5-2 Milestone Four: Enhancement Three: Databases

Richard VanSkike

Southern New Hampshire University

CS 499: Computer Science Capstone

Professor Brooke Goggin

Artifact Overview

This milestone centered on integrating a PostgreSQL database into the Grazioso Salvare Dog and Monkey application, shifting from in-memory data handling to a robust database-driven approach. The original application managed animal data through local lists, which posed limitations in scalability and reliability. By implementing a PostgreSQL database, we enhanced the application's strength and future growth potential.

Key enhancements included significant modifications to driver.py, where in-memory data structures for dogs and monkeys were replaced with processes designed to communicate with the new database. Additionally, the initialize.py module was overhauled to encompass essential database functions, such as connecting to the database, creating tables, and retrieving data. These updates allowed the application to dynamically access and manage data, improving efficiency and accuracy.

A major new feature was the introduction of a secure login system through the addition of the security.py module. This system provided controlled access and established a more secure environment for users. By restricting database access, we significantly improved the security and integrity of the application's data.

To further enhance user experience, a graphical user interface (GUI) was developed using Tkinter. This interface transitioned users from a terminal-based system to a more interactive experience, incorporating tabs for easier navigation. The GUI also included a dark theme option, which can be toggled for user comfort, alongside a dynamic company logo banner that adapts to the theme.

Justification for Inclusion

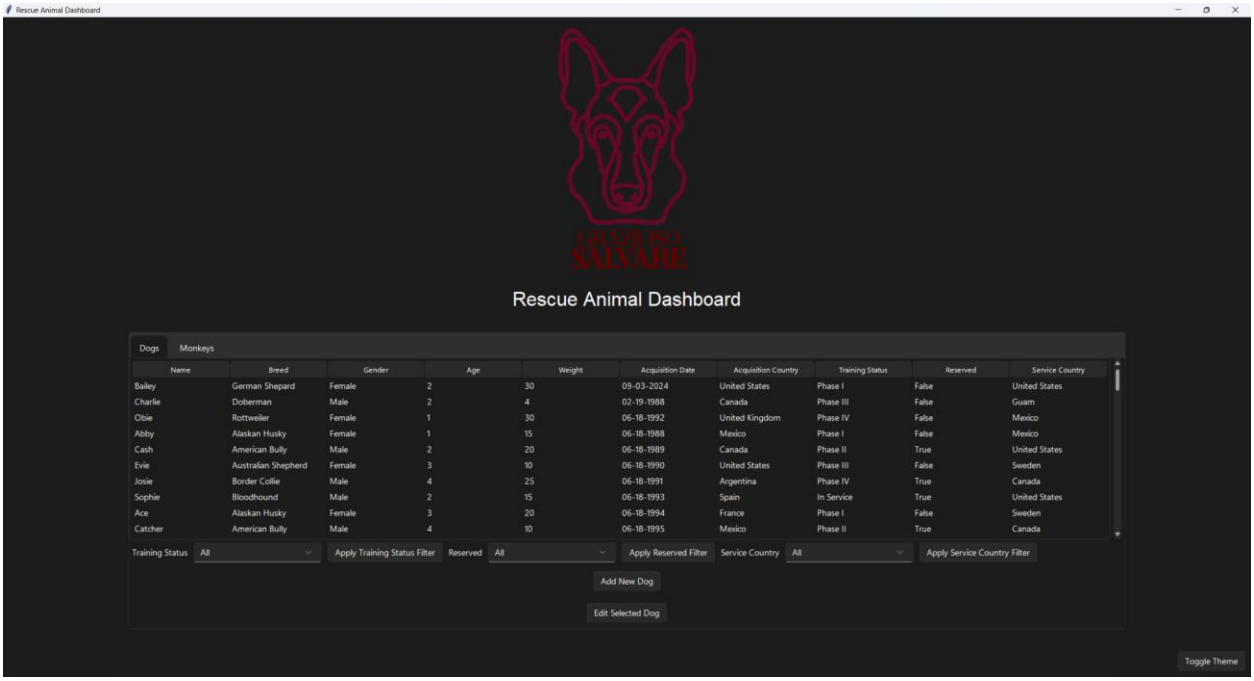
This artifact was chosen for my portfolio to showcase my ability to integrate advanced database technologies into existing applications, significantly enhancing their performance and scalability. Transitioning from in-memory data structures to a PostgreSQL database necessitated extensive modifications to the codebase, highlighting my proficiency in database management and software architecture.

