Final Year Project Report

CareerConnect: A Web-Based Recruitment Platform with Real-Time Communication

Submitted by: Kiran Timalsina
Student ID: 2330491

Supervised by: Utsav Dahal

Department of Computer Science
University of Wolverhampton (via Herald College Kathmandu)

March 21, 2025

Title and Declaration

Project Title: CareerConnect: A Web-Based Recruitment Platform with Real-Time Communication

I, Kiran Timalsina, hereby declare that this report is my original work, conducted under the supervision of Utsav Dahal. All sources used are duly acknowledged, and no part of this work has been submitted elsewhere for assessment.

- **Signature:**

Date: $March 21, 2025$

Abstract

CareerConnect is a web-based recruitment platform designed to streamline the hiring process by facilitating seamless interaction between recruiters and candidates. The system integrates key functionalities such as job posting, application management, real-time chat, user authentication, and document processing, aiming to address inefficiencies in traditional recruitment methods. Built using modern web technologiesNode.js for the backend, React.js for the frontend, MySQL for database management, and Socket.io for real-time communicationCareerConnect offers an intuitive user experience for both recruiters and candidates. The platform supports features like live chat for instant communication, notifications for application updates, and secure user management. While the current implementation focuses on core functionalities, future enhancements include AI-powered job matching and video interviewing capabilities. This report details the projects development process, from requirement gathering to implementation, following the Waterfall methodology. It evaluates the systems design, technologies used, and potential for scalability, while reflecting on challenges, such as internet dependency and the absence of AI features, providing a foundation for future improvements.

Contents

List of Figures

Chapter 1

Introduction

1.1 Project Briefing

1.1.1 Overall Working, Problem Domain, and Technical Solution

CareerConnect is a web-based application designed to enhance the recruitment process by bridging the communication gap between recruiters and candidates. The problem domain lies in traditional recruitment systems, where inefficiencies such as delayed communication, manual application tracking, and lack of real-time interaction often hinder the hiring process. CareerConnect addresses these issues by providing a centralized platform with features like job posting, application management, real-time chat, and user profile management. Recruiters can post job openings, manage applications, and communicate directly with candidates, while candidates can search for jobs, apply with ease, and engage in instant conversations with recruiters. The technical solution leverages modern web technologies: Node.js and Express.js for a scalable backend, React.js for a dynamic frontend, MySQL for data management, and Socket.io for real-time communication. This architecture ensures a seamless user experience, addressing the inefficiencies of traditional methods.

1.1.2 AI Implementation (Future Scope)

CareerConnect has the potential to integrate artificial intelligence (AI) to enhance its functionality, though this has not been implemented in the current version.

AI Aspect Addressed

The planned AI implementation focuses on machine learning, specifically targeting supervised learning for job-candidate matching. This involves analyzing candidate profiles (e.g., skills, experience) and job requirements to recommend the best matches.

Justification of Learning Type

Supervised learning is appropriate because historical data on successful hires can be used as labeled training data. For example, a dataset of past job applications, including candidate profiles and hiring outcomes, can train a model to predict suitability.

Mathematics and Models

The AI system would use feature extraction to convert textual data (e.g., resumes, job descriptions) into numerical vectors using techniques like TF-IDF (Term Frequency-Inverse Document Frequency). A similarity metric, such as cosine similarity, would measure the alignment between candidate and job vectors:

$$\text{Cosine Similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the vectors representing a candidates profile and a job description, respectively. A logistic regression model could then classify matches as suitable or not suitable, with the probability of suitability given by:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n)}}$$

where $\mathbf{x}$ is the feature vector, and β_i are the model parameters.

Agent Description

The AI agent would be a software component within CareerConnect, acting as a reactive agent that processes input data (candidate profiles, job listings) and outputs match recommendations. It would operate in a stateless environment, responding to user queries without maintaining memory of past interactions.

1.2 Aims

The primary aim of CareerConnect is to develop a seamless, real-time communication platform that revolutionizes the recruitment process by enhancing efficiency and engagement between recruiters and candidates. By addressing the limitations of traditional job portals, such as delayed communication and lack of direct interaction, the project seeks to create a user-friendly and scalable solution for modern hiring needs.

1.3 Objectives

The objectives of CareerConnect include:

- Facilitating efficient job posting and application management for recruiters.
- Enabling candidates to search, filter, and apply for jobs seamlessly.
- Providing real-time communication through a live chat feature to reduce hiring delays.
- Ensuring secure user authentication and role-based access control.
- Laying the groundwork for future AI integration to enhance job-candidate matching.

These objectives focus on solving practical recruitment challenges while ensuring scalability and user satisfaction.

