CI/CD Pipeline with Jenkins, SonarQube, Nexus, Slack Notifications

📌 Project Overview

This project demonstrates a **full CI/CD pipeline** setup using **Jenkins, Git, Maven, SonarQube, and Nexus Repository Manager**, with Slack integration for real-time notifications.

It showcases industry-standard DevOps practices and tools commonly used in FAANG-level environments. The goal is to:

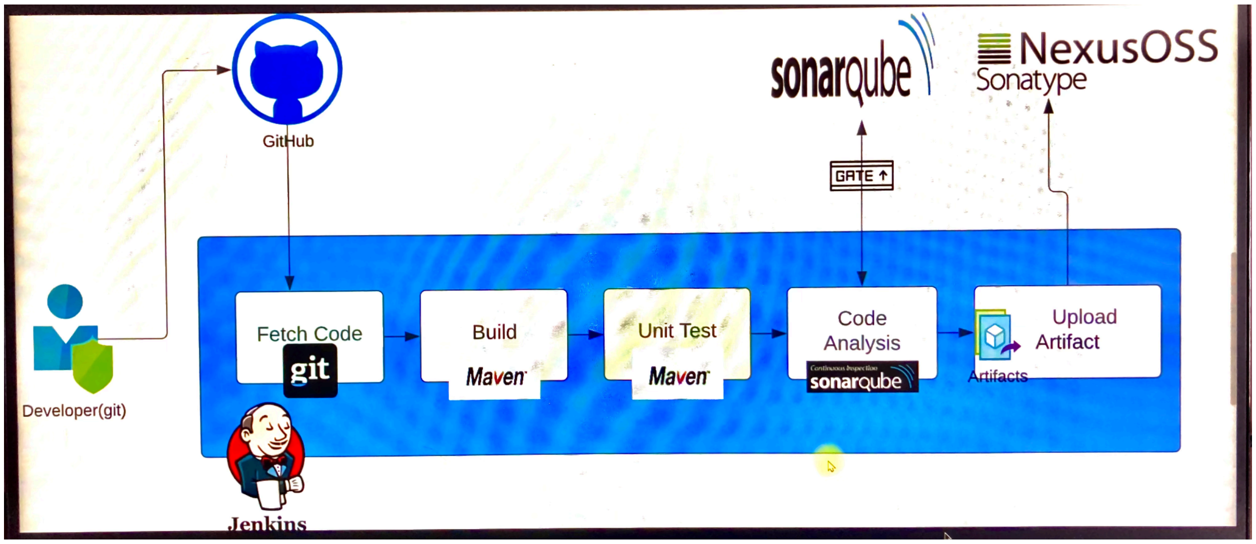
- Automate the **build, test, analysis, and artifact management process**.
- Ensure **code quality and security** through SonarQube's Quality Gates.
- Store and version **artifacts** in Nexus Repository.
- Notify the team on Slack about build results.

This end-to-end pipeline mirrors what top tech companies use to manage their CI/CD workflows.

🛠️ Tools & Technologies

- **Jenkins** → CI/CD Orchestrator
- **GitHub** → Source code repository
- **Maven** → Build & unit testing
- **SonarQube** → Static code analysis + quality gates
- **Nexus OSS** → Artifact repository manager
- **Slack** → Team notifications
- **AWS EC2** → Infrastructure (Jenkins, SonarQube, Nexus servers)

🌐 Architecture Diagram

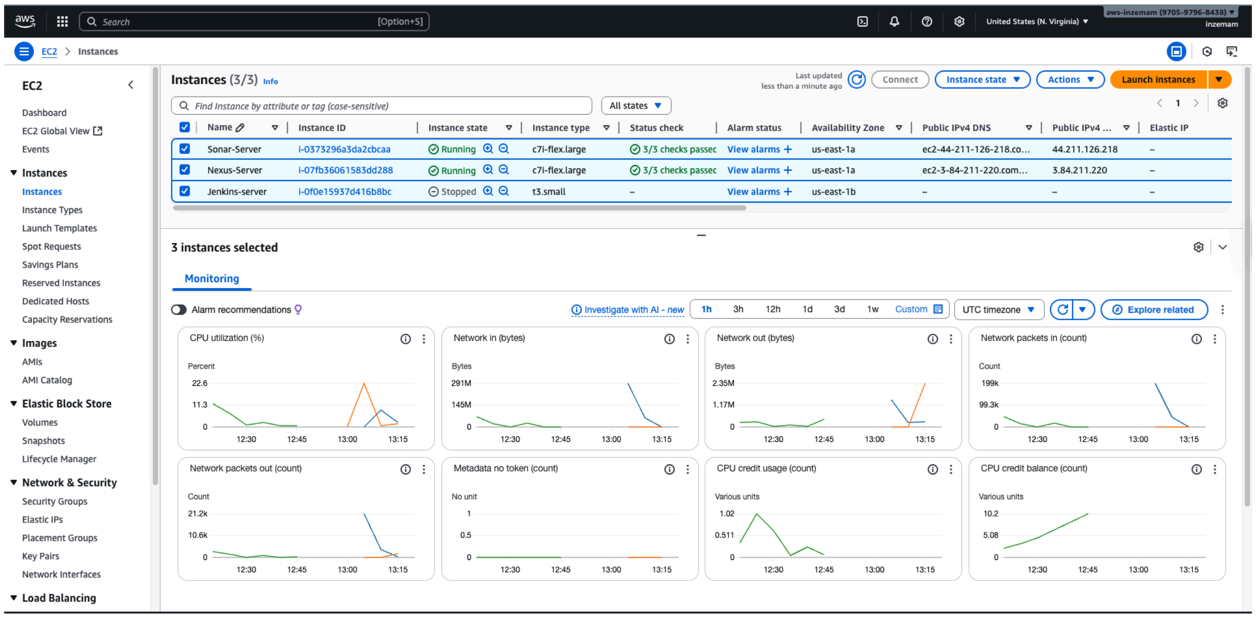


`CI-CD-Jenkins-Sonar-NexusOSS.png`

⚙️ Infrastructure Setup

1. AWS EC2 Instances

- Jenkins → Amazon Linux 2023
- Nexus → Amazon Linux 2023 (Port: 8081)
- SonarQube → Ubuntu 24 (Port: 80/9000)



