****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

Table of Contents

[Document Revision History 3](#_Toc32574607)

[Client 3](#_Toc32574608)

[Instructions 3](#_Toc32574609)

[Developer 4](#_Toc32574610)

[1. Interpreting Client Needs 4](#_Toc32574611)

[2. Areas of Security 4](#_Toc32574612)

[3. Manual Review 4](#_Toc32574613)

[4. Static Testing 4](#_Toc32574614)

[5. Mitigation Plan 4](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **5/21/2022** | **Brian Golden** |  |

## Client



## Instructions

Deliver this completed vulnerability assessment report, identifying your findings of security vulnerabilities and articulating recommendations for next steps to remedy the issues you have found.

Respond to the five steps outlined below and include your findings. Replace the bracketed text on all pages with your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

## Developer

Brian Golden

## 1. Interpreting Client Needs

Determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Consider the following 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 about secure communications to consider?
* What external threats might be present now and in the immediate future?
* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

This company values secure communications to the highest extent possible. They are dealing with financial transactions and sensitive personal information which needs to be very secured. This company does not currently have any international transactions since they are just solely based in the United States at this time. They could do this in the future but should re-consult with Global Rain to make sure their software is ready for that change. Since we are dealing with financial transactions, we need to make sure we are following the Graham–Leach–Bliley Act. This requires the disclosure of information sharing practices require specific steps to protect consumer data. Some external threats that might come up now and in the immediate future are DDOS attacks to shut down the API, man-in-the-middle attacks, and SQL injection if we are using a SQL based database for the data we are storing. Open source libraries are useful because they can be independently tested and verified that they are secure.

## 2. Areas of Security

Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

Input validation, APIs, Cryptography, and code error should be the main areas of focus on this project. Input validation is necessary because the users will be entering information into fields that allow for any text. Without input validation a malicious actor could try and add some code or bad data into the fields which could cause a security violation. APIs are important because we need to be making sure that the API we are creating has secure connections that cannot be intercepted by malicious actors. This brings cryptography into play as well because we need to use encryption on all of our transactions and calls using sensitive user data. Code error is also important because it is possible that without error checking intentional errors could lead to privilege escalation or other unintended consequences.

## 3. Manual Review

Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

Starting with the CRUD class, when using any of the methods or constructors there are no checks to see of the string is empty or valid. The CRUD Controller sends the DocData straight to this program but there are no checks to see if DocData is valid in any way. After looking at DocData, there is nothing done in the constructor which means that the new CRUD line right after DocData would not contain any valid data and could crash the program. The DocData class uses the SQL Connection class with hardcoded connection data. This should be passed as a variable. This also has a hardcoded username and password section which should come from secured variables. The Greeting class does nothing to check if any of the strings that are passed are empty or valid in any way. There is nothing wrong with the GreetingController that I can see. The myDateTime class is incomplete but what is there shows static class variables which should not exist. Those should be private and only accessible via getter and setter methods.

## 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 dependency check report. Include the following:

1. The names or vulnerability codes of the known vulnerabilities
2. A brief description and recommended solutions provided by the dependency check report
3. Attribution (if any) that documents how this vulnerability has been identified or documented previously

Here is a list of the dependencies listed from most severe to least severe according to the Dependency checker, along with a brief description of the recommended solution.

|  |  |  |  |
| --- | --- | --- | --- |
| **Dependency** | **Description** | **Recommended solutions** | **Attribution (if any)** |
| tomcat-embed-websocket-9.0.30.jar and tomcat-embed-core-9.0.30.jar | Core Tomcat implementation | The fixes for the many CVEs listed in the tomcat implementation are making sure that inputs are properly validated, protecting user data, checking for possible errors, and controlling user privileges. | None |
| spring-core-5.2.3.RELEASE.jar and spring-aop-5.2.3.RELEASE.jar | Spring Core and AOP | Upgrade spring to version 5.3.19+ or 5.2.21+ | Some of the CVEs were discovered by responsible 3rd parties, others internally by the Spring team |
| bcprov-jdk15on-1.46.jar | Java implementation of cryptographic algorithms | The best solution for this is to upgrade to the most recent version of bouncycastle | N/A |
| spring-boot-2.2.4.RELEASE.jar | Spring Boot | Upgrade to 2.2.11.RELEASE or later. | Originally spotted by girhub user trugPa |
| snakeyaml-1.25.jar | YAML 1.1 parser and emitter for Java | Upgrade to 1.26 | NA |
| jackson-databind-2.10.2.jar | General data-binding functionality for Jackson: works on core streaming API | Upgrade to 2.12.6.1 or 2.13.2.2 | Reported on github by various users |
| logback-core-1.2.3.jar | Logback-core module | Upgrade to 1.2.9 as the issue is fixed in that release. | N/A |
| hibernate-validator-6.0.18.Final.jar | Hibernate's Bean Validation (JSR-380) reference implementation. | Upgrade to 6.1.5.final | N/A |
| log4j-api-2.12.1.jar | The Apache Log4j API | This should be upgraded to 2.17.1 | N/A |

## 5. Mitigation Plan

After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial’s software application.

There should be steps taken to add proper error checking and handling, have proper input validation in order to check for bad input or malicious input, have proper authentication for users, and encrypt user data properly. We also need to upgrade all of our dependencies to newer versions as the newer versions contain fixes to all know CVEs that were reported in the dependency check.