

7-1 Project Two Submission

Yanika Francis

SNHU CS – 255 Prof. Victorio



G L O B A L R A I N

Practices for Secure Software Report

Table of Contents

DOCUMENT REVISION HISTORY	3
CLIENT	3
INSTRUCTIONS	ERROR! BOOKMARK NOT DEFINED.
DEVELOPER	3
1. ALGORITHM CIPHER.....	3
2. CERTIFICATE GENERATION.....	4
3. DEPLOY CIPHER	5
4. SECURE COMMUNICATIONS.....	6
5. SECONDARY TESTING.....	7
6. FUNCTIONAL TESTING	7
7. SUMMARY.....	8
8. INDUSTRY STANDARD BEST PRACTICES.....	8

Project Two – Yanika Francis

Document Revision History

Version	Date	Author	Comments
1.0	August 15, 2025	Yanika Francis	

Client



Developer

Yanika Francis

1. Algorithm Cipher

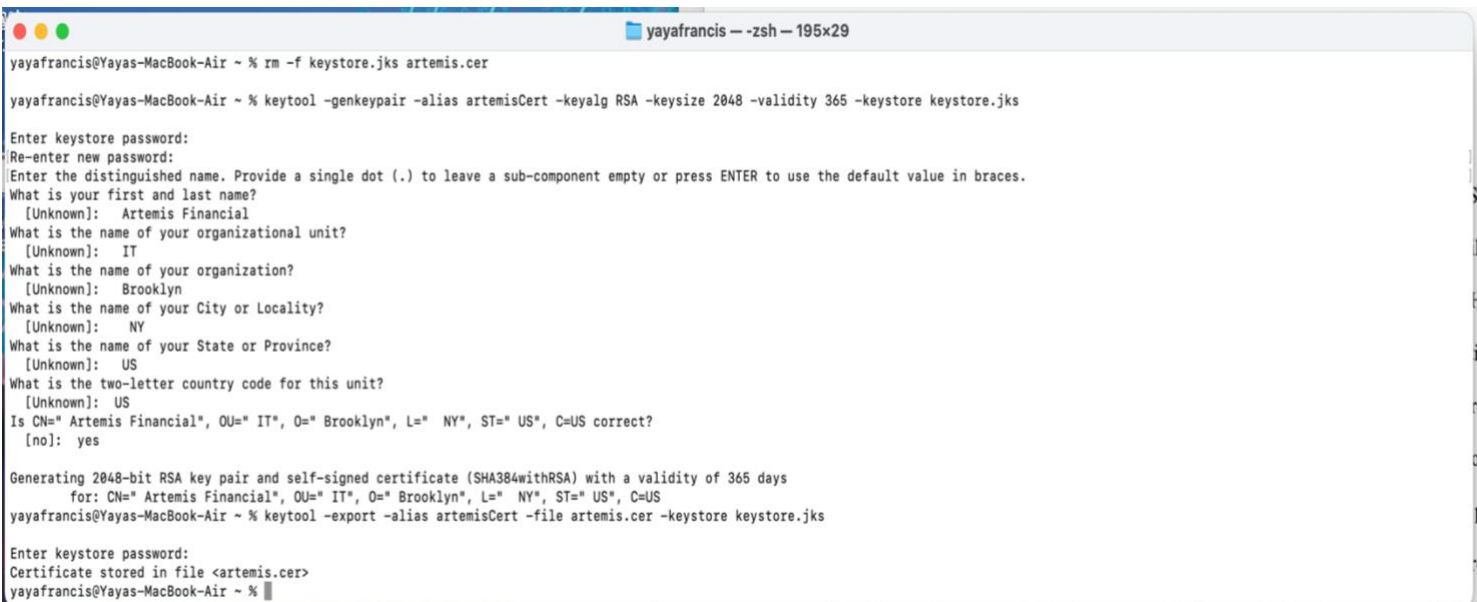
I suggest using the SHA-256 hashing algorithm. It belongs to the SHA-2 family and produces a 256-bit hash, making it very secure. When compared to older algorithms like MD5 or SHA-1, SHA-256 is much more strong. those are now regarded as outdated. You'll find SHA-256 used frequently, such as in SSL certificates, digital signatures, and file integrity checks. it functions as a one way hash, meaning you cannot reverse it to retrieve the original data. It also doesn't require keys like encryption techniques do. Instead, it makes a exclusive checksum for any input, helping confirm that your data hasn't been changed. While random numbers aren't really for hashing itself, they play an important role in encryption,

Project Two – Yanika Francis

especially for creating cryptographic keys. Right now, NIST suggests using SHA-256, and it's trusted worldwide as a safe standard. That's why Artemis Financial should use it to check files and keep client information safe.

