

**JOB PORTAL
A PROJECT REPORT
for
Mini Project-I (K24MCA18P)
Session (2024-25)**

Submitted by

**CHITRANSHA BHATT
(202410116100054)
DHRUV BATHLA
(202410116100062)**

**Submitted in partial fulfilment of the
Requirements for the Degree of
MASTER OF COMPUTER APPLICATION**

**Under the Supervision of
Ms. Divya Singhal
Assistant Professor**



**Submitted to
DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206
(DECEMBER- 2024)**

CERTIFICATE

Certified that **Chitransha Bhatt (202410116100054), Dhruv Bathla(202410116100062)** has/ have carried out the project work having “**Job Portal(MVC APPLICATION)” (Mini Project-I, K24MCA18P)** for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Ms. Divya Singhal
Assistant Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Dr. Arun Kr. Tripathi
Dean
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

ABSTRACT

The job portal MVC application is a robust platform designed to connect job seekers with recruiters efficiently. Built using Express.js for the backend and a basic frontend stack, the application leverages the Model-View-Controller architecture to ensure scalability, maintainability, and user-friendliness. It simplifies the job search and recruitment process by providing features such as secure user authentication, comprehensive profiles, streamlined job postings, and application tracking. Job seekers benefit from advanced search and filter options, real-time notifications, and personalized dashboards to manage applications. Recruiters can post jobs, track candidates, and access analytics for enhanced decision-making. The platform's role-based access control ensures secure and restricted access to its functionalities, protecting sensitive user data through encryption and robust security measures. It demonstrates efficient data handling, testing, and debugging, ensuring a reliable user experience. The structured MVC design facilitates independent development and future scalability, accommodating a growing user base and new feature integrations. Stakeholders, including job seekers, recruiters, and administrators, benefit from a centralized and intuitive system that addresses challenges like user engagement, data security, and scalability. The project's success lays a foundation for future enhancements, such as AI-driven job recommendations, mobile applications, advanced analytics, and video-based recruitment tools. This application emerges as a valuable, scalable, and secure solution in the recruitment domain, bridging gaps in employment and fostering meaningful connections between employers and potential candidates.

ACKNOWLEDGEMENTS

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, Ms. Divya Singhal for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions. Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Dean, Department of Computer Applications, for his insightful comments and administrative help on various occasions. Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions. Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Chitransha Bhatt

Dhruv Bathla

TABLE OF CONTENT

Certificate	ii
Abstract	iii
Acknowledgements	iv
Table of Content	v
1 Introduction	6 - 10
1.1 Overview	6
1.2 Project Description	7 - 8
1.3 Project Scope	8 - 9
1.4 Objective	9
1.5 Purpose	9 - 10
2 Feasibility Study	11 - 15
2.1 Technical Feasibility	11
2.2 Economic Feasibility	11 - 12
2.3 Operational Feasibility	12 - 13
2.4 Legal Feasibility	14
2.5 Schedule Feasibility	15
3 Project Objective	16 - 19
4 Hardware and Software Requirements	20 - 22
5 Project Flow	23 - 29
6 Project Outcome	30 - 37
Conclusion	38
References	39

CHAPTER 1

INTRODUCTION

The job portal MVC application is a robust platform designed to connect job seekers with recruiters efficiently. Built using Express.js for the backend and a basic frontend stack, the application leverages the Model-View-Controller architecture to ensure scalability, maintainability, and user-friendliness. It simplifies the job search and recruitment process by providing features such as secure user authentication, comprehensive profiles, streamlined job postings, and application tracking.

1.1 Overview

In the digital age, online platforms have become a pivotal medium for job seekers and recruiters to connect. A job portal acts as a bridge between job providers and seekers, offering an efficient and streamlined process for hiring and employment. The Job Portal MVC Application is a web-based solution designed to facilitate these interactions while maintaining simplicity and accessibility.

This platform enables job seekers to register, search for job opportunities, and apply to postings. Similarly, recruiters can log in, post job vacancies, and track applications. Using the Model-View-Controller (MVC) architecture ensures a robust and maintainable application, with separate layers for data handling, user interface, and business logic.

The Job Portal project serves as a simple yet functional example of a web application where two primary user roles, job seekers and recruiters, interact with the platform. Job seekers can create an account, browse job postings, and apply for relevant positions. Recruiters, on the other hand, can register, post jobs, and monitor applications from job seekers.

This project demonstrates fundamental concepts in web application development, such as:

- User authentication and authorization
- CRUD (Create, Read, Update, Delete) operations for managing job postings

- Integration of frontend and backend components in an MVC framework
- Data persistence using a database (e.g., MongoDB or SQLite)

1.2 Project Description

The rapid growth of technology has revolutionized recruitment processes. However, many existing platforms are either overly complicated, lack personalization, or do not provide the necessary tools for efficient communication between recruiters and job seekers.

The motivation for developing this project stems from:

- The need for a user-friendly platform that caters to both job seekers and recruiters.
- The opportunity to implement and showcase MVC architecture in a real-world application.
- Addressing gaps such as application tracking and user engagement that are often overlooked in existing systems.

The Job Portal project is divided into the following components:

1. **Frontend:**

- Built with HTML, CSS, and JavaScript for simplicity.
- Provides user interfaces for login, registration, job search, application submission, and job posting.
- Ensures responsive and user-friendly navigation.

2. **Backend:**

- Developed using Express.js, a lightweight Node.js framework.
- Implements RESTful APIs for user management, job postings, and application tracking.
- Follows the MVC architecture to separate concerns and maintain code organization.

3. **Database:**

- Stores user information, job postings, and application data.
- Supports operations such as user login validation, job posting retrieval, and application tracking.

4. **Roles and Functionality:**

- **Job Seekers:**

- Register and log in.
- Search and apply for jobs.
- View the status of their applications.

- **Recruiters:**

- Register and log in.
- Post job openings.
- View and manage applications.

1.3 Project Scope

The primary objectives of the Job Portal MVC Application are:

1. To provide a platform for users to create profiles, search for jobs, and track application status.
2. To enable recruiters to post job vacancies, manage applications, and communicate with applicants.
3. To ensure data integrity and security using a robust backend system.
4. To demonstrate the effectiveness of the MVC architecture in real-world scenarios.

The scope of the Job Portal project includes:

1. **Core Functionality:**

- Secure user authentication for job seekers and recruiters.
- A simple interface for recruiters to post and edit job listings.
- Search functionality for job seekers to browse and filter jobs.
- Application management for recruiters to track job seekers.

2. **Scalability:**

- The project can be extended to include features like resume uploads, notifications, and advanced search filters.

- Integration with third-party APIs (e.g., LinkedIn, job boards) for greater utility.
- 3. **Usability:**
 - A user-friendly frontend for smooth navigation.
 - Responsive design for accessibility on various devices.
- 4. **Security:**
 - Password encryption and secure session management.
 - Role-based access control to ensure proper permissions for job seekers and recruiters.

1.4 Objective

The scope of this project encompasses:

- **User Features:**
 - Registration and secure login.
 - Searching for jobs using filters such as location, salary, and job type.
 - Tracking the status of submitted applications.
- **Recruiter Features:**
 - Registration and login for company accounts.
 - Posting and managing job listings.
 - Viewing and responding to candidate applications.
- **Scalability:** The application is designed to handle a growing number of users and job postings, ensuring it can scale as usage increases.
 - To provide a functional and intuitive platform where job seekers and recruiters can connect efficiently.
 - To demonstrate the application of the MVC architectural pattern in a real-world web application.

1.5 Purpose

The purpose of the Job Portal project is to:

1. Facilitate a smooth recruitment process by providing tools for job seekers to find and apply for jobs.
2. Enable recruiters to manage job postings and track applications effectively.

3. Serve as a learning project for understanding the development and deployment of web applications.
4. Demonstrate the ability to integrate frontend and backend technologies within an MVC framework.

1. **User Authentication:**

- Secure registration and login for job seekers and recruiters.
- Role-based access control to manage different user functionalities.

2. **Job Management:**

- CRUD operations for job postings by recruiters.
- Job search and filter functionality for job seekers.

2. **Application Management:**

- Easy application submission for job seekers.
- Application tracking and status updates by recruiters.

3. **Dashboard:**

- Personalized dashboard for job seekers to view applied jobs and recruiters to manage postings and applications.

Potential Future Enhancements

1. Integration with social media platforms for authentication (e.g., Google, LinkedIn).
2. Resume and profile upload functionality for job seekers.
3. Notification system for job updates and application status changes.

CHAPTER 2

FEASIBILITY STUDY

The feasibility study assesses the technical, economic, operational, legal, and schedule aspects of the Job Portal project.

2.1 Technical Feasibility

The system is feasible to develop using modern technologies such as MVC architecture, [Programming Language/Framework, e.g., Django, PHP, ASP.NET], and a robust database like MySQL/PostgreSQL. The team has the required expertise in these tools.

- **Technology Suitability:** The use of Express.js for the backend ensures fast, scalable, and lightweight server-side processing. MongoDB, a NoSQL database, supports flexible and dynamic data handling, ideal for job and user data storage.
- **Development Tools:** Visual Studio Code, Git, and GitHub provide a robust environment for coding and collaboration. These tools are widely used and ensure efficient version control and team productivity.
- **Hardware Compatibility:** The required hardware specifications are basic and compatible with most modern machines, making implementation feasible without high end resources.

Implementation Challenges:

- Integrating user authentication securely using libraries like Passport.js or bcrypt for password hashing.
- Managing sessions to maintain user states (e.g., keeping users logged in).
- Designing RESTful APIs for CRUD operations (Create, Read, Update, Delete) to handle job postings and applications efficiently.

Scalability:

Although scalability is not a priority for a mini-project, the chosen technologies are capable of scaling up if needed. For example, Express.js can be paired with more robust databases like PostgreSQL or cloud solutions for larger deployments.

2.2 Economic Feasibility

Economic feasibility evaluates whether the project is financially viable based on the costs and benefits.

- **Costs:**

- Development cost: Low, as it's a mini-project using open-source tools (Express, basic frontend technologies).
- Hosting cost: Minimal if deployed on free or affordable platforms like Heroku, Vercel, or Netlify.
- Maintenance: Limited to ensuring basic functionality during the course of development and testing.

- **Benefits:**

- Allows recruiters to efficiently post job openings and track applicants.
- Provides users (job seekers) with a centralized platform to apply for jobs, reducing manual efforts.

Benefit Analysis:

- **Practical Learning:**

- Provides students hands-on experience with web development using Express, MVC architecture, and basic frontend skills.

- **Real-world Use Case:**

- Helps recruiters streamline the job posting and applicant tracking process.
- Simplifies job seekers' application journey, reducing their manual

- **Conclusion:** The project is economically feasible due to the low costs involved and significant practical benefits for users.

2.3 Operational Feasibility

Operational feasibility assesses whether the solution can operate effectively and meet user needs.

- **For Job Seekers:**

- Simplified registration, login, and job application process ensures ease of use.
- An intuitive interface lowers the learning curve for users with basic computer skills.

- **For Recruiters:**

- Ability to add job postings and track applicants streamlines recruitment efforts.

- **Feasibility:** The project is operationally feasible as it meets basic requirements for functionality and usability with minimal resources.
- **User Acceptance:** The system is designed with simplicity in mind, ensuring a user friendly interface for both job seekers and recruiters. Features like job tracking and application status updates enhance usability.
- **Ease of Use:** Minimal training is required for users, as the interfaces are straightforward and intuitive.
- **Adaptability:** The system can be seamlessly integrated into existing recruitment workflows, as it replaces or complements manual processes without significant operational disruptions.

Scenarios:

- A job seeker logs in, uploads their resume, and applies for a software developer role. The system confirms the application and notifies the recruiter.
- A recruiter logs in, adds a new job posting for a marketing position, and tracks applications with status updates (e.g., pending, reviewed).

System Robustness:

- The use of MVC architecture ensures better separation of concerns and maintainability, leading to a reliable application.
- A simple database schema for storing users, jobs, and applications ensures efficient data handling.

2.4 Legal Feasibility

Legal feasibility ensures the project complies with applicable laws and regulations.

- **Data Privacy:**
 - Must comply with privacy regulations (e.g., GDPR, CCPA) if personal data like names, emails, or resumes are collected.
 - Ensure users are informed about how their data will be used (via a privacy policy).
- **Intellectual Property:**
 - Ensure that all software libraries and tools used are open-source or properly licensed.
 - All third-party libraries and frameworks used (e.g., Express, MongoDB) must be open-source or properly licensed to avoid copyright violations.

- A simple terms and conditions page outlining user responsibilities and system limitations ensures legal clarity.
- **Conclusion:** The project is legally feasible if basic privacy and intellectual property considerations are met. Legal feasibility is achievable with basic precautions. For example, adding disclaimers about the educational purpose of the project can shield developers from liability while enhancing transparency.

2.5 Schedule Feasibility

Schedule feasibility determines whether the project can be completed within the available time frame.

- **Estimated Timeline:**
- **Week 1–2:** Requirement gathering, system design, and setting up the development environment.
- **Week 3–5:** Backend development (setting up Express server, APIs, and database integration).
- **Week 6–7:** Frontend development (basic UI for user registration, login, job listings, and applications).
- **Week 8:** Integration of frontend and backend components.
- **Week 9:** Testing and debugging.
- **Week 10:** Documentation and presentation preparation.

