****

# 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 5](#_Toc102040760)

[4. Secure Communications 5](#_Toc102040761)

[5. Secondary Testing 5](#_Toc102040762)

[6. Functional Testing 6](#_Toc102040763)

[7. Summary 7](#_Toc102040764)

[8. Industry Standard Best Practices 7](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **October 15, 2023** | **Victor Sanchez** | **Completed the report** |

## 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

Victor Sanchez

## Algorithm Cipher

The two types of cryptography algorithm ciphers are symmetric and asymmetric. Symmetric encryption uses a single key to encrypt and decrypt the data. A key is created to encrypt the data and then the data cannot be accessed unless the same exact key is used. However, in asymmetric encryption two keys are created, a public key and a private key. The private key is known only by the sender and the public key is made public for anyone; the public key is then used to validate the signature. Since Artemis Financial wants to add a file verification step to their web application, which will then require a checksum verification, I would recommend using an asymmetric algorithm cipher. In particular, I would recommend that the SHA-256 hash function is used for the checksum verification. The SHA-256 hash function creates a 256-bit hash comprised of random numbers and letters, which makes it one of the most secure and strongest hash functions available. The SHA-256 hash function also has never been compromised.

## Certificate Generation

Insert a screenshot below of the CER file.

A screenshot of a certificate

Description automatically generatedA computer screen with white text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

Description automatically generated

## Secure Communications

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

A screenshot of a computer

Description automatically generated

## Secondary Testing

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

Refactored code:

A screenshot of a computer

Description automatically generated

Original code:

A screenshot of a computer

Description automatically generated

(No new vulnerabilities were added)

## Functional Testing

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

A screenshot of a computer screen

Description automatically generated

## Summary

I added an extra level of security to the original code for Artemis Financials’ website by first creating and generating a certificate. Then I implemented a SHA-256 hash function in the code to generate a hash for the checksum verification (which was outputted to the screen). I then verified that the connection was secure by opening it in a new web browser. Referring to the vulnerability assessment process flow diagram, the refactored code addressed cryptography, and code quality. It addressed cryptography since I had to generate a unique hash for the data. Code quality was addressed since security is very crucial to Artemis Financial, the code also had to follow secure coding practices.

## Industry Standard Best Practices

Since Artemis Financial handles customers’ money, security is a top priority. Security should also be a top priority when writing code for their web application. I was able to add a checksum verification process using SHA-256, one of the most secure hash functions. While doing so, I also verified that no new vulnerabilities were added (as shown in the secondary testing section). The code was able to generate a new hash correctly and was able to be outputted to the user as an example (as shown in section 3 and 4). It is crucial to continue revising and updating the code and website software so that no new vulnerabilities are added, especially since other people’s money is involved. If a vulnerability is found and breached, this may cause major backlash for the company. This will cause many customers to leave and will also waste company time and money.