# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **1/26/2025** | **Tristan Maloy** | **Initial Comments Created** |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In this report, identify your security vulnerability findings and recommend the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also include images or supporting materials. If you include them, make certain to insert them in the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Tristan Maloy

**1. Interpreting Client Needs**

Determine your client’s needs and potential threats and attacks associated with the company’s application and software security requirements. Consider the following questions regarding how companies protect against external threats based on the scenario information:

* What is the value of secure communications to the company?
* Are there any international transactions that the company produces?
* Are there governmental restrictions on secure communications to consider?
* What external threats might be present now and in the immediate future?
* What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies?

Artemis Financial is a consulting company that creates customized financial plans for its customers that include savings, retirement, investments, and insurance. Not only will security be of utmost importance to Artemis Financial, but their customers also value security as they will want their financial plans, assets, and PII protected from potential hackers and attacks.

Being a financial firm, there is a real possibility for international transactions which can pose added security vulnerabilities. Within the US government, there is the Secure and Trusted Communications Networks Act of 2019 to consider which established a mechanism to prevent equipment or services that pose a national security risk from entering U.S networks while also removing any such equipment or services that are currently used in U.S. networks. Therefore, it is essential that the security of Artemis Financials program is tightly controlled.

When dealing with finance, assets such as money and PII are big targets for hackers and organized cybercrime. Whether it is through direct hacking methods for access to information within the program, DOS attacks, ransomware, or other forms of malware, there will be plenty of attempts from external sources to access the program which Global Rain is prepared to protect from.

When considering modernization requirements, open-source libraries will need to be updated and evolving web application technologies will also need to be observed and implemented in order to increase knowledge of new potential vulnerabilities.

**2. Areas of Security**

Refer to the vulnerability assessment process flow diagram. Identify which areas of security apply to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

Artemis Financial has a RESTful web API which should allow focus on the following areas of security: input validation, APIs, Cryptography, Client / Server, and encapsulation.

Validating user input can prevent injection attacks like SQL injection and XSS. This will allow the API to only accept expected data types and values while preventing malicious input that can execute unauthorized code or access sensitive data such as PII. A secure API using HTTPS is essential for secure communication over the internet that also helps with cryptography. Especially since the information being shared is private and sensitive data, the data flow should be encrypted to and from the client/server. On top of encryption, authentication and authorization should be secure so the client can be ensured that only the authorized users have access to their respective data. Encapsulation should also be secure to protect data from getting altered during a potential attack or injection.

**3. Manual Review**

Continue working through the vulnerability assessment process flow diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

Pom.xml file should have an updated spring boot version of 3.4.1 to be able to catch more dependency vulnerabilities in static testing.

myDateTime.Java is coded wrong. Accessors and Mutators not named properly i.e. retrieveDateTime() should be public int getMyDateTime() and it also returns the incorrect variable.

setMyDateTime() is incomplete – should contain this.mySecond = seconds, this.myMinute = minutes, this.myHour = hour.

Both methods are also not declared as public in myDateTime.java.

GreetingController.java should have @RequestMapping(“/greeting”) rather than @GetMapping(“/greeting”).

CRUD.java can contain better encapsulation with setters for specifically content1 and content2.

DocData() method incomplete.

Introduce setter setId() under DocData.java

There is no input validation in any of the classes.

**4. Static Testing**

Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from the dependency-check report. Include the following items:

* The names or vulnerability codes of the known vulnerabilities
* A brief description and recommended solutions provided by the dependency-check report
* Any attribution that documents how this vulnerability has been identified or documented previously