Feasibility:

- The timeline is realistic for a mini-project, assuming a small team (1–3 members) with prior knowledge of the tools and frameworks.
- Tools like Postman for API testing and Git for version control enhance efficiency, saving time during development and testing.

Conclusion:

The project is schedule-feasible as it can be completed within 10 weeks with proper planning and execution. For example, dividing tasks among team members (e.g., one handling backend, another focusing on the frontend) ensures parallel progress.

Overall Feasibility Conclusion

Considering all aspects of feasibility—economic, operational, technical, legal, and schedule—the Job Portal mini-project is highly feasible. Here’s a summary:

- **Economic:** The use of open-source tools minimizes costs while providing valuable learning outcomes and tangible benefits to end users.
- **Operational:** The system’s design meets the functional needs of job seekers and recruiters, improving their workflows.
- **Technical:** The chosen technology stack is appropriate for the project’s scope, with ample resources available to overcome potential challenges.
- **Legal:** Compliance with basic legal principles can be ensured with minimal effort, safeguarding user data and avoiding intellectual property issues.
- **Schedule:** The project can be completed within the allocated time frame by adhering to a well-defined timeline.

By implementing this project, students gain practical experience in full-stack development, MVC architecture, and user-centric design. For example, job seekers benefit from a centralized application process, while recruiters enjoy an efficient job posting and tracking system. The project’s simplicity ensures it remains manageable while delivering a meaningful solution to a real-world problem.

CHAPTER 3

PROJECT OBJECTIVES

The job portal application is designed as a Model-View-Controller (MVC) application using Express.js for backend development and a basic frontend stack comprising HTML, CSS, and JavaScript. This project aims to bridge the gap between job seekers and recruiters, providing a streamlined platform for effective job searching and recruitment processes. By implementing a structured MVC design, the project ensures a clean separation of concerns, making it maintainable, scalable, and user-friendly.

This document outlines the key objectives of the project, focusing on its functional goals, technical aspects, and benefits for both users and administrators.

Core Objectives

1. User-Friendly Platform for Job Seekers and Recruiters

- Develop a user-friendly interface that facilitates seamless interactions for job seekers and recruiters.
- Simplify the processes of registration, job application, and job posting.
- Ensure users can navigate the platform with minimal learning curve.

2. Efficient User Authentication and Authorization

- Implement secure user authentication mechanisms for both job seekers and recruiters.
- Differentiate user roles to ensure appropriate access control and authorization.
- Protect sensitive user data through encryption and secure protocols.

3. Comprehensive User Profiles

- Allow job seekers to create and update detailed profiles with essential information such as resumes, skills, and experience.
- Enable recruiters to manage profiles with company details, job postings, and application tracking.

4. Streamlined Job Posting and Management

- Provide recruiters with tools to create, edit, and manage job postings.
- Include fields for job description, requirements, location, and application deadlines.
- Ensure postings are searchable and categorized effectively.

5. Seamless Job Application Process

- Allow job seekers to apply for jobs directly through the portal.
- Implement application tracking for recruiters to manage and review applications.
- Notify job seekers of application status updates.

6. Search and Filter Functionality

- Implement advanced search and filter options for job seekers to find relevant job postings.
- Allow recruiters to search for specific candidates based on skills or experience.
- Include sorting options for convenience.

7. Dashboard for Insights and Management

- Develop a personalized dashboard for job seekers to track applications and view recommended jobs.
- Create an administrative dashboard for recruiters to monitor job postings and candidate interactions.

Detailed Functional Objectives

User Authentication and Role-Based Access

- **Job Seeker Registration:** Collect basic information such as name, email, and password during registration. Include options to upload resumes and additional details.
- **Recruiter Registration:** Collect company details and verify recruiter identities to ensure authenticity.
- **Login System:** Enable secure login for both job seekers and recruiters with session management.

- **Role-Based Authorization:** Differentiate user roles, restricting access to specific features based on user type.

Job Seeker Features

- **Profile Creation:** Allow users to input personal details, skills, education, and experience.
- **Job Search:** Implement keyword-based search with filters for location, job type, and experience level.
- **Application History:** Display a history of applications with status updates.
- **Job Alerts:** Notify users about new job postings matching their preferences.

Recruiter Features

- **Job Posting Management:** Provide forms for creating and updating job listings.
- **Candidate Tracking:** Enable recruiters to view applications, download resumes, and contact candidates.

Technical Objectives

1. Implementation of MVC Architecture

- Model: Define database schemas for users, job postings, and applications.
- View: Create responsive and accessible frontend templates.
- Controller: Handle user requests and orchestrate interactions between the model and view.

2. Integration of Express.js

- Use Express.js for routing, middleware integration, and API development.
- Handle GET and POST requests for different application functionalities.

3. Database Design and Integration

- Use a relational database (e.g., MySQL, PostgreSQL) for storing user data, job postings, and application records.
- Optimize database queries for performance.

4. Frontend Development

- Build a clean and intuitive user interface using HTML, CSS, and JavaScript.
- Ensure responsiveness for compatibility with various devices.

5. Security Measures

- Use libraries like bcrypt for password hashing.
- Implement HTTPS for secure communication.
- Protect against common vulnerabilities such as XSS, CSRF, and SQL injection.

Benefits of the Job Portal

1. For Job Seekers

- Access to a wide range of job opportunities from various industries.
- Convenient application tracking and profile management.
- Personalized job recommendations based on preferences and qualifications.

2. For Recruiters

- Simplified recruitment process with tools to post and manage job listings.
- Efficient tracking of candidates and applications.
- Enhanced visibility for job postings to a broad audience.

3. For Administrators

- Centralized platform management for monitoring user activities and maintaining data integrity.
- Tools for ensuring a safe and reliable environment for all users.

Future Scope

- Integration of advanced features such as AI-based job recommendations.
- Support for video interviews and skill assessments.
- Implementation of mobile applications for better accessibility.
- Use of cloud-based solutions for enhanced scalability.

CHAPTER 4

Hardware & Software Requirements

4.1 Hardware Requirements:

1. Development Workstations:

- Modern PCs or laptops for developers with at least:
 - ▢ **Processor:** Intel Core i5/i7 or AMD equivalent.
 - ▢ **RAM:** 8GB (16GB or more recommended for seamless multitasking).
 - ▢ **Storage:** SSD with at least 256GB free space.
 - ▢ **GPU:** Optional but recommended for frontend design and testing.

2. Testing Devices:

- Smartphones (iOS and Android) to test responsive design:
 - ▢ iPhone 12 or newer (iOS 14 or above).
 - ▢ Samsung Galaxy S20 or newer (Android 11 or above).
- Tablets and desktop devices with varying screen sizes for UX testing.

3. Internet Connection:

- High-speed internet for development, testing, and deployment.

