MICROSOFT ENGAGE-WIT HACKATHON 2021

Job Search App

Submitted by:

Satashree Roy, Department of CSE, NIT Silchar

Problem Statement:

To create a functional prototype for a web based application that allows Blue-collared workers to search for jobs, and renders equally well on Microsoft Edge, Google Chrome, Mozilla Firefox, and Apple Safari.

Solution:

A job search application bridges the gap between employers and job-seekers. Recruiters can advertise their openings and examine the resumes and work experience of potential employees, while candidates can search and apply for jobs posted.

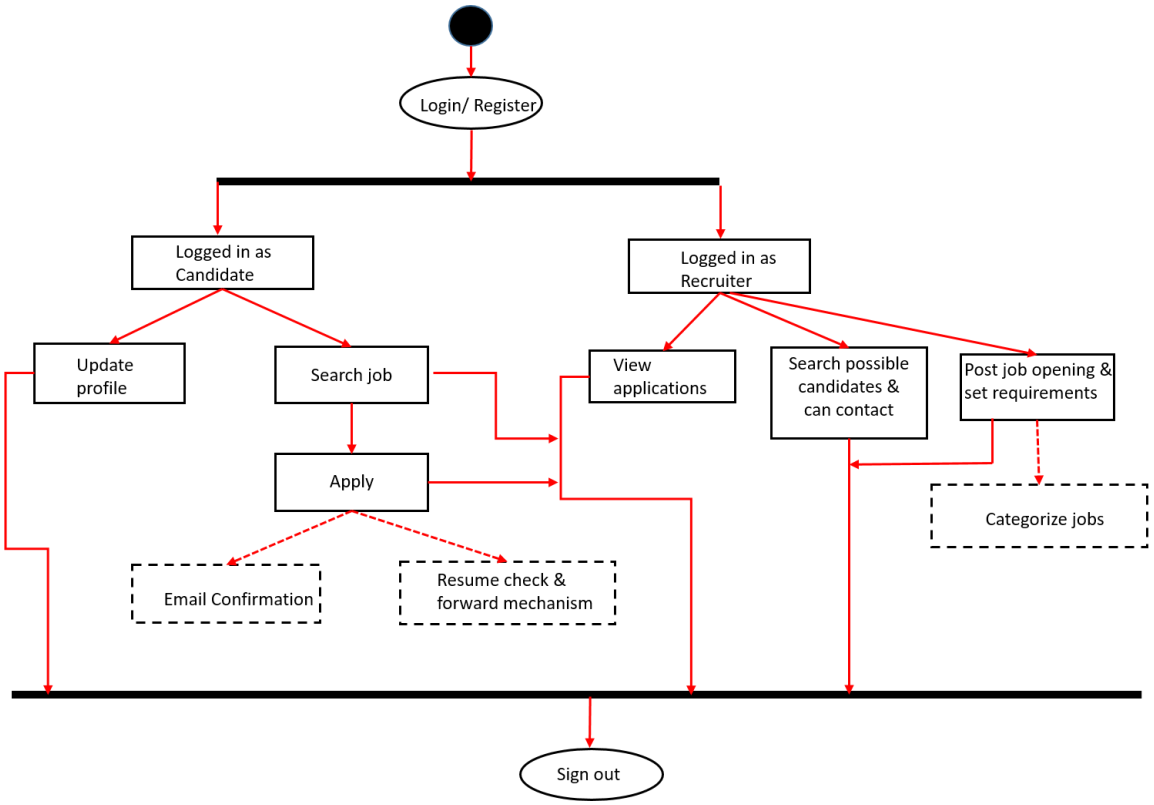
The solution proposed is a web based application that caters to two kinds of users: Candidate and Recruiter. Candidates can build their profile, search for available jobs based on filters and apply. Recruiters can post job openings under the banner of their company and view the applications received. AI and automated tools can be used to make the search and candidate-match experience more effective.

Features:

- Primary: 1) Search, 2) Apply, 3) Post Job
- Simple and intuitive interface to register as Candidate or Recruiter, and create profile.
- Refined search based on filters such as job role, location, experience.
- Show Recruiters the candidates who have applied for the job posted.
- Show Recruiters the people who are actively looking for jobs and are nearby and be able to filter potential hires and contact themselves if necessary. Post new job openings and set the requirements.
- Automatic removal of outdated posts, which ensures reliability of the portal.
- AI (NLP) based resume screening mechanism to improve candidate-match for the Recruiter, as well as provide relevant job suggestions to the Candidate.

Use-Case Diagrams:

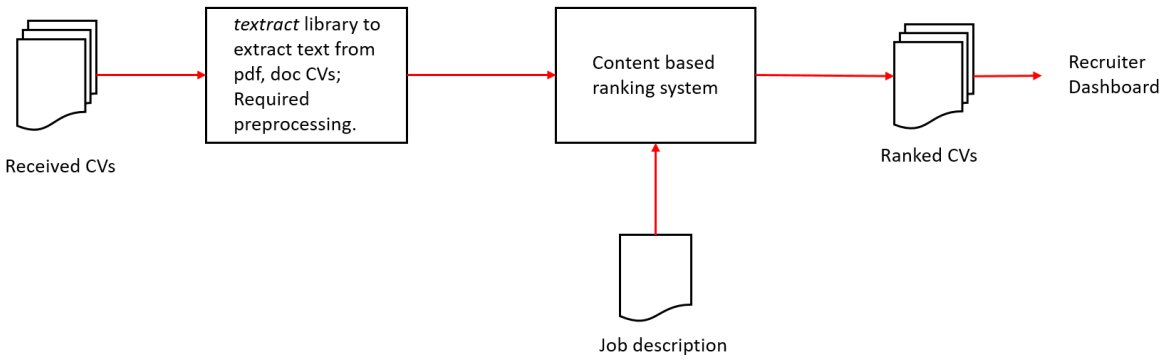
Activity Diagram:



AI (NLP) Based Resume Screening System:

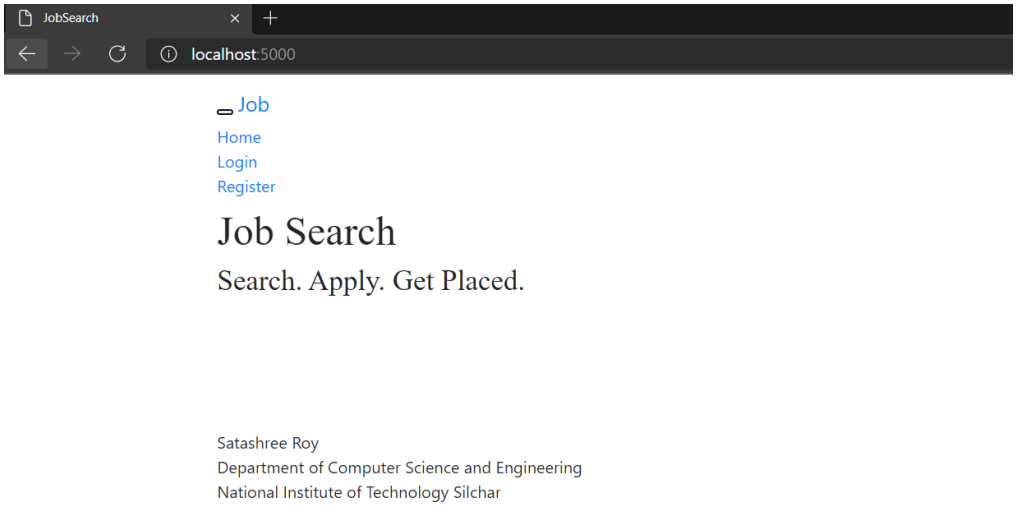
The job search application will be more convenient for the Recruiter if it can pick the right candidates from the pack. Moreover, CVs in the market do not have a universal format. A major challenge for recruiters is mapping the CV to the job description. Hence, an AI based tool to perform a first level screening of resumes can be useful. An example of such a system is a content based recommendation system that measures the similarity of the resumes and the job description using suitable metric such as cosine similarity and outputs top k (k being configurable) resumes to the recruiter (using k-NN algorithm for example). The similarity matching can be done in document-embedding space using doc2vec (which is an extension of word2vec).

Some drawbacks of using such an AI based tool is that the model takes huge amount of data to train to give good accuracy and the model may learn bias inherent in the train data.

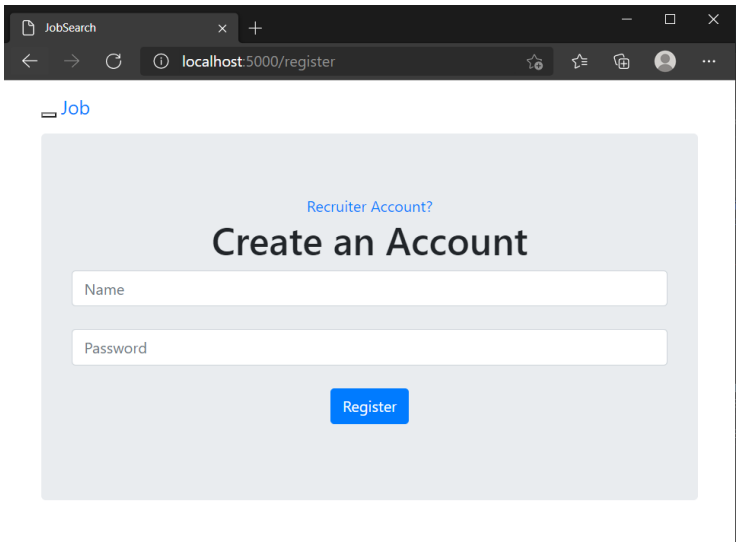


Snapshots:

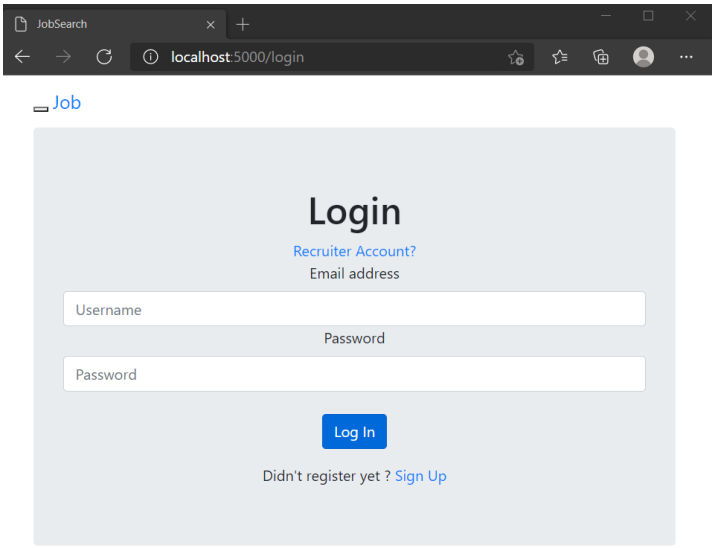
1. Home



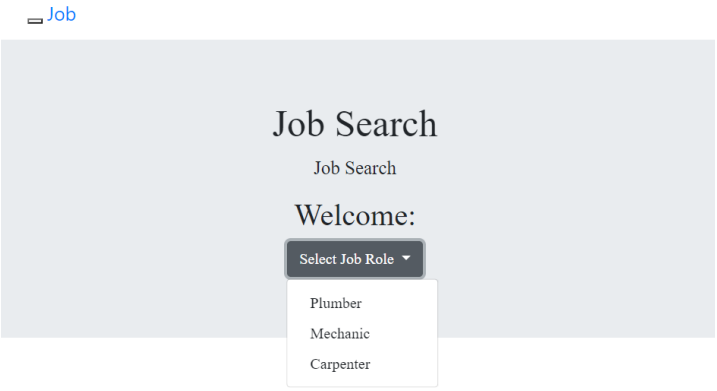
2. Register



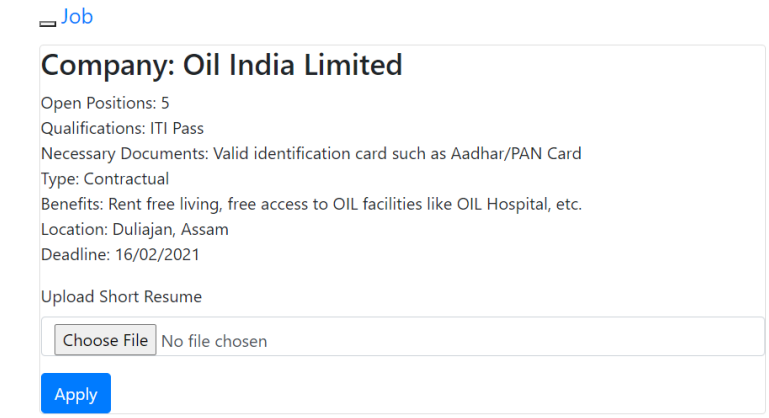
3. Login



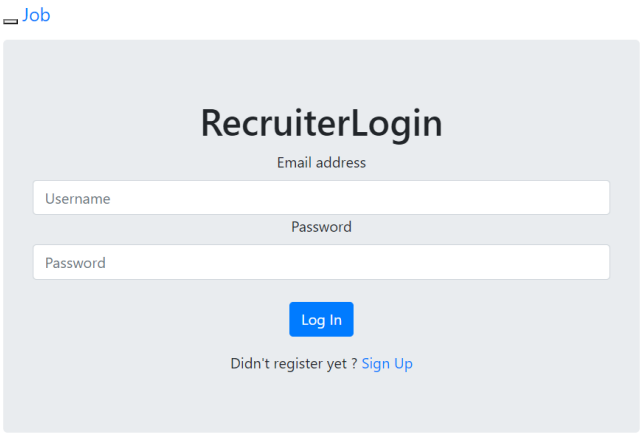
4. Candidate dashboard



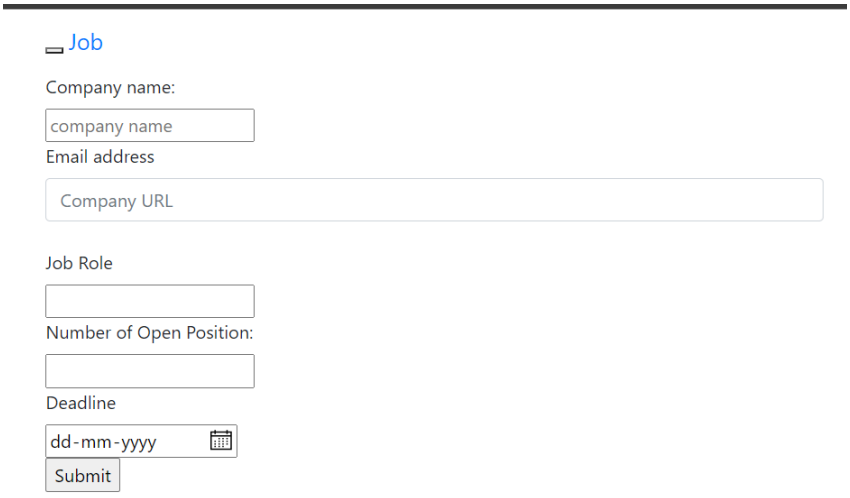
5. Apply



6. Recruiter login



7. Post new job



8. Database

```
> db.users.find().pretty()
{
        "_id" : ObjectId("602601aea1b06901e237a62c"),
        "name" : "rex@tex.com",
        "password" : "sha256$5QOyoHP9$4b7c9b819e4259b83458b666e99395062d4817d33219b656f0225741d4868dc0"
}
{
        "_id" : ObjectId("60268bb7a1b0690566d8e0a9"),
        "name" : "emp@abc.com",
        "password" : "sha256$5PyKc6xA$344b6b05ad70ca1355ac829c64231fa4b727f007adb595c52d4868916b2a4b14"
}
{
        "_id" : ObjectId("6026938fa1b0690646438226"),
        "name" : "sroy@abc.com",
        "password" : "sha256$VE0G4VuL$ecc7358de123d55ca9bca6a6c9ccd045db38a14e2bb601f07df0095aad81f2af"
}
>
```

```
> db.emp.find().pretty()
{
        "_id" : ObjectId("60268cf3a1b069059f53adcd"),
        "name" : "emp@abc.com",
        "password" : "sha256$w1QrXOHE$1649776cb083381cc6b969e2ea5876bb41efbc2e2cd6a745689ed293ddcdbb2b"
}
>
```