****

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

## 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

## 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.

**AES Overview:**

Here we present the advanced encryption standard (AES) as a modern cryptography system and compliant with government regulation. According to the Federal Information processing standards 197 (2001), AES specifies an approved cryptographic algorithm that can be used to protect electronic data. The standards became effective on May 26, 2002. The AES algorithm is a symmetric block cipher which means encryption happens on fixed-length groups of 128 bits supporting key lengths of 128, 192, and 256. The algorithm takes as input each block of our plaintext converted in binary or hexadecimal and a key and output a ciphered block.

**What is the purpose of hash functions and bit-level of cipher?**

The purpose of a hash function is to store or retrieve data items with a key. Hash functions index the hash table with the key associated with the data items. Thus, a hash function maps a key to content. There are used for integrity verification, challenge handshake authentication, digital signatures, etc.

Here, we recommend secure hash Algorithm SHA-256 or SHA-512 or the latest version of SHA3 family to produce irreversible and unique hashes. The bit level of cipher informs on the digest size and is computed respectively with eight 32bits (SHA-256) and eight 64 bits words(SHA-512). Even though SHA-256 and SHA-512 have identical structures, the two functions' initial values are different and the number of rounds comprised between 24 and 64 for SHA-256 and 24 and 80 for SHA-512. The larger the bit level is, the more secure the function is against collision attacks.

**Explain the use of random numbers, symmetric vs. non-symmetric keys, and so on?**

Random numbers are unpredictable numbers algorithmically generated as keys and used to improve message or data authentication quality(pin, password, primes, key confirmation, .etc).

The symmetric algorithm is used for encryption rather than signing with only one key(Payment application, card transaction, random numbers generation). On the other hand, the asymmetric algorithm used two keys (public and private) for signing data.

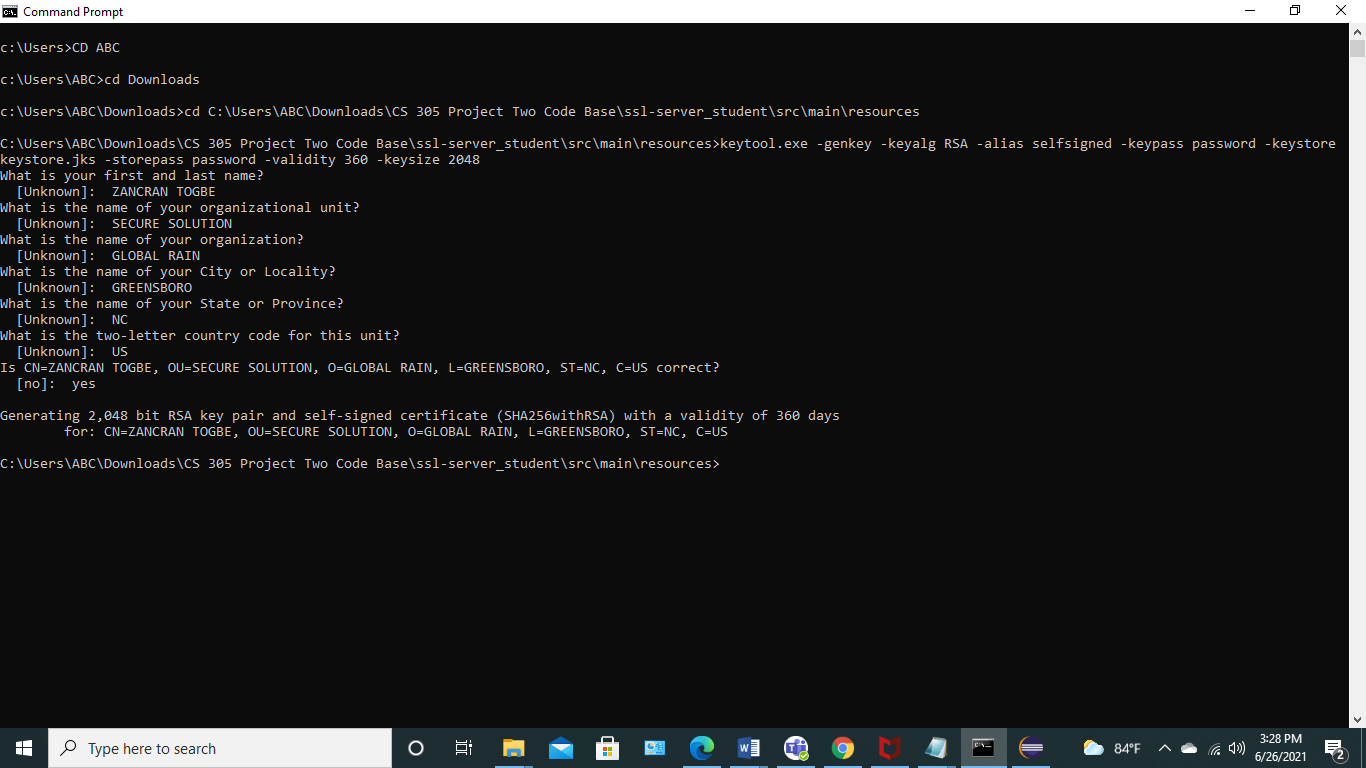
**Describe the history and current state of the encryption algorithm**

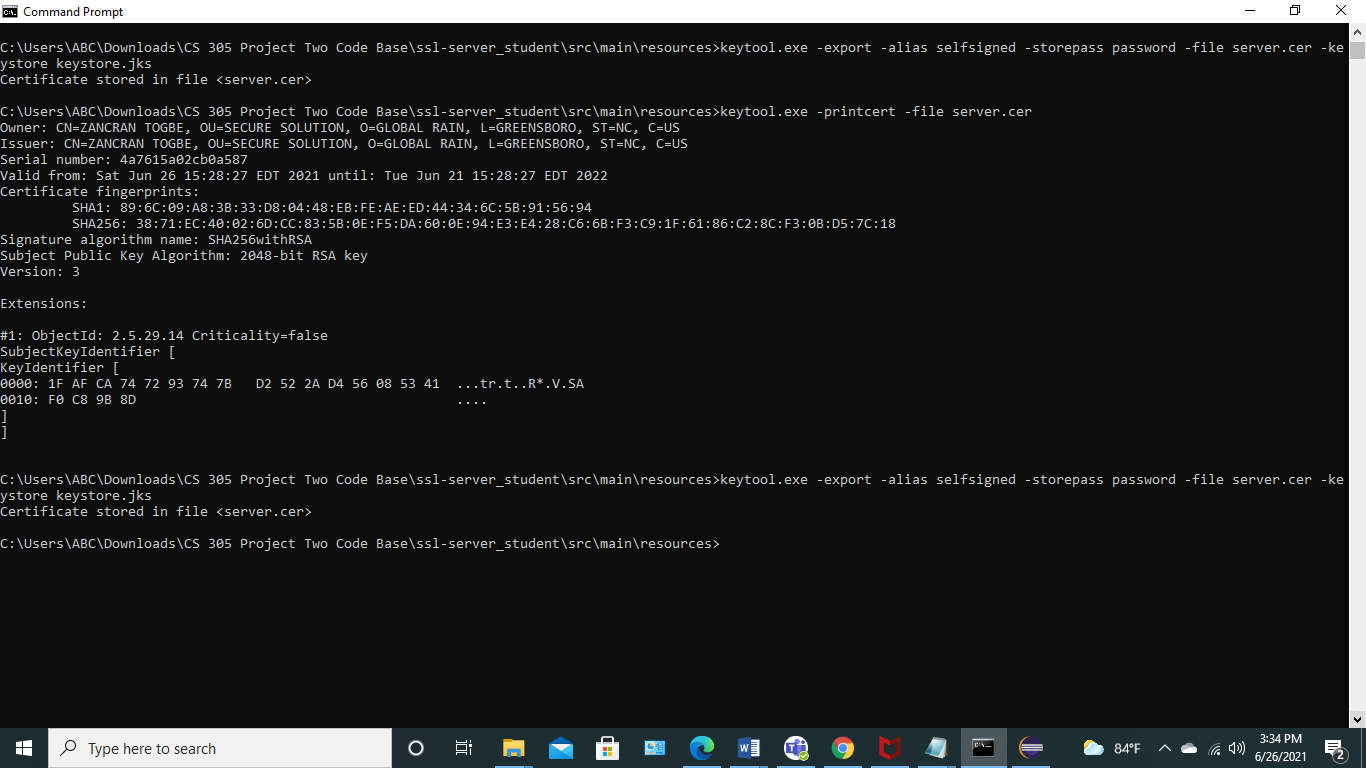
According to (Gustavus J. Simmons, 2016), there have been three well-defined phases in the history of cryptology. The first was the period of manual cryptography (cuneiform tablets, papyruses, scytale), where ciphers were limited to at most a few pages. The second phase is the mechanization of cryptography, starting after world war I and continues even today. The applicable technology involved telephone, telegraph, calculating machines. After eight decades, in 1999, the US designed and fabricated a single silicon chip implementation of the Data Encryption algorithm, later replaced by the Advanced encryption algorithm due to security issues. The third phase, dating only to the last two decades of the 20th century, marked the extension of cryptology to the information age: digital signatures, authentication, shared or distributed capabilities to exercise cryptologic functions. Today, on the NIST website, there is a lot of article on the different project such as post-quantum cryptography, multi-party threshold cryptography, random bit generation, and privacy-enhancing cryptography.

