Version	2.0.1
Prepared by	Profit-Plus
Audience	Park-Innovation
Date	14-Mar-2024

Contents

1. Introduction

Welcome to the comprehensive documentation for deploying our website on a Kubernetes cluster using Minikube, Docker, Jenkins Pipeline, Kubernetes tools, Prometheus, Grafana, and GitHub. This documentation provides detailed steps to ensure a smooth and successful deployment, monitoring, and collaboration process.

2. Prerequisites

Docker: Ensure Docker is installed on your machine. Follow the instructions in the official Docker documentation for your specific operating system.

Minikube: Install Minikube to set up a local Kubernetes cluster. Refer to the official Minikube installation guide for detailed instructions.

Kubernetes Tools:

- kubectl: Install the Kubernetes command-line tool by following the guidelines in the official documentation.
- helm: If using Helm for package management, install Helm by referring to the official Helm installation guide.

Jenkins:

Ensure Jenkins is installed and configured. You can find detailed instructions in the official Jenkins documentation.

Prometheus and Grafana:

For monitoring, install Prometheus by following the steps outlined in the official Prometheus documentation.

Install Grafana using the instructions provided in the official Grafana documentation.

GitHub Account:

Create a GitHub account if you don't have one already. The documentation assumes that the project is hosted on GitHub for version control and collaboration.

3. Setup

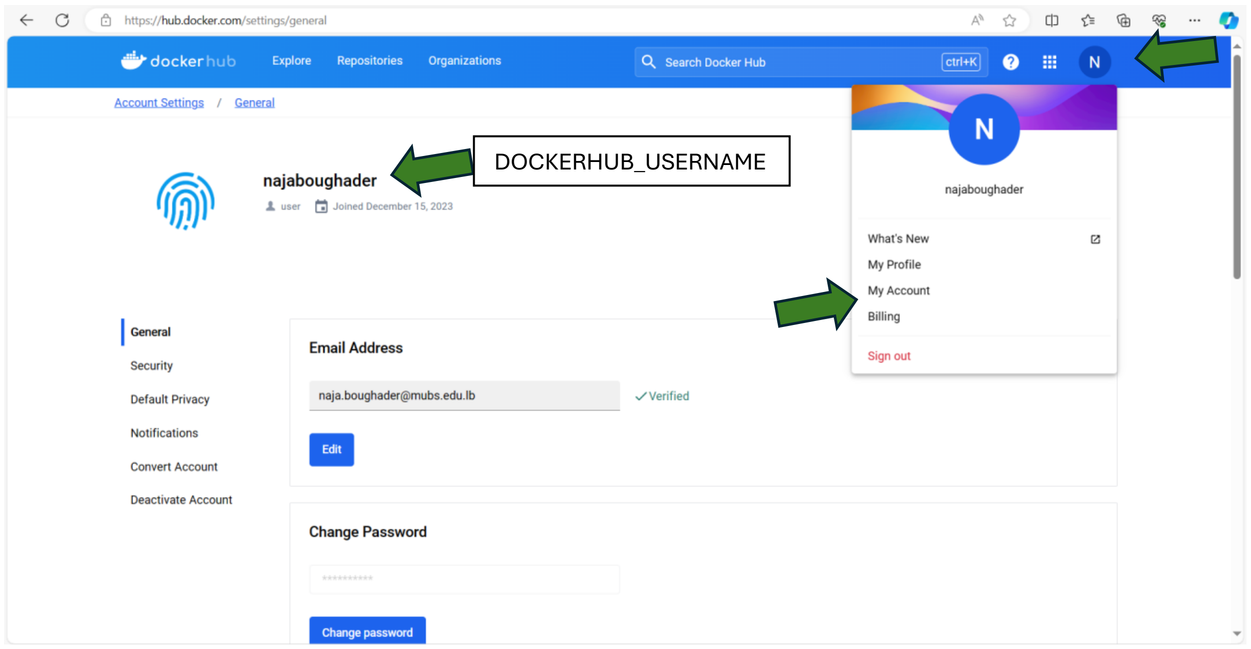
3.1 Organize GitHub repository

1. Create a GitHub organization, then make three repositories (first to push your source code on, second to push the Jenkins file for the configuration of the pipeline).
2. In the first repository that contains the source code you need to add a workflow github actions. (build.yml)

```
name: Build, Push, and Deploy Docker Images
on:
  push:
    branches:
      - Production
jobs:
  build:
    runs-on: ubuntu-latest
    services:
      mysql:
        image: wissamrh/mysql:latest
        env:
          MYSQL_ROOT_PASSWORD: root
          MYSQL_DATABASE: mydatabase
          MYSQL_USER: myuser
          MYSQL_PASSWORD: mypassword
        ports:
          - 3306:3306
        options: --health-cmd="mysqladmin ping" --health-interval=10s --health-timeout=5s --health-
retries=3


    steps:
    - name: Checkout code
      uses: actions/checkout@v2
    - name: Set version as an environment variable
```

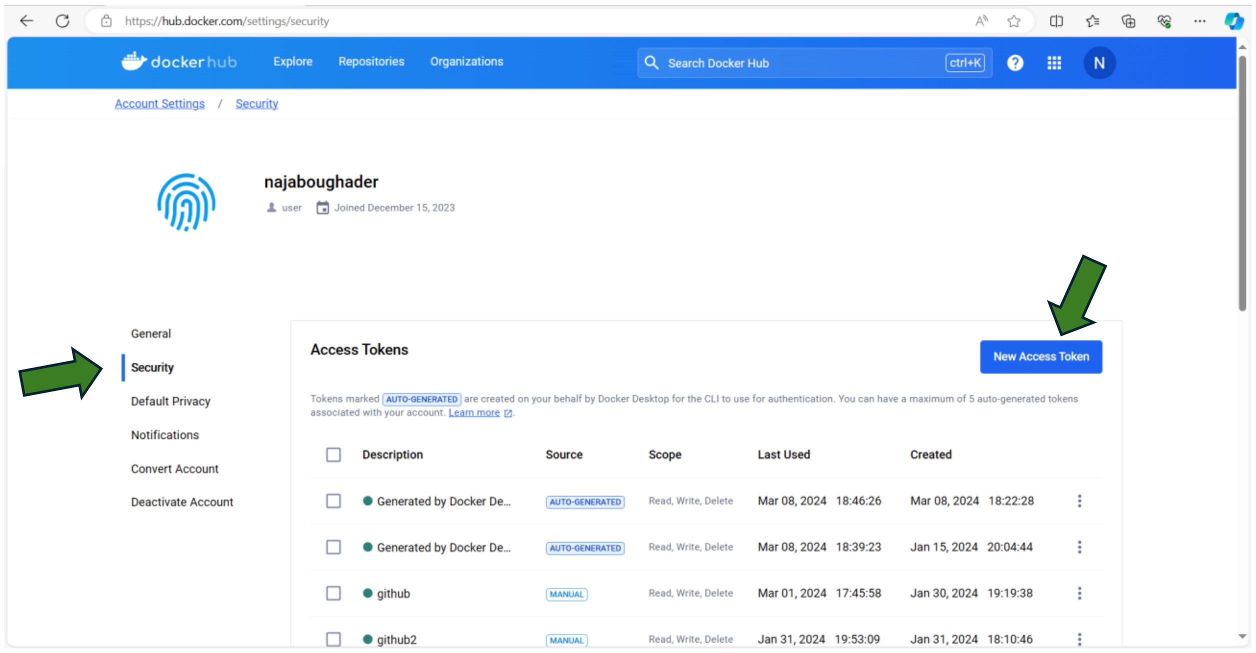
```
      run: echo "VERSION=3.0.${{ github.run_number }}" >> $GITHUB_ENV

    - name: Build App Docker image

      run: docker build -t wissamrh/wissamrh:${{ env.VERSION }} -f First-release/Dockerfiles/Dockerfile
.

    - name: Build Database Docker image

      run: docker build -t wissamrh/mysql:${{ env.VERSION }} -f First-
release/Dockerfiles/Dockerfile.database .

    - name: Build phpMyAdmin Docker image

      run: docker build -t wissamrh/php:${{ env.VERSION }} -f First-
release/Dockerfiles/Dockerfile.phpmyadmin .

    - name: Log in to Docker Hub

      run: echo ${{ secrets.DOCKERHUB_TOKEN }} | docker login -u ${{
secrets.DOCKERHUB_USERNAME }} --password-stdin

    - name: Push App Docker image

      run: docker push wissamrh/wissamrh:${{ env.VERSION }}

    - name: Push Database Docker image

      run: docker push wissamrh/mysql:${{ env.VERSION }}

    - name: Push phpMyAdmin Docker image

      run: docker push wissamrh/php:${{ env.VERSION }}
```

3. Add to the secrets DOCKERHUB_USERNAME and DOCKERHUB_TOKEN
    - DOCKERHUB_USERNAME:
      Go to https://hub.docker.com sign in using your credentials, then go to my account.



Save this username as DOCKERHUB_USERNAME in GitHub secrets.

- DOCKERHUB_TOKEN:
  Go to account settings / security and click on New Access Token.



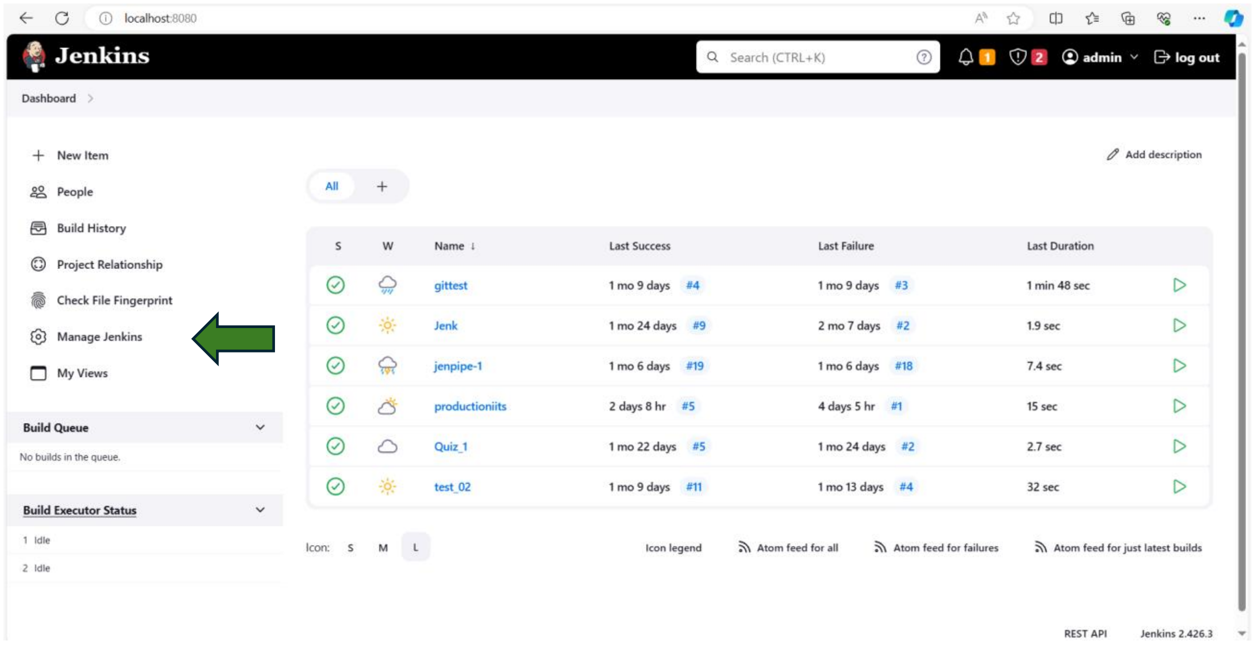
Tokens marked AUTO-GENERATED are created on your behalf by Docker Desktop for the CLI to use for authentication. You can have a maximum of 5 auto-generated tokens associated with your account. Learn more

Description	Source	Scope	Last Used	Created
Generated by Docker De...	AUTO-GENERATED	Read, Write, Delete	Mar 08, 2024 18:46:26	Mar 08, 2024 18:22:28
Generated by Docker De...	AUTO-GENERATED	Read, Write, Delete	Mar 08, 2024 18:39:23	Jan 15, 2024 20:04:44
github	MANUAL	Read, Write, Delete	Mar 01, 2024 17:45:58	Jan 30, 2024 19:19:38
github2	MANUAL	Read, Write, Delete	Jan 31, 2024 19:53:09	Jan 31, 2024 18:10:46

Give the token a name and click generate, then copy the token provided and save it in GitHub secrets as DOCKERHUB_TOKEN.

