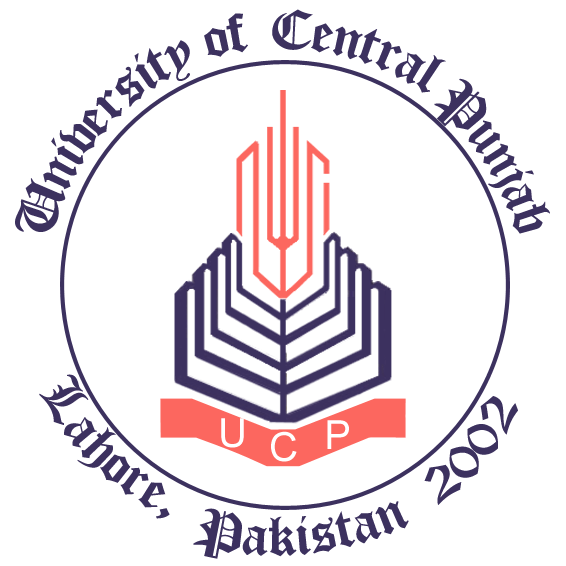
BSCS FINAL PROJECT

<Design and Test Specification>

TalentHub Pro



Project Advisor

**Mr. Ali Abbas**

Presented by:

**Group ID: S25BS020**

Student Reg# Student Name

L1F21BSCS0400 Ahmad Zahid

L1F21BSCS0364 Muhammad Zohaib

L1F21BSCS0366 Ammar Shabbir

**Faculty of Information Technology & Computer Science**

**University of Central Punjab**

Design and Test Specification

SDP Phase III

TalentHub Pro

Advisor: Ali Abbas

Team # S25BS020

|  |  |
| --- | --- |
| Member Name | Primary Responsibility |
| Ahmad Zahid | Lead Developer and backend specialist |
| Muhammad Zohaib | Frontend Developer and UI/UX Designer |
| Ammar Shabbir | Quality Assurance And project coordinator |

Table of Contents

Table of Contents i

Revision History ii

Previous Phases Feedback ii

Abstract 3

1. Introduction 3

1.1 Product 3

1.2 Background 3

1.3 Objective(s)/Aim(s)/Target(s) 4

1.4 Scope 4

1.5 Business Goals 4

1.6 Document Conventions 5

1.7 Miscellaneous 6

2. Technical Architecture 7

2.1 Application and Data Architecture 10

2.2 Component Interactions and Collaborations 21

2.3 Design Reuse and Design Patterns 25

2.4 Technology Architecture 25

2.4.1 Hosting: 25

2.4.2 Connectivity: 26

2.4.3 Modes of Operation: 26

2.4.4 Security Measures 26

2.5 Architecture Evaluation 26

3. Detailed/Component Design 27

3.1 Component-Component Interface 33

3.2 Component-External Entities Interface 34

3.3 Component-Human Interface 35

4. Screenshots/Prototype 37

4.1 Workflow 37

4.2 Screens: 38

4.3 Additional Information 51

5. Other Design Details 52

5.1 Role-Based Access Control 52

5.2 Security Measures 52

5.3 Scalability Features 52

5.4 Future Enhancements 53

6. Test Specification and Results 53

6.1 Test Case Specification: 54

6.2 Summary of Test Results: 59

7. Revised Project Plan 60

8. References 61

Appendix A: Glossary 1

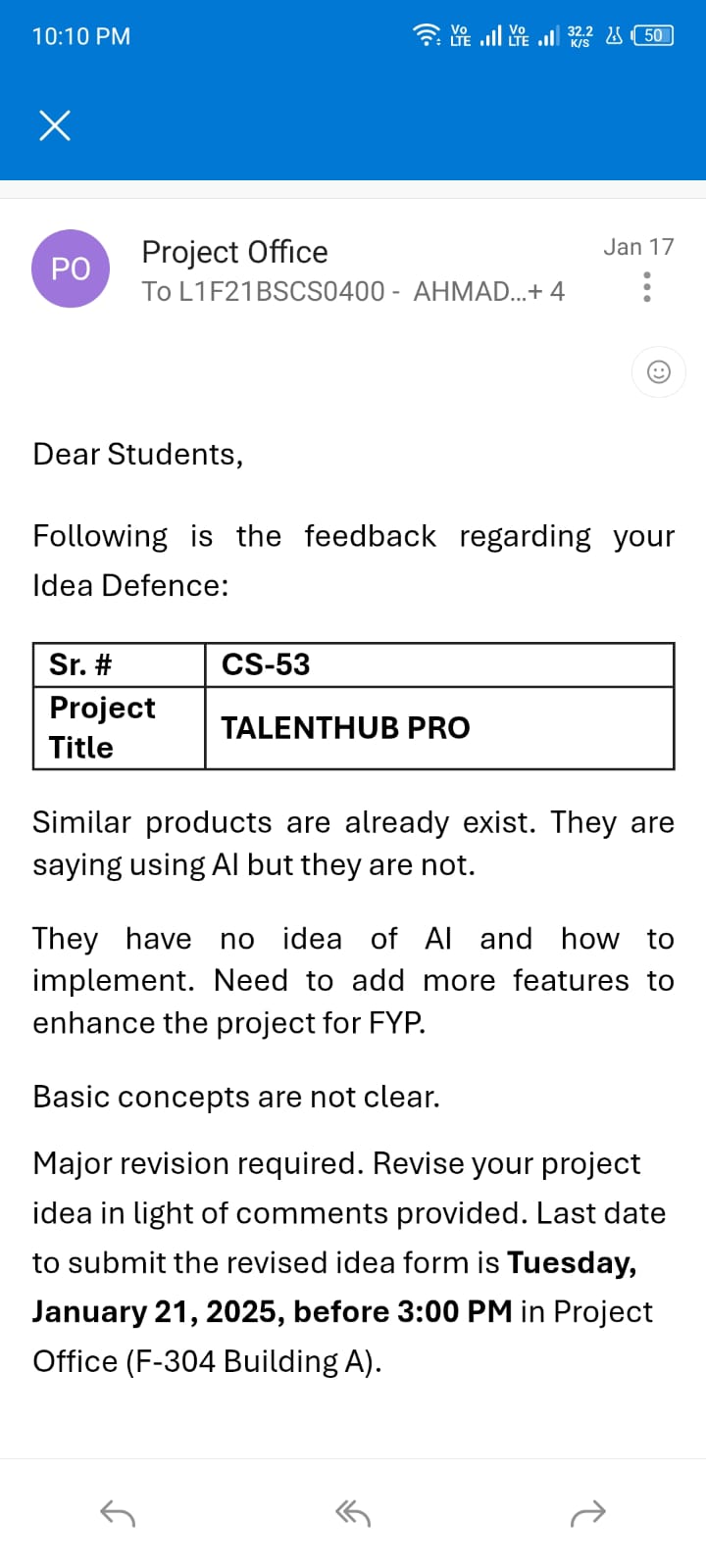
Appendix B: IV & V Report 2

Revision History

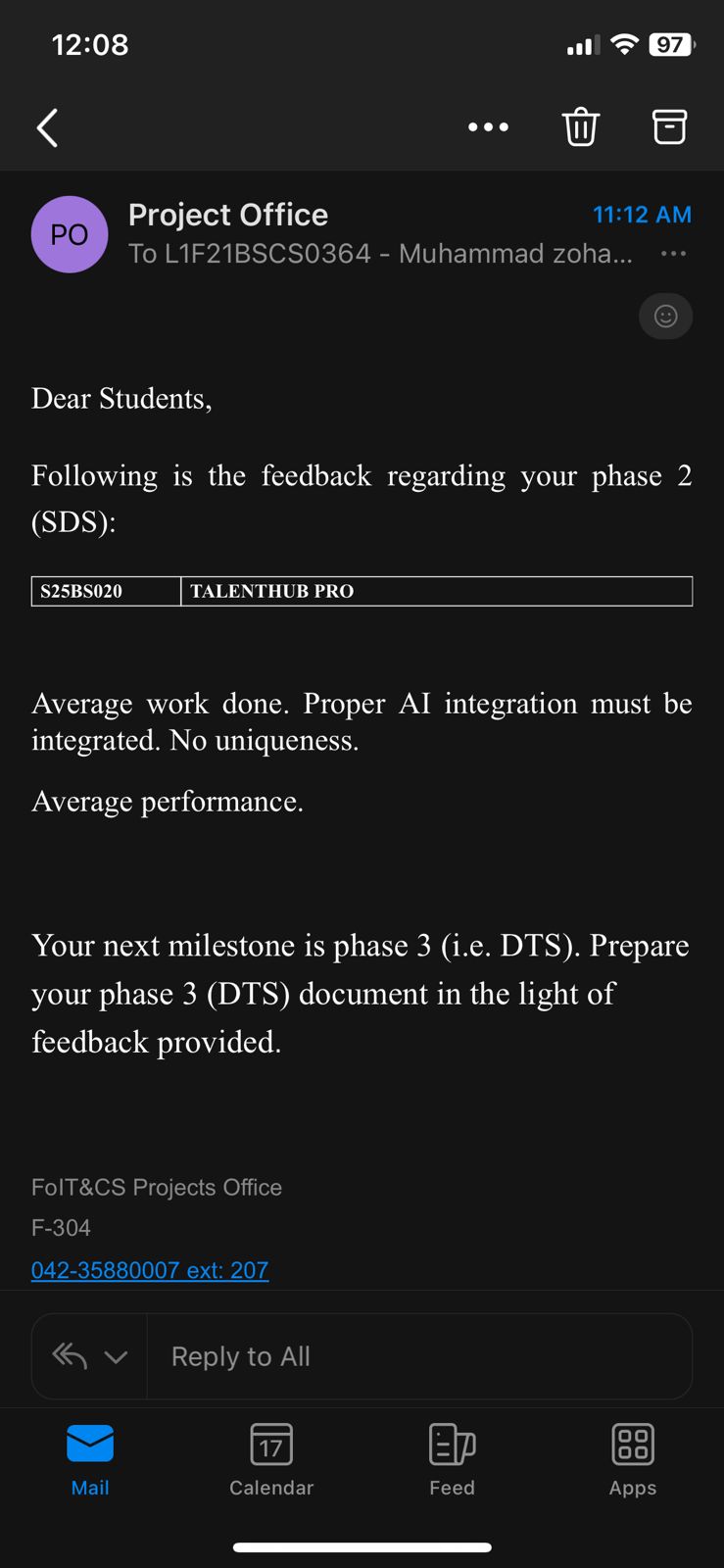
|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

Previous Phases Feedback

**Idea Defence Feedback (Screenshot)**

****

**Phase 2 (SDS) Feedback (Screenshot):**

****

# Abstract

**TalentHub Pro** is an AI-powered recruitment platform designed to streamline the hiring process by automating key tasks such as candidate filtering, CV generation, and interview preparation. The system intelligently collects and processes candidate data—experience, availability, skills, and salary expectations—to match job seekers with employer requirements in real time.

The platform leverages Artificial Intelligence (AI) and Natural Language Processing (NLP) to provide smart features including a chatbot for interactive communication,auto profile update, personalized skill development suggestions, and dynamic interview question generation based on job descriptions and company profiles.

TalentHub Pro offers a secure, scalable web application aimed at reducing manual workload for recruiters while empowering candidates with better visibility and growth recommendations. By bringing together intelligent automation and user-centric design, it provides a complete solution to modern hiring challenges.

# Introduction

## Product

**TalentHub Pro** is a web-based, AI-powered recruitment platform designed to streamline and modernize the hiring process. It utilizes Artificial Intelligence (AI) and Natural Language Processing (NLP) to automate essential hiring functions such as intelligent candidate filtering, structured CV generation, real-time availability tracking, and personalized skill recommendations. The platform also features an integrated smart chatbot that enhances user interaction and dynamically generates interview questions based on job descriptions and company profiles. TalentHub Pro provides an end-to-end solution for both employers and job seekers, emphasizing efficiency, accuracy, and personalized hiring experiences.

## Background

In today’s fast-paced digital world, the recruitment process often remains manual, time-consuming, and inefficient. Employers frequently encounter challenges in quickly identifying the right candidates, while job seekers struggle to effectively present their skills through conventional CVs and outdated application methods. These inefficiencies result in delayed hiring, mismatches, and lost opportunities. With the increasing demand for smart hiring tools, there is a clear need for platforms that offer automation, accuracy, and user-friendly experiences. TalentHub Pro was developed in response to these needs, aiming to overcome traditional recruitment barriers through intelligent, technology-driven solutions.

## Objective(s)/Aim(s)/Target(s)

The primary objective of **TalentHub Pro** is to create an intelligent, AI-powered recruitment platform that simplifies and improves the hiring process for both employers and job seekers.

**Key objectives include:**

* To develop a user-friendly, web-based platform for recruitment
* To implement **AI-based filtering** for accurate candidate-job matching
* To automate **structured CV generation** based on user input
* To integrate a **chatbot** using NLP for real-time interaction and assistance
* To provide **personalized skill development recommendations** to candidates
* To generate **job-specific interview questions** using AI and job data analysis
* To ensure **data security**, scalability, and performance of the system

These targets aim to deliver a complete, efficient, and modern solution for digital hiring.

## Scope

**TalentHub Pro** will serve two primary user groups: employers seeking qualified candidates and job seekers looking for relevant opportunities.

The system will include the following core features:

* **AI-based candidate filtering** based on job requirements and profile data
* **Automated CV generation** with structured formatting
* **Real-time availability tracking** of candidates
* **Personalized skill development recommendations**
* **AI chatbot** for interactive assistance and query handling
* **NLP-based interview question generation** aligned with job descriptions

The platform will operate within a secure, scalable, and user-friendly web environment. This SRS outlines the system’s functional and non-functional requirements, focusing on delivering a complete, intelligent hiring solution for the modern recruitment ecosystem.

## Business Goals

**TalentHub Pro** is designed to meet several key business and corporate goals in the recruitment and HR technology domain:

1. **Streamline the Recruitment Process**
   * Reduce manual effort by automating candidate filtering, CV generation, and interview preparation.
2. **Improve Hiring Accuracy**
   * Use AI to match candidates with job roles more precisely, reducing mismatches and turnover.
3. **Enhance Candidate and Employer Experience**
   * Provide a user-friendly platform with real-time interaction and personalized recommendations.
4. **Offer Competitive Advantage**
   * Differentiate from existing solutions through integrated AI, NLP, and automation features.
5. **Generate Revenue**
   * Introduce monetization through premium recruiter tools, job posting packages, or subscription models.
6. **Support Scalability and Growth**
   * Design the system architecture to support future expansion, new features, and a larger user base.
7. **Establish Market Presence**
   * Position TalentHub Pro as a reliable and intelligent recruitment platform in the local and regional job markets.

These goals aim to ensure long-term sustainability, business value, and relevance in the evolving recruitment industry.

## Document Conventions

The following standards and formatting conventions have been used throughout this Software Requirements Specification (SRS) document to maintain consistency and clarity:

* **Font Style and Size:**
  + Main text is written in *Times New Roman*, size 12 pt
  + Section headings use bold formatting with sizes 14 pt or 16 pt for hierarchy
* **Section Numbering:**
  + Sections are numbered hierarchically (e.g., 2.1, 2.2, etc.) for easy navigation
* **Text Formatting:**
  + **Bold** text is used to highlight important terms, features, and headings
  + *Italicized* words indicate system elements, user actions, or variables
* **Tables and Lists:**
  + Structured data is presented in tables for readability
  + Bullet points are used to list items clearly
* **Acronyms and Abbreviations:**

All acronyms (e.g., AI, NLP, SRS) are defined upon first use and used consistently thereafter

These conventions help ensure the document is readable, well-organized, and easily understood by all stakeholders involved in the development of TalentHub Pro.

## 1.7 Miscellaneous

To ensure the smooth execution and success of the TalentHub Pro project, the following considerations and support elements are noted:

* **Team Collaboration**: The project is being developed by a well-coordinated team with clearly defined roles, ensuring efficient task distribution and accountability throughout the development lifecycle.
* **Technology Stack**: The platform utilizes modern web technologies including ReactJS, Firebase, and Tailwind CSS, which enable scalable, secure, and responsive application development.
* **User-Centric Design**: Special focus has been placed on designing an intuitive and user-friendly interface to cater to both technical and non-technical users.
* **Testing & Feedback Loop**: Continuous testing and user feedback sessions are planned to identify usability issues and gather suggestions for improvement.
* **Scalability & Future Enhancements**: The system is being designed with scalability in mind, allowing future enhancements such as video interview modules, AI-driven talent analytics, and integration with external job portals.
* **Documentation & Version Control**: Proper documentation is maintained for all modules, and GitHub is used for version control to track progress and ensure code integrity.

These additional factors contribute to the project's robustness and will aid in the successful completion and deployment of TalentHub Pro.

# 

# Technical Architecture

The **TalentHub Pro Recruitment Platform** is a custom-built, full-stack AI-powered system designed to automate and streamline modern hiring processes. It supports intelligent skill matching, automated CV analysis, job posting, candidate tracking, secure file storage, and real-time notifications.  
The architecture follows a layered, service-oriented client–server model with RESTful communication and integrated AI microservices. Both transactional operations (e.g., job applications) and analytical processing (e.g., skill recommendations) are core to system functioning.

 **Architecture Type:** Layered, Client–Server, Microservice-Enhanced

 **System Type:** Custom-Built Recruitment & AI Automation System

 **Processing:** Real-time (API interactions, chatbot responses) and batch-based (resume parsing, report generation)

 **Main Responsibilities:** AI skill matching, job posting, CV evaluation, candidate tracking, chat-based support, notifications, recruiter management, analytics

 **Application Components:**

* **Candidate Web Dashboard (React.js, hosted on Vercel)**
* **Recruiter Dashboard (React.js, Vercel)**
* **Admin Panel (React.js, Vercel)**
* **TalentHub Flutter Mobile App (Flutter)**
* **Backend API Server (Node.js + Express.js, hosted on Render)**
* **AI/ML Services (Skill Matcher, Interview Question Generator, Chatbot )**
* **Firebase or SQL Database**
* **Notification Modules (SendGrid, Firebase FCM)**
* **File Storage (Cloudinary / Firebase Storage)**

**Collected & Managed Data:**

* Users (Candidates, Recruiters, Admins)
* Jobs & Job Descriptions
* Applications & Status Tracking
* Skills & AI-Generated Recommendations
* Resume Files, Documents
* Payments & Subscriptions
* Chatbot Logs
* Interview Questions

**Programming Languages:**

* JavaScript (React.js, Node.js)

**Database Platform:**

* Firebase or MYSQL
* Collections: Users, Jobs, Applications, Skills, Payments, Chatbot Logs

**User Interfaces:**

* **Web Dashboards:** Candidate, Recruiter, Admin
* **Chatbot Widget:** Embedded in web dashboards

**Network Architecture:** Fully Internet-Based; HTTPS Secured

**System Hosting:**

* **Frontend:** Vercel
* **Backend API:** Render
* **Database:** SQL or Firebase Atlas
* **File Storage:** Cloudinary / Firebase Storage

**Third-Party Integrations:**

* **SendGrid** (Email Notifications)
* **Firebase Cloud Messaging (FCM)** (Mobile Push Notifications)
* **Cloudinary / Firebase Storage** (File & CV Uploads)
* **AI Microservices** (Skill Matching & Question Generation)

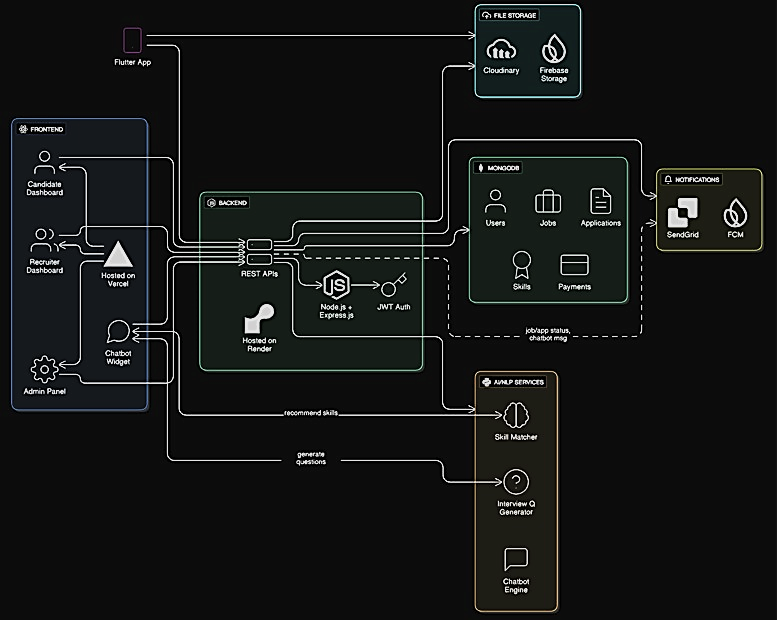
**Security Features:**

* **JWT Authentication (candidates, recruiters, admins)**
* **Role-Based Access Control (RBAC)**
* **Two-Factor Authentication for Admins (optional)**
* **AES-256 Encryption for sensitive data**
* **Secure API Gateway**

**Scalability Support:**

* **Horizontal Scaling with stateless APIs**
* **Firebase or SQL Sharding Support**
* **Load Balancing for API Layer**
* **CDN File Delivery (Cloudinary/Firebase)**

**Technical Architecture Diagram:**



## Application and Data Architecture

The system is built using a modern and scalable technology stack to support intelligent recruitment workflows and seamless user experiences across platforms:

• **Frontend:**  
Developed using **React.js** and **Tailwind CSS**, providing a responsive, dynamic, and user-friendly web interface for candidates, recruiters, and admins.

• **Backend:**  
Implemented using **Node.js** and **Express.js**, handling business logic, API services, authentication, and data processing.

• **Database:**  
Uses **SQL or Firebase** for flexible and scalable NoSQL data storage, ideal for handling unstructured or semi-structured data such as candidate profiles, job listings, and transactions.

• **AI & NLP Services:**  
Built with **Python** (using libraries like **spaCy**, **scikit-learn**, etc.) to support chatbot interaction, personalized skill recommendations, and AI-based interview question generation.

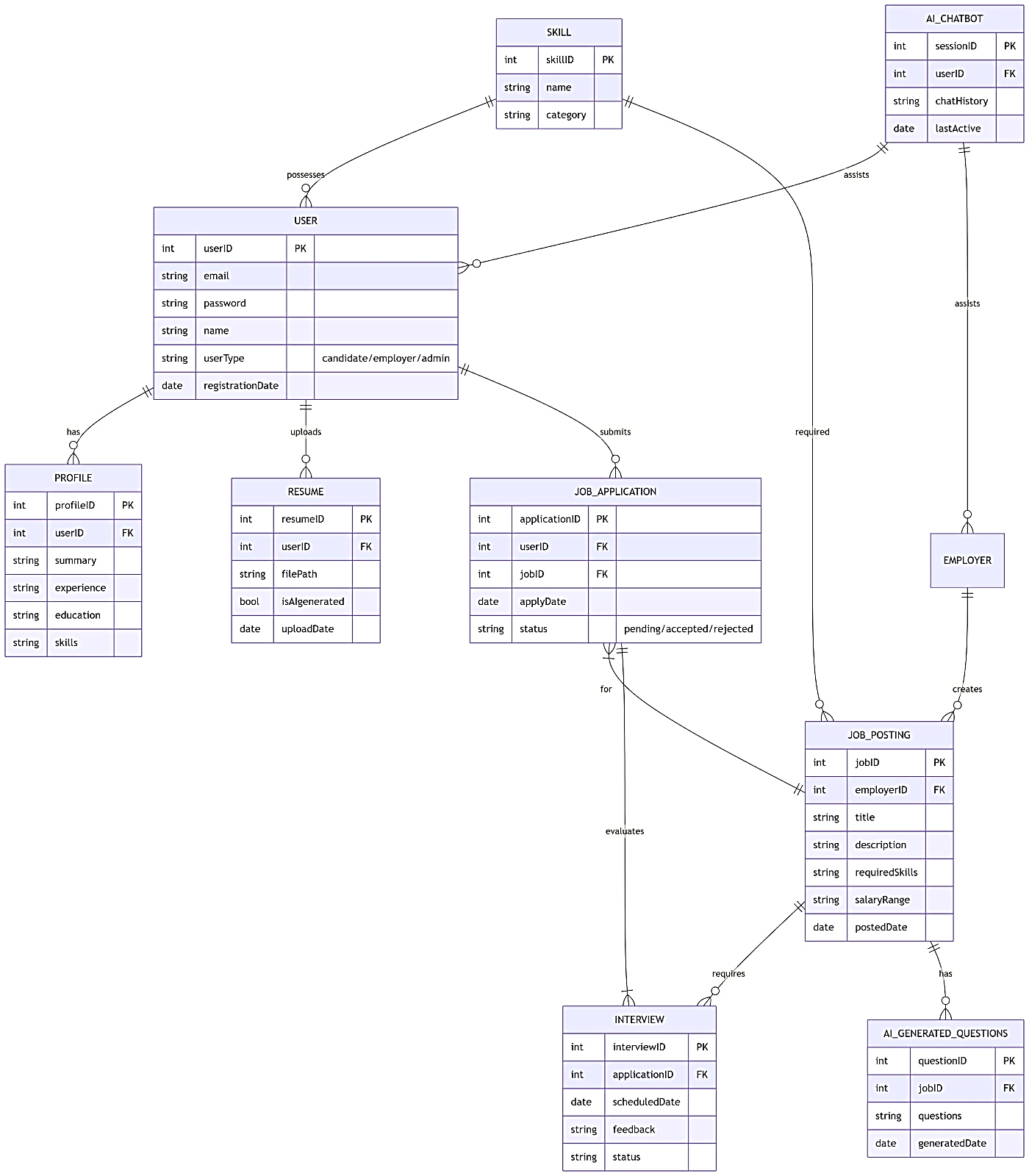
• **Mobile Application (Future Scope):**  
Planned to be developed using **Flutter** for cross-platform compatibility on both Android and iOS devices, offering accessibility and convenience to users on the go.

• **Authentication & Authorization:**  
Implemented using **local authentication** with **JWT** for secure login and session handling, along with **role-based access control (RBAC)** to restrict features by user role.

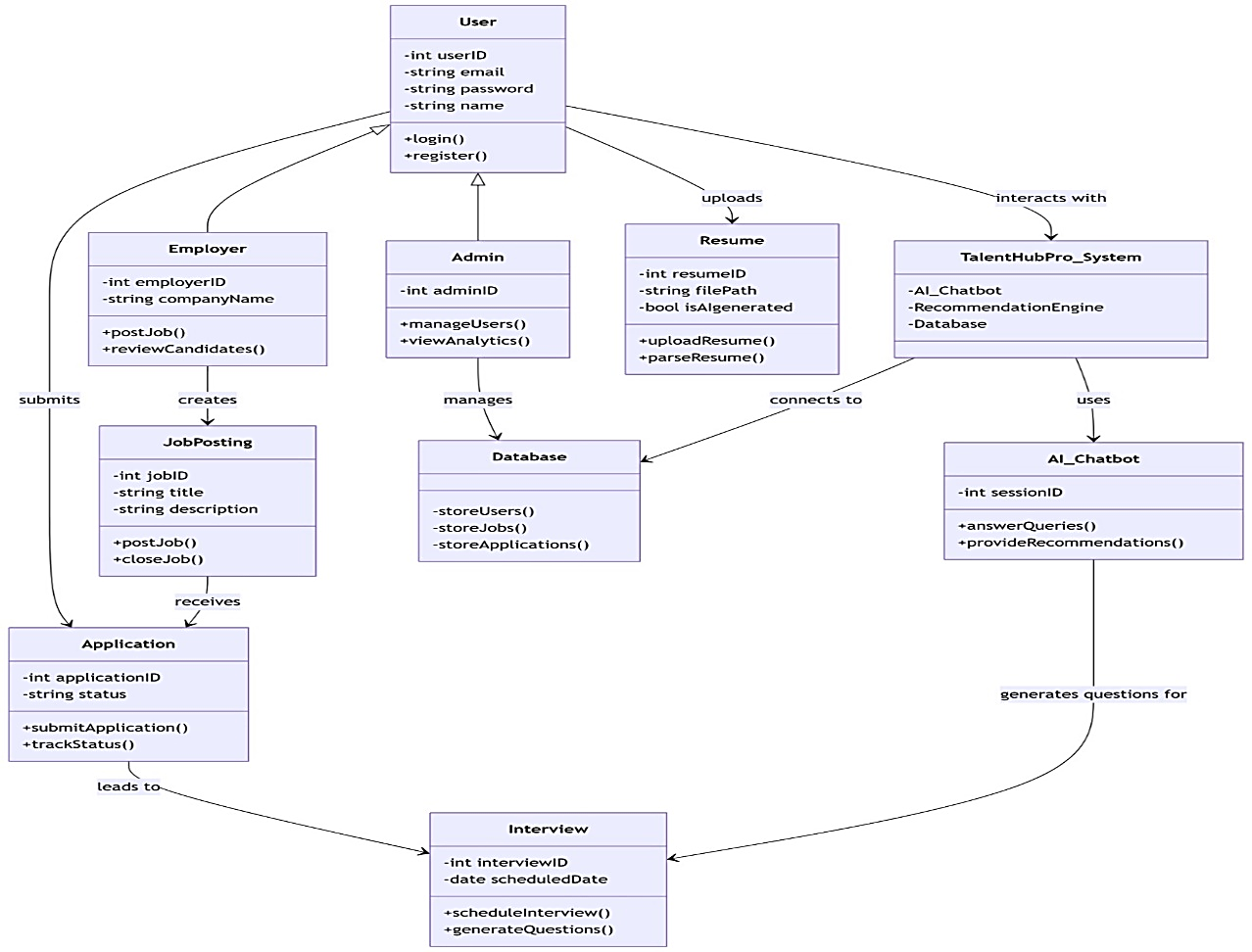
• **File & Document Storage:**  
Integrated with **Cloudinary** (or optionally Firebase Storage) to handle CV uploads, profile images, and other user-generated content securely and efficiently.

• **Hosting & Deployment:**  
Frontend deployed on **Vercel**, and backend deployed on **Render**, ensuring scalable and performant cloud deployment with continuous integration support.

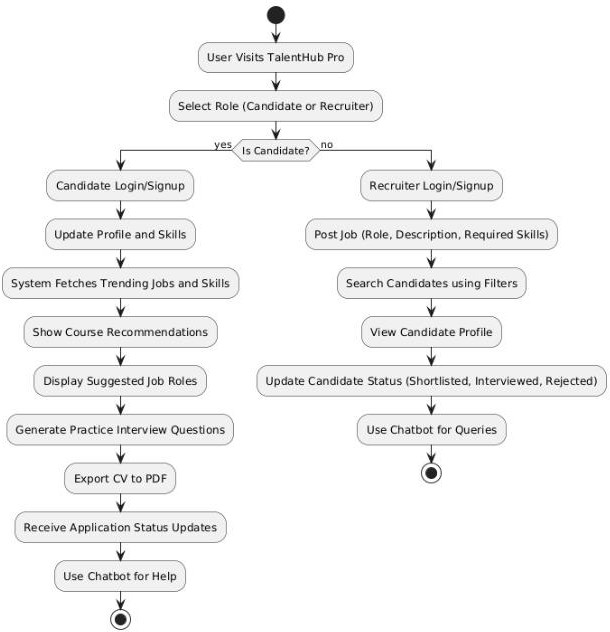
• **Notifications & Messaging:**  
Uses **SendGrid** for email notifications (e.g., job alerts, interview invitations) and optionally **Firebase Cloud Messaging (FCM)** for real-time push updates.

