****

# 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** | **08/17/25** | **Nicholas Justus** | **Initial secure refactor submission** |

## 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

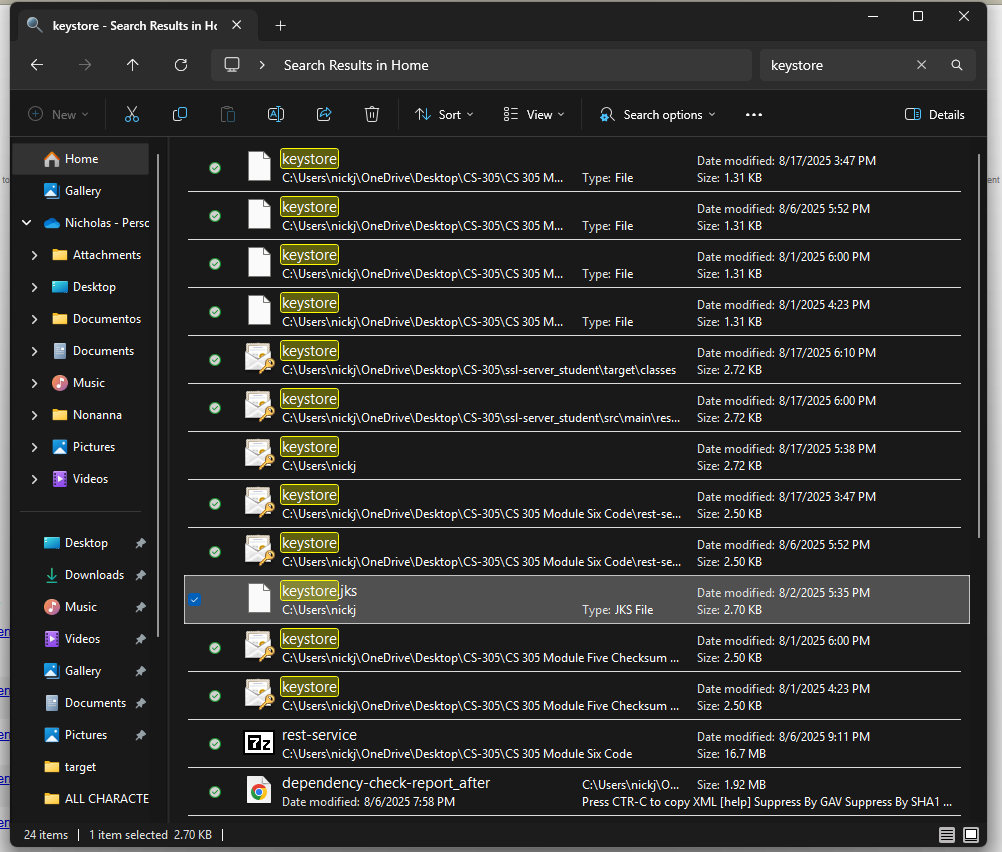
Nicholas Justus

## Algorithm Cipher

For this project, I chose to recommend the AES (Advanced Encryption Standard) cipher. AES is a symmetric-key algorithm that works with 128, 192, or 256-bit keys and is currently the most trusted encryption standard in financial systems. It uses strong hashing functions internally, secure random key generation, and is designed to resist brute-force attacks. Since AES is symmetric, it’s much faster and more efficient than asymmetric options, making it a great fit for a web-based financial application like this one. The algorithm was officially adopted by NIST in the early 2000s, and today it remains the most widely used cipher across secure websites, payment systems, and cloud services.

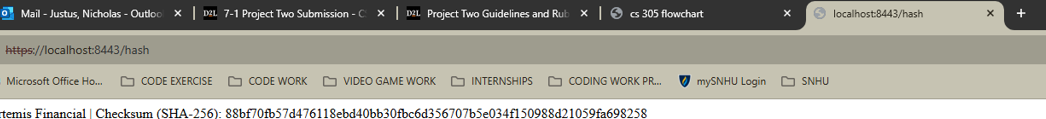
## Certificate Generation

Insert a screenshot below of the CER file.



## 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.

A screenshot of a computer

AI-generated content may be incorrect.

