What is Ansible?

Ansible is an open-source automation tool for managing remote systems for example by eliminating repetitive tasks, managing and maintaining system configuration, deploying complex software on many machines, rolling updates without downtime etc. It uses scripts called playbooks to automate these tasks. (Ansible Community Documentation, 2025)

To set up ansible we need 3 components:

- Control node which is a system on which ansible is installed.
- Inventory which is a list of managed nodes created on the control node. It describes host deployments to Ansible.
- Managed node which is a remote system/host controlled by Ansible. (Ansible Community Documentation, 2025)

Test environment to set up Ansible

The test environment includes:

Ubuntu server installed on hyper-v which will serve as a control node. - 192.168.100.2/24

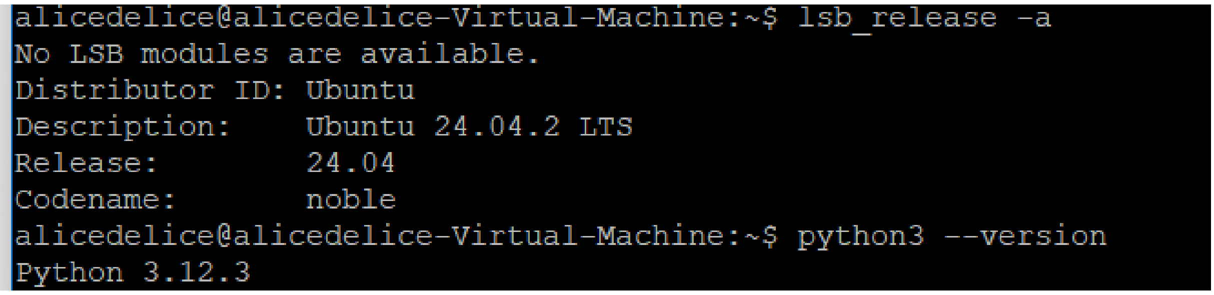
1st Oracle linux installed on hyper-v which will serve as a remote system - 192.168.100.4/24

2nd Oracle linux installed on hyper-v which will serve as a remote system - 192.168.100.5/24

All 3 machines can communicate with each other since they are connected with the same internal network adapter. They also have external network adapter to be able to connect to the internet.

Ansible

According to (Ansible Community Documentation, 2025) the control node needs Unix-like machine (in my case it's Ubuntu) with python installed. I have already python installed which can be checked with command **python3 –version**. Ubuntu version can be checked with command **lsb_release -a**.



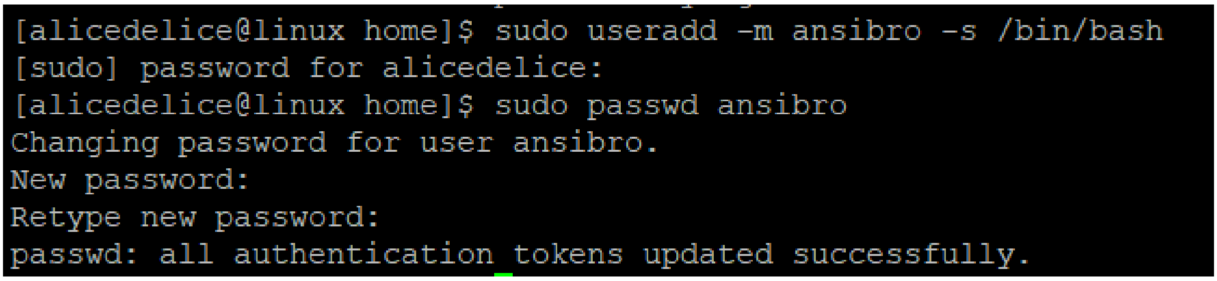
```
alicedelice@alicedelice-Virtual-Machine:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 24.04.2 LTS
Release:        24.04
Codename:       noble
alicedelice@alicedelice-Virtual-Machine:~$ python3 --version
Python 3.12.3
```

Managed nodes do not need ansible to be installed but they require python to run Ansible-generated Python code. Additionally, they need a user account that can connect through SSH to the node with interactive POSIX shell. (Ansible Community Documentation, 2025) As (IEEE, 2001-2018) "POSIX.1-2017 defines a standard operating system interface and environment, including a command interpreter (or “shell”), and common utility programs to support applications portability at the source code level.”

SSH User

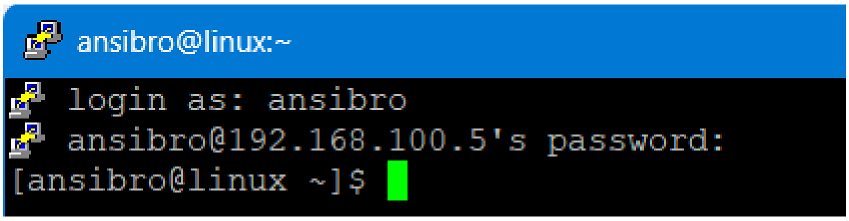
A username used on a control node is used by Ansible to connect to remote devices. (Ansible Community Documentation, 2025)

Created new “technical” user **ansibro** on control node and on hosts machines with command **useradd** which will be used to connect to remote devices



```
[alicedelice@linux home]$ sudo useradd -m ansibro -s /bin/bash
[sudo] password for alicedelice:
[alicedelice@linux home]$ sudo passwd ansibro
Changing password for user ansibro.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

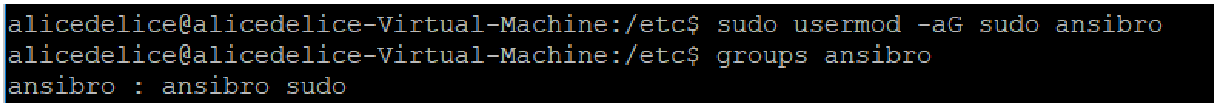
Checked that I can connect to machines with the new user:



```
login as: ansibro
ansibro@192.168.100.5's password:
[ansibro@linux ~]$
```

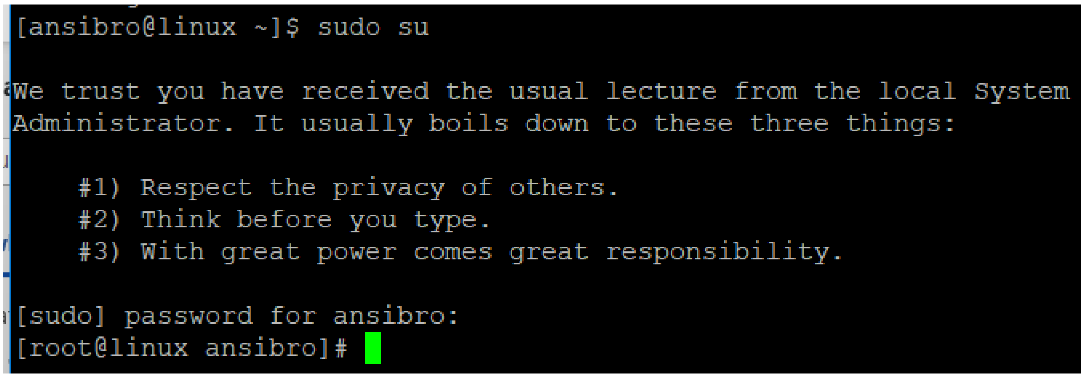
Added user ansibro to sudo group with command **sudo usermod -aG sudo ansibro**

I can see with command **groups ansibro** that user has the sudo group now on Ubuntu.



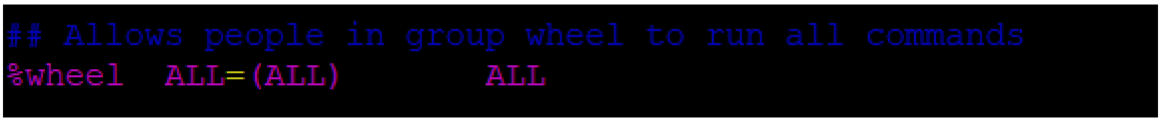
```
alicedelice@alicedelice-Virtual-Machine:/etc$ sudo usermod -aG sudo ansibro
alicedelice@alicedelice-Virtual-Machine:/etc$ groups ansibro
ansibro : ansibro sudo
```

On oracle linux ***wheel*** group is used instead of ***sudo***.

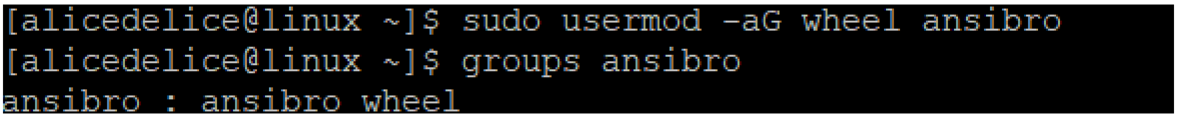


```
[ansibro@linux ~]$ sudo su

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for ansibro:
[root@linux ansibro]#
```

(can be checked in /etc/sudoers file that it grants access to sudo)



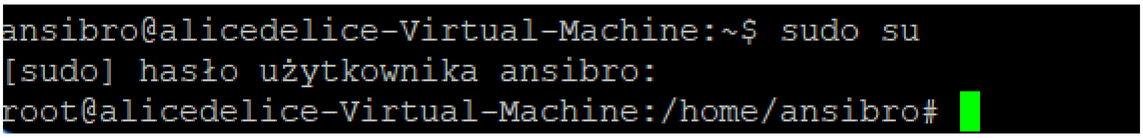
```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```



```
[alicedelice@linux ~]$ sudo usermod -aG wheel ansibro
[alicedelice@linux ~]$ groups ansibro
ansibro : ansibro wheel
```

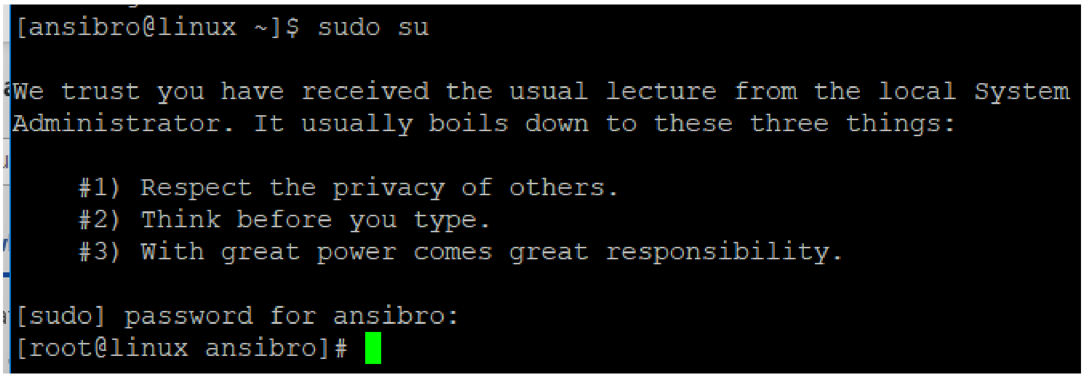
Checked if sudo command is working:

Ubuntu:



```
ansibro@alicedelice-Virtual-Machine:~$ sudo su
[sudo] hasło użytkownika ansibro:
root@alicedelice-Virtual-Machine:/home/ansibro#
```

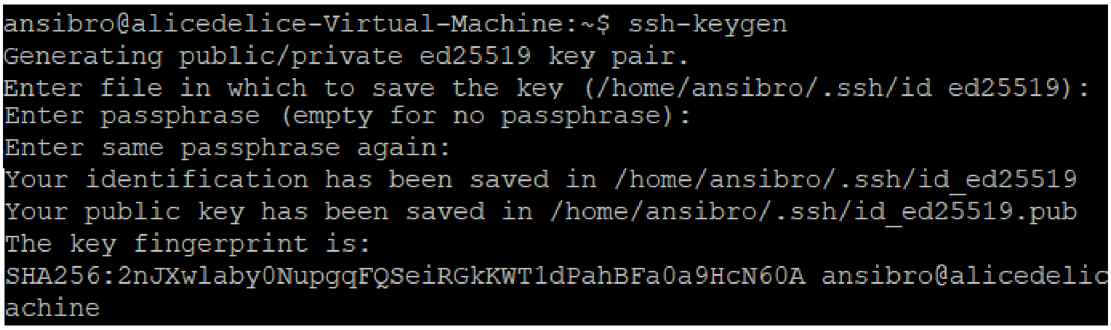
Oracle:



```
[ansibro@linux ~]$ sudo su

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for ansibro:
[root@linux ansibro]#
```

