DEVOPS ADVANCE CONFIGURATION MANAGEMENT – INT-33

CONFIGURING PHOTO-GALLERY-APPLICATION USING ANSIBLE

submitted in partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

In (Computer science and Engineering)

Submitted to Mrs. Chavi Ralhan Mam

Lovely Professional University, Phagwara, Punjab.

Submitted By

Name	**Mugle Sruthi**
Reg No	**12109334**
Roll No	**40**
Section	**K0309**

TABLE OF CONTENT:

CONFIGURING PHOTO-GALLERY-APPLICATION USING ANSIBLE

photo_gallery/

├── app.py

├── static/

│ └── uploads/

└── templates/

└── index.html

Project Overview

The Photo Gallery application is designed to provide a user-friendly interface for uploading and displaying photos. It allows users to manage their personal photo collections efficiently, offering essential features such as photo upload, gallery display, and full-screen viewing. This report outlines the main features, implementation details, and key considerations in developing the application.

Objectives

The main objectives of the Photo Gallery application include:

1. Enabling users to upload photos seamlessly.
2. Displaying uploaded photos in a structured gallery format.
3. Allowing users to view individual photos in full-screen mode upon clicking.

Features

1. Upload Photos

- **Functionality**: Users can upload photos by selecting files from their device. The upload feature supports image file types such as JPEG, PNG, and GIF.
- **Implementation**:
    - Form-based input for file selection and submission.
    - Server-side handling for file validation and storage.
    - Limit checks on file size and type to maintain performance and security.

2. Display Photos in a Gallery Format

- **Functionality**: Once uploaded, photos are displayed in a grid or tile layout, creating a visually appealing gallery. The display can be customized, offering various viewing arrangements (e.g., columns, sizes).
- **Implementation**:
    - Utilization of HTML/CSS for grid layout design.
    - Dynamic rendering of photo entries using a front-end framework or templating engine.
    - Pagination or scrolling to manage large collections.

3. View Photos in Full-Screen Mode

- **Functionality**: Users can click on any photo in the gallery to view it in full-screen mode for a closer and more immersive experience.
- **Implementation**:
    - Modal or lightbox functionality for full-screen display.
    - Navigation options (e.g., next/previous buttons) to browse through photos without returning to the gallery view.

    - Support for keyboard shortcuts (e.g., arrow keys) for navigation and an exit button to close the full-screen view.

Technology Stack

The Photo Gallery application can be developed using the following technologies:

- **Front-end**: HTML5, CSS3, JavaScript (with optional frameworks like React or Vue.js)
- **Back-end**: Node.js, Python (Django/Flask), or any other suitable web framework
- **Database**: SQLite, MySQL, or MongoDB for storing image metadata and user information
- **File Storage**: Local storage, cloud storage (AWS S3), or database storage for images

Security Considerations

- **Input Validation**: Ensure all uploaded files are validated to prevent security threats such as code injection.
- **File Size Limits**: Restrict the maximum upload size to prevent server overload.
- **Access Control**: Implement user authentication and authorization to control who can upload or view photos.

Future Enhancements

Potential enhancements for future development may include:

- User accounts for personalized galleries.
- Photo editing tools (e.g., cropping, filters).
- Tagging and categorization for easier photo management.
- Sharing options for social media platforms.

INTRODUCTION:

Ansible is an open-source IT automation tool that simplifies the management, configuration, and deployment of software across a wide range of systems. It allows users to define infrastructure as code, enabling consistent and repeatable processes that reduce human error and streamline complex tasks. Ansible uses a simple, agentless architecture, relying on standard SSH connections, which makes it lightweight and easy to set up compared to many traditional automation tools.

Designed to be highly flexible and powerful, Ansible can automate a variety of IT processes, including software provisioning, application deployment, configuration management, and orchestrating workflows across environments. By using playbooks written in YAML, Ansible makes it easy to define automation tasks in a human-readable format, enhancing collaboration among development and operations teams. Whether managing a few servers or thousands, Ansible scales efficiently, making it a popular choice for DevOps and system administrators seeking a robust automation solution.

Configuring Ansible on AWS EC2 Instances for a Master-Slave Architecture for a Photo Gallery Application

Introduction

This report outlines the process of configuring Ansible on AWS EC2 instances for a master-slave architecture tailored for deploying and managing a Photo Gallery application. The configuration will involve setting up a master node to control and manage tasks, while slave nodes will execute those tasks. This architecture enables efficient and automated provisioning, configuration management, and deployment of the application, making it scalable and manageable.

