

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

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 our 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

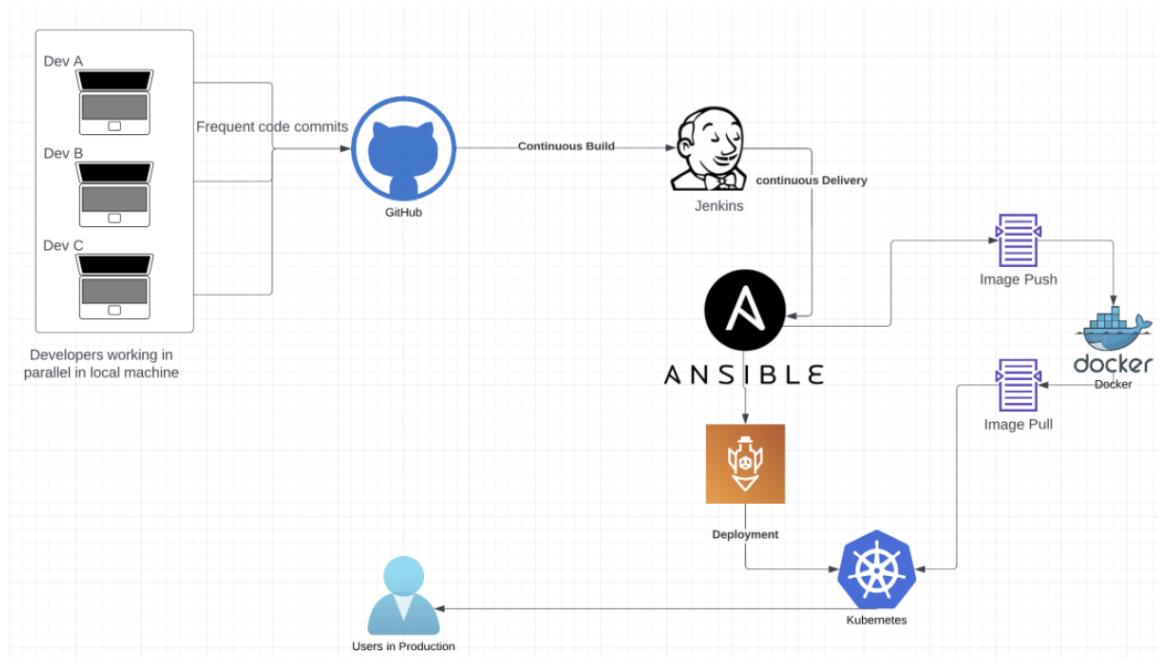


Figure 1: → Design Architecture

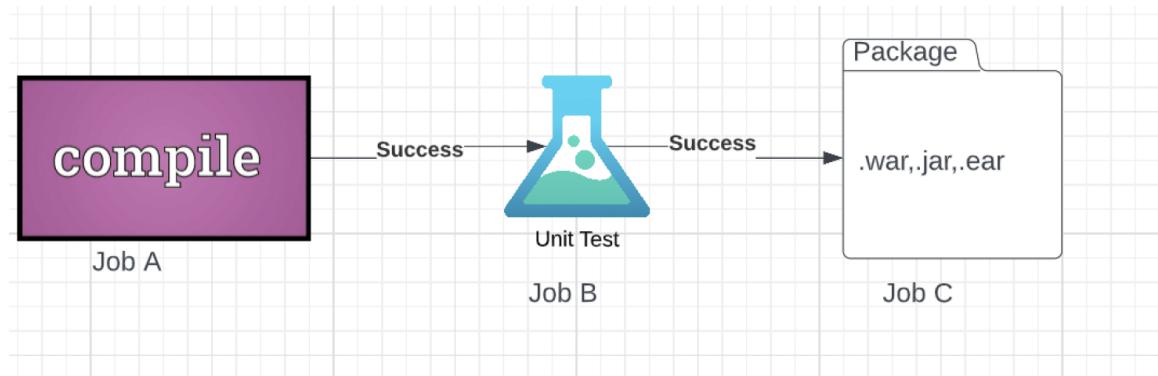


Figure 2: → Continuous Integration Architecture

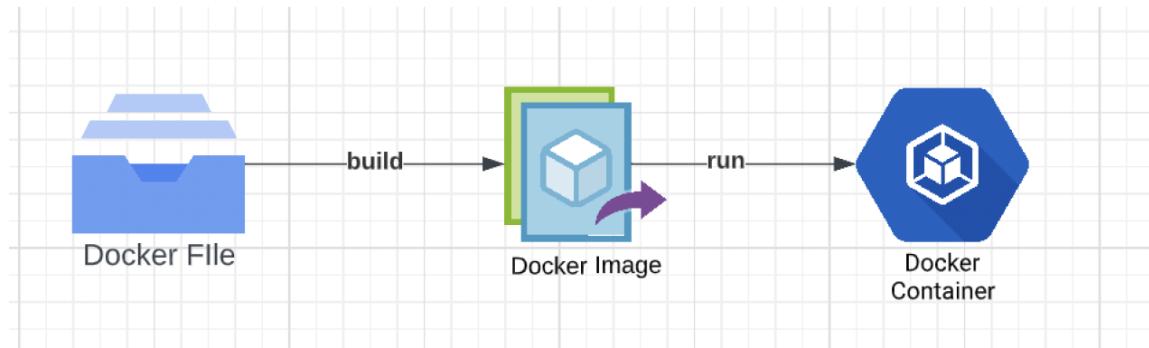


Figure 3: → Docker Containerization Architecture

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/> (optional)

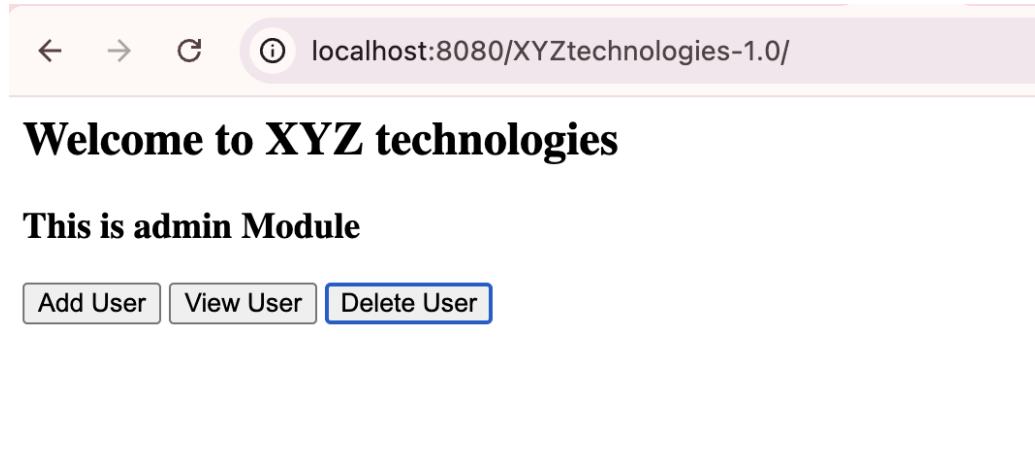


Figure 4: → XYZ Technology landing webpage snapshot

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:

```

From https://github.com/christianRibig5/industry-based-devops-proj
 * branch            main      -> FETCH_HEAD
   dc68a1a..836cccd  main      -> origin/main
Updating dc68a1a..836cccd
Fast-forward
  DevOps_Industry_Project_Ass1.pdf | Bin 0 --> 6745527 bytes
  1 file changed, 0 insertions(+), 0 deletions(-)
  create mode 100644 DevOps_Industry_Project_Ass1.pdf
ubuntu@ip-172-31-8-1:~/my-maven-project/industry-based-devops-proj$ cd ~
ubuntu@ip-172-31-8-1:~$ cd my-maven-project/
ubuntu@ip-172-31-8-1:~/my-maven-project$ mkdir industry-based-devops-proj2
ubuntu@ip-172-31-8-1:~/my-maven-project$ cd industry-based-devops-proj2
ubuntu@ip-172-31-8-1:~/my-maven-project/industry-based-devops-proj2$ cd ..
ubuntu@ip-172-31-8-1:~/my-maven-project$ rmdir industry-based-devops-proj2
ubuntu@ip-172-31-8-1:~/my-maven-project$ ls
industry-based-devops-proj
ubuntu@ip-172-31-8-1:~/my-maven-project$ git clone https://github.com/christianRibig5/industry-based-devops-proj2.git
Cloning into 'industry-based-devops-proj2'...
remote: Enumerating objects: 171, done.
remote: Counting objects: 100% (171/171), done.

```

Figure 5: → Cloning Project2

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

Step 5: Building codes using maven commands

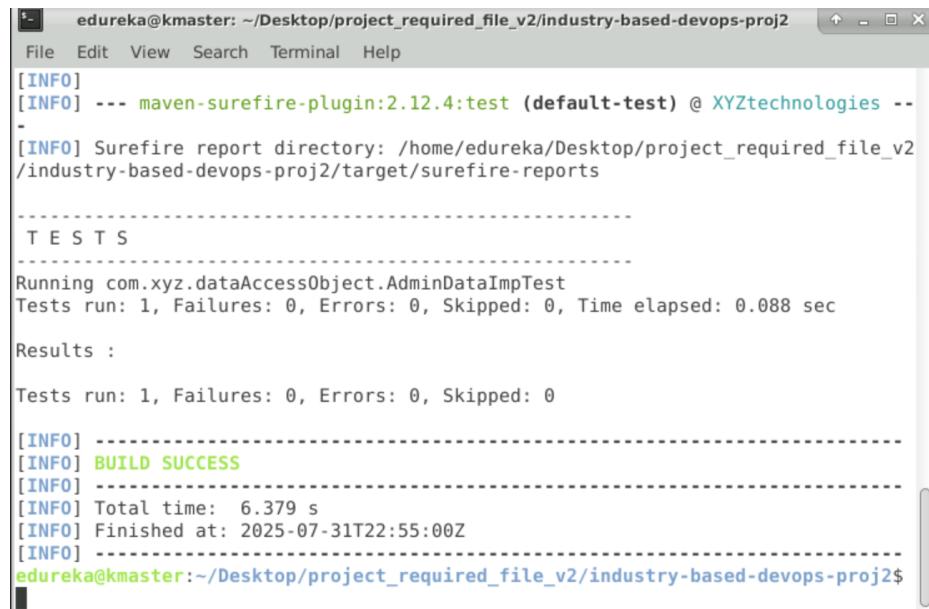
Created a folder on desktop in my lab (project-required-file-v2), cloned the file from my github and build with command

```

edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj2$ mvn compile
[INFO] argLine set to -javaagent:/home/edureka/.m2/repository/org/jacoco/org.jacoco.agent/0.8.6/org.jacoco.agent-0.8.6-runtime.jar=destfile=/home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/jacoco.exec
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ XYZtechnologies ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ XYZtechnologies ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 3 source files to /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/classes
[INFO]
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 19.453 s
[INFO] Finished at: 2025-07-31T22:49:45Z
[INFO] -----
edureka@kmaster:~/Desktop/project_required_file_v2/industry-based-devops-proj2$ 

