Overview: Network Automation Lab with Python & GNS3

This hands-on project simulates a real-world network automation workflow using GNS3, Telnet, and Python scripting. The objective is to automate the configuration of routers and switches across a virtual lab environment—replicating scalable enterprise network scenarios. It involves building a network topology, writing Python scripts to provision network devices, and validating configurations end-to-end.

Through this lab, we explore the following:

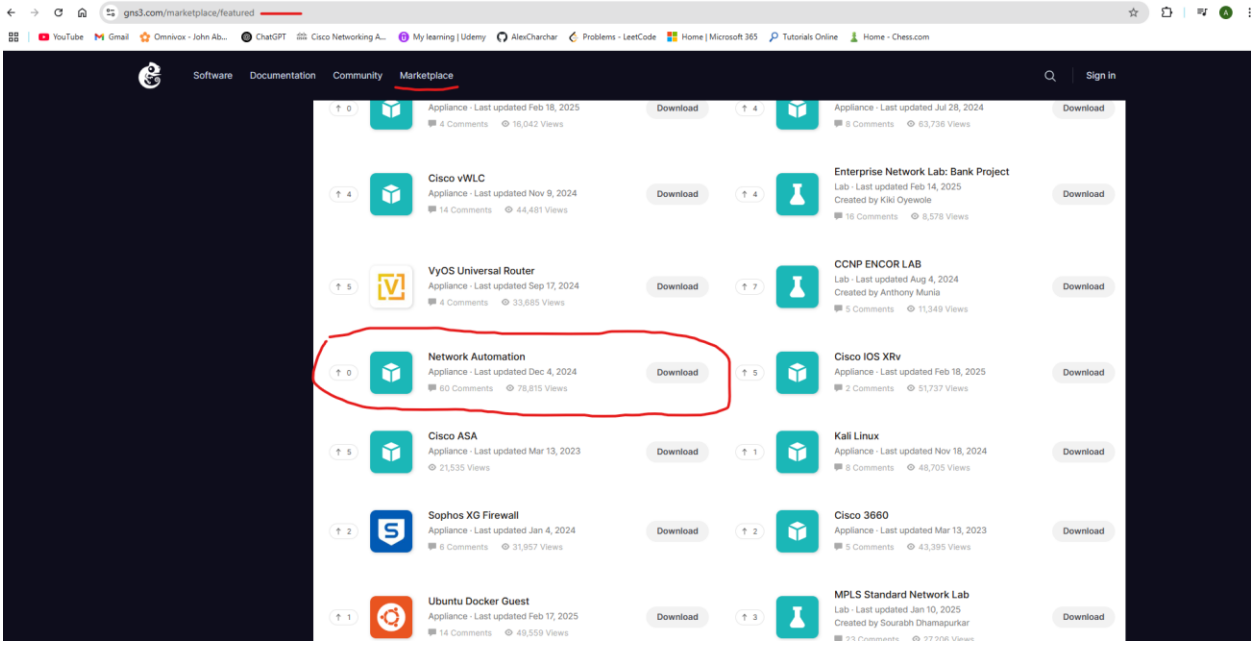
- **Practical scripting experience:** Writing Python automation scripts using the telnetlib library to push VLAN configurations, interface settings, and routing logic to multiple Cisco devices.
- **Progressive complexity:** Starting with a single router and switch, and expanding to orchestrate VLAN setups across five interconnected switches using loops and dynamic IP lists.
- **Real-world skills:** Reading and applying vendor documentation (e.g., Python's official Telnet docs), handling debugging steps, and modifying configuration scripts to remove unnecessary password prompts or streamline command logic.
- **Linux-based deployment:** Scripts are executed from a Dockerized network automation container within GNS3, offering exposure to Linux environments, CLI tooling, and permission handling (chmod, shebang usage, etc.).
- **Outcome:** By the end of this lab, learners are equipped with foundational network automation knowledge, a working understanding of remote device management, and practical exposure to infrastructure-as-code principles

Contents

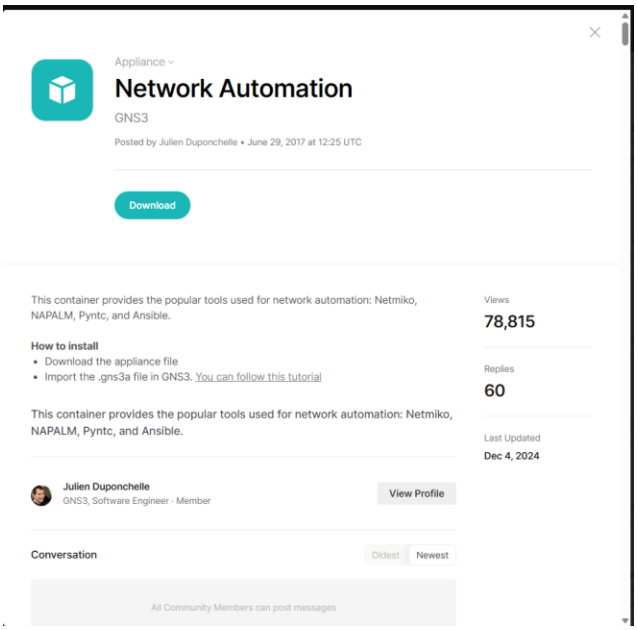
Configuring Network Devices on GNS3 Using Python (Part 1)

1. Download and Add the Network Automation Device to GNS3

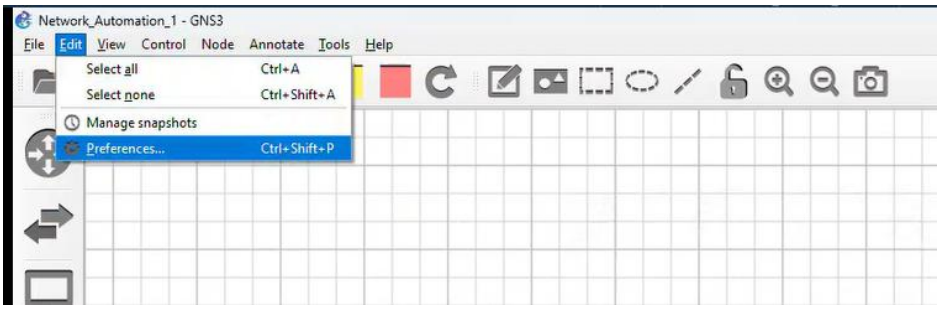
1. Go to the **GNS3 website**.
2. Click on **Marketplace** → Select **Network Automation**.



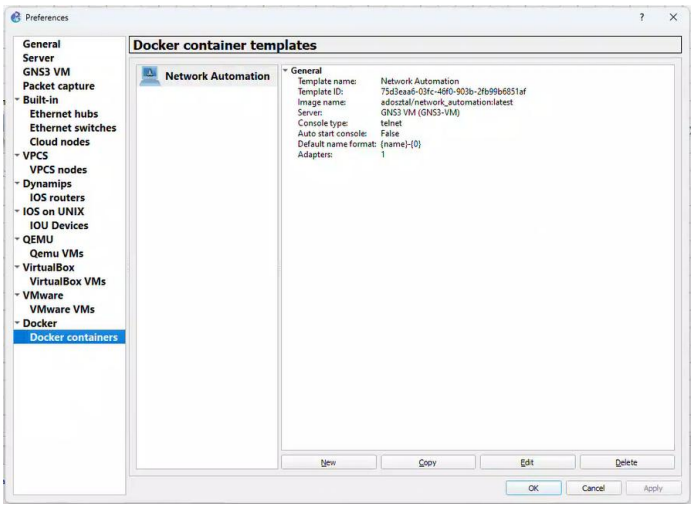
3. Click **Download**.

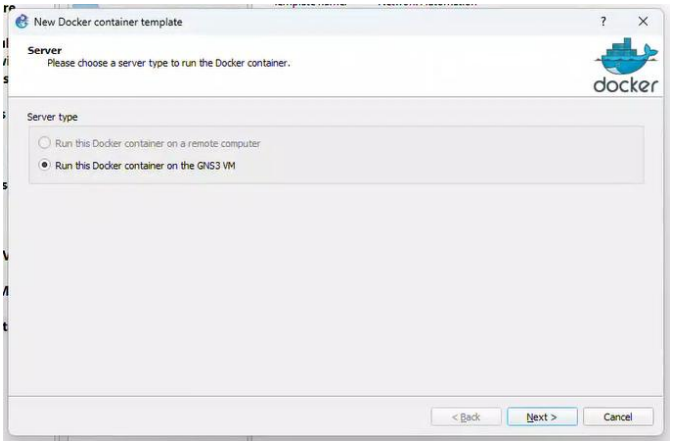


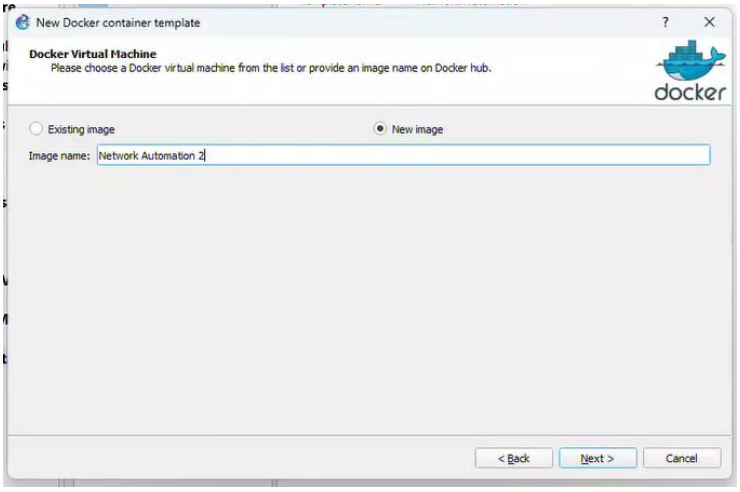
4. Open **GNS3** and add the downloaded Network Automation device.
5. Click on **Edit** (top left) → **Preferences**.

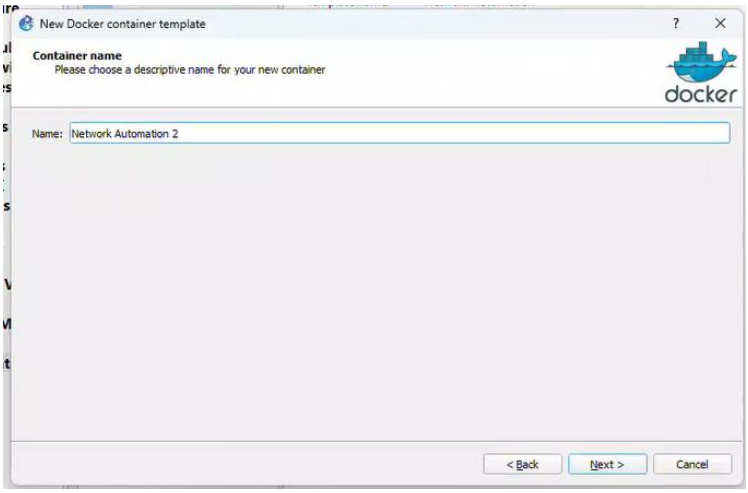


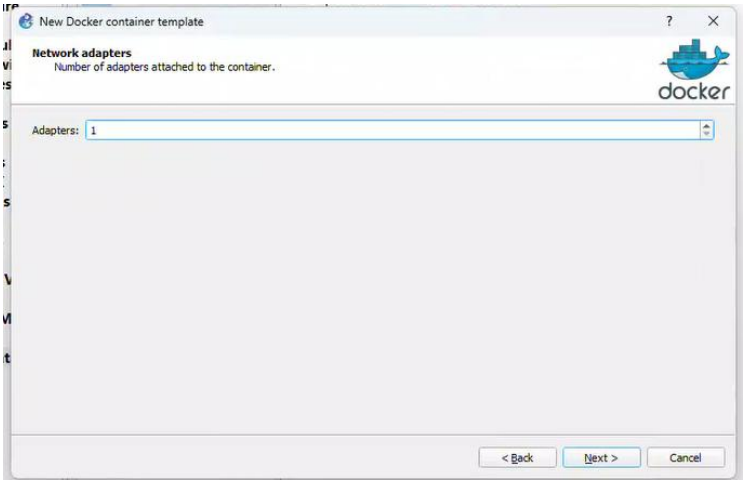
6. Under **Docker** → **Docker Containers**, add the **Network Automation** device.

