

Enhancement Three: Databases

Christopher Sharp

Southern New Hampshire University

Dr. Greg Stefanelli

June 3, 2025

Enhancement Three: Databases

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

The artifact selected for Category Three, Databases, is an extension of the work I completed in Category One, Software Design and Engineering. To recap, the original work is a console-based application that was created in 2022 for IT-145: Foundation in Application Development. It is written in the Java programming language for a client, Grazioso Salvare, and their rescue animal operations. In this application, we had a menu system that received input from the user to perform different tasks. These included the intake of a new animal, displaying all the registered animals, and updating the status of the animal. In Category One, we refactored the codebase into Python to utilize libraries like tkinter for building a graphical user interface (GUI) and implemented data persistence by saving data in JSON format to separate files.

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 to expand on the code that I used in enhancement one because it can demonstrate my ability to take a simple, file-based approach and expand it into a more production-ready application. This also allowed me to expand on the knowledge that I have acquired from previous classes that dealt with using Python and libraries to integrate a MongoDB backend. By shifting to a NoSQL database, I demonstrate my ability to move beyond local JSON file storage into a scalable database solution, while also strengthening the overall architecture and user experience. It also highlights my understanding of data modeling, such as defining

collections, designing document schemas for the records, and ensuring data consistency. Also, included with this enhancement is the shift to role-based access control (RBAC) and secure user authentication. First, I created an `AnimalDatabase` class that handles the CRUD methods, so that the GUI logic and the database logic are separated (Figures 1 & 2). In MongoDB, I set up collections with clearly defined fields with validation rules to catch invalid entries so that the data does not become corrupted. Also, adding to this enhancement, I added a login screen where credentials are hashed and stored in a separate collection from animal data, while allowing for the separation of roles between users and admins (Figures 3, 4, & 5). This is especially evident as the GUI has been updated to be dynamic and update depending on the role of the logged-in user. Furthering this enhancement to the project, new users are prompted to update their passwords on first login. Finally, the filtering and loading logic has been shifted to use queries within MongoDB, which allows for the application to run smoothly and easily scale as the dataset becomes larger.

```
def create_animal(self, animal_data: Dict[str, Any]) -> bool: 4 usages (4 dynamic)  ⚡ Christopher Sharp
    try:
        self.collection.insert_one(animal_data)
        logging.info(msg: "Animal inserted: %s", *args: animal_data.get("name"))
        return True
    except errors.PyMongoError as exc:
        logging.error(msg: "Failed to insert animal: %s", *args: exc)
        return False

def read_all_animals(self, query: Optional[Dict] = None) -> List[Dict]: 6 usages (5 dynamic)  ⚡ Christopher Sharp
    try:
        return list(self.collection.find(query or {}))
    except errors.PyMongoError as exc:
        logging.error(msg: "Failed to read animals: %s", *args: exc)
        return []
```

Figure 1

```

def update_animal(self, animal_id: Union[str, ObjectId], updated_fields: Dict[str, Any]) -> bool: 2 usa
    try:
        result = self.collection.update_one(
            filter: {"_id": ObjectId(animal_id)},
            update: {"$set": updated_fields}
        )
        return result.modified_count > 0
    except errors.PyMongoError as exc:
        logging.error(msg: "Failed to update animal: %s", *args: exc)
        return False

def delete_animal(self, animal_id: Union[str, ObjectId]) -> bool: 2 usages (1 dynamic)  Christopher Sharp
    try:
        result = self.collection.delete_one({"_id": ObjectId(animal_id)})
        return result.deleted_count > 0
    except errors.PyMongoError as exc:
        logging.error(msg: "Failed to delete animal: %s", *args: exc)
        return False

```

Figure 2

Grazioso Salvare Animal Rescue Operations

Name Type Breed/Species Gender Age Weight Acquisition Date Acquisition Country Training Status Reserved In Service Country

