**BUG REPORT**

A memory leak may exist if many rounds in a row (4+) are played causing a crash, but this has not been tested extensively.

A bug where the player can attain multiple turns can be exploited as such:

1. When it is human turn, click “next move”
2. Select a hand card to play
3. Click next move again to initiate the playing of the stock card for the human
4. Normally the player should click the layout to move the stock card appropriately, but if they instead click their hand again, the stock card will be added to the layout and it will remain human turn.
5. The bug stems from the structure of the gameActivity class, the selectedLayout function is supposed to handle all inputs for adding cards to the layout, which it does so properly. If the player clicks the hand, the selectedCard function is called, and since the selectedLayout function has not properly set the game’s logic to advance the turn, the player receives another turn by forcing the function call.
6. This bug can be fixed by modifying the selectedCard function to simply call the selectedLayout function during the gameState of the current exploit.

A bug exists due to a bad checking of array indices when the player tries to capture a 3 stack with a card. The program is attempting to remove a layout card from a bad index. This can be fixed with a predicate (matches[i] > -1) before the removal is called.

**FEATURE REPORT**

**Missing Features**:  
 The class structure is not based on inheritance. There is only a card and a client class for the logical portion of the project. The client class handles all the logical operations involving either player or the game.

**Extra Features:**

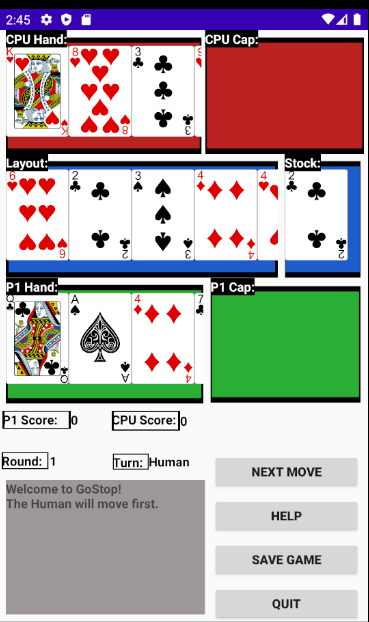
Unless you count card stacks having their own unique images, there are no extra features at this time.

**DESCRIPTION OF THE DATA STRUCTURES/CLASSES**

The card class is used to represent the card objects, the Client class handles all logical operations used in the game (playing the hand card, stock card, scoring, cleanup etc.) including for both players.

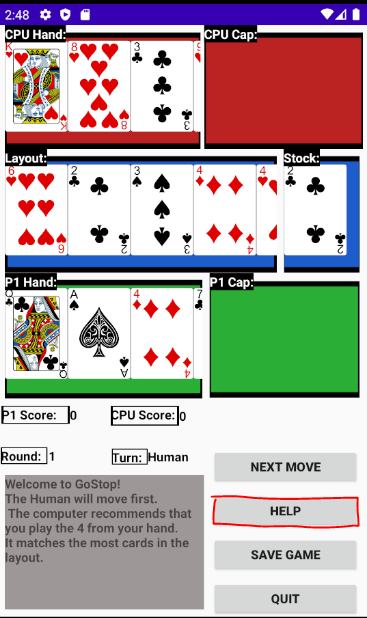
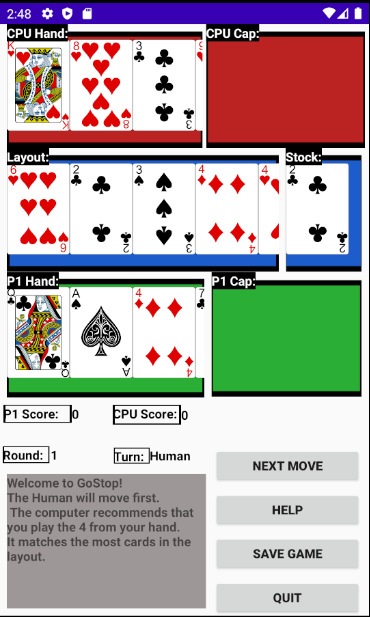
The gameActivity class handles all of the GUI and user input for the game, and the mainActivity class handles the launching of the gameActivity class/view.

