CS 499 Milestone Four Narrative

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

This artifact is a course management program that uses a command-line interface along with a Binary Search Tree (BST) and CSV file for loading data. For this milestone, I enhanced the program by integrating a PostgreSQL database to provide persistent storage of course data.

1. **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? How was the artifact improved?**

I chose this artifact because it allowed me to demonstrate my skill in linking a database into a Python based application. Enhancing the artifact with a database allowed me to use tools like psycopg2 to connect to PostgreSQL, perform CRUD operations, sync between in-memory data structures and persistent storage, and ensure consistent data between applications. Understanding and utilizing databases is a necessary skill of a software engineer, as they often need to be able to manipulate and manage data across various databases. This enhancement helped introduce me to new applications and increase my exposure to learning new skills as well.

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

I believe I did, I was able to meet the planned outcomes from the first module by demonstrating data persistence using PostgreSQL, syncing between in-memory data with a Binary Search Tree and a database. Additionally, the object-oriented approach to appropriate file structuring supports a modular design and a clear separation of concerns regarding functionality.

1. **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?**

I was able to learn and understand more regarding in-memory data structures like a Binary Search Tree interface with a relational database like PostgreSQL. This enhancement helped improve my ability to implement persistent storage by adding logic that synchronizes course data across both applications in real time. Additionally, I made sure to consider writing code that separated the concerns regarding handling data in the database and data stored in the in-memory structure.

Several challenges I faced during development included preventing data duplication when syncing courses from the CST to the BST into the database. I was able to solve this resetting the BST before reloading the file and using PostgreSQL’s ON CONFLICT DO NOTHING statement to avoid inserting duplicate keys.

Another challenge I faced with adequately formatting and storing the course prerequisites. I had to figure out how to transform a dynamic list of prerequisites into a format that PostgreSQL would accept and store consistently. I solved this by ensuring the prerequisites were in a consistent format, especially across cases like insertion, deletion, updates, and being printed, all of which required string manipulation for adequate formatting.

Additionally, managing program state across sessions became increasingly complex as more features were added. Since the BST exists only in memory, I had to build functionality to keep the database and BST in sync without introducing conflicts or desynchronization. For this, I added reset and full-sync options that enabled users to ensure better state management across each application.

Overall, these challenges allowed me to improve my error-handling, input validation, and ability to design a command-line interface for data handling and user interaction with a database.