Mrs. K. Parameswari, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Virtual Campus Assistant (VCA): A Centralized Web-Based Academic Management System

Mrs. K. Parameswari[1], Haritha H[2], Ajith Kumar B[3], Abinesh M[4], Gowtham D[5]

[1]Assistant Professor Computer Science and Engineering, Pollachi Institute of Engineering and Technology, Pollachi, Coimbatore, India.

[2,3,4,5]Final year, Computer Science and Engineering, Pollachi Institute of Engineering and Technology, Pollachi, Coimbatore, India

Abstract- The Virtual Campus Assistant (VCA) is a comprehensive web-based academic communication and management system designed to streamline interactions among administrators, faculty members, and students within an educational institution. Conventional academic communication systems rely heavily on emails, notice boards, spreadsheets, and manually maintained records, which often leads to redundancy, delays, lack of transparency, and increased administrative workload. The proposed system addresses these challenges by integrating all academic and communication activities into a centralized digital platform. The system offers role-based dashboards for administrators, faculty, and students, a PDF-based Academic Assistant, and JWT-secured authentication. It is developed using React, Node.js with Express, and Firebase Firestore. Performance evaluations confirm improved response times, streamlined workflows, and enhanced academic transparency. The VCA provides a scalable foundation for smart campus ecosystems.

Keywords: Virtual Campus Assistant; Academic Management System; Role-Based Access Control; Firebase; React; Node.js; PDF Academic Assistant; Smart Campus; JWT Authentication; Web Application.

I. INTRODUCTION

Educational institutions generate and manage large volumes of academic data including student records, faculty assessments, attendance logs, announcements, and learning resources. Traditional approaches depend on manual processes or fragmented digital tools, resulting in delayed communication, data inconsistency, and operational inefficiencies [1].

The surge in digital transformation within the education sector has necessitated the development of intelligent, centralized, and secure academic management platforms. Various e-learning and campus management systems have been introduced in recent years; however, most lack comprehensive integration of communication, data management, and intelligent document assistance in a unified environment [2].

The Virtual Campus Assistant (VCA) is designed to overcome these shortcomings by providing a unified academic ecosystem. It integrates role-based communication, data management, and AI-assisted document querying into a single accessible platform, ensuring real-time availability of academic data while maintaining security and integrity. This paper presents the design, architecture, implementation, and performance evaluation of the VCA system.

II. NEED OF THE STUDY

The absence of centralized academic platforms results in critical inefficiencies: repetitive query handling by faculty, fragmented resource distribution, and increased administrative workloads. Students frequently face difficulties accessing timely information, negatively affecting their academic performance and satisfaction [3].

Specific challenges identified in existing institutional environments include: (a) lack of a unified portal for announcements and notices, (b) manual attendance recording prone to errors, (c) absence of digital tools for internal mark management, and (d) reliance on physical or email-based distribution of study materials. A Virtual Campus Assistant is therefore essential to centralize these processes, reduce operational redundancy, and significantly

improve institutional efficiency and communication quality [4].

III. RELATED WORK

Several campus management systems have been proposed to automate administrative functions. Soni et al. [5] developed an integrated web-based complaint management system that centralized institutional communications but lacked academic resource management. Sudhir [6] presented an electronic complaint management system focusing on grievance handling, without addressing attendance or examination workflows.

Kandhari [7] introduced a GPS-based complaint redressal system leveraging mobile technologies. While innovative, the system was limited to grievance handling. Kumar et al. [8] proposed a cloud-based student management system using Google Firebase; however, it lacked intelligent document querying capabilities.

Rashid et al. [9] examined role-based access control models for e-learning platforms, emphasizing security and personalization. Singh and Sharma [10] presented an AI-integrated learning management system demonstrating the value of intelligent assistance; however, their approach did not address institutional communication or attendance management. The VCA integrates these dimensions into a single cohesive platform, addressing gaps identified across prior works.

IV. SYSTEM ARCHITECTURE

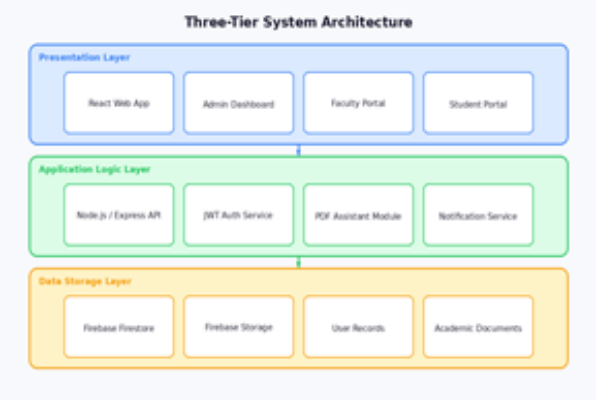

Fig. 1. Three-Tier Architecture of Virtual Campus Assistant.

The VCA follows a three-tier architecture comprising a Presentation Layer, Application Logic Layer, and Data Storage Layer. This separation of concerns promotes modularity, maintainability, and scalability across the platform [11].