**Entity Relationship Diagram:**

**Abstract Class Diagram:**



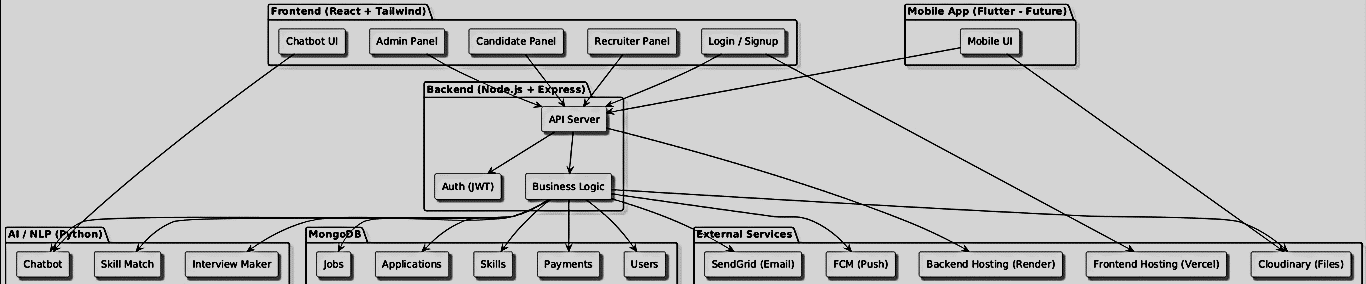
**Flowchart:**

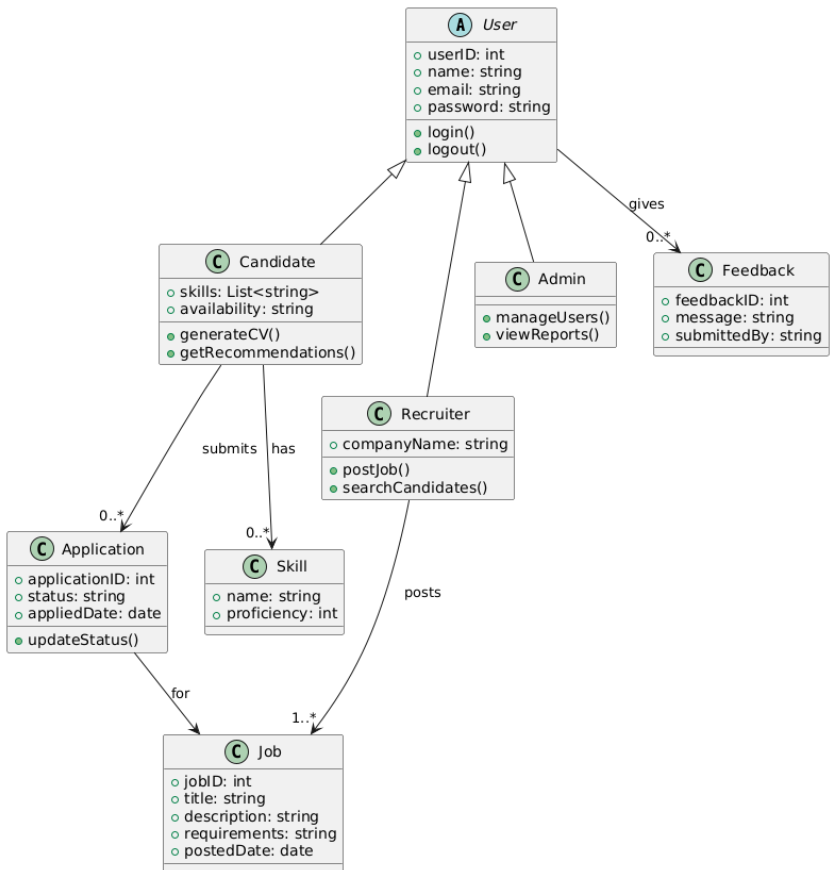
****

**Event Table:**

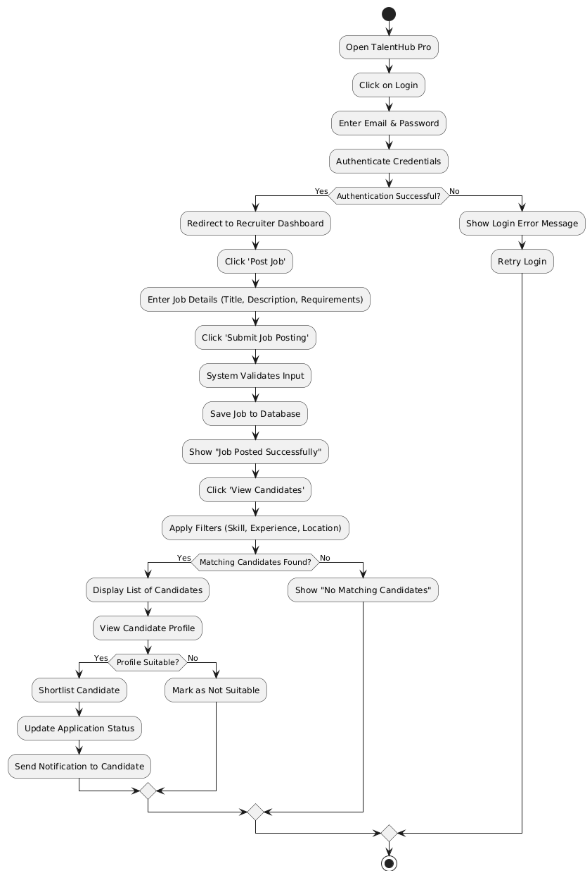
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S#** | **Event** | **Trigger Source** | **System Operation Triggered** | **System Response** |
| 1 | Candidate logs into system | Candidate | Authenticate credentials | Redirect to dashboard |
| 2 | Candidate updates profile | Candidate | Store updated profile data and skills | Confirmation of profile update |
| 3 | Candidate selects a job role for practice questions | Candidate | Fetch relevant interview question templates | Practice questions displayed |
| 4 | Candidate clicks "Get Course Recommendations" | Candidate | Analyze skill set and trending market skills | List of recommended courses and job paths shown |
| 5 | Recruiter logs in | Recruiter | Verify credentials | Redirect to recruiter dashboard |
| 6 | Recruiter posts a new job | Recruiter | Save job description and requirements | Job successfully listed |
| 7 | Recruiter applies filters to search candidates | Recruiter | Query database with selected filters | Display matching candidate profiles |
| 8 | Recruiter updates a  candidate’s status | Recruiter | Modify status in database | Updated status reflected in system |
| 9 | Candidate clicks "Export CV" | Candidate | Format and generate CV document | Downloadable PDF file provided |
| 10 | User (candidate/recruiter) interacts with chatbot | Candidate/Recruiter | Parse query against predefined rule-based Q&A | Relevant answer displayed in chat window |

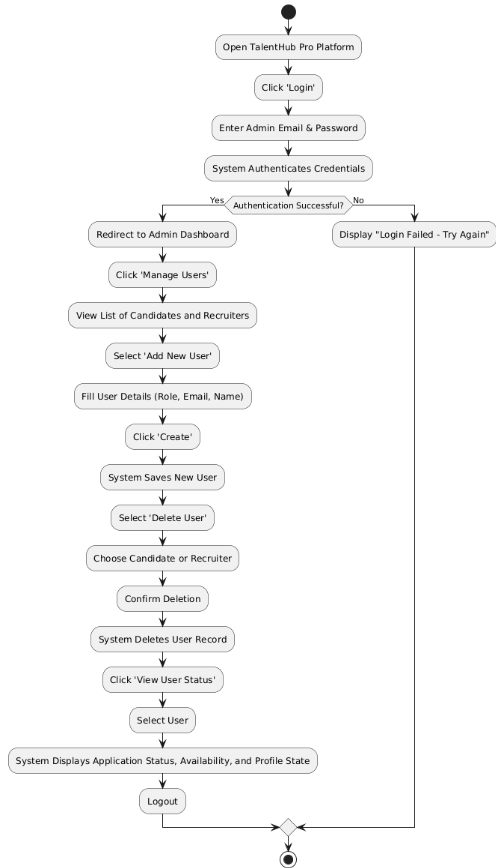
**Component Diagram:**

****

**Class Diagram:**

**Activity Diagram For Candidate:**

**Activity Diagram For Recruiter:**

**Activity Diagram For Admin:**

**Decision Tables:**

**Login Validation:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Condition** | **Valid Email** | **Correct Password** | **User Role** | **Action** |
| Case 1 | Yes | Yes | Candidate | Redirect to Candidate Dashboard |
| Case 2 | Yes | Yes | Recruiter | Redirect to Recruiter Dashboard |
| Case 3 | Yes | Yes | Admin | Redirect to Admin Dashboard |
| Case 4 | Yes | No | Any | Show "Invalid Password" Error |
| Case 5 | No | Any | Any | Show "Email Not Registered" Error |

**Profile Completion Check (Candidate)**

|  |  |  |
| --- | --- | --- |
| Condition | Profile Complete | Action |
| Case 1 | Yes | Allow Job Application |
| Case 2 | No | Redirect to Profile Completion Page |

**Application Status Update (Recruiter)**

|  |  |  |
| --- | --- | --- |
| Condition | Status Selected | Action |
| Case 1 | Selected | Update Status to "Selected" |
| Case 2 | Rejected | Update Status to "Rejected" |
| Case 3 | Interviewed | Update Status to "Interviewed" |
| Case 4 | No Status | Show Error Message |

**Chatbot Query Matching**

|  |  |  |
| --- | --- | --- |
| Condition | Query Matches Predefined FAQ | Action |
| Case 1 | Yes | Display Matching Answer |
| Case 2 | No | Show "Sorry, I don't understand" |

**Candidate Skill Suggestion**

|  |  |  |  |
| --- | --- | --- | --- |
| Condition | Skill Gap Exists | Job Role Available | Action |
| Case 1 | Yes | Yes | Suggest Skill Development Courses |
| Case 2 | No | Yes | No Suggestion |
| Case 3 | Yes | No | Show "No Matching Role" Notification |

**Interview Question Generation**

|  |  |  |  |
| --- | --- | --- | --- |
| Condition | Job Selected | Profile Exists | Action |
| Case 1 | Yes | Yes | Generate Interview Questions |
| Case 2 | No | Any | Show Error: "Select Job Role" |
| Case 3 | Yes | No | Prompt to Complete Profile |

**Admin User Management**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Condition | Action Selected | User Role | Input Valid | Action |
| Case 1 | Add User | Candidate | Yes | Add Candidate to System |
| Case 2 | Add User | Recruiter | Yes | Add Recruiter to System |
| Case 3 | Delete User | Any | Yes | Delete User from System |
| Case 4 | Add/Delete | Any | No | Show Validation Error |

**Job Filtering Logic (Recruiter View)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Condition | Skill Filter | Experience Filter | Location Filter | Action |
| Case 1 | Yes | Yes | Yes | Show Matching Candidates |
| Case 2 | No | Yes | Yes | Show Partial Matches |
| Case 3 | No | No | No | Show All Candidates |
| Case 4 | Invalid Input | Any | Any | Show Error Message |

**Job Posting Validation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Condition | Title Present | Description Present | Requirements Present | Action |
| Case 1 | Yes | Yes | Yes | Post Job Successfully |
| Case 2 | No | Any | Any | Show Error: "Missing Title" |
| Case 3 | Yes | No | Any | Show Error: "Missing Desc" |
| Case 4 | Yes | Yes | No | Show Error: "Add Requirements" |

**Logout Handling**

|  |  |  |  |
| --- | --- | --- | --- |
| Condition | Logout Clicked | Confirmed | Action |
| Case 1 | Yes | Yes | Logout and Redirect |
| Case 2 | Yes | No | Stay on Dashboard |
| Case 3 | No | Any | No Action |

## Component Interactions and Collaborations

The system’s architecture ensures seamless collaboration between components and roles:

#### ****Job Application Workflow (Candidate & Recruiter):****

* **Candidate** browses available jobs via the portal.
* **System** fetches job listings from the database.
* **Candidate** applies for a job.
* **System** validates the candidate’s profile and application.
* **Recruiter** is notified of the new application.
* **Recruiter** reviews candidate details and updates status (e.g., Shortlisted/Rejected).
* **Candidate** receives a notification of their updated application status.

#### ****Interview Question Generation:****

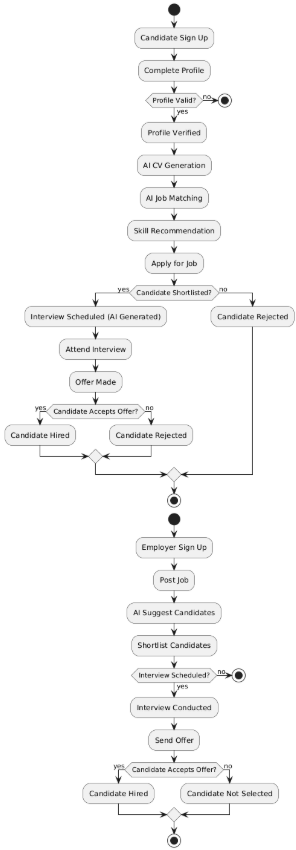
* **Candidate** selects a job role and fill given requirments.
* **System** uses NLP module to fetch job-related interview questions.
* **Questions** are displayed and can be exported by the candidate.

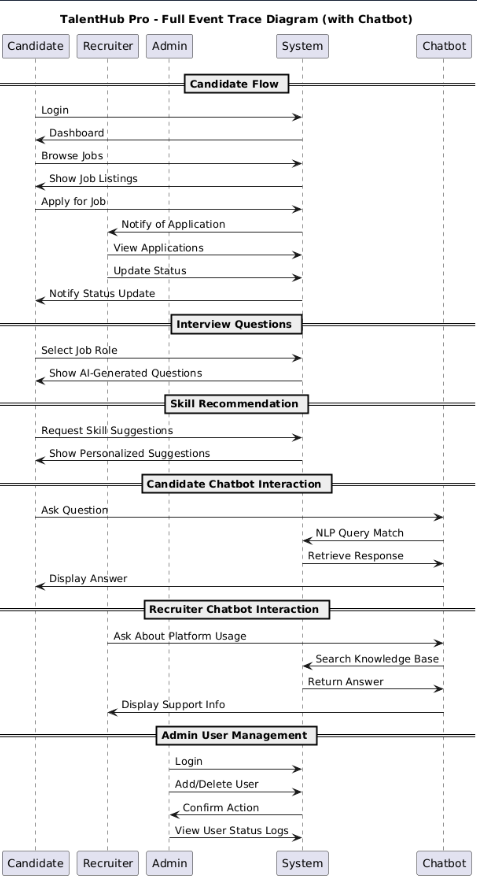
#### ****Skill Recommendation System:****

* **Candidate** clicks on “Get Recommendations.”
* **System** analyzes the candidate's profile and compares it with job market data.
* **Recommendations** (skills/courses) are displayed to the candidate.

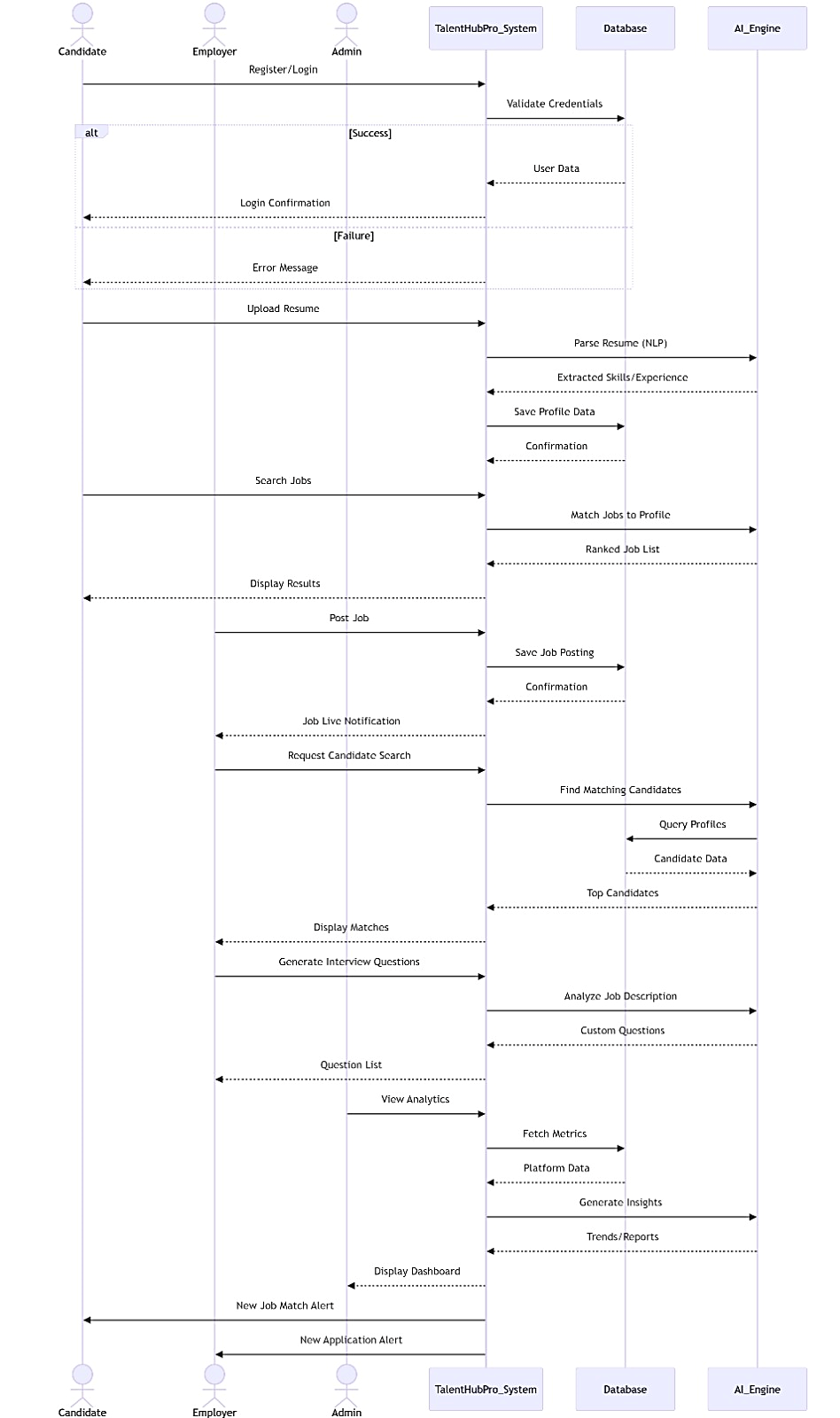
#### ****Admin User Management:****

* **Admin** logs into the platform.
* **Admin** adds or deletes users (Candidates or Recruiters).
* **System** updates the database and confirms action.
* **Admin** can view application activity and user statuses in real-time.

**Activity Diagram:**

**Event trace Diagram**

**Sequence Diagram**

****

## Design Reuse and Design Patterns

The system reuses key components and applies industry-standard design patterns to ensure modularity, maintainability, and scalability.

