Blackjack App

CSE 438

Spring 2019

Introduction

In this assignment, you will make use of gestures and Firebase to create a Blackjack app.

Details

- Due Date: March 21, 2019
- Grading: This entire lab is 110 points. The distribution of points is listed below in the requirements section.
- Partners: For this assignment you are allowed to work with one partner. You and your partner should submit one version of the final assignment. Please put both of your names on the repository.
- Submission: Please put your project into your forked repository. The repository can be found here. After you fork your repository, please provide admin access to the class account: cse438wustl@gmail.com. You can do this in the User Access section of the repository settings.

Blackjack Instructions

For this assignment, we will be using these rules: There will be two players, a player (referred to as Player) and a dealer. At the beginning of the game two cards will be dealt to both players. Both cards dealt to the Player will be dealt face up. One of the cards dealt to the dealer will be left face down. The Player will then choose what move they want to make. The Player can choose to hit to receive a new card, or stand so that it is the dealer's turn.

If the Player's total goes above 21, they will bust and the dealer will win. If the Player is able to stand, the turn goes to the dealer. The dealer's moves are controlled by certain rules. If the dealer was dealt 21, they will automatically win the game. If the dealer's total is above 17, they will stay, and the resulting winner will be whoever has a larger total. If the dealer has below 17, they will keep hitting until they have a total above 17, or they bust. All face cards have a value of 10, and an Ace can change between 11 and 1 depending on the current total.

Description

When the app launches, there should be a login screen which allows the Player to login with a username and password. You should use the Firebase Authentication module to assist in this process. Once the Player successfully logs in, a new activity should open up which has the actual game. The layout should have the cards of the dealer and the player visible according to the rules of the game. Additionally, the app should display the number of wins/losses the player has up to this point. The number of wins and losses that a player has should be stored in a Firestore database.

For the actual gameplay, you should make use of gestures to hit and stand. If the player swipes to the right that should indicate they want to hit. If they double tap the screen, that should indicate they want to stand. The app should handle declaring the winner/busting the player in the appropriate scenarios, and update the Firebase database appropriately.

You should also use animations to "deal" the cards. If a player requests a card, you should make the card move from one corner of the app to the player's set of cards, making sure that all of the cards are visible. The same animation should take place when the dealer requests a card. Make sure your animations are slow enough to be visible, but not so slow that the app feels sluggish.

Additionally, the app should show a leaderboard of all the players and should rank them according to the win to loss ratio. Since all of this is stored in Firebase, the leaderboard should be consistent among multiple installations on the app (Hint: You have partners for this assignment - try testing on multiple phones).

We have covered nearly every component of this project in class. Below is a list of where you can find examples of concepts that you will need to complete this assignment:

- Firebase Auth: Firebase Example Projects, Studio 4
- Firebase Firestore: Firebase Example Projects, Studio 4
- Drawables: animationgesturedemo
- Gestures: animationgesturedemo
- Animations: animationgesturedemo

Dealing the Cards

You will be given the cards as images in the drawables resource folder. There will also be one image for the back of a card which you can use for the dealers face down card. You will be given a class called CardRandomizer. The CardRandomizer object contains one method that will return a list of all of the ID's for the cards in the resource folder. Here is an example of generating a random card:

```
val randomizer: CardRandomizer = CardRandomizer()
var cardList: ArrayList<Int> = randomizer.getIDs(this) as ArrayList<Int>
val rand: Random = Random()

val r: Int = rand.nextInt(cardList.size)
val id: Int = cardList.get(r)

val name: String = resources.getResourceEntryName(id)
```

To display the cards, you will need to pass the card id to an ImageView. There are a couple of different ways you can set this up:

- One approach would add some ImageViews to the layout ahead of time. Once you want to deal a card, you would grab the next image view, set the approprate image resource using the id, then move the card to the appropriate position.
- You can also dynamically create ImageViews in your code. In order to do this, you need to make sure to add the ImageView to the layout after it is created.

ImageViews were used during the animation and gesture demo presented in class. You will likely also want to use the documentation for ImageViews to help you out here.

Creative Portion

For every homework assignment, you will be asked to think of an additional feature to be added to the application that will improve the user experience and provide you an opportunity to learn about concepts that you are personally interested in. Put yourself in the shoes of your users: what features would they like to see in an app like this? Try to make it something new and substantially different from what the app already does - do not just rehash existing requirements.

When you submit your assignment, please include a ReadMe.txt file that explains your creative portion. You should explain what the feature is, why you chose to implement that particular feature, and how you went about implementing it.

To recieve full credit, your feature needs to be substantial as compared to the rest of the assignment. Examine the rubric below to get a feel for how much weight we are putting on the creative portion of the assignment.

Requirements

- (10 points) Player can login and login data is stored in Firebase
- (10 points) Player's win/loss counts are displayed on startup
- (10 points) The Player receives two cards face up, and the dealer receives one card face up and one card face down
- (5 points) Swiping right allows the Player to hit
- (5 points) Double tapping allows the Player to stand
- (10 points) Cards being dealt are animated, and all cards are visible.
- (5 points) If the Player goes over 21, they automatically bust
- (10 points) The dealer behaves appropriately based on the rules described above
- (10 points) Once the game is complete, the winner should be declared and the Firebase database should be updated appropriately
- (5 points) A new round is automatically started after each round
- (5 points) Player can logout
- (10 points) A leaderboard is shown in a separate tab or activity and is consistent among installations of the app
- (15 points) Creative portion!

Helpful Advice

- Take a look at the studio code for how to handle gestures and work with a Firebase database.
- Consider making use of a RelativeLayout to handle displaying the card images - remember that the number of cards that will be displayed is not constant so you will need to handle varying number of cards.