

network automation using python

By Prashant Sharma



July 15, 2020

wipro LTD

Bangalore

# OBJECTIVE

# The objective of networking automation is to generate autonomous processes, allowing your network to execute on essential activities without human intervention. In recent years, networking automation technology has reached new levels of maturity; it’s starting to become possible and feasible for businesses of all sizes to use network automation tools.

Table of Contents

S.No TOPIC NAME PAGE.NO

1. INTRODUCTION ……………………………………………………………………………………………………………………..2
2. SETTING UP ENVIRONMENT……………………………………………………………………………………………………3
3. INSTALLING ANACONDA…………………………………………………………………………………………………………4
4. CREATING VIRTUAL ENIVORNMENT……………………………………………………………………………………….8
5. INSTALLING PYTHON 2 & 3…………………………………………………………………………………………………….12
6. COMPARING PYTHON 2 & 3…………………………………………………………………………………………………..--
7. ABOUT GNS3………………………………………………………………………………………………………………………….13
8. INSTALLING GNS3…………………………………………………………………………………………………………………..14
9. CONNECTING DEVICES IN GNS3……………………………………………………………………………………………..16
10. CONFIGURING ROUTER………………………………………………………………………………………………………… 18
11. ASSIGNING IP ADDRESS TO A ROUTER……………………………………………………………………………………20
12. TELNET LIBRARY……………………………………………………………………………………………………………………..21
13. HOW TO ENABLE TELNET IN WINDOWS…………………………………………………………………………………22
14. INTRODUCTION TO NETMIKO……………………………………………………………………………………………….23
15. HOW TO INSTALL NETMIKO LIBRARY…………………………………………………………………………………….24
16. INTRODUCTION TO PARAMIKO LIBRARY………………………………………………………………………………..25
17. HOW TO INSTALL PARAMIKO LIBRARY…………………………………………………………………………………..26
18. Adding Loopback Interface…………………………………………………………………………………………………….27
19. Pinging router through system cmd……………………………………………………………………………………….42
20. First Script……………………………………………………………………………………………………………………………..44
21. Creating loopbacks and vlan using script……………………………………………………………………………….51
22. Creating N number of vlans using script…………………………………………………………………………………64
23. Fetching device and update configuration……………………………………………………………………………..65
24. Using paramiko…………….………………………………………………………………………………………………………..70
25. Using paramiko for multiple devices……………………………………………………………………………………….73
26. Troubleshooting using paramiko…………………………………………………………………………………………….77
27. Device accessibility using paramiko………………………………………………………………………………………..81
28. Using Netmiko……………………………………………………………………………………………………………………….86
29. Configure multiple devices using txt file…………………………………………………………………………………87
30. Reading commands using function from text file…………………………………………………………………..90
31. Handle device not reachable exceptions in Netmiko……………………………………………………………..91
32. Schedule task in python3……………………………………………………………………………………………………….92
33. How to schedule auto backup of cisco devices………………………………………………………………………94
34. Schedule backup using paramiko…………………………………………………………………………………………..96
35. Installing git on windows……………………………………………………………………………………………………….98
36. How to Parse Cisco Device configuration in Python 3 using TEXTFSM……………………………………100
37. How to get a specific info about device using script………………………………………………………………107
38. How to list Cisco Interfaces based on Status……………………………………………………………………….109
39. If primary interface is down make secondary interface UP using script………………………………111
40. How to search for specific interface using script…………………………………………………………………114
41. NAPALM library………………………………………………………………………………………………………………….116
42. Napalm script to get facts of device……………………………………………………………………………………117
43. Printing facts of device in a cleaner format………………………………………………………………………..119
44. How to rollback the configuration………………………………………………………………………………………120
45. How to rollback for arista device………………………………………………………………………………………..121
46. Nornir Automation framework…………………………………………………………………………………………..122
47. Nornir sample script…………………………………………………………………………………………………………..123
48. Atom IDE python setup………………………………………………………………………………………………………128
49. NETCONF……………………………………………………………………………………………………………………………132
50. Script to get capabilities……………………………………………………………………………………………………..133
51. Script to get running config………………………………………………………………………………………………..134
52. Multithreading in netmiko………………………………………………………………………………………………….135
53. Using json with netmiko……………………………………………………………………………………………………..138
54. Multithreading in detail………………………………………………………………………………………………………140

# INTRODUCTION

Network automation is the automation of routine or repetitive tasks within networking. This includes writing one-off scripts for routers and servers.

# Setting Up Environment

To do the network automation we are using python here.

Here are various approaches to use python in windows

1. We can use notepad or sublime text to write our code and windows command prompt to compile our code
2. We can use Pycharm IDE
3. We can Anaconda which provides you bunch of IDE like Spyder, jupyter notebook

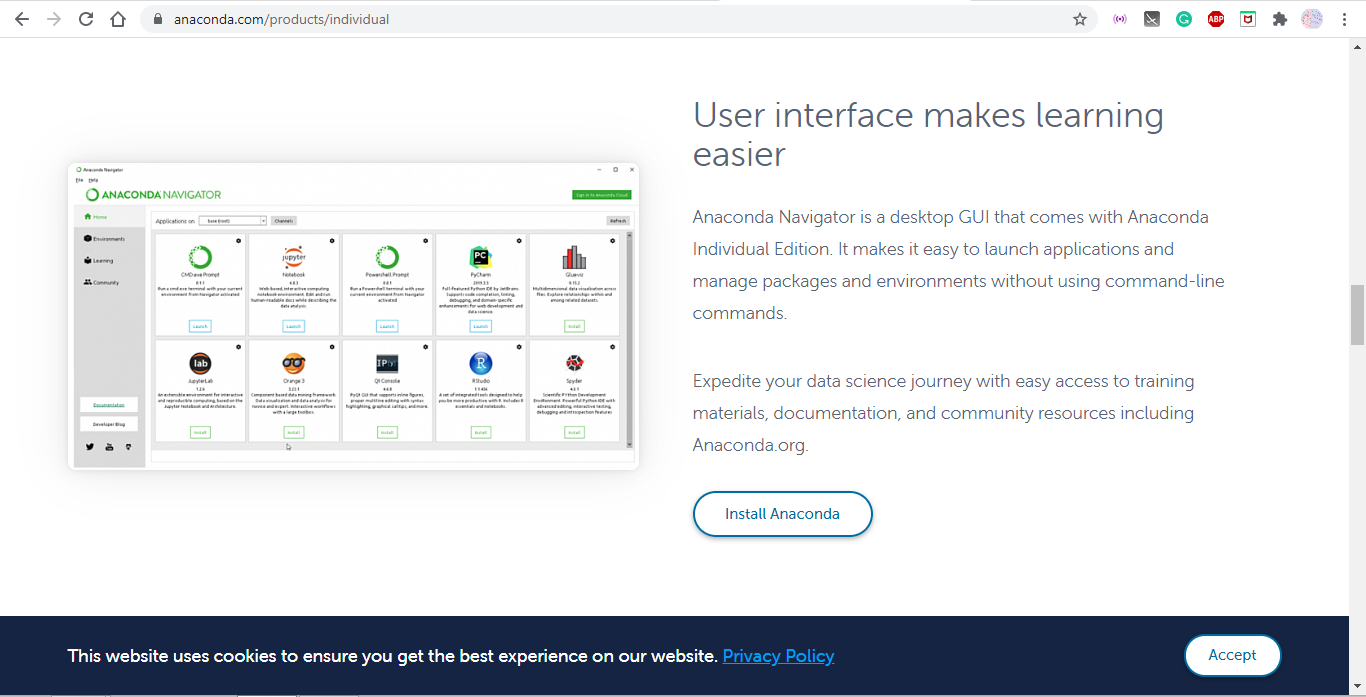
We will go with Anaconda here as it gives us the flexibility to create virtual environment also.

# INSTALLING ANACONDA

Here are the steps which we need to follow to install anaconda

1. Visit <https://www.anaconda.com/download/>

The Anaconda Downloads Page will look something like this:



1. **Select Windows**

Select Windows where the three operating systems are listed.

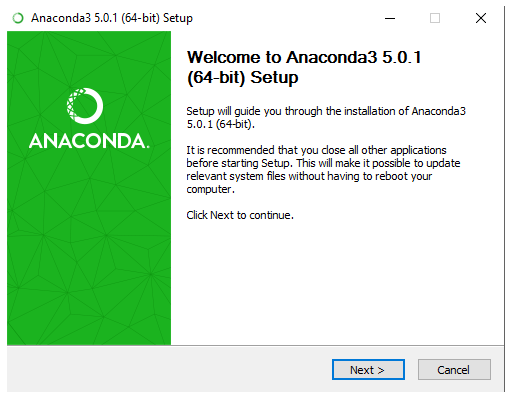


1. **Download**

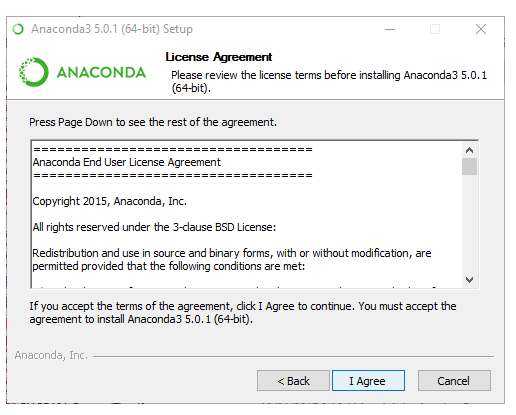
Download the most recent Python 3 release. At the time of writing, the most recent release was the Python 3.8 Version. Python 2.7 is legacy Python. For problem solvers, select the Python 3.8 version. If you are unsure if your computer is running a 64-bit or 32-bit version of Windows, select 64-bit as 64-bit Windows is most common.

When the file will be downloaded, open it.

At the beginning of the install, you need to click **Next** to confirm the installation



Then agree to the license.



After clicking **I Agree**, Click on the install button and you are good to go.

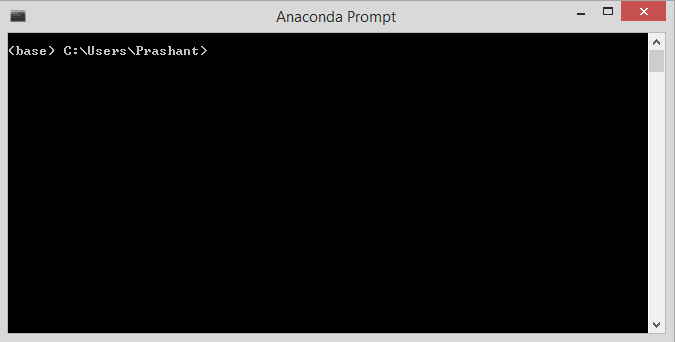
# Creating virtual environment in Anaconda

A Python virtual environment is an isolated container that runs on Python, and it helps make projects self-contained and reproducible without any existing version interference. In other words, it’s like creating your own Python world in your terminal that’s customizable to the point of even specifying what version of Python you want it to run on, be it 2.7 or 3.8.

Way to create virtual environment in anaconda

1. Open anaconda prompt from search menu in windows

It will look like this,

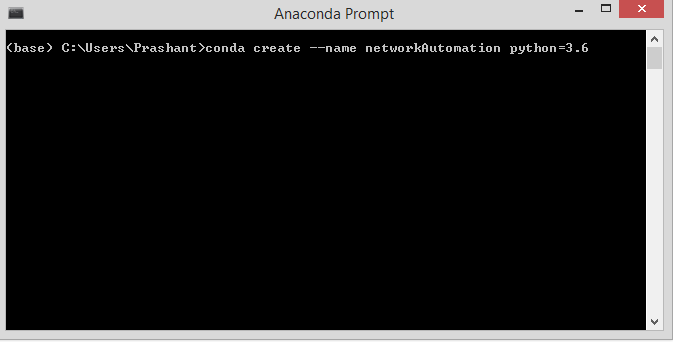


1. Type the below command

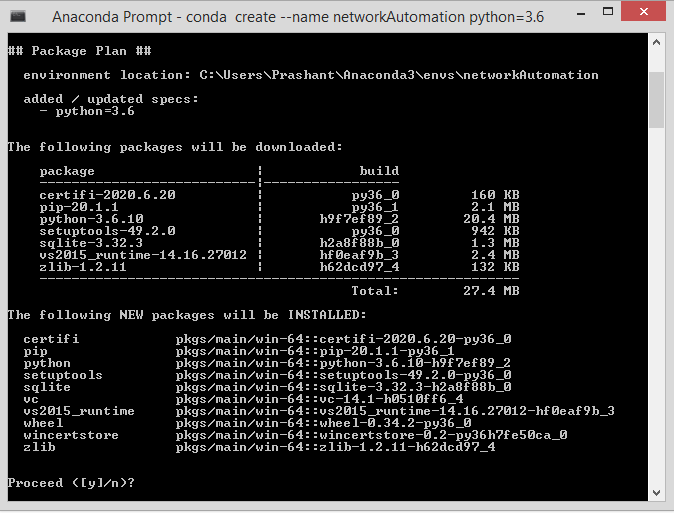
conda create --name env\_name python=version

You can give any name in place of env\_name in command

For ex,



After pressing enter key you will get,



Type Y and click enter, it will create a virtual environment for you with the python version you want.

* To Activate the virtual environment, use this command

**Conda activate name\_of\_our\_env**

* To deactivate the virtual environment, use this command.

**Conda deactivate**

# Installing Python 2 & 3

We have already installed python 3.6 above, using same method you can install python 2.7 as well

# About GNS3

Graphical Network Simulator-3 is a network software emulator first released in 2008. It allows the combination of virtual and real devices, used to simulate complex networks. It uses Dynamips emulation software to simulate Cisco IOS.

# Installing GNS3

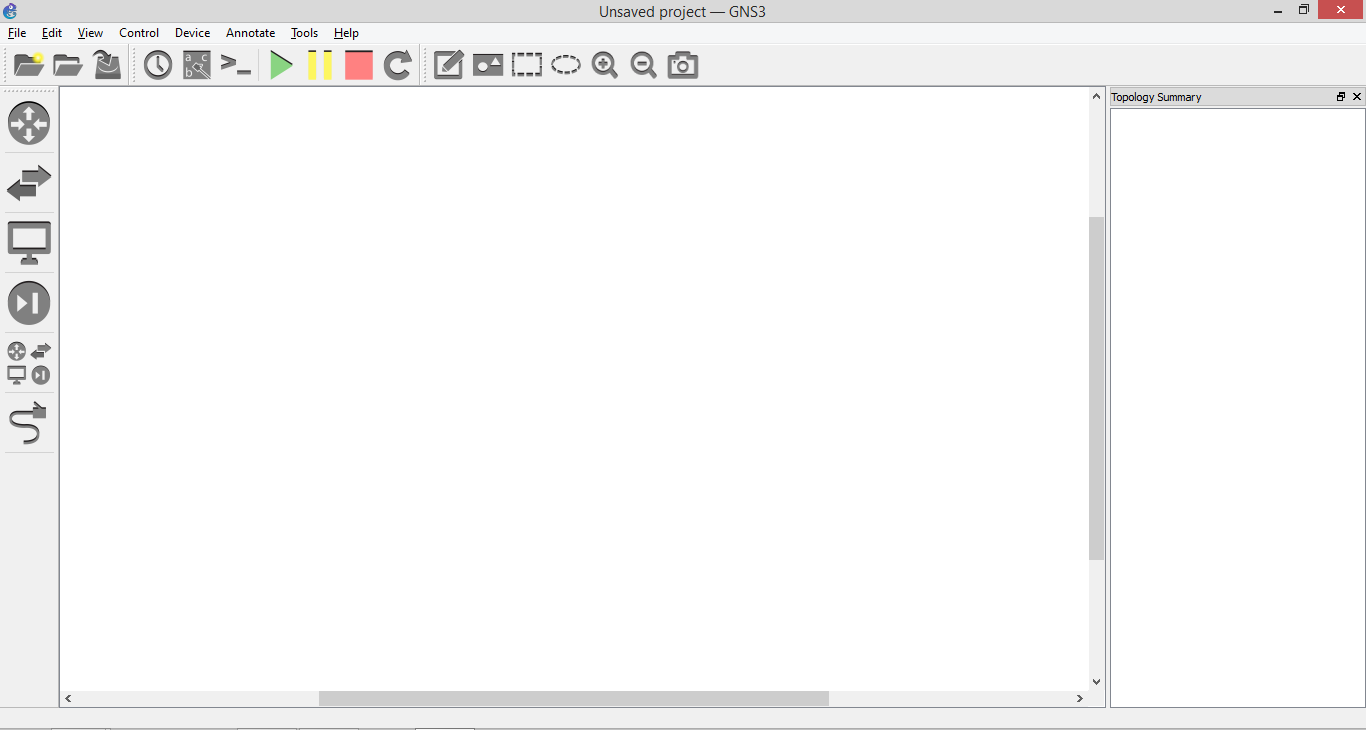
1. Visit <https://www.gns3.com/software/download>
2. Create an account
3. Download GNS3 for windows
4. After downloading the file, open it

It will look like this



1. Click next
2. Click I agree
3. Keep pressing next untill you get install button
4. Press intall and after few seconds it will automatically install

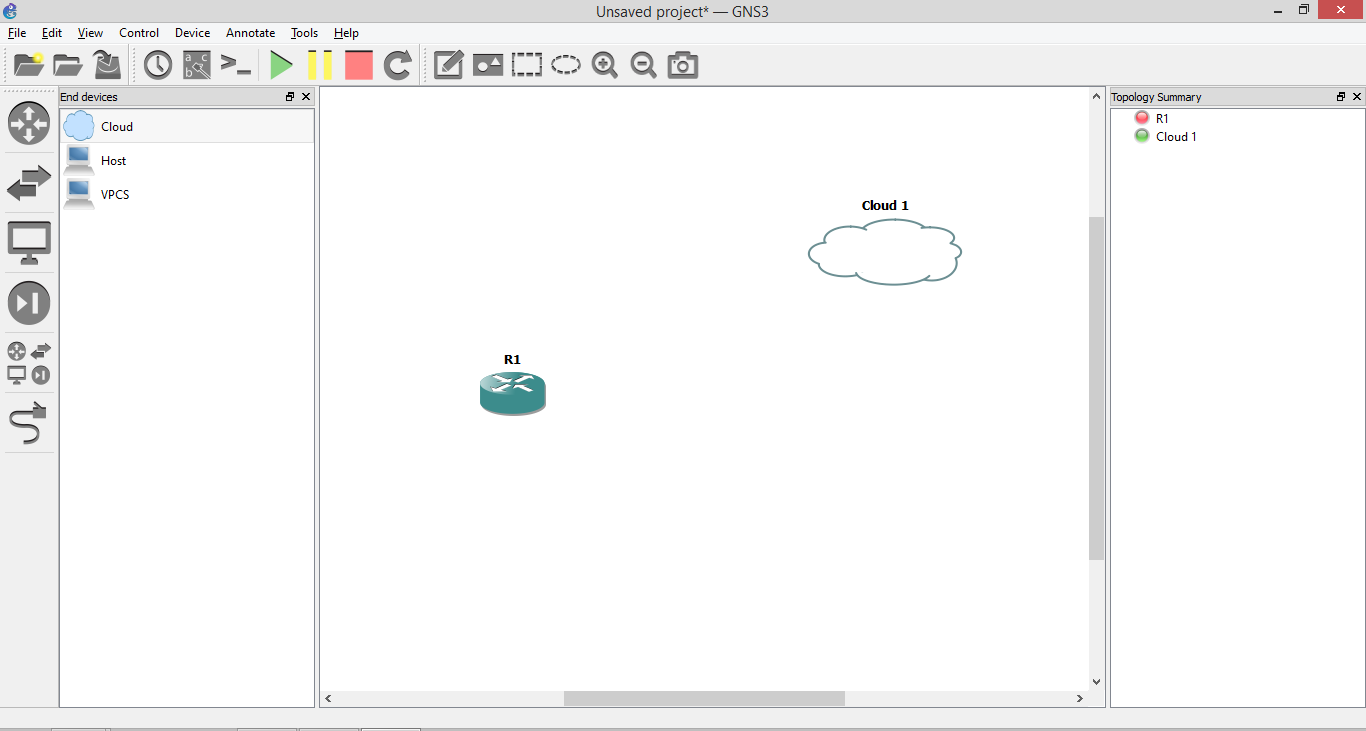
After installing it will look like this



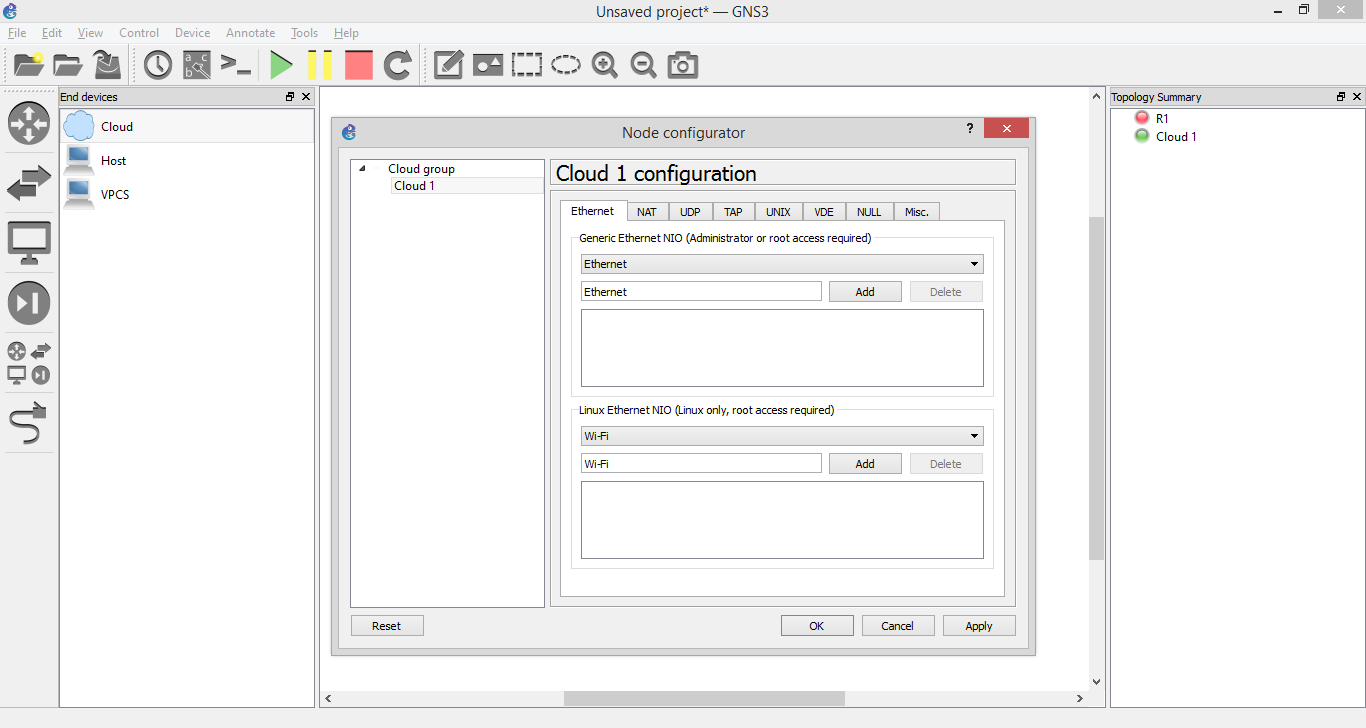
# Connecting Devices and Assigning IP Address To A Router

* First we need to setup our router for that we need to select,
* Edit>>Preferences>>IOS Routers
* Download IOS images from <http://srijit.com/working-cisco-ios-gns3/>
* Browse and select c3745
* Click apply and press ok

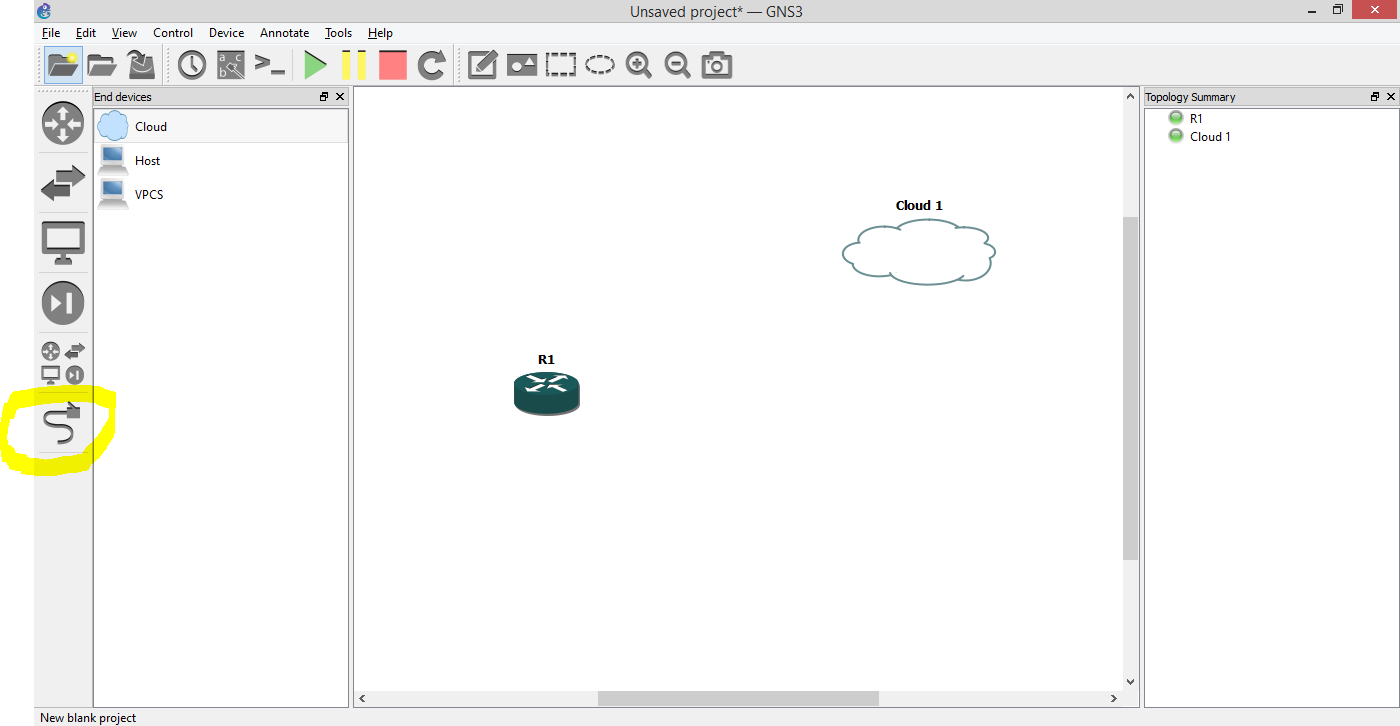
Now drag the router and cloud outside in the middle, It will look like this



* Now, Right click on the cloud and click on configure

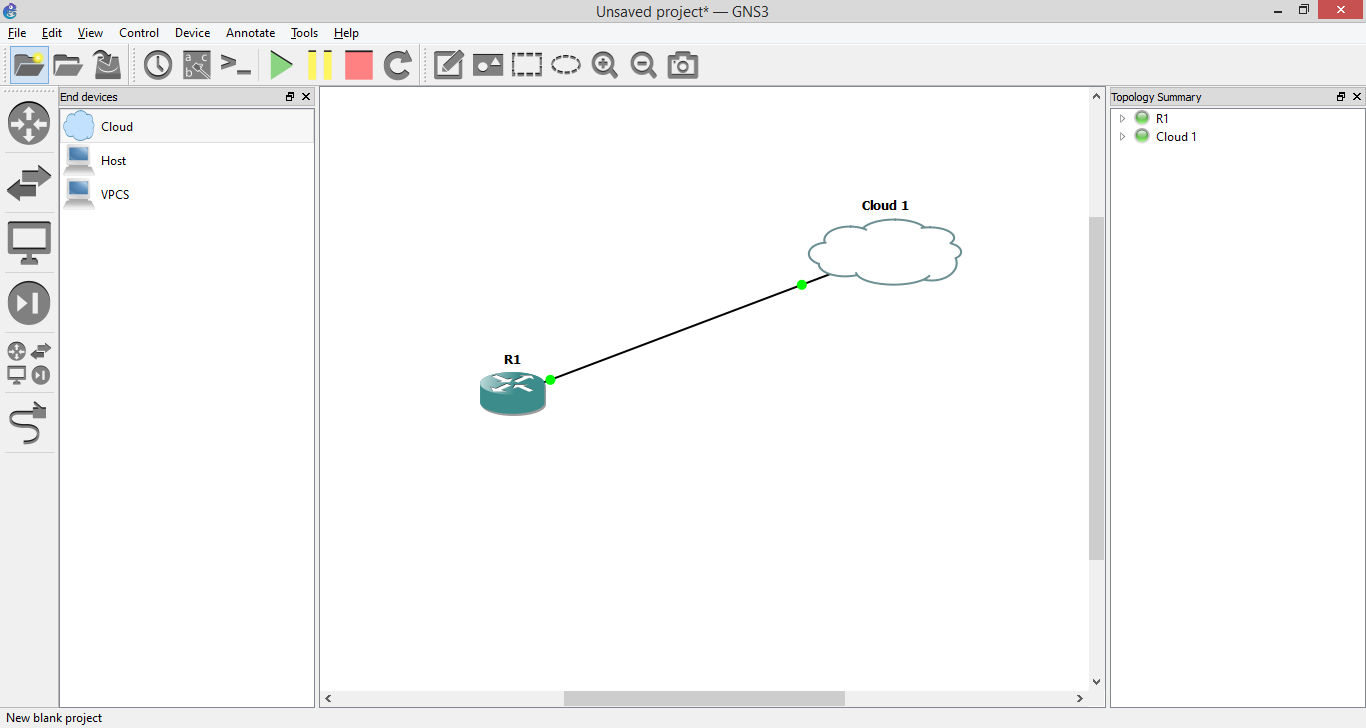


* Choose Ethernet,click Add. After that, click apply and click ok
* Click on the highlighted area (shown in below image)



* Connect both the devices with the wire choosing the first option you get while connecting them through wire.
* Now, right click on the router and click start

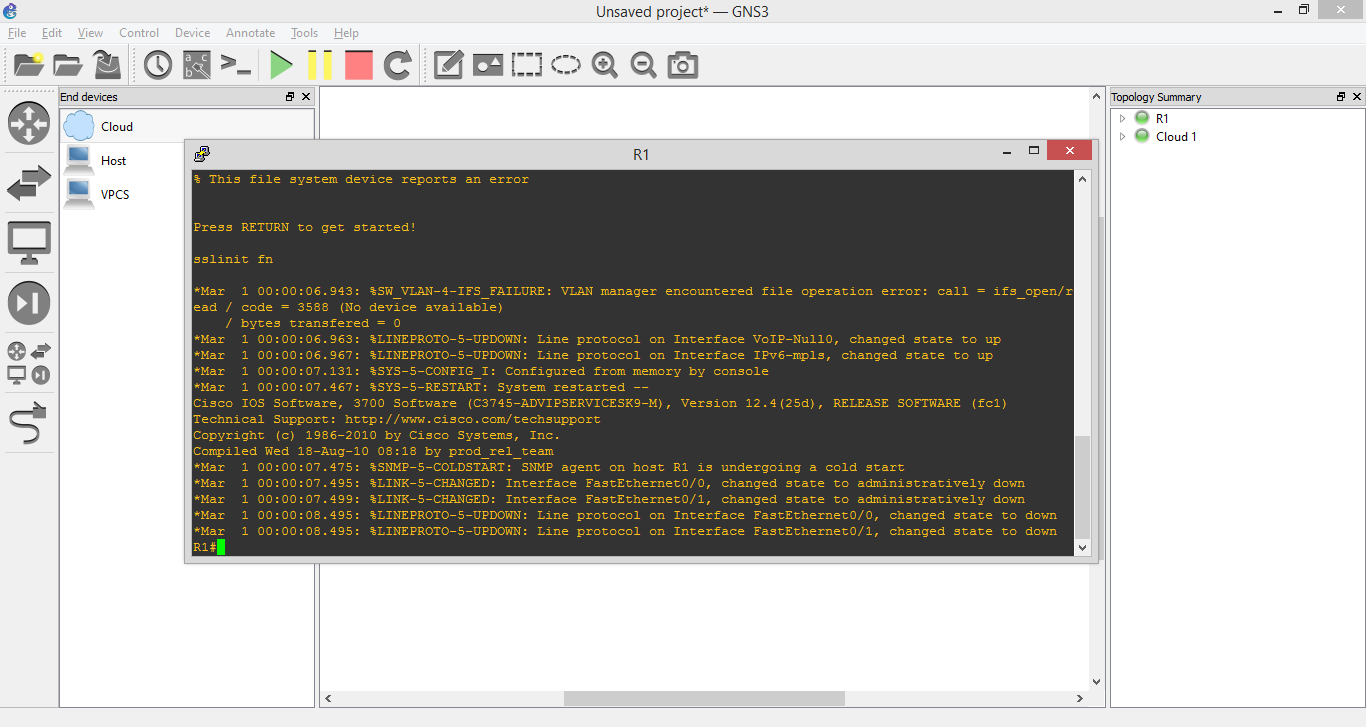
It will look like,



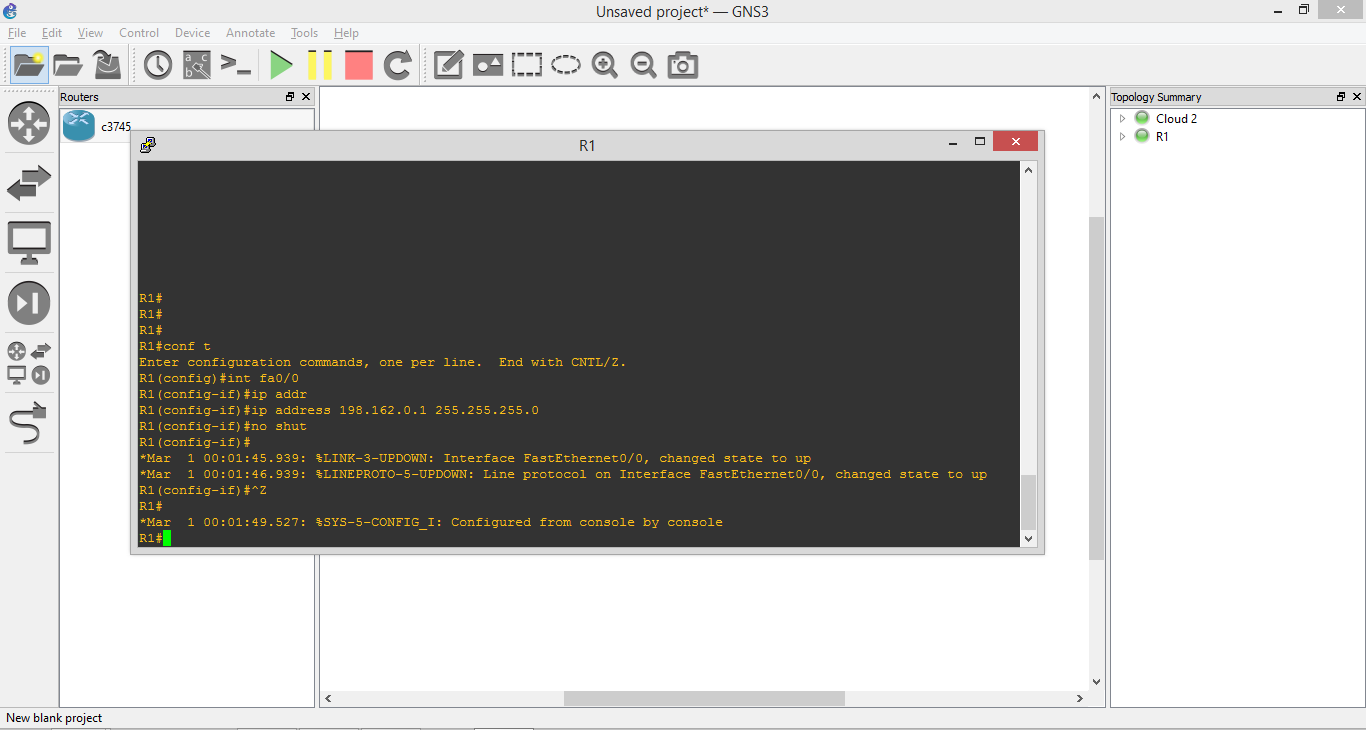
Now it is the time to configure the router

* Right click on the router and select console from the list

It will look like



* Assignin g router an IP address



# TELNET LIBRARY

Telnet is a protocol that allows you to connect to remote computers (called hosts) over a TCP/IP network (such as the internet). Using telnet client software on your computer, you can make a connection to a telnet server (that is, the remote host). Once your telnet client establishes a connection to the remote host, your client becomes a virtual terminal, allowing you to communicate with the remote host from your computer.