|  |  |  |
| --- | --- | --- |
| Dependency | Vulnerability ID | Description |
| bcprov-jdk15on-1.46.jar | CVE-2024-34447 (OSSINDEX) | CWE-297 Improper Validation of Certificate with Host Mismatch |
| CVE-2016-1000338  CVE-2016-1000342 | CWE-347 Improper Verification of Cryptographic Signature |
| CVE-2016-1000343  CVE-2016-1000344  CVE-2016-1000352  CVE-2016-1000339  CVE-2015-7940  CVE-2013-1624 | CWE-310 Cryptographic Issues |
| CVE-2024-29857 (OSSINDEX) | CWE-125 Out-of-bounds Read |
| CVE-2024-30171 (OSSINDEX)  CVE-2017-13098  CVE-2020-26939 (OSSINDEX) | CWE-203 Observable Discrepancy |
| CVE-2016-1000341  CVE-2016-1000345 | CWE-361 7PK - Time and State |
| CVE-2020-15522 | CWE-362 Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition') |
| CVE-2020-0187 (OSSINDEX) | In engineSetMode of BaseBlockCipher.java, there is a possible incorrect cryptographic algorithm chosen due to an incomplete comparison. |
| CVE-2023-33202 | CWE-400 Uncontrolled Resource Consumption |
| CVE-2023-33201 (OSSINDEX) | CWE-295 Improper Certificate Validation |
| CVE-2015-7940 | CWE-200 Exposure of Sensitive Information to an Unauthorized Actor |
| CVE-2018-5382 | CWE-327 Use of a Broken or Risky Cryptographic Algorithm, CWE-354 Improper Validation of Integrity Check Value |
| CVE-2016-1000346 | CWE-320 Key Management Errors |
| CVE-2015-6644 (OSSINDEX) | CWE-200 Exposure of Sensitive Information to an Unauthorized Actor |
| hibernate-validator-6.0.18.Final.jar | CVE-2023-1932 (OSSINDEX) | CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') |
| CVE-2020-10693 | CWE-20 Improper Input Validation |
| jackson-databind-2.10.2.jar | CVE-2020-25649 | CWE-611 Improper Restriction of XML External Entity Reference |
| CVE-2020-36518 | CWE-787 Out-of-bounds Write |
| CVE-2021-46877  CVE-2023-35116 | CWE-770 Allocation of Resources Without Limits or Throttling |
| CVE-2022-42003  CVE-2022-42004 | CWE-502 Deserialization of Untrusted Data |
| log4j-api-2.12.1.jar | CVE-2020-9488 | CWE-295 Improper Certificate Validation |
| logback-classic-1.2.3.jar | CVE-2023-6378  CVE-2021-42550 | CWE-502 Deserialization of Untrusted Data |
| logback-core-1.2.3.jar | CVE-2023-6378  CVE-2021-42550 | CWE-502 Deserialization of Untrusted Data |
| CVE-2024-12798 (OSSINDEX) | CWE-917 Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') |
| CVE-2024-12801 (OSSINDEX) | CWE-918 Server-Side Request Forgery (SSRF) |
| snakeyaml-1.25.jar | CVE-2022-1471 | CWE-502 Deserialization of Untrusted Data, CWE-20 Improper Input Validation |
| CVE-2017-18640  CVE-2022-25857 | CWE-776 Improper Restriction of Recursive Entity References in DTDs ('XML Entity Expansion') |
| CVE-2022-38749 | CWE-787 Out-of-bounds Write, CWE-121 Stack-based Buffer Overflow |
| CVE-2022-38751  CVE-2022-38752  CVE-2022-41854  CVE-2022-38750 | CWE-787 Out-of-bounds Write, CWE-121 Stack-based Buffer Overflow |
| spring-boot-2.2.4.RELEASE.jar | CVE-2023-20873 | In Spring Boot versions 3.0.0 - 3.0.5, 2.7.0 - 2.7.10, and older unsupported versions, an application that is deployed to Cloud Foundry could be susceptible to a security bypass. Users of affected versions should apply the following mitigation: 3.0.x users should upgrade to 3.0.6+. 2.7.x users should upgrade to 2.7.11+. Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+. |
| CVE-2022-27772 | CWE-668 Exposure of Resource to Wrong Sphere |
| CVE-2023-20883 | CWE-400 Uncontrolled Resource Consumption |
| spring-core-5.2.3.RELEASE.jar | CVE-2022-22965 | CWE-94 Improper Control of Generation of Code ('Code Injection') |
| CVE-2021-22118 | CWE-269 Improper Privilege Management, CWE-668 Exposure of Resource to Wrong Sphere |
| CVE-2020-5421 | In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter. |
| CVE-2022-22950  CVE-2022-22971  CVE-2022-22970 | CWE-770 Allocation of Resources Without Limits or Throttling |
| CVE-2023-20861 | In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition. |
| CVE-2023-20863 | CWE-400 Uncontrolled Resource Consumption, CWE-917 Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') |
| CVE-2022-22968 | CWE-178 Improper Handling of Case Sensitivity |
| CVE-2021-22060 | In Spring Framework versions 5.3.0 - 5.3.13, 5.2.0 - 5.2.18, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase. |
| CVE-2021-22096 | In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries.  CWE-117 Improper Output Neutralization for Logs |
| spring-expression-5.2.3.RELEASE.jar | CVE-2022-22965 | CWE-94 Improper Control of Generation of Code ('Code Injection') |
| CVE-2021-22118 | CWE-269 Improper Privilege Management, CWE-668 Exposure of Resource to Wrong Sphere |
| CVE-2020-5421 | In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter. |
| CVE-2022-22950  CVE-2022-22971  CVE-2024-38808 (OSSINDEX)  CVE-2022-22970 | CWE-770 Allocation of Resources Without Limits or Throttling |
| CVE-2023-20861 | In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition. |
| CVE-2023-20863 | CWE-400 Uncontrolled Resource Consumption, CWE-917 Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') |
| CVE-2022-22968 | CWE-178 Improper Handling of Case Sensitivity |
