# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **7/21/2024** | **Randy Ortiz** |  |

## 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

Randy Ortiz

**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?

Secure communication holds tremendous value to Artemis Financial. Secure communication is critical for protecting sensitive financial data that is transmitted between the company and clients. Also, it ensures client trust with the company by safeguarding their information from unauthorized access. Since Global Rain is a company that engineers software in agencies around the world, If Artemis Financial were to engage in internation transactions, they should adopt international data protection regulations. Governmental restrictions on secure communications should be considered. Artemis must comply with regulations for their given location. This could mean adhering to Federal Financial Institutions Examination Council guidelines if located in the United States. External attacks that might be present now or in the immediate future are phishing attacks, SQL injection, and data breaches. Modernization requirements to consider should include integrating open-sources libraries and implementing the latest web application technology. Updates should be applied as close to release as possible.

**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 applies to Artemis Financial software application are Input Validation, API’s, Cryptography, Client/Server, Code Error. Input Validation is an effective security measure to combat SQL injection attempts and is a primary defense mechanism to protect the application. Securing API’s is critical because the software relies on interactions with the RESTful API. Ensuring proper authentication, and authorization are necessary to protect data exchanges. Cryptography ensures that sensitive data is encrypted and protected, which even if the data is accessed without authorization, it will be unreadable. Client/Server is a method to protect communication channels, which can prevent man-in-the-middle attacks, bolstering confidentiality. Lastly, Code Error handling avoids information leakage or potential exploitation. This ensures that the application fails securely and does not expose sensitive data.

**3. Manual Review**

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

The first general observation I have is there is no use of cryptography or encryption across all files. input validation is missing in ‘CRUDController’ class, which risks unsafe data handling. APIs are insecure; the ‘CRUDController’ class lacks input validation in its endpoints. This means attackers could exploit this vulnerability to inject harmful data. Client/Server communication also is not secured due to the lacking HTTPS or a similar protection. Therefore, data sent between the client and server will not be encrypted. Code Error in the ‘DocData’ class is not properly implemented. This can result in error messages that may reveal system vulnerabilities that can be exploited.

**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

|  |  |  |
| --- | --- | --- |
| Name/Code | Description | Recommended Solution |
| bcprov-jdk15on-1.46.jar –  CVE-2013-1624  CVE-2015-6644  CVE-2015-7940  CVE-2016-1000338, CVE-2016-1000339, CVE-2016-1000340, CVE-2016-1000341, CVE-2016-1000342, CVE-2016-1000343, CVE-2016-1000344, CVE-2016-1000345, CVE-2016-1000346  CVE-2017-13098  CVE-2018-5382 | The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This dependency has multiple known vulnerabilities related to cryptographic operations. | Upgrade to latest version of Bouncy Castle Library, the vulnerabilities have been addressed in the later versions. |
| spring-boot-2.2.4.RELEASE.jar –  CVE-2022-27772  CVE-2023-20873  CVE-2023-20883 | Spring Boot | Upgrade to latest Spring Boot Version, since this vulnerability affects versions before 2.2.11. |
| logback-core-1.2.3.jar –  CVE-2021-42550  CVE-2023-6378 | logback-core module which provides core logging functionalities. | Upgrade to Logback version 1.2.7 or later will mitigate the risk. |
| log4j-api-2.12.1.jar –  CVE-2020-9488  CVE-2021-44228  CVE-2021-44832  CVE-2021-45046 | The Apache Log4j API, used for logging messages in Java Applications | Upgrade to latest Log4j version. |
| snakeyaml-1.25.jar –  CVE-2017-18640  CVE-2021-4235  CVE-2022-1471  CVE-2022-25857  CVE-2022-3064  CVE-2022-38749  CVE-2022-38750  CVE-2022-38751 | YAML 1.1 parser and emitter for Java | Upgrade SnakeYAML version to latest version. Consider using SafeConstructor to prevent remote code execution vulnerabilities. |

**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 off the findings, there are may steps that are needed to address the security vulnerabilities in Artemis Financials’ software application. The absence of cryptography poses a risk to sensitive data. To mitigate this risk, we must implement encryption for the data when it is at rest and in transit. A server should be implemented and configured to enforce HTTPS. The ‘CRUDController’ class is missing input validation, which can allow harmful data to be injected. This should be addressed by adding input validation for all API endpoints. A library that provides strong validation mechanisms should be implemented. Client-server communication is also insecure due to the lack of cryptography. HTTPS should be implemented to secure the data in transit. The errors in ‘DocData’ class could expose the application vulnerabilities through the error messages that are displayed. Error handling should be improved by providing user-friendly messages for users and keeping detailed error logs internally.

The static testing revealed many vulnerabilities in the dependencies. Bouncy Castle must be updated to its latest version to fix many of the vulnerabilities. Spring Boot also should be updated to address the known issues. The same goes for Logback Core and Log4j, they should be updated to the latest version. Lastly, SnakeYAML should be updated to its latest version and ‘SafeConstructor’ should be used to prevent remote code execution vulnerabilities.