****

# Practices for Secure Software Report

**Table of Contents**

[**Document Revision History 3**](#_heading=h.30j0zll)

[**Client 3**](#_heading=h.1fob9te)

[**Instructions 3**](#_heading=h.3znysh7)

[**Developer 4**](#_heading=h.2et92p0)

[**1. Algorithm Cipher 4**](#_heading=h.tyjcwt)

[**2. Certificate Generation 4**](#_heading=h.3dy6vkm)

[**3. Deploy Cipher 4**](#_heading=h.1t3h5sf)

[**4. Secure Communications 4**](#_heading=h.4d34og8)

[**5. Secondary Testing 4**](#_heading=h.2s8eyo1)

[**6. Functional Testing 4**](#_heading=h.17dp8vu)

[**7. Summary 4**](#_heading=h.3rdcrjn)

[**8. Industry Standard Best Practices 4**](#_heading=h.26in1rg)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **12/22/2024** | **Kursheeka Milburn** |  |

## 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

[Kursheeka Milburn]

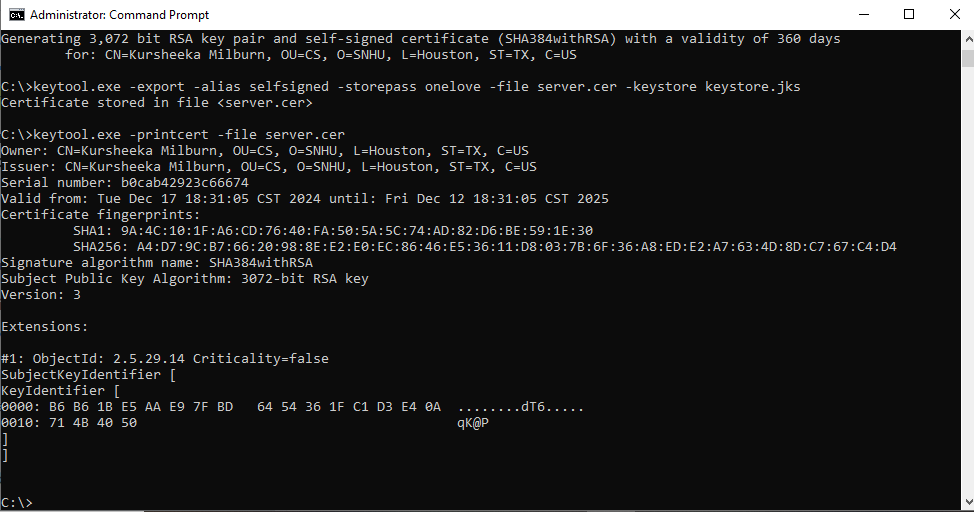
## Algorithm Cipher

[Artemis Financial’s main goal is to help people all over the world with their money by offering awesome financial programs. To keep everything safe, I think using SHA-256 as the encryption algorithm is the best choice. This encryption is strong enough to where no one can break it easily. This is why banks and/or other financial companies usually use SHA-256 to protect their things.SHA-256 takes information, squishes it into a hash value, and makes it almost impossible for hackers to figure out what it was originally. Also it’s randomness of encryption, making things difficult and unpredictable to figure out. 256-bit encryption is amazing because of all the combinations it is able to create making it way harder for anyone to break in and steal info.

There are also two main ways to do encryption: symmetric keys and asymmetric keys. Symmetric keys are simple and fast because they use one key for both locking and unlocking information. AES-256 is one that encrypts plain text, but asymmetric keys use two keys; one to lock and one to unlock. That’s why they’re often used for sending stuff online due to the extra security.Encryption isn’t something new. From a brief reading I read that since around 600 BC people have been trying to keep secrets via encryption algorithms. Back then, they used basic codes, but now we have intelligent and more advanced technology to protect data from hackers. Over time, encryption has gotten stronger and smarter because of the security aspect. Encryption is used to keep passwords and money safe. It helps a lot with keeping hackers from the ability to gain access to any information that is supposed to be kept a secret.]

