### Betting system for Gladiator

[**Users Contract(i.e. The ones who bet)**](#_s8j8d392kfqi) **2**

[placeBet()](#_c8xthu6ec9ym) 3

[checkOutcome()](#_mrubtisplbe6) 4

[matchExists()](#_k6awyg8jtn3s) 4

[getMostRecentMatch()](#_wl8vasa4lxil) 6

[\_matchID()](#_vssfj6ycj2zx) 7

[**Oracle (i.e. Administrator)**](#_owmxs1ogw6sv) **8**

[addMatch()](#_ep08wibvdpu3) 8

[declareOutcome()](#_hsy8pfafx74i) 10

[setMatchCancelled()](#_87cdbi2u47y0) 11

[setMatchUnderway()](#_435gaup6z3v) 12

[matchExists()](#_oj02meavrk6t) 13

[getMostRecentMatch()](#_ogvmctgs6a0i) 14

[\_matchID()](#_dzabt4iel7iz) 15

### 

### Users Contract(i.e. The ones who bet)

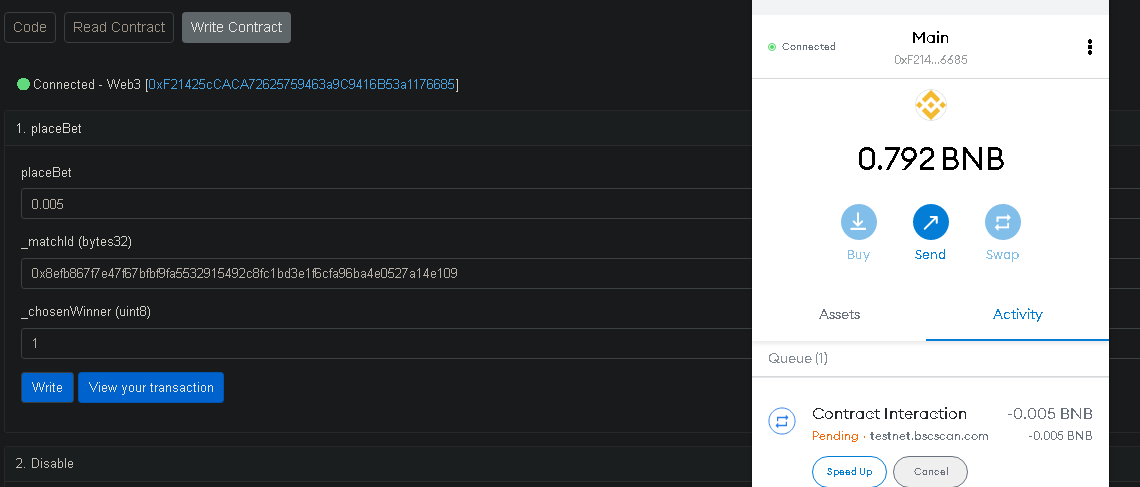
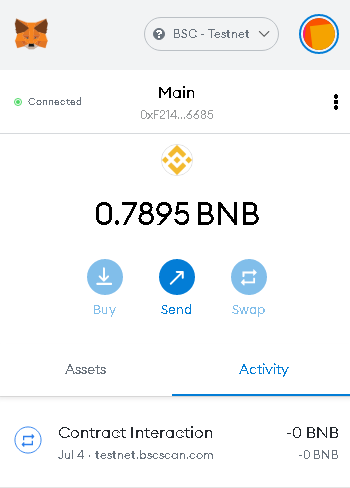
The users can interact with the contract in the following ways:

* Check if match exists via matchExists()
* Check the last Match created via getMostRecentMatch()
* Check a specific match via \_matchID()
* Place bets via placeBet()
* Get paid if win checkOutcome()

##### 

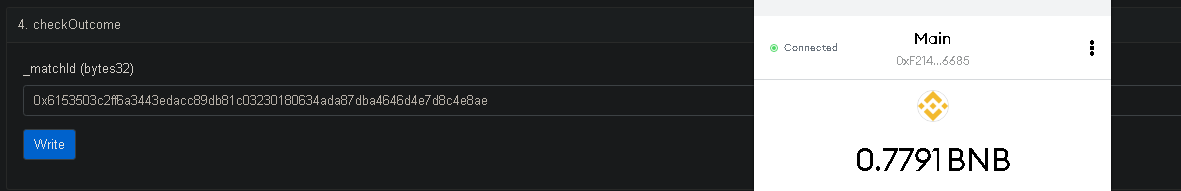
##### placeBet()

