Industry Based Project 2 Assignment Report

Business Challenge/Requirement

XYZ technologies is a leading online repository for downloading online courses. XYZ plans to have its repository managed such that it can have admin login and User Login modules. In the first phase team has created the user login module and has moved it to production. There is a team of developers who are working on source code creation and add it to source code management tool - git repository. When the entire source code of application is coded team has used Maven to build the source code. Once a successful build is created, it is deployed to test server for QE resources to start testing and log issues which they have found. If any bugs are detected, developers are notified with the help of a feedback loop. If no bug is encountered, then the code is deployed to pre-prod and eventually to production server for release. Upon analysis of this process, Management has identified the following flaws:

- Entire source code of application is build and then deployed to test server for testing.
- It takes lot of time for developer to get the test results
- Accumulation of bugs
- Consumes a lot of time to debug as entire source code needs to be checked
- Underutilization of resources as developers are idle until QE give bugs and QE is idle until entire source code is developed
- Software to market time is more
- Feedback mechanism was not robust

In the second phase, XYZ must create the modules such that the admin can view/add/delete Users. Further company wants to expand their repository to add more courses, add content based on most seen topic courses. Eventually, down the line company plans to provide course recommendations based on various user factors to provide good learning experience to the users. In order to achieve the company goals, Management has decided to leverage DevOps model and overcome above mentioned challenges.

The Goal of the Project

Below are some of the high-level goals of this project:

1. Develop a continuous integration pipeline in Jenkins to compile, test and package the code
2. CICD pipeline to resolve business challenges by XYZ technologies.
3. Real-time understanding and hands-on with Git, Jenkins, Ansible, Docker, Kubernetes, and AWS Devops services

AUTHORED BY STUDENT: Christian Onyeukwu (Canada)

System Requirement

Hardware Required

- Student Local Machine (PC → MacBook Air M2)
- Virtual Machine (Edureka Cloud Lab / Amazon EC2)

Software Required :

- Java (for source code implementation)
- maven (build library for maven based project → java)
- Git (version control system)
- jenkins (ci/cd automation tool)
- Docker (containerization tool).
- Ansible (configuration management tool → YAML scripting or playbooks)
- Kubernetes (automated management, scaling, running containers across clusters machines)
- Grafana (visualization and monitoring tool to create an interactive dashboard from source like Prometheus)
- Prometheus (monitoring and alerting tool for storing, collecting and alerting on the system metrics)

Problem Statement/Tasks

We need to develop a CICD pipeline to automate the software development, testing, package, deploy reducing the time to market of app and ensuring good quality service is experienced by end users. In this project we need to

1. Push the code to out github repository
2. Create a continuous integration pipeline using Jenkins to compile, test and package the code present in git hub

3. Write docker file to push the war file to tomcat server
4. Integrate docker with Ansible and write playbook
5. Deploy artifacts to Kubernetes cluster
6. Monitor resources using Prometheus

