**AIM**: Creating CI/CD pipeline using Jenkins.

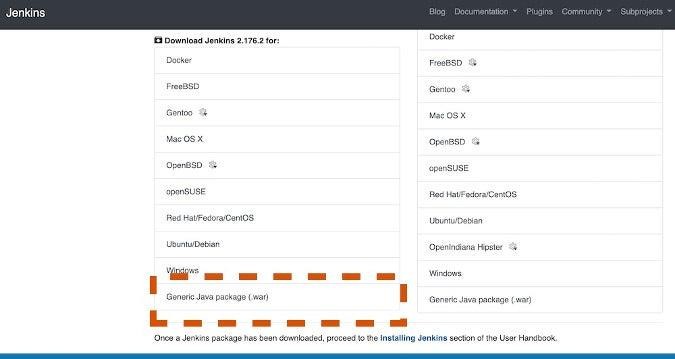
DESCRIPTION:

What is jenkins?

Jenkins is an open-source automation tool for Continuous Integration (CI) and Continuous Deployment (CD). It is a server-based system that runs in servlet containers like Apache Tomcat. Jenkins is one of the most used DevOps tools used along with other cloudnative tools and allows developers to build, test and deploy software seamlessly. What is CI/CD Pipeline? A pipeline is a process that drives software development through a path of building, testing, and deploying code, also known as CI/CD. By automating the process, the objective is to minimize human error and maintain a consistent process for how software is released.

Step 1: Download Jenkins

Navigate to the [Jenkins download page.](https://jenkins.io/download/) Scroll down to **Generic Java package (.war)** and click on it to download the file; save it someplace where you can locate it easily. (If you choose another Jenkins distribution, the rest of tutorial steps should be pretty much the same, except for Step 2.) The reason to use the WAR file is that it is a one-time executable file that is easily executable and removable.



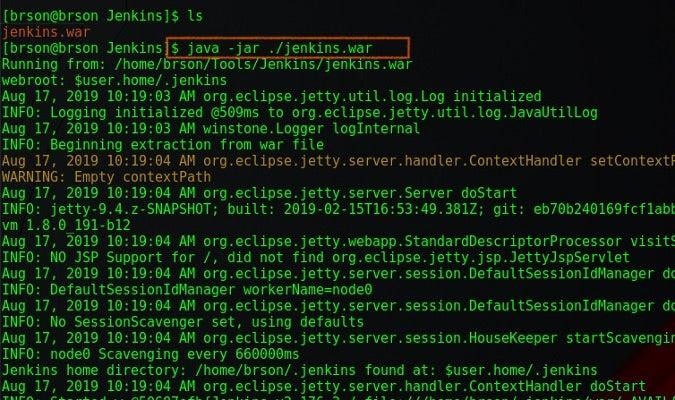
Step 2: Execute Jenkins as a Java binary

Open a terminal window and enter the directory where you downloaded Jenkins with **cd**

**<your path>**. (Before you proceed, make sure JDK is installed and added to the environment path.) Execute the following command, which will run the WAR file as an executable binary:

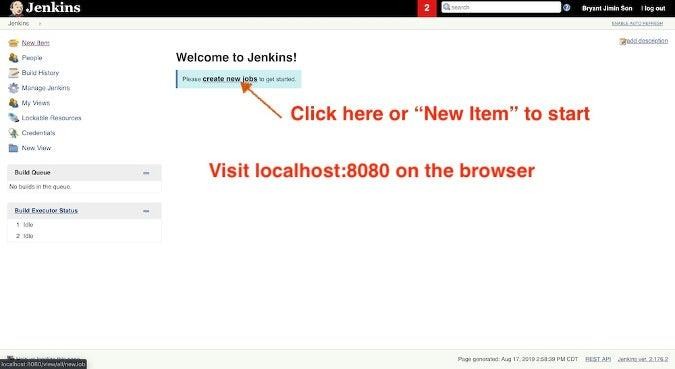
**java** -jar .**/**jenkins.war

If everything goes smoothly, Jenkins should be up and running at the default port 8080.



