COSC 4355/6355 – Introduction to Ubiquitous Computing

Exam - 1

September 21, 2023

Objective

Create a tic tac toe game with a twist (details to follow)

Motivation

Alerts, tab bar controller, auto layout, stacked views, data transfer

Must Follow

You must follow the rules below.

1. Start your XCode project "**Exam1_LastName_FirstName**" (replace **LastName** with your last name and **FirstName** with your first name).

   a. **DON'T DO** the following: Start with any other project name and change the zip file name later. This will not be accepted at all.

2. Your submission must be compatible with **XCode version 14.3.1** or **higher**.

Tips

- **Read the question carefully, then start coding!**
- **Build, Build, and Build**
  - If you add anything on storyboard -> **Build**
  - If you make a reference from storyboard -> **Build**
  - Do not wait until finishing all parts to build.
  - It is easier to debug after each single feature added.

Detailed Description

You are going to build a Tic Tac Toe app with a twist. In this unique version of Tic Tac Toe if you get three Xs or Os in a line you lose rather than win. Call it Tic Tac Toe Reverse. Furthermore, the game board is set randomly by the computer. Hence, it is not like one player plays against the other. In fact, it is a betting game, where you win if the board does not feature any line of Xs or Os; you lose if it does. The app falls under the multiple view iOS application category. Of course, you will use Swift as the programming language.

[3 pts] STEP 1: The app will have two screens: Game Screen and Bank Screen. Design your interface in **Portrait** mode to look like the screenshots [Figures 1 - 3, 5]. Pay attention to images, icons, colors, fonts, and font sizes. **For canvas use iPhone 14 Pro.**

Fonts: Gill Sans / Gill Sans Bold; buttons and labels have size=40.0; title has size=50.0
Initial state of the game: **Credit = 100**, **Won = 0**, **Spins = 0**, **Bet = 1**

[3 pts] STEP 2: Use Auto Layout or SwiftUI to make the app behave properly in both Portrait and **Landscape** mode [Figures 7 - 9]

[2 pt] STEP 3: When the user taps the "Play" button randomly populate the Game Screen with nine X and Os [Figure 3].

[3 pt] STEP 4: Implement the game logic as follows:

- **Loosing Run:** If three pictures are identical in **any** row [Figure 3], column [Figure 11], or diagonal [Figure 10], then highlight such triads in red and perform the following calculations:
  - **Credit = Credit – Bet**
  - reset **Bet = 1;** set **Spins = Spins + 1**

- **Winning Run:** If there are no lines with triads, then perform the following calculations:
  - **Credit = Credit + Bet**; set **Won = Won + 10** $\times$ **Bet**
  - reset **Bet = 1;** set **Spins = Spins + 1**

[1 pt] STEP 5: Check if there is enough credit available, that is, **Bet** $\leq$ **Credit**. If there is not enough 'Credit', raise alert [Figure 4] and reset **Bet = 1.**

[1 pt] STEP 6: Every time the user taps the button "↑ Bet", double the bet, that is, **Bet = Bet** $\times$ **2**. When **Bet** $\geq$ **1000**, then reset **Bet = 1**.

[1 pt] STEP 7: Always keep your 'Bank' and 'Game' screens synchronized. The counters 'Spins', 'Won', 'Credit' should be up to date all the time. [Figures 6,5]

[1 pt.] STEP 8: 'Add credit' button increases `Credit' by the amount entered in the text field [Figures 7,8]. If the user enters the wrong input (e.g., a letter), an alert message is shown [Figure 6] and the input text field is reset to '0'.

[Bonus problem #1: 1pt] Rig the game! Bias it so that at the latest the 4th run gives a win.

[Bonus problem #2: 1pt] Make your app running properly on any type of device (incl. iPad).

Good luck and happy coding! ☺

Submission

Zip your XCode project and submit to TEAMS. The name of your zip file will be automatically "**Exercise4_LastName_FirstName.zip**" (**LastName** is your last name and **FirstName** is your first name). One submission per person.

