WolfJobs

The Ultimate Hub for On-Campus Student Employment

Group 99

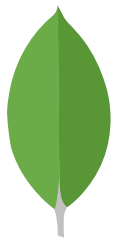
Archit, Bhavishya, Vyom

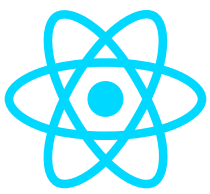
Introduction

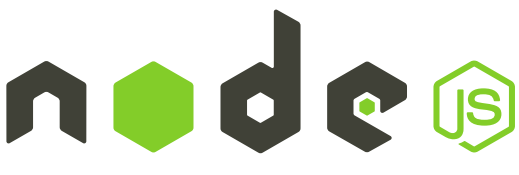
Wolfjobs aims to establish itself as the de-facto on campus employment portal. It streamlines the process right from creating a job posting to hiring the candidate that can help the organization in the best possible way.

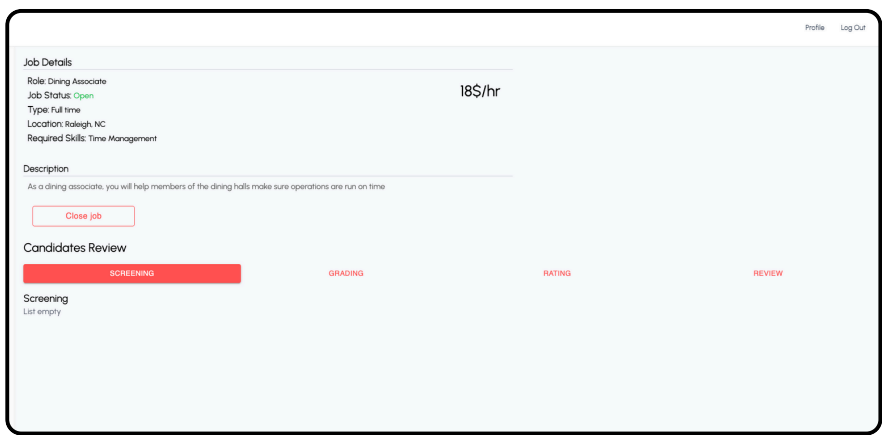
Tech Stack and Architecture

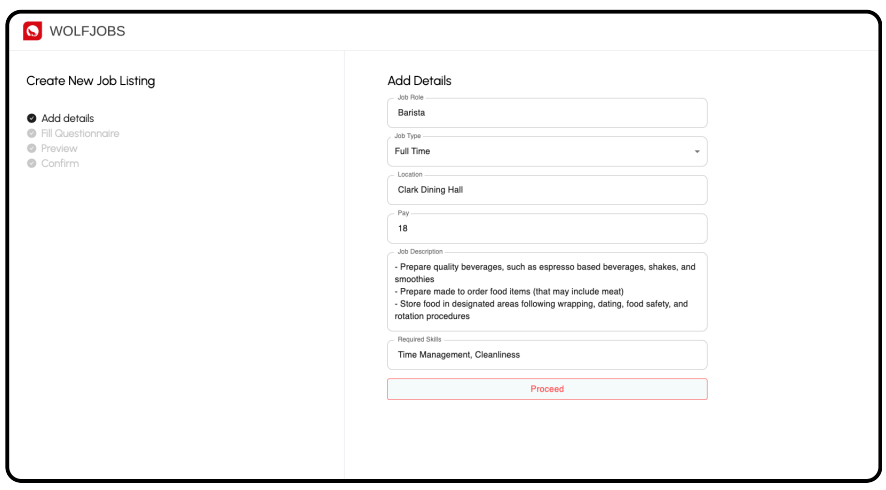
The project uses the very popular **MERN** Stack, i.e. MongoDB, Express, React and Node. The backend consists of an Express server running on Node and the API's are consumed by a React frontend











QR Codes

Github Repository

Live Demo

Notable Features

User Friendly Interface:

The portal follows a clear red and white branding and uniform design throughout the app making it easier for users to recognize actions and increases overall app accessibility

Job Listing Filters:

Searching for the perfect role for a potential candidate becomes a whole lot easier with this portal providing filters to sort by pay, city, etc.

Uniform Listing Format:

Writing out a job description can often be a tedious and long process. With WolfJobs, you can expect the same number of questions to fill out every time while also making the job requirements clear to fellow applicants

Ease of Application:

With a simple one-click application process, WolfJobs makes it easy for any applicant to apply for a relevant job. The listings also show a level of overall match to any applicants to let them know how much of a fit they would be for that particular role

Enhancements / Improvements

Added Filters

Although the portal offers filters in sorting order, it does not offer a way for users to view very specific listings. More filters can be added to get the listings for exact pay ranges, shift timings and more.

Implementation: The backend API can be modified to accept query parameters and filter jobs according to the requirements

Resume Parsing

The portal currently requires new users to type out their skills manually. A feature to parse PDF file formats can be added which would be able to extract relevant skills from the resume itself and associate them with the user profile.

Implementation: This could be a simple implementation with a third-party library like pdf2json for JS, or could also use LLMs to help parse the resume

Easier Contact

Currently the portal requires employers to manually check contact details and contact candidates. A simple contact button that can help reach out to candidate via email or to help schedule interviews could greatly increase usability of the portal

Implementation: Google's Calendar API could be used on the backend to schedule interview requests along with the native email button for regular communication

Enhanced Security and Authentication

On inspecting the code, the passwords are being stored in plaintext. Hashing the passwords and storing them securely as well as giving users the option to authenticate with social accounts will enhance the app

Implementation: bcrypt for password hashing and Google Auth