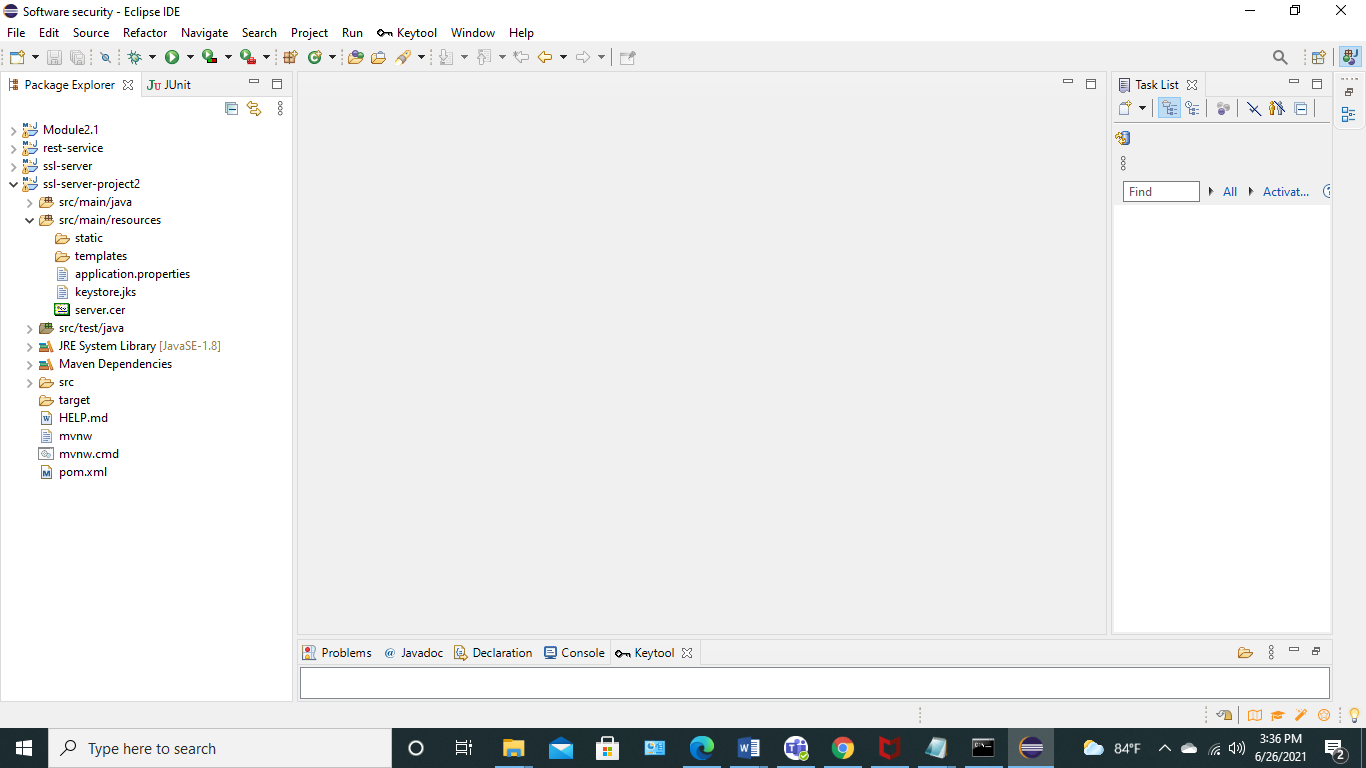
## 2. Certificate Generation

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

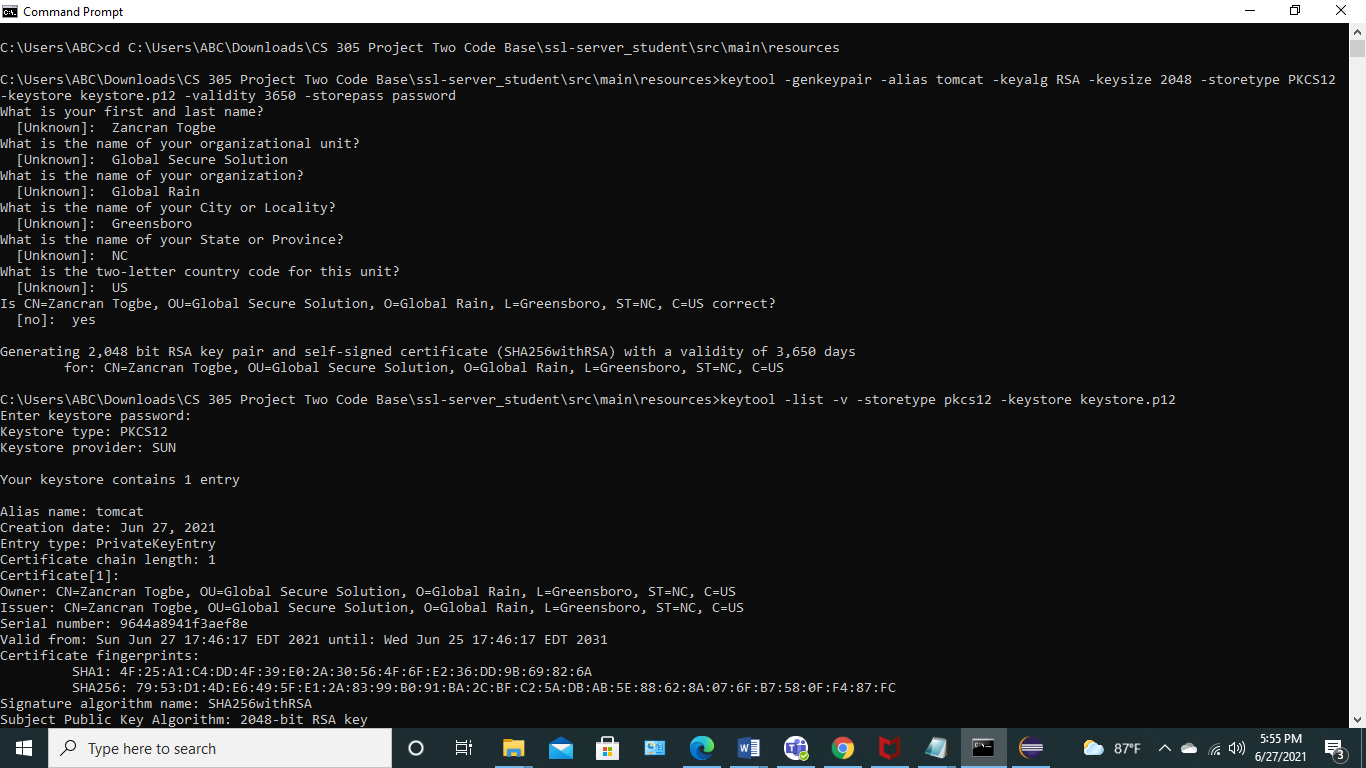
* To demonstrate that the keys were effectively generated, export your certificates (CER file) and submit a screenshot of the CER file below.