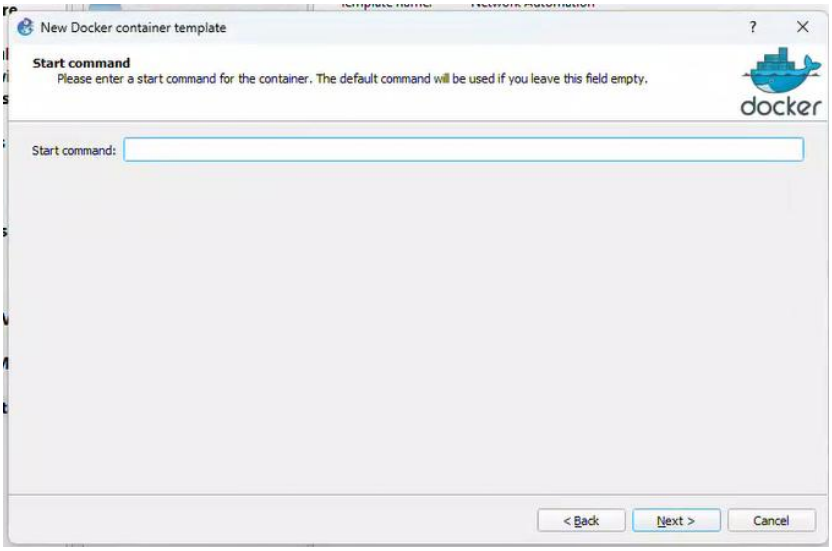

Preferences
General
Server
GNS3 VM
Packet capture
Built-in
Ethernet hubs
Ethernet switches
Cloud nodes
VPCS
VPCS nodes
Dynamips
IOS routers
IOS on UNIX
IOU Devices
QEMU
Qemu VMs
VirtualBox
VirtualBox VMs
VMware
VMware VMs
Docker
Docker containers
Docker container templates
Network Automation
General
Template name: Network Automation
Template ID: 75d3eaa6-03fc-4660-903b-2fb99b6851af
Image name: adosztal/network_automation:latest
Server: GNS3 VM (GNS3-VM)
Console type: telnet
Auto start console: False
Default name format: {name}-{0}
Adapters: 1
New
Copy
Edit
Delete
OK
Cancel
Apply

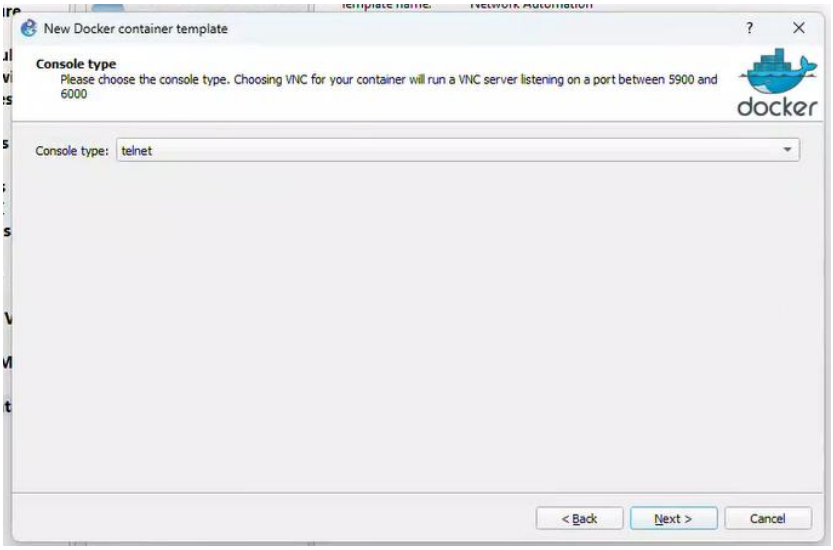

New Docker container template
Server
Please choose a server type to run the Docker container.
docker
Server type
Run this Docker container on a remote computer
Run this Docker container on the GNS3 VM
< Back
Next >
Cancel

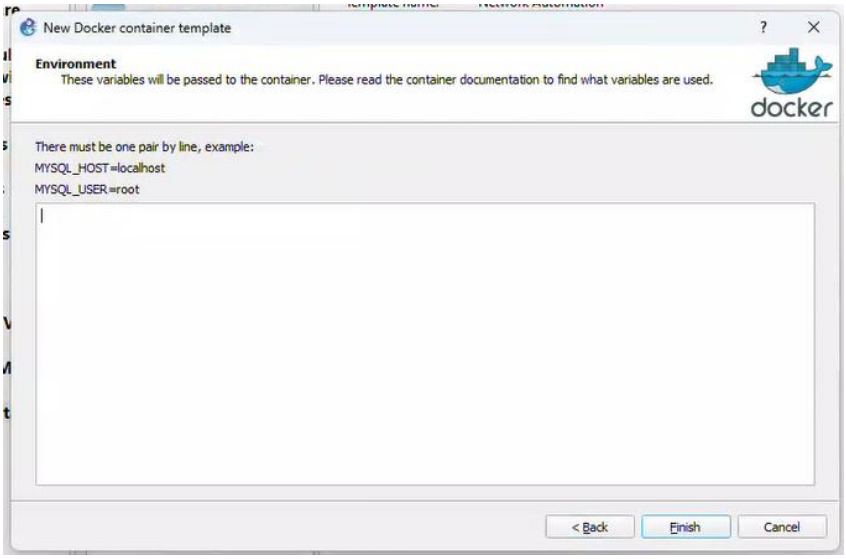

New Docker container template
Docker Virtual Machine
Please choose a Docker virtual machine from the list or provide an image name on Docker hub.
docker
Existing image
New image
Image name: Network Automation 2
< Back
Next >
Cancel


New Docker container template
Container name
Please choose a descriptive name for your new container
docker
Name: Network Automation 2
< Back
Next >
Cancel


New Docker container template
Network adapters
Number of adapters attached to the container.
docker
Adapters: 1
< Back
Next >
Cancel

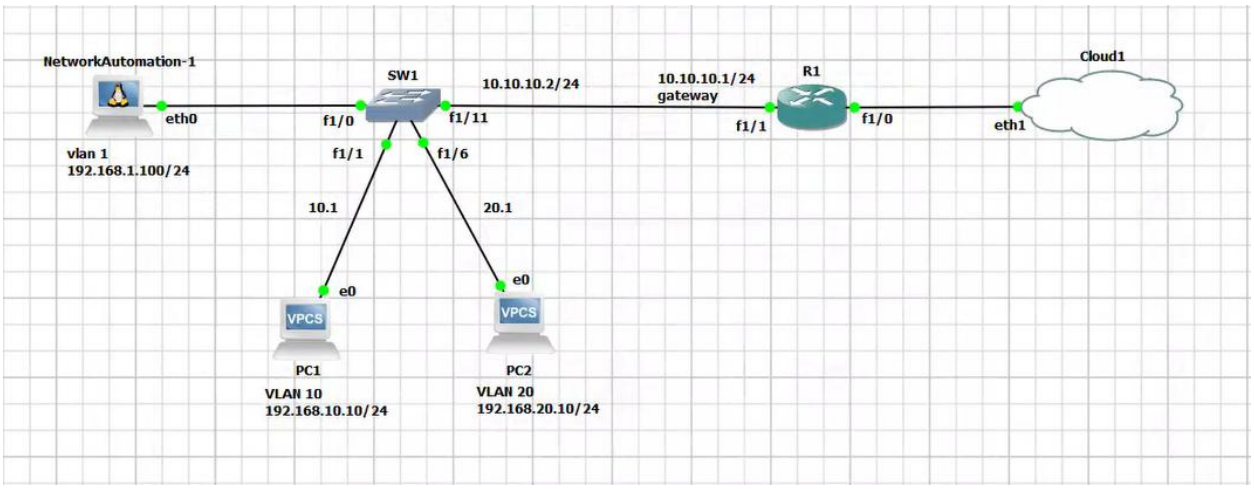

New Docker container template
Start command
Please enter a start command for the container. The default command will be used if you leave this field empty.
docker
Start command:
< Back
Next >
Cancel





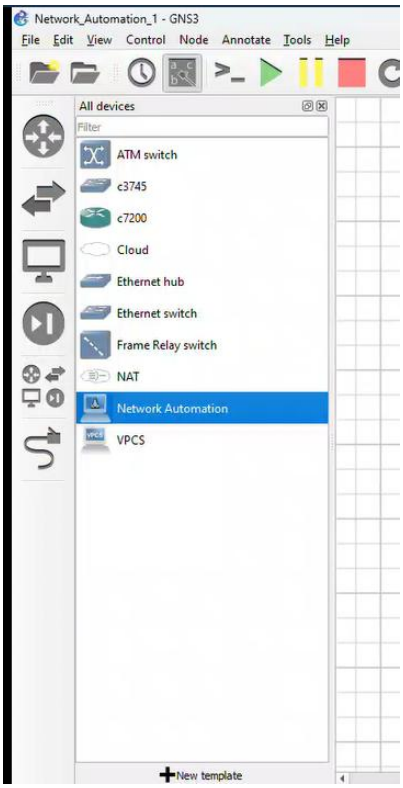
2. Create the Network Topology

1. Create a network topology in GNS3.

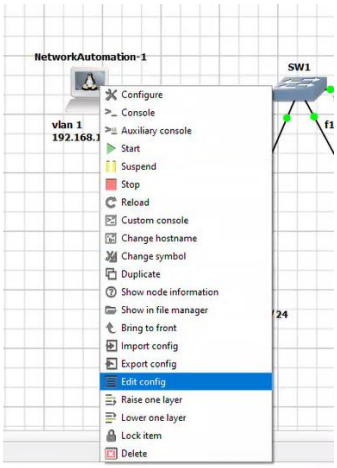


- Note: This topology may include additional configurations such as **Inter-VLAN routing**, but these are not necessary for practicing automation. You only need to ensure that your **Network Automation** device can ping external networks.

2. Drag the **Network Automation** device into the topology.



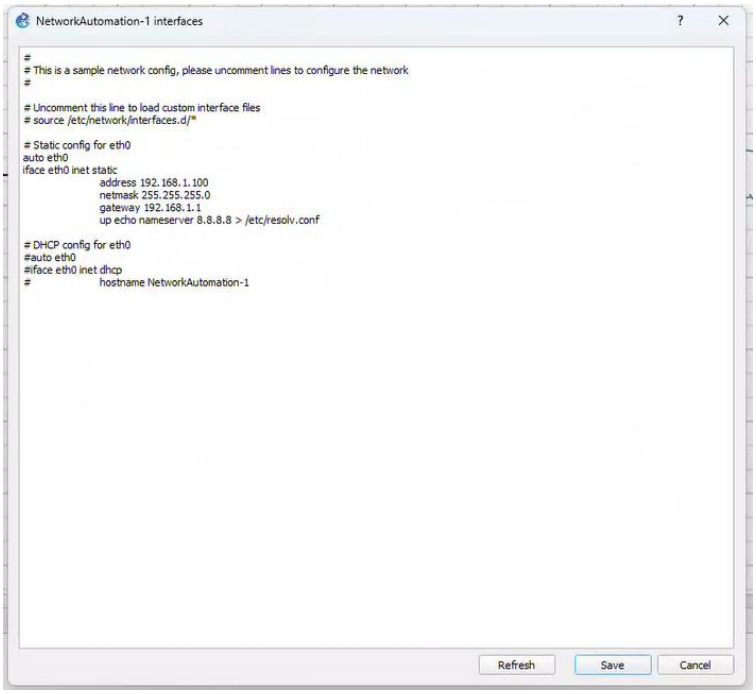
3. **Right-click** on the device → Select **Edit Config**.



4. Uncomment the relevant lines to configure a **static IP address**, **netmask**, **gateway**, and **DNS server**.

5. Your configuration should look something like this:

```
auto eth0
iface eth0 inet static
address 192.168.1.100
netmask 255.255.255.0
gateway 192.168.1.1
up echo nameserver 8.8.8.8 > /etc/resolv.conf
```



3. Prepare for Python-Based Automation

1. Visit the **Python Telnet documentation**: https://docs.python.org/3.12/library/telnetlib.html
2. Scroll down to the **script example** and copy it.
    - This will serve as your template.

Telnet Example

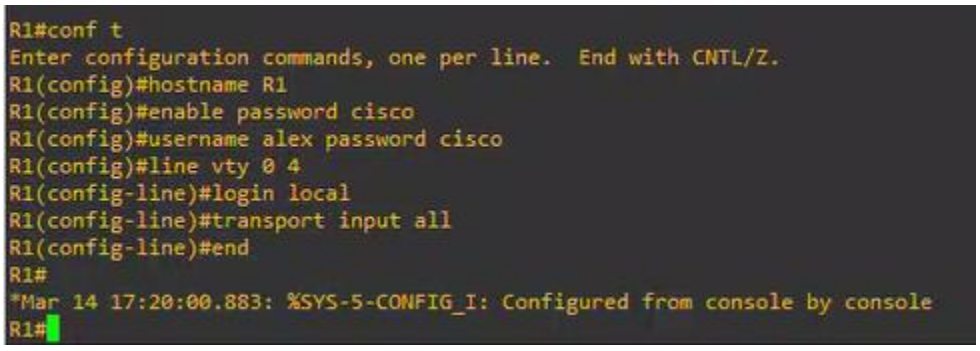
A simple example illustrating typical use:

```
import getpass
import telnetlib

HOST = "localhost"
user = input("Enter your remote account: ")
password = getpass.getpass()

tn = telnetlib.Telnet(HOST)

tn.read_until(b"login: ")
tn.write(user.encode('ascii') + b"\n")
if password:
    tn.read_until(b"Password: ")
    tn.write(password.encode('ascii') + b"\n")

tn.write(b"ls\n")
tn.write(b"exit\n")

print(tn.read_all().decode('ascii'))
```

4. Configure the Router (R1)

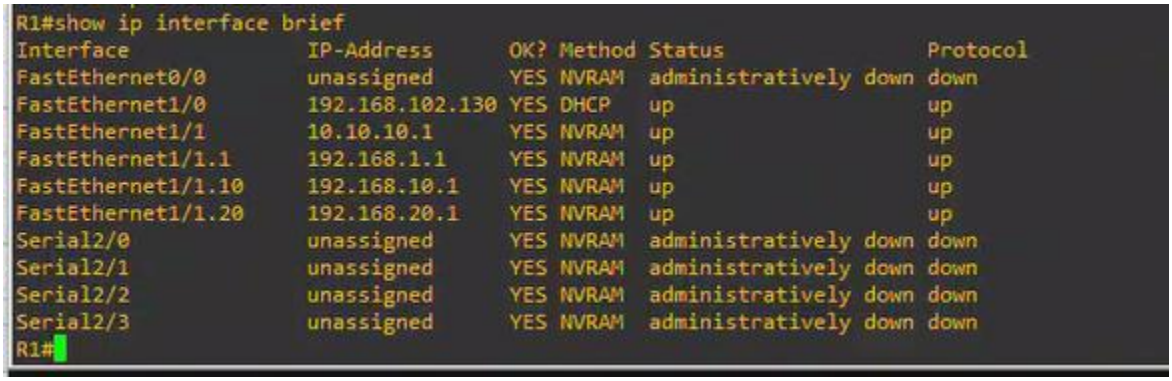
1. Power on all network devices in GNS3.
2. Open the router (R1) and enter the following configuration:

```
conf t
hostname R1
enable password cisco
username alex password cisco
line vty 0 4
login local
transport input all
end
```



3. Assign an IP address to the appropriate interface (e.g., **FastEthernet1/1**):

```
interface FastEthernet1/1
ip address 10.10.10.1 255.255.255.0
no shutdown
```



