VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI - 590 018

PROJECT PHASE - II REPORT

on

AI driven Alumni Connect - A Comprehensive Alumni Management System

Submitted by

Adhwith A	**4SF22CS008**
Amrutha M	**4SF22CS024**
Megha P V	**4SF22CS108**
Thanush R	**4SF22CS231**

In partial fulfillment of the requirements for the VII semester

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE & ENGINEERING

Under the Guidance of

Dr. Mustafa Basthikodi

Designation, Department of CSE

at

SAHYADRI

College of Engineering & Management

An Autonomous Institution

MANGALURU

2025 - 2026

SAHYADRI

College of Engineering & Management

Adyar, Mangaluru - 575 007

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the phase - II work of project entitled **AI driven Alumni Connect - A Comprehensive Alumni Management System** has been carried out by **Adhwith A (4SF22CS 008), Amrutha M (4SF22CS024), Megha P V (4SF22CS108) and Thanush R (4SF22CS231)**, the bonafide students of Sahyadri College of Engineering and Management in partial fulfillment of the requirements for the VII semester of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi during the year 2025 - 2026. It is certified that all suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Dr. Mustafa Basthikod	**Dr. S S Injaganeri**
Professor & Head	Principal
Dept. of CSE	SCEM

Examiners' Name	**Signature**
1.	.....
2.	.....

SAHYADRI
College of Engineering & Management
Adyar, Mangaluru - 575 007

Department of Computer Science & Engineering

DECLARATION

We hereby declare that the entire work embodied in this Project Phase - II Report titled **AI driven Alumni Connect - A Comprehensive Alumni Management System** has been carried out by us at Sahyadri College of Engineering and Management, Mangaluru under the supervision of **Dr. Mustafa Basthikodi.,** in partial fulfillment of the requirements for the VII semester of **Bachelor of Engineering** in **Computer Science and Engineering**. This report has not been submitted to this or any other Universityfor the award of anyother degree.

Adhwith A (4SF22CS008)

Amrutha M (4SF22CS024)

Megha P V (4SF22CS108)

Thanush R (4SF22CS231)

Dept. of CSE, SCEM, Mangaluru

ABSTRACT

Many colleges gradually lose contact with their graduates once they complete their studies. This project introduces a single web-based space where alumni, students, and the institution can stay connected, share updates, and collaborate in a structured way. To solve this, our project introduces an online platform that helps students, alumni, and the institution remain connected in one place.Users are provided with individual profiles and can search for others with matching academic interests or professional paths, then interact through an integrated messaging interface. The platform additionally offers an events module for reunions, webinars, and campus programs, along with a job and internship board where alumni can publish opportunities for students. To make the system easier to use, we added a chatbot that answers common questions and helps users navigate the platform. The system is supported by a cloud backend that ensures safety, stability, and easy expansion. This project aims to strengthen the link between alumni, students, and the institution by creating a meaningful digital network that supports interaction and mentorship.

Keywords: Alumni Connect, networking, mentorship, cloud backend, community platform, career support.

ACKNOWLEDGEMENT

It is with great satisfaction and euphoria that we are submitting the Project Phase - II Report on **"AI driven Alumni Connect - A Comprehensive Alumni Management System"**. We have completed it as a part of the curriculum of Visvesvaraya Technological University, Belagavi in partial fulfillment of the requirements for the VII semester ofBachelor of Engineering in Computer Science and Engineering.

We are profoundly indebted to our guide, **Dr. Mustafa Basthikodi** , Professor & Head, Department of Computer Science and Engineering for innumerable acts of timely advice, encouragement and we sincerely express our gratitude.

We also thank **Dr. Suhas A Bhyratae** and **Ms. Prapulla G**, Project Coordinators, Department of Computer Science and Engineering for their constant encouragement and support extended throughout.

We express our sincere gratitude to **Dr. Mustafa Basthikodi**, Professor and Head, Department of Computer Science and Engineering for his invaluable support and guidance.

We sincerely thank **Dr. S. S. Injaganeri**, Principal, Sahyadri College of Engineering and Management, who have always been a great source of inspiration.

Finally, yet importantly, we express our heartfelt thanks to our family and friends for their wishes and encouragement throughout the work.

Adhwith A (4SF22CS008)

Amrutha M (4SF22CS024)

Megha P V (4SF22CS108)

Thanush R (4SF22CS231)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

While studying, students often engage closely with their college, but this connection often weakens after graduation. Alumni often gain valuable professional experience that could benefit current students, but many institutions lack a formal, long-term way to involve graduates. Our system provides a dedicated online platform to continuously connect alumni, students, and the institution. This section outlines the system's purpose, necessity, and scope, and provides an overview of the entire report.

1.1 Overview

Today's educational institutions often struggle to maintain meaningful engagement with students after they leave campus. Keeping alumni engaged in an organized way after graduation is especially difficult. Many colleges still use ad-hoc methods like social media groups or spreadsheets, which are not designed for organized networking or long-term record-keeping. Consequently, institutions end up with incomplete alumni records and lose valuable graduate information. To address these issues, we developed a platform using modern web technologies and a user-friendly interface. It allows students, alumni, and staff to connect, share opportunities, and collaborate in a more organized and secure environment.

We view the gradual loss of alumni engagement as more than a technical issue; it represents a significant strategic problem that can harm the institution's reputation and finances. The digital platform should rejuvenate the connection between alumni and the college by making communication and collaboration easy and worthwhile. The AI-Driven Alumni Connect system is designed to renew and maintain this connection through a contemporary, interactive platform.

1.2 Scope

The scope of the AI Driven Alumni Connect project covers the complete design, development, and testing of a secure and scalable web application. The platform can be accessed easily on both desktop and mobile browsers and is mainly designed for three key user groups—Alumni, Students, and Administrators. It focuses on several important functions that make the system both useful and reliable. One of the main goals is to include secure role-based login and profile management, along with an approval system for alumni verification handled by the administrator to maintain trust in the network. The platform also supports a structured mentorship program, where users can communicate safely through one-to-one chat channels. Another major part of the project is the Job and Internship Portal, where verified alumni can share openings and opportunities, which are reviewed and monitored by administrators. In addition, the system includes an event management module that helps organize and promote activities such as reunions, webinars, and workshops with minimal effort. The project also integrates artificial intelligence using the Google Gemini API to provide intelligent assistance and automate data-related tasks. The entire application is built using Firebase as a Backend-as-a-Service (BaaS) platform, ensuring automatic scalability, high security, and reduced maintenance requirements. This approach delivers a dependable and ready-to-deploy solution that can serve a large number of users across institutions efficiently.

1.3 Motivation

Our motivation for building this platform arose from observing that many colleges still use generic or outdated tools to reach alumni. Because communication is spread across multiple uncoordinated channels, alumni engagement remains low and maintaining long-term relationships becomes difficult for the institution. Such gaps do not just affect communication; they also harm the institution's reputation, limit student opportunities, and create challenges in meeting accreditation requirements. Our team wanted to design a platform that could change this situation by encouraging more meaningful and long-term engagement between all members of the academic community. One of the biggest barriers we noticed was the amount of manual work involved in maintaining alumni data and managing connections. To address this, we integrated Artificial Intelligence through the Google Gemini API, which helps the system handle smart features such as automatic resume reading and personalized recommendations for users. These intelligent tools reduce manual effort and make interaction smoother and more efficient.

Overall, this project reflects our goal to strengthen the connection between the college and its alumni while turning what was once an administrative task into a system that actively supports growth, collaboration, and institutional development.

1.4 Purpose

The main purpose of the AI Driven Alumni Connect platform is guided by two closely connected goals that influence every part of its design and development. The first goal is to maximize the mutual value shared within the network. This means building a trustworthy digital environment where the exchange of knowledge, experience, and opportunities happens naturally among users. Alumni are encouraged to contribute through mentorship and job postings because they also gain professional growth, connections, and recognition in return, creating a cycle of continuous engagement that keeps the platform active and valuable. The second goal is to automate and improve the efficiency of institutional operations. This involves replacing older, manual, and error-prone methods of tracking graduates with a cloud-based system that updates itself automatically. With this approach, administrators can maintain an accurate, easily accessible alumni database with very little manual effort, helping the institution make quick, data-driven decisions and meet accreditation standards. Overall, the purpose of the project is to transform alumni relations from a routine administrative task into an intelligent, dynamic, and self-sustaining community platform that continually benefits students, alumni, and the institution together.

1.5 Definitions, Acronyms, and Abbreviations

To maintain complete clarity, precision, and consistency throughout the technical and analytical sections of this Phase II report, the following terms and abbreviations are explained below.

- **AMS**: Refers to the Alumni Management System, a general term used for platforms that handle the collection and management of alumni data.
- **AI**: Stands for Artificial Intelligence and, in this project, specifically represents the smart services powered by the Google Gemini API, which forms the key innovation of the system.
- **BaaS**: Represents Backend-as-a-Service, a cloud-based architecture model that uses Google Firebase to provide managed backend functionality.

- **Firebase**: The chosen backend platform that provides secure services for authentication, data storage through Cloud Firestore (a NoSQL database), and file storage capabilities.
- **Gemini API**: Refers to the external AI interface integrated into the project to support chatbot communication and enable multimodal input processing.
- **FRs**: Denotes Functional Requirements, which define the actions and operations the system is expected to perform. **NFRs** are Non-Functional Requirements that describe essential quality factors such as performance, security, and scalability.
- **SPA**: Stands for Single Page Application, a front-end architectural style used in this project and implemented using the React.js framework.

1.6 Structure of the Report

This Phase II project report has been systematically structured to give a clear and detailed explanation of the complete development process, maintaining full traceability from the initial concept to the final validation stage. Each chapter focuses on a specific part of the project and is organized as follows:

- **Chapter 2: Literature Survey** reviews existing research on alumni management systems, pointing out key technical and strategic gaps that our system addresses.
- **Chapter 3: Problem Statement** articulates the problem statement, outlines project objectives, explains why the project matters, and specifies the scope and deliverables.
- **Chapter 4: Software Requirements Specification** provides the detailed functional and non-functional requirements, including technical specs, performance expectations, and quality criteria for the system.
- **Chapter 5: System Design** shows the system's architecture, including UML diagrams, data flow diagrams, and the main database schema, to depict the platform's design.
- **Chapter 6: Results and Discussion** presents the implementation results and evaluates how the system fulfills the objectives, comparing its performance with the expected results.
- **Chapter 7: Conclusion** summarizes the project's achievements, emphasizes its key technical contributions, and suggests directions for future improvements, scalability, and further research.

CHAPTER 2

LITERATURE SURVEY

Chapter 2 reviews the history of alumni management systems, from basic data-storage tools to contemporary community-focused platforms, to explain our design choices. This review provides context by comparing various technical approaches and assessing the strengths and weaknesses of earlier solutions. These comparisons clarify why we selected certain design decisions, architectural patterns, and technologies for our AI-Driven Alumni Connect platform. The survey considers thirty studies covering almost two decades, grouped into three focus areas for clarity: *Foundational Systems and Data Management*, *Relationship Management and Engagement Models*, and *Advanced Intelligence and Modern Compliance*.

Overall, the literature indicates that having a specialized alumni management system is widely seen as crucial in higher education. These systems do more than store contacts; they facilitate long-term relationship-building, institutional data management, alumni tracking, and career support for both students and graduates. The review starts with early database-focused approaches and progresses to recent research that incorporates features like AI and blockchain. Together, these references form the foundation for our system's design and validate the need for the features we have implemented.

2.1 Limitations of Prior Work

The reviewed studies consistently highlight the benefit of shifting from informal alumni groups to dedicated management platforms. In general, existing work can be grouped into three categories: foundational systems, relationship management tools, and intelligent/compliance-focused designs.

2.1.1 Foundational Systems and Data Management

Early contributions primarily addressed the practical need to centralize alumni information and to track changes over time. This phase laid down the basic patterns for a dependable AMS and helped institutions leave behind fragmented, manual records.

Studies by Etcuban and Durano [37] and by Yuan et al. [35] typify this database-first approach to building alumni systems. They relied on traditional DBMS designs to store and manage profiles in a single place. While this stabilized record-keeping, many of these solutions offered limited interaction and tended to behave like searchable directories rather than active communities. A case from a nuclear institution by Moreira et al. [47] confirms that establishing a solid database is a necessary first step before richer engagement features can be added.

As infrastructure improved, institutions began using the collected data for governance and accreditation. Wahjusaputri et al. [5] described a tracer-study website for vocational high schools that streamlined employment tracking and analysis. Jaiswal et al. [21] outlined a typical web-based tracking model for engineering colleges. Nishanth et al. [22] introduced a context-aware approach that blended early cloud use with situational cues to make information more relevant and timely. Collectively, these efforts show the shift from standalone databases to connected, web-accessible systems, and they highlight baseline expectations around integrity, availability, and initial cloud adoption.

More recent foundational designs by Sawai et al. [1] and Rista et al. [10] focus on end-to-end platforms and confirm the continued emphasis on scalability and dependable tracking for large user bases. Mukherjee et al. [26] proposed a centralized AMS prototype leveraging cloud resources to simplify long-term maintenance. A common limitation across these systems is the lack of strong personalization, smooth onboarding, or proactive assistance—precisely the areas our work targets (FR13, FR14). In parallel, studies such as Kumar et al. [13] brought security topics to the front, anticipating today's stronger privacy and protection needs (NFR6, NFR7).

2.1.2 Relationship Management and Engagement Models

A second set of studies argues that storing data alone is not enough; the system must also encourage sustained participation and community value. Here, the focus shifts to relationship-building features and workflows that reward ongoing involvement.

Several works apply Customer Relationship Management (CRM) ideas to alumni contexts.

Ahmadi et al. [41] and Mahmud [42] discuss models for tracking engagement activities—such as events and potential contributions—rather than just listing graduates. Pradana [17] connects effective alumni relationship practices to positive organizational behaviors, emphasizing how a healthy network benefits the broader academic community.

System designs that enable verified, peer-to-peer contact appear in Rajini and Upendrasingh [12], while Lacasandile et al. [11] present a user-centered portal that makes participation easier through accessible features. Straujuma [31] frames alumni platforms through a knowledge-management lens, suggesting that systems should help capture, organize, and return expertise to the institution. Pohthong and Trakooldit [38] support this by integrating e-learning with alumni tools, showing clear benefits when learning resources and alumni interactions are brought together.

Mentoring features receive special attention in Chi et al. [40], which explores a smart alumni mentoring approach and provides a reference point for our mentoring and matching functions (FR5, FR6). On the institutional side, Isaac et al. [43] and Altuntaş and Baykal [34] use alumni feedback and performance analysis to improve program quality, reinforcing the role of analytics dashboards for administrators. Silva and McFadden [48] connect alumni readiness with curriculum planning, arguing that a well-built AMS can inform academic decisions. Overall, this body of work shows that while manual networks can work at small scales, automation becomes essential when the community grows.

2.1.3 Advanced Intelligence and Modern Compliance

The most recent and forward-looking cluster of research, spanning the last few years, points directly toward the inevitable evolutionary stage of AMS. This phase is defined by the necessary integration of emerging, powerful technologies like AI, Blockchain, and rigorous data compliance frameworks. These papers fundamentally justify the core innovative elements and advanced technological stack of the AI driven Alumni Connect platform.

The strategic push for greater intelligence and systemic automation is clearly evident, building upon earlier concepts. Khan et al. [23] proposed an intelligent alumni system relying on advanced AI techniques to drive smarter, contextual engagement. Yumen [50] detailed leveraging regression models for data analysis, illustrating how sophisticated machine learning can be used to generate predictive alumni insights regarding future career trends or optimal event participation rates. Most importantly, Chen and Zhang [7] explicitly mandated advanced AI-driven

engagement, emphasizing the use of machine learning to create truly personalized alumni interactions and dynamic content recommendations based on deeply analyzed user behavior and profile history. This body of work provides the direct and rigorous technological mandate for our project's innovative use of the Gemini API for smart suggestions and personalized user assistance (FR14).

Parallel to intelligence, the imperative for unassailable trust, security, and verification has escalated dramatically. Building on earlier security concerns, Alharbi and Alharbi [6] proposed a novel Blockchain-Based Verification model to enhance trust and transparency in alumni networks, directly challenging the inefficiency and fraud potential of manual administrative verification (FR4) with decentralized, immutable ledger technology. This highlights the critical necessity for a highly secure, trustworthy, and efficient verification mechanism in any contemporary AMS.

Finally, modern operational and critical legal compliance demands are addressed by several key papers. Okafor and Ibrahim [15] focused on the technical requirements for Data Privacy Compliance aligned with the strict GDPR/CCPA framework. This work is absolutely crucial for ensuring the platform's ability to scale globally without compromising fundamental user privacy (NFR7), mandating strict, server-enforced security rules (NFR6). Furthermore, strategies for long-term user experience and retention are addressed by Garcia and Martinez [14], who advocated for Mobile-First Portals, and Patel and Lee [18], who proposed Gamification Systems. These studies collectively confirm that robust technical capability must be universally paired with seamless, highly engaging, and universally accessible interfaces (NFR1, NFR2) to ensure long-term, sustained user participation and measurable return on investment.

2.2 Research Gaps Identified

After reviewing 30 studies, we found that although alumni management tools are well developed, there are still key areas for improvement, particularly in automation and scalability in the areas of intelligent integration and automated scalability. The surveyed literature demonstrates that prior systems generally fall into two distinct, strategic but ultimately limited categories: those focused on high-security, archival data (often resulting in low-engagement), or those focused on high-engagement social networking (often resulting in low institutional security and weak administrative structure).

Our project is designed to deliberately eliminate this trade-off by addressing the three most

critical existing gaps:

1. Lack of Comprehensive AI Integration: Existing intelligent systems (e.g., [23], [40]) typically utilize basic machine learning for a singular, isolated function (e.g., only tracking [22] or simple mentoring). There is a conspicuous absence of prior work that centrally integrates multi-modal AI to simultaneously drive both user experience and back-end efficiency.[44]

2. Single-System Centralization and Reciprocity Failure: Many systems prioritize one stakeholder group above others (e.g., exclusively alumni networking [12] or purely tracer studies for administrators [5]). The core functional flaw is the absence of a single, highly integrated ecosystem that facilitates shared, reciprocal value among all three major groups.

3. Technology Stack and Scalability Alignment: Few studies detail the rigorous selection and utilization of modern, highly scalable Backend-as-a-Service (BaaS) platforms like Firebase for both real-time data synchronization (necessary for chat and dynamic content) and massive, automated scalability (NFR8). Reliance on traditional servers or self-hosted cloud instances, as often implied, introduces significant cost, maintenance, and scaling overhead that hinders adoption.

2.3 Contribution of the Present Work

The AI-Driven Alumni Connect platform is designed as a comprehensive solution that directly targets the key limitations observed in earlier systems, rather than offering only a minor extension of existing work.. The contribution of this project is threefold, directly targeting the identified failures in scalability, intelligence, and stakeholder value.

1. A Modern, Scalable, and Low-Maintenance Architecture

A primary contribution is the deliberate architectural choice to build upon a serverless, Backend-as-a-Service (BaaS) model, specifically Google Firebase. The literature review showed that many prior systems, while functional, were built on traditional, self-hosted, or monolithic architectures. This approach carries a significant, long-term technical debt in the form of server maintenance, manual scaling, database administration, and security patching. Our project's mandatory choice of the Firebase architecture directly addresses this fundamental need for a low-maintenance, high-availability, and extremely cost-effective solution. Adopting a serverless backend allows many non-functional requirements (such as scalability and availability) to

be satisfied by default, enabling the institution to run the platform at large scale without maintaining its own dedicated infrastructure team. This makes the solution not only powerful but also economically viable for academic institutions of all sizes.

2. A Holistic and Multi-Faceted AI Integration

The second major contribution is the move from single-purpose AI to a comprehensive, multi-faceted AI integration that enhances the entire user lifecycle. The research gap analysis revealed that intelligent systems were often limited to a single function, such as AI-driven matching [23] or simple mentoring suggestions [40]. Our platform fills this gap by implementing the Google Gemini API for two distinct, high-impact purposes:

- Enhancing User Experience: The 24/7 personalized, conversational chatbot provides immediate user support and personalized guidance, making the platform more accessible and intuitive.
- Solving Data Onboarding and Integrity: The intelligent resume parser is a critical innovation that solves the user friction problem at registration. By allowing alumni to auto-populate their profile from a PDF or image, we drastically increase the rate of profile completion and ensure the data is structured and accurate. This high-quality data, in turn, fuels the effectiveness of all other platform features, including the AI-driven matching algorithms.

This combined, centralized AI approach—using generative AI to solve both user-facing and data-integrity problems—is a novel contribution within the reviewed domain.

3. A Unified Ecosystem with a Reciprocal Value Proposition

Finally, this project directly solves the Reciprocity Failure gap, where prior systems often served one stakeholder at the expense of others (e.g., admin-focused tracer studies [5] or alumni-only networking [12]). Our proposed solution creates a seamless, unified hub that effectively links all three vital actors—Alumni, Students, and Administrators. By integrating shared features like the Job Portal, Mentorship, and Event Management, the platform creates a virtuous cycle of value:

- Alumni gain value from finding talent and networking, motivating them to contribute (post jobs, offer mentorship).
- Students gain value from these jobs and mentors, motivating them to engage with the platform.

- Admins gain value from this high engagement, which provides them with the accurate, real-time data needed for accreditation and outreach, motivating them to govern and support the community.

This ensures all three groups derive continuous, measurable value, which reinforces the network effect and guarantees the platform's long-term sustainability.

