System Maintenance

System Overview

Graphical User Interface

This part of the system provides the user with an interactable display, increasing the user friendliness of the system. It also gives the system a more commercial feel and look. It contains navigational methods via push buttons, making the system simplistic. Input components such as line edits and drop down boxes are included within the interface allowing the user to commit and amend data in the database through the system. The interface displays appropriate labels to allow the user to understand the data that is being displayed.

Home Screen

This area of the system provides the user with a main menu for which they can use to navigate around the program. Each part of the program is selectable via a clearly labelled push button. The user will go through this screen every time they wish to switch areas on the system.

Squad List

This area of the system provides the user with a list of all the players they have previously entered. Each player is selectable if the user wishes to view/amend their details. It also gives the user the option to add a new player or return to the home screen.

Goal List

This area of the system provides the user with a table of all the data on goals that they have previously entered. The data is displayed under clear headers so that it is easy to read and interpret. The user is also given the option to add a new goal or return to the home screen.

Match List

This area of the system provides the user with a table of all the data on matches that they have previously entered. The data is displayed under clear headers so that it is easy to read and interpret. The user is also given the option to add a new match or return to the home screen.

Player Info

This area of the system provides the user with the all the details on a previously selected player. The data is displayed in line edits so that it can be edited. Each line edit is labelled so that the user knows what data is required in each line edit. At the bottom of the screen the user is given the option to save their changes, delete the selected player from the system or cancel (return to the previous screen).

Add Player

This area of the system provides the user with a blank detail screen, the user is prompted to fill in all the line edits with the details required to add a new player to the system. Each one of these line edits will be validated. At the bottom of the screen the user has the option to add the new player to the system and the option to cancel (return to the previous screen).

Add Goal

This area of the system provides the user with two drop down boxes and a line edit, the user is prompted to select an option from the drop down boxes and fill in the line edit with the details required to add a new goal to the system. The line edit will be validated. At the bottom of the screen the user has the option to add the new goal to the system and the option to cancel (return to the previous screen).

Add Match

This area of the system provides the user with a blank detail screen, the user is prompted to fill in all the line edits with the details required to add a new match to the system. Each one of these line edits will be validated. At the bottom of the screen the user has the option to add the new match to the system and the option to cancel (return to the previous screen).

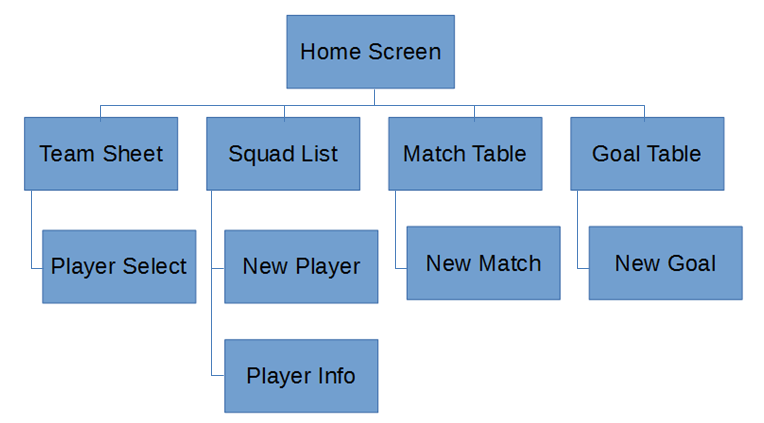
Team Sheet

This area of the system provides the user with a formation to which they can add players to lay out a team for any upcoming matches. Each position is a push button that is clearly labelled, so that it is easy for the user to add a player to the team sheet in a certain position. When a player is entered the position remains a push button so that the player can be swapped. At the bottom of the screen the user has the option to clear the team sheet of all players or return to the home screen.

Player Select

This area of the system provides the user with a list of selectable players (name only), same layout as the squad list, however there is only options to add the selected entry to the team sheet or cancel and return to the team sheet.

