Project Title: API Testing on Cisco IOS XE Sandbox using cat8kv using Postman, cURL script.

Description: This project automates network device monitoring and configuration using RESTCONF APIs on Cisco IOS XE on cat8kv. Here Postman is used to perform the CRUD operations. A bash script using cURL tool is used to fetch system health or operational data and other configurational data. The goal is to demonstrate practical network programmability using Restconf APIs and the pros of network automation in real life environment.

Tools and Technologies used:

1.Restconf API

2.Linux Scripting using cURL

3. Openconnect for VPN connection.

4.Postman

5.Cisco Yangsuite for analyzing yang modules

7.SSH for remote login and manual verification

8.Cisco IOS XE on cat8kv sandbox

9.JSON data format

Yang Modules Used:

1.. Cisco-IOS-XE-processes-cpu-oper

2.ietf-interfaces

3.Cisco-IOS-XE-interfaces-oper

Project Blueprint :

```
       Client(Postman , Linux Script)
             API endpoint(Client)
                         |
                         |
            RESTCONF API CALL
                         |
                         |
            API endpoint(Device)
 Network Devce (Cisco IOS XE on Cat8kv)
```

1. The user initiates a Postman API call or run the Linux script
2. The client sends RESTCONF API call via HTTP
3. The Device checks for user authorization.
4. If the user is authorized, the device responds with valid outputs.
5. If the user is not authorized, the device ignores the call.

Project Workflow :

1. Reserve Cisco IOS XE on Cat8kv sanbox available on cisco sandbox .
2. After the environment is live , its time to create the VPN tunnel .
3. For VPN connection , openconnect tool is used .
4. After the VPN connection is set up , we need to setup our lab environment .
5. Login to Cisco Yangsuite , import the device and all its supported YANG modules to take reference .
6. Setup Cisco Yangsuite , by enabling Netconf , Restconf and SSH on the device .
7. Load all the required Yang Modules thats going to be used , to analyse the modules .
8. In protocol section , under the Restconf section , load all the Yang modules and generate the API calls as per requirements .
9. Create the required Postman environment and variables.
10. Use base64 authorization in Postman , as base64 is only supported in cisco sandbox .
11. Take reference of all the required API call , from Yangsuite and use it in Postman.
12. After the calls are successfully made , we can save the response to a file .
13. Now to fetch data automatically we will create a Linux script using cURL.
14. For the URL we will again take reference from Yangsuite.
15. We can create our script as per our business requirements.

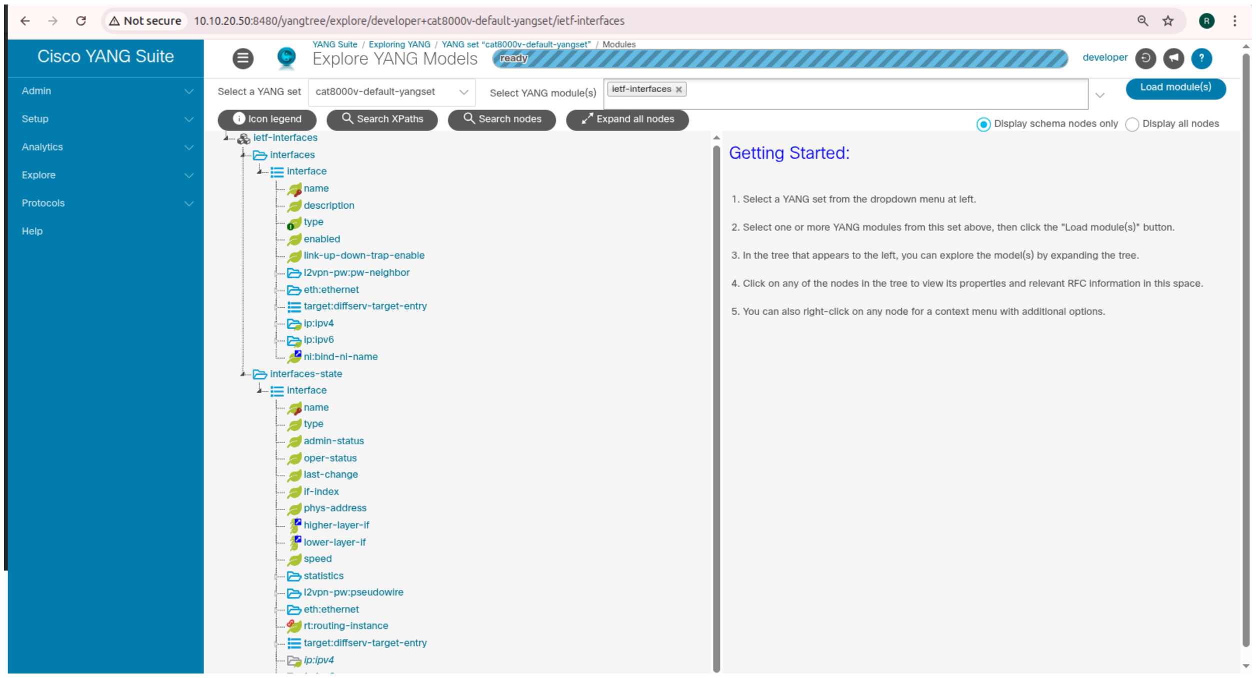
Folder structure (GitHub) :

```
-Automation Projects
        |_Automation Project 1
                        |_ Linux Script
                                        |_automation.sh
                                        |_demo_output_files_1
                                        |_demo_output_file_2
                                        |_demo_output_file_3
                        |_Screenshots
                                        |_ All the project insights
                        |_Video
                                        |_MP4 Video
                                        |_VideoNote.txt
```