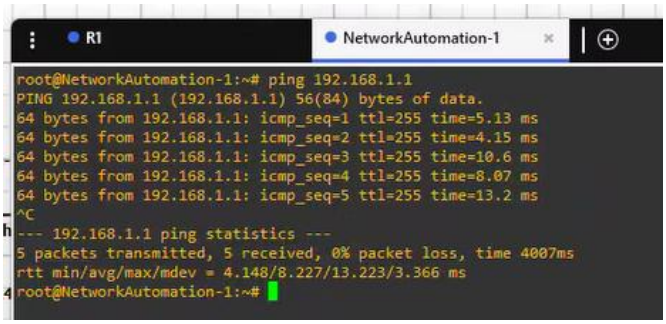
4. Verify that Telnet access works:

```
telnet 10.10.10.1
```

5. Verify Network Connectivity

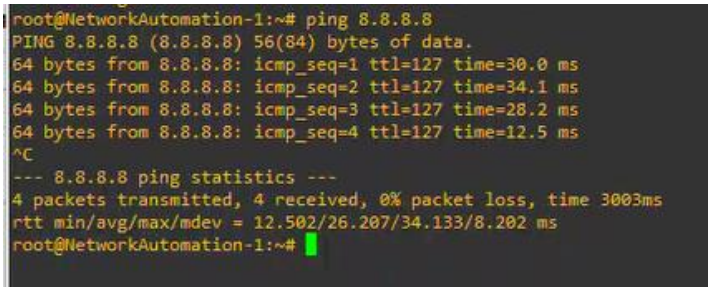
1. From your **Network Automation** device:
    - Ping the **gateway**:

```
ping 192.168.1.1
```



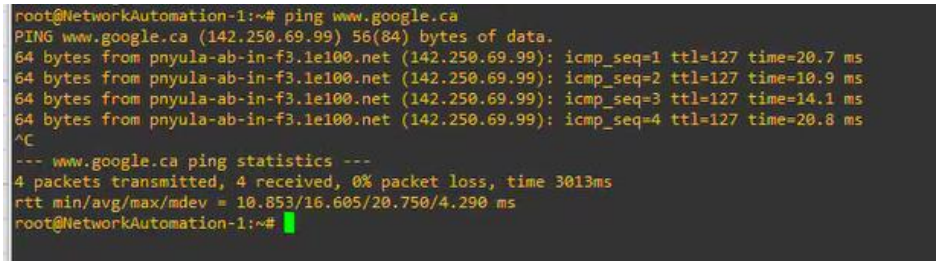
- Ping an external server:

  ping 8.8.8.8



- Ping a website to check DNS resolution:

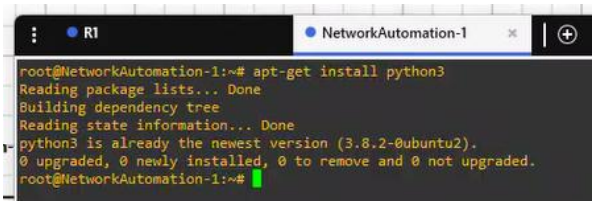
  ping www.google.ca



6. Install Python and Run a Telnet Script

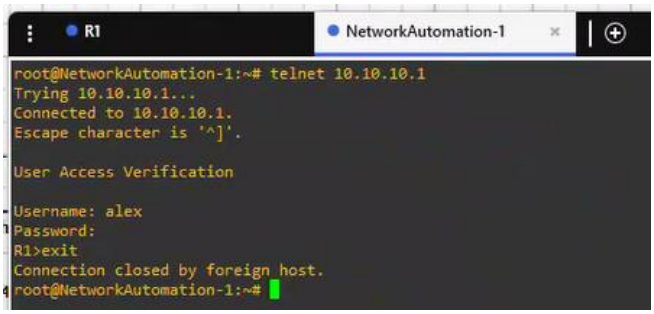
1. Install Python 3 (if not already installed):

   apt-get install python3



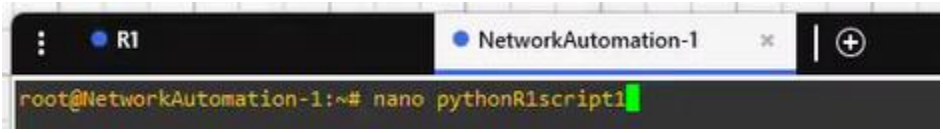
2. Verify that Telnet access works from the **Network Automation** device:

   telnet 10.10.10.1



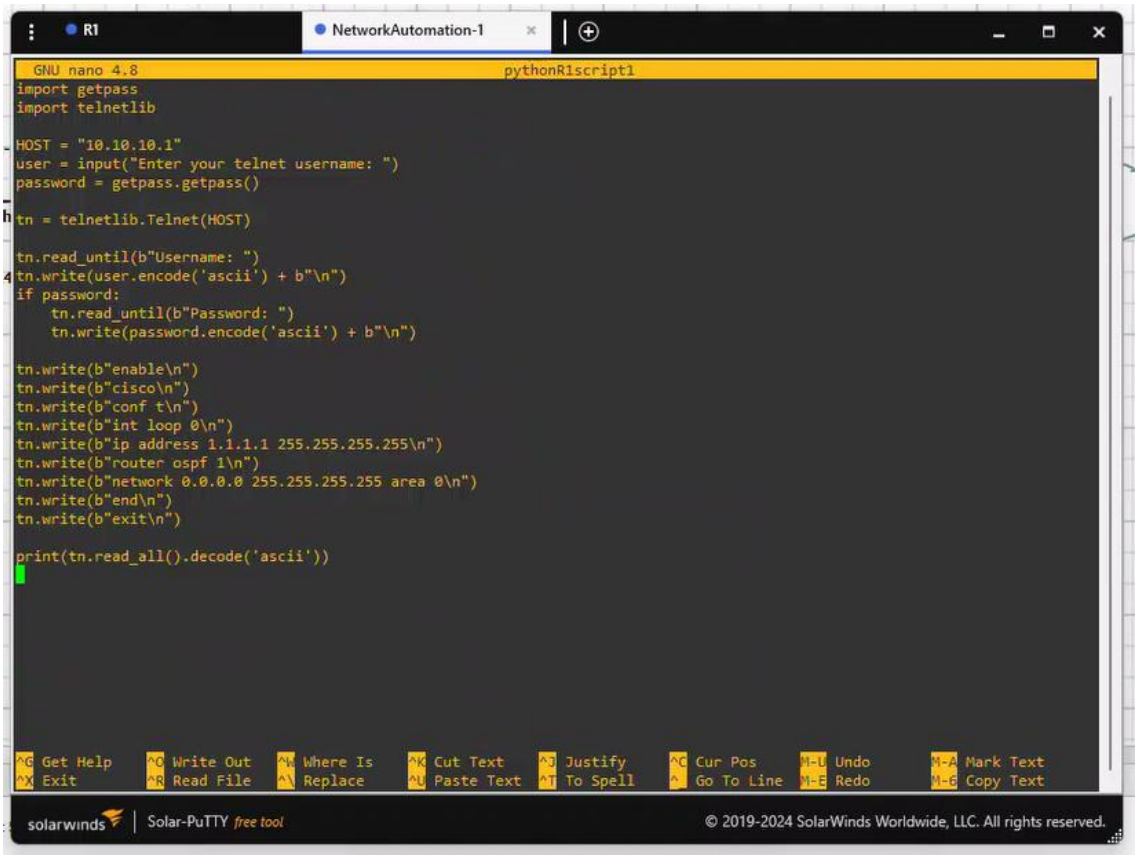
3. Create a new text file to use as our Python script:

   nano pythonR1script1



4. Enter the following script (adjust IP addresses and credentials as needed):

```
import getpass
import telnetlib

HOST = "10.10.10.1"
user = input("Enter your telnet username: ")
password = getpass.getpass()

tn = telnetlib.Telnet(HOST)

tn.read_until(b"Username: ")
tn.write(user.encode('ascii') + b"\n")
if password:
tn.read_until(b"Password: ")
tn.write(password.encode('ascii') + b"\n")

tn.write(b"enable\n")
tn.write(b"cisco\n")
tn.write(b"conf t\n")
tn.write(b"int loop 0\n")
tn.write(b"ip address 1.1.1.1 255.255.255.255\n")
tn.write(b"router ospf 1\n")
tn.write(b"network 0.0.0.0 255.255.255.255 area 0\n")
tn.write(b"end\n")
tn.write(b"exit\n")

print(tn.read_all().decode('ascii'))
```



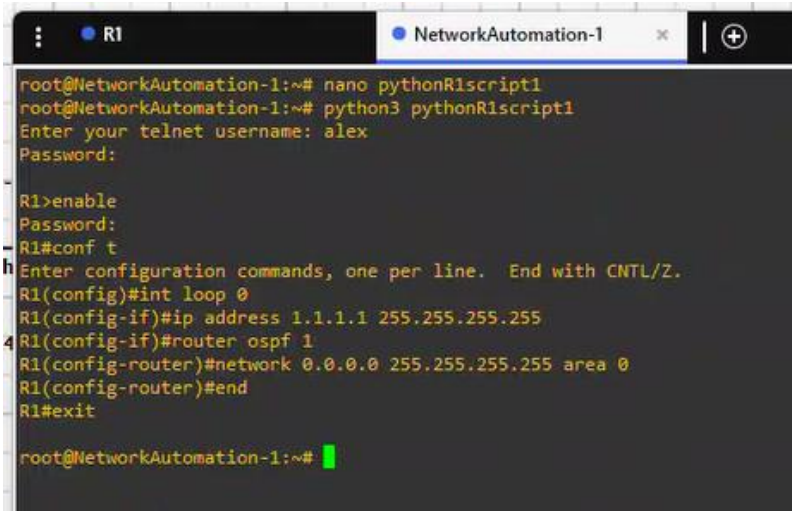
5. Save the file and exit (CTRL + O, then CTRL + X).

7. Execute the Script

1. Run the script:

   ```
   python3 pythonR1script1.py
   ```

2. You should see **output printed in the terminal**.



3. Verify the configuration on the router:

   show ip interface brief
   show run

```
R1#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        unassigned      YES NVRAM  administratively down down
FastEthernet1/0        192.168.102.130 YES DHCP   up                    up
FastEthernet1/1        10.10.10.1      YES NVRAM  up                    up
FastEthernet1/1.1      192.168.1.1     YES NVRAM  up                    up
FastEthernet1/1.10     192.168.10.1    YES NVRAM  up                    up
FastEthernet1/1.20     192.168.20.1    YES NVRAM  up                    up
Serial2/0              unassigned      YES NVRAM  administratively down down
Serial2/1              unassigned      YES NVRAM  administratively down down
Serial2/2              unassigned      YES NVRAM  administratively down down
Serial2/3              unassigned      YES NVRAM  administratively down down
R1#
*Mar 14 17:27:51.939: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback0, changed state to up
R1#
*Mar 14 17:27:52.251: %SYS-5-CONFIG_I: Configured from console by alex on vty0 (192.168.1.100)
R1#show ip interface br
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        unassigned      YES NVRAM  administratively down down
FastEthernet1/0        192.168.102.130 YES DHCP   up                    up
FastEthernet1/1        10.10.10.1      YES NVRAM  up                    up
FastEthernet1/1.1      192.168.1.1     YES NVRAM  up                    up
FastEthernet1/1.10     192.168.10.1    YES NVRAM  up                    up
FastEthernet1/1.20     192.168.20.1    YES NVRAM  up                    up
Serial2/0              unassigned      YES NVRAM  administratively down down
Serial2/1              unassigned      YES NVRAM  administratively down down
Serial2/2              unassigned      YES NVRAM  administratively down down
Serial2/3              unassigned      YES NVRAM  administratively down down
Loopback0              1.1.1.1         YES manual up                    up
R1#
```

```
R1#show running-config
Building configuration...

Current configuration : 1898 bytes
!
! Last configuration change at 17:27:52 UTC Fri Mar 14 2025 by alex
!
version 15.2
service timestamps debug datetime msec
service timestamps log datetime msec
!
hostname R1
!
boot-start-marker
boot-end-marker
!
!
enable password cisco
!
```

```
!
!
interface Loopback0
 ip address 1.1.1.1 255.255.255.255
!
```

```
!
router ospf 1
 network 0.0.0.0 255.255.255.255 area 0
!
```

8. Debugging (Optional)

1. Enable Telnet debugging on **R1**:

   ```
   debug telnet
   ```

2. To disable debugging:

   ```
   no debug telnet
   ```

Conclusion

This method allows you to automate network device configurations using Python. You can build on this script to configure **multiple devices** or add **more complex automation**.

Configure Network Devices Using Python on GNS3 (Part 2) – Configure VLANs on Switches

1. Configure the Switch

1. Open your switch console and enter the following configuration:

   ```
   conf t
   int vlan 1
   ip address 192.168.1.99 255.255.255.0
   no shut
   exit
   hostname S1
   enable password cisco
   username alex password cisco
   line vty 0 4
   login local
   transport input all
   end
   ```

```
SW1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SW1(config)#int vlan 1
SW1(config-if)#ip address 192.168.122.72 255.255.255.0
SW1(config-if)#no shutdown
SW1(config-if)#exit
SW1(config)#hostname S1
S1(config)#enable password cisco
S1(config)#username alex password cisco
S1(config)#line vty 0 4
S1(config-line)#login local
S1(config-line)#transport input all
S1(config-line)#end
S1#
*Mar  1 02:10:59.983: %SYS-5-CONFIG_I: Configured from console by console
S1#
```

```
S1(config)#int vlan 1
S1(config-if)#ip add
S1(config-if)#ip address 192.168.1.99 255.255.255.0
S1(config-if)#
```

2. Test Telnet access:

   telnet 192.168.1.99

3. Enter your credentials:
   - Username: alex
   - Password: cisco
4. Enter privileged EXEC mode:

   enable
   (password: cisco)

5. Exit Telnet:

   exit

```
S1#telnet 192.168.1.99
Trying 192.168.1.99 ... Open


User Access Verification

Username: alex
Password:
S1>en
Password:
S1#exit

[Connection to 192.168.1.99 closed by foreign host]
S1#
```

This confirms that Telnet is working on the switch.