```

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



```
edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj2$ mvn test
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ XYZtechnologies ---
[INFO] Surefire report directory: /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/surefire-reports

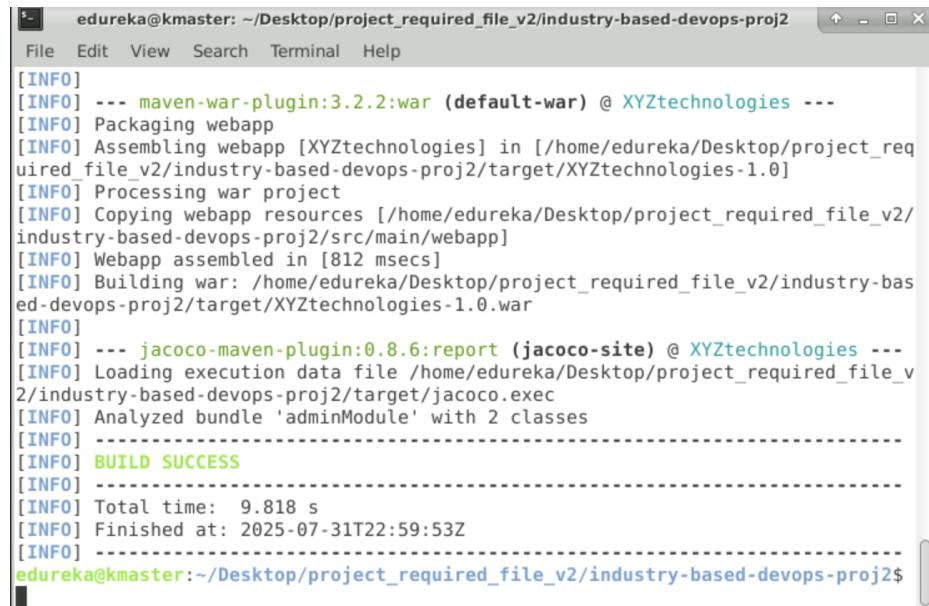
-----
T E S T S
-----
Running com.xyz.dataAccessObject.AdminDataImpTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.088 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  6.379 s
[INFO] Finished at: 2025-07-31T22:55:00Z
[INFO] -----
edureka@kmaster:~/Desktop/project_required_file_v2/industry-based-devops-proj2$
```

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



```
edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj2$ mvn package
[INFO] --- maven-war-plugin:3.2.2:war (default-war) @ XYZtechnologies ---
[INFO] Packaging webapp
[INFO] Assembling webapp [XYZtechnologies] in [/home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/XYZtechnologies-1.0]
[INFO] Processing war project
[INFO] Copying webapp resources [/home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/src/main/webapp]
[INFO] Webapp assembled in [812 ms]
[INFO] Building war: /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/XYZtechnologies-1.0.war
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.6:report (jacoco-site) @ XYZtechnologies ---
[INFO] Loading execution data file /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/jacoco.exec
[INFO] Analyzed bundle 'adminModule' with 2 classes
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  9.818 s
[INFO] Finished at: 2025-07-31T22:59:53Z
[INFO] -----
edureka@kmaster:~/Desktop/project_required_file_v2/industry-based-devops-proj2$
```

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



```
[INFO] Building war: /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/XYZtechnologies-1.0.war
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.6:report (jacoco-site) @ XYZtechnologies ---
[INFO] Loading execution data file /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/jacoco.exec
[INFO] Analyzed bundle 'adminModule' with 2 classes
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ XYZtechnologies ---
[INFO] Installing /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/target/XYZtechnologies-1.0.war to /home/edureka/.m2/repository/com/xyz/XYZtechnologies/1.0/XYZtechnologies-1.0.war
[INFO] Installing /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj2/pom.xml to /home/edureka/.m2/repository/com/xyz/XYZtechnologies/1.0/XYZtechnologies-1.0.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time:  6.841 s
[INFO] Finished at: 2025-07-31T23:05:20Z
[INFO] -----
```

