Task: Build an Advanced Quiz Application

Objective: The goal of this task is to assess your ability to manage dynamic data, handle user input, implement conditional logic, apply basic algorithms within a React application.

Requirements:

1. **Create React App:** Set up a new React application using Create React App or any other preferred method.
2. **Quiz Data:** Create a JSON file or an array of objects containing quiz questions. Each question object should include:
    - Question text
    - Multiple choice options
    - Correct answer
    - Difficulty level (easy, medium, hard)

    Example:

    ```
    [
      {
        "question": "What is the capital of France?",
        "options": ["Berlin", "Madrid", "Paris", "Rome"],
        "correctAnswer": "Paris",
        "difficulty": "easy"
      },
      // Add more questions...
    ]
    ```

3. **Quiz Component:** Create a `Quiz` component that dynamically renders questions and their multiple-choice options.
    - Implement a randomized order of options for each question.
    - Display one question at a time.
4. **Next Question Button:** Include a "Next Question" button that advances to the next question after the user selects an answer.
    - Disable the button if no answer is selected.
5. **Scoring System:** Implement a scoring system based on the difficulty level of the questions.
    - Easy: 1 point
    - Medium: 2 points
    - Hard: 3 points
6. **Timer:** Add a timer for each question. The user should answer within a specified time (e.g., 20 seconds).
    - If the timer runs out, consider the question as unanswered.
7. **Finish Screen:** After the user completes all the questions, display a summary screen showing the user's score, the time taken, and a message (e.g., "Great job!" or "You can do better!").
8. **Styling:** Style the application to make it visually appealing. Focus on clarity and user-friendly design.
9. **Logic and Event Handling:** Ensure that the logic for rendering questions, handling user input, calculating the score, and managing the timer is well-implemented.
10. **Algorithms:**

- Use basic algorithms to implement the randomization of answer options.
- Consider efficient data structures for managing the quiz data and state.

11. **Version Control with Git:**

- Initialize a Git repository for your project.
- Commit your changes incrementally, providing meaningful commit messages.

12. **Documentation:**

- Add comments where necessary to explain complex logic or functionality.
- Provide a brief README file explaining how to run the application locally.