**1.Ansible Introduction(puppet)**

**What is Ansible?**

**Ansible** is an open-source IT automation tool. It helps automate tasks like:

* Installing software
* Configuring servers
* Managing networks
* Deploying applications

Ansible uses simple YAML files called **Playbooks** to describe what should be done. It is **agentless**, meaning you only need SSH access to the remote machines—no need to install extra software on them.

**What is Puppet?**

**Puppet** is another popular configuration management tool. It:

* Uses its own domain-specific language (DSL), similar to Ruby
* Usually follows a **client-server** model, with a central Puppet master
* Uses a **pull** model—clients pull configurations from the server

**Ansible vs Puppet (Comparison):**

|  |  |  |
| --- | --- | --- |
| Feature | Ansible | Puppet |
| Language | YAML (simple and readable) | DSL (Ruby-like, more complex) |
| Setup | Agentless (uses SSH) | Agent-based (requires installation) |
| Architecture | Push-based | Pull-based |
| Learning Curve | Easier | Steeper |
| Best For | Quick automation and provisioning | Large-scale infrastructure management |

**Why choose Ansible?**

* Easy to learn and use
* No need to install agents on target machines
* Works well for quick setup and automation
* Flexible and suitable for various use cases, especially for small to mid-sized environments

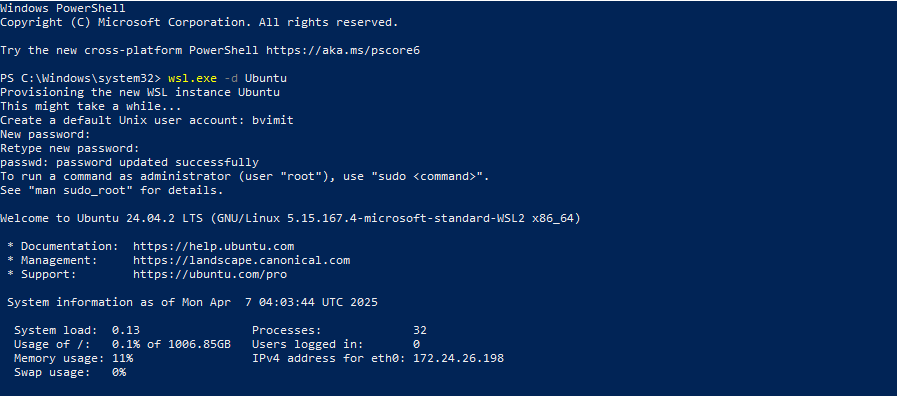
**2. Ad-hoc Ansible(puppet) commands**

**Step 1: Setting Up Ubuntu and Ansible**

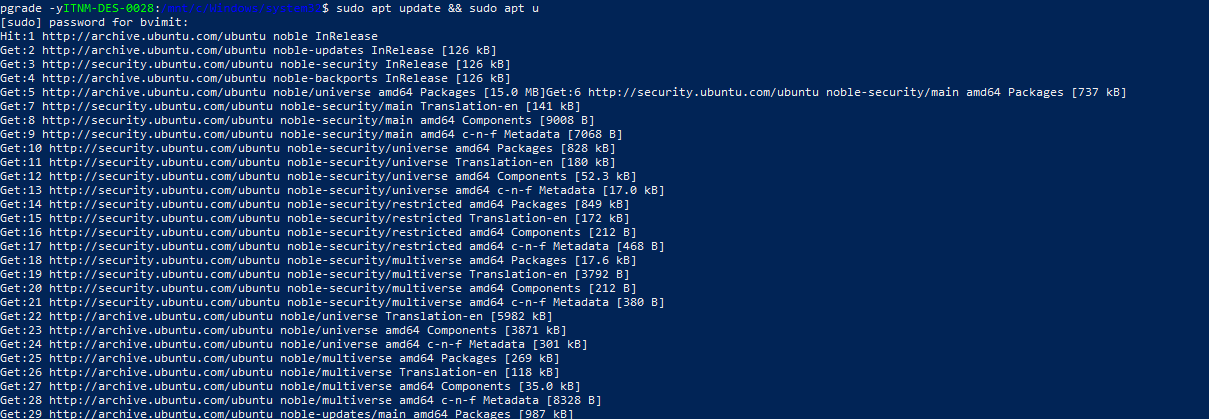
**Install Ubuntu**

**wsl --install –d Ubuntu:** This installs **Ubuntu Linux** on your Windows system using WSL.

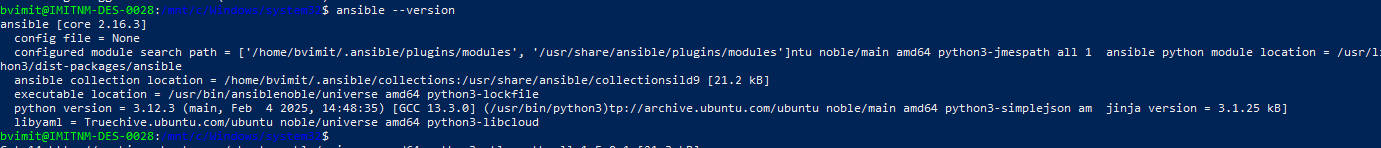
**wsl.exe –d Ubuntu:** This opens Ubuntu in the terminal.



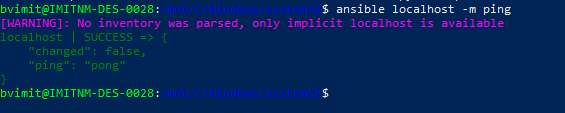
**sudo apt update && sudo apt u:** This updates the list of available software and upgrades everything to the latest version.



**ansible –version:** This checks if Ansible is installed and shows the version.

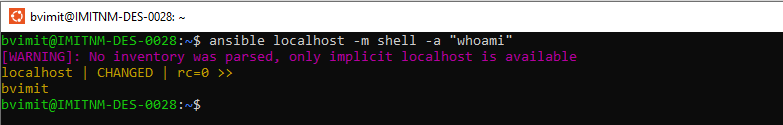


**ansible localhost -m ping:** Sends a test "ping" to your own system using Ansible to check if it works.

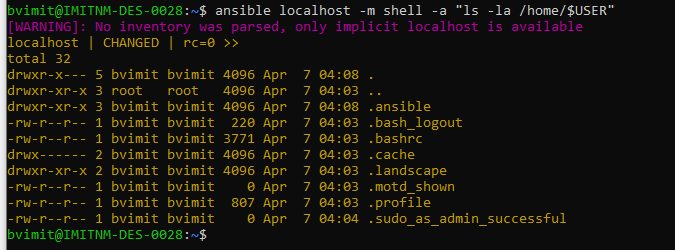


**Run following command in Ubuntu**

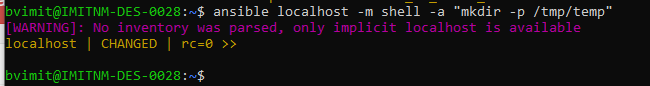
**ansible localhost -m shell -a "whoami":** Runs the whoami command to see the current user using the shell module.



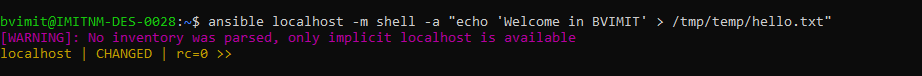
**ansible localhost -m shell -a "ls -la /home/$USER":** Lists all files and folders in your home directory (in detail).



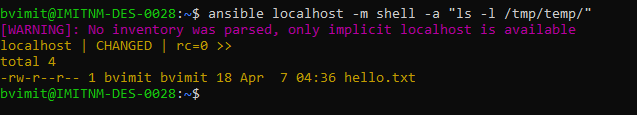
**ansible localhost -m shell -a "mkdir -p /tmp/temp":** Makes a folder called temp inside /tmp. The -p allows it to create parent folders if needed.