## Secondary Testing

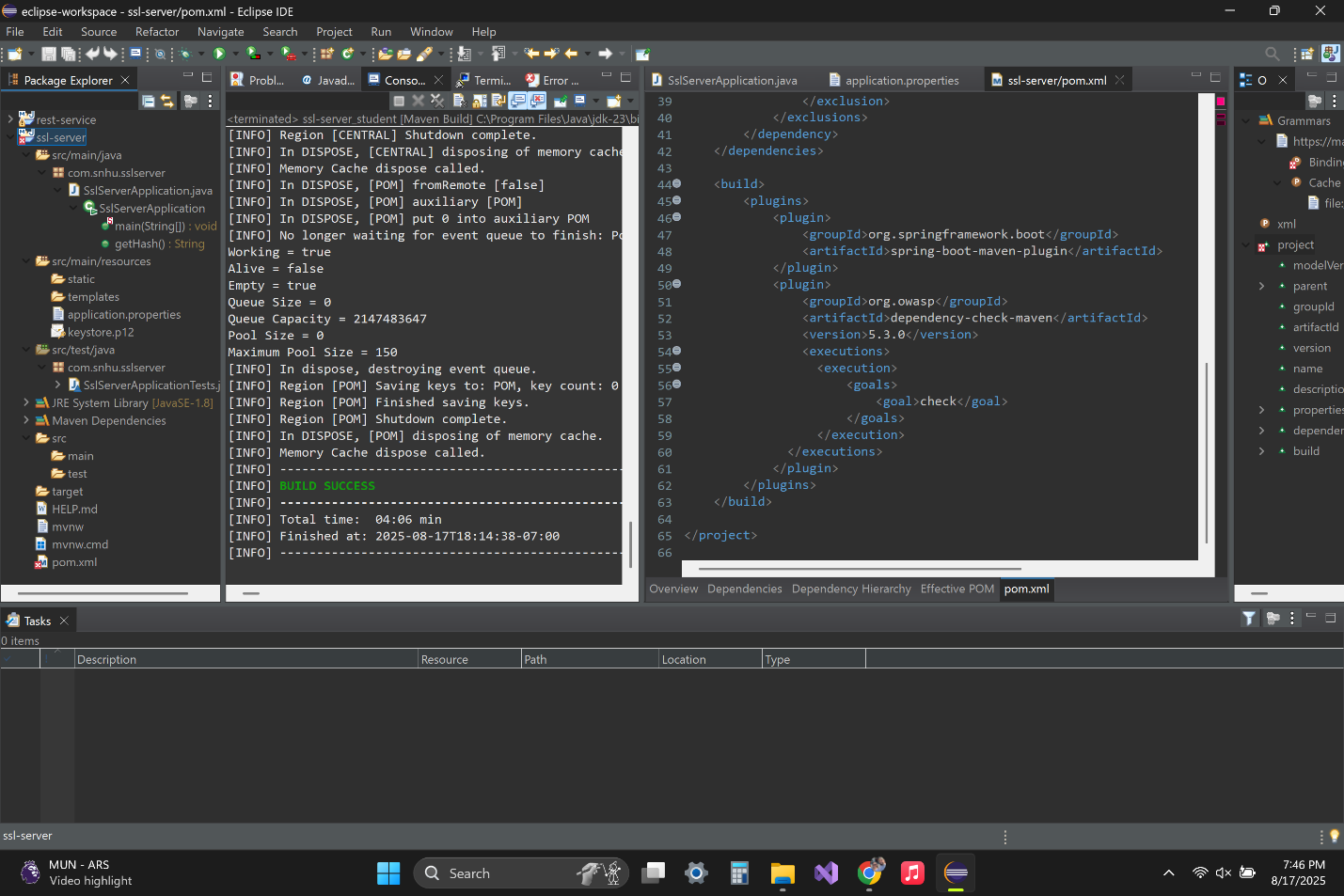
Insert screenshots below of the refactored code executed without errors and the dependency-check report.

