Step 1: Prerequisites

1. **Set up your Ansible control node and managed nodes.**
   - Ensure you have SSH access to the managed nodes.
   - Install Ansible on the control node.
   - Create an inventory file listing the managed nodes.
2. **Install Ansible on the control node:**

   ```
   bash
   Copy code
   sudo apt update
   sudo apt install ansible -y
   ```

3. **Test connectivity:**

   ```
   bash
   Copy code
   ansible all -m ping -i inventory
   ```

Step 2: Create the Ansible Directory Structure

Organize your project:

```
bash
Copy code
mkdir lamp-deployment
cd lamp-deployment
mkdir group_vars roles inventory
```

Step 3: Define Inventory

Create an `inventory` file to specify the target servers:

```
ini
Copy code
[webservers]
web1 ansible_host=192.168.1.10 ansible_user=ubuntu
ansible_ssh_private_key_file=~/.ssh/id_rsa
web2 ansible_host=192.168.1.11 ansible_user=ubuntu
ansible_ssh_private_key_file=~/.ssh/id_rsa
```

Step 4: Create Playbook for LAMP Deployment

Create a playbook file `lamp.yml`:

```
yaml
Copy code
- name: LAMP Stack Deployment
  hosts: webservers
  become: true
  vars_files:
    - group_vars/secure.yml

  tasks:
    - name: Update and upgrade apt packages
      apt:
        update_cache: yes
        upgrade: dist

    - name: Install Apache
      apt:
        name: apache2
        state: present

    - name: Enable and start Apache service
      service:
        name: apache2
        state: started
        enabled: true

    - name: Install MySQL Server
      apt:
        name: mysql-server
        state: present

    - name: Secure MySQL installation
      mysql_secure_installation:
        login_user: root
        login_password: "{{ mysql_root_password }}"
        root_password: "{{ mysql_root_password }}"
        remove_anonymous_users: yes
        disallow_root_login_remotely: yes
        remove_test_database: yes

    - name: Create a MySQL database
      mysql_db:
        name: myapp
        state: present

    - name: Create a MySQL user
```

```
      mysql_user:
        name: myapp_user
        password: "{{ mysql_app_password }}"
        priv: 'myapp.*:ALL'
        state: present

    - name: Install PHP
      apt:
        name: php
        state: present

    - name: Install PHP MySQL module
      apt:
        name: php-mysql
        state: present

    - name: Restart Apache to apply PHP module
      service:
        name: apache2
        state: restarted
```

Step 5: Secure Credentials with Ansible Vault

Create a file `group_vars/secure.yml` to store sensitive information:

```
yaml
Copy code
mysql_root_password: "rootpassword"
mysql_app_password: "appuserpassword"
```

Encrypt this file using Ansible Vault:

```
bash
Copy code
ansible-vault encrypt group_vars/secure.yml
```

Step 6: Role Directory Structure

You can modularize this deployment by creating an Ansible role:

```
bash
Copy code
ansible-galaxy init roles/lamp
```

Replace tasks in `roles/lamp/tasks/main.yml` with:

```
yaml
Copy code
- include_tasks: apache.yml
- include_tasks: mysql.yml
- include_tasks: php.yml
```

Create respective `apache.yml`, `mysql.yml`, and `php.yml` in the `tasks/` folder with specific configurations.

Step 7: Execute the Playbook

Run the playbook to deploy the LAMP stack:

```
bash
Copy code
ansible-playbook lamp.yml -i inventory --ask-vault-pass
```

Step 8: Verify the Deployment

1. SSH into your target servers and verify:
   - Apache: Open the IP in a browser (`http://<server-ip>`).
   - MySQL: Log in using `mysql -u root -p`.
   - PHP: Create a `phpinfo()` file in `/var/www/html` and access it in the browser.
2. Test database connectivity using a PHP script.