**Q: 1**

**Collaborative Learning Management System (LMS)**

You are required to build a **Collaborative Learning Management System (LMS)** using the **MERN stack**, where multiple instructors and students can interact in real-time within courses. The system should include user roles, course management, lessons, comments, and live activity feeds.

**Note:** You may use AI tools to assist in parts of the solution (e.g., authentication, socket setup, UI snippets), but **not to build the entire solution end-to-end**. This evaluation is designed to test your ability to integrate multiple complex functionalities logically.

**Functional Requirements**

**1. Authentication & Authorization**

* Use **JWT-based authentication**.
* There must be **two user roles**:
  + **Instructor**: Can create, edit, and delete their own courses, upload lessons, and manage students.
  + **Student**: Can enroll in courses and interact with lessons (comment, mark completed, etc.).

**2. Course Management**

* Instructors can:
  + Create multiple **Courses** (e.g., “React Basics”, “Data Structures”).
  + Add or remove **Lessons** (video URL, title, description).
  + Invite students via email or username.
* Only the instructor who created the course can delete it.

**3. Lesson Management**

* Each course contains multiple lessons.
* Each lesson includes:
  + title
  + description
  + videoUrl
  + resources (optional links or files)
* Lessons can be **reordered (drag-and-drop)** by instructors (optional).
* Students can **mark lessons as complete**.

**4. Real-Time Features**

* Use **Socket.IO** (or similar) for **real-time** updates:
  + If an instructor adds or reorders a lesson, all enrolled students see the update instantly.
  + Real-time **comment system** under each lesson where both instructors and students can comment live.

**5. Comments**

* Each lesson supports threaded comments (i.e., replies).
* Each comment includes:
  + user
  + message
  + timestamp
* Deletion rules:
  + A user can delete only their own comment.
  + An instructor can delete any inappropriate comment in their course.

**6. Search, Filter & Progress Tracking**

* Students can:
  + Search lessons by **title**.
  + Filter lessons by **completion status**.
* Show a **progress bar** for each course based on completed lessons (optional).

**7. Activity Feed (Notifications)**

* Every major action must be logged and shown in a real-time feed:
  + Lesson added, deleted, or reordered.
  + Comment added or deleted.
  + Student enrolled or unenrolled.
* Feed entries include:
  + Action description
  + Actor (user)
  + Timestamp

**8. Pagination & Performance**

* Courses and lessons must support **pagination** (e.g., load 10 lessons per page).
* Lazy load videos and comments to improve performance.

**Backend Validation & Constraints**

* A student can **only view lessons** of courses they are enrolled in.
* Only the course’s **instructor** can modify lessons.
* The system must validate user roles before allowing actions like lesson reorder or comment deletion.
* All database operations must be secure and validated at the backend level.

**Tech Stack**

* **MongoDB** – Database
* **Express.js** – Server framework
* **React.js** – Frontend
* **Node.js** – Backend runtime
* **Socket.IO** – Real-time updates
* **JWT** – Authentication
* **Multer or Cloudinary** (optional) – For file uploads (course thumbnails or resource files)

**Frontend Requirements**

* Responsive, clean UI (you may use any library like **Bootstrap**, **Material UI**, or **Tailwind**).
* Key screens:
  + Login / Signup
  + Instructor Dashboard
  + Course Page (with lessons)
  + Lesson Page (video + comments)
  + Student Dashboard (progress & enrolled courses)

**Submission Guidelines**

1. Push the entire code into your **Masai GitHub repo** under a folder named collaborative-lms.
2. Submit **only that folder link**, not the full repo.
3. Ensure:
   * Project runs without errors.
   * Socket and backend integrations work correctly.
   * Proper role-based checks are implemented.

**Important Notes**

1. **Start your application on your own.**
2. **Strictly do not give the entire question to AI.**
3. You may take help from AI, but **only in chunks/parts**, not for the whole solution at once.
4. Ensure your **internet connection is stable throughout the contest**.
5. **No matter what the reason is**, if you stop screen sharing or get disconnected, you will be **disqualified**.
6. Please check your **internet connection, laptop condition, and electricity backup** before attempting the contest.
7. **Under no circumstances will any disconnection or technical issue be accepted as a valid reason.**