PROJECT SYNOPSIS REPORT

ON

Job-Portal Application

SUBMITTED

TO

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FOR

Full Stack Engineering (22CS037)

Submitted To:-
Mr. Rahul.

Submitted By:-
Asmita Sharma (2310990307).
Balreet Singh (2310990310).
Bhavisha Ahuja (2310990312).
Dhruv Kumar (2310990325).

CHITKARA UNIVERSITY

Index

Problem Statement:

In the current competitive job market, employers struggle to manage recruitment operations such as creating job postings, evaluating applicants, and coordinating hiring stages. Job seekers face difficulties accessing a unified system that supports resume building, interview preparation, and direct job applications. Most existing platforms provide these features separately, resulting in inefficiencies, poor user experience, and missed opportunities.

The **Job-Portal** project resolves these issues by providing a centralized, feature-rich platform that connects employers and job seekers. The system integrates role-based access, job posting workflows, resume-building tools, interview-preparation modules, application analytics, and a notification system built on the MERN stack (MongoDB, Express.js, React.js, Node.js).

Title of Project:

Job-Portal

Objective & Key Learnings:

Objectives

- Develop a secured role-based authentication system for employers and job seekers.
- Enable employers to post jobs, manage applications, and streamline hiring.
- Provide job seekers with resume builder, interview preparation tools, and job search features.
- Implemented notifications for application acceptance/rejection.
- Add an Analytics Dashboard for employers and admins to analyze job posting performance and user activity.
- Learn seamless integration of React frontend with Node.js backend and MongoDB.

Key Learnings

- JWT authentication and secure user role management.
- End-to-end MERN stack integration.
- Building scalable APIs and modular architecture.
- Database schema design for jobs, applications, analytics, and notifications.
- Implementing charts and dashboards for analytics.

Options Available to Execute the Project:

- **Monolithic Architecture** – Faster to build, single backend for all modules.
- **Microservices Architecture** – Modular division of employer, user, job, analytics, and notification services.
- **React + CSS** – Clean UI with reusable components.
- **MongoDB Database** – Flexible schema for job postings and applications.
- **CI/CD** – GitHub Actions/Jenkins for automated deployment.

Tech Stack:

We will use React, Node.js, Express, and MongoDB due to the following reasons:

React

- Component-based UI
- Fast rendering via Virtual DOM
- Strong ecosystem and reusable components

Node.js & Express

- Handles asynchronous requests efficiently
- REST APIs for authentication, jobs, applications, analytics
- Scalable for high user load

MongoDB

- NoSQL model for flexible, dynamic data
- JSON-like documents integrate naturally with Node.js
- High performance and scalability

Advantages and Disadvantages:

Advantages:

- Unified Platform for job search, resume creation, interview prep, and hiring.
- Role-Based Dashboards for employers, users, and admins.

- Scalability due to MERN stack.
- Notifications for application status updates.
- Analytics Dashboard providing insights into job performance, application trends, and user activity.

Disadvantages:

- Complex Implementation: Role-based features and multiple dashboards increase development time.
- Data Dependency: Performance relies on accurate job postings and user data.
- High Maintenance: Requires regular updates for resume templates, interview questions, and security patches.

Implementation Strategy:

1. Backend Development:

- Build REST APIs with Node.js + Express for authentication, job posting, and application handling.
- Use MongoDB to store user profiles, job postings, applications, and notifications, analytics logs collection.
- JWT-based authentication for secure login.

2. Employer Module:

- Role-based login to Employer Dashboard.
- Features: Post Job (title, description, company, salary, location, job type).
- Manage Applicants: View applicant list, accept/reject requests.
- Store job postings and applicant details in database.
- View Analytics

3. User Module:

Role-based login → User Dashboard

Features:

- Resume Builder with downloadable templates
- Interview Preparation:
    - DSA questions
    - Aptitude questions

- HR interview questions
- Job Search and Apply
- Track application status in real time
- Personalized notifications

4. Notifications:

1. Triggered when employer:

   Accepts user application

   Rejects user application

   Updates job status

2. Notification stored in database + shown in UI

5. Frontend Development:

- React Router for navigation
- CSS for styling (no Tailwind as per user preference)
- Forms for job posting, resume builder, interview prepration
- Employer & User dashboards designed separately
- Notification icon, analytics charts, and job management panel

6. Security Measures:

- JWT authentication for every route
- HTTPS for secure communication
- Input validation & sanitization
- Password hashing with bcrypt
- Role-based route protection
- CORS configuration

7. Testing and Deployment:

- Perform unit, integration, and UAT (User Acceptance Testing).
- Use Docker for containerization and Kubernetes for orchestration.
- Deploy on cloud (AWS/Heroku/Render) for scalability and availability

Conclusion:

The **Job-Portal** provides a comprehensive, modern, and scalable solution for employers and job seekers. By combining job postings, resume generation, interview preparation, real-time notifications, and analytics dashboards, it creates an end-to-end recruitment ecosystem.
Employers benefit from streamlined hiring workflows, real-time applicant management, and data-driven insights through analytics. Job seekers receive a complete toolkit to enhance their profile, prepare for interviews, and stay updated through notifications.
The MERN stack ensures scalability, responsiveness, and efficient development, making this project suitable for real-world deployment.

References:

- **https://nodejs.org/en**
- **https://react.dev/**
- **https://expressjs.com/**
- **https://www.mongodb.com/docs/**
- **https://www.geeksforgeeks.org/mern-stack/**
- **https://developer.mozilla.org/en-US/docs/Learn**