

Jenkins Assignment 4

Step 1: Launched four instances, Jenkins Master, Slave1, Slave2 and Slave3

The screenshot shows the AWS EC2 Instances page with the following details:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Jenkins Master	i-03bfc0e5ce6262048	Running	t3.micro	3/3 checks passed	View alarms	ap-south-1a	ec2-43-205-210-
Slave1	i-07c71e056e325569f	Running	t3.micro	3/3 checks passed	View alarms	ap-south-1a	ec2-52-66-184-1
Slave2	i-0bc1750315c74562f	Running	t3.micro	3/3 checks passed	View alarms	ap-south-1a	ec2-65-2-183-9.a
Slave3	i-06eeb09f151a1ed2f	Running	t3.micro	3/3 checks passed	View alarms	ap-south-1a	ec2-13-201-56-2

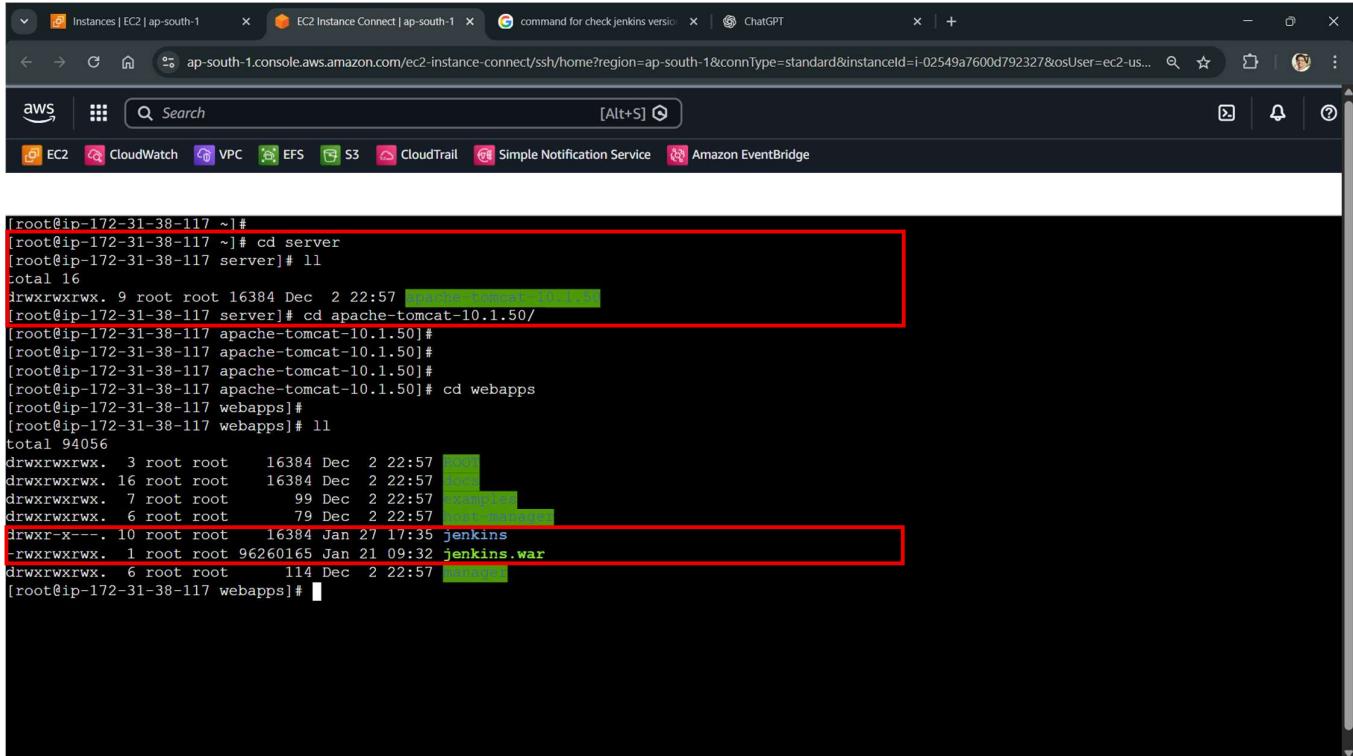
Below the table, it says "5 instances selected". Under the "Monitoring" section, there are four charts showing CPU utilization (%), Network in (bytes), Network out (bytes), and Network packets in (count) over a 1-hour period.

