Internship Project Case Study Task Assignment: School Management Application

Objective

As an intern, you will work on a case study project to design a School Management Application.
This assignment evaluates your ability to analyze requirements, design systems, plan architecture, and think like a professional software developer.

You are not required to build the full system.
Your goal is to research, plan, design, and document the application.

Task Overview

You must create a mini Software Requirements & Design Document for a School Management App based on the details provided.

Your tasks fall into 4 stages:

1. Understand the Problem & Define Goals
2. Identify all user types and their needs
3. Gather and categorize requirements
4. Create system design (architecture + DB + wireframes)

Assignment Tasks (Deliverables)

1. Problem Understanding & Goal Definition

Create a one-page summary describing:

- **What is the purpose of the School App?**

The purpose of the School Management Application is to create a single digital platform that helps schools manage their daily academic and administrative work easily. It aloows the school to store and manage student records, teacher information, attendance, timetable, exams, and fees in one system. The app connects students, teachers, parents, and school staff on one platform so that information can be shared quickly and safely.

What problems does it solve?

Before using a school management system, many schools face these problems:

- Student records are stored on paper, which can be lost or damaged.
- Attendance is taken manually, which is slow and can have mistakes.
- Parents do not get regular updates about attendance, exams, and fees.
- Teachers spend a lot of time preparing results and reports manually.
- Fee records are difficult to track and manage properly.
- Timetables and exam schedules are to share with everyone at the same time.
- The principal and admin do not get proper reports to monitor the school easily.

The School Management App solves all these problems by making everything digital, fast, and well organized.

What goals should the app achieve?

The main goals of the School Management App are:

1. To make school work **paperless and digital.**
2. To help teachers manage **attendance, marks, and students easily.**
3. To allow parents to **track their child's progress, attendance, and fees.**
4. To help the admin manage **students, teachers, and school data properly.**
5. To give the principal **complete control and reports of the school system.**
6. To save time, reduce errors, and improve **communication between school and parents.**
7. To keep all school data **safe and secure.**

Use only the information provided in the research.

2. User Role Analysis

Prepare a table listing:

Each user type

Their responsibilities

Their expected features

User roles to include:

Admin

Teacher

Student

Parent

Accountant

Principal

User Type	User Type	**Expected Features**
Admin	Manages the entire school system, creates users, controls all modules	Add/edit students & teachers, manage classes & subjects, create user accounts, assign roles, view all reports, manage system settings
Teacher	Teaches students, takes attendance, enters marks, manages class activities	Mark attendance, add exam marks, view class timetable, view student profiles, send messages to parents
Student	Attends classes, views personal academic records	View attendance, exam results, timetable, notices, and personal profile

Parent	Monitors child's performance and school activities	View child's attendance, marks, timetable, fee status, receive notifications and messages
Accountant	Manages school fees and payments	Create fee records, collect payments, generate receipts, manage pending dues, view financial reports
Principal	Monitors overall school performance and discipline	View school-wide reports, monitor attendance & results, approve tasks, send announcements

3. Requirement Documentation

A. Functional Requirements

For each module listed below, document what the system should do:

Student Management

Add new student records (name, roll number, class, contact, parent details).

- Edit and update existing student information.
- Delete inactive student records.
- Assign students to classes and sections.
- Search and view student profiles.
- Upload student documents and photos.
- View complete academic history of a student.

Attendance Management

Teachers can mark daily attendance for each class.

- Attendance can be marked as Present, Absent, or Late.
- Parents can view their child's attendance.
- Admin and principal can view full attendance reports.
- Monthly and yearly attendance reports can be generated.
- Absent notifications can be sent to parents.

Timetable Management

Admin can create and update class timetables.

- Assign teachers and subjects to each period.
- Teachers can view their teaching schedule.
- Students can view their class timetable.
- Parents can also view the timetable of their child.
- Changes in timetable can be updated easily.

Exam & Marks Management

Admin can create exam schedules.

- Teachers can enter marks for students.
- System can calculate total marks and grades.
- Students and parents can view exam results.
- Report cards can be generated and downloaded.
- Past exam records can be stored and viewed.

Fees Management

Admin or accountant can create fee structures.

- Generate fee invoices for students.
- Record fee payments (paid, unpaid, pending).
- Parents can check fee status online.
- Fee receipts can be generated.
- Due fee reminders can be sent to parents.
- Monthly and yearly fee reports can be generated.

Teacher Module

Teachers can mark student attendance.

- Teachers can enter and update marks.
- View their teaching timetable.
- View student profiles and performance.
- Send messages or notices to parents and students.
- Apply for leave (if enabled).

Parent Portal

Parents can view their child’s profile.

- Check attendance records.
- View exam results and report cards.
- Check fee status and payment history.
- Receive announcements and notifications.

- Communicate with teachers if required.

Each module should include bullet points of features, as provided in the research.

B. Non-Functional Requirements

Create a section documenting:

Performance

The system should work fast and respond quickly.

- Pages like login, attendance, and results should load within a few seconds.
- The app should handle multiple users at the same time without slowing down.

Security

Only authorized users should be able to access the system.

- Each user must log in using a username and password.
- Different users must have different access rights (Admin, Teacher, Student, etc.).
- Student data, marks, and fee records must be kept private and secure.
- The system should protect data from unauthorized access.

Scalability

The system should support an increase in the number of students, teachers, and users.

- It should work properly even if more classes or sections are added in the future.
- The system should be able to grow as the school grows.

Usability

The system should be easy to use for all users.

- Teachers, parents, and students should be able to understand the system without training.
- The interface should be simple and user-friendly.
- Information should be easy to find.
- **Availability**

The system should be available whenever users need it.

- Parents and students should be able to access the system anytime to check records.
- The system should work properly during school hours without frequent breakdowns.

Use only given details.

4. System Design

A. Architecture Planning

Prepare a diagram or explanation showing:

Frontend (React / Angular / Vue)

Backend (Spring Boot / Node.js / Django)

Database (PostgreSQL / MySQL / MongoDB)

Explain how they interact.

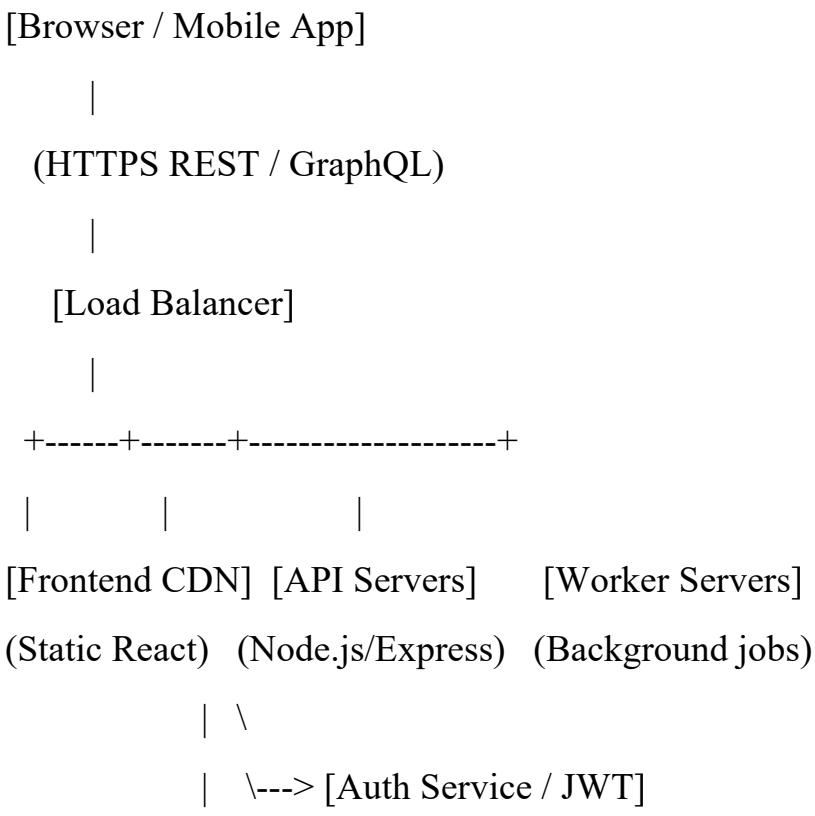
Recommended stack (example)

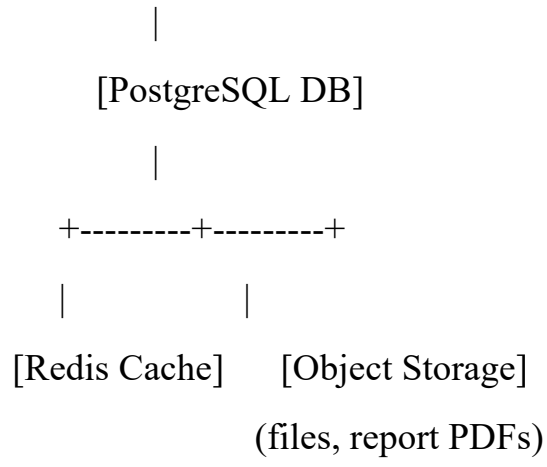
- Frontend: React (web, responsive)

- Backend: Node.js with Express (RESTful API) — alternative: Spring Boot or Django
- Database: PostgreSQL (relational, ACID for student/fee records) — alternative: MySQL
- Other components: Redis (cache + session), Object Storage (S3 or similar) for files, Background Worker (Bull / Celery) for emails/SMS & reports, HTTPS + load balancer

Why this stack (short)

- React: fast, component-based UI, works well for dashboards and mobile-responsive pages.
- Node.js / Express: lightweight, easy to build REST APIs, good ecosystem for background jobs and real-time features.
- PostgreSQL: reliable relational DB suitable for structured school data (students, fees, marks).





Interaction flow (step-by-step, simple)

1. User opens app (React) in browser or mobile web. Static files served from CDN.
2. User logs in → frontend sends credentials to Backend **/api/auth/login** (HTTPS).
3. Backend authenticates (checks PostgreSQL), issues a JWT or session token. Token saved in browser.
4. Frontend calls APIs (attendance, timetable, marks) including token in Authorization header.
5. Backend validates token and checks user role (RBAC).
6. Backend reads/writes structured data in PostgreSQL (students, attendance, fees, marks).
7. Heavy tasks (report generation, bulk notifications, export CSV/PDF) are pushed to Worker Queue and processed asynchronously; results saved in object storage and a link returned.
8. Cache layer (Redis) stores frequently read, non-critical data (e.g., class lists, timetable) to speed responses.
9. Notifications: backend or worker triggers emails/SMS via third-party providers (or internal SMTP/Gateway).

10. Files (student photos, certificates, report PDFs) are uploaded to Object Storage and served via secure links.

Security & infra notes (short)

- Use HTTPS everywhere.
- Store passwords hashed (bcrypt).
- Implement Role-Based Access Control (RBAC) in backend.
- Keep DB backups and use a replication/restore plan.
- Deploy behind a load balancer; scale API servers horizontally.

Alternatives / variations

- Backend: Spring Boot (better for strongly-typed Java shops) or Django (good admin UI).
- DB: MySQL if preferred; MongoDB only if lots of unstructured data (not recommended for this case).
- Realtime: Add WebSocket service if real-time notifications/chats are required.

B. Database Design

Create an ER diagram or table list including:

Students (Main Table)

student_id (Primary Key)

- user_id (Foreign Key)
- roll_number
- student_name
- class_id

- section
- date_of_birth
- parent_name
- parent_contact

Teachers

teacher_id (Primary Key)

- user_id (Foreign Key)
- teacher_name
- email
- phone
- subject_assigned

Classes

class_id (Primary Key)

- class_name (Grade 1, Grade 2, etc.)
- section
- class_teacher_id
- **Subjects**

subject_id (Primary Key)

- subject_name
- class_id

Timetable

timetable_id (Primary Key)

- class_id
- subject_id
- teacher_id
- day
- period_time

Attendance

attendance_id (Primary Key)

- student_id
- class_id
- date
- status (Present / Absent)

Exams

exam_id (Primary Key)

- exam_name
- class_id
- subject_id
- exam_date
- total_marks

Fees

fee_id (Primary Key)

- student_id
- fee_type
- amount
- due_date
- payment_status

Users & Roles

user_id (Primary Key)

- username
- password
- role_id
- email
- phone

You only need to list table names and essential attributes (optional).

C. Wireframe Sketches

Draw or design basic wireframes for:

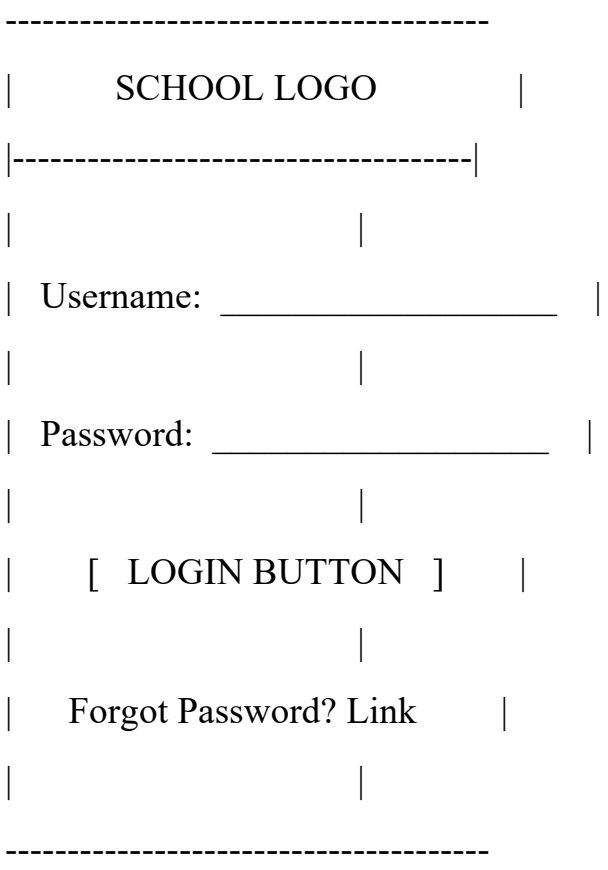
Login page

Dashboard

Student profile page

Attendance screen

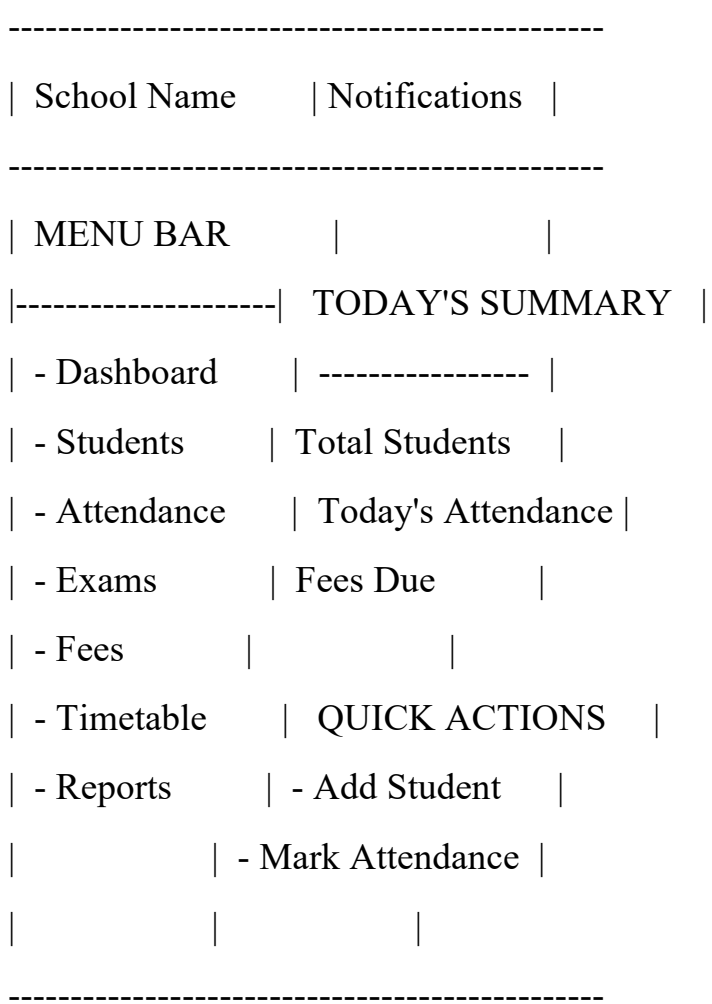
1. Login Page Wireframe



Features on this screen:

- Username input
- Password input
- Login button
- Forgot password option

2. Dashboard Wireframe



Features on this screen:

- Sidebar menu
- Attendance summary
- Student count
- Quick action buttons

- Notifications

3. Student Profile Page Wireframe

```
-------------------------------------------
| Student Photo | Name: Safiya Javed Shaikh      |
|               | Roll No: 26          |
|               | Class: TYBCS         |
-------------------------------------------
| [Profile] [Attendance] [Marks] [Fees]  |
-------------------------------------------
|                                |
|  Personal Details / Attendance Graph  |
|  Exam Results / Fee Status (Tabs)     |
|                                |
-------------------------------------------
|  [  Edit Profile  ]  [ Download Report ]
-------------------------------------------
```

Features on this screen:

- Student photo and basic info
- Attendance tab
- Marks tab
- Fees tab
- Download report card

4. Attendance Screen Wireframe

```
-------------------------------------------------
|  Select Class: [ TYBCS ▼ ]   Date: [ 12-08-25 ] |
-------------------------------------------------
|  Roll No | Student Name | Status              |
|-----------------------------------------------|
|   01     |  Sara    | [ Present ▼ ]      |
|   02     |  Zara  | [ Absent ▼ ]       |
|   03     |  Aqsa  | [ Present ▼ ]      |
-------------------------------------------------
|      [ Mark All Present ]   [ SAVE ]         |
-------------------------------------------------
```

Features on this screen:

- Select class & date
- Student attendance list
- Present/Absent dropdown
- Save button

School Management Application – Workflow Diagram