****

# 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** | **2/25/2023** | **Brianna Marsh** |  |

## 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

## Brianna Nicole Marsh

## Algorithm Cipher

After Integrating the Maven Dependency Check Plug-in Test, I was able to see the current vulnerabilities. The major vulnerabilities that I have observed were that their system was outdated, easily to be hijacked, the lack of entity expansion security, and most of all, the risk of integrity. The appropriate encryption algorithm cipher that I would deploy is AES. Another option would be a more in-depth option of Caesar. With the code scrambled with random numbers, the receiver would be the only one who had the proper key to open it!

## Certificate Generation

Insert a screenshot below of the CER file.

Text

Description automatically generated

## Deploy Cipher

Insert a screenshot below of the checksum verification.

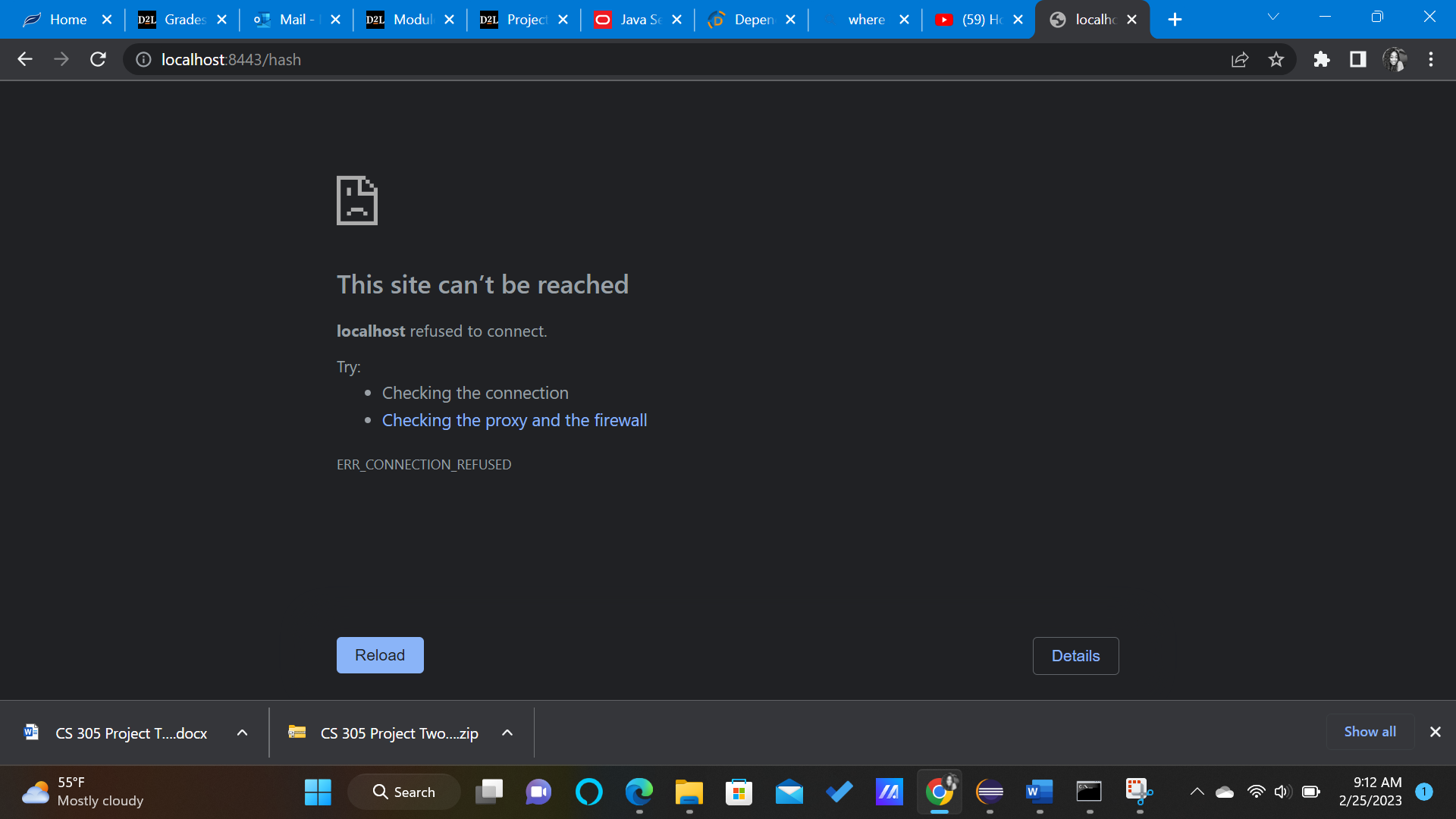
A screenshot of a computer

Description automatically generated with medium confidence

## Secure Communications

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

My page would not pull up for some odd reason.



