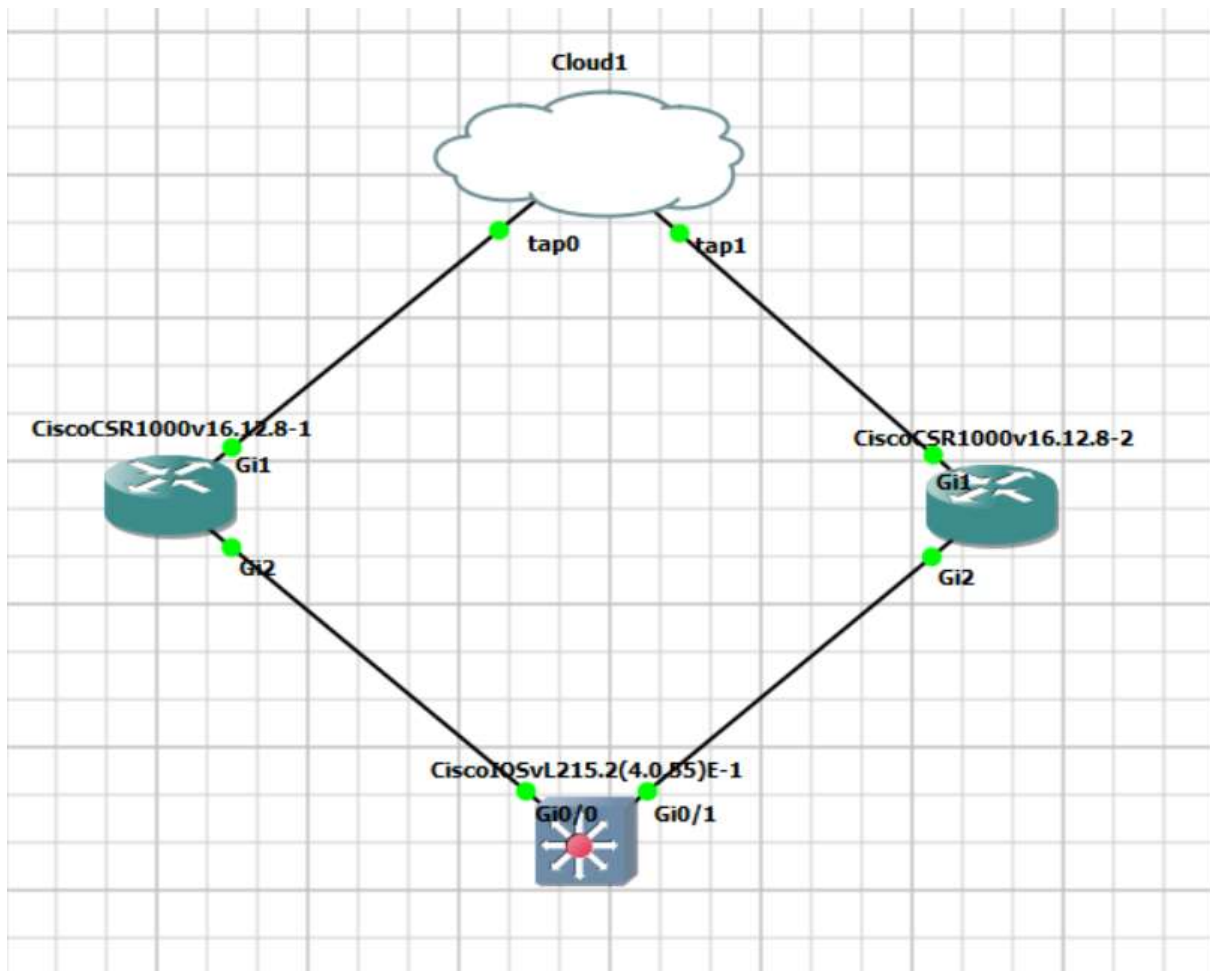


Enable the NETCONF & RESTCONF configuration in router(10-07)

Network Topology:



DHCP will allocate the IP address to the Router:

```
Router>
*Jul 10 13:57:40.193: %SEC_LOGIN-5-LOGIN_SUCCESS: Login Success [user: admin] [S
ource: LOCAL] [localport: 0] at 13:57:40 UTC Wed Jul 10 2024sh ip int br
Interface          IP-Address      OK? Method Status        Protocol
GigabitEthernet1    172.20.0.109    YES DHCP    up            up
GigabitEthernet2    10.0.0.1        YES manual  up            up
GigabitEthernet3    unassigned      YES unset   down          down
GigabitEthernet4    unassigned      YES unset   administrativ down    down
```

Go to Config Mode:

```
Conf t
user admin privilege 15 secret cisco123
aaa new-model
aaa authentication login default local
aaa authorization exec default local
Netconf-yang
```

Check platform software yang-management process:

```
Router#sh platform software yang-management process
confd      : Running
ned        : Running
syncfd     : Running
ncsshd     : Running
mianauthd  : Running
nginx      : Running
nadbmand   : Running
nubd       : Running
Router#_
```

Command prompt and check the yang connectivity:

```
Administrator: Command Prompt - ssh admin@172.20.0.109 -p 830 -s netconf
Microsoft Windows [Version 10.0.19042.1949]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>ssh admin@172.20.0.109 -p 830 -s netconf
The authenticity of host '[172.20.0.109]:830 ([172.20.0.109]:830)' can't be established.
RSA key fingerprint is SHA256:tyvhtLLg6cFfpHx9tRJgaCrRqypzuRKDFpQT+703UQM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[172.20.0.109]:830' (RSA) to the list of known hosts.
admin@172.20.0.109's password:

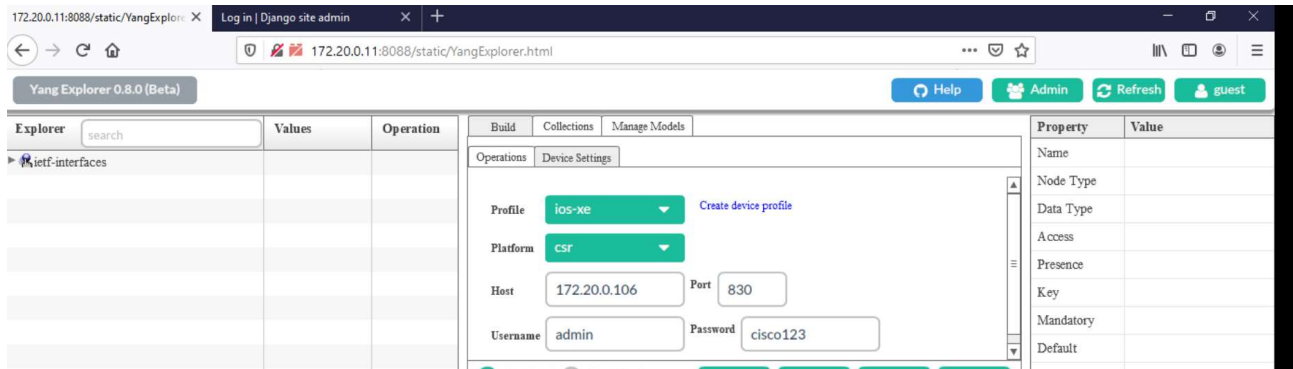
C:\Users\Administrator>ssh admin@172.20.0.109 -p 830 -s netconf
admin@172.20.0.109's password:

C:\Users\Administrator>ssh admin@172.20.0.109 -p 830 -s netconf
admin@172.20.0.109's password:
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
    <capability>urn:ietf:params:netconf:base:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:writable-running:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:xpath:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:validate:1.1</capability>
    <capability>urn:ietf:params:netconf:capability:rollback-on-error:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:notification:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:interleave:1.0</capability>
    <capability>urn:ietf:params:netconf:capability:with-defaults:1.0?basic-mode=explicit&also-supported=report-all-tagged</capability>
    <capability>urn:ietf:params:netconf:capability:yang-library:1.0?revision=2016-06-21&module-set-id=b1077a37217dd85cbd
```