# How to enable telnet in your windows system ?

1. Click on a Start button
2. Start typing Control so that Control Panel desktop app will show
3. Switch to a Category view
4. Click on Programs
5. Click Windows features on or off
6. Select Telnet Client
7. Click OK
8. Restart a computer.

# Introduction to Netmiko library

The purposes of this library are the following:

* Successfully establish an SSH connection to the device
* Simplify the execution of show commands and the retrieval of output data
* Simplify execution of configuration commands including possibly commit actions

# Installing Netmiko

* Open anaconda prompt
* Activate your virtual environment
* Type this command
  + Pip install netmiko

You are good to go now

# PARAMIKO LIBRARY

Paramiko is a Python (2.7, 3.4+) implementation of the SSHv2 protocol [[1]](http://www.paramiko.org/#id2), providing both client and server functionality. While it leverages a Python C extension for low level cryptography ([Cryptography](https://cryptography.io/)), Paramiko itself is a pure Python interface around SSH networking concepts.

# INSTALLING PARAMIKO

* Open anaconda prompt
* Activate your virtual environment
* Type this command
  + conda install -c anaconda paramiko

You are good to go now

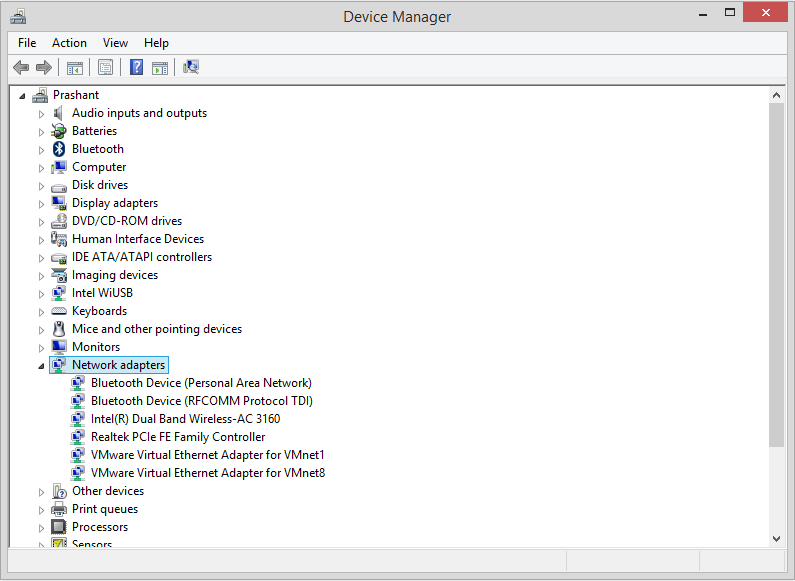
# Adding Loopback Interface

In order to ping the router from our computer console, we have to introduce a loopback interface here.

Steps to follow to add a loopback interface

* Open Device Manager in your system and select network adapters tab.

It will look like,

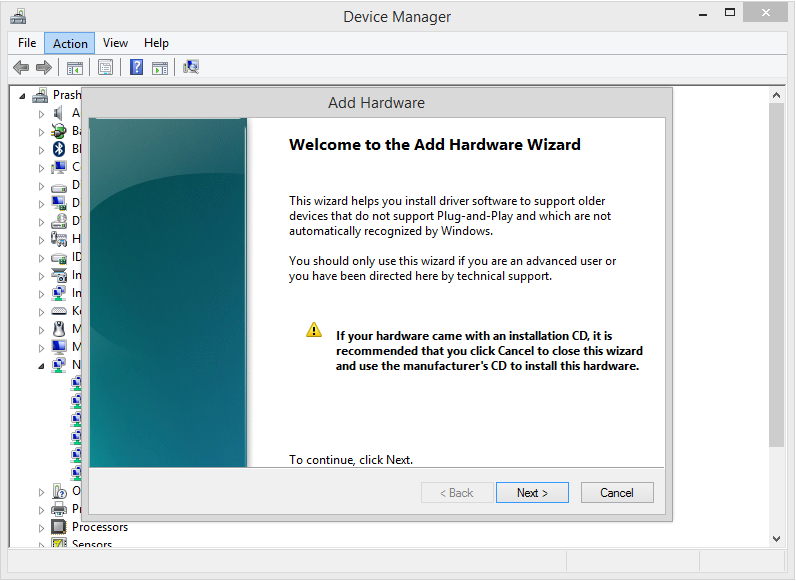


* Now click on the action from above menubar and choose add hardware legacy from available options.

It will look like,

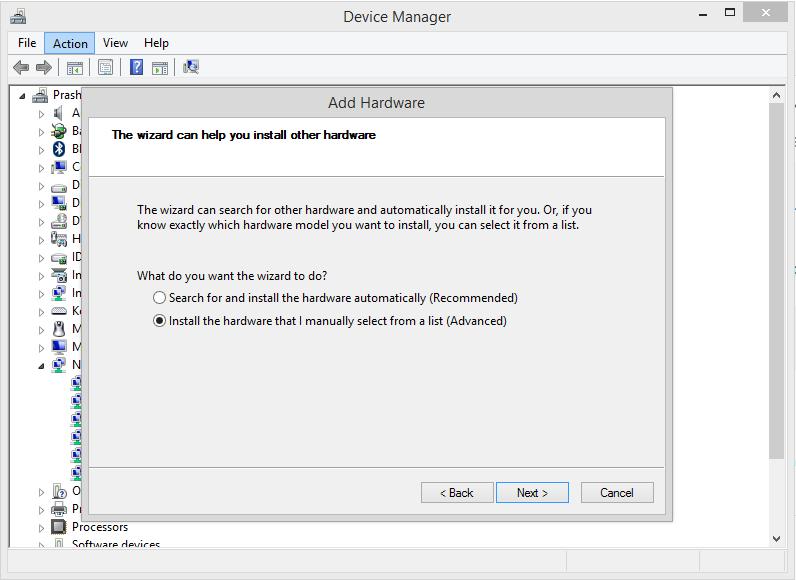


* You will get a Add Hardware window like this,

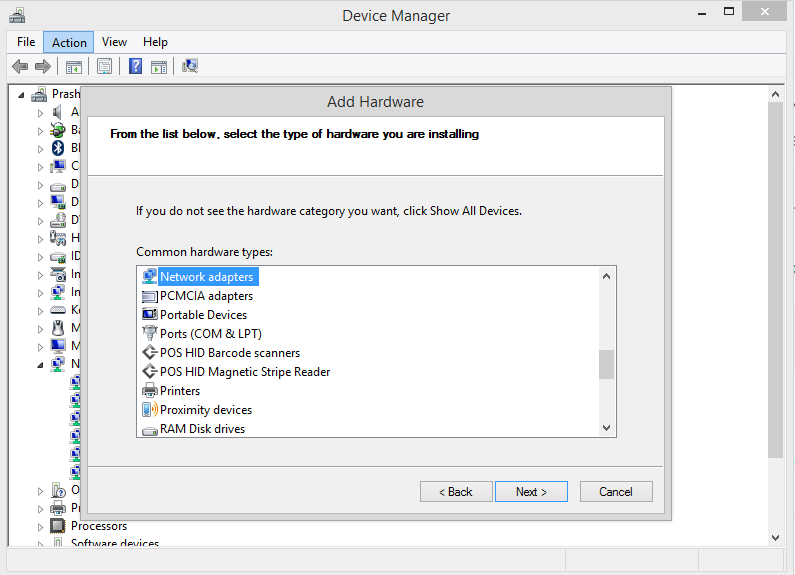


* Click on the next button and then choose,

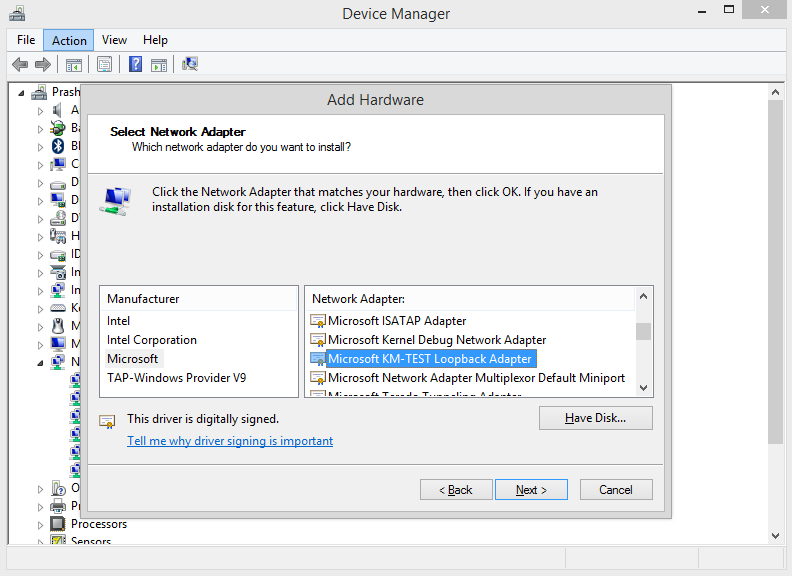
Install the hardware that I manually select from a list



* After that, click on next and choose the network adapter option from the list

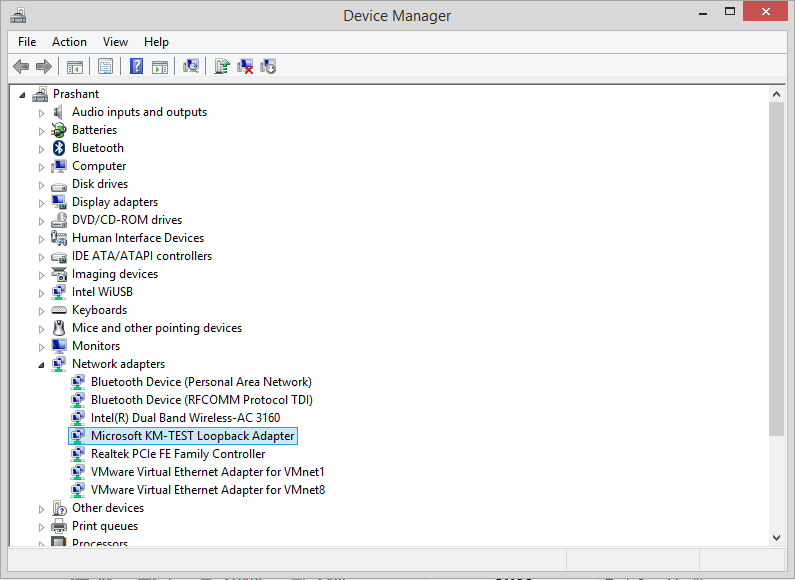


* Click next and select the options as in the picture below.

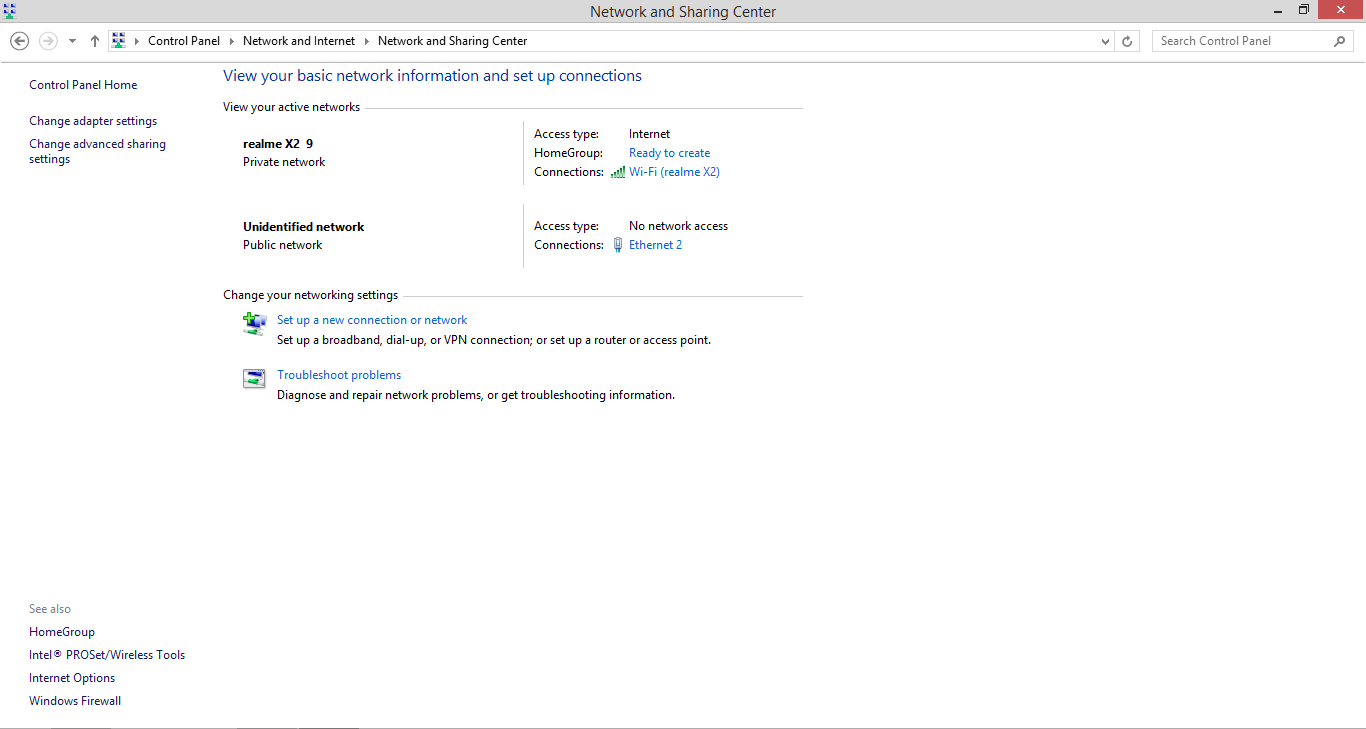


* Click next and it will be installed after few minutes.

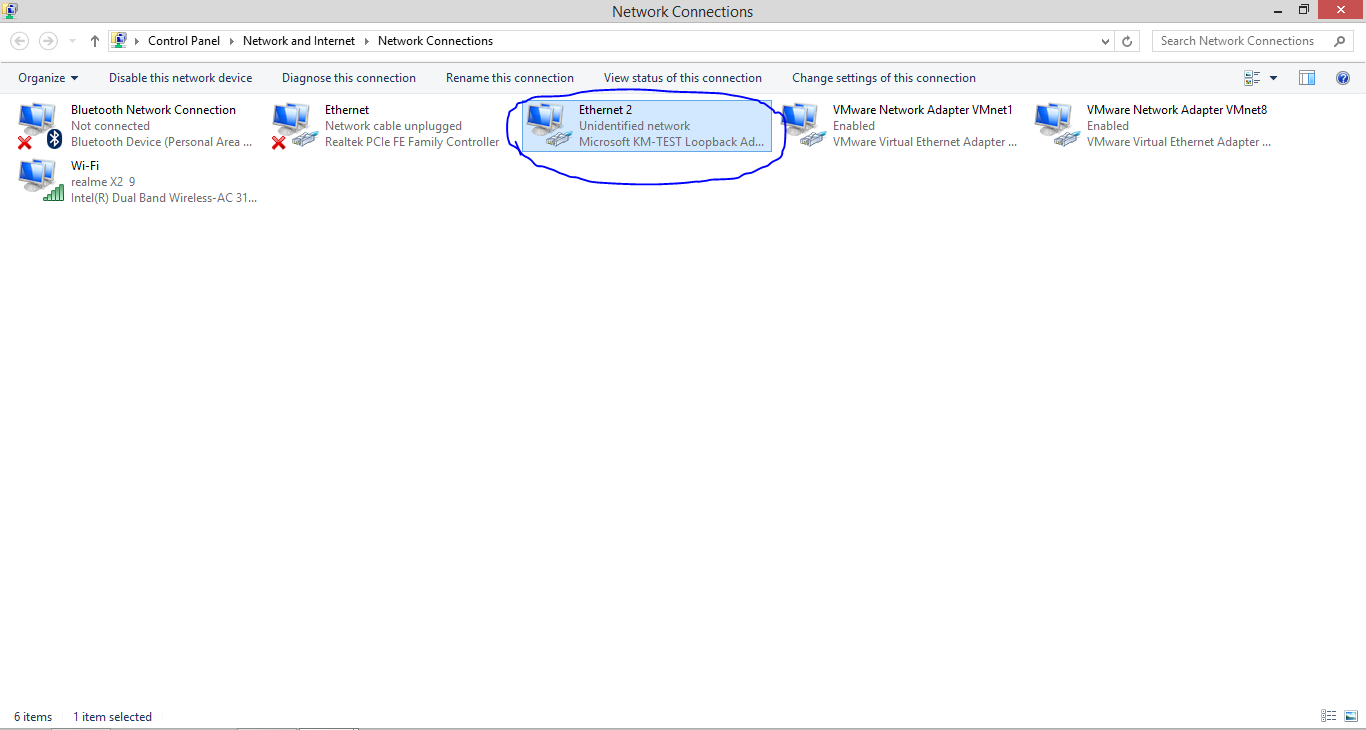
After installation you can see it in your network adapters list



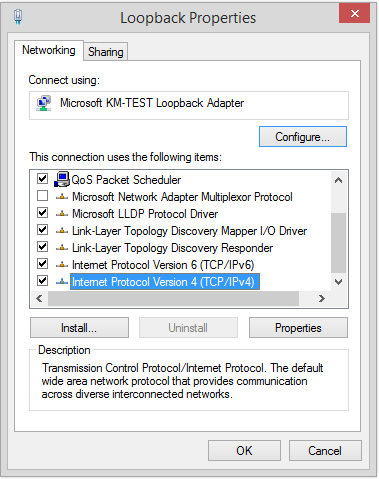
* Now, open network and sharing center on your computer



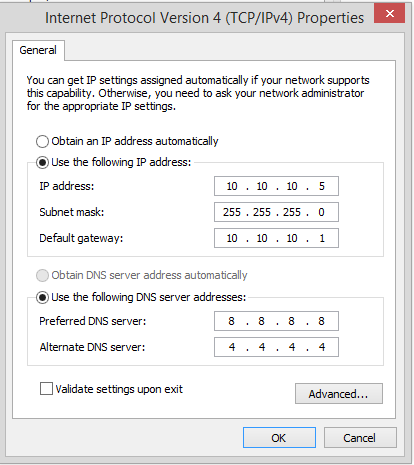
* Choose change adapter setting from the above left corner and there you will see that your loopback is present there



* Right click on it and choose properties from the available options.



* Double click on the highlighted option(TCP/Ipv4) and configure it like below



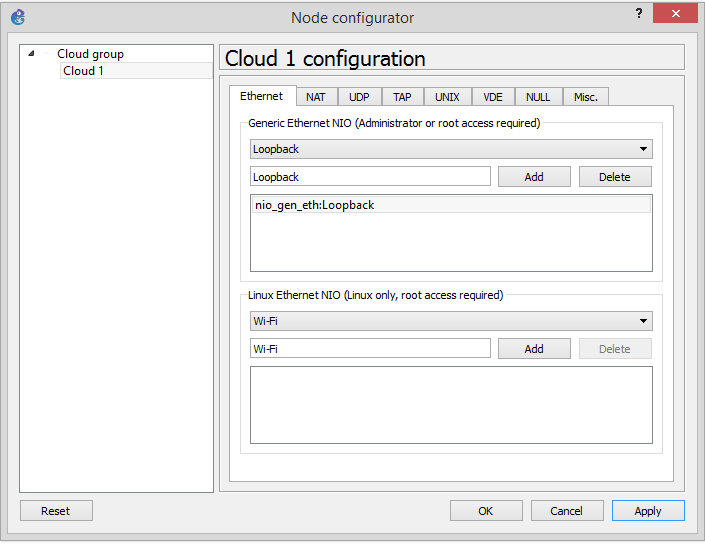
* Click OK

Your loopback has been created successfully.

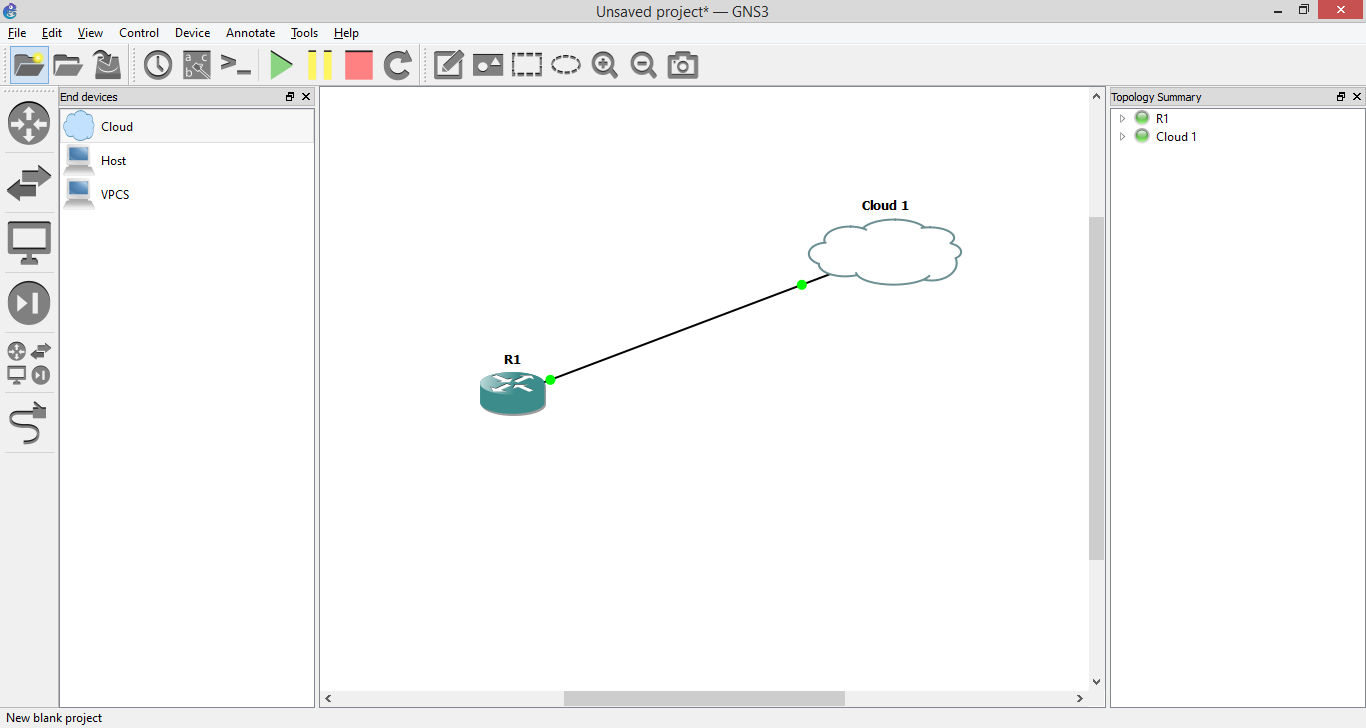
Now, It’s the time to check that can we ping our router from our system or not.

Let’s get back to the GNS3.