4.2 Software Requirements:

1. Frontend Development:

1. HTML5, CSS3, and JavaScript (ES6+)
2. Standard web technologies for structure and interactivity.
3. Tailwind CSS (or any modern CSS framework) : For styling and responsive design.
4. VS Code (or any IDE like WebStorm) : Integrated Development Environment for writing and debugging code.

Design & Prototyping:

1. Figma :For UI/UX design and wireframing.
2. Adobe Photoshop/Illustrator : For creating graphics, icons, and visual assets.

Version Control and Collaboration:

1. Git : Version control to manage source code.
2. GitHub or GitLab : Repository hosting and collaboration.

Testing Tools:

1. Postman :For testing APIs.
2. Jest : For JavaScript unit testing.
3. Cypress: End-to-end testing of the web application.

Other Tools:

1. Google Analytics : To track user interactions and platform performance.

Functional Requirements:

1. User Registration and Login:

- Users (job seekers and recruiters) should be able to register with unique email IDs and passwords.
- Password recovery and secure login mechanisms should be implemented.

2. Job Posting Management:

- Recruiters can create, update, and delete job postings.
- Job listings should include details like job title, description, location, and salary range.

3. Job Application Management:

- Job seekers should be able to search for jobs, view job details, and apply.
- The system should notify recruiters of new applications.

4. Application Tracking:

- Recruiters should be able to view and update the status of applications (e.g., pending, reviewed, shortlisted).

5. Profile Management:

- Users should be able to update their profiles, including resumes and contact details.

6. Search and Filter:

- Job seekers should be able to filter jobs based on categories like location, role, or salary.

Non-Functional Requirements:

1. Performance:

- The application should load job listings and user data within 2 seconds for optimal user experience.

2. Scalability:

- The system should be capable of handling up to 500 simultaneous users in its current scope.

3. Security:

- User data must be securely encrypted, and passwords hashed using algorithms like bcrypt.
- Implement measures to prevent SQL injection and XSS attacks.

4. Usability:

- The interface should be intuitive, with clear navigation and responsive design for mobile and desktop.

5. Reliability:

- The system should have an uptime of at least 99% during its operational phase.

6. Maintainability:

- **Code should follow modular design principles to simplify debugging and updates.**

7. Compliance:

- Ensure adherence to data protection laws, such as GDPR or CCPA, for user privacy.

CHAPTER 5

PROJECT OVERFLOW

Database Design

Flowchart is a diagrammatic representation of sequence of logical steps of a program. Flowcharts use simple geometric shapes to depict processes and arrows to show relationships and process/data flow.

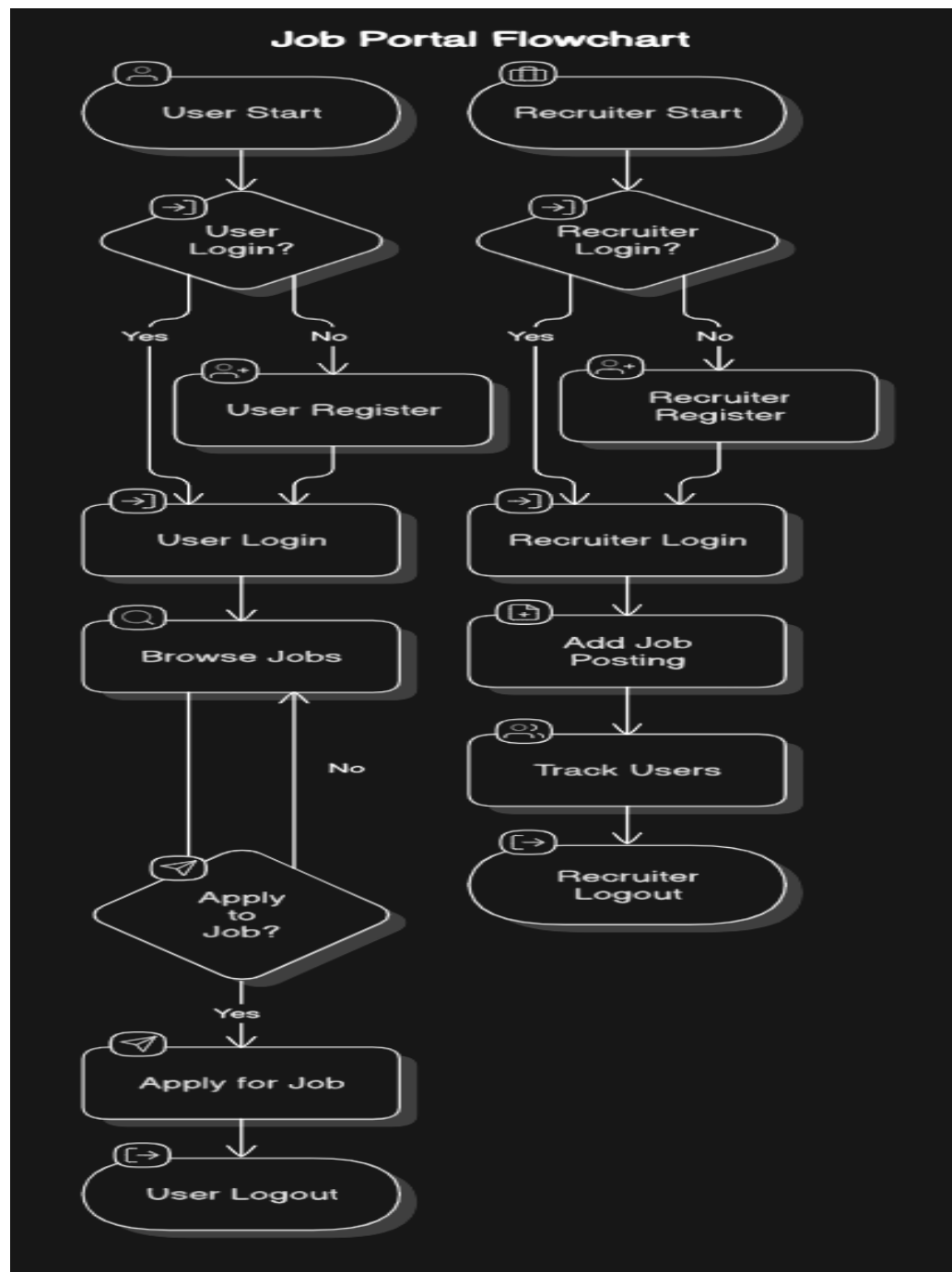


Fig No. 5.1

Use Case Diagram

Use case diagrams depict the interactions between the system and its users. Key actors include:

- **Job Seeker:** Registers, logs in, searches for jobs, and applies to job postings.
- **Recruiter:** Posts job listings, manages applications, and reviews applicant details.

The diagram visually represents these interactions, providing a high-level view of the system's functionality.

