



JOB SEARCHING PORTAL

22AD901 – APP DEVELOPMENT

Submitted by

JOSHITHA 727722EUCS075

NIKILA B 727722EUCS119

KAVIYASREE S R 727722EUCS087

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE ENGINEERING

SRI KRISHNA COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution, Affiliated to Anna University Chennai - 600 025)

AUGUST 2024



SRI KRISHNA COLLEGE OF ENGINEERING AND TECHNOLOGY
(An Autonomous Institution. Affiliated to Anna University, Chennai)
Kuniamuthur, Coimbatore - 641 008



BONAFIDE CERTIFICATE

Certified that this project report titled “**JOB SEARCHING PORTAL**” is the bonafide work of **JOSHITHA M(727722EUCS075), KAVIYASREE S R(727722EUCS087) AND NIKILA B(727722EUCS119)** who carried out the project work under my supervision.

SIGNATURE

Dr. GRANTY REGINA ELWIN

HEAD OF THE DEPARTMENT

Professor
Computer Science and Engineering,
Sri Krishna College of Engineering and
Technology,
Kuniamuthur, Coimbatore-641008.

SIGNATURE

Mrs. N KOUSIKA

SUPERVISOR

Assistant Professor
Computer Science and Engineering,
Sri Krishna College of Engineering and
Technology,
Kuniamuthur, Coimbatore-641008.

Submitted for the Project viva-voce examination held on 13/08/2024

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

At this juncture, we take the opportunity to convey our sincere thanks and gratitude to the management of the college for providing all the facilities to us.

We wish to convey our gratitude to our college principal, **Dr.Porkumaran** for forwarding us to do our project and offering adequate duration to complete our project.

We would like to express our grateful thanks to **Dr.Granty Regina Elwin**, Head of the department, Department of Computer Science and Engineering for her encouragement and valuable guidance to this project.

We extend my gratitude to our beloved guide **Mrs, N Kousika**, Assistant Professor, Department of Computer Science and Engineering for her constant support and immense help at all stages of the project.

ABSTRACT

This project presents the development of an advanced job searching portal aimed at simplifying the recruitment process for job seekers and companies. The portal is built using React.js for the frontend and Spring Boot for the backend, offering a seamless and intuitive user experience. It includes three distinct user roles: Job Seeker, Company, and Admin, each with tailored functionalities to streamline job applications, postings, and management.

The Job seekers can browse and apply for jobs with ease, viewing detailed job descriptions, company information, and specific requirements. Companies can manage their job postings and view applications through a dedicated dashboard. The Admin role oversees the portal's operations, ensuring data integrity and user management.

—

The portal also features a responsive design, user authentication, and a dynamic job listing interface, making it a comprehensive solution for modern recruitment needs. The implementation of both frontend and backend technologies ensures robust performance, security, and scalability.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE No.
	ACKNOWLEDGEMENT	iii
	ABSTRACT	iv
	LIST OF TABLES	v
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1	INTRODUCTION	1
	1.1 OVERVIEW	1
	1.2 COMPONENTS OF SYSTEM	3
	1.3 ADVANCED TECHNOLOGIES	6
	1.4 GLOBAL PERSPECTIVES	8
2	SYSTEM ANALYSIS	10
	2.1 EXISTING SYSTEM	10
	2.1.1 DRAWBACKS	11
	2.2 PROBLEM DEFINITION	11
	2.3 PROPOSED SYSTEM	12
	2.3.1 ADVANTAGES	12
3	SYSTEM REQUIREMENTS	13
	3.1 HARDWARE REQUIREMENTS	13
	3.2 SOFTWARE REQUIREMENTS	13
	3.3 SOFTWARE DESCRIPTION	13
	3.3.1 FRONTEND	14
	3.3.2 BACKEND	16

4	SYSTEM DESIGN	18
	4.1 MODULE DESCRIPTION	18
	4.1.1 ADMIN MANAGEMENT	18
	4.1.2 JOBSEEKER MANAGEMENT	18
	4.1.3 COMPANY MANAGEMENT	19
	4.2 USE CASE DIAGRAM	20
	4.3 SEQUENCE DIAGRAM	20
	4.4 ER DIAGRAM	21
5	TESTING	22
	5.1 UNIT TESTING	22
	5.2 INTEGRATION TESTING	22
	5.3 SECURITY AND AUTHENTICATION	24
	5.4 TEST CASES	24
	5.4.1 TEST CASE I	24
	5.4.2 TEST CASE II	25
6	CONCLUSION AND FUTURE WORK	26
	6.1 CONCLUSION	26
	6.2 FUTURE WORK	26
7	APPENDICES	27
	APPENDIX I	
	SOURCE CODE	27
	APPENDIX II	
	SCREENSHOTS	47
	REFERENCES	55

LIST OF FIGURES

FIGURE No.	TITLE	PAGE No.
3.1	VS Code Logo	10
4.1	Use Case Diagram	20
4.2	Sequence Diagram	20
4.3	Data Flow Diagram	21
5.1	Login Page Diagram	23
5.2	Test case I	24
5.3	Test case II	25
A.2.1	Main page	47
A.2.2	Admin login	47
A.2.3	Admin Dashboard	48
A.2.4	User Login	49
A.2.5	User Registration	49
A.2.6	Joblist Page	50
A.2.7	Application Page	50
A.2.8	Company Dashboard	51
A.2.9	Contact Page	51

LIST OF ABBREVIATIONS

S. No	ABBREVIATIONS	EXPANSION
1	JWT	JSON Web Token
2	RAM	Random Access Memory
3	GB	Giga Bytes
4	VS	Visual Studio
5	OS	Operating System
6	HTTP	Hyper Text Transfer Protocol
7	JPA	Java Persistence API
8	API	Application Programming
9	JDBC	Java Database Connectivity
10	SQL	Sequential Query Language
11	UI	User Interface
12	DOM	Document Object Model
13	JSX	Java Script XML
14	OTP	One Time Password
15	UML	Unified Modelling Language
16	DFD	Data Flow Diagram
17	FAQ	Frequently Asked Questions

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The job searching portal is a dynamic and user-friendly platform designed to connect job seekers with potential employers efficiently. Built using React.js for the frontend and Spring Boot for the backend, the portal caters to three main user roles: Job Seeker, Company, and Admin. Each role has specific features and functionalities that enhance the job searching and recruitment experience.

For job seekers, the portal offers an intuitive interface where they can browse job listings, filter opportunities based on their preferences, and apply directly through the platform. The job listings provide comprehensive details, including company information, job roles, descriptions, required skills, and the number of vacancies. Job seekers can easily submit their applications and track their progress within the portal.

Companies, on the other hand, benefit from a streamlined job posting process. They can create and manage job listings, view applications, and select suitable candidates through a dedicated dashboard. This role-specific feature allows companies to efficiently manage their recruitment process while ensuring that they attract the right talent.

The Admin role is crucial for maintaining the portal's integrity and smooth operation. Admins have the ability to oversee all activities on the platform, including user management, job postings, and application processes. They ensure that the portal remains secure, user-friendly, and up-to-date, thereby providing a reliable service to both job seekers and companies.

Overall, the job searching portal is a comprehensive solution that addresses the needs of modern recruitment. With its responsive design, robust authentication system, and seamless integration of frontend and backend technologies, the portal offers a scalable and efficient platform for connecting employers with qualified candidates.

1.2 COMPONENTS OF SYSTEM

1. User Dashboard

The portal features distinct dashboards for each user role: Job Seeker, Company, and Admin. Job Seekers can manage their profiles, view applied jobs, and track application status. Companies can create, edit, and manage job listings, while Admins oversee the entire portal's activities, including user management and monitoring job listings.

2. Authentication and Security

The Robust authentication mechanisms ensure secure access to the portal. Users are authenticated based on their roles (Job Seeker, Company, Admin), with each role having access to specific functionalities. The system implements password encryption, secure sessions, and role-based access controls to protect sensitive information.

3. Job Listing Management

Companies can create and manage job listings with detailed information such as job title, description, required skills, and number of vacancies. Job Seekers can browse, filter, and search for jobs, with the ability to apply directly through the portal. Listings are dynamically updated, and companies can edit or remove postings as needed.

4. Application Process

Job Seekers can apply for jobs directly on the portal by submitting their resumes and filling out application forms. The application process is streamlined, allowing users to track their application status. Companies receive and review applications through their dashboard, facilitating easy

communication with candidates.

5. Admin Management

Admins have full control over the portal's operations. They can manage user accounts, oversee job listings, monitor application processes, and handle any issues that arise. Admins ensure the platform runs smoothly and securely, maintaining the integrity of data and user interactions.

6. Responsive Design

The portal is designed with a responsive user interface, ensuring a seamless experience across devices. Whether accessed from a desktop, tablet, or smartphone, the layout adjusts to provide an optimal viewing experience, enhancing user engagement and accessibility.

7. Search and Filter Functionality

Advanced search and filtering options are available to help Job Seekers find relevant job listings quickly. Users can filter jobs by criteria such as location, industry, job type, and company. This functionality ensures that Job Seekers can easily find positions that match their skills and preferences.

8. Notification System

The portal includes a notification system that alerts users to important updates. Job Seekers receive notifications about new job postings, application status changes, and messages from employers. Companies are notified about new applications and messages from candidates, ensuring timely communication.

9. Data Management and Analytics

The system collects and analyzes data related to user activity, job postings, and applications. Admins can access analytics dashboards to monitor trends, assess portal performance, and make data-driven decisions to improve the user experience and optimize the recruitment process.

1.3 ADVANCED TECHNOLOGIES

These advanced technologies work together to create a secure, efficient, and scalable job search portal, providing an optimal experience for both job seekers and employers.

1. React.js (Frontend Framework)

React.js is a powerful JavaScript library for building dynamic and responsive user interfaces. It allows the portal to render components efficiently, ensuring a smooth and interactive user experience. React's component-based architecture promotes reusability and maintainability, making the frontend highly scalable.

2. Spring Boot (Backend Framework)

Spring Boot is a robust framework used to create stand-alone, production-grade backend applications in Java. It simplifies the development of RESTful APIs by providing built-in features like dependency injection, data access, and security. Spring Boot enables the backend of the job portal to handle complex business logic and manage data transactions effectively.

