****

# 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** | **11/12/23** | **Pravishna Nand** | **Analyze how advanced security concepts are applied to develop secure code** |

## Client



## Developer

Pravishna Nand

## Interpreting Client Needs

* In the process of interpreting Artemis Financials client needs for enhanced software security, it is crucial to delve deep into the scenario and assess potential threats and attacks that may target their web-based application. Firstly, we must evaluate the value of secure communications to the company. Given that Artemis Financial deals with sensitive financial information, secure communications are of paramount importance to protect client data and maintain trust. Additionally, understanding whether the company engages in international transactions is vital, as this introduces a layer of complexity, including compliance with varying international regulations. Furthermore, considering governmental restrictions on secure communications is essential, as adherence to such regulations is imperative for legal and operational reasons. To comprehensively assess the security landscape, we need to identify both existing and foreseeable external threats, enabling proactive defense strategies. Lastly, modernization requirements, such as the role of open-source libraries and evolving web application technologies, should be considered to ensure the software remains resilient in the face of evolving threats and vulnerabilities. These considerations will form the foundation of our vulnerability assessment report, guiding our efforts to bolster Artemis Financials software security.

## Areas of Security

* **Input Validation**:
  + Justification: Input validation is critical for preventing common vulnerabilities like injection attacks (e.g., SQL injection, cross-site scripting). Artemis Financials application likely receives and processes user input and external data, making input validation essential to mitigate these risks.
* **Secure API Interactions**:
  + Justification: Secure API interactions are relevant because Artemis Financial may communicate with external services or APIs to gather financial data or perform transactions. Ensuring the security of these interactions is crucial to protect sensitive financial information.
* **Encryption Use and Vulnerabilities**:
  + Justification: Encryption is vital for safeguarding confidential financial data such as customer information, transactions, and sensitive documents. Vulnerabilities in encryption could lead to data breaches, making this area highly relevant.
* **Secure Distributed Composing**:
  + Justification: Secure distributed composing is important because modern web applications often rely on distributed systems and services. Ensuring secure communication and data exchange between components is crucial to prevent data breaches and unauthorized access.
* **Secure Code Handling**:
  + Justification: Secure code handling is relevant as improper code handling can lead to unintended vulnerabilities or software errors. Ensuring that the application handles code securely helps maintain its integrity and reliability.
* **Secure Coding Practices/Patterns**:
  + Justification: Secure coding practices and patterns are fundamental to overall code quality and security. Adhering to these practices reduces the risk of common security issues like injection attacks, making it essential for Artemis Financials application.
* **Secure Data Structures**:
  + Justification: Secure data structures are crucial for maintaining data integrity and confidentiality, especially in financial applications where data accuracy is paramount. Vulnerabilities in data structures could lead to data manipulation or exposure.

## Manual Review

* **Outdated Spring Boot Version:**
  + **Security Concern:** The use of Spring Boot version 2.2.4 is outdated, which poses a potential security risk. The latest release, v3.2.0, offers enhanced security features and updates.
  + **Recommendation:** To bolster security and take advantage of the latest protective measures, it is advisable to update to the most recent Spring Boot version, v3.2.0.
* **Input Validation:**
  + **Security Concern:** The "greeting" endpoint lacks adequate input validation logic or sanitization, making it vulnerable to potential injection or scripting attacks.
  + **Recommendation:** To safeguard against security threats like SQL injection and cross-site scripting (XSS) attacks, it is imperative to implement robust input validation and sanitization procedures for user inputs.
* **Data Access:**
  + **Security Concern:** Within the read document function, the code employs hard-coded credentials (root and root) to connect to a SQL database, posing a security hazard.
  + **Recommendation:** Ensure the secure management of database credentials by utilizing secure methods such as environment variables or a reliable configuration management system. Avoid embedding credentials directly into the source code.
* **Authentication and Authorization:**
  + **Security Concern:** None of the API endpoints incorporate authentication or access control mechanisms, allowing unrestricted access to all users.
  + **Recommendation:** Strengthen security measures by implementing authentication and authorization protocols that align with user roles and privileges. Consider leveraging established frameworks like Spring Security for robust access control.
* **Code Errors:**
  + **Security Concern:** Errors and exceptions are inadequately managed, potentially exposing sensitive information to malicious actors through detailed stack traces.
  + **Recommendation:** Elevate security standards by implementing effective error-handling practices. Ensure that error messages do not inadvertently disclose sensitive data and adhere to suitable log levels. Additionally, consider log sanitization practices for an extra layer of defense.

