****

# 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** | **04/14/2021** | **Stanislas Mbengue** |  |

## 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

Stanislas Mbengue

## 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.

An encryption algorithm converts data to an unintelligible form called ciphertext; decrypting the ciphertext converts the data back into its original form, called plaintext. For example, the AES (Advanced Encryption Standard) algorithm can use cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data.

A hash function is also use to secure data. The hash function is a unique identifier for any given piece of content. It is also a process that takes plaintext data of any size and converts it into a unique ciphertext of a specific length. A simple illustration for the hash function is that hash function takes a plaintext data input and use a mathematical algorithm to generate an unreadable output to secure data from attacker.

Random number are used in cryptography for key generator, it removes any prediction, or obscure the deterministic pattern, pad message, remove human biases from procedure. Describe the history and current state of encryption algorithms.

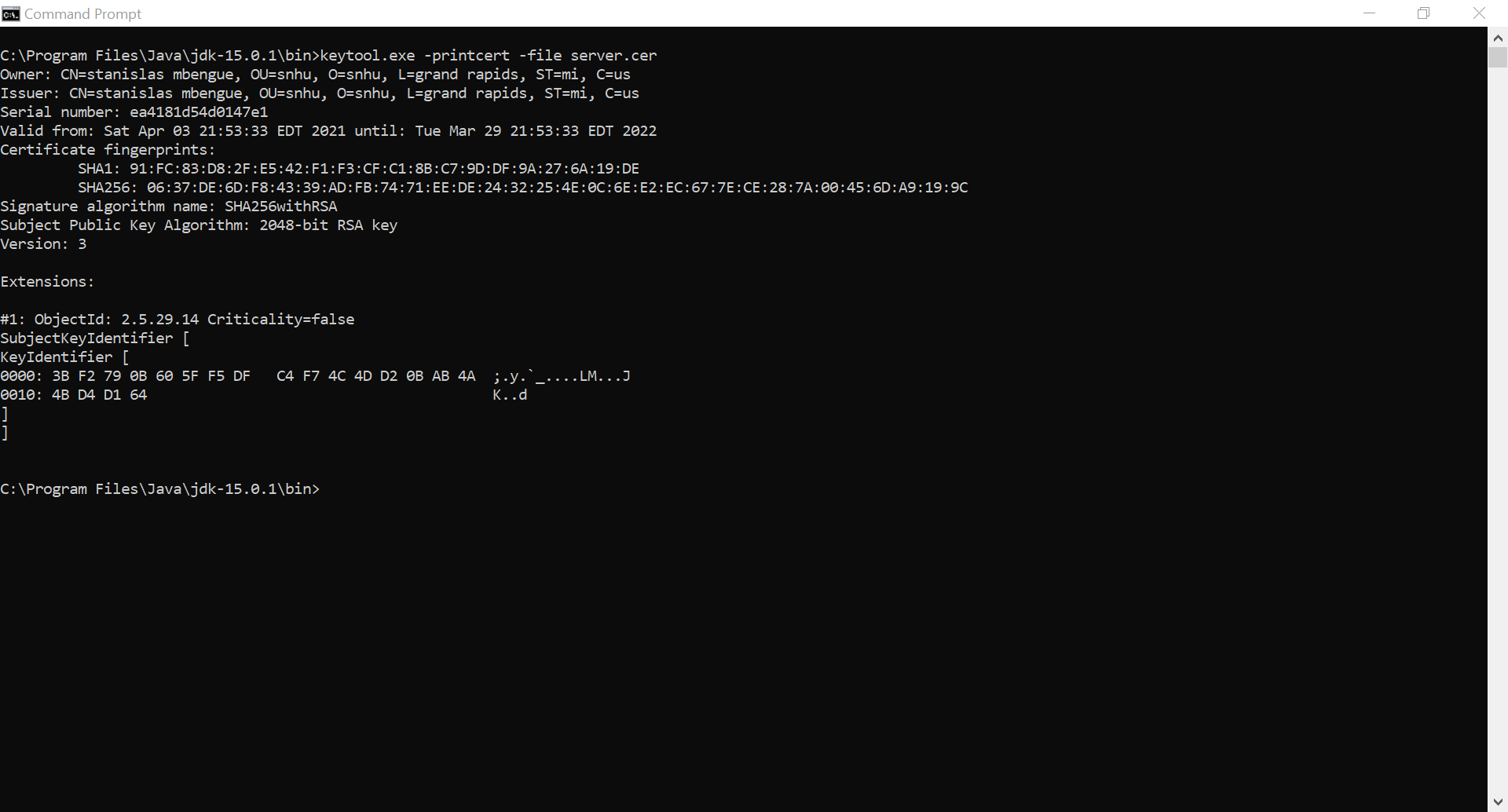
Non symmetric key or asymmetry encryption refer to having a private key and public key. In symmetric key, the same key is used to encrypt and decrypt the message.