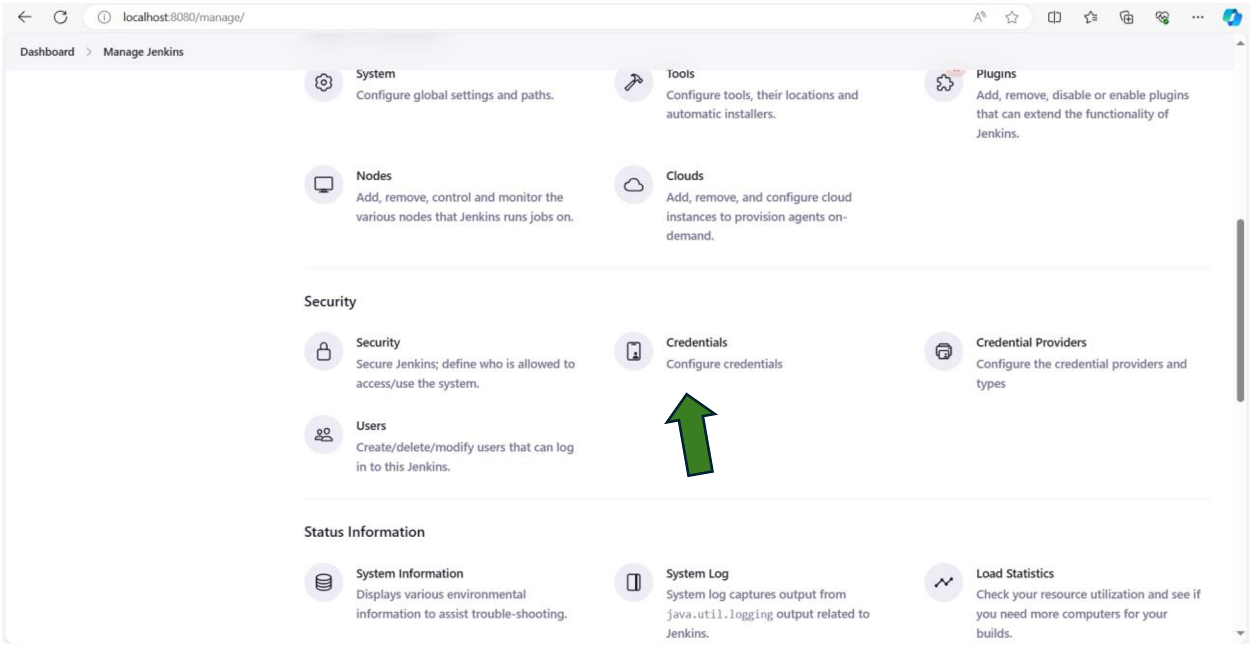
3.2 Jenkins Pipeline Setup

1. After opening Jenkins and running on the localhost:8080 in your browser, go to Manage Jenkins.

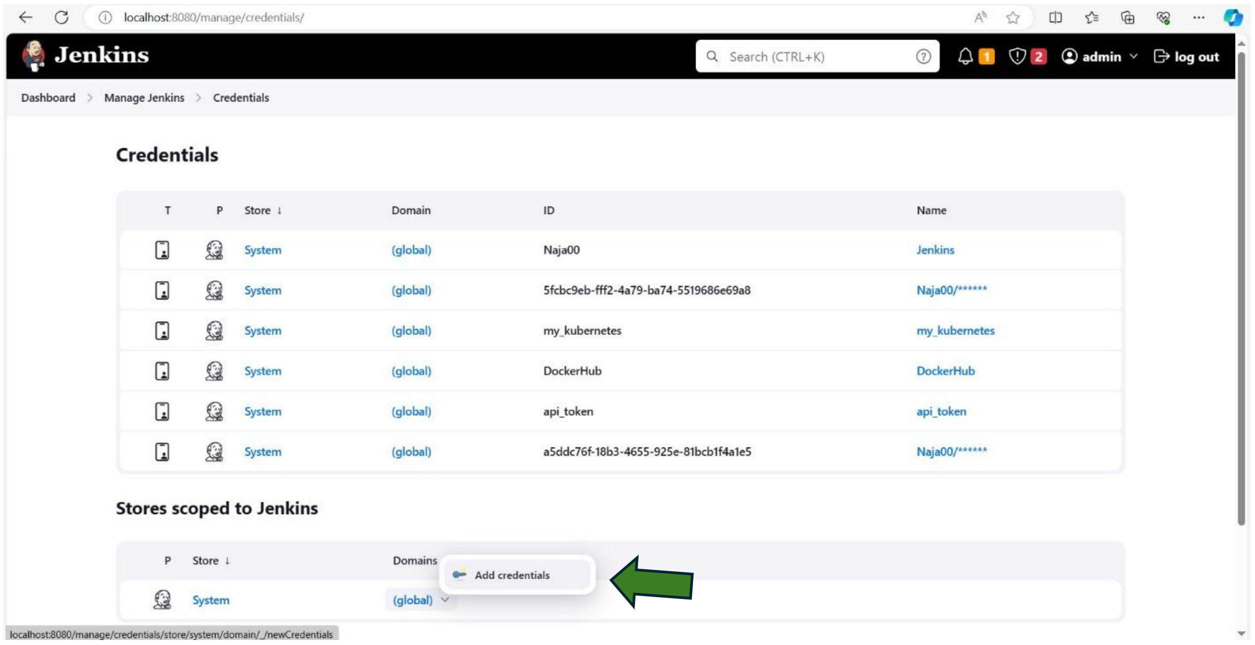


S	W	Name ↓	Last Success	Last Failure	Last Duration
✓		gittest	1 mo 9 days #4	1 mo 9 days #3	1 min 48 sec
✓		Jenk	1 mo 24 days #9	2 mo 7 days #2	1.9 sec
✓		jenpipe-1	1 mo 6 days #19	1 mo 6 days #18	7.4 sec
✓		productioniits	2 days 8 hr #5	4 days 5 hr #1	15 sec
✓		Quiz_1	1 mo 22 days #5	1 mo 24 days #2	2.7 sec
✓		test_02	1 mo 9 days #11	1 mo 13 days #4	32 sec

2. Scroll down for the security section and click on Credentials.


localhost:8080/manage/
Dashboard > Manage Jenkins
System
Configure global settings and paths.
Tools
Configure tools, their locations and automatic installers.
Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
Nodes
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
Clouds
Add, remove, and configure cloud instances to provision agents on-demand.
Security
Security
Secure Jenkins; define who is allowed to access/use the system.
Credentials
Configure credentials
Credential Providers
Configure the credential providers and types
Users
Create/delete/modify users that can log in to this Jenkins.
Status Information
System Information
Displays various environmental information to assist trouble-shooting.
System Log
System log captures output from java.util.logging output related to Jenkins.
Load Statistics
Check your resource utilization and see if you need more computers for your builds.

3. In the credentials section or store scoped to Jenkins move cursor under the Domain to global and press on the icon that appear, then click on Add credentials.



localhost:8080/manage/credentials/

Dashboard > Manage Jenkins > Credentials

Credentials

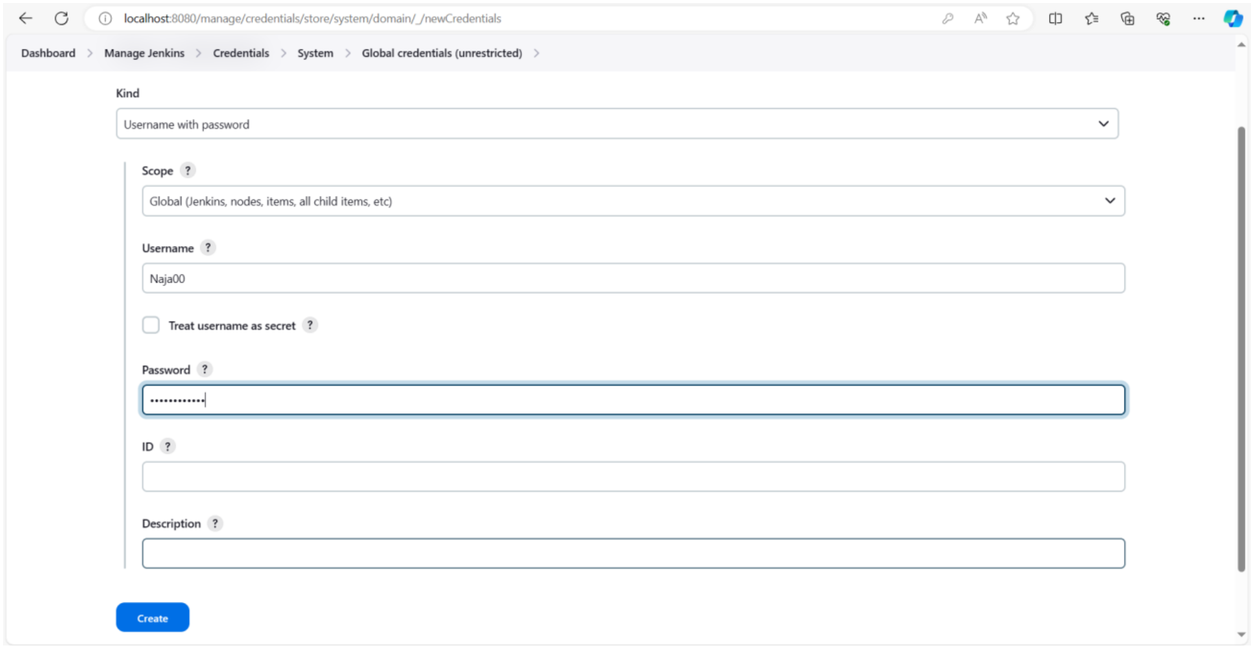
T	P	Store ↓	Domain	ID	Name
		System	(global)	Naja00	Jenkins
		System	(global)	5fcbc9eb-fff2-4a79-ba74-5519686e69a8	Naja00/******
		System	(global)	my_kubernetes	my_kubernetes
		System	(global)	DockerHub	DockerHub
		System	(global)	api_token	api_token
		System	(global)	a5ddc76f-18b3-4655-925e-81bcb1f4a1e5	Naja00/******

Stores scoped to Jenkins

P	Store ↓	Domains
	System	(global)

localhost:8080/manage/credentials/store/system/domain/_/newCredentials

4. Select the Kind: Username with password, Scope: Global(Jenkins, nodes, items, all child items, etc), Username: Enter your GitHub username, Password: Enter your GitHub password, you can keep the ID and Description empty, and then press Create.



localhost:8080/manage/credentials/store/system/domain/_/newCredentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: Naja00