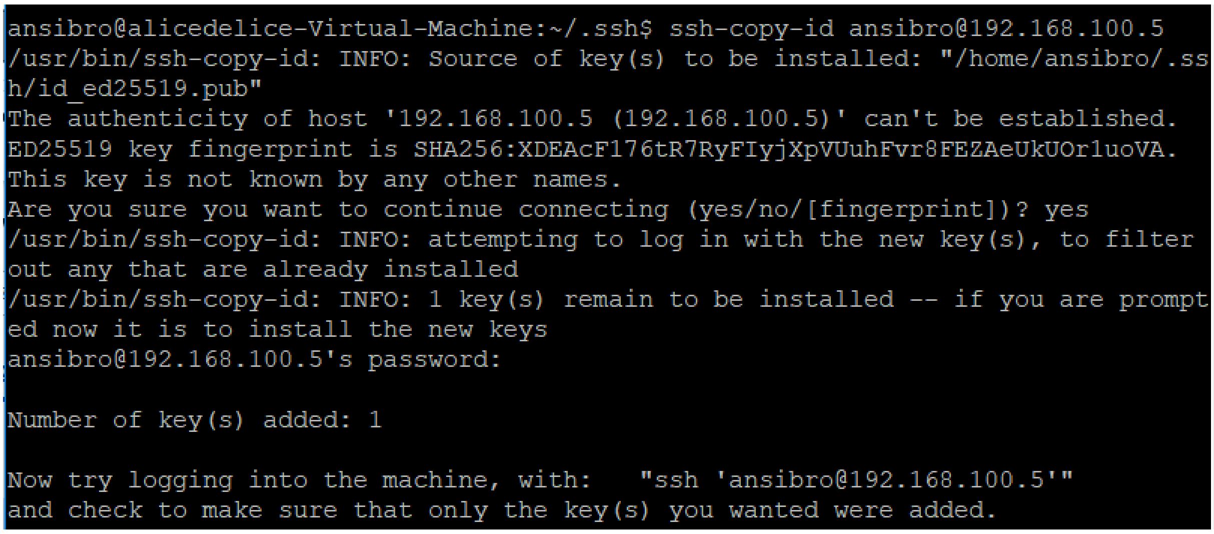
Generated ssh key pair with **ssh-keygen** on Ubuntu machine (control node).

I didn’t create passphrase (additional protection for authentication) since with automation a passphrase would have to be hard-coded in a script or stored in some kind of vault from which it should be retrieved by a script. (SSH Academy, 2025)



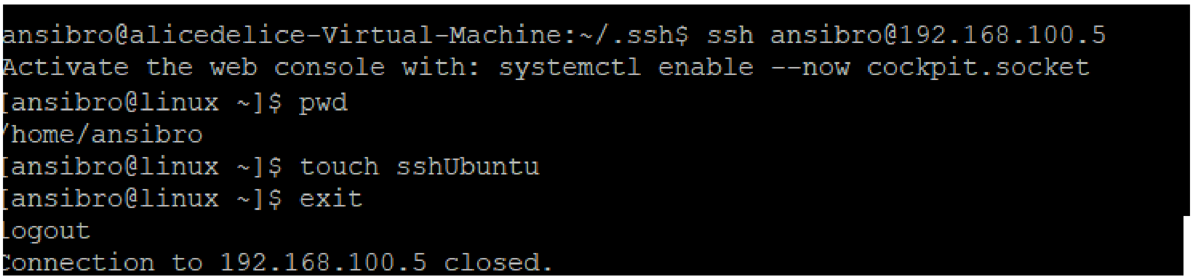
```
ansibro@alicedelice-Virtual-Machine:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ansibro/.ssh/id ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ansibro/.ssh/id_ed25519
Your public key has been saved in /home/ansibro/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:2nJXwlaby0NupgqFQSeiRGkKWT1dPahBFa0a9HcN60A ansibro@alicedelic
achine
```

Then copied the public key to both oracle linux servers with command **ssh-copy-id** ansibro@192.168.100.4 and **ssh-copy-id** ansibro@192.168.100.5 (Oracle Linux, 2025)



```
ansibro@alicedelice-Virtual-Machine:~/.ssh$ ssh-copy-id ansibro@192.168.100.5
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ansibro/.ss
h/id_ed25519.pub"
The authenticity of host '192.168.100.5 (192.168.100.5)' can't be established.
ED25519 key fingerprint is SHA256:XDEAcF176tR7RyFIyjXpVUuhFvr8FEZAeUkUOr1uoVA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
ansibro@192.168.100.5's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ansibro@192.168.100.5'"
and check to make sure that only the key(s) you wanted were added.
```

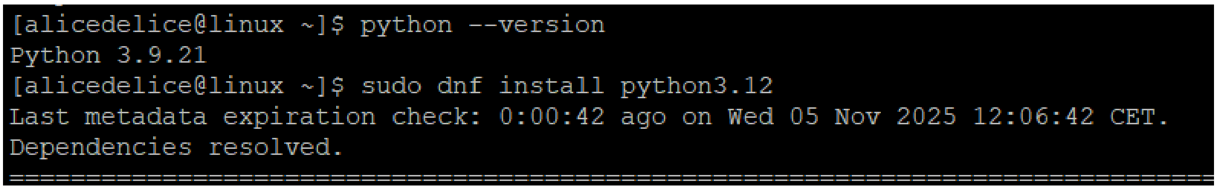
Tested by creating a file in home directory on 192.168.100.5. On below screenshot, it's showed that the connection switched from ansibro@alicedelice-Virtual-Machine to ansibro@linux. Then, after creating the file and exiting, the connection to 192.168.100.5 closed.



```
ansibro@alicedelice-Virtual-Machine:~/.ssh$ ssh ansibro@192.168.100.5
Activate the web console with: systemctl enable --now cockpit.socket
[ansibro@linux ~]$ pwd
/home/ansibro
[ansibro@linux ~]$ touch sshUbuntu
[ansibro@linux ~]$ exit
logout
Connection to 192.168.100.5 closed.
```

