done, Nautobot became the source of truth for my network infrastructure. I configured IP prefixes.



>>> Prefixes

| □ | Prefix | Namespace | Type | Status | Children | Utilization | Tenant | Locations | VLAN | Role | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|
| □ | 11.0.0.0/24 | Global | Network | Active | 0 | 0% | — | 0 | — | — | — |
| □ | 12.0.0.0/24 | Global | Network | Active | 0 | 0% | — | 0 | — | — | — |
| □ | 100.0.0.0/29 | Global | Network | Active | 0 | 66% | — | 0 | — | — | — |
| □ | 103.0.0.0/30 | Global | Network | Active | 0 | 100% | — | 0 | — | — | — |
| □ | 104.0.0.0/30 | Global | Network | Active | 0 | 100% | — | 0 | — | — | — |
| □ | 110.0.0.0/30 | Global | Network | Active | 0 | 50% | — | 0 | — | — | — |
| □ | 192.168.100.0/24 | Global | Network | Active | 0 | 1% | — | 0 | — | — | — |
| □ | 200.0.0.0/30 | Global | Network | Active | 0 | 100% | — | 0 | — | — | — |

*Screenshot 5: Prefixes configuration*

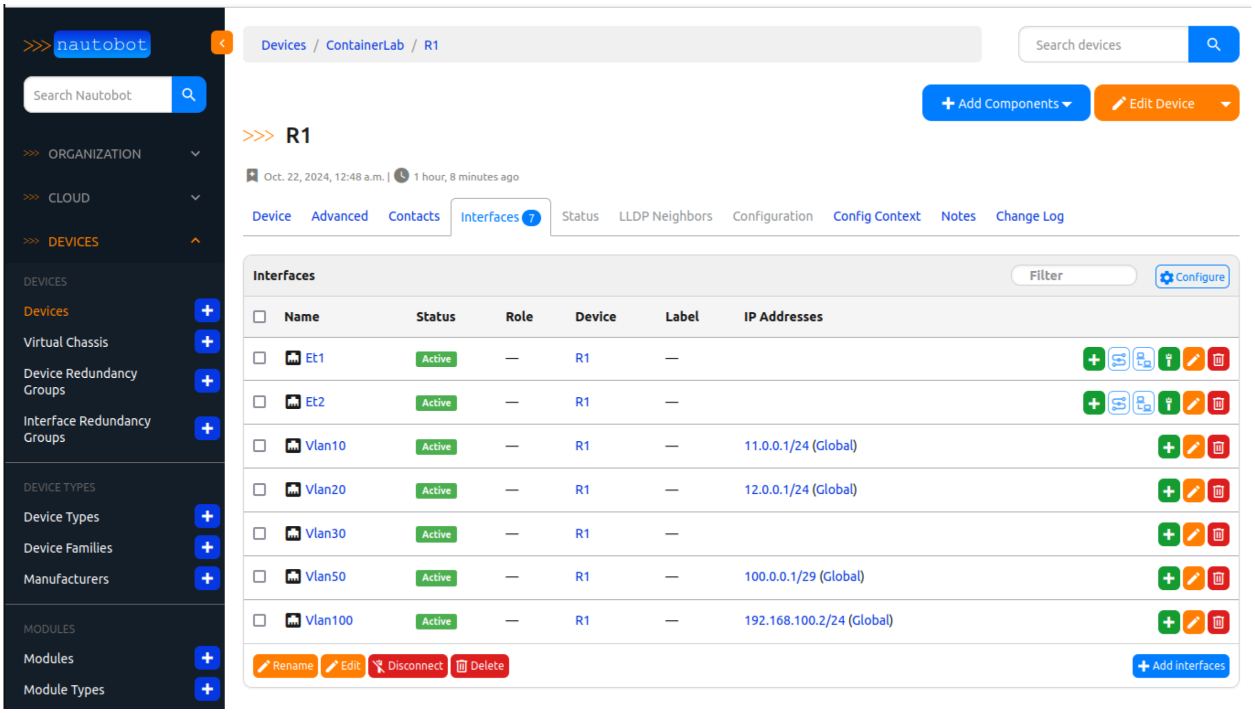


>>> IP Addresses

| □ | IP Address | Namespace | Type | Status | Role | Tenant | Assigned | DNS Name | Description |
|---|---|---|---|---|---|---|---|---|---|
| □ | 11.0.0.1/24 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 11.0.0.2/24 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 12.0.0.1/24 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 12.0.0.2/24 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 100.0.0.1/29 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 100.0.0.2/29 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 100.0.0.3/29 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 100.0.0.4/29 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 103.0.0.1/30 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 103.0.0.2/30 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 104.0.0.1/30 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 104.0.0.2/30 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 110.0.0.1/30 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 192.168.100.2/24 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 192.168.100.3/24 | Global | Host | Active | — | — | ✓ | — | — |
| □ | 192.168.100.4/24 | Global | Host | Active | | | ✓ | | |

*Screenshot 6: IP Addresses Configuration*

Post configuration of the IP addresses and prefixes, I assigned these IP addresses to each device's interfaces.



