Elicitation

Josh Tran: joshtran013@gmail.com
Cam Wang: cammryford1@gmail.com
Bobby Cho: bobcho365@gmail.com

What are some limitations with quiz tools that you have encountered?
Josh: "There was a time when I did a quiz that felt way too difficult for my level. I felt discouraged, and I didn't want to keep going. "

Cam: "Most quiz tools that I've used don't let me communicate with the quiz creator. Sometimes, I want to point out some areas for improvement, but the tools don't have that option."

Bobby: "I remember being part of a quiz that was overrun by bots. It was obvious when random answers started flooding in, and it completely derailed the session."

Can you recall any recent experiences with a quiz where you felt frustrated? What specifically bothered you?
Josh: "Yes, I was stuck on a question that didn't make any sense, and there was no option to get clarification. It felt really frustrating to be left without any support."

Cam: "There was a quiz where multiple questions were confusing, but I had no way to share my thoughts with the quiz creator."

Bobby: "As an admin, I find it frustrating when I can't stop automated systems from interfering with quizzes. It makes it harder to create a meaningful and authentic experience for my real participants."

What makes you decide against using a particular quiz tool?
Josh: "If there's no support when I struggle, I'm less likely to use the tool. I need features that help me stay engaged rather than feeling stuck."

Cam: "I avoid tools that don't value user feedback. I enjoy interactive experiences, and if I can't contribute, I just don't find it worthwhile."

Bobby: "If I can't trust the platform to block bots, I won't use it. My goal is to provide a fair and fun experience for people, and I need quiz tools that ensure real participants are the focus."

How do quiz tools fall short when it comes to personalised learning?
Josh: "If quiz tools offered help when I struggled with a difficult question, it would make the experience feel more personalised. Without that, it can become frustrating and leave me feeling stuck, especially when I'm trying to learn.

Cam: "Sometimes, after a quiz, I'd like to leave feedback so that I can let the tutor know that I am struggling with certain topics, and receive guidance to improve. "

Bobby: "When bots can access the quiz, it becomes harder to personalise. I can't create meaningful content for my participants if automated responses skew the experience."

Brief Draft Solutions:

1. Josh: Implement a request hint feature for quiz questions.
2. Cam: Implement a send feedback feature at the end of a quiz.
3. Bobby: Implement CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) Verification for quiz sessions.

User Story 1 (addressing Josh):

As a quiz participant, I want to request a hint if I get stuck on a question so that I can keep making progress without getting frustrated.

User Acceptance Criteria:

- **Hint Availability:**
    - Each question displays a "Request Hint" button that becomes active after the question has been visible for at least 10 seconds.
- **Hint Usage Limit:**
    - Participants can request only one hint per question. Once a hint is requested, the "Request Hint" button is disabled.
- **Hint Display:**
    - Hints appear in a pop-up box below the question and are limited to 100 characters.
- **Hint Tracking:**
    - The system records each instance when a hint is requested and provides this information in the quiz admin's report for further analysis.

Use Case:

Use Case Background

- Use Case: Request Hint on Quiz Question
- Goal in Context: The participant requests a hint when stuck on a question, helping them make progress and improve their chance of answering correctly.
- Scope: Quiz system

- Preconditions: The participant is viewing a quiz question with the option to request a hint.
- Success End Condition: The participant successfully receives a hint to assist with answering the question.
- Failed End Condition: The participant either does not receive a hint due to time restrictions or already having used the available hint.
- Primary Actor: Quiz participant
- Trigger: The participant encounters a question they need help with.

Use-Case List

1. System: Displays a “Request Hint” button with each question.
2. System: Disables the “Request Hint” button for the first 10 seconds after displaying the question to encourage independent problem-solving.
3. System: After 10 seconds, enables the “Request Hint” button if a hint is available for the question.
4. Participant: Clicks the “Request Hint” button.
5. System: Checks if the participant has already requested a hint for this question.
    - If yes, displays “Hint already shown” and prevents further hint requests.
    - If no, proceeds to the next step.
6. System: Displays the hint in a pop-up box below the question, limited to 100 characters, with a “Close” option to hide the hint.
7. Participant: Views and optionally closes the hint.
8. System: Records the hint request and displays “Hint shown” to indicate that the hint was provided.
9. System: Disables further hint requests for the question.