Objectives

The main objectives of this configuration include:

1. Setting up an Ansible control node (master) on AWS EC2.
2. Configuring slave nodes (managed nodes) to be controlled by the master node.
3. Deploying a Photo Gallery application on the slave nodes using Ansible playbooks.

Prerequisites

- An AWS account with permissions to create EC2 instances.
- SSH key pairs for secure communication between the master and slave instances.
- Basic understanding of Ansible, SSH, and Linux commands.

Step-by-Step Configuration

1. Launch AWS EC2 Instances

- **Create an EC2 Instance for the Master Node**:
    - Choose an Ubuntu 22.04 AMI (or your preferred version).
    - Select an appropriate instance type (e.g., t2.micro for testing).
    - Configure security group rules to allow SSH (port 22).
    - Assign or create an SSH key pair for secure access.
- **Create EC2 Instances for Slave Nodes**:
    - Launch one or more instances with similar configuration steps as above.

After connecting master and slave to Putty
we have specific commands to perform

IN MASTER MACHINE:

sudo apt-get update

sudo apt install software-properties-common

sudo apt-add-repository ppa:ansible/ansible

sudo apt update

sudo apt install ansible

ssh ubuntu@<ip-slave>

cd .ssh

ls

cat known_hosts

cat authorized_keys

ssh-keygen

ls

cat id_rsa.pub

we have to copy hash paste in slave

ssh ubuntu@<ip-slave>

exit

sudo nano /etc/ansible/hosts

[production]

slave1 ansible_ssh_host=slave<ipadd>

ansible -m ping all

ansible -m ping production

ansible -m ping slave1

In Slave Machine :