1.4 Artefact

1.4.1 Functional Decomposition Diagram (FDD)

The FDD breaks down CareerConnect into core subsystems, as outlined in the draft report:

- **User Management**: Handles registration, login, and profile management for candidates and recruiters.
- **Job Management**: Manages job postings, searching, filtering, and applications.
- **Communication Module**: Enables real-time chat between recruiters and candidates.
- **Document Processing**: Supports resume uploads, parsing, and verification.
- **Notifications**: Sends updates on application status, messages, and system reports.

[Placeholder for Figure 1: FDD Diagram]

1.4.2 System as a Whole

CareerConnect is a full-stack web application with a client-server architecture. The frontend, built with React.js, provides a responsive and interactive user interface. The backend, developed using Node.js and Express.js, handles business logic, API requests, and database operations. MySQL serves as the relational database, storing user data, job postings, applications, and chat messages. Socket.io enables real-time, bidirectional communication for the chat feature. The system ensures secure data handling through JWT-based authentication and encrypted storage, making it a robust solution for recruitment needs.

1.4.3 Subsystems

User Management

This subsystem manages user registration, login, and profile updates. Candidates and recruiters can sign up using their email and password, with role-based access control ensuring that recruiters can post jobs while candidates can apply for them. Profile management allows users to update personal and professional details, enhancing user engagement.

Job Management

The job management subsystem enables recruiters to post job openings with details like title, description, location, salary, and skills. Candidates can search for jobs using filters (e.g., location, salary range) and apply by uploading resumes. The subsystem also allows recruiters to view and manage applications, streamlining the hiring process.

Communication Module

The communication module, powered by Socket.io, facilitates real-time chat between recruiters and candidates. Users can send and receive messages instantly, share multimedia content (e.g., resumes), and engage in group discussions, reducing communication delays and enhancing interaction.

Document Processing

This subsystem handles resume uploads and parsing. While the current implementation supports file uploads (as seen in ‘JobController.js‘ with ‘upload.single("photo")‘), future enhancements include OCR-based parsing using libraries like Tesseract to extract structured data from resumes, such as skills and experience.

Notifications

The notification subsystem alerts candidates about application status updates (e.g., pending, accepted, rejected) and informs recruiters of new applications and messages. System reports, such as job posting statistics, are also generated to provide insights into platform usage.

1.5 Academic Question

The academic questions posed in the draft report are expanded here:

- **In what ways can this system improve communication between recruiters and candidates?** CareerConnect enhances communication by providing a real-time chat feature using Socket.io, eliminating delays associated with email-based communication. It allows recruiters to clarify job requirements, schedule interviews, and provide feedback instantly, while candidates can ask questions and receive timely responses, improving engagement and reducing hiring time.
- **How can real-time messaging impact the recruitment process?** Real-time messaging streamlines recruitment by enabling instant interaction, reducing the time to hire, and improving candidate experience. It facilitates quick screening, interview scheduling, and decision-making, as evidenced by the chat functionality in ‘RecruiterDashboard.js‘.
- **What are the difficulties in having a scalable chat feature?** Scalability challenges include handling concurrent users, ensuring low latency, maintaining data consistency, and preventing server overload. CareerConnect addresses these using WebSockets (Socket.io), but future improvements may involve load balancing and caching mechanisms.
- **How does the system ensure data security and privacy?** The system uses JWT tokens for authentication (as seen in API endpoints in ‘JobController.js‘), encrypts sensitive data in the database, and implements role-based access control to protect user information.

1.6 Scope and Limitation

1.6.1 Scope

CareerConnect focuses on providing a web-based platform for recruitment with features like job posting, application management, real-time chat, and user authentication. It targets recruiters and candidates, aiming to reduce hiring delays through instant communication and streamlined processes. The system is designed to be scalable, with plans for future enhancements like AI-based job matching and video interviewing.

1.6.2 Limitations

- **Text-Based Communication Only**: The current chat feature supports only text messaging, lacking voice or video capabilities, which could enhance candidate evaluation.
- **Internet Dependency**: The platform requires a stable internet connection for real-time features, limiting accessibility in areas with poor connectivity.
- **Lack of AI Features**: AI-based job matching is not implemented, which could improve efficiency by automating candidate selection.
- **Limited Mobile Optimization**: The platform is primarily web-based, with limited support for mobile devices, potentially reducing accessibility.

These limitations provide opportunities for future development, such as integrating WebRTC for video calls and implementing offline messaging capabilities.

1.7 Report Structure

This report is organized as follows: Chapter 1 introduces the project, its aims, objectives, and scope. Chapter 2 reviews relevant literature on recruitment platforms, real-time communication, and OCR technologies. Chapter 3 discusses the Waterfall methodology chosen for development. Chapter 4 details the technologies and tools used. Chapter 5 presents the artefact designs, including subsystems and future AI integration. Chapter 6 concludes the project, reflecting on aims and findings. Chapter 7 critically evaluates the project, including self-reflection. Chapters 810 cover project management evidence, references, and appendices.

