

# 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** | **10/15/2025** | **Zack A. Lodi** |  |

## 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

Zack A. Lodi

## Algorithm Cipher

Provide a brief, high-level overview of the encryption algorithm cipher.

The Advanced Encryption Standard (AES) is a strong and reliable encryption method that is widely used today. It is a symmetric block cipher, which means it uses the same key to both encrypt and decrypt data. AES works with 128-bit blocks and supports key sizes of 128, 192, or 256 bits. Because of its efficiency and security, AES has become the standard for protecting sensitive information in industries like finance and government.

Discuss the hash functions and bit levels of the cipher.

AES is often used along with hash functions such as SHA-256 or SHA-512 to make sure data has not been changed. A hash function takes data of any size and produces a fixed-length output (called a digest). SHA-256 creates a 256-bit digest, while SHA-512 creates a 512-bit digest. The longer the bit length, the harder it is for attackers to find two different inputs that create the same hash. Using AES with a secure hash function provides both confidentiality and integrity for the data.

Explain the use of random numbers, symmetric versus non-symmetric keys, and so on.

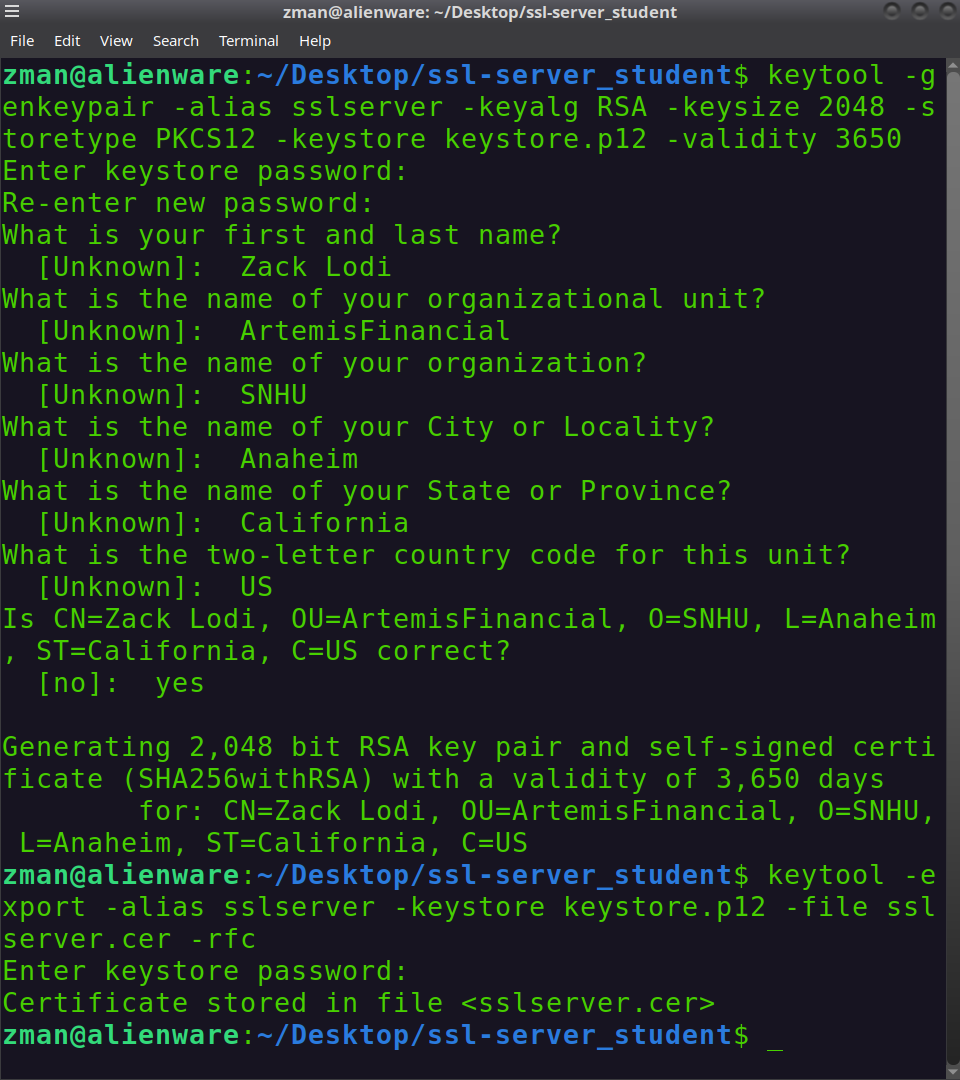
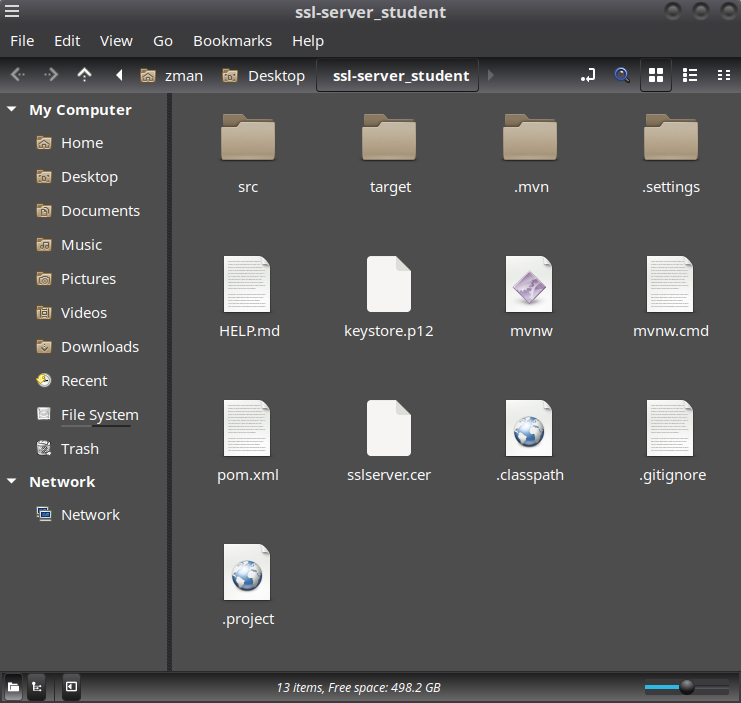
Random numbers are important in encryption because they make keys and initialization values unpredictable. This prevents attackers from guessing or finding patterns. AES is a symmetric algorithm, so the same key is used for both encryption and decryption. This makes it fast and effective for large amounts of data, but it also means the key must be shared securely. To solve this, asymmetric encryption methods like RSA are often used to exchange the symmetric key safely. Asymmetric encryption uses a public key to encrypt and a private key to decrypt.

