Capstone Project
Book-My-Show DevOps Lifecycle

Final Project Documentation

Prepared By: Sakhamuri Veera Brahmanjaneyulu
Trainer: Rashi Rana

Date: September 2025

Q. Final Project Deliverables – Book My Show DevOps Lifecycle mention in the Question

1. Jira Board Export (task progress).

2. GitHub PR link.

3. Jenkins pipeline execution logs.

4. Jenkinsfile (mandatory).

5. SonarQube Quality Gate report.

6. (Optional) Trivy/OWASP reports.

7. DockerHub repository link.

8. Dockerfile (mandatory).

9. EKS deployment proof (kubectl outputs).

10. Kubernetes manifest files – deployment.yaml and service.yaml (mandatory).

11. Prometheus & Grafana screenshots.

12. Email notification screenshot.

13. Final summary report (Word/PPT).

Introduction

The Book-My-Show (BMS) DevOps Capstone Project is designed to provide hands-on experience with the entire DevOps lifecycle. The project covers planning, code collaboration, continuous integration, containerization, orchestration, and observability. The following tools and technologies were used:

- Jira – Task tracking and workflow management
- GitHub – Source code repository and collaboration
- Jenkins – CI/CD pipeline automation
- SonarQube – Static code analysis and quality gates
- Docker – Containerization
- Kubernetes (EKS/Minikube) – Container orchestration
- Prometheus – Monitoring and metrics collection
- Grafana – Visualization and dashboards

Project Deliverables – Table of Contents

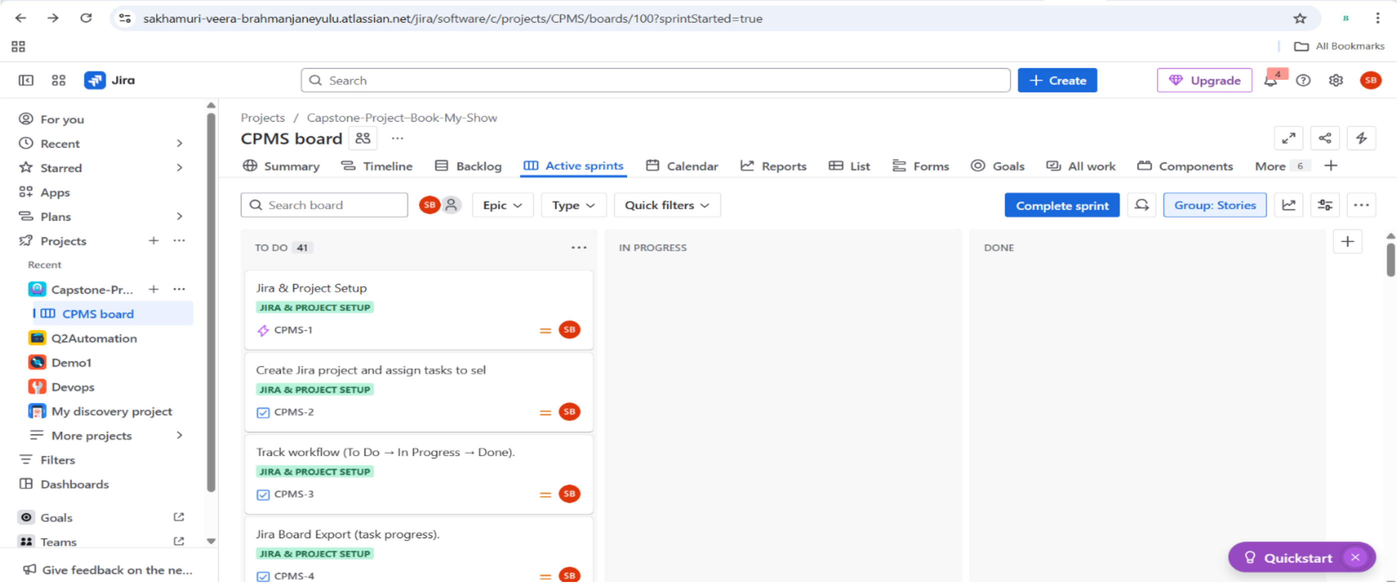
1. **Jira Workflow –** Task assignments and board export.
2. **GitHub Workflow –** Repository link, feature branch, and PRs**.**
3. **Jenkins CI/CD Pipeline –** Jenkinsfile, pipeline logs, SonarQube Quality Gate, OWASP Dependency Check, Trivy scan, and build/deployment proof.
4. **Docker Deployment –** Dockerfile, DockerHub repo, and container accessibility.
5. **Kubernetes Deployment (EKS) –** Deployment/service manifests, kubectl validation, and LoadBalancer access.
6. **Monitoring & Observability** – Prometheus metrics and Grafana dashboards**.**
7. **Notifications –** Email notification proof.

Step 1: Jira Workflow

For this project, I created a new **Jira board** to track the complete DevOps lifecycle of the Book-My-Show application. I added **Epics and Tasks** according to the project flow (Jira → GitHub → Jenkins → Docker → Kubernetes → Monitoring). Each task was assigned to myself and tracked through the workflow stages **To Do → In Progress → Done**. The board helped in monitoring progress and ensuring timely completion. Finally, the **Jira board export and dashboard screenshots** were taken as deliverables.

Epic 1: Jira & Project Setup – Create Jira project, assign tasks, track workflow, export board.
Epic 2: GitHub Workflow – Clone repo, create feature branch, make changes, push, raise PR, review & merge.
Epic 3: Jenkins CI/CD Pipeline – Install Jenkins/plugins, configure credentials, create pipeline, SonarQube analysis, quality gate, install dependencies, Trivy scan, OWASP check, Docker build & push, deploy container, email notification.
Epic 4: Docker Deployment – Write Dockerfile, build image, push to DockerHub, run container locally, validate accessibility.
Epic 5: Kubernetes Deployment (EKS) – Write deployment.yaml/service.yaml, deploy on EKS, expose service, validate with kubectl.
Epic 6: Monitoring & Observability – Install Prometheus/Node Exporter, integrate Jenkins metrics, configure Grafana, create dashboards.
Epic 7: Final Deliverables & Reporting – Submit Jira export, GitHub PR link, Jenkins logs, Jenkinsfile, SonarQube/Trivy/OWASP reports, DockerHub repo, Dockerfile, Kubernetes manifests, EKS proof, Prometheus & Grafana screenshots, email notification, final summary report.

Screenshots of Jira Board Export (task progress) :



sakhamuri-veera-brahmanjaneyulu.atlassian.net/jira/software/c/projects/CPMS/boards/100?sprintStarted=true

Projects / Capstone-Project-Book-My-Show

CPMS board

TO DO 41

Jira & Project Setup
JIRA & PROJECT SETUP
CPMS-1

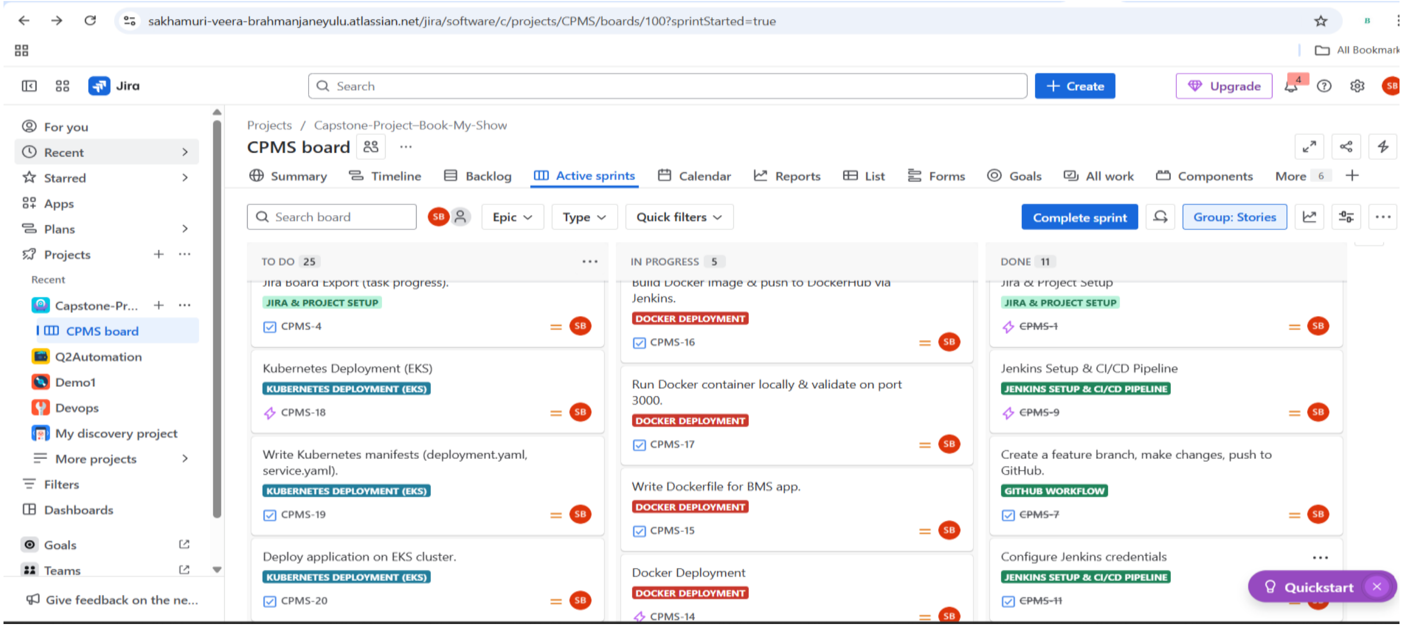
Create Jira project and assign tasks to sel
JIRA & PROJECT SETUP
CPMS-2

Track workflow (To Do → In Progress → Done).
JIRA & PROJECT SETUP
CPMS-3

Jira Board Export (task progress).
JIRA & PROJECT SETUP
CPMS-4

IN PROGRESS

DONE



TO DO 25

Jira Board Export (task progress).
JIRA & PROJECT SETUP
CPMS-4

Kubernetes Deployment (EKS)
KUBERNETES DEPLOYMENT (EKS)
CPMS-18

Write Kubernetes manifests (deployment.yaml, service.yaml).
KUBERNETES DEPLOYMENT (EKS)
CPMS-19

Deploy application on EKS cluster.
KUBERNETES DEPLOYMENT (EKS)
CPMS-20

IN PROGRESS 5

Build Docker image & push to DockerHub via Jenkins.
DOCKER DEPLOYMENT
CPMS-16

Run Docker container locally & validate on port 3000.
DOCKER DEPLOYMENT
CPMS-17

Write Dockerfile for BMS app.
DOCKER DEPLOYMENT
CPMS-15

Docker Deployment
DOCKER DEPLOYMENT
CPMS-14

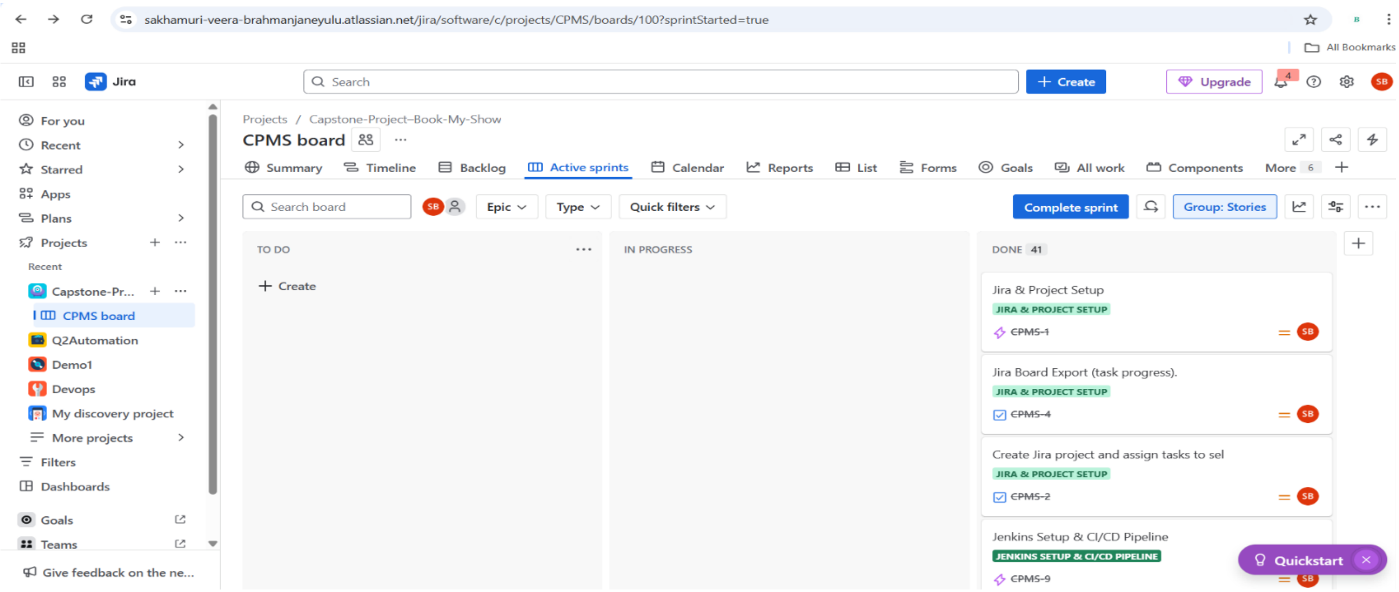
DONE 11

Jira & Project Setup
JIRA & PROJECT SETUP
~~CPMS-1~~

Jenkins Setup & CI/CD Pipeline
JENKINS SETUP & CI/CD PIPELINE
~~CPMS-9~~

Create a feature branch, make changes, push to GitHub.
GITHUB WORKFLOW
~~CPMS-7~~

Configure Jenkins credentials
JENKINS SETUP & CI/CD PIPELINE
~~CPMS-11~~



TO DO

+ Create

IN PROGRESS

DONE 41

Jira & Project Setup
JIRA & PROJECT SETUP
~~CPMS-1~~

Jira Board Export (task progress).
JIRA & PROJECT SETUP
~~CPMS-4~~

Create Jira project and assign tasks to sel
JIRA & PROJECT SETUP
~~CPMS-2~~

Jenkins Setup & CI/CD Pipeline
JENKINS SETUP & CI/CD PIPELINE
~~CPMS-9~~

Step 2: GitHub Workflow:

For source code management and collaboration, **GitHub** was utilized. A dedicated **feature branch** was created to implement changes, ensuring clean separation from the main branch. Code changes were committed and pushed to the feature branch, followed by the creation of **Pull Requests (PRs)**. These PRs underwent review before merging into the main branch to maintain code quality and consistency.

Git Commands Executed:

git clone https://github.com/akshu20791/Book-My-Show.git

cd "Capstone Project-Git"

git checkout feature

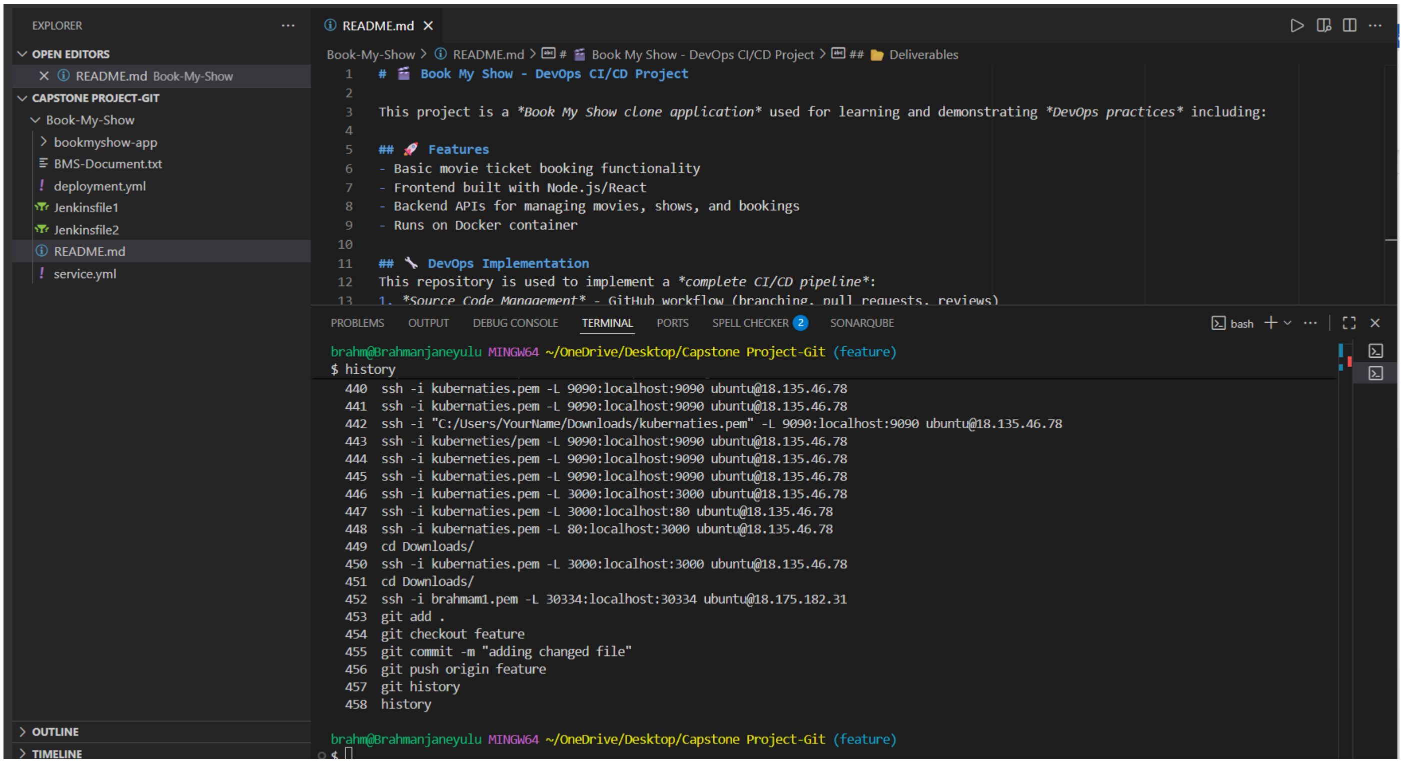
git add .

