Student Management System

A Multi-Role Android and Web Application
For Academic Management Using Flutter and App Write Cloud Backend

Mrs. A Sandhya Rani[1], K. Sri Harshitha[2], L. Bishma Goud[3], K. Vibhas[4], G. Vaishnavi[5]
[1]*Assistant professor, Department of Computer Science and Engineering- Cyber security, Sphoorthy Engineering College.*
[2,3,4,5] *Student, Department of Computer Science and Engineering-Cyber security, Sphoorthy Engineering College.*

***Abstract*—This paper presents a Student Management System (SMS), a cloud-based academic platform built as three role-specific applications Admin, Student, and Teacher each available on both Android and Web. The system is developed using Flutter (Dart) for the frontend and App write as the Backend as a Service (BaaS) for authentication, real-time database, and file storage. A shared Dart package centralizes common logic, models, and services across all three applications. The platform covers the full lifecycle of college operations including student and teacher registration with admin approval workflows, QR code-based attendance tracking, grade management with regulation-based templates, timetable scheduling, assignment management, quiz participation, library access, placement drives, leave management, real-time announcements, campus voice (student feedback), in-app chat, and digital ID card generation. The system eliminates paper-based processes, centralizes institutional data with real-time synchronization, and delivers a premium dark-mode mobile and web experience to all stakeholders.**

***Index Terms*—Student Management System, Flutter, Android Application, Web Application, App write, QR Attendance, River pod, Multi-role Academic Platform.**

I. INTRODUCTION

Managing a modern educational institution requires coordinating large volumes of data across administrators, teachers, and students. Paper-based records and disconnected digital tools create inefficiency, data silos, and poor visibility for all stakeholders. There is a clear need for a unified, real-time platform that serves every role from a single backend. The Student Management System (SMS) addresses this by providing three dedicated applications one for each role built with Flutter, enabling deployment on both Android devices and Web browsers from a single codebase. The App write cloud platform provides a shared backend for all three apps, ensuring data consistency and real-time synchronization.

The three applications are:

- Admin App: Institutional command center for complete system control.
- Student App: Student academic portal for academics and productivity.
- Teacher App: Educator portal for classroom and workload management.

The project aims to replace manual processes with digital workflows, provide role-specific dashboards that surface relevant information instantly, and enable real-time data updates for example, when a teacher starts an attendance session, students can scan in immediately from their app.

II. RELATED WORK

Several academic management tools exist, but most suffer from key limitations:

a) Traditional ERP Systems:

These are monolithic web applications that are difficult to use on mobile devices, lack real-time features, and require heavy IT infrastructure. They often cover only administrative tasks, not the day-to-day academic needs of students and teachers

b) Google Classroom / LMS Platforms:

Platforms like Google Classroom handle learning management effectively but do not cover

administrative functions such as timetable management, leave approval, library management, placement drives, or fee management. They also require a Google Workspace subscription for institutional use.

c) Standalone Mobile Apps:
Many colleges deploy separate apps for attendance, grades, or notifications.
This forces students and staff to manage multiple applications, leading to a fragmented experience and disconnected data.

d) Firebase-Based College Apps:
Some systems use Firebase, but its proprietary nature and cost at scale make it less suitable. App write, used in this project, is open-source and provides equivalent features including authentication, database, storage, and real-time subscriptions.

e) Gap Addressed:
The SMS fills the gap by providing a single platform that covers ALL aspects of college management from user registration and attendance to placement drives and quiz management with a modern, real-time mobile and web interface and strict role-based access control.

III. SYSTEM DESIGN AND METHODOLOGY

1. Problem Definition
College management currently involves separate systems or manual processes for attendance, grading, timetables, and communication. There is no unified platform that connects the admin, teacher, and student into a single real-time ecosystem. The main problem is: how to provide each stakeholder role with relevant, up-to-date academic information through a single, secure, cloud-connected application available on both Android and the Web.