2. Certificate Generation

I used the key tool command to generate a self-signed certificate for Artemis Financial. The screenshot shows the keystore being created successfully.



```
yayafrancis@Yayas-MacBook-Air ~ % rm -f keystore.jks artemis.cer
yayafrancis@Yayas-MacBook-Air ~ % keytool -genkeypair -alias artemisCert -keyalg RSA -keysize 2048 -validity 365 -keystore keystore.jks

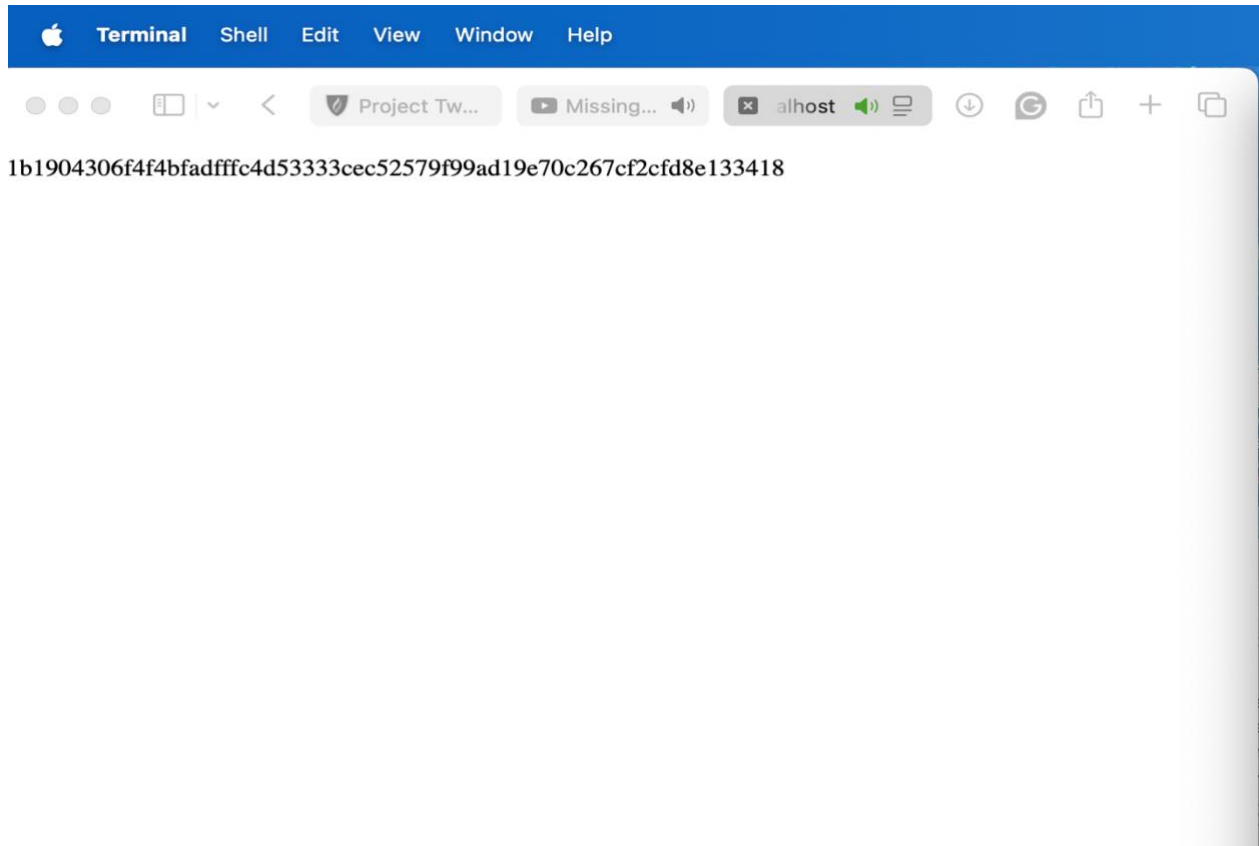
Enter keystore password:
Re-enter new password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces.
What is your first and last name?
[Unknown]: Artemis Financial
What is the name of your organizational unit?
[Unknown]: IT
What is the name of your organization?
[Unknown]: Brooklyn
What is the name of your City or Locality?
[Unknown]: NY
What is the name of your State or Province?
[Unknown]: US
What is the two-letter country code for this unit?
[Unknown]: US
Is CN="Artemis Financial", OU="IT", O="Brooklyn", L="NY", ST="US", C=US correct?
[no]: yes

Generating 2048-bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 365 days
for: CN="Artemis Financial", OU="IT", O="Brooklyn", L="NY", ST="US", C=US
yayafrancis@Yayas-MacBook-Air ~ % keytool -export -alias artemisCert -file artemis.cer -keystore keystore.jks

Enter keystore password:
Certificate stored in file <artemis.cer>
yayafrancis@Yayas-MacBook-Air ~ %
```

