****

# 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** | **12/13/2024** | **Foster Hare** |  |

## 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

Foster Hare

## Algorithm Cipher

SHA-256 is a strong and reliable hashing algorithm that is ideal for this project because it provides a high level of security for verifying data integrity. It produces a fixed 256-bit hash value, making it nearly impossible for two different inputs to produce the same output (collision resistance). This ensures that any changes to the data, intentional or accidental, can be easily detected. SHA-256 is also resistant to common attacks like pre-image attacks, making it a safe choice for financial applications where data integrity and security are critical.

In addition to its security, SHA-256 is widely used and supported in modern systems, making it easy to integrate into this project. Compared to older algorithms like MD5 or SHA-1, SHA-256 is much more secure and trusted for handling sensitive data. It is efficient, dependable, and ensures that data integrity checks, like verifying files or communication, are done accurately and securely. For these reasons, SHA-256 is the best option for Artemis Financial's needs.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a computer

Description automatically generated

## 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 error

Description automatically generated

## Secondary Testing

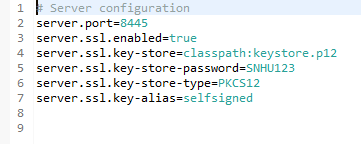
Insert screenshots below of the refactored code executed without errors and the dependency-check report.

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated



A screenshot of a computer program

Description automatically generated

## Functional Testing

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

A screenshot of a computer program

Description automatically generated

## Summary

This code uses AES-256 encryption and SHA-256 hashing to secure data and verify its integrity. The @SpringBootApplication annotation sets up the Spring Boot application and makes it easy to run. The generateAESKey() method creates a secure 256-bit AES key, and the encryptDataAES() method encrypts a sample string using this key and outputs it in Base64 format. The generateChecksum() method creates a SHA-256 hash of the data, which works as a checksum to check for any changes. The main method ties everything together by generating the encryption key, encrypting the data, creating the checksum, and printing both results. If any errors occur during encryption, key creation, or hashing, they are caught and displayed.

## Industry Standard Best Practices

Industry standard best practices for secure software development focus on ensuring the confidentiality, integrity, and availability of systems and data. These practices include implementing strong encryption methods, such as AES-256 for data encryption and SHA-256 for hashing, to protect sensitive information and verify data integrity. Regularly updating libraries, dependencies, and frameworks helps mitigate known vulnerabilities by applying security patches. Dependency scanning tools like OWASP Dependency Check are used to identify and address outdated or vulnerable components.

Additional best practices include secure coding techniques to prevent common threats like SQL injection, cross-site scripting (XSS), and buffer overflows. Proper exception handling and input validation ensure that user data is processed safely, reducing the risk of malicious input. Secure communication protocols, such as HTTPS with TLS, are enforced to protect data transmitted over the network. Logging and monitoring mechanisms help detect and respond to suspicious activities. By following these practices, software applications meet industry security standards, improve resilience to cyber threats, and ensure data protection for users.

**Sources**

*How to Create a Self Signed Certificate using Java Keytool*. (n.d.). Www.sslshopper.com. https://www.sslshopper.com/article-how-to-create-a-self-signed-certificate-using-java-keytool.html

‌

*Java Security Standard Algorithm Names*. (n.d.). Docs.oracle.com. https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html#cipher-algorithm-names

‌

*keytool-Key and Certificate Management Tool*. (n.d.). Docs.oracle.com. https://docs.oracle.com/javase/6/docs/technotes/tools/windows/keytool.html

‌

*dependency-check-maven – Usage*. (n.d.). Jeremylong.github.io. https://jeremylong.github.io/DependencyCheck/dependency-check-maven/index.html

‌

CISA. (2023). *Cybersecurity Best Practices | Cybersecurity and Infrastructure Security Agency CISA*. Www.cisa.gov. https://www.cisa.gov/topics/cybersecurity-best-practices

‌

*CWE - Common Weakness Enumeration*. (n.d.). Cwe.mitre.org. https://cwe.mitre.org/

‌