Chapter 2

Literature Review

2.1 Research Basis

The literature review forms the foundation for CareerConnect by analyzing existing recruitment platforms, real-time communication technologies, and document processing techniques. It compares and contrasts findings to identify best practices and gaps that CareerConnect aims to address, without directly answering the academic questions but providing a basis for them.

2.2 Similar Systems

Platforms like LinkedIn and Zoho Recruit offer features similar to CareerConnect, such as job postings and candidate tracking. LinkedIn integrates real-time messaging and uses AI for job recommendations, while Zoho Recruit provides advanced dashboards for recruitment analytics. CareerConnect differentiates itself by focusing on real-time chat integration within a streamlined dashboard, as seen in 'RecruiterDashboard.js', which combines job management and communication in a single interface.

2.3 OCR and Document Processing

Smith and Taylor (2020) highlight the role of OCR in resume parsing, emphasizing preprocessing techniques like noise removal and segmentation to improve accuracy. They recommend Tesseract OCR for its adaptability, which aligns with CareerConnects planned document processing subsystem. The platform could use NLP to normalize extracted data, addressing challenges like multilingual text recognition, as discussed in the draft report.

2.4 Real-Time Communication

Patel (2022) evaluates WebSockets and WebRTC for real-time communication in recruitment platforms, noting their impact on candidate engagement and hiring speed. CareerConnect adopts WebSockets via Socket.io, as evidenced by its chat functionality, ensuring low-latency messaging. Future integration of WebRTC for video calls could further enhance communication, addressing the limitation of text-only chat.

2.5 Data Privacy and Security

Boldi (2024) stresses the importance of transparency in data handling for e-recruitment platforms. CareerConnect implements JWT-based authentication (seen in 'JobController.js') and encrypted database storage to protect user data, aligning with GDPR principles. However, challenges remain in balancing security with user convenience, such as optimizing encryption for real-time chat scalability.

2.6 Data-Driven Recruitment

Miller (2021) discusses the role of dashboards in recruitment, using visualizations like bar charts to identify hiring bottlenecks. CareerConnects recruiter dashboard ('RecruiterDashboard.js') incorporates progress bars and application status filters, enabling data-driven decisions. Almobark (2023) categorizes visualization techniques into descriptive, diagnostic, and predictive, which CareerConnect could adopt for future predictive analytics via AI.

Chapter 3

Project Methodology

3.1 Choice of Methodology

The Waterfall methodology was chosen for CareerConnect due to its structured approach, which suits a project with well-defined requirements. As noted in the draft, the methodologys linear phasesrequirements gathering, design, development, testing, and deploymentensure clear milestones and comprehensive documentation. This approach is ideal for CareerConnect, as the core features (job posting, chat, user management) were stable from the outset, minimizing the need for iterative changes. Waterfall also facilitates detailed planning, which was crucial for coordinating tasks like backend development (‘JobController.js‘) and frontend implementation (‘RecruiterDashboard.js‘).

3.2 Gantt Chart

[Placeholder for a major milestone Gantt chart, showing phases: Requirements Gathering (Weeks 12), Design (Weeks 35), Development (Weeks 612), Testing (Weeks 1315), Deployment (Week 16).]

Chapter 4

Different Technology and Tools

4.1 Justification

The technologies and tools were chosen for their compatibility, scalability, and community support, ensuring CareerConnects robustness and maintainability.

4.2 Programming Language and Framework

- **JavaScript**: Used for both frontend and backend due to its versatility and extensive ecosystem.
- **Node.js and Express.js**: Selected for the backend to handle asynchronous operations and build a scalable RESTful API, as seen in 'JobController.js'.
- **React.js**: Chosen for the frontend to create a dynamic, component-based interface, as implemented in 'RecruiterDashboard.js'.
- **Material-UI**: Used for styling to ensure a consistent, professional UI design.

4.3 Tools

- **IDE**: Visual Studio Code, chosen for its support for JavaScript development and debugging.
- **GIT Client**: Git, used for version control to manage code changes collaboratively.
- **Postman**: Utilized for API testing, ensuring endpoints like '/api/jobs/recruiter-jobs' function correctly.
- **XAMPP**: Used to manage the MySQL database locally during development.

4.4 Package Manager

NPM was used for dependency management, installing packages like 'express', 'mysql2', 'socket.io', and '@mui/material' to support development.

4.5 Real-Time Communication

Socket.io was chosen for its WebSocket implementation, enabling real-time chat with low latency, as planned for the messages page (Figure 13).

Chapter 5

Artefact Designs

5.1 User Management Subsystem

5.1.1 SRS

The system shall allow users to register, log in, and manage profiles with role-based access (candidate or recruiter).

5.1.2 Design Diagrams

[Placeholders for Activity Diagram, Wireframe (Figures 78), Use Case Diagram (Figure 3), ERD (Figure 2), Class Diagram (Figure 5), Sequence Diagram (Figure 4).]

