****

# Practices for Secure Software Report

Table of Contents

[Document Revision History 3](#_Toc102040754)

[Client 3](#_Toc102040755)

[Instructions 3](#_Toc102040756)

[Developer 4](#_Toc102040757)

[1. Algorithm Cipher 4](#_Toc102040758)

[2. Certificate Generation 4](#_Toc102040759)

[3. Deploy Cipher 4](#_Toc102040760)

[4. Secure Communications 4](#_Toc102040761)

[5. Secondary Testing 4](#_Toc102040762)

[6. Functional Testing 4](#_Toc102040763)

[7. Summary 4](#_Toc102040764)

[8. Industry Standard Best Practices 4](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **22 Jun 2025** | **Jordan Purdy** | **Final Submission** |

## Client



## Developer

Jordan Purdy

## Algorithm Cipher

For secure data transfer, I will recommend using AES (Advanced Encryption Standard) with SHA-256 for checksum verification. AES is a symmetric block cipher that is widely adopted and recommended by NIST. AES encrypts data in 128-bit blocks and can use key sizes of 128, 192, or 256 bits. It is highly efficient in both software and hardware and provides a strong level of security. Using SHA-256 will generate a 256-bit hash that is collision resistant and widely used in the industry for data integrity verification. It will require a secret key, which can be generated securely using a cryptographic random number generator. Being symmetric, the same key will be used for encryption and decryption. AES replaces the outdated DES algorithm, as it has withstood years of cryptoanalysis and remains the standard for secure encryption worldwide.

## Certificate Generation

A screenshot of a certificate

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

## Deploy Cipher

A screenshot of a computer

AI-generated content may be incorrect.

## Secure Communications

A screenshot of a computer

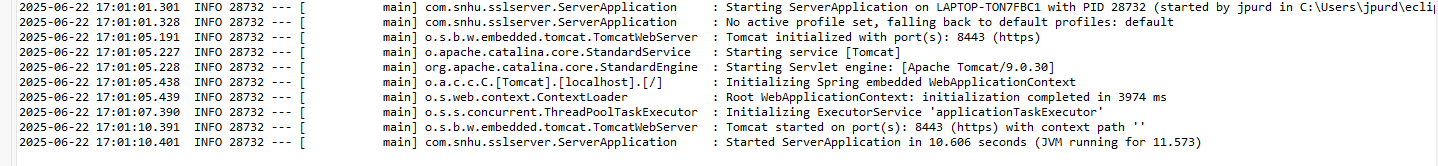
AI-generated content may be incorrect.

## Secondary Testing

## A white background with blue text AI-generated content may be incorrect.

## A screenshot of a computer AI-generated content may be incorrect.

## Functional Testing



## Summary

To meet Artemis Financial’s security requirements I implemented AES-based cryptographic hashing using SHA-256 for secure checksums, generated a self-signed certificate using Java Keytool, converted the application from HTTP to HTTPS, verified secure communication and deployed functional REST endpoints, ran OWASP dependency check to confirm no **new** vulnerabilities were introduced, and performed manual and functional testing to validate code quality and security compliance. I ensured code was enhanced at the stages of certificate validation, secure hashing, and transport security.

## Industry Standard Best Practices

To maintain secure coding standards, I used cryptographic APIs from Java’s standard library to ensure well-tested security routines, HTTPS enforcement to prevent man in the middle attacks, checksum validation to ensure integrity and non-repudiation, OWASP dependency check to identify vulnerable dependencies, and manual code review to identify logical or syntactical issues. These are best practices that not only protect the software from known threats but also enhance Artemis’ ability to safeguard sensitive financial information and ensure regulatory compliance.