**4-2 Milestone Three Narrative**

Name: Ami Akagi

Date: July 27th, 2025

**Overview of the Artifact**

This project was originally built to handle and visualize data from an animal shelter. It used MongoDB to store the information (JSON format), and put it together with Python, more specifically Jupyter Notebook and Dash. The main goal was pretty straightforward: get the data, organize it, and show it in a useful way.

**Why I Picked This for My ePortfolio**

I picked this artifact because I think it is a great place to show how to organize algorithms like dealing with duplicates, classification, and normalization, and how I work with data structures like MongoDB’s documents.

What the original version lacked in complexity, it made up for in potential. It gave me a great chance to show how I can take something basic and upgrade it into a cleaner, and improve it. For example, turning it into a more scalable and reusable API.

**Enhancements and Skills Demonstrated**

Here’s a summary of the enhancements I made and how they connect to various computer science concepts:

**REST API conversion:** I transplanted the logic into a DRF (Django REST Framework) API. This gave the system a lot more extensibility and made it easier to reuse with frontend frameworks.

**Normalization and classification (breed and rescue\_type ):** To make things safer and more consistent, I used ID-based operations and added logic to deal with inconsistent naming, and also added breed and rescue\_type adjustable data that is more flexible to store data.

**Duplicate Date Check Algorithm:** When adding new records, I used logic to check the animal\_id to avoid duplicate entries.

**MongoDB Environment Configuration:** I set up a more robust configuration and moved all my secrets to environmental variables to secure the database.

I also displayed integrated abilities in database design, algorithmic problem solving, and API design while creating these features.

**Connection to Module One Plans**

During Module One, I originally planned to organize a frontend system utilizing static JSON. But during development, I realized that not using a backend system for this project is not enhancing this artifact, and also a more real-world and straightforward solution would be to have an API with a database backend.

This allowed me to improve both the frontend idea and backend architecture, and the final product became much more complete and professional.

**Process and Learning Reflection**

Real-world difficulties during the enhancement process:

* Solving connection and authentication failed issue for MongoDB
* CSV Import Formatting Issues Resolved
* Understanding why some data is missing from the API

I was able to further reinforce my base developing API’s and definitely became a better developer for solving these issues.

**Looking Ahead**

In the future, I’d like to hook up this API to a frontend built in React or Dash so that it can offer a more user-friendly and visually intuitive interface. Now the backend is well poised for that next step.