CASINO ROULETTE APPLICATION

Made by Javier Carrillo González – 71725235B – UO269412

Table of contents

Introduction

For this delivery, we had to program an application that would resemble one of the games that might be found in a casino, that being the roulette.

We also need to include some services the casino offer, adding a shop in which you can either add money to the casino, spend it on chips and even order drinks.

Furthermore, since we need to register data from every user who plays the game, this application also supports many users, saving the chips they bought or earned, as well as the money they have spent playing.

▪ *Graphic user interface prototipes*

Overall, our prototype didn't change much between the seminars, just a couple of details regarding different cases we didn't think about. However, when the real application was made, I approached some things differently, applying some techniques we've learned since the last prototype.

❖ Main Window

It was created just as the big title of the app, just to make the user enter the casino.

This window received no changes between the two prototypes designed in class, staying the same. We wanted to just make a clear window, with two big buttons that would make the user want to either register or log in.

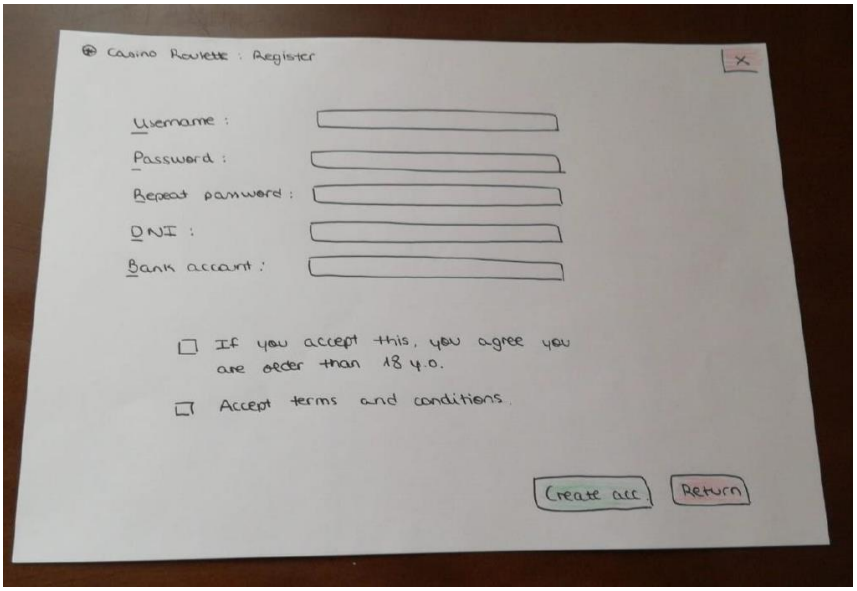
However, when developing the program, I decided to keep the buttons at an average size, as well as moving them all to the bottom right of the window, because it felt more consistent with the window in general. I also moved the image to the centre of the window, giving it a bigger importance.

❖ Register window

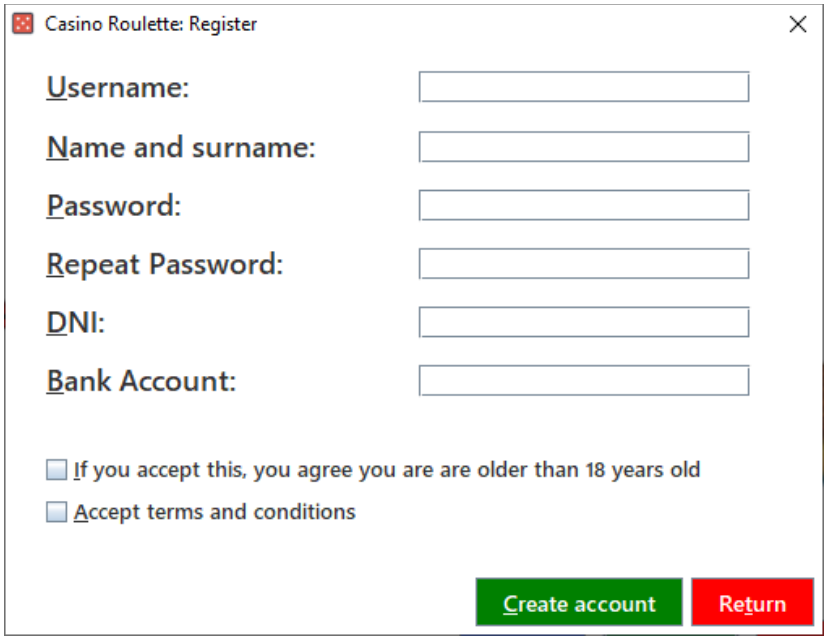
The use of this window is self-explanatory, it's the way to enter the casino if you never accessed it before.

This window received one change during seminars, and it was handling the cases where the user left some fields empty, didn't check the checkboxes or the passwords didn't match by showing a dialog explaining such error. In the first version we didn't open a dialog with this info.

Besides that, the window itself didn't change, so it stayed the same between the seminars.



