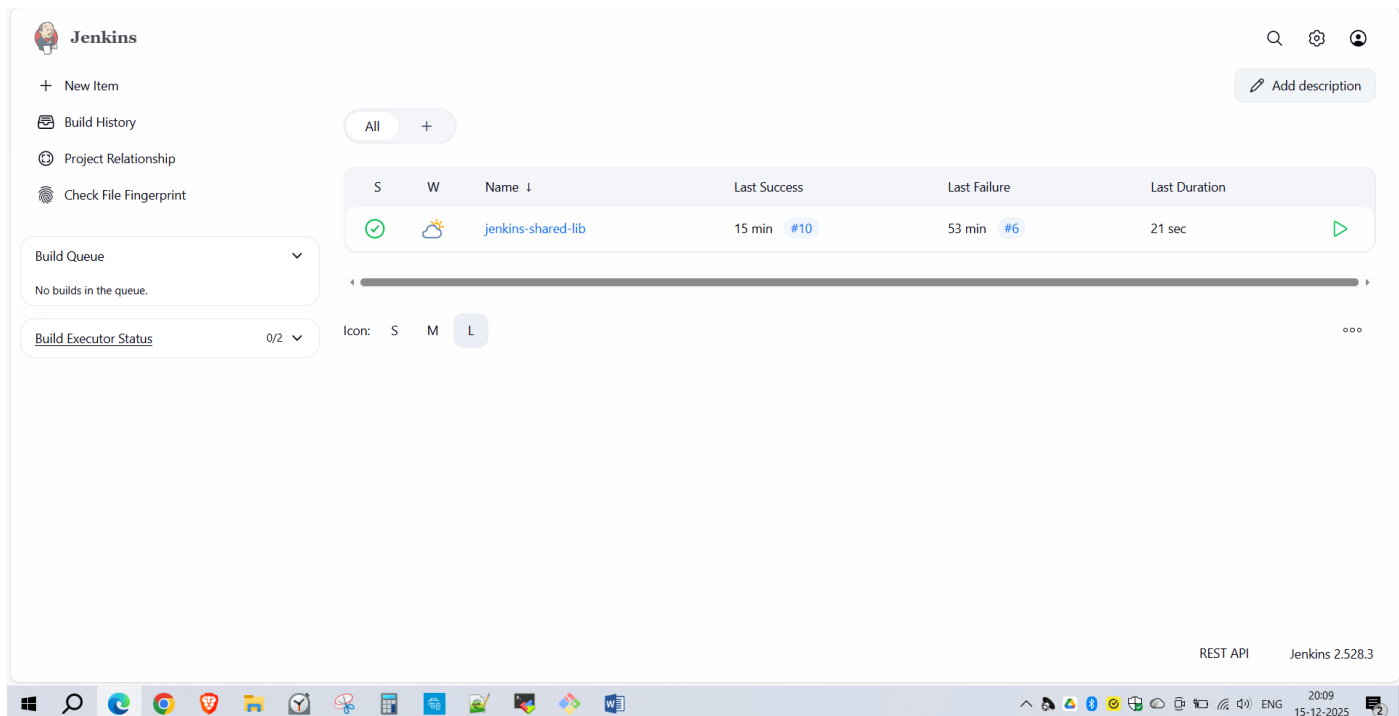


A large, light gray watermark consisting of the letters 'S' and 'K' is positioned in the background, spanning most of the width and height of the page.

