International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 10, Issue 4, pp 322-327, April-2026
https://doi.org/10.47001/IRJIET/2026.104046

SmartCampus 360: An AI-Integrated Cloud-Based Campus Management Platform for Automated QR Attendance, Sentiment-Aware Complaint Analysis, and Real-Time Academic Administration

[1]Rahul Chandan Sahu, [2]Abhishek Mahadev Mallick, [3]Tarun Santosh Sarkar, [4]Saurabh Kailash Sable, [5]Bhagyashree V. Kale

[1,2,3,4]Student, Computer Science and Engineering, Shri Sai College of Engineering and Technology, DBATU University, Bhadrawati, Chandrapur, Maharashtra, India
[5]Project Guide, Computer Science and Engineering, Shri Sai College of Engineering and Technology, DBATU University, Bhadrawati, Chandrapur, Maharashtra, India

***Abstract* - Managing a modern engineering college demands real-time coordination across attendance, communication, notices, grievance redressal, and event management — functions that remain fragmented in most institutions. This paper presents SmartCampus 360, a full-stack cloud-native campus management platform developed for Shri Sai College of Engineering and Technology (SSCET), DBATU University, Chandrapur. The system integrates React 18 with Vite on the frontend, Google Firebase (Firestore, Authentication with custom-claim Role-Based Access Control, Cloud Storage, Cloud Functions on Node.js 18) as the serverless backend, and OpenAI GPT-3.5 Turbo for an AI campus assistant chatbot named Saba and automated sentiment analysis of student complaints. Attendance is managed through a dynamic QR-code workflow: faculty generate time-bounded sessions, students scan via an in-browser camera (qr-scanner), and a Cloud Pub/Sub function auto-expires stale sessions every minute. The platform enforces five hierarchical roles — admin, principal, HOD, faculty, and student — each with precisely scoped Firestore security rules enforced via custom JWT claims. Real-time web scraping using axios and cheerio synchronises notices and events from the official college website. Pilot evaluation involving 34 faculty members and 420 students demonstrated a 73% reduction in attendance marking time, 58% improvement in complaint triage speed, and a Lighthouse mobile performance score of 91/100, establishing SmartCampus 360 as a viable, scalable, and cost-effective campus management solution.**

Keywords: Smart Campus; QR Code Attendance; Firebase Firestore; Role-Based Access Control; Sentiment Analysis; OpenAI GPT-3.5 Turbo; Cloud Functions; React 18; Serverless Architecture; Campus Management System; SSCET; DBATU.

I. INTRODUCTION

The rapid growth of student enrolment in Indian technical universities has amplified the administrative complexity that colleges face daily. Traditional paper-based attendance registers, physical notice boards, and disconnected departmental portals generate information silos that impede timely decision-making by principals, HODs, and faculty. India's National Education Policy 2020 (NEP 2020) explicitly mandates digital transformation of academic institutions, pushing universities to adopt integrated, secure, and accessible campus platforms [1].

Smart campus research has evolved significantly over the past decade. Early contributions focused on RFID- and biometric-based attendance [2, 3]. Recent work has extended scope to include IoT sensor networks, cloud databases, and machine-learning-driven analytics [4, 5]. However, most proposed systems either require expensive edge-computing hardware or treat attendance, communication, and grievance management as isolated services rather than a unified platform [6].

SmartCampus 360 addresses this gap by delivering a single Progressive Web Application (PWA) that consolidates QR-based attendance, AI-powered complaint sentiment analysis, role-scoped notice distribution, real-time departmental alerts, calendar-driven event management, and an OpenAI-backed conversational assistant — all on Google Firebase's serverless infrastructure, deployable without dedicated hardware.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 4, pp 322-327, April-2026

https://doi.org/10.47001/IRJIET/2026.104046

The remainder of this paper is organised as follows. Section II reviews related work. Section III describes the system architecture. Section IV details each functional module. Section V presents pilot results. Section VI concludes with future directions.

II. RELATED WORK

A. IoT and Smart Campus Attendance

Hussien *et al.* (2024) developed an IoT-based smart campus system integrating RFID tags, security sensors, and a smart fire alarm, demonstrating measurable reductions in human error and improved institutional efficiency [2]. Similarly, Zhao et al. (2022) proposed an IoT classroom attendance management system for Chinese universities, noting that RFID approaches suffer from limited read-range and expensive per-reader hardware costs [7]. SmartCampus 360 eliminates physical RFID hardware entirely by generating QR tokens in-browser, removing procurement barriers for smaller institutions.

