International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

CampusXConnect: A Unified Multi-Module Platform for Student Networking, Placement Preparation, and Skill Development

Mr. Rishabh Rawat[1], Mr. Sumiran Sharma[2], Mr. Abhishek Karan[3], Prof. Ashish Patel[4]

[1,2,3]Student, Computer Science and Engineering, Dronacharya Group of Institutions
[4]Assistant Professor, Computer Science and Engineering, Dronacharya Group of Institutions

Abstract
Technical school students use independent collection of separate platforms to manage their academic and professional activities. Students use LinkedIn to network with alumni; LeetCode to prepare for coding interviews; GitHub to host projects; and messaging applications to communicate with peers – allowing each app to work independently, without awareness of other apps. This fragmentation creates a large amount of inefficiency for students during their placement season, where they are required to frequently switch between different contexts while trying to accomplish productive work. CampusXConnect was developed to provide an integrated approach to solving the problems caused by fragmented resources. The CampusXConnect platform will integrate alumni networking, placement resource management (such as job postings), live coding environment (using Judge0), peer discussion forum, project showcase, and real-time chat features into a single authenticated workspace. The CampusXConnect platform is built on a technology stack that consists of Node.js, Express.js, React, Flutter, PostgreSQL, MongoDB, Socket.io, and AWS S3. The purpose of this paper is to outline the architectural choices made, module design documentation, security design rationale, and development rationale for the CampusXConnect platform as well as a comparative analysis of existing platforms used in the technical school student technology ecosystem.

Keywords: campus platform, alumni networking, placement preparation, live code execution, Judge0, student engagement full-stack development.

I. INTRODUCTION

The journey of undergraduate students through their academic and work life is more complex than ever. Typically, a campus placement student will have sessions open on LinkedIn (to connect with alums), LeetCode (to practice code), GitHub (to track version control) and various messaging tools (to discuss problems with peers). There is no single interface to access these tools and they are not connected to each other, so there is a lot of context switching which reduces the productivity and also impacts the ability of students to find the right practice problems, alumni help, study material and peer discussions while preparing for placement.

This is where CampusXConnect comes in. The objective of the platform was to develop a one stop shop

(or unified) for discovering and connecting with alumni; placement content; live code execution; Q&A community; real time chat; and project presentations, all with a single sign-on. The content creators (academics and administrators) will have a platform to manage the content and discussions. Though the first deployment of CampusXConnect is for the Dronacharya Group of Institutions (Greater Noida), the design of the platform (technology stack) is independent of the institution and can be deployed at any institution [1].

This paper has four sections. The second section discusses the technology stack and the choices of technologies. The third section talks about system development and the system architecture.

II. TECHNOLOGICAL LANDSCAPE

The choice of the technology was founded on the following three criteria (considerations): documented well and used by users; scalable (to handle multiple users simultaneously - e.g., multiple students viewing other students' profiles, solving problems and posting code, or participating in forum discussions); and shared across mobile and web (e.g., the same web site is used by mobile and web clients). To build the backend (server to host and serve the application), we selected Node.js as it is non-blocking, event driven (using JavaScript to define APIs and respond to events). This enables the server to process multiple client requests (multiple students viewed profiles, posted code and took part in forum threads) simultaneously. For the frontend (to build an interface to the backend), we chose Express.js because it provides a framework to design RESTful APIs (by modularly routing the APIs). We used two databases for storing data.

PostgreSQL is used for storing relational data (user, alumni, placement resources, and problems) and for semi-structured data (forum postings, chat histories, and project descriptions) we used MongoDB. MongoDB has a semi-structured schema for semi-structured data (that is more suitable for forum postings and chat histories [4]). Storing data in two databases allows the modules to choose their preferred persistence layer (based on the data model required for the data stored in the module) and to use different schemas for different data (i.e., social data and problems).

The web interface was developed using React and the mobile interface using Flutter. Both of these libraries adopt a component-based approach, so the UI can be broken down into smaller components that can be reused on different screens. WebSockets were used to enable real-time communications (such as live chat and push notifications) via Socket.io (Socket Io is a real-time, bi-directional communication protocol). A WebSocket protocol (rather than the more common HTTP polling) was used to allow messages to be delivered without the client needing to refresh the page; this is necessary for a responsive chat interface.

