After checking the POM.XML file and validating that the Maven dependency check was present, the code was then run as a Maven build. After the project was run there was an html within the target folder which allowed me to open the dependency report.

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated

The current dependency report shows how many dependencies there are and how many vulnerabilities are found. There is also a list of dependencies. If there comes a time that there are any false positives that need to be taken care of, so they are no longer visible, then suppression is a viable option.

A computer screen shot of a program

Description automatically generated

The POM.XML file here is reconfigured so that it will read in the suppression.xml file. The following images show the suppression.xml file with the different suppression codes that the pom.xml file will take in and ignore those vulnerabilities.

A computer screen shot of a program

Description automatically generated

A computer screen shot of a program

Description automatically generated

After the program has all the information needed to know what to suppress the dependency report is run again. This time it will give the current information. It now shows how many vulnerabilities are suppressed as well as showing that there is now one less dependency that is vulnerable.

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated

**Citations**

*How to read the reports*. dependency-check – How To Read The Reports. (n.d.). https://jeremylong.github.io/DependencyCheck/general/thereport.html

*Suppressing false positives*. dependency-check – Suppressing False Positives. (n.d.). https://jeremylong.github.io/DependencyCheck/general/suppression.html