# **Project Name: Talent Hub - A Portal for Referral**

**Business Goal:** The goal of Talent Hub is to streamline the candidate selection process for companies while providing candidates with personalized recommendations and learning opportunities to improve their skill sets.

**Overall System:** Talent Hub is a web-based application portal that facilitates the registration of candidates, allows them to input their details manually or upload resumes, matches them with suitable job openings based on their profiles and skill sets, conducts initial testing rounds, and provides personalized learning recommendations for candidates who fail to clear the testing rounds.

**System Components:**

1. **User Authentication and Access Control:** Manage user authentication and authorization to ensure secure access to the portal for candidates and authorized company personnel.
2. **Candidate Registration and Profile Management:** Enable candidates to register, input their personal details, and manage their profiles.
3. **Resume Parsing:** Implement a component to parse resumes uploaded by candidates and extract relevant information.
4. **Job Matching Algorithm:** Develop an algorithm to match candidates with job openings based on their profiles and skill sets.
5. **Testing Module:** Design a module to conduct initial testing rounds for candidates.
6. **Learning Recommendation Engine:** Implement an AI-based recommendation engine to suggest learning programs or courses for candidates based on their areas of improvement.
7. **Database Management:** Manage a centralized database to store candidate information, job openings, test results, and learning recommendations securely.
8. **Scalability and Performance:** Implement measures to ensure the system can scale up or down based on demand and maintain optimal performance.

**Sub-Components:**

1. **Frontend:** User interface for candidate registration, profile management, job search, and test-taking.
2. **Backend:** Logic for job matching, testing, learning recommendation, and database management.
3. **AI Module:** Machine learning models for resume parsing, job matching, and learning recommendations.
4. **Database:** Centralized storage for candidate and job data.
5. **Security Module:** Ensure secure access to the portal and data encryption to protect sensitive information.

**AI-Based Prediction Modeling Scenarios:**

1. **Resume Parsing:** Use natural language processing (NLP) to extract key information from resumes, such as skills, experience, and education.
2. **Job Matching:** Employ machine learning algorithms to match candidate profiles with job requirements based on skill similarity and experience.
3. **Learning Recommendations:** Utilize collaborative filtering or content-based filtering to recommend learning programs or courses tailored to the candidate's skill gaps.

**Possible Use Cases:**

1. Candidate Registration and Profile Creation
2. Resume Uploading and Parsing
3. Job Search and Matching
4. Testing Rounds
5. Learning Recommendations

**Possible Limitations:**

1. **Accuracy of Resume Parsing:** The accuracy of resume parsing may vary depending on the complexity and formatting of resumes.
2. **Job Matching Accuracy:** The accuracy of job matching depends on the effectiveness of the algorithm and the quality of candidate profiles and job descriptions.
3. **Learning Recommendations:** Recommendations may not always perfectly align with the candidate's learning needs and preferences.

**Preferred Technology Stack (Open Source, Windows Environment):**

* **Frontend:** React.js
* **Backend:** Node.js with Express.js
* **Database:** PostgreSQL
* **AI/ML:** Python with libraries like TensorFlow or PyTorch for modeling, NLTK or spaCy for NLP tasks
* **Security:** JSON Web Tokens (JWT) for authentication, HTTPS for secure communication
* **Scalability:** Docker for containerization, Kubernetes for orchestration

