

## Web Application Deployment to Apache Tomcat Server using Jenkins CICD

### Technology Stack Used:

1. Java 1.8
2. Maven 3.9.0
3. Git and GitHub
4. Jenkins
5. Apache Tomcat Web Server

### Prerequisites:

1. Working Web Application on Local System
2. GitHub Public Repo
3. AWS Console Access

### Steps:

1. Build the code on local system using Maven.

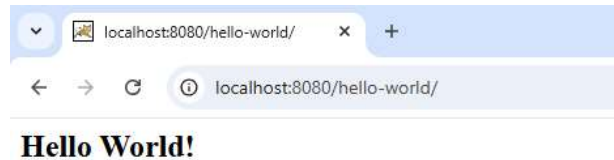
```
C:\Users\sony\Desktop\New folder>mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.test:hello-world >-----
[INFO] Building Hello World 1.0
[INFO] from pom.xml
[INFO] -----[ war ]-----
[INFO]
[INFO] --- clean:3.1.0:clean (default-clean) @ hello-world ---
[INFO]
[INFO] --- resources:3.0.2:resources (default-resources) @ hello-world ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\sony\Desktop\New folder\src\main\resources
[INFO]
[INFO] --- compiler:3.8.0:compile (default-compile) @ hello-world ---
[INFO] No sources to compile
[INFO]
[INFO] --- resources:3.0.2:testResources (default-testResources) @ hello-world ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Users\sony\Desktop\New folder\src\test\resources
[INFO]
[INFO] --- compiler:3.8.0:testCompile (default-testCompile) @ hello-world ---
[INFO] No sources to compile
[INFO]
[INFO] --- surefire:2.22.1:test (default-test) @ hello-world ---
[INFO] No tests to run.
[INFO]
[INFO] --- war:3.2.2:war (default-war) @ hello-world ---
[INFO] Packaging webapp
[INFO] Assembling webapp [hello-world] in [C:\Users\sony\Desktop\New folder\target\hello-world]
[INFO] Processing war project
[INFO] Copying webapp resources [C:\Users\sony\Desktop\New folder\src\main\webapp]
[INFO] Webapp assembled in [47 msecs]
[INFO] Building war: C:\Users\sony\Desktop\New folder\target\hello-world.war
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 2.156 s
[INFO] Finished at: 2024-11-20T22:07:07+05:30
[INFO]
C:\Users\sony\Desktop\New folder>
```

2. Deploy the WAR file which is present on the target folder to Tomcat server.

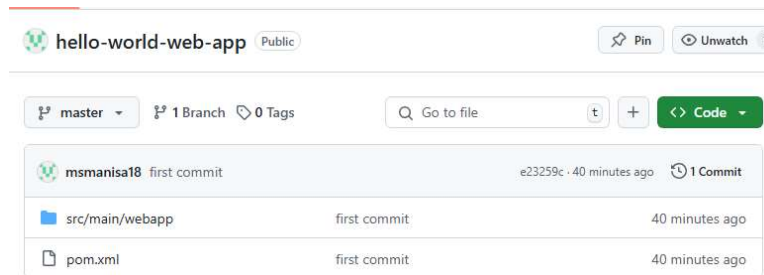
Name	Date modified	Type	Size
hello-world	20-11-2024 22:07	File folder	
maven-archiver	20-11-2024 22:07	File folder	
hello-world.war	20-11-2024 22:07	WAR File	2 KB

apache-tomcat-9.0.89 > webapps				
Name	Date modified	Type	Size	
docs	04-05-2024 01:52	File folder		
examples	04-05-2024 01:52	File folder		
hello-world	20-11-2024 21:28	File folder		
host-manager	04-05-2024 01:52	File folder		
manager	04-05-2024 01:52	File folder		
MvnWebApp	12-06-2024 07:00	File folder		
ROOT	04-05-2024 01:52	File folder		
hello-world.war	20-11-2024 21:27	WAR File	2 KB	
MvnWebApp.war	12-06-2024 07:00	WAR File	3 KB	

3. Start the Apache Tomcat Web Server and test the application on Local System.



4. Commit the code to GitHub.



## 5. Launch one EC2 instance for Jenkins.

**Name and tags** [info](#)

Name

Jenkins

Add additional tags

**▼ Application and OS Images (Amazon Machine Image)** [info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Q Search our full catalog including 1000s of application and OS images

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Linux

SUSE

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI

ami-0aebec83a182ea7ea (64-bit (x86), uefi-preferred) / ami-0279fef81e5a978f6 (64-bit (Arm), uefi)