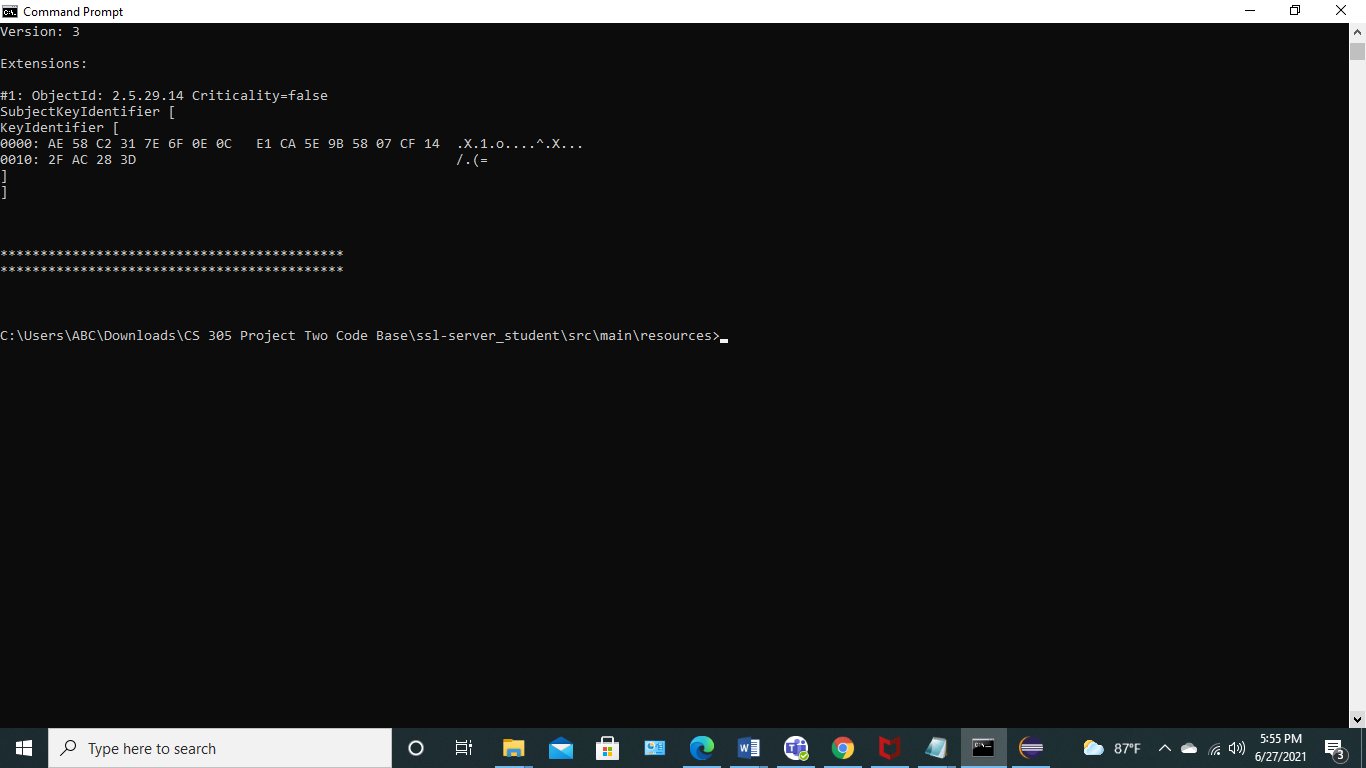






Generating keystore.p12

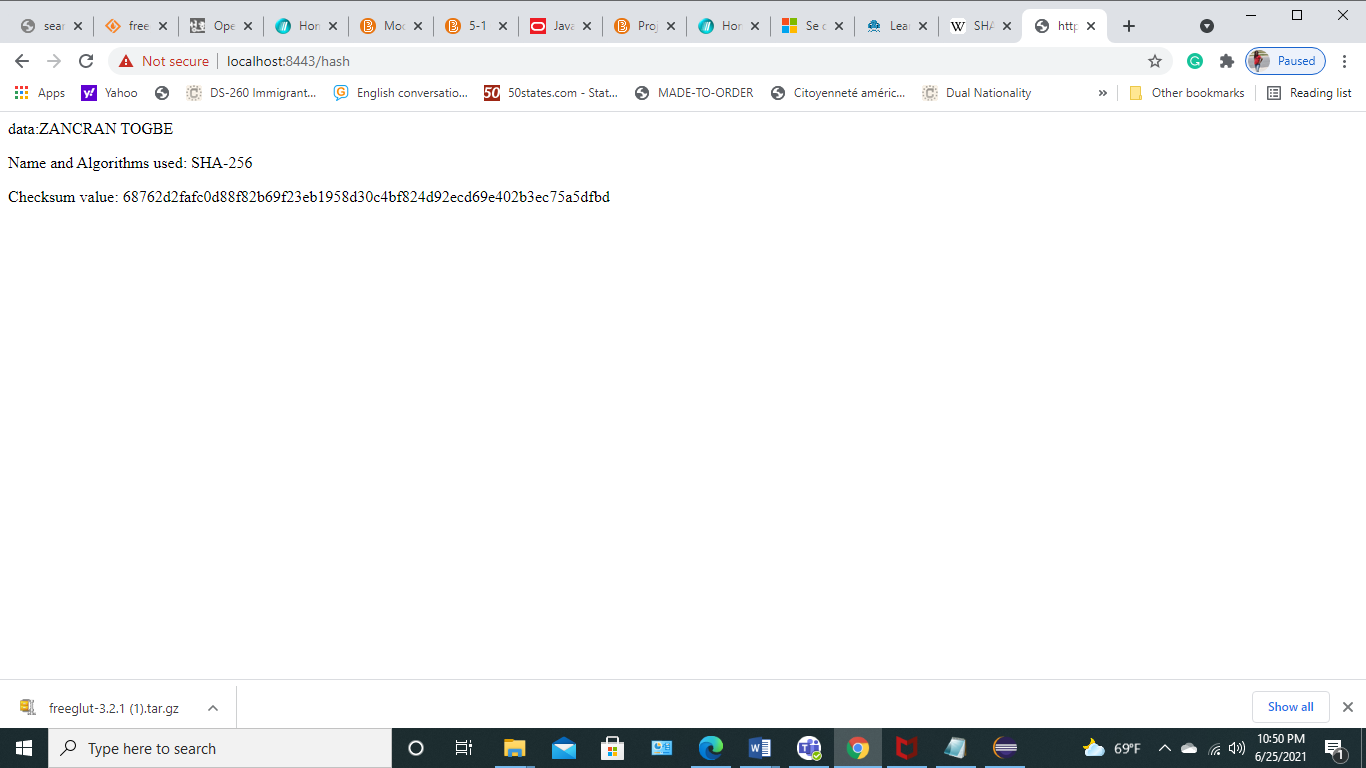




## 3. Deploy Cipher

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.