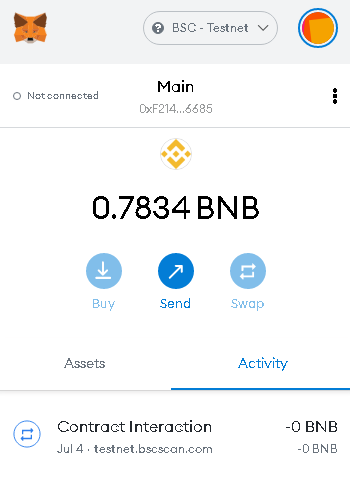
Places bet on the team.

Amount, Match ID and Winner (starts at 0)

##### checkOutcome()

Users get payed

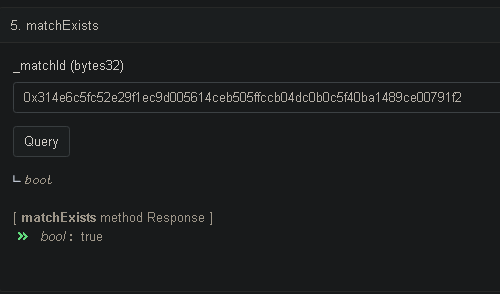




##### 

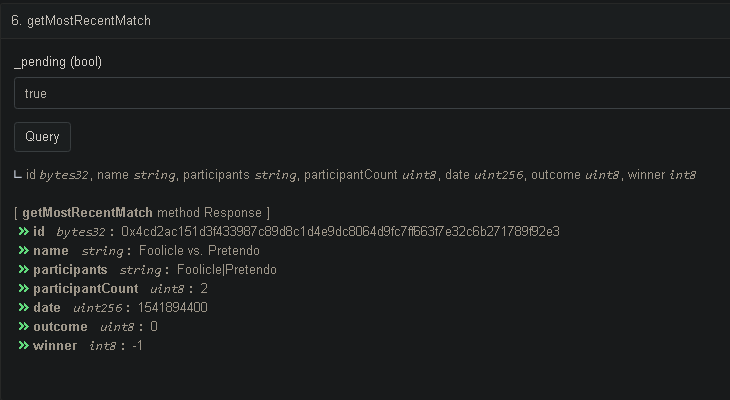
##### matchExists()

Returns true if match exists



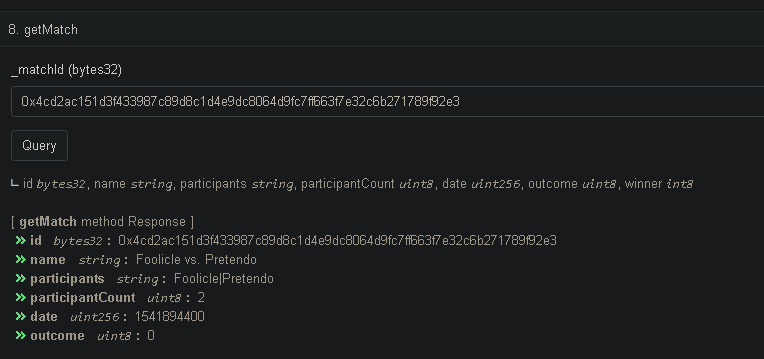
##### getMostRecentMatch()

Shows the latest match



##### \_matchID()

Shows info about a match



### 

### Oracle (i.e. Administrator)

The objective of the Oracle is to have a third party, trusted person, that would control the bets. His capabilities are:

* Creating new bet through addMatch()
* Declaring outcome through declareOutcome()
* Cancel match through setMatchCancelled()
* Set the outcome as pending if it wasn’t declared yet through setMatchUnderway()