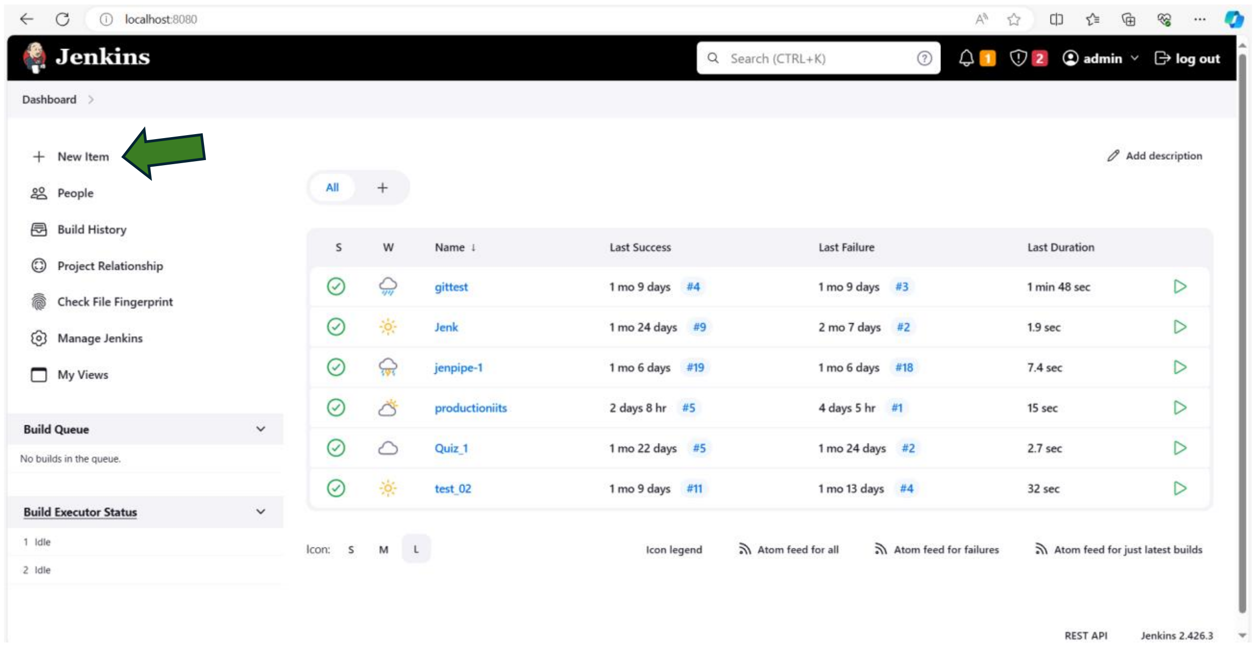
Treat username as secret

Password: ••••••••••••

ID:

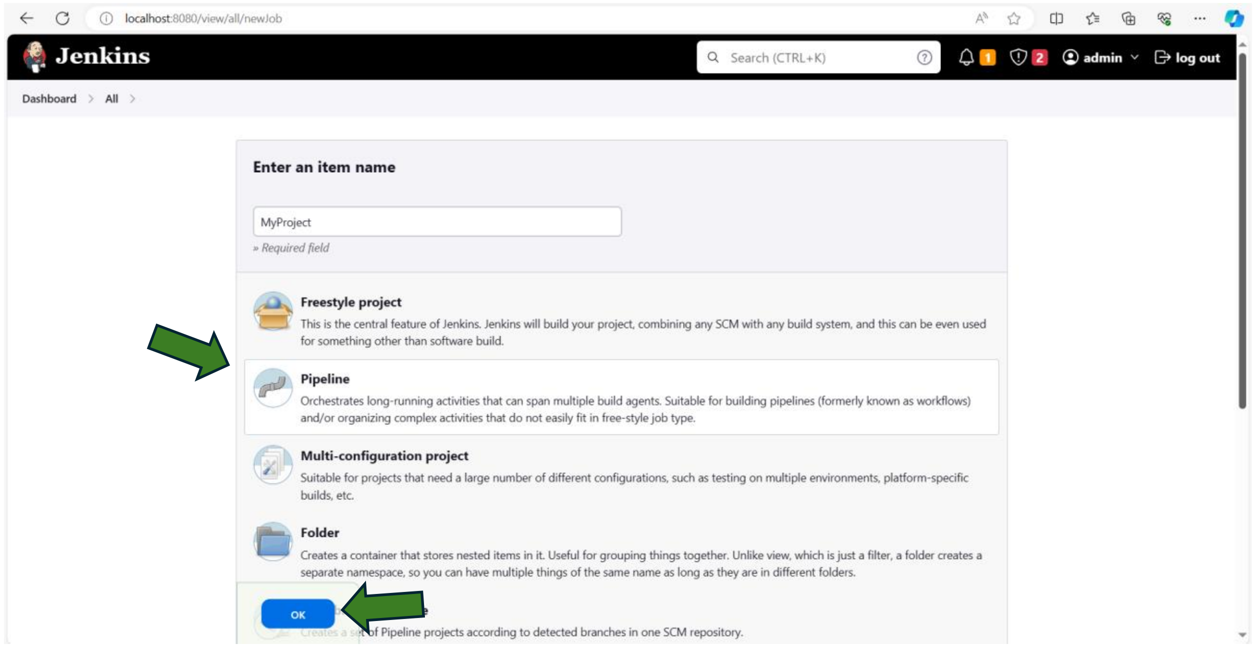
Description:

5. Select new item to create the pipeline.



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓		gittest	1 mo 9 days #4	1 mo 9 days #3	1 min 48 sec
✓		Jenk	1 mo 24 days #9	2 mo 7 days #2	1.9 sec
✓		jenpipe-1	1 mo 6 days #19	1 mo 6 days #18	7.4 sec
✓		productioniits	2 days 8 hr #5	4 days 5 hr #1	15 sec
✓		Quiz_1	1 mo 22 days #5	1 mo 24 days #2	2.7 sec
✓		test_02	1 mo 9 days #11	1 mo 13 days #4	32 sec

6. Enter a name for your pipeline and choose pipeline, then press OK.



Enter an item name

MyProject

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

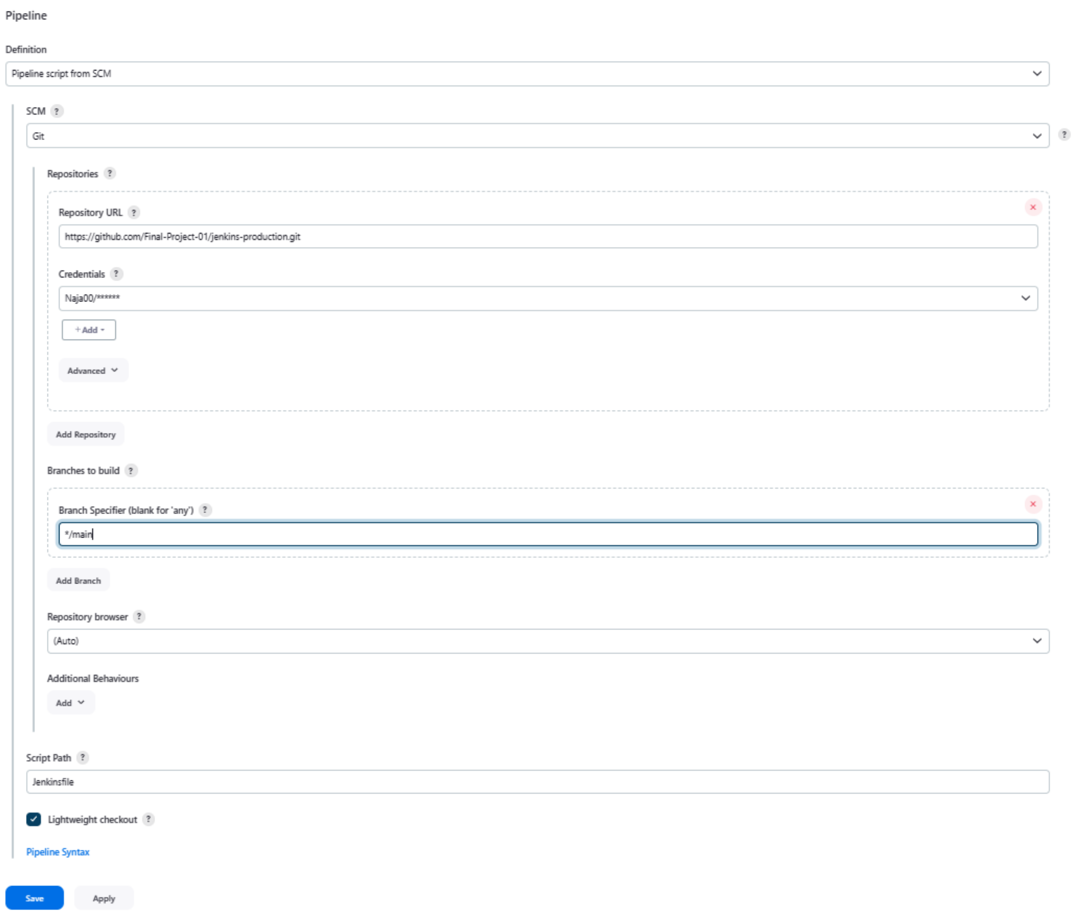
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Creates a set of Pipeline projects according to detected branches in one SCM repository.

7. Under the Pipeline section select the Definition: Pipeline script from SCM, for SCM choose Git, after that enter your repository URL of GitHub where Jenkins file is on, select the credential you added, and make sure to specify the branch, then click Save.


Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/Final-Project-01/jenkins-production.git

Credentials: Naja00/******

Branches to build

Branch Specifier (blank for 'any'): */main

Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

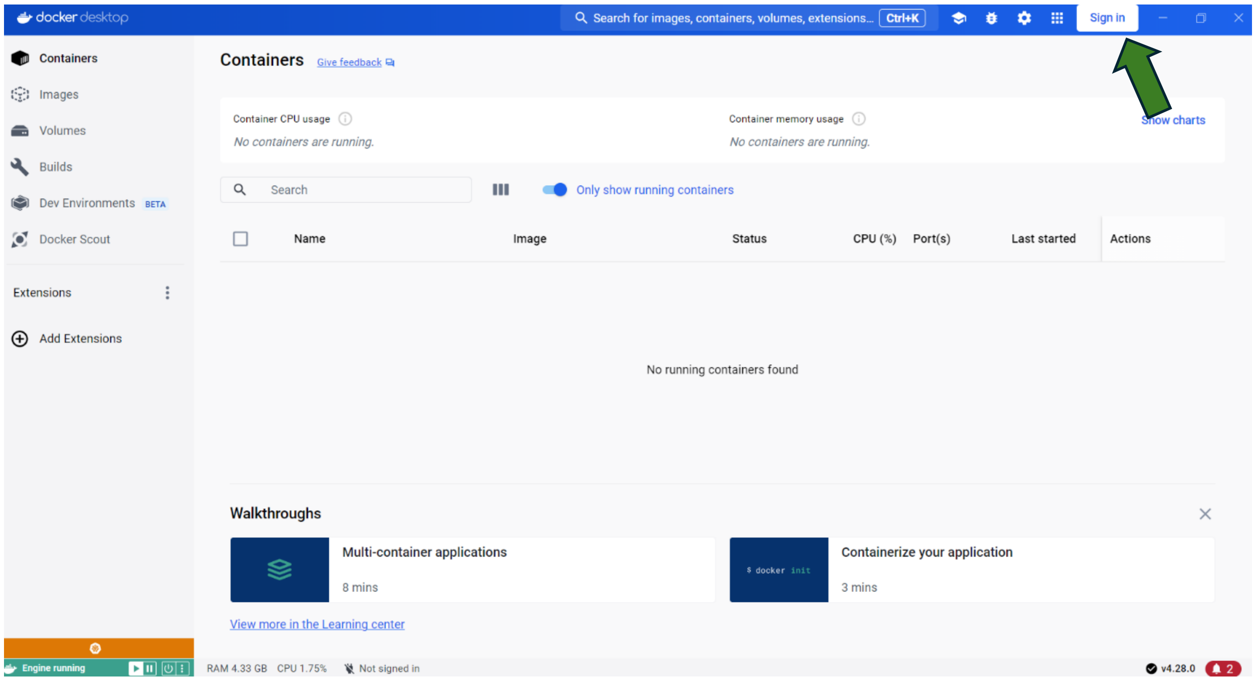
✓ Lightweight checkout

Pipeline Syntax

3.3 Docker Setup:

After downloading docker desktop on your computer follow these steps:

1. Click on the Sign in button on the right top of the screen.

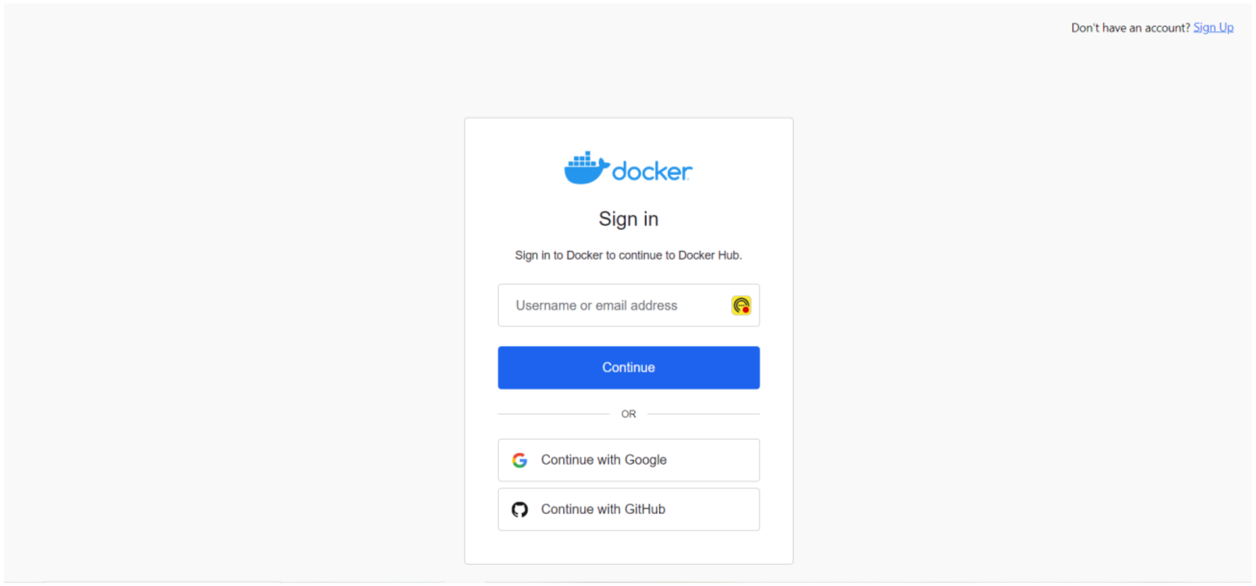


No containers are running.

No containers are running.

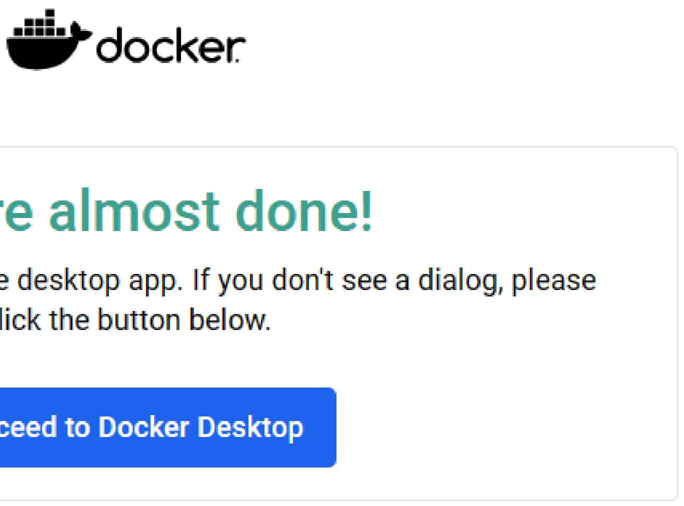
No running containers found

2. Sign in using your username or email address and password or continue with Google or GitHub (if you don't have an account click the sign up in the top right of the screen).



Sign in to Docker to continue to Docker Hub.

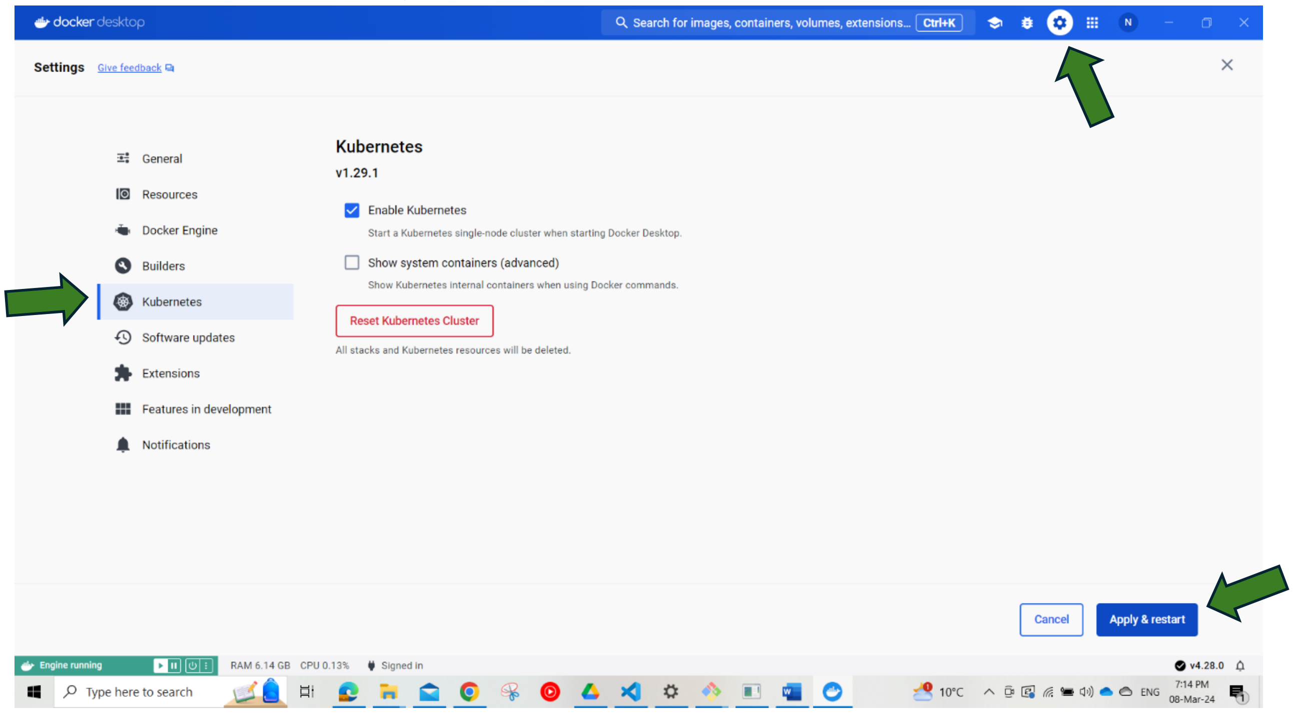
3. After signing in this window will show you to proceed to Docker Desktop and complete the setup of docker.



You're almost done!

We're redirecting you to the desktop app. If you don't see a dialog, please click the button below.

4. Then open the setting panel and go to the Kubernetes, check the Enable Kubernetes box, and click apply & restart.



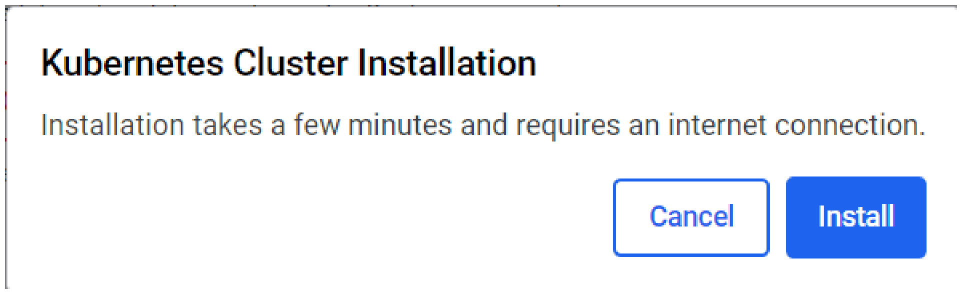
Kubernetes
v1.29.1

[✓] Enable Kubernetes
Start a Kubernetes single-node cluster when starting Docker Desktop.

[ ] Show system containers (advanced)
Show Kubernetes internal containers when using Docker commands.

Reset Kubernetes Cluster
All stacks and Kubernetes resources will be deleted.

5. This window will show up click install and wait few seconds (make sure you have internet connection).


Kubernetes Cluster Installation

Installation takes a few minutes and requires an internet connection.

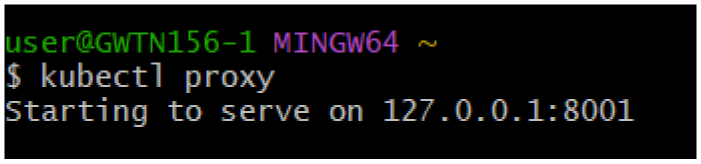
4. Deployment

4.1 Building Docker Images

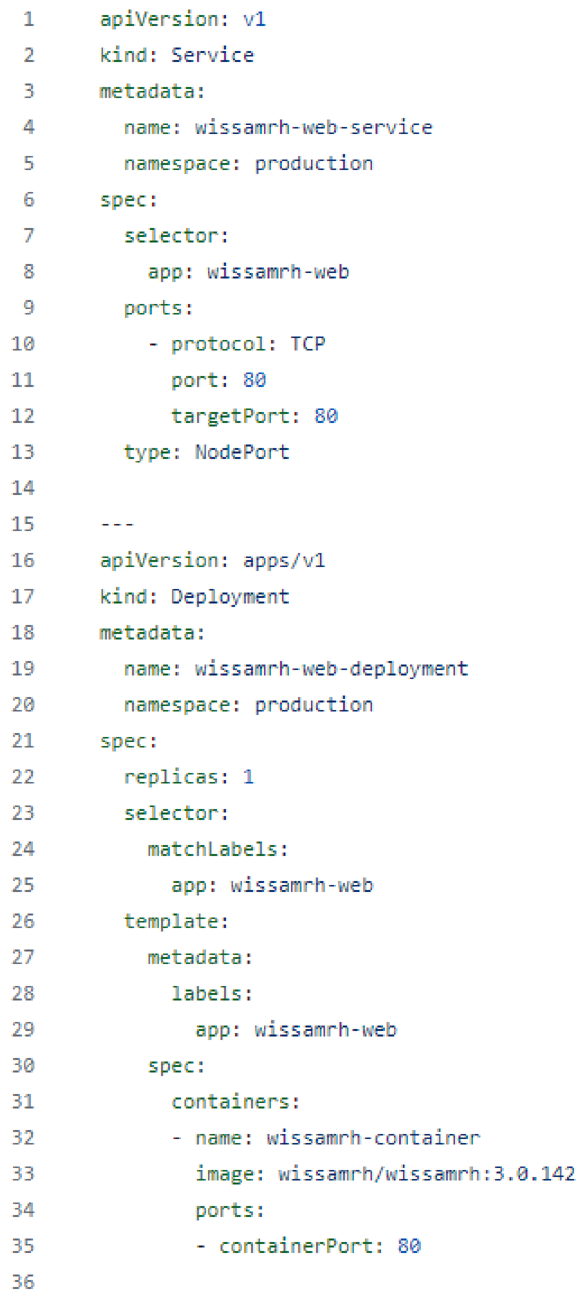
- As mentioned in the build.yml in the github actions it builds app docker image, database docker image and phpMyAdmin docker image on push.
- Then using the secrets provided Docker Hub is accessible by the yaml file and the docker images built are pushed to the selected repository.

