Capstone Project– Book-My-Show DevOps Lifecycle

BATCH 1 – YASHASWINI K L

Step 1: Jira Workflow

• Each candidate must assign themselves tasks in Jira.

• Track task progress using Jira board (To Do → In Progress → Done).

• Export the board as part of the final submission.

Step 2: GitHub Workflow

• Clone the GitHub repository: https://github.com/akshu20791/Book-My-Show/

• Create a feature branch and make changes.

• Push changes and raise a Pull Request (PR).

• Review peers' PRs and approve before merging into the main branch.

• Submit the final PR link as deliverable.

Step 3: Jenkins CI/CD Pipeline

Pipeline stages to implement:

1. Clean Workspace
2. Checkout Code from GitHub
3. SonarQube Analysis (Quality Gate)
4. Install Dependencies (NPM)
5. Trivy FS Scan (Optional)
6. OWASP Dependency Check (Optional)
7. Docker Build & Push to DockerHub (via Jenkins)
8. Deploy to Docker Container
9. (Optional) Deploy to Kubernetes (EKS)
10. Email Notification on build result

Note: Candidates must implement this pipeline using a Jenkinsfile and submit the Jenkinsfile as a deliverable.

Step 4: Docker Deployment

• Candidates must write their own Dockerfile to build the BMS app image.

• Build Docker image and push to DockerHub via Jenkins.

• Run container locally and validate accessibility on port 3000.

Step 5: Kubernetes Deployment (EKS)

• Candidates must write their own Kubernetes manifests (`deployment.yaml`, `service.yaml`).

• Deploy the application on EKS cluster.

• Expose service using NodePort or LoadBalancer.

• Validate deployment using 'kubectl get pods' and 'kubectl get svc'.

Step 6: Monitoring & Observability

• Install Prometheus and Node Exporter to collect metrics.

• Integrate Jenkins metrics into Prometheus.

• Install Grafana, configure Prometheus as a data source.

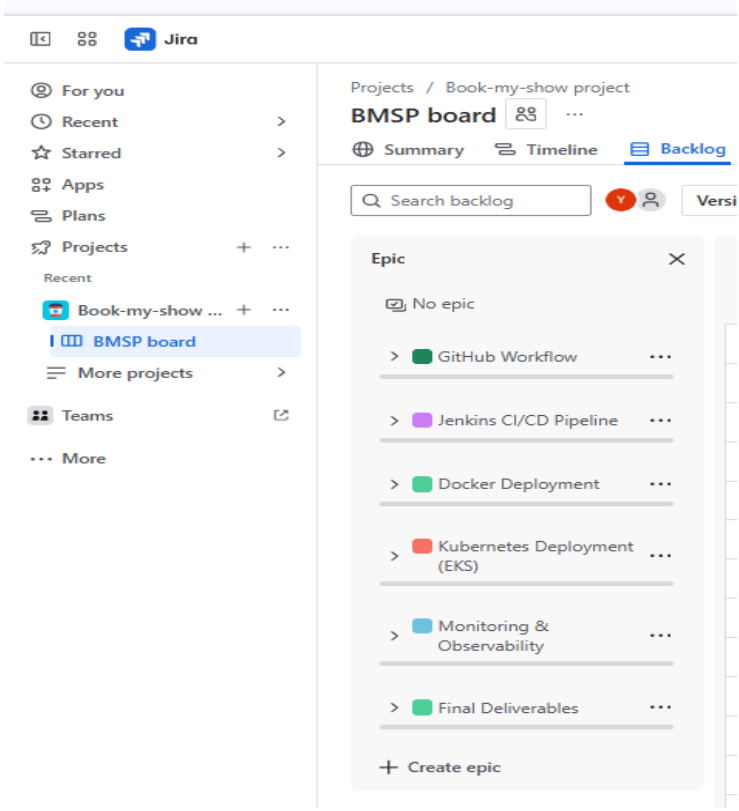
• Add dashboards for Node health and Jenkins performance.

• Submit Grafana screenshots in deliverables.

Step 1: Jira Workflow

1. **Create a Jira Project and Assign Tasks**
   - ➢ Log in to Jira: Access your Jira instance.
   - ➢ Create a New Project:
     - Click on "Projects" in the top navigation bar.
     - Click "Create project."
     - Choose a Project Template: Select "Scrum" and click "Use template."
     - Select "Company-managed project"
     - Name your project: “Book-My-Show Project"
     - Key: Jira will automatically generate a short key ("BMSP").
     - Click "Create project."
   - ➢ Define Epics
     - Click on "Projects" in the top navigation bar.
     - Go to your project's "Backlog" or "Board."
     - Click "Create" or the "+" button.
     - Issue Type: Select "Epic."
     - Summary:

       "GitHub Workflow Setup"
       "Jenkins CI/CD Pipeline"
       "Docker Deployment"
       "Kubernetes Deployment EKS"
       "Monitoring & Observability Setup"
       "Project Deliverables & Reporting"



- To create a task under an Epic:
    - Click "Create."
    - Issue Type: Select "Task."
    - Summary: Detail the specific action.
    - Description: Add any relevant notes or sub-steps.
    - Assignee: Assign it to yourself.
    - Reporter: Set yourself as the reporter.
    - Epic Link: Make sure to link it to the appropriate Epic.
    - Priority: Set a priority (e.g., Medium, High).

Epics, Stories & Tasks

Epic 1: GitHub Workflow

Story 1.1: Set up GitHub Workflow

- Task 1.1.1: Clone the GitHub repository https://github.com/akshu20791/Book-My-Show/
- Task 1.1.2: Create a feature branch and make required code changes
- Task 1.1.3: Push changes to GitHub and raise a Pull Request (PR)
- Task 1.1.4: Review PR (self/peer) and approve before merging
- Task 1.1.5: Submit final PR link as deliverable

Epic 2: Jenkins CI/CD Pipeline

Story 2.1: Set up Jenkins Environment

- Task 2.1.1: Install Jenkins and configure global tools (JDK, Git, NodeJS)
- Task 2.1.2: Install required plugins (Git, Pipeline, Blue Ocean, SonarQube, Docker, Email Extension, etc.)
- Task 2.1.3: Configure Jenkins credentials (GitHub token, DockerHub credentials, SonarQube token)

Story 2.2: Implement Pipeline Stages (Jenkinsfile)

- Task 2.2.1: Stage 1 – Clean Workspace
- Task 2.2.2: Stage 2 – Checkout Code from GitHub
- Task 2.2.3: Stage 3 – SonarQube Analysis (Quality Gate)
- Task 2.2.4: Stage 4 – Install Dependencies (NPM)
- Task 2.2.5: Stage 5 – Trivy FS Scan (Optional)
- Task 2.2.6: Stage 6 – OWASP Dependency Check (Optional)
- Task 2.2.7: Stage 7 – Docker Build & Push to DockerHub
- Task 2.2.8: Stage 8 – Deploy to Docker Container
- Task 2.2.9: Stage 9 – (Optional) Deploy to Kubernetes (EKS)

Epic 3: Docker Deployment

Story 3.1: Containerize BMS App

- Task 3.1.1: Write Dockerfile for BMS App
- Task 3.1.2: Build Docker image locally and test
- Task 3.1.3: Configure Jenkins to build & push Docker image to DockerHub
- Task 3.1.4: Run container locally on port 3000 & validate app accessibility

Epic 4: Kubernetes Deployment (EKS)

Story 4.1: Deploy BMS App to EKS

- Task 4.1.1: Write deployment.yaml and service.yaml manifests
- Task 4.1.2: Deploy manifests to EKS cluster
- Task 4.1.3: Expose service using NodePort/LoadBalancer
- Task 4.1.4: Validate deployment with kubectl get pods & kubectl get svc
- Task 4.1.5: Document deployment proof (screenshots & outputs)

Epic 5: Monitoring & Observability

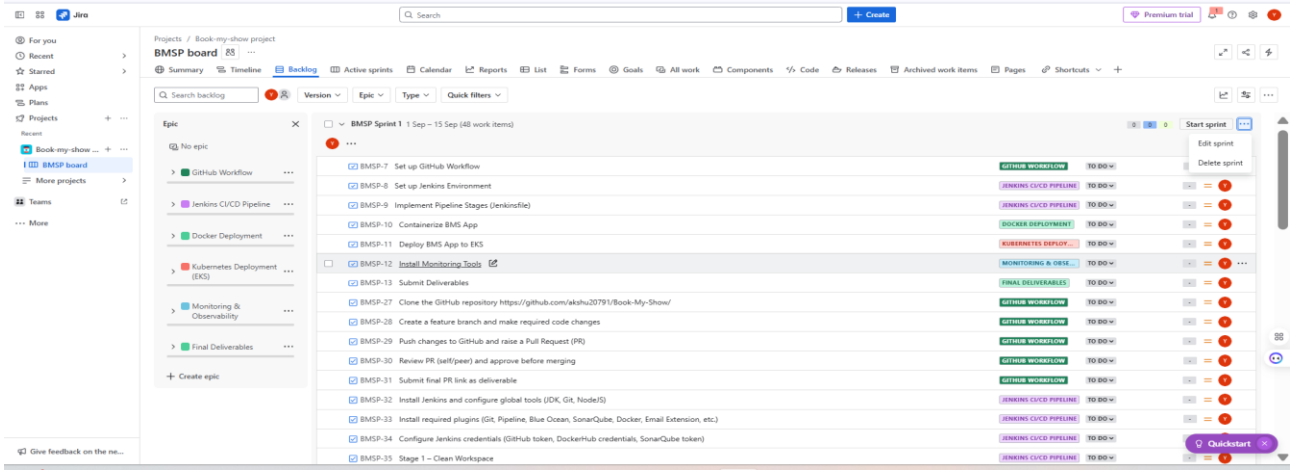
Story 5.1: Install Monitoring Tools

- Task 5.1.1: Install & configure Prometheus + Node Exporter
- Task 5.1.2: Configure Jenkins metrics scraping for Prometheus
- Task 5.1.3: Install Grafana and configure Prometheus as data source
- Task 5.1.4: Create dashboards for Node Health & Jenkins Performance
- Task 5.1.5: Take screenshots of dashboards

