Process Reflection

Truc Le, Owen Woodall, Matthew Kingswood

In this assignment we were using the waterfall method in which we generate everything before the code is even started and then create it off of this plan and test afterwards. This waterfall method is similar to how a lot of the coding we have done in labs is with us getting a prompt and making a full program for it albeit a lot deeper for this assignment since we needed proper documentation and testing. Doing a whole piece of software like this simple as this assignment may be is fine for a project like this as there are no changing requirements nor is there any pressure to have people see anything other than a finished product. Not being able to test pieces as we go however, and needing to test it fully at the end does open up a lot of problems that we need to trace back a good amount if it fails and it can be fails in multiple parts that could’ve been prevented had we tested as we went. Since starting this assignment in class AI was important to generating the documents we needed and helping us by creating the functions and UMLs that allowed us to have a clear picture of this type of software since none of us had created software like this before. Given that help, however, ChatGPT was not without faults. During our generation it gave us suggestions for languages that we didn’t know and had to make it regenerate with ones we did, and we also asked it for pseudocode based off of UML it created. That pseudocode gave us functions that weren’t real that we had to point out to get it to regenerate. The UML it created was the basis for all our functions though and we used all of the classes, functions, and variables that it laid out. The pseudocode it did create with correct functions was long and split each function to an individual choice in main and so we consolidated it into about half of its choices. As stated before with a project like this there is a strength in that needing only a finished product having everything generated and done before starting give a very good layout and easy to follow guide for actually writing the code instead of creating then documenting. As we finished the coding up and hoped to do our testing we came to the realization that many things did not go our way. First we forgot to overload the operator for == and to include what we needed for the functions we needed. After that we found that our code doesn’t actually want to work properly as it only seems to want to take in one single input some times and thus we needed to fix that. As we go through this process of debugging if we could have been testing as we went instead of all at once at the end we could have found these immediately instead of after we are all done hoping that our code finally works.