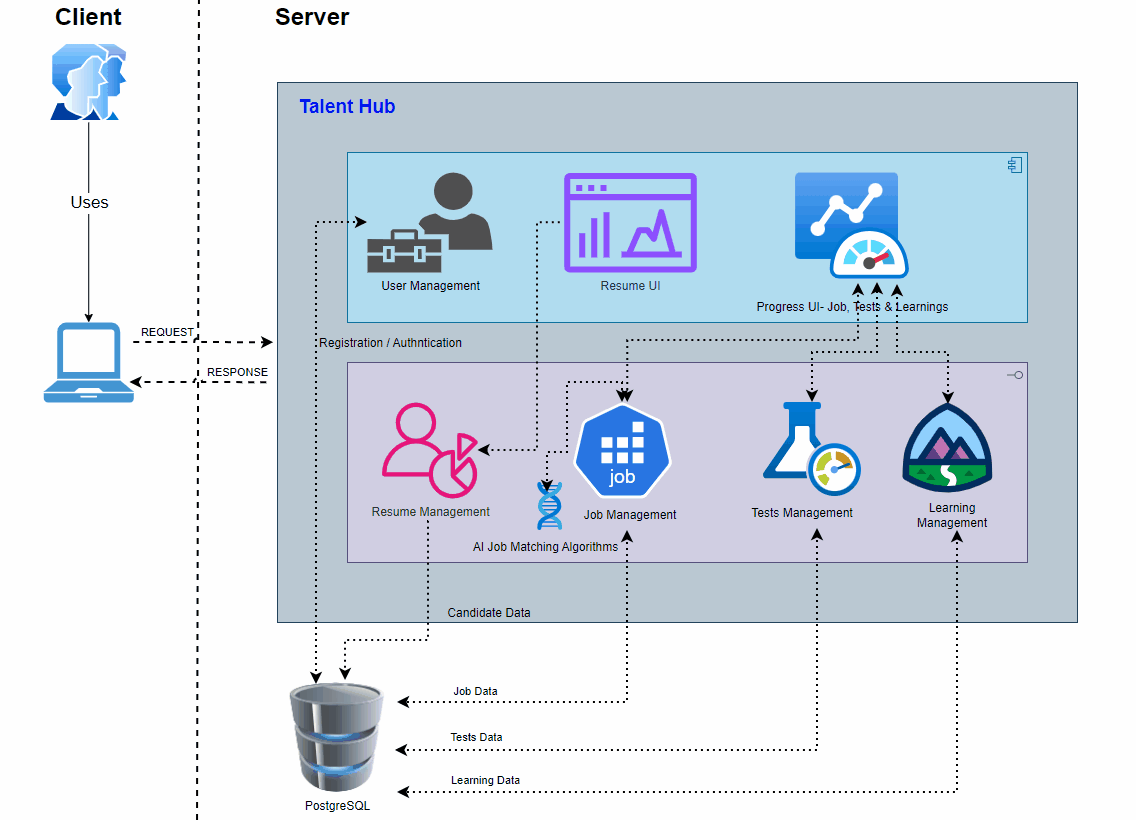
**Tentative Timeline: (9-10 months)**

1. **Requirement Gathering and Analysis:** 1 month.
2. **Design and Architecture:** 1.5 months.
3. **Frontend and Backend Development:** 3 months.
4. **AI/ML Model Development:** 2 months.
5. **Testing and Quality Assurance:** 1.5 months.
6. **Deployment and Launch:** 1 month.

# **Possible Architecture and Design Patterns:**

|  |  |  |
| --- | --- | --- |
| Component | Architecture | Design Pattern |
| User Authentication & Access Control | Client-Server Architecture | OAuth 2.0 |
| Candidate Registration & Profile Management | MVC Architecture | Singleton |
| Resume Parsing | Microservices Architecture | Adapter |
| Job Matching Algorithm | Service-Oriented Architecture | Strategy |
| Testing Module | Layered Architecture | Observer |
| Learning Recommendation Engine | Event-Driven Architecture | Factory |
| Database Management | Database Sharding | Repository |
| Scalability & Performance | Cloud-Native Architecture | Caching with Redis |
| Frontend | Component-Based Architect. | Flux |
| Backend | Hexagonal Architecture | Dependency Injection |
| AI Module | Event-Driven Microservices | Chain of Responsibility |
| Security Module | Defense-in-Depth Security | Builder |
| Overall System | Microservices with Kubernetes | Circuit Breaker |