|_Note.txt
|_README.docs

Project Insights :

1. Loading Yang models in Yangsuite



Not secure 10.10.20.50:8480/yangtree/explore/developer+cat8000v-default-yangset/ietf-interfaces

Select a YANG set: cat8000v-default-yangset

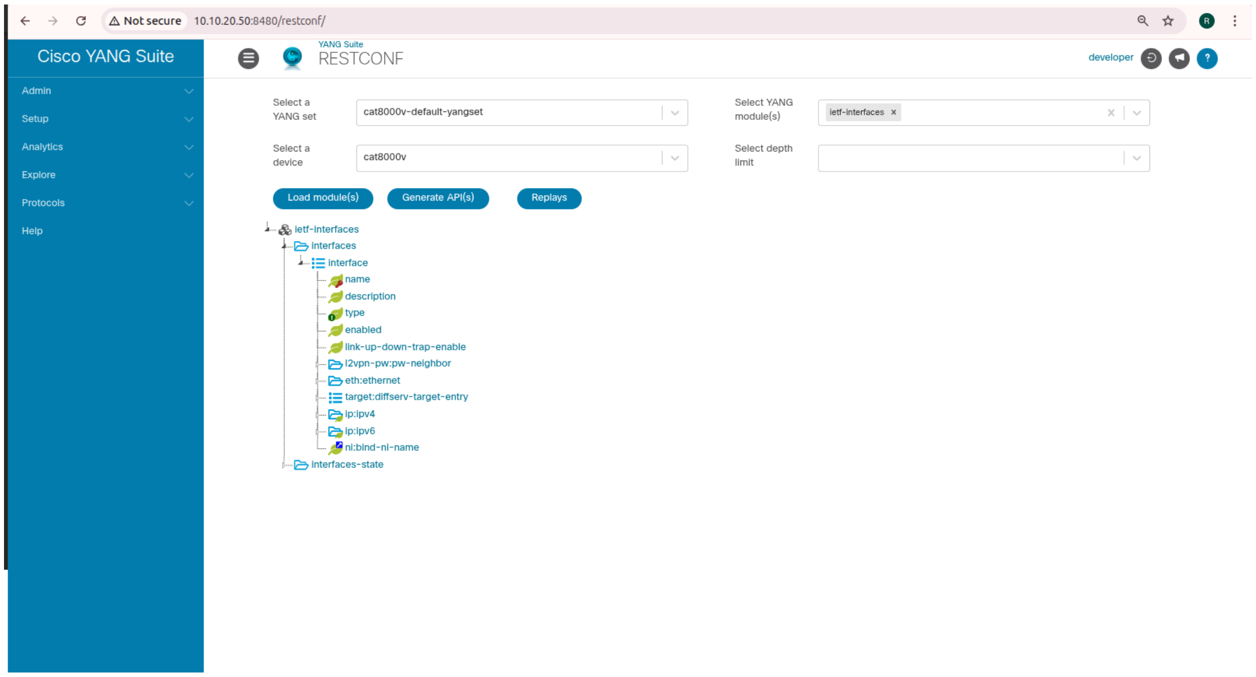
Select YANG module(s): ietf-interfaces

[✓ Display schema nodes only] Display all nodes

Getting Started:

1. Select a YANG set from the dropdown menu at left.
2. Select one or more YANG modules from this set above, then click the "Load module(s)" button.
3. In the tree that appears to the left, you can explore the model(s) by expanding the tree.
4. Click on any of the nodes in the tree to view its properties and relevant RFC information in this space.
5. You can also right-click on any node for a context menu with additional options.

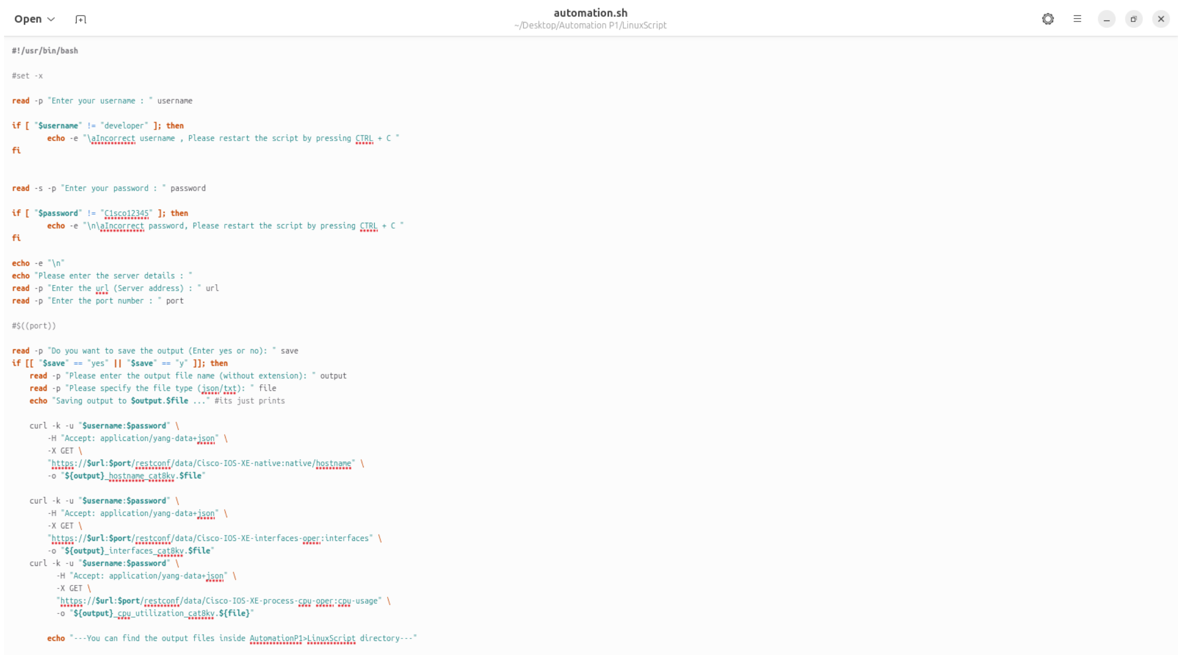
2.Loading yang modules to generate API calls



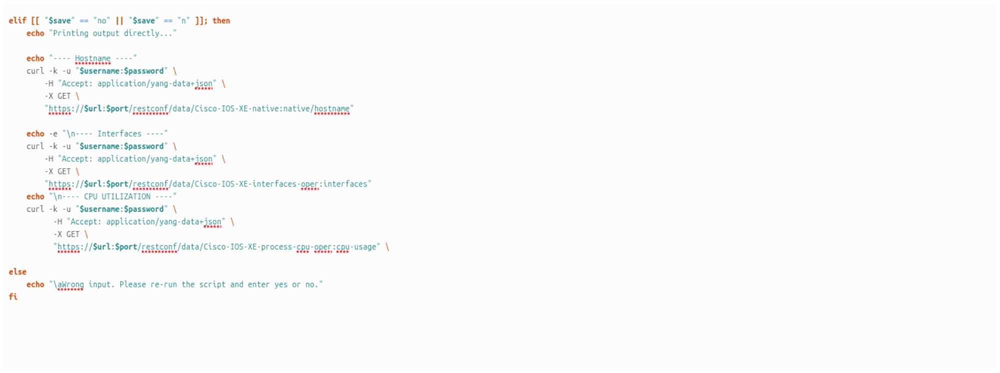
Select a YANG set: cat8000v-default-yangset
Select YANG module(s): ietf-interfaces
Select a device: cat8000v
Select depth limit:

```
ietf-interfaces
  interfaces
    interface
      name
      description
      type
      enabled
      link-up-down-trap-enable
      l2vpn-pw:pw-neighbor
      eth:ethernet
      target:diffserv-target-entry
      ip:ipv4
      ip:ipv6
      ni:bind-ni-name
  interfaces-state
```

