# VICTOR NGETICH

**CS 305**

**05/18/2025**

# CS 305 Module Two Written Assignment Template

## Areas of Security

### ****Areas of Security****

* **Input Authentication**: It’s essential to enforce strict checks on all user input in the command input function. Proper validation helps prevent injection attacks and ensures that only safe, expected data is allowed into the system.
* **API Integration**: If the command input function will be accessible externally, a secure RESTful API should be implemented. The API must be designed to block unauthorized access and should include input validation measures at the interface level.
* **Data Encryption**: When APIs are used, strong encryption protocols should be applied to secure both system data and user information. This prevents sensitive data from being exposed during transmission.
* **Secure Data Transmission**: For any client-server interactions, especially those involving HTTP requests data must be protected to maintain confidentiality and integrity throughout the communication process.
* **Code Security**: All code related to the command input function and API access should be carefully reviewed. This helps identify and resolve coding errors or vulnerabilities that could be exploited by attackers.
* **Encapsulation Practices**: Access to internal data through the command input function should follow proper encapsulation techniques. This ensures that sensitive data is managed and exposed securely within the system's architecture.

## Areas of Security Justification

Malware has the potential to severely disrupt software systems by undermining their integrity, authentication mechanisms, and overall availability. These threats can be significantly reduced when developers incorporate security best practices during the development phase, rather than attempting to address vulnerabilities after the software has been deployed.

## Code Review Summary

In reviewing our project setup, I noticed that the spring-data-rest-webmvc dependency listed in the pom.xml file is currently set to version 2.6.5. Since this version may lack recent security patches and improvements, I recommend updating it to the latest version, 3.3.3, to ensure better security compliance.

Additionally, while examining the GreetingController.java file specifically lines 27 to 30, I observed that data from the parser is being directly assigned to a string variable without any validation. This could pose a potential security risk if someone tries to input a maliciously formatted string. To address this, I suggest implementing a maximum string length limit and adding validation checks to prevent any unexpected behavior in the future.

## Mitigation Plan

First, I will update the spring-data-rest-webmvc dependency in our project to version 3.3.3 to ensure we are using the most secure and up-to-date release. Next, I will revise the parser logic in the GreetingController to include stronger input validation. This will involve setting a maximum limit on the input length to prevent oversized data from entering the system. Additionally, I will explore ways to detect potentially harmful input patterns, either by writing custom checks or integrating a reliable input validation library.