

Ansible

1. [Install Ansible on CentOS 7, Ubuntu and Amazon Linux 2.](#)
2. [Creating my first Ansible Control Node using Terraform.](#)
3. [Creating my first's Ansible Managed Node using Terraform.](#)
4. [My first Ansible project \(inventory file\).](#)
5. [My second Ansible project.](#)
6. [My third Ansible project \(ansible.cfg\).](#)
7. [My fourth Ansible project.](#)
8. [My fifth Ansible project \(vars, children\).](#)
9. [My first Ansible playbook.](#)
10. [My second Ansible playbook.](#)
11. [My third Ansible playbook.](#)
12. [My fourth Ansible playbook.](#)
13. [My fifth Ansible playbook.](#)
14. [My sixth Ansible playbook. Loops.](#)
15. [My seventh Ansible playbook. Loops2.](#)
16. [My eight Ansible playbook. Templates and Jinja.](#)
17. [Ansible Roles.](#)
18. [Ansible. Troubleshooting.](#)
19. [Ansible. Vault.](#)

1. [Install Ansible on CentOS 7, Ubuntu and Amazon Linux 2.](#)

- **Node requirement summary**

The table below lists the current and historical versions of **Python** required on control and managed nodes.

ansible-core Version	Control node Python	Managed node Python
2.11	Python 2.7, Python 3.5 - 3.9 [†]	Python 2.6 - 2.7, Python 3.5 - 3.9
2.12	Python 3.8 - 3.10	Python 2.6 - 2.7, Python 3.5 - 3.10
2.13	Python 3.8 - 3.10	Python 2.7, Python 3.5 - 3.10
2.14	Python 3.9 - 3.11	Python 2.7, Python 3.5 - 3.11

- **Ensuring pip is available**

To verify whether **pip** is already installed for your preferred Python:

```
$ python3 -m pip -V
```

```
vagrant@vm1:~$  
vagrant@vm1:~$ python3 -m pip -V  
pip 22.3.1 from /usr/local/lib/python3.10/site-packages/pip (python 3.10)  
vagrant@vm1:~$
```

If you see an error like *No module named pip*, you'll need to [install pip](#) under your chosen **Python** interpreter before proceeding. This may mean installing an additional OS package (for example, **python3-pip**), or installing the latest **pip** directly from the Python Packaging Authority by running the following:

```
$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py  
$ python3 get-pip.py --user
```

- **Installing Ansible**

Use **pip** in your selected Python environment to install the Ansible package of your choice for the current user:

```
$ python3 -m pip install --user ansible
```

Alternately, you can install a specific version of ansible-core in this Python environment:

```
$ python3 -m pip install --user ansible-core==2.13.5
```

- **Upgrading Ansible**

To upgrade an existing Ansible installation in this Python environment to the latest released version, simply add **--upgrade** to the command above:

```
$ python3 -m pip install --upgrade --user ansible
```

- **Confirming your installation**

You can test that Ansible is installed correctly by checking the version:

```
$ ansible --version
```

To check the version of the ansible package that has been installed:

```
$ python3 -m pip show ansible
```

- **Installing Ansible on specific operating systems**

- a. Install Installing Ansible on CentOS 7

```
$ sudo yum install epel-release
$ sudo yum install ansible
```

RPMs for currently supported versions of CentOS are also available from [EPEL](#)

- b. Install Installing Ansible on Ubuntu

```
$ sudo apt update
$ sudo apt install software-properties-common
$ sudo add-apt-repository --yes --update ppa:ansible/ansible
$ sudo apt install ansible
```

- **Amazon Linux**

- a. Install Ansible on Amazon Linux2 using the Amazon Extras Library

```
$ sudo yum update -y
$ sudo amazon-linux-extras install ansible2 -y
$ ansible --version
```

- b. Install Ansible on Amazon Linux2 using EPEL

```
$ sudo amazon-linux-extras install epel -y
$ sudo yum repolist
$ sudo yum-config-manager --enable epel
$ sudo amazon-linux-extras disable ansible2
$ sudo yum --enablerepo epel install ansible
$ ansible --version
```

2. Creating my first Ansible Control Node using Terraform.

- **Terraform file**

```
$ user_data.sh      ansible_control_node.tf X
ansible_control_node.tf > provider "aws"
1  terraform {
2
3    required_providers {
4      aws = {
5        source = "hashicorp/aws"
6        version = "4.44.0"
7      }
8    }
9
10 }
11
12 provider "aws" {
13
14   region = "eu-central-1"
15
16 }
17
18
19 data "aws_ami" "ubuntu_22_04" {
20
21   most_recent = true
22   owners = ["099720109477"]
23
24   filter {
25     name = "name"
26     values = ["ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-*"]
27   }
28
29 }
30
31
32 resource "aws_instance" "AnsibleController" {
33
34   ami = data.aws_ami.ubuntu_22_04.id
35   instance_type = "t2.micro"
36   vpc_security_group_ids = [aws_security_group.AnansibleController.id]
37   key_name = data.aws_key_pair.current.key_name
38   tags = {
39     Name = "AnsibleController"
40     Project = "AnsibleControlNode"
41     Owner = "yakymov1yevhen@gmail.com"
42   }
43   depends_on = [
44     aws_key_pair.ansible
45   ]
46   user_data = file("user_data.sh")
47
48 }
49
50 resource "aws_eip" "my_static_ip" {
51   instance = aws_instance.AnansibleController.id
52 }
```

```
$ user_data.sh      ansible_control_node.tf X
ansible_control_node.tf > ...

91
92 resource "aws_key_pair" "ansible" {
93
94     key_name = "ansible_key"
95     public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCTvBHezQsUflrdfOHSc2el5R07lxWnyCY/xAc5
96
97 }
98
99 data "aws_key_pair" "current" {
100
101 depends_on = [
102     aws_key_pair.ansible
103 ]
104
105 key_name = "ansible_key"
106
107 }
108
109 data "aws_instance" "current" {
110
111 depends_on = [
112     aws_instance.AnibleController,
113     aws_eip.my_static_ip
114 ]
115
116 filter {
117     name = "tag:Name"
118     values = ["AnsibleController"]
119 }
120
121 }
122
123 output "name" {
124     value = data.aws_key_pair.current.key_name
125 }
126
127 output "data_aws_instance_current" {
128
129     value = data.aws_instance.current.public_ip
130
131 }
132
133 output "data_aws_instance_public_dns" {
134
135     value = data.aws_instance.current.public_dns
136
137 }
```

- **user_data.sh file**

```
$ user_data.sh X ansible_control_node.tf
$ user_data.sh
1  #!/bin/bash
2
3  #---User data for terraform file including Ansible instalation for Ubuntu 22.04---#
4
5  # Installing python3 via update and upgrade command
6 sudo apt update
7 sudo apt -y upgrade
8
9  # checking python version
10 python3 --version > /home/ubuntu/python3_version.txt
11
12 # Installing pip3
13 sudo apt install -y python3-pip
14
15 #checking pip version
16 python3 -m pip -V > /home/ubuntu/pip3_version.txt
17
18
19 # Installing Ansible on Ubuntu
20
21 # 1. Use pip in your selected Python environment to install the Ansible package
22 # of your choice for the current user:
23
24 sudo -u ubuntu python3 -m pip install --user ansible-core==2.14.1
25
26
27 # Add a folder /home/ubuntu/.local/bin to $PATH variable (system wide):
28 sudo cat << EOF >> /etc/bash.bashrc
29 if [ -d "/home/ubuntu/.local/bin" ] ; then
30   PATH="/home/ubuntu/.local/bin:$(/echo '$PATH')"
31 fi
32 EOF
33
34 # Add a folder /home/ubuntu/.local/bin to $PATH variable (for current user):
35 sudo cat << EOF >> /home/ubuntu/.bashrc
36 if [ -d "/home/ubuntu/.local/bin" ] ; then
37   PATH="/home/ubuntu/.local/bin:$(/echo '$PATH')"
38 fi
39 EOF
40
41 # test that Ansible is installed correctly by checking the version:
42 sudo exec bash
43 exec bash
44 sudo -u ubuntu ansible --version > /home/ubuntu/ansible_version.txt
45
46
47 #-----#
48 # 2. Alternative way to install Ansible:
49 # To configure the PPA on your system and install Ansible run these commands:
50 # sudo apt update
51 # sudo apt install -y software-properties-common
52 # sudo add-apt-repository --yes --update ppa:ansible/ansible
53 # sudo apt install -y ansible
54 #-----#
```

Directory: C:\Users\Yevhen Yakymov\Desktop\Programming\DevOps\Terraform\infrastructure\AnsibleControlNode

Mode	LastWriteTime	Length	Name
-a---	06.01.2023 17:56	2874	ansible_control_node.tf
-a---	06.01.2023 20:21	1756	user_data.sh

- terraform plan, deploy and checking ansible version on AnsibleControlNode

```
PS C:\Users\Yevhen Yakymov\Desktop\Programming\DevOps\Terraform\infrastructure\AnsibleControlNo
de> terraform plan ↗
data.aws_ami.ubuntu_22_04: Reading...
data.aws_ami.ubuntu_22_04: Read complete after 0s [id=ami-095a2ffdd817c7a67]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create
<= read (data resources)

Terraform will perform the following actions:

# data.aws_instance.current will be read during apply
# (depends on a resource or a module with changes pending)
<= data "aws_instance" "current" {
    + ami                  = (known after apply)
    + arn                  = (known after apply)
```

Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```
+ data.aws_instance_current      = (known after apply)
+ data.aws_instance_public_dns = (known after apply)
+ name                         = "ansible_key"
```

```
de> terraform apply ↗
```

```
data.aws_ami.ubuntu_22_04: Reading...
data.aws_ami.ubuntu_22_04: Read complete after 0s [id=ami-095a2ffdd817c7a67]
```

```
Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create
<= read (data resources)
```

Terraform will perform the following actions:

```
# data.aws_instance.current will be read during apply
# (depends on a resource or a module with changes pending)
<= data "aws_instance" "current" {
    + ami                  = (known after apply)
    + arn                  = (known after apply)
```

Enter a value: yes

```
aws_key_pair.ansible: Creating...
aws_security_group.AnibleController: Creating...
aws_key_pair.ansible: Creation complete after 1s [id=ansible_key]
data.aws_key_pair.current: Reading...
data.aws_key_pair.current: Read complete after 0s [id=key-0e52c55459832c387]
aws_security_group.AnibleController: Creation complete after 3s [id=sg-00ba31e3a7ee21fb2]
aws_instance.AnibleController: Creating...
aws_instance.AnibleController: Still creating... [10s elapsed]
aws_instance.AnibleController: Still creating... [20s elapsed]
aws_instance.AnibleController: Still creating... [30s elapsed]
aws_instance.AnibleController: Still creating... [40s elapsed]
aws_instance.AnibleController: Creation complete after 43s [id=i-0d5f76edd180bfff8]
aws_eip.my_static_ip: Creating...
aws_eip.my_static_ip: Creation complete after 1s [id=eipalloc-0e49fb284e69bd25c]
data.aws_instance.current: Reading...
data.aws_instance.current: Read complete after 2s [id=i-0d5f76edd180bfff8]
```

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

```
data.aws_instance_current = "18.159.73.74"
data.aws_instance_public_dns = "ec2-18-159-73-74.eu-central-1.compute.amazonaws.com"
name = "ansible_key"
```

Instances (1/3) Info

Name	Instance ID	Instance state	Instance type
JenkinsServer	i-0655a9022e4f52489	Running	t2.micro
MyWebServer	i-0e49692a5bf1e282b	Stopped	t2.micro
AnsibleController	i-0d5f76edd180bfff8	Running	t2.micro

Instance: i-0d5f76edd180bfff8 (AnsibleController)

Details | Security | Networking | Storage | Status checks | Monitoring >

Instance summary | **Info**

Instance ID: i-0d5f76edd180bfff8 (AnsibleController)

Public IPv4 address: 18.159.73.74 | [open address](#)

ssh to our AnsibleController with MobaXterm using public IP and ansible_key

SSH Telnet Rsh Xdmcp RDP VNC FTP SFTP Serial File Shell Browser Mosh Aws S3 WSL

Basic SSH settings

Remote host * 18.159.73.74 | Specify username ubuntu | Port 22

Advanced SSH settings

X11-Forwarding | Compression | Execute command: | Do not exit after command ends
 SSH-browser type: SFTP protocol | Follow SSH path (experimental)
 Use private key C:\Users\YEVHEN~1\Desktop\Prd | Adapt locales on remote server

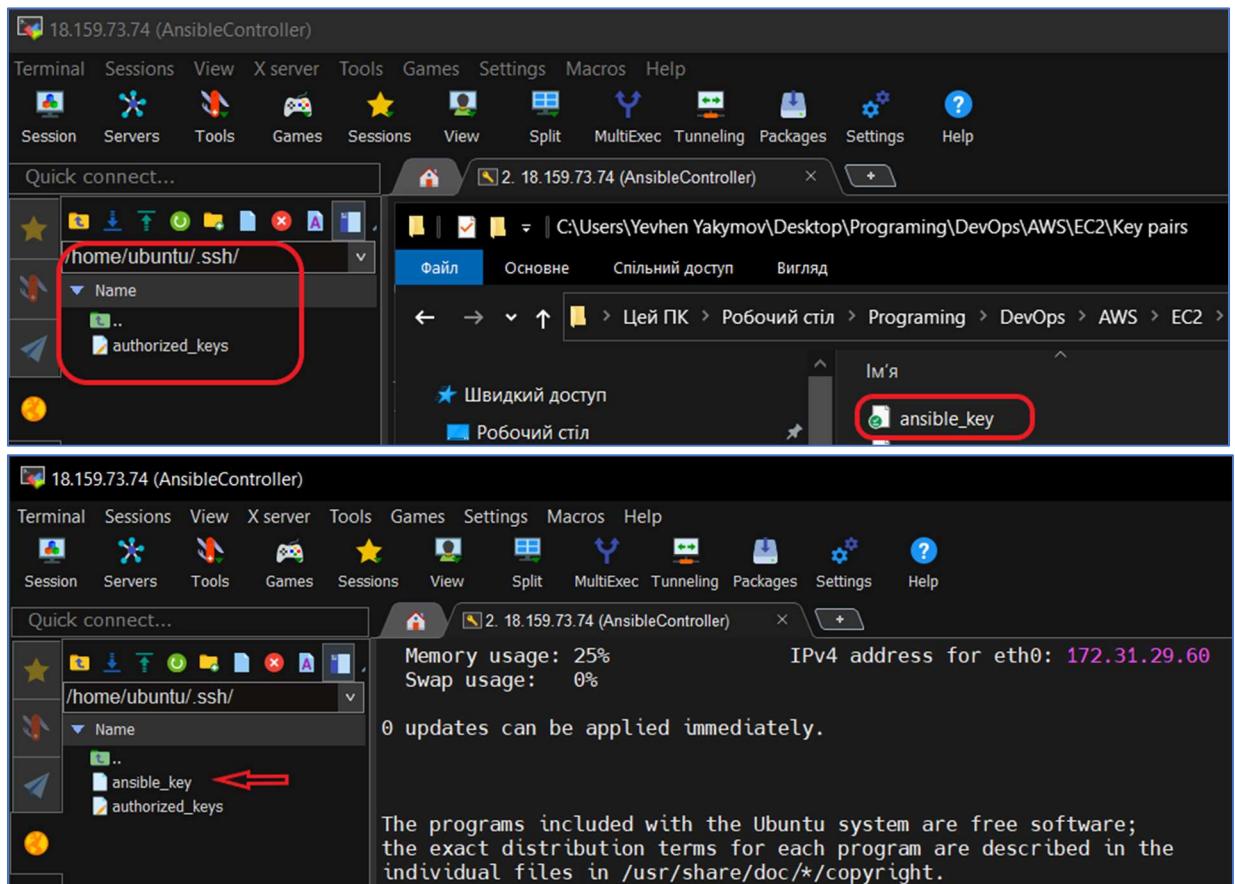
Checking Ansible version

```
ubuntu@ip-172-31-29-60:~$ ansible --version
ansible [core 2.14.1]
  config file = None
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /home/ubuntu/.local/lib/python3.10/site-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /home/ubuntu/.local/bin/ansible
  python version = 3.10.6 (main, Nov 14 2022, 16:10:14) [GCC 11.3.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
ubuntu@ip-172-31-29-60:~$ ll
total 48
drwxr-x--- 6 ubuntu ubuntu 4096 Jan  7 17:37 .
drwxr-xr-x  3 root   root  4096 Jan  7 17:26 ..
-rw-----  1 ubuntu ubuntu  61 Jan  7 17:37 .Xauthority
drwxrwxr-x  3 ubuntu ubuntu 4096 Jan  7 17:37 .ansible/
-rw-r--r--  1 ubuntu ubuntu 220 Jan  6 2022 .bash_logout
-rw-r--r--  1 ubuntu ubuntu 3856 Jan  7 17:27 .bashrc
drwxrwxr-x  3 ubuntu ubuntu 4096 Jan  7 17:37 .cache/
drwxrwxr-x  4 ubuntu ubuntu 4096 Jan  7 17:27 .local/
-rw-r--r--  1 ubuntu ubuntu  807 Jan  6 2022 .profile
drwx----- 2 ubuntu ubuntu 4096 Jan  7 17:26 .ssh/
-rw-r--r--  1 root   root   65 Jan  7 17:27 pip3_version.txt
-rw-r--r--  1 root   root   14 Jan  7 17:26 python3_version.txt
```

Checking Python3 and pip version

```
ubuntu@ip-172-31-29-60:~$ ll
total 48
drwxr-x--- 6 ubuntu ubuntu 4096 Jan  7 17:37 ./
drwxr-xr-x  3 root  root  4096 Jan  7 17:26 ../
-rw-----  1 ubuntu ubuntu   61 Jan  7 17:37 .Xauthority
drwxrwxr-x  3 ubuntu ubuntu 4096 Jan  7 17:37 .ansible/
-rw-r--r--  1 ubuntu ubuntu  220 Jan  6 2022 .bash_logout
-rw-r--r--  1 ubuntu ubuntu 3856 Jan  7 17:27 .bashrc
drwxrwxr-x  3 ubuntu ubuntu 4096 Jan  7 17:37 .cache/
drwxrwxr-x  4 ubuntu ubuntu 4096 Jan  7 17:27 .local/
-rw-r--r--  1 ubuntu ubuntu  807 Jan  6 2022 .profile
drwx----- 2 ubuntu ubuntu 4096 Jan  7 17:57 .ssh/
-rw-r--r--  1 root  root   65 Jan  7 17:27 pip3_version.txt
-rw-r--r--  1 root  root   14 Jan  7 17:26 python3_version.txt
ubuntu@ip-172-31-29-60:~$
ubuntu@ip-172-31-29-60:~$ cat python3_version.txt
Python 3.10.6 ←
ubuntu@ip-172-31-29-60:~$ cat pip3_version.txt
pip 22.0.2 from /usr/lib/python3/dist-packages/pip (python 3.10) ←
ubuntu@ip-172-31-29-60:~$
```

- copy ansible_key to the AnsibleController



3. Creating my first's Ansible Managed Node using Terraform.

- **Terraform file**

