

## java-cicd-maven-jenkins-argocd-k8s Report:

## 1. Jenkins Pipeline Execution

Dashboard > Spring-Boot-CI-CD > #17

- Status
- Changes
- Console Output**
- Edit Build Information
- Timings
- Git Build Data
- Pipeline Overview
- Pipeline Console
- Thread Dump
- Pause/resume
- Download

## Console Output

Download

Copy

View as plain text

```
Started by user abhijot
Obtained spring-boot-app/Jenkinsfile from git https://github.com/abkaur/Ci-cd-with-jenkins-and-argocd.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Spring-Boot-CI-CD
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Spring-Boot-CI-CD/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/abkaur/Ci-cd-with-jenkins-and-argocd.git # timeout=10
```

```
1 file changed, 1 insertion(+), 1 deletion(-)
+ git push https://****@github.com/abkaur/Ci-cd-with-jenkins-and-argocd HEAD:main
To https://github.com/abkaur/Ci-cd-with-jenkins-and-argocd
e35b95c..f56b9c9 HEAD -> main
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Dashboard > Spring-Boot-CI-CD > #17

### Steps Taken:

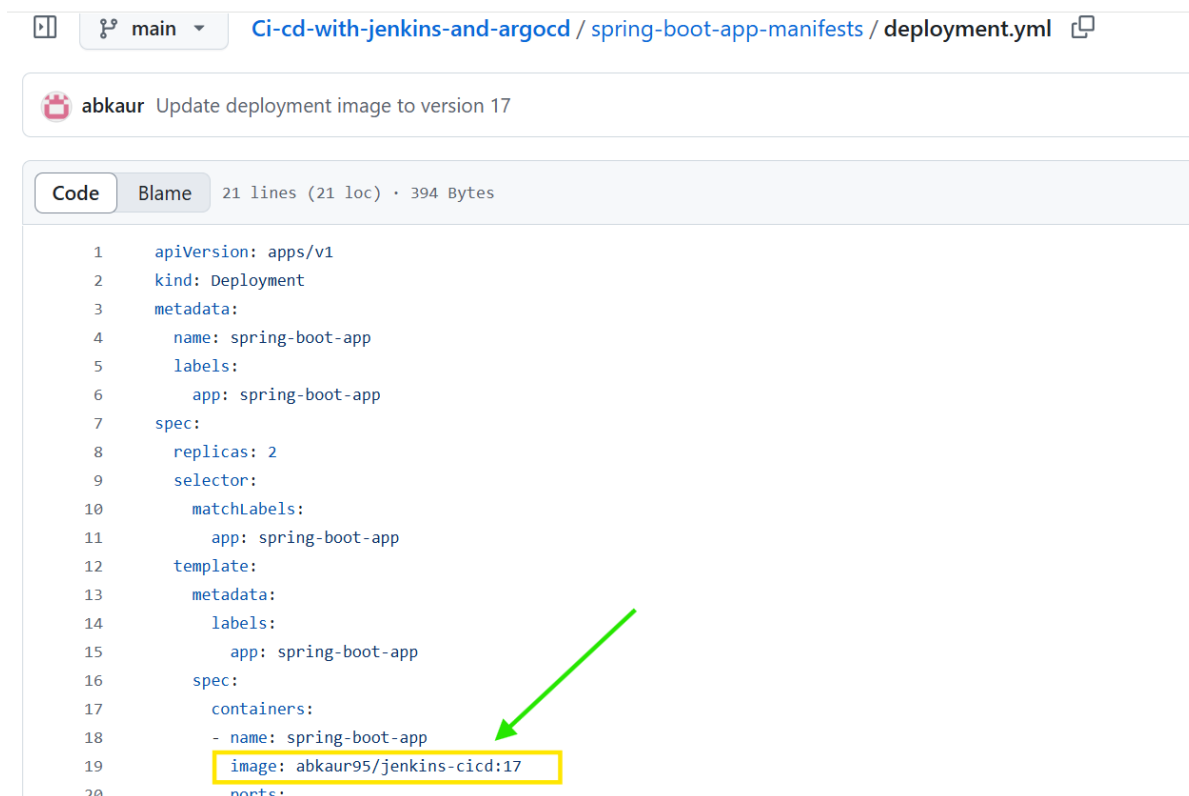
- Created a Jenkins Pipeline to build and push a Docker image.
- Used Maven to package the Spring Boot application.
- Used Docker to build and push the image to Docker Hub.
- Updated the Kubernetes manifest file (deployment.yml) dynamically in GitHub.
- Triggered ArgoCD to deploy the latest version automatically.

### Problems Faced:

- Git Credentials Issue: Missing credentials prevented the commit of deployment.yml.
- Maven Not Found: The pipeline failed because Maven was not installed on the Jenkins instance.
- Docker Plugin Issue: Had to install and configure the Docker Pipeline Plugin in Jenkins.
- Docker Push Failure: Incorrect credentials type in the script led to an authentication failure while pushing the Docker image.

### Improvements:

- Can implement a Jenkins Agent with pre-installed Maven and Docker to avoid installation issues.
- Can use GitHub Actions instead of Jenkins to manage the CI process more efficiently.



The screenshot shows a GitHub repository interface. At the top, the repository name is 'Ci-cd-with-jenkins-and-argocd' and the file path is 'spring-boot-app-manifests / deployment.yml'. Below the repository name, there is a commit message: 'abkaur Update deployment image to version 17'. The file is shown in a code editor with tabs for 'Code' and 'Blame'. The file content is a Kubernetes Deployment manifest for 'spring-boot-app'. A green arrow points to the 'image: abkaur95/jenkins-cicd:17' line, which is highlighted with a yellow box.

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: spring-boot-app
5    labels:
6      app: spring-boot-app
7  spec:
8    replicas: 2
9    selector:
10     matchLabels:
11       app: spring-boot-app
12    template:
13     metadata:
14       labels:
15         app: spring-boot-app
16     spec:
17       containers:
18         - name: spring-boot-app
19           image: abkaur95/jenkins-cicd:17
20       ports:
```

