****

# 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 5](#_Toc32574613)

[4. Static Testing 5](#_Toc32574614)

[5. Mitigation Plan](#_Toc32574615) 6

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **9/19/2022** | **Jonathan Carmichael** | **For official use** |

## 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:

Jonathan Carmichael

## Interpreting Client Needs

*Review the scenario to determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Document your findings in your vulnerability assessment report.*

Artemis Financial is a consulting company that develops individualized financial plans for their customers. These financial plans include savings, retirement, investments, and insurance. For this project, my team and I at Global Rain are examining Artemis Financials’ web-based software application to assist in identifying any security vulnerabilities. With this, it is vital to adhere to secure information and communication between client and clientele. Given such classified information (SSNs, tax information, etc.), it’s necessary to maintain a high level of security.

There is no information provided on whether Artemis Financial is only based in the U.S., but it would be smart to assume that there are international transactions involved. In terms of governmental restrictions, this company needs to be weary of securing communications, so that there isn’t exposure to leaking trade secrets. Targeting client’s information is the largest external threat that is present. This data must be encrypted from outside sources. With this, it is important to have maintenance that is up to date, focusing on bug fixes and weak security threads. These will need to be “modernized” to the best of the company’s abilities.

## Areas of Security

*Use what you’ve learned in step 1 and refer to the Vulnerability Assessment Process Flow Diagram provided. Think about the functionality of the software application to identify which areas of security apply to Artemis Financials’ web application. Document your findings in your vulnerability assessment report and justify why each area is relevant to the software application.*

* Input Validation – Artemis Financial will require input validation to confirm who is the owner of what information being provided. This allows for stronger protection for individual users.
* APIs – API creation is necessary, because this program is running both internally and externally. Meaning, it’s important regarding data breaches and saving time when developing code.
* Cryptography – Implementing cryptography within this program is essential. Not only would it assist in making sure user information isn’t compromised, but if the company does involve international transactions, it would be good when dealing with varying world currencies as well.
* Code Error – Discovering coding errors helps in figuring out what areas of the API need to be fixed, if any. This allows for the company to worry less about their user information and private information being accessed and/or exposed.
* Code Quality – Making sure that a user doesn’t have access to private company information and visa-versa. It allows us to take control of access methods based on whoever is using their services.

## Manual Review

*Refer to the seven security areas outlined in the Vulnerability Assessment Process Flow Diagram. Use what you’ve learned in steps 1 and 2 to guide your manual review. Identify all vulnerabilities in the Project One Code Base, linked in Supporting Materials, by manually inspecting the code. Document your findings in your vulnerability assessment report.*

Applying the seven security areas outlined within our flow diagram, and after reviewing the code, I’ve analyzed all .java files as well as the POM.xml. Overall, the code quality is acceptable albeit a few missing components. In “GreetingController.java” I noticed a lack of input validation, as in a way to prove the individual accessing this information is said individual. This is necessary in confirming who is the owner of what information. There also seems to be an issue with error handling, as it seems it is not there at all, as far as I could see.

## Static Testing

*Integrate the dependency-check plug-in into Maven by following the instructions outlined in the Integrating the Maven Dependency-Check Plug-in tutorial provided in Supporting Materials. Run a dependency check on Artemis Financials’ software application to identify all security vulnerabilities in the code. Specifically, identify all vulnerabilities in the code base by analyzing results from running the code through a static test.*

Graphical user interface, text, application, Teams

Description automatically generated



After scanning “rest-service” within Eclipse, there were a total of 38 dependencies (20 unique) scanned, where 11 of those dependencies were determined to be vulnerable. Of those 11 dependencies, there were 83 varying vulnerabilities discovered.

## Mitigation Plan

*Interpret the results from the manual review and static testing report. Identify steps to mitigate the identified security vulnerabilities by creating an action list that documents how to fix each vulnerability in your vulnerability assessment report.*

* When going through the dependency check, there were multiple vulnerabilities that could potentially give hackers the ability to bypass input validation, potentially giving them control over user data. Developers may have put this here when handling controlled data in error messages. Adding input validation within “GreetingController.java” should be necessary. This is vital for preventing data breaches and preventing reducing massive amounts of unnecessary data to be passed through.
* When we use Apache, it’s important to consider the trust of incoming connections to Apache Tomcat. If connections become (or are already) available to attackers, they can be exploited in a variety of dangerous ways. Encoding a two-step verification process may assist in eliminating vulnerabilities.
* Adding support for specifying an SSL configuration (such as SmtpAppender) can help specify, utilize, and store key information (host name, SSN), making it much more difficult for hackers to gain access.