****

# 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** | **02/27/2025** | **Devin/Madison Bashaw** |  |

## 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

Devin/Madison Bashaw

## Algorithm Cipher

Looking at current threats and weaknesses in older systems, I recommend using SHA-256 due to its hash function that whips out a 256-bit output that offers strong security and directly addresses the vulnerabilities within the application. SHA-256 is part of the SHA-2 family, and standardized by NIST. It takes any input and provides a unique 256-bit output. Even a small change in the input creates a completely different result. This makes SHA-256 a trusty tool for verifying data integrity and supporting digital signatures.

Since the cipher generates a 256-bit hash that is very hard to reverse or guess, brute-force attacks are less likely to happen. It is also used in key derivation, digital signatures, and data verification. When combined with other security measures, it creates a strong, layered defense against any tampering or foul play. Although SHA-256 itself does not generate random numbers, secure systems often use random values to create keys and nonces, so this is not really an issue to be worrying about either. These random numbers help ensure that any data processed with SHA-256 remains protected against attacks. Since it is also widely used, companies are familiar with it, and the more people familiar with it the more user friendly and scalable it becomes.

## 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

AI-generated content may be incorrect.

A screenshot of a computer

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

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.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Summary

## I refactored the code for enhanced security by analyzing thr application and considering what algorithmic cipher to implement, to which I chose SHA-256 which is a versatile algorithm used throughout many industries and provides the best security possible for this application through its usage of the 256 bit system and sole HTTPS connection responses. Additionally, by integrating the OWASP dependency check maven plugin, I was able to repeatedly scan for vulnerabilities within the application and address them accordingly. I also reviewed exception handling and implemented the NoSuchAlgorithmException that prevents the leakage of data in the application. These are the steps I took to ensure a secure, reliable application.

## Industry Standard Best Practices

## I applied industry standard best practices by doing code reviews and automated testing to spot vulnerabilities early in the development cycle, input validation and output encoding to prevent vulnerabilities like injection attacks or cross-site scripting, managed files and secured them in proper folders, granted only necessary permissions to code modules and users (I let no one else see the project, get access to it, or get near my laptop). Applying these standards prepares you for your career, allowing you to have prior knowledge of industry standards before even entering the industry and practicing these beforehand. When projects are hacked, more than just the company is at stake, we are talking about consumers/customers/clients, even patients in hospitals or infrastructure. That is why security is so important.