# CS 305 Module Two Written Assignment Template

## Instructions

Replace the bracketed text with the relevant information in your own words. If you choose to include images or supporting materials, make certain to insert them in all the relevant locations in the document.

## Areas of Security

[The application, a Spring-based API, includes two methods (number and greeting) that return Greeting objects. It handles user input to either return array elements or generate personalized messages. Core security concerns include **Input Validation**, **API Security**, **Error Handling**, and **Data Encapsulation**.

1. **Input Validation** is essential to prevent errors or malicious inputs from compromising functionality.
2. **API Security** ensures the API can’t be exploited through insecure communication channels.

**Error Handling** must avoid exposing internal system details that could assist attackers.

1. **Data Encapsulation** protects sensitive information by using private fields and controlled access via getter/setter methods.]

## Areas of Security Justification

[i) **Input Validation** is prioritized to block malformed or malicious inputs, which are common attack vectors.

ii) **API Security** protects the integrity of client-server communication, especially important for apps processing user data.

iii) **Error Handling** should be generic to avoid leaking internal application details that attackers can exploit.

iv) **Encapsulation** reinforces secure coding practices by restricting direct access to internal data, aligning with principles of object-oriented programming and secure design in Spring applications.]

## Code Review Summary

[The Greeting class uses proper encapsulation with private fields and getter/setter methods, but GreetingController does not fully take advantage of this. Input in the number method directly affects output without sanitization, risking injection or array index errors. Similarly, the greeting method lacks proper input checks, raising potential for buffer overflows or malformed data. Error messages could reveal implementation details, and the pom.xml file shows outdated dependencies that need updating for security and compatibility.]

## Mitigation Plan

[To strengthen application security:

1. Implement strict **input validation and sanitization** for all user inputs.
2. Ensure consistent use of encapsulation through Greeting class methods.
3. Obfuscate or generalize **error messages** to prevent system detail leaks.
4. Regularly update **Spring Framework**, Java JDK, and third-party dependencies, and perform vulnerability scans on all libraries.]