Coding Design Challenge - Vendors and Clients

At Abett we have a growing list of customers (also called clients). Each customer wishes to send data to multiple vendors for analysis. Each customer also needs to receive source data from multiple vendors (their "providers"). The list of vendors is also growing quickly, as are the connections between clients and vendors.

To keep track of this, we need a system to track Vendors and Clients. Each Vendor has the following properties:

- Name
- Direction: Input, Output, Both
    - this represents the direction from the client's perspective, e.g., an Input vendor provides inputs to clients
- Schedule: Daily, Weekly, Monthly
    - if the vendor does both inputs and outputs, there may be one schedule for inputs and a different one for outputs

Each Client has the following properties:

- Name
- Count of Employees

Each Client will work with one or more Vendors. Each Vendor will work with one or more Clients.

The system needs to support:

- CRUD operations on vendors
- CRUD operations on clients
- Add or remove a relationship between a vendor and a client
- For a given Client, list the vendors for each of Inputs and Outputs, ordered by schedule frequency: daily first, then weekly, then monthly
- For a given Vendor, list the delivery schedule(s) to/from the Clients to which the vendor is connected
- For a given Vendor, list the number of unique employees their data covers (assuming no employees work for multiple clients)

Design a system to support these requirements. Implement as much of the system as you can, but pseudo-code and TODO items well commented and documented can be provided in places where implementation is not fully complete. The goal of this exercise is to express how you would design and implement an efficient data model, a well-structured set of REST endpoints, and an extensible system suitable for future enhancement.

You can use whatever coding language and whatever toolset you prefer. The code that does work should be able to be run locally, from a command line, with whatever supporting tools are required.

If you have the time an inclination, you can choose to build a simple UI as a front-end. At a minimum you should provide REST endpoints that can be accessed via curl, Postman, Insomnia, etc.

Create your submission in your personal github account and send a link. Or alternately you can ZIP up the file(s) and email them with instructions for how to run.