****

# 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** | **[06/23/2025]** | **Nick Tilton** |  |

## 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

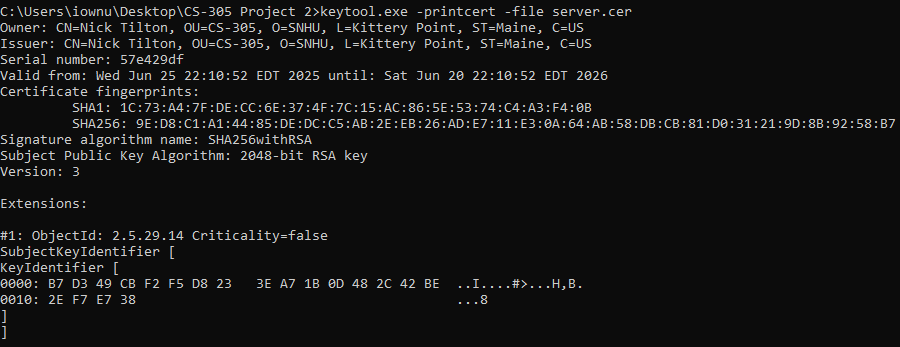
Nick Tilton

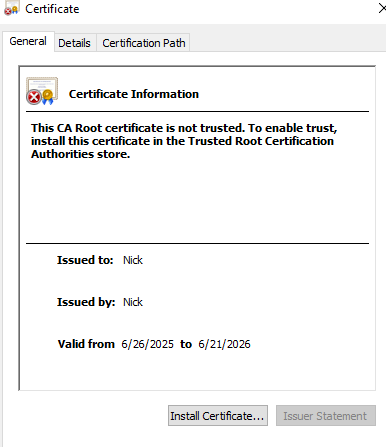
## Algorithm Cipher

For Artemis Finacial's secure communication needs, AES is the recommended encryption method due to its excellent balance of performance, efficiency, and security. This symmetric algorithm encrypts data in 128-bit blocks with 256-bit keys, providing strong protection for client data and transactions. SHA-256 is used to generate unique hashes for each file, allowing recipients to verify data integrity easily. AES employs cryptographically secure random numbers for its IDs, ensuring that identical data encrypted multiple times produces different ciphertexts. While AES handles encryption with symmetric keys, RSA is used for secure key exchange. Since its adoption by NIST in 2001, AES has become the industry standard for encryption, helping Artemis Financial maintain both confidentiality and data integrity within its web systems.

## Certificate Generation

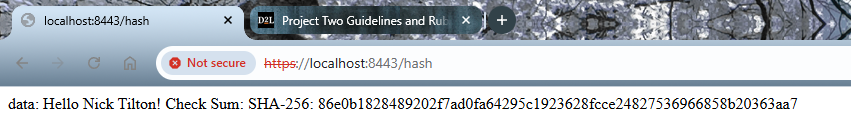
Insert a screenshot below of the CER file.





## Deploy Cipher

Insert a screenshot below of the checksum verification.



package com.snhu.sslserver;

import java.security.MessageDigest;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication

public class SslServerApplication {

public static void main(String[] args) {

SpringApplication.run(SslServerApplication.class, args);

}

}

//FIXME: Add route to enable check sum return of static data example: String data = "Hello World Check Sum!";

@RestController

class ServerController {

@RequestMapping("/hash")

public String myHash() throws Exception{

String data = "Hello Nick Tilton! Check Sum: SHA-256: ";

MessageDigest md = MessageDigest.getInstance("SHA-256");

md.update(data.getBytes());

byte[] digest = md.digest();

StringBuffer hexString = new StringBuffer();

for(int i = 0; i <digest.length;i++){

hexString.append(Integer.toHexString(0xFF & digest[i]));

}

return "<p>data: "+data+hexString.toString();