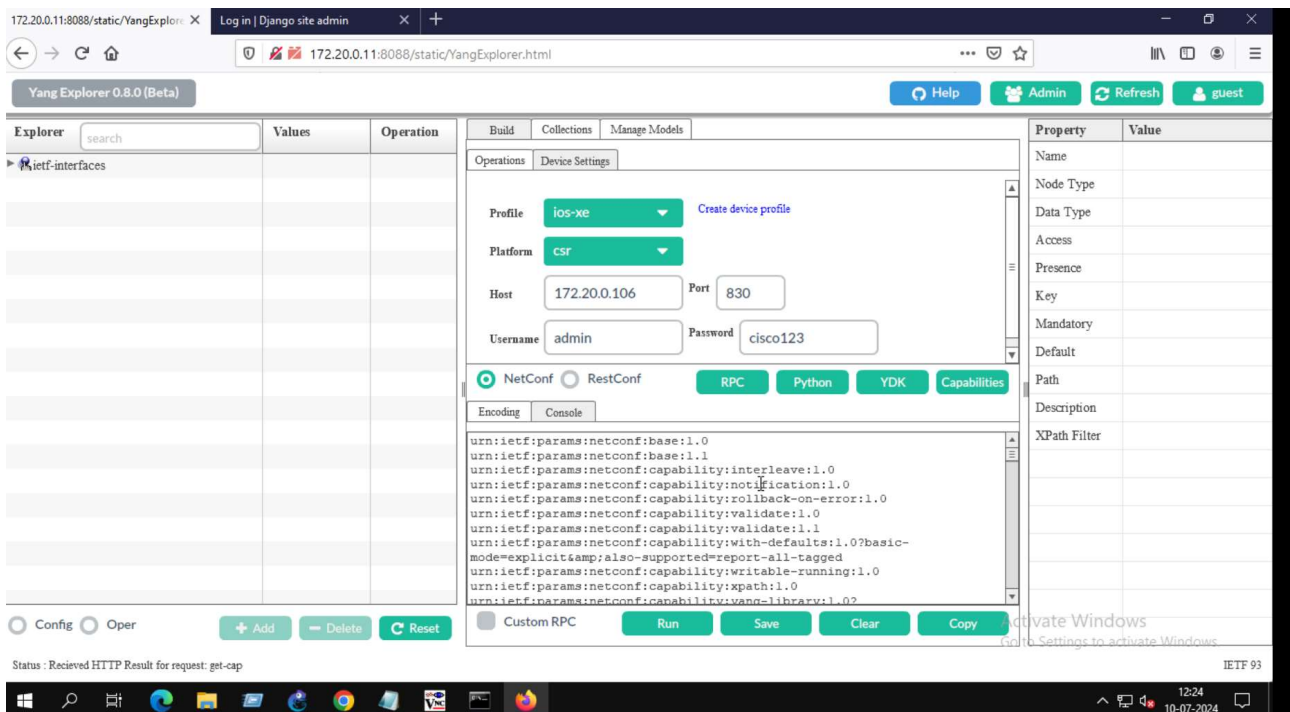
Add the additional Gi2 interface to router and assign the IP 10.0.0.1:

```
Router>en
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int g2
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shut
Router(config-if)#
```

Login to YANG Explorer:



Click on capability, router will exchange the capability.



Click on Manage model and subscribe the ietf-interface yang model

The screenshot shows the Yang Explorer 0.8.0 (Beta) web interface. The 'Manage Models' tab is active, displaying a list of 182 models. The model 'ietf-interfaces@2014-05-08.yang' is selected and has a '[subscribed]' status. The interface includes a search bar, a list of models, and buttons for 'Add', 'Subscribe', 'Un-Subscribe', 'Delete', and 'Graph'. The 'Property' table on the right shows details for the selected model.

Property	Value
Name	
Node Type	
Data Type	
Access	
Presence	
Key	
Mandatory	
Default	
Path	
Description	
XPath Filter	

Example 1; Get the Router Interface Name

The screenshot shows the Yang Explorer 0.8.0 (Beta) web interface with the 'Device Settings' tab active. The 'name' property of the 'interface' node is set to 'GigabitEthernet1'. The 'Platform' is set to 'csr'. The 'Host' is '172.20.0.106' and the 'Port' is '830'. The 'Username' is 'admin' and the 'Password' is 'cisco123'. The 'NetConf' radio button is selected. The 'Console' tab shows the XML configuration for the interface.

```
<name>GigabitEthernet1</name>
</interface>
</interface>
<name>GigabitEthernet2</name>
</interface>
</interface>
<name>GigabitEthernet3</name>
</interface>
</interface>
<name>GigabitEthernet4</name>
</interface>
</interfaces>
```

Example 2; Get the Router Interface admin and Operation state

The screenshot shows the Yang Explorer 0.8.0 (Beta) interface. The left pane displays the 'Explorer' tree with 'interfaces-state' selected. The middle pane shows the 'Operations' tab with 'Device Settings' configured: Profile 'ios-xe', Platform 'csr', Host '172.20.0.106', Port '830', Username 'admin', and Password 'cisco123'. The 'RPC' button is selected. The right pane shows the 'Property' table for 'interfaces-state'.

Property	Value
Name	oper-status
Node Type	leaf
Data Type	enumeration
Access	read-only
Presence	
Key	
Mandatory	true
Default	
Path	ietf-interfaces/interfaces-state/interface/oper-status
Description	The current operational state of the interface. This leaf has the same semantics as ifOperStatus.None
XPath Filter	/if:interfaces-state/interface/oper-status

Status: Received HTTP Result for request: run-rpc

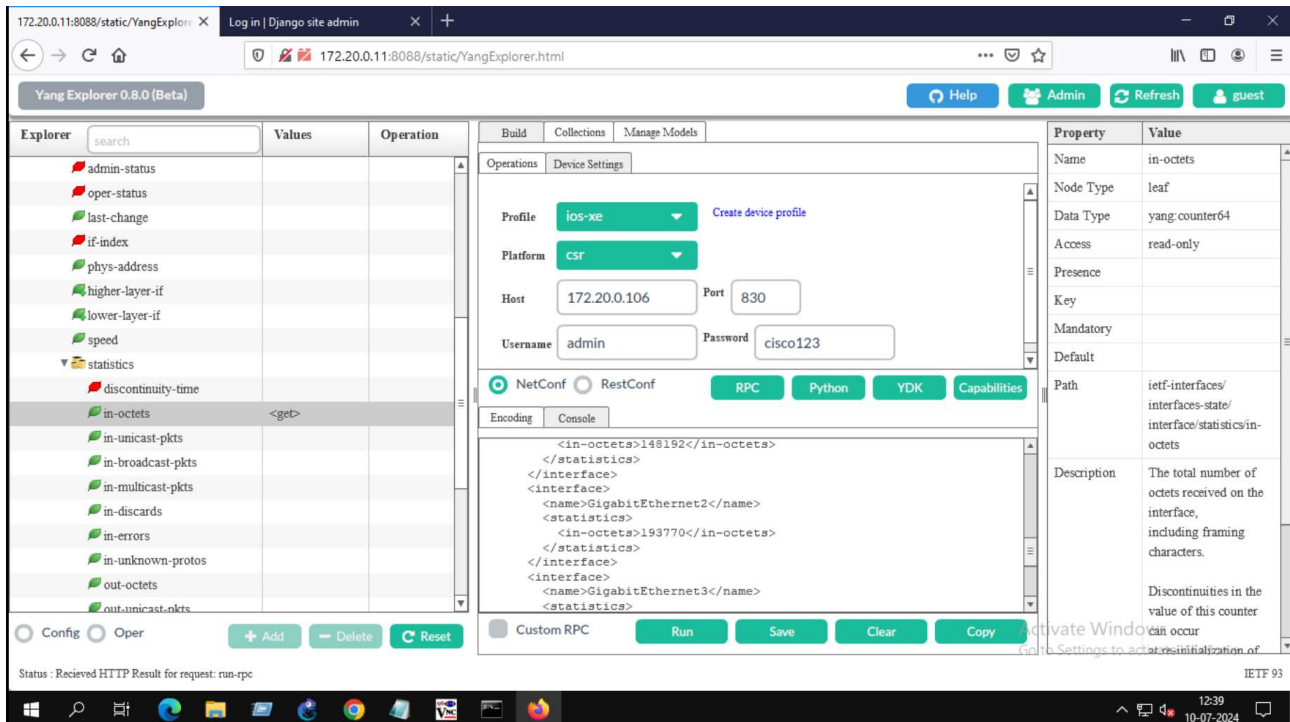
Example 3: Run the YANG query and get the Router interface and IP details

The screenshot shows the Yang Explorer 0.8.0 (Beta) interface. The left pane displays the 'Explorer' tree with 'interface' selected. The middle pane shows the 'Operations' tab with 'Device Settings' configured: Profile 'ios-xe', Platform 'csr', Host '172.20.0.106', Port '830', Username 'admin', and Password 'cisco123'. The 'RPC' button is selected. The right pane shows the 'Property' table for 'interface'.