Devices / ContainerLab / R1

>>> R1

Oct. 22, 2024, 12:48 a.m. | 1 hour, 8 minutes ago

Interfaces

| Name | Status | Role | Device | Label | IP Addresses |
|---|---|---|---|---|---|
| Et1 | Active | — | R1 | — | |
| Et2 | Active | — | R1 | — | |
| Vlan10 | Active | — | R1 | — | 11.0.0.1/24 (Global) |
| Vlan20 | Active | — | R1 | — | 12.0.0.1/24 (Global) |
| Vlan30 | Active | — | R1 | — | |
| Vlan50 | Active | — | R1 | — | 100.0.0.1/29 (Global) |
| Vlan100 | Active | — | R1 | — | 192.168.100.2/24 (Global) |

*Screenshot 7: Device interface IP address*

# Updating password regularly

This section describes the implementation of an automated password update script for routers. The script generates random passwords and updates them on specified devices, with updates logged and reflected in a CSV file. A systemd service is used to ensure the script runs continuously, updating passwords at regular intervals.

## Prerequisites

Ensure the following dependencies are installed before running the script:

- Python 3.10
- Netmiko
- Loguru
- Secrets module

- A CSV file containing router information (I*P addresses, usernames, passwords, and device types*)

# Python Script: updatepassword.py

## Script Overview

The `updatepassword.py` script performs the following tasks:

1. **Generate a Random Password**: A new password is generated using Python's secrets library.
2. **Connect to the Routers**: The script uses Netmiko to establish an SSH connection to the router.
3. **Update Password**: The new password is applied to the router via a configuration command.
4. **Update CSV File**: The router's new password is stored in a CSV file for future reference.
5. **Repeat Every Hour**: The script runs in a continuous loop, updating passwords every hour.

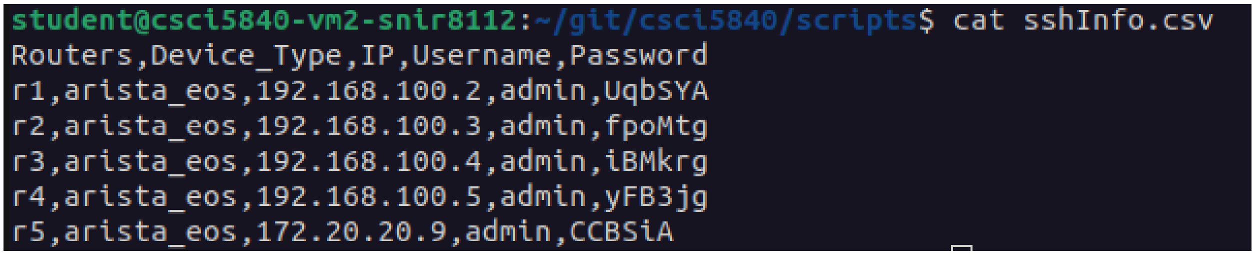
### Key Functions

- **generatePassword()**: Generates a secure, random password using secrets.token_urlsafe().
- **connectRouter(eos, hostname)**: Establishes an SSH connection to the router using Netmiko. The connection information is taken from a CSV file.
- **updatePassword(net_connect, ip, password)**: Sends the configuration command to update the router's password. This function uses Netmiko's send_config_set() method to apply the new password.
- **updatePasswordFile(file_path, ip, new_password)**: Updates the password for the corresponding IP address in the CSV file. This ensures that the new password is recorded.
- **Main Function**: The script runs a continuous loop, pulling router credentials from the sshInfo.csv file, generating new passwords, updating the routers, and saving the new passwords in the CSV file. The loop sleeps for one hour between updates.

### CSV File Format

The `sshInfo.csv` file holds information about routers and their current credentials. The file contains the following fields:

- **IP**: IP address of the router
- **Device_Type**: Router device type
- **Username**: SSH username
- **Password**: Current password for the router



```
student@csci5840-vm2-snir8112:~/git/csci5840/scripts$ cat sshInfo.csv
Routers,Device_Type,IP,Username,Password
r1,arista_eos,192.168.100.2,admin,UqbSYA
r2,arista_eos,192.168.100.3,admin,fpoMtg
r3,arista_eos,192.168.100.4,admin,iBMkrg
r4,arista_eos,192.168.100.5,admin,yFB3jg
r5,arista_eos,172.20.20.9,admin,CCBSiA
```

*Screenshot 8: sshInfo.csv*

## Service Overview

A systemd service is used to manage the continuous execution of the password update script. The service starts automatically on boot and ensures that the script runs continuously, with restarts in case of failure.

