

Docker Assignment 2

Step 1: Launched three instances, Jenkins Master, Slave1 and Slave2

The screenshot shows the AWS Management Console 'Instances' page. Three EC2 instances are listed and highlighted with a red box:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
Jenkins Master (Docker)	i-089629caf8d7bfc2e	Running	t3.micro	Initializing	View alarms +	ap-south-1a	ec2-3-6-88-...
Linux Slave 1	i-05ead13f1fe870d94	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1a	ec2-43-205-...
Linux Slave 2	i-0b533c87befd61e57	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1a	ec2-43-205-...

Below the table, the 'Monitoring' section shows various metrics like CPU utilization, Network in/out, and Network packets in (count) over a 1-hour period.

Step 2: Installed Java-17, Git and Docker on all the instance:

The screenshot shows the AWS Management Console 'Instances' page with a terminal window open for the instance 'i-0bb0216d314c17b07 (Linux Jenkins Master)'. The terminal output shows the installation of Java-17, Git, and Docker:

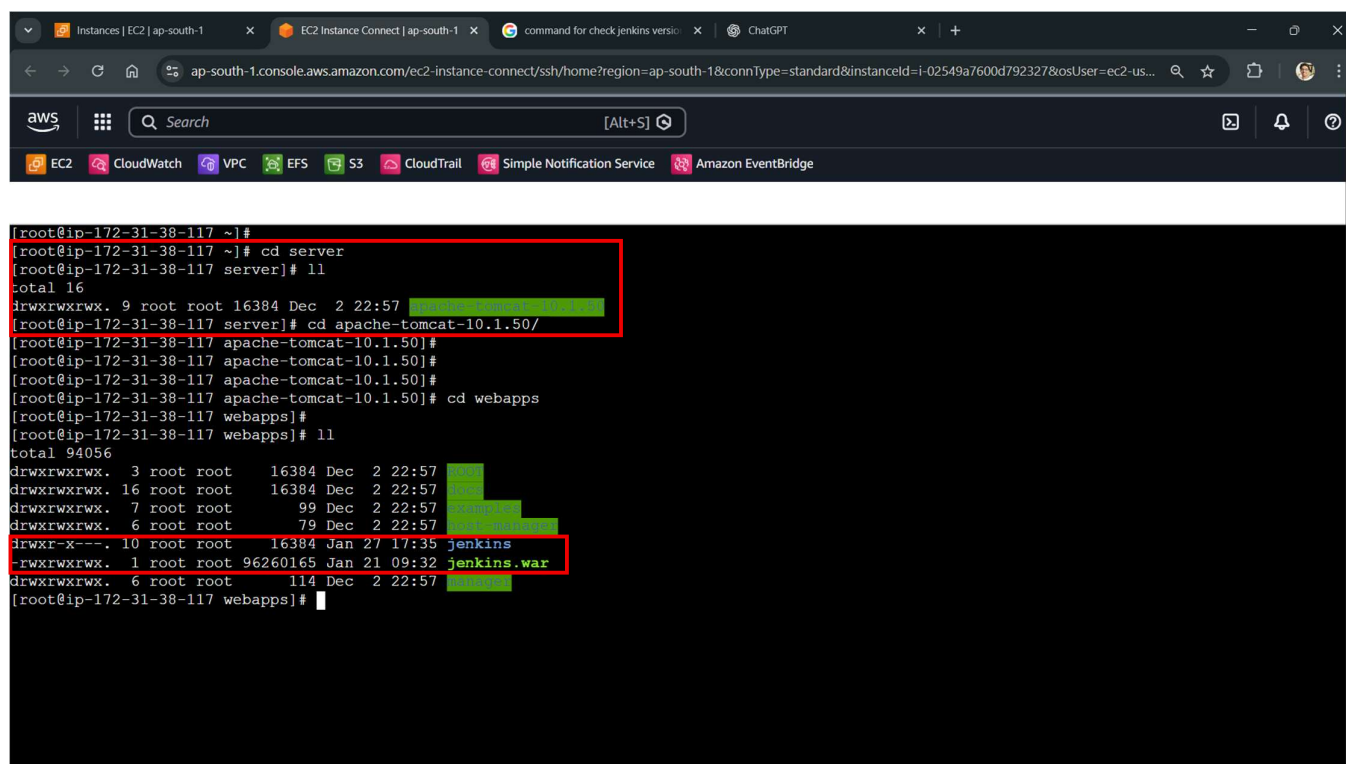
```
[root@ip-172-31-40-216 ~]# java --version
openjdk 17.0.18 2026-01-20 LTS
OpenJDK Runtime Environment Corretto-17.0.18.9.1 (build 17.0.18+9-LTS)
OpenJDK 64-Bit Server VM Corretto-17.0.18.9.1 (build 17.0.18+9-LTS, mixed mode, sharing)

[root@ip-172-31-40-216 ~]# git -v
git version 2.50.1

[root@ip-172-31-40-216 ~]# docker -v
Docker version 25.0.14, build 0bab007
```

The terminal output is highlighted with a red box. Below the terminal window, the instance details for 'i-0bb0216d314c17b07 (Linux Jenkins Master)' are shown, including the public IP address 13.233.132.0 and the private IP address 172.31.40.216.