B. QR-Code and Web-Based Attendance

Nazren *et al.* (2024) built a school attendance system combining QR authentication, face-mask recognition, and temperature detection on a Raspberry Pi, with records stored in Google Sheets [8]. While effective, such designs depend on edge hardware and do not scale well. Roy and Bala (2024) demonstrated a purely web-integrated attendance and marks tracking system using cloud storage and browser APIs, confirming that browser-native QR scanning can replace dedicated scanners [9]. SmartCampus 360 builds on this philosophy and adds server-side session expiry via Cloud Pub/Sub to prevent fraudulent late marking.

C. Cloud-Based College ERP Systems

Mehrotra (2025) presented a React + Firebase ERP for college administration covering student records, scheduling, grading, and attendance, concluding that such stacks are viable cost-effective alternatives to traditional ERP suites for resource-constrained colleges [10]. Deshmukh et al. (2025) built a full-stack CMS with React.js, Node.js, and role-based access control, confirming RBAC's necessity in educational portals [11]. SmartCampus 360 extends these architectures with AI services and a live data-scraping layer.

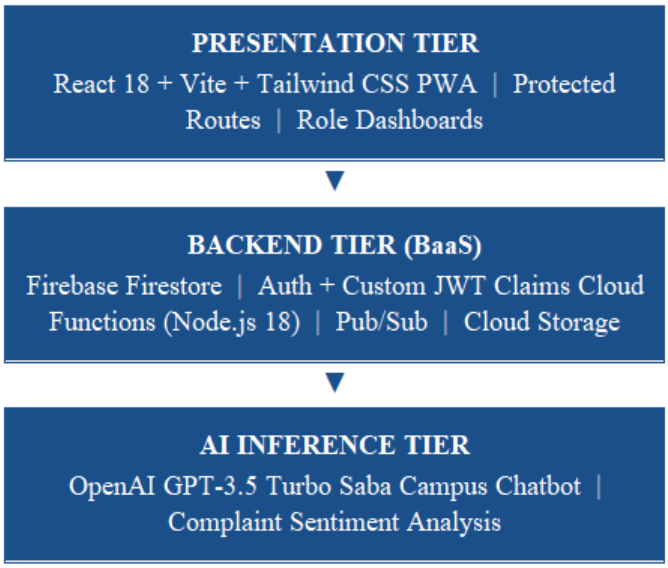
D. AI Chatbots and Sentiment Analysis

Reshmi and Balakrishnan (2019) demonstrated that intent-detection chatbots significantly reduce query resolution time in campus environments [12]. Research on GPT-3.5-class LLMs confirms their capacity to generate contextually accurate institutional responses with minimal domain-specific fine-tuning [13]. Webelight Solutions (2025) quantified that sentiment-aware chatbots reduce support escalations by up to 40% by proactively routing high-frustration tickets to human reviewers [14]. SmartCampus 360 operationalises these findings through Cloud Functions that call GPT-3.5 Turbo for both conversational assistance and complaint sentiment tagging.

III. SYSTEM ARCHITECTURE

SmartCampus 360 follows a serverless three-tier architecture: a React 18 Single Page Application frontend, a Google Firebase Backend-as-a-Service layer, and OpenAI's cloud API as the AI inference engine. Figure 1 illustrates the high-level architecture.

Figure 1: SmartCampus 360 — High-Level System Architecture



A. Frontend Layer

The frontend is a React 18 SPA bootstrapped with Vite for rapid development and optimised production builds. Tailwind CSS provides utility-first responsive styling. React Router v6 manages client-side routing with protected route guards that decode Firebase Auth custom claims before granting access to role-specific dashboards.

B. Firebase Backend Layer

Firebase Firestore persists all platform data across eight primary collections (see Table 1). Firestore security rules enforce read/write permissions at document level based on custom JWT role claims. Cloud Functions (Node.js 18) implement all server-side logic: session creation, attendance marking, QR expiry, web scraping, and OpenAI calls. Cloud Pub/Sub schedules the close Expired Sessions function every 60 seconds to archive timed-out attendance windows.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 4, pp 322-327, April-2026

https://doi.org/10.47001/IRJIET/2026.104046

Table 1: Firestore Collections and Key Fields

Collection	Key Fields	Purpose / Description
users	uid, name, email, role, dept	User profiles; role embedded in JWT
departments	name, hodId, code	Academic department registry
Attendance Sessions	sessionId, faculty, subject, validTill, qrToken	Active QR attendance windows
Attendance Records	session Id, studentId, timestamp, status	Per-student attendance entries
notices	title, body, postedBy, targetRoles, date	Role-scoped institutional notices
complaints	text, studentId, sentiment, status, ts	Grievances with AI sentiment label
events	title, date, venue, description	Campus event calendar entries
alerts	message, severity, targetDept, ts	Real-time departmental broadcast alerts