Cryptography has been used over thousands of years in different form. For example, in the ancient Egypt, they hieroglyphs the encrypt they message. In the military, one of the oldest known ciphers is the Caesar Cipher, attributed to Julius Caesar and was used to send confidential information to his commanders and soldiers in the field. In the modern world digital communication are encrypted, to protect data and people information over the internet. SSL/TLS is used to encrypt data request, cryptography is use in everyday for authentication, digital signature, time stamping, electronic money, and encrypting and decrypting email.

## 2. Certificate Generation

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

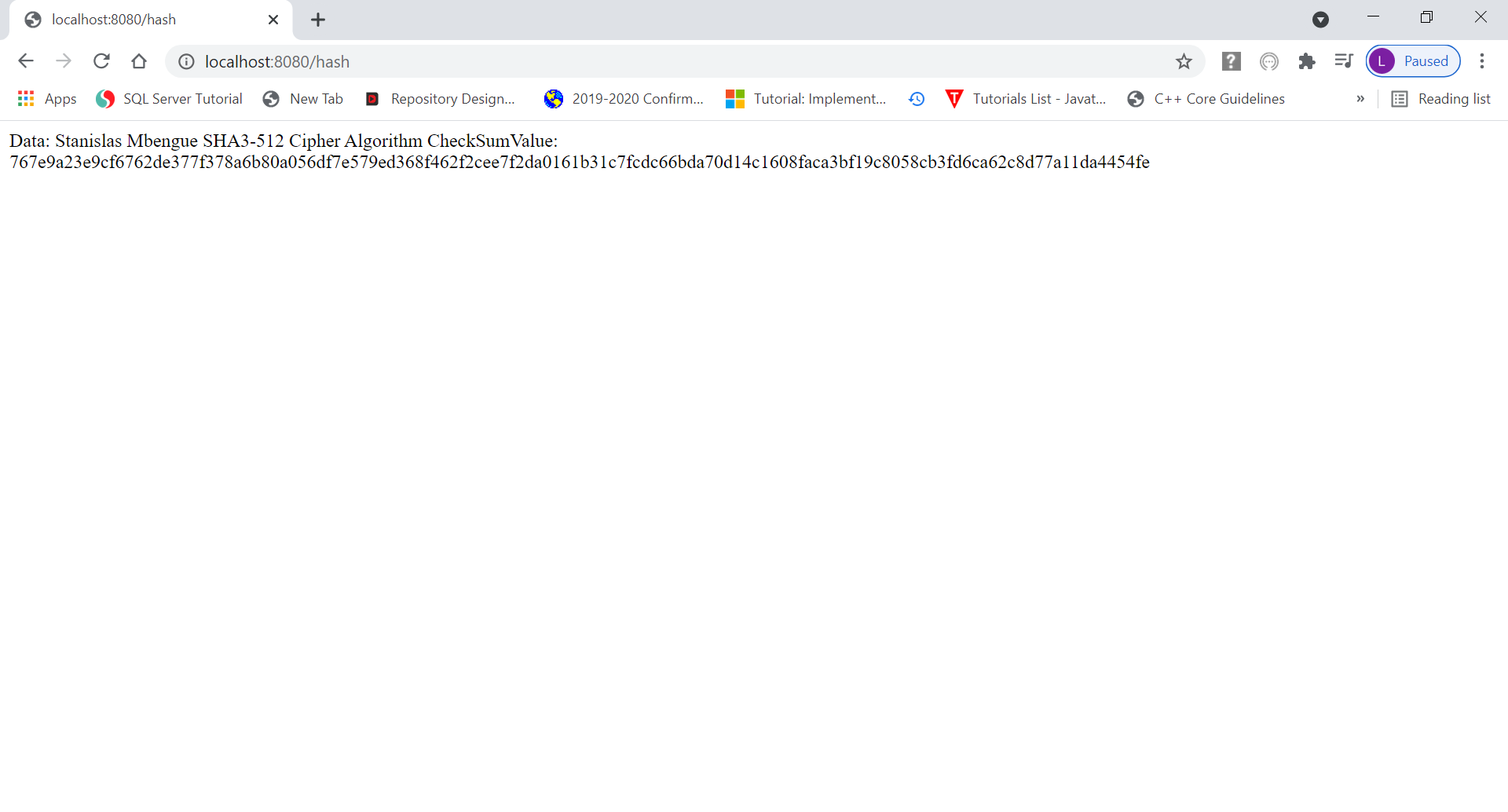
* To demonstrate that the keys were effectively generated, export your certificates (CER file) and submit a screenshot of the CER file below.

