Part 1 General Information

General Demo Information:

- List of users (nickname, password, type): here the user list:

```
abcd,ahmet123+,ahmet,ahmet,55,a@gmail.com,/images/user5.png
ahmet,ahmet123+,ahmet,ahjmet,777,gh@gmail.com,/images/user1.png
ahmett,ahmet123+,ahmet,ahmet,55,a@gmail.com,/images/user2.png
ahmettt,ahmet123+,ajhme,ahmet,55,a@gmail.com,/images/user7.png
suvu,suvu123+,suvu,suvu,18,a@gamil.com,/images/user4.png
emir,emir123+,emir,emir,55,a@gmail.com,/images/user2.png
```

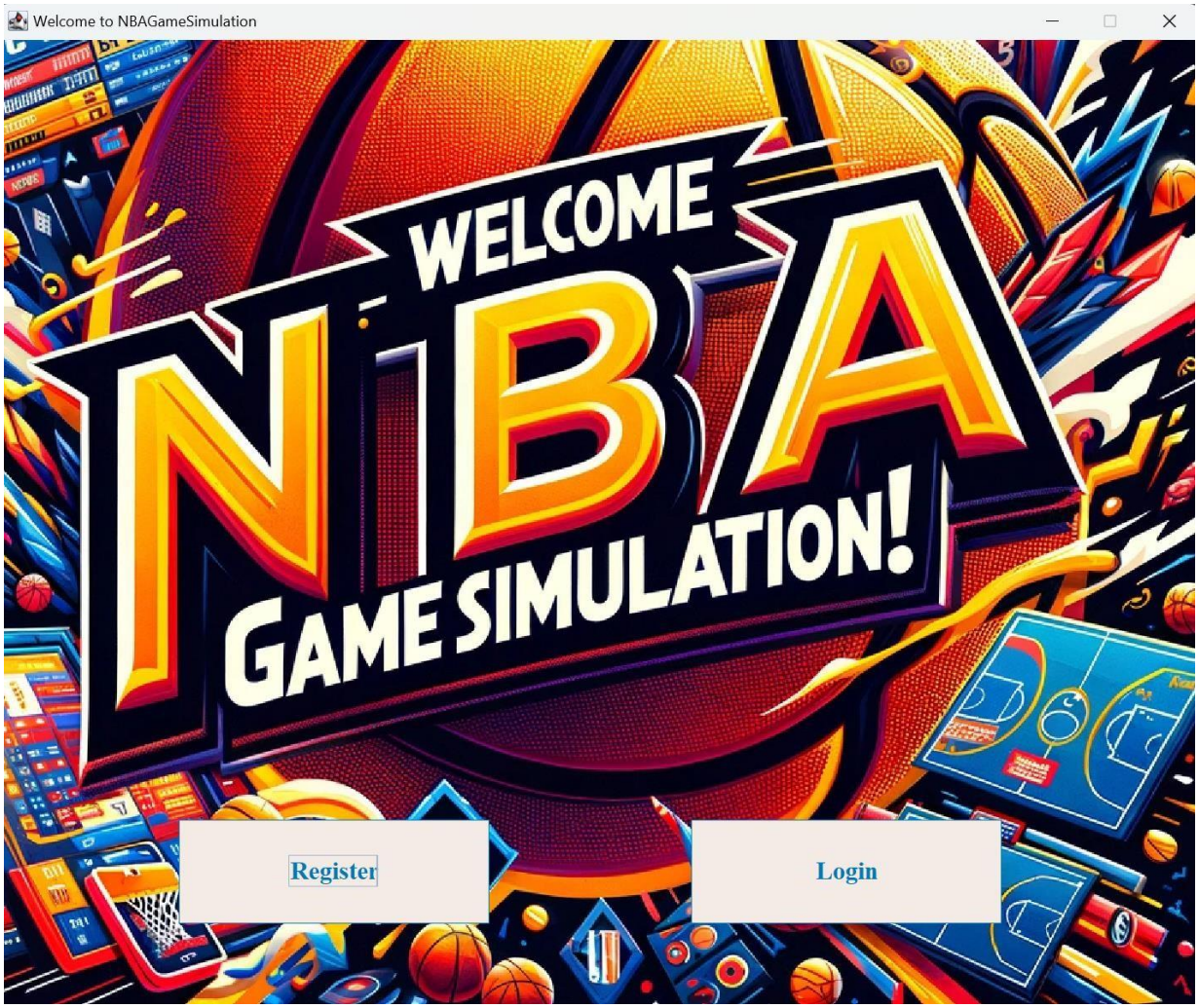
Few users were created to demonstrate the application's features. Also, if you want to check the list of users manually you can check "user.txt" in the project file.

- Information about each user: Each user in the game has a unique nickname, a secure password, and a user type-in this app all user types are 'app users' - that defines their access level within the application.

Application usage information

- **Sign up/Login Guide**

User can start the game and access to login and sign-up pages once they run WelcomePage.java. Welcome Page is seen below:



This page has two buttons, which lead the user to the register page or login page. If the user wants to register and press the register button here what s/he will see

If user clicks the register button, text fields are checked for whether they are valid or not. The user must satisfy these constraints to create an account:

*Password should be at least eight characters, including letters, numbers, and special characters.

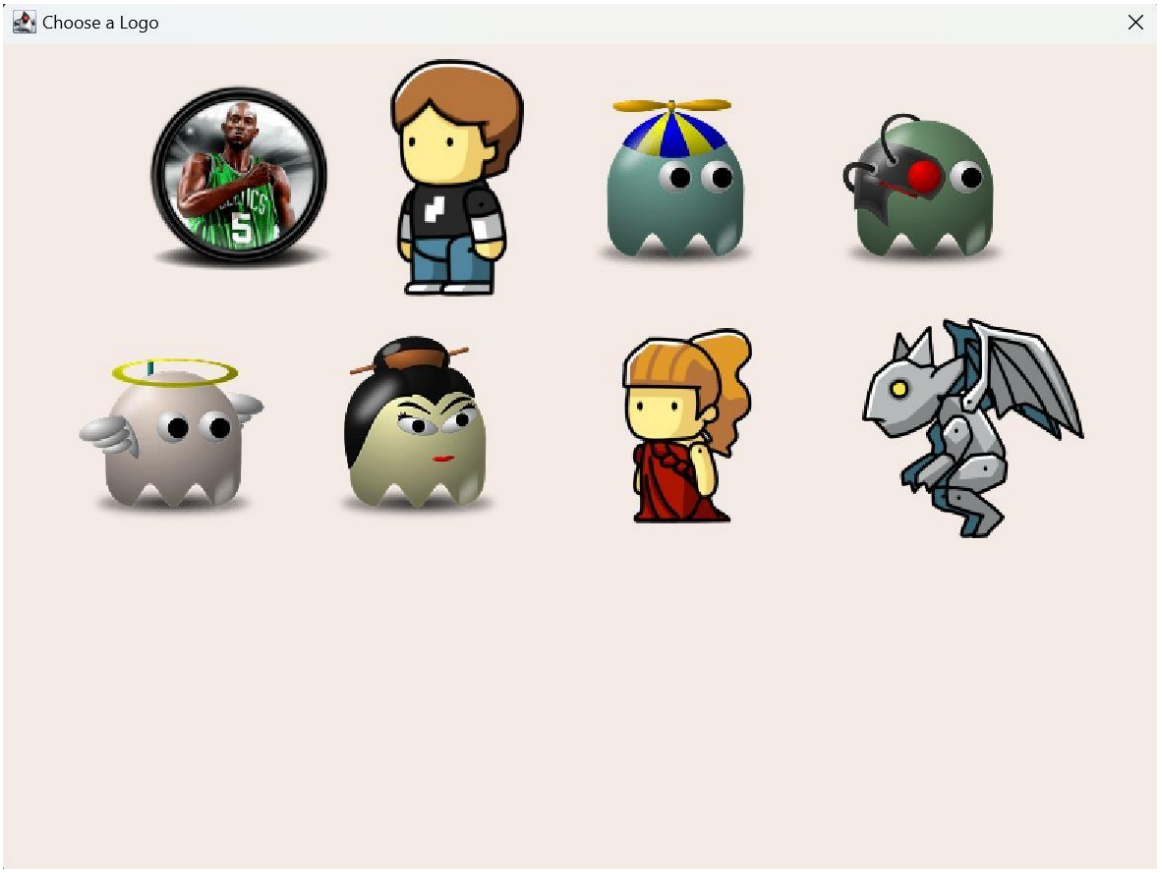
*Name and surname, each should have at least three characters (only letters).
*Username can only include letter and number characters.

*Age must be at least 12

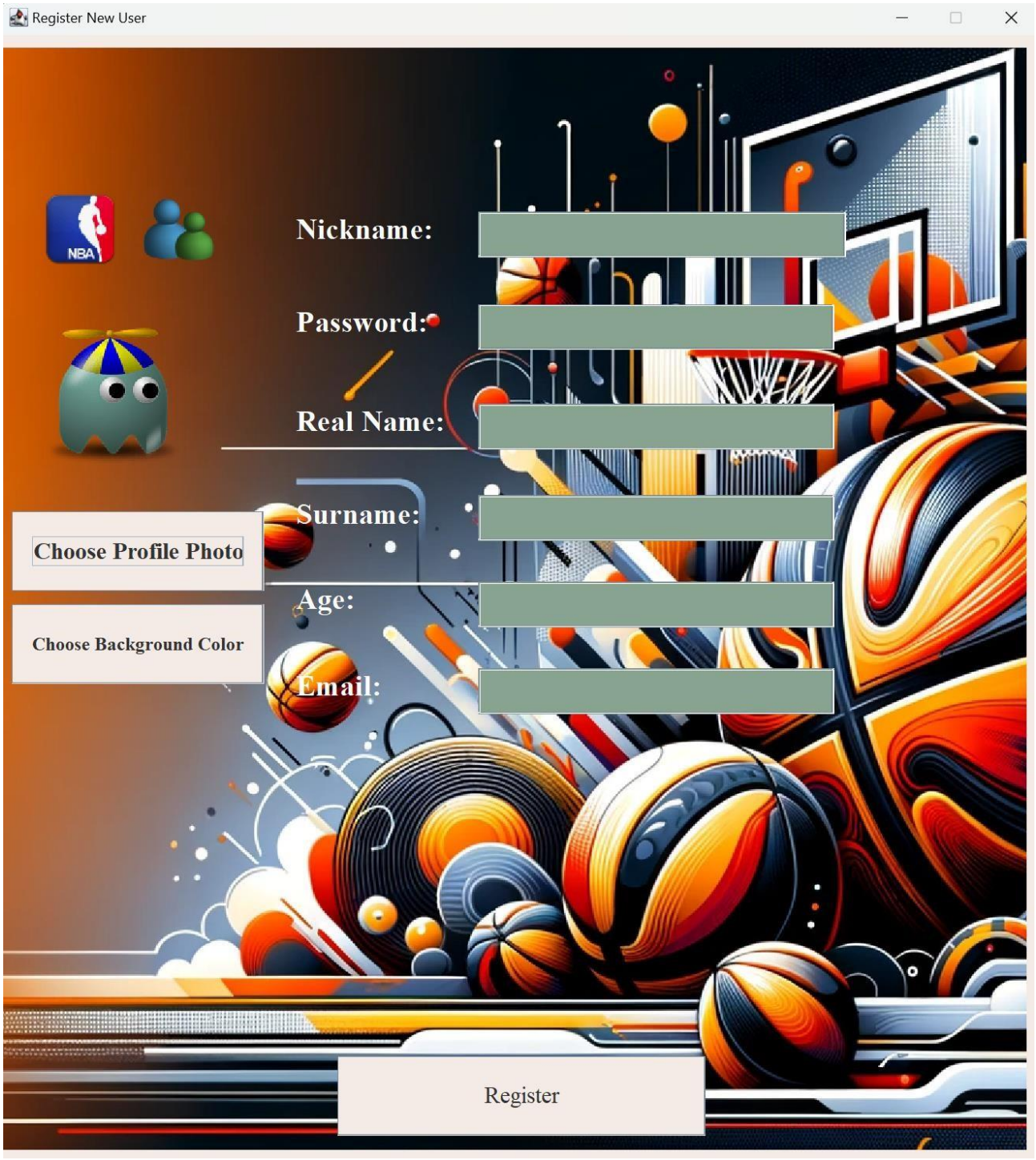
*The email address should be in the correct format. (name@domain.com)

In case of violating any of the above constraints, the system displays a proper message.

Also, on register page user can select a profile photo once pressing “Choose Profile Photo” button.



User can select one of 8 profile photos2 It is not mandatory and if user doesn’t select a profile photo default profile photo will be display rest of the game. Once user selects a profile photo user can see selected profile photo on register page shown below

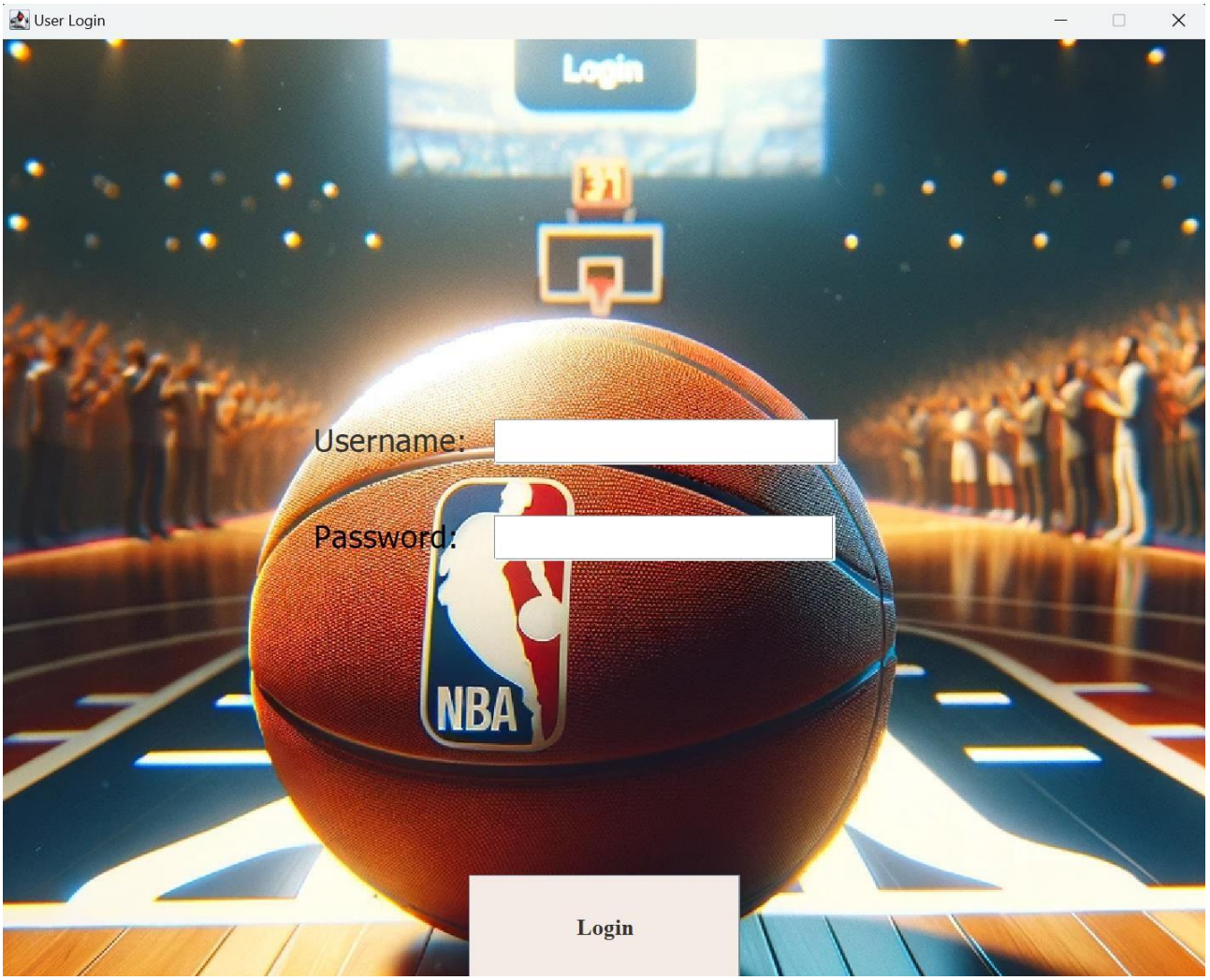


User can change the background color of register page if want to. Here is a photo of what looks like if user presses a Choose Background Color button.


Register New User
Nickname:
Password:
Choose Profile Photo
Choose Background Color
Register
Choose Background Color
Swatches
HSV
HSL
RGB
CMYK
Recent:
Preview
Sample Text Sample Text
Sample Text Sample Text
Sample Text Sample Text
OK
Cancel
Reset

• User’s Guide:

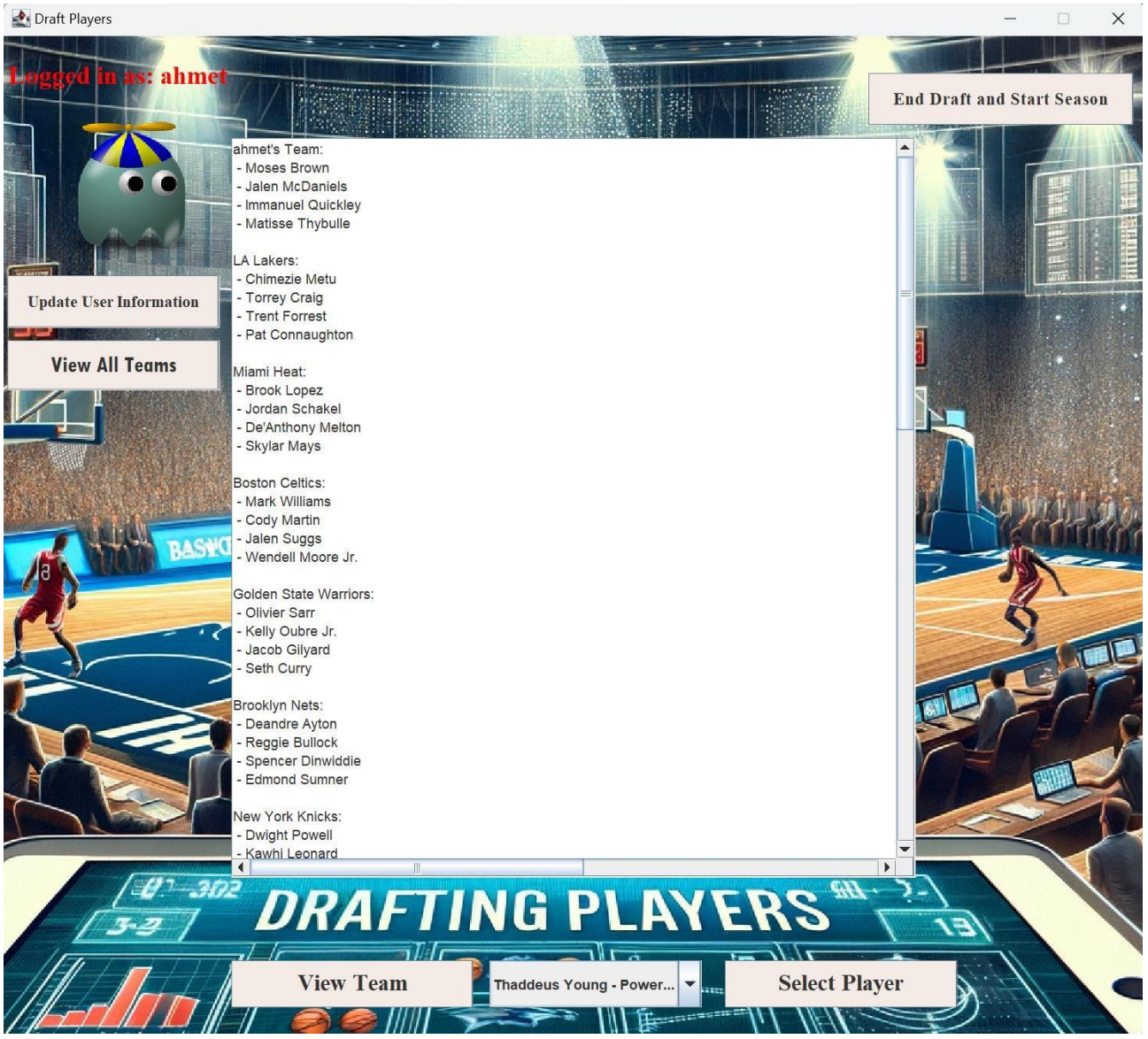
When user registers successfully register a pop-up message shows up which says user to s/he successfully registered then login page opens



The user's username and password are checked to see if they match the stored information, and the user is logged in.