2. Create the Python Script

1. On the **Network Automation** device, create a new text file:

   nano pythonS1script1

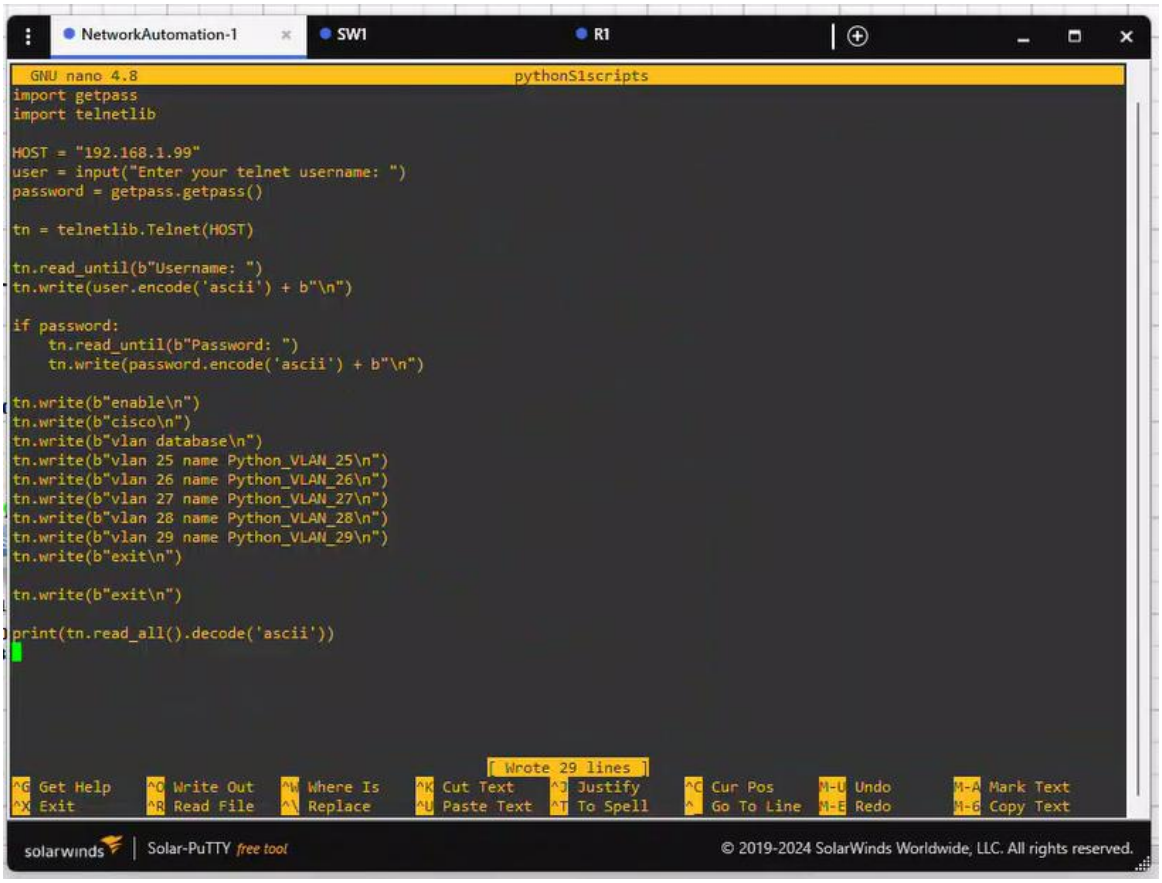
2. Enter the following script:

```
import getpass
import telnetlib


HOST = "192.168.1.99"
user = input("Enter your telnet username: ")
password = getpass.getpass()


tn = telnetlib.Telnet(HOST)


tn.read_until(b"Username: ")
tn.write(user.encode('ascii') + b"\n")


if password:
  tn.read_until(b"Password: ")
  tn.write(password.encode('ascii') + b"\n")


tn.write(b"enable\n")
tn.write(b"cisco\n")
tn.write(b"vlan database\n")
```

```
tn.write(b"vlan 25 name Python_VLAN_25\n")

tn.write(b"vlan 26 name Python_VLAN_26\n")

tn.write(b"vlan 27 name Python_VLAN_27\n")

tn.write(b"vlan 28 name Python_VLAN_28\n")

tn.write(b"vlan 29 name Python_VLAN_29\n")

tn.write(b"exit\n")

tn.write(b"end\n")

tn.write(b"exit\n")


print(tn.read_all().decode('ascii'))
```

3. Save the file and exit:

Press **CTRL + O**, then **Enter** to save.
Press **CTRL + X** to exit.



3. Verify VLANs Before Running the Script

1. On the switch, check the existing VLANs:

   show vlan-switch

```
S1#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/2, Fa1/3, Fa1/4
                                                Fa1/5, Fa1/7, Fa1/8, Fa1/9
                                                Fa1/10, Fa1/12, Fa1/13, Fa1/14
                                                Fa1/15
10   VLAN0010                         active    Fa1/1
20   VLAN0020                         active    Fa1/6
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
10   enet  100010     1500  -      -      -        -    -        0      0
20   enet  100020     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        1      1003
1003 tr    101003     1500  1005   0      -        -    srb      1      1002
1004 fdnet 101004     1500  -      -      1        ibm  -        0      0
1005 trnet 101005     1500  -      -      1        ibm  -        0      0
S1#
```

4. Run the Python Script

1. On the **Network Automation** device, execute the script:

   python3 pythonS1script1

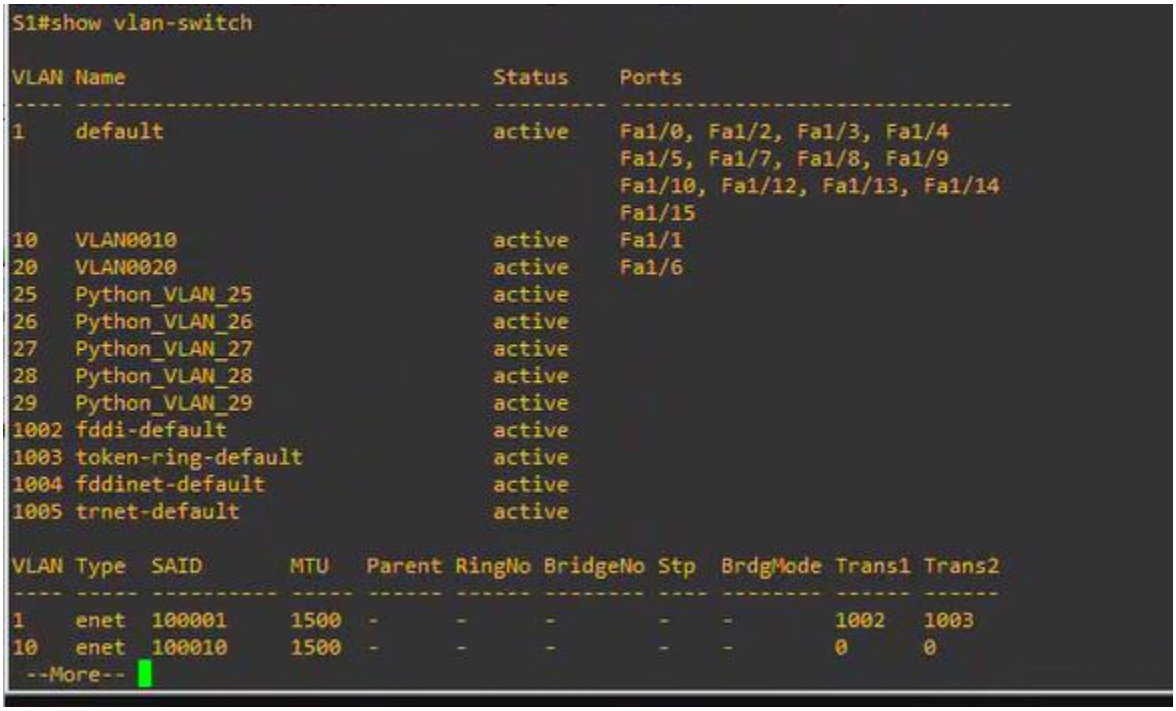
```
NetworkAutomation-1   SW1
root@NetworkAutomation-1:~# nano pythonS1scripts
root@NetworkAutomation-1:~# python3 pythonS1scripts
Enter your telnet username: alex
Password:

S1>enable
Password:
S1#vlan database
S1(vlan)#vlan 25 name Python_VLAN_25
VLAN 25 modified:
    Name: Python_VLAN_25
S1(vlan)#vlan 26 name Python_VLAN_26
VLAN 26 modified:
    Name: Python_VLAN_26
S1(vlan)#vlan 27 name Python_VLAN_27
VLAN 27 modified:
    Name: Python_VLAN_27
S1(vlan)#vlan 28 name Python_VLAN_28
VLAN 28 modified:
    Name: Python_VLAN_28
S1(vlan)#vlan 29 name Python_VLAN_29
VLAN 29 modified:
    Name: Python_VLAN_29
S1(vlan)#exit
APPLY completed.
Exiting....
S1#exit

root@NetworkAutomation-1:~#
```

2. You should see an **output on the terminal** indicating that VLANs were created.

5. Verify VLAN Creation

1. Go to the switch and confirm the VLANs have been created with the correct names:

   show vlan-switch



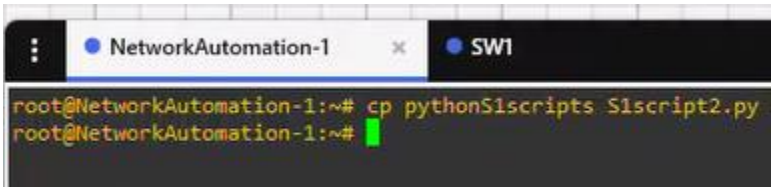
```
S1#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/2, Fa1/3, Fa1/4
                                                Fa1/5, Fa1/7, Fa1/8, Fa1/9
                                                Fa1/10, Fa1/12, Fa1/13, Fa1/14
                                                Fa1/15
10   VLAN0010                         active    Fa1/1
20   VLAN0020                         active    Fa1/6
25   Python_VLAN_25                   active
26   Python_VLAN_26                   active
27   Python_VLAN_27                   active
28   Python_VLAN_28                   active
29   Python_VLAN_29                   active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
10   enet  100010     1500  -      -      -        -    -        0      0
 --More--
```

Configure Network Devices Using Python on GNS3 (Part 3) – Remove password and improve scripts

1. Create a Copy of the Python Script

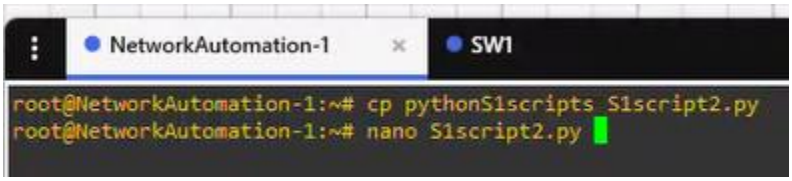
1. On the **Network Automation** device, copy the existing script into a new file:

   cp pythonS1script1 S1script2.py



2. Open the new script file for editing:

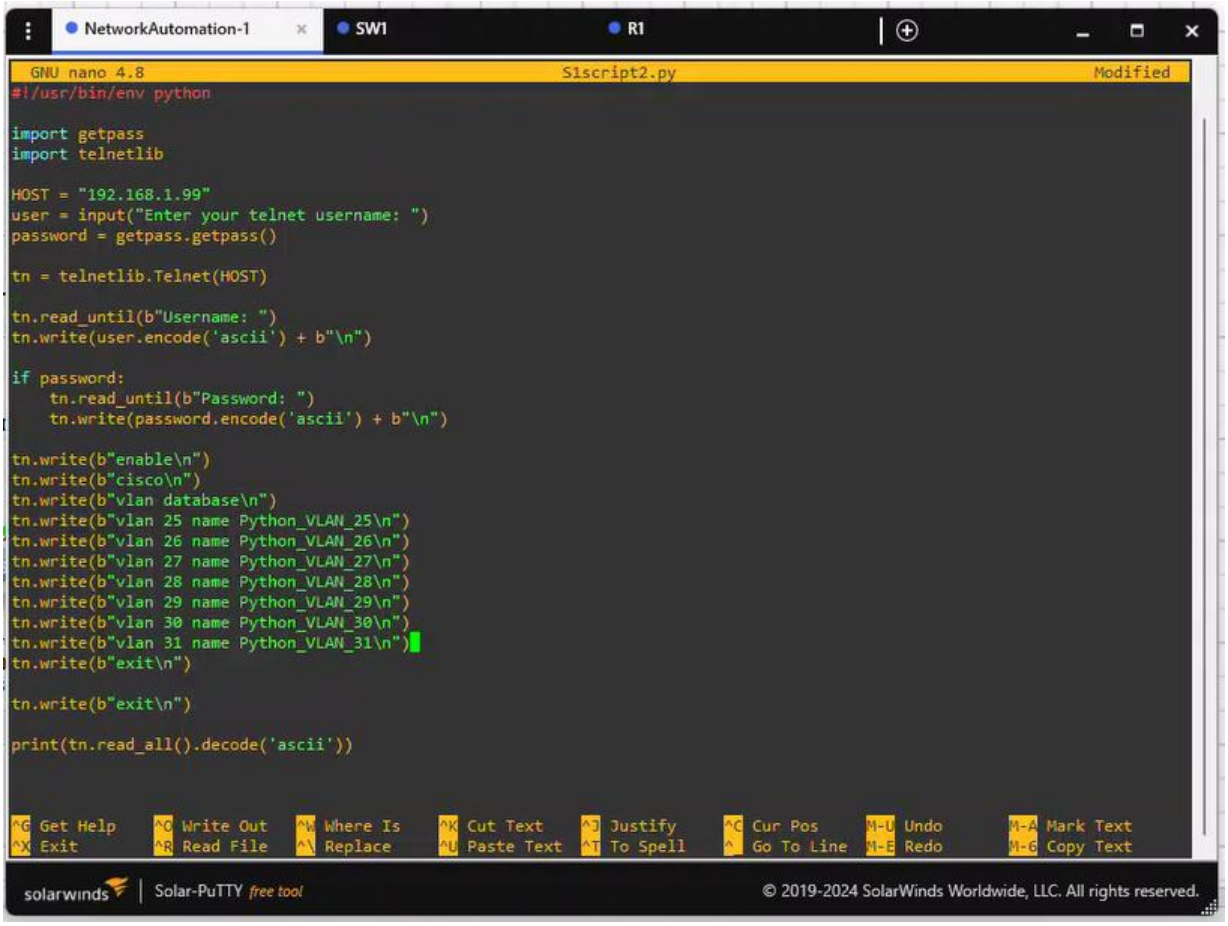
   nano S1script2.py



3. At the very top of the file, add the following **shebang line**:

   #!/usr/bin/env python3

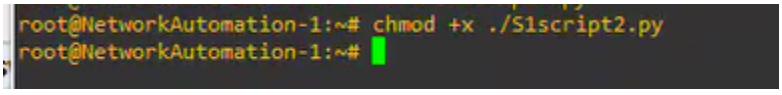
4. Add **a few more VLANs** to differentiate this script from the previous one.
5. Save the file and exit.