Navigation Diagram



Code:

Taken from the Add Goal Class

1. **def** PopulateMatchComboBox(self):
2. *#Retrieving all the Match details from the database*
3. matches = g\_database.GetAllMatches()
4. **for** match **in** matches:
5. *#Adds the match Opposition to the combo box*
6. self.matchCombo.addItem(match[2])

This snippet of code adds the match opposition into a combo box. Using a combo box means that the user can’t incorrectly input data as there is only valid options to choose from. I made this section of code a separate function so that it was separate from the layout code and it could be re-run if required.

Taken from the Add Player Class

1. alphabet\_lower = []
2. alphabet\_upper =[]
3. #Adding each lower case letter to the lower list
4. **for** letter **in** map(chr, range(97, 123)):
5. alphabet\_lower.append(letter)
6. #Adding each upper case letter to the upper list
7. **for** letter **in** map(chr, range(65,91)):
8. alphabet\_upper.append(letter)

This snippet of codes adds each lower and upper case character of the alphabet to empty lists. These lists can then be used to validate user entries. This code is under the btnAdd\_pushed function because it only needs to be executed once the user has decided they want to enter data to the database.

Taken from the Match List code

1. refreshTable(self):
2. *#Gets all matches from the database*
3. Matches = g\_database.GetAllMatches()
4. *#Sets the row count to the number of entries*
5. self.table.setRowCount(len(Matches))
6. *#Sets the column count to 4*
7. self.table.setColumnCount(4)
8. *#Labels each of the column headers*
9. self.table.setHorizontalHeaderLabels(["Id","Date","Oppositon","Result"])
10. row = -1
11. *#Adds each item in the database to table under the correct header row by row*
12. **for** Match **in** Matches:
13. column = 0
14. row = row+1
15. **for** field **in** Match:
16. *#Actually adding the item to the table*
17. self.table.setItem(row, column, QTableWidgetItem(str(field)))
18. column = column +1

This snippet of code takes the data from the match table in the database and puts it in a table widget. I have chosen to use a table widget as it easily displays the data in a readable format. In line 5 I have used Len to find the length of Matches. This enables me to then set the row count to the required number meaning that all the data is displayed without any empty rows also being displayed. This increases the aesthetics of the table. Line 9 sets the horizontal headers to the table; this means that each column is clearly labelled making the table easy to read. All this code is under the refreshTable function so that it can be executed each time a new entry is added to the database without having to execute the entire file.

Taken from the Player Select code:

1. *#Getting all players from the database*
2. players = g\_database.GetAllPlayers()
3. row = -1
4. *#Clearing the list*
5. self.list.clear()
6. **for** player **in** players:
7. *#Checking the player is available*
8. **if** player[6] **not** **in** ["n","N","no","No","NO"]:
9. row = row+1
10. *#Blank list*
11. name\_list = ""
12. *#Adding the initial of the Forename to the list*
13. name\_list = name\_list + (player[1][0]) + " "
14. *#Adding the surname to the list*
15. name\_list = name\_list + (player[2])
16. *#Adding the players inital and surname to the main list*
17. self.list.addItem(name\_list)

This snippet of code populates the list widget. A validation check is executed in line 8 to check that the player is available to play by checking that the availability isn’t equal to an equivalent of no. Line 13 adds the initial of the players’ forename to the recently created blank list. The surname is then added in line 15. I have chosen to do this so that the list the user sees contains the players initial and surname. This will limit the amount of confusion if two players with the same surname are entered into the system.

1. players = g\_database.GetAllPlayers()
2. **for** l\_index, player **in** enumerate(players):
3. *#Checking to see which player in the database is the selected player*
4. **if** player[1][0]+" "+player[2] == player\_data:
5. *#changing index to the database index for the selected player in the*
6. index = l\_index

This snippet of code converts the selected players’ list index to the selected players’ database index. This needs to be done as the two indexes may not be the same due to some players not being displayed in the list.

