Why the homework?

Welcome! Here at Guaranteed Rate we work with the best of the best. That's why for engineering candidates we want to see a sample of their code. It also helps to have a more productive conversation during the on-site interview. Thank you for putting your time and energy into this!

The homework

This assignment is designed to be completed on your own time. You may use any resources you need to complete it, including emailing questions to me. However, please be sure that you've googled and read docs thoroughly prior to asking questions.

The work is due the night before your on-site interview. We also won't complain if it's in a bit early.

Rules and Guidelines

- We will be looking at the readability and cleanliness of your code.
- Provide unit tests. The cleanliness and readability of tests is just as important as your production code.
- Please provide 80% test coverage for your code.
- Think simple. Readability and modularity are better than being clever.
- Make a github repository and commit your work in small cohesive chunks.

Step 1 - Build a system to parse and sort a set of records

Create a command line app that takes as input a file with a set of records in one of three formats described below, and outputs (to the screen) the set of records sorted in one of three ways.

Input

A record consists of the following 5 fields: last name, first name, gender, date of birth and favorite color. The input is 3 files, each containing records stored in a different format. You may generate these files yourself, and you can make certain assumptions if it makes solving your problem easier.

- The pipe-delimited file lists each record as follows:
  LastName | FirstName | Gender | FavoriteColor | DateOfBirth
- The comma-delimited file looks like this:
  LastName, FirstName, Gender, FavoriteColor, DateOfBirth
- The space-delimited file looks like this:
  LastName FirstName Gender FavoriteColor DateOfBirth

You may assume that the delimiters (commas, pipes and spaces) do not appear anywhere in the data values themselves. Write a program in a language of your choice to read in records from these files and combine them into a single set of records.

Output

Create and display 3 different views of the data you read in:

- Output 1 – sorted by gender (females before males) then by last name ascending.
- Output 2 – sorted by birth date, ascending.
- Output 3 – sorted by last name, descending.

Display dates in the format M/D/YYYY.

Step 2 - Build a REST API to access your system

Tests for this section are required as well.

Within the same code base, build a standalone REST API with the following endpoints:

- POST /records - Post a single data line in any of the 3 formats supported by your existing code
- GET /records/gender - returns records sorted by gender
- GET /records/birthdate - returns records sorted by birthdate
- GET /records/name - returns records sorted by name

It's your choice how you render the output from these endpoints as long as it well structured data. These endpoints should return JSON.

To keep it simple, don't worry about using a persistent datastore.

Step 3 - Review and Refactor

When you are invited to interview in person, we will sit down with you and offer feedback and guidance in refactoring your code. We would like to see you pick up these skills quickly and apply them during the interview to improve the structure of your solution.

Please checkin your project files or bring your own laptop for this part.