sudo apt-get update

sudo apt-get install python3

sudo apt-get install python3-pip

sudo apt-get install python-is-python3

python --version

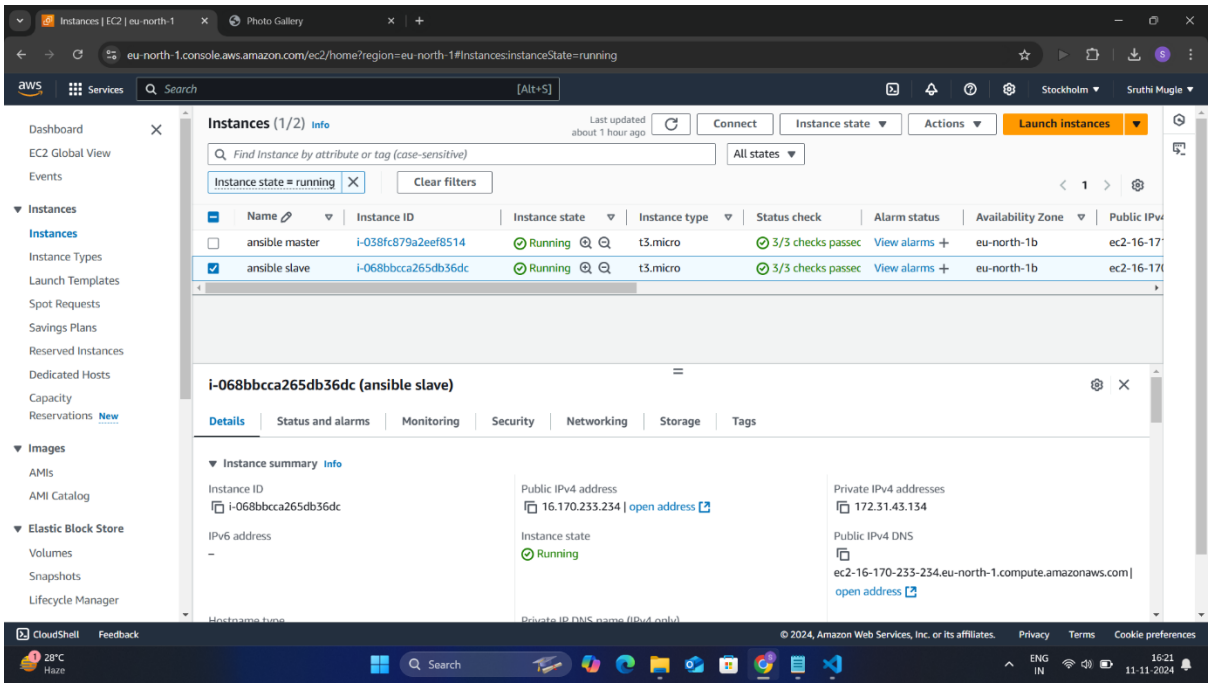
cd .ssh