Step 2: Installed Java -17 and Apache-HTTPD on all the instance:

```
[root@ip-172-31-36-242 ~]# java -version
openjdk version "17.0.17" 2025-10-21 LTS
OpenJDK Runtime Environment Corretto-17.0.17.10.1 (build 17.0.17+10-LTS)
OpenJDK 64-Bit Server VM Corretto-17.0.17.10.1 (build 17.0.17+10-LTS, mixed mode, sharing)

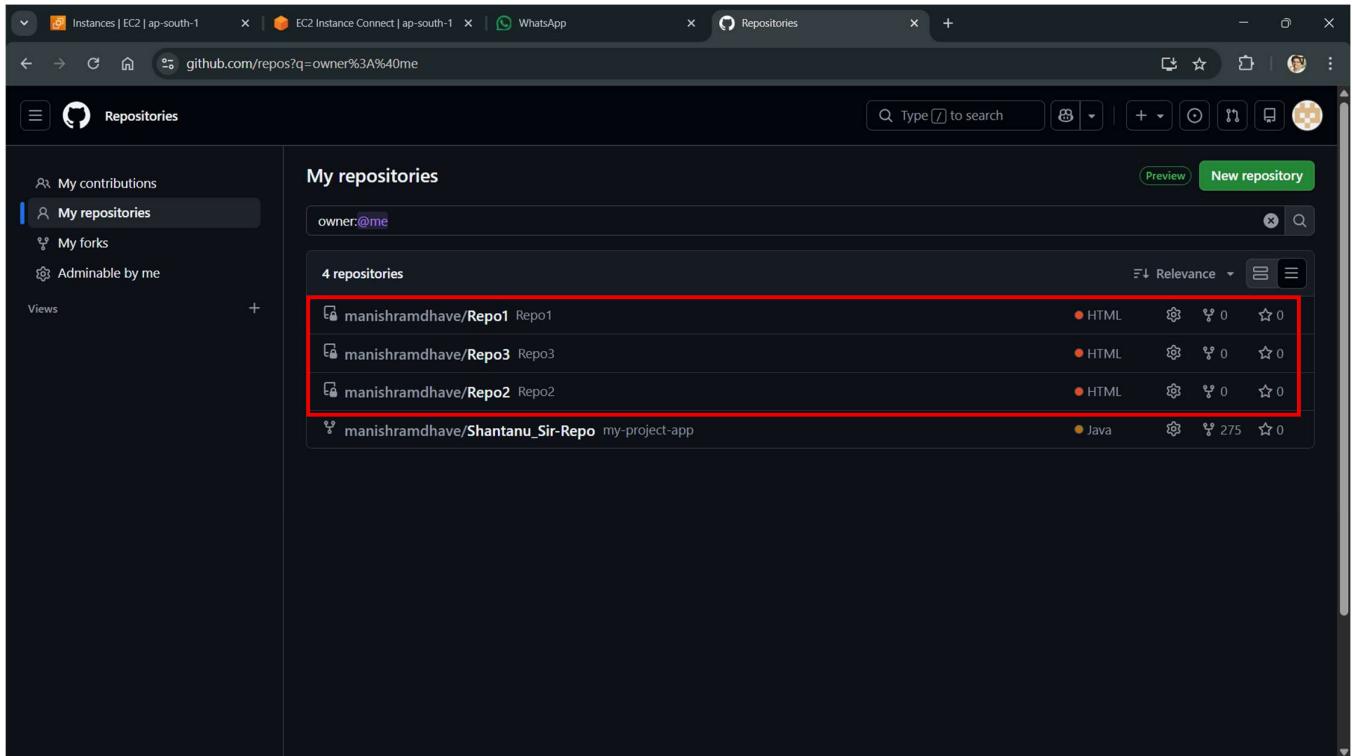
[root@ip-172-31-36-242 ~]# service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
    Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
      Active: inactive (dead)
        Docs: man:httpd.service(8)
[root@ip-172-31-36-242 ~]#
```