Python

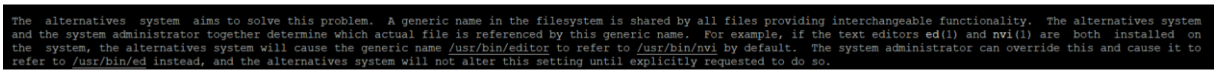
On hosts machines (oracle linux) I had lower version of python 3.9.21, so I used **sudo dnf install python3.12** to download the newest version.



```
[alicedelice@linux ~]$ python --version
Python 3.9.21
[alicedelice@linux ~]$ sudo dnf install python3.12
Last metadata expiration check: 0:00:42 ago on Wed 05 Nov 2025 12:06:42 CET.
Dependencies resolved.
================================================================================
```

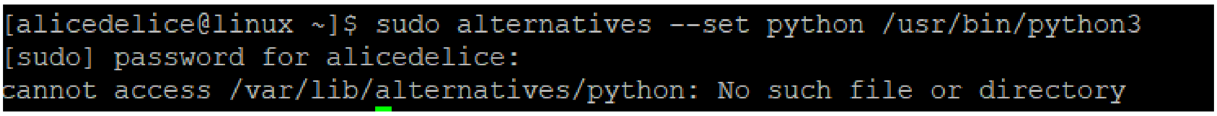
I had to change the version since it kept showing the older one.

I found out guide (Oracle, 2024)that I could use **alternatives** to change the version however the command didn't get any output. This command is used to set up which file should be used by default as I could see in **man alternatives**.



```
The  alternatives  system  aims to solve this problem.  A generic name in the filesystem is shared by all files providing interchangeable functionality.  The alternatives system
and the system administrator together determine which actual file is referenced by this generic name.  For example, if the text editors ed(1) and nvi(1) are  both  installed  on
the  system, the alternatives system will cause the generic name /usr/bin/editor to refer to /usr/bin/nvi by default.  The system administrator can override this and cause it to
refer to /usr/bin/ed instead, and the alternatives system will not alter this setting until explicitly requested to do so.
```

The command **sudo alternatives --set python /usr/bin/python3** from the guide showed error message: ***"cannot access /var/lib/alternatives/python: No such file or directory"***



```
[alicedelice@linux ~]$ sudo alternatives --set python /usr/bin/python3
[sudo] password for alicedelice:
cannot access /var/lib/alternatives/python: No such file or directory
```

As found on forum (Stack Exchange, 2022), first it was needed to install both versions in alternatives:

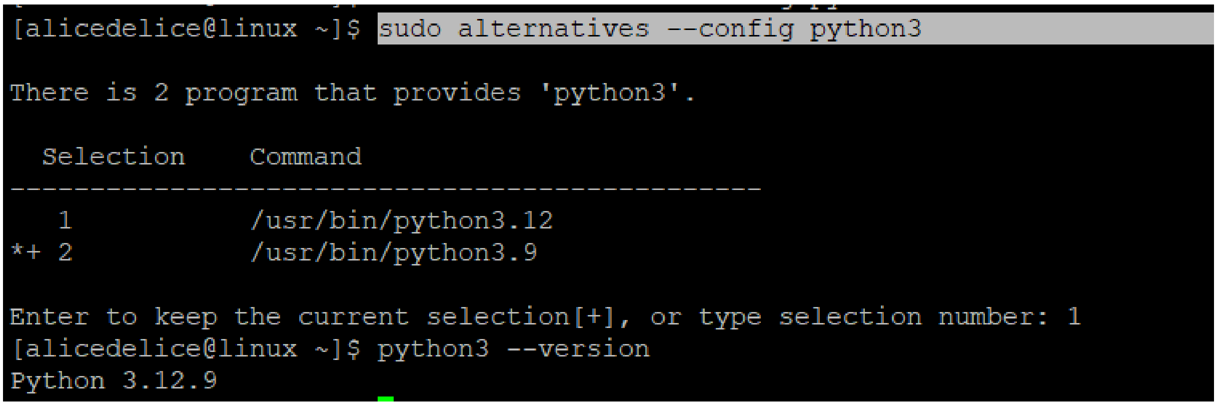
sudo alternatives --install /usr/bin/python3 python3 /usr/bin/python3.12 1

sudo alternatives --install /usr/bin/python3 python3 /usr/bin/python3.9 2

and then in config switched the version to 3.12.9 with command:

sudo alternatives --config python3

Now the version shows 3.12.9.



```
[alicedelice@linux ~]$ sudo alternatives --config python3

There is 2 program that provides 'python3'.

  Selection    Command
-----------------------------------------------
   1           /usr/bin/python3.12
*+ 2           /usr/bin/python3.9

Enter to keep the current selection[+], or type selection number: 1
[alicedelice@linux ~]$ python3 --version
Python 3.12.9
```

Installing Ansible

Installed on ubuntu pipx from (PIPX) since pip command stated in ansible documentation was not available on the machine.

pipx is a tool used to install and run end-user applications written in Python. It adds isolation and still makes the apps available in the shell. It creates an isolated environment for each application and its associated packages. Used the commands for Ubuntu from the "pipx — Install and Run Python Applications in Isolated Environments" website: (PIPX)

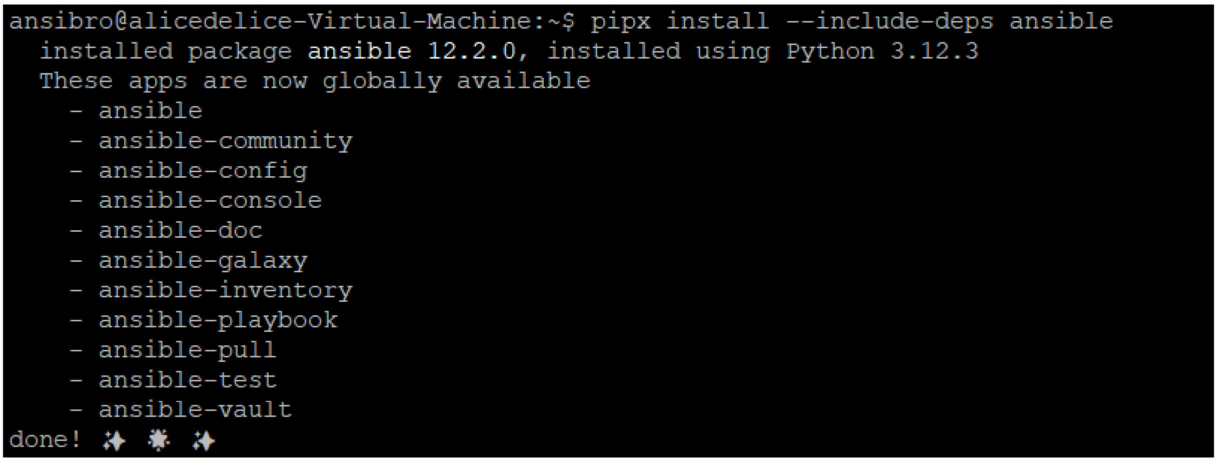
brew install pipx

pipx ensurepath

sudo pipx ensurepath --global # optional to allow pipx actions with --global argument

