PROJECT 12: ANSIBLE REFACTORING AND STATIC ASSIGNMENTS (IMPORTS AND ROLES)

This project progresses from project 11, which means that I will continue working with ansible-config repository and make some improvements to my code. There is a need to refactor the Ansible code, create assignments, and learn how to use the imports functionality. Imports allow you to effectively re-use previously created playbooks in a new playbook – it allows you to organize your tasks and reuse them when needed.

Code Refactoring

Refactoring is a controlled technique for improving the design of an existing code base. By doing them in small steps you reduce the risk of introducing errors. You also avoid having the system broken while you are carrying out the restructuring - which allows you to gradually refactor a system over an extended period of time. The aim of this project is to make some code changes but the overall state of the infrastructure shall remain the same.

Step 1 – Jenkins Job Enhancement

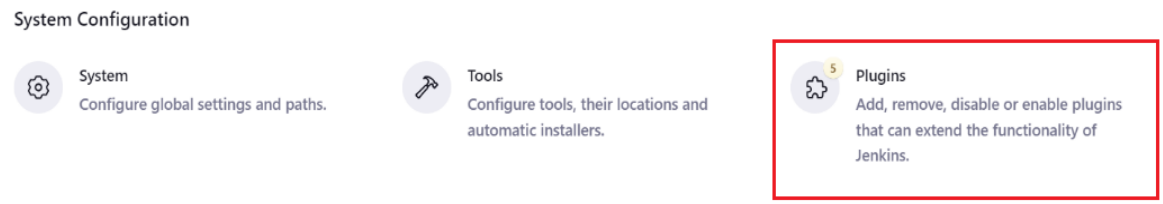
There need to be some changes to be made to the Jenkins job. Following the previous project, every new change in the codes creates a separate directory which is not very convenient when we want to run some commands from one place. The current process also consumes space on Jenkins serves with each subsequent change. This can be enhanced with the help of a new Jenkins project/job which will require Copy Artifact plugin to work with.

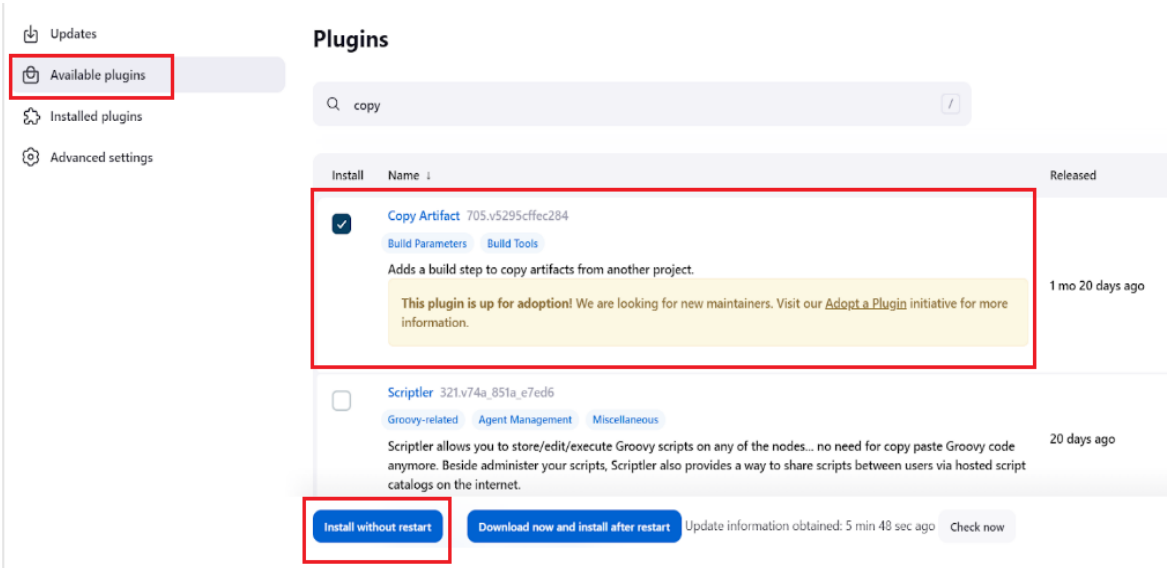
1. Go to your Jenkins-Ansible server and create a new directory called ansible-config-artifact – this will store all artifacts after each build and then change permissions to this directory so that Jenkins could save files there.

```
ubuntu@ip-172-31-40-224:~$ sudo mkdir /home/ubuntu/ansible-config-artifact
```

