****

# 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** | **12/12/2024** | **Payton Castle** | **Project 2** |

## 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

Payton Castle

## Algorithm Cipher

To enhance security, the SHA 256 cryptographic hash algorithm was implemented. This algorithm generates a secure checksum to verify data integrity. Key points include:

* **Algorithm Overview:** SHA 256 belongs to the SHA 2 family, offering robust 256 bit encryption.
* **Randomness:** Uses deterministic cryptographic hashing, ensuring strong protection against tampering.
* **Key Features:** Provides reliable data integrity checks with minimal risk of collision or compromise.

This implementation strengthens the application’s ability to verify data authenticity.

## Certificate Generation

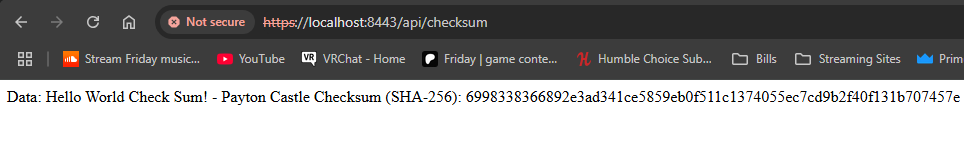
Insert a screenshot below of the CER file.

A screenshot of a computer

Description automatically generated

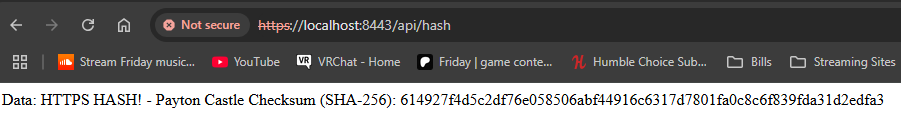
## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

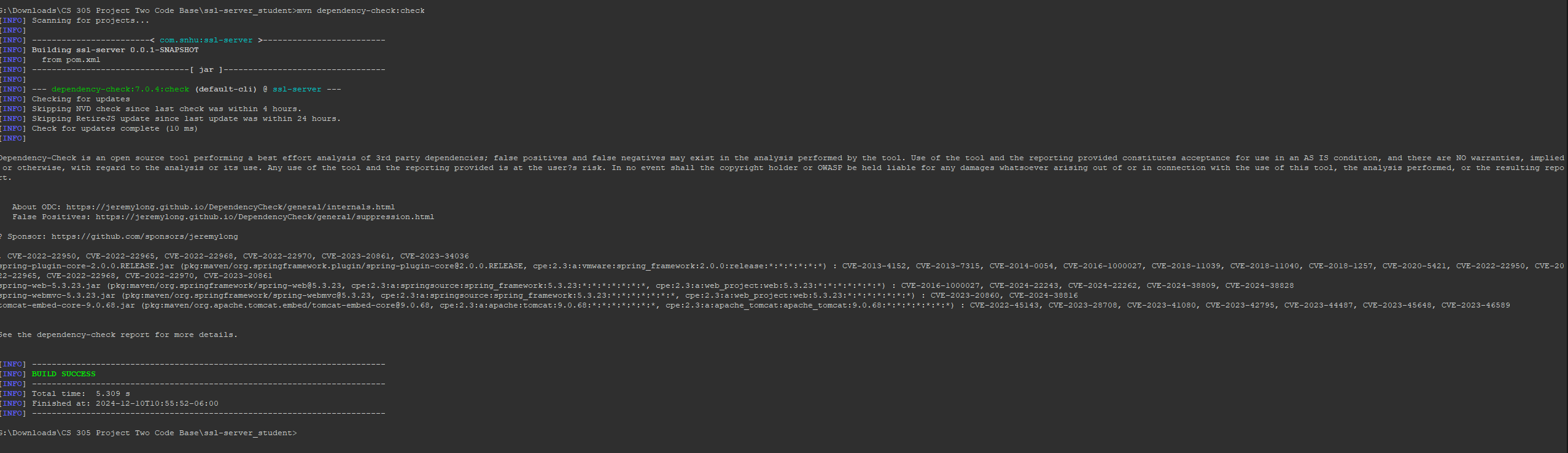
Insert a screenshot below of the web browser that shows a secure webpage.



[Insert screenshots here.]

## Secondary Testing

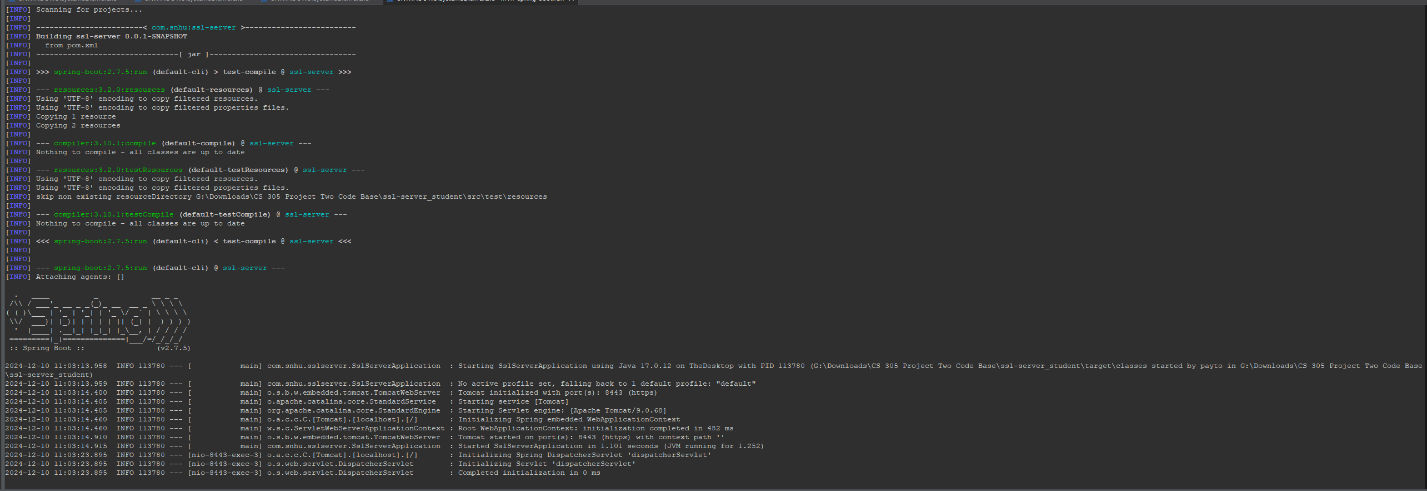
Insert screenshots below of the refactored code executed without errors and the dependency-check report.





## Functional Testing

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



[Insert screenshots here.]

## Summary

The following security improvements were made to Artemis Financial’s application:

1. Added checksum generation using the SHA 256 algorithm to verify data integrity.
2. Enabled secure HTTPS communication by configuring an SSL keystore.
3. Conducted secondary static and functional testing to identify and address vulnerabilities.
4. Managed false positives during dependency checks with a suppression file.

These measures improve the application’s security, ensuring compliance with best practices and protecting sensitive client data from potential threats.

## Industry Standard Best Practices

* **Secure Communications:** Enforced HTTPS for encrypted data transmission.
* **Data Integrity:** Integrated SHA 256 to validate the integrity of transmitted data.
* **Static Testing:** Conducted comprehensive dependency checks to identify vulnerabilities.
* **Code Refactoring:** Improved overall code quality while maintaining functionality.

By using best practices, Artemis Financial can maintain a secure and reliable software application that proves it can be trusted.