**SCREENSHOTS**



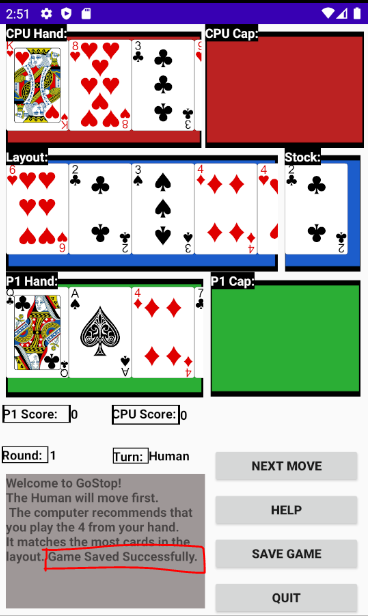
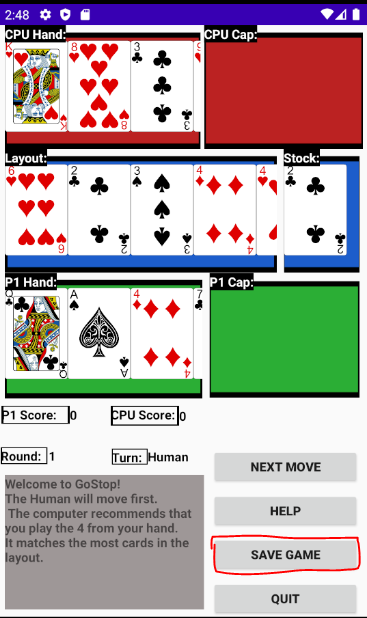
**➜**

