# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **March 23, 2025** | **Jermaine Wiggins** | **Vulnerability assessment report** |

## 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

Jermaine Wiggins

**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 uses a web-based software to conduct its business operation, this includes developing individualized financial plans for its customers for their saving, retirement, investments, and insurance. Since they will need access to sensitive information to conduct this analysis protecting this data is a must to build trust and a strong reputation with current and new customers. If the company chooses to operate globally, it will need to follow international financial security standards as well as any regional cyber security laws, in addition to any encryption standards. The external threats that should be considered now and in the future are brute force attacks, injection attacks, and data interception through http request because of poor API security, while monitoring potential emerging threats. Modernization requirements that should be considered are ensuring all dependencies are up to date and reviewed through OWASP dependency check.

**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.

The areas of security that must be addressed for Artemis Financial’s software application, based on their requirements and the vulnerability assessment process flow diagram, include input validation, secure APIs, encryption/cryptography, secure error handling, and code quality. Input validation will protect against injection attacks by ensuring the input is checked. Securing the APIs will prevent unauthorized access and data interception. Cryptography safeguards sensitive data being stored and in transit. Code quality ensures the code is through checked and follows industry standards/practices which leads to less vulnerabilities.

**3. Manual Review**

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

At first glance and going through all .java files there is no sign of any encryption on any data being returned or gathered anywhere, there is also no authentication on the API.

CRUDController class

* No input validation/ sanitation in business name request
* RequestMapping could allow any type of request with no restriction

DataDoc

* Username and password are hard coded

GreetingController

* No input validation for name and passed directly to the HTTP request

**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

**Vulnerabilities from most critical**

**tomcat-embed-websocket-9.0.30.jar: Critical**

* Description: Apache Tomcat treats Apache JServ Protocol (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.

Required Action: Apply latest updates

CVE codes: CVE-2020-1938, CVE-2025-24813, CVE-2020-8022, CVE-2020-11996, CVE-2020-13934, CVE-2020-13935, CVE-2020-17527, CVE-2021-25122, CVE-2021-41079, CVE-2022-29885, CVE-2022-42252, CVE-2023-44487, CVE-2023-46589, CVE-2024-3828, CVE-2020-948, CVE-2021-25329, CVE-2021-30640, CVE-2022-34305, CVE-2023-41080, CVE-2021-24122, CVE-2021-33037, CVE-2023-42795, CVE-2023-45648, CVE-2024-21733, CVE-2019-17569, CVE-2020-1935, CVE-2020-13943, CVE-2023-28708, CVE-2021-43980.

**tomcat-embed-core-9.0.30.jar: Critical**

* Description: Apache Tomcat treats Apache JServ Protocol (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.

Required Action: Apply latest updates

CVE codes CVE-2020-1938, CVE-2025-24813, CVE-2020-11996, CVE-2020-13934, CVE-2020-13935, CVE-2020-17527, CVE-2021-25122, CVE-2021-41079, CVE-2022-29885, CVE-2022-42252, CVE-2023-44487, CVE-2023-46589, CVE-2024-38286, CVE-2020-9484, CVE-2021-25329, CVE-2021-30640, CVE-2022-34305, CVE-2023-41080, CVE-2021-24122, CVE-2021-33037, CVE-2023-42795, CVE-2023-45648, CVE-2024-21733, CVE-2019-17569, CVE-2020-1935, CVE-2020-13943, CVE-2023-28708, CVE-2021-43980.

**spring-webmvc-5.2.3.RELEASE.jar: Critical**

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

CVE codes: CVE-2022-22965, CVE-2024-38816, CVE-2021-22118, CVE-2020-5421, CVE-2022-22950, CVE-2022-22971, CVE-2023-20861, CVE-2023-20863, CVE-2022-22968, CVE-2022-22970, CVE-2021-22060, CVE-2021-22096.

**spring-web-5.2.3.RELEASE.jar: Critical**

* Description: Spring web MVC. Older versions or JDK 9+ could lead to remote code execution or privilege escalation, websocket endpoint vulnerable to denial-of-service attacks and possible for user to provide malicious input.
* Required Action: Apply latest updates

CVE codes: CVE-2016-1000027, CVE-2022-22965, CVE-2024-38809, CVE-2024-22243, CVE-2021-22118, CVE-2024-38828, CVE-2020-5421, CVE-2022-22950, CVE-2022-22971, CVE-2023-20861, CVE-2023-20863, CVE-2022-22968, CVE-2022-22970, CVE-2021-22060, CVE-2021-22096.

