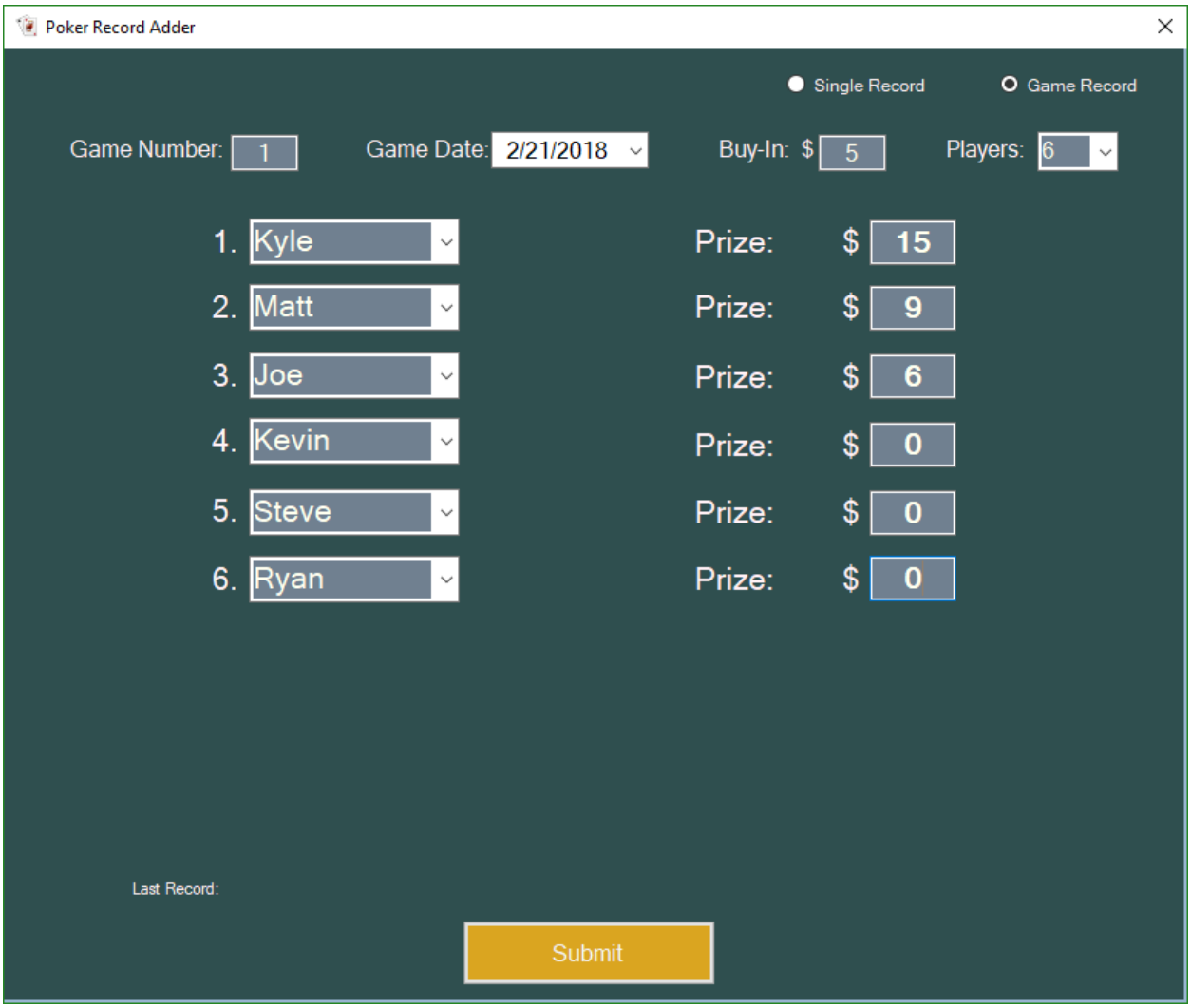
Poker Game Record Adder



Developed By:

Louis Notarino III

Overview

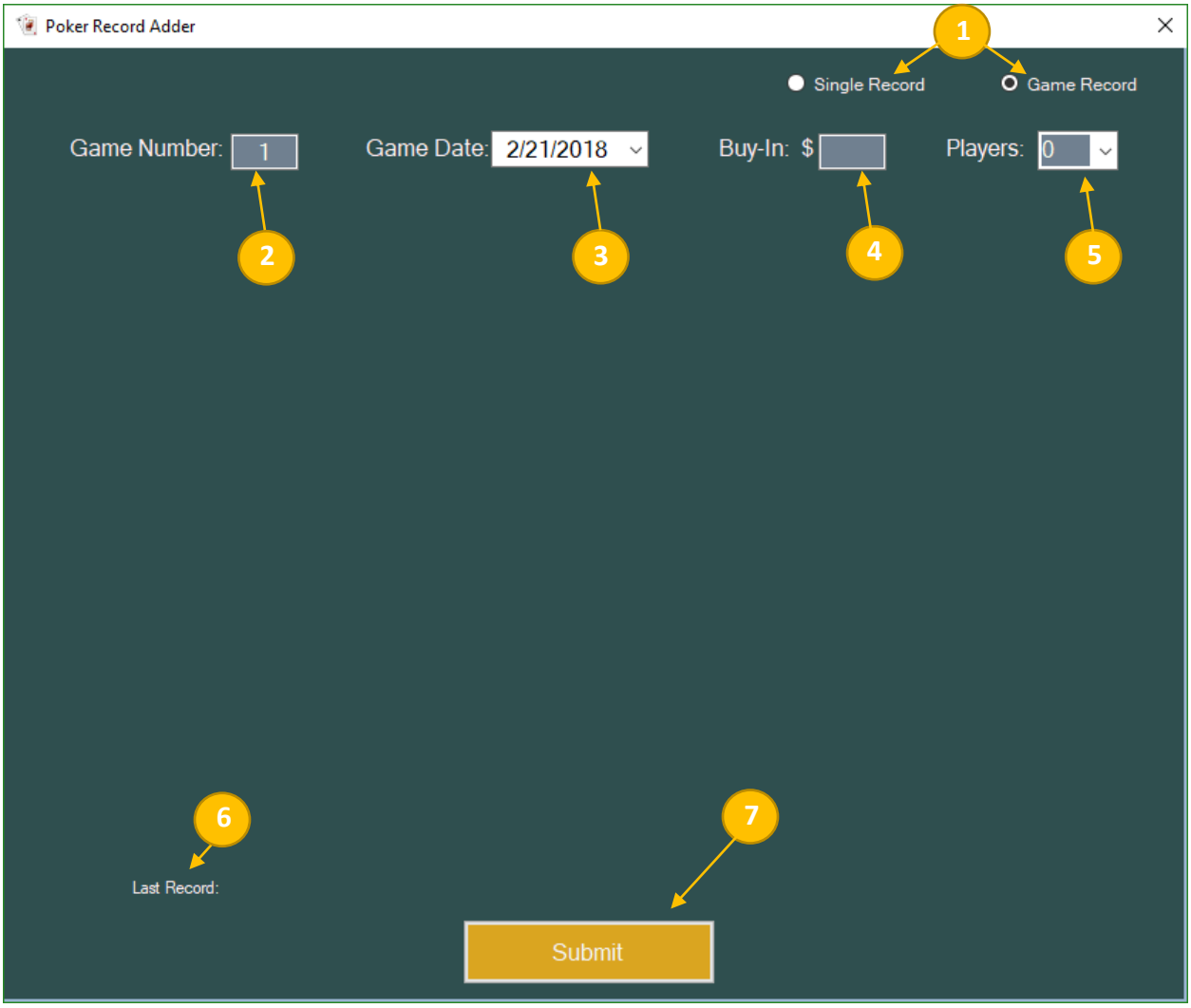
This is a Windows Forms Application Project developed using the .NET framework and C#. The project consists of a single interface that allows the user to keep track of friendly poker game records by inputting and storing their records to a SQL Server database.

The Database

	Column Name	Data Type	Allow Nulls
🔑	game	smallint	☐
	game_date	date	☐
🔑	player	varchar(15)	☐
	position	smallint	☐
	prize	smallmoney	☐
	buy_in	smallmoney	☐

The database that contains all the game records is a small database that consists of one table with six fields. The *game* field is an integer value which gives each game a unique ID that starts at 1 and continues in an ascending fashion. The *game_date* field is a date value that consists of the date a game was played. The *player* field is a character value that consists of the name of the player who participated in the game. The *position* field is an integer value that represents the position a player finished in a game. The *prize* field is a money value that consists of the amount of money (if any) a player won at their finished position in a game. The *buy_in* field is a money value that consists of the buy-in price a player paid to participate in the game. This field is also used to determine the *prize* field. Within this table there is a primary key consisting of two fields, the game and player field. By using these two fields it ensures each record is unique from the rest for proper database usage and storage. All the fields are required within the database, so the database does not consist of any null values.

The Interface



The following is a breakdown of the interface on startup. Each number will correspond to the numbered bubble in the image above to provide a description of each component.

1. The user has the option to input an entire game record at once, or a single game record. These radio buttons provide the ability to toggle back and forth between the two separate panels. On startup, the full game record panel is displayed which involves every other component displayed on the interface.

2. This text box is where the user will input what game number to input records for. On startup, if the database consists of zero previous records the game number will be defaulted at “1”. If the database does consist of records, the interface will check the database for the highest game

number and default the text box to the highest game number plus one. This text box corresponds to the *game* field in the database.

3. This date picker field allows the user to select the date for the game they will be adding records for. On startup, the date is defaulted to the current system date. This date picker corresponds to the *date* field in the database.

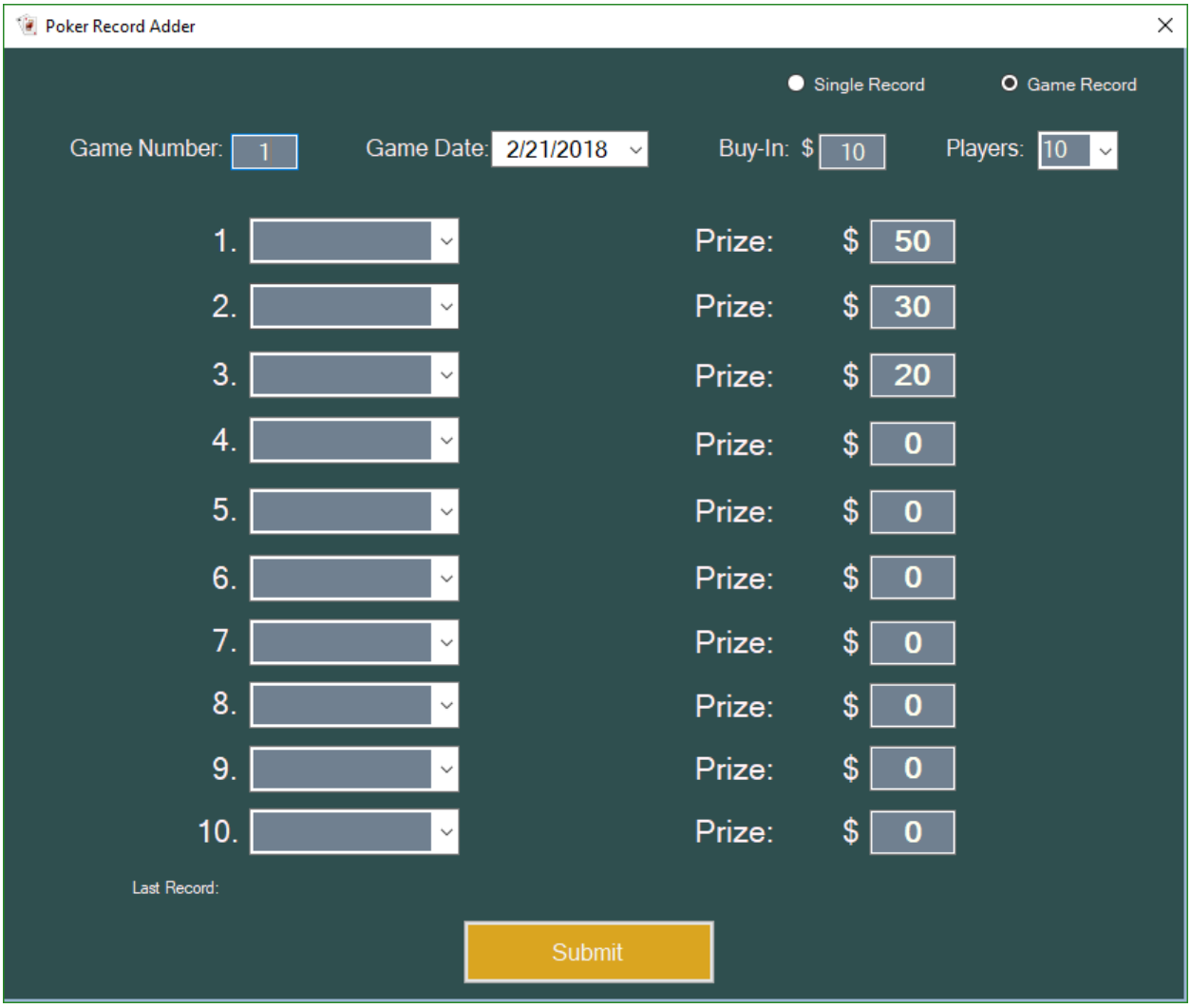
4. This text box is where the user will input the buy-in amount for the game they will be adding records for. This text box corresponds to the *buy_in* field in the database.