**Starting a new game**



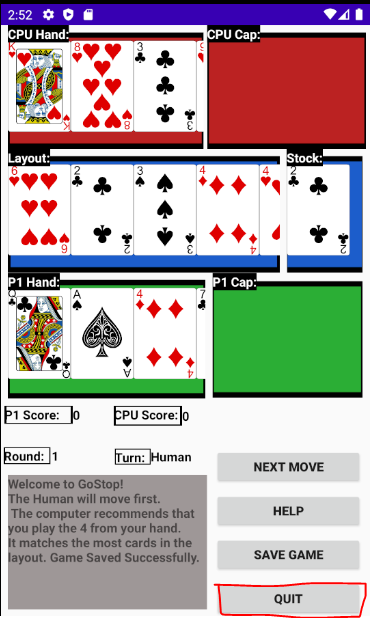
**➜**

**Asking for help**



**➜**

**Saving the game  
(player turn only)**



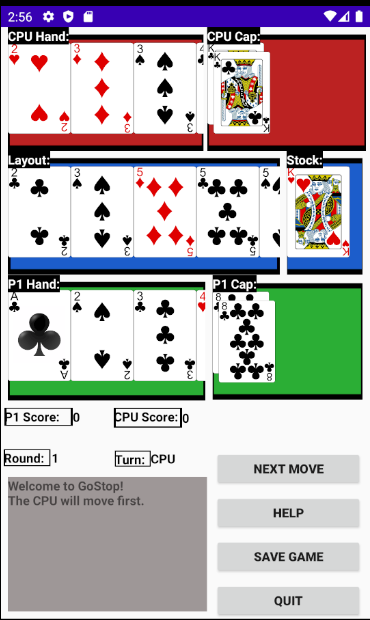
**➜**

**Quitting the game**



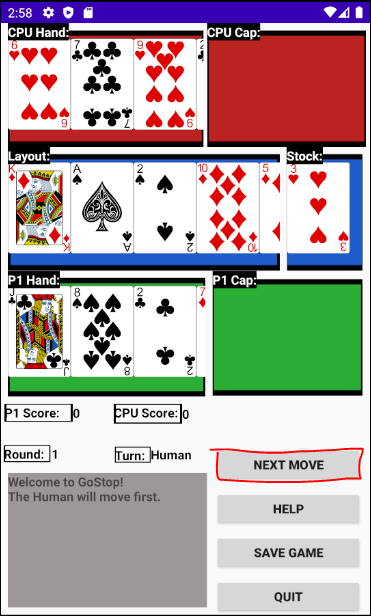
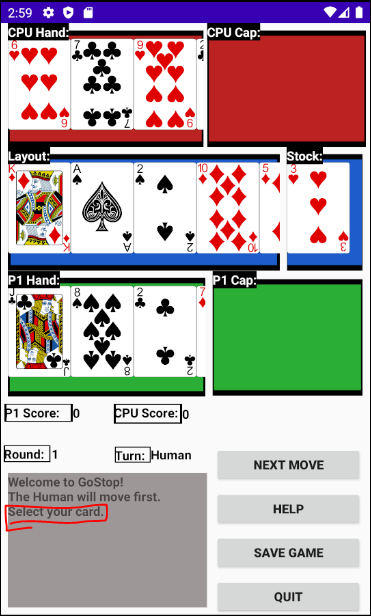
**➜**

**Loading a game**



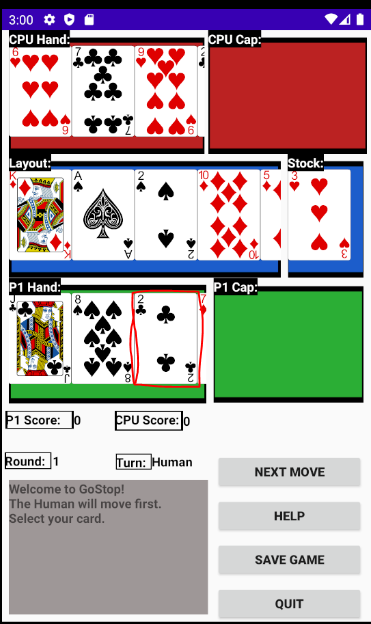
**➜**

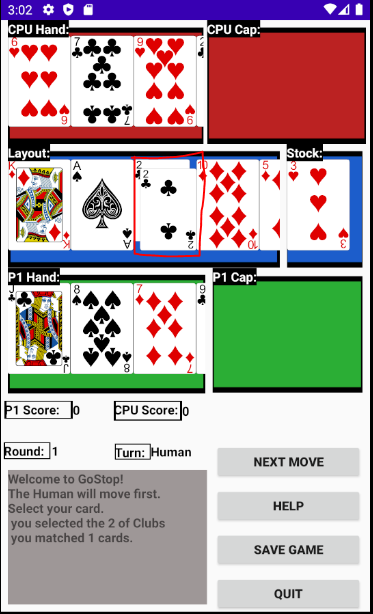
**Selecting any case**



**➜**

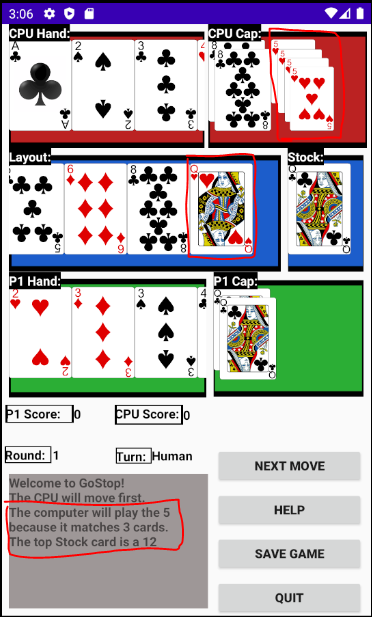
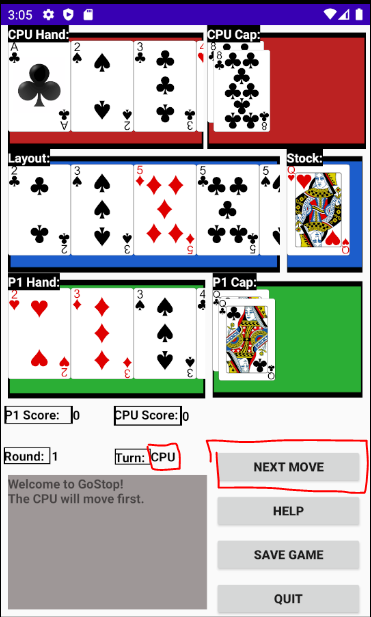
**Next Move (player turn)**





