****

# 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** | **4/20/2025** | **Brooke Slampak** | **Project Two** |

## 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

Brooke Slampak

## Algorithm Cipher

Artemis Financial requires that client data be kept safe during transfers and online communication, therefore AES-128 encryption is the optimal choice. In AES-128, the “AES”, Advanced Encryption Standard, is a type of encryption that uses the same key to lock and unlock data, known as symmetric encryption. The "128" refers to the key size in bits, which is more than enough to keep data secure while also being resource efficient. It scrambles the data so that no one can read it without the right key, which is exactly what you want when handling private financial info.

AES-128 works with 128-bit blocks and keys, and while it doesn’t use a hash function on its own, it works well in systems where hashing is needed for things like checking data integrity. It also uses secure random numbers to help create the keys, which add extra randomness and make the encryption even stronger. Since it’s symmetric, AES-128 is way faster than asymmetric methods like RSA, which use separate keys for encrypting and decrypting. AES became the official standard back in the early 2000s and remains the standard to this day. AES-128, in particular, hits the sweet spot of being fast, secure, and easy to implement, making it the best fit for Artemis Financial’s security needs.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a computer program

AI-generated content may be incorrect.

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

AI-generated content may be incorrect.

## Secure Communications

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

A screenshot of a computer screen

AI-generated content may be incorrect.

## Secondary Testing

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

A screenshot of a computer screen

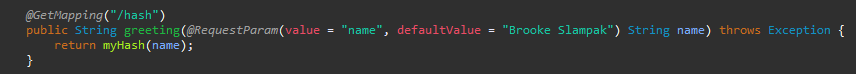
AI-generated content may be incorrect.

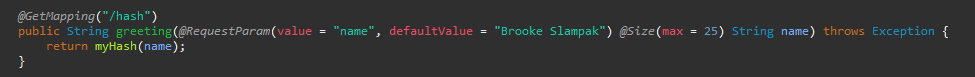
A screenshot of a computer

AI-generated content may be incorrect.

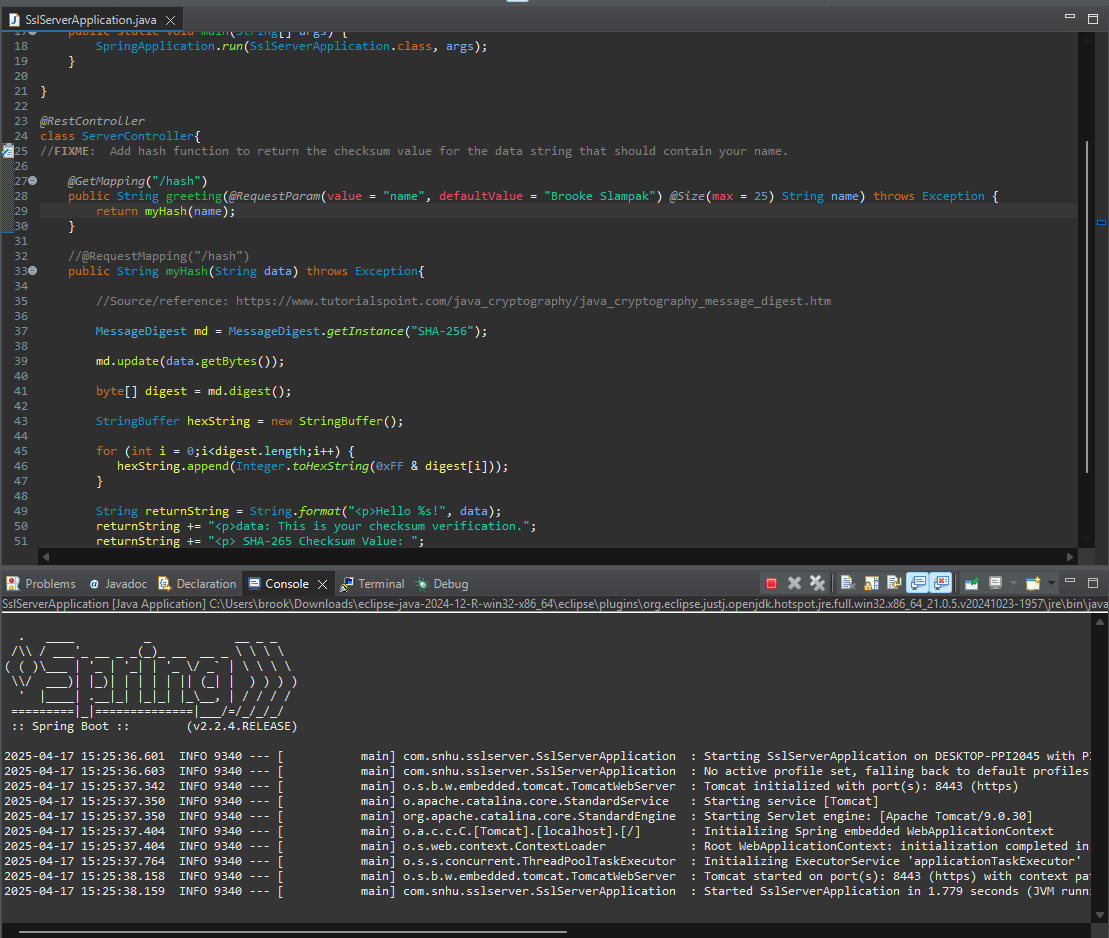
## Functional Testing

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





This change was necessary to enforce input validation and ensure that the name parameter remains within a reasonable and expected length. By adding @Size(max = 25), we protect the application from potential issues such as excessively long input values that could lead to poor performance, buffer overflows, or even security vulnerabilities like DOS attacks. Implementing input validation early in development enhances the security and reliability of the application by catching invalid input early.



## Summary

The code was changed to send all web traffic over HTTPS instead of HTTP. That means information going back and forth is now encrypted, making it a lot harder for anyone spying on the network to see what’s going on. Right now, it uses AES with 128-bit encryption and signs requests with a personal certificate, but that’ll eventually need to be swapped out for one that’s tied to a trusted Certificate Authority (CA), so it’s recognized more widely and securely.

The API that was built returns a checksum for a given string. Basically, it creates a unique digital fingerprint of whatever you send it. This is done using a SHA-256 hashing algorithm, which is very reliable, it’s so unlikely to produce the same result for two different inputs. Older hashing methods like MD5 and SHA-1, on the other hand, have known weaknesses and can produce collisions, which could let someone trick the system or even work backwards to figure out the original file.

The code also went through a static analyzer to look for known security issues (CVE checks). That kind of scan is useful, but it can’t catch everything, especially problems that only show up when the program’s actually running. That’s where a dynamic scanner can help, by catching things like memory leaks, broken HTTPS redirects, and other runtime issues. As a final security touch, the API was updated to only accept input strings between 1 and 25 characters. That helps protect against buffer overrun attacks.

## Industry Standard Best Practices

Keeping the app secure requires regular maintenance, you can’t just build it and forget about it. You need to keep it updated, continuously check for weak spots in the code, and make sure only the right people have access to sensitive data. It also helps to use tools that scan for security issues and to keep an eye on anything that looks suspicious. These best practices make sure the app stays safe over time and can handle whatever new threats may arise.

For the company, writing secure code from the start is important. It helps avoid messy situations like data breaches or getting hit with fines for not following regulations. It also builds trust with customers. If they know their info is safe, they’re more likely to remain customers. Plus, it’s cheaper and easier to fix security issues early on than to have to clean up a problem later.