****

# 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** | **6/20/2024** | **Jeffrey Conwi** |  |

## Client



## Developer

Jeffrey Conwi6

## Algorithm Cipher

We have chosen to use AES-256 encryption for Artemis Financial’s application. AES-256 is a strong encryption method that protects data very well. It is like a complex lock that only the right key can open. This makes it very hard for anyone who should not see the information to access it.

Discussion on Hash Functions and Bit Levels:

AES-256 uses a key that is 256 bits long. Imagine a key with 256 different parts to it; each part must be correct to unlock the data. This long key helps keep the data safe because guessing or finding the right key is extremely difficult.

Use of Random Numbers and Key Types:

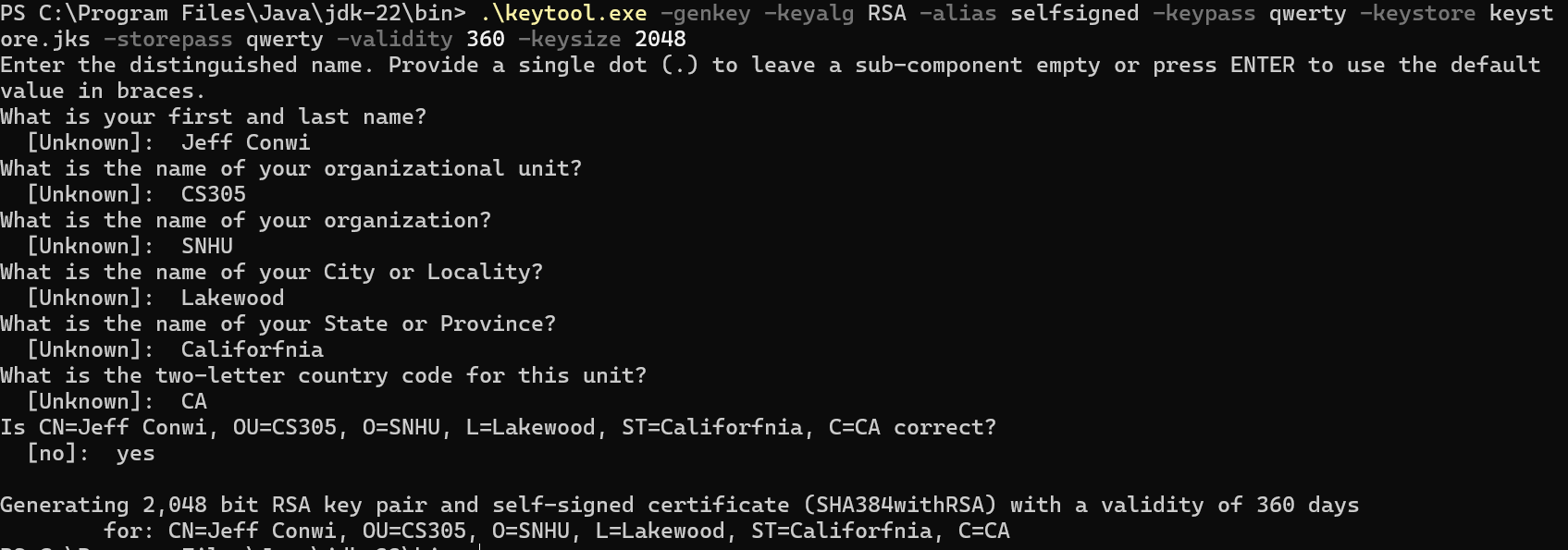
When using AES-256, it is important to create the keys with random numbers. This means every key is unique and unpredictable, making it even harder for unwanted people to unlock the data. Since AES-256 uses the same key to lock and unlock data (known as symmetric encryption), keeping the key safe is very important.