5.1.3 Testing

Unit tests using Jest verified user authentication endpoints, ensuring secure login and registration.

5.2 Job Management Subsystem

5.2.1 SRS

The system shall enable recruiters to post, edit, and delete jobs, and candidates to search and apply for jobs.

5.2.2 Design Diagrams

[Placeholders for diagrams as above.]

5.2.3 Testing

Integration tests confirmed job CRUD operations, with the backend (‘JobController.js‘) successfully handling requests like ‘POST /api/jobs‘.

5.3 Communication Module Subsystem

5.3.1 SRS

The system shall provide real-time chat using Socket.io, supporting text messaging between recruiters and candidates.

5.3.2 Design Diagrams

[Placeholders for diagrams.]

5.3.3 Testing

Socket.io events were tested to ensure message delivery and online status updates.

5.4 AI Matching Subsystem (Future Scope)

5.4.1 Data Collection

Future AI implementation would collect data from user profiles (skills, experience) and job listings (requirements).

5.4.2 Model Development

A logistic regression model would be developed, trained on historical hiring data to predict candidate-job fit.

5.4.3 Optimization Evaluation

Hyperparameter tuning (e.g., learning rate, regularization) would optimize model performance.

5.4.4 Integration

The AI model would be integrated via a REST API, returning match scores to the dashboard.

5.4.5 Algorithm Performance

Cosine similarity and logistic regression performance would be compared using test data, evaluating metrics like precision and recall.

5.4.6 AI Testing

A/B testing would compare manual versus AI-based matching, with accuracy plotted for analysis.

5.4.7 Confusion Matrix and ROC Curve

[Placeholders for these metrics, showing true positives, false positives, etc.]

Chapter 6

Conclusion

6.1 Reflection on Aims and Objectives

CareerConnect achieved its aim of creating a seamless recruitment platform with real-time communication, meeting objectives like job posting and chat functionality. However, AI integration remains a future goal.

6.2 Findings

The project demonstrates the effectiveness of real-time chat in reducing hiring delays, with a scalable architecture supporting future enhancements.

Chapter 7

Critical Evaluation

7.1 Final Report Reflection

The report effectively documents the projects development, with clear sections on design and implementation. However, more detailed testing results could enhance its depth.

7.2 Findings and Process

The Waterfall methodology ensured a structured approach, though its rigidity limited adaptability to new requirements (e.g., AI integration).

7.3 System Evaluation

The system performs well for core features but lacks advanced functionalities like AI matching and mobile optimization.

7.4 Planning and Management

Planning was effective, with milestones met as per the Gantt chart, though time constraints delayed advanced features.

7.5 Self-Reflection

This project enhanced my skills in full-stack development, project management, and teamwork, preparing me for professional roles in software engineering.

Evidence of Project Management

7.6 Log Sheet

[Placeholder for signed log sheet scan.]

7.7 Gantt Chart

[Placeholder for detailed Gantt chart (Figure 15).]

References and Bibliography

Bibliography

[1] Smith, J., & Taylor, R. (2020). Improving Optical Character Recognition Accuracy for Scanned Documents. *Citeseerx*.

[2] Boldi, A. S. (2024). Designing for transparency: A web job board for e-recruitment to explore job seekers' privacy behaviours. *Behaviour and Information Technology*.

[3] Patel, R. (2022). Enhancing User Interaction in Recruitment Platforms via Real-Time Messaging.

[4] Miller, S. (2021). Data-Driven Decision-Making in Recruitment: An Overview of Analytical Tools.

[5] Almobark, B. (2023). A Quantitative Approach for Data Visualization in Human Resource Management. *IJCSNS*.

[6] Adobe Experience. (2024). Waterfall Methodology Overview.

[7] Scribd OCR. (2024). Project Report on OCR. Retrieved from `https://www.scribd.com/document/154124212/project-report-on-OCR`.

Appendices

7.8 User Manual

7.8.1 For Candidates

- **Registration**: Navigate to the signup page (Figure 7), enter your details, and select Candidate as the user type.
- **Job Search**: Use the available jobs page (Figure 11) to filter and apply for jobs.
- **Chat**: Access the messages page (Figure 13) to communicate with recruiters.

7.8.2 For Recruiters

- **Job Posting**: Use the dashboard (‘RecruiterDashboard.js‘) to post jobs with details and photos.
- **Application Management**: View and filter applications under the applications tab.
- **Communication**: Respond to candidates via the chat page (Figure 14).

7.9 System Configuration

- **Backend**: Node.js v16, Express.js, MySQL (XAMPP).
- **Frontend**: React.js v18, Material-UI, Socket.io-client.
- **Dependencies**: Install via ‘npm install express mysql2 socket.io @mui/material‘.