3. Create the Linux Script

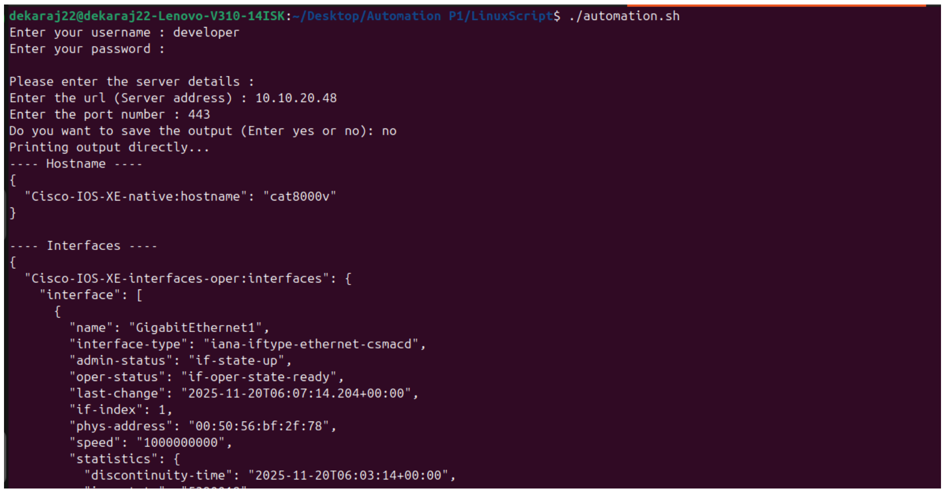


```
#!/usr/bin/bash

#set -x

read -p "Enter your username : " username

if [ "$username" != "developer" ]; then
        echo -e "\aIncorrect username , Please restart the script by pressing CTRL + C "
fi


read -s -p "Enter your password : " password

if [ "$password" != "C1sco12345" ]; then
        echo -e "\n\aIncorrect password, Please restart the script by pressing CTRL + C "
fi

echo -e "\n"
echo "Please enter the server details : "
read -p "Enter the url (Server address) : " url
read -p "Enter the port number : " port

#$((port))

read -p "Do you want to save the output (Enter yes or no): " save
if [[ "$save" == "yes" || "$save" == "y" ]]; then
    read -p "Please enter the output file name (without extension): " output
    read -p "Please specify the file type (json/txt): " file
    echo "Saving output to $output.$file ..." #its just prints

    curl -k -u "$username:$password" \
        -H "Accept: application/yang-data+json" \
        -X GET \
        "https://$url:$port/restconf/data/Cisco-IOS-XE-native:native/hostname" \
        -o "${output}_hostname_cat8kv.$file"

    curl -k -u "$username:$password" \
        -H "Accept: application/yang-data+json" \
        -X GET \
        "https://$url:$port/restconf/data/Cisco-IOS-XE-interfaces-oper:interfaces" \
        -o "${output}_interfaces_cat8kv.$file"
    curl -k -u "$username:$password" \
         -H "Accept: application/yang-data+json" \
         -X GET \
         "https://$url:$port/restconf/data/Cisco-IOS-XE-process-cpu-oper:cpu-usage" \
         -o "${output}_cpu_utilization_cat8kv.${file}"

        echo "---You can find the output files inside AutomationP1>LinuxScript directory---"
```



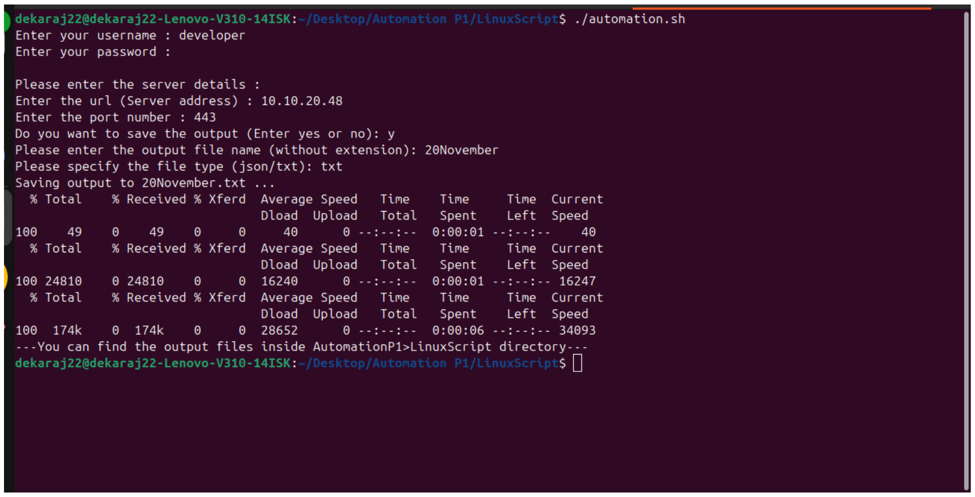
```
elif [[ "$save" == "no" || "$save" == "n" ]]; then
    echo "Printing output directly..."

    echo "---- Hostname ----"
    curl -k -u "$username:$password" \
        -H "Accept: application/yang-data+json" \
        -X GET \
        "https://$url:$port/restconf/data/Cisco-IOS-XE-native:native/hostname"

    echo -e "\n---- Interfaces ----"
    curl -k -u "$username:$password" \
        -H "Accept: application/yang-data+json" \
        -X GET \
        "https://$url:$port/restconf/data/Cisco-IOS-XE-interfaces-oper:interfaces"
    echo "\n---- CPU UTILIZATION ----"
    curl -k -u "$username:$password" \
          -H "Accept: application/yang-data+json" \
          -X GET \
          "https://$url:$port/restconf/data/Cisco-IOS-XE-process-cpu-oper:cpu-usage" \

else
    echo "\aWrong input. Please re-run the script and enter yes or no."
fi
```

4. Script Output (When saved to an external file)



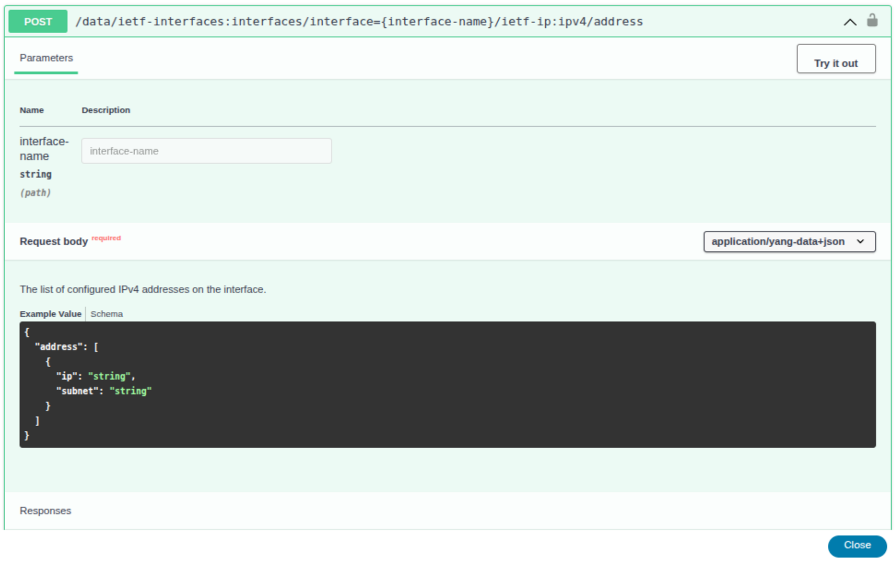
```
dekaraj22@dekaraj22-Lenovo-V310-14ISK:~/Desktop/Automation P1/LinuxScript$ ./automation.sh
Enter your username : developer
Enter your password :

Please enter the server details :
Enter the url (Server address) : 10.10.20.48
Enter the port number : 443
Do you want to save the output (Enter yes or no): no
Printing output directly...
---- Hostname ----
{
  "Cisco-IOS-XE-native:hostname": "cat8000v"
}

---- Interfaces ----
{
  "Cisco-IOS-XE-interfaces-oper:interfaces": {
    "interface": [
      {
        "name": "GigabitEthernet1",
        "interface-type": "iana-iftype-ethernet-csmacd",
        "admin-status": "if-state-up",
        "oper-status": "if-oper-state-ready",
        "last-change": "2025-11-20T06:07:14.204+00:00",
        "if-index": 1,
        "phys-address": "00:50:56:bf:2f:78",
        "speed": "1000000000",
        "statistics": {
          "discontinuity-time": "2025-11-20T06:03:14+00:00",
```