## Service Configuration

Below is the configuration for the `updatepassword.service` file:

```
[Unit]
Description=Update Router Password Service
After=network.target

[Service]
ExecStart=/usr/bin/python3
/home/student/git/csci5840/scripts/updatepassword.py
WorkingDirectory=/home/student/git/csci5840/scripts
StandardOutput=journal
StandardError=journal
Restart=on-failure
User=student

[Install]
WantedBy=multi-user.target
```

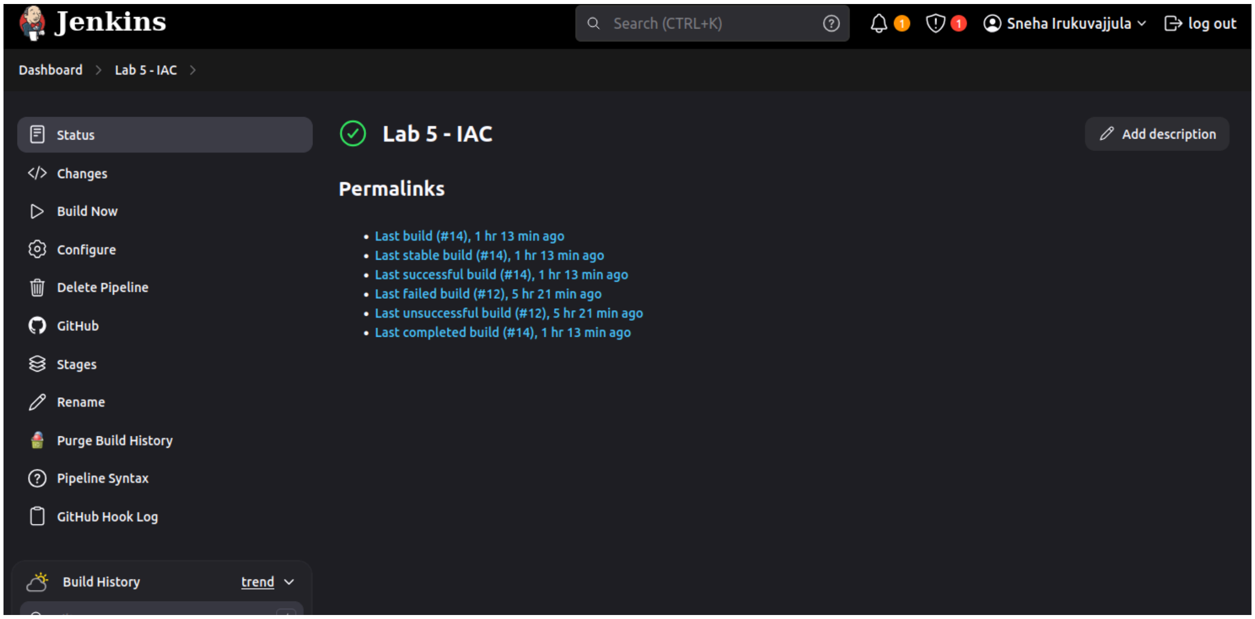
# Jenkins DevOps pipeline

## Installation

I used this link for installing Jenkins.

### Installed Required Plugins

1. I logged in to the Jenkins dashboard.
2. Navigated to **Manage Jenkins** -> **Manage Plugins**.
3. Installed the following plugins:
   - **Git Plugin**: To integrate with a Git repository.
   - **Pipeline Plugin**: For building pipelines using Jenkinsfile.
   - **SSH Plugin**: For remote execution of scripts.