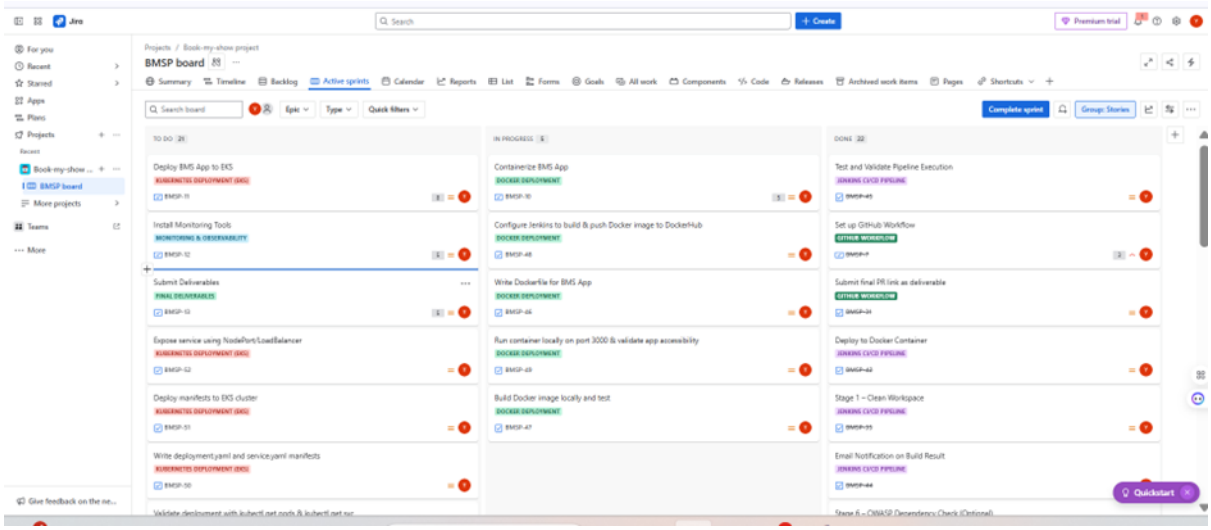
Epic 6: Final Deliverables

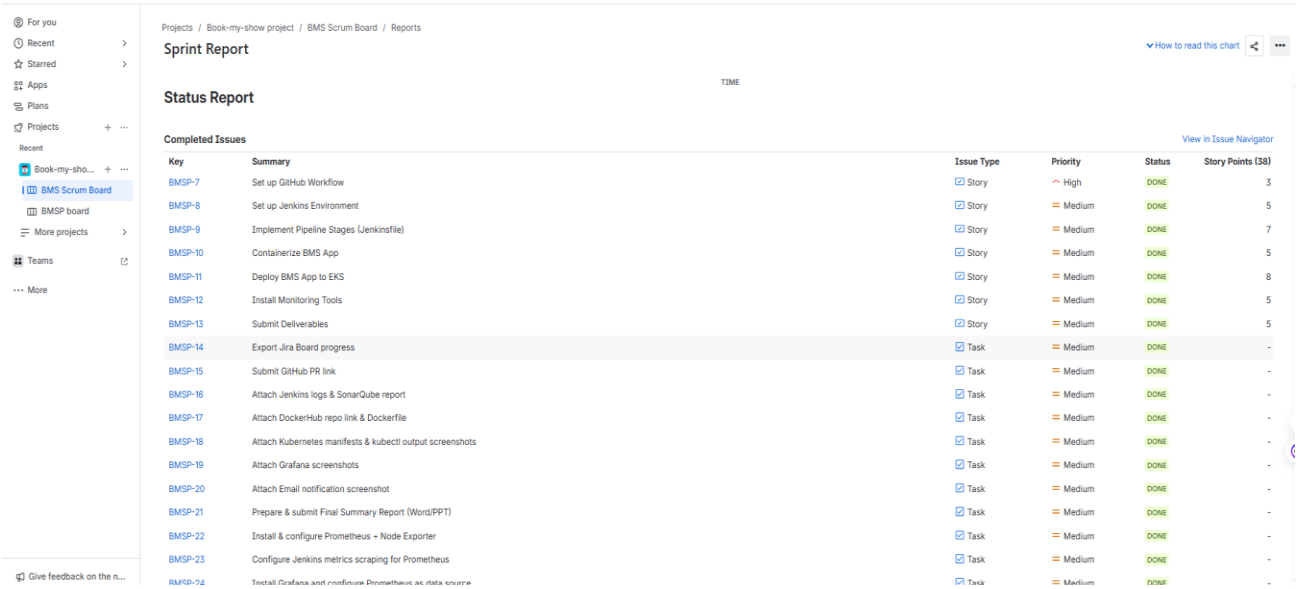
Story 6.1: Submit Deliverables

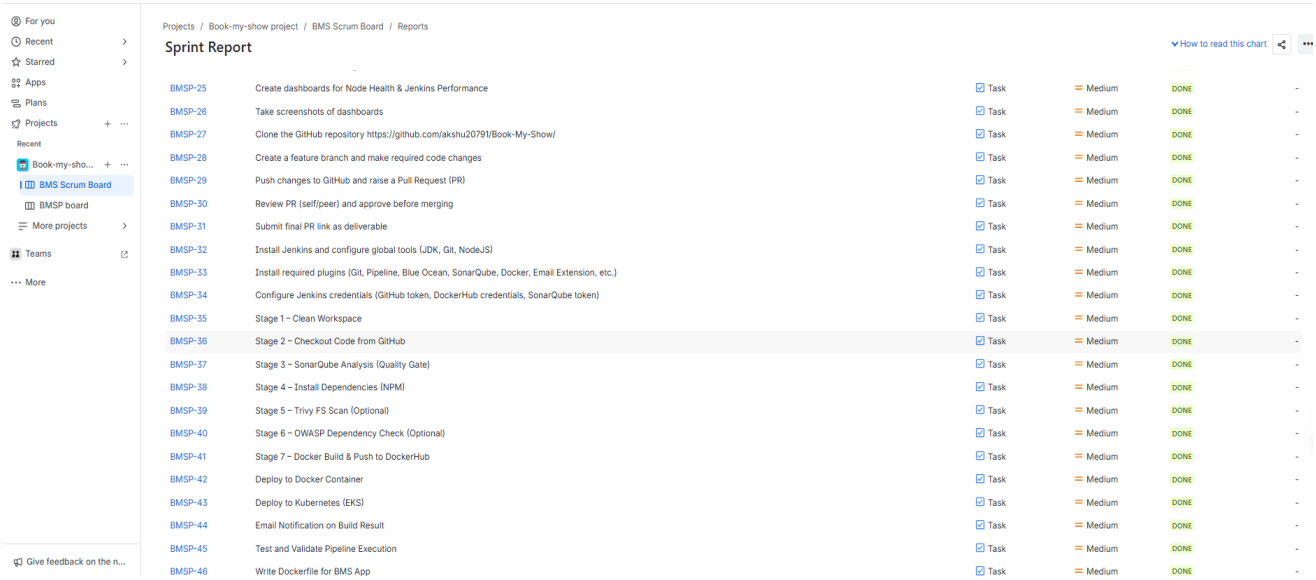
- Task 6.1.1: Export Jira Board progress
- Task 6.1.2: Submit GitHub PR link
- Task 6.1.3: Attach Jenkins logs & SonarQube report
- Task 6.1.4: Attach DockerHub repo link & Dockerfile
- Task 6.1.5: Attach Kubernetes manifests & kubectl output screenshots
- Task 6.1.6: Attach Grafana screenshots
- Task 6.1.7: Attach Email notification screenshot
- Task 6.1.8: Prepare & submit Final Summary Report (Word/PPT)

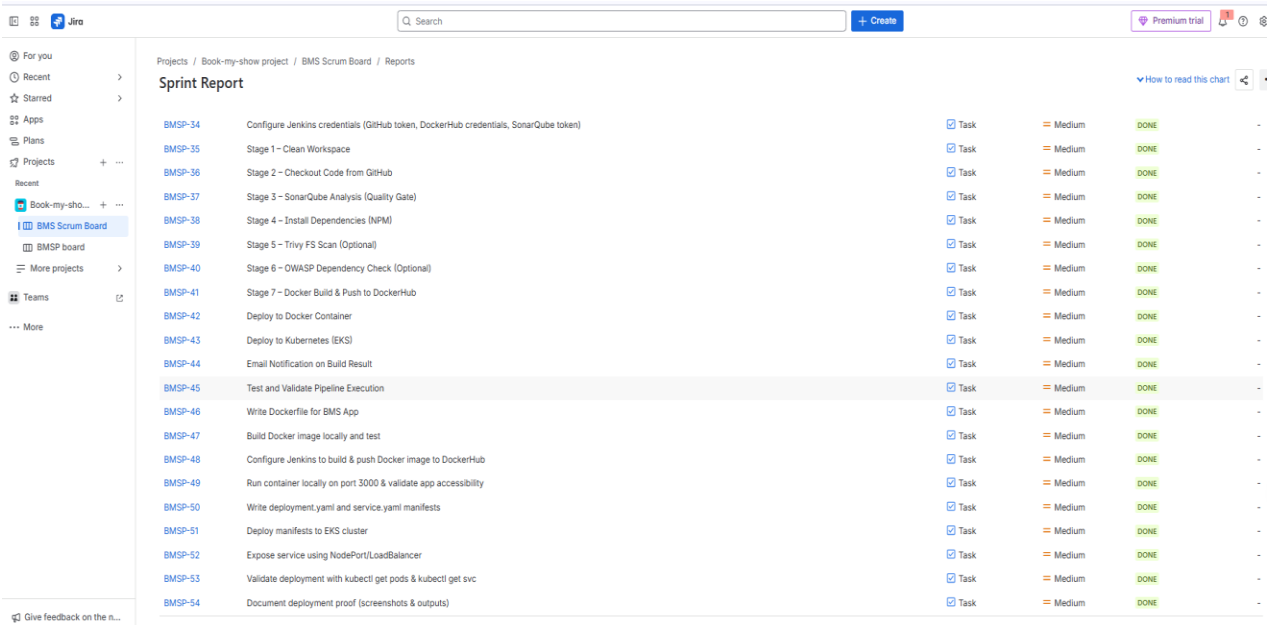


2. **Assign Tasks to Self: As you create each task, make sure to set the "Assignee" field to your own user account.**
3. **Create a Sprint**
    - On your "Backlog" screen, click "Create sprint."
    - You can drag tasks from the backlog into the sprint.
    - Once you've added some tasks you plan to work on, click "Start sprint."
    - Define the sprint duration (e.g.,3weeks).
4. **Manage Your Board:**
    - Go to your project's " Active sprints" (Scrum).
    - Here you will see your tasks in columns like "To Do," "In Progress," and "Done."
    - Drag and drop tasks between columns as you work on them.




For you
Recent
Starred
Apps
Plans
Projects
Recent
Book-my-sho...
BMS Scrum Board
BMSP board
More projects
Teams
More
Give feedback on the n...
Projects / Book-my-show project / BMS Scrum Board / Reports
Sprint Report
How to read this chart
TIME
Status Report
Completed Issues
View in Issue Navigator
Key Summary Issue Type Priority Status Story Points (38)
BMSP-7 Set up GitHub Workflow Story High DONE 3
BMSP-8 Set up Jenkins Environment Story Medium DONE 5
BMSP-9 Implement Pipeline Stages (Jenkinsfile) Story Medium DONE 7
BMSP-10 Containerize BMS App Story Medium DONE 5
BMSP-11 Deploy BMS App to EKS Story Medium DONE 8
BMSP-12 Install Monitoring Tools Story Medium DONE 5
BMSP-13 Submit Deliverables Story Medium DONE 5
BMSP-14 Export Jira Board progress Task Medium DONE -
BMSP-15 Submit GitHub PR link Task Medium DONE -
BMSP-16 Attach Jenkins logs & SonarQube report Task Medium DONE -
BMSP-17 Attach DockerHub repo link & Dockerfile Task Medium DONE -
BMSP-18 Attach Kubernetes manifests & kubectl output screenshots Task Medium DONE -
BMSP-19 Attach Grafana screenshots Task Medium DONE -
BMSP-20 Attach Email notification screenshot Task Medium DONE -
BMSP-21 Prepare & submit Final Summary Report (Word/PPT) Task Medium DONE -
BMSP-22 Install & configure Prometheus + Node Exporter Task Medium DONE -
BMSP-23 Configure Jenkins metrics scraping for Prometheus Task Medium DONE -


For you
Recent
Starred
Apps
Plans
Projects
Recent
Book-my-sho...
BMS Scrum Board
BMSP board
More projects
Teams
More
Give feedback on the n...
Projects / Book-my-show project / BMS Scrum Board / Reports
Sprint Report
How to read this chart
BMSP-25 Create dashboards for Node Health & Jenkins Performance Task Medium DONE -
BMSP-26 Take screenshots of dashboards Task Medium DONE -
BMSP-27 Clone the GitHub repository https://github.com/akshu20791/Book-My-Show/ Task Medium DONE -
BMSP-28 Create a feature branch and make required code changes Task Medium DONE -
BMSP-29 Push changes to GitHub and raise a Pull Request (PR) Task Medium DONE -
BMSP-30 Review PR (self/peer) and approve before merging Task Medium DONE -
BMSP-31 Submit final PR link as deliverable Task Medium DONE -
BMSP-32 Install Jenkins and configure global tools (JDK, Git, NodeJS) Task Medium DONE -
BMSP-33 Install required plugins (Git, Pipeline, Blue Ocean, SonarQube, Docker, Email Extension, etc.) Task Medium DONE -
BMSP-34 Configure Jenkins credentials (GitHub token, DockerHub credentials, SonarQube token) Task Medium DONE -
BMSP-35 Stage 1 – Clean Workspace Task Medium DONE -
BMSP-36 Stage 2 – Checkout Code from GitHub Task Medium DONE -
BMSP-37 Stage 3 – SonarQube Analysis (Quality Gate) Task Medium DONE -
BMSP-38 Stage 4 – Install Dependencies (NPM) Task Medium DONE -
BMSP-39 Stage 5 – Trivy FS Scan (Optional) Task Medium DONE -
BMSP-40 Stage 6 – OWASP Dependency Check (Optional) Task Medium DONE -
BMSP-41 Stage 7 – Docker Build & Push to DockerHub Task Medium DONE -
BMSP-42 Deploy to Docker Container Task Medium DONE -
BMSP-43 Deploy to Kubernetes (EKS) Task Medium DONE -
BMSP-44 Email Notification on Build Result Task Medium DONE -
BMSP-45 Test and Validate Pipeline Execution Task Medium DONE -
BMSP-46 Write Dockerfile for BMS App Task Medium DONE -