## Static Testing

* **bcprov-jdk15on-1.46.jar:**
  + **Vulnerability IDs:** CVE-2013-1624, CVE-2015-6644, CVE-2015-7940, CVE-2016-1000338, CVE-2016-1000339, CVE-2016-1000341, CVE-2016-1000342, CVE-2016-1000343, CVE-2016-1000344, CVE-2016-1000345, CVE-2016-1000346, CVE-2016-1000352, CVE-2017-13098, CVE-2018-5382, CVE-2020-0187, CVE-2020-26939, CVE-2023-33201
  + **Highest Severity:** HIGH
  + **Description:** This library contains numerous vulnerabilities, including CVEs with high severity. These vulnerabilities could lead to security breaches if not addressed.
  + **Recommendation:** Update the bcprov-jdk15on library to the latest secure version to mitigate these vulnerabilities.
* **spring-boot-2.2.4.RELEASE.jar:**
  + **Vulnerability IDs:** CVE-2022-27772, CVE-2023-20873, CVE-2023-20883
  + **Highest Severity:** CRITICAL
  + **Description:** The Spring Boot library contains critical vulnerabilities that could potentially be exploited by attackers.
  + **Recommendation:** Upgrade to the latest version of Spring Boot (v3.2.0) to address these critical vulnerabilities and benefit from security enhancements.
* **logback-core-1.2.3.jar:**
  + **Vulnerability IDs:** CVE-2021-42550
  + **Highest Severity:** MEDIUM
  + **Description:** This library has a medium-severity vulnerability that should be addressed.
  + **Recommendation:** Update the logback-core library to the latest version to fix this vulnerability.
* **log4j-api-2.12.1.jar:**
  + **Vulnerability IDs:** CVE-2020-9488, CVE-2021-44228, CVE-2021-44832, CVE-2021-45046, CVE-2021-45105
  + **Highest Severity:** CRITICAL
  + **Description:** The log4j-api library contains critical vulnerabilities that should be addressed urgently.
  + **Recommendation:** Update the log4j-api library to the latest secure version to mitigate these critical vulnerabilities.
* **snakeyaml-1.25.jar:**
  + **Vulnerability IDs:** CVE-2017-18640, CVE-2021-4235, CVE-2022-1471, CVE-2022-25857, CVE-2022-3064, CVE-2022-38749, CVE-2022-38750, CVE-2022-38751, CVE-2022-38752, CVE-2022-41854
  + **Highest Severity:** CRITICAL
  + **Description:** The snakeyaml library contains critical vulnerabilities, including CVEs with high severity.
  + **Recommendation:** Update the snakeyaml library to the latest secure version to mitigate these critical vulnerabilities.
* **jackson-databind-2.10.2.jar:**
  + **Vulnerability IDs:** CVE-2020-25649, CVE-2020-36518, CVE-2021-46877, CVE-2022-42003, CVE-2022-42004, CVE-2023-35116
  + **Highest Severity:** HIGH
  + **Description:** The jackson-databind library contains high-severity vulnerabilities.
  + **Recommendation:** Update the jackson-databind library to the latest secure version to mitigate these high-severity vulnerabilities.
* **tomcat-embed-core-9.0.30.jar:**
  + **Vulnerability IDs:** CVE-2019-17569, CVE-2020-11996, CVE-2020-13934, CVE-2020-13935, CVE-2020-13943, CVE-2020-17527, CVE-2020-1935, CVE-2020-1938, CVE-2020-8022, CVE-2020-9484, CVE-2021-24122, CVE-2021-25122, CVE-2021-25329, CVE-2021-30640, CVE-2021-33037, CVE-2021-41079, CVE-2021-43980, CVE-2022-29885, CVE-2022-34305, CVE-2022-42252, CVE-2023-28708, CVE-2023-41080, CVE-2023-42795, CVE-2023-44487, CVE-2023-45648
  + **Highest Severity:** CRITICAL
  + **Description:** The tomcat-embed-core library contains numerous critical vulnerabilities that should be addressed urgently.
  + **Recommendation:** Update the tomcat-embed-core library to the latest secure version to mitigate these critical vulnerabilities.