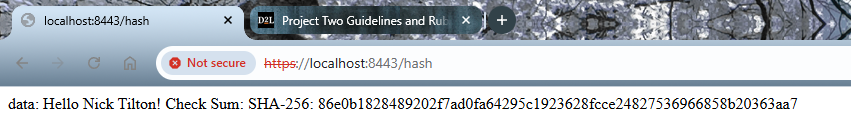
}

}

## Secure Communications

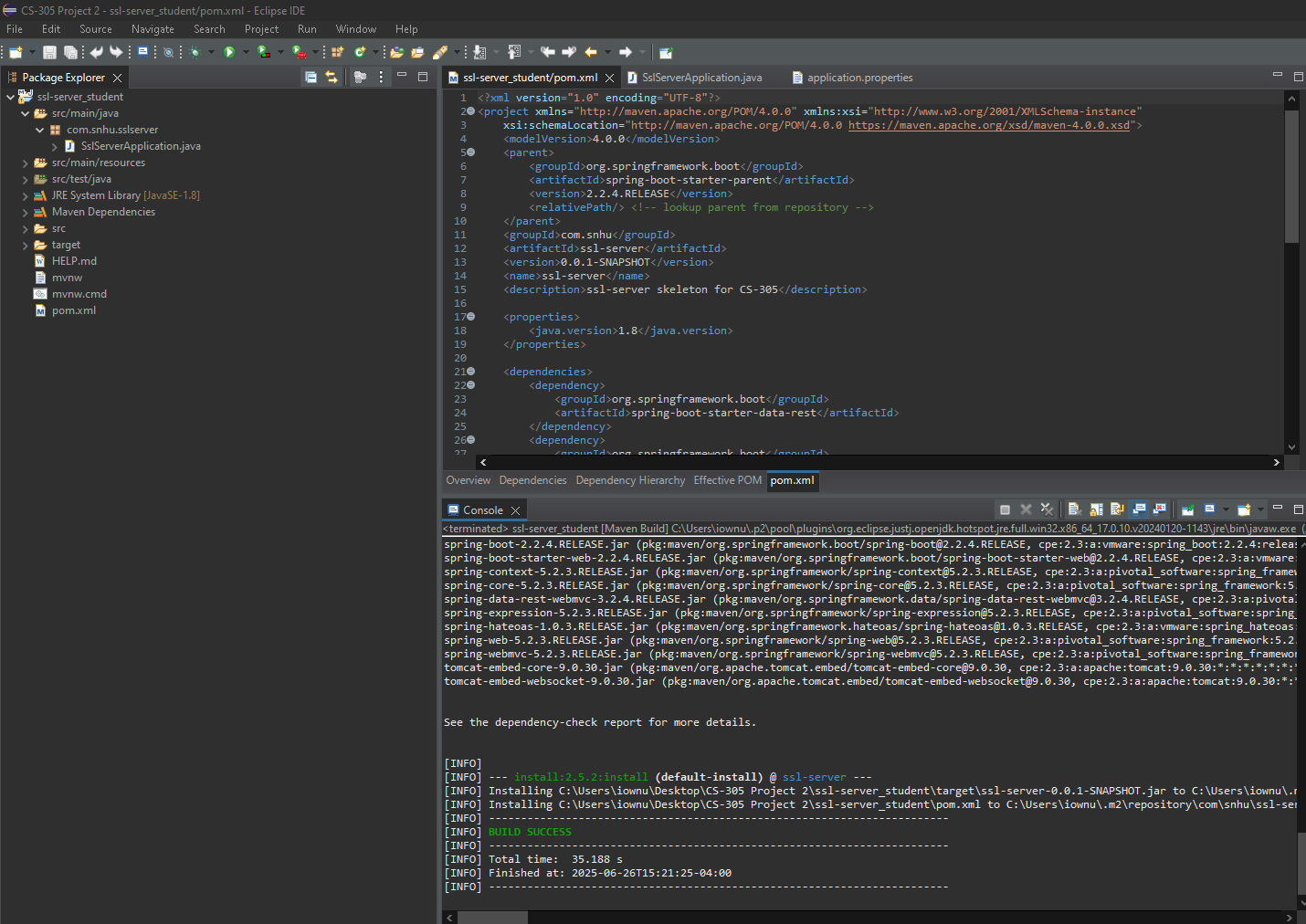
Insert a screenshot below of the web browser that shows a secure webpage.