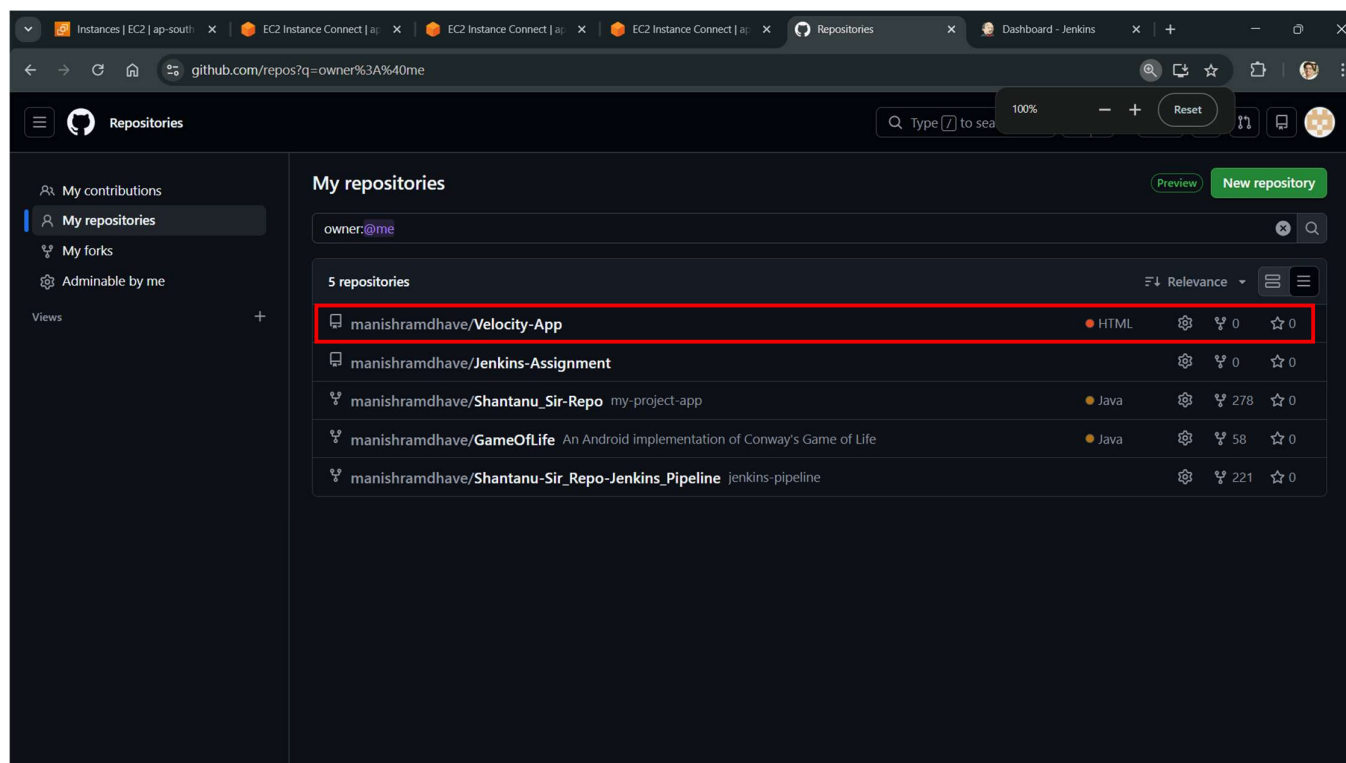
Step 3: Installed Apache-Tomcat-10 and Jenkins on the Jenkins Master Instance:



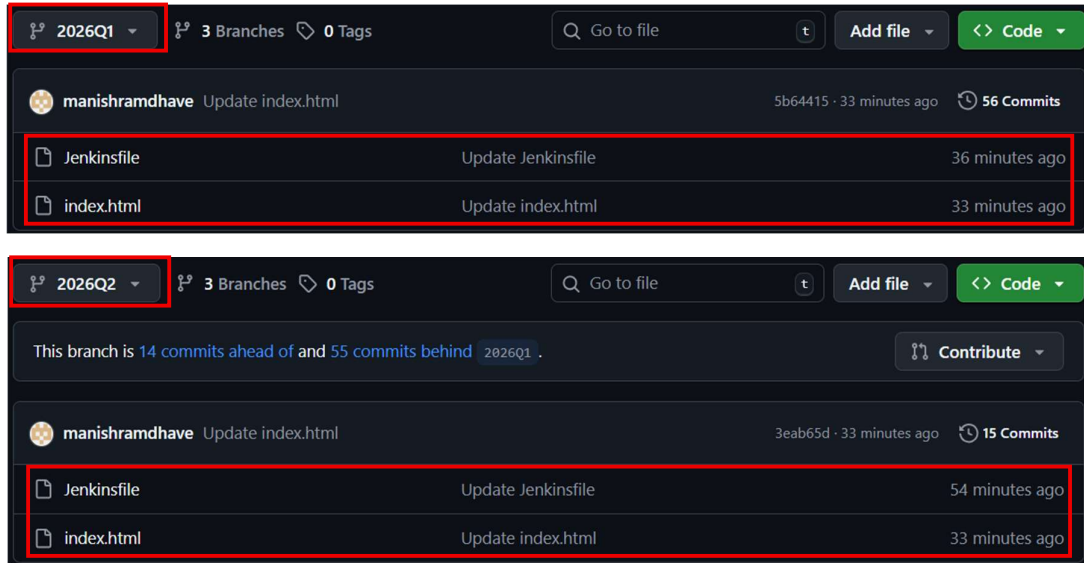
The screenshot shows the AWS Management Console with a terminal window open for an EC2 instance. The terminal output shows the user navigating to the 'server' directory, then to the 'apache-tomcat-10.1.50' directory, and finally to the 'webapps' directory. The 'ls' command is run in the 'webapps' directory, showing a list of files including 'jenkins' and 'jenkins.war'. The 'jenkins' directory is highlighted in red, and the 'jenkins.war' file is highlighted in green.

```
[root@ip-172-31-38-117 ~]#  
[root@ip-172-31-38-117 ~]# cd server  
[root@ip-172-31-38-117 server]# ll  
total 16  
drwxrwxrwx. 9 root root 16384 Dec 2 22:57 apache-tomcat-10.1.50/  
[root@ip-172-31-38-117 apache-tomcat-10.1.50]#  
[root@ip-172-31-38-117 apache-tomcat-10.1.50]#  
[root@ip-172-31-38-117 apache-tomcat-10.1.50]# cd webapps  
[root@ip-172-31-38-117 webapps]#  
[root@ip-172-31-38-117 webapps]# ll  
total 94056  
drwxrwxrwx. 3 root root 16384 Dec 2 22:57 .  
drwxrwxrwx. 16 root root 16384 Dec 2 22:57 ..  
drwxrwxrwx. 7 root root 99 Dec 2 22:57 .svn  
drwxrwxrwx. 6 root root 79 Dec 2 22:57 .svn  
drwxr-xr-x. 10 root root 16384 Jan 27 17:35 jenkins  
-rwxrwxrwx. 1 root root 96260165 Jan 21 09:32 jenkins.war  
drwxrwxrwx. 6 root root 114 Dec 2 22:57 .svn  
[root@ip-172-31-38-117 webapps]#
```

