****

# Practices for Secure Software Report

Table of Contents

[Document Revision History 3](#_Toc102040754)

[Client 3](#_Toc102040755)

[Instructions 3](#_Toc102040756)

[Developer 4](#_Toc102040757)

[1. Algorithm Cipher 4](#_Toc102040758)

[2. Certificate Generation 4](#_Toc102040759)

[3. Deploy Cipher 4](#_Toc102040760)

[4. Secure Communications 4](#_Toc102040761)

[5. Secondary Testing 4](#_Toc102040762)

[6. Functional Testing 4](#_Toc102040763)

[7. Summary 4](#_Toc102040764)

[8. Industry Standard Best Practices 4](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **10/16/22** | **Dorcas Tabitha Sinde** |  |

## 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

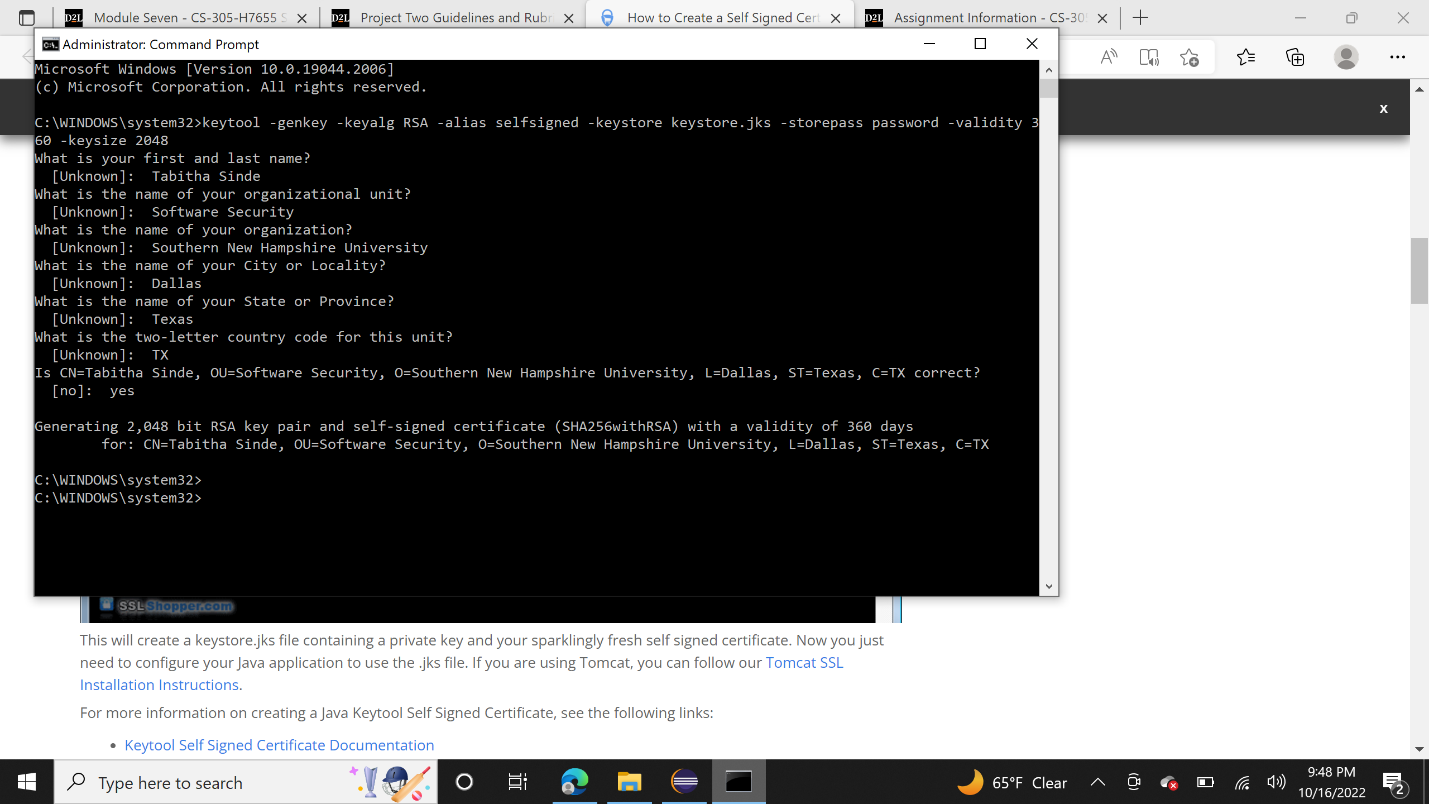
[Dorcas Tabitha Sinde]

## Algorithm Cipher

[The best encryption algorithm cipher I would employ is the AES cipher since it is generally resilient against hacking attempts and supports a variety of key sizes and is among one with the highest standards in the existing technology world. Its symmetric keys creation will allow my application to encrypt data as needed and deliver keys to its clients. AES will allow you to choose between a 128-bit or 256-bit key making it stronger than the 56-bit key of DES. The biggest disadvantage of AES though is associated with symmetric key encryption that needs one to have a way to get the key to the party with whom I am sharing data.]