Lab 5 - IAC

Permalinks

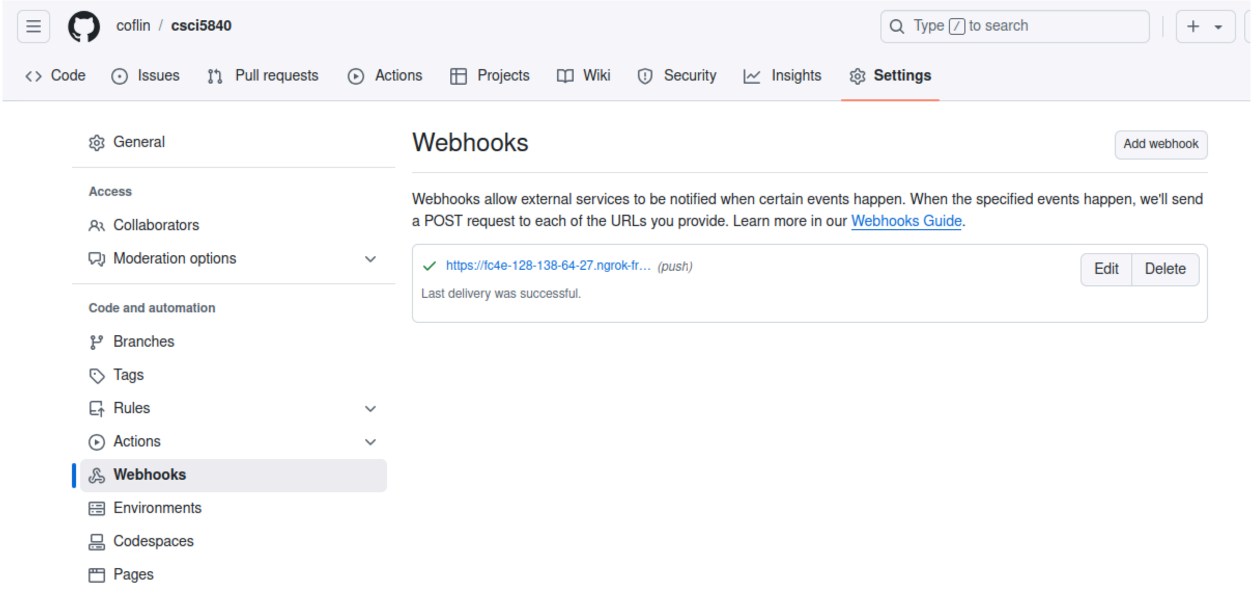
- Last build (#14), 1 hr 13 min ago
- Last stable build (#14), 1 hr 13 min ago
- Last successful build (#14), 1 hr 13 min ago
- Last failed build (#12), 5 hr 21 min ago
- Last unsuccessful build (#12), 5 hr 21 min ago
- Last completed build (#14), 1 hr 13 min ago

*Screenshot 9: Jenkins dashboard with Job*

### Set Up GitHub Webhook

To automatically trigger builds on code and configuration changes, I set up a webhook in the Git repository:

1. Navigated to the GitHub repository.
2. Went to **Settings** -> **Webhooks**.



Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

✓ https://fc4e-128-138-64-27.ngrok-fr… (push)

Last delivery was successful.

*Screenshot 10: Github webhook*

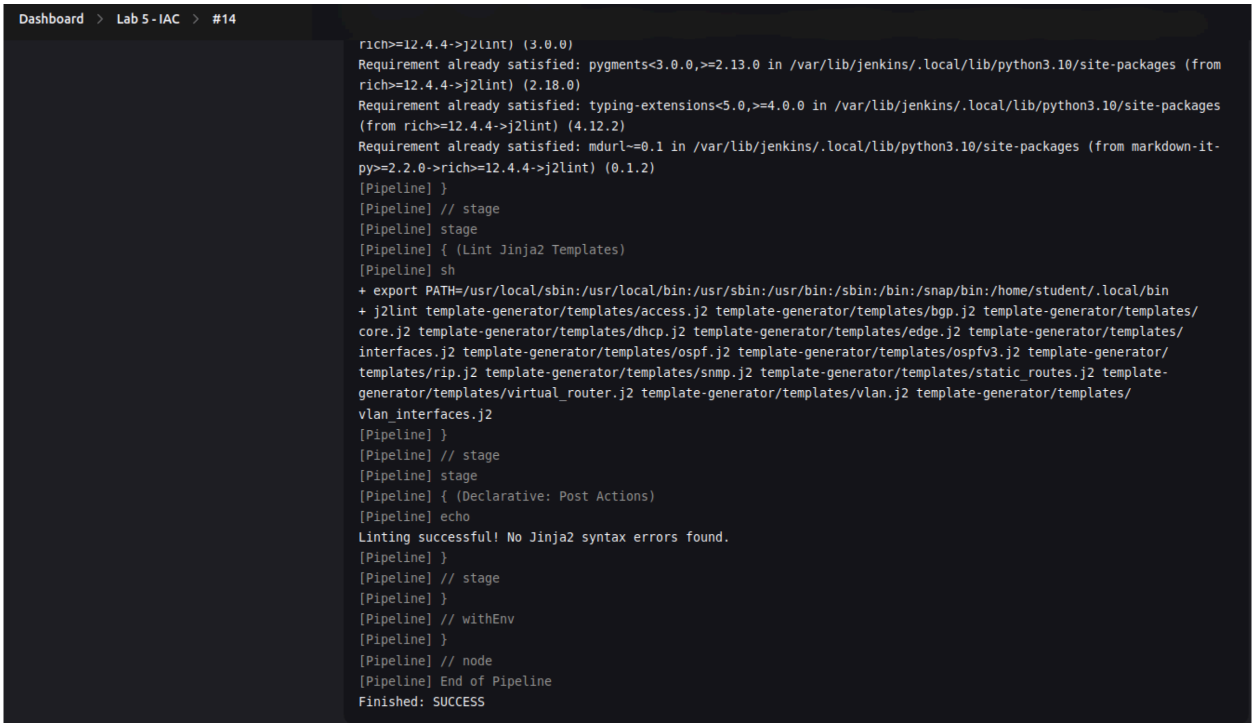
This webhook triggered the Jenkins pipeline whenever a change was pushed to the repository.