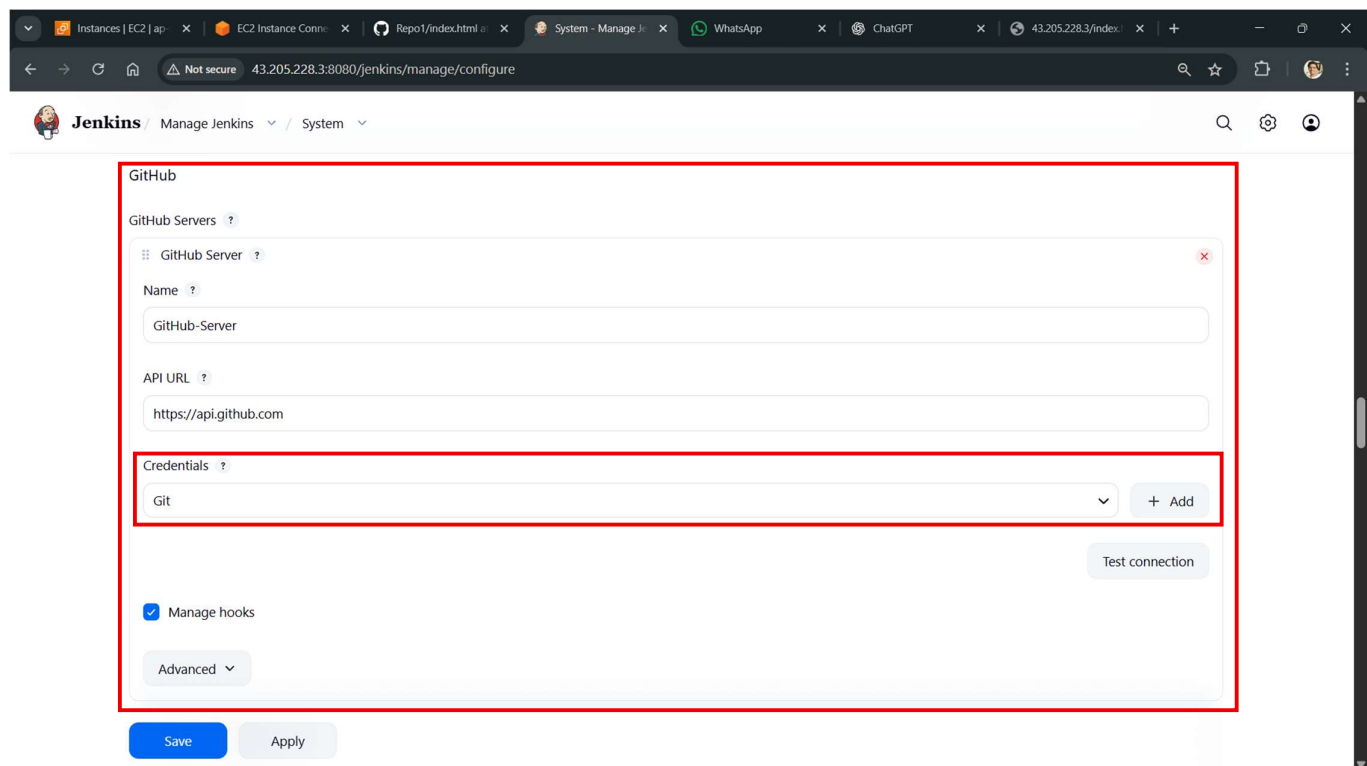
Step 4: Made a Private Repository named 'Velocity-App' in GitHub account:



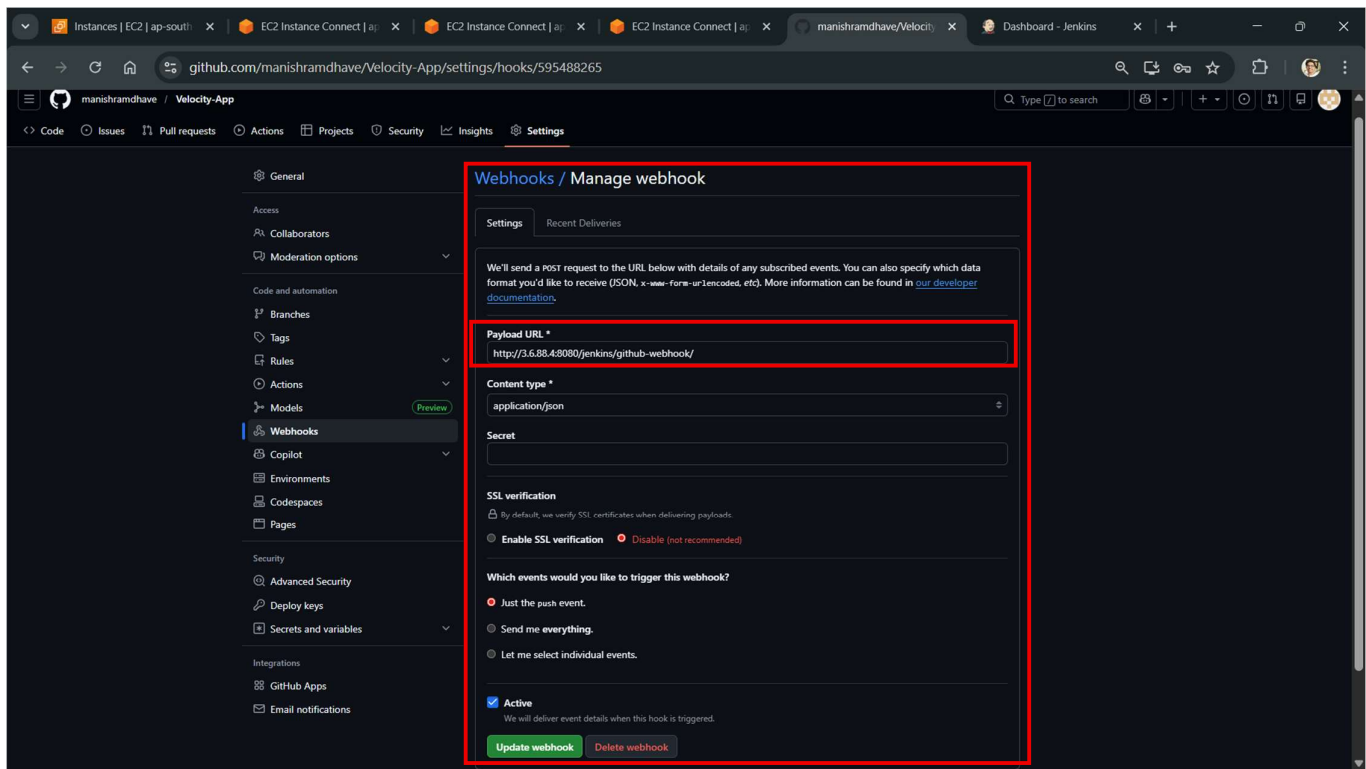
Step 5: Created two branches, **2026Q1** and **2026Q2** in the ‘**Velocity-App**’ Repository and pushed **two different ‘index.html’** files and also created **two different ‘Jenkinsfile’** files in respective branches:



Step 6: Created an **API Connection between Jenkins to GitHub** Repositories in ‘**Manage Jenkins**’ by creating a **Secret Text (Credential)** using a GitHub Token in Jenkins:



Step 7: Created an **API Connection between Jenkins and GitHub Repositories** by using ‘Git Webhooks’ (i.e. setting up Jenkins URL in Git) :



Step 8: Created two Nodes for to establish the connection between Jenkins Master and each Slave Instances using the ‘**Credential**’ and ‘**Manually trusted key Verification Strategy**’:

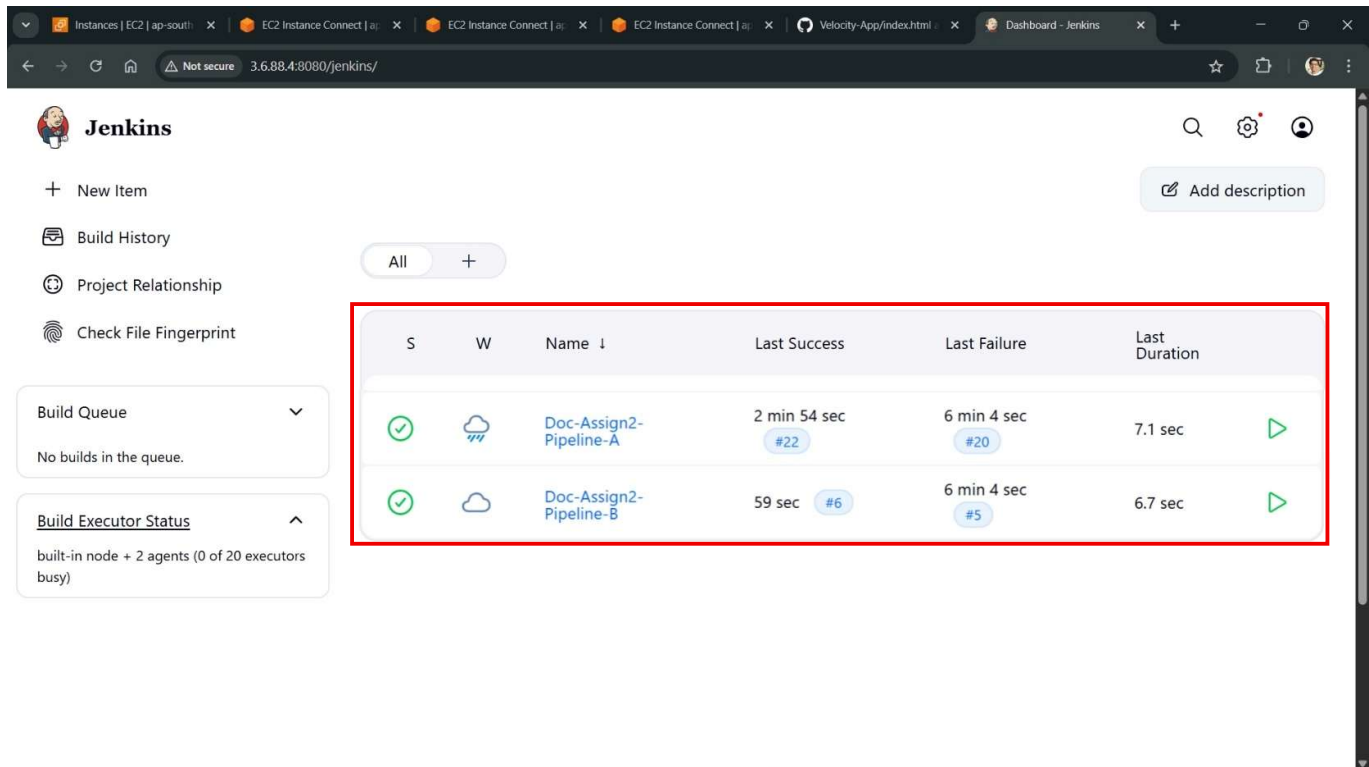
The screenshot shows the Jenkins 'Nodes' page. The table below lists the nodes:

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	4.87 GiB	0 B	4.87 GiB	0ms
	Slave1	Linux (amd64)	In sync	5.62 GiB	0 B	458.39 MiB	46ms
	Slave2	Linux (amd64)	In sync	5.61 GiB	0 B	458.39 MiB	55ms
last checked		53 min	53 min	53 min	53 min	53 min	53 min

Legend: S M L

REST API Jenkins 2.541.1

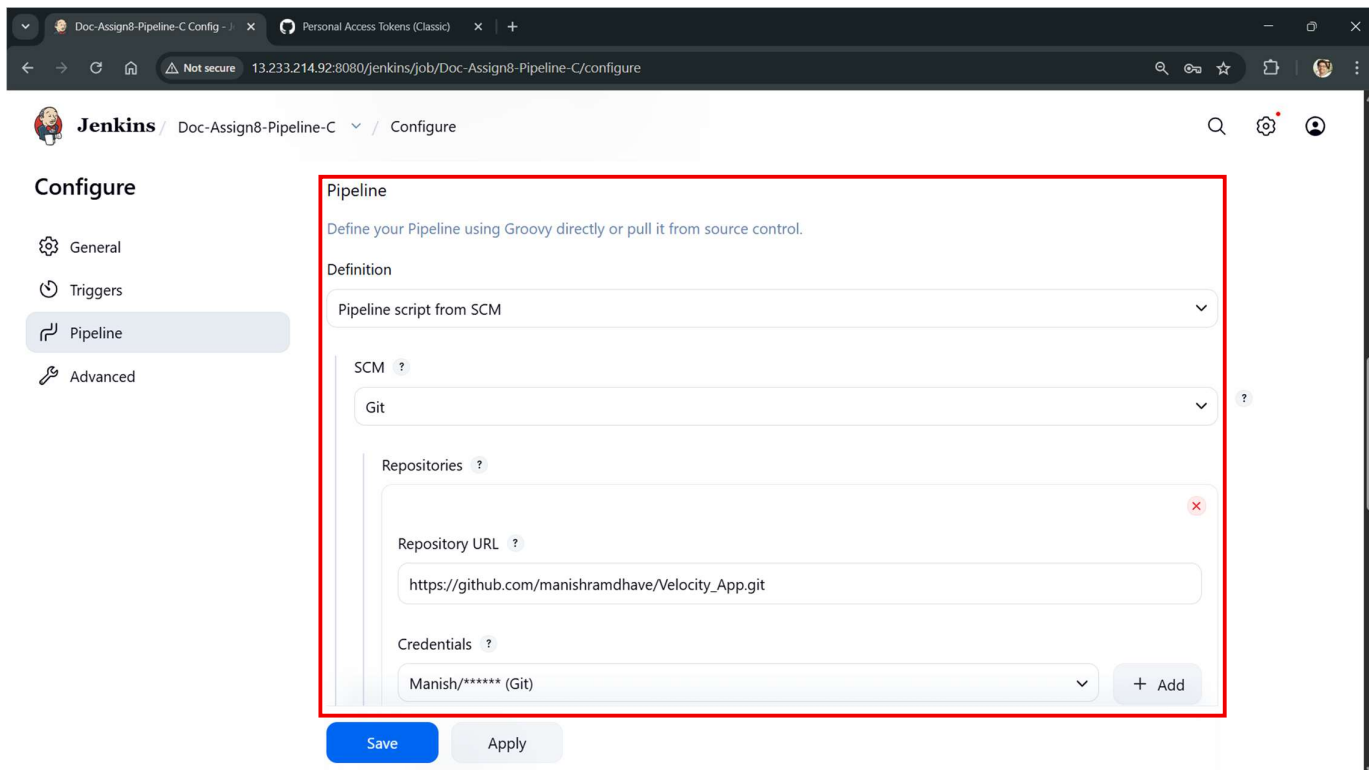
Step 9: Launched the Jenkins and created two different Pipeline Jobs. **Job1 and Job2:**