2. Make the Script Executable

1. Change the file permissions to allow execution:

   chmod +x ./S1script2.py



2. Now you can run the script **without** specifying python3:

   ./S1script2.py

3. Modify the Script to Remove Password Prompt

1. **Create a backup** before making changes:

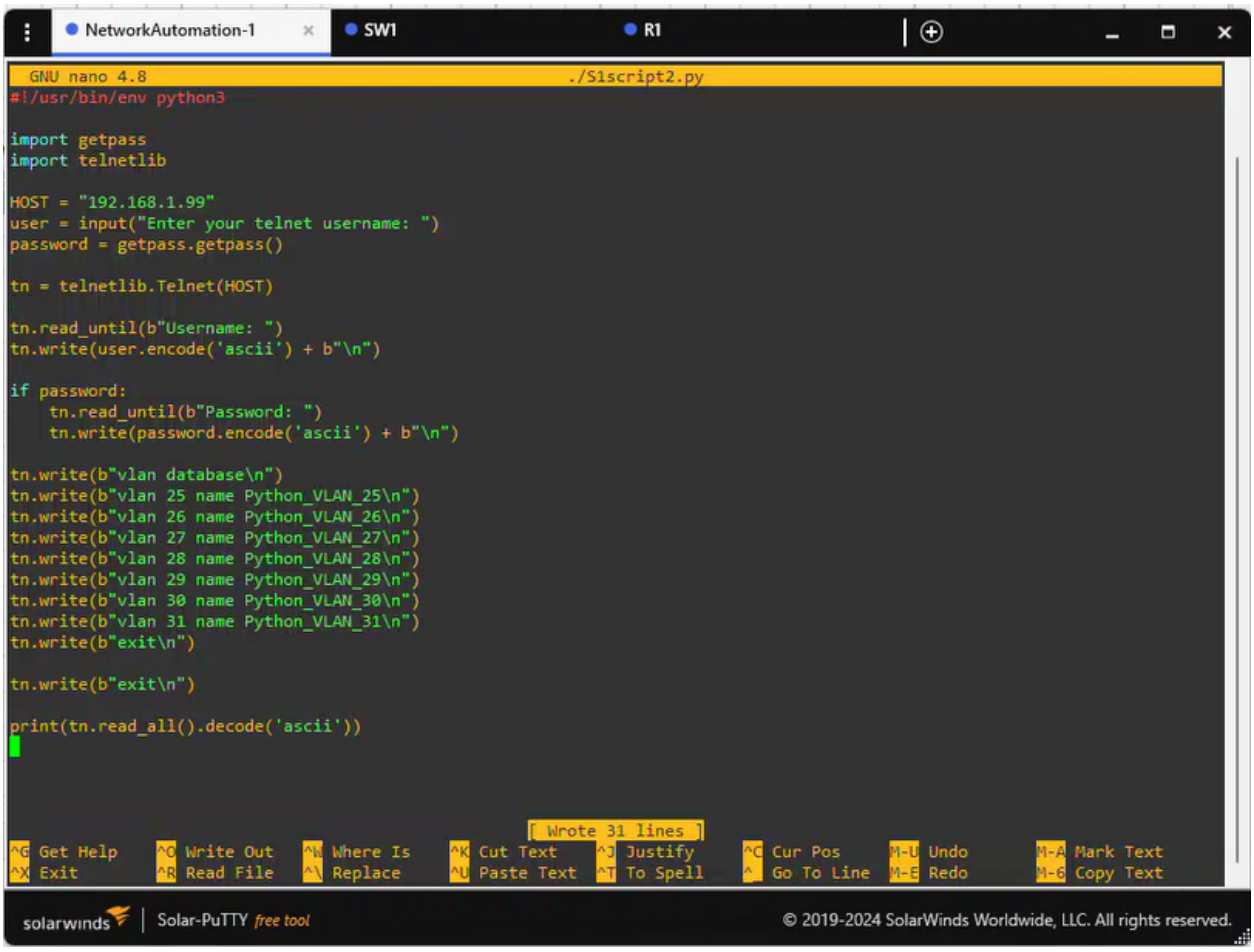
   cp ./S1script2.py ./S1script2.bk

2. Open the script for editing:

   nano S1script2.py

3. **Delete the following two lines**:

   tn.write(b"enable\n")
   tn.write(b"cisco\n")



4. Save the file and exit.
   - This modification removes the requirement to enter the **enable** password.
   - However, this alone will not work—you must adjust the switch configuration.

4. Modify the Switch Configuration

1. Open the switch console.
2. Check the existing user configuration:

   sh run | i user

```
S1#sh run | i user
username alex password 0 cisco
S1#
```


3. Enter global configuration mode and modify the user privileges:

   conf t
   user alex privilege 15
   end

```
S1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
S1(config)#user alex privilege 15
S1(config)#end
S1#
*Mar  1 05:39:27.846: %SYS-5-CONFIG_I: Configured from console by console
S1#
```

5. Test the New Configuration

1. Open a Telnet session to the switch:

   telnet 192.168.1.99

   - You will be **prompted for the line VTY password**, but you will **not need to enter an enable password** to access privileged EXEC mode.

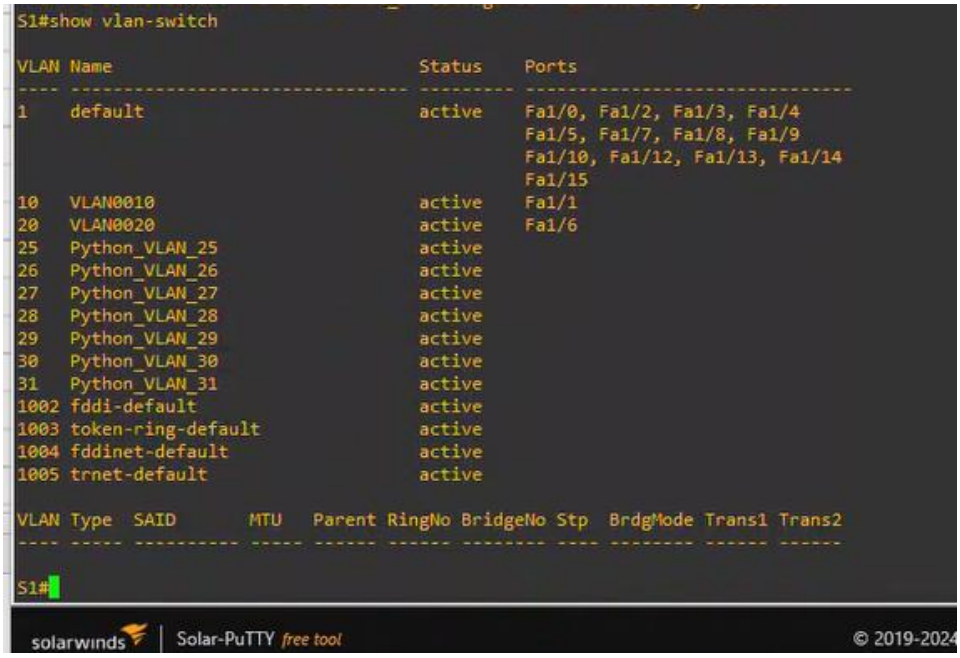
NetworkAutomation-1 | SW1

```
root@NetworkAutomation-1:~# telnet 192.168.1.99
Trying 192.168.1.99...
Connected to 192.168.1.99.
Escape character is '^]'.


User Access Verification

Username: alex
Password:
S1#exit
Connection closed by foreign host.
root@NetworkAutomation-1:~#
```

2. Go back to the **Network Automation** machine and run the modified script:

./S1script2.py

Verify the results on S1 (should go up to vlan 31)



Configure Network Devices Using Python on GNS3 (Part 4) – Create switch VLANs using loops

1. Create a Copy of the Previous Script

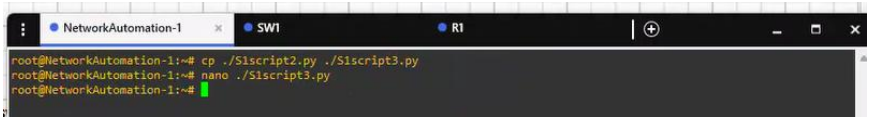
1. On the **Network Automation** device, copy the existing script into a new file:

   cp ./S1script2.py ./S1script3.py

2. Modify the Script to Use a Loop

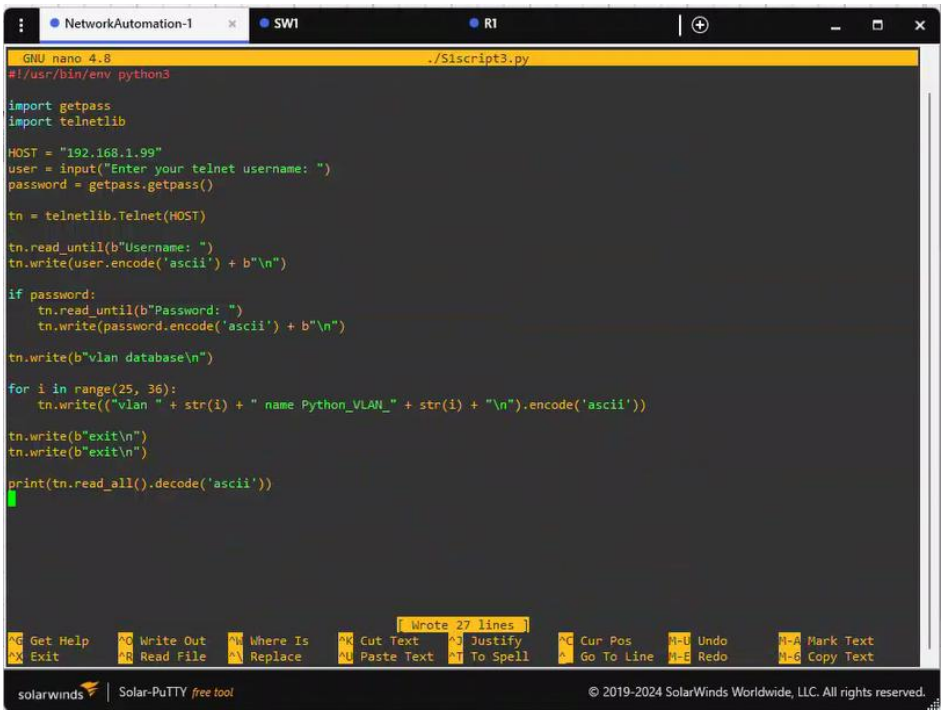
1. Open the new script for editing:

   nano ./S1script3.py



2. Locate the section where VLANs are manually defined and **replace it with the following loop**:

   ```
   for i in range(25, 36):
     tn.write(("vlan " + str(i) + " name Python_VLAN_" + str(i) + "\n").encode('ascii'))
   ```