| CVE-2021-22060 | In Spring Framework versions 5.3.0 - 5.3.13, 5.2.0 - 5.2.18, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase. |
| CVE-2021-22096 | In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries.  CWE-117 Improper Output Neutralization for Logs |
| spring-web-5.2.3.RELEASE.jar | CVE-2016-1000027 | CWE-502 Deserialization of Untrusted Data |
| CVE-2022-22965 | CWE-94 Improper Control of Generation of Code ('Code Injection') |
| CVE-2024-38809 (OSSINDEX)  CVE-2024-38828 (OSSINDEX)  CVE-2023-20863 | CWE-400 Uncontrolled Resource Consumption |
| CVE-2024-22243 (OSSINDEX)  CVE-2024-22262 (OSSINDEX) | CWE-601 URL Redirection to Untrusted Site ('Open Redirect') |
| CVE-2021-22118 | CWE-269 Improper Privilege Management, CWE-668 Exposure of Resource to Wrong Sphere |
| CVE-2020-5421 | In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter. |
| CVE-2022-22950  CVE-2022-22971  CVE-2022-22970 | CWE-770 Allocation of Resources Without Limits or Throttling |
| CVE-2023-20861 | In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition. |
| CVE-2022-22968 | CWE-178 Improper Handling of Case Sensitivity |
| CVE-2021-22060 | In Spring Framework versions 5.3.0 - 5.3.13, 5.2.0 - 5.2.18, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase. |
| CVE-2021-22096 | In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries.  CWE-117 Improper Output Neutralization for Logs |
| spring-webmvc-5.2.3.RELEASE.jar | CVE-2022-22965 | CWE-94 Improper Control of Generation of Code ('Code Injection') |
| CVE-2024-38816 (OSSINDEX) | CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') |
| CVE-2021-22118 | CWE-269 Improper Privilege Management, CWE-668 Exposure of Resource to Wrong Sphere |
| CVE-2020-5421 | In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter. |
| CVE-2022-22950  CVE-2022-22971  CVE-2022-22970 | CWE-770 Allocation of Resources Without Limits or Throttling |
| CVE-2023-20861 | In Spring Framework versions 6.0.0 - 6.0.6, 5.3.0 - 5.3.25, 5.2.0.RELEASE - 5.2.22.RELEASE, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition. |
| CVE-2023-20863 | CWE-400 Uncontrolled Resource Consumption, CWE-917 Improper Neutralization of Special Elements used in an Expression Language Statement ('Expression Language Injection') |
| CVE-2022-22968 | CWE-178 Improper Handling of Case Sensitivity |
| CVE-2021-22060 | In Spring Framework versions 5.3.0 - 5.3.13, 5.2.0 - 5.2.18, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries. This is a follow-up to CVE-2021-22096 that protects against additional types of input and in more places of the Spring Framework codebase. |
| CVE-2021-22096 | In Spring Framework versions 5.3.0 - 5.3.10, 5.2.0 - 5.2.17, and older unsupported versions, it is possible for a user to provide malicious input to cause the insertion of additional log entries.  CWE-117 Improper Output Neutralization for Logs |
| tomcat-embed-core-9.0.30.jar | CVE-2020-1938 | 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. |
| CVE-2020-11996 | A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat 10.0.0-M1 to 10.0.0-M5, 9.0.0.M1 to 9.0.35 and 8.5.0 to 8.5.55 could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive. |
| CVE-2020-13934 | CWE-401 Missing Release of Memory after Effective Lifetime, CWE-476 NULL Pointer Dereference |
| CVE-2020-13935 | CWE-835 Loop with Unreachable Exit Condition ('Infinite Loop') |
| CVE-2020-17527  CVE-2021-25122 | CWE-200 Exposure of Sensitive Information to an Unauthorized Actor |
| CVE-2021-41079 | CWE-20 Improper Input Validation, CWE-835 Loop with Unreachable Exit Condition ('Infinite Loop') |
| CVE-2022-29885 | The documentation of Apache Tomcat 10.1.0-M1 to 10.1.0-M14, 10.0.0-M1 to 10.0.20, 9.0.13 to 9.0.62 and 8.5.38 to 8.5.78 for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. While the EncryptInterceptor does provide confidentiality and integrity protection, it does not protect against all risks associated with running over any untrusted network, particularly DoS risks.  CWE-400 Uncontrolled Resource Consumption |
| CVE-2022-42252  CVE-2023-46589  CVE-2021-33037  CVE-2019-17569  CVE-2020-1935 | CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request/Response Smuggling') |
| CVE-2023-44487 | The HTTP/2 protocol allows a denial of service (server resource consumption) because request cancellation can reset many streams quickly, as exploited in the wild in August through October 2023.  CWE-400 Uncontrolled Resource Consumption |
| CVE-2020-9484 | CWE-502 Deserialization of Untrusted Data |
| CVE-2021-25329 | The fix for CVE-2020-9484 was incomplete. When using Apache Tomcat 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41, 8.5.0 to 8.5.61 or 7.0.0. to 7.0.107 with a configuration edge case that was highly unlikely to be used, the Tomcat instance was still vulnerable to CVE-2020-9494. Note that both the previously published prerequisites for CVE-2020-9484 and the previously published mitigations for CVE-2020-9484 also apply to this issue. |
| CVE-2021-30640 | CWE-116 Improper Encoding or Escaping of Output |
