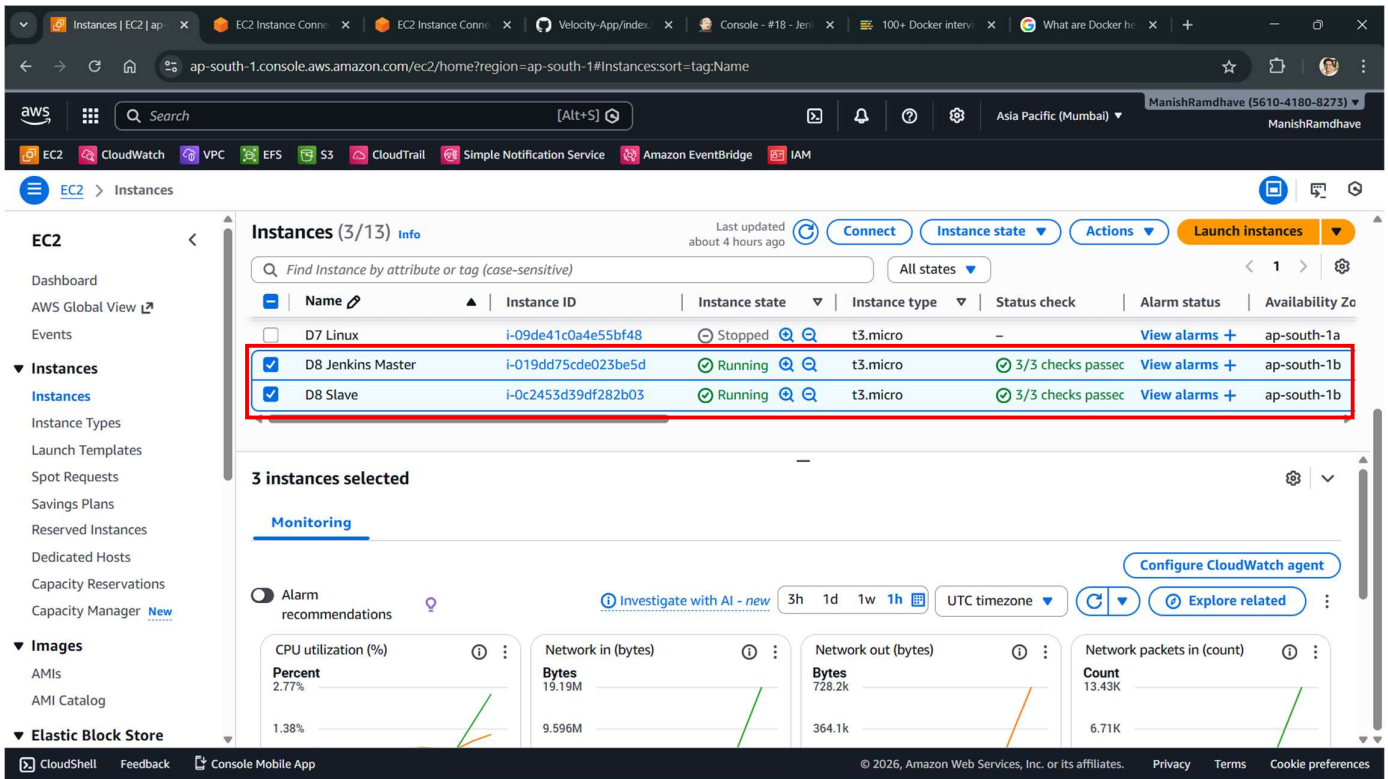


Docker Assignment 8 (Task 2)

Step 1: Launched instances for our Jenkins Master and Slave:

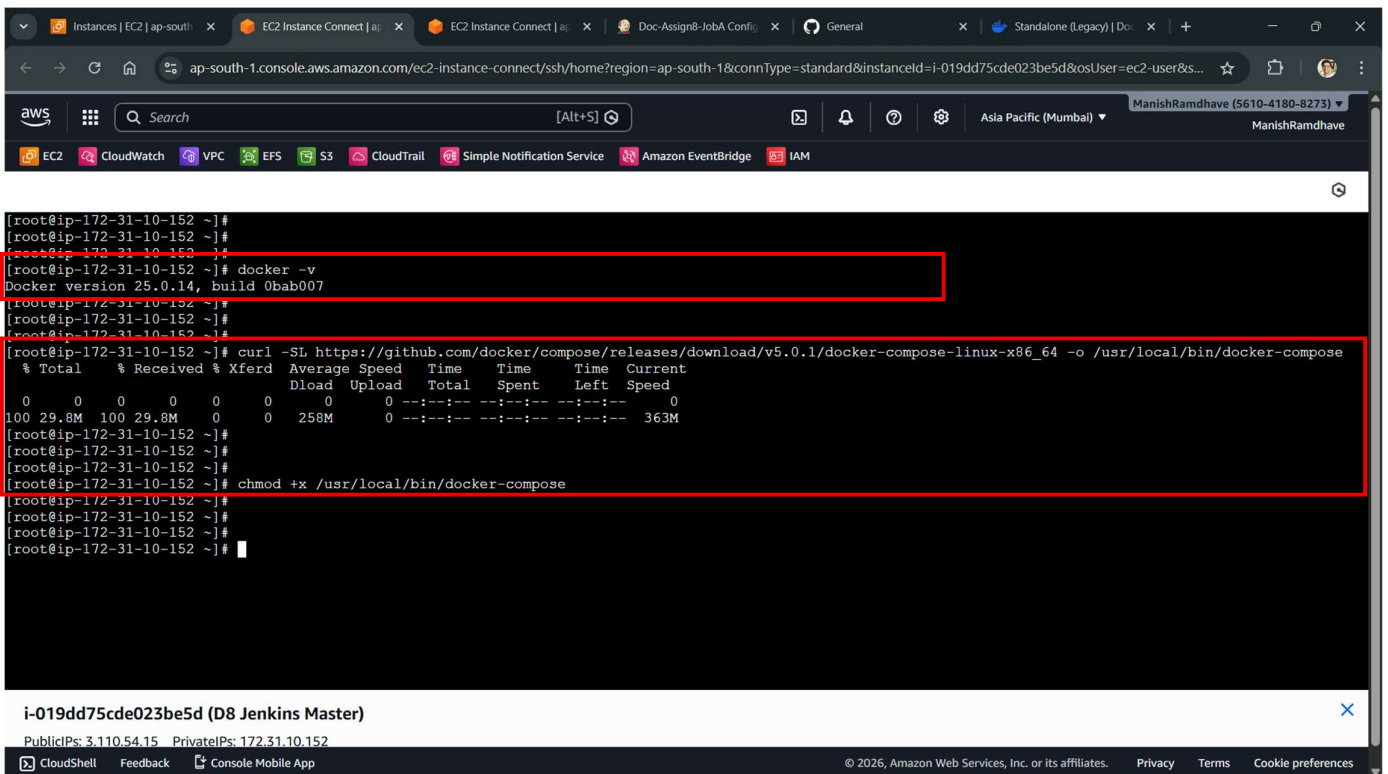


The screenshot shows the AWS Management Console 'Instances' page. Three instances are listed and highlighted with a red box:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
D7 Linux	i-09de41c0a4e55bf48	Stopped	t3.micro	-	View alarms +	ap-south-1a
D8 Jenkins Master	i-019dd75cde023be5d	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1b
D8 Slave	i-0c2453d39df282b03	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1b

Below the table, the 'Monitoring' section shows various metrics for the selected instances, including CPU utilization, network in/out bytes, and network packets in count.