```
ansible_managed_node.tf $ user_data.sh

1  terraform {
2      required_providers {
3          aws = {
4              source = "hashicorp/aws"
5              version = "4.44.0"
6          }
7      }
8  }
9
10 provider "aws" {
11     access_key = "AK[REDACTED]"
12     secret_key = "vH[REDACTED]"
13     region = "eu-central-1"
14 }
15
16
17 data "aws_ami" "ubuntu_22_04" [
18
19     most_recent = true
20     owners = ["099720109477"]
21
22     filter {
23         name = "name"
24         values = ["ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-*"]
25     }
26
27 ]
28
29
30 resource "aws_instance" "AnsibleManagedNode" {
31
32     count = var.instance_count
33     ami = data.aws_ami.ubuntu_22_04.id
34     instance_type = "t2.micro"
35     vpc_security_group_ids = [aws_security_group.AnibleManagedNode.id]
36     key_name = data.aws_key_pair.current.key_name
37     tags = {
38         Name = "AnsibleManagedNode-${count.index + 1}"
39         Project = "AnsibleManagedNode"
40         Owner = "yakymov1yevhen@gmail.com"
41     }
42     depends_on = [
43         aws_key_pair.ansible
44     ]
45     user_data = file("user_data.sh")
46
47 }
48
49 variable "instance_count" {
50     default = "3"
51 }
```

```
ansible_managed_node.tf X $ user_data.sh
ansible_managed_node.tf > data "aws_ami" "ubuntu_22_04" > most_recent
3 /   /
58
59 resource "aws_security_group" "AnsibleManagedNode" {
60
61     name          = "Ansible Managed Security Group"
62     description  = "Allow SSH inbound traffic"
63
64     ingress {
65         description      = "SSH from VPC"
66         from_port        = 22
67         to_port          = 22
68         protocol         = "tcp"
69         cidr_blocks     = ["0.0.0.0/0"]
70     }
71
72     egress {
73         from_port        = 0
74         to_port          = 0
75         protocol         = "-1"
76         cidr_blocks     = ["0.0.0.0/0"]
77     }
78
79     tags = {
80         Name = "allow_ssh"
81     }
82
83 }
84
85
86 resource "aws_key_pair" "ansible" {
87
88     key_name = "ansible_managed_key"
89     public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCTvBHezQsUflrdfoHSc2e15R07lxWnyCY/x/
90
91 }
92
93
94 data "aws_key_pair" "current" {
95
96     depends_on = [
97         aws_key_pair.ansible
98     ]
99
100    key_name  = "ansible_managed_key"
101
102 }
103
```

- **user_data.sh file**

```
$ user_data.sh X ansible_control_node.tf
$ user_data.sh
1  #!/bin/bash
2
3  #---User data for terraform file including Ansible instalation for Ubuntu 22.04---#
4
5  # Installding python3 via update and upgrade command
6 sudo apt update
7 sudo apt -y upgrade
8
9  # checking python version
10 python3 --version > /home/ubuntu/python3_version.txt
11
12  # Installding pip3
13 sudo apt install -y python3-pip
14
15  #checking pip version
16 python3 -m pip -V > /home/ubuntu/pip3_version.txt
17
18
19  # Installding Ansible on Ubuntu
20
21  # 1. Use pip in your selected Python environment to install the Ansible package
22  # of your choice for the current user:
23
24 sudo -u ubuntu python3 -m pip install --user ansible-core==2.14.1
25
26
27  # Add a folder /home/ubuntu/.local/bin to $PATH variable (system wide):
28 sudo cat << EOF >> /etc/bash.bashrc
29 if [ -d "/home/ubuntu/.local/bin" ] ; then
30   PATH="/home/ubuntu/.local/bin:$echo '$PATH'"
31 fi
32 EOF
33
34  # Add a folder /home/ubuntu/.local/bin to $PATH variable (for current user):
35 sudo cat << EOF >> /home/ubuntu/.bashrc
36 if [ -d "/home/ubuntu/.local/bin" ] ; then
37   PATH="/home/ubuntu/.local/bin:$echo '$PATH'"
38 fi
39 EOF
40
41  # test that Ansible is installed correctly by checking the version:
42 sudo exec bash
43 exec bash
44 sudo -u ubuntu ansible --version > /home/ubuntu/ansible_version.txt
45
46
47  #-----#
48  # 2. Alternative way to install Ansible:
49  # To configure the PPA on your system and install Ansible run these commands:
50  # sudo apt update
51  # sudo apt install -y software-properties-common
52  # sudo add-apt-repository --yes --update ppa:ansible/ansible
53  # sudo apt install -y ansible
54  #-----#

```

```
-g---- 01.07.2023 17:38 2023-07-01 17:38
-g---- 01.07.2023 17:38 2023-07-01 17:38
-g---- 01.07.2023 17:38 2023-07-01 17:38
-g---- 08.07.2023 17:38 2023-07-08 17:38
-g---- 01.07.2023 17:38 2023-07-01 17:38
q-L-- 01.07.2023 17:38 2023-07-01 17:38
-----
```

```
W0qe          amfTfTfTwfSf         fneMfHn
-----
```

6

Directorl: C:/Users/syavlevn/Downloads/DevOps/Development/Terraform/Ansible/bo

- terraform plan, deploy and checking ansible version on AnsibleManagedNode

```
PS C:\Users\Yevhen Yakymov\Desktop\Programming\DevOps\Terraform\infrastructure\AnsibleControlNo
de> terraform plan ↗
data.aws_ami.ubuntu_22_04: Reading...
data.aws_ami.ubuntu_22_04: Read complete after 0s [id=ami-095a2ffdd817c7a67]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create
<= read (data resources)

Terraform will perform the following actions:

# data.aws_instance.current will be read during apply
# (depends on a resource or a module with changes pending)
<= data "aws_instance" "current" {
    + ami                  = (known after apply)
    + arn                  = (known after apply)
```

Plan: 5 to add, 0 to change, 0 to destroy.

```
de> terraform apply ↗
data.aws_ami.ubuntu_22_04: Reading...
data.aws_ami.ubuntu_22_04: Read complete after 0s [id=ami-095a2ffdd817c7a67]

Terraform used the selected providers to generate the following execution plan. Resource
actions are indicated with the following symbols:
+ create
<= read (data resources)

Terraform will perform the following actions:

# data.aws_instance.current will be read during apply
# (depends on a resource or a module with changes pending)
<= data "aws_instance" "current" {
    + ami                  = (known after apply)
    + arn                  = (known after apply)
```

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```
aws_key_pair.ansible: Creating...
aws_security_group.AnibleManagedNode: Creating...
aws_key_pair.ansible: Creation complete after 1s [id=ansible_managed_key]
data.aws_key_pair.current: Reading...
data.aws_key_pair.current: Read complete after 0s [id=key-063777b4d57d7f906]
aws_security_group.AnibleManagedNode: Creation complete after 3s [id=sg-09a5dd502e329ea6b]
aws_instance.AnibleManagedNode[1]: Creating...
aws_instance.AnibleManagedNode[0]: Creating...
aws_instance.AnibleManagedNode[2]: Creating...
aws_instance.AnibleManagedNode[1]: Still creating... [10s elapsed]
aws_instance.AnibleManagedNode[2]: Still creating... [10s elapsed]
aws_instance.AnibleManagedNode[0]: Still creating... [10s elapsed]
aws_instance.AnibleManagedNode[1]: Still creating... [20s elapsed]
aws_instance.AnibleManagedNode[0]: Still creating... [20s elapsed]
aws_instance.AnibleManagedNode[2]: Still creating... [20s elapsed]
aws_instance.AnibleManagedNode[1]: Still creating... [30s elapsed]
aws_instance.AnibleManagedNode[2]: Still creating... [30s elapsed]
aws_instance.AnibleManagedNode[0]: Still creating... [30s elapsed]
aws_instance.AnibleManagedNode[0]: Creation complete after 33s [id=i-00438d7e753170b0b]
aws_instance.AnibleManagedNode[2]: Creation complete after 33s [id=i-077ef056384a869be]
aws_instance.AnibleManagedNode[1]: Creation complete after 33s [id=i-0c04df37d47fa0701]
```

Apply complete! Resources: 5 added, 0 changed, 0 destroyed. ↗

Instances (1/5) Info

Name	Instance ID	Instance state	Instance type
AnsibleManagedNode-1	i-00438d7e753170b0b	Running	t2.micro
AnsibleManagedNode-2	i-0c04df37d47fa0701	Running	t2.micro
AnsibleManagedNode-3	i-077ef056384a869be	Running	t2.micro
JenkinsServer	i-0655a9022e4f52489	Stopped	t2.micro
MyWebServer	i-0e49692a5bf1e282b	Stopped	t2.micro

Instance: i-00438d7e753170b0b (AnsibleManagedNode-1)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary

Instance ID: i-00438d7e753170b0b (AnsibleManagedNode-1)

Public IPv4 address: 3.71.18.221 | [open address](#)

Private IPv4 addresses: 172.31.25.100

ssh to our AnsibleManagedNode-1 with MobaXterm using public IP and ansible_key

SSH Telnet Rsh Xdmcp RDP VNC FTP SFTP Serial File Shell Browser Mosh Aws S3 WSL

Basic SSH settings

Remote host * 3.71.18.221 Specify username ubuntu Port 22

Advanced SSH settings

X11-Forwarding Compression Remote environment: Interactive shell Execute command: Do not exit after command ends SSH-browser type: SFTP protocol Follow SSH path (experimental) Use private key C:\Users\YEVHEN~1\Desktop\Pro Adapt locales on remote server

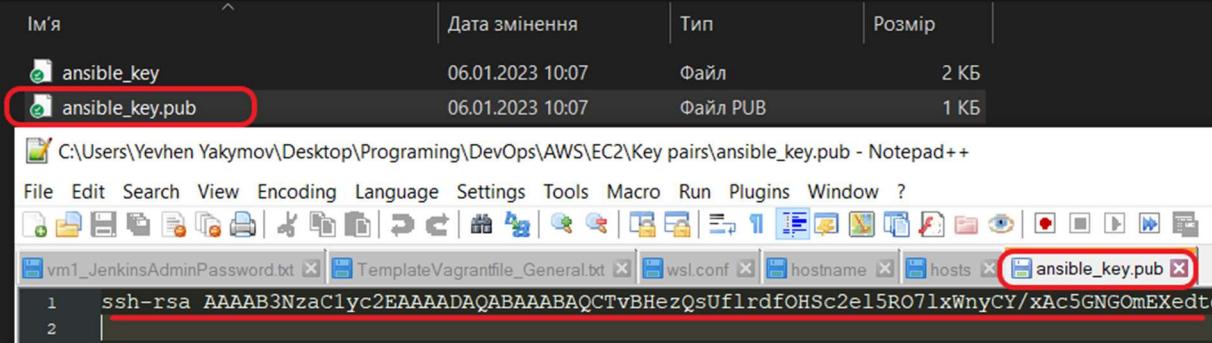
Checking Ansible version

```
ubuntu@ip-172-31-25-100:~$ ansible --version
ansible [core 2.14.1]
  config file = None
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /home/ubuntu/.local/lib/python3.10/site-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /home/ubuntu/.local/bin/ansible
  python version = 3.10.6 (main, Nov 14 2022, 16:10:14) [GCC 11.3.0] (/usr/bin/python3)
  jinja version = 3.0.3
  libyaml = True
ubuntu@ip-172-31-25-100:~$ ll
total 48
drwxr-x--- 6 ubuntu ubuntu 4096 Jan  8 05:41 .
drwxr-xr-x  3 root   root  4096 Jan  8 05:35 ..
-rw-----  1 ubuntu ubuntu  62 Jan  8 05:41 .Xauthority
drwxrwxr-x  3 ubuntu ubuntu 4096 Jan  8 05:41 .ansible/
-rw-r--r--  1 ubuntu ubuntu 220 Jan  6 2022 .bash_logout
-rw-r--r--  1 ubuntu ubuntu 3856 Jan  8 05:36 .bashrc
drwxrwxr-x  3 ubuntu ubuntu 4096 Jan  8 05:41 .cache/
drwxrwxr-x  4 ubuntu ubuntu 4096 Jan  8 05:36 .local/
-rw-r--r--  1 ubuntu ubuntu  807 Jan  6 2022 .profile
drwx----- 2 ubuntu ubuntu 4096 Jan  8 05:35 .ssh/
```

Checking Python3 and pip version

```
drwx----- 2 ubuntu ubuntu 4096 Jan  8 05:35 .ssh/
-rw-r--r-- 1 root   root     65 Jan  8 05:36 pip3_version.txt
-rw-r--r-- 1 root   root    14 Jan  8 05:36 python3_version.txt
ubuntu@ip-172-31-25-100:~$ 
ubuntu@ip-172-31-25-100:~$ cat python3_version.txt
Python 3.10.6 ←
ubuntu@ip-172-31-25-100:~$ cat pip3_version.txt
pip 22.0.2 from /usr/lib/python3/dist-packages/pip (python 3.10) ←
```

- copy ansible_key.pub to the AnsibleManagedNode-1(2,3) ~/.ssh/authorized_keys to have ability connect from AnsibleController to the AnsibleManagedNode-1(2,3)



C:\Users\Yevhen Yakymov\Desktop\Programming\DevOps\AWS\EC2\Key pairs\ansible_key.pub - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

hosts ansible_key.pub

```
1 ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCTvBHezQsUflrdf0HSc2e15R07lxWnyCY/xAc5GNG0mExedtgcx4SRPCy]
FIOw0Hz0TLGZ3kwhDFWHFL50qz7PGTyJpqMdMPvHkDrsiIw+uEp13HNiLYWAMqQIzsriWodM/bD8imcvDlq+TnbBKv+J0i2r
I9VgHh2ppJs0BiIQUhMwY5rHwm03rD7/M5qqdoErpR6oF42n5CuVRzhfSPd60TuLQxNxpK24fuZbs/gkeotqrpUqpIAU+D6c
DmXjgZBxv0Aj+e+Yy7+mHzY6fu2VGu08s+3aYGJxgThnav+3sp ansible_managed_key
```

- checking ssh connection from AnsibleController to the AnsibleManagedNode-1

```
ubuntu@ip-172-31-22-86:~/ssh$ ssh -i "ansible_key" ubuntu@ec2-3-71-18-221.eu-central-1.compute.amazonaws.com
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-1026-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:      https://landscape.canonical.com
 * Support:         https://ubuntu.com/advantage

 System information as of Sun Jan  8 06:03:39 UTC 2023

 System load:  0.0          Processes:           101
 Usage of /:   25.4% of 7.57GB   Users logged in:      1
 Memory usage: 28%
 Swap usage:   0%          IPv4 address for eth0: 172.31.25.100

ubuntu@ip-172-31-25-100:/etc$ curl http://169.254.169.254/latest/meta-data/public-ipv4
3.71.18.221ubuntu@ip-172-31-25-100:/etc$
```

4. My first Ansible project (inventory file).

- Creation directory and file **hosts.txt**(inventory file) with description our managed node

The screenshot shows a terminal window with four tabs at the top: 3. 18.158.196.253 (Ansible), 6. 3.71.18.221 (Ansible), 4. 3.121.162.144 (Ansible), and 5. 3.120.178.95 (Ansible). The main pane displays the following command sequence:

```
ubuntu@ip-172-31-22-86:~$ mkdir -p ansible/project-1
ubuntu@ip-172-31-22-86:~$ cd ansible/project-1/
ubuntu@ip-172-31-22-86:~/ansible/project-1$ nano hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-1$ cat hosts.txt
[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221 ansible_user=ubuntu ansible_ssh_private_key_file=/home/ubuntu/.ssh/ansible_key
ubuntu@ip-172-31-22-86:~/ansible/project-1$
```

Below this, another terminal window is shown with the title "hosts.txt" and the same inventory file content:

```
[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221 ansible_user=ubuntu ansible_ssh_private_key_file=/home/ubuntu/.ssh/ansible_key
```

- Run first **ansible** command

```
$ ansible -i hosts.txt all -m ping
```

Where :

- **i** – option specify inventory host path or comma separated host list

hosts.txt – inventory file

all - apply for all servers described in the inventory file

-m – name of the action to execute

ping – module name

The screenshot shows a terminal window with the following command and its output:

```
ubuntu@ip-172-31-22-86:~/ansible/project-1$ ansible -i hosts.txt all -m ping
The authenticity of host '3.71.18.221 (3.71.18.221)' can't be established.
ED25519 key fingerprint is SHA256:32KMCW85GnxCoyVnu5Lixrw1nx9Ei5jr3SmNXeUg05A.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
AnsibleManagedNode-1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
ubuntu@ip-172-31-22-86:~/ansible/project-1$ ansible -i hosts.txt all -m ping
AnsibleManagedNode-1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

A red arrow points to the first "AnsibleManagedNode-1" entry in the output, and a red box highlights the "discovered_interpreter_python" key in the facts of that entry.

5. My second Ansible project.

- Creation project directory, file **hosts.txt(inventory)** with description our managed node's

The screenshot shows a terminal window titled '2. 18.158.196.253 (AnsibleController)'. The user is navigating through their home directory to the 'ansible' folder. They enter the 'project-2' directory and run 'ls' to show files: total 24, followed by a list of files including 'hosts.txt'. The user then runs 'nano hosts.txt' to edit the inventory file. A red box highlights the 'hosts.txt' command in the terminal. Below this, another terminal window titled 'GNU nano 6.2' shows the contents of the 'hosts.txt' file, which defines two groups: [staging_servers] and [production_servers]. The 'hosts.txt' file path is also highlighted with a red box.

```
Last login: Sun Jan  8 08:56:57 2023 from 88.155.20.19
ubuntu@ip-172-31-22-86:~/ ansible/
ubuntu@ip-172-31-22-86:~/ansible$ cd ansible/
ubuntu@ip-172-31-22-86:~/ansible$ ll
total 24
drwxrwxr-x 6 ubuntu ubuntu 4096 Jan  8 09:50 .
drwxr-x--- 7 ubuntu ubuntu 4096 Jan  8 10:46 ..
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:33 project-1/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:53 project-2/ project-2/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 09:39 project-3/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 09:57 project-4/
ubuntu@ip-172-31-22-86:~/ansible$ cd project-2
ubuntu@ip-172-31-22-86:~/ansible/project-2$ ubuntu@ip-172-31-22-86:~/ansible/project-2$ ll
total 12
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:53 .
drwxrwxr-x 6 ubuntu ubuntu 4096 Jan  8 09:50 ..
-rw-rw-r-- 1 ubuntu ubuntu 302 Jan  8 06:53 hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-2$ nano hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-2$
```



```
[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221 ansible_user=ubuntu ansible_ssh_private_key_file=/home/...
[production_servers]
AnsibleManagedNode-2 ansible_host=3.121.162.144 ansible_user=ubuntu ansible_ssh_private_key_file=/home/...
```

- Run **ansible** command

```
$ ansible -i hosts.txt all -m ping
```

Where :

- **i** – option specify inventory host path or comma separated host list
- hosts.txt** – inventory file
- all** - apply for all servers described in the inventory file
- m** – name of the action(module) to execute
- ping** – module name

The screenshot shows a terminal window titled '2. 18.158.196.253 (AnsibleController)'. The user runs the command '\$ ansible -i hosts.txt all -m ping'. The output shows two hosts: 'AnsibleManagedNode-2' and 'AnsibleManagedNode-1'. Both hosts return a 'SUCCESS' status with facts indicating they have discovered Python 3 as the interpreter. A red arrow points to the output for 'AnsibleManagedNode-1'.