4.2 Configuring Kubernetes Manifests

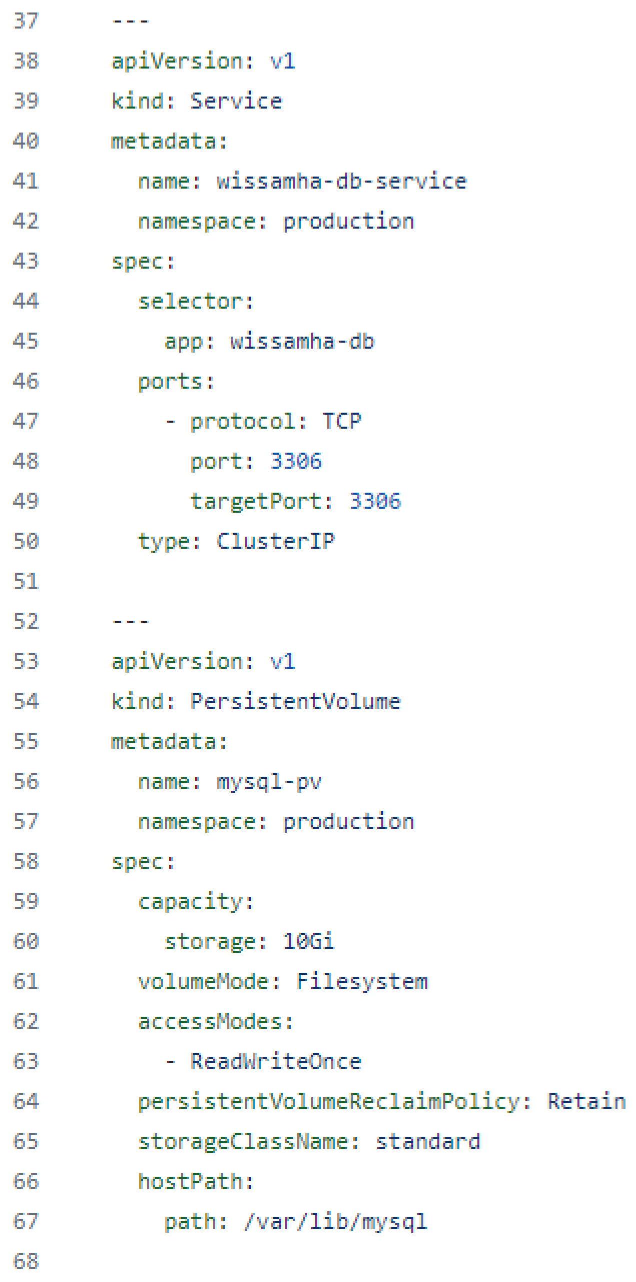
- Start Minikube, then run *kubectl proxy*.


```
user@GWTN156-1 MINGW64 ~
$ kubectl proxy
Starting to serve on 127.0.0.1:8001
```

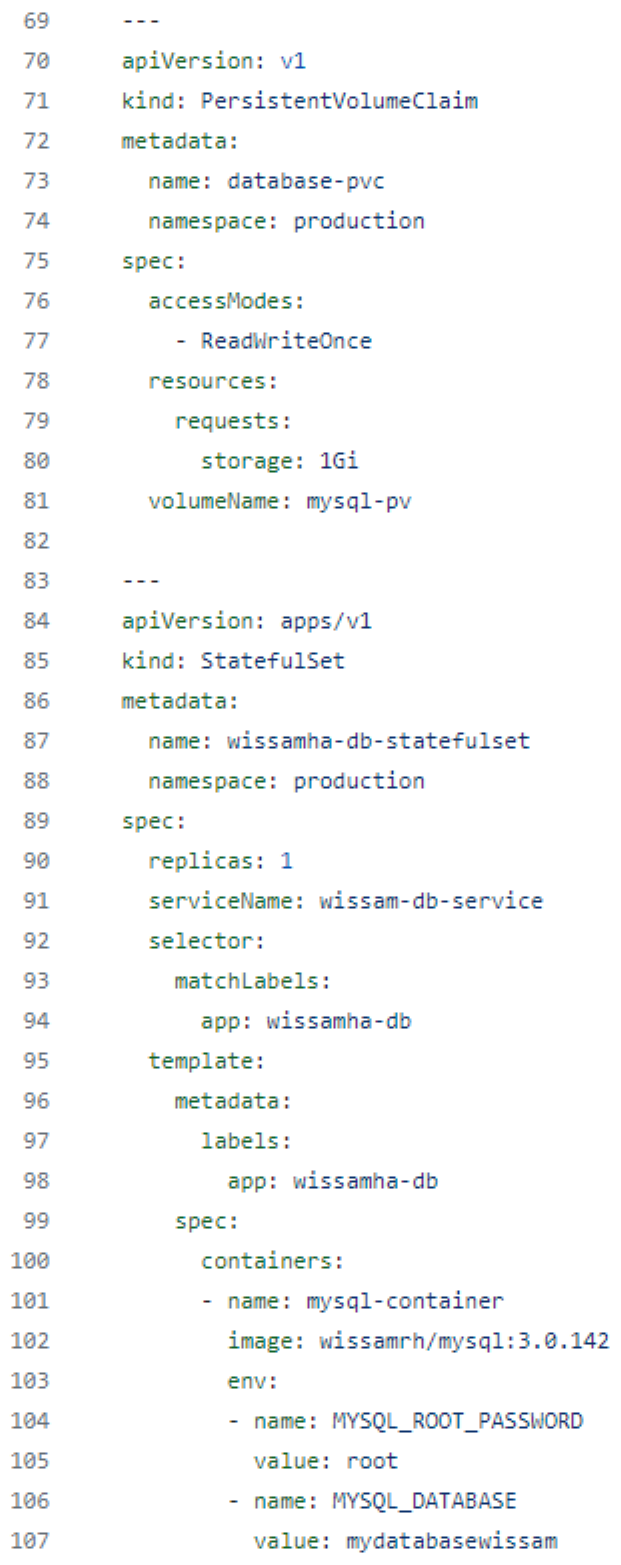
- Create the deploy.yml in the Jenkins repository.



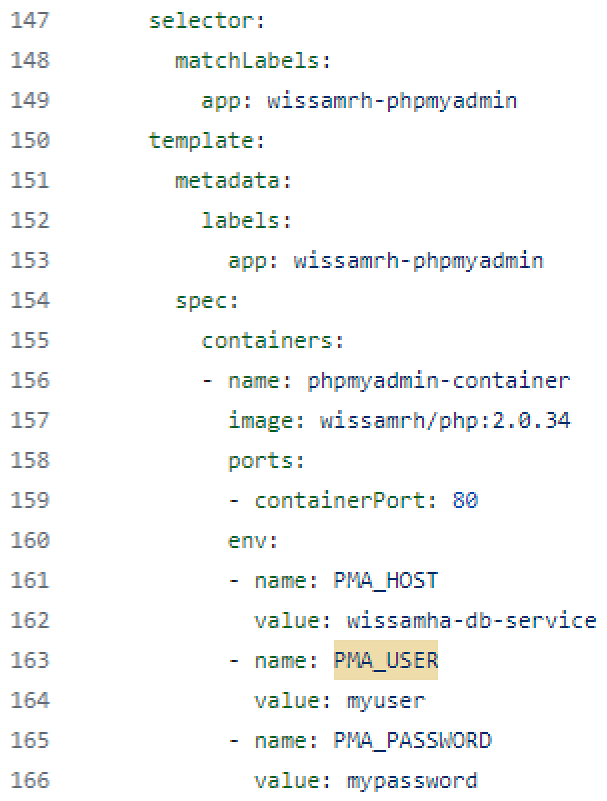
```yaml
apiVersion: v1
kind: Service
metadata:
  name: wissamrh-web-service
  namespace: production
spec:
  selector:
    app: wissamrh-web
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
  type: NodePort

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wissamrh-web-deployment
  namespace: production
spec:
  replicas: 1
  selector:
    matchLabels:
      app: wissamrh-web
  template:
    metadata:
      labels:
        app: wissamrh-web
    spec:
      containers:
      - name: wissamrh-container
        image: wissamrh/wissamrh:3.0.142
        ports:
        - containerPort: 80

```



```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: wissamha-db-service
  namespace: production
spec:
  selector:
    app: wissamha-db
  ports:
    - protocol: TCP
      port: 3306
      targetPort: 3306
  type: ClusterIP

---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mysql-pv
  namespace: production
spec:
  capacity:
    storage: 10Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: standard
  hostPath:
    path: /var/lib/mysql

```



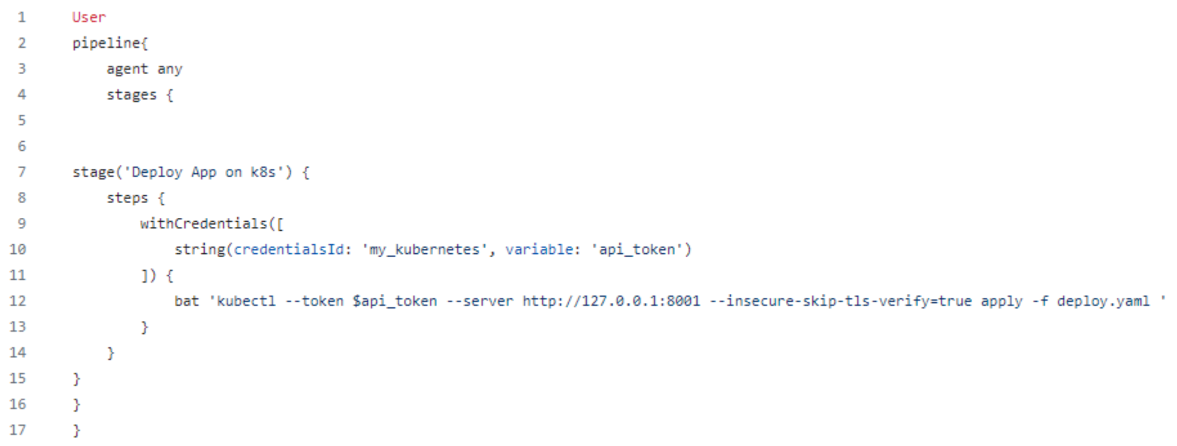
```yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: database-pvc
  namespace: production
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  volumeName: mysql-pv

---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: wissamha-db-statefulset
  namespace: production
spec:
  replicas: 1
  serviceName: wissam-db-service
  selector:
    matchLabels:
      app: wissamha-db
  template:
    metadata:
      labels:
        app: wissamha-db
    spec:
      containers:
      - name: mysql-container
        image: wissamrh/mysql:3.0.142
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: root
        - name: MYSQL_DATABASE
          value: mydatabasewissam
        - name: MYSQL_USER
          value: myuser
        - name: MYSQL_PASSWORD
          value: mypassword
        volumeMounts:
        - name: database-volume
          mountPath: /var/lib/mysql
  volumeClaimTemplates:
  - metadata:
      name: database-volume
    spec:
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 1Gi

---
apiVersion: v1
kind: Service
metadata:
  name: wissamrh-phpmyadmin-service
  namespace: production
spec:
  selector:
    app: wissamrh-phpmyadmin
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 80
  type: NodePort

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wissamrh-phpmyadmin-deployment
  namespace: production
spec:
  replicas: 1
```



```
  selector:
    matchLabels:
      app: wissamrh-phpmyadmin
  template:
    metadata:
      labels:
        app: wissamrh-phpmyadmin
    spec:
      containers:
      - name: phpmyadmin-container
        image: wissamrh/php:2.0.34
        ports:
        - containerPort: 80
        env:
        - name: PMA_HOST
          value: wissamha-db-service
        - name: PMA_USER
          value: myuser
        - name: PMA_PASSWORD
          value: mypassword
```

Make sure that the docker image tag is like the last tag created.

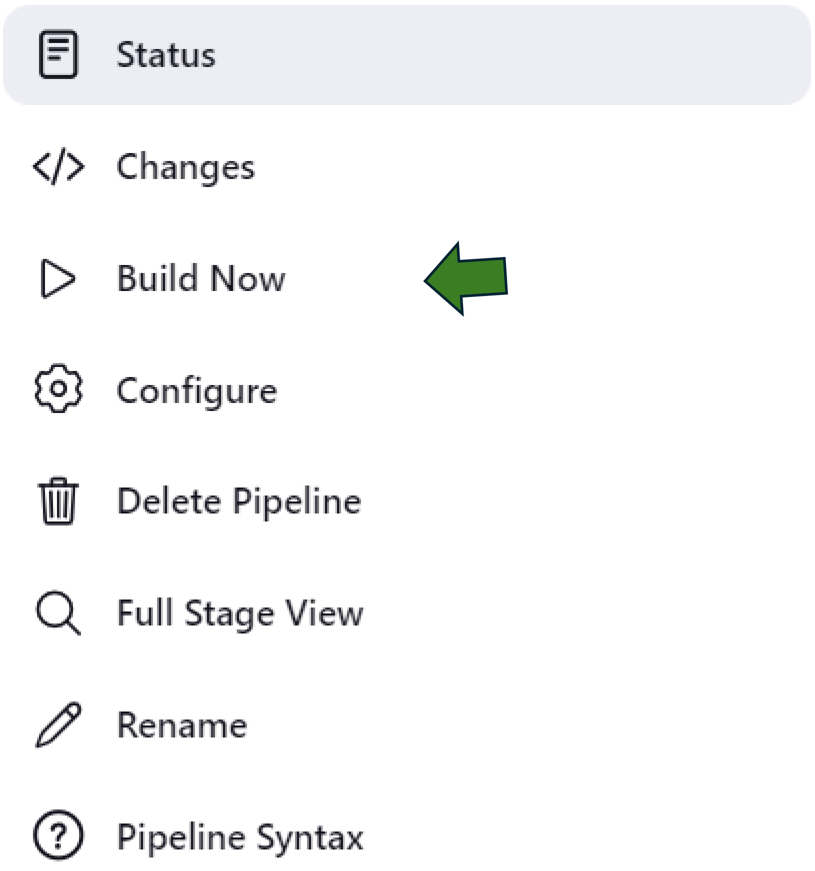
- Create a JenkinsFile.



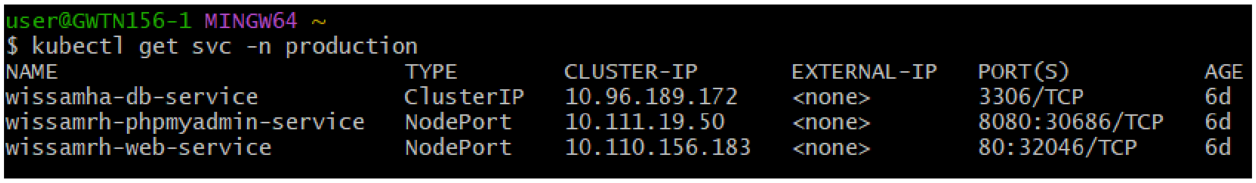
```
User
pipeline{
    agent any
    stages {


stage('Deploy App on k8s') {
    steps {
        withCredentials([
            string(credentialsId: 'my_kubernetes', variable: 'api_token')
        ]) {
            bat 'kubectl --token $api_token --server http://127.0.0.1:8001 --insecure-skip-tls-verify=true apply -f deploy.yaml '
        }
    }
}
}
}
```

4.3 Deploying to Minikube

Go to Jenkins and open the pipeline you created and click Build Now.

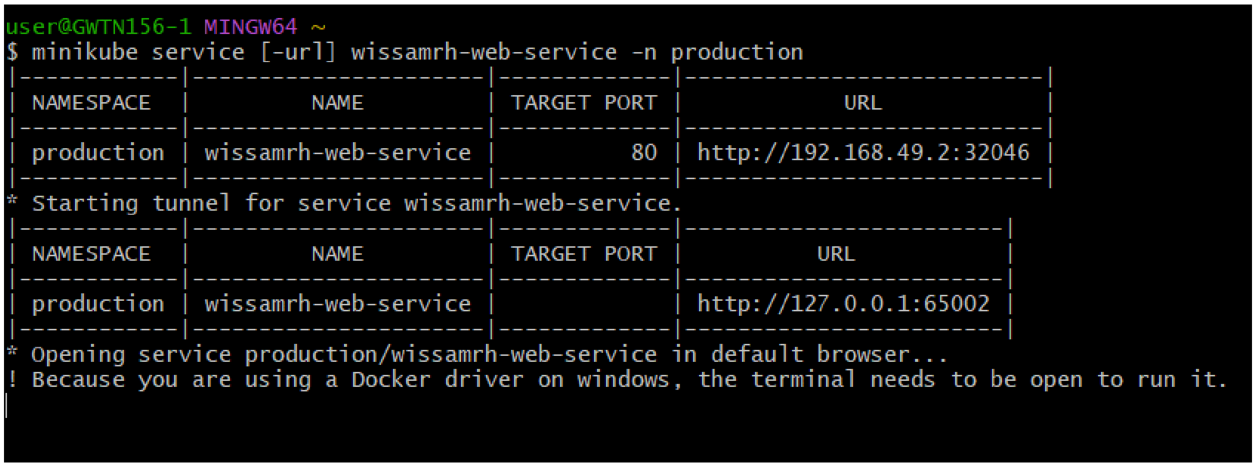


- Open another terminal and run *kubectl get svc -n production*.



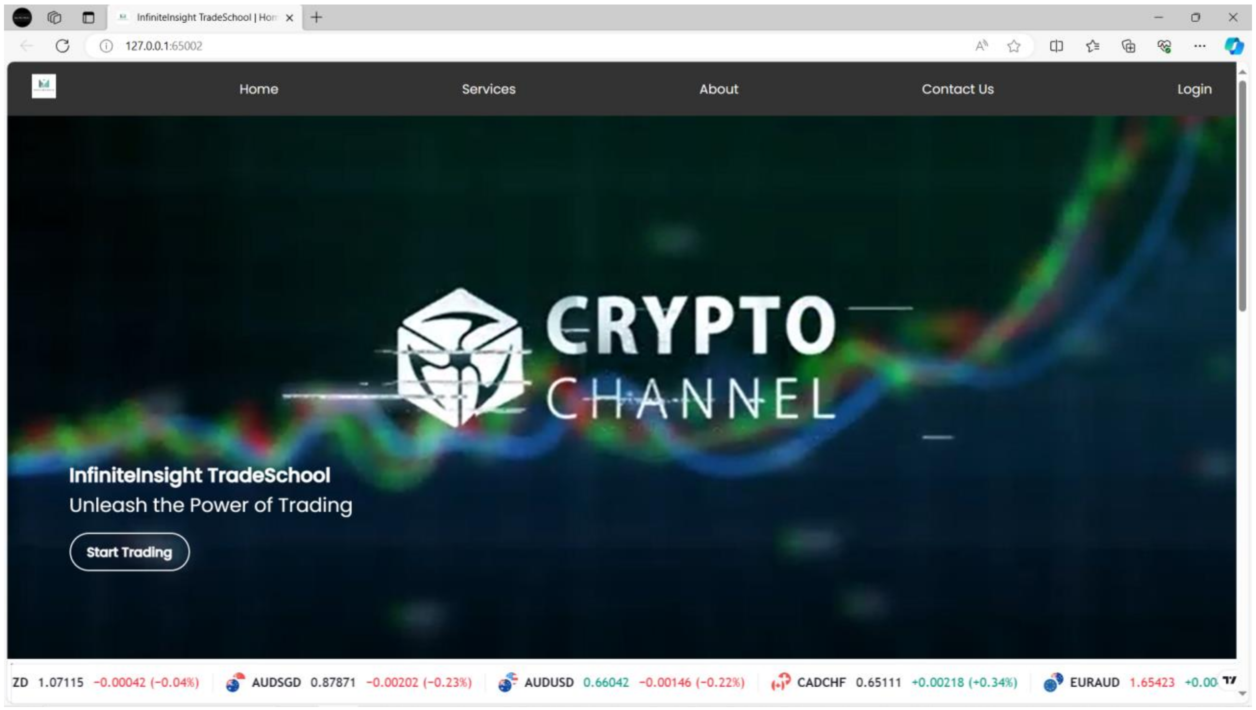
```
user@GWTN156-1 MINGW64 ~
$ kubectl get svc -n production
NAME                          TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
wissamha-db-service           ClusterIP   10.96.189.172    <none>        3306/TCP         6d
wissamrh-phpmyadmin-service   NodePort    10.111.19.50     <none>        8080:30686/TCP   6d
wissamrh-web-service          NodePort    10.110.156.183   <none>        80:32046/TCP     6d
```