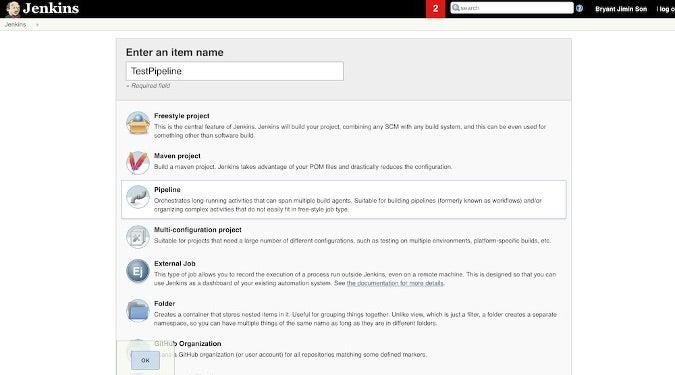
Step 3: Create a new Jenkins job

Open a web browser and navigate to **localhost:8080**. Unless you have a previous Jenkins installation, it should go straight to the Jenkins dashboard. Click **Create New Jobs**. You can also click **New Item** on the left.



Step 4: Create a pipeline job

In this step, you can select and define what type of Jenkins job you want to create. Select **Pipeline** and give it a name (e.g., TestPipeline). Click **OK** to create a pipeline job.



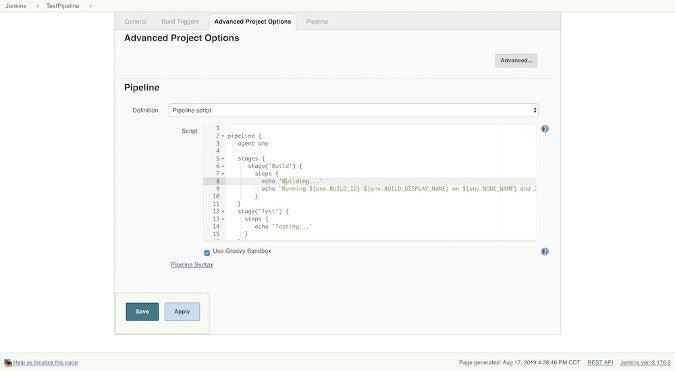
You will see a Jenkins job configuration page. Scroll down to find **Pipeline section**. There are two ways to execute a Jenkins pipeline. One way is by *directly writing a pipeline script* on Jenkins, and the other way is by retrieving the *Jenkins file from SCM* (source control management). We will go through both ways in the next two steps.

Step 5: Configure and execute a pipeline job through a direct script

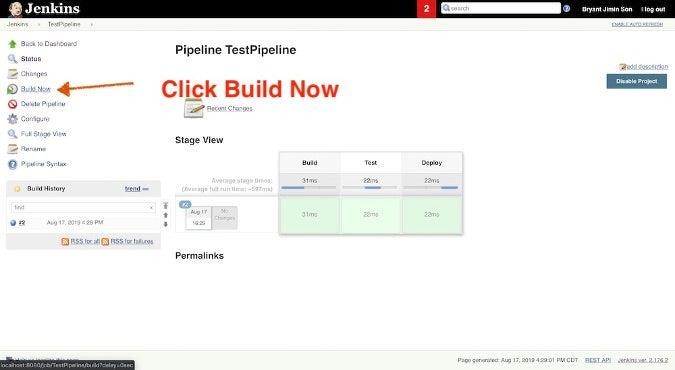
To execute the pipeline with a direct script, begin by copying the contents of the [sample](https://github.com/bryantson/CICDPractice) [Jenkinsfile](https://github.com/bryantson/CICDPractice) from GitHub. Choose **Pipeline script** as the **Destination** and paste

the **Jenkinsfile** contents in **Script**. Spend a little time studying how the Jenkins file is structured. Notice that there are three Stages: Build, Test, and Deploy, which are arbitrary and can be anything. Inside each Stage, there are Steps; in this example, they just print some random messages.