[ ]

## 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.

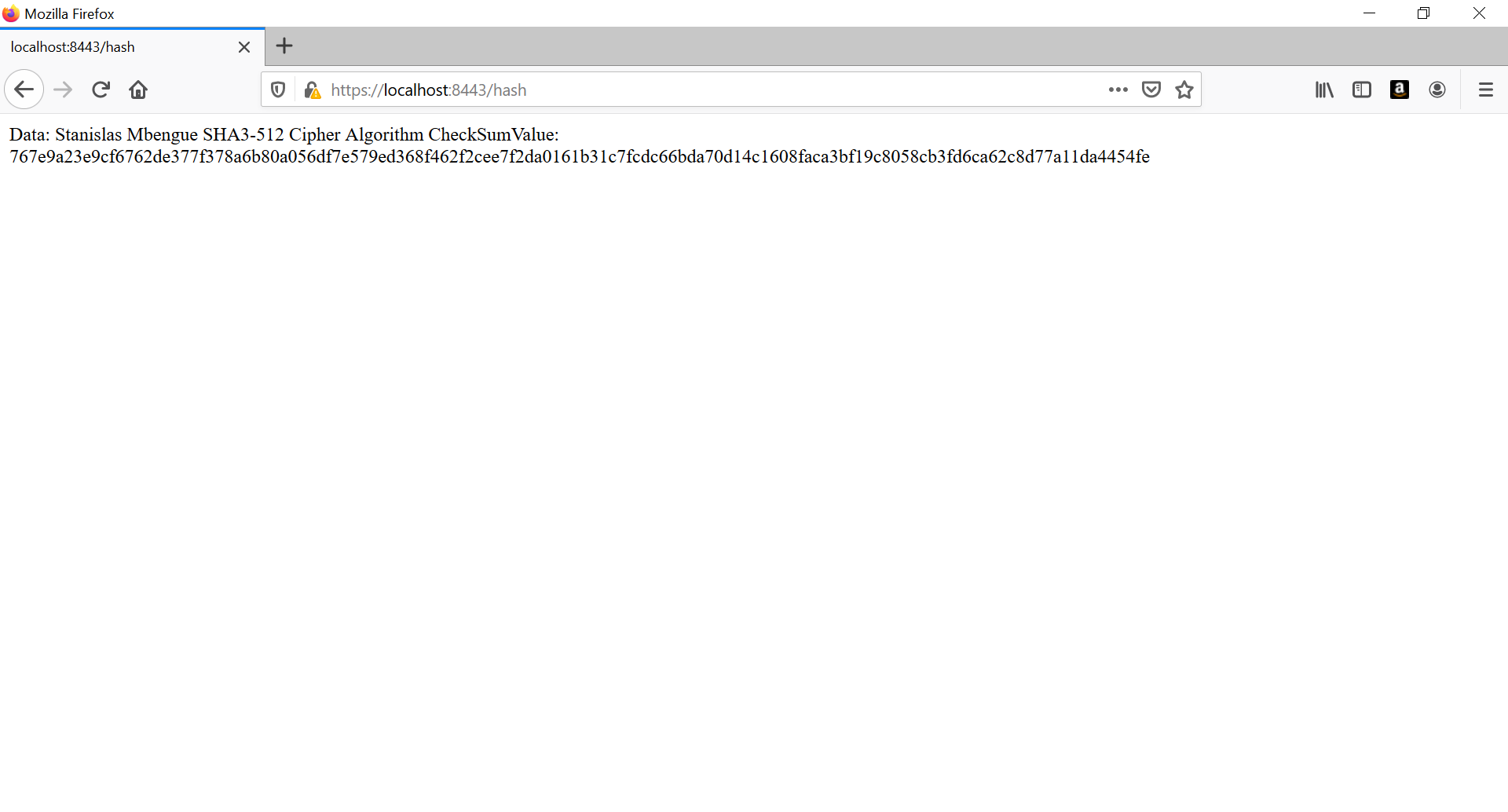
* Insert a screenshot below of the checksum verification. The screenshot must show your name and a unique data string that has been created.

