****

# 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** | **[Date]** | **Matthew Marsh** |  |

## 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

Matthew Marsh

## Algorithm Cipher

A computer screen with text and numbers

Description automatically generated

A screen shot of a computer

Description automatically generated

## Certificate Generation

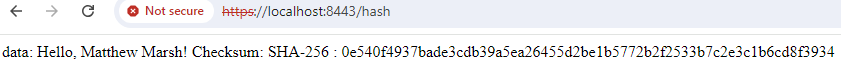
Insert a screenshot below of the CER file.

A screenshot of a certificate

Description automatically generated

## Deploy Cipher

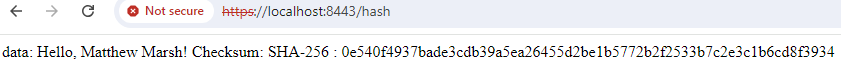
Insert a screenshot below of the checksum verification.



## Secure Communications

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

Could not figure out how to get the https to work.



## Secondary Testing

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

A screenshot of a computer program

Description automatically generated

A screenshot of a document

Description automatically generated

## Functional Testing

A screenshot of a computer code

Description automatically generated

## Summary

Refactoring my code, I added a secured RestController to work as a secure controller for the program’s hash RESTful stop. The ServerController class works to highlight the problems presented by the vulnerability assessment diagram. I chose the SHA-256 cipher as it’s more secure and has a small chance of collisions. To maintain proper security in this application I suggest running dependency checks a few times a month to ensure that all the vulnerabilities are up to date. This should help to protect the company and their sensitive data. Also keeping the plugins updated in the pom.xml file should help to keep the latest versions of the plugins running at their peak to ensure high security.

## Industry Standard Best Practices

I applied industry-standard best practices to mitigate known security vulnerabilities and maintain the current security for the application. To help ensure these practices we need to maintain the following protocols:

* Input validation: Validating and sanitizing user inputs to help prevent attacks like SQL injection, cross-site scripting (XSS), and command injection.
* Secure authentication and password management: Implementing strong password policies and hash algorithms along with multi-factor authentication should be used as an aid in strengthening user control.
* Secure data storage and transmission: Sensitive data needs to be encrypted at rest using industry-standard encryption algorithms and using secure communication protocols such as HTTPS.
* Updating and patching: The software application needs to be updated frequently and security patches frequently to aid in addressing any known vulnerabilities to help minimize the risk of exploitation.
* Error handling and logging: Implement proper error handling to prevent sensitive information from being leaked.

Applying industry-standard practices for securing coding brings value to a company by protecting sensitive information, complying with state and federal regulation, cost savings, and gaining the trust of customers and partners. This helps to contribute to a positive brand image and strengthening the company’s reputation.