CI/CD/Automation AWS EC2 INSTANCE EC2 Instance (Amazon Linux t2 micro)/Docker Deployment Project

1. The code has been “forked”

From: <https://github.com/LableOrg/java-maven-junit-helloworld>

To: <https://github.com/eranfuld/java-maven-junit-helloworld>

It is a maven based “hello world project. see below:

**# A Java/Maven/JUnit HelloWorld example**

A „Hello World!” sample written in Java using Maven for the build, that showcases a few very simple tests.

This example demonstrates:

\* A simple Java 8 application with tests

\* Unit tests written with [JUnit 5](https://junit.org/junit5/)

\* Integration tests written with [JUnit 5](https://junit.org/junit5/)

\* Code coverage reports via [JaCoCo](https://www.jacoco.org/jacoco/)

\* A Maven build that puts it all together

**## Running the tests**

\* To run the unit tests, call `mvn test`

\* To run the integration tests as well, call `mvn verify`

\* Code coverage reports are generated when `mvn verify` (or a full `mvn clean install`) is called.

Point a browser at the output in `target/site/jacoco-both/index.html` to see the report.

**## Conventions**

This example follows the following basic conventions:

| | unit test | integration test |

| --- | --- | --- |

| **\*\*resides in:\*\*** | `src/test/java/\*Test.java` | `src/test/java/\*IT.java` |

| **\*\*executes in Maven phase:\*\*** | test | verify |

| **\*\*handled by Maven plugin:\*\*** | [surefire](http://maven.apache.org/surefire/maven-surefire-plugin/) | [failsafe](http://maven.apache.org/surefire/maven-failsafe-plugin/) |

The Jenkins job is uses a Poll SCM and triggered by the git repository users commits. It runs every minute, checking for new commits. Each commit triggers a new build. It is currently using a HTTPS connection and not SSH. There is a total of 4 Jenkins job steps:

1. The build consists out of 5 stages: clean, test, verify, install, package (install also implement verify).
2. The next Jenkins job step is an execute shell Ansible Playbook that verifies the host’s availability. it uses Ansible’s “Ping Pong” function.
3. The next Jenkins job step is an execute shell Ansible Playbook that prepares the environment (AWS EC2 INSTANCE) for Jenkin’s artifacts (web-based report). The following steps are being done on the target host (AWS EC2 INSTANCE host, at this point the Docker container is still being developed): HTTPD is being installed -> java 8 is being installed -> HTTPD is being started -> the artifacts (web-based report) are being copied to the AWS EC2 INSTANCE host.
4. The next Jenkins job step is an execute shell Ansible Playbook that prepares the docker container (TBD). Currently the Ansible Playbook does the following in order to install and start Docker: install yum-utils -> install device-mapper-persistent-data -> install lvm2 -> configure the right yum repo -> install Docker -> start Docker.
5. A new docker-deploy playbook has been added as an additional Jenkins job step-it creates a docker build using a Dockerfile, followed by the artifact copy into that same directory. A build and run are being launch, redirecting an exposed port 81 to an internal standard http port 80.

P.S

The AWS EC2 Instance deployment has been tested and verified.

The Docker Container deployment has been tested and verified.