Experiment No. 5

Title: Applying CI/CD Principles to Web Development Using Jenkins, Git, and Local HTTP Server.

Objective:

To set up a basic CI/CD pipeline using **Jenkins**, **Git**, and a **local HTTP server** (Apache or Nginx) to automatically deploy a web application when code is pushed to the repository.

Introduction:

Continuous Integration and Continuous Deployment (CI/CD) is a critical practice in modern software development, allowing teams to automate the building, testing, and deployment of applications. This process ensures that software updates are consistently and reliably delivered to end-users, leading to improved development efficiency and product quality. In this context, this introduction sets the stage for an exploration of how to apply CI/CD principles specifically to web development using Jenkins, Git, and a local HTTP server. We will discuss the key components and concepts involved in this process.

Key Components:

● **Jenkins:** Jenkins is a widely used open-source automation server that helps automate various aspects of the software development process. It is known for its flexibility and extensibility and can be employed to create CI/CD pipelines.
● **Git:** Git is a distributed version control system used to manage and track changes in source code. It plays a crucial role in CI/CD by allowing developers to collaborate, track changes, and trigger automation processes when code changes are pushed to a repository.
● **Local HTTP Server:** A local HTTP server is used to host and serve web applications during development. It is where your web application can be tested before being deployed to production servers.

CI/CD Principles:

● Continuous Integration (CI): CI focuses on automating the process of integrating code changes into a shared repository frequently. It involves building and testing the application each time code is pushed to the repository to identify and address issues early in the development cycle.

● Continuous Deployment (CD): CD is the practice of automatically deploying code changes to production or staging environments after successful testing. CD aims to minimize manual intervention and reduce the time between code development and its availability to end-users.

The CI/CD Workflow:

● **Code Changes:** Developers make changes to the web application's source code locally.
● **Git Repository:** Developers push their code changes to a Git repository, such as GitHub or Bitbucket.
● **Webhook:** A webhook is configured in the Git repository to notify Jenkins whenever changes are pushed.
● **Jenkins Job:** Jenkins is set up to listen for webhook triggers. When a trigger occurs, Jenkins initiates a CI/CD pipeline.
● **Build and Test:** Jenkins executes a series of predefined steps, which may include building the application, running tests, and generating artifacts.
● **Deployment:** If all previous steps are successful, Jenkins deploys the application to a local HTTP server for testing.
● **Verification:** The deployed application is tested locally to ensure it functions as expected.
● Optional Staging: For more complex setups, there might be a staging environment where the application undergoes further testing before reaching production.
● **Production Deployment:** If the application passes all tests, it can be deployed to the production server.

Benefits of CI/CD in Web Development:
● **Rapid Development:** CI/CD streamlines development processes, reducing manual tasks and allowing developers to focus on writing code.
● **Improved Quality:** Automated testing helps catch bugs early, ensuring higher code quality.
● **Faster Time to Market:** Automated deployments reduce the time it takes to release new features or updates.
● **Consistency:** The process ensures that code is built, tested, and deployed consistently, reducing the risk of errors.

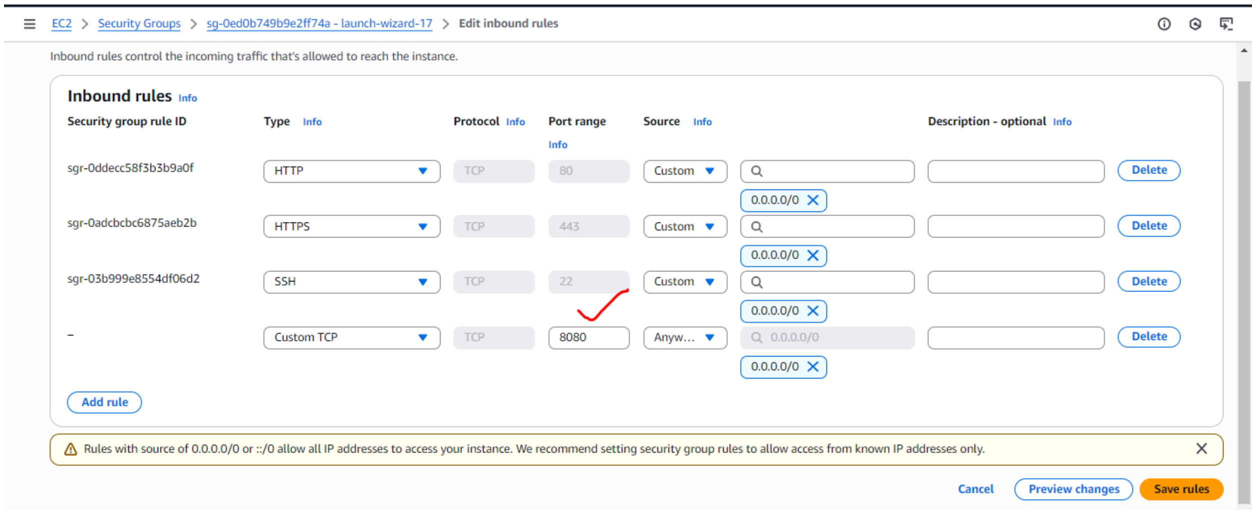
Prerequisites:

- Jenkins installed and running
- Apache2 or Nginx installed
- Git installed
- A basic web application (e.g., HTML/CSS/JS or a small React app)
- GitHub account (or GitLab/Bitbucket)

- Linux environment preferred (Ubuntu/Debian-based)

Experiment Steps:

NOTE: *Make sure that the port 8080 is opened in your EC2 instance for Jenkins.*



EC2 > Security Groups > sg-0ed0b749b9e2ff74a - launch-wizard-17 > Edit inbound rules

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

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-0ddecc58f3b3b9a0f	HTTP	TCP	80	Custom	0.0.0.0/0	
sgr-0adcbcbc6875aeb2b	HTTPS	TCP	443	Custom	0.0.0.0/0	
sgr-03b999e8554df06d2	SSH	TCP	22	Custom	0.0.0.0/0	
-	Custom TCP	TCP	8080	Anyw...	0.0.0.0/0	

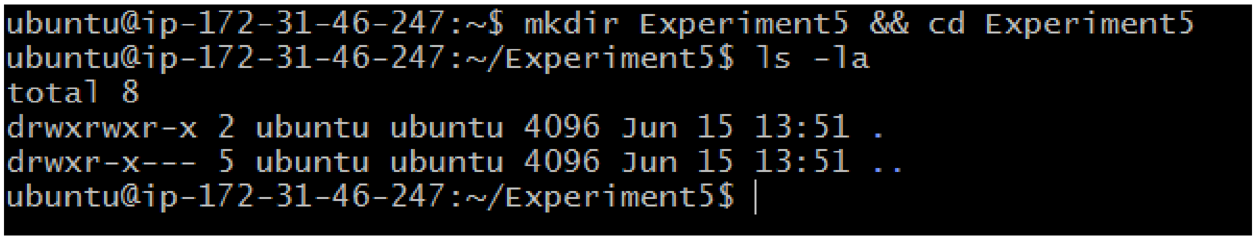
Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Step 1: Set Up the Web Application and Local HTTP Server (Apache2)

- **Web Application Setup**

```
mkdir Experiment5 && cd Experiment5
```

```
ls -la
```

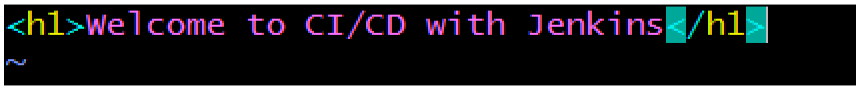


```
ubuntu@ip-172-31-46-247:~$ mkdir Experiment5 && cd Experiment5
ubuntu@ip-172-31-46-247:~/Experiment5$ ls -la
total 8
drwxrwxr-x 2 ubuntu ubuntu 4096 Jun 15 13:51 .
drwxr-x--- 5 ubuntu ubuntu 4096 Jun 15 13:51 ..
ubuntu@ip-172-31-46-247:~/Experiment5$ 
```

```
vi index.html
```

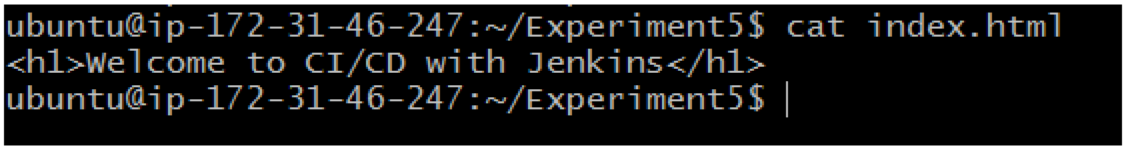
Pate the following content, save and exit.

```
<h1>Welcome to CI/CD with Jenkins</h1>
```



```
<h1>Welcome to CI/CD with Jenkins</h1>
~
```

```
cat index.html
```

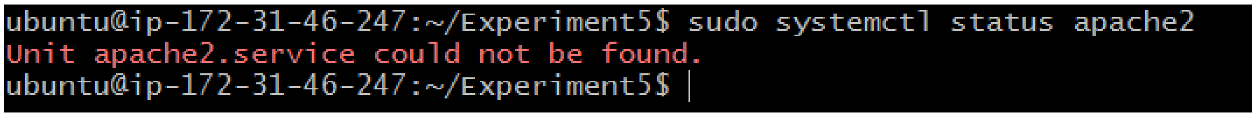


```
ubuntu@ip-172-31-46-247:~/Experiment5$ cat index.html
<h1>Welcome to CI/CD with Jenkins</h1>
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

- **Install and Start Apache2 (or Nginx)**

```
sudo apt update
```

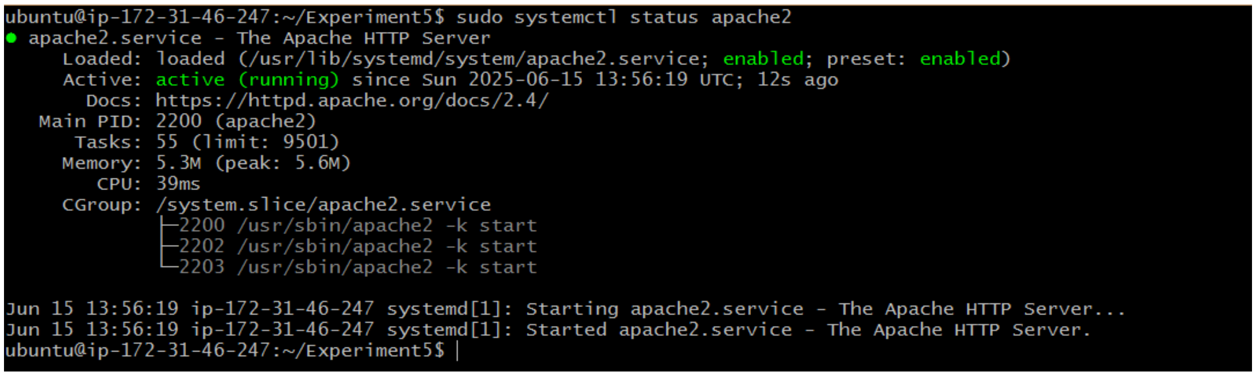
```
sudo systemctl status apache2
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo systemctl status apache2
Unit apache2.service could not be found.
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
sudo apt install apache2 -y
```

```
sudo systemctl status apache2
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Sun 2025-06-15 13:56:19 UTC; 12s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2200 (apache2)
      Tasks: 55 (limit: 9501)
     Memory: 5.3M (peak: 5.6M)
        CPU: 39ms
     CGroup: /system.slice/apache2.service
             ├─2200 /usr/sbin/apache2 -k start
             ├─2202 /usr/sbin/apache2 -k start
             └─2203 /usr/sbin/apache2 -k start

Jun 15 13:56:19 ip-172-31-46-247 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Jun 15 13:56:19 ip-172-31-46-247 systemd[1]: Started apache2.service - The Apache HTTP Server.
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

- **Configure Apache Document Root (Optional)**

```
sudo mkdir -p /var/www/html/webdirectory
```

# Set ownership to jenkins user so it can copy files there during deployment.

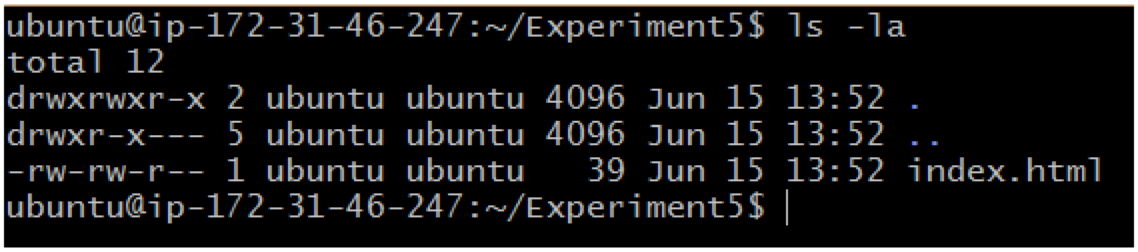
```
sudo chown -R jenkins:www-data /var/www/html/webdirectory
```

Test:
Visit `http://<Public_IP_of_EC2_Instance>/webdirectory` in a browser — it should show the current content.

Step 2: Set Up Git Repository

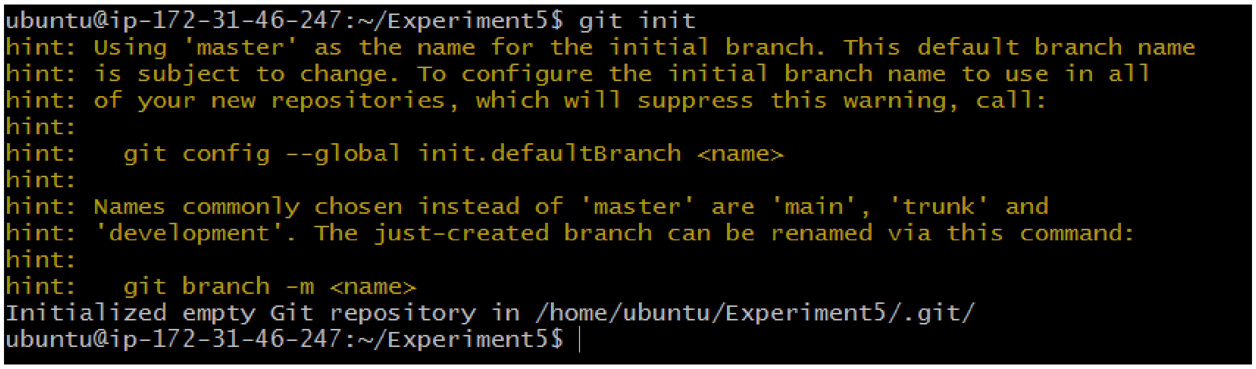
```
cd Experiment5
```

```
ls -la
```



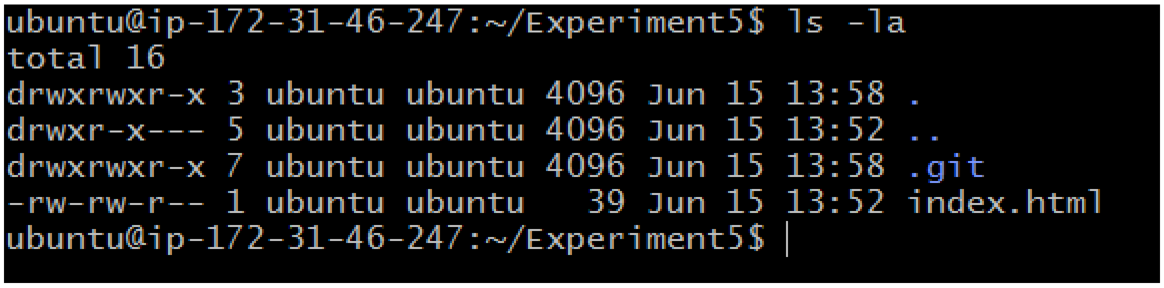
```
ubuntu@ip-172-31-46-247:~/Experiment5$ ls -la
total 12
drwxrwxr-x 2 ubuntu ubuntu 4096 Jun 15 13:52 .
drwxr-x--- 5 ubuntu ubuntu 4096 Jun 15 13:52 ..
-rw-rw-r-- 1 ubuntu ubuntu   39 Jun 15 13:52 index.html
ubuntu@ip-172-31-46-247:~/Experiment5$
```

```
git init
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/ubuntu/Experiment5/.git/
ubuntu@ip-172-31-46-247:~/Experiment5$
```

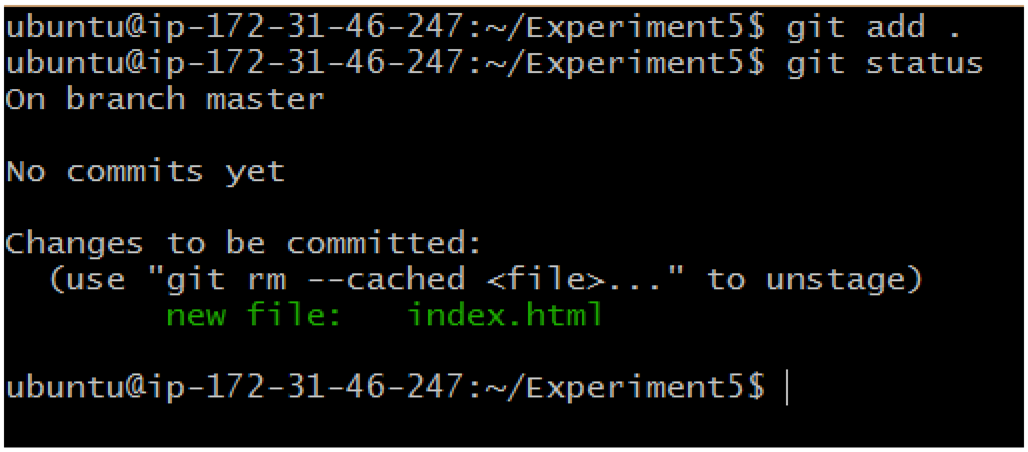
```
ls -la
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ ls -la
total 16
drwxrwxr-x 3 ubuntu ubuntu 4096 Jun 15 13:58 .
drwxr-x--- 5 ubuntu ubuntu 4096 Jun 15 13:52 ..
drwxrwxr-x 7 ubuntu ubuntu 4096 Jun 15 13:58 .git
-rw-rw-r-- 1 ubuntu ubuntu   39 Jun 15 13:52 index.html
ubuntu@ip-172-31-46-247:~/Experiment5$
```

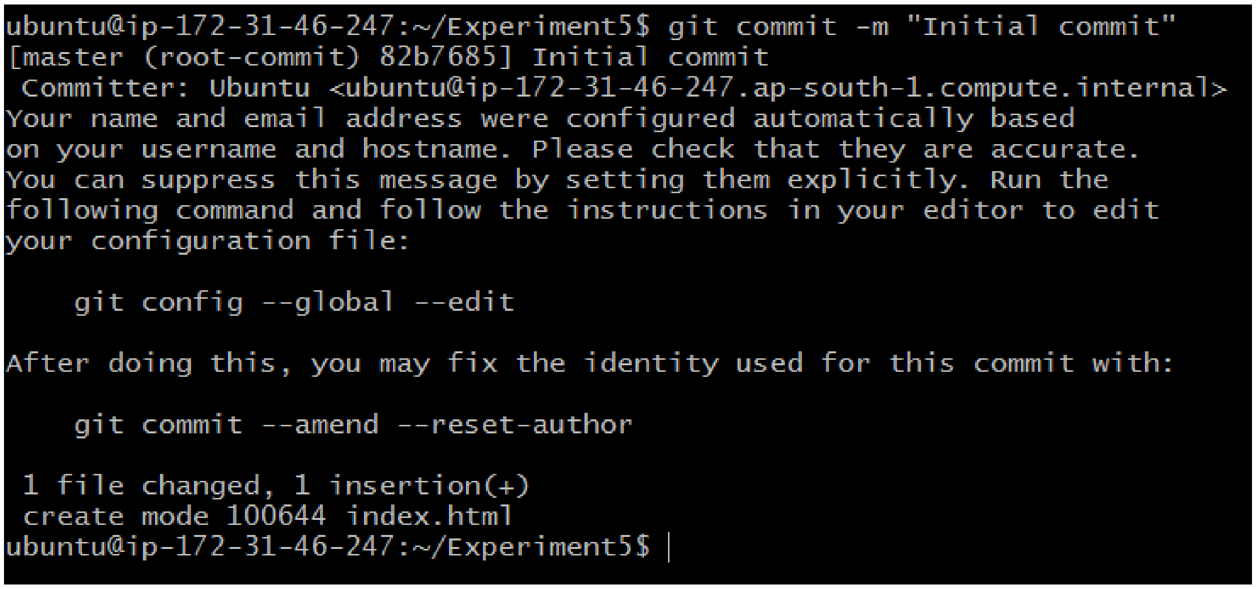
```
git add .
```

```
git status
```



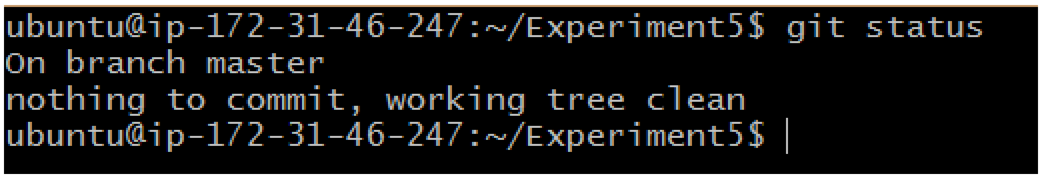
```
ubuntu@ip-172-31-46-247:~/Experiment5$ git add .
ubuntu@ip-172-31-46-247:~/Experiment5$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
git commit -m "Initial commit"
```



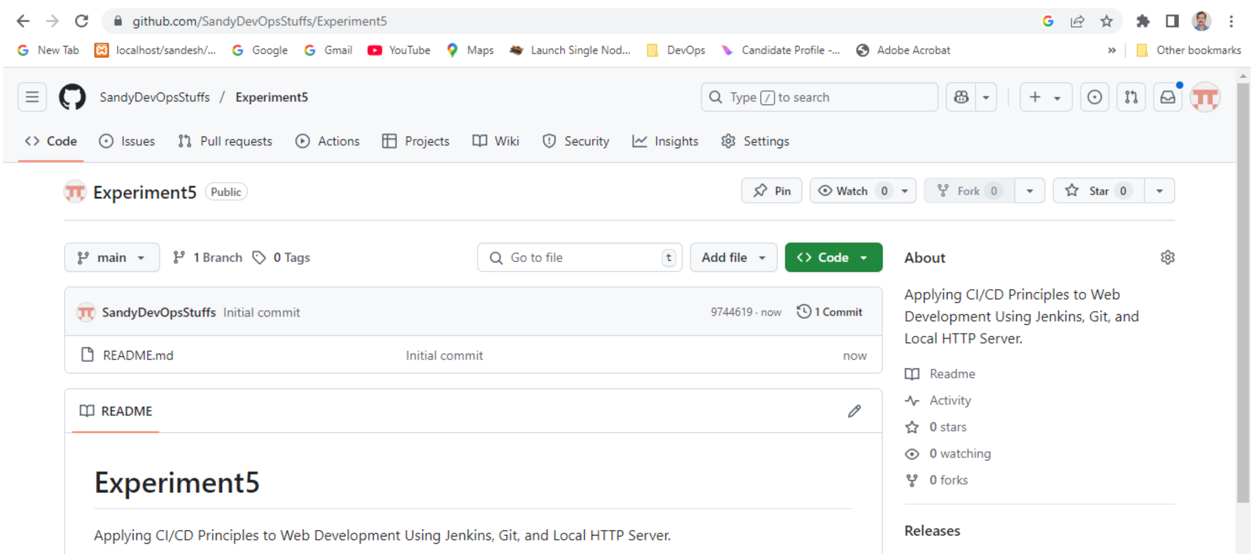
```
ubuntu@ip-172-31-46-247:~/Experiment5$ git commit -m "Initial commit"
[master (root-commit) 82b7685] Initial commit
 Committer: Ubuntu <ubuntu@ip-172-31-46-247.ap-south-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+)
 create mode 100644 index.html
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
git status
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ git status
On branch master
nothing to commit, working tree clean
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

Create a new remote repository on GitHub (or Bitbucket/GitLab):



SandyDevOpsStuffs Initial commit 9744619 · now 1 Commit

README.md Initial commit now

README

Experiment5

Applying CI/CD Principles to Web Development Using Jenkins, Git, and Local HTTP Server.

About

Applying CI/CD Principles to Web Development Using Jenkins, Git, and Local HTTP Server.

Readme
Activity
0 stars
0 watching
0 forks

Releases

```
git remote add origin https://github.com/<your-username>/<repo-name>.git
```

(If you wish to enter username and password every time you run `git push` and `git pull` commands).

OR

```
git remote add origin git@github.com:SandyDevOpsStuffs/Experiment5.git
```

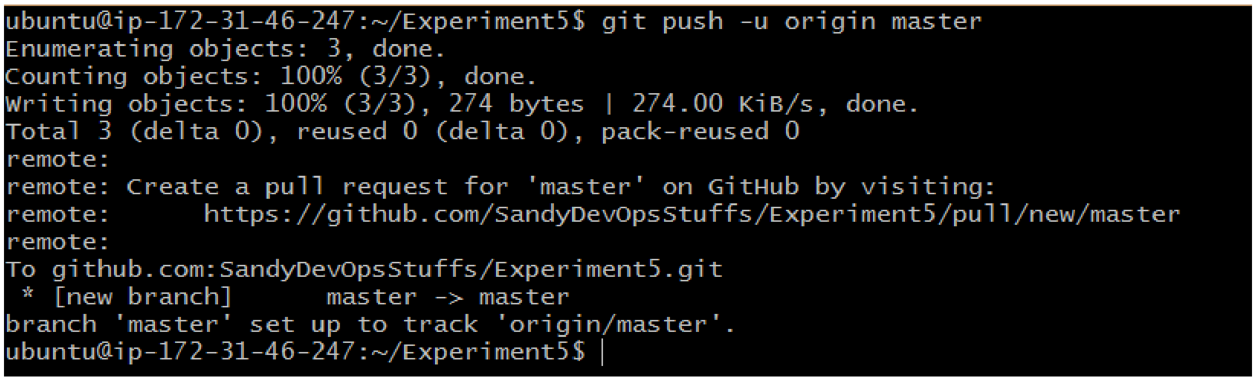
(If you wish to use passwordless authentication).

Create an SSH key pair in local:

```
ssh-keygen
```

Add public key to GitHub then push the code.

```
git push -u origin master
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 274 bytes | 274.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/SandyDevOpsStuffs/Experiment5/pull/new/master
remote:
To github.com:SandyDevOpsStuffs/Experiment5.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

NOTE:

- *You **initialized your local Git repo**, which by default created a branch called `master`.*
- *But **on GitHub**, the default branch is usually `main` (not `master`).*
- *When you run `git push -u origin master`, it **pushes a new `master` branch to GitHub**, which is now available remotely — but **it's not the default branch** there.*
- *Then if you run `git push -u origin main`, Git gave this error:*

  error: src refspec main does not match any

  *Because you **don't have a local branch named `main`**, only `master`.*

  Fix Option 1: Set master as the default branch on GitHub (Recommended for this case).

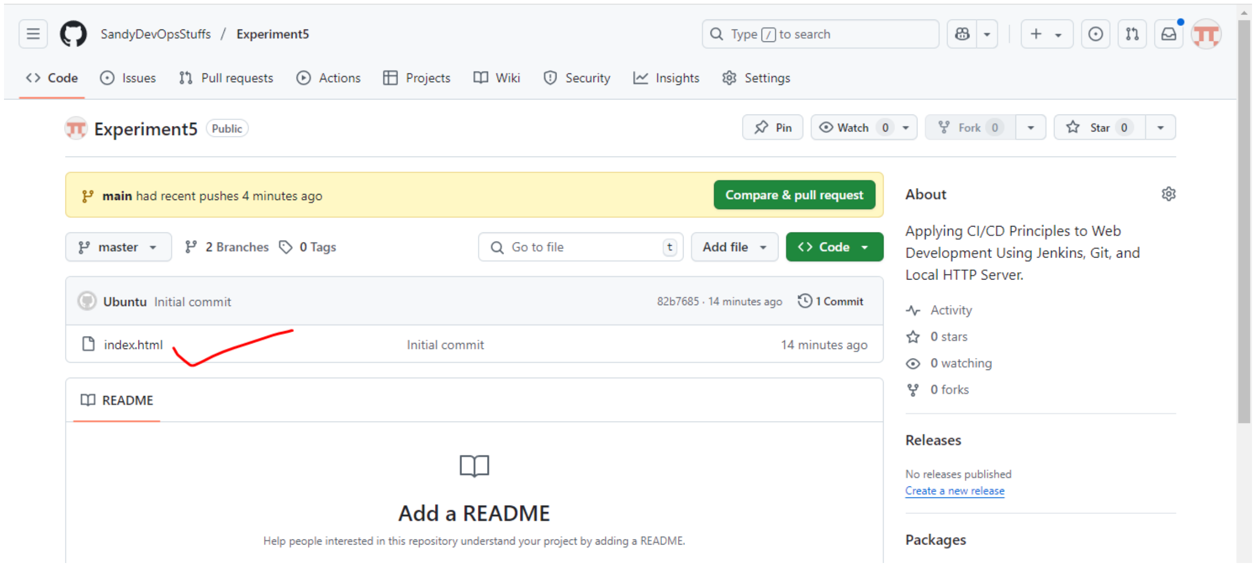
  Since your local branch is master, make GitHub use it:

1. *Go to your repo on GitHub:* [*https://github.com/SandyDevOpsStuffs/Experiment5*](https://github.com/SandyDevOpsStuffs/Experiment5)
2. *Navigate to:*
   Settings → Branches → Default branch
3. *Click **"Change default branch"** → select `master`.*
4. *Optionally, delete the empty `main` branch (if it exists):*
   - *Go to the **Branches** tab.*
   - *Delete `main` if it's unused.*

Fix Option 2: Rename your local branch to main

If you want to follow GitHub's default naming:

```
# Rename master to main
git branch -m master main

# Push the renamed branch and set upstream
git push -u origin main

# Optionally delete remote master if not needed
git push origin --delete master
```


SandyDevOpsStuffs / Experiment5

Experiment5 Public

main had recent pushes 4 minutes ago

master · 2 Branches · 0 Tags

Ubuntu Initial commit 82b7685 · 14 minutes ago 1 Commit

index.html Initial commit 14 minutes ago

About

Applying CI/CD Principles to Web Development Using Jenkins, Git, and Local HTTP Server.

Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

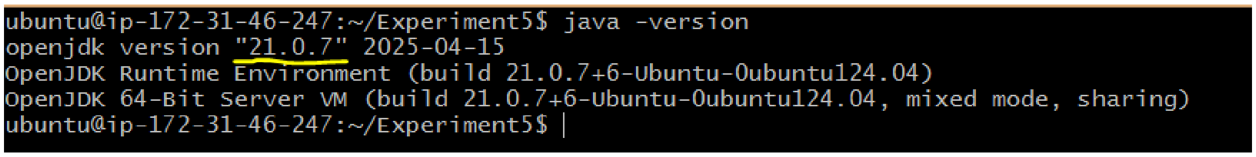
README

Add a README

Help people interested in this repository understand your project by adding a README.

