

# 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]** | **[Your Name]** |  |

## 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

Max Dunphy

## Algorithm Cipher

I recommend using AES-256 (Advanced Encryption Standard with a 256-bit key length) for Artemis Financial's software application to ensure secure communication and data protection. AES-256 is a symmetric encryption algorithm, meaning the same key is used for both encryption and decryption. It is well-suited for protecting sensitive client information, such as financial plans, savings, retirement, and insurance data, due to its balance of strong security and efficiency.

### **Why AES-256?**

AES-256 is widely recognized for its robust security and efficiency, making it one of the most trusted encryption methods in the world. It is a key component in many secure communication protocols, including SSL/TLS, and is endorsed by global standards organizations such as the National Institute of Standards and Technology (NIST). AES-256 offers protection against brute-force attacks due to its large key size (256 bits), which creates an enormous number of possible combinations—2^256, or approximately 10^77. This makes it virtually impossible for attackers to break the encryption through brute-force methods.

In addition to its security benefits, AES-256 is designed to be computationally efficient. It is fast, making it ideal for real-time encryption of large volumes of data, which is crucial for a financial services provider like Artemis Financial. The symmetric nature of AES allows for quick encryption and decryption, reducing latency in communication while ensuring that data remains protected both at rest and in transit.

### **Hash Function: SHA-256 for Data Integrity**

For checksum generation and ensuring data integrity, I recommend using SHA-256 (Secure Hash Algorithm). SHA-256 is a cryptographic hash function that takes an input and produces a fixed-size 256-bit hash value, often referred to as a "digest." This is ideal for generating checksums that verify the integrity of data transmitted between the web application and its users. Any alteration to the original data—whether due to accidental corruption or malicious tampering—will result in a completely different hash value, alerting the system to potential data integrity issues.

SHA-256 is part of the SHA-2 family of cryptographic hash functions and has been widely adopted due to its collision resistance and strength. It is often used in digital signatures, SSL/TLS certificates, and blockchain technology, further underscoring its reliability in security-critical applications.

### **Symmetric vs. Asymmetric Keys**

AES-256 is a symmetric key algorithm, which means the same key is used for both encryption and decryption. This is different from asymmetric encryption algorithms such as RSA, which use two different keys: a public key for encryption and a private key for decryption. Symmetric encryption is generally faster and more efficient for encrypting large amounts of data, as is often the case with financial information, making AES-256 a practical choice for Artemis Financial.

While asymmetric encryption is commonly used for secure key exchange (e.g., in SSL/TLS protocols), symmetric encryption like AES-256 is preferred for bulk data encryption due to its speed. In practice, asymmetric encryption (such as RSA) is often used to securely exchange the keys that will be used for symmetric encryption during a secure session.

### **History and Evolution of Encryption Algorithms**

AES was developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, and was selected by NIST in 2001 to replace the aging Data Encryption Standard (DES), which had become vulnerable to attacks due to advances in computing power. AES was designed to be secure, efficient, and scalable, capable of handling 128-bit, 192-bit, and 256-bit key sizes.

Since its standardization, AES has become the gold standard for data encryption, being used in a wide variety of industries, from finance to healthcare to government. AES-256, in particular, is widely regarded as the most secure version, given its longer key length and resistance to brute-force attacks. Unlike earlier algorithms such as DES, which was limited to a 56-bit key and vulnerable to modern cryptanalysis, AES-256 provides a much higher level of security.

Today, AES is used not only in software encryption solutions but also in hardware, including processors, security tokens, and network devices. Its combination of high security, efficiency, and global acceptance makes it the ideal choice for securing sensitive financial information.