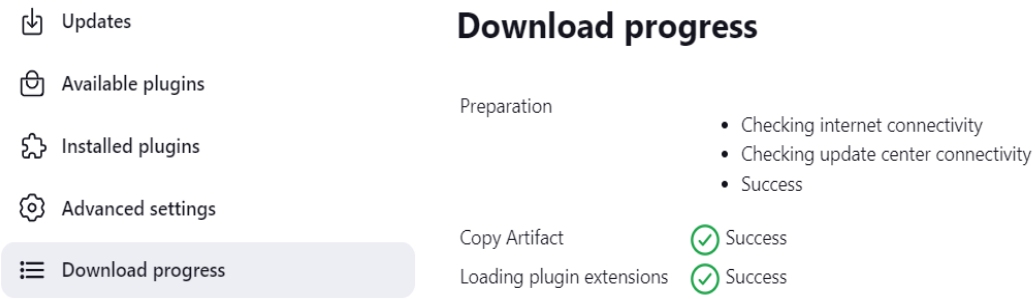
```
ubuntu@ip-172-31-40-224:~$ sudo chmod -R 0777 /home/ubuntu/ansible-config-artifact
```

2. Go to Jenkins web console > Manage Jenkins > Plugins > on Available tab search for Copy Artifact and install this plugin without restarting Jenkins.

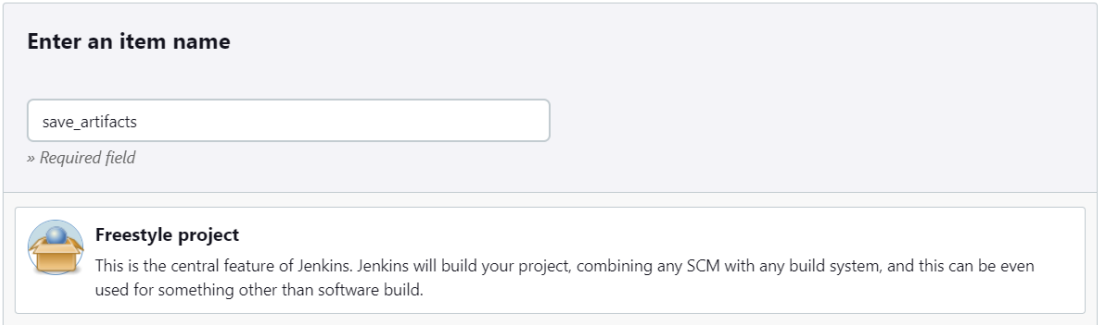




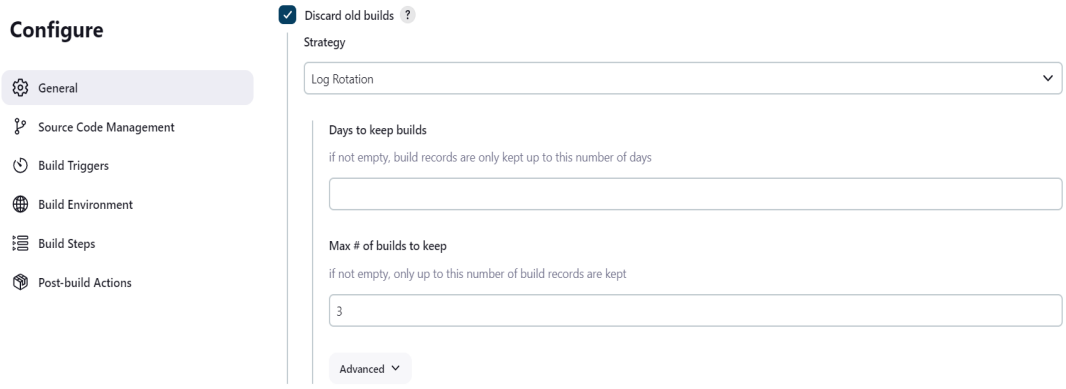
3. This will display the download progress status.



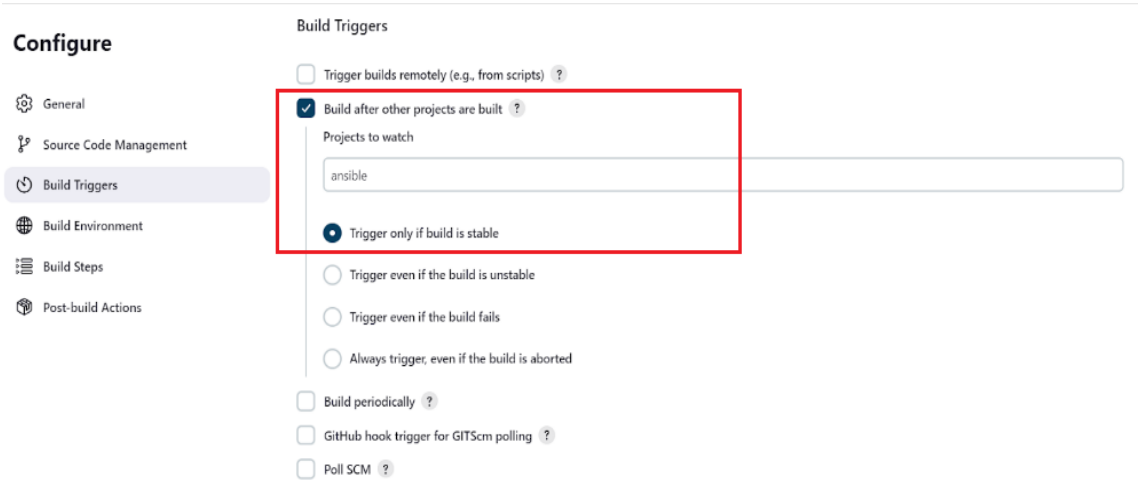
4. Create a new Freestyle project and name it save_artifacts and configure it accordingly. This project will be triggered by the completion of your existing ansible project.



5. I configured the number of builds to 3 to save space on the server.

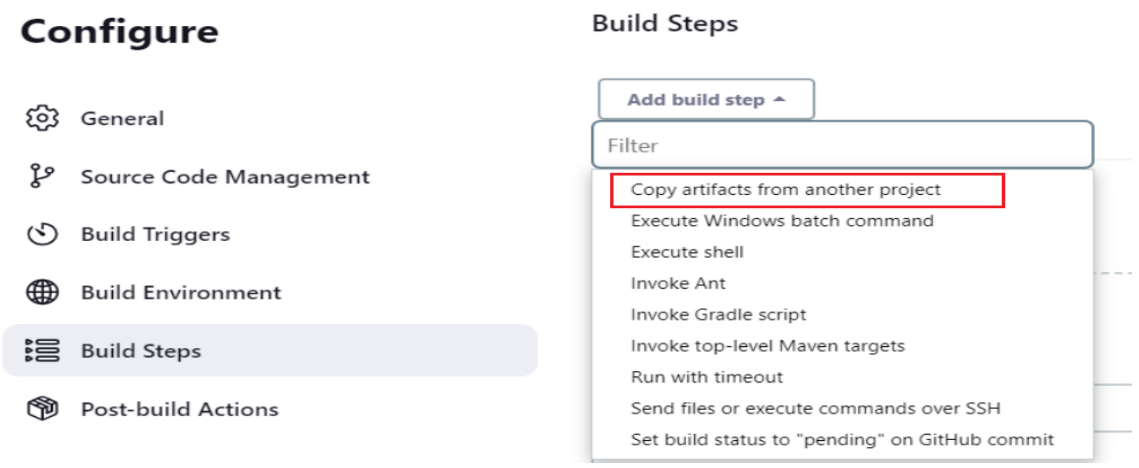


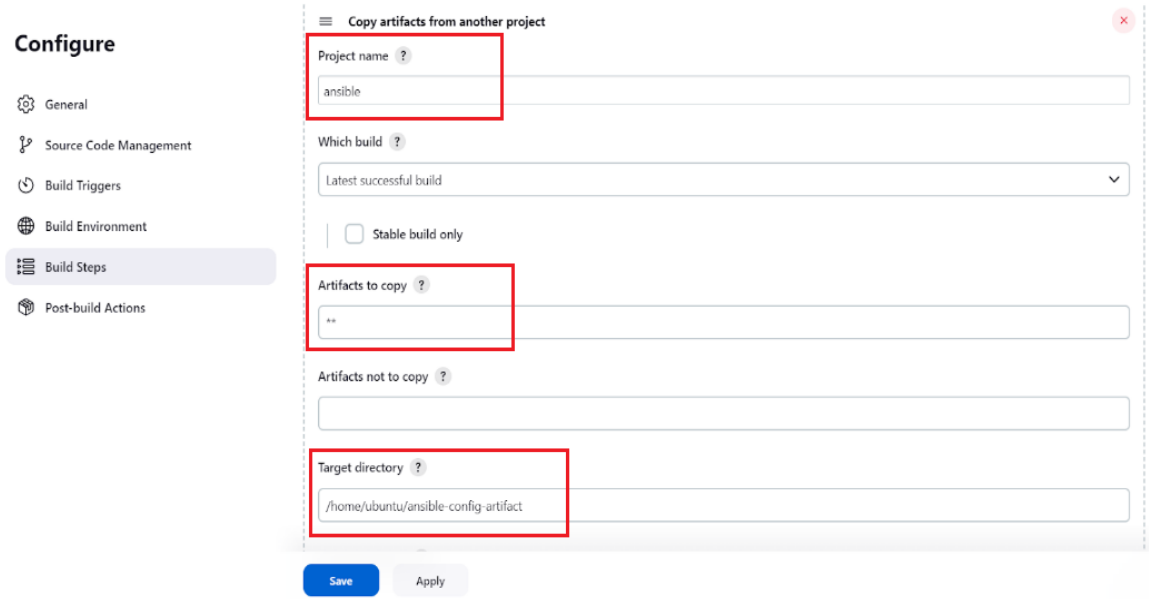
6. The build trigger will watch the project ansible.