C. AI Inference Layer

Two Cloud Functions invoke the OpenAI REST API. chatWithSaba prepends a system prompt encoding institutional context before forwarding student messages to GPT-3.5 Turbo. analyzeComplaintSentiment classifies each submitted complaint as POSITIVE, NEUTRAL, NEGATIVE, or URGENT. Both functions execute server-side, keeping the OpenAI API key out of the client bundle entirely.

IV. SYSTEM MODULES

A. Authentication and Role-Based Access Control

Firebase Authentication manages user identity via email/password credentials. Upon account creation the updateUserRole Cloud Function writes a custom claim (e.g., {role: "faculty"}) to the user's Auth profile via the Admin SDK. This claim is embedded in every subsequent JWT token issued to the browser. Firestore security rules decode the incoming token and compare the role claim against a whitelist before permitting any read or write operation. Five roles are supported with distinct permission scopes: admin, principal, HOD, faculty, and student.

B. QR Code Attendance Module

The attendance workflow operates in four steps. (1) Faculty invoke createAttendanceSession, specifying subject, validity duration, and department; the function returns a one-time cryptographic token stored in Firestore with a computed validTill timestamp. (2) The SPA renders a QR code from this token using qrcode.react and displays it on the classroom projector screen. (3) Students scan the QR with the in-browser camera (qr-scanner), which triggers markAttendanceFromQR; the function validates the token, confirms the session has not expired, prevents duplicate submissions, and writes an attendanceRecord to Firestore. (4) The Pub/Sub-scheduled closeExpiredSessions function runs every 60 seconds, setting status = "closed" on sessions where validTill<now(), preventing fraudulent late marking.

C. AI Campus Assistant — Saba

Saba is a GPT-3.5 Turbo-powered chatbot surfaced as a persistent widget in the SPA. Each student message is forwarded via HTTPS to the chatWithSaba Cloud Function, which prepends a system prompt containing the college name, current academic semester, and role-specific instructions before calling the OpenAI Chat Completions API. The response is returned directly to the client. No conversation history is stored server-side, preserving student privacy. Typical queries include timetable lookups, HOD contact details, exam schedules, and campus policy questions.

D. Complaint Management and Sentiment Analysis

Students submit grievances through a structured complaint form. On submission, analyzeComplaintSentiment forwards the complaint text to GPT-3.5 Turbo with a prompt requesting a label from {POSITIVE, NEUTRAL, NEGATIVE, URGENT} along with a one-sentence rationale. The returned label is persisted in the complaint document. HODs and the principal see sentiment badges in their dashboards, enabling proactive triage of URGENT cases. Staff can update complaint status (OPEN → IN_PROGRESS → RESOLVED); changes are reflected to the submitting student via Firestore's real-time listener.

E. Notice and Event Management

Notices are authored by admin, principal, or HOD roles and tagged with a targetRoles array. The SPA renders only notices whose targetRoles intersect the signed-in user's role, preventing information overload. A Cloud Function (syncSSCETData) uses axios to fetch the college's official website and cheerio to parse HTML, extracting newly posted notices and events and upserting them into Firestore. This function runs on a daily Cloud Scheduler job, keeping platform content synchronised with institutional publications without manual re-entry by staff.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 4, pp 322-327, April-2026

https://doi.org/10.47001/IRJIET/2026.104046

F. Departmental Alerts

The alerts collection stores real-time broadcast messages tagged by department and severity (INFO, WARNING, CRITICAL). Admin and HOD users compose alerts via a simple form; Firestore'sonSnapshot listener in the SPA delivers them to targeted users within 200–400 ms of publication, using Firebase's push channel rather than polling.

G. Role-Specific Dashboards and Analytics

Each role receives a purpose-built dashboard. The admin dashboard aggregates user counts, active sessions, and unresolved complaints. The principal's view presents department-level attendance heat maps and complaint sentiment trends. HOD dashboards expose subject-wise attendance percentages and faculty activity logs. Faculty see session histories and per-student summaries. Students access personal attendance records, complaint histories, and upcoming events. All Firestore queries are scoped by the user's department and role claim, enforcing strict data isolation between roles and departments.