[ ]

## 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.

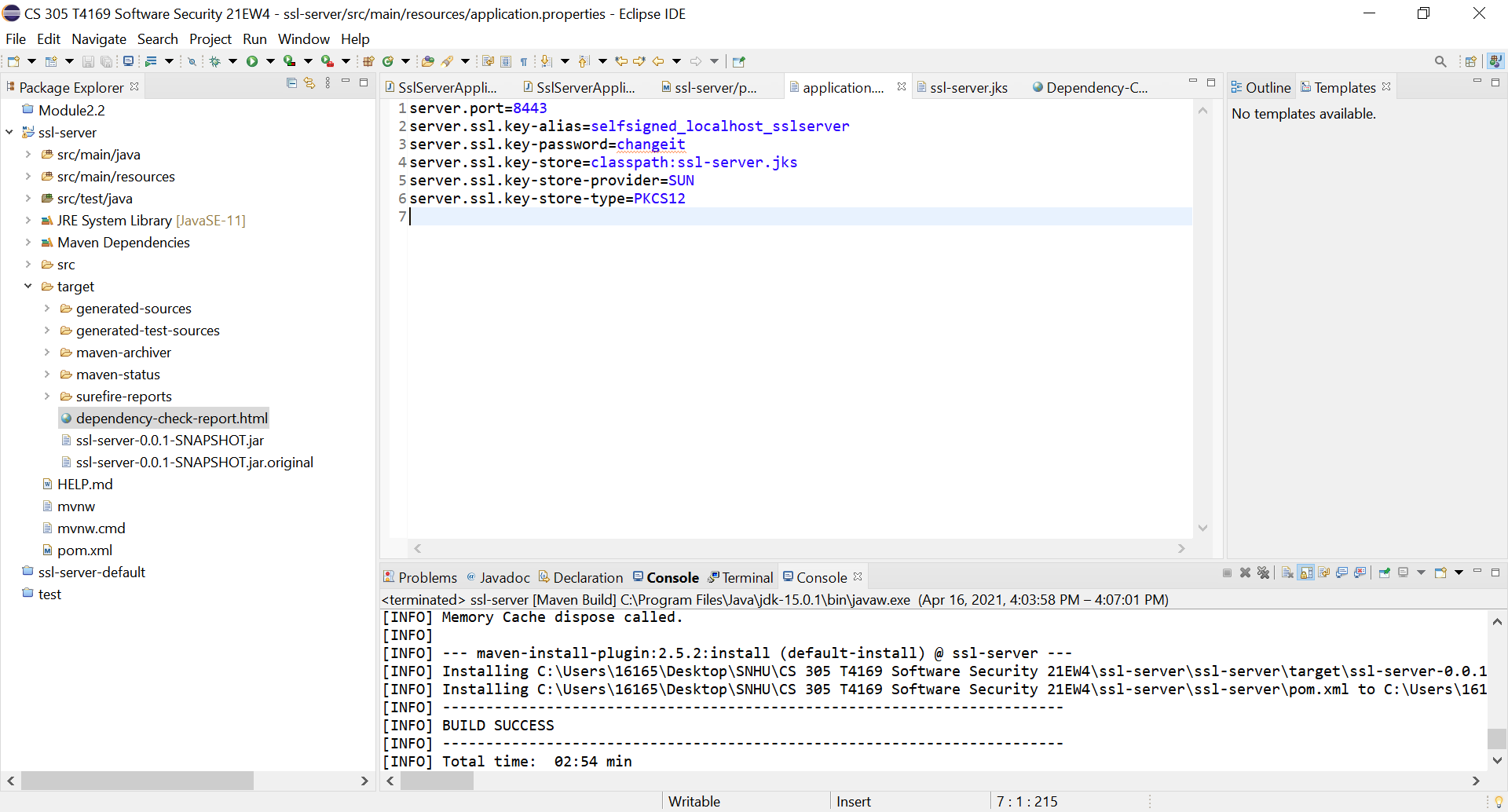
[]

