Friday Practice:

# Subject to be covered:

* Git Operations
* Build Frameworks & Build Targets
* Jenkins
  + Management & configurations
  + Jenkins cluster simulation
  + Jenkins Pipeline:
    - Plugins
    - Triggers
* SDLC – implementing using SonarQube

# Introduction:

In this practice we will provide a CI environment and build a simple maven project and we will deploy its output into a running tomcat and into new Docker Container.

We will mix all the knowledge we collected so far.

Please follow carefully after each step of this manual, since the exercise is a bit complex.

## User story:

You are a DevOps engineer, that called by a client to provide him full CI environment for POC purpose.

The client gave you the following demands:

1. Since it is a POC environment, he doesn’t want you to create or install any new server. The whole SLN should be running from your work station completely
2. Since he gave you only single workstation to present the POC, and you need to simulate multiple server environment, using docker based solution is advised.
3. The environment should build from the following components:
   1. One Jenkins master
   2. 2 Jenkins Slaves
   3. 1 SonarQube server
   4. 1 Nexus server
4. Since the containers might failing from time to time, you need to provide some data persistency. Using folder-based mounts it is advised.
5. Since this is a POC, no access to source code of the organization is allowed
6. You must provide 1 pipelines, that includes SCM pull, build, test and publish into Nexus. Publishing to the nexus repository will be based on the branch type, meaning, develop will publish into a “develop” repository, and master will be published into a “master” repository. Also, in case of security scan fails, the artifact will be published into a “suspected” repository. The client demands that the developers will able to work against single repository rather that those 3 mentioned.
7. The client wants to be able see the unit tests results in the Jenkins job’s page. Using the **Junit** plugin \build step to achieve that
8. The client wants to be able see the security scans results in the Jenkins job’s page.
9. When the artifact passes the security check, the artifact marked as “passed\_sec” in it’s repository

# Important notes:

* You will need to create a new Git repo for the project. The git repo should have 2 branches, master and develop

|  |
| --- |
| TASK 1: Create the Git Repo, with the proper settings defined |

* The source code will be generated from following site: <https://start.spring.io/>
  + Make sure you are selecting the following configs:

Project= Maven Project

Language=Java

SpringBoot=2.1.6

Group=%your name group%

Artifact= %Choose your own%

Packaging=War

Java=8

|  |
| --- |
| Task 2 (Bonus): Create automatic script to get the code  Tip: open browser’s developer tools |

# Steps to solve this exercise:

1. Read and understand