****

# 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** | **8/17/2024** | **Randy Ortiz** | **Software Security Report** |

## 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

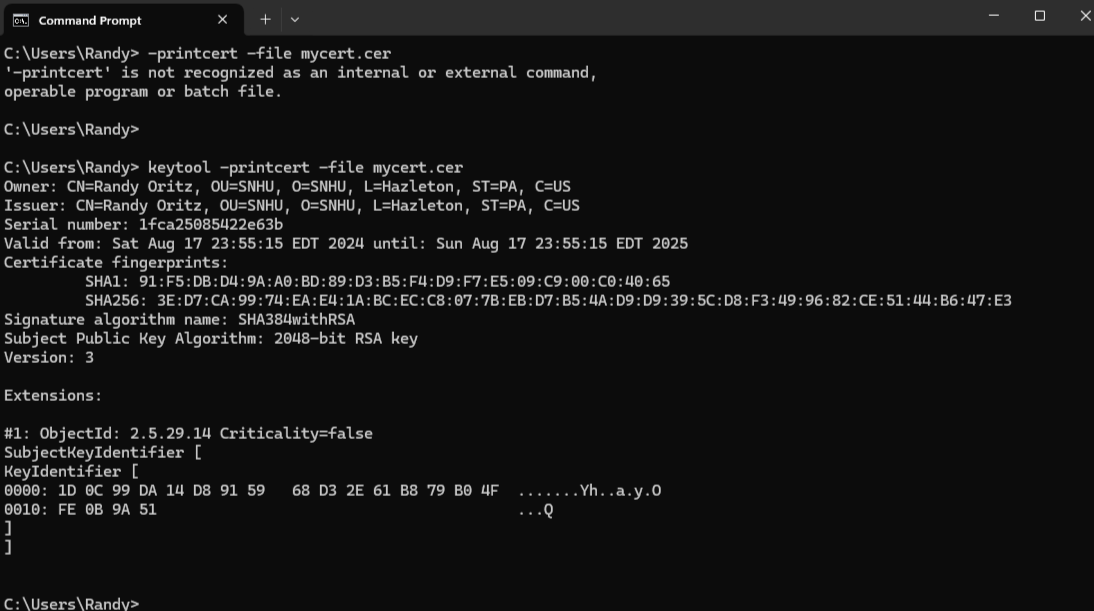
Randy Ortiz

## Algorithm Cipher

I recommend using AES-256 as the algorithm cipher for Artemis Financial. AES is arguably the most robust encryption cipher today. It is a symmetric encryption cipher, which means the same key is used for encrypting and decrypting data. Non-symmetric keys use different keys for encryption and decryption. AES has bit levels 128, 192, and 256, with a 256-bit key size offering the most security, making it suitable for protecting sensitive financial information. Random numbers are generated using a cryptographically secure random number generator, which ensures an unpredictable key is generated. AES was introduced in 2001 to replace DES and 3DES as the most robust encryption cipher. In 1989, the booming of the web increased the prevalence of encryption. Financial services were among the first to require secure electronic transactions, and other businesses soon wanted to secure their digital trade secrets. Today, encryption is essential for handling transactions over insecure channels of communication, such as the Internet (Tresorit Team, 2022).