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

Task 2.Jenkins Setup CICD Pipeline

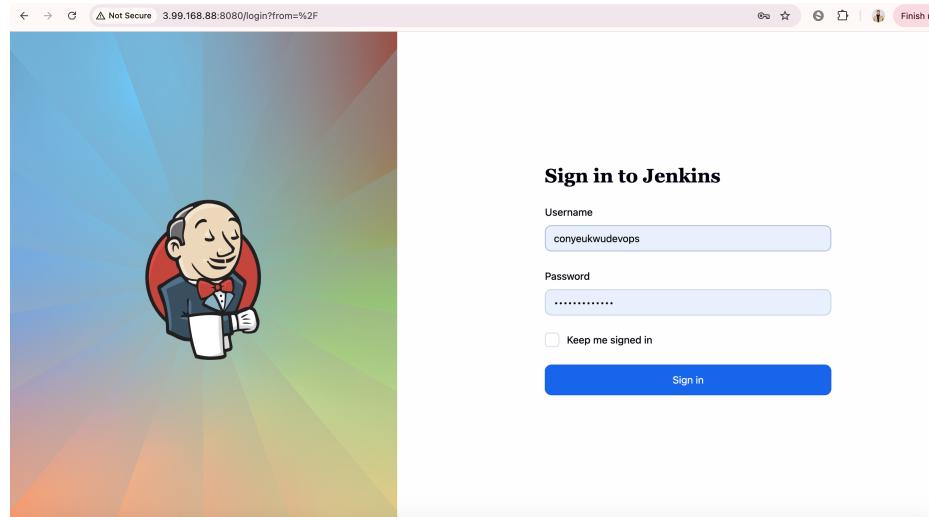


Figure 10: → `http:3.99.168.88:8080/:jenkins` Interface snapshot

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

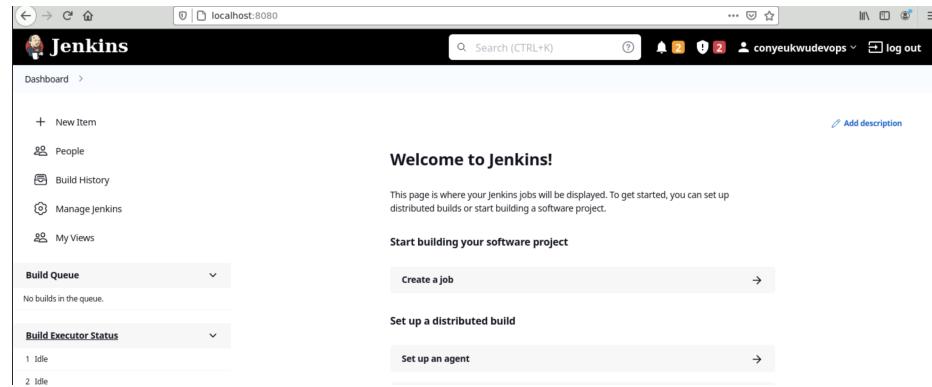


Figure 11: → Welcome Jenkins Interface snapshot

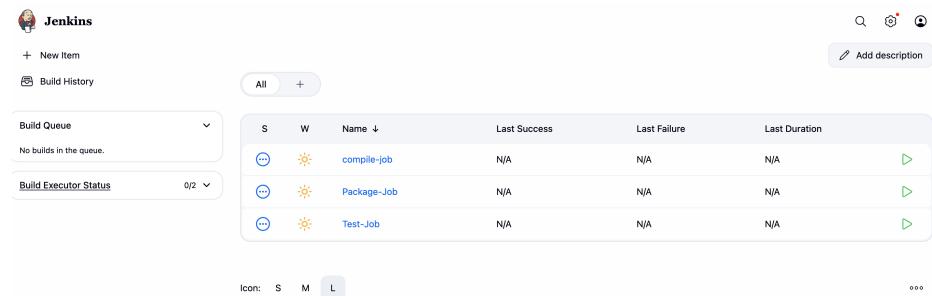


Figure 12: → 3 Jobs schedules Interface snapshot

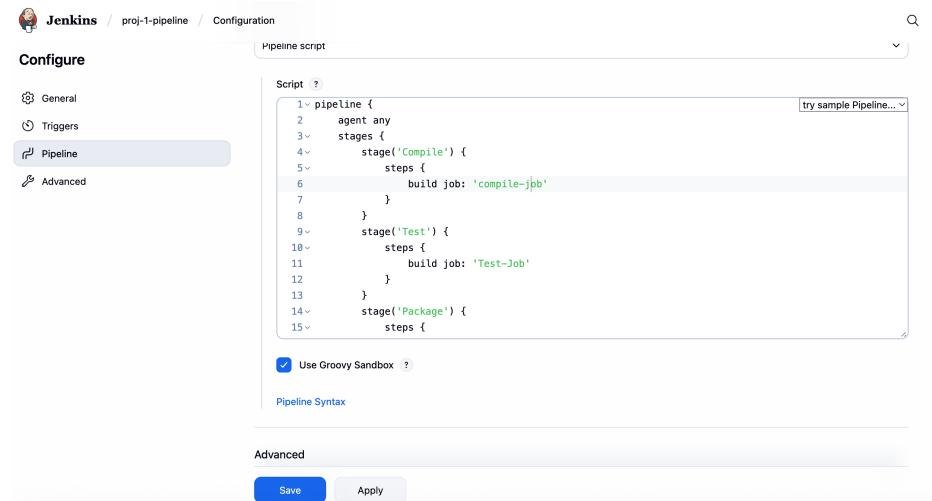


Figure 13: → Pipeline Code

The screenshot shows the Jenkins execution interface. On the left, there's a sidebar with 'Build Queue' and 'Build Executor Status'. Under 'Build Executor Status', it shows 'Built-In Node' (0/2) and 'slave-node-1' (offline). The main area is a table showing the build queue:

S	W	Name ↓	Last Success	Last Failure	Last Duration	F
✓	Cloud	compile-job	44 min #5	1 hr 10 min #2	3.7 sec	
✓	Cloud	Package-Job	44 min #4	1 hr 11 min #1	5.6 sec	
✓	Cloud	proj-1-pipeline	44 min #3	1 hr 10 min #1	35 sec	
✓	Cloud	Test-Job	44 min #6	1 hr 0 min #3	4.4 sec	

Figure 14: → Execution Interface

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

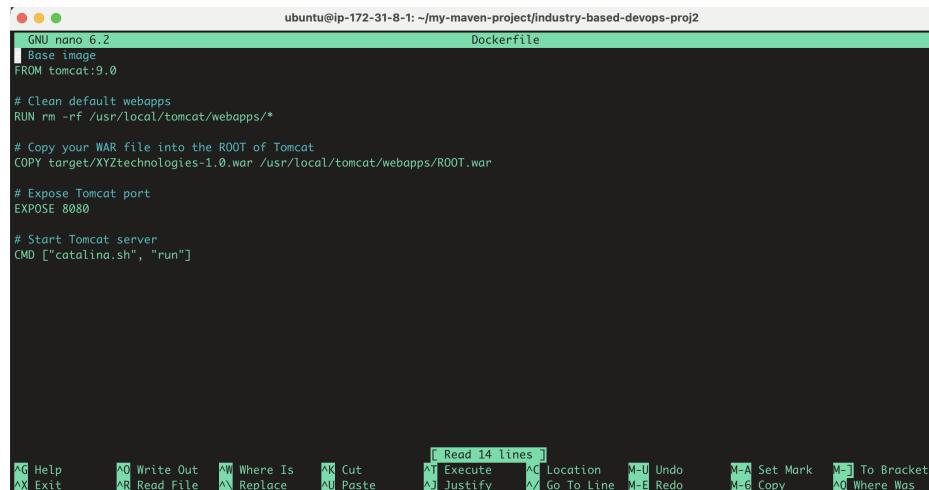
The screenshot shows the Jenkins Nodes management interface. It lists two nodes: 'Built-In Node' (Architecture: Linux (amd64), Clock Difference: In sync, Free Disk Space: 1.34 GiB, Free Swap Space: 0 B, Free Temp Space: 1.34 GiB, Response Time: 0ms) and 'slave-node-1' (Architecture: N/A, Clock Difference: N/A, Free Disk Space: N/A, Free Swap Space: N/A, Free Temp Space: N/A, Response Time: N/A). Below the table, it shows 'Data obtained' for all nodes.

