# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **22Jan2025** | **Steven Copeland** |  |

## 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

Steven Copeland-Helzer

**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?

The client, Artemis Financial, is a consulting company that develops several different financial plans for their customers. In providing service for their customers, Artemis stores and utilizes highly sensitive customer data such as their SSN, banking, and other personal information. The value of secure communications is extremely high due to the customer data Artemis utilizes daily. There was no mention of international transactions, though it should be assumed that the company is prepared for such transactions. The main restriction to secure communications that must be considered is the Gramm-Leach-Bliley Act (GLBA). Under this U.S. law, “Financial entities must establish security controls to protect customer information from any events threatening data integrity and safety. This includes strict financial information access controls to mitigate the chances of unauthorized access and compromise.” (Kost, E. 2025). The main external threat would be cyber attacks targeting the customer’s personal and financial data. Enhanced security measures maintained by an IT team is one modernization technique to consider, as well as API integration which Artemis is already practicing with their RESTful web API.

**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 – Ensures all input is clean and within parameters, helps to prevent injection cyber-attacks.
* API’s – Artemis utilizes the RESTful web API, ensuring secure data accessibility and passage.
* Cryptography – Crucial for adding more layers of security safeguarding their customer’s sensitive data. Must be authenticated to access user accounts and information.
* Code Error – Error handling works in tandem with Input Validation and the API, essential for preventing unauthorized user access.
* Code Quality – Secure coding practices will ensure a properly built application fit for Artemis’ needs with both functionality and security.

**3. Manual Review**

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

My findings:

* POM.xml file – no Apache Validator was found.
* GreetingController.java – lack of input validation.
* DocData.java – lack of proper error handling even with the existing try and catch blocks
* API – data is accessed via URL instead of POST method, accepting input via URL can be easily exploited.
* Cryptography – no implementation of data encryption which is a must-need for Artemis in handing their customer data.
* User-based access – no implementation of access controls for different user types.
* Database – login credentials for the database are set to default.

**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

There were 15 total vulnerabilities found by the dependency check:

1. **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.

2. **hibernate-validator-6.0.18.Final.jar** - Hibernate's Bean Validation (JSR-380) reference implementation.

3. **jackson-databind-2.10.2.jar** - General data-binding functionality for Jackson: works on core streaming API.

4. **log4j-api-2.12.1.jar** - The Apache Log4j API.

5. **logback-classic-1.2.3.jar** - logback-classic module.

6. **logback-core-1.2.3.jar** - logback-core module.

7. **snakeyaml-1.25.jar** - YAML 1.1 parser and emitter for Java.

8. **spring-boot-2.2.4.RELEASE.jar** - Spring Boot.

9. **spring-boot-starter-web-2.2.4.RELEASE.jar** - Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container.

10. **spring-core-5.2.3.RELEASE.jar** - Spring Core.

11. **spring-expression-5.2.3.RELEASE.jar** - Spring Expression Language (SpEL).

12. **spring-web-5.2.3.RELEASE.jar** - Spring Web.

13. **spring-webmvc-5.2.3.RELEASE.jar** - Spring Web MVC.

14. **tomcat-embed-core-9.0.30.jar** - Core Tomcat implementation.

15. **tomcat-embed-websocket-9.0.30.jar** - Core Tomcat implementation.

**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.

All vulnerabilities found in the dependency report can be mitigated by upgrading to the latest version of each one. Proper input validation, error handling, and data encryption methods need to be implemented. Implement user-based access controls to ensure each user has appropriate permissions. Two-factor authentication functionality within the application for added security. Users with database access must have their login credentials changed from their defaults.

**References:**

Kost, E. (2025). Top 9 Cybersecurity Regulations for Financial Services. UpGuard. [https://www.upguard.com/blog/cybersecurity-regulations-financial- industry#:~:text=Laundering%20(FDIC).-,GLBA,a%20subset%20of%20the%20GLBA](https://www.upguard.com/blog/cybersecurity-regulations-financial-%09industry#:~:text=Laundering%20(FDIC).-,GLBA,a%20subset%20of%20the%20GLBA)