****

# 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 5](#_Toc33111307)

[3. Deploy Cipher 5](#_Toc33111308)

[4. Secure Communications 6](#_Toc33111309)

[5. Secondary Testing 7](#_Toc33111310)

[6. Functional Testing 9](#_Toc33111311)

[7. Summary 9](#_Toc33111312)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **2/20/2021** | **Ryan Stork** |  |

## 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

Ryan Stork

## 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.

The encryption algorithm cipher takes an input and encrypts it into a unique identifier. Ciphers are important because they are not able to be stolen like cleartext. These ciphers are used in encrypting data such as usernames and passwords. If a hacker were to gain access to the database of passwords, these would be unusable to them. This is done through the use of a hash function. We take the input and put it through an appropriate hash function that returns the hash. The bit level of the hash is important because these ciphers could potentially run into a collision. A collision when two different inputs result in the same output. In our program we used the SHA-256 cipher. This cipher takes an input and outputs a 256-bit hash. To date, there has been no known collisions with this function. The hash function is a type of asymmetric encryption.

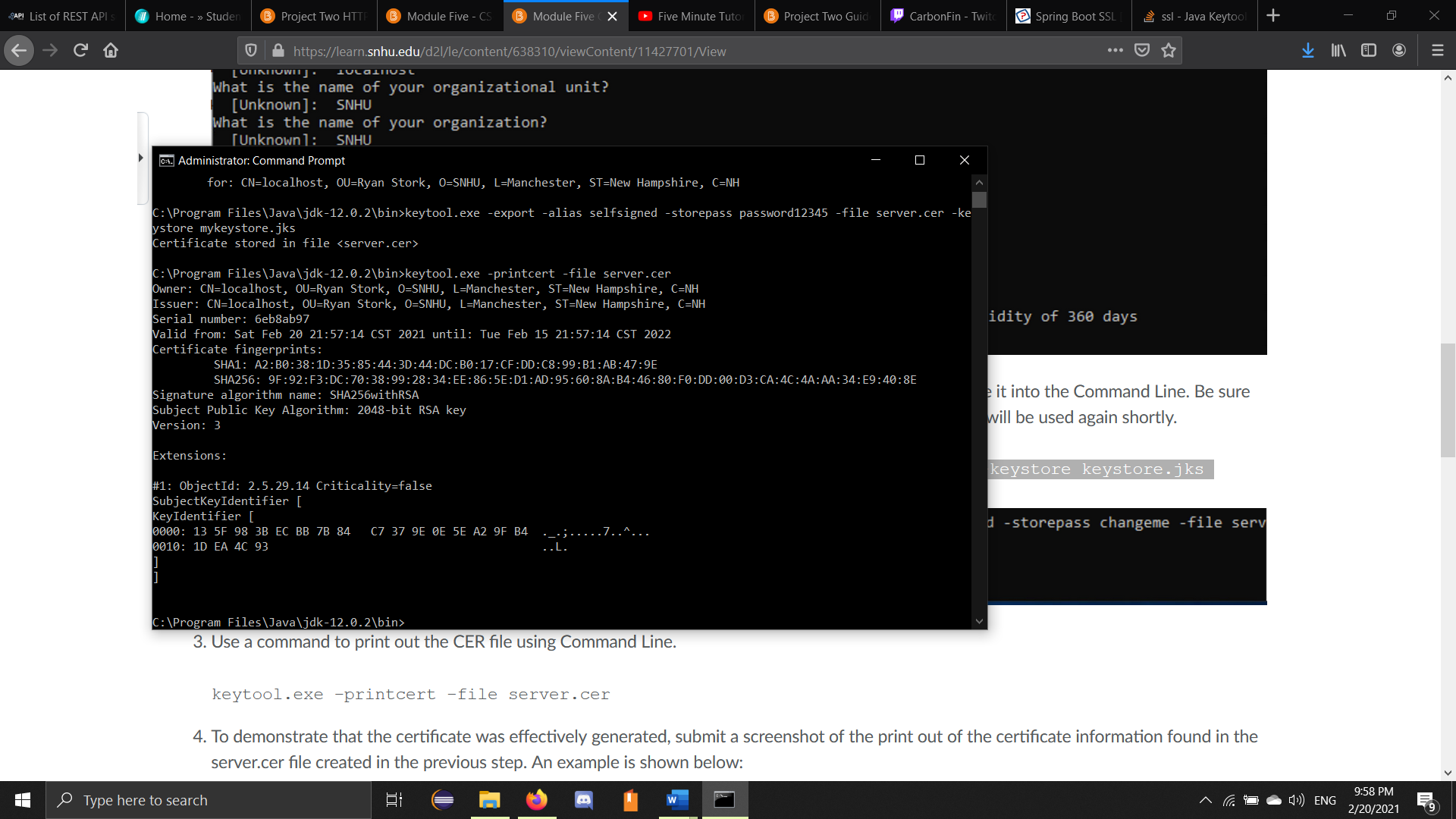
The use of random numbers comes into effect with session ID’s. Each user should have a unique session ID. These session ID’s should involve the use of a random number generator that cannot be predicted. If a session can be predicted, then a hacker could add the session ID to a website to gain access.