3. JSON Web Token (JWT) Authentication

JWT is a modern and secure method for implementing authentication and authorization. In the job portal, JWTs are used to manage user sessions and ensure that only authenticated users can access protected resources. This stateless authentication method enhances security and simplifies the handling of user credentials.

4. BCrypt Password Encryption

BCrypt is a password-hashing function that provides strong security for user passwords stored in the database. It uses a slow hashing algorithm, making it resistant to brute-force attacks. In the job portal, BCrypt is employed to encrypt user passwords, ensuring that sensitive information is securely stored and protected against unauthorized access.

5. RESTful API Architecture

The job portal's backend is designed using RESTful API principles, allowing seamless communication between the frontend and backend. This architecture enables the frontend to request data from the backend via HTTP methods (GET, POST, PUT, DELETE), ensuring that data is handled efficiently and consistently across the application.

6. Role-Based Access Control (RBAC)

RBAC is an advanced security mechanism that restricts access to resources based on user roles. In the job portal, different roles (Job Seeker, Employer, Admin) are assigned specific permissions, ensuring that users only have access to functionalities relevant to their role. This enhances security and streamlines user interactions with the system.

7. Enhanced security measures

Advanced security technologies, including encryption and secure authentication methods, are implemented to protect user data and ensure privacy.

1.4 GLOBAL PERSPECTIVES

A global perspective on a job searching portal encompasses several key aspects:

Global Job Market Trends

The global job market is rapidly evolving, influenced by trends such as the rise of remote work and increasing globalization. Remote work has expanded job opportunities beyond geographical limitations, allowing companies to tap into a global talent pool. This shift necessitates that job search portals accommodate international job listings and offer features that support remote work arrangements. Additionally, the emergence of new industries, such as tech and green energy, is shaping job demand worldwide, making it essential for job portals to stay updated with these trends to provide relevant job opportunities.

Localization and Multilingual Support

To effectively serve a global audience, job search portals must prioritize localization and multilingual support. Localizing job listings involves translating content into various languages and adapting it to regional cultural norms and expectations. This ensures that users in different regions can access job opportunities in their native language, enhancing their overall experience. Additionally, customizing the user interface to reflect regional preferences can significantly improve user engagement and satisfaction, making the portal more accessible and user-friendly.

Compliance with International Regulations

Compliance with international regulations is crucial for operating a global job search portal. Data privacy laws, such as the General Data Protection Regulation (GDPR) in Europe and the California Consumer Privacy Act (CCPA), set stringent requirements for handling user data. Job portals must implement robust data protection measures to ensure compliance with these regulations. Furthermore, adherence to local employment standards is necessary to maintain fair and lawful job listings, respecting the legal requirements of different regions.

Technological Infrastructure and Scalability

Building a globally accessible job search portal requires a scalable technological infrastructure. The portal must be designed to handle varying levels of traffic and data from different regions without compromising performance. Scalable architecture ensures that the portal can grow with increasing user demands. Additionally, integrating with global platforms, such as international job boards and social networks, enhances the portal's functionality, allowing users to access a broader range of job opportunities and network more effectively.

Cultural Sensitivity and User Experience

Cultural sensitivity is vital in designing a job search portal that resonates with users from diverse backgrounds. Content and features should be tailored to respect cultural differences and regional sensibilities, ensuring that the portal is inclusive and welcoming. Providing user support in multiple languages and understanding regional job market nuances can further improve user experience. By addressing these aspects, job portals can foster a more positive and engaging environment for job seekers and employers alike.

Security and Data Protection

Ensuring robust security and data protection is a top priority for global job search portals. Implementing strong authentication measures, such as multi-factor authentication, helps safeguard user accounts from unauthorized access. Additionally, adhering to cybersecurity best practices is essential for protecting against data breaches and cyber threats. By prioritizing security, job portals can build trust with users and protect sensitive information from potential threats.

Economic Factors and Regional Adaptation

Economic conditions vary significantly across regions and influence job market dynamics. A global job search portal must adapt to these regional economic realities to provide relevant job recommendations and features. Understanding local economic conditions helps tailor job listings and opportunities to match regional demand and supply. By aligning the portal's offerings with the economic landscape of different regions, job portals can better serve users and address specific market needs.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Job search portals like LinkedIn, Indeed, Glassdoor, Monster, CareerBuilder, ZipRecruiter, SimplyHired, Indeed Flex, Upwork, and FlexJobs each provide unique platforms to connect job seekers with employment opportunities. These systems aggregate job listings from various sources and offer a range of functionalities to enhance the job search process.

Overall, these portals enable users to search for and apply to jobs, post resumes, and receive job alerts. They often include additional features such as company reviews, salary insights, and career advice to assist users in making informed decisions. While some portals, like LinkedIn and Glassdoor, emphasize professional networking and company insights, others like Indeed and ZipRecruiter focus on job aggregation and matching through AI technology. Platforms such as Upwork and FlexJobs cater specifically to freelance and flexible work arrangements, reflecting the growing demand for non-traditional employment.

Overall, these portals enable users to search for and apply to jobs, post resumes, and receive job alerts. They often include additional features such as company reviews, salary insights, and career advice to assist users in making informed decisions. While some portals, like LinkedIn and Glassdoor, emphasize professional networking and company insights, others like Indeed and ZipRecruiter focus on job aggregation and matching through AI technology. Platforms such as Upwork and FlexJobs cater specifically to freelance and flexible work arrangements, reflecting the growing demand for non-traditional employment.

2.1.1 DRAWBACKS

1. Limitations in Job Listing Quality and Accuracy

Many existing job search portals face challenges related to the quality and accuracy of job listings. Platforms like Indeed and SimplyHired aggregate job postings from multiple sources, which can result in outdated or duplicate listings. Additionally, job matching algorithms, as seen in ZipRecruiter, may not always provide highly relevant job recommendations, leading to a less efficient job search process. Platforms focusing heavily on company reviews, such as Glassdoor, may also offer fewer job listings, potentially limiting the variety of opportunities available to users.

2. User Experience and Accessibility Issues

User experience and accessibility can be problematic across various job search portals. Sites like Monster may have outdated interfaces, affecting ease of use compared to more modern platforms. Additionally, some portals, such as Upwork and FlexJobs, impose subscription or transaction fees that can be a barrier for users, particularly those seeking free resources. The overwhelming amount of information and limited features on certain platforms can further complicate the job search process and affect overall user satisfaction.

2.2 PROBLEM DEFINITION

1. Job Listing Accuracy and Relevance

Existing job search portals often struggle with ensuring the accuracy and relevance of job listings. Aggregation from multiple sources can lead to outdated, duplicate, or irrelevant job postings, making it difficult for users to find up-to-date and suitable opportunities. Additionally, the effectiveness of job matching

algorithms may vary, resulting in job recommendations that do not align with the user's skills, preferences, or career goals. This issue affects both job seekers, who may waste time sifting through unsuitable listings, and employers, who may not reach the right candidates.

2. User Experience and Accessibility Barriers

User experience and accessibility are significant challenges for many job search portals. Some platforms suffer from outdated interfaces, which can hinder navigation and usability, making the job search process cumbersome. Furthermore, certain portals impose subscription or transaction fees, creating financial barriers for users and limiting access to premium features or comprehensive job listings. The overwhelming volume of information and the lack of intuitive design elements can exacerbate these issues, leading to a less effective and more frustrating experience for both job seekers and employers.

2.3 PROPOSED SYSTEM

Unlike many existing platforms where job listings can be outdated or irrelevant due to aggregation from multiple sources, our portal leverages a more controlled approach to ensure up-to-date and relevant job postings. The use of advanced matching algorithms and integration with verified sources helps deliver accurate job recommendations tailored to users' specific skills and preferences, reducing the frustration of sifting through unsuitable listings.

Our job portal emphasizes a modern, intuitive interface designed to improve user experience. By avoiding outdated design elements and focusing on a user-friendly layout, the portal enhances navigation and usability. Additionally, unlike some systems that impose subscription or transaction fees, our portal aims to offer a comprehensive range of features without financial barriers, making it more accessible to a wider audience. This approach addresses the issues of accessibility and usability commonly faced by users of existing job search platforms.

2.3.1 ADVANTAGES

- Accurate and Relevant Job Listings
- User-Friendly Interface
- Enhanced Security
- Comprehensive Feature Set

CHAPTER 3

SYSTEM REQUIREMENTS

The hardware and software requirements and also the platform description of the system are explained under sections 3.1, 3.2 and 3.3 respectively.

3.1 HARDWARE REQUIREMENTS

- | | | |
|-------------------|---|---------------|
| 1. Processor Type | : | intel CORE i3 |
| 2. RAM | : | 8GB RAM |
| 3. Hard Disk | : | 512GB |

3.2 SOFTWARE REQUIREMENTS

- | | | |
|------------------------|---|---------------------------------|
| a. Operating system | : | Windows 11 |
| b. Front End, Back End | : | Visual studio code, Spring Boot |
| c. Coding Language | : | ReactJs, Java |

3.3 SOFTWARE DESCRIPTION



Fig. 3.1. VS Code Logo

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft, it can be used to work with Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add functionality.

3.3.1 FRONTEND

ReactJs

ReactJS is a popular JavaScript library for building user interfaces, particularly for web applications. React follows a component-based architecture, where UIs are broken down into reusable components. Each component encapsulates its own logic and UI making it easier to manage and maintain complex user interfaces. React uses a virtual DOM (Document Object Model) to improve performance. Instead of directly manipulating the DOM, React creates a virtual representation of the DOM in memory and updates it efficiently. When changes occur, React compares the virtual DOM with the actual DOM and only updates the necessary parts, reducing the number of DOM manipulations and improving performance.

React uses JSX, a syntax extension that allows developers to write HTML-like code within JavaScript. JSX makes it easier to write and understand React components, as it closely resembles the final UI structure. React follows a unidirectional data flow, also known as one-way data binding. Data flows from parent components to child components via props, and child components can communicate with parent components via callbacks. This helps maintain a clear and predictable data flow in the application.

Features of ReactJs

1. Declarative

React makes it easy to create interactive UIs by using a declarative programming approach. Developers can describe how the UI should look based on the application state.

2. Component-Based

React uses a component-based architecture, where UIs are composed of reusable and self-contained components. This makes it easier to manage and maintain complex UIs, as each component can be developed, tested, and updated independently.

