**Hi, this is Raju, I prepared pre document for the CI CD pipeline using Jenkins and Docker.i will submit you my complete document next time.**

**Deliverables**: This demonstration will simulate a completely automated CI/CD deployment pipeline using Jenkins. It will essentially do the following steps (phases):

1. Pull the source code for a Java EE based Project from GIT. (SCM AUTOMATION)
2. Compile (build) the code using Maven to generate the .war file (BUILD AUTOMATION)
3. Run Test cases & ensure they pass. (TEST AUTOMATION)
4. Copy the .war to a Docker build workspace (DEPLOYMENT AUTOMATION)
5. Build a Docker image for Jboss server to run the war file. (DEPLOYMENT AUTOMATION)
6. Deploy the Docker container on a target node. (DEPLOYMENT AUTOMATION)

**Prerequisites**: This demonstration has the following prerequisites:

1. Jenkins should be installed with git, maven and shell plugins.
2. In Jenkins Server install using # yum -y install git maven docker before trying out this demo.
3. Changes to be made for Jenkins to be able to run docker.

echo "jenkins ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

echo 'Defaults:jenkins !requiretty' >> /etc/sudoers

setenforce 0 # Else disable SELINUX in /etc/sysconfig/selinux and reboot

**Execution**: Add a Jenkins Build Job As per the below screenshot and build it:

* Note: I have added some shell scripting commands in the build commands from the **jenkins\_build\_commands.md** file. You can check them once.

**My work process:**

**Step 1:** Open your terminal in your VM. Start Jenkins and Docker using these commands:

 systemctl start jenkins

 systemctl enable jenkins

 systemctl start docker

**step 1**:i Opened Jenkins on your specified port. Click on **New Item**to create a Job.

**Step 3:** Select a **freestyle** project and provide the item name (here I have given Job1) and click OK.

**Step 4:** Select **Source Code Management** and provide the **Git** repository. Click on **Apply** and **Save**button.

**Step 5:** Then click on **Build->Select Execute Shell**.

**Step 6:** Provide the shell commands. Here, it will build the archive file to get a war file. After that, it will get the code which is already pulled and then it uses maven to install the package. It simply installs the dependencies and compiles the application.

**Step 7:** Create the new **Job** by clicking on New Item.

**Step 8:** Select **freestyle** project and provide the item name (here I have given Job2) and click on OK.

**Step 9:** Select **Source Code Management** and provide the **Git** repository. Click on **Apply** and **Save**button.

Then click on **Build->Select Execute Shell**.

**Step 11:** Provide the shell commands. Here it will start the integration phase and **build** the Docker Container.

**Step 12:** Create the new **Job** by clicking on New Item.

**Step 13:** Select **freestyle** project and provide the item name (here I have given Job3) and click on OK.

**Step 14:** Select **Source Code Management** and provide the **Git** repository. Click on **Apply** and **Save**button.

Then click on **Build->Select Execute Shell**.

**Step 16:** Provide the shell commands. Here it will check for the Docker Container file and then deploy it on port number 8180. Click on Save button.

**Step 17:** Now click on **Job1 -> Configure**.

**Step 18:** Click on **Post-build Actions -> Build other projects**.

**Step 19:** Provide the project name to build after Job1 (here is Job2) and then click on **Save**.

**Step 20:**Now click on **Job2 -> Configure**.

**Step 21:** Click on **Post-build Actions -> Build other projects**.

**Step 22:** Provide the project name to build after Job2 (here is Job3) and then click on **Save**.

**Step 23:** Now we will be creating a Pipeline view. Click on the "+" sign.

**Step 24:** Select **Build Pipeline View** and provide the view name (here I have provided CI CD Pipeline).

**Step 25:** Select the **initialJob** (here I have provided Job1) and click on OK.

**Step 26:** Click on **Run**button to start the CI/CD process.

**Step 27:** After successful build open **localhost:8180/sample.text**. It will run the application.

These are steps I have followed to finish my CI CD pipeline, using Jenkins and Docker.