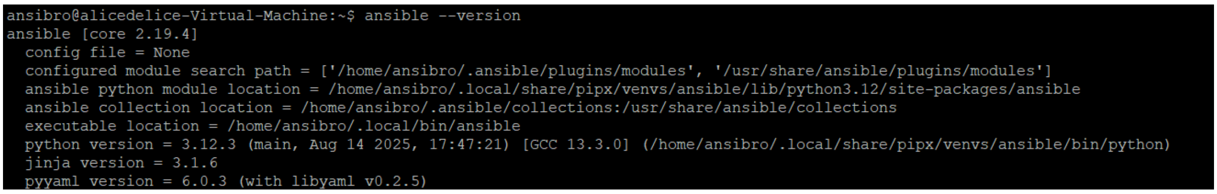
Rebooted the system.

Installed full Ansible package with **sudo pipx install --include-deps ansible** (current version 2.19)



```
ansibro@alicedelice-Virtual-Machine:~$ pipx install --include-deps ansible
  installed package ansible 12.2.0, installed using Python 3.12.3
  These apps are now globally available
    - ansible
    - ansible-community
    - ansible-config
    - ansible-console
    - ansible-doc
    - ansible-galaxy
    - ansible-inventory
    - ansible-playbook
    - ansible-pull
    - ansible-test
    - ansible-vault
done! ✨ 🌟 ✨
```

From the below screenshot I can see that ansible version is 2.19.4 and ansible has been installed in home directory of the ansibro user.



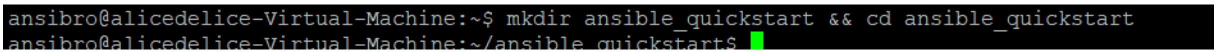
```
ansibro@alicedelice-Virtual-Machine:~$ ansible --version
ansible [core 2.19.4]
  config file = None
  configured module search path = ['/home/ansibro/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /home/ansibro/.local/share/pipx/venvs/ansible/lib/python3.12/site-packages/ansible
  ansible collection location = /home/ansibro/.ansible/collections:/usr/share/ansible/collections
  executable location = /home/ansibro/.local/bin/ansible
  python version = 3.12.3 (main, Aug 14 2025, 17:47:21) [GCC 13.3.0] (/home/ansibro/.local/share/pipx/venvs/ansible/bin/python)
  jinja version = 3.1.6
  pyyaml version = 6.0.3 (with libyaml v0.2.5)
```

On the website (Ansible Community Documentation, 2025) it states it’s possible to install additional python dependencies that may be needed but I skipped those for now. There is also information about installing ansible for development (testing new features), to containers (to use image) etc.

Configuration

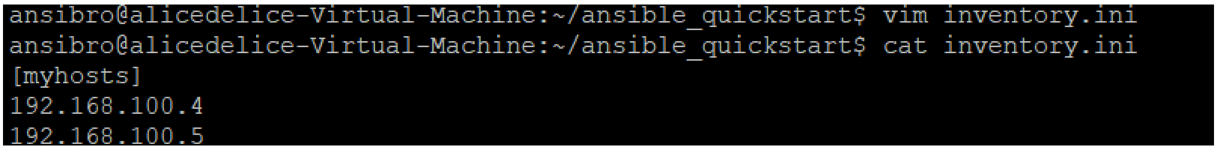
Created directory **ansible_quickstart** as mentioned on ansible documentation website:

mkdir ansible_quickstart && cd ansible_quickstart (Ansible Community Documentation, 2025)



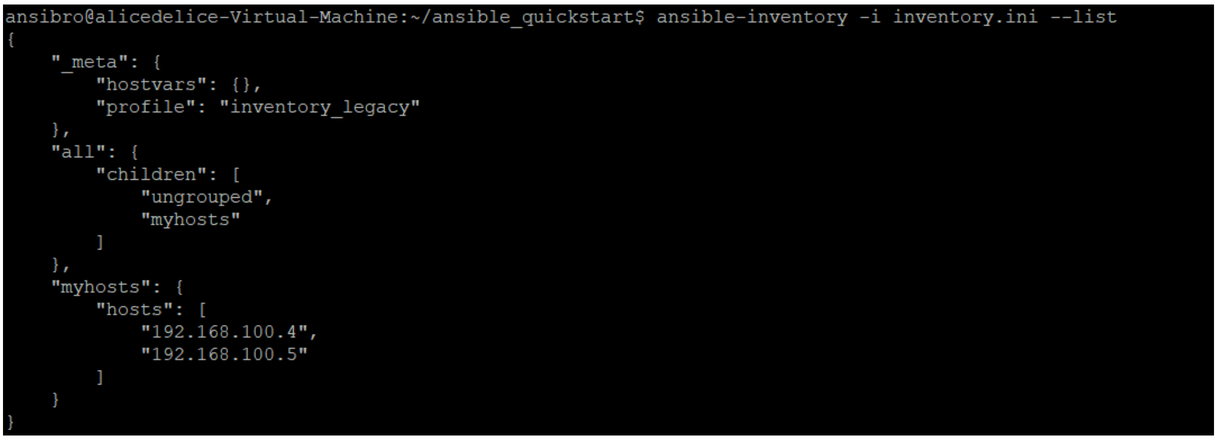
```
ansibro@alicedelice-Virtual-Machine:~$ mkdir ansible_quickstart && cd ansible_quickstart
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ 
```

Created file **inventory.ini** in the mentioned folder and added hosts IP addresses. An inventory file lists all the hosts Ansible can manage and tells it how to connect to them. (Ansible Community Documentation, 2025)



