Thor Nielssen

CS499: Computer Science Capstone

Southern New Hampshire University

9 June 2024

**CS499 Enhancement III Narrative**

**Running instructions:**

* I extracted to, then ran this program from, my C: drive at C:\AnimalShelter.
* Python 3.11/3.12 are required.
* Open C:\AnimalShelter in VSCode
* Open the Driver class
* Select Run from the ribbon, then Run Without Debugging
  + If prompted, run under the default Python Debugger
* The program runs and displays the main menu options.

Briefly describe the artifact. What is it? When was it created?

This artifact is my Rescue Animal final project from IT145. The project was originally done in early 2022, but it was also my first enhancement for this capstone course, when I redesigned and rebuilt it from its original Java code into Python. The Rescue Animal project used Java and array lists to store lists of monkeys and dogs to be reserved as service animals, allowing a user to input new animals as well as view the existing animal lists and reserve them based on input criteria.

Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development?

I selected this item because it exemplifies some of my earliest programming work. The original Java program depended on array lists that were held in memory while the program ran, then purged when it ended. This program fulfilled the requirements for IT145 but now that I have taken many more programming classes, I learned many ways to improve it. The specific components of this artifact that needed improvement were the language in which it was written and the way it manipulated data. Java is a great object-oriented language, but since the original project depended on the aforementioned array lists, it would be immensely difficult to implement better data structures. Completely re-writing the application in Python made the code more concise and easier to read and demonstrated a thorough understanding of both languages and how to design code in each. It also better equipped me to further enhance the artifact in the databases category.

How was the artifact improved?

1. I used SQLite, Python’s built-in SQL handler, to create a database with tables for each animal that lives in the same root folder as the rest of the application’s .py files.
2. This SQLite database persists between each run of the application, meaning that all data stored in the database is stored permanently and is accessible each time the application runs, as opposed to the temporary array lists that were held in memory.
3. The animal classes (Dog.py and Monkey.py) each contain more of their own dedicated code, reducing the complexity of the RescueAnimal.py and Driver.py classes.
4. The menu now functions as a true menu loop, allowing a user to repeatedly select and engage with menu options until they either force stop the application or input “q” when selecting menu options.

Did you meet the course objectives you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

I feel that this enhancement went a step beyond what I expected when I originally planned it and adequately meets the course outcomes I originally selected. The enhancement demonstrates effective software design choices in selecting a SQL technology to use, and properly addresses the databases requirement by creating a new database and all necessary methods and handlers to manipulate said database with SQL statements built into the Python code. I have no updates to my outcome coverage plans at this time since I feel that this enhancement came out better than I had hoped for.

Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

When I started this project, I intended to use MySQL to implement the database requirements and I struggled for quite a while trying to make that work. It was only while troubleshooting why MySQL wasn’t working that I realized that Python’s native SQL handler, SQLite, was better suited for this sort of local database implementation. My biggest challenge was not creating the database, because SQLite’s documentation was clear on the basic statements required to create a database and the tables within it. Rather, the difficulty came in finding a way to get rid of the original array lists and related syntax and replace them with SQL operators and syntax to fulfill the same purpose, but better. This took long hours of research for such a short and concise program, but the end product demonstrates that I persevered and finally arrived at a well-designed and functional solution.