

Blockchain Casino

The aim of this project is to simulate an online casino using blockchain technology. PDF version of all markdown files are found in the [documentation](#) folder.

Source code

















The source code along with its commit history is publicly available on this [GitHub repository](#).

About the system

A smart contract representing the Roulette game has been developed. The croupier, the owner of the smart contract, opens the table by creating a new instance and setting the table stake on creation. In this scenario, players are allowed to bet on odd or even numbers.

After the winning number is set, the players can check if they have any withdrawable balance and request to withdraw said winnings.

Folder Structure

 .git	11/02/2019 13:42	File folder	
 build	12/01/2019 08:42	File folder	
 contracts	26/01/2019 09:34	File folder	
 documentation	12/02/2019 12:12	File folder	
 migrations	26/01/2019 09:34	File folder	
 node_modules	11/02/2019 13:16	File folder	
 screenshots	12/02/2019 12:12	File folder	
 src	02/02/2019 15:50	File folder	
 test	11/02/2019 09:24	File folder	
 .gitignore	12/01/2019 09:19	Text Document	1 KB
 bs-config.json	20/12/2018 18:25	JSON File	1 KB
 commands.txt	02/02/2019 15:56	Text Document	1 KB
 package.json	27/01/2019 08:30	JSON File	1 KB
 package-lock.json	11/02/2019 13:16	JSON File	118 KB
 README.md	11/02/2019 13:29	MD File	2 KB
 truffle.js	11/02/2019 12:52	JavaScript File	1 KB

Usage: Smart Contract

The project has been written in Solidity, using Truffle as a development environment and Ganache as a local RPC server running on the port 8545. The project can be compiled, tested and deployed by opening the CLI at the root folder of the project and executing the commands below.

```
truffle compile
truffle test
truffle migrate --reset
```

The `--reset` flag ensures migrations are run from the beginning ([source](#)).

Tests execution

```
Nicholas Vella@DESKTOP-ENDC34M MINGW64 ~/blockchain-casino (master)
$ truffle test
Using network 'development'.

Compiling .\contracts\Roulette.sol...

Contract: Roulette
  ✓ should return the correct table stake value
  ✓ should not create a session if a player has never placed a bet
  ✓ should create a session if a user placed a bet (67ms)
  ✓ should not allow croupier to place a bet
  ✓ should not allow bets which do not match stake
  ✓ should not allow duplicate bets (91ms)
  ✓ should not allow betting on both odd and even numbers (78ms)
  ✓ should not allow betting when the table is closed for betting (66ms)
  ✓ should set the winning number correctly (95ms)
  ✓ should not set the winning number if bets are allowed
  ✓ should not set the winning number if input is invalid (52ms)
  ✓ should not set the winning number twice (147ms)
  ✓ should provide the correct withdrawable balance (394ms)
  ✓ should not allow to withdraw winnings twice (421ms)
  ✓ should not allow losing accounts to withdraw (204ms)
  ✓ should allow the croupier to destroy the contract (423ms)

16 passing (4s)
```

Usage: Frontend

The frontend has been written using HTML, Bootstrap 4.0 and jQuery. Bootstrap and jQuery are served using CDNs. An npm package called `lite-server` is used to run the frontend application locally ([source](#)). With the CLI open in the directory of the project, run the following commands.

```
npm install
npm run dev
```

The application should open automatically on <http://localhost:3000/>