GLOBALRAIN

Practices for Secure Software Report

Table of Contents

Document Revision History

Version	Date	Author	Comments
1.0	2024-10-24	Daniel Aguiar	

Client

ARTEMIS
FINANCIAL

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.

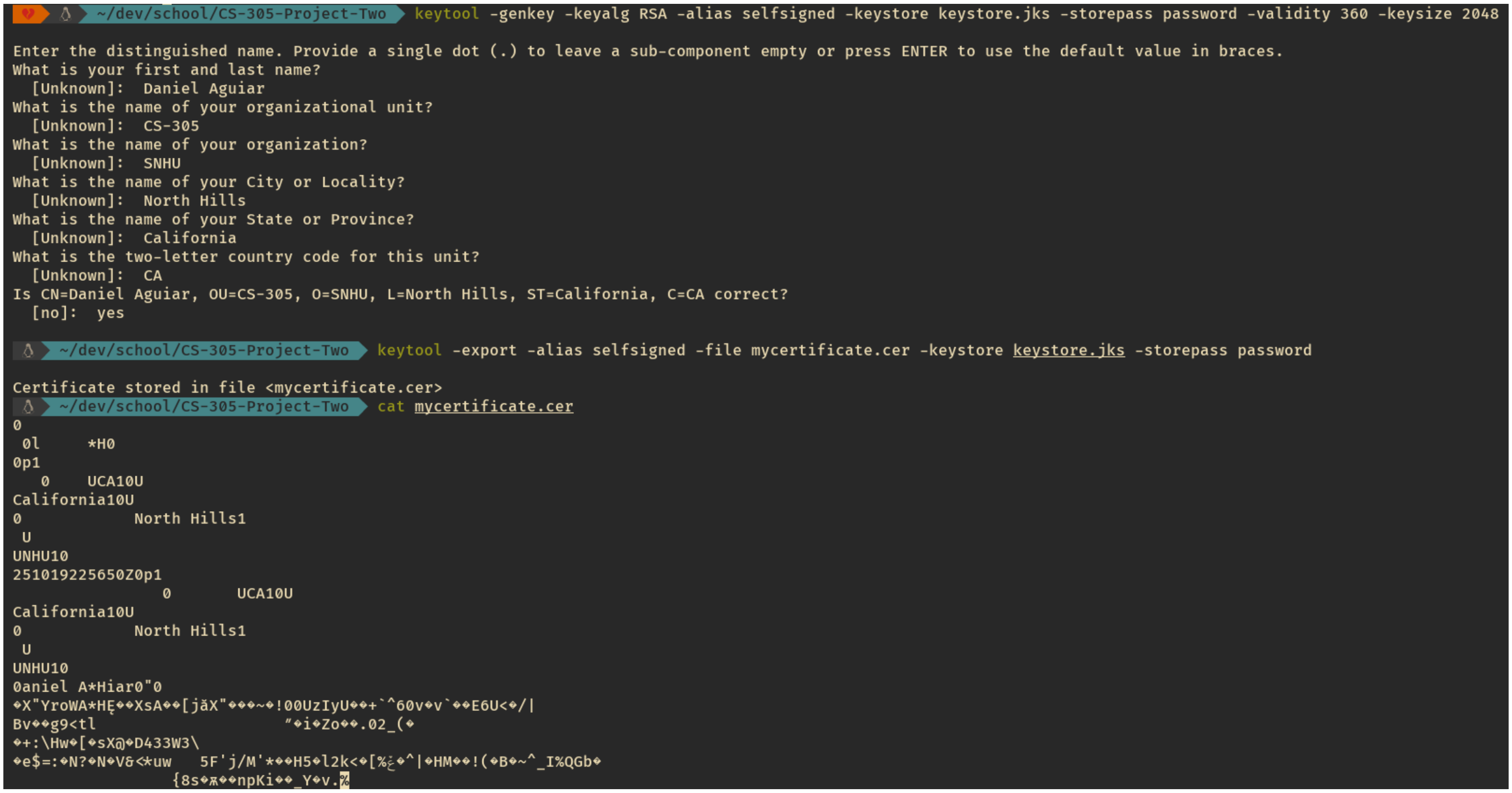
Developer
Daniel Aguiar

1. Algorithm Cipher

Artemis Financial needs to encrypt all of their long-term archive files. After reviewing available options, AES-256 (Advanced Encryption Standard with a 256-bit key) is the encryption standard recommended. AES-256 is symmetric key encryption that is widely recognized as strong and government regulation-compliant, including FIPS 140-2 (Federal Information Processing Standard). Due to the length of its key, it is highly resilient against brute-force attacks, making it suitable for securing sensitive data (National Institute of Standards and Technology [NIST], 2001).

2. Certificate Generation

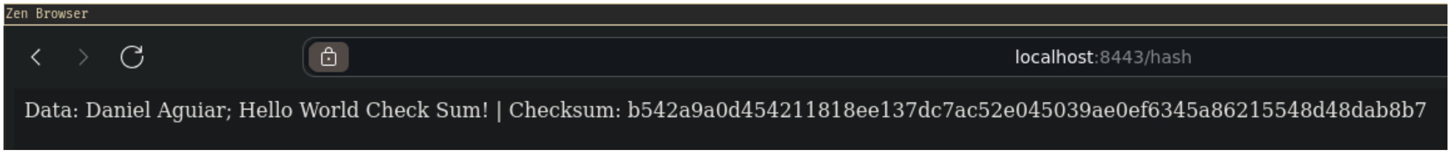
Insert a screenshot below of the CER file.



```
~/dev/school/CS-305-Project-Two  keytool -genkey -keyalg RSA -alias selfsigned -keystore keystore.jks -storepass password -validity 360 -keysize 2048