**Reused Components:**

* **UI Elements:**  
  Reusable React.js components (e.g., input fields, dropdowns, buttons, job cards, modals) ensure consistency across screens.
* **Middleware Functions:**  
  Common Node.js middleware includes authentication guards, input validators, and error handlers reused across API routes.
* **Utility Functions:**  
  Shared logic for tasks such as:
  + Date formatting
  + Role checking
  + User availability status
  + String parsing  
    Used both in frontend and backend layers

**Design Patterns Used:**

* **MVC (Model-View-Controller):**
  + **Model:** SQL or Firebase schemas for Candidates, Recruiters, Jobs, Applications
  + **View:** React.js components for rendering dynamic user interfaces
  + **Controller:** Node.js + Express.js controllers handling logic and DB operations
* **Observer Pattern (Planned):**  
  For enabling **real-time notifications** like job updates and message alerts via Firebase Messaging.
* **Singleton Pattern:**  
  Ensures a **single shared database connection** (e.g., using Mongoose or SQL or Firebase client instance) across the backend to optimize performance.

## ****Technology Architecture****

### ****Hosting:****

* **Frontend Hosting:**  
  Deployed via **Vercel** (or optionally Firebase Hosting) for automated scaling, CDN delivery, and HTTPS support.
* **Backend Hosting:**  
  Node.js + Express.js backend hosted on **Render**, **Railway**, or **Heroku**, managing APIs, authentication, and core logic.
* **Database Hosting:**  
  **SQL or Firebase Atlas** used for secure, cloud-based NoSQL database hosting, replacing earlier MySQL-based structure.

### ****Connectivity:****

* **HTTPS (TLS Encryption):**  
  Ensures all communication between clients and servers is encrypted and secure.
* **RESTful APIs:**  
  Backend provides **stateless and modular REST APIs** consumed by the frontend and mobile clients.
* **WebSockets (Planned):**  
  Will be used for **real-time communication** such as job alerts and recruiter-candidate interactions.

### ****Modes of Operation:****

* **Web Application:**
  + Built using **React.js** and **Tailwind CSS**
  + Fully responsive design for desktops, tablets, and mobile browsers
  + Compatible with Chrome, Firefox, Edge, and Safari

### Security Measures

* **User Authentication**:  
  Handled via **local JWT-based authentication**, replacing Firebase.
* **Data Encryption**:  
  Sensitive data (e.g., credentials, resumes, applications) stored securely using **AES-256** encryption.
* **Two-Factor Authentication (2FA)** (Planned):  
  For admin and recruiter accounts to enhance account security during critical actions.

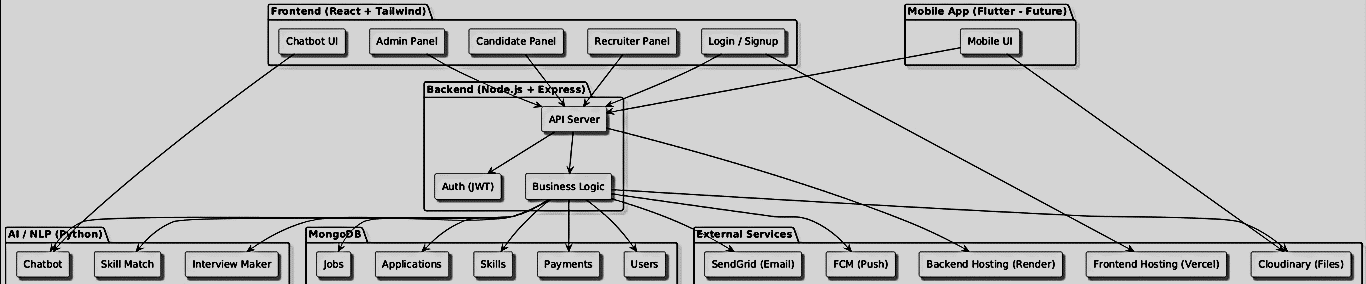
## Architecture Evaluation

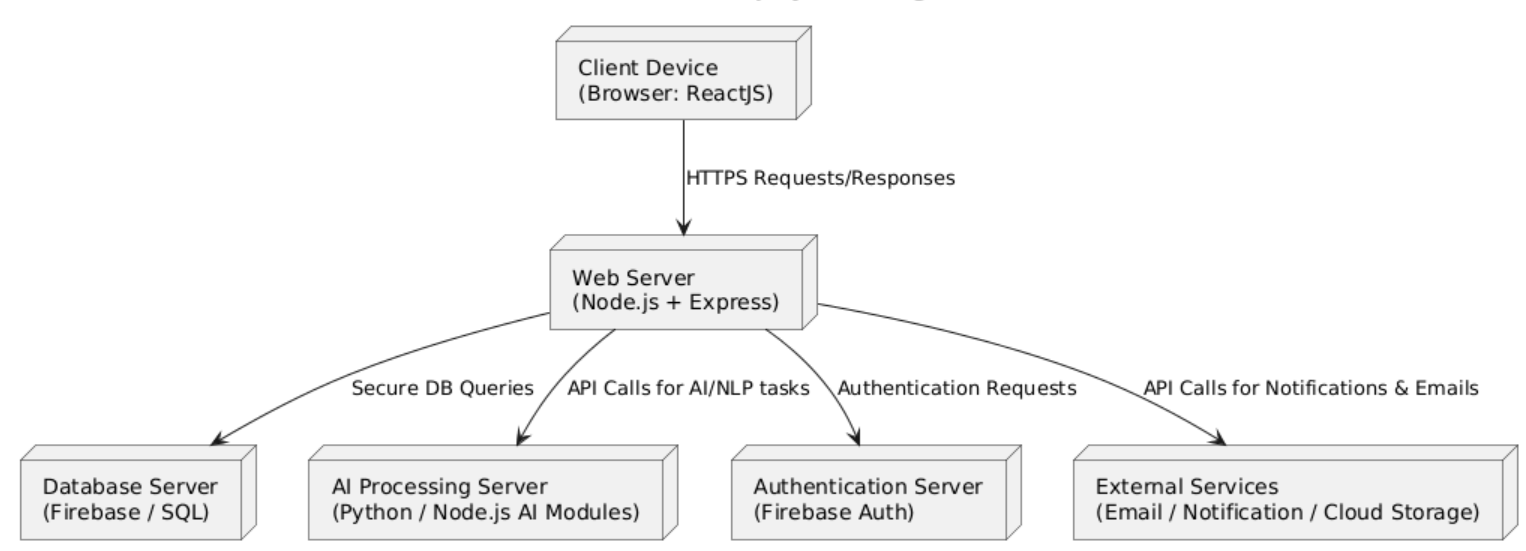
The technologies used in **TalentHub Pro** were selected based on scalability, AI-integration capability, ease of development, and modern recruitment workflow requirements.

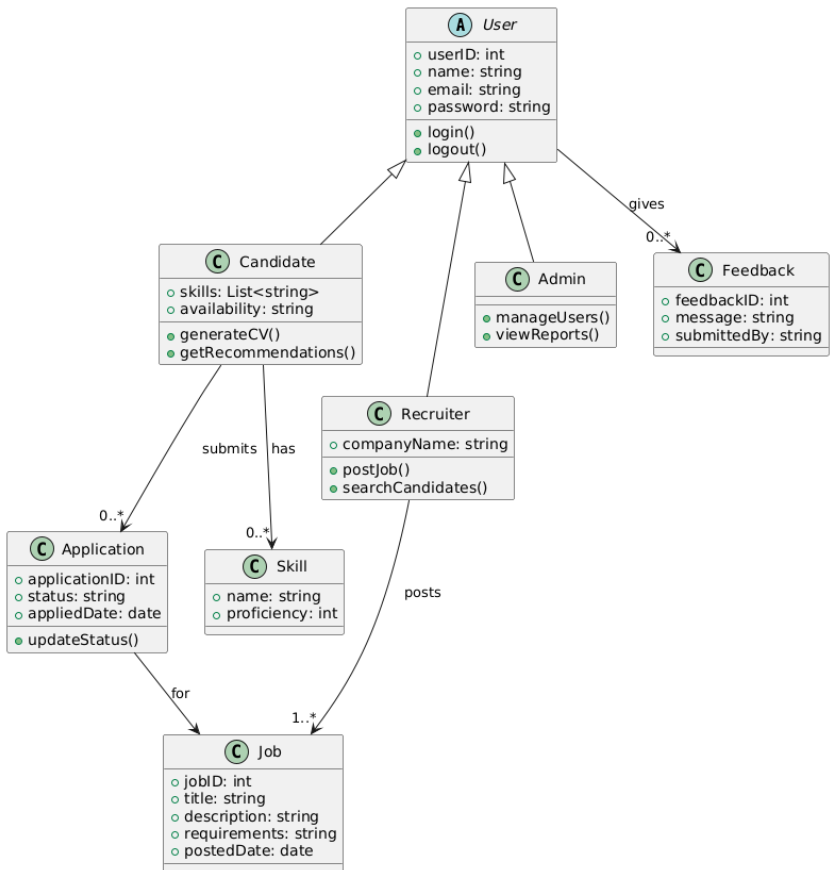
* **Backend (Node.js + Express):** Chosen for its fast, event-driven nature and seamless compatibility with AI modules and the database layer. Alternatives like Django and Laravel were considered but dropped due to heavier setup and lower flexibility for real-time candidate filtering.
* **Frontend (React.js + Flutter):** React.js supports a modular and responsive interface for recruiters and candidates. Flutter was selected for building cross-platform mobile apps with a single codebase. React Native was considered but Flutter offered smoother UI performance and better widget control.
* **Database (Firebase Firestore / SQL):** Selected for its reliability and structured data handling. Firestore provides real-time updates, ideal for job postings, applications, and chat interactions. SQL databases were also evaluated for structured recruitment workflows but were avoided due to less flexibility for rapid AI-driven changes.
* **AI & NLP Modules (Python Microservices / OpenAI API):** Used for CV analysis, skill extraction, and interview question generation. Local ML frameworks were evaluated but avoided due to high computational cost and deployment complexity.
* **Notifications (SendGrid / Firebase Cloud Messaging):** SendGrid provides reliable email delivery for verification and job alerts, while FCM enables real-time mobile notifications. SMS providers like Twilio were avoided due to higher cost.
* **Hosting (Vercel / Render):** Vercel supports fast frontend deployment while Render efficiently manages backend auto-scaling. AWS and Azure were considered but avoided due to higher cost and complex configuration for academic project needs.

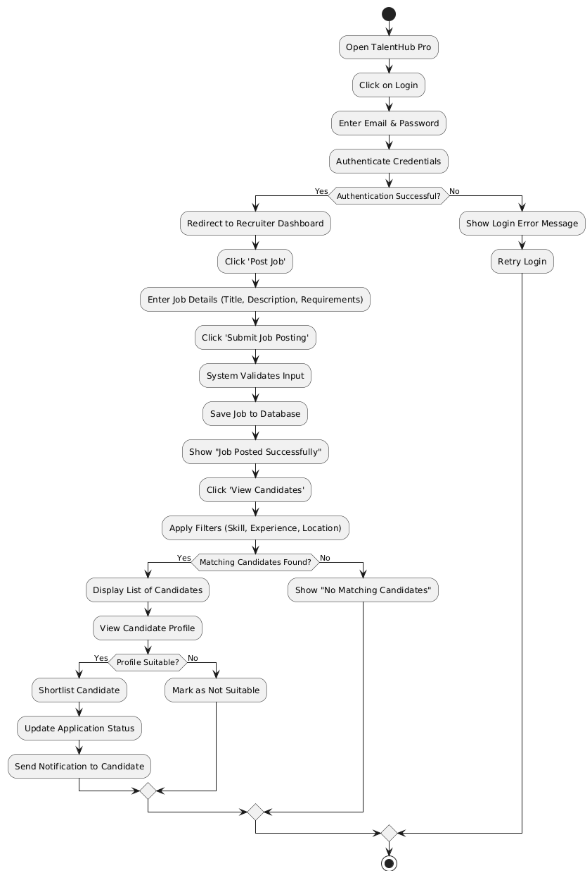
Overall, the selected stack balances performance, cost, scalability, and AI capabilities, making it suitable for a real-world, modern, intelligent recruitment platform.

