****

# Artemis Financial Vulnerability Assessment Report

**Table of Contents**

[Document Revision History 3](#_heading=h.gjdgxs)

[Client 3](#_heading=h.30j0zll)

[Instructions 3](#_heading=h.1fob9te)

[Developer 4](#_heading=h.3znysh7)

[1. Interpreting Client Needs 4](#_heading=h.2et92p0)

[2. Areas of Security 4](#_heading=h.tyjcwt)

[3. Manual Review 4](#_heading=h.3dy6vkm)

[4. Static Testing 4](#_heading=h.1t3h5sf)

[5. Mitigation Plan 4](#_heading=h.4d34og8)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **[Date]** | **[Your name]** |  |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In the report, identify your findings of security vulnerabilities and provide recommendations for the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also choose to include images or supporting materials. If you include them, make certain to insert them in all the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Konnor Crager

## Interpreting Client Needs

Looking over the code, It seems the company wants to make communications as secure as possible. There are internal transactions in the code, as it’s for the company to use for the clients. There are not any governmental restrictions to consider, which is great when proceeding forward with this, as it helps with the protection of the user’s personal information. As far as external threats, data leaks are a common thing everywhere. The thing about data leaks is that it’s only a matter of time before there is one, which will result in the information of the user to be given to any hackers that decide to leak information for their own personal gain. Using open-source libraries would be great for a modernized requirement, along with testing in a sandbox environment, as these will help increase security and functionality of the website.

## Areas of Security

Since this is a web-based service, API’s are necessary as the web service uses RESTful API, which’ll help with the secure communications. Client/Server is another area to be considered. Cryptography is also another necessary area for encryption for the website.

## Manual Review

Checking the Dependency in the pom.xml file, I see the maven dependency check is currently out of date, being 5.3.0, while the current online version is 7.2.0. This can be a problem due to unknown breaches that can happen when working with previous versions.

## Static Testing

Since there are 73 vulnerabilities found, I’ll go through some of the important vulnerabilities. The first would be CVE-2013-1624, which doesn’t time side-channel attacks. CVE 2015-6644 had an internal bug that spilled personal information. CVE-2016-1000027 made it so Remote Code Execution could be performed for java deserialization of untrusted data. CVE 2022-22965 makes the application vulnerable to key binding with remote code execution. CVE 2021-44228 doesn’t protect against attacker-controlled LPAD and other JNDI endpoints. CVE 2021-45026 has a fixed address that wasn’t complete with the default configurations, leading to leaks of data. CVE 2020-1938 had AJP connections as a higher trust than HTTPS, allowing for remote code execution to be performed.

## Mitigation Plan

Looking through the testing and manual review reports, The best way to mitigate the vulnerabilities would be to update all of the dependencies and springwork to the current known version. A lot of these previous versions have been exploited, which seems to be able to be fixed if they’re updated to the latest version.