# CS 305 Module Two Written Assignment

Alexander Freeders

## Areas of Security

Relevant areas of security regarding the scenario laid out in this week’s module include:  
Input Validation, API’s, Cryptography, Client / Server, and Code Quality. All these factors play a part in creating a ‘complex web application’.

## Areas of Security Justification

Input Validation – Since the application is using the Spring Expression Language, without proper sanitization of inputs, the application could be prone to injection attacks. It should be ensured that only proper input parameters for the indicated fields are enabled.

API’s – An Application Programming Interface essentially lets two software components “talk” to each other through various means and protocols based on why they were created. In this instance, it is important to ensure that we are utilizing API’s so our web application can interact with various other platforms, databases and the like to “talk” and keep information flowing.

Cryptography – Having sufficient cryptographic algorithms in place is important to ensure that information is not being siphoned off by a MitM or some other unauthorized actor. Since this is a web application, the usage of HTTP should be discontinued, and HTTPS should be implemented if not already. This factor is partially due to the vagueness of the project.

Client / Server – Somewhat like an API in terms of communication, the Client / server relationship is one that will be in play using our web application. Clients will poll the server for information / data / etc, and the server will reply with the requested resources. There are API’s involved, and the information that is transferred should be encrypted. Now that we can see how these aspects are building onto one another, it is certainly important to include client / server applications in the discussion.

Code Quality – Secure code is generally built from the ground up. Each implementation and revision should consider the security of the code and in what ways a malicious actor may try and bypass them. For instance, in input validation, a programmer would need to consider how they would handle exceptions; would they specify only certain characters? Whitelist appropriate commands? It is important to keep these considerations in one’s mind. Not only could a malicious actor bypass or infiltrate low-quality code, but oftentimes lower-quality code can have issues in utilizing memory effectively, contain memory leaks or even behave in an unintended way.

## Code Review Summary

In reviewing the 2.1 code in Eclipse, two things that stand out are the versions of Spring-Boot-Starter-Parent (SBSP) and spring-data-rest-webmvc (SDRW) . In the code, the SBSP version is listed as 2.2.4, while the latest version is *3.3.4*. Likewise, the SDQRW version in the code is listed as 2.6.5, while the latest version of that is *4.4.4*.

## Mitigation Plan

To mitigate potential threats and vulnerabilities, it is recommended that the team update the versions of their software to be current. Henceforth, there should be a single user test update to ensure that there are no catastrophic bugs, after a period determined by the security team, we can proceed with full team updates. This includes not only the SDRW or SBSP, but also other software being used by the team; to ensure that we are not leaving ourselves open, we should work to ensure that everyone stays up to date.