The coding practice module uses Judge0 (a free open-source code execution engine) which supports more than 60 languages. Judge0 accepts code submissions, which are executed in a sandboxed container and then the output (including execution time and status) is returned to the application client. So there is no need to build a compiler service.

Authentication is implemented via Firebase and push notifications are sent using Firebase and AWS S3 is used to store project media, user profile pictures and user uploads (this is used to separate file storage from the application server).

III. METHODOLOGY

The system development process is well-defined and use-case driven. Prior to development, the team

undertook a requirements analysis in order to decide the scope and functionality of each module, the inter-dependencies between modules, and how the process flows from the moment users sign-up to their account, and how they interact with their placement. This allowed distribution of tasks in the development team according to the strengths of the individual developers. [2]

A. Client-Side Components

1. **User Interface**

The feed interface is the main page users see after logging in, where they can browse community posts, interact with others, and stay updated with university announcements. Content can be reacted, commented and shared. The code interface offers a language selector, syntax highlighting code editor, and an output panel that runs code via the Judge0 engine. The profile page displays the user's skills, activity, projects and friendships. The layouts for the alumni and student profiles are slightly different to reflect their roles in the community.

2. **Client Application**

The real-time notifications module handles socket.io push notifications for new connection requests, forum replies and the addition of new placement resource files. The media storage module is for files uploaded for forum post and project showcase (with the file created on the backend, uploaded to AWS S3-n a storage cloud, and the returned URL stored in the database). The user interaction module holds all user interactions (connection requests, post upvotes, skill endorsements and project comments) and offers a reusable model of social interaction to other modules.

B. Server-Side Components

1. **Application Server**

Sessions handle all user authentication in two ways: 1) JWT or JSON web tokens for employee sessions and 2) OAuth 2.0 for Google-based single sign-on (SSO). Passwords are encrypted using bcrypt hash encryption. RBAC (Role Based Access Control) is used to manage access to features of the platform by user type (Student, Alumni, Faculty, and Admin.) and ensures that only those features required by each persona will be accessible (for example, only certain forums will be accessible to students). Each primary functionality (1. Networking, 2. Forum, 3. Coding, 4. Chat, and 5. Showcase) will be implemented as an Express router, which will have its own Service Layer

2. **Database Layer**

Structured data will be stored in a PostgreSQL database (user data, connection records, placement resource metadata and coding problems) while dynamic information (forum posts, private messaging threads and project descriptions) will be stored in a dynamic schema MongoDB database. Media files will be stored in Amazon AWS S3 buckets and will be referenced by an URL in the relational database management system (PostgreSQL database) to optimise the speed of media retrieval, and the cost of storing media files at scale.

C. External Services and Security

By utilizing Google OAuth 2.0 for improved onboarding (allowing students to log in with their existing Google account and pre-filled with general information as needed upon their initial log in), all REST API interactions as well as I/O operations such as file uploads and calls to all outside services occur over HTTPS. All chat messages will be secured in transit and all non-public student communications will be kept private. The platform's overall security model is upheld through token expiration policies and endpoint level authentication requirements.

D. Interaction Flow and System Architecture

Once a user registers, they can pick the type of user role they want and fill out their user profile with either manual entry or by authenticating with Google. Students will have access to an "alumni - student network" that allows them to filter available alumni profiles by company, the batch of graduation, or the branch of academics and to send connection requests to those alumni. Once a student selects a problem within the coding module, they write/apply their solution in the coding editor and submit it to Judge0 through the Judge0 API for evaluation; the student's solution will be returned to them and linked to their historical user profile. Alternately, when students submit projects for display in the project showcase, they will do this using a structured input process consisting of a title, description, technology stack, members of the project group, and a link to demonstrate the project.

Figure 1 provides an overview of the complete system data flow diagram for CampusXConnect, which illustrates user authentication path, two-step verification for re-entering users, account creation flow, and complete structure of the user interface module for CampusXConnect.