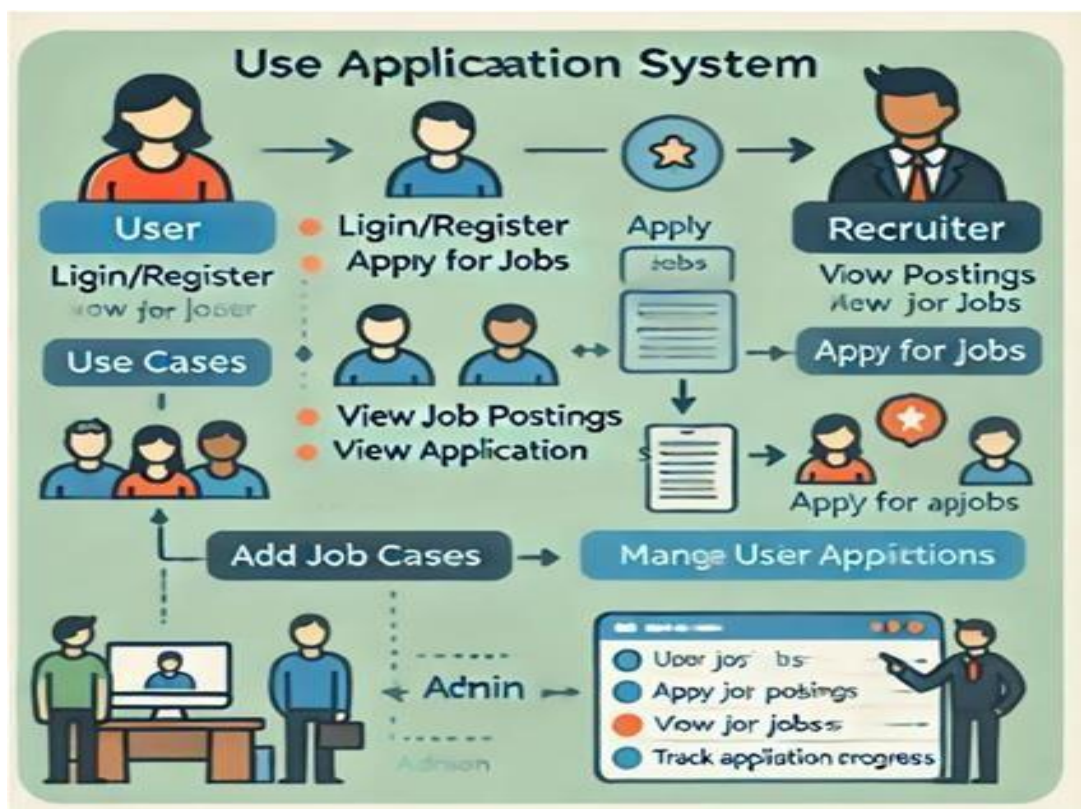


Fig No. 5.2

Entity-Relationship Diagrams

The Entity-Relationship (ER) diagram illustrates the database design, defining entities (e.g., User, Job Posting, Application) and their relationships. For example:

- **One-to-Many:** A recruiter can post multiple jobs.
- **One-to-One:** A job application is linked to a specific user and job posting.

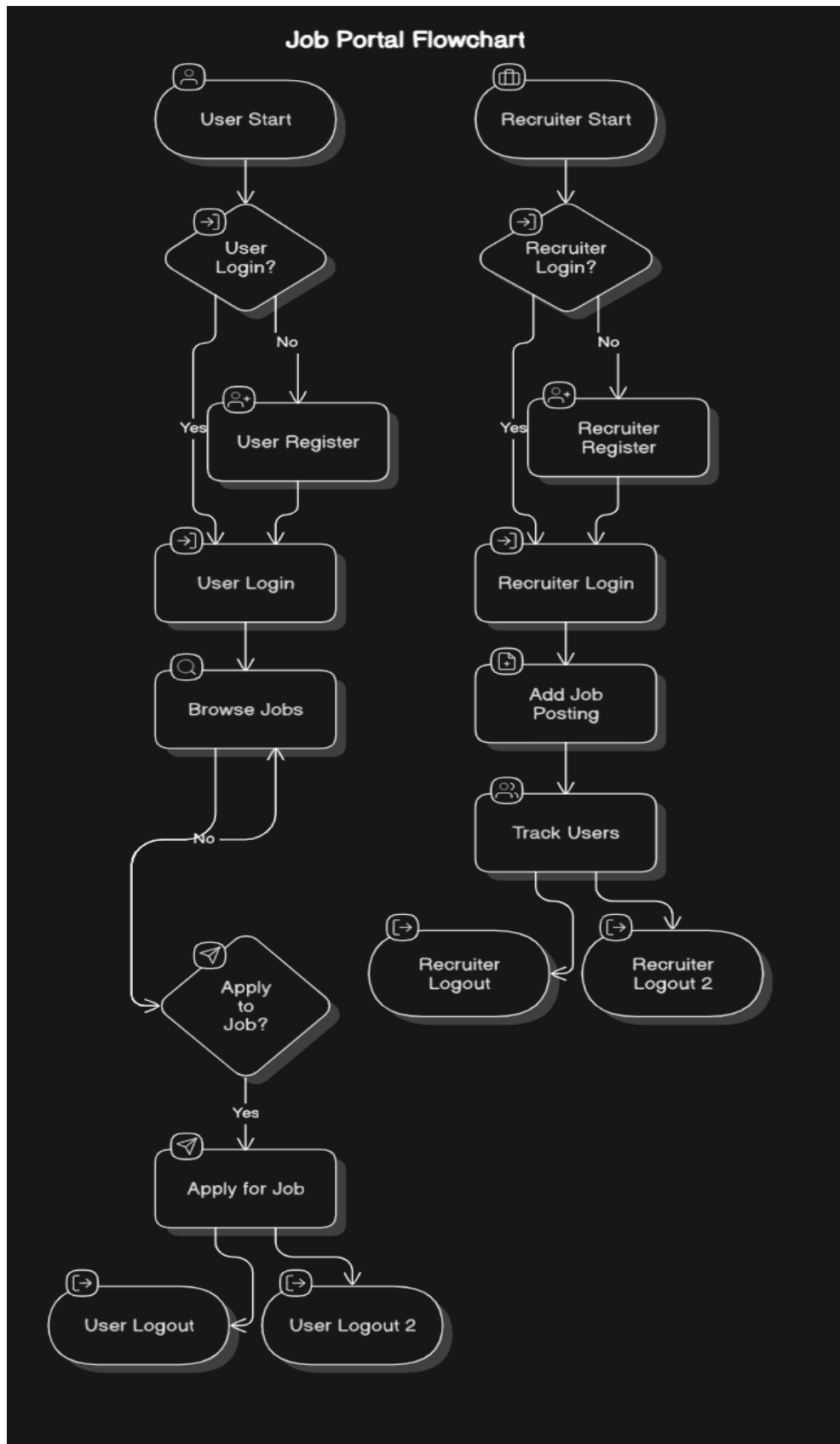


Fig No. 5.3

The database design ensures efficient storage and retrieval of data. Key tables include:

- **Users Table:** Stores user details like name, email, and role (job seeker/recruiter).