Instances (3/3) Info

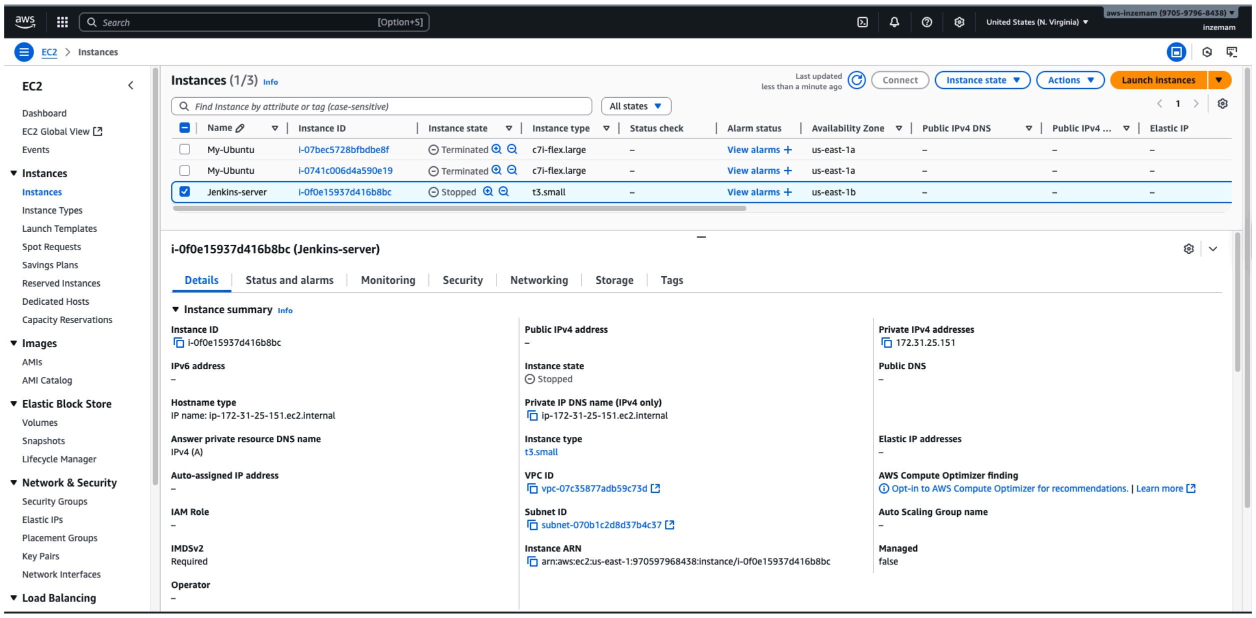
Last updated less than a minute ago

Find Instance by attribute or tag (case-sensitive) · All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Sonar-Server	i-0373296a3da2cbcaa	Running	c7i-flex.large	3/3 checks passec	View alarms +	us-east-1a	ec2-44-211-126-218.co...	44.211.126.218	–
Nexus-Server	i-07fb36061583dd288	Running	c7i-flex.large	3/3 checks passec	View alarms +	us-east-1a	ec2-3-84-211-220.com...	3.84.211.220	–
Jenkins-server	i-0f0e15937d416b8bc	Stopped	t3.small	–	View alarms +	us-east-1b	–	–	–

3 instances selected

created-ec2-jenkins-nexus-sonarq.png



Instances (1/3) Info

Last updated less than a minute ago

Find Instance by attribute or tag (case-sensitive) · All states

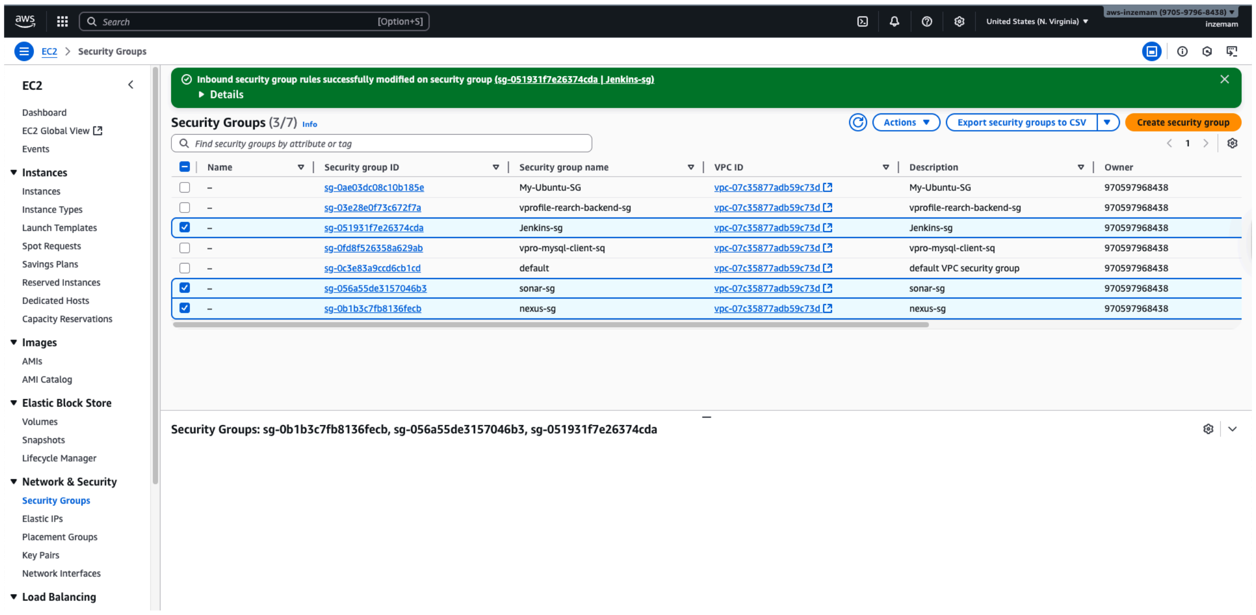
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
My-Ubuntu	i-07bec5728bfbdbe8f	Terminated	c7i-flex.large	–	View alarms +	us-east-1a	–	–	–
My-Ubuntu	i-0741c006d4a590e19	Terminated	c7i-flex.large	–	View alarms +	us-east-1a	–	–	–
Jenkins-server	i-0f0e15937d416b8bc	Stopped	t3.small	–	View alarms +	us-east-1b	–	–	–

i-0f0e15937d416b8bc (Jenkins-server)