**spring-expression-5.2.3.RELEASE.jar: Critical**

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

CVE codes: CVE-2022-22965, CVE-2021-22118, CVE-2020-5421, CVE-2022-22950, CVE-2022-22971, CVE-2023-20861, CVE-2023-20863, CVE-2024-38808, CVE-2022-22968, CVE-2022-22970, CVE-2021-22060, CVE-2021-22096

**spring-core-5.2.3.RELEASE.jar: Critical**

* Description: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.
* Required Action: Apply updates per vendor instructions.

CVE codes: CVE-2022-22965, CVE-2021-22118, CVE-2020-5421, CVE-2022-22950, CVE-2022-22971, CVE-2023-20861, CVE-2023-20863, CVE-2022-22968, CVE-2022-22970, CVE-2021-22060, CVE-2021-22096

**spring-boot-starter-web-2.2.4.RELEASE.jar: Critical**

* Description: Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container. Older version prone to denial of service if used with reverse proxy cache and security bypass.
* Required Action: Update to latest version

CVE codes: CVE-2023-20873, CVE-2022-27772, CVE-2023-20883

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

* Description: 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.
* Required Action: 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 codes: CVE-2023-20873, CVE-2022-27772, CVE-2023-20883

**snakeyaml-1.25.jar: Critical**

* Description: 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. We recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization.
* Required Action: Upgrade to version 2.0 and beyond.

CVE codes: CVE-2022-1471, CVE-2017-18640, CVE-2022-25857, CVE-2022-38749, CVE-2022-38751, CVE-2022-38752, CVE-2022-41854

**bcprov-jdk15on-1.46.jar: High**

* Description: Improper validation of cryptographic parameters
* Required Action: Update to version 2.0 and beyond

CVE codes: CVE-2024-34447, CVE-2016-1000338, CVE-2016-1000342, CVE-2016-1000343, CVE-2024-29857, CVE-2016-1000344, CVE-2016-1000352, CVE-2024-30171, CVE-2016-100034, CVE-2016-1000345, CVE-2017-13098, CVE-2020-15522, CVE-2020-0187, CVE-2023-33202, CVE-2020-26939, CVE-2023-33201, CVE-2016-100033, CVE-2015-7940, CVE-2018-5382, CVE-2013-1624, CVE-2016-1000346, CVE-2015-6644.

**logback-core-1.2.3.jar: High**

* Description: A serialization vulnerability in logback receiver component part of dated version allows an attacker to mount a Denial-Of-Service attack by sending poisoned data, gain required privileges to edit configurations or inject malicious configurations.
* Required Action: Update to latest version

CVE codes: CVE-2023-6378, CVE-2021-42550, CVE-2024-12798, CVE-2024-12801

**logback-classic-1.2.3.jar: High**

* Description: A serialization vulnerability in logback receiver component part of dated version allows an attacker to mount a Denial-Of-Service attack by sending poisoned data, or gaining required privileges to edit configurations.
* Required Action: Update to latest version

CVE codes: CVE-2023-6378, CVE-2021-42550

**jackson-databind-2.10.2.jar: High**

* Description: General data-binding functionality for Jackson: works on core streaming API. Older version allow for denial of service, 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.
* Required Action: Update to latest version

CVE codes: CVE-2020-25649, CVE-2020-36518, CVE-2021-46877, CVE-2022-42003, CVE-2022-42004, CVE-2023-35116

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

Description: Hibernate’s BeanValidation (JSR-380) reference implementation. Vulnerabilities in validation logic that may cause HTML injection or Cross site scripting attacks, older version allow to bypass input sanitation.

CVE codes: CVE-2020-10693, CVE-2023-1932

**log4j-api-2.12.1.jar: Low**

* Description: 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.
* Required Action: update to Apache Log4j 2.12.3 and 2.13.1

CVE code: CVE-2020-9488

**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.

Upon analyzing the manual review, no encryption was found in the code to handle sensitive data. The recommended fix is to implement an industry-standard encryption algorithm for data storage and use Transport Layer Security (TLS) for data in transit. To address the lack of authentication, implementing OAuth 2.0 is recommended to ensure only authorized users can access the API. Additionally, all user inputs should be sanitized and validated before processing to prevent injection attacks. Hardcoded database credentials should be removed and securely stored in a secrets manager. Finally, HTTP methods that don’t specify the type of request should be updated to restrict access to the allowed methods for each endpoint.

Based on the 15 vulnerable dependencies and 159 identified vulnerabilities, it’s recommended to update the following dependencies to their latest versions: Jackson Databind, Log4j API, Logback Classic/Core, Mongo Java Driver, SnakeYAML, Spring Boot, bcprov-jdk15on, and Tomcat Embedded. This will resolve all issues identified in the dependency check. Regular security checks should be performed to address any new vulnerabilities as they arise.