git commit -m "adding changed file"

git push origin feature

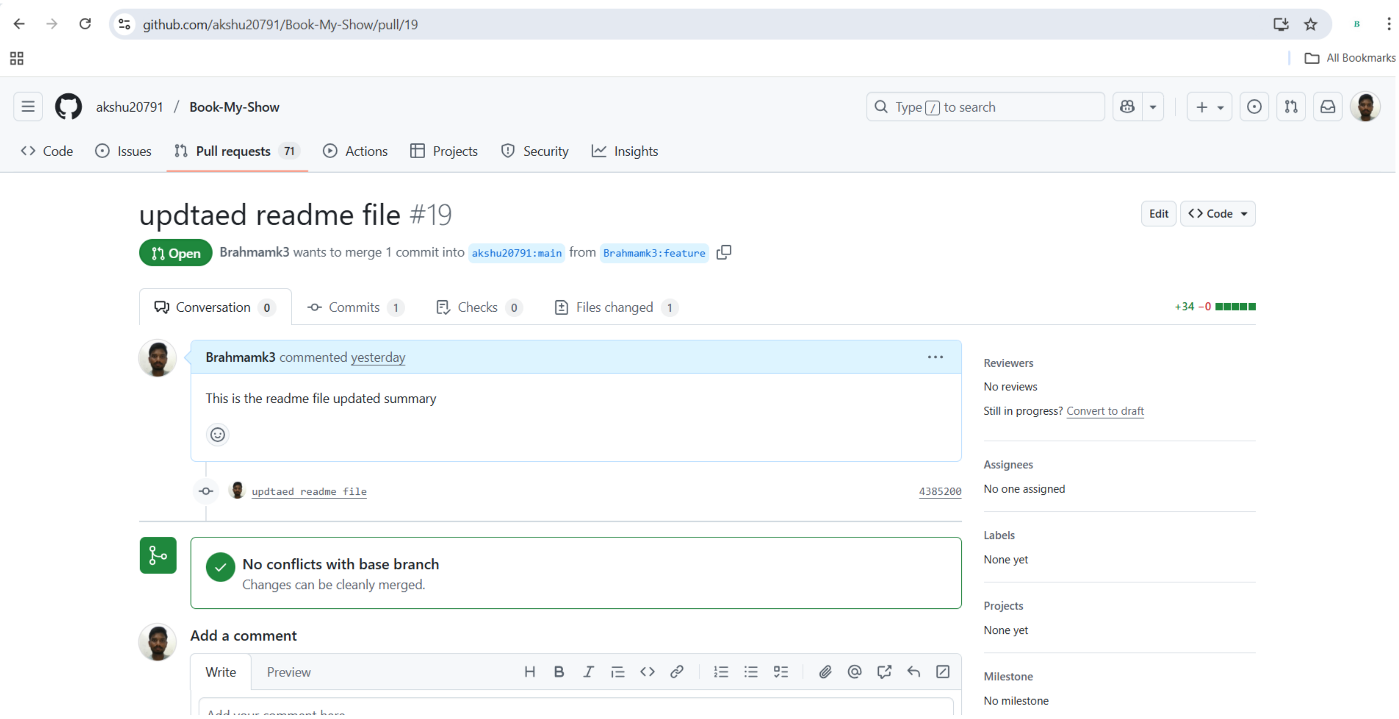
Feature Branch to Main (Parent Repository):

PR Link : https://github.com/akshu20791/Book-My-Show/pull/19



```
# 🎬 Book My Show - DevOps CI/CD Project

This project is a *Book My Show clone application* used for learning and demonstrating *DevOps practices* including:

## 🚀 Features
- Basic movie ticket booking functionality
- Frontend built with Node.js/React
- Backend APIs for managing movies, shows, and bookings
- Runs on Docker container

## 🔧 DevOps Implementation
This repository is used to implement a *complete CI/CD pipeline*:
1. *Source Code Management* - GitHub workflow (branching, pull requests, reviews)
```

```
brahm@Brahmanjaneyulu MINGW64 ~/OneDrive/Desktop/Capstone Project-Git (feature)
$ history
  440  ssh -i kubernaties.pem -L 9090:localhost:9090 ubuntu@18.135.46.78
  441  ssh -i kubernaties.pem -L 9090:localhost:9090 ubuntu@18.135.46.78
  442  ssh -i "C:/Users/YourName/Downloads/kubernaties.pem" -L 9090:localhost:9090 ubuntu@18.135.46.78
  443  ssh -i kuberneties/pem -L 9090:localhost:9090 ubuntu@18.135.46.78
  444  ssh -i kuberneties.pem -L 9090:localhost:9090 ubuntu@18.135.46.78
  445  ssh -i kubernaties.pem -L 9090:localhost:9090 ubuntu@18.135.46.78
  446  ssh -i kubernaties.pem -L 3000:localhost:3000 ubuntu@18.135.46.78
  447  ssh -i kubernaties.pem -L 3000:localhost:80 ubuntu@18.135.46.78
  448  ssh -i kubernaties.pem -L 80:localhost:3000 ubuntu@18.135.46.78
  449  cd Downloads/
  450  ssh -i kubernaties.pem -L 3000:localhost:3000 ubuntu@18.135.46.78
  451  cd Downloads/
  452  ssh -i brahmam1.pem -L 30334:localhost:30334 ubuntu@18.175.182.31
  453  git add .
  454  git checkout feature
  455  git commit -m "adding changed file"
  456  git push origin feature
  457  git history
  458  history

brahm@Brahmanjaneyulu MINGW64 ~/OneDrive/Desktop/Capstone Project-Git (feature)
$
```



github.com/akshu20791/Book-My-Show

akshu20791 / Book-My-Show

Code | Issues | Pull requests 71 | Actions | Projects | Security | Insights

updtaed readme file #19

Open — Brahmamk3 wants to merge 1 commit into `akshu20791:main` from `Brahmamk3:feature`

Conversation 0 | Commits 1 | Checks 0 | Files changed 1 | +34 −0

Brahmamk3 commented yesterday

This is the readme file updated summary

updtaed readme file — 4385200

No conflicts with base branch
Changes can be cleanly merged.

Add a comment

Reviewers: No reviews
Still in progress? Convert to draft

Assignees: No one assigned

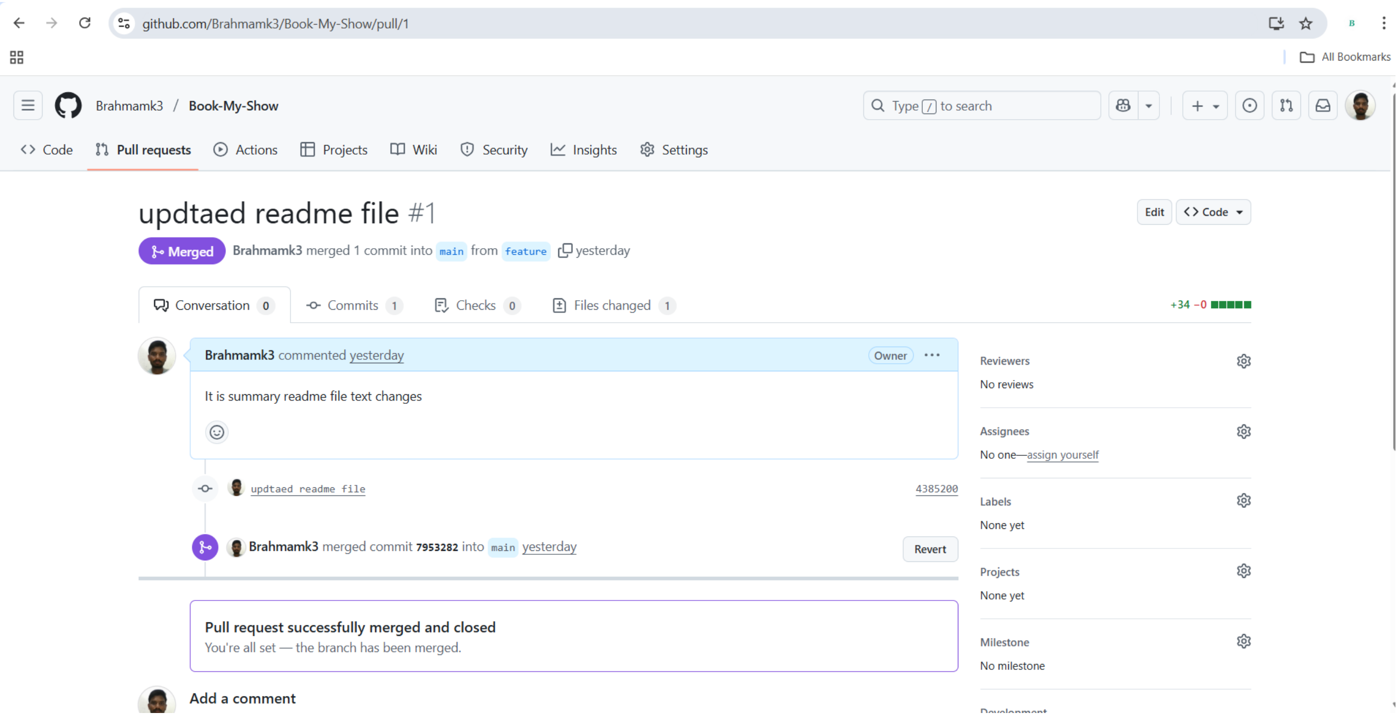
Labels: None yet

Projects: None yet

Milestone: No milestone

Feature Branch to Main (My Repository):

PR Link : https://github.com/Brahmamk3/Book-My-Show/pull/1



github.com/Brahmamk3/Book-My-Show/pull/1

Brahmamk3 / Book-My-Show

Code | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

updtaed readme file #1

Merged — Brahmamk3 merged 1 commit into `main` from `feature` yesterday

Conversation 0 | Commits 1 | Checks 0 | Files changed 1 | +34 −0

Brahmamk3 commented yesterday (Owner)

It is summary readme file text changes

updtaed readme file — 4385200

Brahmamk3 merged commit 7953282 into `main` yesterday — Revert

Pull request successfully merged and closed
You're all set — the branch has been merged.

Add a comment

Reviewers: No reviews

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

Step 3: Jenkins CI/CD Pipeline :

=============================

A Jenkins Declarative Pipeline was implemented to automate the CI/CD lifecycle for the Book-My-Show (BMS) web application. The pipeline integrates GitHub, SonarQube, OWASP Dependency-Check, Trivy, Docker, and AWS EKS to deliver a robust DevOps workflow.

Pipeline 1: DockerHub & Local Deployment (book-my-show-without-k8s)

This pipeline handles automated build, test, and deployment in a local Docker environment.

Implemented Stages:

1. **Clean Workspace** – Prepares a fresh build environment.
2. **Checkout from Git** – Clones the latest source code from GitHub.
3. **SonarQube Analysis** – Runs static code analysis for quality checks.
4. **Quality Gate** – Ensures compliance with defined quality standards.
5. **Install Dependencies** – Installs Node.js dependencies using npm install.
6. **OWASP Dependency Check** – Scans for vulnerabilities in dependencies.
7. **Trivy FS Scan** – Detects filesystem security issues and secrets.
8. **Docker Build & Push** – Builds and pushes Docker image to DockerHub (brahmamk015/bookmyshow:latest).
9. **Deploy to Container** – Deploys application locally in a Docker container on port 3000.
10. **Post Actions (Email Notification)** – Sends build results, logs, and reports via email.

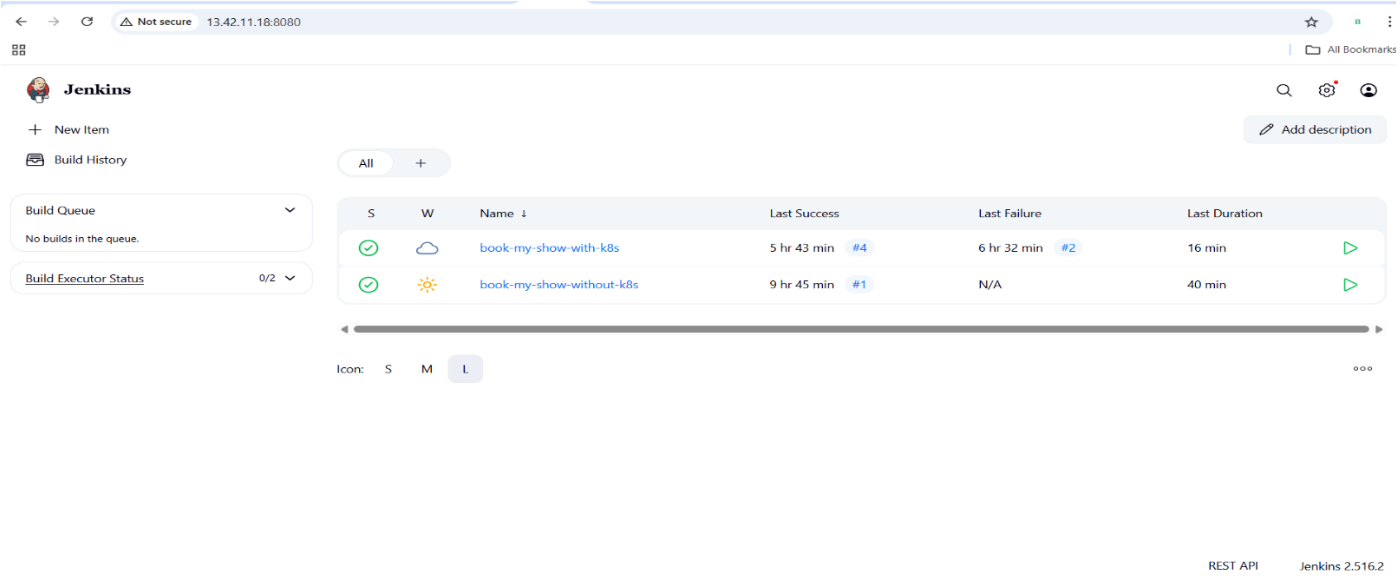
Pipeline 2: AWS EKS Deployment (book-my-show-with-k8s)

This pipeline extends the functionality of Pipeline 1 by deploying the application to an **AWS EKS cluster**.

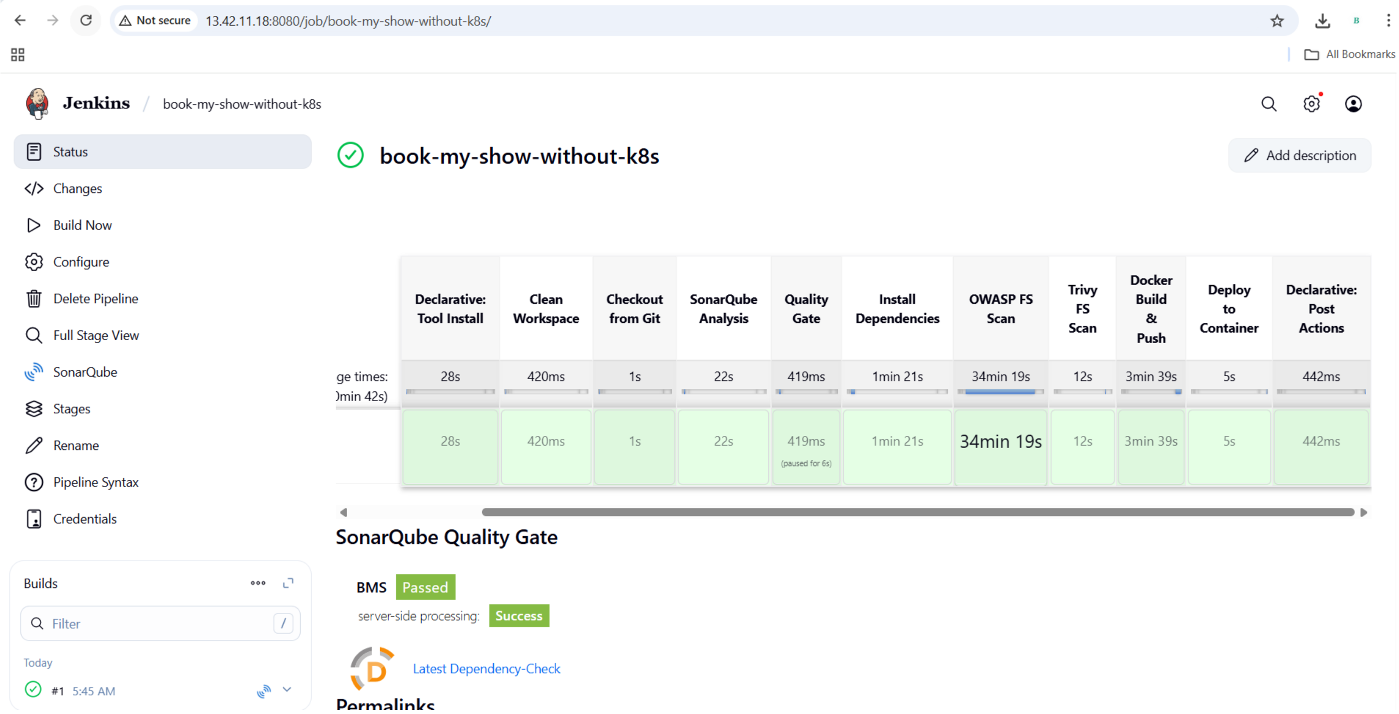
Implemented Stages:

1. All stages from Pipeline 1 (Git Checkout → Trivy Scan → Docker Build & Push → Email Notification).
2. **Deploy to EKS Cluster** –
   - Configures kubectl using AWS CLI.
   - Applies Kubernetes manifests (deployment.yml & service.yml).
   - Validates successful deployment by verifying pods and services inside the cluster.