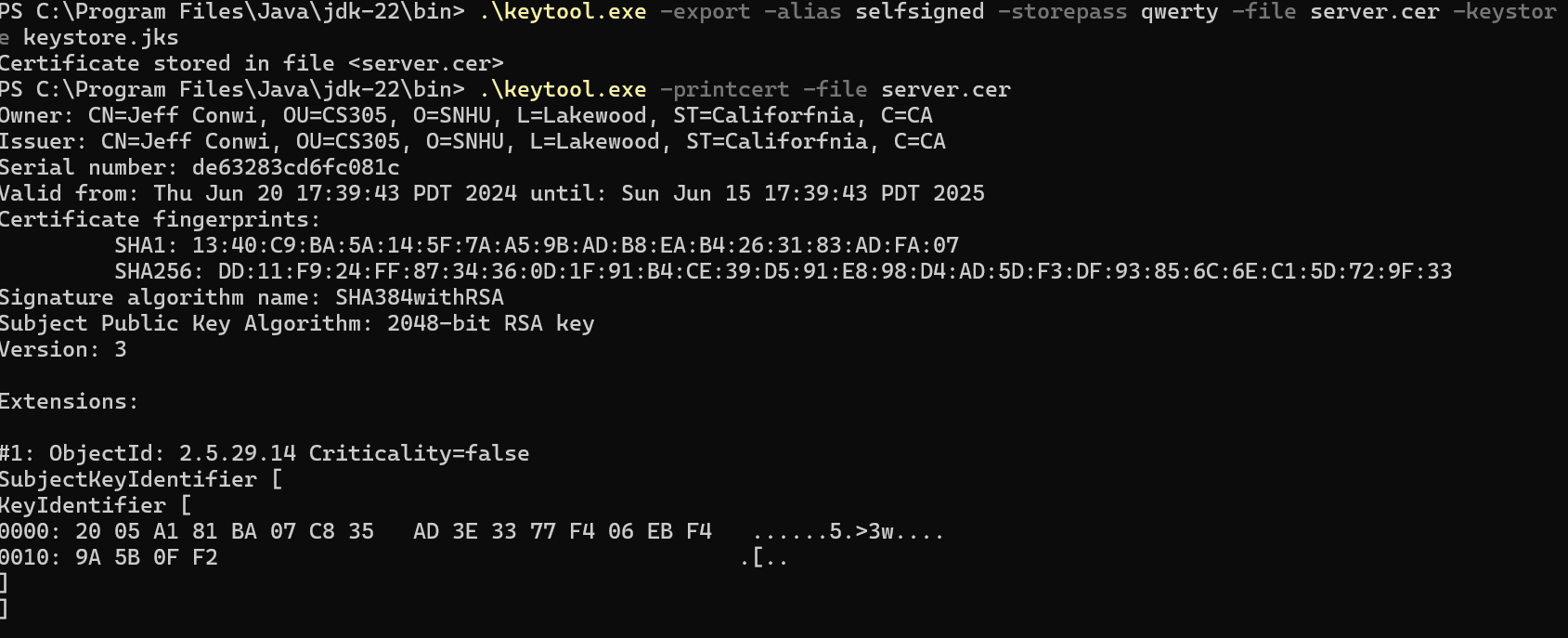
History and Current State:

AES was created to replace an older encryption method that was no longer safe. It has become a standard way to protect data, especially for the U.S. government and businesses around the world. Today, AES-256 is trusted because it has proven to be secure against all known attacks, meaning no one has found a way to break it when it is used correctly.

## Certificate Generation

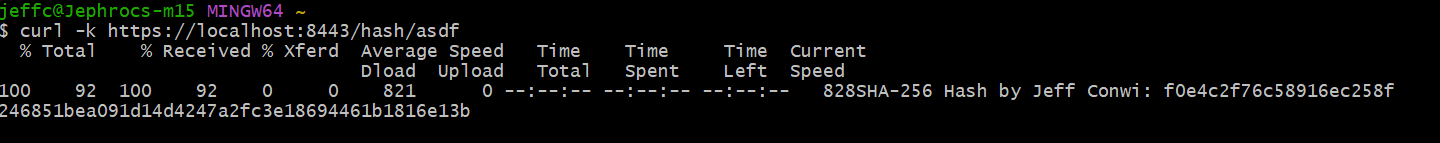
Insert a screenshot below of the CER file.