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



```
[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 .
drwxrwxrwx. 16 root root 16384 Dec  2 22:57 ..
drwxrwxrwx. 7 root root 99 Dec  2 22:57 apache-tomcat-10.1.50
drwxrwxrwx. 6 root root 79 Dec  2 22:57 examples
drwxrwxrwx. 3 root root 16384 Dec  2 22:57 host-manager
drwxr-x--. 10 root root 16384 Jan 27 17:35 jenkins
-rw-rw-rwx. 1 root root 96260165 Jan 21 09:32 jenkins.war
drwxrwxrwx. 6 root root 114 Dec  2 22:57 test
[root@ip-172-31-38-117 webapps]# ll
total 94056
```

Step 4: Made three Private Repositories Repo1, Repo2 and Repo3 in GitHub account:



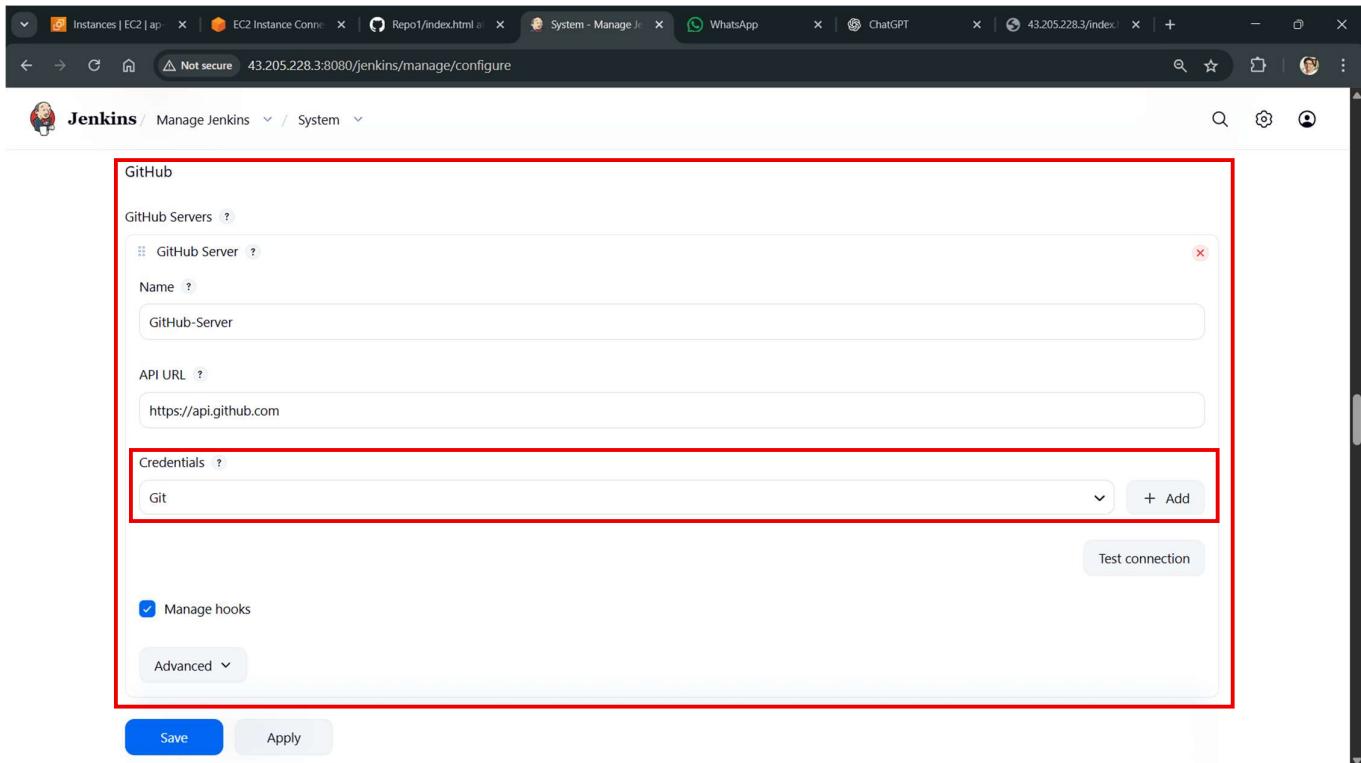
My repositories

Owner	Name	Type	HTML	Issues	Stars
manishramdhave	Repo1	Repo1	HTML	0	0
manishramdhave	Repo3	Repo3	HTML	0	0
manishramdhave	Repo2	Repo2	HTML	0	0
manishramdhave	Shantanu_Sir-Repo	my-project-app	Java	275	0

Step 5: Pushed three different index.html files on repositories Repo1, Repo2 and Repo3 respectively:

```
[root@ip-172-31-36-242 mnt]# [root@ip-172-31-36-242 mnt]# cd R1 [root@ip-172-31-36-242 R1]# git log commit 74f243a21055fb5cd889d73f2903c910a9832c30 (HEAD -> master, origin/master) Author: root <root@ip-172-31-36-242.ap-south-1.compute.internal> Date: Tue Feb 3 17:57:15 2026 +0000 1 [root@ip-172-31-36-242 R1]# cd .. [root@ip-172-31-36-242 mnt]# cd R2 [root@ip-172-31-36-242 R2]# git log commit 702d7ae5c18ea78ce288287e6ef747ffbdal555 (HEAD -> master, origin/master) Author: root <root@ip-172-31-36-242.ap-south-1.compute.internal> Date: Tue Feb 3 17:59:35 2026 +0000 2 [root@ip-172-31-36-242 R2]# cd .. [root@ip-172-31-36-242 mnt]# cd R3 [root@ip-172-31-36-242 R3]# git log commit 3aa8c34f368e9dc5a5cc47b8651d6274dbe2762b (HEAD -> master, origin/master) Author: root <root@ip-172-31-36-242.ap-south-1.compute.internal> Date: Tue Feb 3 18:00:38 2026 +0000 3 [root@ip-172-31-36-242 R3]#
```