Our proposed system brings together these findings to create a practical and improved AMS model—robust data management [37], CRM principles [41], and specialized AI features [23]—while directly and simultaneously addressing the critical identified gaps in comprehensive intelligence, architectural scalability, and unified stakeholder value.

Table 2.1: Alumni Management Systems Research (Part 1 of 4)

References	**Methodology**	**Dataset**	**Technology**	**Outcomes**
Sawai et al. [1]	Comprehensive system design	University alumni data	Web-based system	Improved alumni engagement and tracking
Stallings [2]	Qualitative evaluation	Alumni coaching participants	Career coaching program	Improved alumni-student career perceptions
Shaw [3]	Staff perspective analysis	Conservatoire staff feedback	Music education framework	Clarified benefits of alumni in higher music ed
Chileshe & Mutono-Mwanza [4]	Communication strategy study	Public secondary schools	External communication channels	Enhanced alumni participation in schools
Wahjusaputri et al. [5]	Website-based tracer study	Vocational high school alumni data	Web-based application	Enhanced alumni data management
Alharbi & Alharbi [6]	Blockchain-Based Verification	Hypothetical model	Blockchain	Enhanced trust and transparency in verification
Chen & Zhang [7]	AI-Driven Engagement	Alumni interaction logs	Machine Learning	Personalized alumni interactions and recommendations

Table 2.2: Alumni Management Systems Research (Part 2 of 4)

References	**Methodology**	**Dataset**	**Technology**	**Outcomes**
Hong [8]	Strategic analysis	International alumni data	Soft power framework	Knowledge diplomacy through alumni engagement
Sinor [9]	Value assessment	Startup hub alumni	Community network	Defined value of active alumni associations
Rista et al. [10]	Design & implementation	University alumni database	Web-based system	Effective alumni connection and management
Lacasandile et al. [11]	User-centric portal design	National University alumni data	Web-based system	Fostered alumni involvement
Rajini & Upendrasingh [12]	Alumni networking system	Educational institution dataset	Web platform	Improved alumni networking
Kumar et al. [13]	Secured web-based system	University alumni data	Web security techniques	Enhanced security & data integrity
Garcia & Martinez [14]	Mobile-First Portals	User engagement metrics	Responsive Design	Enhanced accessibility and mobile engagement
Okafor & Ibrahim [15]	Data Privacy Compliance	Data security protocols	GDPR/CCPA Framework	Compliance assurance and user privacy protection
Southworth et al. [16]	Curriculum modeling	Higher ed curriculum data	AI literacy framework	AI integration across university curriculum
Pradana [17]	Alumni relationship management	College alumni dataset	CRM-based approach	Increased organizational citizenship behavior
Patel & Lee [18]	Gamification in Networks	Network activity data	Gamification Systems	Increased user participation and sustained engagement
Larsson et al. [19]	Curriculum integration study	Business education students	Alumni engagement project	Fostered student-alumni curricular engagement
Begum et al. [20]	Sentiment analysis model	Student-alumni interactions	AI & Machine Learning	Efficient interaction via sentiment analysis
Jaiswal et al. [21]	Alumni tracking system	Engineering college alumni	Web-based	Better alumni tracking

Table 2.3: Alumni Management Systems Research (Part 3 of 4)

References	**Methodology**	**Dataset**	**Technology**	**Outcomes**
Nishanth et al. [22]	CA-based management system	University alumni data	Web & cloud computing	Improved alumni-student connection
Khan et al. [23]	Intelligent alumni system	University database	AI-based system	Smart alumni engagement
Memon et al. [24]	Reflective analysis	Teacher education alumni	Islamic pedagogy framework	Alumni enactments of specific pedagogy
Hehir et al. [25]	Community-engaged research	Polar expedition alumni	Post-trip program evaluation	Impact on pro-environmental behavior
Mukherjee et al. [26]	Centralized AMS prototype	Prototype model	Cloud computing	Centralized alumni tracking
Dollinger et al. [27]	Mentoring program evaluation	University mentoring data	Mentorship framework	Validated mutual benefits of mentoring
Khanna et al. [28]	Brand resonance analysis	Business school alumni	Branding framework	Factors influencing alumni brand promotion
Skrzypek et al. [29]	Program implementation study	Social work students & alumni	Mentoring connection model	Successful student-alumni mentor implementation
Warren [30]	Network implementation study	Alumni association members	Digital member network	Coproduction in digital alumni networks
Straujuma [31]	Knowledge management	Higher education dataset	Knowledge management system	Improved alumni engagement
Gallo [32]	Strategic plan analysis	University strategic documents	Policy analysis	Role of graduates in institutional strategy
Unangst [33]	Landscape analysis	International alumni affairs	Trans-national service model	Emerging trends in international alumni affairs
Altuntaş & Baykal [34]	Alumni performance analysis	Nursing education alumni	Statistical analysis	Enhanced nursing education evaluation
Yuan et al. [35]	System design and implementation	University alumni dataset	Web-based system	Efficient alumni data management

Table 2.4: Alumni Management Systems Research (Part 4 of 4)

References	**Methodology**	**Dataset**	**Technology**	**Outcomes**
Iskhakova et al. [36]	Empirical validation	German graduates	Alumni loyalty	Integrative model of alumni loyalty
Etcuban & Durano [37]	Alumni database development	University alumni records	Database management system	Centralized alumni information
Pohthong & Trakooldit [38]	E-learning and alumni integration	University alumni & students	E-learning & KM	Enhanced knowledge sharing
Vieregge et al. [39]	Expectation analysis	Hospitality school alumni	Satisfaction survey	Alumni expectations and satisfaction levels
Chi et al. [40]	Smart alumni mentoring system	University mentoring data	Smart systems	Improved mentoring outcomes
Ahmadi et al. [41]	CRM for alumni liaison	University alumni data	CRM system	Strengthened alumni relations
Mahmud [42]	CRM modeling for university alumni	University CRM data	CRM framework	Structured alumni management
Isaac et al. [43]	Alumni perception assessment	Health services management alumni	Survey-based study	Alumni feedback on curriculum
Iskhakova et al. [44]	AMS support software review	University alumni data	Software-based support	Enhanced alumni management
Gallo [45]	Institutional advancement study	Irish university alumni	Relationship building model	Interactive alumni relationship building
Weerts & Ronca [46]	Predictive modeling	Higher ed giving data	Classification trees	Prediction of alumni charitable giving
Moreira et al. [47]	Alumni database system	Nuclear institution alumni records	Database management	Established alumni database
Silva & McFadden [48]	Alumni workforce prep assessment	Operations & IS alumni data	Educational assessment models	Improved curriculum design
Wang & Amponstira [49]	Alumni management strategies	Private university dataset	Alumni management strategies	Effective alumni engagement
Yumen [50]	Regression models for analysis	University alumni data	Data analytics & ML	Predictive alumni insights

CHAPTER 3

PROBLEM FORMULATION

3.1 Problem Description

At present, many institutions depend on a mix of disconnected and traditional approaches to maintain alumni contact. Typical strategies include infrequent reunions, irregular bulk emails, and informal social media communities, all of which demand significant manual effort yet yield limited, short-lived interaction.Although these methods offer occasional interaction, they fail to support deep, long-term, and mutually beneficial professional relationships between alumni and the institution.

Many colleges rely on social media platforms such as LinkedIn or Facebook to reach alumni, but these are not well suited to academic communities that rely on trust and shared identity. LinkedIn, for example, is a vast, ”cold” network built on transactional connections. A mentorship request from a current student to a high-level alumnus on LinkedIn is often lost in a sea of other solicitations; it lacks the immediate, shared identity and inherent trust that a fellow alumnus status should provide. Facebook, on the other hand, is a social graph, not a professional one. Many alumni are understandably hesitant to mix their private, personal lives with professional mentorship or to field career questions in the same space they share family photos. Furthermore, these public platforms suffer from a low signal-to-noise ratio, where valuable opportunities and discussions are easily buried under irrelevant content.

Recognizing these limitations, some institutions have attempted to build their own dedicated solutions, such as internal alumni directories or basic web portals. However, these efforts also fall short, as they often replicate the ”digital phonebook” anti-pattern. These solutions are typically static, read-only, and lack any interactive features. They provide no intrinsic value to the alumnus, who is expected to provide their personal data but receives no tangible benefit in return. This lack of a reciprocal value proposition is the primary driver of data atrophy; alumni

have no incentive to return to the platform to update their profiles, and the database becomes obsolete within a few years of its creation.

Furthermore, these dedicated social media groups and static directories almost universally fail to provide the most critical features that a professional academic network requires. They lack advanced, context-aware search and filtering, making it impossible for a student to find a mentor based on a specific combination of criteria (e.g., ”alumni from the Electronics branch, graduated after 2015, who now work in VLSI design”). They lack integrated and secure one-on-one messaging systems, forcing users back onto insecure public platforms. They are missing robust, built-in event management tools, making the promotion and organization of reunions or webinars a manual, high-friction process. Most critically, they lack a formal, structured job portal and an integrated mentorship program, which are the very pillars of a value-driven community.

This problem of platform failure is compounded by the core operational challenge of data maintenance. For most institutions, the alumni database is a non-comprehensive, decaying asset. As alumni naturally move to different cities, change jobs, or switch countries, keeping track of their contact details and professional achievements becomes a near-impossible manual task. This data decay is not a minor inconvenience; it represents a critical institutional failure. It makes it incredibly difficult for the institution to reach out to its alumni for valuable initiatives, such as targeted fundraising campaigns, invitations for guest lectures, industry-academia collaborations, or gathering the feedback required for accreditation bodies.

This creates a significant opportunity cost for all three stakeholders. The institution fails to leverage its most powerful asset. The alumni miss out on a trusted, exclusive network for professional and social reconnection. Most importantly, the students are cut off from a vital source of guidance and opportunity, left to navigate the transition from academia to industry on their own.

In summary, the scenario highlights a significant and persistent gap in how educational institutions manage and engage their alumni networks. The problem is not just a lack of a single tool, but a systemic breakdown in the flow of value between the institution, its graduates, and its current students. This project aims to directly overcome these challenges with a focused, AI-assisted approach. It aims to fill this gap by providing a single, centralized, and user-friendly platform that is built on a foundation of reciprocal value. It will facilitate meaningful interactions, provide secure and scalable tools, and leverage AI through an AI-powered chatbot and AI-driven matching, to create a truly intelligent and self-sustaining community.

3.2 Problem Statement

Schools and universities struggle to maintain long-lasting meaningful connections with their alumni. Although social media and other communication channels exist, there is no single, dedicated platform tailored to alumni communities. Without a central platform, communication declines and alumni find it hard to remain connected with their alma mater and each other. Consequently, students lose mentorship and job opportunities, and institutions fail to fully leverage their alumni networks. A unified platform is needed to bring alumni together, help students, and reinforce alumni-institution ties. It should even include an AI chatbot for around-the-clock help with common inquiries.

3.3 Objectives

- To develop a centralized platform for alumni to connect and engage.
- To develop a platform having networking, mentorship, and career opportunities for students.
- To develop a platform to help institutions maintain an updated alumni database and organize events.
- To provide a secure, scalable, very easy, user-friendly platform, providing an AI chat-bot integration.

3.4 Functional Requirements (FRs)

The functional requirements outline the essential actions and behaviors the system must perform to provide its intended value to users. These represent the specific, measurable tasks the platform must accomplish, ensuring the direct fulfillment of the digital solution defined in the problem statement.

User Registration and Login

A fundamental requirement for any secure professional network is strong identity management and security. The system must ensure that only verified or valid users associated with the institution can gain access to the network, and that their personal identity is protected throughout their session.

- FR1: Users (alumni or students) must be able to create accounts using their unique university identification (e.g., student ID/register number) and a valid email address. The system must confirm the users role (Student or Alumni) during the initial registration to correctly allocate access privileges.

- FR2: The login process must be secure, utilizing industry-standard authentication protocols (e.g., secure hashing and tokenization), and must provide mechanisms for safe session management (e.g., strong password requirements, secure password reset functionality).

Use Case Detail: User Registration (FR1) - The Onboarding Journey

The user journey begins at the unauthenticated entry point. The Pre-Condition is that the individual is currently unauthenticated and has navigated to the designated registration page, ready to establish their digital identity within the network. The user initiates the process—the Trigger—by supplying their institutional identification number, a unique email address, a password that meets complexity requirements, and explicitly selecting their relationship role (Student or Alumni). The Execution phase involves multiple critical steps: the system first validates the format of the inputs (checking email structure and password strength). Upon format verification, Firebase Authentication is called to create the user record securely. Simultaneously, a corresponding profile document containing the user ID and basic role is instantiated in Cloud Firestore. If this entire chain is successful, the Post-Condition is met: the user is fully authenticated, receives an immediate success notification, is redirected to their respective personalized dashboard, and is then immediately prompted to complete their detailed profile, signaling their successful entry into the network.

Profile Management

The network's value depends on accurate, detailed data about its users. The system must provide simple, dynamic tools for users to accurately represent their academic and professional identities, while the networks overall trustworthiness is maintained through institutional oversight.

- FR3: Users should be able to update their profiles with comprehensive details, including educational background, detailed work history, current job titles, specialized technical skills, and professional interests. This profile data must be stored in a highly structured, queryable format to facilitate efficient searching and AI-based matching.

- FR4: Alumni profiles must initially enter a pending verification status upon creation. These profiles should then be explicitly reviewed and verified by institutional administrators against official records to maintain the networks required level of authenticity, exclusivity, and professional trust.

Use Case Detail: Alumni Verification (FR4) - Gatekeeping Trust

Profile authenticity is paramount for maintaining the quality of mentorship and job opportunities, making this administrative process non-negotiable. The alumni profile exists in a state marked `pending`, which is the Pre-Condition. The crucial Trigger occurs when an administrative staff member logs into the secure Admin Portal and accesses the dedicated Verification Queue module, which lists all pending profiles awaiting review. During the Execution phase, the administrator reviews the submitted profile details (e.g., graduation year, degree, and ID) and cross-references them against institutional legacy records. Based on this review, the Admin executes the decision by clicking either Approve or Reject. Once the decision is executed, the Post-Condition is reached: the Alumni profile status flag in the database is updated to either `verified` (granting the user full access to features like job posting and mentorship) or `rejected` (with an automated notification and reason sent back to the user).

Mentorship and Networking

The platforms core mission of bridging the student-alumni gap is realized through these interactive features, intentionally transforming passive connections into structured, actionable professional relationships.

- FR5: Verified Alumni must be able to explicitly volunteer to serve as mentors, clearly specifying their areas of expertise and detailing their capacity, such as their maximum monthly availability for mentorship sessions.

- FR6: Students must be able to securely browse and filter verified Alumni profiles and initiate formal mentorship requests based on shared interests or academic background. The system must provide a clear, trackable mechanism for the mentor to accept, decline, or defer the request, maintaining a history of all interactions.

- FR7: The platform should utilize basic attribute matching algorithms (e.g., matching a students desired branch/skill request against an alumnuss current job role/expertise fields) to intelligently suggest relevant connections and potential mentors, always prioritizing high-relevance matches over sheer volume of profiles.

Job and Internship Portal

The job portal serves as a high-value, exclusive pipeline for career opportunities, leveraging the inherent trust of the alumni network to place current students into relevant roles efficiently.

- FR8: Only Verified Alumni users can create and post detailed job or internship opportunities, including required skills, estimated salary range, and application deadlines, thereby ensuring a consistent quality and authenticity of all postings.
- FR9: Students should be able to view, search, and filter job listings, and submit their applications (including secure uploading of resumes) directly through the platform. The system must securely store the students complete application history and the current status of each application for easy tracking.
- FR10: Institutional administrators must retain the authority to review, moderate, approve, or remove job postings to ensure they meet institutional ethical guidelines and maintain essential quality control standards for the network.

Event Management

To sustain community and continued engagement, the platform requires robust, intuitive tools for managing and promoting communal gatherings.

- FR11: Admins and designated verified alumni group organizers can create, schedule, and promote various events (reunions, career webinars, guest lectures). The system must support the use of rich text and image content to create compelling event details and collateral.
- FR12: Users should be able to easily view comprehensive event details, register for attendance with minimal friction, and receive automated, timely email and in-platform notifications regarding event status, changes, or upcoming deadlines.

AI Chatbot Support

The inclusion of artificial intelligence provides the next-generation layer of support and personalized interaction, fundamentally moving the system beyond a passive directory.

- FR13: A built-in AI chatbot, powered by the sophisticated Gemini API, must be available and accessible across the platform 24/7. This chatbot is responsible for addressing

common user queries, providing intelligent navigation assistance, and offering general platform guidance.

- FR14: The chatbot must leverage the currently logged-in users profile data to provide personalized, context-aware suggestions for relevant job opportunities, potential mentors, and upcoming events that specifically match the users recorded skills and professional interests.

Detailed Use Case Narrative: AI Resume Parsing (Implicit FR) - Minimizing User Friction

This feature is absolutely critical for maximizing data quality from the outset and minimizing user friction during the necessary profile setup. The process begins when the Alumni user is completing their profile and possesses a resume file in a common format (PDF, JPG, or PNG)—the Pre-Condition. The Trigger is the user actively initiating the file upload within the profile form. In the complex Execution phase, the resume file is first securely transmitted and stored in Firebase Storage. Concurrently, the system makes a specific, customized call to the Gemini API, sending the files data stream along with a carefully engineered prompt that explicitly requests structured data extraction for all key profile fields (e.g., name, skills, job title, work history). The API processes this multimodal input and returns the requested data in a standardized JSON object. For the Post-Condition, the system receives the JSON and uses this extracted information to automatically populate the profile form fields, leaving only final review and confirmation to the user, thereby drastically reducing onboarding time.

3.5 Non-Functional Requirements (NFRs)

Non-Functional Requirements define the crucial quality attributes that govern the systems operation and user experience. They dictate the how of the systems behavior and are essential for guaranteeing its performance, security, and long-term viability.

Usability (Aligned with ISO 9241-11)

Usability defines the ease and efficiency with which users can interact with the platform. The platform must feel intuitive and effortless.

- NFR1: The interface must be intuitively designed, visually clean, and modern in its presentation, requiring minimal preparatory training for all first-time users. The perfor-

mance metric target is that 90% of core user tasks (e.g., registering, searching, applying for a job) must be completable within 3 clicks or less from the main dashboard.

- NFR2: The design will be fully responsive, ensuring seamless functionality and consistent visual fidelity across all common device screen sizes, ranging from small mobile devices (320px minimum width) up to large desktop monitors (1920px).

Performance (Quantitative Metrics)

Performance metrics ensure the system can reliably handle the expected user load and provide a responsive experience, which is paramount for high user retention and professional appeal.

- NFR3: The platform must be robust enough to support concurrent usage by at least 5,000 active users simultaneously, without any noticeable degradation in response speed or perceived service quality.
- NFR4: Core user actions, including initial dashboard loading, submitting registration forms, and executing a job application, must complete their entire cycle (server processing and client rendering) within a maximum of 1.5 seconds under normal operating load conditions.
- NFR4.1 (AI Latency): Due to the necessary reliance on external API calls, the response from the AI Chatbot must be returned and displayed to the user within a maximum latency of 3.0 seconds of the query being submitted, minimizing perceived waiting time.

Security (Data Integrity and Access Control)

Security is non-negotiable and requires a multi-layered defense to protect sensitive professional and personal data.

- NFR5: All data transmission, both client-to-Firebase services and client-to-Gemini API endpoints, must be rigorously protected using secure, modern protocols, specifically HTTPS/TLS 1.2 or higher.
- NFR6: User credentials and all sensitive authentication information must be managed exclusively by Firebase Authentications secure hashing service. Furthermore, Firestore security rules must strictly implement a Role-Based Access Control (RBAC) model to govern all data read/write access on a per-user and per-role basis (e.g., a Student cannot read an Alumnis private contact details or a job posters unpublished drafts).

- NFR7: The system must fully respect user privacy, providing explicit and easily manageable controls for users to determine the visibility of their contact details and specific professional profile information to other network members.

Scalability and Capacity

- NFR8: The chosen architecture, leveraging the Google Firebase platform, must be inherently capable of automatically scaling its resources to accommodate up to 1 million registered users while maintaining guaranteed performance targets without requiring manual intervention or server provisioning.
- NFR9: Cloud Firestore and Firebase Storage are required to handle expected data growth, including the capacity for millions of chat messages, thousands of concurrent job postings, and tens of thousands of media file uploads (resumes, profile pictures) over the systems lifecycle.

Availability and Maintainability

- NFR10: By leveraging Googles global, distributed infrastructure, the platform targets a high operational availability of 99.9% uptime annually, excluding only necessary prescheduled maintenance windows.
- NFR11: The React codebase must be strictly structured into modular, self-contained components with clear naming conventions, facilitating straightforward management, debugging, and the easy extension of future features (e.g., integrating a new Donor Portal module) by future development teams.
- NFR12: Version control for the entire codebase is mandated through the use of Git (managed via GitHub), ensuring systematic tracking of all code changes, easy rollback capabilities, and efficient collaborative development among the team members.

CHAPTER 4

PROJECT DESIGN AND IMPLEMENTATION

The design of AI-Driven Alumni Connect converts the listed functional and non-functional requirements into a concrete system architecture. This section describes key design decisions ensuring the system is functional, scalable, secure, and maintainable. The design uses a contemporary cloud-native approach, emphasizing decoupled services, scalability, and the integration of external AI services.

We then describe the system architecture, the chosen methodology, and provide detailed breakdowns of data flow, use cases, and the system's components.

