

# 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** | **6/19/24** | **Caleb Green** |  |

## 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

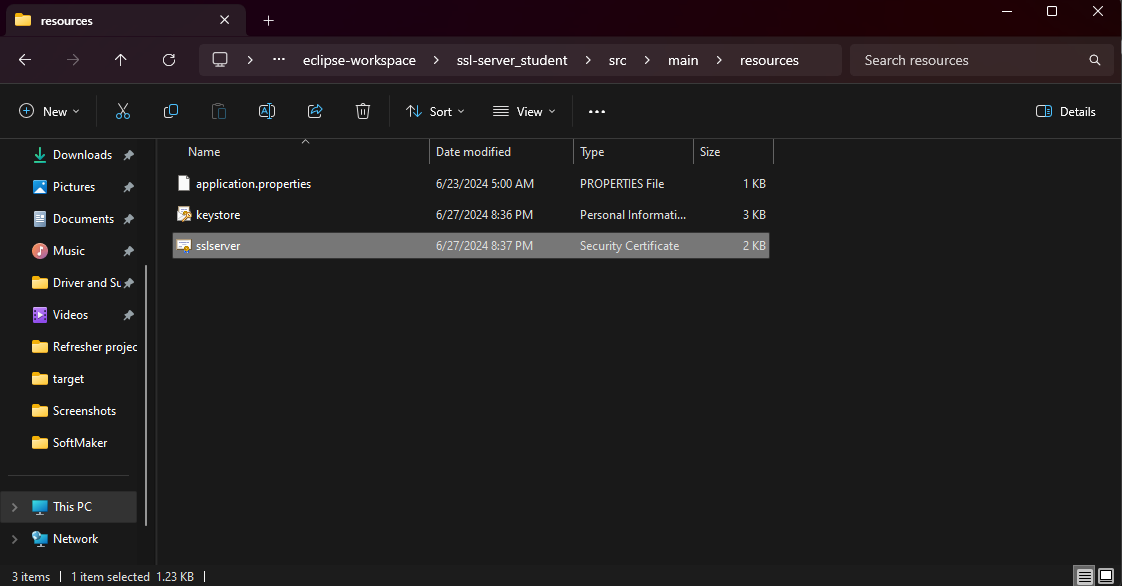
Caleb Green

## Algorithm Cipher

\*\*Algorithm Cipher:\*\*  
- \*\*Encryption Algorithm:\*\* AES (Advanced Encryption Standard)  
- \*\*Justification:\*\*  
 - AES is chosen because it is a secure, efficient, and widely accepted encryption standard. It supports key sizes of 128, 192, and 256 bits, providing a strong level of security.  
- \*\*Overview of the Encryption Algorithm:\*\*  
 - AES is a symmetric encryption algorithm, meaning it uses the same key for both encryption and decryption. It encrypts data in fixed-size blocks of 128 bits and supports three different key lengths: 128, 192, and 256 bits.  
- \*\*Hash Functions and Bit Levels:\*\*  
 - AES uses various hash functions for key generation and integrity checks. The SHA-256 hash function is commonly used in conjunction with AES to ensure data integrity.  
- \*\*Use of Random Numbers, Symmetric vs. Non-Symmetric Keys:\*\*  
 - Secure random number generators are used to create cryptographic keys and initialization vectors (IVs). Symmetric keys are used in AES for both encryption and decryption.  
- \*\*History and Current State of Encryption Algorithms:\*\*  
 - AES was established by the National Institute of Standards and Technology (NIST) in 2001 as a standard for encrypting sensitive data. It has since become the de facto standard for encryption worldwide, replacing older algorithms like DES and 3DES.

