# CS 305 Project One

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **09/16/25** | **Crystal Greenberg** |  |

## Client



## Developer

Crystal Greenberg

**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?

Since Artemis Financial has access to private data, including social security numbers, the organization should place a high premium on secured communications. The source does not mention foreign transactions, but those regulations would have to be followed for whatever country they are dealing with. Government rules and regulations must be followed to by Artemis Financial in all contacts and transactions. Potential or existing dangers would include someone attempting to obtain the client's and the business's financial and/or personal information. If the API is not sufficiently secure, attacks could occur. Since sensitive information will be transferred back and forth between the business and the client, two-factor authentication should be used to help prevent fraudulent login attempts. Additionally, all communications should be conducted via HTTPS.

**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: Since the client would have to log in to their account, there would need to be a secure way for the client to put in their username and password.

Code Error: Errors are managed in such a way that the person attempting to access the system doesn’t know what type of error occurred.

Cryptography: Since sensitive information is being handled, making sure the information is securely encrypted, so that hackers can’t gain access to the information.

**3. Manual Review**

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

\*Business name is public

\*No input validation

\*No input error handling

\* Needs HTTPS

\*Did not see encryption

**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

|  |  |  |  |
| --- | --- | --- | --- |
| **Dependency** | **Description** | **Vulnerability** | **Solution** |
| bcprov-jdk15on-1.46.jar | A java implementation of cryptographic algorithms | Doesn’t validate that certificate matches host | Upgrade to latest version |
| hibernate-validator-6.0.18.Final.jar | Hibernate's Bean Validation (JSR-380) reference implementation. | may interpolate user-supplied input in a constraint violation message with Expression Language | Upgrade to latest version |
| jackson-core-2.10.2.jar | Core Jackson processing abstractions  (streaming API) | jackson-core 2.15.0 contains a configurable limit for how deep Jackson will traverse in an input document, defaulting to an allowable depth of 1000 | users should avoid parsing input files from untrusted sources |
| jackson-databind-2.10.2.jar | General data-binding functionality for Jackson | Flaw was found where databind did not have entity expansion secured properly | Upgrade to latest version |
| log4j-api-2.12.1.jar | The Apache Log4j API | Improper validation of certificate with host mismatch in Apache Log4j SMTP | Upgrade to latest version |
| logback-classic-1.2.3.jar | logback-classic module | A serialization vulnerability in  logback version 1.4.11 allows an attacker to mount a Denial-Of-Service  attack by sending poisoned data. | Upgrade to latest version |
| logback-core-1.2.3.jar | Logback-core module | A serialization vulnerability in logback  version 1.4.11 allows an attacker to mount a Denial-Of-Service  attack by sending poisoned data. | Upgrade to latest version |
| snakeyaml-1.25.jar | YAML 1.1 parser and emitter for Java | Constructor() class does not restrict types which can be instantiated during deserialization | recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization. Also upgrading to version 2.0 and beyond. |
| spring-aop-5.2.3.RELEASE.jar | Spring AOP(Aspect oriented programming) | May be vulnerable to remote code execution if run on Tomcat as WAR deployment | Less vulnerable deployed as a Spring-Boot executable jar |
| spring-boot-2.2.4.RELEASE.jar | Spring-Boot | an application that is deployed to Cloud Foundry could be susceptible to a security bypass | Upgrade to 3.06+ or 2.7.11+ |
| 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 | spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking | Users of affected versions should upgrade to the corresponding fixed version. |
| spring-context-5.2.3.RELEASE.jar | Spring Context | Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | Users of affected versions should upgrade to the corresponding fixed version. |
| spring-core-5.2.3.RELEASE.jar | Spring Core | Spring Framework MVC applications can be vulnerable to a “Path Traversal Vulnerability” when deployed on a non-compliant Servlet container. | Apply updates per vendor instructions. |
| spring-expression-5.2.3.RELEASE.jar | Spring Expression Language (SpEL) | Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding | Apply updates per vendor instructions. |
| spring-web-5.2.3.RELEASE.jar | Spring Web | Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding | Apply updates per vendor instructions |
| spring-webmvc-5.2.3.RELEASE.jar | Spring Web MVC | Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding | Apply updates per vendor instructions |
| tomcat-embed-core-9.0.30.jar | Core Tomcat implementation | Apache Tomcat treats Apache JServ Protocol (AJP) connections as having higher trust than, for example, a similar HTTP connection. | Apply updates per vendor instructions. |
| tomcat-embed-websocket-9.0.30.jar | Core Tomcat implementation | Improper Privilege Management Vulnerability | Apply updates per vendor instructions |

**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.

It looks like most all of the vulnerabilities are for older versions of each program, so they would need to be updated to the most current versions.

Another thing is to work on the code, putting encryption and proper input validation into the code.