## Created a Jenkins Pipeline

### Scripted Pipeline (using a Jenkinsfile)

1. In the Git repository, I created a Jenkinsfile in the root directory.
2. The Jenkinsfile contained the pipeline definition. Below is the scripted pipeline I used for building, testing, and deploying the project:

```
pipeline {
      agent any

      stages {
      stage('Checkout Code') {
            steps {
                  // Pull the latest code (including Jinja2 templates) from
the repository
                  checkout scm
            }
      }

      stage('Install J2Lint') {
            steps {
                  // Install J2Lint if it's not already installed
                  sh 'pip install --user j2lint'
            }
      }

      stage('Lint Jinja2 Templates') {
            steps {
                  // Run J2Lint on all Jinja2 template files in the
directory
                  sh '''
                  export PATH=$PATH:/home/student/.local/bin && j2lint
template-generator/templates/*.j2
                  '''
            }
      }
      }

      post {
      success {
            echo 'Linting successful! No Jinja2 syntax errors found.'
      }
      failure {
            echo 'Linting failed! Jinja2 syntax errors detected.'
      }
      }
}
```



```
rich>=12.4.4->j2lint) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /var/lib/jenkins/.local/lib/python3.10/site-packages (from
rich>=12.4.4->j2lint) (2.18.0)