There are also two types of encryption used: symmetric and asymmetric. Symmetric encryption is when the same encryption cipher is used to encrypt and decrypt the data. An example of symmetric encryption is with the use of https. The data that is transmitted is secure end to end. If a hacker were to try to intercept data in the middle, they would just see unusable data. This is less secure but faster than asymmetric encryption. Asymmetric encryption uses two keys. A public key and a private key. A sender encrypts the message with their private kay and this message can be read by a receiver using the sender’s public key. Also, a site could use a public key that can only be read by a specific private key. Asymmetric encryption is important for two reasons. The first reason is that it is used in digital signatures as the private key is only for that specific user. The second reason is that a public and a private key can be used for certificate generation. Certificates are important because they show that a website is not a malicious source. Certificates are stored with Certificate Authorities to prove identity.

## 2. Certificate Generation

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

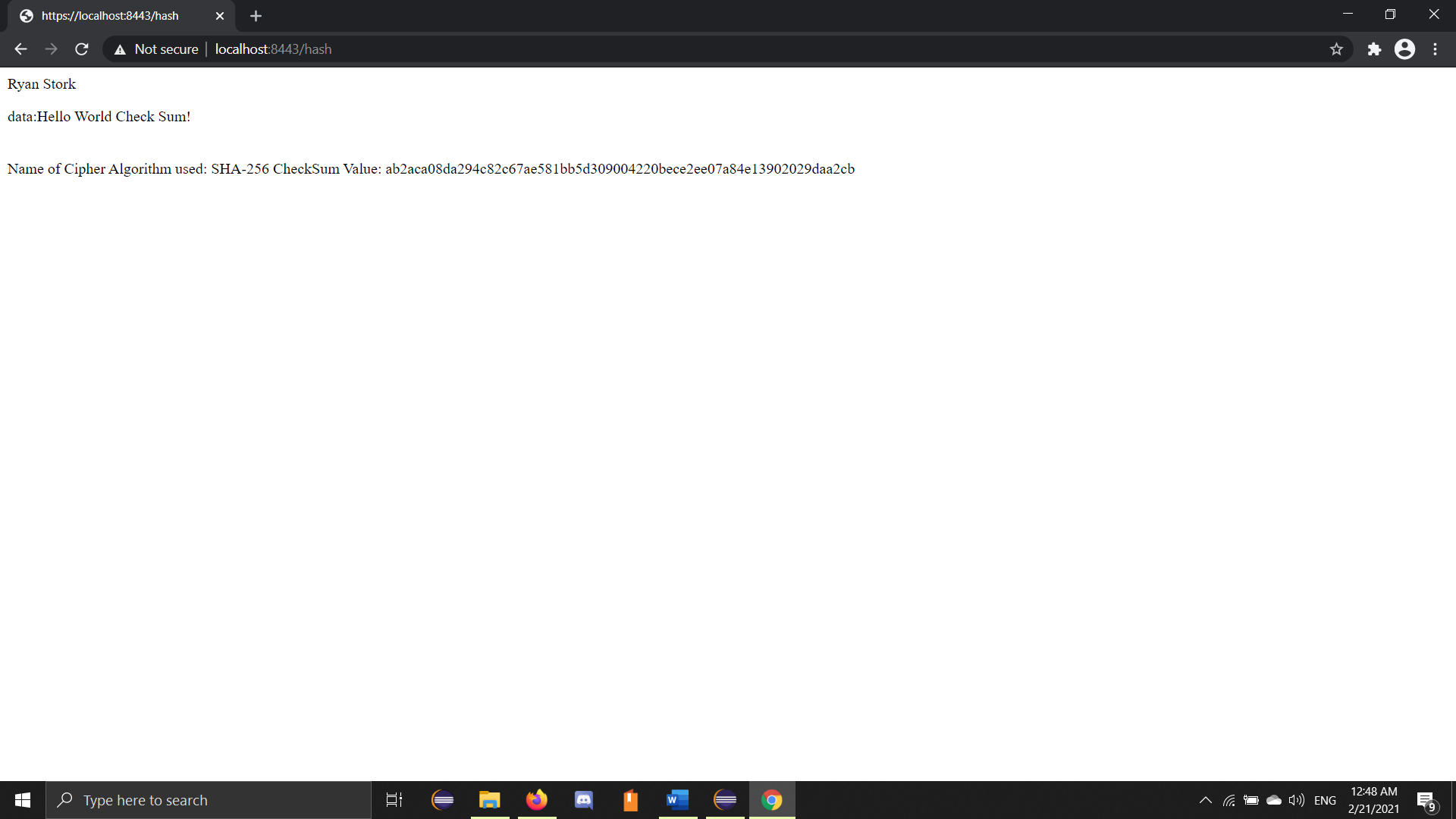
* To demonstrate that the keys were effectively generated, export your certificates (CER file) and submit a screenshot of the CER file below.



