Nikesh Ghimire

13th October 2019

COM313 - Algorithmic Game Theory

Mid-term Documentation

FIFA Draft Simulator

Documentation

RUNNING THE PROGRAM

PREREQUISITES

Make sure everything mentioned below is installed on your machine before trying to run the application. Failure to have any of these prerequisites will result in errors.

INTERNET CONNECTION IS REQUIRED FOR IMAGES

- Python (version ≥ 3.7.4)
- Python Modules
    - graphics.py
    - openpyxl (via pip)
    - lxml (via pip)
    - requests (via pip)
    - pillow (via pip)

INSTRUCTIONS TO RUN THE PROGRAM

To run the program, launch *"Auction-Simulator.py"* found in the Mid-Term Folder.

It is normal for the program to take a while to load as a gigantic database will be iterated and stored in the RAM while loading.

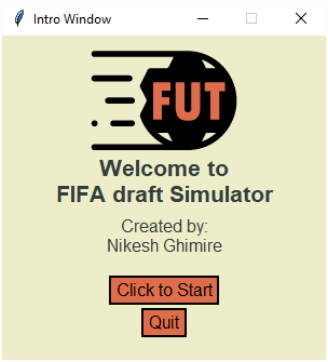
INSTRUCTIONS WITHIN THE APP

The app is very intuitive and self-explanatory; however, each window will be covered in the documentation for the purpose of completeness.

FIFA DRAFT SIMULATOR APP

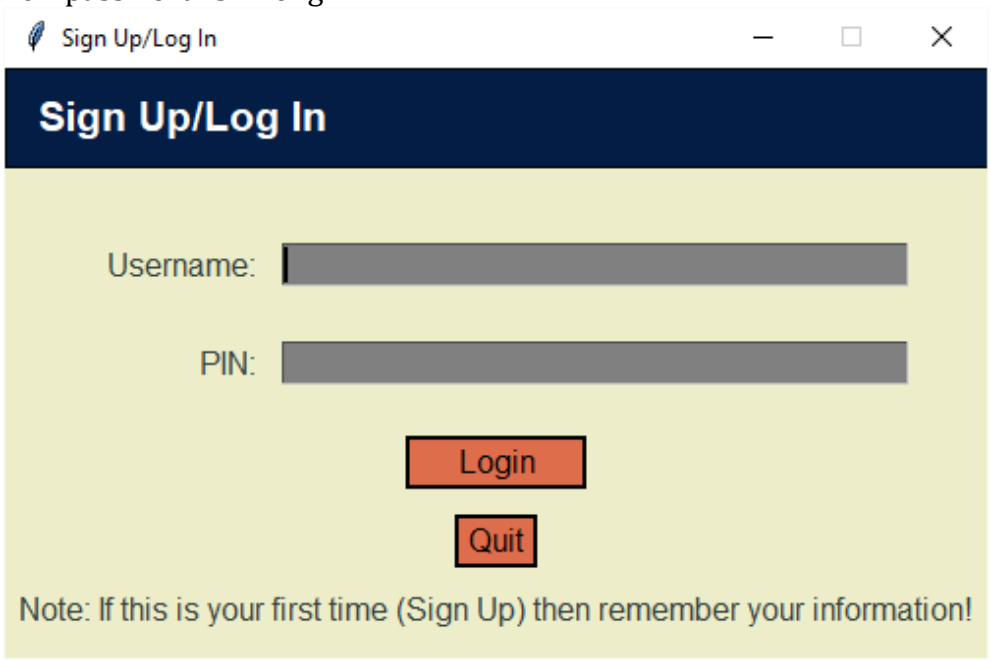
GREETING WINDOW

The user is allowed to either initiate the program or quit out of it at this stage. Hit "Click to Start" to Sign Up or Log In.