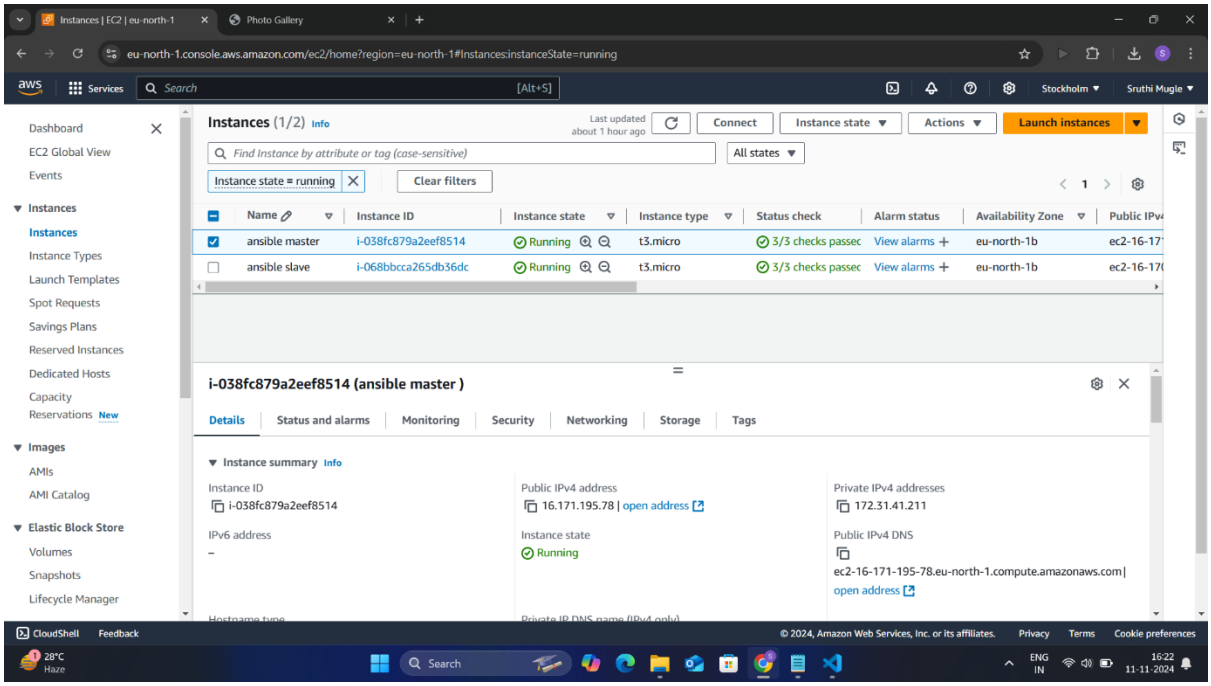
ls

sudo nano authorized_keys

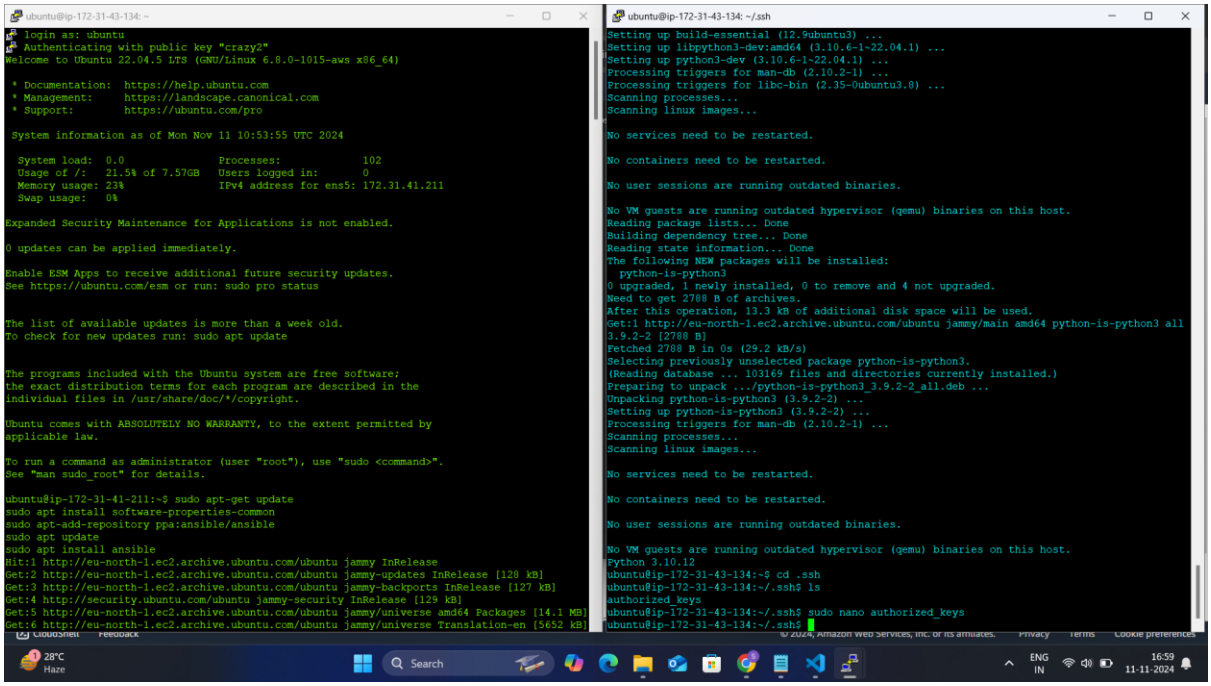
<copy hash>

so delete previous one and copy new





This is the AWS MACHINE:

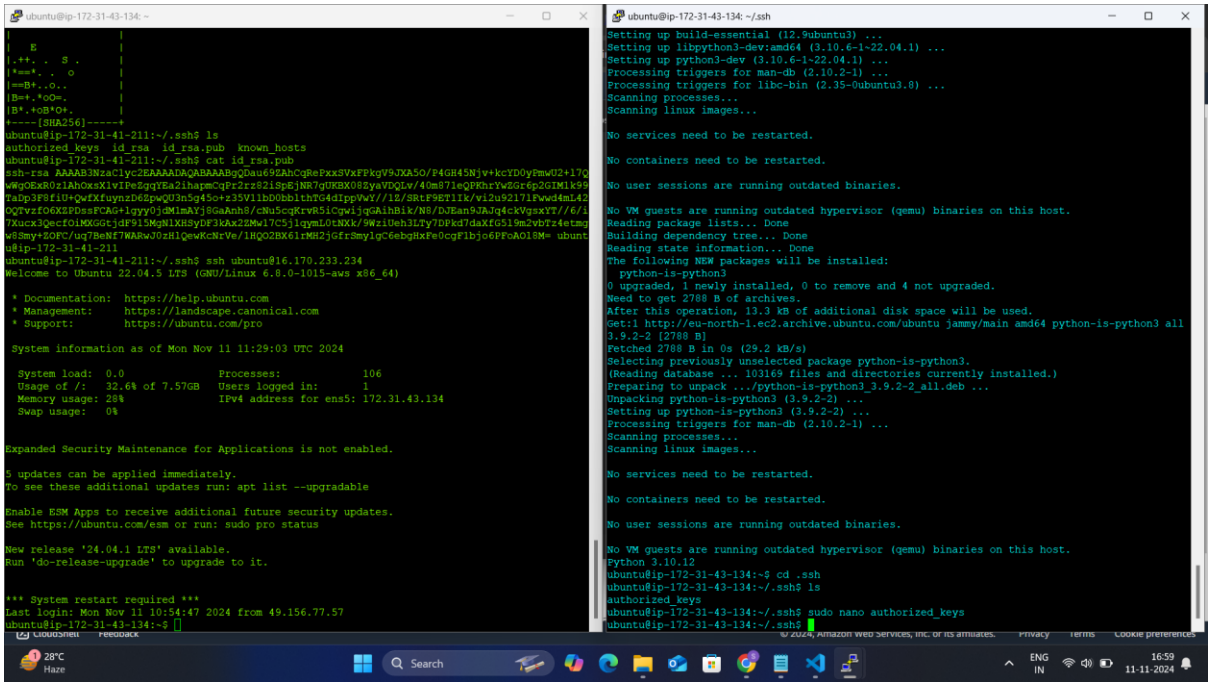


Green: is MASTER

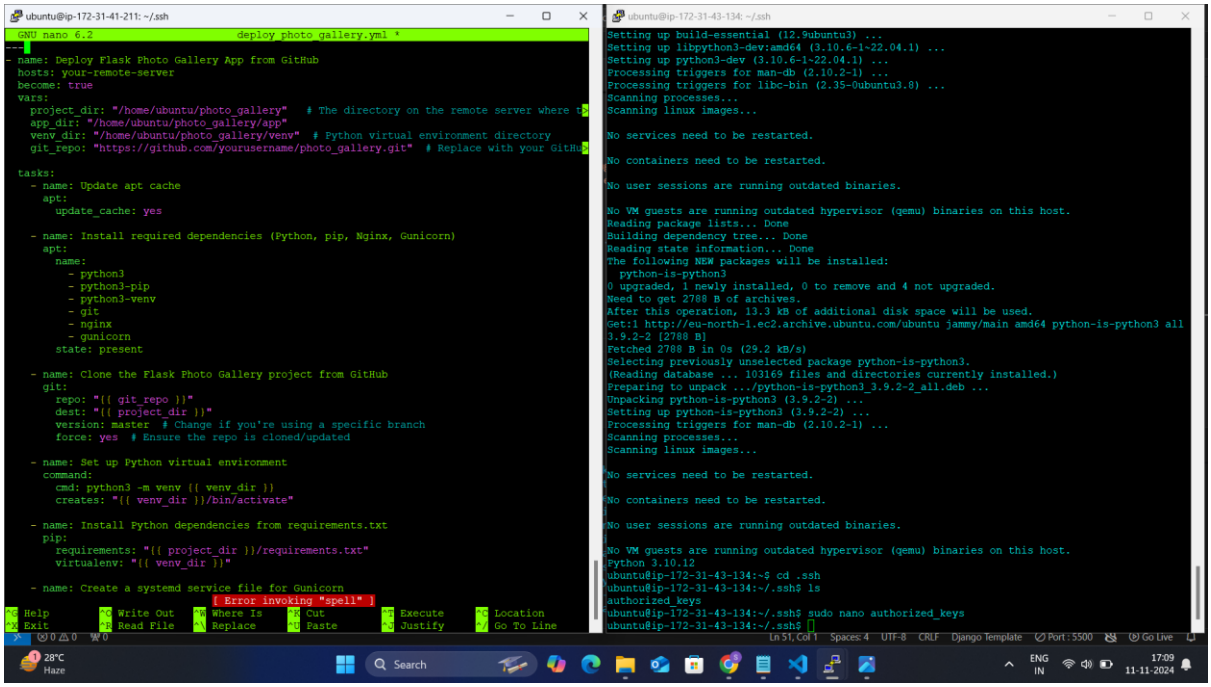
Blue : SLAVE

MASTER IP ADDRESS: 16.171.195.78