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 every GitHub Repositories by using ‘Git Webhooks’ (i.e. setting up Jenkins URL in Git) :

The screenshot shows the GitHub repository settings page for 'manishramdhave / Repo1'. The 'Webhooks' section is selected in the sidebar. A red box highlights the 'Manage webhook' form. The 'Payload URL' field contains the Jenkins URL: `http://43.205.228.3:8080/jenkins/github-webhook/`. The 'Content type' is set to 'application/json'. The 'Secret' field is empty. Under 'SSL verification', 'Enable SSL verification' is selected. In the 'Which events would you like to trigger this webhook?' section, 'Just the push event.' is selected. The 'Active' checkbox is checked. At the bottom are 'Update webhook' and 'Delete webhook' buttons.

Step 8: Created a ‘Username and Password’ Credential for accessing the Private Git Repositories using the GitHub Token:

The screenshot shows the Jenkins 'Global credentials (unrestricted)' page. A red box highlights the 'Git/***** (Token)' credential entry. The token name is 'Git/***** (Token)'. The 'Usage' section shows three Jenkins jobs using this token: Job1, Job2, and Job3. The usage details are as follows:

Job1	#1 - #29	#31 - #38				
Job2	#1	- #5	#9 - #12	#15	#20	#21
Job3	#4-#8					

Step 9: Created a ‘SSH Username with private key’ Credential for Jenkins Master and Slave Connections (One Credential is enough for all the Slaves):

The screenshot shows the Jenkins 'Update credentials' page. A red box highlights the main input fields. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'ID' field contains 'Slave'. The 'Description' field contains 'Slave'. The 'Username' field contains 'ec2-user'. An unchecked checkbox 'Treat username as secret' is present. The 'Private Key' section has a radio button 'Enter directly' selected, and a text area labeled 'Key' contains the text 'Concealed for Confidentiality'. A 'Replace' button is visible next to the key area. A 'Save' button is at the bottom.