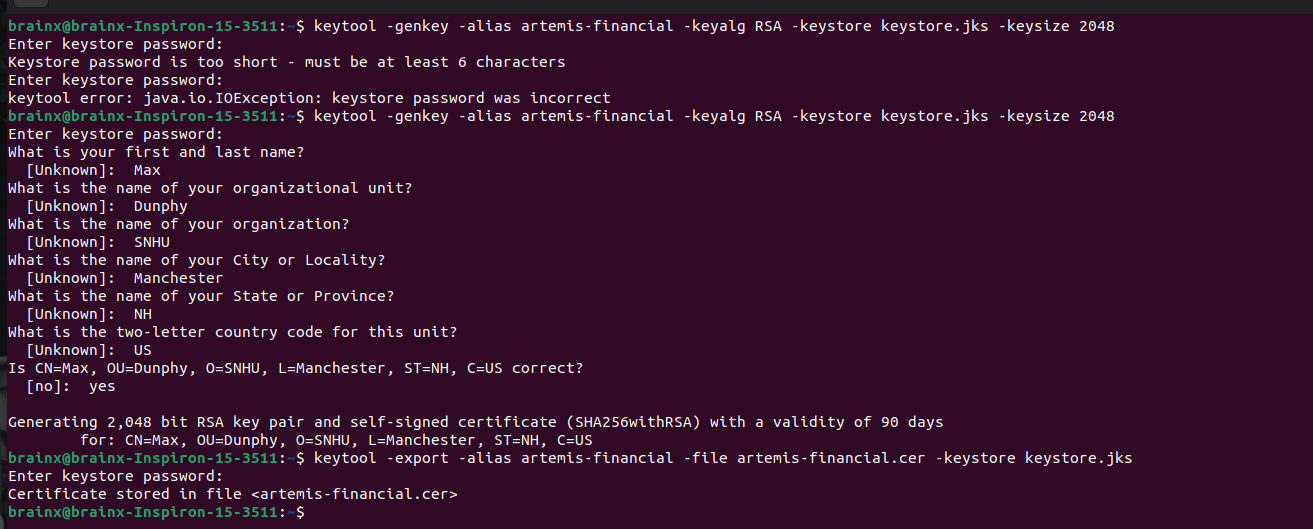
### **Current State of Encryption**

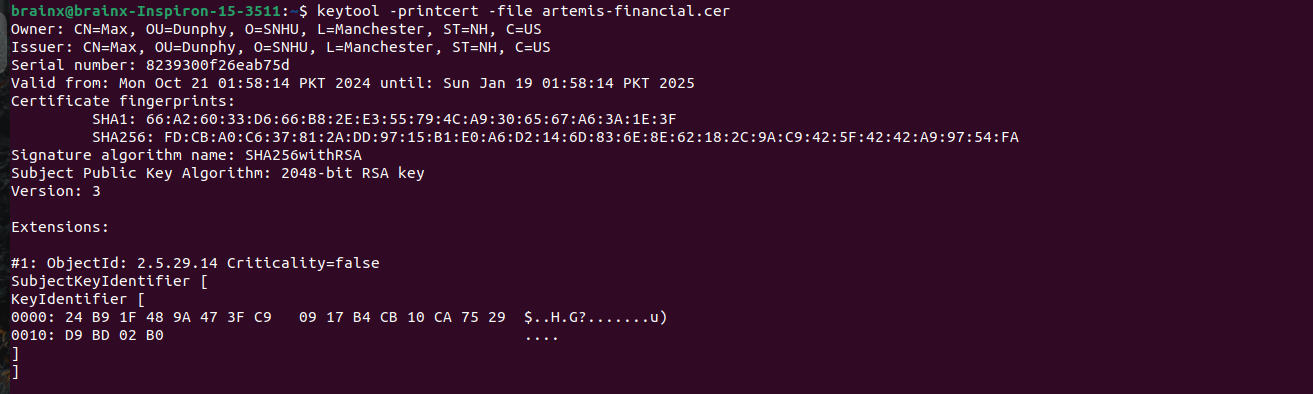
In the modern landscape of cybersecurity, encryption plays a critical role in protecting data from unauthorized access and ensuring privacy. AES continues to be the most trusted encryption algorithm and is a key part of secure protocols like SSL/TLS, VPNs, and disk encryption solutions. It is implemented in almost every major operating system and communication protocol, making it a universal tool for ensuring secure data transmission.

For Artemis Financial, using AES-256 will provide strong protection for their clients’ sensitive financial data, ensuring that it is encrypted both in transit and at rest. This will help the company maintain the trust of its clients and comply with industry standards and regulations regarding data security.

