My first artifact is a C++ recreation of the dice game Farkle. It was created for my IT-312 class on software design in C++. I chose this artifact for a few reasons. The first is that it is a feature complete version of the game Farkle that functions exactly how I envisioned it at the outset. My C++ Farkle project has good design, is well commented, and is very user friendly. The second reason I chose this project was that getting it to this state was a very involved and difficult process.I had to spend a lot of extra time expanding my knowledge on the different functions and features of C++. I had to teach myself how to use things like vectors, and expand my knowledge on pointers to be able to solve all of the hurdles I experienced in development. I chose to port this artifact to the Python language as I see the value of having that same experience with another useful coding language.

My experience with porting this game to Python was surprisingly fun. I really enjoyed the time I spent figuring out the solutions to my problems all over again, and the final product turned out great. It is another piece of well designed and commented software that meets all the features I had planned out in the design phase. This project also did exactly what I hoped in that I learned more about the Python language and was able to take advantage of some of its features. For example, I took advantage of Python’s ability to return multiple values from a function to meet the goals of what I originally had to use pointers for. Since Python does not really have pointers, understanding how it passes values by reference and other factors was very important to seeing how it differed from my C++ code. I am also happy with how much I was able to cut out of the program as it improved efficiency. A project that was nearly six hundred lines in C++ is now down to a little over four hundred lines in Python.

My second artifact that I chose to enhance was my Pokemon Battler that I had coded in Python. This was a project that I had started in my freetime when I was in classes that did not have me working with code. The program is an attempt to recreate the experience of battles in the Pokemon series of games. Coding this artifact originally was a fun experience as I had to try to solve all of the logic issues that come with trying to recreate an existing product that I enjoyed with no prior idea how they accomplished it. I chose this artifact to enhance for two reasons. The first is that it has many complex loops and decision structures to recreate the experience of the game. This artifact has great demonstrations of my understanding of these programming concepts. My second reason was that, since I made this in my spare time, the code was not commented much as I had not intended to use it for anything other than practice and to possibly show my friends. My planned enhancements for this artifact were to make it well documented with comments, add the feature of status conditions to recreate the experience better and improve complexity, and reduce redundancy by introducing functions for repeated lines of code.

Enhancing this artifact went well, but it turned out to be kind of difficult. While I had made the code, coming back with lack of comments meant that I had to spend more time relearning what I had written over a year prior. I appreciated this part of the experience as it showed me how I have grown as a programmer, since now I try to be much more clear and descriptive in my commenting and naming conventions. While someone jumping into my project would likely still need some knowledge of the Pokemon franchise to understand the code fully, anyone could now jump in and understand what my code does.

I was able to properly introduce the status conditions to make the battles closer to the real game experience. To do this, I had to modify my Pokemon class to account for statuses more than being knocked out, modify my move class to carry status conditions and infliction percentages, and introduce checks for the different statuses in the damage calculations and combat sections of my program. The statuses involved adding in more decision structures to check for the Pokemon’s current status, and then apply the desired effect based on the condition. It was also very important to position these algorithms properly, as some status conditions affect which Pokemon goes first in battle or the damage dealt by certain moves. On top of all these new introductions and complexities, I was able to reduce some redundancy when the user selects their character. Originally I had two loops and six variables that would account for player one and player two making their character choice, but I was able to cut those numbers in half by turning the selection into a trainer select function and calling it twice. All of these enhancements come together to make a more complex game that is pretty good at recreating the battle experience from Pokemon.

My third artifact I chose was a Python script to interact with an animal shelter database. I had made this for a class that involved databases and interfacing with them through Python and Mongodb. I chose this because it was one of the most involved experiences I have had with working with databases. The recommended enhancement that I chose to perform was creating a Mongodb interface using Javascript. I chose this enhancement because it would be very challenging. I have not worked with Mongo in about a year, and I actually did not know Javascript going into the project.

This enhancement was pretty simple, but getting to the finished product was very difficult. I had to learn and understand several components to get to this point. I had to learn Javascript and many of its strengths, weaknesses, and quirks in less than a week in order to finish on time. I also had to create a Mongo database, and relearn a lot of the functions it offers. Then I had to learn how to have a Javascript file interact with a cloud hosted Mongo database, which required me to learn how to operate and function with Node.js. It was a valuable experience as I learned how to deal with server promises, asynchronous functions, and the await keyword to ensure that my code does not continue to execute until the promise has been fulfilled or rejected. Upon completion, my script can now implement all aspects of CRUD as it can create, read, update, and delete records on a cloud hosted database.

Throughout the enhancement experience I was able to apply all of my software design skills that I have gained through my program and personal practice. It was also valuable in teaching me areas of Computer Science that I was less familiar with. I kept in mind things that I learned like, one of the most vulnerable points in software is receiving user input. When enhancing my programs I took this into account and switched my input variables from integers to character or string variables. This helps me avoid potential crashes due to typos or malicious actors as I can validate data without having my input stream crash. I was also very thorough in ensuring that I closed all streams that I opened whether they were text file streams or server connections. I am proud of the skills that I have gained and applied in my time in the Computer Science program, and I cannot wait to continue to grow as I pursue my professional career