.

Lesson 01 Demo 01

Implementing the DevOps Model

Objective: To implement DevOps using GitHub to store a Java program and Jenkins to build consistent code packages, enabling continuous integration

Tools required: Git, GitHub, and Jenkins

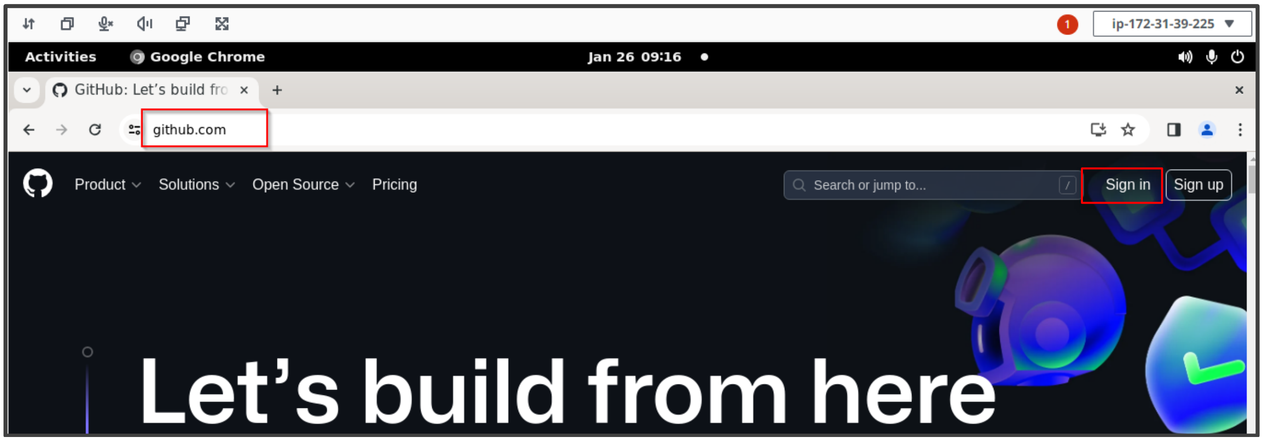
Prerequisites: None

Steps to be followed:

1. Create a GitHub repository
2. Add a Java program to the repository
3. Create a freestyle build job in Jenkins
4. Build the Java program with Jenkins