Instance summary Info

Instance ID i-0f0e15937d416b8bc	**Public IPv4 address** –	**Private IPv4 addresses** 172.31.25.151
IPv6 address –	**Instance state** Stopped	**Public DNS** –
Hostname type IP name: ip-172-31-25-151.ec2.internal	**Private IP DNS name (IPv4 only)** ip-172-31-25-151.ec2.internal	
Answer private resource DNS name IPv4 (A)	**Instance type** t3.small	**Elastic IP addresses** –
Auto-assigned IP address –	**VPC ID** vpc-07c35877adb59c73d	**AWS Compute Optimizer finding** Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role –	**Subnet ID** subnet-070b1c2d8d37b4c37	**Auto Scaling Group name** –
IMDSv2 Required	**Instance ARN** arn:aws:ec2:us-east-1:970597968438:instance/i-0f0e15937d416b8bc	**Managed** false
Operator –		

Created-Jenkins_EC2.png


Inbound security group rules successfully modified on security group (sg-051931f7e26374cda | Jenkins-sg)

Security Groups (3/7)

	Name	Security group ID	Security group name	VPC ID	Description	Owner
☐	–	sg-0ae03dc08c10b185e	My-Ubuntu-SG	vpc-07c35877adb59c73d	My-Ubuntu-SG	970597968438
☐	–	sg-03e28e0f73c672f7a	vprofile-rearch-backend-sg	vpc-07c35877adb59c73d	vprofile-rearch-backend-sg	970597968438
☑	–	sg-051931f7e26374cda	Jenkins-sg	vpc-07c35877adb59c73d	Jenkins-sg	970597968438
☐	–	sg-0fd8f526358a629ab	vpro-mysql-client-sq	vpc-07c35877adb59c73d	vpro-mysql-client-sq	970597968438
☐	–	sg-0c3e83a9ccd6cb1cd	default	vpc-07c35877adb59c73d	default VPC security group	970597968438
☑	–	sg-056a55de3157046b3	sonar-sg	vpc-07c35877adb59c73d	sonar-sg	970597968438
☑	–	sg-0b1b3c7fb8136fecb	nexus-sg	vpc-07c35877adb59c73d	nexus-sg	970597968438

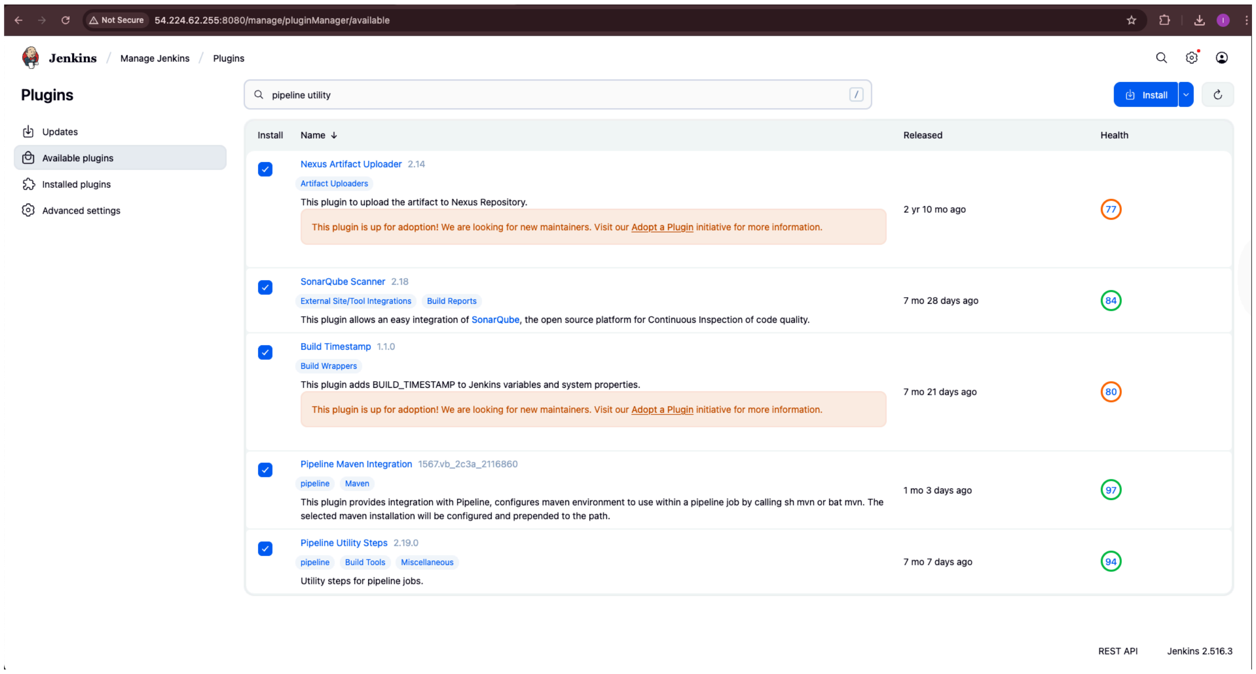
Security Groups: sg-0b1b3c7fb8136fecb, sg-056a55de3157046b3, sg-051931f7e26374cda

created-security-grp-jenkins-nexus-sonarq.png

🔧 Jenkins Setup

1. Install Jenkins and required plugins:
    - Git
    - Maven Integration
    - SonarQube Scanner
    - Nexus Artifact Uploader
    - Build Timestamp Plugin
    - Slack Notification Plugin
2. Configure credentials:
    - GitHub (SSH key or token)
    - Nexus (admin/admin123 for demo)

- SonarQube token
- Slack token



Install	Name	Released	Health
✓	Nexus Artifact Uploader 2.14 Artifact Uploaders This plugin to upload the artifact to Nexus Repository. This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	2 yr 10 mo ago	77
✓	SonarQube Scanner 2.18 External Site/Tool Integrations · Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	7 mo 28 days ago	84
✓	Build Timestamp 1.1.0 Build Wrappers This plugin adds BUILD_TIMESTAMP to Jenkins variables and system properties. This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	7 mo 21 days ago	80
✓	Pipeline Maven Integration 1567.vb_2c3a_2116860 pipeline · Maven This plugin provides integration with Pipeline, configures maven environment to use within a pipeline job by calling sh mvn or bat mvn. The selected maven installation will be configured and prepended to the path.	1 mo 3 days ago	97
✓	Pipeline Utility Steps 2.19.0 pipeline · Build Tools · Miscellaneous Utility steps for pipeline jobs.	7 mo 7 days ago	94

REST API Jenkins 2.516.3

`jenkins-plugins-installed.png`

🧩 Pipeline Stages

1. Source Code Checkout

- Jenkins pulls code from GitHub (`vprofile-devops` repo).