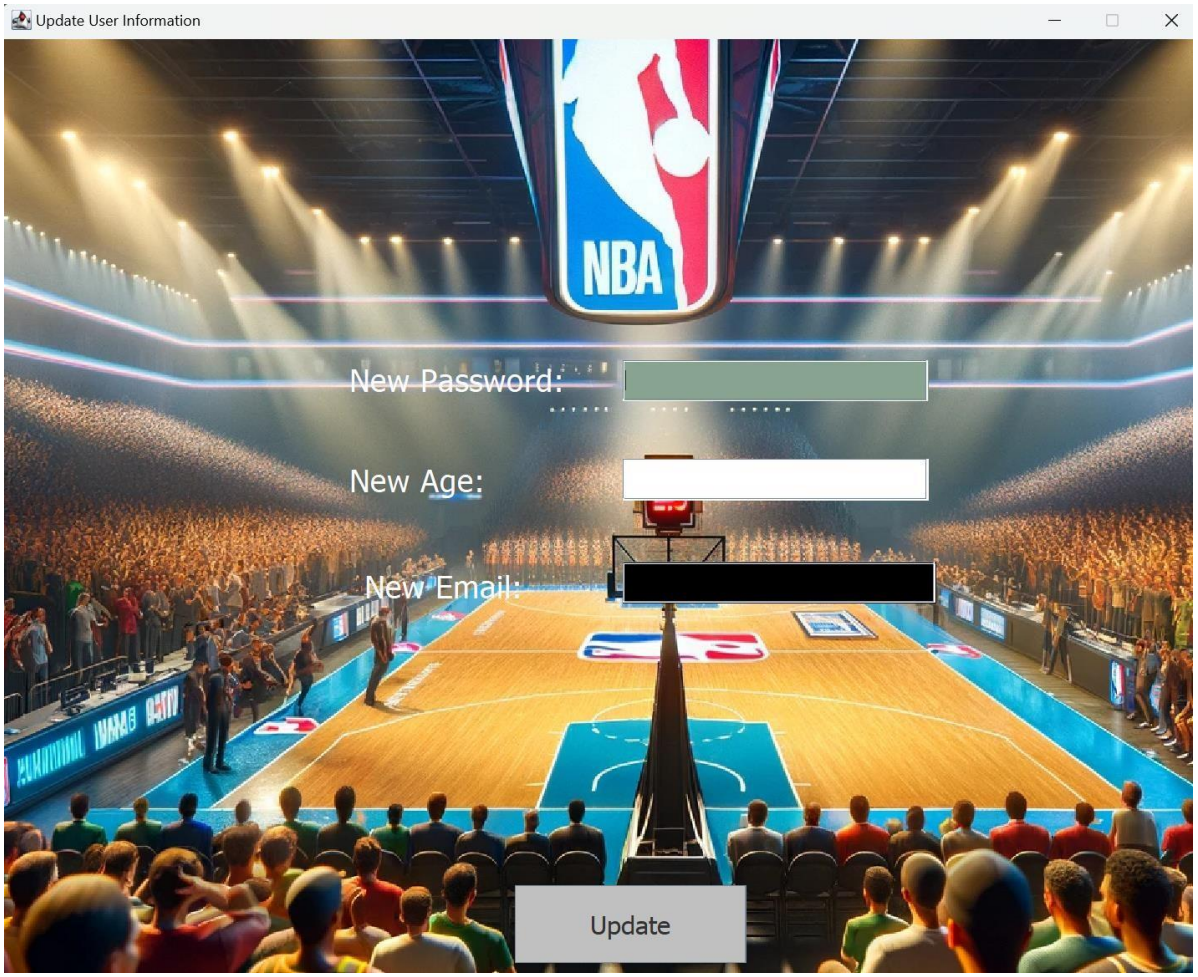
• Drafting

User can access draft page once they login successfully. On draft page:



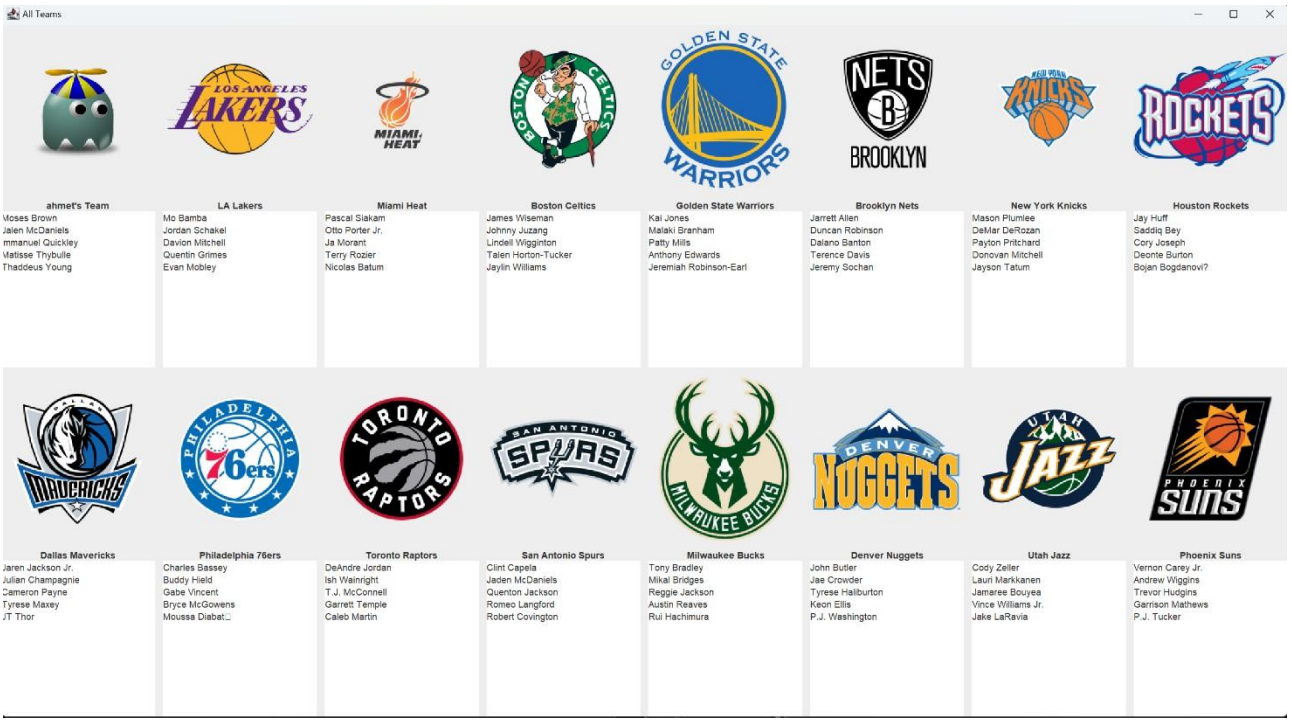
1) At the top left corner, there is a notification stating "Logged in as ahmet" indicating the user's profile name. Also, user can see his/her profile photo which they selected on register page.
2) In the center left of the screen on drafting page, there's a list of basketball teams with associated player names. In the middle, a draft list is shown where players have been selected for different teams. For example, the Boston Celtics can have players like Herbert Jones and Dyson Daniels listed, while the Toronto Raptors have Bam Adebayo and Kyrie Irving, among others.
3) At the bottom, there are navigation buttons or options such as "View Team," and "Select Player," allowing user to view his team and players statistics, and also select a player for his team.