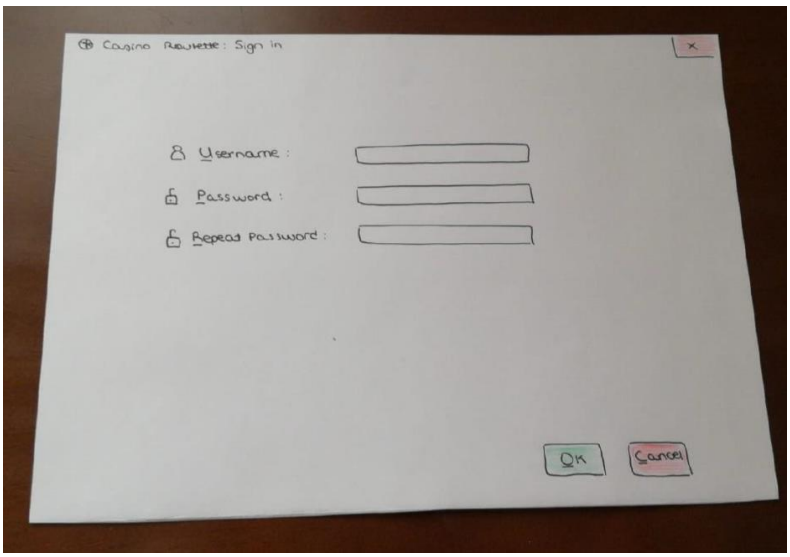
In the final version, a new field was added since we also needed to register the name and surname of the user. That is the only difference between the previous version and the final one



❖ SignIn Window

The functionality of this window is the same as the register window, just accessing the app if you are already an user.

The changes that suffered this window between versions are the same as the register window, handling different cases and showing such info in different dialogs; however, the window itself didn't change.

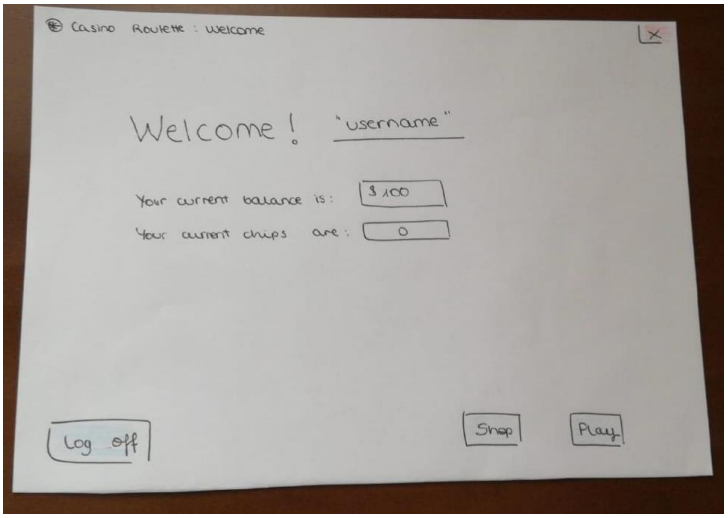


While in the register window, a new field name and username was added, in this window it wasn't necessary since the username and the password are enough to log in, making this window the same as the last seminar version.

❖ Welcome Window

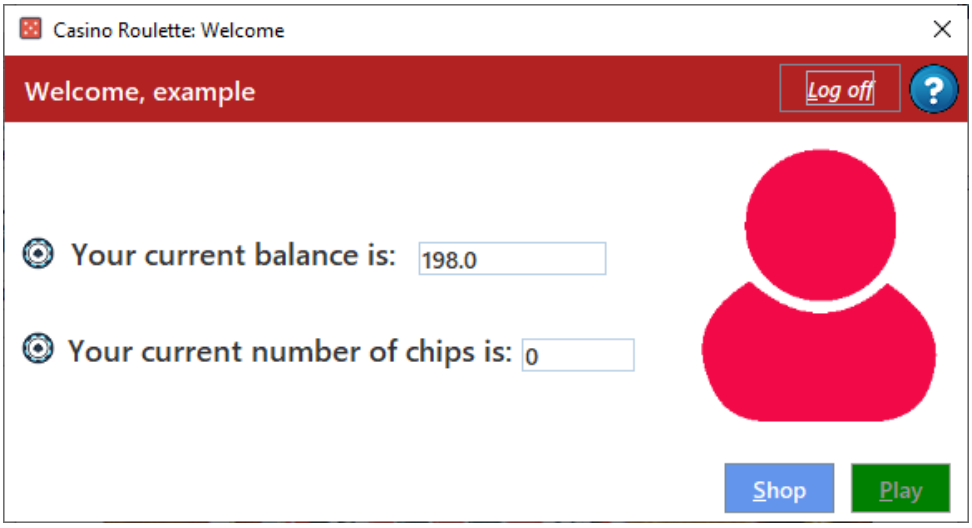
This window was made to fulfil the job of being a gateway from the registration windows and the casino itself. We also decided to display general information about the user so they can get a better understanding about their status before entering "the game".

Between the first seminar and the second, this window just suffered a small change, and it was the addition of a log off button.



In the final version, no new additions were made, but overall, a cleaner look was given to this window. I decided to move the "Welcome, user" and the log off button to the top of the window, making a pseudo-header that is very used in different websites. An image was also added to make the window a bit livelier. No further change was made.