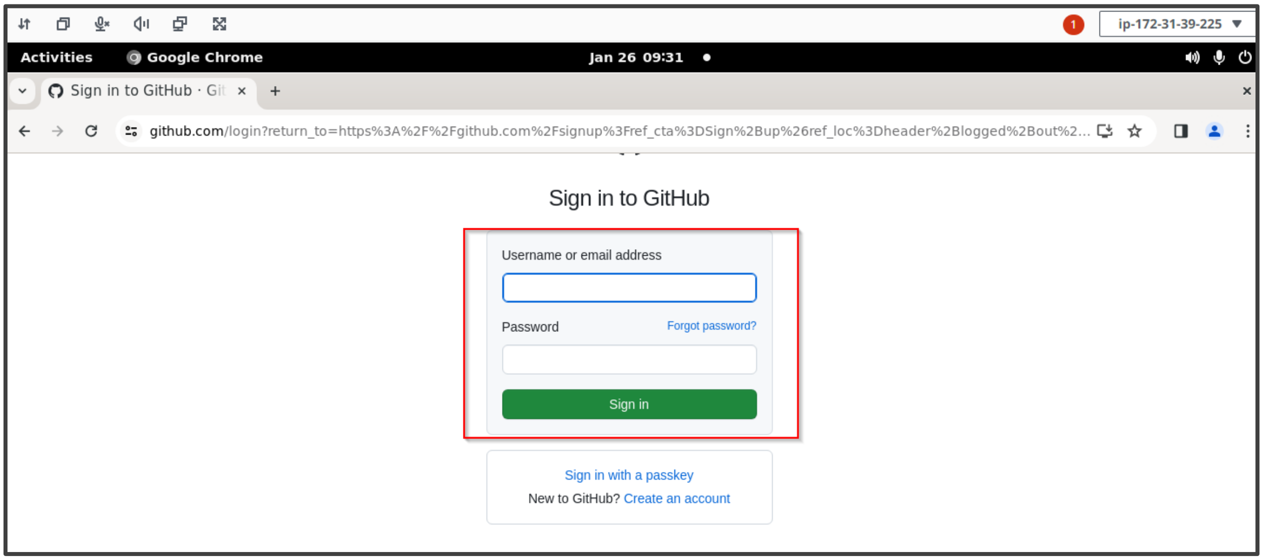
Step 1: Create a GitHub repository

1.1 Open the browser in your lab, go to https://github.com, and click on the **Sign in** button

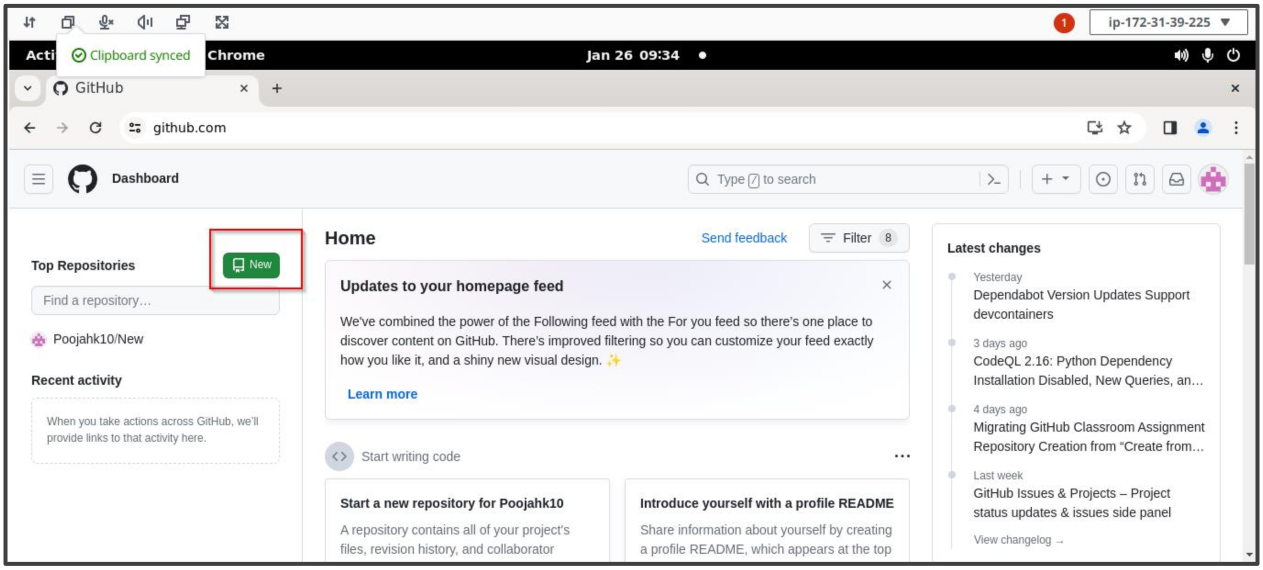


.

1.2 Enter the credentials of your GitHub account and click on **Sign in**


ip-172-31-39-225
Activities
Google Chrome
Jan 26 09:31
Sign in to GitHub · Git
github.com/login?return_to=https%3A%2F%2Fgithub.com%2Fsignup%3Fref_cta%3DSign%2Bup%26ref_loc%3Dheader%2Blogged%2Bout%2...
Sign in to GitHub
Username or email address
Password
Forgot password?
Sign in
Sign in with a passkey
New to GitHub? Create an account

1.3 Click on **New** as shown in the screenshot below:



When you take actions across GitHub, we'll provide links to that activity here.

Updates to your homepage feed

We've combined the power of the Following feed with the For you feed so there's one place to discover content on GitHub. There's improved filtering so you can customize your feed exactly how you like it, and a shiny new visual design. ✨

Learn more

Start a new repository for Poojahk10

A repository contains all of your project's files, revision history, and collaborator

Introduce yourself with a profile README

Share information about yourself by creating a profile README, which appears at the top

Yesterday
Dependabot Version Updates Support devcontainers

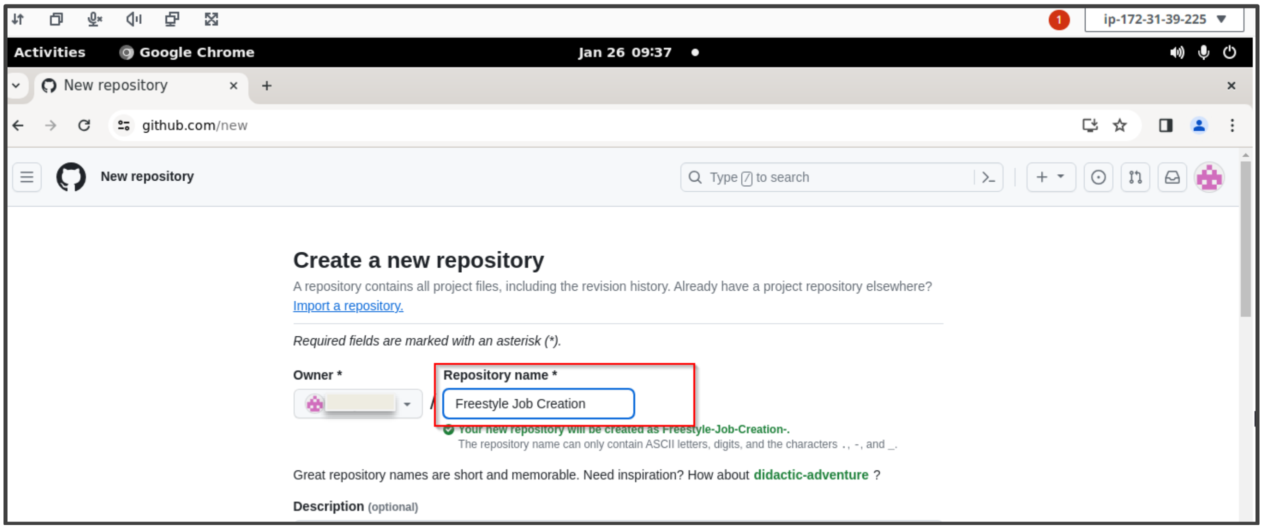
3 days ago
CodeQL 2.16: Python Dependency Installation Disabled, New Queries, an…

4 days ago
Migrating GitHub Classroom Assignment Repository Creation from "Create from…

Last week
GitHub Issues & Projects – Project status updates & issues side panel

.

1.4 Add the **Repository name** as shown in the screenshot below:



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (*).