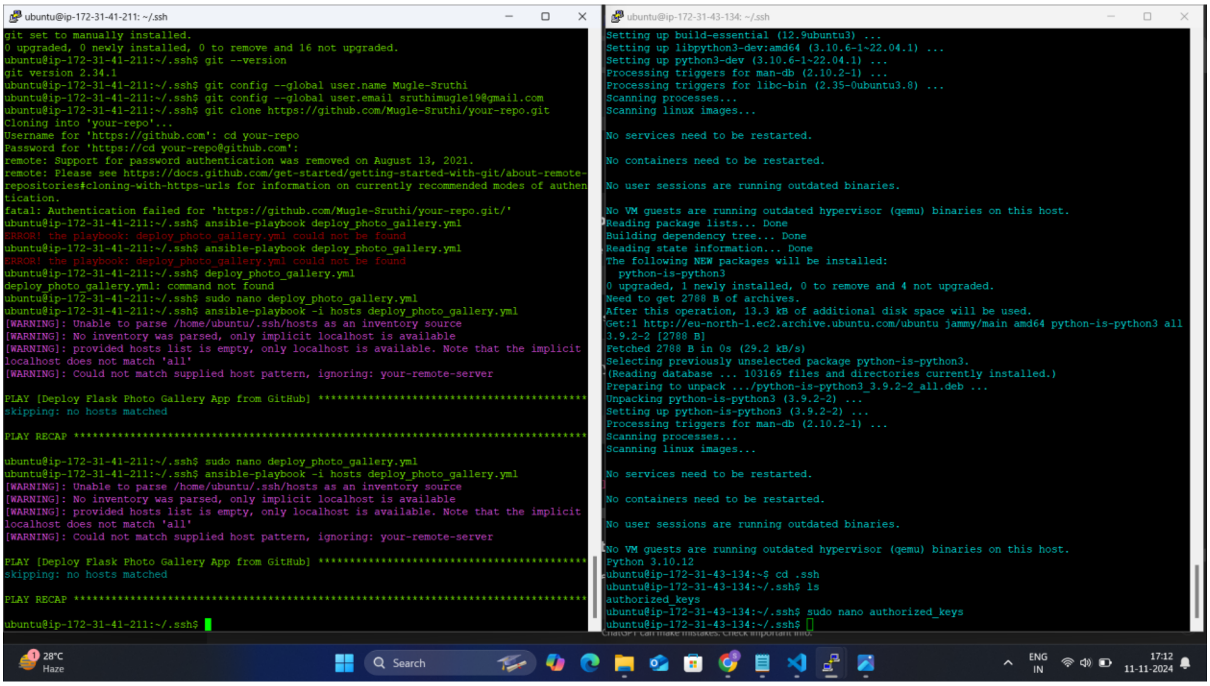
SLAVE IP ADRESS :16.170.233.234

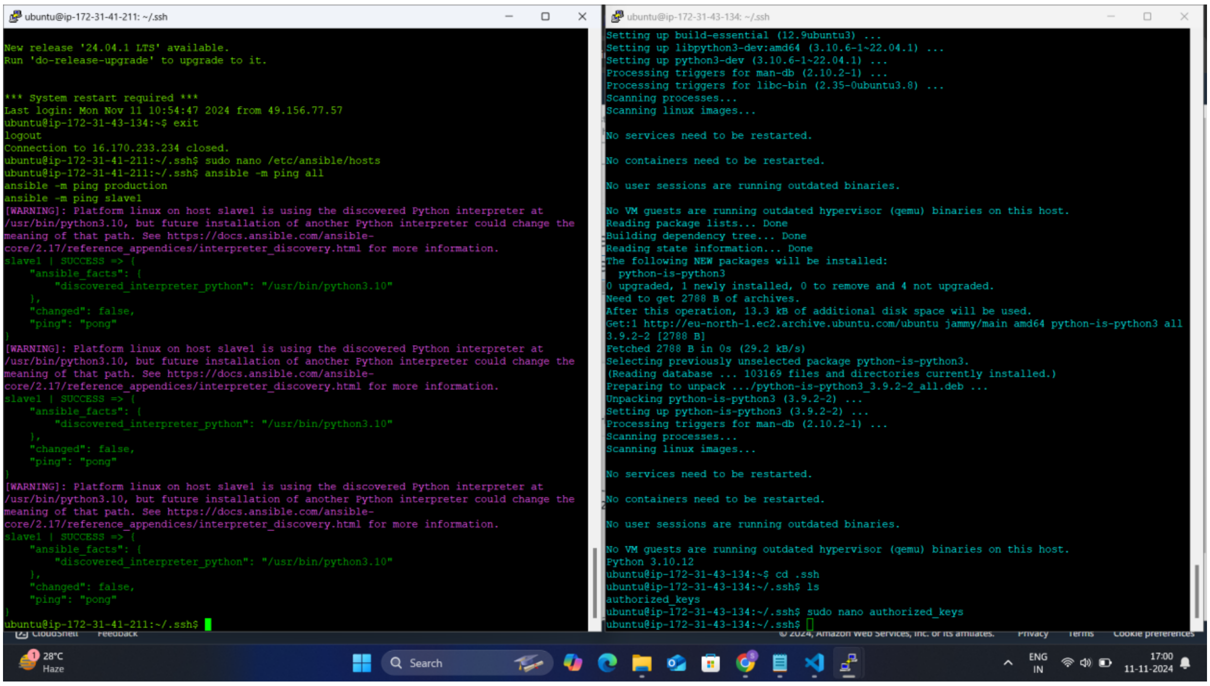


This is the Ansible playbook

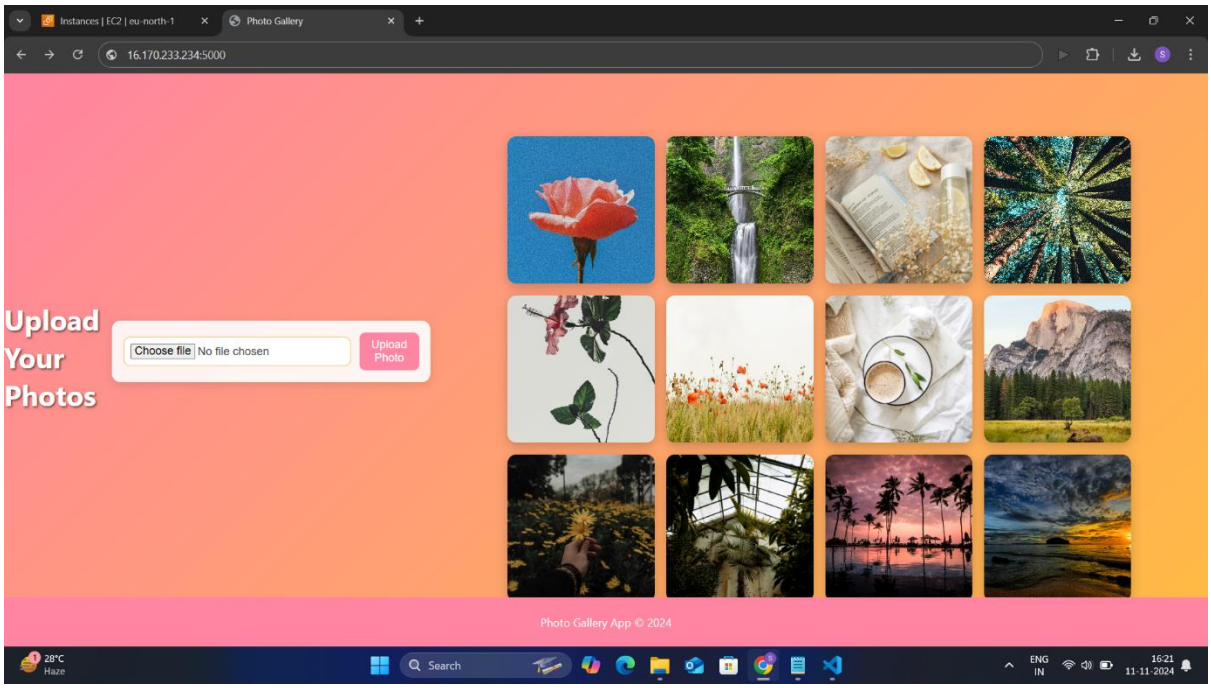


