Tammie Pease

CS 499

Artifact Three

Databases

The artifact that will be discussed here was made a long time ago, as it was my intro to how databases worked and what they were capable of doing. Intro to Databases was the class in which we were learning the basics of CRUD within it early 2021. What is left of the artifact is a document having us build a database and manipulate some items within it to show that we understood how to manipulate a database. We then gathered screenshots and info from the command line database software we were using for the class into a document and submitted it for a grade. The artifact is not exactly an artifact of coding itself, being that it was the compiled results of putting code through a virtual machine within the school. So, taking the data from this document and then placing it into another workable media was the main goal. I selected this item as it was one of my most complete works for databases that would be able to be revamped within the confines of the time given to me. Since this was my basics in trying to understand what was happening with MySQL, compared to my ability to create and understand coding has gotten better in a variety of ways. With those skills as well, another being how to look for answers I did not know of.

With the changes that I have made, I was able to have the artifact made into a code-centric file by making it within a different software that floats around similar database MySQL ideas but different language contexts. Later on, also add a few things to ensure it’s aligned within the scope of what I am able to do within the time specified. The point of this artifact’s glow up was that I was able to take very old information, have it partially retained in its ideas, and then spruce it up with new ideas. Such as the trigger feature within Oracle Live and also adding dates into the system for tracking orders and RMAs. I was able to do something difficult by taking old documentation and making a workable and meaningful file. Because of the fact that I was not able to pull data from a source to place within the database, I had to create my own data that would be manipulated within the code. So, in the end, I was able align the old data and using those concepts to make data that satisfied the idea of what the code was trying to do previously. Along with this, using the resources given to me I was able to help find online database software that could work for a place to code. But also, learning what to look for in software to work with for myself but also possible future companies. Together with this, I have learned about how solidify a scope for my plans, or else I will be prone to indecisiveness for what I want to implement.

Continuing on with what was gained with this artifact, I learned how to understand systems and the pros and cons they provide with what they are capable of doing. I made a trigger that placed a hidden ID into the systems of Orders and RMA for identification verifying each one is different that happens automatically when being implemented into the system. I have also implemented a date and date with time method in which both are implemented so they can be used for different instances. This artifact idea aligns closely with course outcome 04, in which it delivers value for the user and those who are querying for answers. As it allows for two options for trying to find orders or RMA in giving a date with solely a date but also a time for more detailed look if need be. Along with this, having a hidden ID made by the trigger that has value in which it creates something that no others can replicate within the system. This trigger hidden ID system makes it so that each value is unique and thus ensuring that the data can be fully organized even if there are similar people in similar locations with orders.

With this, however, I will make the argument for fulfilling 05 of the computer science outcomes as well, for it is something that deals with security but not of the code itself. Instead, it is of learning what materials and software to use and not to use. For the fact that when I was using Oracle Live, it was a mixed bag in being useful and providing useful skills but the system itself was a struggle to use. In which, if I spent 15 minutes looking and tinkering with code on my Word Document to ensure that it executes correctly the first time in Oracle Live, it will log me out. And yet, not at the same time as when I went to sign in again it did log me in without a prompt. It was very buggy and prone to logging out in a short burst of time, so it made it so eventually I wrote down my password right beside my computer so I can get back to work easier than before. This is a hindrance to security as it made it so I had to write it down my password to quickly get back into working again. Something that in reality we would want to be avoided for security purposes. Another problem was that when it came to MySQL, it was hard to manipulate as we went, as when I added a system into the program and if I did it wrong, those errors were still ‘tracked’ within the script sessions. Meaning I had to go into another tab with horrible loading times and then remove it. The UI didn’t properly finish commands such as never giving confirmations about an action taken until I had to double-check, then it would error because nothing was there. This software was hard to use, and I would likely avoid it later on. However, with learning about what software design flaws and ways it troubles the users and how they would react, being a user and reacting, can help me try to improve my work in the future. By ensuring that the software I make does not have these errors and not needing to jeopardize security because of it.

The struggles mostly were with the software itself for how it worked and ran, as it was rather clunky. As I was recreating it, it was a struggle to ensure that everything worked the way it did as it was a battle against the system itself. To make sure I didn’t have to battle with Oracle Live constantly, I placed all my working code into another document instead of hard coding it and going back again and again. But this documentation also had me track where I was and what changes I did to see if they would work or not. The coding document was an aide to ensuring I was making progress even if it was slower, it was always moving up. Another struggle was when I was learning how triggers operated within the system I was struggling as I was trying to do something with only ‘XXXXXX’ against the normal 32 Xs within example aids for triggers. So, I took a step back and took a break to then when calmed down from being overwhelmed was able to tackle and understand the error was mostly within the idea of needing the 32 Xs. Working with something with little documentation and vague errors made it hard to approach, so you always had to retrace your steps to what could be potential problems instead of actual problems. Taking the code and placing it into little data fragments to submit into a ‘script’ one at a time allowed for each chunk to test in the overall scheme of the program to see where the little problems were. And if any were found, I would go back to my word documentation with my process and then tinker it, refresh the schema, and go back to the beginning. Such a process was easier than having to go into the column and edit through as the software was… jarringly slow at best and unresponsive and confusing at worse. But another problem I had was the vague scope of what I wanted to do because it was vaguer than my other two. It created a case where I didn’t have a clear picture of my design and my ideas so it was more about what I could do instead of doing it. The battle was against myself and Oracle Live for this artifact, but I got to where I wanted to be!