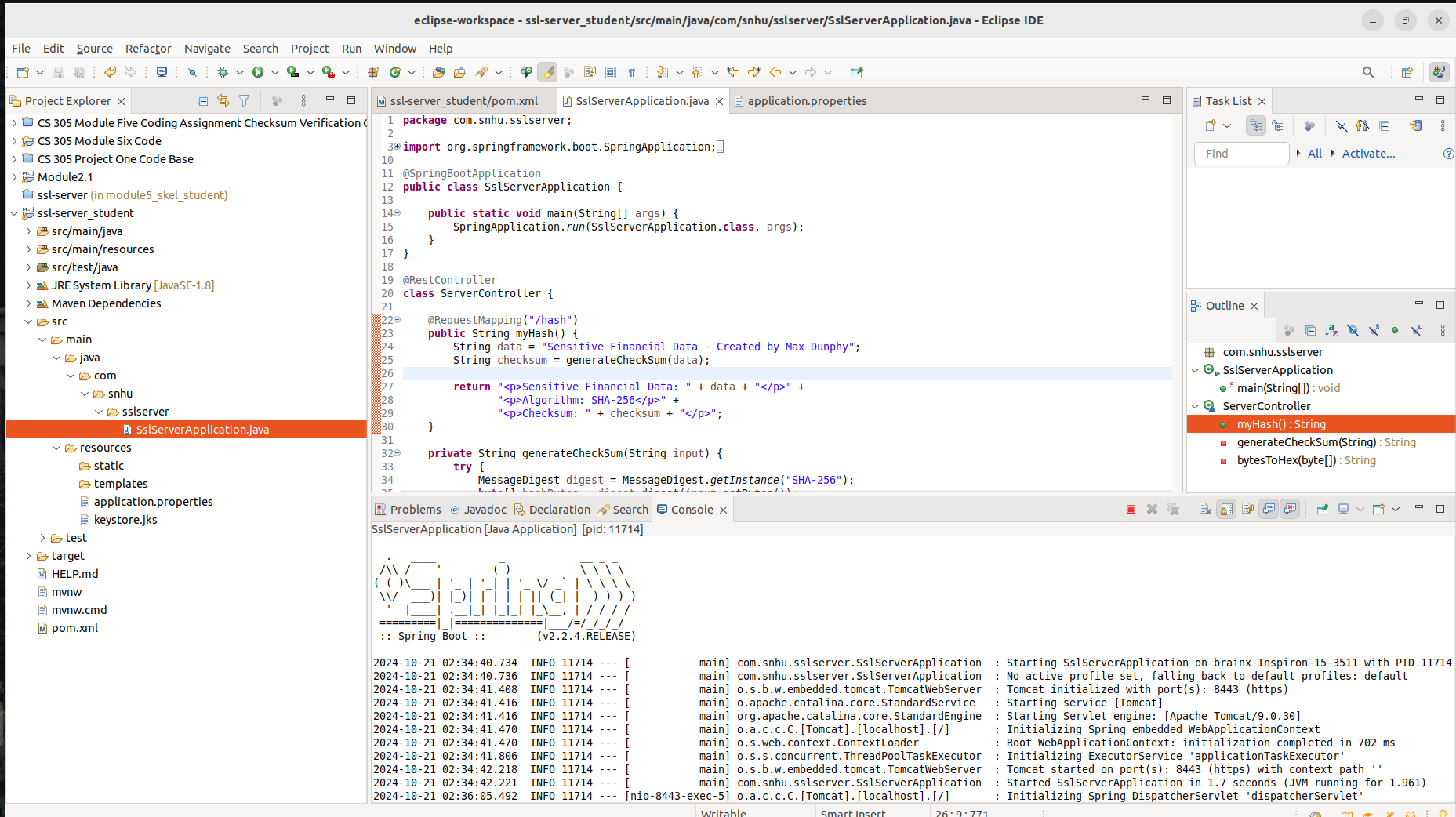
By integrating AES-256 into Artemis Financial's software and using SHA-256 for checksums, we can ensure the confidentiality, integrity, and authenticity of all data communications, providing a secure foundation for the company's financial services.

