CS485 Final Project

Alexander Schneider

05-09-2020

Parking Ticket System

Requirements

3 different kinds of user Admin(head officer/dept heads), enforcement officer, citizen.

Keep general information about citizens including car registration, address, outstanding and past tickets.

Keep information on car registration

Allow admin to remove department heads, officers.

Allow department heads to waive fees, transfer officers to other departments.

Allow officers to give tickets, view all outstanding tickets that are past a certain due date, view citizens with more than a given number of outstanding tickets.

Allow citizens to view their tickets, clear tickets at the department office, or challenge tickets.

Host a web app for citizens to access and pay tickets remotely, and an intranet webpage for officers to enter tickets

Allow for people to update vehicle registration and to transfer ownership to someone else

Use Cases

An enforcement officer comes back from a shift and enters the data from the tickets that they have given into the database.(completed)

An enforcement officer can view all unpaid tickets sorted by oldest and can receive contact information to reach the individual(partially completed, can view unpaid tickets)

A citizen walks into the office or goes onto the online portal and enters the ticket number of the ticket they want to pay off or requests an administrator to review the ticket if they feel that it is unjust.(partially completed, can walk into office, no web portal)

A citizen walks in and can update their owned vehicles and transfer ownership to someone else(not completed)

An enforcement officer requests a list of tickets that are past due, or a list of people with over a certain number of tickets with the persons contact information attached, for them to contact.(not completed)

An administrator can request to view all tickets that are challenged by citizens and review the data and choose whether to invalidate the ticket or to uphold it. (completed)

An administrator transfers officers between departments, hires new officers and gives them their id and password, and view contact list for the department they supervise. (completed)

The use cases with interaction with the program beyond the command line were not completed as I ran out of time. If I had more time, I feel the program is in a good enough state that it is at a good point to start working on a proper user interface. Initially I wanted to keep a fully updateable vehicle registration database, but after considering that this is likely just a local ticket cop system, I deemed that the full database with the capability of updating vehicle reference to be out of scope. Instead I decided to store some license plate numbers as a back reference. Also a lot of the outputs are not in the best format and how I search the database needs to be optimized

Schema/Embedding

Schema is in a separate text file

I chose not to embed documents within any of my collections, instead opting to imbed lists of ids, and in the case of admin, a list of arrays of contact information. This was a great use for mongodb as it allowed me to find an embedded list of tickets, embedded list of contact information copied from person to officer and, contact information of an entire department and then iterate those lists accordingly for the information needed, but it would have been much better if I had decided to embed full ticket information inside the person class instead of the id, as it probably would have been a lot better use of the tools available to me. I actually considered redesigning the entire schema to embed full ticket information into the person collection and full officer information into the administrator’s department array and other embedded things that would have made interacting with the database easier throughout the program. I feel that the way that I used the embedded lists that I do have in the schema was a good use of mongodb but I feel that throughout I underutilized the options available to me.

Keys

The person collection’s key is a mongodb generated objected, as the person is rarely searched for, it does not need to be a natural key.

The ticket id is a string that is based off of the idea that each officer gets a book of tickets that they use to fill out tickets they are giving, therefore the key is the officer id combined with the number of tickets given by the officer so far (ex. Tk01+089).

The vehicles id is a string that is based off a license plate(ex. A12345).

The officer id is a string assigned by the admin at the time that the officer is hired.

The admin id is a duplicate of the officer id, kept in a separate collection, as admin is a superset of officer, and has the same associated password.

Programming language

At the start of the semester I only had a rudimentary knowledge of python and how it worked, and I decided that if possible I would use this class as a way to gain a better understanding of the language, the midterm project and the assignments throughout the course have given me what I would say is a pretty good working knowledge of the language, and especially how to interact with databases with python, so it seemed like the natural choice over java for this project.

Groupwork

My teammate sucked 0/10 would not work with again.

Other/what I would have done differently

If I had to do this all again I would have made a better effort of understanding the functions of mongodb ahead of time, and I would have embedded more documents into other collections. A lot of the ways I decided to do things in the design of the database were done because I did not fully understand the way I could interact with the data, and still kind of taking an SQL approach to the database rather than a noSQL approach.