****

# 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** | **11.6.23** | **Suellen Greene** |  |

## Client



## Developer

Suellen Greene

## Interpreting Client Needs

Artemis Financial is a consulting company that develops individualized financial plans for their customers. Artemis Financial develops individualized financial plans for their customers that include savings, retirement, investments, and insurance. Artemis Financial wants to modernize their company with the most current and effective software security. They currently use a RESTful web application programming interface (API). As a developer at Global Rain, I am committed to providing a detailed vulnerability analysis for their company. I will detail all security vulnerabilities that could affect Artemis Financial.

Artemis Financial deals with sensitive data so secure communications are crucial to the company. Secure communications will ensure the integrity of data passed between the application and the user. A security breach of sensitive data would be detrimental to the company, therefore Artemis Financial will need to consider including input validation and access control. Input validation will ensure data entered by the customer is valid and access control will ensure both authorized and unauthorized users are limited to functionality they are permitted to use.

Artemis Financial makes international transactions, so there are governmental restrictions pertaining to secure communications that must be considered. “PCI-DSS is the payment card industry data security standard. If you are processing, storing, or using credit cards in your software, this standard applies to you.” (I*ron-Clad Java [Book]*, n.d.) Other government restrictions to consider are as follows:

* The European General Data Protection Regulation (EU-GDPR)
* United Kingdom General Data Protection Regulation (UK-GDPR)
* SOX
* Bank Secrecy Act (BSA)
* Gramm-Leach-Bliley Act (GLBA)

Government restrictions are mandatory. Severe fines could be imposed if Artemis Financial does not comply with these restrictions. For example, “Failure to comply with PCI DSS could result in fines ranging from $5,000 to $100,000 per month until compliance is achieved." (Kost, 2023)

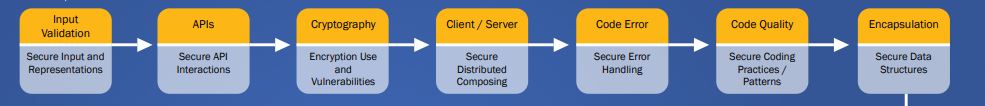
There are several external threats that Artemis Financial must consider. One such threat is clickjacking, “a form of attack that tricks a user into thinking he is clicking on one location when he is really clicking on another location that could cause harm.” (I*ron-Clad Java [Book]*, n.d.) Another external threat is privilege escalation. “This is when a (normally authenticated) attacker is able to manipulate input in a way that allows him to access features or data he should not be able to access.” (I*ron-Clad Java [Book]*, n.d.) Other threats, such as SQL injection and other injection threats will need to be considered as well.

Modernization requirements such as the role of open-source libraries are crucial to web development and will be beneficial to Artemis Finacial. Open-source libraries provide reusable code and save time by offering solutions for common tasks used in applications.

Web development is constantly changing for the better. Evolving web application technologies, such as new frameworks like serverless architecture, allow programmers to build applications without having the burden of managing the servers. Modern solutions like implementing the Maven Dependency-Check will be beneficial to the security of Artemis Finacial's project.

## Areas of Security

Image 1: Vulnerability Assessment Process Flow Diagram

 It is important to consider security threats at every level in the software development life cycle. There are several areas of security I will focus on as the developer. Regarding the first layer of the Vulnerability Assessment Process Flow Diagram, at a minimum, I would implement the following security measures:

* Input Validation
  + Input validation is important because it ensures that user entered data is safe. “Safe data handling means properly validating all sources of data that enter your web application and also properly encoding data when it exits the application to a web page, database, or other location.” (I*ron-Clad Java [Book]*, n.d.)
* APIs
  + APIs allow different software systems to communicate with each other. APIs help to improve efficiency and enable the integration of different systems and technologies. APIs also provide pre-built functions that developers can use that makes developing complex applications easier. Artemis Financial already uses an API and should continue to do so.
* Cryptography
  + Cryptography is important because it helps protect sensitive data from unauthorized access. Cryptography is a primary defense when handling the transfer of sensitive information.
* Code Quality
  + Code quality is always important. It ensures the code is readable, functional, and manageable. Good code helps to reduce bugs, improve performance, and makes the code easier to understand and modify.

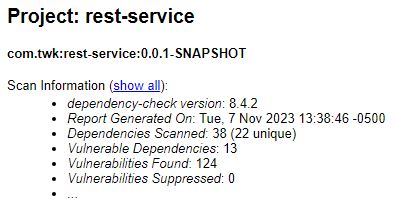
## Manual Review

After a manual review of the base code, I have discovered security threats. I will detail the threats below.

* No authentication exists for validating users
  + Authentication “ensures that the user is properly authorized for all actions (that is, the user has the right access control permissions).” (I*ron-Clad Java [Book]*, n.d.)
* In the Crud Controller Class, business names are sent as request parameters
  + Sensitive information can be leaked to unauthorized users when certain data is included in the request.
* Requests are not validated
  + One important rule of secure coding is to “not trust anything from the request.” (I*ron-Clad Java [Book]*, n.d.)
* HTTP POST is not used
  + The HTTP POST request is a secure way to transport sensitive data. “The following POST has three properties that allow for the secure transport of sensitive data: The request is an HTTPS request, the HTTPS verb is POST, and the sensitive data is in the body of the request instead of the URL.” (I*ron-Clad Java [Book]*, n.d.)

## Static Testing

I ran a dependency check on Artemis Finacial's software application to identify all security vulnerabilities in the code. A snapshot of the dependency check is below. Take note of the number of vulnerable dependencies found: 13. I will detail the 13 vulnerabilities found.



|  |  |  |
| --- | --- | --- |
| **Vulnerability ID** | **CVE Count** | **Description** |
| tomcat-embed-websocket-9.0.30.jar | 25 | Denial of service, information leaks, HTTP Request/Response Smuggling |
| tomcat-embed-core-9.0.30.jar | 24 | Connections can be exploited, Unresponsive servers, Denial of service, Information leaks, |
| spring-webmvc-5.2.3.RELEASE.jar | 11 | Remote code execution, Privilege escalation, Denial of service, Malicious input |
| spring-web-5.2.3.RELEASE.jar | 12 | Remote code execution, Privilege escalation, Denial of service, Malicious input |
| [spring-core-5.2.3.RELEASE.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l17_3734223040040e8c3fecd5faa3ae8a1ed6da146b) | 11 | Remote code execution, Privilege escalation, RFD attacks, Denial of service, |
| [spring-boot-starter-web-2.2.4.RELEASE.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l16_ec75d01d212b5229c16d872fb127744c0ed46ed8) | 3 | Security bypass, Directory hijacking, Denial of service, |
| [spring-boot-2.2.4.RELEASE.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l15_225a4fd31156c254e3bb92adb42ee8c6de812714) | 3 | Security bypass, Directory hacking, Denial of service |
| [snakeyaml-1.25.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l14_8b6e01ef661d8378ae6dd7b511a7f2a33fae1421) | 8 | Remote code execution, Denial of service, Stack overflow, |
| [jackson-databind-2.10.2.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l5_0528de95f198afafbcfb0c09d2e43b6e0ea663ec) | 6 | Data integrity, Denial of service, Resource exhaustion |
| [bcprov-jdk15on-1.46.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l1_991c96a4e31e6c19e2b9136c8955bd423f2dc4c7) | 18 | Improper verification of cryptographic signature, Time and state, Improper validation, |
| [logback-core-1.2.3.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l12_864344400c3d4d92dfeb0a305dc87d953677c03c) | 1 | Derealization of untrusted data |
| [hibernate-validator-6.0.18.Final.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l3_7fd00bcd87e14b6ba66279282ef15efa30dd2492) | 1 | Improper input validation |
| [log4j-api-2.12.1.jar](https://usc-word-edit.officeapps.live.com/we/wordeditorframe.aspx?ui=en%2DUS&rs=en%2DUS&wopisrc=https%3A%2F%2Fsnhu-my.sharepoint.com%2Fpersonal%2Fsuellen_greene_snhu_edu%2F_vti_bin%2Fwopi.ashx%2Ffiles%2F6d3cacc3c0be4c6d90950048a978db0c&wdenableroaming=1&mscc=1&wdodb=1&hid=733DEDA0-206F-4000-6860-D9C48CD7A24D&wdorigin=ItemsView&wdhostclicktime=1699761256076&jsapi=1&jsapiver=v1&newsession=1&corrid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&usid=cd35f6df-dfdc-4b2b-9988-eeac6ffbe78f&sftc=1&cac=1&mtf=1&sfp=1&instantedit=1&wopicomplete=1&wdredirectionreason=Unified_SingleFlush&rct=Normal&ctp=LeastProtected#l10_a55e6d987f50a515c9260b0451b4fa217dc539cb) | 1 | Improper certification validation |

## Mitigation Plan

The manual review of the code and the static testing of the code discovered many vulnerabilities. I have identified steps to mitigate the identified security vulnerabilities. I have created an action list to document how to fix these vulnerabilities.

1. Follow a secure software development lifecycle (SSDL)
2. Address vulnerabilities earlier in the lifecycle
3. For cross-site scripting defense, "JavaServer Pages Standard Tag Library, the OWASP Java Encoder for output encoding XSS defense, the OWASP Java HTML Sanitizer for untrusted HTML sanitization, and the Java JSON Sanitizer for outbound.” (I*ron-Clad Java [Book]*, n.d.)
4. For applied cryptography, “the Google Keyczar library is leaps above the rest in this space and can be used in conjunction with the open-source Bouncy Castle library for a more modern crypto provider.” (I*ron-Clad Java [Book]*, n.d.)
5. Use the OWASP dependency checker “will help you find publicly disclosed vulnerabilities in third-party code.” (I*ron-Clad Java [Book]*, n.d.)
6. Consider security at every level of the code

Sources

*Iron-Clad Java [Book]*. (n.d.). [Www.oreilly.com](http://www.oreilly.com/). <https://learning.oreilly.com/library/view/iron-clad-java/9780071835886/?sso_link=yes&sso_link_from=SNHU>

Kost, E. (2023, June 27). *Top 12 Cybersecurity Regulations for Financial Services | UpGuard*. Www.upguard.com. <https://www.upguard.com/blog/cybersecurity-regulations-financial-industry>

‌