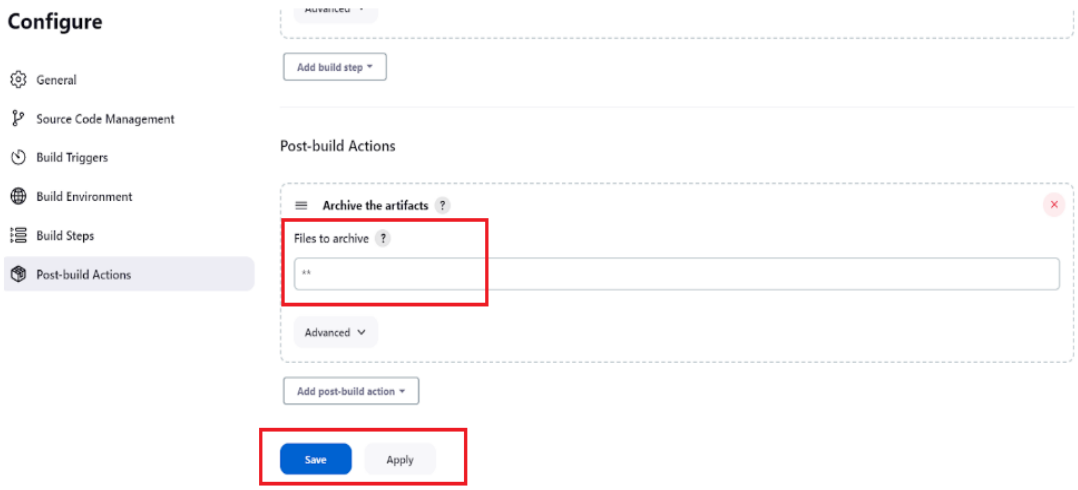
7. The main idea of save_artifacts project is to save artifacts into /home/ubuntu/ansible-config-artifact directory. To achieve this, I created a Build step and

chose Copy artifacts from another project, specifying ansible as a source project and /home/ubuntu/ansible-config-artifact as a target directory.



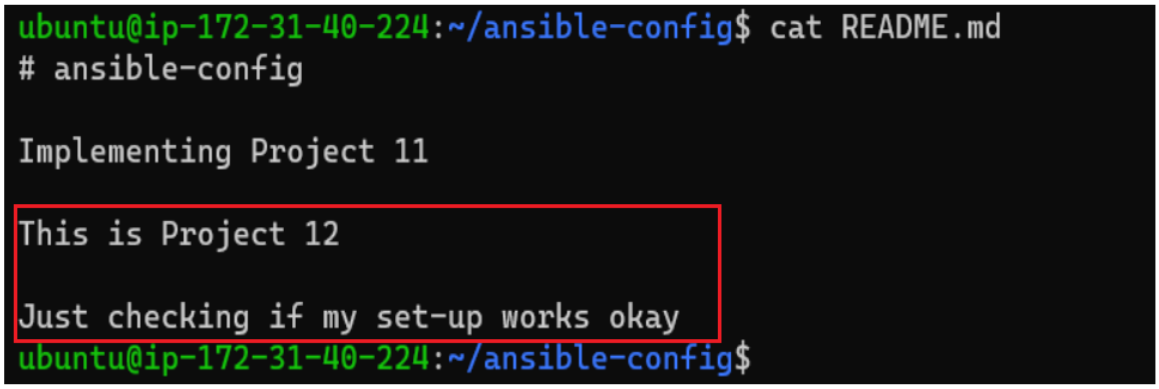


8. Select all files to archive the artifacts under the post-build action, apply and then save the changes.



9. The console output should display a successful build for the save_artifacts project.

10. Now it is time to test the Jenkins Pipeline by making some changes in the README.md file inside the ansible-config repository, right inside the main branch. If both Jenkins jobs have been completed one after another, the files will be in /home/ubuntu/ansible-config-artifact directory and it will be updated with every commit to the main branch.



