Module 4 Narrative

Alexander Dabrow

Southern New Hampshire University

1 June 2025

Enhancement Two Objectives:

* Implement a page for users to upload a CSV file in order to add new animals – drawing from the LoadCourses method from courseProgram.cpp from CS-300.
* Take the insertion sort method from the courseProgram.cpp from CS-300 and port it to Java for use in the Spring Boot backend.

A computer screen shot of a program

AI-generated content may be incorrect.

This was the code used to load the CSV file in the C++ program. This original artifact was meant only for a console application, so it required the frontend aspect to be built from scratch.

The insertion sort method was more straightforward to implement. I also added another sort method using Java’s built in sort() method, as well as a means of calculating the time it takes for each sort method run.

A computer screen shot of text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

The times vary and is probably not the most accurate – you would really want an isolated environment where you have more control over the compute resources, but it’s a fun comparison.

**CSV Page**

Choosing a file:  
A screenshot of a computer

AI-generated content may be incorrect.

After choosing a file, the file name appears, and the “Upload” button will turn from gray to green signaling to the user that their file is ready to submit.  
A screenshot of a computer

AI-generated content may be incorrect.

Uploading the CSV:  
A screenshot of a computer

AI-generated content may be incorrect.

I included an alert to show the user all the valid data that will be added to the database. If there are invalid rows found, they will be excluded from the submission and logged to the console.

When the data is sent from the frontend, it is sent to as a POST request to the address “https://wonderful-sky-03a377e10.6.azurestaticapps.net/api/v1/animals/batch-upload”.

A screen shot of a computer code

AI-generated content may be incorrect.

The POST request is mapped to function takes a list of AnimalUploadDTO objects, which is a combination of dog and monkey data types, and returns a Response Entity with a map of string keys and values (for success and error messages).

Then in a try-catch block, we call the batchProcessAnimals function from the AnimalService class and pass in the animals list that is received from the frontend.

A computer screen shot of a program code

AI-generated content may be incorrect.

This function then goes through the list and separates the dogs and the monkeys into their own respective lists. After they are separated, we then save them to the database.A screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Here we can see with these two screenshots from Postman and a CSV file with test data, that the data was successfully added to the database.

A screenshot of a computer

AI-generated content may be incorrect.

**Sorting**

For sorting the data, we have defined the sort methods in the Animal Service class.  
A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

We then record how long it takes for these sort functions to run when they are called in the GET request function. There is an optional parameter that the function can take - a string that will determine the sort method to be used. If no string is given, then we just return the unsorted data.

In the response back to the frontend, we store the sorted list and the duration of how long the chosen sort method took to run.

**Database**  
I had to change databases from Azure SQL because I went over the free monthly limit in a matter of 2-3 days. A screen shot of a graph

AI-generated content may be incorrect.

This only happened after deploying and connecting the backend. I figured it must be draining it somehow with idle background connections.

I switched to using Azure Database for MySQL Flexible Server to avoid incurring a monthly cost as well as adding these parameters to limit idle connections in the application.properties file.  
A screen shot of a computer screen

AI-generated content may be incorrect.

Links:

Live site: https://wonderful-sky-03a377e10.6.azurestaticapps.net/

Frontend: https://github.com/alexdabrowSNHU/cs-499-frontend

Backend: https://github.com/alexdabrowSNHU/cs-499-api/tree/main