Pipeline Stage Views & Screenshots were captured for both jobs to demonstrate execution flow and results :



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☁	book-my-show-with-k8s	5 hr 43 min #4	6 hr 32 min #2	16 min
✓	☀	book-my-show-without-k8s	9 hr 45 min #1	N/A	40 min



book-my-show-without-k8s

	Declarative: Tool Install	Clean Workspace	Checkout from Git	SonarQube Analysis	Quality Gate	Install Dependencies	OWASP FS Scan	Trivy FS Scan	Docker Build & Push	Deploy to Container	Declarative: Post Actions
ge times:)min 42s)	28s	420ms	1s	22s	419ms	1min 21s	34min 19s	12s	3min 39s	5s	442ms
	28s	420ms	1s	22s	419ms (paused for 6s)	1min 21s	34min 19s	12s	3min 39s	5s	442ms

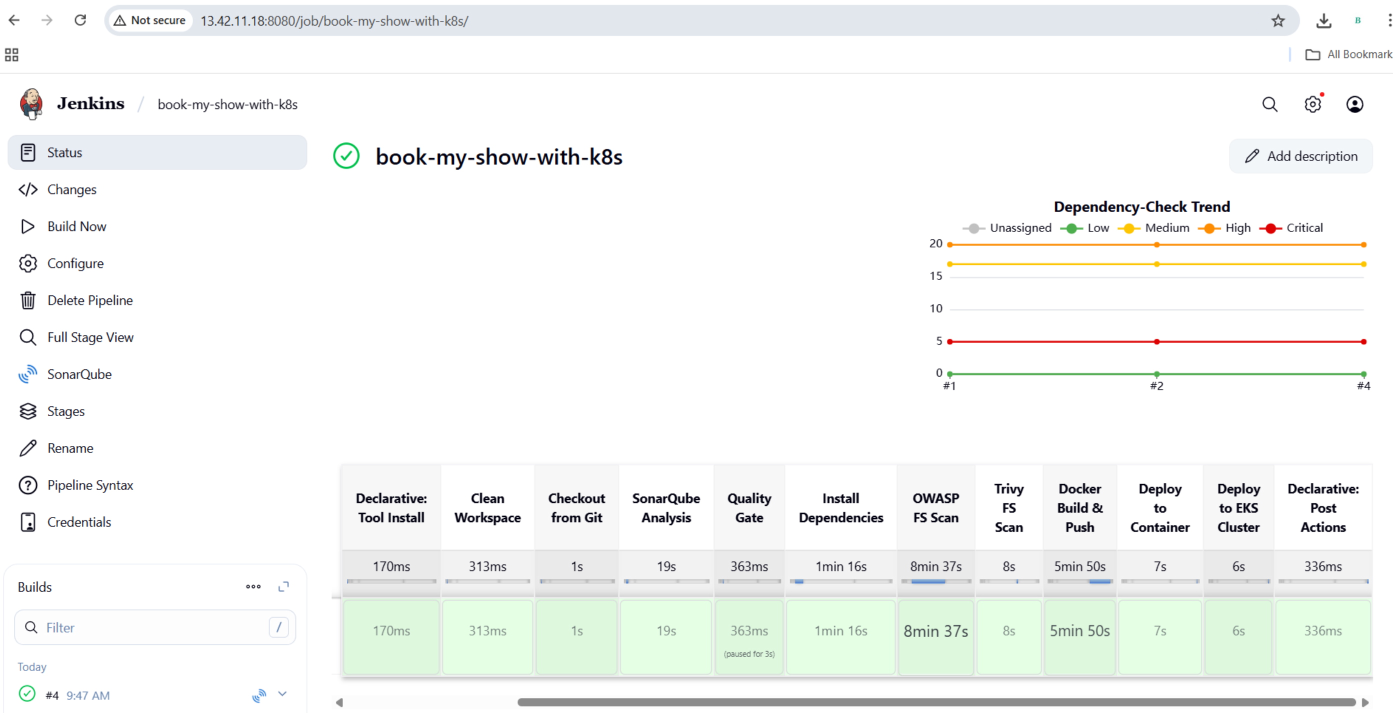
SonarQube Quality Gate

BMS Passed

server-side processing: Success

Latest Dependency-Check

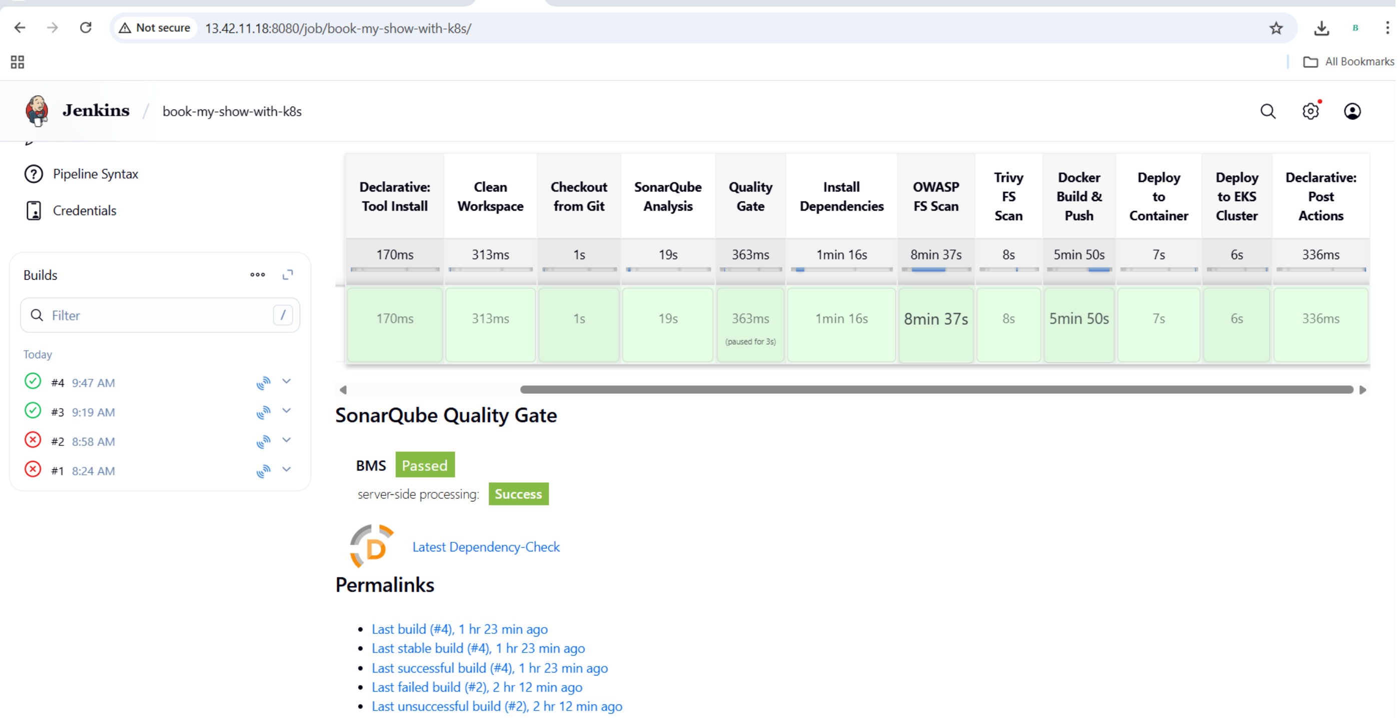
Permalinks



book-my-show-with-k8s

Declarative: Tool Install	Clean Workspace	Checkout from Git	SonarQube Analysis	Quality Gate	Install Dependencies	OWASP FS Scan	Trivy FS Scan	Docker Build & Push	Deploy to Container	Deploy to EKS Cluster	Declarative: Post Actions
170ms	313ms	1s	19s	363ms	1min 16s	8min 37s	8s	5min 50s	7s	6s	336ms
170ms	313ms	1s	19s	363ms (paused for 3s)	1min 16s	8min 37s	8s	5min 50s	7s	6s	336ms

#4 9:47 AM



Declarative: Tool Install	Clean Workspace	Checkout from Git	SonarQube Analysis	Quality Gate	Install Dependencies	OWASP FS Scan	Trivy FS Scan	Docker Build & Push	Deploy to Container	Deploy to EKS Cluster	Declarative: Post Actions
170ms	313ms	1s	19s	363ms	1min 16s	8min 37s	8s	5min 50s	7s	6s	336ms
170ms	313ms	1s	19s	363ms (paused for 3s)	1min 16s	8min 37s	8s	5min 50s	7s	6s	336ms

#4 9:47 AM
#3 9:19 AM
#2 8:58 AM
#1 8:24 AM

SonarQube Quality Gate

BMS Passed

server-side processing: Success

Latest Dependency-Check

Permalinks

- Last build (#4), 1 hr 23 min ago
- Last stable build (#4), 1 hr 23 min ago
- Last successful build (#4), 1 hr 23 min ago
- Last failed build (#2), 2 hr 12 min ago
- Last unsuccessful build (#2), 2 hr 12 min ago

Pipeline Scripts:

1. **Book-my-show-without-k8s :**

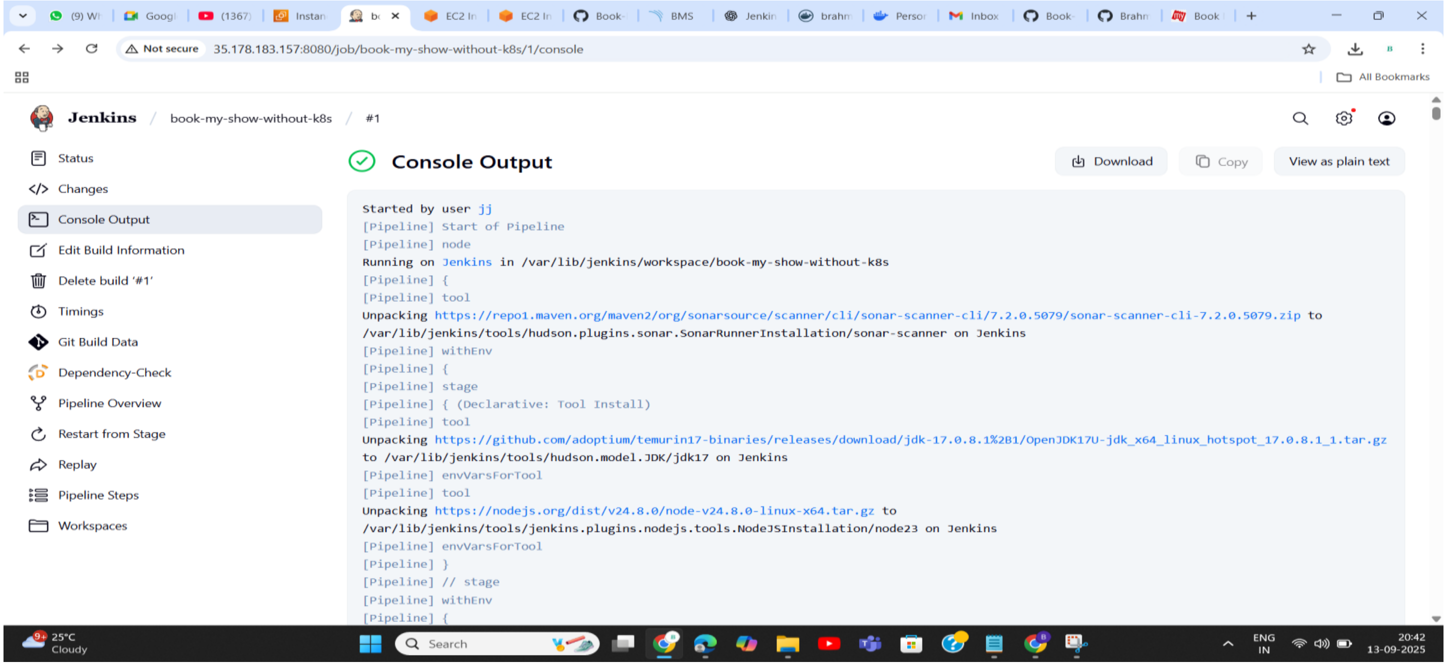
```
pipeline {
    agent any
    tools {
        jdk 'jdk17'
        nodejs 'node23'
    }
    environment {
        SCANNER_HOME = tool 'sonar-scanner'
    }
    stages {
        stage('Clean Workspace') {
            steps {
                cleanWs()
            }
        }
        stage('Checkout from Git') {
            steps {
                git branch: 'main', url: 'https://github.com/Brahmamk3/Book-My-Show.git'
                sh 'ls -la'  // Verify files after checkout
            }
        }
        stage('SonarQube Analysis') {
            steps {
                withSonarQubeEnv('sonar-server') {
                    sh '''
                    $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=BMS \
                    -Dsonar.projectKey=BMS
                    '''
                }
            }
        }
        stage('Quality Gate') {
            steps {
                script {
                    waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
                }
            }
        }
        stage('Install Dependencies') {
            steps {
                sh '''
                cd bookmyshow-app
                ls -la  # Verify package.json exists
                if [ -f package.json ]; then
                    rm -rf node_modules package-lock.json  # Remove old dependencies
                    npm install  # Install fresh dependencies
                else
                    echo "Error: package.json not found in bookmyshow-app!"
                    exit 1
                fi
                '''
            }
        }
        stage('OWASP FS Scan') {
            steps {
                dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'
                dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
            }
        }
        stage('Trivy FS Scan') {
            steps {
                sh 'trivy fs . > trivyfs.txt'
            }
        }
        stage('Docker Build & Push') {
            steps {
                script {
                    withDockerRegistry(credentialsId: 'docker', toolName: 'docker') {
                        sh '''
                        echo "Building Docker image..."
                        docker build --no-cache -t brahmamk015/bookmyshow:latest -f bookmyshow-app/Dockerfile bookmyshow-app

                        echo "Pushing Docker image to registry..."
                        docker push brahmamk015/bookmyshow:latest
                        '''
                    }
                }
            }
        }
        stage('Deploy to Container') {
            steps {
                sh '''
                echo "Stopping and removing old container..."
                docker stop bms || true
                docker rm bms || true

                echo "Running new container on port 3000..."
                docker run -d --restart=always --name bms -p 3000:3000 brahmamk015/bookmyshow:latest

                echo "Checking running containers..."
                docker ps -a

```

```
              echo "Fetching logs..."
              sleep 5  # Give time for the app to start
              docker logs bms
              '''
          }
      }
        }
        post {
            always {
                emailext(
                    attachLog: true,
                    subject: "'${currentBuild.result}'",
                    body: """
                       Capstone Project Case Study – Book-My-Show DevOps Lifecycle
                        Project: ${env.JOB_NAME}<br/>
                        Build Number: ${env.BUILD_NUMBER}<br/>
                        URL: <a href="${env.BUILD_URL}">${env.BUILD_URL}</a><br/>
                    """,
                    to: 'sakamuriveera@gmail.com',
                    attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
                )
            }
        }
    }
```

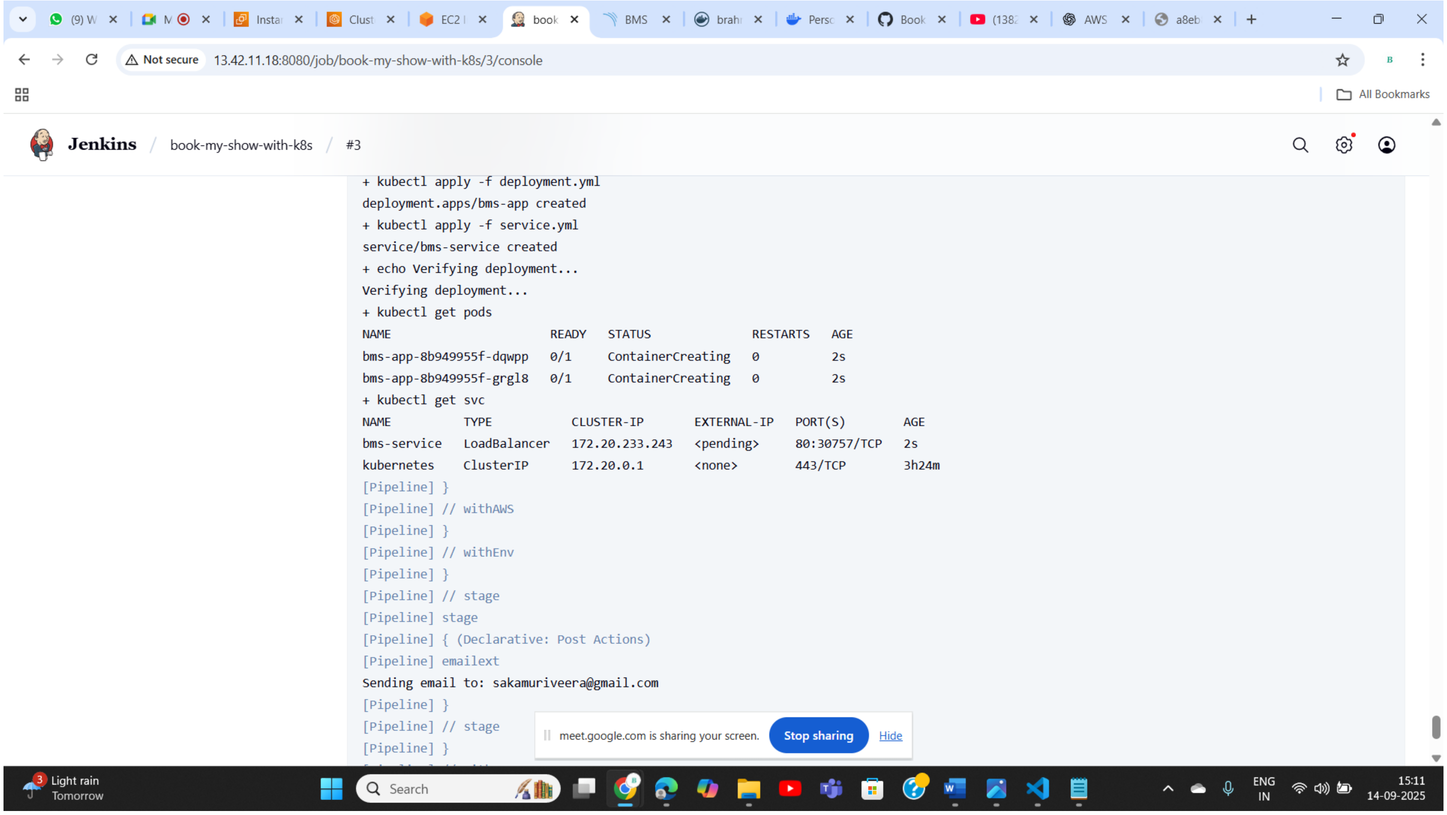
2. Pipeline Script for book-my-show-with-k8s :

```
pipeline {
    agent any

    tools {
        jdk 'jdk17'
        nodejs 'node23'
    }

    environment {
        SCANNER_HOME = tool 'sonar-scanner'
        DOCKER_IMAGE = 'brahmamk015/bookmyshow:latest'
        EKS_CLUSTER_NAME = 'team4-eks-cluster'
        AWS_REGION = 'eu-west-2'
    }

    stages {
        stage('Clean Workspace') {
            steps {
                cleanWs()
            }
        }

        stage('Checkout from Git') {
            steps {
                git branch: 'main', url: 'https://github.com/Brahmamk3/Book-My-Show.git'
                sh 'ls -la'  // Verify files after checkout
            }
        }

        stage('SonarQube Analysis') {
            steps {
                withSonarQubeEnv('sonar-server') {
                    sh '''
                    $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=BMS \
                    -Dsonar.projectKey=BMS
                    '''
                }
            }
        }

        stage('Quality Gate') {
            steps {
                script {
                    waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
                }
            }
        }

        stage('Install Dependencies') {
            steps {
                sh '''
                cd bookmyshow-app
                ls -la  # Verify package.json exists
                if [ -f package.json ]; then
                    rm -rf node_modules package-lock.json
                    npm install
                else
                    echo "Error: package.json not found in bookmyshow-app!"
                    exit 1
                fi
                '''
            }
        }
        stage('OWASP FS Scan') {
            steps {
                dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'
                dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
            }
        }

        stage('Trivy FS Scan') {
            steps {
                sh 'trivy fs . > trivyfs.txt'
            }
        }

        stage('Docker Build & Push') {
            steps {
                script {
                    withDockerRegistry(credentialsId: 'docker', toolName: 'docker') {
                        sh '''
                        echo "Building Docker image..."
                        docker build --no-cache -t $DOCKER_IMAGE -f bookmyshow-app/Dockerfile bookmyshow-app

                        echo "Pushing Docker image to registry..."
                        docker push $DOCKER_IMAGE
                        '''
                    }
                }
            }
        }

        stage('Deploy to Container') {
            steps {
                sh '''
            echo "Stopping and removing old container..."
            docker stop bms || true
                docker rm bms || true

                echo "Running new container on port 3000..."
                docker run -d --restart=always --name bms -p 3000:3000 $DOCKER_IMAGE

                echo "Checking running containers..."
                docker ps -a

                echo "Fetching logs..."
                sleep 5
                docker logs bms
                '''
            }
        }

        stage('Deploy to EKS Cluster') {
            steps {
                withAWS(credentials: 'aws', region: "${AWS_REGION}") {
                    sh '''
                    echo "Verifying AWS credentials..."
                    aws sts get-caller-identity

                    echo "Configuring kubectl for EKS cluster..."
                    aws eks update-kubeconfig --name $EKS_CLUSTER_NAME --region $AWS_REGION

                    echo "Verifying kubeconfig..."
                    kubectl config view

                    echo "Deploying application to EKS..."
                    kubectl apply -f deployment.yml
                    kubectl apply -f service.yml

                    echo "Verifying deployment..."
                    kubectl get pods
                    kubectl get svc
                    '''
                }
            }
        }
    }

    post {
        always {
            emailext(
                attachLog: true,
                subject: "'${currentBuild.result}'",
                body: """
                   Capstone Project Case Study – Book-My-Show DevOps Lifecycle
                    Project: ${env.JOB_NAME}<br/>
                    Build Number: ${env.BUILD_NUMBER}<br/>
                    URL: <a href="${env.BUILD_URL}">${env.BUILD_URL}</a><br/>
                """,
                to: 'sakamuriveera@gmail.com',
                attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
            )
        }
    }
}
```

