Work Sample

The goal of this work sample is to construct a sample memory game called ”Colour Memory”. The game board consists of a 4x4 grid with 8 pairs of color cards.

The game starts initially with all cards facing down. The player is to then flip two cards each round, trying to find a match. If the flipped pair is a match, the player receives two (2) points, and the cards may be removed from the game board. Otherwise, the cards are turned face- down again and the player loses one (1) point. This continues until all pairs have been found.

After the game is finished, the user should be prompted to input his name. The user's name and the score would then be stored in a database, and the user should be notified of his score and the current rankings.

Specifications and Features

- It is highly recommended to write the app using the standard language of your selected target platform.
    - For Android, it should be written in Java and must support devices running 4.0 and above.
    - For iOS, it should be written in Swift and/or Objective-C and must support devices running iOS 8.0 and above.
    - If the app is written as a hybrid web app, then you must provide the source for both the Android and iOS platform.
    - If the app is written as a pure web app, then it MUST work on ALL platforms (including the desktop, Windows Phone 8, etc.).
- The app must at least be runnable on the emulator. Tablet or iPad devices do not have to be supported, but will be considered as an extra feature.
- The game board should be displayed in portrait orientation only.
- After each round, a brief one (1) second pause should be implemented before scoring to allow the player to see what the second selected card is.
- When a player submits his name, the entered value must pass basic validation (no empty strings) in order to continue.
- Clicking the High Scores button will take the user to the table of high scores.
- The high scores table may be displayed in either landscape or portrait orientation.
- The high scores table may contain negative numbers.
- A score is considered a high score if the score is higher than a current high score OR there is space in the table.
- High scores must be stored and persist as long as the user does not uninstall the app or clear the app's stored data.
- High scores database may be implemented using the device's internal database. Network-based high scores will be considered as an extra feature with the exception of a pure web app.

Design

- The full game must fit exactly inside a device's available window area.
- The logo must be displayed in the top left of the window.
- The High Scores button must be visible to the top right of the window.
- The current score should be displayed in the top, centered between the logo and the High Score button.
- The game board must be displayed below the logo and high score button
- The user input may be implemented as a separate screen or as a popup
- The high scores table must be displayed as a proper table with three columns: Rank, Name, and Score
- All necessary graphics for the cards, logo, and app icon have been supplied.
- Usage of any other graphical elements will not be provided, but may be used as long as it does not detract from the user experience.

Delivery

- The full source and documentation of the app should be delivered in a compressed archive file containing only the necessary files and resources needed to build and launch the app. The archive must also contain the packaged installer (for Android, an .apk file; for iOS, an .ipa file).
- If the app is built a as a web app, then the packaged installer is not necessary.
- If a server is required for the high scores table, the server implementation and installation guide must also be included.