# **Systemic View:**



# **DB Tables:**

|  |  |  |
| --- | --- | --- |
| Table Name | Columns | Relationships |
| Users | user\_id (PK), email, password, user\_type | NA |
| Candidate Profile | candidate\_id (PK), user\_id (FK), name, education, experience, skills | One-to-One with Users |
| Company Profile | company\_id (PK), user\_id (FK), name, industry, size, description | One-to-One with Users |
| Resumes | resume\_id (PK), candidate\_id (FK), resume\_file\_path, parsed\_information | One-to-Many with Candidate Profile |
| Jobs | job\_id (PK), company\_id (FK), title, description, requirements, location, salary | One-to-Many with Company Profile |
| Tests | test\_id (PK), job\_id (FK), test\_type, duration, difficulty\_level | One-to-Many with Jobs |
| Test Results | result\_id (PK), test\_id (FK), candidate\_id (FK), score, status | One-to-Many with Tests |
| Learning Programs | program\_id (PK), candidate\_id (FK), program\_name, provider, duration, description | One-to-Many with Candidate Profile |
| UserSessions | session\_id (PK), user\_id (FK), expiry\_time | NA |
| UserRole | role\_id (PK), role\_name | NA |

# **High-Level User Stories (Epics):**

|  |  |  |
| --- | --- | --- |
| User Story ID | User Story Title | User Story Description |
| 1 | Candidate Registration and Profile Creation | As a candidate, I want to register on the platform and create a profile so that I can apply for job openings. |
| 2 | Resume Uploading and Parsing | As a candidate, I want to upload my resume, and have it parsed so that my skills and experience can be added to my profile. |
| 3 | Job Search and Matching | As a candidate, I want to search for job openings and see relevant job matches based on my profile. |
| 4 | Testing Rounds | As a candidate, I want to participate in testing rounds to demonstrate my skills to potential employers. |
| 5 | Learning Recommendations | As a candidate, I want to receive personalized learning recommendations to improve my skills and increase my chances of getting hired. |

# **Low-Level User Stories (Functionality):**

## **1. Candidate Registration and Profile Creation:**

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Story Description | Acceptance Criteria | Size |
| 1.1(FR) | As a new candidate, I want to register on the platform with my email and password. | User can successfully register with a valid email and password. | Small |
| 1.2 (FR) | As a registered candidate, I want to fill in my profile details (e.g., education, experience, skills). | User can successfully fill in their profile details. | Medium |
| 1.3 (NFR) | As a candidate, I want my password to be securely stored (e.g., hashed) to protect my account. | Passwords are securely stored using a secure hashing algorithm. | Small |
| 1.4 (NFR) | As a candidate, I want the registration process to be fast and responsive. | Registration process completes within <#> minute. | Small |

## **2. Resume Uploading and Parsing:**

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Story Description | Acceptance Criteria | Size |
| 2.1 | As a candidate, I want to upload my resume in PDF or document format. | User can successfully upload a resume in PDF or document format. | Small |
| 2.2 | As a candidate, I want to see the parsed information from my resume (e.g., skills, experience). | Parsed information is displayed accurately for user review. | Medium |
| 2.3 (NFR) | As a candidate, I want the resume parsing process to be fast, even for large resumes. | Resume parsing completes within <#> seconds for a <#>-page resume. | Small |
| 2.4 (NFR) | As a candidate, I want clear feedback on the parsing process. | User receives a confirmation message after successful parsing. | Small |

## **3. Job Search and Matching:**

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Story Description | Acceptance Criteria | Size |
| 3.1 | As a candidate, I want to search for job openings based on criteria such as location, industry, and job type. | User can search for job openings based on specified criteria. | Small |
| 3.2 | As a candidate, I want to see job openings that match my profile and skills. | Search results include job openings that match the user's profile and skills. | Medium |
| 3.3 (NFR) | As a candidate, I want the search results to be displayed quickly. | Search results are displayed within <#> seconds. | Small |
| 3.4 (NFR) | As a candidate, I want the job descriptions to be displayed accurately and clearly. | Job descriptions are displayed accurately and clearly. | Small |

