# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **22/03/25** | **Valeria Duharte** |  |

## 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

Valeria Duharte

**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 company that deals with the finance of individuals. Secure communication is important for the company because ensures that customer’s sensitive data is being kept safe from hackers or unauthorized users. Proper security measurements and data encryption is a must when dealing with financial records. Failing to do so could result in penalties or lawsuits while damaging the company’s reputation. If the company produces international transactions, it must follow international rules about data security and payments. International transactions are highly risky and vulnerable to cyberattacks from foreign entities, threats, hacks or fraud. The need for secure financial communication systems is higher when international affairs are involved. Governmental restriction to consider include restrictions imposed by different governments on cryptographic algorithms that can alter the company’s choice of security measures. External threats Artemis Financial might encounter at some point include cyberattacks, data breaches, advanced persistent threats, and manipulation of data in transit. Weak authentication, poor input validation, unsecured APIs are signs of a vulnerable software giving hackers an advantage to access sensitive data. Modernization requirements the company should consider are open-source libraries and cloud security. Libraries will need to be regularly updated and carefully managed to keep the data secured. Artemis could also use a cloud-based infrastructure to store data, shared resources and multi tenancy while encrypting the data in transit and at rest.

**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.

To reference the vulnerability assessment process, we will apply the findings from Step 1.

* Input Validation. Is needed to prevent attacks that can compromise the web application. As user inputs sensitive data, like username, passwords, credit card numbers, the application will check and validate the owner’s information, providing protection for users.
* APIs. Used to communicate between the web application and other systems. If vulnerable it could expose sensitive data or allow unauthorized access. Authentication mechanisms like encrypted data and input/output being properly validated is a way to prevent data breaches or unauthorized transactions for Artemis Financial.
* Cryptography. It protects stored data and in transit. Encryption is critical in protecting user data and meeting regulatory requirements. It secured sensitive data handling like financial details, user credentials, transaction data, and personally identifiable information.
* Code Error. If code errors or bugs are found in the application, attackers will find it and exploit vulnerabilities. It is important to check that error handling is secure and no sensitive information is exposed to end-users.

**3. Manual Review**

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

* Vulnerabilities in the code need better security practices in input validation, secure communication, encryption and error handling.
* HTTPS communication and secure API endpoints need be enforced in the system.
* The code doesn’t have any encryption for sensitive data, or protocols for encrypting data stored or in transit.
* GreetingController.java API endpoint doesn’t have authorization checks or restrictions.
* CROUDController.java has sensitive data exposure without encapsulating.

**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
* bcprov-jdk15on-1.46.jar

The Bouncy Castle is recommended to upgrade to a version 1.56 or later.

[**CVE-2013-1624**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-1624)

**CVE-2015-6644** (OSSINDEX)

**CVE-2015-7940** (OSSINDEX)

[**CVE-2016-1000338**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000338)

[**CVE-2016-1000339**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000339)

[**CVE-2016-1000341**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000341)

[**CVE-2016-1000342**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000342)

[**CVE-2016-1000343**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000343)

[**CVE-2016-1000344**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000344)

[**CVE-2016-1000345**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000345)

[**CVE-2016-1000346**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000346)

[**CVE-2016-1000352**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000352)

[**CVE-2017-13098**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-13098)

[**CVE-2018-5382**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2018-5382)

**CVE-2020-0187** (OSSINDEX)

**CVE-2020-26939** (OSSINDEX)

**CVE-2023-33201** (OSSINDEX)

**CVE-2024-29857** (OSSINDEX)

**CVE-2024-30171** (OSSINDEX)

**CVE-2024-34447** (OSSINDEX)

* spring-boot-2.2.4.RELEASE.jar

Spring Boot. Should be upgraded to 2.7.11 or higher.

