****

# 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** | **[Date]** | **Dexter Melton** | **Initial Creation** |

## 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

Dexter Melton

## Algorithm Cipher

For Artemis Financial’s secure communication needs, the Advanced Encryption Standard (AES) is the recommended encryption cipher due to its optimal balance of performance, efficiency, and resistance to cryptographic threats. This symmetric algorithm encrypts data in 128-bit blocks using 256-bit keys, providing robust protection for client data and financial transactions. To meet the company’s checksum requirement, the SHA-256 hash function generates a unique 256-bit digest for each transferred file, enabling recipients to verify data integrity by comparing the original and received hash values.

AES employs cryptographically secure random numbers for its initialization vectors (IVs), ensuring unique ciphertexts even when the same plaintext is encrypted multiple times. While AES relies on symmetric keys for both encryption and decryption, RSA facilitates secure key exchange. Adopted by NIST in 2001 as the successor to DES, AES remains the gold standard for data encryption, widely implemented in TLS protocols and secure file transfers. This approach ensures Artemis Financial maintains both confidentiality and integrity within its web application, effectively safeguarding sensitive client information.

## Certificate Generation

Insert a screenshot below of the CER file.

A computer screen with white text

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

AI-generated content may be incorrect.

## Secondary Testing

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

A screen shot of a computer program

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 screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer screen

AI-generated content may be incorrect.

## Summary

I refactored the code to implement a secure RestController as the protected endpoint for my program's hashing functionality. The ServerController class was designed to address the vulnerabilities identified in the assessment diagram, including cryptography, code error, and code quality. I chose the SHA-256 algorithm for hashing because of its strong security and low risk of collisions. To maintain the application's security, I recommend performing dependency checks every one to two months to identify and resolve potential vulnerabilities promptly. This proactive approach will help safeguard the company’s sensitive information. Additionally, regularly updating the plugins in the pom.xml file will further enhance security by ensuring the use of the latest versions of essential tools and frameworks.

## Industry Standard Best Practices

To maintain the security of the software application, I followed industry-standard best practices by implementing a secure RestController as the protected endpoint for the program's hashing functionality. This approach aligns with secure coding principles, ensuring data transmissions are managed and safeguarded. To ensure data integrity, I employed the SHA-256 hashing algorithm, which is recognized for its strong security features and low collision risk. Additionally, I recommended conducting regular dependency checks every one to two months to identify and address potential vulnerabilities quickly. Keeping the plugins in the pom.xml file updated further enhances security by using the latest versions of essential tools and frameworks.

Applying industry-standard best practices for secure coding significantly boosts the company’s overall well-being by minimizing the risk of data breaches and protecting sensitive information. Regular vulnerability assessments and dependency updates help maintain a resilient security posture, ensuring potential exploits are countered before exploitation. This proactive strategy safeguards client data, strengthens the company’s reputation, builds customer trust, and ensures compliance with regulatory standards.