**Final labs**

**Objectives**

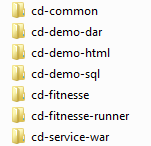
In this lab you are going to implement a Continuous Integration pipeline for a new project “sample-app”.

**LAB**

1. Clone sample-app from GitHub:

git clone https://github.com/HerbertSchuurmans/sample-app.git

1. This will create the following directory structure:



cd-common: a project that cd-service-war depends on

cd-demo-dar: will generate a deployable package

cd-demo-html: static content for Apache webserver

cd-demo-sql: sql-scripts for MySQL

cd-fitnesse: test suites

cd-fitnesse-runner

cd-service-war: the webapplication to be deployed in JBoss

1. In the pom files are some exercises to make the project work
   1. cd-common: add dependency for junit
   2. cd-demo-html: add maven-assembly-plugin
   3. cd-fitnesse: add maven-dependency-plugin
   4. cd-fitnesse-runner: add maven-failsafe-plugin
   5. cd-service-war: add maven-war-plugin

Next to these exercises there are some configurations missing as well

Test the project using “mvn clean install”

**Tip**: do this on your local image first by cloning the project and running a “mvn clean install” until the project has build successfully.

1. Create a Jenkins pipeline in order to build, test and deploy the sample-app.
   1. Use 3 jobs:
   * Create: build the sample-app
   * integration-test: executes the integration tests
   * deploy: deploys the application to the web-, application and databaseserver

**Tip**: Jenkins is unable to clone a repository from GitHub. Use a shell script to do this:

“git clone <https://github.com/HerbertSchuurmans/sample-app.git>”

After that change directory to “sample-app”

* 1. Let all jobs execute a “git clone”

1. Add Fitnesse tests for:
   1. <a class="btn btn-large btn-success" href="[http://10.20.20.20:8080/cd-demo](view-source:http://10.20.20.20:8080/cd-demo)">Open the continuous delivery demo application</a> on <http://10.20.20.20>
   2. <html><body><h1>hello Xebia</h1></body></html> on http://10.20.20.20:8080/cd-demo/service
   3. <table>

<tr><td>name</td><td>cd-service-war</td></tr>

<tr><td>version</td><td>1.0.0.0-SNAPSHOT</td></tr>

<tr><td>build</td><td>2013-05-17 14:18:18</td></tr>

**Tip**: Run Fitnesse locally in order to have quick feedback. To do this, do “mvn jetty:run” in cd-service-war and “mvn exec:java” in cd-fitnesse

1. Use Sonar for static analysis of the application
   1. Add an alert in order to let the build break
2. Add an extra view “sample-app build-pipeline” in Jenkins in order to check the pipeline
3. Break one of the integration tests by check for a status “123” and rebuild the project

Details

* SSH user/pass: vagrant/vagrant
* Git repo: https://github.com/HerbertSchuurmans/sample-app.git
* Sonar: <http://10.20.20.20:9000/>
* Jenkins: <http://10.20.20.20:9090/>
* Deployit: <http://10.20.20.20:4516/>
* Deployit home: /opt/deployit-server/
* Jboss AS console: <http://10.20.20.20:8080/admin-console/> (admin/admin)
* Mysql: root/root
* Maven home: /usr/share/maven (Global env variable MAVEN\_HOME is available)
* Java (jdk) home: /usr/lib/jvm/java-6-openjdk-i386 (Global env variable JAVA\_HOME is available)