Advisor Dashboard - Take Home Assignment

Introduction

Hello! We're glad you're interested in joining the team at Compound! In this step of the interview process you will be completing a take home assessment that will be the basis for future technical interviews. Please use this exercise to demonstrate your talent, creativity, and problem solving ability. This exercise is inspired by problems we encounter daily and we hope that it is a enjoyable challenge to solve.

There is no time limit for the exercise, but we expect that you should be able to complete it within a few hours. While a working solution is expected, we understand that this will not be production ready code. Please make sure to both showcase your abilities and capture the additional steps you would take to make the code production ready.

We ask that you do not share your solution publicly. Please create a private repository and add the **compound-interview-bot** as a collaborator of the project.

What do I need to submit?

- Build an application as described in the next section. You can use any language and frameworks you choose.
    - It is not required, but please feel free to try languages and tools that are common at Compound listed in the job description. This will give you a feel for our environment as you complete the exercise.

- We must be able to run your app and see a response containing answers to the questions below. Provide any documentation necessary to accomplish this as part of the repository you submit.
- Please assume the reviewer has not executed an application in your language/framework before when developing your documentation.

Business background

Compound’s business runs on tools to enable financial advisors provide a magical experience for their clients. Financial advisors can be part of an organizational group which we refer to as a firm. In this scenario you will be aggregating client account information for the advisory firm. This will provide insights into the underlying account investments for the advisor and the firm.

Terminology

- Firm - Company that the Advisors work for. You can assume the Company here is Compound Planning and the Advisors are the ones that work with us.
- Client - Someone who receives financial guidance and advice from a financial advisor.
- Advisor - A person who is employed to provide financial services or guidance to clients
- Holding - The contents of an investment portfolio held by an individual or an entity.
- Security - A tradable financial asset such as equities or fixed income instruments
- Account - Investment accounts can hold stocks, bonds, funds and other securities, as well as cash.
- Custodian - A financial institution that holds customers' securities for safekeeping

What are you building?

You are working to build features that power a Firm's Advisor Dashboard. The advisor dashboard will help us gain valuable insights across our advisors. Listed are 2 options for what you could submit. Please only **choose 1 option**.

Option A - Backend emphasis

- Build a data processing job to parse Advisor, Account, and Security information. The goal is to analyze the data sets so it can be exposed via an API for a frontend application to consume. Assume 3 different data streams containing Advisor, Account, and Security data listed below.
    - Answer the following questions as a result of processing the data from the script:
        - **What is the total value across all of the accounts that the advisors are managing?**
        - **Identify the top securities held to get an idea of our risk exposure in the markets.**
        - **For each custodian provide an ordered list of which advisors has the most assets at the custodian.**
- Build an API to interact with the data
    - Setup a simple server and create the endpoints for interacting with the data.
    - Expose the statistics we collected in the first part and also the underlying advisor, account, and security information
    - The data exposed in the API doesn't have to be aligned with how it's collected in the job. Think about designing it so a consumer of the data can work with it effectively.

Option B - Frontend emphasis

- Create Frontend components for the Firm-Advisor Dashboard.

    - Design and implement a table view to see information about the advisors. The view should:
        - Show us the advisors and a summary of the assets that they are managing.
        - Provide the ability to drill in and see information about the individual accounts that they are managing
        - See the holdings that are in the account.
        - Consider the example data below.
    - Add 1 sort or filter feature.
        - You can add the sort/filter wherever makes sense to you based on the implementation from the first part.
        - One possible example could be sort the advisors by name or total assets that they are managing.
- API Design
    - We want to retrieve the data needed to power the dashboard features you are building. Please retrieve the data via API calls to a backend server
        - When building the backend server, the API endpoints can have mock data in them.
        - Focus on how you would design the interactions with a backend service and the interfaces are needed.

Example Data:

This data does not need to be used as is. This is the core shape of the data, but you may change it, normalize it, or transform it as you build out the frontend/backend and API layer accordingly.

```
// Advisor:
{
  "id": "4",
  "name": "Randall",
  "custodians": [
```

```
    { "name": "Schwab", "repId": "1271" },
    { "name": "Fidelity", "repId": "8996" }
  ]
}

// Account:
{
  "name": "Bradley Green - 401k",
  "number": "21889645",
  "repId": "9883",
  "holdings": [
    { "ticker": "HEMCX", "units": 77, "unitPrice": 398.63 }
  ],
  "custodian": "Schwab"
}

// Security:
{
  "id": "2e5012db-3a39-415d-93b4-8b1e3b453c6c",
  "ticker": "ICKAX",
  "name": "Delaware Ivy Crossover Credit Fund Class A",
  "dateAdded": "2001-06-07T11:12:56.205Z"
}
```

Expectations of the Exercise:

- Requirements:
  - [Specific to backend option] Have a mechanism to run your script to process the data.
    - We will test it against separate data files to check the results.
  - Provide instructions for running the app/service.
  - Add Documentation for:
    - Any assumptions you make.

        - Document any todos, optimizations, or next steps you would take in order to get this feature ready to launch.
    - Try to put your best foot forward with the submission. Include polish in your submission that's comparable to what you would put into code review.
- Submit your solution!
    - Invite the compound-interview-bot (https://github.com/compound-interview-bot) as a collaborator.

FAQ:

Private repository submission details.

- Make sure your Github repository is private.
- Add compound-interview-bot as a collaborator
    - Here's a link to the account: https://github.com/compound-interview-bot
    - You can search how to add Github collaborators, but here's a link if helpful
    - https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-access-to-your-personal-repositories/inviting-collaborators-to-a-personal-repository

How will this exercise be evaluated?

An engineer will review the code you submit. At a minimum they must be able to run the script and the service must provide the expected results.

You should provide any necessary documentation within the repository. While your solution does not need to be fully production ready, you are being evaluated so put your best foot forward.

I have questions about the problem statement

For any requirements not specified via an example, use your best judgment to determine the expected result. Document any assumptions you make.

How long do I have to complete the exercise?

There is no time limit for the exercise. Out of respect for your time, we designed this exercise with the intent that it should take a few hours. But, please take as much time as you need to complete the work.