Step 2 – Refactor Ansible code by importing other playbooks into site.yml

Before starting to refactor the codes, I need to pull down the latest code from the main branch and created a new branch called refactor.

```
ubuntu@ip-172-31-40-224:~/ansible-config$ git status
On branch feature/proj45
nothing to commit, working tree clean
ubuntu@ip-172-31-40-224:~/ansible-config$ git checkout -b refactor
Switched to a new branch 'refactor'
ubuntu@ip-172-31-40-224:~/ansible-config$
```

In Project 11 all tasks were in a single playbook common.yml, but this approach will not work if there are more tasks and you need to apply this playbook to other servers with different requirements. Ansible allows for the one-file approach first, breaking tasks up into different files with complex sets of tasks to reuse them.

1. Within the playbooks folder, I created a new file and named it site.yml – This file will now be considered as an entry point into the entire infrastructure configuration. Other playbooks will be included here as a reference. In other words, site.yml will become a parent to all other playbooks that will be developed.

```
ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ ls
site.yml
```

2. Create a new folder in the root of the repository and name it static-assignments. The static-assignments folder is where all other children playbooks will be stored. This is merely for easy organization of your work.

```
ubuntu@ip-172-31-40-224:~/ansible-config$ ls
README.md  ansible-config  inventory  playbooks  roles  webserver
ubuntu@ip-172-31-40-224:~/ansible-config$ mkdir static-assignments
```

3. Move the common.yml file into the newly created static-assignments folder.

```
ubuntu@ip-172-31-40-224:~/ansible-config/static-assignments$ ls
common.yml
```

4. Inside the site.yml file, import common.yml playbook.

```
ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ sudo vi site.yml
```

```
---
- hosts: all
- import_playbook: ../static-assignments/common.yml
```

5. Install tree so that the tree structure should now look like this.

```
ubuntu@ip-172-31-40-224:~/ansible-config/roles$ sudo apt install tree -y
```

```
ubuntu@ip-172-31-40-224:~$ tree ansible-config/
ansible-config/
├── README.md
├── ansible-config
│   ├── README.md
│   ├── inventory
│   │   ├── dev.yml
│   │   ├── prod.yml
│   │   ├── staging.yml
│   │   └── uat.yml
│   ├── playbooks
│   │   └── playbooks
│   │       └── site.yml
│   └── static-assignments
│       └── common.yml
```

6. Run the ansible-playbook command against the dev environment. I will need to apply some tasks to the dev servers and wireshark is already installed. I created another playbook under static-assignments and named it common-del.yml. In this playbook, I configured the deletion of Wireshark utility.

```
ubuntu@ip-172-31-40-224:~/ansible-config/static-assignments$ touch common-del.yml
ubuntu@ip-172-31-40-224:~/ansible-config/static-assignments$ sudo vi common-del.yml
```

```
---
- name: update web, nfs and db servers
  hosts: webservers, nfs, db
  remote_user: ec2-user
  become: yes
  become_user: root
  tasks:
  - name: delete wireshark
    yum:
      name: wireshark
      state: removed

- name: update LB server
  hosts: lb
  remote_user: ubuntu
  become: yes
  become_user: root
  tasks:
  - name: delete wireshark
    apt:
      name: wireshark-qt
      state: absent
      autoremove: yes
      purge: yes
      autoclean: yes
~
```

7. Update site.yml with - import_playbook: ../static-assignments/common-del.yml instead of common.yml and run it against dev servers:

```
ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ sudo vi site.yml
```

```
---
: all
- import_playbook: ../static-assignments/common-del.yml
```

8. Run the below and then make sure that Wireshark is deleted on all the servers by running Wireshark --version

cd /home/ubuntu/ansible-config/

ansible-playbook -i inventory/dev.yml playbooks/site.yml

```
ubuntu@ip-172-31-40-224:~/ansible-config$ ansible-playbook -i invent
ory/dev.yml playbooks/site.yml

PLAY [all] ********************************************************
************

TASK [Gathering Facts] ********************************************
************
```

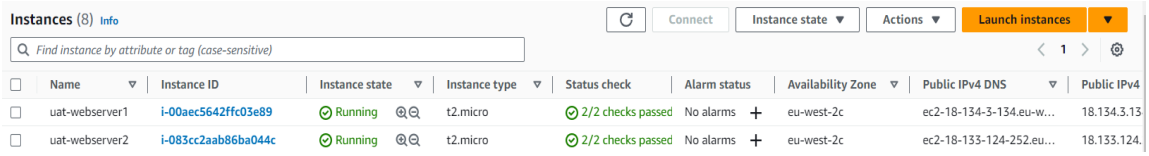
```
ubuntu@ip-172-31-40-224:~/ansible-config$ wireshark --version

Command 'wireshark' not found, but can be installed with:

sudo apt install wireshark-qt
```

Now that we have used the import_playbooks module, next is install or delete packages on multiple servers with just one command.