The screenshot shows the Jenkins Dashboard. On the left, there are links for 'New Item', 'Build History', 'Project Relationship', and 'Check File Fingerprint'. Below these are two boxes: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (built-in node + 2 agents (0 of 20 executors busy)). The main area displays a table of pipeline jobs, with two jobs highlighted by a red box:

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☁	Doc-Assign2-Pipeline-A	2 min 54 sec #22	6 min 4 sec #20	7.1 sec	▶
✓	☁	Doc-Assign2-Pipeline-B	59 sec #6	6 min 4 sec #5	6.7 sec	▶

Step 11: Used ‘Pipeline script from SCM’ and selected the SCM as a ‘Git’ which will follow the Jenkins pipeline script by using the ‘Jenkinsfile’ in the repository:



The screenshot shows the Jenkins Configuration page for a pipeline job. The left sidebar has links for 'General', 'Triggers', 'Pipeline', and 'Advanced'. The main area is titled 'Configure' and contains a red box highlighting the 'Pipeline' configuration section:

Pipeline
Define your Pipeline using Groovy directly or pull it from source control.

Definition
Pipeline script from SCM

SCM ?
Git

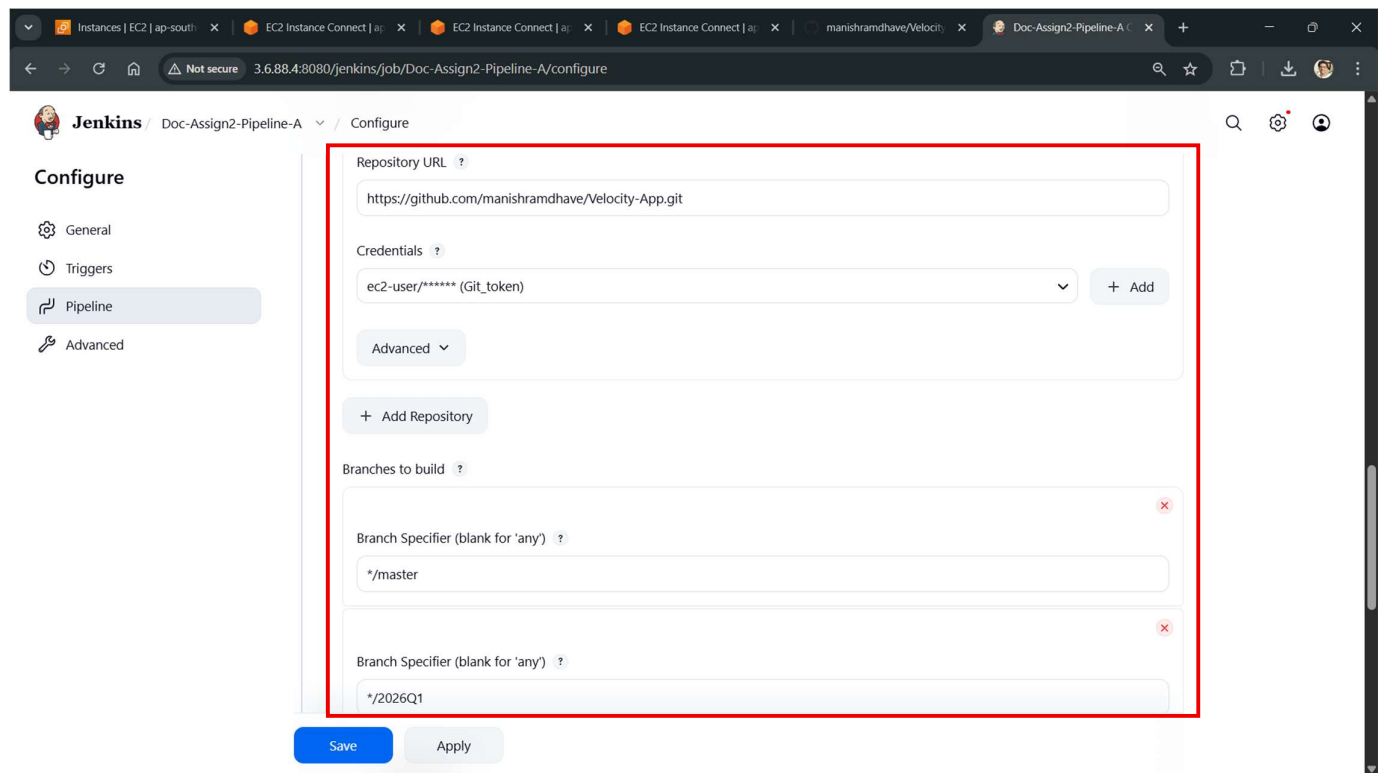
Repositories ?

