****

# 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** | **02/18/23** | **Luke Peters** |  |

## 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

Luke Peters

## Algorithm Cipher

Artemis Financial has requested an additional security measure for their web application to ensure secure communication. To best protect against potential hackers from exploiting or seeking financial gain by accessing sensitive information, encryption is recommended. Encryption renders files useless to unauthorized parties without the necessary key. To achieve secure communication, I recommend using Asymmetric communication, where the encryption key is public and the decryption key is private. To maximize security, particularly for external transmissions, I suggest utilizing the SHA-256 cipher algorithm with 256-bit keys for encryption. This algorithm offers excellent high-level bit encryption and a vast number of possible key combinations with a 256-bit length. Furthermore, the SHA-256 algorithm incorporates Java's random number generator, which enhances security by creating a non-reversible checksum that verifies file validity. The hash function will use the SHA-256 cipher to generate a checksum of the given message.

## Certificate Generation

Insert a screenshot below of the CER file.

Text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

Graphical user interface, text, application

Description automatically generated

## Secure Communications

Insert a screenshot below of the web browser that shows a secure webpage.  
  
\*\*\* I believe since this was self-signed cert, the HTTPS is not displaying properly, since it is not an official cert. \*\*\*

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## 

## Secondary Testing

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

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Functional Testing

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

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## Summary

To comply with Artemis Financials’ request for improved software security, I refactored their existing code and addressed areas such as input validation, encryption, access control, and error handling. I added layers of security to the software application by implementing measures like a file verification step through the use of checksums, TLS and SSL encryption technologies, and proper error handling and memory management techniques. I also ensured that access control was in place to restrict sensitive data and functionalities. By following industry-standard best practices for secure coding, I helped Artemis Financial meet regulatory requirements, gain customer trust, and maintain a competitive edge.

## Industry Standard Best Practices

To keep Artemis Financials’ custom software secure, I applied various industry-standard best practices. These included input validation, encryption technologies like TLS and SSL, error handling, memory management, and access control. By doing so, I ensured that sensitive data was protected from cyber-attacks, which could harm the company's finances, reputation, and legal compliance. Adhering to industry-standard best practices also helped the company meet regulatory requirements, gain customer trust, and stay competitive. Therefore, it's vital for any software company to apply these practices to secure its products and ensure its success.