Solution Report

High Level Design Architecture

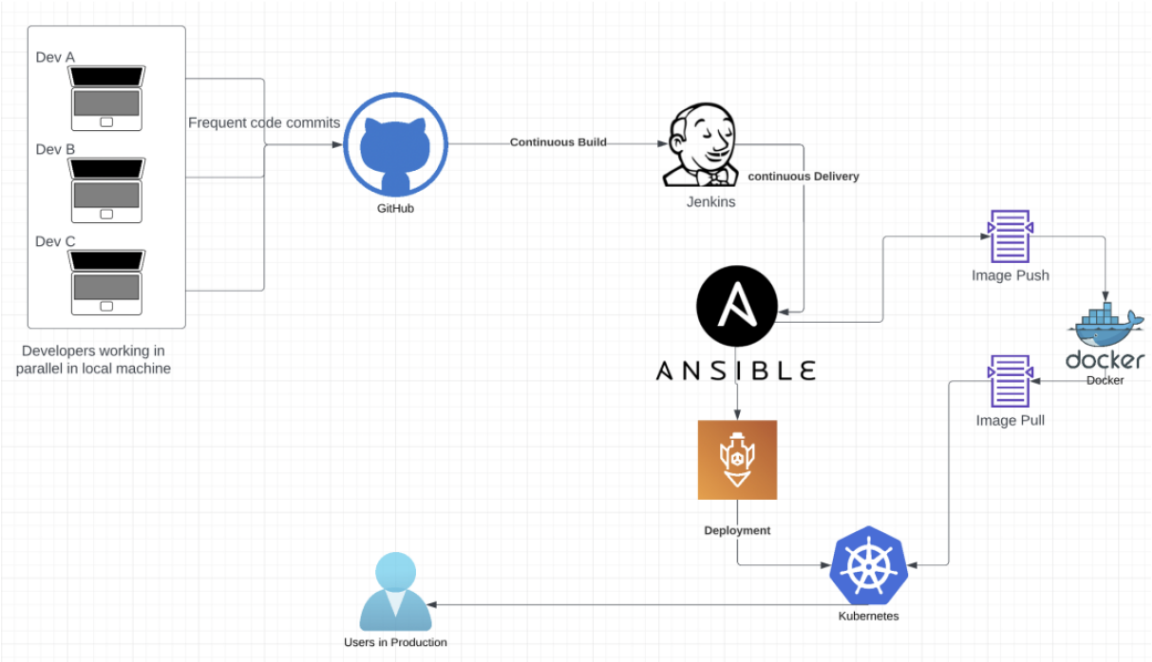

Figure 1: → Design Architecture

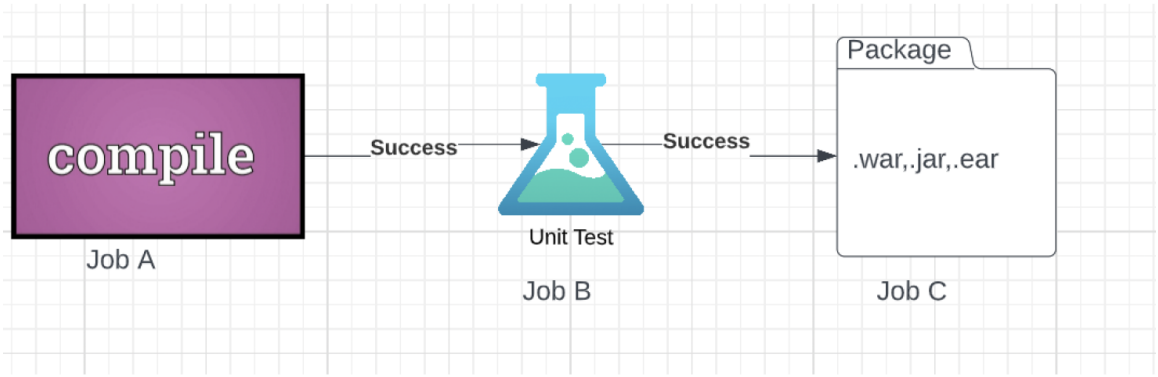

Figure 2: → Continuous Integration Architecture

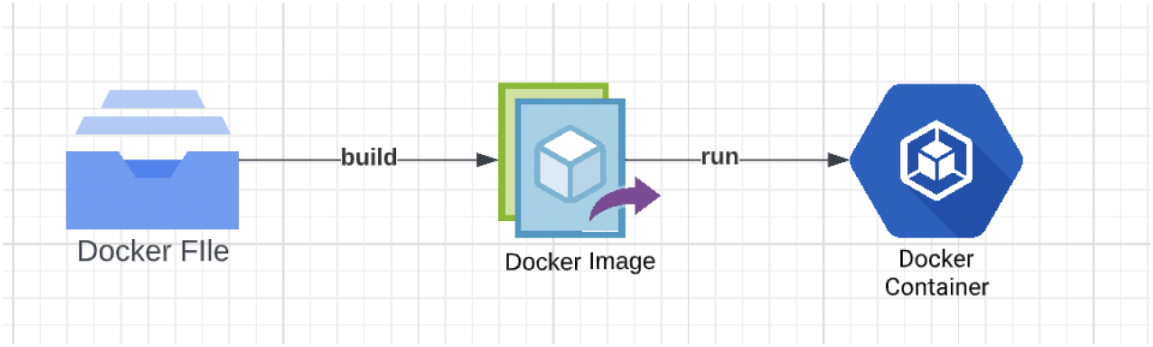

Figure 3: → Docker Containerization Architecture

Task 1: Processes

Step 1: Retrieve source file

The zip file of the source file for the project was downloaded from my classroom to my local machine, unzipped, and moved to the project folder I created on my machine.

Step 2:Prepare source file in local machine

Opened the file from IntelliJ IDE as a Maven-based project, rebuild

with maven command (*mvn clean install*), installed apache tomcat in my local machine to view the webapp locally at. See image
→ *http://localhost:8080/XYZTechnologies-1.0/* (optinal)

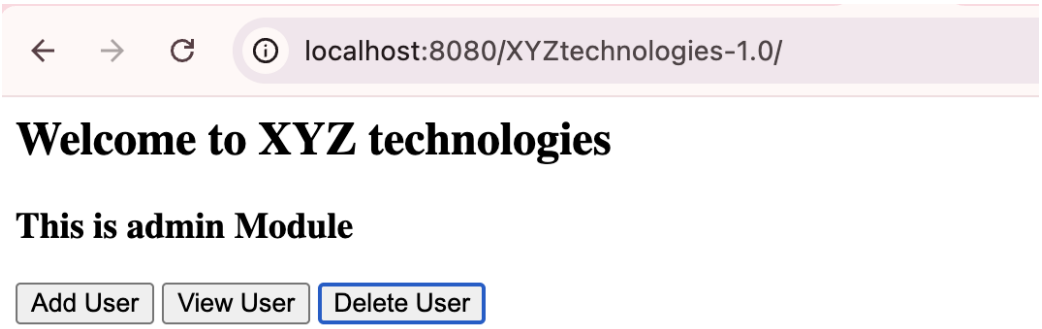

Figure 4: → XYZ Technology landing webpage snapshot

Step 3:Push source file to VCS or Branch for the DevOps

Initialized a git repository, logged in to my github account and created a new repository for this assignment, and then go back to my local terminal window and add, commit and push all my source code into my github

Step 4:Accessing the Virtual Machine (Edureka Cloud Lab)

Logged in to my Cloud Lab, accessed my virtual desktop via *Master Web Desktop* access link. After that I ran some command line code to confirm that all the required tools such as Java, maven, git, docker, kubernetes, jenkins, ansible, grafana and prometheus are pre-installed in the virtual lab. See GitHub repository link:

Figure 5: → Cloning Project2

→ *https://github.com/christianRibig5/industry-based-devops-proj2.git*

Step 5: Building codes using maven commands

Created a folder on desktop in my lab (project-required-file-v2), cloned the file from my github and build with command

Figure 6: → */opt/maven/bin/mvn compile*:Compile Interface snapshot

Figure 7: → */opt/maven/bin/mvn test*:Test Interface snapshot

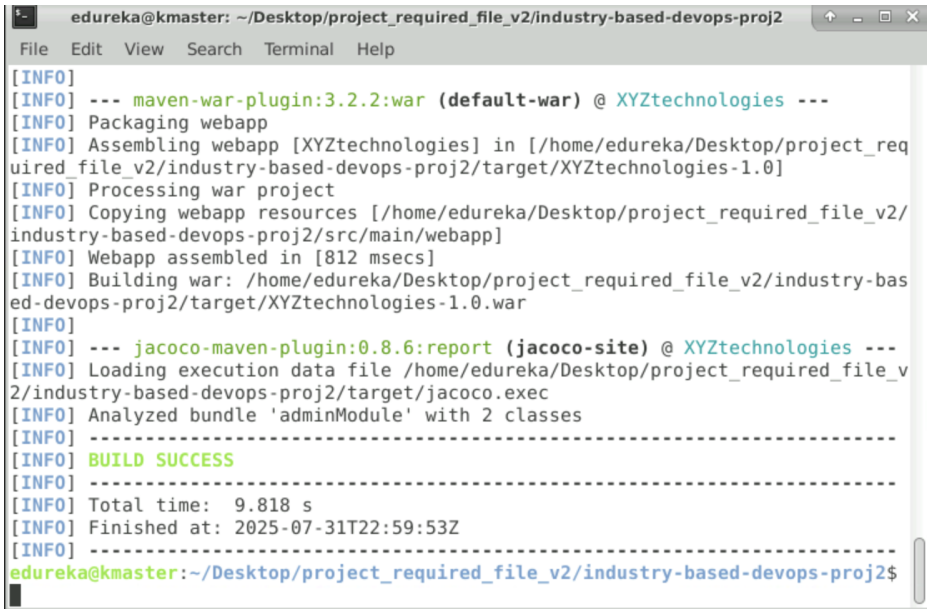

Figure 8: →*/opt/maven/bin/mvn package*:Package Interface snapshot

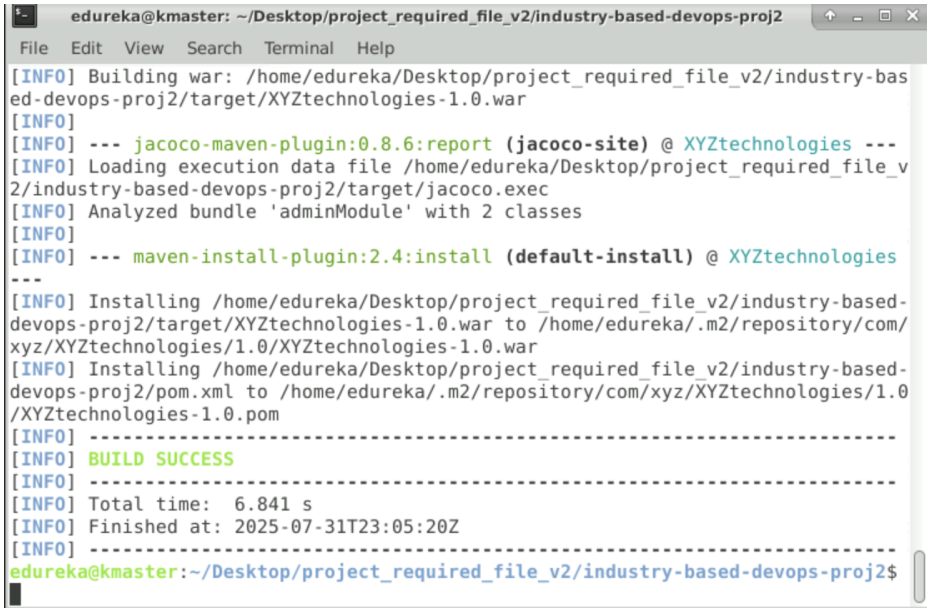

Figure 9: → */opt/maven/bin/mvn clean install*:clean install Interface snapshot

Task 2.Jenkins Setup CICD Pipeline

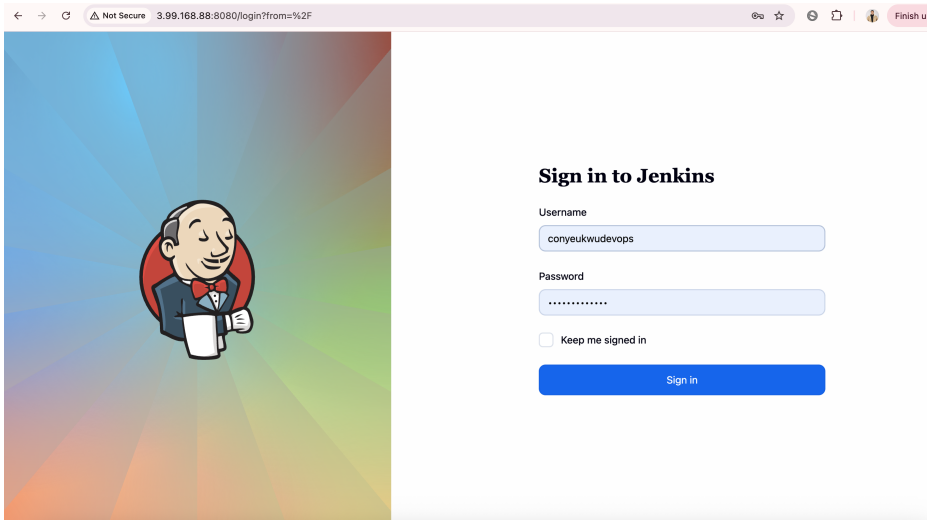

Figure 10: → *http:3.99.168.88:8080/*:jenkins Interface snapshot

Note that other stages of jekins setup has been done in assignment 1, such as, installation of required plugins, and global tool configuration.