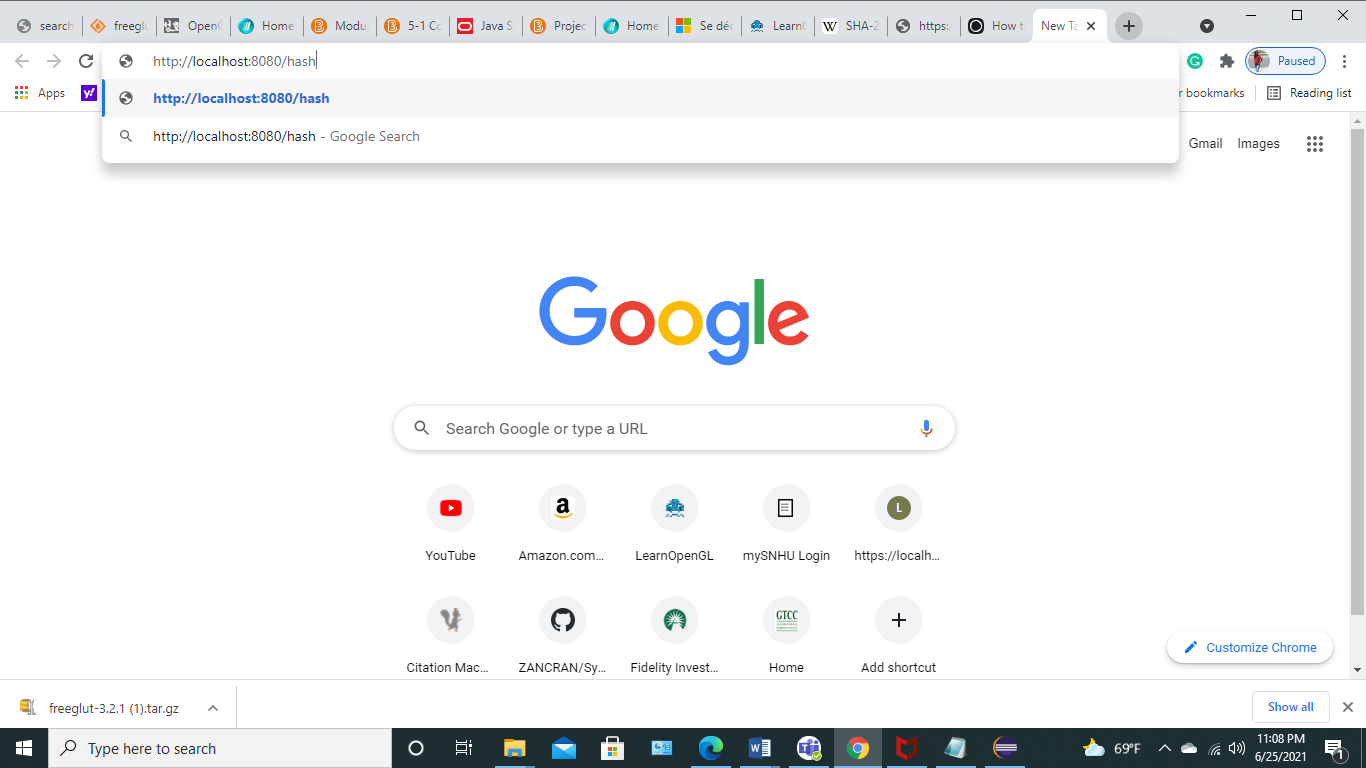
* Insert a screenshot below of the checksum verification. The screenshot must show your name and a unique data string that has been created.