Jenkins Shared Libraries

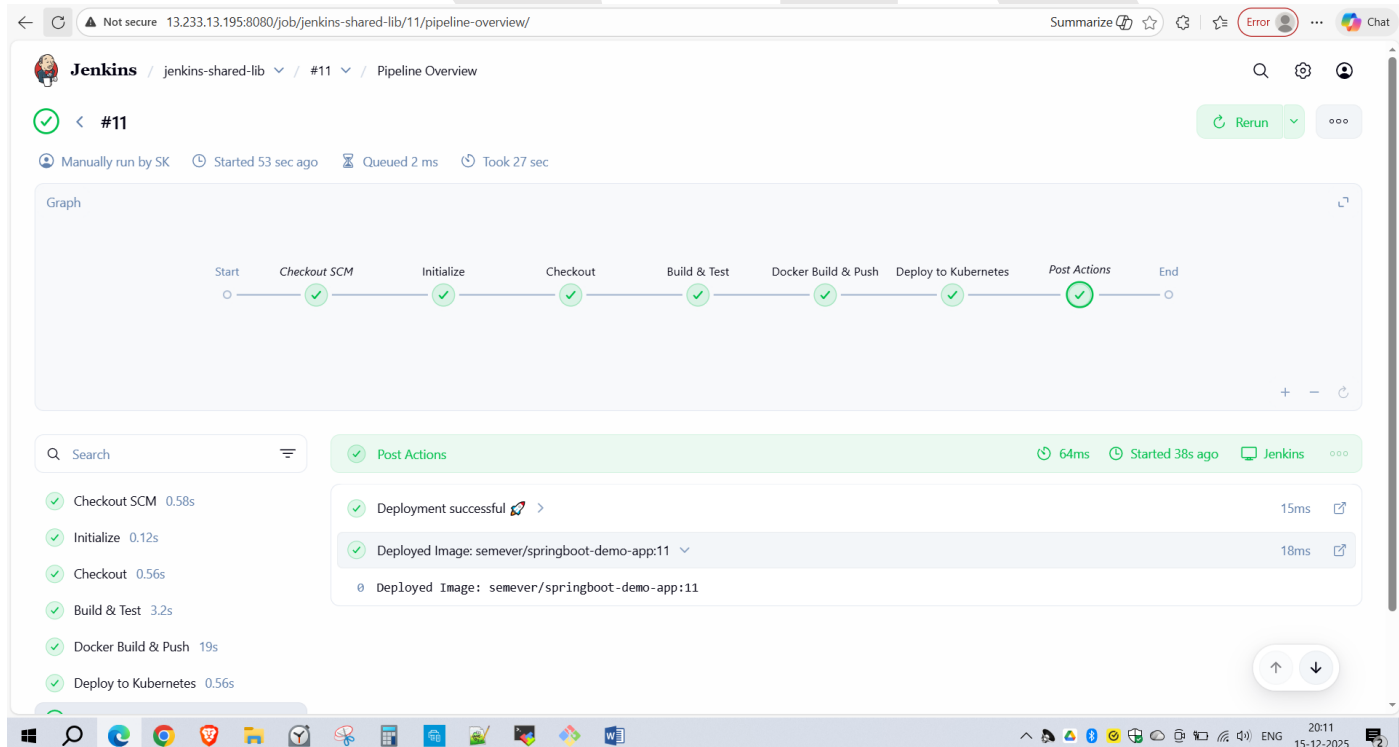
Jenkins Pipeline Job Using Shared Libraries:



The screenshot shows the Jenkins dashboard. On the left, there's a sidebar with 'New Item', 'Build History', 'Project Relationship', and 'Check File Fingerprint'. The main area displays a table of jobs. The 'jenkins-shared-lib' job is highlighted, showing its last success at 15 min and last failure at 53 min. Below the table, there's a 'Build Queue' section indicating no builds in the queue, and a 'Build Executor Status' section showing 0/2 executors. The bottom of the dashboard includes a 'REST API' link and the version 'Jenkins 2.528.3'.

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☁	jenkins-shared-lib	15 min #10	53 min #6	21 sec

Standardized Pipeline Job Execution Across Stages Using Centralized Shared Libraries:



The screenshot shows the Jenkins Pipeline Overview for the 'jenkins-shared-lib' job, build #11. The pipeline graph consists of the following stages: Start, Checkout SCM, Initialize, Checkout, Build & Test, Docker Build & Push, Deploy to Kubernetes, Post Actions, and End. All stages are marked as successful with green checkmarks. Below the graph, a search bar is present, and a list of stages with their durations is shown: Checkout SCM (0.58s), Initialize (0.12s), Checkout (0.56s), Build & Test (3.2s), Docker Build & Push (19s), and Deploy to Kubernetes (0.56s). The 'Post Actions' section is expanded, showing 'Deployment successful' (15ms) and 'Deployed Image: semever/springboot-demo-app:11' (18ms). The bottom of the dashboard includes a 'REST API' link and the version 'Jenkins 2.528.3'.

