School Management System - Developer Documentation

# Technologies Used

- Frontend & Backend: Next.js (App Router, Dynamic Rendering)  
- UI Library: shadcn/ui  
- Database: Relational (PostgreSQL recommended)  
- ORM: Prisma  
- Authentication: NextAuth@beta  
- File Upload: Uploadthing / Cloudinary  
- State Management: Zustand or Context API  
- API Communication: REST / Server Actions  
- Other: Tailwind CSS, framer-motion, zod (for validation)

# System Architecture Overview

Client & Server Side Features:  
- Client Side: Forms, page transitions, conditional UI  
- Server Side: API routes, authentication, database operations

# Core Modules (Beat by Beat)

## 1. School Setup

Each School has: Name, Address, Phone, Logo, Email. Managed by Super Admin only.

## 2. Academic Structure

Session contains Terms (First, Second, Third). Transitions between terms and sessions are automated.

## 3. Class & Section Management

Sections like Primary, Secondary. Classes under sections like Primary 1, 2, etc.

## 4. Subject Management

Subjects have name and code. Assigned per class.

## 5. Student Management

Students have personal info, registration number, class, term, session, guardian.

## 6. Guardian Management

Guardians can view child performance, attendance, payment. One guardian → many students.

## 7. Teacher Management

Assigned subjects per class/term. Record CA and exam scores. Head Teacher can take attendance.

## 8. Exams & CA

Each term includes 3 CA and 1 Exam per subject. Teachers record scores.

## 9. Term & Session Transition

Triggered by Admin to move students to next term/session.

## 10. Reporting System

Term reports, session cumulative, class summaries, CA sheets, etc.

## 11. School Fees Management

Payment tracked per term per student. Accessible by Admin, Parent, Student.

## 12. Authentication Flow

Role-based login using credentials (email/phone, PSN number, registration number).

## 13. Security & Logs

Login tracking, role-based access control, audit logs for Super Admin.

# Suggested File Structure

/app  
 /admin  
 /super-admin  
 /teacher  
 /student  
 /parent  
 /auth  
/api  
 /students  
 /guardians  
 /classes  
 /sessions  
 /subjects  
/libs  
 /auth  
 /db  
 /validation

# Additional Notes / Recommendations

- Use middleware for role checks (RBAC)  
- Use Zod for schema validation  
- Implement server actions for form handling  
- Use Prisma Relations for joining models  
- Use PDFKit or Puppeteer to generate signed report sheets  
- Maintain an Audit Table to track role actions