

# 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** | **2024-06-21** | **Eric Nagley** |  |

## 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

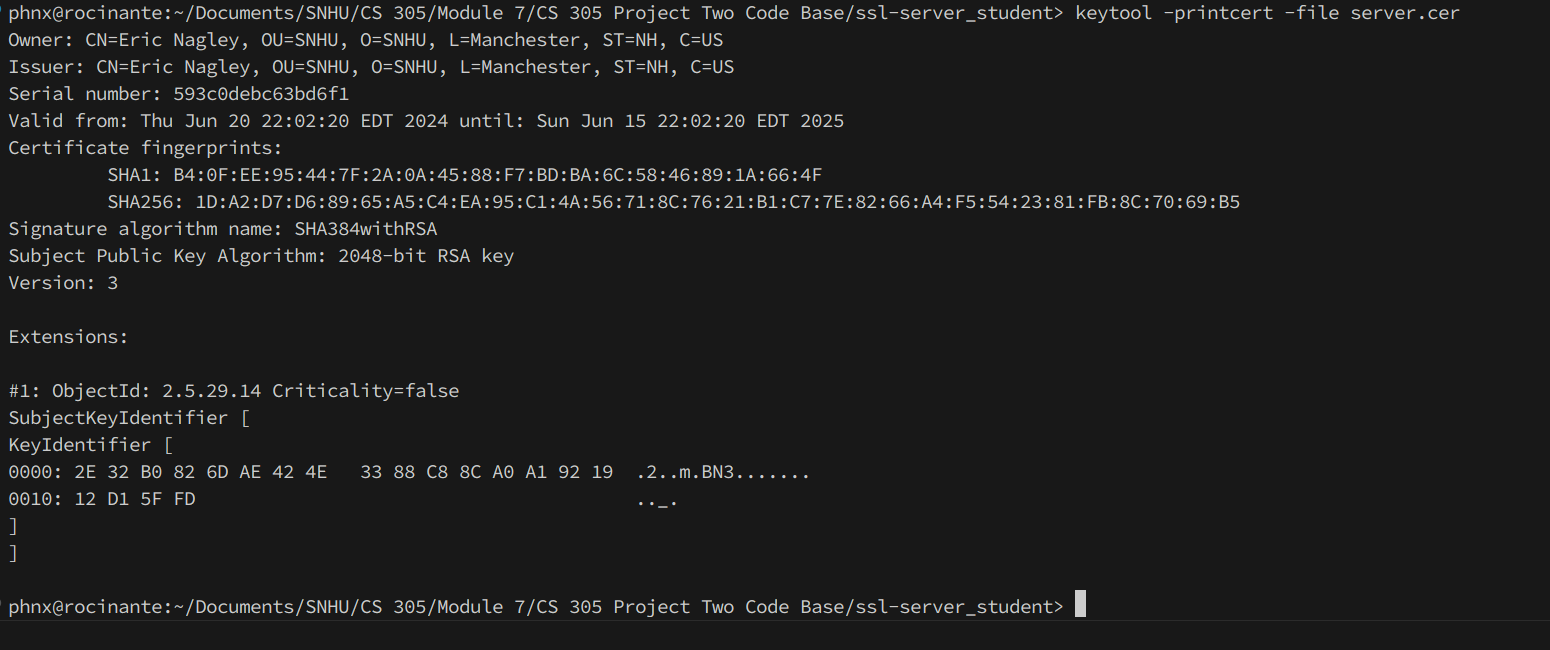
Eric Nagley

## Algorithm Cipher

SHA-256

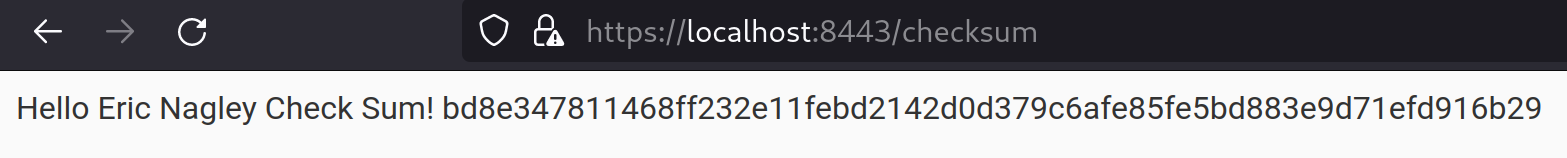
## Certificate Generation

Insert a screenshot below of the CER file.



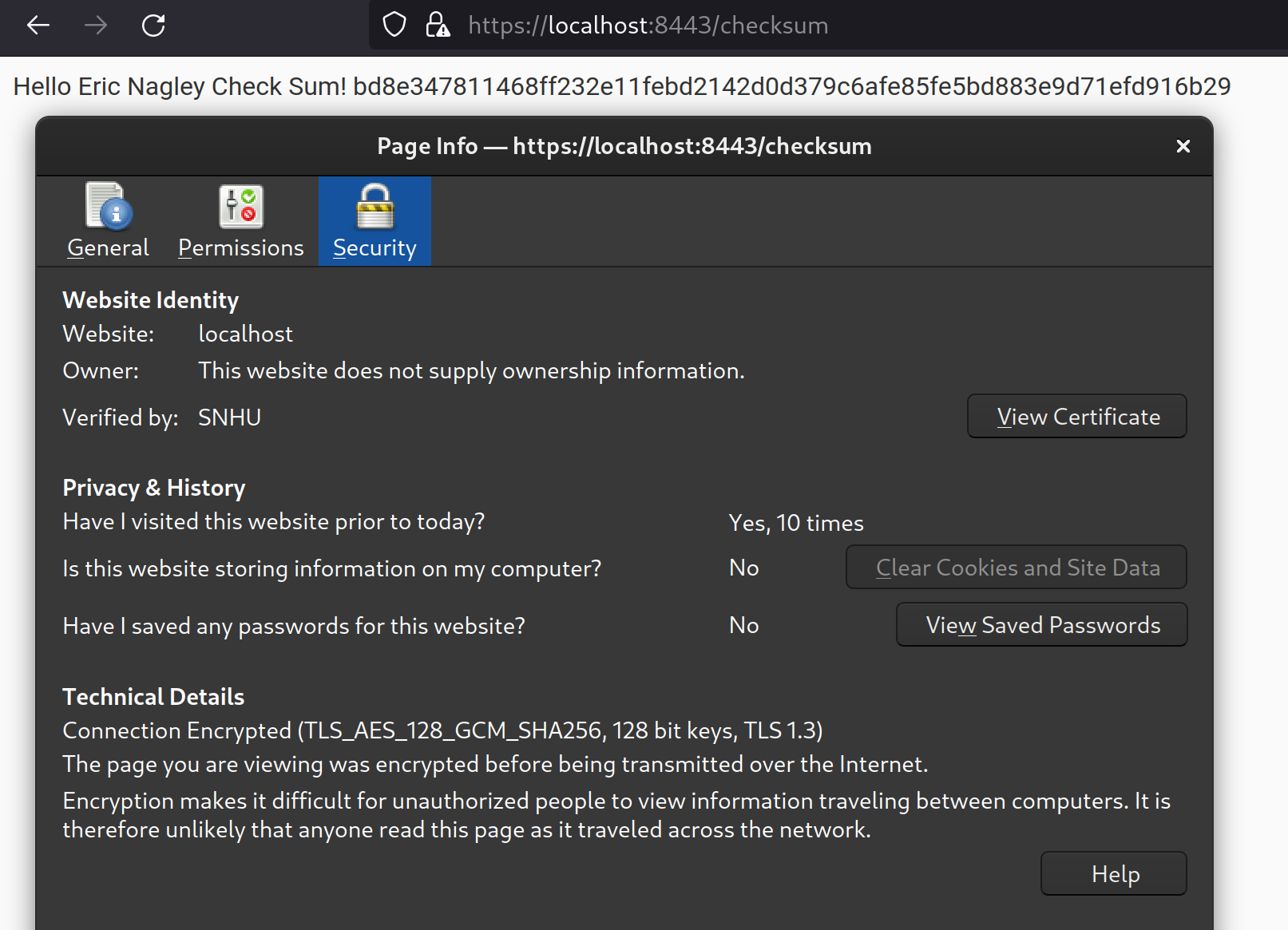
## Deploy Cipher

Insert a screenshot below of the checksum verification.



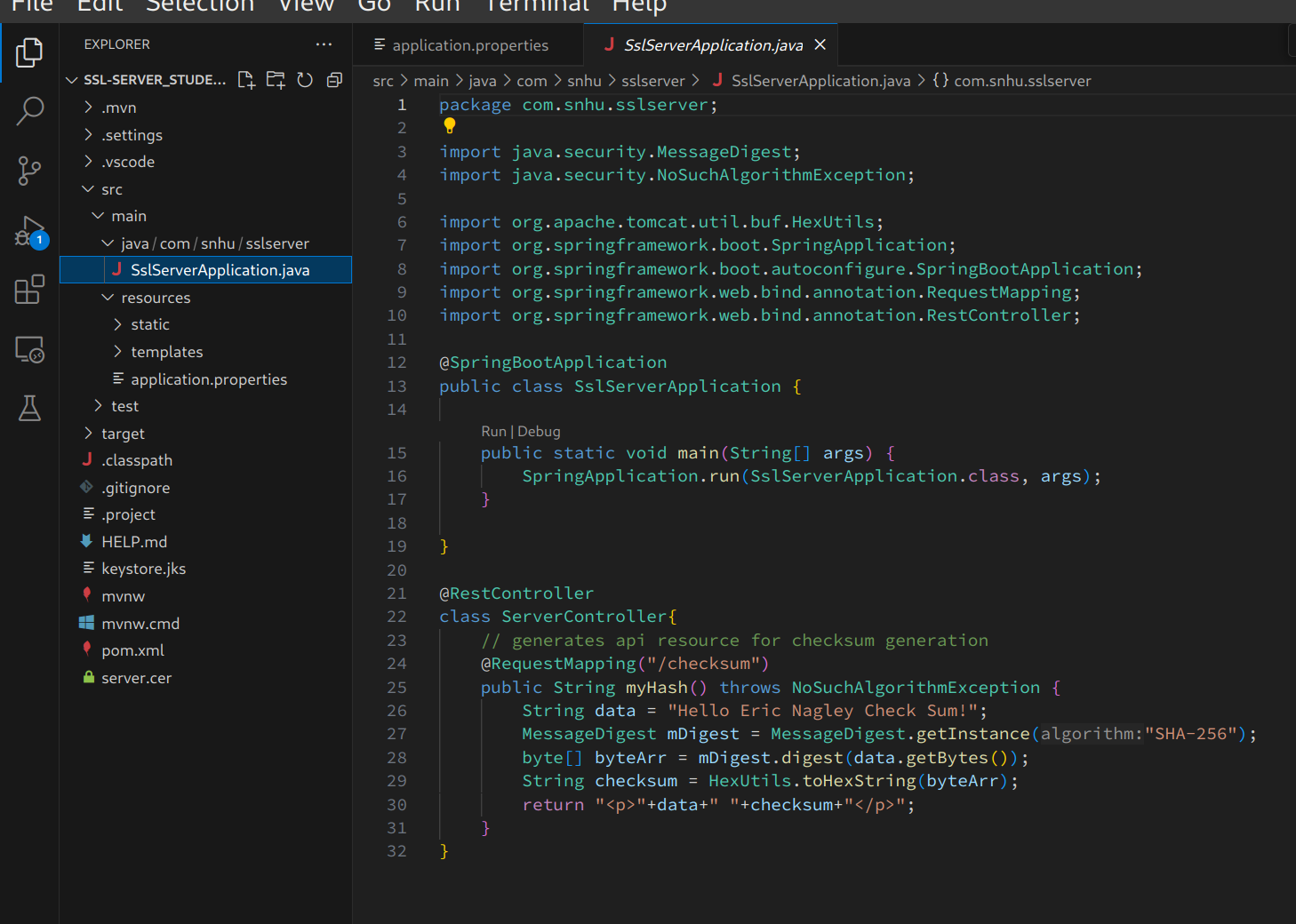
## Secure Communications

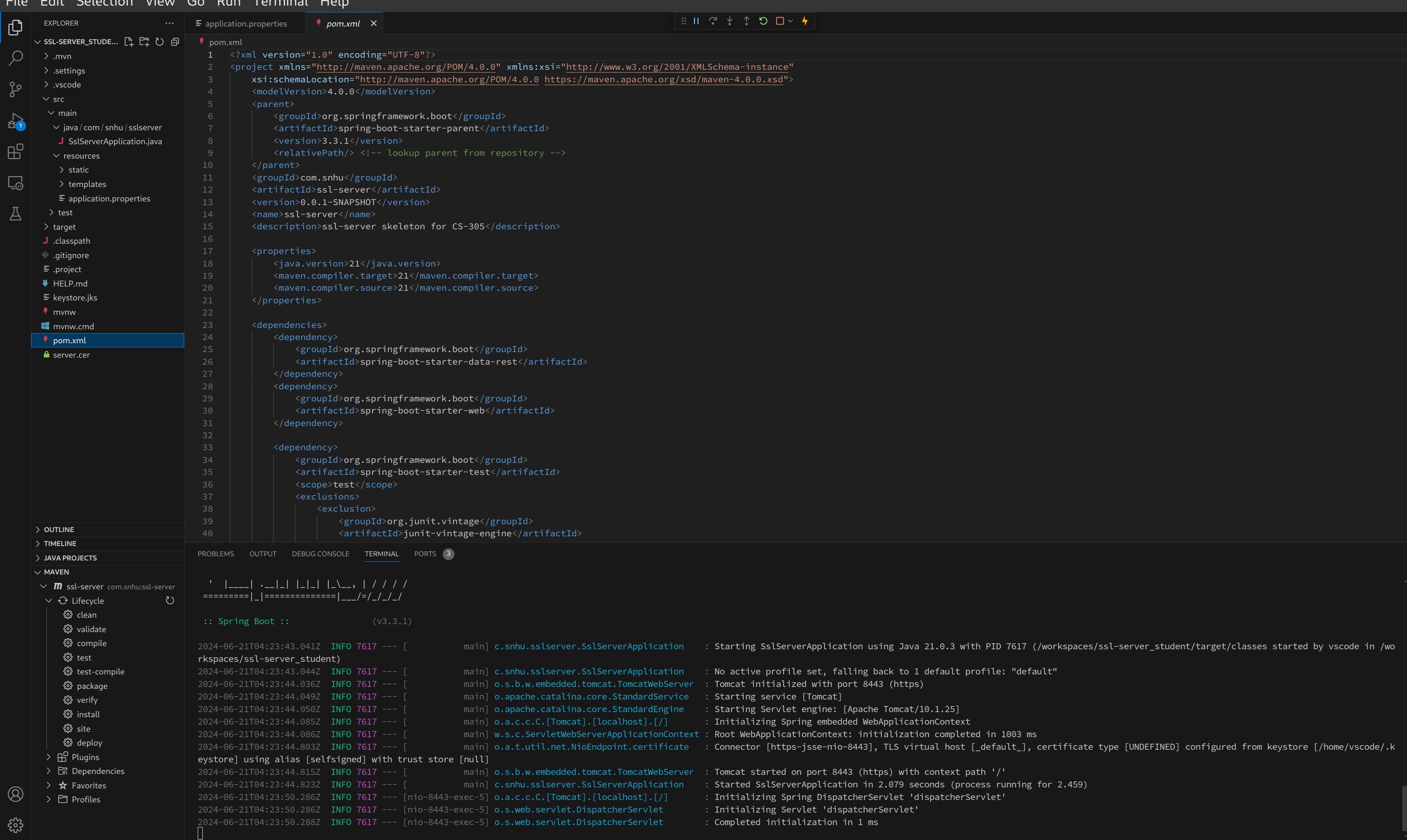
Insert a screenshot below of the web browser that shows a secure webpage.

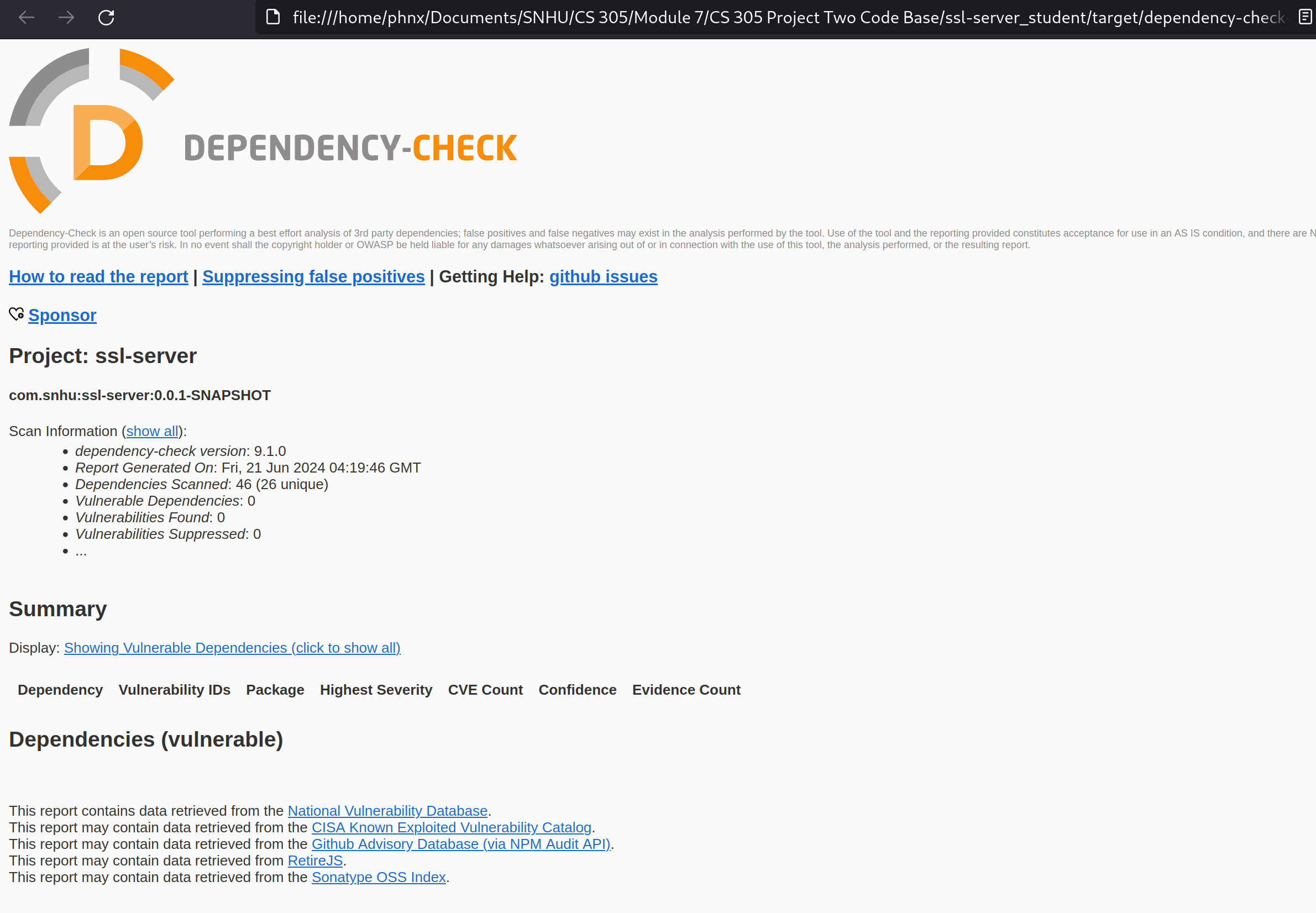


## Secondary Testing

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

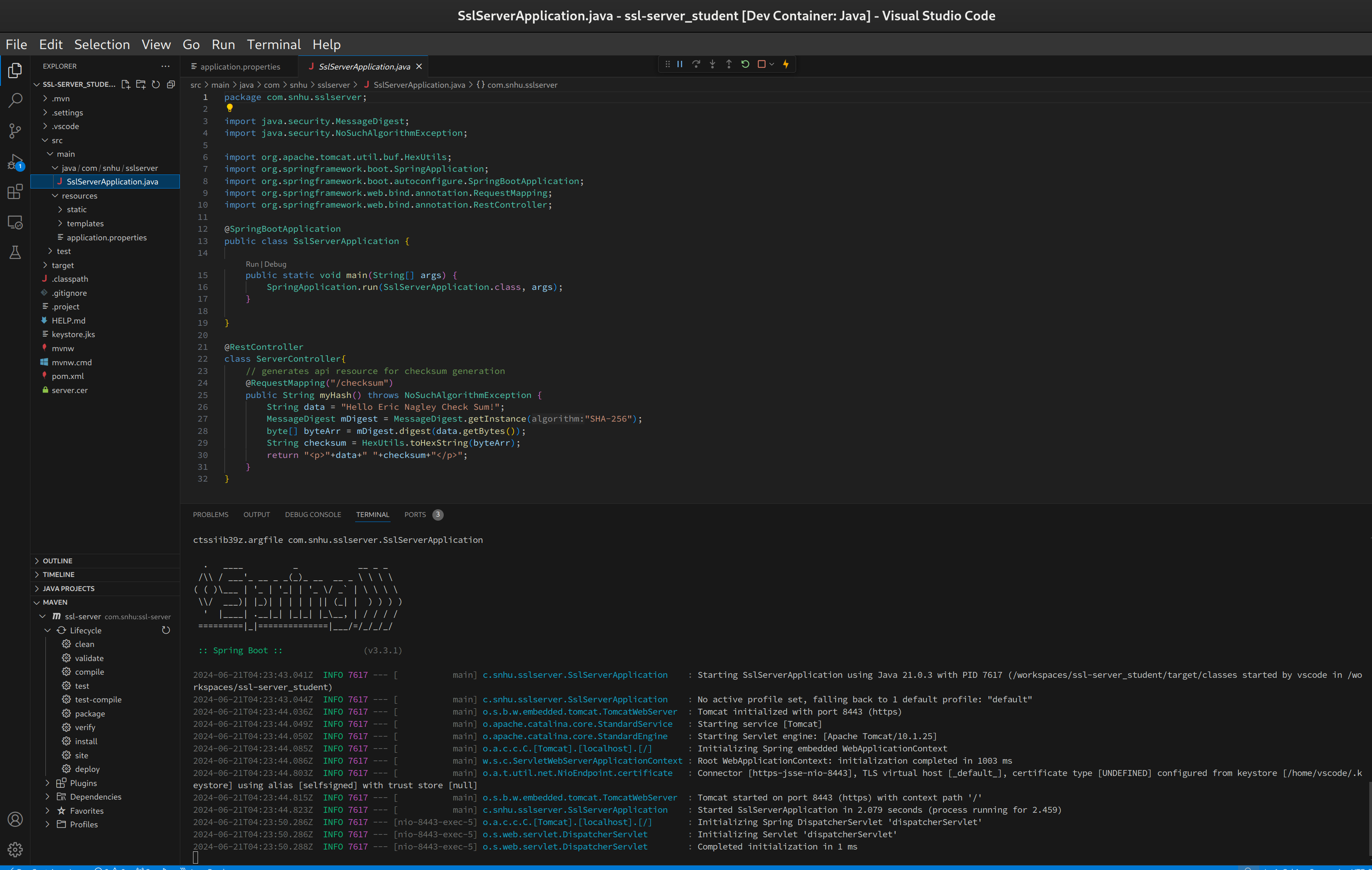






## Functional Testing

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



## Summary

Updated all dependencies to their latest versions via the `version` plugin. This resolved all the dependency-check findings.

This code does not use any external inputs.  
The codebase has been updated to use HTTPS for all web traffic. While using a self-signed certificate is not ideal it is suitable for the current use-cases.

A suitable hashing algorithm has been selected for hashing the payload data.

## Industry Standard Best Practices

Best practices implemented:  
Keep all dependencies updated

Use TLS1.3 for all web traffic, api and pages alike.

Use modern ciphers for all hashing and web traffic.

These practices are a minimum standard to block known vulnerabilities with all tools and modules used in your code. Failing to keep packages updated or securing communications with your application can allow for data to be stolen which can lead to expensive fines and lawsuits as well as loss of trust in the organization.