Laravel Developer Hiring Task

Multi-Tenant SaaS: User with Multiple Companies

Objective

Build a minimal backend in Laravel where a **registered user can create, manage, and switch between multiple companies** under their profile. All subsequent data and actions should be scoped to the “current” company.

Requirements

1. Authentication

- User registration, login, logout (use Laravel Breeze or Fortify).

2. Multi-Company (Multi-Tenant) Logic

- Each user can create, list, update, and delete multiple companies under their account.
- Each company must have a profile: name, address, and industry (minimum fields).
- A user can only access/modify their own companies.

3. Company Switching

- User can **switch the “active” company** via an API or endpoint.
- All subsequent data and actions (future modules, e.g., invoices, projects) should be **scoped to the current active company**.

4. API Endpoints

- Auth: Register, Login, Logout.
- Companies: Create, List, Update, Delete.
- Set/Switch Active Company.

5. Data Scoping

- Enforce data isolation so that:
  - A user can only access/modify their own companies.
  - All operations are performed in the context of the current active company.

6. Database

- Use MySQL for the database.
- Clean schema design showing users, companies, and their relationships.

- Use Eloquent relationships for ORM.

7. Validation & Error Handling

- All actions must have proper validation and error messages.

Database Structure (MySQL Example)

users

- id (PK)
- name
- email
- password

companies

- id (PK)
- user_id (FK to users)
- name
- address
- industry
- created_at
- updated_at
- deleted_at (nullable for soft deletes)

user_active_companies (optional helper table for tracking active company per user)

- id (PK)
- user_id (FK to users)
- company_id (FK to companies)

Alternatively, store active company_id in the users table if only one active allowed per user.

Submission

- Public GitHub repository.
- Concise `README.md` with:
  - Setup instructions.
  - API endpoints and example requests.
  - Explanation of multi-tenant logic and data scoping.