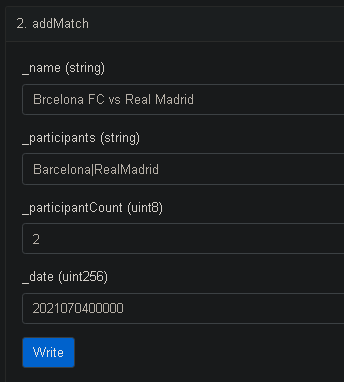
The objective of the Oracle is to have a third party, trusted person, that would control the bets. His capabilities are:

* Creating new bet through addMatch()
* Declaring outcome through declareOutcome()
* Cancel match through setMatchCancelled()
* Set the outcome as pending if it wasn’t declared yet through setMatchUnderway()

Over this contract it’d also be possible to the public to:

* Check if match exists via matchExists()
* Check the last Match created via getMostRecentMatch()
* Check a specific match via \_matchID()

##### addMatch()



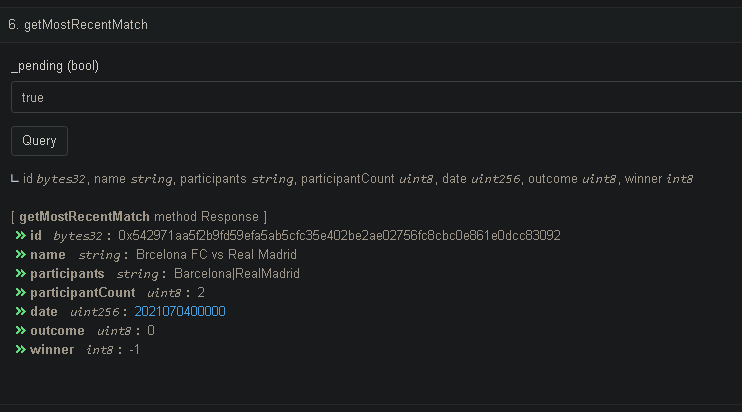
Name for the public

Participants it has to be “ParticipantA|ParticipantB|Participant...Z”

Number of participants

Date of match YYYYMMDDHHMM

Result



The match ID needs to be sent to the persons who are going to bet in order for them to bet on that match.

##### 

##### 

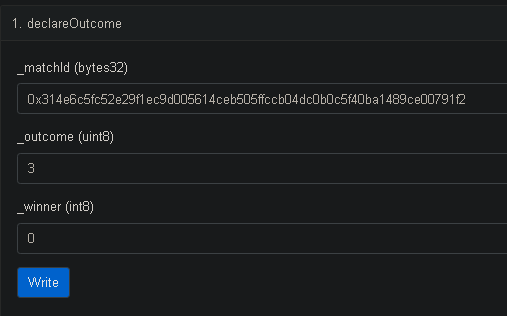
##### declareOutcome()

Possible Outcomes:

0. Pending, //match has not been fought to decision

1. Underway, //match has started & is underway
2. Draw, //anything other than a clear winner (e.g. cancelled)
3. Decided, //index of participant who is the winner
4. Cancelled //match was cancelled; there's no winner

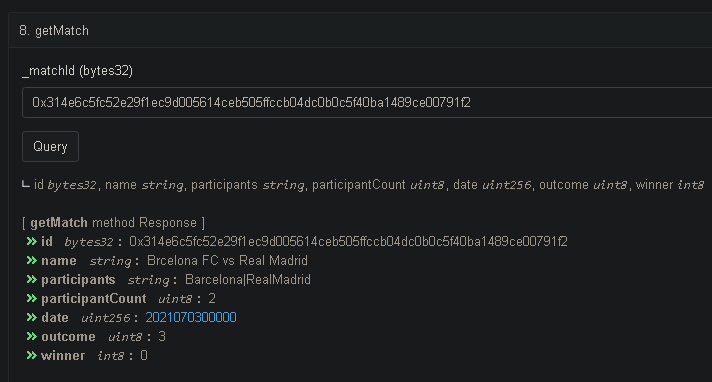
(NEED TO SET UNDERWAY BEFORE DECIDED)



