****

# 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** | **10/20/2024** | **Jerimey Burnside** |  |

## 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

Jerimey Burnside

## Algorithm Cipher

Artemis Financial Recommendation: AES

Overview: AES is an encryption type that is fast and secure. It is in wide usage today for keeping data secure.

Hash Functions and Bit Levels: AES can use keys of sizes 128, 192, or 256 bits. Larger keys imply stronger security.

Symmetric vs. Non-Symmetric: AES is symmetric, meaning it uses the same key for locking and unlocking data. That is why it does an excellent job in quickly locking lots of information, such as money transfers.

Random Number Usage: Secure random numbers in creating keys and commencing encryption make AES very strong.

## Certificate Generation

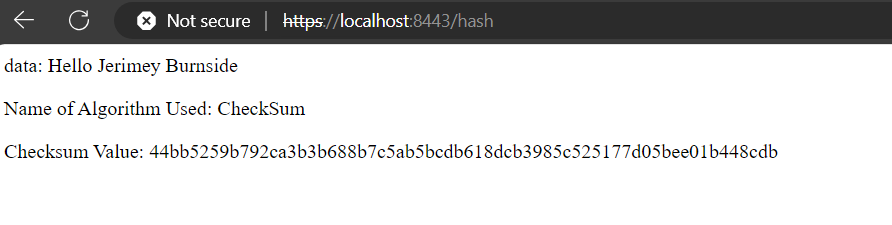
Insert a screenshot below of the CER file.

A computer screen with white text

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.

Work computer wouldn’t allow the use of an unsigned certificate.

A screenshot of a computer

Description automatically generated

A screenshot of a certificate

Description automatically generated

## Secondary Testing

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

A screen shot of a computer program

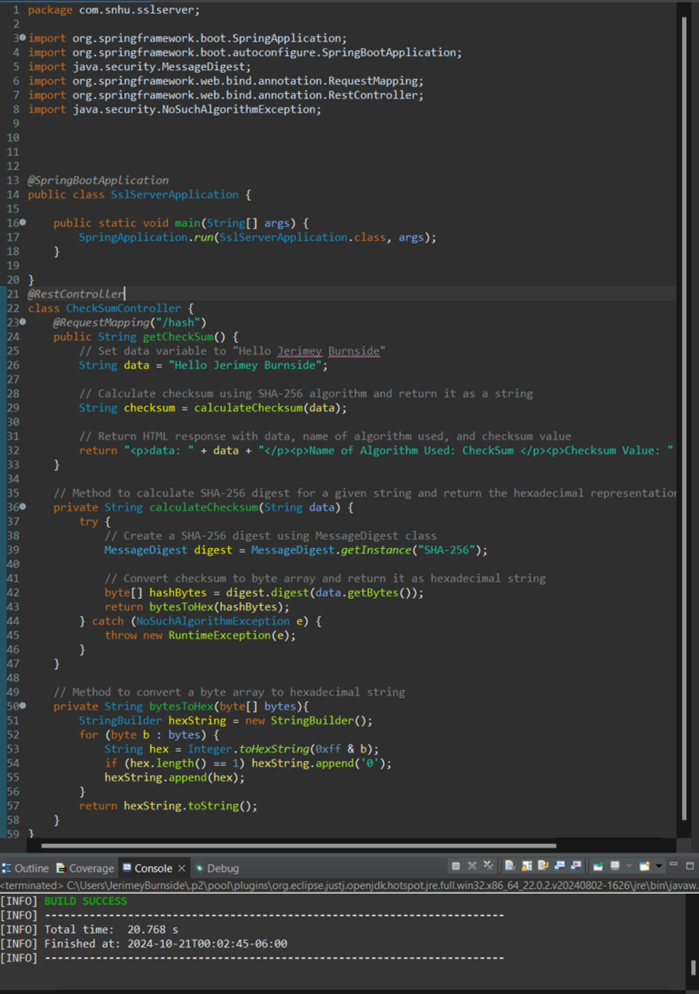
Description automatically generated

A screenshot of a computer program

Description automatically generated

## Functional Testing

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



## Summary

In the code added, I have added a secure RestController to act as the safe controller for my program hash's RESTful endpoint. The ServerController class will be constructed in a way that it will handle the issues shown in the vulnerability assessment diagram. I decided to go with the SHA-256 hashing algorithm because it's very secure, barely having any possibility of collision. To keep this application secure, I highly recommend checking its dependencies once or twice a month to get updated on possible vulnerabilities. This will help protect the company and its sensitive information. Also, if all the plugins in the pom.xml file are up-to-date, that will ensure the usage of the latest versions, which is good for security.

## Industry Standard Best Practices

**Using Best Practices to Keep the Application Secure**:  
  
Switching to HTTPS with TLS:  
I ensured that the application is secure with some known security tips. One big thing I did was migrating to HTTPS through the use of something called Transport Layer Security (TLS). This encrypts all the data sent between a user and server into a secret, making it impossible for someone to steal this data while in transit.  
  
Using SHA-256 for Hashing:  
I also introduced the SHA-256 method, which generates a kind of "fingerprint" for data. Thus, it helps us to be sure whether any data has been messed with or not. SHA-256 is quite strong, allowing it to make it super hard for hackers to make fake data look real.  
  
Input Validation:  
The other important step is the verification of all the information that is fed into the system. This will prevent common attacks, such as SQL injection and cross-site scripting-XSS-by ensuring that no harmful stuff gets in.  
  
OWASP Dependency-Check:  
To review all extra code libraries that were in use within our project, I used the tool OWASP Dependency-Check. This tool has made it very easy to track known security issues in these libraries, which could otherwise make the system unsafe.

**Why Best Practices Matter to the Company**:

Secure Coding Practices to Keep Data Private:

The use of this application is a representation of secure coding practices to ensure data privacy and security. I restricted the risk of data theft by using HTTPS protocol and strong encryption. This is an important procedure that will help in the protection of the clients of the company by safeguarding the trust of the same clients.

Preventing Security Issues:

Many different best practices, such as input checking and vulnerability scanning, will prevent the majority of common attacks from occurring at all. This makes an application a great deal safer and reduces the number of emergency fixes later.

Following the best practices for security, Artemis Financial will avoid fines and possible legal complications due to specific legislations, most of all those concerning finance, such as GDPR or PCI DSS.

Building a good reputation: Security-conscious firms are perceived as more reliable. A customer is more likely to want to do business with Artemis Financial if they can be assured that their information will not be compromised. This may also provide a competitive edge against other firms.

The advantage of saving time and money: Saves the company from spending precious time and money repairing security issues at later stages; thus, it is considered a great way of starting off a company to ensure its growth without being in constant tension on the security breach aspect.

Conclusively, through best practices concerning security, I have made the application stronger and safer, therefore protecting Artemis Financial's clients from such threats while making sure that the company avoids such risks and builds its reputation as a trustworthy business.