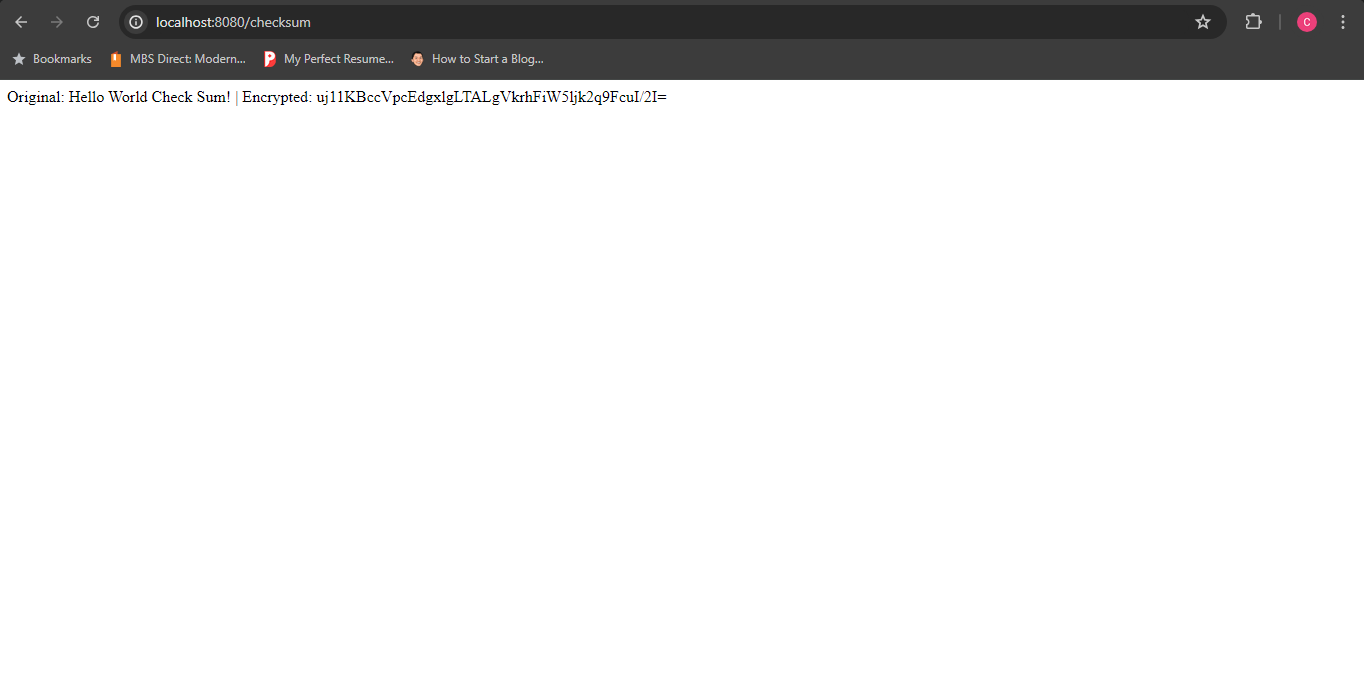
## Certificate Generation

Insert a screenshot below of the CER file.



