****

# 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** | **Dec 14, 2024** | **Seema Chavesta** |  |

## 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

Seema Chavesta

## Algorithm Cipher

For this project, we have chosen **AES (Advanced Encryption Standard)** as the encryption algorithm due to its strength and efficiency in securing sensitive data. AES is a symmetric encryption algorithm, meaning it uses the same key for both encryption and decryption. It supports key sizes of 128, 192, or 256 bits, with 256-bit keys providing the highest level of security. AES is widely used in industries such as finance and healthcare for protecting data at rest and in transit. Additionally, AES is fast and suitable for large data encryption, making it ideal for Artemis Financial’s application. By using AES, we address the need for strong data protection, ensuring confidentiality and integrity.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

[Insert screenshots here.]

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

Description automatically generated

## Secure Communications

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

A screenshot of a computer

Description automatically generated

[Insert screenshots here.]

## 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.

A screen shot of a computer

Description automatically generated

## Summary

The completion of this project marks a significant step in understanding secure software development and implementing best practices in coding, dependency management, and testing. By integrating the Maven dependency-check plugin, I successfully identified and addressed vulnerabilities within the project dependencies. This not only improved the overall security posture of the application but also ensured compliance with modern software standards.

The project involved creating a functional SHA-256 checksum generator that validates data integrity effectively. The refactored code was executed without errors, as demonstrated in the functional testing stage. Along the way, I resolved challenges such as dependency updates, ensuring compatibility, and mitigating risks posed by outdated libraries.

Overall, this exercise enhanced my skills in secure coding, dependency management, and application testing while reinforcing the importance of maintaining updated and secure libraries. It also emphasized the value of functional testing to ensure that code runs smoothly in production-like environments.

## Industry Standard Best Practices

In this project, I adhered to industry-standard best practices to ensure secure, efficient, and maintainable development. Secure dependency management was achieved using the Maven dependency-check plugin to identify and update vulnerable libraries, aligning with OWASP guidelines. The code was refactored for improved readability, maintainability, and performance, following clean code principles. Functional testing validated that the refactored SHA-256 checksum generator worked as expected, ensuring reliability. Version control and clear documentation were maintained for traceability, while compliance with security standards, such as addressing CVEs, reduced potential risks. By emphasizing continuous improvement through testing and iterative refinement, the project meets modern software quality and security standards, making it robust and scalable for real-world use.