3. Virtual DOM

React uses a virtual DOM (Document Object Model) to improve performance. Instead of updating the entire DOM when the state changes, React compares the virtual DOM with the actual DOM and only updates the parts that have changed.

4. JSX

JSX is a syntax extension for JavaScript that allows developers to write HTML-like code within their JavaScript code. This makes it easier to create and manage UI components, as JSX code can be more readable.

5. Unidirectional Data Flow

React follows a unidirectional data flow, from parent components to child components. This helps to maintain the consistency of the application state and understand the data flow in the application.

6. React Native

React Native is a framework for building native mobile applications using React. It allows developers to use the same codebase to build both iOS and Android applications, saving time and effort in development.

7. Community and Ecosystem

React has a large and active community of developers, which has led to the development of a rich ecosystem of libraries, tools, and resources that can be used to enhance and extend React applications.

3.3.2 BACKEND

Java

Java is a versatile, object-oriented programming language renowned for its platform independence, security, and portability. Java is a high-level, general-purpose programming language that is widely used for developing a variety of applications. Java is object-oriented, emphasizing the use of classes and objects for organizing code and data.

It boasts a comprehensive standard library with built-in classes and APIs for various tasks, from data manipulation to networking. Java enforces strong type checking, enhancing code reliability and reducing runtime errors. The language includes automatic memory management through garbage collection, simplifying memory allocation and deallocation.

The extensions used to develop my backend part of the project are,

1. Spring Boot Extension Pack by VMware

This extension pack provides a set of tools and features to enhance your development experience with Spring Boot, including code snippets, syntax highlighting, and project templates.

2. Extension Pack for Java by Microsoft

This extension pack includes essential tools for Java developers, such as debugging support, code navigation, and IntelliSense for Java files.

3. Spring Boot Snippets by Developer Soapbox

This extension provides a collection of code snippets for commonly used Spring Boot annotations and configurations, helping you write code more efficiently.

Dependencies used to build my project are,

1. Spring Web

This dependency provides the necessary components for building web applications with Spring, including controllers, request mappings, and HTTP message converters.

2. Dev Tools

Spring Boot DevTools provides a set of tools to improve the development experience, including automatic application restarts, live reload, and enhanced debugging capabilities.

3. Data JPA

Spring Data JPA provides support for easily working with JPA (Java Persistence API) repositories, simplifying the implementation of data access logic in your application.

4. Spring Security Web

This dependency provides support for securing your web application using Spring Security, including authentication and authorization mechanisms.

5. MySQL Driver

This dependency provides the JDBC driver for MySQL, enabling your Spring Boot application to connect to a MySQL database, which is crucial for applications requiring MySQL as their database solution.

CHAPTER 4

SYSTEM DESIGN

4.1 MODULE DESCRIPTION

- Admin management
- Jobseeker management
- Company management

4.1.1 ADMIN MANAGEMENT

Overview:

Admin manages provider profiles, including qualifications, availability, and contact details.

Features:

Add Providers: Create new profiles with essential details.

Edit Profiles: Update qualifications, contact info, and availability.

Delete Providers: Remove profiles and reallocate services if needed.

Manage Qualifications: Ensure compliance and update certifications.

Benefits:

Keeps provider information accurate and current.

Enhances platform efficiency.

4.1.2 JOBSEEKER MANAGEMENT

Overview:

It enables users to create and update profiles, browse job listings, apply for positions, and track their application progress efficiently

Features:

Add Profiles: Create and update profiles

Apply jobs: Search and apply for jobs

Tracking: Track application status

Manage application history: Managing the history of applications

Benefits:

Enhanced job search efficiency

Improved Personalization and Privacy Control

4.1.3 COMPANY MANAGEMENT

Overview:

Company management in your job portal allows organizations to create and manage profiles, post job listings, track applications, and manage their recruitment processes efficiently.

Features:

Create company profiles: Create and update company profiles

Manage Joblistings: Post and manage Joblistings

Track Status: Tracking the applications

Benefits:

Enhanced company visibility

Efficient application tracking

4.2 USE CASE DIAGRAM

A use case diagram is a visual representation in UML (Unified Modelling Language) that illustrates the interactions between actors and a system or software application. It is used to depict the various ways users or external entities can interact with the system and the specific functionalities or use cases it offers.

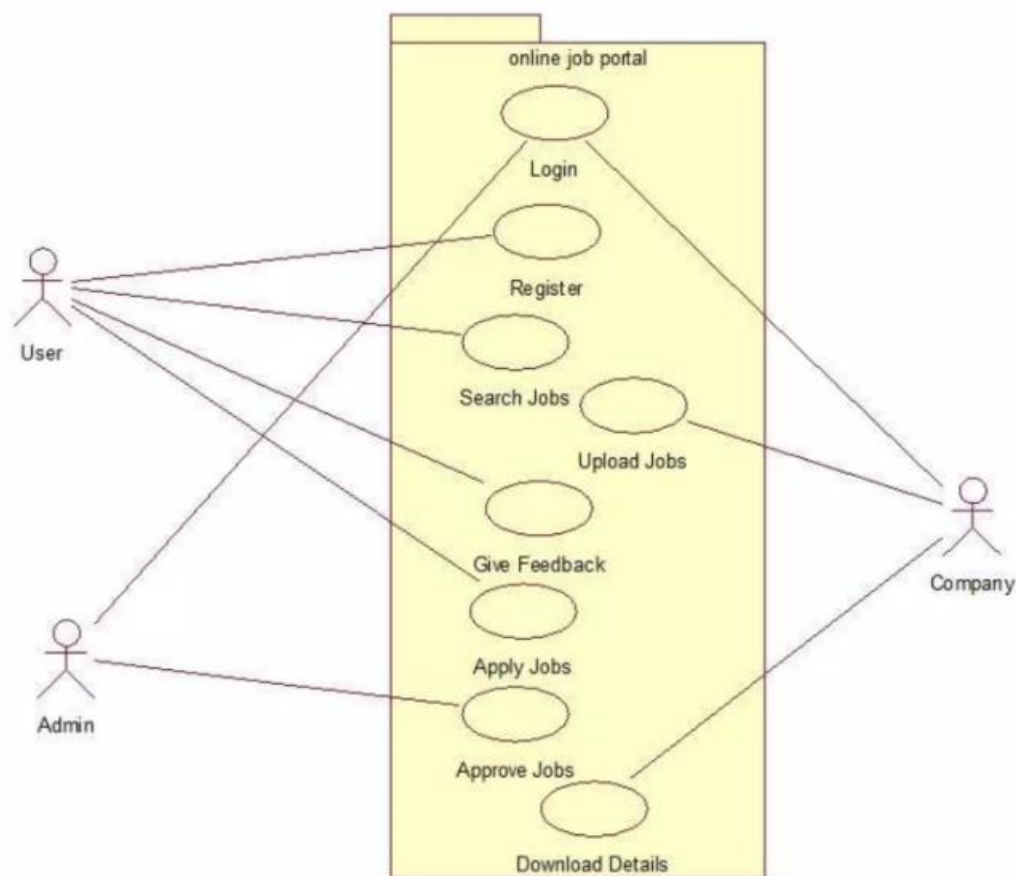


Fig. 4.1. Use Case Diagram

4.3 SEQUENCE DIAGRAM

A sequence diagram is a visual representation used in software engineering to illustrate the interactions and communication between different objects or components in a system over a specific period of time. It shows the chronological order of messages or method calls exchanged between these entities, helping to depict the dynamic behaviour of a system or a particular scenario. In essence, it

provides a time-ordered view of how various parts of a system collaborate to achieve a particular task or functionality.

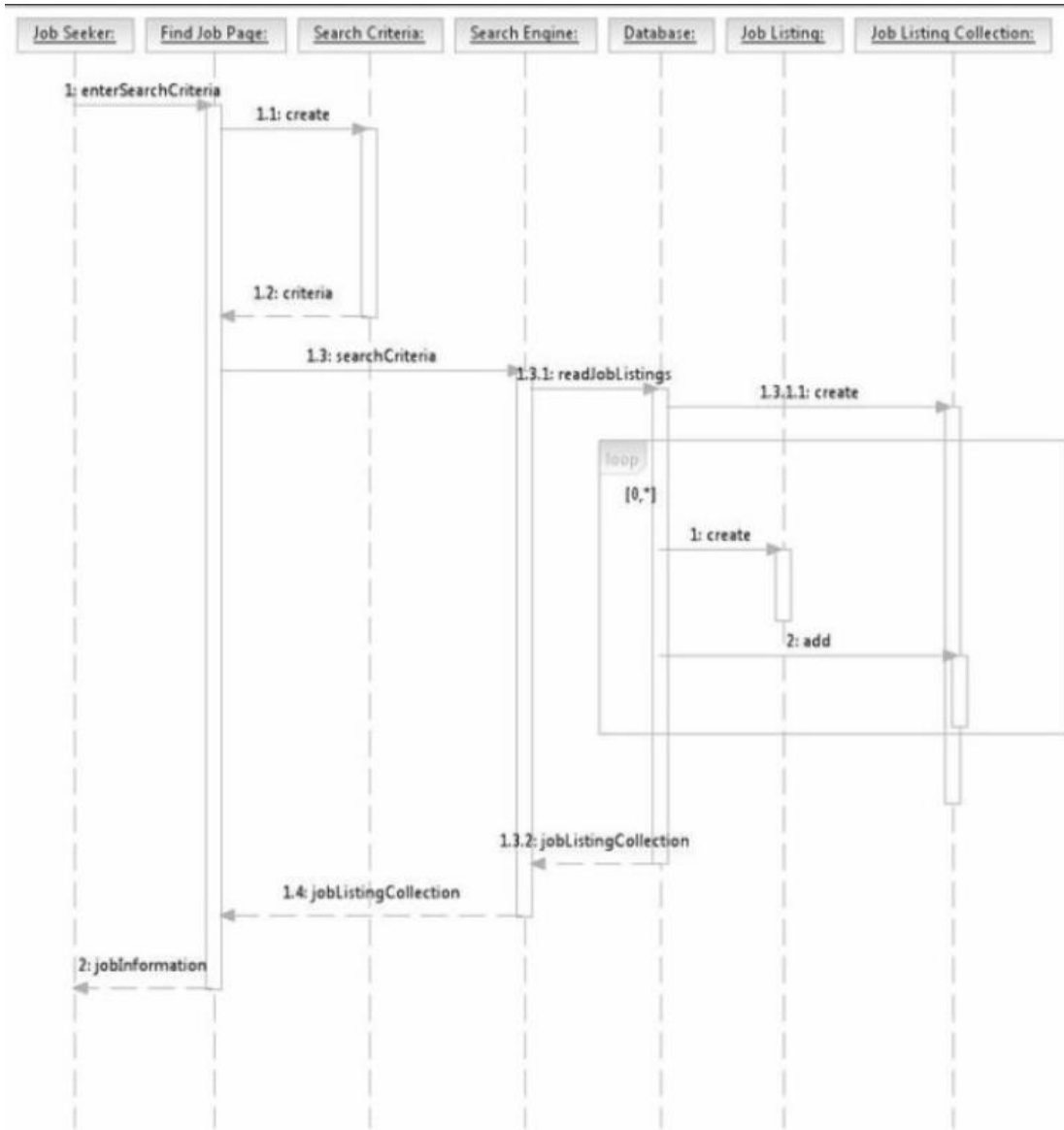
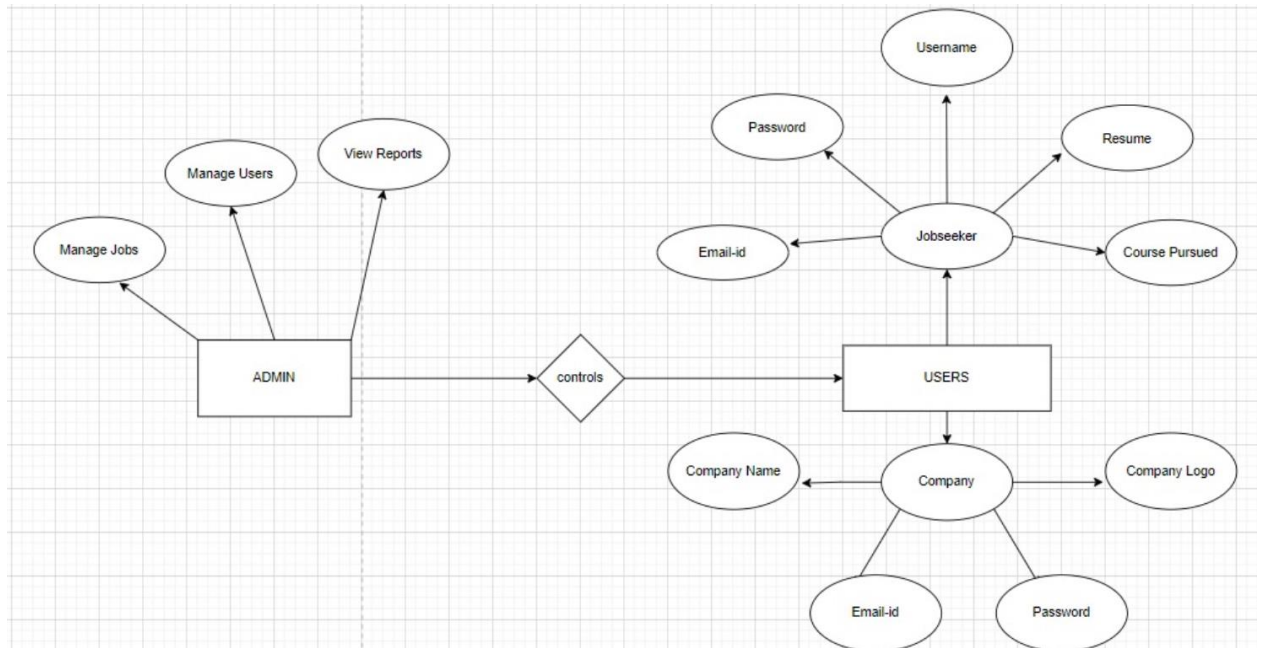


Fig. 4.2. Sequence Diagram

4.4 ER DIAGRAM



An Entity-Relationship (ER) Diagram is a visual representation of the data structure for a database, showing the entities (such as tables or objects) and their relationships. It typically includes entities with attributes (e.g., fields or properties), primary keys (unique identifiers for records), and foreign keys (references to other entities). Relationships between entities, such as one-to-many or many-to-many, illustrate how data is interconnected.

Fig. 4.3. ER Diagram

CHAPTER 5

TESTING

5.1 UNIT TESTING

Unit testing in our job portal focuses on verifying the functionality of individual components and functions to ensure they work correctly in isolation. This involves testing elements such as user input validation, job listing management, application processing, and profile management to confirm they perform their intended tasks accurately. By conducting these tests, you identify and address potential issues early, ensuring that each component operates reliably and contributes to the overall stability and performance of the job portal.

5.2 INTEGRATION TESTING

Integration testing in our job portal involves verifying that different modules and components work together seamlessly. This includes testing interactions between features such as job listing creation, application submission, and user profile management to ensure they function cohesively. Integration tests focus on detecting issues that may arise when components interact, such as data inconsistencies or communication failures between the frontend and backend. By conducting these tests, you ensure that integrated parts of the system collaborate effectively, providing a smooth and functional user experience across the entire portal.

5.3 SECURITY AND AUTHENTICATION

Security and authentication are critical aspects of our job search portal, designed to protect user data and ensure secure access to the platform. The portal implements robust security measures, including encrypted communication and secure storage of sensitive information, such as passwords and personal details. Encryption protocols, such as SSL/TLS, safeguard data transmitted between **users and the server, while**

hashed and salted passwords enhance protection against unauthorized access.

Authentication processes are designed to verify user identities through secure login mechanisms. The portal employs multi-factor authentication (MFA) to add an extra layer of security, requiring users to provide additional verification, such as a one-time code sent to their mobile device, alongside their standard login credentials. This approach mitigates the risk of unauthorized access and enhances overall account security.

Additionally, role-based access control (RBAC) is implemented to manage permissions and ensure that users only have access to the features and data relevant to their roles. Job seekers, employers, and administrators have distinct access levels, preventing unauthorized users from accessing sensitive information or performing actions outside their designated scope. These security and authentication measures collectively ensure a secure and trustworthy environment for all users of the job search portal.

Fig. 5.1. Login Page

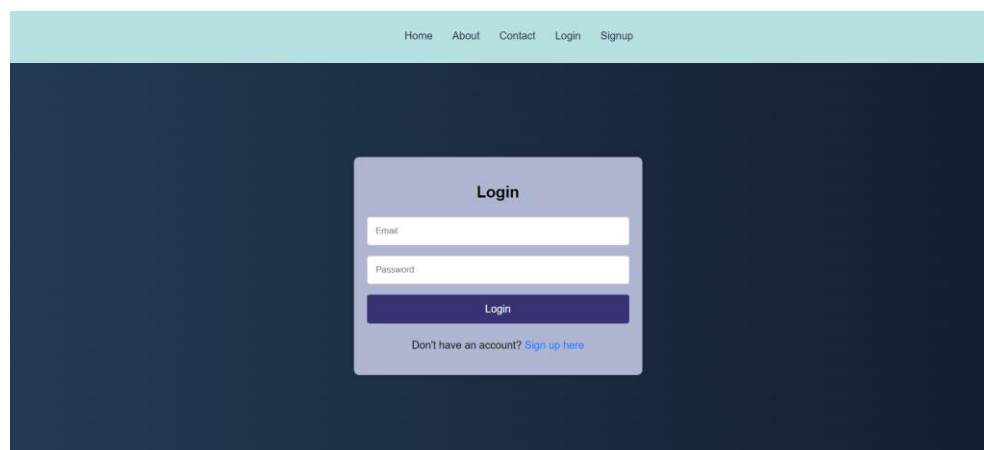
The image shows a web application's login page. At the top, there is a light blue navigation bar with links for 'Home', 'About', 'Contact', 'Login', and 'Signup'. The main content area has a dark blue background. In the center, there is a light purple rectangular box titled 'Login'. Inside this box, there are two white input fields: 'Email' and 'Password'. Below these fields is a dark purple button labeled 'Login'. At the bottom of the box, there is a link that says 'Don't have an account? Sign up here'.

Fig 5.2 Signup page

[Home](#) [About](#) [Contact](#) [Login](#) [Signup](#)

Job Seeker
Signup

Company
Signup

Admin
Signup

Sign Up

Name

Enter your name

Email

Enter your email

Password

Enter your password

Course Pursued

Enter your course pursued

Resume

Choose File | No file chosen

City

Enter your city

Country

Enter your country

Sign Up

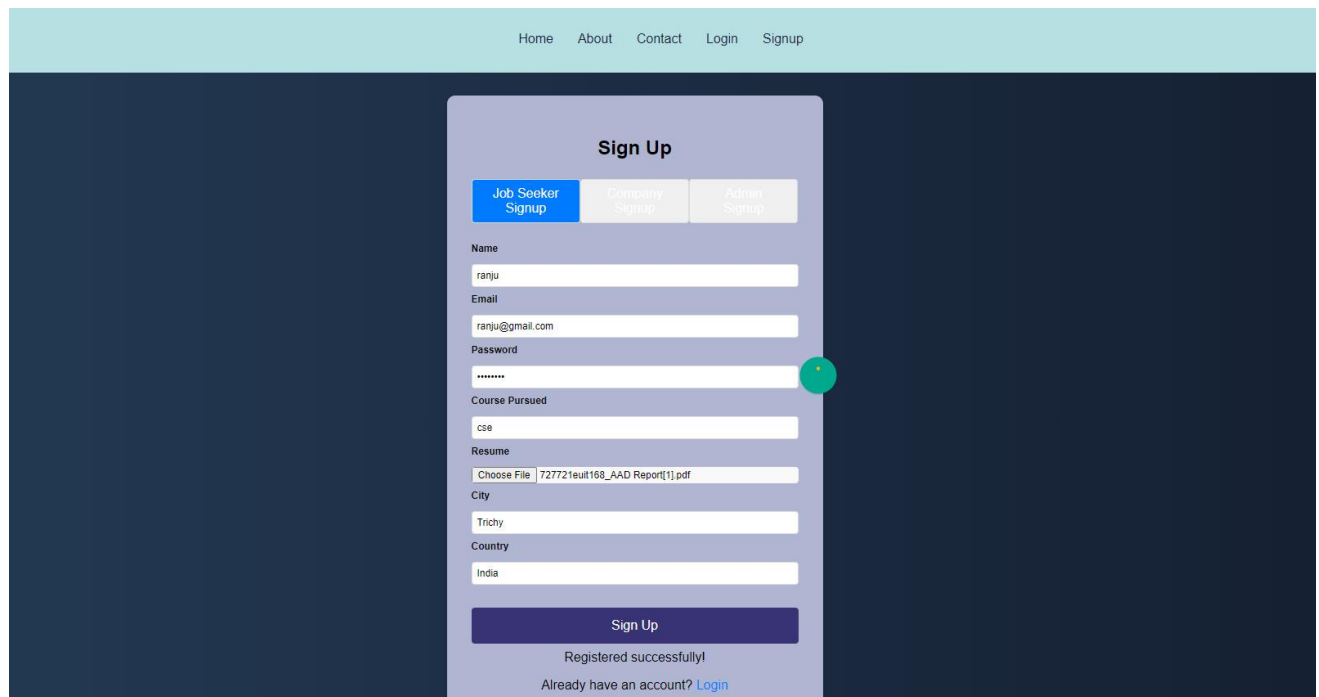
Already have an account? [Login](#)