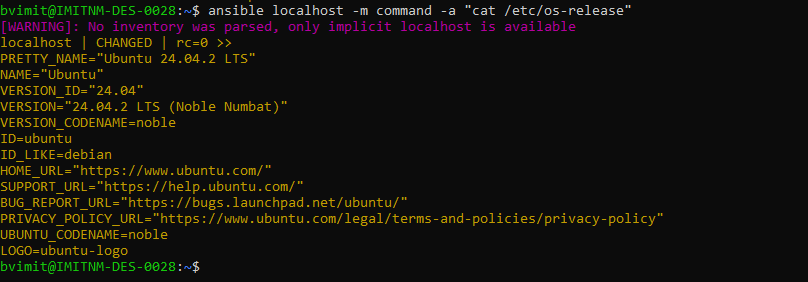
**ansible localhost -m shell -a "echo 'Welcome in BVIMIT' > /tmp/temp/hello.txt":** Creates a file called hello.txt in the temp folder and adds the text “Welcome in BVIMIT”.

****

**ansible localhost -m shell -a "ls -l /tmp/temp/":** Lists all files in the /tmp/temp directory.

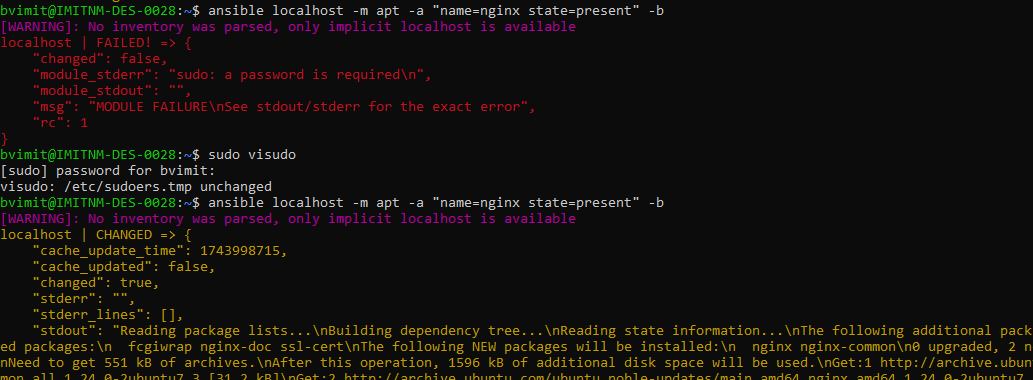
****

**ansible localhost -m command -a "cat /etc/os-release" :** Shows information about the Ubuntu version you are using.



**Here we try to install the ngix package**

**ansible localhost -m apt -a "name=nginx state=present" -b:** Installs **nginx**. The -b means **"become root"**, which gives admin access (needs password).