Match ID

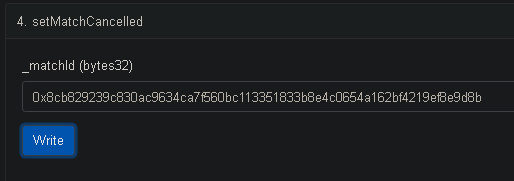
Outcome

Winner



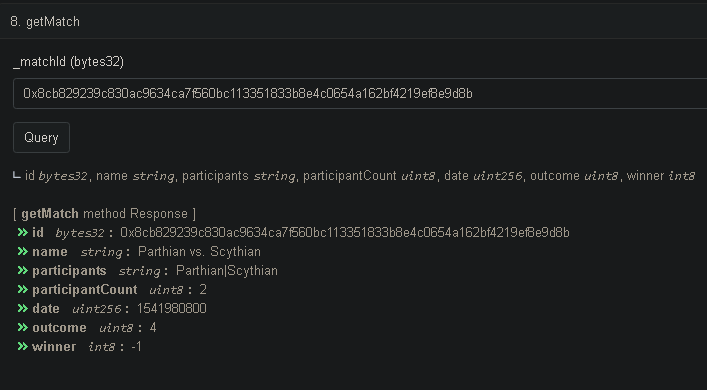
##### setMatchCancelled()

Cancels match



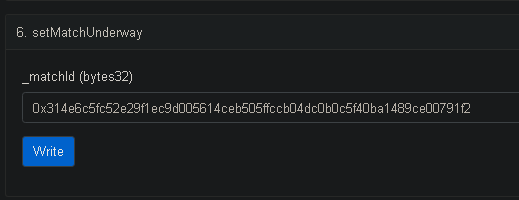
Match ID

Result, outcome 4 = Canceled

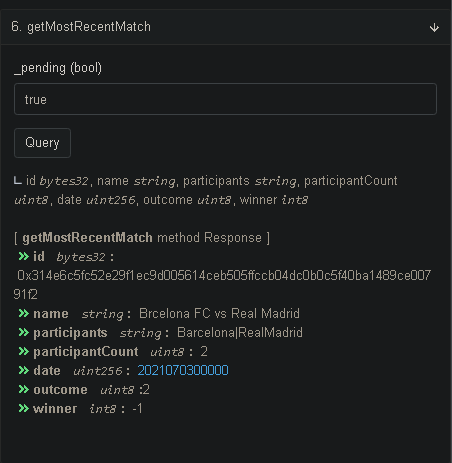


##### setMatchUnderway()

Previous step before deciding the match has finished

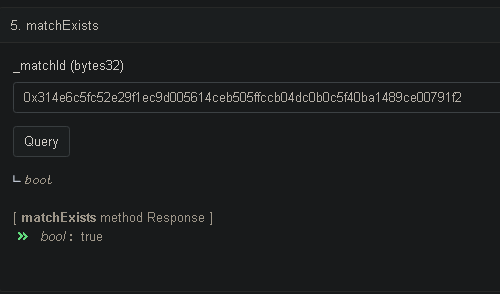


Result



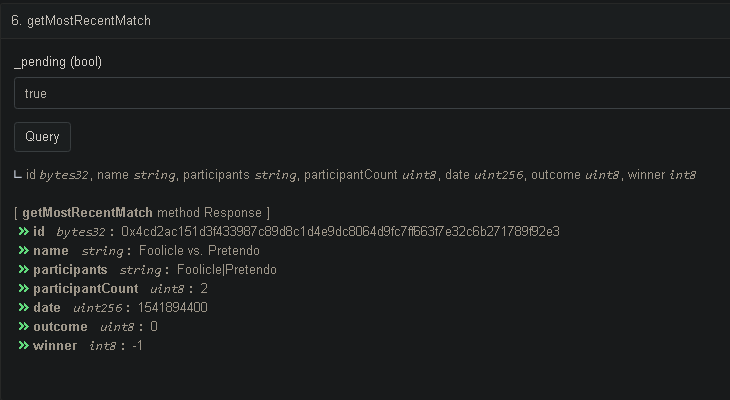
##### matchExists()

Returns true if match exists



##### getMostRecentMatch()

Shows the latest match



##### \_matchID()

Shows info about a match