Owner * [redacted] / Repository name * Freestyle Job Creation

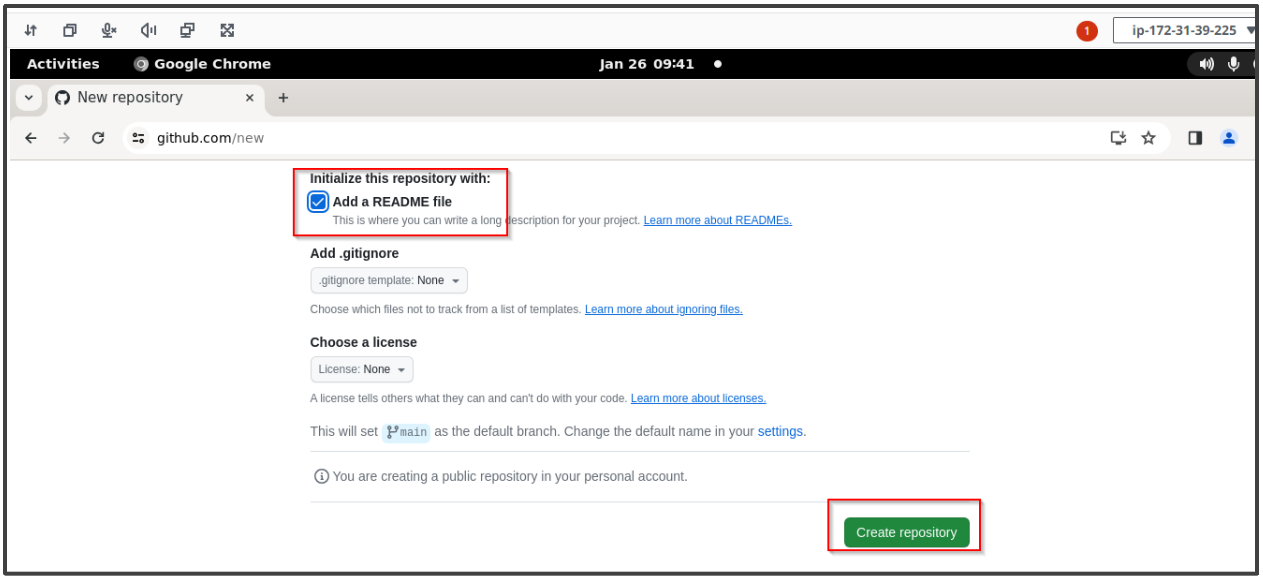
Your new repository will be created as Freestyle-Job-Creation-.

The repository name can only contain ASCII letters, digits, and the characters ., -, and _.

Great repository names are short and memorable. Need inspiration? How about didactic-adventure ?

Description (optional)

1.5 Select the check box of **ADD a README file** and click on **Create repository**



Initialize this repository with:

☑ **Add a README file**

This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license

License: None

A license tells others what they can and can't do with your code. Learn more about licenses.

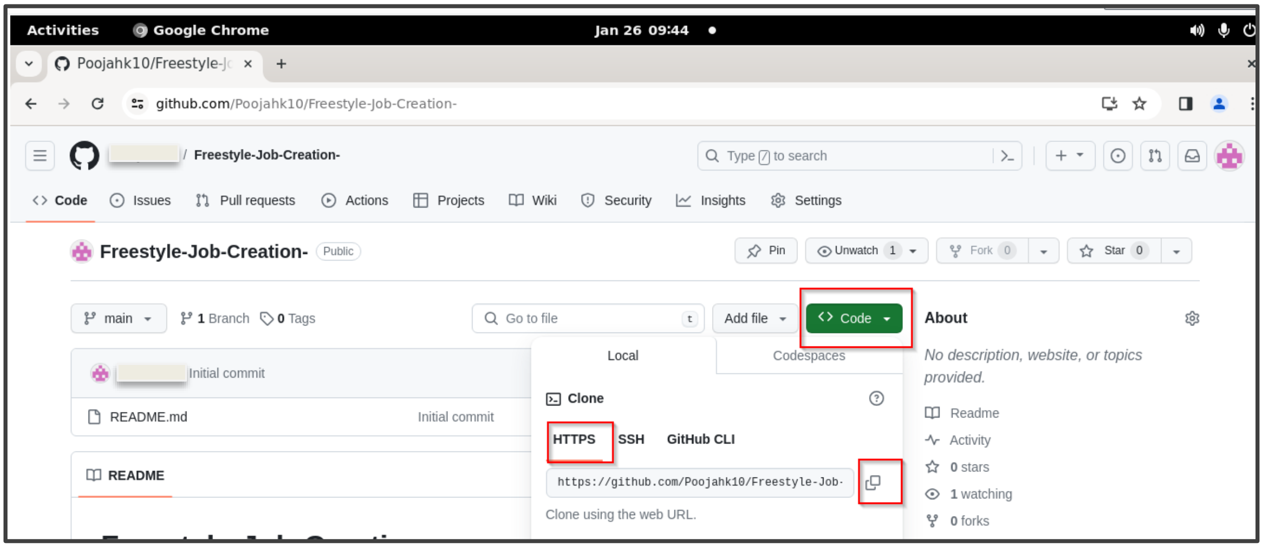
This will set `main` as the default branch. Change the default name in your settings.

You are creating a public repository in your personal account.