Repository URL ?
https://github.com/manishramdhave/VelocityEngine

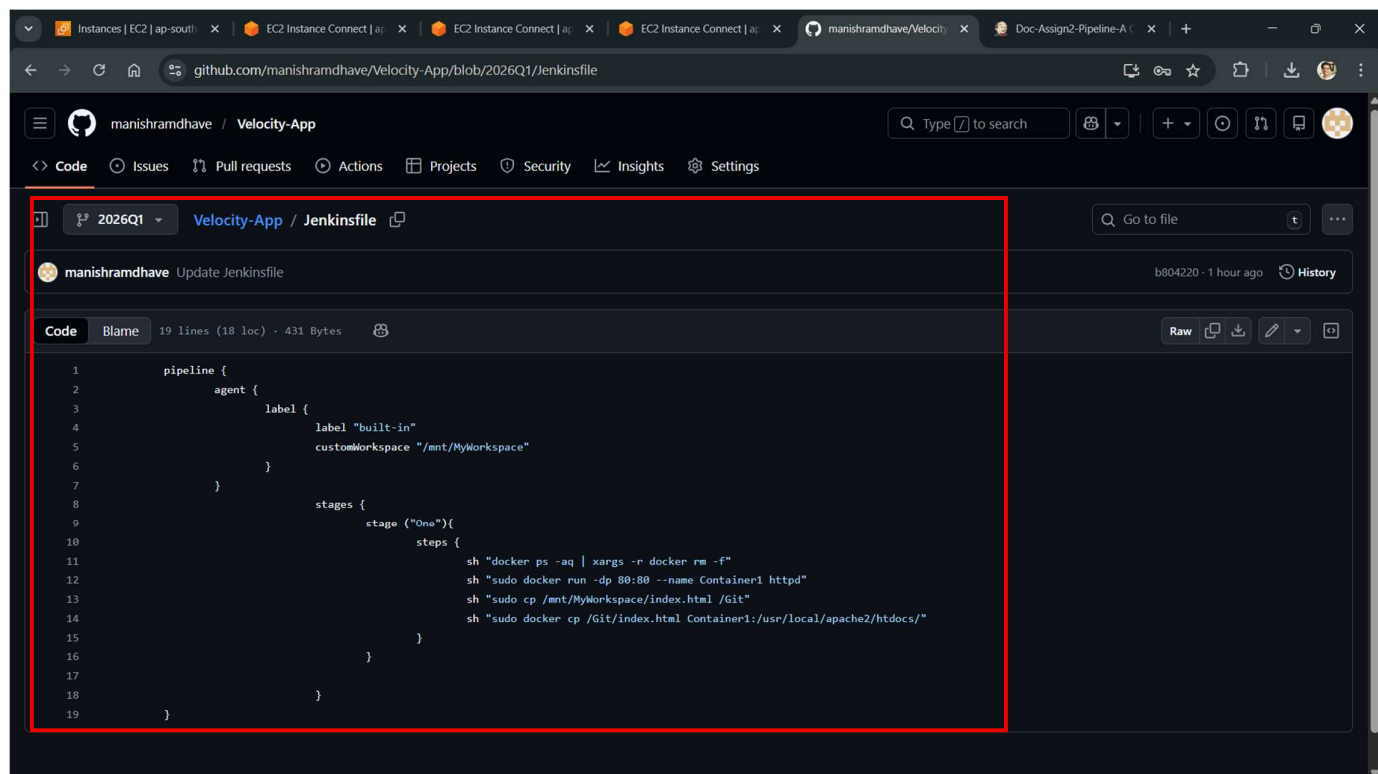
Credentials ?
Manish/***** (Git)