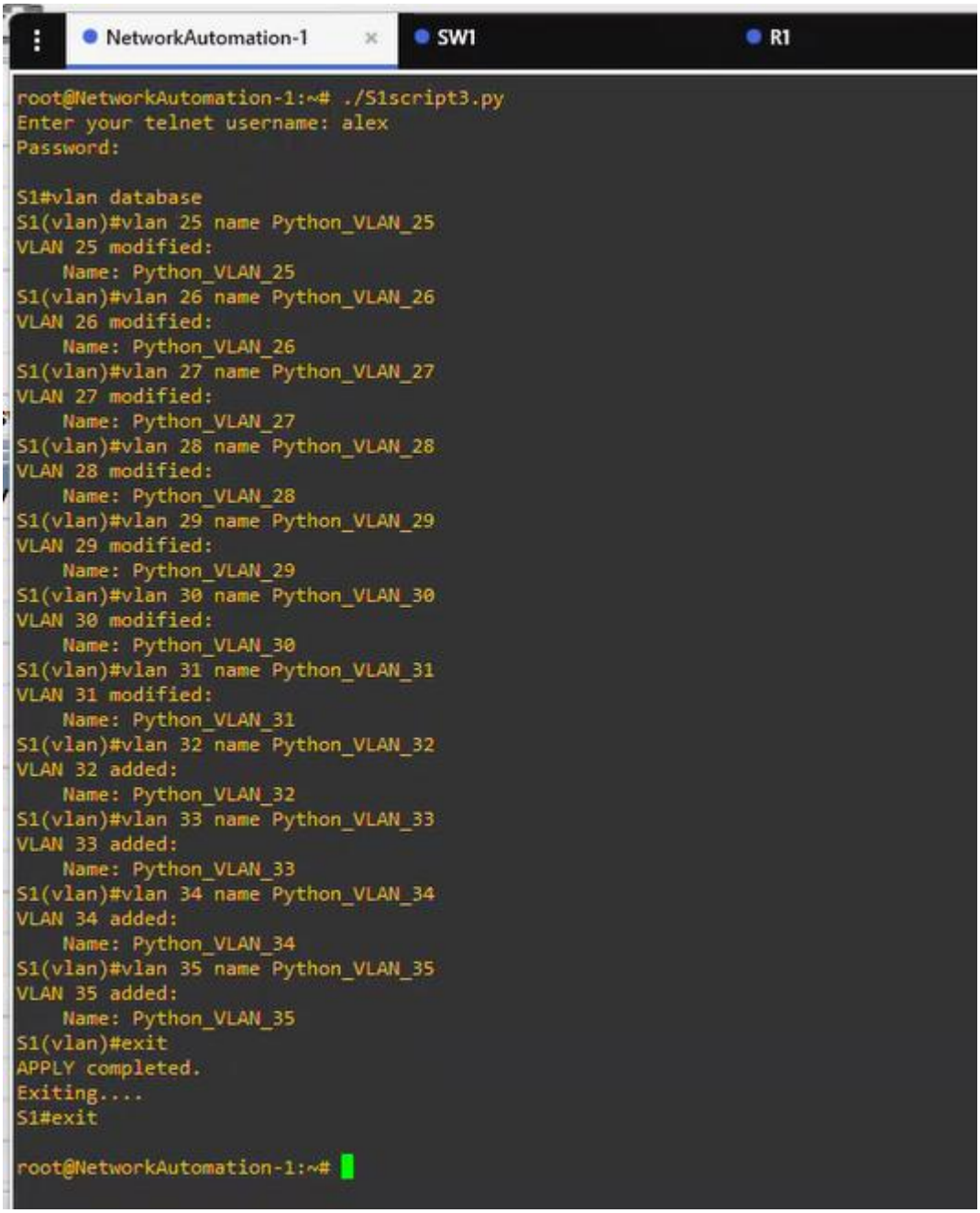


3. Save the file and exit.

3. Run the Script

1. Execute the script to create VLANs **25 to 35**:

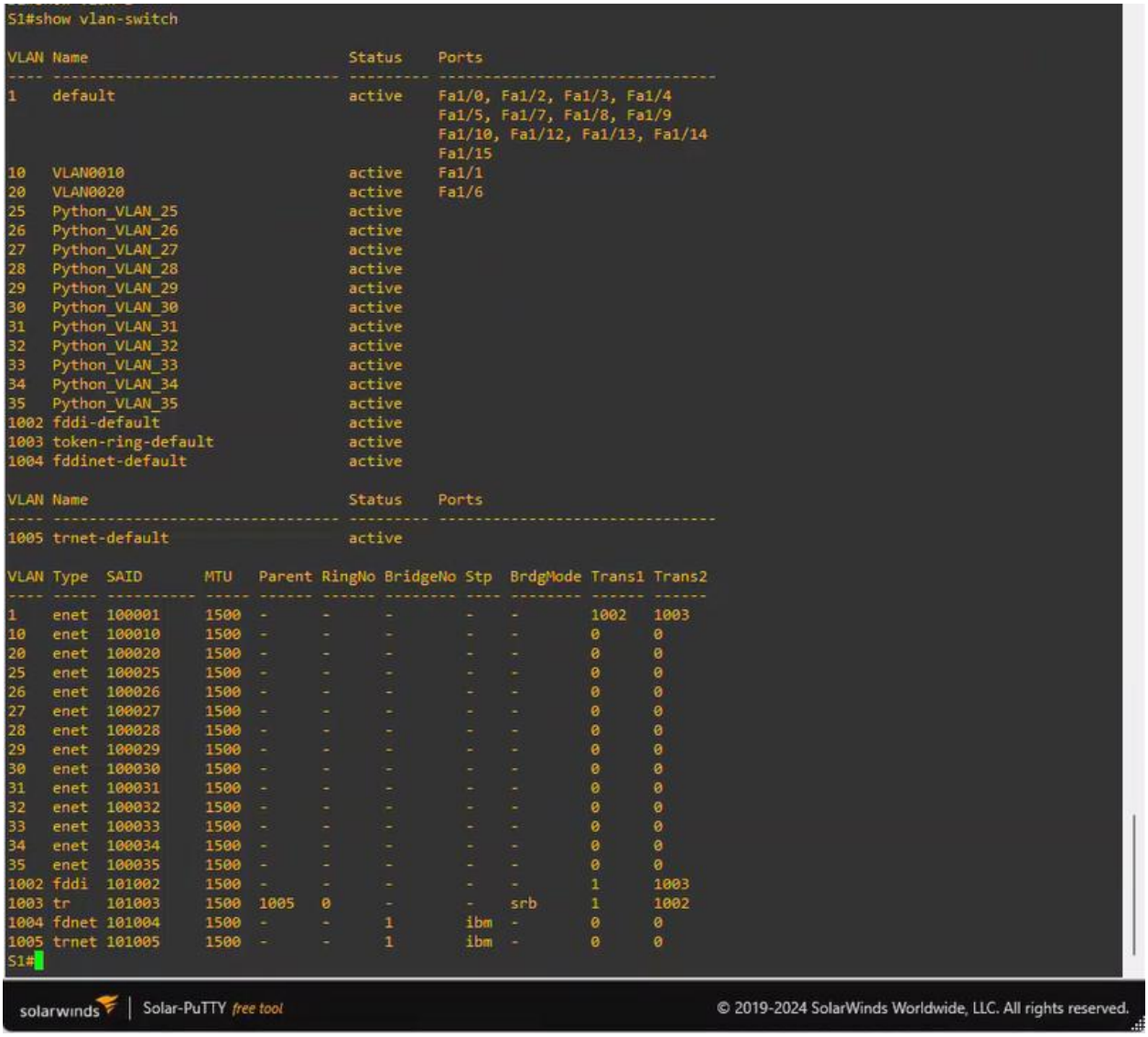
   ./S1script3.py



4. Verify VLAN Creation on the Switch

1. On the switch, check if the VLANs were created successfully:

   show vlan-switch

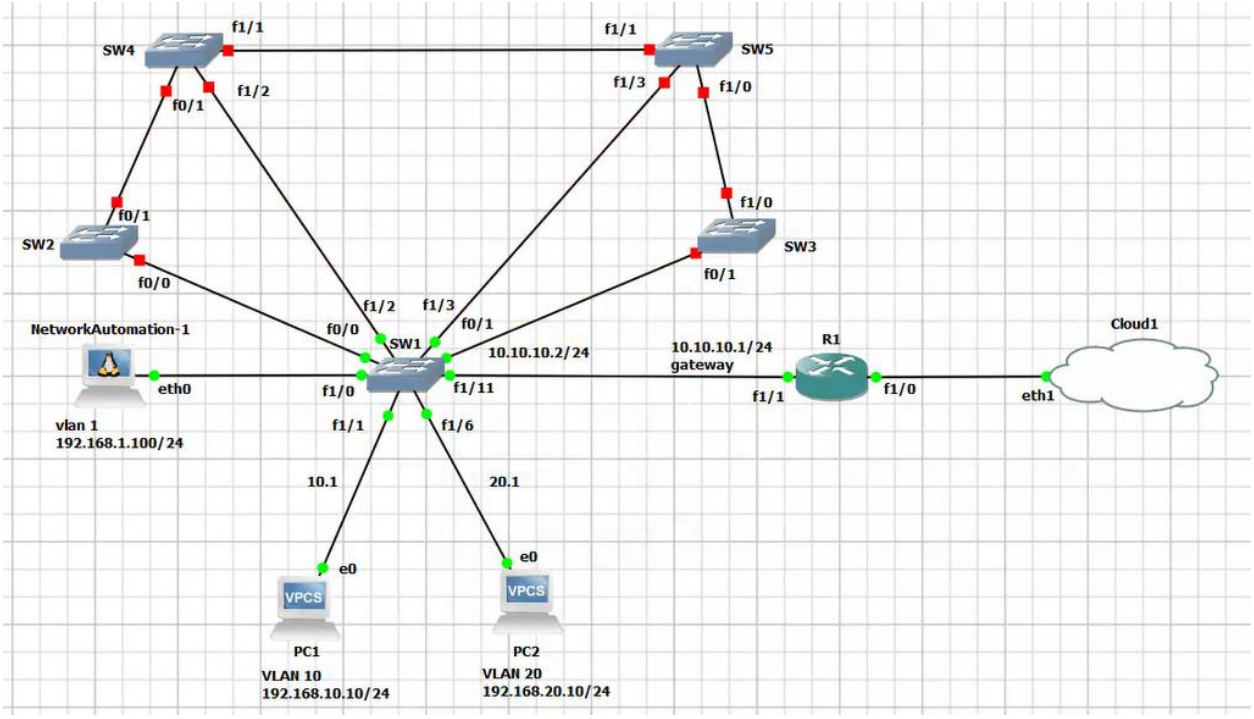


```
S1#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/2, Fa1/3, Fa1/4
                                                Fa1/5, Fa1/7, Fa1/8, Fa1/9
                                                Fa1/10, Fa1/12, Fa1/13, Fa1/14
                                                Fa1/15
10   VLAN0010                         active    Fa1/1
20   VLAN0020                         active    Fa1/6
25   Python_VLAN_25                   active
26   Python_VLAN_26                   active
27   Python_VLAN_27                   active
28   Python_VLAN_28                   active
29   Python_VLAN_29                   active
30   Python_VLAN_30                   active
31   Python_VLAN_31                   active
32   Python_VLAN_32                   active
33   Python_VLAN_33                   active
34   Python_VLAN_34                   active
35   Python_VLAN_35                   active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
10   enet  100010     1500  -      -      -        -    -        0      0
20   enet  100020     1500  -      -      -        -    -        0      0
25   enet  100025     1500  -      -      -        -    -        0      0
26   enet  100026     1500  -      -      -        -    -        0      0
27   enet  100027     1500  -      -      -        -    -        0      0
28   enet  100028     1500  -      -      -        -    -        0      0
29   enet  100029     1500  -      -      -        -    -        0      0
30   enet  100030     1500  -      -      -        -    -        0      0
31   enet  100031     1500  -      -      -        -    -        0      0
32   enet  100032     1500  -      -      -        -    -        0      0
33   enet  100033     1500  -      -      -        -    -        0      0
34   enet  100034     1500  -      -      -        -    -        0      0
35   enet  100035     1500  -      -      -        -    -        0      0
1002 fddi  101002     1500  -      -      -        -    -        1      1003
1003 tr    101003     1500  1005   0      -        -    srb      1      1002
1004 fdnet 101004     1500  -      -      1        ibm  -        0      0
1005 trnet 101005     1500  -      -      1        ibm  -        0      0
S1#
```

Configure Network Devices Using Python on GNS3 (Part 5) - Multiple switches, multiple VLANs

1. Expand the Network

1. In your GNS3 topology, **add 4 new switches** and connect them to **S1**.
    - S1 will now serve as the **management switch**.

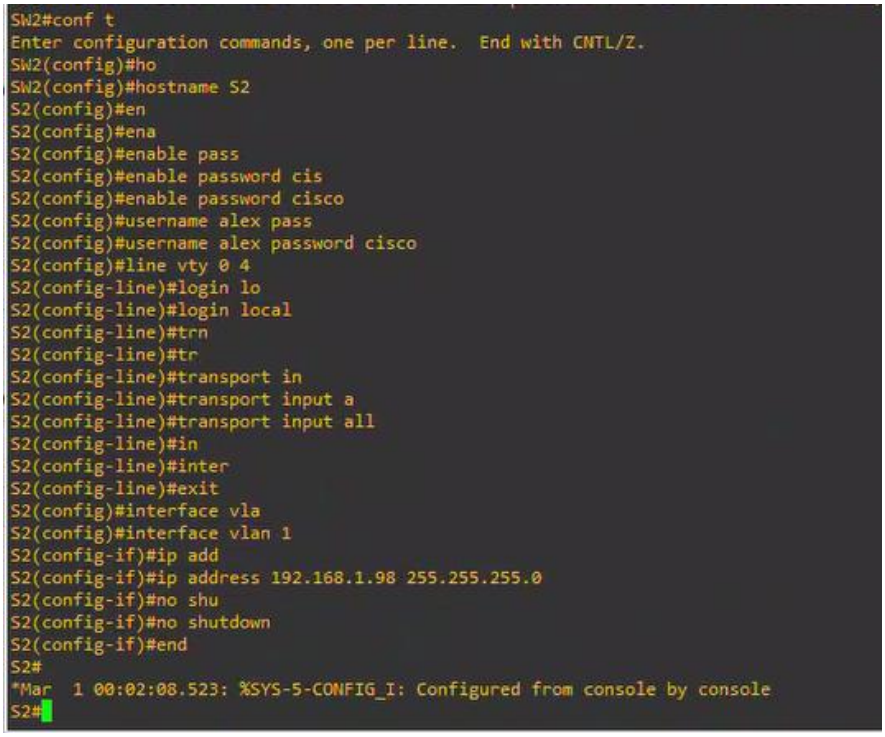


2. Configure the New Switches

S2 Configuration

```
conf t
hostname S2
enable password cisco
username alex password cisco
line vty 0 4
login local
transport input all
interface vlan 1
ip address 192.168.1.98 255.255.255.0
no shutdown
```

exit
interface fa 1/10
switchport mode access
switchport access vlan1
no shutdown
end