Graph

Start → Checkout SCM → Initialize → Checkout → Build & Test → Docker Build & Push → Deploy to Kubernetes → Post Actions → End

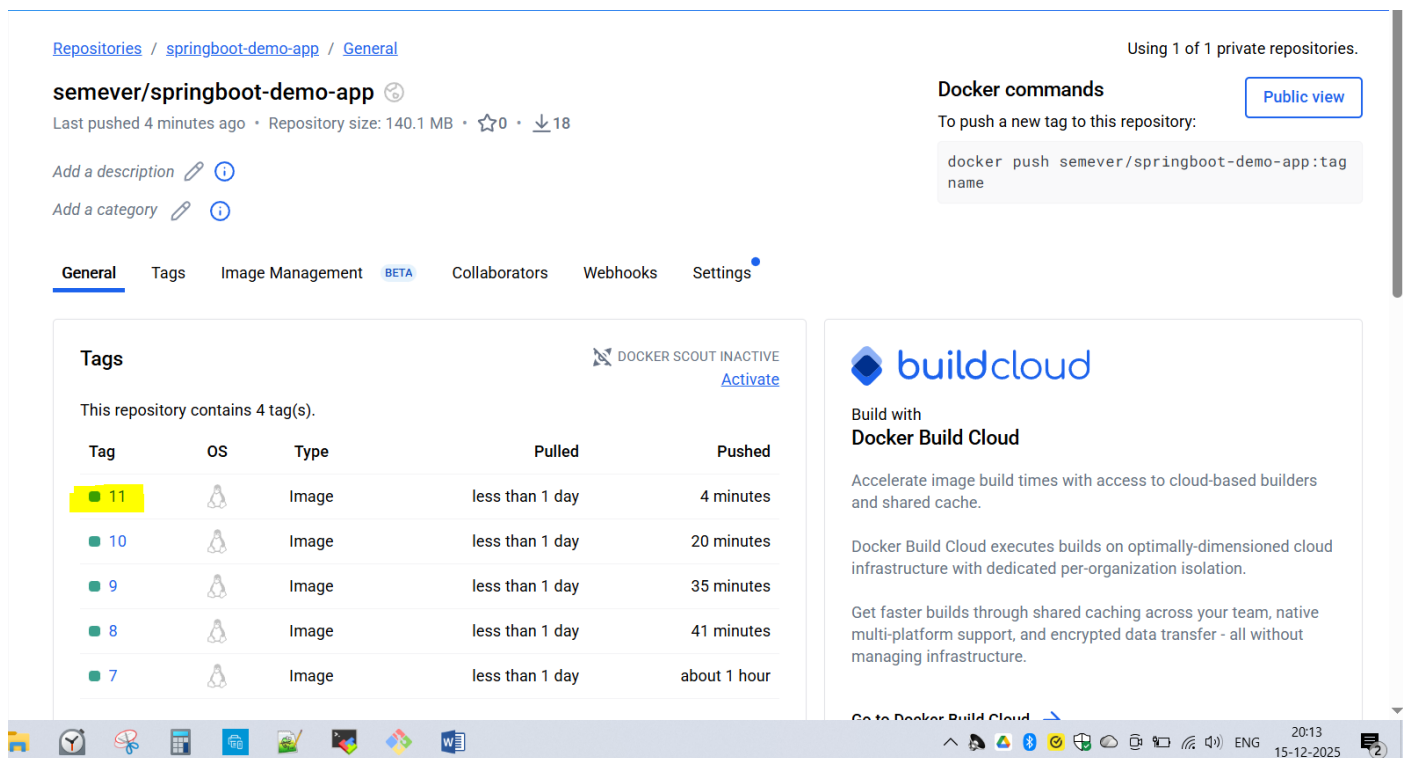
Search

- ✓ Checkout SCM 0.58s
- ✓ Initialize 0.12s
- ✓ Checkout 0.56s
- ✓ Build & Test 3.2s
- ✓ Docker Build & Push 19s
- ✓ Deploy to Kubernetes 0.56s