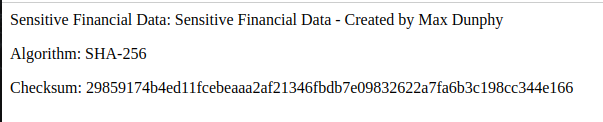
## Certificate Generation





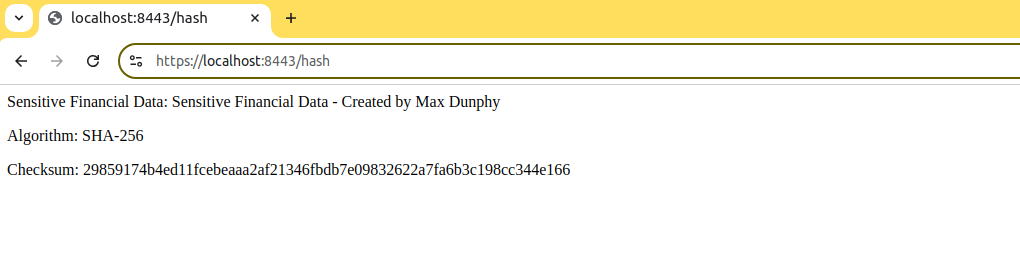
## Deploy Cipher





## Secure Communications

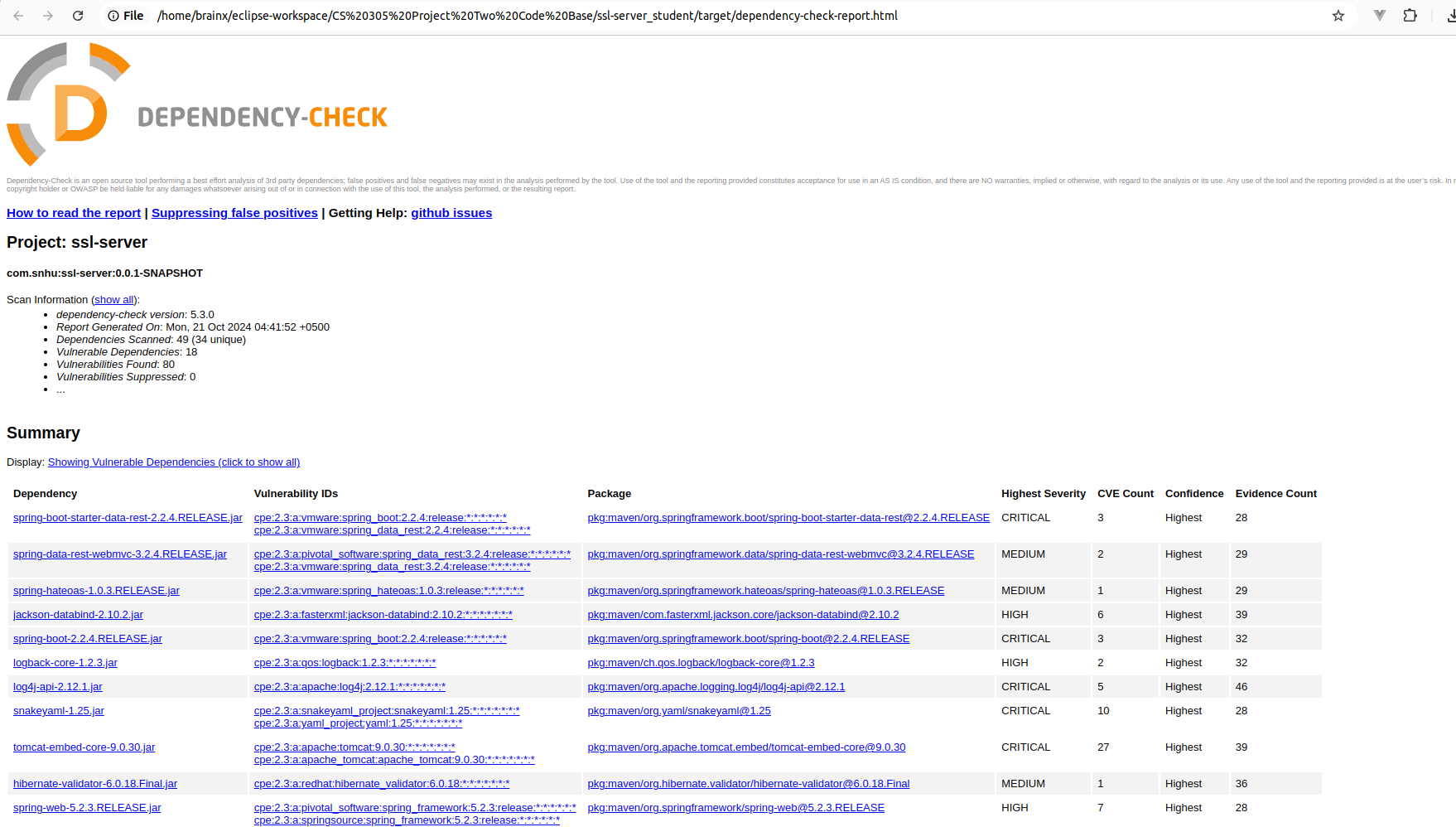
Insert a screenshot below of the web browser that shows a secure webpage.