Requirement already satisfied: typing-extensions<5.0,>=4.0.0 in /var/lib/jenkins/.local/lib/python3.10/site-packages
(from rich>=12.4.4->j2lint) (4.12.2)
Requirement already satisfied: mdurl~=0.1 in /var/lib/jenkins/.local/lib/python3.10/site-packages (from markdown-it-
py>=2.2.0->rich>=12.4.4->j2lint) (0.1.2)
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Lint Jinja2 Templates)
[Pipeline] sh
+ export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin:/home/student/.local/bin
+ j2lint template-generator/templates/access.j2 template-generator/templates/bgp.j2 template-generator/templates/
core.j2 template-generator/templates/dhcp.j2 template-generator/templates/edge.j2 template-generator/templates/
interfaces.j2 template-generator/templates/ospf.j2 template-generator/templates/ospfv3.j2 template-generator/
templates/rip.j2 template-generator/templates/snmp.j2 template-generator/templates/static_routes.j2 template-
generator/templates/virtual_router.j2 template-generator/templates/vlan.j2 template-generator/templates/
vlan_interfaces.j2
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Linting successful! No Jinja2 syntax errors found.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

*Screenshot 11: Jenkins build successful*

# Summary: From Front-End to DevOps pipeline

1. **Front-End YAML File Generation**