SIGN UP/LOG IN WINDOW

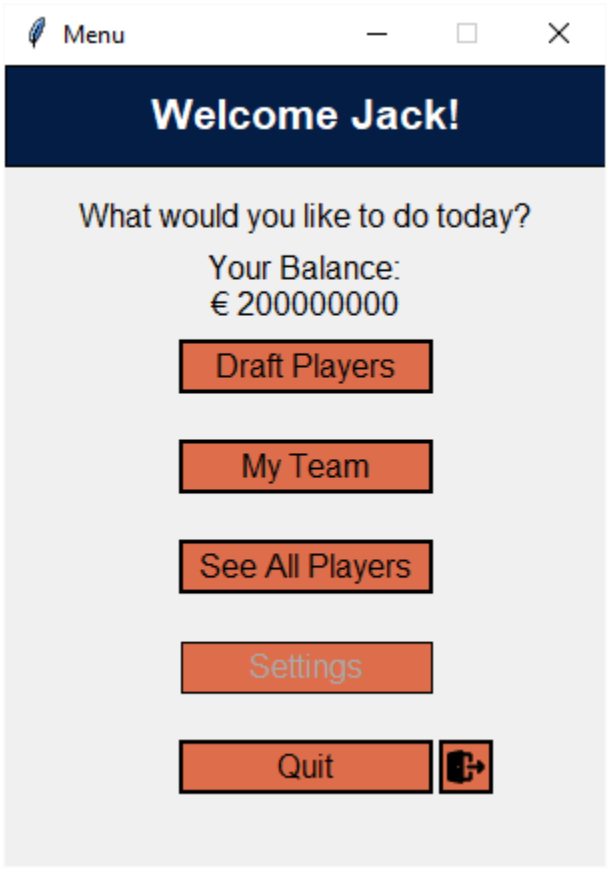
The "Sign Up/Log In" window simply takes in a username and password from the user. If the user already exists in the database, then a correct password will let them in. If the username is in the database, but the password doesn't match their password then the program will tell the user their password is wrong.



If the username entered does not exist in the database (new account), then entering any password will create an account for the new username with the entered password. The user must remember their details (username and password) to access their saved information.

MENU WINDOW

The Menu window is where the user is allowed to access all their operations in the app. The user is presented with their balance at the top so they can easily glance at their current financial position.



The user has the following options:

- Draft Players
    - The User can enter the transfer market and buy players put up for auction
- My Team
    - The user can see and manipulate the list of players he has bought.
- See All Players
    - Allows the user to traverse through the entire database of Players (up to 18000+ players) and view their individual stats.
- Settings (Greyed Out)
    - Allows the user to delete save, change password and change color of interface (future implementation)
- Quit and Log Out
    - Allows the user to kill the application or log Out to let another player Log In.