Step 10: Created three Nodes for to establish the connection between Jenkins Master and Slave Instances using the ‘Credential’ and ‘Manually trusted key Verification Strategy’:

The screenshot shows the Jenkins 'Nodes' page. A red box highlights the list of slave nodes. The table has columns: S, Name, Architecture, Clock Difference, Free Disk Space, Free Swap Space, Free Temp Space, and Response Time. The nodes listed are: Built-In Node (Architecture: Linux (amd64), Status: In sync, Free Disk Space: 5.64 GiB, Free Swap Space: 0 B, Free Temp Space: 5.64 GiB, Response Time: 0ms), Slave1 (Architecture: Linux (amd64), Status: In sync, Free Disk Space: 6.02 GiB, Free Swap Space: 0 B, Free Temp Space: 458.39 MiB, Response Time: 90ms), Slave2 (Architecture: Linux (amd64), Status: In sync, Free Disk Space: 6.02 GiB, Free Swap Space: 0 B, Free Temp Space: 458.38 MiB, Response Time: 71ms), and Slave3 (Architecture: Linux (amd64), Status: In sync, Free Disk Space: 6.03 GiB, Free Swap Space: 0 B, Free Temp Space: 458.26 MiB, Response Time: 51ms). The 'last checked' column shows values like 44 min for all nodes. A legend at the bottom indicates icons for Status (S), Maintenance (M), and Last checked (L).

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
💻	Built-In Node	Linux (amd64)	In sync	5.64 GiB	0 B	5.64 GiB	0ms
💻	Slave1	Linux (amd64)	In sync	6.02 GiB	0 B	458.39 MiB	90ms
💻	Slave2	Linux (amd64)	In sync	6.02 GiB	0 B	458.38 MiB	71ms
💻	Slave3	Linux (amd64)	In sync	6.03 GiB	0 B	458.26 MiB	51ms

Step 11: Launched the Jenkins and created three different Jobs Job1, Job2 and Job3:

The screenshot shows the Jenkins dashboard at 13.201.1.129:8080/jenkins. The main content area displays a table of jobs with the following data:

S	W	Name	Last Success	Last Failure	Last Duration
		Job1	1 hr 20 min #2	N/A	2.3 sec
		Job2	1 hr 30 min #2	N/A	2.3 sec
		Job3	1 hr 17 min #1	N/A	2.2 sec

Below the table, there are filter buttons for 'All' and '+', and a status indicator 'Icon: S M L'. At the bottom right, there are links for 'REST API' and 'Jenkins 2.541.1'.

Step 12: Integrated Git with Jenkins by creating ‘Credentials’ by using the Git Token:

The screenshot shows the configuration page for 'Job1' at 3.110.132.215:8080/jenkins/job/Job1/configure. The left sidebar includes options like General, Source Code Management, Triggers, Environment, Build Steps, and Post-build Actions. The 'Source Code Management' section is highlighted with a red box and contains the following details:

- Source Code Management: 'Git' is selected.
- Repositories:
 - Repository URL: <https://github.com/manishramdhave/Repo1.git>
 - Credentials: 'Git-token/***** (Git-token)' is listed.

At the bottom, there are 'Save' and 'Apply' buttons.

Step 13: Enabled ‘GitHub hook Trigger for GITScm polling’ while configuring all three Jenkins Jobs and also enabled ‘Delete workspace before build starts’:

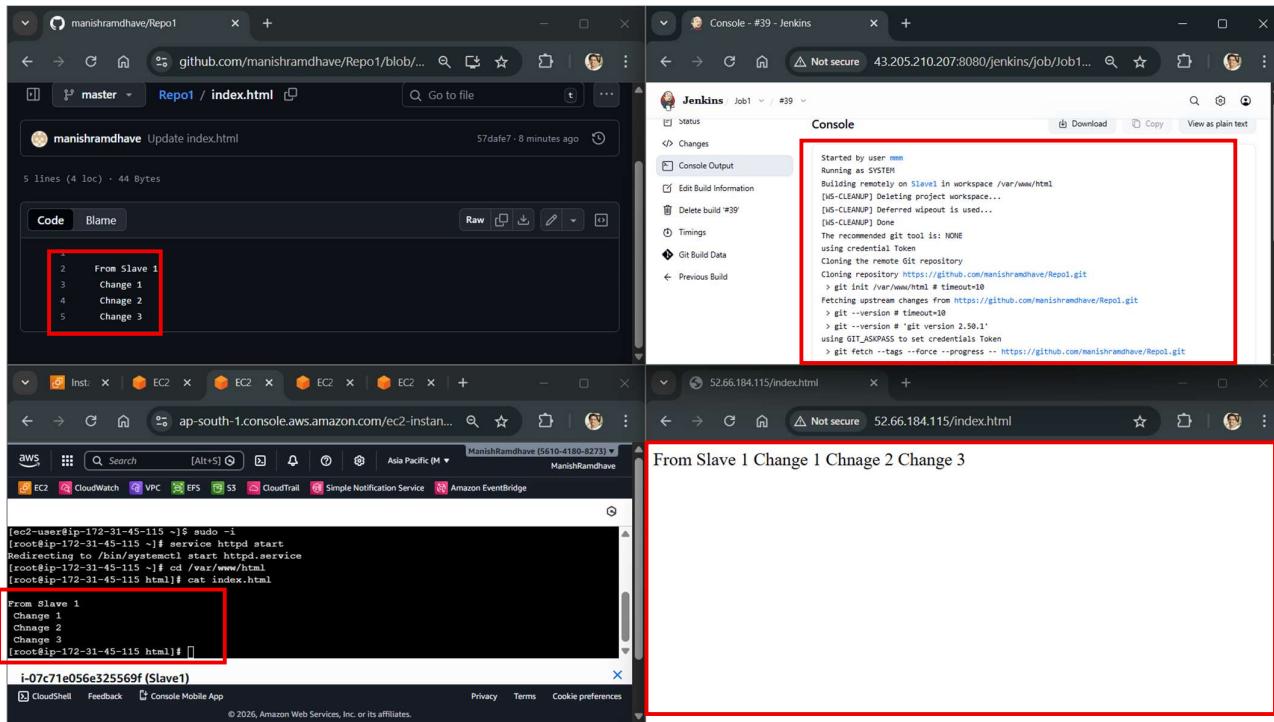
The screenshot shows the Jenkins Job2 configuration page. The left sidebar has tabs for General, Source Code Management, Triggers, Environment, Build Steps, and Post-build Actions. The Triggers tab is selected and highlighted in grey. Under the Triggers section, there is a 'Triggers' heading and a list of options: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling' (which is checked and highlighted with a red box), and 'Poll SCM'. Below the triggers is an 'Environment' section with the 'Delete workspace before build starts' option checked (also highlighted with a red box). At the bottom are 'Save' and 'Apply' buttons.

