

Challenges -01

- **Launch one aws ec2 and install jenkins.**
 - First, we create an AWS EC2 instance.
 - Then we set up a SSH password-less login for ease.
 - Use GitBash to connect to the instance using ssh. - ssh ec2-user@publicip.
 - Then we install Java on the instance using : sudo dnf install -y java-17-amazon-corretto.
 - Check after installing by : java -version.
 - Next, we download the jenkins repo file using : sudo wget link
 - Download the jenkins gpg key using : sudo wget -O link
 - Then we need to set up proper permissions : sudo chmod 644 keypath
 - Then we import the key : sudo rpm --import keypath.
 - Then we install Jenkins : sudo dnf install -y jenkins.
 - Enable Jenkins : sudo systemctl enable --now jenkins.
 - Now we get Jenkins initial password : sudo cat path to file.
 - Open browser and type <http://yourpublicip:8080>.
 - Install suggested plugins.
 - DONE!

```
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; disabled; preset: disabled)
   Active: active (running) since Fri 2025-12-12 07:30:26 UTC; 6s ago
     Main PID: 28362 (java)
        Tasks: 42 (limit: 1012)
      Memory: 362.4M
         CPU: 17.721s
    CGroup: /system.slice/jenkins.service
            └─28362 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war

Dec 12 07:30:22 ip-172-31-72-11.ec2.internal jenkins[28362]: [LF]>
Dec 12 07:30:22 ip-172-31-72-11.ec2.internal jenkins[28362]: [LF]> *****
Dec 12 07:30:22 ip-172-31-72-11.ec2.internal jenkins[28362]: [LF]> *****
Dec 12 07:30:22 ip-172-31-72-11.ec2.internal jenkins[28362]: [LF]> *****
```


Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.


Start building your software project

Create a job +

Set up a distributed build

Set up an agent 

Configure a cloud 

Learn more about distributed builds 

Challenges -01

- Setup Apache Tomcat on Ubuntu Virtual Machine.

- First, we install java on our virtual machine, Ubuntu.
- Then we download Tomcat using: `curl -O link`.
- Then we install Tomcat into /opt using: `sudo mkdir -p /opt/tomcat` and extract using tar command
- Then we create a tomcat service file.: `sudo vi path to tomcat.service`. Then paste the service file from the internet in it.
- Now enable tomcat using `systemctl` command and then start the service using `systemctl` command.
- Check the status : `systemctl status tomcat`
- Got to the browser and open <http://yourserverip:8080>.
- Tomcat should run.

```
vboxuser@Ubuntu:/tmp$ systemctl status tomcat
● tomcat.service - Apache Tomcat Application Server
   Loaded: loaded (/etc/systemd/system/tomcat.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-12-11 07:52:01 UTC; 3min 16s ago
     Process: 6919 ExecStart=/opt/tomcat/bin/startup.sh (code=exited, status=0/SUCCESS)
    Main PID: 6926 (java)
      Tasks: 38 (limit: 14256)
     Memory: 112.4M (peak: 170.1M)
        CPU: 6.311s
      CGroup: /system.slice/tomcat.service
```

Apache Tomcat/9.0.86



If you're seeing this, you've successfully installed Tomcat. Congratulations!



Recommended Reading:

[Security Considerations How-To](#)

[Manager Application How-To](#)

[Clustering/Session Replication How-To](#)

Server Status

Manager App

Host Manager

Developer Quick Start

[Tomcat Setup](#)

[First Web Application](#)

[Realms & AAA](#)

[JDBC DataSources](#)

[Examples](#)

[Servlet Specifications](#)

[Tomcat Versions](#)

- Setup Docker Desktop on your local windows.

- Check your system bios to check if virtualization is enabled. Enable it if not.
- Go to Docker Desktop Website and download the setup from the link.
- Run the setup and follow along to install it.

Challenges -01

- Restart the computer and follow the setup wizard as it guides the homepage.

```
PS C:\Users\Viqaas> docker --version
Docker version 29.1.2, build 890dcca
PS C:\Users\Viqaas> docker run hello-world
```

Hello from Docker!

This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:

1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
(amd64)
3. The Docker daemon created a new container from that image which runs the executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it to your terminal.

To try something more ambitious, you can run an Ubuntu container with:

```
$ docker run -it ubuntu bash
```