

Assignment: Interactive Presentation Platform (React + Node.js)

Overview

Build a simplified interactive classroom presentation platform where a **teacher presents slides** and **students join in real time** to view the slides and participate in activities embedded within the presentation flow.

The system should support **PDF/PPT upload**, convert slides into images, and allow **activities (MCQ / open-ended)** to appear at specific slide positions.

Target age group: **5–14 years**

Tech Stack (Mandatory)

- **Frontend:** React (preferred with hooks)
 - **Backend:** Node.js + Express
 - **Storage:** Local / Cloud (Aleesha's choice)
-

Core Requirements

1. Presentation Upload & Processing

- Teacher can upload a **PDF or PPT** file.
 - Backend should:
 - Convert the presentation into **slide images**.
 - Maintain slide order.
 - Frontend should display slides as images.
-

2. Activity Insertion

- Teacher can configure activities at specific slide numbers.
 - Supported activity types:
 - **MCQ** (single correct answer)
 - **Open-ended question**
 - Example:
 - After slide 3 → show an activity
 - After Activity completes → resume slides
-

3. Live Session Flow (Teacher ↔ Student Sync)

Session Creation

- Teacher starts a session.

- System generates a **random 6-character session code**.
- Students join using this code.

Synchronization Rules

- Teacher controls slide navigation.
 - Students automatically stay in sync with the teacher's slide.
 - When teacher moves to the next slide:
 - Students move to the same slide.
 - No manual slide control for students.
-

4. Activity Experience

Student Side

- When an activity slide is reached:
 - Slide view is replaced by a **full-screen activity view**.
 - Student submits answer (MCQ or text).
 - Student waits after submission.

Teacher Side

- Teacher can:
 - See live student responses.
 - View aggregated results.
 - Click **"Show Results"** to reveal outcomes to students.



Flow Control

- When teacher clicks **Next Slide**:
 - Activity closes automatically for all students.
 - Presentation resumes from the next slide.
-

5. Real-Time Communication

- Sync slides
 - Trigger activity start/end
 - Send student responses
 - Broadcast results
-

Bonus (Plus Points – Optional)

-  **AI Integration**
 - Auto-generate MCQs or open-ended questions from slide content
 - AI-based answer evaluation or summary
 -  Child-friendly UI design
-

Deliverables

- GitHub repository (frontend + backend)
 - Demo-ready implementation (even if partial ready)
-

Evaluation Criteria

- Real-time synchronization logic
 - UX clarity (teacher vs student)
 - Bonus features (if implemented)
-

Time Expectation

This assignment is expected to take **2 days**.

Partial completion with good design decisions is acceptable.