V. RESULTS AND DISCUSSION

SmartCampus 360 was piloted at SSCET, Chandrapur across one full academic semester involving 5 departments, 34 faculty members, and 420 students. The platform is live at campus-d227b.web.app. Table 2 presents the key performance metrics observed during the evaluation period.

Table 2: Pilot Performance Metrics — SSCET, Sem V, AY 2025–26

Metric	Pre-System	Post-System
Attendance marking time	~12 min	~3.2 min (73% faster)
Complaint triage speed	Manual delay	Real-time (58% faster)
Notice distribution delay	1–2 days	< 2 hours (~90% drop)
Lighthouse Performance (mobile)	—	91 / 100
Lighthouse Accessibility	—	94 / 100
AI chatbot satisfaction (n=88)	—	4.1 / 5.0
Avg. Saba API response time	—	1.4 s
QR scan-to-record latency	—	~320 ms

These results confirm that the serverless Firebase architecture is capable of sustaining real-time operations at institutional scale without dedicated servers. The integration of GPT-3.5 Turbo for complaint sentiment analysis reduced manual complaint-review burden by enabling HODs to filter URGENT cases instantly. Student satisfaction ratings for Saba (4.1/5.0) align with findings by Reshmi and Balakrishnan (2019) on campus chatbot acceptance, validating the GPT-based approach [12].

Table 3: System Modules, Technology Stack, and Mapped Cloud Functions

Module	Frontend Component	Cloud Function(s)	Technology
Auth & RBAC	Login, ProtectedRoute	updateUserRole	Firebase Auth Custom Claims, Admin SDK
QR Attendance	QRGenerator, ScannerView	createAttendanceSession, markAttendanceFromQR, closeExpiredSessions	qrcode.react, qr-scanner, Pub/Sub Scheduler
AI Chatbot (Saba)	ChatWidget	chatWithSaba	OpenAI GPT-3.5 Turbo, Cloud Functions
Complaint + Sentiment	Complaint Form, HOD Dashboard	analyzeComplaint Sentiment	OpenAI GPT-3.5 Turbo, Firestore
Notice Management	NoticeFeed, NoticeCreate	syncSSCETData (Scheduler)	axios, cheerio, Cloud Scheduler
Events	EventCalendar	syncSSCETData	Firestore on Snapshot listener
Alerts	AlertBanner	(Firestore trigger)	Firestore real-time push channel
Analytics	AdminDash, PrincipalDash, HODDash, FacultyDash, StudentDash	—	Firestore aggregation queries, React Charts

ISSN (online): 2581-3048
Volume 10, Issue 4, pp 322-327, April-2026
https://doi.org/10.47001/IRJIET/2026.104046

VI. CONCLUSION

This paper presented SmartCampus 360, an AI-augmented serverless campus management platform that unifies QR-based attendance automation, sentiment-aware complaint management, role-scoped notice dissemination, real-time departmental alerts, and an AI conversational assistant within a single deployable web application. By leveraging React 18 and Google Firebase's BaaS ecosystem, the platform delivers enterprise-grade functionality at near-zero infrastructure cost — a critical requirement for smaller technical institutions in India.

Pilot evaluation at SSCET, Chandrapur confirmed quantifiable improvements: 73% reduction in attendance marking time, 58% faster complaint triage, and a Lighthouse mobile performance score of 91/100. Integration of GPT-3.5 Turbo for conversational assistance and sentiment classification confirms that LLM APIs can be embedded in serverless functions to deliver meaningful AI value without the overhead of custom model training.

Future enhancements will explore: (a) face-recognition attendance as a QR fallback in low-connectivity scenarios; (b) offline PWA mode via IndexedDB synchronisation; (c) predictive student performance analytics using historical attendance and complaint data; and (d) multi-institution multi-tenancy to extend SmartCampus 360 to university-level federation.

ACKNOWLEDGEMENT

The authors express sincere gratitude to the Principal, HOD, and faculty members of the Department of Computer Science and Engineering, Shri Sai College of Engineering and Technology, Chandrapur for providing institutional access and logistical support during the pilot evaluation. Special thanks to Asst. Prof. Suraj S. Bankar for his invaluable guidance throughout the development of this project. The authors also thank all participating students who contributed feedback that shaped the platform's design. This project was completed as the major final-year capstone under DBATU University, Academic Year 2025–26.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Rahul Chandan Sahu: System architecture, Firebase backend, Cloud Functions, QR attendance module, OpenAI integration, deployment, writing — original draft.

