# 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

* Input Validation: Secure Input and Representations
  + Input Validation ensures that all user supplied data is checked for things like data type, length, format, and range.
* APIs: Secure API Interactions
  + Secure API interactions help to ensure that sensitive data is not exposing sensitive data through unsecured APIs.
* Cryptography: Encryption Use and Vulnerabilities
  + The proper use of encryption can help to protect sensitive data and secure communication between devices.
* Client/Server: Secure Distributed Composing
  + Establishing a secure connection between the server and the client is extremely important and can be implemented/controlled in a variety of ways.
* Code Error: Secure Error Handling
  + Handling errors both in terms of user input errors and data errors can help to ensure that sensitive information is not exposed to the end users.
* Code Quality: Secure Coding Practices/Patterns
  + Secure coding practices help to ensure that the code is robust and bug free to prevent security vulnerabilities.
* Encapsulation: Secure Data Structures
  + Encapsulation can be critical to restrict access to internal fields and controlling the way data can be modified.

## Areas of Security Justification

* Input Validation: Secure Input and Representations
  + Input validation is critical to security because it can help prevent attackers from inputting malicious commands into the system or executing unauthorized commands to negatively impact the system.
* APIs: Secure API Interactions
  + Properly securing APIs can help to prevent attackers from bypassing authentication and accessing sensitive data.
* Cryptography: Encryption Use and Vulnerabilities
  + Using proper cryptography can help to prevent data breaches. This encryption can be used to protect sensitive data on the system so they remain safe.
* Client/Server: Secure Distributed Composing
  + Secure client/server communication helps to prevent attackers from intercepting or altering data while it is in transit.
* Code Error: Secure Error Handling
  + Proper error handling is critical for ensuring that errors do not inadvertently leak sensitive information to end users.
* Code Quality: Secure Coding Practices/Patterns
  + High quality code tends to have fewer security vulnerabilities and can reduce the avenues of attack. Avoiding low quality code helps to keep the code free of vulnerabilities, bugs, and helps to keep it resistant to attacks.
* Encapsulation: Secure Data Structures
  + Encapsulation helps to reduce unintended interactions between components in the system. This can help to reduce unauthorized access to sensitive data.

## Code Review Summary

The first security vulnerability I discovered in the Java package is the pom.xml file is outdated and should be updated to the current version. The spring boot version in the POM file is 2.2.4 and the most recent version I was able to find is 3.5.5. Updating to the current version helps to protect against security vulnerabilities and will help to keep the system secure.



The next issue I found was with the Greetings class. There is no input validation or limitations. This could lead to null values in the system or abnormally long data entered erroneously. This is a potential security concern and limits should be placed on what data the user is able to input into the system.

A screen shot of a computer code

AI-generated content may be incorrect.

In the GreetingController class there is also no input validation/limitation for “name” This can lead to the same issues as above and should be corrected to ensure that the code is robust and secure.

A screen shot of a computer code

AI-generated content may be incorrect.

The last issue I found was the lack of comments throughout the entire code package. While this itself is not a security concern, it can lead to future development issues. This lack of comments can cause confusion and lead to vulnerabilities in the future.

## Mitigation Plan

The first part of the mitigation plan would be to upgrade the Spring boot version to the most recent release to ensure that all known security vulnerabilities are patched. The next step would be to include input validation to ensure that user inputs are properly limited and checked for null values. This will help to ensure that user inputs are not able to adversely affect the system. It is also extremely important to add comments to the code to ensure everyone who looks at the code knows what each section is intended to do and interprets it accordingly. This will help the development team find any issues with the code quicker and will make future development more streamlined. This will also help to keep the code high quality and secure.