Step 2: Installed Java-17 and Docker and Docker Compose on the Jenkins Master Instance and Slave also:



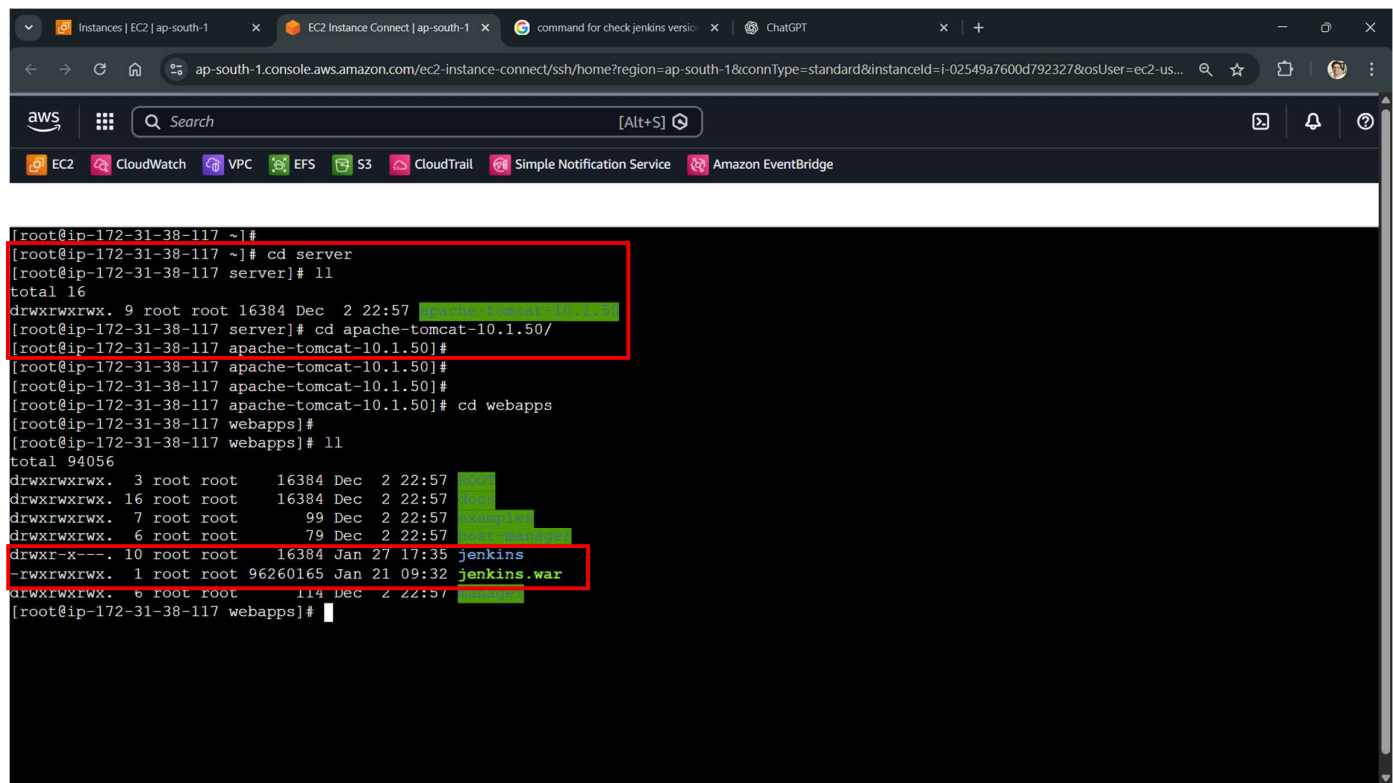
The screenshot shows the AWS CloudShell terminal with the following commands and output:

```
[root@ip-172-31-10-152 ~]# docker -v
Docker version 25.0.14, build 0bab007

[root@ip-172-31-10-152 ~]# curl -SL https://github.com/docker/compose/releases/download/v5.0.1/docker-compose-linux-x86_64 -o /usr/local/bin/docker-compose
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
100 29.8M 100 29.8M 0 0 258M 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[root@ip-172-31-10-152 ~]# chmod +x /usr/local/bin/docker-compose
```

The terminal output shows the successful installation of Docker and Docker Compose on the Jenkins Master Instance.

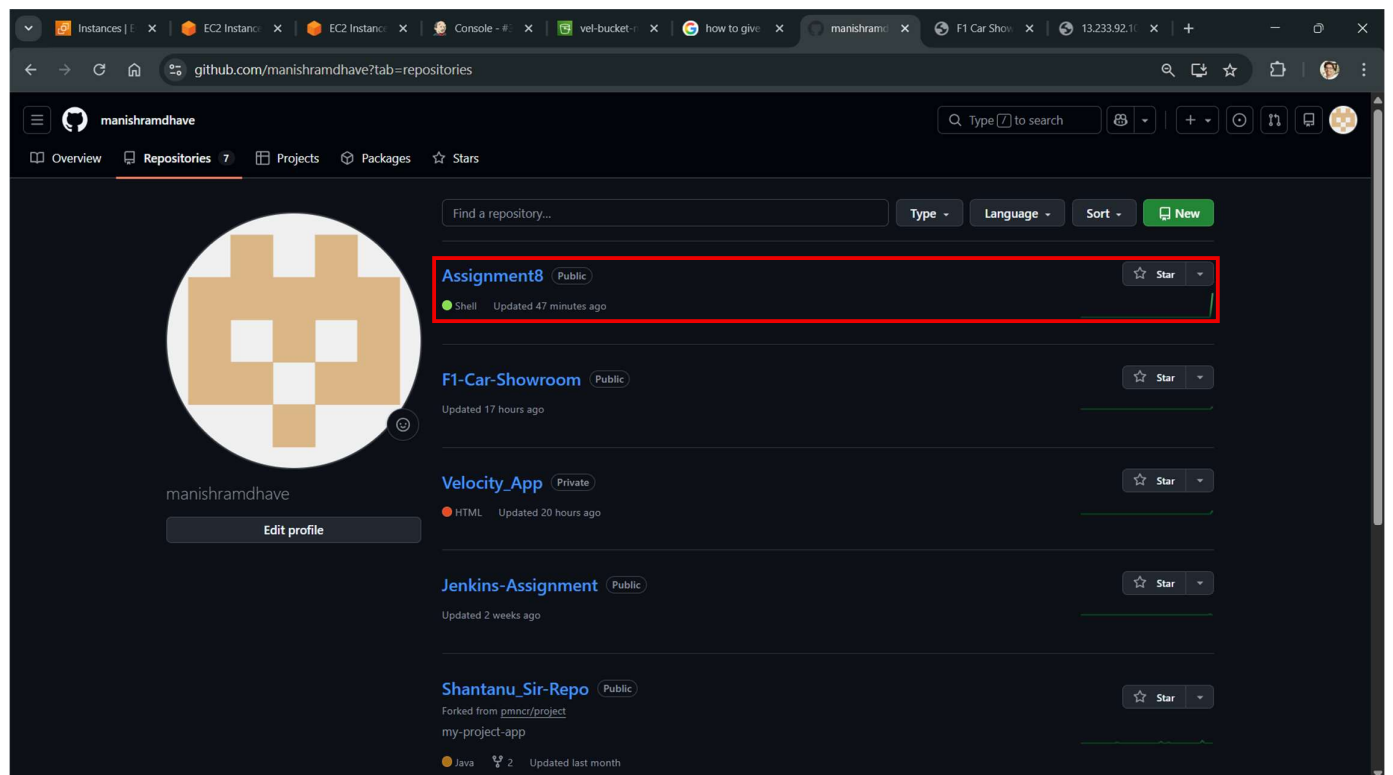
Step 3: Installed Apache-Tomcat-10 and Jenkins on the Jenkins Master Instance:



The screenshot shows the AWS Management Console with a terminal window open for an EC2 instance. The terminal output shows the user navigating through the file system to install Apache Tomcat and Jenkins. The following commands and their outputs are visible:

```
[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 server]# cd 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 jenkins  
drwxrwxrwx. 16 root root 16384 Dec 2 22:57 jenkins-war  
drwxrwxrwx. 7 root root 99 Dec 2 22:57 jenkins-war  
drwxrwxrwx. 6 root root 79 Dec 2 22:57 jenkins-war  
drwxr-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 jenkins-war  
[root@ip-172-31-38-117 webapps]#
```

Step 4: Made a Private Repository named 'Assignment8' in GitHub account:



Step 5: Created three branches, **2026Q1**, **2026Q2** and **2026Q3** in the ‘Velocity-App’ Repository and pushed **three different ‘Access.sh’, ‘Jenkinsfile’ and ‘docker-compose.yaml’** files in all the respective branches:

The image displays three screenshots of the GitHub repository interface for the 'Velocity-App' repository, showing the state of three different branches: 2026Q1, 2026Q2, and 2026Q3. Each screenshot shows the repository's file list with the following files: Access.sh, Jenkinsfile, and docker-compose.yaml. The 2026Q1 branch shows 72 commits, while 2026Q2 and 2026Q3 show 10 and 9 commits respectively. The 2026Q2 and 2026Q3 branches are 8 and 7 commits ahead of the 2026Q1 branch, respectively.

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:

The image shows a screenshot of the Jenkins 'Manage Jenkins' page, specifically the 'System' tab. The 'GitHub Servers' section is visible, showing a configuration for a GitHub server. The configuration includes the following fields:

- Name:** GitHub-Server
- API URL:** https://api.github.com
- Credentials:** Git (selected from a dropdown menu)
- Manage hooks:** ☒
- Advanced:** (dropdown menu)

A red box highlights the 'Credentials' section, which is the focus of the step. The 'Test connection' button is also visible. At the bottom of the page, there are 'Save' and 'Apply' buttons.

Step 7: Launched the **Jenkins** and created three different **Pipeline Jobs** in it:

Instances |

EC2 Instance

EC2 Instance

Dashboard

vel-bucket

how to give

manishram

F1 Car Show

13.233.92.1

← → ↻ ↗

Not secure 13.201.88.8:8080/jenkins/

🔍 ☆ 📄 📌

Jenkins

+ New Item

Build History

Project Relationship

Check File Fingerprint

Build Queue

No builds in the queue.

Build Executor Status

Built-In Node

0/2

Slave

0/10

All

+

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☁	Doc-Assign8-Pipeline-A	1 hr 55 min #48	1 hr 55 min #47	22 sec ▶
✓	☀	Doc-Assign8-Pipeline-B	1 hr 46 min #3	N/A	24 sec ▶
✓	☁	Doc-Assign8-Pipeline-C	41 sec #2	16 hr #1	33 sec ▶

Icon: S M L

REST API

Jenkins 2.541.1

Step 8: Created an ‘Node’ Connection between Jenkins Master and Slave instances ‘Manage Jenkins’ by creating a **SSH Username and Key (Credential)** using a Key-Pair and **Manually trusted key-verification Strategy**:

Instances |

EC2 Instance

EC2 Instance

Nodes - Ma

vel-bucket

how to give

manishram

F1 Car Show

13.233.92.1

← → ↻ ↗

Not secure 15.206.82.99:8080/jenkins/manage/computer/

☆ 📄 📌

Jenkins / Manage Jenkins

Nodes

+ New Node

Configure Monitors

↻

Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
🖥	Built-In Node	Linux (amd64)	In sync	4.70 GiB	0 B	4.70 GiB	0ms
🖥	Slave	Linux (amd64)	In sync	5.09 GiB	0 B	455.69 MiB	33ms
last checked		42 min	42 min	42 min	42 min	42 min	42 min

