****

# Practices for Secure Software Report

**Table of Contents**

[**Document Revision History 3**](#_heading=h.30j0zll)

[**Client 3**](#_heading=h.1fob9te)

[**Instructions 3**](#_heading=h.3znysh7)

[**Developer 4**](#_heading=h.2et92p0)

[**1. Algorithm Cipher 4**](#_heading=h.tyjcwt)

[**2. Certificate Generation 4**](#_heading=h.3dy6vkm)

[**3. Deploy Cipher 4**](#_heading=h.1t3h5sf)

[**4. Secure Communications 4**](#_heading=h.4d34og8)

[**5. Secondary Testing 4**](#_heading=h.2s8eyo1)

[**6. Functional Testing 4**](#_heading=h.17dp8vu)

[**7. Summary 4**](#_heading=h.3rdcrjn)

[**8. Industry Standard Best Practices 4**](#_heading=h.26in1rg)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **2/23/2025** | **Jacob Wisniewski** | **Initial release** |

## Client



## Instructions

Submit this completed practices for secure software report. Replace the bracketed text with the relevant information. You must document your process for writing secure communications and refactoring code that complies with software security testing protocols.

* Respond to the 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 Two Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Jacob Wisniewski

## Algorithm Cipher

AES-256 is the recommended encryption algorithm for securing sensitive financial data at Artemis Financial. As a symmetric block cipher, AES-256 provides a high level of security, making it a great choice for entities working with sensitive data. With its 256-bit key length AES-256 offers resistance against brute-force attacks and ensures long-term data confidentiality. AES was standardized by the NIST in 2001, replacing older encryption algorithms such as DES and 3DES, which have been deprecated due to their vulnerability to modern cryptanalysis. AES-256 remains the industry standard for secure encryption, and would be a fantastic choice for Artemis Financial's security needs.

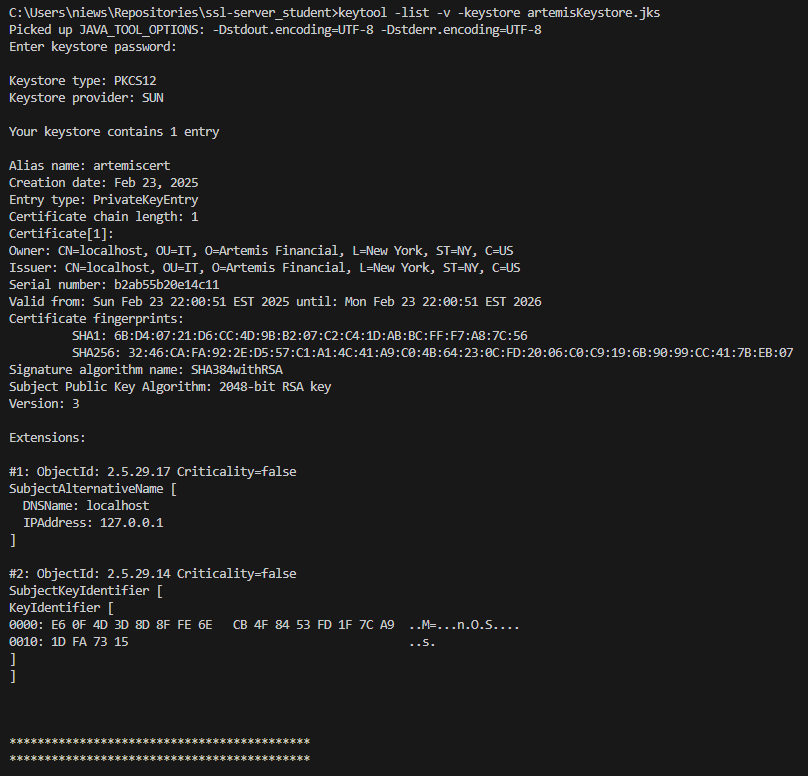
AES-256 operates on 128-bit blocks and processes data through 14 rounds of encryption, ensuring that plaintext is transformed into ciphertext through a combination of substitution, permutation, and key expansion techniques. These processes make AES resistant to newer cryptanalysis attacks. A key aspect of AES-256 encryption is the use of hash functions to ensure data integrity. While AES itself is not a hashing algorithm it is often combined with SHA-256 to create cryptographic hashes that verify data integrity and prevent unauthorized modifications. This pairing strengthens Artemis Financial's security by ensuring that encrypted financial data remains both confidential and tamper-proof.