AUTHORED BY STUDENT: Christian Onyeukwu (Canada)

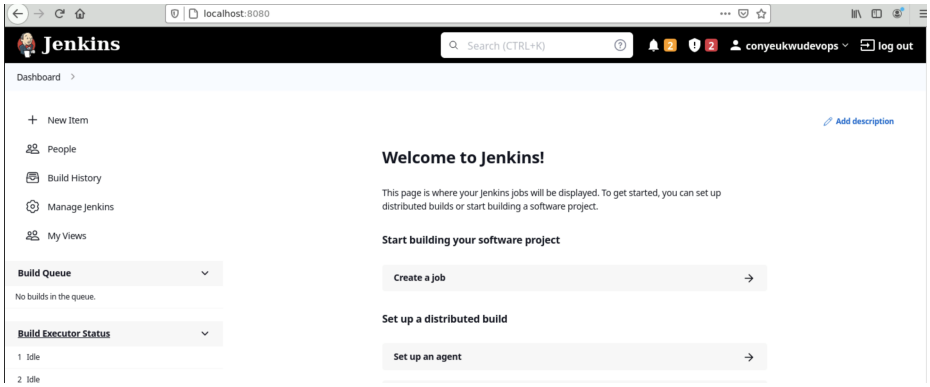

Figure 11: → Welcome Jenkins Interface snapshot

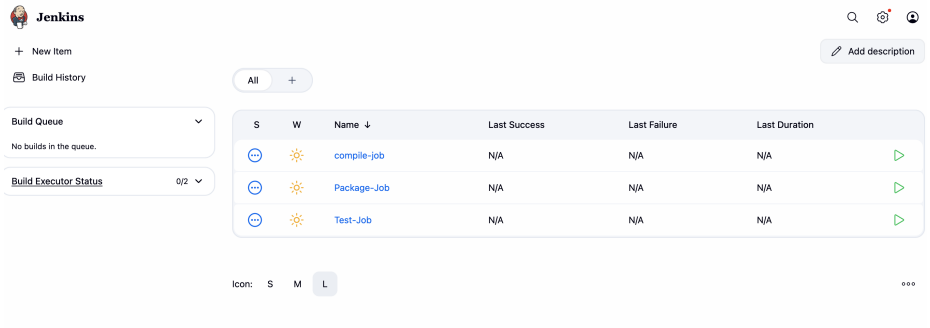

Figure 12: → 3 Jobs schedules Interface snapshot

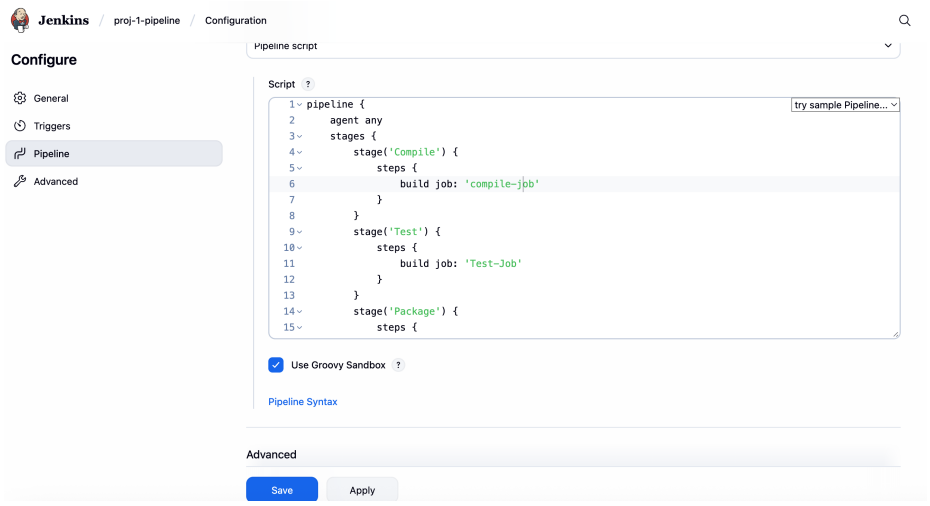

Figure 13: → Pipeline Code

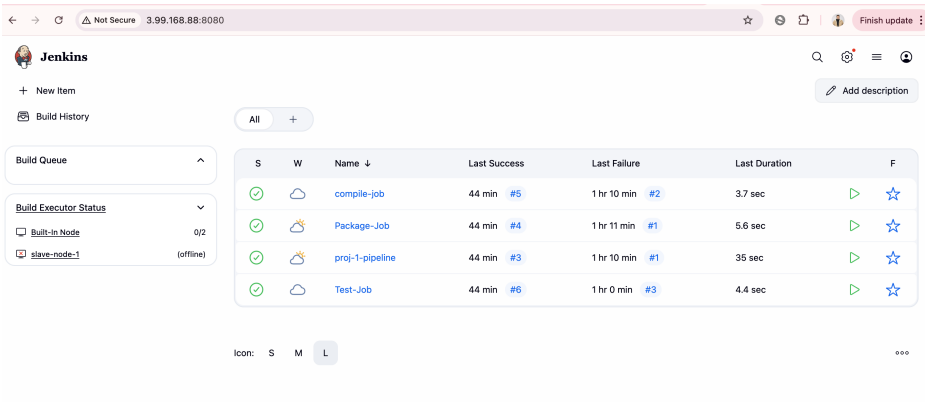

Figure 14: → Execution Interface