5.4 TEST CASES

Test cases for your job search portal encompass a range of functionalities to ensure system reliability and performance. For user registration and login, test cases include verifying successful registration with valid data, handling invalid data entries, and checking correct and incorrect login credentials. Job listing management involves creating, editing, and deleting job postings to ensure proper functionality. Application process test cases focus on the successful submission of job applications and accurate tracking of application status. Profile management test cases ensure that users can create, update, and view their profiles and company profiles correctly. Security and authentication tests include verifying password encryption and multi-factor authentication (MFA) to ensure secure user access and data protection.

5.4.1 TEST CASE I

Fig. 5.3. Test Case I



Home About Contact Login Signup

Sign Up

Job Seeker Signup Company Signup Admin Signup

Name
ranju

Email
ranju@gmail.com

Password

Course Pursued
cse

Resume
Choose File | 727721eult168_AAD Report[1].pdf

City
Trichy

Country
India

Sign Up

Registered successfully!

Already have an account? [Login](#)

EXPECTED OUTPUT: Registration failed and error message occurs, to enter the credentials.

ACTUAL OUTPUT: Registration failed, indicating to enter the credentials.

5.4.2 TEST CASE II

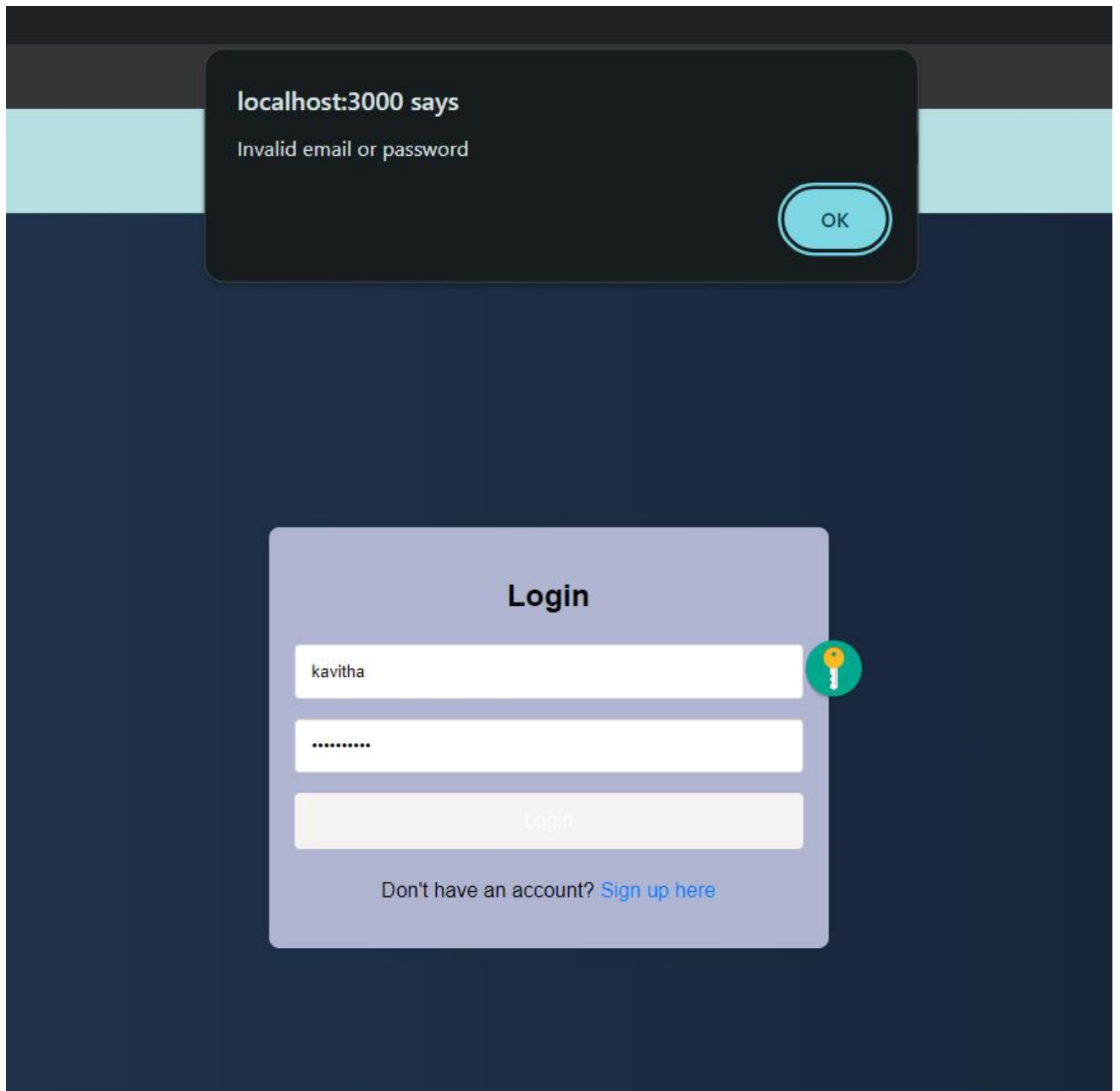


Fig. 5.4. Test Case II

EXPECTED OUTPUT: Login fails, and an error message prompts the user to enter valid credentials or correct any issues with the login information.

ACTUAL OUTPUT: Login fails, and an error message prompts the user to enter valid credentials or correct any issues with the login information.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

The development of our job search portal represents a significant advancement in creating a streamlined and efficient platform for job seekers and employers. The portal integrates key features such as user registration, job listing management, application tracking, and profile management, ensuring a comprehensive and user-friendly experience. Rigorous testing, including unit, integration, security, and usability tests, has validated the system's functionality, performance, and security, addressing potential issues and enhancing overall reliability. By incorporating advanced technologies and adhering to best practices in software development, the portal not only meets the current needs of its users but also positions itself for future scalability and adaptability. The project successfully bridges the gap between job seekers and employers, offering a robust solution for the evolving job market.

6.2 FUTURE WORK

In Future ,our job search portal includes enhancing its capabilities to stay relevant and competitive in the evolving job market. One key area for development is the integration of advanced artificial intelligence (AI) and machine learning algorithms to provide more personalized job recommendations, improve matching accuracy, and automate resume screening. Additionally, expanding the portal's features to include more comprehensive career resources, such as virtual career coaching, skills assessment tools, and interactive webinars, can further support job seekers in their career development and job search efforts.

Another important aspect of future work is improving user engagement and retention through enhanced mobile support and user experience **improvements**.

This includes optimizing the portal for various devices and platforms, developing a dedicated mobile application, and incorporating user feedback to continuously refine and expand functionality. Additionally, implementing more robust analytics and reporting tools for both job seekers and employers can provide valuable insights and help in making data-driven decisions. These initiatives will help ensure that the job portal remains a valuable and dynamic tool for users, adapting to their needs and the trends of the job market.

CHAPTER 7

APPENDICE

APPENDIX I

SOURCE CODE

HomePage.js

```
import React from 'react';
import { useNavigate } from 'react-router-dom';
import './home.css'; // Import your CSS file for styling

const Homepage = () => {
  const navigate = useNavigate();

  const handleApplyClick = () => {
    navigate('/signup');
  };

  const handleReadMoreClick = (page) => {
    navigate(`${page}`);
  };

  return (
    <div className="homepage">
      { /* Call to Action Section */ }
      <section className="call-to-action">
        <h2>Ready to Dive into JobXplore?</h2>
      </section>
    </div>
  );
};
```

<p>Sign up today and start exploring exciting job opportunities tailored to your career goals.</p>

Get Started

</section>

{/* Career Advice Section */}

<section className="career-advice">

<h2>Career Advice & Tips</h2>

<div className="advice-list">

<div className="advice-card">

<h3>How to Ace Your Interview</h3>

<p>Get expert tips on how to prepare for and succeed in your job interviews.</p>

Read More

</div>

<div className="advice-card">

<h3>Building a Standout Resume</h3>

<p>Learn the best practices for crafting a resume that gets noticed by employers.</p>

<a href="https://enhancv.com/blog/how-to-make-your-resume-stand-out/"

`className="read-more" target="_blank" rel="noopener noreferrer">Read More`

`</div>`

`<div className="advice-card">`

``

`<h3>Networking Strategies for Success</h3>`

`<p>Discover effective networking strategies to expand your professional connections and advance your career.</p>`

`Read More`

`</div>`

`{/* Add more advice cards as needed */}`

`</div>`

`{/* Testimonials Section */}`

`<section className="testimonials">`

`<h2>Success Stories</h2>`

`<div className="testimonial-list">`

`<div className="testimonial-card">`

`<p>"I found the perfect job through this platform! The process was smooth and straightforward."</p>`

`- Sarah Lee`

`</div>`

`<div className="testimonial-card">`

`<p>"A great resource for job seekers. The job recommendations were spot on and helped me land a great role."</p>`

`- Michael Brown`

`</div>`

```
</div>
</section>
```

```
</section>
</div>
);
};

export default Homepage;
```

About.js

```
import React from 'react';
import '../About.css';
```

```
const About = () => {
  return (
    <div className="about-container">
      <header className="about-header">
        <h1>About Us</h1>
        <div className="image-text-container">
          <div className="text-content">
            <p>
```

Welcome to [Your Job Portal Name], your one-stop destination for all your career aspirations.