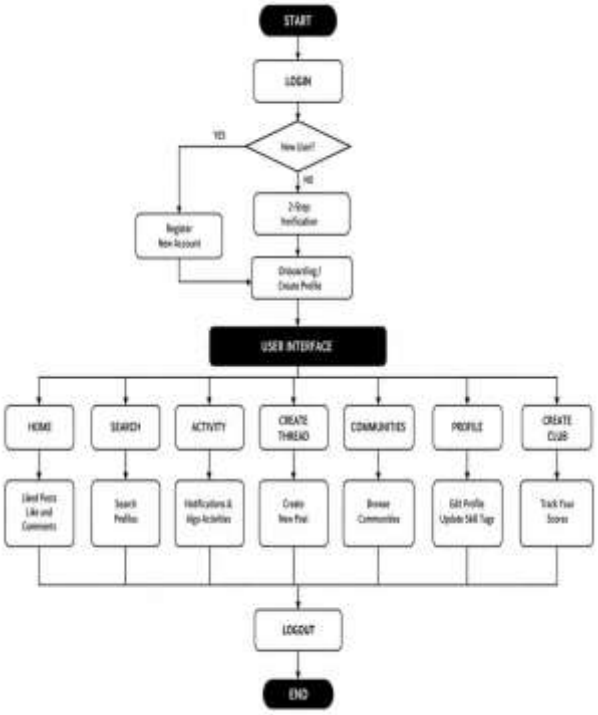

Fig. 1. System Data Flow Diagram (DFD) of CampusXConnect.

IV. RELATED WORK

Upon examining the current platforms, there appears to be no single platform that addresses the complete continuum of student's needs within an institutional context. Each of the reference platforms does very well at solving one piece of the overall problem, but none of them combine the entire workflow that a student needs from orientation through preparing for placement and engaging with alumni.

A. Professional and Community Platforms

LinkedIn is the main overall professional networking platform and offers substantial ability for alumni to discover one another once they enter the workforce. Because of its general audience, LinkedIn does not lend itself to typical academic engagement in a campus setting because it is conceptualized as a professional network and has a very formal tone [1]. LeetCode and HackerRank provide exceptional

quality, coding preparation, but there is no way to tie their usage to placement calendars, alumni networks or institutional study materials, resulting in students using these applications independently of one another, creating separate instances as students work through the setup of these tools. The platform uses a Reddit-style Q&A system where users can vote on answers, helping the most useful and high-quality responses stand out and making them easier for others to find.

In a similar way, CampusXConnect provides a voting component to its forum module, but also allows faculty to designate responses as institutionally verified, helping to establish authority for responses that are accurate, and ultimately helping to improve the quality of academic discussion and reduce the signal-to-noise ratio in academic dialogue.

B. Organization-Scoped Communication

Conversation structures within Facebook Groups and subreddit-based communities have been shown to promote user engagement whereby users will communicate more consistently when communicating within a defined set of boundaries versus being exposed to an ambiguous set of audiences [5]. Based upon this data, CampusXConnect defines the scope of discussion spaces for its users as follows: discussion areas of the CampusXConnect platform will be surrounding a specific school/academic affinity group and should result in discussions that are more relevant to the context of use by the users involved; thereby reducing the impact of random content dilution of user-generated content when communicating via Facebook Groups or subreddit-based communitie Compared to LinkedIn, CampusXConnect will allow users to have a higher degree of granularity in managing the visibility of his/her profile and related projects when using the CampusXConnect platform, beyond just being a statistic in the school.

C. Project Portfolio and Security Practices

GitHub provides examples of structured, verifiable project portfolios and how these types of project portfolios provide credibility and validation/verification for projects being built on the GitHub platform vs. self-reported accomplishments [6]. When applying this principle to the CampusXConnect platform, it will allow the user to build a project portfolio by attaching links to active demonstrations of the project, screenshots of the project and team-based attributions associated with the project. Outstanding projects that will be displayed on the homepage of the CampusXConnect platform will serve as a visible incentive to students to document their project in a quality manner. The methodologies in which each of the CampusXConnect project portfolios will be build upon an assigned role-based permission structure as a supporting architecture, similar to how project-based repositories are assigned in GitHub, and the project portfolios will be secured through token expiration and bcrypt-hashed credentials [8].

D. Future Directions in Campus Technology

Personalisation through Machine Learning (ML) is the biggest improvement that we can make in the immediate future. At the moment, when a user is recommended an alumni, that recommendation is based on filters which the user manually sets. If we build a recommendation engine model from student's academic performance, project history and interests then the alumni a user is recommended will be more relevant to that user than if they were to search for them. We can also apply predictive analytics on some of the coding and forum activity of students to allow placement coordinators to better understand which students may benefit from extra academic support[1].