# Detailed/Component Design

**Component Diagram:**

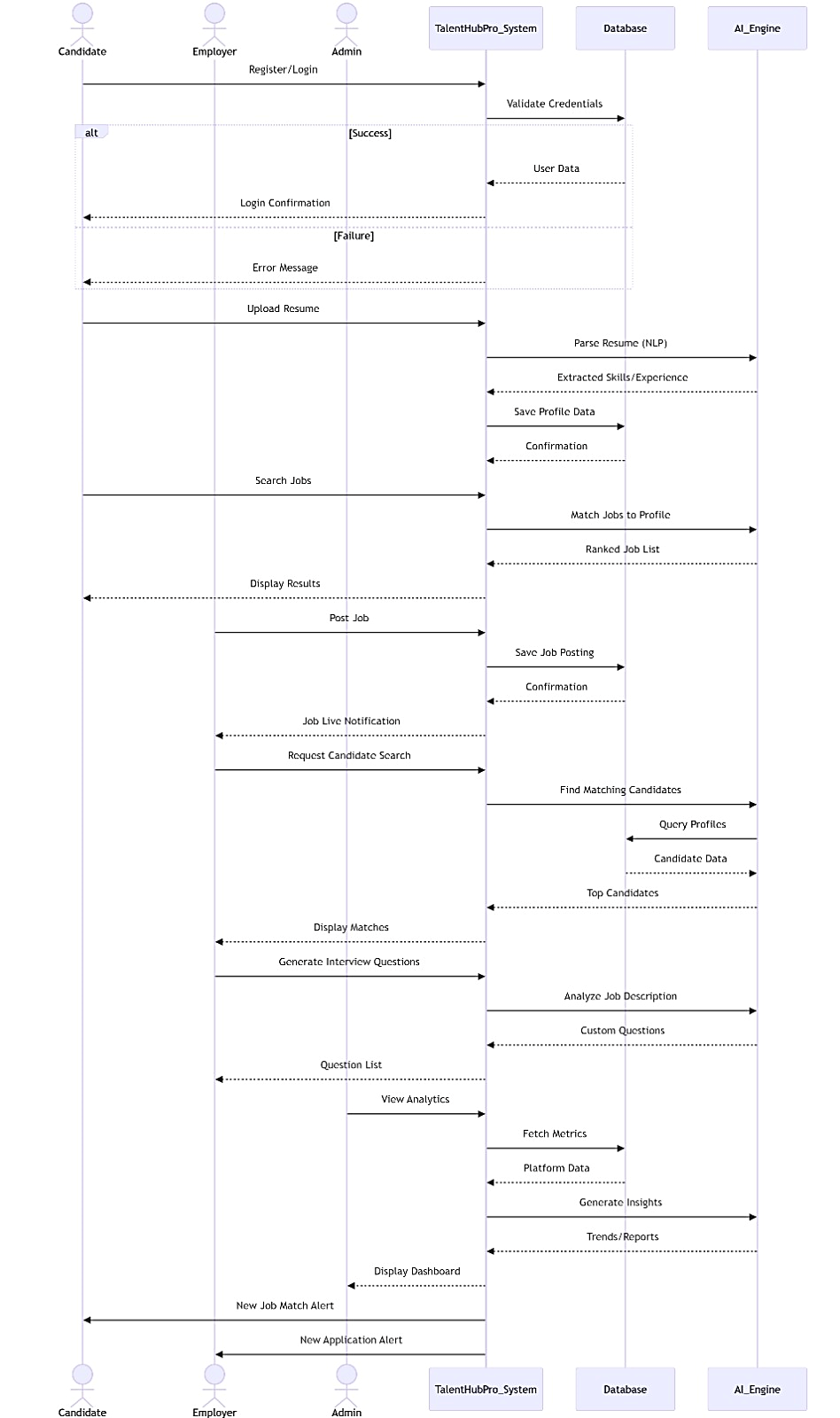
**Deployment diagram:**

**Class Diagram:**

**Activity Diagram For Recruiter:**

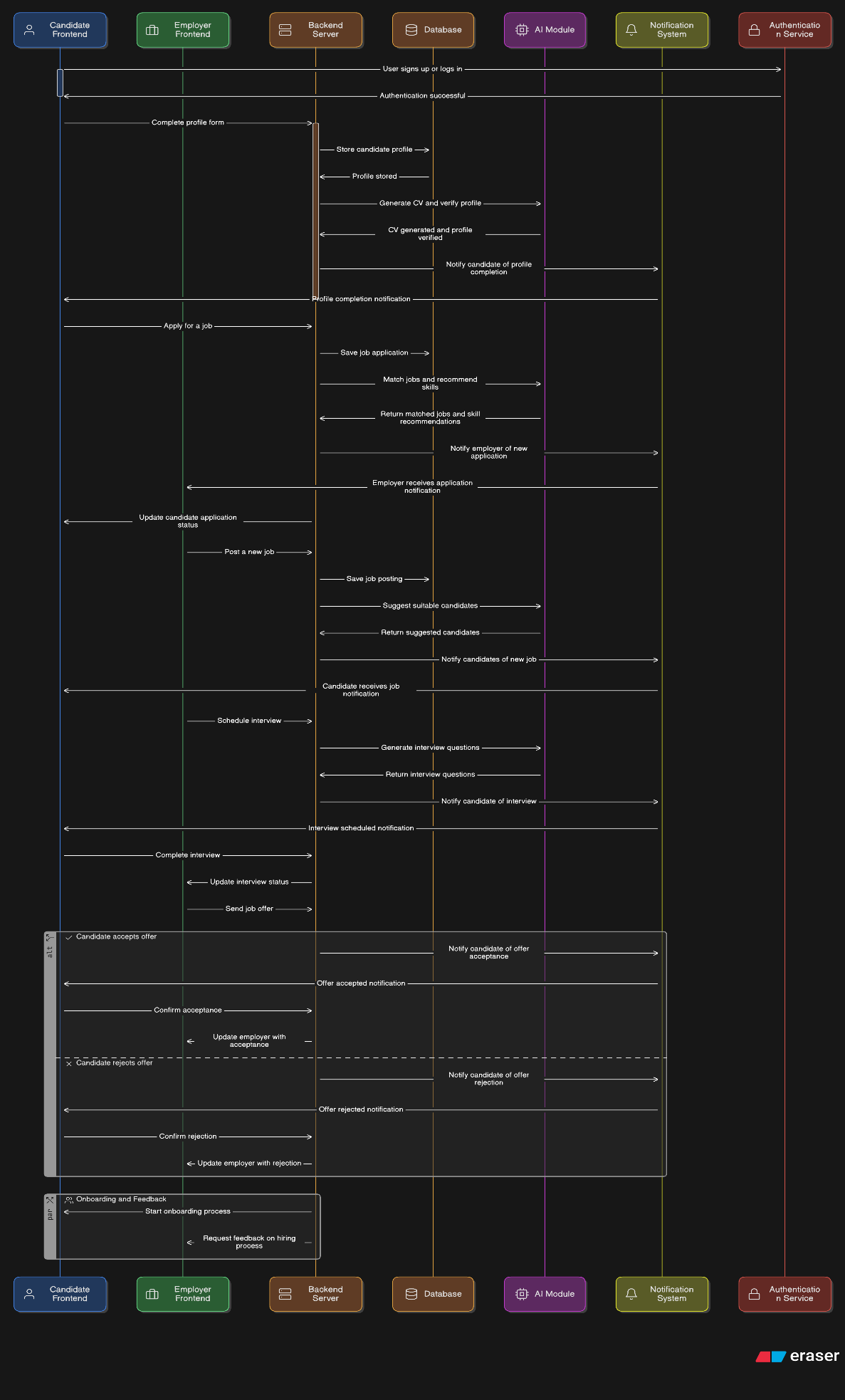
**Activity Diagram For Candidate:**

**State Diagram:**

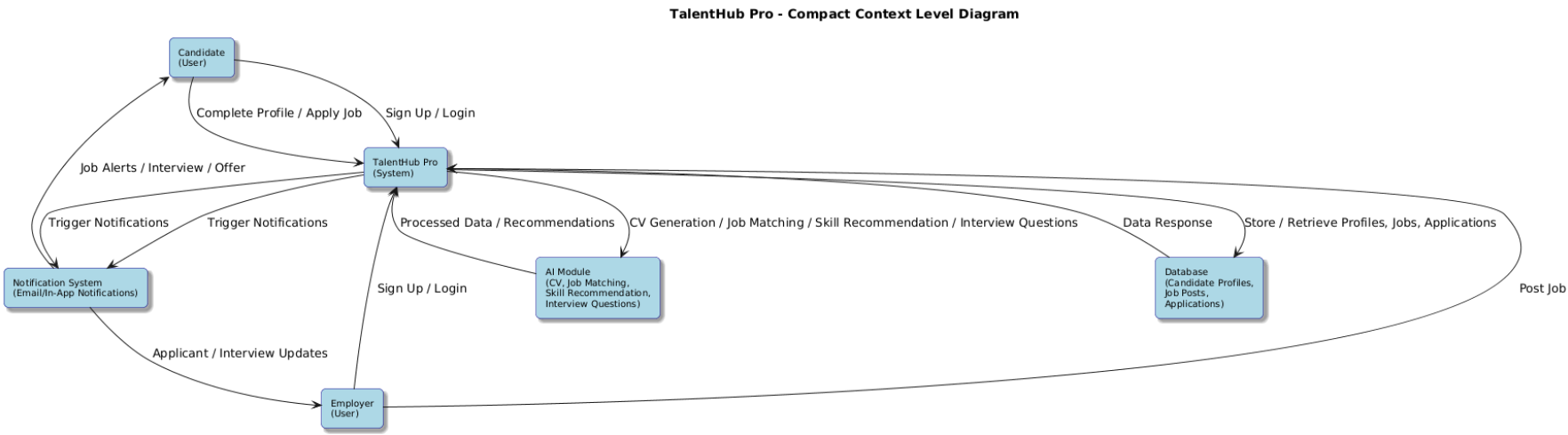
**Sequence Diagram**

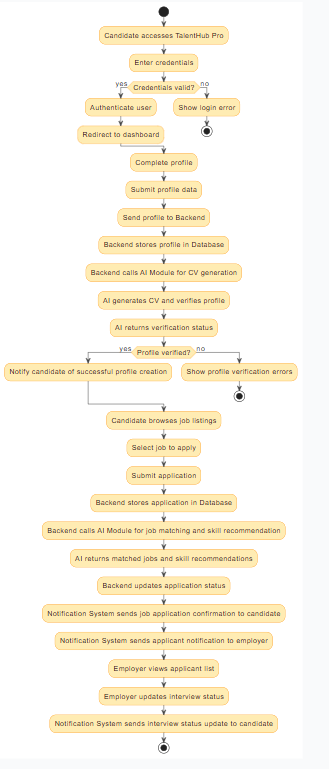
## Component-Component Interface

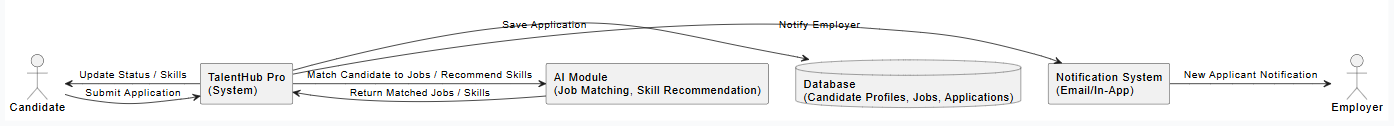
**Collaboration Diagram:**



## Component-External Entities Interface

**Context Level Diagram:**

**Dataflow Diagram Level 1:**

**Dataflow Diagram Level 2:**

## Component-Human Interface

| **Screen / Interface** | **Input Provided By User** | **Output Provided To User** |
| --- | --- | --- |
| **Login / Registration** | Username/Email, Password, User Role (Job Seeker/Recruiter) | Login success/failure, redirect to respective **Dashboard** |
| **Job Seeker Dashboard** | View posted jobs, access skill recommendations, click for CV generation | Summary of application status, personalized skill suggestions, quick links to core features |
| **Recruiter Dashboard** | View candidate pipeline, manage job postings, initiate filtering | Summary of active job posts, candidate filtering metrics, access to AI tools |
| **Profile Setup / Update** | Experience, Availability, Skills, Salary Expectations, Educational details | Confirmation of data update, visual completeness score, **auto-profile update** success message |
| **Job Posting Form (Recruiter)** | Job Title, Description, Required Skills, Salary Range, Company Profile | Confirmation message, Job ID, link to view the live posting |
| **CV Generation Interface** | Selection of required CV format/template | Structured, formatted CV document (PDF/DOCX) for download |
| **Smart Chatbot Interface** | Text/Voice input for queries (e.g., *“How to apply?”, “Update my skills”*) | Real-time response, interactive communication, system-assisted actions (e.g., initiating a profile update) |
| **Interview Preparation Module** | Selection of Job Description, desired company profile | **Dynamic interview questions** generated based on input data, preparation tips |
| **Application Confirmation Screen** | Confirmation of submission (Job Seeker) | Confirmation message with application ID, estimated response time |
| **Admin Panel** | User management (add/edit/delete), content moderation (job posts/profiles) | Updated user lists, success/error alerts, system performance metrics |

**HCI (Human-Computer Interaction) Norms Followed:**

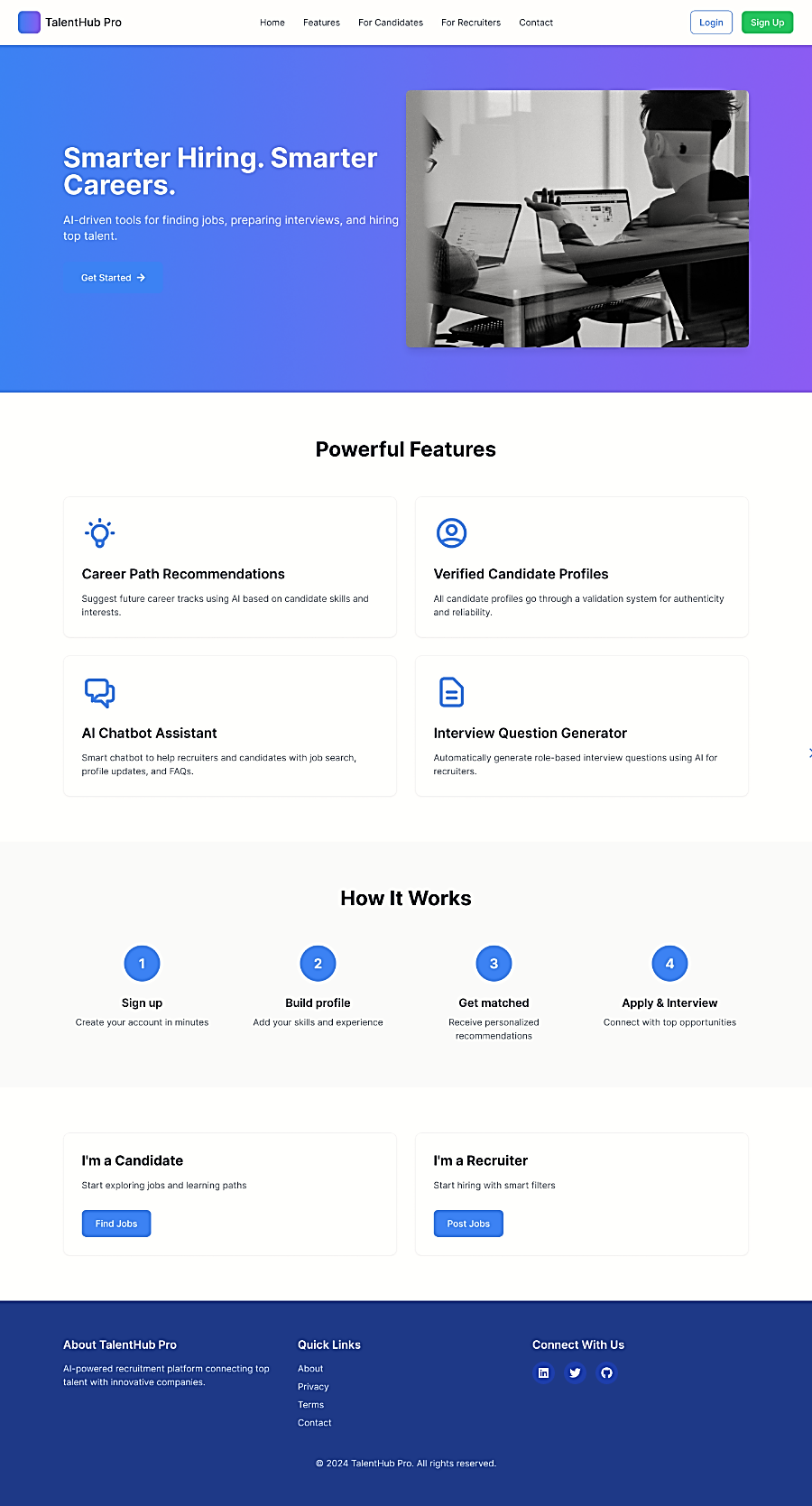
* **Consistency:** Uniform layout, colors, and components across screens.
* **Feedback:** Instant visual responses for key actions (e.g., application submission, AI filtering).
* **Error Handling:** Validations and clear error messages to guide corrections.
* **User Control:** Users can edit profiles and manage job listings/customizations.
* **Minimalist Design:** Simple, functional UI with no unnecessary elements.
* **Accessibility:** High contrast, readable fonts, and large clickable areas.
* **Mobile-Friendly:** Fully responsive design for various screen sizes.