Create repository

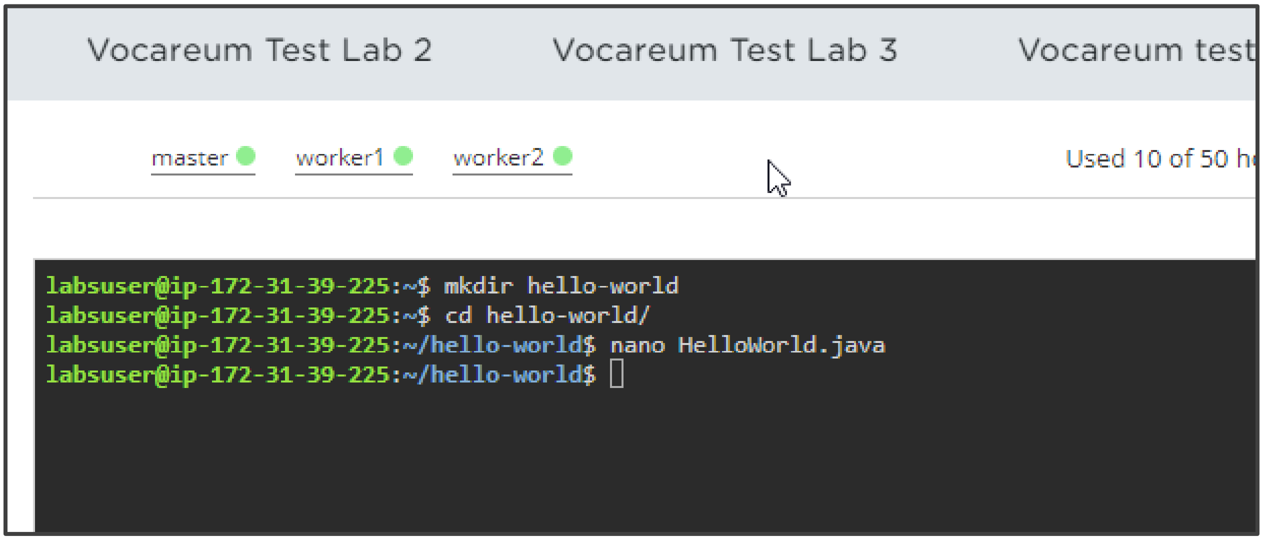
.

1.6 Click on **<> Code**, then **HTTPS**, and finally copy the repository URL



Step 2: Add a Java program to the repository

2.1 Open the terminal, run the following commands to create a directory, navigate to the **hello-world** directory, and open the Java file in a text editor as shown in the screenshot below:
mkdir hello-world
cd hello-world
nano HelloWorld.java

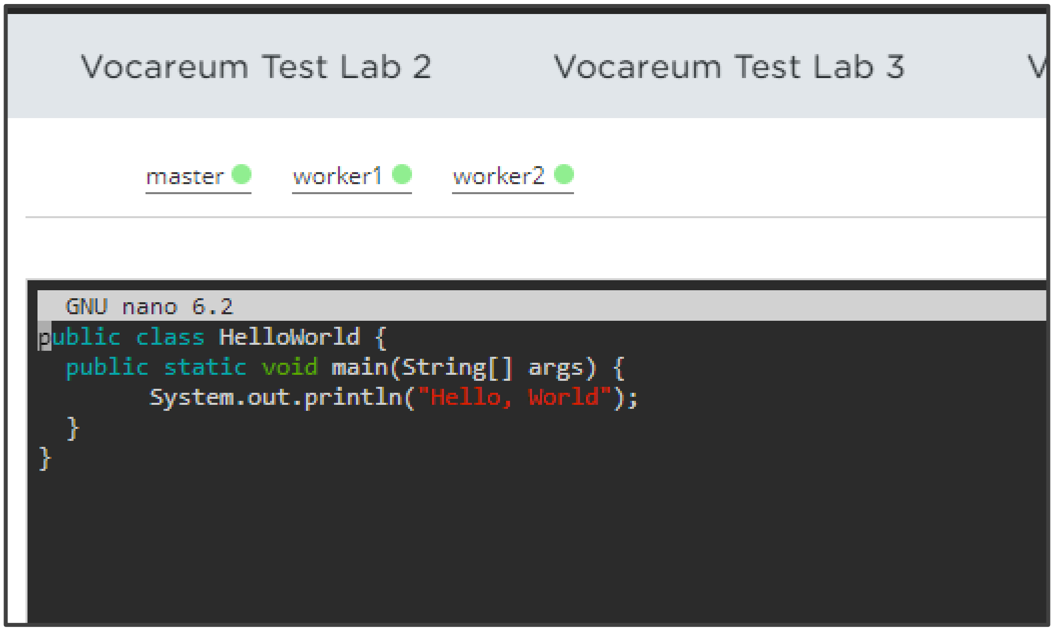


```
labsuser@ip-172-31-39-225:~$ mkdir hello-world
labsuser@ip-172-31-39-225:~$ cd hello-world/
labsuser@ip-172-31-39-225:~/hello-world$ nano HelloWorld.java
labsuser@ip-172-31-39-225:~/hello-world$ 
```

.

