IMPORTANT NOTE: Unless mentioned below, all assumptions are open to your interpretation. Please make your own assumptions and build this system. We do not entertain any queries or clarifications associated with the problem statement below. As a team, you are open to drawing your own boundaries. Your grade has no impact on incompleteness of your assumptions.

Problem Code: VidyaVichar

Problem Statement: **VidyaVichara** is a classroom Q&A sticky board where students can post questions in real-time during lectures. Questions appear as colorful sticky notes that instructors can view, mark, and organize. The system must be built using the **MERN stack**, where React handles the interactive frontend, Express + Node.js manages the backend logic, and MongoDB stores all questions, statuses, and timestamps. This ensures questions are persistently saved, retrievable across sessions, and available for later review and analytics.

Skills: Your team will be assessed on Git proficiency, database design (SQL & NoSQL), full-stack development with MERN, and problem-solving in handling complex data relationships and workflows.

Deliverables:

Part 1: VidyaVichara Web – MERN (70%)
Build a responsive React-based interface where questions can be posted during lectures and displayed as sticky notes. The board should allow instructors to mark questions as answered or important, filter them (such as showing only unanswered ones), and clear them when needed. Smooth user interactions, validation to prevent empty or duplicate questions, and a clean interface are key expectations of this part.

Part 2: VidyaVichara Database (30%)
The database should store details such as question text, author, status, and timestamp, enabling questions to be saved across sessions and reviewed later.

Example: During a lecture on **System Design principles** in the *Software System Development* course, the instructor is explaining concepts like load balancing, caching, and database sharding. Students use **VidyaVichara** to post their questions in real-time without interrupting the flow of teaching. One student asks *"What's the difference between horizontal and vertical scaling?"*, another posts *"Can caching introduce consistency issues?"*, and a third asks *"How does CAP theorem apply to microservices?"*.

All these questions appear as sticky notes on the instructor's board. The instructor can quickly filter to view only unanswered questions and mark questions as "answered" once addressed. At the end of class, the full set of questions (along with timestamps and statuses) is stored in the **MongoDB backend**, allowing the teaching assistant to review them later and prepare additional clarifications or study resources.

Submission Criteria:

- **Source Code:** Git repo link with all frontend and backend code (commit history reviewed).
- **README.md** with MERN implementation details and set-up steps along with short details on design decisions, solution diagram.
- **Submit Peer Review Form**: https://forms.office.com/Pages/ResponsePage.aspx?id=vDsaA3zPK06W7IZ1VVQKHFzW4INMf2JMjyL9qPnlPbNURjRUSDA0QURKWFBCTzVCNFdDU1pDVTJZSy4u
- **LLM Usage Form**: https://forms.office.com/Pages/ResponsePage.aspx?id=vDsaA3zPK06W7IZ1VVQKHFzW4INMf2JMjyL9qPnlPbNUNFFSOTM3MDE4T0NPWUpNOTZKSFFQSUJTRC4u