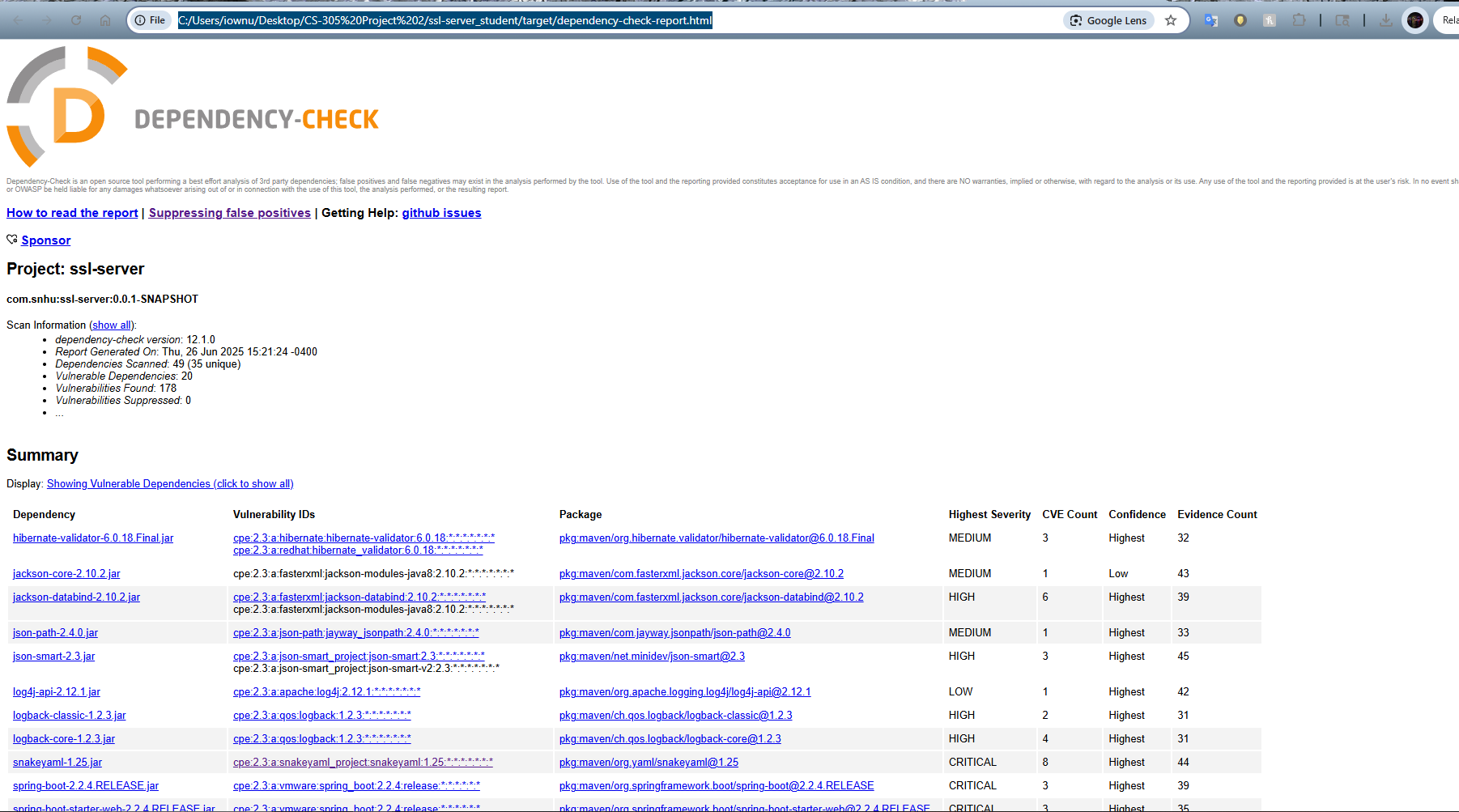
Since localhost is considered an untrusted webpage, it’s not deemed secure via google chrome. However, even though the connection isn’t secure, it was still successfully established. Having a secure connection is really important because it verifies the website’s identity and helps build trust with users. If everything is set up correctly, the website should show HTTPS instead of HTTP, indicating that the connection is secure as shown below.