The screenshot shows a GitHub repository page for 'Ci-cd-with-jenkins-and-argocd', which is a public fork of 'DevOpsGodd/Ci-cd-with-jenkins-and-argocd'. The repository has 1 branch (main) and 0 tags. A status bar indicates the current branch is 13 commits ahead of the upstream. Below this, a commit history table is displayed:

Commit Message	Author	Time
Update deployment image to version 17	abkaur	6 minutes ago
Update deployment image to version 17		6 minutes ago
Update Jenkinsfile		6 minutes ago
new commit		last year

## 2. ArgoCD Deployment & Sync

### Steps Taken:

- Connected ArgoCD to the GitHub repository where Kubernetes manifests are stored.
- Used the ArgoCD UI to monitor deployments.
- Verified that ArgoCD automatically updates the cluster with the latest changes in deployment.yml.

### Problems Faced:

- Incorrect Repository Path: The incorrect path in ArgoCD prevented it from syncing deployments.

### Improvements:

- Can automate ArgoCD sync using webhooks to detect repository changes.
- Can expose ArgoCD with an Ingress Controller instead of a LoadBalancer.
- In future iterations, Helm charts can be integrated so that Argo CD tracks chart versions instead of individual manifest files, improving scalability and modularity.

spring-boot-app - Application | x +

Not secure https://34.58.21.7/applications/argocd/spring-boot-app?view=tree&resource=

Future Members an... Shopping cart | Mar... Job Description for... Computer Basics: W... Careers at Marsh M... Ch.01\_Introduction\_... All Bookmarks

argo v2.14.2+ad7246

Applications  
Settings  
User Info  
Documentation

Resource filters  
NAME  
NAME  
KINDS  
KINDS  
SYNC STATUS

APPLICATION DETAILS TREE

DETAILS DIFF SYNC SYNC STATUS HISTORY AND ROLLBACK DELETE REFRESH

APP HEALTH Healthy

SYNC STATUS Synced to main (f56b9c9)

LAST SYNC Sync OK to f56b9c9

Auto sync is enabled.  
Author: abkaur <kaurab@sheridancollege.ca> -  
Comment: Update deployment image to version 17

Succeeded 3 minutes ago (Sat Feb 15 2025 18:56:01 GMT-0500)  
Author: abkaur <kaurab@sheridancollege.ca> -  
Comment: Update deployment image to version 17

spring-boot-app 100% 3 minutes

spring-boot-app-service 3 minutes

spring-boot-app-service-5257w 3 minutes

spring-boot-app 3 minutes rev:1

spring-boot-app-68dd6c9d69 3 minutes rev:1

CREATED AT 02/15/2025 18:55:55 (6 minutes ago)

REPO URL <https://github.com/abkaur/Ci-cd-with-jenkins-and-argocd.git>

TARGET REVISION main

PATH spring-boot-app-manifests

SYNC OPTIONS

RETRY OPTIONS Retry disabled

STATUS Synced to main (f56b9c9)

HEALTH Healthy

LINKS

IMAGES abkaur95/jenkins-cicd:17

### 3. Application Deployment & Exposure

Steps Taken:

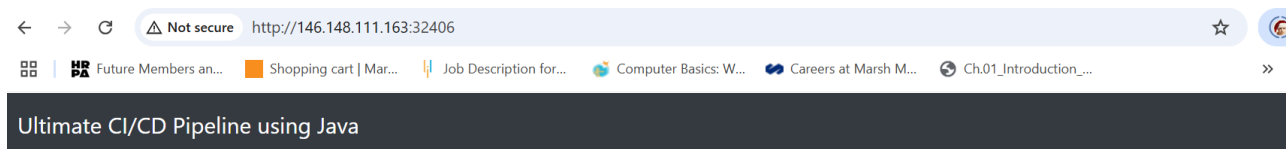
- Deployed the application using Kubernetes Deployment and Service.
- Exposed the application using a NodePort service.
- Accessed the application through `http://<external ip of clusterNode>:<NodePort>`.

Problems Faced:

- Service Not Accessible: The application service was not reachable due to incorrect firewall rules.
- Wrong External IP Used: Initially, the wrong external IP (VM instead of cluster node) was used.

Improvements:

- Can use an Ingress Controller (e.g., Nginx) to manage application routing.
- Can use LoadBalancer Service instead of NodePort for better scalability.



**I have successfully built a spring boot application using Maven**

This application is deployed on to Kubernetes using Argo CD

**Key Learnings:**

- CI/CD automation with Jenkins and ArgoCD reduces deployment efforts.

- Ensuring correct firewall rules is essential for external access.
- GitHub Webhooks can help auto-trigger deployments.

### **Future Enhancements:**

- Monitoring & Logging: Prometheus & Grafana can be used for observability.
- Kubernetes Ingress: Can replace NodePort with an Ingress Controller for cleaner routing.
- **Helm Charts:** Package Kubernetes manifests for better management.

Currently, the pipeline uses plain Kubernetes manifests to deploy the application. In the future, Helm can be introduced to simplify deployment management by templating and parameterizing Kubernetes resources. Helm charts would allow versioned, reusable deployment definitions that can handle multiple environments (dev, test, prod) with different values. This would also make Argo CD integration cleaner, as Argo CD can directly track and sync Helm chart releases from the repository.