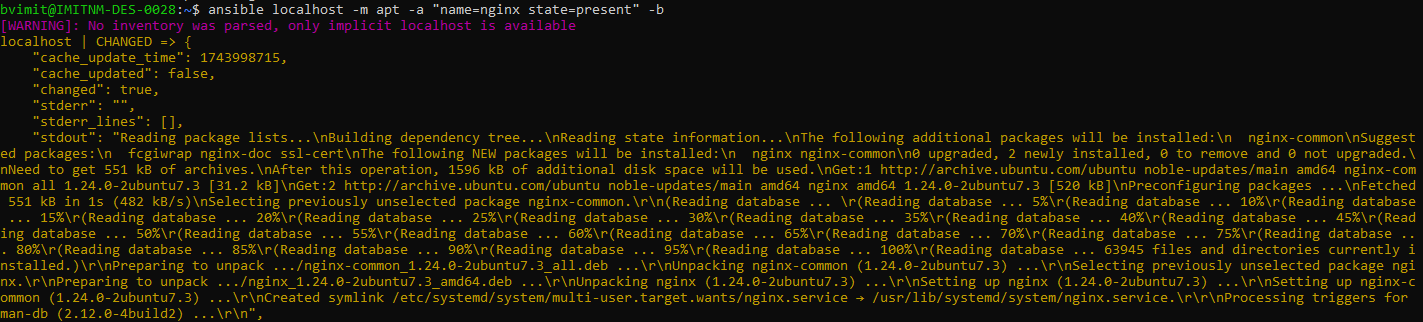


**This error because it’s require password so we enter following command**

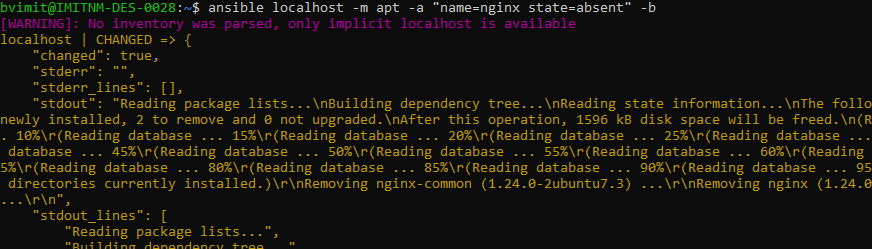
If you get a permission error, you may have to enter your password.

****

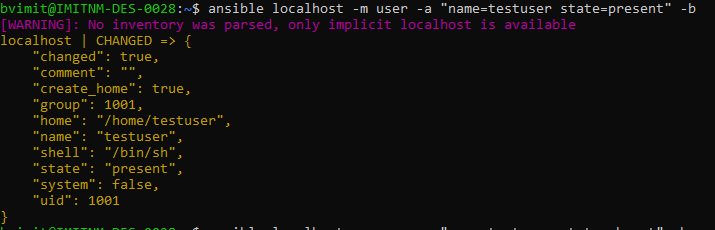
**ansible localhost -m apt -a "name=nginx state=present" -b**

****

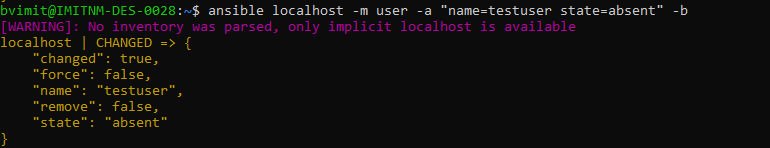
**ansible localhost -m apt -a "name=nginx state=absent" -b :** Uninstalls nginx.

****

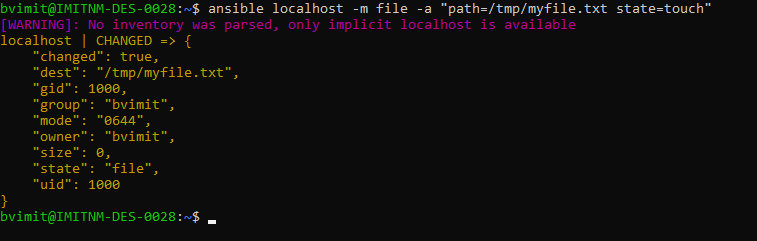
**ansible localhost -m user -a "name=testuser state=present" -b :** Creates a user named testuser.

****

**ansible localhost -m user -a "name=testuser state=absent" -b:** Deletes the user named testuser.

****

**ansible localhost -m file -a "path=/tmp/myfile.txt state=touch":** Creates an empty file named myfile.txt in the /tmp folder.

