GLOBALRAIN

Practices for Secure Software Report

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	2/26/2024	Felix Carela	

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
Felix Carela

1. Algorithm Cipher

Overview of AES

The Advanced Encryption Standard (AES) is a symmetric encryption algorithm widely recognized for its robustness and efficiency. It has been adopted as a standard by the U.S. government and numerous other entities across the globe for securing sensitive data. AES operates on fixed block sizes of 128 bits, with key sizes of 128, 192, or 256 bits, offering a good balance between security and performance.

Hash Functions and Bit Levels

While AES itself is not a hash function but an encryption standard, it's important to note that in the context of Artemis Financial's requirements, encryption works hand-in-hand with hash functions for data integrity and verification. For hashing, a complementary algorithm like SHA-256 (Secure Hash Algorithm 256-bit) is recommended. SHA-256 is part of the SHA-2 family and is known for its strong hash properties, ensuring that the data integrity is maintained.

Use of Random Numbers, Symmetric vs. Non-Symmetric Keys

AES is a symmetric key algorithm, meaning it uses the same key for both encryption and decryption. This approach necessitates secure key management practices. The use of random numbers is crucial in generating strong, unpredictable keys. Proper entropy sources and key management strategies must be implemented to prevent key prediction or duplication.

History and Current State of Encryption Algorithms

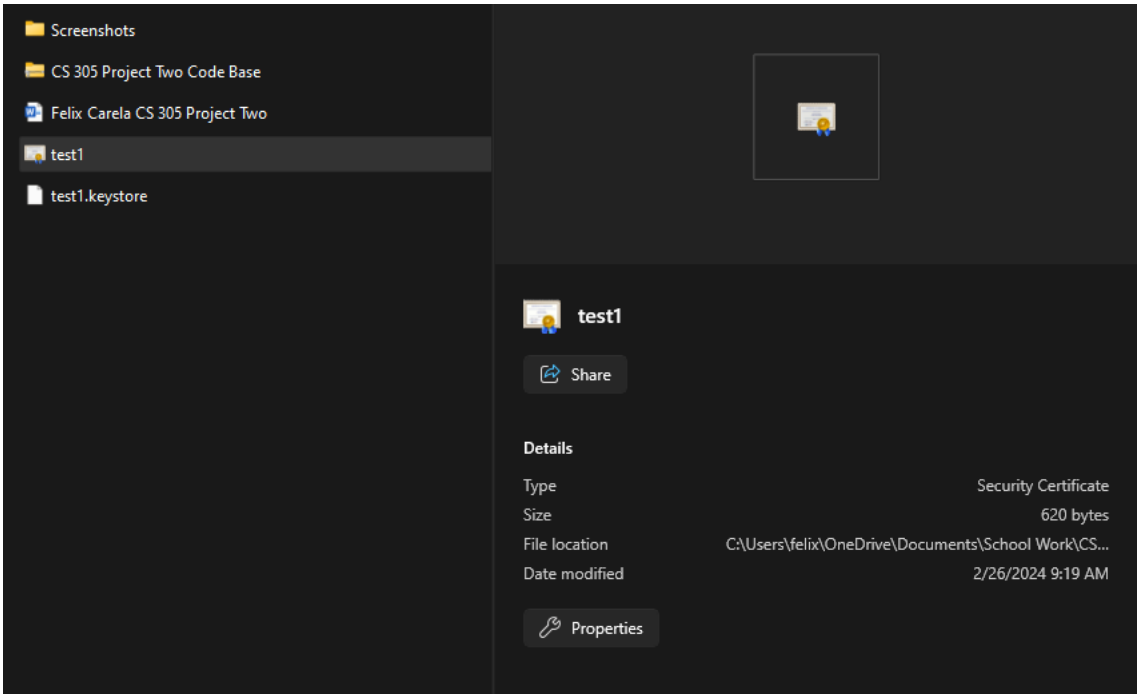
Encryption algorithms have evolved significantly over time. The need for robust encryption grew with the advent of the digital era, leading to the development of various algorithms. AES was established by the U.S. National Institute of Standards and Technology (NIST) as a successor to the older DES (Data Encryption Standard) (Oswald, 2022). Since its adoption, AES has stood the test of time, proving resistant to most forms of cryptographic attacks. Its efficiency in both software and hardware implementations has made it the go-to choice for securing sensitive data in diverse applications.

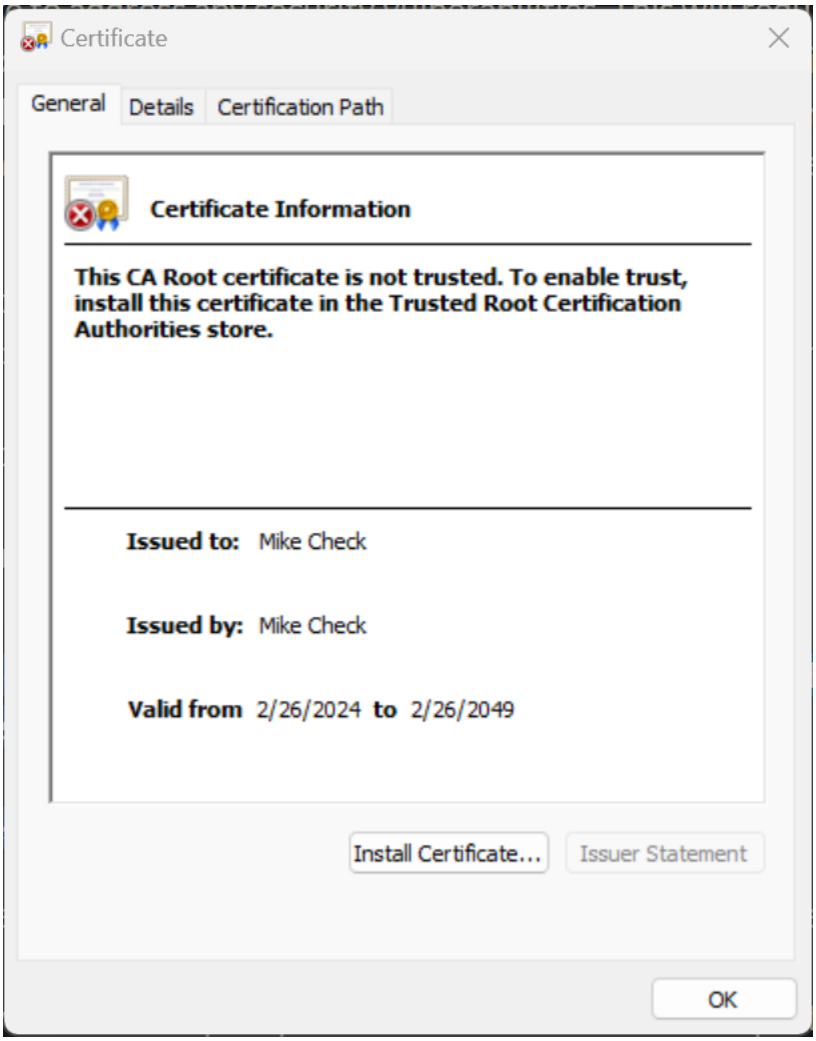
In the current landscape, AES continues to be regarded as highly secure. It's used in various protocols like SSL/TLS for secure internet communications, VPNs for secure remote access, and by government agencies for protecting classified information (Oswald, 2022). Given the sensitive nature of financial data handled by Artemis Financial, AES, with its proven track record and robust security, is an appropriate choice. It provides a high level of security while

maintaining efficient performance, essential for processing financial transactions and data securely and swiftly.

2. Certificate Generation

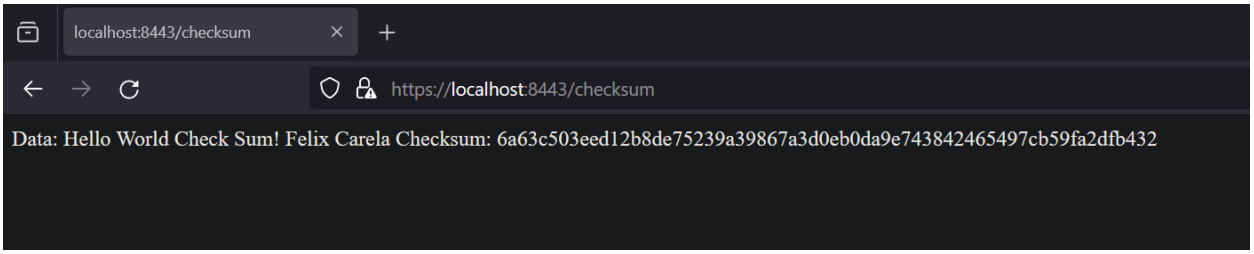
Insert a screenshot below of the CER file.





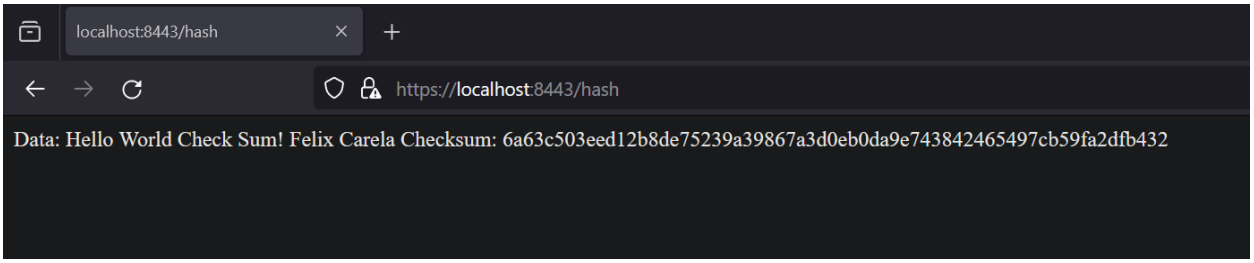
3. Deploy Cipher

Insert a screenshot below of the checksum verification.



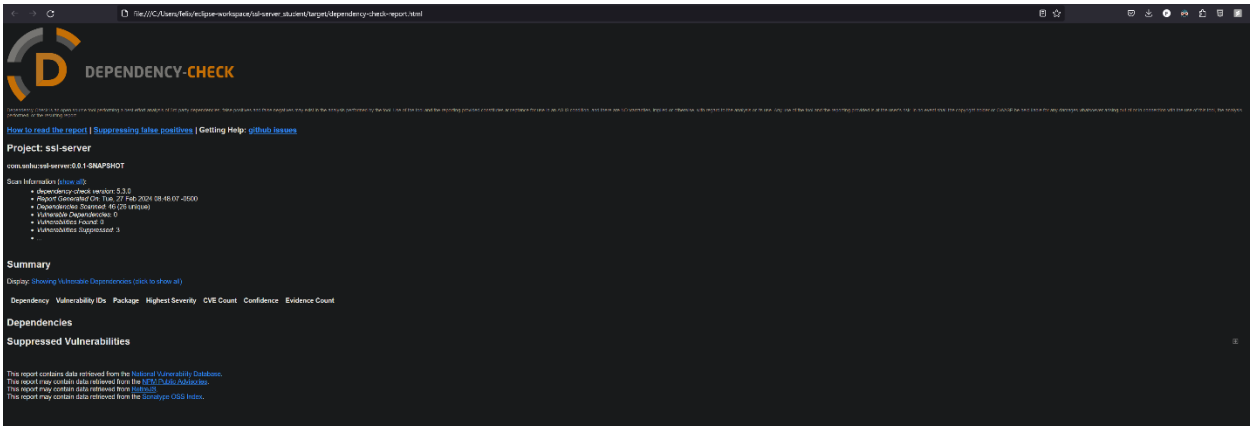
4. Secure Communications

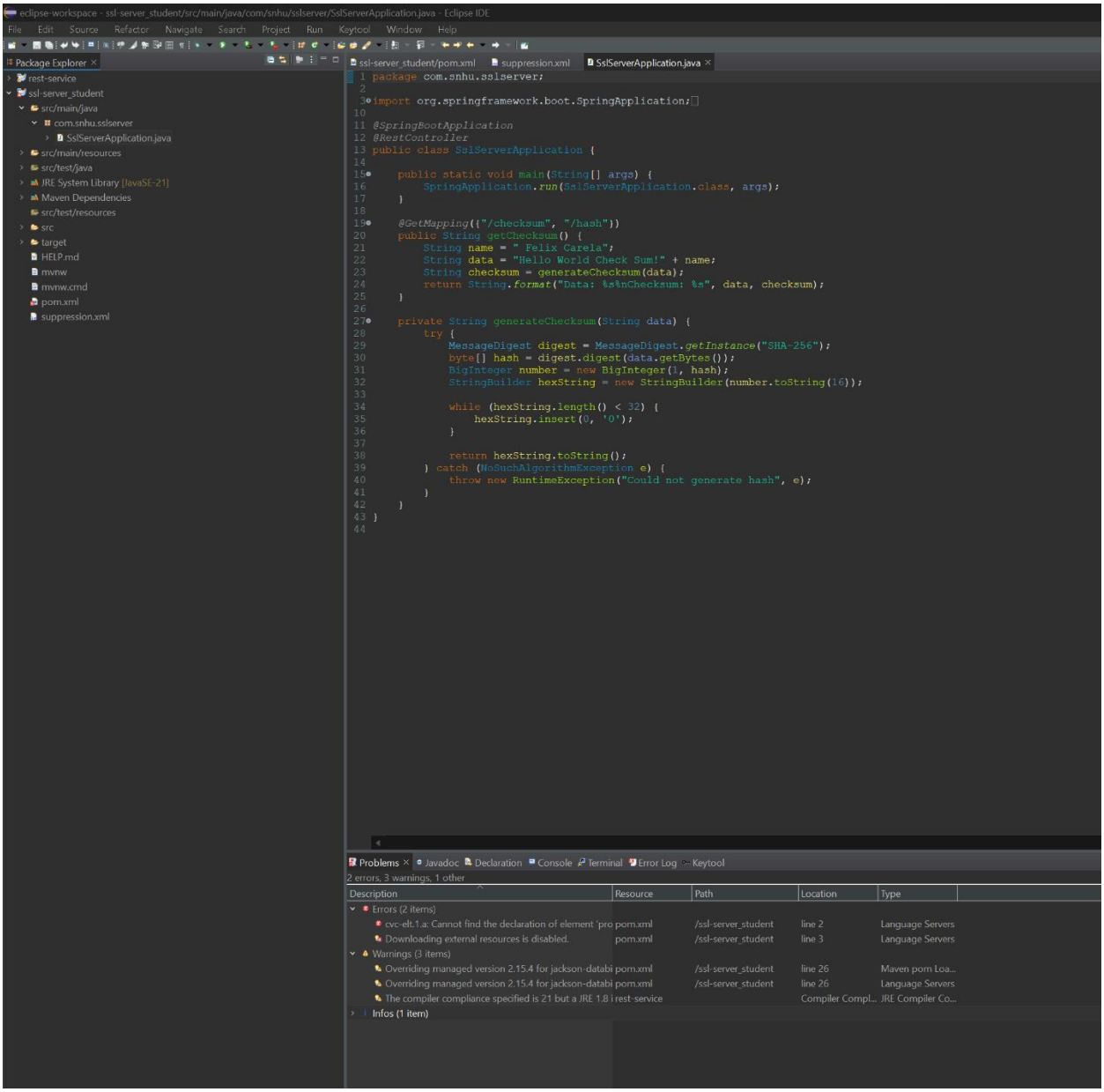
Insert a screenshot below of the web browser that shows a secure webpage.