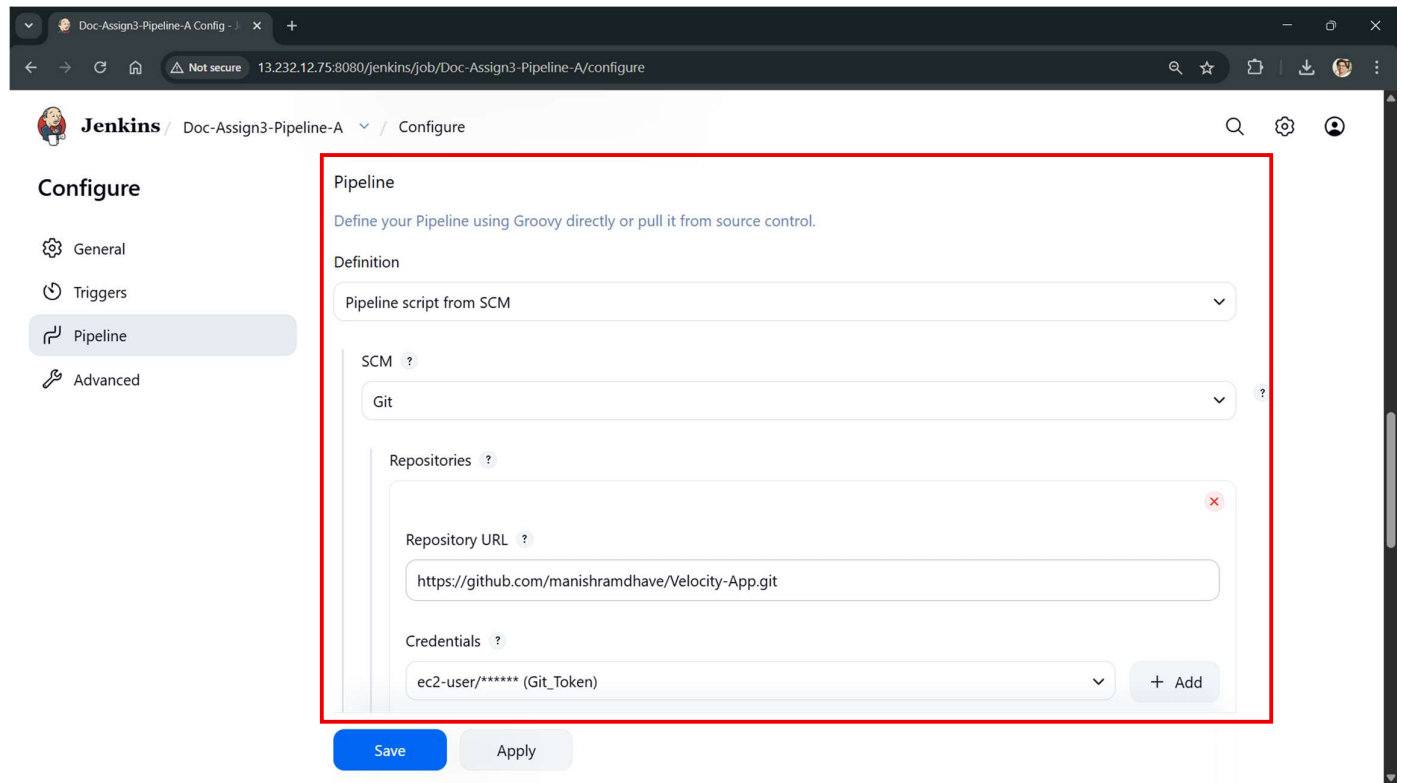
Icon: S M L

Legend

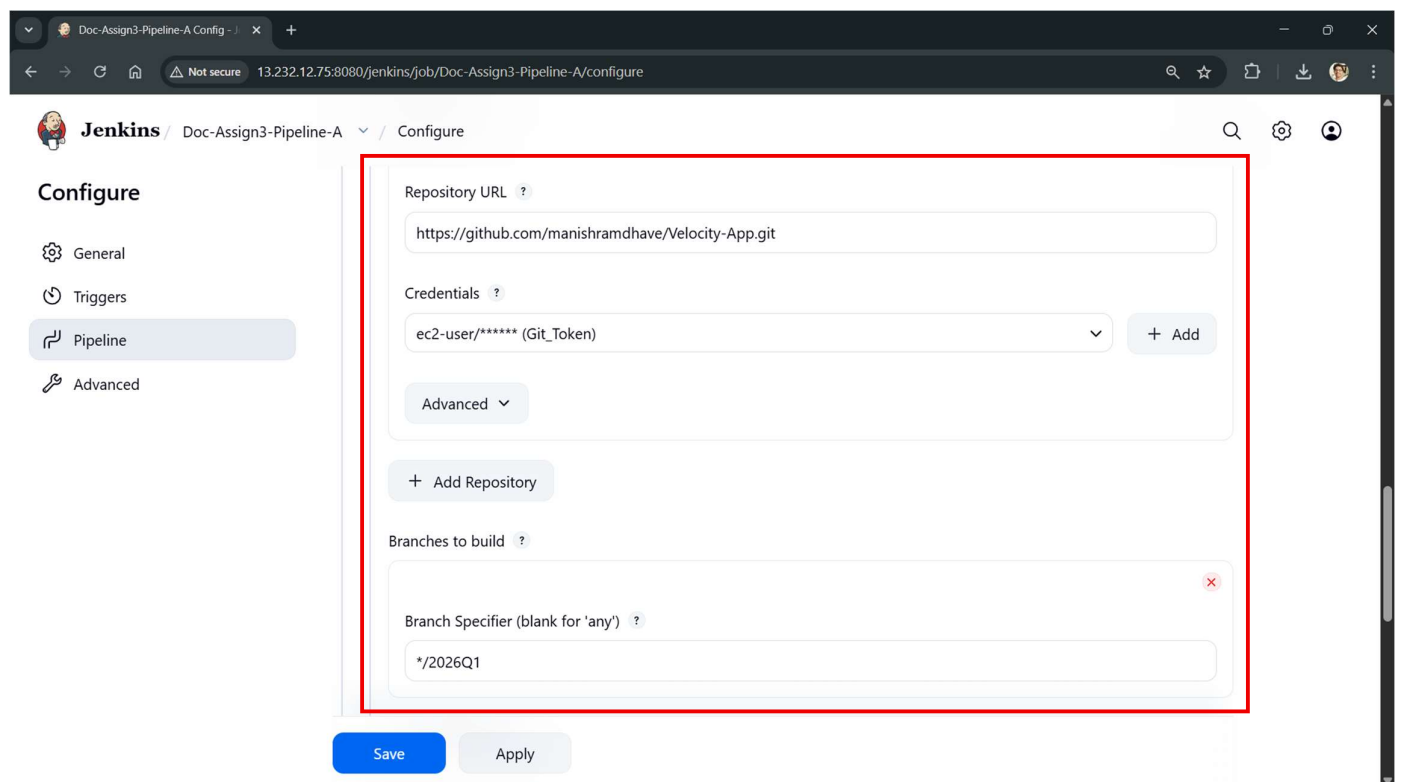
REST API

Jenkins 2.541.1

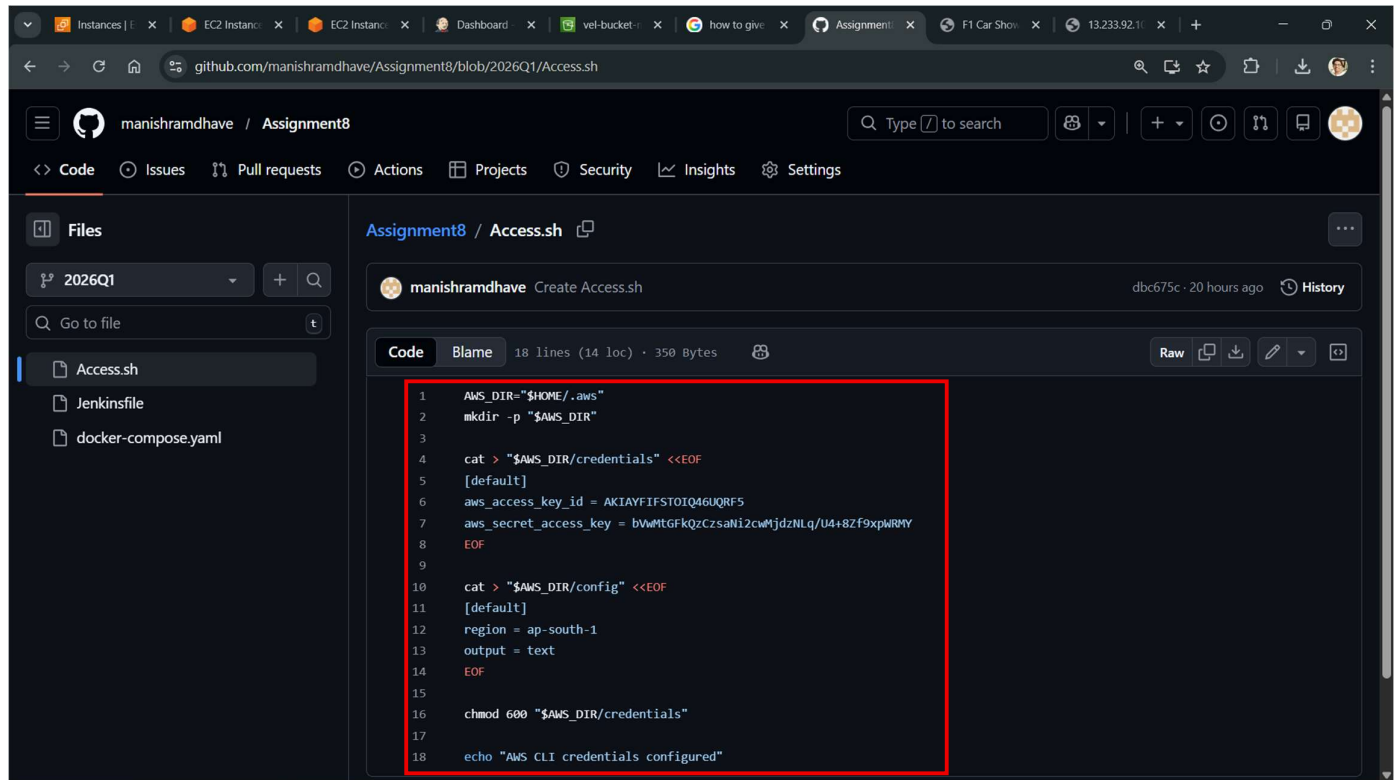
Step 9: Used **‘Pipeline script from SCM’** and selected the SCM as a **‘Git’** which will follow the Jenkins pipeline script by using the **‘Jenkinsfile’** in the repository **in each pipeline job**:



Step 10: Integrated all Git branches, **2026Q1**, **2026Q2** and **2026Q3** with Jenkins by creating **‘Credentials’** by using the Git Token **on all the pipeline** respectively:



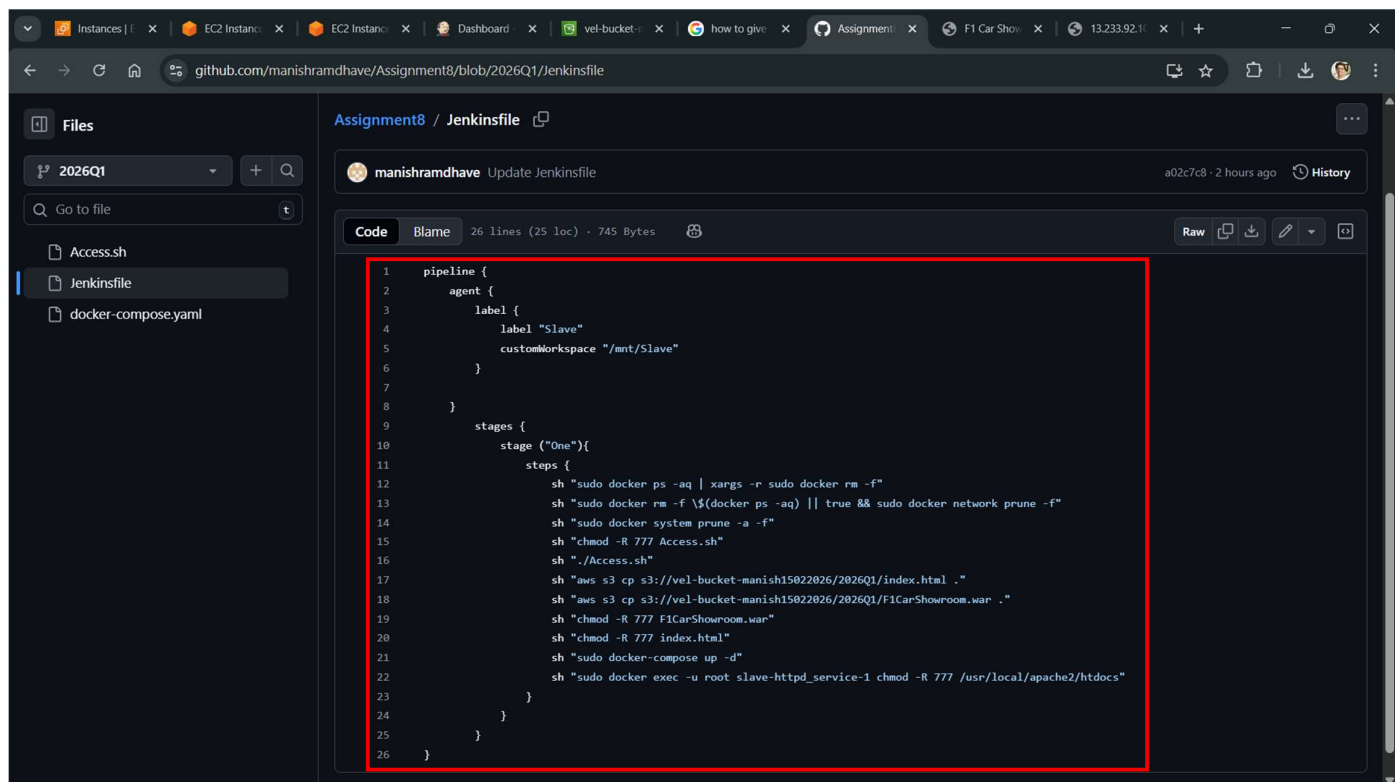
Step 11: Added AWS Configurations (i.e. Access Key, Secret Key etc.) in the 'Access.sh' file which run by Jenkinsfile for AWS CLI command processes in each Branch:



The screenshot shows a GitHub repository page for 'manishramdhare / Assignment8'. The file 'Access.sh' is selected, showing its content. The file is 18 lines long, 14 lines of code, and 350 bytes. The content of the file is as follows:

```
1 AWS_DIR="$HOME/.aws"
2 mkdir -p "$AWS_DIR"
3
4 cat > "$AWS_DIR/credentials" <<EOF
5 [default]
6 aws_access_key_id = AKIAVF1FSTOIQ46UQRF5
7 aws_secret_access_key = bVwMtGfKQzCzsaNi2cwMjdZMLq/U4+8Zf9xpWRWY
8 EOF
9
10 cat > "$AWS_DIR/config" <<EOF
11 [default]
12 region = ap-south-1
13 output = text
14 EOF
15
16 chmod 600 "$AWS_DIR/credentials"
17
18 echo "AWS CLI credentials configured"
```

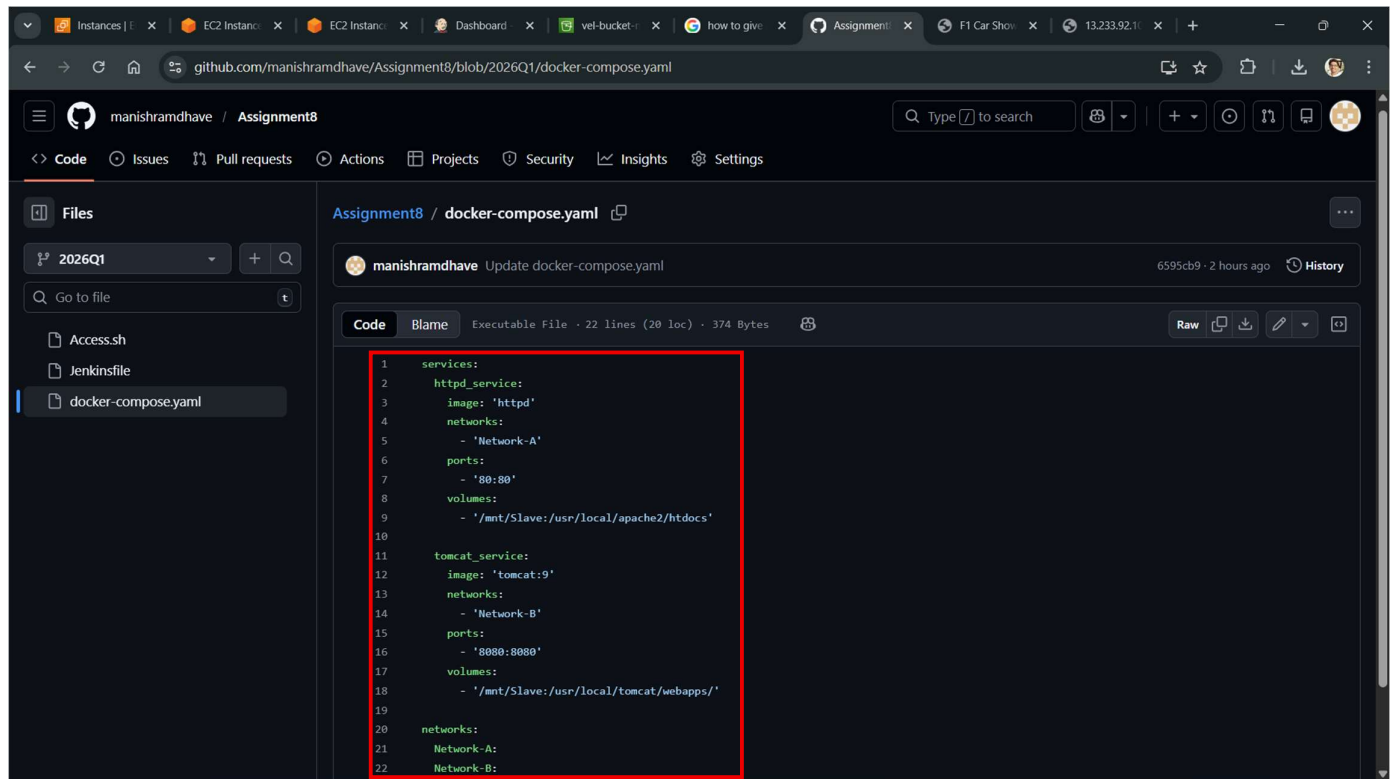
Step 12: Added AWS Configurations (i.e. Access Key, Secret Key etc.) in the 'Access.sh' file which run by Jenkinsfile for AWS CLI command processes:



The screenshot shows a GitHub repository page for 'manishramdhare / Assignment8'. The file 'Jenkinsfile' is selected, showing its content. The file is 26 lines long, 25 lines of code, and 745 bytes. The content of the file is as follows:

```
1 pipeline {
2   agent {
3     label {
4       label "Slave"
5       customWorkspace "/mnt/Slave"
6     }
7   }
8   stages {
9     stage ("One"){
10      steps {
11        sh "sudo docker ps -aq | xargs -r sudo docker rm -f"
12        sh "sudo docker rm -f \$(docker ps -aq) || true && sudo docker network prune -f"
13        sh "sudo docker system prune -a -f"
14        sh "chmod -R 777 Access.sh"
15        sh ". /Access.sh"
16        sh "aws s3 cp s3://vel-bucket-manish15022026/2026Q1/index.html ."
17        sh "aws s3 cp s3://vel-bucket-manish15022026/2026Q1/F1CarShowroom.war ."
18        sh "chmod -R 777 F1CarShowroom.war"
19        sh "chmod -R 777 index.html"
20        sh "sudo docker-compose up -d"
21        sh "sudo docker exec -u root slave-httpd_service-1 chmod -R 777 /usr/local/apache2/htdocs"
22      }
23    }
24  }
25 }
26 }
```

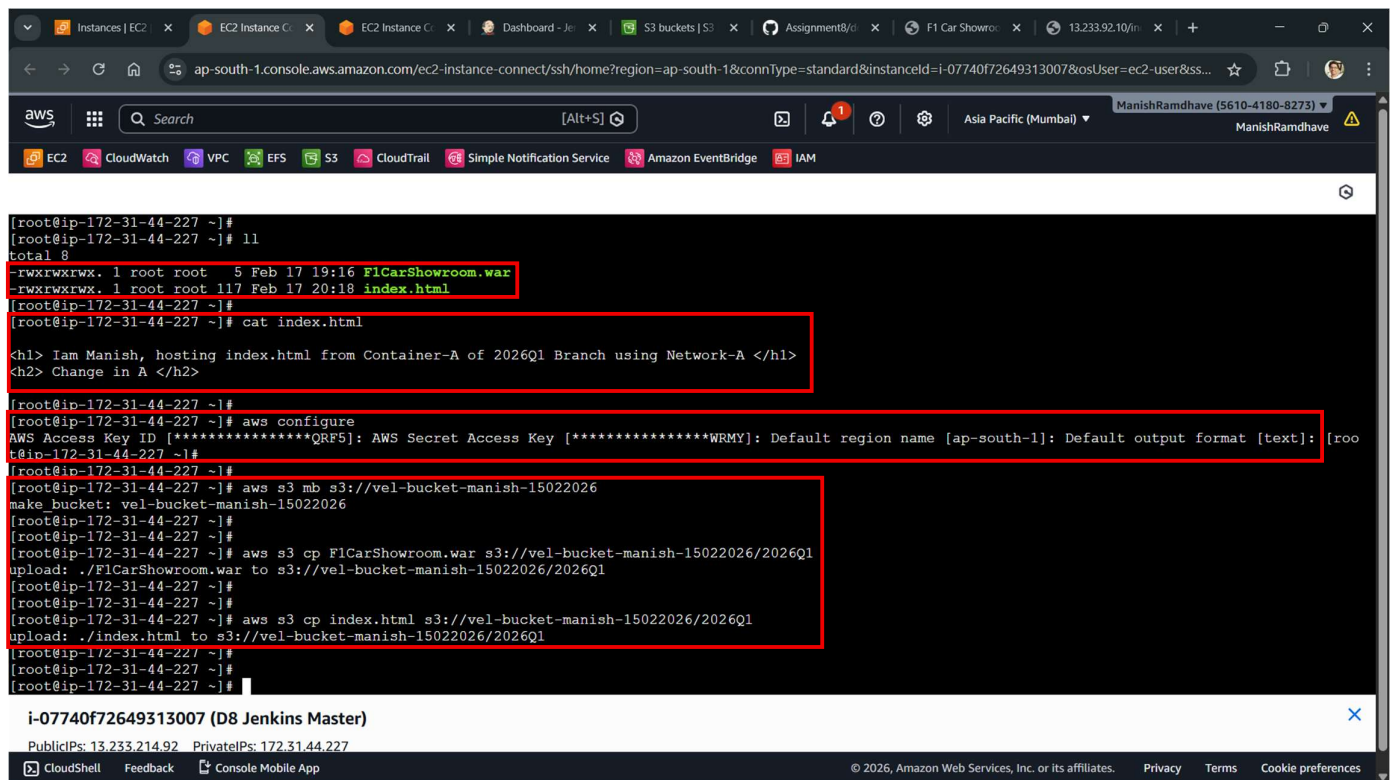

Step 13: Added '**docker-compose.yaml**' file which will run by Jenkinsfile for Container creation in each branch:



The screenshot shows a GitHub repository named 'Assignment8' by user 'manishramdhav'. The file 'docker-compose.yaml' is selected, showing its content. The file defines two services: 'httpd_service' and 'tomcat_service', each with specific images, networks, ports, and volumes. The file is 22 lines long and 374 bytes.

```
1 services:
2   httpd_service:
3     image: 'httpd'
4     networks:
5       - 'Network-A'
6     ports:
7       - '80:80'
8     volumes:
9       - '/mnt/Slave:/usr/local/apache2/htdocs'
10
11   tomcat_service:
12     image: 'tomcat:9'
13     networks:
14       - 'Network-B'
15     ports:
16       - '8080:8080'
17     volumes:
18       - '/mnt/Slave:/usr/local/tomcat/webapps/'
19
20 networks:
21   Network-A:
22   Network-B:
```