- Then run *minikube service [-url] wissamrh-web-service -n production*.



```
user@GWTN156-1 MINGW64 ~
$ minikube service [-url] wissamrh-web-service -n production
|------------|----------------------|-------------|---------------------------|
| NAMESPACE  |         NAME         | TARGET PORT |            URL            |
|------------|----------------------|-------------|---------------------------|
| production | wissamrh-web-service |          80 | http://192.168.49.2:32046 |
|------------|----------------------|-------------|---------------------------|
* Starting tunnel for service wissamrh-web-service.
|------------|----------------------|-------------|------------------------|
| NAMESPACE  |         NAME         | TARGET PORT |          URL           |
|------------|----------------------|-------------|------------------------|
| production | wissamrh-web-service |             | http://127.0.0.1:65002 |
|------------|----------------------|-------------|------------------------|
* Opening service production/wissamrh-web-service in default browser...
! Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

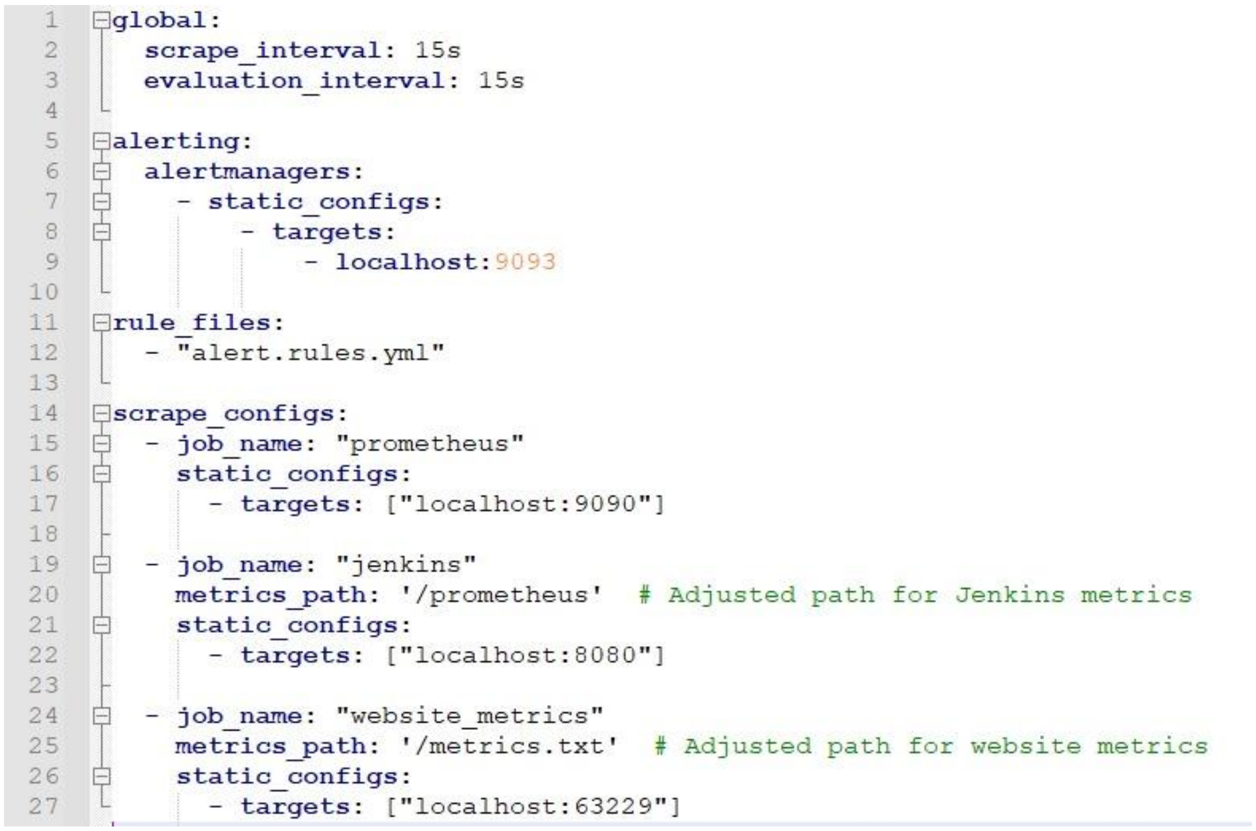
- The website will automatically open in your browser.


InfiniteInsight TradeSchool | Hom
127.0.0.1:65002
Home
Services
About
Contact Us
Login
CRYPTO
CHANNEL
InfiniteInsight TradeSchool
Unleash the Power of Trading
Start Trading
ZD 1.07115 -0.00042 (-0.04%)
AUDSGD 0.87871 -0.00202 (-0.23%)
AUDUSD 0.66042 -0.00146 (-0.22%)
CADCHF 0.65111 +0.00218 (+0.34%)
EURAUD 1.65423 +0.00

5. Monitoring

5.1 Prometheus Setup

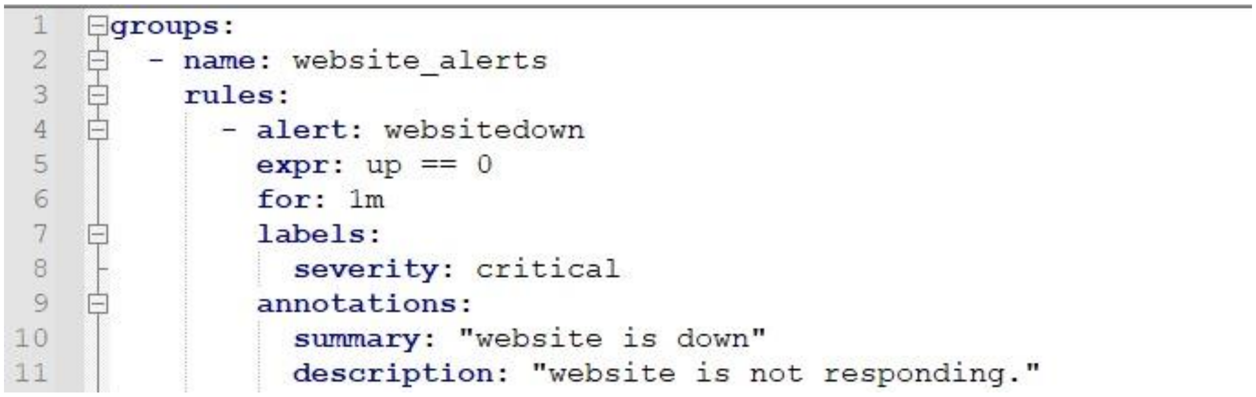
- Modify your prometheus.yaml file in order to monitor Jenkins and the website.



```yaml
global:
  scrape_interval: 15s
  evaluation_interval: 15s

alerting:
  alertmanagers:
    - static_configs:
        - targets:
            - localhost:9093

rule_files:
  - "alert.rules.yml"

scrape_configs:
  - job_name: "prometheus"
    static_configs:
      - targets: ["localhost:9090"]

  - job_name: "jenkins"
    metrics_path: '/prometheus'  # Adjusted path for Jenkins metrics
    static_configs:
      - targets: ["localhost:8080"]