   - I built a front-end form with fields for router configuration (VLAN, interfaces, static routes, OSPF, RIP) from the previous assignment.
   - Based on user inputs, the form dynamically generated a YAML file using Flask in the backend.
   - The generated YAML file was saved with a specific naming convention (*`hostname_routertype.yaml`*).
2. **Pushing YAML to GitHub**
   - Once the frontend form is submitted, it creates a YAML file. Once the YAML file is created, it pushes this file to the github repository.
3. **Jenkins Pipeline Trigger**
   - The GitHub repository was integrated with Jenkins through a webhook.

   - Every time the YAML file was pushed to GitHub, the webhook triggered the Jenkins pipeline.
   - Jenkins automatically pulled the latest YAML file from the repository.
4. **Pipeline Execution**:
   - **Build Stage**: Jenkins validated the structure of the YAML file.
   - **Test Stage**: It ran a J2 syntax error to verify the configuration.
   - **Deploy Stage**: The validated YAML configuration was deployed to network devices or a simulated environment.

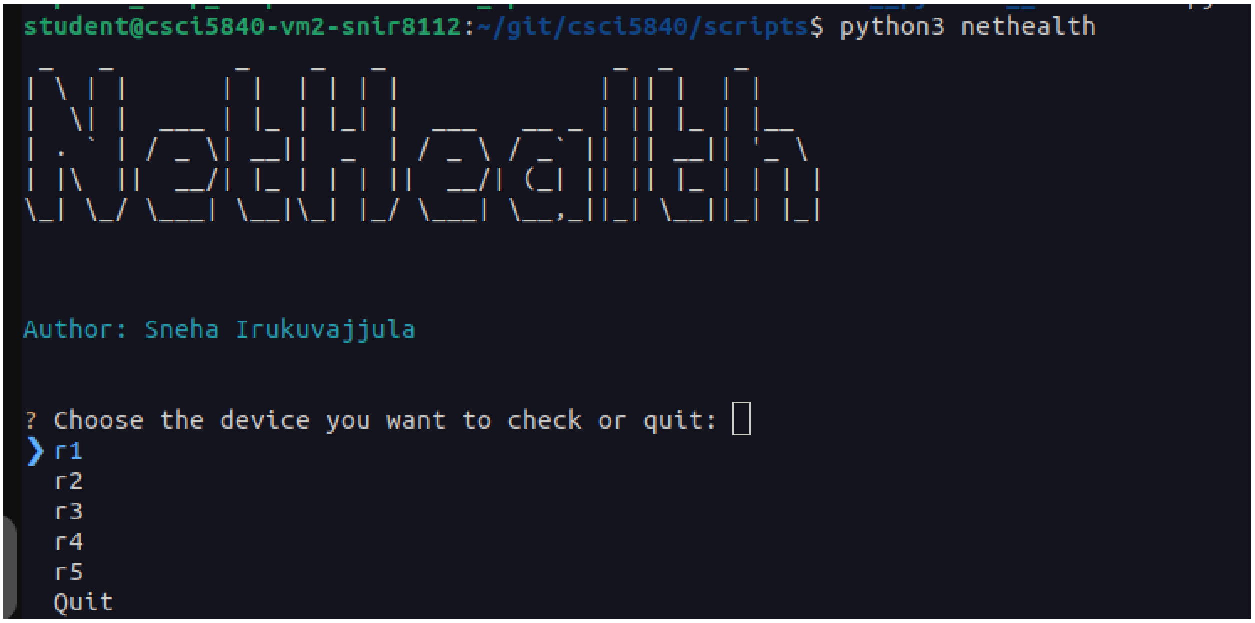
# Network Health Check Tool

### Device Health Checks

- CPU Usage (extracted percentage).
- OSPF/BGP Neighborship summaries.
- Route Table details.
- IP Connectivity via ping.

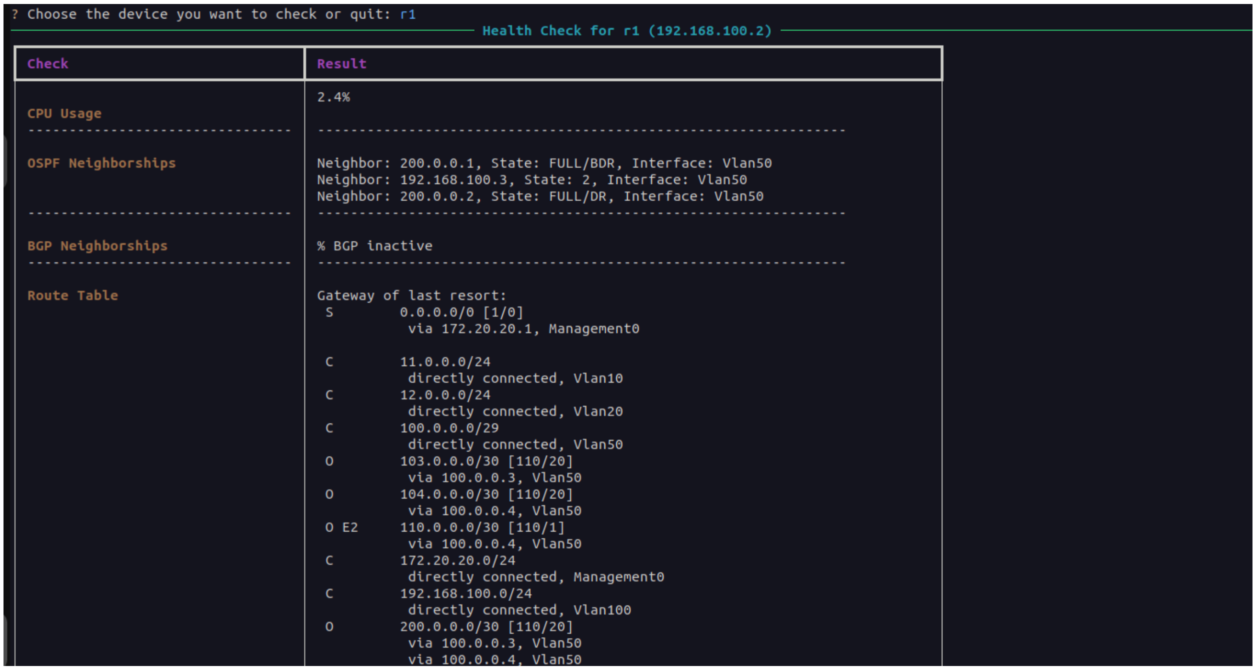
### Custom SSH Details

Automatically fetches SSH credentials from a CSV file (sshInfo.csv).

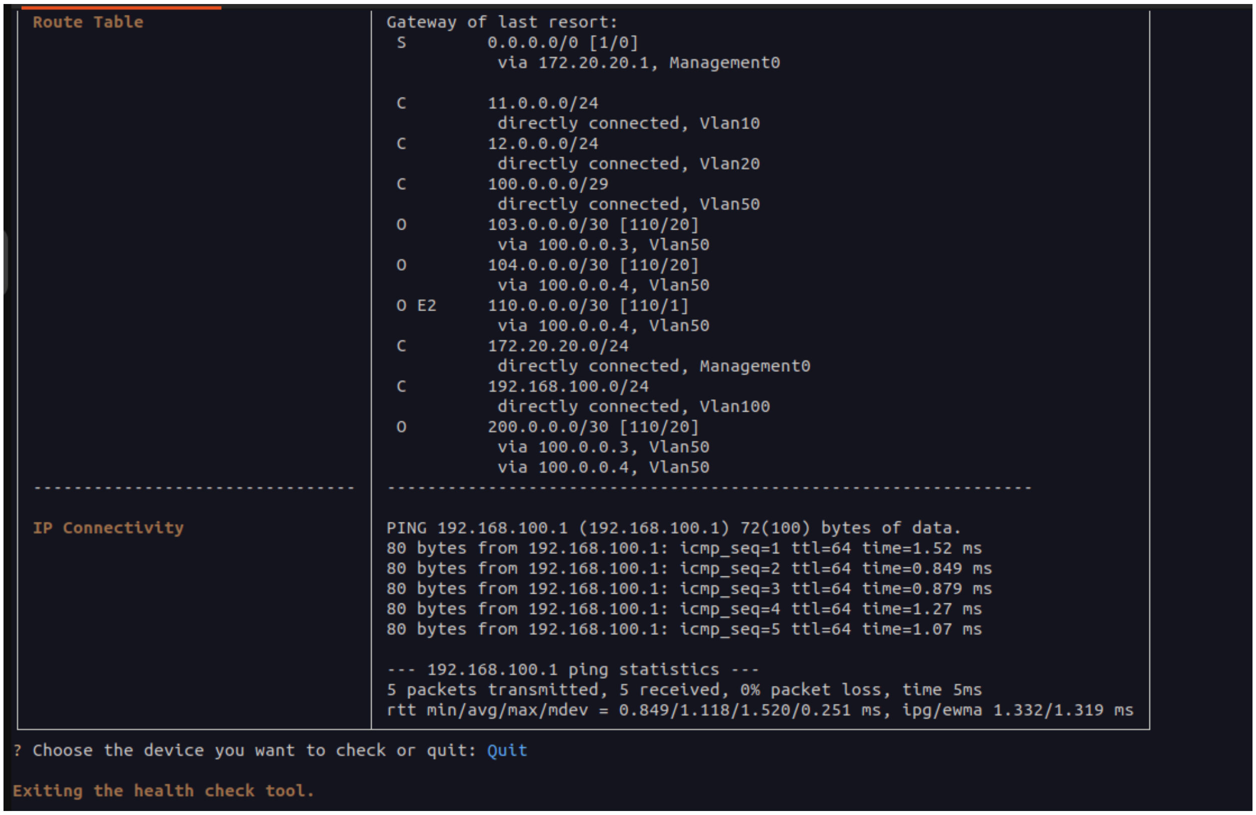