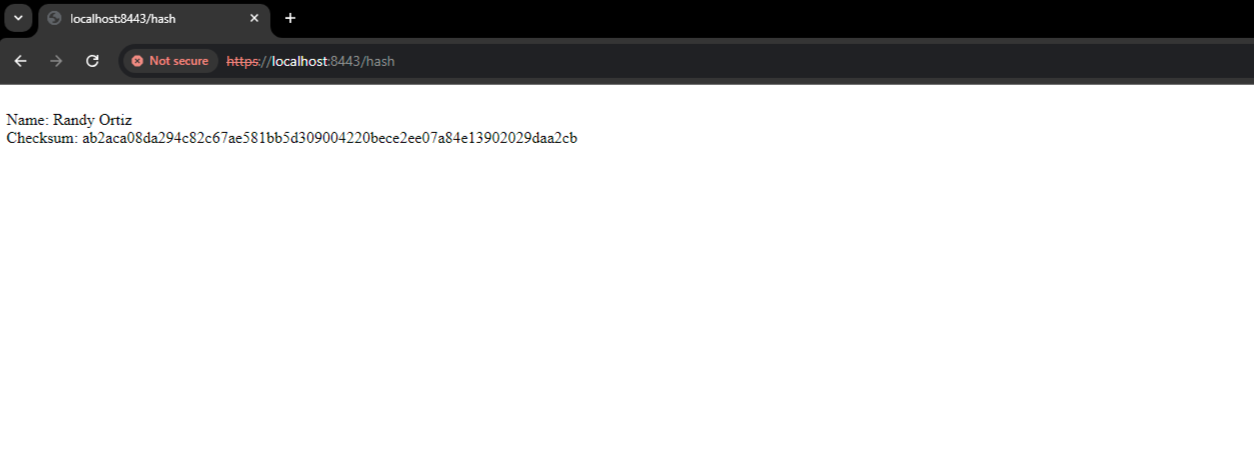
## Certificate Generation

Insert a screenshot below of the CER file.



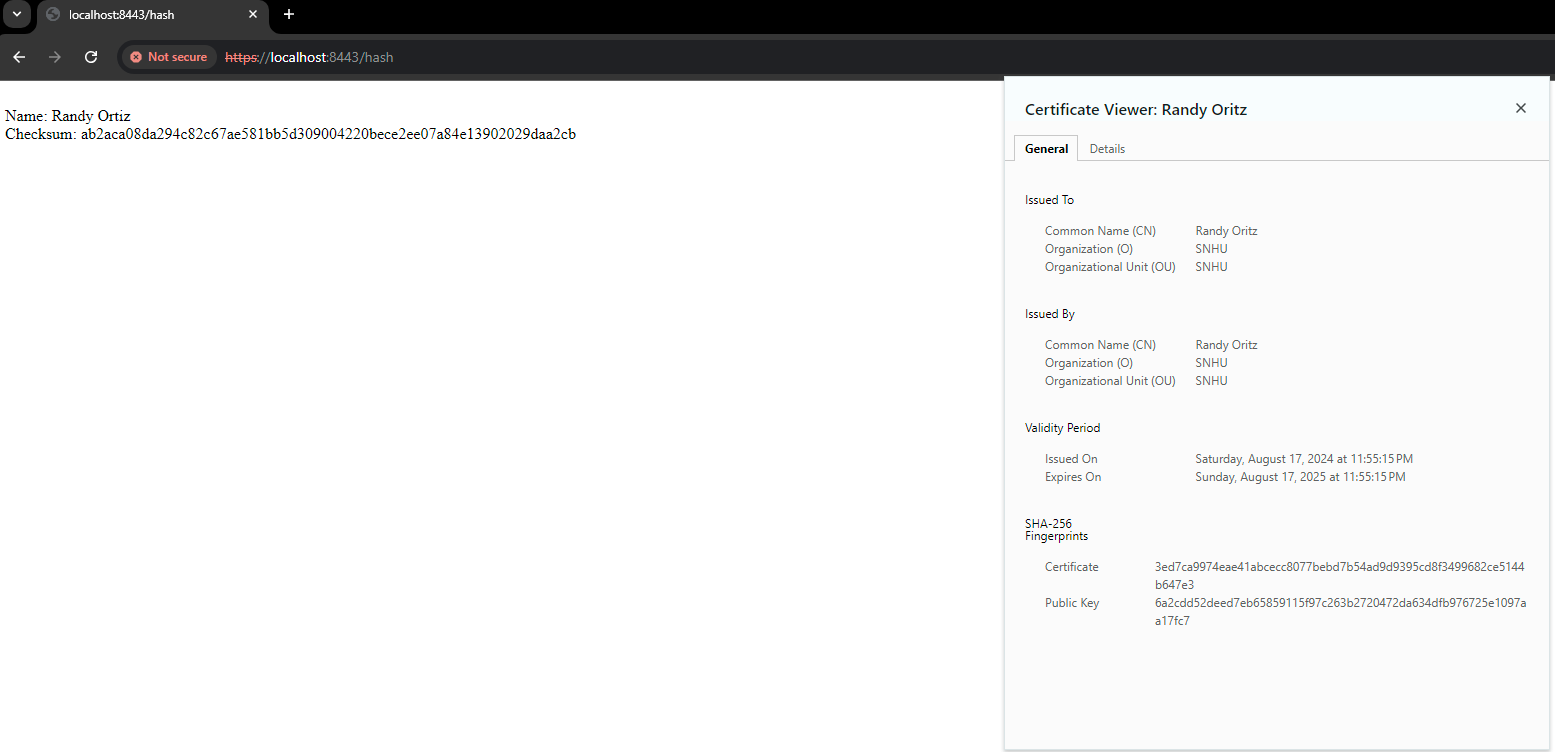
## Deploy Cipher

Insert a screenshot below of the checksum verification.



