Homework – 03
CSC 4360 - Mobile App Development
Due- 10/14/2024

Objective: The objective of this assignment is to create a Flutter card-matching game that combines animation and state management. Students will build a game where players need to match pairs of cards.

In this assignment, you will create a Flutter card-matching game. Players will be presented with a grid of face-down cards, and they must flip the cards to find matching pairs.

Requirements:

1. **Create a Flutter Project:**
    - Start a new Flutter project using your preferred development environment
2. **Design the User Interface:**
    - Design a user interface that includes a grid of face-down cards (e.g., 4x4 or 6x6 grid).
    - Each card should initially display a back design (e.g., a common pattern or image).
    - You can use **GridView** or other Flutter widgets to create the card grid.
3. **Implement State Management:**
    - Use a state management approach (e.g., Provider) to manage the game state.
    - Define a data model for the cards, including properties for their front and back designs, and their current state (face-up or face-down).
4. **Card Flipping Animation:**
    - Implement an animation that flips the cards from face-down to face-up and vice versa when tapped.
    - Utilize animation widgets like **AnimatedBuilder** or **AnimatedContainer** to create the flip animation effect.
5. **Game Logic:**
    - Implement the game logic to track which cards are currently face-up and face-down.
    - When a player taps two face-down cards, check if they match. If they do, keep them face-up; otherwise, flip them face-down again.

Extra Task for graduate students : Timer and Scoring System

Introduce a timer and a scoring system to make the game more challenging and interactive.

Requirements:

- Add a timer that starts when the game begins.
- Display the timer on the screen, counting the time taken to match all pairs.
- Implement a basic scoring system where players earn points for each matched pair.
- Deduct points if the player mismatches cards, or alternatively, provide bonus points for matching pairs quickly.

Guidelines:

- The timer should stop when all pairs are matched and the victory message is displayed.
- Optionally, allow players to restart the game and try to beat their previous time/score.
- Design the UI to display both the timer and the current score at the top of the screen.

6. **Win Condition:**
    - Implement a win condition to check if all pairs have been matched. When all pairs are matched, display a victory message.
7. **Testing:**
    - Run the app on an emulator or physical device.
    - Verify that players can flip the cards to find matching pairs.
    - Test the win condition and display a victory message when the game is won.

Submission Instructions:

- To ensure accurate grading, all students are required to upload the **APK file** and furnish a GitHub repository link directing to the current work submission for assessment. Merely submitting the file will not suffice for evaluation. You should not just upload the files to Git Hub but use Git Hub for continuous development and we will check the commit history.