A crucial consideration in AES encryption is the management of random numbers and key distribution. To prevent security vulnerabilities, cryptographically secure random number generators must be used to generate unpredictable values for cryptographic operations. Proper key management is equally important in preventing unauthorized access. AES-256 is a symmetric encryption algorithm that uses the same key for both encryption and decryption, requiring a secure mechanism for key distribution. To achieve this RSA is often used to securely exchange AES keys between parties, ensuring that only authorized entities can access encrypted data.

The current state of encryption includes AES-256 as the industry standard for securing sensitive data in transit and at rest. By implementing AES-256 encryption, Artemis Financial ensures compliance with regulatory standards such as PCI-DSS, GDPR, and HIPAA, while maintaining the confidentiality and integrity of its financial transactions.

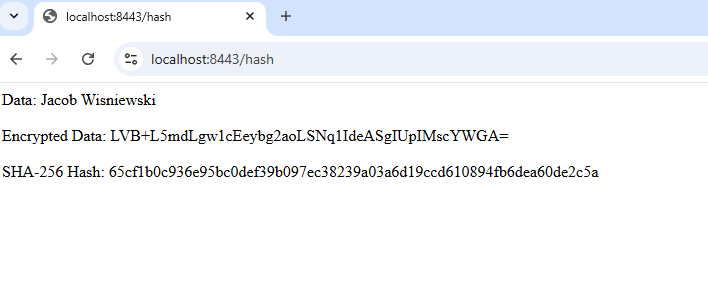
## Certificate Generation

Insert a screenshot below of the CER file.



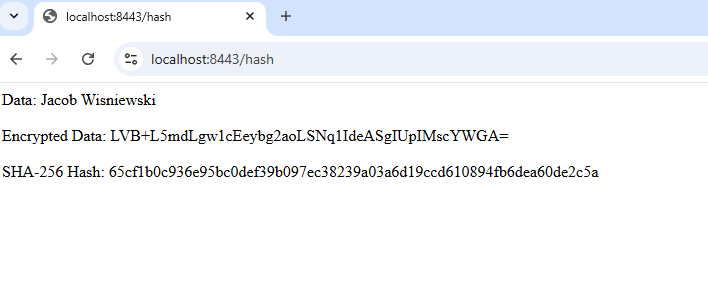
## Deploy Cipher

Insert a screenshot below of the checksum verification.



