Jenkins Build 2

Objectives:

- Configure Unit tests with Jenkins
- Configure a second build node
  - This requires a second vm (we are going to use linux)
- Configure email-based notifications
- Integrate Jenkins and SonarQube

Requirements:

Continue where you left off on your last project with Jenkins. You will start in chapter 2 at the ‘Configuring JUnit’ section, and work to the end of the chapter.

Plugins

When configuring the dashboard view plugin, after configuring it, you must attach it to a build job. Attach it to your PetClinic build job you created in the last assignment. I don’t care what portlets you add.

Nodes

Configure a second VM as a build node. Though the book uses a windows machine, we will use a Linux vm as our second node. Make sure to install `default-jre`, `default-jdk`, `git`. (You could use your production vm, if you still have that one.)

On that machine:

- Add a new user called `jenkins`
- As that new user, in their home directory, create a directory called `jenkins_slave`.
- From the jenkins web interface, as you are adding your new node, use the following configuration:
  - remote root directory `/home/jenkins/jenkins_slave`
  - Launch slave agent via SSH
  - Configure the host ip of second machine
  - Add new credentials (will ask you for name and password of jenkins user)
  - Use non-verifying host strategy
- Verify the log status of the agent that it is online
- I ran my build project on this node at this point. You only need to edit your project settings and select where the project can run. (Look for the checkbox restrict where this can run, then enter the name of your node)
- Make sure that you can build your project on BOTH the `master` node and your new `slave` node.

SonarQube

- I installed on same node as jenkins (though you probably want to bump the RAM to 4G if possible)
- **DO NOT USE SUDO WHEN YOU UNZIP OR RUN THIS OR YOU WILL BREAK THINGS!**
- Default username/password = admin/admin
- The real command to run it is `./sonar.sh start`
  - If it doesn’t seem to run, look for the logs directory inside the sonarqube directory that you extracted. See if you can figure what the error is.
  - This has to be re-run if you reboot the node
- After you start it, you have to visit port 9000 on that node (as indicated in the instructions).
- One thing that the text doesn’t make clear is that you MUST add a build step to your project to have the SonarQube scanner run.
- I also found that my build failed with SonarQube, unless I added the following line to sonar-project.properties (I already added it to the repo, you don’t have to do anything).

```
sonar.java.binaries=src
```
Check off procedure:

For this project, you should submit the following screenshots:

- JUnit tests are showing up
- That you have added some dashboard plugins
- That you can successfully execute a build job on a second node
- That you can receive an email notification when the build fails
- That sonarqube has received some information from your build job.