****

# 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** | **02/19/2023** | **Austin Frey** |  |

## Client



## Developer

Austin Frey

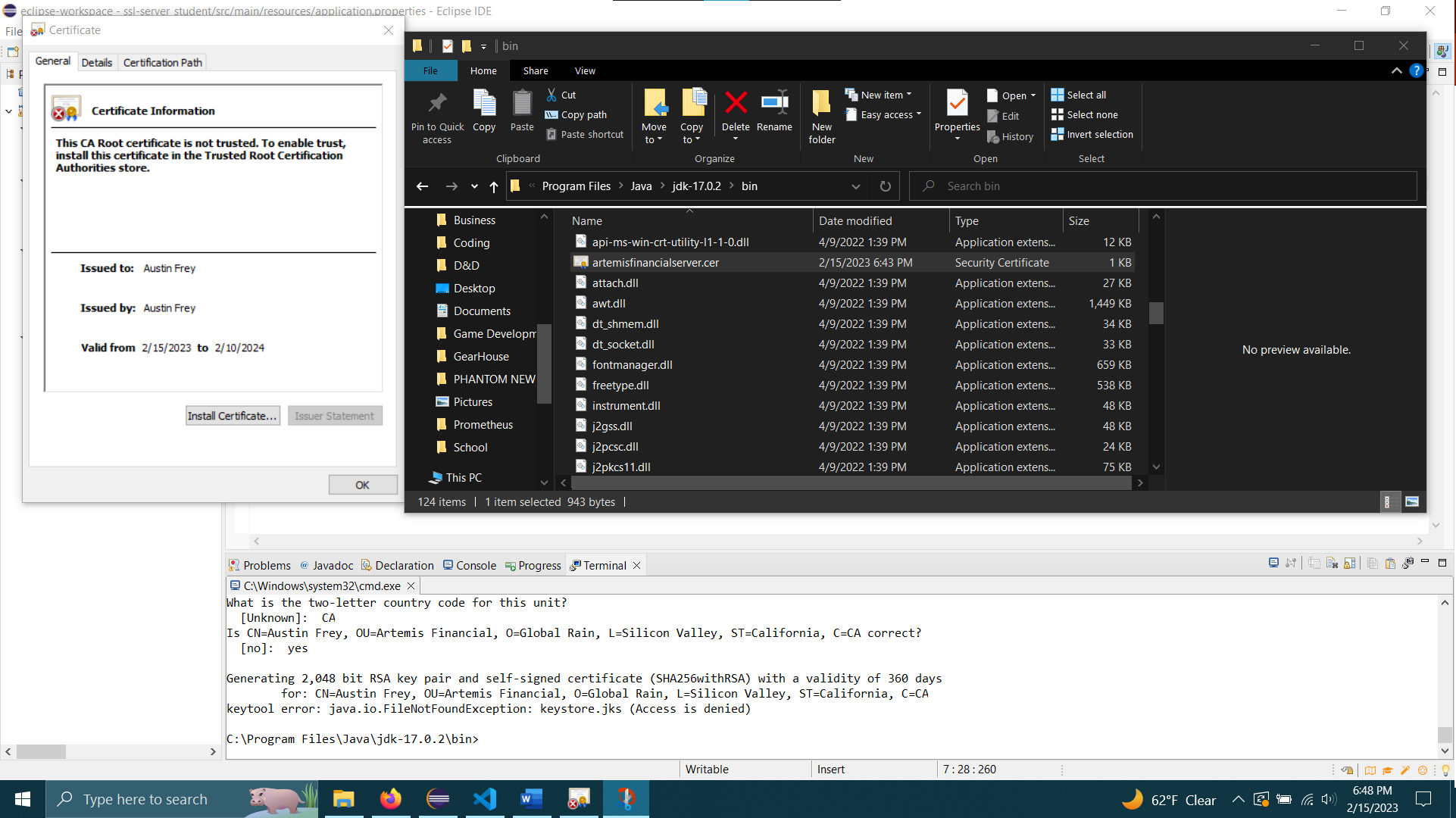
## Algorithm Cipher

Oracle offers numerous standard security algorithms for Java programmers. However, we will implement a secure hashing algorithm (SHA) using Java's Message Digest class. Progress Documentation (n.d.) defines Message Digest algorithms as "algorithms [that] rely on cryptographic hash functions to generate a unique value that is computed from data and a unique symmetric key." Message Digest algorithms have several variations including *MD2, MD5, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384*, and *SHA3-512* (*Java Security Standard Algorithm Names*, n.d.). So many variations of the message digest algorithm grant a wider spectrum of security, which is why we chose a Message Digest algorithm. Not only is SHA-256 a great compromise between speed and security, but it also has ". . . no known collisions found to date" *(CodeSigningStore*, 2022), making it one of the most versatile SHA algorithms.

A quick explanation of how SHA-256 operates is that whatever text is submitted to the algorithm goes through an internal hashing function and returns a ciphered version of the information. SHA algorithms have various hash functions and bit level options thanks to the several iterations of the algorithm (see the list of Message Digest algorithms above, for example). SHA offers several bit depths, such as 224, 256, 384, and 512. SHA-256 and SHA-512 are two of the most common implementations. SHA-256 is a one-way algorithm (also known as an asymmetric algorithm) that defies some standard expectations regarding cipher algorithms. SHA-256 does not seem to utilize random numbers for generating a hash code. Instead, the plain text goes through multiple iterations of being modified, where each block of 512 bits gets *64 rounds of operations* (Jena, 2022). In addition, the algorithm is an asymmetric algorithm, meaning that anyone can know the public key (the output of the algorithm) because there is a private key to "follow asymmetyric encryption methodology to verify the authenticity of a document/file" (Jena, 2022).

Following all the nuances of asymmetric vs. symmetric cryptography, hashing functions, random number generations, and the other numerous factors of cryptography can be overwhelming. So here is a short debrief to help you comprehend the progress of encryption algorithms and where the world of cybersecurity is now to provide some perspective. Once upon a time, the idea of cryptography would have been lost on people. Nowadays, most people have at least heard the word "encryption." The first evidence of cryptography was found in Egypt in 1900 BC, where "The scribe used some unusual hieroglyphic symbols here and there in place of more ordinary ones" (Sidhpurwala, 2023). Since then, cryptography has shown up sprinkled throughout early civilizations. However, the digital age brought a new swath of encryption algorithms and legitimized cryptography as a science. Now, cryptography isn't just a few examples of ciphered language. Organizations like the National Institute of Standards and Technology (NIST) are dedicated to researching the most secure encryption algorithms. NIST has been known to put out calls to develop the most secure encryption algorithms and can be held loosely responsible for the discovery of the Advanced Encryption Standard (AES), the most widely accepted symmetric encryption method. While AES is not the method we employed, SHA-256 is a standard for asymmetric encryption. According to Raghavan (2020), it remains one of the most robust encryption algorithms. Thus, you can trust your files are secured!

## Certificate Generation



## Deploy Cipher

Graphical user interface, application

Description automatically generated

## Secure Communications

Graphical user interface, application

Description automatically generated

## Secondary Testing

*Executed Code Without Errors*

Graphical user interface, text

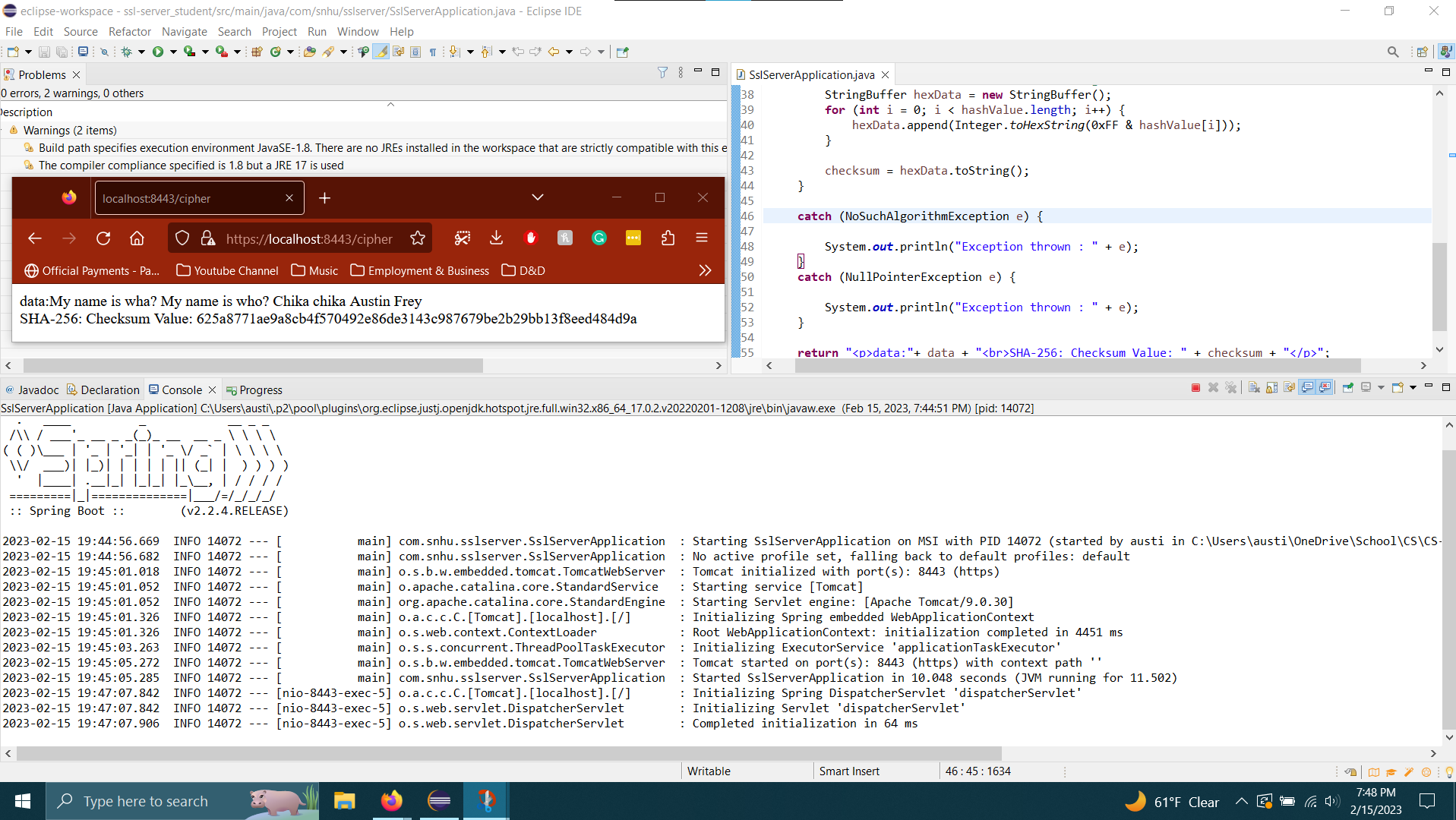
Description automatically generated with medium confidence

*Dependency-Check Report*

Graphical user interface, application, table

Description automatically generated

## Functional Testing



## Summary

The code I received was not so much "refactored" by me as it was developed. The code base I received had only two class files, one for the server application and one for the tests. Therefore, I have no code review on the original code base. However, the refactored code ("Project Two Code Base – Austin Frey") and the original code base ("CS-305 Project Two Code Base - Original") have both been included with this document's submission for review. So, without further ado, here is a summary of *my* contributions to the code base.

It may help to understand the components of an application architecture review before explaining how I contributed to various security aspects. Application architecture reviews analyze seven elements of a code base:

1. Input validation
2. APIs
3. Cryptography
4. Client/Server
5. Code errors
6. Code quality
7. Encapsulation

It isn't easy to assess the nature of several of the components above because I do not have access to the entire code base. However, the specific sections I contributed to with my development are cryptography, client/server interactions, code error handling, and code quality. I explain each section's improvements and implementations below.

*Cryptography*

I implemented secure cryptography practices using an SHA-256 message digest algorithm to create secure encryptions for file verification.

*Client/Server Interactions*

I ensured secure client/server interactions by generating a checksum value from the hash created by the encryption algorithm above. I also implemented code to convert HTTP to HTTPS, forcing more secure communications between the client and server.

*Code Error*

I contributed to code error handling by ensuring the encryption algorithm catches many exceptions that may arise during the execution of the function.

*Code Quality*

I ensured optimal code quality by utilizing clear variable names, code comments, frequent line breaks, properly indented brackets, and other standard programming practices to make the code easy to comprehend.

## Industry Standard Best Practices

I applied industry standard best practices in multiple ways. The most notable examples are below:

* Static testing to test dependencies for vulnerabilities
* Reduce external code reliance by programming in "vanilla" Java
* Deliver functioning code that requires no additional alteration to be implemented in the primary application code
* Other industry standards, as detailed in the summary section of this report

The methodologies above are valuable for a business like Artemis Financial because they mitigate security breach risks. Security breaches are *especially* detrimental to financial institutions because the slightest vulnerability being exploited can cost hundreds of thousands or even millions of dollars. It is unclear what the files contain that Artemis Financial is exchanging between their local devices and the web application. Still, it is most likely a safe assumption to assume sensitive data is within the file, at least sometimes. In today's day and age, every bit of information an attacker can gather is a threat to your business because it's a threat to the consumer. Just as Global Rain will be responsible to you for securing your web application, Artemis Financial is accountable to its customers to implement the most secure technologies. Let's create a safe web environment for the company and the customer's overall well-being.

## References

CodeSigningStore. (2022, March 23). *Hash Algorithm Comparison: MD5, SHA-1, SHA-2 & SHA-3*. Code Signing Store. <https://codesigningstore.com/hash-algorithm-comparison>

*Java Security Standard Algorithm Names*. (n.d.). <https://docs.oracle.com/javase/9/docs/specs/security/standard-names.html#messagedigest-algorithms>

Jena, B. K. (2022, November 11). *A Definitive Guide to Learn The SHA-256 (Secure Hash Algorithms)*. Simplilearn.com. <https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm#what_is_hashing>

*Progress Documentation*. (n.d.). <https://docs.progress.com/bundle/openedge-security-auditing-introduction-117/page/Message-digest-algorithms.html>

Sidhpurwala, H. (2023, January 12). *A Brief History of Cryptography*. <https://www.redhat.com/en/blog/brief-history-cryptography>