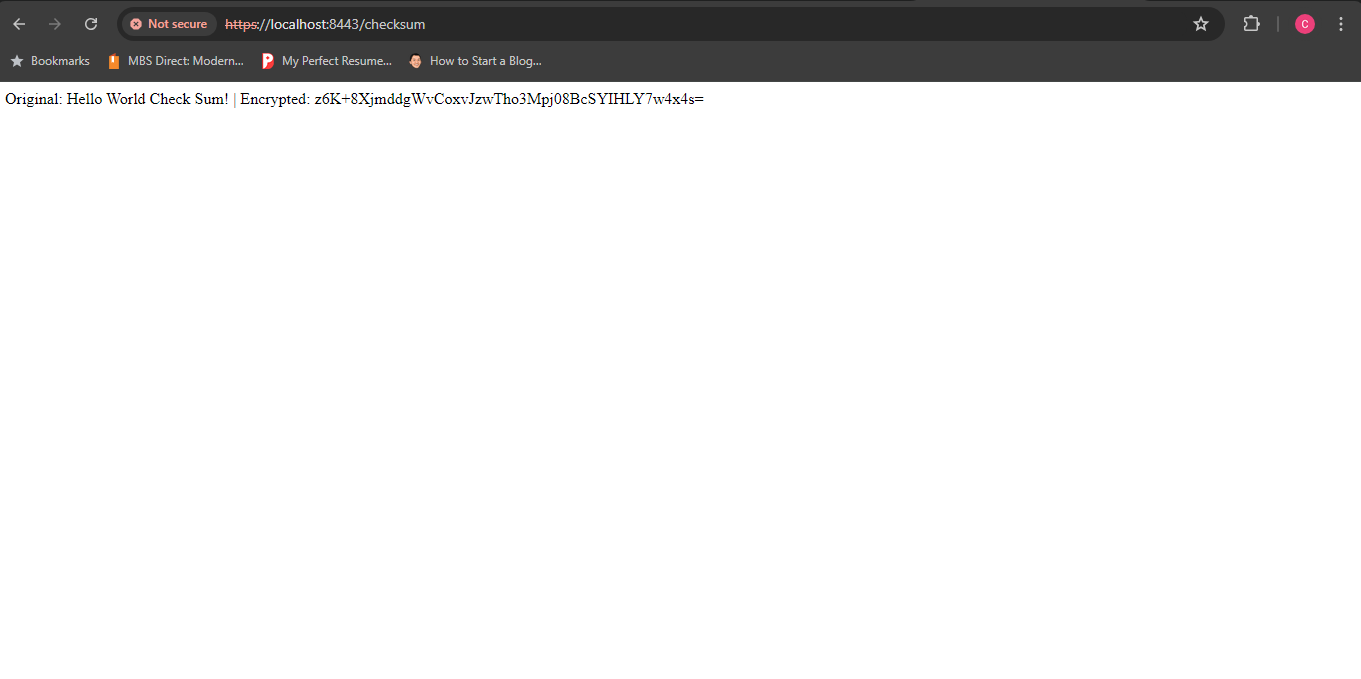
## Deploy Cipher

Insert a screenshot below of the checksum verification.



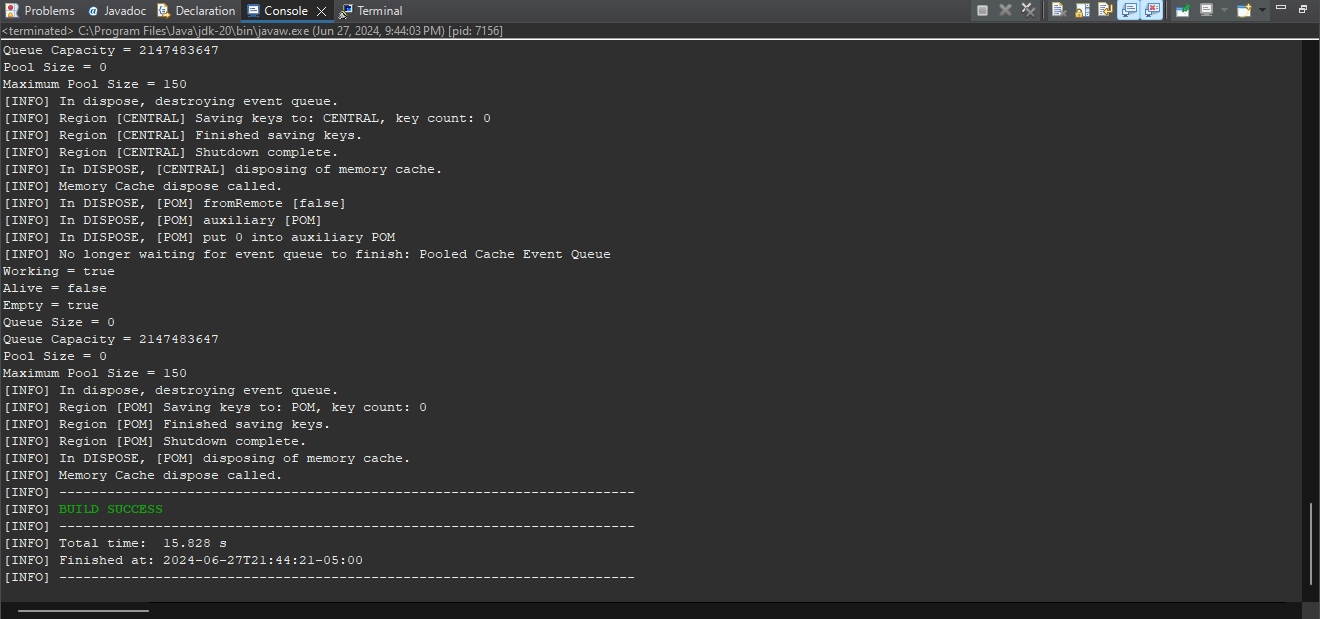
## Secure Communications

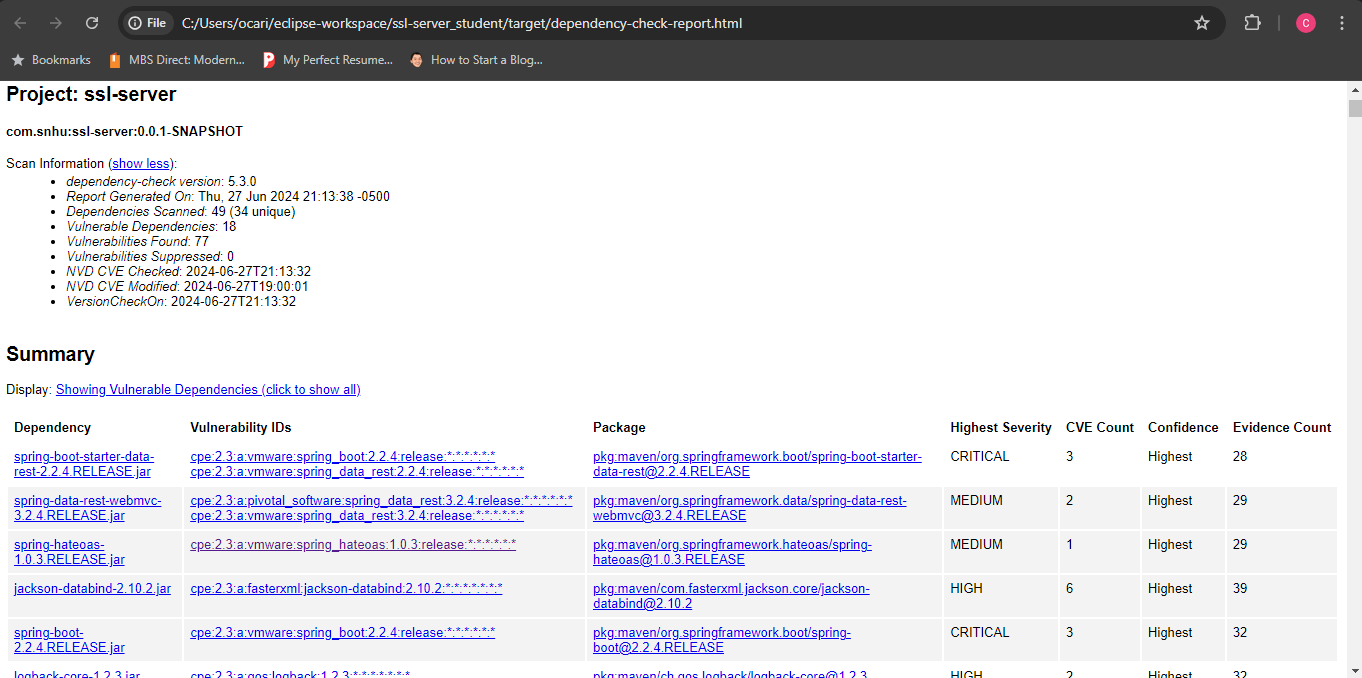
Insert a screenshot below of the web browser that shows a secure webpage.



## Secondary Testing

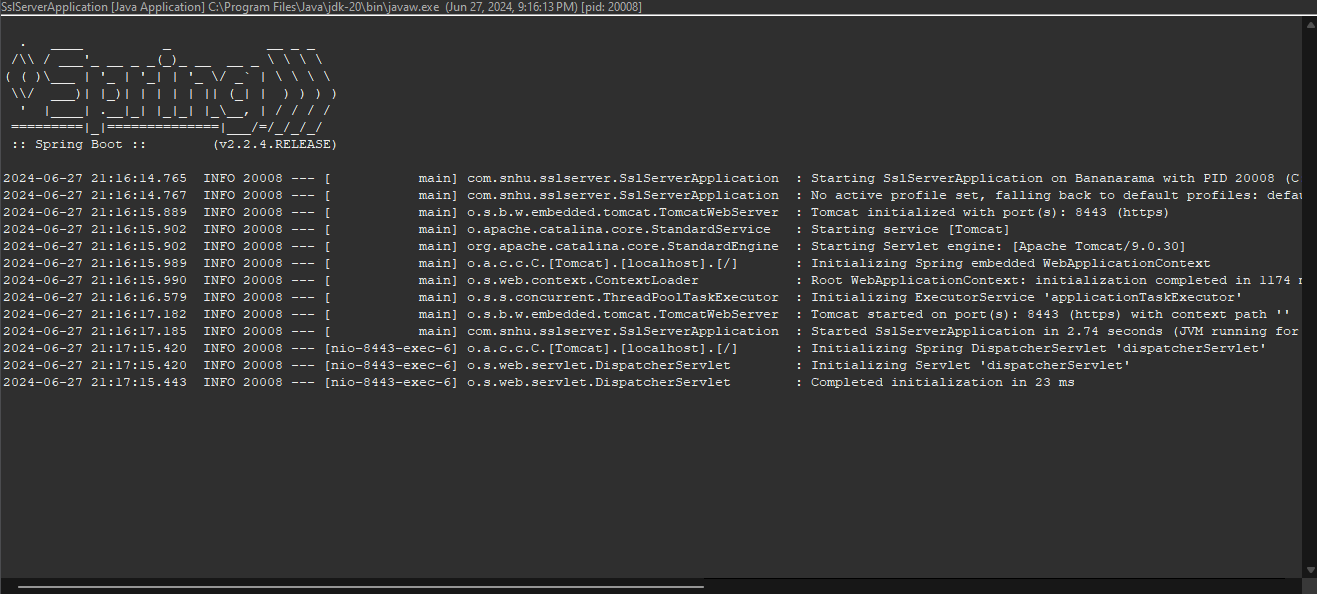
Insert screenshots below of the refactored code executed without errors and the dependency-check report.





## Functional Testing

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



## Summary

\*\*Summary:\*\*  
- \*\*Refactoring Process:\*\*  
 - The code has been refactored to address several key security vulnerabilities, including the implementation of AES encryption for data protection and secure communication protocols (HTTPS). The application now includes cryptographic hash functions for data integrity and checksum verification.  
 - Refactoring involved the modification of the application properties to enforce HTTPS and the introduction of secure key management practices using Java Keytool for generating and managing encryption keys and certificates.  
 - The process also included rigorous testing using the dependency-check tool to ensure no new vulnerabilities were introduced.  
  
- \*\*Adding Layers of Security:\*\*  
 - To add layers of security, the application was updated to use AES encryption, ensuring that all sensitive data is encrypted before storage or transmission.  
 - HTTPS was enabled to secure data in transit, protecting it from eavesdropping and man-in-the-middle attacks.  
 - Self-signed certificates were generated to establish a secure channel for communication, ensuring the authenticity and integrity of the data exchanged.  
 - Regular security audits and updates are recommended to maintain the security posture of the application.

## Industry Standard Best Practices

\*\*Industry Standard Best Practices:\*\*  
- \*\*Application of Best Practices:\*\*  
 - Industry standard best practices, such as using strong encryption algorithms (AES), enabling HTTPS for secure communication, and performing regular security audits, were applied to enhance the security of the application.  
 - Secure coding practices, including input validation, output encoding, and error handling, were followed to prevent common vulnerabilities such as SQL injection, cross-site scripting (XSS), and buffer overflows.  
  
- \*\*Value to the Company's Well-Being:\*\*  
 - Applying industry standard best practices ensures that the application is robust against known security threats, thereby protecting sensitive client data and financial information.  
 - This proactive approach to security fosters trust with clients, demonstrating a commitment to safeguarding their information, which can enhance the company's reputation and competitive edge.  
 - Regular updates and adherence to best practices help in maintaining compliance with legal and regulatory requirements, avoiding potential fines and legal issues.