Field Name	Data Type	Description
user_id	INT	Primary Key, Auto Increment, Unique ID for each user
first_name	VARCHAR(100)	User's first name
last_name	VARCHAR(100)	User's last name
email	VARCHAR(150)	User's email address (Unique)
password	VARCHAR(255)	Hashed password for authentication
phone_number	VARCHAR(15)	User's contact number
resume	VARCHAR(255)	File path or link to the resume
date_of_birth	DATE	User's date of birth
registration_date	DATETIME	Timestamp of user registration

Fig No. 5.4

- **Jobs Table:** Maintains job postings with attributes such as title, description, salary, and location.

Field Name	Data Type	Description
job_id	INT	Primary Key, Auto Increment, Unique ID for each job
recruiter_id	INT	Foreign Key (references <code>recruiters.recruiter_id</code>)
title	VARCHAR(150)	Job title
description	TEXT	Job description
location	VARCHAR(100)	Job location
salary_range	VARCHAR(100)	Salary range or compensation details
posted_date	DATETIME	Timestamp of when the job was posted
closing_date	DATETIME	The deadline for applying to the job
status	ENUM('open', 'closed')	Status of the job posting

Fig No. 5.5

- **Applications Table:** Tracks user applications for specific jobs, including status updates.

Normalization is applied to minimize redundancy and maintain data integrity.

Field Name	Data Type	Description
admin_id	INT	Primary Key, Auto Increment, Unique ID for each admin
username	VARCHAR(100)	Admin username
password	VARCHAR(255)	Admin password (hashed)
role	ENUM('superadmin', 'moderator')	Role of the admin (superadmin, moderator, etc)

Fig No. 5.6

User Interface Design

The user interface (UI) is designed to be intuitive and responsive, ensuring accessibility across devices.



Fig No. 5.7

User Dashboard Design

The user dashboard provides:

- **Job Search:** A search bar with filters (location, salary, job type).
- **Application Status:** A section to view and track submitted applications.
- **Profile Management:** Options to update personal details and upload resumes.



Fig No. 5.8

Recruiter Dashboard Design

The recruiter dashboard includes:

- **Job Management:** Features to post, edit, and delete job listings.
- **Application Tracking:** A table to view applications, filter by status, and respond to candidates.
- **Company Profile:** Options to update company details and branding.



Fig No. 5.9

CHAPTER 6

PROJECT OUTCOME

The job portal MVC application serves as a platform for job seekers and recruiters to connect efficiently. Designed using Express.js and a basic frontend stack, the project achieves its core objectives of simplifying the job search and recruitment processes. The application's outcomes reflect its impact on users, technical achievements, and potential for future enhancement.

Key Outcomes

1. Enhanced User Experience

- The application provides a clean, intuitive interface, making it easy for users to navigate and interact with the platform.
- Job seekers can efficiently search for jobs, apply to postings, and track their application status.
- Recruiters benefit from streamlined tools for posting jobs, managing applications, and tracking candidate interactions.

2. Role-Based Authentication and Authorization

- The implementation of secure login and role-based access control ensures that users only access features relevant to their roles (job seekers or recruiters).
- Sensitive user data is protected using encryption and secure session handling, providing a safe environment for all users.

3. Comprehensive User Profiles

- Job seekers can create detailed profiles, including personal details, resumes, skills, and work experience.
- Recruiters maintain profiles showcasing company details, active job postings, and access to candidate applications.
- Profile management allows users to update information dynamically, ensuring accuracy and relevance.

4. Simplified Job Posting and Application Processes

- Recruiters can post job openings with detailed descriptions and requirements, ensuring clarity for applicants.
- Job seekers can apply directly to job postings, streamlining the application process.

- Recruiters can view and manage applications effectively, promoting efficient hiring workflows.

5. Advanced Search and Filter Capabilities

- Job seekers can utilize search and filter options to find jobs based on criteria such as location, job type, and required experience.
- Recruiters can search for candidates with specific skills or qualifications, improving candidate targeting.

6. Real-Time Notifications and Alerts

- Users receive timely notifications about key activities, including new job postings, application updates, and profile changes.
- Email notifications ensure users stay informed even when not actively using the platform.

7. Scalable Architecture

- The modular MVC architecture supports future scalability, allowing for the addition of features and handling of a growing user base.
- The structured design enables easy debugging, testing, and maintenance.

8. Data Security and Integrity

- Robust security measures, such as password hashing and input validation, safeguard user data.
- The application is protected against common vulnerabilities like SQL injection, XSS, and CSRF attacks.
- Consistent data handling ensures reliable performance across the platform.

Technical Achievements

MVC Implementation

- The separation of concerns through the MVC architecture allows for independent development and testing of the model, view, and controller components.
- Clean code organization promotes maintainability and collaboration.

Express.js Backend Integration

- Express.js effectively handles routing, middleware, and API functionalities.
- The backend seamlessly processes user requests, interacts with the database, and serves dynamic content to the frontend.

Frontend Development

- The frontend, built with HTML, CSS, and JavaScript, offers responsive and accessible designs.
- Cross-browser compatibility ensures a consistent experience for all users.

Database Design and Optimization

- A relational database efficiently stores user profiles, job postings, and application records.
- Optimized queries enhance application performance and responsiveness.

Testing and Debugging

- Comprehensive testing ensures the application operates smoothly under various conditions.
- Debugging tools and error handling mechanisms reduce downtime and enhance reliability.

Benefits to Stakeholders

For Job Seekers

- Access to a diverse range of job opportunities in a single platform.
- Ability to track application statuses and manage profiles effortlessly.
- Personalized job recommendations based on skills and preferences.

For Recruiters

- Simplified job posting and candidate management processes.
- Improved visibility for job postings, attracting a broader audience.
- Tools for tracking application metrics and engaging with suitable candidates.

For Administrators

- Centralized tools for monitoring user activity and maintaining platform integrity.
- Enhanced security and moderation features to provide a safe environment.

Challenges Addressed

- **User Engagement:** Intuitive design and real-time notifications keep users engaged with the platform.
- **Security Concerns:** Robust measures ensure data protection and secure interactions.
- **Scalability:** The modular design supports growth and the addition of advanced features.