```
student@csci5840-vm2-snir8112:~/git/csci5840/scripts$ python3 nethealth
 _   _      _   _   _            _ _   _
| \ | | ___| |_| | | | ___  __ _| | |_| |__
|  \| |/ _ \ __| |_| |/ _ \/ _` | | __| '_ \
| |\  |  __/ |_|  _  |  __/ (_| | | |_| | | |
|_| \_|\___|\__|_| |_|\___|\__,_|_|\__|_| |_|



Author: Sneha Irukuvajjula

? Choose the device you want to check or quit:
❯ r1
  r2
  r3
  r4
  r5
  Quit
```

*Screenshot 12: CLI Tool to perform health checks*



```
? Choose the device you want to check or quit: r1
──────────────────────────────── Health Check for r1 (192.168.100.2) ────────────────────────────────
┌────────────────────────────────┬──────────────────────────────────────────────────────────────────┐
│ Check                          │ Result                                                           │
├────────────────────────────────┼──────────────────────────────────────────────────────────────────┤
│                                │ 2.4%                                                             │
│ CPU Usage                      │                                                                  │
│ ------------------------------ │ ---------------------------------------------------------------- │
│                                │                                                                  │
│ OSPF Neighborships             │ Neighbor: 200.0.0.1, State: FULL/BDR, Interface: Vlan50          │
│                                │ Neighbor: 192.168.100.3, State: 2, Interface: Vlan50             │
│                                │ Neighbor: 200.0.0.2, State: FULL/DR, Interface: Vlan50           │
│ ------------------------------ │ ---------------------------------------------------------------- │
│                                │                                                                  │
│ BGP Neighborships              │ % BGP inactive                                                   │
│ ------------------------------ │ ---------------------------------------------------------------- │
│                                │                                                                  │
│ Route Table                    │ Gateway of last resort:                                          │
│                                │  S        0.0.0.0/0 [1/0]                                        │
│                                │            via 172.20.20.1, Management0                          │
│                                │                                                                  │
│                                │  C        11.0.0.0/24                                            │
│                                │            directly connected, Vlan10                            │
│                                │  C        12.0.0.0/24                                            │
│                                │            directly connected, Vlan20                            │
│                                │  C        100.0.0.0/29                                           │
│                                │            directly connected, Vlan50                            │
│                                │  O        103.0.0.0/30 [110/20]                                  │
│                                │            via 100.0.0.3, Vlan50                                 │
│                                │  O        104.0.0.0/30 [110/20]                                  │
│                                │            via 100.0.0.4, Vlan50                                 │
│                                │  O E2     110.0.0.0/30 [110/1]                                   │
│                                │            via 100.0.0.4, Vlan50                                 │
│                                │  C        172.20.20.0/24                                         │
│                                │            directly connected, Management0                       │
│                                │  C        192.168.100.0/24                                       │
│                                │            directly connected, Vlan100                           │
│                                │  O        200.0.0.0/30 [110/20]                                  │
│                                │            via 100.0.0.3, Vlan50                                 │
│                                │            via 100.0.0.4, Vlan50                                 │
```



```
│ Route Table                    │ Gateway of last resort:                                                  │
│                                │  S        0.0.0.0/0 [1/0]                                                │
│                                │            via 172.20.20.1, Management0                                  │
│                                │                                                                          │
│                                │  C        11.0.0.0/24                                                    │
│                                │            directly connected, Vlan10                                    │
│                                │  C        12.0.0.0/24                                                    │
│                                │            directly connected, Vlan20                                    │
│                                │  C        100.0.0.0/29                                                   │
│                                │            directly connected, Vlan50                                    │
│                                │  O        103.0.0.0/30 [110/20]                                          │
│                                │            via 100.0.0.3, Vlan50                                         │
│                                │  O        104.0.0.0/30 [110/20]                                          │
│                                │            via 100.0.0.4, Vlan50                                         │
│                                │  O E2     110.0.0.0/30 [110/1]                                           │
│                                │            via 100.0.0.4, Vlan50                                         │
│                                │  C        172.20.20.0/24                                                 │
│                                │            directly connected, Management0                               │
│                                │  C        192.168.100.0/24                                               │
│                                │            directly connected, Vlan100                                   │
│                                │  O        200.0.0.0/30 [110/20]                                          │
│                                │            via 100.0.0.3, Vlan50                                         │
│                                │            via 100.0.0.4, Vlan50                                         │
│ ------------------------------ │ ---------------------------------------------------------------- │
│                                │                                                                          │
│ IP Connectivity                │ PING 192.168.100.1 (192.168.100.1) 72(100) bytes of data.                │
│                                │ 80 bytes from 192.168.100.1: icmp_seq=1 ttl=64 time=1.52 ms              │
│                                │ 80 bytes from 192.168.100.1: icmp_seq=2 ttl=64 time=0.849 ms             │
│                                │ 80 bytes from 192.168.100.1: icmp_seq=3 ttl=64 time=0.879 ms             │
│                                │ 80 bytes from 192.168.100.1: icmp_seq=4 ttl=64 time=1.27 ms              │
│                                │ 80 bytes from 192.168.100.1: icmp_seq=5 ttl=64 time=1.07 ms              │
│                                │                                                                          │
│                                │ --- 192.168.100.1 ping statistics ---                                    │
│                                │ 5 packets transmitted, 5 received, 0% packet loss, time 5ms              │
│                                │ rtt min/avg/max/mdev = 0.849/1.118/1.520/0.251 ms, ipg/ewma 1.332/1.319 ms │
└────────────────────────────────┴──────────────────────────────────────────────────────────────────────────┘
? Choose the device you want to check or quit: Quit

Exiting the health check tool.
```

*Screenshot 13: Device health check of R1*