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]:  Daniel Aguiar
What is the name of your organizational unit?
  [Unknown]:  CS-305
What is the name of your organization?
  [Unknown]:  SNHU
What is the name of your City or Locality?
  [Unknown]:  North Hills
What is the name of your State or Province?
  [Unknown]:  California
What is the two-letter country code for this unit?
  [Unknown]:  CA
Is CN=Daniel Aguiar, OU=CS-305, O=SNHU, L=North Hills, ST=California, C=CA correct?
  [no]:  yes

~/dev/school/CS-305-Project-Two  keytool -export -alias selfsigned -file mycertificate.cer -keystore keystore.jks -storepass password

Certificate stored in file <mycertificate.cer>
~/dev/school/CS-305-Project-Two  cat mycertificate.cer
0
 0l     *H0
0p1
   0    UCA10U
California10U
0            North Hills1
 U
UNHU10
251019225650Z0p1
                0       UCA10U
California10U
0            North Hills1
 U
UNHU10
0aniel A*Hiar0"0
�X"YroWA*HĘ��XsA��[jăX"���~�!00UzIyU��+`^60v�v`��E6U<�/|
Bv��g9<tl                    "�i�Zo��.02_(�
�+:\Hw�[�sX@�D433W3\
�e$=:�N?�N�V&<*uw   5F'j/M'*��H5�l2k<�[%ݹ�^|�HM��!(�B�~^_I%QGb�
                 {8s�ҡ��npKi��_Y�v.%
```

