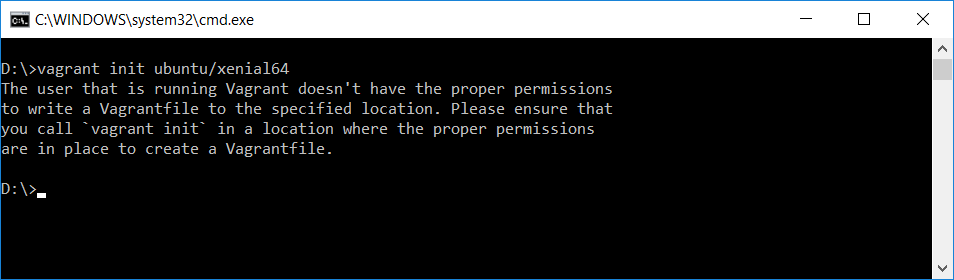
**Jenkins CI**

# Vagrant + Virtualbox

**Step1.** As prerequisite download and install Vagrant and Virtualbox binaries:

<https://www.vagrantup.com/downloads.html>

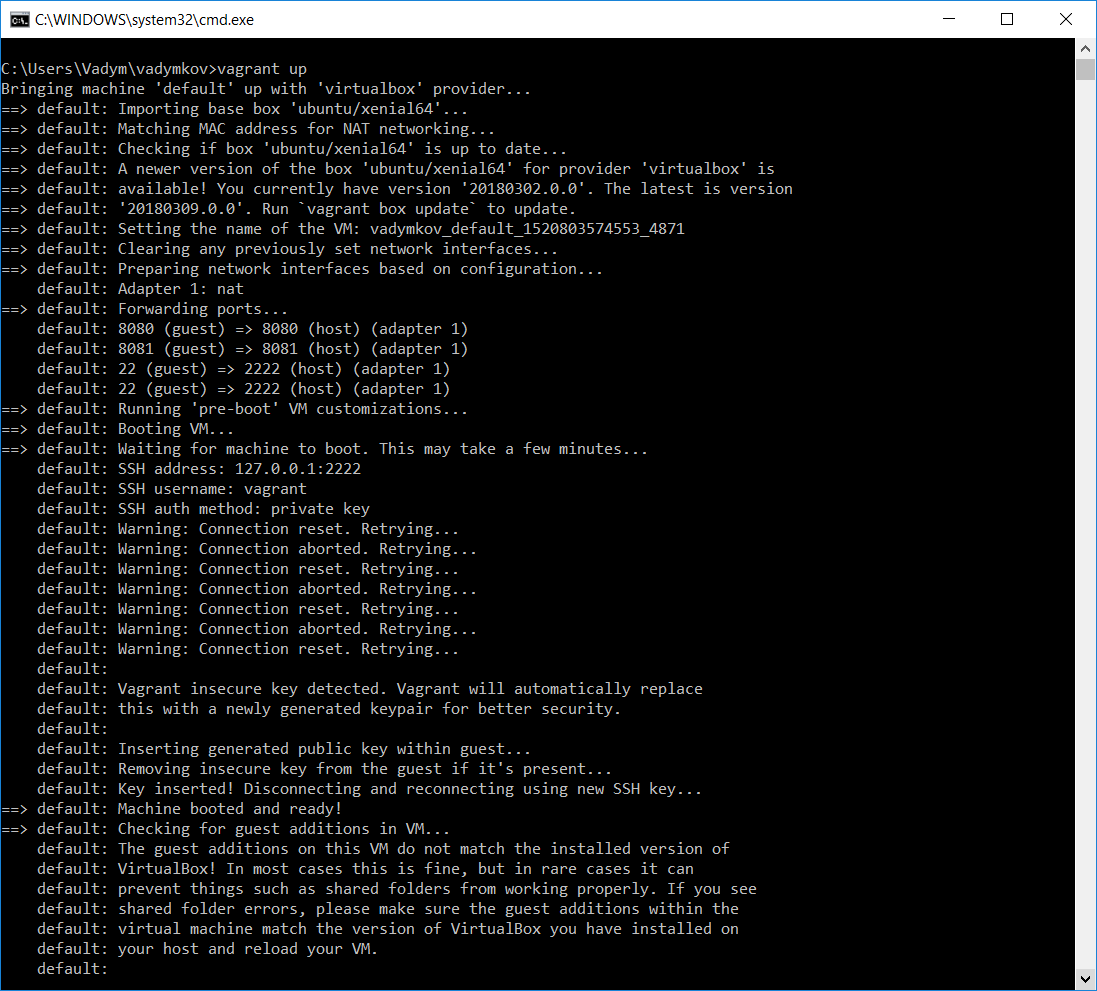
<https://www.virtualbox.org/wiki/Downloads>

**Step 2.** Initialize and start ubuntu/xenial64 environment. ($vagrant init ubuntu/xenial64)

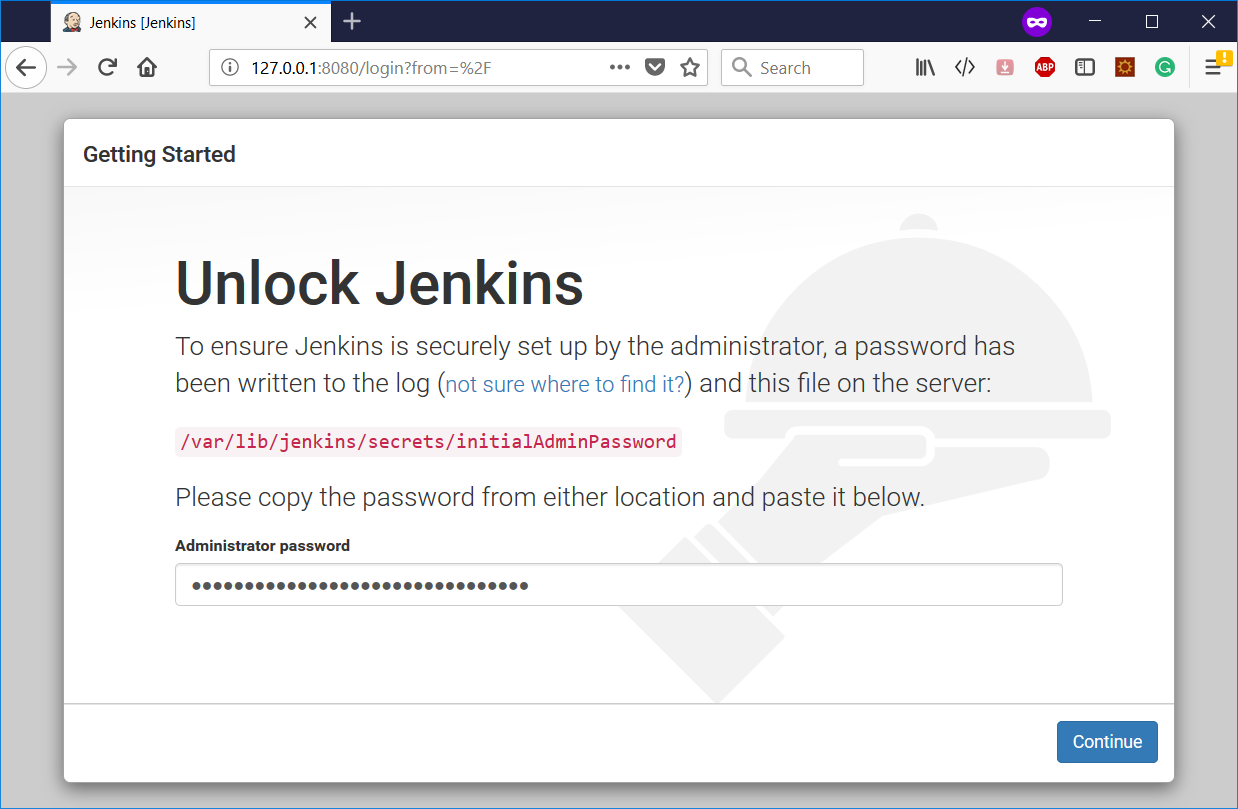
**Step 3.** Replace Vagrant file with a file from bitbucket repo:

[**https://bitbucket.org/toorroot/devops\_base/src/daf84bcfcfe9784848830a1c1fcc6849d9b1a5a3/Labs/Vagrantfile?at=master&fileviewer=file-view-default**](https://bitbucket.org/toorroot/devops_base/src/daf84bcfcfe9784848830a1c1fcc6849d9b1a5a3/Labs/Vagrantfile?at=master&fileviewer=file-view-default)

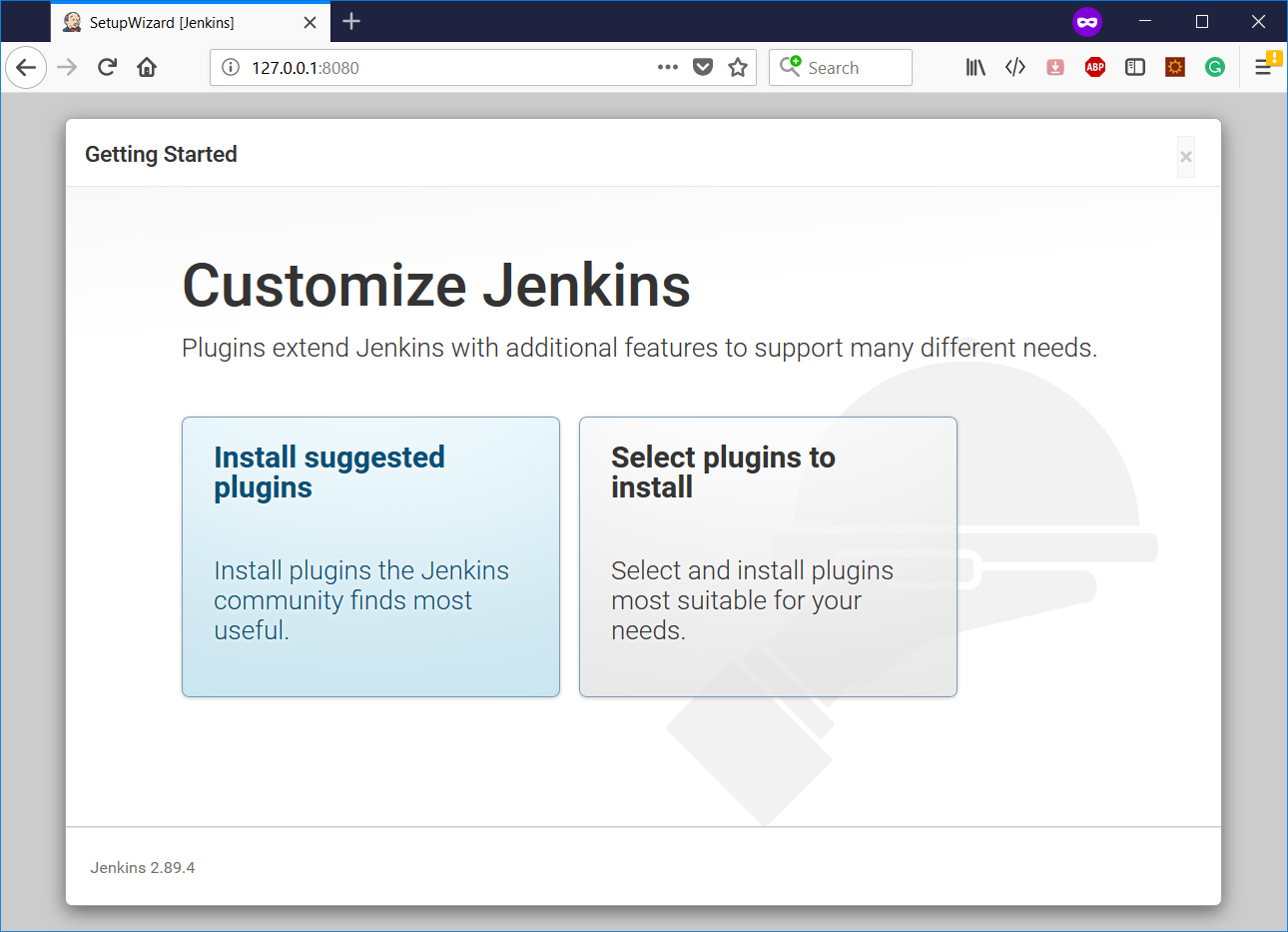
**Step 4.** Start predefined Virtualbox instance.



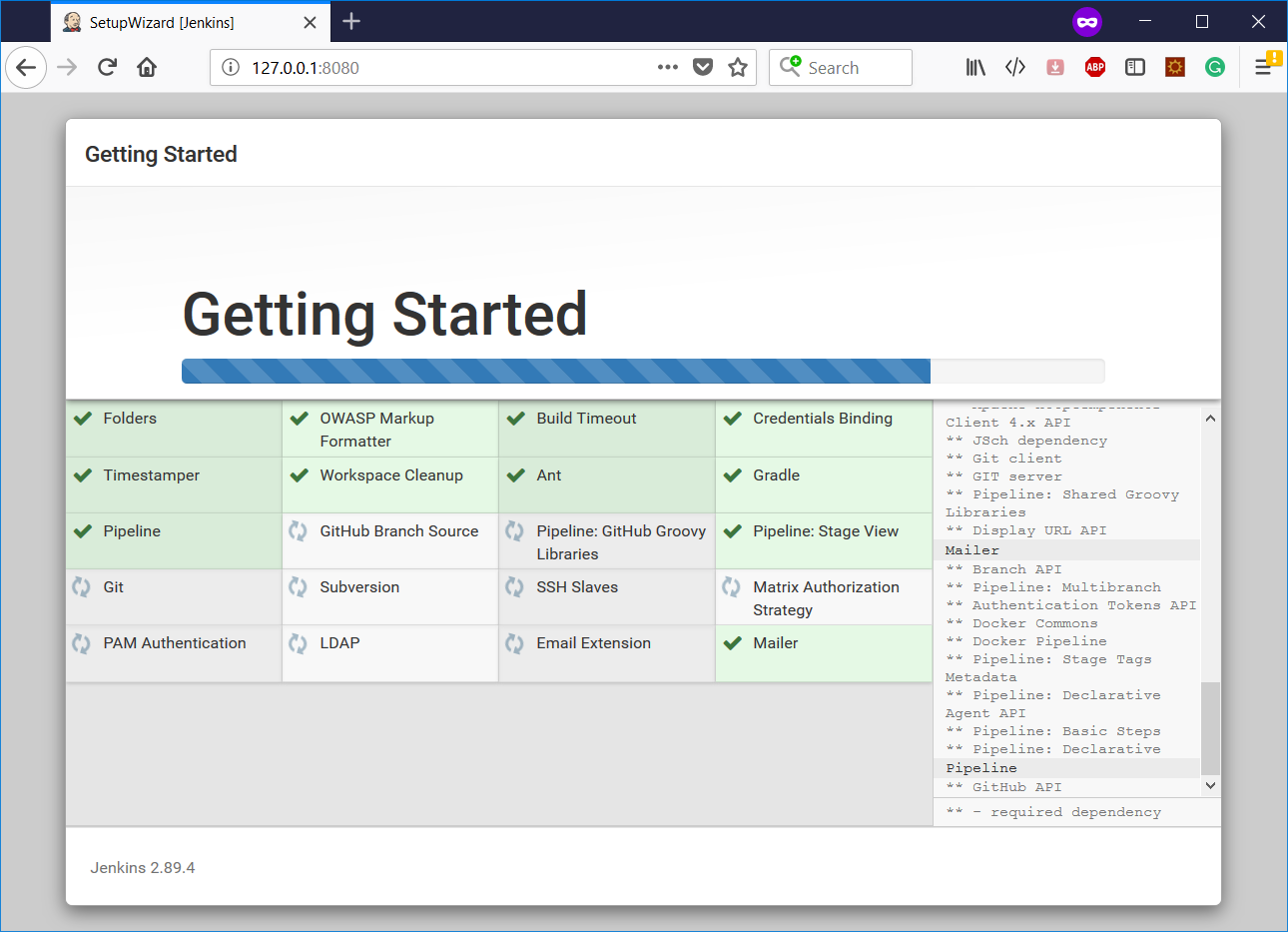
**Step 5.** Navigate to [*http://127.0.0.1:8080*](http://127.0.0.1:8080)and provide password obtained from previous step:



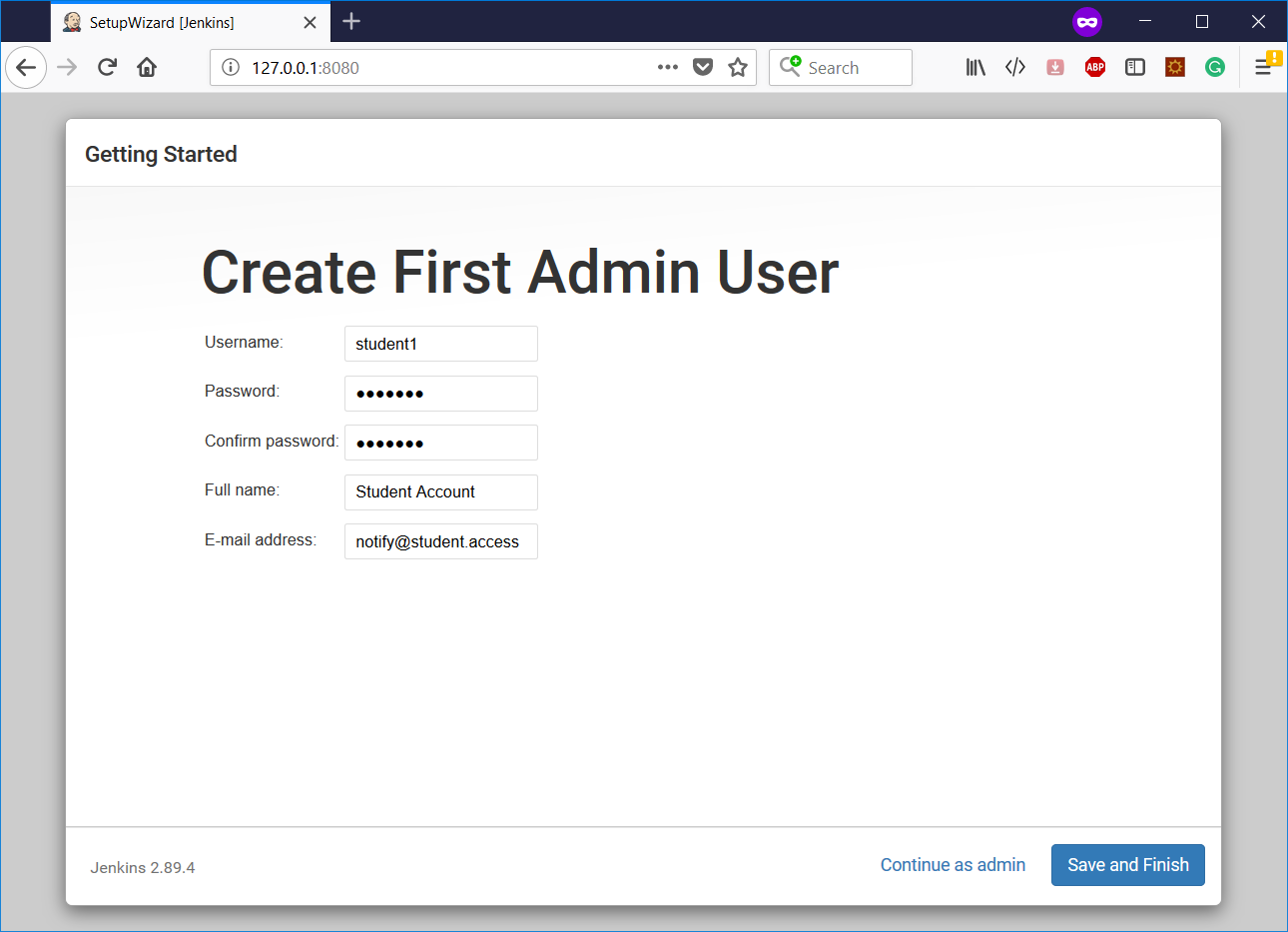
**Step 6.** Install suggested plugins:



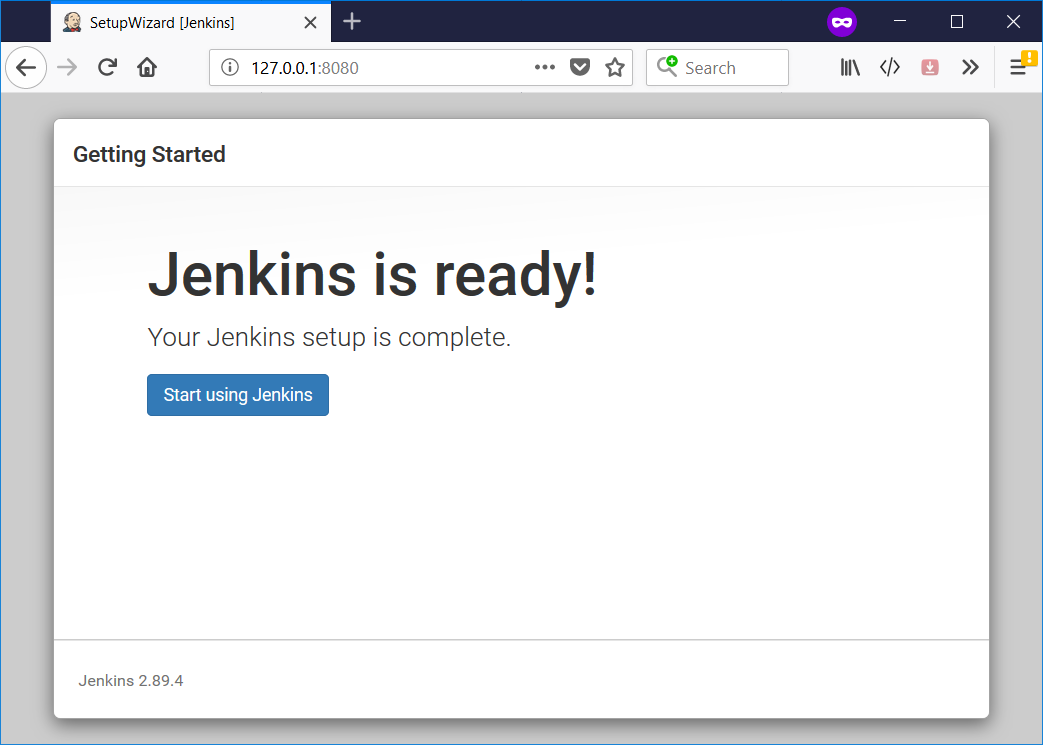
**Step 7.** Plugins installations progress



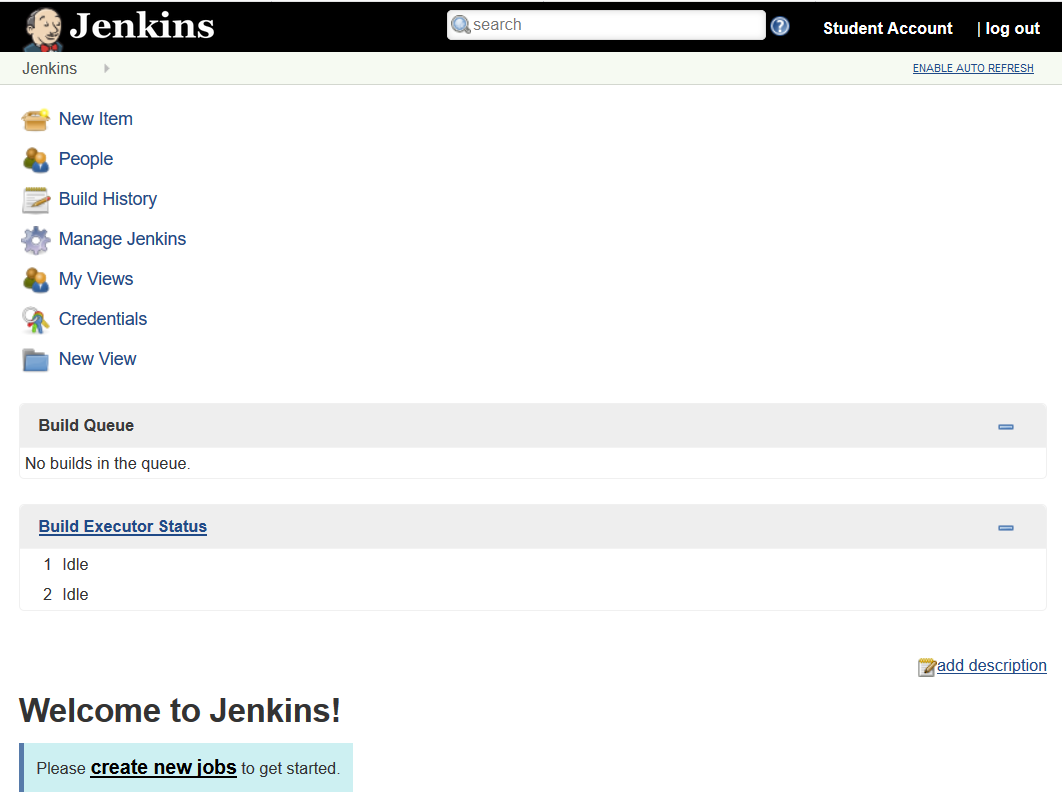
**Step 8.** Finish installation by provisioning Jenkins admin user:



**Step 9.** Jenkins is installed and ready for further configuration.

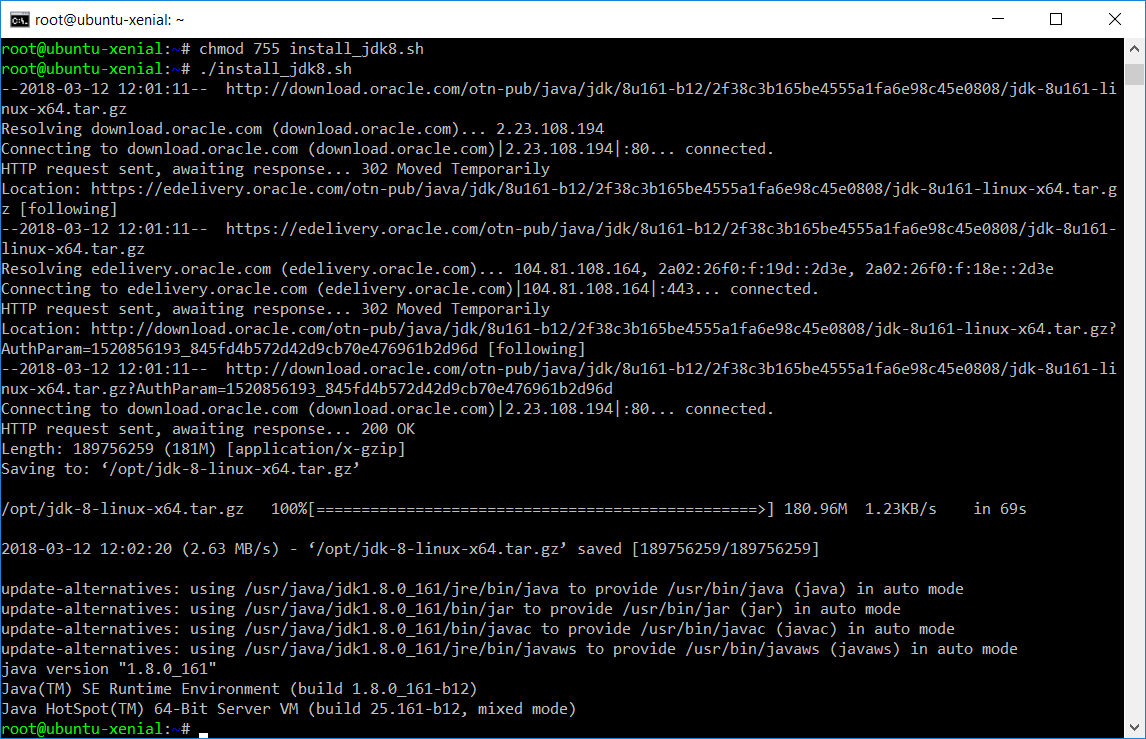


**Step 10.** Login using admin account:



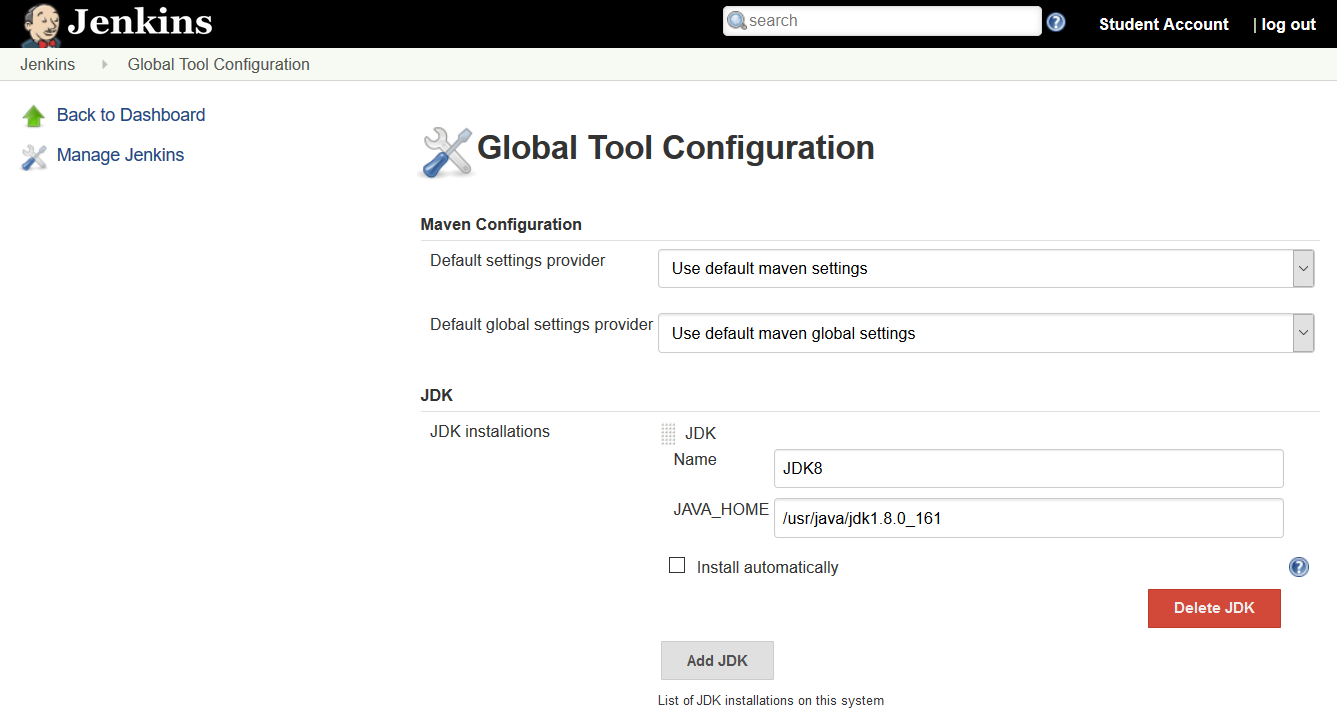
# Configuring plugins and tools

**Step 1.** Install JDK using [install\_jdk8.sh](https://bitbucket.org/toorroot/devops_base/src/daf84bcfcfe9784848830a1c1fcc6849d9b1a5a3/misc/install_jdk8.sh?at=master)

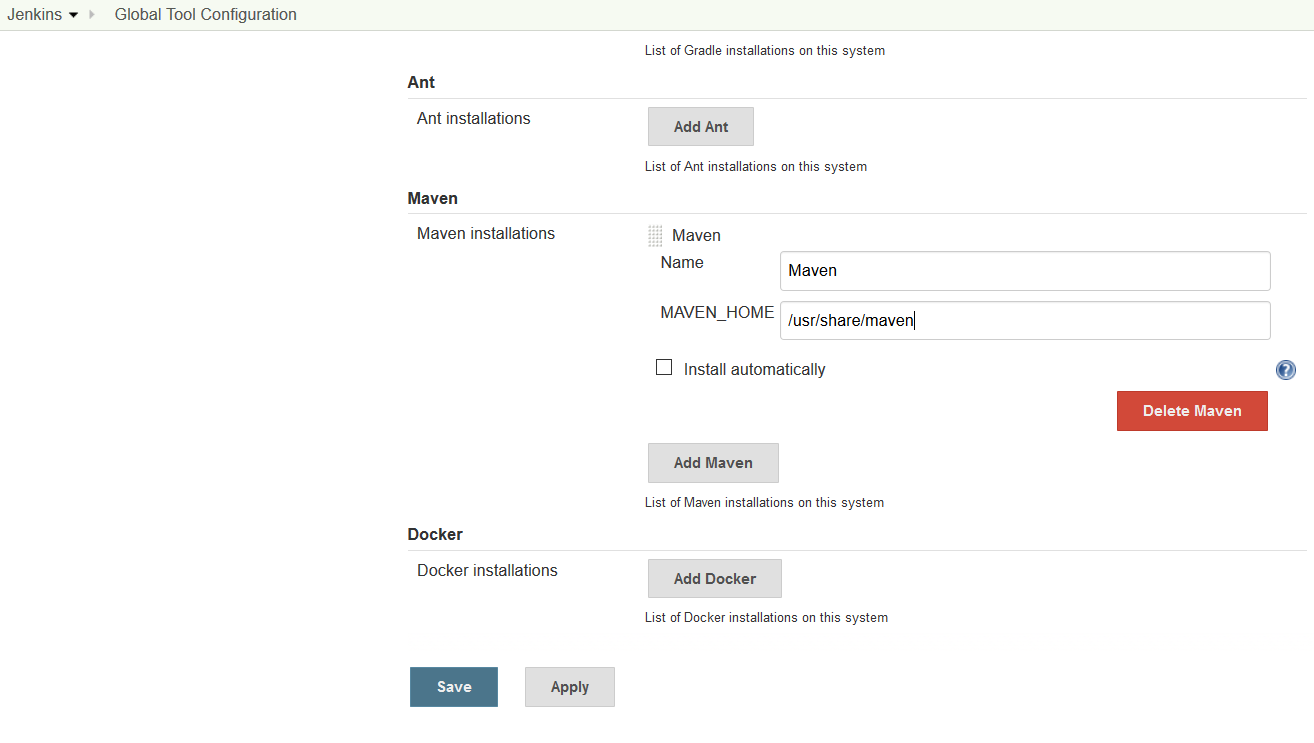


Step 2. We do need to do a little configuration. More precisely, we need to tell Jenkins

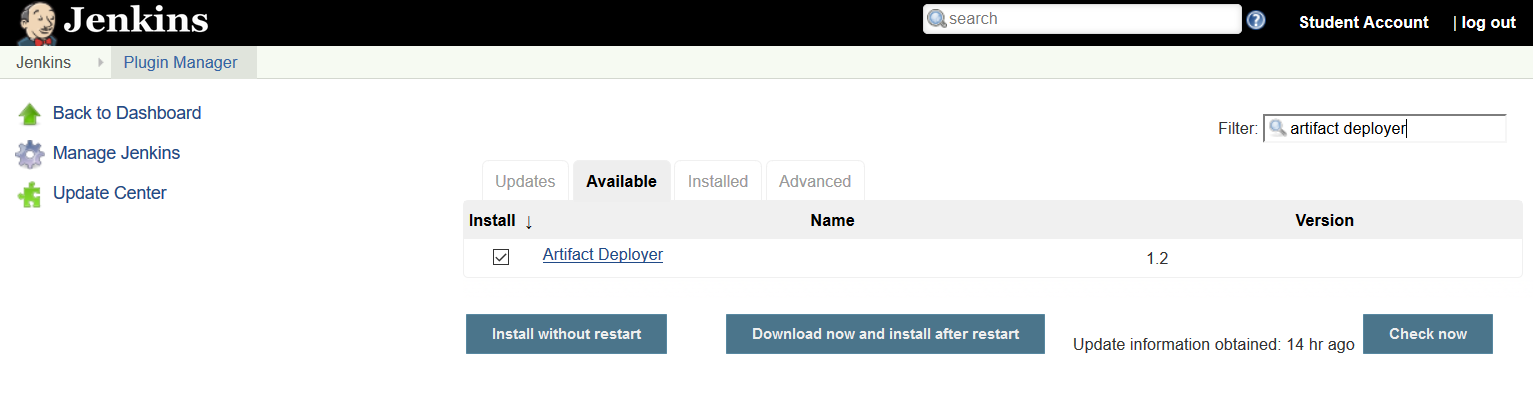
about the build tools and JDK versions we will be using for our builds.

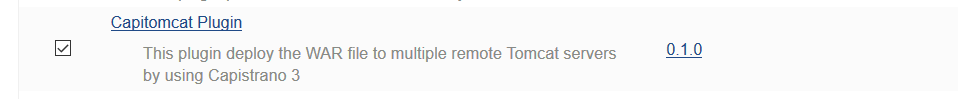


Configure path for Maven applying changes and saving config:



Step 3. Configure plugins installing ‘capitomcat’ and ‘Deployer Artifact’