## **4. Testing Rounds:**

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Story Description | Acceptance Criteria | Size |
| 4.1 | As a candidate, I want to access and complete testing rounds. | User can access and complete testing rounds. | Medium |
| 4.2 | As a candidate, I want to receive feedback on my test results. | Test results are evaluated, and feedback is provided to the user. | Small |
| 4.3 (NFR) | As a candidate, I want the testing platform to be reliable and available during scheduled testing times. | Testing platform is available 99.9% of the time during scheduled testing times. | Small |
| 4.4 (NFR) | As a candidate, I want the testing platform to be able to handle multiple concurrent users. | Testing platform can handle <#> concurrent users. | Medium |

## **5. Learning Recommendations:**

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Story Description | Acceptance Criteria | Size |
| 5.1 | As a candidate, I want to view personalized learning recommendations. | User can view personalized learning recommendations. | Small |
| 5.2 | As a candidate, I want to enroll in recommended programs. | User can enroll in recommended programs. | Small |
| 5.3 (NFR) | As a candidate, I want the recommendations to be relevant and aligned with my skills and areas for improvement. | Recommendations are relevant and aligned with the user's skills and areas for improvement. | Small |
| 5.4 (NFR) | As a candidate, I want the recommendations to be presented clearly and be easy to understand. | Recommendations are presented clearly and are easy to understand. | Small |

Hello, this is Copilot, an AI companion. I can help you with your project by providing some information and suggestions based on your requirements. Here is a possible outline of a professional proposal for your application portal:

**Title: Jobify - A Smart Application Portal for Job Candidates**

**Business goal: To create a user-friendly and secure web portal that connects job seekers with relevant job opportunities and provides them with feedback and guidance throughout the hiring process.**

**Solution overview: Jobify is a web portal that allows candidates to register and create their profiles by filling in their details manually or uploading their resumes in PDF or DOC format. The portal uses natural language processing (NLP) and machine learning (ML) techniques to parse the resumes and extract the skills, education, experience, and other relevant information of the candidates. The portal then matches the candidates with the available job openings based on their profile and skill set. The portal also provides the candidates with an initial testing round to assess their competencies and suitability for the job. The testing round consists of multiple-choice questions, coding challenges, or video interviews, depending on the job role and level. The portal uses artificial intelligence (AI) to evaluate the candidates’ responses and provide them with feedback and scores. The portal also guides the candidates with the next steps of the hiring process, such as scheduling interviews, negotiating salaries, or signing contracts. The portal also offers the candidates a list of programs or courses to improve their skills and competencies in the areas where they are lacking. The portal is accessible only to the candidates and the company running the portal. The portal ensures secure access, data protection, scalability, and performance by using encryption, authentication, cloud computing, and load balancing technologies.**

**System components and subcomponents: The portal consists of the following main components and subcomponents:**

* **Front-end:** The front-end is the user interface of the portal that interacts with the candidates and displays the information and features of the portal. The front-end is developed using HTML, CSS, and JavaScript frameworks, such as React or Angular.
* **Back-end:** The back-end is the server-side of the portal that handles the business logic, data processing, and communication with the database and the AI models. The back-end is developed using Python or Java programming languages and Django or Spring Boot frameworks.
* **Database:** The database is the storage system of the portal that stores the data of the candidates, the jobs, the tests, and the feedback. The database is implemented using MongoDB or PostgreSQL database management systems.
* **AI models:** The AI models are the core of the portal that provide the functionalities of resume parsing, job matching, testing, evaluation, and feedback. The AI models are developed using Python or R programming languages and TensorFlow or PyTorch libraries.
* **Cloud services:** The cloud services are the platforms that host the portal and provide the resources and tools for deployment, security, scalability, and performance. The cloud services are provided by Microsoft Azure or Amazon Web Services.

