****

# 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** | **07/10/2023** | **Kelly Perez** |  |

## 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

Kelly Perez

## Interpreting Client Needs

Secure communications secure sensitive data, build stakeholder confidence, and reduce cyberattack risk for Artemis Financial. Artemis Financial transacts internationally, making secure communications essential to protect financial data and transaction integrity. Artemis Financial must consider government constraints on secure communications. To protect the firm, external threats like hacking, data breaches, and phishing must be actively addressed in the ever-changing threat landscape. To stay ahead of growing dangers and ensure communication security, Artemis Financial must modernize, including using open-source frameworks and new web application technologies. Open-source libraries can help Artemis Financial secure communications. Open-source libraries offer cost-effective solutions and community assistance, but careful selection and regular upgrades are needed to address security concerns. Staying current with web application technologies is essential to integrate secure protocols, encryption, and authentication techniques to protect the company's communication systems from growing dangers.

## Areas of Security

Unsecured RESTful APIs can cause data breaches and theft. To reduce these risks, employ HTTPS and comprehensive user input error handling. Comprehensive verification and validation methods can assist the authenticity and integrity of API data, improving security.

## Manual Review

1. **DocData.java:** The database doesn’t have a secure connection. The code establishes a database connection without using secure practices such as encrypting credentials or using secure protocols.
2. **RestServiceApplication.java:** The code does not provide authentication or authorization, allowing uncontrolled REST service access.
3. **myDateTime.java:** The code exposes date and time representation accessor and mutator methods without input validation or sanitization, which could lead to security vulnerabilities.
4. **GreetingController.java:** The "name" parameter in the "/greeting" endpoint is not validated, which could lead to injection attacks or unexpected behavior.
5. **Greeting.java:** If untrusted data is used to instantiate the object, the function does not validate constructor inputs, which might cause unexpected behavior or security concerns.
6. **customer.java:** The code does not restrict access to sensitive functions like showInfo and deposit, allowing unauthorized access or change of account information.
7. **CRUDController.java:** The "business name" parameter in the "/read" endpoint is not validated, which could lead to attacks or unexpected behavior.
8. **CRUD.java:** The code has two constructors that initialize the same fields, which can lead to confusion and potential misuse of the class.

## Static Testing

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

**bcprov-jdk15on-1.46.jar: Many vulnerabilities listed under 1.55. Updating to latest version will help.**

CVE-2016-1000338

CVE-2016-1000342

CVE-2016-1000343

CVE-2016-1000344

CVE-2016-1000352

CVE-2023-33201

CVE-2016-1000341

CVE-2016-1000345

CVE-2017-13098

CVE-2020-15522

CVE-2020-0187

CVE-2016-1000339

CVE-2020-26939

CVE-2015-7940

CVE-2018-5382

CVE-2013-1624

CVE-2016-1000346

CVE-2015-6644

**hibernate-validator-6.0.18.Final.jar: Several listed vulnerabilities. Updating to latest version will help.**

CVE-2020-10693

**jackson-databind-2.10.2.jar: Several listed vulnerabilities. Updating to latest version will help.**

CVE-2020-25649

CVE-2020-36518

CVE-2021-46877

CVE-2022-42003

CVE-2022-42004

CVE-2023-35116

log4j-api-2.12.1.jar: **Several listed vulnerabilities. Updating to latest version will help.**

CVE-2020-9488

logback-core-1.2.3.jar: **Several listed vulnerabilities. Updating to latest version will help.**

CVE-2021-42550

**snakeyaml-1.25.jar: vulnerable to Denial of Service (DoS). Update to latest version.**

CVE-2022-1471

CVE-2017-18640

CVE-2022-25857

CVE-2022-38749

CVE-2022-38751

CVE-2022-41854

CVE-2022-38750

**spring-boot-2.2.4.RELEASE.jar: vulnerable to temporary directory hijacking. Update to latest version**

CVE-2022-27772

CVE-2023-20883

**spring-boot-starter-web-2.2.4.RELEASE.jar: vulnerable to temporary directory hijacking. Update to latest version**

CVE-2022-27772

CVE-2023-20883

**spring-core-5.2.3.RELEASE.jar: vulnerable to remote code execution (RCE) via data binding. Update to latest version.**

CVE-2022-22965

CVE-2021-22118

CVE-2020-5421

CVE-2022-22950

CVE-2022-22971

CVE-2023-20861

CVE-2023-20863

CVE-2022-22968

CVE-2022-22970

CVE-2021-22060

**spring-web-5.2.3.RELEASE.jar: remote code execution (RCE) issue if used for Java deserialization of untrusted data. Updating to latest version will help.**

CVE-2016-1000027

CVE-2022-22965

CVE-2021-22118

CVE-2020-5421

CVE-2022-22950

CVE-2022-22971

CVE-2023-20861

CVE-2023-20863

CVE-2022-22968

CVE-2022-22970

CVE-2021-22060

CVE-2021-22096

## Mitigation Plan

HTTPS (Hypertext Transfer Protocol Secure) is safer than HTTP. HTTPS encrypts and authenticates client-server data, preventing interception. Instead of the URI, add sensitive information like user credentials or API keys in queries. This protects server logs and other sensitive data. Token-based authentication and OAuth increase application security. It restricts resources and actions to approved users.

To stay current on security updates and bug fixes, you must update all dependencies, including libraries and frameworks. Attackers may abuse outdated dependencies.

Avoid hard-coding sensitive data like usernames, passwords, and API credentials into source code. Instead, use secure storage, environment variables, or configuration files. Hard-coded data is vulnerable to hackers.

These techniques improve application security, protect user data, and reduce vulnerabilities.