| CVE-2022-34305 | CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') |
| CVE-2023-41080 | CWE-601 URL Redirection to Untrusted Site ('Open Redirect') |
| CVE-2021-24122 | CWE-706 Use of Incorrectly-Resolved Name or Reference, CWE-200 Exposure of Sensitive Information to an Unauthorized Actor |
| CVE-2023-42795 | CWE-459 Incomplete Cleanup |
| CVE-2023-45648 | CWE-20 Improper Input Validation, Users are recommended to upgrade to version 11.0.0-M12 onwards, 10.1.14 onwards, 9.0.81 onwards or 8.5.94 onwards, which fix the issue. |
| CVE-2024-21733 | CWE-209 Generation of Error Message Containing Sensitive Information |
| CVE-2020-13943 | If an HTTP/2 client connecting to Apache Tomcat 10.0.0-M1 to 10.0.0-M7, 9.0.0.M1 to 9.0.37 or 8.5.0 to 8.5.57 exceeded the agreed maximum number of concurrent streams for a connection (in violation of the HTTP/2 protocol), it was possible that a subsequent request made on that connection could contain HTTP headers - including HTTP/2 pseudo headers - from a previous request rather than the intended headers. This could lead to users seeing responses for unexpected resources. |
| CVE-2023-28708 | CWE-523 Unprotected Transport of Credentials |
| CVE-2021-43980 | CWE-362 Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition') |
| tomcat-embed-websocket-9.0.30.jar | CVE-2020-1938 | 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. |
| CVE-2020-8022 | CWE-276 Incorrect Default Permissions |
| CVE-2020-11996 | A specially crafted sequence of HTTP/2 requests sent to Apache Tomcat 10.0.0-M1 to 10.0.0-M5, 9.0.0.M1 to 9.0.35 and 8.5.0 to 8.5.55 could trigger high CPU usage for several seconds. If a sufficient number of such requests were made on concurrent HTTP/2 connections, the server could become unresponsive. |
| CVE-2020-13934 | CWE-401 Missing Release of Memory after Effective Lifetime, CWE-476 NULL Pointer Dereference |
| CVE-2020-13935 | CWE-835 Loop with Unreachable Exit Condition ('Infinite Loop') |
| CVE-2020-17527  CVE-2021-25122 | CWE-200 Exposure of Sensitive Information to an Unauthorized Actor |
| CVE-2021-41079 | CWE-20 Improper Input Validation, CWE-835 Loop with Unreachable Exit Condition ('Infinite Loop') |
| CVE-2022-29885 | The documentation of Apache Tomcat 10.1.0-M1 to 10.1.0-M14, 10.0.0-M1 to 10.0.20, 9.0.13 to 9.0.62 and 8.5.38 to 8.5.78 for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. While the EncryptInterceptor does provide confidentiality and integrity protection, it does not protect against all risks associated with running over any untrusted network, particularly DoS risks.  CWE-400 Uncontrolled Resource Consumption |
| CVE-2022-42252 | CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request/Response Smuggling') |
| CVE-2023-44487 | The HTTP/2 protocol allows a denial of service (server resource consumption) because request cancellation can reset many streams quickly, as exploited in the wild in August through October 2023.  CWE-400 Uncontrolled Resource Consumption |
| CVE-2023-46589  CVE-2019-17569  CVE-2020-1935 | CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request/Response Smuggling') |
| CVE-2020-9484 | CWE-502 Deserialization of Untrusted Data |
| CVE-2021-25329 | The fix for CVE-2020-9484 was incomplete. When using Apache Tomcat 10.0.0-M1 to 10.0.0, 9.0.0.M1 to 9.0.41, 8.5.0 to 8.5.61 or 7.0.0. to 7.0.107 with a configuration edge case that was highly unlikely to be used, the Tomcat instance was still vulnerable to CVE-2020-9494. Note that both the previously published prerequisites for CVE-2020-9484 and the previously published mitigations for CVE-2020-9484 also apply to this issue. |
| CVE-2021-30640 | CWE-116 Improper Encoding or Escaping of Output |
| CVE-2022-34305 | CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') |
| CVE-2023-41080 | CWE-601 URL Redirection to Untrusted Site ('Open Redirect') |
| CVE-2021-24122 | CWE-706 Use of Incorrectly-Resolved Name or Reference, CWE-200 Exposure of Sensitive Information to an Unauthorized Actor |
| CVE-2021-33037 | CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request/Response Smuggling') |
| CVE-2023-42795 | CWE-459 Incomplete Cleanup |
| CVE-2023-45648 | Users are recommended to upgrade to version 11.0.0-M12 onwards, 10.1.14 onwards, 9.0.81 onwards or 8.5.94 onwards, which fix the issue.  CWE-20 Improper Input Validation |
| CVE-2024-21733 | CWE-209 Generation of Error Message Containing Sensitive Information |
| CVE-2020-13943 | If an HTTP/2 client connecting to Apache Tomcat 10.0.0-M1 to 10.0.0-M7, 9.0.0.M1 to 9.0.37 or 8.5.0 to 8.5.57 exceeded the agreed maximum number of concurrent streams for a connection (in violation of the HTTP/2 protocol), it was possible that a subsequent request made on that connection could contain HTTP headers - including HTTP/2 pseudo headers - from a previous request rather than the intended headers. This could lead to users seeing responses for unexpected resources. |
| CVE-2023-28708 | CWE-523 Unprotected Transport of Credentials |
| CVE-2021-43980 | CWE-362 Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition') |