Describe the history and current state of encryption algorithms.

Encryption has been used for centuries, starting with simple substitution ciphers like Caesar’s cipher. During World War II, machines like the Enigma were used to create more complex codes. In the 1970s, the Data Encryption Standard (DES) became widely used, but it was eventually found to be insecure because of its short key length. AES replaced DES in 2001 and has been the standard ever since. Today, AES is trusted around the world and continues to be tested against new threats, making it the best choice for protecting sensitive data.

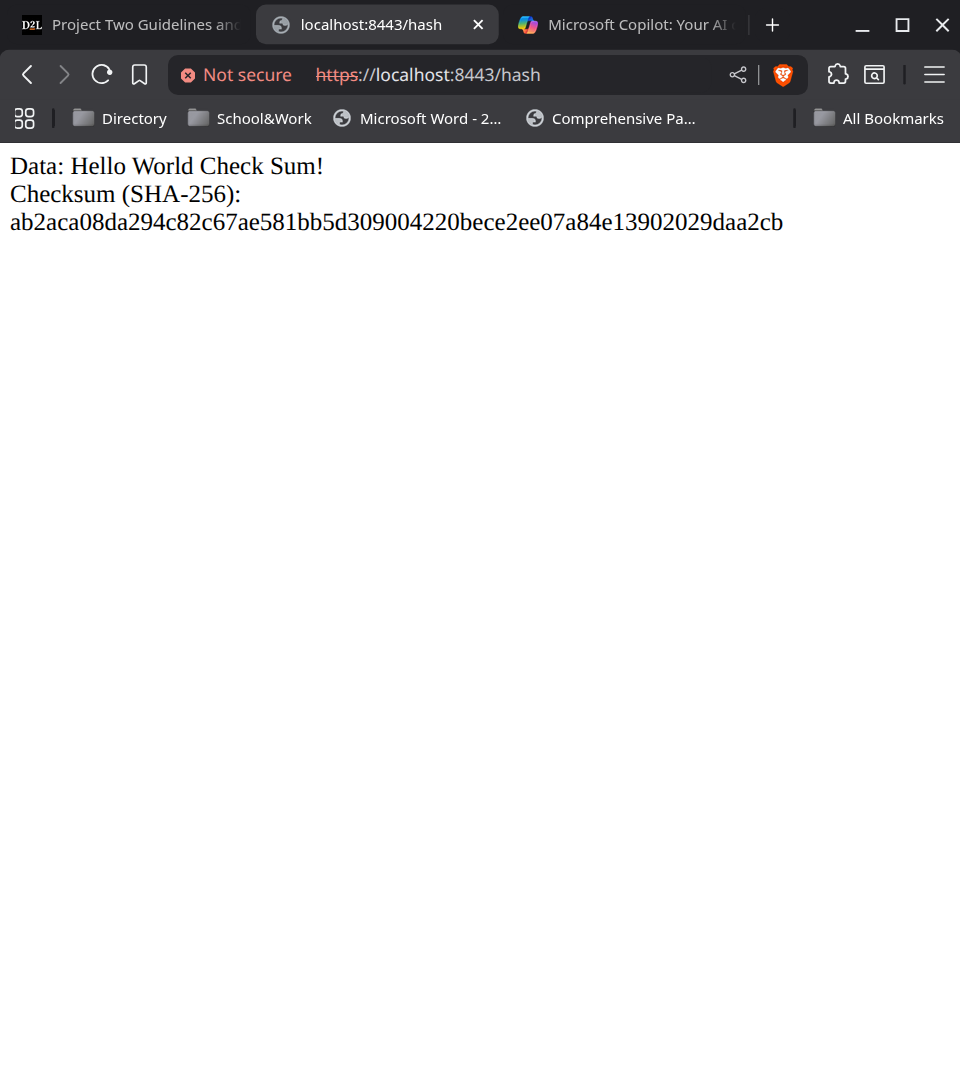
## Certificate Generation

Insert a screenshot below of the CER file.



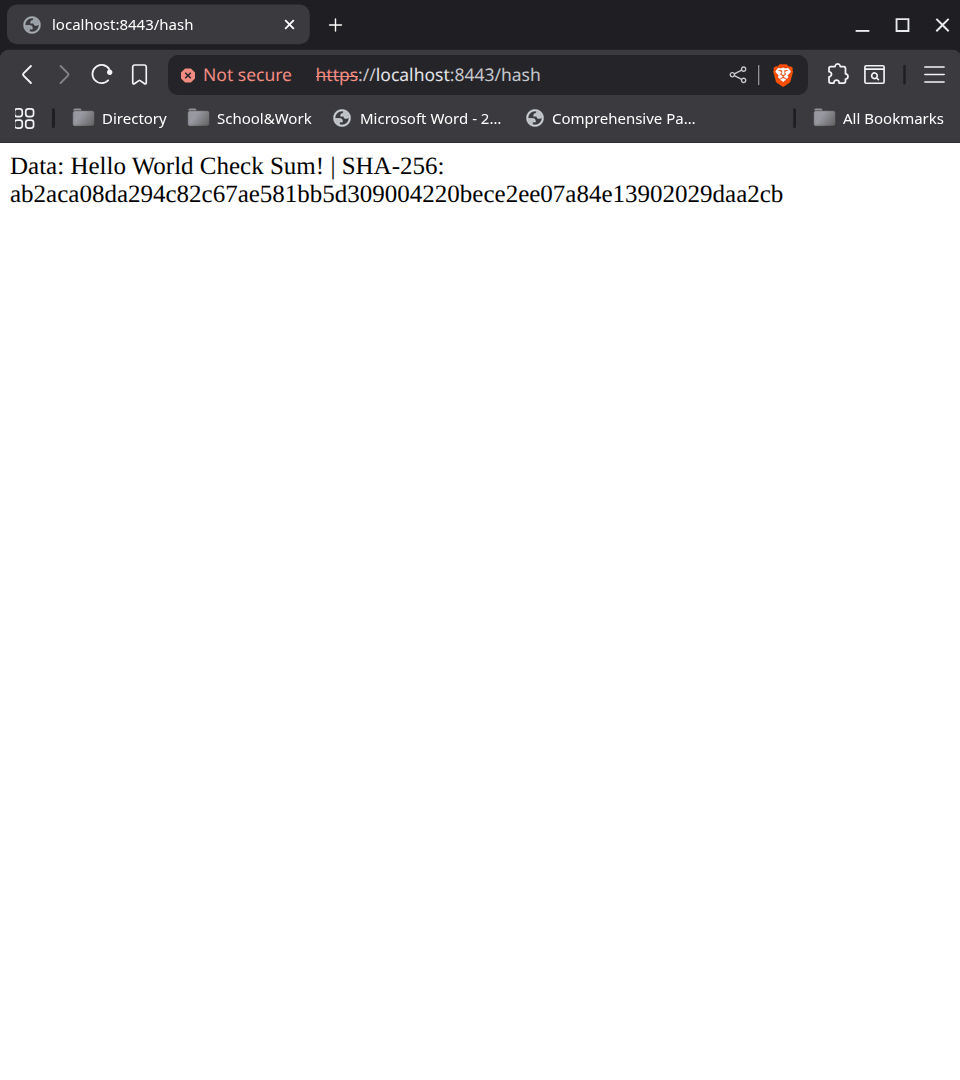
## Deploy Cipher

Insert a screenshot below of the checksum verification.