Verify Telnet access to **S2**:

telnet 192.168.1.98

S3 Configuration

conf t
hostname S3
enable password cisco
username alex password cisco
line vty 0 4
login local
transport input all
interface vlan 1
ip address 192.168.1.97 255.255.255.0

no shutdown

end

```
SW3#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SW3(config)#f
SW3(config)#int f 1/12
SW3(config-if)#sw
SW3(config-if)#switchport mode a
SW3(config-if)#switchport mode access
SW3(config-if)#sw
SW3(config-if)#switchport ac
SW3(config-if)#switchport access vl
SW3(config-if)#switchport access vlan 1
SW3(config-if)#no sh
SW3(config-if)#no shutdown
SW3(config-if)#exit
SW3(config)#hotstname S3
             ^
% Invalid input detected at '^' marker.

SW3(config)#hotsname S3
             ^
% Invalid input detected at '^' marker.

SW3(config)#ho
SW3(config)#hostname S3
S3(config)#ena
S3(config)#enable pa
S3(config)#enable password cisco
S3(config)#uner
S3(config)#user
S3(config)#username alex pass
S3(config)#username alex password cisco
S3(config)#line vty 0 4
S3(config-line)#login local
S3(config-line)#tr
S3(config-line)#transport in
S3(config-line)#transport input a
S3(config-line)#transport input all
S3(config-line)#exit
S3(config)#us
S3(config)#username alex pr
S3(config)#username alex privilege 15
S3(config)#in
S3(config)#interface vl
S3(config)#interface vlan 1
S3(config-if)#ip add 192.168.1.97 255.255.255.0
S3(config-if)#no sh
S3(config-if)#no shutdown
S3(config-if)#exit
S3(config)#
```

solarwinds | Solar-PuTTY *free tool*

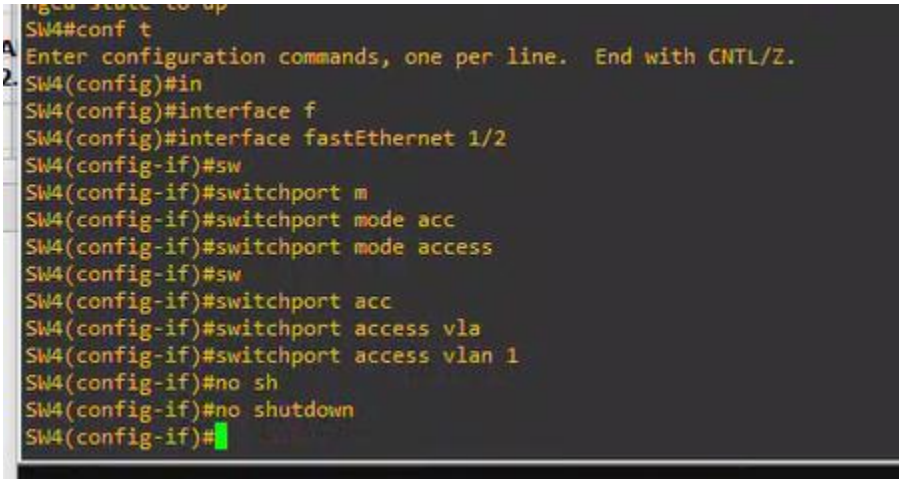
Verify Telnet access to **S3**:

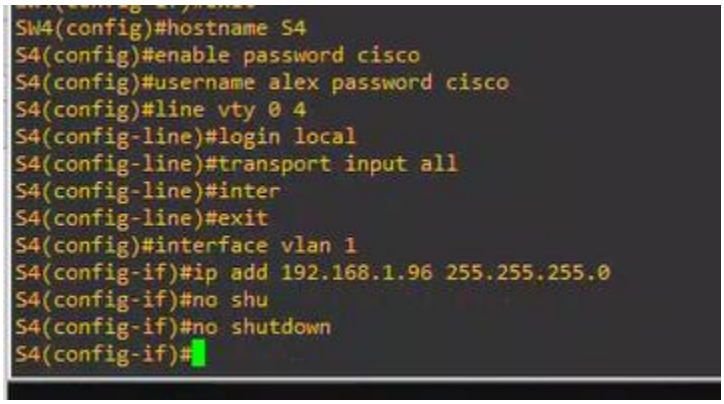
telnet 192.168.1.97

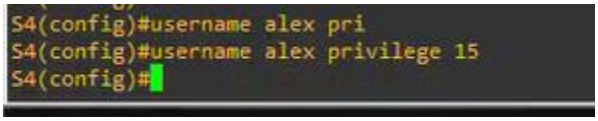
```
S3#conf t
*Mar  1 00:04:24.091: %SYS-5-CONFIG_I: Configured from console by console
S3#telnet 192.168.1.97
Trying 192.168.1.97 ... Open


User Access Verification

Username: alex
Password:
S3#
```

S4 Configuration

conf t
hostname S4
enable password cisco
username alex password cisco
line vty 0 4
login local
transport input all
interface vlan 1
ip address 192.168.1.96 255.255.255.0
no shutdown
end

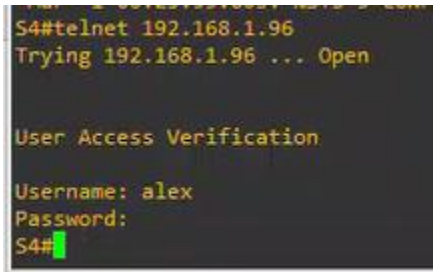






Verify Telnet access to **S4**:

telnet 192.168.1.96



S5 Configuration

conf t
hostname S5
enable password cisco
username alex password cisco
line vty 0 4
login local
transport input all
interface vlan 1
ip address 192.168.1.95 255.255.255.0
no shutdown
end

```
SW5#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
SW5(config)#in
SW5(config)#interface vl
SW5(config)#interface f 1/3
SW5(config-if)#sw
SW5(config-if)#switchport m
SW5(config-if)#switchport mode acc
SW5(config-if)#switchport mode access
SW5(config-if)#sw
SW5(config-if)#switchport ac
SW5(config-if)#switchport access vla
SW5(config-if)#switchport access vlan 1
SW5(config-if)#no shu
SW5(config-if)#no shutdown
SW5(config-if)#exit
SW5(config)#hos
SW5(config)#hostname S5
S5(config)#enable password cisco
S5(config)#username alex password cisco
S5(config)#username alex pr
S5(config)#username alex privilege 15
S5(config)#line vty 0 4
S5(config-line)#login local
S5(config-line)#transport input all
S5(config-line)#exit
S5(config)#interface vlan 1
S5(config-if)#ip add
S5(config-if)#ip address 192.168.1.95 255.255.255.0
S5(config-if)#no shu
S5(config-if)#no shutdown
S5(config-if)#end
S5#
*Mar  1 00:28:00.359: %SYS-5-CONFIG_I: Configured from console by console
S5#
```

Verify Telnet access to **S5**:

telnet 192.168.1.95

```
S5#telnet 192.168.1.95
Trying 192.168.1.95 ... Open


User Access Verification

Username: alex
Password:
S5#
```

---

3. Verify Connectivity

1. From your **Network Automation** device, ping all the new switches:

   ping 192.168.1.98
   ping 192.168.1.97

ping 192.168.1.96
ping 192.168.1.95

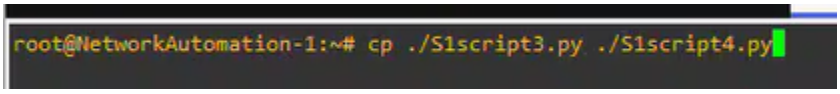
```
root@NetworkAutomation-1:~# ping 192.168.1.98
PING 192.168.1.98 (192.168.1.98) 56(84) bytes of data.
64 bytes from 192.168.1.98: icmp_seq=1 ttl=255 time=10.5 ms
From 192.168.1.96 icmp_seq=1 Redirect Network(New nexthop: 98.1.168.
From 192.168.1.95 icmp_seq=1 Redirect Network(New nexthop: 98.1.168.
64 bytes from 192.168.1.98: icmp_seq=2 ttl=255 time=10.8 ms
64 bytes from 192.168.1.98: icmp_seq=3 ttl=255 time=13.7 ms
^C
--- 192.168.1.98 ping statistics ---
3 packets transmitted, 3 received, +2 errors, 0% packet loss, time 2
rtt min/avg/max/mdev = 10.502/11.648/13.652/1.421 ms
root@NetworkAutomation-1:~# ping 192.168.1.97
PING 192.168.1.97 (192.168.1.97) 56(84) bytes of data.
64 bytes from 192.168.1.97: icmp_seq=1 ttl=255 time=9.53 ms
64 bytes from 192.168.1.97: icmp_seq=2 ttl=255 time=9.46 ms
64 bytes from 192.168.1.97: icmp_seq=3 ttl=255 time=4.29 ms
64 bytes from 192.168.1.97: icmp_seq=4 ttl=255 time=17.8 ms
^C
--- 192.168.1.97 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 4.285/10.264/17.789/4.836 ms
root@NetworkAutomation-1:~# ping 192.168.1.96
PING 192.168.1.96 (192.168.1.96) 56(84) bytes of data.
64 bytes from 192.168.1.96: icmp_seq=1 ttl=255 time=11.3 ms
64 bytes from 192.168.1.96: icmp_seq=2 ttl=255 time=3.23 ms
64 bytes from 192.168.1.96: icmp_seq=3 ttl=255 time=6.94 ms
64 bytes from 192.168.1.96: icmp_seq=4 ttl=255 time=10.1 ms
^C
--- 192.168.1.96 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 3.226/7.888/11.295/3.125 ms
root@NetworkAutomation-1:~# ping 192.168.1.95
PING 192.168.1.95 (192.168.1.95) 56(84) bytes of data.
64 bytes from 192.168.1.95: icmp_seq=1 ttl=255 time=7.57 ms
64 bytes from 192.168.1.95: icmp_seq=2 ttl=255 time=4.40 ms
64 bytes from 192.168.1.95: icmp_seq=3 ttl=255 time=3.43 ms
64 bytes from 192.168.1.95: icmp_seq=4 ttl=255 time=7.29 ms
^C
--- 192.168.1.95 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 3.431/5.672/7.569/1.793 ms
```

```
root@NetworkAutomation-1:~# ping 192.168.1.99
PING 192.168.1.99 (192.168.1.99) 56(84) bytes of data.
64 bytes from 192.168.1.99: icmp_seq=1 ttl=255 time=11.3 ms
64 bytes from 192.168.1.99: icmp_seq=2 ttl=255 time=3.68 ms
64 bytes from 192.168.1.99: icmp_seq=3 ttl=255 time=7.97 ms
64 bytes from 192.168.1.99: icmp_seq=4 ttl=255 time=9.06 ms
^C
--- 192.168.1.99 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 3.676/8.004/11.314/2.774 ms
root@NetworkAutomation-1:~#
```

4. Create a New Python Script to Configure All Switches

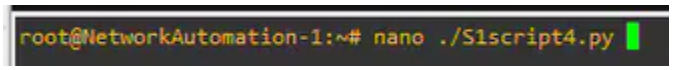
1. Copy your previous script to a new file:

   cp ./S1script3.py ./S1script4.py



2. Open the new script:

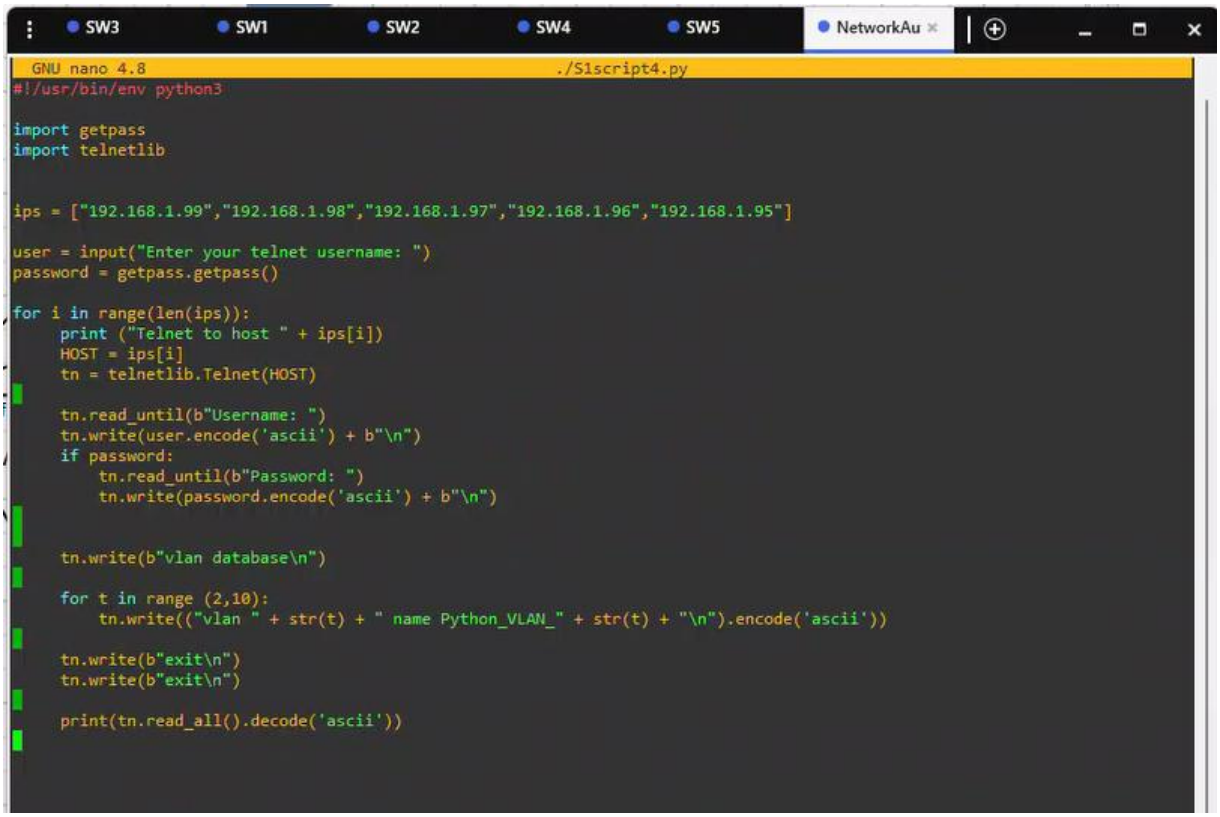
   nano ./S1script4.py



3. Delete the existing contents and enter the following script:

```
#!/usr/bin/env python3


import getpass
import telnetlib


ips = ["192.168.1.99","192.168.1.98","192.168.1.97","192.168.1.96","192.168.1.95"]


user = input("Enter your telnet username: ")
password = getpass.getpass()


for i in range(len(ips)):
    print ("Telnet to host " + ips[i])
    HOST = ips[i]
    tn = telnetlib.Telnet(HOST)


    tn.read_until(b"Username: ")
    tn.write(user.encode('ascii') + b"\n")
```

```
if password:
   tn.read_until(b"Password: ")
   tn.write(password.encode('ascii') + b"\n")


tn.write(b"vlan database\n")


for t in range (2,10):
   tn.write(("vlan " + str(t) + " name Python_VLAN_" + str(t) + "\n").encode('ascii'))


tn.write(b"exit\n")
tn.write(b"exit\n")


print(tn.read_all().decode('ascii'))
```