# Screenshots/Prototype

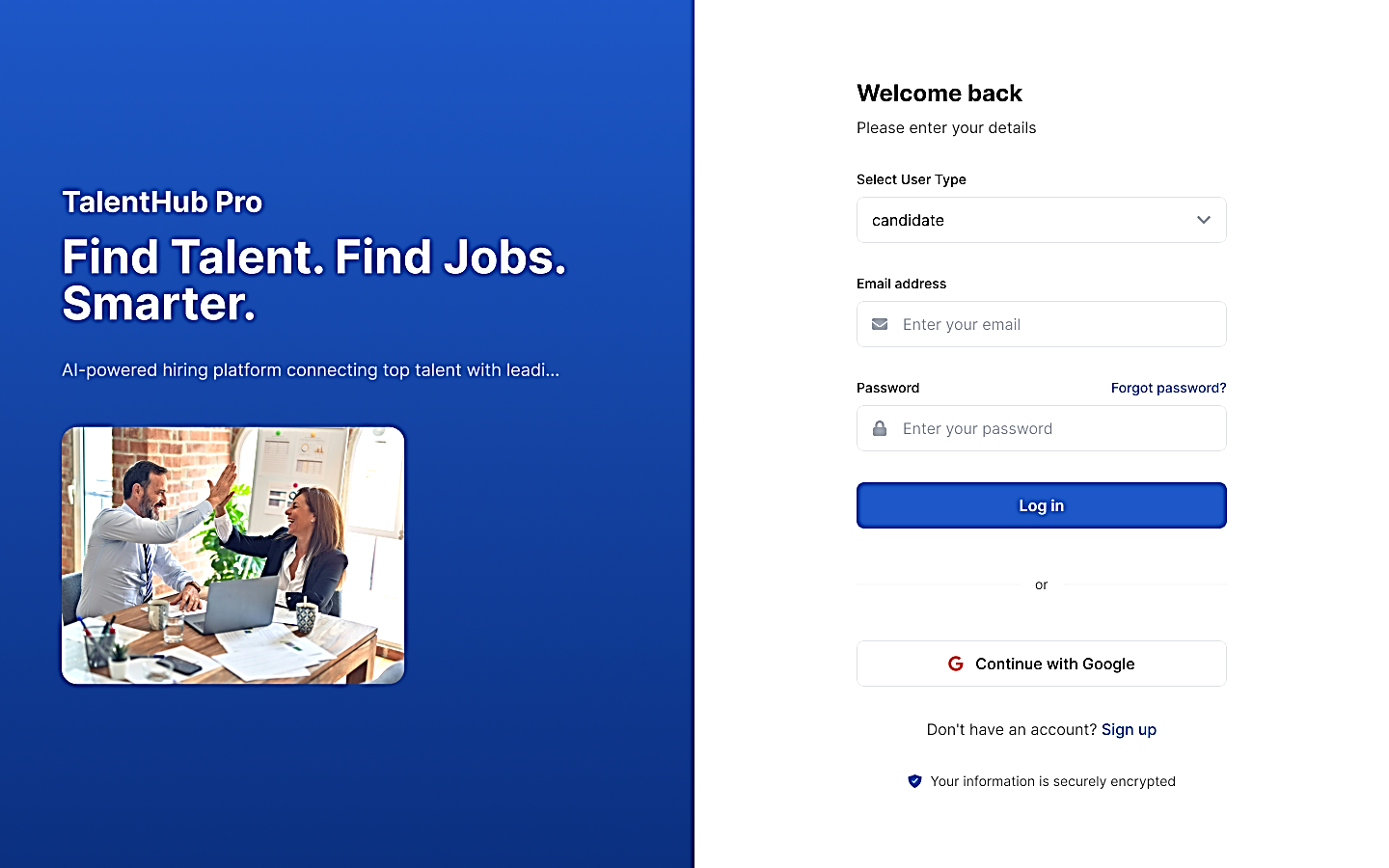
## Workflow

## Screens:

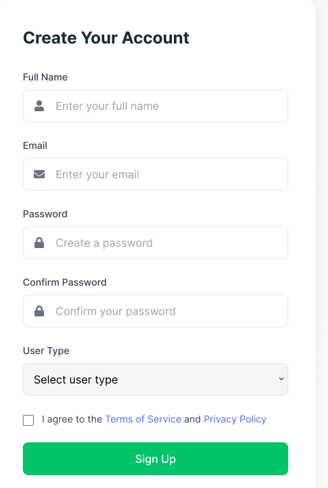
**Landscape:**



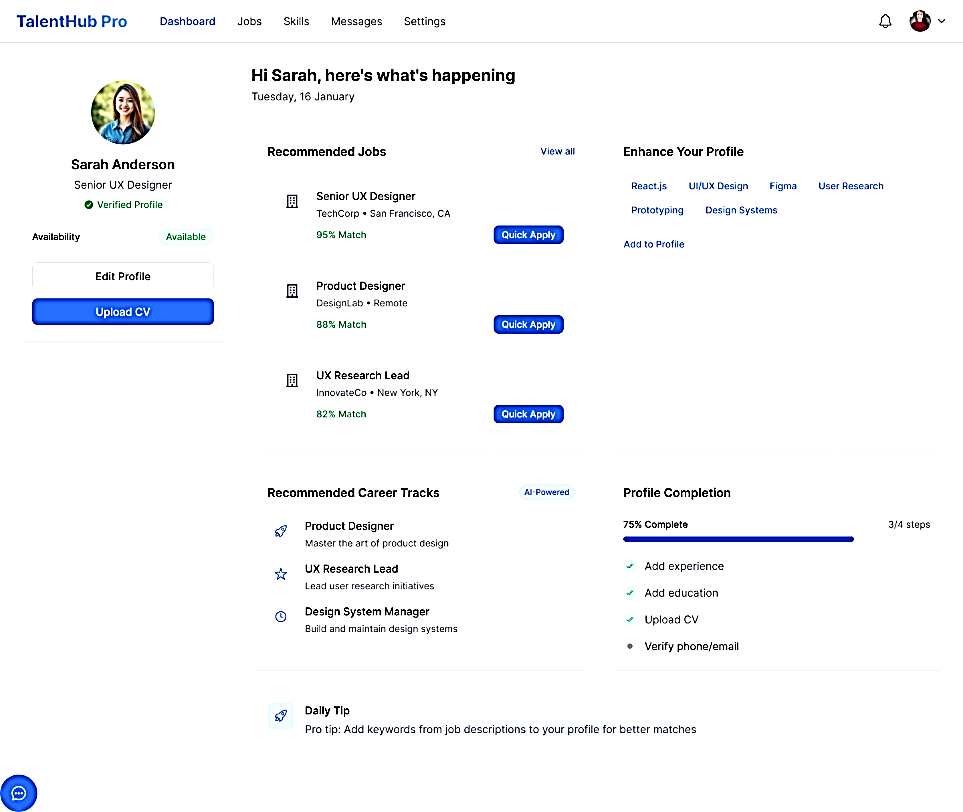
**Login :**

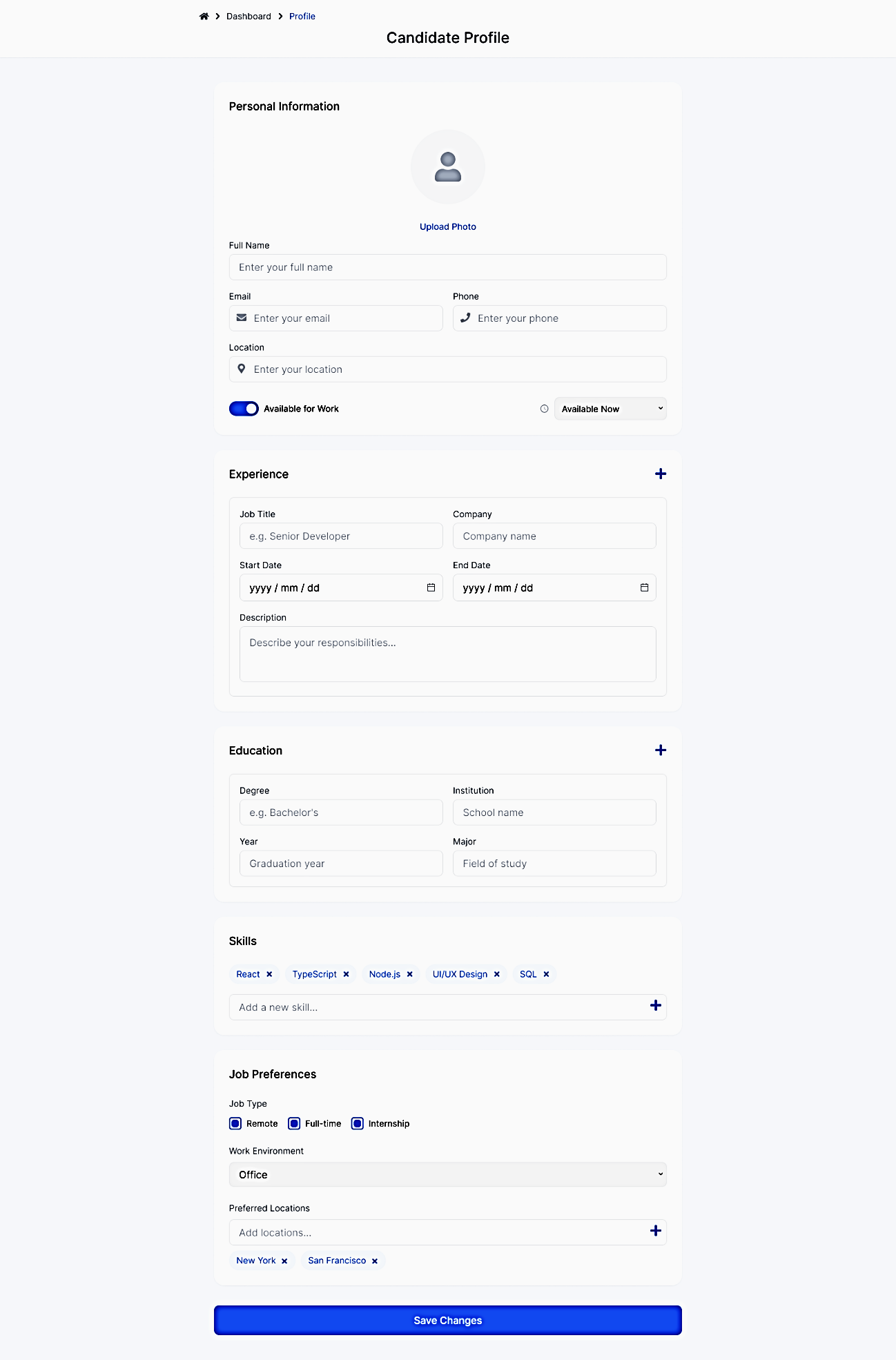


**Signup:**

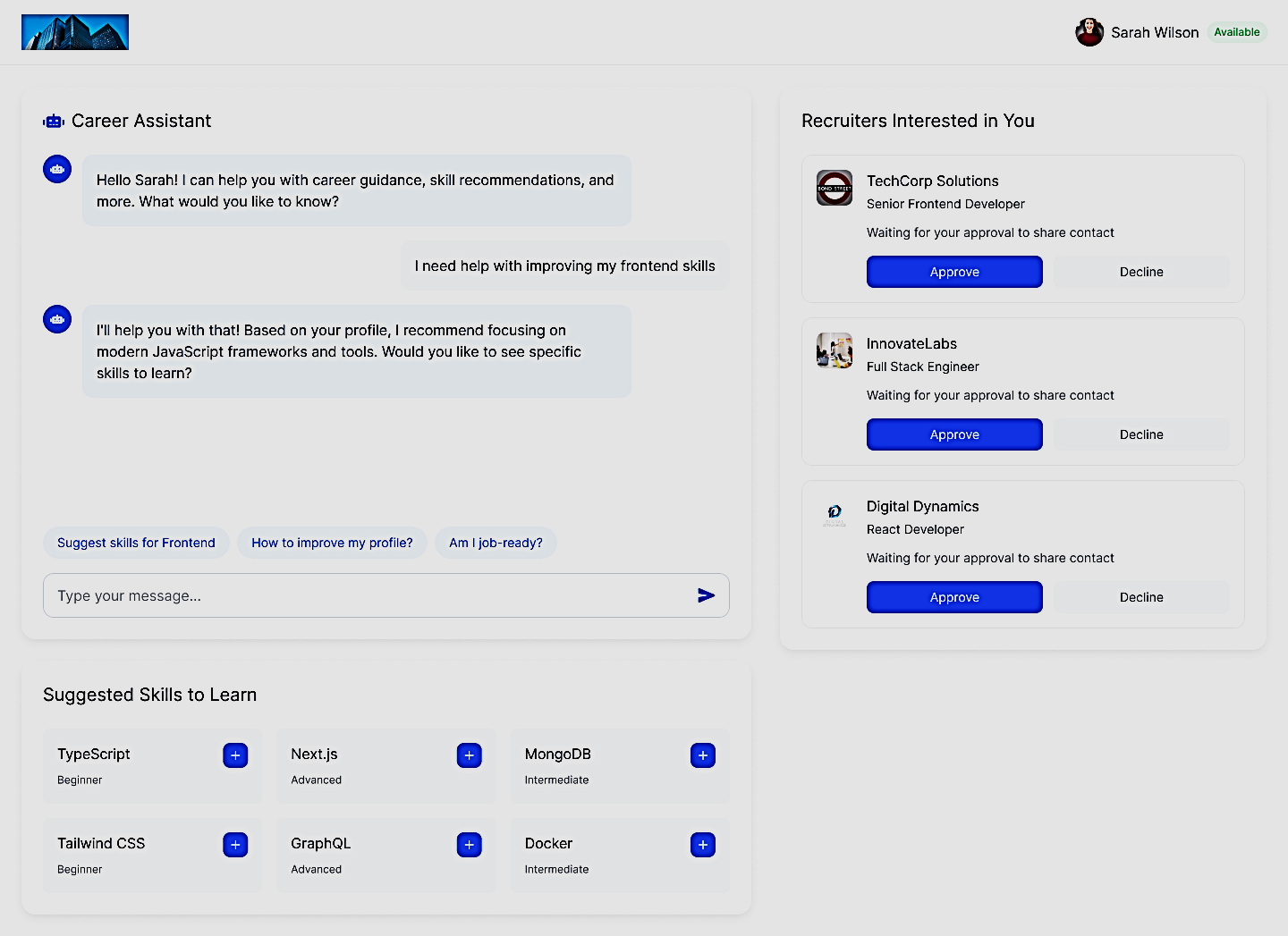
****

**Candidate Dashboad:**

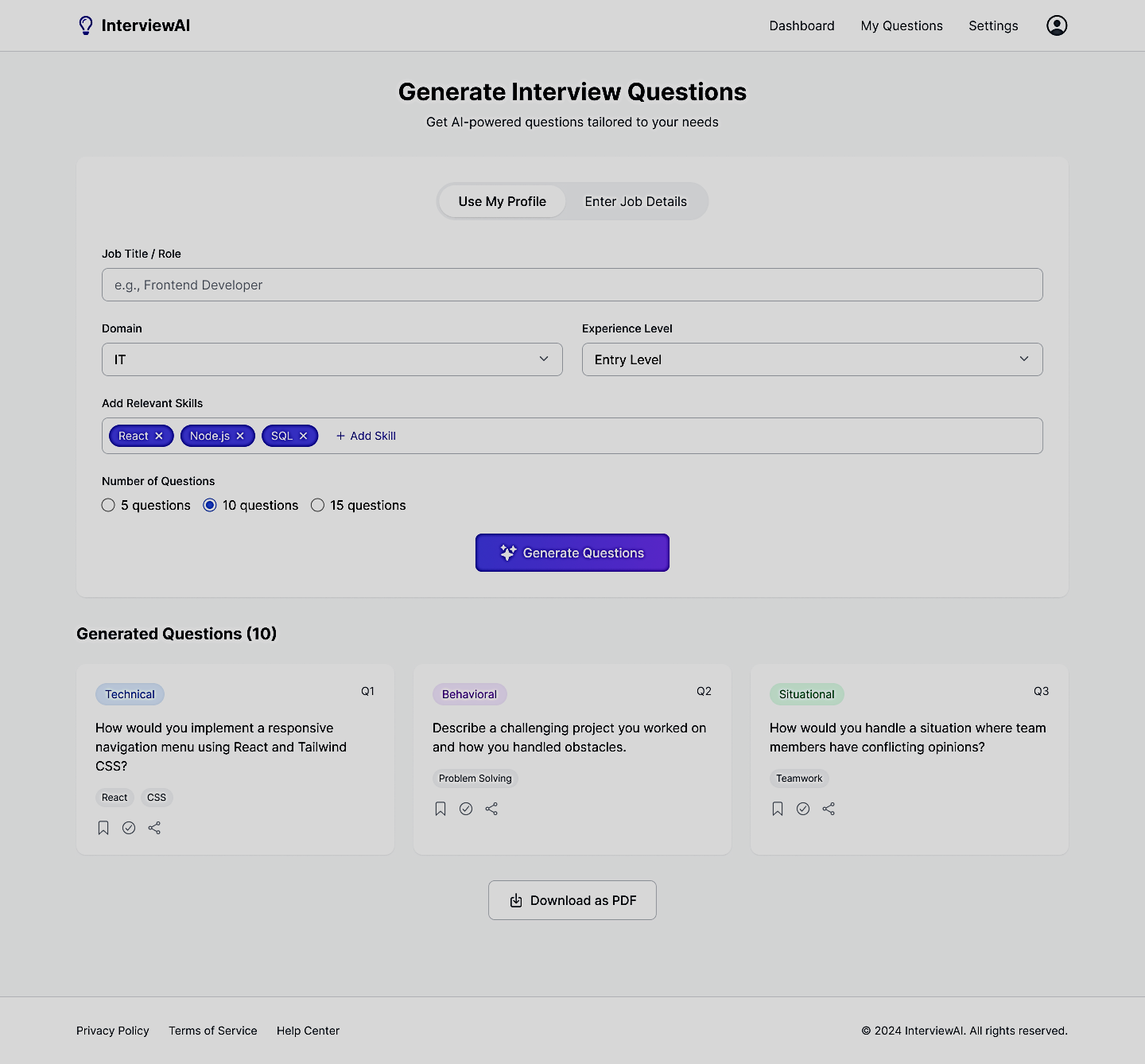
****

**Candidate Profile:**

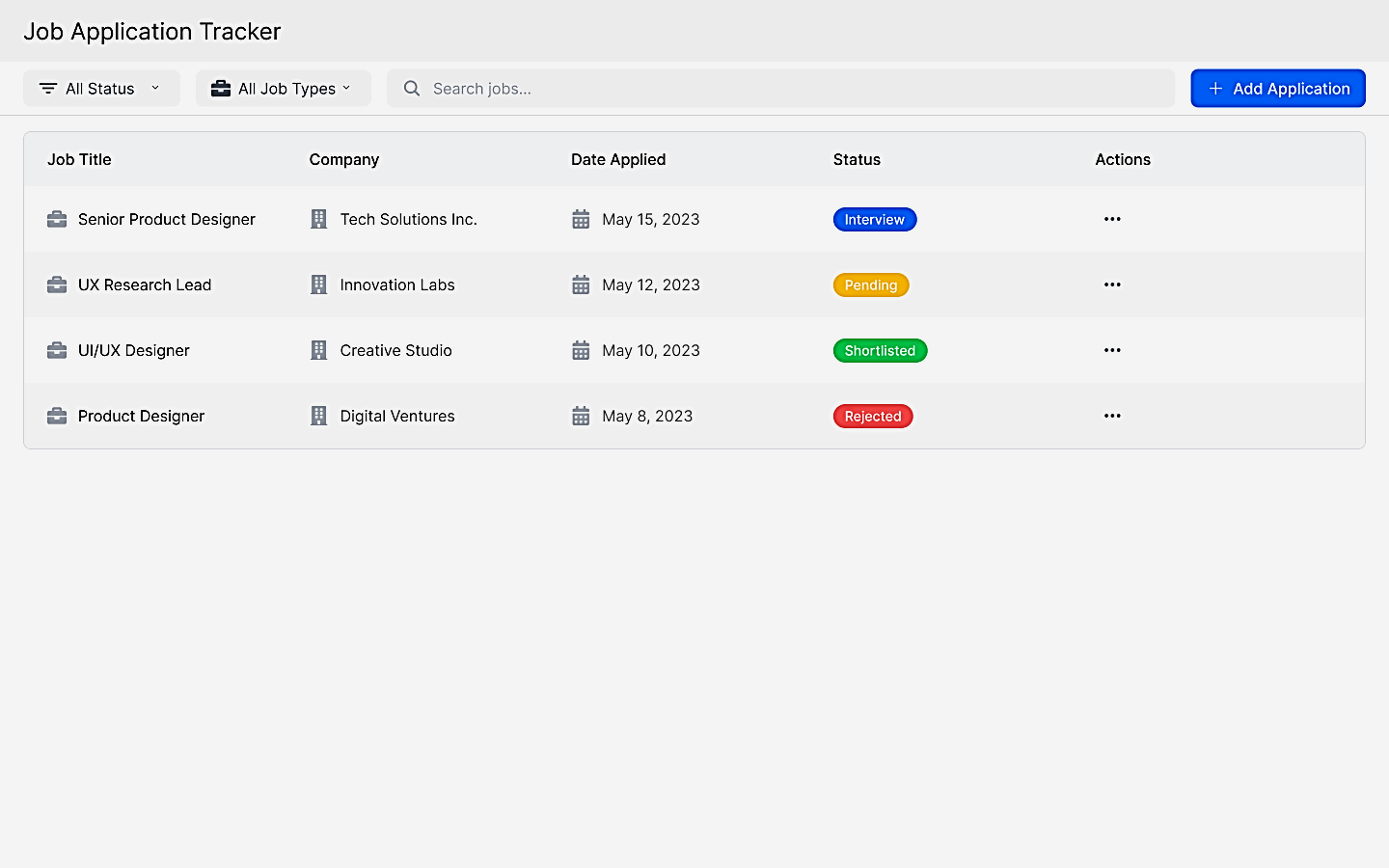
