**Task Overview**
This task involves designing and implementing a system for a sales-oriented organization that pitches various services and products to certain brands. The organization manages a repository of brands, their interests, and other sales-related information. The system needs to accommodate different employee roles and brand details.

**Following are the nature of employees:**
They can have one or more of the following roles - PO, BO, TO

- ADMIN - add and update brands and users
- PO (project owner) - leads and manages a sales campaign
- BO (salesperson/ brand owner) - talks to brand and pitches offers
- TO (team owner/ manager) - oversees the work of BOs

**Following is the nature of brands:**

- They have a few details like:
  brand name, revenue, deal closed value, contact_person_name, contact_person_phone, contact_person_email
- They are managed by one or more BOs

**Requirements:**

- Design the database schema to accommodate the following entities:
    - Users (with roles: ADMIN, PO, BO, TO) - name, department, phone number, email
    - Brands (with details: brand name, revenue, deal closed value, contact person name, contact person phone, contact person email)
    - Teams (automatically created when a user with TO role is created)
- Implement a hierarchical relationship to manage users within the organization.
- The hierarchy involves **TOs or TO+POs** managing **BOs**, and other users report to their direct team owners.
- Ensure scalability and query/API performance by:
    - Using efficient database indexing: For example, indexing the user_id column in the teams table to improve query performance.
    - Optimizing SQL queries: For example, using JOIN instead of subqueries to improve query performance.
    - Using connection pooling: For example, using a connection pool to manage multiple database connections and improve performance.

**API Requirements**
1. **User and Brand Management** (ADMIN Only)

- Add and update users:
    - Create/Edit user.
    - Assign roles to users
    - Select a TO for the user
    - A user can have multiple roles

        - A user must have at least one role
        - PO role can only be selected in conjunction with the TO role
    - Automatically create a team when a user with TO role is created
    - Add and update brands:
        - Assign a brand to a BO
    - List brands, their details, and their brand owners
    - Prevent cycles/loops in the hierarchy
    - List users:
        - List all TOs above the hierarchy of a user in top-down fashion
        - Example, If 3 level of employees, getting 3rd level employee detail should show all parent TOs also.

2. **Brand Management** (BO)
    - People with a BO role can manage their brand details and also manage contact persons for the brand.
    - Add or update brand details:
        - Contact person name
        - Contact person phone
        - Contact person email
    - Allow multiple contact persons to exist
    - List and view details of the brands they own
3. **Team Management** (TO)
    - List and view all users in their team, including users down the entire hierarchy
4. **Brand Details**
    - If BO, he should get access to all the details about the brand.
    - If Just TO role, he should get access to the brand but contact person details should not be available. TO will get access to brand info only if the brand is owned by the team members down the hierarchy.
    - If the user has a TO and PO role, the same conditions as just TO but since PO role is there he can also see contact person details.
5. **All Users**
    - List and view their teammates and their contact details
    - Search and view contact details of any user in the organization

Consider the points below while implementing the feature.

- Correctness and completeness of the implementation
- Performance and scalability of the API endpoints
- Code quality, readability, and maintainability
- Well-documented API endpoints
- Well-designed and optimized database schema