3. Deploy Cipher

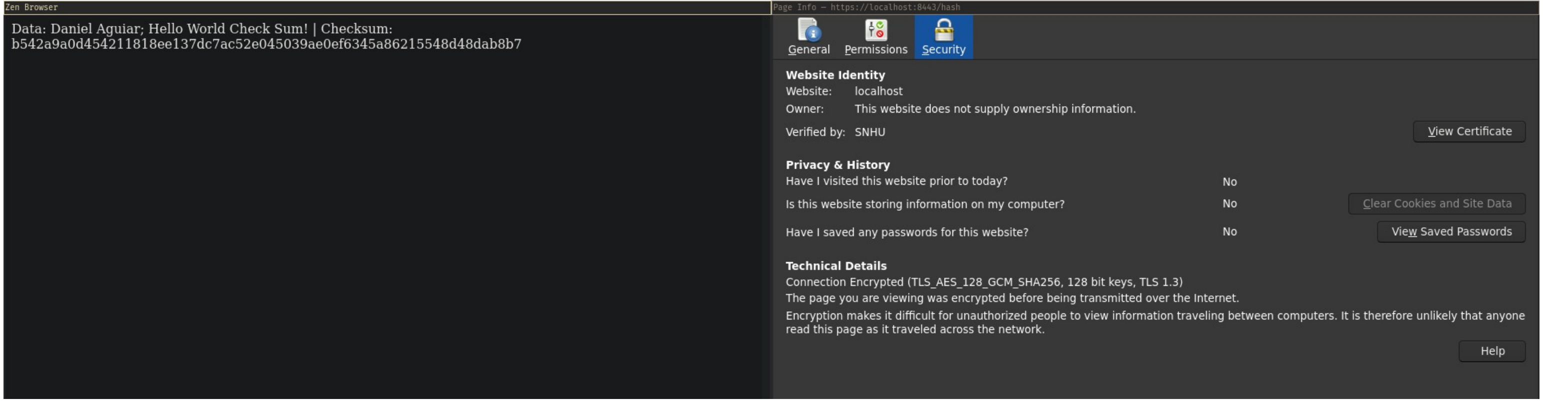
Insert a screenshot below of the checksum verification.



Data: Daniel Aguiar; Hello World Check Sum! | Checksum: b542a9a0d454211818ee137dc7ac52e045039ae0ef6345a86215548d48dab8b7

4. Secure Communications

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



```
Data: Daniel Aguiar; Hello World Check Sum! | Checksum:
b542a9a0d454211818ee137dc7ac52e045039ae0ef6345a86215548d48dab8b7
```

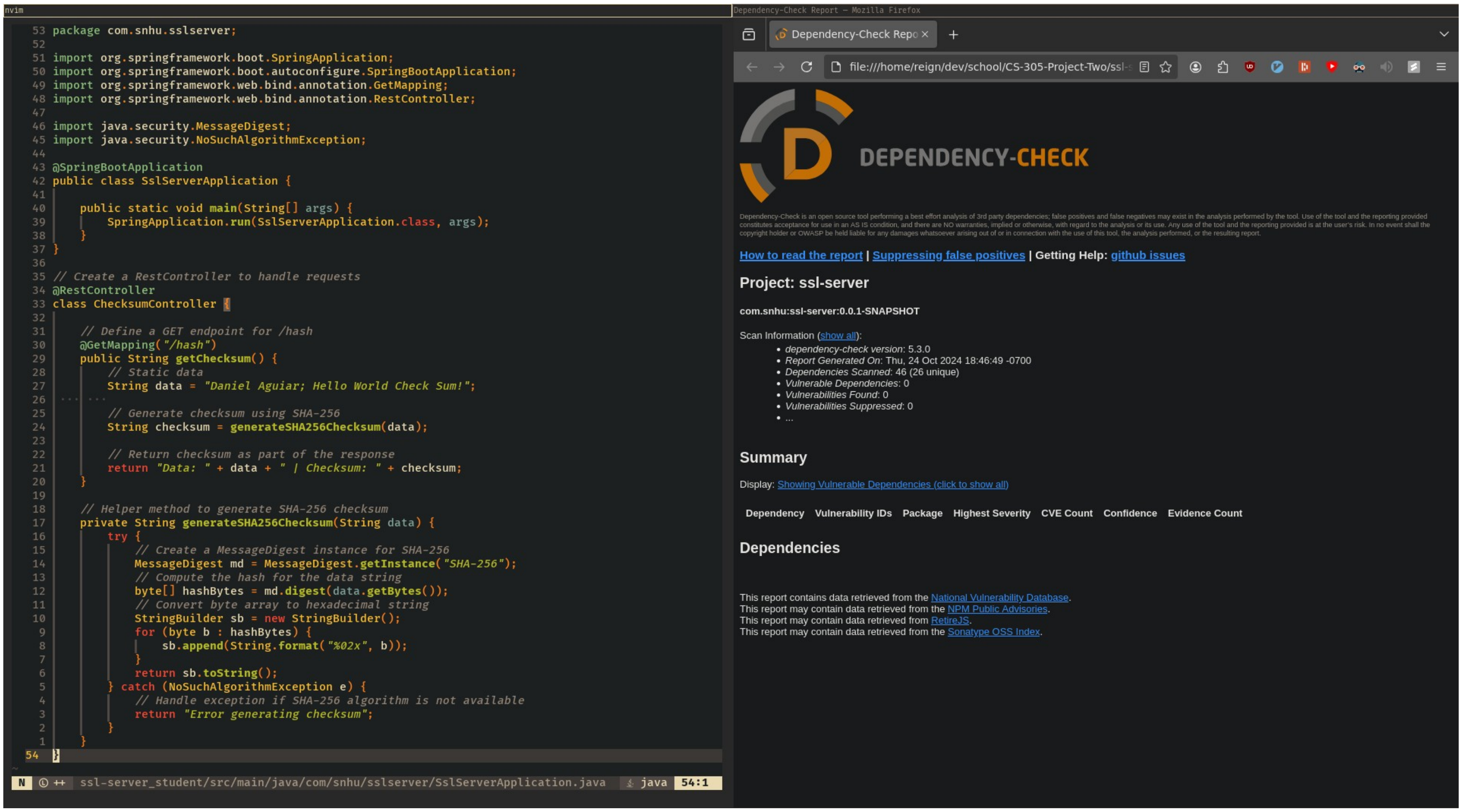
Website Identity
Website: localhost
Owner: This website does not supply ownership information.
Verified by: SNHU