Task 2.3: Set up slave node to distribute the tasks in pipeline

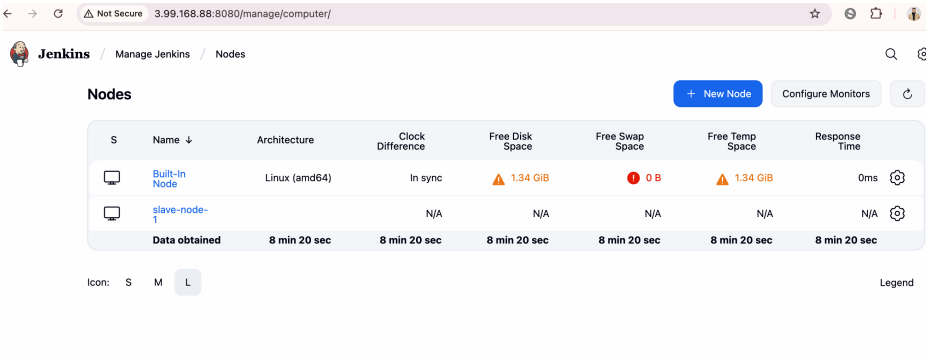

Figure 15: → Slave Node

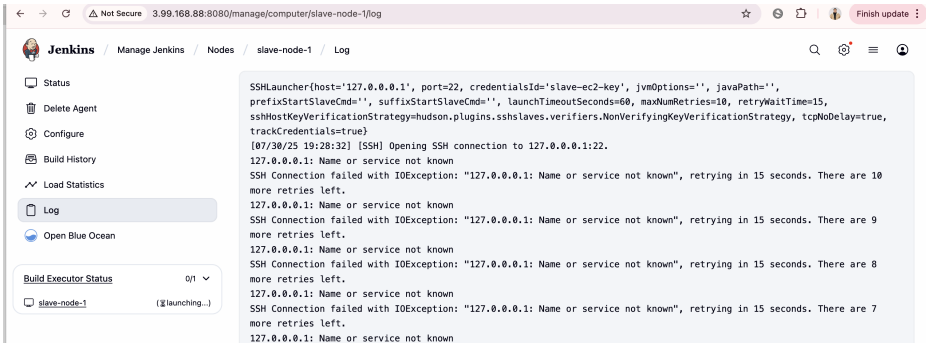

Figure 16: → Lunching Slave Node

Task 3: Write Dockerfile to create Image and container on Docker host

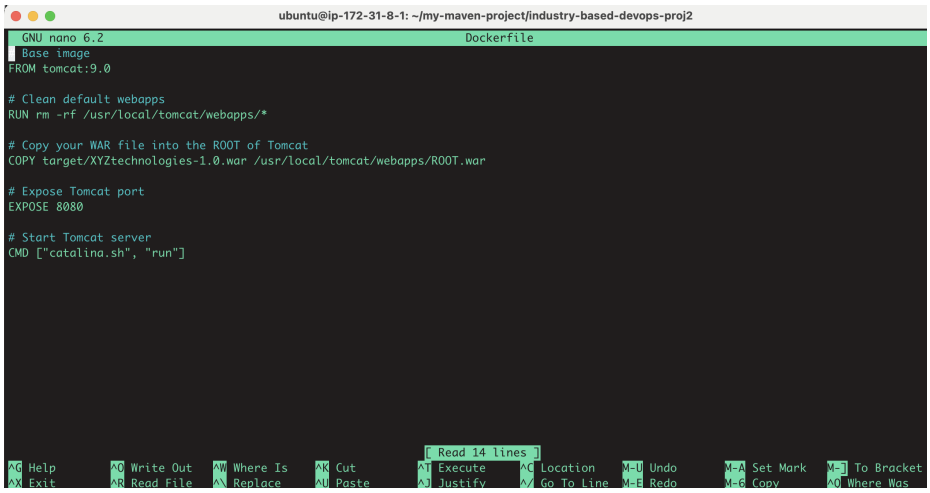

Figure 17: → Docker File Script

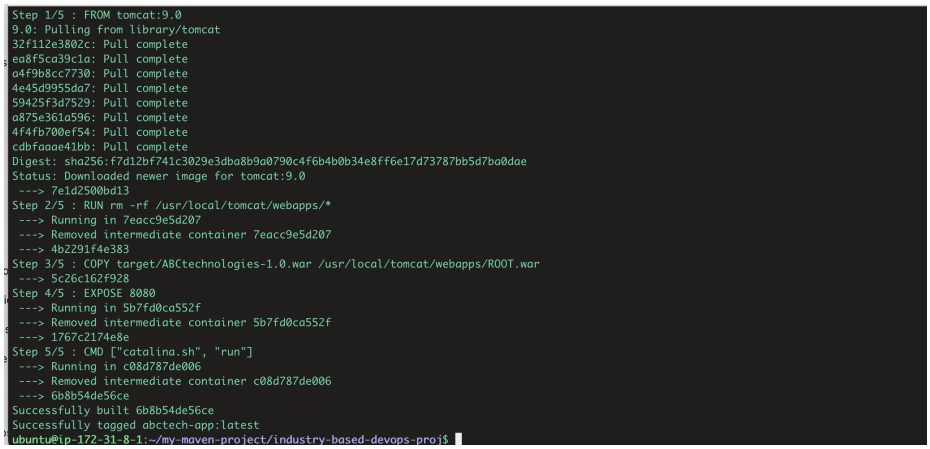