```
ubuntu@ip-172-31-22-86:~/ansible/project-2$ ansible -i hosts.txt all -m ping
AnsibleManagedNode-2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
AnsibleManagedNode-1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ubuntu@ip-172-31-22-86:~/ansible/project-2$ ubuntu@ip-172-31-22-86:~/ansible/project-2$
```

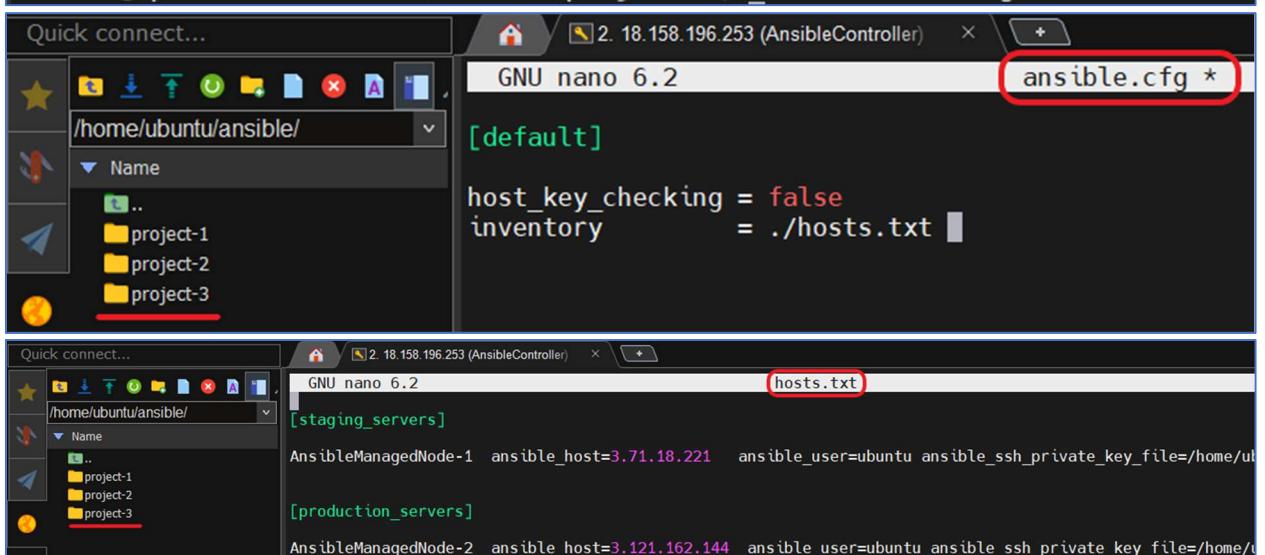
6. My third Ansible project (`ansible.cfg`).

- Creation project directory, file `hosts.txt` with description our managed node's and `ansible.cfg` file with `host_key_checking`, `inventory` parametrs.

Where :

host_key_checking - set this to “False” if you want to avoid host key checking by the underlying tools Ansible uses to connect to the host.
inventory – relative path to the inventory file.

```
ubuntu@ip-172-31-22-86:~/ansible$ cp -r project-2/ project-3/
ubuntu@ip-172-31-22-86:~/ansible$ 
ubuntu@ip-172-31-22-86:~/ansible$ ll
total 20
drwxrwxr-x 5 ubuntu ubuntu 4096 Jan  8 09:19 .
drwxr-x--- 7 ubuntu ubuntu 4096 Jan  8 08:56 ..
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:33 project-1/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:53 project-2/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 09:19 project-3/
ubuntu@ip-172-31-22-86:~/ansible$ 
ubuntu@ip-172-31-22-86:~/ansible$ cd project-3
ubuntu@ip-172-31-22-86:~/ansible/project-3$ 
ubuntu@ip-172-31-22-86:~/ansible/project-3$ nano ansible.cfg
```



The screenshot shows the WinSCP interface with two open nano editors. The top editor is titled "GNU nano 6.2" and contains the `ansible.cfg` file with the following content:

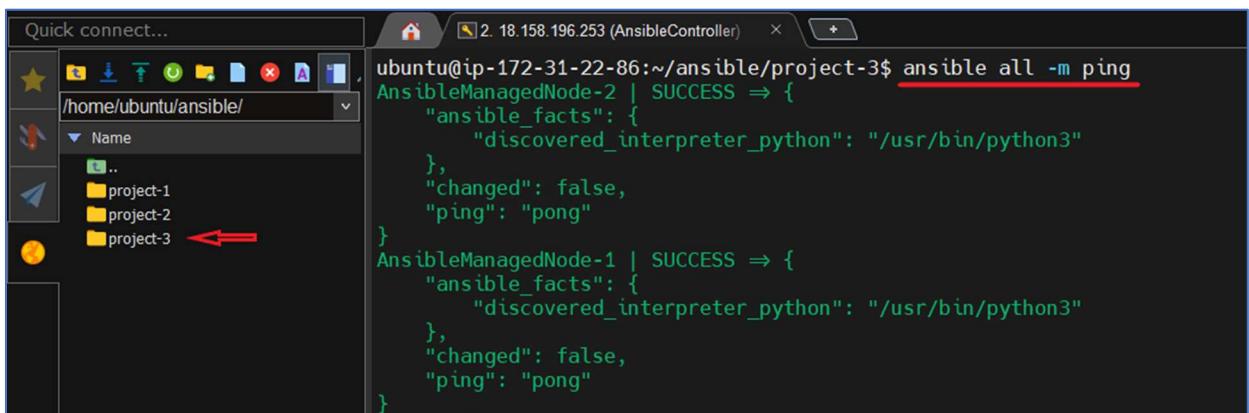
```
[default]
host_key_checking = false
inventory          = ./hosts.txt
```

The bottom editor is also titled "GNU nano 6.2" and contains the `hosts.txt` file with the following content:

```
[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221 ansible_user=ubuntu ansible_ssh_private_key_file=/home/ubuntu/.ssh/staging_node1

[production_servers]
AnsibleManagedNode-2 ansible_host=3.121.162.144 ansible_user=ubuntu ansible_ssh_private_key_file=/home/ubuntu/.ssh/production_node2
```

- Run `ansible` command without `-i` option and specifying `inventory` file



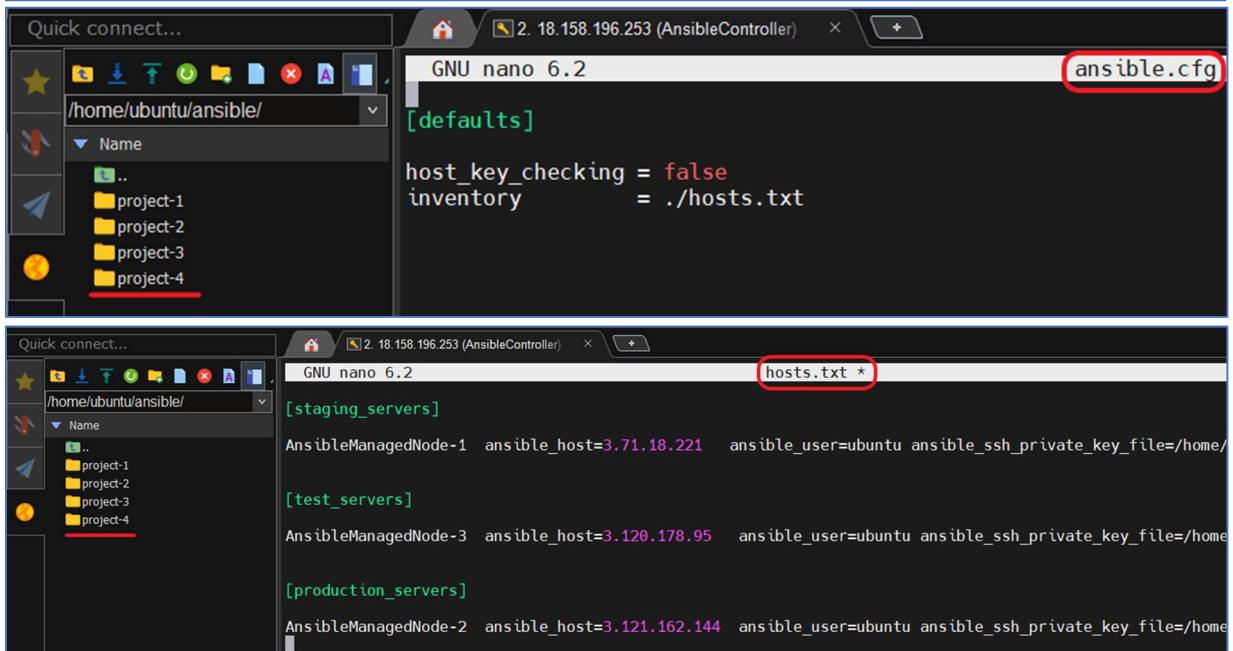
The screenshot shows the WinSCP interface with a terminal window. The command `ansible all -m ping` is being run, and the output shows successful ping results from both managed nodes:

```
ubuntu@ip-172-31-22-86:~/ansible/project-3$ ansible all -m ping
AnsibleManagedNode-2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
AnsibleManagedNode-1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

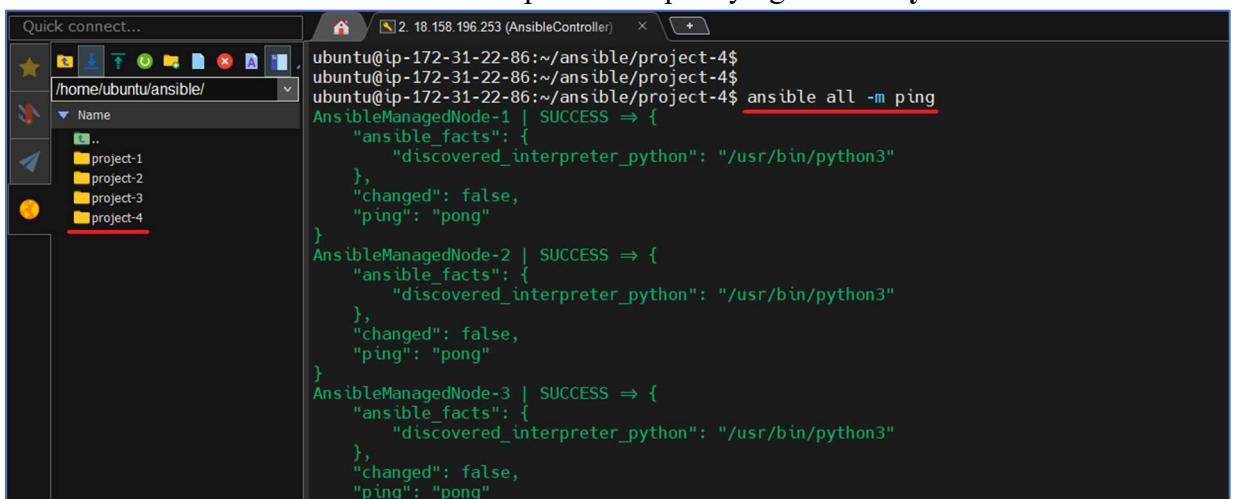
7. My fourth Ansible project.

- Creation project directory, file **hosts.txt** with description our managed node's and **ansible.cfg** file with **host_key_checking**, **inventory** parametrs.

```
ubuntu@ip-172-31-22-86:~/ansible$ ll
total 24
drwxrwxr-x 6 ubuntu ubuntu 4096 Jan  8 09:50 .
drwxr-x--- 7 ubuntu ubuntu 4096 Jan  8 08:56 ..
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:33 project-1/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 06:53 project-2/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 09:39 project-3/
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 09:49 project-4/
ubuntu@ip-172-31-22-86:~/ansible$ cd project-4/
ubuntu@ip-172-31-22-86:~/ansible/project-4$ 
ubuntu@ip-172-31-22-86:~/ansible/project-4$ ll
total 16
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 09:49 .
drwxrwxr-x 6 ubuntu ubuntu 4096 Jan  8 09:50 ..
-rw-rw-r-- 1 ubuntu ubuntu    71 Jan  8 09:49 ansible.cfg
-rw-rw-r-- 1 ubuntu ubuntu   302 Jan  8 09:49 hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-4$ 
ubuntu@ip-172-31-22-86:~/ansible/project-4$ nano hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-4$ 
ubuntu@ip-172-31-22-86:~/ansible/project-4$ nano ansible.cfg
ubuntu@ip-172-31-22-86:~/ansible/project-4$ nano ansible.cfg
```



- Run **ansible** command without **-i** option and specifying **inventory** file



8. My fifth Ansible project (vars, children).

- Creation project directory, file **hosts.txt** with description our managed node's using **vars**, **children** parametrs and **ansible.cfg** file with **host_key_checking**, **inventory** parametrs.

The screenshot shows three terminal windows from a terminal application. The top window shows the command line with the user navigating through directories and copying files. The middle window shows the configuration file `ansible.cfg` being edited in nano, with the `host_key_checking` and `inventory` parameters set. The bottom window shows the `hosts.txt` file being edited in nano, defining groups of servers and their ansible hosts.

```
ubuntu@ip-172-31-22-86:~/ansible/project-2$ cp -r project-4/ project-5/
ubuntu@ip-172-31-22-86:~/ansible/project-5$ cd project-5
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ll
total 16
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 10:54 .
drwxrwxr-x 7 ubuntu ubuntu 4096 Jan  8 10:54 ..
-rw-rw-r-- 1 ubuntu ubuntu 71 Jan  8 10:54 ansible.cfg
-rw-rw-r-- 1 ubuntu ubuntu 449 Jan  8 10:54 hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-5$ nano hosts.txt

[defaults]
host_key_checking = false
inventory      = ./hosts.txt

[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221

[test_servers]
AnsibleManagedNode-3 ansible_host=3.120.178.95

[production_servers]
AnsibleManagedNode-2 ansible_host=3.121.162.144

[all_servers:children] <-->
staging_servers
test_servers
production_servers

[all_servers:vars] <-->
ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/.ssh/ansible_key
```

- Run **ansible** command without **-i** option and specifying **inventory** file

The screenshot shows a terminal window with the command `ansible all -m ping` being run. The output shows successful ping results for three managed nodes: `AnsibleManagedNode-3`, `AnsibleManagedNode-1`, and `AnsibleManagedNode-2`.

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible all -m ping
AnsibleManagedNode-3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
AnsibleManagedNode-1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
AnsibleManagedNode-2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
```

- Run **ansible** commands that show **list** and **graph** based on the **inventory file**

```
$ ansible-inventory --list
$ ansible-inventory --graph
```

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible-inventory --list
{
    "_meta": {
        "hostvars": {
            "AnsibleManagedNode-1": {
                "ansible_host": "3.71.18.221",
                "ansible_ssh_private_key_file": "/home/ubuntu/.ssh/ansible_key",
                "ansible_user": "ubuntu"
            },
            "AnsibleManagedNode-2": {
                "ansible_host": "3.121.162.144",
                "ansible_ssh_private_key_file": "/home/ubuntu/.ssh/ansible_key",
                "ansible_user": "ubuntu"
            },
            "AnsibleManagedNode-3": {
                "ansible_host": "3.120.178.95",
                "ansible_ssh_private_key_file": "/home/ubuntu/.ssh/ansible_key",
                "ansible_user": "ubuntu"
            }
        }
    },
    "all": {
        "children": [
            "all_servers",
            "ungrouped"
        ]
    },
    "all_servers": {
        "children": [
            "production_servers",
            "staging_servers",
            "test_servers"
        ]
    },
    "production_servers": {
        "hosts": [
            "AnsibleManagedNode-2"
        ]
    },
    "staging_servers": {
        "hosts": [
            "AnsibleManagedNode-1"
        ]
    },
    "test_servers": {
        "hosts": [
            "AnsibleManagedNode-3"
        ]
    }
}
```

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible-inventory --graph
@all:
|--@all_servers:
| |--@production_servers:
| | |--AnsibleManagedNode-2
| |--@staging_servers:
| | |--AnsibleManagedNode-1
| |--@test_servers:
| | |--AnsibleManagedNode-3
| |--@ungrouped:
```

- Run **ansible** command with **setup** module.

```
$ ansible staging_servers -m setup
```

Where :

staging_servers – name of the grouped servers from the inventory file.
setup – module that outputs information about Ansible Managed Node's .

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible staging_servers -m setup
AnsibleManagedNode-1 | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "172.31.25.100"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::cc:d9ff:feba:d004"
        ],
        "ansible_apparmor": {
            "status": "enabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "08/24/2006",
        "ansible_bios_vendor": "Xen",
        "ansible_bios_version": "4.11.amazon",
        "ansible_board_asset_tag": "NA",
        "ansible_board_name": "NA",
        "ansible_board_serial": "NA",
        "ansible_board_vendor": "NA",
        "ansible_board_version": "NA",
        "ansible_chassis_asset_tag": "NA",
        "ansible_chassis_serial": "NA",
        "ansible_chassis_vendor": "Xen",
        "ansible_chassis_version": "NA",
        "ansible_cmdline": {
            "BOOT_IMAGE": "/boot/vmlinuz-5.15.0-1026-aws",
        }
    }
}
```

- Run **ansible** command with **shell** module and **uptime** argument.

```
$ ansible all -m shell -a "uptime"
```

Where :

all – for all servers from the inventory file.
shell – a module that runs the shell on all servers.
“uptime” – an argument: command that is executed in shells on each servers group all

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible all -m shell -a "uptime"
AnsibleManagedNode-1 | CHANGED | rc=0 >>
11:40:02 up 6:04, 1 user, load average: 0.00, 0.00, 0.00
AnsibleManagedNode-2 | CHANGED | rc=0 >>
11:40:02 up 6:04, 1 user, load average: 0.08, 0.02, 0.01
AnsibleManagedNode-3 | CHANGED | rc=0 >>
11:40:02 up 6:04, 1 user, load average: 0.00, 0.00, 0.00
```

- Run **ansible** command that copies the file to all Ansible Managed Node's.