We understand that finding the right job can be challenging, whether you're a

fresh graduate

stepping into the professional world or an experienced professional seeking new opportunities.

That's why we've created a platform that connects job seekers with their dream jobs and companies with the right talent.

</p>

</div>

</div>

</header>

<section className="about-content">

<div className="image-text-container reverse">

<div className="text-content">

<h2>Why Choose Us?</h2>

Extensive Job Listings: With thousands of job listings updated daily, you have access to the latest opportunities from top companies across the globe.

User-Friendly Interface: Our platform is designed to be intuitive and easy to navigate, ensuring that you can find and apply for jobs with just a few clicks.

Customized Job Alerts: Stay ahead of the competition with personalized job alerts tailored to your preferences.

Career Resources: We offer a wealth of resources, including resume tips, interview advice, and career guidance to help you succeed in your job search.

Trusted by Top Employers: We partner with leading companies who trust us to help them find the right talent.

</div>

</div>

<div className="image-text-container">

<div className="text-content">

<h2>Our Vision</h2>

<p>

Our vision is to empower every job seeker to achieve their career goals and every employer to find

the perfect match for their organization. We are committed to providing a platform that is not only

efficient and effective but also transparent and trustworthy.

</p>

</div>

</div>

<div className="join-us">

<h2>Join Us</h2>

<p>

Whether you're looking for your next job or the next great hire, [Your Job Portal Name] is here to help

you every step of the way. Join us today and take the first step toward a brighter future.

</p>

Join Us

</div>

</section>

</div>

);

};

export default About;

Navbar.js

import React from 'react';

import { Link } from 'react-router-dom';

```

function Navbar() {
  return (
    <nav>
      <ul>
        <li><Link to="/">Home</Link></li>
        <li><Link to="/about">About</Link></li>
        <li><Link to="/contact">Contact</Link></li>
        <li><Link to="/login">Login</Link></li>
        <li><Link to="/signup">Signup</Link></li>
      </ul>
    </nav>
  );
}

```

```
export default Navbar;
```

SignupPage.js

```

import React, { useState } from 'react';
import axios from 'axios';
import '../styles.css'; // Ensure you have this CSS file for styles

```

```

function SignupPage() {
  const [userType, setUserType] = useState('jobseeker');
  const [isRegistered, setIsRegistered] = useState(false);

  const handleSubmit = async (e) => {
    e.preventDefault();
    const formData = new FormData(e.target);
    const resumeFile = formData.get('resume');

```



```

const logoFile = formData.get('logo');
// const allowedResumeTypes = ['application/pdf', 'application/msword',
'application/vnd.openxmlformats-officedocument.wordprocessingml.document'];
// const allowedLogoTypes = ['image/png', 'image/jpeg'];

// if (userType === 'jobseeker' && resumeFile &&
!allowedResumeTypes.includes(resumeFile.type)) {
//   alert('Please upload a PDF or Word document. ');
//   return;
// }

// if (userType === 'company' && logoFile &&
!allowedLogoTypes.includes(logoFile.type)) {
//   alert('Please upload a PNG or JPEG image. ');
//   return;
// }

const payload = {
  name: formData.get('name'),
  email: formData.get('email'),
  password: formData.get('password'),
  coursepursued: formData.get('coursepursued'),
  city: formData.get('city'),
  country: formData.get('country'),
  resume: "Resume.pdf",
  companyName: formData.get('company-name'),
  companyLogo: "pic.jpg",
  role: userType,
};

```

```

try {
  const endpoint = 'http://localhost:8080/users/register';
  console.log(payload.resume);
  console.log(payload);
  const response = await axios.post(endpoint, payload);
  console.log(response.data);
  if (response.status === 200) {
    setIsRegistered(true);
  }
} catch (error) {
  console.error('Signup error:', error);
  alert('An error occurred during signup.');
```

```

};

const fileToBase64 = (file) => {
  return new Promise((resolve, reject) => {
    const reader = new FileReader();
    reader.readAsDataURL(file);
    reader.onload = () => resolve(reader.result);
    reader.onerror = (error) => reject(error);
  });
};
```

```

return (
  <div className="signup-page">
    <div className="signup-container">
      <h2>Sign Up</h2>
```

```

<div className="tabs">
  <button className={ userType === 'jobseeker' ? 'active' : '' } onClick={() =>
setUserType('jobseeker')}>
    Job Seeker Signup
  </button>
  <button className={ userType === 'company' ? 'active' : '' } onClick={() =>
setUserType('company')}>
    Company Signup
  </button>
  <button className={ userType === 'admin' ? 'active' : '' } onClick={() =>
setUserType('admin')}>
    Admin Signup
  </button>
</div>
<form onSubmit={handleSubmit}>
  { userType === 'jobseeker' && (
    <div className="form-group">
      <label htmlFor="name">Name</label>
      <input type="text" id="name" name="name" placeholder="Enter your name"
required />
      <label htmlFor="email">Email</label>
      <input type="email" id="email" name="email" placeholder="Enter your
email" required />
      <label htmlFor="password">Password</label>
      <input type="password" id="password" name="password"
placeholder="Enter your password" required />
      <label htmlFor="course">Course Pursued</label>
      <input type="text" id="course" name="coursepursued" placeholder="Enter
your course pursued" required />

```

```

    <label htmlFor="resume">Resume</label>
    <input type="file" id="resume" />
    <label htmlFor="city">City</label>
    <input type="text" id="city" name="city" placeholder="Enter your city"
required />
    <label htmlFor="country">Country</label>
    <input type="text" id="country" name="country" placeholder="Enter your
country" required />
</div>
)}}

{userType === 'company' && (
    <div className="form-group">
        <label htmlFor="company-name">Company Name</label>
        <input type="text" id="company-name" name="company-name"
placeholder="Enter company name" required />
        <label htmlFor="email">Email</label>
        <input type="email" id="email" name="email" placeholder="Enter your
email" required />
        <label htmlFor="password">Password</label>
        <input type="password" id="password" name="password"
placeholder="Enter your password" required />
        <label htmlFor="confirm-password">Confirm Password</label>
        <input type="password" id="confirm-password" name="confirm-password"
placeholder="Confirm your password" required />
        <label htmlFor="logo">Company Logo</label>
        <input type="file" id="logo" />
    </div>
)}}

```

```

    {userType === 'admin' && (
      <div className="form-group">
        <label htmlFor="email">Email</label>
        <input type="email" id="email" name="email" placeholder="Enter your
email" required />
        <label htmlFor="password">Password</label>
        <input type="password" id="password" name="password"
placeholder="Enter your password" required />
      </div>
    )}
    <button type="submit">Sign Up</button>
    {isRegistered && (
      <div className="success-message">
        Registered successfully!
      </div>
    )}
    <p>
      Already have an account? <a href="/login">Login</a>
    </p>
  </form>
</div>
</div>
);
}

```

```

export default SignupPage;

```

LoginPage.js

```
import React, { useState } from 'react';

import { useNavigate } from 'react-router-dom';

import '../styles.css'; // Ensure you have a
CSS file for styling

import axios from 'axios';

function LoginPage() {

  const [isJobSeeker, setIsJobSeeker] =
useState(true);

  const [email, setEmail] = useState("");

  const [password, setPassword] =
useState("");

  const navigate = useNavigate();

  const handleLogin = async () => {

    const response = await
axios.get(http://localhost:8080/users/login
?email=${email}&password=${password}
);

    console.log(response.data);

    if (response.data) {

      switch (response.data.role) {
```

```

        case 'jobseeker':
            navigate('/jobs');
            break;
        case 'company':
            navigate('/company-dashboard');
            break;
        case 'admin':
            navigate('/admin-dashboard');
            break;
        default:
            alert('Invalid role');
    }
} else {
    alert('Invalid email or password');
}
};

```

```

return (
    <div className="login-page">
        <div className="login-container">
            <h2>Login</h2>

            <input
                type="email"
                placeholder="Email"
                value={email}
            />
        </div>
    </div>
)

```

```

        onChange={ (e) =>
setEmail(e.target.value)}
    />
    <input
        type="password"
        placeholder="Password"
        value={password}
        onChange={ (e) =>
setPassword(e.target.value)}
    />
    <button
onClick={ handleLogin }>Login</button>
    <p>
        Don't have an account? <a
href="/signup">Sign up here</a>
    </p>
</div>
</div>
);
}

```

```
export default LoginPage;
```

ApplyPage.js

```

import React, { useState } from 'react';
import { useNavigate } from 'react-router-
dom';

```



```

import './apply.css';

function ApplyPage() {
  const [formData, setFormData] =
useState({
  name: "",
  email: "",
  phone: "",
  coverLetter: "",
  resume: null
});

  const [submitted, setSubmitted] =
useState(false);

  const navigate = useNavigate();

  const handleChange = (e) => {
    const { name, value, type, files } =
e.target;

    setFormData({
      ...formData,
      [name]: type === 'file' ? files[0] : value
    });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
  };

```

```

    // Logic to handle form submission
    (e.g., API call)

    setSubmitted(true);

    setTimeout(() => {

        navigate('/'); // Navigate to a thank-you
page or similar

    }, 2000); // Wait for 2 seconds before
redirecting

    };

    return (

        <div className="apply-page">

            <h2>Apply for the Job</h2>

            <form onSubmit={handleSubmit}
className="apply-form">

                <div className="form-group">

                    <label htmlFor="name">Full
Name</label>

                    <input

                        type="text"

                        id="name"

                        name="name"

                        value={formData.name}

                        onChange={handleChange}

                        placeholder="John Doe"

                        required

                    />

```

```

</div>

<div className="form-group">
  <label htmlFor="email">Email
Address</label>
  <input
    type="email"
    id="email"
    name="email"
    value={ formData.email }
    onChange={ handleChange }

placeholder="john.doe@example.com"
    required
  />
</div>

<div className="form-group">
  <label htmlFor="phone">Phone
Number</label>
  <input
    type="tel"
    id="phone"
    name="phone"
    value={ formData.phone }
    onChange={ handleChange }
    placeholder="+91 234 567 890"
    required

