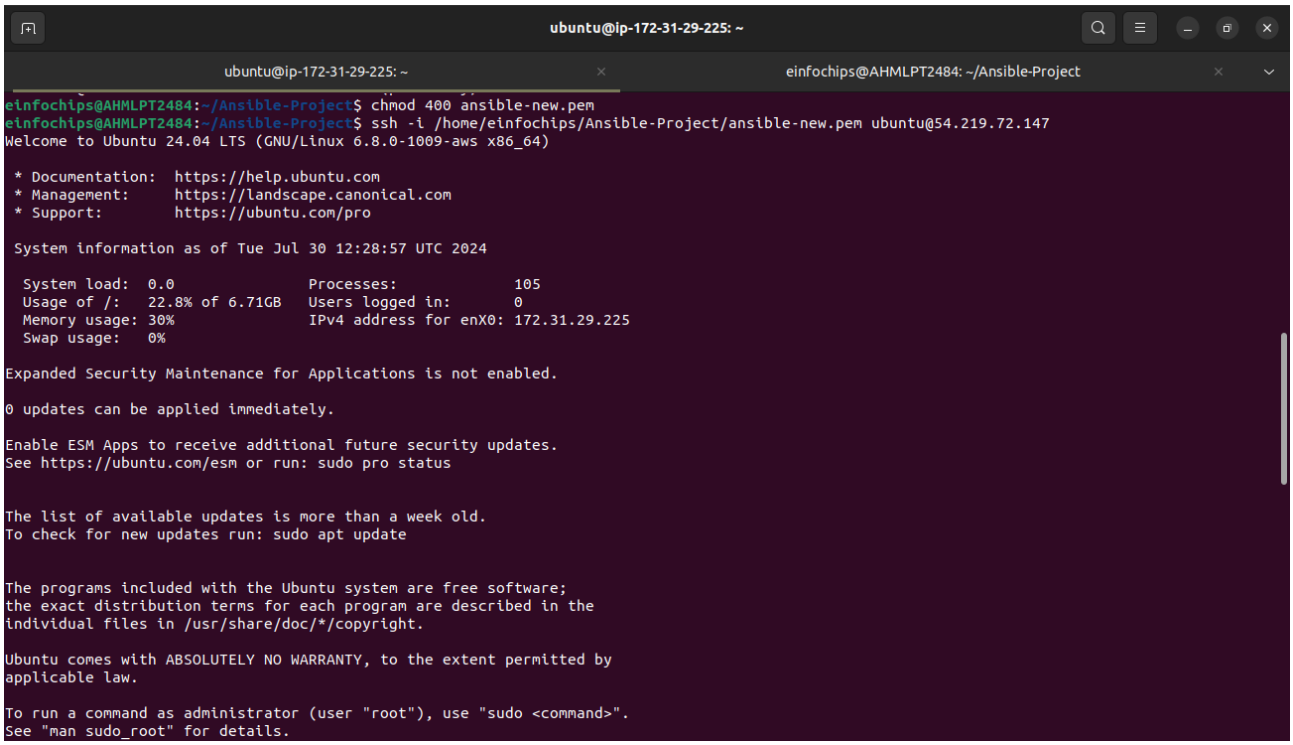
Project 1: Deploying Ansible

Problem Statement: You are tasked with deploying Ansible in a multi-node environment consisting of multiple Linux servers. The goal is to set up Ansible on a control node and configure it to manage several managed nodes. This setup will be used for automating system administration tasks across the network.

Deliverables:

1. **Control Node Setup:**
   - Install Ansible on the control node.
   - Configure SSH key-based authentication between the control node and managed nodes.
2. **Managed Nodes Configuration:**
   - Ensure all managed nodes are properly configured to be controlled by Ansible.
   - Verify connectivity and proper setup between the control node and managed nodes.
3. **Documentation:**
   - Detailed installation and configuration steps.
   - Troubleshooting guide for common issues encountered during deployment.



```
einfochips@AHMLPT2484: ~/Ansible-Project                einfochips@AHMLPT2484: ~/An
Get:21 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [40.7 kB]
Get:22 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [416 B]
Get:23 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.1 kB]
Get:24 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:25 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:26 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:27 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:28 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:29 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.3 kB]
Get:30 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.5 kB]
Get:31 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:32 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [988 B]
Get:33 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:34 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:35 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:36 http://us-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:37 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [256 kB]
Get:38 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [60.5 kB]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [2680 B]
Get:40 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [239 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [105 kB]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [8632 B]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [4564 B]
Get:44 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [208 kB]
Get:45 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [40.7 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [420 B]
Get:47 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [10.6 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2808 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 28.1 MB in 5s (5165 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-29-225:~$ python3 --version
Python 3.12.3
ubuntu@ip-172-31-29-225:~$ client_loop: send disconnect: Broken pipe
einfochips@AHMLPT2484:~/Ansible-Project$
```

```
einfochips@AHMLPT2484:~/Ansible-Project$ sudo nano /etc/ansible//hosts
[sudo] password for einfochips:
einfochips@AHMLPT2484:~/Ansible-Project$ ansible all -m ping -i /etc/ansible/hosts
worker01 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
einfochips@AHMLPT2484:~/Ansible-Project$
```