Step 14: Enabled ‘Restrict where this project can be run’ and used a custom workspace to deploy the index.html file from Git Repositories to ‘/var/www/html/’:

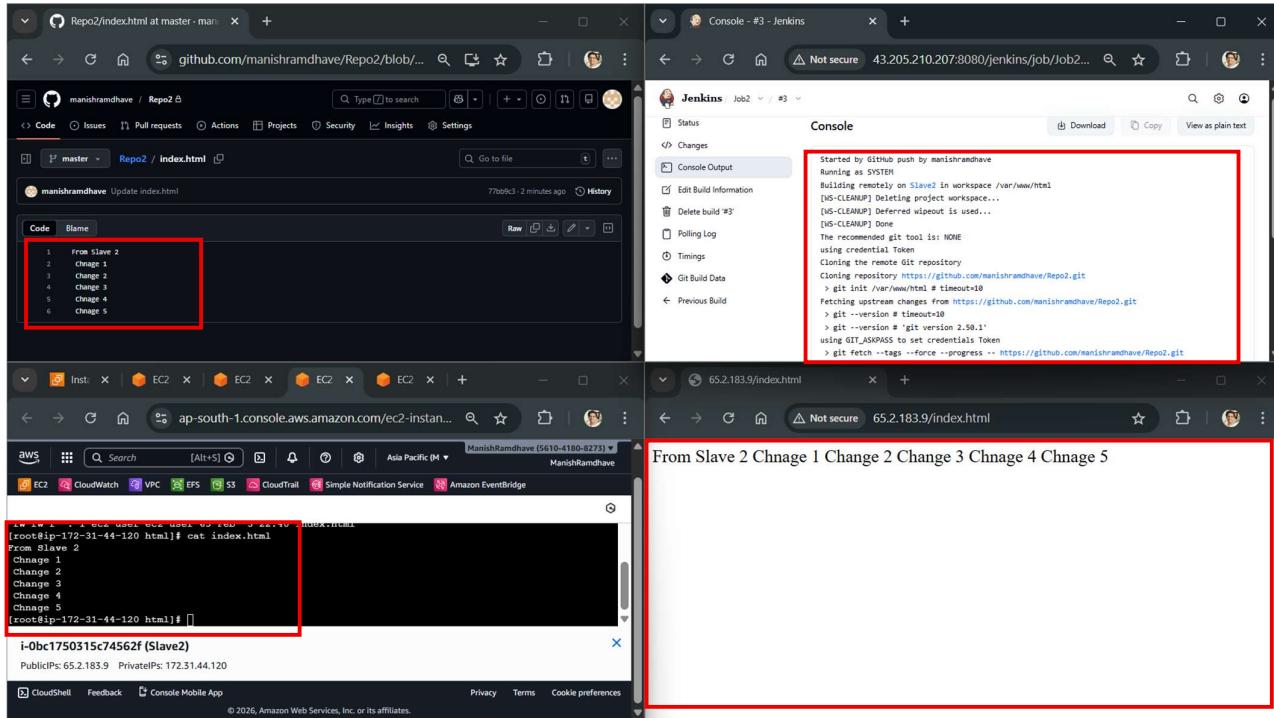
The screenshot shows the Jenkins Job2 configuration page. The left sidebar has tabs for General, Source Code Management, Triggers, Environment, Build Steps, and Post-build Actions. The General tab is selected and highlighted in grey. Under the General section, there are several unchecked checkboxes: 'This project is parameterised', 'Throttle builds', 'Execute concurrent builds if necessary', and 'Restrict where this project can be run' (which is checked and highlighted with a red box). Below this is a 'Label Expression' field containing 'Slave2'. A note states 'Label Slave2 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.' Under the 'Advanced' section, there are several unchecked checkboxes: 'Quiet period', 'Retry Count', 'Block build when upstream project is building', 'Block build when downstream project is building', and 'Use custom workspace' (which is checked and highlighted with a red box). The 'Directory' field under 'Use custom workspace' contains '/var/www/html/'. At the bottom are 'Save' and 'Apply' buttons.

Result:

- When changes are done in Git Repo1, Build is triggered by Job1 and the index.html file from Repo1 is hosted in the deployment folder of a **Slave1 instance**:



- When changes are done in Git Repo2, Build is triggered by Job2 and the index.html file from Repo2 is hosted in the deployment folder of a **Slave2 instance**:



- When changes are done in Git Repo3, Build is triggered by Job3 and the index.html file from Repo3 is hosted in the deployment folder of a **Slave3 instance**:::