4) In the top right corner, there is a button labeled "End Draft and Start Season," which finalizes the draft choices and begins the basketball season within the game.
5) Below user's profile photo, there is a button titled "Update User Information," suggesting the user can edit their profile details, if user presses to that button here what will s/he see



On this page, user can change password, age and email address. These text fields are also checked for whether they are valid or not. If one of them is not valid according to the constraint mentioned above system displays a proper message.

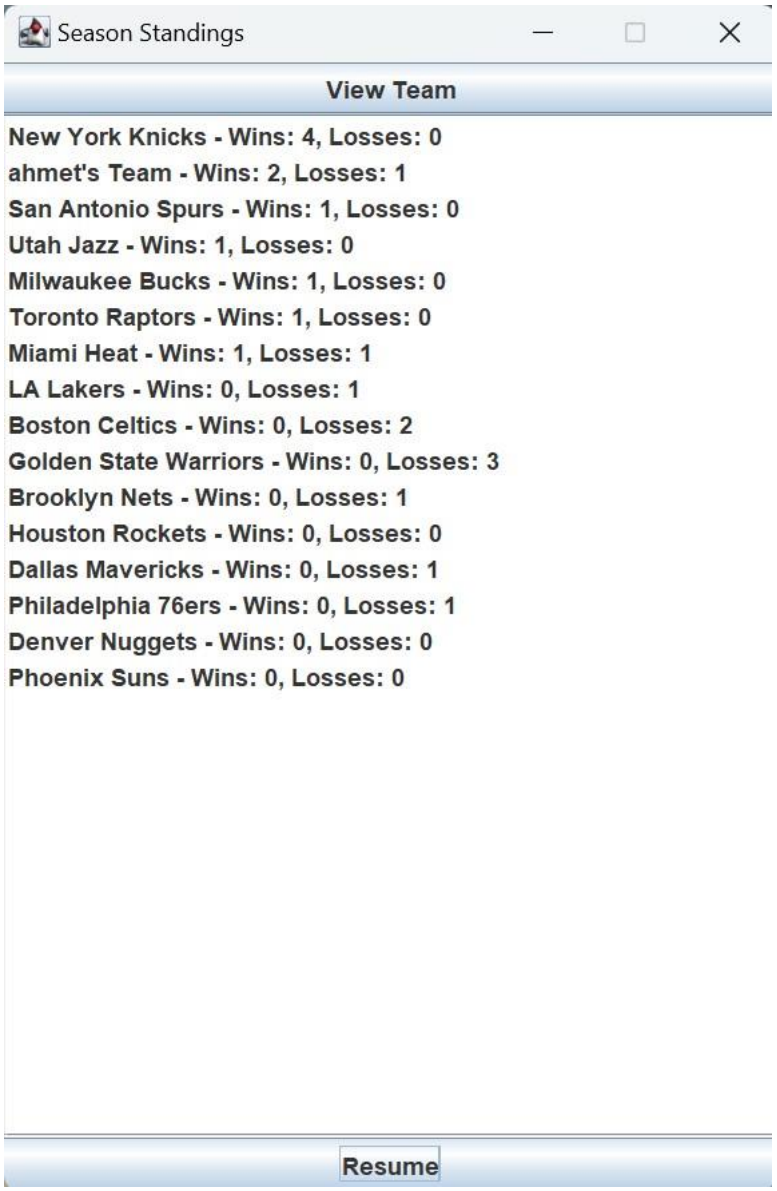
6) Below update user information button there is a button for view all teams with their logos[3] and players which called “View All Teams” if user press that button here what will s/he see



In this example user-in this case “ahmet”- can see all teams with their logos and selected players.

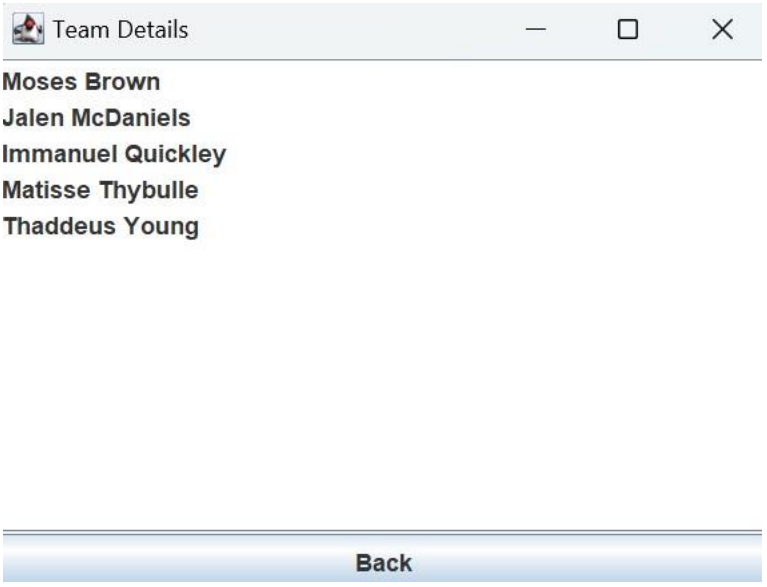
- **MatchMaking:**

1) Once user press the “End Draft and Start Season” button season starts automatically. Each team plays 30 matches and they pick randomly the team they play. User can see his/her team in real time while is season is happening.