Validation:

Josh’s Comment: *"The hint feature looks great. It’s easy to understand and gives participants just the right amount of help when they’re stuck. I think it will make the quizzes much more enjoyable."*

API Documentation: Admin Add Hint to Quiz Question

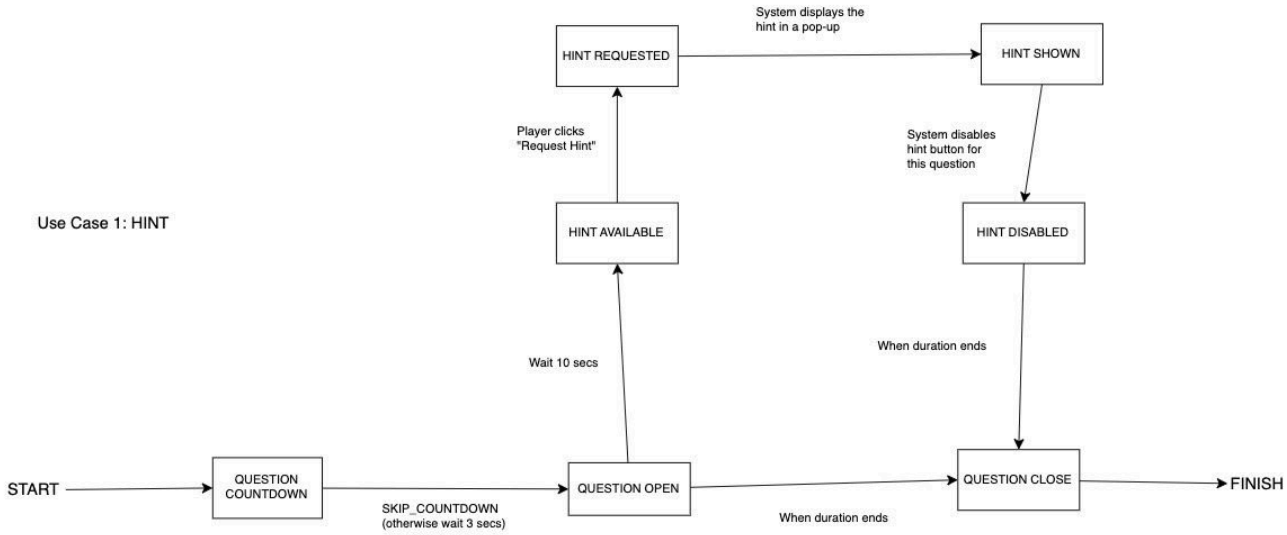
- **Endpoint:**
  POST /v2/admin/quiz/{quizid}/question/{questionid}/hint
- **Description:**
  Adds a hint to a specific quiz question to assist participants.
- **Parameters:**
    - Path Parameters:
        - quizid: The unique ID of the quiz (Integer, Required)
        - questionid: The unique ID of the question (Integer, Required)
    - Headers:

        - token: The admin's authorization token (String, Required)
    - Request Body:
        - hint: The hint text (String, Required, 1 to 100 characters)
- **Example Request Body:**
    - { "hint": "Use the triangle inequality" }
- **Responses:**
    - 200 OK: Hint successfully added
        - { "message": "Hint successfully added to the question." }
    - 400 Bad Request: Invalid hint length (less than 1 or greater than 100 characters)
        - { "error": "Hint must be between 1 and 100 characters." }
    - 401 Unauthorised: Invalid token
        - { "error": "Invalid or missing token." }
    - 403 Forbidden: Invalid quizid or questionid
        - { "error": "error message" }