Hint

Use helper function for implementing the game logic in STEP 4. The helper function should take three 'UIImageView' as arguments and return Boolean as an output. In the body of the function check if all three images in the input views are identical. If they are identical, then change images to red and return true. If they are different, then return false. This helper function should help you minimize your code for performing 8 comparisons (3 columns, 3 rows, 2 diagonals).

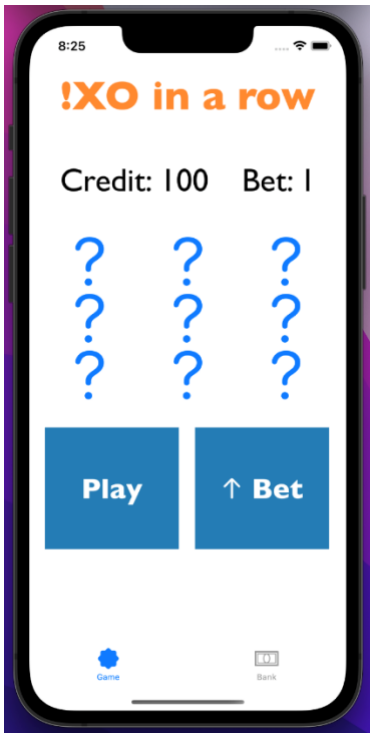

Figure 1. Game screen

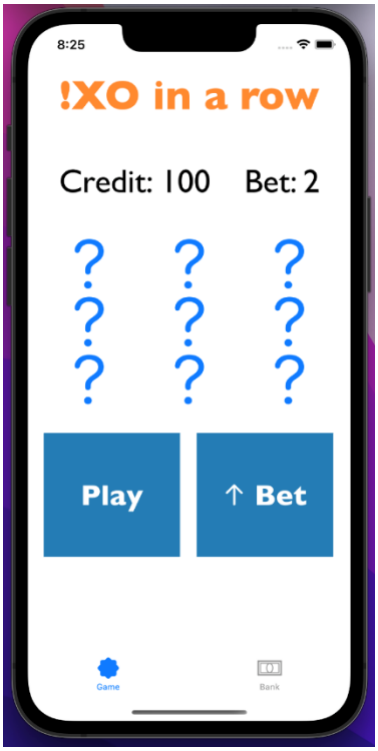

Figure 2. Game screen

Figure 3 Game screen

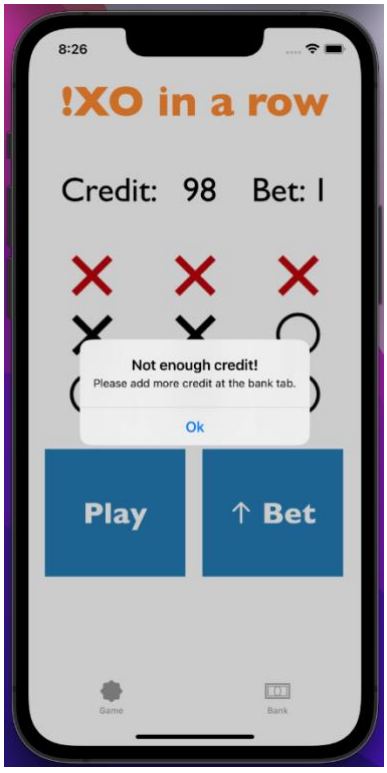

Figure 4. Game screen

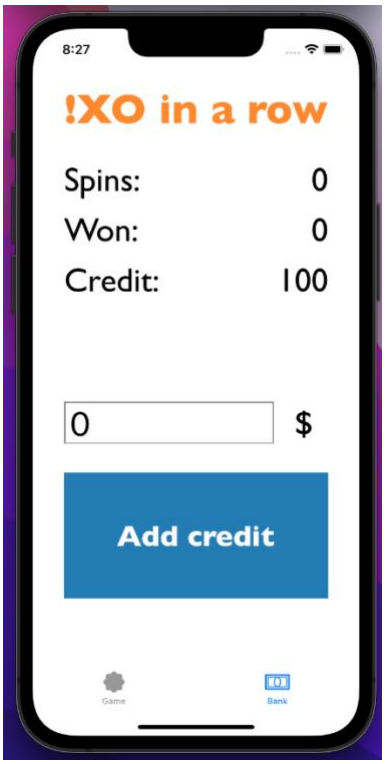

Figure 5. Bank screen

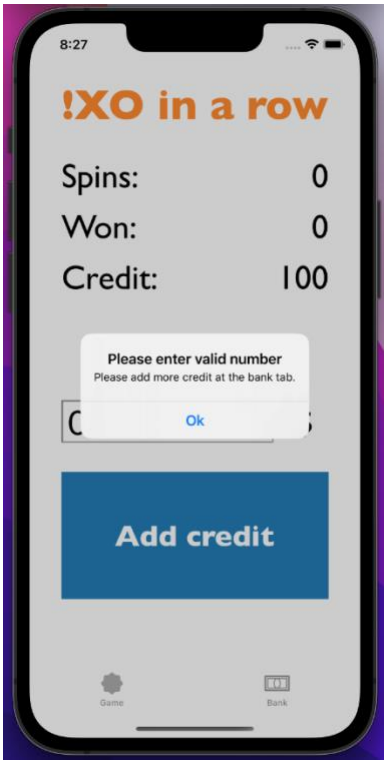

Figure 6. Bank screen

UNIVERSITY of HOUSTON

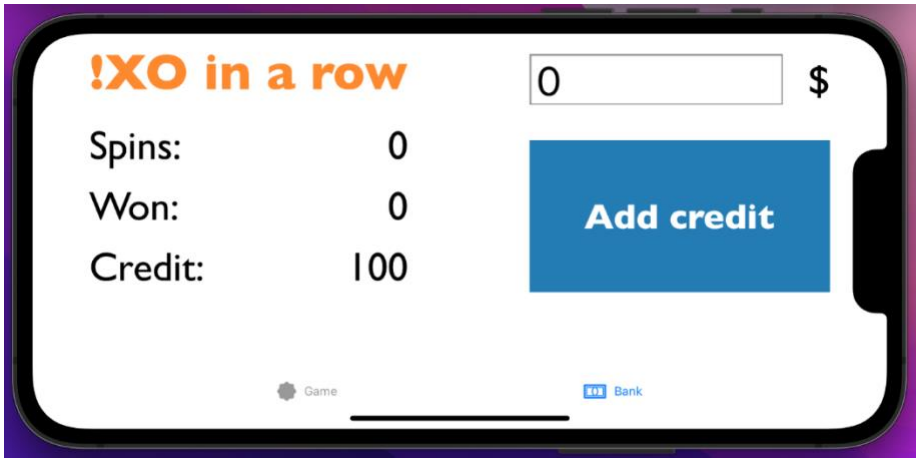

Figure 7. Bank screen

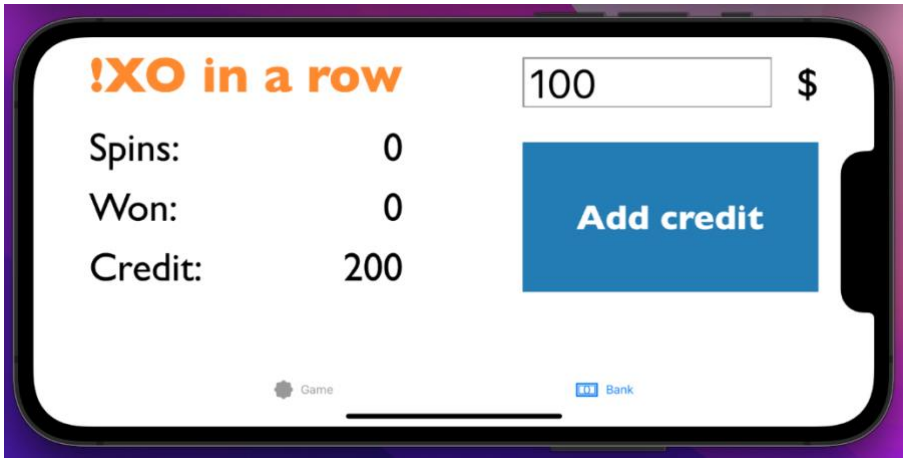

Figure 8. Bank screen

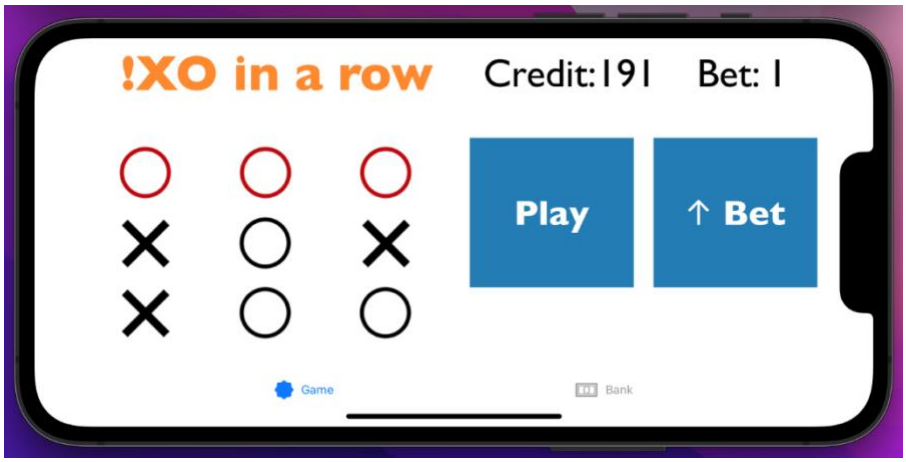

Figure 9. Game screen

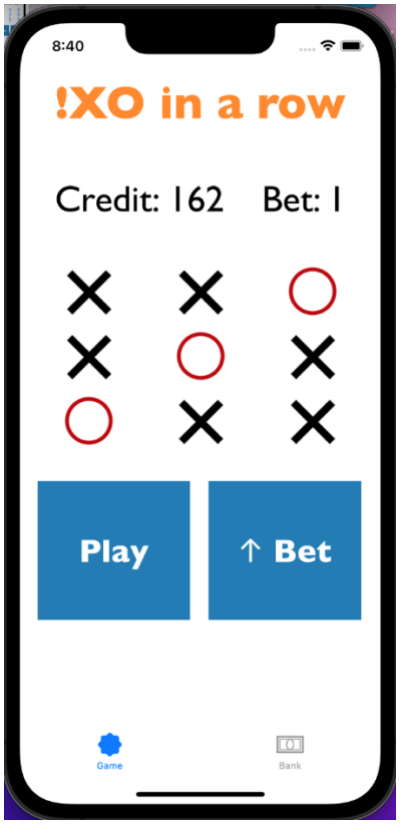

Figure 10. Game screen

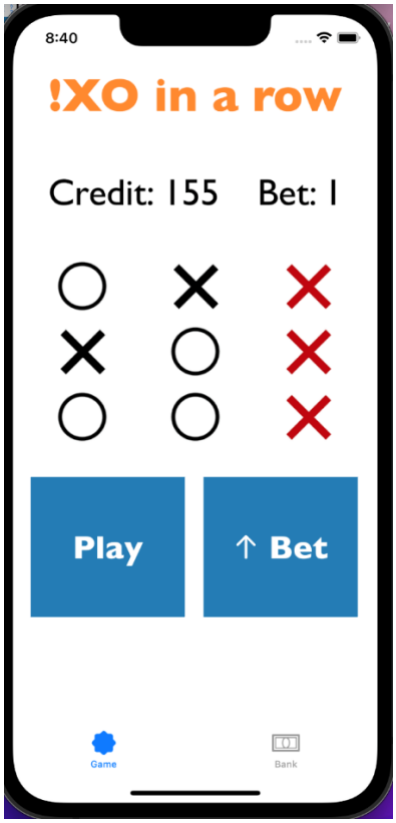

Figure 11. Game screen