# CS 305 Module Two Written Assignment Template

## Areas of Security

The most relevant areas of security to assess out of the seven from the first level of the vulnerability assessment process flow diagram are input validation, API’s, client/server, and code quality. These four areas are relevant to assess because they address vulnerabilities from the web application of user input, system interactions, and communications.

## Areas of Security Justification

Input validation is important because the web application will have an expressive command input function for the application where the application can be vulnerable to attacks since it relies on the user to input commands. “Input validation is one of the “first lines of defense” when in any web application” (Manico & Detlefsen, 2014). Input validation ensures that the commands entered by the user are sanitized and prevent injection attacks.

API’s (application programming interfaces) are what is used for clients to interact with the system. Ensuring APIs are secure is important to protect the web application while the API’s process user commands to get or update data. This will help protect the system from unauthorized users interacting with the system and from any potential data leaks.

Having secure client/server focus is important because there will be communication between the client and server where attackers can impersonate users, session hijack, or steal data between the client and server. The web application will be using expressive command input where communication is from the user to send commands to the server that must be secure to ensure data confidentiality, integrity, and availability.

Having secure code quality avoids having improper code quality can increase vulnerabilities and create more issues in the code that can be exploited. Since web applications manage sensitive data from users, secure and quality code can help minimize or prevent breaches. Writing quality code also reduces the chances of having bugs and ensures that the application will run correctly. Having secure and quality code will help stay in compliance with legal requirements.

## Code Review Summary

The first step of the code review process is “views” and looking at the code the first issue I noticed is not using industry best practices of implementing in line commenting. Comments help with the readability of the code, especially when trying to debug.

Another potential vulnerability lies in the GreetingController.java file, when creating the array, it should provide an error message to the user rather than ending the system which could potentially reveal code structure to hackers. Using a vector would also help this vulnerability since vectors are safer and perform bounds checking.

The version of spring-data-rest-webmvc in the Spring framework being used is 2.6.5 and should be upgraded to the current version. According to the homepage of Maven Repository (n.d.) the current version to use is 4.4.0 or later. The image below shows the latest version of the Spring Data REST WebMVC 4.4.2 script that should be added into the pom.xml file for the application.

A computer screen shot of a computer

Description automatically generated

*Screenshot of Maven Repository pom.xml file script version 4.4.2* (Maven Repository, n.d.)

## Mitigation Plan

Implementation of mitigation techniques should be added to help improve the overall security of the code. Following the code review process will help ensure the code is secure and improves the overall quality of the code. One important mitigation technique is to implement is input validation to ensure the user input is sanitized from malicious data being added. Additionally, keeping the system and dependencies up to date is especially important to ensure systems are patched for well-known security vulnerabilities. By implementing these mitigation techniques, the code for the web application will be more secure and less vulnerable to common security threats.

**References**

Manico, J. & Detlefsen, A. (2014, September). *Iron-Clad Java: Building Secure Web Applications*. O’Reilly. <https://learning.oreilly.com/library/view/iron-clad> java/9780071835886/ch01.html#ch01lev1sec15

Maven Repository. (n.d.). Screenshot of Maven Repository [Screenshot]. [https://mvnrepository.com/artifact/org.springframework.data/spring-data-rest webmvc/4.4.2](https://mvnrepository.com/artifact/org.springframework.data/spring-data-rest%09webmvc/4.4.2)