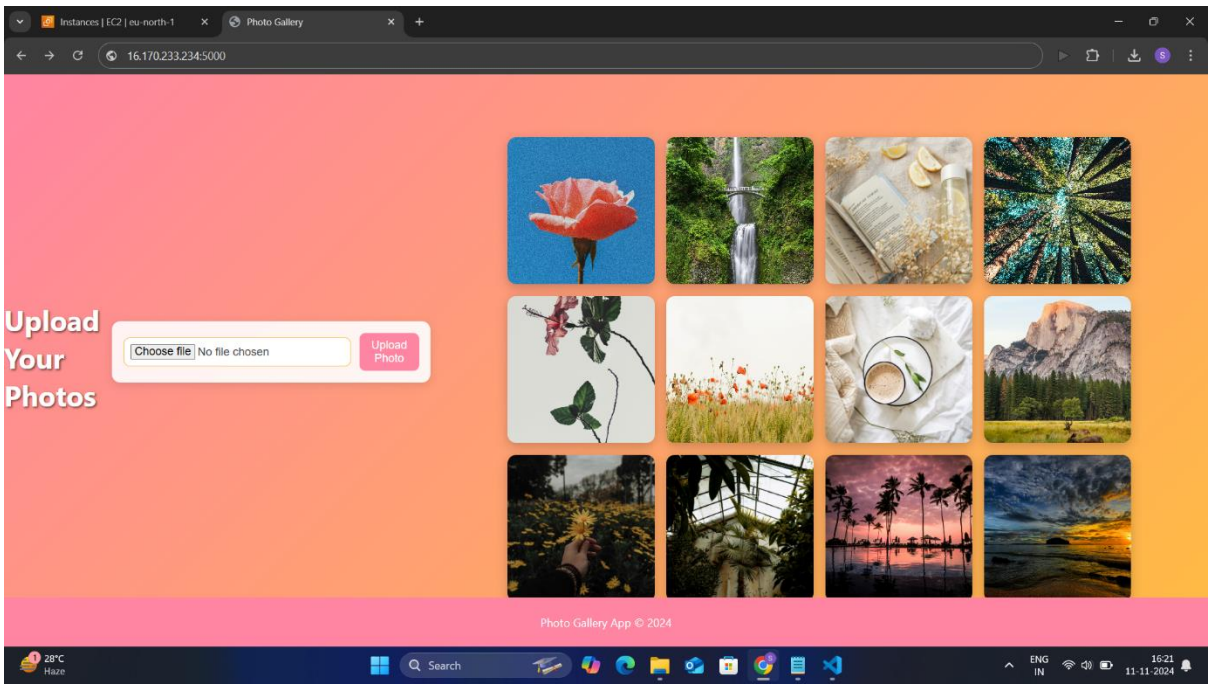
Before that I have cloned my Application from GitHub then I have performed ansible playbook and automated the configuration.