**Skill Recommendation And Chatbot:**



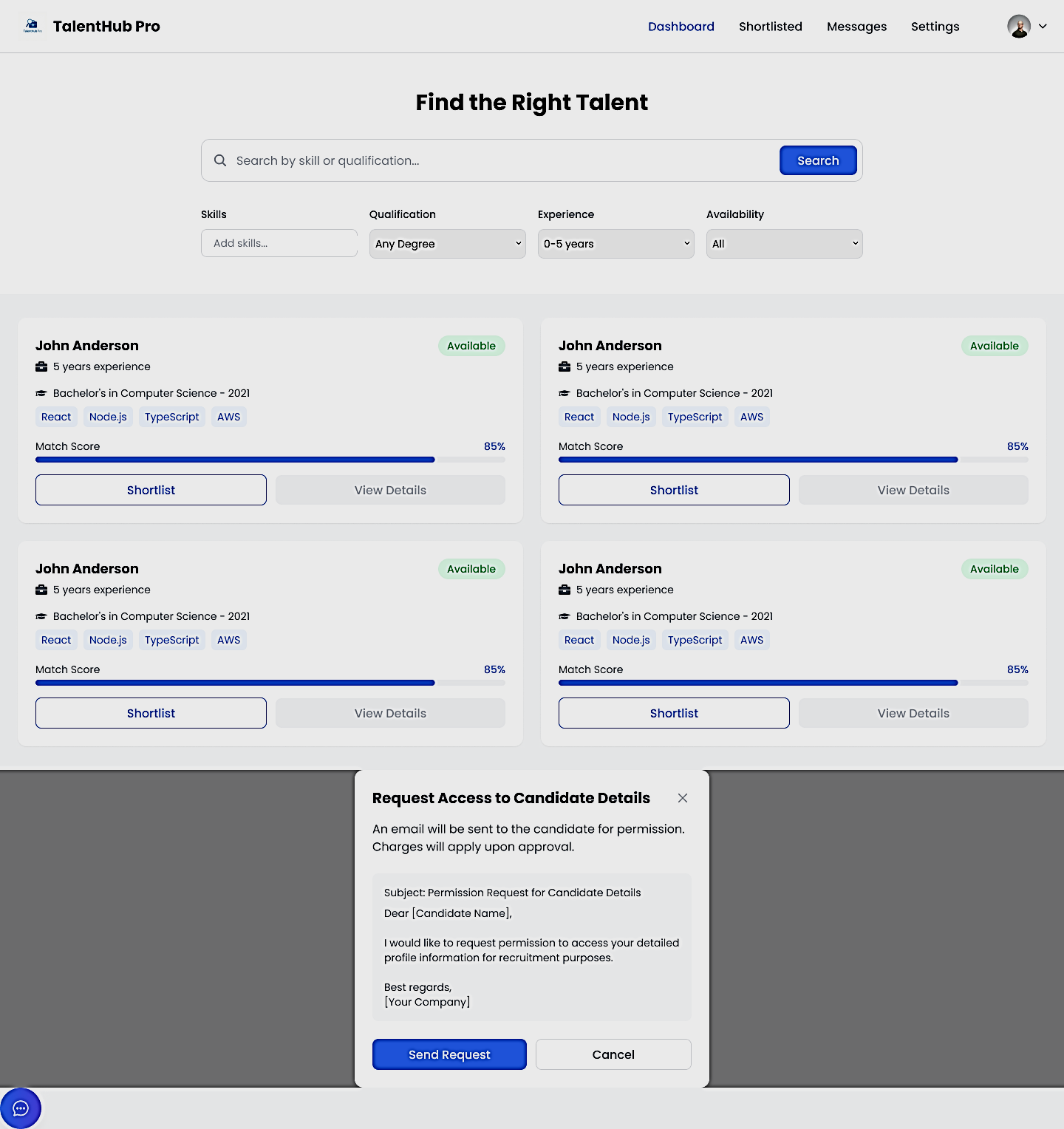
**Interview Question For Candidate:**



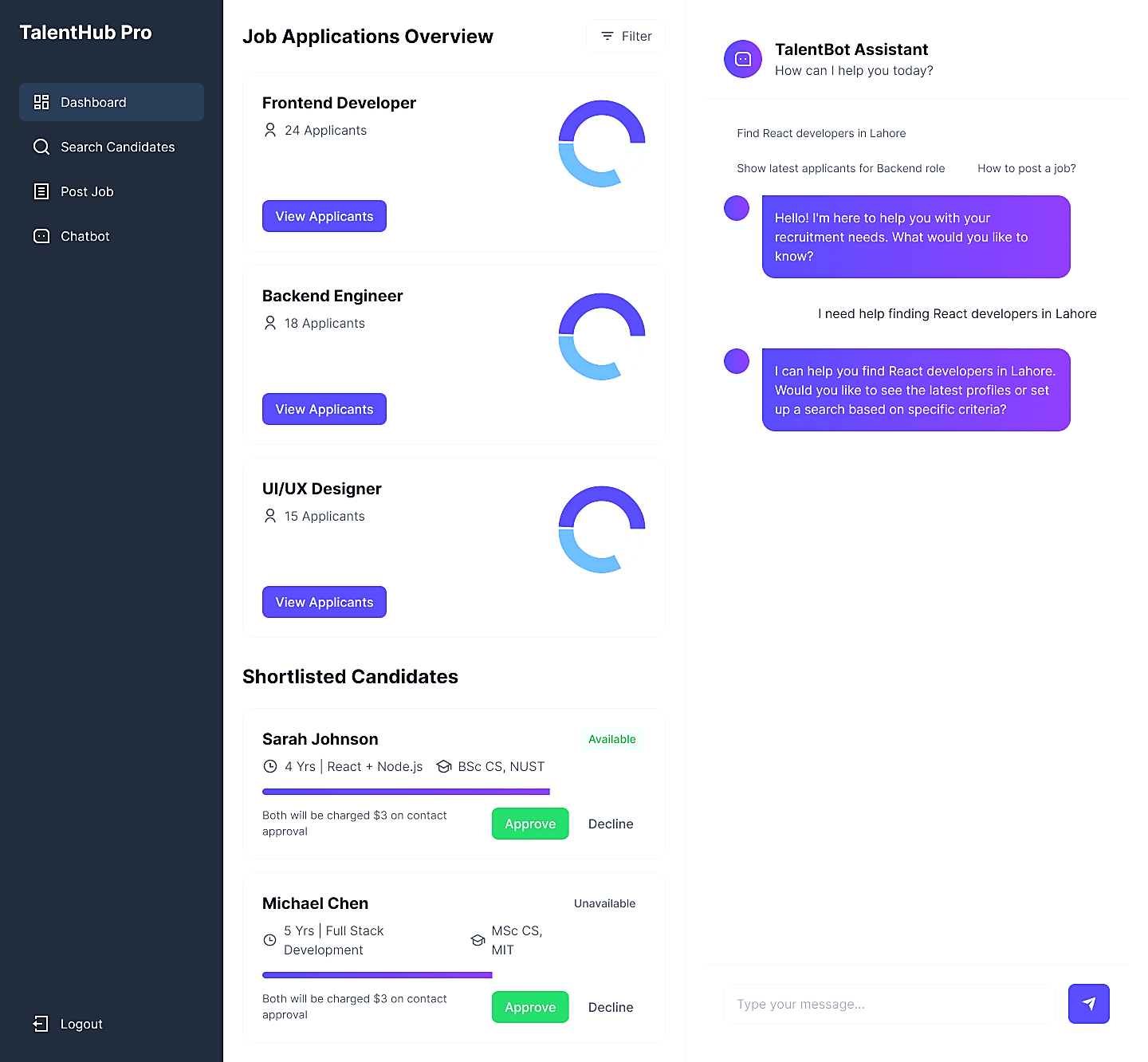
**Application Tracker For Candidate:**



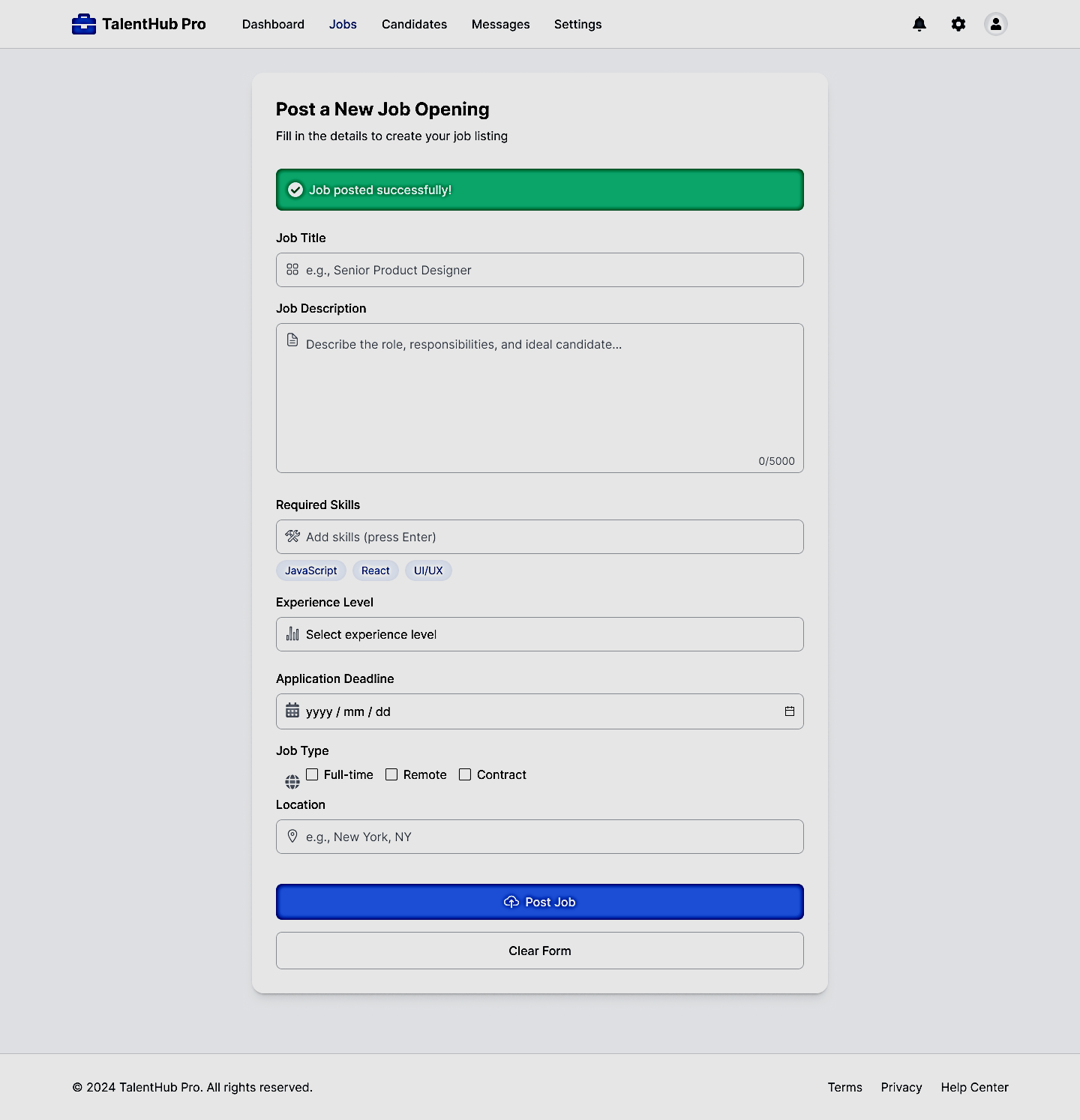
**Recruiter Dashboard And Candidate Searching:**



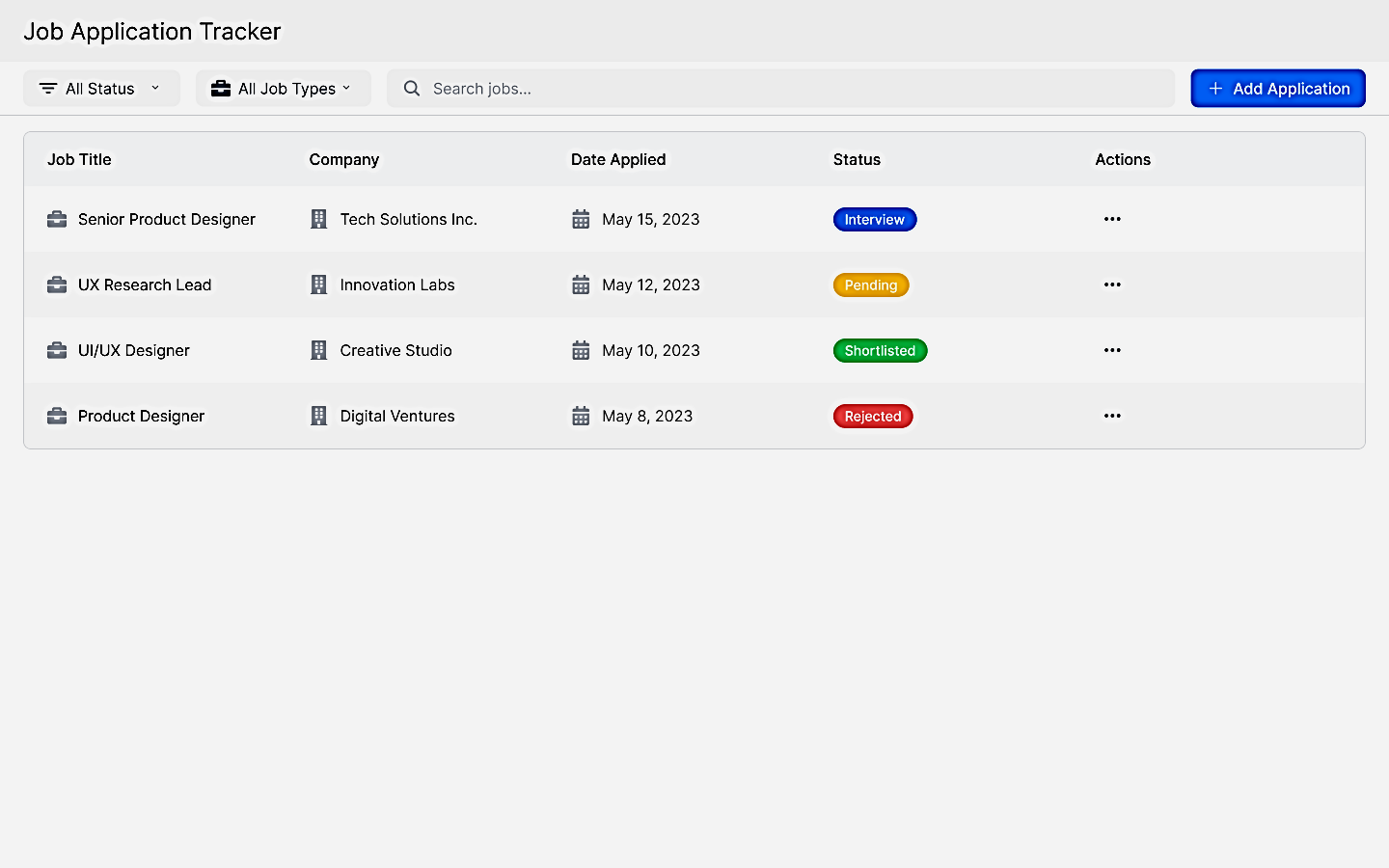
**Shortlisted Candidate And Chatbot For Recruiter:**



