JOB APPLICATION TRACKER SYSTEM

I. ABSTRACT

In the current digital era, tracking job applications manually or through scattered tools results in inefficiencies and lost opportunities. This project presents a centralized and intuitive **Job Application Tracker System** designed to help users streamline the job search process. The system enables users to add, update, and categorize job applications while managing application status, deadlines, and progress. This full-stack web application was developed using the **MERN stack** (MongoDB, Express.js, React.js, and Node.js), offering a scalable, responsive, and user-friendly interface. It includes features such as filtering by status, automated date tracking, and persistent storage. The system addresses a critical gap for job seekers looking for a single platform to manage applications and improves efficiency through automation and dynamic status management.

II. INTRODUCTION

A Job Application Tracker is essential for active job seekers who apply to multiple roles across various platforms. Conventional tools like spreadsheets lack functionality for timely reminders, filtering by custom criteria, and status insights. Our solution provides a dynamic web-based interface integrated with a persistent backend to handle CRUD operations, status changes, and deadline alerts. The system architecture separates concerns through a RESTful API backend and React frontend. MongoDB is used for data persistence, ensuring flexibility and scalability. This system provides end-to-end support for application lifecycle tracking.

III. SYSTEM REQUIREMENTS

A. Hardware Requirements

- System: Desktop or Laptop
- RAM: Minimum 4 GB
- Storage: Minimum 10 GB free disk space
- Processor: Intel i3 or higher
- Network: Stable internet connection

B. Software Requirements

Frontend:

- React.js (v18+)
- HTML5, CSS3 (with Bootstrap 5)

- Axios for API calls

Backend:

- Node.js (v18+)
- Express.js
- MongoDB (via Mongoose)
- dotenv for environment variables
- Nodemon for development

Development Tools:

- Visual Studio Code
- MongoDB Compass
- Postman
- Git & GitHub
- Render (backend hosting) & GitHub Pages (frontend)

IV. SYSTEM DESIGN AND ARCHITECTURE

The system is divided into a **frontend client** and a **backend server**, communicating via HTTP requests using REST API.

A. Frontend

The React client presents:

- A dynamic dashboard with sortable job cards
- Forms for creating and updating application details
- Filters to display jobs by status (e.g., Applied, Interview, Offer)
- Responsive design using Bootstrap grid layout
- Context API for state management across components

B. Backend

The Express.js backend provides:

- RESTful endpoints (/api/jobs) for full CRUD functionality

- MongoDB data schema for job entries
- Job statuses: Applied, Interviewing, Rejected, Offered, etc.
- Middleware for error handling and CORS

C. Data Flow

The frontend collects input from the user, sends it to the server, which stores it in MongoDB. All updates reflect in real-time via state refreshes post-response.

V. IMPLEMENTATION MODULES

The system comprises the following functional modules:

A. User Interface Module

- Home Page
- Job List Dashboard
- Add New Job Form
- Edit Job Form
- Status Filter Component

B. Job Management Module

- Create Job Entry: Title, Company, Location, Link, Notes, and Status
- Edit/Update Existing Entries
- Delete Application
- View All Applications with Sorting

C. Status Filtering & Tracker

- Filters by: Application Status, Deadline, Date Applied
- Automatic sorting by application date

D. Backend API Module

- GET /api/jobs – Fetch all jobs
- POST /api/jobs – Add new job
- PUT /api/jobs/:id – Edit job

- DELETE /api/jobs/:id – Delete job

VI. TESTING AND DEPLOYMENT

A. Testing

- Frontend tested with React Testing Library
- Backend tested using Postman for API validation
- Manual UI/UX testing performed across browsers

B. Deployment

- Frontend deployed on GitHub Pages
- Backend deployed on Render with persistent MongoDB Atlas cluster
- GitHub Actions configured for CI/CD

VII. RESULTS AND DISCUSSION

The Job Application Tracker system successfully streamlines the job search process by eliminating the overhead of scattered tracking. Users can view all their applications in one place, edit them dynamically, and track their progress through intuitive status labels. Unlike traditional spreadsheets, the system offers status-based filtering and live updates. Deployment and responsiveness tests confirm the system performs reliably across platforms and browsers.

VIII. CONCLUSION

This project demonstrates the practical application of full-stack development to solve a real-world productivity challenge faced by job seekers. By leveraging MERN stack technologies, the system provides a responsive, efficient, and easy-to-use interface for tracking applications. The modular design allows future extensibility such as calendar integrations, notification systems, or resume uploads. The Job Application Tracker stands out as a personalized assistant for organizing the job search journey efficiently.

IX. FUTURE SCOPE

1. **User Authentication:** Integrate secure login system for personalized tracking
2. **Resume & Cover Letter Uploads:** Attach documents to each job entry

3. **Notification System:** Reminders via email or SMS for upcoming interviews or deadlines
4. **Analytics Dashboard:** Visualize job trends, response rates, and offer ratios
5. **Export/Import Options:** Download application data as CSV/Excel