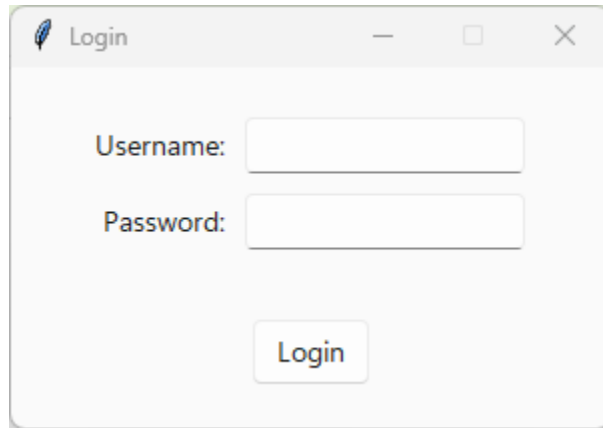
Actions

Login Load Dogs Add Dog Delete Animal Available

Load Monkey Add Monkey Toggle Reserved

Load All

Figure 3



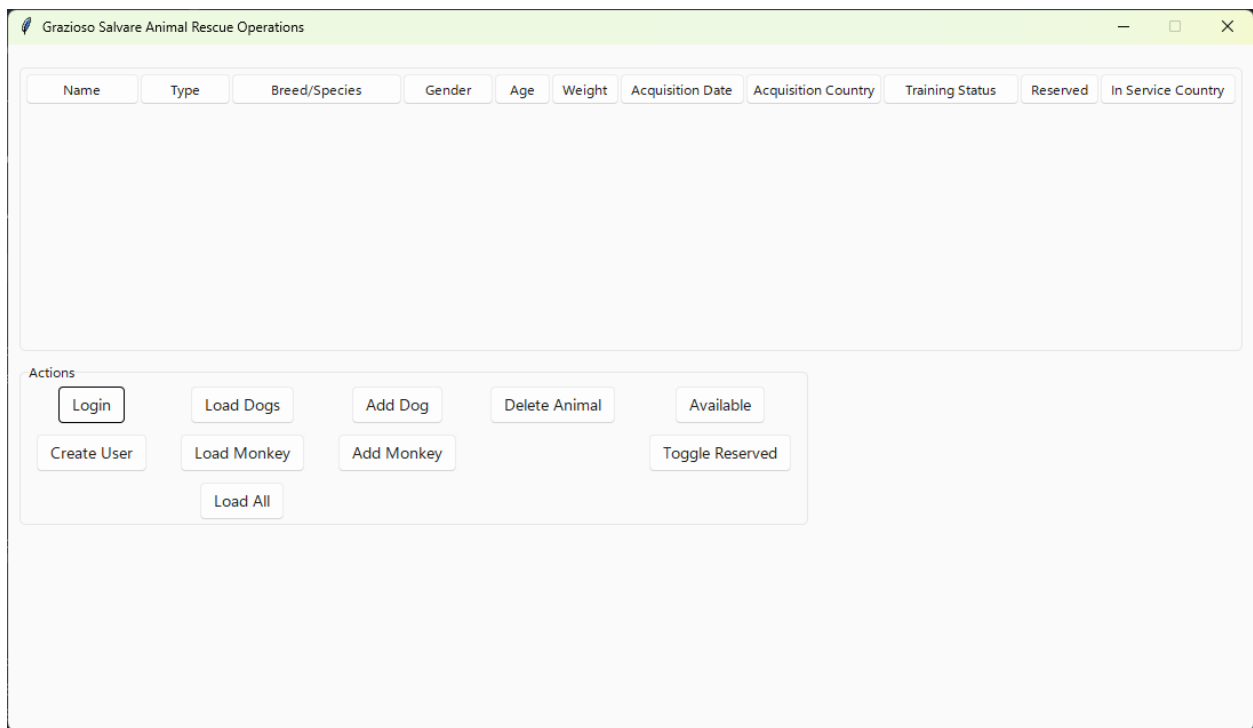
A login window titled "Login" with a feather icon. It contains two input fields: "Username:" and "Password:". Below the fields is a "Login" button.