[**CVE-2022-27772**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-27772)

[**CVE-2023-20873**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20873)

[**CVE-2023-20883**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20883)

* logback-classic-1.2.3.jar

logback-classic. Should be upgraded to 1.2.13 or higher

[**CVE-2021-42550**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-42550)

[**CVE-2023-6378**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-6378)

* logback-core-1.2.3.jar

logback-core. Should be upgraded to 1.2.13 or higher

[**CVE-2021-42550**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-42550)

[**CVE-2023-6378**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-6378)

**CVE-2024-12798** (OSSINDEX)

**CVE-2024-12801** (OSSINDEX)

* log4j-api-2.12.1.jar

The Apalache Log4j API. Should be upgraded to 2.12.3 or higher

[**CVE-2020-9488**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9488)

[**CVE-2021-44228**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-44228)

[**CVE-2021-44832**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-44832)

[**CVE-2021-45046**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-45046)

[**CVE-2021-45105**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-45105)

* snakeyaml-1.25.jar

Upgrade to 1.26 or higher

[**CVE-2017-18640**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-18640)

[**CVE-2021-4235**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-4235)

[**CVE-2022-1471**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-1471)

[**CVE-2022-25857**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-25857)

[**CVE-2022-3064**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-3064)

[**CVE-2022-38749**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38749)

[**CVE-2022-38750**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38750)

[**CVE-2022-38751**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38751)

[**CVE-2022-38752**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38752)

[**CVE-2022-41854**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-41854)

* Jackson-databind-2.10.2.jar

Upgrade to 2.12.6.1 or higher

[**CVE-2020-25649**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-25649)

[**CVE-2020-36518**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-36518)

[**CVE-2021-46877**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-46877)

[**CVE-2022-42003**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42003)

[**CVE-2022-42004**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42004)

[**CVE-2023-35116**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-35116)

* Tomcat-embed-core-9.0.30.jar

Core Tomcat implementation. It's critical to upgrade to a secure version that addresses these vulnerabilities.

[**CVE-2019-17569**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569)

[**CVE-2020-11996**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996)

[**CVE-2020-13934**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934)

[**CVE-2020-13935**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13935)

[**CVE-2020-13943**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13943)

[**CVE-2020-17527**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-17527)

[**CVE-2020-1935**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1935)

[**CVE-2020-1938**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1938)

[**CVE-2020-8022**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-8022)

[**CVE-2020-9484**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9484)

[**CVE-2021-24122**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-24122)

* Hibernate-validator-6.0.18.Final.jar

Hibernate’s Bean Validation. Upgrade to 6.0.20 version or higher.

[**CVE-2020-10693**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-10693)

**CVE-2023-1932** (OSSINDEX)

* Spring-web-5.2.3.RELEASE.jar

Spring Web. Upgrade to 5.2.17 or higher.

**CVE-2016-1000027** (OSSINDEX)

**CVE-2020-5421** (OSSINDEX)

**CVE-2021-22096** (OSSINDEX)

* Spring-beans-5.2.3.RELEASE.jar

Spring Beans. Upgrade to 5.2.3.RELEASE version or higher.

**CVE-2022-22965** (OSSINDEX)

* Spring-webmvc-5.2.3.RELEASE.jar

Spring Web MVC. Upgrade to newer version.

**CVE-2021-22060** (OSSINDEX)

**CVE-2024-38816** (OSSINDEX)

* Spring-context-5.2.3.RELEASE.jar

Spring Context. Upgrade to 5.3.19 or higher.

**CVE-2022-22968** (OSSINDEX)

* Spring-expression-5.2.3.RELEASE.jar

Spring Expression Language. Upgrade to 6.0.8 or higher.

**CVE-2022-22950** (OSSINDEX)

**CVE-2023-20861** (OSSINDEX)

**CVE-2023-20863** (OSSINDEX)

**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.

Most of the vulnerabilities would be fixed by updating the libraries to the latest versions. Is also important to keep up with regular conduct security audits, vulnerability scans, and code reviews to identify new security risks early.