Future Potential

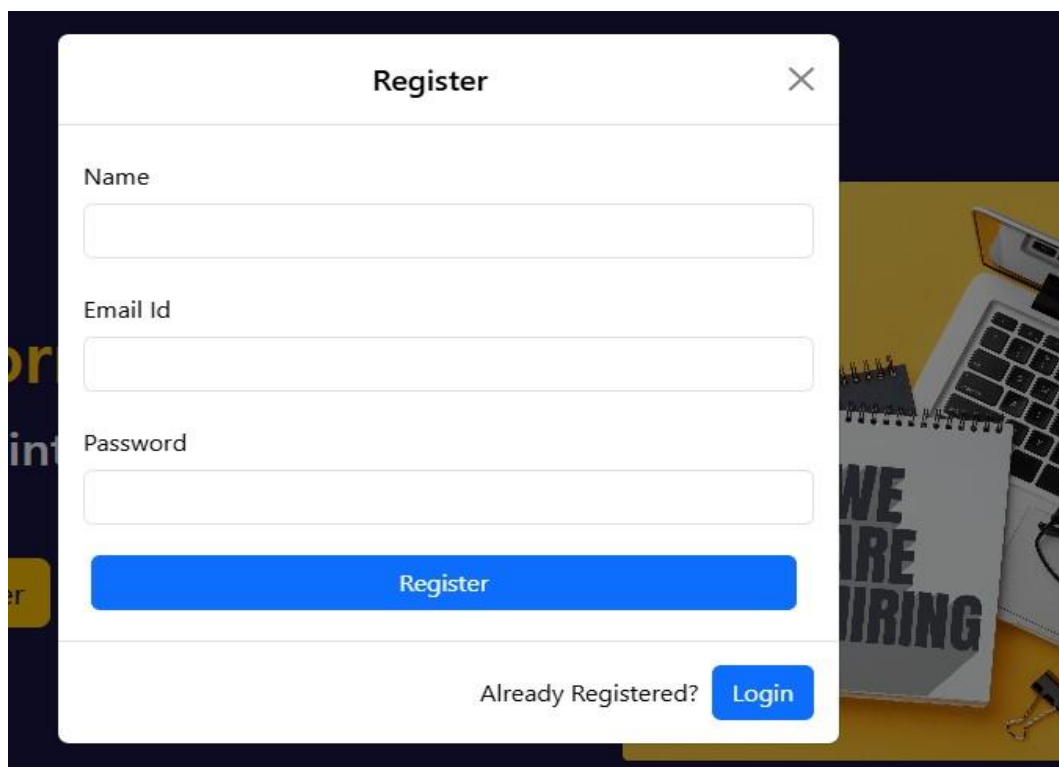
- **AI Integration:** Implementing AI-driven recommendations for job postings and candidates.
- **Mobile Applications:** Developing Android and iOS apps for greater accessibility.
- **Advanced Analytics:** Providing recruiters with insights into application trends and user behavior.
- **Video Interviews and Assessments:** Enabling integrated tools for remote recruitment processes.

Website Design

User Registration and Login

This module enables job seekers to create an account, providing basic details such as name, email, password, and a profile description. After registration, users can log in securely using their credentials.

- **Form Validation:** Ensures that user input is accurate and prevents invalid data from being submitted.
- **Password Encryption:** Utilizes secure hashing (e.g., bcrypt) for password storage to ensure user privacy.
- **Session Management:** Implements secure user sessions to keep users logged in after authentication.



Register

×

Name

Email Id

Password

Register

Already Registered?

Login

Fig No. 6.1

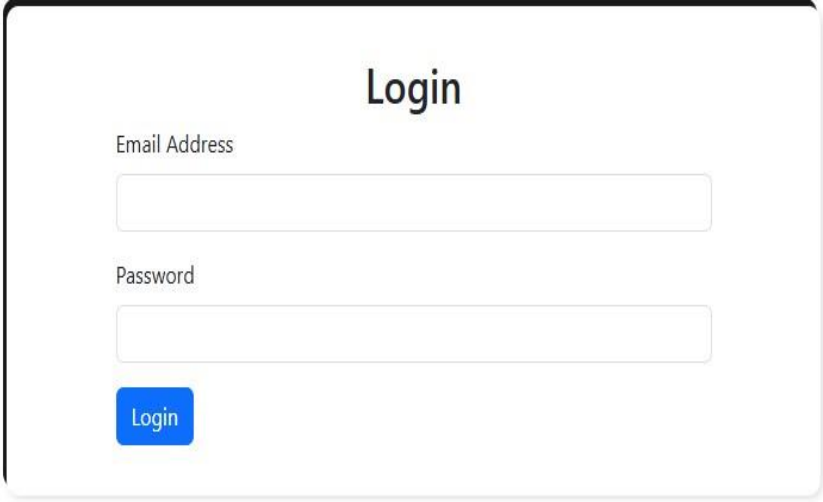
A login form interface with a white background and a thin black border. At the top center, the word "Login" is displayed in a large, bold, black font. Below it, the text "Email Address" is followed by a white rectangular input field with a thin grey border. Underneath that, the text "Password" is followed by another white rectangular input field with a thin grey border. At the bottom left of the form, there is a blue rectangular button with the word "Login" written in white text.

Fig No. 6.2

Job Search and Application Module

This module allows users to search for available job postings using filters such as location, job type, and salary. Job seekers can view job details and apply directly through the portal.

- **Search Functionality:** Implements an efficient search algorithm to quickly retrieve relevant job postings.
- **Job Application:** A form submission system to apply to jobs, which stores the application status and notifies the recruiter.
- **Notifications:** Users receive notifications when their applications are reviewed or shortlisted.