Multiple changes emerged during this enhancement. First, the initialize.py module was revamped to encapsulate all essential database functions, allowing for seamless connection and data retrieval. Second, the new security.py module introduced a secure login system, consolidating essential login functions. This modular design not only promotes maintainability but also facilitates future extensions or modifications. By centralizing these operations, the application adheres to best practices in modularity and separation of concerns.

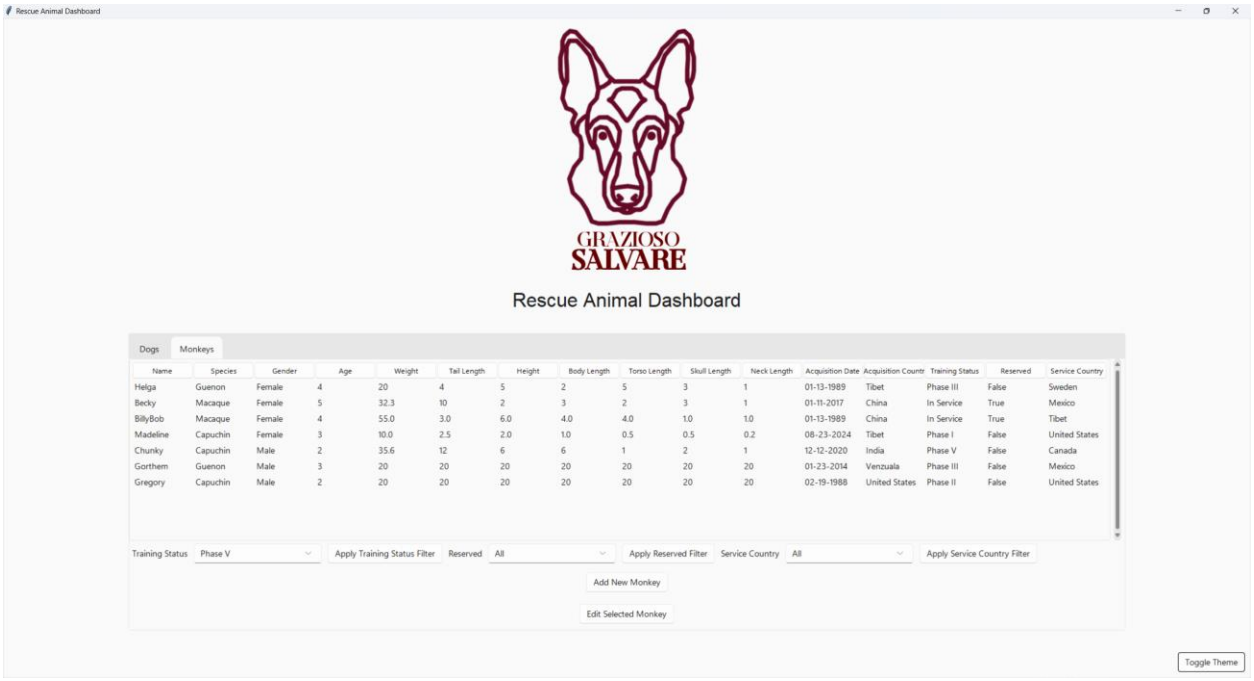
The final, and another significant addition was the integration of a graphical user interface (GUI) using Tkinter. This transition from a terminal-based system to an interactive GUI greatly improved user experience, providing intuitive navigation through tabs and features. The GUI also included a dark theme option and a dynamic company logo banner, enhancing visual appeal and usability.

Screenshots of Graphical User Interface

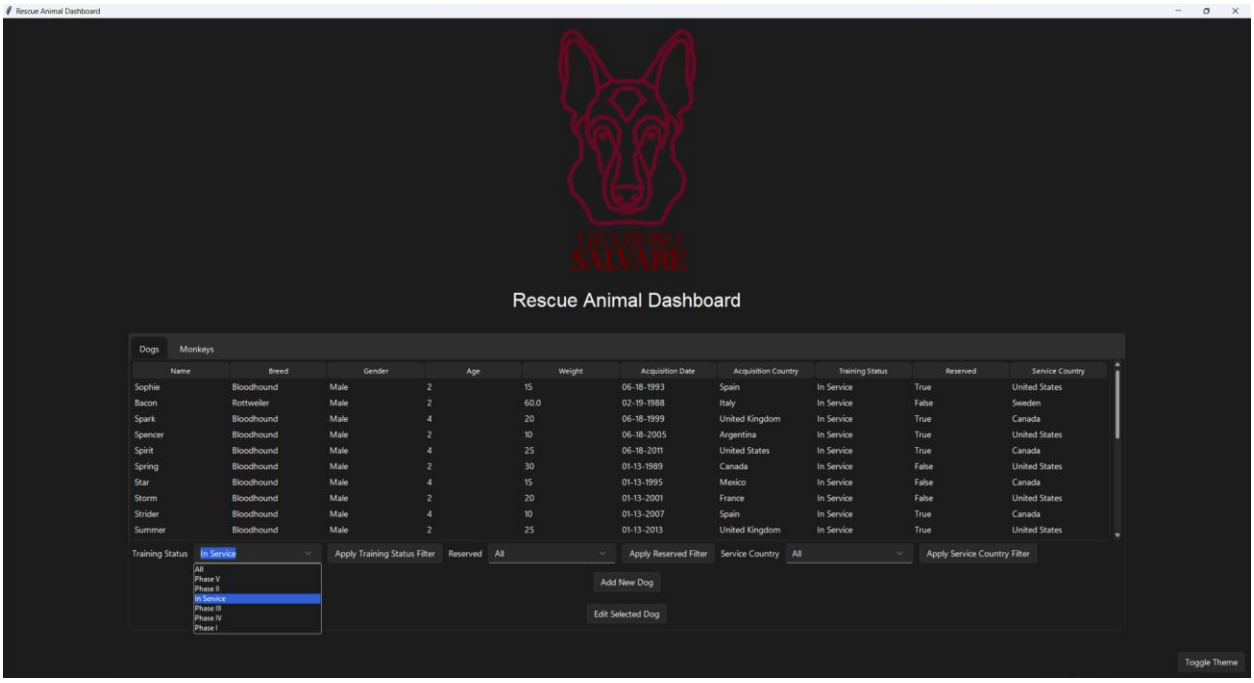
Main Landing with Dogs Selected



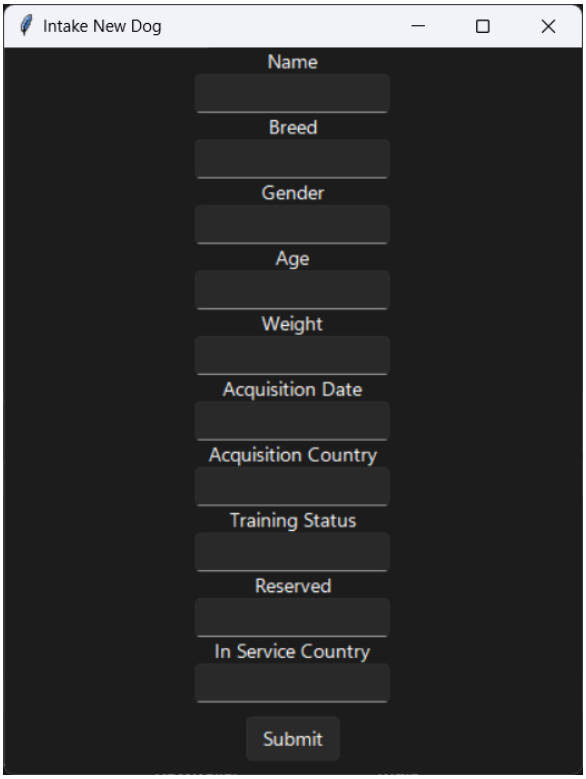
Main Landing with Monkeys Selected with Theme Toggled