To solve this, the system uses:
- A multi-app Flutter architecture with a shared backend
- App write cloud for authentication, database, and file storage
- Riverpod for reactive state management
- App write Realtime for WebSocket-based live data synchronization
- Feature-first modular code organization for scalability

2. System Architecture
The system follows a three-tier architecture:
a) Presentation Layer:
Three Flutter applications (Admin, Student, Teacher) built with Flutter and deployed as Android APKs and Web apps on Firebase Hosting.

b) Business Logic Layer:
Flutter Riverpod providers managing app state, data fetching, and reactive UI updates shared via a common Dart package.

c) Backend / Data Layer:
App write cloud platform providing Authentication, NoSQL Database, File Storage, and Realtime WebSocket subscriptions.

d) The project is organized as a mono-repo:

```
student_management_system/
 ├── apps/
 │  ├── admin/ → Admin Android & Web Application
 │  ├── student/ → Student Android & Web Application
 │  └── teacher/ → Teacher Android & Web Application
 └── packages/
  └── shared/ → Common package: models, services, providers, theme
```

The shared package contains App write Constants (all collection IDs and project configuration), Auth Service, data models, Riverpod providers, the App Theme design system, and reusable UI components such as Glass Container and Spring Button, used across all three applications.

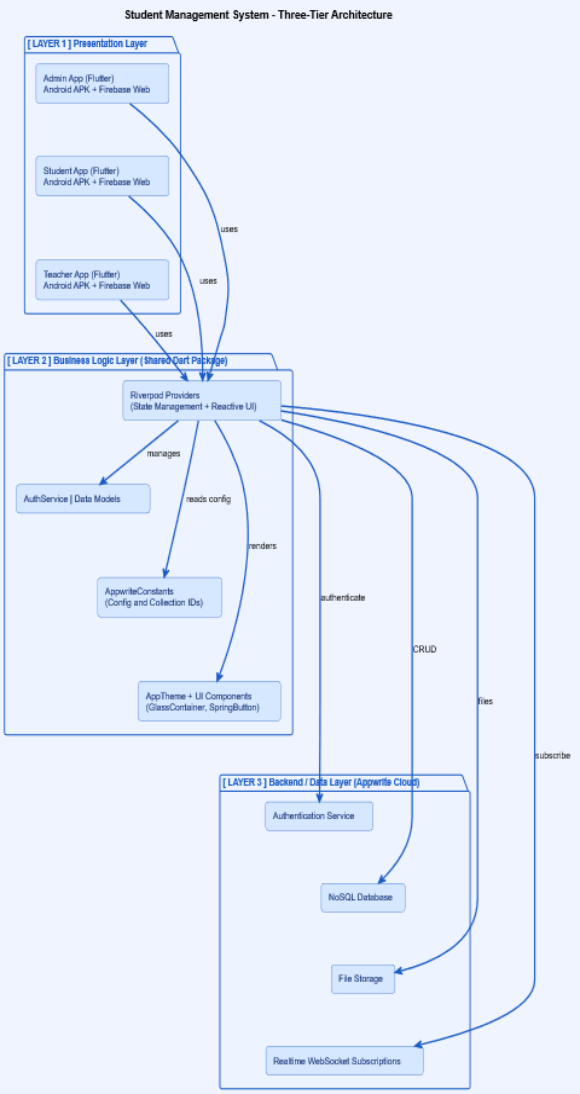


Fig 1: Three-Tier System Architecture Diagram

3. Technology Stack

Frontend:

- Flutter (Dart, SDK ^3.10.7) - cross-platform Android and Web
- flutter_riverpod ^2.6.1 - reactive state management
- App write ^22.0.0 - App write client SDK
- table_calendar ^3.2.0 - Academic calendar display
- fl_chart ^1.1.1 - Grade and analytics charts
- qr_flutter ^4.1.0 - QR code generation (teacher attendance)
- mobile_scanner ^7.2.0 - QR code scanning (student attendance)
- flutter_local_notifications - Local alarm-based reminders
- shared_preferences - Local data caching and session storage
- image_picker, file_picker - Avatar and study material uploads
- url_launcher, intl, uuid - Utility and formatting packages

Backend (App write Cloud):

- Authentication Email-password sessions, OTP magic-link login, email verification, password recovery
- Database NoSQL document storage with 35+ collections
- Storage study_materials_bucket for file uploads
- Realtime WebSocket subscriptions for live data updates

Deployment:

- Android APK builds distributed directly to devices
- Web Flutter web build hosted on Firebase Hosting (3 sites)

4. Database Design

The App write database (sms_main_db) contains the following collections:

a) User & Auth: profiles, students, teachers

b) Academic: departments, branches, classes, subjects, class_subjects, regulations, syllabi, grading_templates