2. Build with Maven

- Run `mvn clean install`.
- Generate `.war` artifact (`vprofile-v2.war`).

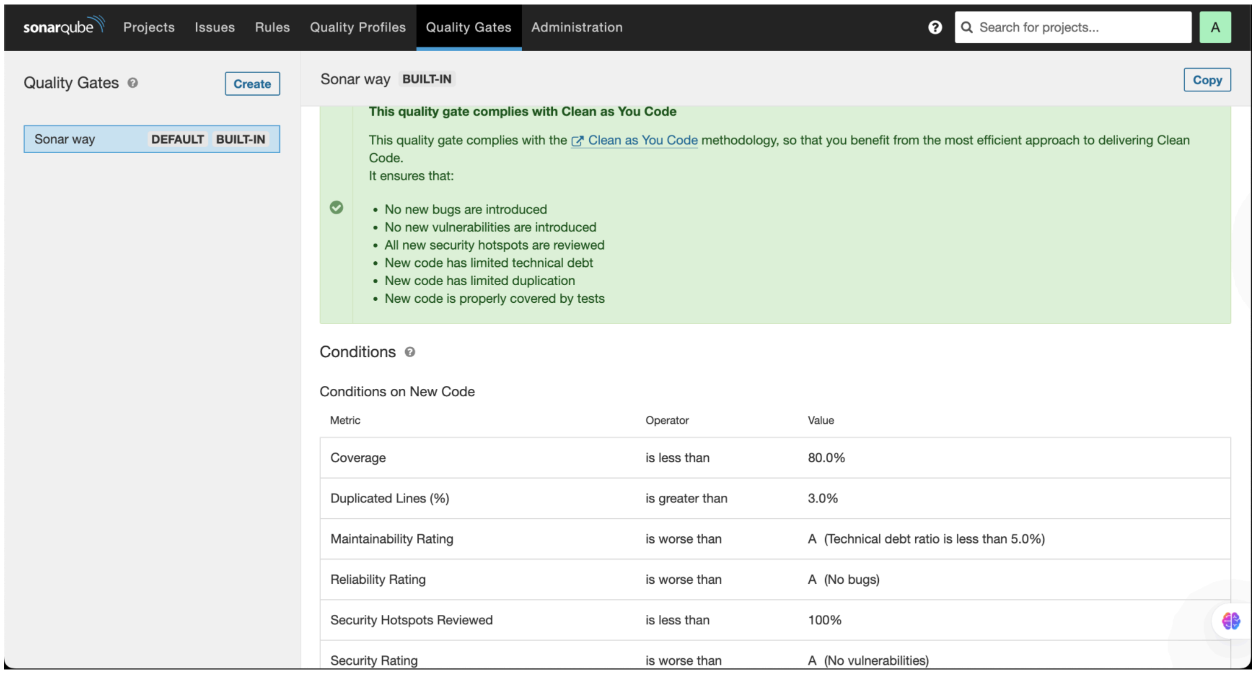
3. Unit Testing

- Execute `mvn test`.
- Test reports generated.

4. Static Code Analysis (SonarQube)

- Code scanned for bugs, vulnerabilities, and code smells.
- Enforce **SonarQube Quality Gates**.

Placeholder:



This quality gate complies with Clean as You Code

This quality gate complies with the Clean as You Code methodology, so that you benefit from the most efficient approach to delivering Clean Code.
It ensures that:

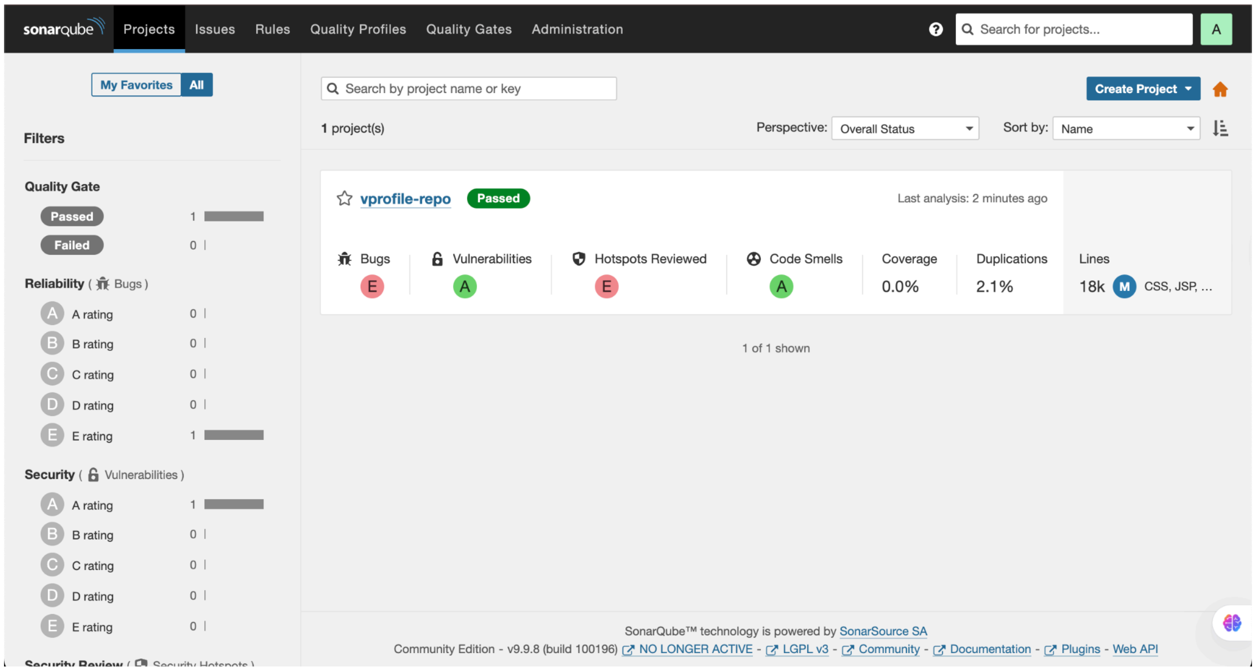
- No new bugs are introduced
- No new vulnerabilities are introduced
- All new security hotspots are reviewed
- New code has limited technical debt
- New code has limited duplication
- New code is properly covered by tests