**AI-based prediction modeling scenarios: The portal uses the following AI-based prediction modeling scenarios to provide its features:**

* **Resume parsing:** The portal uses NLP techniques, such as named entity recognition, keyword extraction, and document classification, to parse the resumes and extract the relevant information of the candidates, such as skills, education, experience, and achievements. The portal then converts the extracted information into a structured format that can be stored in the database and used for job matching and testing.
* **Job matching:** The portal uses ML techniques, such as cosine similarity, k-nearest neighbors, and collaborative filtering, to match the candidates with the available job openings based on their profile and skill set. The portal then ranks the job openings according to the degree of match and displays them to the candidates.
* **Testing:** The portal uses AI techniques, such as natural language generation, computer vision, and speech recognition, to generate and administer the testing round for the candidates. The testing round consists of multiple-choice questions, coding challenges, or video interviews, depending on the job role and level. The portal then collects the candidates’ responses and passes them to the evaluation model.
* **Evaluation:** The portal uses AI techniques, such as natural language understanding, sentiment analysis, and facial expression recognition, to evaluate the candidates’ responses and provide them with feedback and scores. The evaluation model considers the correctness, completeness, and quality of the responses, as well as the candidates’ confidence, attitude, and personality. The portal then stores the feedback and scores in the database and displays them to the candidates.
* **Feedback:** The portal uses AI techniques, such as natural language generation, recommender systems, and reinforcement learning, to provide the candidates with feedback and guidance throughout the hiring process. The feedback model considers the candidates’ performance, preferences, and goals, and provides them with personalized and constructive feedback, such as strengths, weaknesses, areas of improvement, and tips. The feedback model also guides the candidates with the next steps of the hiring process, such as scheduling interviews, negotiating salaries, or signing contracts. The feedback model also offers the candidates a list of programs or courses to improve their skills and competencies in the areas where they are lacking.

**Possible use cases: The portal supports the following possible use cases for the candidates:**

* **Register and create profile:** The candidate can register and create their profile by filling in their details manually or uploading their resume in PDF or DOC format. The portal then parses the resume and extracts the relevant information of the candidate.
* **Browse and apply for jobs:** The candidate can browse and apply for the available job openings that match their profile and skill set. The portal then matches the candidate with the job openings and ranks them according to the degree of match.
* **Take the testing round:** The candidate can take the testing round to assess their competencies and suitability for the job. The portal then generates and administers the testing round for the candidate and collects their responses.
* **Receive feedback and scores:** The candidate can receive feedback and scores on their performance in the testing round. The portal then evaluates the candidate’s responses and provides them with feedback and scores.
* **Follow the next steps:** The candidate can follow the next steps of the hiring process, such as scheduling interviews, negotiating salaries, or signing contracts. The portal then guides the candidate with the next steps and provides them with tips and resources.
* **Improve skills and competencies:** The candidate can improve their skills and competencies in the areas where they are lacking. The portal then offers the candidate a list of programs or courses to improve their skills and competencies.

**Possible limitations: The portal may face the following possible limitations:**

* **Data quality and availability:** The portal relies on the quality and availability of the data of the candidates, the jobs, the tests, and the feedback. The portal may encounter issues such as incomplete, inaccurate, outdated, or inconsistent data, which may affect the performance and accuracy of the portal and the AI models.
* **AI model complexity and explainability:** The portal uses complex and sophisticated AI models to provide its features. The portal may encounter issues such as overfitting, underfitting, bias, or errors, which may affect the reliability and validity of the portal and the AI models. The portal may also face challenges in explaining the logic and reasoning behind the AI models and their outputs, which may affect the transparency and trustworthiness of the portal and the AI models.
* **User privacy and security:** The portal handles sensitive and personal data of the candidates, such as their resumes, responses, feedback, and scores. The portal may encounter issues such as data breaches, unauthorized access, or misuse, which may affect the privacy and security of the portal and the users.

**Possible technology stack: The portal uses the following possible technology stack:**

* Front-end: HTML, CSS, JavaScript, React, or Angular
* Back-end: Python or Java, Django or Spring Boot
* Database: MongoDB or PostgreSQL
* AI models: Python or R, TensorFlow or PyTorch
* Cloud services: Microsoft Azure or Amazon Web Services

**Tentative timeline: The portal can be developed and deployed in the following tentative timeline:**

* Phase 1: Planning and design (4 weeks)
* Phase 2: Development and testing (12 weeks)
* Phase 3: Deployment and maintenance (4 weeks)

I hope this helps you with your project. If you have any questions or feedback, please let me know. 😊