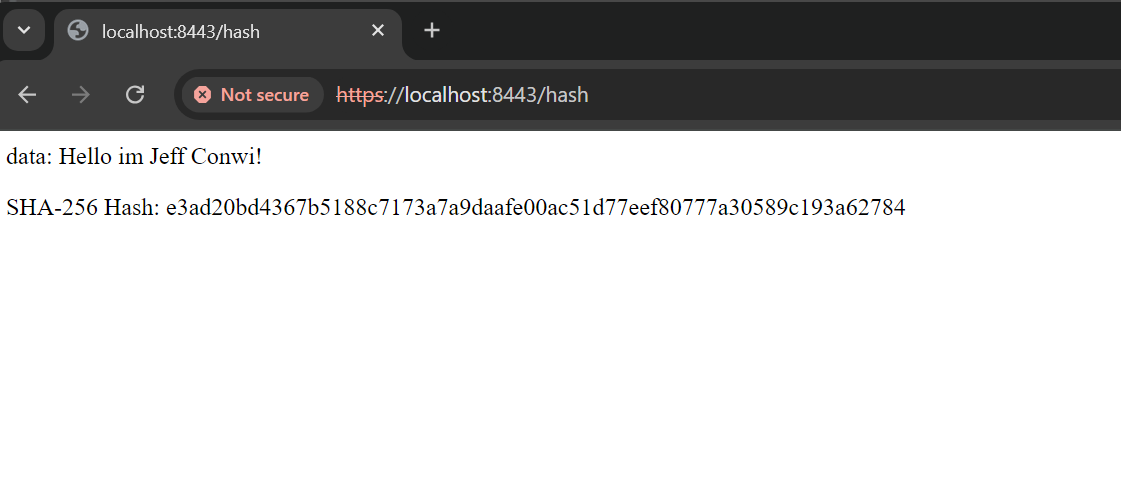
## Deploy Cipher

Insert a screenshot below of the checksum verification.



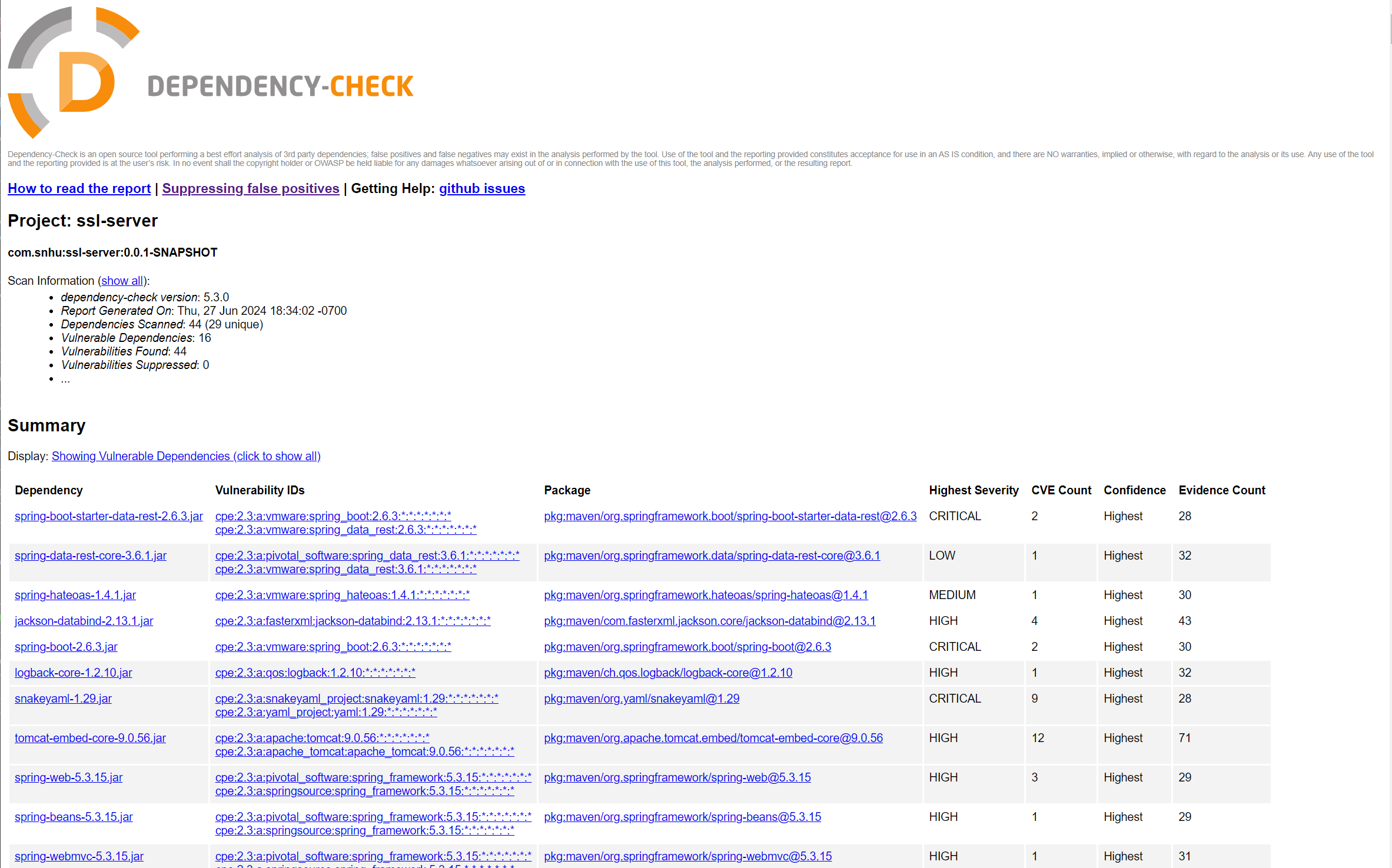
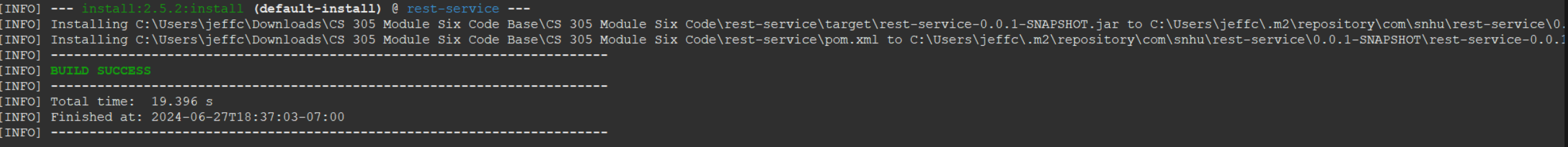
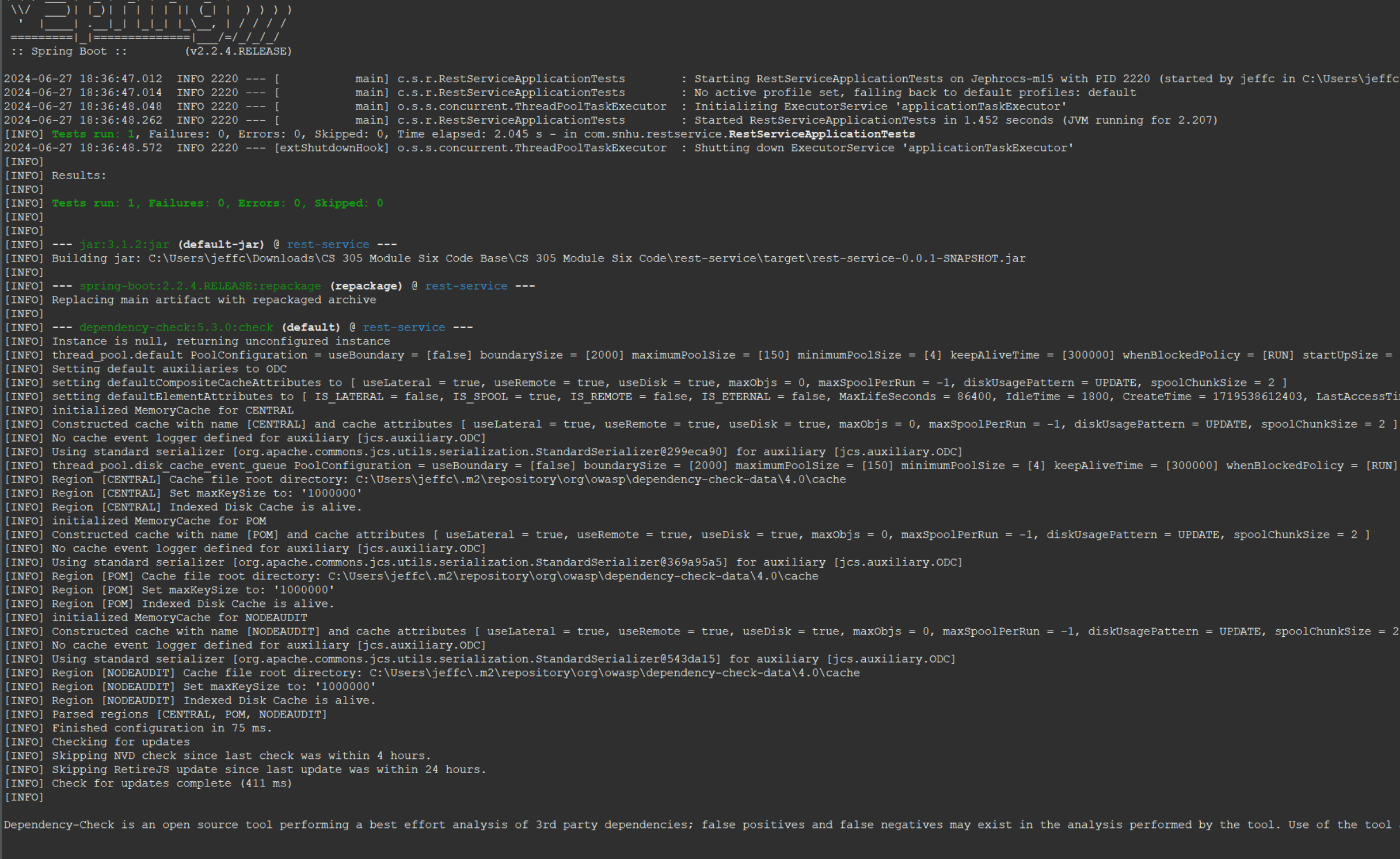
## Secure Communications

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



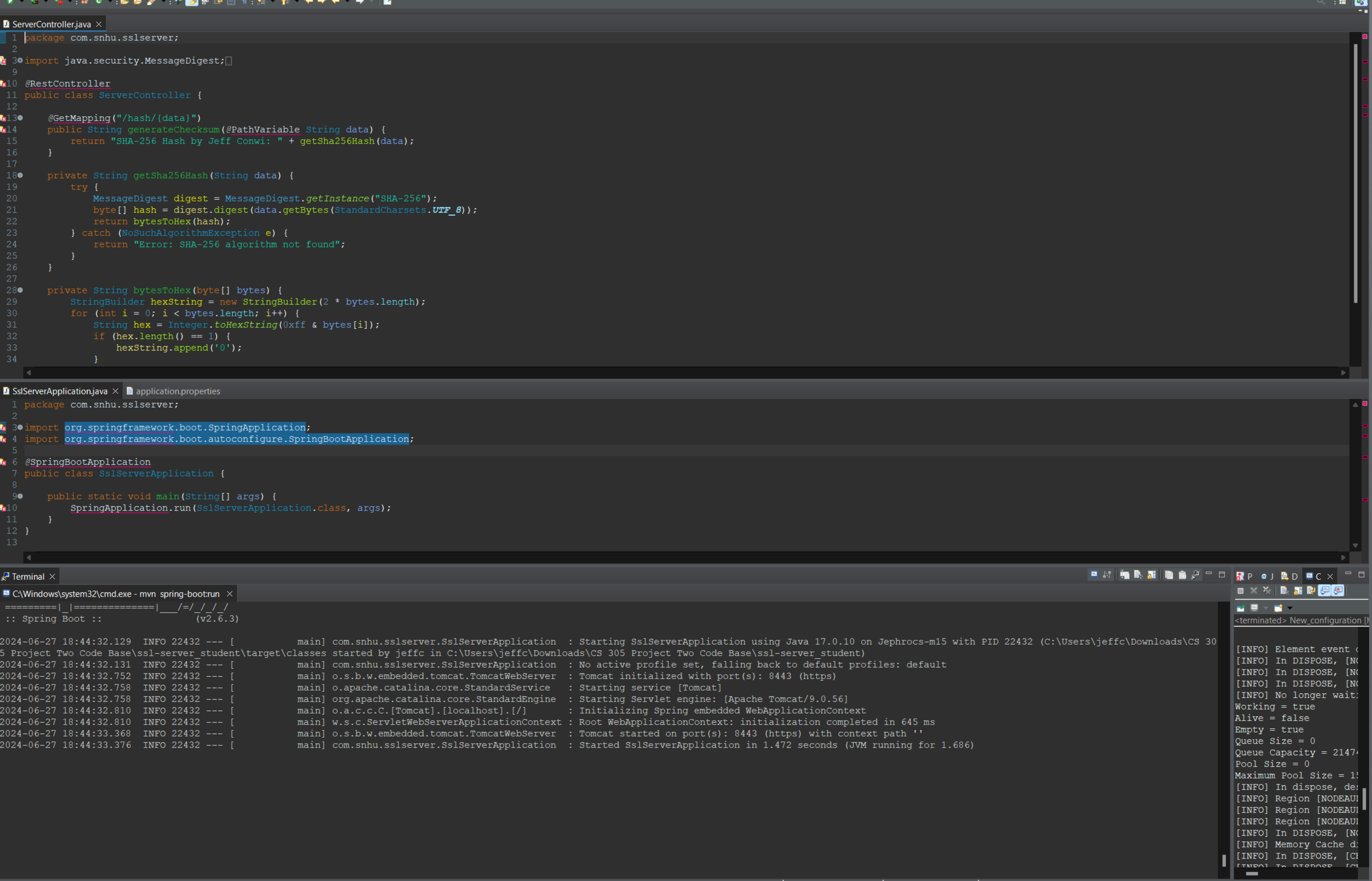
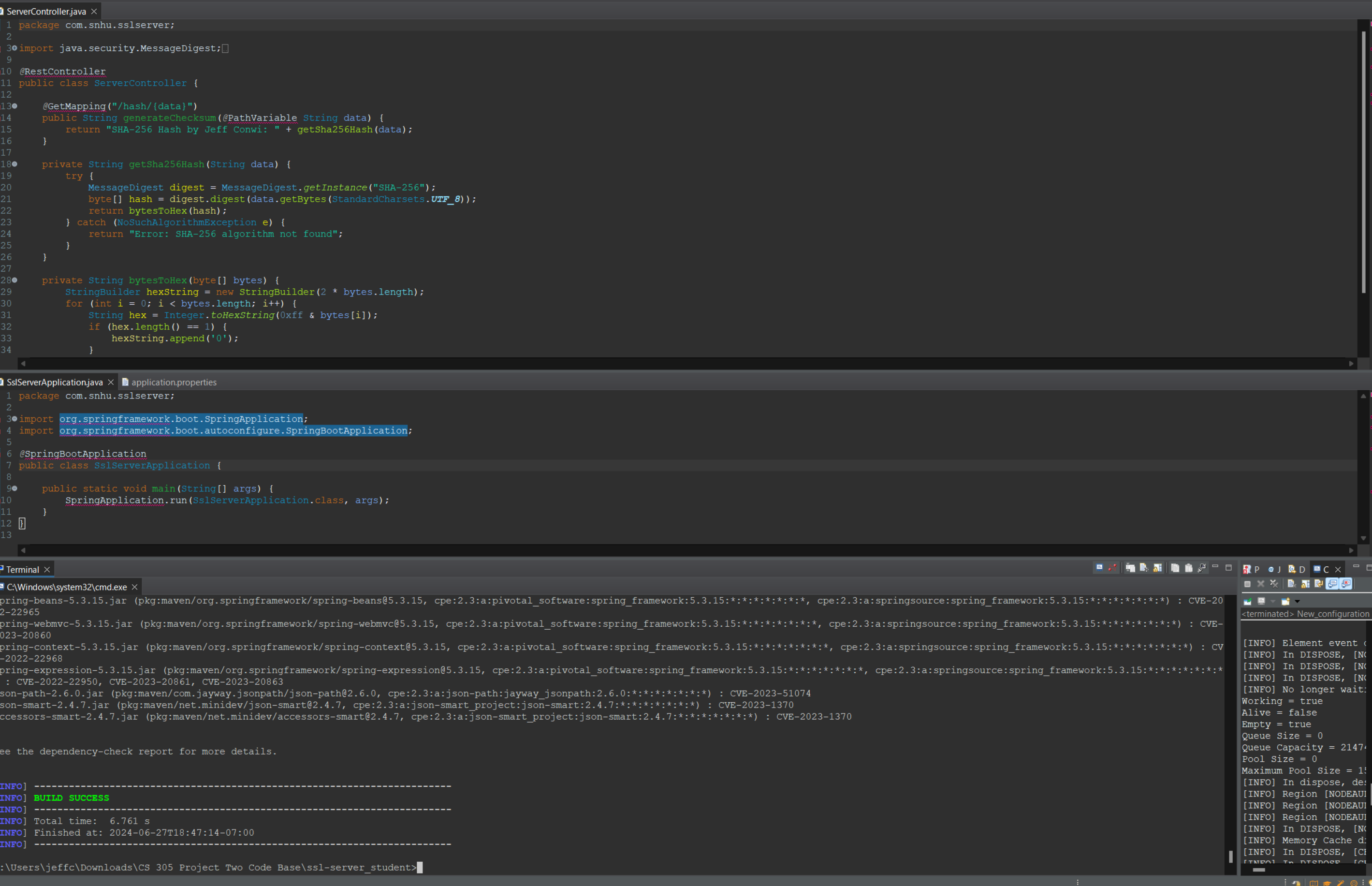
## Secondary Testing

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 errors.

