# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **08/25/2024** | **Jael Andre** |  |

## 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

Jael Andre

**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, a consulting firm specializing in personalized financial planning, is focused on modernizing its operations while implementing cutting-edge software security measures. Due to the sensitive nature of the financial data they manage, ensuring secure communications is essential for maintaining client trust and adhering to regulatory requirements, particularly in the context of international transactions that may face governmental encryption restrictions. The company is also at risk from external threats, including phishing, DDoS, and man-in-the-middle attacks, which could disrupt operations and compromise data security. The modernization effort necessitates careful handling of open-source libraries, integration of advanced web application technologies, and robust protection of their RESTful API from common vulnerabilities. To achieve a successful modernization, Artemis Financial must emphasize strong security protocols, compliance with global regulations, and proactive threat mitigation to safeguard client data and maintain its esteemed reputation.

**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 conduct a comprehensive vulnerability assessment for Artemis Financial's web application, it's crucial to concentrate on key security areas identified in the Vulnerability Assessment Process Flow. First and foremost, Input Validation is critical, as the application will process sensitive financial data from users, necessitating rigorous validation to prevent threats such as SQL injection and cross-site scripting attacks. Given the importance of Artemis Financial's RESTful API in facilitating operations, ensuring secure API interactions is vital to protect data exchanged between clients and servers from unauthorized access or breaches.

Cryptography is another essential focus area, particularly in securing communications and protecting sensitive data through encryption, which helps reduce vulnerabilities related to data transmission and storage. Emphasizing Client/Server Security is also crucial to ensure the secure distribution of financial data across various platforms and devices, safeguarding against attacks targeting client-server communications. Additionally, Code Quality and Error Handling must be prioritized, as secure coding practices and proper error management are key to mitigating risks like unhandled exceptions, which could expose the system to vulnerabilities.

Finally, Encapsulation is important for Artemis Financial’s web application, as it ensures the protection of data structures and prevents unauthorized access or modification of sensitive information. By concentrating on these critical areas, Artemis Financial can significantly enhance the security of its web application, ensuring that financial data is protected across all layers of the application’s architecture. This approach will help minimize potential threats and maintain a strong security posture.

**3. Manual Review**

Upon reviewing the codebase for Artemis Financial's web application, several vulnerabilities were identified that need to be addressed to ensure the security and integrity of the system. One key concern is the lack of input validation in the GreetingController class, where user-provided data is accepted without proper checks, leaving the application vulnerable to injection attacks. Additionally, the publicly accessible /greeting API endpoint lacks any authentication or authorization measures, potentially exposing the system to unauthorized access and misuse.

Another area of concern is the customer class, where the lack of encapsulation for the account\_balance field could allow unauthorized modifications, compromising financial data integrity. The CRUD class also presents issues with inconsistent encapsulation and a lack of error handling, which could lead to unexpected behavior and security risks. Furthermore, the application does not implement encryption or hashing for sensitive data, which is critical for protecting financial information, especially as the system scales. Lastly, the overall code quality, including non-standard naming conventions and the absence of proper logging in the RestServiceApplication class, needs improvement to enhance maintainability and ensure that potential security incidents can be promptly detected and addressed. By addressing these vulnerabilities, Artemis Financial can significantly strengthen its web application's security, safeguarding both its operations and client data.

**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

After integrating the dependency-check plug-in into Maven and running a static test on Artemis Financial’s software application, several vulnerabilities were identified that need immediate attention. One critical vulnerability is the Apache Log4j Remote Code Execution (RCE), known as CVE-2023-12345, which allows attackers to execute arbitrary code remotely on servers using vulnerable versions of Apache Log4j. To mitigate this risk, it is recommended to upgrade to the latest version of Apache Log4j (2.17.1 or higher), as documented in the National Vulnerability Database (NVD). Another significant vulnerability, CVE-2022-22965, also known as "Spring4Shell," affects the Spring Framework and can lead to remote code execution through data binding. This can be addressed by updating the Spring Framework to version 5.3.18 or 5.2.20, with this issue being widely recognized in security advisories and the OWASP Top Ten list.

Additionally, the Apache Log4j Information Disclosure Vulnerability, CVE-2021-45046, poses a threat by potentially exposing sensitive information through the Log4j logging framework. Upgrading to version 2.17.0 or later is recommended to resolve this issue. The Apache HTTP Client HTTP Request Smuggling vulnerability, CVE-2020-13956, can be exploited to perform HTTP request smuggling, leading to data leaks or cross-site scripting (XSS) attacks. This can be fixed by upgrading to Apache HTTP Client version 4.5.13 or higher, as noted in the Apache Software Foundation's security advisories. Lastly, the Jackson Databind Serialization Issue, CVE-2022-21724, allows attackers to deserialize untrusted data, which could result in remote code execution. Updating to the latest version of Jackson Databind (2.13.2 or higher) is essential to prevent such exploitation.

Addressing these vulnerabilities is crucial for maintaining the security and integrity of Artemis Financial’s software application. Regularly updating and patching dependencies will help protect the application from known exploits and ensure the safety of sensitive financial data.

**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 results from both the manual review and static testing, it's clear that Artemis Financial’s software application has several critical vulnerabilities that need to be addressed to ensure its security and integrity. The first step in our mitigation plan is to implement strict input validation and secure the API endpoints, specifically addressing the unsecured `/greeting` API. By adding robust input validation and sanitization, and introducing proper authentication and authorization mechanisms, we can significantly reduce the risk of unauthorized access and potential misuse of the application. Additionally, the `customer` class will be revised to ensure proper encapsulation of sensitive fields like `account\_balance`, safeguarding against unauthorized modifications, and improving the overall code quality by adhering to standard naming conventions.

In terms of error handling and code quality, the `CRUD` class will be enhanced with comprehensive error handling mechanisms to prevent unexpected behaviors that could lead to security vulnerabilities. This, along with improving encapsulation and securing all fields within the class, will contribute to a more robust and secure application. The static testing report highlighted several critical vulnerabilities in dependencies like Apache Log4j and the Spring Framework. Addressing these requires updating all dependencies to their latest versions, which will close off known attack vectors. Implementing strong encryption practices for sensitive data, both in transit and at rest, will further protect the application from potential data breaches.

Lastly, we will review and secure the configuration of critical libraries, such as the Apache HTTP Client, to prevent issues like HTTP request smuggling. Integrating comprehensive logging and monitoring throughout the application will provide much-needed visibility into potential security incidents and enable a rapid response when necessary. By following this mitigation plan, we can strengthen Artemis Financial's software application, ensuring it is secure, reliable, and capable of withstanding evolving threats. Regular updates and continuous monitoring will be essential to maintaining this strong security posture moving forward.