Figure 15: → Slave Node

The screenshot shows the Jenkins slave node log interface for 'slave-node-1'. The 'Log' tab is selected. The log output shows the SSH launcher attempting to connect to the slave node, encountering a 'Name or service not known' error, and retrying multiple times. The log ends with 'SSH Connection failed with IOException: "127.0.0.1: Name or service not known", retrying in 15 seconds. There are 7 more retries left.'

Figure 16: → Lunching Slave Node

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



```

ubuntu@ip-172-31-8-1: ~/my-maven-project/industry-based-devops-proj2
GNU nano 6.2                               Dockerfile
Base image
FROM tomcat:9.0

# Clean default webapps
RUN rm -rf /usr/local/tomcat/webapps/*

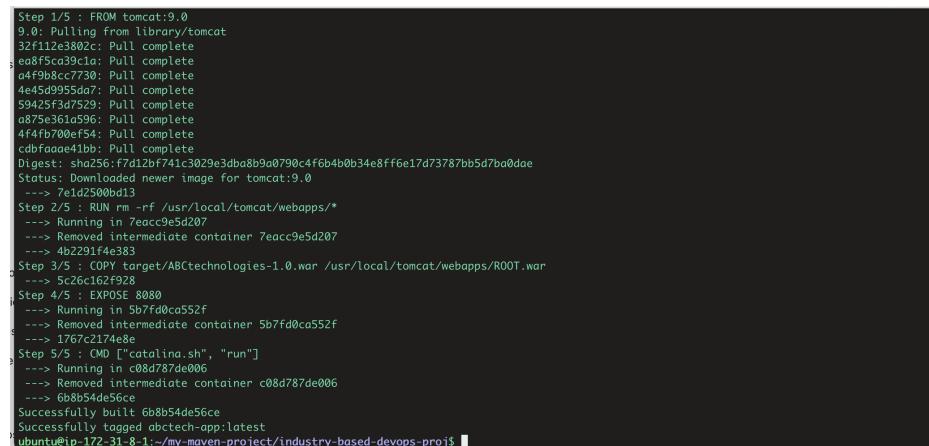
# Copy your WAR file into the ROOT of Tomcat
COPY target/XYZtechnologies-1.0.war /usr/local/tomcat/webapps/ROOT.war

# Expose Tomcat port
EXPOSE 8080

# Start Tomcat server
CMD ["catalina.sh", "run"]

```

Figure 17: → Docker File Script



```

Step 1/5 : FROM tomcat:9.0
9.0: Pulling from library/tomcat
32f112e3802c: Pull complete
e0af5f5ca39c1a: Pull complete
a4f9b8c7730: Pull complete
4ed5d9955dd7: Pull complete
59425f3d7529: Pull complete
a875e361a596: Pull complete
4fffb700ef54: Pull complete
cdff0aae41bb: Pull complete
Digest: sha256:f7d12bf741c3029e3dba8b9a0790c4f6b4b0b34e8ff6e17d73787bb5d7ba0d0e
Status: Downloaded newer image for tomcat:9.0
--> 7e1d2500bd13
Step 2/5 : RUN rm -rf /usr/local/tomcat/webapps/*
--> Running in 7e0cc9e5d207
--> Removed intermediate container 7e0cc9e5d207
--> 4b2291f4e385
Step 3/5 : COPY target/ABCtechnologies-1.0.war /usr/local/tomcat/webapps/ROOT.war
--> 5c26c162f928
Step 4/5 : EXPOSE 8080
--> Running in 5b7fd0ca552f
--> Removed intermediate container 5b7fd0ca552f
--> 1767c2174e8e
Step 5/5 : CMD ["catalina.sh", "run"]
--> Running in c08d787de006
--> Removed intermediate container c08d787de006
--> 6b8b54de56ce
Successfully built 6b8b54de56ce
Successfully tagged abctech-opp:latest
ubuntu@ip-172-31-8-1:~/my-maven-project/industry-based-devops-proj2

```

Figure 18: → Docker Script Execution

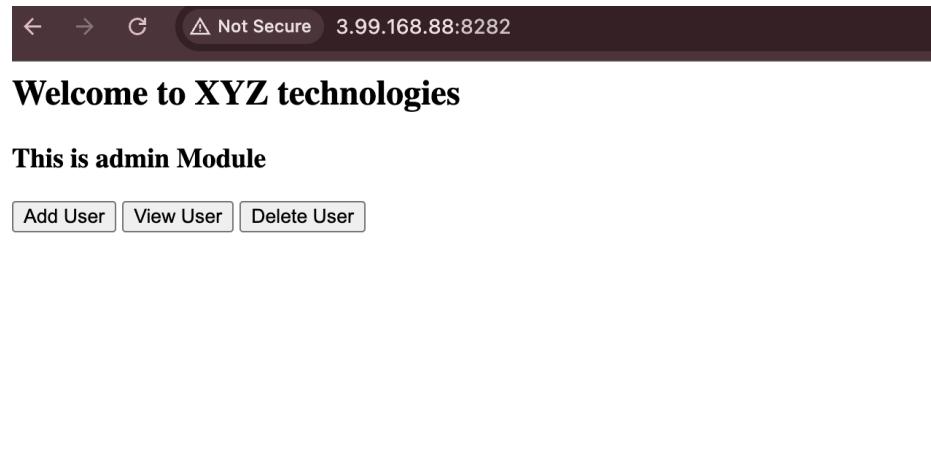


Figure 19: → Image running on Docker container snapshot

Task 4:Integrate Docker host with Ansible

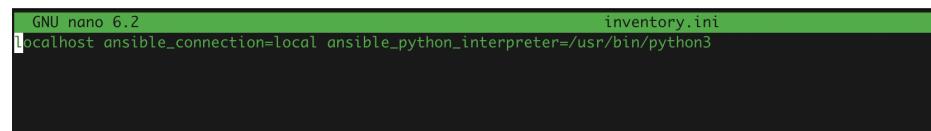


Figure 20: → 1: Deployment artifact snapshot

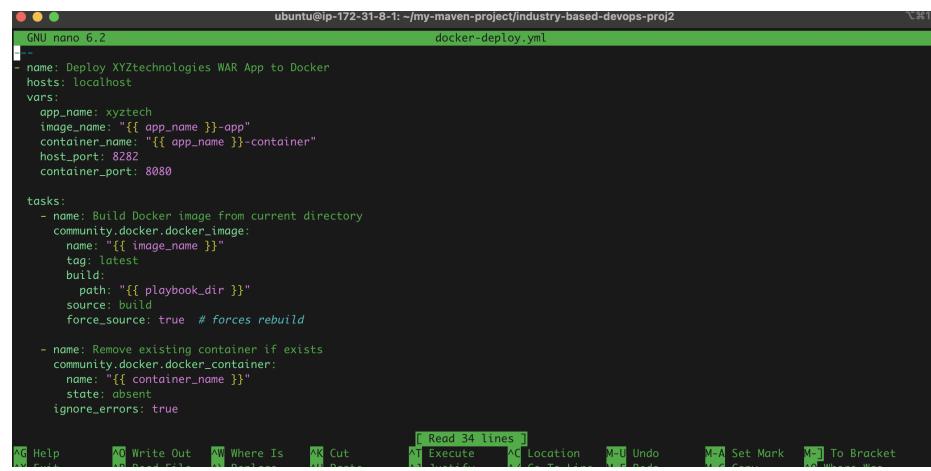
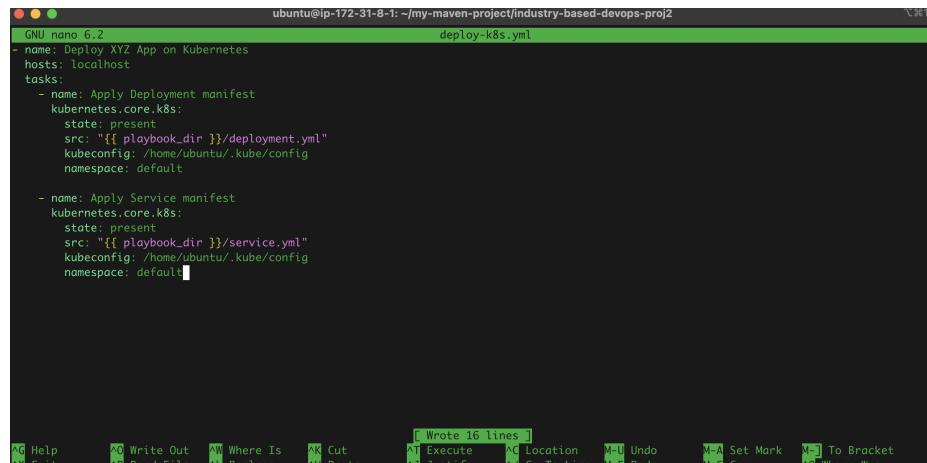


Figure 21: → 2: Ansible Playbook Script snapshot

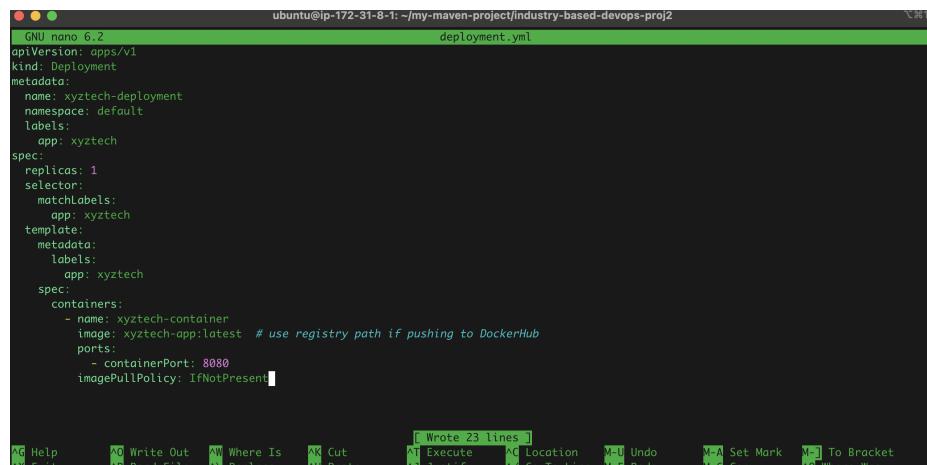


The screenshot shows a terminal window titled "ubuntu@ip-172-31-8-1: ~/my-maven-project/industry-based-devops-proj2". It displays a file named "deploy-k8s.yml" containing YML configuration for Kubernetes. The file includes sections for deploying a service and applying a deployment manifest. The terminal interface includes standard nano editor navigation keys at the bottom.

```
GNU nano 6.2                               deploy-k8s.yml
- name: Deploy XYZ App on Kubernetes
  hosts: localhost
  tasks:
    - name: Apply Deployment manifest
      kubernetes.core.k8s:
        state: present
        src: "{{ playbook_dir }}/deployment.yml"
        kubeconfig: /home/ubuntu/.kube/config
        namespace: default

    - name: Apply Service manifest
      kubernetes.core.k8s:
        state: present
        src: "{{ playbook_dir }}/service.yml"
        kubeconfig: /home/ubuntu/.kube/config
        namespace: default
```

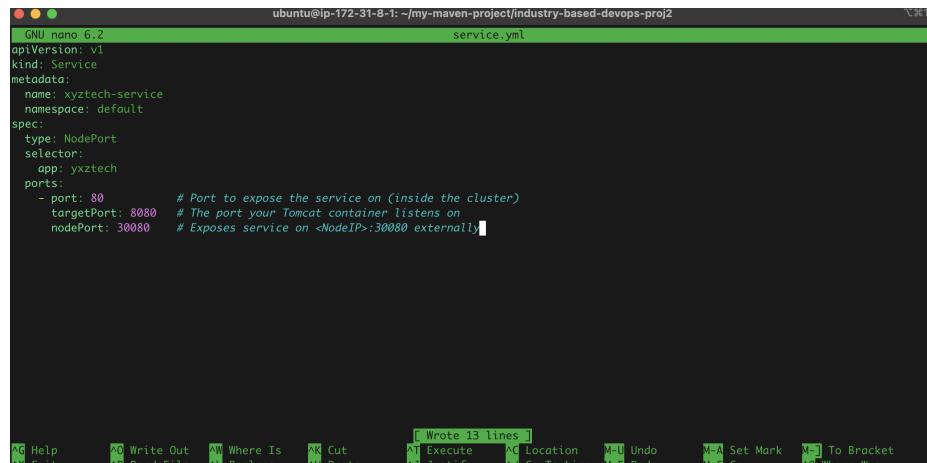
Figure 22: → 2: Deployment & Service Playbook snapshot



The screenshot shows a terminal window titled "ubuntu@ip-172-31-8-1: ~/my-maven-project/industry-based-devops-proj2". It displays a file named "deployment.yml" containing a single Deployment resource. The deployment has one replica, selects pods with the label "app: xyztech", and runs a container image "xyztech-app:latest" on port 8080. The terminal interface includes standard nano editor navigation keys at the bottom.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: xyztech-deployment
  namespace: default
  labels:
    app: xyztech
spec:
  replicas: 1
  selector:
    matchLabels:
      app: xyztech
  template:
    metadata:
      labels:
        app: xyztech
  spec:
    containers:
      - name: xyztech-container
        image: xyztech-app:latest  # use registry path if pushing to DockerHub
        ports:
          - containerPort: 8080
        imagePullPolicy: IfNotPresent
```

Figure 23: → 3: Kubernetes playbook script snapshot



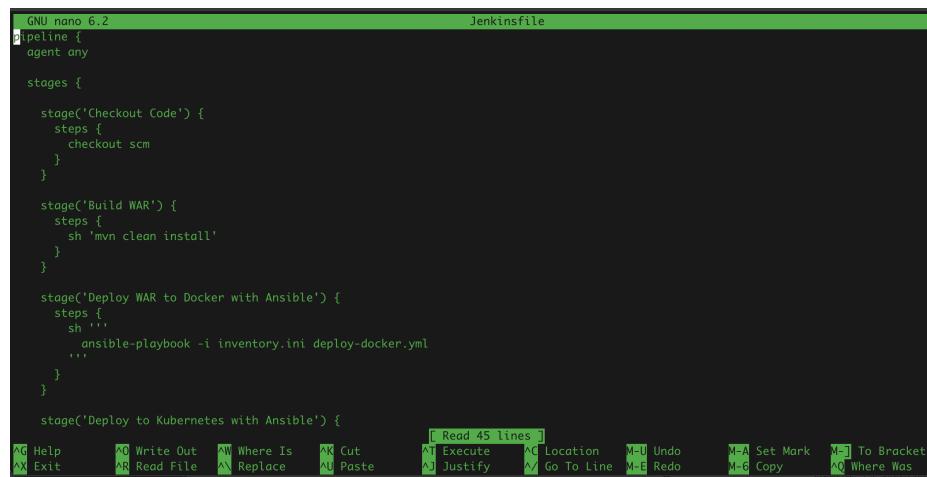
```

GNU nano 6.2                                         service.yml
apiVersion: v1
kind: Service
metadata:
  name: xyztech-service
  namespace: default
spec:
  type: NodePort
  selector:
    app: yxztech
  ports:
    - port: 80          # Port to expose the service on (inside the cluster)
      targetPort: 8080  # The port your Tomcat container listens on
      nodePort: 30080   # Exposes service on <NodeIP>:30080 externally

```

The screenshot shows a terminal window with a nano editor session. The file is named 'service.yml' and contains a Kubernetes Service configuration. The terminal has a green header bar with the title 'ubuntu@ip-172-31-8-1: ~/my-maven-project/industry-based-devops-proj2'. The nano editor interface includes a status bar at the bottom with various keyboard shortcuts like Help, Write Out, Where Is, Cut, Paste, Execute, Location, Undo, Set Mark, To Bracket, Exit, Read File, Replace, Justify, Go To Line, Redo, Copy, and Where Was.

Figure 24: → 3: Service Playbook Script snapshot



