PHASE – 1 FUNCTIONAL REQUIREMENTS DONE:

1. Login Screen:

- Implemented secure login using email and password authentication.
- Integrated authentication with the PostgreSQL database hosted on Hetzner.
- Added “Forgot Password” functionality to verify user emails against the database.
- Developed a Reset Password screen available only for Admin users.
- Recruiters and Clients attempting to reset passwords are shown an alert prompting them to contact the Admin.

2. Admin Dashboard:

a. Role Based Access:

- Implemented dynamic dashboard rendering based on user roles (Admin, Recruiter, Client).
- Admins can access all dashboard modules, while Recruiters and Clients have limited visibility.
- Integrated functionality to display all logged-in users, their roles, and login timestamps on the dashboard.

b. Users Management:

- Added functionality to create new users with role selection (Admin, Recruiter, Client).
- Integrated APIs to fetch all users from the database and display them on the Users screen.
- Enabled user deletion and update capabilities directly from the interface.
- Implemented search and filter functionalities by user name or email.
- Added alert notifications for successful operations (add, update, delete).

c. **Recruiters Management**:

- Developed Recruiters screen to fetch all recruiters from the database.
- Enabled update and delete functionalities for recruiter records.
- Alerts displayed for every action respectively.
- Filtering emabled using email and name.

3. User Interface (UI)

- Designed and implemented a consistent and responsive UI for all screens.
- Ensured uniform styling across login, dashboard, and management modules.
- Implemented company branding by adding the logo to the web app interface and browser tab.

4. APIs

- Developed and integrated backend APIs for all database operations.
- Successfully connected and consumed these APIs in the frontend application for seamless data flow.
- Each API was individually tested and validated on Postman before frontend integration, ensuring correct HTTP methods, response codes and data structures.

5. Database Setup (Hetzner)

- Configured a Hetzner VPS instance for hosting the PostgreSQL database
- Ensured secure access by allowing authorized IP connections only.
- Deployed Prisma ORM schema migrations to initialize and manage the database structure.
- Verified connection stability between the backend server and Hetzner-hosted database.

6. Database Creation and Management

- Created the main database named “hrbs” with secure credentials.
- Defined and structured key tables required for application functionality.
- Used pgAdmin4 as the primary database management tool to:

1) Visually monitor database schema and table structures.

2) Run SQL queries for initial data testing and validation.

3) Verify CRUD operations performed via APIs (cross-checking updates, inserts, and deletes).

4) Inspect logs and relationships between the User, Recruiter, Client, and LoginActivity tables.

a. Tables Created:

- **User:** Stores login credentials, user roles, and personal details.
- **Recruiter:** Stores recruiter-specific details and linked user information.
- **Client:** Stores client-specific information if separate from user records.
- **LoginActivity:** Tracks user login and logout times for monitoring and audit purposes.