Variable Listing

Home screen

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.btnTeamSheet | Provides a button for the user to press if they want to go the Team Sheet |
| self.btnSquad | Provides a button for the user to press if they want to go the Squad List |
| self.btnMatch | Provides a button for the user to press if they want to go the Match Table |
| self.btnGoals | Provides a button for the user to press if they want to go the Goal Table |
| self.btnQuit | Provides a button for the user to press if they want to exit the program |
| self.layout | Provides the layout for all the buttons to be displayed on |
| self.team |  |
| Self.squad |  |
| self.match |  |
| self.goals |  |

Squad List

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.btnInfo | Provides a button for the user to press if they want to view the Player Info screen |
| self.Home | Provides a button for the user to press if they want to go back to the Home screen |
| self.New | Provides a button for the user to press if they want to add a new Player |
| self.squad\_list | Used to display a list of all the players in the database |
| self.VlayoutMain | Provides the main layout for all the screen to be displayed on |
| self.hlayout1 | Provides the secondary layout for all the screen to be displayed on |
| self.widget |  |
| self.new\_player |  |
| self.playerInfo |  |
| Players | Used as a search term |
| Row | Used to sort through the list |
| name\_list | Used to temporarily store the Players forename initial and surname |

Goal List

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.btnNew | Provides a button for the user to press if they want to add a new goal |
| self.btnHome | Provides a button for the user to press if they want to go back to the Home screen |
| self.table | Used to display a table of all the goals in the database |
| self.VlayoutMAIN | Provides the main layout for all the screen to be displayed on |
| self.Vlayout | Provides a secondary layout for all the screen to be displayed on |
| self.Hlayout | Provides a secondary layout for all the screen to be displayed on |
| self.new\_goal | Holds the class for the new window |
| Goals | Holds all the goals from the database to be sorted into the table |
| Row | Helps sort through each position the table |
| Column | Helps sort through each position the table |

Match List

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.btnNew | Provides a button for the user to press if they want to add a new match |
| self.btnHome | Provides a button for the user to press if they want to go back to the Home screen |
| self.table | Used to display a table of all the goals in the database |
| self.VlayoutMAIN | Provides the main layout for all the screen to be displayed on |
| self.Vlayout | Provides a secondary layout for all the screen to be displayed on |
| self.Hlayout | Provides a secondary layout for all the screen to be displayed on |
| Matches | Holds all the matches from the database to be sorted into the table |
| Row | Helps sort through each position the table |
| Column | Helps sort through each position the table |

Add Player/Player Info(Same variables)

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.forename | Provides a line edit for the user to enter data |
| self.surname | Provides a line edit for the user to enter data |
| self.rating | Provides a line edit for the user to enter data |
| self.email | Provides a line edit for the user to enter data |
| self.position | Provides a line edit for the user to enter data |
| self.avaliable | Provides a line edit for the user to enter data |
| self.btnAdd | Provides a button for the user to press if they want to add the data to the database |
| self.btnCancel | Provides a button for the user to press if they want to cancel |
| self.labelF | Provides a label so that the user knows what data to enter |
| self.labelS | Provides a label so that the user knows what data to enter |
| self.labelR | Provides a label so that the user knows what data to enter |
| self.labelE | Provides a label so that the user knows what data to enter |
| self.labelP | Provides a label so that the user knows what data to enter |
| self.labelA | Provides a label so that the user knows what data to enter |
| self.hlayout1 | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout1 | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout2 | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout3 | Provides the main layout for all the screen to be displayed on |
| self.hlayout2 | Provides a secondary layout for all the screen to be displayed on |
| Forename | Holds the text version of the data inputted by the user |
| Surname | Holds the text version of the data inputted by the user |
| Rating | Holds the text version of the data inputted by the user |
| Email | Holds the text version of the data inputted by the user |
| Position | Holds the text version of the data inputted by the user |
| Available | Holds the text version of the data inputted by the user |
| email\_valid | Holds the result of the validation check |
| rating\_valid | Holds the result of the validation check |
| position\_valid | Holds the result of the validation check |
| avaliable\_valid | Holds the result of the validation check |
| forename\_valid | Holds the result of the validation check |
| surname\_valid | Holds the result of the validation check |
| alphabet\_lower | List of all the lower case letters in alphabet |
| alphabet\_upper | List of all the upper case letters in alphabet |
| Count | Sorts through each position in the list |
| self.error | Calls the error message function, passing in the message |
| at\_valid | Holds the result of the validation check |
| dot\_valid | Holds the result of the validation check |

Add Goal

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.labelM | Provides a label so that the user knows what data to enter |
| self.matchComb | Provides a combo box to display the possible options |
| self.labelP | Provides a label so that the user knows what data to enter |
| self.playerCombo | Provides a combo box to display the possible options |
| self.quantity | Provides a line edit for the user to enter data |
| self.labelQ | Provides a label so that the user knows what data to enter |
| self.btnAdd | Provides a button for the user to press if they want to add the new goal |
| self.btnCancel | Provides a button for the user to press if they want to cancel |
| self.hlayout1 | Provides a secondary layout for all the screen to be displayed on |
| self.hlayout2 | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout1 | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout2 | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout3 | Provides the main layout for all the screen to be displayed on |
| Players | Holds all the players in the database |
| Matches | Holds all the matches in the database |
| indexP | Holds the selected player index |
| Name | Holds the surname of the selected player |
| indexM | Holds the selected match index |
| Match | Holds the opposition of the selected match |
| Quantity | Holds the text version of the entered integer |
| self.error | Calls the error message function, passing in the message |

Add Match

|  |  |  |
| --- | --- | --- |
| Variable Name | | Purpose |
| self.parent | | Used to reassign the passed in term |
| self.date | | Provides a line edit for the user to enter data |
| self.labelD | | Provides a label so that the user knows what data to enter |
| self.opposition | | Provides a line edit for the user to enter data |
| self.labelO | | Provides a label so that the user knows what data to enter |
| self.result | | Provides a line edit for the user to enter data |
| self.labelR | | Provides a label so that the user knows what data to enter |
| self.btnAdd | | Provides a button for the user to press if they want to add the new match |
| self.btnCancel | | Provides a button for the user to press if they want to cancel |
| self.hlayout1 | | Provides a secondary layout for all the screen to be displayed on |
| self.hlayout2 | | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout1 | | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout2 | | Provides a secondary layout for all the screen to be displayed on |
| self.vlayout3 | | Provides the main layout for all the screen to be displayed on |
| date | | Holds the text version of the data inputted into the date box |
| result | Holds the text version of the data inputted into the result box |
| opposition | Holds the text version of the data inputted into the opposition box |
| self.error | Provides a label so that the user knows what data to enter |

Player Select

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.teamindex | Used to reassign the passed in term |
| self.list | Provides a list that can be displayed to the user |
| self.btnAdd | Provides a button for the user to press if they want to add the new match |
| self.btnCancel | Provides a button for the user to press if they want to cancel |
| self.VlayoutMAIN | Provides the main layout for all the screen to be displayed on |
| self.hlayout1 | Provides a secondary layout for all the screen to be displayed on |
| players | Holds all the players in the database, used as the search criteria |
| row | Used to sort through the list |
| name\_list | Holds the selected players initial and surname |
| index | Holds the selected index from the list |
| player\_list | Holds a list of all the players already in the team sheet. |

Team Sheet

|  |  |
| --- | --- |
| Variable Name | Purpose |
| self.parent | Used to reassign the passed in term |
| self.lblSTR | Provides the user with a button for the position Right Striker, enabling them to enter a player |
| self.lblSTL | Provides the user with a button for the position Left Striker, enabling them to enter a player |
| self.lblRM | Provides the user with a button for the position Right Midfield, enabling them to enter a player |
| self.lblCMR | Provides the user with a button for the position Right Centre Midfield , enabling them to enter a player |
| self.lblCML | Provides the user with a button for the position Left centre Midfield , enabling them to enter a player |
| self.lblLM | Provides the user with a button for the position Left Midfield , enabling them to enter a player |
| self.lblRB | Provides the user with a button for the position Right Back, enabling them to enter a player |
| self.lblCBR | Provides the user with a button for the position Right Centre Back , enabling them to enter a player |
| self.lblCBL | Provides the user with a button for the position Left Centre Back , enabling them to enter a player |
| self.lblLB | Provides the user with a button for the position Left Back, enabling them to enter a player |
| self.lblGK | Provides the user with a button for the position Goal Keeper, enabling them to enter a player |
| self.btnHome | Provides the user with a button to return to the home page |
| self.btnClear | Provides the user with a button to clear the team sheet |
| player\_list | Holds a list of all the players in the database |
| self.PlayerAdd | Used to call the PlayerSelect function |