2.2 Copy and paste the below code into the file, save the file, and exit from the text editor:

```
public class HelloWorld {
  public static void main(String[] args) {
          System.out.println("Hello, World");
  }
 }
```



```
 GNU nano 6.2
public class HelloWorld {
  public static void main(String[] args) {
        System.out.println("Hello, World");
  }
}
```

2.3 Run the following commands:

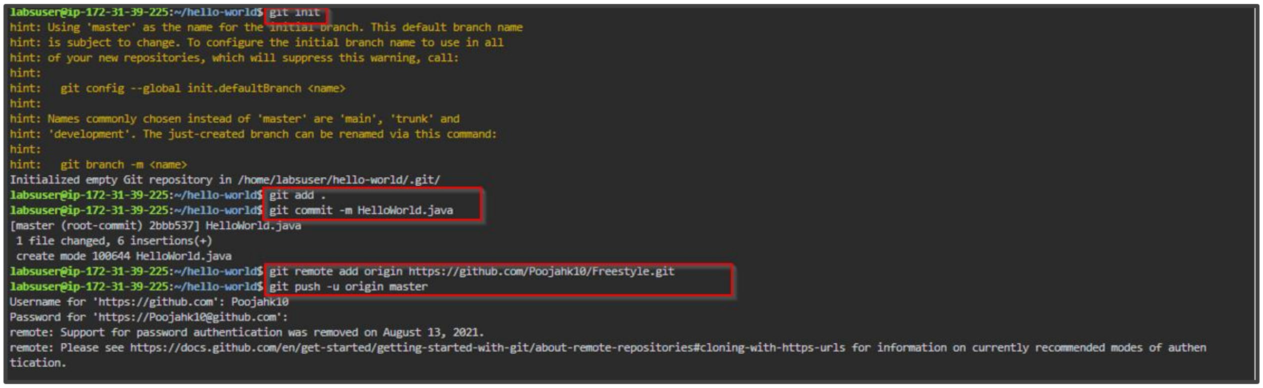
git init

git add .

git commit -m “Add new files”

git remote add origin <Repository_URL>

git push -u origin master

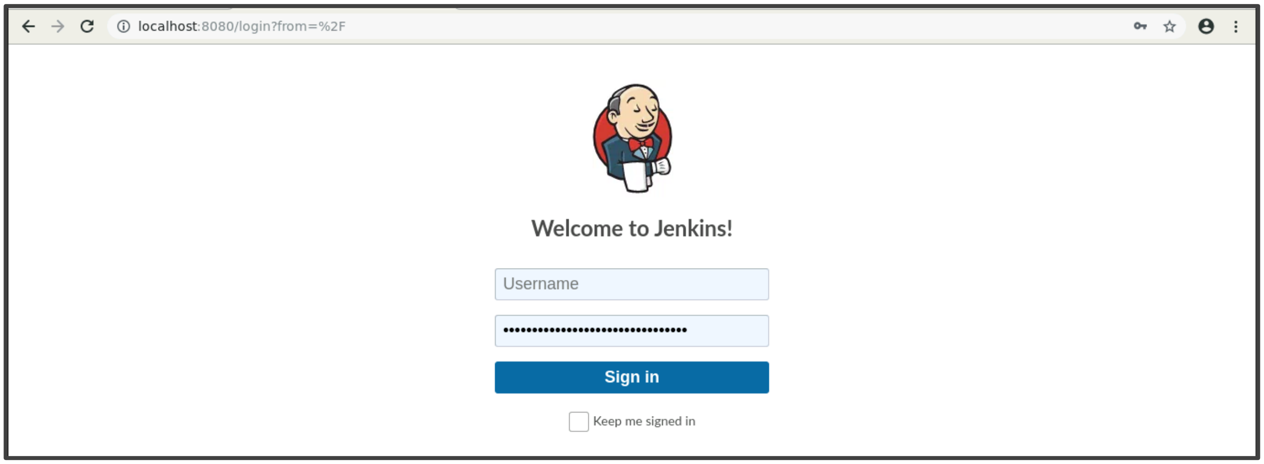


```
labsuser@ip-172-31-39-225:~/hello-world$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   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/labsuser/hello-world/.git/
labsuser@ip-172-31-39-225:~/hello-world$ git add .
labsuser@ip-172-31-39-225:~/hello-world$ git commit -m HelloWorld.java
[master (root-commit) 2bbb537] HelloWorld.java
 1 file changed, 6 insertions(+)
 create mode 100644 HelloWorld.java
labsuser@ip-172-31-39-225:~/hello-world$ git remote add origin https://github.com/Poojahk10/Freestyle.git
labsuser@ip-172-31-39-225:~/hello-world$ git push -u origin master
Username for 'https://github.com': Poojahk10
Password for 'https://Poojahk10@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently recommended modes of authen
tication.
```