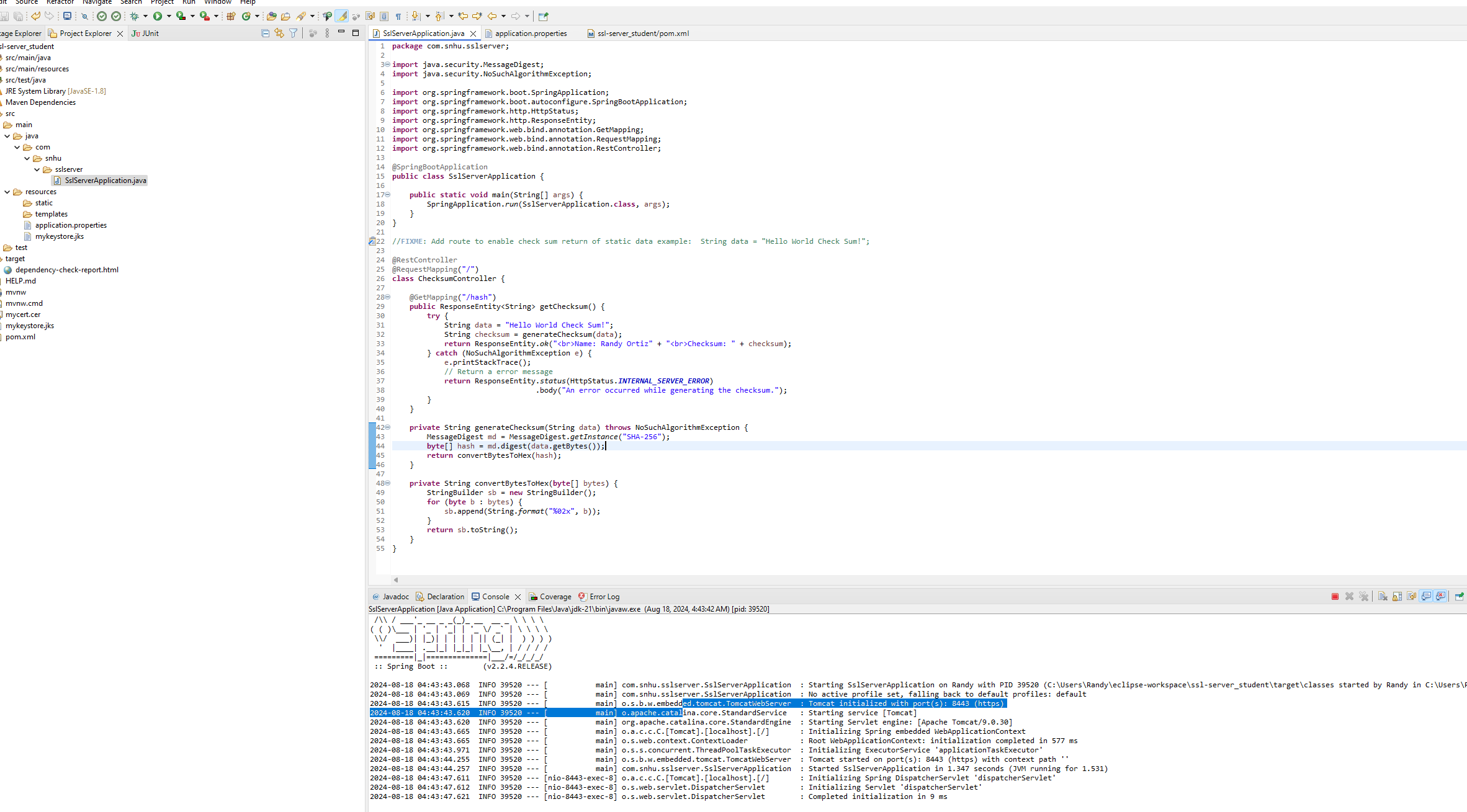