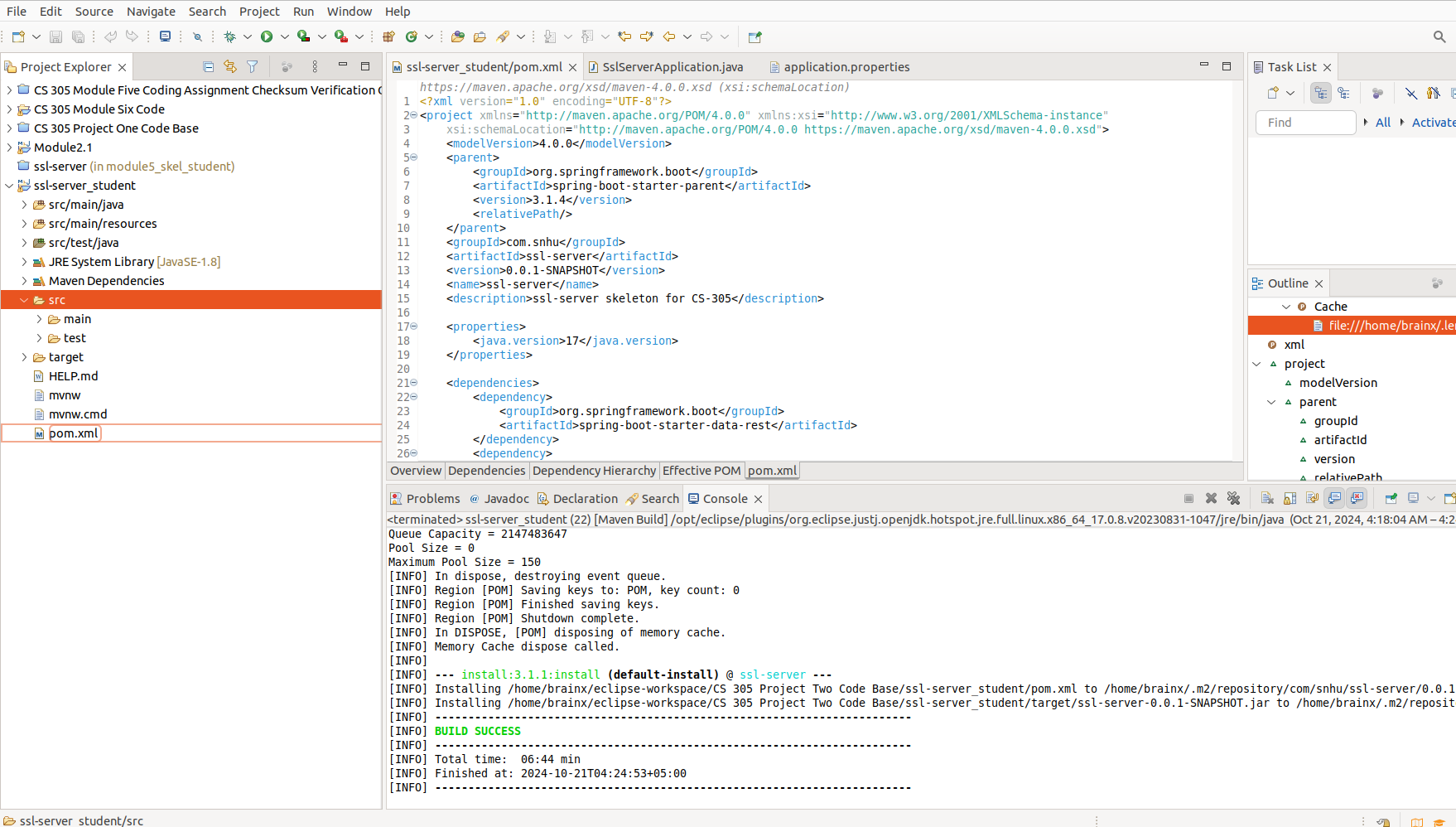
## Secondary Testing

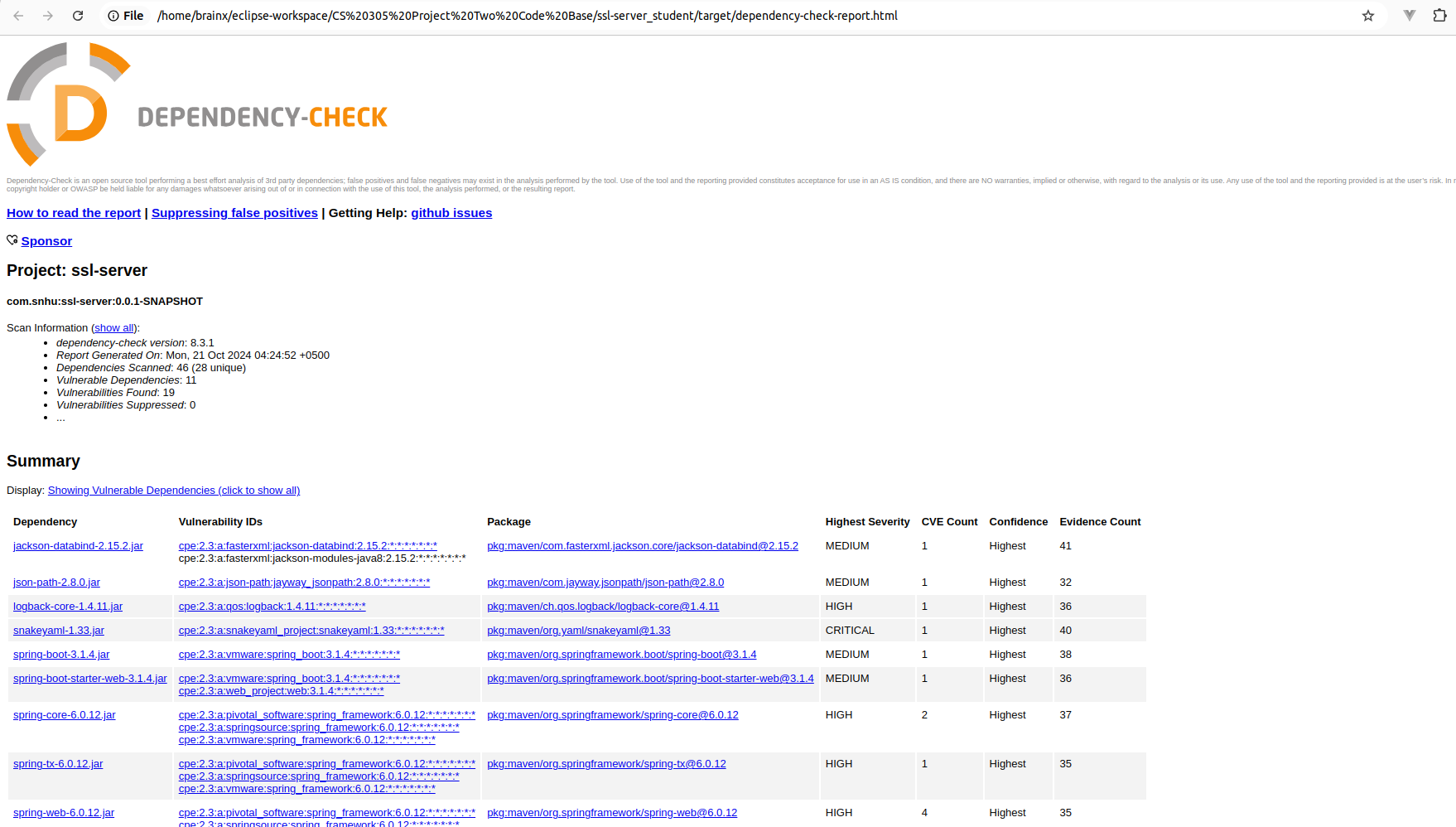
Insert screenshots below of the refactored code executed without errors and the dependency-check report.