Figure 18: → Docker Script Execution

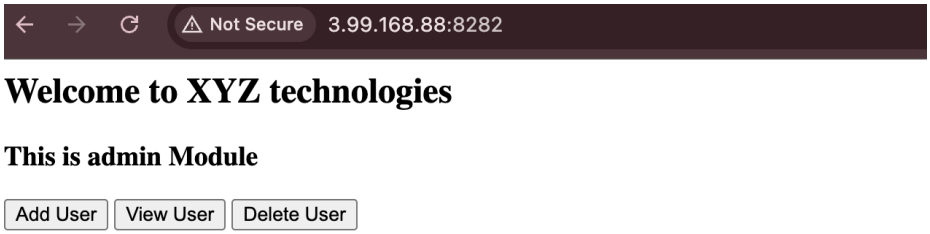

Figure 19: → Image running on Docker container snapshot

Task 4:Integrate Docker host with Ansible

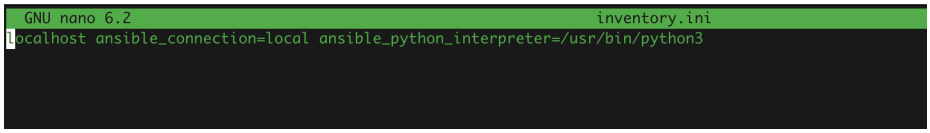

Figure 20: → 1: Deployment artifact snapshot

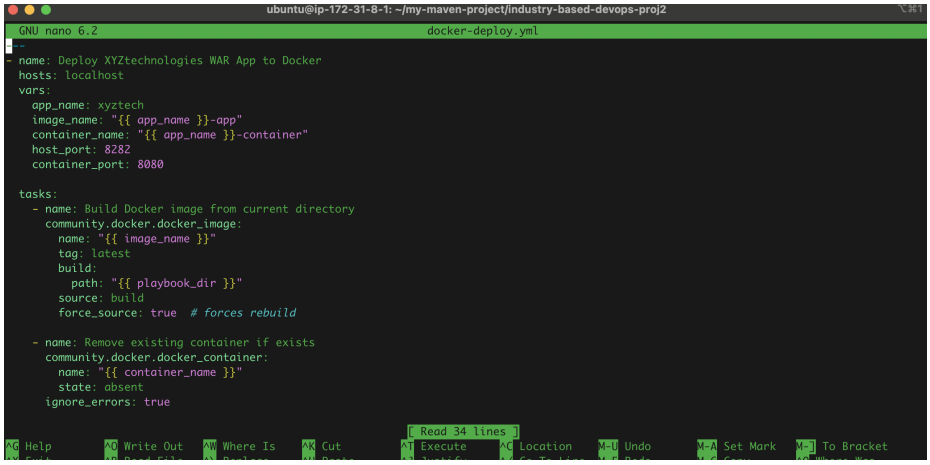

Figure 21: → 2: Ansible Playbook Script snapshot

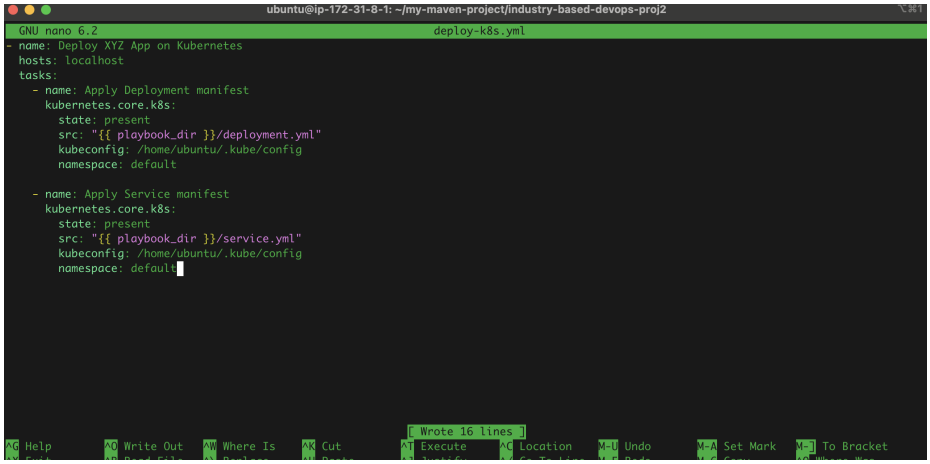