```
$ echo "Hello World!" > hello.txt
$ ansible all -m copy -a "src=hello.txt dest=/home" -b
```

Where :

all – for all servers from the inventory file.

copy – a module that copies file from Ansible Control to the Ansible Managed Node's.

“src” – an argument: the relative path to the file hello.txt on the Ansible Control Node

“dest” – an argument: absolute path on the Managed Node where hello.txt must be copied

“-b” – an option: run operations as this user (default=root)

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ echo "Hello World!" > hello.txt
ubuntu@ip-172-31-22-86:~/ansible/project-5$
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ll
total 20
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 12:04 .
drwxrwxr-x 7 ubuntu ubuntu 4096 Jan  8 10:54 ..
-rw-rw-r-- 1 ubuntu ubuntu   71 Jan  8 10:54 ansible.cfg
-rw-rw-r-- 1 ubuntu ubuntu  13 Jan  8 12:04 hello.txt
-rw-rw-r-- 1 ubuntu ubuntu  376 Jan  8 11:07 hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/project-5$
ubuntu@ip-172-31-22-86:~/ansible/project-5$ cat hello.txt
Hello World!
ubuntu@ip-172-31-22-86:~/ansible/project-5$
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible all -m copy -a "src=hello.txt dest=/home" -b
AnsibleManagedNode-3 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "checksum": "a0b65939670bc2c010f4d5d6a0b3e4e4590fb92b",
    "dest": "/home/hello.txt",
    "gid": 0,
    "group": "root",
    "md5sum": "8ddd8be4b179a529afa5f2ffae4b9858",
    "mode": "0644",
    "owner": "root",
    "size": 13,
    "src": "/home/ubuntu/.ansible/tmp/ansible-tmp-1673179554.1568837-4314-60291496338293/source"
},
    "state": "file",
    "uid": 0
}
AnsibleManagedNode-1 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "checksum": "a0b65939670bc2c010f4d5d6a0b3e4e4590fb92b",
    "dest": "/home/hello.txt",
    "gid": 0,
    "group": "root",
    "md5sum": "8ddd8be4b179a529afa5f2ffae4b9858",
    "mode": "0644",
    "owner": "root",
    "size": 13,
    "src": "/home/ubuntu/.ansible/tmp/ansible-tmp-1673179554.1690552-4313-19861402793851/source"
},
    "state": "file",
    "uid": 0
}
AnsibleManagedNode-2 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "checksum": "a0b65939670bc2c010f4d5d6a0b3e4e4590fb92b",
    "dest": "/home/hello.txt",
    "gid": 0,
    "group": "root",
    "md5sum": "8ddd8be4b179a529afa5f2ffae4b9858",
```

- Run **ansible** command that checking copy command from previous point.

```
$ ansible all -m shell -a "ls -la /home" -b
```

Where :

all – for all servers from the inventory file.

shell – module that runs the shell on all servers.

“ls -la /home” – argument:output the content **/home** on the Ansible Managed Node

“-b” – option: run operations as this user (default=root)

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible all -m shell -a "ls -la /home" -b
AnsibleManagedNode-2 | CHANGED | rc=0 >>
total 16
drwxr-xr-x 3 root root 4096 Jan 8 12:05 .
drwxr-xr-x 19 root root 4096 Jan 8 05:35 ..
-rw-r--r-- 1 root root 13 Jan 8 12:05 hello.txt ←
drwxr-x--- 6 ubuntu ubuntu 4096 Jan 8 12:05 ubuntu
AnsibleManagedNode-3 | CHANGED | rc=0 >>
total 16
drwxr-xr-x 3 root root 4096 Jan 8 12:05 .
drwxr-xr-x 19 root root 4096 Jan 8 05:35 ..
-rw-r--r-- 1 root root 13 Jan 8 12:05 hello.txt ←
drwxr-x--- 6 ubuntu ubuntu 4096 Jan 8 12:05 ubuntu
AnsibleManagedNode-1 | CHANGED | rc=0 >>
total 16
drwxr-xr-x 3 root root 4096 Jan 8 12:05 .
drwxr-xr-x 19 root root 4096 Jan 8 05:35 ..
-rw-r--r-- 1 root root 13 Jan 8 12:05 hello.txt ←
drwxr-x--- 6 ubuntu ubuntu 4096 Jan 8 12:05 ubuntu
ubuntu@ip-172-31-22-86:~/ansible/project-5$
```

- Run **ansible** command that delete hello.txt file from Ansible managed Node's.

```
$ ansible all -m file -a "path=/home/hello.txt state=absent" -b
```

Where :

file – module that manage files and file properties.

“path” – argument: path to the file being managed on the Ansible Managed Node

“state” – argument: if absent, directories will be recursively deleted.

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible all -m file -a "path=/home/hello.txt state=absent" -b
AnsibleManagedNode-2 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "path": "/home/hello.txt",
    "state": "absent"
}
AnsibleManagedNode-1 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "path": "/home/hello.txt",
    "state": "absent"
}
AnsibleManagedNode-3 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "path": "/home/hello.txt",
    "state": "absent"
}
```

- Run **ansible** command that install Apache Web Server on the Ansible Managed Node's.

```
$ ansible production_servers -m apt -a "name=apache2 state=latest" -b
```

Where :

apt – module that manages apt packages (such as for Debian/Ubuntu).

“name” – argument: a list of package names.

“state” – argument: indicates the desired package state. *latest* ensures that the latest version is installed.

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible production_servers -m apt -a
"name=apache2 state=latest" -b
AnsibleManagedNode-2 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1673173644,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists ... \nBuilding dependency tree ... \nReading sta
    "Created symlink /etc/systemd/system/multi-user.target.wants/apache2.serv
ice → /lib/systemd/system/apache2.service.",
    "Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcac
heclean.service → /lib/systemd/system/apache-htcacheload.service.",
    "Processing triggers for ufw (0.36.1-4build1) ...",
    "Processing triggers for man-db (2.10.2-1) ...",
    "Processing triggers for libc-bin (2.35-0ubuntu3.1) ...",
    "NEEDRESTART-VER: 3.5",
    "NEEDRESTART-KCUR: 5.15.0-1026-aws",
    "NEEDRESTART-KEXP: 5.15.0-1026-aws",
    "NEEDRESTART-KSTA: 1"
}
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible production_servers -m apt -a "name=apache2 state=latest" -b
AnsibleManagedNode-2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1673173644,
    "cache_updated": false,
    "changed": false
}
```

- Run **ansible** command that start and enable Apache Web Server on the Ansible Managed.

```
$ ansible production_servers -m service -a "name=apache2
state=started enabled=yes" -b
```

Where :

service – module controls services on remote hosts.

“name” – argument: a list of package names.

“state” – argument: started/stopped are idempotent actions that will not run commands unless necessary. At least one of state and enabled are required

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible production_servers -m service -a "name=apache2 state=started enabled=yes" -b
AnsibleManagedNode-2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "enabled": true,
    "name": "apache2",
    "state": "started",
    "status": {
        "ActiveEnterTimestamp": "Sun 2023-01-08 12:48:11 UTC",
        "ActiveExitTimestamp": null,
        "ActiveState": "active"
    }
}
```

- Checking that our Apache Web Server is running.

Instances (1/6) Info

Name	Instance ID	Instance state	Instance type
AnsibleController	i-06d3a841429a42a6a	Running	t2.micro
AnsibleManagedNode-1	i-00438d7e753170b0b	Running	t2.micro
AnsibleManagedNode-2	i-0c04df37d47fa0701	Running	t2.micro
AnsibleManagedNode-3	i-077ef056384a869be	Running	t2.micro
JenkinsServer	i-0655a9022e4f52489	Stopped	t2.micro
MyWebServer	i-0e49692a5bf1e282b	Stopped	t2.micro

Instance: i-0c04df37d47fa0701 (AnsibleManagedNode-2)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary [Info](#)

Instance ID i-0c04df37d47fa0701 (AnsibleManagedNode-2)	Public IPv4 address 3.121.162.144 open address	Private IPv4 addresses 172.31.31.88
--	---	--

3.121.162.144

Apache2 Default Page

Ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   '-- ports.conf
|-- mods-enabled
|   '-- *.Load
|   '-- *.conf
|-- conf-enabled
|   '-- *.conf
|-- sites-enabled
|   '-- *.conf
```

- Remove Apache Web Server from Ansible Managed Node.

```
$ ansible production_servers -m apt -a "name=apache2 state=absent" -b
```

Where :

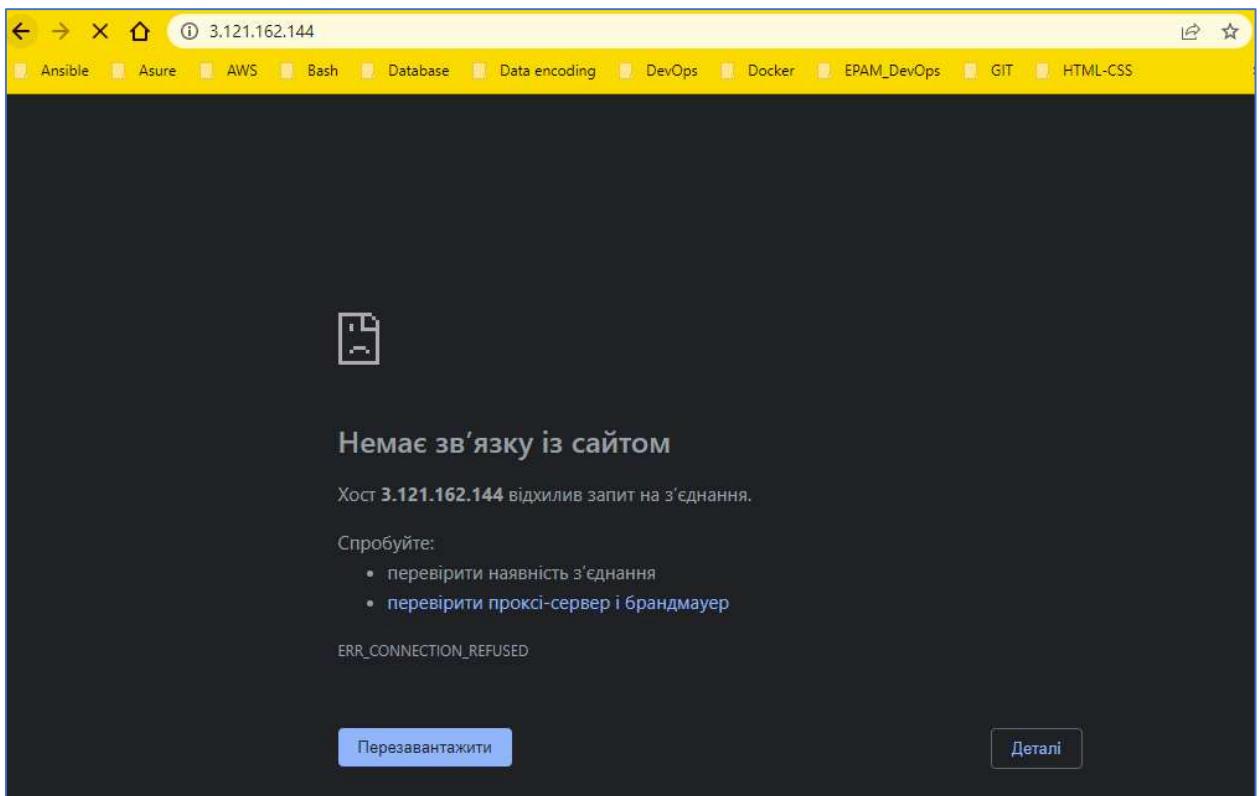
apt – module that manages apt packages (such as for Debian/Ubuntu).

“name” – argument: a list of package names.

“state” – argument: *absent* will remove the specified package.

```
ubuntu@ip-172-31-22-86:~/ansible/project-5$ ansible production_servers -m apt -a "name=apache2 state=absent" -b
AnsibleManagedNode-2 | CHANGED => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists... \nBuilding dependency tree... \nReading state information... \nThe following

```



9. My first Ansible playbook.

Task: Test connection to all servers, owned by your organization.

- Creation first **playbook1.yml**

```
ubuntu@ip-172-31-22-86:~/ansible$ mkdir playbook-1/
ubuntu@ip-172-31-22-86:~/ansible$ cd playbook-1/
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$ nano playbook1.yml ←
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$ ll
total 20
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 17:15 .
drwxrwxr-x 8 ubuntu ubuntu 4096 Jan  8 17:04 ..
-rw-rw-r-- 1 ubuntu ubuntu    71 Jan  8 17:15 ansible.cfg
-rw-rw-r-- 1 ubuntu ubuntu   376 Jan  8 17:15 hosts.txt
-rw-rw-r-- 1 ubuntu ubuntu   104 Jan  8 17:07 playbook1.yml
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$ cat ansible.cfg
[defaults]
host_key_checking = false
inventory      = ./hosts.txt
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$ cat hosts.txt
[staging_servers]
AnsibleManagedNode-1  ansible_host=3.71.18.221

[test_servers]
AnsibleManagedNode-3  ansible_host=3.120.178.95

[production_servers]
AnsibleManagedNode-2  ansible_host=3.121.162.144

[all_servers:children]
staging_servers
test_servers
production_servers

[all_servers:vars]
ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/.ssh/ansible_key
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook1.yml
```

```
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$ ansible-playbook playbook1.yml
PLAY [Connection Testing] ****
TASK [Gathering Facts] ****
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]

TASK [Ping servers] ****
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]

PLAY RECAP ****
AnsibleManagedNode-1      : ok=2    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
AnsibleManagedNode-2      : ok=2    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
AnsibleManagedNode-3      : ok=2    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0

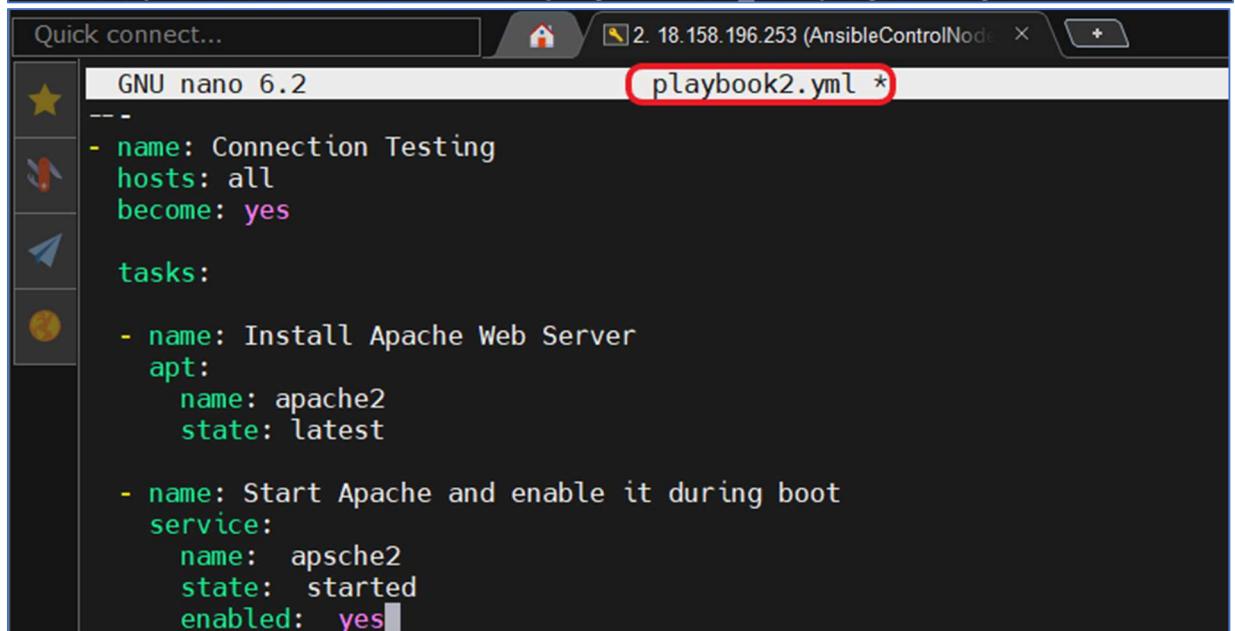
ubuntu@ip-172-31-22-86:~/ansible/playbook-1$
```

10. My second Ansible playbook.

Task: Install Apache Web Server on all servers, owned by your organization. Also needed to start Apache Web Server and enable it during boot.

- Creation project directory, file **hosts.txt** with description our managed node's and **ansible.cfg** file with **host_key_checking**, **inventory** parametrs and **playbook2.yml** with Apache Web Server instalation.

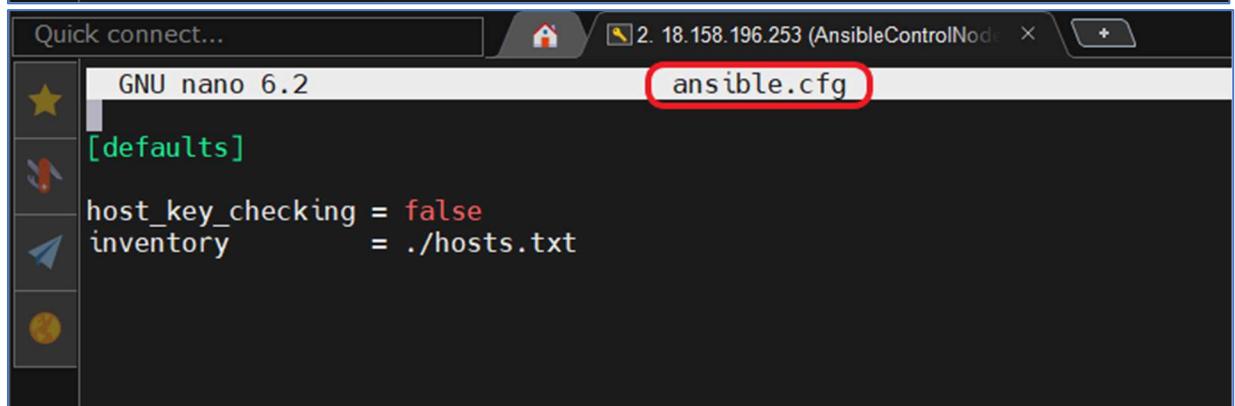
```
ubuntu@ip-172-31-22-86:~/ansible$ cp -r playbook-1/ playbook-2/
ubuntu@ip-172-31-22-86:~/ansible$ 
ubuntu@ip-172-31-22-86:~/ansible$ cd playbook-2/
ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ 
ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ mv playbook1.yml playbook2.yml
ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ 
ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ ll
total 20
drwxrwxr-x 2 ubuntu ubuntu 4096 Jan  8 17:29 .
drwxrwxr-x 9 ubuntu ubuntu 4096 Jan  8 17:29 ..
-rw-rw-r-- 1 ubuntu ubuntu   71 Jan  8 17:29 ansible.cfg
-rw-rw-r-- 1 ubuntu ubuntu  376 Jan  8 17:29 hosts.txt
-rw-rw-r-- 1 ubuntu ubuntu  104 Jan  8 17:29 playbook2.yml
ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ 
ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ nano playbook2.yml
```



```
GNU nano 6.2
playbook2.yml *
---
- name: Connection Testing
  hosts: all
  become: yes

  tasks:
    - name: Install Apache Web Server
      apt:
        name: apache2
        state: latest

    - name: Start Apache and enable it during boot
      service:
        name: apache2
        state: started
        enabled: yes
```



```
GNU nano 6.2
ansible.cfg
[defaults]
host_key_checking = false
inventory      = ./hosts.txt
```