Jira
Search
Create
Premium trial
For you
Recent
Starred
Apps
Plans
Projects
Recent
Book-my-sho...
BMS Scrum Board
BMSP board
More projects
Teams
More
Give feedback on the n...
Projects / Book-my-show project / BMS Scrum Board / Reports
Sprint Report
How to read this chart
BMSP-34 Configure Jenkins credentials (GitHub token, DockerHub credentials, SonarQube token) Task Medium DONE -
BMSP-35 Stage 1 – Clean Workspace Task Medium DONE -
BMSP-36 Stage 2 – Checkout Code from GitHub Task Medium DONE -
BMSP-37 Stage 3 – SonarQube Analysis (Quality Gate) Task Medium DONE -
BMSP-38 Stage 4 – Install Dependencies (NPM) Task Medium DONE -
BMSP-39 Stage 5 – Trivy FS Scan (Optional) Task Medium DONE -
BMSP-40 Stage 6 – OWASP Dependency Check (Optional) Task Medium DONE -
BMSP-41 Stage 7 – Docker Build & Push to DockerHub Task Medium DONE -
BMSP-42 Deploy to Docker Container Task Medium DONE -
BMSP-43 Deploy to Kubernetes (EKS) Task Medium DONE -
BMSP-44 Email Notification on Build Result Task Medium DONE -
BMSP-45 Test and Validate Pipeline Execution Task Medium DONE -
BMSP-46 Write Dockerfile for BMS App Task Medium DONE -
BMSP-47 Build Docker image locally and test Task Medium DONE -
BMSP-48 Configure Jenkins to build & push Docker image to DockerHub Task Medium DONE -
BMSP-49 Run container locally on port 3000 & validate app accessibility Task Medium DONE -
BMSP-50 Write deployment.yaml and service.yaml manifests Task Medium DONE -
BMSP-51 Deploy manifests to EKS cluster Task Medium DONE -
BMSP-52 Expose service using NodePort/LoadBalancer Task Medium DONE -
BMSP-53 Validate deployment with kubectl get pods & kubectl get svc Task Medium DONE -
BMSP-54 Document deployment proof (screenshots & outputs) Task Medium DONE -

Step 2: GitHub Workflow

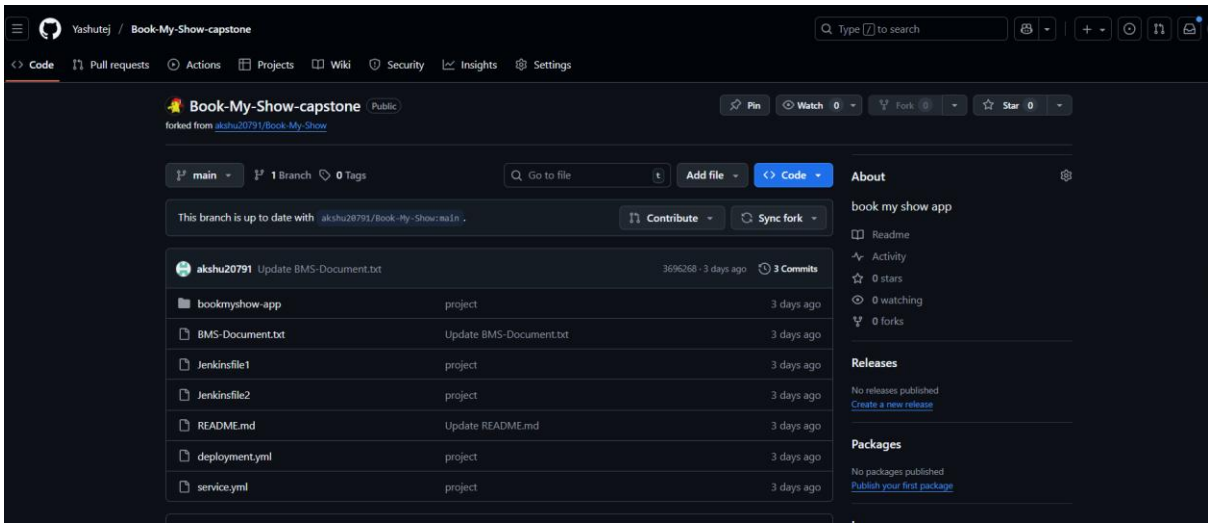
Workflow:

Clone Repo → Create Feature Branch → Make Changes → Commit & Push →

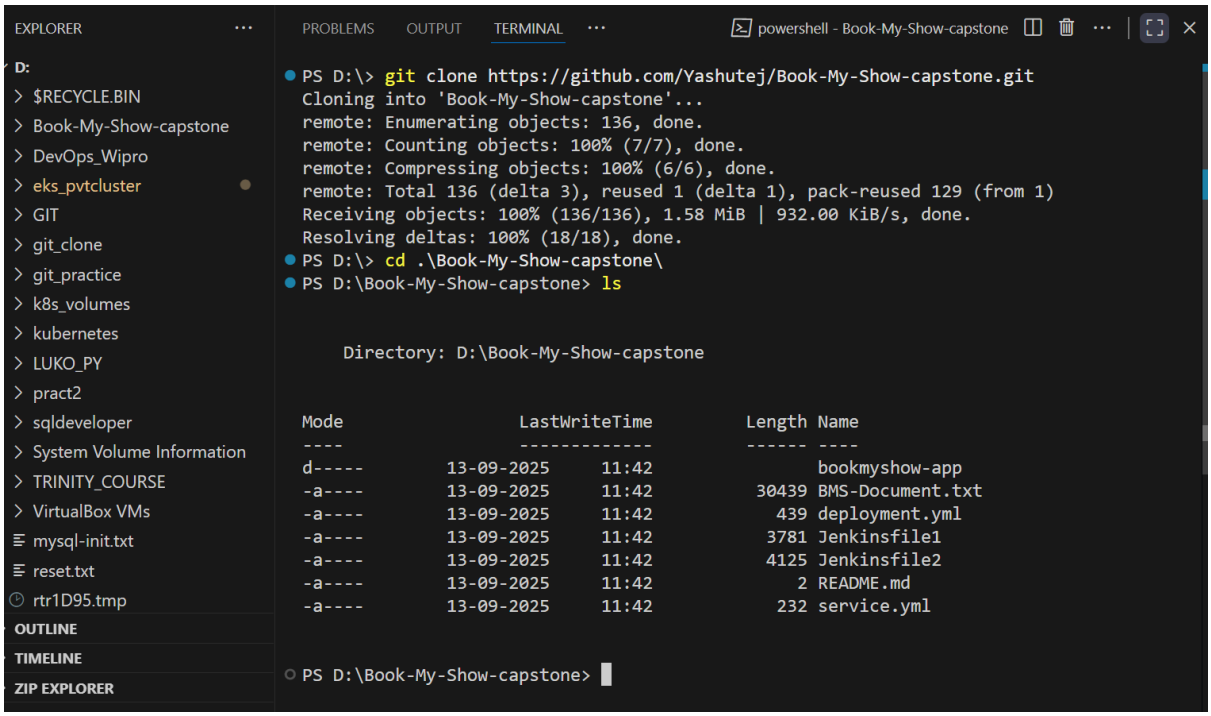
Raise Pull Request → Peer Review & Approval → Merge to Main

Step 1 – Fork Repository

- Go to: https://github.com/akshu20791/Book-My-Show
- Click Fork (top-right corner).
- Select your GitHub account → wait for fork to complete.



Step 2 – Clone Your Fork



Step 3 – Add Upstream

```
PS D:\Book-My-Show-capstone> git remote add upstream https://github.com/akshu20791/Book-My-Show.git
PS D:\Book-My-Show-capstone> git remote -v
origin  https://github.com/Yashutej/Book-My-Show-capstone.git (fetch)
origin  https://github.com/Yashutej/Book-My-Show-capstone.git (push)
upstream        https://github.com/akshu20791/Book-My-Show.git (fetch)
upstream        https://github.com/akshu20791/Book-My-Show.git (push)
```

Step 4 – Create Feature Branch

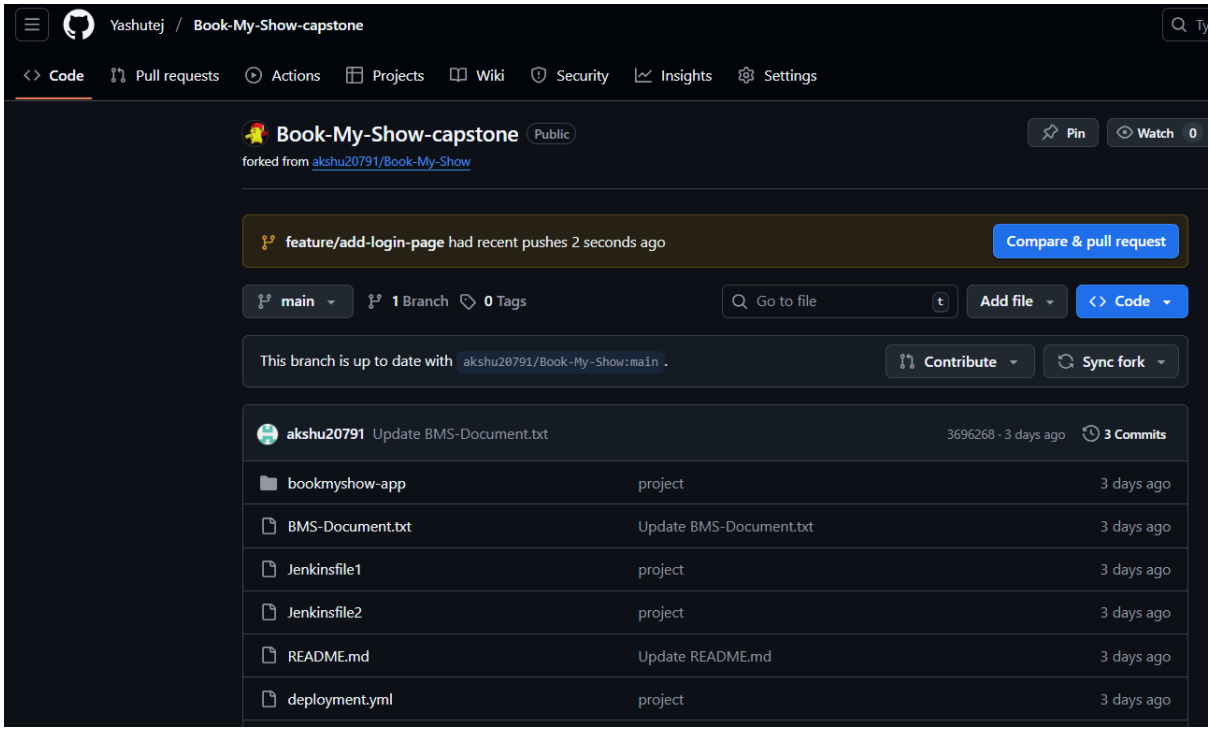
```
PS D:\Book-My-Show-capstone> git checkout -b feature/add-login-page
Switched to a new branch 'feature/add-login-page'
```

Step 5 – Make Changes, Commit, Push

```
PS D:\Book-My-Show-capstone> git add .
PS D:\Book-My-Show-capstone> git commit -m "Added login page feature"
[feature/add-login-page 28a5914] Added login page feature
 2 files changed, 2 insertions(+)
PS D:\Book-My-Show-capstone> git push origin feature/add-login-page
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 420 bytes | 420.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote:
remote: Create a pull request for 'feature/add-login-page' on GitHub by visiting:
remote:      https://github.com/Yashutej/Book-My-Show-capstone/pull/new/feature/add-login-page
remote:
To https://github.com/Yashutej/Book-My-Show-capstone.git
 * [new branch]      feature/add-login-page -> feature/add-login-page
PS D:\Book-My-Show-capstone>
```