Jenkins pipeline execution logs Screenshots and github links where this pipeline execution logs are stored:



```
Started by user jj
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/book-my-show-without-k8s
[Pipeline] {
[Pipeline] tool
Unpacking https://repo1.maven.org/maven2/org/sonarsource/scanner/cli/sonar-scanner-cli/7.2.0.5079/sonar-scanner-cli-7.2.0.5079.zip to
/var/lib/jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation/sonar-scanner on Jenkins
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Tool Install)
[Pipeline] tool
Unpacking https://github.com/adoptium/temurin17-binaries/releases/download/jdk-17.0.8.1%2B1/OpenJDK17U-jdk_x64_linux_hotspot_17.0.8.1_1.tar.gz
to /var/lib/jenkins/tools/hudson.model.JDK/jdk17 on Jenkins
[Pipeline] envVarsForTool
[Pipeline] tool
Unpacking https://nodejs.org/dist/v24.8.0/node-v24.8.0-linux-x64.tar.gz to
/var/lib/jenkins/tools/jenkins.plugins.nodejs.tools.NodeJSInstallation/node23 on Jenkins
[Pipeline] envVarsForTool
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
```



```
+ kubectl apply -f deployment.yml
deployment.apps/bms-app created
+ kubectl apply -f service.yml
service/bms-service created
+ echo Verifying deployment...
Verifying deployment...
+ kubectl get pods
NAME                       READY   STATUS              RESTARTS   AGE
bms-app-8b949955f-dqwpp    0/1     ContainerCreating   0          2s
bms-app-8b949955f-grgl8    0/1     ContainerCreating   0          2s
+ kubectl get svc
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
bms-service   LoadBalancer   172.20.233.243   <pending>     80:30757/TCP   2s
kubernetes    ClusterIP      172.20.0.1       <none>        443/TCP        3h24m
[Pipeline] }
[Pipeline] // withAWS
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] emailext
Sending email to: sakamuriveera@gmail.com
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
```

Pipeline execution logs github repository link are:

Jenkins pipeline execution log for Book-my-show-with-k8s : https://github.com/Brahmamk3/Capstone-Project-Case-Study-Book-My-Show-/blob/main/jenkins-logs-with-k8s.txt

Jenkins pipeline execution log Book-my-show-without-k8s : https://github.com/Brahmamk3/Capstone-Project-Case-Study-Book-My-Show-/blob/main/jenkins-logs-without-k8s.txt

SonarQube Quality Gate Report

As part of the CI/CD pipeline, SonarQube was integrated to enforce code quality standards and detect potential bugs, vulnerabilities, and code smells.

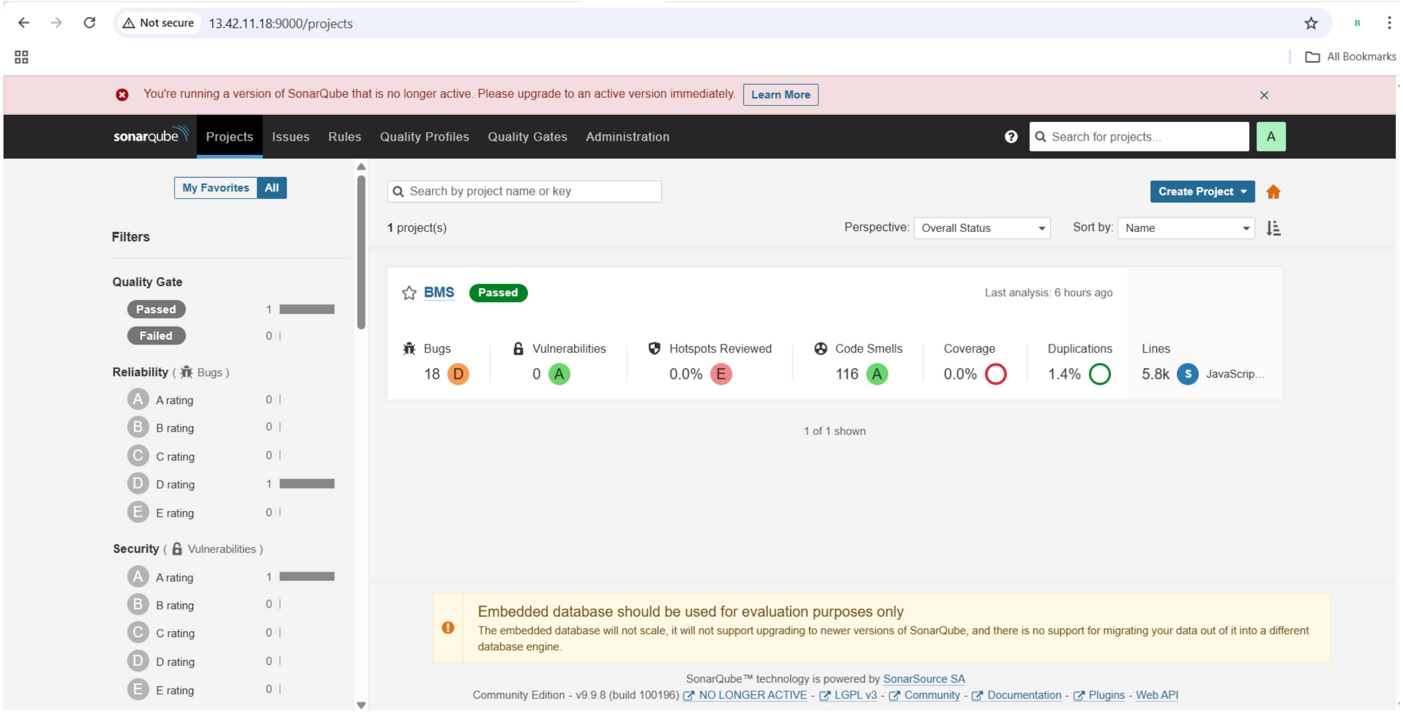
Implementation Details:

- Source code was scanned using SonarQube during the Jenkins pipeline execution.
- The Quality Gate was applied to ensure that the project met predefined thresholds for maintainability, reliability, and security.
- The report highlighted metrics such as code coverage, duplications, bugs, and vulnerabilities.
- Only when the code successfully passed the Quality Gate did the pipeline proceed to the subsequent stages.

Validation:

The SonarQube Quality Gate report confirmed that the Book-My-Show (BMS) application met the required quality standards, ensuring a cleaner and more maintainable codebase before deployment.

Figure: SonarQube Quality Gate report verifying code quality compliance and successful static analysis of the BMS application.



You're running a version of SonarQube that is no longer active. Please upgrade to an active version immediately.

Filters

Quality Gate
Passed 1
Failed 0

Reliability (Bugs)
A rating 0
B rating 0
C rating 0
D rating 1
E rating 0

Security (Vulnerabilities)
A rating 1
B rating 0
C rating 0
D rating 0
E rating 0

1 project(s)

Perspective: Overall Status
Sort by: Name

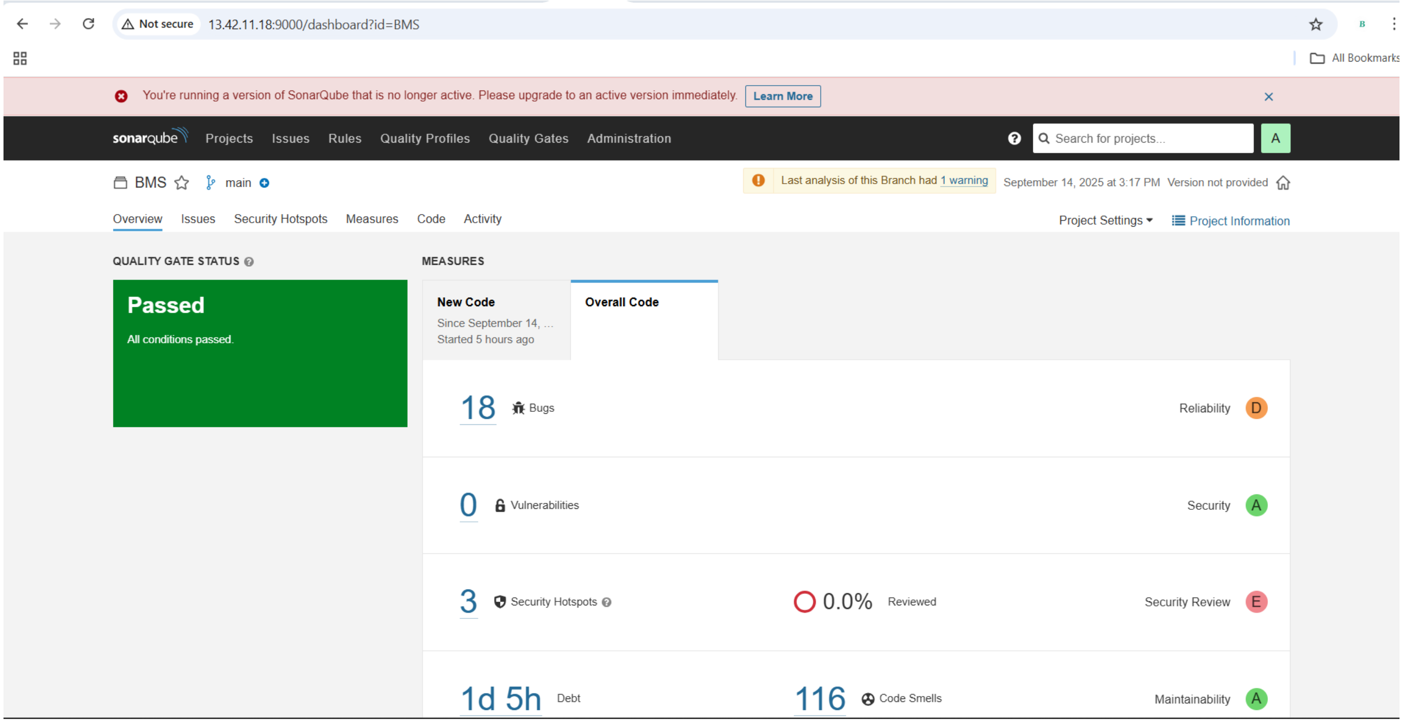
BMS Passed
Last analysis: 6 hours ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
18 D	0 A	0.0% E	116 A	0.0%	1.4%	5.8k S JavaScrip...

1 of 1 shown

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

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



Not secure 13.42.11.18:9000/dashboard?id=BMS

All Bookmarks

You're running a version of SonarQube that is no longer active. Please upgrade to an active version immediately. Learn More

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration Search for projects... A

BMS main

Last analysis of this Branch had 1 warning September 14, 2025 at 3:17 PM Version not provided

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

QUALITY GATE STATUS

Passed

All conditions passed.

MEASURES

New Code Since September 14, ... Started 5 hours ago

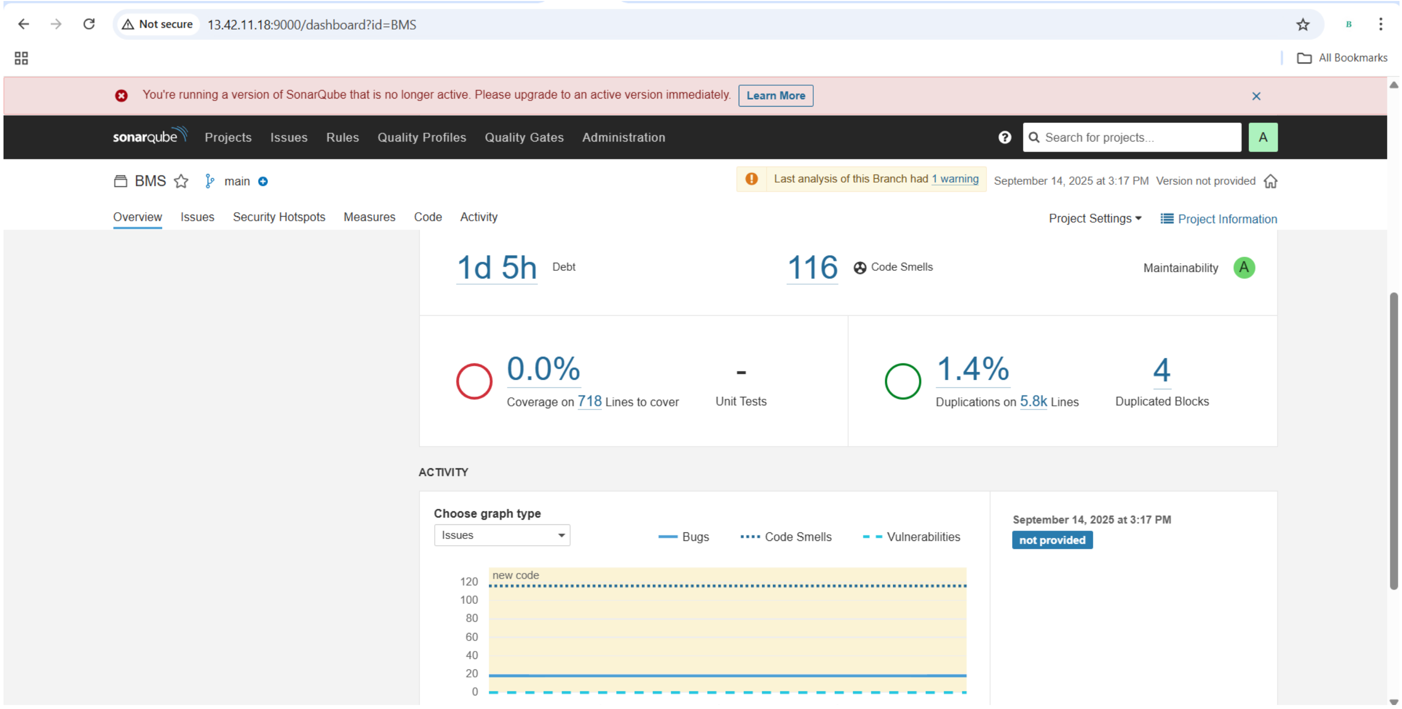
Overall Code

18 Bugs Reliability D

0 Vulnerabilities Security A

3 Security Hotspots 0.0% Reviewed Security Review E

1d 5h Debt 116 Code Smells Maintainability A



Not secure 13.42.11.18:9000/dashboard?id=BMS

All Bookmarks

You're running a version of SonarQube that is no longer active. Please upgrade to an active version immediately. Learn More

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration Search for projects... A

BMS main

Last analysis of this Branch had 1 warning September 14, 2025 at 3:17 PM Version not provided

Overview Issues Security Hotspots Measures Code Activity Project Settings Project Information

1d 5h Debt 116 Code Smells Maintainability A

0.0% Coverage on 718 Lines to cover

- Unit Tests

1.4% Duplications on 5.8k Lines

4 Duplicated Blocks

ACTIVITY

Choose graph type: Issues

September 14, 2025 at 3:17 PM

not provided

OWASP Dependency Check Report

OWASP Dependency Check was integrated into the CI/CD pipeline to identify known vulnerabilities (CVEs) in third-party libraries and dependencies used by the BMS application.