Property	Value
Name	name
Node Type	leaf
Data Type	string
Access	read-write
Presence	
Key	true
Mandatory	true
Default	
Path	ietf-interfaces/interfaces/interface/name
Description	The name of the interface. A device MAY restrict the allowed values for this leaf, possibly depending on the type of the interface. For system-controlled interfaces, this leaf is

Status: Received HTTP Result for request: run-rpc

Example 4; Run the YANG query and get the Router interface stats

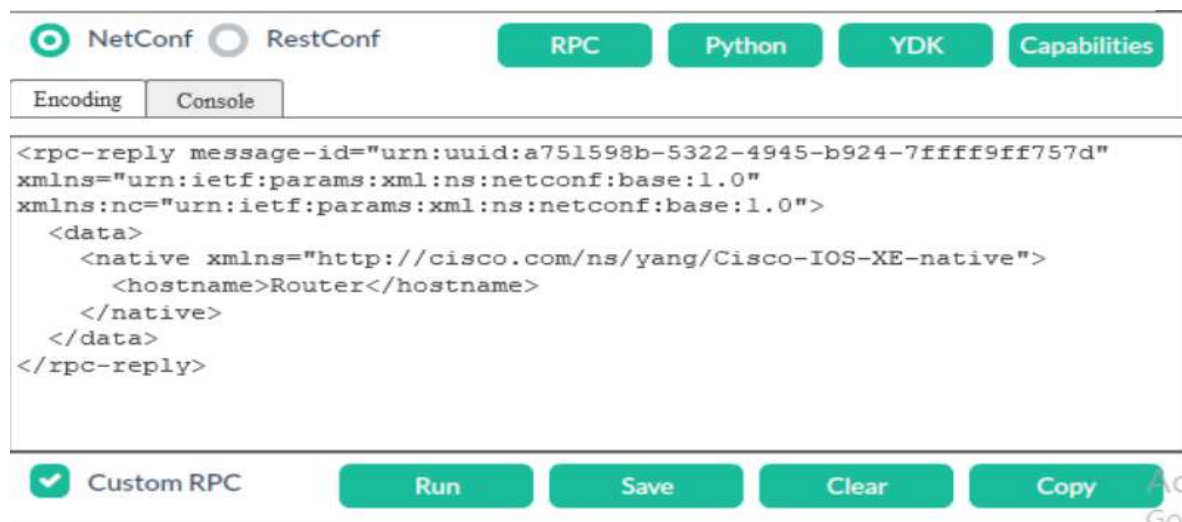


The screenshot shows the Yang Explorer 0.8.0 (Beta) web interface. The interface is divided into several sections:

- Explorer:** A tree view on the left showing the hierarchy of YANG models. The 'statistics' node is expanded, showing various statistics like 'in-octets', 'in-unicast-pkts', 'in-broadcast-pkts', etc.
- Values:** A table showing the values of the selected statistics. The 'in-octets' value is displayed as '<get>'. Below this, there are buttons for '+ Add', '- Delete', and 'Reset'.
- Operation:** A section for performing operations on the selected statistics. It includes a 'Run' button and a 'Custom RPC' checkbox.
- Build:** A section for building the YANG model. It includes a 'Profile' dropdown (set to 'ios-xe'), a 'Platform' dropdown (set to 'csr'), and fields for 'Host' (172.20.0.106), 'Port' (830), 'Username' (admin), and 'Password' (cisco123). There are also buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'.
- Console:** A text area displaying the output of the YANG query. The output is an XML response from the NetConf server, showing the number of octets received on the interface.
- Property:** A table on the right showing the properties of the selected statistics. The 'Name' is 'in-octets', the 'Node Type' is 'leaf', the 'Data Type' is 'yang:counter64', and the 'Access' is 'read-only'. The 'Description' is 'The total number of octets received on the interface, including framing characters.' and 'Discontinuities in the value of this counter can occur at reinitialization of the counter'.

Status: Received HTTP Result for request: run-rpc

1: Get the router hostname:



The screenshot shows the Yang Explorer 0.8.0 (Beta) web interface. The interface is divided into several sections:

- NetConf:** A section for performing operations on the selected statistics. It includes a 'Run' button and a 'Custom RPC' checkbox.
- Console:** A text area displaying the output of the YANG query. The output is an XML response from the NetConf server, showing the hostname of the router.

```
<rpc-reply message-id="urn:uuid:a751598b-5322-4945-b924-7ffff9ff757d"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
      <hostname>Router</hostname>
    </native>
  </data>
</rpc-reply>
```

2: Change the Router hostname:

☒ NetConf ☐ RestConf

RPC Python YDK Capabilities

Encoding Console

```
<rpc-reply message-id="urn:uuid:4344d02d-3c2f-469e-b3cd-73fb52bcae86"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

☒ NetConf ☐ RestConf

RPC Python YDK Capabilities

Encoding Console

```
<rpc-reply message-id="urn:uuid:61b02590-a4cd-47a7-ba88-37fc803bacdf"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
      <hostname>ROUTER1</hostname>
    </native>
  </data>
</rpc-reply>
```

☒ Custom RPC

Run Save Clear Copy

3: Change the interface operational status:

☒ NetConf ☐ RestConf

RPC Python YDK Capabilities

Encoding Console

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
        <interface>
          <name>GigabitEthernet2</name>
          <enabled>false</enabled>
        </interface>
      </interfaces>
    </config>
  </edit-config>
</rpc>
```

☒ Custom RPC

Run Save Clear Copy

4: Check the IP address:



The screenshot shows the NetConf console interface. At the top, there are tabs for 'NetConf' (selected) and 'RestConf'. Below these are buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'. The main area is a text editor with 'Encoding' and 'Console' tabs. The console displays an XML response from a device. At the bottom, there is a 'Custom RPC' checkbox (checked) and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

```
<rpc-reply message-id="urn:uuid:2354a8f8-ffa7-4e73-9f7b-950585a98e96"
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>GigabitEthernet2</name>
        <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
          <address>
            <ip>10.0.0.1</ip>
            <netmask>255.0.0.0</netmask>
          </address>
        </ipv4>
      </interface>
    </interfaces>
  </data>
</rpc-reply>
```


5: Assign the IP address on interface:



The screenshot shows the NetConf console interface. At the top, there are tabs for 'NetConf' (selected) and 'RestConf'. Below these are buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'. The main area is a text editor with 'Encoding' and 'Console' tabs. The console displays an XML configuration for an interface. At the bottom, there is a 'Custom RPC' checkbox (checked) and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

```
<running/>
</target>
<config>
  <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <interface>
      <name>GigabitEthernet2</name>
      <enabled>true</enabled>
      <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
        <address>
          <ip>22.22.22.2</ip>
          <netmask>255.255.255.0</netmask>
        </address>
      </ipv4>
    </interface>
  </interfaces>
</config>
```

6: configure the loopback IP address



The screenshot shows the NetConf console interface. At the top, there are tabs for 'NetConf' (selected) and 'RestConf'. Below these are buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'. The main area is a text editor with 'Encoding' and 'Console' tabs. The console displays an XML configuration for a loopback interface. At the bottom, there is a 'Custom RPC' checkbox (checked) and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

```
<Loopback>
  <name>4</name>
  <ip>
    <address>
      <primary>
        <address>14.1.1.1</address>
        <mask>255.255.255.0</mask>
      </primary>
    </address>
  </ip>
</Loopback>
</interface>
```