## 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

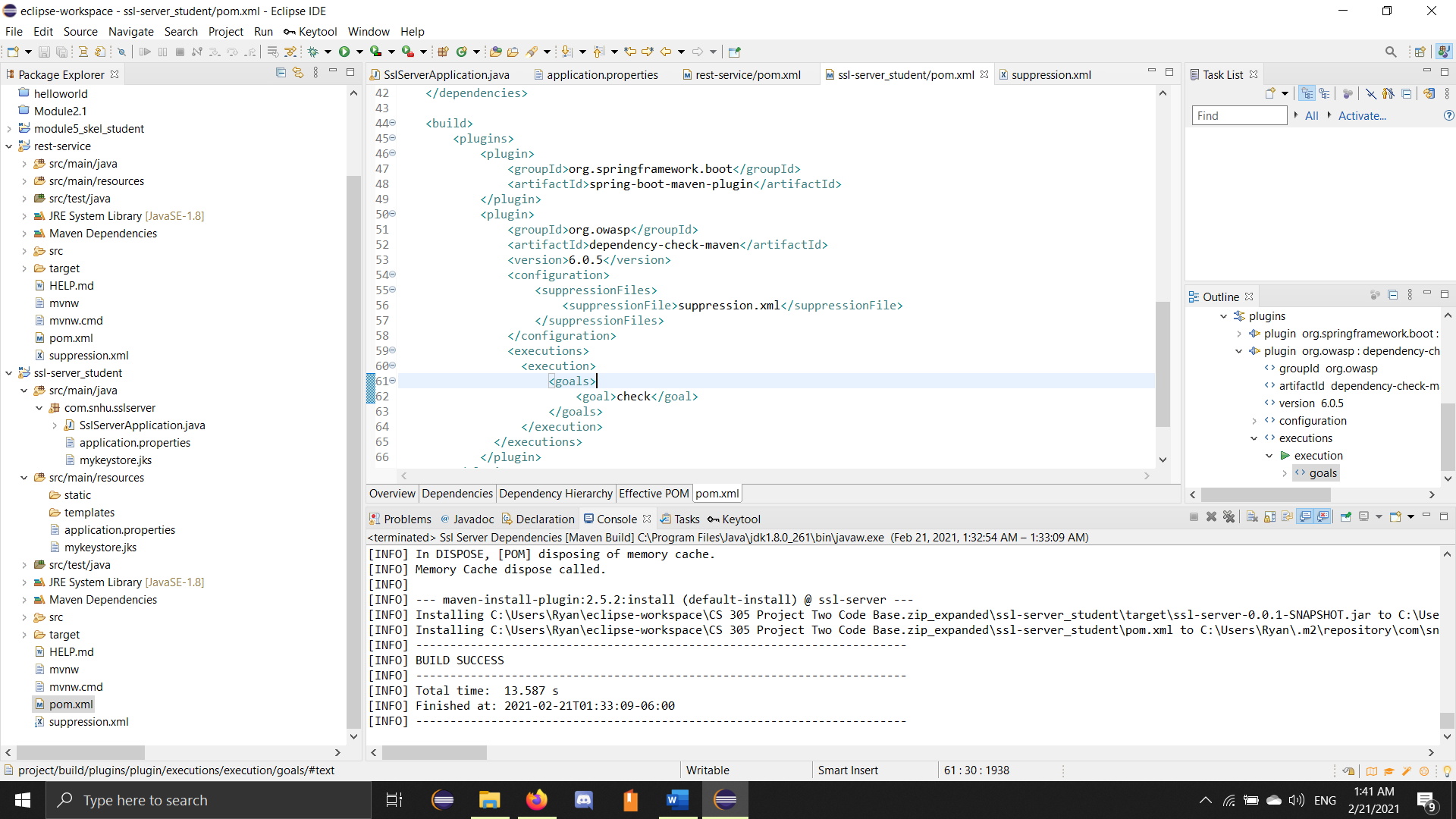
Refactor the code to convert HTTP to the HTTPS protocol. Compile and run the refactored code to verify secure communication by typing **https://localhost:8443/hash** in a new browser window to demonstrate that the secure communication works successfully.

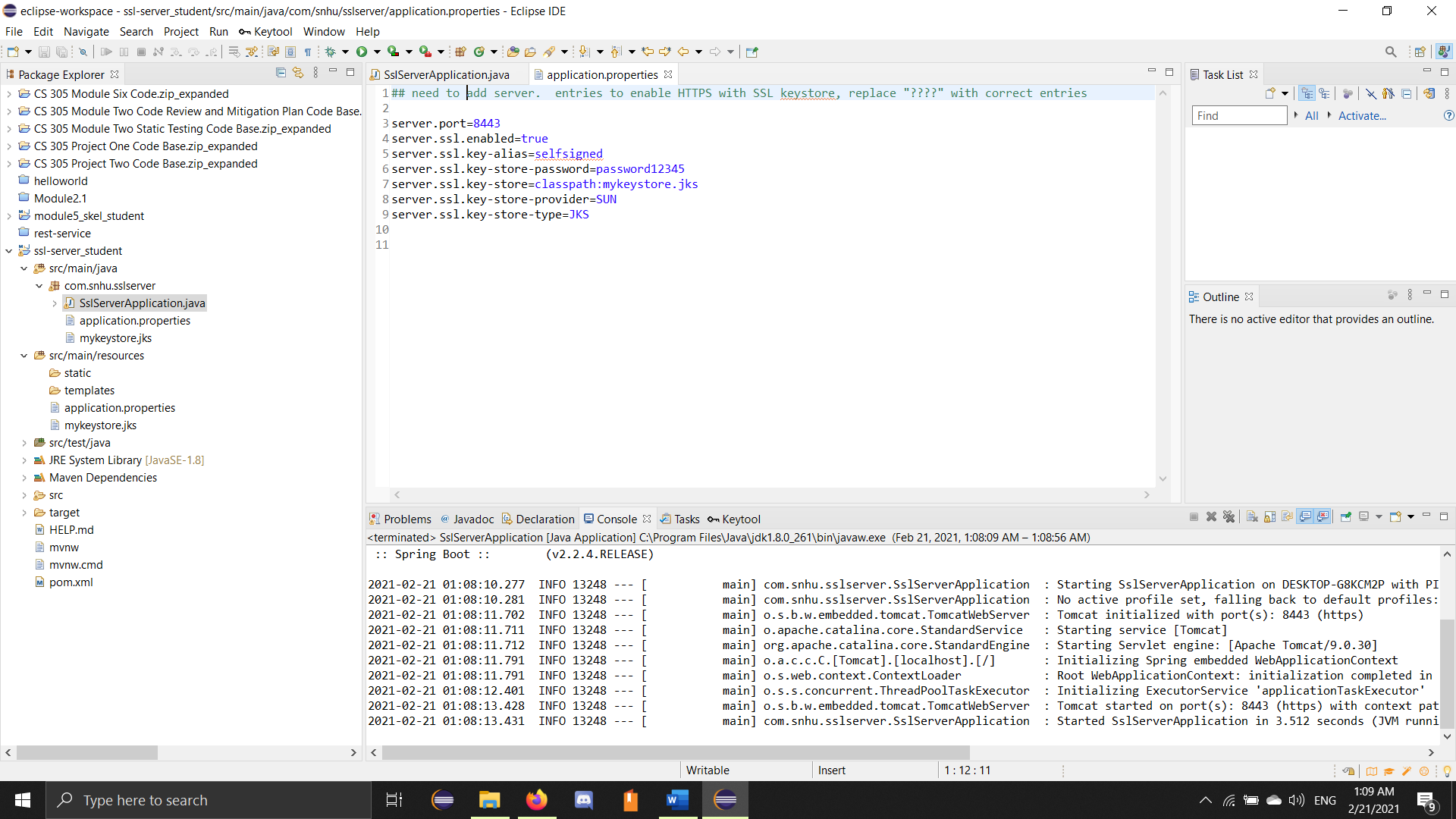
Graphical user interface, text, application, chat or text message