Project 2: Ad-Hoc Ansible Commands

Problem Statement: Your organization needs to perform frequent, one-off administrative tasks across a fleet of servers. These tasks include checking disk usage, restarting services, and updating packages. You are required to use Ansible ad-hoc commands to accomplish these tasks efficiently.

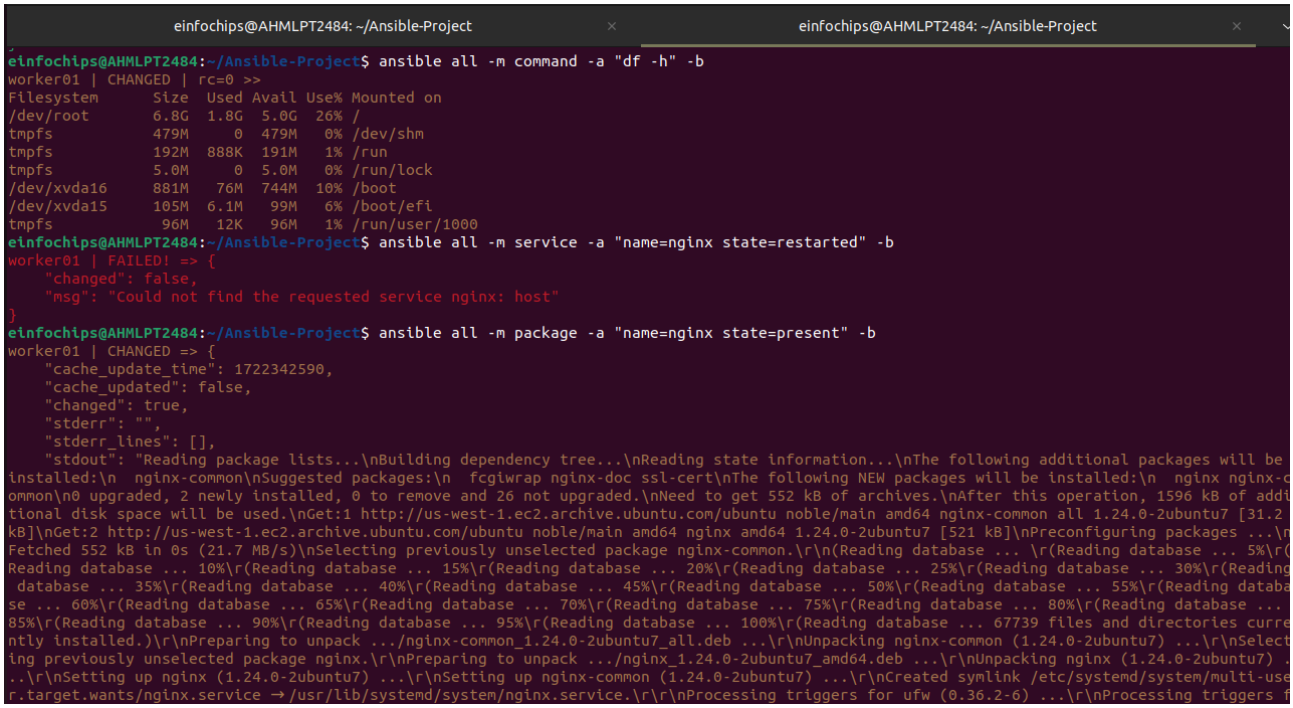
Deliverables:

1. **Task Execution:**
   - Execute commands to check disk usage across all managed nodes.
   - Restart a specific service on all managed nodes.
   - Update all packages on a subset of managed nodes.