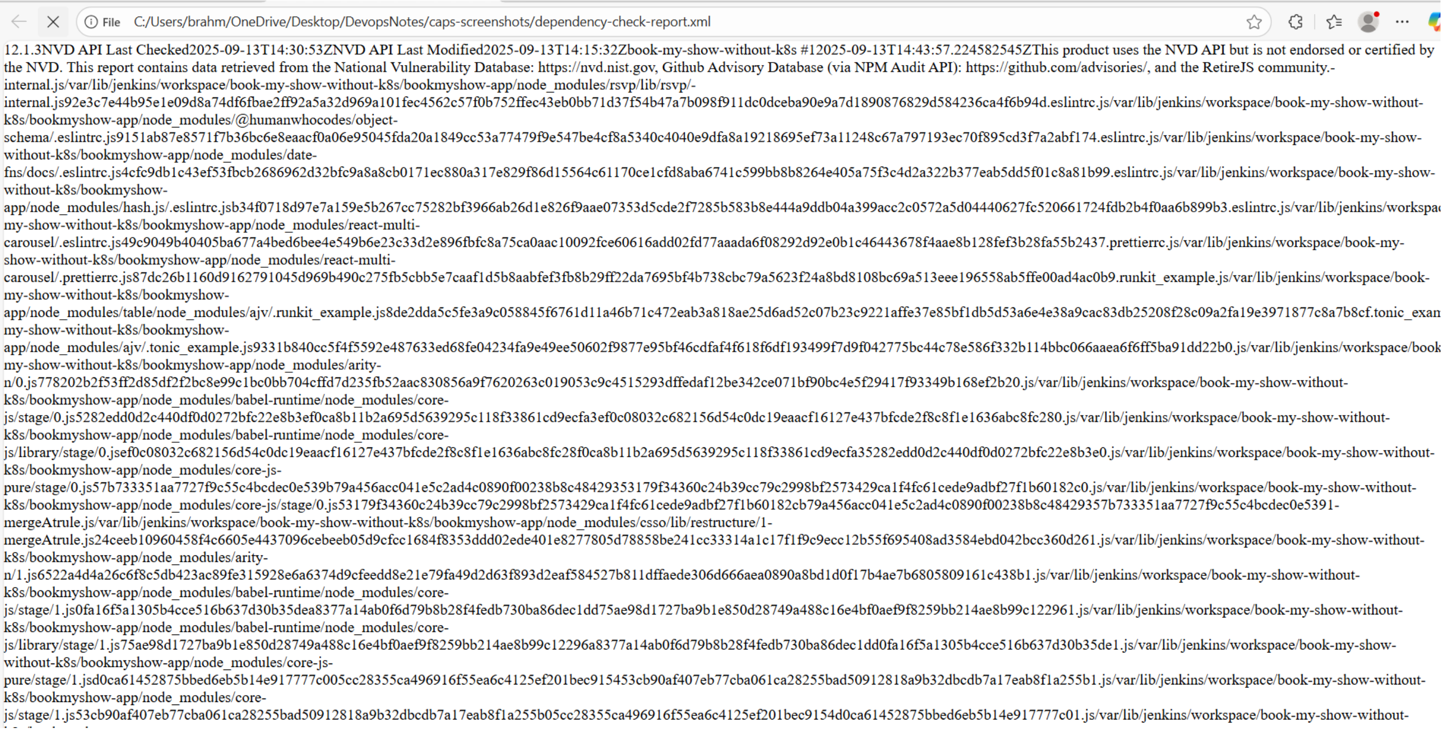
Implementation Details:

- The Jenkins pipeline executed OWASP Dependency Check as a security stage.
- It scanned all project dependencies for publicly disclosed vulnerabilities.
- A detailed HTML report was generated highlighting CVE IDs, severity levels, and recommendations.

Validation:

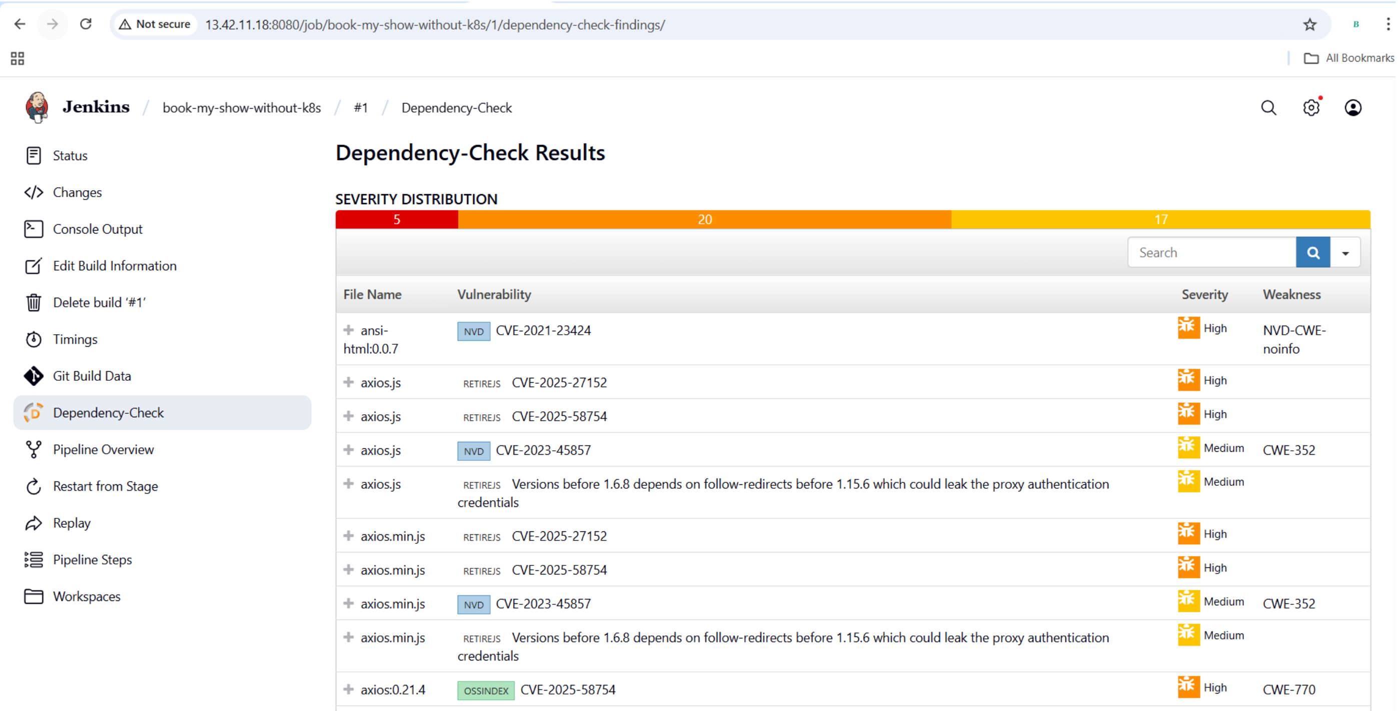
The OWASP report confirmed that dependencies were evaluated against vulnerability databases, ensuring proactive mitigation of security risks.

Figures: OWASP Dependency Check report showing dependency vulnerability scan results.



File
C:/Users/brahm/OneDrive/Desktop/DevopsNotes/caps-screenshots/dependency-check-report.xml

```
12.1.3NVD API Last Checked2025-09-13T14:30:53ZNVD API Last Modified2025-09-13T14:15:32Zbook-my-show-without-k8s #12025-09-13T14:43:57.224582545ZThis product uses the NVD API but is not endorsed or certified by
the NVD. This report contains data retrieved from the National Vulnerability Database: https://nvd.nist.gov, Github Advisory Database (via NPM Audit API): https://github.com/advisories/, and the RetireJS community.-
internal.js/var/lib/jenkins/workspace/book-my-show-without-k8s/bookmyshow-app/node_modules/rsvp/lib/rsvp/-
internal.js92e3c7e44b95e1e09d8a74df6fbae2ff92a5a32d969a101fec4562c57f0b752ffec43eb0bb71d37f54b47a7b098f911dc0dceba90e9a7d1890876829d584236ca4f6b94d.eslintrc.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/@humanwhocodes/object-
schema/.eslintrc.js9151ab87e8571f7b36bc6e8eaacf0a06e95045fda20a1849cc53a77479f9e547be4cf8a5340c4040e9dfa8a19218695ef73a11248c67a797193ec70f895cd3f7a2abf174.eslintrc.js/var/lib/jenkins/workspace/book-my-show-
without-k8s/bookmyshow-app/node_modules/date-
fns/docs/.eslintrc.js4cfc9db1c43ef53fbcb2686962d32bfc9a8a8cb0171ec880a317e829f86d15564c61170ce1cfd8aba6741c599bb8b8264e405a75f3c4d2a322b377eab5dd5f01c8a81b99.eslintrc.js/var/lib/jenkins/workspace/book-my-show-
without-k8s/bookmyshow-
app/node_modules/hash.js/.eslintrc.jsb34f0718d97e7a159e5b267cc75282bf3966ab26d1e826f9aae07353d5cde2f7285b583b8e444a9ddb04a399acc2c0572a5d04440627fc520661724fdb2b4f0aa6b899b3.eslintrc.js/var/lib/jenkins/workspa
my-show-without-k8s/bookmyshow-app/node_modules/react-multi-
carousel/.eslintrc.js49c9049b40405ba677a4bed6bee4e549b6e23c33d2e896fbfc8a75ca0aac10092fce60616add02fd77aaada6f08292d92e0b1c46443678f4aae8b128fef3b28fa55b2437.prettierrc.js/var/lib/jenkins/workspace/book-my-
show-without-k8s/bookmyshow-app/node_modules/react-multi-
carousel/.prettierrc.js87dc26b1160d9162791045d969b490c275fb5cbb5e7caaf1d5b8aabfef3fb8b29ff22da7695bf4b738cbc79a5623f24a8bd8108bc69a513eee196558ab5ffe00ad4ac0b9.runkit_example.js/var/lib/jenkins/workspace/book-
my-show-without-k8s/bookmyshow-
app/node_modules/table/node_modules/ajv/.runkit_example.js8de2dda5c5fe3a9c058845f6761d11a46b71c472eab3a818ae25d6ad52c07b23c9221affe37e85bf1db5d53a6e4e38a9cac83db25208f28c09a2fa19e3971877c8a7b8cf.tonic_exan
my-show-without-k8s/bookmyshow-
app/node_modules/ajv/.tonic_example.js9331b840cc5f4f5592e487633ed68fe04234fa9e49ee50602f9877e95bf46cdfaf4f618f6df193499f7d9f042775bc44c78e586f332b114bbc066aaea6f6ff5ba91dd22b0.js/var/lib/jenkins/workspace/book
my-show-without-k8s/bookmyshow-app/node_modules/arity-
n/0.js778202b2f53ff2d85df2f2bc8e99c1bc0bb704cffd7d235fb52aac830856a9f7620263c019053c9c4515293dffedaf12be342ce071bf90bc4e5f29417f93349b168ef2b20.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/babel-runtime/node_modules/core-
js/stage/0.js5282edd0d2c440df0d0272bfc22e8b3ef0ca8b11b2a695d5639295c118f33861cd9ecfa3ef0c08032c682156d54c0dc19eaacf16127e437bfcde2f8c8f1e1636abc8fc280.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/babel-runtime/node_modules/core-
js/library/stage/0.jsef0c08032c682156d54c0dc19eaacf16127e437bfcde2f8c8f1e1636abc8fc28f0ca8b11b2a695d5639295c118f33861cd9ecfa35282edd0d2c440df0d0272bfc22e8b3e0.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/core-js-
pure/stage/0.js57b733351aa7727f9c55c4bcdec0e539b79a456acc041e5c2ad4c0890f00238b8c48429353179f34360c24b39cc79c2998bf2573429ca1f4fc61cede9adbf27f1b60182c0.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/core-js/stage/0.js53179f34360c24b39cc79c2998bf2573429ca1f4fc61cede9adbf27f1b60182cb79a456acc041e5c2ad4c0890f00238b8c48429357b733351aa7727f9c55c4bcdec0e5391-
mergeAtrule.js/var/lib/jenkins/workspace/book-my-show-without-k8s/bookmyshow-app/node_modules/csso/lib/restructure/1-
mergeAtrule.js24ceeb10960458f4c6605e4437096cebeeb05d9cfcc1684f8353ddd02ede401e8277805d78858be241cc33314a1c17f1f9c9ecc12b55f695408ad3584ebd042bcc360d261.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/arity-
n/1.js6522a4d4a26c6f8c5db423ac89fe315928e6a6374d9cfeedd8e21e79fa49d2d63f893d2eaf584527b811dffaede306d666aea0890a8bd1d0f17b4ae7b6805809161c438b1.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/babel-runtime/node_modules/core-
js/stage/1.js0fa16f5a1305b4cce516b637d30b35dea8377a14ab0f6d79b8b28f4fedb730ba86dec1dd75ae98d1727ba9b1e850d28749a488c16e4bf0aef9f8259bb214ae8b99c122961.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/babel-runtime/node_modules/core-
js/library/stage/1.js75ae98d1727ba9b1e850d28749a488c16e4bf0aef9f8259bb214ae8b99c12296a8377a14ab0f6d79b8b28f4fedb730ba86dec1dd0fa16f5a1305b4cce516b637d30b35de1.js/var/lib/jenkins/workspace/book-my-show-
without-k8s/bookmyshow-app/node_modules/core-js-
pure/stage/1.jsd0ca61452875bbed6eb5b14e917777c005cc28355ca496916f55ea6c4125ef201bec915453cb90af407eb77cba061ca28255bad50912818a9b32dbcdb7a17eab8f1a255b1.js/var/lib/jenkins/workspace/book-my-show-without-
k8s/bookmyshow-app/node_modules/core-
js/stage/1.js53cb90af407eb77cba061ca28255bad50912818a9b32dbcdb7a17eab8f1a255b05cc28355ca496916f55ea6c4125ef201bec9154d0ca61452875bbed6eb5b14e917777c01.js/var/lib/jenkins/workspace/book-my-show-without-
```



Dependency-Check Results

SEVERITY DISTRIBUTION

5	20	17

File Name	Vulnerability	Severity	Weakness
ansi-html:0.0.7	NVD CVE-2021-23424	High	NVD-CWE-noinfo
axios.js	RETIREJS CVE-2025-27152	High	
axios.js	RETIREJS CVE-2025-58754	High	
axios.js	NVD CVE-2023-45857	Medium	CWE-352
axios.js	RETIREJS Versions before 1.6.8 depends on follow-redirects before 1.15.6 which could leak the proxy authentication credentials	Medium	
axios.min.js	RETIREJS CVE-2025-27152	High	
axios.min.js	RETIREJS CVE-2025-58754	High	
axios.min.js	NVD CVE-2023-45857	Medium	CWE-352
axios.min.js	RETIREJS Versions before 1.6.8 depends on follow-redirects before 1.15.6 which could leak the proxy authentication credentials	Medium	
axios:0.21.4	OSSINDEX CVE-2025-58754	High	CWE-770

GitHub Link for OWASP Dependency Check Report: https://github.com/Brahmamk3/Capstone-Project-Case-Study-Book-My-Show-/blob/main/dependency-check-report.7z

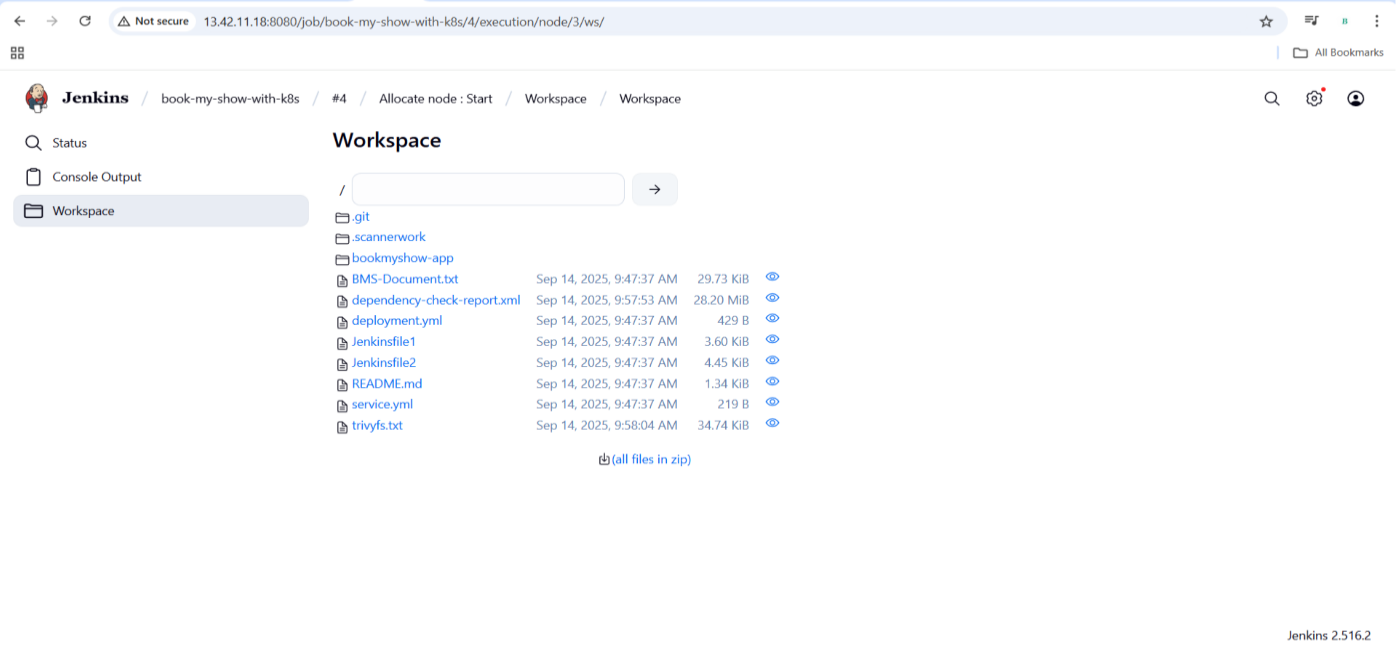
Trivy Security Scan Report :

Trivy was used to perform filesystem and Docker image security scans, ensuring that the containerized BMS application adhered to security best practices.