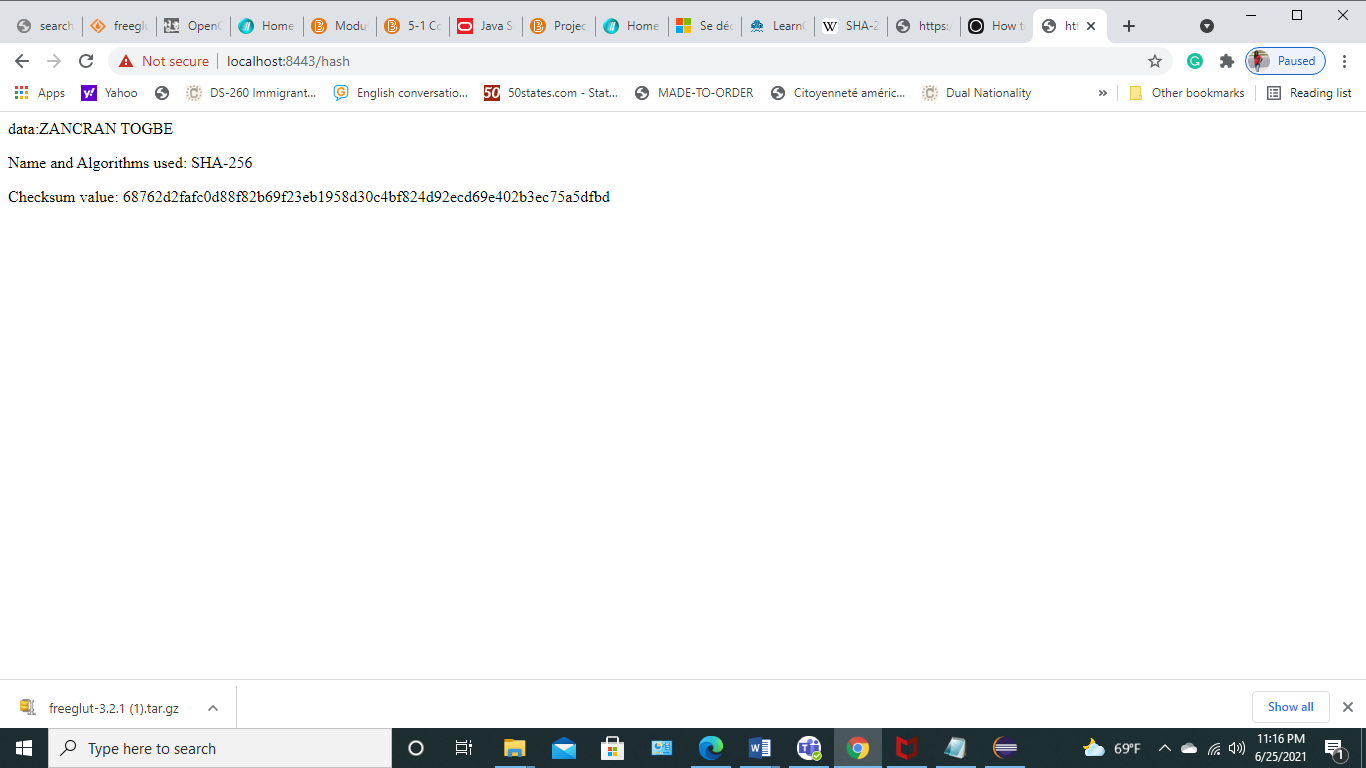
## 4. Secure Communications

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.



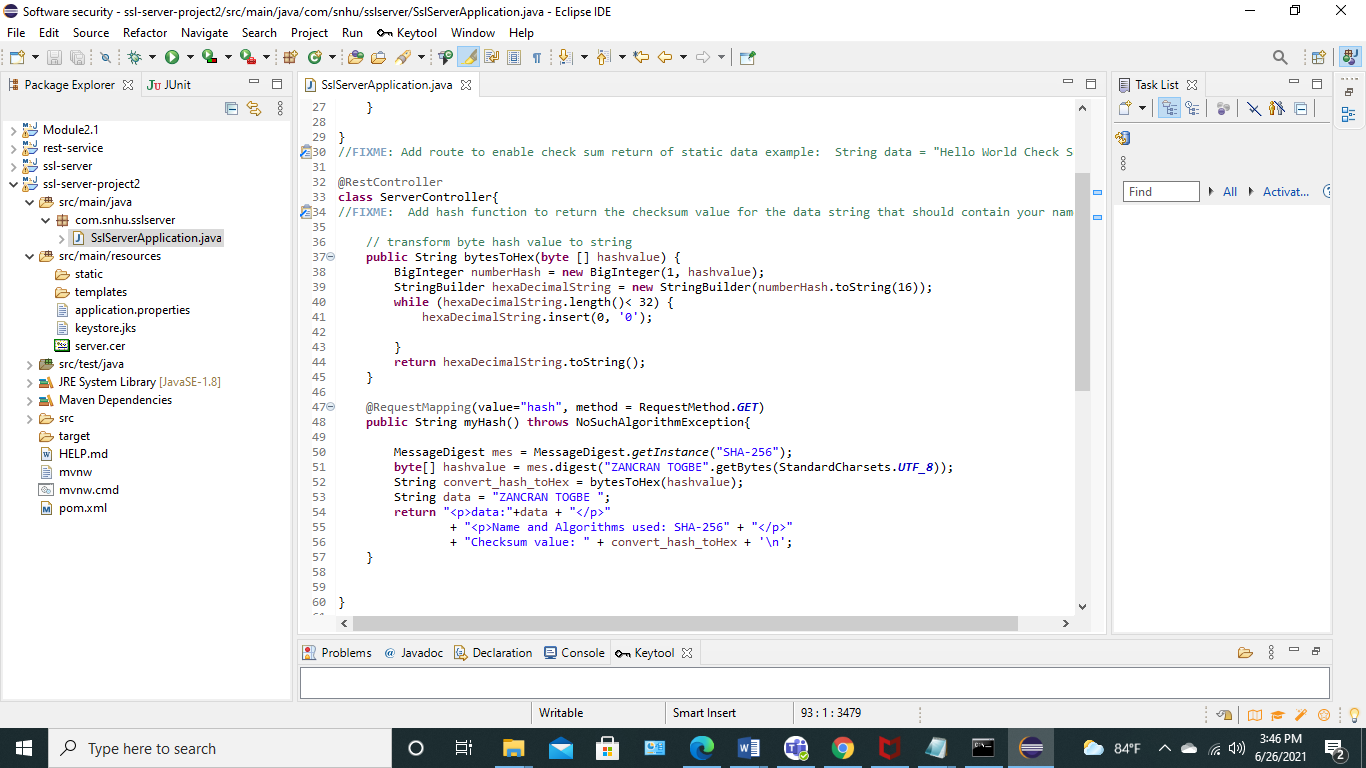
<http://localhost:8080/hash> redirected automatically to <https://localhost:8443/hash>

