CAN-ID : CAN\_33898338  
NAME: Ravi Anantapur  
College : BLDEA's V P Dr PG Halakatti College of Engineering & Technology

College in Bijapur, Karnataka

This is an elaborate, step-by-step procedure on how to set up a Jenkins job for a CI/CD pipeline from source, using GitHub as your source control, in as simple steps as possible with all the relevant commands and explanations.

**Step 1: Install Jenkins**

1. Download Jenkins:

No need to go the Jenkins download page to fetch your relevant version.

**2. Install Jenkins:**

No Problem.

In case you are running on an **Ubuntu/Debian** system use:

**sudo apt update**

**sudo apt install openjdk-11-jdk -y # Make sure that java is installed**

**wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -**

**sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ >> /etc/apt/sources.list.d/jenkins.list'**

**sudo apt update**

**sudo apt install jenkins -y**

o For **Windows**:

 Jenkins should now be visible in Services. You can then start it through Services **MGR** as detailed in the next page

**3. Run Jenkins:**

o For **Linux**:

**sudo systemctl start jenkins**

**sudo systemctl enable jenkins**

o For **Windows**:

 It should be automatic but once done, you may want to get started by using services **MGR**:

**4. Open Jenkins:**

o Open up your favorite web browser and point to: http://localhost:8080 or the IP / hostname of the Jenkins Server 5. Unlocks Jenkins:

o Copy the initial admin password **from /var/lib/jenkins/secrets/initialAdminPassword** on Linux or the secrets folder in the Jenkins directory on Windows.

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

**6. Complete Setup Wizard:**

o Install Suggested Plugins: In setup click, "Install Suggested Plugins" to install necessary plugins like Git.

**Step 2: Setup Git Plugin in Jenkins**

**1. Install the Git Plugin:**

o **Go to Dashboard > Manage Jenkins > Manage Plugins > Available.**

o Locate Git Plugin and install. No restart of Jenkins is required.

2.**Config Git in Jenkins**:

Go to **Manage Jenkins > Global Tool Configuration**

Under Git, click on Add Git and then type the name and path of Git. In case the global installation is available for Git, it will automatically be discovered by Jenkins.

**Step 3: Jenkins connect with GitHub Repository**

**1.Get Personal Access Token for GitHub**

path Go to **GitHub > Settings > Developer settings > Personal Access Tokens > Tokens (classic) > Generate new token.**

o Provide a name, select repo scope, and generate the token. Copy the token as it will be used in Jenkins.

2. Add GitHub Credentials in Jenkins:

o **Go to Dashboard > Manage Jenkins > Manage Credentials > (select scope) > Add Credentials.**

o Select Kind: Secret text and paste the GitHub token, give it an ID (e.g., **GitHubToken**), and save.

**Step 4: Create a New Jenkins Job with GitHub Integration**

**1. Create a New Job:**

o From the Jenkins Dashboard, click New Item.

o Input a job name, select Freestyle project, and click OK.

**2. Configure Git Repository:**

o Under Source Code Management section, select Git.

o Paste the GitHub repository URL.

o Under Credentials, select the GitHub token created earlier.

o Input the branch to build, e.g., main or master.

**3. Set up Build Triggers:**

o Under Build Triggers, select GitHub hook trigger for GITScm polling. This will automatically trigger builds when code is pushed to the repository.

**4. Configure Build Steps:**

Under Build, click Add build step and select Execute shell.

Enter commands for your build, test, and deploy process. For example:

# Example build command

**./gradlew build** # Replace with relevant build commands

**Step 5: Set up GitHub Webhook to Trigger Jenkins Builds**

1. Get the Jenkins Webhook URL:

Your Jenkins webhook URL is typically: **http://<jenkins\_server\_ip>:8080/github-webhook/**

**2. Configure Webhook in GitHub**:

Go to your GitHub repository, **click on Settings > Webhooks > Add webhook**.

Input the Jenkins webhook URL.

Select **application/json** as the content type and select **Push to trigger it**.

**Step 6: Save and Test**

**1.** **Save the Jenkins Job**:

o Click on Save at the bottom of the configuration page

**2. Build it**:

o In Jenkins click Build Now to trigger the build manually.

o Else, you could make a change in your GitHub repository and push that change which would trigger a build.

**3. Check the Build Logs**:

o. Open Build History and then click on the build number to see the console output for debugging and verification

It will now pull code from GitHub, build it and, at the same time, have automated version control with CI/CD integration by means of Jenkins.  
  
Github repository link :