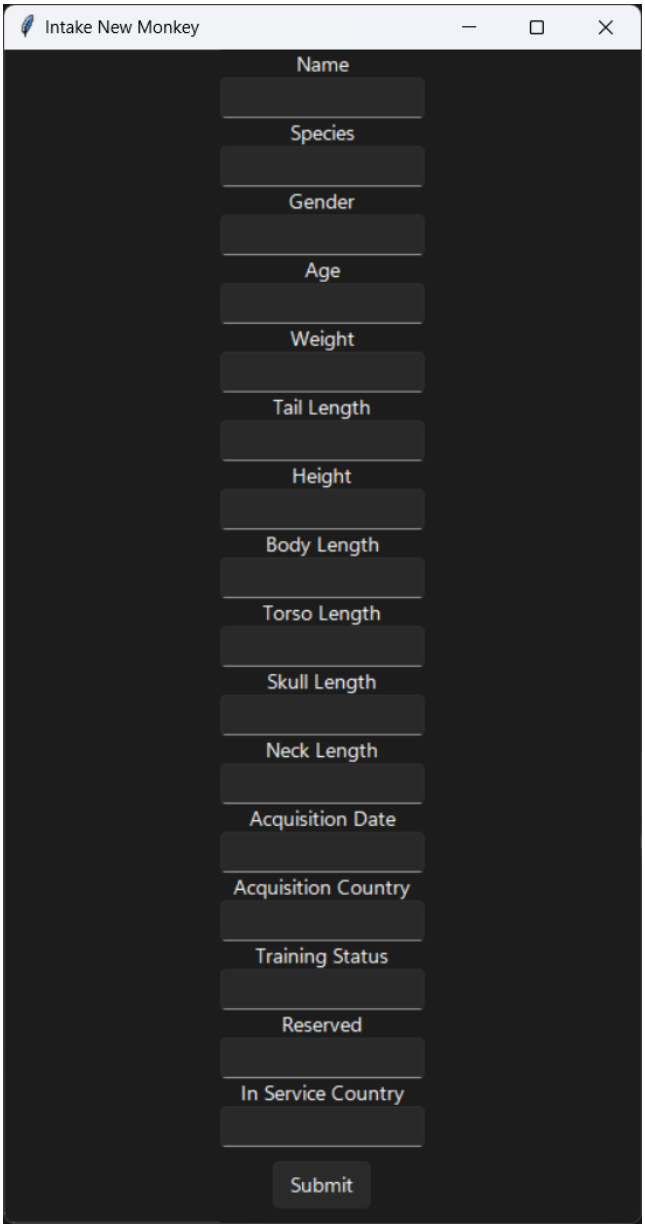
Example of Filters



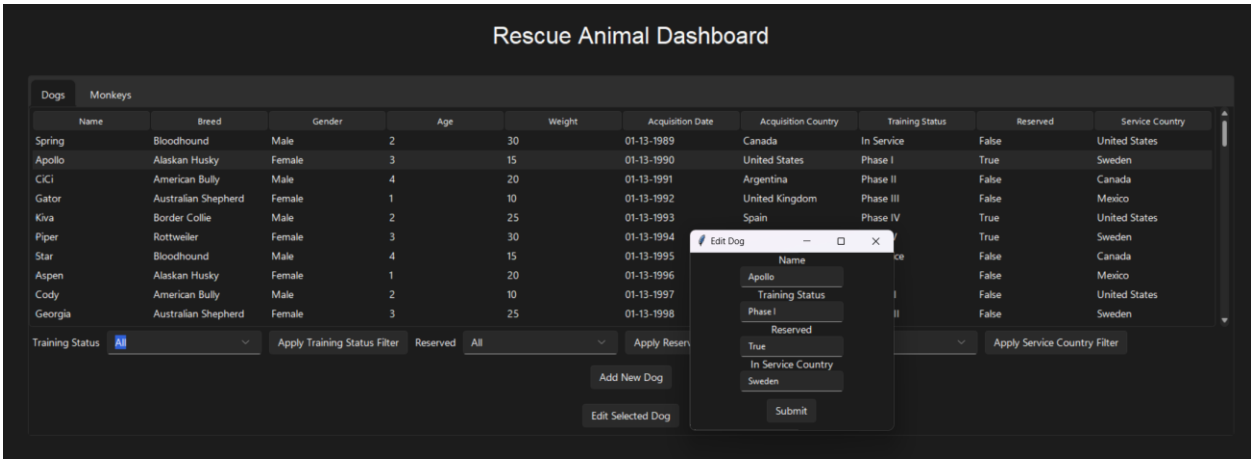
Add New Dog



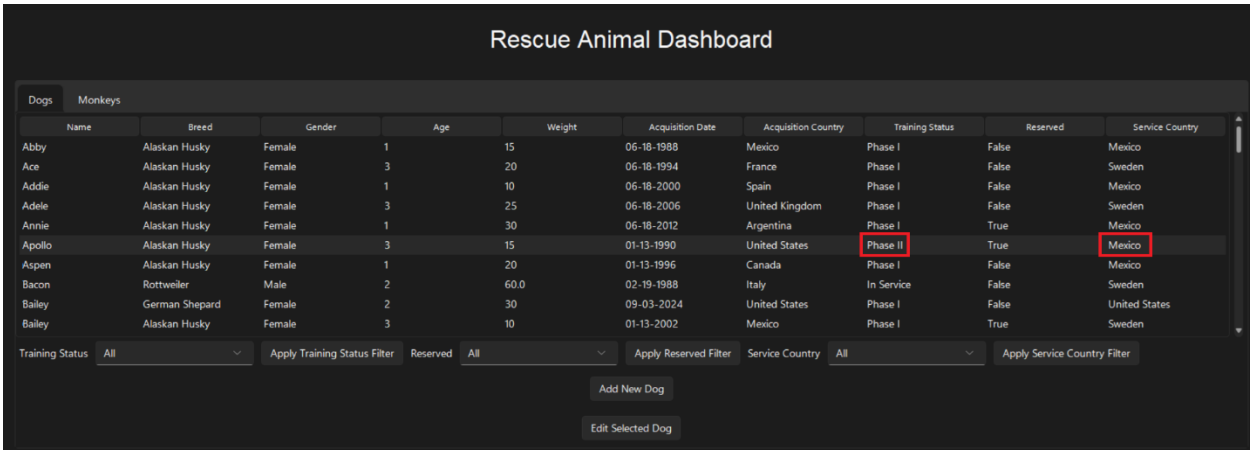
Add New Monkey



Example of Animal Edit Before



Example of Animal Edit After



By integrating the PostgreSQL database, the system changed how data is retrieved and manipulated. The implementation of functions like fetch_all_dogs and fetch_all_monkeys allow the application to query the database directly, leading to more efficient data handling, reduced redundancy, and improved performance. This enhancement underscores my ability to optimize data access patterns, ensuring the application remains scalable and responsive as the dataset grows.

Achievement of Course Outcomes

For this course, I am striving to reach five key course outcomes. They are as follows:

- Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision making in the field of computer science.
- Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.
- Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution, while managing the trade-offs involved in design choices.
- Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals.
- Develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources.

With Enhancement One, I transitioned the code from Java to Python and restructured the Grazioso Salvare Animal Rescue database to improve modularity and documentation, meeting algorithmic and structural goals. In Enhancement Two, I optimized the application's performance by transitioning from lists to dictionaries, which improved search functionality and scalability, and implemented input validation to enhance security. In Enhancement Three, I achieved four key outcomes.

First, I employed strategies for building collaborative environments that enable diverse audiences to support organizational decision-making in the field of computer science by moving all data to a centralized PostgreSQL database and providing a user-friendly Tkinter dashboard. This allows multiple users to access a consistent dataset and view real-time updates through the dashboard, enhancing collaboration and efficiency.

Second, I designed and evaluated computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution while managing the trade-offs involved in design choices by implementing the PostgreSQL database, secure login system, and Tkinter dashboard. These enhancements balanced the need for security and performance while ensuring that the application remained scalable and user-friendly.

Third, I demonstrated an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals by integrating the PostgreSQL database, developing the Tkinter GUI, and implementing bcrypt for password encryption. These innovations enhanced both the functionality and security of the application, aligning it with industry standards.

Finally, I developed a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources by implementing a robust secure login system using bcrypt hashing and salt generation. This system protected the application from unauthorized access and ensured that the data remained secure.

Reflection on the Enhancement Process

Integrating a PostgreSQL database into the Grazioso Salvare Dog and Monkey application was both challenging and rewarding. This required a solid understanding of database management and its application in software development. I utilized resources like GeeksforGeeks (2019 & 2022) to effectively implement PostgreSQL interactions in Python, ensuring the integration followed best practices.

Originally, the application used local lists then dictionaries to store animal data, limiting collaboration. By switching to a remote PostgreSQL database, multiple users could access and maintain it. I also implemented a security system for user login. After extensive research, I learned to convert strings to bytes, generate salts, and encrypt passwords with hashing, significantly enhancing security. Implementing the system required researching password hashing libraries. I ultimately chose bcrypt for its effectiveness in handling both hashing and validation, utilizing resources like Pypi (2024) to understand how to leverage it.

Another major challenge was refactoring the initialize.py module to replace in-memory data handling with database-driven processes. This underscored the importance of modular design and clear separation of concerns. Modifying driver.py to replace lists with database queries required meticulous attention to performance and accuracy, reinforcing my understanding of software design trade-offs.

To enhance user experience, I developed a GUI using Tkinter, moving away from a terminal-only interface. This involved creating tabs and managing data retrieval with lambda functions, which proved difficult. I faced challenges like ensuring real-time updates and correctly pulling unique IDs for editing. Debugging was essential to resolve issues with dropdown filtering and data selection. Through this process, I had to research Tkinter extensively, leveraging documentation from multiple sources such as RealPython (2023) and CodersLegacy (2021). This

research led to me implementing quality-of-life features like a dark theme using sv_ttk and a dynamic company logo banner.

Overall, this enhancement significantly boosted my technical skills in database integration and software architecture, emphasizing the importance of continuous learning and adaptation with new technologies. This experience was challenging but immensely rewarding.

REFERENCES

Bcrypt. (2024, July 22). *PyPI.* https://pypi.org/project/bcrypt/

Burgaud, A. (2023, December 1). *How to use python lambda functions.* Real Python. https://realpython.com/python-lambda/

GeeksforGeeks (2019, February 20). *Python: Getting started with Psycopg2-PostGreSQL.* GeeksforGeeks. https://www.geeksforgeeks.org/python-getting-started-with-psycopg2-postgresql/

GeeksforGeeks (2022, November 28). *Python: Database management in PostgreSQL.* GeeksforGeeks. https://www.geeksforgeeks.org/python-database-management-in-postgresql/

CodersLegacy. (2021, December 26). *Using lambda with "command" in Tkinter.* https://coderslegacy.com/python/tkinter-lambda/