****

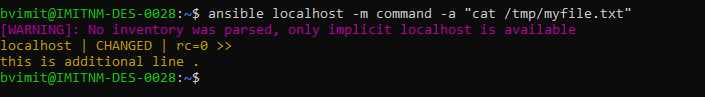
**ansible localhost -m shell -a "echo 'this is first line .'>/tmp/myfile.txt":** Adds the text "this is first line ." to the file (overwrites if file already exists).

****

**ansible localhost -m shell -a "echo 'this is additional line .'>/tmp/myfile.txt": A**dds another line to the file (appends without deleting existing content).

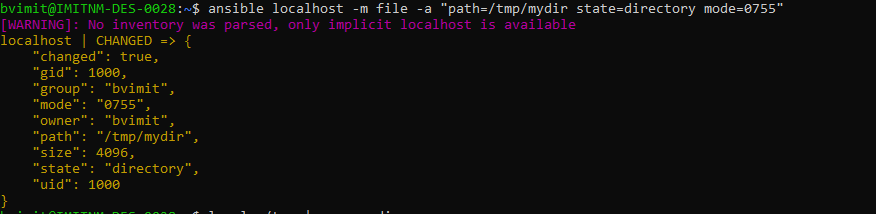
****

**ansible localhost -m command -a "cat /tmp/myfile.txt" :** Shows the content of the file.

****

**How to create,display,delete directory**

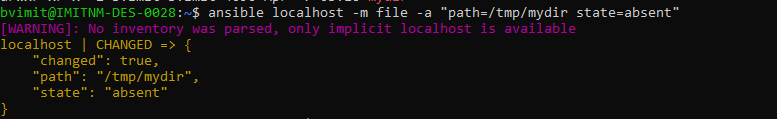
**ansible localhost -m file -a "path=/tmp/mydir state=directory mode=0755":** Creates a directory called mydir with permission 0755 (read/write/execute for owner, read/execute for others).

****

**ls -la /tmp | grep mydir:** Manually checks if the mydir folder exists inside /tmp.

****

**ansible localhost -m file -a "path=/tmp/mydir state=absent":** Deletes the folder mydir.



**Now we try to display the file, it’s display null because we delete that directory**



**3. Using Ansible(puppet) playbooks**

Playbook

A **Playbook** is a YAML file that defines a series of actions to be executed on managed nodes. It contains one or more "plays" that map groups of hosts to roles.

**Example**

---

* name: Update web servers hosts: webservers remote\_user: root

tasks:

* + name: Ensure apache is at the latest version ansible.builtin.yum:

name: httpd state: latest

* + name: Write the apache config file

ansible.builtin.template: src: /srv/httpd.j2 dest: /etc/httpd.conf

* name: Update db servers hosts: databases remote\_user: root

tasks:

* + name: Ensure postgresql is at the latest version ansible.builtin.yum:

name: postgresql state: latest

* + name: Ensure that postgresql is started

ansible.builtin.service: name: postgresql

state: started

Play

A Play is a single, complete execution unit within a playbook. It specifies which hosts to target and what tasks to execute on those hosts. Plays are used to group related tasks and execute them in a specific order.

- name: Install and configure Nginx

hosts: webservers

tasks:

- name: Install Nginx apt:

name: nginx

state: present

Modules

Modules are the building blocks of Ansible tasks. They are small programs that perform a specific action on a managed node, such as installing a package, copying a file, or

managing services. Example

The apt module used in a task to install a package:

- name: Install Nginx apt:

name: nginx

state: present

Tasks

Tasks are individual actions within a play that use modules to perform operations on

managed nodes. Each task is executed in order and can include conditionals, loops, and handlers.

- name: Install Nginx

apt: name: nginx state: present

-

name: Start Nginx

service service:

name: nginx state: started

Collections

Collections are a distribution format for Ansible content. They bundle together multiple roles, modules, plugins, and other Ansible artifacts. Collections make it easier to share and reuse Ansible content. Example

A collection structure might look like this:

├── plugins/

│ └── modules/

│ └── my\_module.py

└── README.md

└── tasks/

└── main.yml

│

│

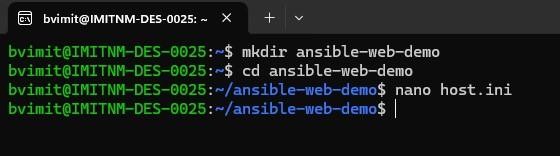
my\_collection/ ├── roles/

│ └── my\_role/

**1. Directory Setup**

Create a folder to work in:

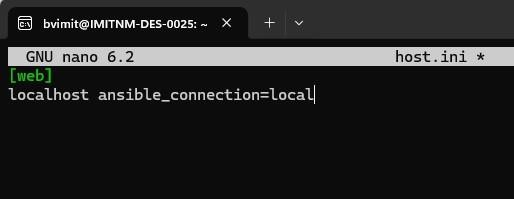
mkdir ansible-web-demo cd ansible-web-demo



**2. Create the Inventory File (hosts.ini)**

Create a file named hosts.ini:

ini CopyEdit [web]

localhost ansible\_connection=local

If you're testing on your own machine, use localhost.

**3. Create a Simple Playbook (hello.yml)**



yaml CopyEdit

---

* name: Serve Hello World HTML Page hosts: web become: yes

tasks:

* name: Install Apache Web Server apt:

name: apache2 state: present update\_cache: yes

* name: Create custom index.html copy:

dest: /var/www/html/index.html content: |

<html>

<head><title>Hello</title></head>

<body>

<h1>Hello from Ansible 👋</h1>

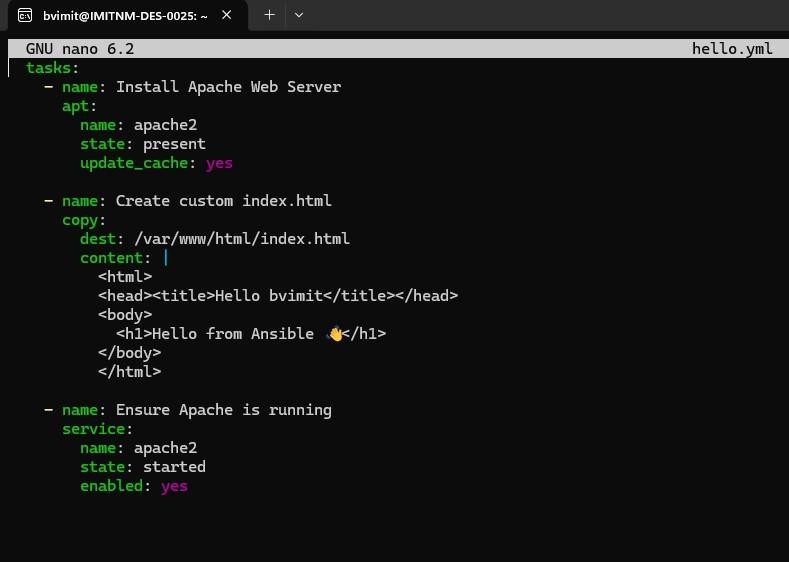
</body>

</html>

* name: Ensure Apache is running service:

name: apache2 state: started enabled: yes

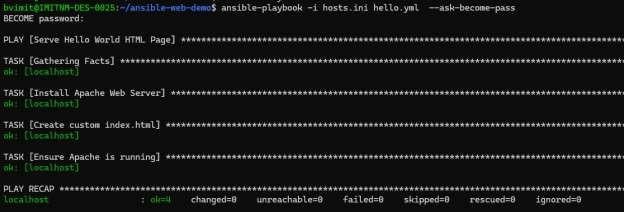
This installs Apache, creates /var/www/html/index.html, and starts the service.



**Save and Exit**

* + Press Ctrl + O, then Enter to save
  + Press Ctrl + X to exit

**Run the Playbook**



**Check the Result**

Open a browser and go to:

