

Industry Based Project 1 Assignment Report

Business Challenge/Requirement

ABC technologies is a leading online retail store. ABC has recently acquired a large retail offline business store. The business store has large number of stores across the globe but is following conventional pattern of development and deployment. As a result, it has landed to great loss and are facing below challenges.

- low available
- low scalable
- low Performance
- Hard built and maintained
- Developed and deployed is time consuming

ABC will acquire the data from all these storage systems and plan to use it for analytics and prediction of the firm's growth and sales prospect. In the first phase ABC has to create the servlets to Add a product and Display product details. Add servlet dependencies required to compile the servlets. Create an HTML page which will be used to add a product. Team is using git to keep all the source code. ABC has decided to use DevOps model and Once source code is available in Github, we need to integrate it with Jenkins and provide continuous build generation for continuous Delivery, integrate with Ansible and Kubernetes for deployment. Use docker hub to pull and push images between ansible and Kubernetes.

The Goal of the Project

Implement CI/CD such that *ABC* Company can be

1. Highly available
2. Highly scalable
3. Highly Performance
4. Easily built and maintained
5. Developed and deployed quickly

System Requirement

Hardware Required

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

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 grafana

Solution Report

Task 1

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/ABCTechnologies-1.0/> (optional)

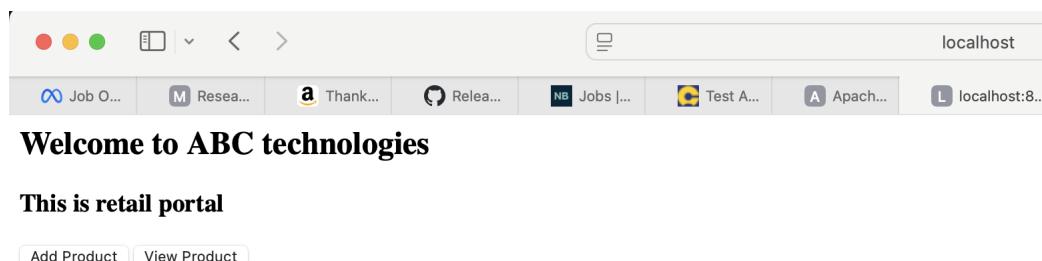


Figure 1: ABC Retail Data Entry & View webpage snapshot

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

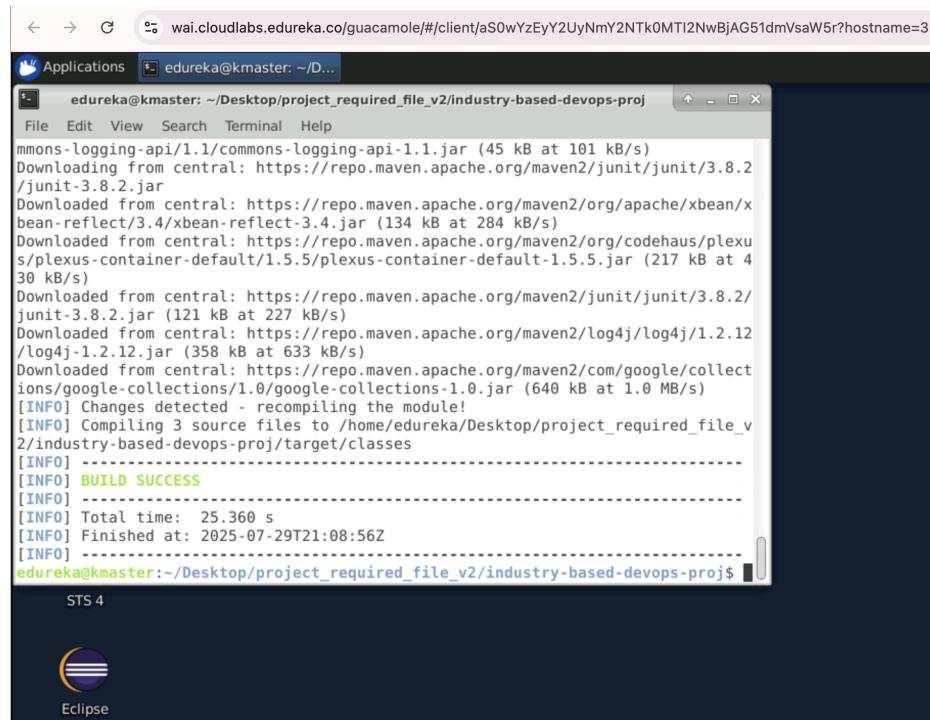
Step 4: Accessing the Virtual Machine (Edureka Cloud Lab)

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:

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

Step 5: Building codes using maven commands

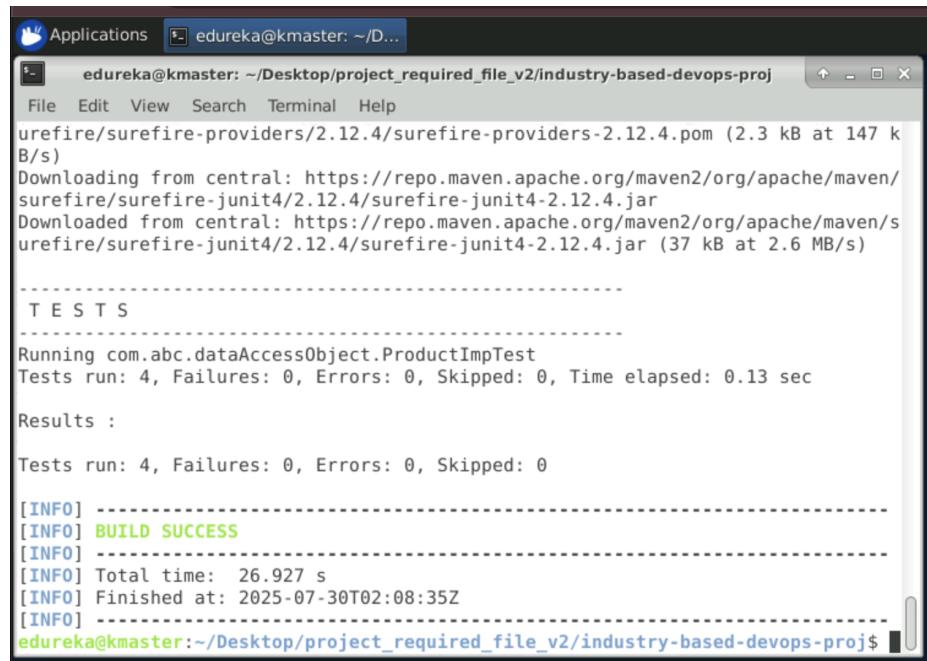
Created a folder on desktop in my lab (project-required-file-v2), cloned the file from my github and build with command



The screenshot shows a terminal window titled "edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj". The window displays the output of a Maven compile command. It shows the download of several JAR files from central Maven repositories, followed by the compilation of three source files into the target directory. The output ends with a green "BUILD SUCCESS" message and completion details.

```
maven:commons-logging-api/1.1/commons-logging-api-1.1.jar (45 kB at 101 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.2/junit-3.8.2.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/xbean/xbean-reflect/3.4/xbean-reflect-3.4.jar (134 kB at 284 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.5.5/plexus-container-default-1.5.5.jar (217 kB at 430 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/junit/junit/3.8.2/junit-3.8.2.jar (121 kB at 227 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/log4j/log4j/1.2.12/log4j-1.2.12.jar (358 kB at 633 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/google/collections/google-collections/1.0/google-collections-1.0.jar (640 kB at 1.0 MB/s)
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 3 source files to /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/target/classes
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 25.360 s
[INFO] Finished at: 2025-07-29T21:08:56Z
[INFO] -----
```

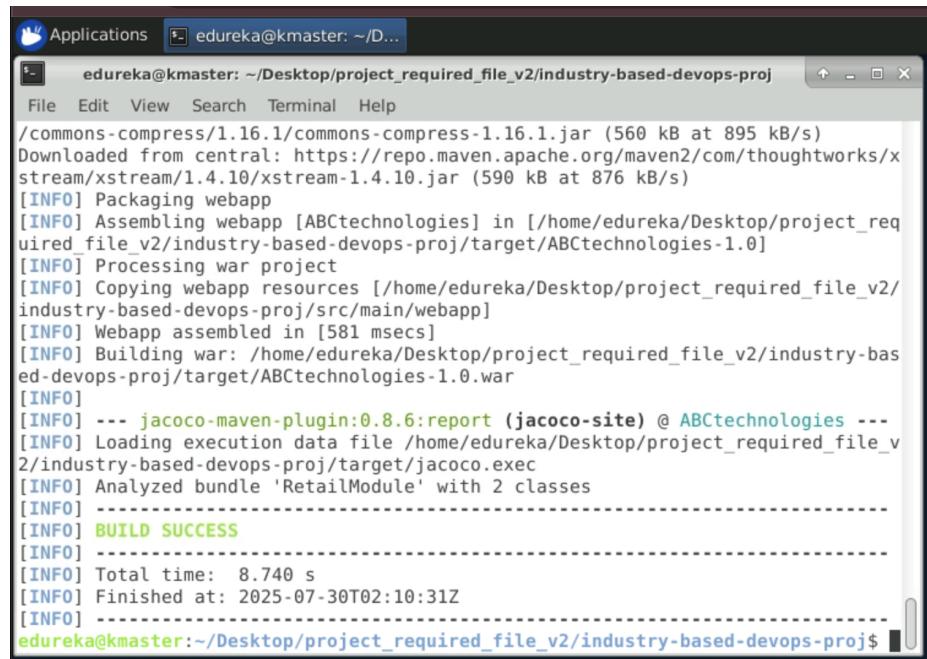
Figure 2: → `/opt/maven/bin/mvn compile`:Compile Interface snapshot



```
edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj$ mvn test
[INFO] Scanning for projects...
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire-junit4/2.12.4/surefire-junit4-2.12.4.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/surefire-junit4/2.12.4/surefire-junit4-2.12.4.jar (37 kB at 2.6 MB/s)

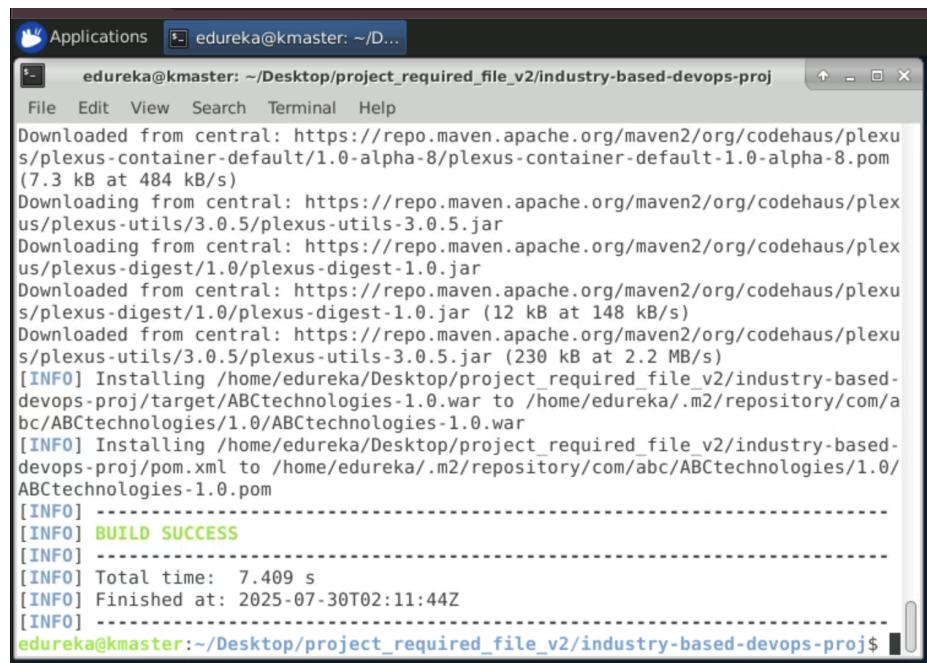
[INFO] [INFO] T E S T S
[INFO] [INFO] Running com.abc.dataAccessObject.ProductImpTest
[INFO] [INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.13 sec
[INFO] [INFO] Results :
[INFO] [INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO] [INFO] [INFO] [INFO] BUILD SUCCESS
[INFO] [INFO] [INFO] [INFO] Total time: 26.927 s
[INFO] [INFO] [INFO] [INFO] Finished at: 2025-07-30T02:08:35Z
[INFO] [INFO] [INFO] [INFO] edureka@kmaster:~/Desktop/project_required_file_v2/industry-based-devops-proj$
```

Figure 3: → /opt/maven/bin/mvn test:Test Interface snapshot



```
edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj$ mvn package
[INFO] Scanning for projects...
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/com/thoughtworks/xstream/xstream/1.4.10/xstream-1.4.10.jar (590 kB at 876 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/com/thoughtworks/xstream/xstream/1.4.10/xstream-1.4.10.jar (590 kB at 876 kB/s)
[INFO] [INFO] Packaging webapp
[INFO] [INFO] Assembling webapp [ABCtechnologies] in [/home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/target/ABCtechnologies-1.0]
[INFO] [INFO] Processing war project
[INFO] [INFO] Copying webapp resources [/home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/src/main/webapp]
[INFO] [INFO] Webapp assembled in [581 ms]
[INFO] [INFO] Building war: /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/target/ABCtechnologies-1.0.war
[INFO] [INFO] [INFO] --- jacoco-maven-plugin:0.8.6:report (jacoco-site) @ ABCtechnologies ---
[INFO] [INFO] Loading execution data file /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/target/jacoco.exec
[INFO] [INFO] Analyzed bundle 'RetailModule' with 2 classes
[INFO] [INFO] [INFO] [INFO] BUILD SUCCESS
[INFO] [INFO] [INFO] [INFO] Total time: 8.740 s
[INFO] [INFO] [INFO] [INFO] Finished at: 2025-07-30T02:10:31Z
[INFO] [INFO] [INFO] [INFO] edureka@kmaster:~/Desktop/project_required_file_v2/industry-based-devops-proj$
```

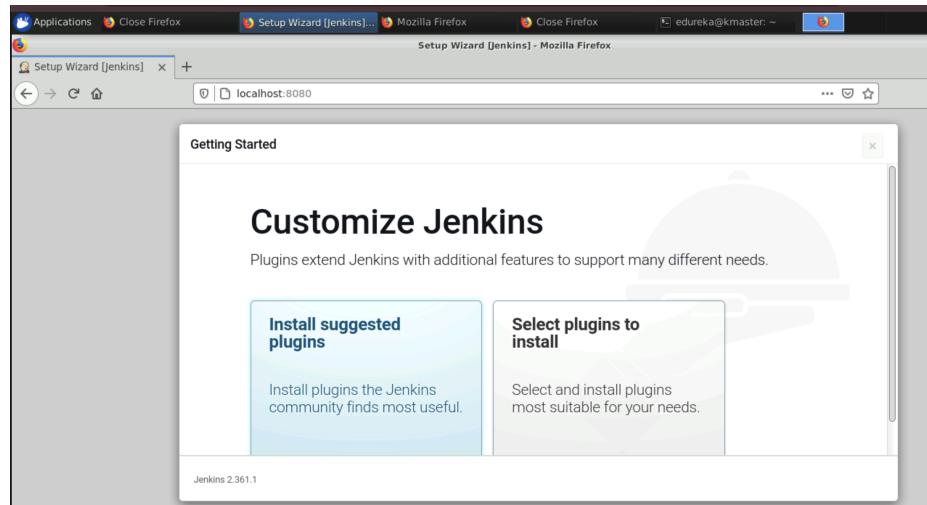
Figure 4: → /opt/maven/bin/mvn package:Package Interface snapshot



```
edureka@kmaster: ~/Desktop/project_required_file_v2/industry-based-devops-proj
File Edit View Search Terminal Help
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-container-default/1.0-alpha-8/plexus-container-default-1.0-alpha-8.pom (7.3 kB at 484 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.5/plexus-utils-3.0.5.jar
Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-digest/1.0/plexus-digest-1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-digest/1.0/plexus-digest-1.0.jar (12 kB at 148 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.5/plexus-utils-3.0.5.jar (230 kB at 2.2 MB/s)
[INFO] Installing /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/target/ABCtechnologies-1.0.war to /home/edureka/.m2/repository/com/abc/ABCtechnologies/1.0/ABCtechnologies-1.0.war
[INFO] Installing /home/edureka/Desktop/project_required_file_v2/industry-based-devops-proj/pom.xml to /home/edureka/.m2/repository/com/abc/ABCtechnologies/1.0/ABCtechnologies-1.0.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 7.409 s
[INFO] Finished at: 2025-07-30T02:11:44Z
[INFO] -----
edureka@kmaster:~/Desktop/project_required_file_v2/industry-based-devops-proj$
```

Figure 5: → */opt/maven/bin/mvn clean install:clean install* Interface snapshot

Task 2.1 Setting Jenkins CI/CD pipeline

Figure 6: → *http:localhost:8080/login:jenkins* Interface snapshot

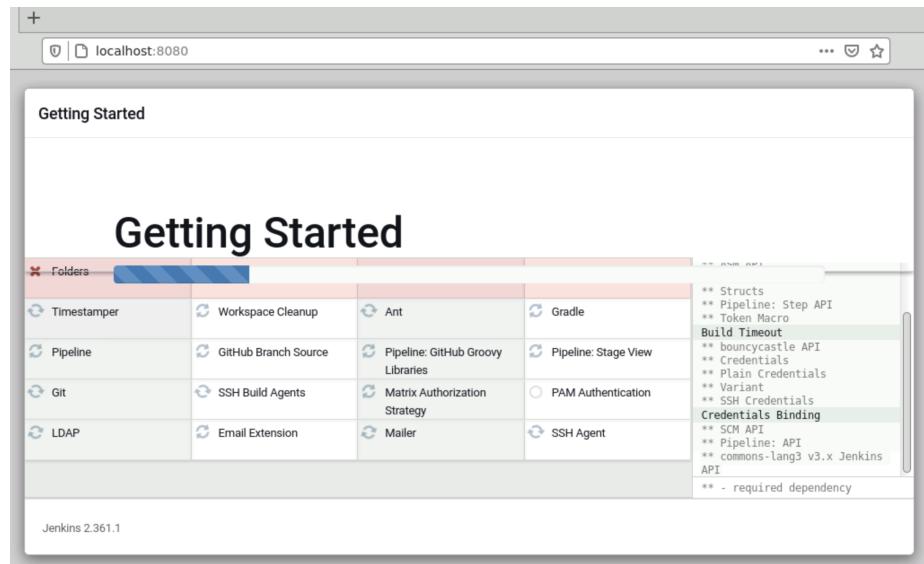


Figure 7: → Selected plugins installation snapshot

Exception:

The plugin installation stopped and I made extra effort to fix the plugin issue but no success except the jenkins version is updated to the latest:

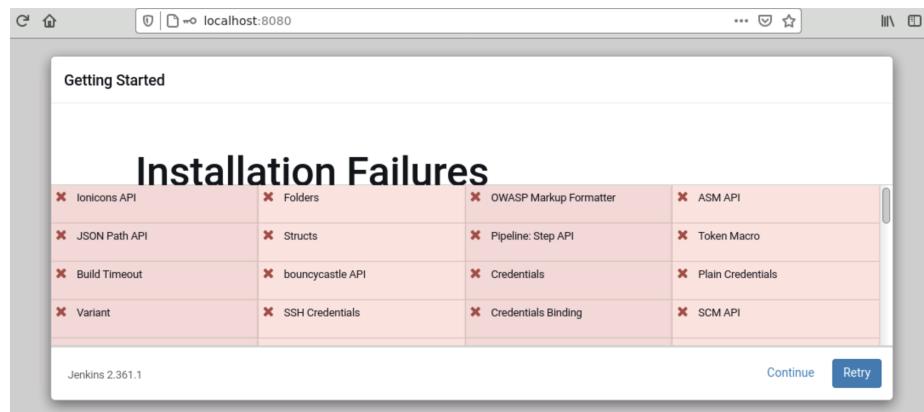


Figure 8: → plugin Installation Error

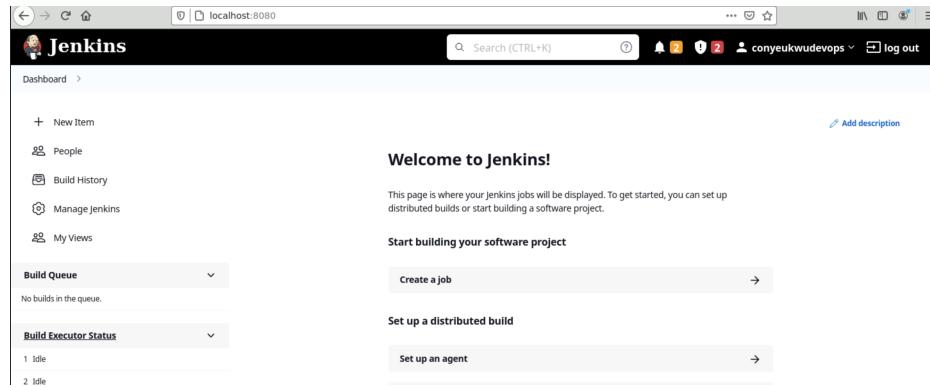


Figure 9: → Welcome Jenkins Interface snapshot

Exception handled

Resolved my exception by moving my devops virtual platform to Amazon EC2 Cloud t2.large instance at cost 0.126 dollar/hour. And after success installation i continued to set global tool configuration:

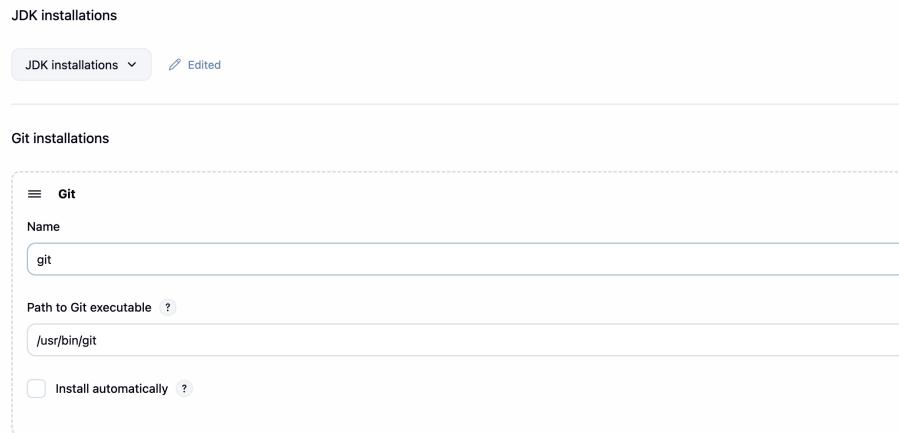


Figure 10: → Git Installation configuration Interface snapshot

Task 2.2 Create build pipeline containing 3 jobs

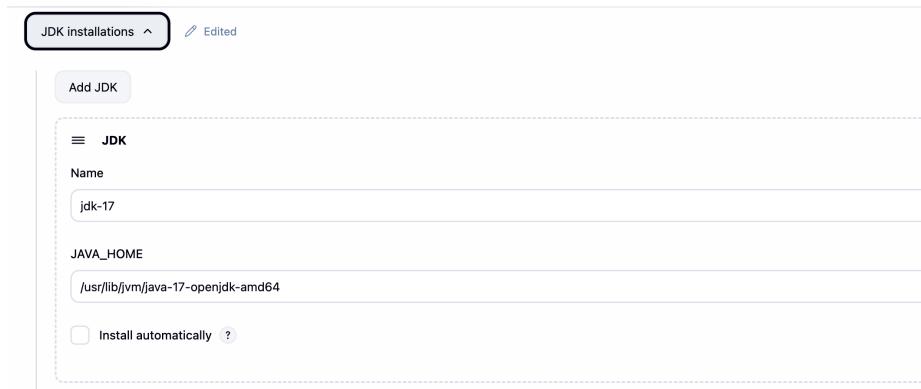


Figure 11: → JDK Installation configuration Interface snapshot

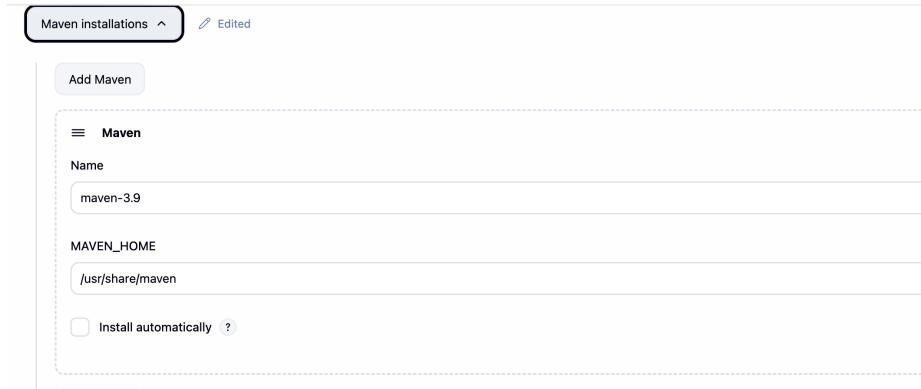


Figure 12: → Maven Installation configuration Interface snapshot

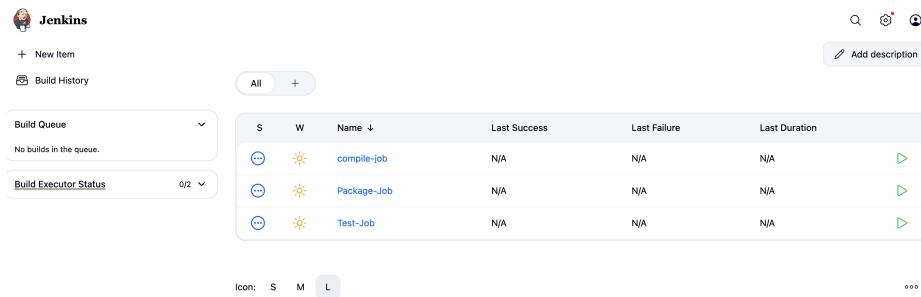


Figure 13: → 3 Jobs schedules Interface snapshot

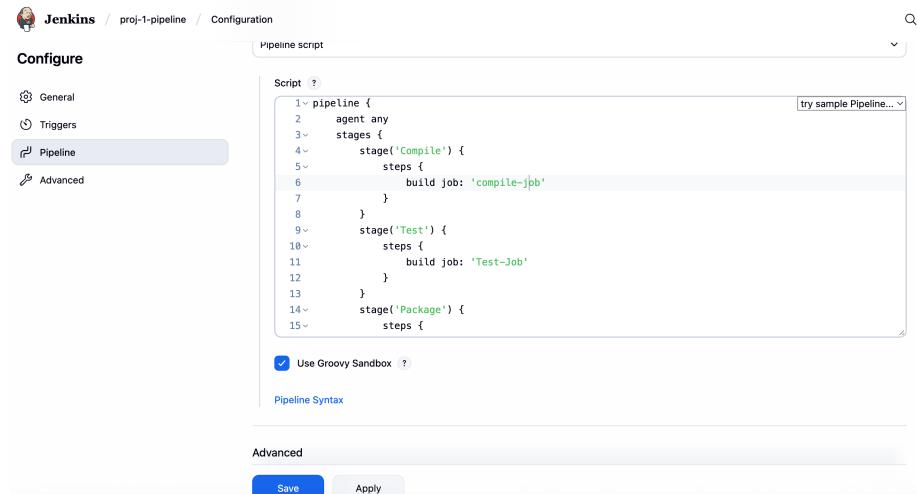


Figure 14: → Pipeline Code

The screenshot shows the Jenkins execution interface. The top navigation bar includes links for New Item, Build History, and Add description. The main area displays the Build Queue and Build Executor Status. The Build Queue table shows four jobs: compile-job, Package-Job, proj-1-pipeline, and Test-Job, all of which have passed. The Build Executor Status table shows two nodes: Built-in Node (0/2) and slave-node-1 (offline). At the bottom, there are icons for S, M, and L, and a status bar showing 600.

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 15: → Execution Interface

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

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
💻	Built-In Node	Linux (amd64)	In sync	⚠ 1.34 GiB	⚠ 0 B	⚠ 1.34 GiB	0ms 🕒
💻	slave-node-1		N/A	N/A	N/A	N/A	N/A 🕒
	Data obtained		8 min 20 sec	8 min 20 sec	8 min 20 sec	8 min 20 sec	8 min 20 sec

Figure 16: → Slave Node

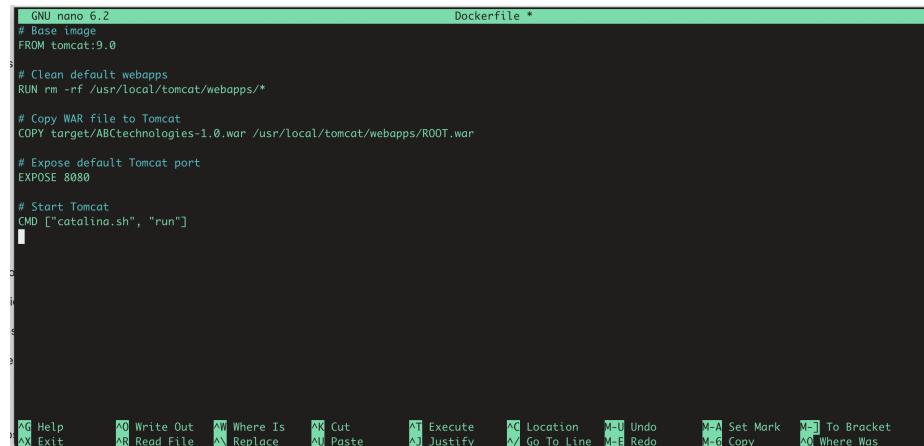
```

SSHLauncher(host='127.0.0.1', port=22, credentialsId='slave-ec2-key', jvmOptions='', javaPath='',
prefixStartSlaveCmd='', suffixStartSlaveCmd='', launchTimeoutSeconds=60, maxNumRetries=10, retryWaitTime=15,
sshHostKeyVerificationStrategy=hudson.plugins.sshslaves.verifiers.NonVerifyingKeyVerificationStrategy, tcpNoDelay=true,
trackCredentials=true)
[07/30/25 19:28:32] [SSH] Opening SSH connection to 127.0.0.1:22.
127.0.0.1: Name or service not known
SSH Connection failed with IOException: "127.0.0.1: Name or service not known", retrying in 15 seconds. There are 10
more retries left.
127.0.0.1: Name or service not known
SSH Connection failed with IOException: "127.0.0.1: Name or service not known", retrying in 15 seconds. There are 9
more retries left.
127.0.0.1: Name or service not known
SSH Connection failed with IOException: "127.0.0.1: Name or service not known", retrying in 15 seconds. There are 8
more retries left.
127.0.0.1: Name or service not known
SSH Connection failed with IOException: "127.0.0.1: Name or service not known", retrying in 15 seconds. There are 7
more retries left.
127.0.0.1: Name or service not known

```

Figure 17: → Lunching Slave Node

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



```

GNU nano 6.2                               Dockerfile *

# Base image
FROM tomcat:9.0

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

# Copy WAR file to Tomcat
COPY target/ABCtechnologies-1.0.war /usr/local/tomcat/webapps/ROOT.war

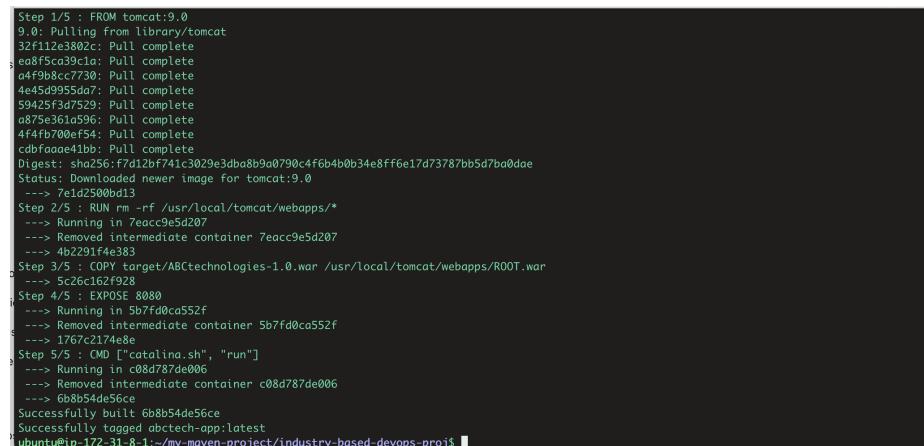
# Expose default Tomcat port
EXPOSE 8080

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

```

The screenshot shows a terminal window with a nano text editor open. The file is named 'Dockerfile'. The content of the file is a Dockerfile script. It starts with a base image 'tomcat:9.0', removes the default 'webapps' directory, copies a 'ROOT.war' file from the local machine to the Tomcat 'webapps' directory, exposes port 8080, and runs the Tomcat container using the 'catalina.sh' command.

Figure 18: → Docker File Script



```

Step 1/5 : FROM tomcat:9.0
9.0: Pulling from library/tomcat
32f112e3802c: Pull complete
...
Status: Downloaded newer image for tomcat:9.0
--> 7eacc9e5d207
Step 2/5 : RUN rm -rf /usr/local/tomcat/webapps/*
--> Running in 7eacc9e5d207
--> Removed intermediate container 7eacc9e5d207
--> 4b291f4e383
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
--> Gb8b54de56ce
Successfully built 6b8b54de56ce
Successfully tagged abctech-app:latest
ubuntu@ip-172-31-8-1:~/my-maven-project/industry-based-devops-proj$ 

```

The screenshot shows a terminal window executing a Docker build command. The output shows the steps: pulling the 'tomcat:9.0' image, removing the 'webapps' directory, copying the 'target/ABCtechnologies-1.0.war' file to the 'webapps' directory, exposing port 8080, and running the container using the 'catalina.sh' command. The final message indicates the image was successfully built and tagged as 'abctech-app:latest'.

Figure 19: → Docker Script Execution

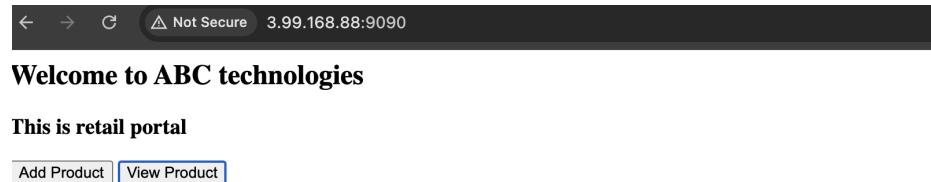


Figure 20: → Image running on Docker container snapshot

Task 4:Integrate Docker host with Ansible



Figure 21: → 1: Deployment artifact snapshot

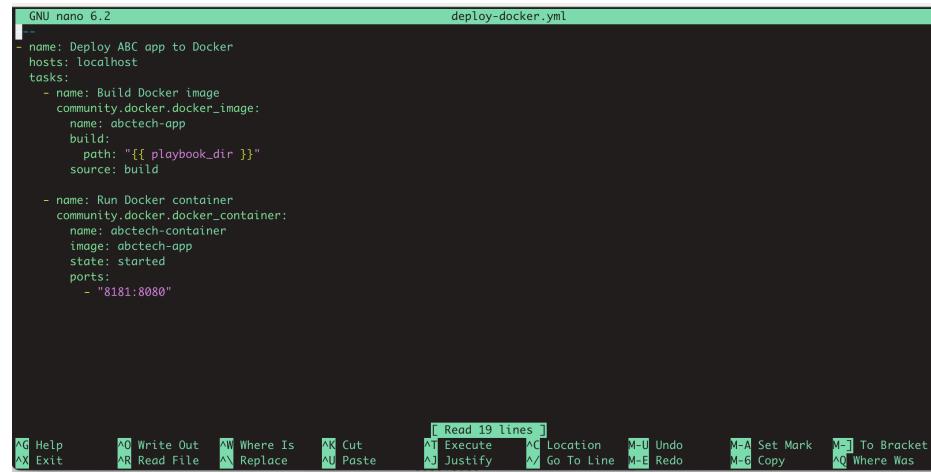


Figure 22: → 2: Ansible Playbook Script snapshot

```
GNU nano 6.2 deploy-k8s.yml
name: Deploy ABC 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

[ Read 17 lines ]

```

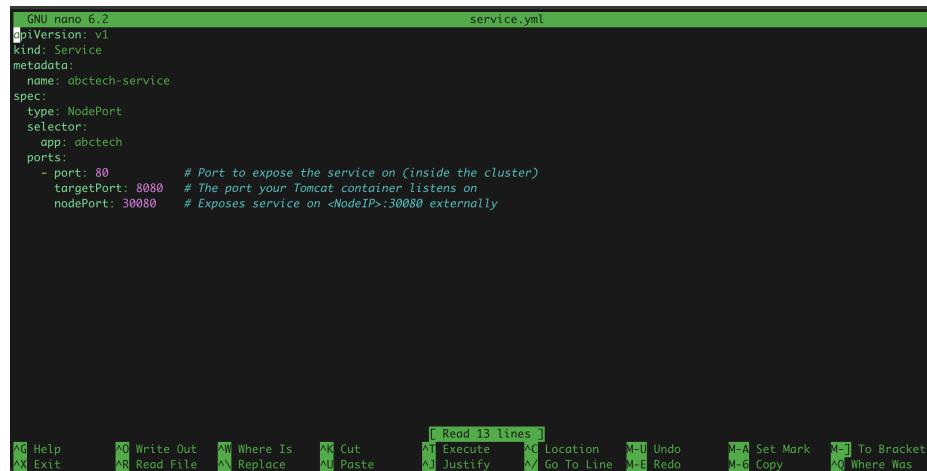
Figure 23: → 3: pod kube8s playbook snapshot

```
GNU nano 6.2 deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: abctech-deployment
  labels:
    app: abctech
spec:
  replicas: 1
  selector:
    matchLabels:
      app: abctech
  template:
    metadata:
      labels:
        app: abctech
    spec:
      containers:
        - name: abctech-container
          image: abctech-app:latest # use registry path if pushing to DockerHub
          ports:
            - containerPort: 8080
          imagePullPolicy: IfNotPresent

[ Read 23 lines ]

```

Figure 24: → 3: Kubernetes playbook script snapshot



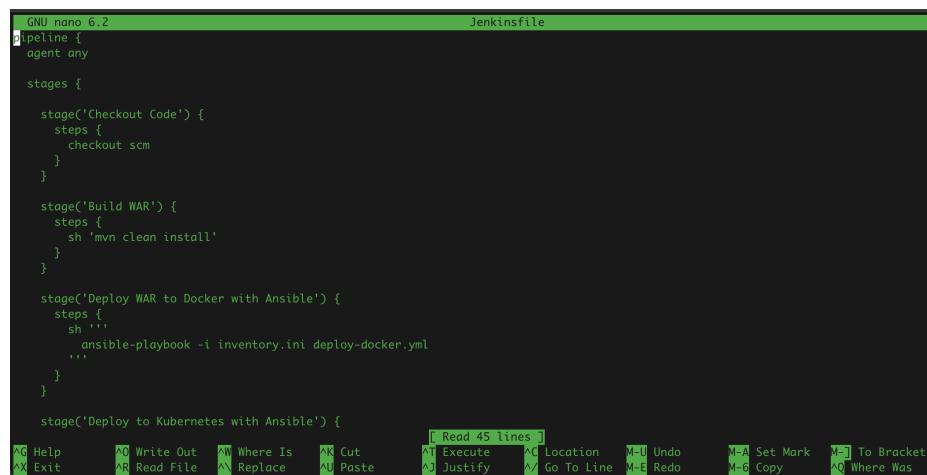
```

  GNU nano 6.2
apiVersion: v1
kind: Service
metadata:
  name: abctech-service
spec:
  type: NodePort
  selector:
    app: abctech
  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 the file 'service.yml' open in the nano editor. The file contains a single YAML object representing a Kubernetes Service. It specifies the API version, kind, metadata (name), spec (type: NodePort, selector: app: abctech), and ports (port: 80, targetPort: 8080, nodePort: 30080). The terminal has a green header bar with the title 'service.yml'. The bottom of the screen shows the nano editor's command-line interface with various keyboard shortcuts.

Figure 25: → 3: Service Playbook Script snapshot



```

  GNU nano 6.2
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') {

```

The screenshot shows a terminal window with the file 'Jenkinsfile' open in the nano editor. The file defines a Jenkins pipeline. It starts with an 'agent any' block, followed by four stages: 'Checkout Code', 'Build WAR', 'Deploy WAR to Docker with Ansible', and 'Deploy to Kubernetes with Ansible'. Each stage contains a 'steps' block with specific commands. The terminal has a green header bar with the title 'Jenkinsfile'. The bottom of the screen shows the nano editor's command-line interface with various keyboard shortcuts.

Figure 26: → 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 with a placeholder 'Filter by endpoint or labels' and a checkbox group for 'Unknown', 'Unhealthy', and 'Healthy' status filters. A 'Targets' section follows, divided into two sections: 'node_exporter_metrics (1/1 up)' and 'prometheus (1/1 up)'. Each section contains a table with columns: Endpoint, State, Labels, Last Scrape, Scrape Duration, and Error. In the 'node_exporter_metrics' section, there is one entry for 'http://localhost:9100/metrics' with state 'UP', labels 'instance="localhost:9100"', and 'job="node_exporter_metrics"'. The 'Last Scrape' was 2.718s ago and the 'Scrape Duration' was 16.902ms. In the 'prometheus' section, there is one entry for 'http://localhost:9090/metrics' with state 'UP', labels 'instance="localhost:9090"', and 'job="prometheus"'. The 'Last Scrape' was 5.776s ago and the 'Scrape Duration' was 5.072ms.

Figure 27: → 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' (with a guide for setting up Grafana), 'TUTORIAL DATA SOURCE AND DASHBOARDS Grafana fundamentals' (with a link to 'Learn how in the docs'), 'DATA SOURCES Add your first data source' (with a link to 'Learn how'), and 'DASHBOARDS Create your dashboard' (with a link to 'Learn how'). Below these sections, there are 'Dashboards' and 'Latest from the blog' sections. The 'Dashboards' section includes links for 'Starred dashboards' and 'Recently viewed dashboards'.

Figure 28: → 2: Grafana Dashboard snapshot

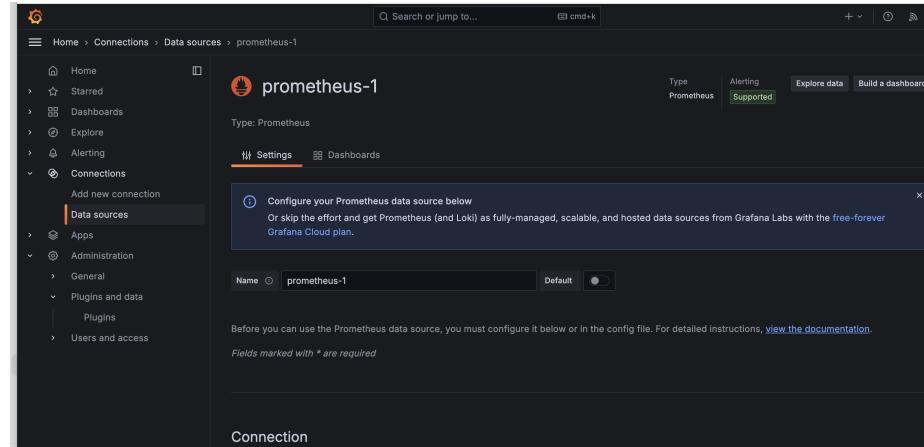


Figure 29: → 3: Added Prometheus as Data Source snapshot

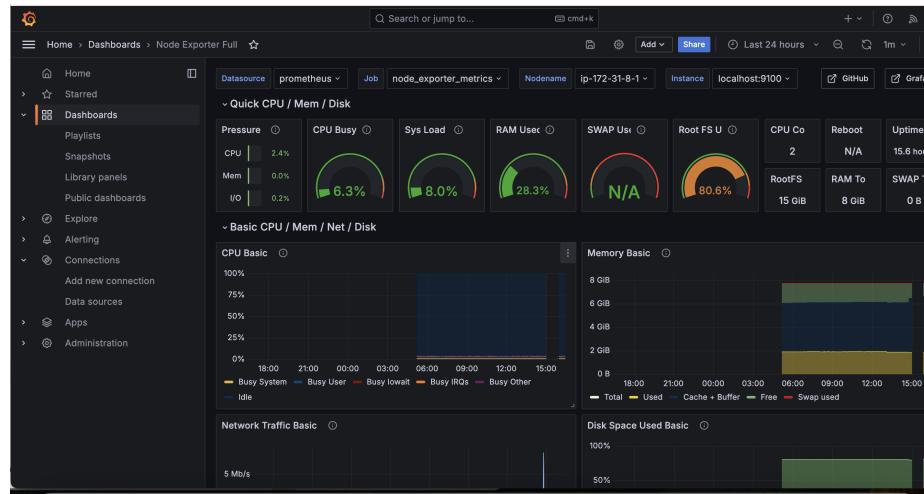


Figure 30: → 4: Monitoring Dashboard snapshot