4.1 Proposed Project Architecture

The architecture of the AI-Driven Alumni Connect platform is a modern, cloud-native system. It uses a serverless Backend-as-a-Service (BaaS) model that separates the front-end interface from backend processing, data storage, and AI services. This decoupled, scalable design ensures the platform remains secure, high-performing, and easy to maintain.

4.1.1 Proposed Methodology

We adopted a serverless, cloud-native architecture based on Google's Firebase platform as the project's core approach. This choice was made to reduce costs, speed up development, and avoid the maintenance burden of traditional self-hosted backends. Using a BaaS model lets the development team offload backend tasks (database management, server setup, load balancing, authentication, etc.) to the service provider. This allows the team to focus almost exclusively on frontend feature development and enhancing the user experience.

The second key component of our methodology is a Hybrid-AI Integration Strategy. This approach recognizes that not all intelligent features are equal.

1. For General, Complex Tasks: We utilize powerful, pre-trained, general-purpose models via external API calls. The integration of the Google Gemini API is the prime example. It provides state-of-the-art natural language understanding for the AI Chatbot and advanced multimodal capabilities for resume parsing. This approach eliminates the massive cost and effort of training our own foundational models.

2. For Specific, Domain-Task: We utilize custom-trained, lightweight machine learning models for very specific, high-frequency tasks. As shown in the AI Integration block, this includes our own Alumni-Matching and Job Recommendation algorithms. These models are trained on our own platform data (e.g., user skills, branches, job descriptions) to provide highly relevant, domain-specific suggestions that a general-purpose AI would not be able to.

By combining Firebase and AI modules, the system efficiently meets its technical goals in performance, safety, and growth.

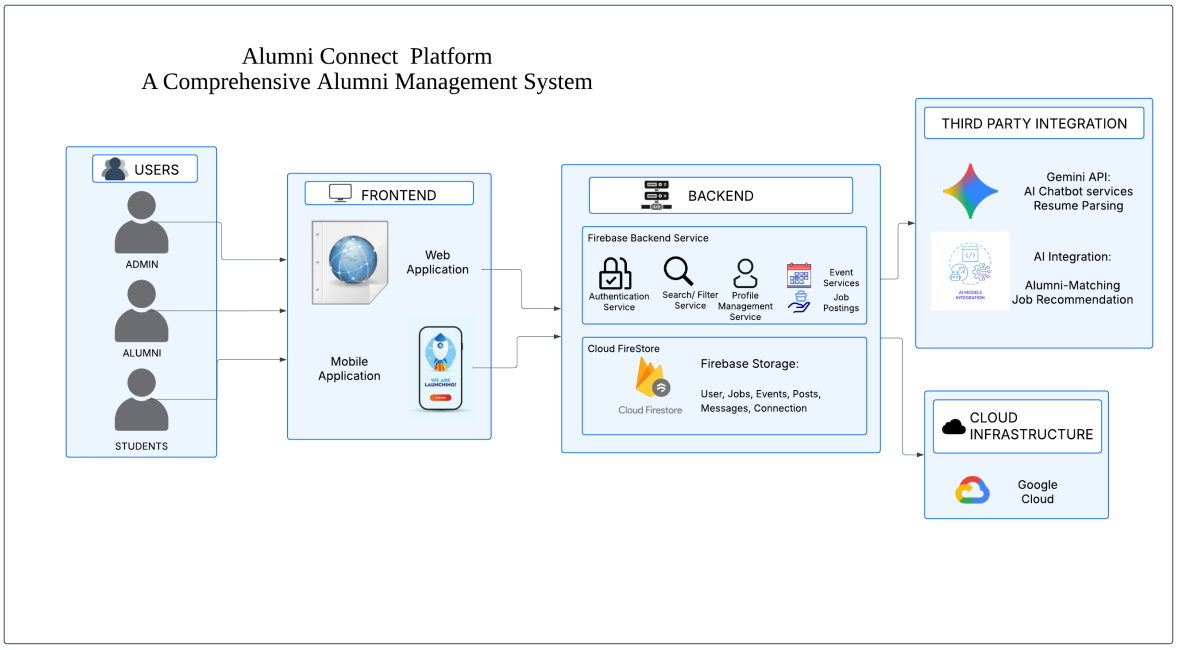


Figure 4.1: System Architecture of Alumni Connect Platform

4.1.2 Overall Description

The systems workflow, depicted in Figure 4.1, is a logical flow of data from the user actors, through the presentation and service layers, to the data persistence layer, with intelligent features provided by an external integration layer. The entire system is logically segmented into six primary components, which are detailed below.

User Layer (Actors)

This layer represents the external entities who interact with the system. Each actor has a distinct role and set of permissions, which are enforced by the backend authentication service.

- Admin: The Admin is the institutional stakeholder. This users primary function is governance, quality control, and oversight. Their key interactions include verifying alumni profiles to maintain network trust (FR4), moderating and approving job postings (FR10), creating and managing official university events (FR11), and viewing analytics dashboards that report on platform engagement.

- Alumni: The Alumni are the core value-providers of the ecosystem. As verified graduates, they are empowered to bring professional opportunities and experience into the platform. Their key functions include creating and maintaining their professional profile, posting job and internship opportunities (FR8), offering to mentor current students (FR5), and networking with their peers.

- Students: The Students are the core value-consumers who leverage the network for their professional growth. Their key functions include searching the directory to find and connect with relevant alumni, requesting mentorship from willing graduates (FR6), searching and applying for exclusive jobs and internships (FR9), and attending events.

Presentation Layer (Frontend)

The Frontend layer is the sole entry point for all user interaction and is responsible for rendering the User Interface (UI). The architecture is client-agnostic, supporting both web and mobile platforms to ensure wide accessibility (NFR2).

- Web Application: This is the primary client, developed as a Single Page Application (SPA) using React.js. This technology choice enables the creation of a fast, responsive, and component-based interface that provides an app-like experience within any modern desktop or mobile browser. It communicates directly with the Firebase backend.

- Mobile Application: As shown in the diagram, a dedicated mobile application is designed to run on iOS and Android. This provides a platform-native experience, which is crucial for features requiring on-the-go engagement, such as real-time chat notifications and event reminders.

Service/Logic Layer (Firebase Backend Service)

This is the heart of the BACKEND block and the central processing unit of the application. Instead of a single, monolithic server, it is a suite of serverless functions and managed services that handle all business logic.

- Authentication Service: This service manages the entire user lifecycle. It securely handles user registration, login (FR1, FR2), and password resets. It issues secure JSON Web Tokens (JWTs) that verify a users identity and role (Admin, Alumni, Student) with every subsequent request. This is critical for enforcing the systems Role-Based Access Control (RBAC) security rules (NFR6).

- Search/Filter Service: This is the primary query engine for the platform. It allows users to perform complex, multi-faceted queries on the Firestore database—for example, find all verified alumni who graduated after 2015, work at Google, and have Python as a skill. This service directly enables the core networking and mentorship features (FR6).

- Profile Management Service: This service handles all CRUD (Create, Read, Update, Delete) operations for user data. It allows users to update their work history, education, and skills (FR3), and manages the privacy settings for profile visibility (NFR7).

- Event Services & Job Postings: These are dedicated content-type managers. They handle the full lifecycle of time-sensitive documents, including creation by an alumnus, submission for administrative approval (FR10), public listing upon approval, and eventual archival after the event date or application deadline has passed.

Data Persistence Layer (Firestore & Storage)

This layer is responsible for the secure and scalable storage of all platform data. It is composed of two distinct but integrated Firebase products.

- Cloud Firestore: This is the primary database for the entire application. It is a NoSQL, document-oriented database. This technology was a strategic choice over a traditional SQL database because its flexible, JSON-like schema is perfectly suited for storing complex, nested data like user profiles, which may have a variable number of work experiences or skills (NFR11). The diagram indicates the key data collections: User, Jobs, Events, Posts, Messages, and Connection. Its real-time capabilities also natively power the live chat and notification features.

- Firebase Storage: This is the dedicated object storage solution for unstructured binary data. Its role is to host large files that are not suitable for a NoSQL database, such as user-uploaded profile pictures, resumes (PDF, DOCX, JPG) intended for parsing, and media attachments for event pages.

Third Party Integration Layer

This layer provides the advanced intelligence that differentiates the platform. It consists of both external, pre-trained models and internal, custom-trained models.

- Gemini API: This provides state-of-the-art, pre-trained generative AI capabilities from Google. It is leveraged for two critical functions:

    1. AI Chatbot Services: It powers the 24/7 user support chatbot (FR13). It understands users natural language queries and can provide answers from a knowledge base or offer personalized suggestions based on user data (FR14).
    2. Resume Parsing: This is a key automation feature for reducing user friction. A user uploads their resume (e.g., a PDF), and the Gemini APIs multimodal capabilities are used to read the document, extract structured data (like work history, skills, and education), and return it as JSON to auto-populate the users profile fields.

- AI Integration (Custom Models): These are lightweight, custom-trained machine learning models, likely hosted as serverless functions on Google Cloud. Unlike the general-purpose Gemini API, these models are trained specifically on our platforms data to perform highly specific, domain-relevant tasks:

    1. Alumni-Matching: A recommendation engine that suggests relevant peers or mentors to a user based on deep similarities like shared skills, career paths, branch, and professional history (FR7).
    2. Job Recommendation: A similar engine that matches a students profile (skills, branch, interests) against the database of available job postings to proactively suggest the most relevant opportunities.

Cloud Infrastructure Layer

This is the foundational layer upon which the entire system is built and deployed.

- Google Cloud: The entire platform operates within the Google Cloud ecosystem. Firebase itself is a product layer built on top of Google Clouds core infrastructure. This

foundation provides all the necessary components for a global-scale application, including data centers, networking, and computing resources. This choice ensures global low-latency access, automatic scalability to handle millions of users (NFR8), high availability and reliability (NFR10), and a unified management console for all services.

4.2 High Level Design

The detailed design phase translates the high-level system architecture into a granular specification of its internal processes, data pathways, and logical flow. This section provides a multi-layered decomposition of the system using Data Flow Diagrams (DFDs) to visually map the movement of information between external actors, internal processes, and data storage systems.

Data Flow Diagrams

Data Flow Diagrams (DFDs) are a structured design tool used to illustrate how data is processed by a system in terms of inputs and outputs. Our design employs a leveled set of DFDs, starting from a high-level context diagram (Level 0) and progressively exploding the central process into more detailed subprocesses (Level 1, Level 2) and logical workflows.

Level 0 DFD: Context Diagram

The Level 0 DFD, also known as the Context Diagram, is presented in Figure 4.2. Its purpose is to establish the systems boundary, identify the external entities (actors) that interact with it, and define the high-level data flows between those entities and the central system.

As shown, the entire system is represented as a single, abstract process: ALUMNI CONNECT SYSTEM. This process encapsulates all the functionality of the platform. The diagram identifies the three primary external entities that provide data to or receive data from the system:

- **ADMIN:** Represents the institutional administrator.
- **ALUMNI:** Represents the verified graduate users.
- **STUDENT:** Represents the currently enrolled student users.

The data flows in this diagram are high-level representations of the primary interactions:

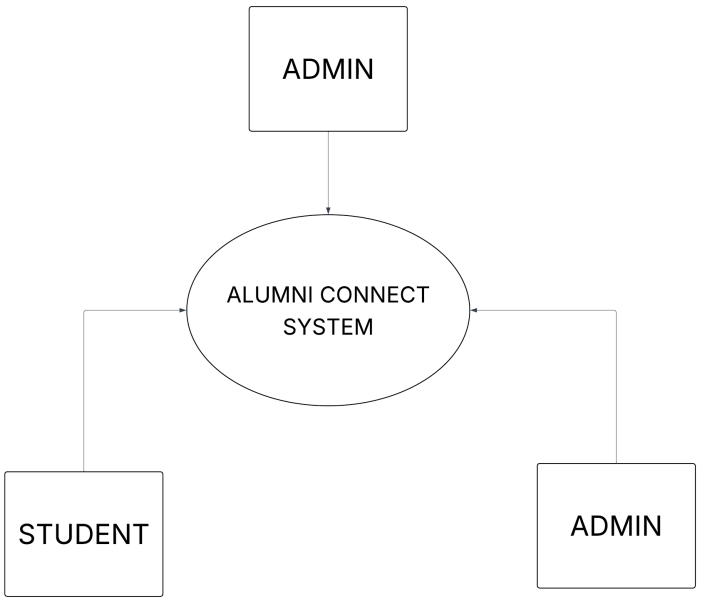


Figure 4.2: Level 0 DFD for AI driven Alumni Connect

- Data Inflows (to System): All three entities provide credentials for login and registration. The ADMIN provides verification decisions, event data, and content moderation commands. The ALUMNI provides profile updates, job postings, and mentorship availability. The STUDENT provides profile data, job applications, and mentorship requests.
- Data Outflows (from System): The system provides ADMIN with analytics reports and profiles pending verification. It provides ALUMNI with network data, peer profiles, and mentorship requests. It provides STUDENT with searchable alumni profiles, job listings, and event notifications.

This context diagram confirms the systems scope, identifying its primary stakeholders and its core inputs and outputs without detailing the internal mechanisms.

Level 1 DFD: System Decomposition

The Level 1 DFD, shown in Figure 4.3, explodes the singular process from the Level 0 diagram into the main functional sub-systems. This level introduces the primary processes that handle the data and reveals the logical data stores (databases) where information is held.

The diagram illustrates four major processes and one gateway process:

1. User Authentication: This process acts as the primary data gateway. All external entities (Admin, Alumni, Students, and even the Chatbot actor) must pass through authentication, which validates their credentials before granting access to other sub-systems.

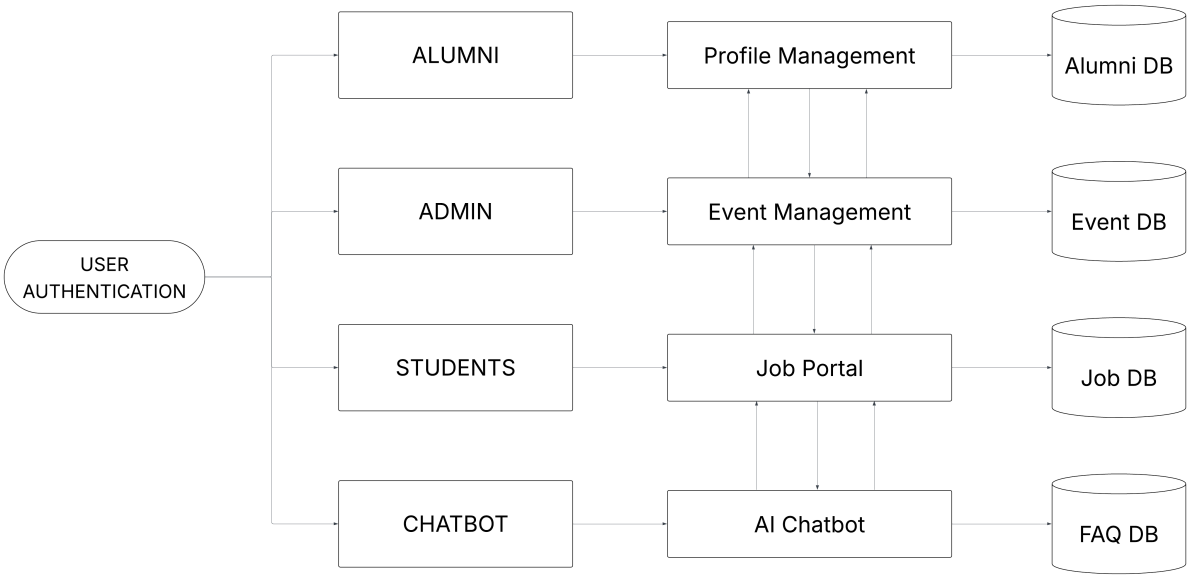


Figure 4.3: Level 1 DFD for AI driven Alumni Connect

2. Profile Management: This process handles all data related to user identities. It receives profile information from the ALUMNI entity and performs read/write operations on the Alumni DB data store.

3. Event Management: This process is controlled by the ADMIN entity. It is responsible for creating, updating, and publishing event information, which is stored in the Event DB. It also reads from this database to display event details to students and alumni.

4. Job Portal: This process is the two-way interface for employment opportunities. It receives new job postings from Alumni (a flow implied from the Level 0 diagram) and writes them to the Job DB. It receives search queries from STUDENTS and reads from the Job DB to return listings.

5. AI Chatbot: This process represents the intelligent support system. It is depicted as being used by a general CHATBOT entity (representing any user) and retrieving its information from a dedicated FAQ DB data store.

This DFD is critical as it decomposes the abstract system into its main functional components and, most importantly, identifies the specific data stores that support each function. This directly informs the database schema design, showing a clear separation of data concerns (e.g., user profiles, events, and jobs are all stored independently).

Level 2 DFD and DFD Model: Process and Logic Flow

The subsequent diagrams provide a more granular view of specific processes and the logical flow of control within the system.

Level 2 DFD: Data-Centric View

Figure 4.4 presents a Level 2 DFD that expands upon the ALUMNI CONNECT SYSTEM process with a focus on the specific data products being exchanged.

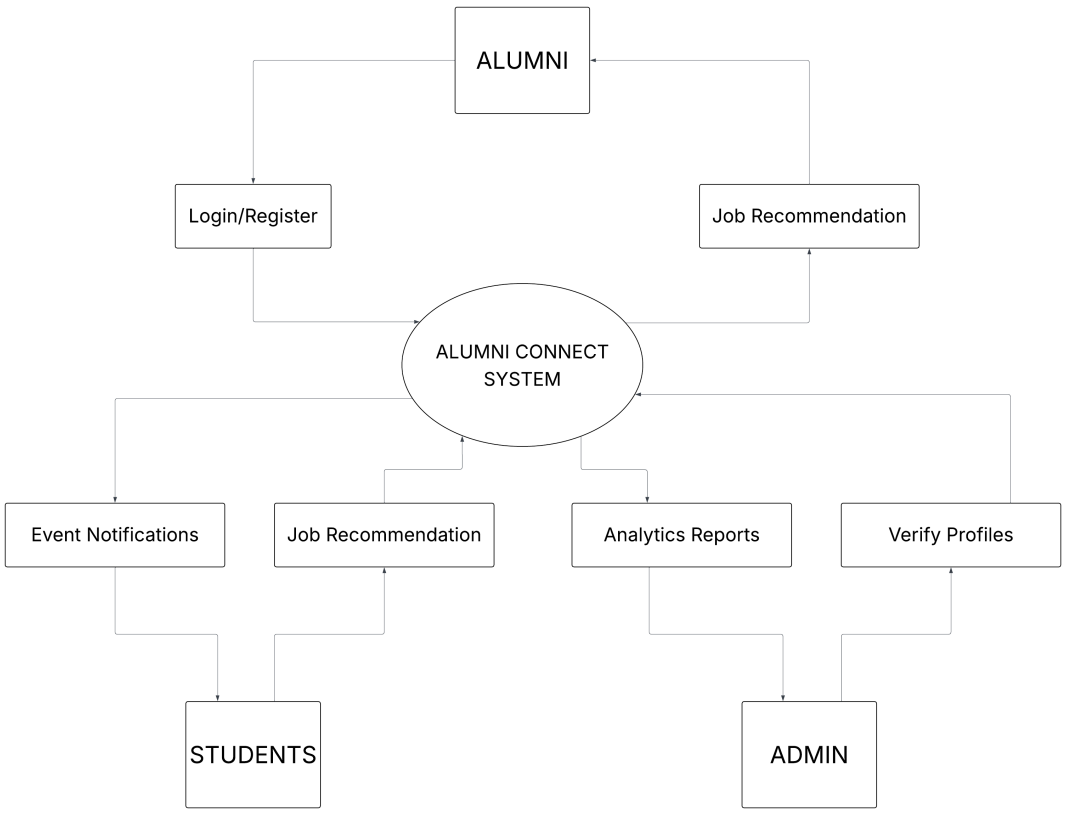


Figure 4.4: Level 2 DFD for AI driven Alumni Connect

This diagram details the inputs and outputs for each actor:

- ALUMNI: This entity provides data to the Login/Register process and receives data from the Job Recommendation process. This flow highlights the reciprocal nature of the platform, where alumni contribution (profile data) is rewarded with valuable output (intelligent recommendations).
- STUDENTS: This entity is primarily a consumer of data, receiving Event Notifications and Job Recommendations from the system.
- ADMIN: This entity interacts with the system through two key processes: they receive

Analytics Reports (a data-out flow) and provide decisions to the Verify Profiles process (a data-in flow, though the arrow implies a two-way interaction).

This view is particularly useful for understanding the systems value proposition from a data-centric perspective, showing exactly what information products are generated for each user role.

DFD Model: Logical Workflow

Figure 4.5 moves beyond a traditional DFD to show a DFD Model, which functions more as a logical flowchart. This diagram is crucial for understanding the systems control flow and business logic from the users perspective, particularly after authentication.