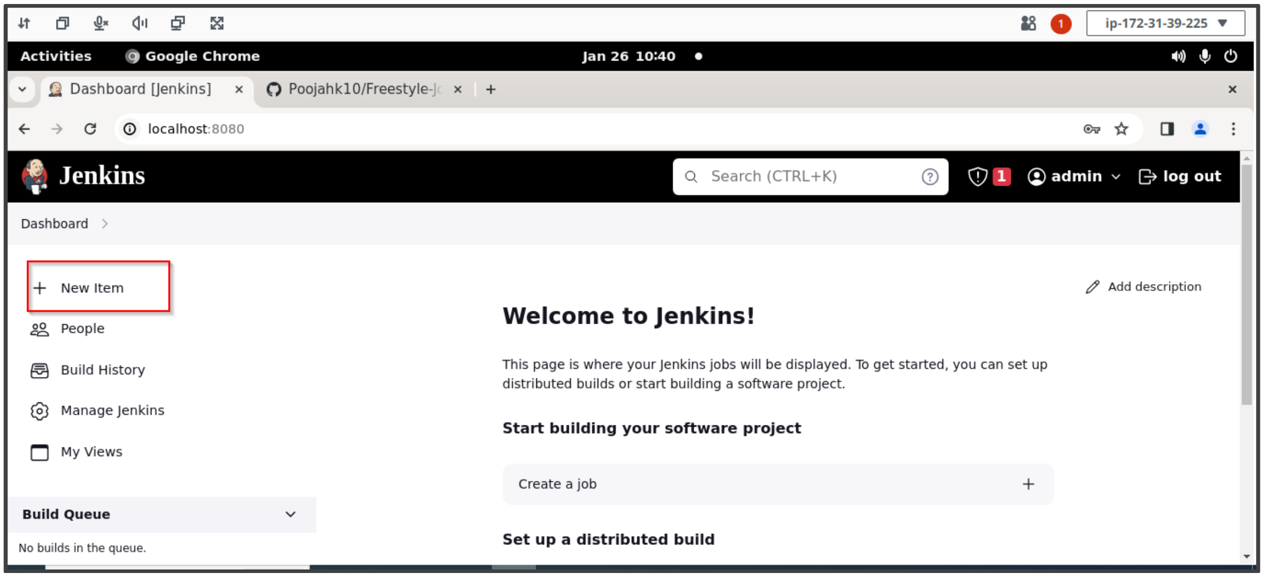
Note: Ensure that the password to be added is your **GitHub** account **Token**

Step 3: Create a freestyle build job in Jenkins

3.1 Open the browser, type **localhost:8080**; this will open Jenkins. Provide the credentials and then click on **Sign in**



3.2 Click on **New Item** in the Jenkins Dashboard



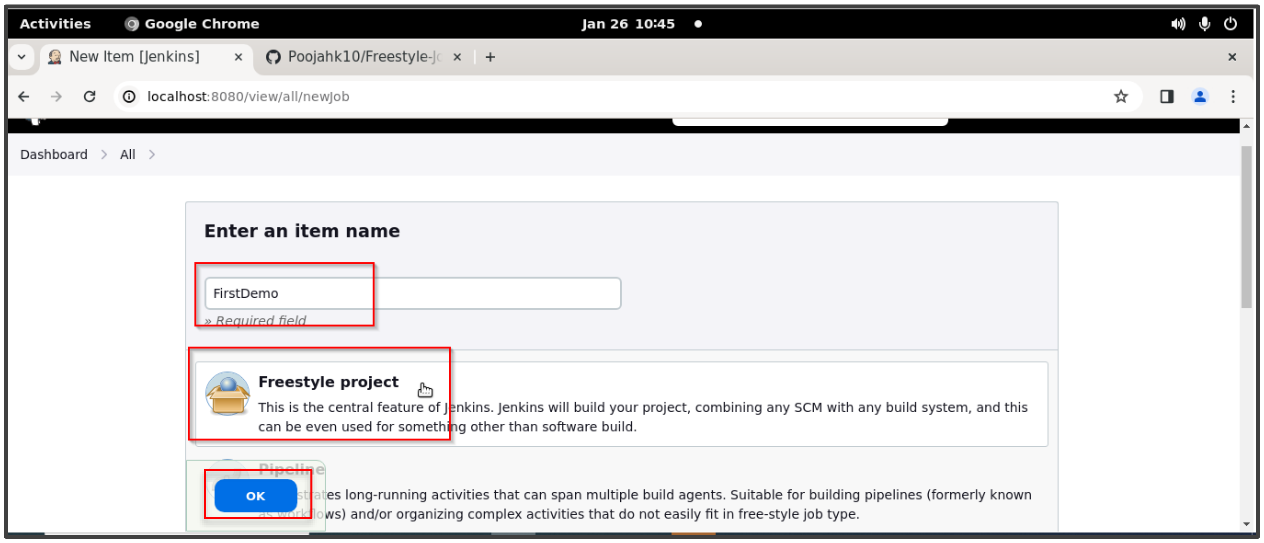
Welcome to Jenkins!

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

Set up a distributed build

3.3 Enter a name for your project, select **Freestyle project** as the build job type, and click on **OK**



Enter an item name

FirstDemo

» Required field

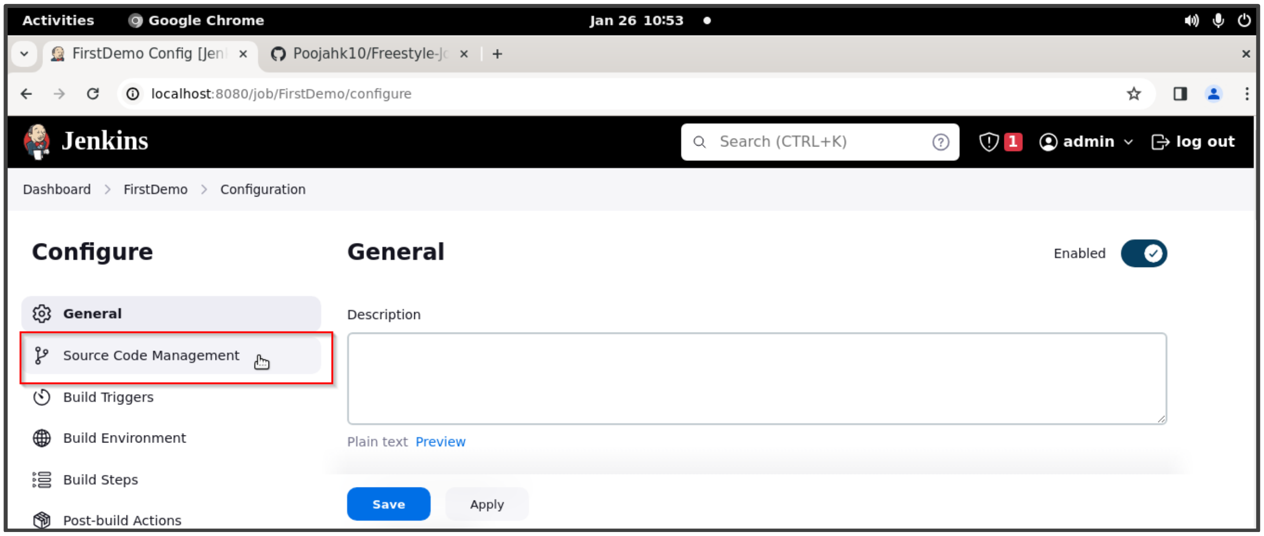
Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline

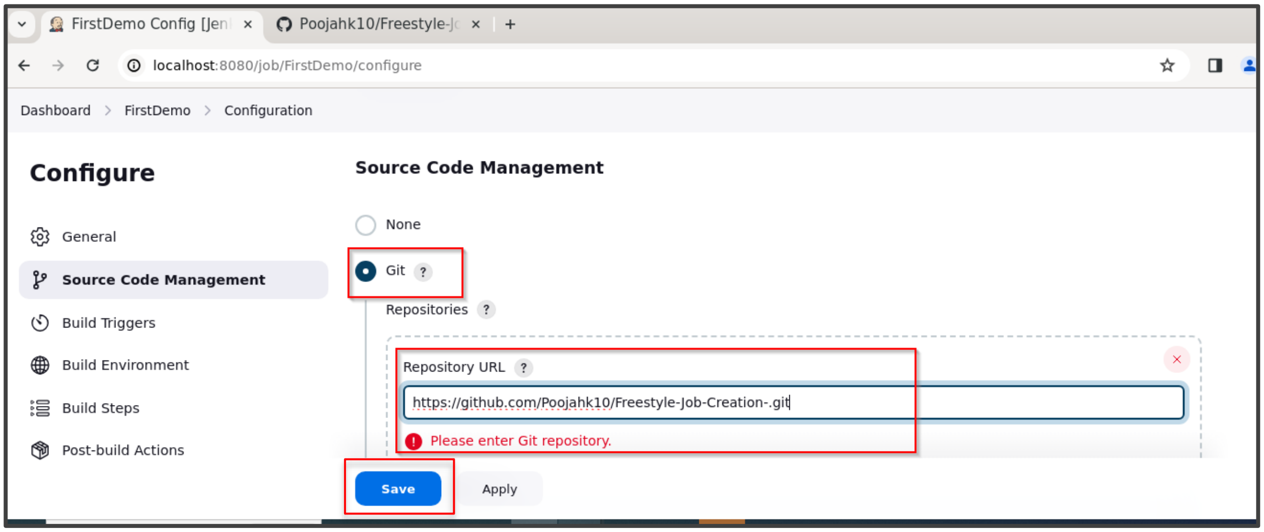
...strates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

3.4 Click on **Source Code Management**



.

3.5 Select **Git**, enter the **Repository URL**, and then click on **Save**



Source Code Management

○ None

● Git

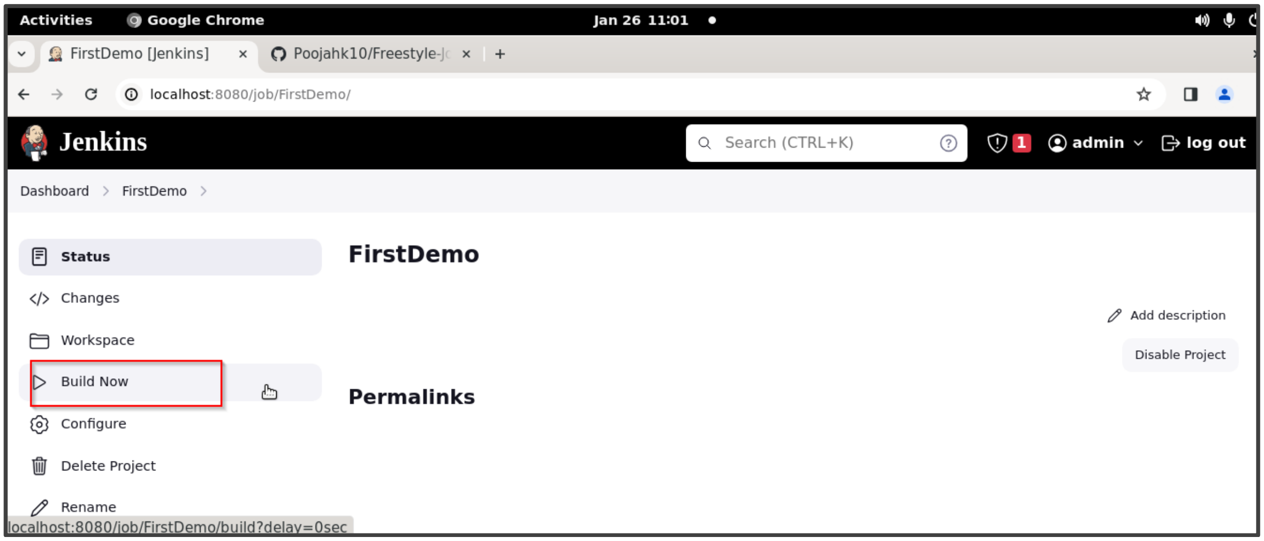
Repositories

Repository URL: https://github.com/Poojahk10/Freestyle-Job-Creation-.git

Please enter Git repository.