Virtualization: hvm    ENA enabled: true    Root device type: ebs

Free tier eligible

Description

Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.6.20241111.0 x86\_64 HVM kernel-6.1

**▼ Summary**

Number of instances [info](#)

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.6.2...read more  
ami-0aebec83a182ea7ea

Virtual server type (instance type)

t2.micro

Firewall (security group)

ec2-sg

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel

Launch instance

Preview code

## 6. Install Java on it.

```
[ec2-user@ip-172-31-1-202 ~]$ sudo yum install java-17-amazon-corretto.x86_64 -y
Last metadata expiration check: 0:01:26 ago on Wed Nov 20 16:50:16 2024.
Dependencies resolved.
=====
Package                                                    Architecture
-----
Installing:
java-17-amazon-corretto                                   x86_64
Installing dependencies:
alsa-lib                                                    x86_64
cairo                                                        x86_64
dejavu-sans-fonts                                           noarch
dejavu-sans-mono-fonts                                     noarch
```

## 7. Install Maven on it.

```
[ec2-user@ip-172-31-1-202 ~]$ sudo yum install maven -y
Last metadata expiration check: 0:03:00 ago on Wed Nov 20 16:50:16 2024.
Dependencies resolved.
=====
Package                                                    Architecture
-----
Installing:
maven                                                        noarch
Installing dependencies:
apache-commons-cli                                          noarch
apache-commons-codec                                        noarch
apache-commons-io                                           noarch
```

## 8. Install Git on it.

```
[ec2-user@ip-172-31-1-202 ~]$ sudo yum install git -y
Last metadata expiration check: 0:04:23 ago on Wed Nov 20 16:50:16 2024.
Dependencies resolved.
```

Package	Architecture
Installing:	
git	x86_64
Installing dependencies:	
git-core	x86_64

## 9. Install Jenkins

```
[ec2-user@ip-172-31-1-202 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2024-11-20 16:58:53-- https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)[151.101.154.133]:443... connected.
HTTP request sent, awaiting response... 200 OK
length: 85
saving to: '/etc/yum.repos.d/jenkins.repo'

/etc/yum.repos.d/jenkins.repo          100%[=====]
2024-11-20 16:58:54 (3.36 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[ec2-user@ip-172-31-1-202 ~]$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
[ec2-user@ip-172-31-1-202 ~]$ sudo yum upgrade
jenkins-stable
Dependencies resolved.

Problem 1: package dracut-102-3.amzn2023.0.1.x86_64 from amazonlinux conflicts with dracut-config-ec2 < 3.1 provided by dracut-config-ec2-3.0-4.amzn2023.0.2.noarch
- cannot install the best update candidate for package dracut-055-6.amzn2023.0.8.x86_64
Problem 2: problem with installed package dracut-config-ec2-3.0-4.amzn2023.0.2.noarch
- package dracut-102-3.amzn2023.0.1.x86_64 from amazonlinux conflicts with dracut-config-ec2 < 3.1 provided by dracut-config-ec2-3.0-4.amzn2023.0.2.noarch
- package dracut-102-3.amzn2023.0.1.x86_64 from amazonlinux conflicts with dracut-config-ec2 < 3.1 provided by dracut-config-ec2-3.0-4.amzn2023.0.2.noarch
- package dracut-config-generic-102-3.amzn2023.0.1.x86_64 from amazonlinux requires dracut = 102-3.amzn2023.0.1, but none of the provided packages can provide it
- cannot install the best update candidate for package dracut-config-generic-055-6.amzn2023.0.8.x86_64

Package Architecture Version
-----
dracut x86_64 102-3.amzn2023.0.1
dracut-config-generic x86_64 102-3.amzn2023.0.1

Transaction Summary
-----
Skip 2 Packages
Nothing to do.
Complete!
[ec2-user@ip-172-31-1-202 ~]$ sudo yum install jenkins -y
Last metadata expiration check: 0:00:17 ago on Wed Nov 20 16:59:23 2024.
Dependencies resolved.

Package Architecture Version
-----
jenkins noarch 2.479.1-1.1
```

