****

# 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** | **5/20/2022** | **Laura McAroy** |  |

## 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

Laura McAroy

## 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?

As a company that handles financial information, it is imperative that Artemis Financial has a secure system to protect their clients’ most personal information. A security breach that results in theft or loss of clients’ financial information could mean the destruction of the company. It can only be assumed that if Artemis Financial is not already producing international transactions, they will be in the near future. Appropriate actions need to be taken to ensure that those international transactions are secure and that all governmental regulations concerning those transactions, including Graham-Leach-Bliley and PCI-DSS, are followed. This system will be at a very high risk for external threats, as it will contain a large number of desirable assets for hackers. HTTPS, OAuth, and input parameter requirements are basic steps that can be taken to better secure the REST API. It is also necessary that only reviewed third-party libraries are included in the system.

## 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 – All input into the system will need to be validated
* API’s- Since the company is using a RESTful API, it will need to be secured to maintain system integrity.
* Cryptography- All information will need to be fully encrypted to prevent data leaks and interception.
* Encapsulation – Sensitive data is stored in the system, so it will need to be stored securely in encapsulated data structures.

## 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 no HTTPS included in the code base
* In the CRUDController class, business names are used as request parameters and are a public variable in a public class.
* The request parameters are not validated
* There is no authentication included

## 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

* Bouncy Castle JCE contains several vulnerabilities in versions 1.55 and earlier, recommended to upgrade to a newer version to avoid vulnerabilities.
  + CVE-2016-1000352
  + CVE-2016-1000346
  + CVE-2016-1000345
  + CVE-2016-1000344
  + CVE-2016-1000343
  + CVE-2016-1000342
  + CVE-2016-1000341
  + CVE-2016-1000339
  + CVE-2016-1000338
  + CVE-2018-5382
  + CVE-2017-13098
  + CVE-2013-1624
* Logback version 1.2.3 contains a vulnerability that would allow an attacker with privileges to maliciously edit the configuration. This vulnerability exists in versions up to and including 1.2.7, so a higher version would be recommended.
  + CVE-2021-42550
* Apache Log4j2 contains several dependencies that were fixed in later versions. Recommended to upgrade to version 2.17.1 to avoid dependencies
  + CVE-2021-44832
  + CVE-2021-45105
  + CVE-2021-45046
  + CVE-2021-44228
  + CVE-2020-9488
* SnakeYAML-1.25 allows entity expansion during a load operation. Upgrade to version 1.26
  + CVE-2017-18640
* FasterXML Jackson Databind also contains an issue with entity expansion, risking data integrity. Upgrade to at least a 2.10.5.1 is recommended.
  + CVE-2020-25649
* Apache Tomcat contains several vulnerabilities. Recommended to upgrade to a higher version.
  + CVE-2022-29885
  + CVE-2021-41079
  + CVE-2021-33037
  + CVE-2021-30640
  + CVE-2021-25329
  + CVE-2021-25122
  + CVE-2021-24122
  + CVE-2020-17527
  + CVE-2020-13935
  + CVE-2020-13934
  + CVE-2020-8022
  + CVE-2020-11996
  + CVE-2020-9484
  + CVE-2020-1938
  + CVE-2020-1935
  + CVE-2019-17569
* SpringCore 5.2.3 has a vulnerability. Upgrade to at least a 5.3.17
  + CVE-2022-22950

## 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.

* Include authentication measures and input validation
* Implement HTTPS
* Update listed dependencies
* Encapsulate business names so that they are private if they must be used a request parameters, otherwise change request parameters.