```

GNU nano 6.2
hosts.txt

[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221

[test_servers]
AnsibleManagedNode-3 ansible_host=3.120.178.95

[production_servers]
AnsibleManagedNode-2 ansible_host=3.121.162.144

[all_servers:children]
staging_servers
test_servers
production_servers

[all_servers:vars]
ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/.ssh/ansible_key

```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook2.yml
```

```

ubuntu@ip-172-31-22-86:~/ansible/playbook-2$ ansible-playbook playbook2.yml

PLAY [Connection Testing] ****
TASK [Gathering Facts] ****
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-3]

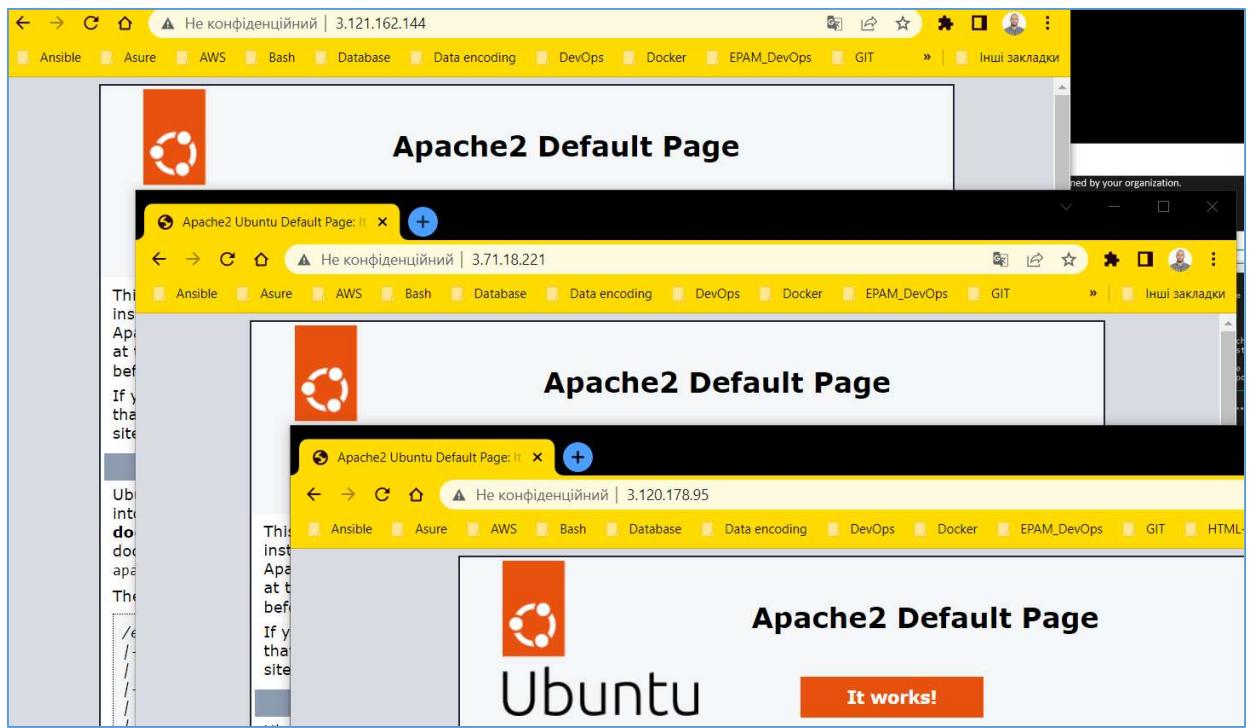
TASK [Install Apache Web Server] ****
changed: [AnsibleManagedNode-2]
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-1]

TASK [Start Apache and enable it during boot] ****
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]

PLAY RECAP ****
AnsibleManagedNode-1      : ok=3    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
AnsibleManagedNode-2      : ok=3    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0
AnsibleManagedNode-3      : ok=3    changed=0    unreachable=0    failed=0
kipped=0    rescued=0    ignored=0

```

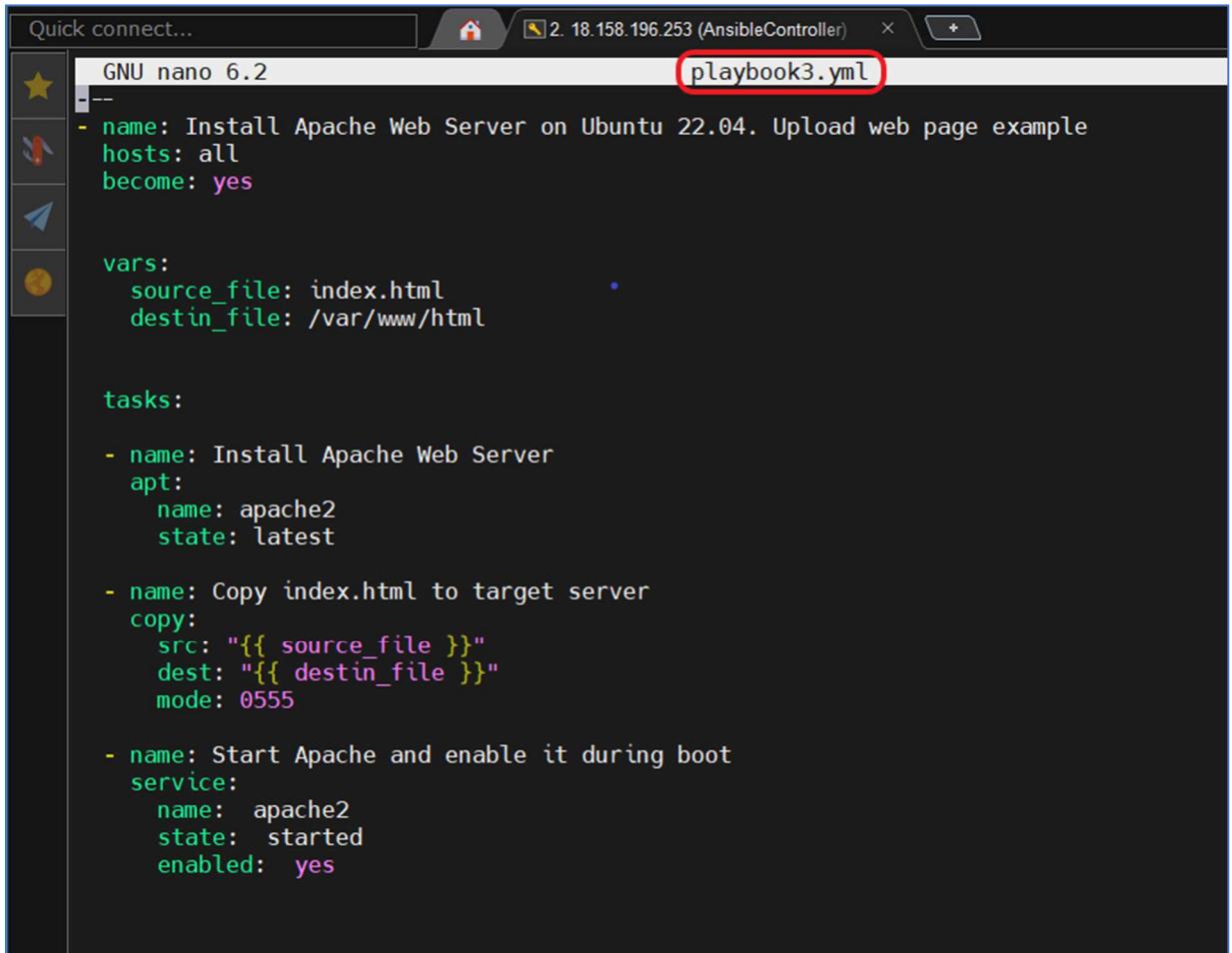
- Checking the availability of the Apache 2 default page on all Ansible Managed Node's.



11. My third Ansible playbook.

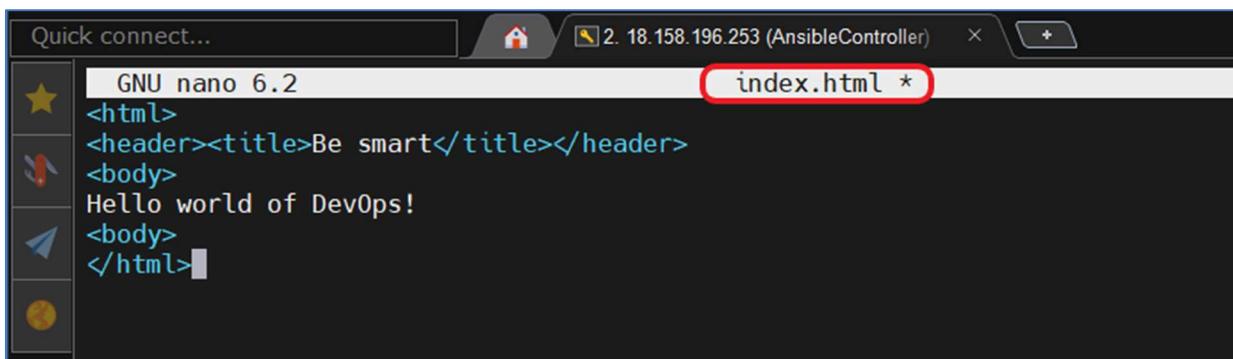
Task: Install Apache Web Server on all servers, owned by your organization. Also needed to start Apache Web Server and enable it during boot. Also needed to change default index.html

- Creation **playbook3.yml** with Apache Web Server instalation and copying **index.html**.



```
Quick connect... 2. 18.158.196.253 (AnsibleController) +  
GNU nano 6.2 playbook3.yml  
--  
- name: Install Apache Web Server on Ubuntu 22.04. Upload web page example  
hosts: all  
become: yes  
  
vars:  
  source_file: index.html  
  destin_file: /var/www/html  
  
tasks:  
- name: Install Apache Web Server  
  apt:  
    name: apache2  
    state: latest  
  
- name: Copy index.html to target server  
  copy:  
    src: "{{ source_file }}"  
    dest: "{{ destin_file }}"  
    mode: 0555  
  
- name: Start Apache and enable it during boot  
  service:  
    name: apache2  
    state: started  
    enabled: yes
```

- Creation **index.html** file.



```
Quick connect... 2. 18.158.196.253 (AnsibleController) +  
GNU nano 6.2 index.html *  
<html>  
<header><title>Be smart</title></header>  
<body>  
Hello world of DevOps!  
</body>  
</html>
```

- Working directory of the project **playbook-3**.

```
ubuntu@ip-172-31-22-86:~/ansible/playbook-3$ ll
total 24
drwxrwxr-x  2 ubuntu ubuntu 4096 Jan  8 18:49 .
drwxrwxr-x 10 ubuntu ubuntu 4096 Jan  8 18:37 ../
-rw-rw-r--  1 ubuntu ubuntu   71 Jan  8 18:37 ansible.cfg
-rw-rw-r--  1 ubuntu ubuntu  376 Jan  8 18:37 hosts.txt
-rw-rw-r--  1 ubuntu ubuntu  93 Jan  8 18:49 index.html
-rw-rw-r--  1 ubuntu ubuntu 530 Jan  8 18:45 playbook3.yml
ubuntu@ip-172-31-22-86:~/ansible/playbook-3$
ubuntu@ip-172-31-22-86:~/ansible/playbook-3$
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook3.yml
```

```
ubuntu@ip-172-31-22-86:~/ansible/playbook-3$ ansible-playbook playbook3.yml

PLAY [Install Apache Web Server on Ubuntu 22.04. Upload web page example] *****

TASK [Gathering Facts] *****
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]

TASK [Install Apache Web Server] *****
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-2]
changed: [AnsibleManagedNode-3]

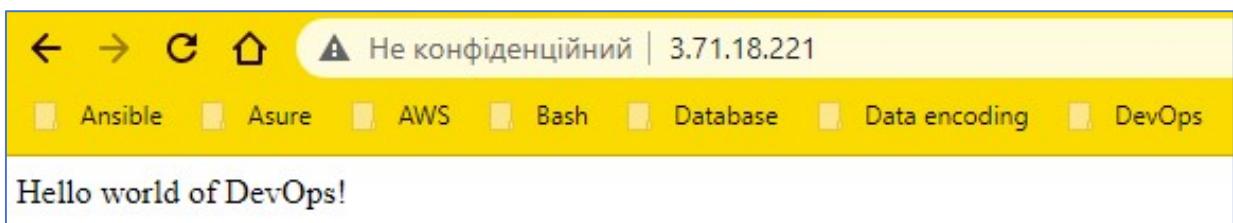
TASK [Copy index.html to target server] *****
changed: [AnsibleManagedNode-2]
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-3]

TASK [Start Apache and enable it during boot] *****
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-1]

PLAY RECAP *****
AnsibleManagedNode-1      : ok=4    changed=2    unreachable=0    failed=0    skipped=0
d=0          ignored=0
AnsibleManagedNode-2      : ok=4    changed=2    unreachable=0    failed=0    skipped=0
d=0          ignored=0
AnsibleManagedNode-3      : ok=4    changed=2    unreachable=0    failed=0    skipped=0
d=0          ignored=0

ubuntu@ip-172-31-22-86:~/ansible/playbook-3$
```

- Checking the availability of the **index.html** page on all Ansible Managed Node's.



- Restart Apache after copying **index.html** to the Ansible Managed Node's using **notify** and **handlers**.

```

Quick connect... 2. 18.158.196.253 (AnsibleController) +
GNU nano 6.2      playbook3.yml *
-- 
- name: Install Apache Web Server on Ubuntu 22.04. Upload web page example
  hosts: all
  become: yes

  vars:
    source_file: index.html
    destin_file: /var/www/html

  tasks:
    - name: Install Apache Web Server
      apt:
        name: apache2
        state: latest

    - name: Copy index.html to target server
      copy:
        src: "{{ source_file }}"
        dest: "{{ destin_file }}"
        mode: 0555
      notify: Restart Apache ←

    - name: Start Apache and enable it during boot
      service:
        name: apache2
        state: started
        enabled: yes

  handlers:
    - name: Restart Apache
      service:
        name: apache2
        state: restarted

```

- Changing **index.html** file.

```

Quick connect... 2. 18.158.196.253 (AnsibleController) +
GNU nano 6.2      index.html *
<html>
<header><title>Be smart</title></header>
<body>
Hello world of DevOps!
Some changing has been made ... again and again! ←
</body>
</html>

```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook3.yml
```

```
ubuntu@ip-172-31-22-86:~/ansible/playbook-3$ ansible-playbook playbook3.yml
PLAY [Install Apache Web Server on Ubuntu 22.04. Upload web page example] *****,
TASK [Gathering Facts] *****
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]

TASK [Install Apache Web Server] *****
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-1]

TASK [Copy index.html to target server] *****
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-2]

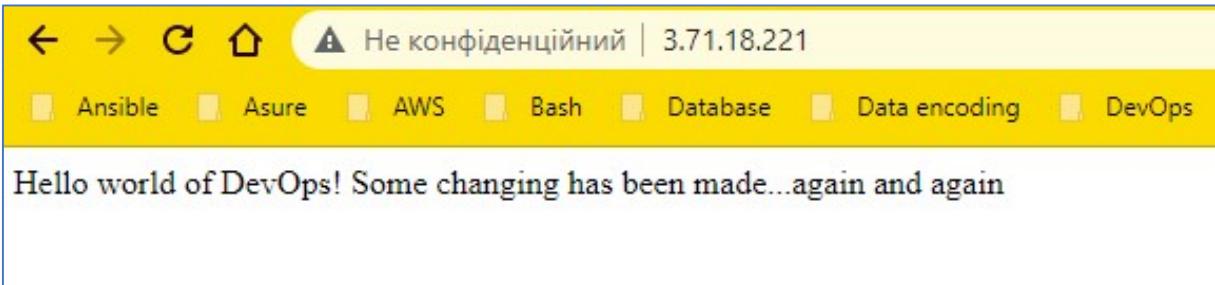
TASK [Start Apache and enable it during boot] *****
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

RUNNING HANDLER [Restart Apache] *****
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-2]

PLAY RECAP *****
AnsibleManagedNode-1      : ok=5    changed=2    unreachable=0    failed=0    skipped=0
d=0          ignored=0
AnsibleManagedNode-2      : ok=5    changed=2    unreachable=0    failed=0    skipped=0
d=0          ignored=0
AnsibleManagedNode-3      : ok=5    changed=2    unreachable=0    failed=0    skipped=0
d=0          ignored=0

ubuntu@ip-172-31-22-86:~/ansible/playbook-3$
```

- Checking the availability of the **index.html** page on all Ansible Managed Node's.



12. My fourth Ansible playbook.

Task: Install Apache Web Server on all servers, owned by your organization. Also needed to start Apache Web Server and enable it during boot. Also needed to change default index.html. Servers are on different Linux-based OS(Amazon Linux and Ubuntu).

- Creation files **playbook4.yml**, **hosts.txt**, **group_vars/ami_cred**, **group_vars/deb_cred**, **ansible.cfg**

The screenshot shows three terminal windows. The top window lists files in the directory `/home/ubuntu/ansible/playbook-4`. The middle window shows the `ansible.cfg` file being edited in nano, with the `hosts` section highlighted. The bottom window shows the `hosts.txt` file being edited in nano, with the `[dev_servers]` group highlighted.

File Structure:

```
/home/ubuntu/ansible/playbook-4/
  - playbook4.yml
  - hosts.txt
  - group_vars/
    - ami_cred
    - deb_cred
  - ansible.cfg
  - hosts.txt
  - index.html
```

Content of ansible.cfg (highlighted in red box):

```
host_key_checking = false
inventory = ./hosts.txt
```

Content of hosts.txt (highlighted in red box):

```
[dev_servers]
AnsibleManagedNode-4 ansible_host=54.93.62.109

[staging_servers]
AnsibleManagedNode-1 ansible_host=3.71.18.221

[test_servers]
AnsibleManagedNode-3 ansible_host=3.120.178.95

[production_servers]
AnsibleManagedNode-2 ansible_host=3.121.162.144

[deb_cred:children]
staging_servers
test_servers
production_servers

[ami_cred:children]
dev_servers
```

Quick connect... 2. 18.158.196.253 (AnsibleControlNode) 3. 54.93.62.109 (AnsibleManagedNode)

GNU nano 6.2 ami_cred

```
---  
ansible_user : ec2-user  
ansible_ssh_private_key_file : /home/ubuntu/.ssh/jenkins-frankfurt.pem
```

Quick connect... 2. 18.158.196.253 (AnsibleControlNode) 3. 54.93.62.109 (AnsibleManagedNode)

GNU nano 6.2 deb_cred

```
---  
ansible_user : ubuntu  
ansible_ssh_private_key_file : /home/ubuntu/.ssh/ansible_key
```

Quick connect... 2. 18.158.196.253 (AnsibleControlNode) 3. 54.93.62.109 (AnsibleManagedNode)

GNU nano 6.2 playbook4.yml

```
- name: Install Apache Web Server on Ubuntu22.04 and AMI Linux. Upload web page.  
hosts: all  
become: yes  
  
vars:  
  source_file: index.html  
  destin_file: /var/www/html  
  
tasks:  
- name: Check Linux distro  
  debug:  
    var: ansible_os_family  
  
- name: Install Apache Web Server on Debian Family  
  apt:  
    name: apache2  
    state: latest  
  when: ansible_os_family == "Debian"  
  
- name: Install Apache Web Server on RedHat Family  
  yum:  
    name: httpd  
    state: latest  
  when: ansible_os_family == "RedHat"  
  
- name: Copy index.html to target server  
  copy:  
    src: "{{ source_file }}"  
    dest: "{{ destin_file }}"  
    mode: 0555  
  #notify: Restart Apache  
  