* In GNS3, drag router and cloud
* Right click on the cloud and choose configure

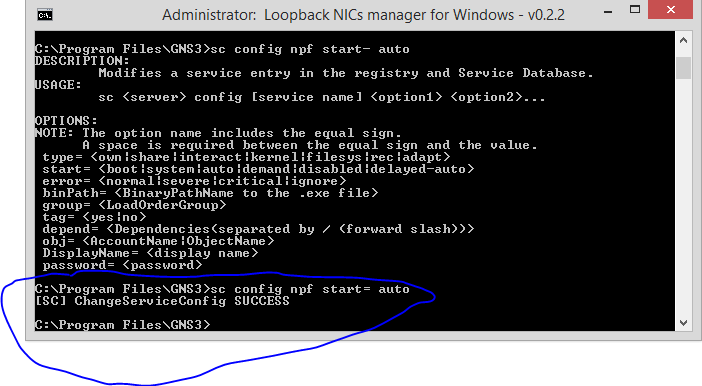


* Choose Loopback in Generic Ethernet NIO and click add
* Click on apply and then Ok
* Now connect router and cloud through a wire, after that start the router



* Now, open cmd as an admin
* Change the directory to GNS3 directory
  + In my case it is c:\Program Files\GNS3
* Now we need to change the ServiceConfig, for that type this command
  + sc config npf start= auto

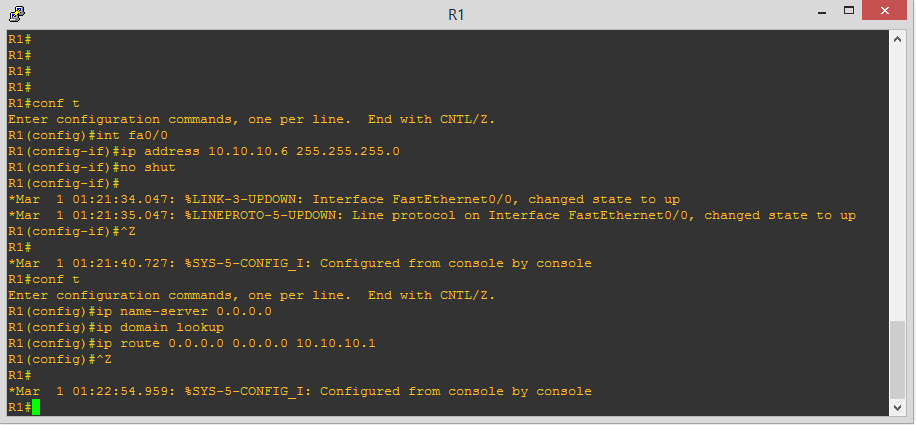
It will look like,



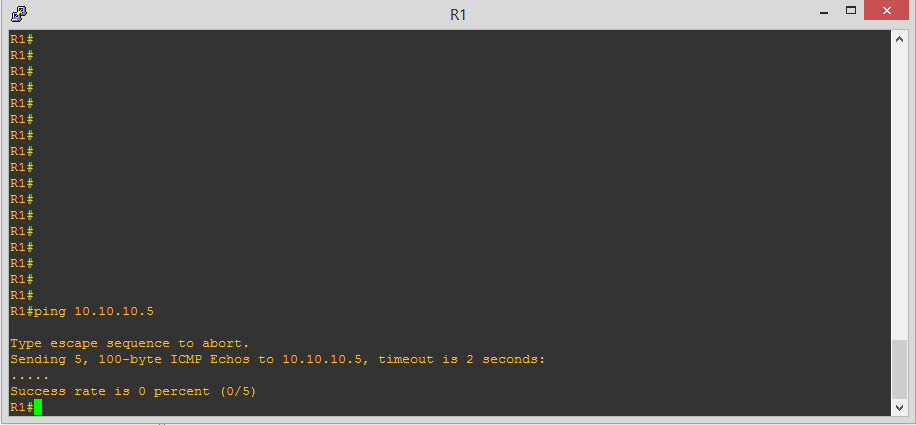
* Now, type in your cmd
  + loopback-manager.cmd
* Choose option 4 to remove all loopback interfaces if installed
* Choose option 2 to install a new loopback interface
* Choose option 6 to quit
* Reboot your computer and use the newly installed loopback

# Pinging Router from our system cmd

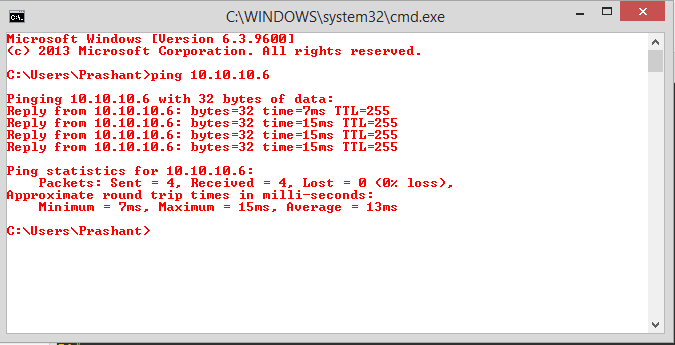
* In GNS3, right click on your router and open console
* Follow these steps as shown in pic



* Pinging system through router



* Pinging router through system



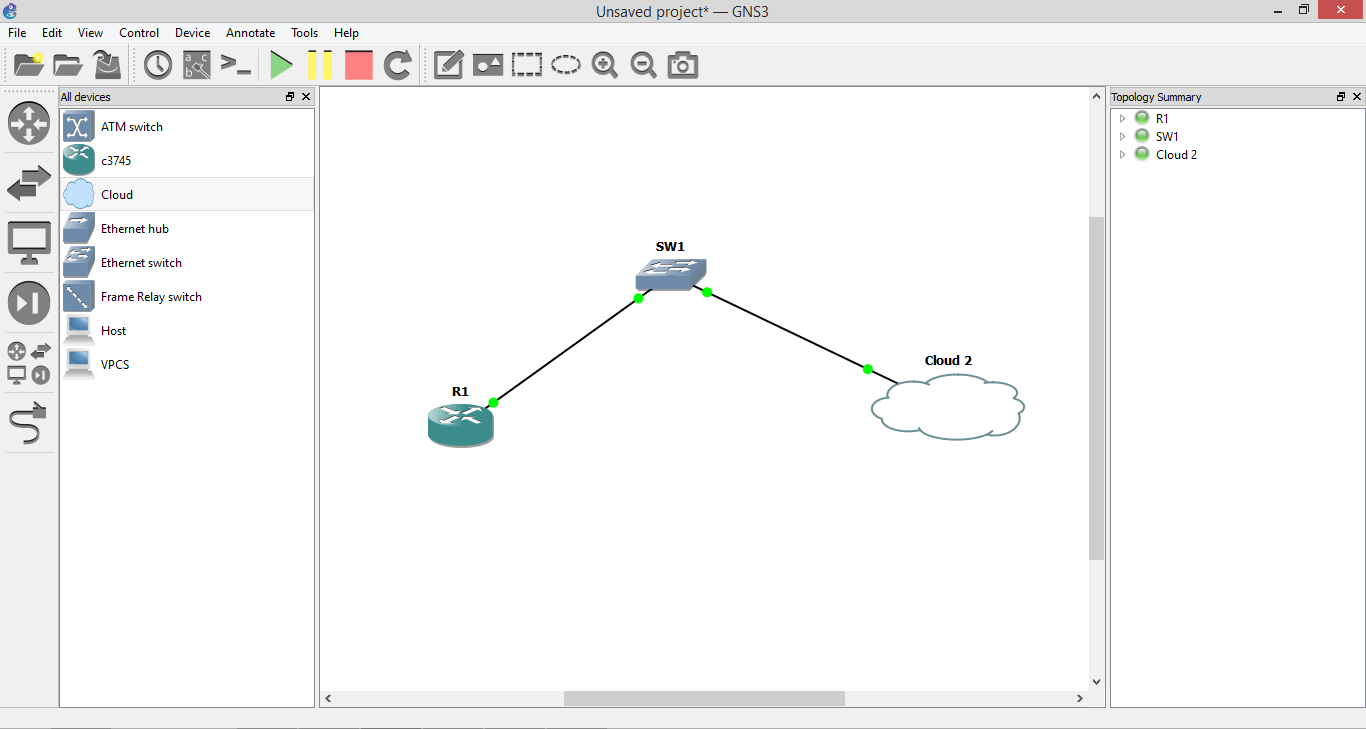
# First Script

Before writing our script we need to download putty for connection purpose

Link to download putty

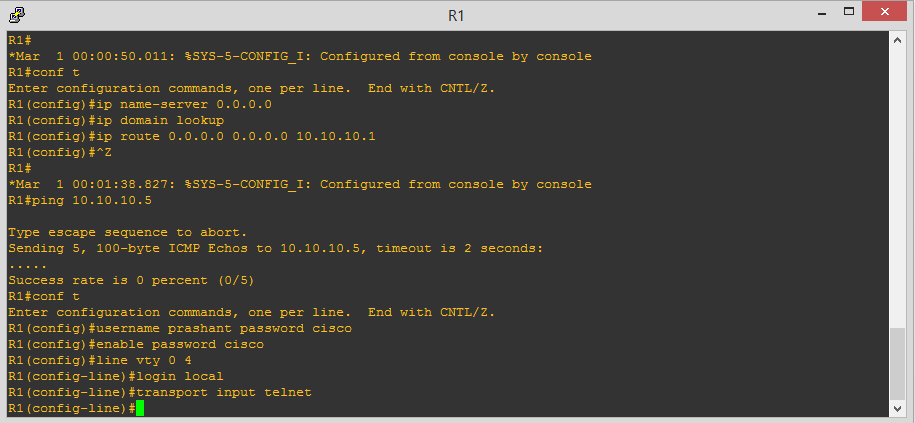
* + <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

Let’s have a look at our GNS3

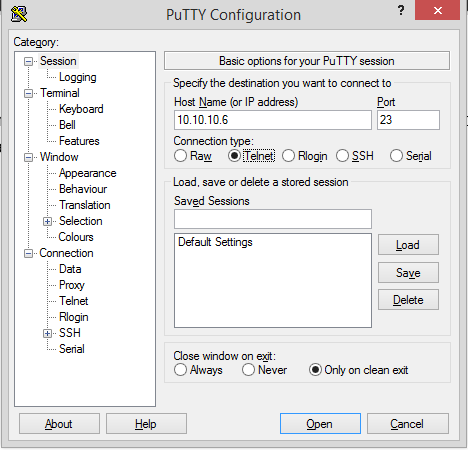


After assigning ip address and configuring router, set username and password so that we can have access to our router from our putty console.

Commands for doing the same



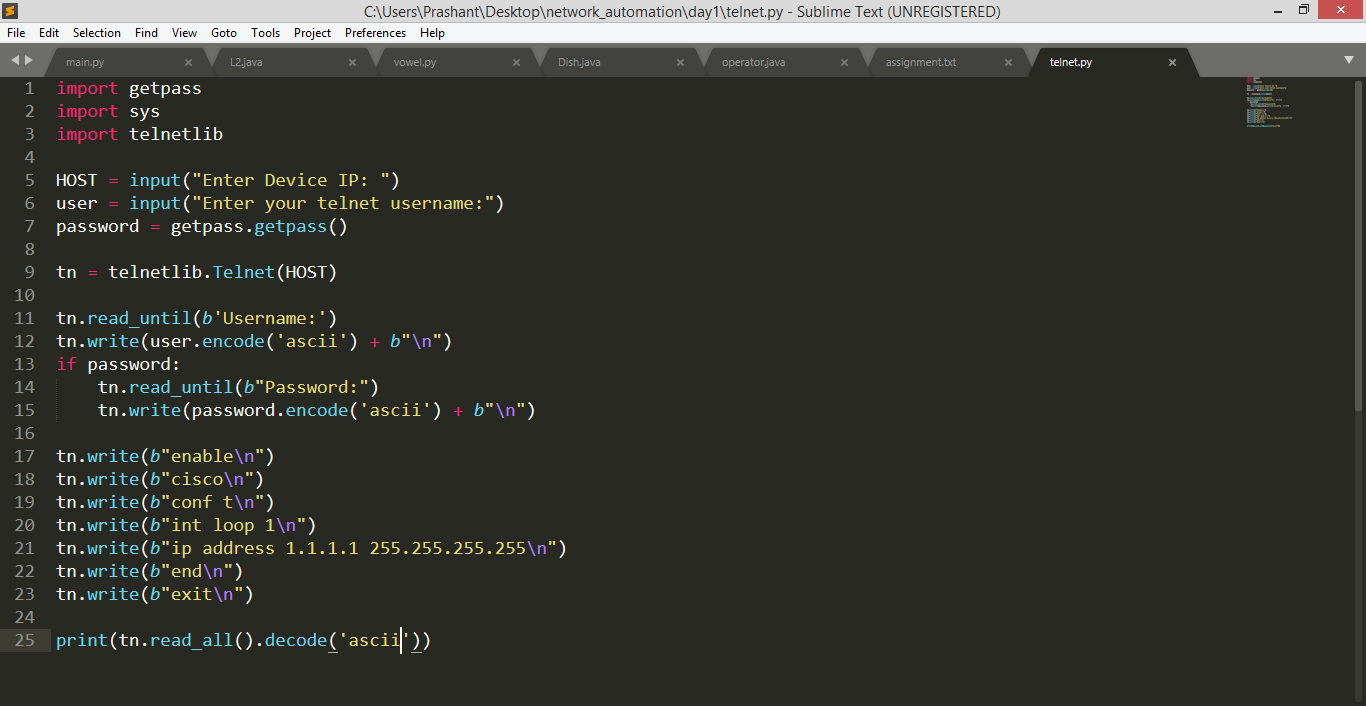
* Open putty from search menu in your windows and give your router ip address and choose telnet in connection type



* Click on open and it will redirect you to console

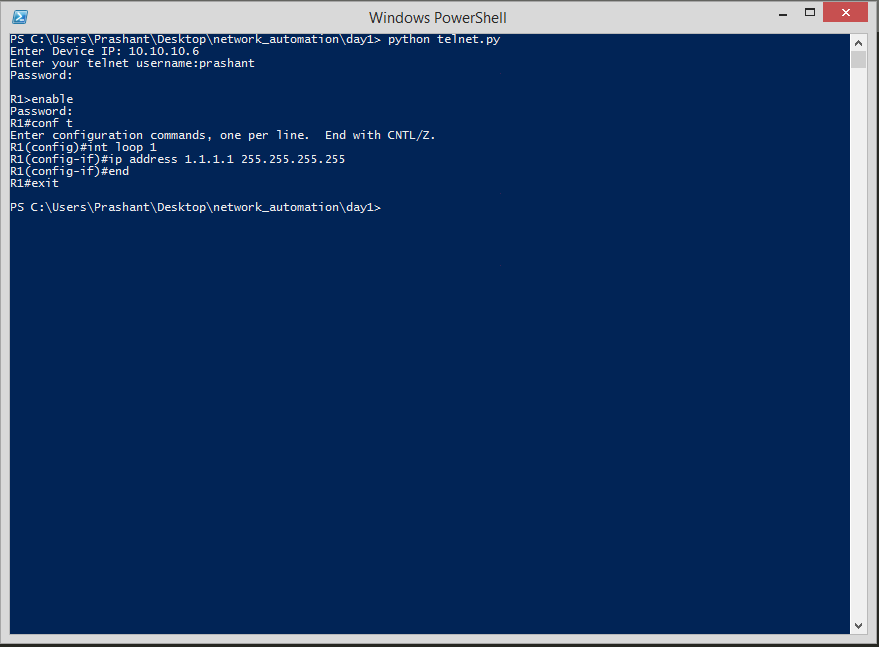


It is the time to write out first script

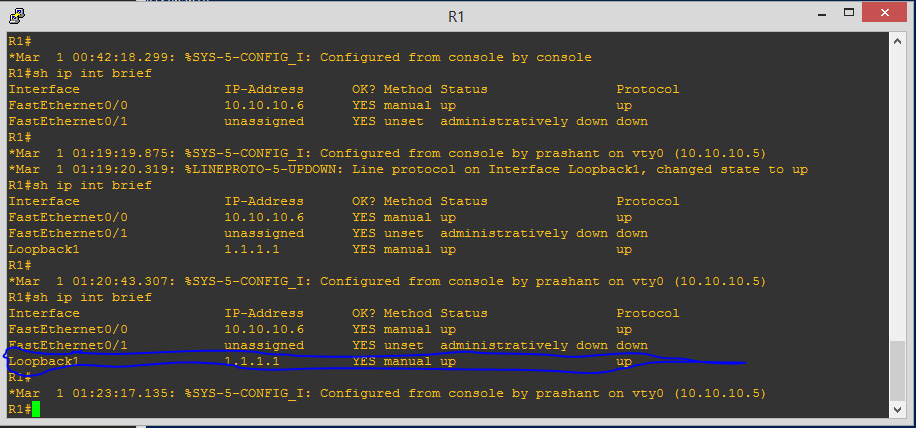


* Run this script on windows powershell, using command
  + python script\_name.py

It will give output like

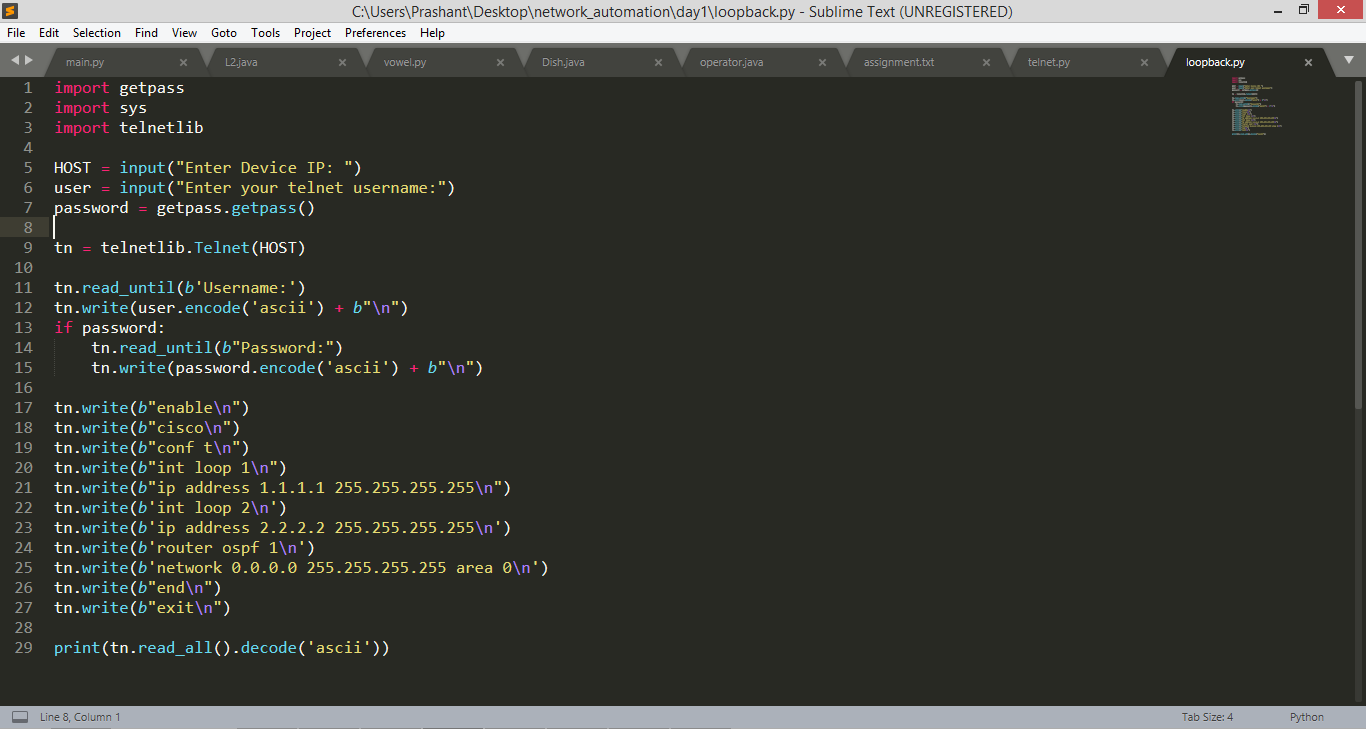


* It can be seen in our GNS3 console that we have configured router using our python script

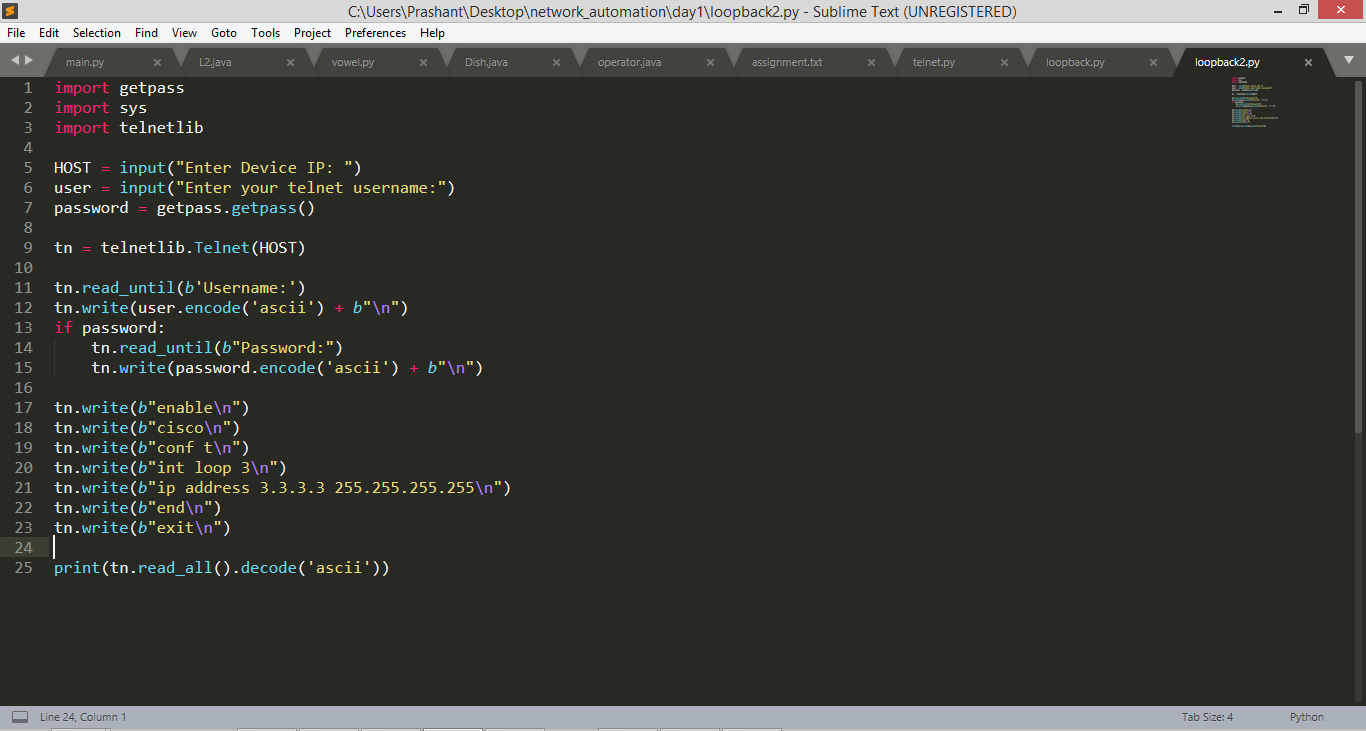


# Creating loopbacks and Vlan Using Script

Creating loopbacks:

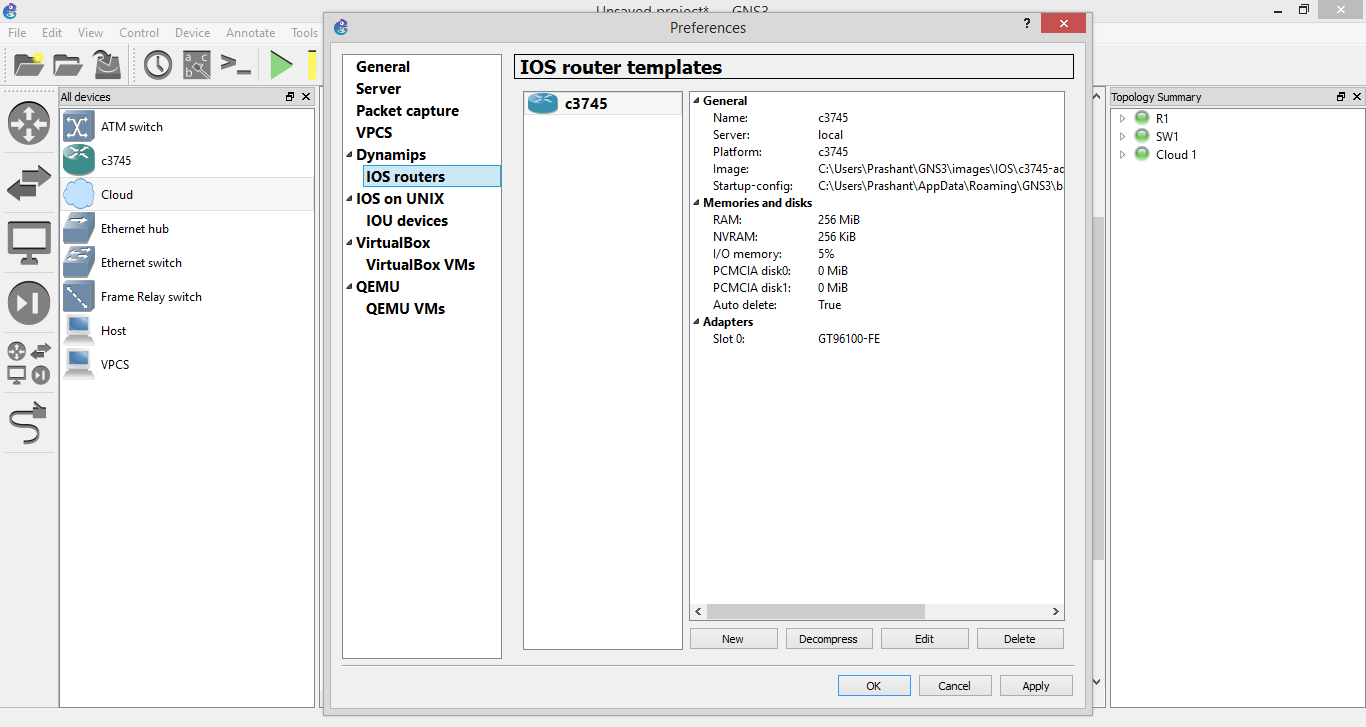


Creating another loopback

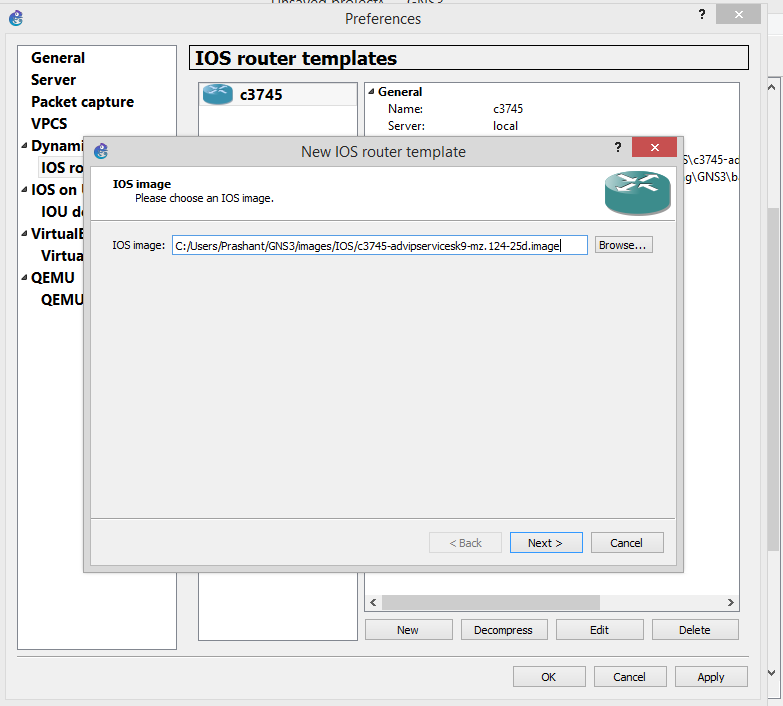


Uploading IOS for a Ethernet switch

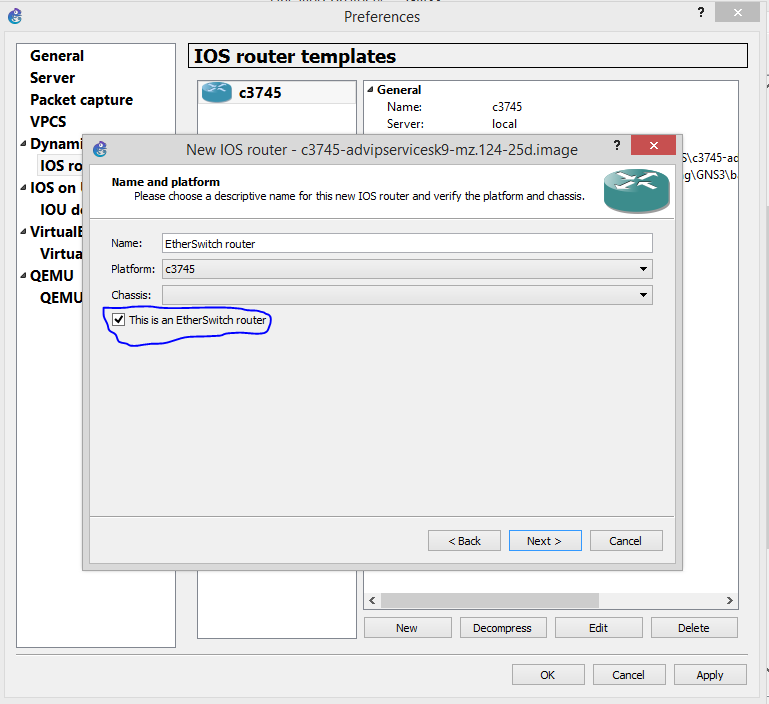
* GO to menubar
  + Edit>> preferences



* Then
  + Next>>browse>>select the ios file
  + Click next



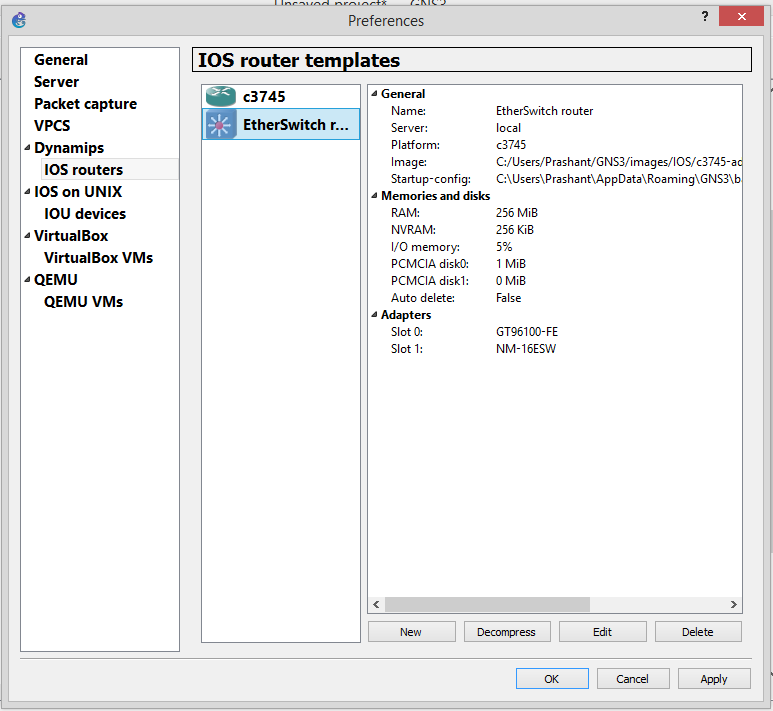
* Check the box



* Keep clicking next until you get fininsh button

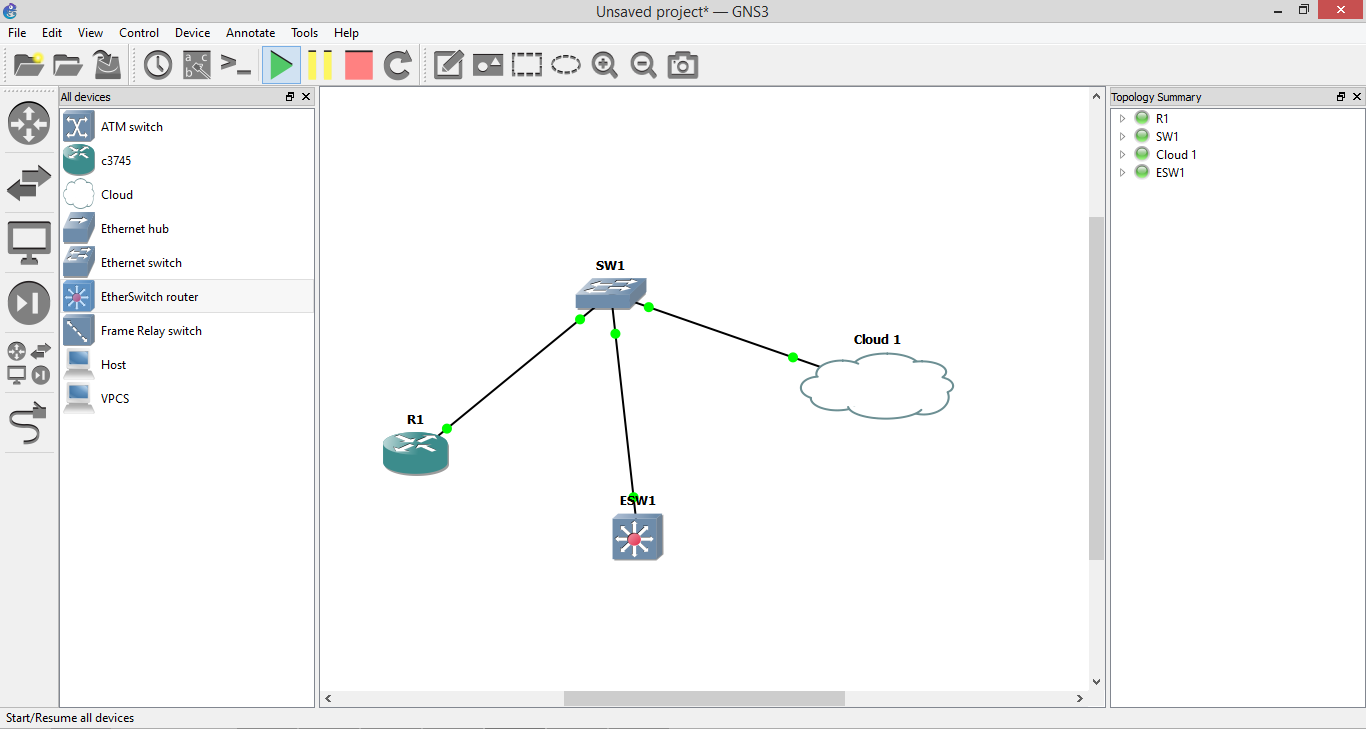
Now you are good to go.

* Now you can see your ethernet switch below your router

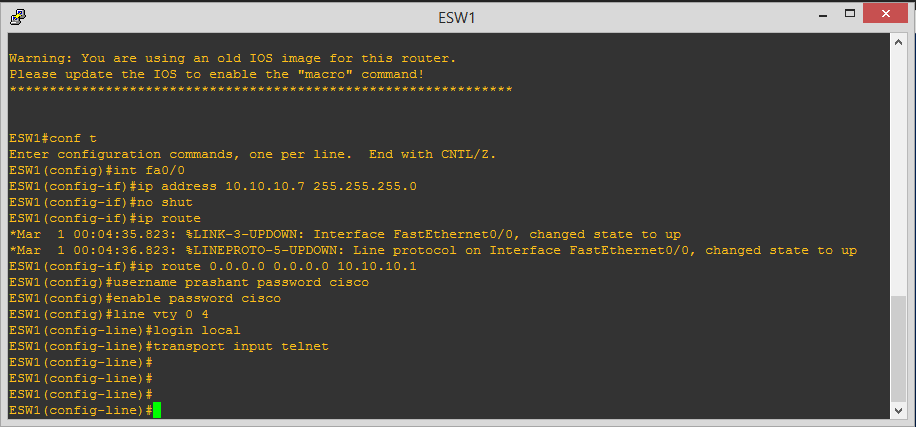


* Creating one more ethernet switch in gns3 to create a vlan

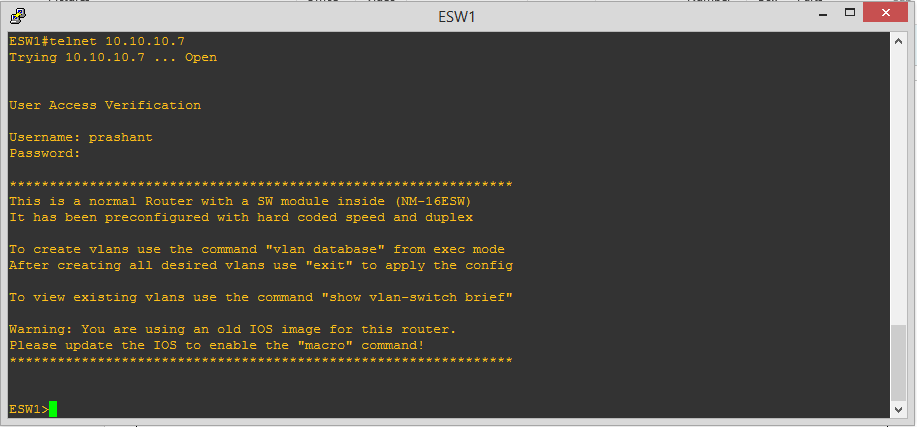
After connnecting it with switch it will look like



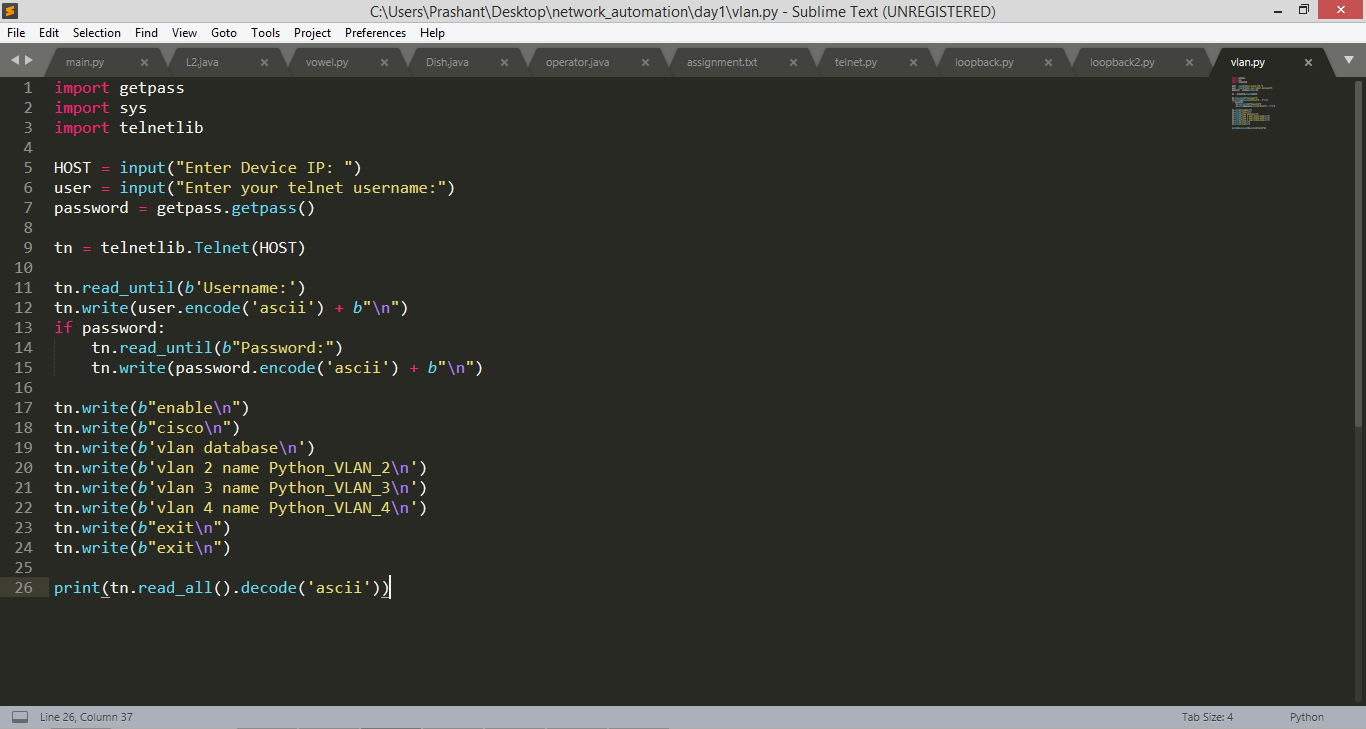
Configuring the new switch



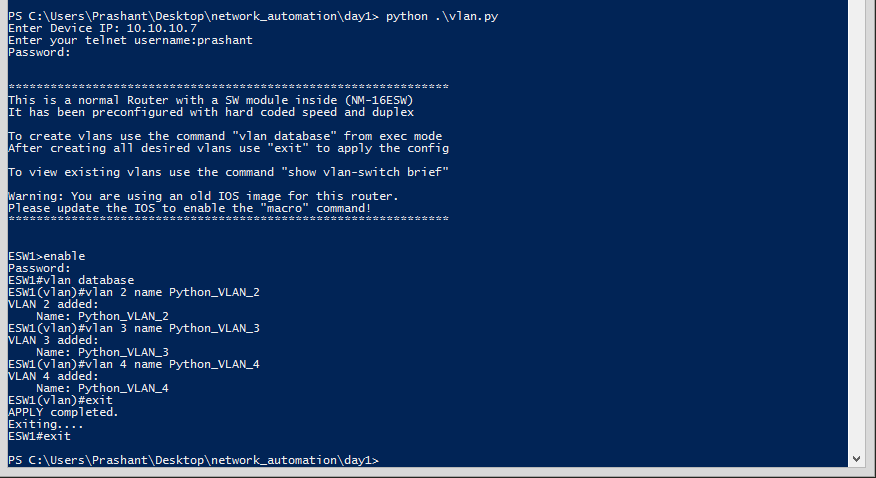
Logging in to our ethernet switch, so that we can use script on it to create vlan



Script for vlan

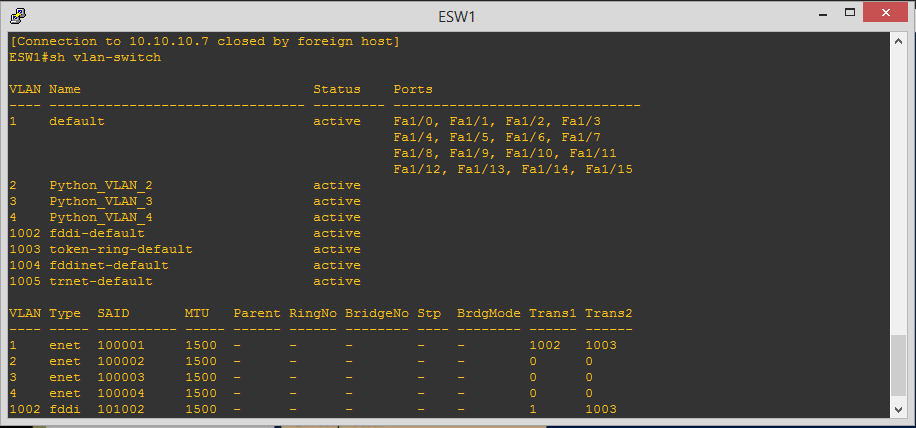


Output:

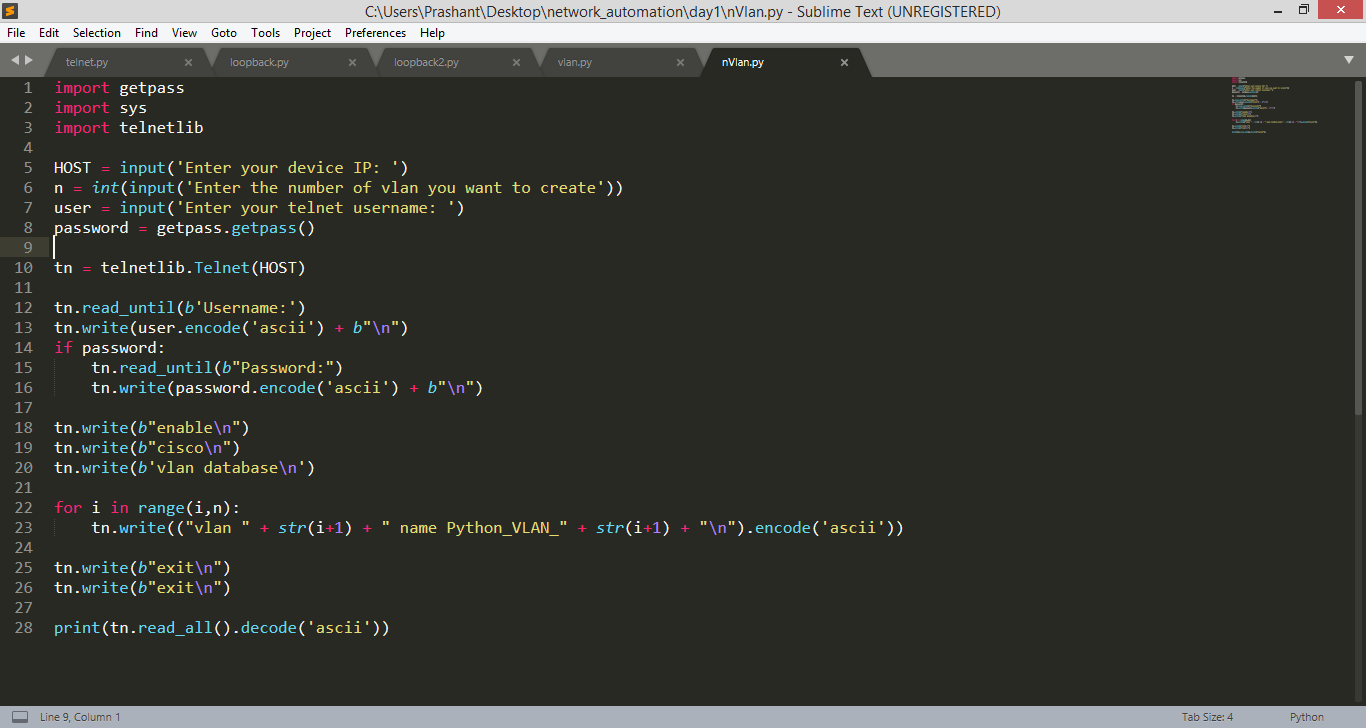


Checking vlan in putty using command

* + Sh vlan-switch

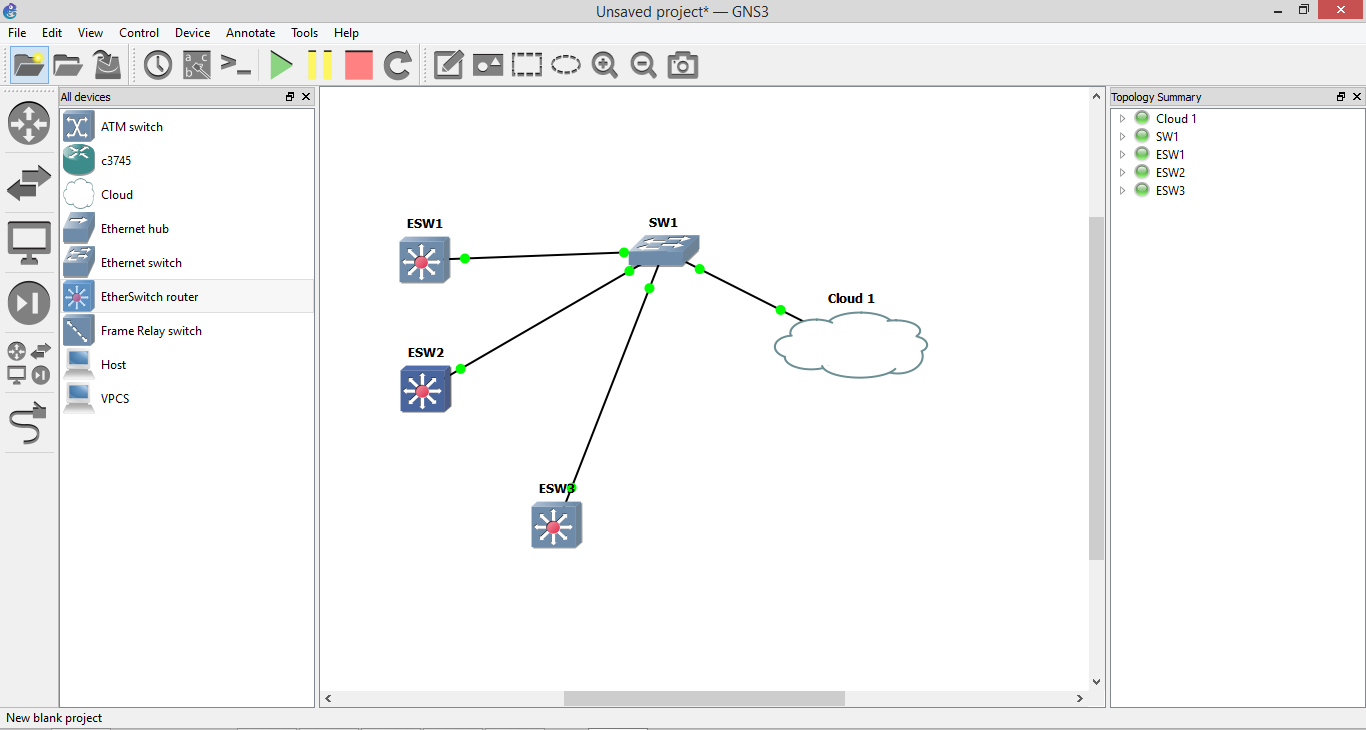


# Creating N number of Vlan using script



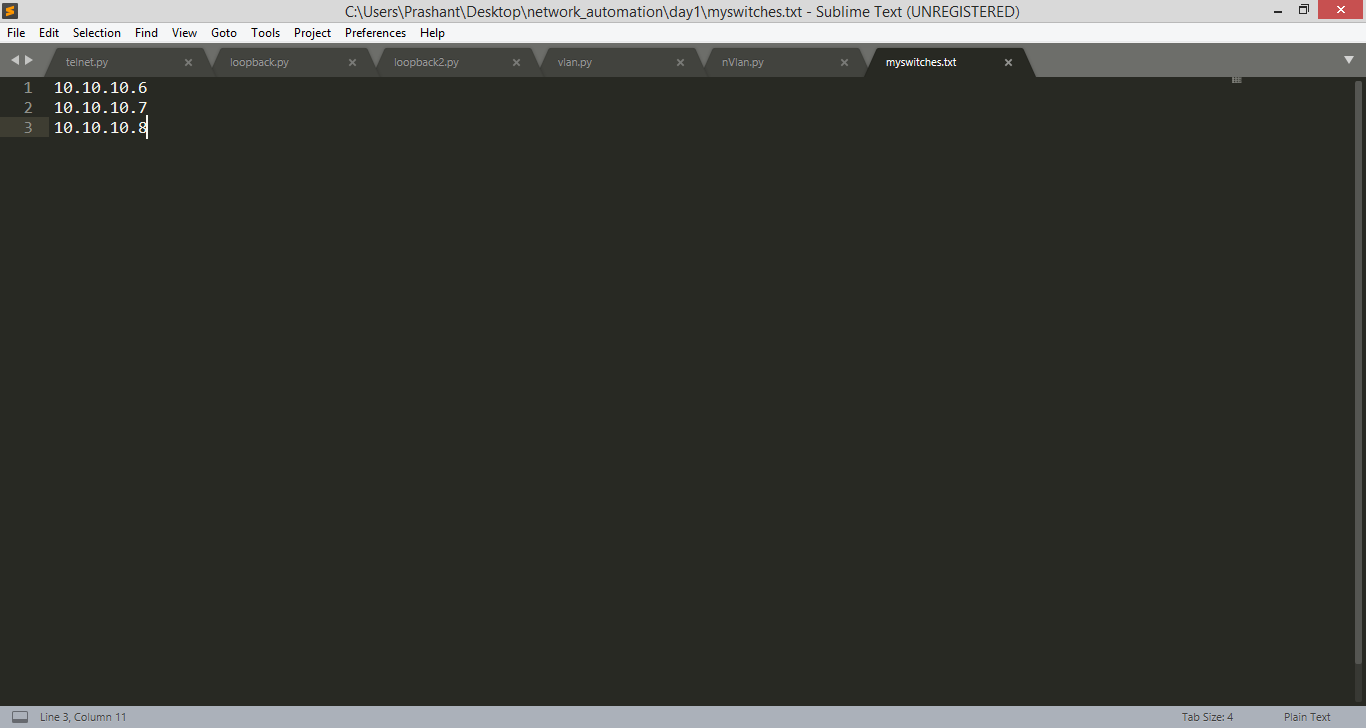
# Fetching device and update configuration

Our topology looks like:

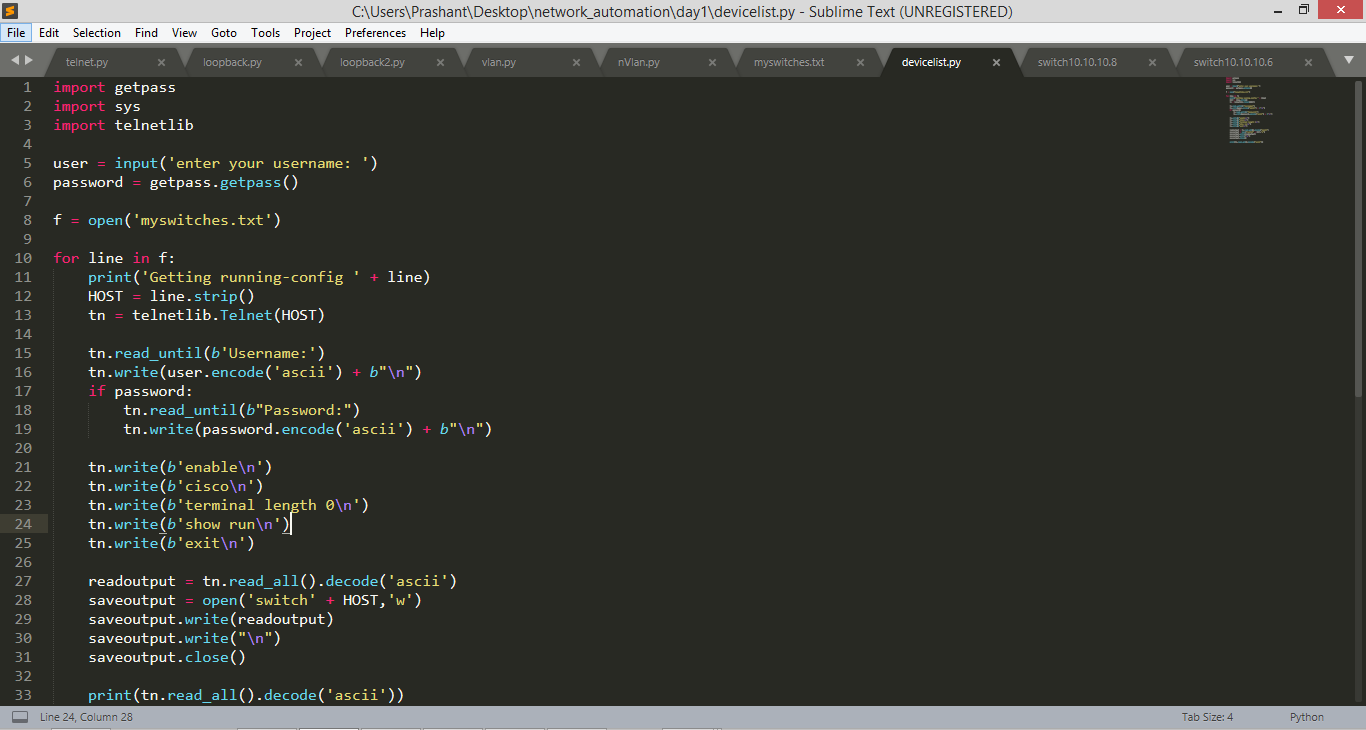


Configure all switches and store their ip address in a txt file so that we can access all switches one by one and can store their configuration in separate txt file.

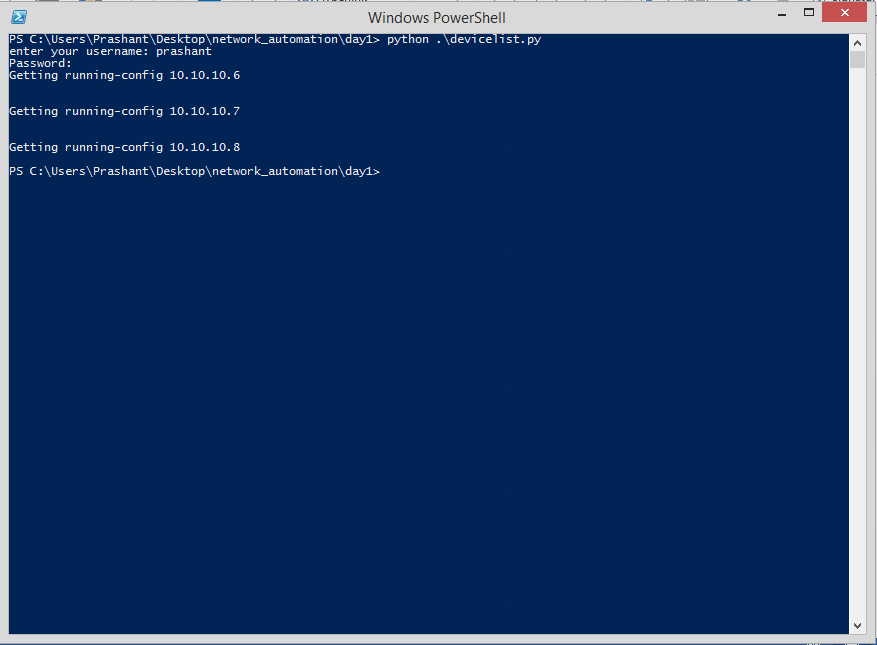
Our txt looks like:



Script:

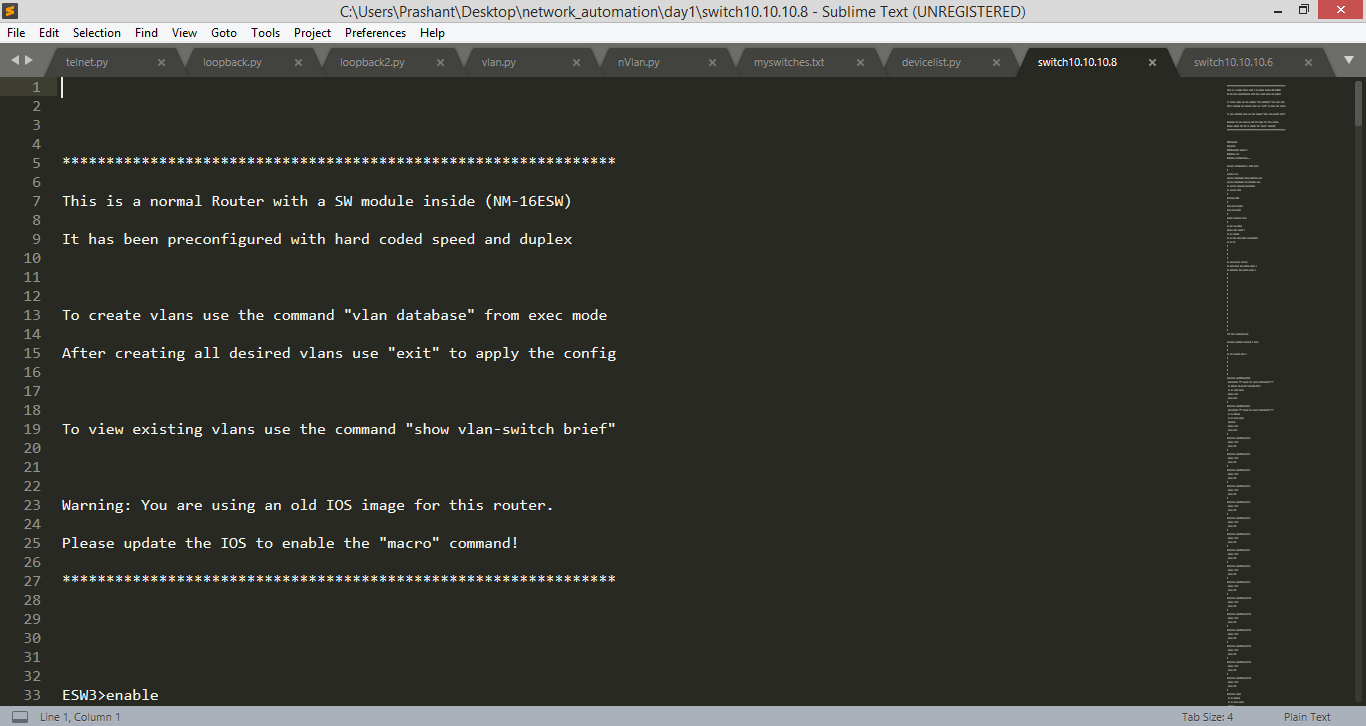


It will give a output like



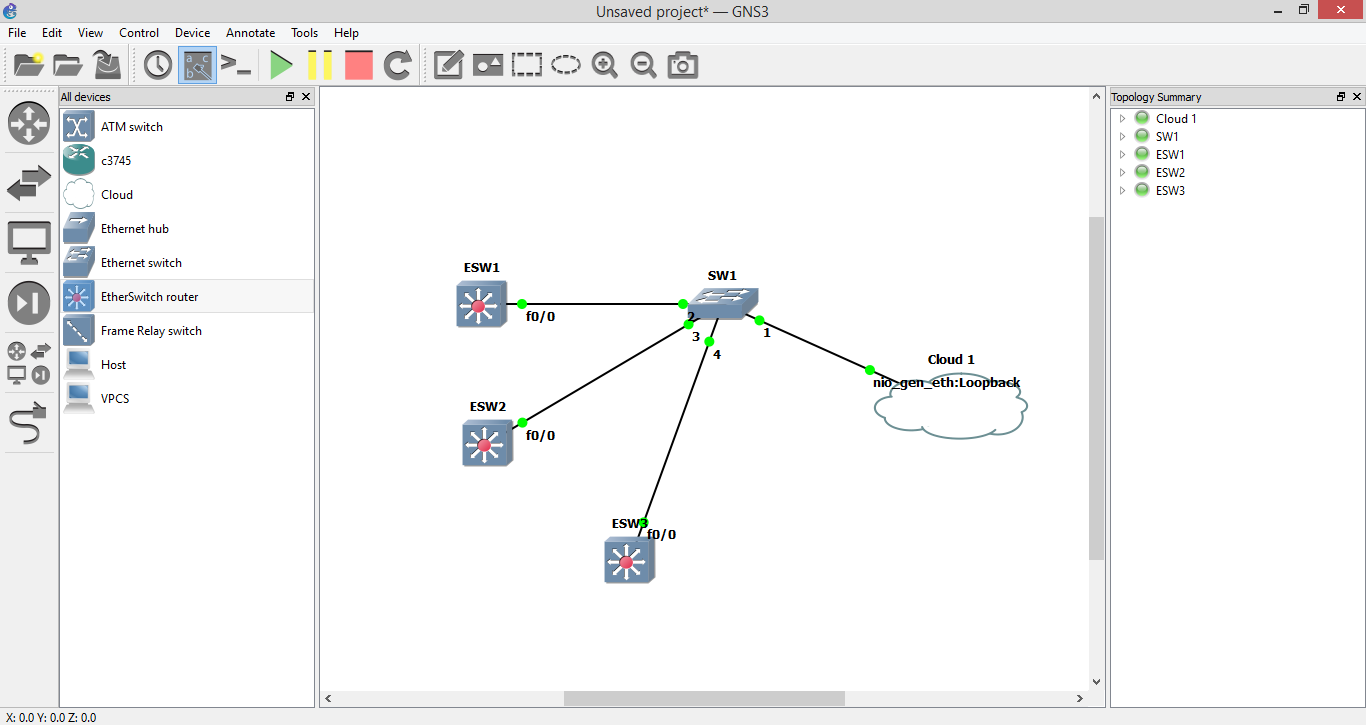
Now we have all the configuration in switch 10.10.10.\_ files

It contains 428 lines. Let’s see how it looks like

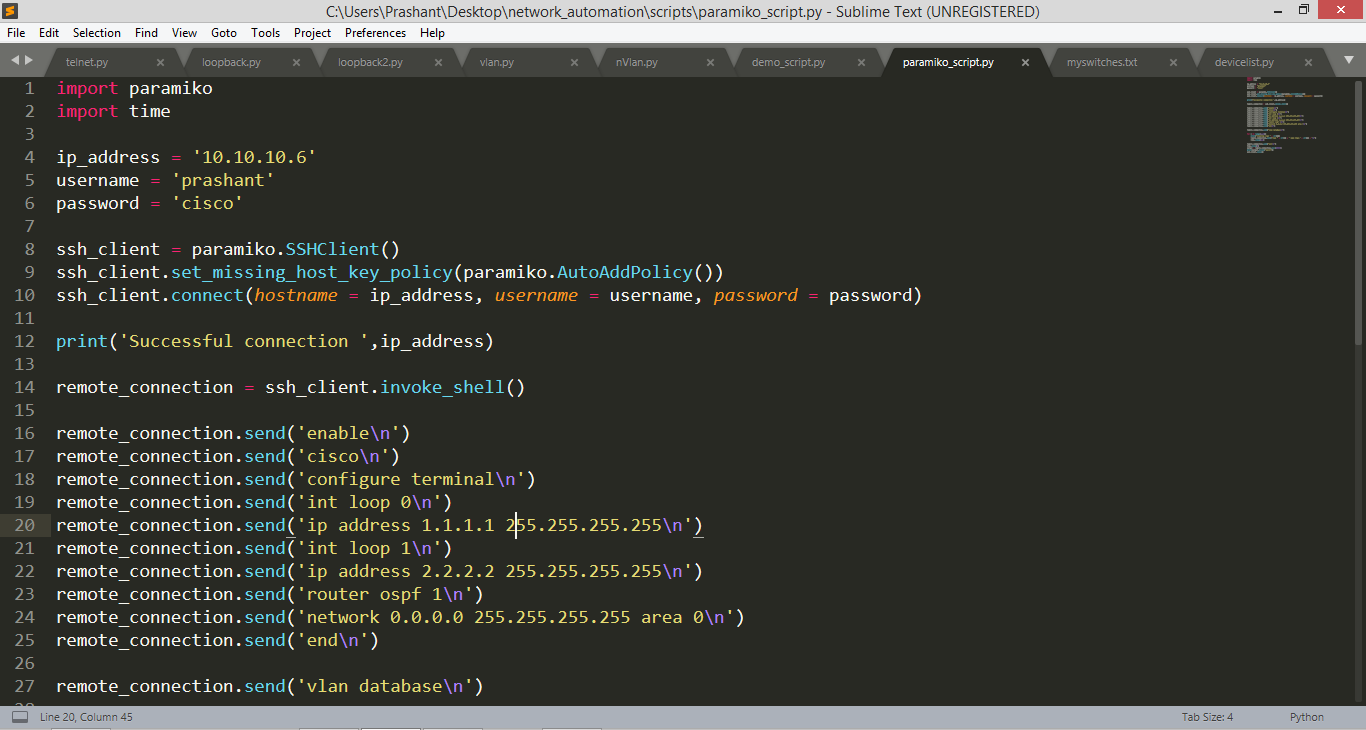


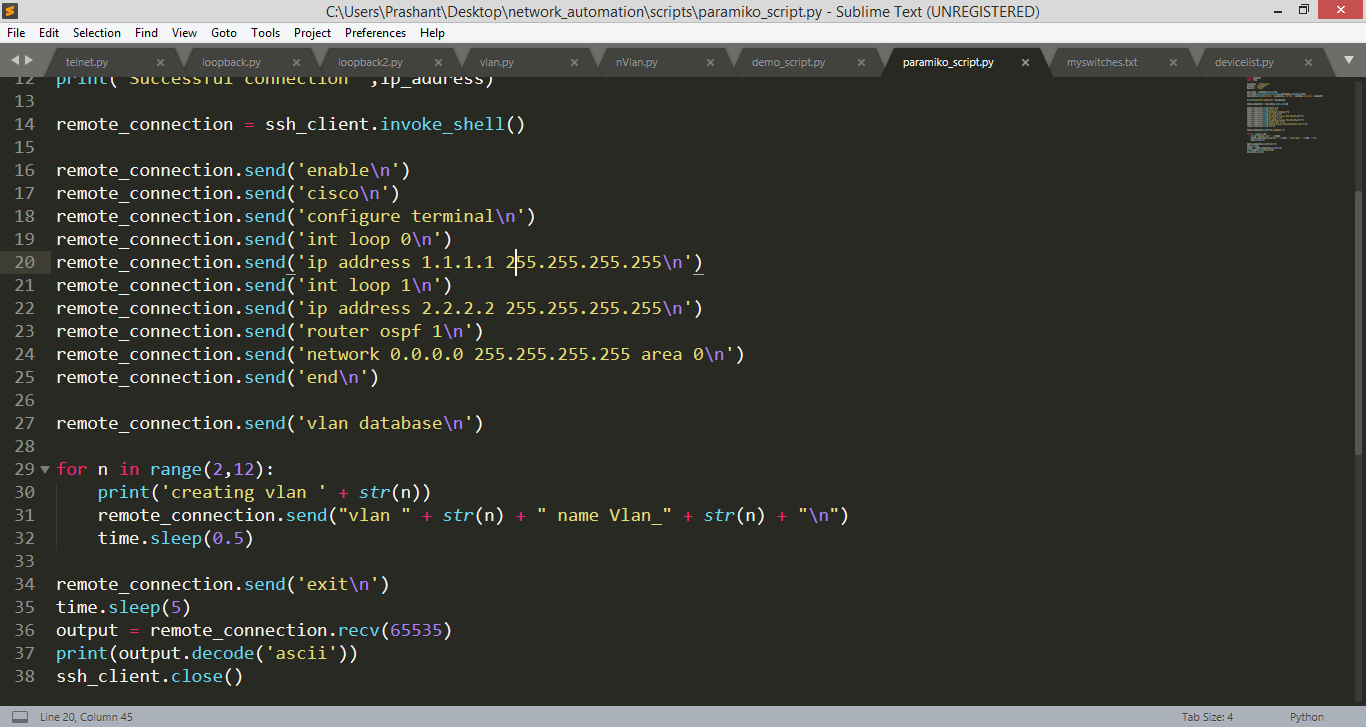
# Using paramiko

Our topology:



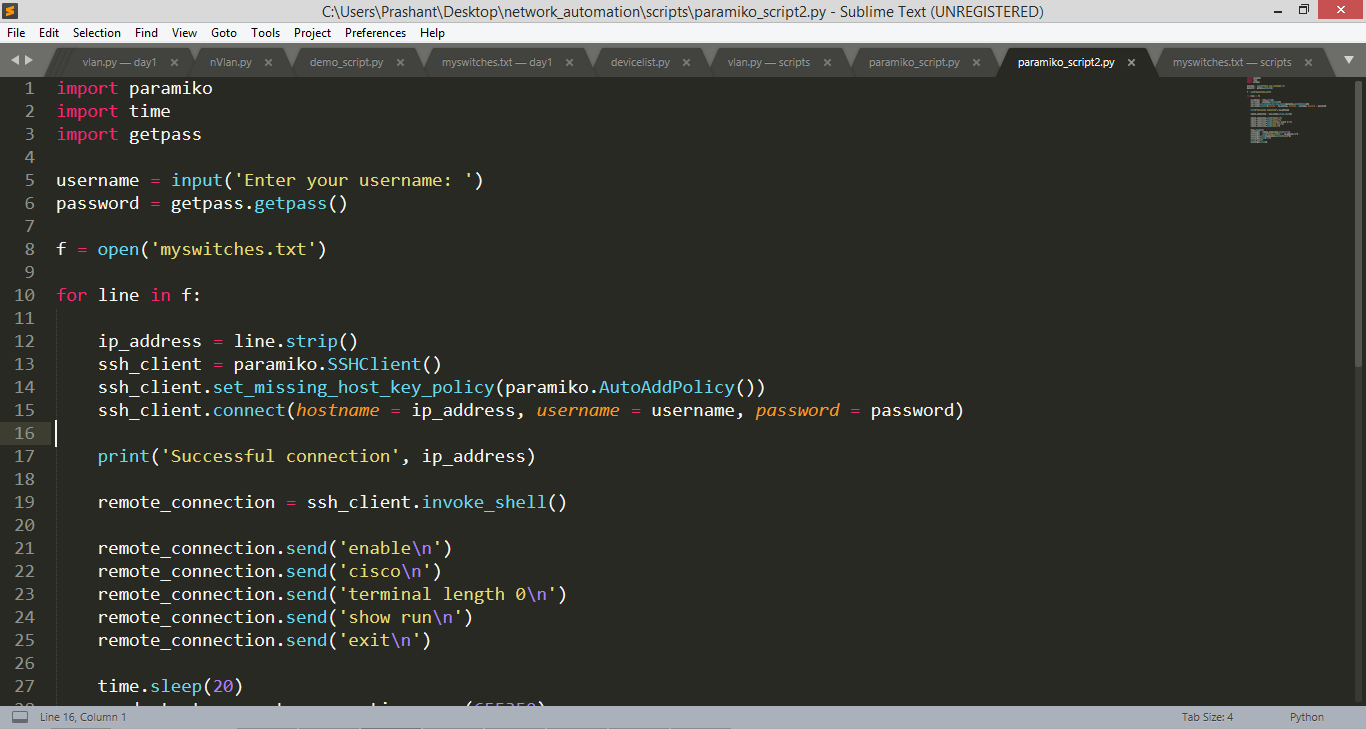
Our paramiko script:





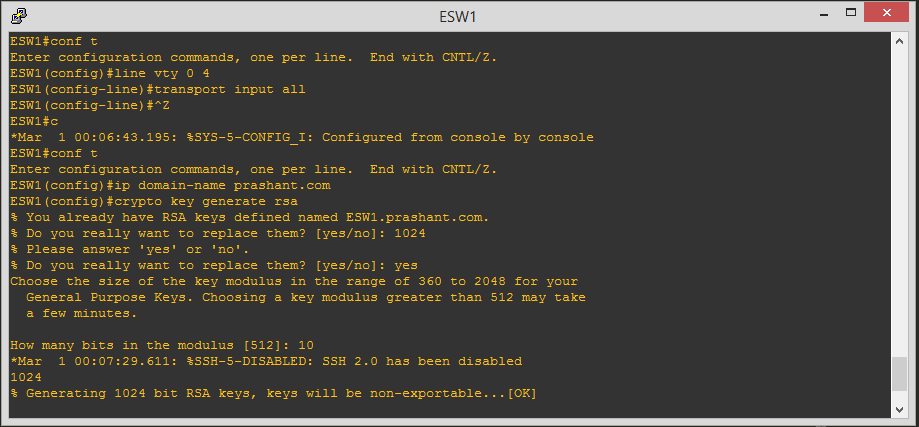
# Using paramiko for multiple device

We are doing the same as we did in telnet, script for paramiko is below:





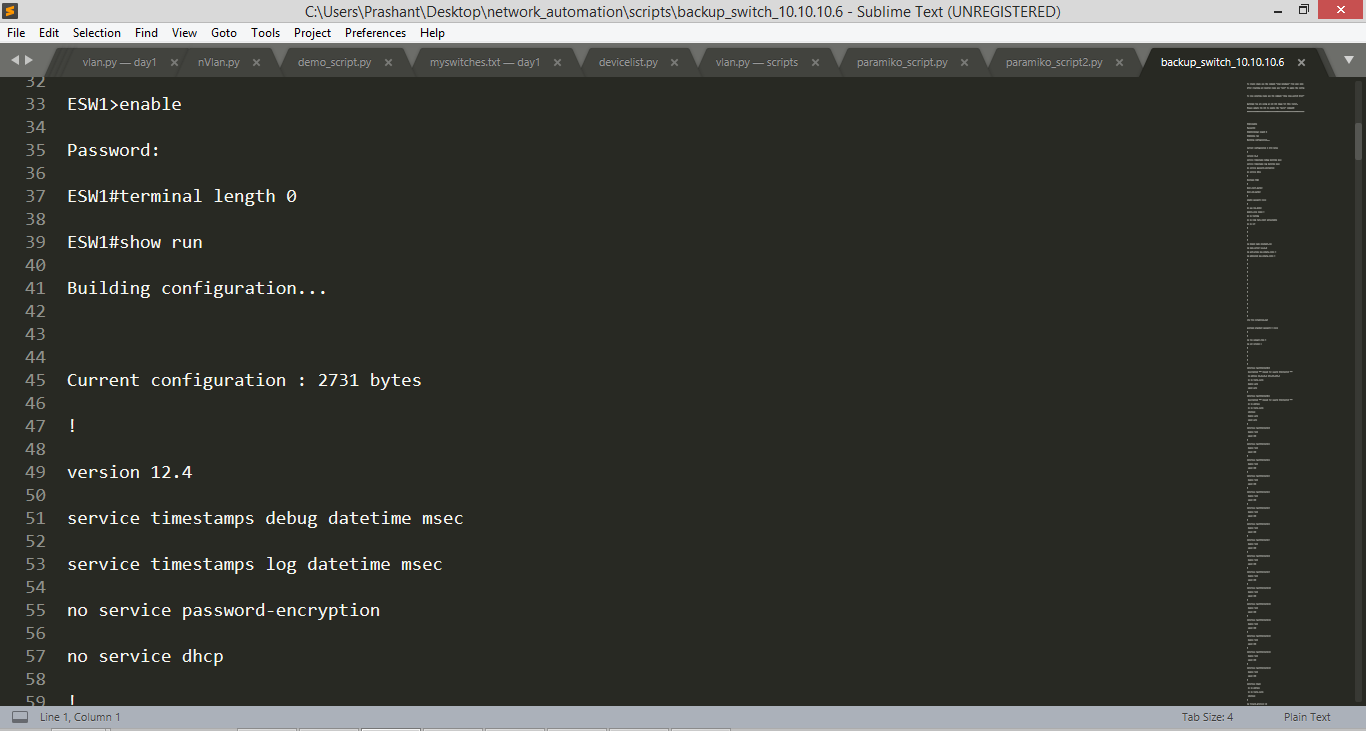
Updating configuration for ethernet switch





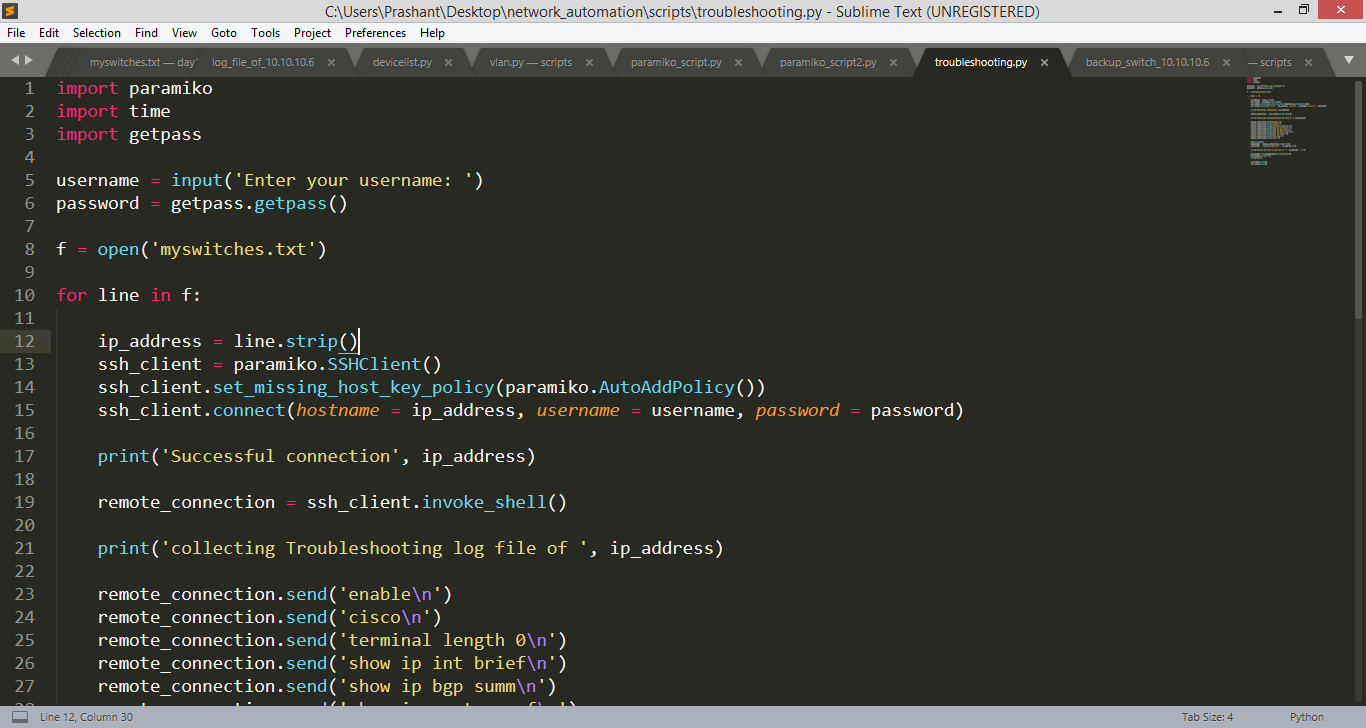
Our script will save your all configuration in a file. Which will work as a backup

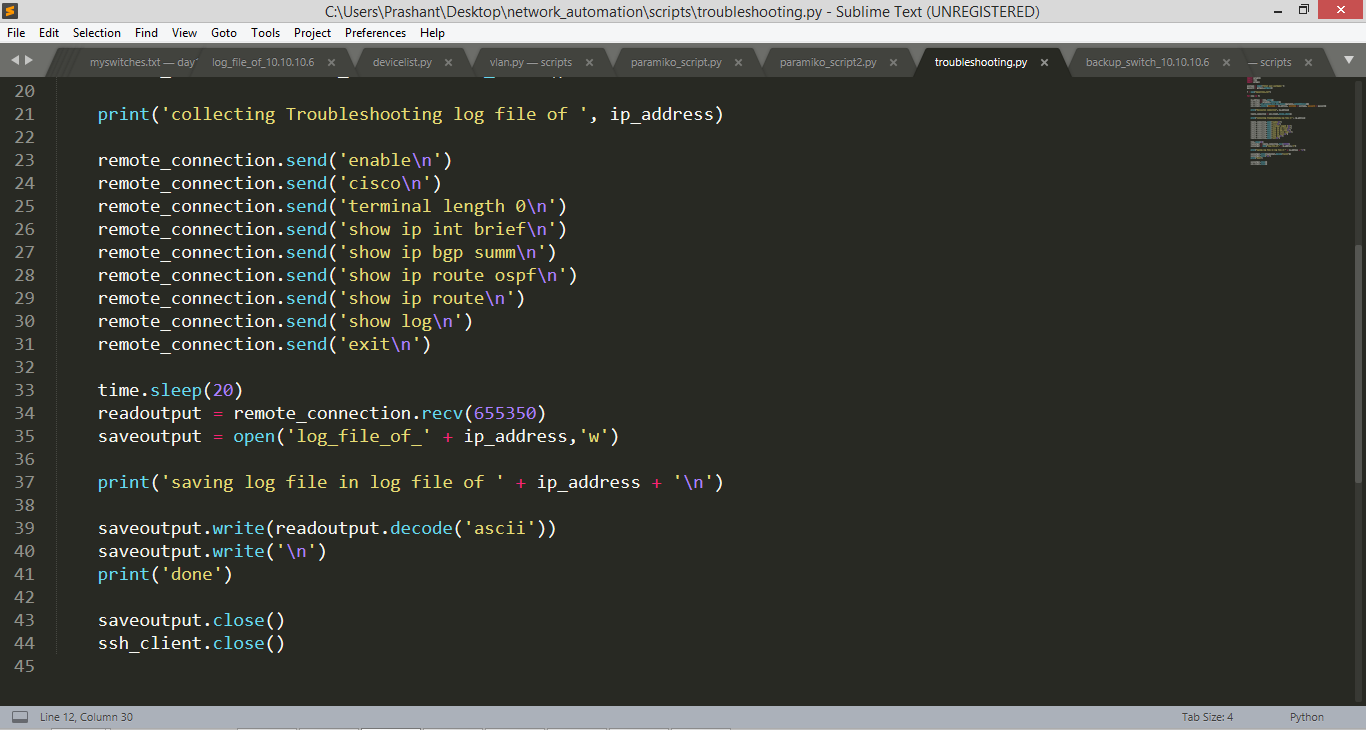
Your File will look like:



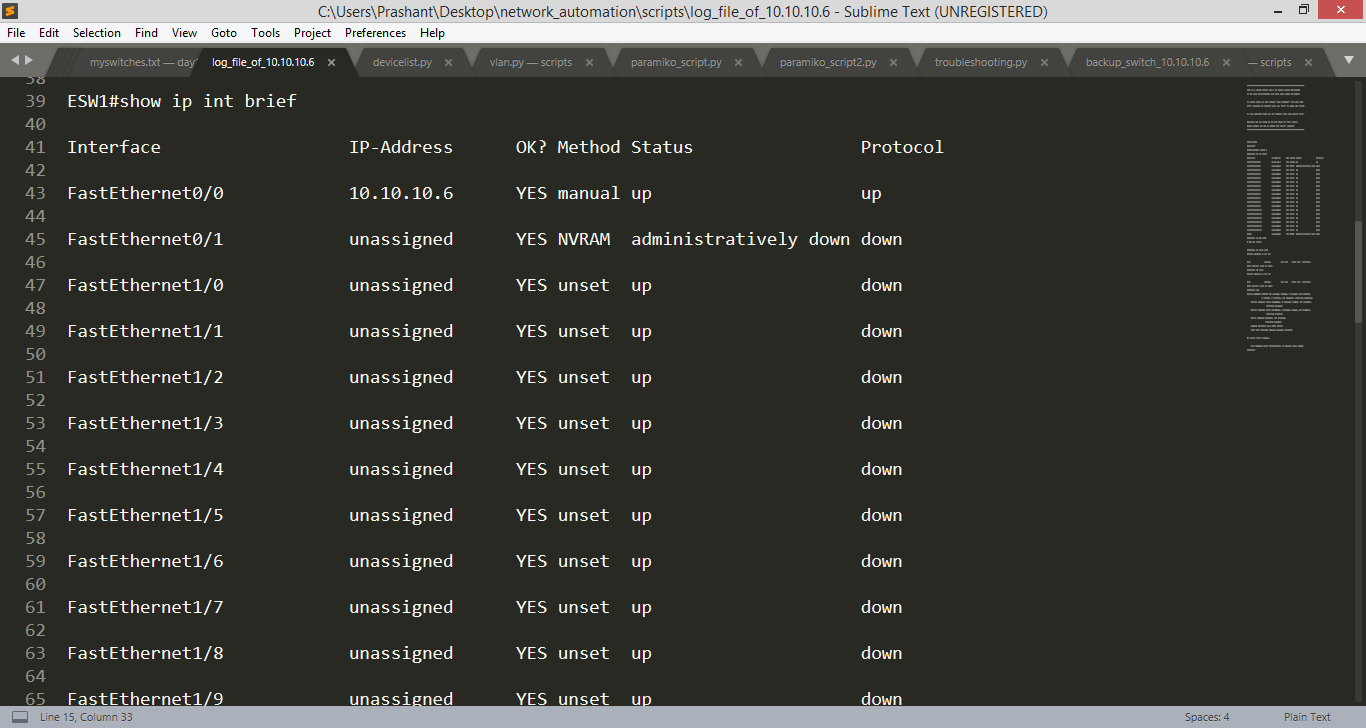
# Troubleshooting using paramiko

Paramiko script for fetching ip address, logs, bgp summary e.t.c





It will give you logs and all other stuff in the form of txt file, which will look like



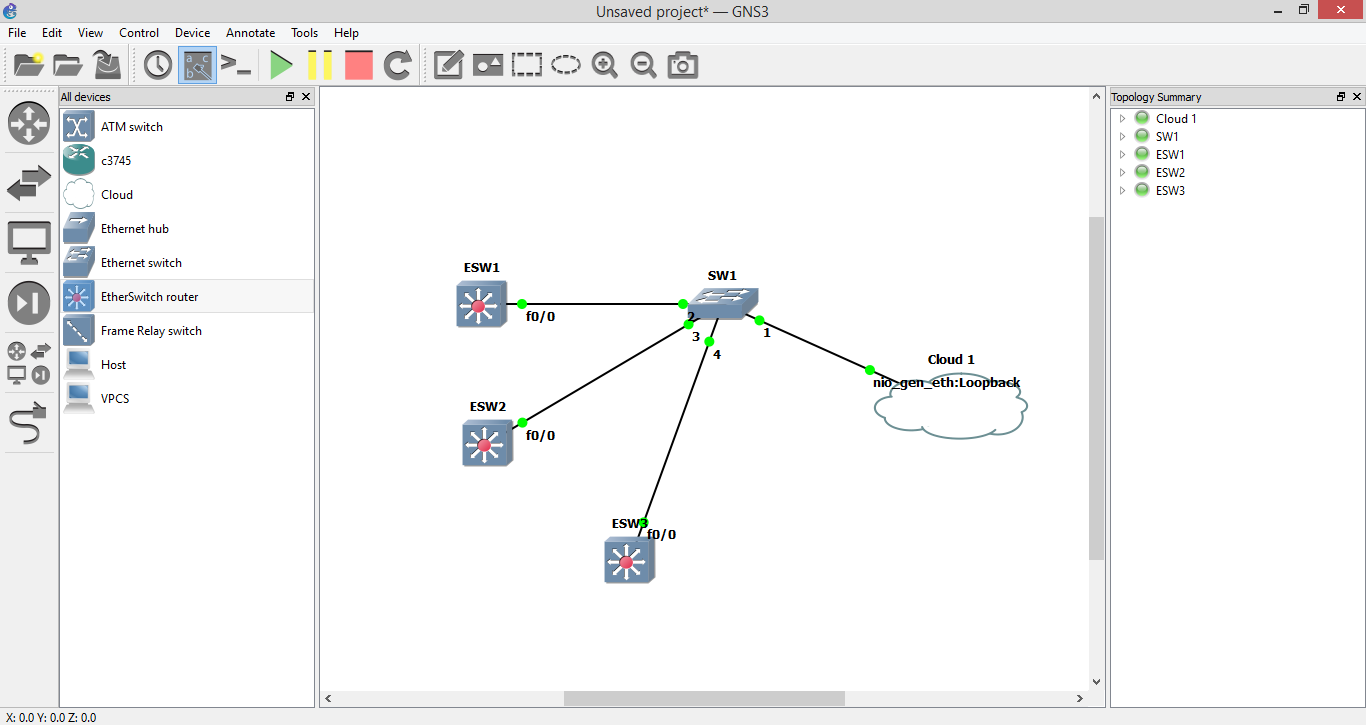
Here is the log



# Device accessibility using paramiko

Our python script to check that all devices are accessible or not.

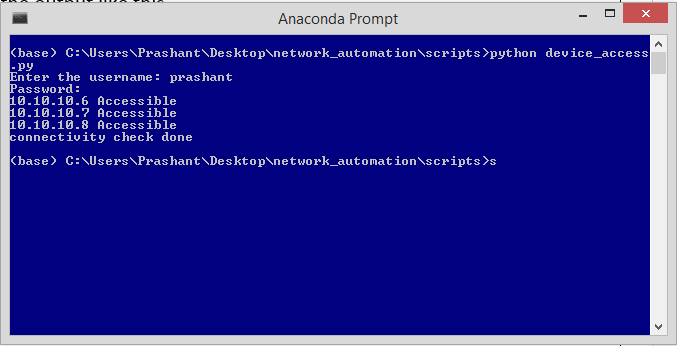
Here is our topology,



Python script to check the accessibility



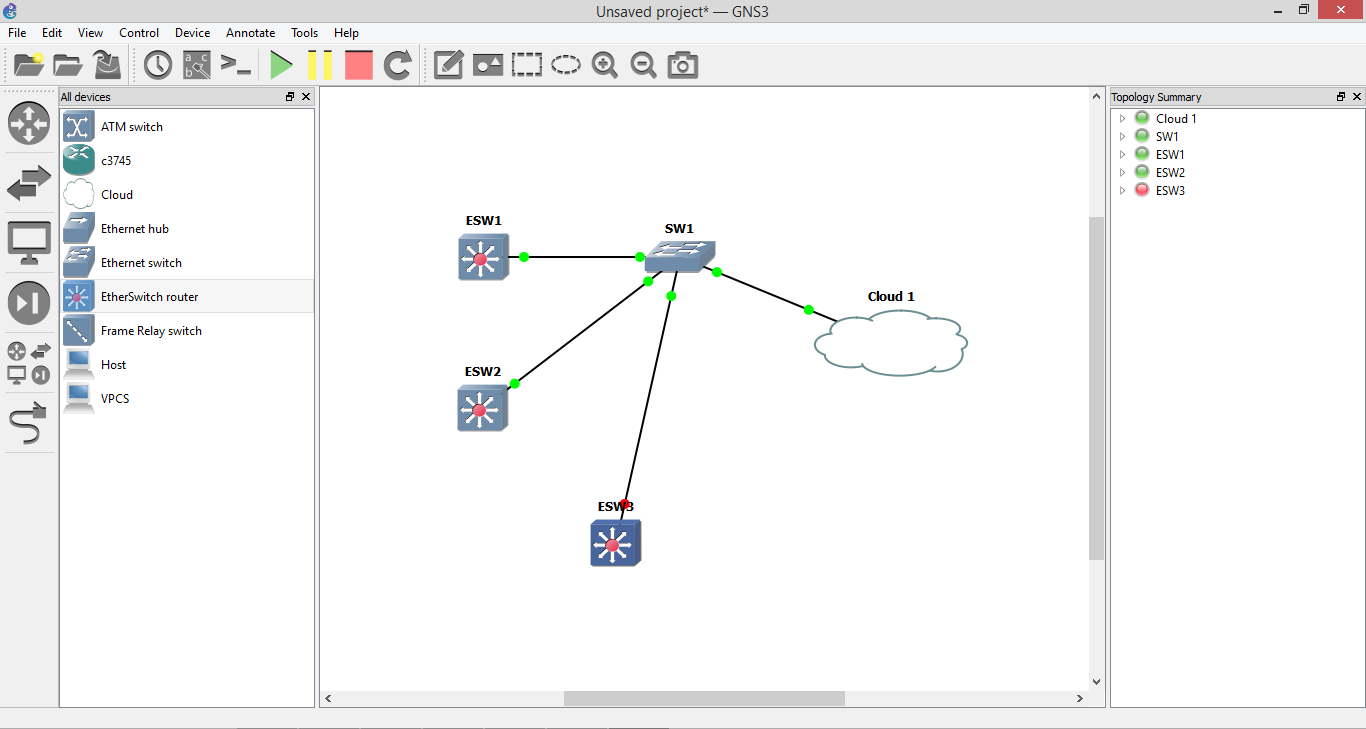
As you can see in our topology that all the devices are connected, so we will get the output like this,



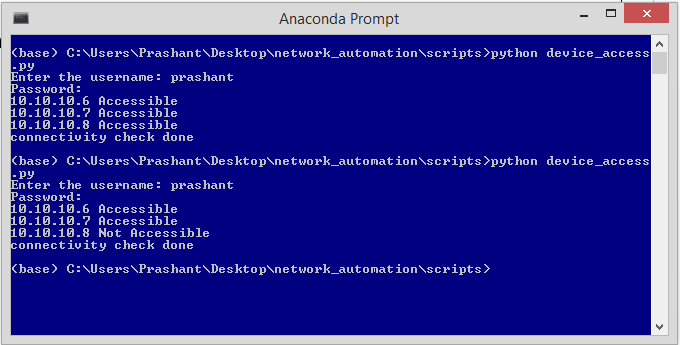
As you can see in the above output that all the devices are accessibile.

Now we will stop one device and will see the output.

We will stop ES3(10.10.10.8),

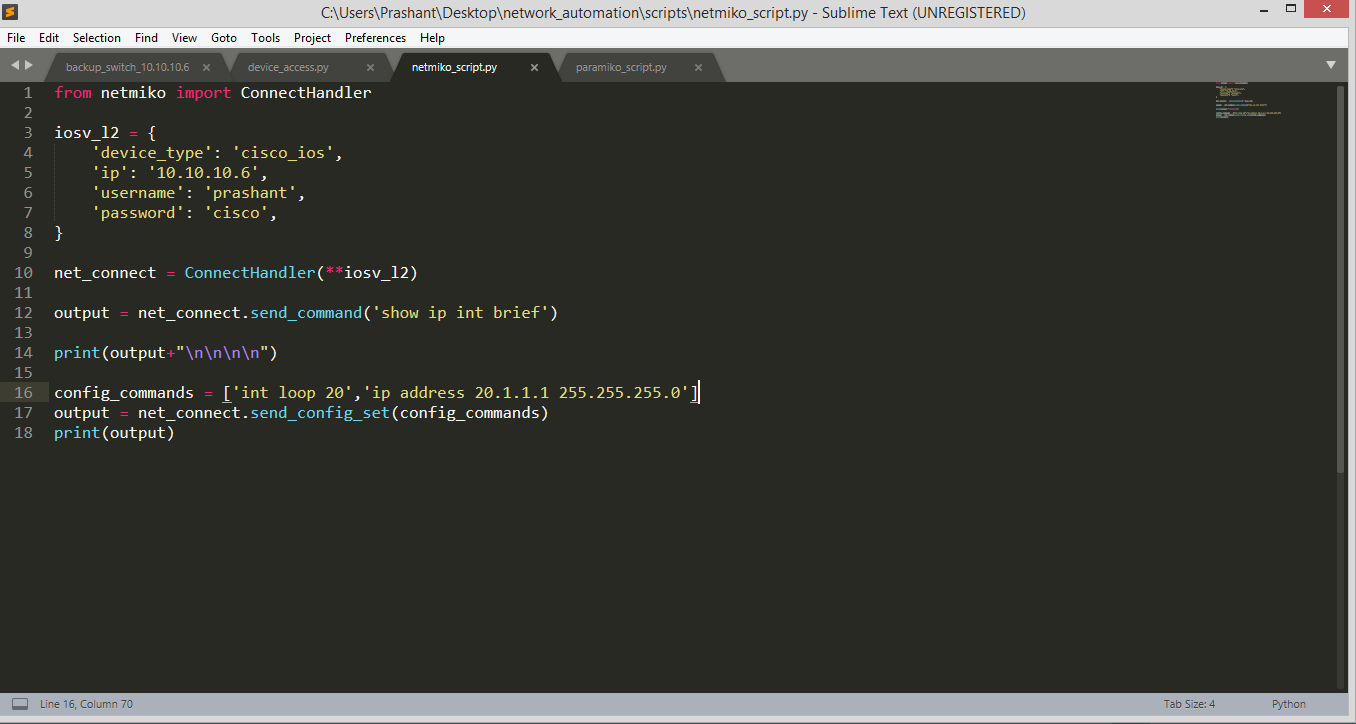


Now run the python script and see the output

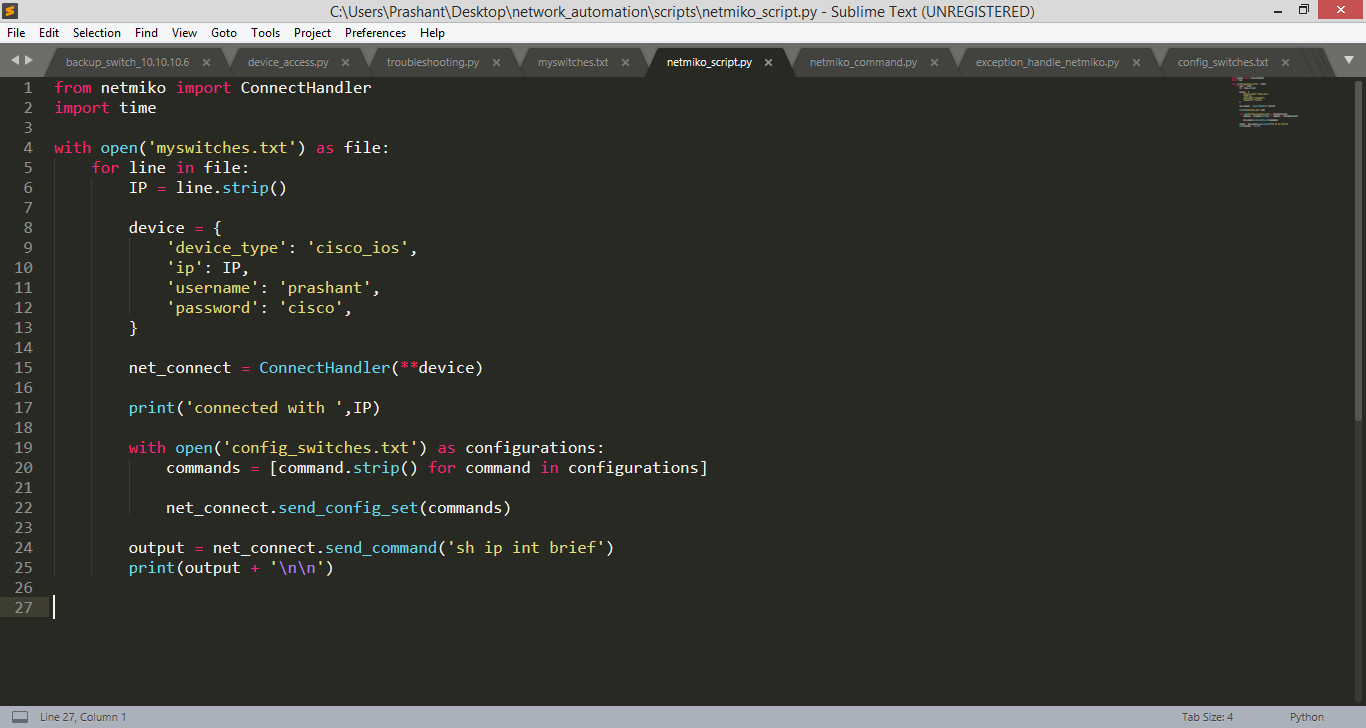


You can clearly see in the output that 10.10.10.8 is not accessibile now.

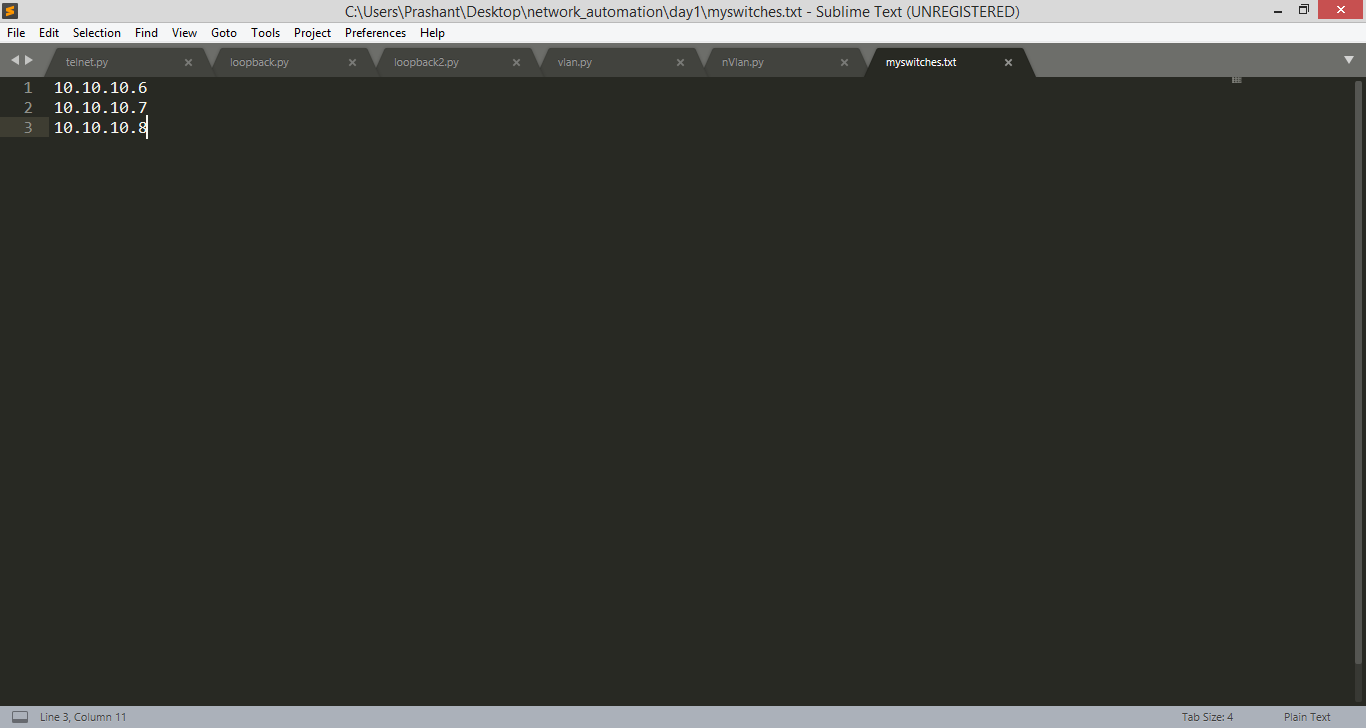
# Using Netmiko



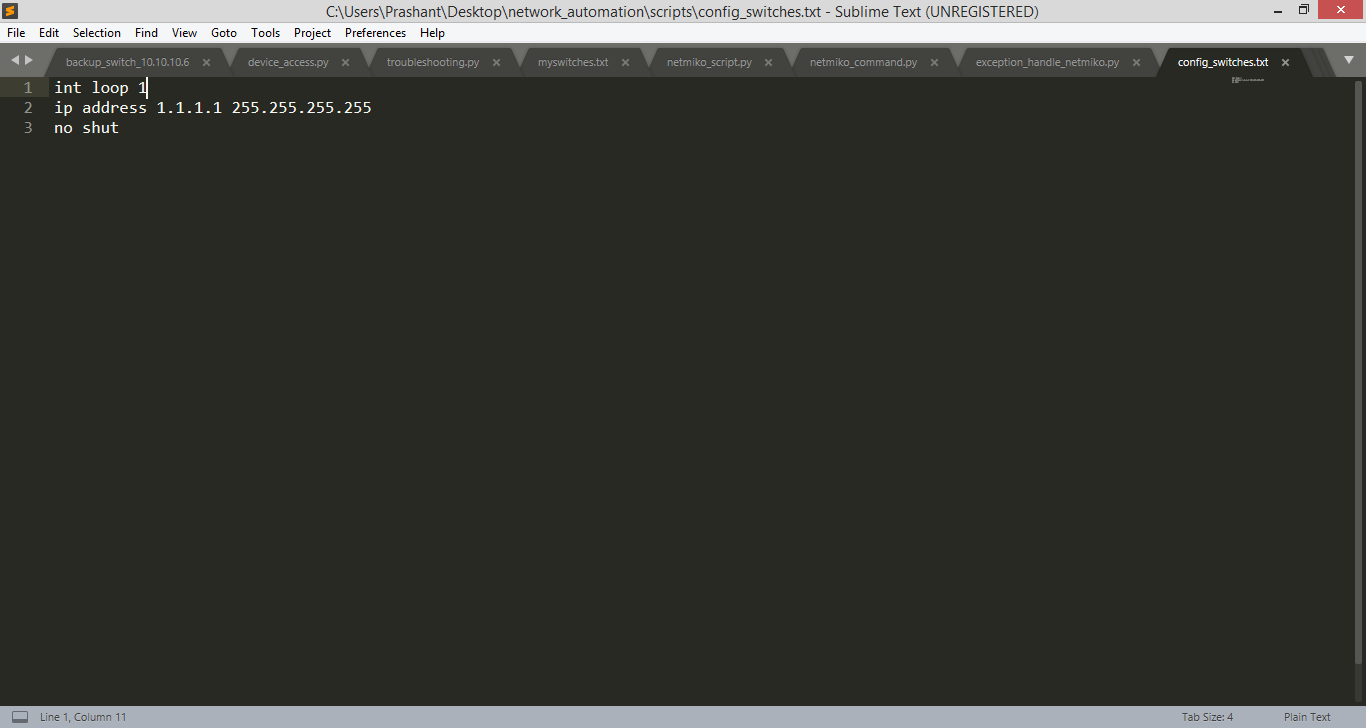
# Configure multiple devices using a text file



* Our myswitches.txt looks like,

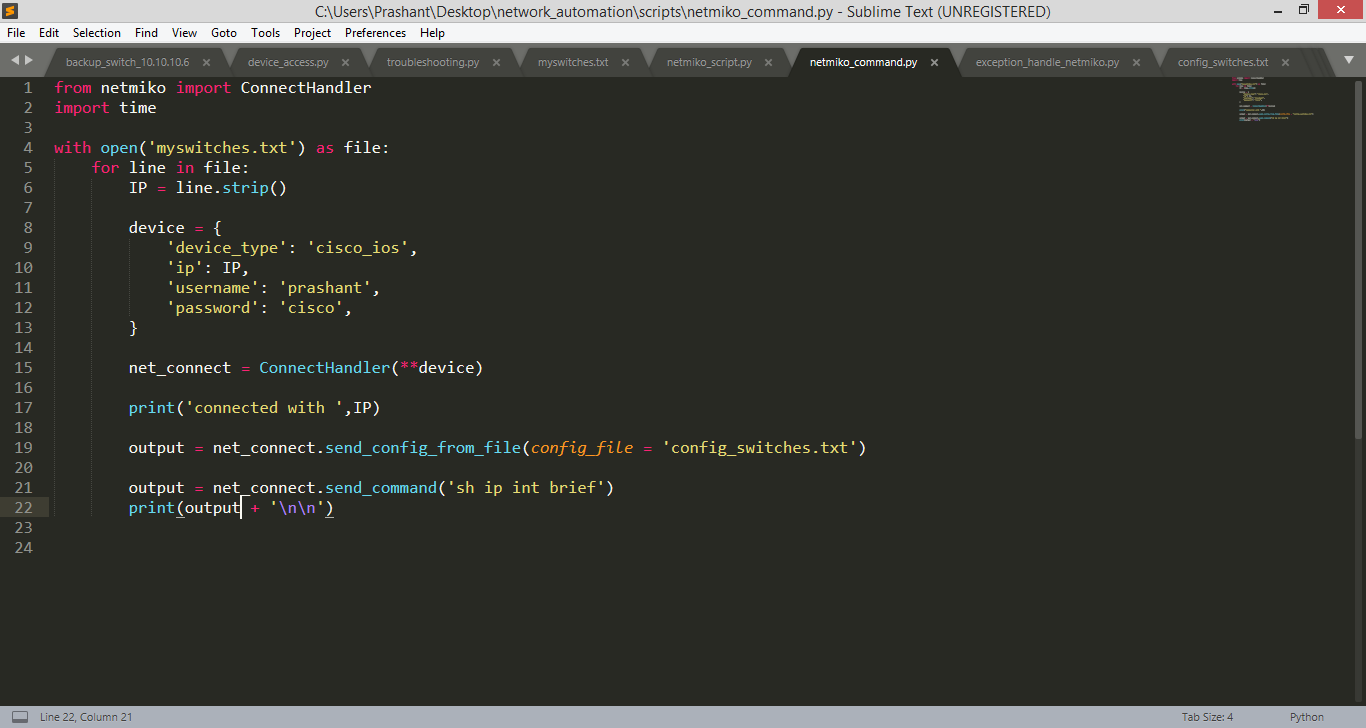


* Our config\_switches.txt looks like,

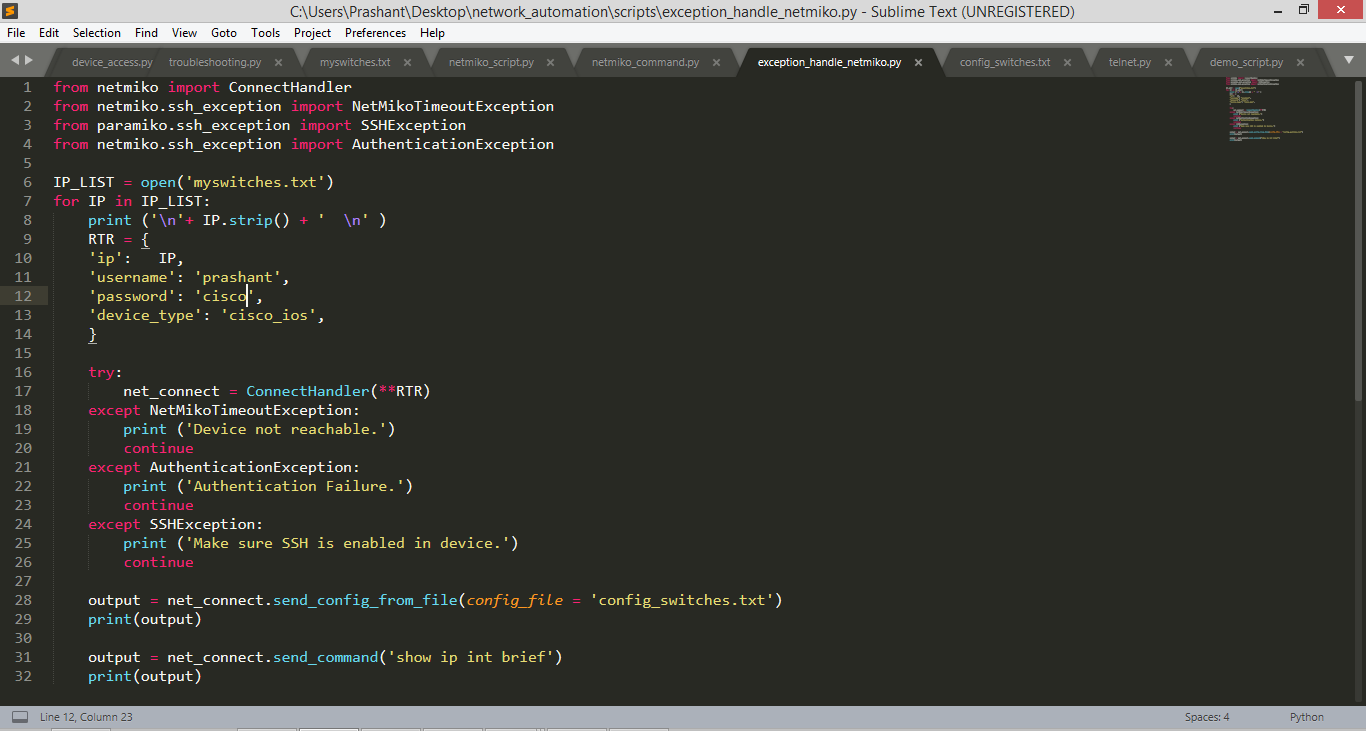


# Reading commands using function from text file

* Using send\_config\_from\_file() function, we can read all the configurations from a txt at once



# **Handle device not reachable exceptions in Netmiko**

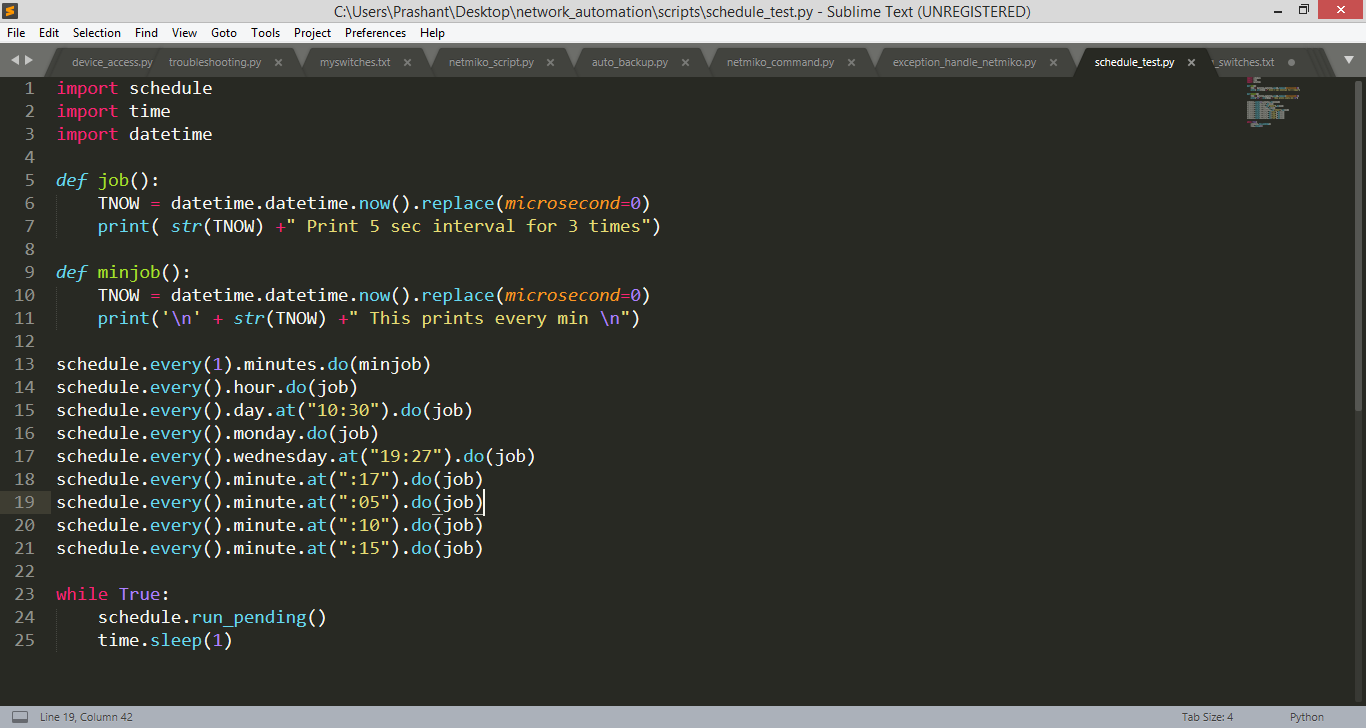


# Schedule Task

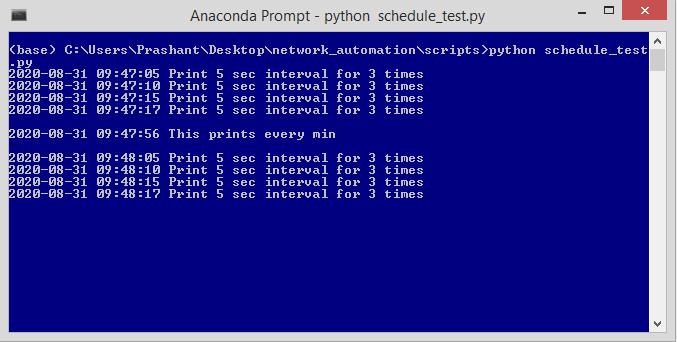
For this we need to install a library named Schedule

* Command to install schedule library
  + Pip install library

Script to Schedule Task



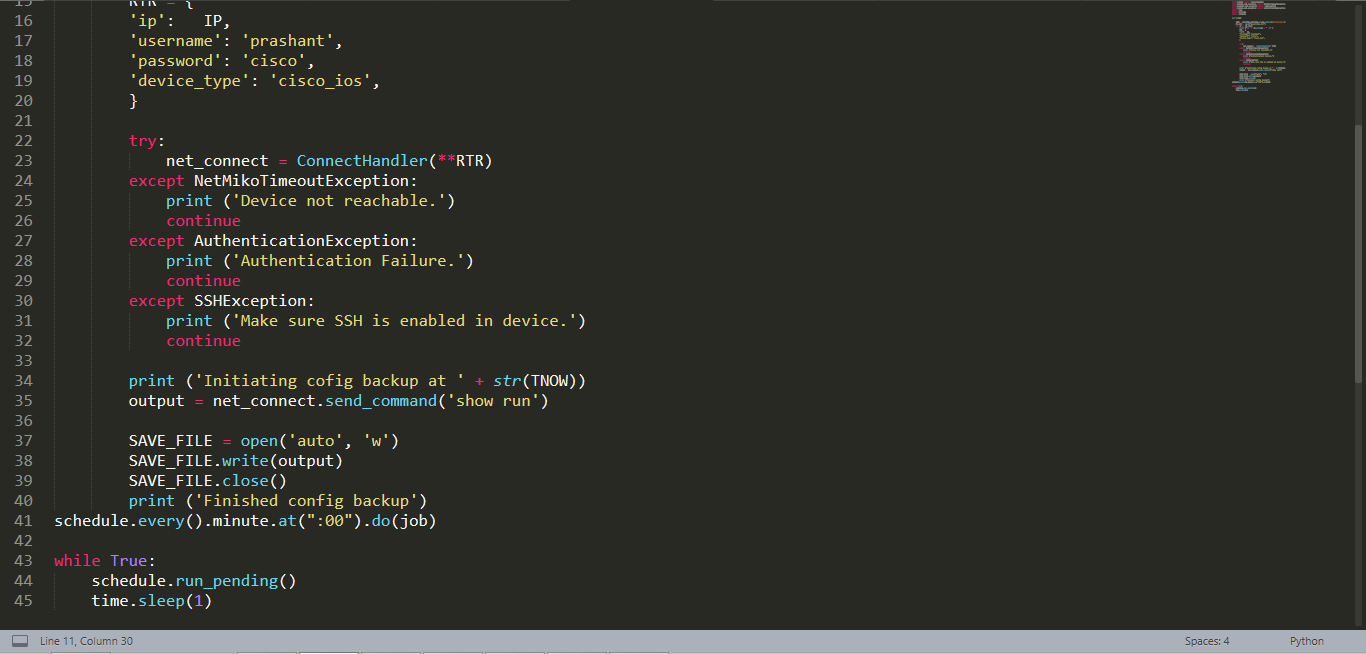
Output:



# How to schedule auto backup on cisco devices

Script:

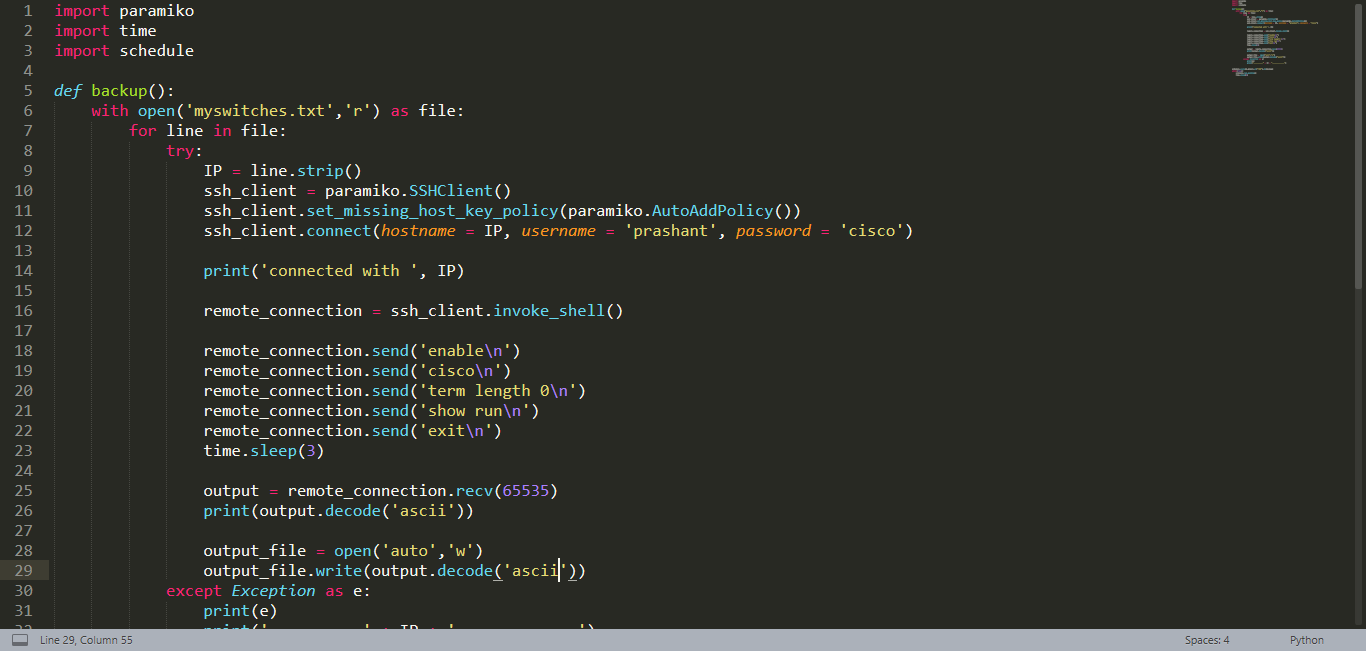


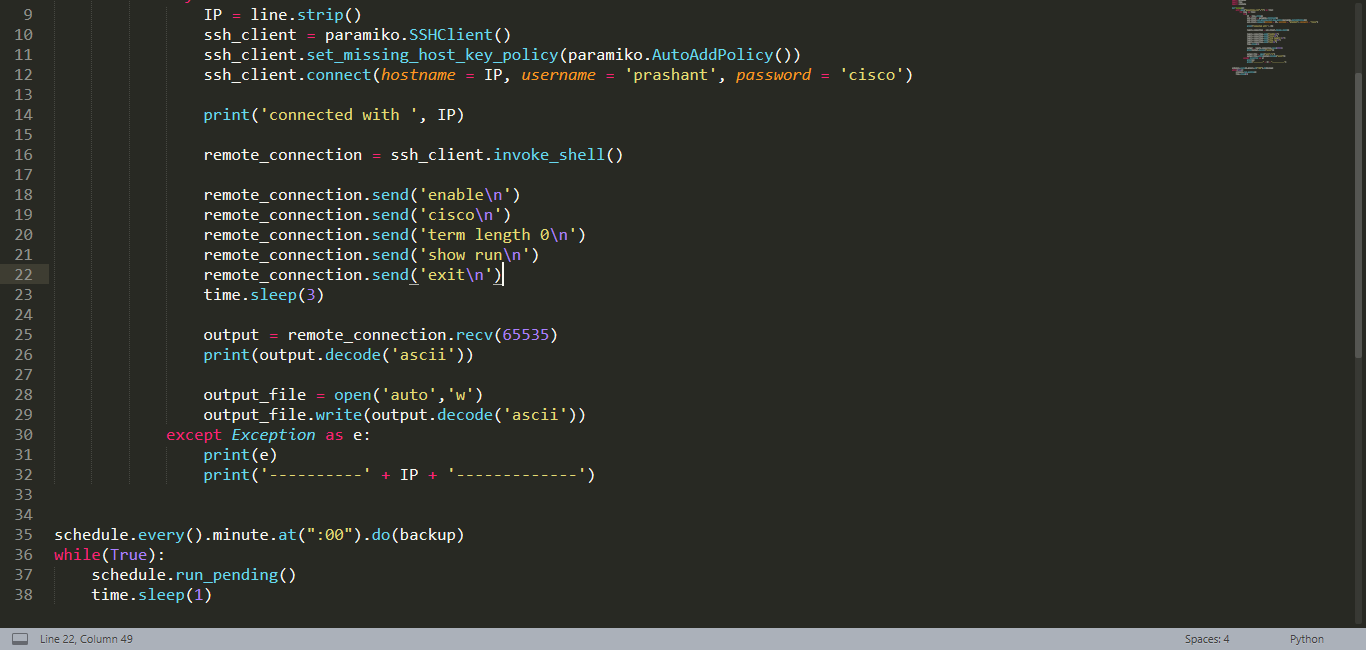


Output: After every minute it will create a file named auto with device backup in it.

# Schedule Automatic backup of Cisco device configuration Paramiko

Script:





Output: After every minute it will create a file named auto with device backup in it.

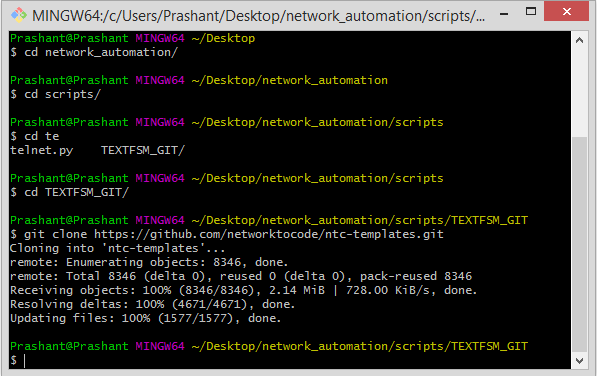
# INSTALLING GIT ON WINDOWS

In case, if you don’t have git installed on your local machine, follow these steps as shown in this site

* + <https://phoenixnap.com/kb/how-to-install-git-windows>

# How to Parse Cisco Device configuration in Python 3 using TEXTFSM

* You can see the list of commands it supports(TEXTFSM)
  + <https://github.com/networktocode/ntc-templates/blob/master/templates/index>
* Using TEXTFSM we can get structured output in the form of list, dictionary etc
* We will see it using a script
* Now we need to sync above template(link) to our local machine using a command in git bash
  + Git clone <https://github.com/networktocode/ntc-templates.git>
* Make a folder named TEXTFSM\_GIT and clone it there

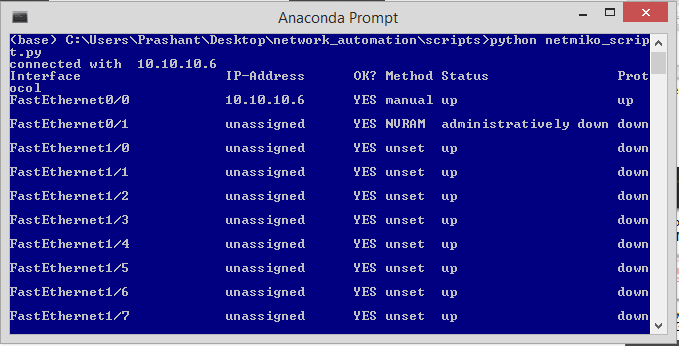


* After the cloning, it will create a folder automatically by name ntc-template, which will look like

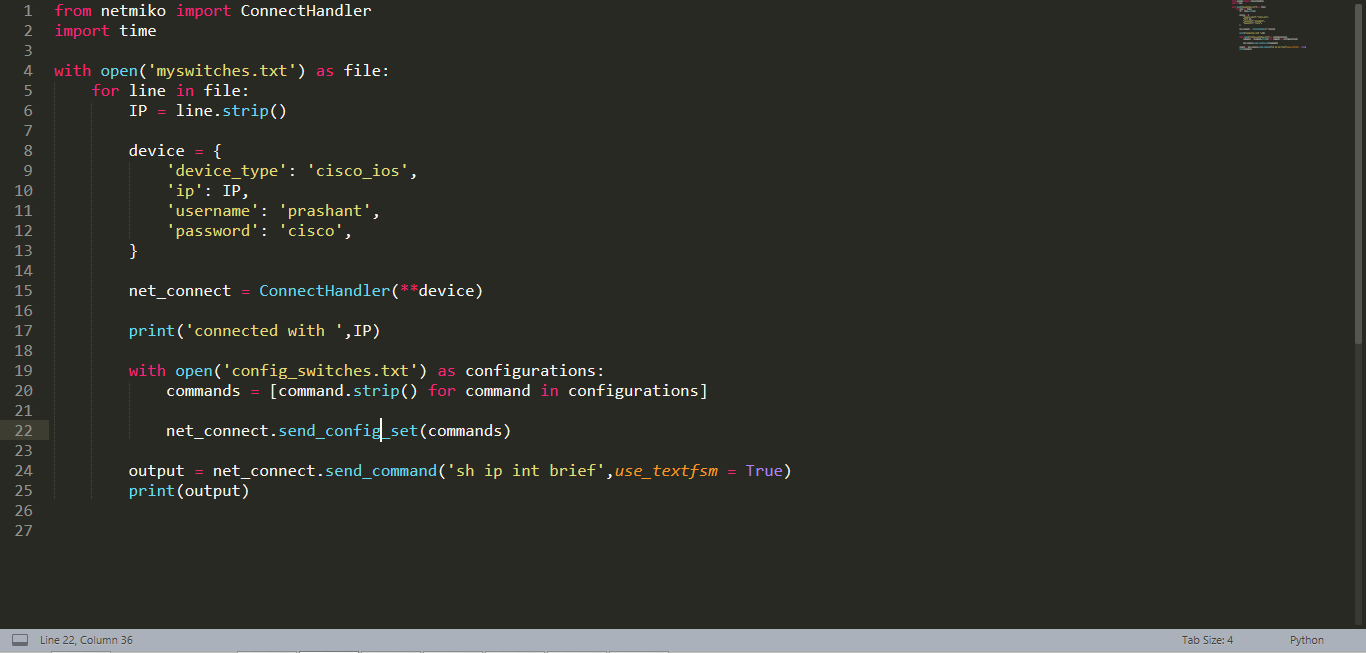


* Now we need to export the template file using command
  + set NET\_TEXTFSM = C:\Users\Prashant\Desktop\network\_automation\scripts\TEXTFSM\_GIT\ntc-templates\templates

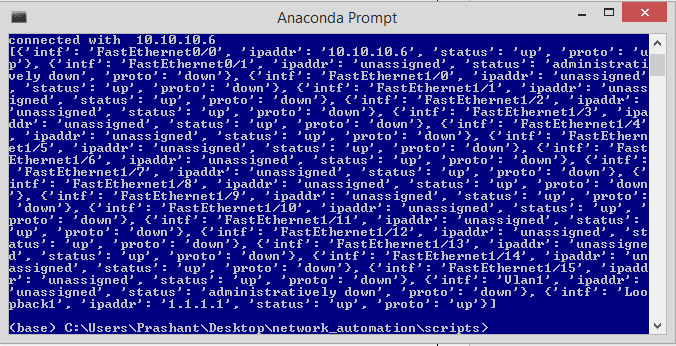
Before using TEXTFSM, we used to get output in the form of string like this



But Now after using TEXTFSM we will get a structured output in a dictionary format, let’s see a script for TEXTFSM on the next page



Output:



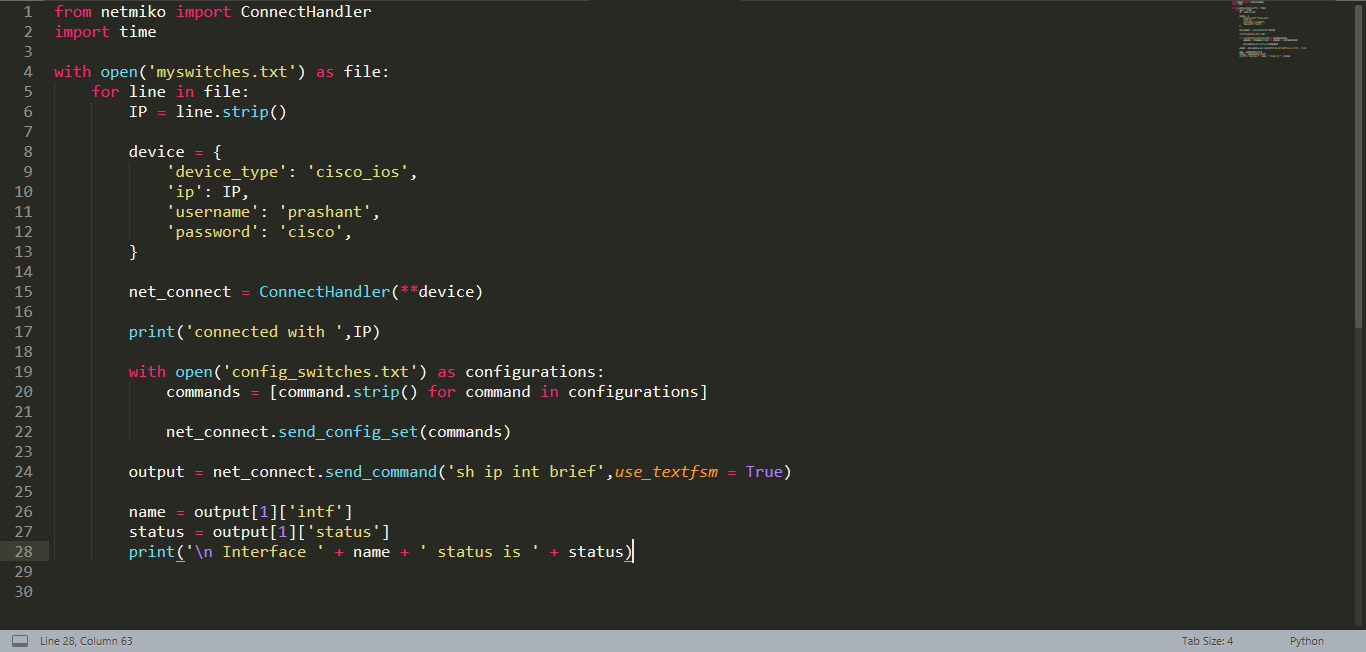
As you can see the output is stored in a data-structure now.

Now you can easily fetch whatever value or info you need.

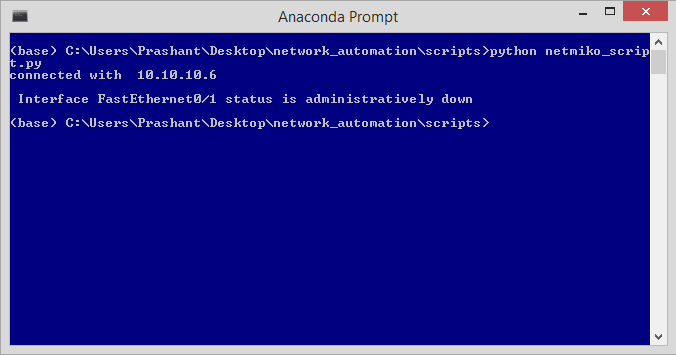
For ex: If you want to see a specific interface and it’s status we can do it through a script, let’s see the script on the next page.

# How to get a specific info about device using script

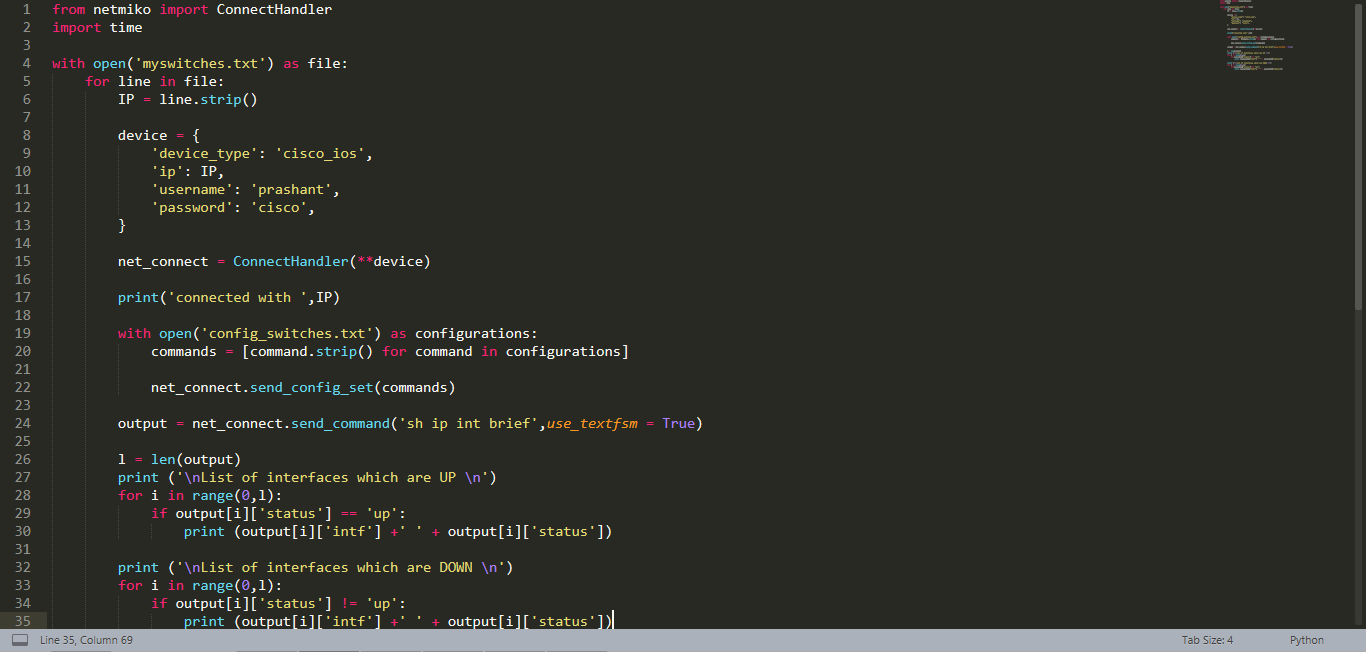
Script to see a specific interface and it’s status.



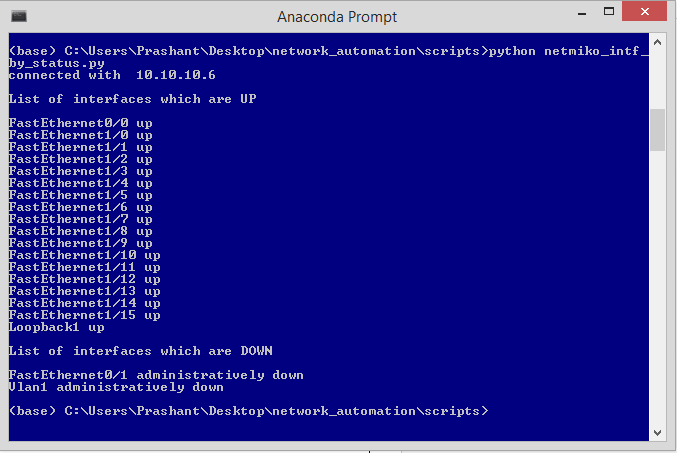
Output:



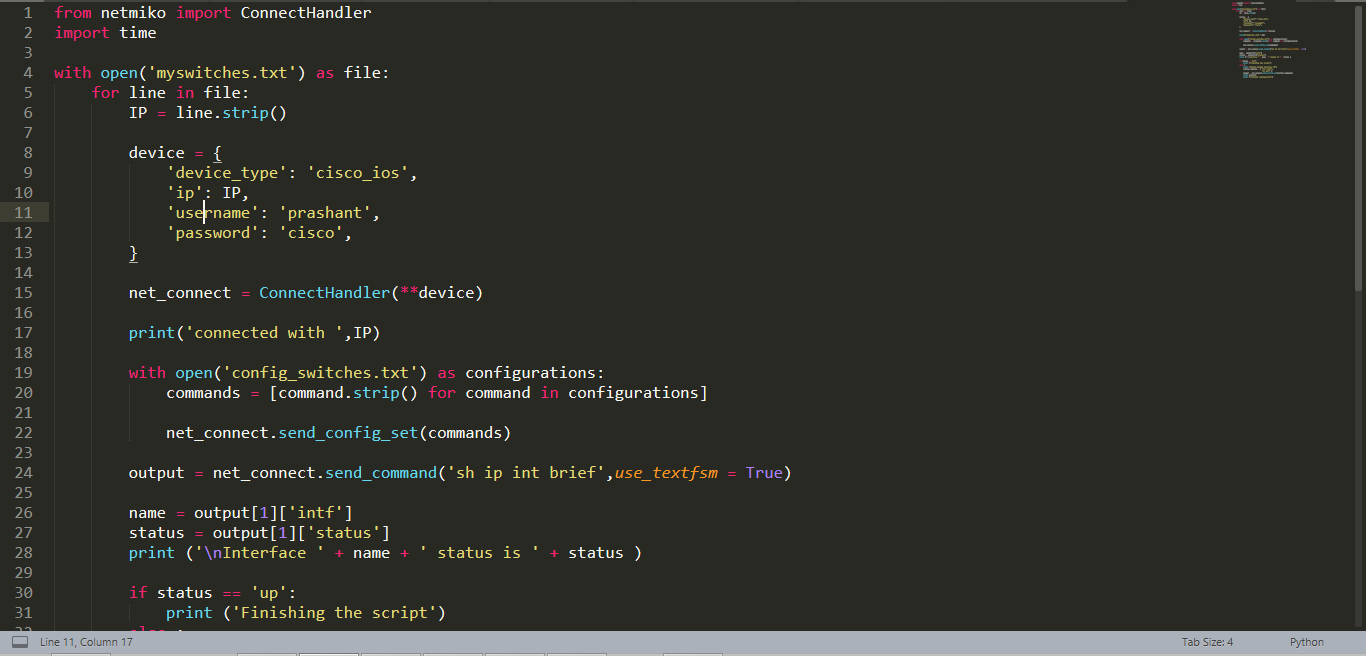
# **How to list Cisco Interfaces based on Status**

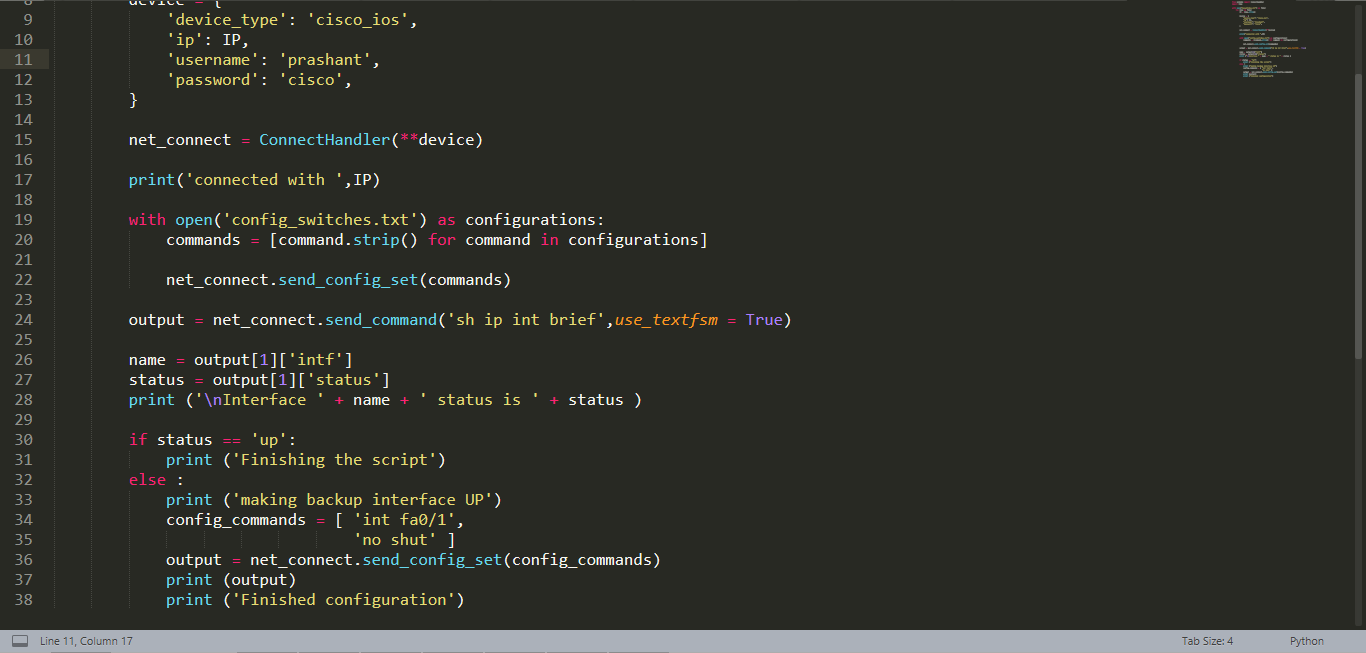


Output:

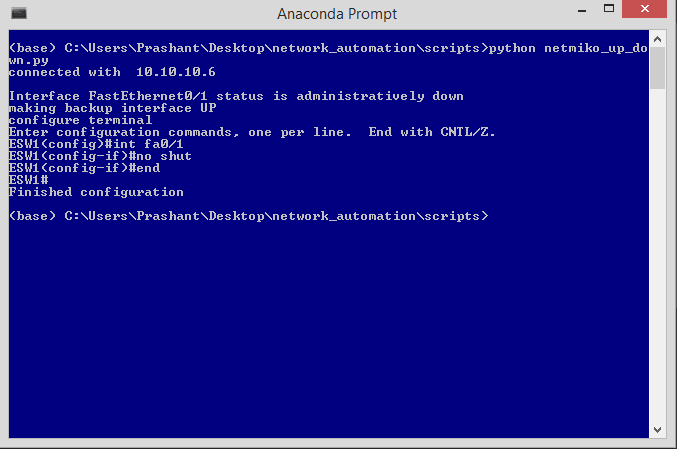


# If primary interface is down make secondary interface UP using script



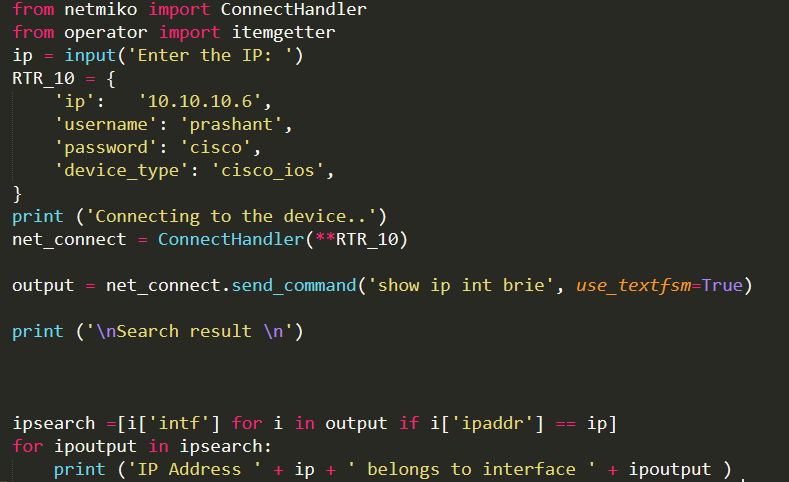


Output:

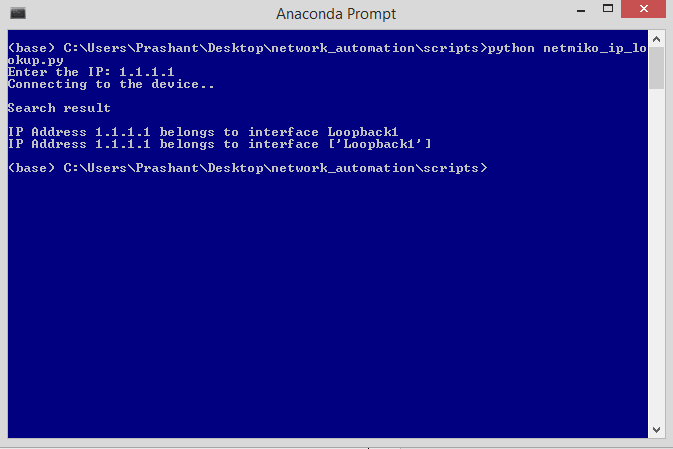


# How to search for a specific interface using script

SCRIPT:



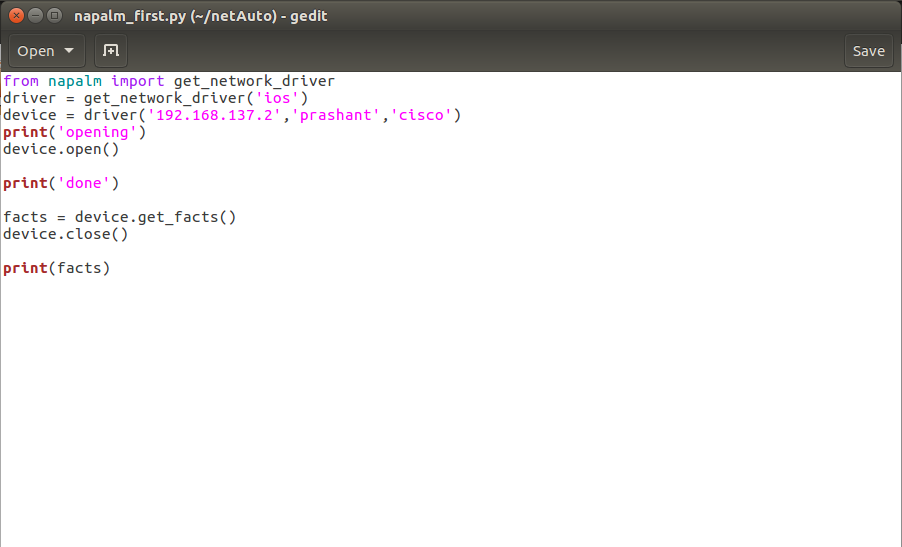
Output:



# NAPALM LIBRARY

* NAPALM (Network Automation and Programmability Abstraction Layer with Multivendor support) is a Python library that implements a set of functions to interact with different network device Operating Systems using a unified API.
* Command to install NAPALM library
  + Pip install NAPALM

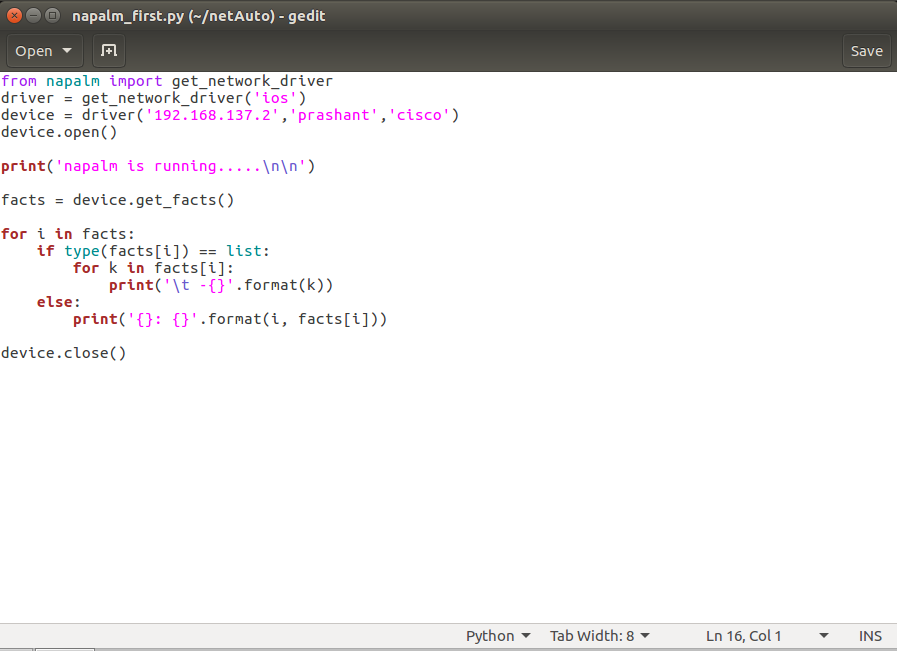
# Napalm script to get facts



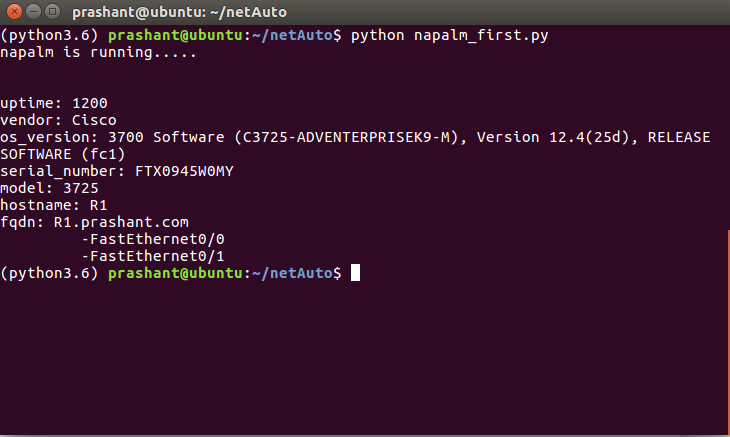
Output:



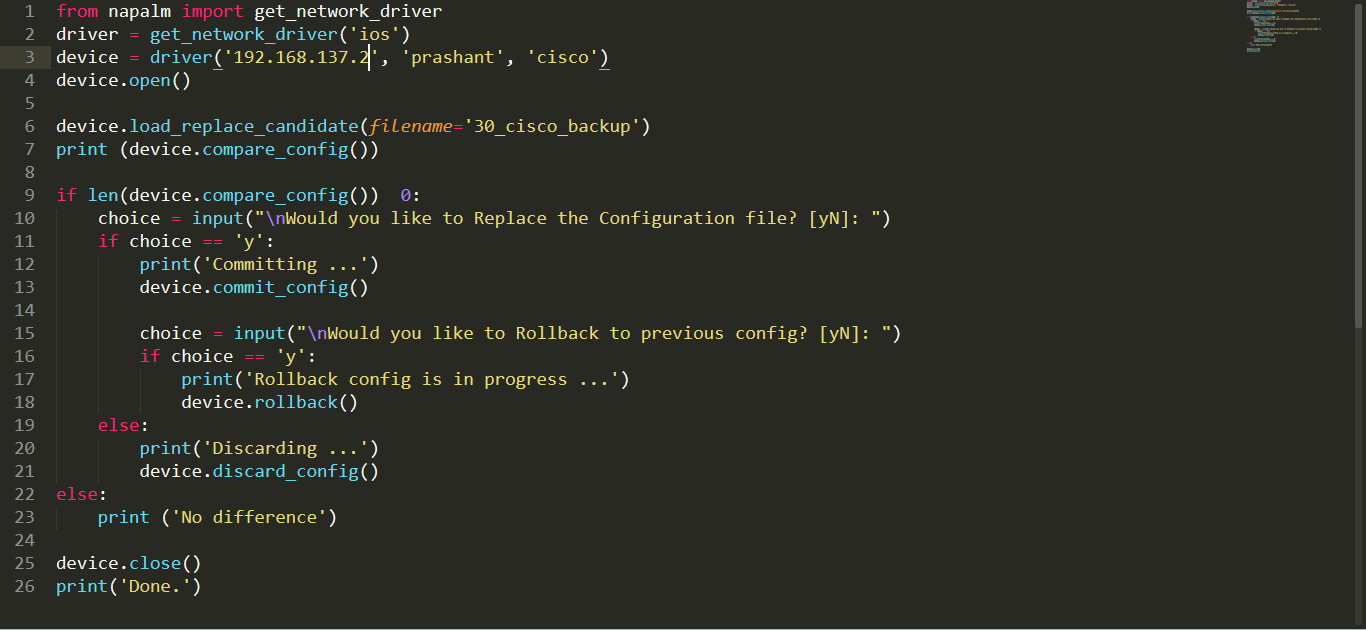
# Printing facts of device in a cleaner format



Output:



# How to rollback the configuration



# How to rollback configuration for arista device



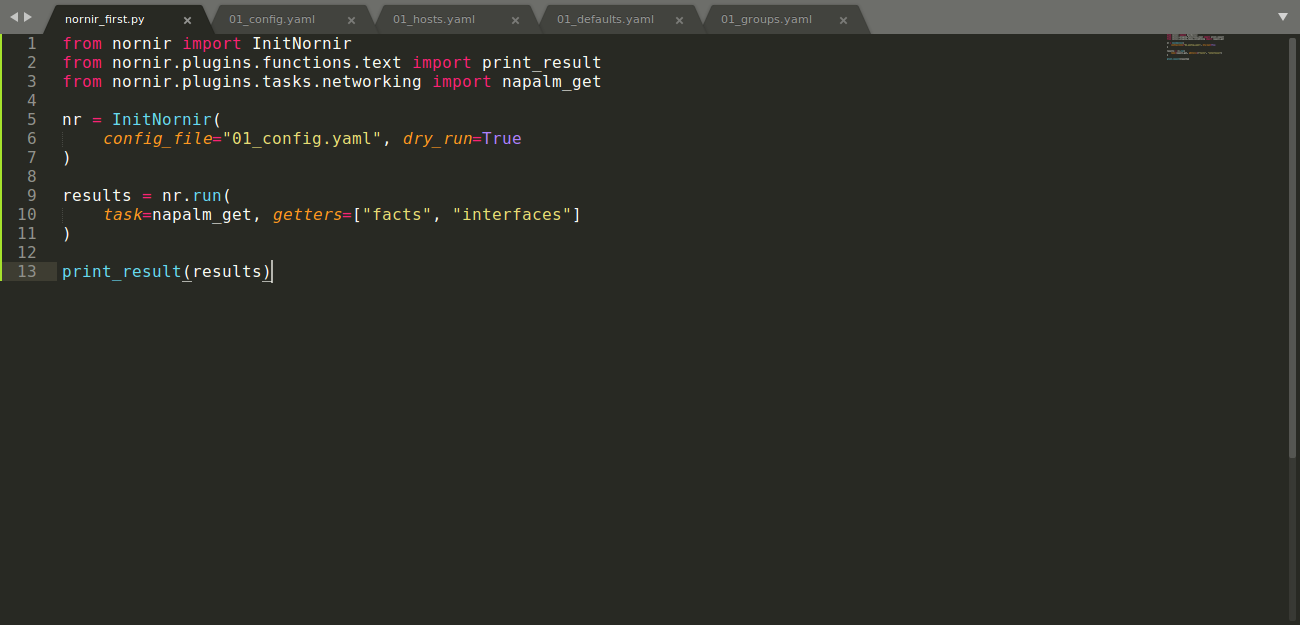
# Nornir Automation Framework

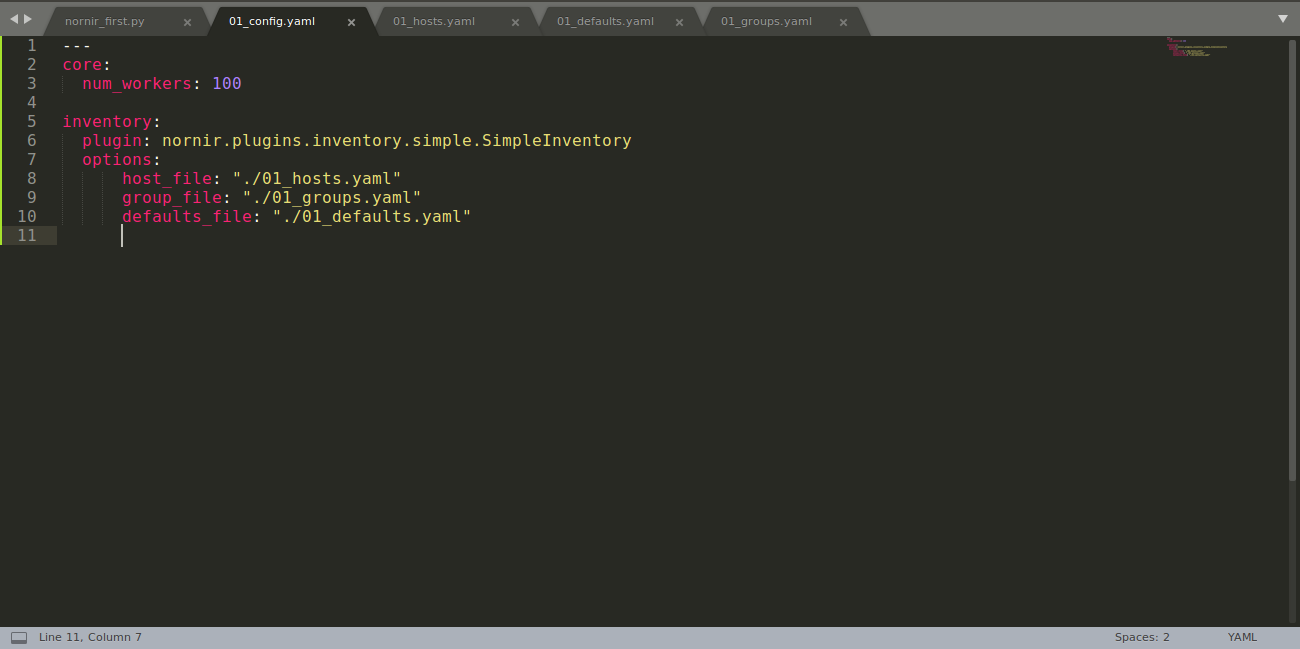
Nornir is an automation framework written in python to be used with python. Most automation frameworks hide the language they are written in by using some cumbersome pseudo-language which usually is almost Turing complete, but lacks tooling to debug and troubleshoot. Integrating with other systems is also usually quite hard as they usually have complex APIs if any at all. Some of the other common problems of those pseudo-languages is that are usually quite bad at dealing with data and re-usability is limited.

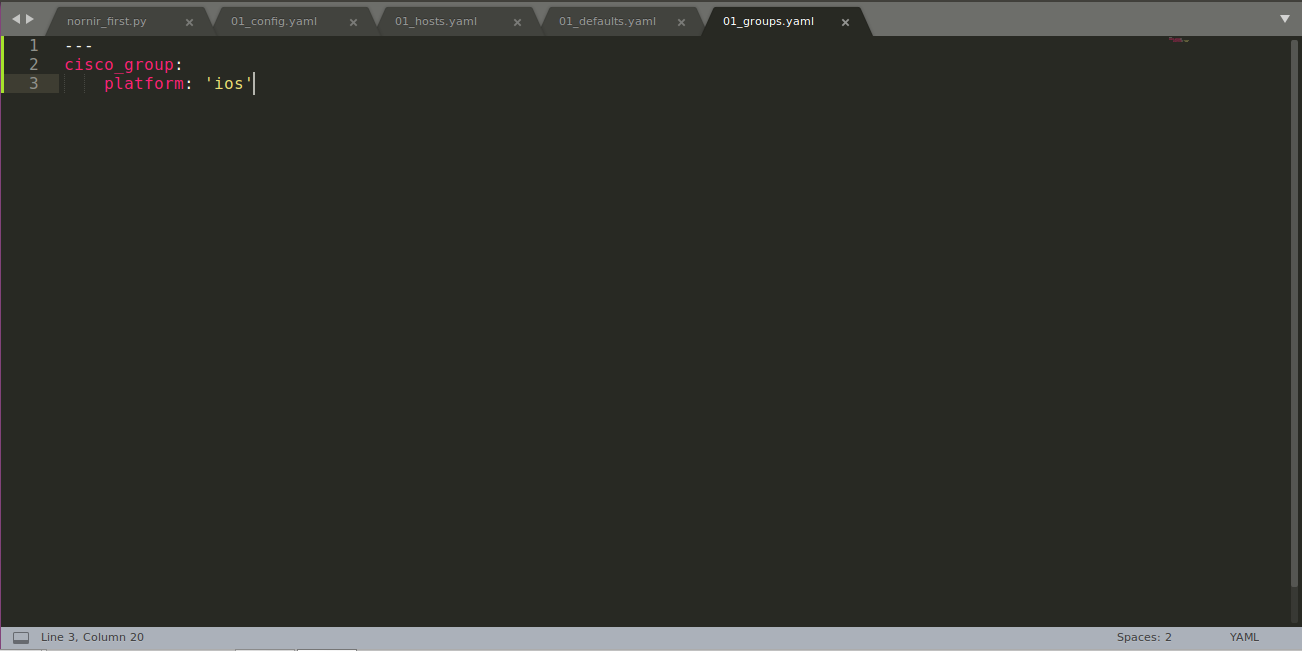
Nornir aims to solve those problems by providing a pure python framework. Just imagine Nornir as the Flask of automation. Nornir will take care of dealing with the inventory where you have your host information, it will take care of dispatching the tasks to your devices and will provide a common framework to write “plugins”.

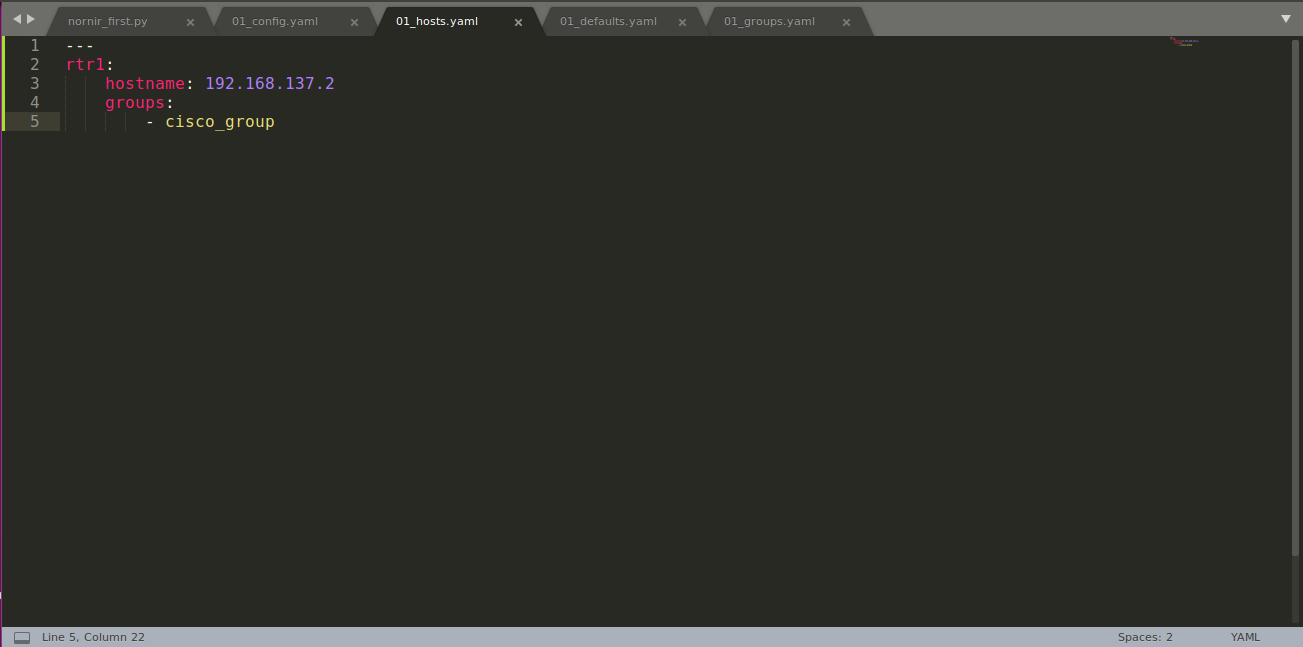
Nornir requires Python 3.6.2 or higher to be installed.

# Sample Script







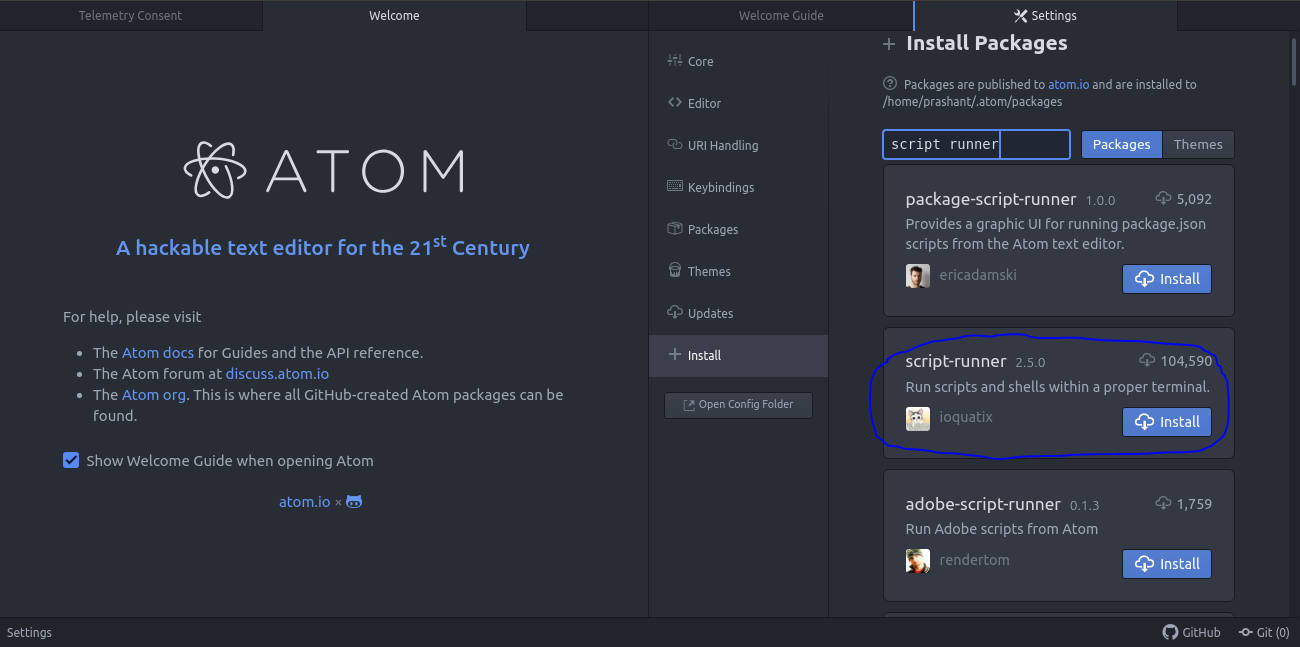


# ATOM Python IDE Setup

* Go to <atom.io>
* Download .Deb version
* Go to the downloaded folder
* Open the terminal
* Give the command
  + sudo dpkg –i atom-amd64.deb
* After installation, open atom on your machine

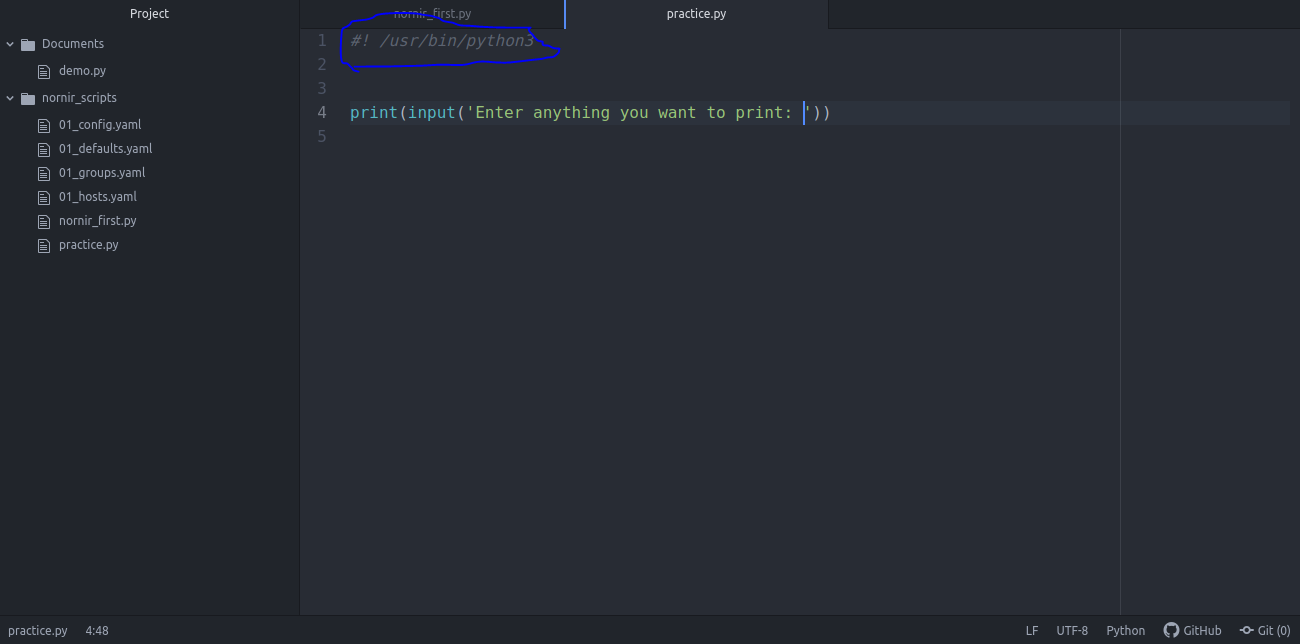
Now we need to install packages to run python on atom

* Go to:
  + Edit -> preferences
* Go to install
* Type script runner and install it



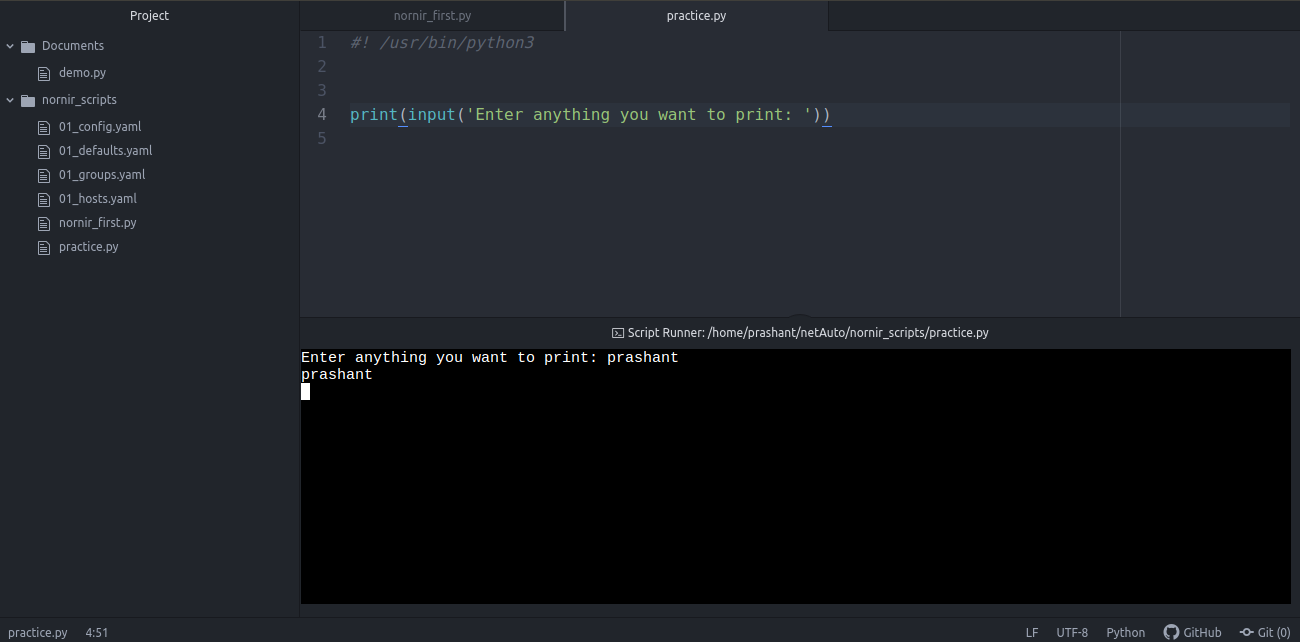
* Now we will run a demo script

We have to set python3 path in our script as shown in below fig.



In order to run our script:

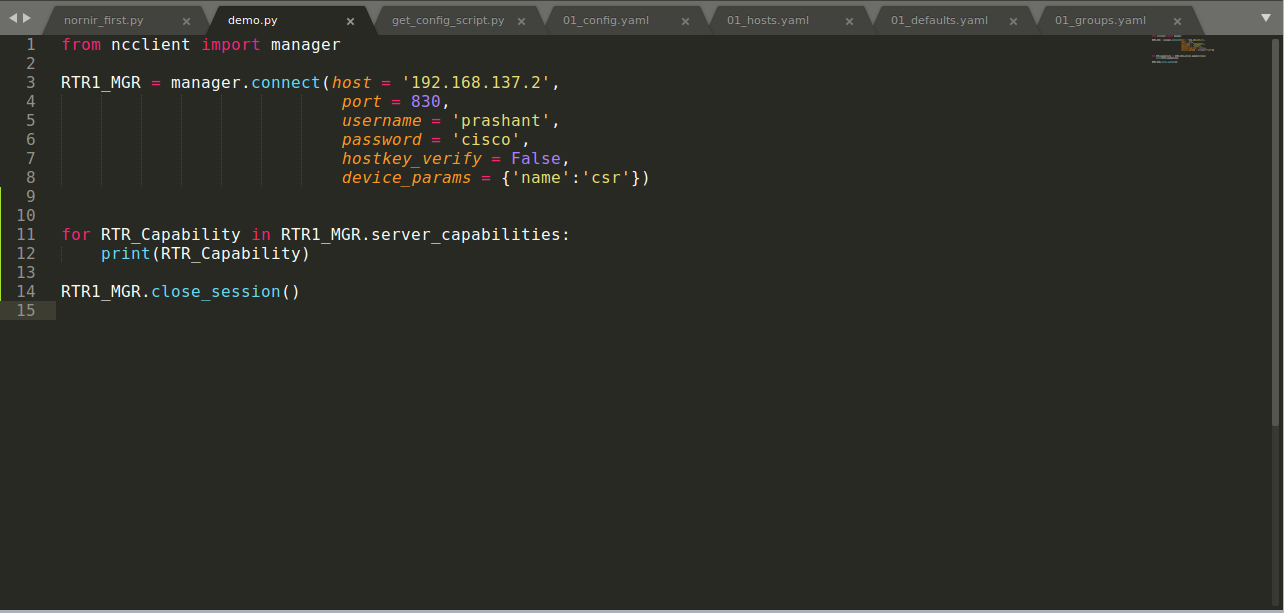
* Press Alt + X



# NETCONF

* First we will clone the yang repository using command
  + git clone <https://github.com/YangModels/yang>
* Install ncclient using command
  + pip install ncclient

# Script to fetch capabilities



# Script to get running config



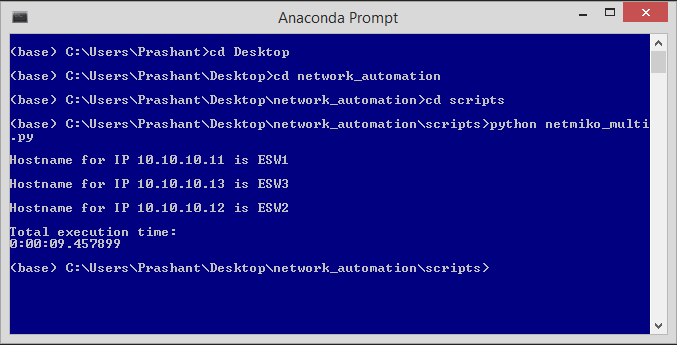
# Multithreading in Netmiko

A key focus area while performing operations on multiple devices is how quickly we can perform the actions. To put this into perspective, if each router takes around 10 seconds to log in, gather the output, and log out, and we have around 30 routers that we need to get this information from, we would need 10\*30 = 300 seconds for the program to complete the execution. If we are looking for more advanced or complex calculations on each output, which might take up to a minute, then it will take 30 minutes for just 30 routers.

This starts becoming very inefficient when our complexity and scalability grows. To help with this, we need to add parallelism to our programs.  Let us log in to each of the routers and fetch the show version using a parallel calling (or multithreading):



Output:



# Using JSON with Netmiko

Our JSON file looks like this:

# 

Code:



# Multithreading in detail

**What is multithreading?**

* **Multithreading** is defined as the ability of a processor to execute multiple threads concurrently.

Lets’s see an example of simple program and later we will see it using multi-threading.



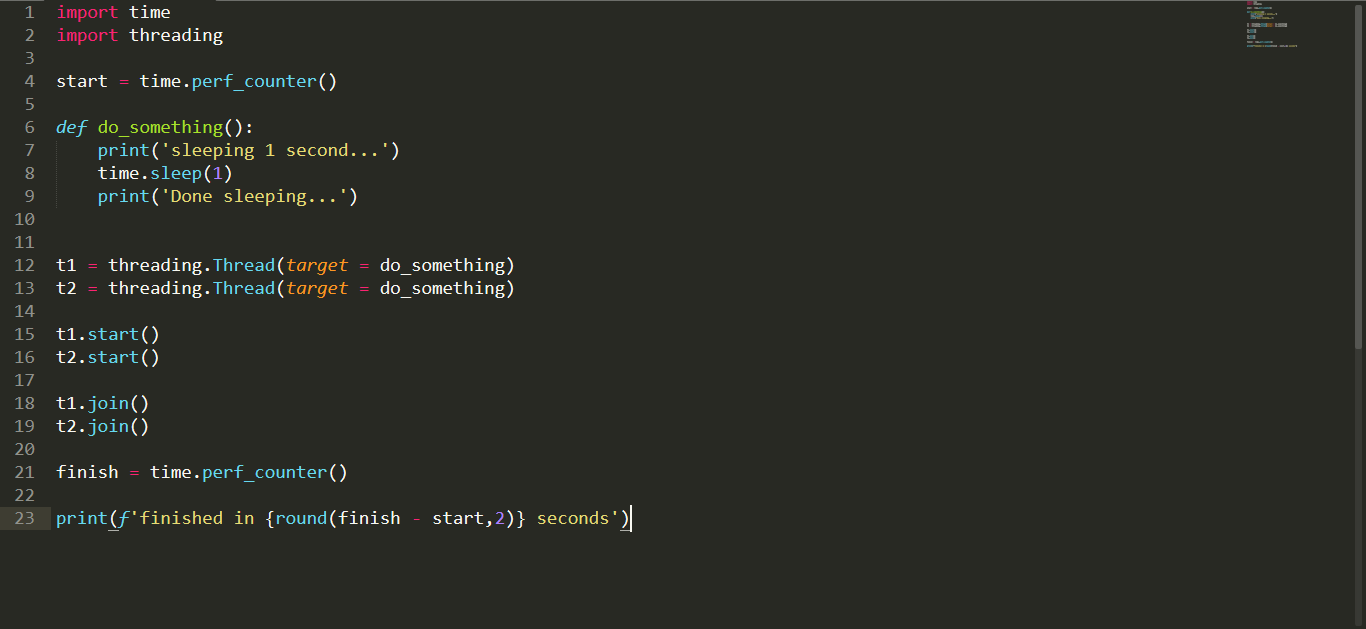
Output:



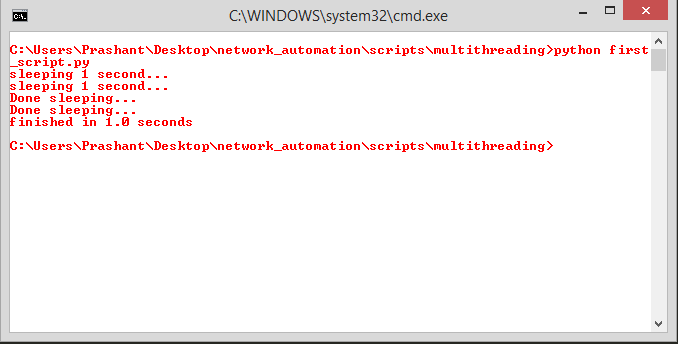
As you can see, our script is taking 2 seconds to execute completely.

Now we will do this using multithreading

Script:

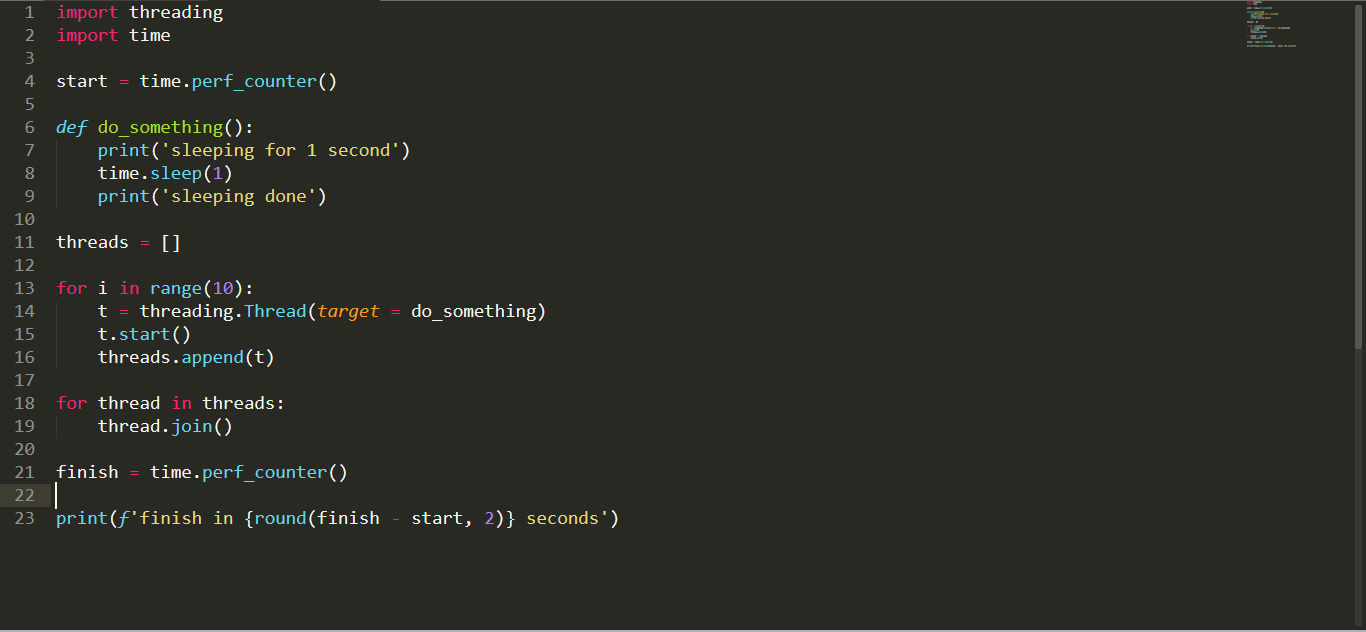


Output:

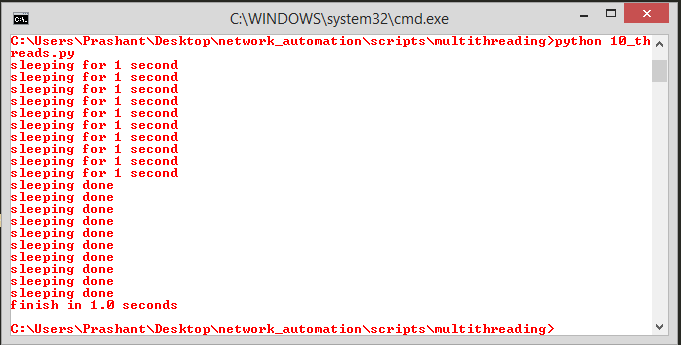


As you can see, We did the same task in 1 second using multithreading.

Let’s make 10 threads and see how much will it take?



Output:

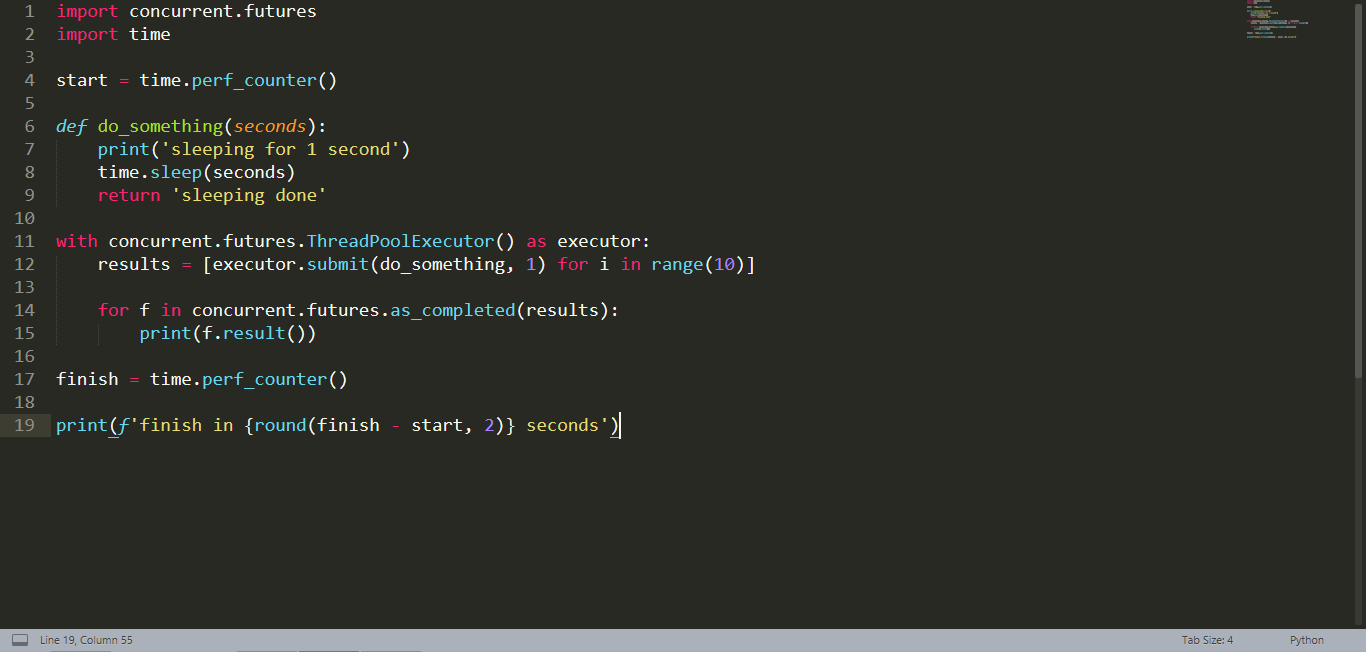


It has also been done in 1 second.

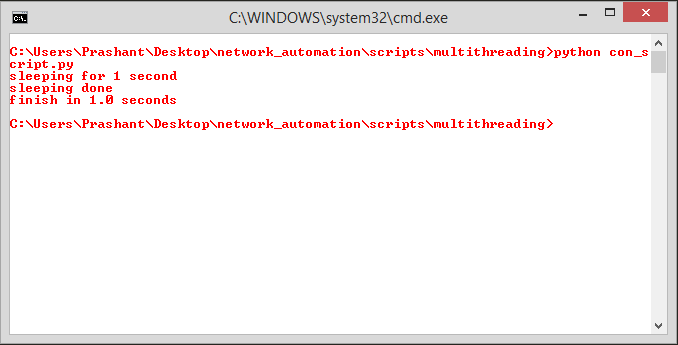
Now it is quite understandable that using multithreading can save our a lot of time.

To do the same task in less number of line and in much clear way, we can use threadpoll module:

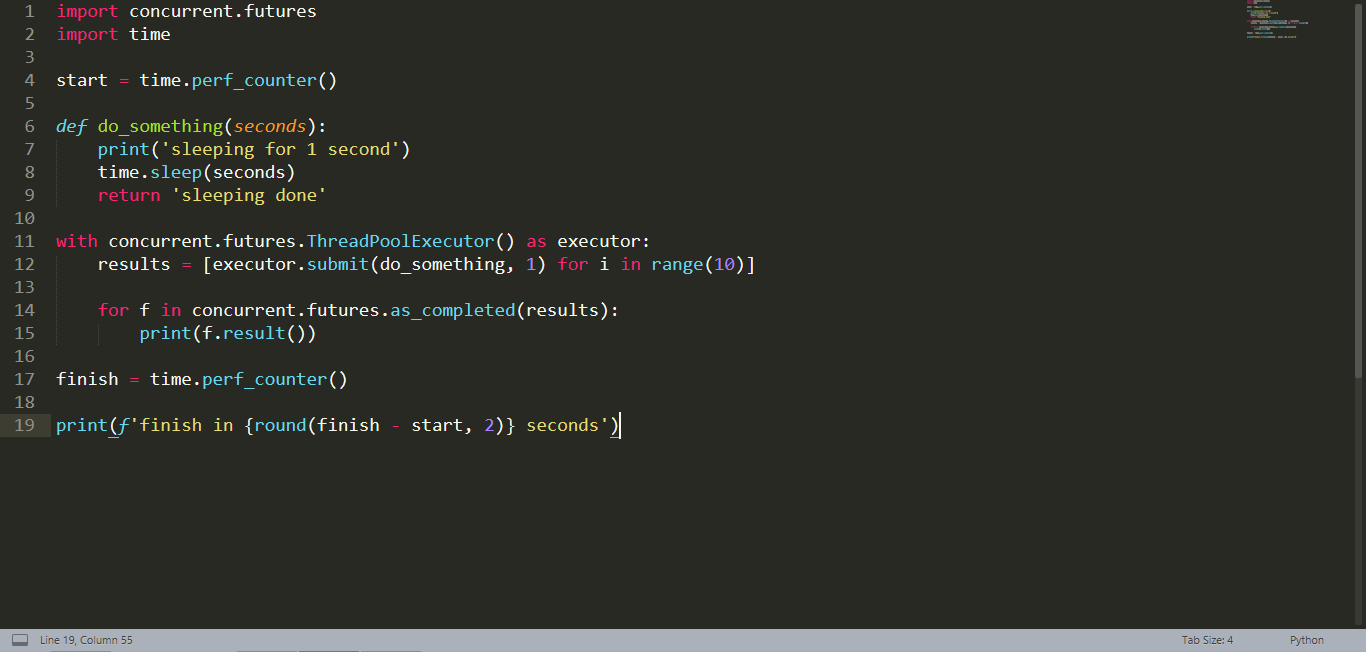
Script:



Output:



Let’s try it for multiple thread:



Output:

