****

# 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** | **4/15/24** | **Nathan** |  |

## 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

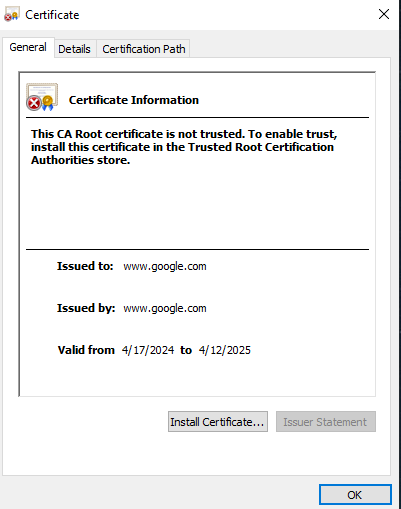
Nathan Book

## Algorithm Cipher

Our client, Artemis Financial, has a public web interface in which they would like to use the most current and effective software security to protect their client’s data and financial information. The most likely of attackers to their databases would be a cyber threat actor, to prevent this intrusion, encryption is the most effective defense against these hackers. This will render the files completely useless without the hash to decrypt the files. An asymmetric encryption key is recommended meaning the encryption key will be public and the decryption key will be private. For the most up-to-date security hashing algorithm, the client should use SHA-256. This is because it will prevent values from being reversed back to its original content and help avoid collisions. Meaning two separate inputs cannot produce an identical hash.

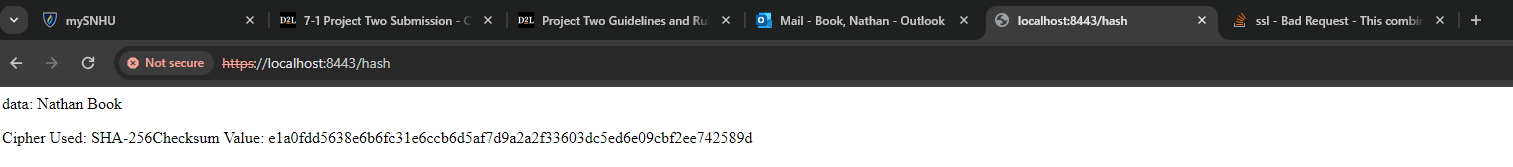
## Certificate Generation

Insert a screenshot below of the CER file.



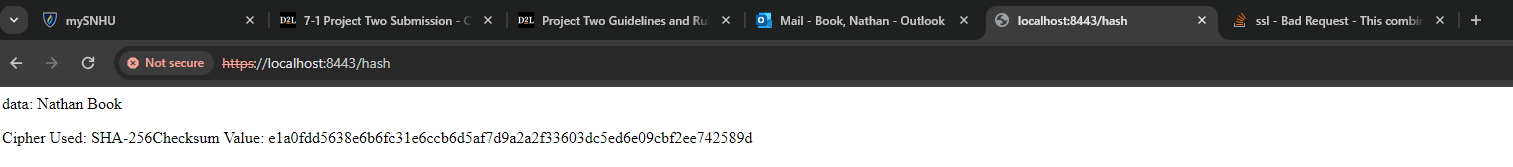
## Deploy Cipher

Insert a screenshot below of the checksum verification.



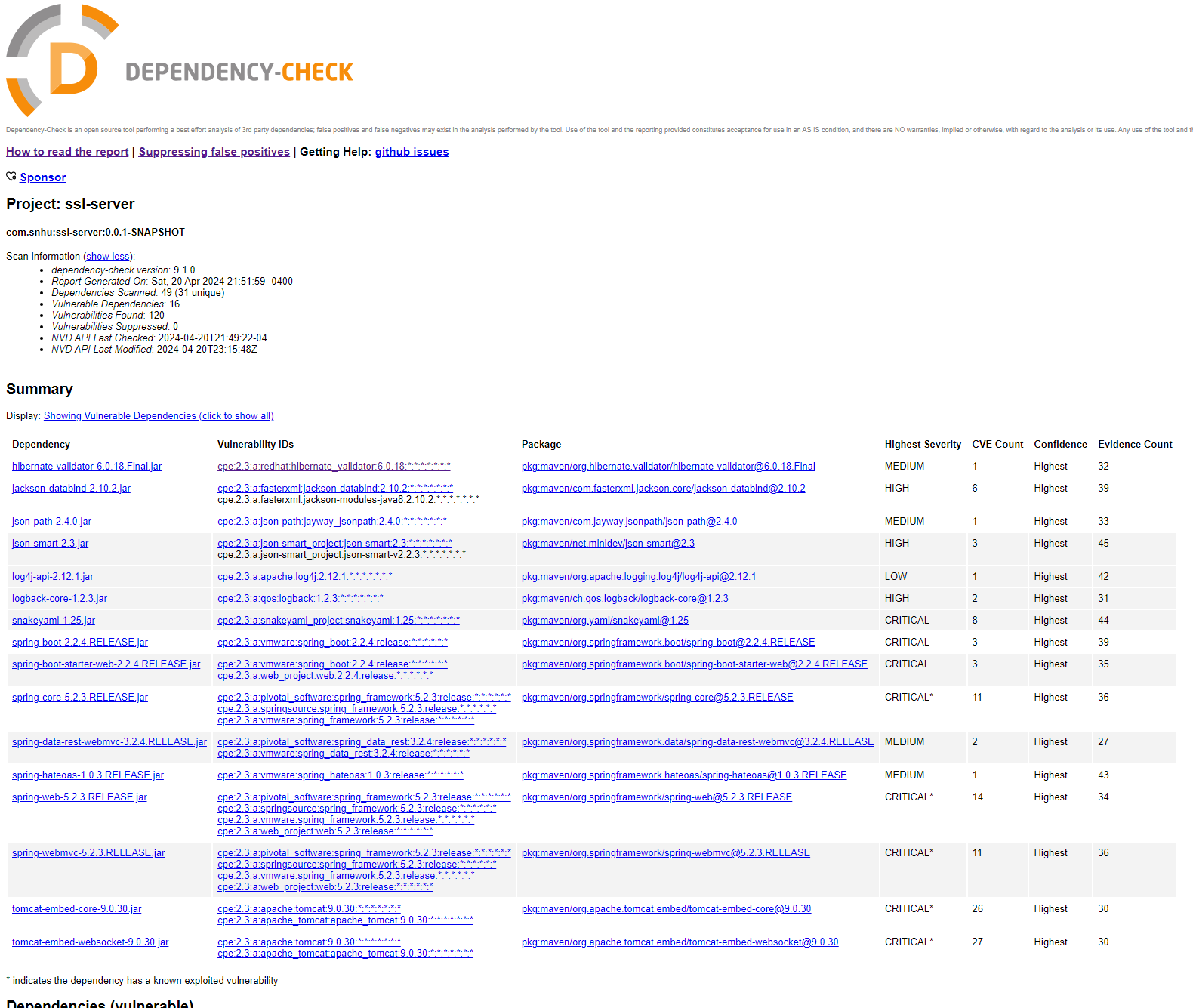
## Secure Communications

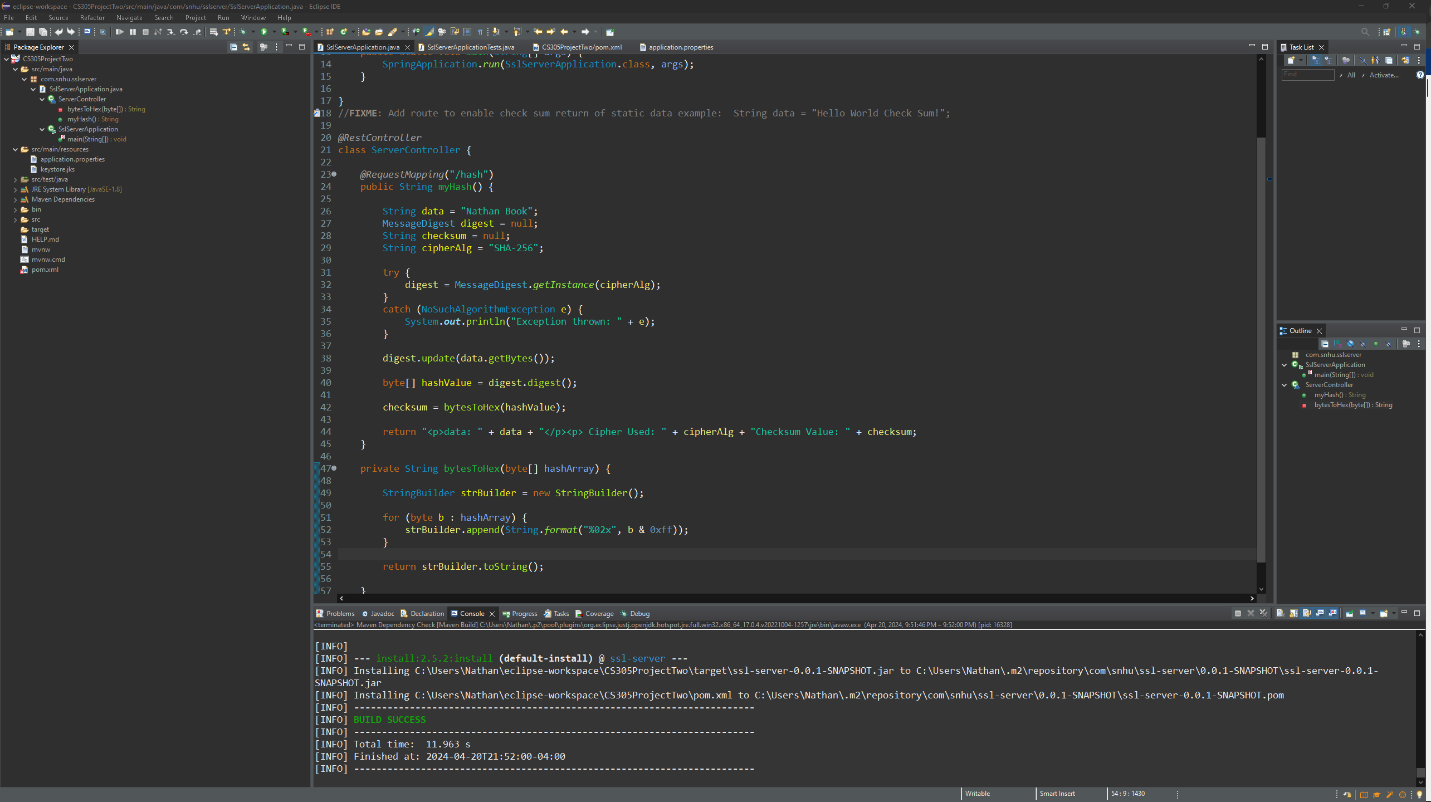
Insert a screenshot below of the web browser that shows a secure webpage.



## Secondary Testing

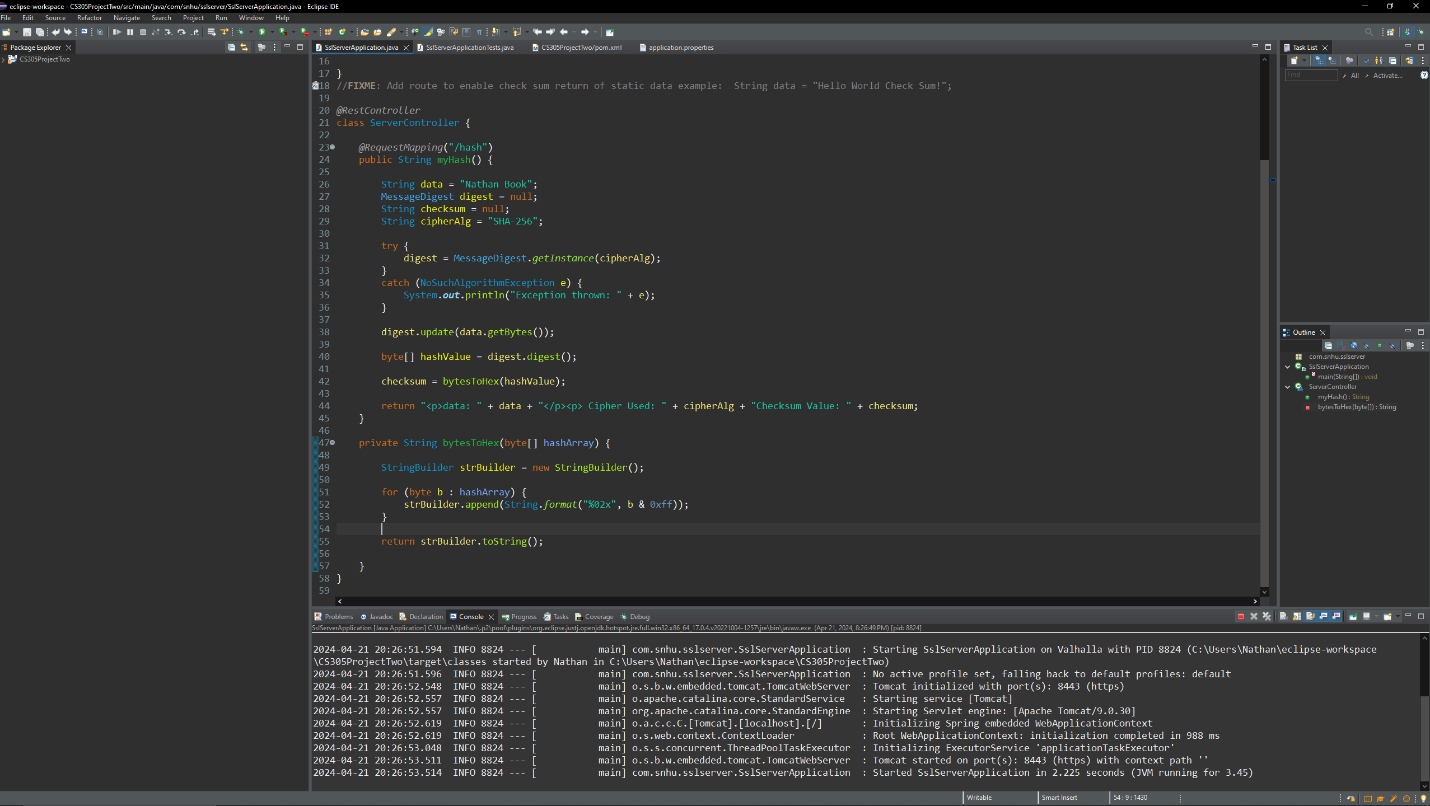
Insert screenshots below of the refactored code executed without errors and the dependency-check report.





## Functional Testing

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



## Summary

The areas of security address when refactoring the code are; APIs, Cryptography, Client/Server, and Code Quality. I used a self-signed certificate and generated key values using SHA-256 cipher algorithm. Through these security measures, I have ensured that only authorized users will have access to the site if they possess the hash value. In order to verify the validity of my secure hash, I ran a checksum to ensure the certification was in use. Once I was certain the I certain of the secure connection, I ran a dependency check on the refactored code to identify any threats currently known to the OWASP tool. Finally, I manually scanned through the code looking for any unsecure coding practices to ensure its integrity.

## Industry Standard Best Practices

To comply with industry standard best practices, I researched the latest technology in security algorithms with Java. After which I applied the method of encryption that secures the data at rest and in transit, this was a requirement for Artemis Financial considering the types of data they would be handling. There are several tools that help a developer test their code for any vulnerabilities, but the most highly recommended tool is the Maven Dependency-Check Plug-in. This tool allow me to test my code for known and potential vulnerabilities in the development community. All of these methods are pertinent to cyber security and should be used in any web-development environment. Without a secure database, attackers could gain access to Artemis Financial’s client information and ruin the integrity of the company and the well-being of their clients. To maintain a healthy relationship with your users, you should always secure your data and actively make efforts to prevent any future vulnerabilities.