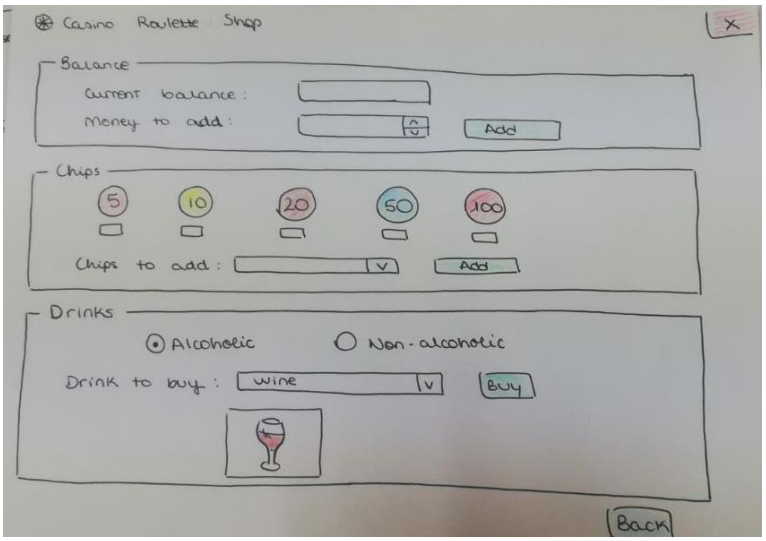
Just as a preventive measure, trying to close the window will function the same as the log off button, successfully logging off the user and updating its information in the .dat file.

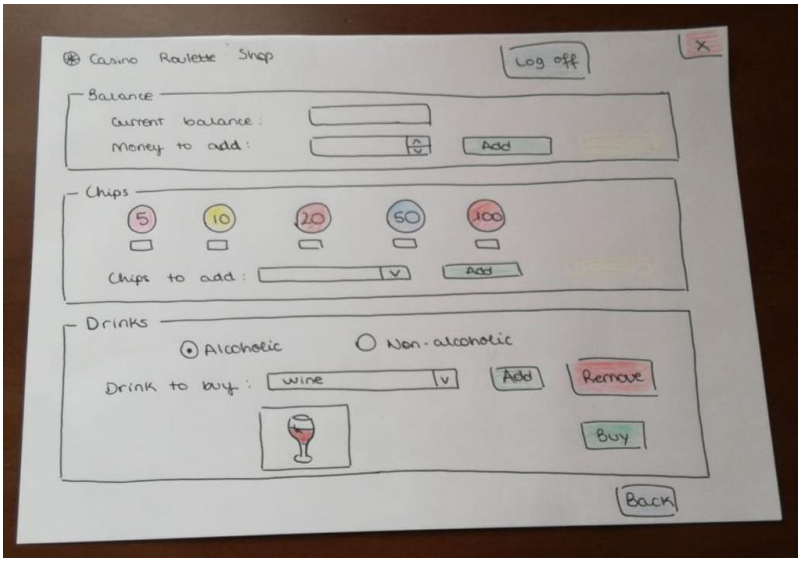


❖ Shop Window

Just as its name, this window functions as the main shop of the whole application and its packed with functionalities. Every single operation regarding adding money, buying or exchanging chips and ordering drinks is stored here.

The changes between the first seminar and last were the addition of a log off button, and the option to add and remove drinks from the order.

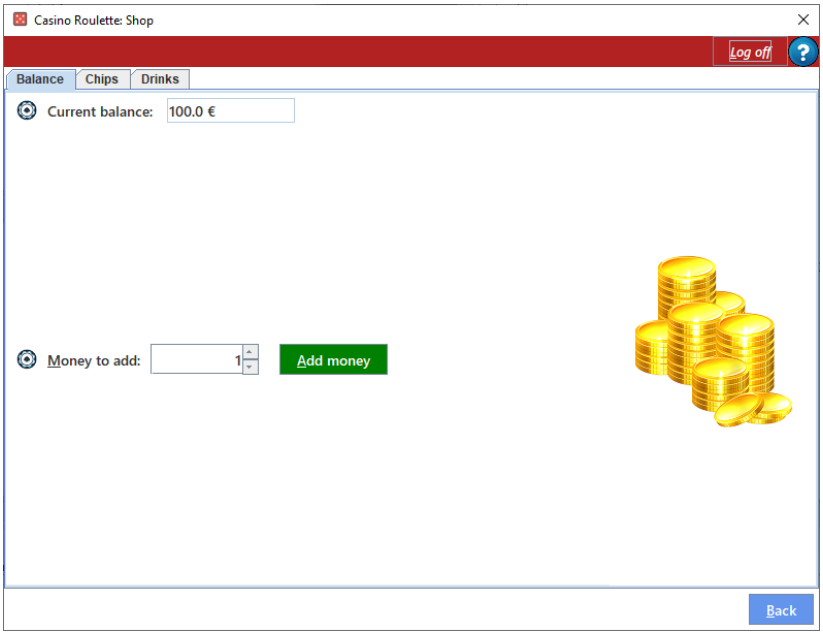




The final version kept the way we differentiated between balance, chips and drinks, expanding the functionalities of the last version while also changing the distribution of the window by using a card layout. The pseudo-header used in the welcome window is also used here, as well as used in the game window, keeping it consistent.