Implementation Details:

- Trivy scanned the Docker image and local filesystem during the Jenkins pipeline.
- It detected misconfigurations, exposed secrets, and image vulnerabilities.
- A comprehensive report was generated, highlighting any issues along with severity classifications.

Figure : workspace where **Trivy Security Scan** Reportand **OWASP Dependency Check Report**

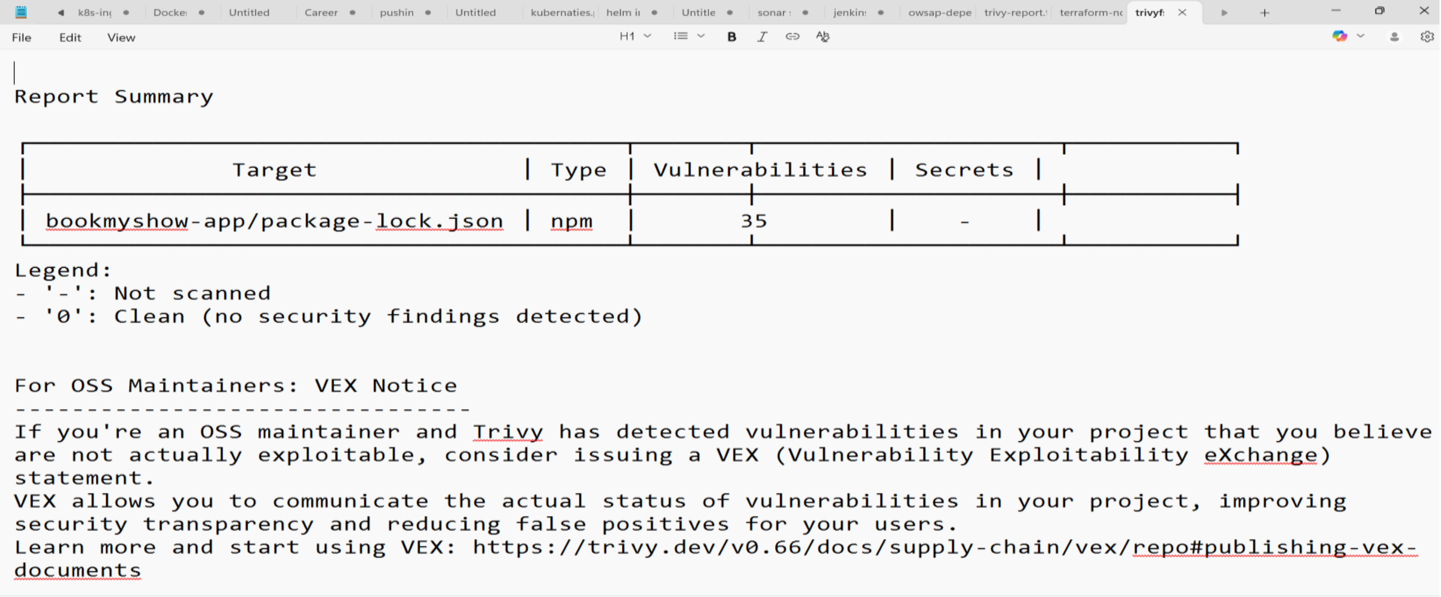

Not secure
13.42.11.18:8080/job/book-my-show-with-k8s/4/execution/node/3/ws/
All Bookmarks
Jenkins / book-my-show-with-k8s / #4 / Allocate node : Start / Workspace / Workspace
Status
Console Output
Workspace
Jenkins 2.516.2

Workspace

/

.git		
.scannerwork		
bookmyshow-app		
BMS-Document.txt	Sep 14, 2025, 9:47:37 AM	29.73 KiB
dependency-check-report.xml	Sep 14, 2025, 9:57:53 AM	28.20 MiB
deployment.yml	Sep 14, 2025, 9:47:37 AM	429 B
Jenkinsfile1	Sep 14, 2025, 9:47:37 AM	3.60 KiB
Jenkinsfile2	Sep 14, 2025, 9:47:37 AM	4.45 KiB
README.md	Sep 14, 2025, 9:47:37 AM	1.34 KiB
service.yml	Sep 14, 2025, 9:47:37 AM	219 B
trivyfs.txt	Sep 14, 2025, 9:58:04 AM	34.74 KiB

(all files in zip)



```
Report Summary

┌────────────────────────────────────┬──────┬─────────────────┬─────────┐
│               Target               │ Type │ Vulnerabilities │ Secrets │
├────────────────────────────────────┼──────┼─────────────────┼─────────┤
│ bookmyshow-app/package-lock.json   │ npm  │       35        │    -    │
└────────────────────────────────────┴──────┴─────────────────┴─────────┘
Legend:
- '-': Not scanned
- '0': Clean (no security findings detected)


For OSS Maintainers: VEX Notice
--------------------------------
If you're an OSS maintainer and Trivy has detected vulnerabilities in your project that you believe
are not actually exploitable, consider issuing a VEX (Vulnerability Exploitability eXchange)
statement.
VEX allows you to communicate the actual status of vulnerabilities in your project, improving
security transparency and reducing false positives for your users.
Learn more and start using VEX: https://trivy.dev/v0.66/docs/supply-chain/vex/repo#publishing-vex-
documents
```

Github Link for Trivy Security Scan Report: https://github.com/Brahmamk3/Capstone-Project-Case-Study-Book-My-Show-/blob/main/trivyfs.txt

Step 4: Docker Deployment :

==========================

To containerize the **Book-My-Show (BMS)** application, a **custom Dockerfile** was written. This ensured that the application could run consistently across different environments.

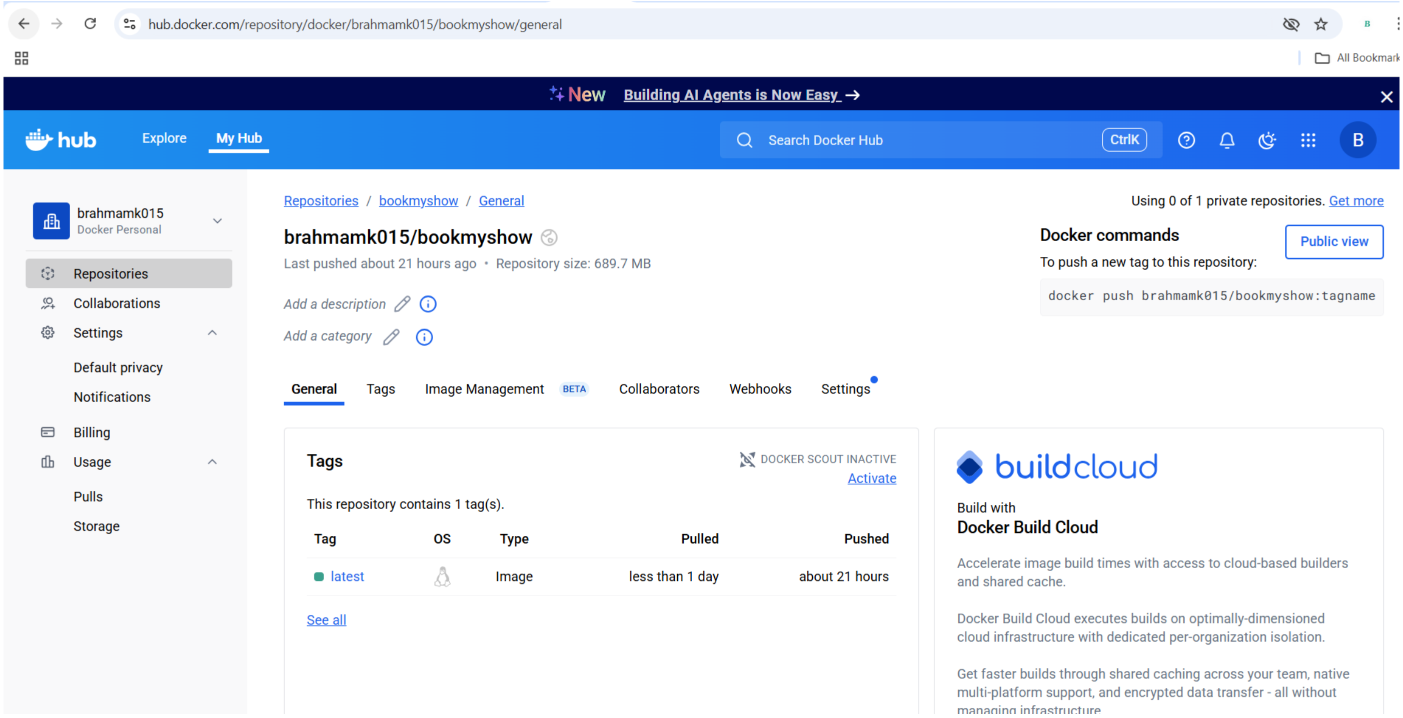
Implementation Details:

- A **Docker image** was built using the custom Dockerfile.
- The image was **pushed to DockerHub** directly via the Jenkins pipeline to maintain automation.
- Jenkins job validated the deployment by running the container and verifying accessibility on **port 3000**.
- Additionally, the **Docker image was pulled from DockerHub to a local machine** (laptop) and tested by running the container.
- The application was successfully accessed via **localhost:8080** in the local environment, proving portability and consistency of the image.

Validation

- **DockerHub Repository:** https://hub.docker.com/r/brahmamk015/bookmyshow/tags

- **Screenshot Evidence:**
  - DockerHub repository containing the pushed image.
  - Application running and accessible via http://localhost:8080 on the local machine.
  - Application running and accessible on **AWS EC2 instance** → http://<EC2-Public-IP>:8080



Using 0 of 1 private repositories. Get more

brahmamk015/bookmyshow

Last pushed about 21 hours ago • Repository size: 689.7 MB

Docker commands

To push a new tag to this repository:

```
docker push brahmamk015/bookmyshow:tagname
```

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	less than 1 day	about 21 hours

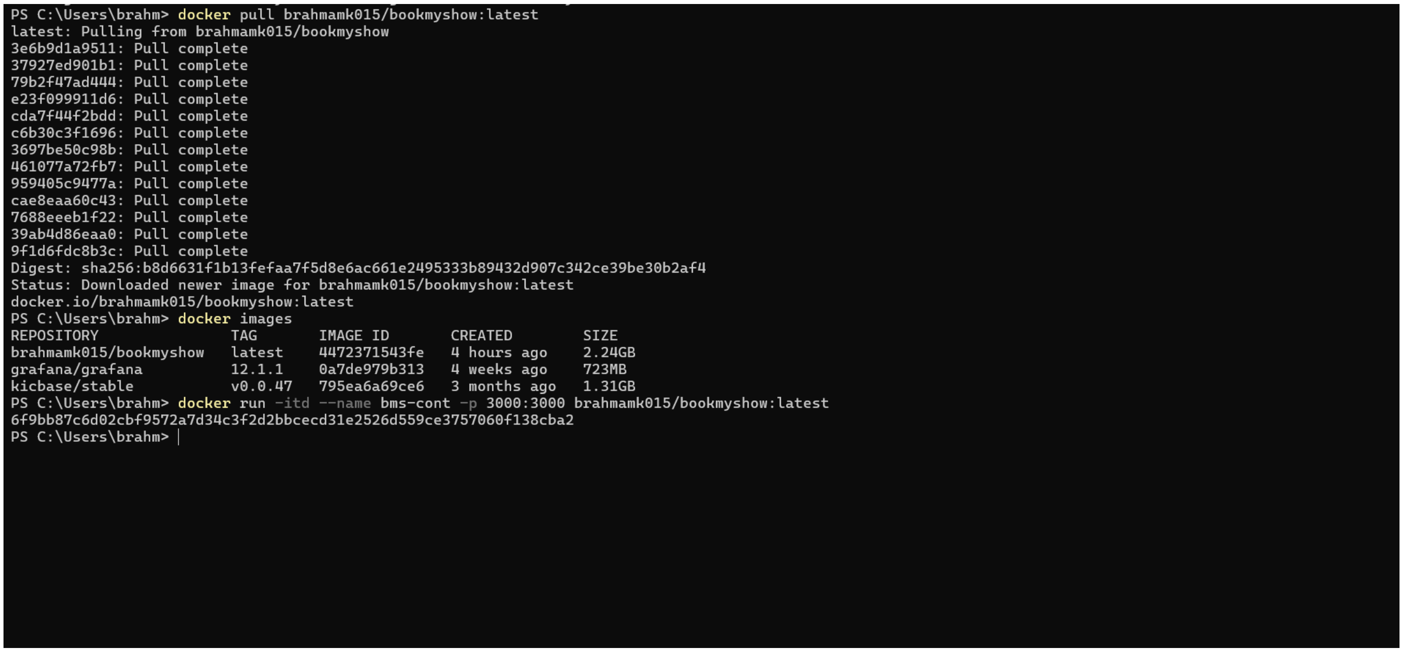
Build with

Docker Build Cloud

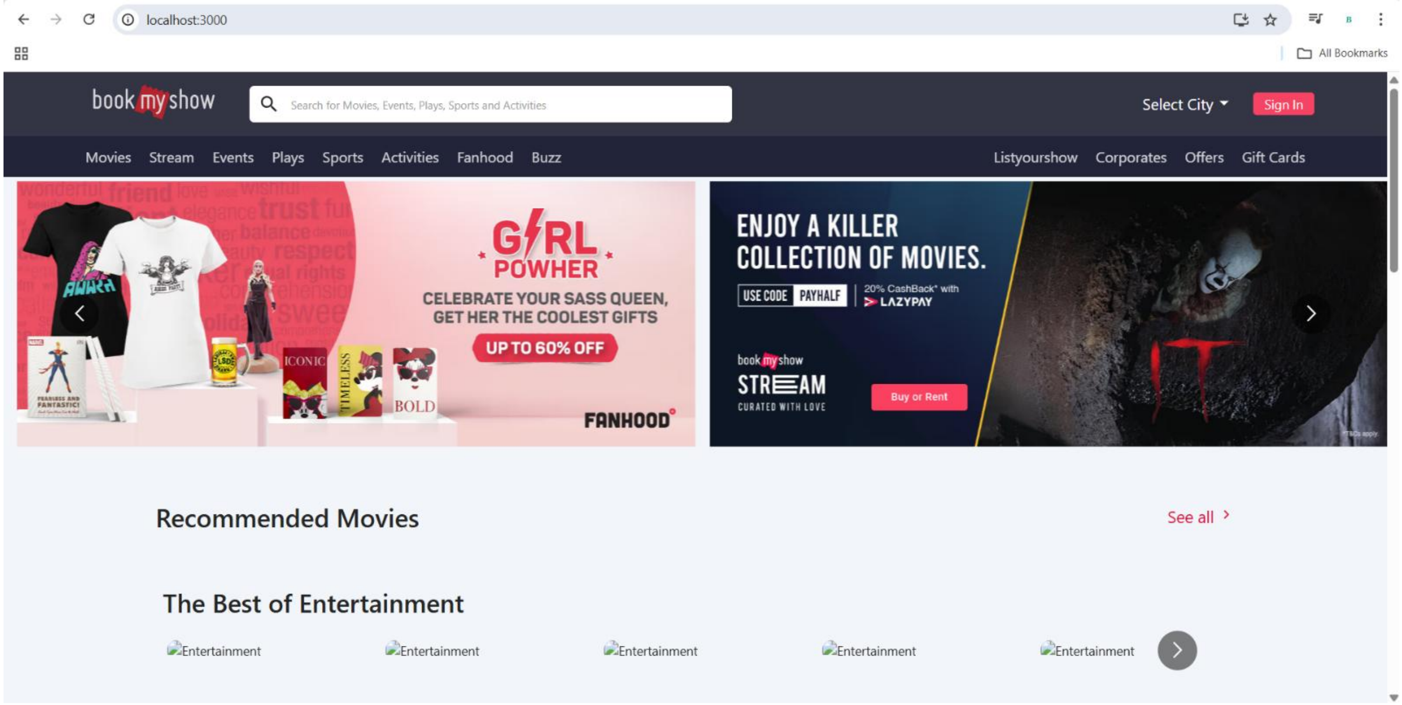
Accelerate image build times with access to cloud-based builders and shared cache.

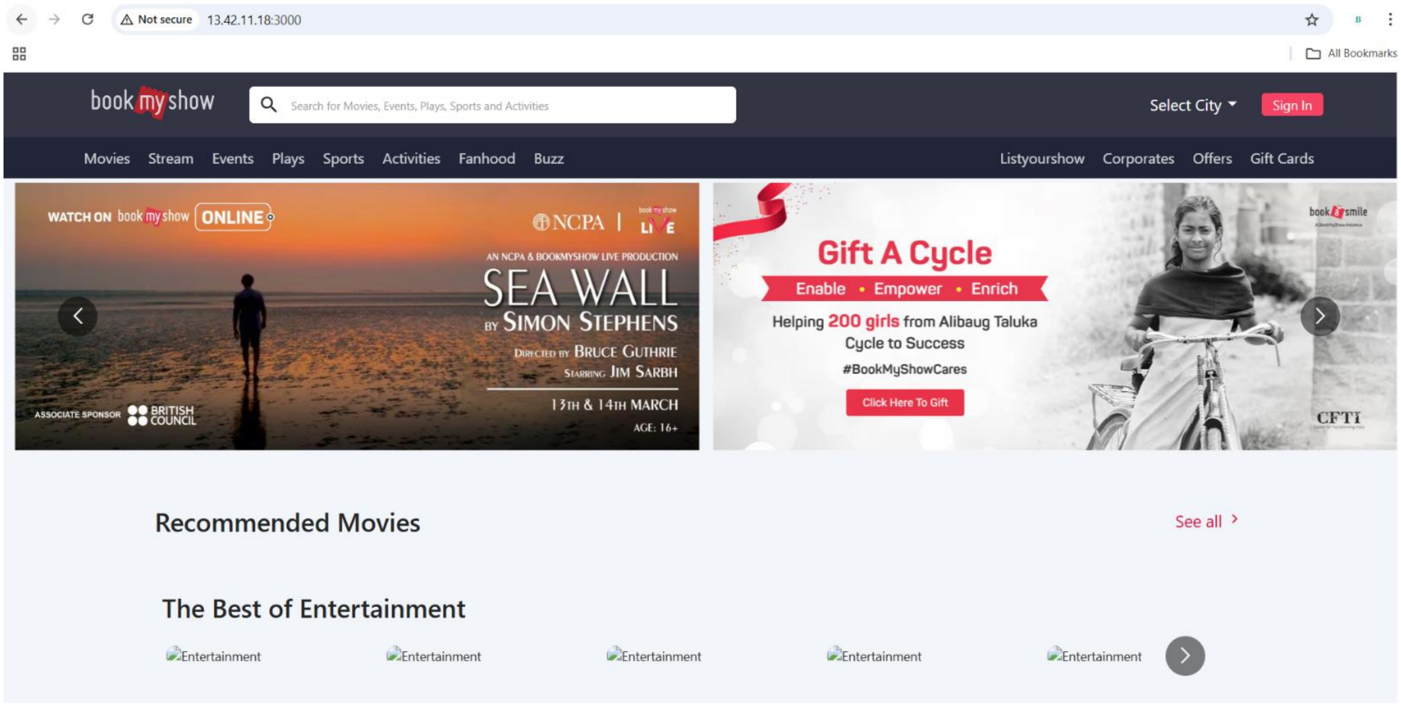
Docker Build Cloud executes builds on optimally-dimensioned cloud infrastructure with dedicated per-organization isolation.