TABLE I Comparative Analysis of Existing Platforms

Platform	Primary Function and Campus Limitation
LinkedIn	Professional networking and alumni connections; lacks integration with academic workflows and campus placement resources.
GitHub	Version control and open-source project hosting; no community forum, coding judge, or alumni networking capability.
LeetCode	Coding practice and interview preparation; operates independently without academic or social campus context.
Stack Overflow	Community-driven Q&A for technical queries; not scoped to institutional communities or placement pipelines.
Discord	Real-time voice and text communication; lacks structured academic content, role-based permissions, and placement tools.

V. DISCUSSION

Many architectural and user experience problems were determined in the development process which will continue to affect more than just the system being developed. The biggest ongoing technical challenge was to deliver the same experience across all eight modules. The sharing of some of the components, such as navigation structure, the notification delivery pipeline and the unified user context object required a number of refactoring to ensure the components work across the different user surfaces of the modules.Although the modularization approach is useful in building independently testable modules, the overhead costs of having to integrate the source code across the different modules was too expensive and it was not recognised in the early stages of the development process, which led to additional sprints being required to fix integration problems.

Based on the data collected from user data during the internal testing, students gave positive feedback for the coding practice module. For instance; students liked the fact they could practice coding problems without having to navigate away from the website and were encouraged to complete coding problems as they completed their coding assignments on their personal profiles. Students who played the coding practice at least ten times a week had a higher completion of their profile completion and alumni network activity, compared to students who did not regularly complete coding assignments. The coding module was an indicator of platform usage.

On the other hand, the alumni connection module has a challenge with adoption that is more social than technical in nature as the value of the module increases as more alumni use the module, so restricting development in this area will likely not have the desired effect.

VI. CONTRIBUTION

This paper introduces a new campus-specific system that allows students a single sign-on (SSO) to multiple applications for social interaction with alumni (e.g. LinkedIn), live programming (e.g. Judge0), placement content, community question-and-answer (Q&A), real-time communication (e.g. chat) and project showcase within a single authentication domain. To the best of the authors' knowledge, this is the first product to provide such a campus-level solution, where all modules use a single user identity, role-based access control and notification mechanism.

One of the main technical highlights of this paper is how Judge0 is integrated into the campus platform, making it a natural part of how students interact and learn.. This integration enables a student to interact with a placement preparation resource, solve the corresponding programming exercise, post a question on the forum and connect to an alumni mentor, all while staying logged in. This seamless experience is a significant improvement on the current fragmented experience of having to use multiple platforms for preparation for placement at many technical institutes.

The creation of two databases using PostgreSQL and MongoDB to run alongside a single Express.js service layer demonstrates a design pattern that can be used for the development of educational platforms that need to manage both structured and unstructured data. This design pattern should help the development of educational platforms, as they will face the same problem in building their educational platforms to serve students in the same way as this design.

VII. CONCLUSION

CampusXConnect is a new online system that addresses the balance of two student needs: academic and placement. This system provides students with a way to manage their alumni connections, code and access placement preparation resources, engage in discussions and forum activities, present projects and exchange real-time chat communications in a secure single-sign-on (SSO) environment. This integration provides an integrated platform for students to develop for their future placement into the workforce, as well as a reduction in the time wasted navigating between applications throughout the students' existing workflows in the technical college.

All of the components of the technology stack worked seamlessly throughout the development process and the internal testing period. The use of two types of databases allowed easy handling of the different types of data in the different modules and the modular composition of the service allowed developers to test the different components of the service separately from each other while improving their performance, without impacting other components. The development priorities include: - An alumni recommendation engine to help users find suitable career paths based on their interests and previous career - A leaderboard of activity to promote competition - Integration with colleges' existing ERP systems to help automate the verification of student enrollment and the connection between the student records and their academic record Alongside these improvements, a longitudinal study of student use of the service and its impact will be conducted to determine whether the use of CampusXConnect enhances the student experience over students' past use of discrete systems.

