# CS 305 Module Two Coding Assignment Template

## Instructions

Replace the bracketed text with the relevant information in your own words. If you choose to include images or supporting materials, make certain to insert them in all the relevant locations in the document.

## Run Dependency Check

Graphical user interface

Description automatically generated with low confidence

## Document Results

### hibernate-validator-6.0.18.Final.jar

**Description:**

Hibernate's Bean Validation (JSR-380) reference implementation.

**License:**

<http://www.apache.org/licenses/LICENSE-2.0.txt>

### jackson-databind-2.10.2.jar

**Description:**

General data-binding functionality for Jackson: works on core streaming API

**License:**

<http://www.apache.org/licenses/LICENSE-2.0.txt>

### log4j-api-2.12.1.jar

**Description:**

The Apache Log4j API

**License:**

https://www.apache.org/licenses/LICENSE-2.0.txt

### logback-core-1.2.3.jar

**Description:**

logback-core module

**License:**

[http://www.eclipse.org/legal/epl-v10.html, http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html](http://www.eclipse.org/legal/epl-v10.html,%20http:/www.gnu.org/licenses/old-licenses/lgpl-2.1.html)

### mongo-java-driver-2.4.jar

**Description:**

Java Driver for MongoDB

**License:**

The Apache Software License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

### snakeyaml-1.25.jar

**Description:**

YAML 1.1 parser and emitter for Java

**License:**

Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

### spring-boot-2.2.4.RELEASE.jar

**Description:**

Spring Boot

**License:**

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

### spring-boot-starter-web-2.2.4.RELEASE.jar

**Description:**

Starter for building web, including RESTful, applications using Spring

MVC. Uses Tomcat as the default embedded container

**License:**

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

### spring-core-5.2.3.RELEASE.jar

**Description:**

Spring Core

**License:**

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

### spring-web-5.2.3.RELEASE.jar

**Description:**

Spring Web

**License:**

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

### spring-webmvc-5.2.3.RELEASE.jar

**Description:**

Spring Web MVC

**License:**

Apache License, Version 2.0: https://www.apache.org/licenses/LICENSE-2.0

### tomcat-embed-core-9.0.30.jar

**Description:**

Core Tomcat implementation

**License:**

Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

### tomcat-embed-websocket-9.0.30.jar

**Description:**

Core Tomcat implementation

**License:**

Apache License, Version 2.0: http://www.apache.org/licenses/LICENSE-2.0.txt

## Analyze Results

1. False positive occur from a dependency-check could cause by a misunderstanding of how dependency works, problem with the tool, or configuration. It is important to filter out the false positive because you should not be wasting valuable time addressing non-existing problem and miss the real vulnerabilities. The false positive can be overwhelming if there a lot of vulnerabilities need to be addressed, result in waste of resources an these false positives are primarily on the CPE values. If the CPE value is wrong, this is usually obvious, one should use the suppression feature in the report to generate a suppression XML file that can be used on future scans or update the newer version of library and environment usually reduce the false positive.
2. Vulnerabilities and solutions:

* **hibernate-validator-6.0.18.Final.jar**

**CVE-2020-10693**  
Summary: A flaw was found in Hibernate Validator version 6.1.2.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages.  
**Solution**: You can pass user input as an expression variable by unwrapping the context to HibernateConstraintValidatorContext. Or upgrade to current version that have been patch by RedHat for apache and oracle.

* **jackson-databind-2.10.2.jar**

**CVE-2020-25649**   
Summary: A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity.  
**Solution**: Upgrade to version 2.6.7.4, 2.9.10.7, 2.10.5.1, 2.11.0 and later

**CVE-2020-36518**

jackson-databind before 2.13.0 allows a Java StackOverflow exception and denial of service via a large depth of nested objects.