Project Two – Yanika Francis

3. Deploy Cipher



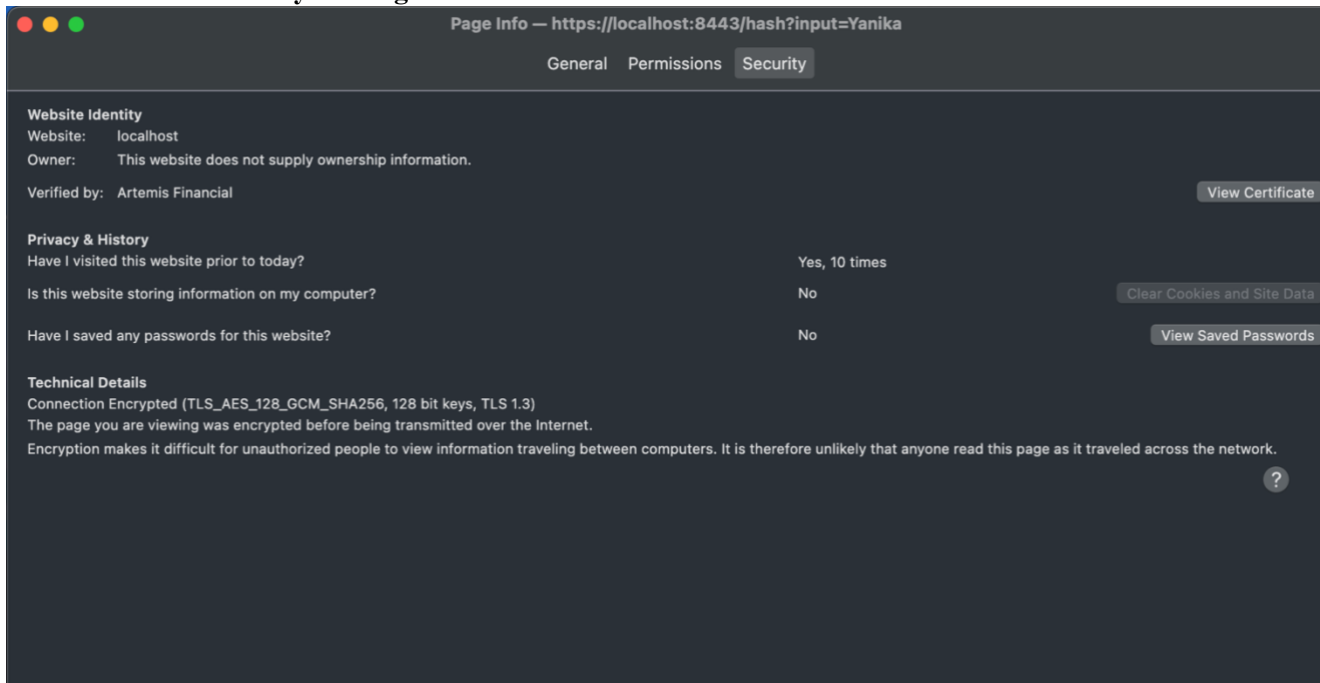
The application was successfully launched, and when you visit the /hash endpoint, it shows the correct SHA-256 hash.

4. Secure Communications



Project Two – Yanika Francis

5. Secondary Testing



The certificate information shows that the connection is secure and uses TLS_AES_128_GCM_SHA256 with TLS 1.3.

6. Functional Testing



Project Two – Yanika Francis

The server logs indicate that Tomcat started up and the Spring Boot application ran smoothly without any errors.

7. Summary

In this project, I created a Spring Boot server and added security with SSL using my own certificate. I made a keystore, configured the app to run on HTTPS at port 8443, and tested it by visiting the /hash page. The server revisited the correct SHA-256 hash for my input, and I looked at the SSL details in the browser. The logs showed that the server started without any errors. This shows that SSL was set up properly and the app functions securely. Not only securely but safe. Causing no errors in between.

8. Industry Standard Best Practices

Working on this project helped me better understand the hands on point of view of SSL certificates and HTTPS. I discovered how to create a keystore, set up a Spring Boot application to use SSL, and verify the configuration using a web browser. A problem I faced was viewing the certificate details in Safari, as the option was not immediately visible, but switching to Firefox or Chrome made it simpler. Also, I learned that browsers often warn about self-signed certificates, emphasizing the importance of trusted Certificate Authorities. Overall, this experience gave me hands on knowledge in creating secure connections and troubleshooting SSL setup problems.