****

# 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** | **Feb 22, 2024** | **Thanh Nguyen** |  |

## 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

Thanh Nguyen

## Algorithm Cipher

I recommend using the Advanced Encryption Standard (AES) algorithm for encryption.

Overview: AES is a symmetric encryption algorithm widely adopted for its security and efficiency.

Hash Functions and Bit Levels: AES operates on fixed-size blocks and supports key sizes of 128, 192, or 256 bits.

Random Numbers, Symmetric vs. Non-Symmetric Keys: AES is a symmetric key algorithm, meaning the same key is used for both encryption and decryption. Random numbers can be utilized in key generation for added security.

History and Current State: AES was established as a standard by the U.S. National Institute of Standards and Technology (NIST) in 2001. It is considered secure and is extensively used worldwide.

## Certificate GenerationA computer screen shot of a black screen Description automatically generated

## Deploy Cipher

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

## Secure Communications

A screenshot of a computer

Description automatically generated

## Secondary TestingA screenshot of a computer Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

Through my manual code review, I identified and addressed any syntactical, logical, or security vulnerabilities in the refactored code. The accompanying screenshot demonstrates that the code now executes without errors.

One potential vulnerability is the password in application.properties file. I would suggest have that saved in environment variable instead of raw password.

## Summary

## In my summary, I've focused on enhancing security through systematic refactoring, addressing key areas outlined in the Vulnerability Assessment Process Flow Diagram.

## Areas Addressed by Refactoring: My attention was primarily on securing communication by transitioning to HTTPS, deploying a robust cryptographic hash algorithm (SHA-256), and implementing secure coding practices. These actions collectively fortify the application against potential vulnerabilities.

## Password Security Enhancement: One potential vulnerability identified is the presence of the password in the application.properties file. To address this, I recommend storing sensitive information, such as passwords, in environment variables instead of using raw values within the code. This measure enhances the overall security of the application by minimizing the risk of unauthorized access to sensitive credentials.

## Process for Adding Layers of Security: I strategically added layers of security by choosing the Advanced Encryption Standard (AES) algorithm for encryption, generating self-signed certificates using Java Keytool, and converting HTTP to HTTPS. This multi-layered approach enhances the overall security posture, ensuring data confidentiality and integrity.

## By meticulously following these steps, I've not only adhered to industry standards but also proactively fortified the software application against potential threats. This comprehensive approach, coupled with the additional measure of securing sensitive information, is vital for maintaining the integrity and security of Artemis Financial's system.

## Industry Standard Best Practices

To maintain the current security of the software application, we diligently adhered to recognized industry standards such as OWASP, CERT, and NIST. By implementing secure coding guidelines, robust authentication and authorization mechanisms, strict input validation, and encryption practices, we fortified the application against common vulnerabilities like injection attacks, unauthorized access, and data breaches. Regular updates, thorough error handling, and comprehensive logging further contribute to a resilient security posture. Our commitment to staying abreast of security best practices ensures not only regulatory compliance but also safeguards sensitive data, mitigates risks, and enables proactive response to potential security incidents.

The value of applying industry-standard best practices extends beyond mere compliance. It serves as a strategic investment in the company's overall wellbeing. By prioritizing security, we mitigate known risks, comply with regulations, and uphold a trustworthy reputation. This, in turn, minimizes potential costs associated with security breaches, enhances business continuity, and positions the company as a competitive leader in the market. Ultimately, a proactive approach to cybersecurity not only protects the organization and its customers but also fosters a resilient and trustworthy digital ecosystem.