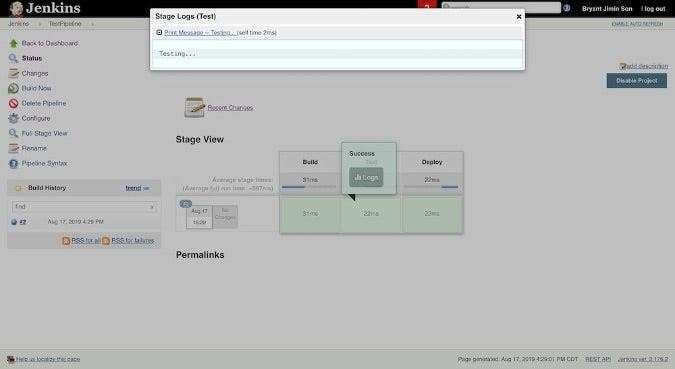
Click **Save** to keep the changes, and it should automatically take you back to the Job Overview.



To start the process to build the pipeline, click **Build Now**. If everything works, you will see your first pipeline (like the one below).



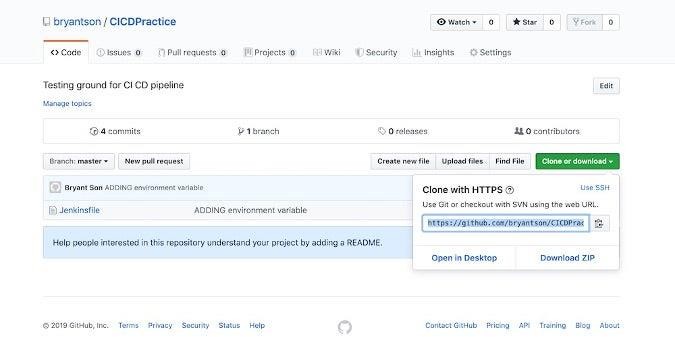
To see the output from the pipeline script build, click any of the Stages and click **Log**. You will see a message like this.



Step 6: Configure and execute a pipeline job with SCM

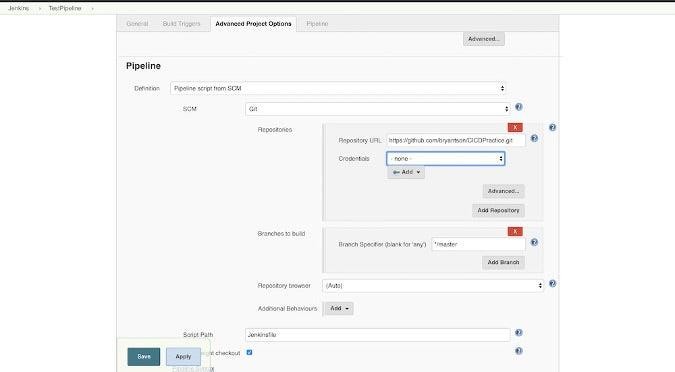
Now, switch gears: In this step, you will Deploy the same Jenkins job by copying

the **Jenkinsfile** from a source-controlled GitHub. In the same [GitHub repository](https://github.com/bryantson/CICDPractice), pick up the repository URL by clicking **Clone or download** and copying its URL.

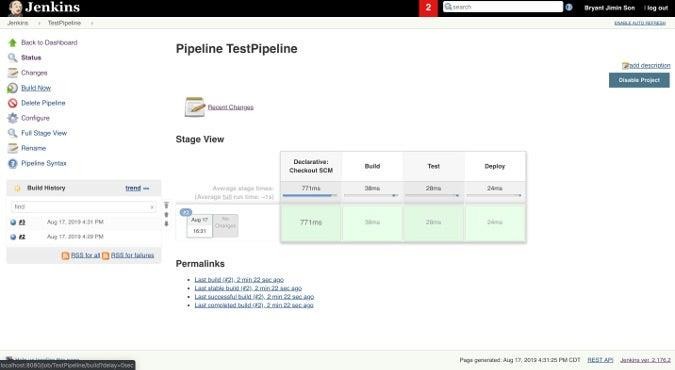


Click **Configure** to modify the existing job. Scroll to the **Advanced Project Options** setting, but this time, select the **Pipeline script from SCM** option in the **Destination** dropdown.

Paste the GitHub repo's URL in the **Repository URL**, and type **Jenkinsfile** in the **Script Path**. Save by clicking the **Save** button.



To build the pipeline, once you are back to the Task Overview page, click **Build Now** to execute the job again. The result will be the same as before, except you have one additional stage called **Declaration: Checkout SCM**.



To see the pipeline's output from the SCM build, click the Stage and view the **Log** to check how the source control cloning process went.