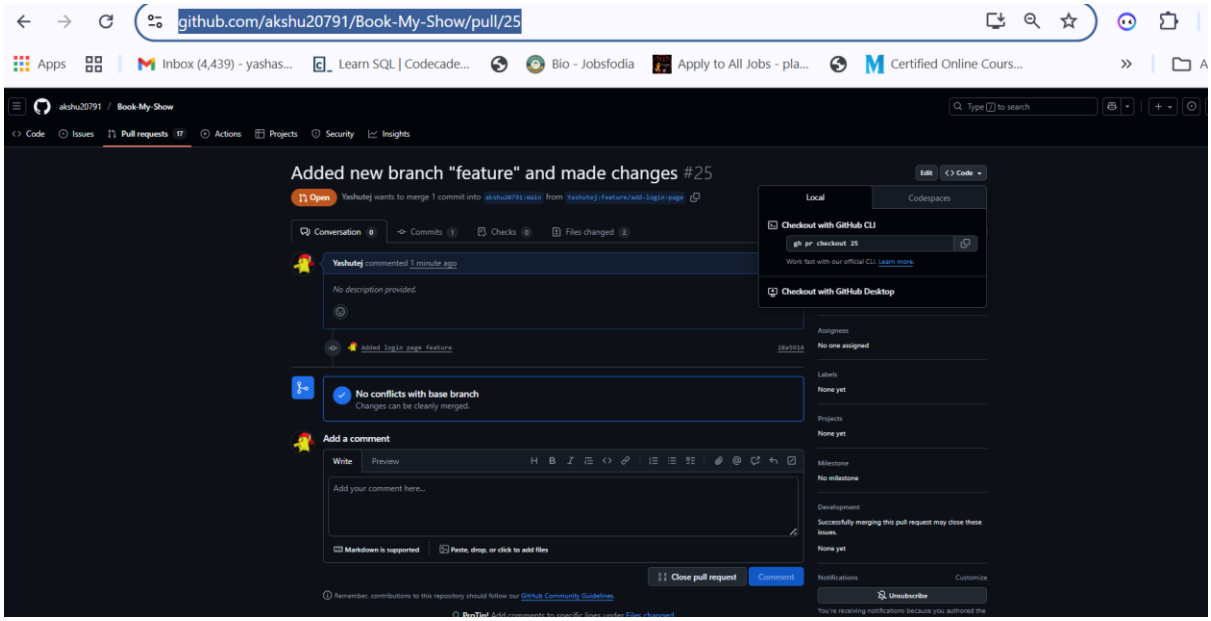
Step 6 – Open Pull Request

- Go to your fork → feature branch → click Compare & Pull Request.
- Base repo: akshu20791/Book-My-Show (main branch)
  Head repo: Yashutej/Book-My-Show-capstone (feature branch)
- Add title + description → Submit PR.



Step 7 – Copy the PR Link

- After submitting, you will land on the **PR details page**.
- PR link: https://github.com/akshu20791/Book-My-Show/pull/25



Step 3: Jenkins CI/CD Pipeline

1. Jenkins and EKS Cluster Setup

Prerequisites:

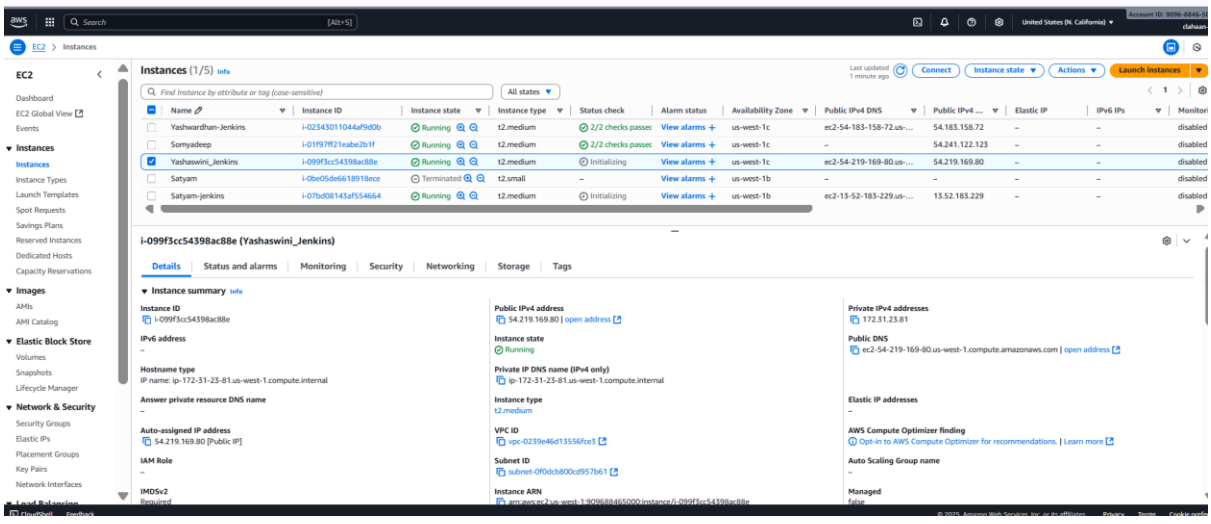
Infrastructure:

- AWS EC2 instance (Ubuntu 20.04+) with Jenkins installed
- Docker installed and configured
- AWS CLI installed and configured
- kubectl installed and configured (or installed at runtime by Jenkins)

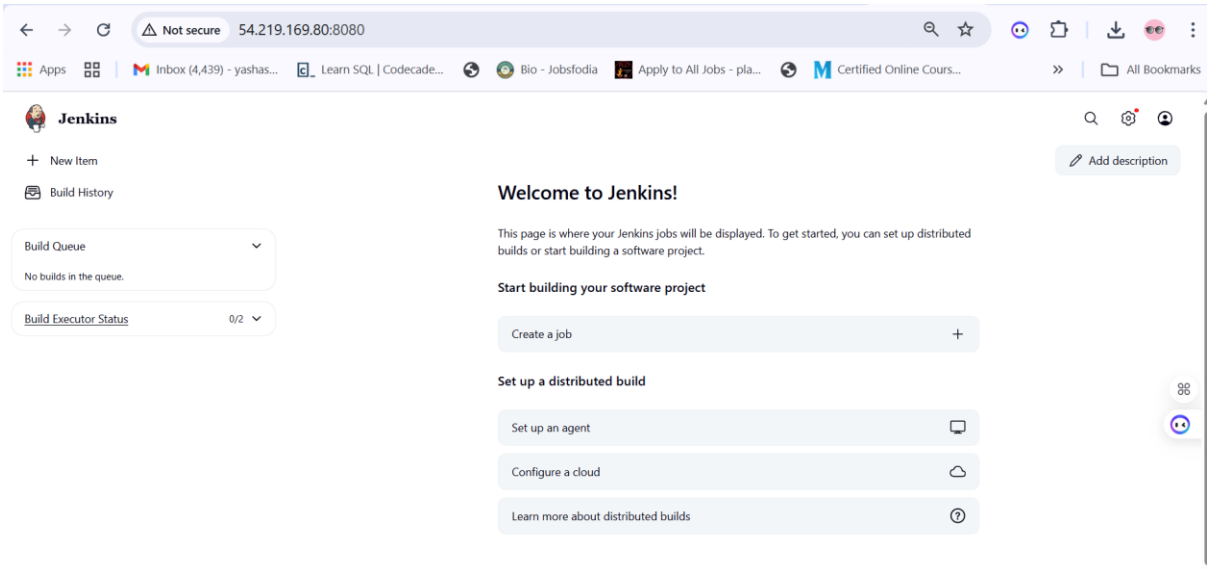
Tools Installed on Jenkins:

- JDK 17
- NodeJS 23
- SonarQube Scanner
- Trivy
- OWASP Dependency Check

➢ Launch EC2 Instance to set up Jenkins Server
Name: Yashaswini_Jenkins
AMI: Ubuntu
Instance type: t2.medium
Select key pair, Security Group and Subnet
Click on "Launch instance"



➢ Connect to putty server using SSH client.
➢ Install Java and Jenkins. Next Enable and start Jenkins on server. Make sure you have to install kubectl and aws cli on Jenkins server.
➢ Access your Jenkins server on browser using instance public IP address with port number
➢ Open port number 8080 in your security group for Jenkins



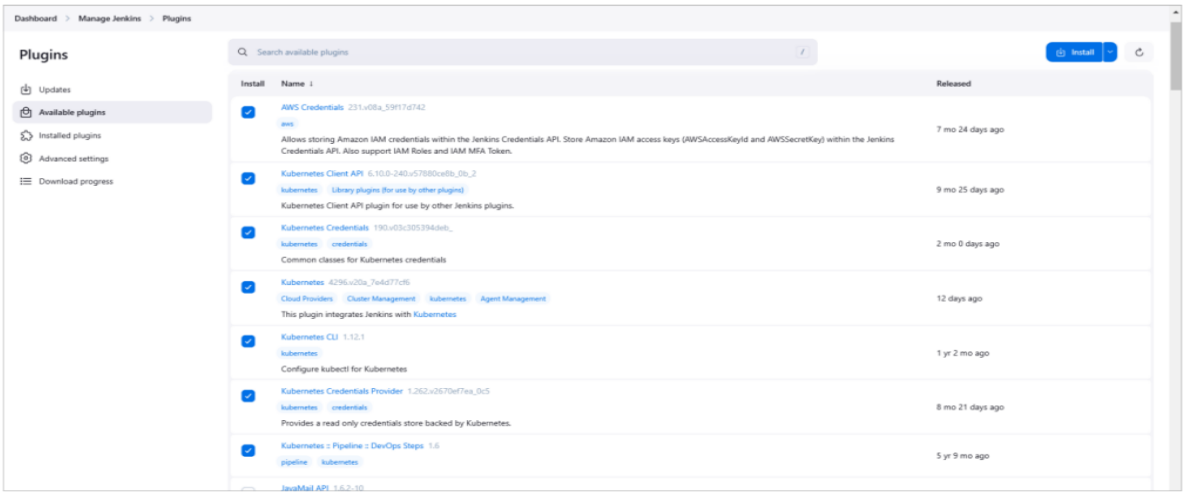
- Set up EKS Cluster
  Launch EC2 Instance to set up EKS Cluster
  Name: yashu-cluster
  AMI: Ubuntu
  Instance type: t2.medium
  Select key pair, Security Group and Subnet, Config storage =22
  Click on "Launch instance"

- Connect to putty server
  Install Kubectl, AWS CLI & EKSCTL
  Provide AWS access key and secret key

- Next, we need to copy Kube config file into the Jenkins server from Cluster
  Provide AWS access keys on Jenkins server
  Execute following command on Jenkins server
  >> aws eks update-kubeconfig --name yashu-cluster --region us-west-1

- Kube-config file has copied to Jenkins serve from master node i.e. cluster.
  Now execute following commands in Jenkins server one by one
  sudo mkdir -p /home/jenkins/.kube
  sudo cp /root/.kube/config /home/jenkins/.kube/config
  sudo chown -R jenkins:jenkins /home/jenkins/.kube
  sudo su - jenkins
  aws configure (provide access keys on Jenkins server)

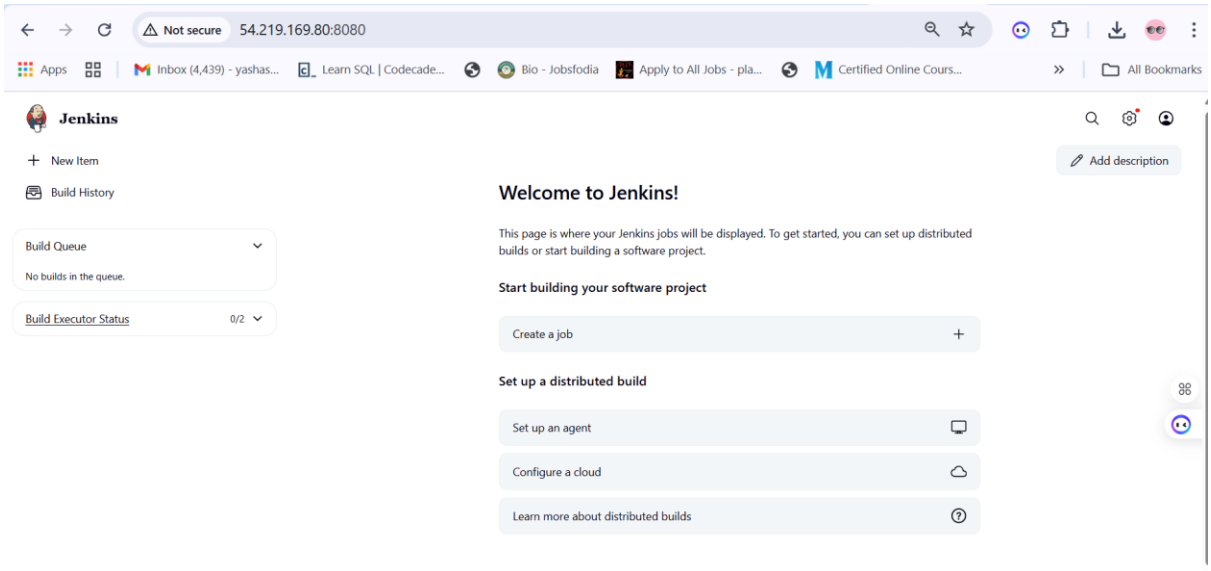
- Install following plugins in Jenkins



- Now we have to provide Kube-config file in Jenkins web server
- Firstly, we need to copy and save it in a document or file in your local
- Provide AWS access keys and kubeconfig file to credentials in Jenkins