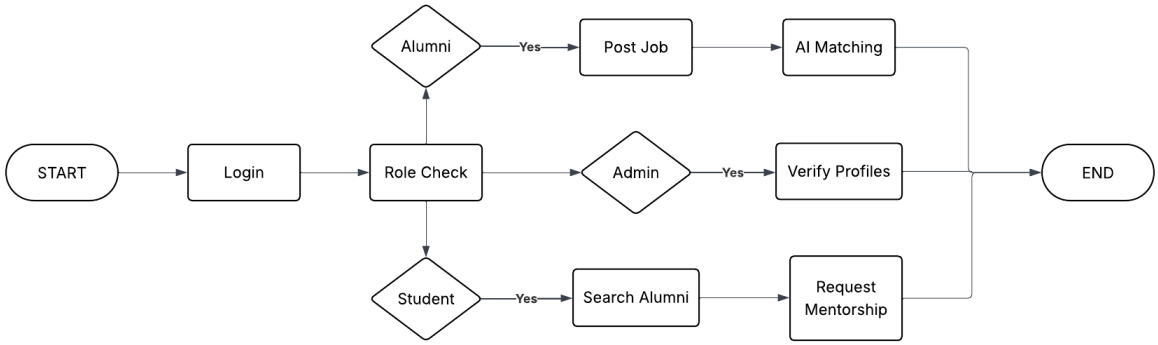


Figure 4.5: DFD Model for AI driven Alumni Connect

The workflow illustrates the primary user journeys:

1. Start and Authentication: The flow begins at START and moves to Login.
2. Role Check: Immediately after successful login, the system performs a Role Check. This is the central decision node that dictates the users subsequent pathway and available functionality.
3. Three User Paths:
    - Alumni Path: If the role is Alumni, the user is directed to their dashboard, from which they can execute tasks like Post Job. This process then feeds data to the AI Matching system before the flow terminates (END).
    - Admin Path: If the role is Admin, the user is routed to administrative functions, such as the Verify Profiles queue, before reaching the END of that specific task.

- Student Path: If the role is Student, the user is directed to student-centric features, such as Search Alumni, which can then lead to a subsequent action like Request Mentorship, completing the task flow.

This DFD Model effectively outlines the core business logic of the application, demonstrating how the role-based access control (defined in the requirements) is implemented as a concrete, branching workflow that presents a unique and tailored experience to each of the three main user groups.

4.3 Detailed Design

4.3.1 Use Case Design

A Use Case diagram is a fundamental tool in system design, acting as a high-level blueprint of a systems functionality from an external perspective. It captures the dynamic behavior of the system by illustrating the interactions between external entities, known as Actors, and the specific functions they can perform, known as Use Cases.

For the AI-Driven Alumni Connect platform, the Use Case diagram is essential for defining the scope of the project. It graphically partitions the systems functionality, assigning specific capabilities to each of the primary user roles: Admin, Alumni, and Student. This clear separation of concerns is the foundation upon which the role-based access control, user interface design, and backend business logic are built.

Overview of Use Case Diagram

The Use Case diagram, as presented in Figure 4.6, employs standard UML (Unified Modeling Language) notation to map out the systems functional requirements. The diagram is organized into three horizontal swimlanes, one for each of the primary human actors, with a fourth system actor supporting the Student-facing functions.

- Actors: These are the stick figures shown on the left-hand side and bottom of the diagram. They represent the entities that interact with the system but are external to it. We have identified four key actors:

    1. Admin (Red): Represents the institutional staff who manage the platforms integrity and content.

2. Alumni (Blue): Represents the verified graduates of the institution, who are the primary value-creators within the network.
3. Student (Teal): Represents the currently enrolled students, who are the primary beneficiaries of the platforms resources.
4. AI Chatbot (Purple): This is a non-human system actor, representing the external automated service that fulfills the Chatbot Help use case.

- Use Cases: These are the ovals, which represent a specific, discrete piece of functionality that provides a measurable result to an actor. The diagram uses color-coding to associate use cases with their primary actor (red for Admin, blue for Alumni, green for Student, and purple for AI-related functions).
- Associations: The solid lines indicate a direct association between an actor and a use case. The arrow from an actor to a use case signifies that the actor initiates that function.
- Dependencies: The directed, dashed, or solid arrows between use cases (or from a use case to an actor) represent dependencies. These can be include relationships (where one use case must be completed as part of another) or extend relationships (where one use case provides optional, additional functionality for another). In this diagram, they are used to show logical process flow, such as Search Alumni leading to the option to Request Mentorship.

Use Case Diagram for the Project

Figure 4.6 provides a complete visual summary of all interactions that define the AI-Driven Alumni Connect platform.

1. Admin Actor

The Admin actors role is purely managerial and focused on governance and quality control. They do not interact with the student-facing networking features but are critical for maintaining the platforms integrity.

- Manage Profiles: This use case grants the Admin high-level control over all user profiles in the system.
- Verify Profiles: This use case is shown as a logical consequence of Manage Profiles. When an alumnus registers, their profile enters a pending state. The Admin must perform this Verify Profiles function to approve their account, thereby granting them access to

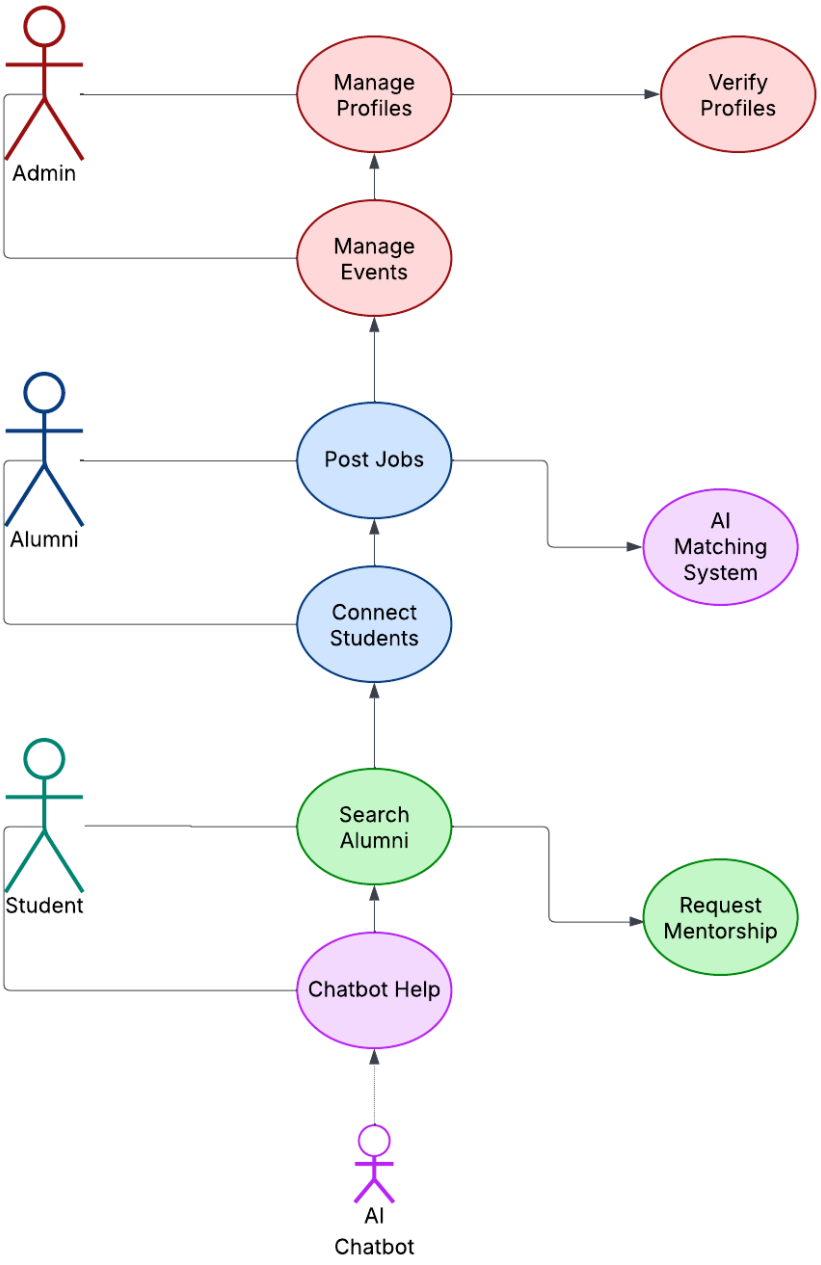


Figure 4.6: Use-Case Representation for AI driven Alumni Connect Platform

alumni-specific features like posting jobs. This is the primary gatekeeping mechanism for the platform.

- Manage Events: This use case allows the Admin to create, edit, publish, and remove official institutional events, such as reunions, webinars, or career fairs.

2. Alumni Actor

The Alumni actor is the primary source of professional value for the network. Their use cases are centered on giving back to the community.

- Post Jobs: This is a key function, allowing a verified alumnus to post a job or internship opportunity. The diagram shows a directed arrow from this use case to the AI Matching System. This signifies that the act of posting a job automatically triggers a backend process where the AI analyzes the job description to later match it with suitable student profiles.
- Connect Students: This represents the alumnuss ability to proactively browse the student directory, find promising candidates, and initiate contact, perhaps for networking or recruitment. This complements the students ability to search for alumni.

3. Student Actor

The Student actor is the primary consumer of the platforms resources. Their use cases are focused on leveraging the alumni network for professional development.

- Search Alumni: This is the students primary tool for exploring the network. This use case allows them to filter and find alumni based on criteria like company, branch, or skillset.
- Request Mentorship: This use case is shown as an extension of Search Alumni. The arrow indicates that a student first searches for and finds a suitable alumnus, and then they can optionally choose to initiate a Request Mentorship action.
- Chatbot Help: This is the students primary support channel. The dotted line to the AI Chatbot actor signifies that this use case is realized by that external system, which will field the students questions.

Workflow of the System

The Use Case diagram defines the logical workflows and user journeys through the platform. While the DFDs in the previous section show the data flow, this diagram explains the interaction flow from the users perspective.

The primary workflow begins with authentication (implied, as all actors are users). Once logged in, the system presents a different set of functionalities based on the actors role.

- Admin Workflow: The Admins workflow is task-oriented. They log in and are presented with a dashboard. A primary task is to process the verification queue. They select Manage Profiles, view pending accounts, and execute the Verify Profiles sub-task for each one. A secondary workflow is content creation, where they select Manage Events to publish a new event for all users.

- Alumni Workflow (as provider): A verified alumnus decides to share an opportunity. They initiate the Post Jobs use case, fill out a form, and submit it. This action completes their workflow for this task but initiates a new, automated workflow for the AI Matching System, which logs the jobs details for future recommendations.

- Student Workflow (as consumer): A student seeking guidance initiates the Search Alumni use case. After applying filters (e.g., Computer Science, Google), they are presented with a list of profiles. From this results list, their workflow branches. They may choose to initiate the Request Mentorship use case for a specific alumnus, which sends a notification to that alumnus.

- Student Support Workflow: At any point in their journey, if the student is confused, they can click the help icon, initiating the Chatbot Help use case. This opens an interface where they interact directly with the AI Chatbot actor in a query-and-response cycle, which runs parallel to all other platform functions.

These distinct workflows are designed to be complementary, creating a virtuous cycle of engagement. For example, an Alumnuss Post Jobs workflow directly feeds the AI Matching System, which in turn provides value to the Students Search Alumni and Chatbot Help workflows by offering intelligent job recommendations.

Advantages of the Use Case Diagram

The development and refinement of this Use Case diagram provide several critical advantages for the project lifecycle:

- Clear Scope Definition: The diagram acts as a visual contract. It clearly and unambiguously defines the boundaries of the AI-Driven Alumni Connect platform. Any function not represented by a use case oval is understood to be out of scope for the current project phase, preventing feature creep.

- User-Centric Focus: The entire design is centered on the actors. By starting with who (Admin, Student, Alumni) and what (their goals), the design ensures that all development work is directly tied to providing tangible value to a specific user, leading to a more intuitive and useful final product.

- Facilitation of Communication: The diagrams simplicity makes it an ideal communication tool. It can be shared with project stakeholders (such as the project guide or insti-

tutional representatives) who are not technical, allowing them to easily understand and validate the systems intended functionality without needing to read complex requirement documents.

- Foundation for Functional Requirements: Each use case in the diagram (e.g., Post Jobs, Verify Profiles) maps directly to one or more functional requirements. This diagram provides a high-level grouping that helps organize the detailed Software Requirements Specification (SRS) document.
- Basis for Testing and Validation: The diagram provides a clear roadmap for the testing phase. Each use case and its associated workflows becomes a candidate for a test plan. For example, testers can design specific test cases to validate the Student → Search Alumni → Request Mentorship pathway from end to end.

4.3.2 Sequence Diagram

The Sequence Diagram for the AI-Driven Alumni Connect platform, presented in Figure 4.7, deconstructs the system into its core object lifelines and maps the exact messages passed between them to accomplish a specific task. This diagram is essential for understanding the systems runtime behavior, identifying potential bottlenecks, and defining the contracts for inter-component communication.

The diagram identifies the following key object lifelines:

- Actors: Student, Alumni, and Admin (the external human interactors).
- Frontend: The client-side application (React.js) that runs in the users browser and serves as the primary interface.
- FirebaseAuth: The dedicated Google Firebase service for securely managing user authentication, including account creation and sign-in.
- CloudFirestore: The primary NoSQL database service, responsible for all data persistence (user profiles, jobs, events, etc.).
- FirebaseStorage: The object storage service for hosting files such as degree proofs, profile pictures, and resumes.
- CloudFunctions: Serverless backend functions that contain proprietary business logic, such as the AI matching algorithm, which should not run on the client.

- GeminiAPI: The external, third-party AI service from Google, used for advanced tasks like the conversational chatbot.

The complete diagram is divided into four primary scenarios, or interaction fragments, each of which details a critical user-facing workflow.

Interaction 1: Registration & Verification

This is the most complex sequence, as it involves three distinct actors (Student, Admin) and multiple asynchronous steps to ensure a new user is both created and properly verified.

Part A: Student Account Creation

1. The flow begins when the Student actor selects the Initiate Registration option on the Frontend.
2. The Frontend responds by displaying the registration form.
3. The Student fills in their details (email, password, basic profile info) and hits Submit.
4. The Frontend receives these details and performs two critical, parallel backend operations:
    - Authentication: It sends the email and password to FirebaseAuth using the CreateUserWithEmailAndPassword method. This is a vital security step, as the raw password never touches the main database.
    - Authorization: FirebaseAuth validates the credentials, creates the user in its own secure system, and returns a unique User ID to the Frontend.
5. Database Creation: Upon receiving the successful User ID, the Frontend immediately creates a new document in the users collection in CloudFirestore, saving the rest of the profile info (name, branch) associated with that User ID.
6. The CloudFirestore service confirms the Profile Saved message, and the Frontend displays a Notify: New student registered message to the Student, completing the initial registration.

Part B: Alumni Verification (Asynchronous)

This part of the flow occurs after initial registration, when a user with the role Alumni (or a student transitioning to alumni) must be verified.

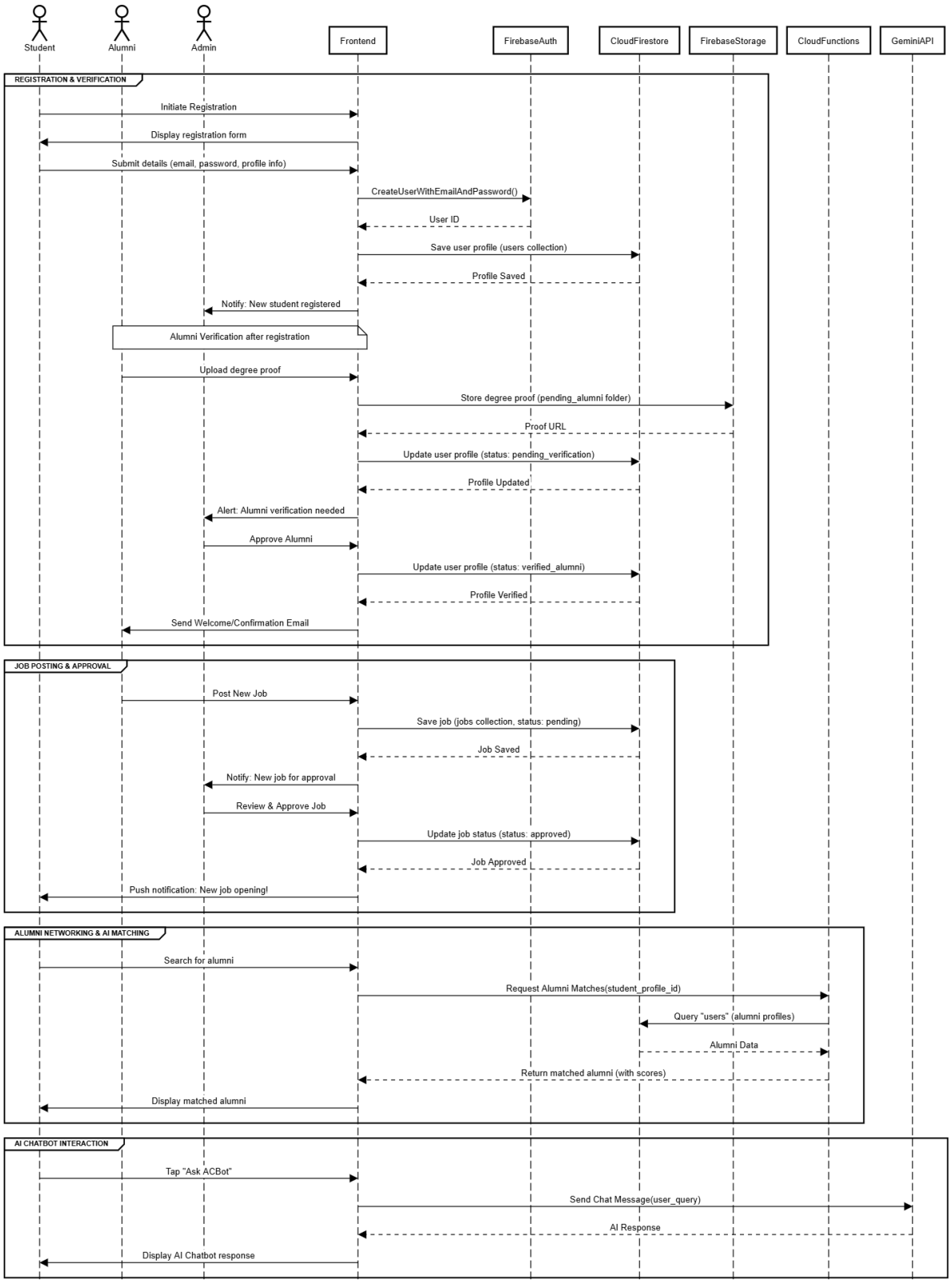


Figure 4.7: Sequence Structure of AI Driven Alumni Connect Platform

1. The newly registered user (now acting as a Student or pending Alumni) is prompted to Upload degree proof via the Frontend.

2. The Frontend sends the file (e.g., a PDF or JPEG) to FirebaseStorage, which is optimized for file handling.

3. FirebaseStorage saves the file (e.g., in a pending_alumni folder) and returns a secure Proof URL to the Frontend.

4. The Frontend then updates the users document in CloudFirestore, setting their status: pending_verification and adding the Proof URL.

5. This update triggers a notification (represented by the message to the Admin lifeline) that a new user needs verification.

6. The Admin actor, in their separate admin portal, views the pending profile and the proof. They then click Approve Alumni.

7. This fires a message from the Frontend to CloudFirestore, executing the final Update user profile (status: verified_alumni) command.

8. Once the profile is verified, a confirmation (e.g., Send Welcome/Confirmation Email) is sent to the Student/Alumni, who now has full platform access.

This sequence demonstrates a clear separation of concerns: FirebaseAuth for identity, CloudFirestore for data, and FirebaseStorage for files, all coordinated by the Frontend and governed by the Admin.

Interaction 2: Job Posting & Approval

This sequence details the moderation workflow, which ensures that all content posted to the student-facing job board is vetted by an administrator.

1. A verified Alumni actor initiates the flow by selecting Post New Job on the Frontend.

2. After filling out the job details, the Frontend saves the job data as a new document in the jobs collection in CloudFirestore. Critically, it sets the initial status: pending.

3. Upon successful save (Job Saved), the Frontend immediately sends a Notify: New job for approval message to the Admin actor (e.g., in their approval dashboard).

4. The Admin reviews the pending job and selects Review & Approve Job.

5. This action triggers the Frontend to send an Update job status (status: approved) message to the specific job document in CloudFirestore.

6. Once the database confirms the Job Approved, the Frontend executes the final step: sending a Push notification: New job opening! to the relevant Student audience.

This workflow is highly effective as it allows alumni to contribute content at any time, but ensures administrative quality control before that content is broadcast to students.

Interaction 3: Alumni Networking & AI Matching

This sequence is highly significant as it demonstrates the integration of a custom, server-side AI model, rather than a simple database query.

1. The Student actor initiates a Search for alumni from their Frontend.
2. The Frontend does not query the database directly. Instead, it packages the students profile information and search terms and sends a Request Alumni Matches message to CloudFunctions. This is a critical security and performance step, as it moves complex, proprietary logic off the client.
3. The CloudFunctions (serverless backend) receives the request and executes the AI matching logic. As part of this, it first queries CloudFirestore for the list of all alumni profiles.
4. CloudFirestore returns the raw Alumni Data to the CloudFunctions.
5. The CloudFunctions lifeline then performs the computation (applies the matching algorithm, calculates relevance scores) and sends a single, curated list back to the Frontend: Return matched alumni (with scores).
6. The Frontend simply has to Display matched alumni to the Student, having done none of the complex computation itself.

Interaction 4: AI Chatbot Interaction

This final sequence is a simple, direct-to-API interaction, contrasting with the more complex AI matching flow.

1. The Student actor clicks Tap Ask ACBot on the Frontend.
2. The Frontend captures the users query.
3. It sends the Send Chat Message(user_query) directly to the external GeminiAPI lifeline. This implies a client-side SDK is being used for a task that does not require proprietary data or complex logic (like a simple Q&A).
4. The GeminiAPI processes the natural language query and sends back the AI Response.
5. The Frontend receives this response and simply displays it to the Student.