5. Secondary Testing

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

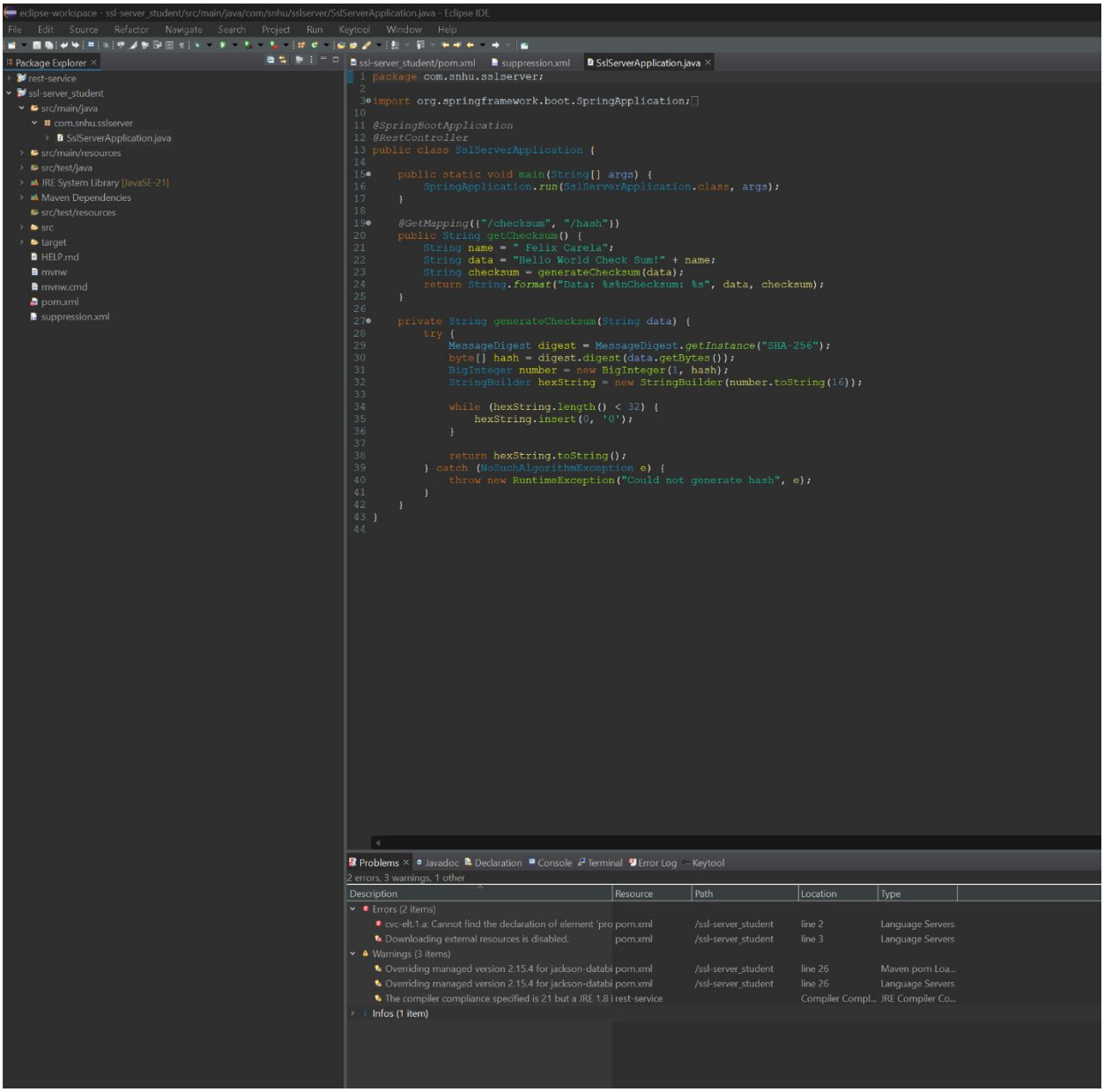




The errors are not for the refactored code but for the pom.xml file.