Step 3 – Configure UAT Webservers with the role 'Webserver'

Now we have a clean dev environment, it is time to configure 2 new Web Servers as uat.

1. Launch 2 new EC2 instances using RHEL 8 image.



2. Create a directory called roles relative to the playbook file or in /etc/ansible/ directory. Use an Ansible utility called ansible-galaxy inside the ansible-config/roles directory (you need to create a roles directory upfront).

   mkdir roles
   cd roles
   ansible-galaxy init webserver

```
ubuntu@ip-172-31-40-224:~/ansible-config$ mkdir  roles
ubuntu@ip-172-31-40-224:~/ansible-config$ ls
README.md  inventory  playbooks  roles  static-assignments
```

```
ubuntu@ip-172-31-40-224:~/ansible-config$ cd roles
ubuntu@ip-172-31-40-224:~/ansible-config/roles$ ansible-galaxy init webserver
- Role webserver was created successfully
ubuntu@ip-172-31-40-224:~/ansible-config/roles$
```

3. The entire folder structure should look like below before and after removing unnecessary directories and files.

```
ubuntu@ip-172-31-40-224:~/ansible-config/roles$ tree webserver/
webserver/
├── README.md
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
ubuntu@ip-172-31-40-224:~/ansible-config/roles$
```

```
ubuntu@ip-172-31-40-224:~/ansible-config/roles$ tree webserver/
webserver/
├── README.md
├── defaults
│   └── main.yml
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
└── templates

5 directories, 5 files
```

4. Update your inventory ansible-config/inventory/uat.yml file with IP addresses of the 2 UAT Web servers. This must be done through ssh-agent to ssh into the Jenkins-Ansible instance just as I have done in project 11;

   [uat-webservers]
   <Web1-UAT-Server-Private-IP-Address> ansible_ssh_user='ec2-user'
   <Web2-UAT-Server-Private-IP-Address> ansible_ssh_user='ec2-user'

```
ubuntu@ip-172-31-40-224:~/ansible-config/ansible-config/inventory$ sudo vi uat.yml
```

```
[uat-webservers]
172.31.5.75 ansible_ssh_user='ec2-user'
172.31.2.13 ansible_ssh_user='ec2-user'
```

5. On both uat webservers to allow ssh, add the pub key and then ssh from the Jenkins-Ansible server (ubuntu) to the uat server (RHEL).

```
[ec2-user@ip-172-31-2-13 ~]$ cd .ssh
[ec2-user@ip-172-31-2-13 .ssh]$ sudo vi authorized_keys
[ec2-user@ip-172-31-2-13 .ssh]$
```

```
ubuntu@ip-172-31-40-224:~$ ssh -A ec2-user@18.133.124.252
The authenticity of host '18.133.124.252 (18.133.124.252)' can't be established.
ECDSA key fingerprint is SHA256:CXvPXLGhysL4Ty0jRbyaWAJBnSzJBbtj5CC8k1TCkHg.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '18.133.124.252' (ECDSA) to the list of known hosts.
Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Mon Jul 10 22:33:33 2023 from 83.137.6.232
[ec2-user@ip-172-31-2-13 ~]$
```

6. Ping the uat servers from the Jenkins-Ansisible server to ensure they respond correctly.

```
ubuntu@ip-172-31-40-224:~/ansible-config-artifact/inventory$ ansible all -m ping
[DEPRECATION WARNING]: Distribution rhel 9.2 on host 172.31.18.150 should use
/usr/libexec/platform-python, but is using /usr/bin/python for backward compatibility with prior
Ansible releases. A future Ansible release will default to using the discovered platform python for
 this host. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
172.31.18.150 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
[DEPRECATION WARNING]: Distribution rhel 9.2 on host 172.31.26.195 should use
/usr/libexec/platform-python, but is using /usr/bin/python for backward compatibility with prior
Ansible releases. A future Ansible release will default to using the discovered platform python for
 this host. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
172.31.26.195 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

7. In /etc/ansible/ansible.cfg file uncomment roles_path string and provide a full path to your roles directory roles_path = /home/ubuntu/ansible-config/roles, so Ansible could know where to find configured roles.

```
# config file for ansible -- https://ansible.com/
# ===============================================

# nearly all parameters can be overridden in ansible-playbook
# or with command line flags. ansible will read ANSIBLE_CONFIG,
# ansible.cfg in the current working directory, .ansible.cfg in
# the home directory or /etc/ansible/ansible.cfg, whichever it
# finds first

[defaults]

# some basic default values...

inventory      = /home/ubuntu/ansible-config-artifact/inventory
#library        = /usr/share/my_modules/
```

8. In /etc/ansible/ansible.cfg file uncomment roles_path string and provide a full path to your roles directory

```
ubuntu@ip-172-31-40-224:~$ sudo vi /etc/ansible/ansible.cfg
```

```
# additional paths to search for roles in, colon separated
roles_path    =/home/ubuntu/ansible-config/roles
```

9. Start adding some logic to the webserver role. Go into the tasks directory, and within the main.yml file, start writing configuration tasks to do the following:

- Install and configure Apache (httpd service)
- Clone Tooling website from GitHub https://github.com/<your-name>/tooling.git.
- Ensure the tooling website code is deployed to /var/www/html on each of the 2 UAT Web servers.
- Make sure httpd service is started

Your main.yml may consist of the following tasks:

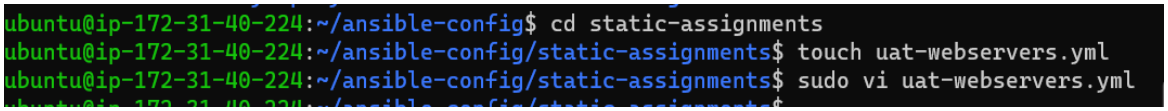
```
---
- name: install apache
  become: true
  ansible.builtin.yum:
    name: "httpd"
    state: present

- name: install git
  become: true
  ansible.builtin.yum:
    name: "git"
    state: present

- name: clone a repo
  become: true
  ansible.builtin.git:
    repo: https://github.com/<your-name>/tooling.git
```

```
  dest: /var/www/html
  force: yes

- name: copy html content to one level up
 become: true
 command: cp -r /var/www/html/html/ /var/www/

- name: Start service httpd, if not started
 become: true
 ansible.builtin.service:
  name: httpd
  state: started

- name: recursively remove /var/www/html/html/ directory
 become: true
 ansible.builtin.file:
  path: /var/www/html/html
  state: absent
```

Step 4 – Reference 'Webserver' role

1. Within the static-assignment folder, create a new assignment for uat-webservers known as uat-webservers.yml. This is where you will reference the role.

   ```
   —
   - hosts: uat-webservers
    Roles:
     - webserver
   ```



```
---
- hosts: uat-webservers
  roles:
      - webserver
~
```

2. Remember that the entry point to our ansible configuration is the site.yml file. Therefore, you need to refer to the uat-webservers.yml role inside site.yml. So, we should have this in site.yml

```
ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ sudo vi site.yml
```

```
---
: all
- import_playbook: ../static-assignments/common-del.yml


 ---
- hosts: uat-webservers
- import_playbook: ../static-assignments/uat-webservers.yml
```

Step 5 – Commit & Test

1. Commit your changes, create a Pull Request and merge them to the main branch, make sure the webhook triggered two consequent Jenkins jobs, they ran successfully and copied all the files to your Jenkins-Ansible server into /home/ubuntu/ansible-config/ directory.

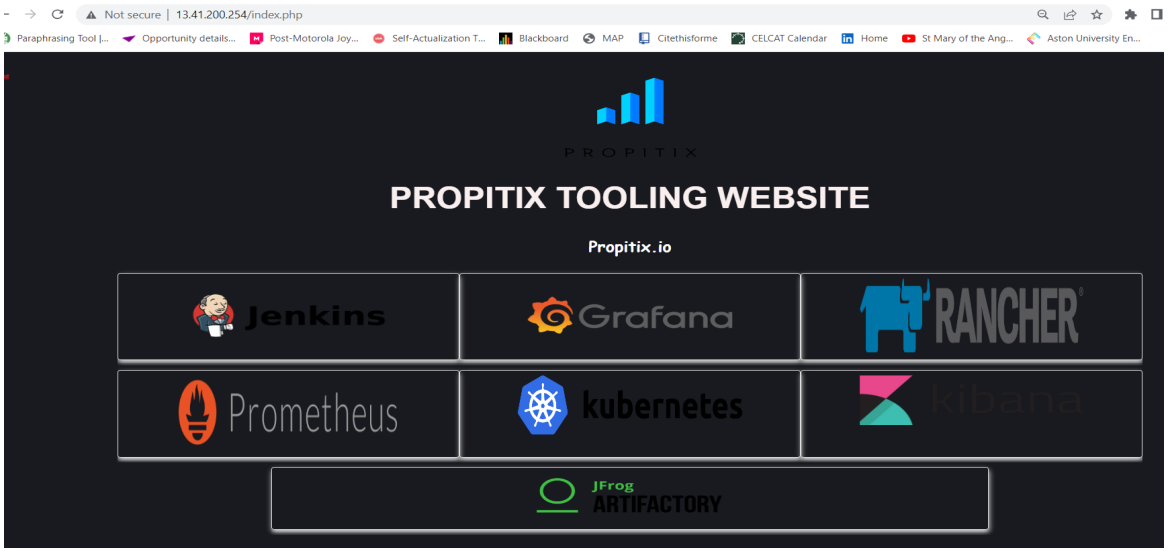
```
ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ git status
On branch refactor
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   site.yml
        modified:   ../roles/webserver/tasks/main.yml
        modified:   ../static-assignments/uat-webservers.yml

ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ git add .
ubuntu@ip-172-31-40-224:~/ansible-config/playbooks$ git commit -m "made multiple change
s"
[refactor 8b87e76] made multiple changes
 3 files changed, 46 insertions(+), 2 deletions(-)
 rewrite roles/webserver/tasks/main.yml (86%)
```

2. Run the playbook against your uat inventory and see what happens: sudo ansible-playbook -i /home/ubuntu/ansible-config/inventory/uat.yml /home/ubuntu/ansible-config/playbooks/site.yml

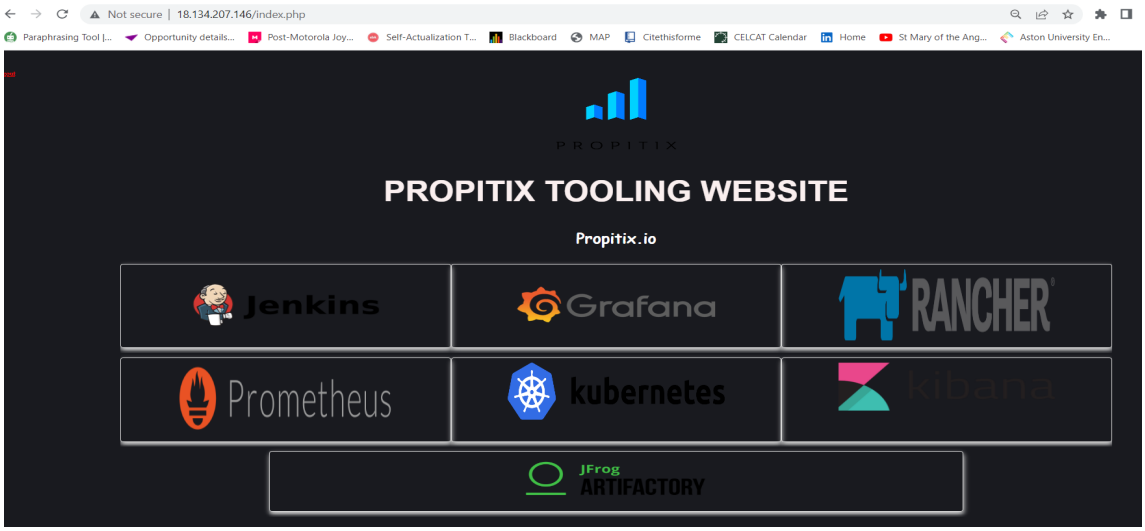
```
TASK [Gathering Facts] ***********************************************************************************************************************
ok: [172.31.34.12]
ok: [172.31.34.164]

PLAY [uat-webservers] ************************************************************************************************************************

TASK [Gathering Facts] ***********************************************************************************************************************
ok: [172.31.34.12]
ok: [172.31.34.164]

TASK [webserver : install apache] ************************************************************************************************************
changed: [172.31.34.12]
changed: [172.31.34.164]

TASK [webserver : install git] ***************************************************************************************************************
changed: [172.31.34.12]
changed: [172.31.34.164]

TASK [webserver : clone a repo] **************************************************************************************************************
changed: [172.31.34.164]
changed: [172.31.34.12]

TASK [webserver : copy html content to one level up] *****************************************************************************************
changed: [172.31.34.164]
changed: [172.31.34.12]

TASK [webserver : Start service httpd, if not started] ***************************************************************************************
changed: [172.31.34.12]
changed: [172.31.34.164]

TASK [webserver : recursively remove /var/www/html/html/ directory] **************************************************************************
changed: [172.31.34.164]
changed: [172.31.34.12]

PLAY RECAP ***********************************************************************************************************************************
172.31.34.12               : ok=9    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.34.164              : ok=9    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-40-224:/etc/ansible$
```

3. You should be able to see both of your UAT Web servers configured and you can try to reach them from your browser http://<Web1-UAT-Server-Public-IP-or-Public-DNS-Name>/index.php

Webserver 1



Webserver 2



I have successfully deployed and configured UAT Web Servers using Ansible imports and roles. My architecture should now look like this.