```

```

    />
  </div>

  <div className="form-group">
    <label
htmlFor="coverLetter">Cover
Letter</label>

    <textarea
      id="coverLetter"
      name="coverLetter"
      value={ formData.coverLetter }
      onChange={ handleChange }
      rows="6"
      placeholder="Write your cover
letter here..."
      required
    />
  </div>

  <div className="form-group">
    <label htmlFor="resume">Upload
Resume</label>

    <input
      type="file"
      id="resume"
      name="resume"
      onChange={ handleChange }
      accept=".pdf,.doc,.docx"
      required

```

```

        />
    </div>

    <button type="submit"
className="submit-button">Submit
Application</button>

    </form>

    {submitted && <div
className="submission-
message">Application submitted
successfully!</div>}

    </div>

);

}

export default ApplyPage;

```

APPENDIX II SCREENSHOTS

Fig. A.2.1. Main Page

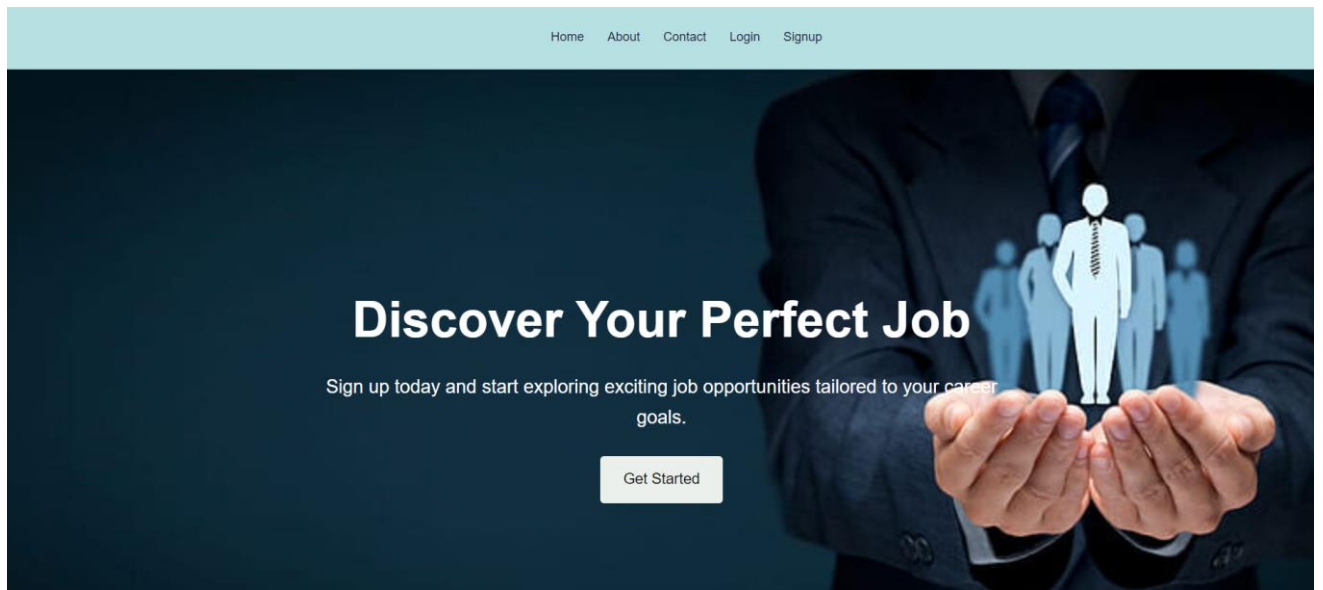


Fig. A.2.2. Admin Login

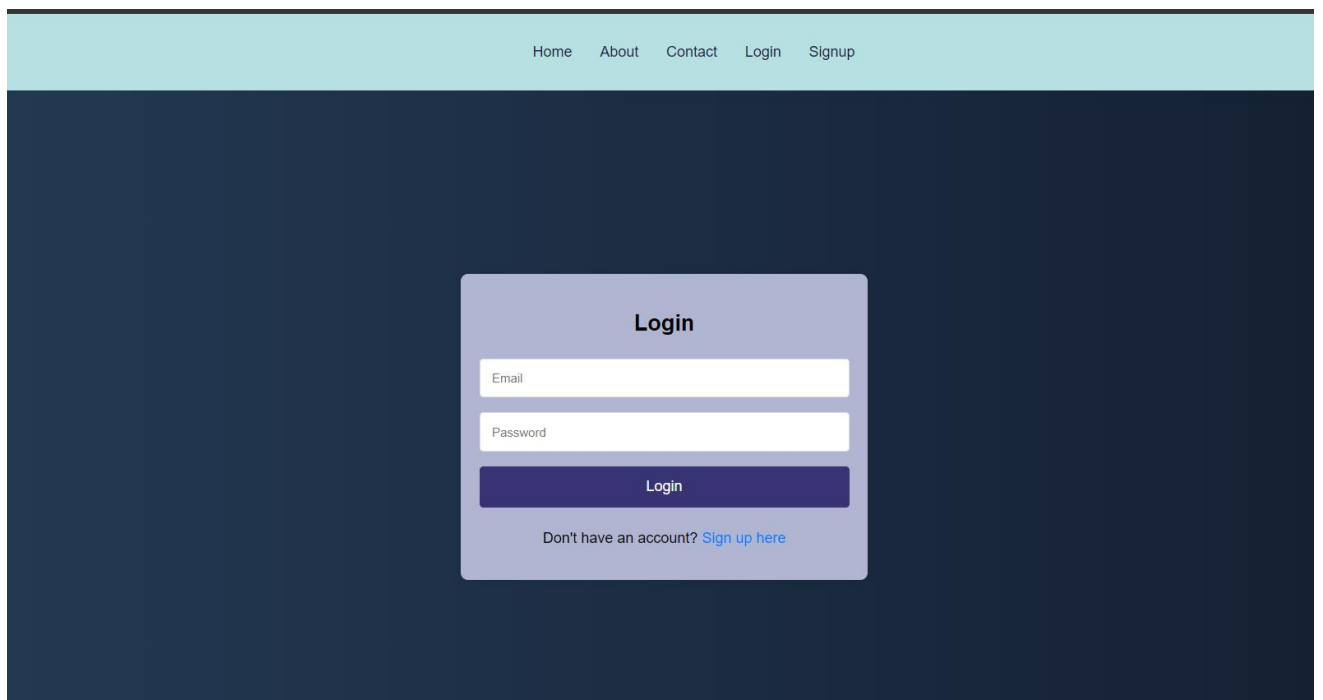


Fig. A.2.3. Admin Dashboard

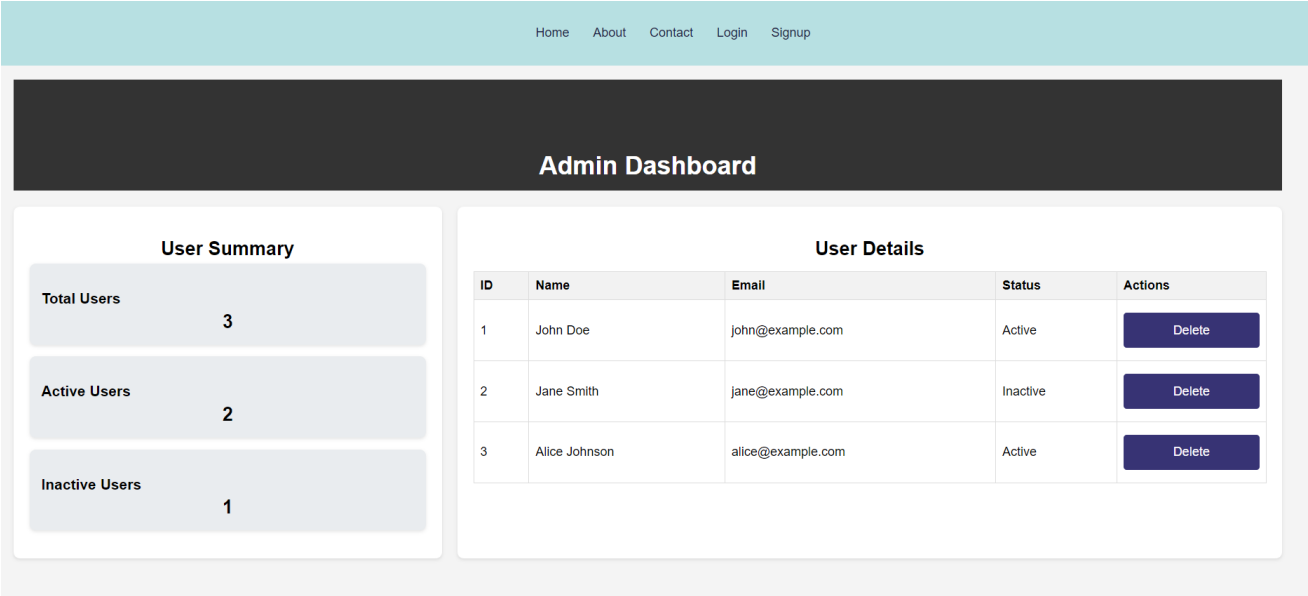
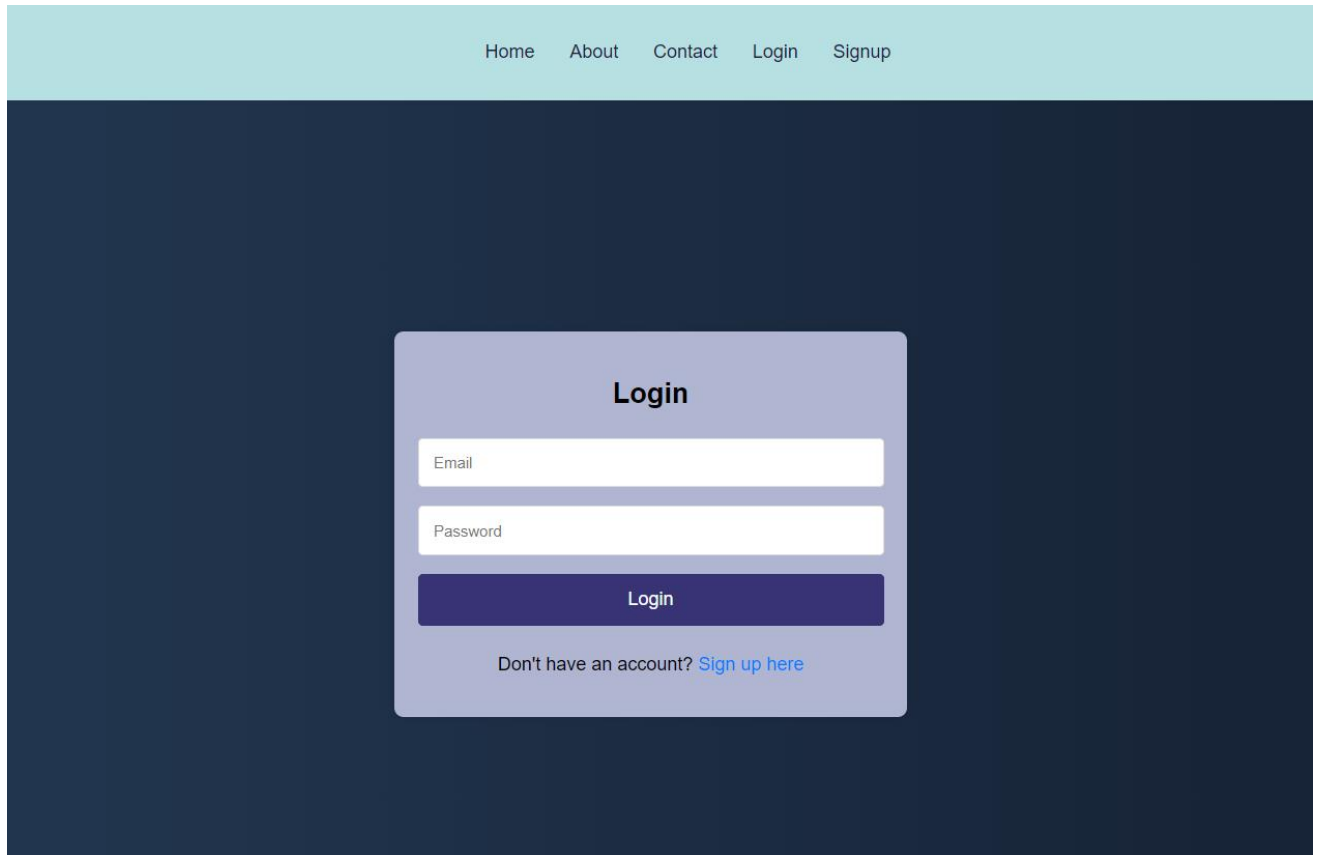