2. **Command Scripts:**

- Create a script or documentation for each task, detailing the ad-hoc command used and its output.

3. **Documentation:**
   - Provide a comprehensive guide on using Ansible ad-hoc commands.
   - Include examples of common administrative tasks that can be performed with ad-hoc commands.



Project 3: Working with Ansible Inventories

Problem Statement: You need to manage a dynamic and diverse set of servers, which requires an organized and flexible inventory system. The project involves creating static and dynamic inventories in Ansible to categorize servers based on different attributes such as environment (development, staging, production) and roles (web servers, database servers).

Deliverables:

1. **Static Inventory:**
   - Create a static inventory file with different groups for various environments and roles.
   - Verify that the inventory is correctly structured and accessible by Ansible.

2. **Dynamic Inventory:**
   - Implement a dynamic inventory script or use a dynamic inventory plugin.
   - Configure the dynamic inventory to categorize servers automatically based on predefined criteria.

3. **Documentation:**
    - Instructions for setting up and using static and dynamic inventories.
    - Examples of playbooks utilizing both types of inventories.

```
einfochips@AHMLPT2484: ~/Ansible-Project
        "No containers need to be restarted.",
        "",
        "User sessions running outdated binaries:",
        " ubuntu @ session #18: sshd[1203]",
        " ubuntu @ session #23: sh[2465], sshd[1973]"
    ]
}
einfochips@AHMLPT2484:~/Ansible-Project$ nano inventory.ini
einfochips@AHMLPT2484:~/Ansible-Project$ sudo nano /etc/ansible/hosts
[sudo] password for einfochips:
einfochips@AHMLPT2484:~/Ansible-Project$ nano inventory.ini
einfochips@AHMLPT2484:~/Ansible-Project$ ansible all -m ping -i inventory.ini
worker01 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
einfochips@AHMLPT2484:~/Ansible-Project$ nano install_nginx.yaml
einfochips@AHMLPT2484:~/Ansible-Project$ nano config_nginx.yaml
einfochips@AHMLPT2484:~/Ansible-Project$ cat install_nginx.yaml
---
hosts: worker01
become: yes
tasks:
  -name: Install nginx
  package:
    name: nginx
    state: present
einfochips@AHMLPT2484:~/Ansible-Project$ nano config_nginx.yaml
einfochips@AHMLPT2484:~/Ansible-Project$ cat install_nginx.yaml
---
hosts: worker01
become: yes
tasks:
  -name: Install nginx
  package:
    name: nginx
    state: present
```

Project 4: Ansible Playbooks: The Basics

Problem Statement: Your team needs to automate repetitive tasks such as installing packages, configuring services, and managing files on multiple servers. The project involves writing basic Ansible playbooks to automate these tasks, ensuring consistency and efficiency in the operations.

Deliverables:

1. **Playbook Creation:**
    - Write a playbook to install a specific package on all managed nodes.

    - Create a playbook to configure a service with specific parameters.

- Develop a playbook to manage files, such as creating, deleting, and modifying files on managed nodes.

2. **Testing and Verification:**
   - Test the playbooks to ensure they run successfully and perform the intended tasks.
   - Validate the changes made by the playbooks on the managed nodes.
3. **Documentation:**
   - Detailed explanation of each playbook, including the tasks and modules used.
   - Guide on how to run the playbooks and interpret the results.

a) install_nginx.yaml file

```
  GNU nano 6.2
---
- hosts: worker01
  become: yes
  tasks:
    - name: Install nginx
      apt:
        name: nginx
        state: present
```

2) config_nginx.yaml

```
  GNU nano 6.2
---
- hosts: worker01
  become: yes
  tasks:
    - name: Ensure nginx is enabled and started
      service:
        name: nginx
        state: started
        enabled: yes
```