2. Pipeline Creation

- Click on “New item” for creating pipeline
- Provide a name for pipeline and select pipeline then click on “ok”



- Next write pipeline script by adding all necessary stages

Book-My-Show > pipeline_script

```groovy
pipeline {
    agent any

    tools {
        jdk 'jdk21'
        nodejs 'node-js'
    }

    environment {
        SCANNER_HOME = tool 'sonar'
        DOCKER_IMAGE = 'yashukl2002/bms:latest'
        EKS_CLUSTER_NAME = 'yashu-cluster'
        AWS_REGION = 'us-west-1'
    }

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

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

        stage('SonarQube Analysis') {
            steps {
```

Book-My-Show > pipeline_script

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

        stage('Quality Gate') {
            steps {
                script {
                    // Added timeout to avoid infinite waiting
                    timeout(time: 1, unit: 'MINUTES') {
                        def qg = waitForQualityGate abortPipeline: true, credentialsId: 'sonar-token'
                        echo "Quality Gate status: ${qg.status}"
                    }
                }
            }
        }

```

Book-My-Show > pipeline_script

```
pipeline {
    stages {
        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', oc
                dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
            }
        }

```

Book-My-Show > pipeline_script

```
pipeline {
    stages {
        stage('Trivy FS Scan') {
    steps {
        sh '''
        if ! command -v trivy &> /dev/null
        then
            echo "Trivy not installed! Skipping scan."
            exit 0
        fi

        echo "Running Trivy FS scan..."
        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-

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

Book-My-Show > pipeline_script

```
90  }
107         stage('Deploy to EKS Cluster') {
108             steps {
109                 script {
110                     sh '''
111                     echo "Verifying AWS credentials..."
112                     aws sts get-caller-identity
113 
114                     echo "Configuring kubectl for EKS cluster..."
115                     aws eks update-kubeconfig --name $EKS_CLUSTER_NAME --region $AWS_REGION
116 
117                     echo "Verifying kubeconfig..."
118                     kubectl config view
119 
120                     echo "Deploying application to EKS..."
121                     kubectl apply -f deployment.yml
122                     kubectl apply -f service.yml
123 
124                     echo "Verifying deployment..."
125                     kubectl get pods
126                     kubectl get svc
127                     '''
128                 }
129             }
130         }
131     }
132
```

! deployment.yml 1, M | ! service.yml M | ! prometheus.yml U | Jenkinsfile2 | pipeline_script U

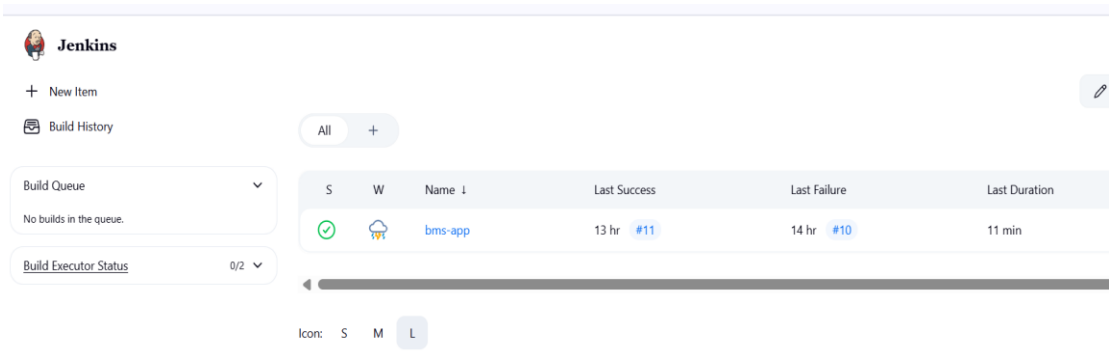
Book-My-Show > pipeline_script

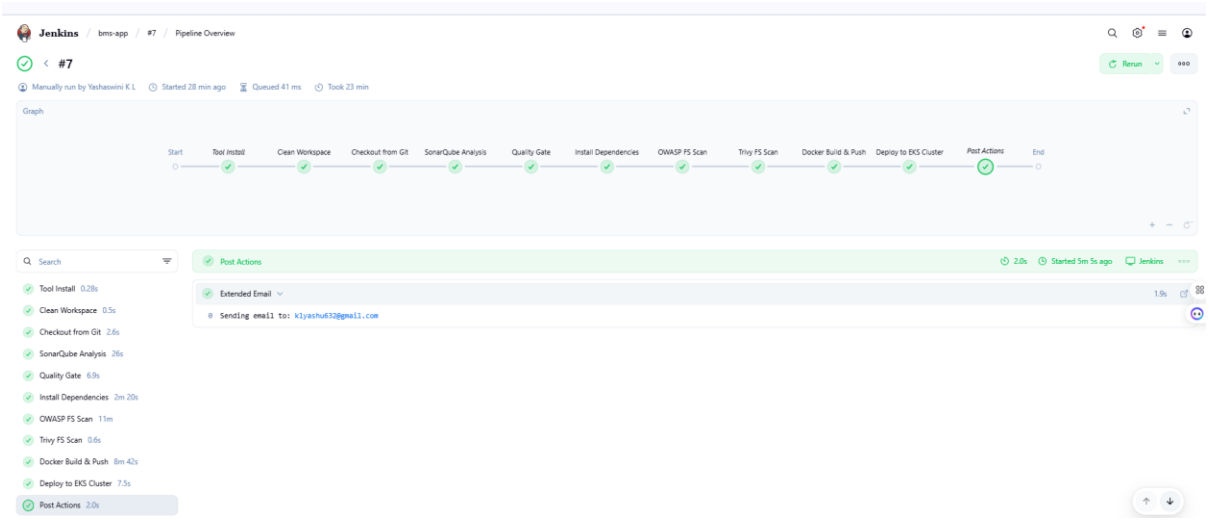
```
90  }
132 
133     post {
134         always {
135             emailext attachLog: true,
136                 subject: "'${currentBuild.result}'",
137                 body: "Project: ${env.JOB_NAME}<br/>" +
138                       "Build Number: ${env.BUILD_NUMBER}<br/>" +
139                       "URL: ${env.BUILD_URL}<br/>",
140                 to: 'klyashu632@gmail.com',
141                 attachmentsPattern: 'trivyfs.txt'
142         }
143     }
144 }
145
```

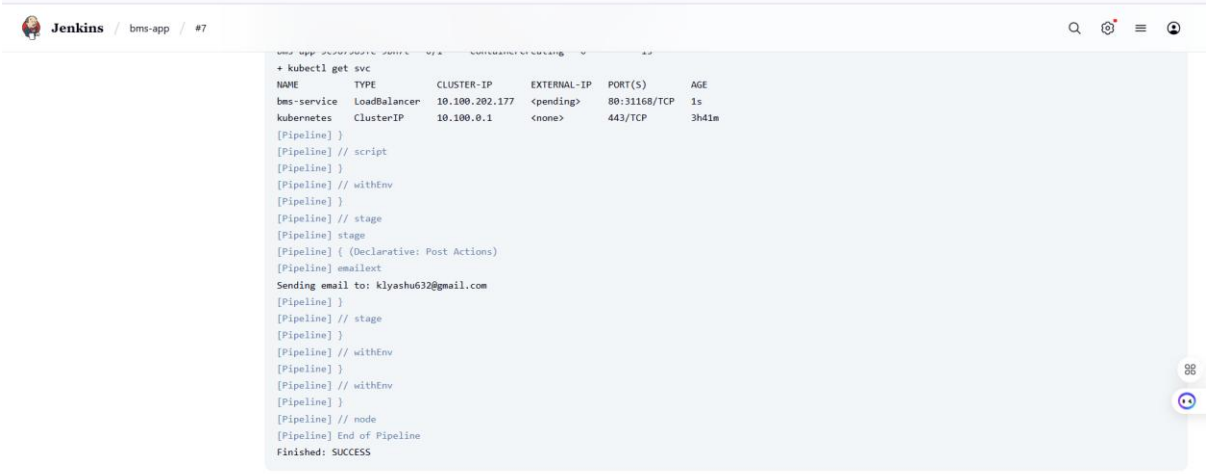
➢ After saving your script come back to your pipeline and click on “Build Now” to running your pipeline.

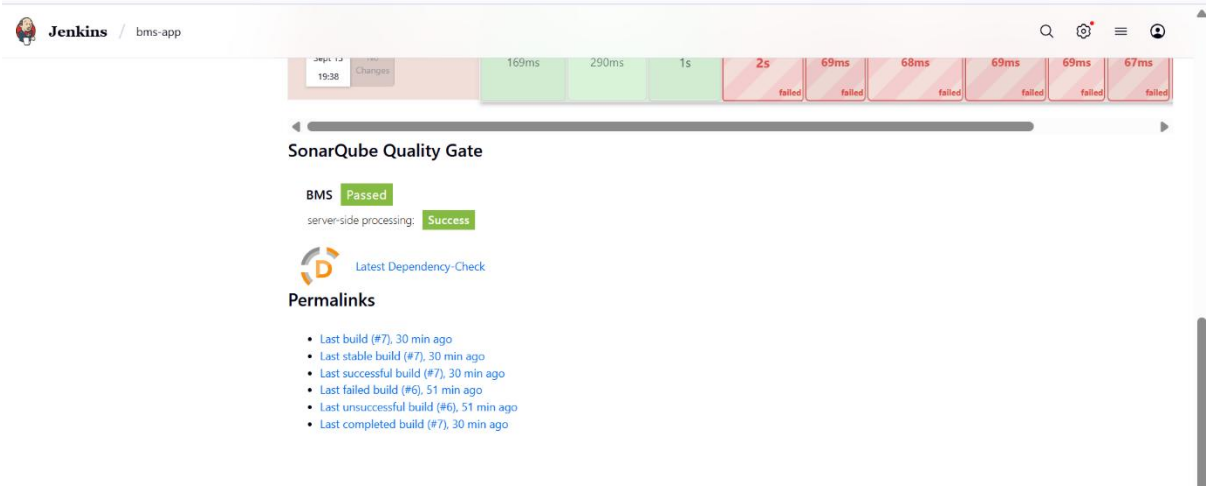
Outputs:

Jenkins UI

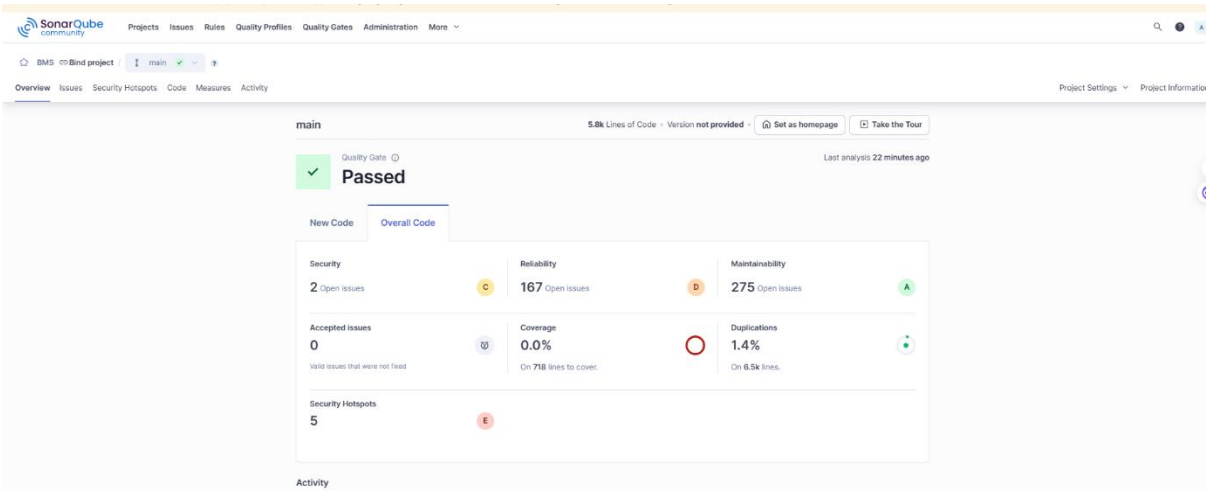


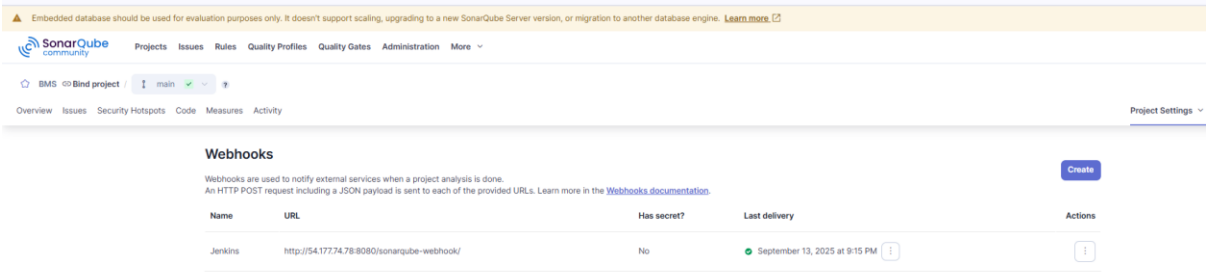

Jenkins / bms-app / #7 / Pipeline Overview
#7
Rerun
Manually run by Yashaswini K L
Started 28 min ago
Queued 41 ms
Took 23 min
Graph
Start
Tool Install
Clean Workspace
Checkout from Git
SonarQube Analysis
Quality Gate
Install Dependencies
OWASP FS Scan
Trivy FS Scan
Docker Build & Push
Deploy to EKS Cluster
Post Actions
End
Search
Post Actions
2.0s
Started 5m 5s ago
Jenkins
Tool Install 0.28s
Clean Workspace 0.5s
Checkout from Git 2.6s
SonarQube Analysis 28s
Quality Gate 6.9s
Install Dependencies 2m 20s
OWASP FS Scan 11m
Trivy FS Scan 0.6s
Docker Build & Push 8m 42s
Deploy to EKS Cluster 7.5s
Post Actions 2.0s
Extended Email
1.9s
Sending email to: klyashu632@gmail.com


Jenkins / bms-app / #7
+ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
bms-service LoadBalancer 10.100.202.177 <pending> 80:31168/TCP 1s
kubernetes ClusterIP 10.100.0.1 <none> 443/TCP 3h41m
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] emailext
Sending email to: klyashu632@gmail.com
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

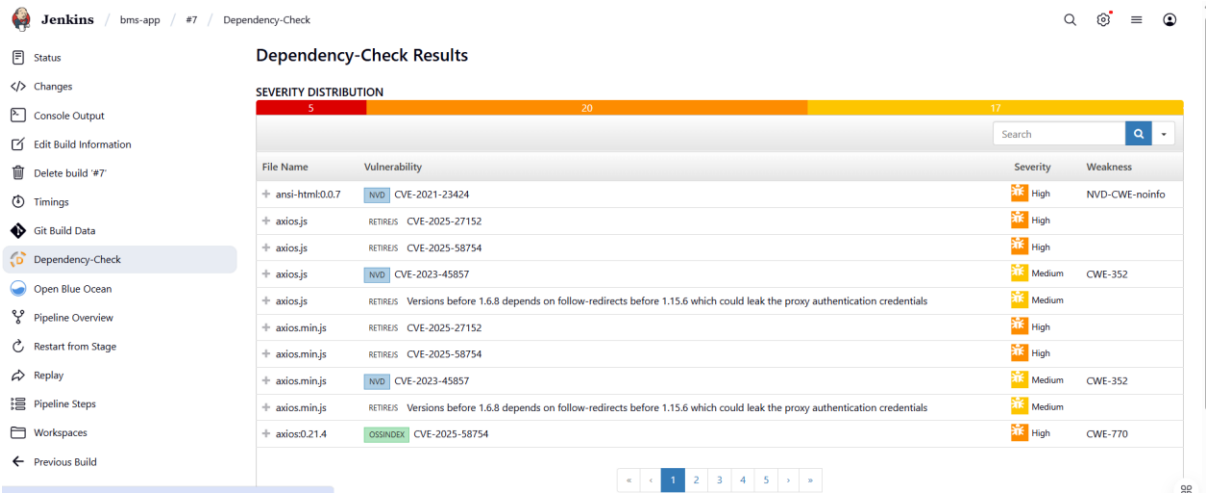

Jenkins / bms-app
19:38
Changes
169ms
290ms
1s
2s
failed
69ms
failed
68ms
failed
69ms
failed
69ms
failed
67ms
failed
SonarQube Quality Gate
BMS
Passed
server-side processing:
Success
Latest Dependency-Check
Permalinks
Last build (#7), 30 min ago
Last stable build (#7), 30 min ago
Last successful build (#7), 30 min ago
Last failed build (#6), 51 min ago
Last unsuccessful build (#6), 51 min ago
Last completed build (#7), 30 min ago

SonarQube Analysis





Dependency Check



Docker Build & push

Jenkins / bms-app / #7

```
 ---> e7bde9b36dae
Step 7/10 : EXPOSE 3000
 ---> Running in f87a332cc99b
 ---> Removed intermediate container f87a332cc99b
 ---> a69323140d43
Step 8/10 : ENV NODE_OPTIONS=--openssl-legacy-provider
 ---> Running in 887de853dd28
 ---> Removed intermediate container 887de853dd28
 ---> 9dc67353c454
Step 9/10 : ENV PORT=3000
 ---> Running in 584f5b9d8bd2
 ---> Removed intermediate container 584f5b9d8bd2
 ---> f477370fd80e
Step 10/10 : CMD ["npm", "start"]
 ---> Running in 5df1f8b08dd7
 ---> Removed intermediate container 5df1f8b08dd7
 ---> a5a132475186
Successfully built a5a132475186
Successfully tagged yashukl2002/bms:latest
+ echo Pushing Docker image to Docker Hub...
Pushing Docker image to Docker Hub...
+ docker push yashukl2002/bms:latest
The push refers to repository [docker.io/yashukl2002/bms]
8dd2f2b46566: Preparing
33e332e4067d: Preparing
3234ac9a82dc: Preparing
5c62565b78f4: Preparing
```

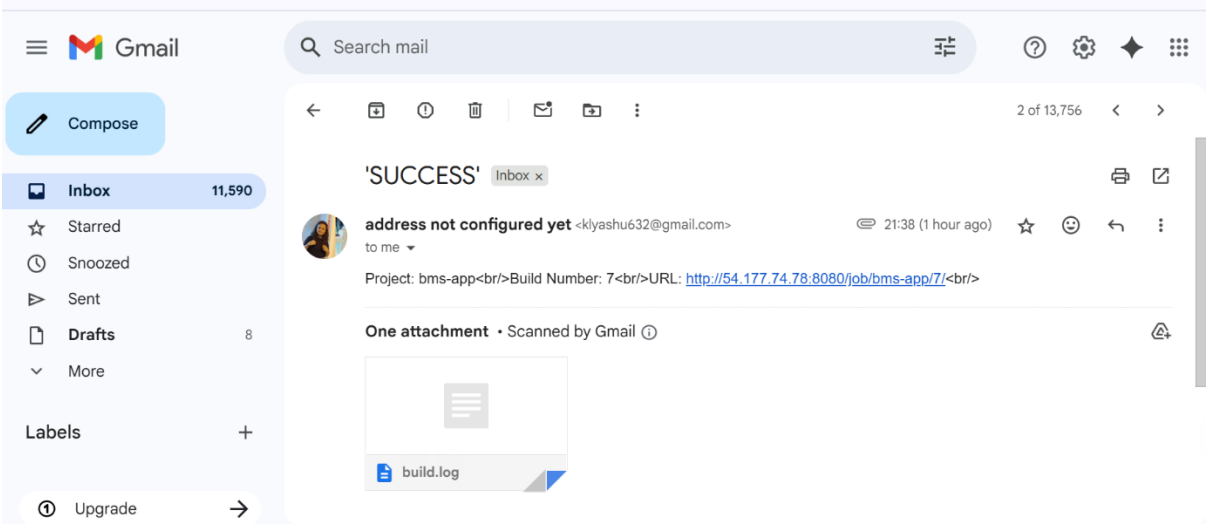
EKS Deployment

Jenkins / bms-app / #7

```
      - --cluster-name
      - yashu-cluster
      - --output
      - json
      command: aws
      env: null
+ echo Deploying application to EKS...
Deploying application to EKS...
+ 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-5c987585fc-2m8gz     0/1     ContainerCreating   0          1s
bms-app-5c987585fc-9bh7t     0/1     ContainerCreating   0          1s
+ kubectl get svc
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
bms-service   LoadBalancer   10.100.202.177   <pending>     80:31168/TCP   1s
kubernetes    ClusterIP      10.100.0.1       <none>        443/TCP        3h41m
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
```

```
root@ip-172-31-22-136:/home/ubuntu# kubectl get pods
NAME                         READY   STATUS    RESTARTS   AGE
bms-app-5c987585fc-2m8gz     1/1     Running   0          16m
bms-app-5c987585fc-9bh7t     1/1     Running   0          16m
root@ip-172-31-22-136:/home/ubuntu# kubectl get svc
NAME          TYPE           CLUSTER-IP       EXTERNAL-IP
        PORT(S)        AGE
bms-service   LoadBalancer   10.100.202.177   ae95b4efd3970497d8afb06f97ca6100-361306071.us-west-1.elb.amazonaw
s.com   80:31168/TCP   16m
kubernetes    ClusterIP      10.100.0.1       <none>
        443/TCP        3h57m
```

Email notification



Step 4: Docker Deployment

- Docker file

deployment.yml | service.yml | Jenkinsfile2 | Dockerfile M

Book-My-Show-capstone > bookmyshow-app > Dockerfile > ...

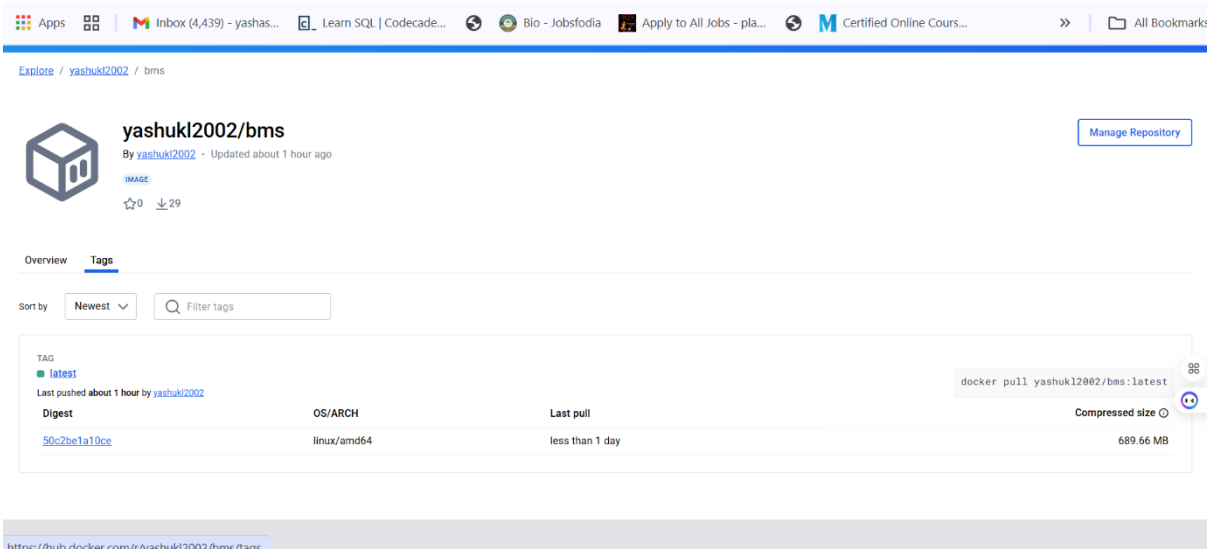
```
3
4   WORKDIR /app
5
6   COPY package.json package-lock.json ./
7
8   RUN npm install postcss@8.4.21 postcss-safe-parser@6.0.0 --legacy-peer-deps
9
10  RUN npm install
11
12  COPY . .
13
14  EXPOSE 3000
15
16  ENV NODE_OPTIONS=--openssl-legacy-provider
17  ENV PORT=3000
18
19  CMD ["npm", "start"]
20
```

- Build Docker image and push to DockerHub via Jenkins.

Jenkins / bms-app / #7

```
 ---> e7bde9b36dae
Step 7/10 : EXPOSE 3000
 ---> Running in f87a332cc99b
 ---> Removed intermediate container f87a332cc99b
 ---> a69323140d43
Step 8/10 : ENV NODE_OPTIONS=--openssl-legacy-provider
 ---> Running in 887de853dd28
 ---> Removed intermediate container 887de853dd28
 ---> 9dc67353c454
Step 9/10 : ENV PORT=3000
 ---> Running in 584f5b9d8bd2
 ---> Removed intermediate container 584f5b9d8bd2
 ---> f477370fd80e
Step 10/10 : CMD ["npm", "start"]
 ---> Running in 5df1f8b08dd7
 ---> Removed intermediate container 5df1f8b08dd7
 ---> a5a132475186
Successfully built a5a132475186
Successfully tagged yashukl2002/bms:latest
+ echo Pushing Docker image to Docker Hub...
Pushing Docker image to Docker Hub...
+ docker push yashukl2002/bms:latest
The push refers to repository [docker.io/yashukl2002/bms]
8dd2f2b46566: Preparing
33e332e4067d: Preparing
3234ac9a82dc: Preparing
5c62565b78f4: Preparing
```

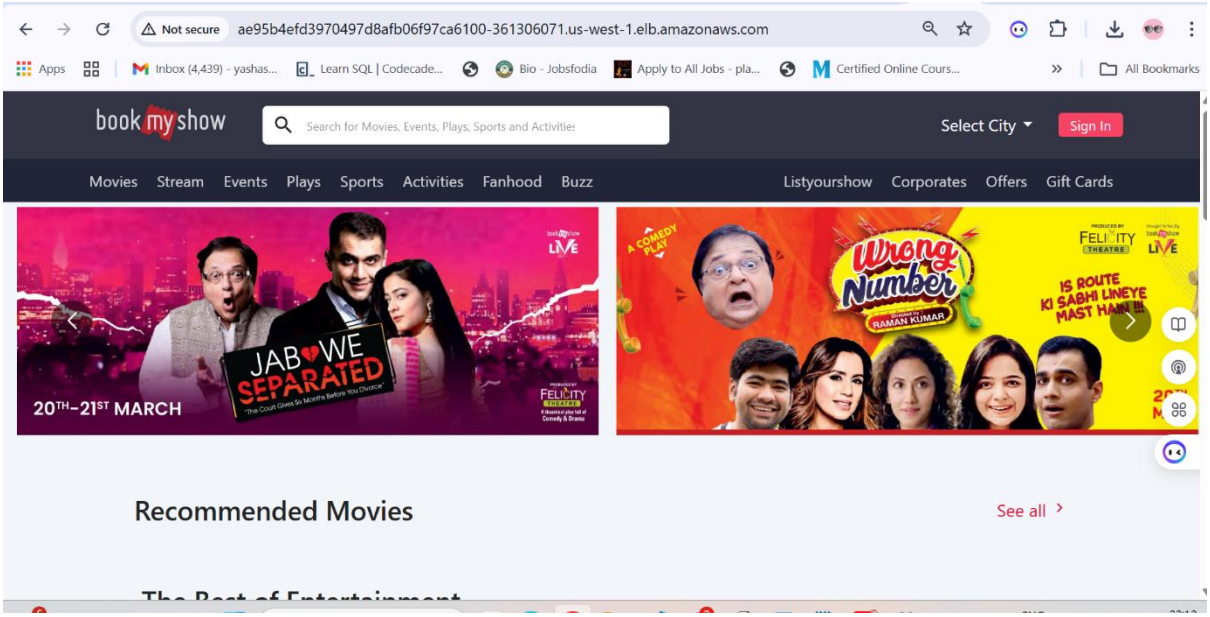
- Image pushed to dockerhub



- Dockerhub repository link

https://hub.docker.com/repository/docker/yashukl2002/bms

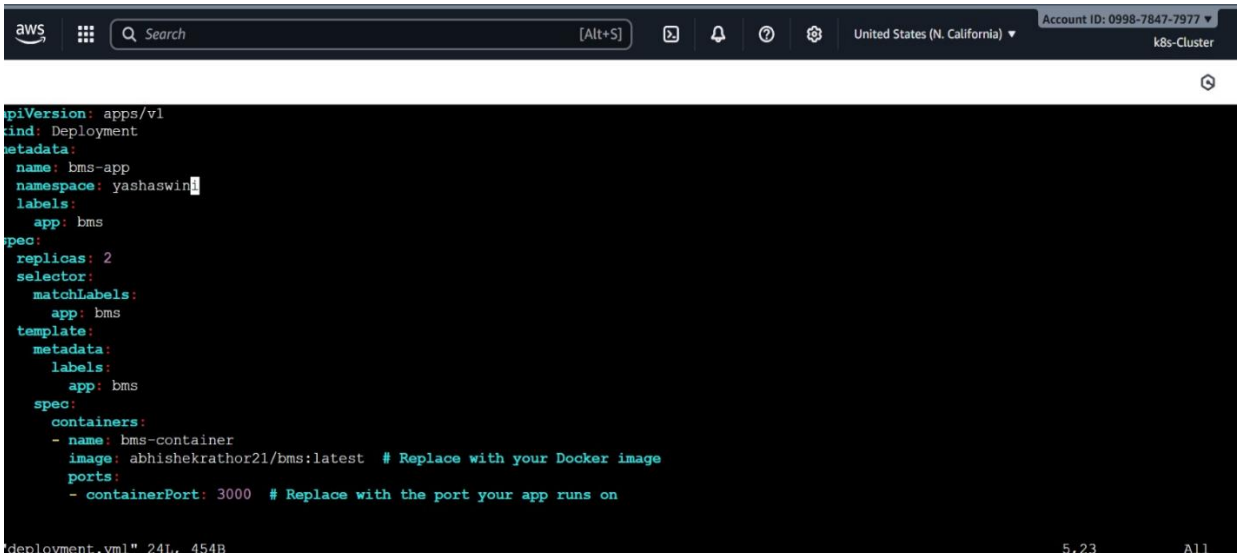
- Run container locally and validate accessibility



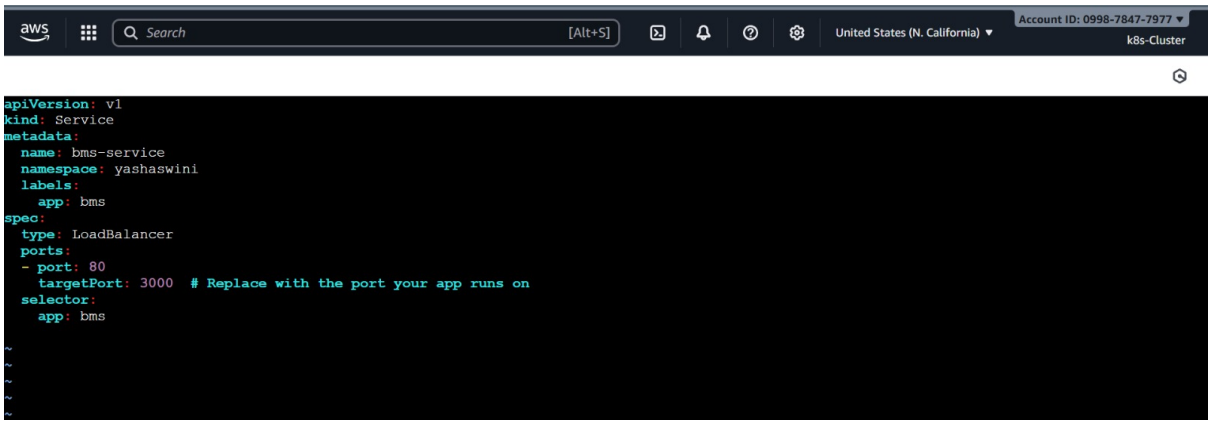
Step 5: Kubernetes Deployment EKS

Manifest Files:

- deployment.yml



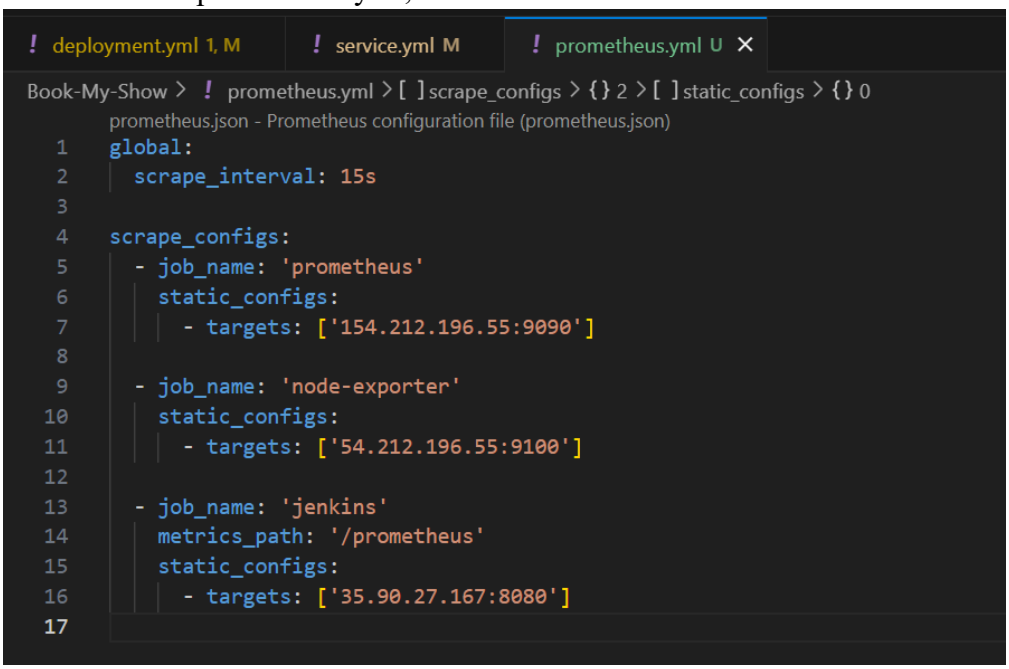
- service.yml



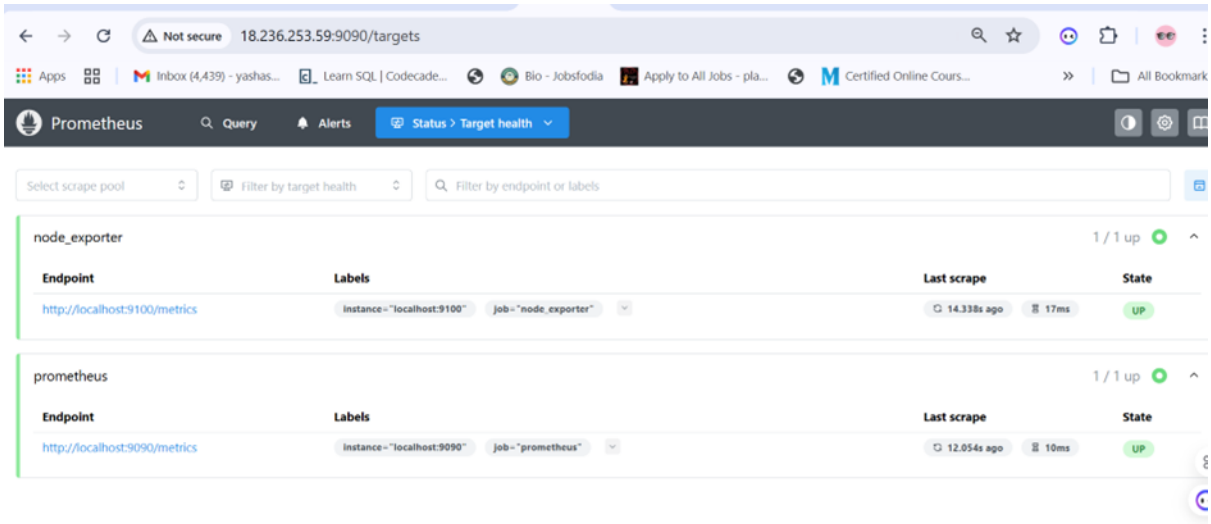

```
root@ip-172-31-22-61:/home/ubuntu# kubectl create namespace yashaswini
namespace/yashaswini created
root@ip-172-31-22-61:/home/ubuntu# vi deployment.yml
root@ip-172-31-22-61:/home/ubuntu# vi service.yml
root@ip-172-31-22-61:/home/ubuntu# vi deployment.yml
root@ip-172-31-22-61:/home/ubuntu# kubectl apply -f deployment.yml
deployment.apps/bms-app created
root@ip-172-31-22-61:/home/ubuntu# kubectl apply -f service.yml
service/bms-service created
root@ip-172-31-22-61:/home/ubuntu# kubectl get pods -n yashaswini
NAME                      READY   STATUS    RESTARTS   AGE
bms-app-67c4b84589-cn8zn  1/1     Running   0          45s
bms-app-67c4b84589-mb9hh  1/1     Running   0          45s
root@ip-172-31-22-61:/home/ubuntu# kubectl get svc -n yashaswini
NAME          TYPE           CLUSTER-IP     EXTERNAL-IP                                                              PORT(S)        AGE
bms-service   LoadBalancer   10.100.1.240   a3d4a25dc77f746cfa3c8a379e61899f-1845912715.us-west-1.elb.amazonaws.com   80:30987/TCP   51s
```

Step 6: Monitoring & Observability

- Launch an AWS EC2 Instance
- Install Prometheus, Grafana and Node Exporter on the server
- Add Jenkins Performance Dashboard
  Install Jenkins Prometheus plugin on Jenkins server (if not already installed):
  Go to Manage Jenkins → Plugins → Available → Search "Prometheus metrics" → Install
  Enable metrics endpoint:
  Go to Manage Jenkins → Configure System
  Scroll to Prometheus section → Check Enable Prometheus metrics
  Save
  In Prometheus prometheus.yml, add:

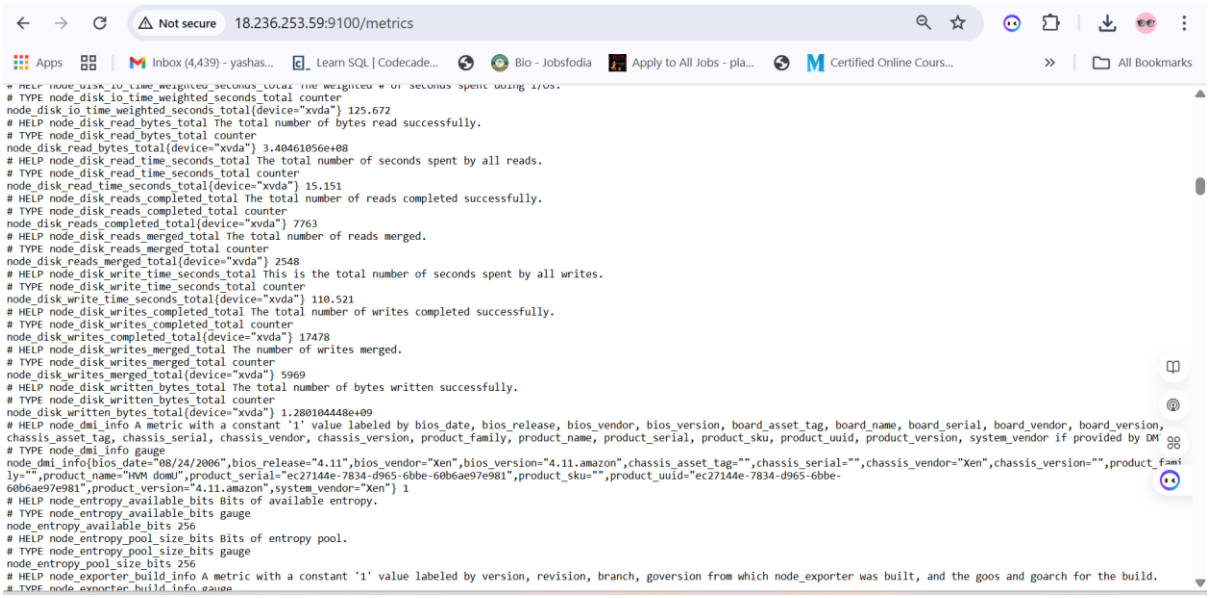


- Configure Grafana dashboards to visualize node health (CPU, memory, disk usage) and Jenkins performance metrics using Prometheus as a data source
- Open Prometheus UI:
  http://18.236.253.59:9090

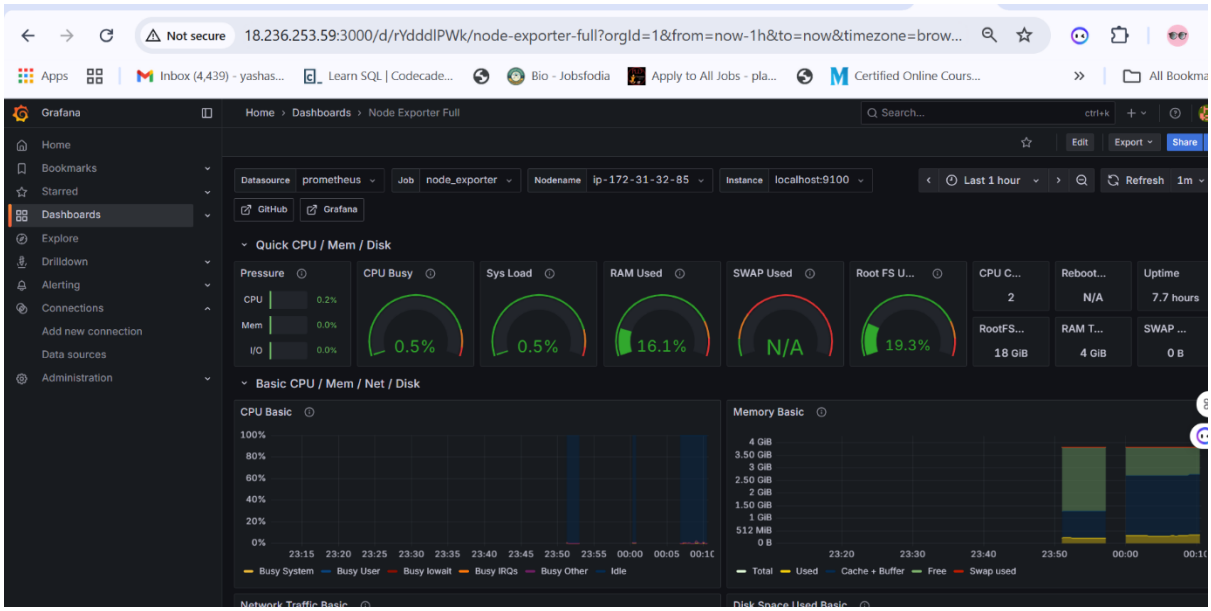


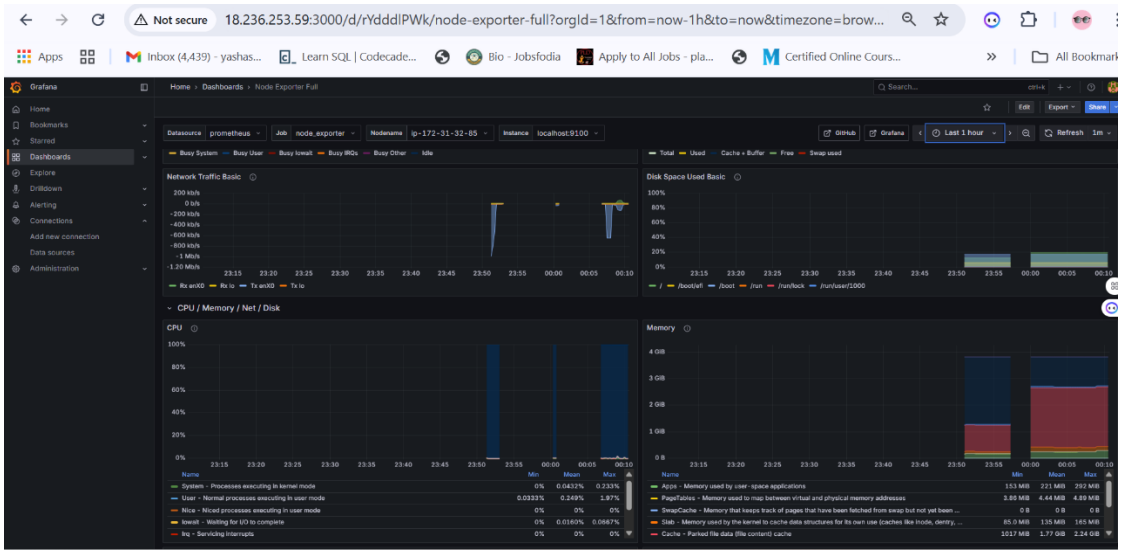
- **Access Grafana UI → http://<server-ip>:3000**
  Default Login: admin / admin → Change password after first login.
  Configure Prometheus as a Grafana Data Source
  In Grafana:
  Go to Connections → Data Sources → Add data source
  Select Prometheus
  URL → http://<prometheus-server>:9090
  Click Save & Test

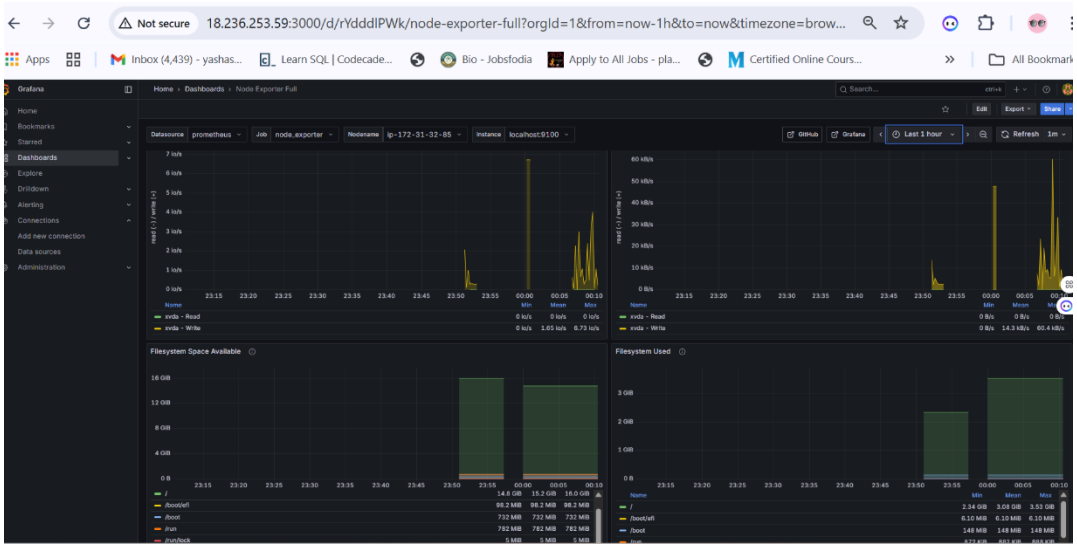
- **Node Exporter metrics**



- **Add Dashboards**
  Node Health Dashboard:
  Import from Grafana's dashboard library.
  Recommended dashboard ID: 1860 (Node Exporter Full)
  Grafana.com Dashboard


Not secure
18.236.253.59:3000/d/rYdddlPWk/node-exporter-full?orgId=1&from=now-1h&to=now&timezone=brow...
Grafana
Home › Dashboards › Node Exporter Full
Quick CPU / Mem / Disk
Basic CPU / Mem / Net / Disk
CPU Basic
Memory Basic


Not secure
18.236.253.59:3000/d/rYdddlPWk/node-exporter-full?orgId=1&from=now-1h&to=now&timezone=brow...
Network Traffic Basic
Disk Space Used Basic
CPU / Memory / Net / Disk
CPU
Memory


Not secure
18.236.253.59:3000/d/rYdddlPWk/node-exporter-full?orgId=1&from=now-1h&to=now&timezone=brow...
Filesystem Space Available
Filesystem Used

Jenkins Performance Dashboard:

Import dashboard ID: 9964 (Jenkins Prometheus Exporter Dashboard)

Click Import