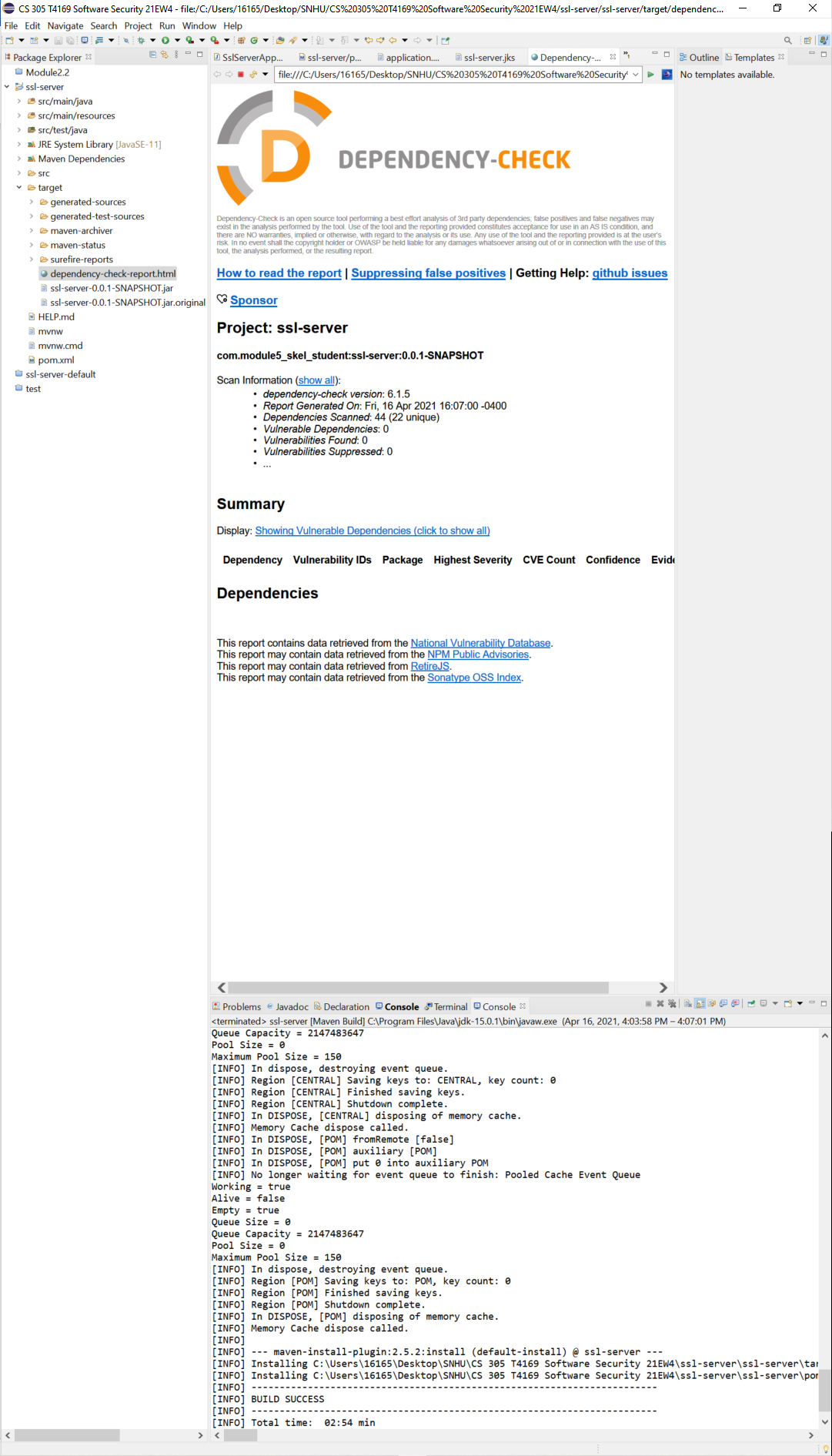
## 5. Secondary Testing

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 report

[

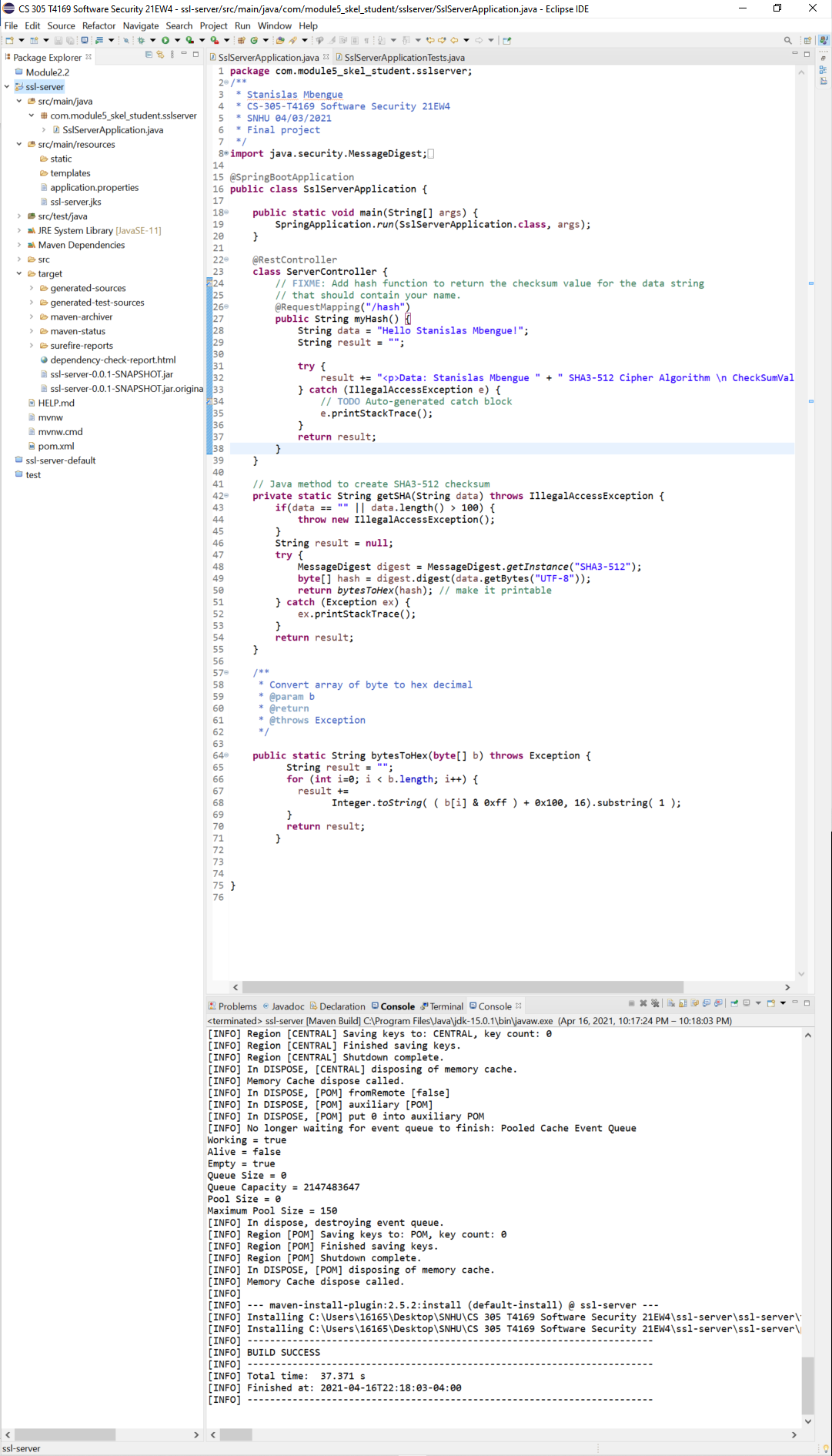


.]

## 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 refactoring code follow secure coding principle to make sure data will are checked, I did add input validation to make sure user will not input a very long string for name. I limited the number of characters to 100 which I think is reasonable for the name in this exercise and a null input will throw and exception.

Best practices for maintaining security of the software application is to make sure input are validated, follow secure coding practice, encrypt data, use authentication and authorization.