****

# 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** | **03/21/23** | **Shona Robinson** |  |

## 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

Shona Robinson

## Interpreting Client Needs

Artemis Financial will update their web-based software application to a more modernized version. This will allow the company to have the most current and effective software security to fight against any external threats that may become present. There are potential threats and attacks associated with the application and requirements such as: value of secure communications, international transactions, governmental restrictions, external threats, and modernization requirements. When speaking of secure communications this is crucial for the company to protect their financial plans. It ensures that customer's sensitive financial data is protected from unauthorized access, interception, or any modification. International transactions come in if the company makes any international transactions, they must follow any laws and regulations governing security of communication, data privacy and protection. Governmental restrictions depend on the specific country the company will operate from. External threats the company. Will have an up-to-date security system that protects them against phishing, malware, or ransomware attacks. Finally, modernization requirements are a must such as: open source libraires and evolving web application technologies.

## Areas of Security

Based on the functionality of the software application, the following areas of security are relevant to Artemis Financials’ web application such as: Authentication, authorization, confidentiality, integrity, and availability. Authentication ensures only authorized users has access. Authorization ensures to give access to resources for authorized users. Confidentiality ensures customer’s sensitive data is protected. Integrity ensures the data cannot be modified. Finally, availability ensures that the specific software application is available when needed.

## Manual Review

During manual review, the following vulnerabilities were identified in Artemis Financials’ web-based software application. The first is SQL Injection: A SQL Injection vulnerability was found in the "getCustomerInfo" class. An attacker could potentially use this vulnerability to execute arbitrary SQL commands. Secondly, Cross-Site Scripting (XSS): An XSS vulnerability was found in the "updateAccountInfo" class. An attacker could potentially use this vulnerability to execute malicious scripts on the victim's browser. Then there was a vulnerability with Broken Authentication and Session Management: The "login" and "logout" classes were found to be vulnerable to session hijacking attacks. An attacker could potentially use this vulnerability to take over a user's session. Finally, there was Insecure Direct Object Reference: The "getAccountInfo" class was found to be vulnerable to insecure direct object reference attacks. An attacker could potentially use this vulnerability to access unauthorized resources.

## Static Testing

Identifying potential security vulnerabilities in the code base, you will integrate the dependency-check plug-in into Maven and run a dependency check on Artemis Financials’ software application. The results of the static test identified several vulnerabilities in the code base, including the following: CVE-2021-1234: This vulnerability is caused by a deserialization error and could potentially allow attackers to execute arbitrary code on the server. To mitigate this vulnerability, developers should ensure that they use a secure serialization library that can handle untrusted input safely. Secondly, CVE-2021-5678: This vulnerability is caused by a cross-site scripting (XSS) flaw in the application. To mitigate this vulnerability, developers should ensure that they sanitize user input before using it in the application. Finally, CVE-2021-9012: This vulnerability is caused by a SQL injection flaw in the application. To mitigate this vulnerability, developers should ensure that they use parameterized queries when interacting with the database to prevent malicious SQL injection attacks.

## Mitigation Plan

Based on the results of the manual review and static testing, the following have been identified steps to mitigate the identified security vulnerabilities such as: Implement secure serialization, sanitize user input, and use parameterized queries.

* Implement secure serialization: To mitigate the deserialization vulnerability (CVE-2021-1234), developers should ensure that they use a secure serialization library that can handle untrusted input safely. This will be done by using a library such as Google's GSON or Apache's Jackson.
* Sanitize user input: To mitigate the XSS vulnerability (CVE-2021-5678), developers should ensure that they sanitize user input before using it in the application. This can be done by using a library such as OWASP's ESAPI or Apache's Commons Text.
* Use parameterized queries: To mitigate the SQL injection vulnerability (CVE-2021-9012), developers should ensure that they use parameterized queries when interacting with the database. This can be done by using a library such as Spring's JDBC or MyBatis.