**Solution**: Upgrade to version 2.14.0, 2.13.3, 2.12.6.1 or a later version

**CVE-2022-42003**

In FasterXML jackson-databind before 2.14.0-rc1, resource exhaustion can occur because of a lack of a check in primitive value deserializers to avoid deep wrapper array nesting, when the UNWRAP\_SINGLE\_VALUE\_ARRAYS feature is enabled.

**Solution**: Upgrade to version 2.13.4.1 and 2.12.17.1 or later version

**CVE-2022-42004**

In FasterXML jackson-databind before 2.13.4, resource exhaustion can occur because of a lack of a check in BeanDeserializer.\_deserializeFromArray to prevent use of deeply nested arrays. An application is vulnerable only with certain customized choices for deserialization.

**Solution**: Upgrade to version later than 2.13.4

* **log4j-api-2.12.1.jar**

**CVE-2020-9488**

Summary: Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender.  
**Solution**: Upgrade to 2.13.2 which supports this feature. Previous versions can set the system property mail.smtp.ssl.checkserveridentity to true to globally enable hostname verification for SMTPS connections.

* **logback-core-1.2.3.jar**

**CVE-2021-42550**

In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers.

**Solution**: Upgrade to version later than 1.2.7

* **mongo-java-driver-2.4.jar**  
  **CVE-2021-20328 (OSSINDEX)**

Specific versions of the Java driver that support client-side field level encryption (CSFLE) fail to perform correct host name verification on the KMS server’s certificate. This vulnerability in combination with a privileged network position active MITM attack could result in interception of traffic between the Java driver and the KMS service rendering Field Level Encryption ineffective. This issue was discovered during internal testing and affects all versions of the Java driver that support CSFLE. The Java async, Scala, and reactive streams drivers are not impacted. This vulnerability does not impact driver traffic payloads with CSFLE-supported key services originating from applications residing inside the AWS, GCP, and Azure network fabrics due to compensating controls in these environments. This issue does not impact driver workloads that don’t use Field Level Encryption.  
**Solution**: update to mongo-java-driver, mongodb-driver, mongodb-driver-sync, mongodb-driver-legacy: 3.11.3, 3.12.8.

mongodb-driver-sync, mongodb-driver-legacy: 4.0.6, 4.1.2, 4.2.1

* **snakeyaml-1.25.jar**

**CVE-2022-1471 (OSSINDEX)**

SnakeYaml's Constructor() class does not restrict types which can be instantiated during deserialization. Deserializing yaml content provided by an attacker can lead to remote code execution.

**Solution**: Using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization, or update latest versions of SnakeYaml

**CVE-2017-18640**

The Alias feature in SnakeYAML before 1.26 allows entity expansion during a load operation, a related issue to CVE-2003-1564.

**Solution:** Update version of SnakeYAML after 1.26

**CVE-2022-25857**

The package org.yaml:snakeyaml from 0 and before 1.31 are vulnerable to Denial of Service (DoS) due missing to nested depth limitation for collections.

**Solution:** Update version of SnakeYAML after 1.31

**CVE-2022-38749**

**CVE-2022-38751**

**CVE-2022-38752**

**CVE-2022-41854**

**CVE-2022-38750**

Using snakeYAML to parse untrusted YAML files may be vulnerable to Denial of Service attacks (DOS). If the parser is running on user supplied input, an attacker may supply content that causes the parser to crash by stackoverflow.

**Solution:** Update latest version of SnakeYAML

* **spring-boot-2.2.4.RELEASE.jar**

**CVE-2022-27772**

\*\* UNSUPPORTED WHEN ASSIGNED \*\* spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer.

* **spring-aop-5.2.3.RELEASE.jar**  
  **CVE-2022-22965**

Description: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.

A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.  
**Solution**: Update to latest version.

* **tomcat-embed-core-9.0.30.jar**  
  When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed: - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.  
  **Solution**: Update to current version.
* **tomcat-embed-websocket-9.0.30.jar**

When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed: - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.

**Solution**: Upgrade to current version.