  - job_name: "website_metrics"
    metrics_path: '/metrics.txt'  # Adjusted path for website metrics
    static_configs:
      - targets: ["localhost:63229"]
```

Make sure the port of the localhost of your website matches the one provided by Minikube.

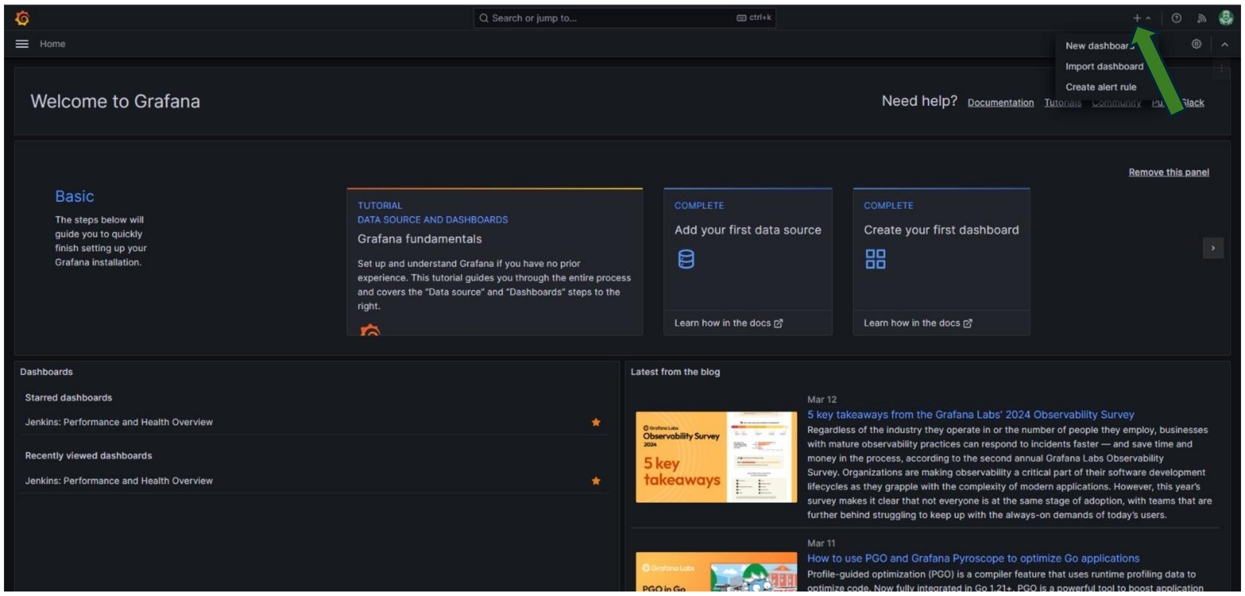
- Modify the alert.rules.yml in the same directory as the Prometheus.yml.



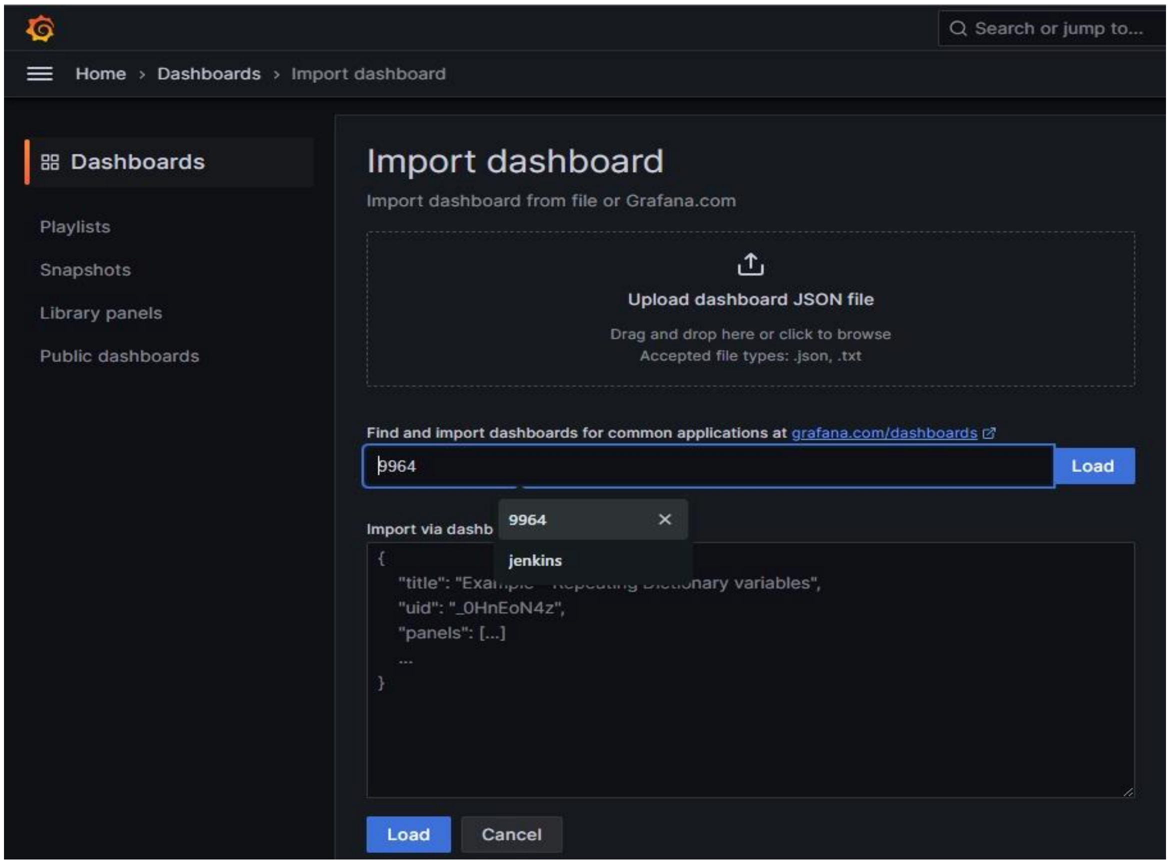
```yaml
groups:
  - name: website_alerts
    rules:
      - alert: websitedown
        expr: up == 0
        for: 1m
        labels:
          severity: critical
        annotations:
          summary: "website is down"
          description: "website is not responding."
```

5.2 Grafana Dashboard Setup

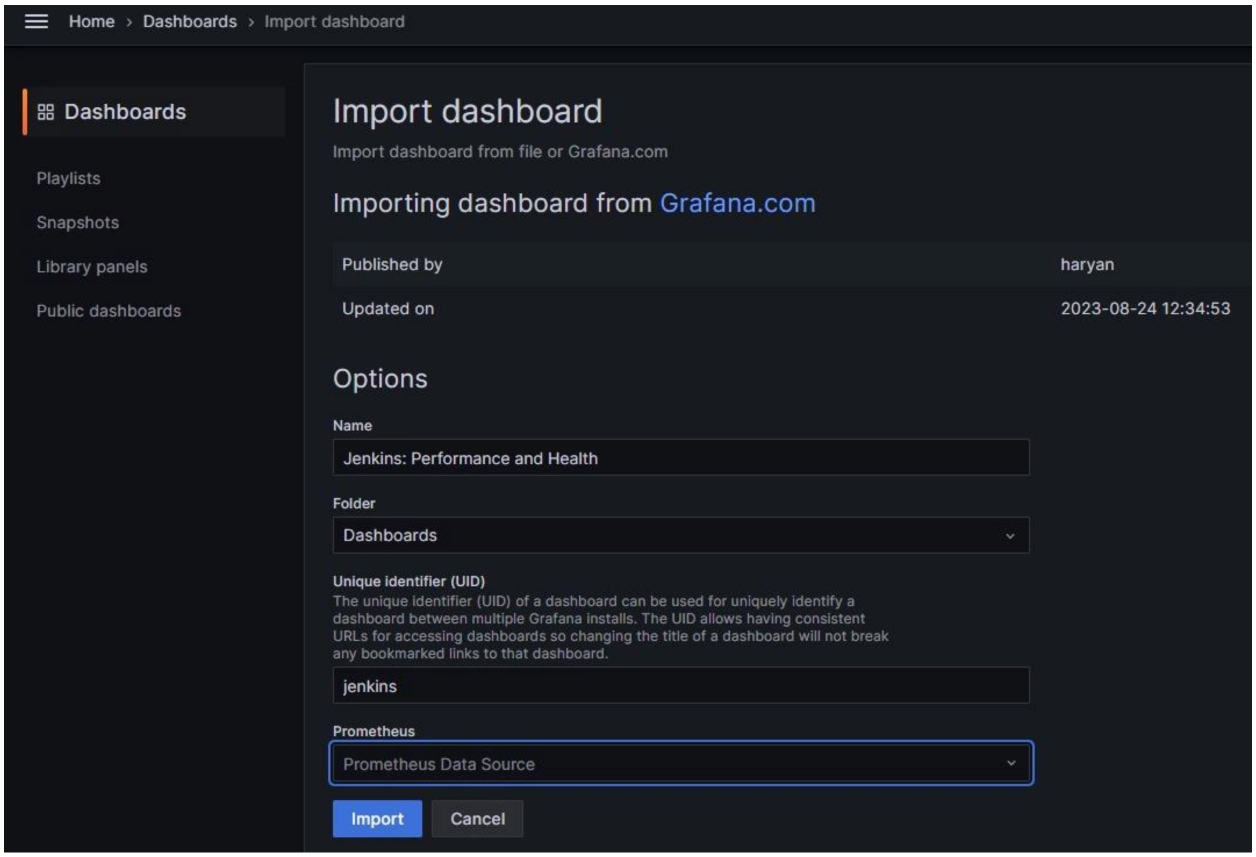
- Login to Grafana (localhost:3000) click on the (+) button, then Import Dashboard.


Search or jump to...
ctrl+k
Home
New dashboard
Import dashboard
Create alert rule
Welcome to Grafana
Need help? Documentation Tutorials Community Public Slack
Remove this panel
Basic
The steps below will guide you to quickly finish setting up your Grafana installation.
TUTORIAL
DATA SOURCE AND DASHBOARDS
Grafana fundamentals
Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the "Data source" and "Dashboards" steps to the right.
COMPLETE
Add your first data source
Learn how in the docs
COMPLETE
Create your first dashboard
Learn how in the docs
Dashboards
Starred dashboards
Jenkins: Performance and Health Overview
Recently viewed dashboards
Jenkins: Performance and Health Overview
Latest from the blog
Mar 12
5 key takeaways from the Grafana Labs' 2024 Observability Survey
Mar 11
How to use PGO and Grafana Pyroscope to optimize Go applications

- Enter 9964 as an ID for the dashboard to be used in monitoring and click Load.


Search or jump to...
Home › Dashboards › Import dashboard
Dashboards
Playlists
Snapshots
Library panels
Public dashboards
Import dashboard
Import dashboard from file or Grafana.com
Upload dashboard JSON file
Drag and drop here or click to browse
Accepted file types: .json, .txt
Find and import dashboards for common applications at grafana.com/dashboards
9964
Load
9964
jenkins
Import via dashb
Load
Cancel

- Choose Prometheus as a data source, then click Import.


Home › Dashboards › Import dashboard

Dashboards
Playlists
Snapshots
Library panels
Public dashboards

Import dashboard

Import dashboard from file or Grafana.com

Importing dashboard from Grafana.com

Published by	haryan
Updated on	2023-08-24 12:34:53

Options

Name: Jenkins: Performance and Health

Folder: Dashboards

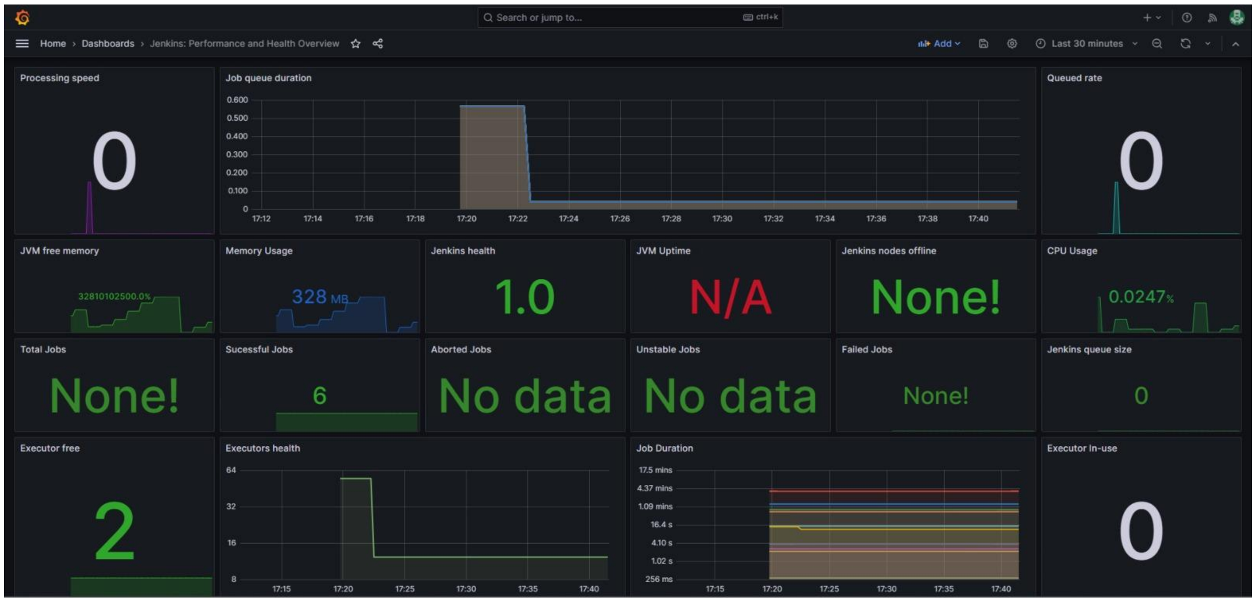
Unique identifier (UID)
The unique identifier (UID) of a dashboard can be used for uniquely identify a dashboard between multiple Grafana installs. The UID allows having consistent URLs for accessing dashboards so changing the title of a dashboard will not break any bookmarked links to that dashboard.

jenkins

Prometheus: Prometheus Data Source

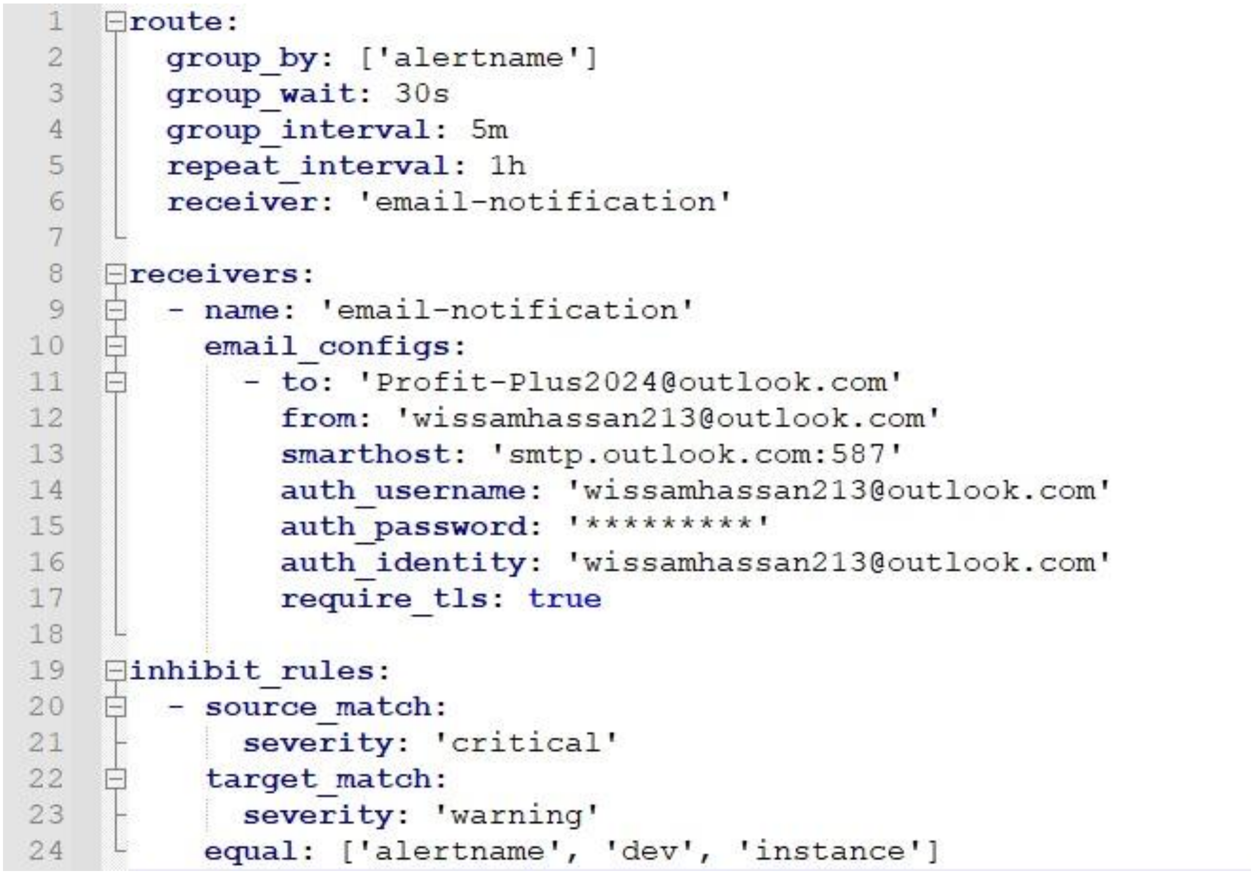
Import Cancel

- After importing this dashboard will appear and you can monitor Jenkins.



5.3 Alert Manager Setup

- Modify alertmanager.yml to send emails in case of firing of the targets for Prometheus to the selected email.



```yaml
route:
  group_by: ['alertname']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 1h
  receiver: 'email-notification'

receivers:
  - name: 'email-notification'
    email_configs:
      - to: 'Profit-Plus2024@outlook.com'
        from: 'wissamhassan213@outlook.com'
        smarthost: 'smtp.outlook.com:587'
        auth_username: 'wissamhassan213@outlook.com'
        auth_password: '*********'
        auth_identity: 'wissamhassan213@outlook.com'
        require_tls: true

inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```

Make sure to change the email_configs and put your email and password, plus the receiver email.

6. Collaboration and Version Control

6.1 GitHub Repository Structure

The structure of the repositories should be as the following:

- Source code and workflows to build docker images.
- Jenkins file with the deployment file that contains the latest image tag built for staging and production environment (two repositories).

6.2 Branching Strategy

In the source code repository must have two branches first for dev, second for staging and testing, third for the production and final release.

7. Conclusion

In conclusion, this documentation aims to guide you through a successful deployment on a Kubernetes cluster using a combination of powerful tools. For any questions or further assistance, please reach out to Profit-Plus2024@outlook.com.