## Secondary Testing

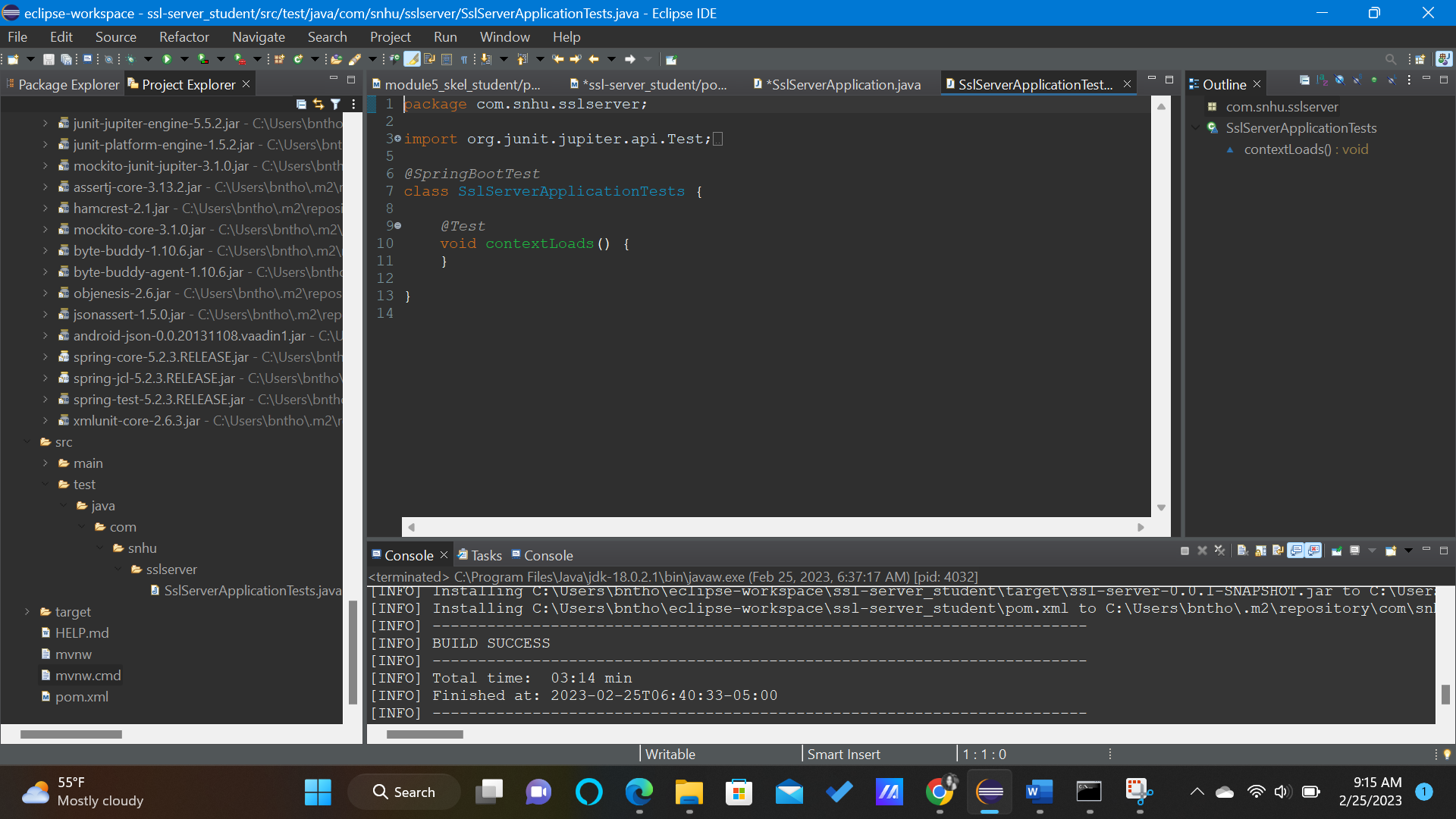
Insert screenshots below of the refactored code executed without errors and the dependency-check report.

A screenshot of a computer

Description automatically generated with medium confidence

Graphical user interface, text

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

## Functional Testing

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

## Summary

I addressed the vulnerabilities prior to fixing the code.

I was able to fix the *“#FIX ME”* section and put the correct information in the code as well as follow the directions that were given to me in the document.

## Industry Standard Best Practices

I applied industry best practices for secure coding by making sure that my indentions and spaces were in perfect alignment with the rest of the code. It holds value to use industry best practices for secure coding because it keeps your code neat. This is also important because the next programmer will be able to neatly read your code while being able to understand it. The neater your code is, the fewer you will find mistakes, because you are able to see the mistakes more so when the code is neater!