Figure 22: → 2: Deployment & Service Playbook snapshot

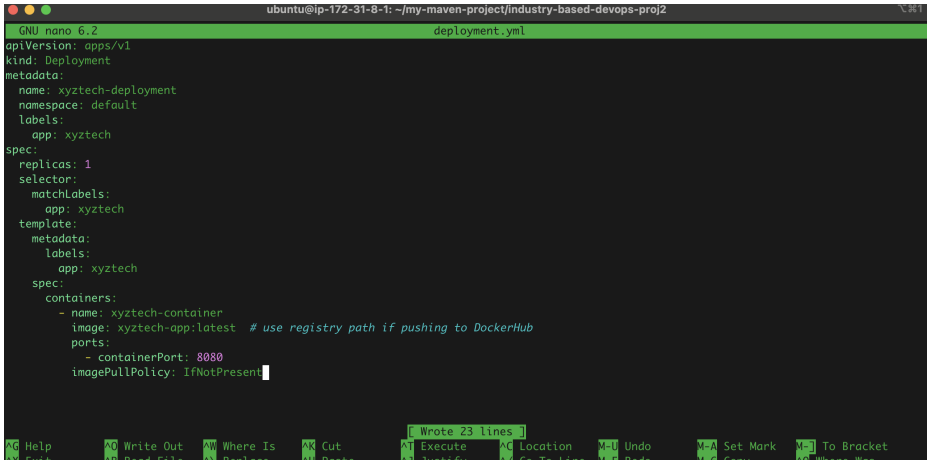

Figure 23: → 3: Kubernetes playbook script snapshot

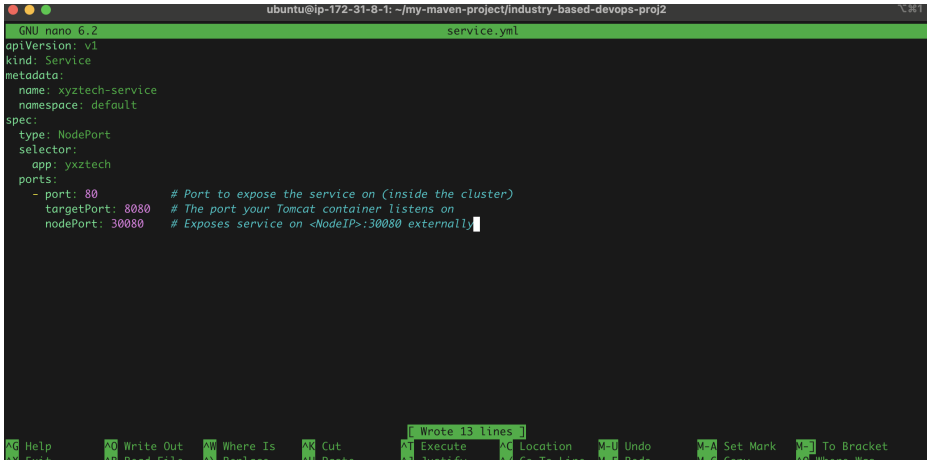

Figure 24: → 3: Service Playbook Script snapshot

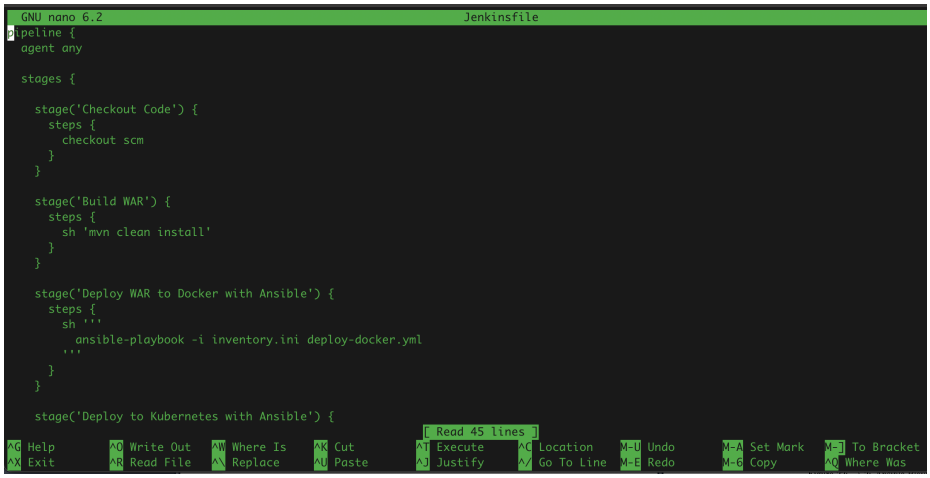

Figure 25: → 4: Pipeline Script snapshot

Task 5: Using Prometheus Monitor and Grafana Dashbord for Metrics Monitor