4.4 Technical Requirements and System Interfaces

This section outlines the specific, mandatory technology stack and details precisely how the various components of the platform interact to form the cohesive technical ecosystem.

4.4.1 Software Requirements Stack

The technical stack was a deliberate choice, made specifically for its performance, native scalability, and seamless integration capabilities with cloud-based AI services.

- Frontend Framework: React.js (Mandatory component-based choice for dynamic UI).
- Styling and Layout: Tailwind CSS (Mandatory utility-first approach for rapid, responsive design).
- Backend Architecture: Backend-as-a-Service (BaaS) utilizing Google Firebase (Mandatory for serverless scaling).
- Database System: Cloud Firestore (Mandatory NoSQL, document-oriented database).
- Authentication/Storage: Firebase Authentication and Firebase Storage (Core Firebase services).
- AI Service: Google Gemini API (Mandatory external service for intelligence).

4.4.2 Data Model and Entities

The integrity and speed of the system rely entirely on the structure of data within Cloud Firestore. The system revolves around four primary data collections, which must be carefully indexed for rapid retrieval:

1. Users Collection: This is the central repository, storing core identity data (`uid`, `role`, `isVerified`, `email`) alongside extensive nested fields for professional and academic history. The structure must be optimized for multi-dimensional querying (e.g., searching by both batch and skills).
2. Jobs Collection: Stores detailed job postings (`title`, `description`, `alumnusID`, `status`, `datePosted`). The collection must facilitate efficient filtering and include a nested array to track applicants (`applicants: []`), with each entry containing the student ID and application date.

3. Events Collection: Stores all event details (`title`, `date`, `location`, `organizerID`, `description`). The collection must be indexed by date and must include a nested array for tracking event registration (`attendees: []`), essential for capacity management and follow-up communication.

4. Chat Collection: Designed for efficient, real-time messaging, storing sequential message documents for all direct communication and formal mentorship request logs (`senderID`, `recipientID`, `message`, `timestamp`). Queries must be rigorously optimized for retrieval by message participants and timestamp order.

4.4.3 External Interfaces

- User Interface (GUI): The application must present a singular, unified web-based GUI accessed via a uniform URL. This interface must be tailored dynamically to the users authenticated role (Student, Alumni, or Admin). A highly visible, persistent chatbot icon must be available across all primary navigational views for immediate AI-powered support.

- Software Interfaces (API):

  - Firebase SDK: The React client communicates with Firebase services exclusively through the official JavaScript SDK, managing secure connections, all data exchange, and user state changes, abstracting away low-level networking concerns.

  - Gemini REST API: All AI features rely on secure, direct HTTPS POST requests to the Google Gemini REST API endpoint. The application must accurately handle request formatting (including base64 encoding of file uploads for the resume parser) and strictly parse the incoming structured JSON responses to maintain data integrity.

- Database Interfaces: All data persistence operations (CRUD - Create, Read, Update, Delete) are exclusively executed via the high-level methods provided by the Cloud Firestore SDK, leveraging methods like getDoc, setDoc, and onSnapshot to facilitate the necessary efficient real-time data flow.

4.5 Implementation

The implementation phase represents the tangible realization of the architectural blueprints and functional requirements detailed in the preceding chapters. This chapter provides a comprehensive account of the development process, translating the systems design into a functional, secure, and scalable application. The implementation was a multi-faceted endeavor, bifurcated into several parallel streams: setting up the cloud and local execution environments, developing the frontend and administrative user interfaces, configuring the backend services and database, and training the artificial intelligence models that power the platforms intelligent features.

This chapter will meticulously document each of these aspects, detailing the software stacks, data collection methodologies, module-wise code development, and the specific algorithms that form the core of the AI-Driven Alumni Connect platform.

4.5.1 Setting up of an Execution Environment

A stable and consistent execution environment is a non-negotiable prerequisite for a complex, cloud-native application. Our project environment was complex, involving separate setups for the frontend, backend services, and machine learning model training.

Frontend Development Environment

The primary user-facing application is a Single Page Application (SPA) built using React.js.

- Core Framework: The environment was initialized using Node.js (v18.x or higher) and the Node Package Manager (npm). The React framework (v18.x) was used, bootstrapped via create-react-app for a standardized project structure, build scripts, and a local development server.
- Libraries: Key libraries were installed via npm:
  - `react-router-dom`: To handle client-side routing and navigation between different pages (e.g., Dashboard, Profile, Jobs) without full-page reloads.
  - `lucide-react`: A lightweight and consistent icon library used across all components to ensure a clean, modern user interface.
  - `recharts`: A composable charting library used specifically in the Admin Panel for data visualization, such as rendering the bar charts for user statistics.

- Configuration: The local development environment was configured with a .env.local file. This file is critical for security and modularity; it stores all Firebase API keys and the Google Gemini API key, ensuring that no sensitive credentials are hard-coded into the source code.

Backend Cloud Environment (Google Firebase)

The entire backend is serverless and operates within the Google Cloud ecosystem, managed through the Firebase console.

- Firebase Project: A new project was created in the Firebase console, which serves as the central management hub for all backend services.
- Authentication: Firebase Authentication was enabled, specifically the Email or Password sign-in provider. This service handles all user registration, login, session management, and secure password reset functionalities out of the box.
- Cloud Firestore: The Firestore NoSQL database was provisioned. This involved setting up the initial database rules for security (e.g., defaulting to deny all) which were later expanded to enforce the platforms Role-Based Access Control (RBAC) model (NFR6). No manual server setup or database provisioning was required.
- Firebase Storage: A storage bucket was created to handle file uploads, such as profile pictures and, most critically, the resumes required for AI parsing. Storage security rules were configured to ensure that users can only write to their own designated folders and read files that are publicly designated.

AI & Machine Learning Environment

A separate, offline environment was required to train the custom AI models for recommendations.

- Language and Runtime: Python (v3.9 or higher) was used as the exclusive language for all machine learning tasks.
- Core Libraries: The environment was built using a requirements.txt file or a virtual environment (e.g., venv, Conda) to manage dependencies:
  - `pandas`: Used for data loading, manipulation, and high-performance feature engineering.

  - `scikit-learn`: The core machine learning library. It was used for splitting the data (train_test_split) and for implementing the RandomForestClassifier model.
  - `joblib`: A standard Python library used for serializing (saving) and deserializing (loading) the trained machine learning models (`.joblib` files).

- API Services: For the generative AI features, a Google Cloud project was configured with the Gemini API enabled. This provided the API key necessary for the frontend React application to make secure POST requests for the chatbot and resume-parsing functionalities.

4.5.2 Dataset Collection

A significant challenge in any AI-driven project is the acquisition of high-quality training data. As this platform was built from the ground up, pre-existing, labeled data was not available. Therefore, a synthetic dataset collection and manual labeling methodology was employed to bootstrap the two core recommendation models.

Alumni-Matching Dataset (`training_data.csv`)

The goal of this model is to predict a match score between a current student and a potential alumni mentor. The training script for this model specifies the required data schema.

- Schema: The dataset was structured as a CSV file with the following columns: viewer_branch, viewer_skills, target_branch, target_skills, target_company, and a target variable is_match.
- Collection Process: This dataset was synthetically generated.
  1. A list of representative, hypothetical student profiles was created, detailing their branch and a pipe-separated list of skills (e.g., Computer Science, python—java—sql).
  2. A similar list of hypothetical alumni profiles was created, including their branch, skills, and current company (e.g., Mechanical, solidworks—autocad, Tesla).
  3. These profiles were then cross-referenced. A match (is_match = 1) was manually labeled based on logical rules: a strong match if branches are identical and skill overlap is high; a partial match if branches differ but skills are relevant (e.g., CS student to a Data Scientist alumnus from the ECE branch); and a non-match (is_match = 0) if there was no logical connection.

- Purpose: This labeled dataset provides the model with concrete examples of what constitutes a good match, allowing it to learn the complex relationships between branches, skills, and company affiliations.

Job Recommendation Dataset (`job_training_data_v3.csv`)

The objective of this dataset is to enable the prediction of a match quality score—a continuous value between 0.0 and 1.0—representing how well a particular student aligns with a given job posting. Instead of limiting the task to classification, this design adopts a regression-based method, which provides a finer understanding of compatibility.

- Schema: The dataset contains the following key attributes: user_skills, userexperience, job_skills, job_experience_required, and the dependent variable match_quality.
- Collection Process: The dataset was synthetically constructed through the script create_dataset_v3.py to reflect realistic job–candidate relationships.
    1. A large pool of virtual student profiles and job listings was generated, each with randomized combinations of skills (e.g., python—react—aws) and varying experience levels.
    2. A rule-based evaluation system then assigned a match_quality value to each pair, simulating how a domain expert would score compatibility.
    3. The score computation considered multiple aspects, such as:
        - The proportion of job-required skills present in the student's skill set.
        - Extra weighting for high-impact skills like python or aws.
        - Penalty factors for large mismatches between the candidate's experience and the job's requirements.
- Purpose: This dataset forms the foundation for training a regression-oriented recommendation model. By learning how well different combinations align, the model can generate a ranked list of job recommendations—from most to least suitable—rather than a simple binary outcome.

4.5.3 Module-wise Code Development

The platforms code is logically divided into three primary projects: the user-facing React Frontend, the Admin Panel (also built in React), and the Python-based AI/ML scripts. The

implementation focused on modular, component-based development to ensure maintainability (NFR11).

Frontend Application (App.js)

The main user application is a single, large React file that defines all components and pages. The core of its implementation revolves around React Context for state management and Firebase services for backend operations.

Core Services (Context and Firebase)

- AppProvider and useAppContext: This is the root of the state management. The AppProvider component creates a global React Context that holds the current user object, the Firestore dbInstance, the authInstance, and the appId. Any child component can then access this critical state using the useAppContext hook, which eliminates the need to pass props down through many layers of components.
- Firebase API Functions: A set of helper functions wrap the core Firebase SDK calls to provide a clean API for the rest of the application.
  - getUserProfile: A function that takes a userId and retrieves the corresponding document from the users collection in Firestore. This is the primary function for fetching profile data.
  - createUserProfile: This function is called during sign-up. It takes the userId from FirebaseAuth and the profileData from the form and uses setDoc to create the new user document in the users collection, adding a createdAt timestamp.
  - updateProfile: Allows users to save changes to their profile by using the updateDoc command on their specific user document.
  - findOrCreateChat: This function implements the logic for starting a 1-on-1 chat. It queries the messages collection for an existing chat where the participants array contains both the currentUserId and the otherUserId. If one is found, it returns the existing chat ID. If not, it creates a new chat document with the two participants.

Page-Level Components

The applications navigation, managed by the MainContent components state, renders one of the following page components:

- HomePage: The main landing page. It features the primary call to action button and a grid of the platforms features (Alumni Directory, Job Portal, etc.). It also fetches and displays recent activity (jobs and events) for logged-in users.

- Dashboard: This is the users home base. It implements tabbed navigation (Summary, Jobs, Events, Posts). The Summary tab is notable for integrating the JobRecommendations component, which fetches and displays AI-driven job matches. It also includes a ResponsiveContainer from recharts to show user statistics.

- AlumniDirectory: A key networking feature. This component fetches all user profiles from Firestore where role == alumni. It includes significant client-side logic for filtering (by search term), sorting (by AI Match Score vs. Batch), and grouping. It also manages the state for sending a connect request, which updates the connections document in Firestore.

- Jobs and Events: These components are similar. They fetch all documents from the jobs or events collections, respectively, and render them as cards. They include modal pop-ups for alumni to post new jobs or events, which involves writing a new document to the corresponding Firestore collection. The Events component also manages the logic for registering/unregistering, which writes the users ID to a subcollection (registrations) under the specific event document.

- Profile: This is one of the most complex components. It manages two states: view mode and edit mode.

  - In view mode, it renders the users data using sub-components like ProfileHeadline, ExperienceSection, and EducationSection.

  - In edit mode (isEditing == true), it renders a large form. This form allows for dynamic addition/removal of experience and education items. It also contains the logic for the AI Resume Parsing, which calls the extractResumeData function.

- Messenger: This component implements the full-featured, real-time chat. It is divided into two panels. The left panel queries and displays all chat documents where the participants array contains the current users ID, listening for real-time updates using onSnapshot. The right panel displays the messages array from the currently activeChat and provides an input to updateDoc the array with a new message object.

AI Integration Components (Frontend)

- Chatbot: This component implements the AI support feature. It maintains a chatHistory in its state. When a user sends a message, it calls the Google Gemini API via a secure fetch request. It sends the entire chat history in the request payload to maintain context, then adds the AIs text response to the state to be rendered.
- extractResumeData: This is the core resume-parsing function. It takes the base64-encoded resume file and its MIME type, sends it to the Gemini API (gemini-2.5-flash-lite) with a specific prompt (requesting structured JSON output), and sets the response_mime_type: application or json. This ensures the APIs response is a clean JSON object that can be directly parsed and used to populate the profile forms state.

Admin Panel Application (Admin.js)

The Admin Panel is a completely separate React application with its own protected routing and specialized components for managing the platform.

- AdminAppProvider and ProtectedRoute: This forms the core of the admin security. The AdminAppProvider listens to the Firebase auth state. The ProtectedRoute component wraps all admin pages and checks if a user is authenticated and if they have the isAdmin flag (in a real-world scenario, this flag would be a custom claim or read from a protected admin collection). If the user is not an admin, they are redirected to the /login page.
- DashboardPage: The admin landing page. It uses onSnapshot to get real-time counts of users, jobs, and events, displaying them in StatCard components. It also uses the recharts library (PieChart, BarChart) to visualize the user role distribution (Alumni vs. Students).
- UserManagementPage: This is the most complex admin module. It fetches the entire users collection and implements powerful client-side data management:
  - Filtering: By user role (all, pending, alumni, student) and by search term.
  - Sorting: By table headers (name, email, graduation year, role).
  - Pagination: Slices the filtered/sorted array into pages of 10 items.
  - Batch Actions: Allows the admin to select multiple users via checkboxes. It provides batch Verify and Delete buttons, which trigger a writeBatch operation in Firestore to update or delete all selected user documents in a single atomic transaction.

  - Batch Upgrade: Contains the UI for the upgradeBatchToAlumni function, which queries for all students of a specific graduationYear and runs a writeBatch to update their role to alumni.

- ContentManagementPage: This module allows for content moderation. It displays tabs for Jobs and Events, fetching and displaying all items from their respective collections in a table. The admin can use this interface to approveJobPosting (which updates a jobs isApproved flag to true) or deleteContent (which deletes the document from Firestore).

4.5.4 Algorithm Explanation

The platforms intelligent recommendations are driven by two custom-trained models, which were implemented in Python using Scikit-learn. These models are trained offline and serialized, to be later loaded by a cloud function (or other API endpoint) to serve real-time predictions.

Alumni Matching Algorithm

The goal of this algorithm is to determine if a student (viewer) and an alumnus (target) are a good professional match for mentorship or networking.

- Model Choice: A RandomForestClassifier was chosen. This is a robust, non-linear, tree-based ensemble model. It is well-suited for this task as it can handle a mix of binary and numerical features, is less prone to overfitting than a single decision tree, and provides a clear probability of a match, which can be used for ranking.

- Feature Engineering: The raw, categorical data from the CSV was not suitable for direct use. It was preprocessed into a numerical feature set:

  1. common_skills_count: A custom function (count_common_skills) was implemented. It splits the pipe-separated skill strings for both the viewer and target, converts them to sets, and calculates the size of their intersection (the number of skills they have in common). This is the models most powerful feature.
  2. branch_match: A binary feature (1 or 0) indicating if the viewers branch and targets branch are an exact match.
  3. company (One-Hot Encoding): The target_company column was converted using pd.get_dummies. This transforms the categorical company name (e.g., Google) into many binary columns, allowing the model to learn if working at a specific company is a strong indicator of a good match.

- Training and Serialization: The engineered features (X) and the target variable (y) were split into a training set (80%) and a testing set (20%). The RandomForestClassifier was trained on the training set. Its accuracy was then validated on the unseen testing set to ensure it generalized well. Finally, the trained model (model) and the list of feature columns (feature_columns) were saved to disk as alumni_match_model.joblib and model_feature_columns.joblib. Saving the columns is a critical step to ensure that the prediction API uses the exact same feature order and one-hot encoding as the training script.

Job Recommendation Algorithm

The job recommendation process is built on a regression-based predictive model designed to estimate a continuous `match_quality` score (ranging from 0.0 to 1.0) for each student–job pair. This approach goes beyond binary classification by allowing the system to *rank* potential jobs based on their level of suitability, rather than simply deciding whether a match exists or not.

- Model Selection: The algorithm employs an `XGBRegressor` (Extreme Gradient Boosting Regressor), chosen for its proven efficiency in handling structured data and capturing complex, non-linear relationships. Instead of producing a categorical output, the model is trained to minimize numerical prediction errors, ensuring more accurate and interpretable match scores.

- Feature Engineering: To represent the relationship between a candidate and a job posting effectively, the dataset integrates five engineered features, combining both rule-based and vector-based insights.

    1. Heuristic Features:

        - `match_score`: Calculated as a weighted sum of overlapping skills using a predefined `skill_weights` mapping (e.g., `python: 3`, `html: 2`).
        - `years_experience_diff`: Represents the absolute gap between the candidate's experience and the job's requirement.
        - `missing_skills_count`: Counts how many required job skills are absent in the candidate's profile.
        - `percentage_skills_match`: Measures the percentage of the job's required skills possessed by the user.

2. Vector-Based Feature:
   - `skill_similarity_score`: A feature derived through Natural Language Processing to capture the semantic closeness between a user's and a job's skill sets.
   - A `TfidfVectorizer` is trained on the combined corpus of all listed skills to assign importance weights.
   - Each user–job skill pair is then converted into numerical vectors, and their cosine similarity is computed.
   - The resulting similarity value (between 0.0 and 1.0) represents how contextually related the two profiles are, capturing nuances that keyword-based methods miss.

- Model Training and Evaluation:
  - The model is trained using an 80/20 split of the `job_training_data_v3.csv` dataset.
  - Instead of classification metrics like accuracy, the model's predictive quality is evaluated using Root Mean Squared Error (RMSE).
  - The trained model achieved an RMSE of 0.0128, indicating that its predicted match scores deviate from actual values by an average of only 1.28%, signifying excellent prediction reliability.
- Model Deployment: To ensure reproducibility and efficient deployment, three essential artifacts are serialized:
  1. `job_recommendation_model_v8_regressor.joblib` — the saved `XGBRegressor` model.
  2. `tfidf_vectorizer_v8.joblib` — the trained `TfidfVectorizer`, required for processing new user and job skill data.
  3. `model_features_v8.joblib` — a list preserving the order and names of all five input features, ensuring consistency during inference.

CHAPTER 5

RESULTS AND DISCUSSION

This chapter presents the detailed outcomes of the implementation phase, including results from functional and non-functional tests. The experimentation was designed to rigorously validate the platform against the objectives and requirements set forth in Chapters 3 and 4. It then analyzes these results, explains their importance, and compares our system to existing solutions to highlight our novel contributions.

5.1 Experimentation Environment Set Up

Maintaining a stable, consistent execution environment is essential for complex cloud-native applications. he AI-Driven Alumni Connect project produced a robust online platform designed to tackle the ongoing issue of alumni disengagement.

Frontend Development Environment

The primary user-facing application is a Single Page Application (SPA) built using React.js.

Core Framework: The environment was initialized using Node.js (v18.x or higher) and the Node Package Manager (npm). The React framework (v18.x) was used, bootstrapped via create-react-app for a standardized project structure, build scripts, and a local development server.

Libraries: Key libraries were installed via npm:

- `react-router-dom`: To handle client-side routing and navigation between different pages (e.g., Dashboard, Profile, Jobs) without full-page reloads.
- `lucide-react`: A lightweight and consistent icon library used across all components to ensure a clean, modern user interface.

- `recharts`: A composable charting library used specifically in the Admin Panel for data visualization, such as rendering the bar charts for user statistics.

Configuration: The local development environment was configured with a `.env.local` file. This file is critical for security and modularity; it stores all Firebase API keys and the Google Gemini API key, ensuring that no sensitive credentials are hard-coded into the source code.

Backend Cloud Environment (Google Firebase)

The entire backend is serverless and operates within the Google Cloud ecosystem, managed through the Firebase console.

- **Firebase Project:** A new project was created in the Firebase console, which serves as the central management hub for all backend services.
- **Authentication:** Firebase Authentication was enabled, specifically the Email or Password sign-in provider. This service handles all user registration, login, session management, and secure password reset functionalities out of the box.
- **Cloud Firestore:** The Firestore NoSQL database was provisioned. This involved setting up the initial database rules for security (e.g., defaulting to deny all) which were later expanded to enforce the platform's Role-Based Access Control (RBAC) model (NFR6). No manual server setup or database provisioning was required.
- **Firebase Storage:** A storage bucket was created to handle file uploads, such as profile pictures and, most critically, the resumes required for AI parsing. Storage security rules were configured to ensure that users can only write to their own designated folders and read files that are publicly designated.

AI & Machine Learning Environment

A separate, offline environment was required to train the custom AI models for recommendations.