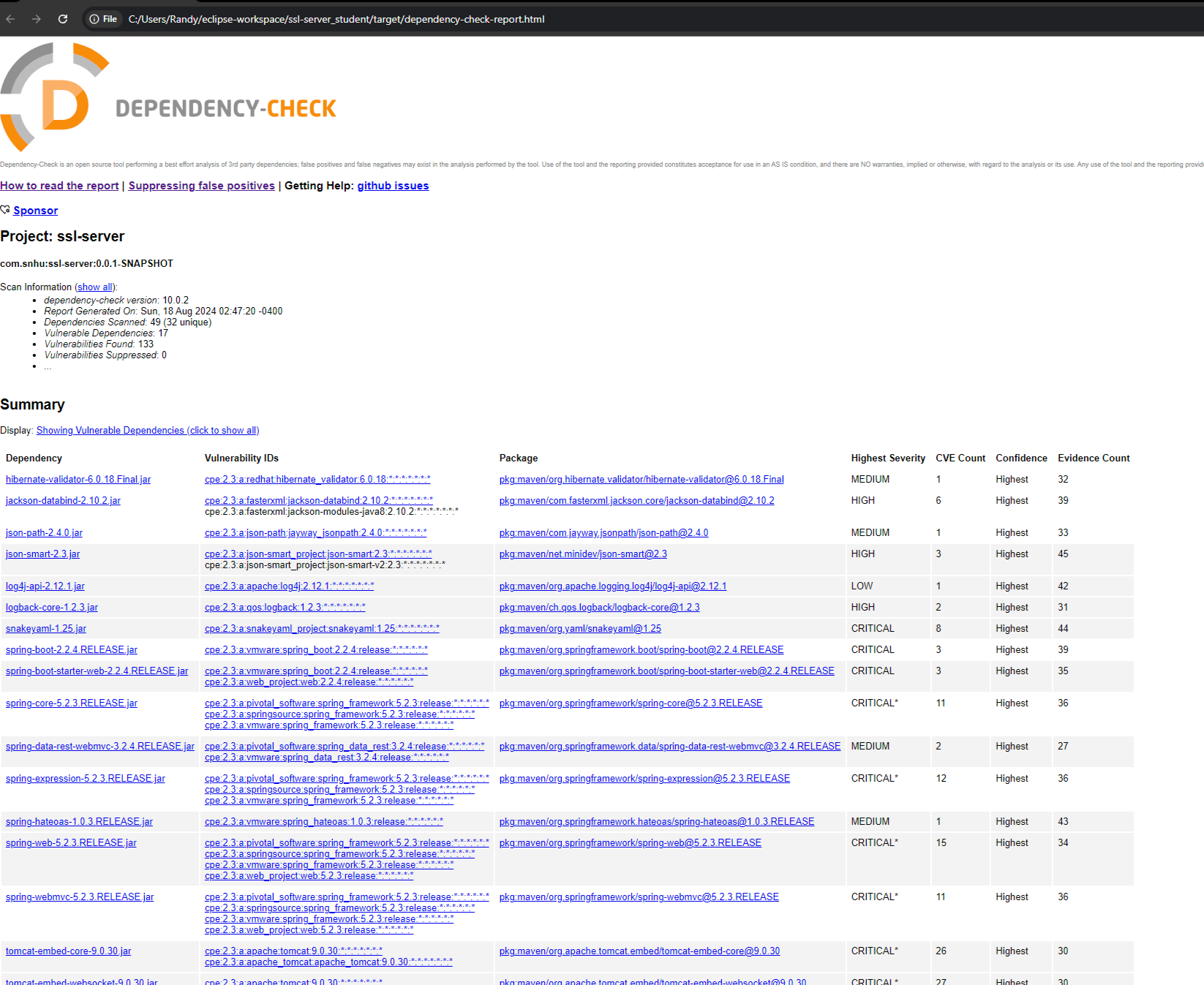
## Secure Communications