Description automatically generated

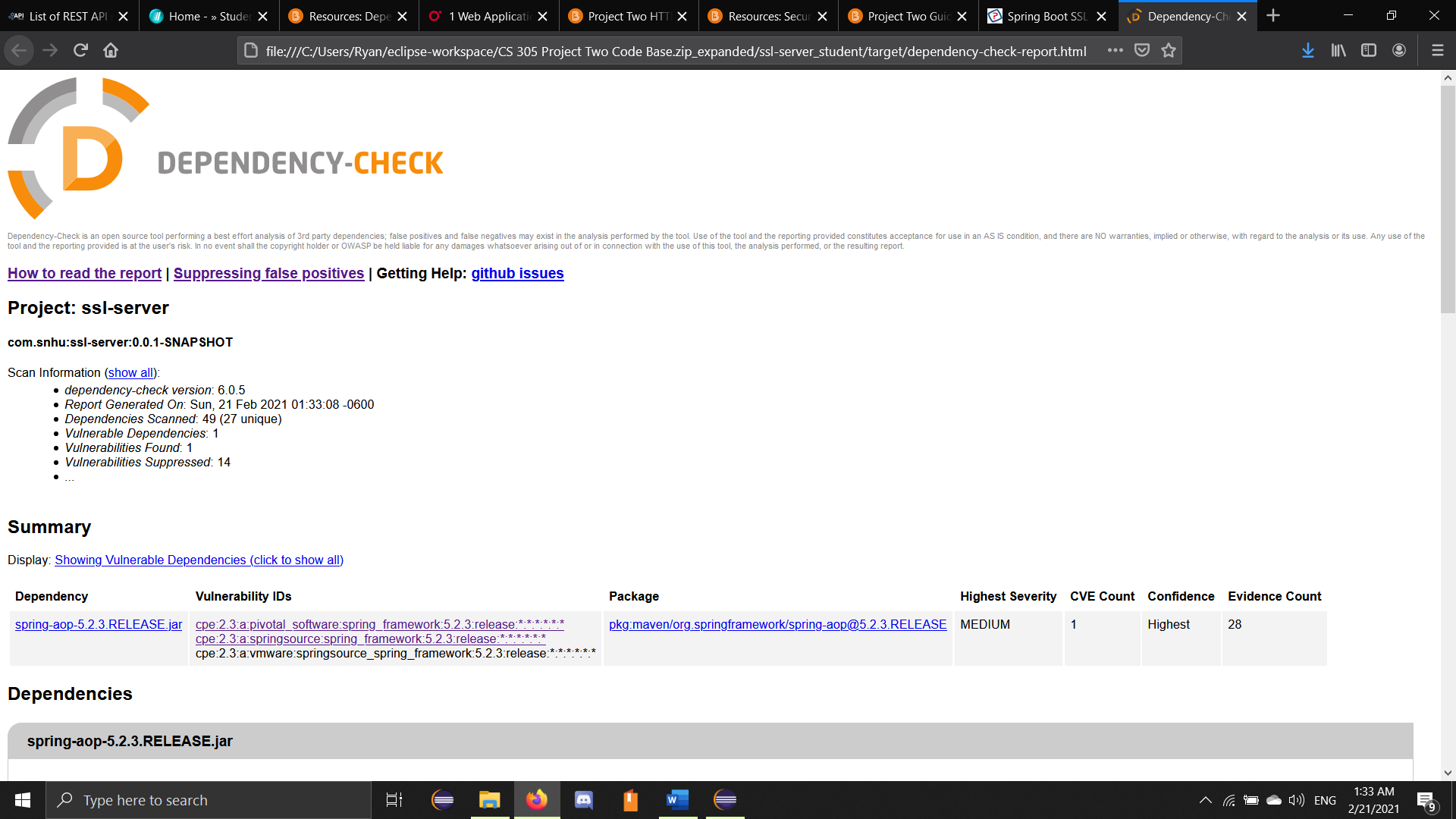
## 5. Secondary Testing

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

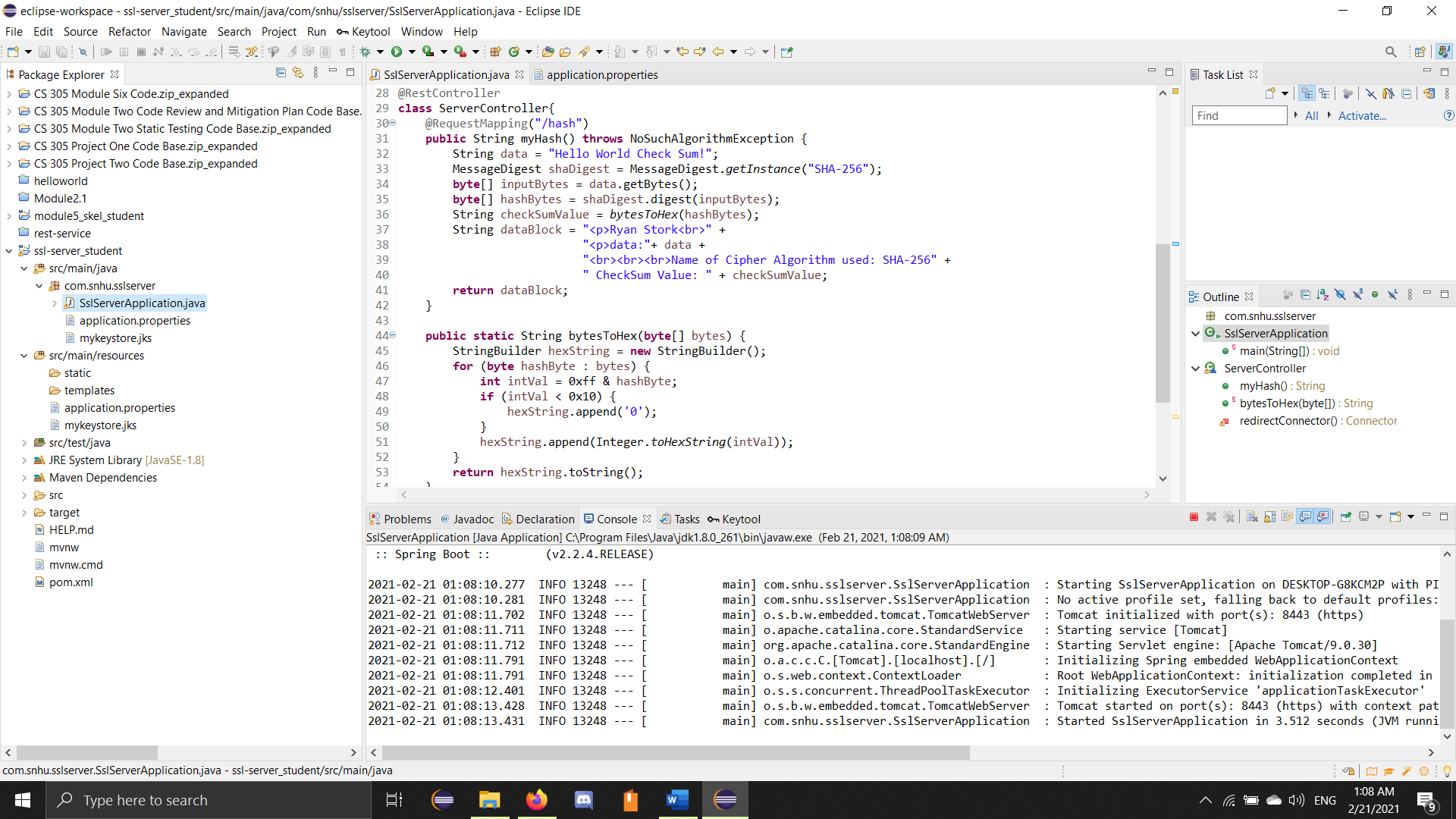


## 

## 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.

Through the Vulnerability Assessment Process Flow Diagram, there are 7 total areas of concern that you need to be aware of as a developer. In this project, we are developing a web application using the Spring Framework. We are implementing an expressive command input function. Therefore, we must be concerned with 3 areas from the Flow Diagram. Those areas are input validation, API’s, and code quality. Web applications need secure verification of input. Since our program is built in the Spring Framework, that can allow API interaction. Code Quality is a superset of both input validation and API’s.

In this project, we added SHA-256 encryption for input. This will allow us to verify identity securely before allowing access to our application. We also upgraded the port access from Port 80 which is HTTP input to port 443, which is HTTPS input. HTTPS allows for certificate verification from Certificate Authorities. These authorities keep track of the owners of each site or application. They can ensure if a site can be trusted. HTTPS also protects against man-in-the-middle attacks, as the data is encrypted from end to end where an interceptor cannot read the data.

The best practices for maintaining the current security of the software application are to keep customer’s data secure. Hackers are always trying to find new ways to exploit systems to gain unauthorized access. By maintaining the current security, we are patching flaws that have been discovered by others and making sure that data is secure. We must keep confidentiality, integrity and availability. Confidentiality to take measures to protect data from unauthorized access and misuse. Integrity to protect that data from unauthorized alteration. And availability of data at all times to make sure that customers are not burdened by an attack that may have taken the system offline.