Fig. A.2.4. User Login



The image shows a user login interface. At the top, there is a light blue navigation bar with links for Home, About, Contact, Login, and Signup. The main background is dark blue. In the center, there is a light purple rounded rectangle containing the login form. The form has a title 'Login', two input fields for 'Email' and 'Password', a dark purple 'Login' button, and a link 'Sign up here' for users who don't have an account.

Home About Contact Login Signup

Login

Email

Password

Login

Don't have an account? [Sign up here](#)

Fig. A.2.5. User Registration

Sign Up

Job Seeker
Signup

Company
Signup

Admin
Signup

Name

Email

Password

Course Pursued

Resume

City

Country

Sign Up

Already have an account? [Login](#)

Fig. A.2.6.Joblist page

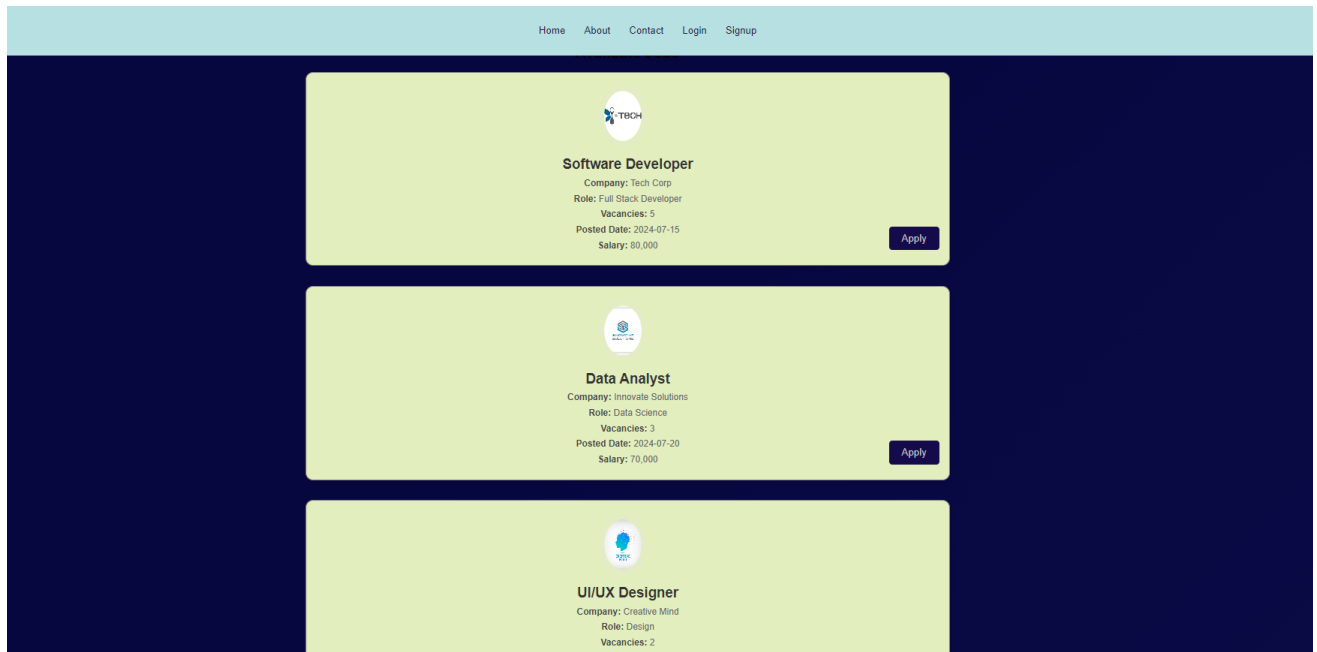


Fig. A.2.7. Application page

Apply for the Job

Full Name

Email Address

Phone Number

Cover Letter

Write your cover letter here...

Submit Application

Fig. A.2.8.Company dashboard

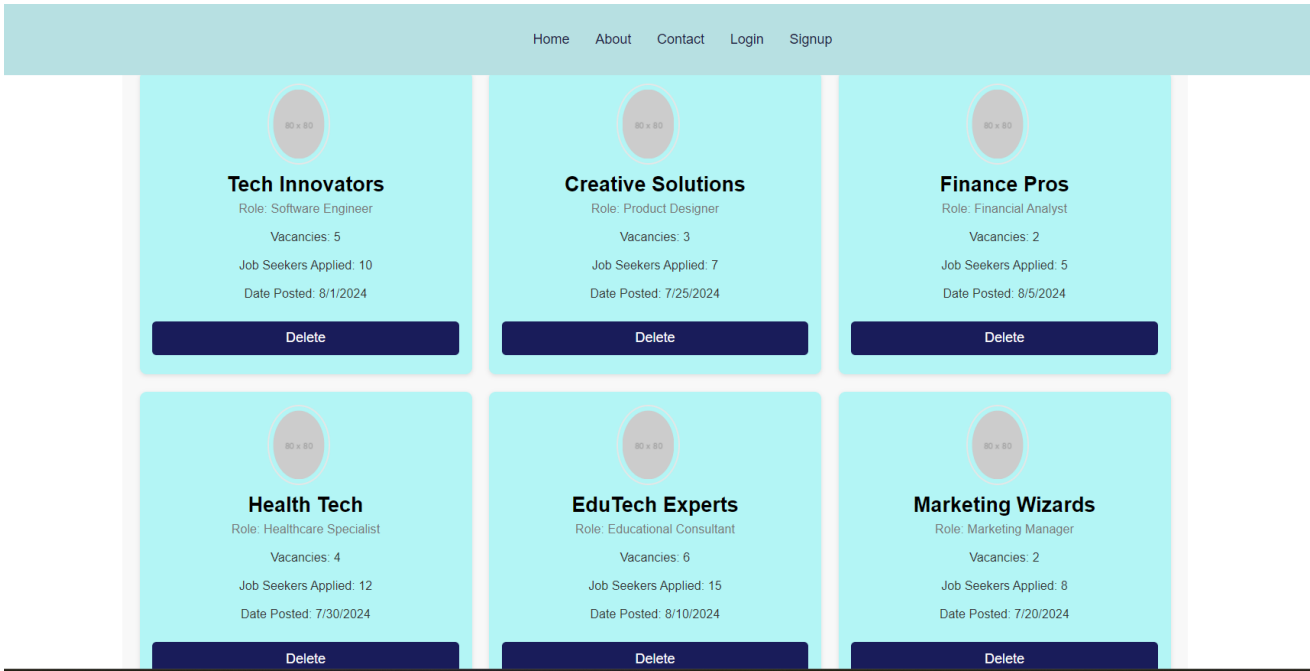


Fig. A.2.9. Contact page

Contact Us

If you have any questions or feedback, feel free to reach out to us using the contact form below or through our email and phone.

Name:

Email:

Message:

Send Message

REFERENCES

Web references:

- [1] PostgreSQL Official Documentation: <https://www.postgresql.org/docs/>
- [2] React Official Documentation: <https://reactjs.org/docs/getting-started.html>
- [3] Swagger Official Documentation: <https://swagger.io/tools/swagger-ui/>
- [4] Java Extensions for Visual Studio Code Official Documentation: <https://code.visualstudio.com/docs/java/extensions>

Book references:

- [1] "eCommerce 2024: Business, Technology, Society" by Kenneth C. Laudon and Carol Guercio Traver Emily Wilson (2022), "The Complete Guide to Home Care for Seniors",
- [2] "eCommerce Essentials: How to Start a Successful Online Business" by Eric Butow and Jacob Forger
- [3] "Don't Make Me Think: A Common Sense Approach to Web Usability" by Steve Krug
- [4] "Digital Marketing for Dummies" by Ryan Deiss and Russ Henneberry
- [5] "Clinical Optics" by Andrew R. Elkington, Frank M. S. Wright, and John J. Sparrow