API Documentation: Player Request Hint on Quiz Question

- **Endpoint:**
  GET /v1/player/{playerid}/question/{questionposition}/hint
- **Description:**
  Allows a player to request a hint for the current question after it has been visible for 10 seconds.
- **Parameters:**
    - Path Parameters:
        - playerid: The unique Id of the player (Integer, Required)
        - questionposition: The position of the question (Integer, Required)
- **Request Body:**
    - None
- **Responses:**
    - 200 OK: Hint successfully provided
        - { "hint": "Use the triangle inequality" }
    - 400 Bad Request: Returned if any of the following conditions are met:
        - Player ID does not exist.
        - Question position is not valid for the session this player is in.
        - The session is not currently on this question.
        - The question has not been visible for at least 10 seconds.
        - The player has already requested a hint for this question.
            - { "error": "Hint already shown" }

State Diagram for Hint:



User Story 2 (addressing Cam):

As a quiz participant, I want to submit feedback on the quiz at the end so that I can share my experience with the admin.

User Acceptance Criteria:

1. **Feedback Prompt:**
   After completing the quiz, participants automatically see a prompt asking them to share feedback on their experience.
2. **Feedback Entry:**
   Participants can provide feedback in a text box with a 500-character limit, or choose to skip this step.
3. **Submission Process:**
   - A "Submit Feedback" button activates when feedback is entered.
   - Clicking "Submit Feedback" finalises submission and displays a thank-you message.
4. **Feedback Handling:**
   - Feedback is saved and accessible to the quiz admin.
   - Feedback remains anonymous, and once submitted, cannot be modified.

Use Case:

Use Case Background

- Use Case: Submit Feedback on Quiz
- Goal in Context: The participant completes the quiz and provides feedback about their experience to help improve future quizzes.
- Scope: Quiz system
- Preconditions: The quiz has been completed by the participant.
- Success End Condition: The participant's feedback is submitted and stored in the system, accessible by the quiz admin.
- Failed End Condition: The participant skips the feedback section or the feedback is not submitted.
- Primary Actor: Quiz participant
- Trigger: The participant submits the quiz.

Use-Case List

1. System: Displays a feedback prompt to the participant upon quiz submission.
2. Participant: Views the feedback prompt and the text field for entering feedback.
3. System: Shows a placeholder in the feedback text field: "Enter your feedback here…" and a character limit of 500 characters.
4. Participant: Enters feedback or chooses to skip the feedback section.
    - If choosing to skip:
        - System: Finalises the quiz submission without storing feedback.
    - If entering feedback:
        - System: Activates the "Submit Feedback" button.
5. Participant: Clicks the "Submit Feedback" button.
6. System: Checks that feedback is less than 500 characters.
    - If valid character length:
        - System: Stores the feedback and displays a success message: "Thank you for your feedback!"
    - If invalid character length:
        - System: Displays an error message and returns to step 4.
7. System: Marks the feedback as anonymous and makes it accessible for the quiz admin to review later.

Validation:

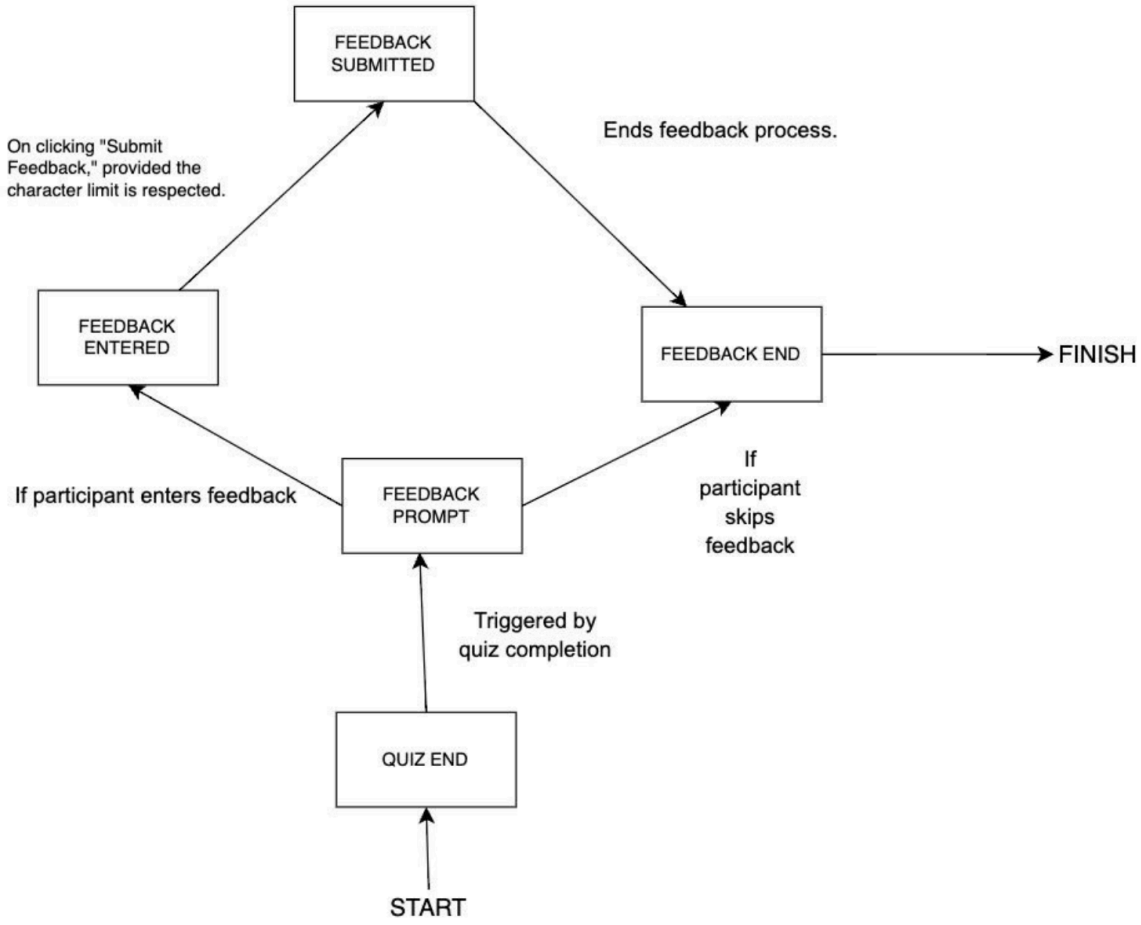
Cam's Comment: *"This solution is great. It makes it easy for participants to reach out to the quiz admin. I like how the feedback feature facilitates communication and sharing ideas."*

API Documentation: Player Feedback Submission

- **Endpoint:**
  POST /v1/player/{playerid}/feedback

- **Description:**
  Allows a player to submit feedback on their quiz experience after completion.
- **Parameters:**
  - Path Parameters:
    - playerid: The unique ID of the player (Integer, Required)
- **Request Body:**
  - feedback: The feedback text (Optional, max 500 characters)
  - Example Request Body:
    - { "feedback": "Question 5 of the quiz had multiple spelling errors" }
    - If skipping feedback, use an empty body: {}
- **Responses:**
  - 200 OK: Feedback successfully submitted
    - { "message": "Thank you for your feedback!" }
  - 400 Bad Request: Returned if any of the following conditions are met:
    - The player ID does not exist.
    - The feedback exceeds the 500-character limit.
      - { "error": "Feedback must be 500 characters or less." }

State Diagram for Feedback:



User Story 3 (addressing Bobby):

As a quiz admin, I want to enable CAPTCHA verification for my quizzes so that only human participants (not bots) can access the quiz.

User Acceptance Criteria:

- **CAPTCHA Setup:**
  - In the quiz creation/editing page, the admin can enable CAPTCHA verification by toggling an enable CAPTCHA verification option.
  - The system saves the CAPTCHA verification setting for the quiz.
- **CAPTCHA Challenge for Participants:**
  - If CAPTCHA verification is enabled, participants are required to complete a CAPTCHA challenge before accessing the quiz.
  - The participant submits their CAPTCHA response, which is verified by the system.
- **Access Grant or Denial:**
  - If the CAPTCHA response is valid, the participant gains access to the quiz.
  - If the CAPTCHA response is invalid or incomplete, the participant will see an error message.