5. Script output (when not saved to an external file)



```
dekaraj22@dekaraj22-Lenovo-V310-14ISK:~/Desktop/Automation P1/LinuxScript$ ./automation.sh
Enter your username : developer
Enter your password :

Please enter the server details :
Enter the url (Server address) : 10.10.20.48
Enter the port number : 443
Do you want to save the output (Enter yes or no): y
Please enter the output file name (without extension): 20November
Please specify the file type (json/txt): txt
Saving output to 20November.txt ...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    49    0    49    0     0     40      0 --:--:--  0:00:01 --:--:--    40
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 24810    0 24810    0     0  16240      0 --:--:--  0:00:01 --:--:-- 16247
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  174k    0  174k    0     0  28652      0 --:--:--  0:00:06 --:--:-- 34093
---You can find the output files inside AutomationP1>LinuxScript directory---
dekaraj22@dekaraj22-Lenovo-V310-14ISK:~/Desktop/Automation P1/LinuxScript$
```

6. Reference and content format from Yangsuite :



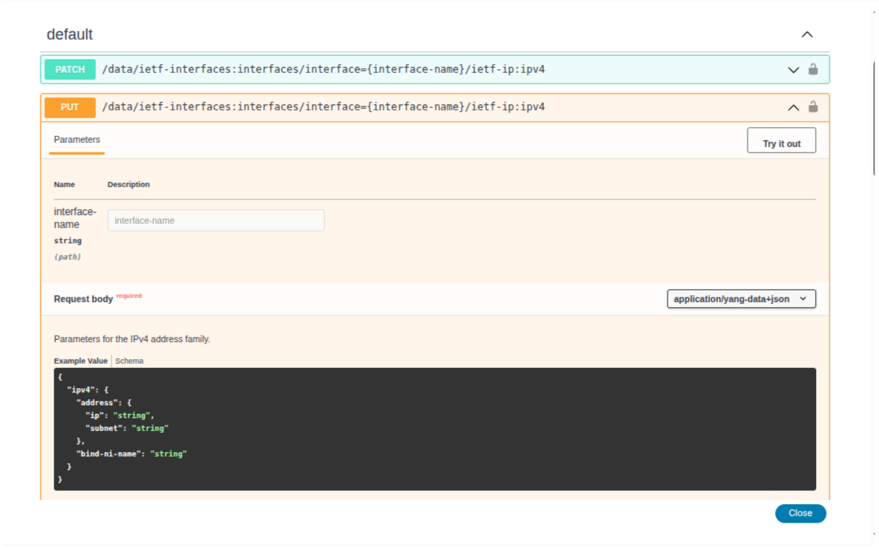
POST `/data/ietf-interfaces:interfaces/interface={interface-name}/ietf-ip:ipv4/address`

interface-name
string
(path)

Request body required
application/yang-data+json

The list of configured IPv4 addresses on the interface.

```
{
  "address": [
    {
      "ip": "string",
      "subnet": "string"
    }
  ]
}
```