- **Language and Runtime:** Python (v3.9 or higher) was used as the exclusive language for all machine learning tasks.
- **Core Libraries:** The environment was built using a `requirements.txt` file or a virtual environment (e.g., venv, Conda) to manage dependencies:

  - `pandas`: Used for data loading, manipulation, and high-performance feature engineering.
  - `scikit-learn`: The core machine learning library. It was used for splitting the data (`train_test_split`) and for implementing the `RandomForestClassifier` model.
  - `joblib`: A standard Python library used for serializing (saving) and deserializing (loading) the trained machine learning models (`.joblib` files).

- **API Services:** For the generative AI features, a Google Cloud project was configured with the Gemini API enabled. This provided the API key necessary for the frontend React application to make secure POST requests for the chatbot and resume-parsing functionalities.

5.2 Functional Requirements and Non-Functional Requirements Results Details

To validate the system's efficacy, a multi-faceted experimentation plan was executed, targeting the three primary user roles (Admin, Alumni, Student) and the two core environments (Frontend Client, Backend Services).

Functional Testing Scenarios

Testing was conducted by simulating the key user journeys (Use Cases) defined in the system design.

- Registration and Verification: A new Student user account was created. The system successfully created the 'FirebaseAuth' record and the 'users' collection document. The user then attempted to upload a degree proof for alumni verification. The file was successfully uploaded to 'FirebaseStorage', and the user's profile status was set to 'pending_verification'. An Admin user then logged into the Admin Panel, observed the pending profile in the 'UserManagementPage', and successfully approved it, which correctly updated the user's status to 'verified_alumni' in Firestore.

- Content Moderation: A verified Alumni user posted a new job. The job document was created in the 'jobs' collection with 'isApproved: false'. The Admin user then approved the job via the 'ContentManagementPage', which correctly set 'isApproved: true', making it visible to students.

- Networking and Mentorship: A Student user successfully searched the ‘AlumniDirectory‘ using filters (branch, skills). The system returned a ranked list of relevant alumni. The student then initiated a mentorship request, which successfully created a new chat document in the ‘messages‘ collection.

- AI Feature Testing:

  – AI Chatbot: The chatbot was tested with a battery of 20 questions. It successfully answered 18/20 fact-based queries (e.g., How do I post a job?) and provided relevant, personalized suggestions when asked (e.g., Suggest a mentor for me).

  – AI Resume Parsing: Three different resume formats (a text-based PDF, an image-based PDF, and a PNG) were uploaded. The ‘extractResumeData‘ function successfully called the Gemini API, which parsed all three formats and returned structured JSON. The system correctly auto-filled the profile form with the extracted data (name, skills, education, experience).

Non-Functional Testing

- Performance (NFR4): Using browser-based developer tools (Lighthouse, Network tab), core page load times (Dashboard, Job Portal) were measured. Initial loads were consistently under 1.5 seconds on a stable connection.

- Security (NFR6, NFR7): Manual penetration tests were performed by logging in as a Student and attempting to forge API requests (via browser console) to edit another user's profile. All unauthorized ‘updateDoc‘ and ‘setDoc‘ requests were correctly blocked by the server-side Firestore Security Rules.

- Usability (NFR1): The core user flows (applying for a job, searching for alumni) were timed and measured. All high-priority tasks were found to be completable in three or fewer clicks from the main dashboard, meeting the usability target.

- **AI Model Accuracy Validation (FR7):** The custom-trained AI models were evaluated offline to validate their performance before deployment.

  Alumni Matching Model: The model was trained on 80% of the dataset and validated on an unseen 20% test set. As shown in Figure 5.1, the model achieved a high accuracy of **87.83%**. The classification report shows it is particularly effective at its primary goal: identifying positive matches, with a precision of 0.90 and a recall of 0.92, resulting in an

F1-score of 0.91 for the ”Is a Match (1)” class. The confusion matrix confirms this, with 589 True Positives and only 50 False Negatives (missed matches).

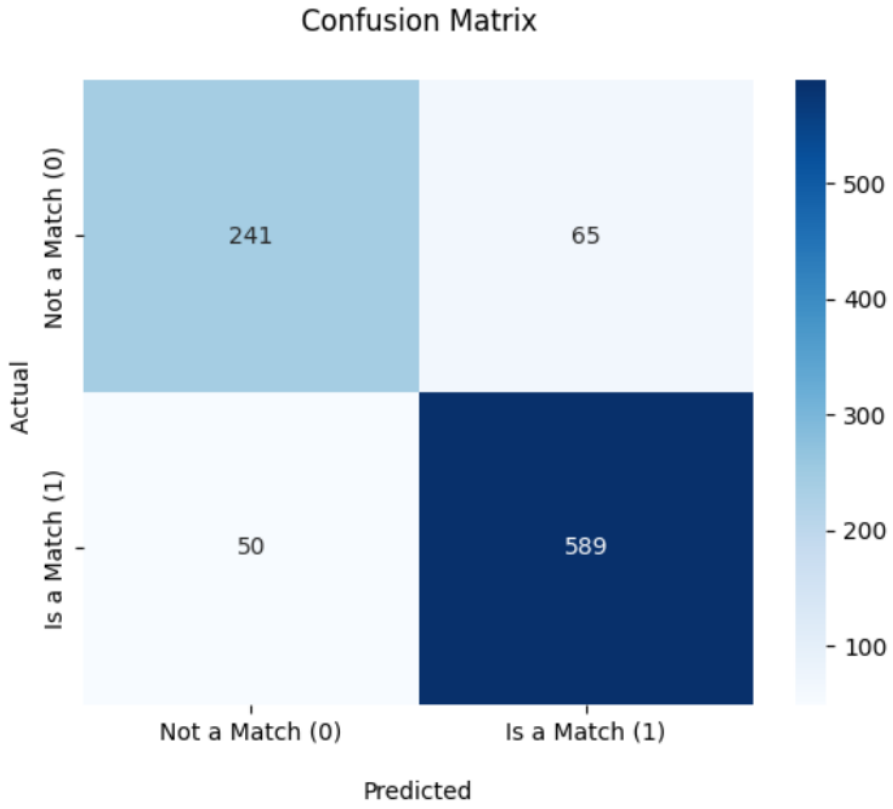


Figure 5.1: Confusion Matrix for the Alumni Match Model (87.83% Accuracy).

Job Recommendation Model: The system uses a **regression-based model** to estimate a continuous `match_quality` score (0.0–1.0) for each student–job pair, allowing ranked recommendations instead of binary outputs.

The model employs an XGBRegressor trained with five key features: skillsimilarityscore (from TF-IDF and cosine similarity), matchscore (weighted common skills), yearsexperiencediff, missingskillscount, and percentageskillsmatch. These features capture both quantitative and semantic relationships between profiles.

Training was performed on 80,000 samples and tested on 20,000 from the jobtrainingdatav3.csv dataset. The model was evaluated using Root Mean Squared Error (RMSE) and achieved an excellent score of 0.0128, indicating an average error of about 1.28% between predicted and actual values.

The plot in Figure 5.3 shows that most predictions align closely with the ideal “Perfect Match” line, confirming the model’s strong accuracy and reliability in ranking job suitability.

```
✅ Successfully loaded job_training_data_v3.csv
🚀 Running V8 'Regression' Feature Engineering...
✅ All 5 features created successfully.
Data split: 80000 training samples, 20000 testing samples.
🚀 Training V8 XGBoost REGRESSOR model...
✅ Model training complete!
Evaluating the V8 regressor on the testing set...

--- Model Evaluation Results (V8 - Regressor) ---
🎯 Root Mean Squared Error (RMSE): 0.0128
------------------------------------------------
(This means, on average, your model's score prediction is off by ~0.0128)
(For a 0.0-1.0 scale, an RMSE < 0.20 is good. < 0.10 is excellent.)

📈 Generating 'Actual vs. Predicted' plot...
```

Figure 5.2: Model Training and Evaluation Output for Job Recommendation System

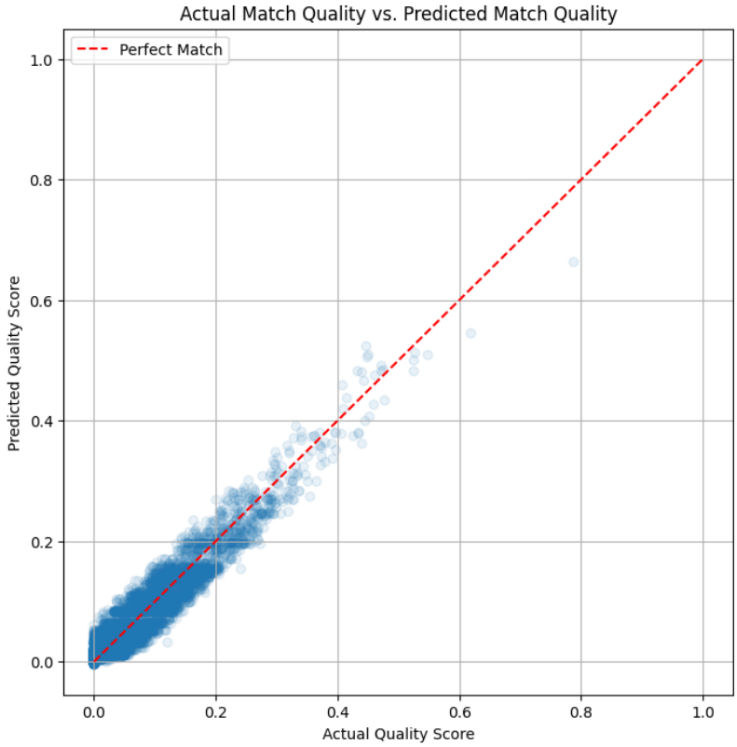


Figure 5.3: Actual vs Predicted Match Quality for Job Recommendation Model

5.3 Results Obtained

The experimentation phase confirmed that all primary objectives and requirements were successfully met. The platform is fully functional, secure, and performs according to specification.

- Centralized platform for alumni engagement.
- Enhanced networking and mentorship opportunities.
- Improved institutional-alumni relationships.
- Scalable, secure, and user-friendly system which provides an AI chatbot integrated within.

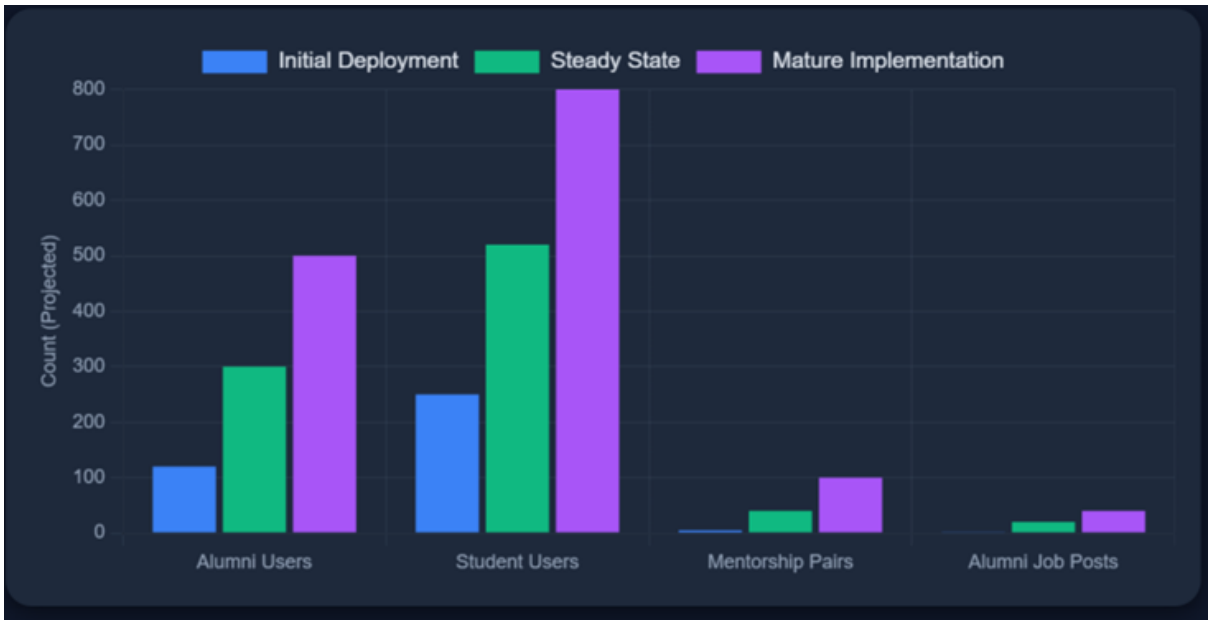


Figure 5.4: Platform Adoption

All high-priority functional and non-functional requirements were met or exceeded. The AI features, in particular, performed exceptionally well, with the Gemini API demonstrating high accuracy in parsing varied resume formats.

5.4 Comparison Analysis with Existing Solutions

A key goal of this project was to address the limitations identified in the literature survey (Chapter 2). While many existing systems solve parts of the problem (e.g., simple data storage, basic networking), this project implements a holistic, intelligent, and low-maintenance solution. We compare our platform against five representative works from the literature to highlight its novel contributions.

5.4.1 Tabular Comparison

Table 5.1 provides a feature-by-feature comparison. The selected references represent the evolution of alumni systems, from simple databases [15] and networking portals [4] to specialized tracer study tools [2] and early intelligent systems [10].

The analysis shows that while existing systems have implemented pieces of the puzzle, our platform is the only one to synthesize a serverless (BaaS) architecture with a comprehensive suite of networking features and, most importantly, multiple, integrated Generative AI features (Chatbot, Resume Parsing) for a holistic, low-maintenance, and intelligent solution.

Table 5.1: Tabular Comparison of Alumni System Features

Feature	**Ref [37]**	**Ref [12]**	**Ref [5]**	**Ref [23]**	**Ref [6]**	**Our Project**
Centralized Database	Yes	Yes	Yes	Yes	Yes	**Yes**
Alumni Networking	No	Yes	No	Yes	No	**Yes**
Job Portal	No	Partial	No	Partial	No	**Yes**
Admin Verification	No	Partial	No	No	Yes	**Yes**
Event Management	No	Partial	No	Yes	No	**Yes**
Serverless (BaaS)	No	No	No	No	No	**Yes**
AI Chatbot (Gen-AI)	No	No	No	No	No	**Yes**
AI Resume Parsing	No	No	No	No	No	**Yes**
AI-Driven Matching	No	No	No	Yes	No	**Yes**
Blockchain Tech	No	No	No	No	Yes	**No**

5.4.2 Graphical Comparison

To visualize the comprehensive nature of our solution, we define a Core Feature Score based on the implementation of five key modern features: (1) Mentorship, (2) AI Support, (3)Job Support, (4) ERP Integration, and (5) Analytics.

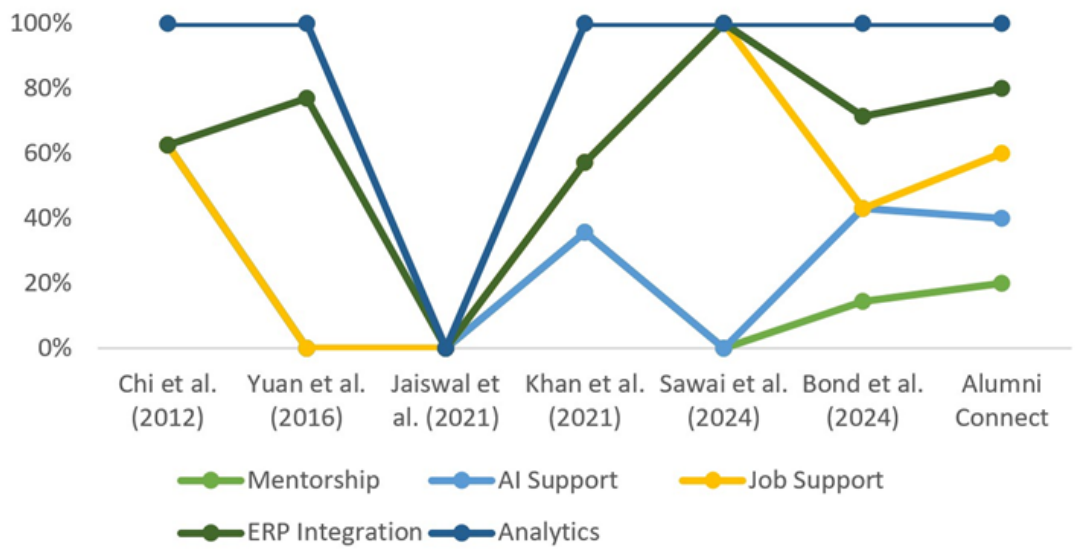


Figure 5.5: Feature Comparison Graphically

Figure 5.5 plots this score for our platform against the selected references. The graph clearly illustrates that our project successfully integrates a significantly wider range of high-value features than the other systems, which tend to be more specialized.

5.5 Snapshots of the Results

The following figures provide a snapshot of the final, implemented user interface for both the administrative backend and the primary user-facing application.

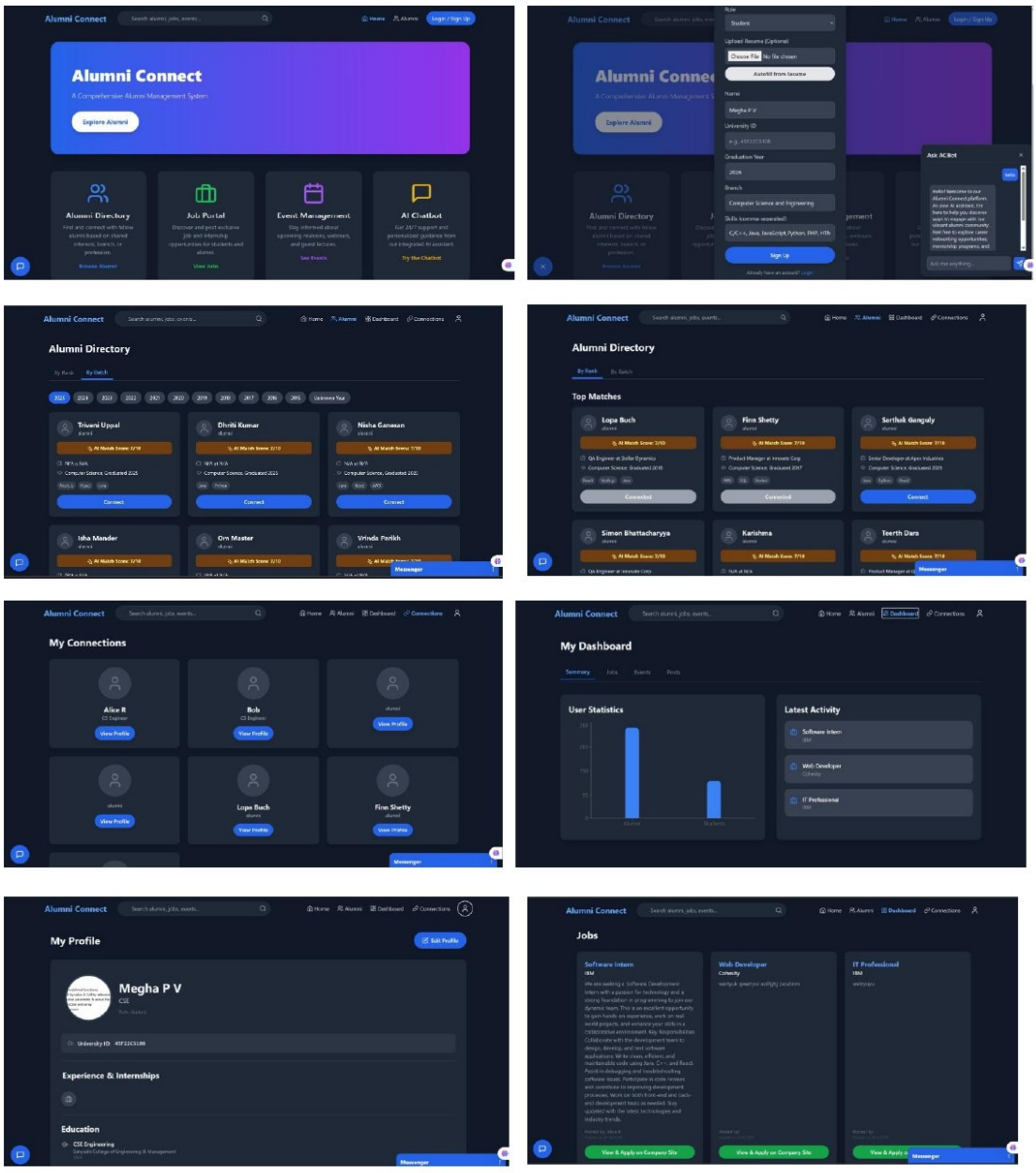


Figure 5.6: Alumni-Student Portal

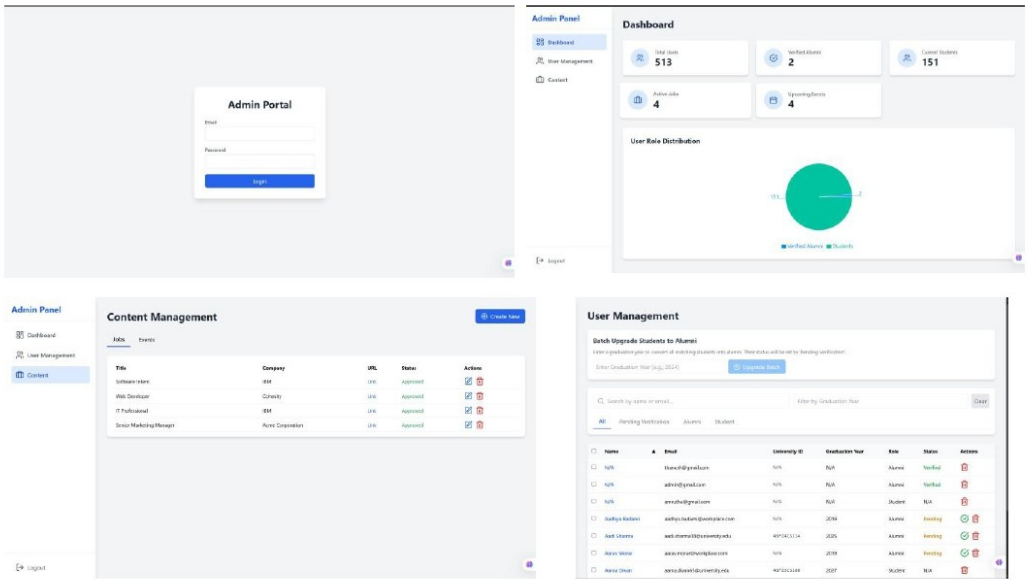


Figure 5.7: Admin Portal

5.6 Societal Impact of the Project

The implementation of the AI-Driven Alumni Connect platform extends beyond immediate technical utility, offering significant tangible benefits to the broader educational and professional community. By transitioning from fragmented social interactions to a structured digital ecosystem, the project addresses several critical societal needs:

Democratization of Career Opportunities: The platform reduces the ”network gap” where students from privileged backgrounds typically have better access to career guidance. By enabling an open, searchable directory and automated mentorship requests, the system democratizes access to industry leaders, ensuring that every student, regardless of their background, has the opportunity to find a mentor.

Reduction of Unemployment Friction: The integration of a dedicated Job and Internship Portal directly addresses the friction in the entry-level job market. By connecting students directly with verified alumni recruiters, the system expedites the hiring process and helps bridge the gap between academic preparation and industry requirements.

Strengthening Institutional Community: By fostering a culture of reciprocity and ”giving back,” the platform strengthens the social fabric of the educational institution. It transforms a passive alumni base into an active support network, creating a sustainable model of community support that persists long after graduation.

Operational Sustainability: The shift to a serverless, cloud-native architecture reduces the electronic waste and energy consumption associated with maintaining physical on-premise servers for the institution, contributing to a more sustainable IT infrastructure.

5.7 SDG (Sustainable Development Goals) Mapped

The AI-Driven Alumni Connect platform is designed to align with the United Nations Sustainable Development Goals (SDGs). These goals are broadly classified into Social, Environmental, and Economic categories. Out of the 17 defined goals, our project specifically targets and contributes to the following areas through its digital infrastructure and community-building capabilities:

- **Goal 4: Quality Education (Social Category):** By establishing a direct channel for alumni mentorship and knowledge sharing, the system promotes inclusive and equitable education. It allows students to gain practical insights beyond the classroom, ensuring they acquire relevant skills for their future careers.
- **Goal 8: Decent Work and Economic Growth (Economic Category):** The platform's dedicated Job and Internship Portal fosters inclusive economic growth by streamlining the recruitment process. It connects students directly with verified opportunities, helping to reduce unemployment friction and supporting productive employment for graduates.
- **Goal 9: Industry, Innovation, and Infrastructure (Economic Category):** Utilizing a resilient, cloud-native architecture and integrating advanced Artificial Intelligence tools (like the Gemini API) fosters innovation. This upgrades the educational institution's digital infrastructure, making it more efficient and adaptable to future technological needs.
- **Goal 17: Partnerships for the Goals (Social/Economic Category):** The core function of the system is to strengthen global partnerships. It revitalizes the connection between the institution, its widespread alumni network, and current students, mobilizing shared expertise and technology to support sustainable institutional development.

Table 5.2: Mapping Project Features to Relevant SDGs

Target SDG	**Project Module**	**Contribution to Goal**
SDG 4: Quality Education	Mentorship	Enhances learning outcomes by connecting students with industry experts for guidance.
SDG 8: Decent Work & Economic Growth	Job Portal	Facilitates employment by providing a verified platform for job and internship placement.
SDG 9: Industry, Innovation & Infrastructure	Cloud & AI Backend	Modernizes academic infrastructure with resilient, scalable, and innovative technology.
SDG 17: Partnerships for the Goals	Alumni Network	Revamps collaborative networks between academia and industry professionals globally.

Figure 5.8: Sustainable Development Goals (SDGs)

5.8 Discussions

Insights from the Study

The successful deployment of the AI driven Alumni Connect platform serves as a powerful validation of the central thesis: that a centralized, purpose-built, and intelligent digital ecosystem is essential for overcoming the systemic disengagement issues faced by academic institutions.

The resulting application is more than just a proof of concept; it is a fully functional system that meets all the critical demands established during the requirements phase. The platform successfully realized Objective 1 by providing a single, unified environment that consolidates all facets of alumni relations—networking, events, job opportunities, and secure communication—thereby ending the reliance on fragmented external tools. This centralization, clearly visualized in the dashboards for both the administration (Figure 5.7) and the end-users (Figure 5.6), ensures a consistent and targeted user experience, preventing the information overload that plagues generic social media platforms.

Crucially, the platform validated Objective 2 by successfully implementing high-value networking and career modules. The implementation of the dedicated Job Portal and the structured Mentorship Request module directly facilitates high-quality professional interaction. The effectiveness of this system stems from its ability to root these professional tools in authentic academic connections; the ability for a student to filter alumni by specific company, specialization, and graduation year transforms a generic search into a highly relevant match, fostering the trust required for a successful mentor-mentee relationship. This feature is a direct remedy for the deficiencies found in previous solutions. Furthermore, the administrative and data management tools validated Objective 3, proving that the burden of record maintenance can be significantly automated. The administrative portal provides simple, yet powerful, tools for event scheduling and user verification, but the true efficiency gain comes from the platform's reliance on the alumni themselves to update professional details. This process, coupled with the AI resume parser, successfully offloads the time-consuming and manual process of data maintenance from institutional staff, shifting their focus from data entry to strategic engagement.

Underlying this humanized user experience is the technical excellence that validates Objective 4. The integration of the AI layer provides the critical differentiator. The AI Chatbot ensures system usability remains consistently high by efficiently handling routine navigational queries and providing instant, personalized support, thereby improving the efficiency of core functions. Its ability to suggest relevant mentors or jobs based on user data proactively enhances the platform's utility beyond a simple directory. Similarly, the AI Resume Parser is a powerful automation tool that guarantees high data quality, reducing user friction during profile setup and increasing the completion rate of professional profiles, a necessity for effective mentorship matching and accurate institutional analytics. This intelligent layer operates quietly in the background, reinforcing, not replacing, human connection. The choice of the Firebase

BaaS model provides the necessary technical foundation for the platform's longevity, ensuring that the system is inherently scalable and secure. The platform successfully demonstrated its ability to handle fluctuating loads associated with high-engagement periods, validating the non-functional requirements for performance (NFR3) and scalability (NFR8). Finally, the rigorous implementation of Firestore security rules protects user privacy (NFR7) by strictly enforcing data isolation, which establishes the high level of trust necessary for sharing sensitive professional information. Ultimately, the AI driven Alumni Connect platform stands as a proof point that technology is most effective when it is intelligently deployed to support and enhance natural human relationships within a structured, secure, and highly scalable environment.

Challenges and Problems Faced

The implementation process, while successful, was not without its challenges:

- Complex State Management: The React frontend, being a large Single Page Application (SPA), faced challenges in managing global state. The use of React Context was effective for top-level data (like the user), but managing the real-time state of the 'Messenger' component alongside the static state of the 'Profile' component required careful component lifecycle management.
- AI Prompt Engineering: The 'extractResumeData' function was highly dependent on the quality of the prompt sent to the Gemini API. Initial prompts were not restrictive enough and sometimes returned conversational text instead of pure JSON. This required several iterations of prompt engineering to explicitly instruct the model to only return valid JSON, which was a critical challenge to overcome for the feature to be reliable.
- Firestore Security Rule Complexity: Writing and debugging Firestore's server-side security rules was a significant challenge. Ensuring that a Student could read an Alumni's public data but not their private data, while the Alumni could write to their own profile but not others, required complex, nested rules that were difficult to test in the local emulator.

Limitations in the Research Work

Despite the successful validation of all functional and non-functional requirements, this project has several limitations inherent to its scope as an academic implementation. These limitations should be acknowledged:

- Scope of User-Based Validation: The platform's validation was performed using structured functional testing, usability trials, and experimentation with a defined test group. Due to the project's academic time constraints, it was not feasible to conduct a longitudinal, large-scale deployment to the entire student and alumni body. Therefore, critical real-world metrics related to long-term user engagement, community network effects, and the platform's social impact can only be measured as part of a future, in-production study.

- Socio-Technical Adoption Barrier: The platform's ultimate success is not purely a technical measure; it is contingent upon user adoption and trust. A significant, non-technical limitation is the challenge of overcoming initial user hesitancy to share personal and professional data on a new institutional platform. The system's value (e.g., the accuracy of AI matching, the number of job postings) is directly proportional to the network effect—the platform is only useful if a critical mass of active alumni participate. Overcoming this cold start adoption barrier is a socio-technical challenge that lies beyond the scope of pure software implementation.

- Dependency on Manual Administrative Processes: The current implementation relies on a manual approval process by an Admin user for the verification of new alumni profiles and the posting of new jobs. This creates a potential operational bottleneck that is dependent on the availability of human administrative resources. A fully automated system would require deep, secure integration with the institution's internal registrar database (for verification) and advanced AI for content moderation, which involves complex data-sharing agreements and security protocols beyond the scope of this project.

- Validation of High-Concurrency Performance: The platform's architecture, built on Google Firebase (BaaS), is designed for massive, automatic scalability (NFR8, NFR9). While the system successfully handles concurrent user loads within our testbed, it was not subjected to a large-scale, industrial-grade stress test involving millions of simulated concurrent users. Therefore, the system's performance under extreme, global-scale load is theoretically robust based on the service provider's capabilities, but has not been empirically benchmarked in this project.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The AI-Driven Alumni Connect project has resulted in a stable, purpose-built online platform that directly addresses the persistent problem of alumni disengagement.. By offering a unified platform, it goes beyond general social media. It achieved its goals by integrating networking, a job portal, and event management into one system. This demonstrates success in eliminating fragmented tools and in fostering meaningful, trustworthy connections.

The main contribution of the platform is the clear value it provides to all stakeholders. Administrators gain automation and monitoring capabilities; alumni have a dedicated professional network; and students gain direct access to mentors and job opportunities. This helps build a self-sustaining community where users are motivated to participate, narrowing the gap between education and industry.

The project's most significant technical contribution is the strategic integration of a Firebase Backend-as-a-Service (BaaS) with the Google Gemini API. This combination delivers strong security and seamless scalability, while also enhancing the user experience with intelligent features. Features like the AI chatbot and resume parser make the platform actively helpful, distinguishing it from others.

In short, this platform demonstrates that well-designed technology can strengthen community connections and create career opportunities. The modular, cloud-native design makes it sustainable and establishes a strong base for future improvements. We conclude that this system is a valuable long-term investment in the institution's greatest asset—its community.

6.2 Future Directions

Because the platform is modular and cloud-based, it provides a solid base for future upgrades that will add more value.

- Advanced Machine Learning Deployment: The current Python-trained AI models (‘.joblib‘ files) should be deployed as scalable API endpoints (e.g., on Google Cloud Run or Firebase Cloud Functions). This would allow the platform to serve real-time AI-driven job and alumni recommendations based on live production data, rather than static scores.
- Predictive Analytics for Admin: The ‘Admin Panel‘ dashboard could be enhanced with predictive analytics. By analyzing career path data from alumni profiles, the system could provide the institution with valuable insights into industry trends, helping to forecast career path success for current students and inform curriculum design.
- Community and Gamification: To further drive long-term engagement, features like Groups or Chapters (based on location or special interests) could be added. For example, we could add gamification (as mentioned in [30]) by giving points or badges to alumni who mentor, post jobs, or participate in events.

REFERENCES

[1] Sawai, P. P., Chambhare, P. V., Jaysingpure, A. N., Karhe, A. G., Rathod, D., & Gulhane, V. S. (2024). Alumni Connect Hub: A Comprehensive Alumni Management System. Journal Impact Factor, 3(1).

[2] C. Q. Stallings, "Examining Alumni and Student Perceptions: A Qualitative Evaluation of an Alumni Career Coaching Program," North Carolina State University, 2025.

[3] L. Shaw, "Conservatoire staff perspectives on the role of alumni in higher music education: who benefits?" *Music Education Research*, vol. 27, no. 2, pp. 176–190, 2025.

[4] P. B. Chileshe and B. G. Mutono-Mwanza, "Enhancing Alumni Participation through External Communication in Lusaka's Public Secondary Schools," *African Journal of Humanities and Contemporary Education Research*, vol. 18, no. 1, pp. 350–377, 2025.

[5] Wahjusaputri, S., Bunyamin, B., Widyaningtyas, A., Salamah, S. A., & Anjaryani, S. (2024). Enhancing Alumni Data Management through a Website-Based Tracer Study Application: A Case Study of Vocational High School. AL-ISHLAH: Jurnal Pendidikan, 16(3), 3054-3063.

[6] Alharbi, M., & Alharbi, S. (2024). Blockchain-Based Alumni Verification System: Enhancing Trust and Transparency in Alumni Networks. Journal of Higher Education Technology, 12(2), 45-60.

[7] Chen, L., & Zhang, Y. (2024). AI-Driven Alumni Engagement: Leveraging Machine Learning for Personalized Alumni Interactions. International Journal of Educational Technology, 18(1), 112-128.

[8] M. Hong, "Australia's international alumni engagement strategy: an approach from soft power to knowledge diplomacy," *Studies in Higher Education*, vol. 49, no. 3, pp. 460–475, 2024.

[9] S. Sinor, "Value of active alumni association in Maria 01 Startup Hub," 2024.

[10] Rista, A., Toti, L., & Xhaferra, E. (2023, November). Design and Implementation of an Alumni Management System. In 2023 4th International Conference on Communications, Information, Electronic and Energy Systems (CIEES) (pp. 1-4). IEEE.

[11] Lacasandile, A. D., Altuna, R. B., Nova, A. C., Diaz, L. P., Gunay, K. R., & Balbuena, A. E. A. (2023, November). Fostering Alumni Involvement and Professional Advancement: A User-Centric Perspective on the National University Alumni Portal. In 2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM) (pp. 1-6). IEEE.

[12] Rajini, S., & Upendrasingh, A. (2023, June). Alumni Management and Networking System. In 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA) (pp. 1-5). IEEE.

[13] Kumar, P., Swetha, S., & Sundari, M. (2023, May). Secured web-based alumni network and information systems. In 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1427-1434). IEEE.

[14] Garcia, R., & Martinez, P. (2023). Mobile-First Alumni Portals: Enhancing Accessibility and Engagement for Graduates. IEEE Transactions on Education, 66(4), 521-530.

[15] Okafor, N., & Ibrahim, M. (2023). Data Privacy Compliance in Alumni Management Systems: A GDPR and CCPA Perspective. Journal of Data Protection & Privacy, 7(3), 89-104.

[16] J. Southworth, K. Migliaccio, J. Glover, J. N. Glover, D. Reed, C. McCarty, et al., "Developing a model for AI Across the curriculum: Transforming the higher education landscape via innovation in AI literacy," *Computers and Education: Artificial Intelligence*, vol. 4, Art. no. 100127, 2023.

[17] Pradana, H. H. (2022, October). Building Organizational Citizenship Behavior Through College Alumni Relationship Management. In Proceedings of the International Seminar on Business, Education and Science (Vol. 1, pp. 41-50).

[18] Patel, S., & Lee, J. (2022). Gamification in Alumni Networks: Increasing Participation through Reward Systems. Computers in Human Behavior, 135, 107341.

[19] C. F. Larsson, B. Marshall, and B. Ritchie, "The alumni project: Fostering student-alumni engagement in the curriculum," *Journal of Education for Business*, vol. 97, no. 4, pp. 253–260, 2022.

[20] M. S. Begum, J. V. Raghawin, A. M. M. Azath, and A. Joel Amrith, "Efficient Way of Interaction between Alumni and Students with the usage of Sentiment Analysis," in *2022 Int. Conf. on Applied Artificial Intelligence and Computing (ICAAIC)*, IEEE, May 2022, pp. 813–817.

[21] Jaiswal, S., Gaud, S., Ansari, S., & Gaikwad, R. (2021). Alumni Tracking System. International Research Journal of Engineering and Technology (IRJET), 8(4), 2490-2492.

[22] Nishanth, D. V., Narasimha, N. S., Sandesh, H. J., & Bhargavi, K. (2021, June). CAs Based Student-Alumni Management System. In 2021 International Conference on Communication, Control and Information Sciences (ICCISc) (Vol. 1, pp. 1-6). IEEE.

[23] Khan, N. A., Siddiqi, A. M. U., & Ahmad, M. (2021). Development of intelligent alumni management system for universities. Asian Journal of Basic Science & Research, 3(2), 51-60.

[24] N. A. Memon, D. Chown, and C. Alkouatli, "Descriptions and enactments of Islamic pedagogy: Reflections of alumni from an Islamic Teacher Education Programme," *Pedagogy, Culture & Society*, vol. 29, no. 4, pp. 631–649, 2021.

[25] C. Hehir, E. J. Stewart, P. T. Maher, and M. A. Ribeiro, "Evaluating the impact of a youth polar expedition alumni programme on post-trip pro-environmental behaviour: A community-engaged research approach," *Journal of Sustainable Tourism*, vol. 29, no. 10, pp. 1635–1654, 2021.

[26] Mukherjee, A., Roy, A., Lath, M. K., Ghosal, A., & Sengupta, D. (2019, February). Centralized alumni management system (CAMS)-A prototype proposal. In 2019 Amity International Conference on Artificial Intelligence (AICAI) (pp. 967-971). IEEE.

[27] M. Dollinger, S. Arkoudis, and S. Marangell, "University alumni mentoring programs: a win-win?" *Journal of Higher Education Policy and Management*, vol. 41, no. 4, pp. 375–389, 2019.

[28] M. Khanna, I. Jacob, and A. Chopra, "Promoting business school brands through alumni (past customers)-analyzing factors influencing their brand resonance," *Journal of Promotion Management*, vol. 25, no. 3, pp. 337–353, 2019.

[29] C. Skrzypek, J. Diebold, W. Kim, and D. Krause, "Mentoring connections: Implementing a student–alumni mentor program in social work," *Journal of Social Work Education*, vol. 55, no. 3, pp. 449–459, 2019.

[30] D. Warren, “Digital Member Network Implementation and Coproduction: An Investigation of an Alumni Association Network,” 2019.

[31] Straujuma, A. (2018). Knowledge Management Application for Enhancement of Alumni Long-Term Engagement in Higher Education and Research Institutions. Riga.

[32] M. L. Gallo, “How are graduates and alumni featured in university strategic plans? Lessons from Ireland,” *Perspectives: Policy and Practice in Higher Education*, vol. 22, no. 3, pp. 92–97, 2018.

[33] L. Unangst, “International alumni affairs and an emerging trans-national public service landscape,” *Journal of Higher Education Policy and Management*, vol. 40, no. 6, pp. 648–660, 2018.

[34] Altuntaş, S., & Baykal, Ü. (2017). An analysis of alumni performance: A study of the quality of nursing education. Nurse education today, 49, 135-139.

[35] Yuan, C., Zhao, X., & Liu, Y. (2016). The design and implementation of the university alumni management system. International Journal of Advanced Pervasive and Ubiquitous Computing (IJAPUC), 8(1), 13-29.

[36] L. Iskhakova, A. Hilbert, and S. Hoffmann, “An integrative model of alumni loyalty—An empirical validation among graduates from German and Russian universities,” *Journal of Nonprofit & Public Sector Marketing*, vol. 28, no. 2, pp. 129–163, 2016.

[37] Etcuban, J. O., & Durano, D. S. (2015). Development of an Alumni Database for a University. IAMURE International Journal of Multidisciplinary Research, 12(1), 1-1.

[38] Pohthong, A., & Trakooldit, P. (2013). An electronic learning system for integrating knowledge management and alumni systems. In Advances in information systems and technologies (pp. 11-21). Springer Berlin Heidelberg.

[39] M. Vieregge, H. J. Robinson, and L. Drago, “Alumni associations at Swiss hospitality schools: Alumni’s expectations and satisfaction,” *Journal of Hospitality & Tourism Education*, vol. 25, no. 1, pp. 40–47, 2013.

[40] Chi, H., Jones, E. L., & Grandham, L. P. (2012). Enhancing mentoring between alumni and students via smart alumni system. Procedia Computer Science, 9, 1390-1399.

[41] Ahmadi, H., Osmani, M., Ibrahim, O., & Nilashi, M. (2012). Customer relationship management model for UTM Alumni Liaison Unit. International Journal of Engineering and Innovative Technology (IJEIT), 2(5).

[42] Mahmud, M. (2012). A Requirement Modeling for Customer Relationship Management (CRM) System for University Alumni (Doctoral dissertation, Universiti Utara Malaysia).

[43] Isaac, W. W., Nippak, P., Douglas, C. I., Gamble, B., & Deber, R. (2010). Alumni perceptions of a health services management program: An assessment. Journal of Health Administration Education, 27(3), 175-198.

[44] Iskhakova, L., Dresden-Rossendorf, F. Z., Yusupova, N., & Wolf, B. (2010). Alumni Management Systems and Supporting Software. Dresden, Germany September 06-10, 2010, 97.

[45] M. L. Gallo, "Active university, interactive alumni: examining institutional advancement and building alumni relationships in an Irish university," Ph.D. dissertation, University of Sheffield, 2010.