Step 3: Install Java

```
sudo apt install fontconfig openjdk-21-jre

java -version
```

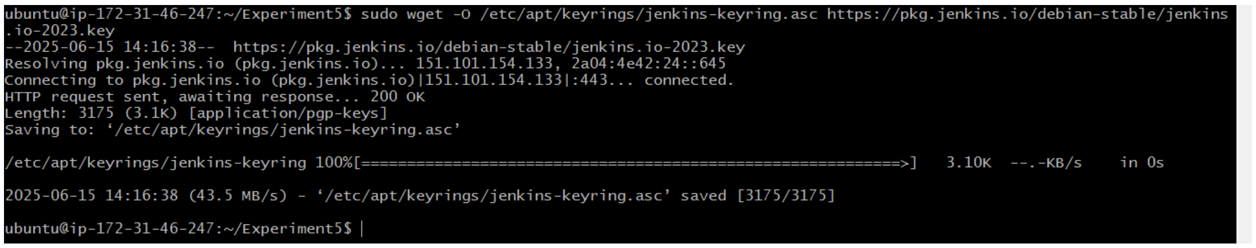


```
ubuntu@ip-172-31-46-247:~/Experiment5$ java -version
openjdk version "21.0.7" 2025-04-15
OpenJDK Runtime Environment (build 21.0.7+6-Ubuntu-0ubuntu124.04)
OpenJDK 64-Bit Server VM (build 21.0.7+6-Ubuntu-0ubuntu124.04, mixed mode, sharing)
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

Step 4: Install and Configure Jenkins

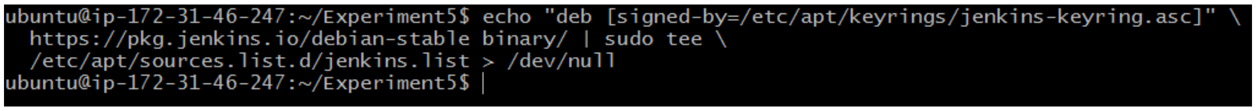
- **Install Jenkins (Ubuntu example)**

```
sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
```



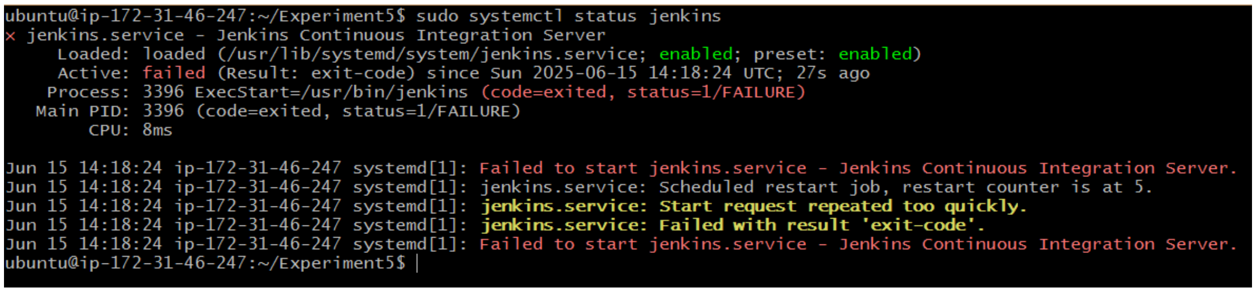
```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc https://pkg.jenkins.io/debian-stable/jenkins
.io-2023.key
--2025-06-15 14:16:38--  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 151.101.154.133, 2a04:4e42:24::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|151.101.154.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1K) [application/pgp-keys]
Saving to: '/etc/apt/keyrings/jenkins-keyring.asc'

/etc/apt/keyrings/jenkins-keyring 100%[=============================================================>]   3.10K  --.-KB/s    in 0s

2025-06-15 14:16:38 (43.5 MB/s) - '/etc/apt/keyrings/jenkins-keyring.asc' saved [3175/3175]

ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
```



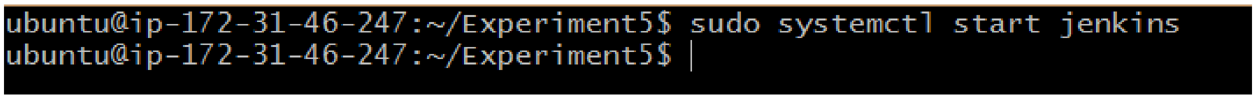
```
ubuntu@ip-172-31-46-247:~/Experiment5$ echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
sudo apt update

sudo apt-get install jenkins

sudo systemctl status jenkins
```



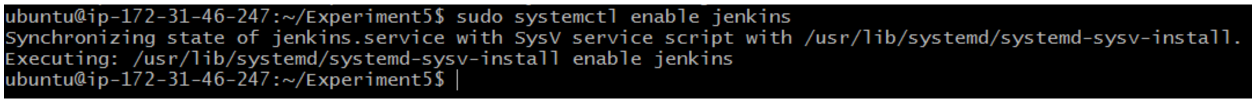
```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo systemctl status jenkins
× jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: failed (Result: exit-code) since Sun 2025-06-15 14:18:24 UTC; 27s ago
    Process: 3396 ExecStart=/usr/bin/jenkins (code=exited, status=1/FAILURE)
   Main PID: 3396 (code=exited, status=1/FAILURE)
        CPU: 8ms

Jun 15 14:18:24 ip-172-31-46-247 systemd[1]: Failed to start jenkins.service - Jenkins Continuous Integration Server.
Jun 15 14:18:24 ip-172-31-46-247 systemd[1]: jenkins.service: Scheduled restart job, restart counter is at 5.
Jun 15 14:18:24 ip-172-31-46-247 systemd[1]: jenkins.service: Start request repeated too quickly.
Jun 15 14:18:24 ip-172-31-46-247 systemd[1]: jenkins.service: Failed with result 'exit-code'.
Jun 15 14:18:24 ip-172-31-46-247 systemd[1]: Failed to start jenkins.service - Jenkins Continuous Integration Server.
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
sudo systemctl start Jenkins
```



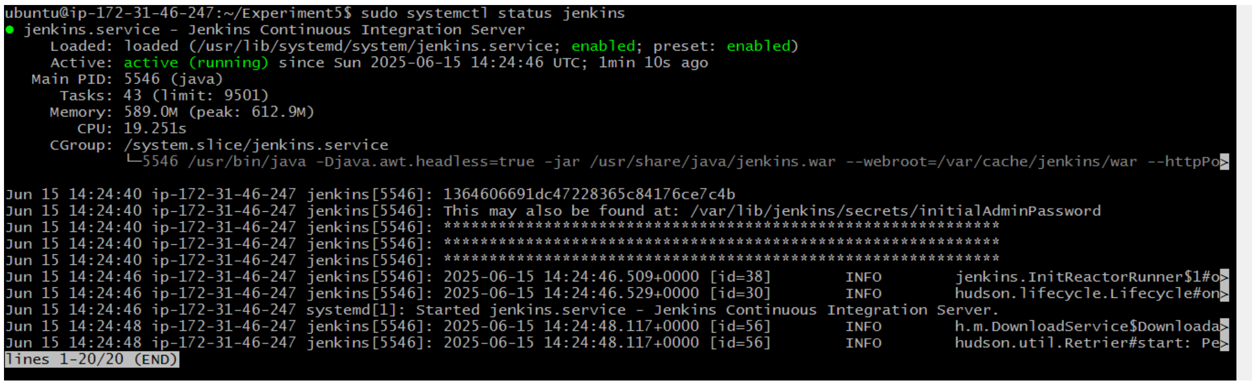
```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo systemctl start jenkins
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
sudo systemctl enable Jenkins
```



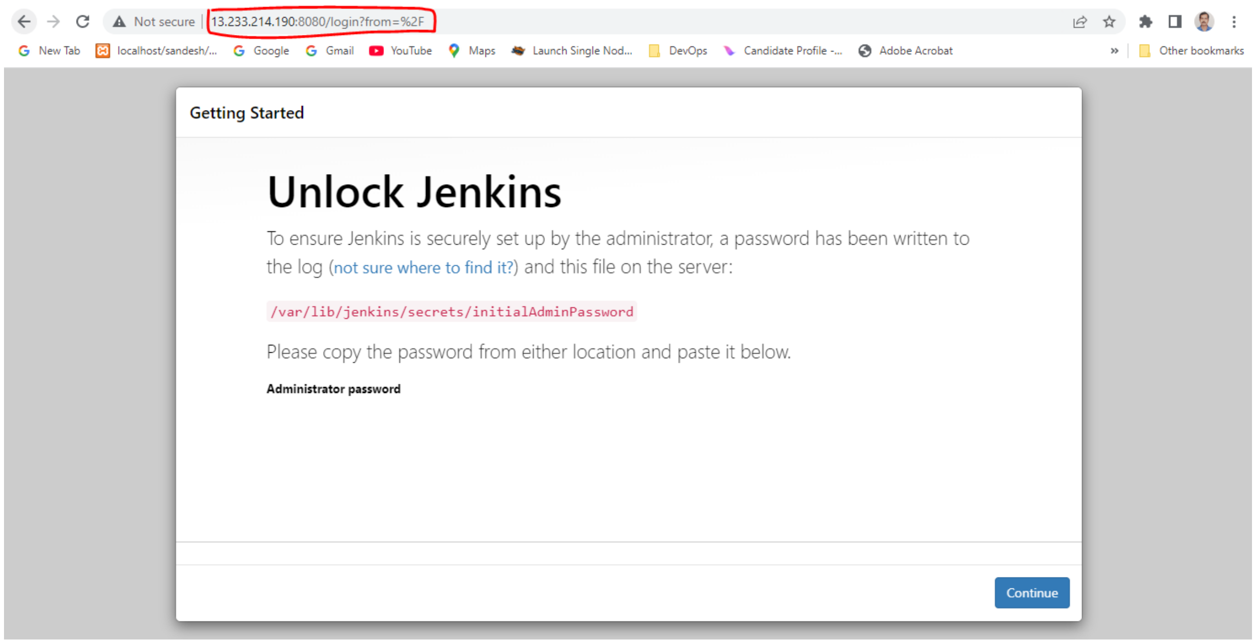
```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

```
sudo systemctl status jenkins
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Sun 2025-06-15 14:24:46 UTC; 1min 10s ago
   Main PID: 5546 (java)
      Tasks: 43 (limit: 9501)
     Memory: 589.0M (peak: 612.9M)
        CPU: 19.251s
     CGroup: /system.slice/jenkins.service
             └─5546 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPo>

Jun 15 14:24:40 ip-172-31-46-247 jenkins[5546]: 1364606691dc47228365c84176ce7c4b
Jun 15 14:24:40 ip-172-31-46-247 jenkins[5546]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jun 15 14:24:40 ip-172-31-46-247 jenkins[5546]: *************************************************************
Jun 15 14:24:40 ip-172-31-46-247 jenkins[5546]: *************************************************************
Jun 15 14:24:40 ip-172-31-46-247 jenkins[5546]: *************************************************************
Jun 15 14:24:46 ip-172-31-46-247 jenkins[5546]: 2025-06-15 14:24:46.509+0000 [id=38]        INFO        jenkins.InitReactorRunner$1#o>
Jun 15 14:24:46 ip-172-31-46-247 jenkins[5546]: 2025-06-15 14:24:46.529+0000 [id=30]        INFO        hudson.lifecycle.Lifecycle#on>
Jun 15 14:24:46 ip-172-31-46-247 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Jun 15 14:24:48 ip-172-31-46-247 jenkins[5546]: 2025-06-15 14:24:48.117+0000 [id=56]        INFO        h.m.DownloadService$Downloada>
Jun 15 14:24:48 ip-172-31-46-247 jenkins[5546]: 2025-06-15 14:24:48.117+0000 [id=56]        INFO        hudson.util.Retrier#start: Pe>
lines 1-20/20 (END)
```

Visit `http://localhost:8080` OR `http://<Public_IP_of_EC2_Instance>:8080` and complete setup using the initial password:


Not secure | 13.233.214.190:8080/login?from=%2F

Getting Started

Unlock Jenkins

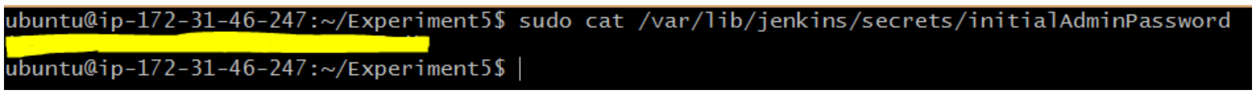
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

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

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

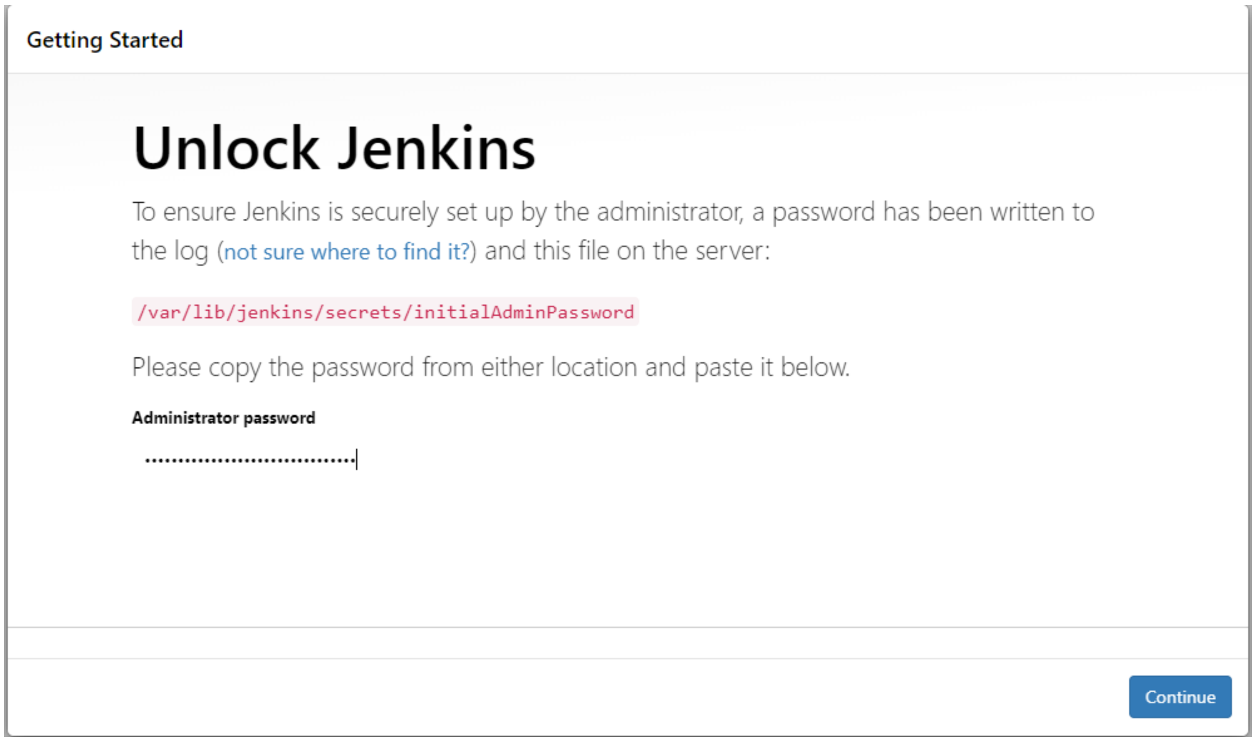
Administrator password

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```



```
ubuntu@ip-172-31-46-247:~/Experiment5$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
[redacted]
ubuntu@ip-172-31-46-247:~/Experiment5$ |
```

Paste it here.


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:

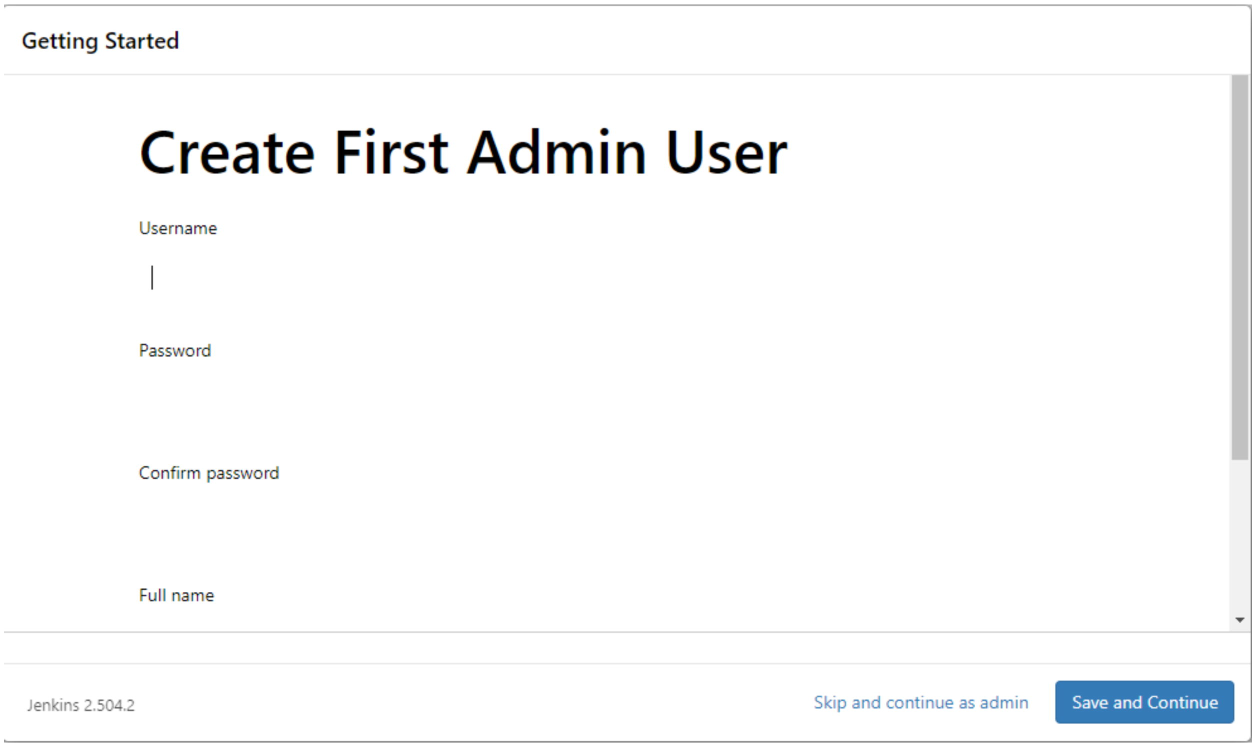
`/var/lib/jenkins/secrets/initialAdminPassword`

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

Administrator password

································

Install recommended plugins and create an admin user.


Getting Started

Create First Admin User

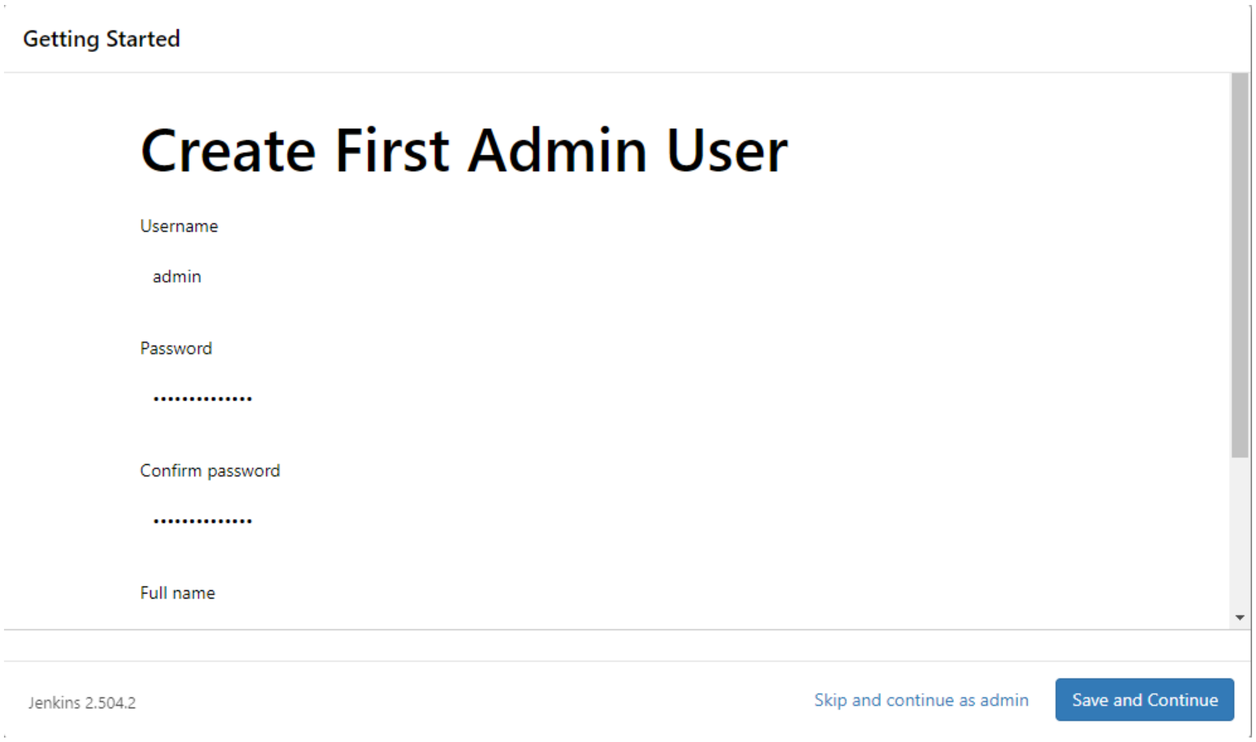
Username

Password

Confirm password

Full name

Jenkins 2.504.2


Getting Started

Create First Admin User

Username

admin

Password

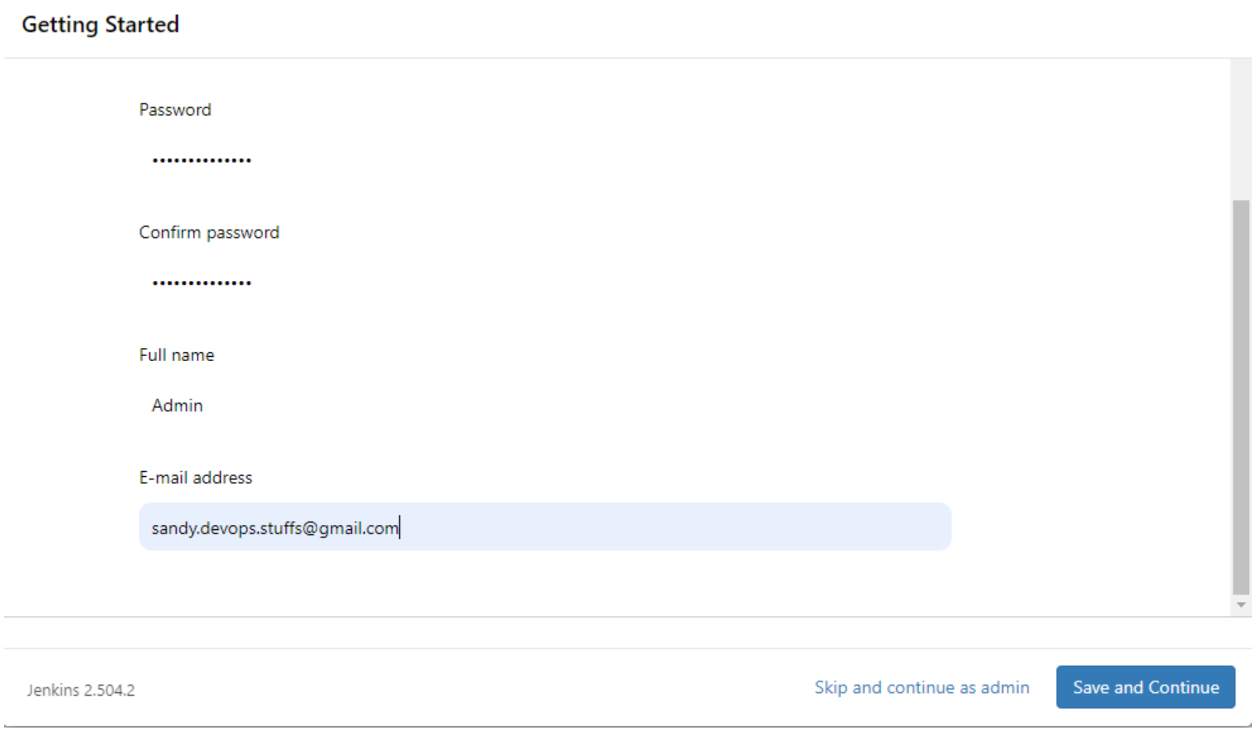
•••••••••••••

Confirm password

•••••••••••••

Full name

Jenkins 2.504.2



Getting Started

Password

••••••••••••••

Confirm password

••••••••••••••

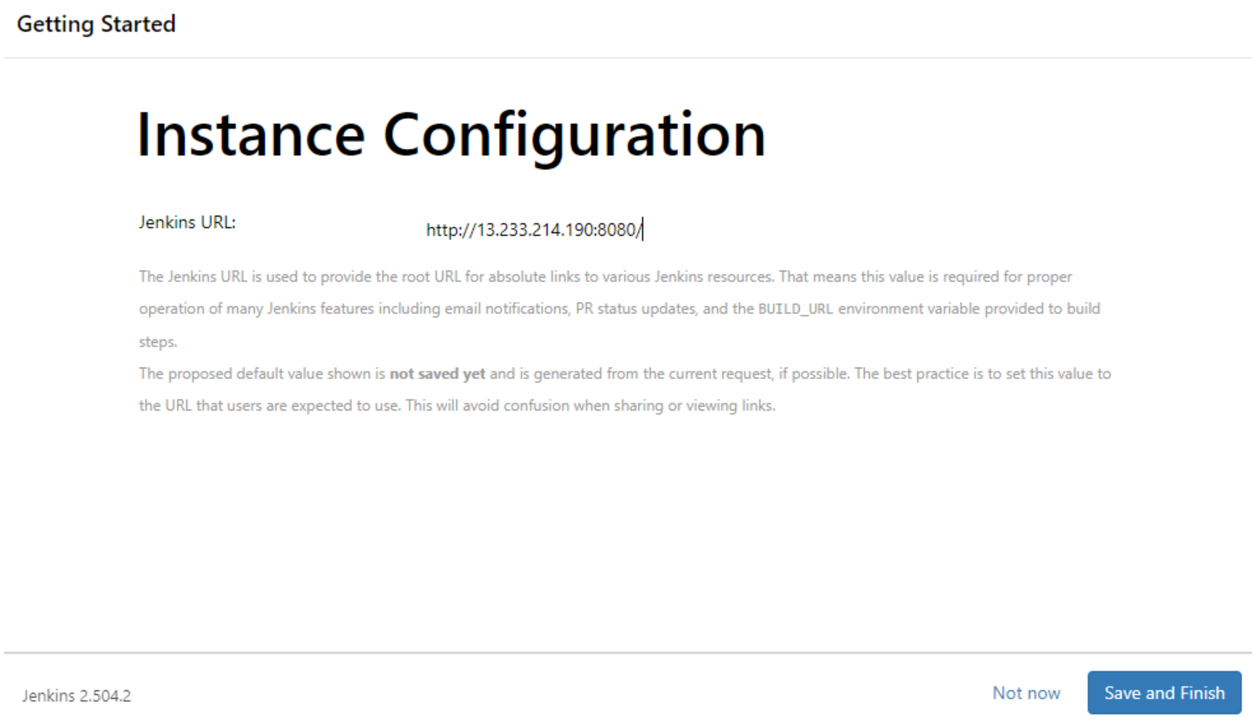
Full name

Admin

E-mail address

sandy.devops.stuffs@gmail.com

Jenkins 2.504.2



Getting Started

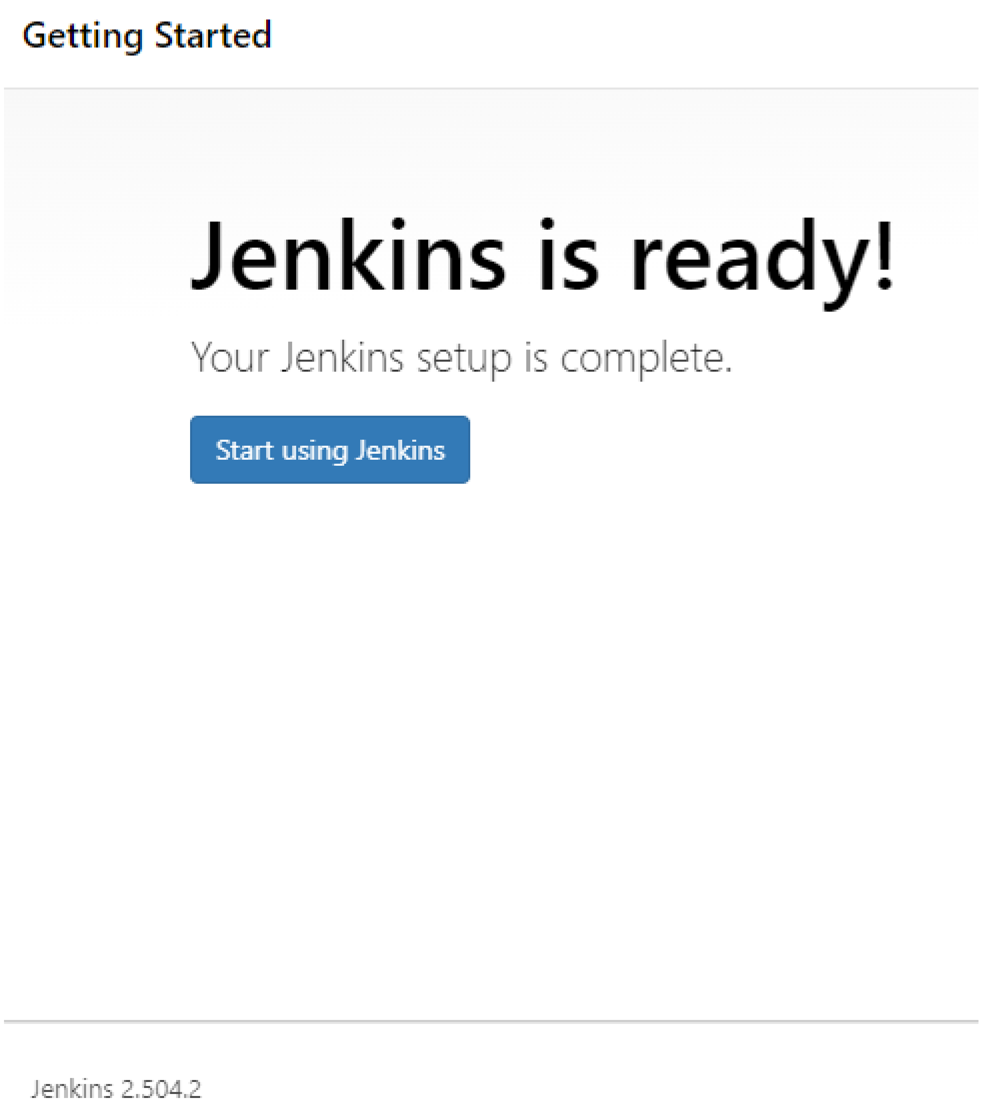
Instance Configuration

Jenkins URL: http://13.233.214.190:8080/

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

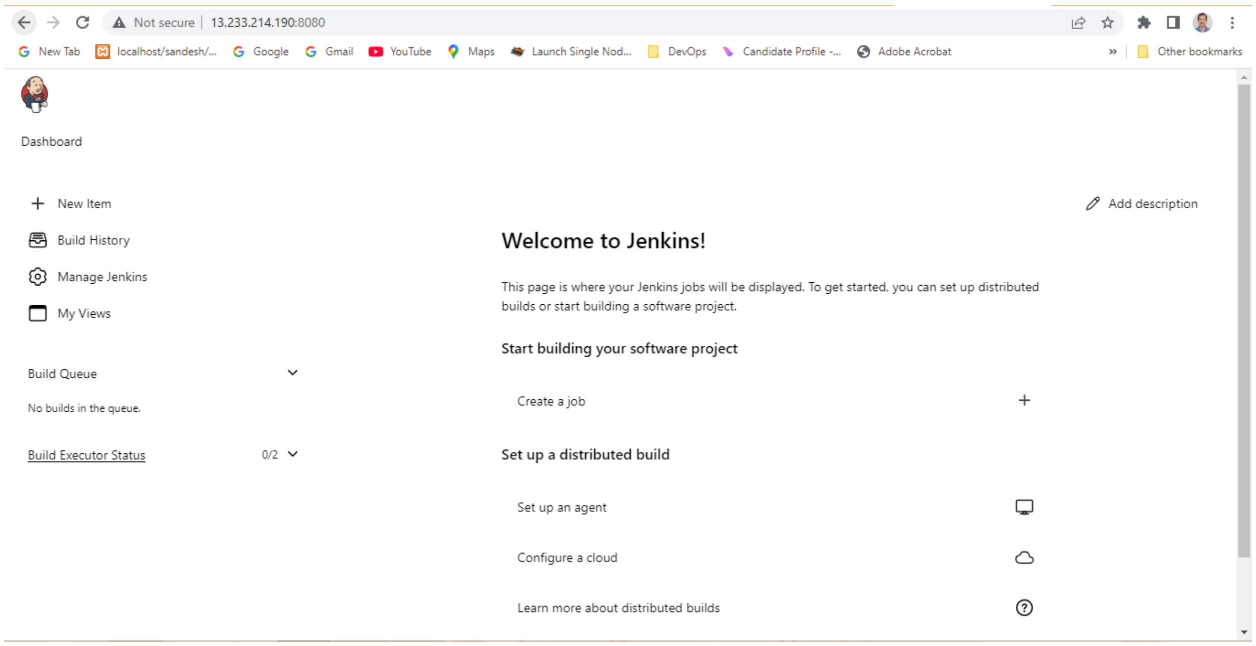
Jenkins 2.504.2


Getting Started

Jenkins is ready!

Your Jenkins setup is complete.

Jenkins 2.504.2


Not secure | 13.233.214.190:8080
New Tab localhost/sandesh/... Google Gmail YouTube Maps Launch Single Nod... DevOps Candidate Profile -... Adobe Acrobat Other bookmarks
Dashboard
New Item
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status 0/2
Add description

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

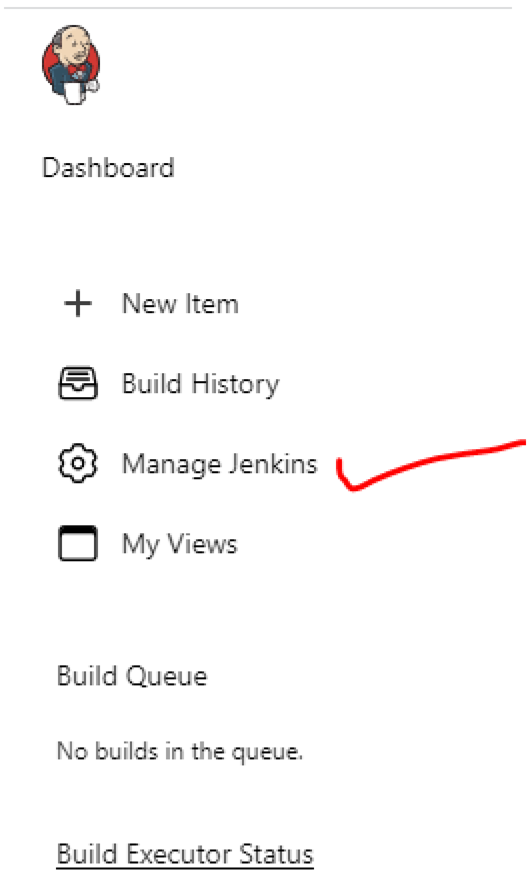
Configure a cloud

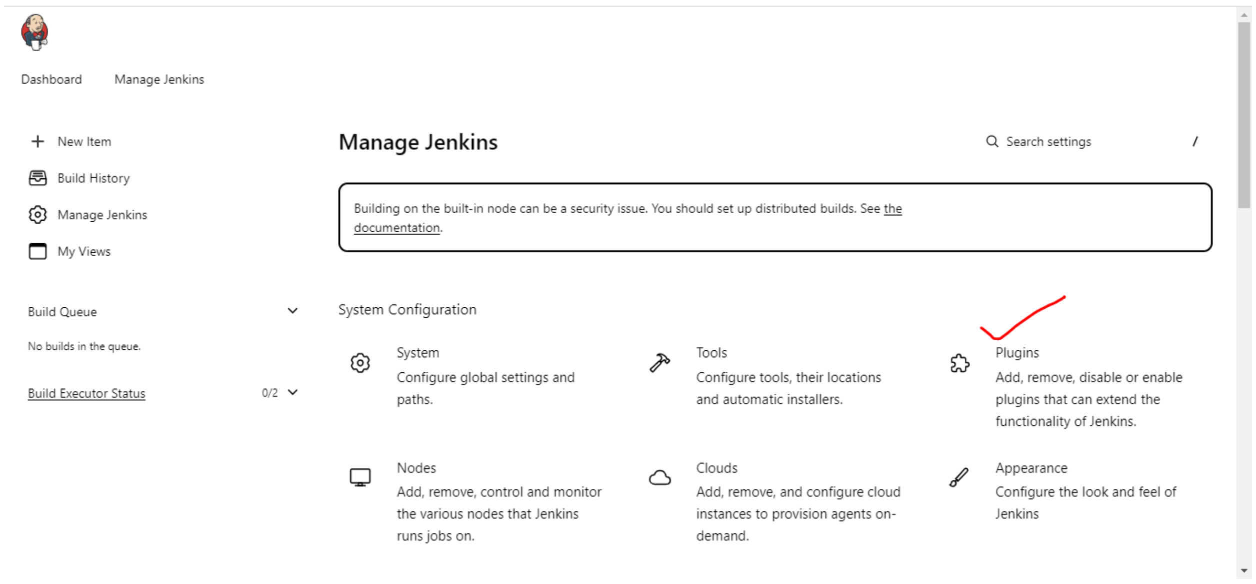
Learn more about distributed builds

Step 4: Install Required Jenkins Plugins

Install:

- Git Plugin
- GitHub Integration Plugin
- Pipeline Plugin (optional)
- Any required Authentication Plugins





Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.

System Configuration

System
Configure global settings and paths.

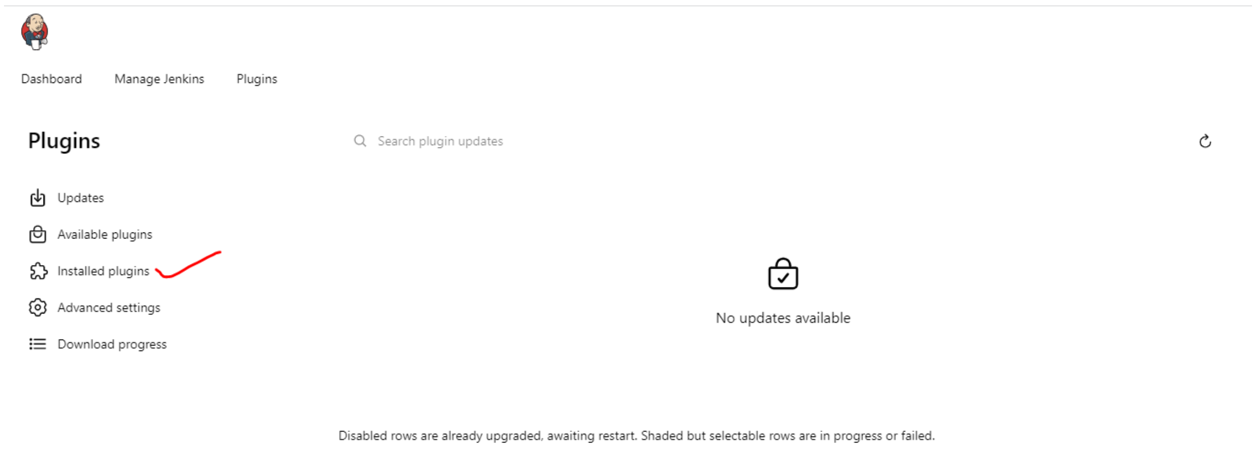
Tools
Configure tools, their locations and automatic installers.

Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

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

Clouds
Add, remove, and configure cloud instances to provision agents on-demand.

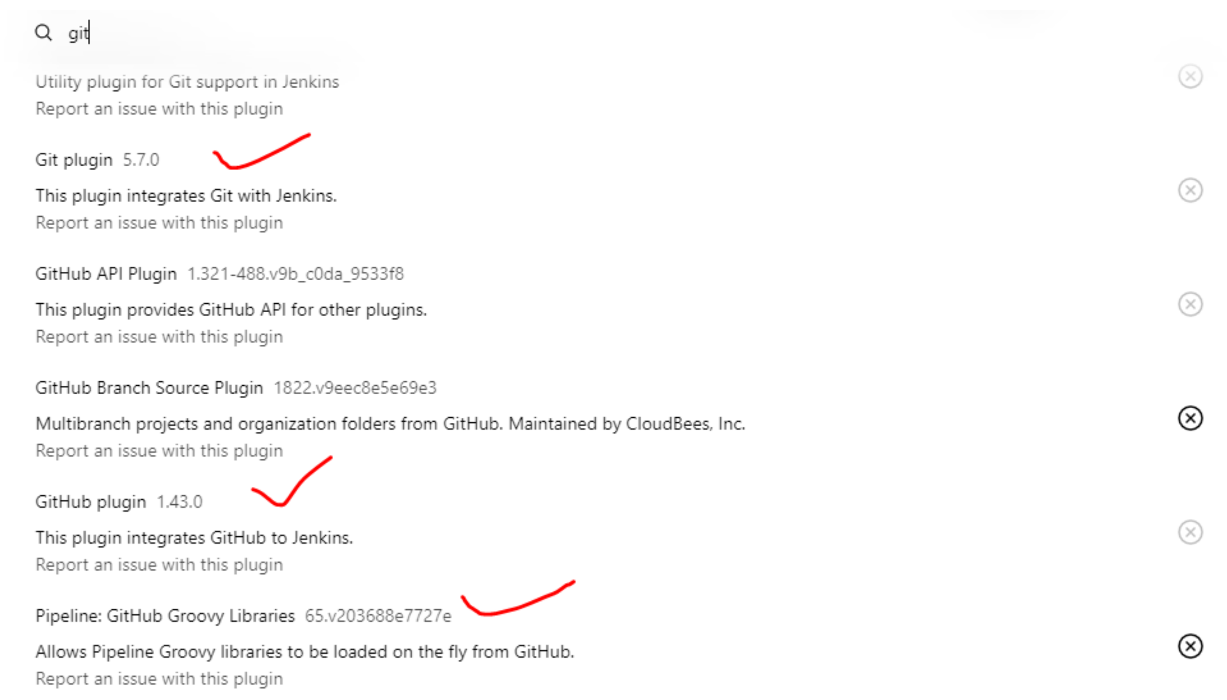
Appearance
Configure the look and feel of Jenkins



Plugins

No updates available

Disabled rows are already upgraded, awaiting restart. Shaded but selectable rows are in progress or failed.



Utility plugin for Git support in Jenkins
Report an issue with this plugin

Git plugin 5.7.0
This plugin integrates Git with Jenkins.
Report an issue with this plugin

GitHub API Plugin 1.321-488.v9b_c0da_9533f8
This plugin provides GitHub API for other plugins.
Report an issue with this plugin

GitHub Branch Source Plugin 1822.v9eec8e5e69e3
Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.
Report an issue with this plugin

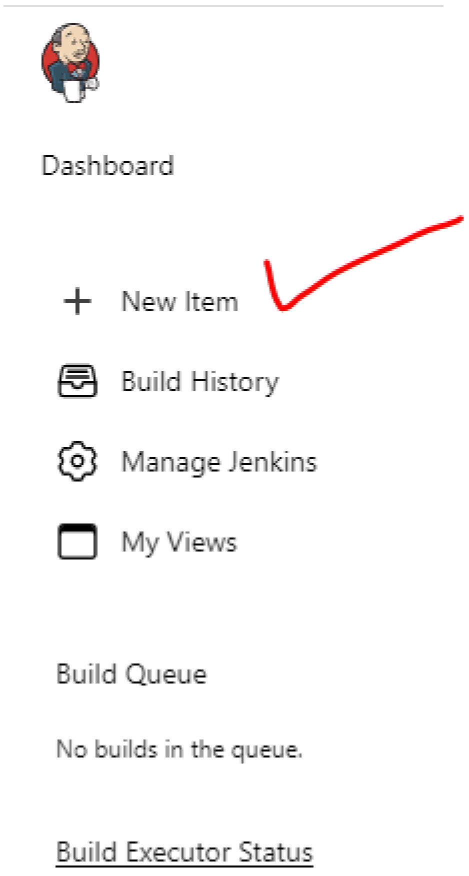
GitHub plugin 1.43.0
This plugin integrates GitHub to Jenkins.
Report an issue with this plugin

Pipeline: GitHub Groovy Libraries 65.v203688e7727e
Allows Pipeline Groovy libraries to be loaded on the fly from GitHub.
Report an issue with this plugin

Step 5: Create and Configure Jenkins Job

- **Create Freestyle Project**

  ✓ Open Jenkins Dashboard → **New Item** → **Freestyle Project** → Name it: WebApp-CICD



Dashboard

New Item

Build History

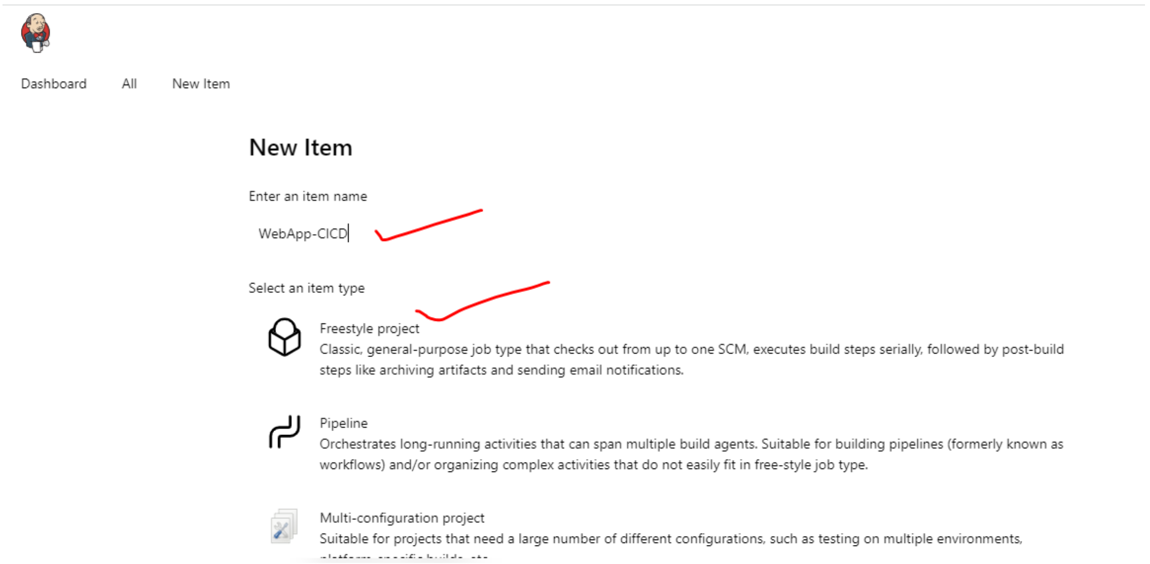
Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status


Dashboard All New Item

New Item

Enter an item name

WebApp-CICD

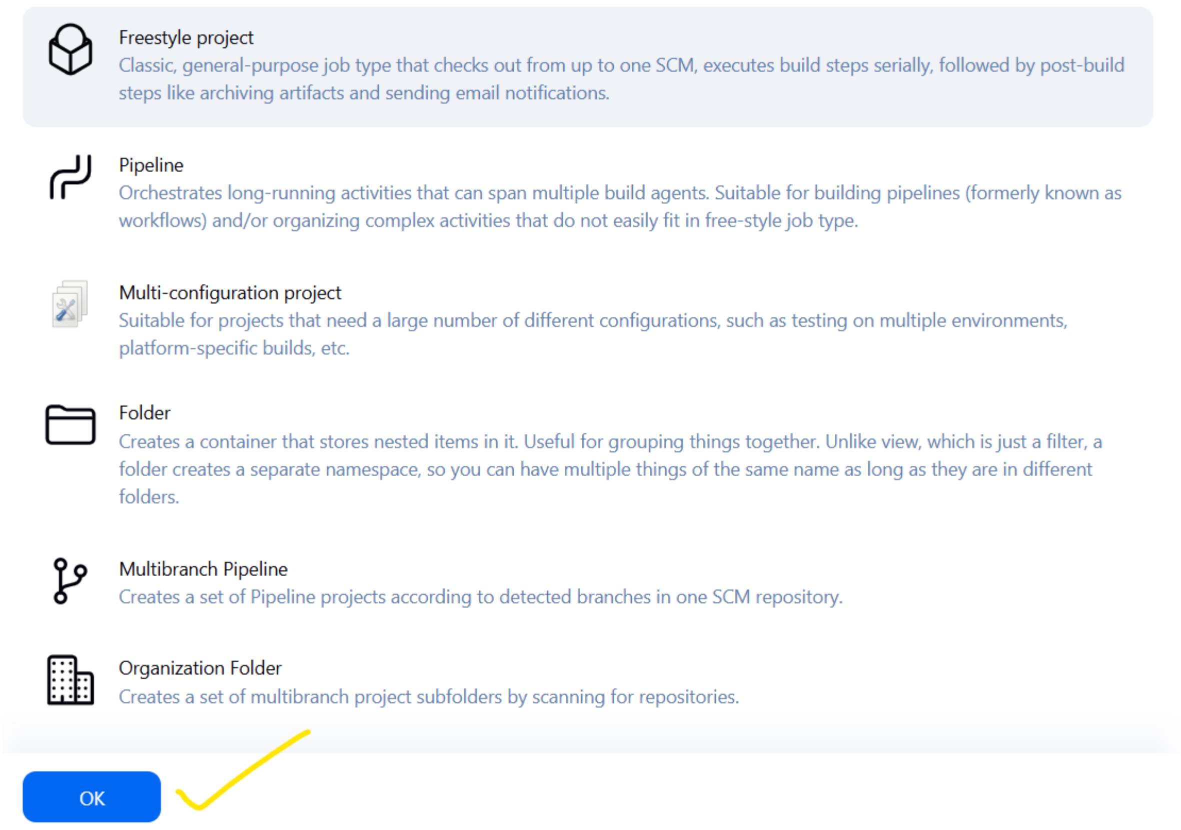
Select an item type

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.

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.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments,

Scroll down and click `OK`.


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.

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.

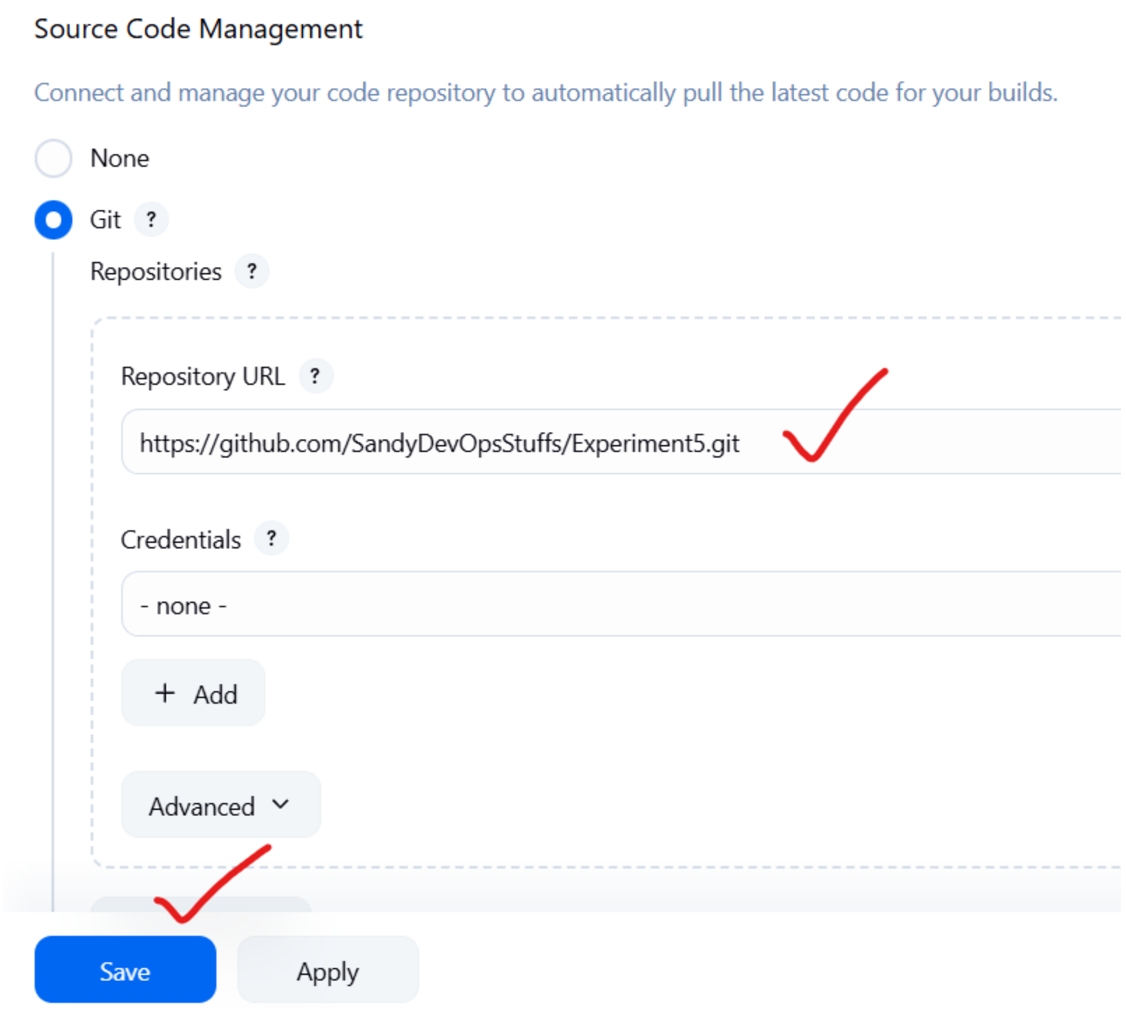
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

✓ In **Source Code Management**, select Git → add your repository URL



Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None [✓ Git]

Repositories

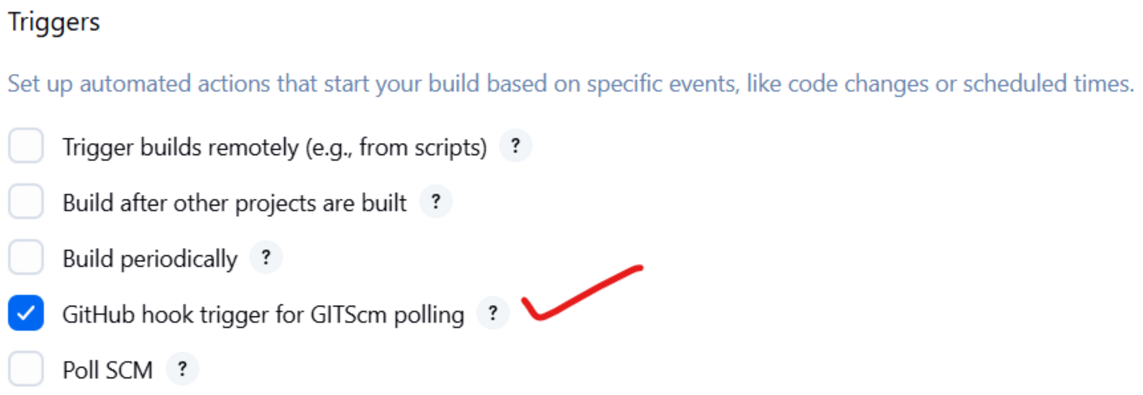
Repository URL: https://github.com/SandyDevOpsStuffs/Experiment5.git

Credentials: - none -

✓ Use credentials if needed.

- **Build Triggers**

✓ Select: **GitHub hook trigger for GITScm polling**



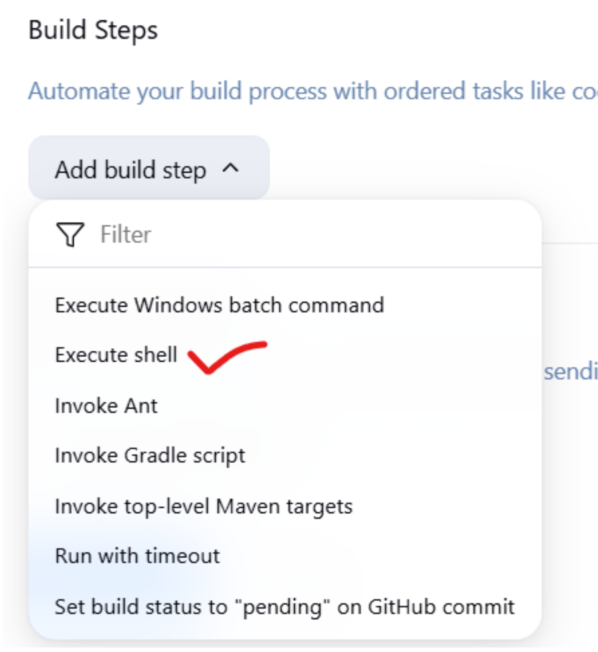
Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☑ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

- **Build Step**

Choose "Execute Shell"


Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

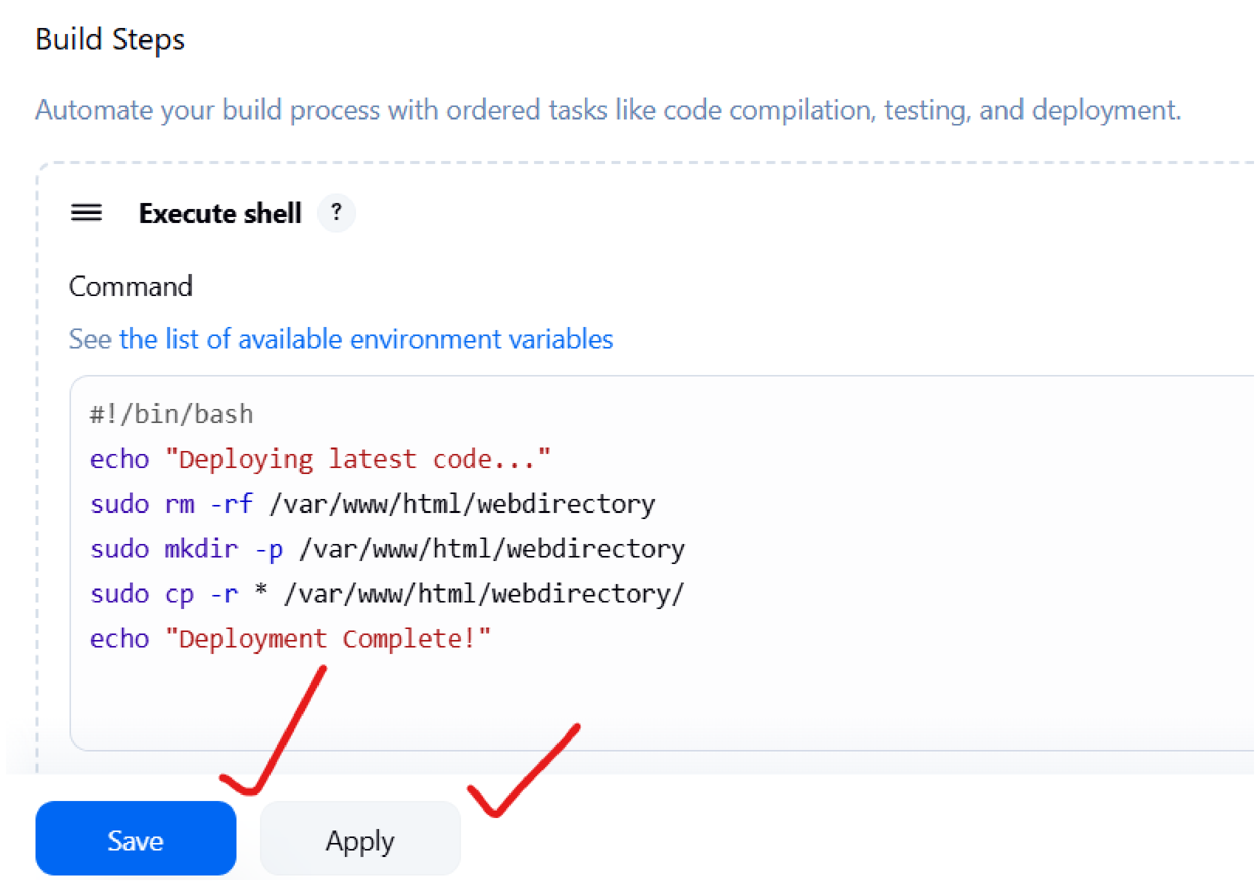
Add build step
Filter
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Run with timeout
Set build status to "pending" on GitHub commit

sending notifications, archiving artifacts, or triggering other jobs.

Enter the following:

```
#!/bin/bash

echo "Deploying latest code..."

sudo rm -rf /var/www/html/webdirectory

sudo mkdir -p /var/www/html/webdirectory

sudo cp -r * /var/www/html/webdirectory/

echo "Deployment Complete!"
```

Click `Save` and `Apply.`


Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Execute shell

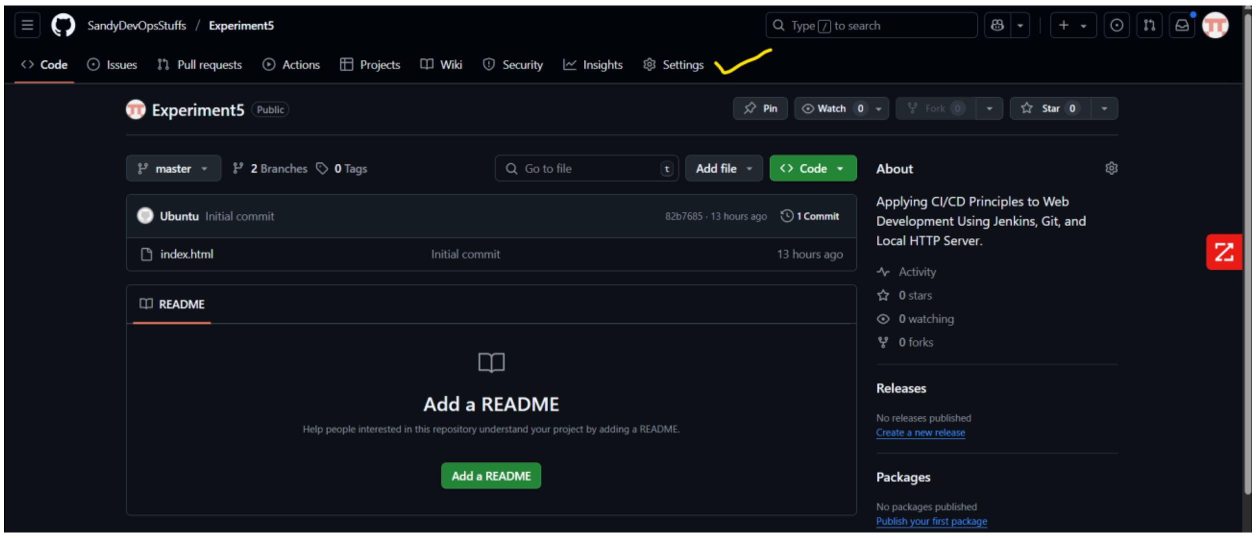
Command

See the list of available environment variables

```
#!/bin/bash
echo "Deploying latest code..."
sudo rm -rf /var/www/html/webdirectory
sudo mkdir -p /var/www/html/webdirectory
sudo cp -r * /var/www/html/webdirectory/
echo "Deployment Complete!"
```

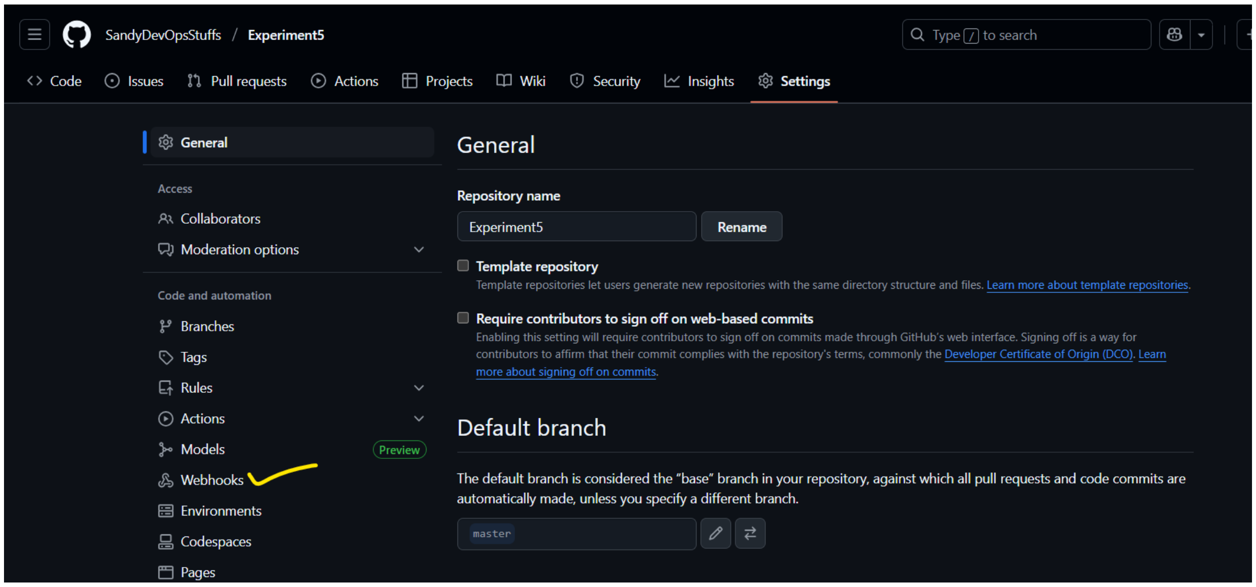
Step 6: Configure Webhook in GitHub

1. Go to your GitHub repo → **Settings** → **Webhooks**



Applying CI/CD Principles to Web Development Using Jenkins, Git, and Local HTTP Server.

Ubuntu Initial commit	82b7685 · 13 hours ago	1 Commit
index.html	Initial commit	13 hours ago



General

Repository name

Experiment5 Rename

Template repository
Template repositories let users generate new repositories with the same directory structure and files. Learn more about template repositories.

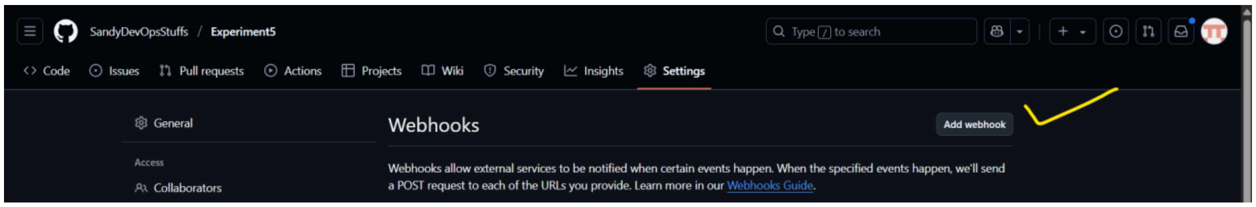
Require contributors to sign off on web-based commits
Enabling this setting will require contributors to sign off on commits made through GitHub's web interface. Signing off is a way for contributors to affirm that their commit complies with the repository's terms, commonly the Developer Certificate of Origin (DCO). Learn more about signing off on commits.

Default branch

The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

master

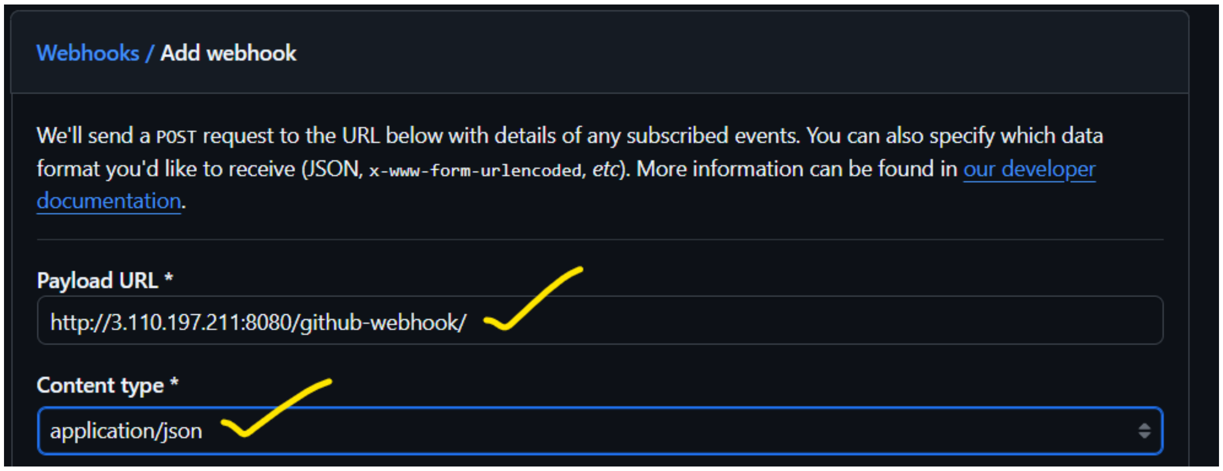
2. Click **Add webhook**



Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

- **Payload URL**: `http://<your-jenkins-ip>:8080/github-webhook/`
- **Content type**: `application/json`



Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, *etc*). More information can be found in our developer documentation.

Payload URL *

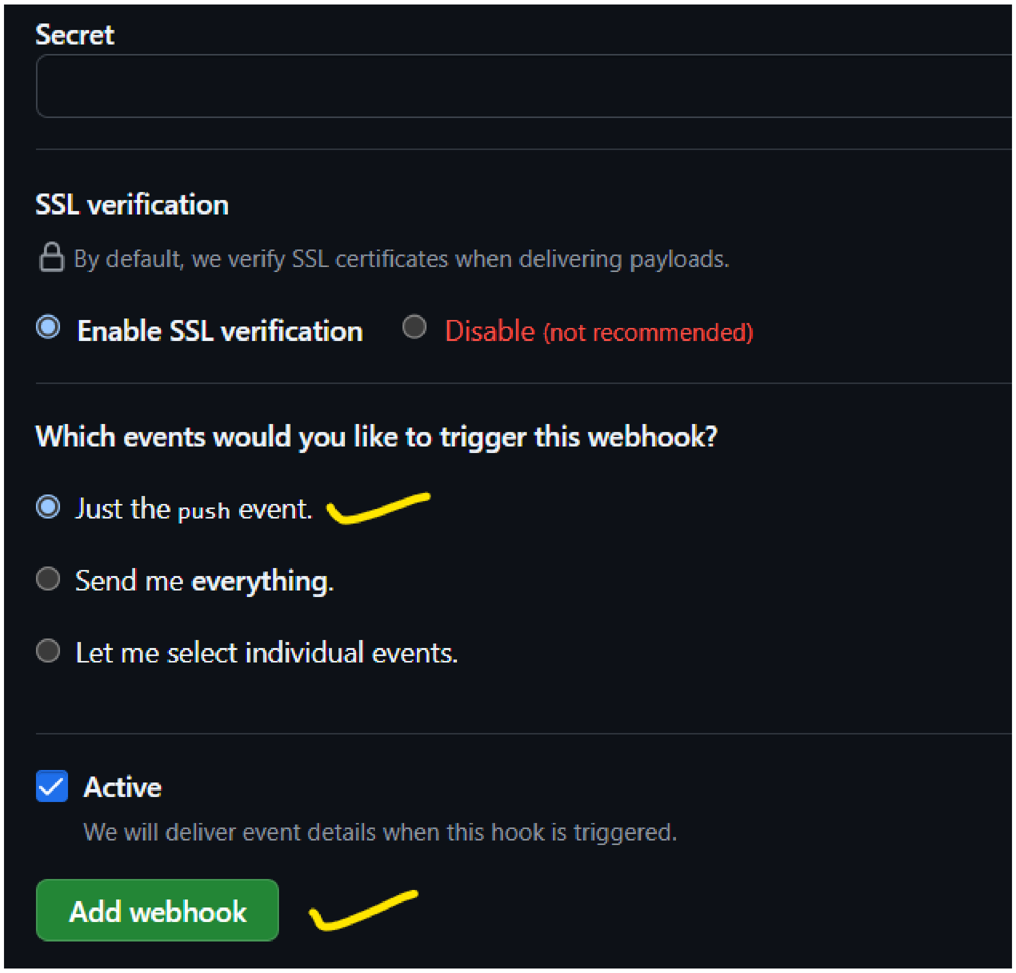
http://3.110.197.211:8080/github-webhook/

Content type *

application/json

NOTE: *Each time your EC2 instance restarts public IP will be changed. So, if you have taken a break or your EC2 instance is restarted then do not forget to edit the Payload URL accordingly.*

- **Events**: Just the push event
- Add webhook



Secret

SSL verification

By default, we verify SSL certificates when delivering payloads.

[✓ Enable SSL verification] Disable (not recommended)

Which events would you like to trigger this webhook?

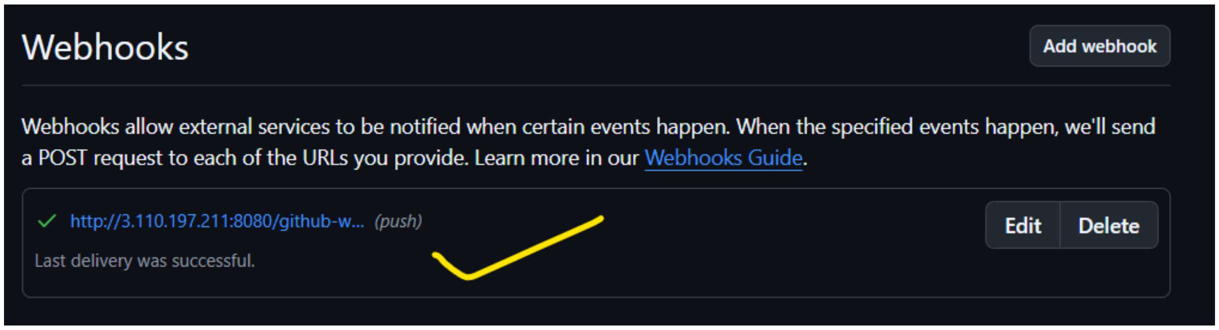
[✓ Just the push event.]

Send me everything.

Let me select individual events.

[✓ Active]

We will deliver event details when this hook is triggered.



Webhooks

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

✓ http://3.110.197.211:8080/github-w... (push)

Last delivery was successful.

***NOTE:** Ensure Jenkins is accessible from GitHub (use **ngrok** or deploy Jenkins on a public IP for remote tests).*

Step 7: Trigger the Pipeline

- **Edit the sudoers file**

  Run on your EC2 terminal (not inside Jenkins):

  ```
  sudo visudo
  ```

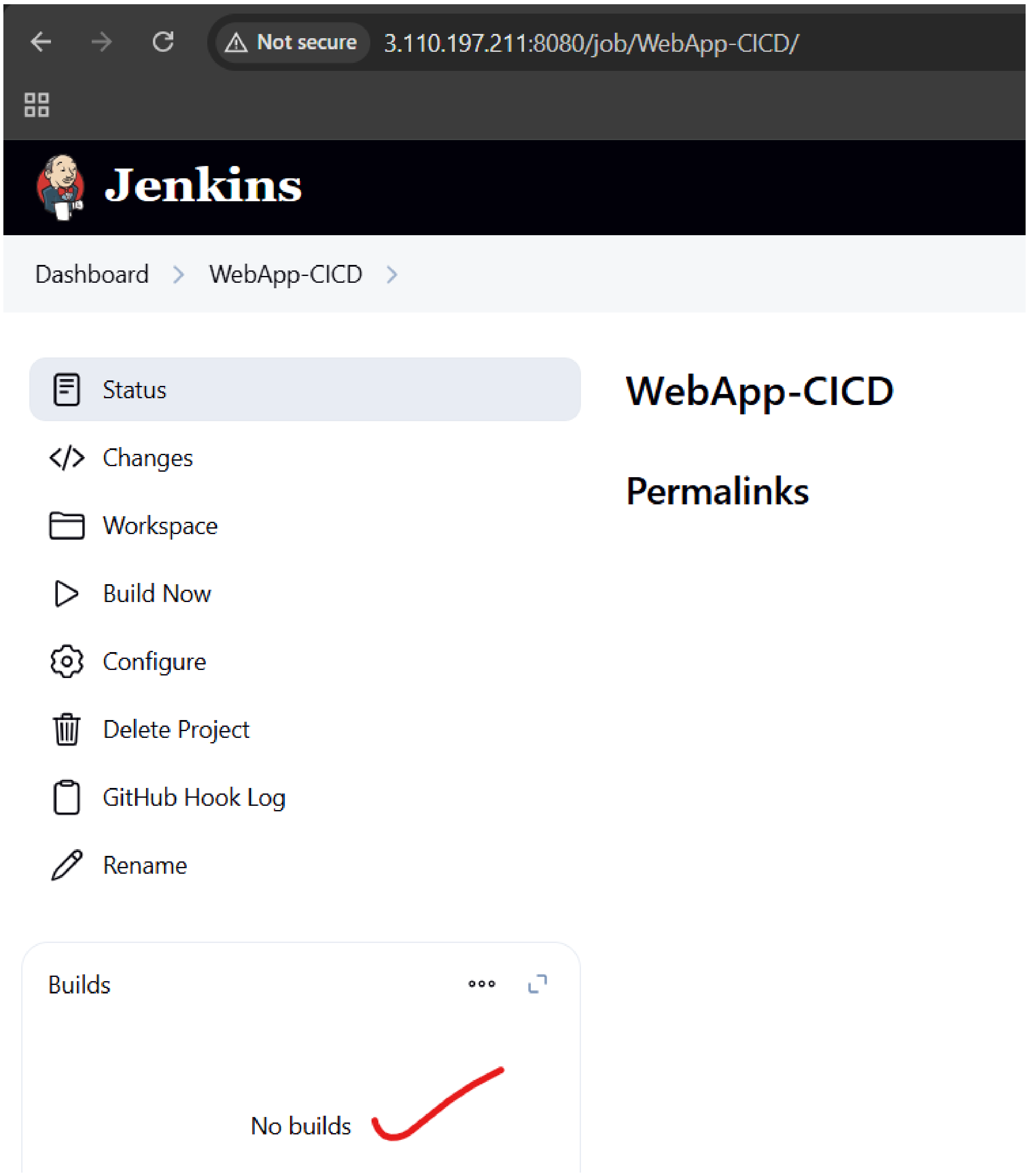
  At the end of the file, **add this line**:

  ```
  jenkins ALL=(ALL) NOPASSWD: /bin/rm, /bin/mkdir, /bin/cp
  ```

  Press `Ctrl+O` and `Ctrl+X` to save and exit.

  This allows the jenkins user to run rm, mkdir, and cp with sudo **without prompting for a password**. This is secure because it's limited to only those commands.

- Now confirm that there are no builds yet in Jenkins since this is our first build.

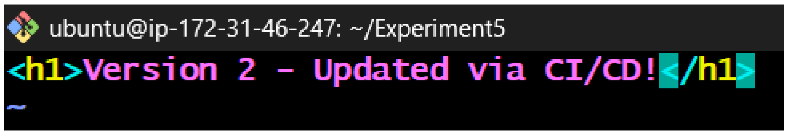


- Now edit index.html which will be copied from current directory `Experiement5` to `/var/www/html` later.

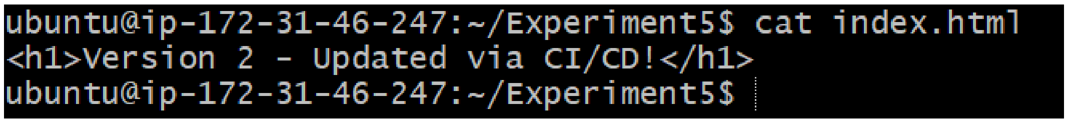
```
vi index.html
```

Replace the old content with the following new content:

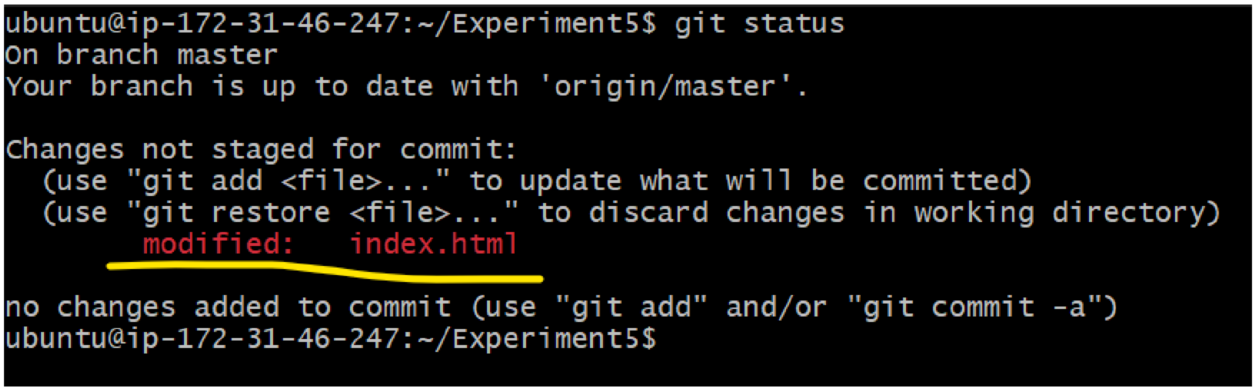
```
<h1>Version 2 - Updated via CI/CD!</h1>
```


```
ubuntu@ip-172-31-46-247: ~/Experiment5
<h1>Version 2 - Updated via CI/CD!</h1>
~
```

cat index.html

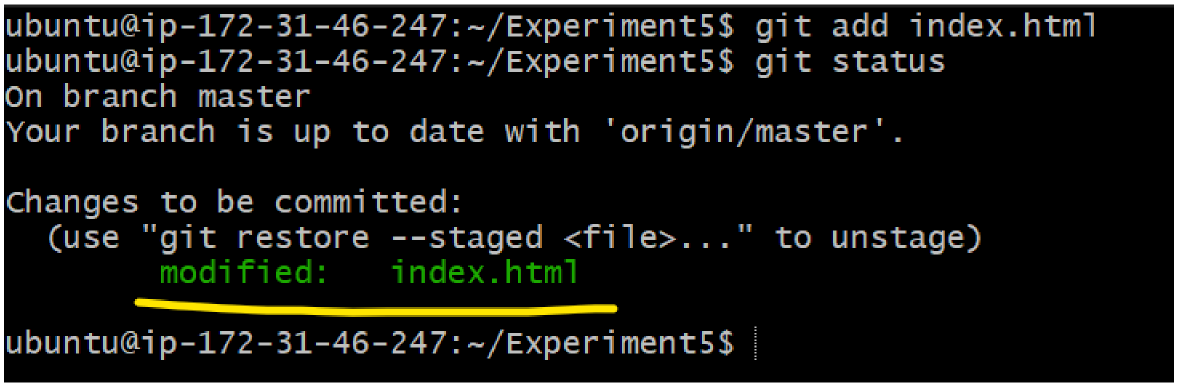

```
ubuntu@ip-172-31-46-247:~/Experiment5$ cat index.html
<h1>Version 2 - Updated via CI/CD!</h1>
ubuntu@ip-172-31-46-247:~/Experiment5$
```

git status


```
ubuntu@ip-172-31-46-247:~/Experiment5$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
ubuntu@ip-172-31-46-247:~/Experiment5$
```

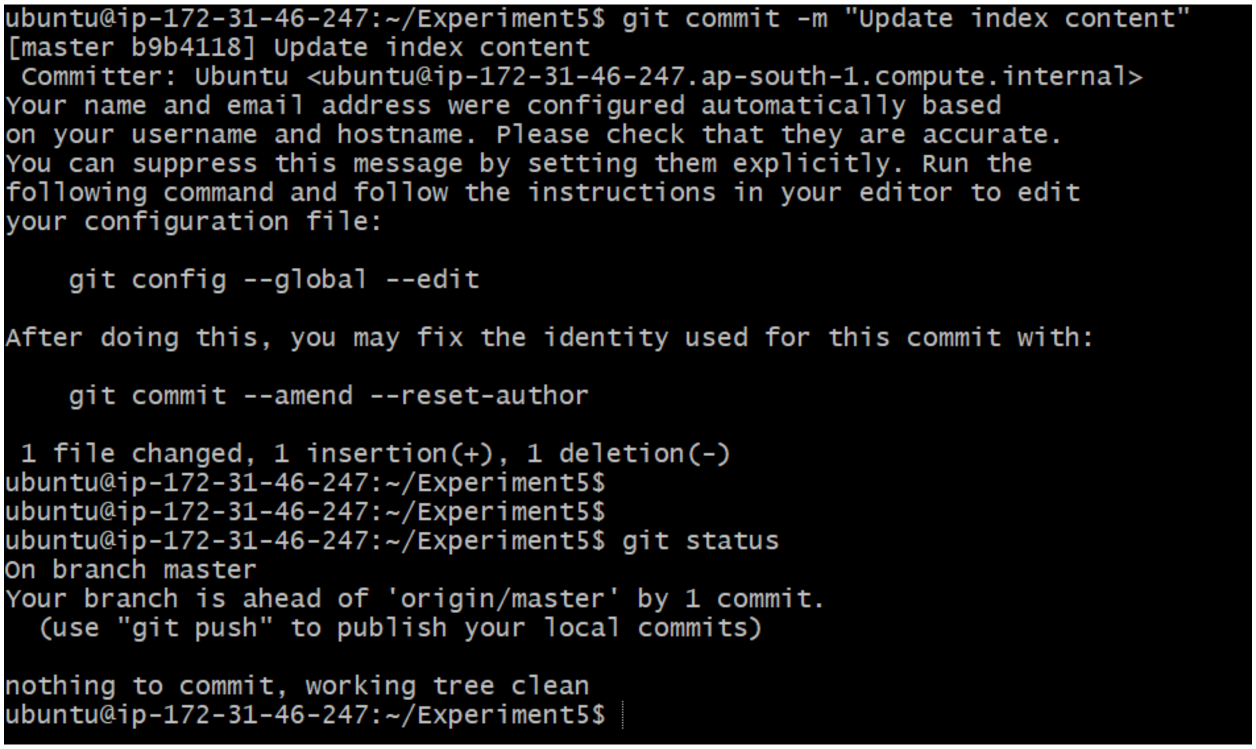
git add index.html

git status


```
ubuntu@ip-172-31-46-247:~/Experiment5$ git add index.html
ubuntu@ip-172-31-46-247:~/Experiment5$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html

ubuntu@ip-172-31-46-247:~/Experiment5$
```

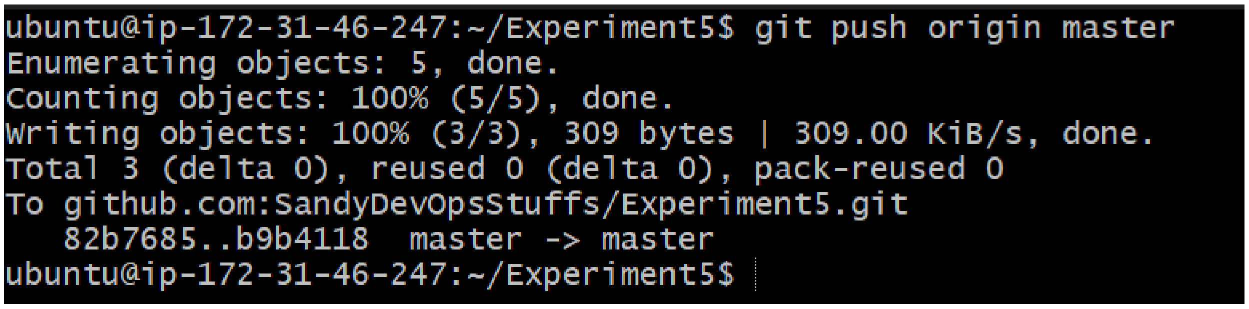
git commit -m "Update index content"

git status



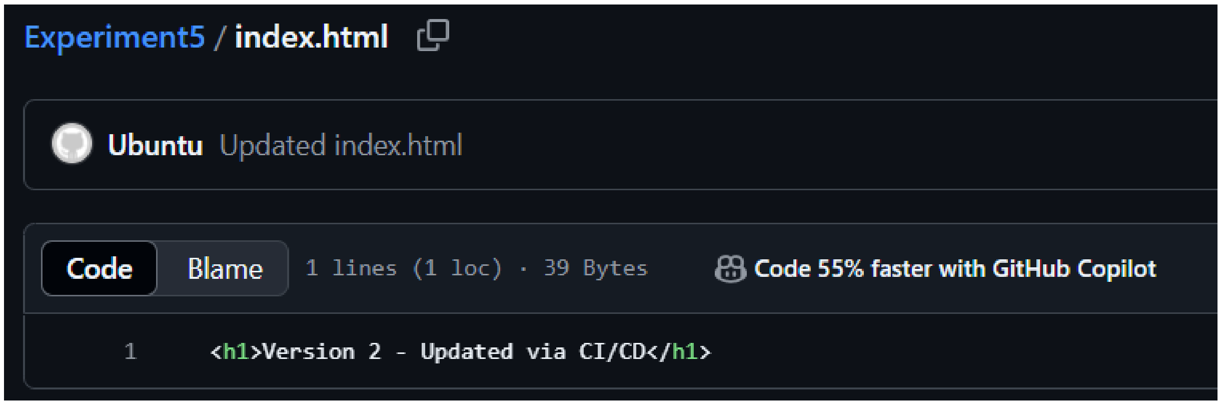
```
ubuntu@ip-172-31-46-247:~/Experiment5$ git commit -m "Update index content"
[master b9b4118] Update index content
 Committer: Ubuntu <ubuntu@ip-172-31-46-247.ap-south-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+), 1 deletion(-)
ubuntu@ip-172-31-46-247:~/Experiment5$
ubuntu@ip-172-31-46-247:~/Experiment5$
ubuntu@ip-172-31-46-247:~/Experiment5$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
ubuntu@ip-172-31-46-247:~/Experiment5$
```

```
git push origin master
```



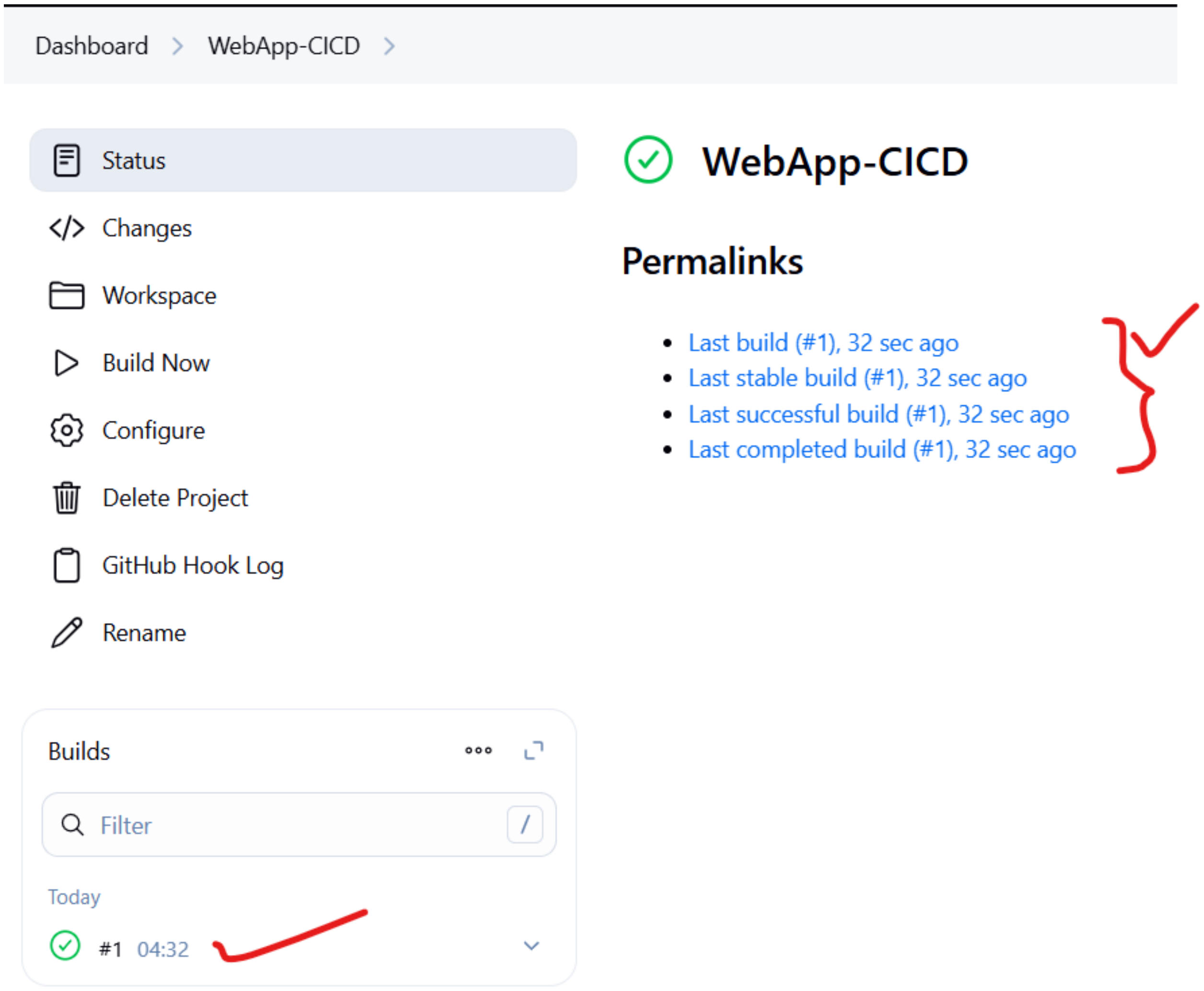
```
ubuntu@ip-172-31-46-247:~/Experiment5$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 309 bytes | 309.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:SandyDevOpsStuffs/Experiment5.git
   82b7685..b9b4118  master -> master
ubuntu@ip-172-31-46-247:~/Experiment5$
```

Confirm that the latest code is pushed.


Experiment5 / index.html
Ubuntu Updated index.html
Code
Blame
1 lines (1 loc) · 39 Bytes
Code 55% faster with GitHub Copilot

```
<h1>Version 2 - Updated via CI/CD</h1>
```

This should trigger your Jenkins job automatically.

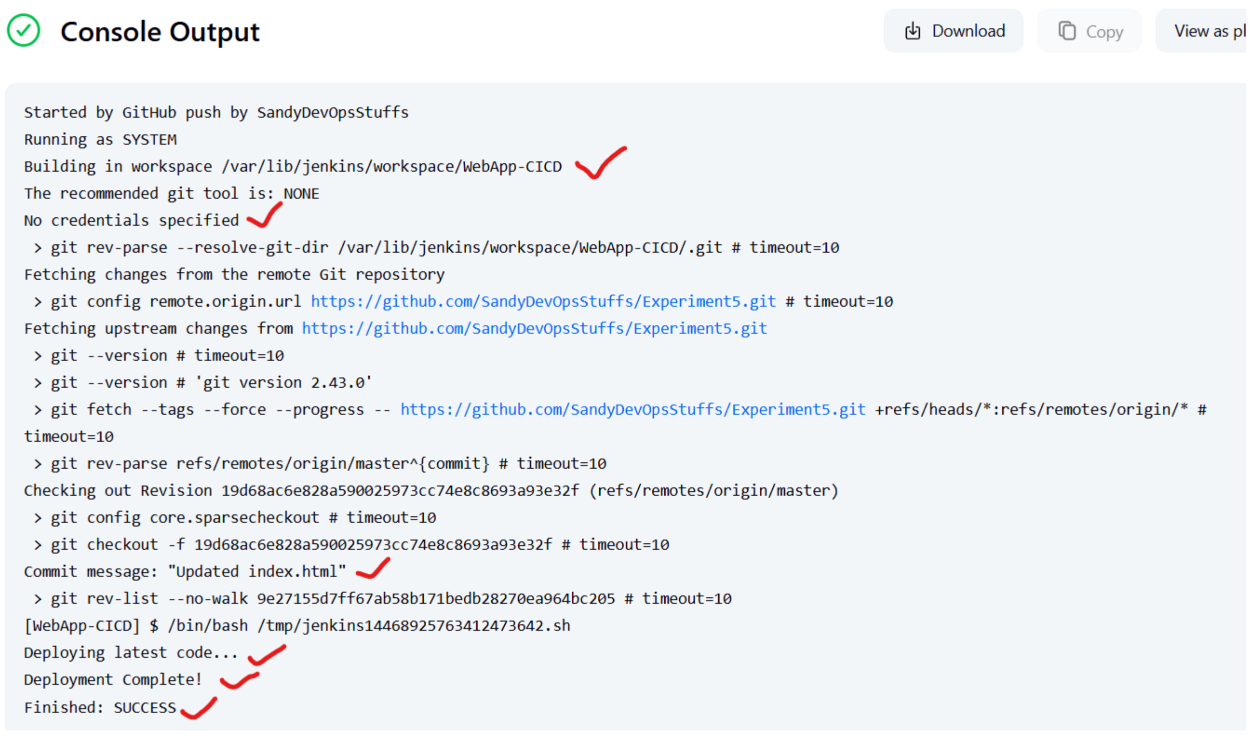


WebApp-CICD

Permalinks

- Last build (#1), 32 sec ago
- Last stable build (#1), 32 sec ago
- Last successful build (#1), 32 sec ago
- Last completed build (#1), 32 sec ago

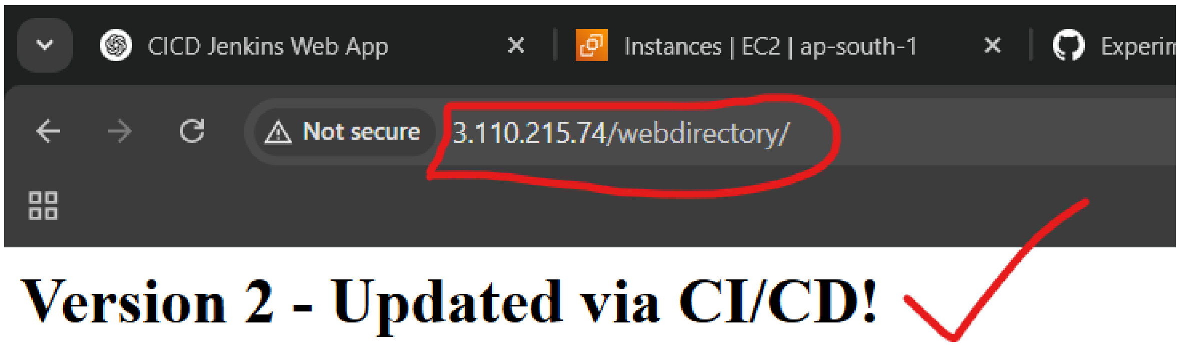
#1 04:32



```
Started by GitHub push by SandyDevOpsStuffs
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/WebApp-CICD
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/WebApp-CICD/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/SandyDevOpsStuffs/Experiment5.git # timeout=10
Fetching upstream changes from https://github.com/SandyDevOpsStuffs/Experiment5.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/SandyDevOpsStuffs/Experiment5.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 19d68ac6e828a590025973cc74e8c8693a93e32f (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 19d68ac6e828a590025973cc74e8c8693a93e32f # timeout=10
Commit message: "Updated index.html"
 > git rev-list --no-walk 9e27155d7ff67ab58b171bedb28270ea964bc205 # timeout=10
[WebApp-CICD] $ /bin/bash /tmp/jenkins14468925763412473642.sh
Deploying latest code...
Deployment Complete!
Finished: SUCCESS
```

Step 8: Verify the CI/CD Pipeline

- Open browser → Visit http://<Public_IP_of_EC2_Instance>/webdirectory/
- You should see the **updated content** deployed automatically by Jenkins.



Version 2 - Updated via CI/CD!

Conclusion

This experiment demonstrates a simple CI/CD pipeline that:

- Pulls code from GitHub via a webhook
- Builds and deploys to a local HTTP server

- Uses Jenkins as the automation server

This approach forms the base for real-world CI/CD practices and can be extended to support test automation, Docker, cloud servers, and more.