Robert Bostrom

08/03/2025

**Enhancement 3 Narrative – Databases**

The artifact I selected is the database layer of my Dash-based web application for Grazioso Salvare, originally created during my work in CS-340. Specifically, it’s the crud\_module.py file, which acts as a custom Python module to manage MongoDB operations such as reading, writing, and updating animal rescue data. This module connects to the database securely using environment variables and allows the application to filter and display animal records dynamically.

I chose this artifact for my ePortfolio because it represents a significant piece of the backend logic that powers my application and demonstrates my ability to write modular, efficient, and secure code for interacting with a NoSQL database. It shows that I can separate concerns by creating a reusable class to handle database access, a common practice in professional software development. The most important improvements I made to this module included adding field projection to limit the data returned by MongoDB, implementing an in-memory caching system to prevent redundant queries, and building a read\_df() method that returns clean, analysis-ready pandas DataFrames. I also added logic to clean and convert data types to prevent runtime errors in the dashboard. These improvements made the application faster, more secure, and easier to maintain.

Yes, I met the course outcomes I originally set for this enhancement in Module One. I aimed to show my ability to apply database management techniques effectively, and I believe this artifact now reflects that clearly. I don’t have any changes to my outcome-coverage plans because this enhancement completed my goals for the Databases category.

Through this enhancement, I learned how important it is to structure your data access layer carefully. One challenge I faced was handling inconsistent or missing data returned from MongoDB , for example, fields with mixed data types or unexpected ObjectId formats. I solved this by writing a cleanup method that automatically formats the data in a way that Dash and pandas can use without errors. I also gained experience with performance tuning through field projection and caching.