```

pipeline {
  agent any

  stages {
    stage('Checkout Code') {
      steps {
        checkout scm
      }
    }

    stage('Build WAR') {
      steps {
        sh 'mvn clean install'
      }
    }

    stage('Deploy WAR to Docker with Ansible') {
      steps {
        sh ...
        ansible-playbook -i inventory.ini deploy-docker.yml
      }
    }

    stage('Deploy to Kubernetes with Ansible') {
      steps {
        sh ...
      }
    }
  }
}

```

The screenshot shows a terminal window with a nano editor session for a Jenkins pipeline script named 'Jenkinsfile'. The script uses Groovy syntax to define a pipeline with four stages: 'Checkout Code', 'Build WAR', 'Deploy WAR to Docker with Ansible', and 'Deploy to Kubernetes with Ansible'. Each stage contains specific Jenkins steps like 'checkout scm' or 'sh' commands. The terminal has a green header bar with the title 'Jenkinsfile'. The nano editor interface includes a status bar at the bottom with various keyboard shortcuts like Help, Write Out, Where Is, Cut, Paste, Execute, Location, Undo, Set Mark, To Bracket, Exit, Read File, Replace, Justify, Go To Line, Redo, Copy, and Where Was.

Figure 25: → 4: Pipeline Script snapshot

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

The screenshot shows the Prometheus web interface. At the top, there are tabs for 'Alerts', 'Graph', 'Status', and 'Help'. Below the tabs, there's a search bar and a filter section with checkboxes for 'Unknown', 'Unhealthy', and 'Healthy'. There are two sections for targets:

- Targets**: Shows the 'node_exporter_metrics' target with 1/1 up. The table has columns: Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error. One row is shown: http://localhost:9100/metrics (UP, instance="localhost:9100", job="node_exporter_metrics", last scrape 2.718s ago, 16.902ms duration).
- prometheus**: Shows the 'prometheus' target with 1/1 up. The table has columns: Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error. One row is shown: http://localhost:9090/metrics (UP, instance="localhost:9090", job="prometheus", last scrape 5.776s ago, 5.072ms duration).

Figure 26: → 1: Prometheus Dashboard snapshot

The screenshot shows the Grafana web interface. At the top, it says 'Welcome to Grafana' and 'Need help? Documentation Tutorials Community Public Slack'. On the left, there's a sidebar with links: Home, Starred, Dashboards, Explore, Alerting, Connections, Apps, and Administration. The main area has several sections:

- Basic**: A guide for setting up Grafana.
- TUTORIAL DATA SOURCE AND DASHBOARDS**: A section about Grafana fundamentals.
- DATA SOURCES**: A section for adding data sources.
- DASHBOARDS**: Links for starred dashboards and recently viewed dashboards.
- Latest from the blog**: A featured blog post.

Figure 27: → 2: Grafana Dashboard snapshot

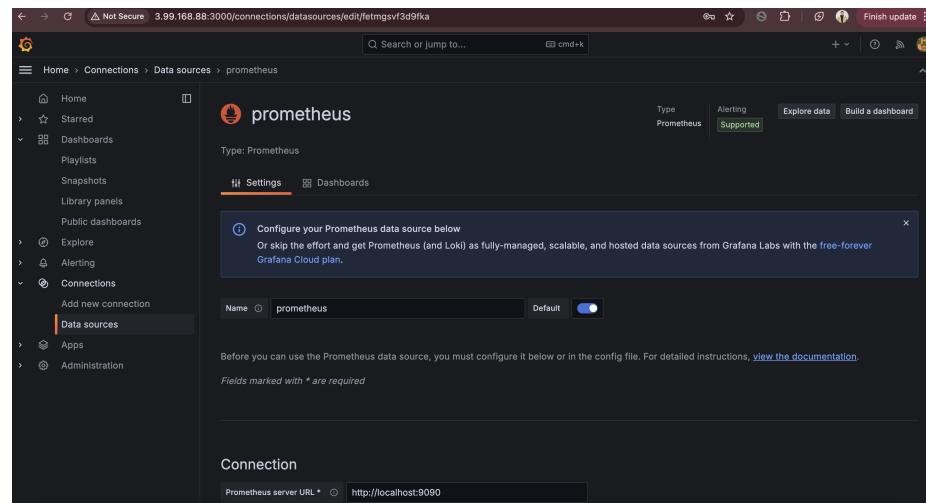


Figure 28: → 3: Added Prometheus as Data Source snapshot

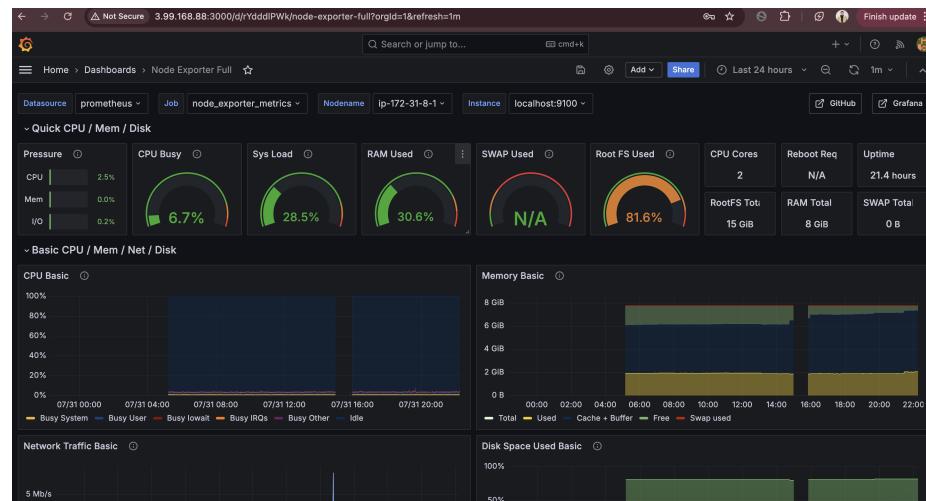


Figure 29: → 4: Monitoring Dashboard snapshot