ReactJs Practical Exam (2 hours)

Server Url: https://jsonplaceholder.typicode.com
Use **JSONPlaceholder** + **local mock state** for moderation state.

Below are the Task details which need to be completed in the given time.

- **Project Title => Social Feed + Comment Moderation Dashboard**
- **You are building a mini social platform where:**
  - Users can browse posts
  - Load comments on demand
  - Add/edit/delete posts & comments
  - Moderate comments (approve/reject)
  - Handle pagination, search, caching
  - Manage optimistic updates & rollback
  - Handle nested UI state cleanly

🧩 Core Features

Area	Requirement
📜 Posts Feed	Show list of posts with pagination / infinite scroll
🔍 Search	Search posts by title (debounced)
💬 Comments	When post clicked, load comments **lazy loaded**
➕ Add	Add post & comment
✏️ Edit	Edit post & comment
❌ Delete	Remove post & comment
✅ Moderation	Comments have states: *pending*, *approved*, *rejected (optional)*
⚡ Optimistic UI	For post & comment actions
🔄 Retry	Failed update should show retry option
💾 Persistence	Store approved comments count locally

📊 Sorting	Sort posts by: latest / most commented / alphabetical

🧠 Business Rules

✅ Comments default to `pending`
✅ Moderator can `approve` or `reject` comment(optional)
✅ Approved comments count visible per post
✅ Rejected comments not visible in feed
✅ Search applies to posts only, not comments
✅ Max comment length = 200 chars
✅ Loading comments only when needed (lazy load)

📤 Deliverables

- GitHub repo
- README:
  - Architecture explanation
  - Key trade-offs
  - How caching is designed
  - Improvements if more time
- Optional: 2-min Loom walkthrough

- **Authentication**
  - Mock a simple login page, store accessToken in local storage and authenticate using below credentials and need to pass access token in headers in all api requests.
  - username: testuser@logicwind.com
  - password: Test123!
  - We just need to mock Authentication without API call and all pages should be accessible only after authentication, if token is not found then should go back to login

- **NOTES**:
    - Use reusable UI components, Form components.
    - Use react Router for routing and pages as needed
    - Use Context API for managing global states
    - Use any design library of your choice (Antd preferable)
    - Proper navigation should be working.
    - Basic proper ui, with proper layout & paddings.
    - Proper folder structure in the code following the coding standards.
    - Use the appropriate apis for getting the data as per your convenience.
    - Focus on Scalable architecture, State mastery, performance optimizations, code should showcase the thinking ability as a senior DEV.