Step-by-step Lab Guide to Learn Python Network Automation:

Lab Details

How to Access Our Lab:

Add RDP Details
Ubuntu Lab Diagram with Device List and IPs
[Lab credentials will be provided here]

Lab Device List and Credentials

Ubuntu : 1
Cisco vIOS Router: 2
Cisco vIOS Switch: 1
Cisco CSR1000v: 1

Instance Type	Access Methods	Name	Console Name	Lab IPs	Credentials
Ubuntu 22.04	RDP		NA	10.1.1.10	via email
Cisco:vIOS-R1	SSH,Console	r1	con-r1	10.1.1.21	admin/admin
Cisco:vIOS-R2	SSH,Console	r2	con-r2	10.1.1.22	admin/admin
Cisco:vIOS-SW1	SSH,Console	sw1	con-sw1	10.1.1.23	admin/admin
Cisco:CSR1000v	SSH,Console, NETCONF, REST-API	csr1	con-csr1	10.1.1.31	admin/admin
Fortigate 7.0	SSH, HTTP, REST-API	fgt1	Yet to	10.1.1.41	admin/password123

Access Direct Console of the Devices:

(supported : vIOS-R,vIOS-Switch, CSR1000v)

From the Ubuntu Machine terminal type “con-” and press tab

Which will list all the devices which supports direct Console: give the name and press “Enter”

```
user1@user1-virtual-machine:~/Desktop/scripts$ con-
con-csr1  con-r1    con-r2
```

```
user1@user1-virtual-machine:~/Desktop/scripts$ con-r1
Trying 10.1.1.11...
Connected to 10.1.1.11.
Escape character is '^]'.

Router>
Router>
Router>
Router>
```

How to Close the terminal console Connection:
In the Console type “ Ctrl+] ”
Prompt will change from Router> to telnet>
In telnet prompt enter “ close ”

```
user1@user1-virtual-machine:~/Desktop/scripts$ con-r1
Trying 10.1.1.11...
Connected to 10.1.1.11.
Escape character is '^]'.

Router>
Router>
Router>
Router>
telnet> close
Connection closed.
user1@user1-virtual-machine:~/Desktop/scripts$
```

1. Understand Your Ubuntu:

[]

- [ ] If you are familiar with Ubuntu basic file and terminal operations pls skip this section

Basic commands for navigating through files and folders

[]

Reading and writing file logic

[]

Basic tasks: add or remove contents to a file. Basic file permissions

[]

2. Introduction to default Python executor

[]

3. Initial Packages required for Automation

Note: These are installed in Lab-ubuntu

3.1 Update:

```
sudo apt-get update
```

3.2 Install nettools:

```
sudo apt install net-tools
```

3.3 Openssh server:

```
sudo apt install openssh-server
systemctl restart sshd
```

3.4 Add older SSH algorithm support in Latest Ubuntu:

[]

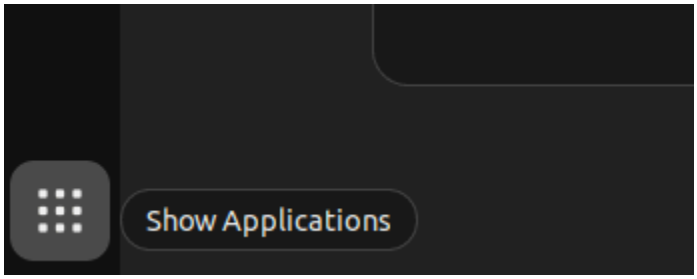
```
# override default of no subsystems
Subsystem       sftp    /usr/lib/openssh/sftp-server

# Example of overriding settings on a per-user basis
#Match User anoncvs
#       X11Forwarding no
#       AllowTcpForwarding no
#       PermitTTY no
#       ForceCommand cvs server
#KexAlgorithms diffie-hellman-group1-sha1,curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-s
ha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-group14-sha1
-- INSERT --                                                                    123,2         Bot
```

4. Installation and Setup of Pycharm

4.1 Check Pycharm is already Installed:

- ☐ Click on the Bottom left ”Show application button in Ubuntu desktop”, search for PyCharm



- ➢ If PyCharm is present and you want to try Pycharm installation from scratch: Proceed to Uninstall Existing PyCharm
- ➢ If PyCharm is installed and you are familiar with installation you can proceed to How to install libraries: (Netmiko, Paramiko)
- ➢ If PyCharm is not installed, proceed to PyCharm snap package Installation

4.2 Uninstall Existing PyCharm

- ☐ Open scripts directory in Desktop
- ☐ Right Click -> Open in terminal:
- ☐ Execute below command

```
sudo ./remove_pycharm.py
```

```
user1@user1-virtual-machine:~/Desktop/scripts$ sudo ./remove_pycharm.py
pycharm-community removed
user1@user1-virtual-machine:~/Desktop/scripts$
```

- ➢ This will remove Pycharm projects and config files

4.3 Make sure python3-virtualenv is installed in the Machine

- ☐ This is necessary for PyCharm to setup dedicated virtual environments for the Projects

```
sudo apt install python3-virtualenv
```

4.4 PyCharm snap package Installation

- ☐ Install using snap package is the easiest method in Linux.

reference:https://www.jetbrains.com/help/pycharm/installation-guide.html

Install using snap packages

1. For Ubuntu 16.04 and later, you can use snap packages to install PyCharm.

   PyCharm is distributed via two channels:

   - The *stable* channel includes only stable versions. To install the latest stable release of PyCharm, run the following command:

     Professional Edition | Community Edition | Edu Edition

     ```
     sudo snap install pycharm-community --classic
     ```

     The `--classic` option is required because the PyCharm snap requires full access to the system, like a traditionally packaged application.

```
sudo snap install pycharm-community --classic
```

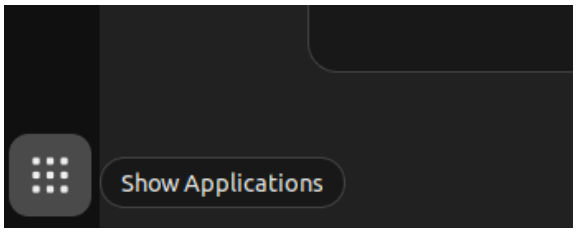
- ☐ It will start downloading Pycharm from the internet, wait for this to get completed.
- ☐ Once it is completed you will get

```
pycharm-community <version> from jetbrains** installed
```

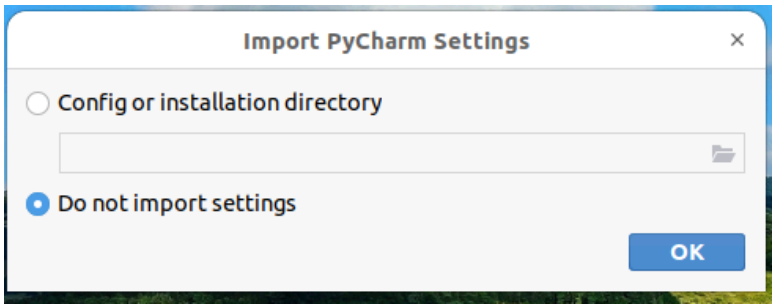
```
user1@user1-virtual-machine:~$ sudo vi /etc/ssh/sshd_config
user1@user1-virtual-machine:~$ sudo snap install pycharm-community --classic
Download snap "pycharm-community" (281) from channel "stable"
Download snap "pycharm-community" (281) from channel "stable"                    14% 7.07MB/s 1m12s
Download snap "pycharm-community" (281) from channel "stable"                    14% 6.99MB/s 1m13s
pycharm-community 2022.1.2 from jetbrains** installed
user1@user1-virtual-machine:~$
```

4.5 Setup Pycharm Environment

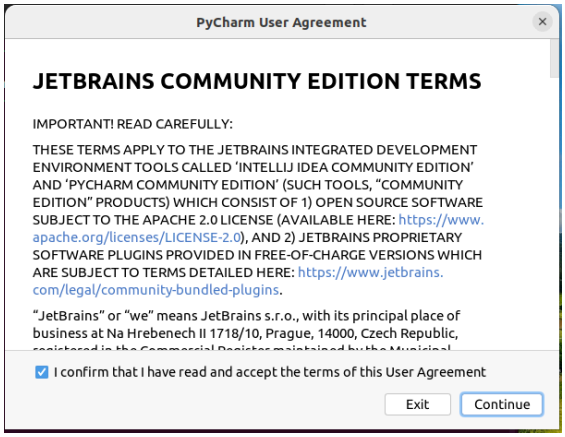
- ☐ Click on Bottom left ”Show application”, search for PyCharm and Open



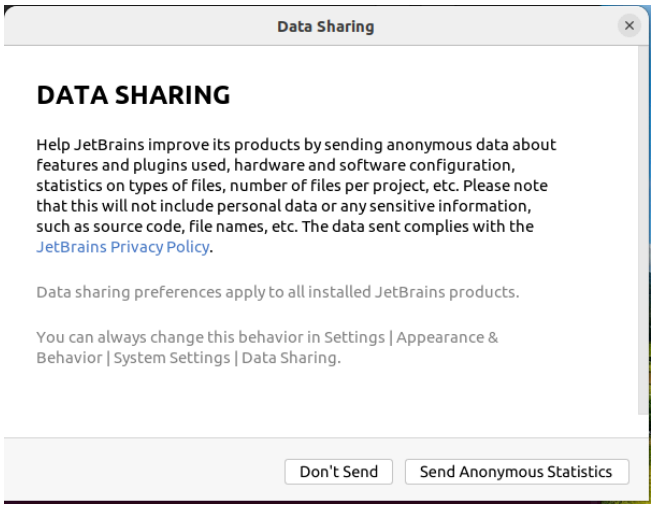
- ☐ If you get Import Pycharm Settings:
  - ➔ Select Do not import settings



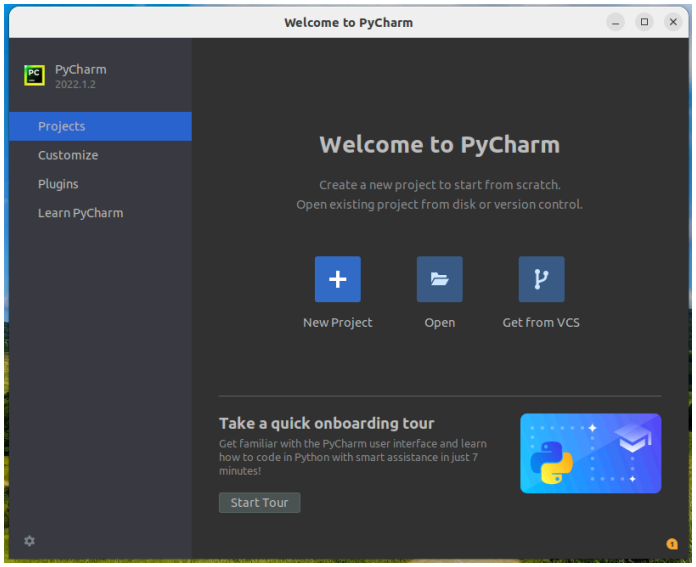
- ☐ Click Continue for Terms:



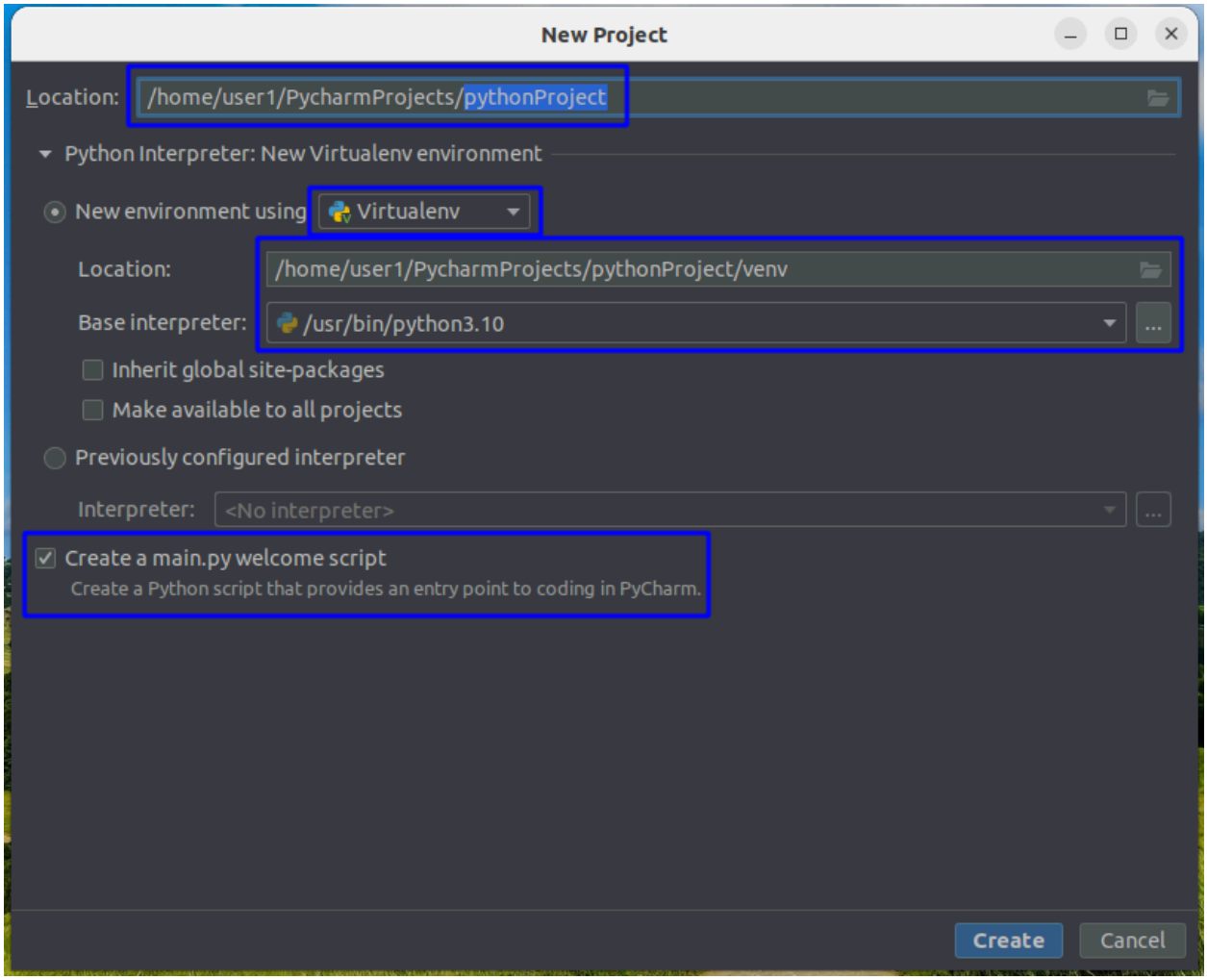
- [ ] Click on Don't Send for Data sharing in our Lab



- [ ] Pycharm Welcome Window will open
- [ ] Click on + New Project:



➔ Below are the Default Options

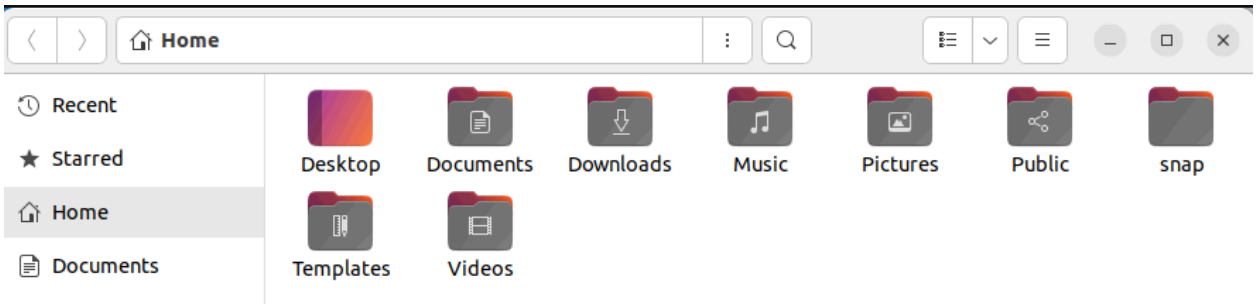


- **Location:** Location for the PycharmProject, this creates new folders in User Home folder
- **New Virtual Environment:** This creates a New Python VirtualEnvironment with name “venv” in PythonProject Folder

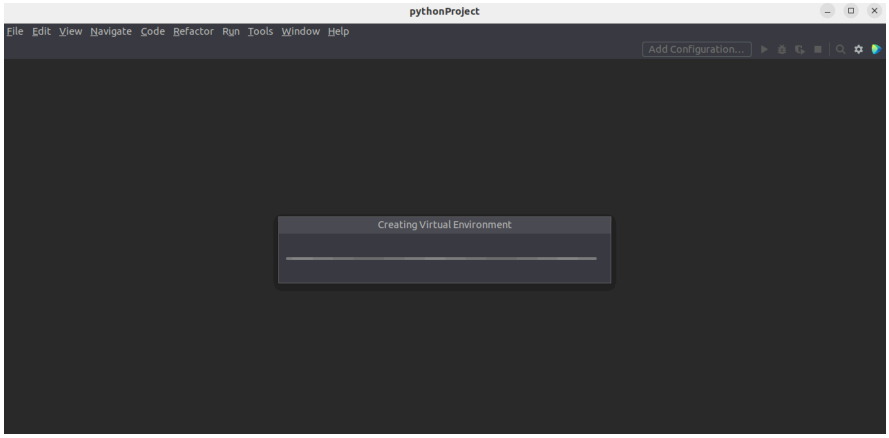
> Note: This will be using python-virtual environment for creating new environment.

- **Base Interpreter:** PyCharm Reads OS default Python and lists here
- **Create main.py :** Once the project is created this adds a main.py file for testing the python

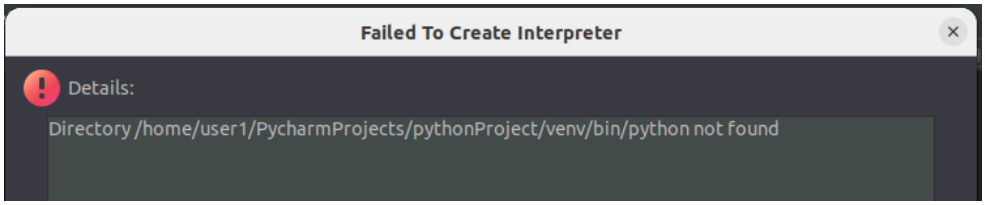
- [ ] Before Installing you can verify User Home Directory:



- [ ] Click on Create: This will start creating a new Project Folder and virtual environment in the Home DIrectory



- ➢ Optional:
  - At this point If you get below error:

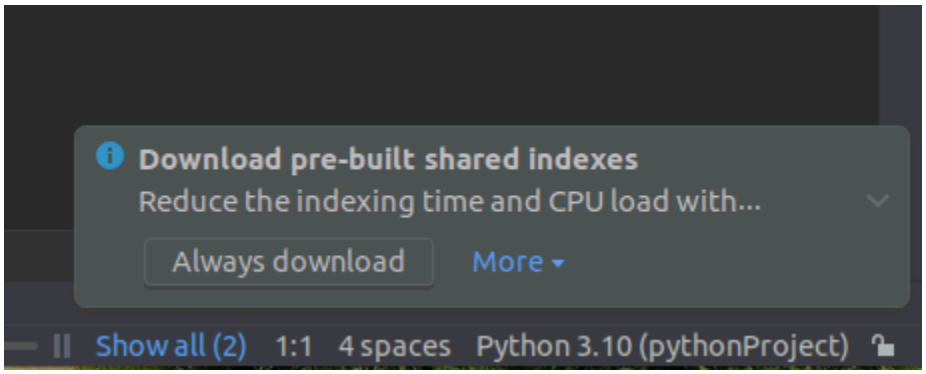


  - ➢ This implies you don't have virtualenv installed in the OS try installing using

```
sudo apt install python3-virtualenv
```

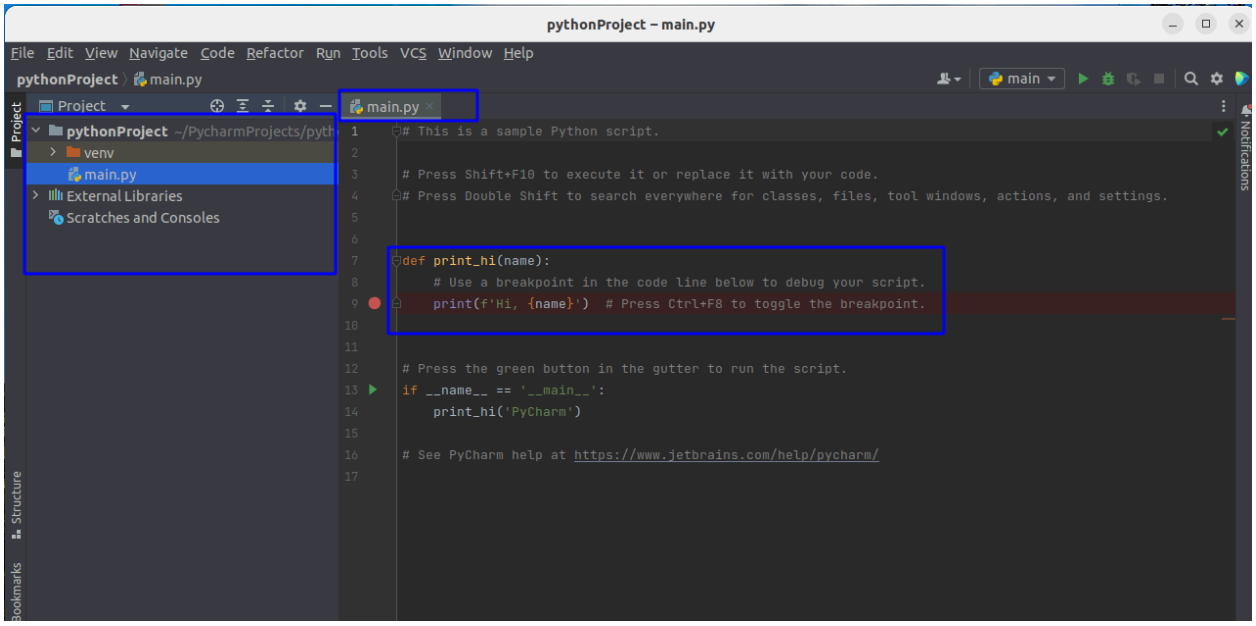
4.6 (Optional)Indexing you can select Always Download:

➢ This is applicable only for larger Pycharm projects



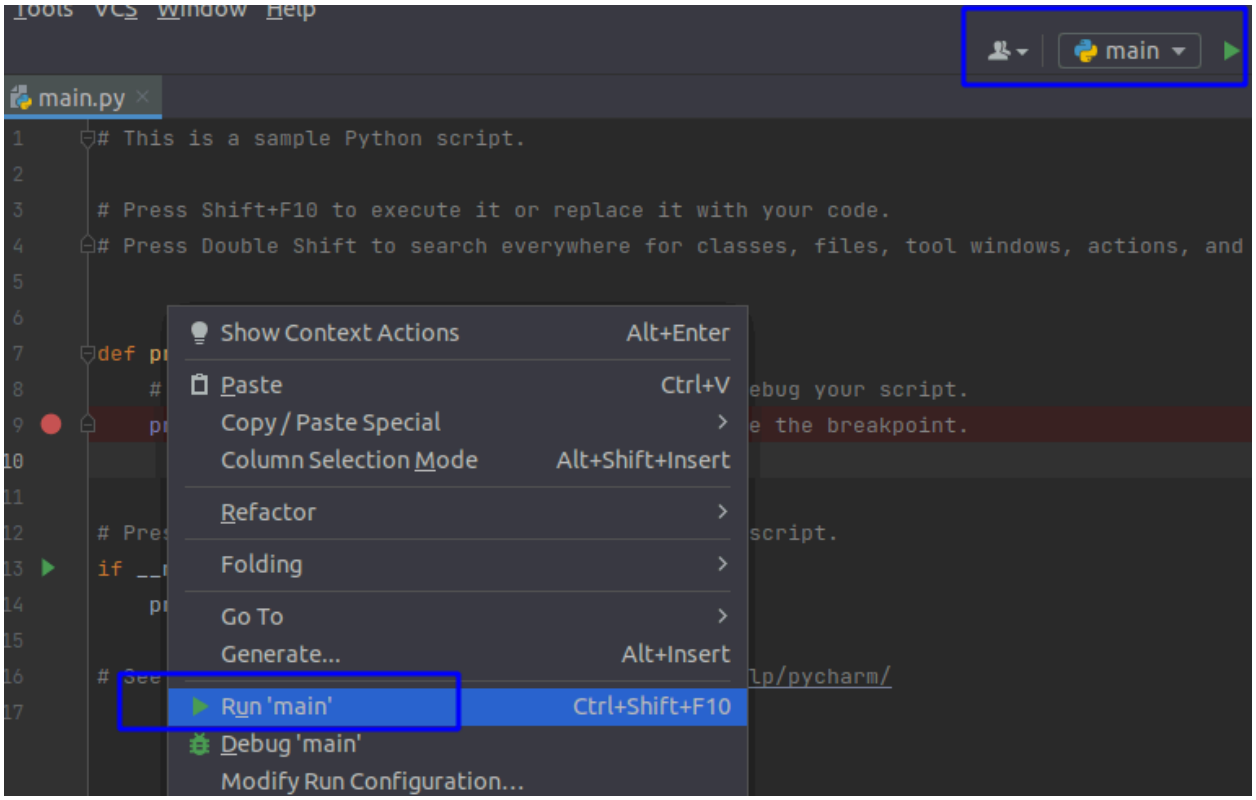
4.7 Execute your first script:

☐ Pycharm Application Window:



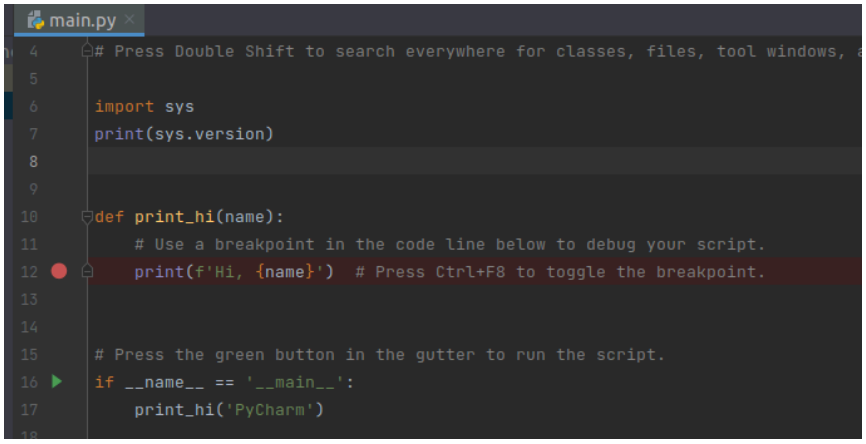
- **Venv:** is the default virtual environment created for the project
- **Main.py :** Sample python file to test first script execution in Pycharm

- ☐ To run the program Either Right-click and Run
  OR
- ☐ Click on Run on top Right



- ☐ If you want to get current Python version of venv and path
- ☐ Add these lines in the code and run the script again

```
import sys
print(sys.version)
```



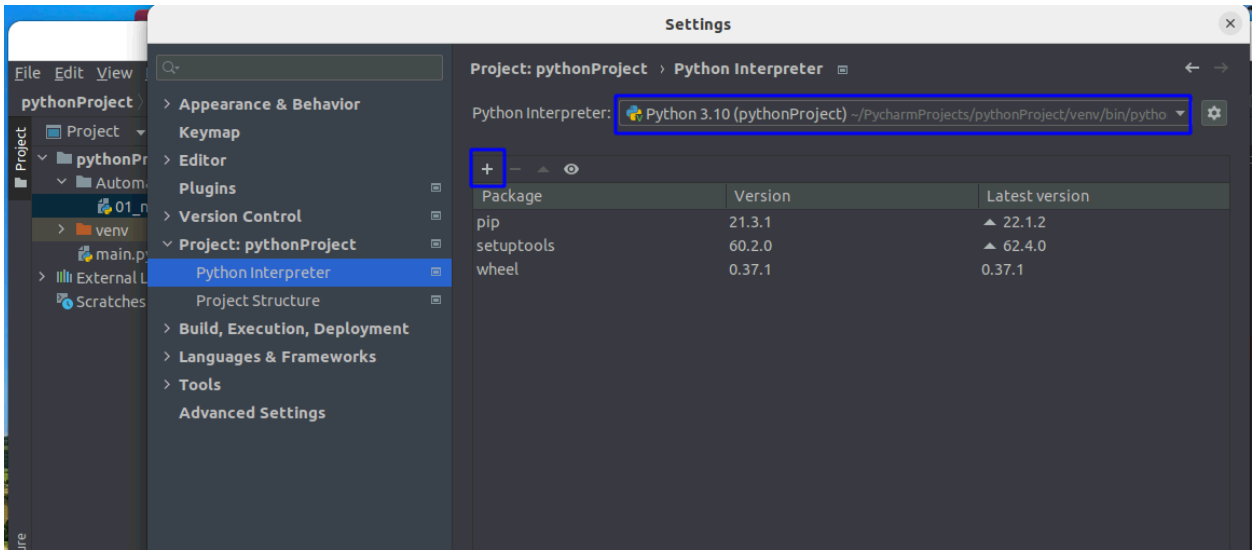
4.8 Explore environment variable settings:

[Will be adding soon]

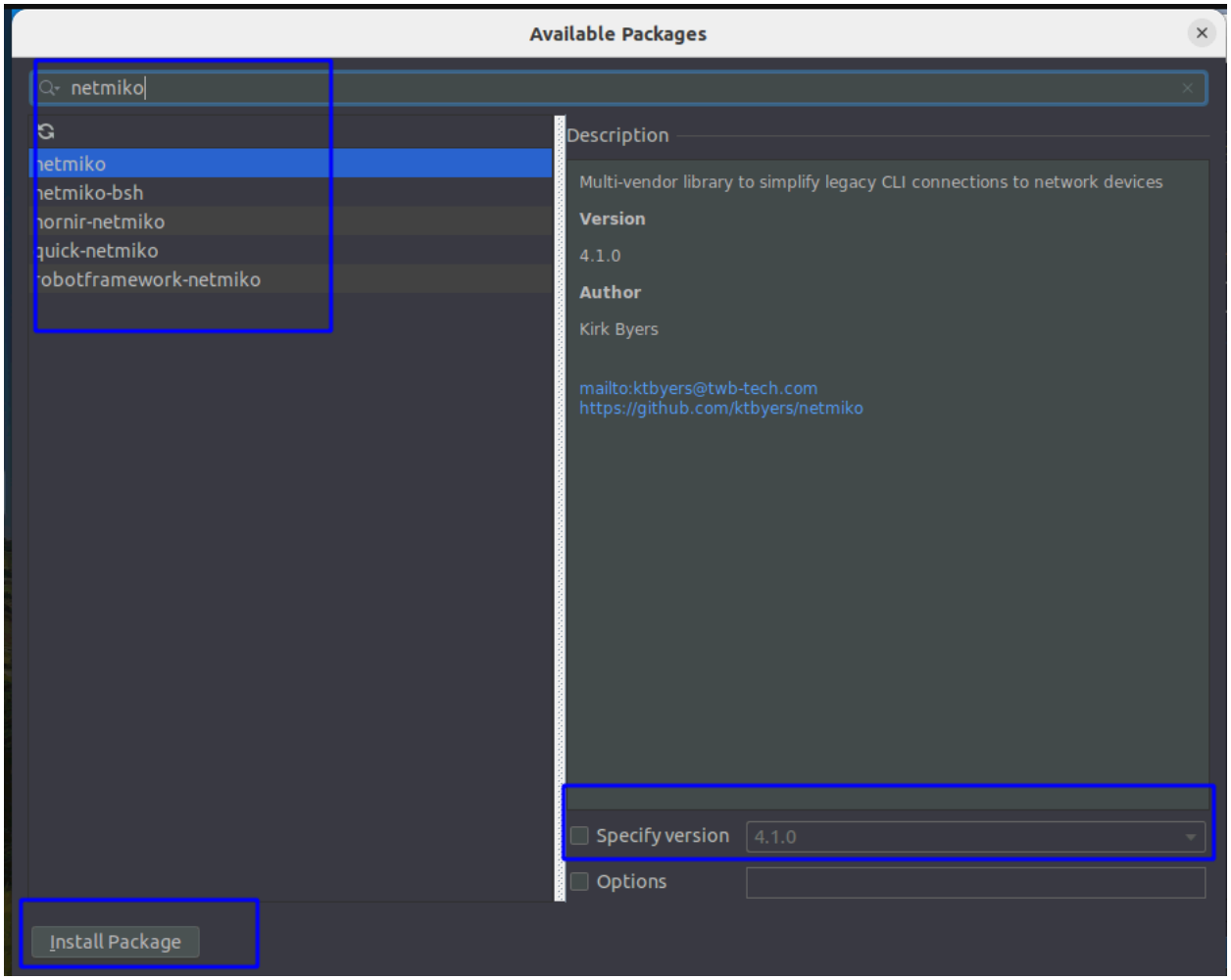
4.9How to install libraries: (Netmiko, Paramiko)

File -> Setting -> PythonProject -> Python Interpreter

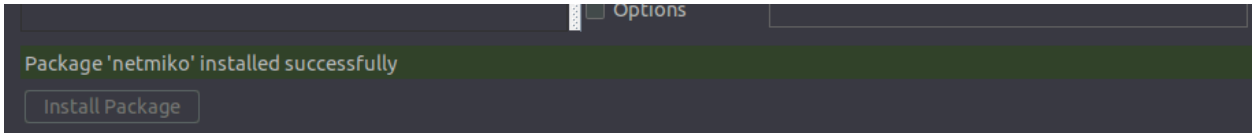
- ➢ You should be able to see the default “venv” which is created by Pycharm during installation



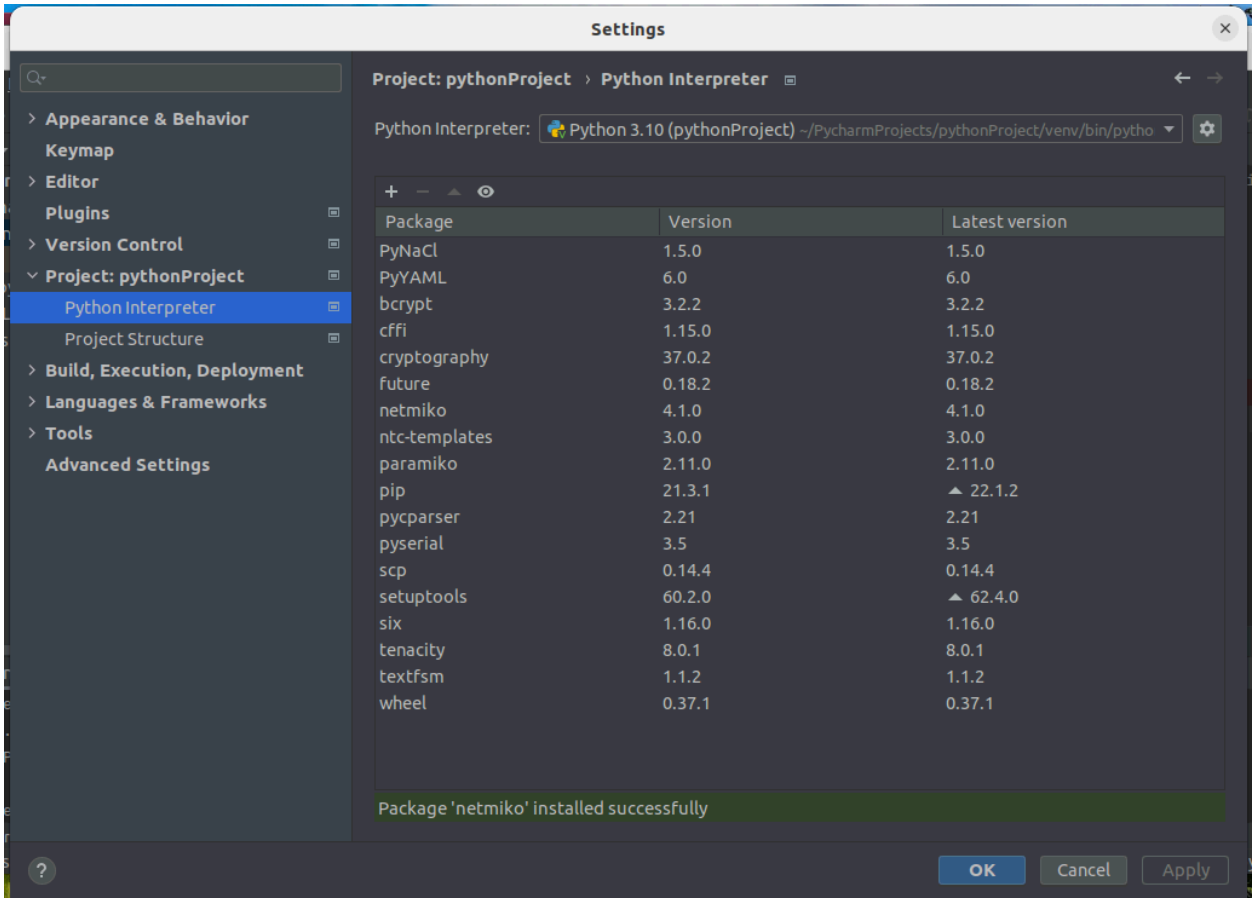
- ☐ **To install a library:** Click on “+”
- ☐ **Search for any library:** eg: netmiko.
  You can leave the Specify version to default. (until you need a specific version of the library)
- ☐ Click on **Install**



- [ ] Once it is successful you will get success message in the bottom:



- [ ] Now verify the current package list with previous package list screen shot
- [ ] This installs all the dependency packages by default.
- [ ] Keep a note of Paramiko (Python base SSH library)
- [ ] Paramiko is necessary for Netmiko to run


Settings
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Project: pythonProject
Python Interpreter
Project Structure
Build, Execution, Deployment
Languages & Frameworks
Tools
Advanced Settings
Project: pythonProject › Python Interpreter
Python Interpreter: Python 3.10 (pythonProject) ~/PycharmProjects/pythonProject/venv/bin/pytho
Package | Version | Latest version
PyNaCl | 1.5.0 | 1.5.0
PyYAML | 6.0 | 6.0
bcrypt | 3.2.2 | 3.2.2
cffi | 1.15.0 | 1.15.0
cryptography | 37.0.2 | 37.0.2
future | 0.18.2 | 0.18.2
netmiko | 4.1.0 | 4.1.0
ntc-templates | 3.0.0 | 3.0.0
paramiko | 2.11.0 | 2.11.0
pip | 21.3.1 | 22.1.2
pycparser | 2.21 | 2.21
pyserial | 3.5 | 3.5
scp | 0.14.4 | 0.14.4
setuptools | 60.2.0 | 62.4.0
six | 1.16.0 | 1.16.0
tenacity | 8.0.1 | 8.0.1
textfsm | 1.1.2 | 1.1.2
wheel | 0.37.1 | 0.37.1
Package 'netmiko' installed successfully
OK
Cancel
Apply

5. Your First Netmiko Script in the Lab

- ☐ Create a New folder “Automation_Scripts”Under PythonProject:

5.1 Backup Devnet Router Config

- ☐ Create a new Python file “01_netmiko_devnet.py” under “Automation_Scripts”
- ☐ Copy paste the below code in the file and run the script

```
from netmiko import Netmiko

my_router = Netmiko(ip='ios-xe-mgmt-latest.cisco.com',
                    username='developer',
                    password='C1sco12345',
                    device_type='cisco_ios')

print(my_router.find_prompt())
print("Connected successfully")
show_run = my_router.send_command('show run')
print(show_run)
with open('devnet_backup.txt', 'w') as my_data:
   my_data.write(show_run)

my_router.disconnect()
```

- ☐ Script creates backup.txt file with device’s current config

5.2 Backup Lab Router Config

- ☐ Create a new python file in same folder replacing devnet with lab, and the content in the file and run the script

➔ IP: 10.1.1.21
➔ Username: admin/admin

```
from netmiko import Netmiko

# my_router = Netmiko(ip='ios-xe-mgmt-latest.cisco.com',
#                     username='developer',
#                     password='C1sco12345',
#                     device_type='cisco_ios')

my_router = Netmiko(ip='10.1.1.21',
                    username='admin',
                    password='admin',
                    device_type='cisco_ios')
print(my_router.find_prompt())
print("Connected successfully")
show_run = my_router.send_command('show run')
print(show_run)
with open('backup1.txt', 'w') as my_data:
   my_data.write(show_run)
```