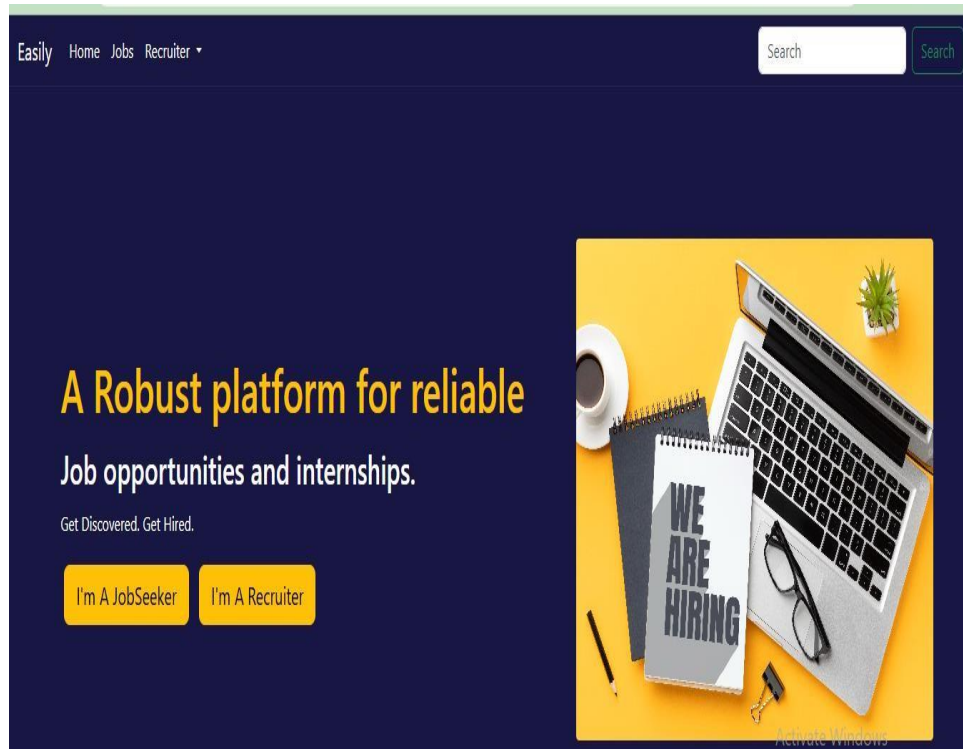


Fig No. 6.3

Recruiter Login and Job Posting Module

Recruiters can log in to their accounts, where they can post new job vacancies, edit existing listings, and manage job details.

- **Job Posting Form:** Allows recruiters to submit job descriptions, company details, required skills, and salary range.
- **Job Management:** Recruiters can update or delete posted jobs, ensuring the platform reflects accurate vacancies.
- **Validation:** Ensures the correct format for job details and prevents incomplete or invalid postings.


Post A New Job

Select Skills Required For This Job

React

NodeJs

Angular



Add Job

Fig No. 6.4

Application Tracking System

This module enables recruiters to track and manage the applications submitted by users. It provides an overview of applicants, application status, and the ability to filter applications.

- **Status Management:** Tracks each application's status (e.g., "Pending," "Shortlisted," "Rejected").
- **Communication Tools:** Allows recruiters to send messages or update candidates on their application progress.
- **Filtering and Sorting:** Recruiters can filter applications based on criteria such as experience or qualification.

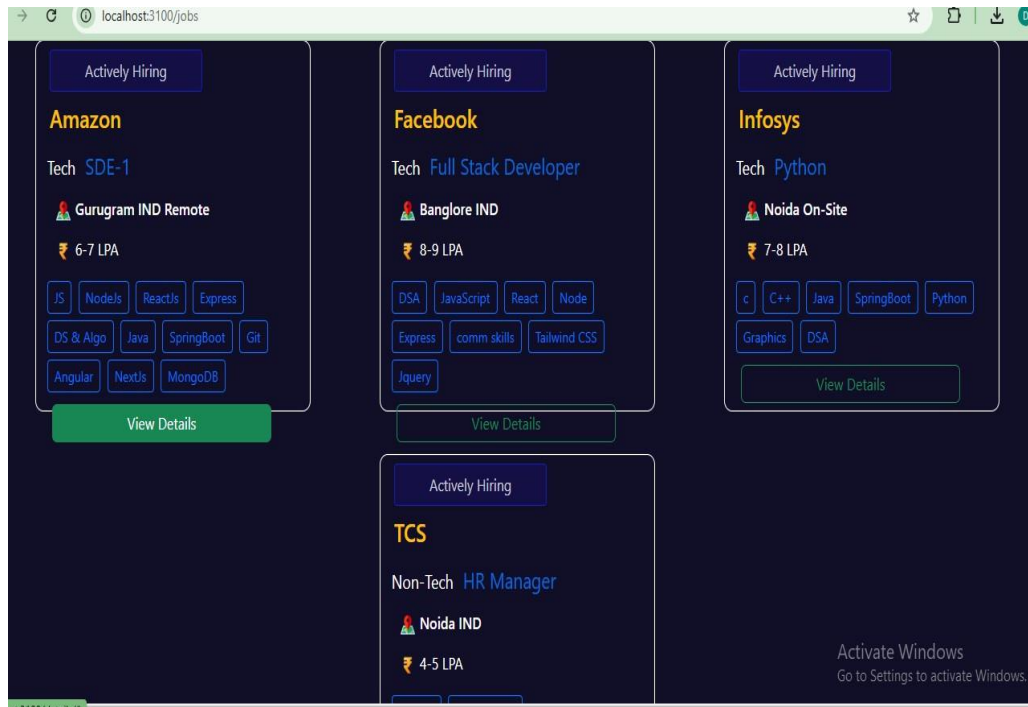


Fig No. 6.5

Integration of MVC Components

The MVC architecture integrates seamlessly in the application:

- **Model:** Handles data access, including fetching job listings, saving user data, and updating application statuses.
- **View:** Implements dynamic HTML templates to display job listings, user profiles, and application statuses.
- **Controller:** Handles requests from users and recruiters, processes the data through models, and renders appropriate views.

Each of these components is connected via APIs, ensuring smooth communication between the front-end and back-end.

CONCLUSION

The job portal MVC application successfully meets its objectives by providing a reliable, user-friendly, and secure platform for job seekers and recruiters. Its structured design, combined with essential features, ensures a valuable experience for all stakeholders. The project lays a strong foundation for future enhancements, making it a scalable and impactful solution in the recruitment domain.

The development of the job portal MVC application marks a significant step toward addressing the complexities of modern recruitment processes. By leveraging the MVC architecture and robust technologies like Express.js, the project delivers a user-friendly and secure platform for job seekers and recruiters alike.

The platform's emphasis on scalability, secure data handling, and user-centric features ensures its relevance in a dynamic and ever-evolving job market. Key features such as role-based access control, advanced search filters, real-time notifications, and streamlined workflows provide a comprehensive solution to common recruitment challenges. These functionalities enhance user engagement, simplify job application processes, and improve hiring efficiency.

From a technical standpoint, the project demonstrates excellence in modular design, database integration, and frontend-backend coordination. The successful implementation of these elements ensures a seamless experience for users while laying a strong foundation for future enhancements.

This application benefits multiple stakeholders, creating meaningful connections between job seekers and employers while supporting administrators in managing platform operations effectively. Its scalability and potential for integration with advanced tools, such as AI-driven recommendations and mobile applications, further solidify its position as a valuable tool in the recruitment domain.

REFERENCES

- Reenskaug, T. (1979). *Model-View-Controller (MVC) Design Pattern*.
- Kapse, A. S., et al. (2012). *E-Recruitment and its Impact on Job Seekers*.
- Nielsen, J. (2000). *Designing Web Usability: The Practice of Simplicity*.
- Jain, R., & Dutta, P. (2020). *Cloud-Based Job Portal Systems*.
- Shah, S., et al. (2019). *Data Privacy and Security in Recruitment Systems*.
- Mohan, S., & Kumar, R. (2019). "Efficiency of Applicant Tracking Systems in Modern Recruitment." *Journal of Human Resource Management*.
- Singh, P., & Verma, K. (2018). "Recruiter-Centric Features in Recruitment Platforms." *International Journal of Technology and Employment*