ACKNOWLEDGMENT

The guidance and mentorship of Mr. Ashish Patel, Assistant Professor of Computer Science and Engineering at Dronacharya Group of Institutions, Greater Noida, is greatly appreciated by the authors. His technical expertise, coupled with his continual availability for consultation, was instrumental in helping the Authors make architectural decisions regarding the platform. In particular, his advice on selecting a database, structuring an API, and designing security helped the team navigate important design trade-offs that affected the overall quality of the final product.

The authors also appreciate the faculty and administrative staff of the Department of Computer Science and Engineering at Dronacharya Group of Institutions for providing an academic environment, lab facilities, and institutional support for the project. Their support during the development and testing phases of CampusXConnect were essential in allowing the Authors to complete this work through all Modules.

Special thanks are given to the student community at Dronacharya Group of Institutions who assisted in testing and evaluating the platform for its users. The Students provided candid feedback on usability, feature expectations, and workflow preferences which were useful for improving the end-user experience. Furthermore, the authors would like to thank the Open Source communities behind Node.js, React, Flutter, MongoDB, PostgreSQL, Socket.io, and Judge0 for their contributions to make this project feasible from a technical perspective.

REFERENCES

1. R. Kumar and S. Sharma, "Design and implementation of a campus social network for enhancing student engagement," IEEE Access, vol. 10, pp. 22010–22025, 2022. doi: 10.1109/ACCESS.2022.3152301
2. S. Jhaver, A. Bruckman, and E. Gilbert, "Does transparency in moderation really matter? User behavior after content removal explanations on Reddit," Proc. ACM Hum.-Comput. Interact., vol. 3, no. CSCW, pp. 1–27, 2019. doi: 10.1145/3359252
3. C. Toraman, F. Sahinuc, E. H. Yilmaz, and I. B. Akkaya, "Understanding social engagements: A comparative analysis of user and text features in Twitter," Soc. Netw. Anal. Min., vol. 12, no. 1, p. 47, 2022. doi: 10.1007/s13278-022-00872-1
4. G. H. Prinster, C. E. Smith, C. Tan, and B. C. Keegan, "Community archetypes: An empirical framework for guiding research methodologies," Proc. ACM Hum.-Comput. Interact., vol. 8, no. CSCW1, pp. 1–33, 2024. doi: 10.1145/3637310
5. P. Shrestha, A. Sathanur, S. Maharjan, E. Saldanha, D. Arendt, and S. Volkova, "Multiple social platforms reveal actionable signals for software vulnerability awareness: A study of GitHub, Twitter and Reddit," PLoS ONE, vol. 15, no. 3, p.
6. e0230250, 2020. doi: 10.1371/journal.pone.0230250
7. T. A. Maniou and P. Bantimaroudis, "Hybrid salience: Examining the role of traditional and digital media," Journalism, vol. 22, no. 4, pp. 1127–1144, 2021. doi:
8. 10.1177/1464884918796587
9. A. N. Smith, E. Fischer, and C. Yongjian, "How does brandrelated user-generated content differ across YouTube, Facebook, and Twitter?" J. Interact. Mark., vol. 26, no. 2, pp. 102–113, 2012. doi: 10.1016/j.intmar.2012.01.002
10. I. Literat and N. Kligler-Vilenchik, "How popular culture prompts youth collective political expression on social media,"
11. W. He, S. Zha, and L. Li, "Social media competitive analysis and text mining: A case study in the pizza industry," Int. J. Inf.
12. Manage., vol. 33, no. 3, pp. 464–472, 2013. doi: 10.1016/j.ijinfomgt.2013.01.001
13. H. Kassim, R. Ahmad, and M. Yusof, "The role of mentorship in undergraduate placement outcomes: A systematic review," Int. J. Educ. Technol., vol. 15, no. 2, pp. 45–58, 2022.
14. S. Singha, "Exploring the role of social media in enhancing K12 economics education," in Cases on Economics Education and Tools for Educators, IGI Global, 2024, pp. 160–182. doi: 10.4018/978-1-6684-7583-6.ch007
15. A. Verma, M. Harper, S. Assi, A. Al-Hamid, M. G. Yousif, and J. Mustafina, "Suicide ideation detection: A comparative study of sequential and transformer hybrid algorithms," in Proc. Int. Conf. Data Sci. Emerg. Technol., Singapore: Springer Nature, 2022, pp. 373–387. doi: 10.1007/978-981-99-

0741-0_27