## 10. Enable Jenkins

```
[ec2-user@ip-172-31-1-202 ~]$ sudo systemctl enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-172-31-1-202 ~]$ sudo systemctl start jenkins
[ec2-user@ip-172-31-1-202 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: disabled)
   Active: active (running) since Wed 2024-11-20 17:02:52 UTC; 3s ago
     Main PID: 28289 (java)
       Tasks: 40 (limit: 1113)
      Memory: 357.1M
         CPU: 17.915s
    CGroup: /system.slice/jenkins.service
            └─28289 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins

Nov 20 17:02:43 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: b4234613b80a4f1db85256456cb2b994
Nov 20 17:02:43 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: This may also be found at: /var/lib/jenkins
Nov 20 17:02:43 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: *****
Nov 20 17:02:43 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: *****
Nov 20 17:02:43 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: *****
Nov 20 17:02:52 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: 2024-11-20 17:02:52.389+0000 [id=30]
Nov 20 17:02:52 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: 2024-11-20 17:02:52.412+0000 [id=23]
Nov 20 17:02:52 ip-172-31-1-202.ap-south-1.compute.internal systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server
Nov 20 17:02:53 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: 2024-11-20 17:02:53.931+0000 [id=46]
Nov 20 17:02:53 ip-172-31-1-202.ap-south-1.compute.internal jenkins[28289]: 2024-11-20 17:02:53.932+0000 [id=46]
lines 1-20/20 (END)
[ec2-user@ip-172-31-1-202 ~]$
```

## 11. Enable port 8080 for Jenkins and 8085 for Tomcat under security group.

## Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

### Inbound rules [Info](#)

Security group rule ID	Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>	Source <a href="#">Info</a>	Description - optional <a href="#">Info</a>	
sgr-01b884a4e17d5343d	SSH	TCP	22	Custom	For SSH Access	Delete
-	Custom TCP	TCP	8080	Anywhere-I...	For Jenkins	Delete
-	Custom TCP	TCP	8085	Anywhere-I...	For Tomcat	Delete

Add rule

## 12. Configure Jenkins

65.0.176.8:8080/login?from=%2F

### Getting Started

## Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

## 13. Install required plugins and create a CICD pipeline on Jenkins.



## New Item

Enter an item name

CICD

Select an item type



### Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



### Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.



### Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

## 14. Launch another EC2 instance for Tomcat.

EC2 > ... > Launch an instance

### Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

**Name and tags**

Name

Tomcat

Add additional tags

**Application and OS Images (Amazon Machine Image)**

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Linux

SUS

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI

ami-0aebec83a182ea7ea (64-bit (x86), uefi-preferred) / ami-0279fef81e5a978f6 (64-bit (Arm), uefi)

Free tier eligible

**Summary**

Number of instances

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.6.2...read more

ami-0aebec83a182ea7ea

Virtual server type (instance type)

t2.micro

Firewall (security group)

