Twine Technical Project - Full-Stack Engineer

Your goal is to build and deploy a web application that loads, processes, and returns data related to employees and jobs. By design this project is somewhat open-ended: feel free to make appropriate choices on structure, data storage, and data format given your time limit.

Requirements

The application should:

- Load employee and job data from CSV files, as defined below in **Data**
- Calculate a score for each employee-job pair, as specified below in **Scoring**
- Return JSON data responses to several URL endpoints, as listed below in **Requests**

Data

Two CSV files contain data on **employees** and **jobs**, located here:

https://s3.amazonaws.com/twine-labs-misc-data/recruiting/employees.csv

https://s3.amazonaws.com/twine-labs-misc-data/recruiting/jobs.csv

Here's a summary of this data:

```
 column       | description
--------------|--------------------------------------
 id           | employee ID (unique, not null)
 name_first   |
 name_last    |
 salary       |
 department   | current department of employee/manager
 high_risk    | whether employee is at risk of leaving
 time_in_role | number of years employee has spent in role
 location     | current employee location

 column       | description
--------------|--------------------------------------
 id           | job ID (unique, not null)
 title        |
 department   |
 location     | primary office location
 salary_min   | minimum of salary range
 salary_max   | maximum of salary range
 priority     | whether job is a priority role to fill
```

Scoring

The score between each employee and job is a numeric value between 0 and 1, defined as the product of the following 3 numbers:

1. [salary]: This value is
    - **1.0** if *employee:salary* is halfway between *job:salary_min* and *job:salary_max*
    - **0.0** if *employee:salary* >= *job:salary_max* or <= *job:salary_min*
    - scales linearly from **1.0** to **0.0** in between

    For example, if *salary_min* was $40k and *salary_max* was $50k, a *salary* of $42.5k would correspond to a value of **0.5**, and a *salary* of $58k to a value of **0.4**
2. [at risk]: This value is **1.0** if *employee:high_risk* is True, otherwise it is **0.8**.
3. [priority]: This value is **1.0** if *job:priority* is True, otherwise it is **0.8**.

If there are data constraints or edge cases not addressed in the above definitions, please use your discretion to resolve them.

Requests

Your web application server should provide API endpoints to support the requests below. Please note you will need to define the URLS and parameters for these endpoints as well as the structure of the output data.

1. Return the highest-scoring ***n*** jobs for employee with ID ***employee_id***
2. Return the highest-scoring ***n*** employees for job with ID ***job_id***
3. Return the highest-scoring ***n*** overall employee-job pairs

For the requests above, the payload returned should be valid JSON format (GeoJSON ideally, but don't spend time on this). In addition, please return data in CSV format for the final request:

1. Return the full data set of employee-job score pairs

Criteria/Guidelines

- Feel free to use any languages or frameworks you like, and any online resources or documentation. However, please do not consult with anyone and certainly do not use anyone else's code.
- Build solutions that are correct first before optimizing. However, think through your design choices and trade-offs in performance/maintainability.
- Aim to produce high-quality, extensible, legible code.
- If something is ambiguous or unclear in the requirements, please use your discretion and make reasonable assumptions to resolve them - and document these accordingly.

Submission Instructions

Please reply to the project email with a link to your code (hosted anywhere, consider Github).

The code should include a README that contains:

- Instructions on how to build and run your application.
- A brief list (2-5 bullet points) of other features or improvements you'd want to build if you had more time.
- Any other instructions or details you want us to consider.

Good luck!