- name: Start Apache and enable it during boot Ubuntu  
  service:  
    name: apache2  
    state: started  
    enabled: yes  
  when: ansible_os_family == "Debian"  
  
- name: Start Apache and enable it during boot Amazon Linux  
  service:  
    name: httpd  
    state: started  
    enabled: yes  
  when: ansible_os_family == "RedHat"
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook4.yml
```

```

ubuntu@ip-172-31-22-86:~/ansible/playbook-4$ ansible-playbook playbook4.yml
PLAY [Install Apache Web Server on Ubuntu22.04 and AMI Linux. Upload web page.] *****
TASK [Gathering Facts] *****
[WARNING]: Platform linux on host AnsibleMabagedNode-4 is using the discovered
Python interpreter at /usr/bin/python3.7, but future installation of another
Python interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.14/reference_appendices/interpreter_discovery.html for more information.
ok: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-1]

TASK [Check Linux distro] *****
ok: [AnsibleManagedNode-1] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-3] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-2] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleMabagedNode-4] => {
    "ansible_os_family": "RedHat"
}

TASK [Install Apache Web Server on Debian Family] *****
skipping: [AnsibleMabagedNode-4]
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-2]

TASK [Install Apache Web Server on RedHat Family] *****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

TASK [Copy index.html to target server] *****
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

TASK [Start Apache and enable it during boot Ubuntu] *****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]

TASK [Start Apache and enable it during boot Amazon Linux] *****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

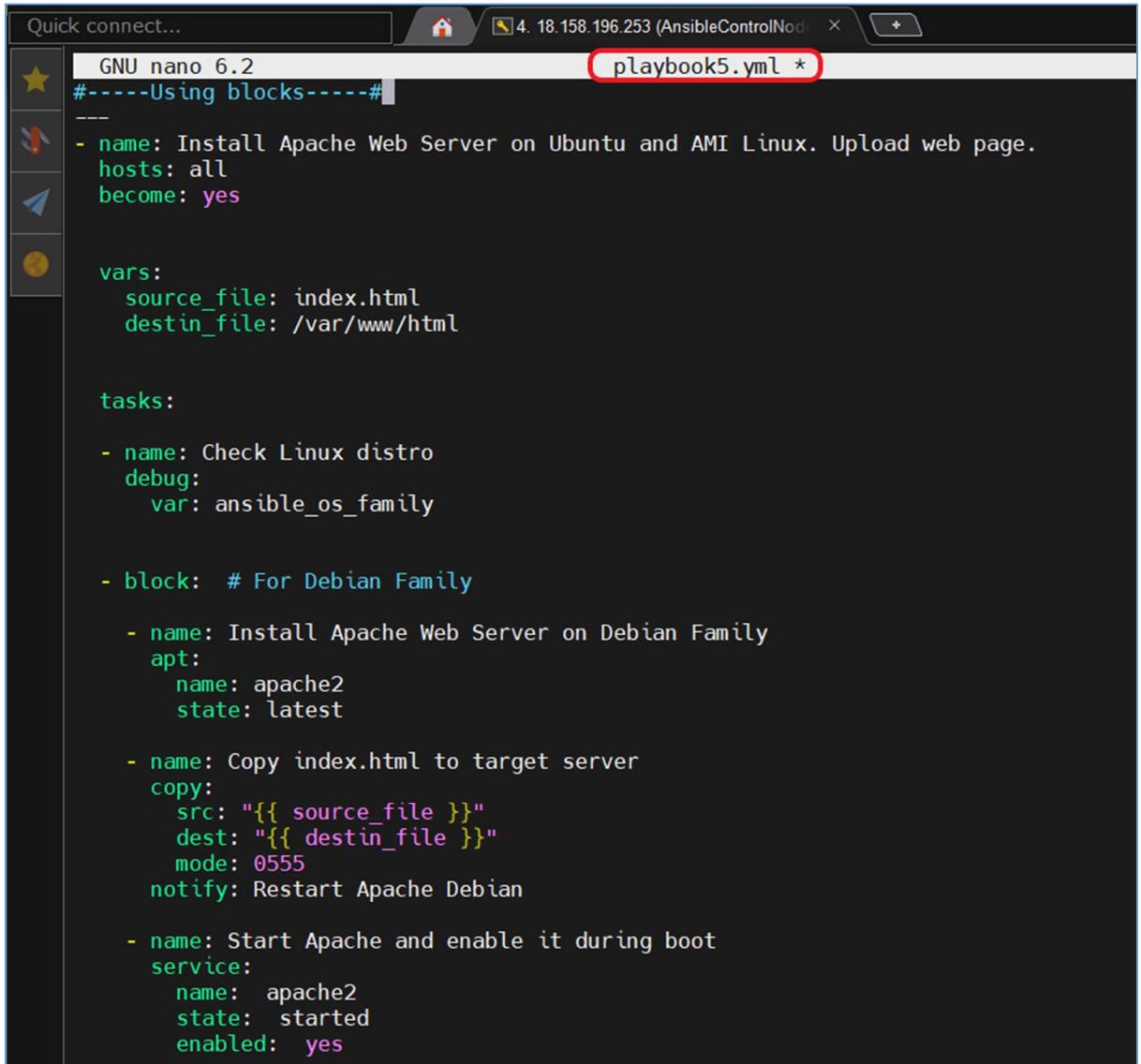
PLAY RECAP *****
AnsibleMabagedNode-4      : ok=5    changed=3    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0
AnsibleManagedNode-1      : ok=5    changed=1    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0
AnsibleManagedNode-2      : ok=5    changed=1    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0
AnsibleManagedNode-3      : ok=5    changed=1    unreachable=0    failed=0
kipped=2    rescued=0    ignored=0

```

13. My fifth Ansible playbook.

Task: Install Apache Web Server on all servers, owned by your organization. Also needed to start Apache Web Server and enable it during boot. Also needed to change default index.html. Servers are on different Linux-based OS(Amazon Linux and Ubuntu).

- Creation file **playbook5.yml** with using blocks parametr (files **hosts.txt**, **group_vars/ami_cred**, **group_vars/deb_cred**, **ansible.cfg** are used from the previous task)



The screenshot shows a terminal window titled "GNU nano 6.2" with the file "playbook5.yml" open. The terminal interface includes a toolbar with icons for quick connect, a home icon, and a plus sign. The file content is as follows:

```
#----Using blocks----#
---
- name: Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.
  hosts: all
  become: yes

vars:
  source_file: index.html
  destin_file: /var/www/html

tasks:
  - name: Check Linux distro
    debug:
      var: ansible_os_family

  - block: # For Debian Family
    - name: Install Apache Web Server on Debian Family
      apt:
        name: apache2
        state: latest

    - name: Copy index.html to target server
      copy:
        src: "{{ source_file }}"
        dest: "{{ destin_file }}"
        mode: 0555
      notify: Restart Apache Debian

    - name: Start Apache and enable it during boot
      service:
        name: apache2
        state: started
        enabled: yes
```

Quick connect...

GNU nano 6.2

playbook5.yml *

```
when: ansible_os_family == "Debian"

- block: # For RedHat Family

  - name: Install Apache Web Server on RedHat Family
    yum:
      name: httpd
      state: latest

  - name: Copy index.html to target server
    copy:
      src: "{{ source_file }}"
      dest: "{{ destin_file }}"
      mode: 0555
    notify: Restart Apache RedHat

  - name: Start Apache and enable it during boot
    service:
      name: httpd
      state: started
      enabled: yes

when: ansible_os_family == "RedHat"

handlers:
- name: Restart Apache Debian
  service:
    name: apache2
    state: restarted

- name: Restart Apache RedHat
  service:
    name: httpd
    state: restarted
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook5.yml
```

```

Quick connect... 4. 18.158.196.253 (AnsibleControlNode)
ubuntu@ip-172-31-22-86:~/ansible/playbook-5$ ansible-playbook playbook5.yml
PLAY [Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.] ****
TASK [Gathering Facts] ****
[WARNING]: Platform linux on host AnsibleMabagedNode-4 is using the discovered Python
interpreter at /usr/bin/python3.7, but future installation of another Python interpreter
could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.14/reference_appendices/interpreter_discovery.html for more information.
ok: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [Check Linux distro] ****
ok: [AnsibleManagedNode-1] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-3] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-2] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleMabagedNode-4] => {
    "ansible_os_family": "RedHat"
}

TASK [Install Apache Web Server on Debian Family] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-1]

TASK [Copy index.html to target server] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [Start Apache and enable it during boot] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]

TASK [Install Apache Web Server on RedHat Family] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

TASK [Copy index.html to target server] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

TASK [Start Apache and enable it during boot Amazon Linux] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

RUNNING HANDLER [Restart Apache RedHat] ****
changed: [AnsibleMabagedNode-4]

PLAY RECAP ****
AnsibleMabagedNode-4      : ok=6    changed=4    unreachable=0    failed=0    skipped=3
escued=0  ignored=0
AnsibleManagedNode-1      : ok=5    changed=0    unreachable=0    failed=0    skipped=3
escued=0  ignored=0
AnsibleManagedNode-2      : ok=5    changed=0    unreachable=0    failed=0    skipped=3
escued=0  ignored=0
AnsibleManagedNode-3      : ok=5    changed=0    unreachable=0    failed=0    skipped=3
escued=0  ignored=0

```

14. My sixth Ansible playbook. Loops.

Task: Make some actions on remote server, results should be viewed by output console.

- Creation file **playbook_loop1.yml**

```
GNU nano 6.2
#-----Loops-----#
---
- name: Loop Playbook
  hosts: AnsibleManagedNode-1
  become: yes

  tasks:
    - name: Check Linux distro
      debug: msg="Hello {{ item }}"
      with_items:
        - "Frontend"
        - ".Net"
        - "Java"

    - name: Until example
      shell: echo -n A >> example.txt & cat example.txt
      register: output
      delay: 2
      retries: 5
      until: output.stdout.find("AAA") == false

    - name: Print output
      debug:
        var: output.stdout
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook_loop1.yml
```

```
ubuntu@ip-172-31-22-86:~/ansible/playbook-6$ ansible-playbook playbook_loop1.yml
PLAY [Loop Playbook] ****
TASK [Gathering Facts] ****
ok: [AnsibleManagedNode-1]

TASK [Check Linux distro] ****
ok: [AnsibleManagedNode-1] => (item=Frontend) => {
    "msg": "Hello Frontend"
}
ok: [AnsibleManagedNode-1] => (item=.Net) => {
    "msg": "Hello .Net"
}
ok: [AnsibleManagedNode-1] => (item=Java) => {
    "msg": "Hello Java"
}

TASK [Until example] ****
FAILED - RETRYING: [AnsibleManagedNode-1]: Until example (5 retries left).
FAILED - RETRYING: [AnsibleManagedNode-1]: Until example (4 retries left).
changed: [AnsibleManagedNode-1]

TASK [Print output] ****
ok: [AnsibleManagedNode-1] => {
    "output.stdout": "AAA"
}

PLAY RECAP ****
AnsibleManagedNode-1 : ok=4    changed=1    unreachable=0    failed=0    skipped=0
                      rescued=0   ignored=0

ubuntu@ip-172-31-22-86:~/ansible/playbook-6$
```

15. My seventh Ansible playbook. Loops2.

Task: Copy WebPage (some files) to remote server using loop.

- Creation file **playbook_loop2.yml**

```
GNU nano 6.2                               playbook_loop2.yml
-----
#----Loops example----#
---
- name: Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.
  hosts: all
  become: yes

  vars:
    source_dir: ./MyWebSite
    destin_dir: /var/www/html

  tasks:
    - name: Check Linux distro
      debug:
        var: ansible_os_family

    - block: # For Debian Family
      - name: Install Apache Web Server on Debian Family
        apt:
          name: apache2
          state: latest

      - name: Start Apache and enable it during boot
        service:
          name: apache2
          state: started
          enabled: yes

      when: ansible_os_family == "Debian"

    - block: # For RedHat Family
      - name: Install Apache Web Server on RedHat Family
        yum:
          name: httpd
          state: latest

      - name: Start Apache and enable it during boot
        service:
          name: httpd
          state: started
          enabled: yes

      when: ansible_os_family == "RedHat"

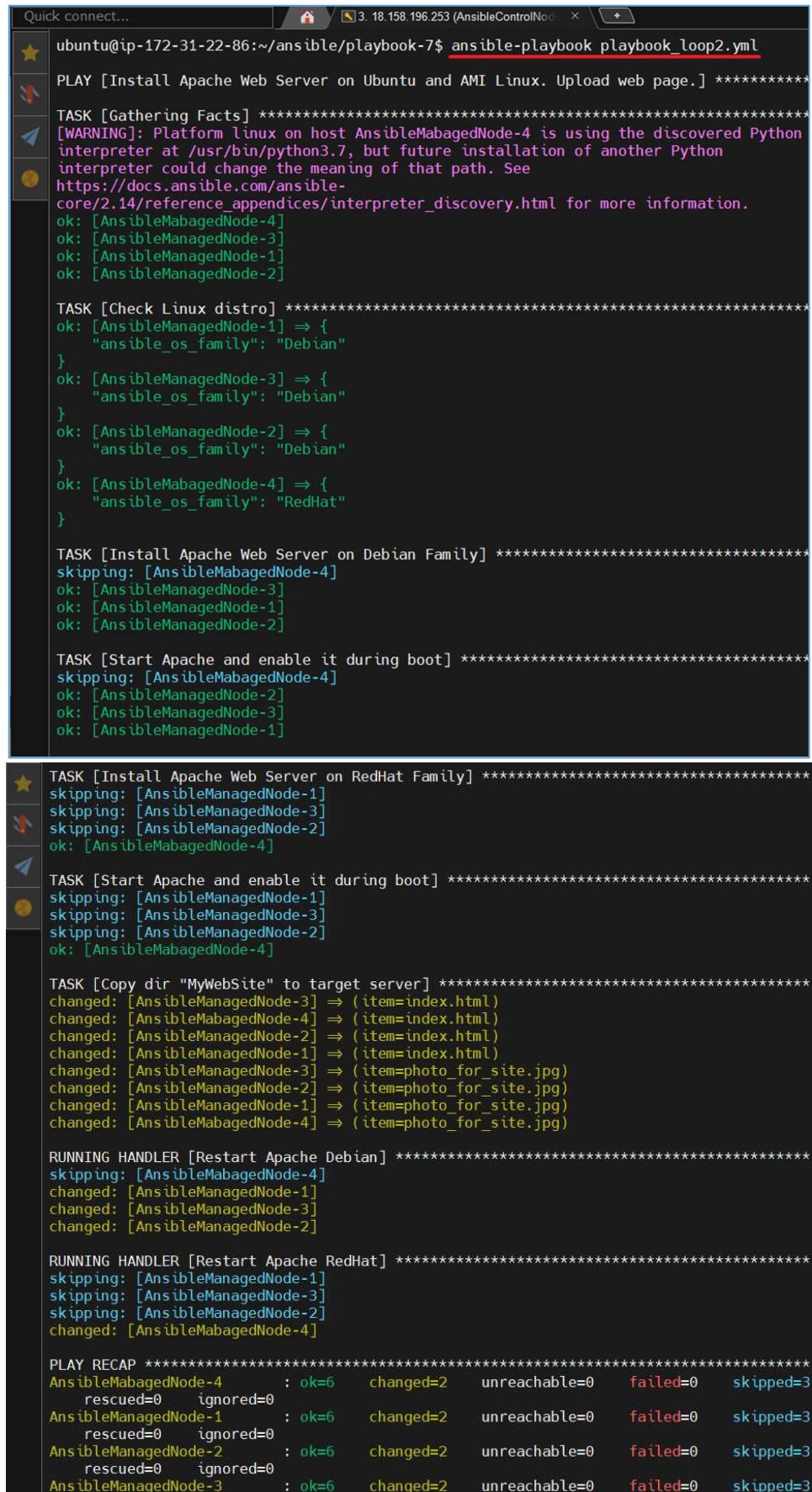
    - name: Copy dir "MyWebSite" to target server
      copy:
        src: "{{ source_dir }}/{{ item }}"
        dest: "{{ destin_dir }}"
        mode: 0555
      loop:
        - "index.html"
        - "photo_for_site.jpg"
      notify:
        - Restart Apache RedHat
        - Restart Apache Debian

  handlers:
    - name: Restart Apache Debian
      service:
        name: apache2
        state: restarted
      when: ansible_os_family == "Debian"

    - name: Restart Apache RedHat
      service:
        name: httpd
        state: restarted
      when: ansible_os_family == "RedHat"
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook_loop2.yml
```



```

Quick connect... 3. 18.158.196.253 (AnsibleControlNode)
ubuntu@ip-172-31-22-86:~/ansible/playbook-7$ ansible-playbook playbook_loop2.yml
PLAY [Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.] ****
TASK [Gathering Facts] ****
[WARNING]: Platform linux on host AnsibleMabagedNode-4 is using the discovered Python
interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.14/reference_appendices/interpreter_discovery.html for more information.
ok: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [Check Linux distro] ****
ok: [AnsibleManagedNode-1] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-3] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-2] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleMabagedNode-4] => {
    "ansible_os_family": "RedHat"
}

TASK [Install Apache Web Server on Debian Family] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [Start Apache and enable it during boot] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]

TASK [Install Apache Web Server on RedHat Family] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
ok: [AnsibleMabagedNode-4]

TASK [Start Apache and enable it during boot] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
ok: [AnsibleMabagedNode-4]

TASK [Copy dir "MyWebSite" to target server] ****
changed: [AnsibleManagedNode-3] => (item=index.html)
changed: [AnsibleMabagedNode-4] => (item=index.html)
changed: [AnsibleManagedNode-2] => (item=index.html)
changed: [AnsibleManagedNode-1] => (item=index.html)
changed: [AnsibleManagedNode-3] => (item=photo_for_site.jpg)
changed: [AnsibleManagedNode-2] => (item=photo_for_site.jpg)
changed: [AnsibleManagedNode-1] => (item=photo_for_site.jpg)
changed: [AnsibleMabagedNode-4] => (item=photo_for_site.jpg)