7: Attempt to create new loopback interface with same IP address:

☒ NetConf ☐ RestConf

RPC Python YDK Capabilities

Encoding Console

```
<nc:rpc-error xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <nc:error-type>application</nc:error-type>
  <nc:error-tag>invalid-value</nc:error-tag>
  <nc:error-severity>error</nc:error-severity>
  <nc:error-message lang="en" xmlns="http://www.w3.org/XML/1998/
namespace">inconsistent value: Device refused one or more commands</
nc:error-message>
  <nc:error-info>
    <severity xmlns="http://cisco.com/yang/cisco-ia">error_cli</
severity>
    <detail xmlns="http://cisco.com/yang/cisco-ia">
      <bad-cli>
```

☒ Custom RPC

Run Save Clear Copy

8: Delete the Loopback IP address:

☒ NetConf ☐ RestConf

RPC Python YDK Capabilities

Encoding Console

```
<native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
  <interface operation='delete'>
    <Loopback>
      <name>4</name>
      <ip>
        <address operation='delete'>
          <primary>
            <address>14.1.1.1</address>
            <mask>255.255.255.0</mask>
          </primary>
        </address>
      </ip>
```

☒ Custom RPC

Run Save Clear Copy

9: Delete the loopback interface:

☒ NetConf ☐ RestConf

RPC Python YDK Capabilities

Encoding Console

```
<edit-config>
  <target>
    <running/>
  </target>
  <config>
    <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
      <interface>
        <Loopback operation='delete'>
          <name>1</name>
        </Loopback>
      </interface>
    </native>
```

☒ Custom RPC

Run Save Clear Copy

10: Get Complete Config:

The screenshot shows a web-based NetConf console. At the top, there are radio buttons for 'NetConf' (selected) and 'RestConf'. To the right are buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'. Below these are tabs for 'Encoding' and 'Console'. The main text area contains an XML RPC request:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <get-config>
    <source>
      <running/>
    </source>
    <filter type="subtree">
      <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
      </native>
    </filter>
  </get-config>
</rpc>
```

 At the bottom, there is a checked checkbox for 'Custom RPC' and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

11: Get the filtered configuration:

The screenshot shows the same NetConf console interface. The main text area now displays an XML response:

```
<data>
  <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
    <version>16.12</version>
    <boot-start-marker/>
    <boot-end-marker/>
    <memory>
      <free>
        <low-watermark>
          <processor>72291</processor>
        </low-watermark>
      </free>
    </memory>
  </native>
</data>
```

 The bottom controls remain the same: a checked 'Custom RPC' checkbox and 'Run', 'Save', 'Clear', and 'Copy' buttons.

Enable Candidate Data Store

```
Conf t
user admin privilege 15 secret cisco123
aaa new-model
aaa authentication login default local
aaa authorization exec default local
Netconf-yang feature candidate-datastore
Netconf-yang
```

12: Enable Candidate data store:

The screenshot shows the NetConf console interface. At the top, there are tabs for 'NetConf' (selected) and 'RestConf'. Below these are buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'. The 'Encoding' and 'Console' tabs are visible. The main text area contains an XML RPC message:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <get>
    <source>
      <candidate/>
    </source>
    <filter>
      <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
        <interface>
          <name>GigabitEthernet2</name>
          <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
            <address/>
          </ipv4>
        </interface>
      </interfaces>
    </filter>
  </get>
</rpc>
```

At the bottom, there is a 'Custom RPC' checkbox (checked) and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

13: make the changes on candidate data store:

Change the interface status to down (Gi2) in candidate data store:

The screenshot shows the NetConf console interface. At the top, there are tabs for 'NetConf' (selected) and 'RestConf'. Below these are buttons for 'RPC', 'Python', 'YDK', and 'Capabilities'. The 'Encoding' and 'Console' tabs are visible. The main text area contains an XML RPC message:

```
<rpc message-id="101" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <candidate/>
    </target>
    <config>
      <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
        <interface>
          <name>GigabitEthernet2</name>
          <enabled>false</enabled>
        </interface>
      </interfaces>
    </config>
  </edit-config>
</rpc>
```

At the bottom, there is a 'Custom RPC' checkbox (checked) and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

Commit the changes:

The screenshot shows the NetConf console interface. At the top, there are tabs for 'Encoding' and 'Console'. The main text area contains an XML RPC message:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <commit></commit>
</rpc>
```

At the bottom, there is a 'Custom RPC' checkbox (checked) and buttons for 'Run', 'Save', 'Clear', and 'Copy'.

Create the Loopback interface on candidate data store:

☒ NetConf ☐ RestConf RPC Python YDK Capabilities

Encoding Console

```
<native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
  <interface>
    <Loopback>
      <name>l</name>
      <ip>
        <address>
          <primary>
            <address>10.1.1.1</address>
            <mask>255.255.255.0</mask>
          </primary>
        </address>
      </ip>
    </interface>
  </native>
```

☒ Custom RPC Run Save Clear Copy

Commit the changes:

Encoding Console

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <commit></commit>
</rpc>
```

☒ Custom RPC Run Save Clear Copy

Change the router Hostname on candidate data store and commit the change:

☒ NetConf ☐ RestConf RPC Python YDK Capabilities

Encoding Console

```
<edit-config>
  <target>
    <running/>
  </target>
  <config>
    <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
      <hostname>R1</hostname>
    </native>
  </config>
</edit-config>
</rpc>
```

☒ Custom RPC Run Save Clear Copy

Commit the changes:

Encoding

Console

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <commit></commit>
</rpc>
```

☒ Custom RPC

Run

Save

Clear

Copy

15. Copy the configuration from running data store to candidate data store:

☒ NetConf ☐ RestConf

RPC

Python

YDK

Capabilities

Encoding

Console

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <copy-config>
    <target>
      <candidate/>
    </target>
    <source>
      <running/>
    </source>
  </copy-config>
</rpc>
```

☒ Custom RPC

Run

Save

Clear

Copy

16: Close the session:

☒ NetConf ☐ RestConf

RPC

Python

YDK

Capabilities

Encoding

Console

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <close-session/>
</rpc>
```

☒ Custom RPC

Run

Save

Clear

Copy

RestConf:

Enable the router with Restconf capability

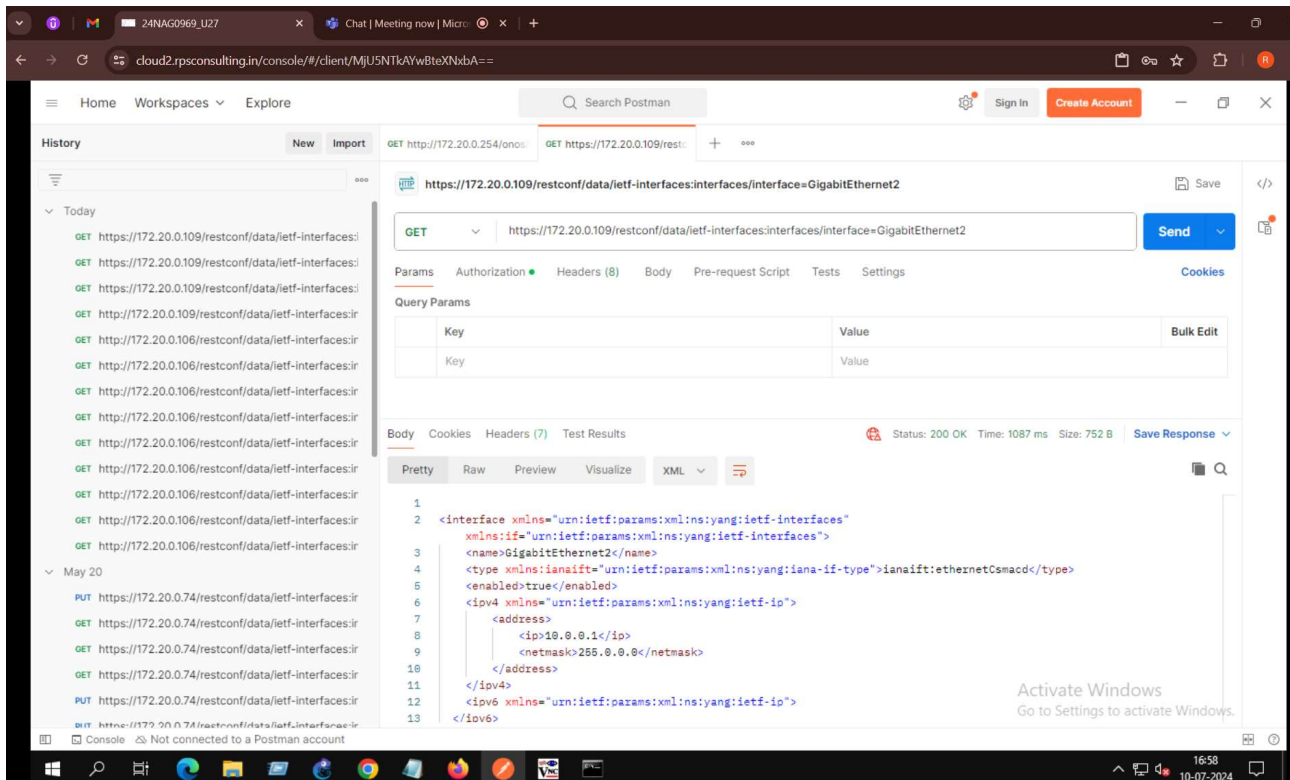
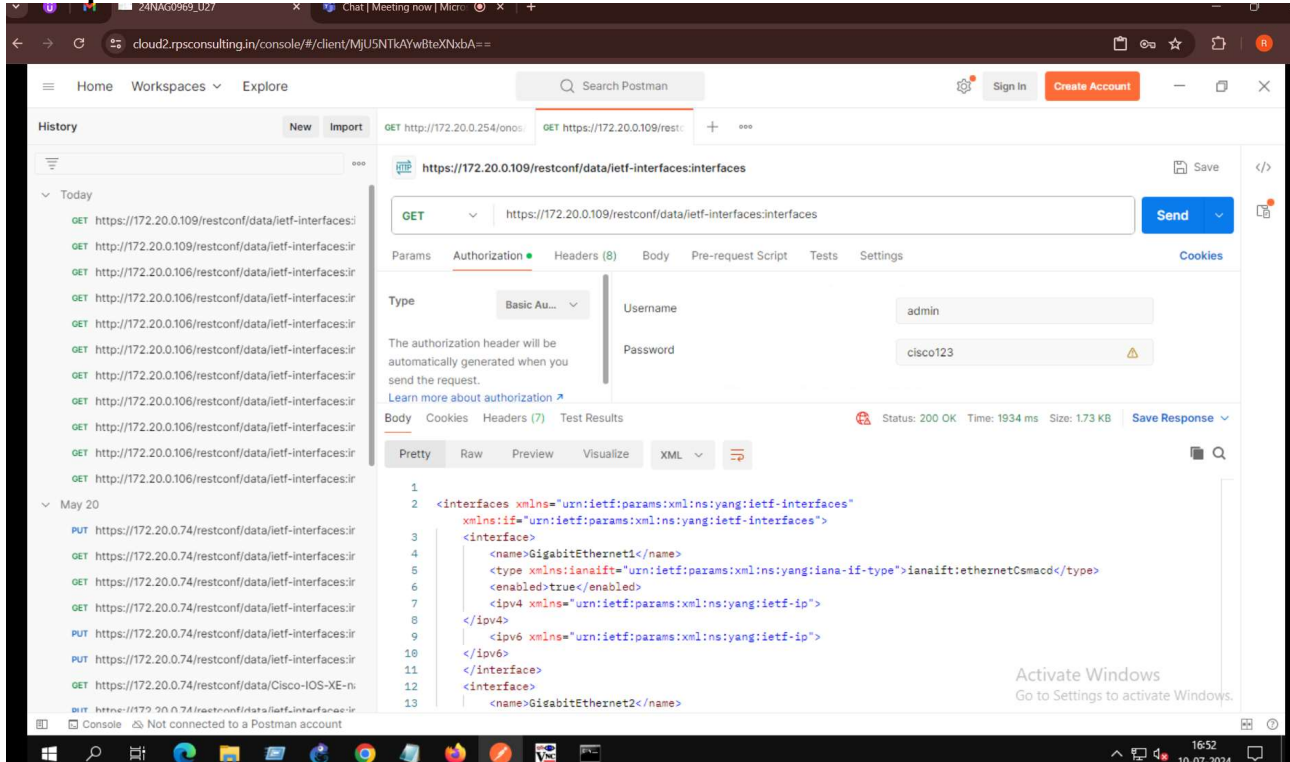
Conf t

Restconf

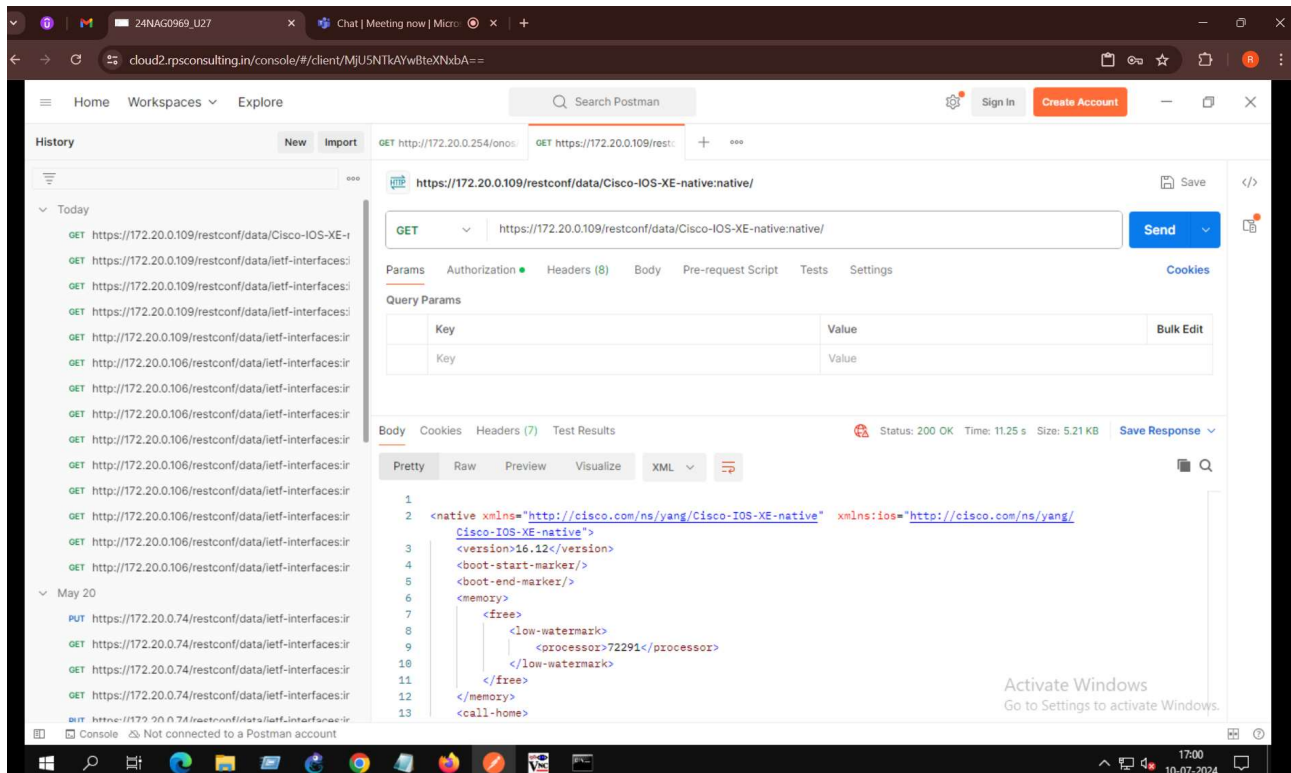
Ip http secure-server

Login to Postman:

Example : Get the Router interface details

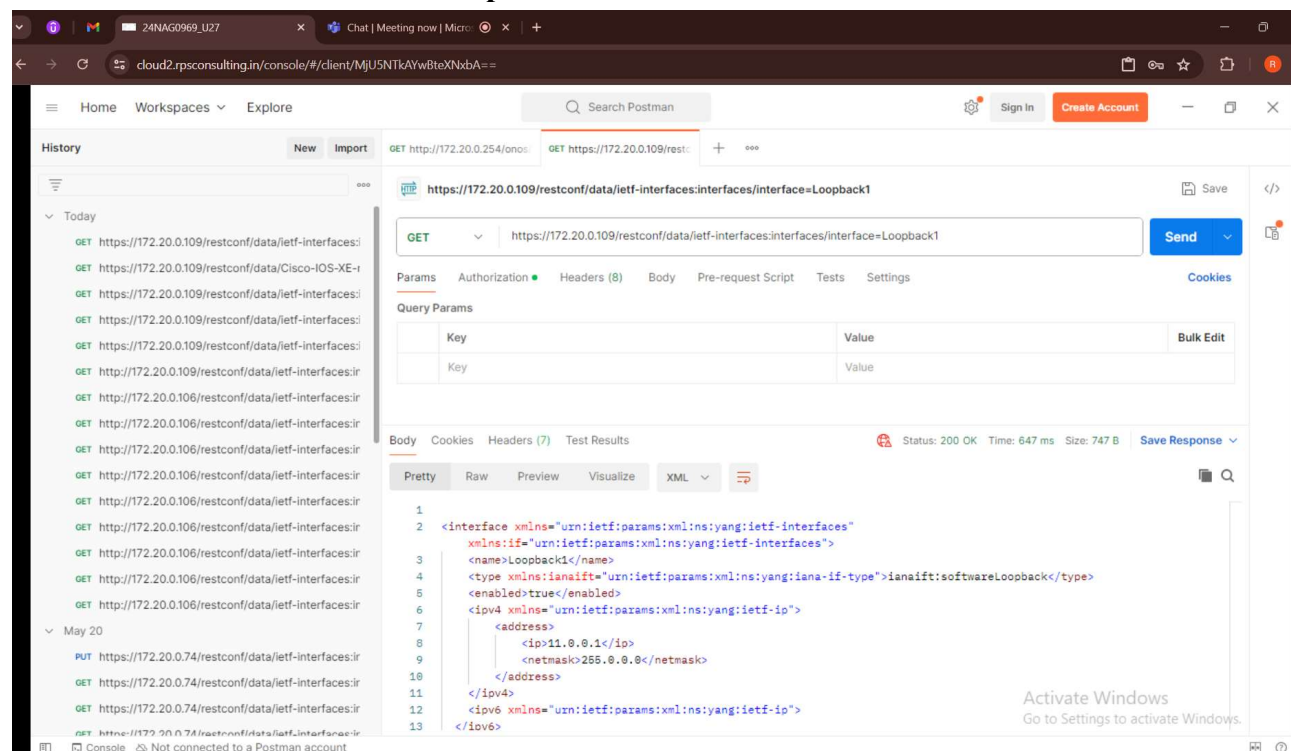


Get Config



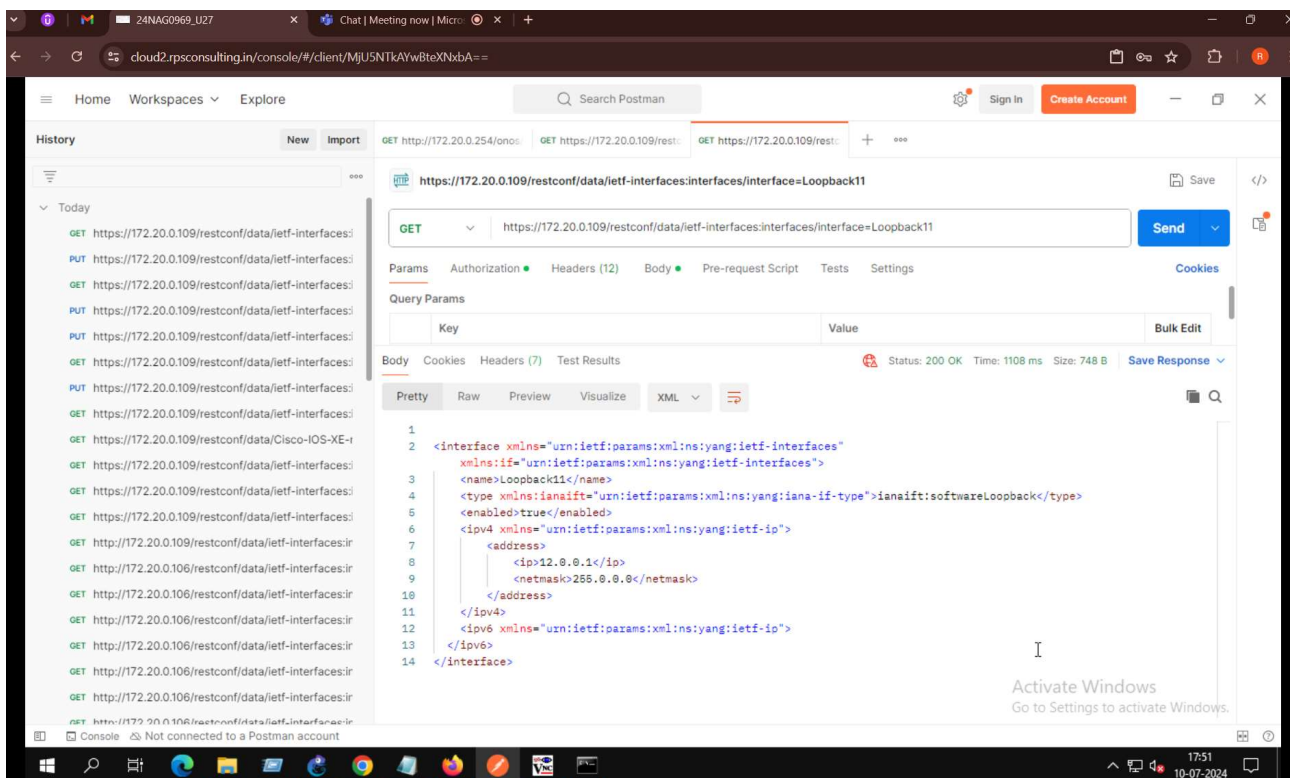
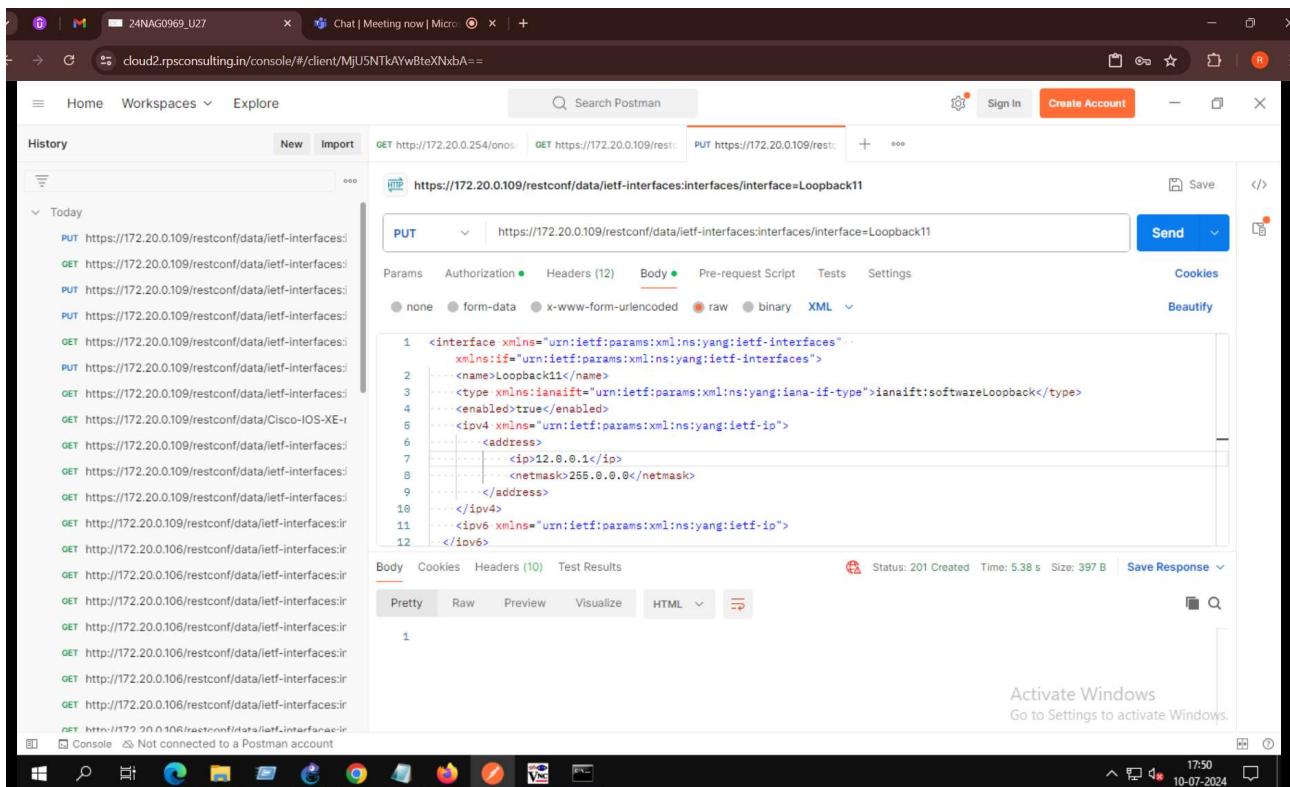