Use Case:

Use Case Background

- Use Case: Restrict Quiz Access Using CAPTCHA Verification.
- Goal in Context: The quiz admin restricts quiz access to ensure participants are human, preventing automated bots from accessing the quiz.
- Scope: Quiz system
- Preconditions: The quiz admin has enabled CAPTCHA verification for the quiz.
- Success End Condition: Participants pass the CAPTCHA verification and gain access to the quiz.
- Failed End Condition: Participants who fail the CAPTCHA verification or fail to complete it correctly cannot access the quiz.
- Primary Actor: Quiz admin
- Trigger: Participant attempts to access a CAPTCHA-enabled quiz.

Use-Case List

1. System: Displays a "CAPTCHA Verification" option on the quiz creation/editing page for the quiz admin.
2. Quiz Admin: Enables CAPTCHA verification by toggling the relevant option.
3. System: Saves the CAPTCHA verification setting for the quiz.
4. Participant: Attempts to access the quiz.

5. System: Detects that CAPTCHA verification is required and displays a CAPTCHA challenge to the participant.
6. Participant: Completes the CAPTCHA challenge.
7. System: Verifies the CAPTCHA response.
    - If the CAPTCHA is successfully completed:
        - Grants access to the quiz.
    - If the CAPTCHA is not completed or invalid:
        - Displays an error message: “CAPTCHA verification failed. Please try again to access the quiz.”
8. System: Finalises quiz access by either granting or denying it based on the CAPTCHA verification result.

Validation:

Bobby’s Comment: *"The CAPTCHA solution addresses my concern. It keeps bots away without making it too hard for real people to join the quiz. It’s just what’s needed."*

API Documentation: Admin Add Quiz CAPTCHA Verification

- **Endpoint:**
  POST /v1/admin/quiz/{quizid}/captcha
- **Description:**
  Allows the admin to add CAPTCHA verification for a quiz.
- **Parameters:**
    - Path Parameters:
        - quizid: The unique ID of the quiz (Integer, Required)
    - Headers:
        - token: The admin's authorization token (String, Required)
- **Request Body:**
    - enableCaptcha: Boolean flag to enable/disable CAPTCHA (Boolean, Required)
    - Example Request Body:
        - To enable: { "enableCaptcha": true }
        - To disable: { "enableCaptcha": false }
- **Responses:**
    - 200 OK: CAPTCHA successfully enabled/disabled
        - { "message": "CAPTCHA verification successfully enabled for the quiz." }
    - 400 Bad Request: Invalid quiz ID
        - { "error": "message" }
    - 401 Unauthorised: Invalid or missing token
        - { "error": "Invalid or missing token." }

API Documentation: Player CAPTCHA Verification

- **Endpoint:**
  POST /v1/player/{playerid}/quiz/{quizid}/session/{sessionid}/captcha
- **Description:**
  Verifies the CAPTCHA response for a player accessing a quiz session.
- **Parameters:**
  - Path Parameters:
    - playerid: The unique ID of the player (Integer, Required)
    - quizid: The unique ID of the quiz (Integer, Required)
    - sessionid: The unique ID of the session (Integer, Required)
- **Request Body:**
  - captchaResponse: The CAPTCHA response token (String, Required)
  - Example Request Body:
    - { "captchaResponse": "03AF6jK6..." }
- **Responses:**
  - 200 OK: CAPTCHA verified, access granted
    - { "message": "CAPTCHA verified successfully. You now have access to the quiz session." }
  - 400 Bad Request: Invalid player ID, quiz ID, or session ID
    - { "error": "message" }
  - 403 Forbidden: CAPTCHA not completed or skipped
    - { "error": "CAPTCHA verification required. Please complete the CAPTCHA challenge to proceed." }

State Diagram for CAPTCHA: