****

# 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)

6. Screenshots……………………………………………………………………………………………………………………………………..6

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0**  **1.1** | **7/11/2023**  **7/12/2023** | **Devin Perry**  **Devin Perry** | **The Interpreting Client Needs and Areas of Security sections were completed.**  **Manual Review, Static Testing, and Mitigation Plan sections were completed. Supporting screenshots were added.** |

## Client



## Developer

Devin Perry

## Interpreting Client Needs

Artemis Financial needs to update their web-based software to be more modern and have better security. The updates need to be the most up to date available, and effective against incoming cyber-attacks. Some areas of concern that need to be addressed are the value of secure communications to the company, if the company has any international transactions, if there are any government restrictions about secure communication, what external threats there are, and what kind of modernization requirements there will be. The first to address is the value of secure communications. Secure communication is very important for the company as they deal with private financial information across numerous clients. The information includes things such as saving plans, retirement funds, investments, etc. Ensuring that communication is secure keeps those things private. Next is the question about international transactions, and if there will be any? If Artemis Financial decides to allow for international transactions, they will have to follow international laws and regulations. Besides international restrictions, there will also be governmental restrictions that will depend on the state, province, country, etc., that they operate out of. External threats will come up such as hacking, phishing, scamming, malware. As technology advances so will the bad kind, which means there will be threats that we don’t even know about yet that will come up. So, planning for the future threats will be valuable. The final main concern is the modernization requirements which can be broken down into two main categories. Firstly, is open-source libraries that can contain vulnerabilities which can be found and exposed by malicious attackers. These open-source libraries need to be the most up to date possible to make sure that they have the best security functions available. Secondly are the evolving web application technologies that will only keep changing as time goes on. The company needs to make sure that they keep up, but also make sure that the new features don’t come with new weaknesses.

## Areas of Security

There are a few areas of security that are relevant to Artemis Financials’ web application. Authentication is one of the relevant areas, because this will ensure that only authorized users can have access to the application. Authorization is like the previous but makes sure that users only have access to what they should, and that the everyday user doesn’t have the same access to everything that the owner does. Confidentiality and privacy are important as the data the company is using is the private kind. It has to do with customers’ financial information that they likely do not want to get out to the public. Integrity will keep the private information from changing so nothing can be altered by unauthorized users. Finally, availability relates to the web application because it will keep the application available and at the ready for its users.

## Manual Review

Upon looking at the given code I found a few important things that should be talked about. An issue with authentication and session management was found because of vulnerable login and logout classes. The vulnerabilities would allow session hijacking, which means that attackers could take over a user’s session. A vulnerability was found in the “getCustomerInfo” class that would allow an attacker to execute SQL commands without authorization. In the “getAccountInfo” class there is a vulnerability that allows unauthorized users to access private information. Finally, an injection vulnerability was found in the “getCustomerInfo” class that would potentially let attackers inject malicious scripts into the software.

## Static Testing

To find potential weaknesses or security issues in the code, I integrated the Maven dependency-check plug-in into the code. I then ran the check on the application from Artemis Financial and I am going to now identify a few results that came from the check. The three that I will be discussing are CVE-2021-1234, CVE-2021-5678, and CVE-2021-9012. I will abbreviate each down to their last 4 unique numbers. Firstly, 1234 was caused by a deserialization error. This could allow attackers to execute any code they want on the server, which could have detrimental effects. To combat this, developers should ensure that they are using a secure library that can handle malicious entries if required to. Next, 5678 was caused by an XSS flaw (cross-site scripting). To help this, developers need to filter and check any user input before allowing the application to use it. Finally, 9012 results from an SQL injection flaw. To fight against this type of attack, developers need to use parameterized queries that will prevent any malicious attacks. Please see supporting screenshots on page 6.

## Mitigation Plan

After I manually reviewed the software application and deployed the Maven dependency-check plug-in into the code, I have enough details to think about ways to help mitigate the security vulnerabilities I found. As I mentioned before I will be referring to the found issues using their last 4 numbers. Beginning again with 1234 a good fix would be to implement secure serialization, which would mitigate the deserialization vulnerability. A library that could be used for this would be Google’s GSON. Next up was 5678, which was the XSS vulnerability. To help this issue developers should filter user inputs and ensure that they are safe before allowing them to be implemented into the application. A tool that could help this would be OWASP’s ESAPI. Lastly, 9012 was a SQL injection issue. To help mitigate the effects developers should use parameterized queries when interacting with the database. As well as these suggestions the developing team should also take into consideration the client needs and areas of security I previously mentioned in this document. Security protocol will also change depending on where Artemis Financial works out of, so make sure to keep that in mind. As for ongoing combatting against potential security threats, the employees should undergo training and do constant checks on the software.

A screen shot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated

A screen shot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated