School Management Application Case Study Task

Problem Understanding & Goal Definition

Purpose of the School App
Provide a single, easy-to-use web application that centralises daily school operations: student records, attendance, timetables, exams & marks, fees, teacher workflows and parent communication. It should give authorised users (admins, principal, teachers, accountants, parents, students) the right views and tools to manage their responsibilities without relying on paper or scattered spreadsheets.

Problems it solves:

Replaces manual/paper processes (student files, attendance registers, fee ledgers).

Reduces communication gaps between school — teachers — parents.

Gives a single source of truth for student data, timetables, exam schedules and fee status.

Simplifies scheduling (timetables/exams) and prevents teacher/class conflicts.

Makes fee invoicing, payment recording and simple accounting tracking easier and traceable.

Goals the app should achieve:

Accurate, secure digital storage of school data (students, teachers, classes).

Fast everyday workflows: attendance marking, viewing timetables, entering marks.

Clear, accessible student progress reporting for parents and students.

Role-based access so each user sees only relevant data and actions.

Responsive and simple UI that non-technical users (teachers, parents) can use on desktop and mobile.

User Role Analysis

User Type	Responsibilities	Expected Features
Admin	System setup and configuration; user & role management; maintain school master data (classes, sections, subjects).	Create/manage users & roles, configure academic year/classes/sections, backup/config settings, view global reports, manage uploads.
Principal	School oversight; review reports and approvals (policies, promotions, important announcements).	View school-level reports/dashboards, approve/authorize policies or exceptions, send announcements, view audit logs.
Teacher	Day-to-day classroom activities:	Mark attendance, enter/edit marks, access class

User Type	Responsibilities	Expected Features
	attendance, marks, assignments, parent communication.	lists/student profiles, upload notes/assignments, message parents, view timetable.
Student	Consume academic info: timetable, marks, attendance, notices.	View personal profile, timetable, attendance summary, exam results/report cards, download study material/announcements.
Parent	Monitor child's progress, fee status and communicate with school.	View child's attendance/marks/timetable, receive fee reminders/receipts, download notices, contact teachers.
Accountant	Fees and receipts management; simple reporting of dues & payments.	Create fee structures/invoices, record payments, generate receipts, view outstanding/received reports, export finance summaries.

Requirement Documentation

Functional Requirements

Student Management

Create, view, update and archive student profiles (personal details, DOB, enrollment no, guardians, contact info).

Assign students to class & section; move students between classes/years.

Upload and store documents (birth certificate, transfer certificate).

Search/filter students by name, enrollment ID, class/section.

Attendance Management

Daily attendance marking per class/section (individual & bulk modes).

Ability to import/export attendance (CSV).

Calculation of attendance percentage per student and historical view by date range.

Automatic absence alerts/notifications to parents (on absence or low attendance).

Timetable Management

Create and maintain weekly timetables per class/section.

Assign teachers to subjects and time slots.

Detect and warn about conflicts (teacher double-booking / room clashes).

Publish timetable to students, parents and teachers.

Exam & Marks Management

Create exam schedules (exam name, dates, applicable classes/subjects).

Define weightages or max marks per subject/exam if applicable.

Enter/modify marks per student & subject; support bulk upload for marks.

Calculate totals/grades and generate report cards; keep audit log of edits.

Fees Management

Define fee structures per class (tuition, transport, misc components).

Generate invoices for students/guardians and mark dues.

Record payments, partial payments and generate receipts.

View outstanding balances, payment history and basic accountant reports.

Teacher Module

Access class lists and student profiles.

Mark attendance and submit marks.

Share assignments, study materials and notices with class/parents.

Request meetings and message parents (in-app communication).

Parent Portal

View child's profile, attendance, marks and timetable.

Receive announcements and fee reminders.

Download receipts and report cards.

Update guardian contact information for notifications.

Non-Functional Requirements

Performance

Responsive UI with fast load/response times for standard school loads (typical pages should render within a few seconds).

Backend APIs should support concurrent daily usage for typical school sizes (hundreds–low thousands of active users).

Security

Role-based access control (RBAC) ensuring least privilege.

Secure authentication (password storage as hashes, sessions and HTTPS).

Sensitive data protected in transit (TLS) and appropriate measures for data at rest.

Scalability

System designed to allow horizontal scaling of backend and read replicas for reporting as load grows.

Modular services (or clear separation of concerns) so heavy reporting tasks don't block core operations.

Usability

Simple, clear UI with mobile-friendly/responsive layouts.

Minimal steps for common tasks (mark attendance, enter marks, create invoice) so non-technical users can adopt easily.

Availability

High availability during school hours; planned maintenance windows outside peak times.

Regular backups (nightly or scheduled) and ability to restore recent data.

System Design

Architecture Planning

Recommended stack options:

Frontend: **React** (SPA) / Angular / Vue — handles UI, client-side routing and forms.

Backend: Node.js (Express) / **Spring Boot** / Django — provides RESTful APIs, business logic, auth and validation.

Database: **PostgreSQL** / MySQL / MongoDB — persistent storage for students, teachers, attendance, fees, etc.

Interaction flow:

User opens the frontend app (React/Angular/Vue) in browser/mobile.

Frontend calls backend REST APIs over HTTPS for all data operations (login, CRUD, reports).

Backend authenticates requests, authorises by role, applies business logic and reads/writes to Database.

Background workers (cron or queue) handle scheduled tasks: fee reminders, email/sms notifications, backups, and heavy reporting.

Static assets (frontend bundle, uploaded documents) served via CDN/object storage for performance.

Database Design

users: user_id (PK), name, email, password_hash, role_id (FK), created_at, last_login

roles: role_id (PK), role_name, permissions (json / bitmask)

students: student_id (PK), enrollment_no, first_name, last_name, dob, class_id (FK), guardian_name, guardian_contact, address, status

teachers: teacher_id (PK), staff_no, name, email, contact_no, subjects (json or relation)

classes: class_id (PK), name (e.g., Grade 10), section, academic_year, class_teacher_id (FK)

subjects: subject_id (PK), name, code, class_id (FK)

timetable: timetable_id (PK), class_id (FK), day_of_week, start_time, end_time, subject_id (FK), teacher_id (FK), room (optional)

attendance: attendance_id (PK), student_id (FK), date, status (present/absent/leave), marked_by (user_id), timestamp

exams: exam_id (PK), name, start_date, end_date, class_id (FK), description

marks: mark_id (PK), exam_id (FK), student_id (FK), subject_id (FK), marks_obtained, max_marks, remarks, entered_by, entered_at

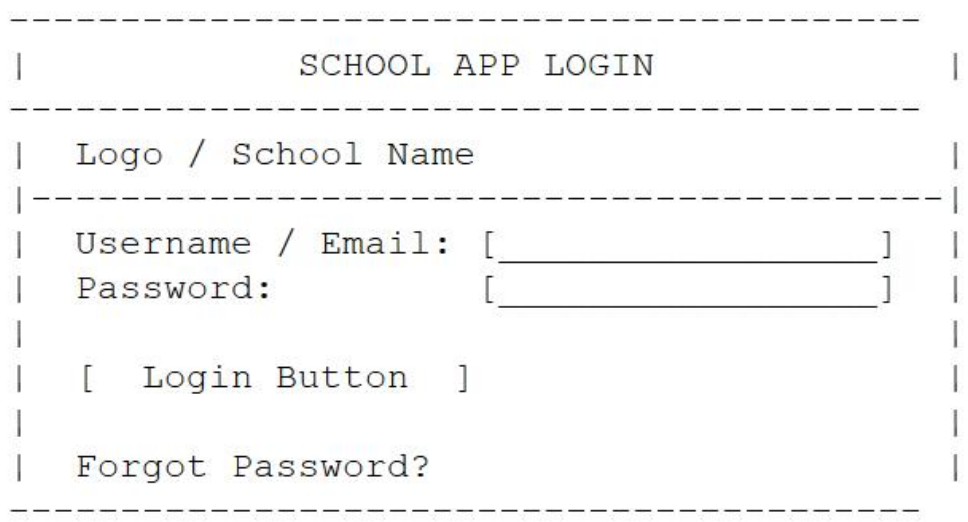
fees: fee_id (PK), student_id (FK), fee_type, amount, due_date, status (paid/unpaid/partial), invoice_no, created_at

payments: payment_id (PK), fee_id (FK), paid_amount, paid_on, payment_mode, receipt_no, recorded_by

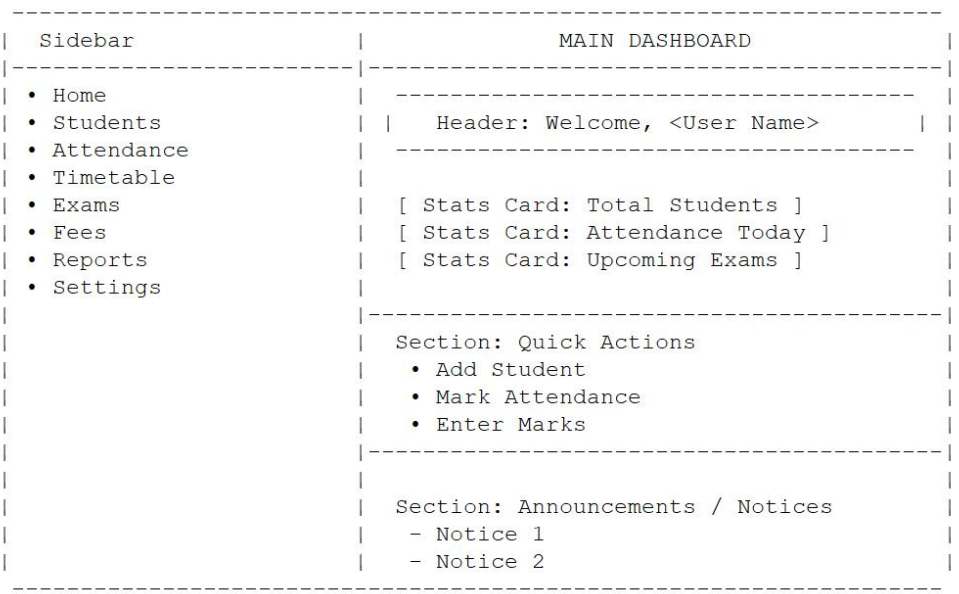
---

Wireframe Sketches (text-based layouts)

Login page



Dashboard (role-specific content)



Student profile page

```
 ------------------------------------------------------------------
|                        STUDENT PROFILE                           |
 ------------------------------------------------------------------
|  [Student Photo]        Name: John Doe                           |
|                         Class: 8-A                               |
|                         Roll No: 21                              |
|------------------------------------------------------------------|
|  Tabs:  | Profile | Attendance | Exams | Fees | Documents |     |
|------------------------------------------------------------------|
|  Profile Details:                                                |
|   • Date of Birth: 12-05-2010                                    |
|   • Gender: Male                                                 |
|   • Contact: 9876543210                                          |
|   • Parent Name: Mr. Doe                                         |
|   • Address: XYZ Street, City                                    |
|------------------------------------------------------------------|
|  Academic Info:                                                  |
|   • Subjects: Math, Science, English, Social Science, Hindi      |
|   • Class Teacher: Mrs. Sharma                                   |
 ------------------------------------------------------------------
```

Attendance screen (teacher view)

```
|------------------------------------------------------------|
|                    ATTENDANCE MANAGEMENT                   |
|------------------------------------------------------------|
|  Class: 8-A        Date: [ 12 / Jan / 2025 ]               |
|------------------------------------------------------------|
|  Roll | Student Name         | Status (P/A)                |
|------------------------------------------------------------|
|  01   | John Doe             | [ P ] [ A ]                 |
|  02   | Riya Sharma          | [ P ] [ A ]                 |
|  03   | Kabir Singh          | [ P ] [ A ]                 |
|  ...                                                       |
|------------------------------------------------------------|
|  [ Save Attendance ]   [ Reset ]                           |
|------------------------------------------------------------|
```
