****

# 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** | **10/15/2023** | **Ethan Hutchison** |  |

## 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

Ethan Hutchison

## Algorithm Cipher

Artemis Financial is looking to modernize their operation and with that comes making sure the systems that they are using are safe and secure from any threats. Since they are trying to protect user data and financial information while also adding a file verification step to the web application to ensure secure connections, I would recommend using SHA-256. SHA-256 uses 256 bits for its encryption sequence and is strong enough that it can prevent hackers from being able to access the original message just from the hash value. SHA-256 also has an extremely low chance of collisions with basically none being found so far.

## Certificate Generation

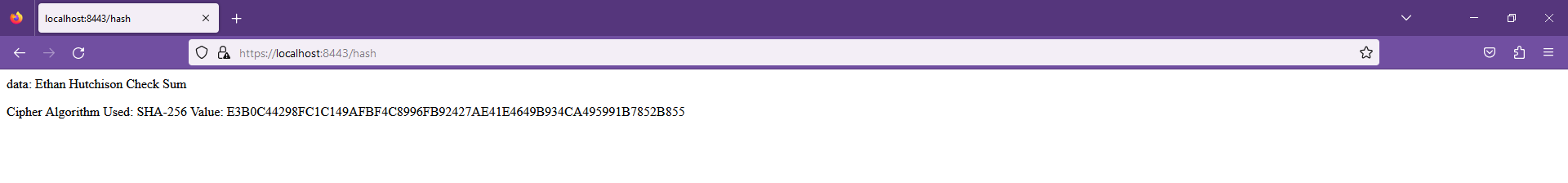
Insert a screenshot below of the CER file.

A screenshot of a certificate

Description automatically generated

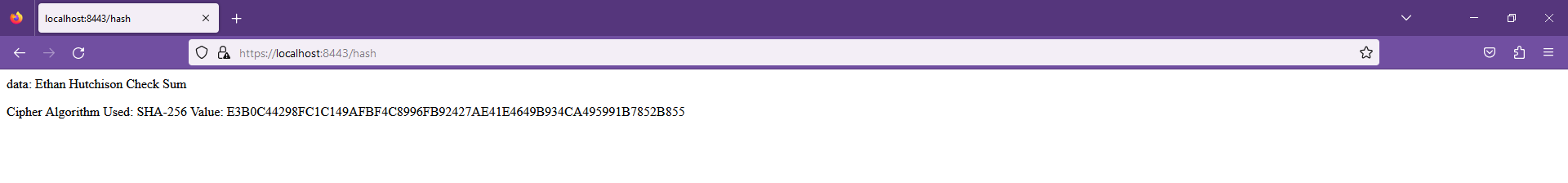
## Deploy Cipher

Insert a screenshot below of the checksum verification.



## Secure Communications

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



## Secondary Testing

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

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

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

A screenshot of a computer

Description automatically generated

## Summary

For this code I refactored it to have a RestController for the RESTful API to help make it more secure. This contains a SHA-256 algorithm to cipher the data passed through it. This covers the Cryptogrophy area of security. I have made sure the Maven Dependency check is up to date so that it is accurate as well.

## Industry Standard Best Practices

To make sure that the current security level is maintained it was important to keep everything up to date so that old vulnerabilities may be fixed. Also making sure to use a verifiably secure cipher algorithm to maintain as much security as possible when sending data. This makes sure that there are no known exploits in the algorithm that can be used to breach the data being sent.