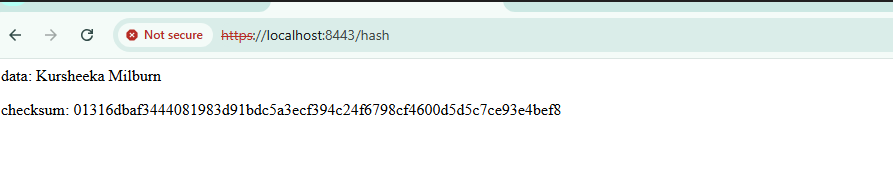
## Certificate Generation

Insert a screenshot below of the CER file.



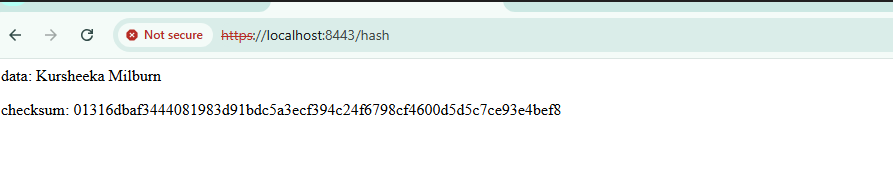
## Deploy Cipher

Insert a screenshot below of the checksum verification.



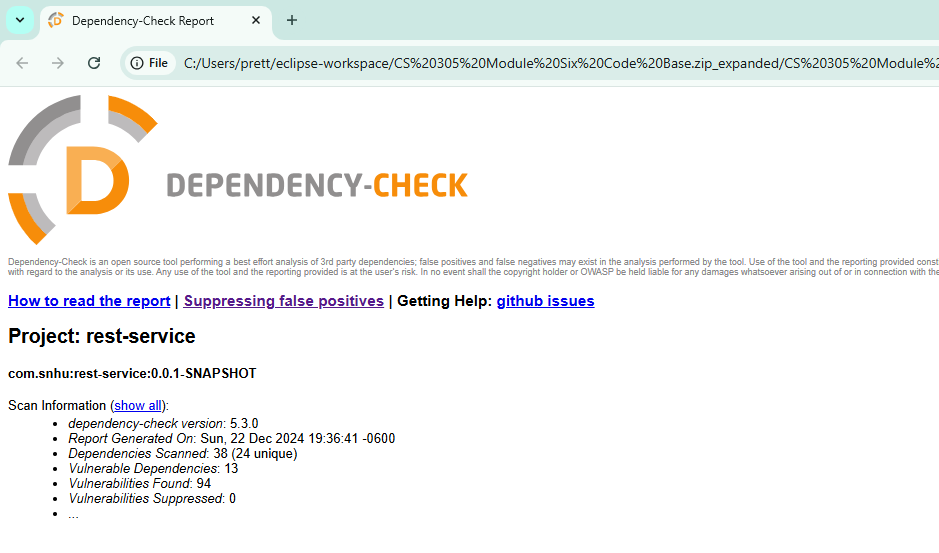
## Secure Communications

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



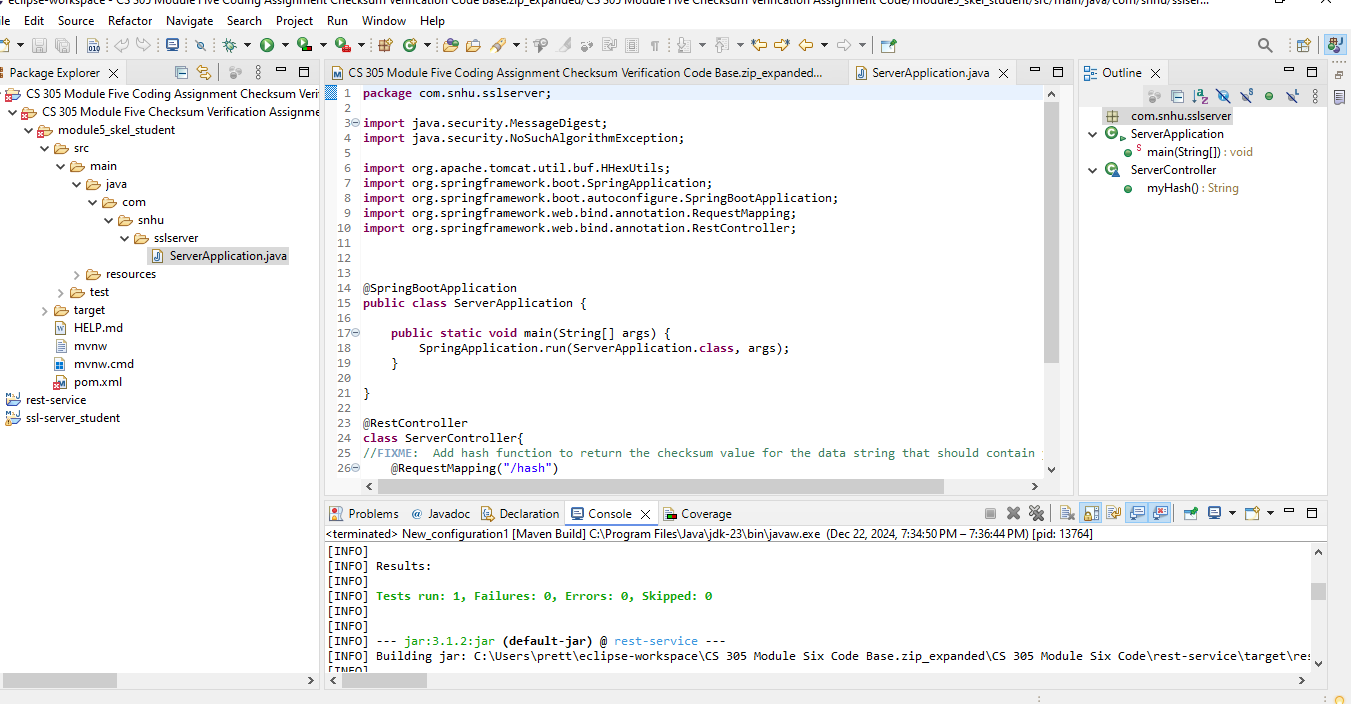
## Secondary Testing

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.



## Summary

[I was able to make the app more secure by using self-signed certificates to enable HTTPS. This was an important step because HTTPS makes sure that people's connection is safe, and they are using the real site and not a fake one. To do this, the first thing was to create the certificates. Doing this made the website more secure and helped build trust with the users. Afterwards, I worked on fixing issues in the pom.xml file. This made sure that any problems found during the dependency check were solved. I also checked the hashing function to make sure it was working properly by using a checksum. Helping scramble customer data so that it’s hard for hackers to steal. This keeps everyone’s info safe and gives us peace of mind that the business is doing the right thing to protect our users. Lastly, was to make sure all vulnerabilities were patched. Just making sure everything is updated in the app so it is protected. Doing this ensures everything is running the way it is supposed to and makes sure there are no weak spots that hackers could use. Also enforcing least privilege ensures that everyone in the company only has access to what they really need and nothing more. This helps protect the company from attacks that could come from the inside. Even though we can’t fully do this yet with our current program, it’s still something we should aim for. By taking these steps we’re making sure our app is safe and ready for anything that may transpire.]

## Industry Standard Best Practices

[I followed industry standard best practices by making sure the app used HTTPS for secure connections and fixed dependency issues to prevent weak spots. I also used proper hashing methods to protect customer data and kept everything up-to-date with patches.Using these best practices is good for the whole company when we keep our software secure, customers trust us more, and that trust helps the business grow. It also saves the company from data breaches or hackers, which could ruin the company’s reputation. We’re making sure the company stays safe, successful, and protected.]