## Secondary Testing

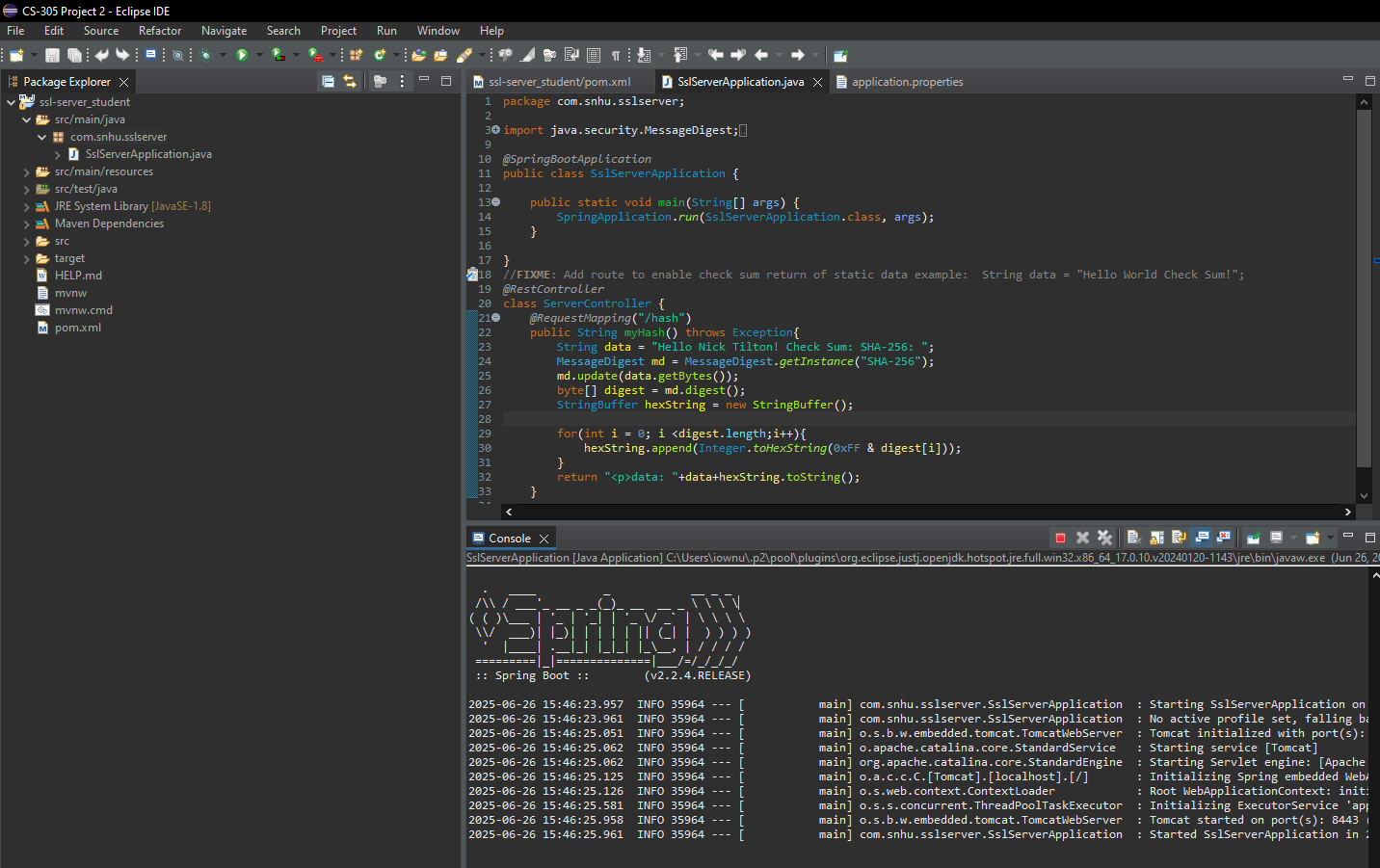
Insert screenshots below of the refactored code executed without errors and the dependency-check report.



ODC shows a valid test result the code ran without error

## Functional Testing

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



## Summary

I redesigned the code to include a secure RestController as the protected endpoint for the program’s hashing functions. The ServerController class was created to address the vulnerabilities identified during the assessment, such as issues with cryptography, coding errors, and overall code quality. I chose SHA-256 for hashing because it offers strong security and a low chance of collisions. To keep the system secure, I recommend checking dependencies every one to two months to spot and fix any potential vulnerabilities early. Additionally, regularly updating the plugins in the pom.xml file helps ensure I'm using the latest versions of key tools and other frameworks, which further boosts the application's security.

## Industry Standard Best Practices

Industry best practices cover a wide range of areas, with security being a crucial aspect. One key approach is using cryptography to safeguard data from unauthorized access, and in our case, we implemented hashing to protect sensitive information. We also created a self-signed certificate to verify that incoming data is legitimate and free from malicious content, opting not to purchase a certificate from a Certificate Authority (CA). These security measures not only protect users but also contribute to the overall health of the business. When a company consistently adopts such practices, it builds trust with clients, which can lead to increased growth and better client retention. Ultimately, the more trust a business earns, the stronger and more successful it becomes.