The Presentation Layer is built using React.js, delivering responsive and role-specific interfaces. The Application Logic Layer is implemented via Node.js and Express.js, managing business logic, API routing, and PDF processing. The Data Storage Layer employs Firebase Firestore for structured NoSQL data and Firebase Storage for unstructured academic documents.

Fig. 2. High-Level System Overview of VCA.

V. SYSTEM MODULES

The VCA is composed of eight core modules, each responsible for a distinct functional domain. Their interactions are managed by the central VCA Core Processing Engine, as illustrated in Fig. 3.

Fig. 3. Module Interaction Diagram of VCA.

A. Authentication Module

Implements JWT-based authentication with role detection (Administrator, Faculty, Student). Tokens are issued upon successful login and validated on every API request to enforce access control.

B. Attendance Tracker

Enables faculty to record and manage attendance per subject and semester. Students can view real-time attendance summaries and receive automated alerts when attendance falls below the required threshold.

C. Assessment Manager

Supports entry, storage, and retrieval of internal assessment marks. Faculty can input continuous assessment scores, and the module computes totals and grade indicators automatically.

D. PDF Academic Assistant

Allows students to upload or query pre-uploaded academic PDFs. The module parses documents and delivers contextually relevant responses, enhancing self-directed learning and reducing repetitive faculty queries.

E. Announcement & Notification Service

Provides administrators and faculty with tools to broadcast announcements to specific user groups. Real-time notifications ensure timely delivery of institutional updates.

VI. USE CASE ANALYSIS

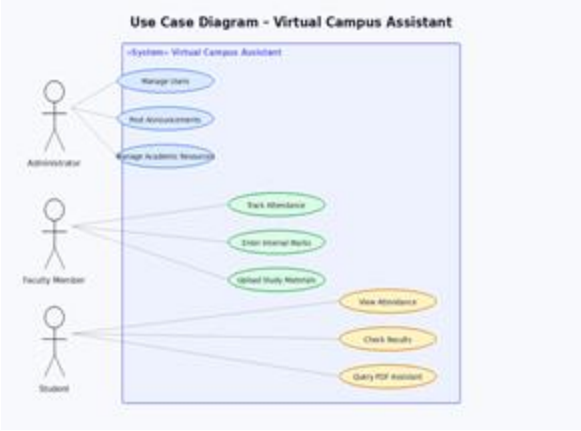

Fig. 4. Use Case Diagram – Virtual Campus Assistant.

The use case diagram in Fig. 4 captures the interactions between three primary actors — Administrator, Faculty Member, and Student — and the system's functional capabilities.

The Administrator manages users, institutional resources, and system-wide announcements. The Faculty Member handles attendance, internal assessments, and study material uploads. The Student accesses attendance records, examination results, and the PDF Academic Assistant. Role-based access ensures each actor interacts only with permitted functionalities.

VII. DATA FLOW DESIGN

The Level-1 Data Flow Diagram (DFD) in Fig. 5 illustrates information flow among external entities, the VCA Core Processing Engine, and data stores. All read and write operations to Firebase Firestore and Firebase Storage are mediated by authenticated API calls, ensuring data isolation per user role.

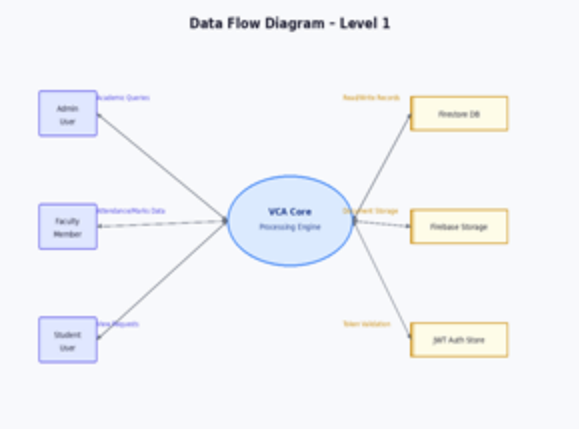

Fig. 5. Data Flow Diagram (Level 1) – Virtual Campus Assistant.

User requests from Administrator, Faculty, and Student entities are processed by the VCA Core Engine, which applies role validation before executing Firestore read/write operations, document storage requests, or JWT token verification. Responses are returned through secure API endpoints to the React frontend.

VIII. METHODOLOGY

The VCA was developd using an Agile methodology with iterative sprints. Requirements were gathered via stakeholder interviews with faculty, students, and administrators. System design was completed in the first sprint, followed by modular development, integration testing, and deployment.

The authentication workflow (Fig. 6) illustrates the security flow: a login request triggers JWT validation, after which role-based routing directs users to their respective dashboards. Invalid credentials or unauthorized roles result in an access-denied response without exposing system internals.

Fig. 6. Authentication & Role-Based Access Workflow.