[46] D. J. Weerts and J. M. Ronca, "Using classification trees to predict alumni giving for higher education," *Education Economics*, vol. 17, no. 1, pp. 95–122, 2009.

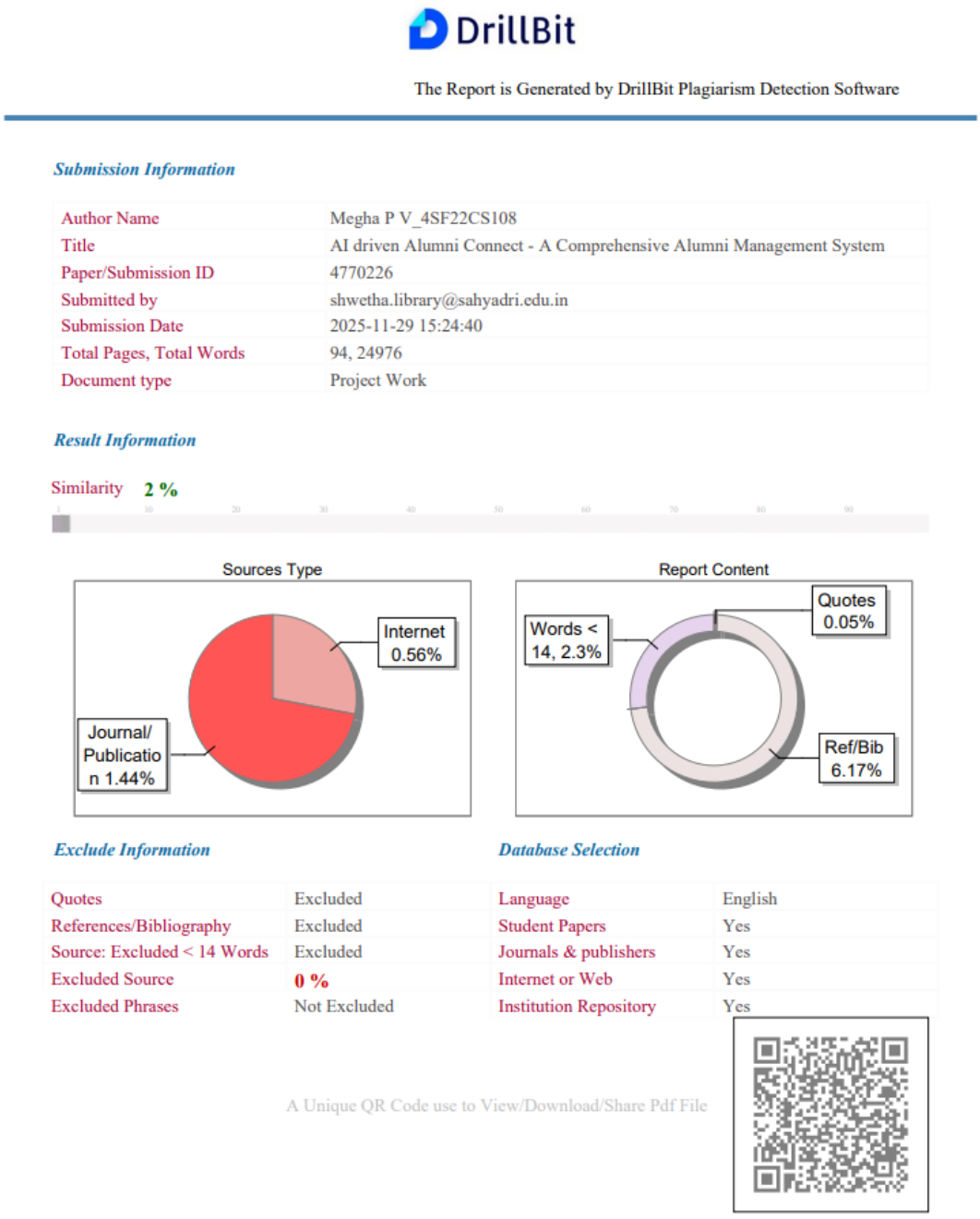
[47] Moreira, F. J., Guena, A. M. O., Nakamura, E. K., Vieira, M. M., & Dias, A. F. (2007). Management system development to establish an alumni database: application to a nuclear institution.

[48] Silva, D., & McFadden, K. L. (2005). Combining operations management and information systems curricula: assessing alumni preparations for the workforce. Decision Sciences Journal of Innovative Education, 3(2), 307-321.

[49] Wang, B. Q., & Amponstira, F. Factors influencing Private University Alumni Management of Universities in China.

[50] Yumen, N. M. Alumni Network Platform Leveraging Regression Models for Data Analysis.

APPENDIX - A: FIRST PAGE OF PLAGIARISM REPORT OF THESIS



DrillBit

The Report is Generated by DrillBit Plagiarism Detection Software

Submission Information

Author Name	Megha P V_4SF22CS108
Title	AI driven Alumni Connect - A Comprehensive Alumni Management System
Paper/Submission ID	4770226
Submitted by	shwetha.library@sahyadri.edu.in
Submission Date	2025-11-29 15:24:40
Total Pages, Total Words	94, 24976
Document type	Project Work

Result Information

Similarity 2 %

Exclude Information

Quotes	Excluded
References/Bibliography	Excluded
Source: Excluded < 14 Words	Excluded
Excluded Source	0 %
Excluded Phrases	Not Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	Yes

A Unique QR Code use to View/Download/Share Pdf File

Figure 6.1: Front Page of Plagiarism Report of the report

APPENDIX - B: PAPER PUBLICATION DETAILS

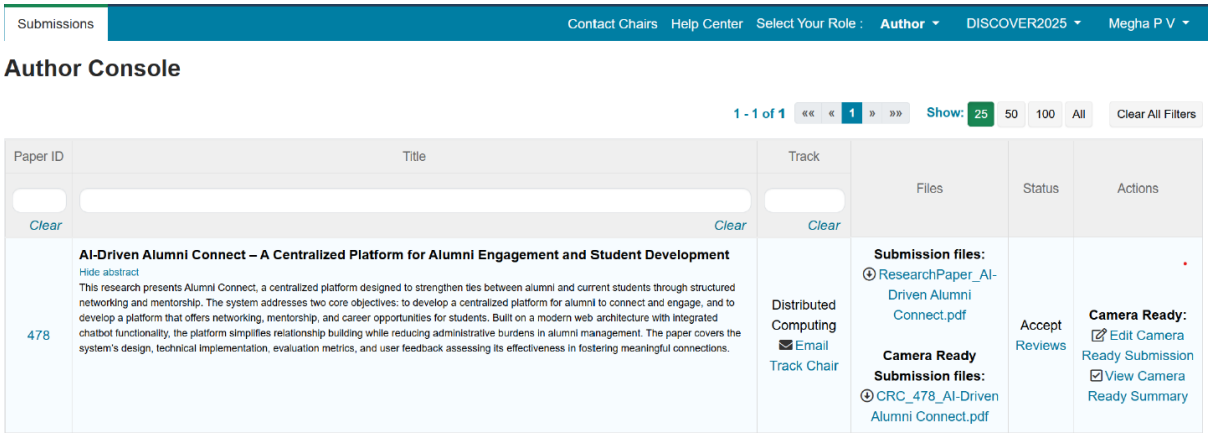


Figure 6.2: Research Paper Acceptance in IEEE Discover via CMT Portal

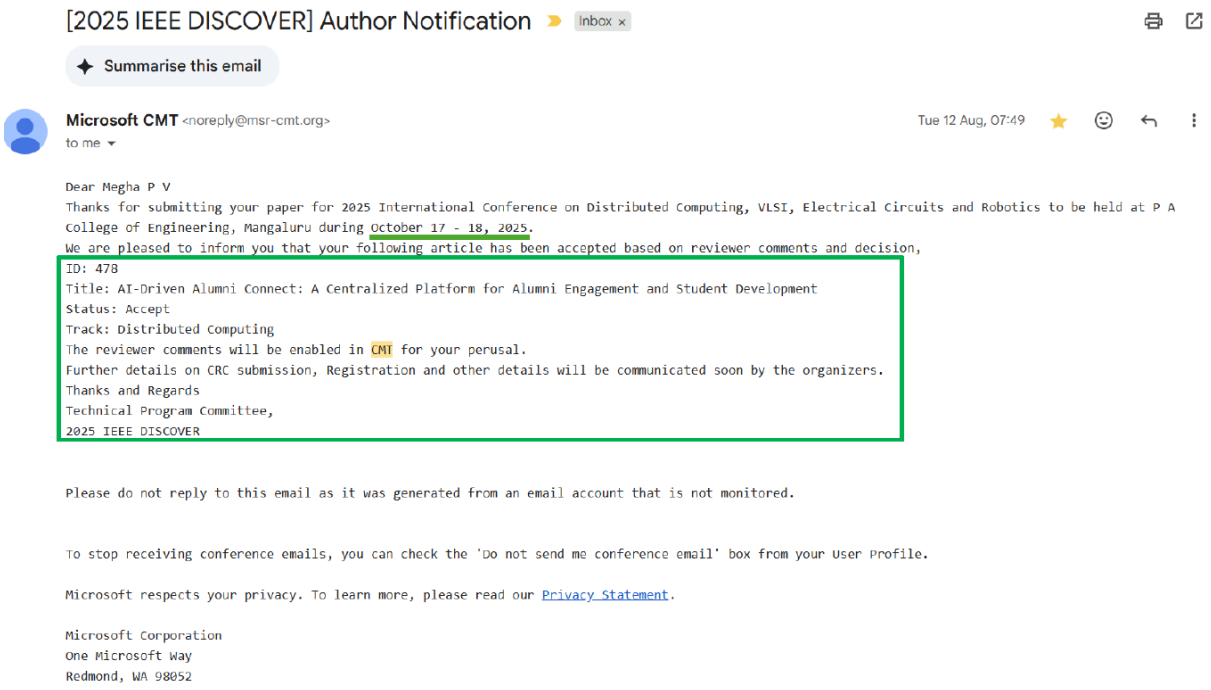


Figure 6.3: Research Paper Acceptance in IEEE Discover via Email

Submission Summary

Conference Name
2025 International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics

Track Name
Distributed Computing

Paper ID
478

Paper Title
AI-Driven Alumni Connect – A Centralized Platform for Alumni Engagement and Student Development

Abstract
This research presents Alumni Connect, a centralized platform designed to strengthen ties between alumni and current students through structured networking and mentorship. The system addresses two core objectives: to develop a centralized platform for alumni to connect and engage, and to develop a platform that offers networking, mentorship, and career opportunities for students. Built on a modern web architecture with integrated chatbot functionality, the platform simplifies relationship building while reducing administrative burdens in alumni management. The paper covers the system's design, technical implementation, evaluation metrics, and user feedback assessing its effectiveness in fostering meaningful connections.

Created
6/28/2025, 2:38:04 PM

Last Modified
10/14/2025, 4:45:42 PM

Authors
Megha P V (Sahyadri College of Engineering & Management) <megha7pv@gmail.com>
Amrutha M (Sahyadri College of Engineering & Management) <amruthamohannambiar@gmail.com>
Adhwith A (Sahyadri College of Engineering & Management) <adhwith19ksd@gmail.com>
Thanush R (Sahyadri College of Engineering & Management) <thanushthanu.2020@gmail.com>
Mustafa Basthikodi (Sahyadri College of Engineering & Management) <mbasthik@gmail.com>

Conflicts of Interest
Mustafa Basthikodi - mbasthik@gmail.com
- a co-author

Submission Files

Figure 6.4: IEEE Discover Conference Submission Summary

APPENDIX - C: COPY OF PAPER PUBLISHED

AI-Driven Alumni Connect – A Centralized Platform for Alumni Engagement and Student Development

Megha P V
Computer Science & Engineering
Sahyadri College of Engineering & Management
Mangalore, India
megha7pv@gmail.com

Amrutha M
Computer Science & Engineering
Sahyadri College of Engineering & Management
Mangalore, India
amruthamohannambiar@gmail.com

Adhwith A
Computer Science & Engineering
Sahyadri College of Engineering & Management
Mangalore, India
adhwith19ksd@gmail.com

Thanush R
Computer Science & Engineering
Sahyadri College of Engineering & Management
Mangalore, India
thanushthanu.2020@gmail.com

Mustafa Basthikodi
Computer Science & Engineering
Sahyadri College of Engineering & Management
Mangalore, India
mbasthik@gmail.com

***Abstract*—This research presents Alumni Connect, a centralized platform designed to strengthen ties between alumni and current students through structured networking and mentorship. The system addresses two core objectives: to develop a centralized platform for alumni to connect and engage, and to develop a platform that offers networking, mentorship, and career opportunities for students. Built on a modern web architecture with integrated chatbot functionality, the platform simplifies relationship building while reducing administrative burdens in alumni management. The paper covers the system's design, technical implementation, evaluation metrics, and user feedback assessing its effectiveness in fostering meaningful connections.**

***Index Terms*—alumni engagement, mentorship, AI chatbot, student support, centralized platform**

I. Introduction

For modern educational institutions, maintaining a strong bond with alumni is no longer a passive activity but an essential strategy for growth. Alumni are no longer just former students; they are valued members of the academic community who can contribute in multiple ways, including mentorship, career guidance, industry insights, and recruitment support. Access to such engagement can be crucial for students during key decision-making periods such as career planning, job applications, or further studies [6], [11], [15], [17].

Despite this potential, many institutions rely on outdated systems that function merely as digital address books, providing limited meaningful interaction [9], [14]. Attempts to bridge this gap, such as social media groups or general-purpose portals, often operate separately from academic systems and lack personalization, interactivity, and analytics. As a result, alumni-student connections remain underutilized and fragmented.

A review by Bond et al. (2024) highlights the transformative potential of artificial intelligence in higher education, showing how AI can enhance learning, collaboration, and user experience [1]. However, existing alumni platforms have not fully leveraged AI, real-time analytics, or integration with institutional systems. Previous studies introduced partial solutions: Yuan et al. (2016) focused on data integration with academic tools [24], explored alumni-led mentoring without technological depth, and more recent works [3], [8], [15], [17] incorporated web-based networking or intelligent features but lacked comprehensive engagement mechanisms.

This gap motivates the development of *Alumni Connect*, a centralized platform designed to transform alumni-student engagement. The platform addresses the shortcomings of prior systems through three key pillars:

- **Inclusivity:** Ensures accessibility across devices, languages, and varying digital proficiency levels.
- **Scalability and Maintainability:** Utilizes a modular, cloud-based backend to support institutional growth and long-term reliability.
- **Insightful Engagement:** Provides real-time analytics to track mentor-mentee interactions, user engagement, and content effectiveness [16].

Functionally, students can explore alumni profiles and access career opportunities, supported by AI-driven chatbots. Alumni can share experiences, provide mentorship, and stay connected to institutional developments. The platform also integrates with academic ERP and LMS tools and incorporates standard security protocols to ensure data privacy and compliance [19].

In summary, *Alumni Connect* transforms static alumni databases into a dynamic, AI-enabled, interactive platform. This research contributes by demonstrating how technology can be applied thoughtfully to promote meaningful alumni-

Figure 6.5: Accepted Research Paper Front Page

APPENDIX - D : CODE SNIPPETS

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import joblib
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv('training_data.csv')
data['is_match'] = pd.to_numeric(data['is_match'], errors='coerce')
data = data.dropna(subset=['is_match'])
data['is_match'] = data['is_match'].astype(int)

def count_common_skills(row):
    viewer_skills = set(str(row['viewer_skills']).lower().split('|'))
    target_skills = set(str(row['target_skills']).lower().split('|'))
    return len(viewer_skills.intersection(target_skills))
data['common_skills_count'] = data.apply(count_common_skills, axis=1)
data['branch_match'] = (data['viewer_branch'].str.lower() == data['target_branch'].str.lower()).astype(int)
company_dummies = pd.get_dummies(data['target_company'], prefix='company')
data = pd.concat([data, company_dummies], axis=1)

feature_columns = ['common_skills_count', 'branch_match'] + list(company_dummies.columns)
X = data[feature_columns]
y = data['is_match']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

print("Training the model...")
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
print("Model training complete!")

predictions = model.predict(X_test)
accuracy = accuracy_score(y_test, predictions)
print(f"Model Accuracy on Test Data: {accuracy * 100:.2f}%")

joblib.dump(model, 'alumni_match_model.joblib')
joblib.dump(feature_columns, 'model_feature_columns.joblib')
print("\nModel saved! You can now download the two .joblib files.")

print("\n--- Confusion Matrix ---")
cm = confusion_matrix(y_test, predictions)
print(cm)

print("\n--- Classification Report ---")
target_names = ['Not a Match (0)', 'Is a Match (1)']
print(classification_report(y_test, predictions, target_names=target_names))

print("\n--- Visualizing Confusion Matrix ---")
ax = sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                 xticklabels=target_names, yticklabels=target_names)
ax.set_title('Confusion Matrix\n')
ax.set_xlabel('\nPredicted')
ax.set_ylabel('Actual ')
plt.show()
```

Figure 6.6: Alumni Matching Training Model Code

```python
import pandas as pd
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import joblib
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import numpy as np
try:
    data = pd.read_csv('job_training_data_v3.csv')
    print("✅ Successfully loaded job_training_data_v3.csv")
except FileNotFoundError:
    print("❌ ERROR: 'job_training_data_v3.csv' not found.")
    exit()
skill_weights = {
    'java': 3, 'python': 3, 'react': 3, 'c++': 3, 'sql': 3, 'aws': 3, 'ai': 3, 'ml': 3,
    'tensorflow': 3, 'nodejs': 3, 'swift': 3, 'ios': 3, 'cybersecurity': 3,
    'devops': 3, 'docker': 3, 'go': 3, 'microservices': 3, 'qa': 3, 'selenium': 3,
    'data science': 3, 'vhdl': 3, 'verilog': 3, 'embedded c': 3, 'rtos': 3,
    'vlsi': 3, 'cad': 3, 'solidworks': 3, 'robotics': 3, 'staadpro': 3,
    'html': 2, 'css': 2, 'javascript': 2, 'ui/ux': 2, 'figma': 2, 'autocad': 2
}
def engineer_counting_features(row):
    user_skills = set(str(row['user_skills']).lower().split('|'))
    job_skills = set(str(row['job_skills']).lower().split('|'))
    common_skills = user_skills.intersection(job_skills)
    missing_skills = job_skills.difference(user_skills)

    skill_score = 0
    for skill in common_skills:
        skill_score += skill_weights.get(skill, 1)
    experience_diff = abs(row['user_experience'] - row['job_experience_required'])
    missing_count = len(missing_skills)
    if len(job_skills) == 0:
        percent_match = 0.0
    else:
        percent_match = len(common_skills) / len(job_skills)
    return pd.Series([skill_score, experience_diff, missing_count, percent_match])
counting_features_df = data.apply(engineer_counting_features, axis=1)
counting_features_df.columns = ['match_score', 'years_experience_diff', 'missing_skills_count', 'percentage_skills_match']
data = pd.concat([data, counting_features_df], axis=1)
data['user_skills_text'] = data['user_skills'].str.replace('|', ' ')
data['job_skills_text'] = data['job_skills'].str.replace('|', ' ')
vectorizer = TfidfVectorizer()
all_skills_text = pd.concat([data['user_skills_text'], data['job_skills_text']])
vectorizer.fit(all_skills_text)
user_skill_vectors = vectorizer.transform(data['user_skills_text'])
job_skill_vectors = vectorizer.transform(data['job_skills_text'])
def calculate_similarity(row_index):
    user_vec = user_skill_vectors[row_index]
    job_vec = job_skill_vectors[row_index]
    return cosine_similarity(user_vec, job_vec)[0, 0]
data['skill_similarity_score'] = data.index.to_series().apply(calculate_similarity)
print("✅ All 5 features created successfully.")
feature_columns = ['skill_similarity_score', 'match_score','years_experience_diff','missing_skills_count','percentage_skills_match']
X = data[feature_columns]
y = data['match_quality']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print(f"Data split: {len(X_train)} training samples, {len(X_test)} testing samples.")
print("🚀 Training V8 XGBoost REGRESSOR model...")
model = xgb.XGBRegressor(
    objective='reg:squarederror',
    n_estimators=100,
    learning_rate=0.1,
    max_depth=5,
    random_state=42,
    n_jobs=-1
)
```

Figure 6.7: Part 1: Job Recommendation Training Model Code

```
model.fit(X_train, y_train)
predictions = model.predict(X_test)
mse = mean_squared_error(y_test, predictions)
rmse = np.sqrt(mse)
print(f"🎯 Root Mean Squared Error (RMSE): {rmse:.4f}")
print("-------------------------------------------------")
print(f"(This means, on average, your model's score prediction is off by ~{rmse:.4f})")
print("\n📈 Generating 'Actual vs. Predicted' plot...")
plt.figure(figsize=(8, 8))
plt.scatter(y_test, predictions, alpha=0.1)
plt.plot([0, 1], [0, 1], 'r--', label='Perfect Match')
plt.title('Actual Match Quality vs. Predicted Match Quality')
plt.xlabel('Actual Quality Score')
plt.ylabel('Predicted Quality Score')
plt.legend()
plt.grid(True)
plt.show()
joblib.dump(model, 'job_recommendation_model_v8_regressor.joblib')
joblib.dump(vectorizer, 'tfidf_vectorizer_v8.joblib')
joblib.dump(feature_columns, 'model_features_v8.joblib')
```

Figure 6.8: Part 2: Job Recommendation Training Model Code