PATCH `/data/ietf-interfaces:interfaces/interface={interface-name}/ietf-ip:ipv4`

PUT `/data/ietf-interfaces:interfaces/interface={interface-name}/ietf-ip:ipv4`

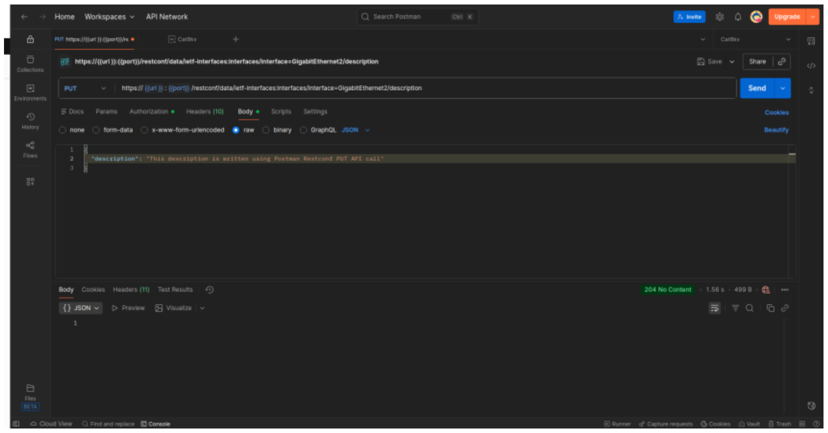
interface-name
string
(path)

Request body required
application/yang-data+json

Parameters for the IPv4 address family.

```
{
  "ipv4": {
    "address": {
      "ip": "string",
      "subnet": "string"
    },
    "bind-ni-name": "string"
  }
}
```

7. POSTMAN PUT CALL

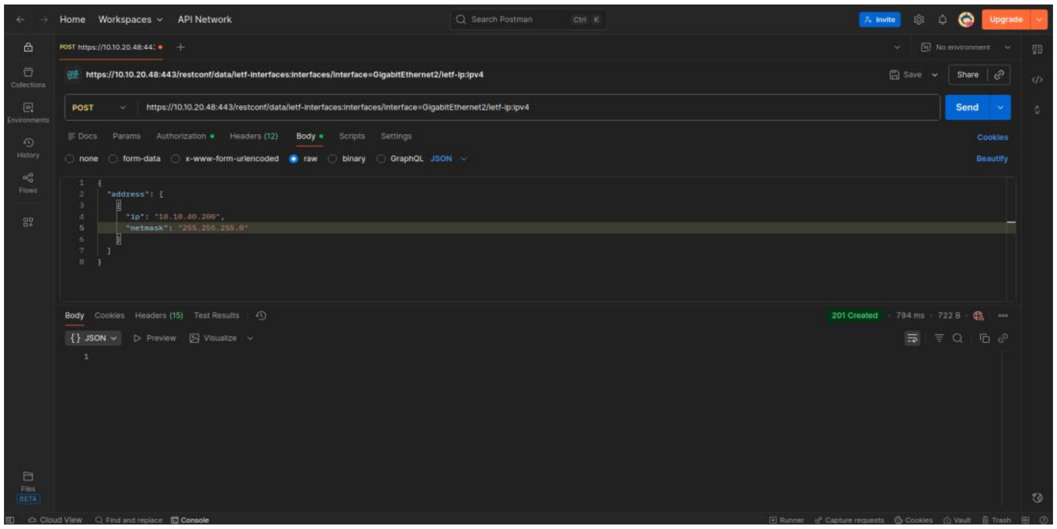


```
PUT https://{{url}}:{{port}}/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/description
```

```
{
"description": "This description is written using Postman Restconf PUT API call"
}
```

```
204 No Content · 1.56 s · 499 B
```