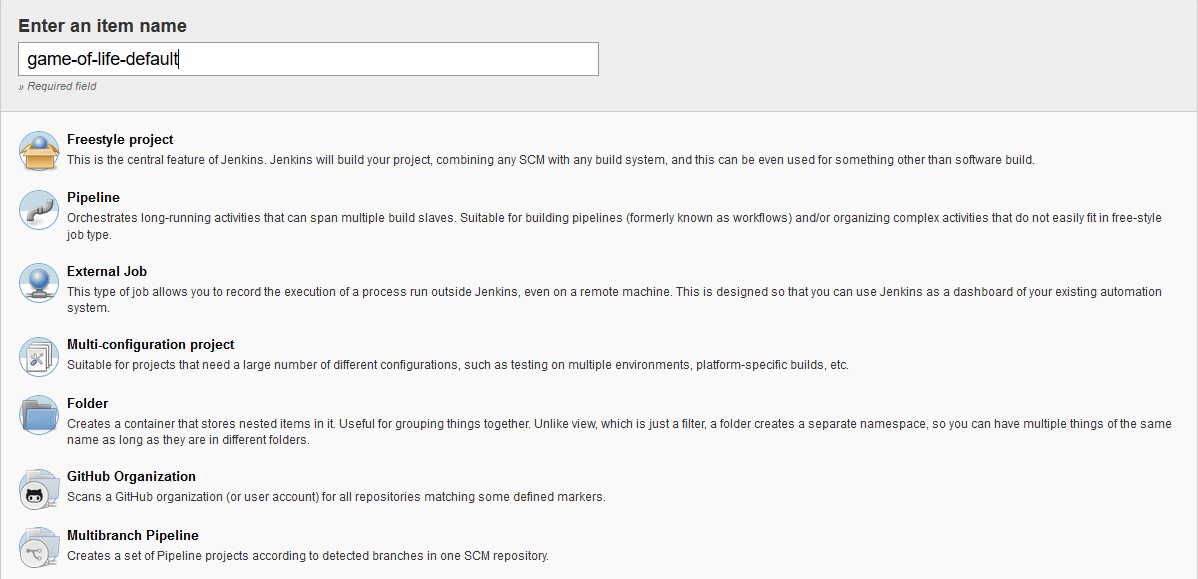




## Your First Jenkins Build Job

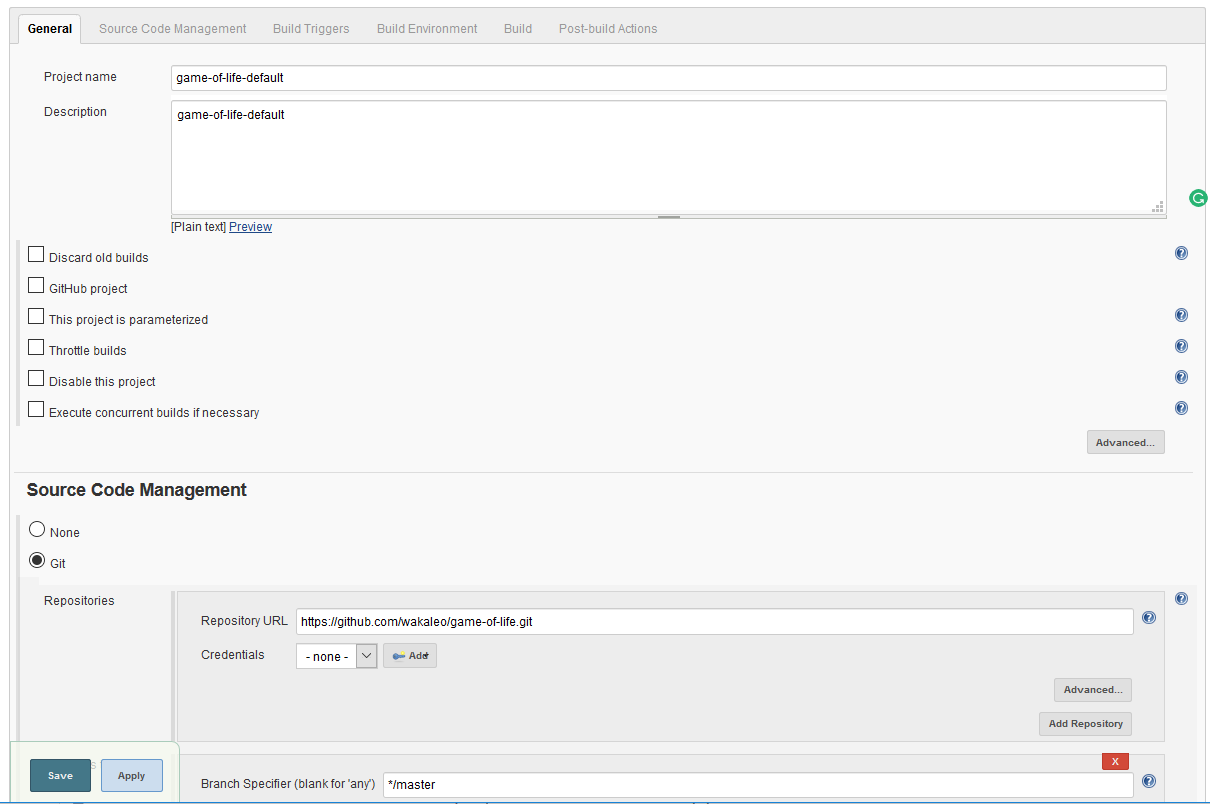
Build jobs are at the heart of the Jenkins build process. Simply put, you can think of a Jenkins build job as a particular task or step in your build process. This may involve simply compiling your source code 22 and running your unit tests. Or you might want a build job to do other related tasks, such as running your integration tests, measuring code coverage or code quality metrics, generating technical documentation, or even deploying your application to a web server.

You’ll also need to give your build job a sensible name. In this case, call it game-of-life-default, as it will be the default CI build for our Game of Life project.



**Step 4.** Once you click on OK, Jenkins will display the project configuration screen.

Define git repo as: **https://github.com/wakaleo/game-of-life.git**



Pick the Poll SCM option and enter “H/15 \* \* \* \*” (that’s five asterisks separated by spaces) in the Schedule box. Jenkins schedules are configured using the cron syntax, well-known in the Unix world. The cron syntax consists of five fields separated by white space, indicating respectively the minute (0–59), hour (0–23), day of the month (1–31), month (1–12) and the day of the week (0–7, with 0 and 7 being Sunday).

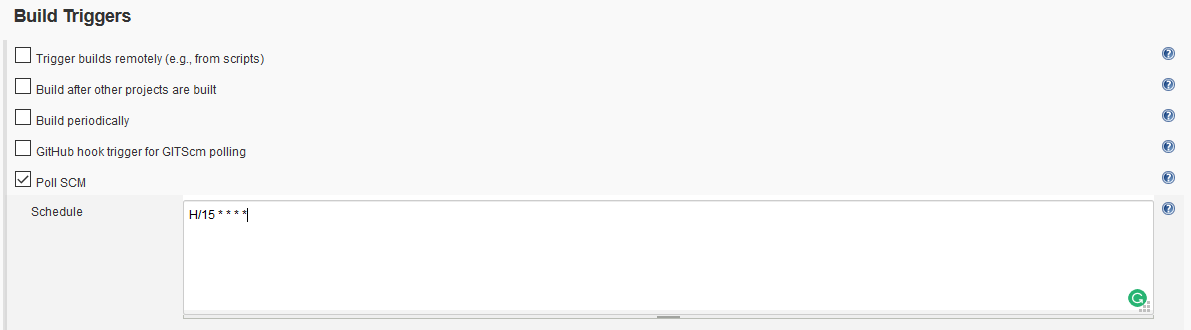
The star is a wildcard character which accepts any valid value for that field. So five stars basically means

