# StakeFree Digital Casino: Sprint 2

EECS Team 6

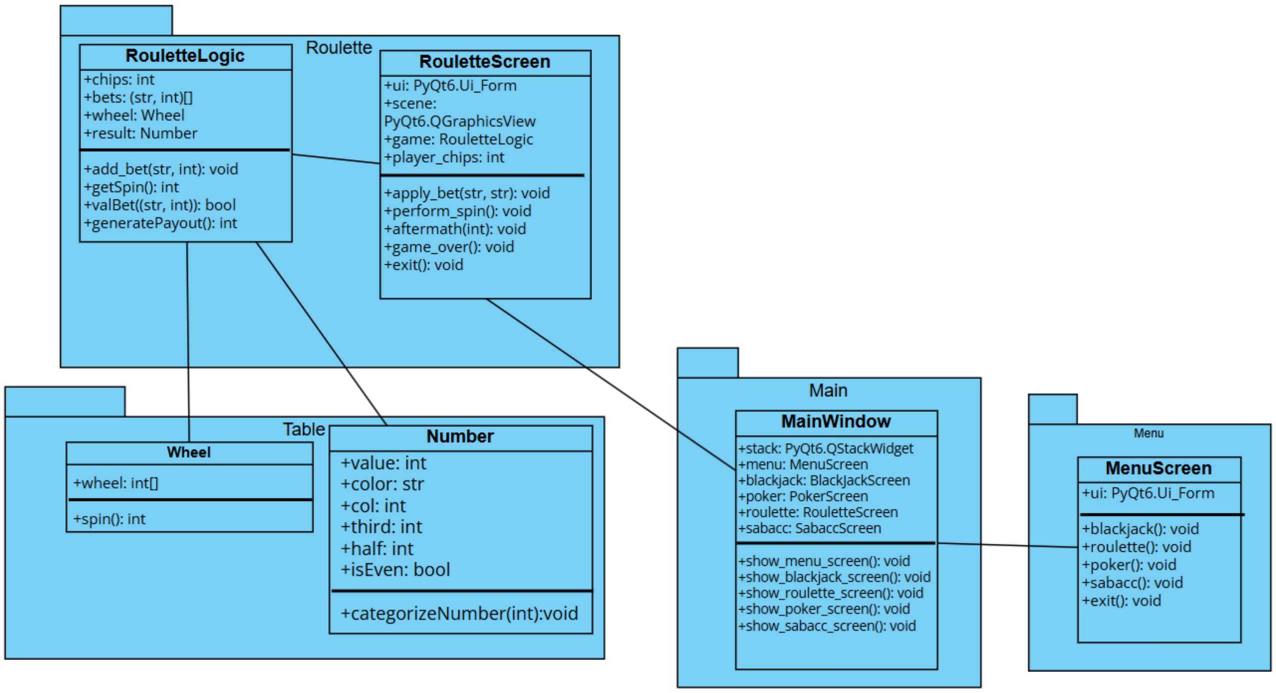
Bisshoy Bhattacharjee, Maximilian Biundo, Gavin Billinger, Mark Kitchin, Joshua Welicky

## Project Synopsis

StakeFree Digital Casino provides gamblers with the opportunity to play games such as Roulette, Blackjack, Poker, and Sabacc without the risk of financial loss.

## Architecture Description

The UML Class Diagram for the Sprint 2 architecture of StakeFree Digitial Casino is shown below.



### Dependencies

This project utilizes the PyQt6 libraries to implement GUIs. To install all required libraries, enter “pip install pyqt6-tools". This library provides a designer program that allows for the customization of a GUI in a user-friendly manner, saving the resulting GUI as a .ui file. This project also takes advantage of another tool provided by PyQt6, pyuic6, which converts a .ui file into a Python class for ease of use. Each element in the GUI is made available as a class attribute.

## Components

### *MainWindow*

- Description:
  - Core skeleton of the project. Manages switching between the menu GUI and game GUIs, as well as ultimate program termination.
- Attributes:
  - stack: A PyQt6.QStackedWidget instance that will store all of the GUI handlers, allowing switching.
  - menu: A MenuScreen instance which handles the main menu GUI.
  - blackjack: A BlackJackScreen instance which handles the blackjack GUI.
  - poker: A PokerScreen instance which handles the poker GUI.
  - roulette: A RouletteScreen instance which handles the roulette GUI.
  - sabacc: A SabaccScreen instance which handles the sabacc GUI.
- Methods:
  - show_[screen]_screen():
    - Simply switches to the desired screen (menu, blackjack, poker, sabacc, or roulette).

### *MenuScreen*

- Description:
  - A very simple GUI manager for the menu screen. It merely contains buttons that redirect to the games.
- Attribute:
  - ui: A PyQt6.Ui_Form that generates and manages the menu GUI.
- Methods:
  - blackjack(): Redirects to the blackjack game.
  - roulette(): Redirects to the roulette game.
  - poker(): Redirects to the poker game.
  - sabacc(): Redirects to the sabacc game.
  - exit(): Requests clean program termination.

### *Wheel*

- Description:

    - Logical implementation of the roulette wheel. It is simply an array of integers with a custom method for random sampling.
- Attributes:
    - wheel: An array of integers ranging from 0-36, ordered as they appear on a standard roulette wheel.
- Methods:
    - spin():
        - Returns a randomly selected integer from the wheel attribute. Meant to simulate a roulette wheel spinning.

### *Number*

- Description:
    - Represents a number in a game of roulette. It stores important information about a number, such as its membership in different betting groups (first 12, last half, etc.).
- Attributes:
    - value: The numerical value of the number.
    - color: A string representing whether the number is red or black.
    - col: An integer representing which column of the board it falls into.
    - third: An integer representing which third of the range the number falls into.
    - half: An integer representing which half of the range the number falls into.
    - isEven: A Boolean representing whether or not the number is even.
- Methods:
    - categorizeNumber(int):
        - Takes in an integer and fills in the attributes above based on standard roulette rules.

### *RouletteScreen*

- Description:
    - Manages the GUI displayed while playing roulette. Connects actions on GUI elements to roulette betting and playing actions.
- Attributes:
    - ui: A PyQt6.Ui_Form that generates the initial state of the roulette GUI.
    - scene: A PyQt6.QGraphicsScene that enables the addition of animated sprites to the GUI.
    - game: The RouletteLogic instance handling the actual actions of the game.
    - player_chips: The player's chip balance before any payouts or deductions take effect.
- Methods:
    - apply_bet(str, int):

        - Responds to actions on the Roulette board to place chips towards a specific bet. The integer represents an amount of chips. The string is a coded phrase to specify which bet the chips should go towards. Bet records are handled by RouletteLogic.
    - perform_spin():
        - Obtains a randomized wheel spin result from the RouletteLogic class, and animates the spinning of the board, ensuring that the proper number is landed on.
    - aftermath(int):
        - Passes the spin result back the GameLogic class to obtain the proper payout/deduction for the player. Applies those changes to the player's chip balance.
    - game_over():
        - Handles end-game actions, such as booting a player once chips run out, resetting the game state, and prompting replays.
    - exit():
        - Resets the game-state, deducts any forfeitures from the player's chip balance, and returns them to the menu.

## *RouletteLogic*

- Description:
    - Main class handling game logic for roulette, including the generation of wheel results, recording and validating of bets, and generating payouts.
- Attributes:
    - chips: The total number of chips bet by the user.
    - bets: An array of tuples, each tuple containing a string bet-code(pair of 1 and 2 = p_1_2;first twelve = tw_1) and an integer representing the number of chips bet.
    - wheel: A Wheel instance, which manages the generation of results.
    - result: A Number instance, which represents the result of a wheel spin.
- Methods:
    - add_bet(str, int):
        - Adds a bet sent by the RouletteScreen class to the bet record. If it already exists, its number of chips is merely increased. The string is the bet code, the integer is the number of chips bet.
    - getSpin():
        - Spins the wheel and returns a randomly selected integer. Returns the integer value, but also saves it as a Number to the result attribute.
    - valBet((str, int)):

        - Takes in a bet tuple and integer, presumably the wheel result, and returns true if the number is included in the bet's group. It could use the Number instance for ease of implementation.
    - generatePayout():
        - Using the result attribute, presumably the wheel results. It cycles through all bets in the record, accumulating(or deducting) the final payout of chips, which it then returns as an integer.