* **hibernate-validator-6.0.18.Final.jar:**
  + **Vulnerability IDs:** CVE-2020-10693
  + **Highest Severity:** MEDIUM
  + **Description:** The hibernate-validator library has a medium-severity vulnerability that should be addressed.
  + **Recommendation:** Update the hibernate-validator library to the latest version to fix this vulnerability.
* **spring-web-5.2.3.RELEASE.jar:**
  + **Vulnerability IDs:** CVE-2016-1000027, CVE-2020-5421, CVE-2021-22096, CVE-2021-22118
  + **Highest Severity:** HIGH
  + **Description:** The spring-web library contains high-severity vulnerabilities.
  + **Recommendation:** Update the spring-web library to the latest secure version to mitigate these high-severity vulnerabilities.
* **spring-beans-5.2.3.RELEASE.jar:**
  + **Vulnerability IDs:** CVE-2022-22965
  + **Highest Severity:** HIGH
  + **Description:** The spring-beans library has a high-severity vulnerability that should be addressed.
  + **Recommendation:** Update the spring-beans library to the latest secure version to mitigate this high-severity vulnerability.
* **spring-webmvc-5.2.3.RELEASE.jar:**
  + **Vulnerability IDs:** CVE-2021-22060
  + **Highest Severity:** MEDIUM
  + **Description:** The spring-webmvc library contains a medium-severity vulnerability.
  + **Recommendation:** Update the spring-webmvc library to the latest version to fix this vulnerability.
* **spring-context-5.2.3.RELEASE.jar:**
  + **Vulnerability IDs:** CVE-2022-22968
  + **Highest Severity:** MEDIUM
  + **Description:** The spring-context library has a medium-severity vulnerability.
  + **Recommendation:** Update the spring-context library to the latest version to address this medium-severity vulnerability.
* **spring-expression-5.2.3.RELEASE.jar:**
  + **Vulnerability IDs:** CVE-2022-22950, CVE-2023-20861, CVE-2023-20863
  + **Highest Severity:** MEDIUM
  + **Description:** The spring-expression library contains medium-severity vulnerabilities.
  + **Recommendation:** Update the spring-expression library to the latest secure version to mitigate these medium-severity vulnerabilities.

## Mitigation Plan

* The mitigation plan for Artemis Financials software application involves a systematic approach to addressing the identified security vulnerabilities from both the manual review and static testing reports. Each vulnerability has been carefully analyzed, and an action list has been compiled to guide the mitigation process. For instance, in the case of the outdated Spring Boot version, the plan recommends upgrading to the latest version (v3.2.0) to benefit from security improvements. Input validation issues in the "greeting" endpoint necessitate implementing input validation and sanitization mechanisms to prevent injection and scripting attacks. Storing database credentials directly in the code in the "read\_document" function is a security risk, so the plan suggests securely storing credentials using environment variables or a secure configuration management system. Authentication and authorization gaps in API controllers require implementing access control mechanisms like Spring Security. Additionally, for code error handling, the plan advises the implementation of secure error handling practices to avoid exposing sensitive information in error messages. Each action item is outlined in the vulnerability assessment report to guide the mitigation process comprehensively, ensuring that Artemis Financials software becomes more secure and resilient against potential threats.