Conditions

Conditions on New Code

Metric	Operator	Value
Coverage	is less than	80.0%
Duplicated Lines (%)	is greater than	3.0%
Maintainability Rating	is worse than	A (Technical debt ratio is less than 5.0%)
Reliability Rating	is worse than	A (No bugs)
Security Hotspots Reviewed	is less than	100%
Security Rating	is worse than	A (No vulnerabilities)

`sonarqube-quality-gates.png`



Search by project name or key

1 project(s) Perspective: Overall Status Sort by: Name

Filters

Quality Gate

Passed	1	
Failed	0	

Reliability (Bugs)

A	A rating	0
B	B rating	0
C	C rating	0
D	D rating	0
E	E rating	1

Security (Vulnerabilities)

A	A rating	1
B	B rating	0
C	C rating	0
D	D rating	0
E	E rating	0

Security Review (Security Hotspots)

vprofile-repo Passed Last analysis: 2 minutes ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
E	A	E	A	0.0%	2.1%	18k M CSS, JSP, ...

1 of 1 shown

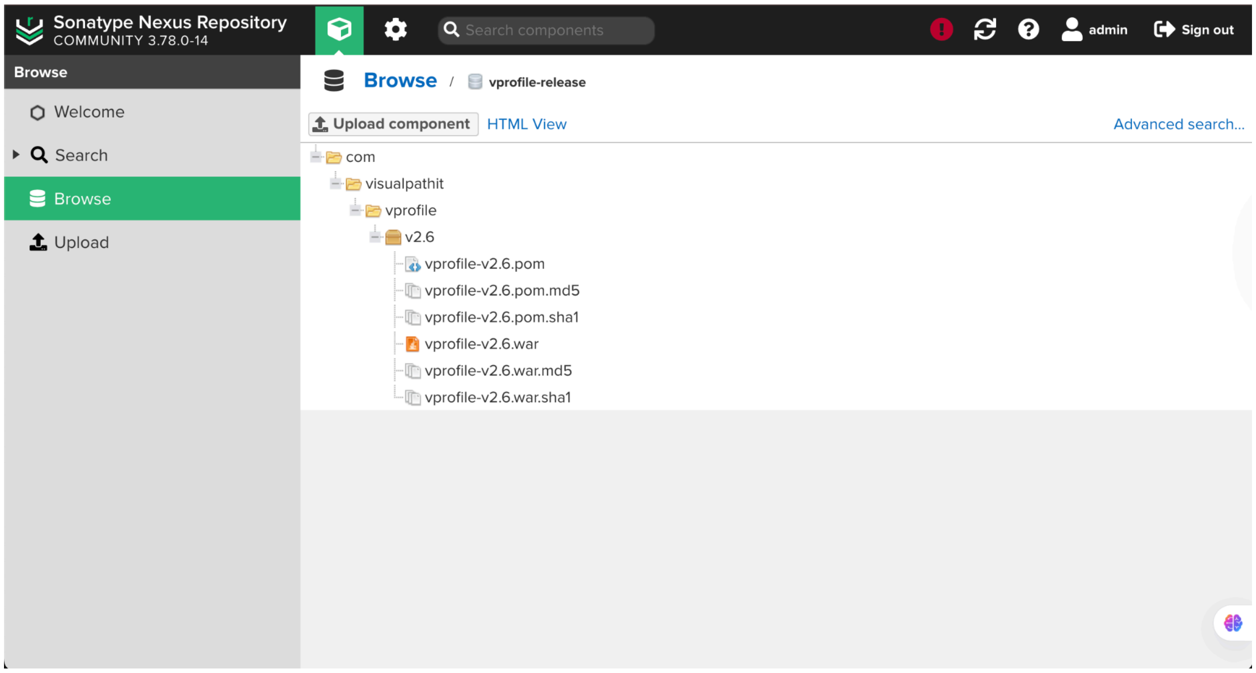
SonarQube™ technology is powered by SonarSource SA
Community Edition - v9.9.8 (build 100196) NO LONGER ACTIVE - LGPL v3 - Community - Documentation - Plugins - Web API