4. Save the file and exit.

5. Verify VLANs Before Running the Script

1. On each switch, check the current VLANs:

   show vlan-switch

```
SW1#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/2, Fa1/3, Fa1/4
                                                Fa1/5, Fa1/7, Fa1/8, Fa1/9
                                                Fa1/10, Fa1/12, Fa1/13, Fa1/14
                                                Fa1/15
10   VLAN0010                         active    Fa1/1
20   VLAN0020                         active    Fa1/6
25   Python_VLAN_25                   active
26   Python_VLAN_26                   active
27   Python_VLAN_27                   active
28   Python_VLAN_28                   active
29   Python_VLAN_29                   active
30   Python_VLAN_30                   active
31   Python_VLAN_31                   active
32   Python_VLAN_32                   active
33   Python_VLAN_33                   active
34   Python_VLAN_34                   active
35   Python_VLAN_35                   active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
 --More--
```

```
S2#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
1002 fddi  101002     1500  -      -      -        -    -        1      1003
1003 tr    101003     1500  1005   0      -        -    srb      1      1002
1004 fdnet 101004     1500  -      -      1        ibm  -        0      0
1005 trnet 101005     1500  -      -      1        ibm  -        0      0
S2#
```

```
S3#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
1002 fddi  101002     1500  -      -      -        -    -        1      1003
1003 tr    101003     1500  1005   0      -        -    srb      1      1002
1004 fdnet 101004     1500  -      -      1        ibm  -        0      0
1005 trnet 101005     1500  -      -      1        ibm  -        0      0
S3#
```

```
S4#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
1002 fddi  101002     1500  -      -      -        -    -        1      1003
1003 tr    101003     1500  1005   0      -        -    srb      1      1002
1004 fdnet 101004     1500  -      -      1        ibm  -        0      0
1005 trnet 101005     1500  -      -      1        ibm  -        0      0
S4#
```

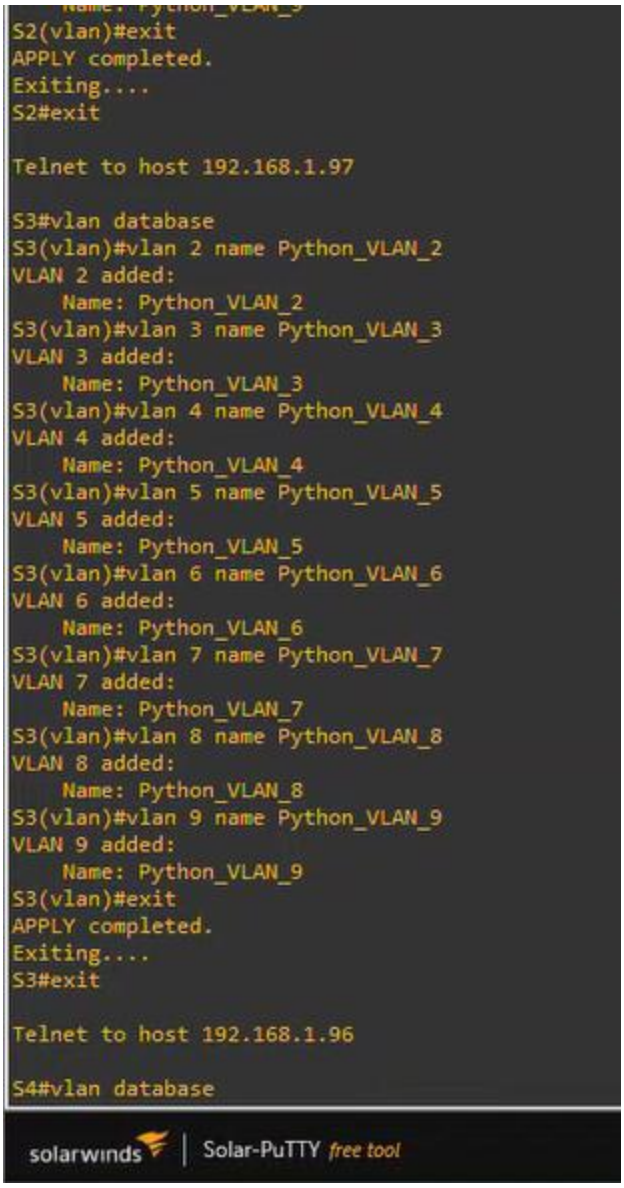
```
S5#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
1002 fddi  101002     1500  -      -      -        -    -        1      1003
1003 tr    101003     1500  1005   0      -        -    srb      1      1002
1004 fdnet 101004     1500  -      -      1        ibm  -        0      0
1005 trnet 101005     1500  -      -      1        ibm  -        0      0
S5#
```

6. Run the Automation Script

1. On your **Network Automation** device, execute the script:

   ./S1script4.py


```
root@NetworkAutomation-1:~# ./S1script4.py
Enter your telnet username: alex
Password:
Telnet to host 192.168.1.99

SW1#vlan database
SW1(vlan)#vlan 2 name Python_VLAN_2
VLAN 2 modified:
    Name: Python_VLAN_2
SW1(vlan)#vlan 3 name Python_VLAN_3
VLAN 3 modified:
    Name: Python_VLAN_3
SW1(vlan)#vlan 4 name Python_VLAN_4
VLAN 4 modified:
    Name: Python_VLAN_4
SW1(vlan)#vlan 5 name Python_VLAN_5
VLAN 5 modified:
    Name: Python_VLAN_5
SW1(vlan)#vlan 6 name Python_VLAN_6
VLAN 6 modified:
    Name: Python_VLAN_6
SW1(vlan)#vlan 7 name Python_VLAN_7
VLAN 7 modified:
    Name: Python_VLAN_7
SW1(vlan)#vlan 8 name Python_VLAN_8
VLAN 8 modified:
    Name: Python_VLAN_8
SW1(vlan)#vlan 9 name Python_VLAN_9
VLAN 9 modified:
    Name: Python_VLAN_9
SW1(vlan)#exit
APPLY completed.
Exiting....
SW1#exit

Telnet to host 192.168.1.98

S2#vlan database
S2(vlan)#vlan 2 name Python_VLAN_2
VLAN 2 added:
    Name: Python_VLAN_2
S2(vlan)#vlan 3 name Python_VLAN_3
VLAN 3 added:
    Name: Python_VLAN_3
S2(vlan)#vlan 4 name Python_VLAN_4
VLAN 4 added:
    Name: Python_VLAN_4
S2(vlan)#vlan 5 name Python_VLAN_5
VLAN 5 added:
    Name: Python_VLAN_5
S2(vlan)#vlan 6 name Python_VLAN_6
VLAN 6 added:
    Name: Python_VLAN_6
S2(vlan)#vlan 7 name Python_VLAN_7
VLAN 7 added:
    Name: Python_VLAN_7
S2(vlan)#vlan 8 name Python_VLAN_8
VLAN 8 added:
    Name: Python_VLAN_8
S2(vlan)#vlan 9 name Python_VLAN_9
VLAN 9 added:
    Name: Python_VLAN_9
S2(vlan)#exit
```


S2(vlan)#exit
APPLY completed.
Exiting....
S2#exit
Telnet to host 192.168.1.97
S3#vlan database
S3(vlan)#vlan 2 name Python_VLAN_2
VLAN 2 added:
Name: Python_VLAN_2
S3(vlan)#vlan 3 name Python_VLAN_3
VLAN 3 added:
Name: Python_VLAN_3
S3(vlan)#vlan 4 name Python_VLAN_4
VLAN 4 added:
Name: Python_VLAN_4
S3(vlan)#vlan 5 name Python_VLAN_5
VLAN 5 added:
Name: Python_VLAN_5
S3(vlan)#vlan 6 name Python_VLAN_6
VLAN 6 added:
Name: Python_VLAN_6
S3(vlan)#vlan 7 name Python_VLAN_7
VLAN 7 added:
Name: Python_VLAN_7
S3(vlan)#vlan 8 name Python_VLAN_8
VLAN 8 added:
Name: Python_VLAN_8
S3(vlan)#vlan 9 name Python_VLAN_9
VLAN 9 added:
Name: Python_VLAN_9
S3(vlan)#exit
APPLY completed.
Exiting....
S3#exit
Telnet to host 192.168.1.96
S4#vlan database
solarwinds
Solar-PuTTY free tool

2. Check each switch to confirm the VLANs were created:

   show vlan-switch

```
SW1#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/2, Fa1/3, Fa1/4
                                                Fa1/5, Fa1/7, Fa1/8, Fa1/9
                                                Fa1/10, Fa1/12, Fa1/13, Fa1/14
                                                Fa1/15
2    Python_VLAN_2                    active
3    Python_VLAN_3                    active
4    Python_VLAN_4                    active
5    Python_VLAN_5                    active
6    Python_VLAN_6                    active
7    Python_VLAN_7                    active
8    Python_VLAN_8                    active
9    Python_VLAN_9                    active
10   VLAN0010                         active    Fa1/1
20   VLAN0020                         active    Fa1/6
25   Python_VLAN_25                   active
26   Python_VLAN_26                   active
27   Python_VLAN_27                   active
28   Python_VLAN_28                   active
29   Python_VLAN_29                   active
30   Python_VLAN_30                   active
 --More--
```

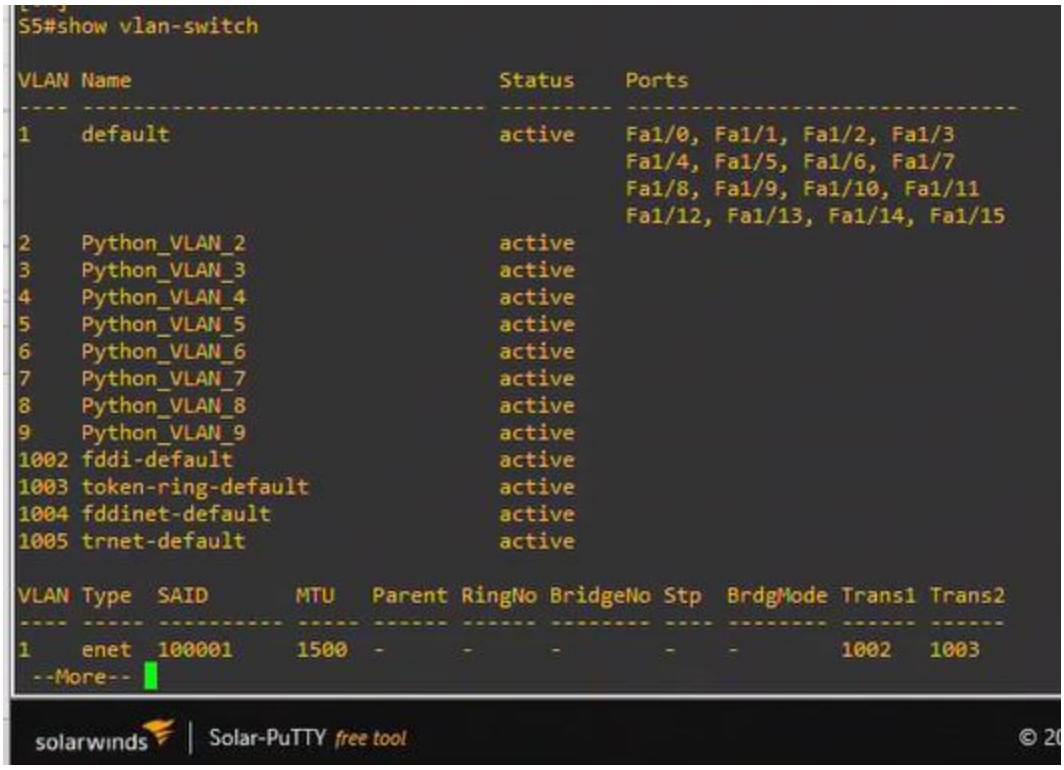
```
S2#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
2    Python_VLAN_2                    active
3    Python_VLAN_3                    active
4    Python_VLAN_4                    active
5    Python_VLAN_5                    active
6    Python_VLAN_6                    active
7    Python_VLAN_7                    active
8    Python_VLAN_8                    active
9    Python_VLAN_9                    active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
 --More--
```

```
S3#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
2    Python_VLAN_2                    active
3    Python_VLAN_3                    active
4    Python_VLAN_4                    active
5    Python_VLAN_5                    active
6    Python_VLAN_6                    active
7    Python_VLAN_7                    active
8    Python_VLAN_8                    active
9    Python_VLAN_9                    active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
 --More--
```

solarwinds | Solar-PuTTY *free tool* ©

```
S4#show vlan-switch

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0, Fa1/1, Fa1/2, Fa1/3
                                                Fa1/4, Fa1/5, Fa1/6, Fa1/7
                                                Fa1/8, Fa1/9, Fa1/10, Fa1/11
                                                Fa1/12, Fa1/13, Fa1/14, Fa1/15
2    Python_VLAN_2                    active
3    Python_VLAN_3                    active
4    Python_VLAN_4                    active
5    Python_VLAN_5                    active
6    Python_VLAN_6                    active
7    Python_VLAN_7                    active
8    Python_VLAN_8                    active
9    Python_VLAN_9                    active
1002 fddi-default                     active
1003 token-ring-default               active
1004 fddinet-default                  active
1005 trnet-default                    active

VLAN Type  SAID       MTU   Parent RingNo BridgeNo Stp  BrdgMode Trans1 Trans2
---- ----- ---------- ----- ------ ------ -------- ---- -------- ------ ------
1    enet  100001     1500  -      -      -        -    -        1002   1003
 --More--
```

solarwinds | Solar-PuTTY *free tool* © 2019-



Conclusion

You have now **automated VLAN configuration across multiple switches**.

- You can **modify the script** to add more VLANs.
- Running the script will apply changes to **all connected switches simultaneously**.

Congratulations! Welcome to the world of network automation! 🎉