### **1. Profile Module**

**Functionality**:

* **Student Details**: Store and manage personal information of students, such as name, age, class, contact information, and more.
* **Parent Details**: Store and manage parent/guardian details for communication.

**Technical Requirements**:

* **Frontend**: React components for student and parent profiles.
* **Backend**: Node.js/Express for profile creation, update, and retrieval.
* **Database**: MongoDB for storing profile information (student and parent data).
* **Authentication**: JWT for secure access to profile details.

### **2. Classroom Module**

**Functionality**:

* **Homework**: Teachers can assign homework, and students can view and submit homework.
* **Teacher Notification**: Unidirectional communication, where teachers can notify students about syllabus updates or homework.
* **Syllabus**: Teachers can upload or update syllabus details for the class.

**Technical Requirements**:

* **Frontend**: React components for submitting/viewing homework and syllabus.
* **Backend**: Node.js/Express for handling homework and notifications.
* **Database**: MongoDB for homework submissions, syllabus storage.
* **Real-time Updates**: Socket.io for instant teacher notifications.

### **3. Attendance Module**

**Functionality**:

* **Attendance Marking**: Teachers can mark attendance for students.
* **Leave Application**: Students can submit leave applications via the app if absent.

**Technical Requirements**:

* **Frontend**: React for attendance marking by the teacher and leave application submission by the student.
* **Backend**: Node.js/Express for attendance logging and leave application management.
* **Database**: MongoDB to store attendance records and leave applications.

### **4. Notification Module**

**Functionality**:

* **Send Notifications**: Management and Admin can send notifications to individual classes, students, or the entire school.
* **Customizable Alerts**: Admins can customize the type of notifications, including time-based alerts or event-based reminders.

**Technical Requirements**:

* **Frontend**: React components to display and manage notifications.
* **Backend**: Node.js/Express for notification sending and managing.
* **Database**: MongoDB to store notification history.
* **Real-time Notifications**: Socket.io for real-time push notifications to students and teachers.

### **5. Timetable Module**

**Functionality**:

* **View Timetable**: Students can view their class timetable, which is managed by the school.
* **Dynamic Updates**: Admins or teachers can update the timetable, and students can receive updates.

**Technical Requirements**:

* **Frontend**: React component to view the timetable.
* **Backend**: Node.js/Express to handle timetable data updates.
* **Database**: MongoDB to store and manage the timetable data.
* **Real-time Updates**: Use Socket.io for timetable update notifications.

### **6. School Result Module**

**Functionality**:

* **Result Upload**: Teachers can upload student results.
* **Inbuilt Template**: An automatic result generation based on predefined metrics.
* **Student View**: Students can check their results.

**Technical Requirements**:

* **Frontend**: React components for teachers to upload and students to view results.
* **Backend**: Node.js/Express for result management.
* **Database**: MongoDB for result storage.
* **File Uploading**: Multer for handling result file uploads (Excel/CSV).

### **7. Quiz and Result Assessment Module**

**Functionality**:

* **Conduct Quiz**: Edtus can organize quizzes for students.
* **Result Analysis**: Provide a detailed assessment based on the quiz results.

**Technical Requirements**:

* **Frontend**: React components for quiz-taking and result viewing.
* **Backend**: Node.js/Express for quiz creation, management, and result generation.
* **Database**: MongoDB for quiz results and data storage.
* **Real-time Assessment**: Use Socket.io for updating quiz results dynamically.

### **8. Fees Module**

**Functionality**:

* **Fee Submission**: Students/Parents can submit fees via the app.
* **Fee Status**: Users can check the status of submitted fees and view past payment history.

**Technical Requirements**:

* **Frontend**: React for fee submission and status viewing.
* **Backend**: Node.js/Express for fee submission management.
* **Payment Integration**: Stripe or PayPal integration for fee transactions.
* **Database**: MongoDB for transaction history and status tracking.

### **9. App Run Time Module**

**Functionality**:

* **App Usage Tracking**: Parents can check the total runtime of the app by their child, ensuring accountability.

**Technical Requirements**:

* **Frontend**: React for viewing app usage time.
* **Backend**: Node.js/Express for logging app usage time.
* **Database**: MongoDB for storing app run time data.

### **10. School GPT Module**

**Functionality**:

* **Customized GPT**: A customized version of GPT for educational purposes, where students can ask questions and get answers relevant to their curriculum.

**Technical Requirements**:

* **Frontend**: React interface for querying the GPT system.
* **Backend**: Node.js/Express for sending queries to OpenAI’s API.
* **Integration**: Integration with OpenAI API (or similar) for generating GPT responses.

### **11. Counseling Module**

**Functionality**:

* **Raise Query**: Students can raise academic or personal queries.
* **Video Counseling**: The Edtus team provides virtual counseling via Google Meet or other VC platforms.

**Technical Requirements**:

* **Frontend**: React for raising queries and initiating VC sessions.
* **Backend**: Node.js/Express to manage queries and schedule counseling sessions.
* **VC Integration**: Google Meet API for video conferencing.

### **Technology Stack Summary:**

* **Frontend**: React.js (for all user interfaces and interaction features)
* **Backend**: Node.js with Express.js (for all API logic and handling requests)
* **Database**: MongoDB (for storing user data, attendance, homework, notifications, etc.)
* **Real-time Communication**: Socket.io (for notifications, updates, and chats)
* **Authentication**: JWT (for secure access control)
* **Payment Integration**: Stripe or PayPal (for fees module)
* **External API Integration**: Google Meet API, OpenAI GPT API (for counseling and school GPT)