Live Quiz Competition MVP – 2 Hour Take-Home Test

Overview

Build a minimal, functional MVP for a live quiz platform with an admin panel. Your goal is to implement real-time quiz functionality while designing user flows that make logical sense and enhance the overall experience for both participants and the host. We're less focused on UI aesthetics and more interested in your UX process and product thinking.

Time Expectation

This test is designed to be completed in approximately **2 hours**.

Requirements

1. User Quiz Interface

Example:
https://www.figma.com/design/dcYeJnZSbrNpgfcP6CjPvB/AlgoEd-Take-Home-Test?node-id=0-1&t=Oa8iIJqlQnHilqZq-1

- **Quiz Flow:**
    - Present one question at a time with multiple-choice answers.
    - Include a basic countdown timer for each question.
    - Provide immediate feedback (correct/incorrect) after an answer is submitted.
    - Update and display a running score.
    - You can use this json file for questions and answers: https://drive.google.com/file/d/1BnwTRwDl73ApCn50ZSMJa00RwPeHF6Hy/view?usp=sharing
- **UX Focus:**
    - Design the flow so that a participant can understand how the quiz progresses.

    - Ensure error states (e.g., missed answers, late submissions) are clearly handled.

2. Admin Panel (Optional, nice to have)

- **Host Controls:**
    - Allow the host to initiate the quiz, progress through questions, and monitor participant scores in real time.
    - Ensure the flow in the admin panel is logical and supports smooth quiz management.

3. Real-Time Updates

- **Live Functionality:**
    - Implement real-time communications to:
        - Push quiz question changes to participants.
        - Update scores as answers come in.
    - No custom backend code required — can just read/write to Firebase.

4. Basic Anti-Cheat Mechanism (Optional, nice to have)

- **Simple Measure:**
    - Apply a basic anti-cheating technique, such as randomizing the answer order or limiting the answer submission window.

Product & UX Component

Alongside your code, include a brief explanation covering:

- **User Flow Explanation:**
    - Describe the user journey for participants and the host. Explain how your flows facilitate a smooth experience.
- **Feature Rationale:**

    - Why did you choose your specific approach for quiz progression and error handling?

- **Future Enhancements:**

    - What improvements or additional features could be incorporated to enhance the experience in a full-scale product?

Technical Guidelines

- **Tech Stack:**

    - Use a stack you're comfortable with (AlgoEd currently uses Angular-Node stack but we could also switch to React-Node in the future)
- **Implementation:**

    - It is acceptable to combine front-end and back-end in one project.

- **UI/UX:**

    - Focus on clean, functional flows over visual design details.
- AI
    - Feel free to use AI assistance
    - Please share the prompts you used (in a text file)

Deliverables

1. **Source Code:**

    - Provide a runnable codebase through a Git repository.
    - AI prompts, if any

2. **README:**

    - Include setup instructions.

    - Provide a brief document explaining your user flows, design decisions, and ideas for future enhancements.

Submission Instructions

- **Deadline:** Submit your solution within 2 hours.
- **How to Submit:**
    - Provide a link to your Git repository.
    - Include a README with installation instructions, flow explanations, design rationale and time taken.