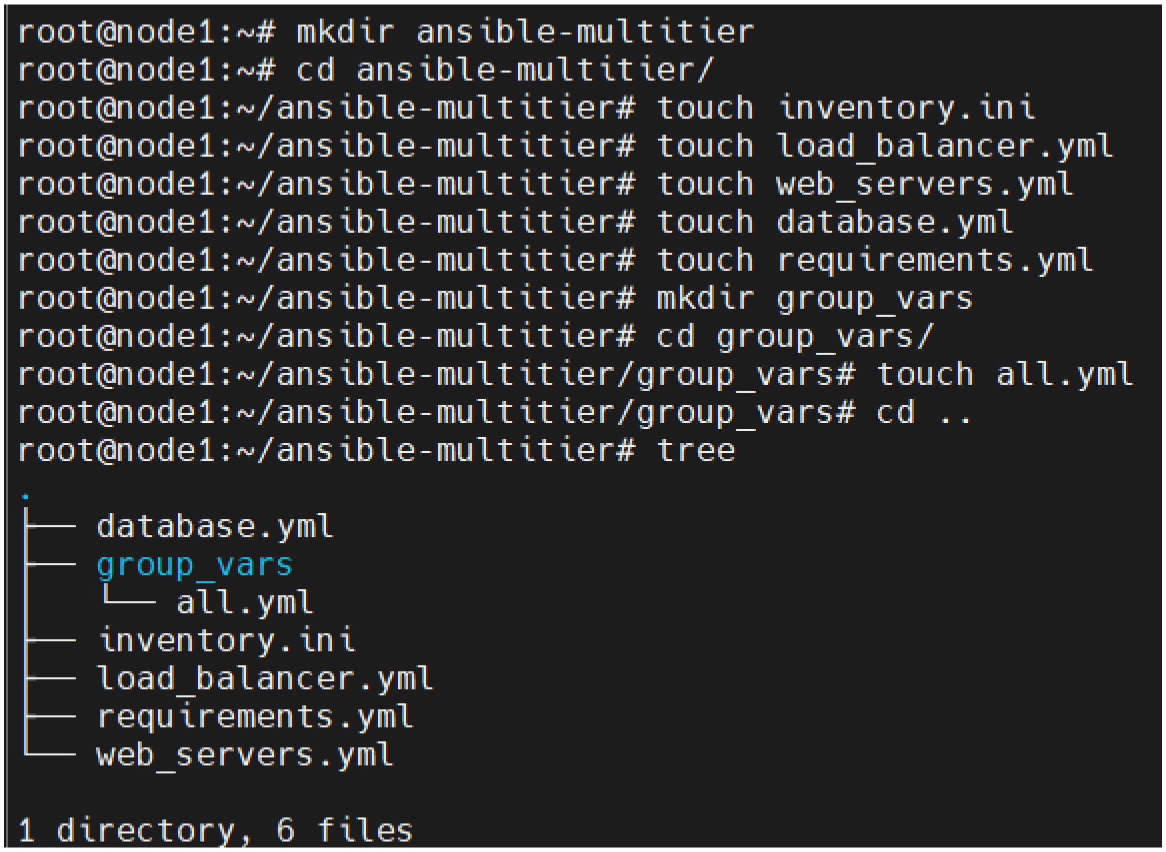
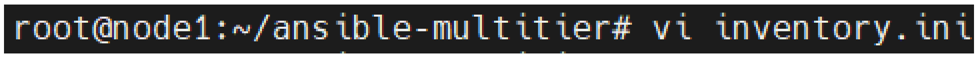
Zeynalabdin Hamidov

1. **Prepare project directory structure:**

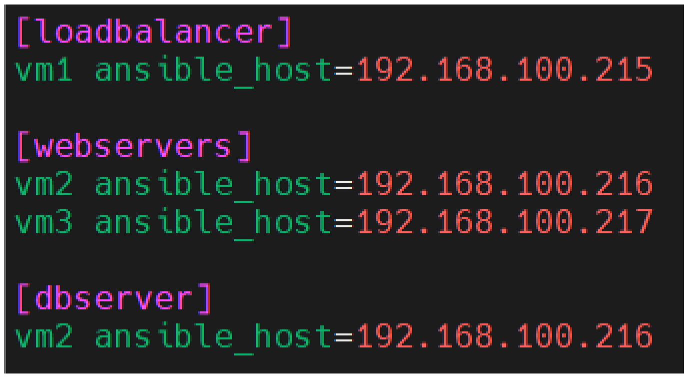


```
root@node1:~# mkdir ansible-multitier
root@node1:~# cd ansible-multitier/
root@node1:~/ansible-multitier# touch inventory.ini
root@node1:~/ansible-multitier# touch load_balancer.yml
root@node1:~/ansible-multitier# touch web_servers.yml
root@node1:~/ansible-multitier# touch database.yml
root@node1:~/ansible-multitier# touch requirements.yml
root@node1:~/ansible-multitier# mkdir group_vars
root@node1:~/ansible-multitier# cd group_vars/
root@node1:~/ansible-multitier/group_vars# touch all.yml
root@node1:~/ansible-multitier/group_vars# cd ..
root@node1:~/ansible-multitier# tree
.
├── database.yml
├── group_vars
│   └── all.yml
├── inventory.ini
├── load_balancer.yml
├── requirements.yml
└── web_servers.yml

1 directory, 6 files
```

Write inventory.ini file:

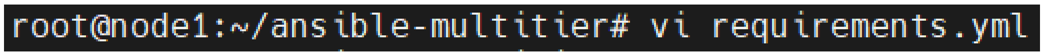


```
root@node1:~/ansible-multitier# vi inventory.ini
```

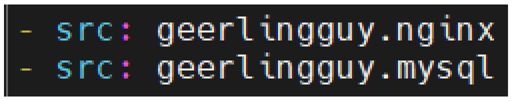


```
[loadbalancer]
vm1 ansible_host=192.168.100.215

[webservers]
vm2 ansible_host=192.168.100.216
vm3 ansible_host=192.168.100.217

[dbserver]
vm2 ansible_host=192.168.100.216
```

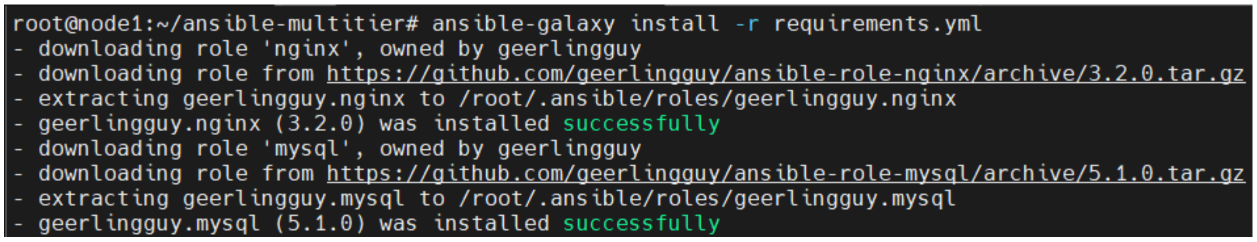
2. **Download Roles with ansible-galaxy using "requirements.yml" file:**



```
root@node1:~/ansible-multitier# vi requirements.yml
```

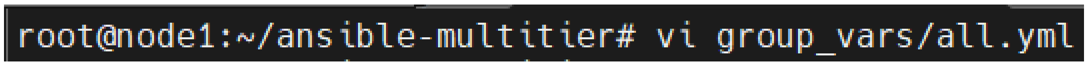


```
- src: geerlingguy.nginx
- src: geerlingguy.mysql
```

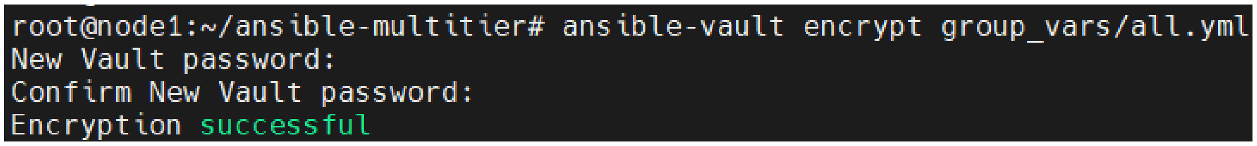


```
root@node1:~/ansible-multitier# ansible-galaxy install -r requirements.yml
- downloading role 'nginx', owned by geerlingguy
- downloading role from https://github.com/geerlingguy/ansible-role-nginx/archive/3.2.0.tar.gz
- extracting geerlingguy.nginx to /root/.ansible/roles/geerlingguy.nginx
- geerlingguy.nginx (3.2.0) was installed successfully
- downloading role 'mysql', owned by geerlingguy
- downloading role from https://github.com/geerlingguy/ansible-role-mysql/archive/5.1.0.tar.gz
- extracting geerlingguy.mysql to /root/.ansible/roles/geerlingguy.mysql
- geerlingguy.mysql (5.1.0) was installed successfully
```

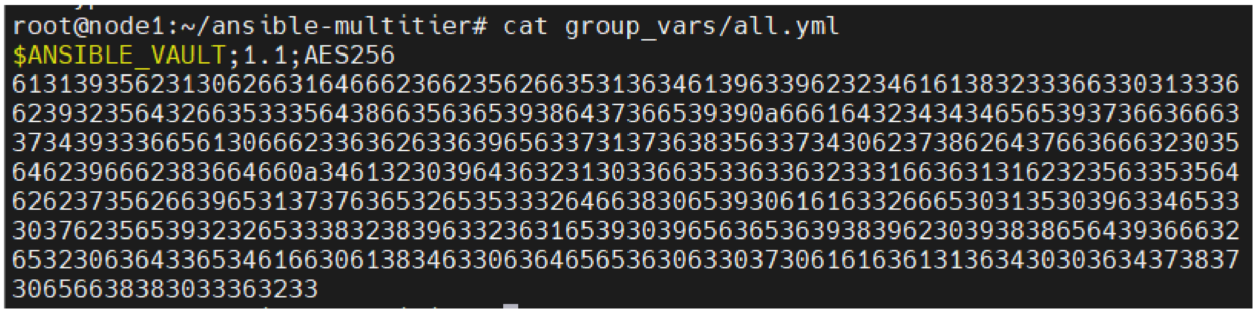
3. Write security credentials inside "group_vars/all.yml" file and encrypt using ansible-vault:



```
root@node1:~/ansible-multitier# vi group_vars/all.yml
```

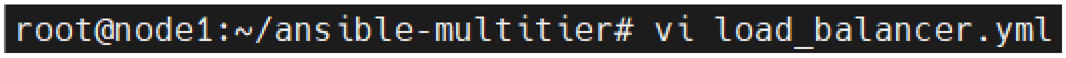


```
root@node1:~/ansible-multitier# ansible-vault encrypt group_vars/all.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

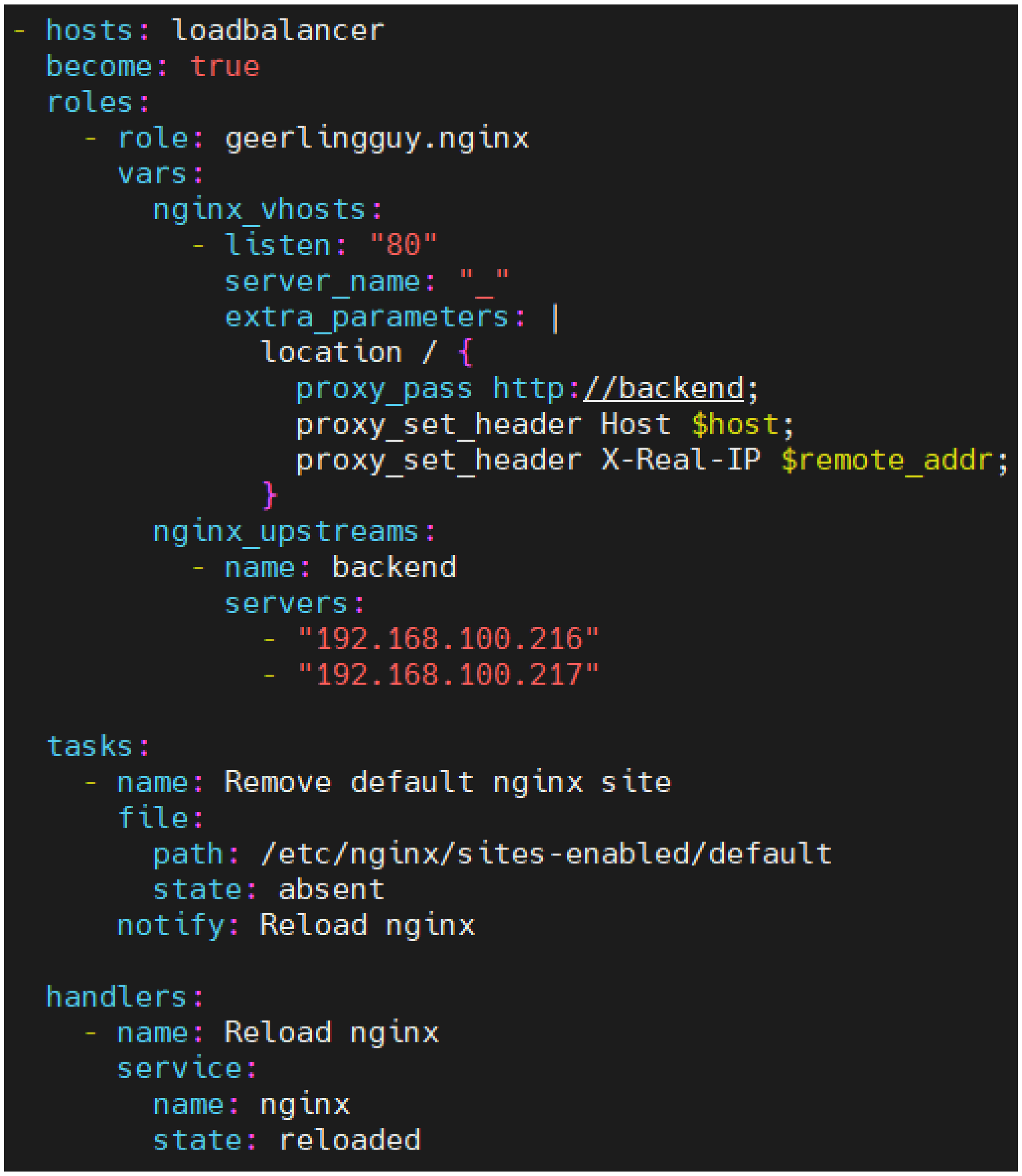


```
root@node1:~/ansible-multitier# cat group_vars/all.yml
$ANSIBLE_VAULT;1.1;AES256
61313935623130626631646662366235626635313634613963396232346161383233366330313336
6239323564326635333564386635636539386437366539390a666164323434346565393736636663
37343933366561306662336362633639656337313736383563373430623738626437663666323035
6462396662383664660a346132303964363231303366353363363233316636313162323563353564
62623735626639653137376365326535333264663830653930616163326665303135303963346533
30376235653932326533383238396332363165393039656365363938396230393838656439366632
65323063643365346166306138346330636465653630633037306161636131363430303634373837
3065663838303336323 3
```

4. Write "load_balancer.yml" playbook:

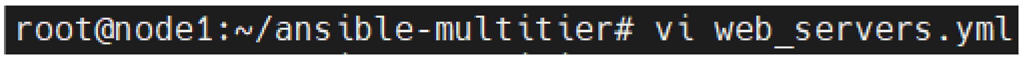


```
root@node1:~/ansible-multitier# vi load_balancer.yml
```

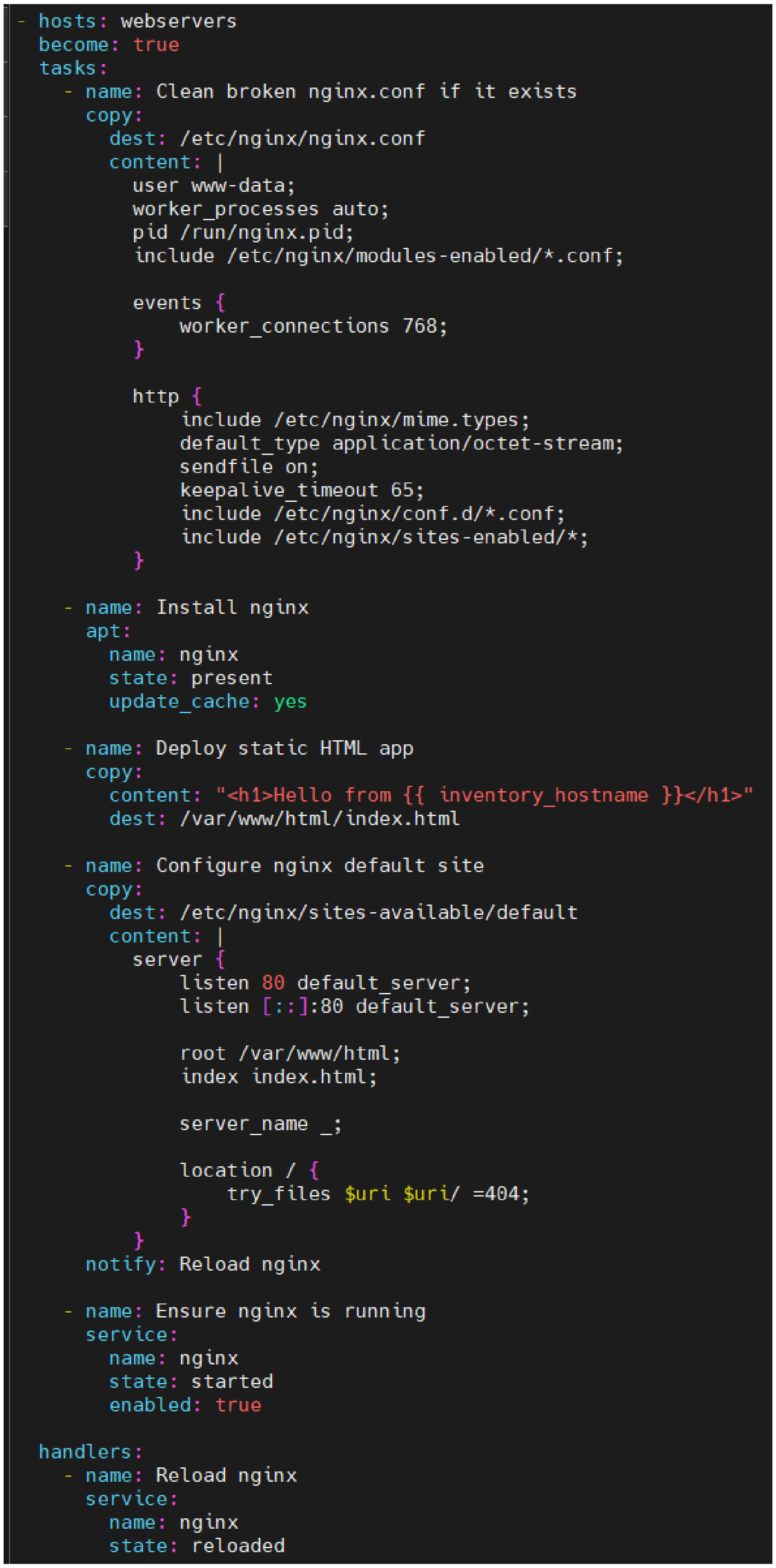


```
- hosts: loadbalancer
  become: true
  roles:
    - role: geerlingguy.nginx
      vars:
        nginx_vhosts:
          - listen: "80"
            server_name: "_"
            extra_parameters: |
              location / {
                proxy_pass http://backend;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
              }
        nginx_upstreams:
          - name: backend
            servers:
              - "192.168.100.216"
              - "192.168.100.217"

  tasks:
    - name: Remove default nginx site
      file:
        path: /etc/nginx/sites-enabled/default
        state: absent
      notify: Reload nginx

  handlers:
    - name: Reload nginx
      service:
        name: nginx
        state: reloaded
```

5. Write "web_servers.yml" playbook:

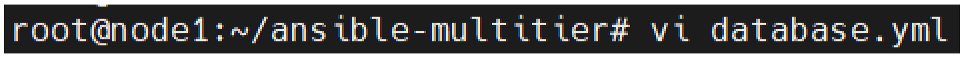


```
root@node1:~/ansible-multitier# vi web_servers.yml
```

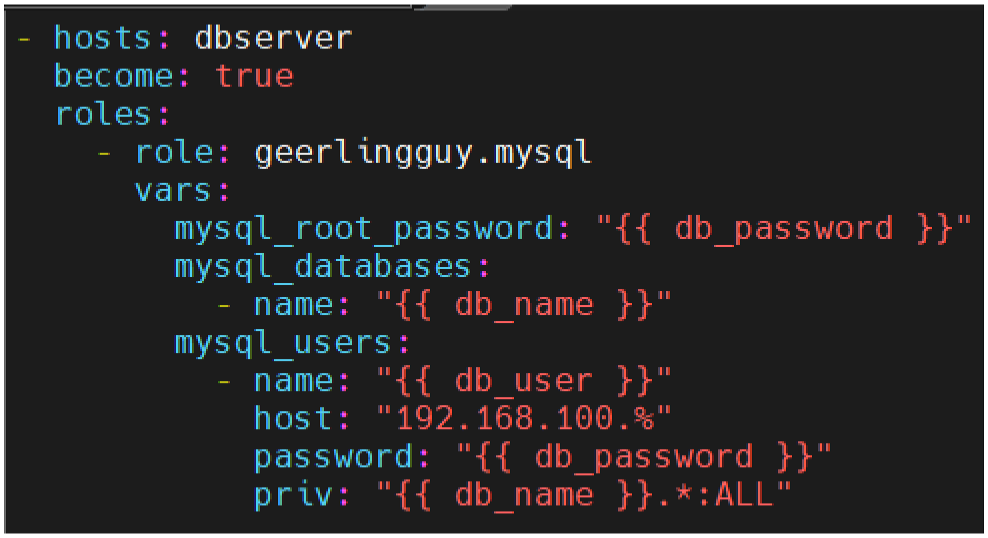


```
- hosts: webservers
  become: true
  tasks:
    - name: Clean broken nginx.conf if it exists
      copy:
        dest: /etc/nginx/nginx.conf
        content: |
          user www-data;
          worker_processes auto;
          pid /run/nginx.pid;
          include /etc/nginx/modules-enabled/*.conf;

          events {
              worker_connections 768;
          }

          http {
              include /etc/nginx/mime.types;
              default_type application/octet-stream;
              sendfile on;
              keepalive_timeout 65;
              include /etc/nginx/conf.d/*.conf;
              include /etc/nginx/sites-enabled/*;
          }

    - name: Install nginx
      apt:
        name: nginx
        state: present
        update_cache: yes

    - name: Deploy static HTML app
      copy:
        content: "<h1>Hello from {{ inventory_hostname }}</h1>"
        dest: /var/www/html/index.html

    - name: Configure nginx default site
      copy:
        dest: /etc/nginx/sites-available/default
        content: |
          server {
              listen 80 default_server;
              listen [::]:80 default_server;

              root /var/www/html;
              index index.html;

              server_name _;

              location / {
                  try_files $uri $uri/ =404;
              }
          }
      notify: Reload nginx

    - name: Ensure nginx is running
      service:
        name: nginx
        state: started
        enabled: true

  handlers:
    - name: Reload nginx
      service:
        name: nginx
        state: reloaded
```

6. **Write “database.yml" playbook:**



```
root@node1:~/ansible-multitier# vi database.yml
```



```
- hosts: dbserver
  become: true
  roles:
    - role: geerlingguy.mysql
      vars:
        mysql_root_password: "{{ db_password }}"
        mysql_databases:
          - name: "{{ db_name }}"
        mysql_users:
          - name: "{{ db_user }}"
            host: "192.168.100.%"
            password: "{{ db_password }}"
            priv: "{{ db_name }}.*:ALL"
```

7. **Run playbooks in sequence using commands:**

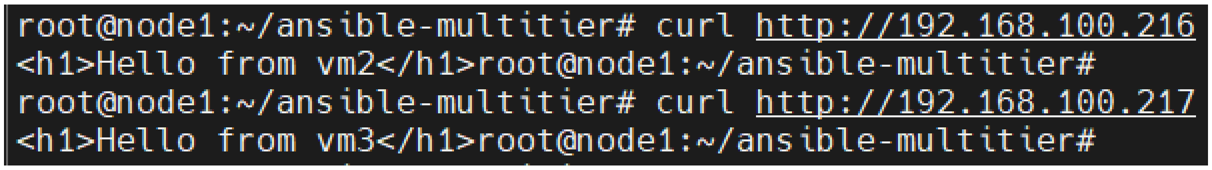
ansible-playbook -i inventory.ini load_balancer.yml--ask-vault-pass

ansible-playbook -i inventory.ini web_servers.yml--ask-vault-pass

ansible-playbook -i inventory.ini database.yml --ask-vault-pass

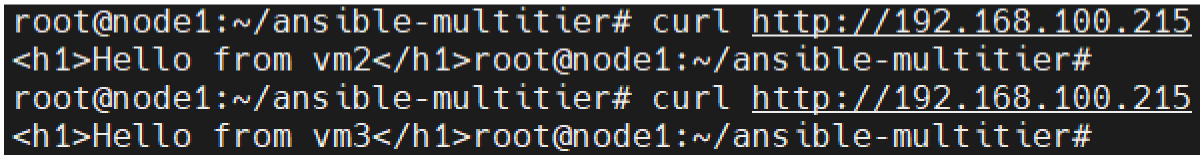
Tests:

1) **Web Servers:**



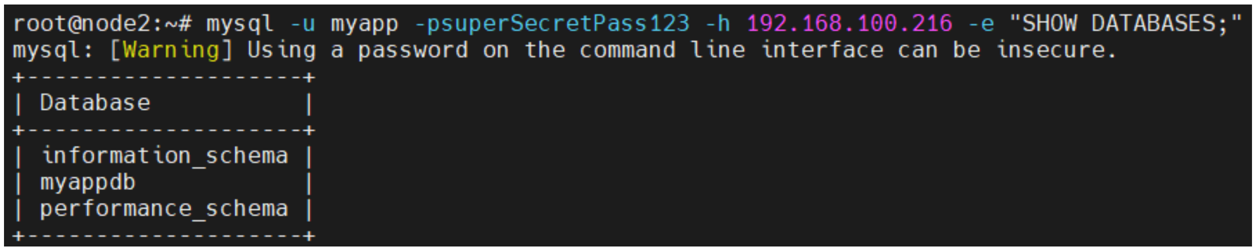
```
root@node1:~/ansible-multitier# curl http://192.168.100.216
<h1>Hello from vm2</h1>root@node1:~/ansible-multitier#
root@node1:~/ansible-multitier# curl http://192.168.100.217
<h1>Hello from vm3</h1>root@node1:~/ansible-multitier#
```

2) **LoadBalancer:**



```
root@node1:~/ansible-multitier# curl http://192.168.100.215
<h1>Hello from vm2</h1>root@node1:~/ansible-multitier#
root@node1:~/ansible-multitier# curl http://192.168.100.215
<h1>Hello from vm3</h1>root@node1:~/ansible-multitier#
```

3) **DB:**



```
root@node2:~# mysql -u myapp -psuperSecretPass123 -h 192.168.100.216 -e "SHOW DATABASES;"
mysql: [Warning] Using a password on the command line interface can be insecure.
+--------------------+
| Database           |
+--------------------+
| information_schema |
| myappdb            |
| performance_schema |
+--------------------+
```