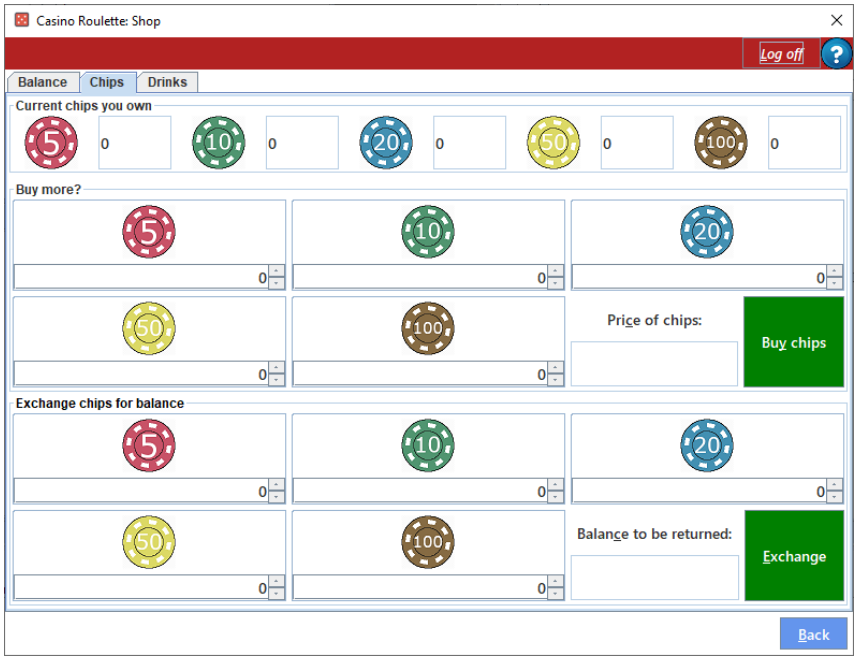
A help button was also added, and just to be preventive, if the user closes this window it will function the same as clicking the back button, with its respective logic that includes updating the information displayed in the parent's window.

The balance tab just received an image to not make it so empty.



The chip tab received more changes, to add a functionality we overlooked in the seminars, which is exchanging the chips you own for money. By using a panel in which we have the counter of every chip type the user owns, we can also add two panels which purpose is just buying or exchanging chips. Spinners were used instead of combo boxes just to make the application easier and cleaner to use.

Proper handling of rare cases was also included, displaying a dialog telling the user he doesn't have the cash to buy the chips, and updating the spinner for the exchanging chips by making the maximum of the spinner the number of chips of that type the user has.



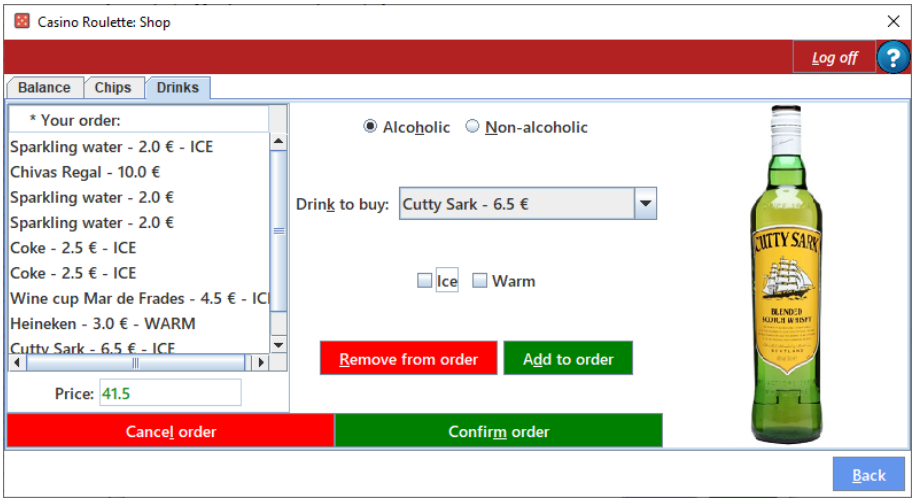
The drinks tab also suffered from a change of look. The distribution was changed to show the drink in the right part of the window, and a new scroll pane with a list of the drinks you have ordered was included to make the order visible for the user.

The checkboxes for ice and warm were added, and they are displayed as well in the order.

A text for the price of the order was added at the bottom of the list, and it changes its colour depending on if the user has enough cash to buy such order or not.

The previous button buy was changed by confirm order to make clearer its functionality, and a button called cancel order was added, which cancels the order completely.