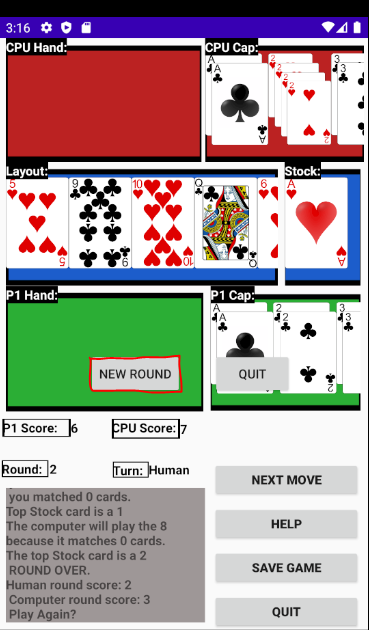
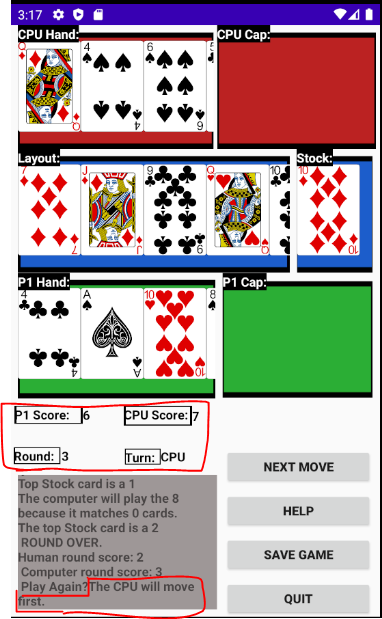
**➜**

**Selecting a P1 hand card**



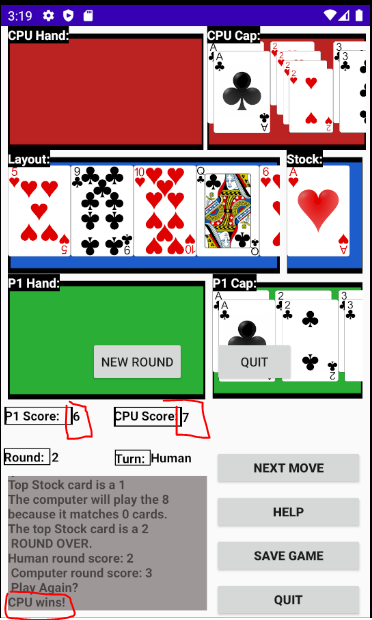
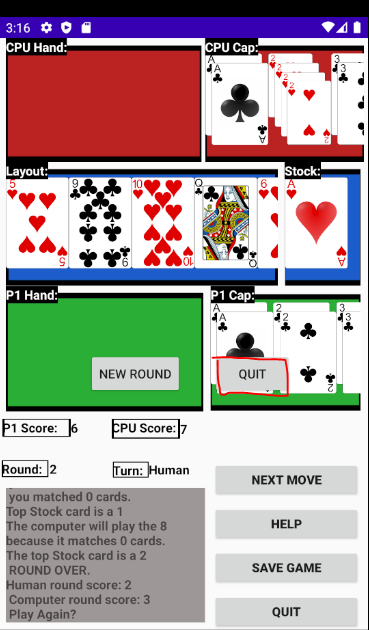
**➜**

**Next Move (CPU Turn)**



**➜**

**New Round (continued game)**

**➜** 

**Quit (End Game)**

**LOG**

**March 27 2020 -** created MainActivity, created logic for newGame button to create a new activity. (20 minutes).

Created gameActivity, the activity to handle the frontend of the game itself. Added handling of “New Game” call from MainActivity (30 minutes.)

Created basic game GUI elements, including spaces for all the cards, labels for these spaces, the chatLog, and the game buttons. (30 minutes. )

Created GUI elements for the scores, the round number, and the turn. (15 minutes).

**March 28 2020**

Implemented Card class and getters and setters for the data members (value, suit) (10 minutes)

Created Client class. Implemented data structure members (layout, stock, cpuHand, p1Hand, capture piles) and other immediately obvious data members (scores, turn, round number) (30 minutes).

Created constructor for Client class, as well as createCards and dealCards functions, responsible for the creation of the deck and the distribution of cards at the start of a new game or new round. (45 minutes).

**March 30 2020**

Refined gameActivity data structure GUI elements to be scrollable windows, and created the ‘draw’ functions which handle iterating through the various data structures and printing the correct drawable accordingly. (120 minutes).

**April 1 2020**

Created nextMove function in gameActivity, which assesses the gamestate and has the client object call the correct client functions as a result. (30 minutes).

Created the ‘playerTurn’ functions (A, B, C, D) which segment the player turn in order to both divide the work done over the course of a human turn between functions, and also to allow the gameActivity class to output to the screen during the turn to create a more informative and responsive GUI (120 minutes).

Created the ‘computerTurn’ functions (A, B, C, D) which perform the same tasks as the playerTurn functions documented above, but for the computer player. (120 minutes).

**April 2 2020**

Added extra functions to client class to support ‘playerturn’ and ‘computerturn’ functions. These functions check the layout for number of matches, find matching indices,

Created SelectedCard and selectedLayout functions in gameActivity class, which handle player selection of cards at the appropriate times (selecting which card to player, selecting which layout index to pair at (if multiple)) Ensured that the selection passes the correct index value to the playerturn function that is called based on the gamestate. (100 minutes).

**April 4 2020**

Refined the conditionals on the playerTurn and computerTurn functions so that they adhere correctly to GoStop rules. (80 minutes with debugging)

**April 11 2020**

Created the gameActivity ‘help’ function and the client class ‘getRecco’ function, which handle the frontend and back end of the help to be given by a computer per the project specifications (90 minutes).

**April 15 2020**

Created the gameActivity ‘save’ function which created a text file in local memory of the current game (60 minutes).

**April 16 2020**

Modified the OnCreate of the gameActivity class to handle “New Game’ and “Load Game” options from the mainActivity. Handles preloaded test cases 1, 2, and 3, as well as the local save (available after saving a game) option from the mainActivity screen.Did not complete the parsing of the text files at this time. (60 minutes).

**April 17 2020**

Completed text file parser for the loading game feature, ensured correct values are loaded into cards and that all data structures are filled correctly. Reformed the constructor for the card class to handle assigning a suit to the cards correctly (80 minutes).

**April 20 2020**

Created newRound function to handle starting a new round after one is completed. Ensured the data structures in the client class are reset appropriately, and the correct player is to move first in the new round (higher round score followed by most high value cards). Created a client class function to handle calculating round scores, and ensured counters incremented correctly (both scores, round number). (120 minutes).

**April 28 2020 (after demo 1)**

Modified data structure access in various client functions to prevent crashes in certain spots, mostly due to accessing out of bounds indices (60 minutes with debugging)

Changed gameState data member assignment at various spots of the gameActivity function call chain as well as modifying predicates in the selectedCard and selectedLayout functions in order to remove so turn order bugs. ( 50 minutes with debugging).

**May 2 2020**

Modified computerTurn functions and check3stack function to prevent incorrect computer behavior during turns, also added functionality to prioritize capturing an already owned pair over other cards in order to score more points. (45 minutes).

Modified data structure behavior when creating or clearing stacks of cards, to prevent incorrect index accessing and improper game developments as a result. (30 minutes).