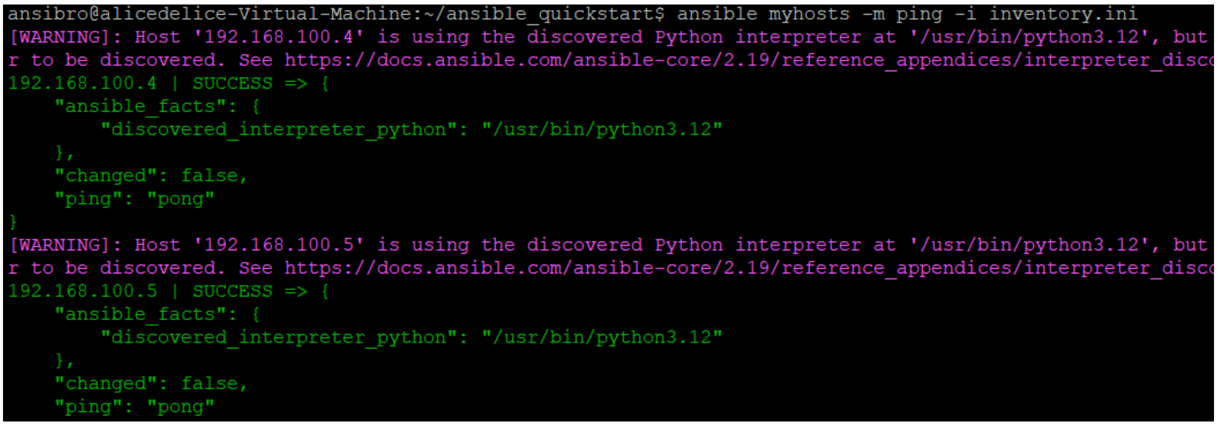
```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ vim inventory.ini
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ cat inventory.ini
[myhosts]
192.168.100.4
192.168.100.5
```

Inventory can be verified with command **ansible-inventory -i inventory.ini --list**



```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ ansible-inventory -i inventory.ini --list
{
    "_meta": {
        "hostvars": {},
        "profile": "inventory_legacy"
    },
    "all": {
        "children": [
            "ungrouped",
            "myhosts"
        ]
    },
    "myhosts": {
        "hosts": [
            "192.168.100.4",
            "192.168.100.5"
        ]
    }
}
```

It’s possible now to check ping connection with myhosts group provided in the **inventory.ini** file with command **ansible myhosts -m ping -i inventory.ini**



```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ ansible myhosts -m ping -i inventory.ini
[WARNING]: Host '192.168.100.4' is using the discovered Python interpreter at '/usr/bin/python3.12', but
r to be discovered. See https://docs.ansible.com/ansible-core/2.19/reference_appendices/interpreter_disco
192.168.100.4 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.12"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Host '192.168.100.5' is using the discovered Python interpreter at '/usr/bin/python3.12', but
r to be discovered. See https://docs.ansible.com/ansible-core/2.19/reference_appendices/interpreter_disco
192.168.100.5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.12"
    },
    "changed": false,
    "ping": "pong"
```

Ping was successful as it’s visible on screenshot.

There is also a warning that says:

[WARNING]: Host '192.168.100.4' is using the discovered Python interpreter at '/usr/bin/python3.12', but future installation of another Python interpreter could cause a different interpreter to be discovered. See https://docs.ansible.com/ansible-core/2.19/reference_appendices/interpreter_discovery.html for more information.

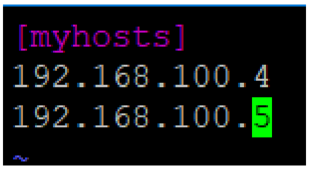
Ansible automatically tries to discover which Python interpreter to use on each host.

If the version changes in future, it could cause Ansible to pick a different interpreter, which might break modules.

It may be solved for example by creating variable in playbook or other files. (Ansible Core documentation, 2025)

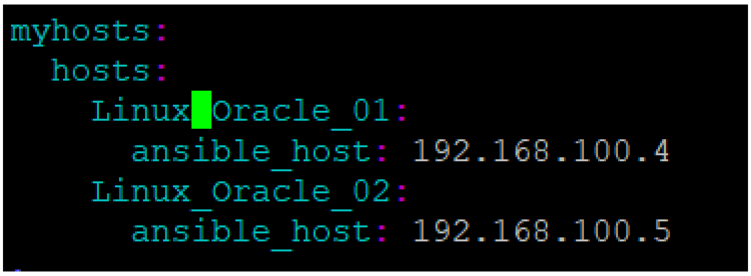
Inventories can be either created in ini or yaml For a few hosts, ini files are simpler and easier to read.

inventory.ini



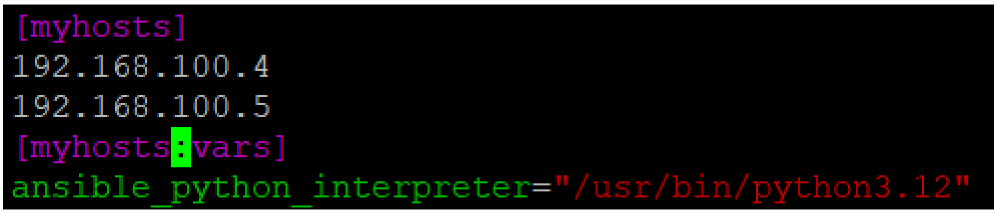
```
[myhosts]
192.168.100.4
192.168.100.5
~
```

inventory.yaml



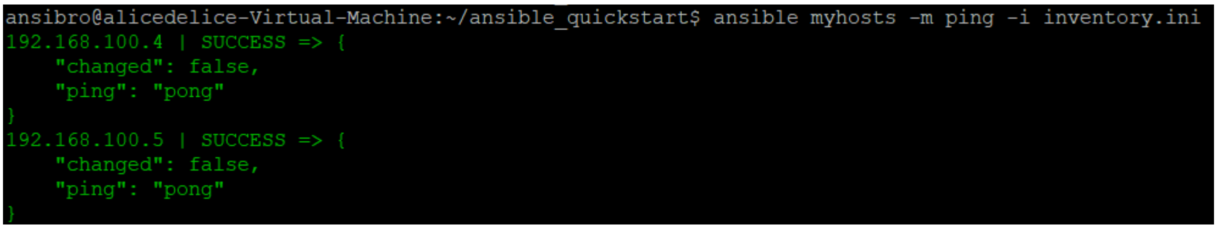
```
myhosts:
  hosts:
    Linux_Oracle_01:
      ansible_host: 192.168.100.4
    Linux_Oracle_02:
      ansible_host: 192.168.100.5
```

I modified inventory.ini with **ansible_python_interpreter** variable (vars) and ran the ping again:



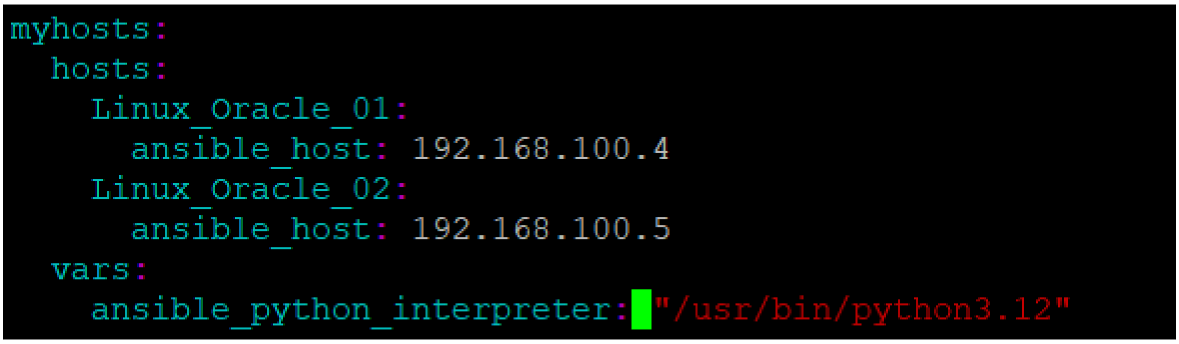
```
[myhosts]
192.168.100.4
192.168.100.5
[myhosts:vars]
ansible_python_interpreter="/usr/bin/python3.12"
```

No more warning:

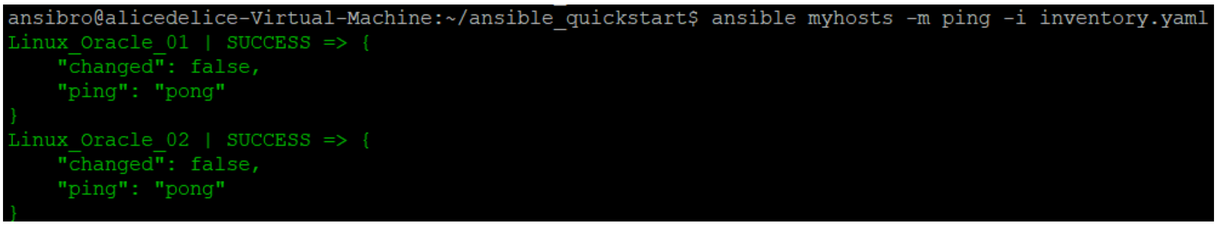


```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ ansible myhosts -m ping -i inventory.ini
192.168.100.4 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
192.168.100.5 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

The same for yaml file



```
myhosts:
  hosts:
    Linux_Oracle_01:
      ansible_host: 192.168.100.4
    Linux_Oracle_02:
      ansible_host: 192.168.100.5
  vars:
    ansible_python_interpreter: "/usr/bin/python3.12"
```



```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ ansible myhosts -m ping -i inventory.yaml
Linux_Oracle_01 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
Linux_Oracle_02 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

(Ansible Community Documentation, 2025)

Example playbook

As stated in the Ansible documentation (Ansible Community Documentaion, 2025)

“Playbooks are automation blueprints, in YAML format, that Ansible uses to deploy and configure managed nodes.

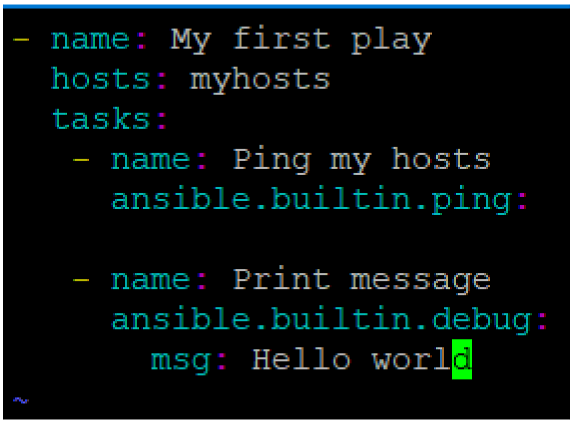
Playbook A list of plays that define the order in which Ansible performs operations, from top to bottom, to achieve an overall goal.

Play An ordered list of tasks that maps to managed nodes in an inventory.

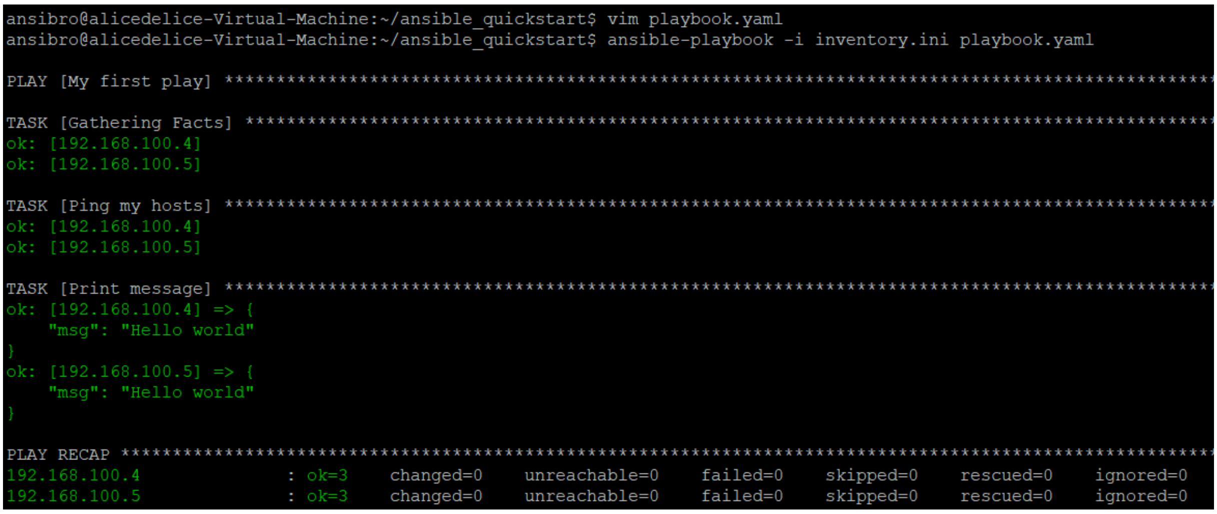
Task A reference to a single module that defines the operations that Ansible performs.

Module A unit of code or binary that Ansible runs on managed nodes. Ansible modules are grouped in collections with a Fully Qualified Collection Name (FQCN) for each module.”

Created an example **“Hello World”** playbook called **playbook.yaml** from the documentation:

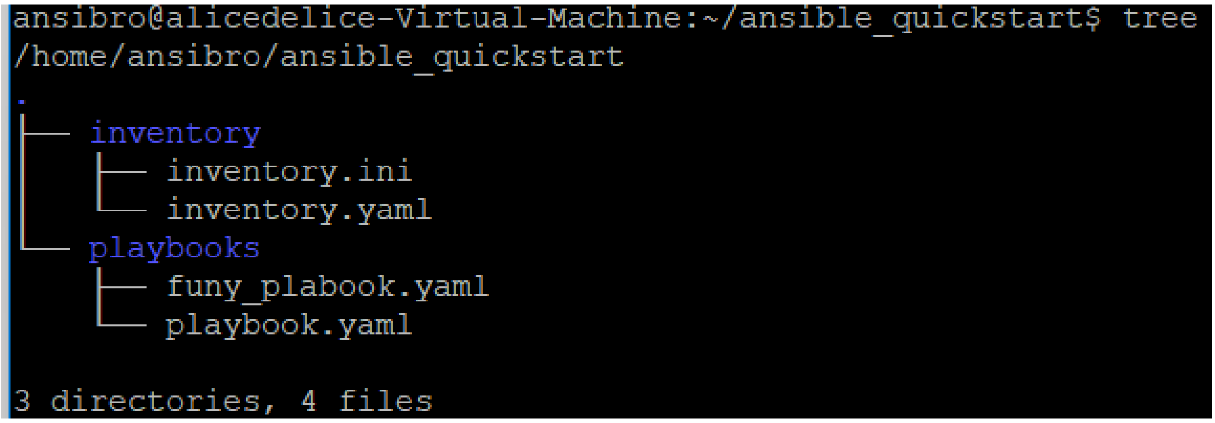


```
- name: My first play
  hosts: myhosts
  tasks:
    - name: Ping my hosts
      ansible.builtin.ping:

    - name: Print message
      ansible.builtin.debug:
        msg: Hello world
~
```



```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ vim playbook.yaml
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ ansible-playbook -i inventory.ini playbook.yaml

PLAY [My first play] ************************************************************************************************

TASK [Gathering Facts] **********************************************************************************************
ok: [192.168.100.4]
ok: [192.168.100.5]

TASK [Ping my hosts] ************************************************************************************************
ok: [192.168.100.4]
ok: [192.168.100.5]

TASK [Print message] ************************************************************************************************
ok: [192.168.100.4] => {
    "msg": "Hello world"
}
ok: [192.168.100.5] => {
    "msg": "Hello world"
}

PLAY RECAP **********************************************************************************************************
192.168.100.4              : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
192.168.100.5              : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Created separate directories for inventory files and for playbooks and moved relevant files there.



```
ansibro@alicedelice-Virtual-Machine:~/ansible_quickstart$ tree
/home/ansibro/ansible_quickstart
.
├── inventory
│   ├── inventory.ini
│   └── inventory.yaml
└── playbooks
    ├── funy_plabook.yaml
    └── playbook.yaml

3 directories, 4 files
```

Now playbooks can be run with providing the full path: **ansible-playbook -i /home/ansibro/ansible_quickstart/inventory/inventory.yaml /home/ansibro/ansible_quickstart/playbooks/playbook.yaml**

Bibliografia

Ansible Community Documentaion. (2025, 10 08). *Creating a playbook.* Retrieved from https://docs.ansible.com/ansible/latest/getting_started/get_started_playbook.html

Ansible Community Documentation. (2025, 10 08). *Building an inventory.* Retrieved from https://docs.ansible.com/ansible/latest/getting_started/get_started_inventory.html#get-started-inventory

Ansible Community Documentation. (2025, 10 08). *Getting started with Ansible.* Retrieved from https://docs.ansible.com/ansible/latest/getting_started/index.html

Ansible Community Documentation. (2025, 10 08). *How to build your inventory.* Retrieved from https://docs.ansible.com/ansible/latest/inventory_guide/intro_inventory.html#defining-variables-in-ini-format

Ansible Community Documentation. (2025, 10 08). *Installing Ansible.* Retrieved from https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

Ansible Community Documentation. (2025, 10 08). *Start automating with Ansible.* Retrieved from https://docs.ansible.com/ansible/latest/getting_started/get_started_ansible.html

Ansible Community Documentation. (2025, 10 08). *Start automating with Ansibleℑ.* Retrieved from https://docs.ansible.com/ansible/latest/getting_started/get_started_ansible.html#get-started-ansible

Ansible Core documentation. (2025, 11 05). *Interpreter Discovery.* Retrieved from https://docs.ansible.com/ansible-core/2.19/reference_appendices/interpreter_discovery.html

IEEE. (2001-2018). *The Open Group Base Specifications Issue 7, 2018 edition.* Retrieved from https://pubs.opengroup.org/onlinepubs/9699919799/

Oracle. (2024, 06). Retrieved from Installing and Managing Python: https://docs.oracle.com/cd/F22978_01/python/OL8-PYTHONG.pdf

Oracle Linux. (2025, 07). *Connecting to Remote Systems With.* Retrieved from https://docs.oracle.com/en/operating-systems/oracle-linux/openssh/OL-OPENSSH.pdf

PIPX. (n.d.). *pipx — Install and Run Python Applications in Isolated Environments.* Retrieved from https://pipx.pypa.io/stable/

SSH Academy. (2025). *What is Passphrase & How to Use It?* Retrieved from https://www.ssh.com/academy/ssh/passphrase

Stack Exchange. (2022, 04 23). *system cannot find alternative python3 versions on ubuntu 20.04.* Retrieved from https://askubuntu.com/questions/1403759/system-cannot-find-alternative-python3-versions-on-ubuntu-20-04