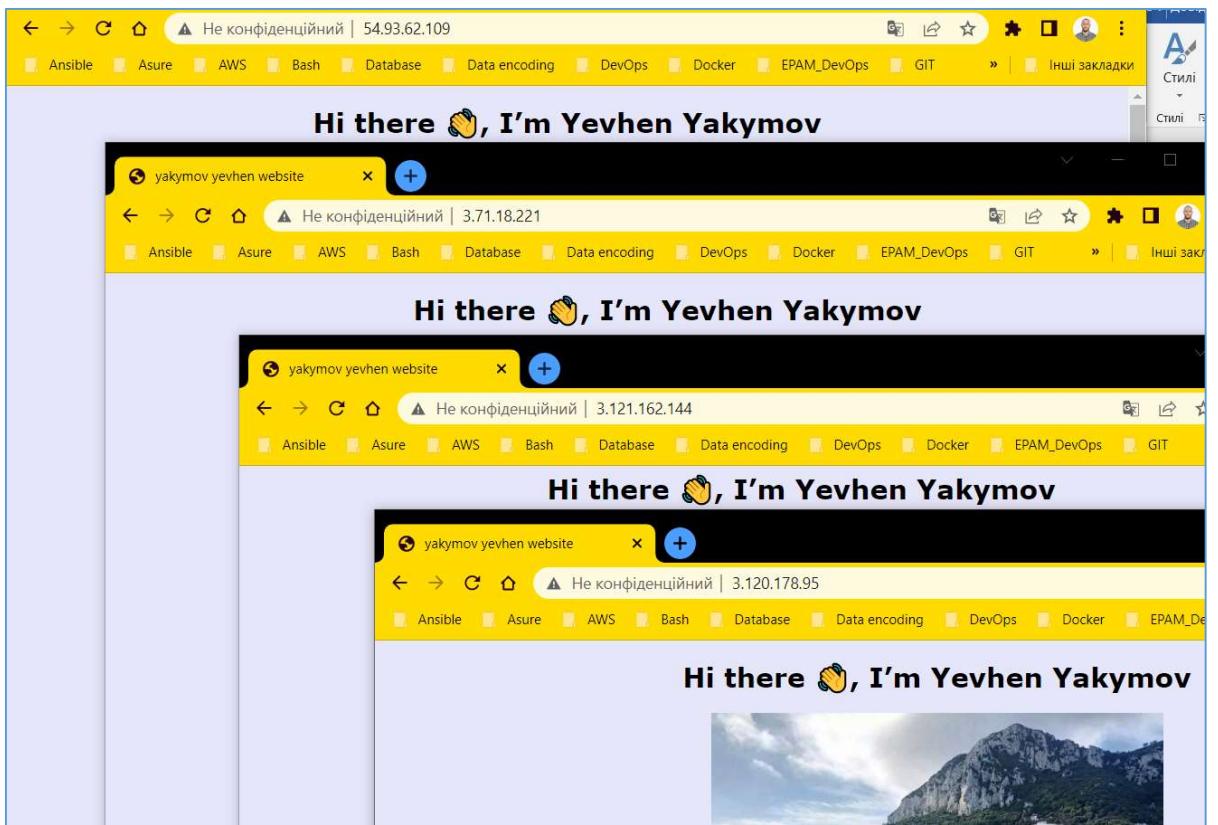
RUNNING HANDLER [Restart Apache Debian] ****
skipping: [AnsibleMabagedNode-4]
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-2]

RUNNING HANDLER [Restart Apache RedHat] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

PLAY RECAP ****
AnsibleMabagedNode-4 : ok=6    changed=2    unreachable=0    failed=0    skipped=3
AnsibleManagedNode-1 : ok=6    changed=2    unreachable=0    failed=0    skipped=3
AnsibleManagedNode-2 : ok=6    changed=2    unreachable=0    failed=0    skipped=3
AnsibleManagedNode-3 : ok=6    changed=2    unreachable=0    failed=0    skipped=3

```

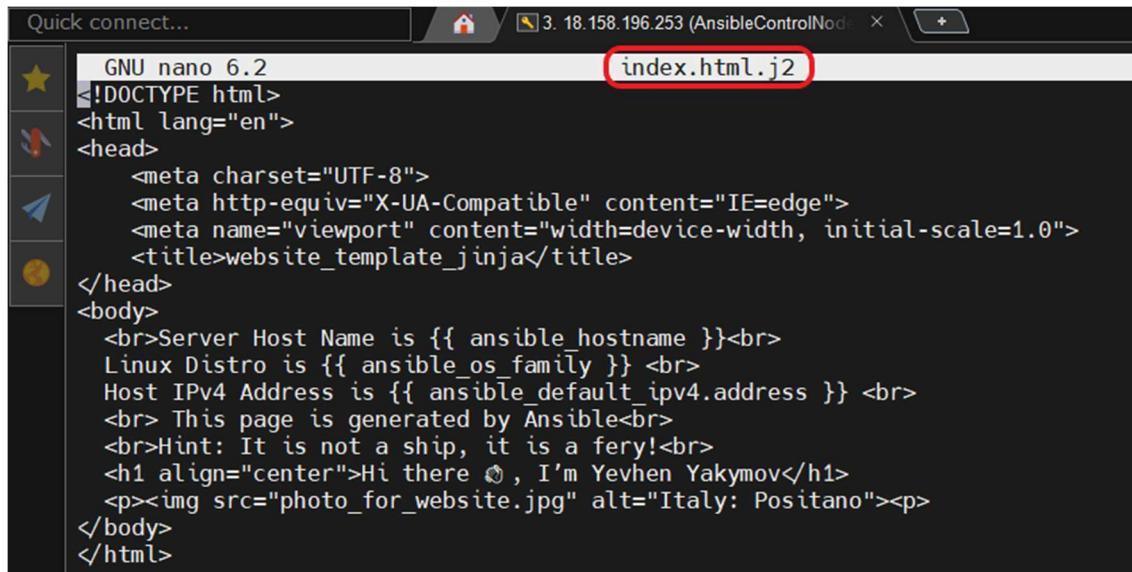
- Checking the availability of the **index.html** page on all Ansible Managed Node's.



16. My eight Ansible playbook. Templates and Jinja.

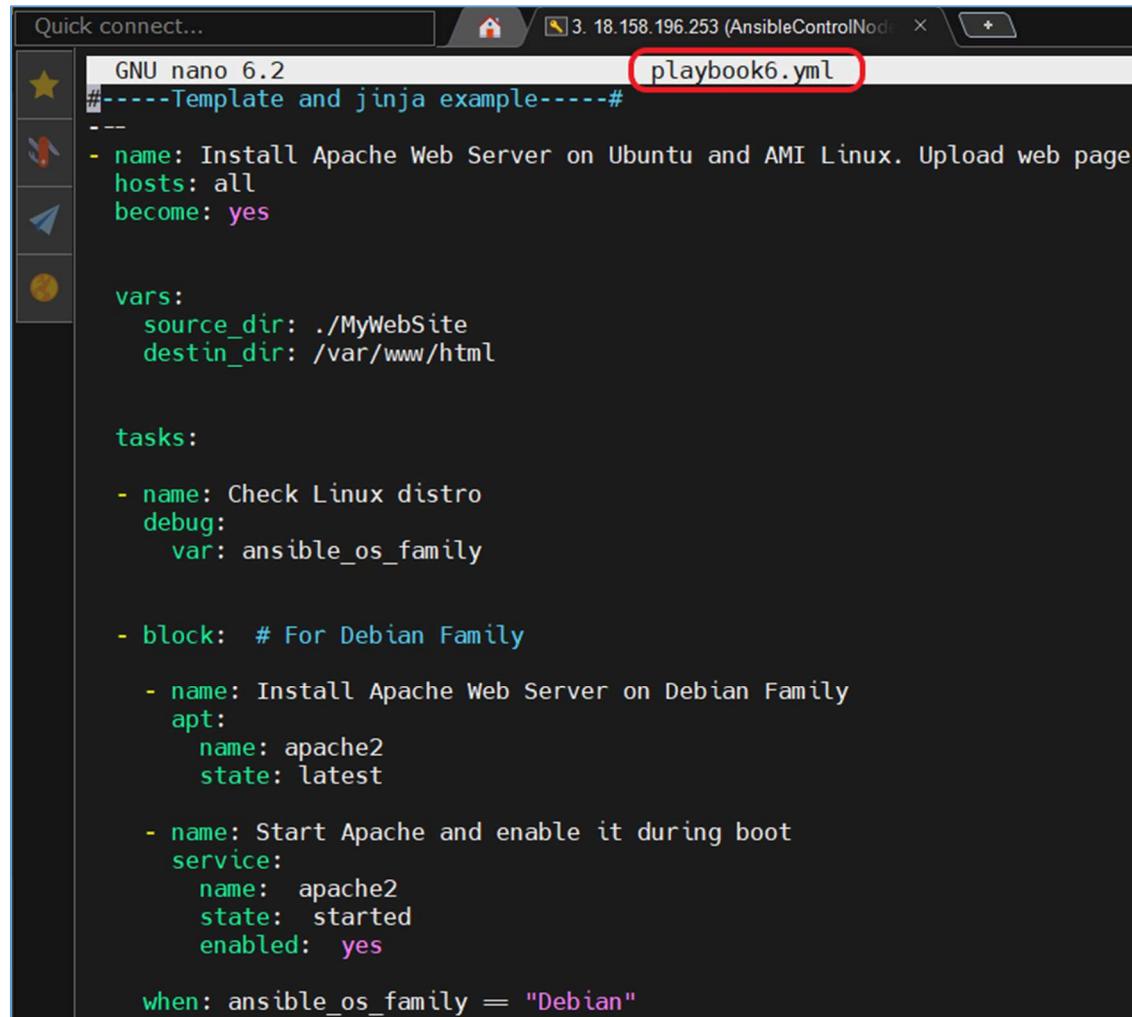
Task: Customize WebPage on remote servers using template and jinja.

- Creation file **index.html.j2**



```
GNU nano 6.2
index.html.j2
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>website_template_jinja</title>
</head>
<body>
    <br>Server Host Name is {{ ansible_hostname }}<br>
    Linux Distro is {{ ansible_os_family }} <br>
    Host IPv4 Address is {{ ansible_default_ipv4.address }} <br>
    <br> This page is generated by Ansible<br>
    <br>Hint: It is not a ship, it is a ferry!<br>
    <h1 align="center">Hi there ☺, I'm Yevhen Yakymov</h1>
    <p></p>
</body>
</html>
```

- Creation file **playbook6.yml**



```
GNU nano 6.2
playbook6.yml
#-----Template and jinja example-----#
---

- name: Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.
  hosts: all
  become: yes

  vars:
    source_dir: ./MyWebSite
    destin_dir: /var/www/html

  tasks:
    - name: Check Linux distro
      debug:
        var: ansible_os_family

    - block: # For Debian Family
        - name: Install Apache Web Server on Debian Family
          apt:
            name: apache2
            state: latest

        - name: Start Apache and enable it during boot
          service:
            name: apache2
            state: started
            enabled: yes

        when: ansible_os_family == "Debian"
```

```
- block: # For RedHat Family

  - name: Install Apache Web Server on RedHat Family
    yum:
      name: httpd
      state: latest

  - name: Start Apache and enable it during boot
    service:
      name: httpd
      state: started
      enabled: yes

when: ansible_os_family == "RedHat"

- name: Generate index.html using template
  template:
    src: "{{ source_dir }}/{{ index.html.j2 }}"
    dest: "{{ destin_dir }}/{{ index.html }}"
    mode: 0555
  notify:
    - Restart Apache RedHat
    - Restart Apache Debian

- name: Copy dir MyWebSite to target server
  copy:
    src: "{{ source_dir }}/{{ photo_for_site.jpg }}"
    dest: "{{ destin_dir }}"
    mode: 0555
  notify:
    - Restart Apache RedHat
    - Restart Apache Debian
```

```
handlers:
- name: Restart Apache Debian
  service:
    name: apache2
    state: restarted
  when: ansible_os_family == "Debian"

- name: Restart Apache RedHat
  service:
    name: httpd
    state: restarted
  when: ansible_os_family == "RedHat"
```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook6.yml
```

```

Quick connect... 3. 18.158.196.253 (AnsibleControlNode) +
ubuntu@ip-172-31-22-86:~/ansible/playbook-8$ ansible-playbook playbook6.yml
PLAY [Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.] ****
TASK [Gathering Facts] ****
[WARNING]: Platform linux on host AnsibleMabagedNode-4 is using the discovered Python
interpreter at /usr/bin/python3.7, but future installation of another Python
interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.14/reference_appendices/interpreter_discovery.html for more information.
ok: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [Check Linux distro] ****
ok: [AnsibleManagedNode-1] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-3] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-2] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleMabagedNode-4] => {
    "ansible_os_family": "RedHat"
}

TASK [Install Apache Web Server on Debian Family] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-1]

TASK [Start Apache and enable it during boot] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-2]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]

TASK [Install Apache Web Server on RedHat Family] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
ok: [AnsibleMabagedNode-4]

TASK [Start Apache and enable it during boot] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
ok: [AnsibleMabagedNode-4]

TASK [Generate index.html using template] ****
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-1]
changed: [AnsibleMabagedNode-4]
changed: [AnsibleManagedNode-2]

TASK [Copy dir MyWebSite to target server] ****
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-3]
ok: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-2]

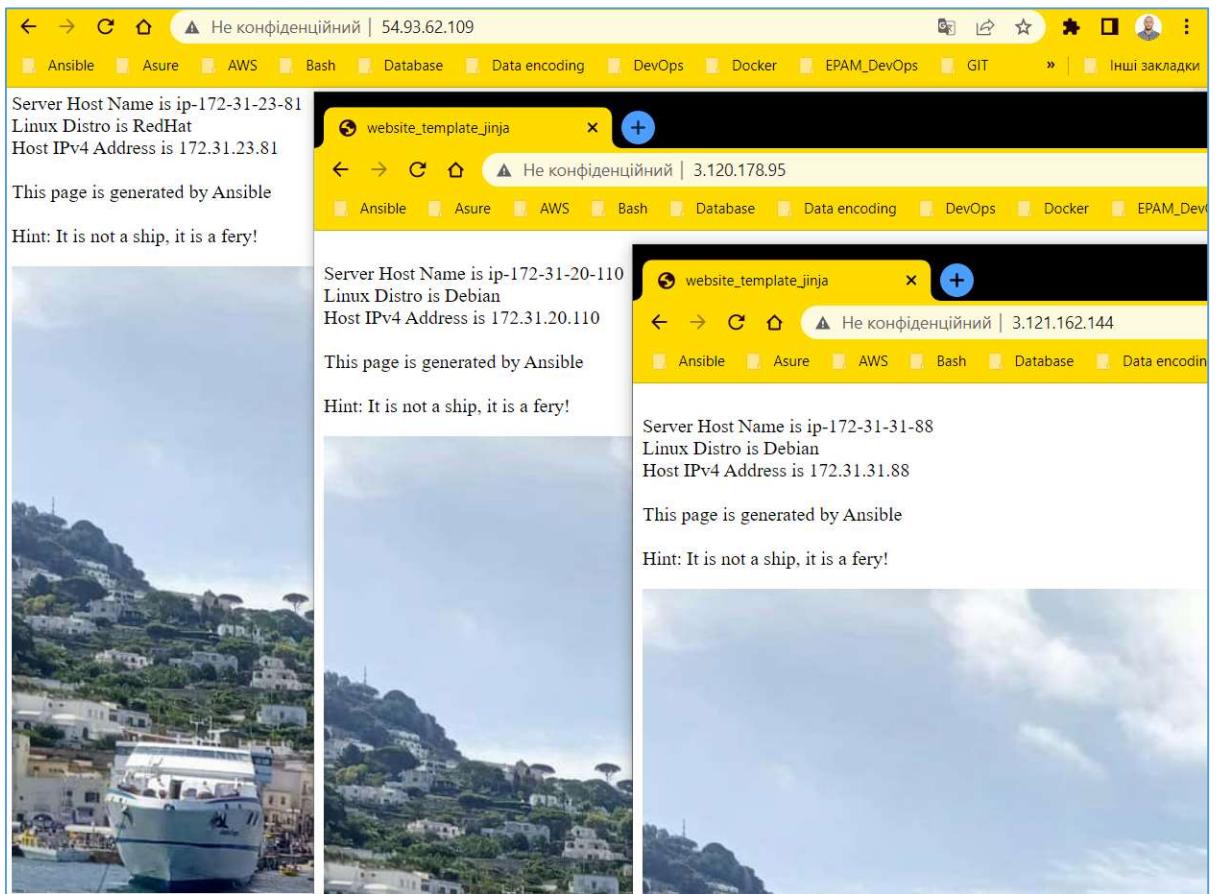
RUNNING HANDLER [Restart Apache Debian] ****
skipping: [AnsibleMabagedNode-4]
changed: [AnsibleManagedNode-1]
changed: [AnsibleManagedNode-3]
changed: [AnsibleManagedNode-2]

RUNNING HANDLER [Restart Apache RedHat] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
changed: [AnsibleMabagedNode-4]

PLAY RECAP ****
AnsibleMabagedNode-4      : ok=7      changed=2      unreachable=0      failed=0      skipped=3
                           rescued=0      ignored=0
AnsibleManagedNode-1      : ok=7      changed=2      unreachable=0      failed=0      skipped=3
                           rescued=0      ignored=0

```

- Checking the availability of the **index.html** page on all Ansible Managed Node's.



17. Ansible Roles.

Task: modifying playbook_loop2.yml used ansible roles.

- Creation **roles** directory and run **ansible** command

```
$ ansible-galaxy init deploy-apache
```

The screenshot shows a terminal window with the following session:

```
Quick connect... 3. 18.158.196.253 (AnsibleControlNode) +  
ubuntu@ip-172-31-22-86:~$ cd ansible/  
ubuntu@ip-172-31-22-86:~/ansible$ mkdir roles  
ubuntu@ip-172-31-22-86:~/ansible$ cd roles/  
ubuntu@ip-172-31-22-86:~/ansible/roles$ ansible-galaxy init deploy-apache  
- Role deploy-apache was created successfully  
ubuntu@ip-172-31-22-86:~/ansible/roles$ ll  
total 12  
drwxrwxr-x 3 ubuntu ubuntu 4096 Jan 10 15:21 ./  
drwxrwxr-x 16 ubuntu ubuntu 4096 Jan 10 15:20 ../  
drwxrwxr-x 10 ubuntu ubuntu 4096 Jan 10 15:21 deploy-apache/  
ubuntu@ip-172-31-22-86:~/ansible/roles$ tree  
└── deploy-apache  
    ├── README.md  
    ├── defaults  
    │   └── main.yml  
    ├── files  
    ├── handlers  
    │   └── main.yml  
    ├── meta  
    │   └── main.yml  
    ├── tasks  
    │   └── main.yml  
    ├── templates  
    ├── tests  
    │   └── inventory  
    │       └── test.yml  
    └── vars  
        └── main.yml
```

We have got the clean directory structure with the **ansible-galaxy** command. Each directory must contain a **main.yml** file, which contains the relevant content.

Where :

tasks – contains the main list of tasks to be executed by the role.

handlers – contains handlers, which may be used by this role or even anywhere outside this role.

defaults – default variables for the role.

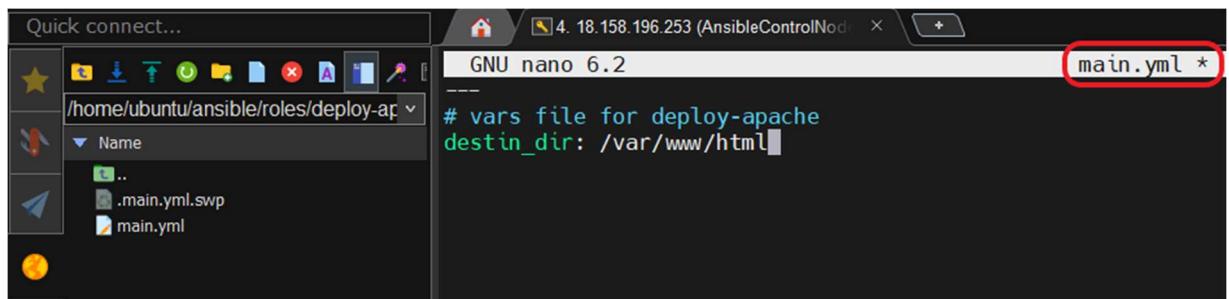
vars – other variables for the role. Var has the higher priority than defaults.

files – contains files required to transfer or deploy to the target machines via this role.

templates – contains templates which can be deployed via this role.

meta – defines some data / information about this role(author, dependency, versions, examples, etc).