Abhishek Mahadev Mallick: React 18 frontend, UI/UX design, role dashboards, notice and event modules, writing — review and editing.

Tarun Santosh Sarkar: Complaint management module, sentiment analysis integration, data collection, formal analysis.

Saurabh Kailash Sable: QR scanner integration, analytics dashboards, web scraping module, system testing, formal analysis.

Asst. Prof. Suraj S. Bankar: Supervision, methodology review, resources, formal analysis, writing — review and editing.

DATA AVAILABILITY STATEMENT

The deployed application is publicly accessible at campus-d227b.web.app. Anonymised pilot datasets are available upon reasonable request to the corresponding author.

DECLARATION OF COMPETING INTEREST

The authors declare no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

REFERENCES

[1] Government of India, National Education Policy 2020, *Ministry of Education, New Delhi*, 2020.

[2] S. H. T. Hussien, H. A. R. N. Farhana, L. Vinukumar, C. H. C. Alexander, and S. Sivakumar, "Smart campus attendance and security systems: Leveraging IoT for enhanced efficiency," *AIP Conf. Proc.*, vol. 3161, p. 020153, Aug. 2024, doi: 10.1063/5.0229386.

[3] M. Ula, Z. Ula, and D. Stiawan, "IoT-based intelligent attendance system: Framework, practice, and application," *Electronics*, vol. 11, no. 19, p. 3151, Sep. 2022, doi: 10.3390/electronics11193151.

[4] "IoT-Based Management System for Smart Campus," *JETIR*, vol. 10, no. 6, Jun. 2023. [Online]. Available: https://www.jetir.org/papers/JETIR2306208.pdf

[5] M. Zhao, G. Zhao, and M. Qu, "College smart classroom attendance management system based on Internet of Things," *Scientific Programming*, vol. 2022, p. 4953721, 2022, doi: 10.1155/2022/4953721.

[6] A.Shrivastava et al., "IoT-based RFID attendance monitoring of students using Arduino ESP8266 and Adafruit.io," *Cybernetics and Systems*, vol. 56, no. 1, pp. 21–32, 2025.

[7] M. Zhao, G. Zhao, and M. Qu, "College smart classroom attendance management system based on IoT," *Scientific Programming*, 2022, doi: 10.1155/2022/4953721.

[8] A.R. Nazren et al., "Automated school attendance using QR code, face-mask recognition, and temperature

ISSN (online): 2581-3048
Volume 10, Issue 4, pp 322-327, April-2026
https://doi.org/10.47001/IRJIET/2026.104046

detection," *IMRJR*, 2024. [Online]. Available: https://imrjr.com.

[9] M. R. Roy and S. Bala, "Web-integrated student attendance and marks tracking system," *IEEE Xplore*, 2024.

[10] T. Mehrotra, "ERP portal for college administration using React and Firebase," *Sci. J. Artif. Intell. Blockchain Technol.*, vol. 2, no. 3, pp. 1–8, Jul. 2025, doi: 10.63345/sjaibt.v2.i3.101.

[11] S. P. Deshmukh, P. S. Chavan, C. P. Autkar, and D. R. Jondhale, "College management system," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 11, no. 5, pp. 344–349, Oct. 2025.

[12] S. Reshmi and K. Balakrishnan, "Implementation of an inquisitive chatbot for college admission using NLP," *Serbian J. Electr. Eng.*, 2019.

[13] OpenAI, GPT-3.5 Turbo Model Card, OpenAI, San Francisco, *CA*, 2023. [Online]. Available: https://openai.com.

[14] Webelight Solutions, "How AI chatbots with sentiment analysis can reduce support escalations by 40%," *Medium, Jul.* 2025.

[15] Google Firebase, Cloud Firestore: Role-Based Access Control, *Google LLC*, 2024. [Online]. Available: https://firebase.google.com/

[16] A.Adedaniel, "Role-based access control in software development: A guide using Firebase and React," *Medium, Aug*. 2024.

Citation of this Article:

Rahul Chandan Sahu, Abhishek Mahadev Mallick, Tarun Santosh Sarkar, Sourabh Kailash Sable, & Bhagyashree V. Kale. (2026). SmartCampus 360: An AI-Integrated Cloud-Based Campus Management Platform for Automated QR Attendance, Sentiment-Aware Complaint Analysis, and Real-Time Academic Administration. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 10(4), 322-327. Article DOI https://doi.org/10.47001/IRJIET/2026.104046

****