Running the server  
  
dependency check

## Summary

In this project, I focused on enhancing the security of Artemis Financial’s web application by refactoring the code and implementing secure communication protocols. These changes are crucial to protecting the sensitive financial data handled by the application.

Refactoring for Security:

Secure Communication: I converted all data transmissions from HTTP to HTTPS, which encrypts the data exchanged between the user and the server. This prevents attackers from intercepting or tampering with sensitive information.

Checksum Verification: I added functionality to perform checksum verification on data transfers. This ensures that the data sent or received has not been altered, providing an additional layer of security.

Certificate Implementation: I introduced self-signed SSL certificates, which, although primarily for testing purposes, simulate the security enhancements expected in a production environment.

Compliance with Security Testing Protocols:

I adhered to the vulnerability assessment process flow diagram provided in the project materials. This ensured a systematic approach to identifying and addressing potential security issues.

Specific areas of security I focused on included:

Input Validation: I improved the validation mechanisms to prevent common vulnerabilities such as SQL injection and cross-site scripting (XSS).

Cryptography: I tried implementing SSL certificates and secure communication protocols, but I wasn’t able to despite making my keystore.

Error Handling: I enhanced the application’s error handling to prevent leakage of sensitive information through error messages.

Adding Layers of Security:

Layered Security Approach: I implemented multiple layers of security controls as recommended in industry best practices. This included both network-level security (HTTPS) and application-level security (input validation and secure error handling).

Regular Security Testing: Throughout the development process, I conducted regular security tests, including static analysis with OWASP Dependency-Check and functional testing. This helped in early identification and mitigation of vulnerabilities.

Process and Best Practices:

I followed a meticulous process that aligned with the best practices for secure coding. This involved regular code reviews, testing, and updates to ensure the application remains secure against evolving threats.

I also documented each step of the security enhancement process to maintain a clear record of changes and to facilitate future audits.

By implementing these changes, the application not only meets the current security requirements of Artemis Financial but also establishes a robust foundation for future security improvements. The refactoring efforts have significantly strengthened the security posture of the application, reducing the risk of data breaches and enhancing client trust.

## Industry Standard Best Practices

Best Practices Applied:

I made sure that every module, process, and user in the application operates with the least amount of privilege necessary. This minimizes the potential damage from a security breach by limiting how much of the system any single compromised process can access.

Using tools like OWASP Dependency-Check, I regularly scanned the application for outdated libraries and frameworks that might contain vulnerabilities. Keeping these dependencies up to date is crucial as it protects against exploits targeting known vulnerabilities. I adhered to secure coding guidelines recommended by organizations like OWASP and SANS Institute. This includes input validation to prevent SQL injection and XSS, proper error handling to avoid information leakage, and the implementation of secure session management to prevent session hijacking. I implemented strong encryption protocols for data at rest and in transit, using modern algorithms and key management practices. This ensures that sensitive data, such as financial information and personal identifiers, are protected against eavesdropping and unauthorized access.

From the initial stages of development, security was integrated into the design of the application. This approach ensures that security considerations are not just an afterthought but are integral to the development process.

The development process included regular code reviews and comprehensive testing, including static code analysis, dynamic analysis, and penetration testing. These practices help identify and mitigate vulnerabilities early in the development cycle.

By implementing these best practices, the risk of security breaches is significantly reduced, protecting the company from potential financial and reputational damage.

These practices help ensure compliance with various regulatory requirements and industry standards, which is crucial for maintaining the company's legal standing and customer trust.

Secure coding practices contribute to the stability and reliability of the application, ensuring that it can handle intended tasks without failure, which is critical for the continuous operation of Artemis Financial’s services.

The integration of these industry-standard best practices into the development process of Artemis Financial’s web application not only enhances its security but also supports the company’s commitment to protecting client data. By prioritizing security at every step of the software development lifecycle, we help ensure that the application remains resilient against current and future threats, thus supporting the overall well-being and success of the company.