Username:

Password:

Login

Figure 4



A window titled "Grazioso Salvare Animal Rescue Operations" with a feather icon. It features a table with 11 columns: Name, Type, Breed/Species, Gender, Age, Weight, Acquisition Date, Acquisition Country, Training Status, Reserved, and In Service Country. Below the table is an "Actions" section with buttons: Login, Load Dogs, Add Dog, Delete Animal, Available, Create User, Load Monkey, Add Monkey, Toggle Reserved, and Load All.

Name	Type	Breed/Species	Gender	Age	Weight	Acquisition Date	Acquisition Country	Training Status	Reserved	In Service Country
------	------	---------------	--------	-----	--------	------------------	---------------------	-----------------	----------	--------------------

Actions

Login Load Dogs Add Dog Delete Animal Available

Create User Load Monkey Add Monkey Toggle Reserved

Load All

Figure 5

**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?**

[CS499-01] Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision making in the field of computer science.

In this enhancement, I organized the code into clear modules that separate the different functions within the application with clear, concise comments that explain the code functionality. Also, included within the repository is a readme file that includes information on the code base, including a brief description, screenshots, and instructions on how to build and run the application. Furthermore, with GitHub Actions implemented to test, lint, and scan the code, this ensures that multiple contributors can implement the same processes to reduce any potential complications.

[CS499-02] Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.

For this enhancement, I provided code comments at the top of each module to explain the code functionality and within the code when it made sense to discuss the functionality even further. Also, included in the accompanying readme are sections that discuss the features included in the code, such as role-based access control. Furthermore, the readme provides documentation that includes detailed set up instructions, including any outside software that is needed. This provides a detailed and technically sound documentation for both users and developers.

[CS499-03] 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.

By analyzing the design tradeoffs of using the database instead of the JSON files, I was able to make an informed decision to improve not only the speed and performance of the application, but I was also able to increase the security of the application and data. For instance, using the JSON files required that the entire dataset be loaded into memory for filtering. By using modern queries for the database, I can quickly search for the documents that are needed for the filtering.

[CS499-04] 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.

During the enhancement, I employed iterative testing on the database schema to ensure that the proper data was being implemented and returned. In essence, I followed a short and compact version of the software development lifecycle to bring this enhancement to light. Also, I used industry-standard practices by separating concerns for the different levels of the application, such as data, models, and GUI. Finally, with the dynamic filtering of the database, I align with the goals of the original artifact. Also, with the implementation of GitHub Actions, I touch on the use of continuous improvement/continuous development that is found throughout the industry, such as automatic testing, code linting, and static code analysis.

[CS499-05] 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 this enhancement, I focused on expanding on security that was nonexistent from the start. For instance, document validation was implemented within the application so that only predefined fields and types would be accepted. Furthermore, with the implementation of role-based access control and hashed passwords, I ensure that the data is secure and unauthorized access is denied. In another instance, the application functions are locked down until the user logs in. Finally, users are required to change their passwords with their first login, so that the risk is lower than with the use of default passwords.

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?

The process of enhancing and modifying this enhancement was incredibly fulfilling because it allowed me to apply concepts I had learned in previous classes. The first step in the process was to ensure that MongoDB was still installed from previous courses. After that, I started working on the code base to change from using a file I/O system to using CRUD methods for the database. After these methods were created, I tested them to make sure they were working correctly. The next step was to integrate this new module into the existing GUI framework, as well as the changes that I implemented for role-based access control and dynamic GUI. The main thing that I learned from this experience was the ability to initialize the databases, even if they do not exist from the first run. Also, the role-based access controls that were implemented were a

concept that I knew but had never implemented. Also, the enhanced security features, such as hashed passwords and requiring a new password upon first login, were something I had not implemented into any other software that I have created. The biggest challenge that I had was how to set up the database. Since I am not using a cloud service for this, I had to come up with a way for new installations to initialize the correct collections for the database, with the correct parameters. In the future, this would be hosted on a cloud service to share all of the database information with multiple users.