Privacy & History
Have I visited this website prior to today? No
Is this website storing information on my computer? No
Have I saved any passwords for this website? No

Technical Details
Connection Encrypted (TLS_AES_128_GCM_SHA256, 128 bit keys, TLS 1.3)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.

5. Secondary Testing

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


nvim
Dependency-Check Report — Mozilla Firefox
Dependency-Check Repo ×
file:///home/reign/dev/school/CS-305-Project-Two/ssl-
DEPENDENCY-CHECK

```java
package com.snhu.sslserver;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

@SpringBootApplication
public class SslServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(SslServerApplication.class, args);
    }
}

// Create a RestController to handle requests
@RestController
class ChecksumController {

    // Define a GET endpoint for /hash
    @GetMapping("/hash")
    public String getChecksum() {
        // Static data
        String data = "Daniel Aguiar; Hello World Check Sum!";

        // Generate checksum using SHA-256
        String checksum = generateSHA256Checksum(data);

        // Return checksum as part of the response
        return "Data: " + data + " | Checksum: " + checksum;
    }

    // Helper method to generate SHA-256 checksum
    private String generateSHA256Checksum(String data) {
        try {
            // Create a MessageDigest instance for SHA-256
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            // Compute the hash for the data string
            byte[] hashBytes = md.digest(data.getBytes());
            // Convert byte array to hexadecimal string
            StringBuilder sb = new StringBuilder();
            for (byte b : hashBytes) {
                sb.append(String.format("%02x", b));
            }
            return sb.toString();
        } catch (NoSuchAlgorithmException e) {
            // Handle exception if SHA-256 algorithm is not available
            return "Error generating checksum";
        }
    }
}
```

```
N  ++  ssl-server_student/src/main/java/com/snhu/sslserver/SslServerApplication.java   java  54:1
```

Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, with regard to the analysis or its use. Any use of the tool and the reporting provided is at the user's risk. In no event shall the copyright holder or OWASP be held liable for any damages whatsoever arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

How to read the report | Suppressing false positives | Getting Help: github issues

Project: ssl-server

com.snhu:ssl-server:0.0.1-SNAPSHOT

Scan Information (show all):
- *dependency-check version*: 5.3.0
- *Report Generated On*: Thu, 24 Oct 2024 18:46:49 -0700
- *Dependencies Scanned*: 46 (26 unique)
- *Vulnerable Dependencies*: 0
- *Vulnerabilities Found*: 0
- *Vulnerabilities Suppressed*: 0
- ...