Get faster builds through shared caching across your team, native multi-platform support, and encrypted data transfer - all without managing infrastructure



```
PS C:\Users\brahm> docker pull brahmamk015/bookmyshow:latest
latest: Pulling from brahmamk015/bookmyshow
3e6b9d1a9511: Pull complete
37927ed901b1: Pull complete
79b2f47ad444: Pull complete
e23f099911d6: Pull complete
cda7f44f2bdd: Pull complete
c6b30c3f1696: Pull complete
3697be50c98b: Pull complete
461077a72fb7: Pull complete
959405c9477a: Pull complete
cae8eaa60c43: Pull complete
7688eeeb1f22: Pull complete
39ab4d86eaa0: Pull complete
9f1d6fdc8b3c: Pull complete
Digest: sha256:b8d6631f1b13fefaa7f5d8e6ac661e2495333b89432d907c342ce39be30b2af4
Status: Downloaded newer image for brahmamk015/bookmyshow:latest
docker.io/brahmamk015/bookmyshow:latest
PS C:\Users\brahm> docker images
REPOSITORY               TAG       IMAGE ID       CREATED        SIZE
brahmamk015/bookmyshow   latest    44723715 43fe   4 hours ago    2.24GB
grafana/grafana          12.1.1    0a7de979b313   4 weeks ago    723MB
kicbase/stable           v0.0.47   795ea6a69ce6   3 months ago   1.31GB
PS C:\Users\brahm> docker run -itd --name bms-cont -p 3000:3000 brahmamk015/bookmyshow:latest
6f9bb87c6d02cbf9572a7d34c3f2d2bbcecd31e2526d559ce3757060f138cba2
PS C:\Users\brahm> |
```





Dockerfile :

```
# Use Node.js 18 (or your Jenkins-configured version)
FROM node:18

# Set working directory
WORKDIR /app

# Copy package.json and package-lock.json
COPY package.json package-lock.json ./

# Force install a compatible PostCSS version to fix the issue
RUN npm install postcss@8.4.21 postcss-safe-parser@6.0.0 --legacy-peer-deps

# Install dependencies
RUN npm install

# Copy the entire project
COPY . .

# Expose port 3000
EXPOSE 3000

# Set environment variable to prevent OpenSSL errors
ENV NODE_OPTIONS=--openssl-legacy-provider
ENV PORT=3000

# Start the application
CMD ["npm", "start"]
```

Step 5: Kubernetes Deployment (EKS) :

===================================

The application was deployed to **AWS EKS** using Kubernetes manifests.

Deployment.yml :

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: bms-app
  labels:
    app: bms
spec:
  replicas: 2
  selector:
    matchLabels:
      app: bms
  template:
    metadata:
      labels:
        app: bms
    spec:
      containers:
      - name: bms-container
        image: brahmamk015/bookmyshow:latest # Replace with your Docker image
        ports:
        - containerPort: 3000  # Replace with the port your app runs on
```

service.yml :

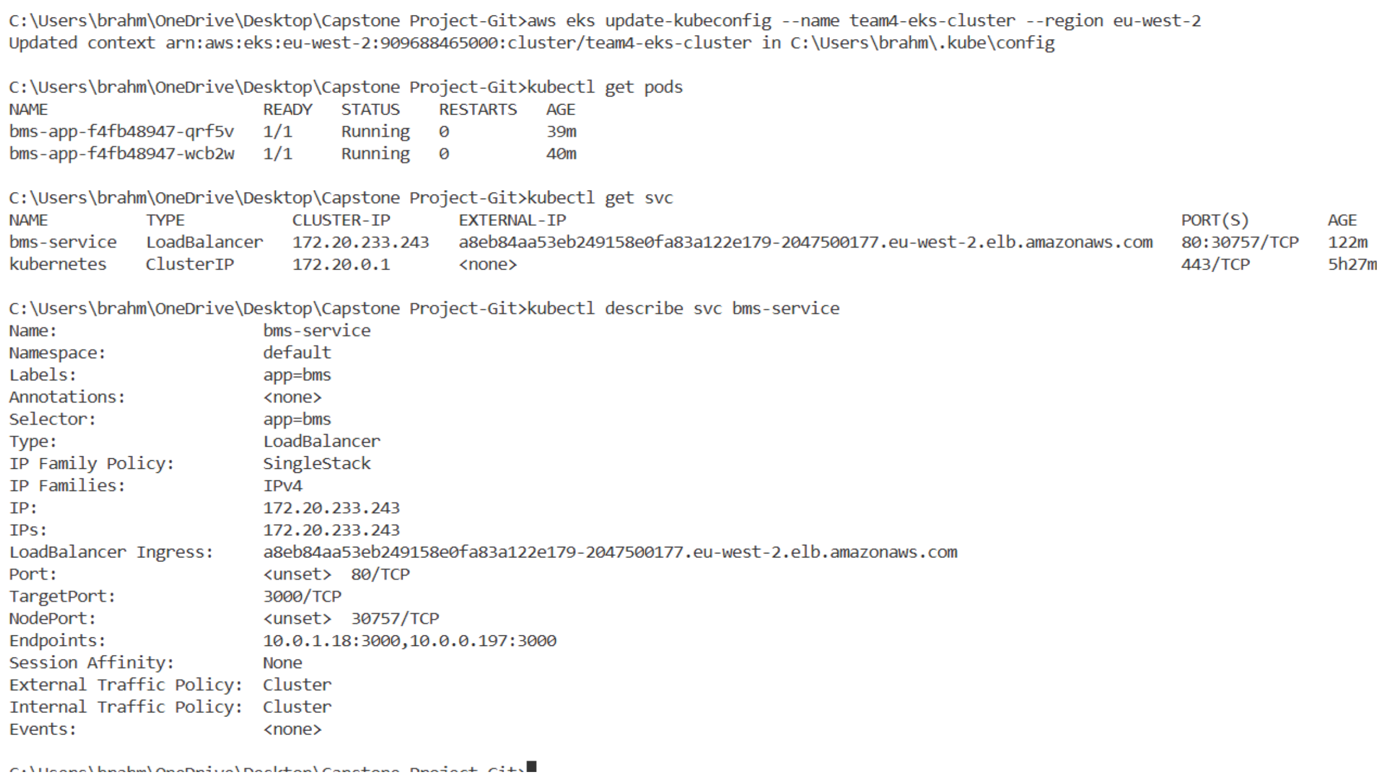
```
apiVersion: v1
kind: Service
metadata:
  name: bms-service
  labels:
    app: bms
spec:
  type: LoadBalancer
  ports:
  - port: 80
    targetPort: 3000  # Replace with the port your app runs on
  selector:
    app: bms
```

Validation:

- Verified via kubectl get pods & kubectl get svc outputs.

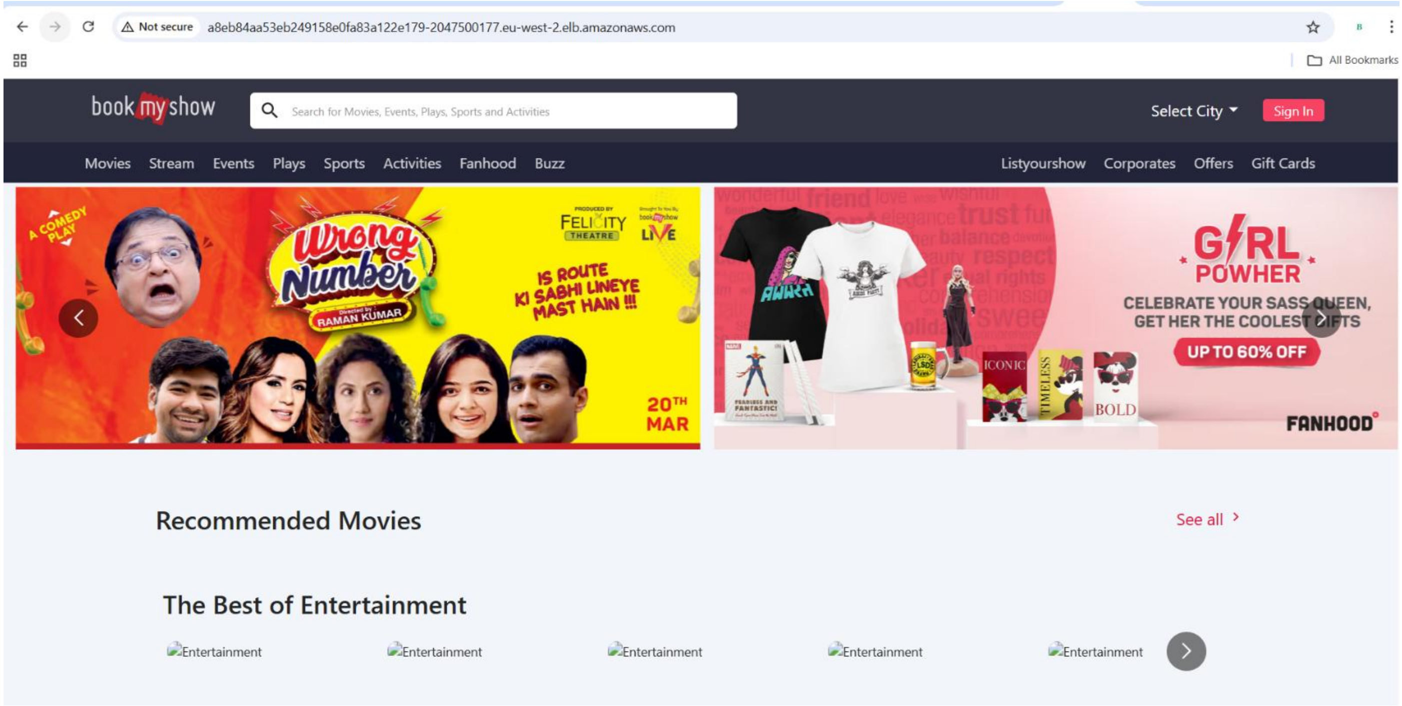
- Application successfully deployed on EKS cluster.

Figure: Verified deployment on EKS using kubectl get pods and kubectl get svc outputs, confirming running pods and LoadBalancer service exposure.



```
C:\Users\brahm\OneDrive\Desktop\Capstone Project-Git>aws eks update-kubeconfig --name team4-eks-cluster --region eu-west-2
Updated context arn:aws:eks:eu-west-2:909688465000:cluster/team4-eks-cluster in C:\Users\brahm\.kube\config

C:\Users\brahm\OneDrive\Desktop\Capstone Project-Git>kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
bms-app-f4fb48947-qrf5v     1/1     Running   0          39m
bms-app-f4fb48947-wcb2w     1/1     Running   0          40m

C:\Users\brahm\OneDrive\Desktop\Capstone Project-Git>kubectl get svc
NAME          TYPE           CLUSTER-IP      EXTERNAL-IP                                                                PORT(S)        AGE
bms-service   LoadBalancer   172.20.233.243  a8eb84aa53eb249158e0fa83a122e179-2047500177.eu-west-2.elb.amazonaws.com   80:30757/TCP   122m
kubernetes    ClusterIP      172.20.0.1      <none>                                                                     443/TCP        5h27m

C:\Users\brahm\OneDrive\Desktop\Capstone Project-Git>kubectl describe svc bms-service
Name:                     bms-service
Namespace:                default
Labels:                   app=bms
Annotations:              <none>
Selector:                 app=bms
Type:                     LoadBalancer
IP Family Policy:         SingleStack
IP Families:              IPv4
IP:                       172.20.233.243
IPs:                      172.20.233.243
LoadBalancer Ingress:     a8eb84aa53eb249158e0fa83a122e179-2047500177.eu-west-2.elb.amazonaws.com
Port:                     <unset>  80/TCP
TargetPort:               3000/TCP
NodePort:                 <unset>  30757/TCP
Endpoints:                10.0.1.18:3000,10.0.0.197:3000
Session Affinity:         None
External Traffic Policy:  Cluster
Internal Traffic Policy:  Cluster
Events:                   <none>