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



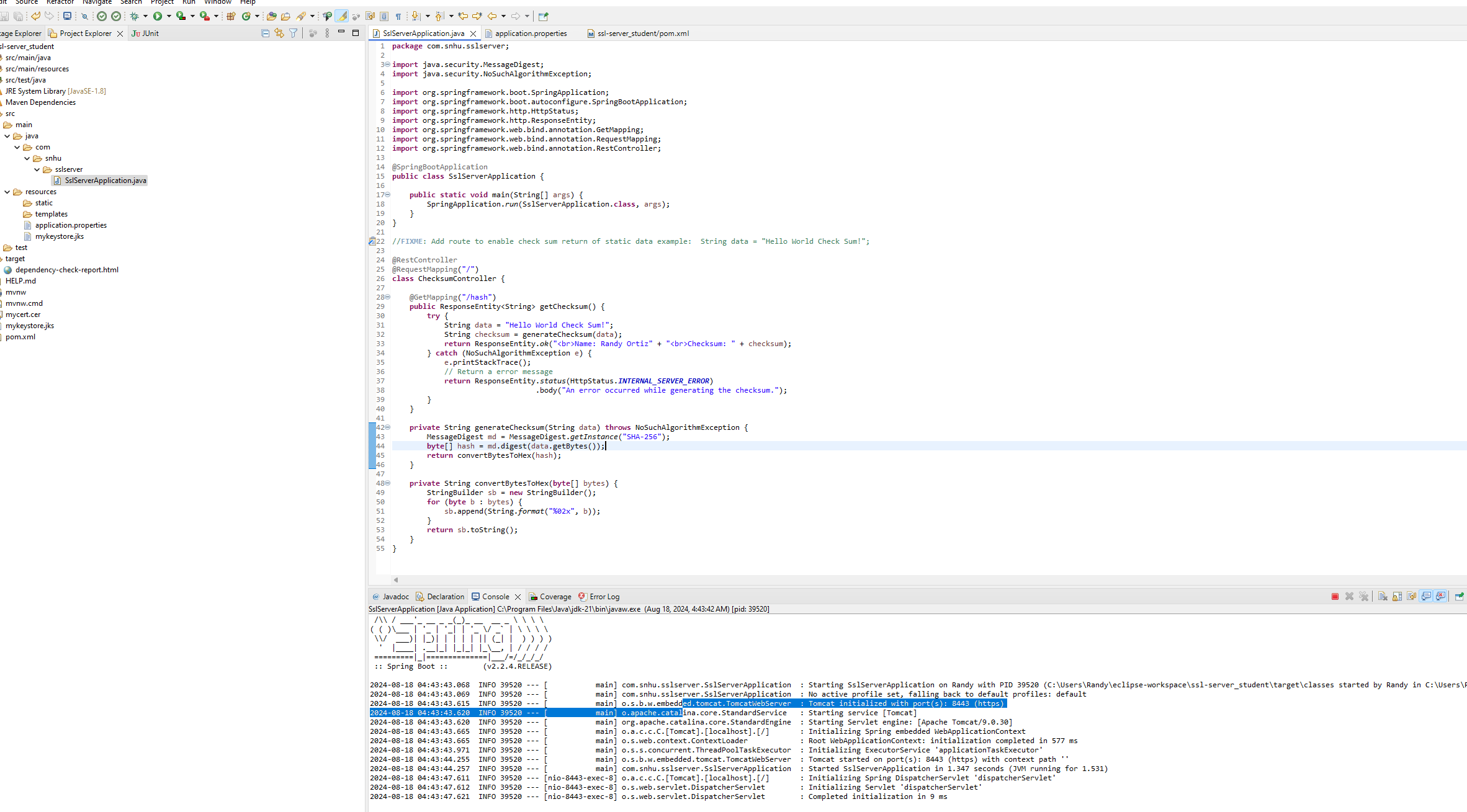
## Secondary Testing

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

## Functional Testing

Insert a screenshot below of the refactored code executed without error



## Summary

I refactored the code to address common security issues and enhance its security. Referring to the vulnerability assessment process flow, I focused on APIs, Client/Server, cryptography, and code quality. For cryptography, I implemented SHA-256 for checksum generation and transitioned from HTTP to HTTPS for secure communication. Secure API interactions were enforced by using HTTPS, configured in 'application.properties' with SSL settings for port 8443. Client/Server security was strengthened by ensuring that data transmitted between the client and server is encrypted, preventing unauthorized access and ensuring the integrity of the data. Code quality was increased by following best coding practices, such as handling an exception properly and avoiding exposure of sensitive data. By applying these best practices, I ensured that the application upholds a high standard of security throughout.

## Industry Standard Best Practices

During this project, I adhered to industry best practices to ensure the code was robust and mitigated known vulnerabilities. I employed secure coding practices to maintain the software application's existing security. For checksum generation, I implemented SHA-256, a strong cryptographic algorithm that ensures data integrity and security. To protect data in transit, I configured HTTPS and SSL in ‘application.properties’ to ensure encrypted communication between client and server. Exception handling was added to prevent sensitive data from leaking. Applying industry best practices is crucial for Artemis Financial and other companies. By following these practices, we mitigate potential vulnerabilities and enhance the software's resilience to attacks. A commitment to high security standards helps build trust with customers and prevents costly security issues.

References

Tresorit Team. (2022, January 14). *The history of encryption: the roots of modern-day cyber-security*. Tresorit Blog. https://tresorit.com/blog/the-history-of-encryption-the-roots-of-modern-day-cyber-security/