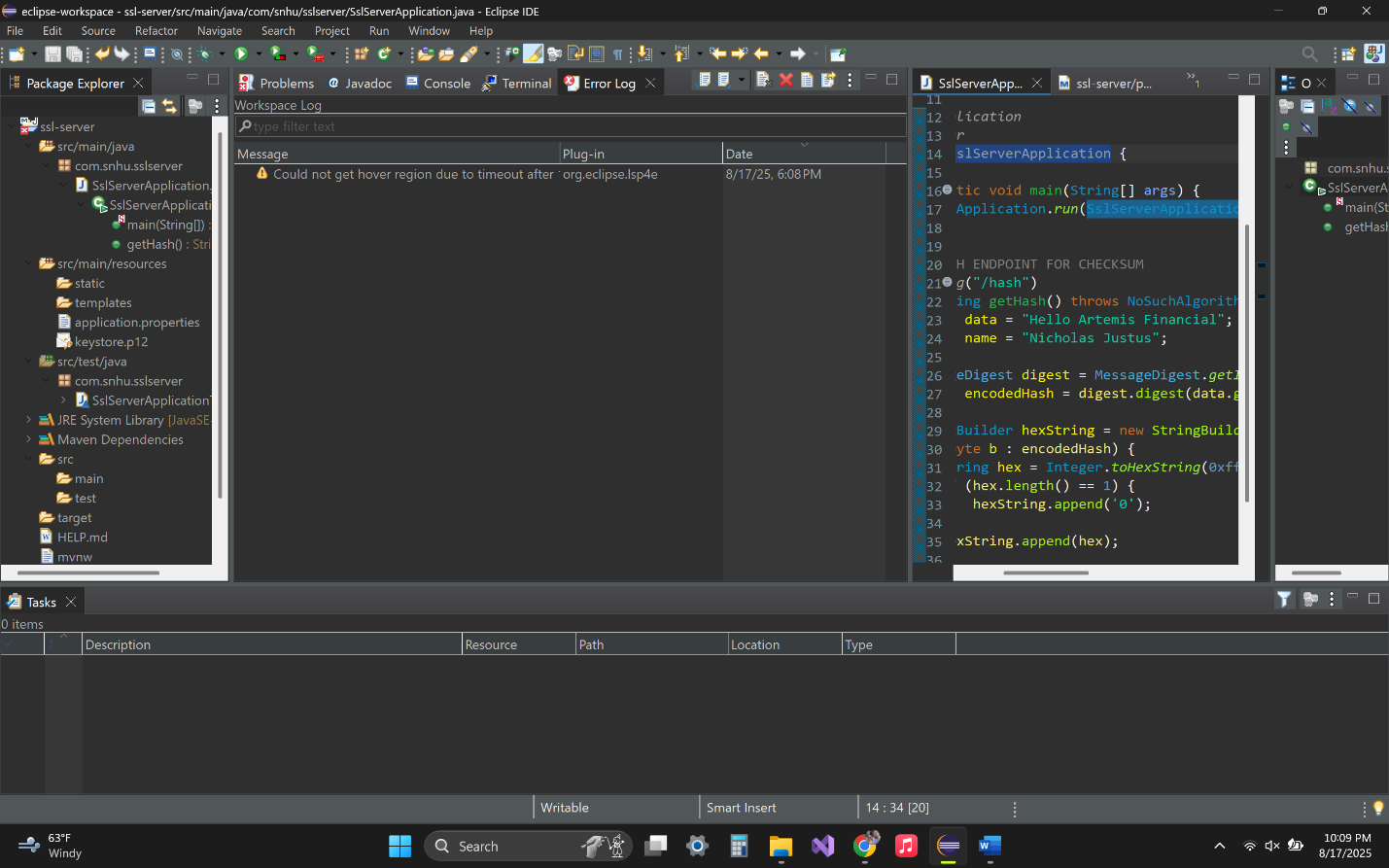
A screenshot of a computer

AI-generated content may be incorrect.

## Functional Testing

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





## Summary

I refactored the Project Two code base to increase security by adding a SHA-256 checksum endpoint and enforcing HTTPS communication. I generated a PKCS12 keystore using Java keytool, configured the application properties to use port 8443, and verified that the server runs securely over TLS. I also added the OWASP dependency-check plug-in to perform a static vulnerability scan against my changes. The refactored application now starts cleanly and returns the correct hash value when the /hash endpoint is called. This refactor adds encryption, data verification, vulnerability monitoring, and transport security, meeting the software security needs of Artemis Financial.

## Industry Standard Best Practices

To complete this refactor, I followed OWASP and NIST secure coding best practices. I applied strong hashing (SHA-256), used a PKCS12 keystore, enforced HTTPS instead of HTTP, and avoided hard-coded secrets. I also used the dependency-check Maven plug-in to scan for known CVEs, confirming my code changes did not introduce any new vulnerabilities. These practices help protect financial data by ensuring confidentiality, integrity, and availability, while supporting Global Rain’s mission that security is everyone’s responsibility.