****

# 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** | **4/26/24** | **Michael Isenhour** |  |

## 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

Michael Isenhour

## Algorithm Cipher

When it comes to the encryption cipher that Artemis Financial should use, the one I would have to recommend is the Advanced Encryption Standard (AES). After doing some research on AES I see that it “is the standard set by the U.S. National Institute of Standards and Technology in 2001 for the encryption of electronic data” (Smirnoff & Turner, 2020). This cipher is known for its strong security and effectiveness when defending against a wide range of attacks including brute force attacks. This is due to its large key sizes which are 128, 192, and 256 bits. Another reason I would choose this cipher is because of its flexibility in operations as it allows its security to be enhanced even further by providing additional features. These features include things like data authentication and integrity checks which are extremely important when handling customers financial data and other sensitive information.

While AES might be the standard with many benefits there are some potential risks in using it as well. For example, even though AES itself is very secure it relies on correct implementation. If this encryption is implemented poorly or developers use weak key management then this could lead to security gaps. Another potential risk is that its high levels of encryption can be very intensive for the system its running on which could impact overall system performance.

When it comes to government regulations this cipher meets them exceptionally well as again it “is the standard set by the U.S. National Institute of Standards and Technology in 2001 for the encryption of electronic data” (Smirnoff & Turner, 2020). Since it is government standard it is an excellent choice for handling sensitive financial information.

When it comes to using this cipher though proper setup is critical, it can be seamlessly integrated into many programming languages. For example, the Java language has Cryptography Architecture libraries that ensures AES compatibility and would be the best choice to implement into Artemis Financials infrastructure as this implementation would ensure smooth integration.

Though there could be other ciphers that may be better then AES there is no such thing as one best cipher. With AES being set as the standard when it comes to encryption of electronic data, as well as being super effective and easy to implement there is currently no reason to choose another cipher that is less tested and potentially more complex to implement.

**Justification:**

When it comes to the purpose of hash functions and bit levels hash functions convert input into compressed values. By compressing the data input into compressed values it makes the data unreadable to outside eyes and makes it hard for hackers to tamper with. The bit levels are the size of the key and is used to encrypt the data. The higher the bit levels the stronger the encryption and more possible combinations making it harder for hackers to use brute-force attacks.

The use of random numbers in encryption is for unpredictability when creating secure key generation. This unpredictability ensures that the keys generated are not easily guessable making it extremely difficult for hackers to obtain access. Symmetric key encryption uses one generated key for both encrypting and decrypting the data. While this is a fast method for encryption it requires a secure method to share the generated key between both parties. Non-symmetric key encryption however creates two generated keys one is a public key for encrypting the data while the other is a private key for decrypting the data. This non-symmetric key encryption method is the one that is typically used when it comes to encryption on the internet.

While encryption has been used in places throughout history “the first known evidence of the use of cryptography was found in an inscription carved around 1900 BC in Egypt” (Sidhpurwala 2023). Others who used encryption in the past was Julius Caesar when sending messages to his army as well as the United States using it in World War 2 to send messages secretly. As time passed and we reached the digital age encryption evolved also and encryption algorithms were created to keep digital information private between parties and secure from outside individuals.