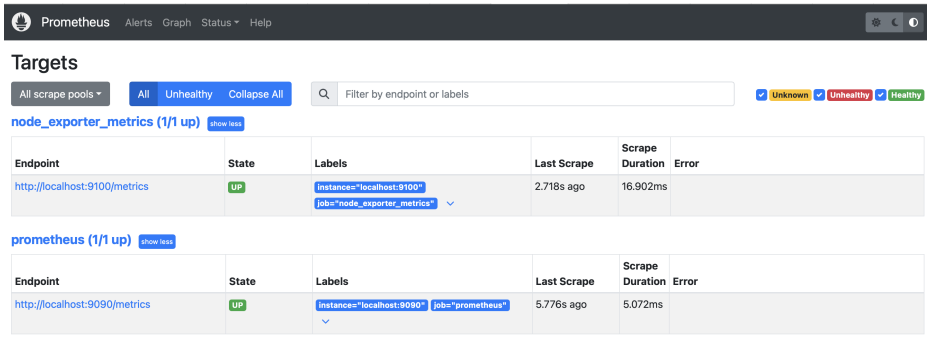

Figure 26: → 1: Prometheus Dashboard snapshot

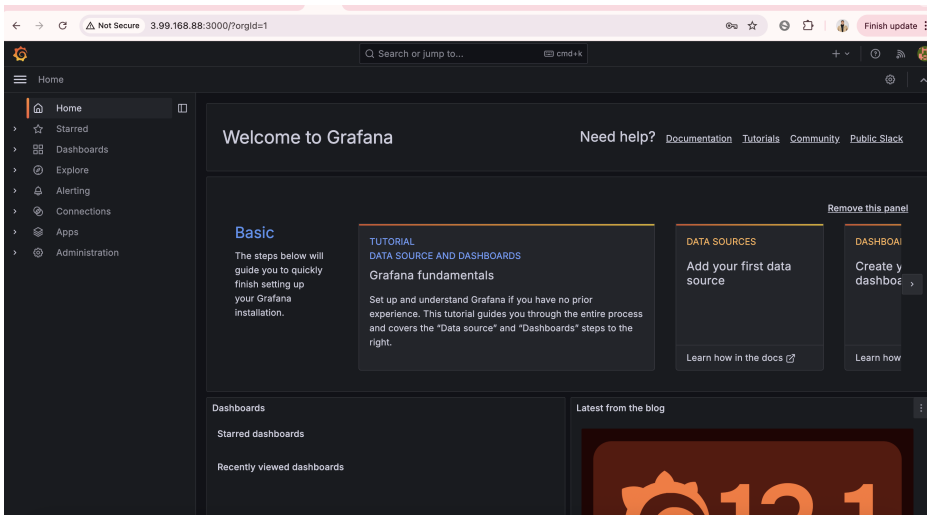

Figure 27: → 2: Grafana Dashboard snapshot

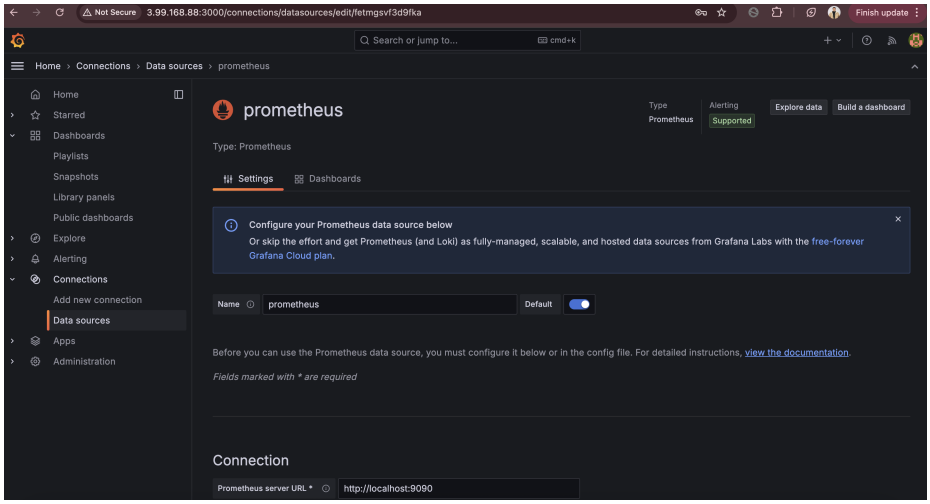

Figure 28: → 3: Added Prometheus as Data Source snapshot

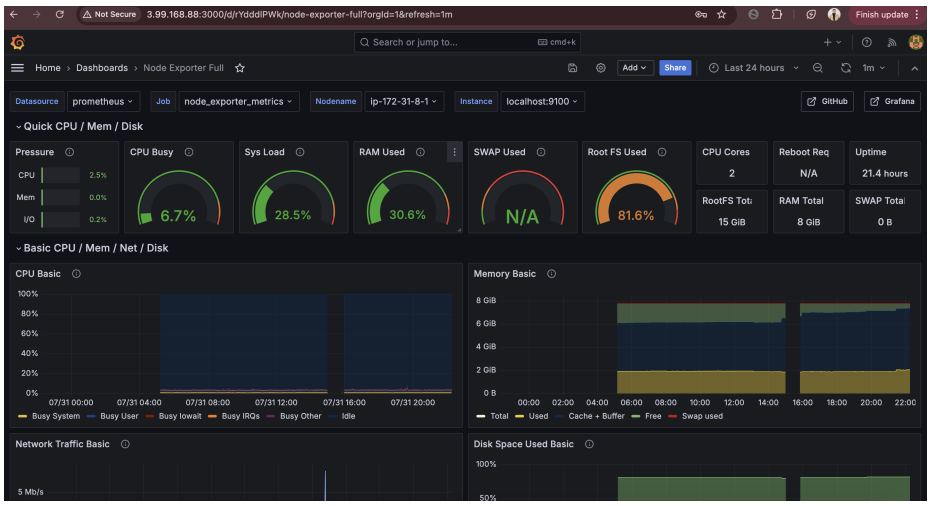

Figure 29: → 4: Monitoring Dashboard snapshot