`sonarqube-tests-passed.png`

5. Artifact Upload (Nexus Repository)

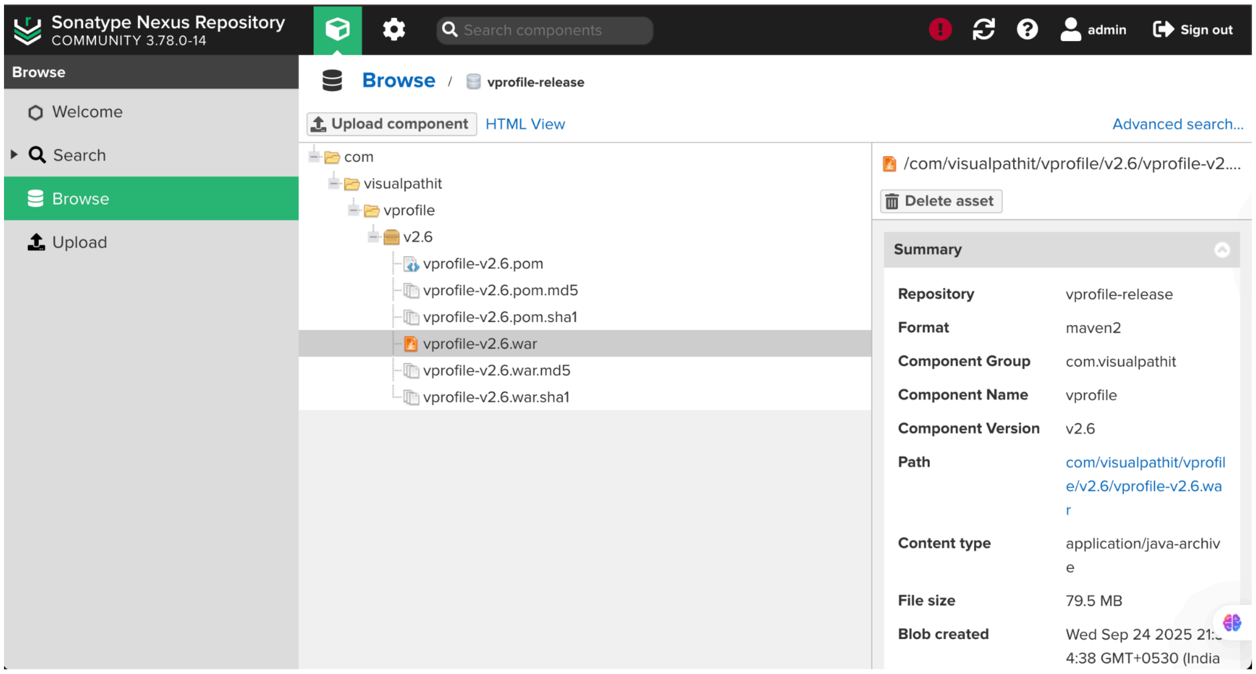
- Use Nexus Artifact Uploader plugin.
- Dynamic versioning using:
  - `${env.BUILD_ID}`
  - `${env.BUILD_TIMESTAMP}`
- Artifacts stored in Nexus under `vprofile-repo`.

Placeholder:



```
com
  visualpathit
    vprofile
      v2.6
        vprofile-v2.6.pom
        vprofile-v2.6.pom.md5
        vprofile-v2.6.pom.sha1
        vprofile-v2.6.war
        vprofile-v2.6.war.md5
        vprofile-v2.6.war.sha1
```

nexus-repo-build-uploaded.png



```
com
  visualpathit
    vprofile
      v2.6
        vprofile-v2.6.pom
        vprofile-v2.6.pom.md5
        vprofile-v2.6.pom.sha1
        vprofile-v2.6.war
        vprofile-v2.6.war.md5
        vprofile-v2.6.war.sha1
```

/com/visualpathit/vprofile/v2.6/vprofile-v2....

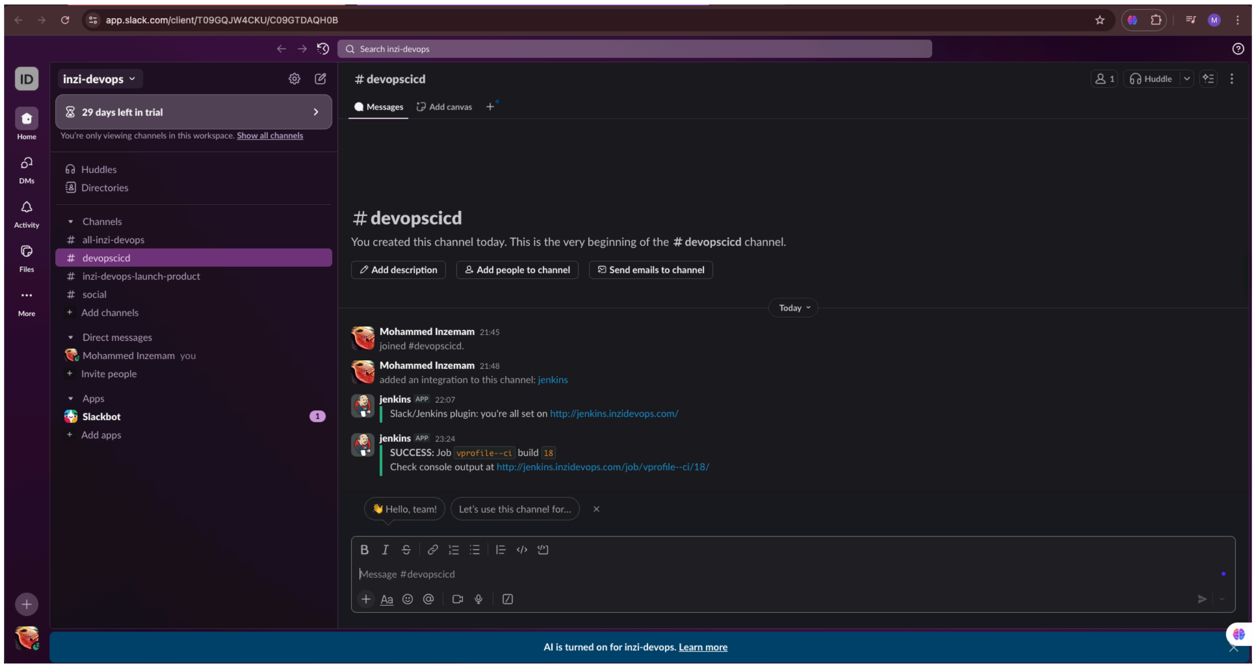
Repository	vprofile-release
Format	maven2
Component Group	com.visualpathit
Component Name	vprofile
Component Version	v2.6
Path	com/visualpathit/vprofile/v2.6/vprofile-v2.6.war
Content type	application/java-archive
File size	79.5 MB
Blob created	Wed Sep 24 2025 21: 4:38 GMT+0530 (India

nexus-repo-build-uploaded-detailed.png

6. Notifications (Slack)

- Slack plugin sends messages to `#devopscicd` channel.
- Notifications color-coded:
    - 🟢 Success = Green
    - 🔴 Failure = Red

Placeholder:



devopscicd

You created this channel today. This is the very beginning of the # devopscicd channel.

Mohammed Inzemam 21:45
joined #devopscicd.

Mohammed Inzemam 21:48
added an integration to this channel: jenkins

jenkins APP 22:07
Slack/Jenkins plugin: you're all set on http://jenkins.inzidevops.com/

jenkins APP 23:24
SUCCESS: Job `vprofile--ci` build `18`
Check console output at http://jenkins.inzidevops.com/job/vprofile--ci/18/

`jenkins-sent-notifications-to-slack.png`

📜 Jenkinsfile (Pipeline as Code)

```
def COLOR_MAP(status) {
    if (status == 'SUCCESS') {
        return 'good'
    } else if (status == 'FAILURE') {
        return 'danger'
    }
}

pipeline {
```

```
agent any

tools {
    maven 'Maven'
    jdk 'JDK11'
}

stages {
    stage('Checkout') {
        steps {
            git branch: 'main', url: 'https://github.com/mohammedinzi/vprofile-devops.git'
        }
    }

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

    stage('Unit Test') {
        steps {
            sh 'mvn test'
        }
    }

    stage('SonarQube Analysis') {
        steps {
            withSonarQubeEnv('SonarQubeServer') {
                sh 'mvn sonar:sonar'
            }
        }
    }

    stage('Upload Artifact to Nexus') {
        steps {
            nexusArtifactUploader(
                nexusVersion: 'nexus3',
                protocol: 'http',
                nexusUrl: 'http://<NEXUS_PRIVATE_IP>:8081',
                groupId: 'com.vprofile',
                version: "${env.BUILD_ID}-${env.BUILD_TIMESTAMP}",
                repository: 'vprofile-repo',
                credentialsId: 'nexus-login',
                artifacts: [
                    [artifactId: 'vpro-app', classifier: '', file: 'target/vprofile-v2.war', type: 'war']
                ]
            )
        }
    }
}

    post {
        always {
            echo 'Slack Notification'
            slackSend(
                channel: '#devopscicd',
                color: COLOR_MAP(currentBuild.currentResult),
                message: "Job ${env.JOB_NAME} build ${env.BUILD_NUMBER} -
${currentBuild.currentResult}"
            )
        }
    }
}
```

🐛 Troubleshooting & Challenges

1. Jenkins Disk Space Issue

- Error: *"Waiting for next available executor"*
- Cause: Jenkins master out of disk space.

Fix:

```
 cd /var/lib/jenkins/workspace
rm -rf *
systemctl restart jenkins
```

-

2. Wrong Nexus Private IP

- Artifact upload failed due to misconfigured IP.

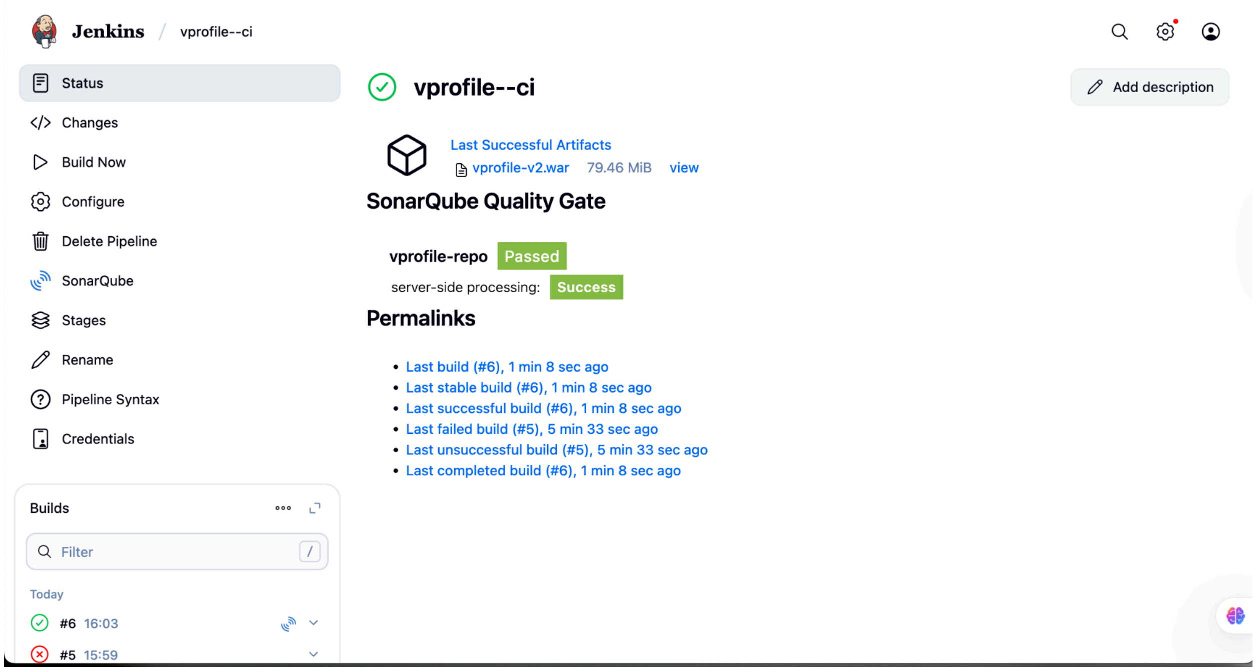
- Fix: Updated Nexus private IP in Jenkins pipeline.

3. SonarQube Quality Gates

- Build failed initially due to failing quality gates.
- Fix: Updated rules + fixed vulnerabilities in code.

📊 Results

- **Pipeline executed successfully** with all stages green.
- Multiple artifacts uploaded in Nexus with unique versions.
- SonarQube analysis ensured high-quality code.
- Slack delivered real-time build notifications.



vprofile--ci

Last Successful Artifacts
vprofile-v2.war 79.46 MiB view

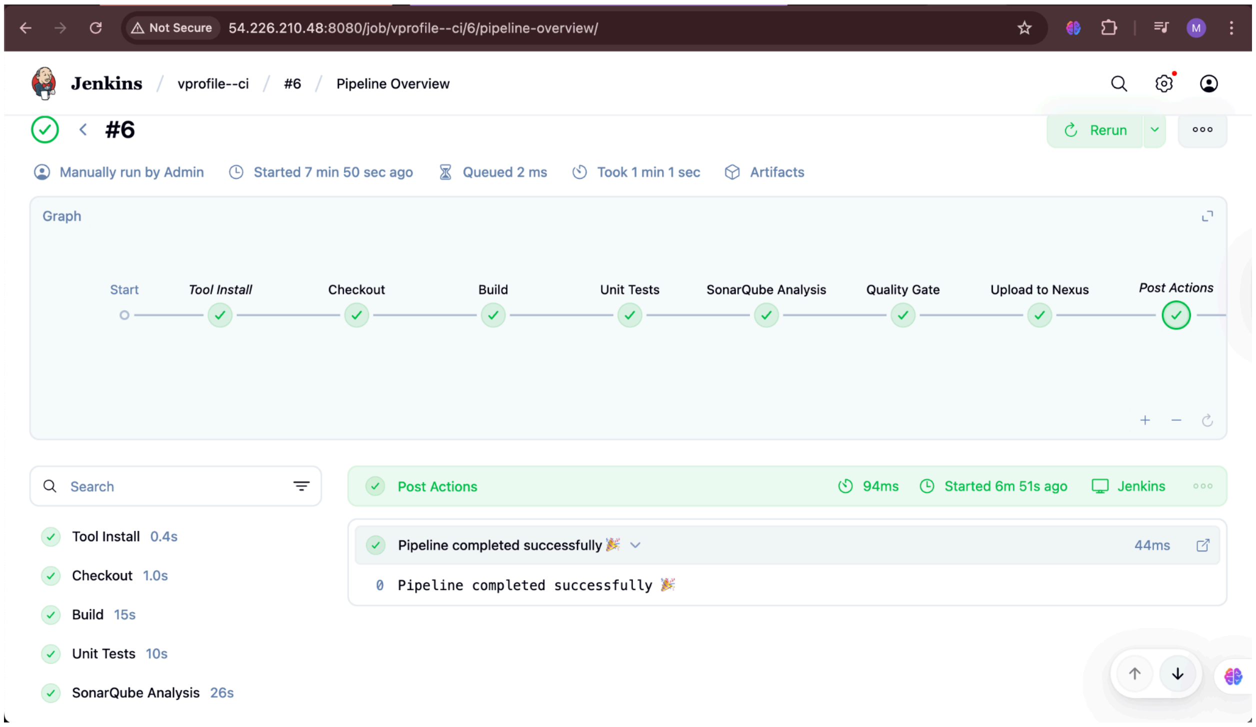
SonarQube Quality Gate

vprofile-repo Passed
server-side processing: Success

Permalinks

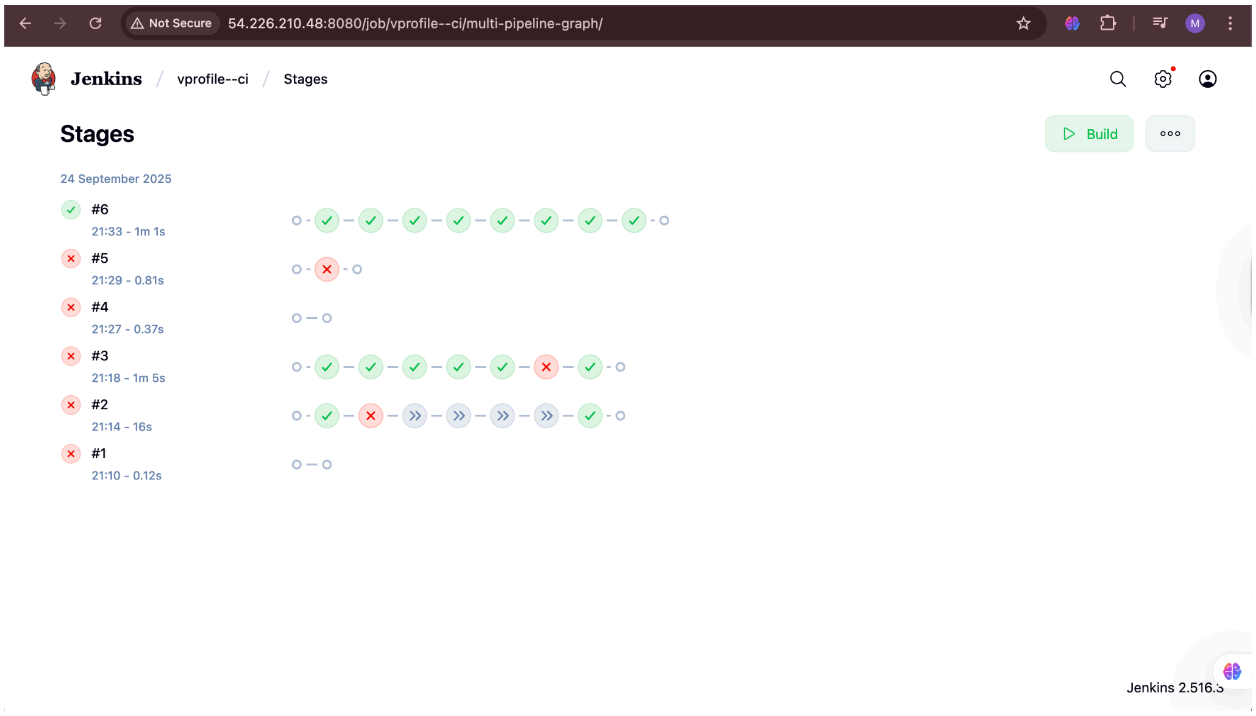
- Last build (#6), 1 min 8 sec ago
- Last stable build (#6), 1 min 8 sec ago
- Last successful build (#6), 1 min 8 sec ago
- Last failed build (#5), 5 min 33 sec ago
- Last unsuccessful build (#5), 5 min 33 sec ago
- Last completed build (#6), 1 min 8 sec ago

`jenkins-build-successfull.png`



```
0 Pipeline completed successfully 🎉
```

jenkins-pipeline-overview.png



jenkins-stages-passed.png

✅ Key Takeaways

- Implemented a **real-world CI/CD pipeline** with industry tools.
- Automated artifact versioning and uploads to Nexus.
- Integrated **SonarQube for quality checks** and **Slack for notifications**.
- Solved real-world issues (disk cleanup, IP misconfig, quality gates).

🔗 GitHub Repository

👉 mohammedinzi/vprofile-devops

🔥 This documentation:

1. Shows **end-to-end DevOps knowledge**.
2. Highlights **real troubleshooting**.
3. Reads like a **case study**, not just a tutorial.
