

# 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** | **5/20/21** | **Jacob Perry** |  |

## 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

Jacob Perry

## 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?**

Because Artemis is a financial institution, security is of the utmost importance. Allowing for secure communications between advisors and clients will be critical to maintain high levels of security. There is no mention of their client base being international or not, so we will say that there are no government restrictions that they will need to adhere to.   
  
The largest external threats are going to be coming from malicious users who’re hoping to gain access to all of the financial data that will be kept in the Artemis database.

## 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.**

When reviewing the VAPFD we can see many of the areas listed need to be addressed such as input validation, Client/Server, Code Error, Code Quality, and Encapsulation are most critical. Input validation will be important to ensure that the company isn’t opening itself up for any types of input attacks looking to compromise the database. Client/Server ensures that any and all communication happening with the client is taking place on a secured and encrypted communication stream. Code error and code quality will help ensure that not only is code itself tight, but that when an error occurs the development team has the appropriate knowledge to address the issue based on error codes generated. Finally, encapsulation acts as a centralizing focus for the sensitive data that Artemis is working with. Due to Artemis being a financial institution, the critical importance of maintaining a closed off database with tight security is of utmost importance.

## 3. Manual Review

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

**Upon initial review of the code, I can see multiple issues that are jumping out at me. Firstly, at no point does the code run any types of checks on the input coming into the system. This opens the system up for incoming attacks utilizing input data to access the sensitive data. Since the system is going to have user data separated by a unique ID of some value, making sure that the ID is being validated against a data table.**

The second largest issue that jumps out is a lack of any type of data encryption. We can see in the code that the application is simply taking raw data and passing into objects as unencrypted data. Similar to our validation issue explained above, without encrypting the data, we are leaving our database open to critical information such as bank account information and transaction history to anyone who can gain access.

## 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**

Looking at our list of dependencies with the largest and most severe lists of vulnerabilities include, tomcat 9.0.3, spring-core 5.2, and snakeyaml-1.25 hold the largest threats.

The vulnerabilities found with tomcat and Hibernate validation dependency include exploits involving the HTTP and validation requests that can allow for incorrect validations. Jackson databind vulnerabilities refer specifically to the connection between the user interface and the data. Even though these have the highest level of severity, they do have the lowest count of evidence.

Meanwhile we have a vulnerability with the Apache Log4, while being the lowest of level of severity, we see that it does have the highest volume of associated evidence.

List of vulnerability codes:

[**CVE-2020-10693**](https://nvd.nist.gov/vuln/detail/CVE-2020-10693)

[**CVE-2020-25649**](https://nvd.nist.gov/vuln/detail/CVE-2020-25649)

[**CVE-2020-9488**](https://nvd.nist.gov/vuln/detail/CVE-2020-9488)

[**CVE-2017-18640**](https://nvd.nist.gov/vuln/detail/CVE-2017-18640)

|  |  |
| --- | --- |
| [**CVE-2020-5421**](https://nvd.nist.gov/vuln/detail/CVE-2020-5421)**CVE-2021-25329** |  |

## 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.**

After looking through the code manually, reviewing the vulnerabilities, and researching the options for updating and fixing the most critical problems. When looking over the Jackson data-binding vulnerability, we see that it needs to be updated beyond version 2.10.5.1. While our Hibernate Validator needs to be updated beyond version 6.0.2. Some of our vulnerabilities have supported xml code that we can also integrate into our application when the option for updating the version isn’t an option.