8.POSTMAN POST CALL

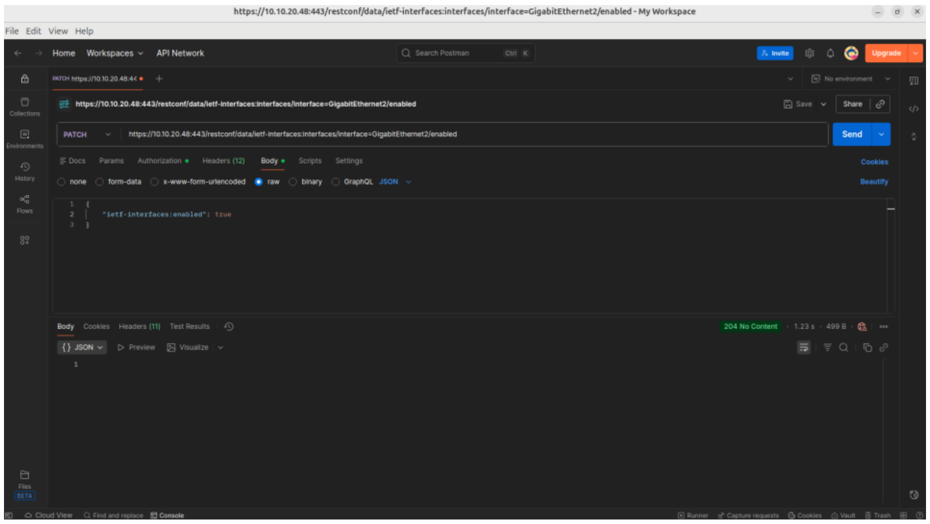


```
POST https://10.10.20.48:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/ietf-ip:ipv4
```

```
{
  "address": [
    {
      "ip": "10.10.40.200",
      "netmask": "255.255.255.0"
    }
  ]
}
```

```
201 Created · 794 ms · 722 B
```

9. POSTMAN PATCH CALL



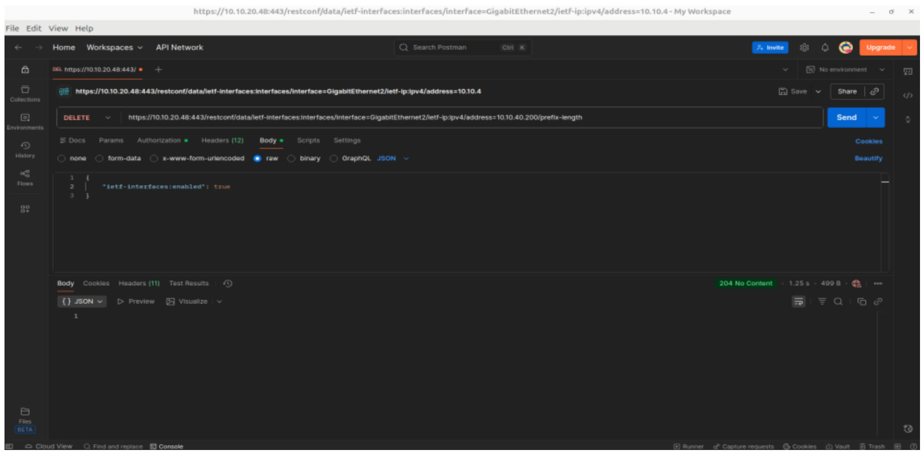
https://10.10.20.48:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/enabled - My Workspace

```
PATCH https://10.10.20.48:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/enabled
```

```
{
  "ietf-interfaces:enabled": true
}
```

```
204 No Content · 1.23 s · 499 B
```

10. POSTMAN DELETE CALL


https://10.10.20.48:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/ietf-ip:ipv4/address=10.10.4 - My Workspace
File Edit View Help
Home Workspaces API Network
Search Postman
Invite
Upgrade
No environment
Save
Share
Send
Docs Params Authorization Headers (12) Body Scripts Settings
Cookies
Beautify
Body Cookies Headers (11) Test Results
Preview Visualize
Cloud View Find and replace Console

https://10.10.20.48:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/ietf-ip:ipv4/address=10.10.4

DELETE https://10.10.20.48:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2/ietf-ip:ipv4/address=10.10.40.200/prefix-length

Body: raw [✓ raw] JSON

```
{
  "ietf-interfaces:enabled": true
}
```

204 No Content · 1.25 s · 499 B

Challenges Faced:

1. Syntax issues with cURL , each line ends with “ \ ” and nothing else after it and no “ \ ” at the final line. Without this bash thinks each line is a new command and throws error.
2. While working with IETF yang modules, we can face some keyword issues which is not shown in yangsuite . For e.g. “subnet ” in IETF modules is not accepted by Cisco IOS XE on cat8kv, instead can be replaced with the keyword “netmask”.
3. We should create fresh environments and variables in Postman , or else old environments and variables may override the new ones and cause connectivity issues.

Future Enhancements:

1. Integration of Python Scripts and Netconf can be done. Netconf is more modular and versatile than RESTCONF.

2. Grafana can be used for CPU utilization of the network devices.

3. Integration of Machine Learning can be done to create a predictive and responsive model.

4. Integration of several Python libraries can be done.