Example :
Create the new Loopback interface on router via CLI
Conf t
Int loopback1
Ip address 11.0.0.1 255.0.0.0

GET : https://IP address of server/restconf/data/ietf-interfaces:interfaces/interface=Loopback1

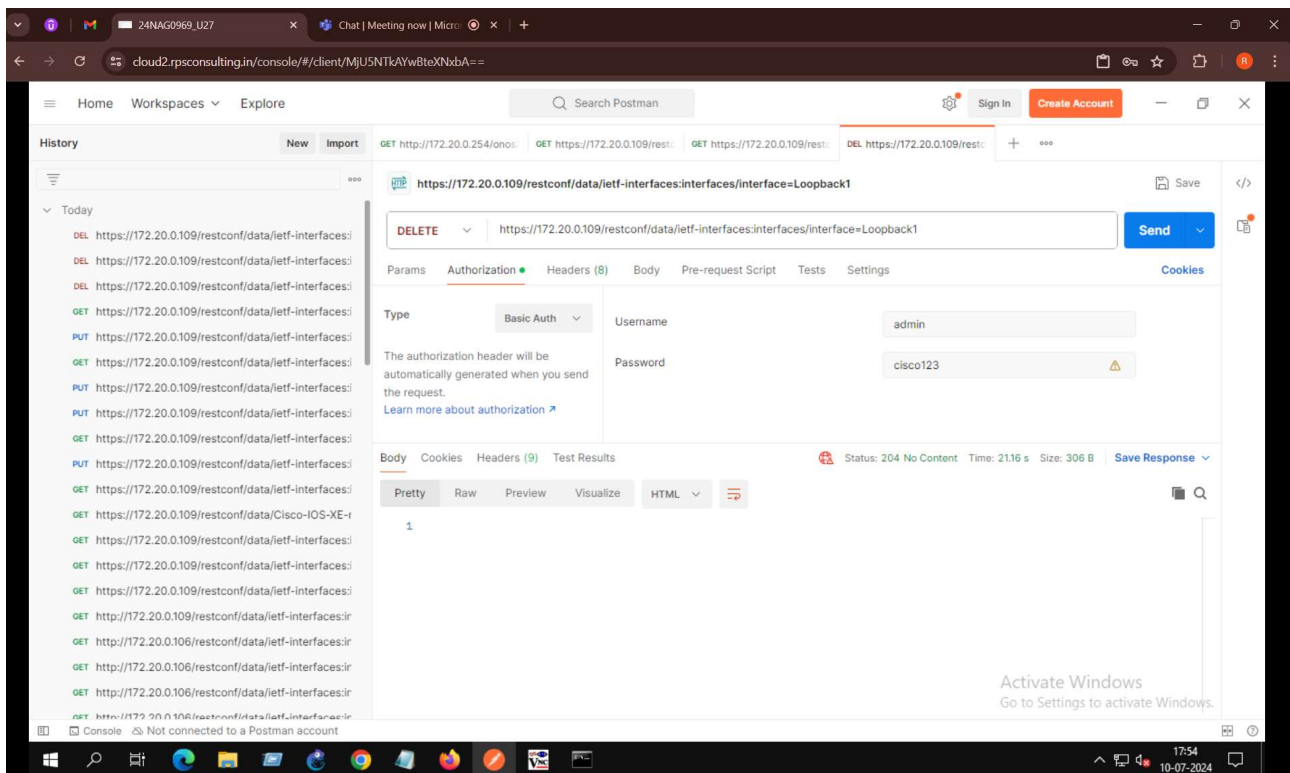


Example : Create the Loopback interface via Postman

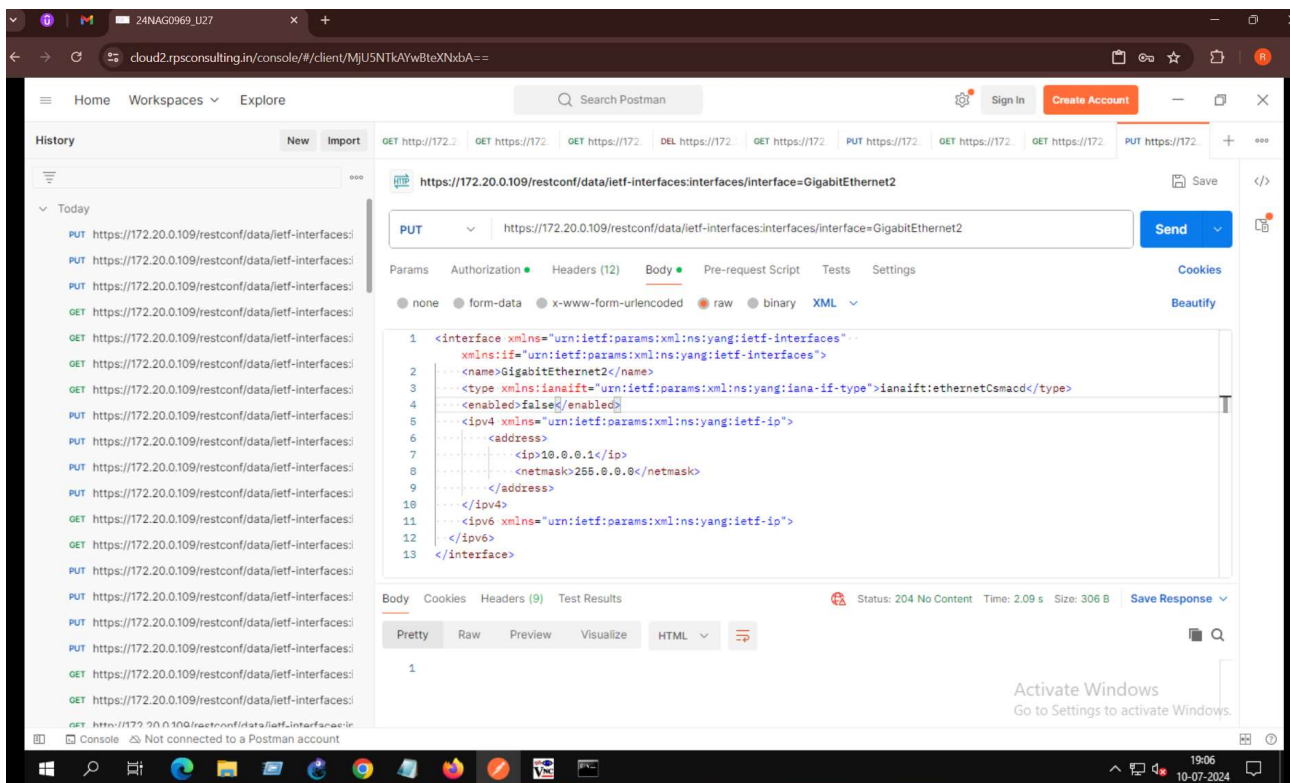
PUT : https://IP address of server/restconf/data/ietf-interfaces:interfaces/interface=Loopback11



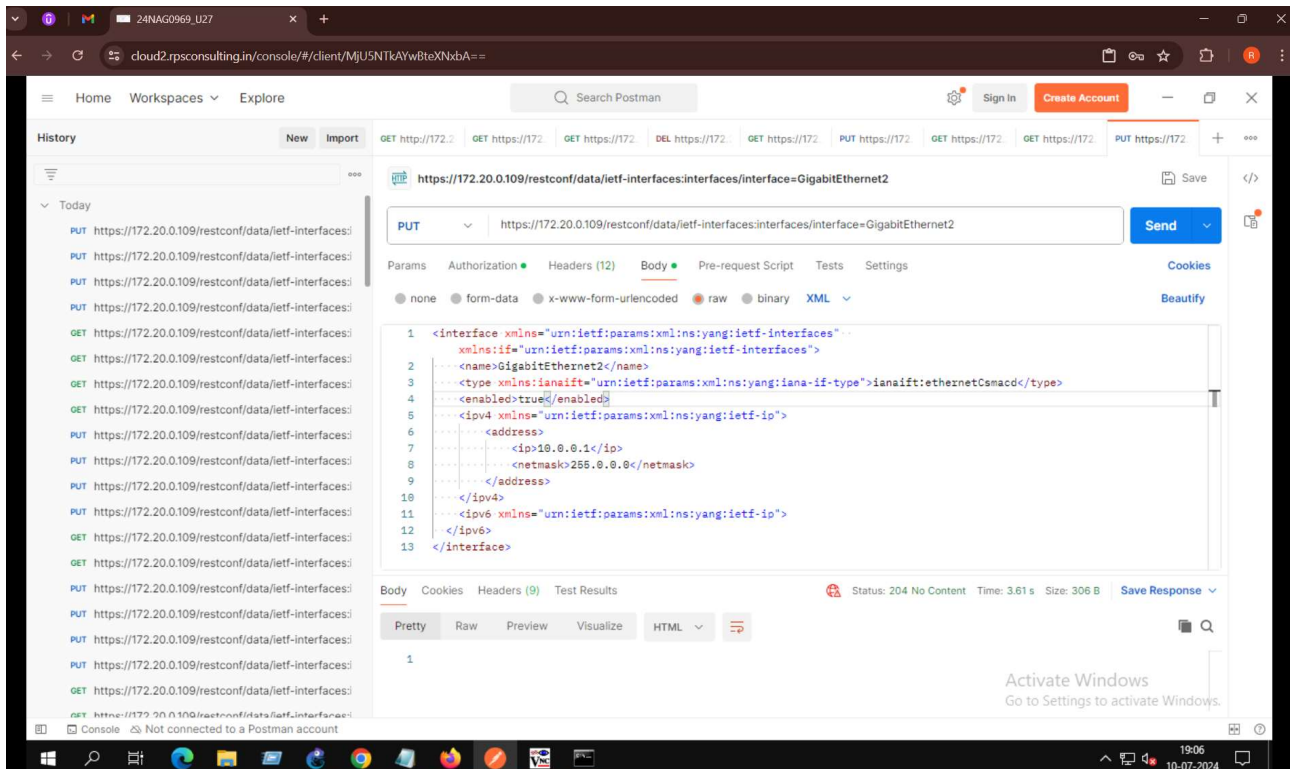
Delete the Loopback interface.



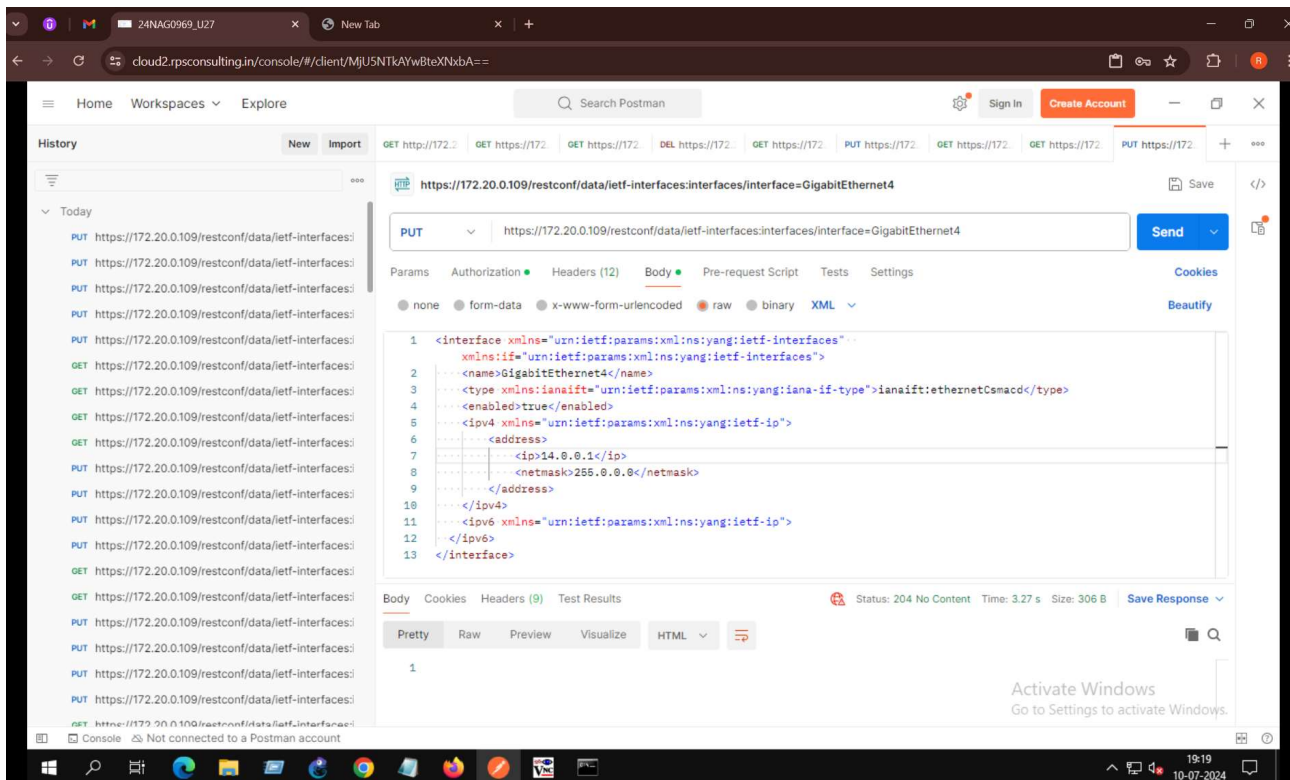
Change the Interface status to down for GigabitEthernet2 interface



Change the Interface status to UP for GigabitEthernet2 interface



Assign the interface IP address



Remove the Interface IP address

The screenshot shows the Postman application interface. The top bar includes navigation links (Home, Workspaces, Explore), a search bar, and user options (Sign In, Create Account). The left sidebar displays a 'History' list of recent requests, all targeting the URL `https://172.20.0.109/restconf/data/ietf-interfaces:interfaces`. The main workspace is configured for a **PUT** request to `https://172.20.0.109/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet4`. The request body is an XML payload:

```
1 <interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
2   xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces">
3   <name>GigabitEthernet4</name>
4   <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:ethernetCsmacd</type>
5   <enabled>false</enabled>
6   <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
7   <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
8 </ipv6>
9 </interface>
```

The response status is **204 No Content**, with a time of 3.60 s and a size of 306 B. The bottom status bar shows the system time as 19:20 on 10-07-2024.