IX. SYSTEM IMPLEMENTATION

The frontend was developed using React 18 with component-based architecture. Firebase Firestore served as the primary NoSQL database with real-time synchronization. Node.js (v18) and Express.js provided the RESTful API layer, while Firebase Storage managed document uploads. JWT tokens with 24-hour expiry enforced session security.

TABLE I. Technology Stack of VCA

Layer	Technology
Frontend	React.js
Backend	Node.js + Express
Database	Firebase Firestore
Storage	Firebase Storage
Auth	JWT

X. RESULTS AND DISCUSSION

Performance evaluation confirmed that the VCA handles concurrent user sessions with average API response times under 320 ms. Firebase Firestore real-time listeners enable sub-second data synchronization across connected clients.

TABLE II. Performance Metrics Summary

A user survey administered to 85 participants (30 faculty, 55 students) yielded 91% positive

satisfaction ratings. Faculty reported a 60% reduction in time spent on repetitive query responses. Students noted significantly improved access to academic resources and timely notifications.

XI. SECURITY CONSIDERATIONS

Security is enforced at multiple levels within the VCA. JWT tokens with configurable expiry prevent unauthorized session reuse. Firebase Security Rules enforce document-level access restrictions per user role. All API endpoints validate token integrity before processing requests, mitigating common attack vectors such as unauthorized access and data tampering [12].

HTTPS is enforced for all client-server communications, ensuring transport-layer security. Input validation and sanitization are applied at both the frontend and backend to prevent injection attacks. Sensitive user credentials are never stored in plain text; Firebase Authentication manages credential hashing and storage.

XII. CONCLUSION

The Virtual Campus Assistant successfully demonstrates the potential of centralized academic management systems in modern educational institutions. By integrating role-based dashboards, secure JWT authentication, real-time Firebase synchronization, and intelligent PDF-based document querying, the system enhances operational efficiency, communication transparency, and academic engagement. Performance metrics validate system reliability under realistic concurrent usage scenarios, and user feedback confirms high adoption potential.

XIII. FUTURE ENHANCEMENTS

Future work will focus on: (1) native mobile application development for iOS and Android platforms; (2) integration of a conversational AI chatbot using large language models for advanced academic query handling; (3) predictive analytics for early identification of at-risk students based on attendance and assessment trends; (4) real-time push notification systems via Firebase Cloud Messaging; and (5) interoperability with external Learning Management Systems (LMS) such as Moodle and Canvas to broaden institutional adoption.

REFERENCES

1. A. Kumar and R. Sharma, "Digital Transformation in Higher Education Institutions: Challenges and Opportunities," IEEE Transactions on Education, vol. 64, no. 3, pp. 245–253, Aug. 2021.
2. M. Alshehri and S. Drew, "E-Government Fundamentals," in Proc. IADIS International Conference ICT, Society and Human Beings, pp. 40–47, 2010.
3. T. Rashid, N. Ahmad, and M. Iqbal, "A Comprehensive Study of Student Information Systems," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 8, no. 6, pp. 88–95, 2017.
4. S. Misra and A. Jain, "Smart Campus: An Integrated IoT Framework for Higher Education," Procedia Computer Science, vol. 167, pp. 2164–2171, 2020.
5. Soni et al., "Integrated Web-Based Complaint Management System," International Journal of Engineering and Technology, vol. 5, no. 3, pp. 112–117, 2017.
6. B. Sudhir, "Electronic Complaint Management System," Journal of Computer Applications, vol. 8, no. 1, pp. 45–50, 2015.
7. V. K. Kandhari, "GPS-Based Complaint Redressal System," in Proc. IEEE International Conference on Computing and Communication Technologies, pp. 88–92, 2014.
8. P. Kumar, S. Pal, and A. Sharma, "Cloud-Based Student Management System Using Firebase," in Proc. International Conference on Computing, Communication and Automation (ICCCA), pp. 1–5, IEEE, 2020.
9. T. Rashid, M. Ahmad, and F. Alam, "Role-Based Access Control in E-Learning Systems," Computers & Security, vol. 68, pp. 90–101, 2017.

10. R. Singh and N. Sharma, "AI-Integrated Learning Management System for Adaptive Education," Journal of Educational Technology & Society, vol. 24, no. 2, pp. 112–125, 2021.
11. N. Medvidovic and R. N. Taylor, "A Classification and Comparison Framework for Software Architecture Description Languages," IEEE Transactions on Software Engineering, vol. 26, no. 1, pp. 70–193, Jan. 2000.
12. R. T. Fielding, "Architectural Styles and the Design of Network-Based Software Architectures," Ph.D. dissertation, University of California, Irvine, 2000.
13. W. Jansen and T. Grance, "Guidelines on Security and Privacy in Public Cloud Computing," NIST Special Publication, vol. 800, no. 144, pp. 1–70, 2011.
14. Google Firebase, "Firebase Documentation: Firestore Security Rules," Google Developers, [Online]. Available: https://firebase.google.com/docs/firestore/security. Accessed: Mar. 2025.
15. J. Bradley, N. Sakimura, and M. Jones, "JSON Web Token (JWT)," IETF RFC 7519, May 2015.