5. This combo box allows the user to select how many players were in the game that records will be inputted for. The interface holds up to ten players for a single game and requires a minimum of four players.

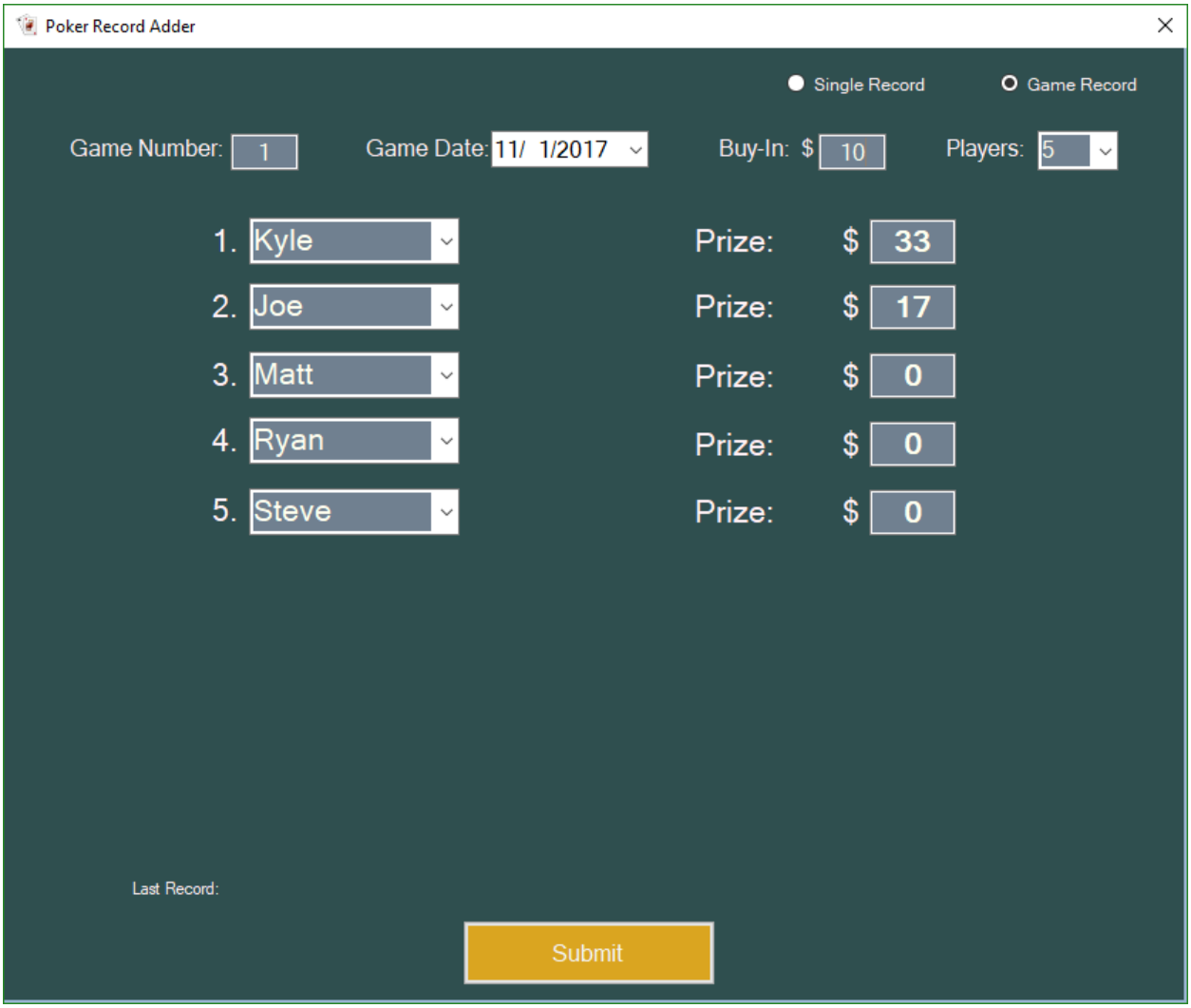
6. This label is where the previous successfully submitted records within the session are displayed.

7. This button is for the user to click when they are done inputting the records in the interface and wish to submit them to the database.

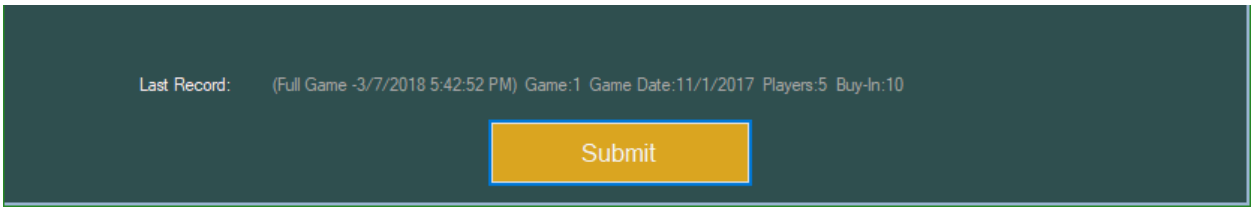
Adding a Full Game Record



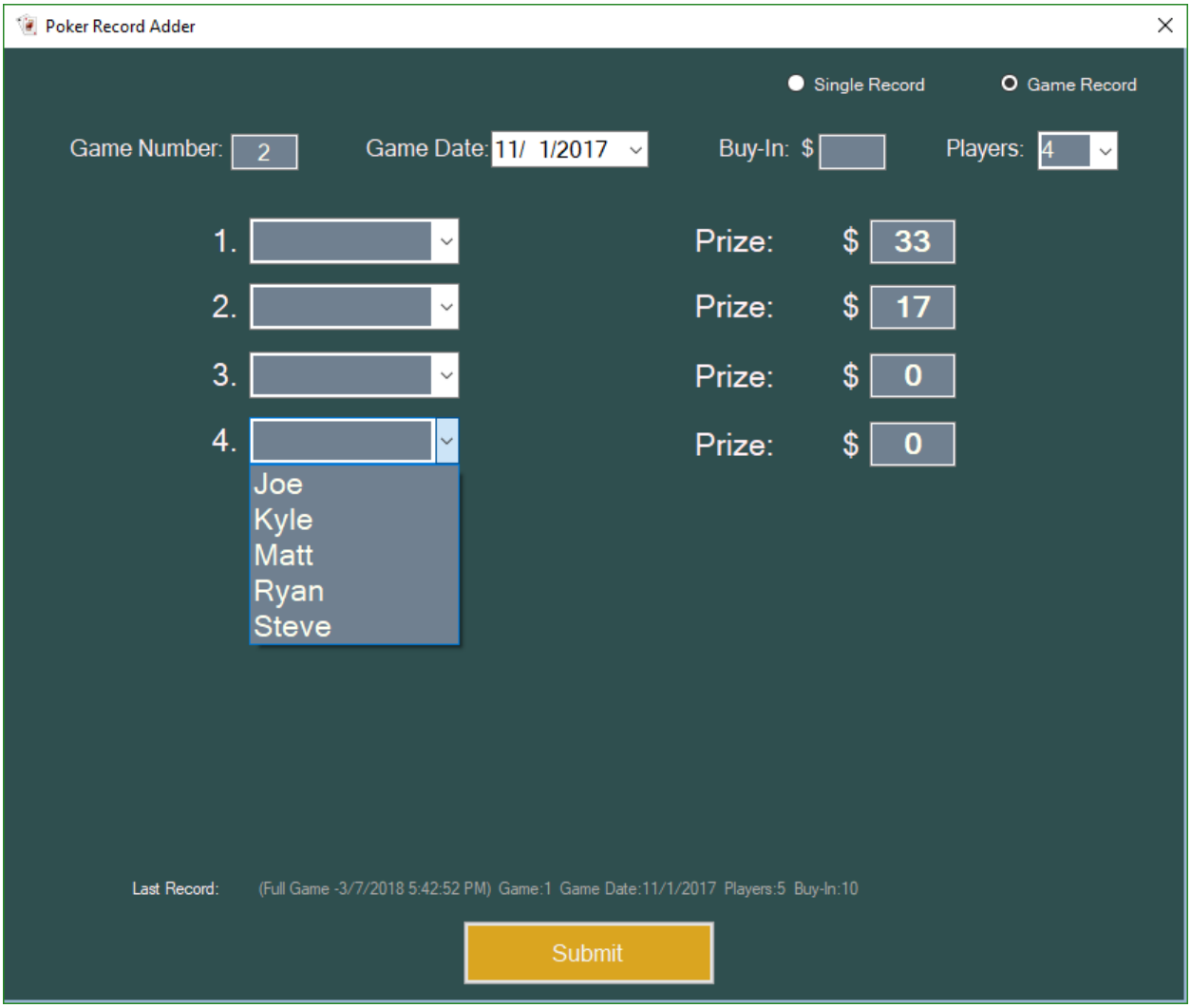
When the user changes the player combo box the proper number of rows are displayed for the user to input the game records. The prizes are auto-filled based on the buy-in field and number of the players. These are recommended prizes, but the user has the ability input their own prizes. The combo boxes corresponding to the position in a game are for inputting the player names and are dynamically displayed. If the database consists of no previous records the combo box will have zero items in it, requiring the user to manually input the names of the players. Once the database has records in it, the combo boxes consists of all the names within the database.