Dependency check report before refactoring the code:



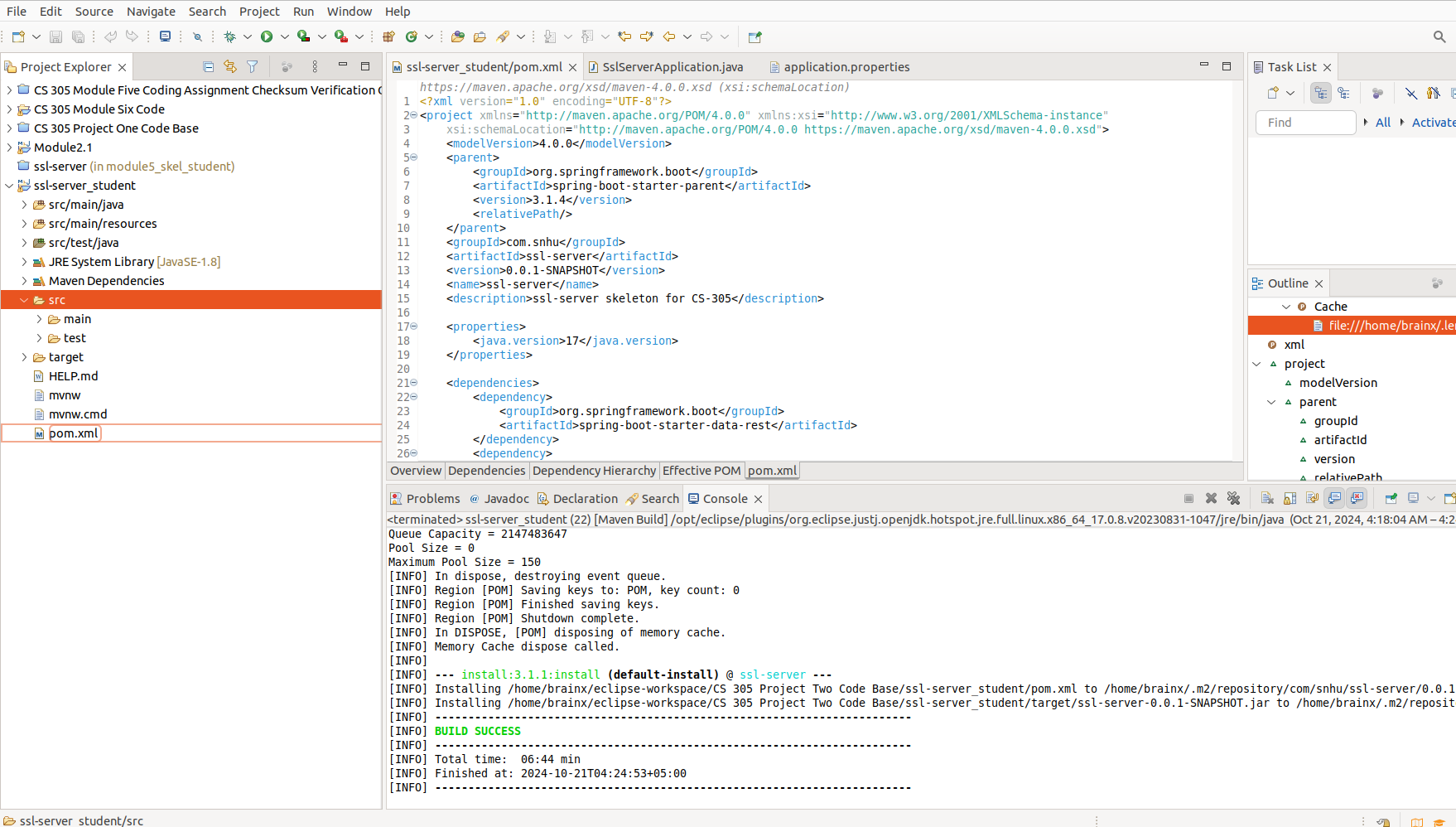
Code Refactored:

Dependency check report after refactoring:



## Functional Testing

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



## Summary

Refactoring Overview In this project, I focused on improving the security of Artemis Financial’s web application by refactoring key aspects to address potential vulnerabilities. The primary goals were to enhance encryption, ensure data integrity, secure communications, and resolve vulnerabilities in third-party dependencies. This refactoring strengthens the application’s overall security posture and ensures compliance with industry standards.

* Encryption: Implemented AES-256 encryption to secure sensitive client data such as financial plans and investments. AES-256, being a robust and efficient encryption standard, provides strong protection for both data at rest and in transit, mitigating risks of unauthorized access.
* Data Integrity: Introduced SHA-256 checksum verification to ensure the integrity of data transmitted between the client and the server. This mechanism detects any unauthorized changes to the data, ensuring it remains accurate and unaltered.
* Secure Communication: Refactored the application to enforce HTTPS using SSL/TLS, ensuring that all data exchanged between the web application and its users is encrypted. This prevents attackers from intercepting or tampering with sensitive communications.

