****

# 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** | **4/20/2025** | **Christopher O’Dell** |  |

## Client



## Developer

Christopher O’Dell

## Algorithm Cipher

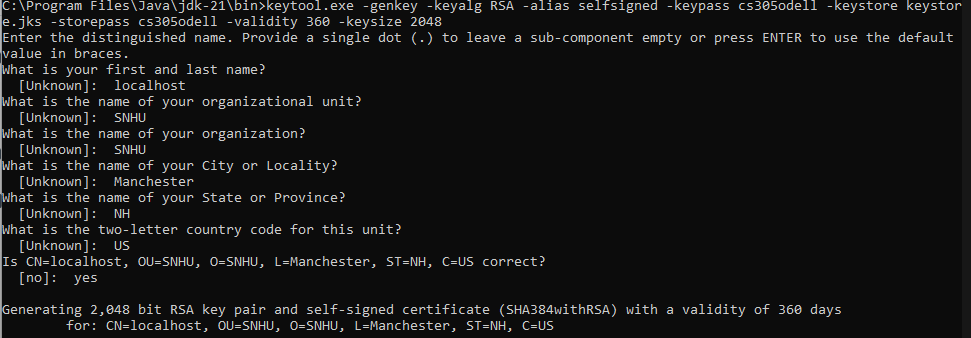
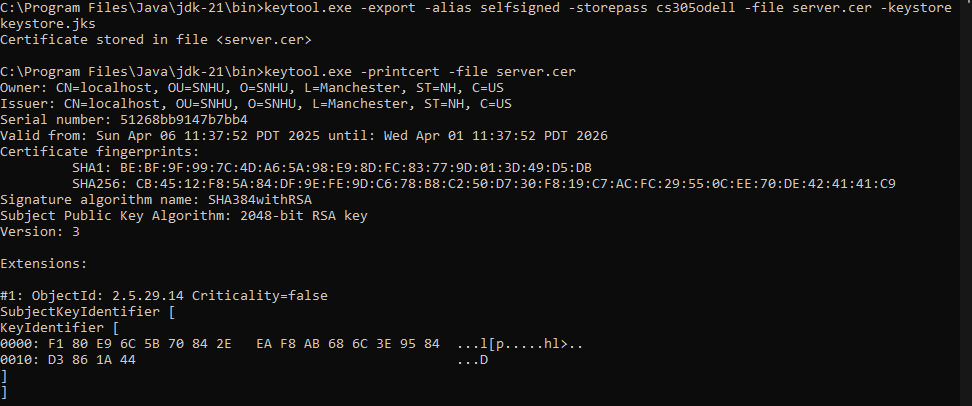
Artemis Financial is dedicated to delivering robust financial solutions to clients around the globe. In support of this mission, implementing strong encryption practices is essential. One of the most reliable options available today is the SHA-256 algorithm, a cryptographic hash function that offers high-level security against unauthorized access. Its design makes it practically immune to brute-force attacks, requiring an extraordinary amount of time and computing power to crack—often considered computationally infeasible.

SHA-256 is widely recommended in the financial sector for secure communication due to its strength and consistency. The algorithm works by compressing input data into a fixed-size hash value, with randomness playing a critical role in the process. The 256-bit hash ensures a vast number of possible combinations, adding to the difficulty of predicting or reversing the output, thereby enhancing data protection.

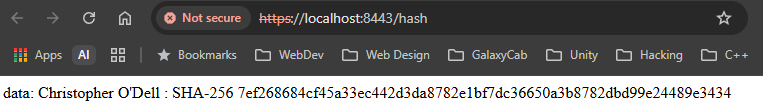
When it comes to encryption methods, symmetric and asymmetric keys both play vital roles. Symmetric encryption, used in algorithms like AES-256, relies on a single shared key to encrypt and decrypt information. This approach is efficient and fast, making it ideal for situations where speed and performance are priorities. On the other hand, asymmetric encryption uses a pair of keys—one public and one private—and is commonly employed in secure internet communications due to its enhanced security.

Encryption has a long and storied history, with origins tracing back to ancient civilizations. Over time, encryption has evolved into a sophisticated field of study and innovation. Today, as data security continues to be a global concern, advanced encryption standards remain at the forefront of protecting sensitive financial information and upholding privacy across digital platforms.