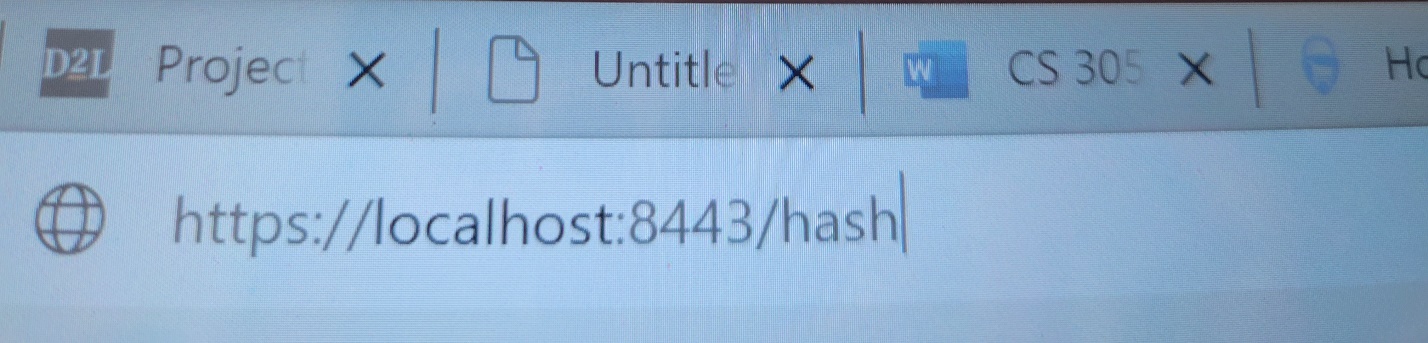
## Certificate Generation

Insert a screenshot below of the CER file.



## Deploy Cipher

Insert a screenshot below of the checksum verification.



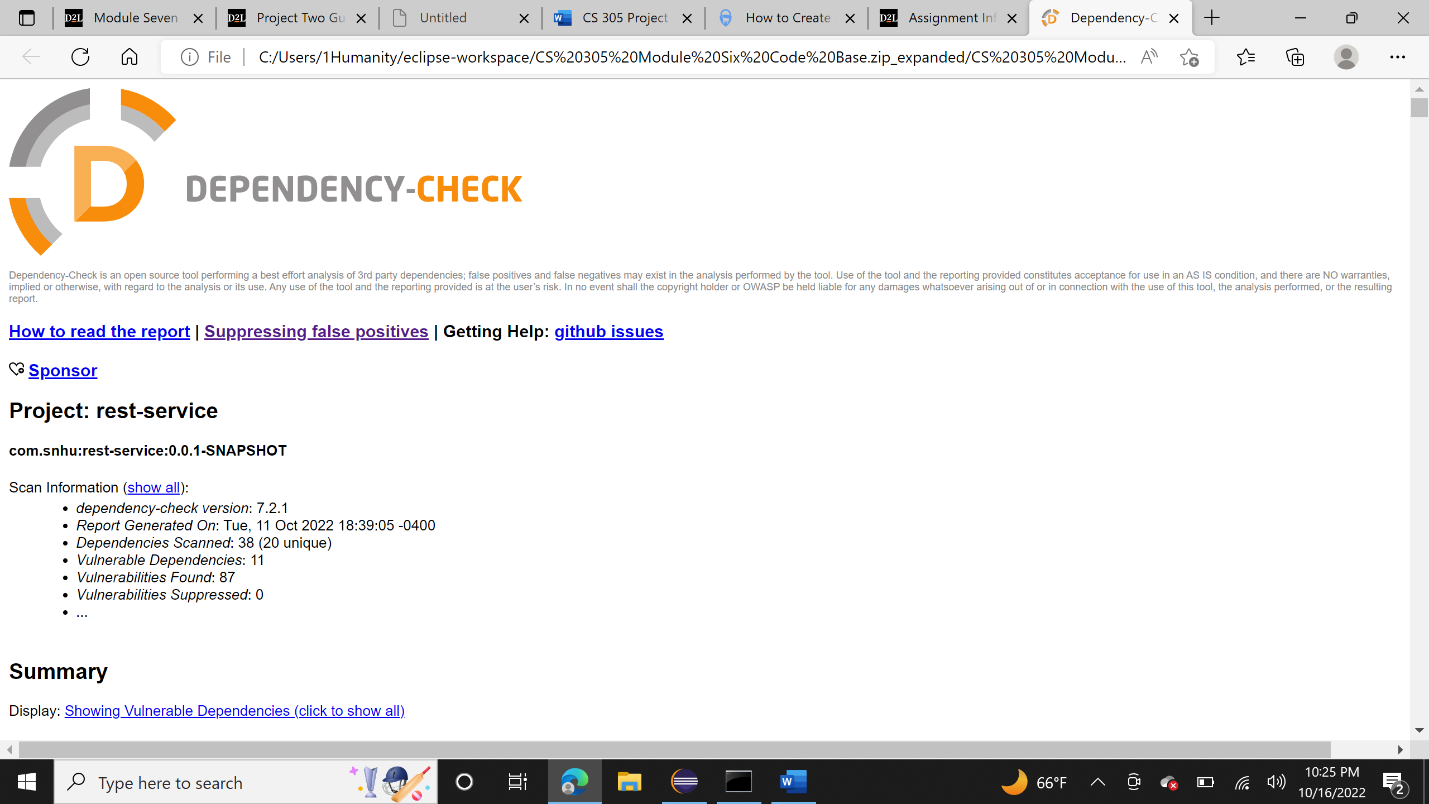
## Secure Communications

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



## Secondary Testing

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



## Functional Testing

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

Graphical user interface, text, application

Description automatically generated

## Summary

Creating my self-signed certificate at the beginning of the project and generating keys for the usage in this application, was a convenient way to connect me to the 128-bit AES encryption, which allows the intended recipients to read the data and provide security to the applications communications. Paying attention to the security of the application will help protect the company assets and maintain the client’s trust. By refactoring the code, I have been able to address important areas of security as APIs, cryptography, client server and code quality.

## Industry Standard Best Practices

The best Industry Standard practices is to frequently check the code for vulnerabilities especially after the implementation of new functions or introduction of any changes, and if any vulnerabilities are found during development the team should employ methods to get rid off the vulnerabilities as early as possible. Thereby maintaining security should be the highest priority for any company.