Fixing Vulnerabilities As part of securing the application, I addressed vulnerabilities in third-party libraries flagged by OWASP Dependency Check. Key steps taken included:

* Updating Libraries: Vulnerable dependencies such as Jackson and Log4j were updated to secure versions, resolving known vulnerabilities that could have been exploited by attackers.
* Excluding Vulnerable Transitive Dependencies: In cases where vulnerabilities were introduced indirectly through transitive dependencies, I excluded them from the project to prevent security risks.
* Ensuring Version Consistency: Ensured all Spring-related dependencies were aligned to compatible versions, which prevented runtime errors and provided a stable, secure environment.
* Testing for Vulnerabilities: After updating and excluding vulnerable dependencies, I reran security testing to confirm that the application was secure and free from the previously identified vulnerabilities.

Security Testing Compliance To ensure the refactored code met security standards, I conducted thorough testing, covering both static analysis and functional validation:

* Static Testing: Used OWASP Dependency Check to identify and fix security vulnerabilities in the third-party libraries and ensure no insecure code was part of the project.
* Encryption Testing (AES-256): Verified that AES-256 was properly encrypting sensitive data, testing both encryption and decryption to ensure data security.
* Checksum Verification (SHA-256): Tested the checksum mechanism by simulating tampered data transmissions, confirming that altered data was detected.
* HTTPS Testing: Ensured SSL/TLS was correctly configured and that all communications were securely encrypted by accessing the application over HTTPS and verifying secure connections.

## Industry Standard Best Practices

Secure Coding Practices Throughout the refactoring process, I adhered to secure coding best practices to ensure Artemis Financial’s application was protected against potential threats:

* AES-256 Encryption: Applied AES-256, a highly regarded encryption algorithm, to protect sensitive financial data. AES-256 is used widely in financial services, healthcare, and other industries requiring secure data handling, offering a strong balance of performance and security.
* SHA-256 Hashing: Implemented SHA-256 to verify data integrity. SHA-256 is collision-resistant, ensuring that no two different inputs produce the same hash value, which helps prevent data tampering.
* SSL/TLS and HTTPS: Enforced HTTPS to secure communications, ensuring all data exchanged between the client and server is encrypted, authenticated, and protected from interception.

Impact on Artemis Financial The security enhancements made during this refactoring process have had a significant positive impact on Artemis Financial’s ability to protect sensitive client data and maintain its reputation as a trusted financial service provider.

* Data Protection: With AES-256 encryption in place, sensitive client information is now securely stored and transmitted, reducing the risk of data breaches or unauthorized access.
* Data Integrity: The integration of SHA-256 checksum verification ensures that client data remains accurate and untampered during transmission, protecting against costly errors and fraud.
* Compliance with Security Standards: By enforcing HTTPS and using SSL/TLS, the application is now aligned with security regulations such as PCI-DSS and GDPR. This not only reduces legal risks but also protects against potential regulatory penalties.
* Increased Client Trust: The enhancements in encryption, data integrity, and secure communications instill greater confidence in Artemis Financial’s clients, improving trust and strengthening the company’s competitive position in the industry.

By addressing vulnerabilities in dependencies, enhancing encryption, and securing communications, the refactored application provides a strong foundation for protecting sensitive financial data and ensuring that Artemis Financial continues to meet industry standards for secure software development.

**References:**

* Eastlake, D., & Hansen, T. (2011). *US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)* (RFC 6234). Internet Engineering Task Force (IETF). <https://doi.org/10.17487/RFC6234>
* National Institute of Standards and Technology (NIST). (2001). *Announcing the Advanced Encryption Standard (AES)* (Federal Information Processing Standards Publication 197). <https://doi.org/10.6028/NIST.FIPS.197>
* Rescorla, E. (2018). *The Transport Layer Security (TLS) Protocol Version 1.3* (RFC 8446). Internet Engineering Task Force (IETF). <https://doi.org/10.17487/RFC8446>
* OWASP Foundation. (2023). *OWASP Dependency-Check*. Retrieved from <https://owasp.org/www-project-dependency-check/>