****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

Table of Contents

CS 305 Project One 1

Document Revision History 3

Client 3

Instructions 3

Developer 4

1. Interpreting Client Needs 4

2. Areas of Security 4

3. Manual Review 4

4. Static Testing 4

5. Mitigation Plan 5

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **7/14/2022** | **Andrew Cruz** |  |

## 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

Andrew Cruz

## 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?

Due to the client being a financial company, secure communication is of the utmost priority. Since this is a web app, the company could easily be doing international transaction. There are certainly government restrictions on communications and money transfer depending on where a person lives. External threats that would be likely are people trying to hack into this system and steal the data of our users and potentially bank information. It is critical to make sure the systems are up to date and on the latest patch to prevent simple attacks from occurring.

## 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:**

Input validation is very important for a web app like this because the company needs to make sure the user attempting to login is using legitimate credentials and that those credentials weren’t taken from the system itself.

**Secure APIs:**

When developing an application like this, APIs can be an extremely helpful part of the process, so it is imperative that they are properly secured and scanned for vulnerabilities.

**Client/Server:**

This web app needs to have secure client/server interactions. The data the company deals with is very precious and sensitive, therefore it is critical that the data transmission and systems it is stored on are properly secure.

**Encapsulation:**

The data structure that holds the clients data must be secure. If the structure isn’t secure, this could compromise the entire app.

## 3. Manual Review

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

There is a variable in doc.java that isn’t used. This code doesn’t seem to have any major glaring errors.

## 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

There were 9 vulnerable dependencies found with this app.

**Bouncy castle:** Oracle has released a patch for this.

**Spring-boot:** updating to a newer version will get rid of this vulnerability.

**Logback:** There is a new version that has been pathed. An update will solve this problem.

**Log4j:** All versions affected by this exploit have released an update.

**Snakeyaml:** There is a patch released on bitbucket for this vulnerability by the developer.

**Jackson-databind:** There was a patch released for this version, however the current version also has a medium security risk.

**Tomcat-embed:** The version used here has an update, however that update also has a security risk.

**Hibernate-validator:** This vulnerability has been patched and needs to be updated.

**Spring-core:** Updating will also solve this vulnerability, however a new bug has been discovered.

## 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.

The majority of these vulnerabilities have come from outdated software. Updating to the newest version will solve 6 of the nine vulnerable dependencies. There are three dependencies that have since had new vulnerabilities discovered with them, but they are still more secure than the old versions. Every vulnerability that didn’t have a patch had a mitigation plan by the developer and a patch in progress. The best thing to do would be to update the software to the current versions and implement the recommended mitigation plan for the three dependencies: Jackson-databind, Tomcat, and Spring-Core.