ubuntu@ip-172-31-41-211: ~/.ssh
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 16 not upgraded.
ubuntu@ip-172-31-41-211:~/.ssh$ git --version
git version 2.34.1
ubuntu@ip-172-31-41-211:~/.ssh$ git config --global user.name Mugle-Sruthi
ubuntu@ip-172-31-41-211:~/.ssh$ git config --global user.email sruthimugle19@gmail.com
ubuntu@ip-172-31-41-211:~/.ssh$ git clone https://github.com/Mugle-Sruthi/your-repo.git
Cloning into 'your-repo'...
Username for 'https://github.com': cd your-repo
Password for 'https://cd your-repo@github.com':
remote: Support for password authentication was removed on August 13, 2021.
fatal: Authentication failed for 'https://github.com/Mugle-Sruthi/your-repo.git/'
ubuntu@ip-172-31-41-211:~/.ssh$ ansible-playbook deploy_photo_gallery.yml
ERROR! the playbook: deploy_photo_gallery.yml could not be found
ubuntu@ip-172-31-41-211:~/.ssh$ deploy_photo_gallery.yml
deploy_photo_gallery.yml: command not found
ubuntu@ip-172-31-41-211:~/.ssh$ sudo nano deploy_photo_gallery.yml
ubuntu@ip-172-31-41-211:~/.ssh$ ansible-playbook -i hosts deploy_photo_gallery.yml
[WARNING]: Unable to parse /home/ubuntu/.ssh/hosts as an inventory source
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: Could not match supplied host pattern, ignoring: your-remote-server
PLAY [Deploy Flask Photo Gallery App from GitHub]
skipping: no hosts matched
PLAY RECAP
ubuntu@ip-172-31-43-134: ~/.ssh
Setting up build-essential (12.9ubuntu3) ...
Setting up python3-dev (3.10.6-1~22.04.1) ...
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.
No VM guests are running outdated hypervisor (qemu) binaries on this host.
The following NEW packages will be installed:
python-is-python3
Python 3.10.12
ubuntu@ip-172-31-43-134:~$ cd .ssh
ubuntu@ip-172-31-43-134:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-43-134:~/.ssh$ sudo nano authorized_keys
17:12
11-11-2024


ubuntu@ip-172-31-41-211: ~/.ssh
New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
*** System restart required ***
Last login: Mon Nov 11 10:54:47 2024 from 49.156.77.57
ubuntu@ip-172-31-43-134:~$ exit
logout
Connection to 16.170.233.234 closed.
ubuntu@ip-172-31-41-211:~/.ssh$ sudo nano /etc/ansible/hosts
ubuntu@ip-172-31-41-211:~/.ssh$ ansible -m ping all
ansible -m ping production
ansible -m ping slave1
[WARNING]: Platform linux on host slave1 is using the discovered Python interpreter at /usr/bin/python3.10, but future installation of another Python interpreter could change the meaning of that path.
slave1 | SUCCESS => {
"discovered_interpreter_python": "/usr/bin/python3.10"
"changed": false,
"ping": "pong"
ubuntu@ip-172-31-43-134: ~/.ssh
Python 3.10.12
ubuntu@ip-172-31-43-134:~/.ssh$ sudo nano authorized_keys
17:00
11-11-2024

After that playbook starts executing on web Browser :
It has been exposed in web browser through slave machine Ip address using 5000 port number





Conclusion:

Configuring Ansible in a master-slave architecture on AWS EC2

provides a robust and efficient method for deploying the Photo Gallery application. By automating tasks such as software installation, configuration management, and deployment, Ansible streamlines operational workflows, enhancing the scalability and maintainability of the application.