Step 4: Build the Java program with Jenkins

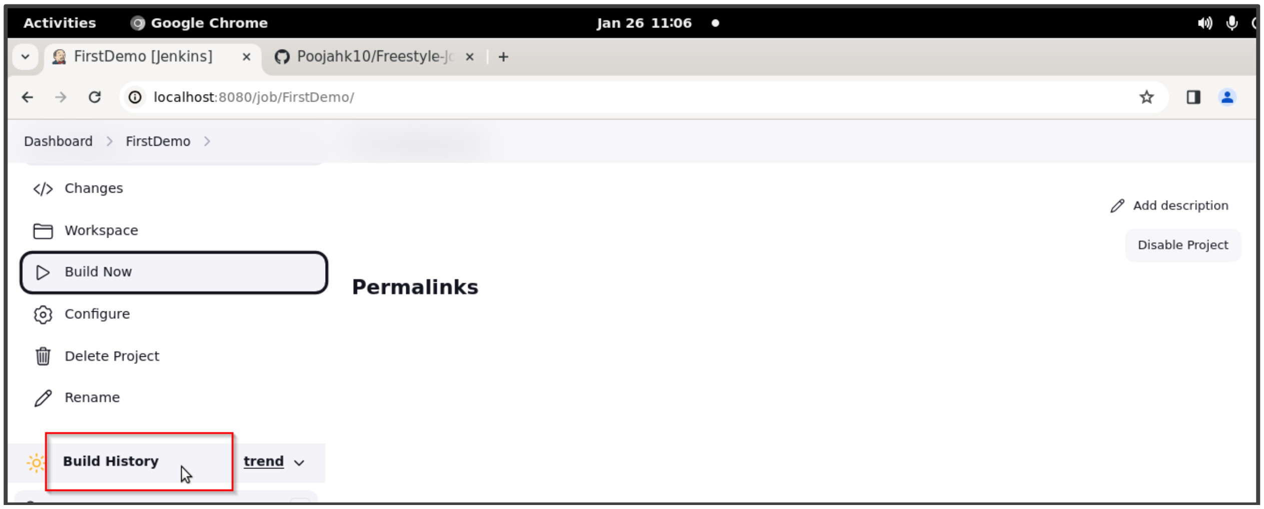
4.1 Click on **Build Now** to build your project



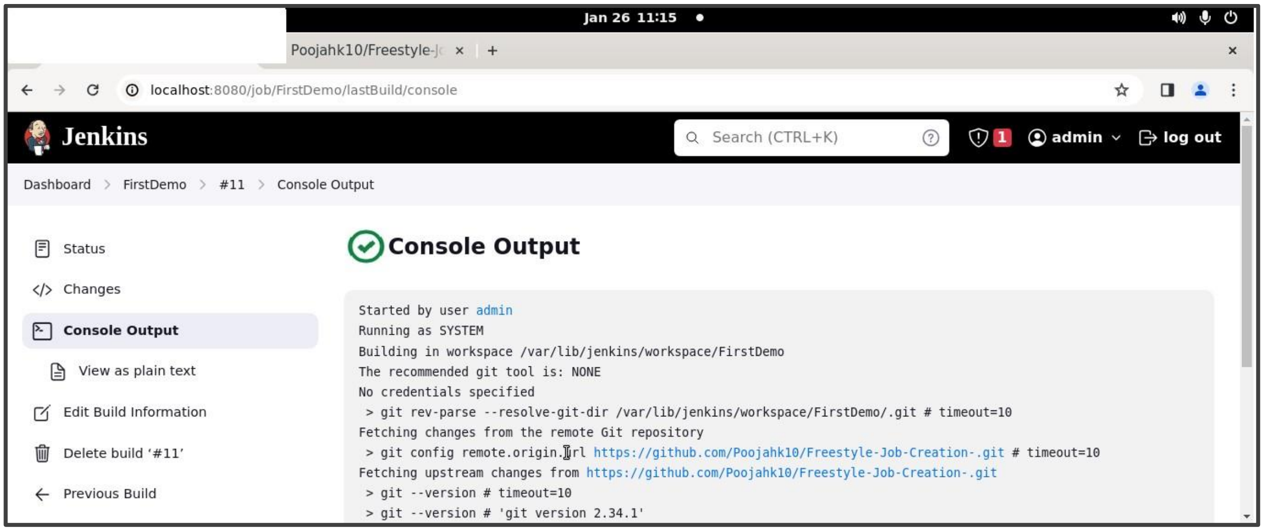
localhost:8080/job/FirstDemo/build?delay=0sec

.

4.2 Click on **Build History** to view the build results



4.3 Click on the **Console Output** to view the build logs



```
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/FirstDemo
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/FirstDemo/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/Poojahk10/Freestyle-Job-Creation-.git # timeout=10
Fetching upstream changes from https://github.com/Poojahk10/Freestyle-Job-Creation-.git
 > git --version # timeout=10
 > git --version # 'git version 2.34.1'
```

By following these steps, you have successfully implemented DevOps using GitHub to store a Java program and Jenkins to build consistent code packages, enabling continuous integration.

.