Step 14: By using the AWS CLI commands, we have uploaded the '**F1CarShowroom.war**' and '**index.html**' files in the '**vel-bucket-manish15022026/2026Q1**' S3 Bucket folder:



The screenshot shows a terminal window with AWS CLI commands and their output. The commands include listing files, viewing the content of index.html, configuring AWS CLI, creating an S3 bucket, and uploading files to the bucket.

```
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]# ll
total 8
-rwxrwxrwx. 1 root root 5 Feb 17 19:16 F1CarShowroom.war
-rwxrwxrwx. 1 root root 117 Feb 17 20:18 index.html
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]# cat index.html

<h1> Iam Manish, hosting index.html from Container-A of 2026Q1 Branch using Network-A </h1>
<h2> Change in A </h2>

[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]# aws configure
AWS Access Key ID [*****QRF5]: AWS Secret Access Key [*****WRMY]: Default region name [ap-south-1]: Default output format [text]: [roo
t@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]# aws s3 mb s3://vel-bucket-manish-15022026
make_bucket: vel-bucket-manish-15022026
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]# aws s3 cp F1CarShowroom.war s3://vel-bucket-manish-15022026/2026Q1
upload: ./F1CarShowroom.war to s3://vel-bucket-manish-15022026/2026Q1
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]# aws s3 cp index.html s3://vel-bucket-manish-15022026/2026Q1
upload: ./index.html to s3://vel-bucket-manish-15022026/2026Q1
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]#
[root@ip-172-31-44-227 ~]#
```

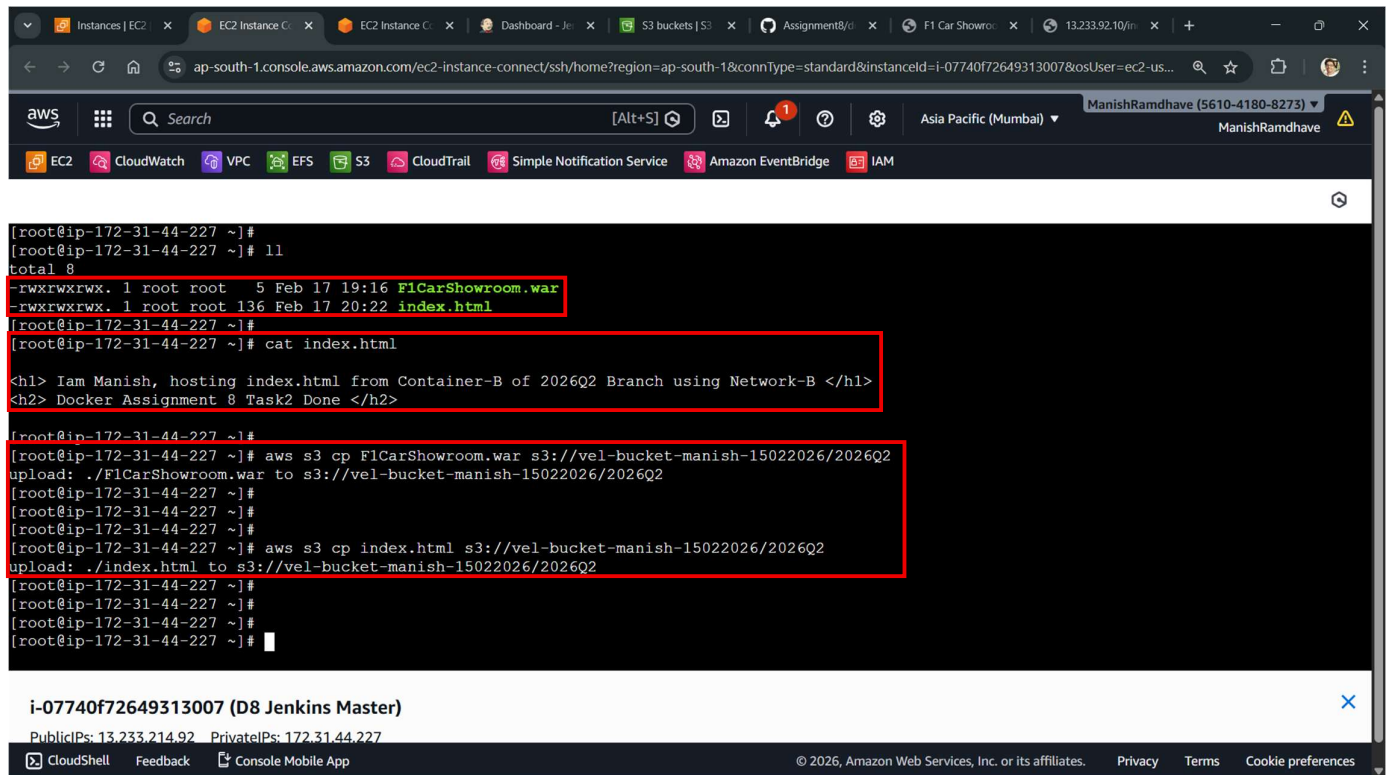
i-07740f72649313007 (D8 Jenkins Master)

PublicIPs: 13.233.214.92 PrivateIPs: 172.31.44.227

CloudShell Feedback Console Mobile App

© 2026, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 15: By using the AWS CLI commands, we have uploaded the 'F1CarShowroom.war' and 'index.html' files in the 'vel-bucket-manish15022026/2026Q1' S3 Bucket folder:

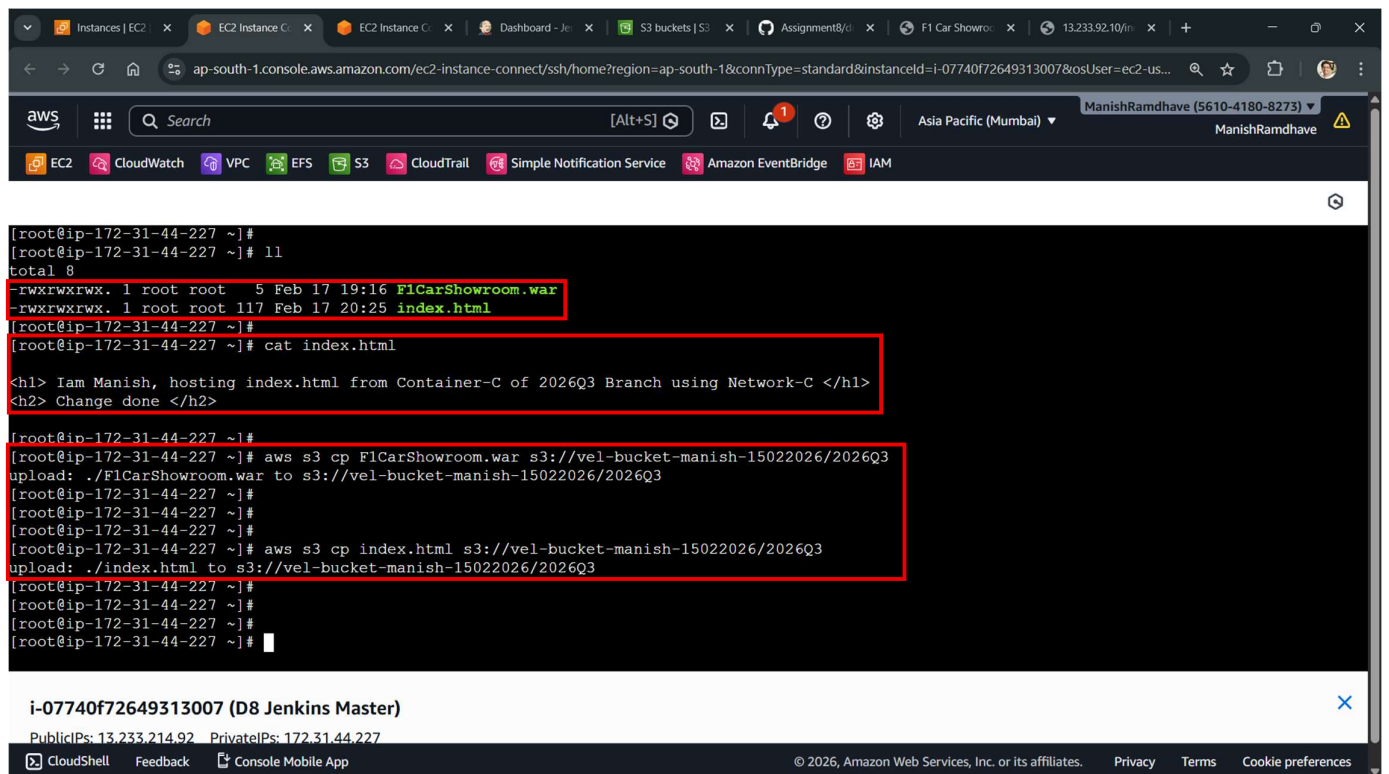


The screenshot shows the AWS CloudShell interface for an EC2 instance. The terminal output is as follows:

```
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# ll  
total 8  
-rw-rw-rw-. 1 root root 5 Feb 17 19:16 F1CarShowroom.war  
-rw-rw-rw-. 1 root root 136 Feb 17 20:22 index.html  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# cat index.html  
  
<h1> Iam Manish, hosting index.html from Container-B of 2026Q2 Branch using Network-B </h1>  
<h2> Docker Assignment 8 Task2 Done </h2>  
  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# aws s3 cp F1CarShowroom.war s3://vel-bucket-manish-15022026/2026Q2  
upload: ./F1CarShowroom.war to s3://vel-bucket-manish-15022026/2026Q2  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# aws s3 cp index.html s3://vel-bucket-manish-15022026/2026Q2  
upload: ./index.html to s3://vel-bucket-manish-15022026/2026Q2  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#
```

Below the terminal, the instance details for 'i-07740f72649313007 (D8 Jenkins Master)' are visible, including public and private IP addresses.

Step 16: By using the AWS CLI commands, we have uploaded the 'F1CarShowroom.war' and 'index.html' files in the 'vel-bucket-manish15022026/2026Q1' S3 Bucket folder:



The screenshot shows the AWS CloudShell interface for an EC2 instance. The terminal output is as follows:

```
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# ll  
total 8  
-rw-rw-rw-. 1 root root 5 Feb 17 19:16 F1CarShowroom.war  
-rw-rw-rw-. 1 root root 117 Feb 17 20:25 index.html  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# cat index.html  
  
<h1> Iam Manish, hosting index.html from Container-C of 2026Q3 Branch using Network-C </h1>  
<h2> Change done </h2>  
  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# aws s3 cp F1CarShowroom.war s3://vel-bucket-manish-15022026/2026Q3  
upload: ./F1CarShowroom.war to s3://vel-bucket-manish-15022026/2026Q3  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]# aws s3 cp index.html s3://vel-bucket-manish-15022026/2026Q3  
upload: ./index.html to s3://vel-bucket-manish-15022026/2026Q3  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#  
[root@ip-172-31-44-227 ~]#
```

Below the terminal, the instance details for 'i-07740f72649313007 (D8 Jenkins Master)' are visible, including public and private IP addresses.

Results:

1. When **Build** is done in by 'Doc-Assign8-Pipeline-A', the 'F1CarShowroom' application and 'index.html' file is hosted from the container of 'Network-A' by following the 'docker-compose.yaml' script in the GitHub Repository of the 2026Q1 Branch by using Port No.80:

The screenshot displays the Jenkins interface for build #57 of 'Doc-Assign8-Pipeline-A' on 17 Feb 2026 at 20:43:30. The build status is successful (green checkmark) and was started by user 'mmm'. Below the Jenkins interface, two browser windows are shown. The left window, at '13.233.92.10/index.html', displays the text 'Hosting index.html from Container-A of 2026Q1 Branch using Network-A' and 'Change in A'. The right window, at '13.233.92.10:8080/F1CarShowroom/', shows the 'F1 Car Showroom' application with a red header, navigation links (Home, Cars, Teams, Contact), and a main image of Formula 1 cars on a wet track with the text 'Speed | Power | Precision' and 'Featured Formula 1 Cars' at the bottom.

2. When **Build** is done in by 'Doc-Assign8-Pipeline-B', the 'F1CarShowroom' application and 'index.html' file is hosted from the container of 'Network-B' by following the 'docker-compose.yaml' script in the GitHub Repository of the 2026Q2 Branch by using Port No.80:

The screenshot displays the Jenkins interface for build #5 of 'Doc-Assign8-Pipeline-B' on 17 Feb 2026 at 20:47:16. The build status is successful (green checkmark) and was started by user 'mmm'. Below the Jenkins interface, two browser windows are shown. The left window, at '13.233.92.10/index.html', displays the text 'Hosting index.html from Container-B of 2026Q2 Branch using Network-B' and 'Done with Docker Assignment 8'. The right window, at '13.233.92.10:8080/F1CarShowroom/', shows the 'F1 Car Showroom' application with a red header, navigation links (Home, Cars, Teams, Contact), and a main image of Formula 1 cars on a wet track with the text 'Speed | Power | Precision' and 'Featured Formula 1 Cars' at the bottom.

3. When **Build** is done in by **‘Doc-Assign8-Pipeline-C’**, the **‘F1CarShowroom’** application and **‘index.html’** file is hosted from the **container** of **‘Network-A’** by following the **‘docker-compose.yml’** script in the **GitHub Repository** of the **2026Q3 Branch** by using **Port No.80**:

The image shows a Jenkins build interface and two web browser windows. The Jenkins window at the top displays the build status for 'Doc-Assign8-Pipeline-C' #7, which is successful (green checkmark) and started 7.9 seconds ago. Below the Jenkins window, there are two browser windows. The left browser window shows the 'index.html' file hosted from Container-C of the 2026Q3 Branch, with the text 'Done with Docker Assignment 8'. The right browser window shows the 'F1 Car Showroom' application, which features a red header, navigation links (Home, Cars, Teams, Contact), and a large image of Formula 1 cars on a wet track.

Jenkins build status for **Doc-Assign8-Pipeline-C** #7 (17 Feb 2026, 20:49:11). The build is successful and started 7.9 seconds ago.

Left browser window: **13.233.92.10/index.html**. Content: **Hosting index.html from Container-C of 2026Q3 Branch using Network-C**. Done with Docker Assignment 8.

Right browser window: **13.233.92.10:8080/F1CarShowroom/**. Content: **F1 Car Showroom**. Ultimate Formula 1 Experience. Speed | Power | Precision. Featured Formula 1 Cars.