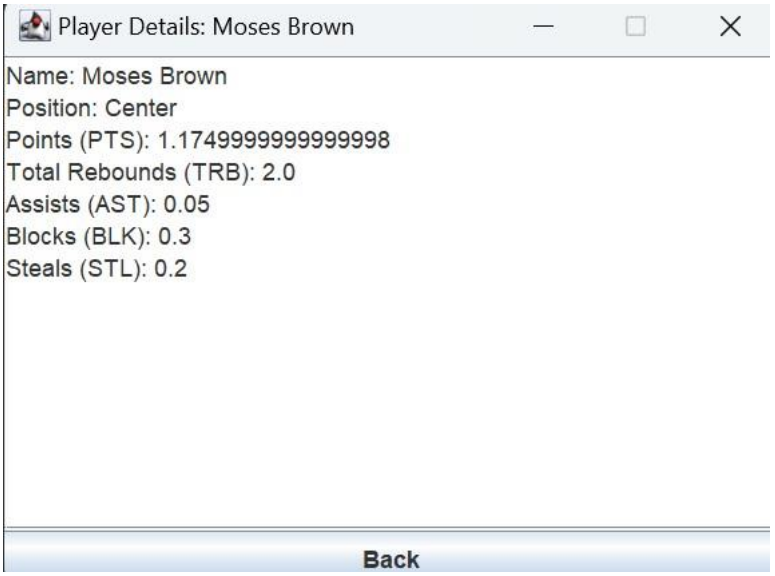


Like in this example user can see all teams and their wins and losses. if user wants to pause the simulation there is a pause button for it. If user wants to resume the simulation, s/he must press resume.

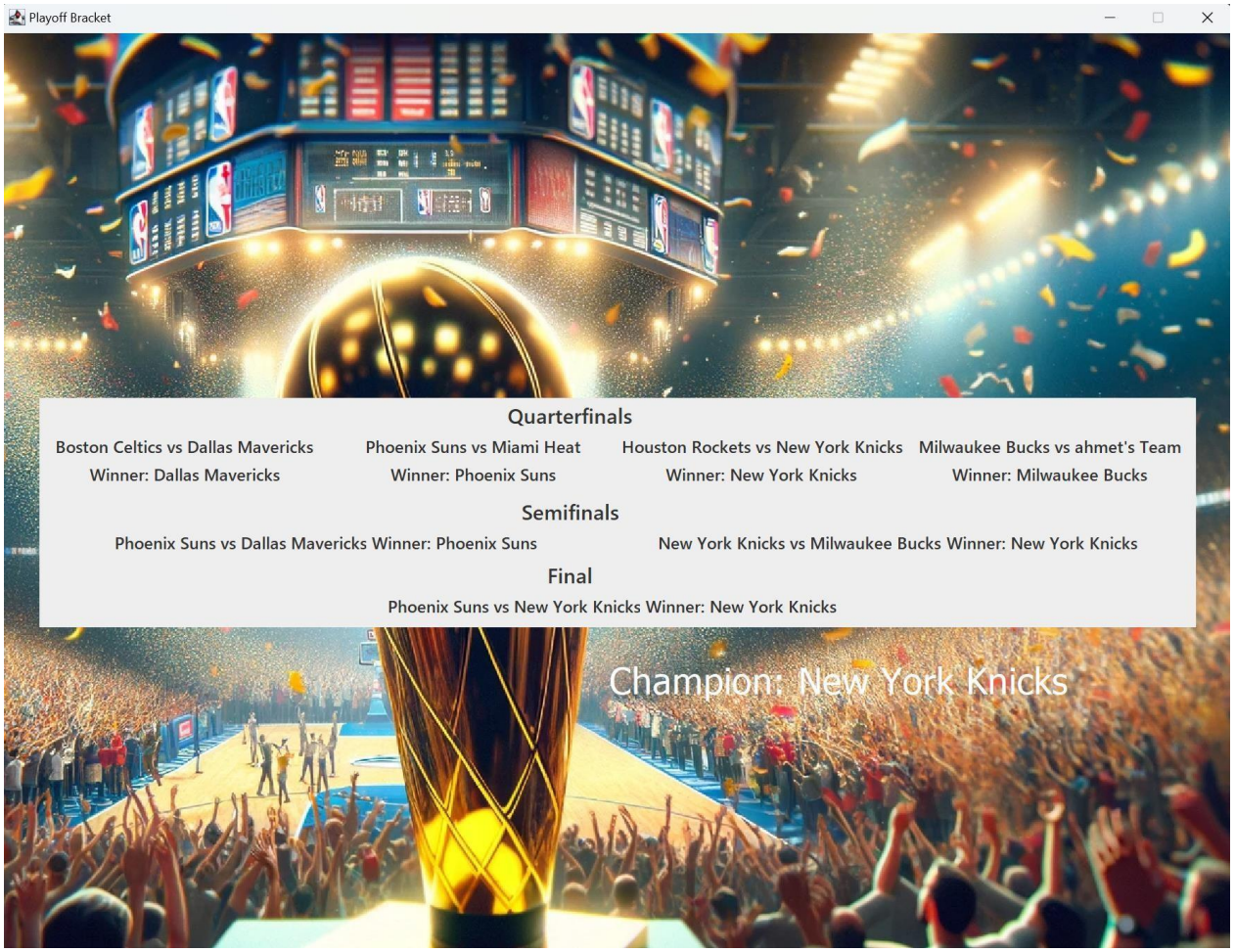
2) Also, there is a “View Team” button which allows user to view their team and players



3) If user clicks one of their players s/he can see information about that specific player such as position assist blocks etc.



4) Once every team plays all season matches playoff page opens up



This page's background photo created with AI[1]

In playoff page user can see all qualified teams, playoff tree and champion team.

- Scoring:

To calculate the score of a player in one instance, for each of the data values, system randomly select an integer in the range [VALUE - 5, VALUE + 5],
"5" is the same for the different data values above. none of the integers go below 0. Then system multiply this randomly selected integer by the weight corresponding to that data value and adds these together for all of them. The system rounds to the nearest whole number.