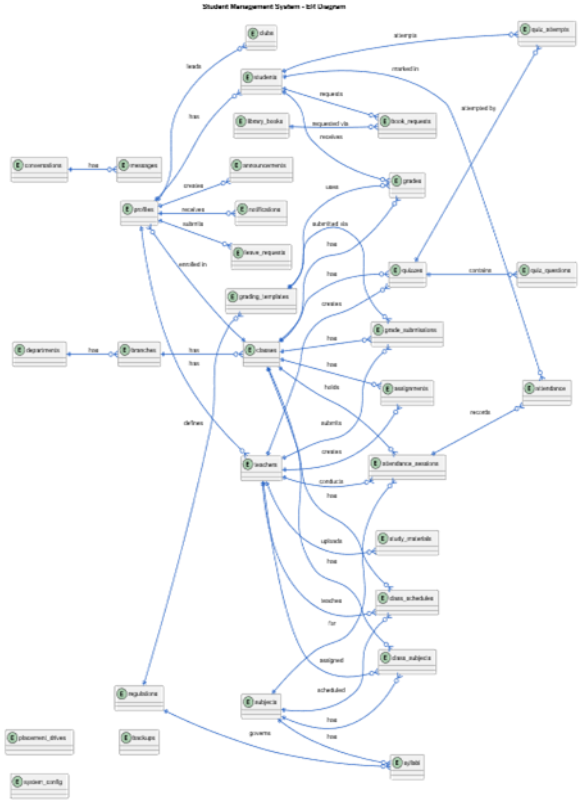
c) Attendance: attendance_sessions, attendance Grades & Exams: grades, grade_submissions, quizzes, quiz_questions, quiz_attempts

d) Timetable: class_schedules

e) Communication: announcements, campus_voices, campus_news, admin_banners, conversations, messages, notifications, academic_events

f) Services: assignments, study_materials, leave_requests, placement_drives, library_books, book_requests, clubs

g) System: system_config, backups



5. Admin Application Features
The Admin App is the institutional command center with the following modules:

a) Dashboard (Command Center):
Real-time system health panel showing total students, teachers, classes, and active users, fetched in parallel using Future wait. A Live Audit Stream displays the last five database events via App write Realtime.
Security controls include a Kill Switch (Maintenance Mode) to instantly lock all users out, a Database Pause toggle to halt read/write operations, and a Backup & Restore portal.

b) Student & Teacher Management:
View, search, and filter all students and teachers by year, branch, or department. Approve or reject new registration requests. Edit user details and class/department assignments.

c) Academic Structure Management:
Create and manage departments, branches, classes, subjects, and class-subject assignments. Build the department curriculum and set subject-level regulations and grading templates.

d) Timetable Management:
Create and assign timetable slots for any class, binding each slot to a teacher-subject pair.

e) Announcements & Calendar:
Post global or role-targeted broadcast announcements. Create academic events (exams, holidays, deadlines) that appear in student and teacher calendars.

f) Leaves, Grades, Placements, Library, Clubs:
Review and approve leave requests. Manage grading regulations and view teacher-submitted grade batches. Manage placement drives, library book catalogue, booking requests, and student clubs.

g) Campus Voice & Chat:
View and respond to anonymous student feedback. Direct messaging with students and teachers.

6. Student Application Features
The Student App provides the student's complete academic companion:

a) Registration & Login:
Students register with name, email, roll number, year, and branch. An email verification link is sent automatically. After verification, the admin approves the account. Login supports email-password and OTP (email magic link) for password less access.

b) Dashboard:
Personalized greeting with the student's name, current date, and live attendance percentage. A Quick Scan Attendance button opens the QR scanner directly. The Attendance Progress Card shows percentage with a circular indicator and a status message (warning if below 75%). A live announcements feed, a Quick Action Grid, an admin-customizable dynamic banner, and a Leave portal shortcut complete the dashboard.

c) QR-Based Attendance:
The student opens the in-app QR scanner, scans the teacher's session QR code, and attendance is marked instantly in App write. The system validates the session is active, the time window is open, and the student belongs to the correct class. Duplicate scans are prevented.

d) Academic Modules:
- Timetable: Weekly view with subject, teacher, time, and room.

- Subjects: Enrolled subjects with full syllabus view per subject.
- Study Material: Browse and download teacher-uploaded PDFs per subject.
- Assignments: View pending and past assignments with due dates.
- Grades: Internal marks, midterm scores, grade letters by subject.
- Quiz: Participate in teacher-created timed multiple-choice quizzes.
- Results: End-semester result declarations and exam schedules.
- Calendar: Academic events from admin in a full calendar view.

e) Campus Services:

- Library: Search books, request borrows, track request status.
- Placements: View upcoming drives with company, CTC, and eligibility.
- Clubs: View all student clubs and membership details.
- Campus Voice: Submit anonymous feedback and view admin responses.
- Leaves: Apply for leave, track Pending/Approved/Rejected status.
- Faculty: Directory of all teachers with department details.

f) Productivity My Desk:

- Notes: Create and edit rich-text notes stored locally offline.
- To-Do: Add and manage daily tasks with completion tracking.
- Reminders: Set alarm-based reminders using local notifications.
- Virtual Class: Save and launch meeting links (Google Meet, Zoom).

g) Digital ID Card:

View a digital student ID card showing photo, name, roll number, branch, year, and an embedded QR code, usable as a campus identity document.

h) Profile & Chat:

Edit name, avatar, and contact details. Change password. Direct messaging with teachers and admin via in-app chat.

7. Teacher Application Features

The Teacher App gives educators full control over their teaching workload:

a) Attendance Management (QR-Based Session):

The teacher selects a subject and class, taps Start Session, and a QR code is generated from a unique session token. Students scan the QR with their app. The teacher sees a real-time live list of students who have scanned in and can also manually mark absentees. Sessions have an expiry window to prevent late entries.

b) Assignment & Grade Management:

Create assignments with title, description, subject, and due date. Enter internal marks and midterm scores per student using the admin-defined grading regulation template. Submit grade batches for admin publishing.

c) Quiz Creation:

Create multiple-choice quizzes per subject with questions, answer options, correct answers, time limit, and availability window. View student attempt results after the quiz closes.

d) Study Material Upload:

Upload PDFs and documents to subject-specific folders in App write Storage. Students can browse and download these files from the student app.

e) Analytics:

Class-level performance summaries, attendance rate charts per subject, and grade distribution visualizations using fl_chart.

f) Other Modules:

Timetable view, class rosters, academic calendar, announcements, campus voice response, leave application, library access, clubs, digital teacher ID card, My Desk productivity (notes, todos, reminders), and in-app chat with admin and students.

8. UI/UX Design

All three applications use a premium dark-mode design system defined in the shared AppTheme:

a) Admin App:

Deep dark background (#0A0A0F) with cyan/teal primary accents. Gradient action cards with colored drop shadows. Glass Container (glass morphism frosted effect) for panels. Real-time pulsing status indicators.

b) Student App:
Dark background with Neon Cyan (#00E5FF) as primary accent and Neon Green (#00FF88) for positive metrics like attendance percentage. Circular progress indicators with rounded stroke caps. Horizontal-scroll announcement cards. Mascot loader animation on page transitions.

c) Teacher App:
Consistent with the shared dark theme using teal and amber accents for classroom-specific actions.

d) Shared UI Components:
- GlassContainer: Frosted-glass card effect used across all apps.
- SpringButton: Press-spring animation on all interactive elements.
- NotificationCenter: Unified notification viewer shared by all roles.
- AppTheme tokens: Centralized colors, typography, and spacing.

Material Design 3 (Material You) and Google Fonts are used throughout for modern typography and component styling.

9. Real-Time Synchronization
All three apps use App write Realtime WebSocket subscriptions to receive live updates without manual refresh:

Admin App: Subscribes to profiles, classes, and campus_voices collections. The Live Audit Stream and system health stats update automatically.

Student App: Subscribes to announcements, attendance, admin_banners, and profiles. The dashboard refreshes after every scan or admin action.
Teacher App: Subscribes to attendance_sessions for live student scan-in list.

The pattern used is: subscribe in initState(), listen on the stream, and call ref.invalidate(provider) to trigger a Riverpod-driven UI refresh.

10. Authentication and Security
a) The AuthService handles all authentication operations:
- signIn(): Email + password session via App write Account API.
- sendOtp(): Email token for passwordless magic-link login.
- signUp(): Creates App write account + profiles document.
- sendVerification(): Sends email verification link on registration.
- signOut(): Deletes the current session.
- isLoggedIn: Checks for active session on app startup.

b) Student Registration Security Flow:
Student registers → email verification sent → student clicks link → account.updateVerification() called → admin sees pending account → admin approves (is_active: true) → student gains full access.

c) Admin Security Controls:
- Kill Switch: Sets maintenance_mode = true in system_config collection. All student and teacher apps read this flag on launch and display a maintenance screen, locking them out instantly.
- DB Pause: Halts standard database read/write queries across the apps.

