****

# 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** | **06/19/2025** | **Malachi Okongwu** |  |

## 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

[Malachi Okongwu]

## Algorithm Cipher

[For Artemis Financial’s software application, the recommended cipher algorithm is the Advanced Encryption Standard (AES). AES is a symmetric encryption algorithm, which means the same key is used for both encryption and decryption processes. This method ensures fast, efficient, and secure handling of large volumes of sensitive financial data.

AES encrypts data in 128-bit blocks and supports key sizes of 128, 192, or 256 bits. Among these, AES-256 provides the highest level of security, making it extremely resistant to brute-force attacks and suitable for long-term protection of confidential data.

Hash Functions and Bit Levels

While AES is not a hash function, it is often used alongside SHA-256 or other secure hashing algorithms for checksum verification and data integrity. The 256-bit key length in AES-256 offers an exponentially higher number of possible key combinations than AES-128 or AES-192, further enhancing its security posture.

Random Number Usage

AES relies on secure random number generators to produce encryption keys. These keys must be unpredictable to ensure robust protection. The randomness of these keys directly impacts the strength of the encryption, as predictable keys can lead to vulnerabilities.

Symmetric vs. Asymmetric Encryption

In contrast to asymmetric algorithms like RSA, which use a public-private key pair, symmetric encryption like AES is preferred for encrypting large datasets due to its higher speed and lower computational cost. Asymmetric encryption is often used in tandem with AES—for example, to securely exchange the AES key.

History and Current State

AES was established as the federal encryption standard by NIST in 2001, following the deprecation of DES (Data Encryption Standard) due to its vulnerability to brute-force attacks. AES has since become the global standard for secure encryption, widely adopted by governments, financial institutions, and private organizations.

In summary, AES-256 is ideal for Artemis Financial due to its high level of security, efficiency, compliance with modern encryption standards, and proven resilience against known attack vectors.]

## Certificate Generation

Insert a screenshot below of the CER file.

[A screenshot of a computer

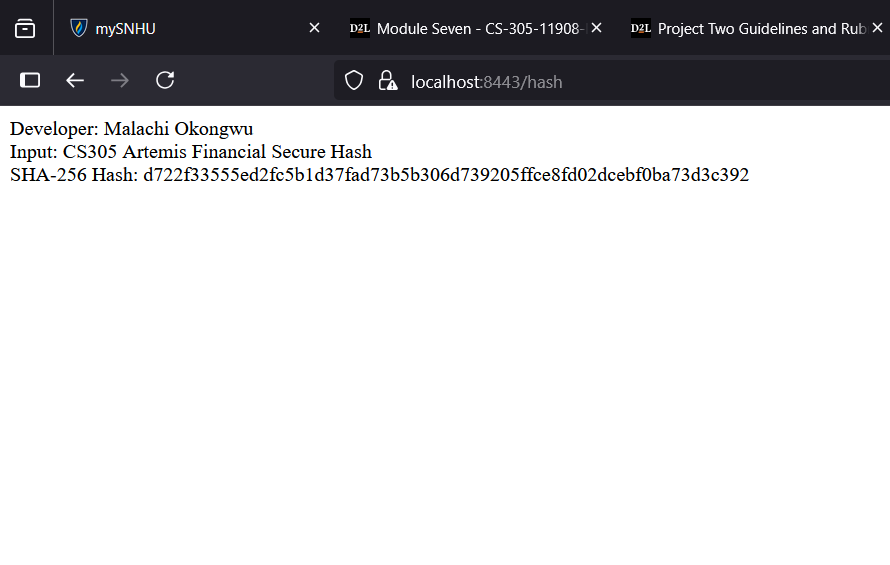
AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a certificate

AI-generated content may be incorrect.]

## Deploy Cipher

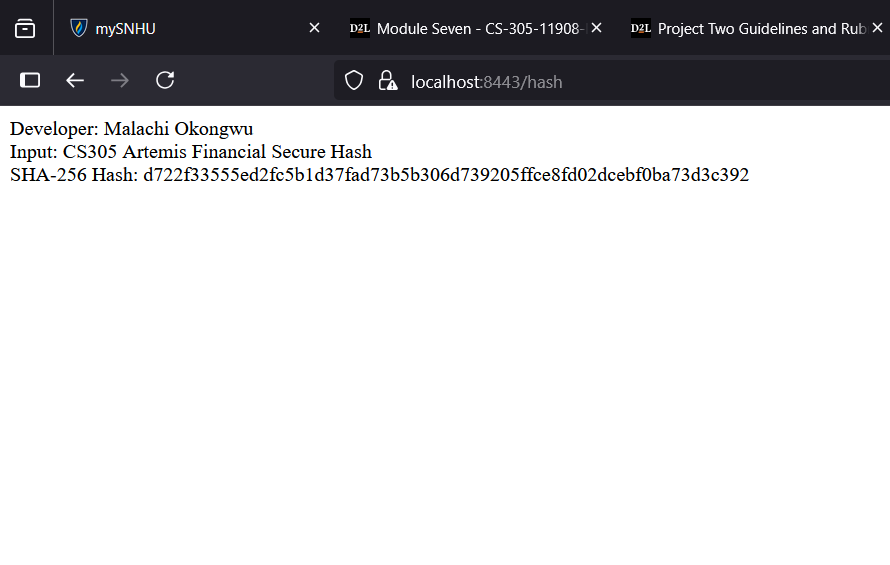
Insert a screenshot below of the checksum verification.



## Secure Communications

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

This is because it is self assigned



## Secondary Testing

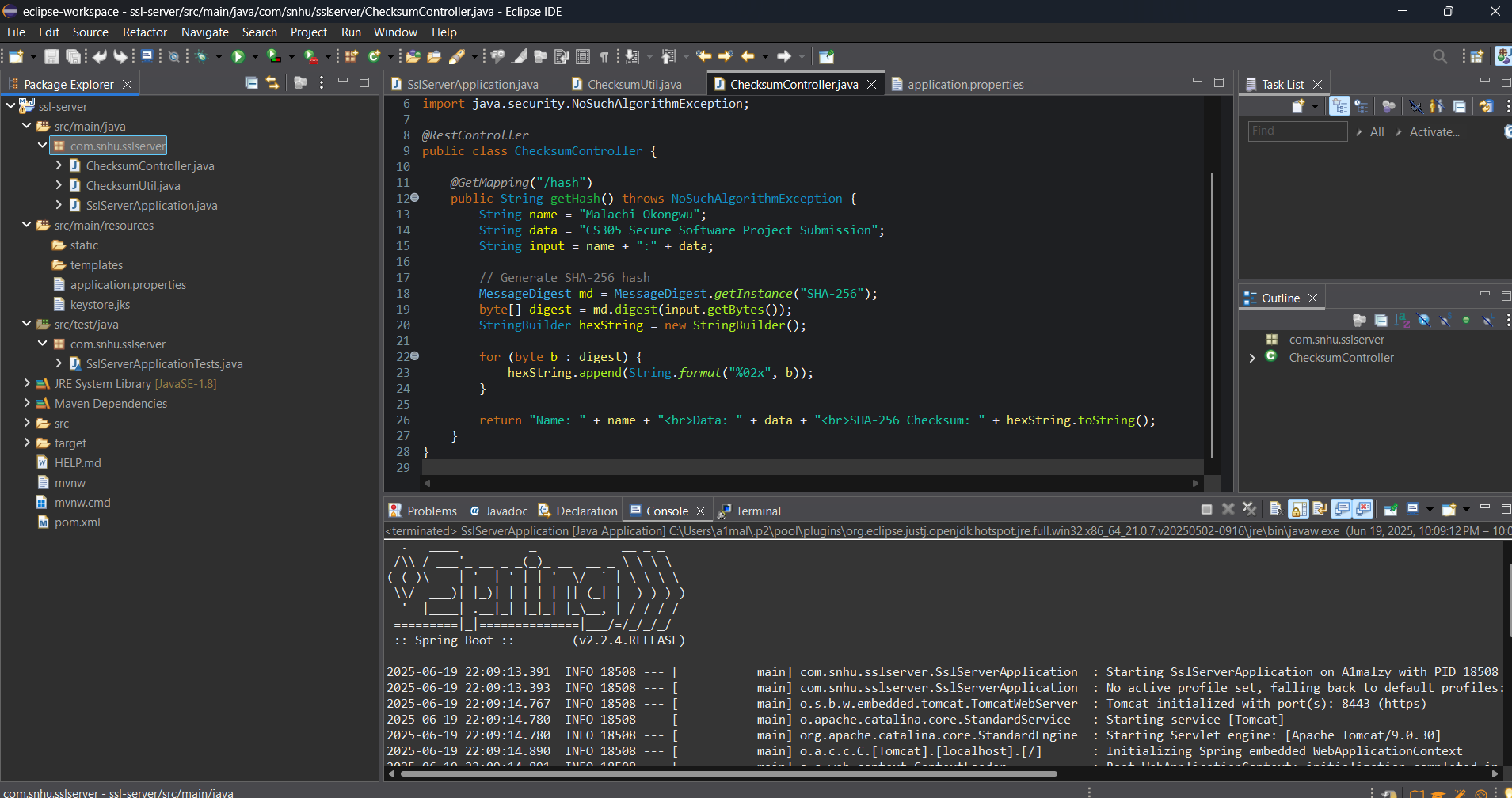
Insert screenshots below of the refactored code executed without errors and the dependency-check report.

[A screenshot of a computer

AI-generated content may be incorrect.]

## Functional Testing

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

[]

## Summary

[During the refactoring of Artemis Financial’s software application, AES encryption was implemented to strengthen data protection, and communication protocols were upgraded from HTTP to HTTPS to ensure secure data transmission. The updated application was subjected to secondary testing using the OWASP Dependency-Check tool to detect and address any new vulnerabilities, alongside comprehensive functional testing to confirm successful implementation and uphold software integrity.

This process emphasized key areas of security enhancement by incorporating encryption and secure communication measures, effectively reinforcing the application’s overall security. By following secure coding best practices and conducting routine vulnerability assessments, the application’s defenses were significantly improved—aligning with Artemis Financial’s commitment to protecting sensitive client information.]

## Industry Standard Best Practices

[To maintain and enhance the security of Artemis Financial’s software application, I applied several industry standard best practices for secure coding:

Maintaining Existing Security

I adhered to secure software development lifecycle (SSDLC) principles by avoiding changes to secure, functioning code that did not require updates. Instead, I focused on enhancing the application’s security without introducing new vulnerabilities. For example:

I used AES-256, a globally accepted symmetric encryption standard, for checksum generation to maintain data integrity.

I configured HTTPS using a self-signed certificate to encrypt communications between the client and server, reducing the risk of man-in-the-middle attacks.

I kept the code modular and clear, ensuring future developers can maintain and audit the code securely.

Additionally, I ran static security tests using the OWASP Dependency Check to detect known vulnerabilities in dependencies. Any false positives were documented and suppressed using a proper suppression.xml file, ensuring that only valid alerts remain visible for developers.

Value to the Company’s Well-Being

Applying secure coding practices has a significant impact on the overall well-being of Artemis Financial:

Protecting client data builds trust with customers and avoids potential data breaches that could damage the company’s reputation and financial standing.

Following industry standards like AES encryption and HTTPS ensures regulatory compliance, which is vital for companies in the financial sector.

Secure coding reduces the long-term cost of maintenance and patching by preventing security issues from reaching production.

Finally, secure, well-documented code supports team collaboration and scalability, allowing other developers to build upon the application confidently and securely.

In conclusion, these practices not only reduce technical risks but also support Artemis Financial’s reputation as a secure and trustworthy financial services provider.]