6. Functional Testing

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



The errors are not for the refactored code but for the pom.xml file.

7. Summary

The refactoring of the code for the SSL Server Application has been informed by a thorough vulnerability assessment process. Initially, an architecture review was used to analyze the application's structure, ensuring it conformed to best practices for security. This review included scrutinizing the input validation methods to secure input and representation, thereby preventing common vulnerabilities such as SQL injection and cross-site scripting (XSS).

Attention was also given to the interactions with APIs, which were secured to prevent unauthorized access and data leaks. This is critical as APIs often serve as gateways to sensitive

data and functionality within applications. Furthermore, various aspects of the code were reviewed, including the controller, view, data access layers, plug-ins, and APIs. This code review process helped identify and mitigate potential vulnerabilities within each component.

Cryptography practices were assessed, with an emphasis on the proper use of encryption and the identification of any cryptographic vulnerabilities. This ensures that data is encrypted in transit and at rest, reducing the risk of data breaches and exposure of sensitive information.

Secure coding practices and patterns were employed to enhance code quality, including the use of encapsulation to create secure data structures. This not only aids in the prevention of security flaws but also contributes to the maintainability and scalability of the codebase.

Finally, the architecture review and outcomes from static testing informed which manual code reviews were necessary, leading to a comprehensive mitigation plan. The summary of findings from this extensive process has laid the foundation for a more secure and robust SSL Server Application, designed to withstand a variety of security threats.

8. Industry Standard Best Practices

To adhere to industry standard best practices for secure coding and mitigate against known security vulnerabilities, the following steps were applied within the software application:

- **Application Architecture Analysis**: The architecture of the application was thoroughly reviewed, considering how different components interact and how data flows through the system. This ensures that any architectural vulnerabilities can be identified and addressed early in the development process.
- **Input Validation**: All inputs to the application were validated to prevent common vulnerabilities such as SQL injection, cross-site scripting (XSS), and command injection. Secure input handling prevents malicious data from impacting the application or backend systems.
- **API Security**: The application's API interactions were secured to ensure that only authenticated and authorized users could access sensitive functions and data. This step reduces the risk of unauthorized access and potential data breaches.
- **Cryptography Usage**: Cryptographic practices were evaluated, ensuring that data is encrypted in transit and at rest. This safeguards sensitive data from interception or unauthorized access.
- **Secure Error Handling**: The application was designed to handle errors securely, not revealing sensitive information in error messages that could be exploited by an attacker.
- **Code Review and Quality**: Regular code reviews should be conducted to ensure that the codebase adheres to secure coding standards. This includes reviewing the code for models, views, controllers, services, and plugins.
- **Encapsulation and Secure Data Structures**: The application's data structures were designed with security in mind, ensuring that data is encapsulated and can only be accessed through well-defined interfaces.

Applying these best practices is essential not only to maintain the current security posture of the software but also for the company's overall wellbeing. It helps to prevent data breaches, which can be costly in terms of financial loss, reputation damage, and legal liability. By proactively addressing vulnerabilities and adhering to security best practices, the company can ensure the integrity, confidentiality, and availability of its data and services, maintaining trust with customers and stakeholders.

References

Oswald, E. (2022, December 16). *What is the Advanced Encryption Standard (AES)? | U.S. news*. U.S. News. https://www.usnews.com/360-reviews/privacy/what-is-advanced-encryption-standard