Summary

Display: Showing Vulnerable Dependencies (click to show all)

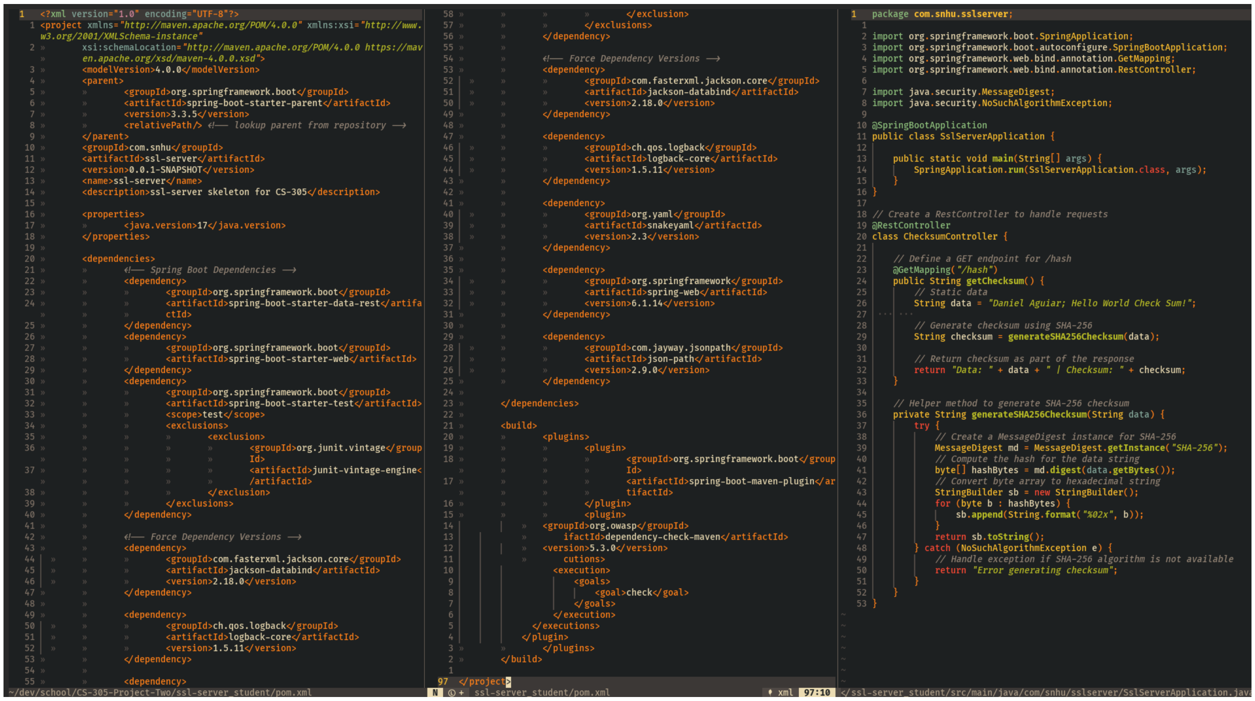
Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count

Dependencies

This report contains data retrieved from the National Vulnerability Database.
This report may contain data retrieved from the NPM Public Advisories.
This report may contain data retrieved from RetireJS.
This report may contain data retrieved from the Sonatype OSS Index.

6. Functional Testing

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



```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.3.5</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.snhu</groupId>
    <artifactId>ssl-server</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>ssl-server</name>
    <description>ssl-server skeleton for CS-305</description>

    <properties>
        <java.version>17</java.version>
    </properties>

    <dependencies>
        <!-- Spring Boot Dependencies -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-rest</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
            <exclusions>
                <exclusion>
                    <groupId>org.junit.vintage</groupId>
                    <artifactId>junit-vintage-engine</artifactId>
                </exclusion>
            </exclusions>
        </dependency>

        <!-- Force Dependency Versions -->
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.18.0</version>
        </dependency>

        <dependency>
            <groupId>ch.qos.logback</groupId>
            <artifactId>logback-core</artifactId>
            <version>1.5.11</version>
        </dependency>

        <dependency>
~/dev/school/CS-305-Project-Two/ssl-server_student/pom.xml
```

```xml
                </exclusion>
            </exclusions>
        </dependency>

        <!-- Force Dependency Versions -->
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.18.0</version>
        </dependency>

        <dependency>
            <groupId>ch.qos.logback</groupId>
            <artifactId>logback-core</artifactId>
            <version>1.5.11</version>
        </dependency>

        <dependency>
            <groupId>org.yaml</groupId>
            <artifactId>snakeyaml</artifactId>
            <version>2.3</version>
        </dependency>

        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-web</artifactId>
            <version>6.1.14</version>
        </dependency>

        <dependency>
            <groupId>com.jayway.jsonpath</groupId>
            <artifactId>json-path</artifactId>
            <version>2.9.0</version>
        </dependency>

    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
            <plugin>
        <groupId>org.owasp</groupId>
          ifactId>dependency-check-maven</artifactId>
        <version>5.3.0</version>
          cutions>
          <execution>
            <goals>
              <goal>check</goal>
            </goals>
          </execution>
        </executions>
      </plugin>
        </plugins>
    </build>

</project>
N  ssl-server_student/pom.xml                    xml  97:10
```

```java
package com.snhu.sslserver;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

@SpringBootApplication
public class SslServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(SslServerApplication.class, args);
    }
}

// Create a RestController to handle requests
@RestController
class ChecksumController {

    // Define a GET endpoint for /hash
    @GetMapping("/hash")
    public String getChecksum() {
        // Static data
        String data = "Daniel Aguiar; Hello World Check Sum!";

        // Generate checksum using SHA-256
        String checksum = generateSHA256Checksum(data);

        // Return checksum as part of the response
        return "Data: " + data + " | Checksum: " + checksum;
    }

    // Helper method to generate SHA-256 checksum
    private String generateSHA256Checksum(String data) {
        try {
            // Create a MessageDigest instance for SHA-256
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            // Compute the hash for the data string
            byte[] hashBytes = md.digest(data.getBytes());
            // Convert byte array to hexadecimal string
            StringBuilder sb = new StringBuilder();
            for (byte b : hashBytes) {
                sb.append(String.format("%02x", b));
            }
            return sb.toString();
        } catch (NoSuchAlgorithmException e) {
            // Handle exception if SHA-256 algorithm is not available
            return "Error generating checksum";
        }
    }
}
</ssl-server_student/src/main/java/com/snhu/sslserver/SslServerApplication.java
```

7. Summary

This project successfully enhanced the security in Artemis Financial's software through the implementation of AES-256 encryption, a self-signed SSL certificate to enable communication over HTTPS, verification of data integrity via a cryptographic hash, and by upgrading dependencies to fix vulnerabilities. I refactored code to support generating SHA-256 checksums and performed secondary testing with OWASP's Dependency-Check to ensure no new vulnerabilities were being introduced. Functional testing confirmed that data was handled securely; hence, all changes are appropriate and per Artemis's security goals and industry standards.

8. Industry Standard Best Practices

Industry-standard best practices to secure Artemis Financial software: AES-256 encryption, and HTTPS with a self-signed SSL certificate protects data in transit; use of the dependency-check tooling in code to keep that code secure by identifying known vulnerabilities. The project follows the OWASP guidelines and secure coding principles, which reduce risks and strengthen Artemis Financial data security.