- Writing variables to the `./roles/vars/main.yml` file

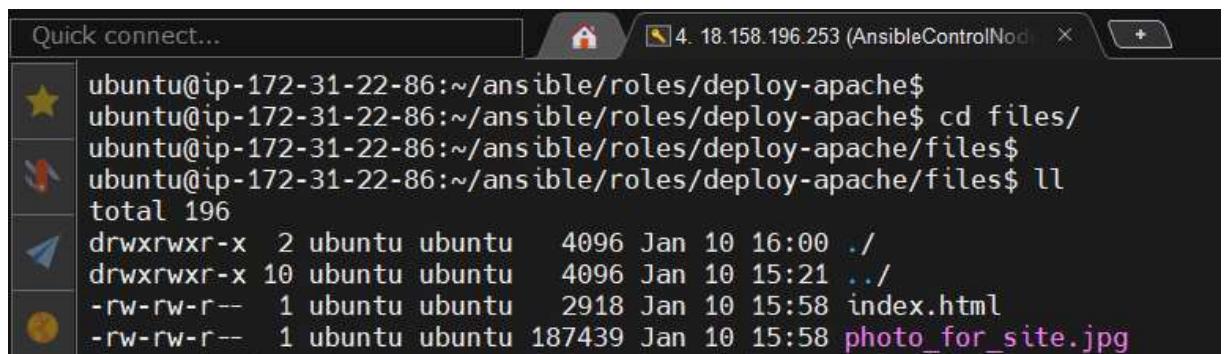


The screenshot shows a terminal window titled "GNU nano 6.2" with the file "main.yml" open. The file contains the following YAML code:

```
# vars file for deploy-apache
destin_dir: /var/www/html
```

The title bar of the terminal window has a red box around the word "main.yml". The left side of the window shows a file tree with a folder named "Name" containing ".main.yml.swp" and "main.yml".

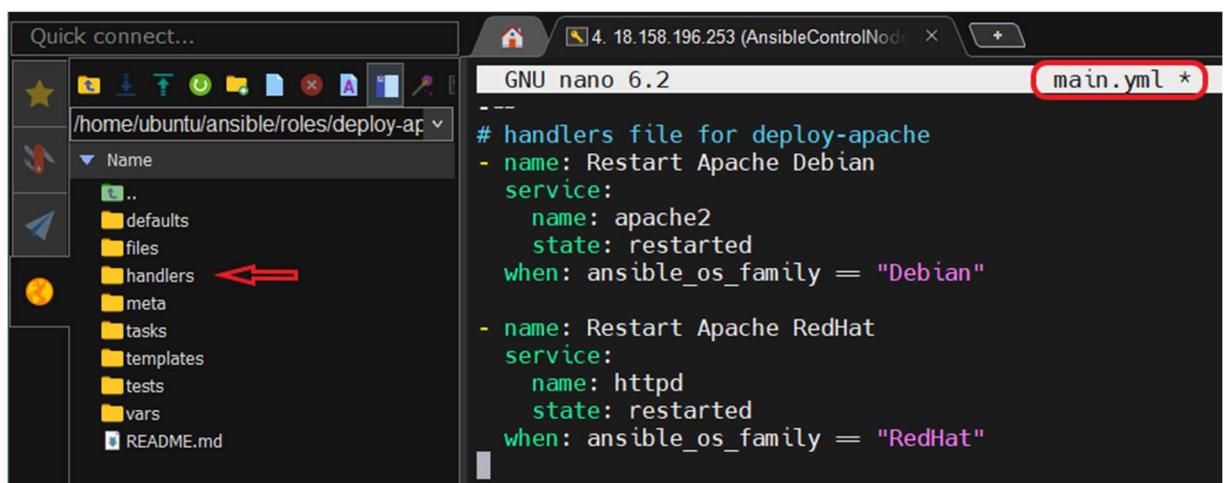
- Copy files that must be deployed to the `./roles/files/` directory



The screenshot shows a terminal window with the following command history and output:

```
ubuntu@ip-172-31-22-86:~/ansible/roles/deploy-apache$ cd files/
ubuntu@ip-172-31-22-86:~/ansible/roles/deploy-apache/files$ ll
total 196
drwxrwxr-x  2 ubuntu  ubuntu   4096 Jan 10 16:00 .
drwxrwxr-x 10 ubuntu  ubuntu   4096 Jan 10 15:21 ..
-rw-rw-r--  1 ubuntu  ubuntu  2918 Jan 10 15:58 index.html
-rw-rw-r--  1 ubuntu  ubuntu 187439 Jan 10 15:58 photo_for_site.jpg
```

- Writing handlers to the `./roles/handlers/main.yml` file



The screenshot shows a terminal window titled "GNU nano 6.2" with the file "main.yml" open. The file contains the following YAML code:

```
# handlers file for deploy-apache
- name: Restart Apache Debian
  service:
    name: apache2
    state: restarted
  when: ansible_os_family = "Debian"

- name: Restart Apache RedHat
  service:
    name: httpd
    state: restarted
  when: ansible_os_family = "RedHat"
```

The title bar of the terminal window has a red box around the word "main.yml". The left side of the window shows a file tree with a folder named "Name" containing "defaults", "files", "handlers" (with a red arrow pointing to it), "meta", "tasks", "templates", "tests", "vars", and "README.md".

- Writing tasks to the `.roles/tasks/main.yml` file

```

# tasks file for deploy-apache
- name: Check Linux distro
  debug:
    var: ansible_os_family

- block: # For Debian Family
  - name: Install Apache Web Server on Debian Family
    apt:
      name: apache2
      state: latest

  - name: Start Apache and enable it during boot
    service:
      name: apache2
      state: started
      enabled: yes

  when: ansible_os_family == "Debian"

- block: # For RedHat Family
  - name: Install Apache Web Server on RedHat Family
    yum:
      name: httpd
      state: latest

  - name: Start Apache and enable it during boot
    service:
      name: httpd
      state: started
      enabled: yes

  when: ansible_os_family == "RedHat"

- name: Copy dir "MyWebSite" to target server
  copy:
    src: "{{ source_dir }}/{{ item }}"
    dest: "{{ destin_dir }}"
    mode: 0555
  loop:
    - "index.html"
    - "photo_for_site.jpg"
  notify:
    - Restart Apache RedHat
    - Restart Apache Debian

```

- Writing **playbook7.yml**

```

Quick connect... 4. 18.158.196.253 (Ansible) 6. 54.93.62.109 (Ansible)
GNU nano 6.2
#-----Roles example-----#
---
- name: Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.
hosts: all
become: yes

vars:
  source_dir: ./MyWebSite

roles:
  - role: deploy-apache

```

- Directory structure

```

ubuntu@ip-172-31-22-86:~/ansible/playbook-9$ tree
.
├── MyWebSite
│   ├── index.html
│   └── photo_for_site.jpg
├── ansible.cfg
├── group_vars
│   ├── ami_cred
│   └── deb_cred
├── hosts.txt
└── playbook7.yml

roles
└── deploy-apache
    ├── README.md
    ├── defaults
    │   └── main.yml
    ├── files
    │   ├── index.html
    │   └── photo_for_site.jpg
    ├── handlers
    │   └── main.yml
    ├── meta
    │   └── main.yml
    ├── tasks
    │   └── main.yml
    ├── templates
    ├── tests
    │   ├── inventory
    │   └── test.yml
    └── vars
        └── main.yml

12 directories, 17 files

```

- Run **ansible-playbook** command.

```
$ ansible-playbook playbook7.yml
```

```

Quick connect... 4. 18.158.196.253 (Ansible) 6. 54.93.62.109 (Ansible)
ubuntu@ip-172-31-22-86:~/ansible/playbook-9$ ansible-playbook playbook7.yml
PLAY [Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.] ****
TASK [Gathering Facts] ****
[WARNING]: Platform linux on host AnsibleMabagedNode-4 is using the discovered
Python interpreter at /usr/bin/python3.7, but future installation of another
Python interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-
core/2.14/reference_appendices/interpreter_discovery.html for more information.
ok: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [deploy-apache : Check Linux distro] ****
ok: [AnsibleManagedNode-1] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-3] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleManagedNode-2] => {
    "ansible_os_family": "Debian"
}
ok: [AnsibleMabagedNode-4] => {
    "ansible_os_family": "RedHat"
}

TASK [deploy-apache : Install Apache Web Server on Debian Family] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [deploy-apache : Start Apache and enable it during boot] ****
skipping: [AnsibleMabagedNode-4]
ok: [AnsibleManagedNode-3]
ok: [AnsibleManagedNode-1]
ok: [AnsibleManagedNode-2]

TASK [deploy-apache : Install Apache Web Server on RedHat Family] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
ok: [AnsibleMabagedNode-4]

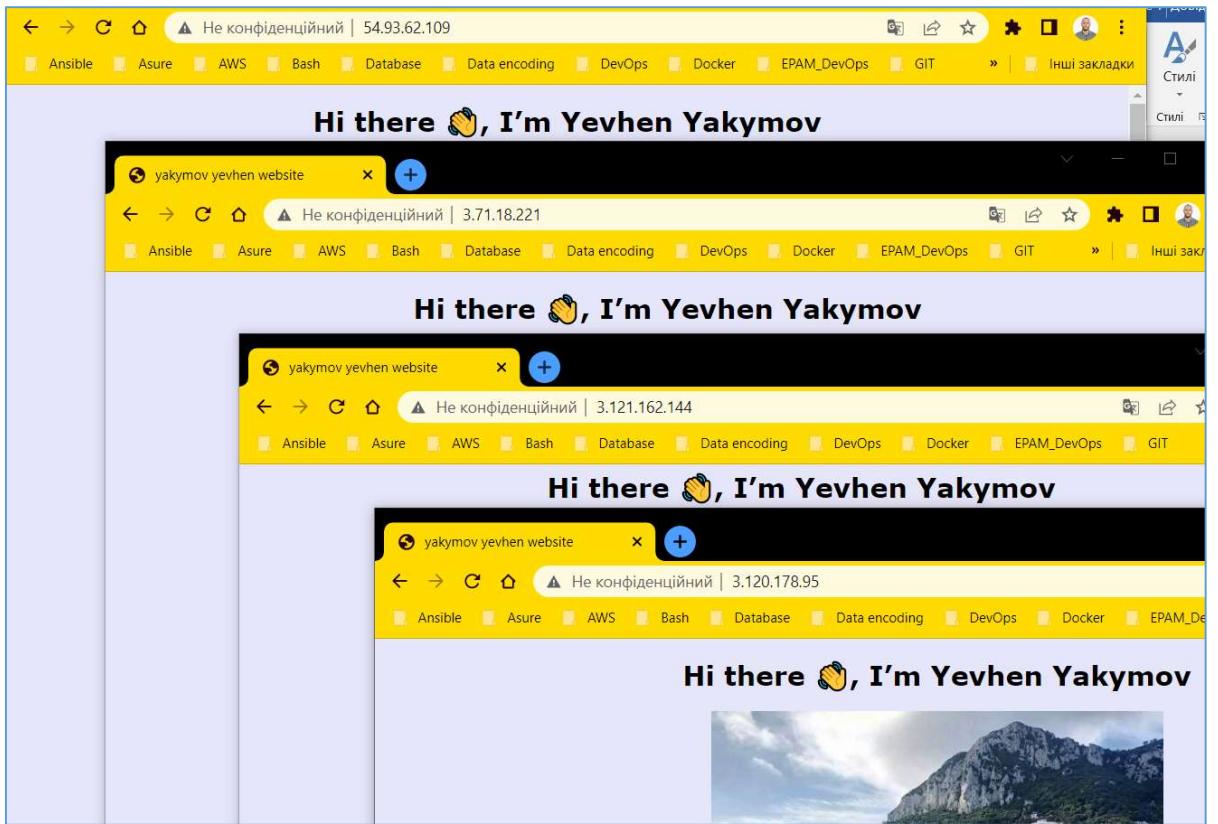
TASK [deploy-apache : Start Apache and enable it during boot] ****
skipping: [AnsibleManagedNode-1]
skipping: [AnsibleManagedNode-3]
skipping: [AnsibleManagedNode-2]
ok: [AnsibleMabagedNode-4]

TASK [deploy-apache : Copy dir "MyWebSite" to target server] ****
ok: [AnsibleManagedNode-3] => (item=index.html)
ok: [AnsibleManagedNode-1] => (item=index.html)
ok: [AnsibleMabagedNode-4] => (item=index.html)
ok: [AnsibleManagedNode-2] => (item=index.html)
ok: [AnsibleManagedNode-3] => (item=photo_for_site.jpg)
ok: [AnsibleManagedNode-1] => (item=photo_for_site.jpg)
ok: [AnsibleManagedNode-2] => (item=photo_for_site.jpg)
ok: [AnsibleMabagedNode-4] => (item=photo_for_site.jpg)

PLAY RECAP ****
AnsibleMabagedNode-4      : ok=5    changed=0    unreachable=0    failed=0    skip
ped=2  rescued=0  ignored=0
AnsibleManagedNode-1      : ok=5    changed=0    unreachable=0    failed=0    skip
ped=2  rescued=0  ignored=0
AnsibleManagedNode-2      : ok=5    changed=0    unreachable=0    failed=0    skip
ped=2  rescued=0  ignored=0
AnsibleManagedNode-3      : ok=5    changed=0    unreachable=0    failed=0    skip
ped=2  rescued=0  ignored=0

```

- Checking the availability of the **index.html** page on all Ansible Managed Node's.



18. Ansible Troubleshooting.

The most common strategies for debugging Ansible playbooks are using the modules given below:

- **Debug and Register**

These two are the modules available in Ansible. For debugging purpose, we need to use the two modules judiciously.

- **Verbosity**

With the Ansible command, one can provide the verbosity level. You can run the commands with verbosity level one (-v), two (-vv) or three (-vvv)

- **--start-at-task and step mode**

When you are testing new plays or debugging playbooks, you may need to run the same play multiple times. To make this more efficient, Ansible offers two alternative ways to execute a playbook: **start at task** and **step mode**.

To start executing your playbook at a particular task (usually the task that failed on the previous run), use **--start-at-task** option:

```
$ ansible-playbook playbook_name.yml --start-at-task "Task_name"
```

To known list tasks of the current playbook, use **--list-tasks** option:

```
$ ansible-playbook playbook_name.yml --list-tasks
```

```
ubuntu@ip-172-31-22-86:~/ansible/playbook-9$ ansible-playbook playbook7.yml --list-tasks
playbook: playbook7.yml

play #1 (all): Install Apache Web Server on Ubuntu and AMI Linux. Upload web page.    T
AGS: []
  tasks:
    deploy-apache : Check Linux distro          TAGS: []
    deploy-apache : Install Apache Web Server on Debian Family      TAGS: []
    deploy-apache : Start Apache and enable it during boot    TAGS: []
    deploy-apache : Install Apache Web Server on RedHat Family     TAGS: []
    deploy-apache : Start Apache and enable it during boot    TAGS: []
    deploy-apache : Copy dir "MyWebSite" to target server      TAGS: []
ubuntu@ip-172-31-22-86:~/ansible/playbook-9$ █
```

Very useful approach to use in playbooks and roles **tag**:

```
tag:
  - nginx
```

To execute a playbook interactively, use **--start-at-task** option:

```
$ ansible-playbook playbook_name.yml --step
```

ANSIBLE. Troubleshooting -- Example

```
- name: Print the gateway for each host when defined
ansible.builtin.debug:
  msg: System {{ inventory_hostname }} has gateway {{ ansible_default_ipv4.gateway }}
when: ansible_default_ipv4.gateway is defined

- name: Get uptime information
ansible.builtin.shell: /usr/bin/uptime
register: result

- name: Print return information from the previous task
ansible.builtin.debug:
  var: result
  verbosity: 2

- name: Display all variables/facts known for a host
ansible.builtin.debug:
  var: hostvars[inventory_hostname]
  verbosity: 4

- name: Prints two lines of messages, but only if there is an environment value set
ansible.builtin.debug:
  msg:
    - "Provisioning based on YOUR_KEY which is: {{ lookup('env', 'YOUR_KEY') }}"
    - "These servers were built using the password of '{{ password_used }}'. Please retain this for later use."
```

ANSIBLE. Vault

Once you have a strategy for managing and storing vault passwords, you can start encrypting content. You can encrypt two types of content with Ansible Vault: variables and files. Encrypted content always includes the `!vault` tag, which tells Ansible and YAML that the content needs to be decrypted, and a `|` character, which allows multi-line strings. Encrypted content created with `--vault-id` also contains the vault ID label.

This table shows the main differences between encrypted variables and encrypted files.

	Encrypted variables	Encrypted files
How much is encrypted?	Variables within a plaintext file	The entire file
When is it decrypted?	On demand, only when needed	Whenever loaded or referenced
What can be encrypted?	Only variables	Any structured data file

19. Ansible. Vault.

Encrypting content with Ansible Vault:

Ansible Vault encrypts variables and files so you can protect sensitive content such as passwords or keys rather than leaving it visible as plaintext in playbooks or roles.

To use Ansible Vault you need one or more passwords to encrypt and decrypt content. If you store your vault passwords in a third-party tool such as a secret manager, you need a script to access them.

Use the password with the **ansible-vault** command-line tool to create and view encrypted variables, create encrypted files, encrypt existing files, or edit, re-key, or decrypt files. You can then place encrypted content under source control and share it more safely.

Managing vault passwords:

Managing your encrypted content is easier if you develop a strategy for managing your vault passwords. A vault password can be any string you choose. There is no special command to create a vault password. However, you need to keep track of your vault passwords. Each time you encrypt a variable or file with **Ansible Vault**, you must provide a password. When you use an encrypted variable or file in a command or playbook, you must provide the same password that was used to encrypt it. To develop a strategy for managing vault passwords, start with two questions:

Do you want to encrypt all your content with the same password, or use different passwords for different needs?

Where do you want to store your password or passwords?

Choosing between a single password and multiple passwords:

If you have a small team or few sensitive values, you can use a single password for everything you encrypt with Ansible Vault. Store your vault password securely in a file or a secret manager as described below.

If you have a larger team or many sensitive values, you can use multiple passwords. For example, you can use different passwords for *different users* or *different levels* of access. Depending on your needs, you might want a different password for each encrypted *file*, for each *directory*, or for each *environment*.

For example, you might have a playbook that includes two vars files, one for the dev environment and one for the production environment, encrypted with two different passwords. When you run the playbook, select the correct vault password for the environment you are targeting, using a vault ID.

Managing multiple passwords with vault ID's:

If you use multiple vault passwords, you can differentiate one password from another with vault IDs. You use the vault ID in three ways:

- Pass it with **--vault-id** to the ansible-vault command when you create encrypted content
- Include it wherever you store the password for that vault ID (see Storing and accessing vault passwords)
- Pass it with **--vault-id** to the ansible-playbook command when you run a playbook that uses content you encrypted with that vault ID