**Enhancement One Software Design and Engineering**

Anderson Forestal

Science department, Computer Science

CS-499-T4250 Computer Science Capstone

Southern New Hampshire University

Brooke Goggin

March 19, 2023

Enhancement of one software design and engineering

This paragraph is a narrative that accompanies the artifact, which aligns with the category of software design and engineering. The enhancement of the artifact selected demonstrates my proficiency, knowledge, and skills in implementing techniques to complete a polished project and the ability to solve complex problems in the software development process. This paper explains why the selected artifact was chosen and demonstrates my ability to use collaborative tools and techniques to implement the project to solve logical problems and address security flaws and defects embedded in software design early in development. Thus far, the goal is to develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources.

The original artifact contains many markers that indicate safety type that can lead to potential exploitation. With these safety types, the code executes successfully without crashing the program. As the code runs, the code compiles normally, and there are no untrapped run time errors or any conditional statements halting the execution of the code at any given point. Given this point, enhance the code as necessary. First, I added suppress\_warning to clear the marker to the top destination list. Next, as for the default list model, JList label, and the remaining questionable lines, I parameterized them to clear the markers since there were raws type references to a generic type of variables name. Second, I wrote a comment for each line of code to explain the code logic. Last, to ensure unauthorized users cannot access the program without validating their credentials. I implemented two Java classes that ask users to enter their credentials. In doing so, I created an authentication class that stored a user ID and password that was not sent to a server to be validated. Next, a login page class contained the logic to validate the user ID and password. Then, import both java classes in the main function so that both classes can be invoked. When the login page opens, the user clicks the login button and enters the user ID and password to access resources. If they match, then an alert dialog is shown, and the page reloads; otherwise, it displays an error message. Unfortunately, the login page is not working as expected because I wanted it to execute first and then ask the users to enter their credentials before the top destination list JPanel opens.

While modifying the artifact, I learned that enhancing the artifact not only revolving logical problems or security defects but researching documentation to apply to appropriate APIs and libraries for the corresponding language. With the project's implementations, I am strengthening my skills by integrating components to develop practical techniques to accomplish the goal.

References