Exam Portal – Secure Login & User Authentication System

Core Goal

To authenticate users securely on an exam portal, upload the login user's photo, track every login attempt, and automatically **restrict the user account after more than 4 failed login attempts**.

Technology Stack

- **Frontend:** React.js
- **Backend:** Express.js (Node.js)
- **Database:** SQL (MySQL / PostgreSQL / SQLite)

Database Entities & Fields

1. users

- user_id (BIGINT, Primary Key, Auto-increment)
- full_name (VARCHAR(255), NOT NULL)
- email (VARCHAR(255), NOT NULL, UNIQUE)
- password (VARCHAR(255), NOT NULL)
- profile_image (VARCHAR(255), NULL)
- failed_attempts (INT, DEFAULT 0)
- is_locked (BOOLEAN, DEFAULT FALSE)
- created_at (DATETIME, DEFAULT CURRENT_TIMESTAMP)
- updated_at (DATETIME, DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP)

---

2. login_attempts

Stores audit logs for every login attempt.

- attempt_id (BIGINT, Primary Key, Auto-increment)
- user_id (BIGINT, Foreign Key → users.user_id, NULL allowed)
- ip_address (VARCHAR(50), NOT NULL)
- status (ENUM: 'SUCCESS', 'FAILED', 'LOCKED')
- attempted_at (DATETIME, DEFAULT CURRENT_TIMESTAMP)
- user_image (VARCHAR(255), NULL)

Backend Auto-Restriction Logic

• When a user attempts to log in:

- Validate email and password.
- If password is incorrect:
- Increment failed_attempts by 1.
- Insert a FAILED record into login_attempts.

• If failed_attempts > 4:

- Set is_locked = TRUE.
- Block further login attempts.
- Insert a LOCKED record into login_attempts.

• If login is successful:

- Reset failed_attempts = 0.
- Set is_locked = FALSE.
- Save uploaded user photo
- Insert a SUCCESS record into login_attempts.

Important:
All validation, attempt counting, and lock logic must be handled **only on the backend**.

Key Features

• Upload user photo during login
• Track every login attempt with IP & device info
• Automatic account lock after 4 failed attempts
• Reset attempts on successful login

Form Validation

Login Form (Frontend)

• email:

- Required
- Valid email format

• password:

- Required
- Minimum 6 characters

• user_image:

- Optional
- Image format only (jpg, png, jpeg)

Responsive UI with Modals

• Login page with clean UI
• File upload field for user photo
• Error modal for locked account
• Warning message showing remaining attempts

Instructions for Task Submission Before submitting your task:

1. Show your completed task in the group meeting to get initial feedback. After completing your assigned task:

2. Create two separate GitHub repositories:

• Frontend (FE): Contains all React code, components, and UI files.

• Backend (BE): Contains all Express/Node.js code, APIs, and database logic.

3. Make sure both repositories are publicly accessible.

4. Do not reuse existing repositories — create new ones for this assignment.

5. Do not commit node_modules or any dependency folders. Use a proper .gitignore file.

6. Push your code to the respective repositories.

7. Copy the GitHub links for both FE and BE repositories.

8. Submit both links using this Google Form: https://forms.gle/9cWBHnYHxvNcmtECA
Ensure both repositories are complete, properly structured, and published before

submitting. Incomplete, private repositories, or repositories with node_modules committed cannot be evaluated.