Buttons: Save, Apply

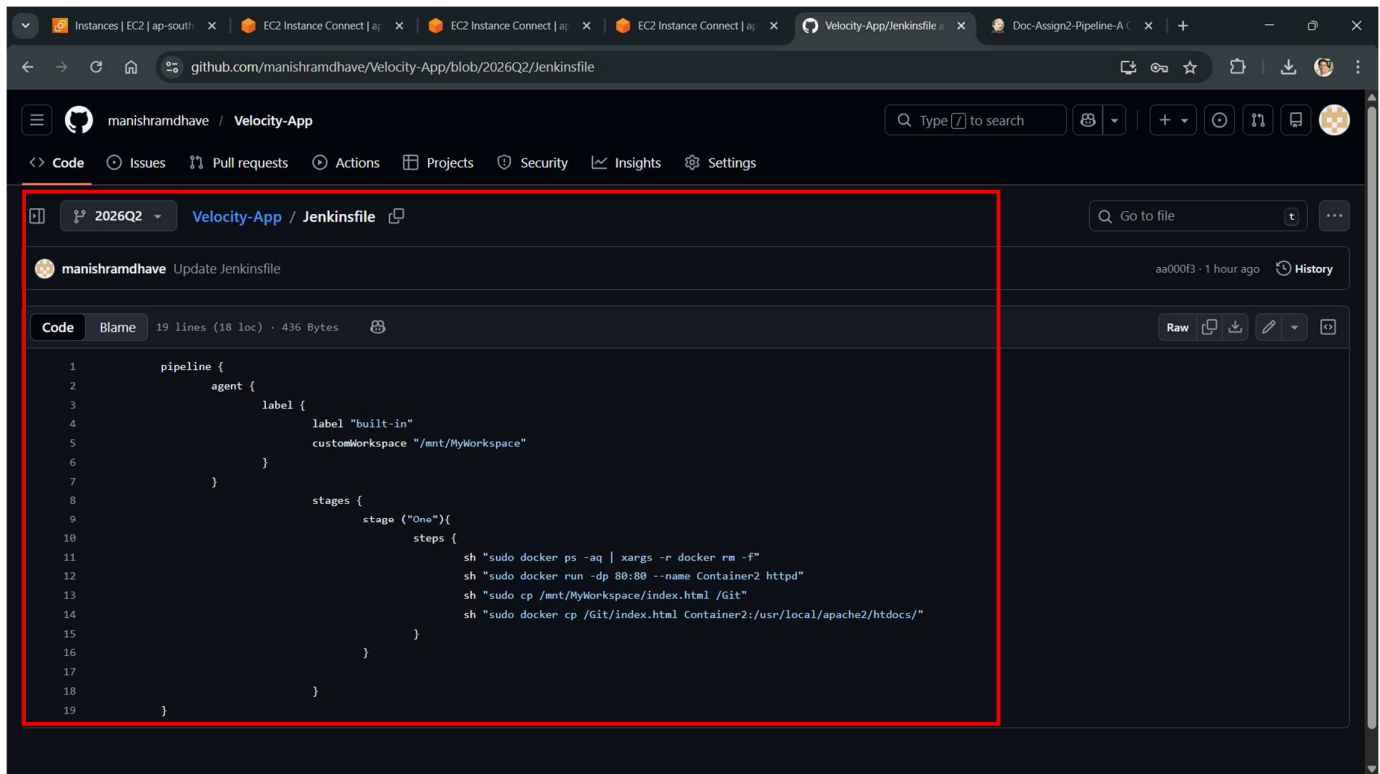
Step 12: Integrated Git branches, **2026Q1** and **2026Q2** with Jenkins by creating ‘Credentials’ by using the Git Token on **both the pipeline Job-A and Job-B** respectively:



Step 13: Created a ‘Jenkinsfile’ on **2026Q1** branch in which out script is mentioned:



Step 14: Created a 'Jenkinsfile' on 2026Q2 branch in which out script is mentioned:



The screenshot shows a GitHub web interface for the repository 'manishramdhare / Velocity-App'. The '2026Q2' branch is selected, and the file 'Jenkinsfile' is open. The file content is a Jenkins pipeline script. The script defines a pipeline with an agent labeled 'built-in' and a custom workspace. It contains a single stage named 'Ono' with four steps: listing Docker containers, running a Docker container named 'Container2' as 'httpd', copying the index.html file from the workspace to the container, and copying the Git directory from the workspace to the container's local Apache2 directory.

```
1 pipeline {
2   agent {
3     label {
4       label "built-in"
5       customWorkspace "/mnt/MyWorkspace"
6     }
7   }
8   stages {
9     stage ("Ono"){
10      steps {
11        sh "sudo docker ps -aq | xargs -r docker rm -f"
12        sh "sudo docker run -dp 80:80 --name Container2 httpd"
13        sh "sudo cp /mnt/MyWorkspace/index.html /Git"
14        sh "sudo docker cp /Git/index.html Container2:/usr/local/apache2/htdocs/"
15      }
16    }
17  }
18 }
19 }
```


Result:

1. When changes are done in 2026Q1 branch, Build is triggered by **'Doc-Assign2-Pipeline-A'** and the updated index.html file is hosted from the container of a **Slave1** instance by using the Pipeline script in the **'Jenkinsfile'** file of the same branch:

The screenshot displays the Jenkins interface for build #24 of 'Doc-Assign2-Pipeline-A'. The build is successful, as indicated by the green checkmark. The commit history shows four changes: 'Hosting updated index.html from Container 1 of 2026Q1 Branch', 'Change 1', 'Change 2', and 'Done with Docker Assignment 2'. The Jenkinsfile script is shown, defining a pipeline with a 'slave1' agent and a stage 'One' containing steps for running a Docker container, copying files, and updating the index.html file. The build output shows the updated index.html file being hosted from Container 1 of the 2026Q1 Branch.

2. When changes are done in 2026Q2 branch, Build is triggered by **'Doc-Assign2-Pipeline-B'** and the updated index.html file is hosted from the container of a **Slave2** instance by using the Pipeline script in the **'Jenkinsfile'** file of the same branch:

The screenshot displays the Jenkins interface for build #16 of 'Doc-Assign2-Pipeline-B'. The build is successful, as indicated by the green checkmark. The commit history shows four changes: 'Hosting updated index.html from container 2 of 2026Q2 Branch', 'Change 1', 'Change 2', and 'Manish here'. The Jenkinsfile script is shown, defining a pipeline with a 'slave2' agent and a stage 'One' containing steps for running a Docker container, copying files, and updating the index.html file. The build output shows the updated index.html file being hosted from Container 2 of the 2026Q2 Branch.