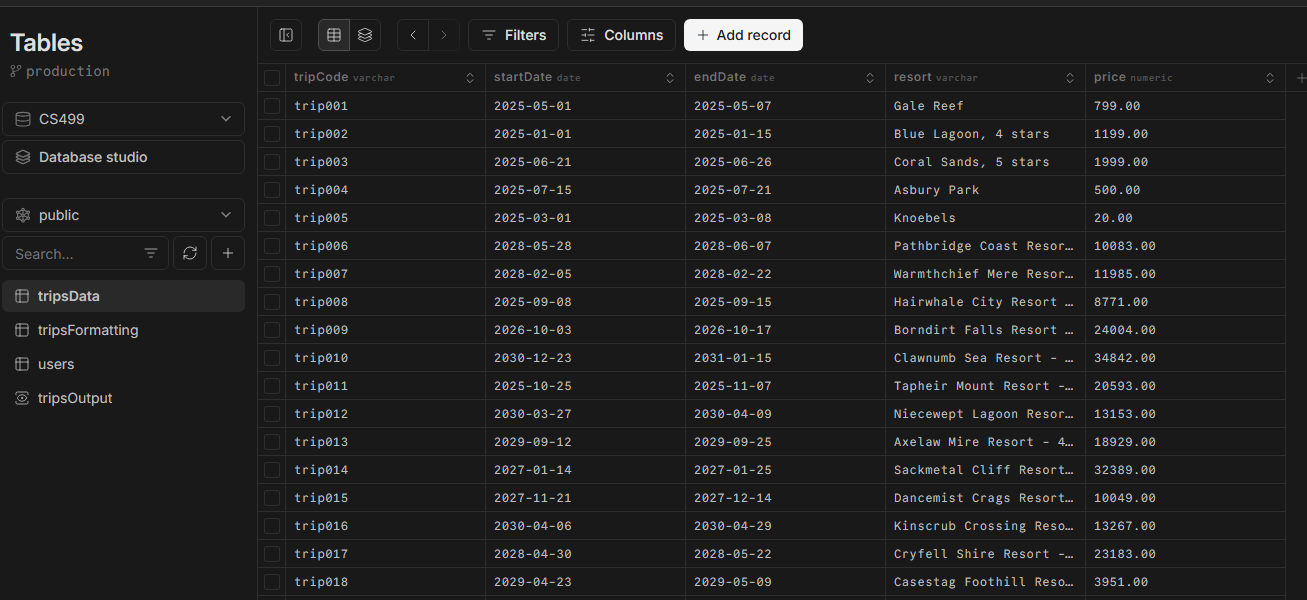
# CS499 Milestone Four

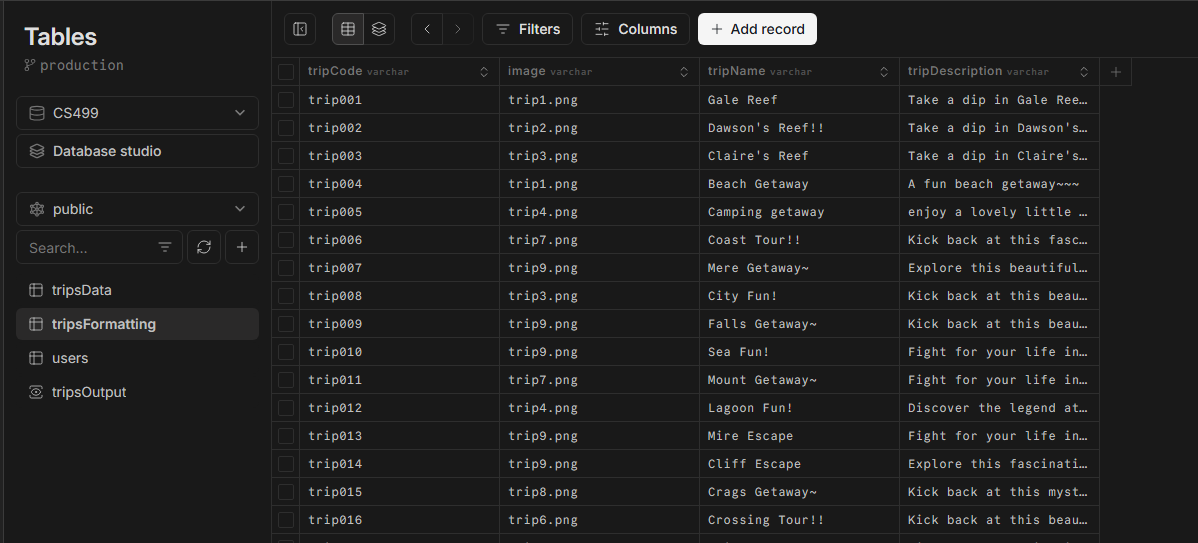
Tanner Reichard

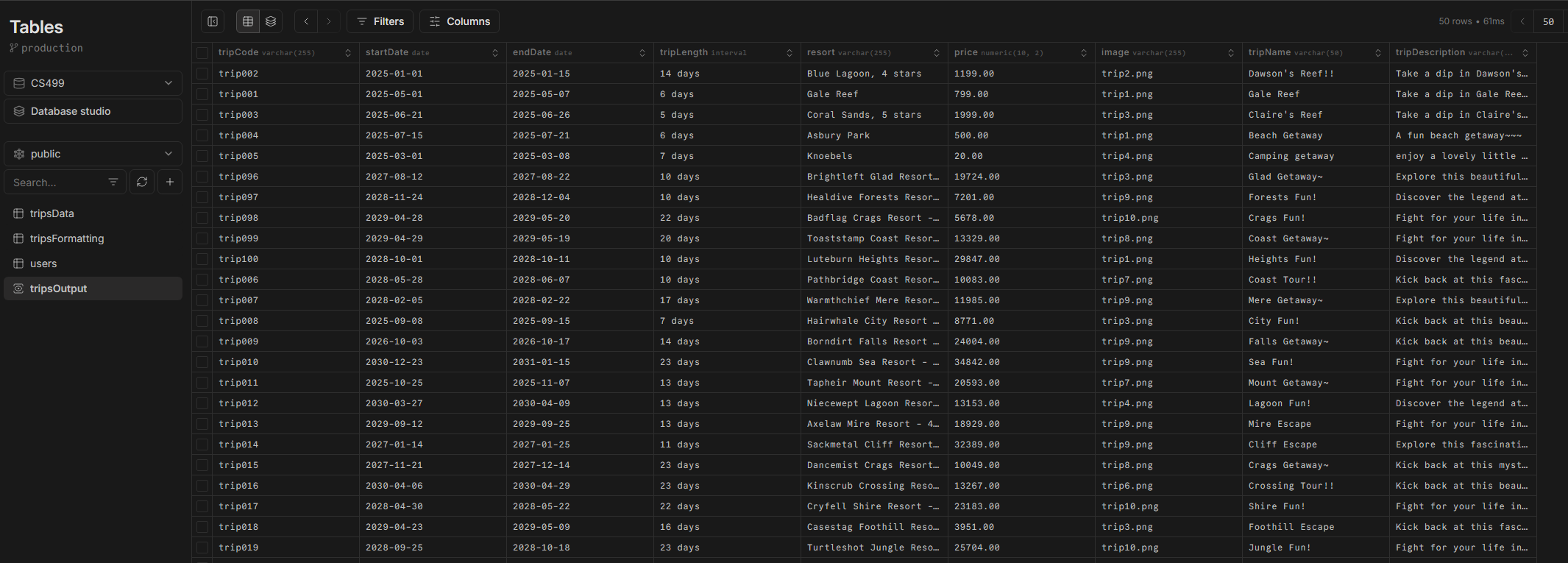
Tanner.reichard@snhu.edu

Southern New Hampshire University

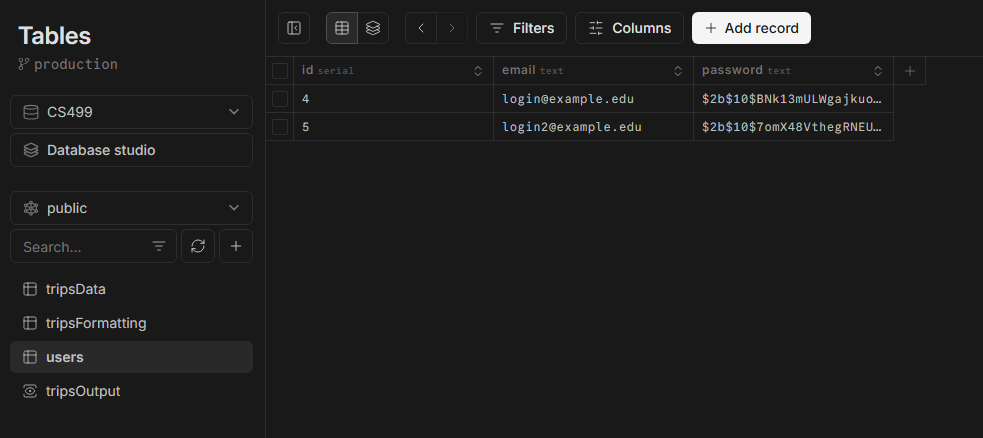
For this milestone, I’ve used a postgreSQL database as the host for the data, rather than the MongoDB database collections in my original artifact. I’ve separated the trip information into two tables for organizational purposes, as well as given them better data types for data analysis and sorting purposes. I also created a view that joins them for ease of access within the application, and included useful fields such as the length column which automatically calculates the length between the start and end date of the trip.







I also am storing the user log in information in this database, with their id, email, and password (encrypted for security best practices) for the application to reference when logging users in.



The sqlConn database connection variable in the tripsController.js file in the backend contains the database calls, and uses prepared statements to avoid potential SQL injections by default with how neon is set up. The tripsRoutes.js file also in the backend shows the functions for editting and creating trips are utilizing the verifyToken function to prevent them from being used if the user isn’t logged in.

I’ve included this as part of my enhancements because it handily shows an equal but better formatted and secured version of database storage than in my original artifact. Setting up the database with secured calls to get information from it shows my ability to 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 through the setup of this database along with its security features, as well as 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 by using modern databases with built in security, as well as applying a standard security layer to keep non-admins from modifying or adding trip data should the site go live.

Overall this enhancement was doable but some extenuating circumstances made it quite difficult to finish quickly. Using bcrypt to store encrypted passwords and compare them when the user is logging in was deceptively easy, and if I had more time I would’ve liked to delve further into what the standard is like in the industry today both for smaller scale apps an the behemoths. My biggest challenge was the time crunch, I usually do have some things come up in my classes but this final class the universe seems hell-bent on making sure I can’t finish things quickly and cleanly.