For example, if a player has PTS = 10, TRB = 3, AST = 6, STL=3, BLK=1, the weights are different, and N is always 5, we might get 0.3 * 12 (PTS) + 0.1 * 2 (TRB) + 0.5 * 5 (AST) + 0.1 * 0 (STL) + 0.1* 3 (BLK) = 6.6, rounded up to 7

Positions	Points	Rebounds	Assists	Blocks	Steals	Total
Center	0.15	0.35	0.1	0.3	0.1	1
Shooting Guard	0.35	0.1	0.15	0.1	0.3	1
Small Fordward	0.25	0.25	0.2	0.15	0.15	1
Point Guard	0.3	0.1	0.4	0.1	0.1	1
Power Fordward	0.2	0.3	0.15	0.25	0.1	1

- **Logging**

to check season match results with teams scores, playoff results and all user's information, user can check match_result.txt, playoffs_result.txt, and user.txt.

```
89 Dallas Mavericks vs Phoenix Suns - Winner: Phoenix Suns, Loser: Dallas Mavericks
90 Golden State Warriors vs ahmet's Team - Winner: ahmet's Team, Loser: Golden State Warriors
91 Utah Jazz vs Milwaukee Bucks - Winner: Milwaukee Bucks, Loser: Utah Jazz
92 New York Knicks vs LA Lakers - Winner: LA Lakers, Loser: New York Knicks
93 Phoenix Suns vs Milwaukee Bucks - Winner: Milwaukee Bucks, Loser: Phoenix Suns
94 ahmet's Team vs LA Lakers - Winner: ahmet's Team, Loser: LA Lakers
95 ahmet's Team vs Milwaukee Bucks - Winner: ahmet's Team, Loser: Milwaukee Bucks
```

Example of playoff_tesult.txt

```
1 Champion: San Antonio Spurs
2
```

Example of last_champion_team.txt

```
528 Houston Rockets vs ahmet's Team: 8,820000 - 12,000000, Loss
529 LA Lakers vs Dallas Mavericks: 14,332500 - 17,000000, Loss
530 ahmet's Team vs Dallas Mavericks: 13,230000 - 17,000000, Loss
531 San Antonio Spurs vs New York Knicks: 16,537500 - 12,000000, Win
532 Milwaukee Bucks vs Brooklyn Nets: 16,537500 - 12,000000, Win
533 San Antonio Spurs vs Utah Jazz: 16,537500 - 18,000000, Loss
534 Dallas Mavericks vs Philadelphia 76ers: 18,742500 - 20,000000, Loss
535 New York Knicks vs ahmet's Team: 13,230000 - 12,000000, Win
536 Boston Celtics vs LA Lakers: 23,152500 - 13,000000, Win
537 LA Lakers vs Miami Heat: 14,332500 - 18,000000, Loss
538 Golden State Warriors vs Milwaukee Bucks: 7,717500 - 15,000000, Loss
539 Toronto Raptors vs Miami Heat: 18,742500 - 18,000000, Win
540 Denver Nuggets vs Toronto Raptors: 18,742500 - 17,000000, Win
541 Houston Rockets vs Golden State Warriors: 8,820000 - 7,000000, Win
542 Denver Nuggets vs Milwaukee Bucks: 18,742500 - 15,000000, Win
543 Denver Nuggets vs Utah Jazz: 18,742500 - 18,000000, Win
544 San Antonio Spurs vs Milwaukee Bucks: 16,537500 - 15,000000, Win
545 Utah Jazz vs Denver Nuggets: 19,845000 - 17,000000, Win
546 New York Knicks vs Toronto Raptors: 13,230000 - 17,000000, Loss
547 ahmet's Team vs Brooklyn Nets: 13,230000 - 12,000000, Win
548 Miami Heat vs Boston Celtics: 19,845000 - 21,000000, Loss
549 ahmet's Team vs Denver Nuggets: 13,230000 - 17,000000, Loss
550 Houston Rockets vs Denver Nuggets: 8,820000 - 17,000000, Loss
551 Utah Jazz vs Philadelphia 76ers: 19,845000 - 20,000000, Loss
552 ahmet's Team vs Miami Heat: 13,230000 - 18,000000, Loss
553 Houston Rockets vs San Antonio Spurs: 8,820000 - 15,000000, Loss
554 New York Knicks vs LA Lakers: 13,230000 - 13,000000, Win
555 Philadelphia 76ers vs Utah Jazz: 22,050000 - 18,000000, Win
556 San Antonio Spurs vs Denver Nuggets: 16,537500 - 17,000000, Loss
557 Brooklyn Nets vs ahmet's Team: 13,230000 - 12,000000, Win
558 Miami Heat vs Utah Jazz: 19,845000 - 18,000000, Win
559 Miami Heat vs Toronto Raptors: 19,845000 - 17,000000, Win
```

Example of match_result.txt

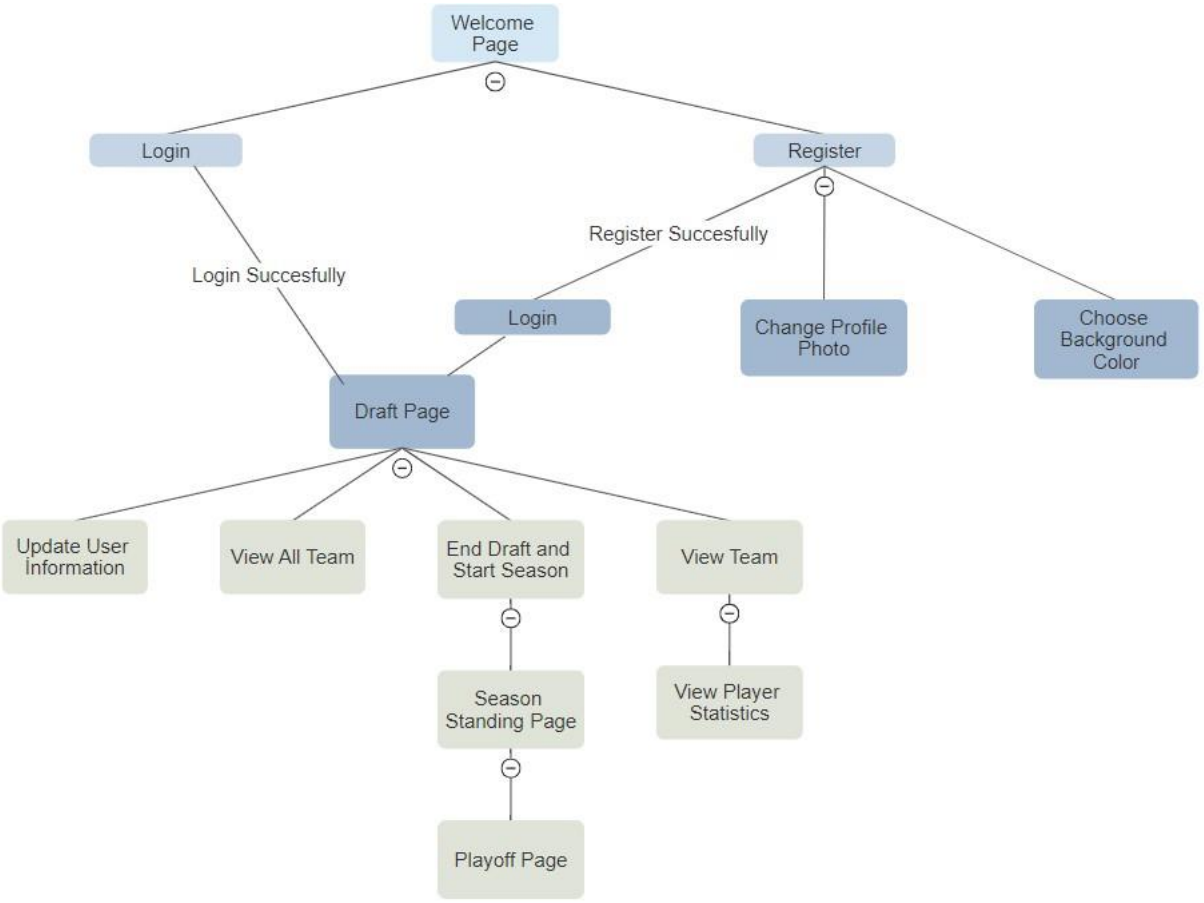
```
1 suv12,ahmet123+,Suveybe,Ugut,20,sugut22@gmail.com,/images/user5.png
2 doga_k,ahmet123+,Doga,Kalkandelen,19,dkalkandelen22@gmail.com,/images/user1.png
3 ahmet,ahmet123+,Ahmet,Benlice,20,abenlice22@gmail.com,/images/user2.png
4 keremyilmaz,ahmet123+,Kerem,Yilmaz,21,kyilmaz22@gmail.com,/images/user7.png
5 sarisacli,suvu123+,Aykut,Bir,18,a@gamil.com,/images/user4.png
6 emir,emir12!,Emir,Ozdemir,19,eozdemir22@gmail.com,/images/user2.png
```

Example of user.txt

Part 2 Design

Project Design Description:

- Application Flow



Ex: Application Flow between GUI classes

- Basic Class Information

1) FileManager.java: Central to data persistence, this class handles all file operations, including saving and loading user profiles and game records.
2) LoginPage.java & RegisterPage.java: These are the gateways to the game, handling user authentication (login) and account creation (registration). They interact with FileManager to validate or store user credentials.

3) UpdateUserPage.java: Allows users to update their data. This class is critical for maintaining user engagement by enabling personalization of user profiles.

4) User.java: Represents the user's data model with attributes like username and password. It is used across the application to reference the logged-in user.

5) WelcomePage.java: Acts as the application's landing page, directing users to either login or register. This is where the user journey in the application begins.

6) Player Classes (Center, PointGuard, etc.): These classes, derived from the Player class, represent different positions in basketball, each potentially having unique attributes or methods that reflect their roles in the game.

7) Player.java: The superclass for player types, encapsulating attributes like scoring averages and defensive capabilities. It forms the backbone of the player-related operations.

8) PlayerDetailsGUI.java & TeamDetailsGUI.java: These GUI classes provide detailed views of individual players and teams, enhancing user engagement by offering insights into player performance and team composition.

9) PlayerManager.java: Responsible for loading player data from csv file and creating player objects. This class is crucial for the game's drafting process.

10) Team.java: Models a basketball team, holding a collection of Player objects. It plays a vital role in the game simulation, where team performance is evaluated.

11) DraftGUI.java: The interface for the player draft, where user and computer select players to form teams.

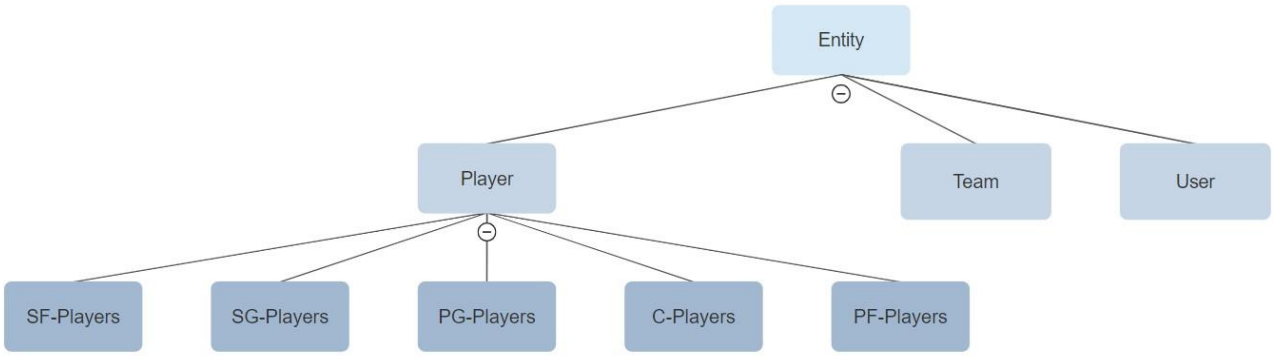
12) DraftManager.java: Manages the logic behind the player draft, ensuring that teams are balanced and that the drafting process is fair and engaging.

13) GameSimulator.java: Simulates the actual games based on team composition and player performance. It uses a scoring algorithm to determine game outcomes

14) PlayoffBracketGUI.java: Visualizes the playoff stages, It displays matchups and outcomes.

15) SeasonStandingsGUI.java: Shows the regular season standings, updating after each round of games. It keeps players informed about their team's performance and standings in the league.

- Inheritances, type hierarchies, interfaces, abstract classes.



Different types of players inherit from a common Player class, allowing for polymorphic behavior.

- GUI components

As a UI designer tool, the window builder of the Eclipse is used for designing the positions and properties of the components. I also use built-in JColorChooser to change background color of RegisterPage.java. Extensive use of Swing components to create an interactive and intuitive user interface. Each GUI class is tailored to specific functionalities, like drafting players, viewing team details, and navigating through the game's stages.

- **CSV File Processing Details**

CSV file processing involves several key steps. Initially, the PlayerManager class reads player statistics from a CSV file using the readPlayersFromCSV method, which employs a BufferedReader to read data line by line. The data is split into an array of strings using the delimiter (comma), and player objects are instantiated with these attributes. This data is then

used to populate the players list with player entities, each corresponding to a row in the CSV.

Furthermore, the removeDuplicatesAndAverageStats method ensures that any duplicate entries for players are resolved by averaging their statistics, which prevents redundancy in the game's dataset.

The FileManager class handles the output, writing game results and draft choices back to CSV files to record the simulation's progress and outcomes.

- **Drafting Method**

  Key Features of Drafting Method:

  Team and Player Lists: Manages lists of teams and available players, allowing for dynamic drafting.

  Position-Based Selection: Ensures teams pick players in unique positions, promoting strategic diversity.

  User Interaction: Enables users to select players for their team, enhancing engagement.

  Computer Drafting: Automates player selection for non-user teams, based on available positions and user choices.

  Position Validation: The system checks if a player's position is already filled in a team before selection.

  User Feedback: Provides ongoing information about available players and team composition.

  Seamlessly integrated into the game's GUI, the DraftManager offers an interactive and intuitive drafting experience.