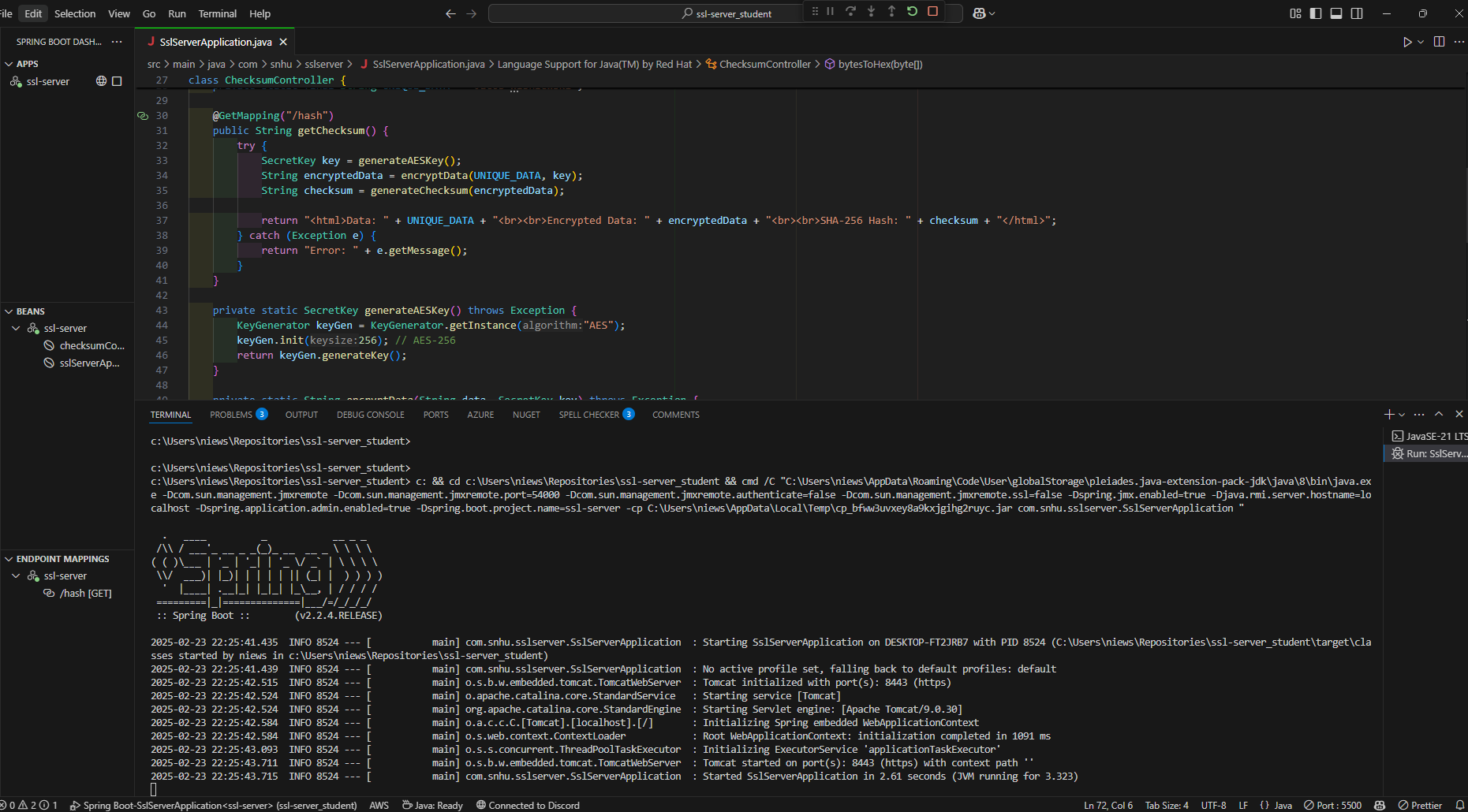
## Secure Communications

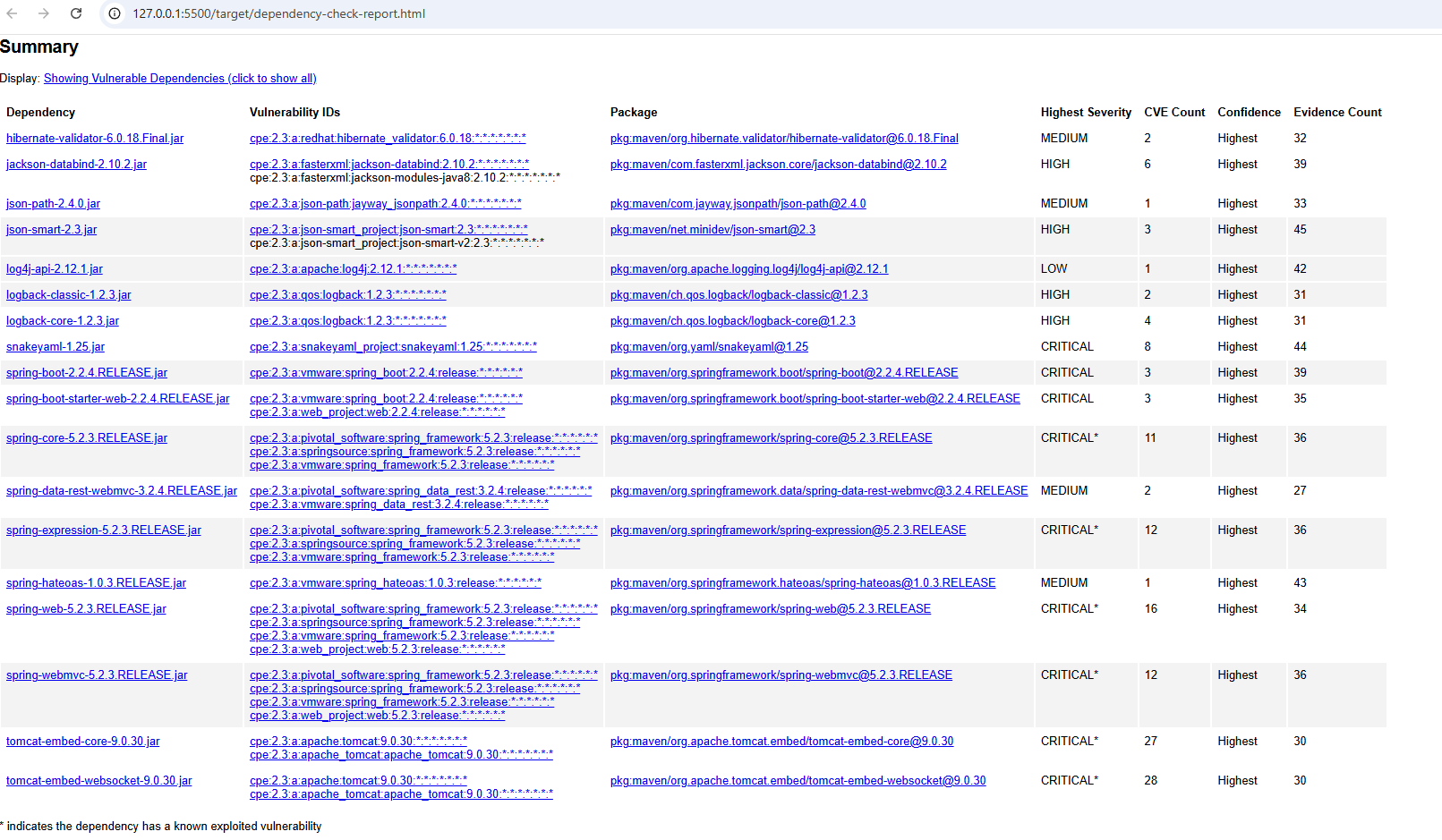
Insert a screenshot below of the web browser that shows a secure webpage.