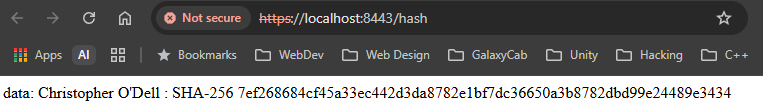
## Certificate Generation

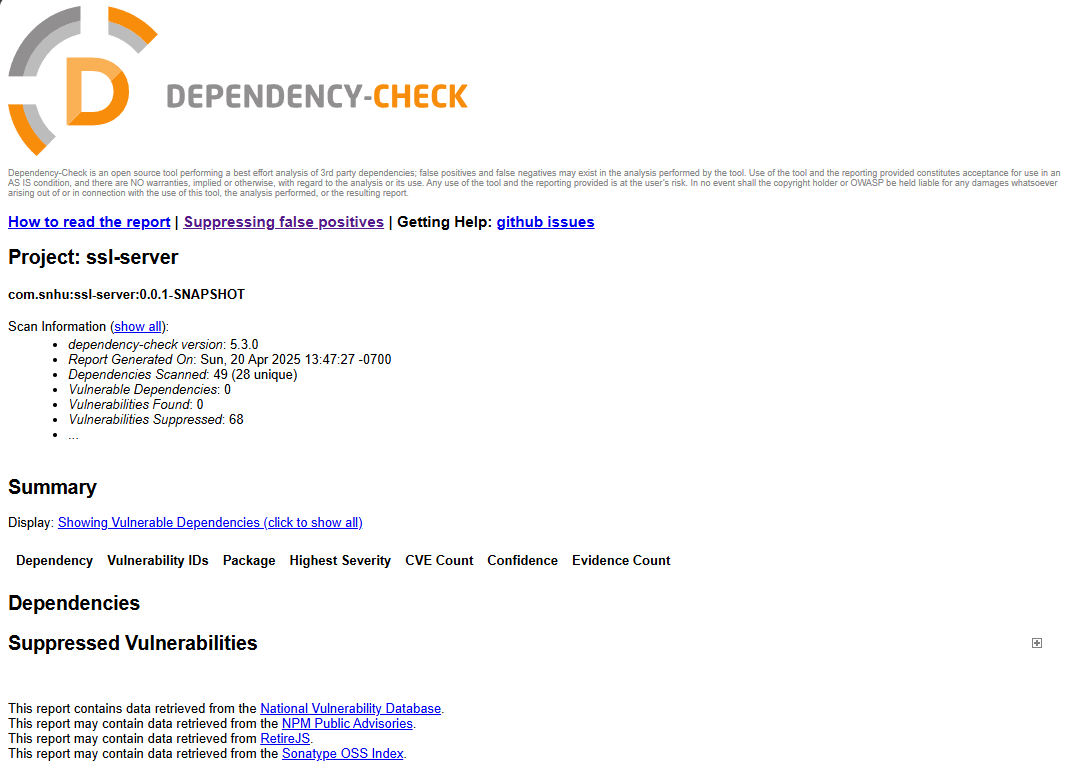
## Deploy Cipher



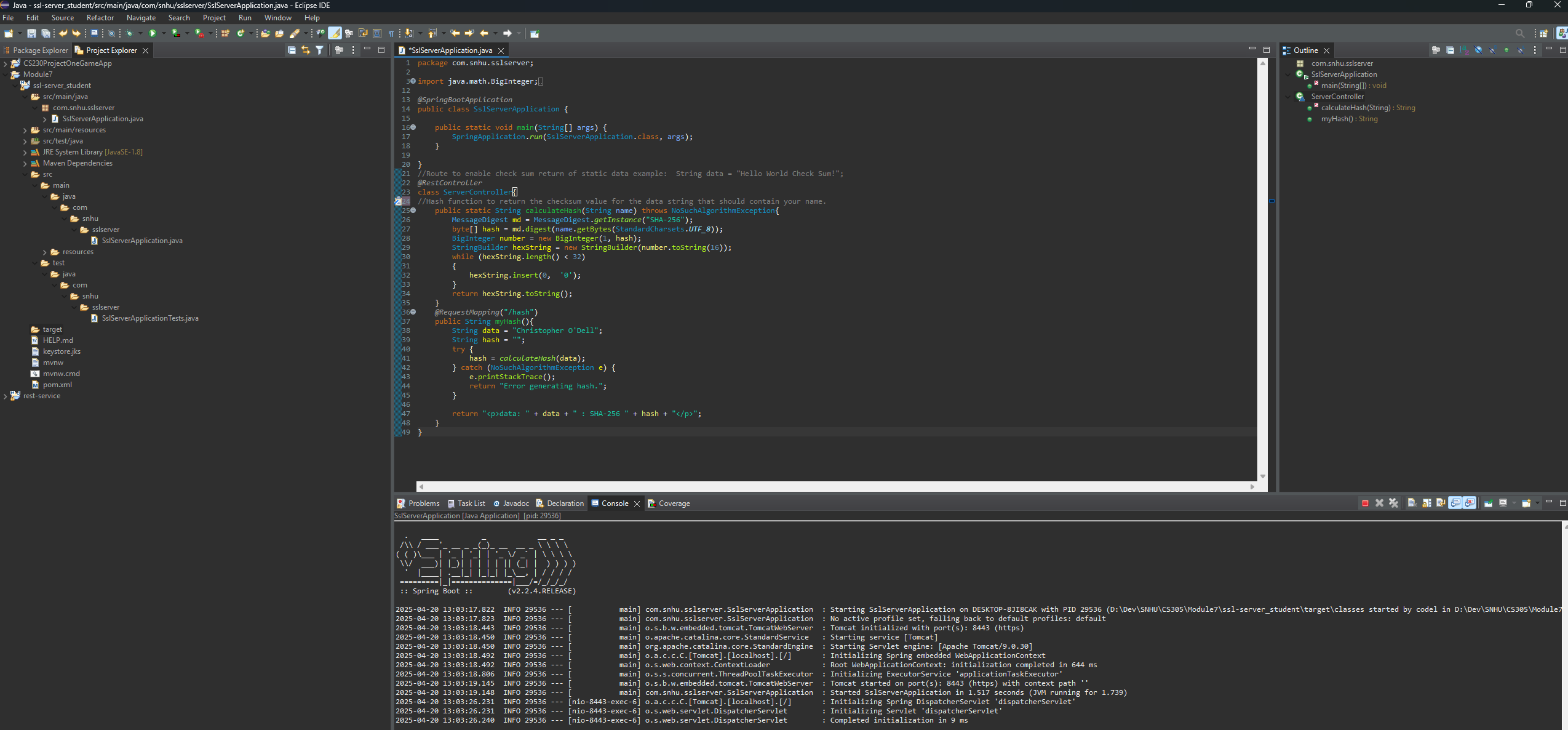
## Secure Communications

It should be noted that my browser does not allow trusting of self-signed certificates.

## Secondary Testing



## Functional Testing



## Summary

In refactoring my code, I focused on enhancing the security of the application by implementing a secure RestController to manage the application's RESTful services. The SecureEndpointController is designed to handle sensitive requests securely, ensuring that data integrity and privacy are maintained. As part of the security enhancements, I switched to the SHA-256 hashing algorithm, which is widely recognized for its robustness and low likelihood of collision vulnerabilities.

To further strengthen security, I recommend automated dependency scanning tools into the build pipeline. I recommend conducting these scans on a monthly basis to detect any outdated libraries or known vulnerabilities. This proactive approach helps mitigate potential threats before they can impact the application. Additionally, I ensured that all dependencies in the pom.xml file are configured to pull the latest stable versions, which contain important security patches.

By following these practices and integrating multiple layers of security, such as secure communication channels and dependency management, the application is now better protected against potential threats. Regular audits and updates to both code and dependencies are crucial in maintaining the integrity of the system and safeguarding sensitive data.

## Industry Standard Best Practices

In the refactored code, I incorporated the @RestController and @RequestMapping annotations to ensure the program operates smoothly whenever a hash is generated. I also selected the SHA-256 cipher for hashing, which is widely recognized as a secure algorithm. Given that this approach aligns with best practices followed by other financial institutions, it provides a solid foundation for securing sensitive data, particularly in the context of a financial services firm.

To maintain a high level of security, I strongly recommend the establishment of a dedicated cybersecurity or IT team. This team would be responsible for conducting daily security checks, applying regular updates, addressing bug fixes, and performing dependency audits. By maintaining a proactive approach to cybersecurity, the team can help safeguard the organization’s network and ensure the integrity of its systems.