IV. IMPLEMENTATION

The system was implemented in six phases:

Phase 1 Backend Setup:
Created the App write project, configured all 35+ collections with appropriate attributes and indexes, set up the file storage bucket, and registered Android and Web platforms in the App write console.

Phase 2 Shared Package:
Built the AuthService, all data models, Riverpod providers, reusable UI widgets, and the AppTheme color and typography system.

Phase 3 Admin App:
Implemented the Command Center dashboard with real-time statistics, all management portals (students, teachers, classes, subjects, grades, leaves, placements, library, clubs), security controls, and backup.

Phase 4 Student App:
Built the registration and email verification flow, the dashboard with live attendance, QR scanner, all academic modules, My Desk productivity suite, digital ID card, and campus services.

Phase 5 Teacher App:
Implemented QR attendance session generation, grade entry with templates, study material upload, quiz creation, and analytics dashboard.

Phase 6 Deployment:
Built Flutter web release builds, deployed to Firebase Hosting for all three apps, and generated release APKs for Android distribution.

The feature-first modular architecture ensures each feature folder is self-contained with its own screen, logic, and local widgets, making the codebase easy to maintain and extend without cross-feature dependencies.

V. FURTHER SCOPE

The following enhancements are planned for future versions:
1. iOS Platform Support:
Flutter already supports iOS. The apps can be submitted to the Apple App Store with minimal additional platform-specific changes.

2. Offline Mode:
Implement a local SQLite cache for timetable and grades so students can access critical data without an internet connection.

3. Push Notifications:
Server-side push alerts via App write Messaging or Firebase Cloud Messaging for new announcements, leave approvals, and grade releases.

4. Parent Portal:
A fourth application for parents to monitor attendance, grades, announcements, and leave status of their ward.

5. Fee Management:
Integration with Razorpay or Stripe for online fee payment, automated receipt generation, and fee dues tracking within the admin portal.

6. AI-Powered Attendance Alerts:
Predictive alerts to students when their attendance is projected to drop below the 75% threshold based on current trends.

7. Video Lectures:
Teacher-uploaded recorded lecture videos stream able within the student app, integrated with App write Storage or a video CDN.

VI. CONCLUSION

The Student Management System successfully delivers a unified, cloud-connected academic platform for Android and Web that serves administrators, teachers, and students through three dedicated, role-specific applications. By combining Flutter's cross-platform capability with App write's real-time backend, the system eliminates paper-based processes and centralizes all institutional data in one secure, always-synchronized platform.

The key achievements of this project are:

- Three role-specific Flutter apps sharing a common backend and shared package.
- Complete academic workflow: user registration, QR attendance, grading, assignments, quizzes, timetable, placements, library, and campus voice.
- Real-time data synchronization via App writes Realtime WebSockets.
- A secure multi-step registration workflow with email verification and admin approval to prevent unauthorized access.
- A premium dark-mode UI with glass morphism, spring animations, and real-time visual feedback designed for professional-grade usability.
- Deployment on both Android (APK) and Web (Firebase Hosting).

The project demonstrates that a full-featured, multi-role academic management platform can be built effectively using modern open-source technologies and a shared codebase architecture, making it highly maintainable and scalable for real-world institutional use.

REFERNCES

[1] Flutter Documentation. Google. https://flutter.dev/docs
[2] App write Documentation. App write Inc. https://App write.io/docs
[3] Riverpod State Management. Remi Rousselet. https://riverpod.dev

[4] Firebase Hosting Documentation. Google. https://firebase.google.com/docs/hosting
[5] Dart Programming Language. Google. https://dart.dev
[6] Material Design 3 Guidelines. Google. https://m3.material.io
[7] QR Code Standard. ISO/IEC 18004:2015. International Organization for Standardization.
[8] Biørn-Hansen, A., et al. "Progressive Web Apps: The Possible Web-Native Unifier for Mobile Development." Proceedings of the International Conference on Web Engineering, 2017.
[9] Fowler, M. "Patterns of Enterprise Application Architecture." Addison-Wesley Professional, 2002.
[10] Richardson, C. "Microservices Patterns." Manning Publications, 2018.