Name: Steven Valdivieso Lemus

Class: CS-499-13167-M01

Date: March 23rd, 2025

3-2 Milestone Two: Enhancement One: Software Design and Engineering

Artifact:

The artifact is a Contact Management System written in Java. It allows users to manage contacts by performing CRUD (Create, Read, Update, Delete) operations. Initially created as part of the CS320 course that I participated in around a year ago, the system is focused on testing and ensuring correct behavior. The original version was functional but lacked robust input validation, error handling, and efficiency improvements. The artifact here is the original code and its enhanced version after applying modifications to improve its modularity, validation, and overall efficiency.

Justification for Inclusion:

This artifact showcases my skills in software development, particularly in the following areas:

Modular design: The refactored code is more modular, with helper functions for validation and error handling, which enhances code maintainability and readability.

Validation and error handling: The improved code introduces thorough validation of input data and proper exception handling, making the system more robust.

Performance improvement: Although the original code uses ArrayList to store contacts, the refactored version focuses on improving data validation and streamlining the processes to make the system more efficient.

I selected this artifact because it allowed me to apply real world best practices to a project and improve its functionality, aligning with the objectives of testing, validation, and modular design.

Specific Components Showcasing Skills:

Modularity: The creation of helper functions like isValidName, isValidPhoneNumber, and validateContact refactors the larger methods, making the code cleaner and easier to manage.

Input Validation: The original code lacked input validation, but I introduced checks to ensure contact details meet specific criteria, ensuring more reliable data input.

Error Handling: The use of specific exception messages enhances the clarity of error responses, helping developers and users better understand what went wrong.

Efficiency Improvement: The validation and error handling are improvements that will make the system more stable in its current form.

Course Outcome Coverage:

I successfully met the course outcomes related to software design, particularly:

Course Outcome 3: I designed and evaluated computing solutions that improved the original system through algorithmic improvements such as validation and a more robust structure.

Course Outcome 4: I demonstrated my ability to use techniques and tools like validation functions and helper methods to improve the system's functionality and ensure it works reliably in a real-world scenario.

Reflection on the Enhancement Process:

The enhancement process provided valuable learning opportunities:

Validation and Testing: I realized the importance of thorough validation to ensure that input data adheres to expected formats, which helps prevent runtime errors and data corruption.

Modularization: Breaking down large functions into smaller, focused ones helped me understand solid design principles, particularly Single Responsibility. This not only made the code easier to read but also easier to test.

Challenges: A significant challenge was ensuring that all validation checks were comprehensive while avoiding overly complex logic. It's also been a bit since I've fully sat down and worked with java in quite a time, so I had to remind myself of certain things about the language.