**🚀 Build CareerNest – A Complete Job Portal (PHP + MySQL)**

We are developing **CareerNest**, a fully functional job portal website inspired by platforms like JobStreet. The platform will be built using:

* **PHP** (Backend)
* **MySQL** (Database)
* **HTML, CSS, Bootstrap** (Frontend)

The system must support three distinct user roles: **Applicants**, **Employers (Companies)**, and **Admin**, each with dedicated dashboards, access controls, and feature sets.

**🔧 Core Features & Requirements**

**📁 1. Database Design**

Create a MySQL database named careernest\_db with the following tables:

| **Table Name** | **Purpose** |
| --- | --- |
| users | Individual job applicants' profiles and credentials |
| companies | Registered employer accounts and company data |
| job\_listings | Job posts created by companies |
| job\_applications | Applications submitted by users |
| schedules | Interview schedules proposed by companies |
| notifications | System messages to users or companies |

**Static Admin Account** (pre-inserted into users table):

* **Email:** adminCareerNest@gmail.com
* **Password:** C@reerNest\_Group8 (securely hashed using password\_hash())

**👥 2. User Roles & Access**

| **Role** | **Capabilities** |
| --- | --- |
| **Users** | Register/login, view/edit profile, search/view job listings, apply to jobs, view applications and system notifications |
| **Companies** | Register/login, post/edit jobs, view applicants and their profiles, propose interview schedules, view notifications, **close job listings** |
| **Admin** | View and manage all users and companies, approve/edit job listings and schedules, oversee system data, **receive notifications when companies close job listings** |

**🔐 3. Authentication & Session Management**

Implement a secure login/registration system with session management (session.php). Role-based redirection upon login:

* **Users** → user\_dashboard.php
* **Companies** → company\_dashboard.php
* **Admin** → admin\_dashboard.php

Use password\_hash() and password\_verify() for password security.

**🏠 4. Homepage (index.php)**

* Job search functionality with the following filters:
  + Keyword
  + Location
  + Work type
  + Work arrangement (remote, onsite, hybrid)
  + Salary range
  + Date posted
* Results dynamically pulled from the job\_listings table.
* Navigation links:
  + **Sign In** → login.php
  + **Register (Applicant)** → register\_individual.php
  + **Employer Site** → register\_company.php

**Note:** Even after applying to jobs, users should still be able to search and view additional job listings via the homepage or user dashboard.

**🧭 5. Dashboards by Role**

**🧑‍💼 User Dashboard**

* View/edit profile
* View jobs the user has applied to
* Search and view more job listings
* View notifications (e.g., interview updates, application responses)

**🏢 Company Dashboard**

* Post and manage job listings
* View all job applicants
* View complete profiles of applicants (e.g., resume, skills, experience)
* Propose interview schedules
* Receive system notifications (e.g., admin responses)
* **Edit and close job listings**:
  + Once a job is marked as "closed", it becomes invisible to users.
  + Admin receives a notification but cannot override or reopen the listing unless granted special permission.

**🛠 Admin Dashboard**

* View summary cards:
  + Total registered users
  + Total registered companies
* Access:
  + users\_list.php
  + companies\_list.php
* Review, edit, and approve:
  + Job postings
  + Interview schedules
* **Receive notifications when companies close job listings**

**🔁 6. Interview Scheduling Approval Logic**

* Companies propose interview schedules for selected applicants.
* Admin reviews schedules:
  + If any changes are made, the company is notified with a reason.
  + The company must confirm the changes before proceeding.
* Once confirmed:
  + The schedule is finalized.
  + The user (applicant) receives a final notification.
* Admin **cannot notify users directly** after making schedule changes — company confirmation is required first.

**💡 Tech Stack & Development Guidelines**

* **Backend:** PHP (structured using procedural or object-oriented design)
* **Database:** MySQL (with normalized tables and indexed columns)
* **Frontend:** HTML5, CSS3, Bootstrap (for responsive UI)

**Security Best Practices:**

* Passwords hashed using password\_hash() and verified with password\_verify()
* CSRF protection
* All user input must be validated and sanitized

**UI/UX Guidelines:**

* Clean, modern, and mobile-responsive layout
* Bootstrap for consistent styling
* User-friendly navigation and dashboard layouts for all roles