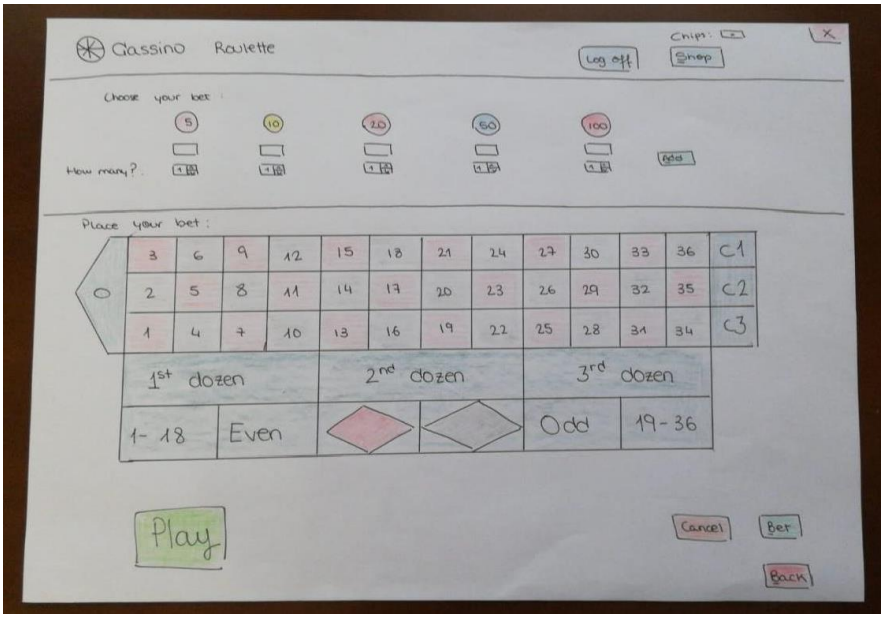
Handling different cases such as not having enough cash to buy the order were made, and such info is displayed in dialogs. A dialog if the user confirms the order with enough money is also displayed.



❖ Game window

This window just functions as the game itself, giving access to the user to bet different chips in different cells of the game, as well as giving a second access to the shop in case he runs out of chips, wants to order a drink and all the functions the shop offers.

The differences between the first seminar were the addition of the log off button and the play button which would start the game.



The last version of the game window received a complete change in design. pseudo-header of before is also applied in the game window but including the shop as an option up there as well. Card layout was applied to this window, making it less packed with different things. The tabs start appearing when the user makes different actions, and different restrictions appear when playing the game to avoid that the user leaves the game without redeeming the chips they could have earned. A total of three tabs are created, containing the board and game itself, the bets the user has made, and the results of the game, respectively.

If the user just closes the window or logs off when he has bet some chips but before playing, they will keep the chips when they return to the game/log in again. The back button functions exactly the same.

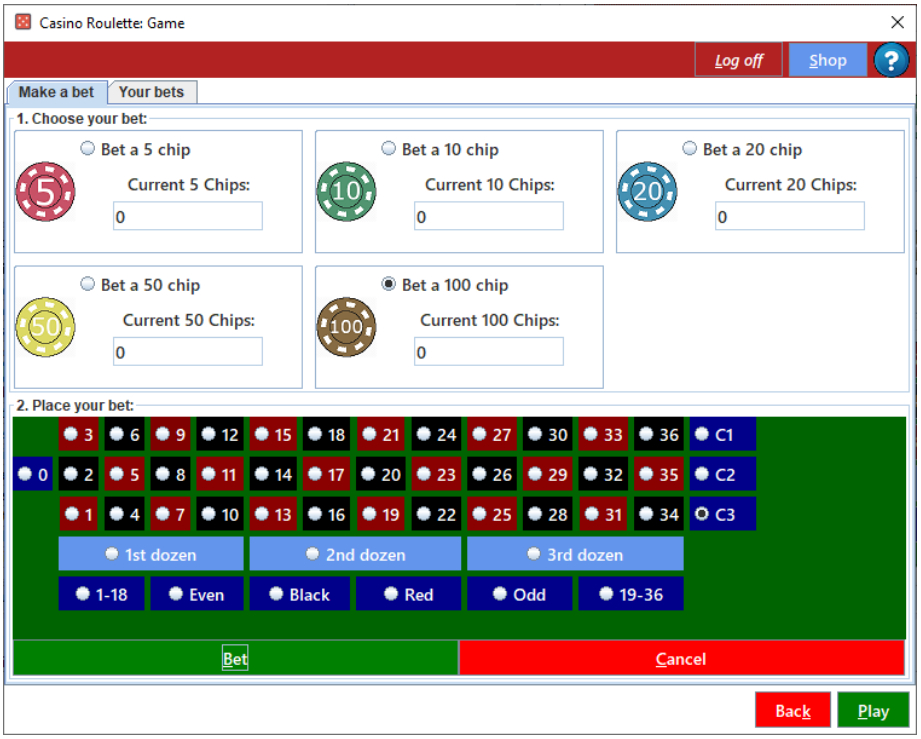
The play button will be enabled when the user has placed at least a bet, and it will compute the results of the game, which will be seen when explaining the third tab.

In the first tab, the "Make a bet" tab (which is a shifted version of the game window of the previous version), the user is shown which chips they own and they are faced with the option of choosing a chip and a cell to bet.

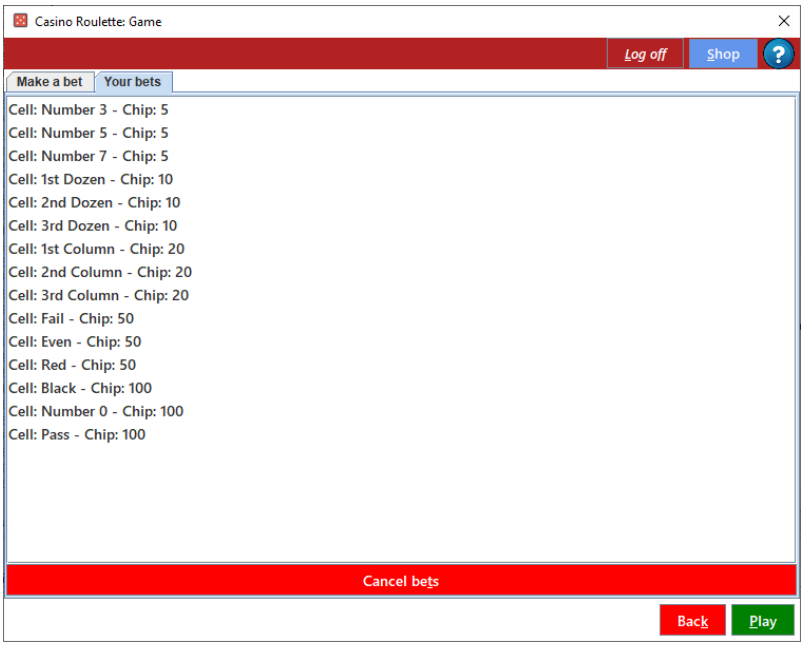
The previous spinners were changed to a radio button, since only a chip of one type can be bet in a cell. Choosing a cell in the board is also made with radio buttons, to give the user a clearer sense that they can only bet a type of a chip in only a cell, as well as making the user see which options are they picking before clicking on "Bet".

The bet button will create a bet which will appear in the tab we will be looking next. However, if such cell was already used in a bet or the user is betting a chip that they don't have, a dialog will be displayed explaining the error. Furthermore, if the user hasn't picked a chip or a cell, another dialog will be displayed, telling them to pick before betting.

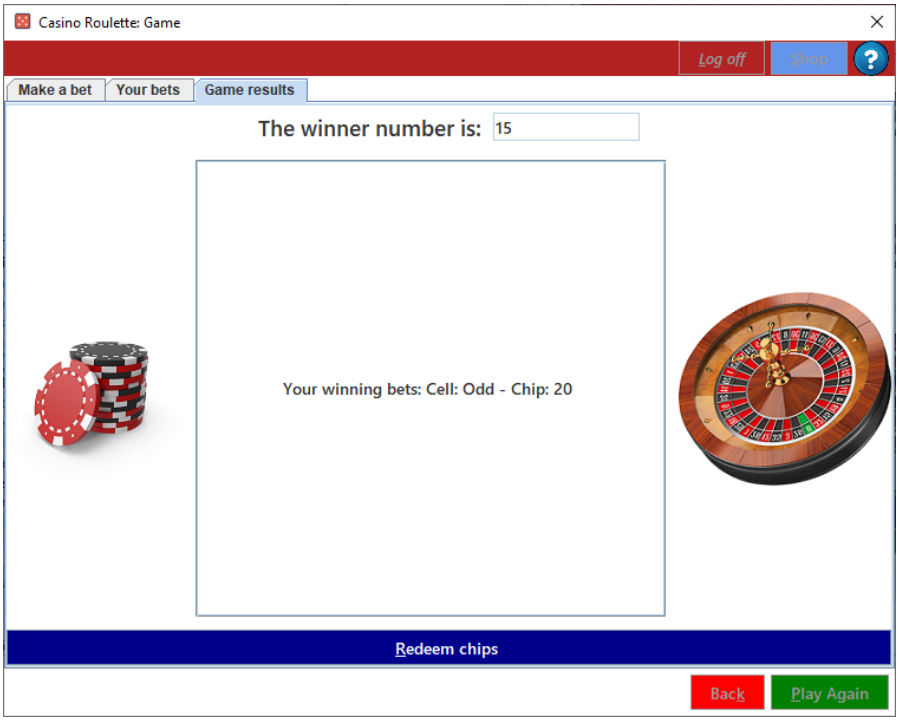
The cancel button will just reset the radio buttons.



After making a proper bet, a new tab will be created, named "Your bets". Here, the user can see which cells and which chips they have used to bet in the game displayed on a list that includes a scroll pane. This tab also gives the option to cancel bets, which would return the chips the user has bet, close this tab and restart the game.