“every minute of every hour of every day.” You can also provide ranges of values: “\* 9-17 \* \* \*” would

mean “every minute of every day, between 9am and 5pm.” You can also space out the schedule using

intervals: “\*/5 \* \* \* \*” means “every 5 minutes,” for example. Finally, there are some other convenient

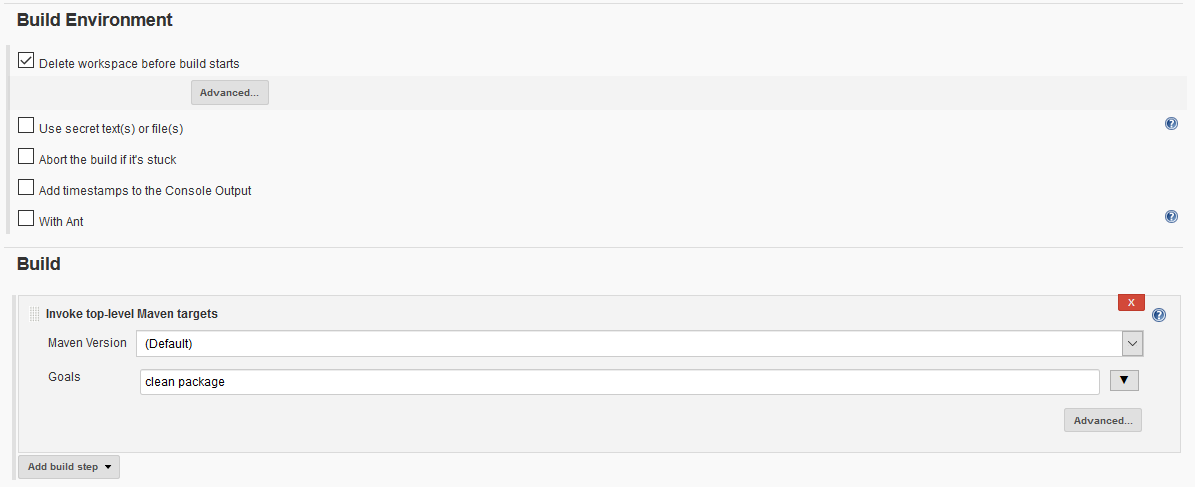
short-hands, such as “@daily” and “@hourly”.



For now, we just want to run a simple Maven build. Scroll down to the Build section and click on the “Add build step” and choose “Invoke top-level Maven targets”.

Then enter “clean package” in the Goals field. If you are not familiar with Maven, this will delete any

previous build artifacts, compile our code, run our unit tests, and generate a JAR file.

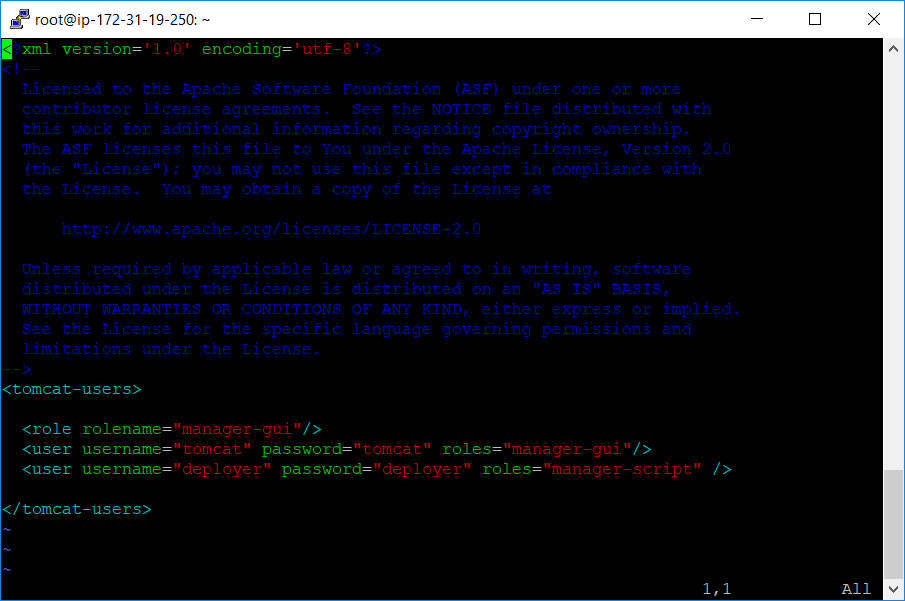


## Remote deployment to Tomcat 8 container:

### Step 1. Install Jenkins plugin

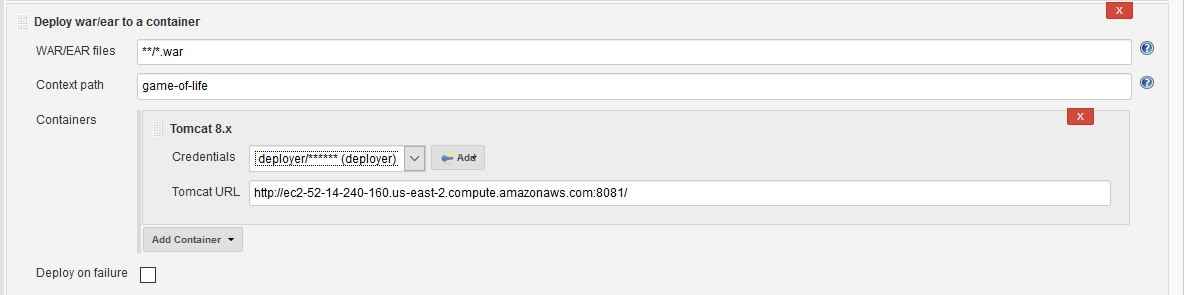
Open your favorite browser and navigate to Jenkins. Log in and select “Manage Jenkins” followed by “Manage Plugins”. Select the “Available” tab, locate the “[Deploy to container](https://wiki.jenkins-ci.org/display/JENKINS/Deploy+Plugin)” plugin and install it.

### Step 2. Adapt your */opt/tomcat/conf/tomcat-users.xml* as the following:



Step 3. Configure Jenkins job

Back in Jenkins, go to your job and select “Configure”. Next, scroll down to the bottom of the page to the “Post-build Actions”. Select the option “Deploy war/ear to a container” from the “Add post-build action” dropdown button. Fill in the new fields, e.g.:



Obtain deployment status from console:

