****

# 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** | **[10/19/2025]** | **Nikki Yanez** |  |

## 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

Nikki Yanez

## Algorithm Cipher

For this project, I used the SHA-256 cipher for checksum verification. SHA-256 is a one-way cryptographic hash function that creates a unique, fixed-length 256-bit hash value for any input.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a certificate

AI-generated content may be incorrect.

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

AI-generated content may be incorrect.

## Secure Communications

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

A screenshot of a computer error

AI-generated content may be incorrect.

## Secondary Testing

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

A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Functional Testing

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

A screenshot of a computer screen

AI-generated content may be incorrect.

A computer screen shot of a program code

AI-generated content may be incorrect.

A computer screen shot of text

AI-generated content may be incorrect.

## Summary

In this project, I refactored Artemis Financial’s application to add encryption and secure communication features. I implemented a SHA-256 hashing algorithm to create a checksum verification that ensures data integrity during transmission. A self-signed SSL certificate was generated and integrated into the project to enable HTTPS communication, protecting data transferred between the client and the server. I also conducted static testing using the OWASP Dependency-Check tool and created a suppression file to hide false positives while maintaining awareness of potential vulnerabilities. These steps confirmed that the application runs securely, uses encryption for communication, and complies with software security testing protocols.

## Industry Standard Best Practices

Throughout this project, I followed industry standard best practices for secure coding and software maintenance. I used encryption algorithms such as SHA-256 and HTTPS protocols to ensure confidentiality and integrity of data in transit. Regular dependency scanning with OWASP tools was implemented to identify and manage vulnerabilities in third party libraries. I also maintained proper key and certificate management through Java Keytool and avoided hard coding sensitive information. By following these practices, Artemis Financial’s application aligns with DevSecOps principles, reduces potential attack surfaces, and improves overall system reliability and trustworthiness.