BIRZEIT UNIVERSITY

Advanced Computer Systems Engineering Laboratory – ENCS5150

ToDo2: SQLite Databases in Android

Question: Mobile Math Quiz Game

In this quiz, you are required to build a simple mathematical quiz game. Users will register for the quiz application by submitting their usernames, email addresses, and birthdates. The game will consist of 5 questions, and users need to answer each question before a timer finishes. If the answer is correct, the score is incremented by 1. Users will lose one point if the answer is incorrect or the timer expires. The math questions for the quiz will be stored in a CSV file named Questions.csv, which you will read in your Android app to load the questions and answers.

First Screen (Main Activity)

1. **User Registration**:
    - ❖ The screen will have text fields for new users to enter their:
        - **Username**: Must be unique and cannot be empty.
        - **Email**: Must be in a valid format (e.g., tzidan@birzeit.com).
        - **Birthdate**: A valid date must be valid.
    - ❖ A **Start** button will initiate the game once registration is complete.
    - ❖ Note that player ID is an auto-incremented attribute, not entered by the user.
2. **Check for Existing Nicknames**:
    - ❖ Before adding a new nickname to the scores table, check if it already exists in the database. You may use SQL query to count occurrences of a particular nickname.

Second Screen (Quiz)

1. **Loading Questions**:
    - ❖ The math questions for the quiz will be stored in a CSV file, Questions.csv. You will read this file in your Android app to load the questions and answers into the game. The CSV file format includes the following structure:

- Each row should contain a question followed by its correct answer (e.g., What is 2 + 2? 4). Display the question above, and the answer should be one of the options to choose from below the question.

2. **Random Question Selection**:
    - You must randomly select five unique math questions from the Questions.csv file. Ensure that the selected questions do not repeat during the game session. You can achieve this by maintaining a temporary list of already-selected question indices.
3. **Timer Implementation**:
    - Each question will have a 10-second countdown timer. You may use the CountDownTimer class to implement this (do your research here). Ensure the game automatically progresses to the next question when the timer runs out.
4. **Scoring System**:
    - The score will be incremented by one for every correct answer and decremented for every incorrect answer or timeout.
5. **Question Transition**:
    - If the timer finishes for a question, the next question will be displayed automatically.
6. **End of Quiz**:
    - Once all five questions are completed, insert the user's score and nickname into the database and go to the final screen.

Third Screen (End)

1. **Queries to perform in the last screen**:
    1. **Retrieve the Top 5 Scores Query**: This query fetches the top 5 scores from the scores table, sorted in descending order, showing their nickname and score.
    2. **Count Total Number of Players Query:** This query counts the number of unique players registered in the player's table.
    3. **Retrieve All Scores of a Specific Player Query:** To display a specific player's scores, you can use this query, replacing the nickname with the actual player's nickname. Leave Plaintext here for the user to enter the name.
    4. **Calculate Average Score Query:** This query calculates the average score of all players.
    5. **Find the Highest Score Query:** This query retrieves the highest score information recorded in the scores table.
2. **Navigation**:
    - Include a button that allows users to return to the first screen to start a new quiz.

3. **User Feedback**:
   - Provide toast message feedback for the user, such as "Correct!" or "Wrong!" after each answer, and a summary of their performance at the end, including the number of correct answers.
4. **Error Handling**:
   - Implement error handling for potential issues, such as failures in reading from the CSV file or connecting to the database. This will enhance the robustness of your application.

Suggested schema (You may or may not build the same one it is up to you):

Table 1: Scores

Schema:

- id: Unique identifier for each score entry must be auto-incremented.
- nickname: The player's username must be unique.
- score: The score achieved by the player.
- timestamp: The date and time when the score was recorded.

Table 2: Questions

Schema:

- id: Unique identifier for each question.
- question: The text of the math question.
- correct_answer: The correct answer to the question.
- option1: An additional answer option (can't be null).
- option2: Another additional answer option (can't be null).
- option3: Another additional answer option (can't be null).

Notes:

- You can create any design as long as it is clear, arranged, and understandable.
- The name of the application must follow the convention **ID_FirstName_FamilyName.**
- I will use a Pixel 3a XL device with API Level 26 (Graphic=Software). So, please match these requirements to avoid unexpected issues once I test your solution.
- ToDo is individual, and Cheating or ChatGPT is not allowed (it will lead to a 0 mark!). **We'll be very strict about that**.

What to submit?

1. Project.zip file (Size in KB)

   From Android Studio: File → Export to zip file

2. APK file

From Android Studio: Build → Build Bundle(s) / APK(s) → Build APK(s)

3. Record your screen in a video that does not exceed 2 minutes, show all possible test cases, schemas, and queries, upload your video to Google Drive, and attach your link (make sure it is accessible using the link), or upload your video through Ritaj if possible.

Where to submit?

- Send all the APK file, ZIP file, and drive links as a reply to my message

Deadline?

- Sunday, October 27, 2024 at 11:59 PM

Dataset?

https://drive.google.com/file/d/1scJ2xuhrzpXDfdb9m9Nh6oDgQkLHeUWQ/view?usp=sharing

Marking Criteria?

Design	5
Main Activity	10
Quiz Activity	10
End activity	10
Databases	10
Video content	5
Total	**50**