Above is the interface all filled with the proper input from the user and is ready to be submitted into the database using the submit button.

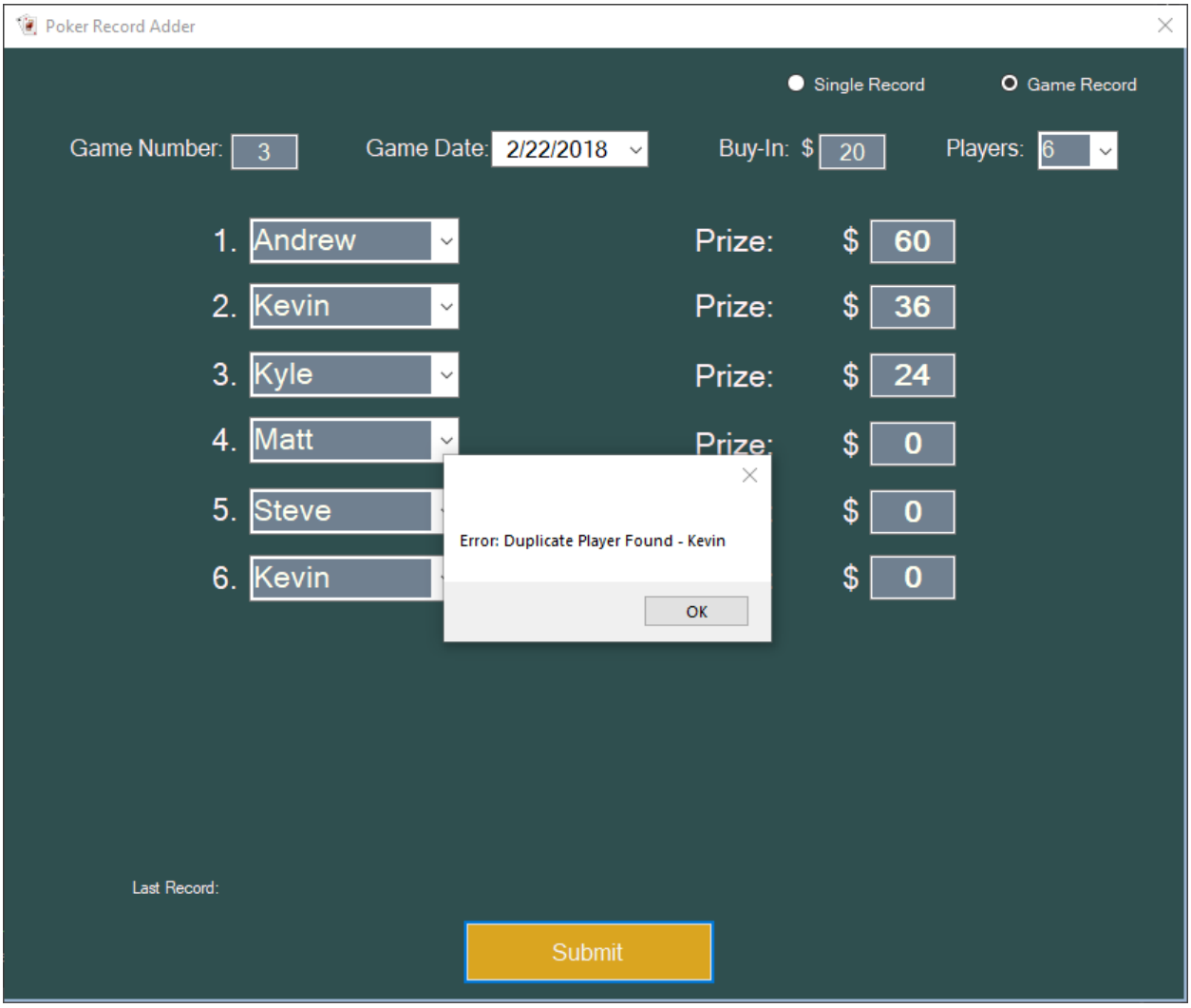


Above shows the last record label displaying the previously successful records submitted to the database within the session.

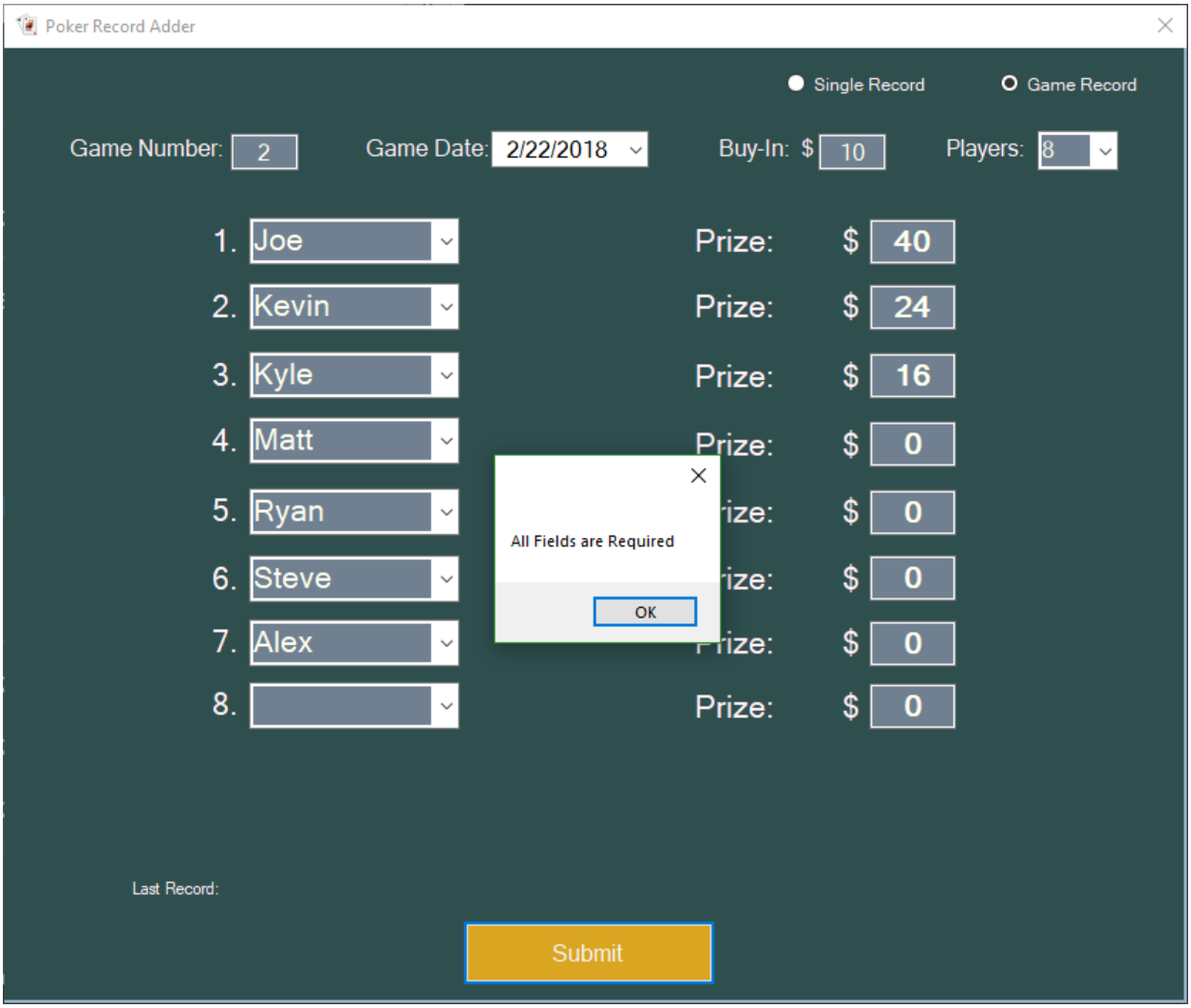


Above shows the combo boxes now consisting of all the players that were in the previously submitted game available to select from. The game number text box is also dynamically updated to display the next available game number. The user has the ability to change the game number to whatever they choose. If the user changes the game number to a number which already corresponds to a game within the database, the interface is prefilled with all the game details of the corresponding game. This allows the user to view the previously submitted games and the option to update an existing game.

Error Checking

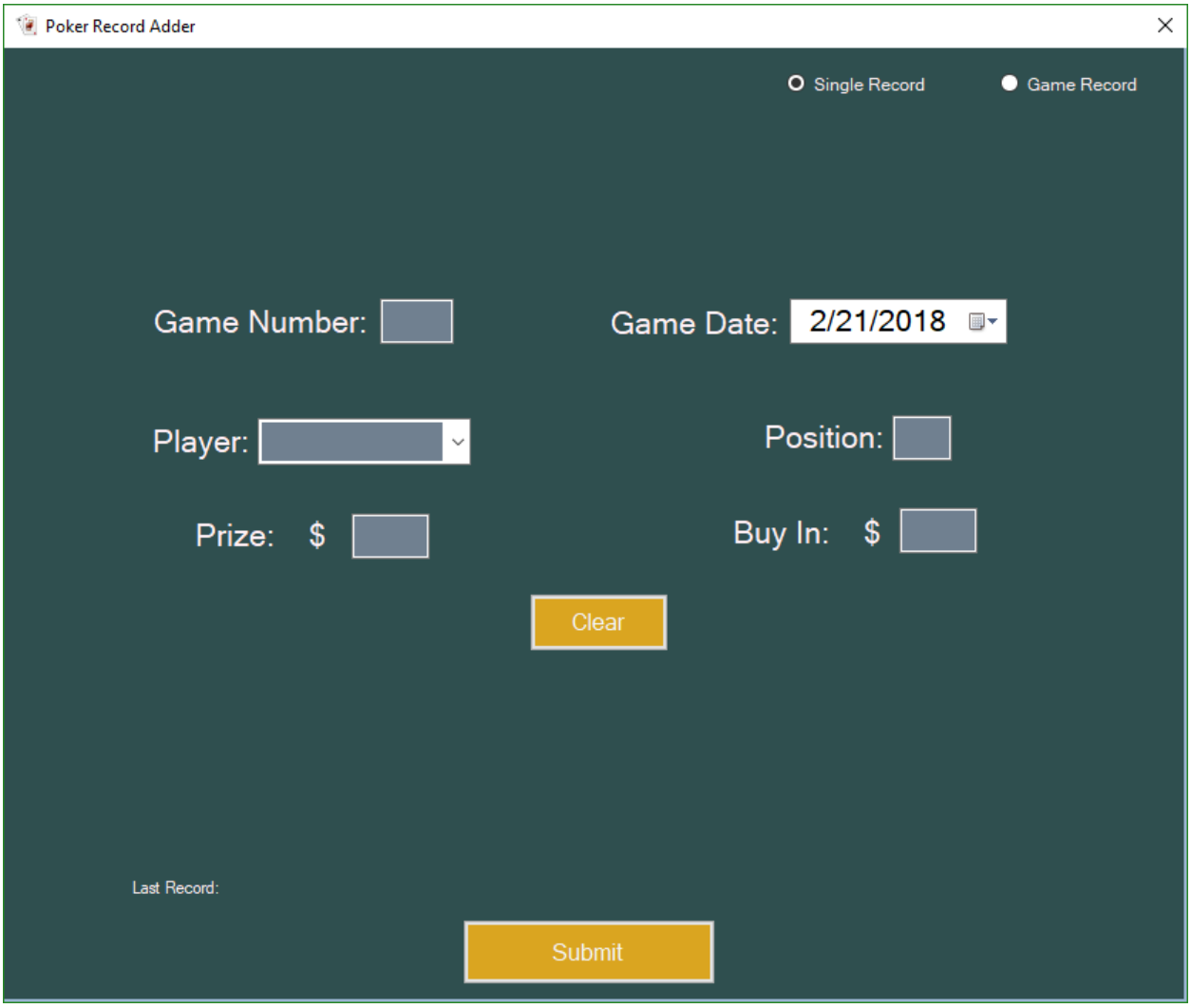


Above shows the interface displaying an error message when the user has duplicate player names for the same game. The user cannot use the same name more than once for a single game because of the primary key requirements in the database. This error message identifies the duplicate name for the user, so the correct changes can be made.



Above shows the interface displaying an error message for a null field. The submit button will be disabled if the game number, buy-in, or player field is null. This error message will be displayed if any of the records to be added to the database consists of a null value.

Adding a Single Record



Above shows the interface when the user has toggled from the full game record panel to the single record panel using the radio buttons. This interface is similar to the full game record panel. The interface consists of all the same fields, with the same labels just for the submission of a single record for a game to the database. None of the fields get auto-filled or updated, giving the user full control over the inputs. The user has the option to clear all the fields by using the "Clear" button on the interface. The application also checks to make sure none of the fields are null when the user attempts to submit the record into the database.