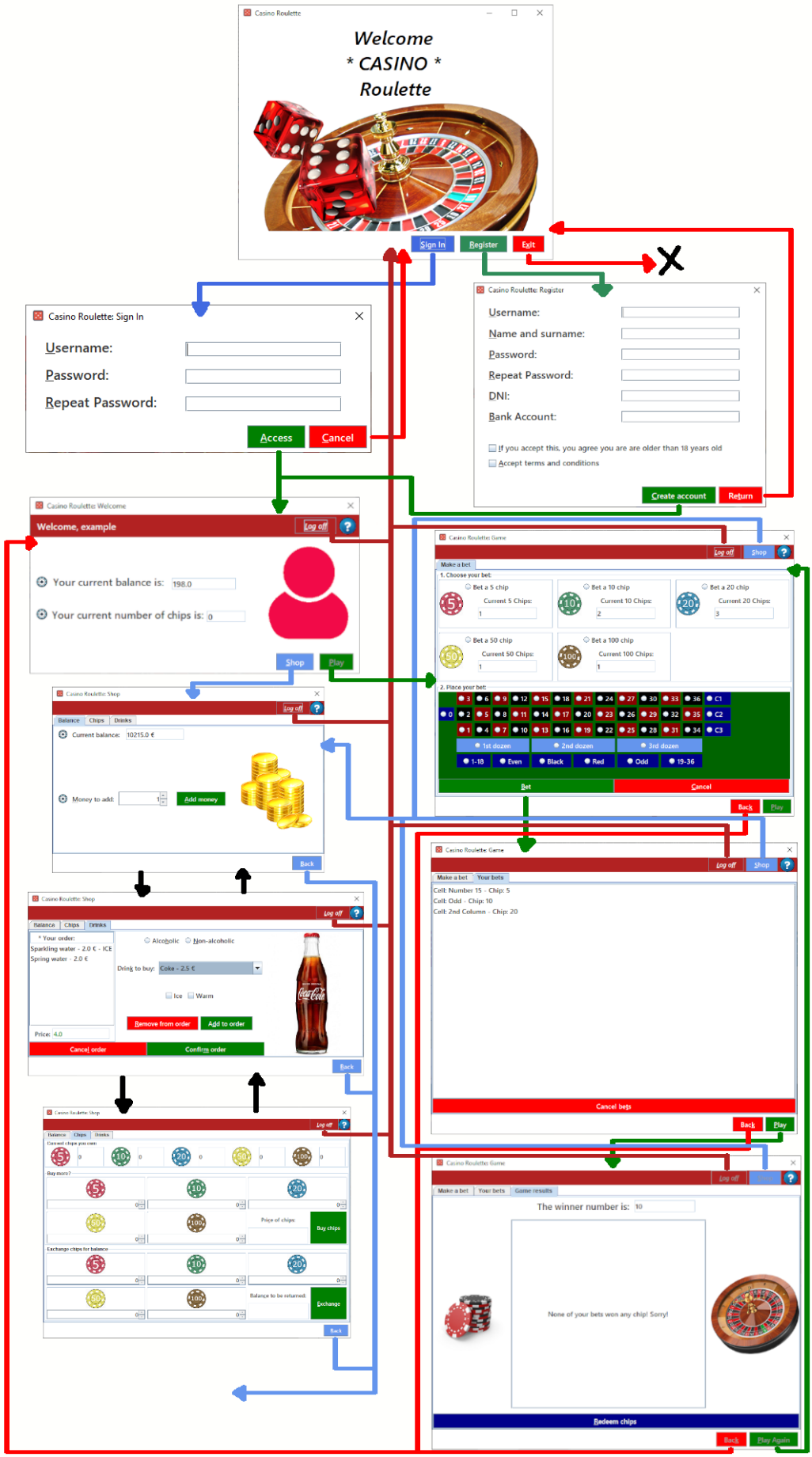


The third tab is generated when the user clicks on the play button. It's automatically focused, and it displays the result immediately. They user here can see which bets won them chips, as well as go back to the previous tabs to check which bets they have made in this game. As soon as this tab is generated, every button in this window is disabled, and the only option the user faces is redeeming the chips, which would give them the appropriate prize. As soon as the redeem chips button is pressed, the user can either go back, log off, visit the shop, or play again. Play again just initializes the game again.



- ***General view of the application and relationship between windows***

The relationship of the windows are as follows. I didn't include additional dialogs generated when exceptional cases nor the functionalities of the buttons which are more related to the logic than to the gui.


Casino Roulette
Welcome
* CASINO *
Roulette
Sign In
Register
Exit
Casino Roulette: Sign In
Username:
Password:
Repeat Password:
Access
Cancel
Casino Roulette: Register
Username:
Name and surname:
Password:
Repeat Password:
DNI:
Bank Account:
If you accept this, you agree you are older than 18 years old
Accept terms and conditions
Create account
Return
Casino Roulette: Welcome
Welcome, example
Log off
Your current balance is: 198.0
Your current number of chips is: 0
Shop
Play
Casino Roulette: Game
Make a bet
1. Choose your bet:
Bet a 5 chip
Current 5 Chips:
Bet a 10 chip
Current 10 Chips:
Bet a 20 chip
Current 20 Chips:
Bet a 50 chip
Current 50 Chips:
Bet a 100 chip
Current 100 Chips:
2. Place your bet:
1st dozen
2nd dozen
3rd dozen
1-18
Even
Black
Red
Odd
19-36
Bet
Cancel
Back
Play
Casino Roulette: Shop
Balance
Chips
Drinks
Current balance: 10215.0 €
Money to add:
Add money
Back
Your bets
Cell: Number 15 - Chip: 5
Cell: Odd - Chip: 10
Cell: 2nd Column - Chip: 20
Cancel bets
Your order:
Sparkling water - 2.0 € - ICE
Spring water - 2.0 €
Alcoholic
Non-alcoholic
Drink to buy:
Coke - 2.5 €
Ice
Warm
Remove from order
Add to order
Price: 4.0
Cancel order
Confirm order
Buy more?
Price of chips:
Buy chips
Exchange chips for balance
Balance to be returned:
Exchange
Game results
The winner number is: 10
None of your bets won any chip! Sorry!
Redeem chips
Play Again

▪ *Logic classes*

In order to provide the appropriate relation between the graphic user interface and the logic, several classes were made. Such classes were:

❖ UserManagement

This class is the one in charge of registering, signing in and logging off users by using the FileUtil class that will be explained later.