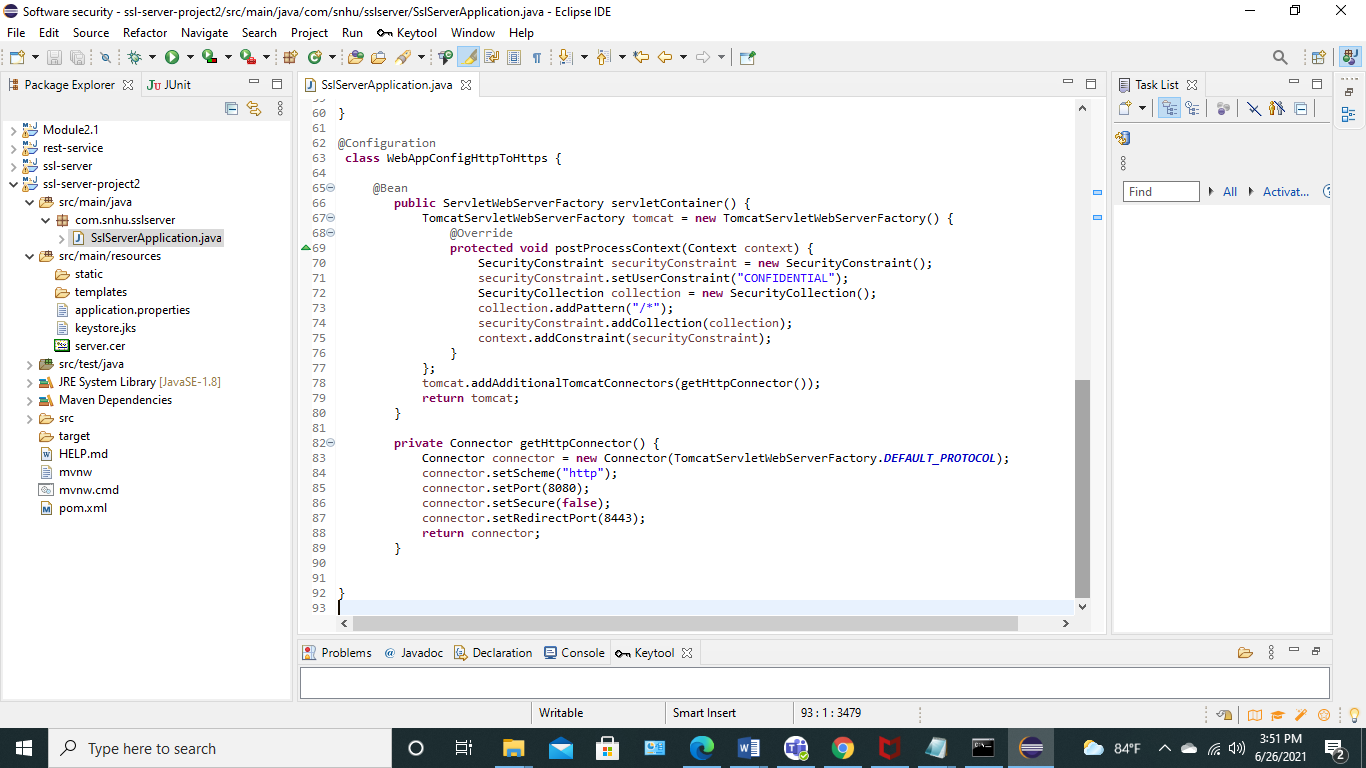


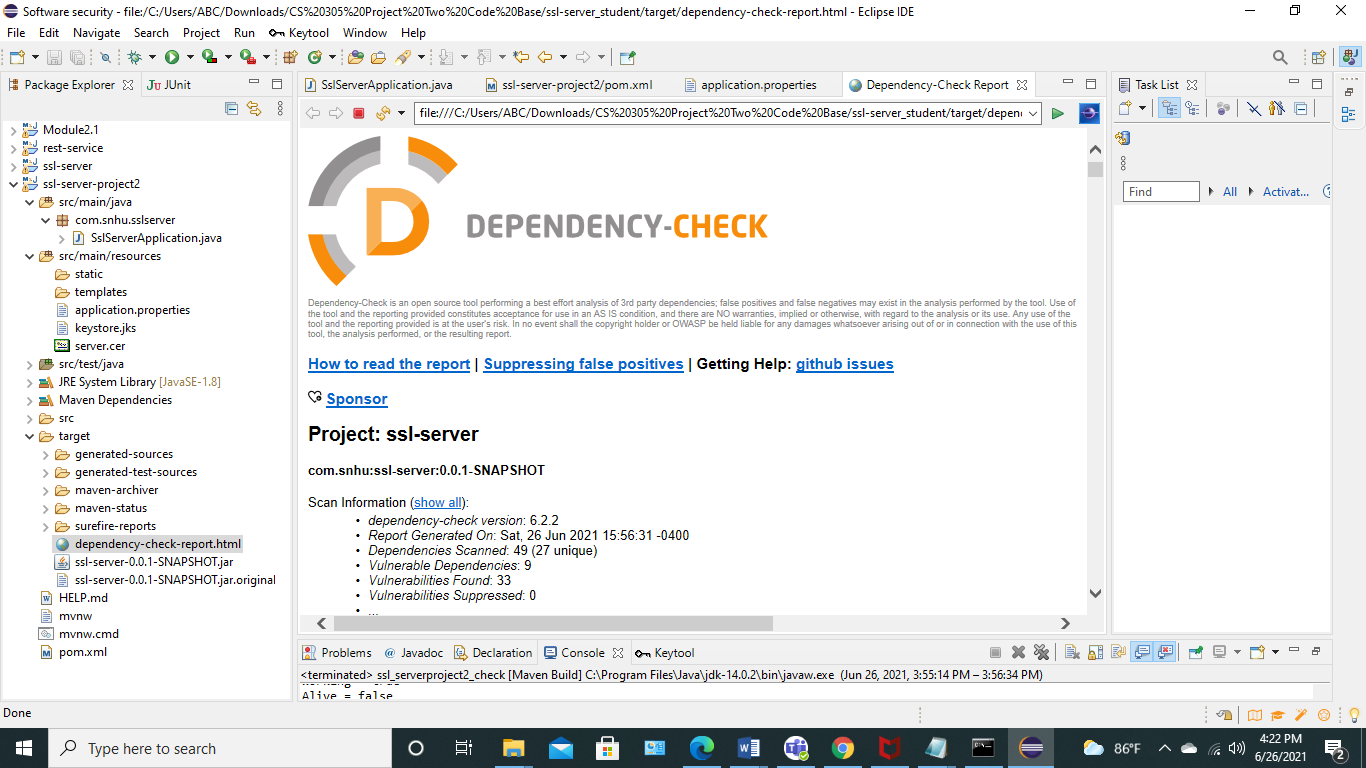
## 5. Secondary Testing

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 report



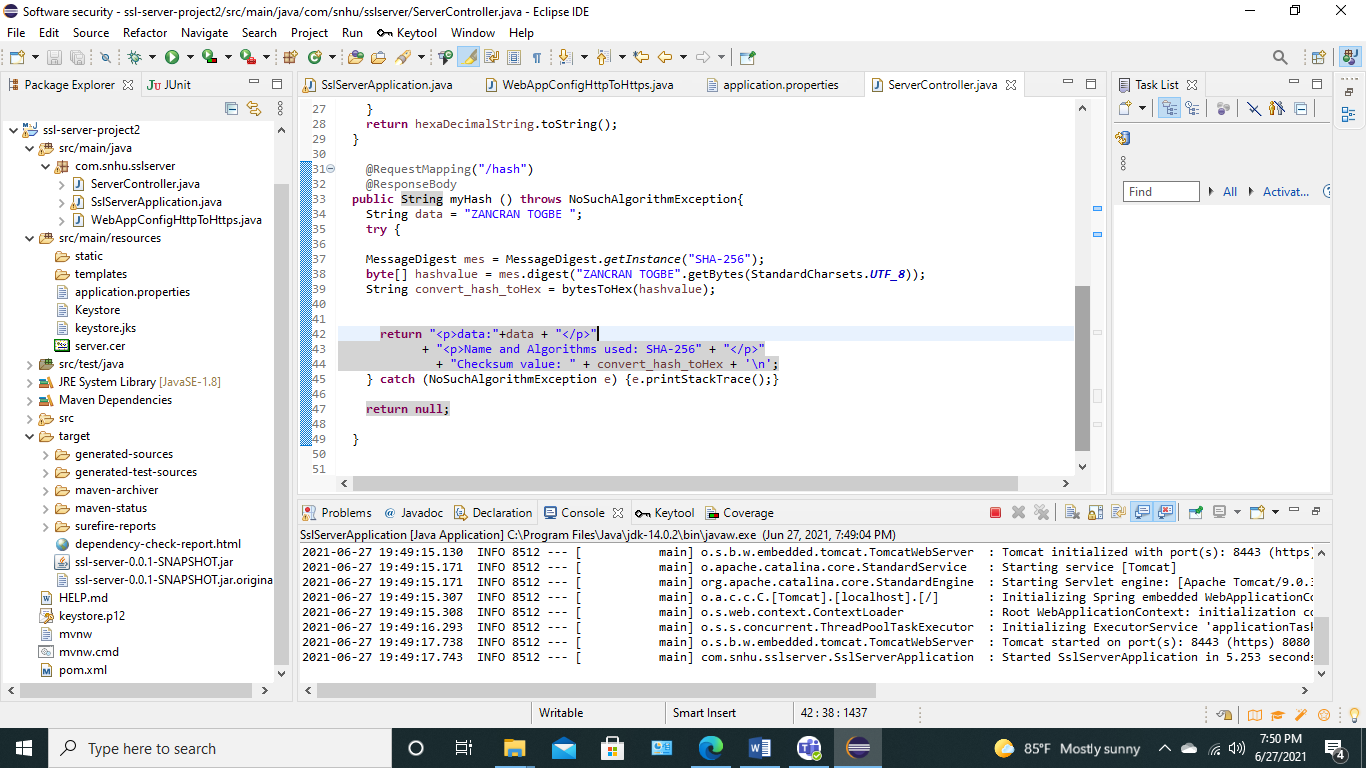




## 6. Functional Testing

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.

The areas of security addressed when refactoring the code are:

* code quality: I have used secure coding best practices by following the model view controller architecture
* code error: An exception is thrown from the function myhash(), but it is not caught, so I caught the exception
* cryptography: I ensured the integrity and the confidentiality of the messages by using SHA-256 hash function

By adding layers of security to the software application, Artemis secures customer's data at rest and in transit. The company protects its information systems against future attacks or exploits. Therefore, Artemis will offer secure products and will be more competitive in the financial market.

Maintaining the current level of security requires to:

* consider every update as potential security issues.
* test the current and the updated software periodically by applying multiple test strategies and tools ( OWASP dependency check, SAST, DAST, penetration testing, etc.)
* consider every data as untrusted
* manage authentication and session
* define access control and use certificates validation
* parametrize SQL request to prevent SQL injection and other injection attacks
* have monitoring and response or recovery plans in place in case a vulnerability is detected