Post Actions

- ✓ Deployment successful 15ms
- ✓ Deployed Image: semever/springboot-demo-app:11 18ms

Docker Image Published to Docker Hub Registry:



The screenshot shows the Docker Hub repository page for `semever/springboot-demo-app`. The page includes navigation tabs for General, Tags, Image Management, Collaborators, Webhooks, and Settings. The 'Tags' tab is active, displaying a table of image tags. The table lists tags 7 through 11, all of which are 'Image' type and were pushed 'less than 1 day' ago. Tag 11 is the most recent, pushed 4 minutes ago. To the right, there are sections for 'Docker commands' and 'Build with Docker Build Cloud'. The 'Docker commands' section shows the command `docker push semever/springboot-demo-app:tag name`. The 'Build with Docker Build Cloud' section promotes cloud-based builds.

Repositories / [springboot-demo-app](#) / General

semever/springboot-demo-app

Last pushed 4 minutes ago • Repository size: 140.1 MB • ☆0 • ↓18

[Add a description](#) [Add a category](#)

Using 1 of 1 private repositories.

Docker commands [Public view](#)

To push a new tag to this repository:

```
docker push semever/springboot-demo-app:tag name
```

Tags [DOCKER SCOUT INACTIVE](#) [Activate](#)

This repository contains 4 tag(s).

Tag	OS	Type	Pulled	Pushed
11		Image	less than 1 day	4 minutes
10		Image	less than 1 day	20 minutes
9		Image	less than 1 day	35 minutes
8		Image	less than 1 day	41 minutes
7		Image	less than 1 day	about 1 hour

Build with Docker Build Cloud

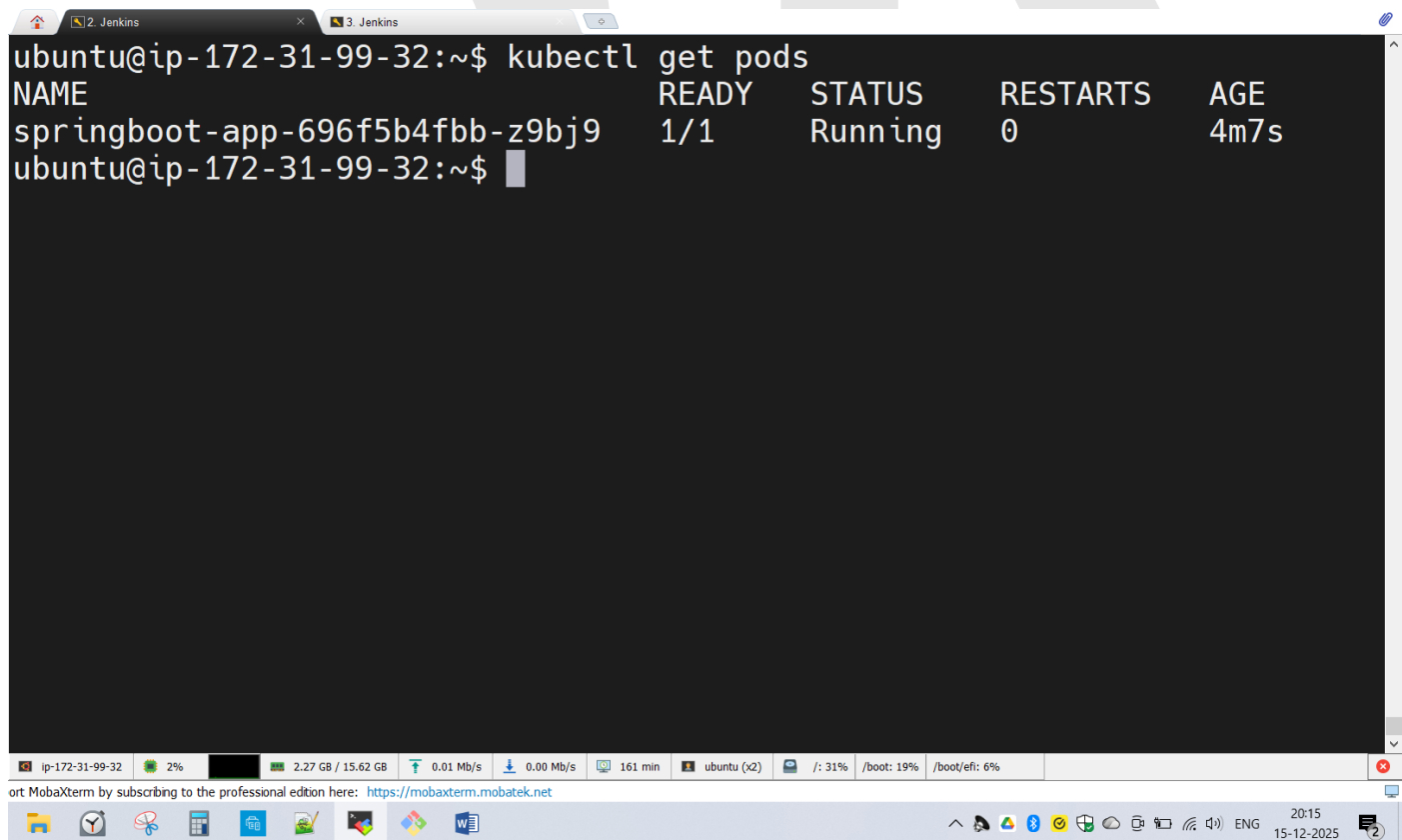
Accelerate image build times with access to cloud-based builders and shared cache.

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.

[Go to Docker Build Cloud](#)

Application Deployed to Kubernetes Cluster Using Images Pulled from Docker Hub:



The screenshot shows a terminal window with a dark background. The user is logged into an Ubuntu machine with IP `ip-172-31-99-32`. They have run the command `kubectl get pods`, which displays the status of a pod named `springboot-app-696f5b4fbb-z9bj9`. The pod is in a 'Running' state with 1/1 ready containers, 0 restarts, and has been running for 4m7s.

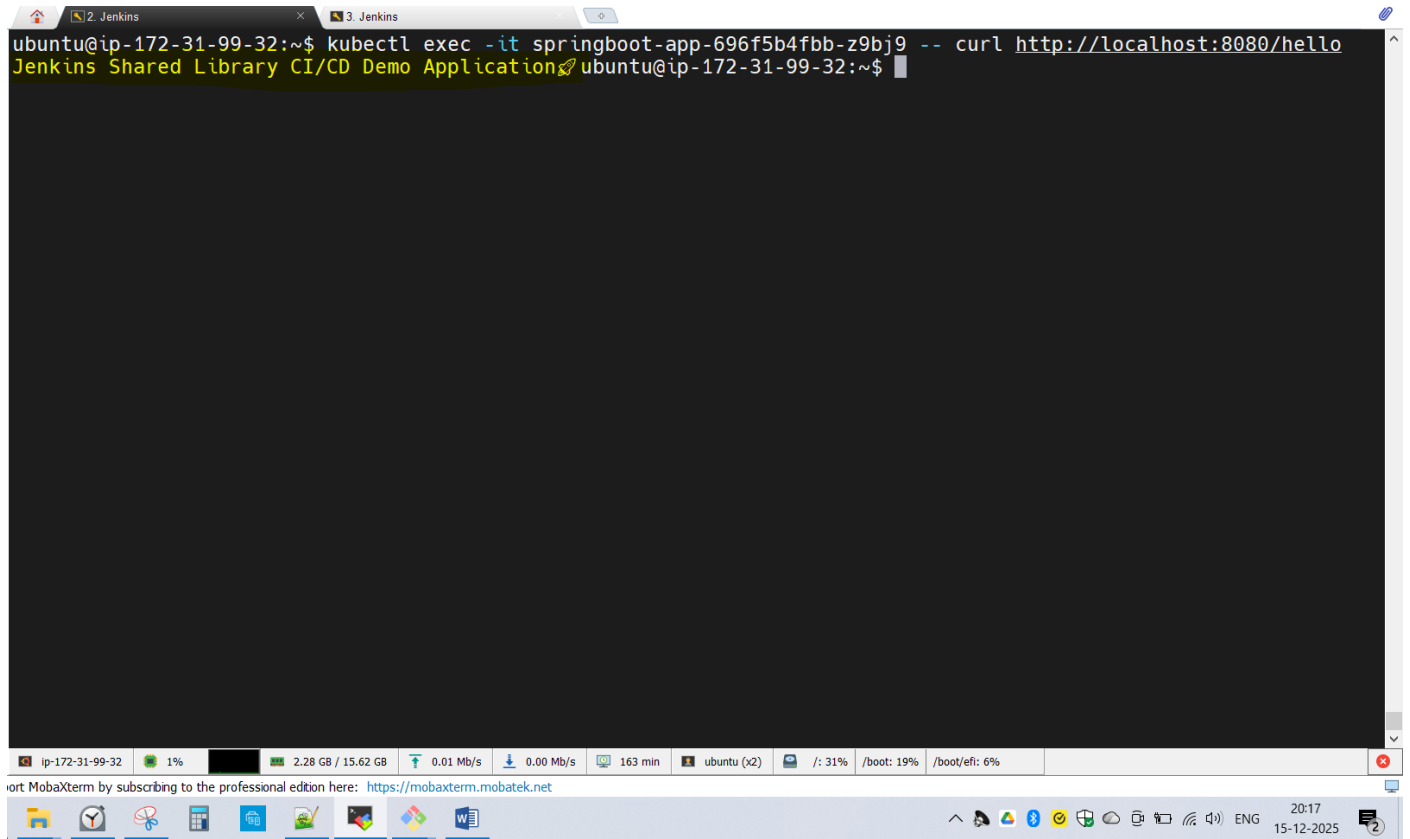
```
ubuntu@ip-172-31-99-32:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
springboot-app-696f5b4fbb-z9bj9    1/1     Running   0           4m7s
ubuntu@ip-172-31-99-32:~$
```

2. Jenkins 3. Jenkins

ip-172-31-99-32 2% 2.27 GB / 15.62 GB 0.01 Mb/s 0.00 Mb/s 161 min ubuntu (x2) /: 31% /boot: 19% /boot/efi: 6%

port MobaXterm by subscribing to the professional edition here: <https://mobaxterm.mobatek.net>

Successful Application Response Confirmed through curl Command:



The screenshot shows a terminal window with a dark background. The command prompt is `ubuntu@ip-172-31-99-32:~$`. The command entered is `kubectl exec -it springboot-app-696f5b4fbb-z9bj9 -- curl http://localhost:8080/hello`. The output of the command is `Jenkins Shared Library CI/CD Demo Application`. The terminal window is titled "2. Jenkins" and "3. Jenkins". The status bar at the bottom shows the IP address `ip-172-31-99-32`, memory usage `2.28 GB / 15.62 GB`, network speed `0.01 Mb/s`, and other system metrics. The taskbar at the bottom includes icons for various applications and the system clock showing `20:17` on `15-12-2025`.

```
ubuntu@ip-172-31-99-32:~$ kubectl exec -it springboot-app-696f5b4fbb-z9bj9 -- curl http://localhost:8080/hello
Jenkins Shared Library CI/CD Demo Application
ubuntu@ip-172-31-99-32:~$
```