## Secondary Testing

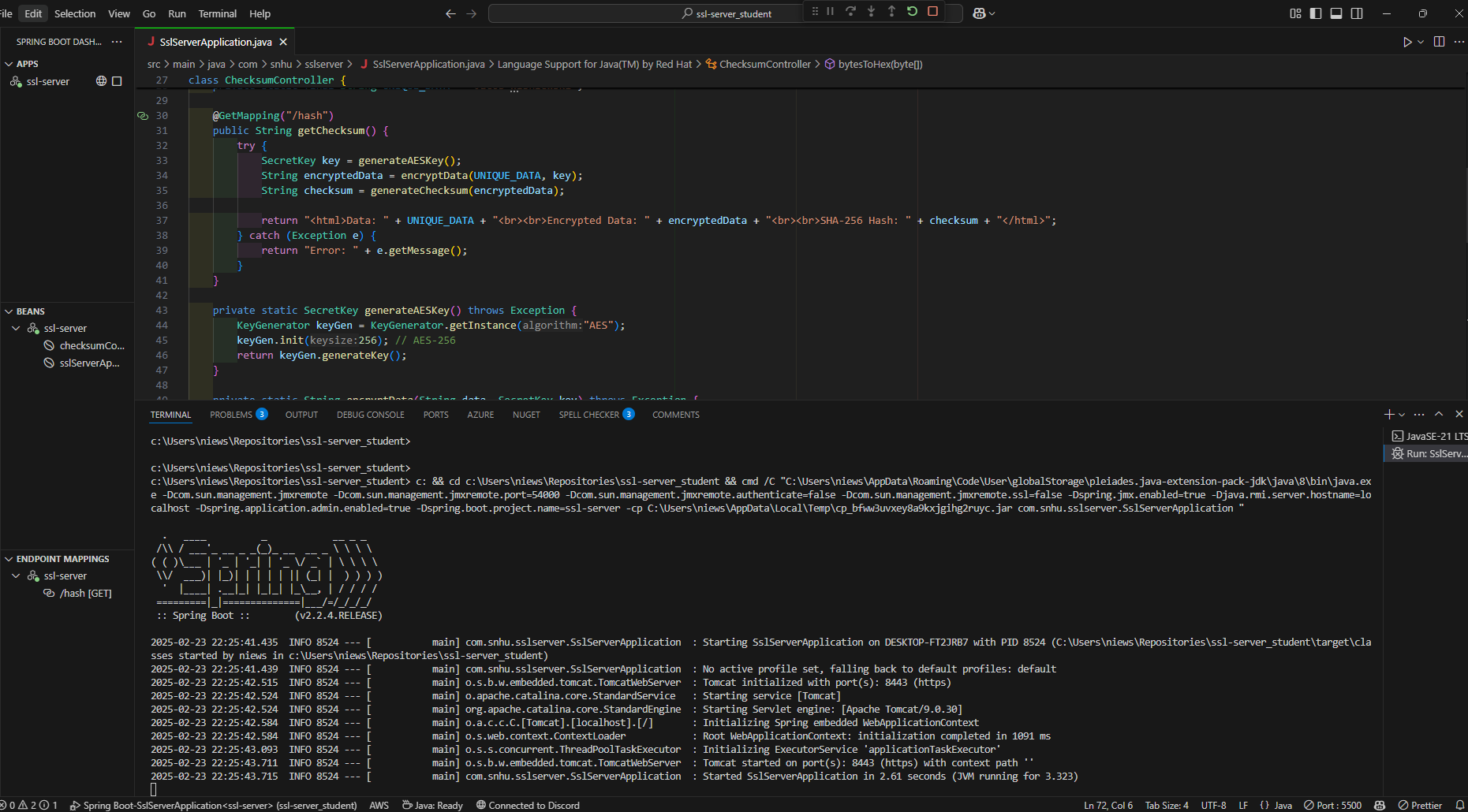
Insert screenshots below of the refactored code executed without errors and the dependency-check report.





## Functional Testing

Insert a screenshot below of the refactored code executed without errors.



## Summary

The code has been refactored to enhance security and ensure compliance with industry-standard security testing protocols. By following the Vulnerability Assessment Process Flow Diagram, key areas of security were addressed, including cryptographic security, secure client-server communication, and data integrity verification. The main focus was on strengthening the encryption mechanism using AES-256, implementing checksum verification, and enabling HTTPS for secure communication. These enhancements mitigate risks related to data tampering, unauthorized access, and transmission security.

To add layers of security, the application now integrates checksum verification to ensure data integrity during transmission, preventing unauthorized modifications. The use of self-signed SSL certificates enables encrypted communication between the client and the server, reducing the risk of man-in-the-middle attacks. We also implemented secure key management practices to ensure the confidentiality of encryption keys, which prevents unauthorized decryption.

## Industry Standard Best Practices

Industry-standard best practices were applied throughout the refactoring process to ensure that the software remains secure and resilient against known vulnerabilities. Secure coding principles from the OWASP Secure Coding Practices Guide were followed, including proper encryption techniques, secure key management, and HTTPS enforcement. The AES-256 encryption implementation follows NIST recommendations, ensuring strong encryption methods to protect financial data.

Maintaining security in the software application required proactive security testing using static code analysis and dependency checks to identify potential vulnerabilities. By ensuring that sensitive data is always encrypted at rest and in transit, the application remains compliant with regulatory requirements such as PCI-DSS, GDPR, and HIPAA.

Applying industry standard best practices in secure coding is vital to the overall wellbeing of Artemis Financial. These practices reduce the risk of data breaches, financial fraud, and compliance violations. This protects both the company’s reputation and its clients’ sensitive financial information. Secure coding not only enhances trust and reliability in the software application, but also strengthens the company’s ability to adapt to evolving cybersecurity threats. By prioritizing security in the development process, Artemis Financial ensures long-term sustainability, regulatory compliance, and customer confidence in its financial services.