Module 5 Narrative

Alexander Dabrow

Southern New Hampshire University

15 June 2025

The enhancement I chose for the database aspect of the Capstone project was to persist data between sessions with the application from IT-145, the rescue animal management system.

Previously, the application stored the data in ArrayLists.  
A screenshot of a computer code

AI-generated content may be incorrect.  
This is great when working with and manipulating data, but any changes that are made to those lists are lost when the application terminates. To save changes that are made to the data, we implemented a solution using Azure Database for MySQL flexible server.

The original plan was to use Azure SQL, but because of improper configuration, it led to draining the free trial resources, so to stay on a budget of $0, we switched to using Azure Database for MySQL. This was an easy switch as we are using the Java/Jakarta Persistence API (JPA) which abstracts most of the database layer. The only changes necessary were adding the MySQL dependency in the pom.xml, and changing the connection properties in the “application.properties” file.

Since the animal types we are working with inherit from a parent class of RescueAnimal, I decided to store the parent information in the a parent table, and then the children’s unique attributes would be stored in child tables that are then linked via foreign key using the animal ID that links it to the parent entity in the parent table.

In our RescueAnimal class, we define the parent table name and utilize the JOINED InheritanceType from the Jakarta Persistence library.  
A screen shot of a computer

AI-generated content may be incorrect.

A blue background with white text

AI-generated content may be incorrect.

This is perfect for our application since that is exactly what we are doing, fields of a specific subclass, like “breed” from “dogs” are mapped to a separate “dogs” table, while the attributes that are found in the parent class of “RescueAnimal” are mapped to the table of “rescue\_animals”.

Here you can see the schema better visualized with MySQL Workbench.

A screenshot of a computer

AI-generated content may be incorrect.

The motivation for this strategy was to avoid wide and confusing tables. Say we started to include different animal types, like cats, horses, etc., and we stored all animals under one table and one of those columns defined its type. We would then have to make additional columns for each animal type’s unique attributes and these columns would have to be included for animals that do not have those attributes. Our strategy we have implemented keeps each animal type’s attributes in their own respective table.

**Security**

To ensure connections to the database are only achieved through authorized means, we must take care in managing the credentials used to access the database. Prior to the changes made in this enhancement the username and password used to access the database were hardcoded into the pom.xml and the application.properties file. This is not good practice, so we have found a solution in using Azure Key Vault.

To implement this solution, we first needed to add the dependency to the pom.xml in the Spring Boot application.  
A screen shot of a computer code

AI-generated content may be incorrect.

We also have deleted the hard coded credentials in the pom.xml as well. Credentials are only necessary in the application.properties file.

Then we created and deployed a Key Vault.  
A screenshot of a computer

AI-generated content may be incorrect.

Then we added an IAM role that permits it to read secrets.A screenshot of a computer

AI-generated content may be incorrect.

Then we added our Spring Boot application as a member to assume that role.

A screenshot of a computer

AI-generated content may be incorrect.

Then we created the secrets for the username and password.  
A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

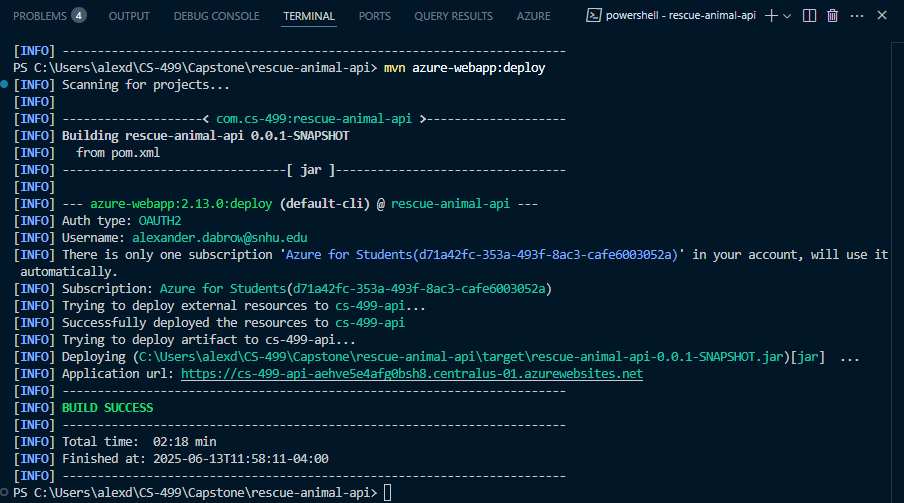
AI-generated content may be incorrect.

Then we updated the application.properties file to use the secrets.  
A screen shot of a computer program

AI-generated content may be incorrect.

We can see now that the username and password use placeholders which will get their values from Azure Key Vault.





Then we deployed the new build.

Screens screenshot of a computer

AI-generated content may be incorrect.  
We have now successfully deployed the Azure MySQL database using Azure Key Vault for storing credentials securely.

Credentials should never be hardcoded into a codebase. Using services like Azure Key Vault to securely store and access credentials help to maintain a robust layer of defense for the data being stored in our database.

The enhancement of this artifact aligns with the course outcome of developing 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.

To accompany this artifact in the final enhancement, I will include a narrated video discussing and comparing some popular Azure database services. This is to design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.