ec2-sg

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel

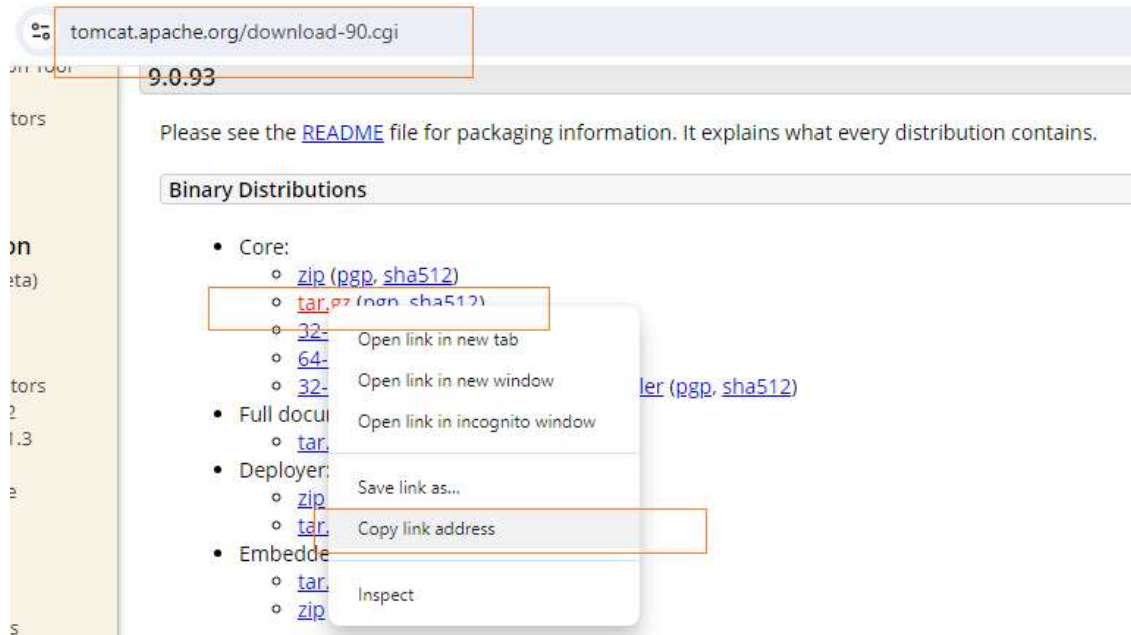
Launch instance

Preview code

## 15. Install Java on it.

## 16. Download Apache Tomcat tar.gz file.

6



17. Untar the tar file by using:

```
tar -xzf apache-tomcat-9.0.93.tar.gz
```

18. Change Connector port from 8080 to 8085 on server.xml file present on conf directory.

```
-->
<Connector port="8085" protocol="HTTP/1.1"
  connectionTimeout="20000"
  redirectPort="8443"
  maxParameterCount="1000"
/>
```

19. Update tomcat-users.xml file present on conf directory, add user details on it.

```
<role rolename="manager-gui,admin-gui,manager-script"/>
<user username="admin" password="admin" roles="manager-gui,admin-gui,manager-script"/>
```

```
<role rolename="manager-gui,admin-gui,manager-script"/>
<user username="admin" password="admin" roles="manager-gui,admin-gui,manager-script"/>

</tomcat-users>
```

20. Update the context.xml file present under /tomcat/webapps/host-manager/META-INF  
Comment out following line on it:

```
<!--
<valve className="org.apache.catalina.valves.RemoteAddrValve"
  allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />

-->
```

```
-->
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
    sameSiteCookies="strict" />

  <!--
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
    allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />

  -->
    <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Integer|
:\$1)?|java\.util\.(?:Linked)?HashMap"/>
</Context>
[ec2-user@ip-172-31-46-236 META-INF]$ |
```

21. Update the context.xml file present under /tomcat/webapps/manager/META-INF  
Comment out following line on it:

```
<!--
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
  allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />

-->
```

```

Limitations under the License.
-->
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
    sameSiteCookies="strict" />

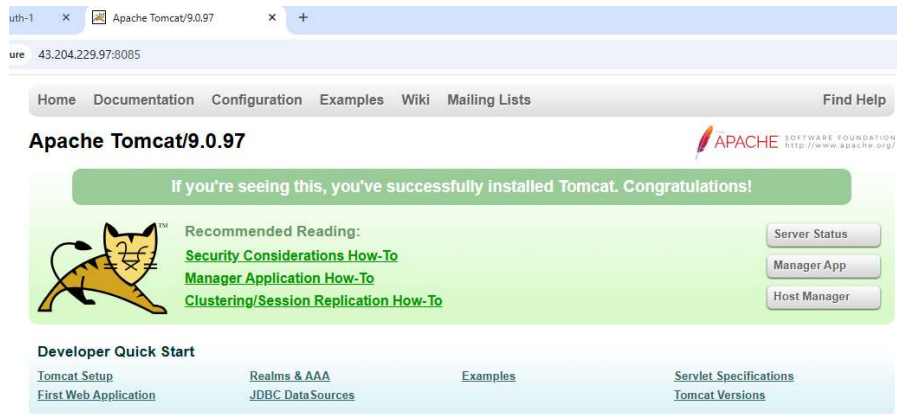
  <!--
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
    allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />

  -->
```

22. Start tomcat server, it will run on port 8085 now.

```
[root@ip-172-31-46-236 tomcat]# ./bin/startup.sh
Using CATALINA_BASE:   /opt/tomcat
Using CATALINA_HOME:   /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME:        /
Using CLASSPATH:       /opt/tomcat/bin/bootstrap.jar:/opt/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
[root@ip-172-31-46-236 tomcat]# |
```