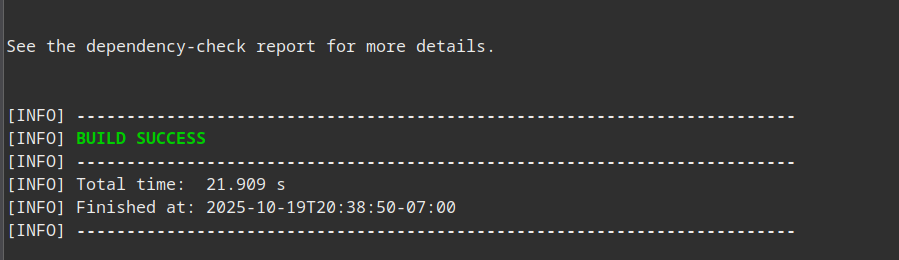
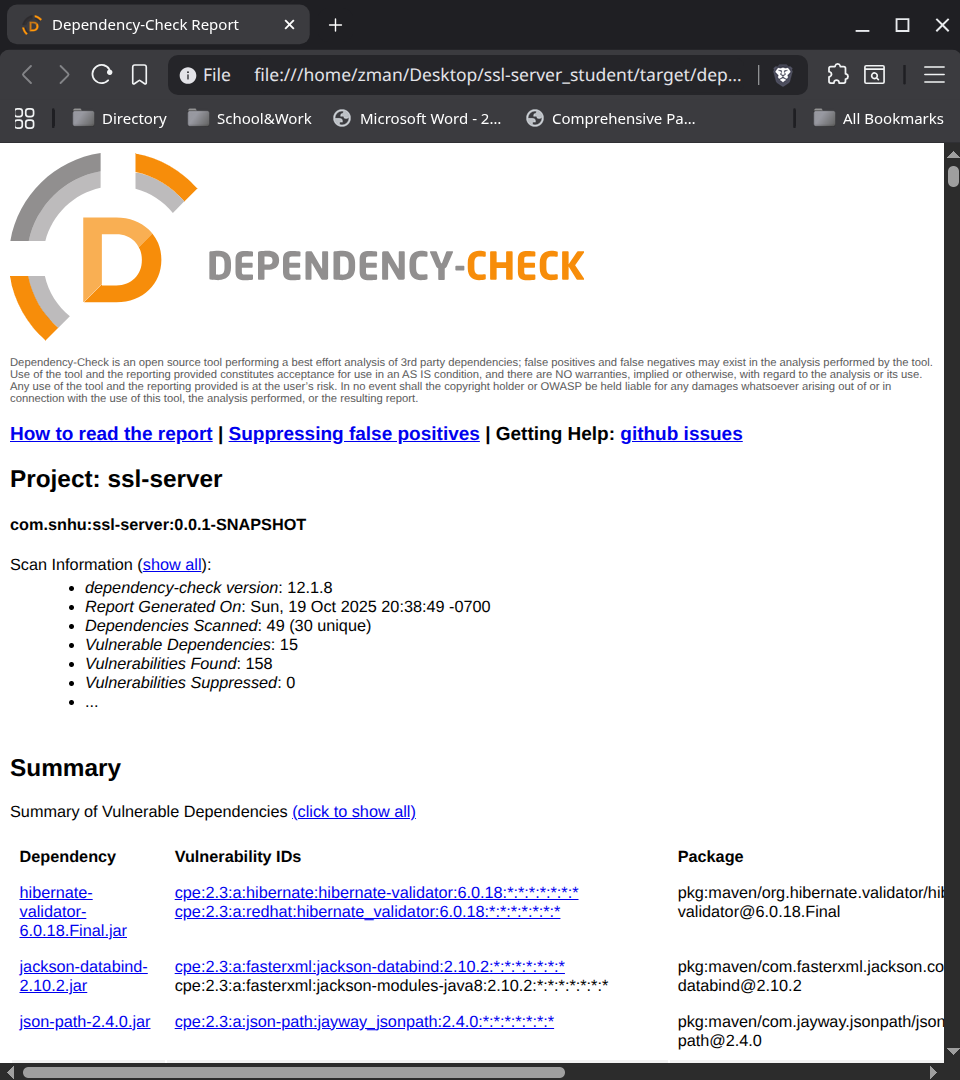
## Secure Communications

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



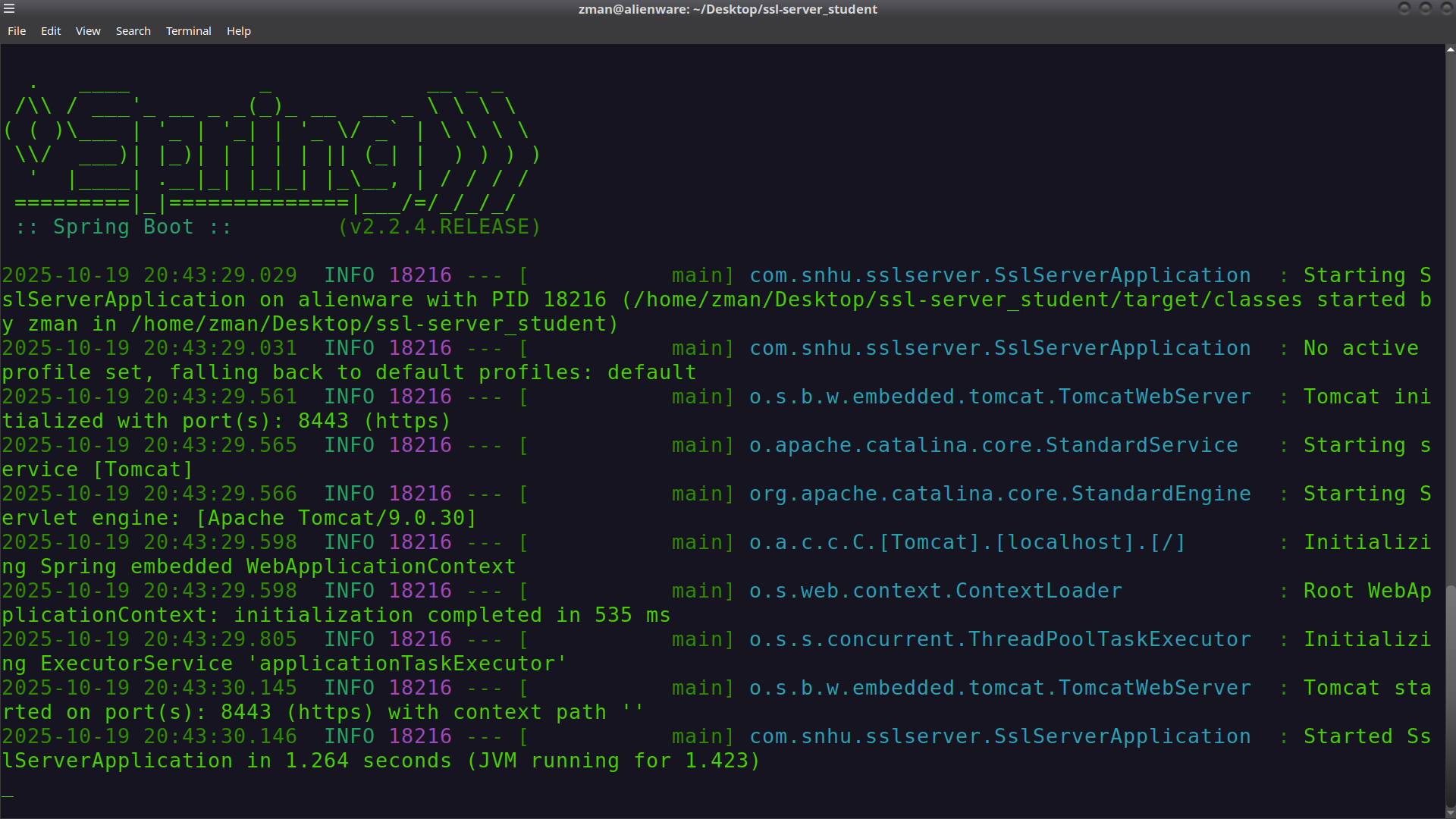
## Secondary Testing

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

- Refer to the vulnerability assessment process flow diagram in the Supporting Materials section. Highlight the areas of security that you addressed by refactoring the code.

Using the vulnerability assessment process flow diagram as a guide, I focused on the areas cryptography, client/server communication, and code quality when refactoring the Artemis Financial code base. I replaced HTTP with HTTPS to secure communication between the client and server, which directly addresses the client/server security step in the diagram. I also added checksum verification with a secure hash algorithm to confirm that files are not changed during transfer, which ties into the cryptography stage. Finally, I ran dependency checks to identify and update libraries with known issues, improving overall code quality and reducing risks from outdated components.

- Discuss your process for adding layers of security to the software application.

My process for adding layers of security followed the “defense-in-depth” approach shown in the diagram. I started with secure input and representation by validating data, then moved to cryptography by adding hashing for integrity checks. I also improved client/server security by enabling SSL/TLS (which prevents eavesdropping). On the code side, I used static analysis and dependency checks to catch errors early, which connects to the code error and code quality steps in the diagram. By layering these protections, the application now has multiple defenses that work together to reduce the chance of a successful attack.

## Industry Standard Best Practices

- Explain how you used industry standard best practices to maintain the software application’s existing security.

To maintain the application’s security, I followed common best practices such as secure coding standards, input validation, and proper error handling. I avoided hard-coding sensitive information like keys and certificates, and I made sure user input was checked to prevent injection attacks. I also added the Maven dependency-check plug-in to monitor third-party libraries for known vulnerabilities. These steps help keep the application secure over time and reduce the chance of introducing new risks.

- Explain the value of applying industry standard best practices for secure coding to the company’s overall well-being.

Following secure coding best practices is important for the company’s long-term success. Strong security protects client trust, which is especially important in financial services. It also helps avoid the costs and damage that come with data breaches, such as fines, lawsuits, and loss of reputation. By building security into the development process, Global Rain shows its commitment to Artemis Financial’s mission and ensures the software is safe, reliable, and competitive.