- It contains the attribute players, which contains all the players that have accessed the application, as well as an attribute currentPlayer which stores the player that has accessed the application.
- The most remarkable methods this class contains are the register, the signIn and the logOff methods which do exactly what their names suggest they do.

❖ FileUtil

This class is an abstract class used to read and write different files that are used in the application. It is used by the UserManagement and the Shop to manage their files.

- It contains methods such as loadUsersFile, which loads the users list, saveToUsersFile, which saves the users file the updated player list, saveNewPlayer, which saves a new player to the list and loadDrinksFile, which loads the drinks file.

❖ Player

It represents the player, and overall just contains data about it.

- It contains the attributes asked in the register window, and an a list of chips that represents the chips the player currently holds.

❖ Chip

This class just represents a chip and is general used by other classes in order to compute the prize given or the value in the shop.

- Its only attribute is the value of the chip, which in this casino is set to be either 5, 10, 20, 50 or 100.

❖ Drink

Another class that is used more to store data than to executing complex methods. It represents one of the drinks the user can request in the shop.

- The attributes of this class are the code, used to read the drink file and proper load the images, the name of the drink, whether if its alcoholic or not, the price, and whether if its warm or not.

❖ Order

It represents the list of drinks the user is ordering in the shop. It just contains an array list of drinks and only has method that involve simple changes to such list.

❖ Bet

It represents the bet that the user makes when he selected the chip and cell he wants to play with. As well as the previous classes, it just contains information.

- The attributes of the bet class are the chip that was bet, and as well as the cell where it is bet. A boolean isWinner is also an attribute, in order to differentiate the bets that resulted in a win or in a loss of chips when the game is played.

❖ Shop

This class is one of the heaviest regarding methods, since it represents the shop of the application. It's only invoked in the shop where it belongs, it being the shop window.

- It contains the list of drinks that are available in the shop, as well as the order the user is making in case they are doing it.
- The methods of this class usually involve the player, since it's the one who is spending the money in the shop. Some of the methods are addMoney, which adds money to the current player, calcTotalPriceChips, which calculates the total value of the chips the player have, addChips, which adds chips to a player, returnChips, which returns the chips of a

player in favour of money, loadDrinks and loadDrinksByType, which load the drinks in the combo box and filter them and different basic operations involving the modification of the order of drinks.

❖ Game

Game is responsible of all the logic present in the game window, and as such, it has a lot of methods that revolve around creating bets, returning or collecting chips from the player, picking the winner cell, and compute the winner bets of the player

- This class contains a board, which is used to compute things such as the winner cell and adding bets to a specific cell, the bets the player is making, and a list of chips which is the prize the user will obtain with the bets they have made
- Some of the methods that it includes are getWinnerCell which picks a winner cell, addBet, which adds a bet to the game, addBetToPrize, which adds the winning chips of such winning bet to the list of prizes, resetBets, which resets all the bets and returns the bet chips to the player, redeemChips, which gives the player the prize they have earned and computeEndOfGame, which involves searching in the bets which bets are going to make the user earn chips and which not, as well as computing the chips they will make them win, as well as taking into consideration the special case where 0 is the winner number.

❖ Board

This class represents the board of the casino, and as such, it contains the different cells where the ball can drop and so the different cells where the user can bet.

- It contains the list of cells of the board.
- The main method of the board is loading the board by making the different cells when the board is created (loadBoard)

❖ Cell

This abstract class is made to extend the different types of cells the board can store. The different attributes that are used in the different extended classes are used to compute whether the bet that is on such cell is a winner cell or not.

- It contains the attribute type (made to later associate the cells with the gui) and the name of the cell (the one displayed in the rest of places)
- This cell contains a very important abstract method named getChipsToWin, which will return a list of chips depending on the cell.

❖ ColorCell

This class just extends the cell, adding the attribute isRed to check whether the cell is red or black. The getChipsToWin returns the chip you used to bet in this cell, as well as another one, following the rules of the game.

❖ ColumnCell

This class just extends the cell, adding the attribute column that indicates in which column the cell is. The getChipsToWin returns the chip you used to bet in this cell, as well as another one, following the rules of the game.

❖ DozenCell

This class just extends the cell, adding the attribute dozen that indicates in which dozen the cell is. The getChipsToWin returns the chip you used to bet in this cell, as well as another two, following the rules of the game.

❖ FailPassCell

This class just extends the cell, adding the attribute isFail that indicates whether the cell is between different intervals. The getChipsToWin returns the chip you used to bet in this cell, as well as another one, following the rules of the game.

❖ OddEvenCell

This class just extends the cell, adding the attribute isOdd that indicates whether the cell is odd or even. The getChipsToWin returns the chip you used to bet in this cell, as well as another one, following the rules of the game.

❖ NumberCell

This class extends the cell, adding the attribute number and all the attributes of the previous cells. Whenever it is created it invokes the computeNumber

function, which assigns to the rest of the attributes the different value according to the number. The getChipsToWin returns the chip you used to bet in this cell, as well as another thirty-five chips, following the rules of the game.

- ❖ Exceptions

Some exceptions were made to handle exceptional cases. They were mostly used in the logic regarding the user management. They are the Already Registered Exception, Empty Fields Exception, Passwords Don't Match Exception and User Not Found Exception.