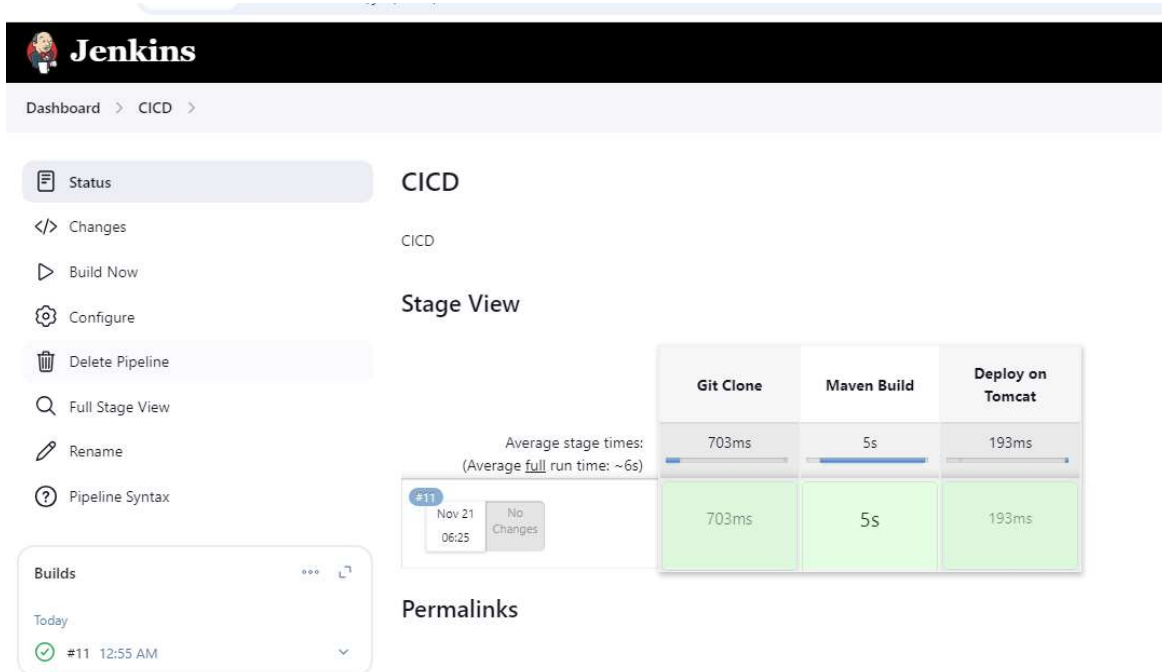


23. Create a pipeline on Jenkins using following Pipeline Script:

```
pipeline {
    agent any

    stages {
        stage('Git Clone') {
            steps {
                git 'https://github.com/msmanisa18/hello-world-web-app.git'
            }
        }
        stage('Maven Build') {
            steps {
                sh 'mvn clean package'
            }
        }
        stage('Deploy on Tomcat') {
            steps {
                deploy adapters: [tomcat9(credentialsId: 'c76464ce-c3ce-41f0-9c44-db521110a52c', path: '', url:
'http://43.204.229.97:8085/']], contextPath: null, war: '**/*.war'
            }
        }
    }
}
```

24. Build the pipeline now:



The image shows the Jenkins web interface for a pipeline named 'CICD'. The left sidebar contains navigation links: Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, and Pipeline Syntax. The main area displays the 'Stage View' for the 'CICD' pipeline. It shows a table of stage times for three stages: Git Clone (703ms), Maven Build (5s), and Deploy on Tomcat (193ms). The average stage times are 703ms, 5s, and 193ms respectively. The average full run time is approximately 6s. A build #11 is shown as completed on Nov 21 at 06:25 with no changes. Below the stage view, there is a 'Permalinks' section.

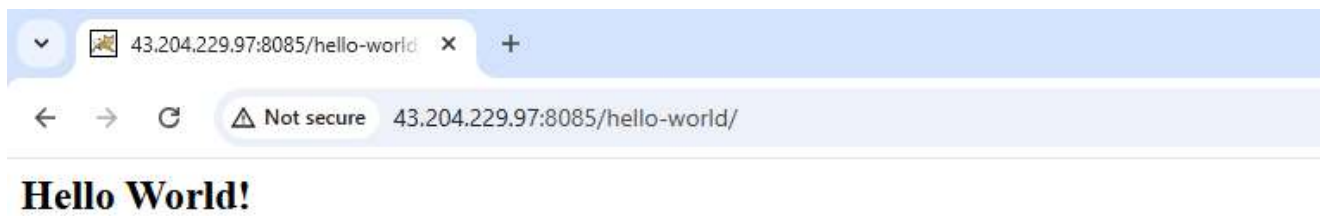
Stage	Git Clone	Maven Build	Deploy on Tomcat
Average stage times:	703ms	5s	193ms
(Average full run time: ~6s)			

Builds

Today

#11 12:55 AM

25. Final Output:



The image shows a web browser window with the address bar displaying '43.204.229.97:8085/hello-world/'. The page content displays 'Hello World!' in a large, bold, black font.