Telegram Bot Development Document

Project Objective

Develop a Telegram bot that provides a comprehension quiz with multiple topics. Users can choose a comprehension topic, read a brief passage, and answer 10 multiple-choice questions. Each response receives immediate feedback, and after 10 questions, the bot displays the user's final score and prompts them to choose another comprehension topic.

Bot Flow and Features

1. Main Menu
    - Comprehension Topics: Display 4-5 topics for users to choose from, e.g., Nature, Science, History, Literature, Technology.
    - Topic Selection: After the user selects a topic, the bot retrieves a brief comprehension passage (100-150 words) and presents it to the user.
2. Comprehension and Quiz Flow
    - Comprehension Display: Once a topic is selected, show a brief 100-150 word passage on the chosen topic.
    - Question Format: After showing the comprehension, the bot asks 10 questions, each with 4 options (one correct answer).
        - Options: Display four multiple-choice options (A, B, C, D).
        - Answer Evaluation:
            - Correct Answer: If the user selects the correct answer, display a congratulatory message (e.g., "Great job! That's correct!").
            - Incorrect Answer: If the user selects the wrong answer, display an appreciation message (e.g., "Good try! The correct answer is [Correct Answer].").

    - Navigation Buttons:
        - Next Question: Shows the next question if fewer than 10 questions have been answered.
        - Select Another Topic: Allows the user to return to the main menu to pick a new comprehension topic.
3. Quiz Completion and Scoring
    - After 10 Questions:
        - Display a Summary Screen with the user’s score out of 10.
        - Prompt the user to Select a New Comprehension Topic to start a new quiz round.

Database Requirements

1. Database Structure
   The bot will use a database to track user responses, current question index, and score. The bot should retrieve and update user data from the database for each interaction.
2. Data Points to Track:
    - User ID: Unique identifier for each user.
    - Topic Selected: The chosen comprehension topic.
    - Current Question Index: Track the current question (0-9) for each user.
    - User’s Score: Track the number of correct answers.
3. Database Operations:
    - Insert: Add a new entry when a user selects a comprehension topic.
    - Update: For each question response, update the question index and score based on the user’s answer.
    - Reset: After each quiz, reset the question index and score, allowing the user to start a new quiz round.

Technical Requirements

1. Telegram Bot Framework:
    - Use Python , node ,javascript, etc or another preferred language and framework compatible with Telegram’s API.
2. Database:
    - Suggested Database: Use a relational database (e.g., PostgreSQL or MySQL) to store user data securely.
3. Logic Implementation:

    - Dynamic Retrieval: Questions, options, and correct answers should be stored in and fetched from the database based on the selected topic and question index.
    - State Management: All user data should be retrieved from and stored in the database rather than using in-code variables to ensure consistency and persistence.
4. Additional Considerations:
    - Error Handling: Ensure the bot gracefully handles any unexpected inputs or database connectivity issues.
    - User Experience: Ensure response times are optimized, and messages are clear and user-friendly.

Deliverables

- Bot Source Code: Complete bot implementation, including code to handle Telegram interactions and database management.
- Database Schema: A schema for the database showing all necessary tables and relationships.
- README Documentation: Outline instructions for setting up the bot locally, setting up the database, and a high-level overview of the bot's structure and functionality.
- Provide a short video of output.

Timeline

Completion Date: 5 november