****

# CS 305 Project Two

**Practices for Secure Software Report**

Table of Contents

[Document Revision History 3](#_Toc33111302)

[Client 3](#_Toc33111303)

[Instructions 3](#_Toc33111304)

[Developer 4](#_Toc33111305)

[1. Algorithm Cipher 4](#_Toc33111306)

[2. Certificate Generation 4](#_Toc33111307)

[3. Deploy Cipher 4](#_Toc33111308)

[4. Secure Communications 4](#_Toc33111309)

[5. Secondary Testing 4](#_Toc33111310)

[6. Functional Testing 5](#_Toc33111311)

[7. Summary 5](#_Toc33111312)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **12/12/2021** | **Oliver Milani** |  |

## Client



## Instructions

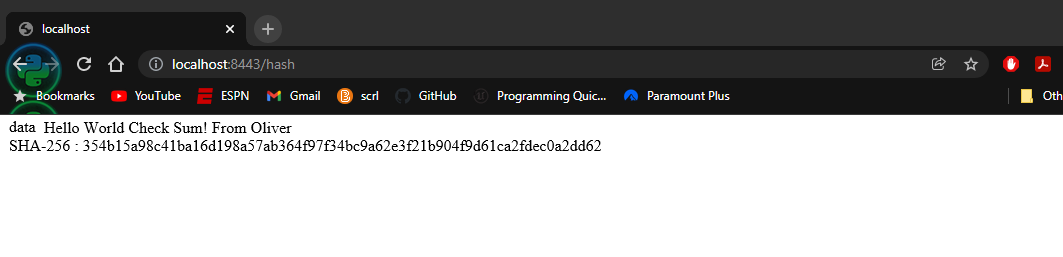
Deliver this completed Practices for Secure Software Report documenting your process for writing secure communications and refactoring code that complies with software security testing protocols.

Respond to the steps outlined below and replace the bracketed text with your findings in your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

## Developer

Oliver Milani

## 1. Algorithm Cipher

Determine an appropriate encryption algorithm cipher to deploy given the security vulnerabilities, justifying your reasoning. Be sure to address the following:

* Provide a brief, high-level overview of the encryption algorithm cipher.
* Discuss the hash functions and bit levels of the cipher.
* Explain the use of random numbers, symmetric vs non-symmetric keys, and so on.
* Describe the history and current state of encryption algorithms.

Considering the needs of Artemis Financial, I believe the Secure Hash Algorithm 256 is a great choice for security protection. SHA-256 revolves around encrypting sensitive data, which is Artemis Financials’ number one priority and area of possible vulnerability. SHA is also used by the government and follows guidelines and rules set by the government to make sure the company is in compliance. This is a big concern for any financial organization so the fact that it will meet government regulations is a big factor.

There are some risks to this approach. SHA is one of the most common ciphers, therefore there are a lot of people that understand it and can potentially know when to force their way into a system. The key that is used for the cipher is universal for encryption and decryption for the whole system, this means it will be transferred between people which leaves for vulnerable moments. If transferring of this key is not done securely, any bad actors will have easy access to any files.

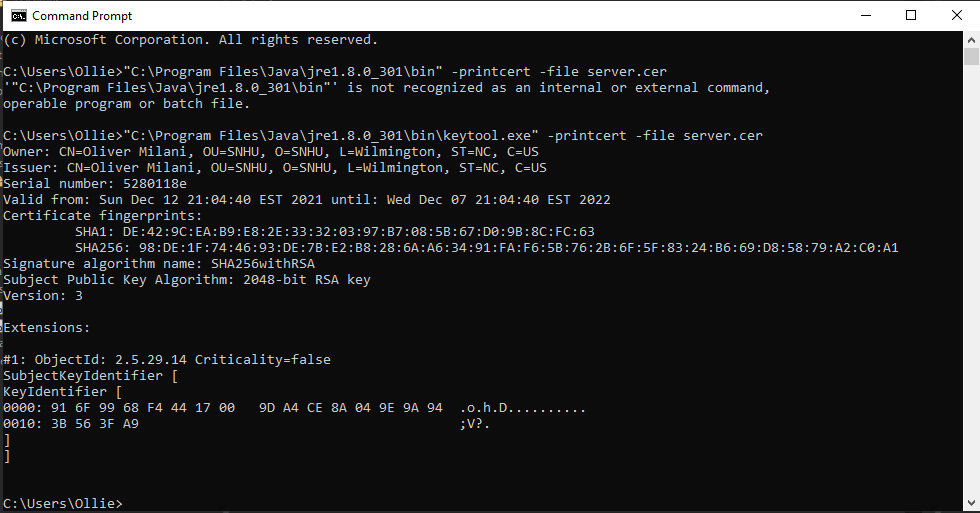
The bit levels in a cipher are essentially the size of the key and also the amount of operations it would take for someone to attempt a brute force attack. The bit is represented as a number and is what the key is comprised of. This key is what will be used to decrypt the information. The hash function will take in information and assign that information with a unique identifier, using the key, and store it. This means that it takes the information from plain text and applies a sort of “code” to it that can only be read using the key.

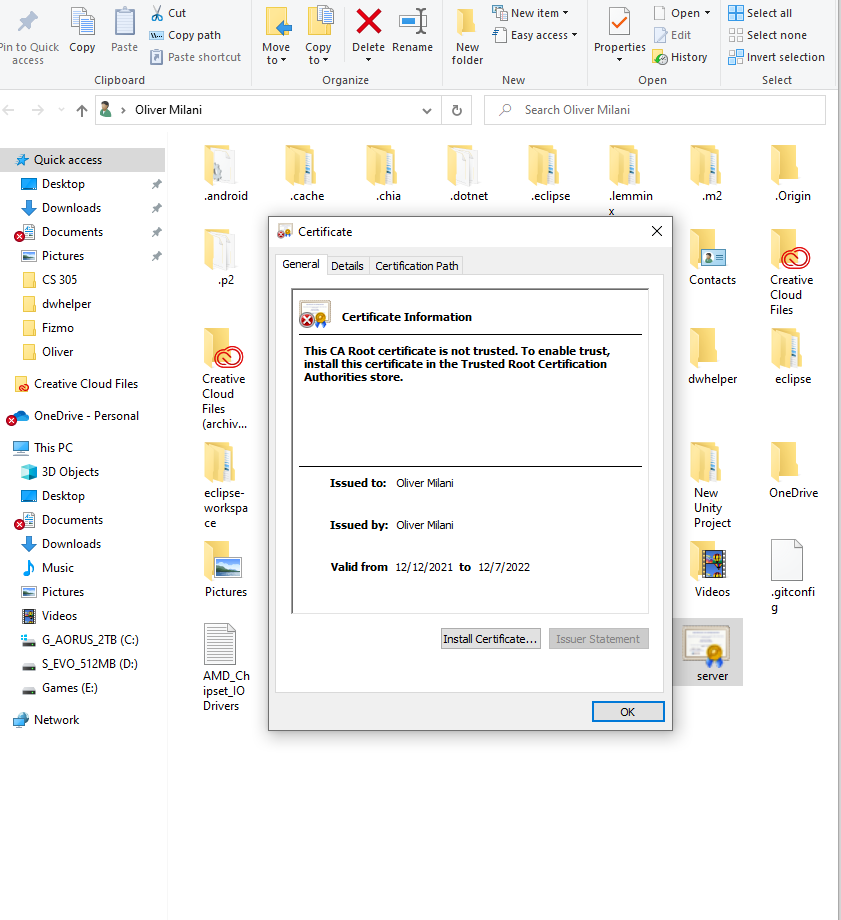
Symmetric encryptions are simple and most used encryption techniques. It works as described above, there is a key that encrypts and decrypts plaintext. An Asymmetric encryption has two keys, a public key and a private key. Each key can encrypt and decrypt but only in certain ways. In general, asymmetric encryption is going to take a lot longer. Asymmetric encryption is a newer type of encryption and get grow to get better over time.

There are some encryptions that have been around for a very long time. The idea of encryption is said to date back to the Egyptians. It has been studied that they possibly have purposefully swapped symbols or parts of symbols to hide the true meaning behind the writing. In the more modern era, we start seeing a lot of encryption start with World War 2 as countries attempt to transfer information and battle instructions without enemies being able to understand. The keys for these types of encryptions were fought over themselves with the most famous being the German Enigma cipher code that was solved by Alan Turing. Now we have modern adaptations and evolutions of old encryptions and they are continuing to evolve going forward.

## 2. Certificate Generation

Generate appropriate self-signed certificates using the Java Keytool, which is used through the command line.

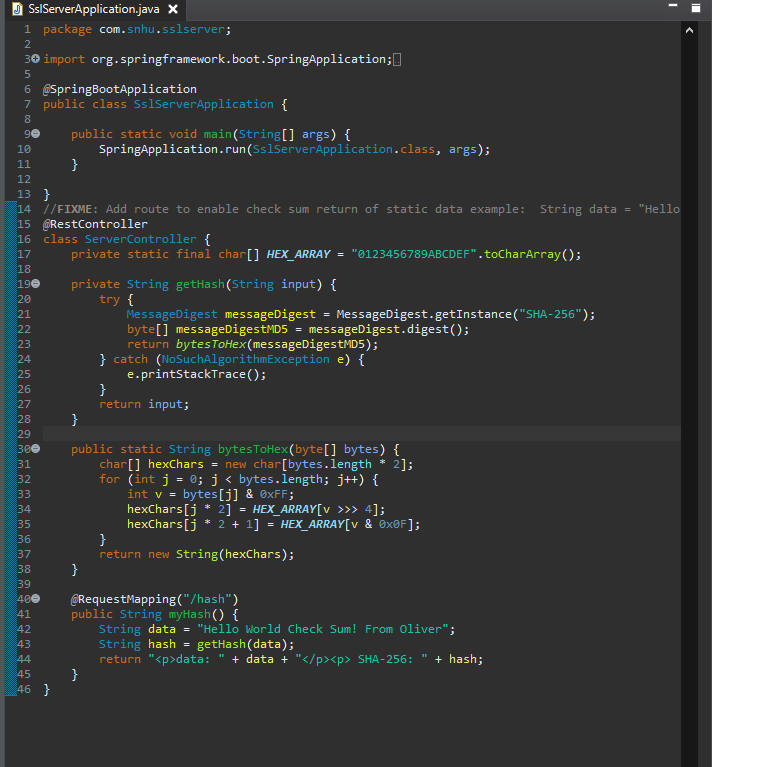




## 3. Deploy Cipher

Refactor the code and use security libraries to deploy and implement the encryption algorithm cipher to the software application. Verify this additional functionality with a checksum.

* Insert a screenshot below of the checksum verification. The screenshot must show your name and a unique data string that has been created.

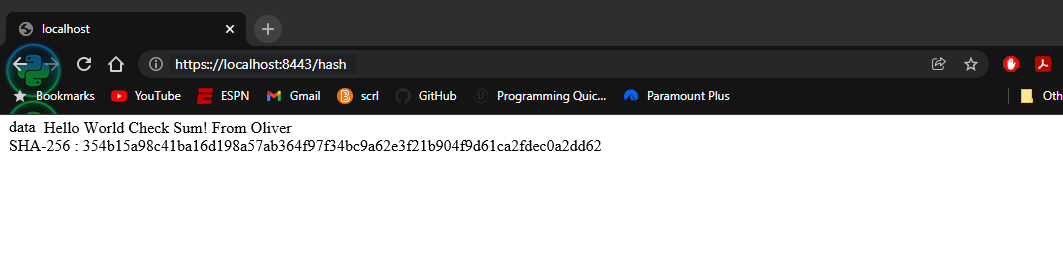


## 

## 4. Secure Communications

Refactor the code to convert HTTP to the HTTPS protocol. Compile and run the refactored code to verify secure communication by typing **https://localhost:8443/hash** in a new browser window to demonstrate that the secure communication works successfully.

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



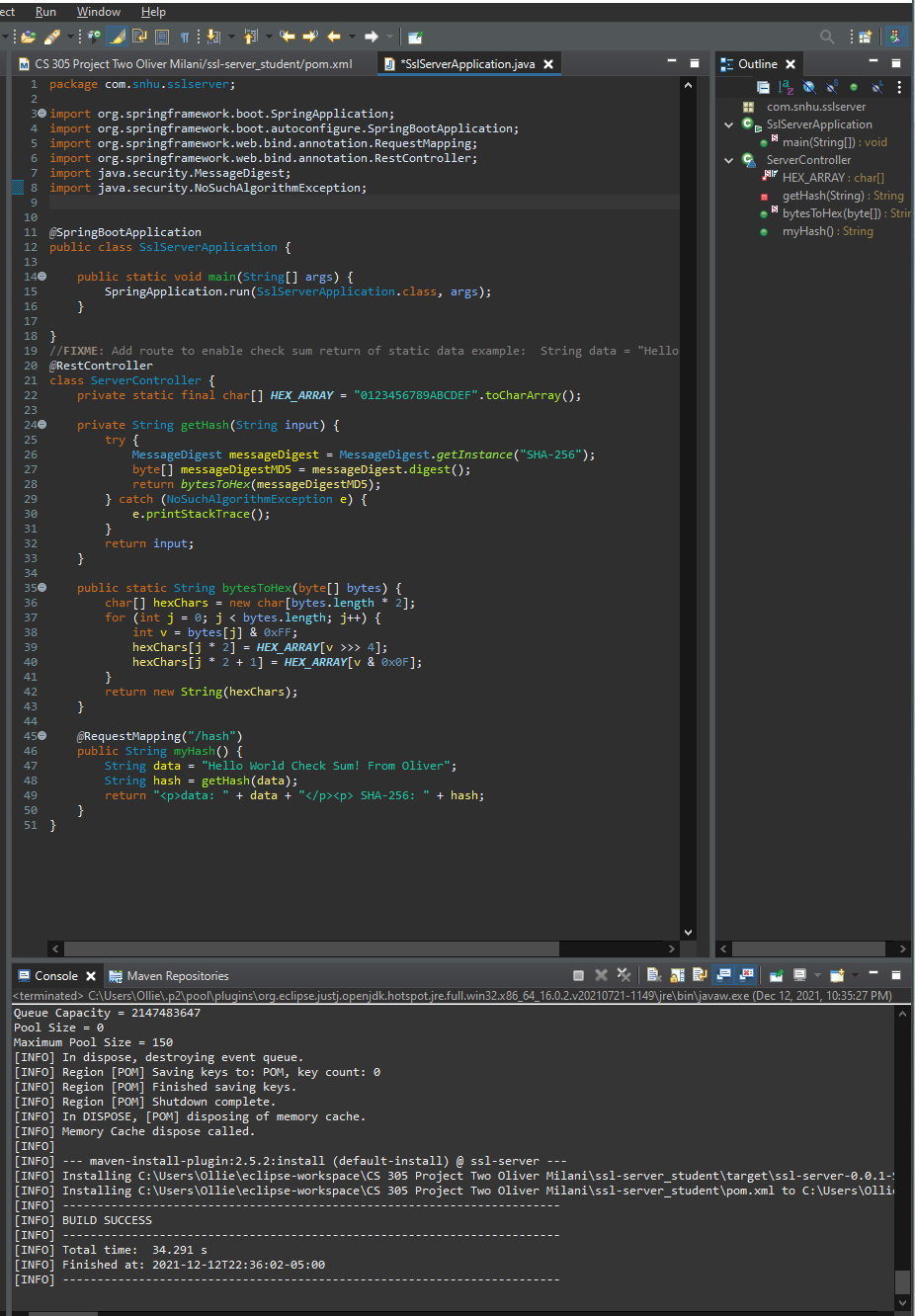
## 

## 5. Secondary Testing

Complete a secondary static testing of the refactored code using the dependency check tool to ensure code complies with software security enhancements. You only need to focus on the code you have added as part of the refactoring. Complete the dependency check and review the output to ensure you did not introduce additional security vulnerabilities.

* Include the following below:
  + A screenshot of the refactored code executed without errors
  + A screenshot of the dependency check reportGraphical user interface, text

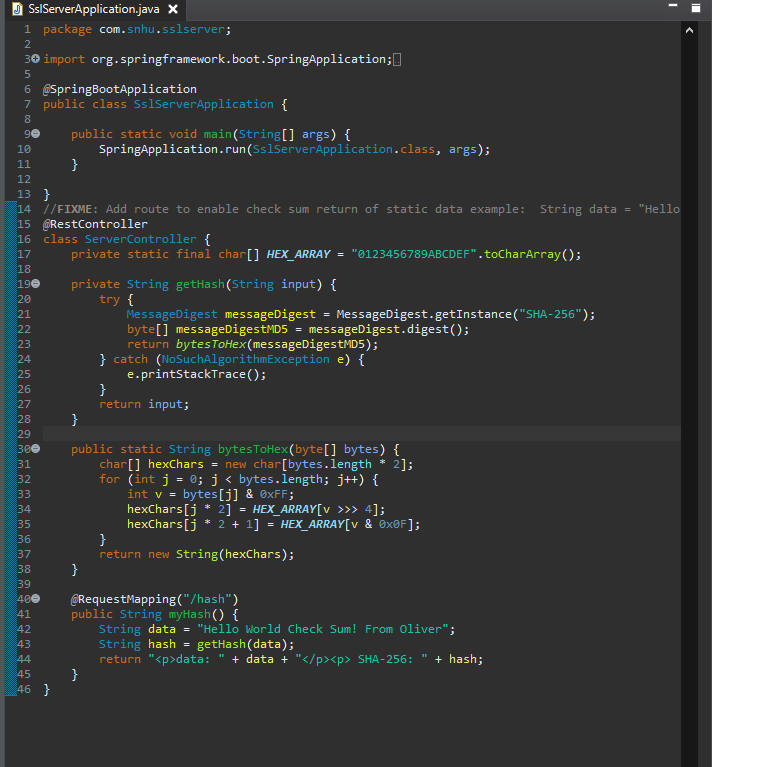
    Description automatically generated



## 6. Functional Testing

Identify syntactical, logical, and security vulnerabilities for the software application by manually reviewing code.

* Complete this functional testing and include a screenshot below of the refactored code executed without errors.



## 

## 7. Summary

Discuss how the code has been refactored and how it complies with security testing protocols. Be sure to address the following:

* Refer to the Vulnerability Assessment Process Flow Diagram and highlight the areas of security that you addressed by refactoring the code.
* Discuss your process for adding layers of security to the software application and the value that security adds to the company’s overall wellbeing.
* Point out best practices for maintaining the current security of the software application to your customer.

I made adjustments by refactoring the code and how it compiles with security testing protocols by using a secured RestContoller and chose SHA256 as the cipher. The RestController is used as the secure controller for RESTful. SHA256 allowed me to have a hashing cipher that wouldn’t run into any issues. SHA256 is efficient and has a low probability of collisions. This is a huge factor for the company’s overall wellbeing. Collisions within a financial company can lead to some big issues with important real world consequences. SHA256 gives the company the confidence that people’s money will be secure and the customers can trust them. The best practices that they can use to help maintain the current security is to consistently make updates using the guidance of dependency checks. The dependency check allows for an accurate view of where any growing concerns will be found.