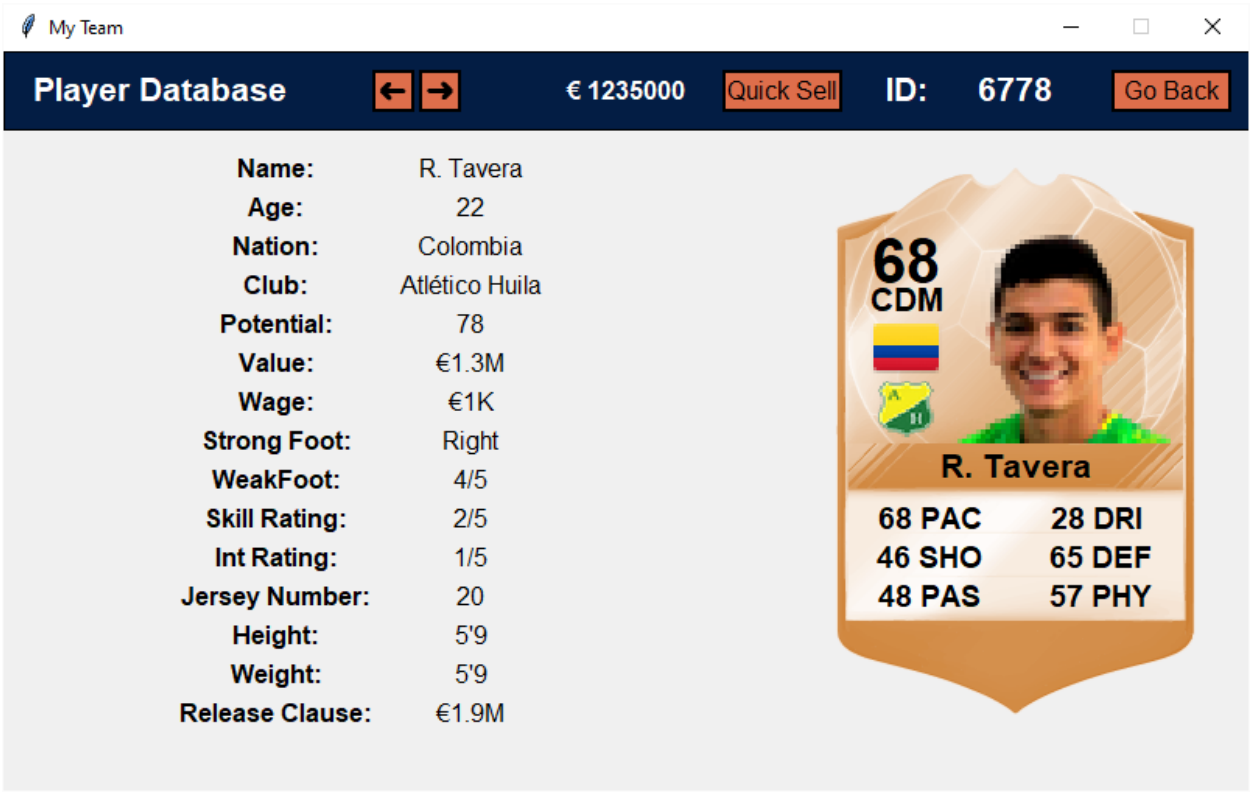
AUCTION WINDOW

The Auction Window allows the player to comb through the list of possible players available for sale. The Auction details section displays information about the player current in the auction. The user can click the View Player button to view detailed statistics of the player. The user is also allowed to skip players if they have no interest in bidding for some players.

The entry box located at the bottom right of the window is where the user enters their bid for the player in question. Once a bid is entered and placed (via clicking the Place bid button), the user is not allowed to bid for the player again this round. The user can win or lose the auction and a text box above the entry box will display the result to the user. If the user won, the player is appended to the list of players owned by the user. The player must then click next player to get to the next player in the auction.

[The Game theory part of this assignment is located in the Auction Section]

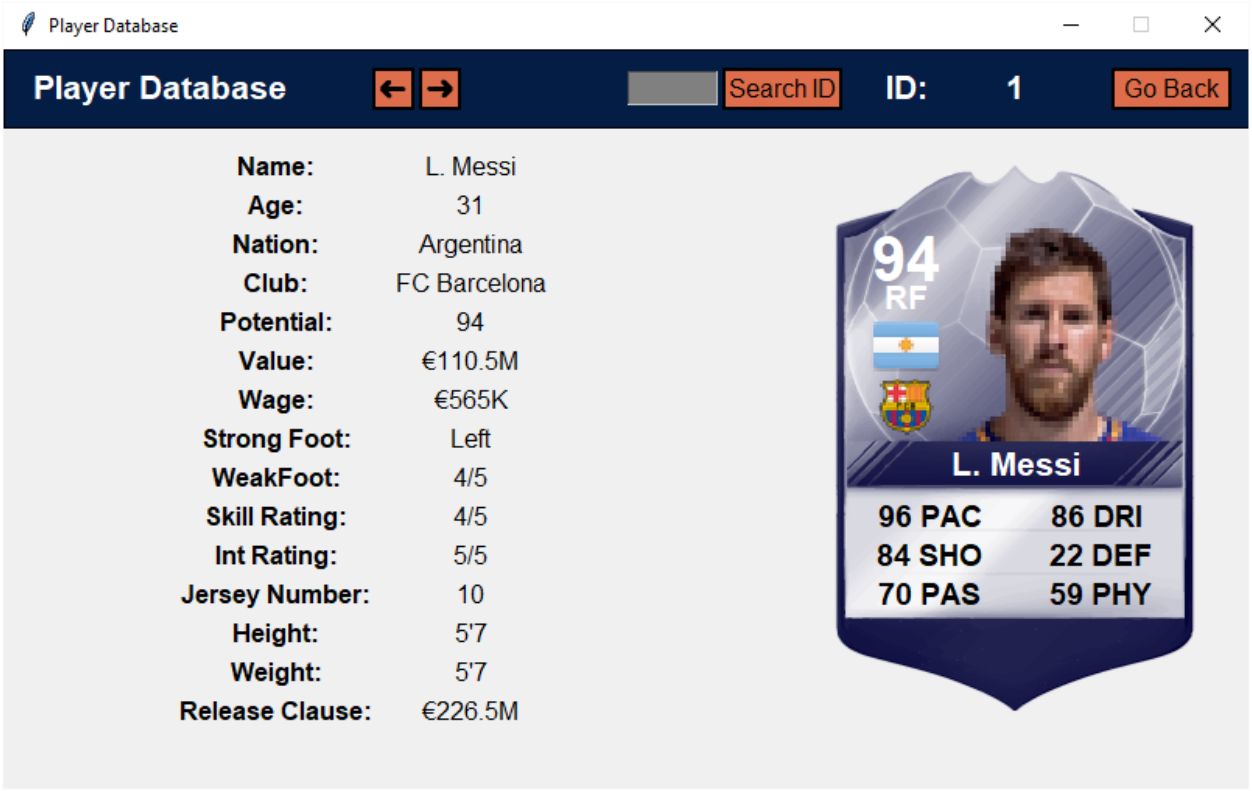
MY TEAM WINDOW



The user can see all the players he has successfully bought through the auction section of the application. Unlike the “See All Players” option from the menu, the user is only allowed to traverse through bought players and has the option to quick sell them for 95% their market value (to the left of the quick sell button).

The user will also able to put their own players up for auction. (future implementation)

SEE ALL PLAYERS



In this window, the user is allowed to go through the entire database of 18000+ players. Many stats of each player can be seen in this window. The user is allowed to search by ID to look for any specific player. The previous and next buttons found at the top can go to the previous or next player by ID.

GAME THEORY IN APPLICATION

If the user wins when bidding, the user is charged the second highest bid throughout the whole auction. This type of auction is called Vickrey's Second Price (VSP) Auction Mechanism. However, an obvious issue with this mechanism is that the user can secure the player if they bid ridiculously high amounts when the computer-generated bids are near the market value of the player. To get around this, the user is punished if he bids a multiplier (hard-coded float value) times the second highest bidder's bid amount. When punished, the user has to pay the high amount he entered for the player instead of the second highest bid in the auction (VSP). Thus, this mechanism rewards fair play and punishes cheap tactics normally called "big-bidders".

As the market value of the player is known, it is easier to simulate an auction knowing that other bidders will bid around the market value of the player. However, if the market value was not known, then an arbitrary generator would be extremely difficult to implement given the size of the player database.

The code used to generate the results of the user winning, losing and the amount the user has to pay to secure the player is shown in the next page.

```
def playerWin(bidVal, num):

    bidList = []

    #Difficulty Modifier
    dMod = 1.5

    #Max number of bidders
    numBidders = 50

    #Getting value of player
    playerValue = player.getValueInt(num)

    #Generating bidders
    for bids in range(randrange(1, numBidders)):
        inflationValue = rand.uniform(0.8, 1.2)
        bidList.append(randrange(round(0.5 * playerValue), round(inflationValue * playerValue)))

    bidList.sort(reverse = True)

    #Printing number of bidders in shell
    print()
    print("Number of bidders: " + str(len(bidList)))
    print()
    print(bidList)
    print()

    #Setting initial value of b2
    b2 = 0

    #Loop to check through list of bids
    for bids in bidList:
        if bidVal <= bids:
            return False
        if bids > b2:
            b2 = bids
        if bidVal >= dMod * b2:
            b2 = bidVal
    return b2
```