DevOps Project: One-Click Jenkins Pipeline Deployment

Objective

The objective of this project is to create a fully automated DevOps pipeline using Jenkins, containerized with Docker. The pipeline will automate the provisioning of an Ubuntu virtual machine on Azure using Terraform, install an Apache web server on the VM using Ansible, deploy a static web application to the server, and execute all steps through a single Jenkins pipeline with one click.

Technology Stack

The project incorporates several tools to achieve full automation:
- Docker is used to host Jenkins in a containerized environment.
- Jenkins is the CI/CD tool that automates the entire workflow.
- Terraform is responsible for provisioning the Azure virtual machine.
- Ansible is used to configure the VM and deploy the static web application.
- Azure provides the cloud infrastructure where the VM is hosted.
- Vagrant to provision ubuntu VM for hosting Jenkins container
- Git is used to store all infrastructure-as-code files, Ansible playbooks, Jenkinsfile, and the application code.

Implementation

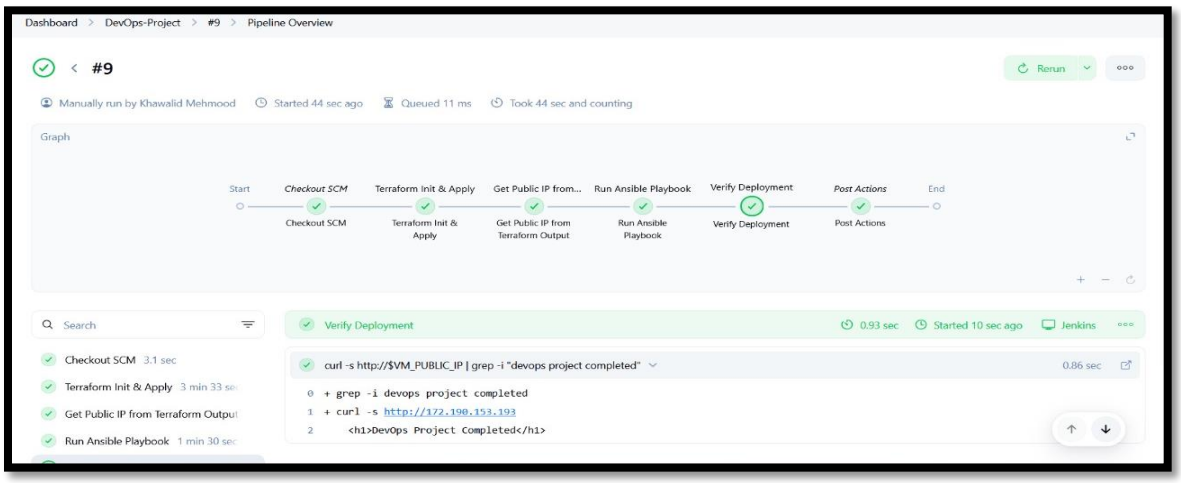
The implementation process involved several sequential steps to achieve a fully automated CI/CD pipeline from infrastructure provisioning to deployment:

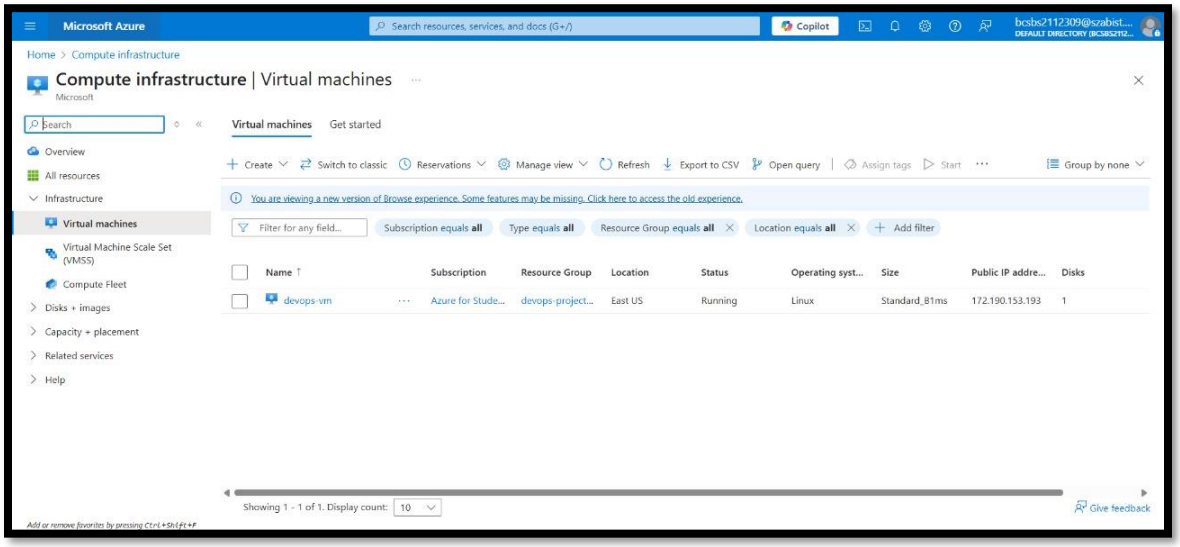
- Initialized GitHub repo with Jenkinsfile, Terraform, app, and Ansible files.
- Installed Docker on Ubuntu vm provisioned through vagrant and launched Jenkins container with Terraform mounted.
- Installed required Jenkins plugins: Git, Pipeline, and SSH Agent
- Installed Git and Ansible inside the Jenkins container manually.
- Created SSH key pair for azure vm and added the private key to Jenkins credentials so Ansible can use to setup web server on that vm.
- Added Azure credentials in Jenkins credentials manager for secure access in Jenkinsfile to provision infrastructure on Azure using service principal
- Allowed SSH (22) and HTTP (80) ports in the NSG for remote and web access.
- Used Terraform output block to expose VM public IP for Ansible.
- Configured Jenkins pipeline to run Terraform and retrieve VM IP dynamically.
- Generated a dynamic Ansible inventory with the VM IP inside the pipeline.
- Executed Ansible playbook from Jenkins to install Apache and deploy HTML.

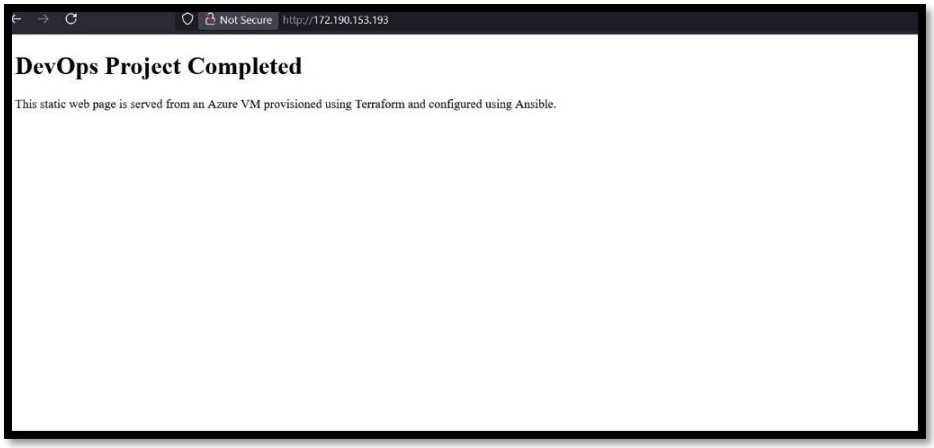
- Verified web server deployment via VM public IP in browser and curl

Output

After the Jenkins pipeline executes successfully, a virtual machine is created on Azure with Apache installed, and the static website is deployed to it. The application should be accessible by entering the public IP of the VM in a browser.






Not Secure
http://172.190.153.193
DevOps Project Completed
This static web page is served from an Azure VM provisioned using Terraform and configured using Ansible.