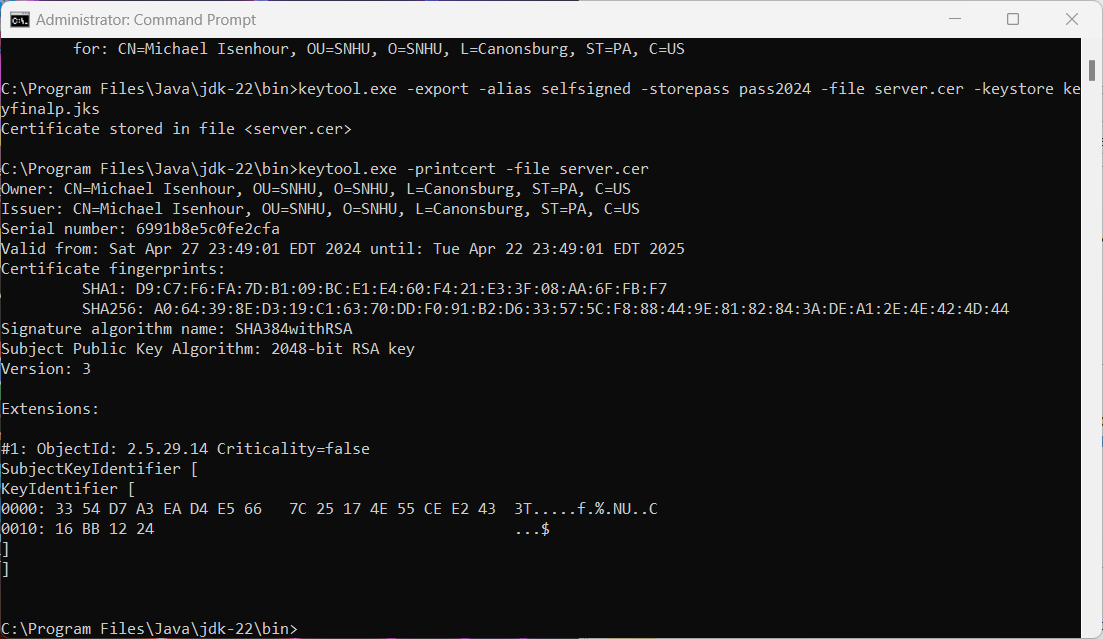
**Works Cited**

Smirnoff, P., & Turner, D. M. (2020, January 3). Symmetric Key Encryption - why, where and how it’s used in banking. Cryptomathic. https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking

Sidhpurwala, H. (2023, January 12). A brief history of cryptography. Red Hat. https://www.redhat.com/en/blog/brief-history-cryptography

## Certificate Generation

Insert a screenshot below of the CER file.



## Deploy Cipher

Insert a screenshot below of the checksum verification.

A screenshot of a computer

Description automatically generated

## Secure Communications

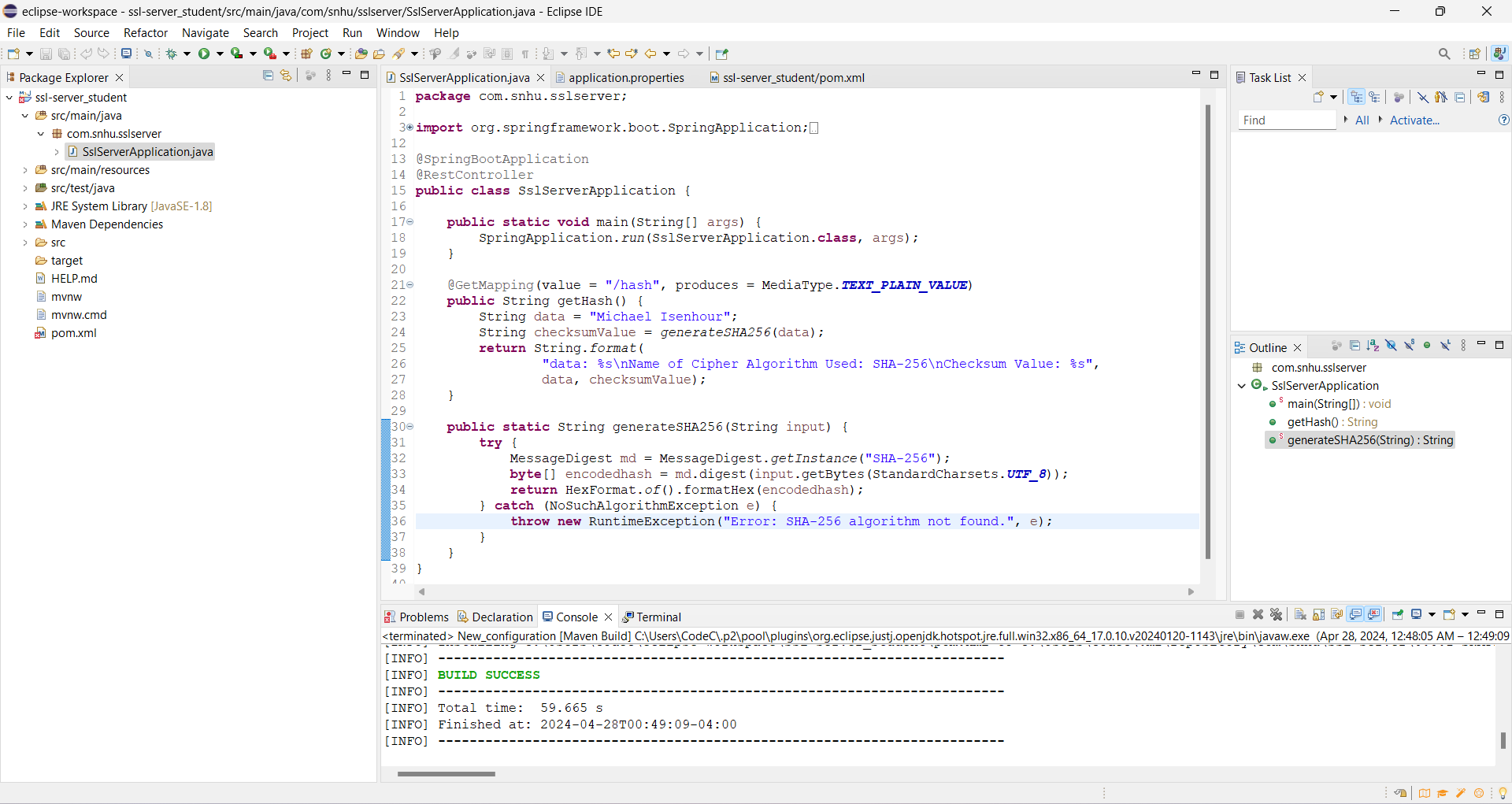
Insert a screenshot below of the web browser that shows a secure webpage.

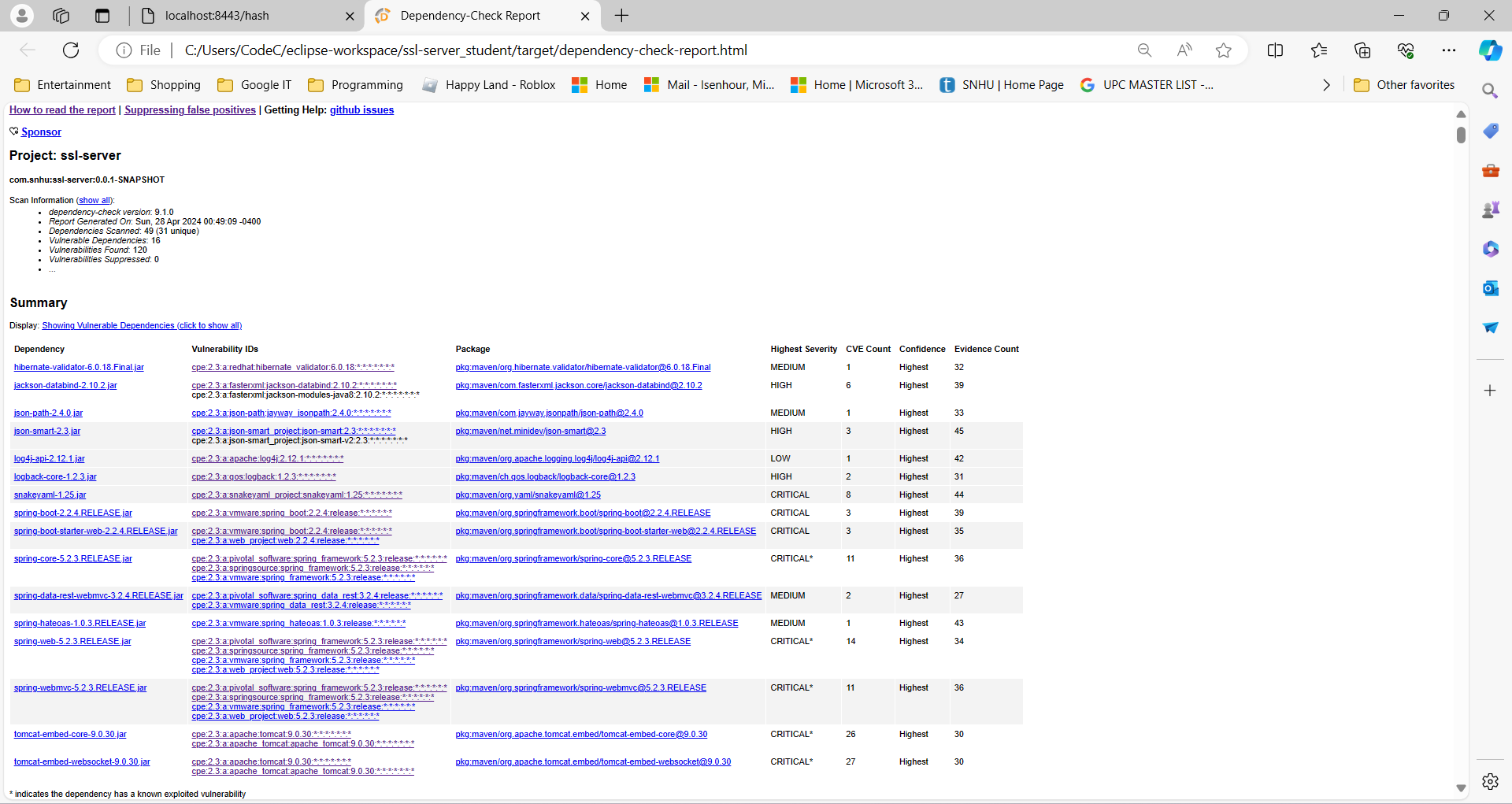
A screenshot of a computer

Description automatically generated

## Secondary Testing

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





## Functional Testing

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

A screenshot of a computer

Description automatically generated

When performing Functional testing on this application I can confirm that it performs as expected. It is generating secure hash outputs without errors. The program also uses proper use of error handling and encryption.

## Summary

To align this project with the Vulnerability Assessment Process Flow Diagram and make it more secure I implemented a RestController for handling SHA-256 hashing. By implementing SHA-256 which is known for being impenetrable against attacks I have maximized our protection against data tampering or potential leakage. Along with this I also updated our pom file and upgraded to the latest version of Maven Dependency Check. This update will enhance our ability to detect vulnerabilities associated with outdated libraries. I feel these steps have greatly enhanced our overall security and will have great success protecting Artemis Financial and its customers.

## Industry Standard Best Practices

When it comes to the refactoring of the Artemis Financial software application I feel I was successful at applying the latest industry standards for secure coding. One of the main practices I implemented was cryptography when I added the SHA-256 hashing algorithm. This addition aligns with the industries recommendations when it comes to secure data handling and also ensures that sensitive financial data will remain safe and confidential.

I also integrated regular updates and checks into the system by upgrading to the latest version of the Maven Dependency Check. Updating Maven so it can enhance our ability to detect vulnerabilities is another industry standard best practice and will keep our software from being exploited by hackers.

While implementing industry standards helps keep our software secure, it goes beyond software and also helps the company’s overall well-being. By keeping to the industry standards it ensures that Artemis Financial has good reputation and maintains trust with its customers and also that they keep in compliance with regulations regarding data protection and user privacy.