**Recruiter Post A Job:**

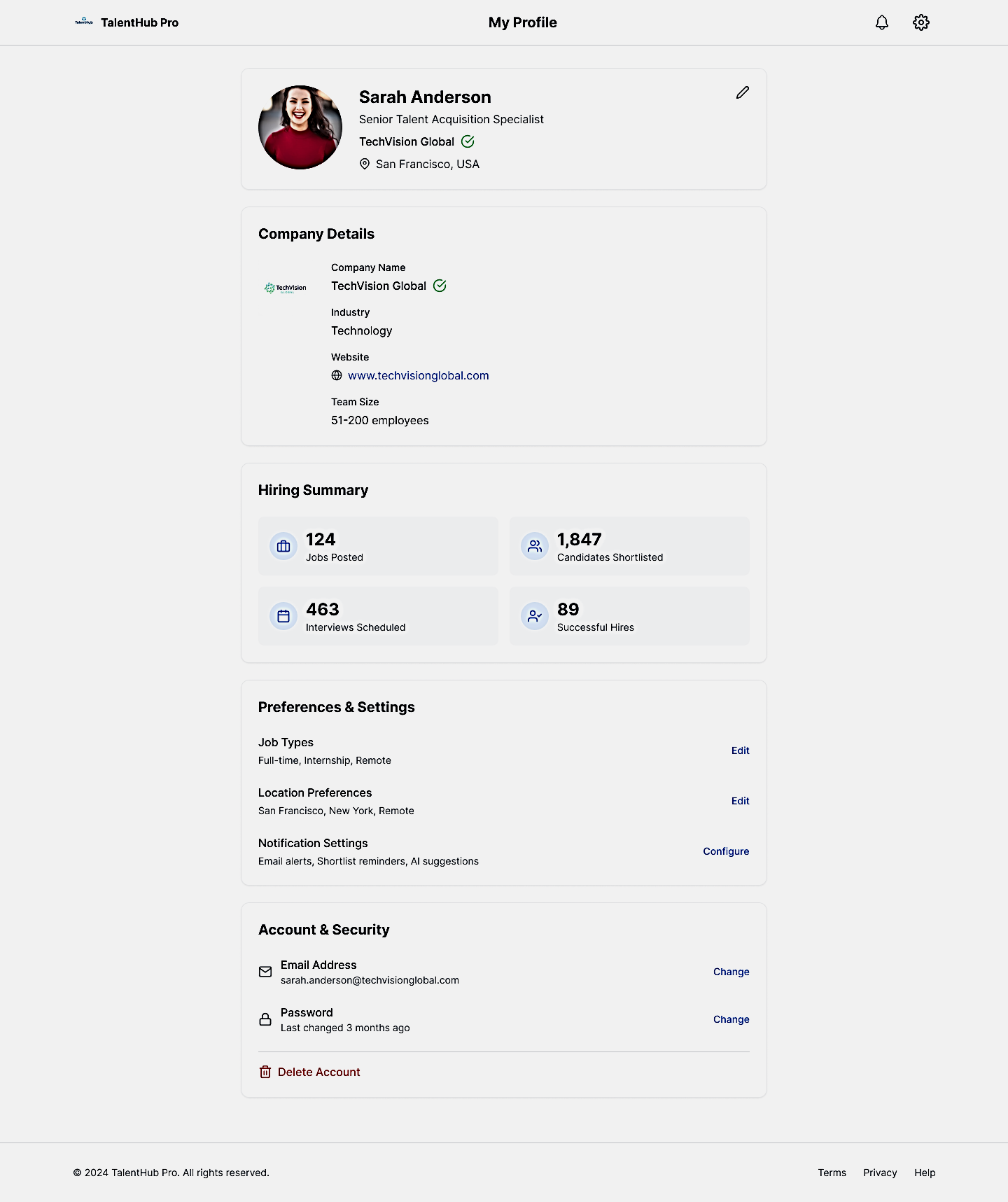


**Application Tracker:**



**Admin Profile And Management:**



**Recruiter Profile:**

## Additional Information

* All screens are responsive and mobile-friendly.
* Built-in validation and tooltips improve usability.
* System uses role-based access to dynamically adjust visible options.

# Other Design Details

### ****5.1 Role-Based Access Control****

* **Admin**:
  + Full platform access.
  + Manage users (Candidates, Recruiters).
  + View system logs and analytics.
  + Handle complaints (if module is included).
  + Monitor platform usage and activity.
* **Recruiter**:
  + Post jobs and manage listings.
  + Search candidates using filters (experience, education, skills).
  + Shortlist candidates and download their CVs.
  + Update candidate status and interact via chatbot for assistance.
* **Candidate**:
  + Register/login and manage their profile.
  + Upload CV, input education/experience/skills.
  + Apply for jobs.
  + Receive job matches, skill suggestions, and AI-generated interview questions.
  + Chat with the virtual assistant (chatbot) for help.

.

### ****5.2 Security Measures****

* **Local Authentication**:
  + Secure login via email and password using local **Node.js authentication** (bcrypt for password hashing).
  + Role-based route protection through middleware in Express.js.
* **Data Encryption**:
  + AES-256 encryption used for storing sensitive data like CVs, profile content, and application notes.
* **Session Handling**:
  + JWT (JSON Web Tokens) for secure session management.
  + Tokens expire after a set period to prevent session hijacking.
* **Local Backups**:
  + SQL or Firebase database is backed up locally (or optionally to external drive/cloud) once per day via automated scripts.

### ****5.3 Scalability Features****

 **SQL-Based Filtering:**

* Candidate-job matching is handled via optimized SQL or Firebase queries using indexes on key fields like skills, experience, and location.

 **Schema Design**:

* SQL or Firebase collections (users, jobs, applications, etc.) are well-structured and normalized where needed to avoid redundancy.

 **Modular Architecture**:

* Clear separation of frontend, backend, database, and AI services makes scaling easy by decoupling components.

 **Future-Ready Deployment**:

* While currently run locally, the architecture supports future deployment on Vercel, Render, or any cloud-based Node.js host.

### ****5.4 Future Enhancements****

* **AI-Based Candidate Ranking**:  
  Scoring algorithm to prioritize candidate profiles based on recruiter requirements and match percentage.
* **Multilingual UI Support**:  
  Add Urdu and other languages for regional accessibility.
* **Flutter-Based Mobile App**:  
  A cross-platform mobile app for recruiters and candidates to interact with TalentHub Pro from their phones.
* **Video Interview Module**:  
  Allow recruiters to send interview questions and get video responses from candidates asynchronously.
* **Skill Quiz Module**:  
  Built-in assessments (MCQs or task-based) to evaluate candidates before shortlisting.

# Test Specification and Results

## Test Case Specification:

**1.**

| **Identifier** | **TC-1** |
| --- | --- |
| Related requirements(s) | AI Candidate Filtering |
| Short description | Verify the system correctly filters candidates based on required skills. |
| Pre-condition(s) | Recruiter must be logged in. Job Post created with mandatory skills. |
| Input data | Candidate profiles (one high-match, one low-match). |
| Detailed steps | 1. Recruiter runs AI filter for the job. 2. View filtered list. |
| Expected result(s) | Only the candidate matching all mandatory skills appears in the top tier. |
| Post-condition(s) | Filter results saved in the Recruiter Dashboard. |
| Actual result(s) | Filter correctly matched and prioritized the high-fit candidate. |
| **Test Case Result** | **Pass** |

**2.**

| **Identifier** | **TC-2** |
| --- | --- |
| Related requirements(s) | Structured CV Generation |
| Short description | Verify if a Job Seeker can successfully generate a CV based on their profile data. |
| Pre-condition(s) | Job Seeker must be logged in. Profile must be completed (Experience, Education, Skills). |
| Input data | Selection of a CV template/format. |
| Detailed steps | 1. Navigate to CV Generation module. 2. Select template. 3. Click 'Generate'. |
| Expected result(s) | A structured CV document (PDF/DOCX) is generated and ready for download. |
| Post-condition(s) | Generated CV is available in the user's document history. |
| Actual result(s) | CV generated accurately reflecting profile data in the selected format. |
| **Test Case Result** | **Pass** |

**3.**

| **Identifier** | **TC-3** |
| --- | --- |
| Related requirements(s) | Smart Chatbot Interaction |
| Short description | Verify the chatbot provides relevant, context-aware assistance for a query. |
| Pre-condition(s) | User (Job Seeker) must be logged in. |
| Input data | Text input: "How do I apply for the Senior Developer role?" |
| Detailed steps | 1. Open Chatbot interface. 2. Enter the query. |
| Expected result(s) | Chatbot uses NLP to provide the direct steps or a link for applying to the specified role. |
| Post-condition(s) | Interaction history is logged. |
| Actual result(s) | Chatbot provided correct, context-aware instructions for the specific job. |
| **Test Case Result** | **Pass** |

**4.**

| **Identifier** | **TC-4** |
| --- | --- |
| Related requirements(s) | Dynamic Interview Question Generation |
| Short description | Verify the system generates relevant questions based on a specific job description. |
| Pre-condition(s) | Recruiter must be logged in. Job Post must exist (e.g., 'Data Scientist'). |
| Input data | Job Post ID for the 'Data Scientist' role. |
| Detailed steps | 1. Recruiter goes to Interview Prep module. 2. Selects the Job Post ID. 3. Clicks 'Generate Questions'. |
| Expected result(s) | A list of technical and behavioral questions specific to Data Science is displayed. |
| Post-condition(s) | Generated questions are saved for recruiter use. |
| Actual result(s) | Questions generated were highly relevant and specific to the job requirements. |
| **Test Case Result** | **Pass** |

**5.**

| **Identifier** | **TC-5** |
| --- | --- |
| Related requirements(s) | Auto Profile Update / Data Security |
| Short description | Verify the system shows an error when attempting to update a profile with missing mandatory data. |
| Pre-condition(s) | Job Seeker must be logged in. |
| Input data | Profile submission with the mandatory 'Salary Expectations' field left blank. |
| Detailed steps | 1. Job Seeker goes to Profile Update. 2. Leaves mandatory fields blank. 3. Clicks 'Submit'. |
| Expected result(s) | System displays a validation error and prevents the submission until the field is populated. |
| Post-condition(s) | No profile update is saved. |
| Actual result(s) | System accepted the empty mandatory field and saved the profile, losing data integrity. |
| **Test Case Result** | **Fail** |

**6.**

| **Identifier** | **TC-6** |
| --- | --- |
| Related requirements(s) | User Role Control / Security |
| Short description | Verify a Job Seeker user cannot access the Recruiter's dedicated job posting module. |
| Pre-condition(s) | Job Seeker must be logged in. |
| Input data | Direct URL access attempt to the Recruiter's Job Posting path. |
| Detailed steps | 1. Job Seeker attempts to navigate directly to the Recruiter module URL. |
| Expected result(s) | System redirects the user to their respective Dashboard or shows a clear 'Access Denied' error. |
| Post-condition(s) | User remains on an authorized page. |
| Actual result(s) | The Job Seeker was able to view and partially interact with the Recruiter's Job Posting form fields. |
| **Test Case Result** | **Fail** |

**7.**

| **Identifier** | **TC-7** |
| --- | --- |
| Related requirements(s) | Real-time Availability Tracking |
| Short description | Verify the system automatically updates a candidate's availability status when their interview slots are full. |
| Pre-condition(s) | Recruiter viewing candidate list. Candidate status is 'Available'. |
| Input data | Recruiter sends 5 simultaneous interview requests to the candidate. |
| Detailed steps | 1. Recruiter sends multiple booking requests to the candidate. 2. Refresh candidate list view. |
| Expected result(s) | Candidate status is automatically updated to 'Busy' or 'Limited Availability' to prevent overbooking. |
| Post-condition(s) | Status update is logged in the system. |
| Actual result(s) | Status did not update; candidate remained 'Available', leading to potential double-booking issues. |
| **Test Case Result** | **Fail** |

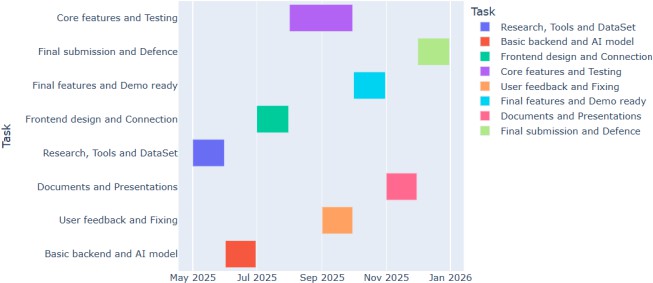
**8.**

| **Identifier** | **TC-8** |
| --- | --- |
| Related requirements(s) | User Login / Error Handling |
| Short description | Verify system displays a clear, secure error message for invalid login credentials. |
| Pre-condition(s) | User is on the login page. |
| Input data | Invalid Username and Invalid Password. |
| Detailed steps | 1. Go to Login Page. 2. Enter invalid credentials. 3. Click 'Login'. |
| Expected result(s) | System displays a secure and clear error: "Invalid Username or Password." |
| Post-condition(s) | User remains on the login page. |
| Actual result(s) | System returned a technical backend error code (e.g., '401 Unauthorized') instead of a user-friendly message. |
| **Test Case Result** | **Fail** |

## Summary of Test Results:

| **Module Name** | **Test Cases Run** | **Number of Defects Found** | **Number of Defects Corrected So Far** | **Number of Defects Still Need to be Corrected** |
| --- | --- | --- | --- | --- |
| **Module 1 (DB Candidate Filtering)** | TC-1 | 0 | 0 | 0 |
| **Module 2 (Structured CV Generation)** | TC-2 | 0 | 0 | 0 |
| **Module 3 (Smart Chatbot/NLP)** | TC-3 | 0 | 0 | 0 |
| **Module 4 (Dynamic Interview Generation)** | TC-4 | 0 | 0 | 0 |
| **Module 5 (Auto Profile Update)** | TC-5 | 1 | 0 | 1 |
| **Module 6 (Role-based Access Control)** | TC-6 | 1 | 0 | 1 |
| **Module 7 (Availability Tracking)** | TC-7 | 1 | 0 | 1 |
| **Module 8 (Login/System Errors)** | TC-8 | 1 | 0 | 1 |
| **Complete System** | TC-1 to TC-8 | **4** | **0** | **4** |

# Revised Project Plan



**Project Completion Status:**

| **Module Name** | **Status (Complete, Partially Implemented, Not Implemented)** |
| --- | --- |
| **Module 1 (User Interface & Login/Registration)** | Complete |
| **Module 2 (AI Candidate Filtering & Matching)** | Partially Implemented |
| **Module 3 (Structured CV Generation)** | Complete |
| **Module 4 (Smart Chatbot & NLP Integration)** | Partially Implemented |
| **Module 5 (Dynamic Interview Question Generation)** | Not Implemented |
| **Module 6 (Personalized Skill Recommendations)** | Not Implemented |
| **Complete System** | Partially Implemented |

# References

1. **React.js Documentation** – <https://react.dev>
2. **Node.js Docs** – <https://nodejs.org/en/docs>
3. **Express.js Framework** – <https://expressjs.com>
4. **SQL or Firebase Docs** – [https://www.SQL or Firebase.com/docs](https://www.mongodb.com/docs)
5. **JWT Authentication** – <https://jwt.io>
6. **Cloudinary (File Storage)** – <https://cloudinary.com>
7. **scikit-learn (ML Library)** – <https://scikit-learn.org>
8. **Rozee.pk Job Portal** – <https://www.rozee.pk>

Appendix A: Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **SRS** | Software Requirements Specification – A document that outlines system needs. |
| **Talent Hub Pro** | The name of the AI-based recruitment platform developed in this project. |
| **Recruiter** | A user who posts jobs, searches candidates, and manages the hiring  process. |
| **Candidate** | A user who applies for jobs and receives skill recommendations. |
| **AI (Artificial Intelligence)** | The simulation of human intelligence in machines to automate decision-making tasks. |
| **Skill Recommendation**  **Engine** | An AI tool that suggests relevant skills based on job roles or  resumes. |
| **Interview Question Generator** | A system feature that uses AI to create job-specific interview questions. |
| **Chatbot** | An automated assistant that answers FAQs and helps users navigate the platform. |
| **MERN Stack** | A development stack made up of SQL or Firebase, Express.js, React.js, and Node.js. |
| **Open AI API** | A service that provides AI capabilities like text generation for apps. |
| **SQL or Firebase** | A NoSQL database used to store user profiles, jobs, and system data. |
| **React.js** | A frontend JavaScript library for building user interfaces. |
| **Node.js** | A backend JavaScript runtime for server-side programming. |
| **HTTPS** | A secure version of HTTP that encrypts data between client and server. |
| **Role-Based Access Control (RBAC)** | A security model where users get permissions based on their role (e.g., Recruiter or Candidate). |

Appendix B: IV & V Report

**(Independent verification & validation)**

**IV & V Resource**

Name Signature

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S#** | **Defect Description** | **Origin Stage** | **Status** | **Fix Time** | |
| **Hours** | **Minutes** |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| … |  |  |  |  |  |

**Table 1: List of non-trivial defects**

This document has been adapted from the following:

1. Previous project templates at UCP
2. High-level Technical Design, Centers for Medicare & Medicaid Services. (www.cms.gov)