# CS 305 Project One

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **5/22/2024** | **Lauren-Ann Javier** |  |

## 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

Lauren-Ann Javier

**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 specializes in creating comprehensive financial plans for individuals, including savings, retirement, investments, and insurance. Given the sensitive nature of the information they handle, such as Social Security numbers and tax details, secure communications are essential. While there are no indications that Artemis Financial operates solely within the US, the company likely engages in international transactions. Therefore, they must consider governmental restrictions, particularly those related to the protection of trade secrets. To safeguard client information from external threats, Artemis Financial must employ strong encryption methods. Ensuring the confidentiality of client data is important, requiring stringent security measures.

**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.

* Input Validation: Artemis Financial requires strong input validation to verify the identity of information owners, thereby protecting user data. This validation process would involve handling inputs as strings.
* Code Quality: High code quality allows precise control over method access based on user roles. For instance, users would only be able to access their information, preventing access to other users' data or the server itself.
* APIs: Developing APIs is essential for both internal and external operations, as they define acceptable data access parameters.
* Error Handling: Using comprehensive error handling helps identify and solve issues within the API, ensuring that user information remains secure and inaccessible to unauthorized parties.
* Cryptography: Integrating cryptography is crucial for Artemis Financial to protect user information from global threats, especially given the involvement of different currencies. This ensures data integrity and confidentiality across various regions.

**3. Manual Review**

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

* Reviewed code and conducted Vulnerability Assessment
* Analyzed POM.XML and Greeting Controller
* POM.XML: Sought verification of Apache Validator presence
* Greeting Controller: Identified lack of input validation and noted as important for future feature consideration
* API inspection revealed deficiencies
* Vulnerability exposing user input due to POST method omission
* Attempted to verify cryptography presence, but inconclusive

**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 Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7.
* Spring-boot-2.2.4.RELEASE.jar: Spring Boot
* Logback-core-1.2.3.jar: logback-core module
* Log4j-api-2.12.1.jar: The Apache Log4j API
* Snakeyaml-1.25.jar: YAML 1.1 parser and emitter for Java
* Jackson-databind-2.10.2.jar: General data-binding functionality for Jackson: works on core streaming API
* Tomcat-embed-core-9.0.20.jar: Core Tomcat implementation
* Hibernate-validator-6.0.18.Final.jar: Hibernate's Bean Validation (JSR-380) reference implementation.
* Spring-web-5.2.3.RELEASE.jar: Spring Web
* Spring-beans-5.2.3.RELEASE.jar: Spring Beans
* Spring-webmvc-5.2.3.RELEASE.jar: Spring Web MVC
* Spring-context-5.2.3.RELEASE.jar: Spring Context
* Spring-expression-5.2.3.RELEASE.jar: Spring Expression Language (SpEL)

**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.

To proactively tackle both existing and potential future issues, prioritizing the security of company and client data is important. Adopting HTTPS protocol for all communications serves as the initial step, encouraging against unauthorized access. Secondly, transitioning request parameters to headers enhances security measures. Thirdly, removing any instances of business names from hardcoded database credentials contributes to robust data protection. Implementing and activating two-factor authentication systems constitutes another critical layer in protecting user information. Finally, regularly updating all dependencies selected by the dependency check further strengthens the overall security posture.