2) manage_files.yaml

```
GNU nano 6.2
---
- hosts: worker01
  become: yes
  tasks:
    - name: Create a file with specific content
      copy:
        dest: /tmp/example_file.txt
        content: |
          This is a test file.
        owner: root
        group: root
        mode: '0644'

    - name: Modify the file content
      lineinfile:
        path: /tmp/example_file.txt
        line: 'Additional line added to the file.'
        create: yes

    - name: Delete the file
      file:
        path: /tmp/example_file.txt
        state: absent
```

```
-hosts: worker01
  become: yes
        ^ here
einfochips@AHMLPT2484:~/Ansible-Project$ nano install_nginx.yaml
einfochips@AHMLPT2484:~/Ansible-Project$ ansible-playbook install_nginx.yaml -i inventory.ini

PLAY [worker01] ***************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************
ok: [worker01]

TASK [Install nginx] **********************************************************************************************
ok: [worker01]

PLAY RECAP ********************************************************************************************************
worker01                   : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

einfochips@AHMLPT2484:~/Ansible-Project$
```

```
ubuntu@ip-172-31-29-225: ~        einfochips@AHMLPT2484: ~/Ansible-Project

einfochips@AHMLPT2484:~/Ansible-Project$ ansible-playbook config_nginx.yaml -i inventory.ini

PLAY [worker01] ***************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************
ok: [worker01]

TASK [Ensure nginx is enabled and started] ************************************************************************
ok: [worker01]

PLAY RECAP ********************************************************************************************************
worker01                   : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

einfochips@AHMLPT2484:~/Ansible-Project$ ls
ansible-new.pem  config_nginx.yaml  install_nginx.yaml  inventory.ini  manage_files.yaml  README.md
einfochips@AHMLPT2484:~/Ansible-Project$ nano manage_files.yaml
einfochips@AHMLPT2484:~/Ansible-Project$ ansible-playbook manage_files.yaml -i inventory.ini

PLAY [worker01] ***************************************************************************************************

TASK [Gathering Facts] ********************************************************************************************
ok: [worker01]

TASK [Create a file with specific content] ************************************************************************
changed: [worker01]

TASK [Modify the file content] ************************************************************************************
changed: [worker01]

TASK [Delete the file] ********************************************************************************************
changed: [worker01]

PLAY RECAP ********************************************************************************************************
worker01                   : ok=4    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

einfochips@AHMLPT2484:~/Ansible-Project$
```

Project 5: Ansible Playbooks - Error Handling

Problem Statement: In a complex IT environment, tasks automated by Ansible playbooks may encounter errors due to various reasons such as incorrect configurations, unavailable resources, or network issues. The project focuses on implementing error handling in Ansible playbooks to ensure resilience and proper reporting of issues.

Deliverables:

1. **Playbook with Error Handling:**
   - Write a playbook that includes tasks likely to fail, such as starting a non-existent service or accessing a non-existent file.
   - Implement error handling strategies using modules like `block`, `rescue`, and `always`.

```
  GNU nano 6.2                                                    error_handling.yaml
---
- name: Playbook with Error Handling
  hosts: worker01
  become: yes
  tasks:
    - name: Example task that might fail
      block:
        - name: Try to start a non-existent service
          service:
            name: non_existent_service
            state: started

        - name: Try to access a non-existent file
          command: cat /path/to/non_existent_file.txt
      rescue:
        - name: Log the error message
          debug:
            msg: "An error occurred during the execution of a task. Please check the logs."
      always:
        - name: Always executed task
          debug:
            msg: "The block completed, regardless of success or failure of the tasks."
```

```
einfochips@AHMLPT2484:~/Ansible-Project$ nano error_handling.yaml
einfochips@AHMLPT2484:~/Ansible-Project$ ansible-playbook error_handling.yaml -i inventory.ini

PLAY [Playbook with Error Handling] ********************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
ok: [worker01]

TASK [Try to start a non-existent service] *************************************************************************************
fatal: [worker01]: FAILED! => {"changed": false, "msg": "Could not find the requested service non_existent_service: host"}

TASK [Log the error message] ***************************************************************************************************
ok: [worker01] => {
    "msg": "An error occurred during the execution of a task. Please check the logs."
}

TASK [Always executed task] ****************************************************************************************************
ok: [worker01] => {
    "msg": "The block completed, regardless of success or failure of the tasks."
}

PLAY RECAP *********************************************************************************************************************
worker01                   : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=1    ignored=0

einfochips@AHMLPT2484:~/Ansible-Project$
```

2. **Logging and Notifications:**

    - Configure the playbook to log errors and notify administrators of any issues encountered during execution.
    - Use Ansible modules to send notifications via email or other communication channels.
3. **Documentation:**
    - Comprehensive guide on error handling in Ansible playbooks, including examples of common scenarios and solutions.
    - Detailed instructions on setting up logging and notifications for error handling.