C:\Users\brahm\OneDrive\Desktop\Capstone Project-Git>
```

Figure: The BMS application successfully deployed on the EKS cluster.Access confirmed via the LoadBalancer endpoint in a web browser :



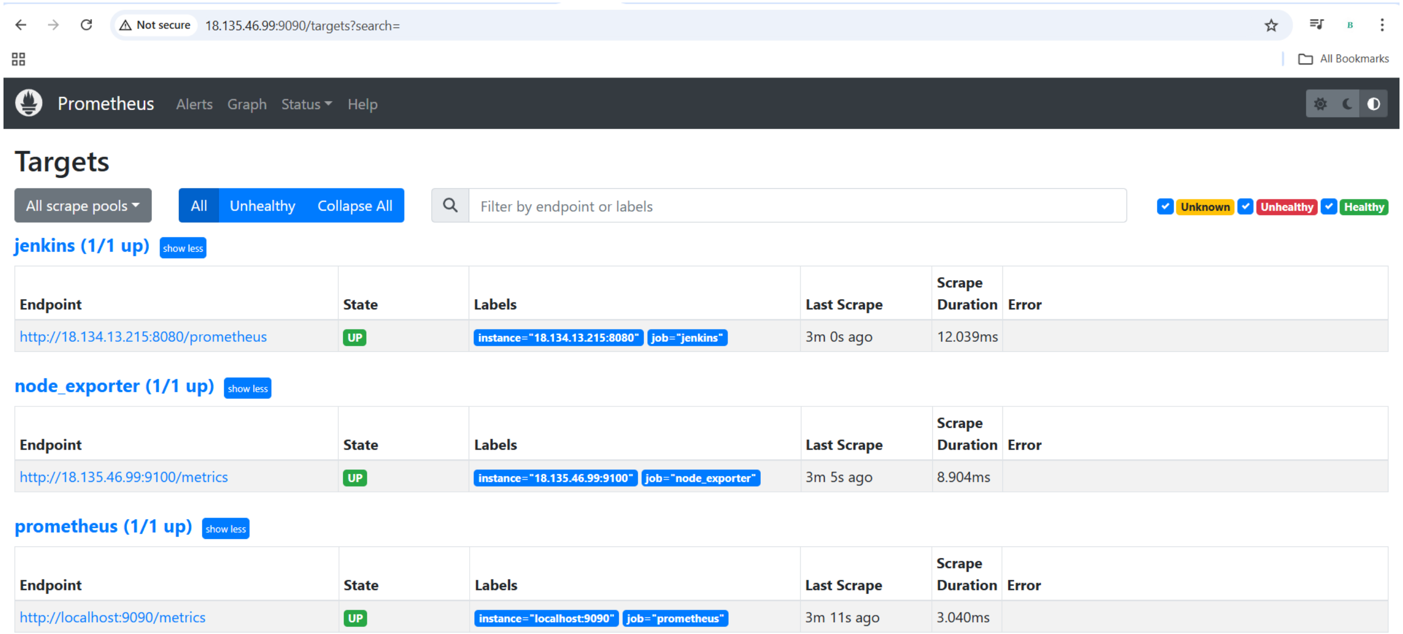
Step 6: Monitoring & Observability :

=================================

- **Prometheus & Node Exporter** installed for system metrics.
- **Grafana** configured with Prometheus as a data source.
- Dashboards created for:
  - Node Health
  - Jenkins Performance

Deliverables:

- Prometheus & Grafana screenshots



Targets

jenkins (1/1 up)

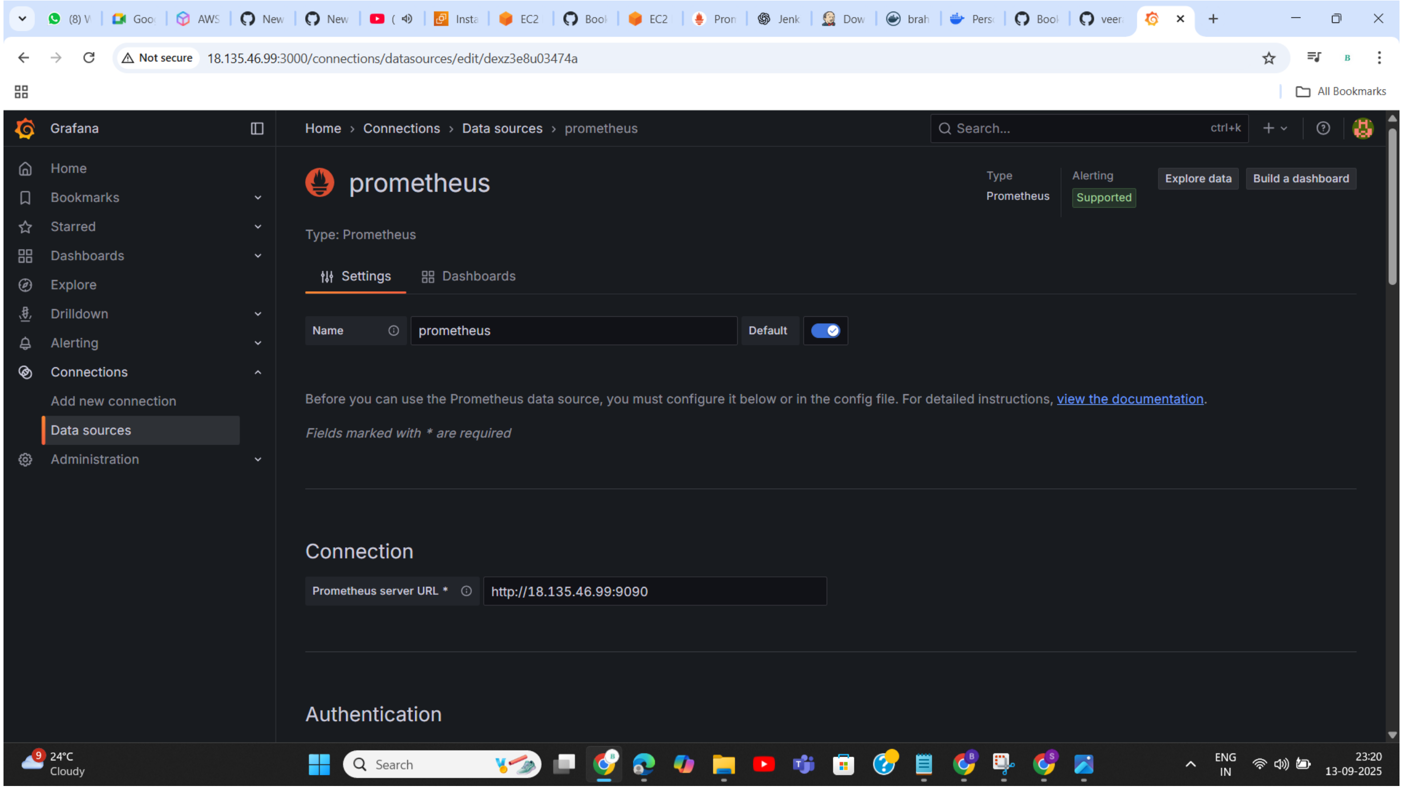
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://18.134.13.215:8080/prometheus	UP	instance="18.134.13.215:8080" job="jenkins"	3m 0s ago	12.039ms	

node_exporter (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://18.135.46.99:9100/metrics	UP	instance="18.135.46.99:9100" job="node_exporter"	3m 5s ago	8.904ms	

prometheus (1/1 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://localhost:9090/metrics	UP	instance="localhost:9090" job="prometheus"	3m 11s ago	3.040ms	



18.135.46.99:3000/connections/datasources/edit/dexz3e8u03474a

Home › Connections › Data sources › prometheus

prometheus

Type: Prometheus

Type: Prometheus | Alerting: Supported

Name: prometheus | Default: on

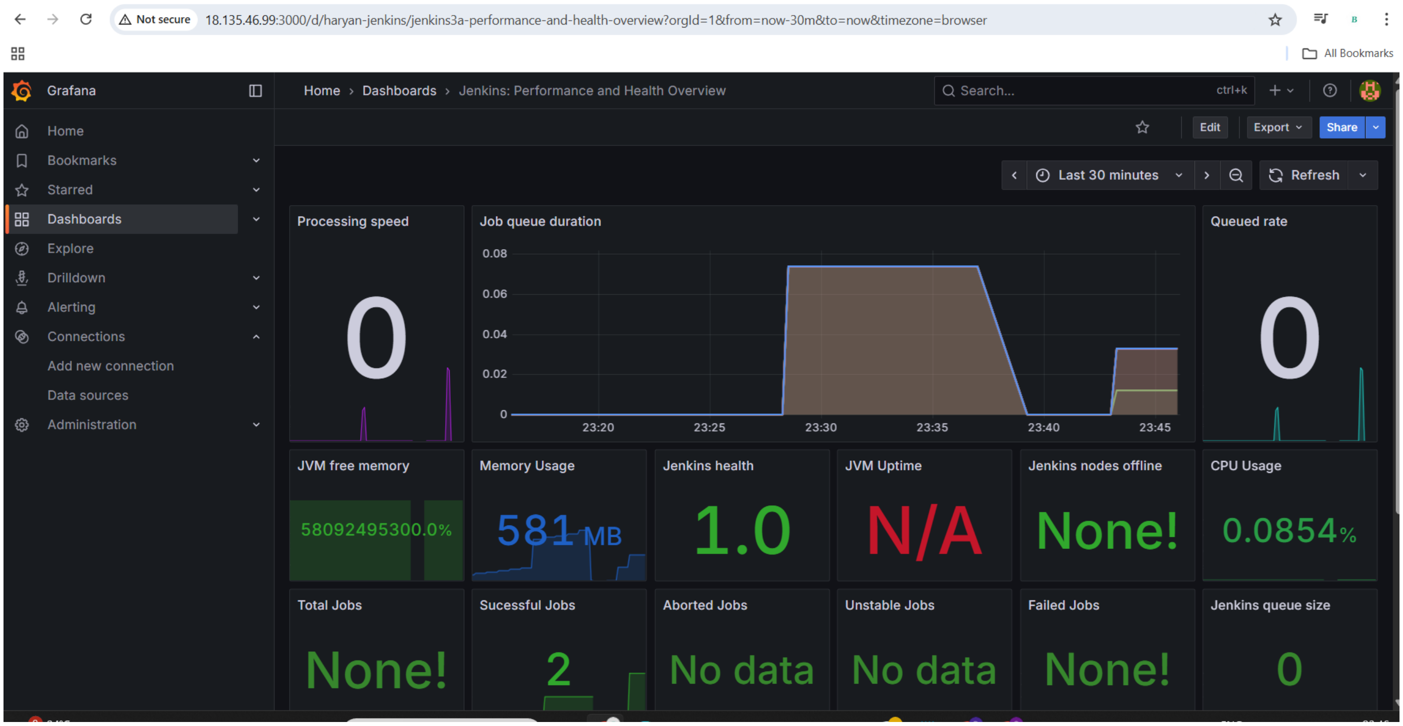
Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, view the documentation.

*Fields marked with * are required*

Connection

Prometheus server URL *: http://18.135.46.99:9090

Authentication

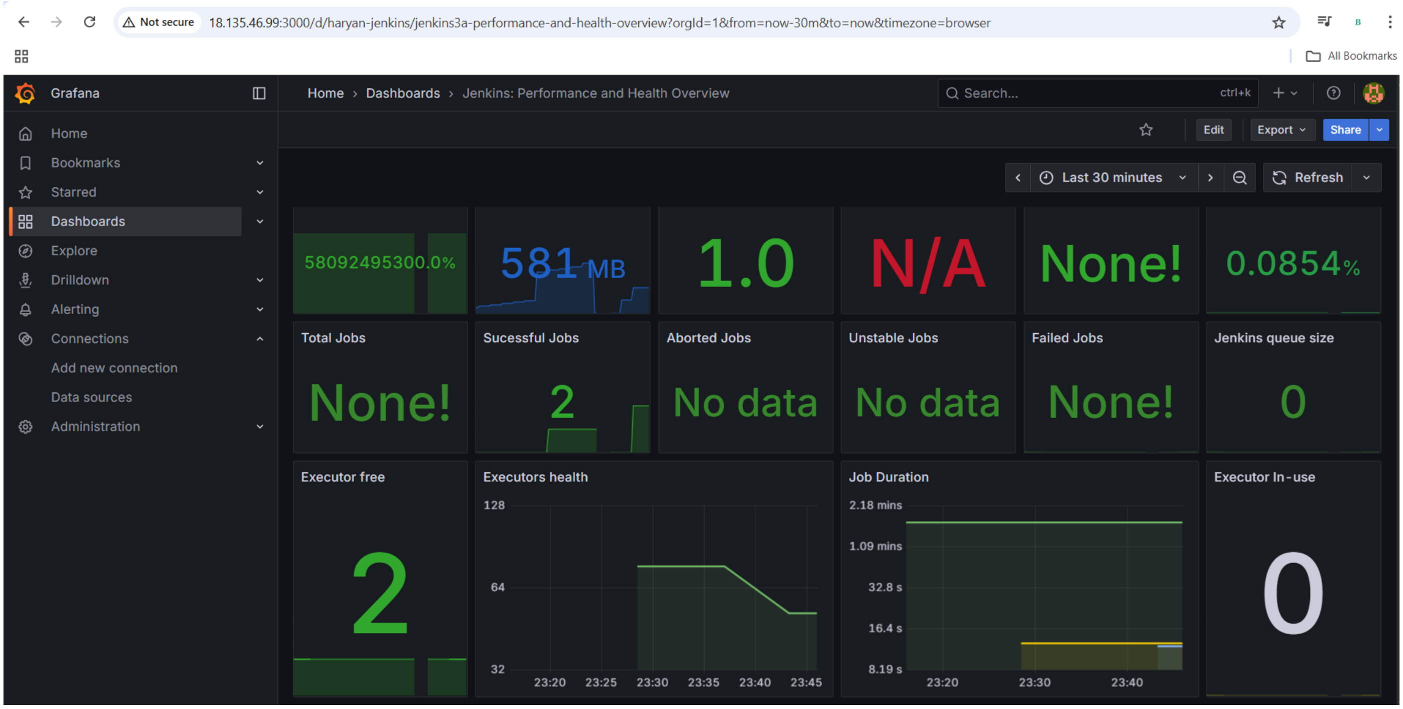


18.135.46.99:3000/d/haryan-jenkins/jenkins3a-performance-and-health-overview?orgId=1&from=now-30m&to=now&timezone=browser

Home › Dashboards › Jenkins: Performance and Health Overview

Last 30 minutes

Panel	Value
Processing speed	0
Queued rate	0
JVM free memory	58092495300.0%
Memory Usage	581 MB
Jenkins health	1.0
JVM Uptime	N/A
Jenkins nodes offline	None!
CPU Usage	0.0854%
Total Jobs	None!
Sucessful Jobs	2
Aborted Jobs	No data
Unstable Jobs	No data
Failed Jobs	None!
Jenkins queue size	0



Not secure 18.135.46.99:3000/d/haryan-jenkins/jenkins3a-performance-and-health-overview?orgId=1&from=now-30m&to=now&timezone=browser

Home › Dashboards › Jenkins: Performance and Health Overview

Last 30 minutes

58092495300.0%

581 MB

1.0

N/A

None!

0.0854%

Total Jobs: None!

Sucessful Jobs: 2

Aborted Jobs: No data

Unstable Jobs: No data

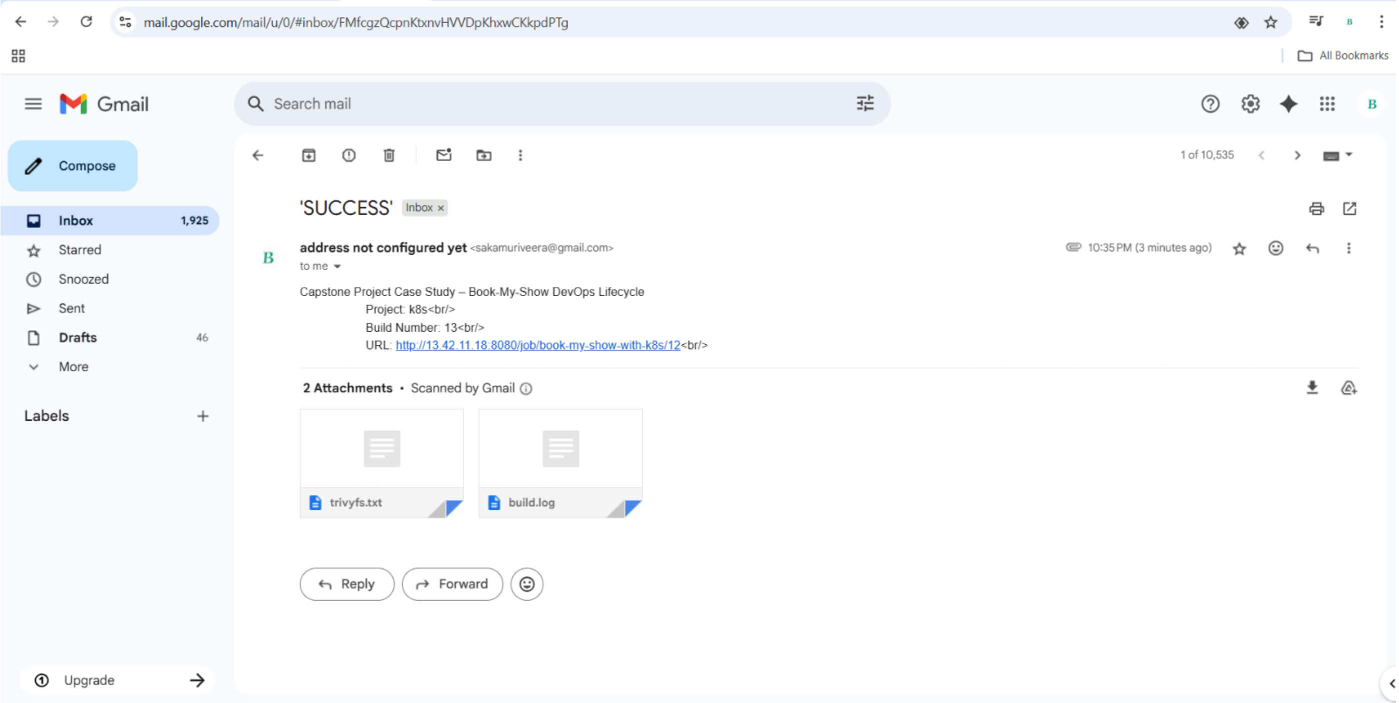
Failed Jobs: None!

Jenkins queue size: 0

Executor free: 2

Executor In-use: 0

Step-7 : Email Notifications configured in Jenkins with logs and scan reports.



'SUCCESS' Inbox ×

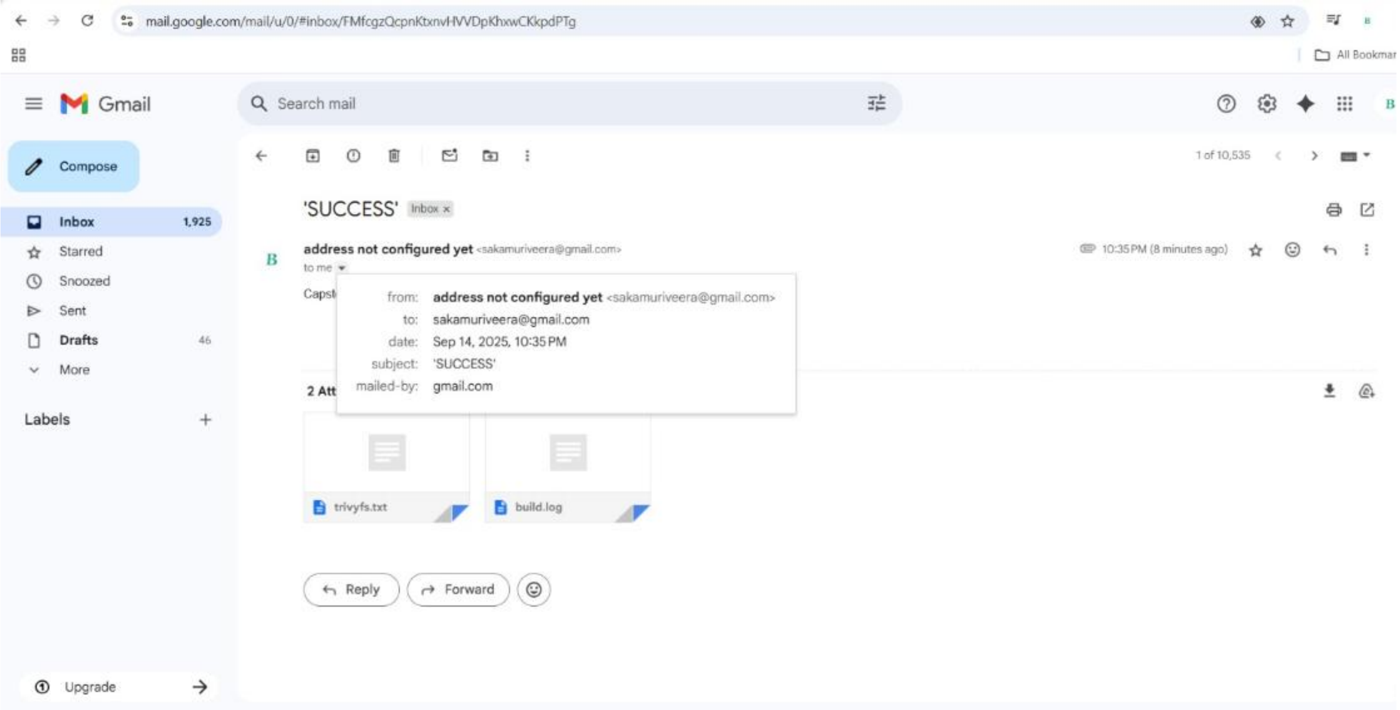
address not configured yet <sakamuriveera@gmail.com>
to me

10:35 PM (3 minutes ago)

Capstone Project Case Study – Book-My-Show DevOps Lifecycle
Project: k8s<br/>
Build Number: 13<br/>
URL: http://13.42.11.18:8080/job/book-my-show-with-k8s/12<br/>

2 Attachments • Scanned by Gmail

trivyfs.txt
build.log



'SUCCESS' Inbox ×

address not configured yet <sakamuriveera@gmail.com>
to me

10:35 PM (8 minutes ago)

from: **address not configured yet** <sakamuriveera@gmail.com>
to: sakamuriveera@gmail.com
date: Sep 14, 2025, 10:35 PM
subject: 'SUCCESS'
mailed-by: gmail.com

trivyfs.txt
build.log

Architure Diagrams :

Pipeline 1: Book-My-Show without Kubernetes

Flow:

- **GitHub → Jenkins → SonarQube → Install Dependencies → OWASP/Trivy → Docker Build & Push → Local Docker Container → Email Notification**

Pipeline 2: Book-My-Show with Kubernetes

Flow:

- **GitHub → Jenkins → SonarQube → Install Dependencies → OWASP/Trivy → Docker Build & Push → Local Docker Container → EKS Deployment → kubectl Validation → Email Notification**

Conclusion :

===========

This project simulated a real-world DevOps pipeline, covering CI/CD automation, containerization, orchestration, and observability. The hands-on integration of tools like Jenkins, Docker, Kubernetes (EKS), and Prometheus/Grafana reflects a production-ready workflow, preparing me for enterprise DevOps roles.

Github Link of my Book My Show Project : https://github.com/Brahmamk3/Book-My-Show.git