**5. Mitigation Plan**

Interpret the results from the manual review and static testing report. Then identify the steps to mitigate the identified security vulnerabilities for Artemis Financial’s software application.

Based on the manual review, all the classes are going to need updates. There are spring boot updates that need to be performed for example utilizing @RequestMapping instead of @GetMapping in GreetingController.java. Encapsulation needs to be implemented better amongst the classes as well. There are some variables that should be private. The program also includes multiple unfinished methods that do get recalled in other classes. There is no input validation, and input formatting is also absent with the exception of the Greeting() method in GreetingController.java. These all need to be implemented into the program to help increase security.

The static testing report found 15 vulnerable dependencies with a total of 155 vulnerabilities identified. Bouncy Castle needs to be updated, as this is a huge opportunity for cryptography security increase. Hibernate Validator should also be updated as there are bugs that allow potential XSS attacks, and invalid expressions being evaluated as valid. FasterXML Jackson Databind contains a vulnerability that affects data integrity and DOS attacks via out of bounds exceptions so updating this is required. Apache Log4j should also be updated to at least version 2.12.3 and 2.13.1 to allow proper certificate verification. Logback version needs updating to prevent DOS attacks, and malicious code injections. Updating snakeYAML will help prevent DOS attacks and remote code execution. It is recommended to use SnakeYaml’s SafeConstructor when parsing untrusted content to restrict deserialization. Spring Boot needs to be updated as well. According to the pom.xml file, the version being used is 2.2.4 while the most current version is 3.4.1. Apache Tomcat can also be updated to help reduce server congestion while also decreasing the chances of information leaks, deserialization of untrusted data, and aiding in input validation.