**Online Examination Module**

**Objective:** You are tasked with developing a full-stack application to conduct online examinations that include MCQ and programming sections. Below are the detailed specifications and requirements for the project.

**Sections:**

1. **MCQ (Multiple Choice Questions)**
2. **Programming**

**Must-Have Features**

**MCQ Section:**

**For Admin / Examiner:**

1. **Student Profile Management:**
   * **Create Student Profiles:**Admins should be able to create and manage student profiles, which include essential data such as name, email, student ID, and other relevant information.
2. **Question Management:**
   * **Enter Questions:**Admins can enter question text into a text area.
   * **Categorize Questions:**Questions should be categorized into three categories: Logical (Aptitude), Technical, and Programming.
   * **Options for Questions:**Each question can have between 2 to 4 options.
   * **Input Options:**Admins should be able to enter text for each option.
   * **Correct Answer:**Admins should specify the correct answer for each question.
3. **Exam Management:**
   * **Create Exams:**Admins should be able to create a new exam by selecting a number of questions from each category.
   * **View Results:**Admins should be able to view summaries and detailed results of all completed exams, with filtering options for easier analysis.
   * **Set Passing Criteria:**Admins should set passing criteria and identify students who have passed each exam.

**For Examinees:**

1. **Question Navigation:**
   * **Answer Selection:**Examinees can select one answer from the available options.
   * **Change Answers:**Examinees can change their answers as long as the exam hasn’t been submitted.
   * **Navigate Questions**: Examinees should be able to navigate back and forth through the questions.
2. **Exam Submission:**
   * **Auto-Submit:**The test should auto-submit when the set time expires.
   * **Prevent Copy-Paste:**The system should prevent copy-pasting of questions and answers.

**Programming Section:**

**For Admin / Examiner:**

1. **Programming Question Management:**
   * **Enter Programming Questions:**Admins can enter programming questions and reference answers. Note that these answers are for reference only and are not used for automatic code comparison.
   * **Select Programming Questions:**Admins can select the number of programming questions for each exam.
   * **Assign Difficulty Levels:**Admins should assign a difficulty level to each programming question.
2. **Exam Management:**
   * **Same Functionalities as MCQ:**The same functionalities as the MCQ section for managing exams apply here.

**For Examinees:**

1. **Programming Environment:**
   * **Answer Input**: Examinees can write answers in a programming editor where copy-paste (only from inside the editor) is allowed.

**Nice-to-Have Features:**

**For Admin / Examiner:**

1. **Advanced Question Management:**
   * **Add Difficulty Levels:**Admins should be able to add difficulty levels to each question.
   * **Design Tests by Difficulty and Category:**Admins should be able to design tests based on difficulty level and category, using an algorithm to auto-pick questions to maintain an overall exam difficulty level.
   * **Add Images:**Admins should be able to add images in the question and answer boxes.
2. **Network and Exam Continuity:**
   * **Network Disconnection Detection:**Detect network disconnections and pause the exam. Resume the exam with re-entry of credentials while maintaining the exam status.
   * **Prevent Intentional Pausing**: Implement innovative ways to prevent students from intentionally pausing exams.
   * **Login Logging**: Log each login attempt.
   * **Browser Closure Warning:**Provide a warning before the browsercloses and maintain the exam status if the browser is accidentally closed.

* **Single Device Enforcement:** Ensure exams start and end on the same computer by detecting IP or MAC address.
* **Browser Tab Monitoring:** Pause the exam if the browser tab is changed or minimized. Disable certain keys (Ctrl, Shift, Tab, Alt, Windows key).

1. **Security and Monitoring:**
   * **Periodic Screenshots:**Take periodic screenshots of the screen and pictures of the user during the exam to monitor activity.
   * **Exam Continuity:**Ensure seamless resumption of exams after interruptions.
2. **Exam Interface:**
   * **Question Status Display:**Display answered and unanswered questions to the user and facilitate easy navigation to specific questions.
3. **Data Import:**
   * **Import Student Data:**Admins should be able to import student data from an Excel sheet.
   * **Import Questions:**Admins should be able to import questions from an Excel sheet.

**For Examinees:**

1. **Programming Environment:**
   * **Proper Editor:**Use a proper editor that supports limited programming languages with auto-completion.
   * **Compile and Run:**Allow programs to be compiled and run within the editor.

**Other Instructions:**

**Technologies:**

* **Backend:**
  + Develop the API using either .NET (ASP.NET Core) or Java (Spring Boot).
* **Frontend:**
  + Use React.js for building the front end.
  + You are permitted to use third-party open-source themes and libraries to enhance the UI.
* **Database:**
  + Utilize SQL Express for database management.
* **Version Control:**
  + Host the project on GitHub with a clearly visible commit history.
  + Ensure daily (or frequent) commits to maintain a consistent development flow.
  + Develop each feature in separate feature branches to maintain clean and organized code.