# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **1/26/25** | **Danielle Frankin** |  |

## 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

Danielle Franklin

**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 specializing in individualized financial plans for customers. It is critical that the company ensures strong security measures to protect sensitive financial data, have customer trust, and comply with any regulatory requirements. Financial transactions involve highly sensitive customer data like social security numbers, banking details, and investment recorders and secure communication protocols and encryption are essential to protect customer data. If there were to be any international transactions, Artemis Financial needs to comply with cross-border protection laws or regulations like GDPR and PCI DSS. Artemis Financial will need to adhere to the strict guidelines that government regulatory bodies impose. External threats could include phishing attacks, ransomware and malware, and API security risks. Artemis Financial may rely on open-source components that will need to be continuously monitored to prevent dependency-based exploits. If a cloud-base infrastructure is used, then security best practices for cloud-based security must be used like multi-factor authentication and encryption at rest and in transit.

**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 following were identified as areas of security with potential vulnerabilities:

* Input Validation – input validation is critical to preventing SQL injection.
* Cryptography – financial transactions must be encrypted to comply with data protection regulations and weak encryption algorithms can expose sensitive data.
* APIs – financial data exchanged by APIs must be properly authenticated and encrypted
* Code Error – proper error handling is critical to ensure sensitive system details are not leaked.
* Code Quality – secure coding is critical to ensure the prevention of vulnerabilities such as overflows, error handling, and weak authentication mechanisms.

**3. Manual Review**

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

Next a manual review of the code was conducted and found the following vulnerabilities. One vulnerability identified is that user input is not validated or sanitized before being processed. Another is that currently there is no authentication mechanism enforced on the API endpoints that could cause any user to access sensitive data without verifying their identity. There was also no adequate error handling found, and any error messages message may expose sensitive database details. Another key vulnerability found was that there was no data encryption, which is critical for storing information and handling transactions to comply with regulations.

**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

[Include your findings here.]

|  |  |  |  |
| --- | --- | --- | --- |
| **Dependency** | **Vulnerability** | **Description** | **Solution** |
| bcprov-jdk15on-1.46.jar | * CVE-2024-34447 (OSSINDEX) * CVE-2016-1000338 * cpe:2.3:a:bouncycastle:bouncy-castle-crypto-package:1.46:\*:\*:\*:\*:\*:\*:\* * cpe:2.3:a:bouncycastle:bouncy\_castle\_crypto\_package:1.46:\*:\*:\*:\*:\*:\*:\* * cpe:2.3:a:bouncycastle:bouncy\_castle\_for\_java:1.46:\*:\*:\*:\*:\*:\*:\* * cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:\*:\*:\*:\*:\*:\*:\* * cpe:2.3:a:bouncycastle:the\_bouncy\_castle\_crypto\_package\_for\_java:1.46:\*:\*:\*:\*:\*:\*:\* | The software communicates with a host that provides a certificate, but the software does not properly ensure that the certificate is actually associated with that host. In Bouncy Castle JCE Provider version 1.55 and earlier the DSA does not fully validate ASN.1 encoding of signature on verification. It is possible to inject extra elements in the sequence making up the signature and still have it validate, which in some cases may allow the introduction of 'invisible' data into a signed structure. | Upgrade to latest version |
| hibernate-validator-6.0.18.Final.jar | * CVE-2020-10693 * cpe:2.3:a:redhat:hibernate\_validator:6.0.18:\*:\*:\*:\*:\*:\*:\* | 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. | Upgrade to latest version |
| jackson-databind-2.10.2.jar | * CVE-2020-36518 * cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\* * cpe:2.3:a:fasterxml:jackson-modules-java8:2.10.2:\*:\*:\*:\*:\*:\*:\* | jackson-databind before 2.13.0 allows a Java StackOverflow exception and denial of service via a large depth of nested objects. | Upgrade to latest version |
| logback-classic-1.2.3.jar | * CVE-2021-42550 * cpe:2.3:a:qos:logback:1.2.3:\*:\*:\*:\*:\*:\*:\* | 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. | Upgrade to latest version |
| spring-core-5.2.3.RELEASE.jar | * CVE-2022-22965 * cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\* * cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\* * cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\* | Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | Upgrade to latest version |
| tomcat-embed-core-9.0.30.jar | * CVE-2021-25329 * CVE-2020-13935 * cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\* * cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\* | 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. The payload length in a WebSocket frame was not correctly validated in Apache Tomcat 10.0.0-M1 to 10.0.0-M6, 9.0.0.M1 to 9.0.36, 8.5.0 to 8.5.56 and 7.0.27 to 7.0.104. Invalid payload lengths could trigger an infinite loop. Multiple requests with invalid payload lengths could lead to a denial of service. | Upgrade to latest version |
| snakeyaml-1.25.jar | * CVE-2022-1471 * cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\* | 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. We recommend upgrading to version 2.0 and beyond. | Migrate to SnakeYAML Engine. It has a configuration option to restrict aliases for collections (the aliases for scalars cannot grow and they are not restricted) |

**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 findings from the manual review and static testing report, it is recommended to upgrade Bouncy Castle, Hibernate Validator, jackson-databind, logback, Spring core, and Apache Tomcat to their current versions. It is recommended to migrate to SnakeYAML engine and restrict aliases for collection.