

ESPORTS BETTING NETWORK

Lower Fees. Lighter Conscience.



Esports Betting Network (ESBN) is a smart contract built on the Ethereum network which aims to provide a trustless system to enable peer-to-peer betting on your favourite esports. We aim to provide coverage for both professional matches, and live streamed matches of professional and non-professional gamers. As opposed to traditional betting systems where the house takes a significant portion, we instead aim to provide lower fees to the end user and offset the social costs of providing gambling services by donating to the community.

By utilizing smart contracts and publicly available information of match results we can provide instantaneous and automatically realized payouts. The peer-to-peer nature and use of a betting pool allows us to let the market decide the odds, which alleviates the risk associated with taking bets, and in turn allows us to offer far better payout odds to our clients.

Our Team



Solas

Legal Advisor



Comet

Marketing Consultant



Matt

CEO



Tidus

Senior Programmer



Oberon

Junior Programmer

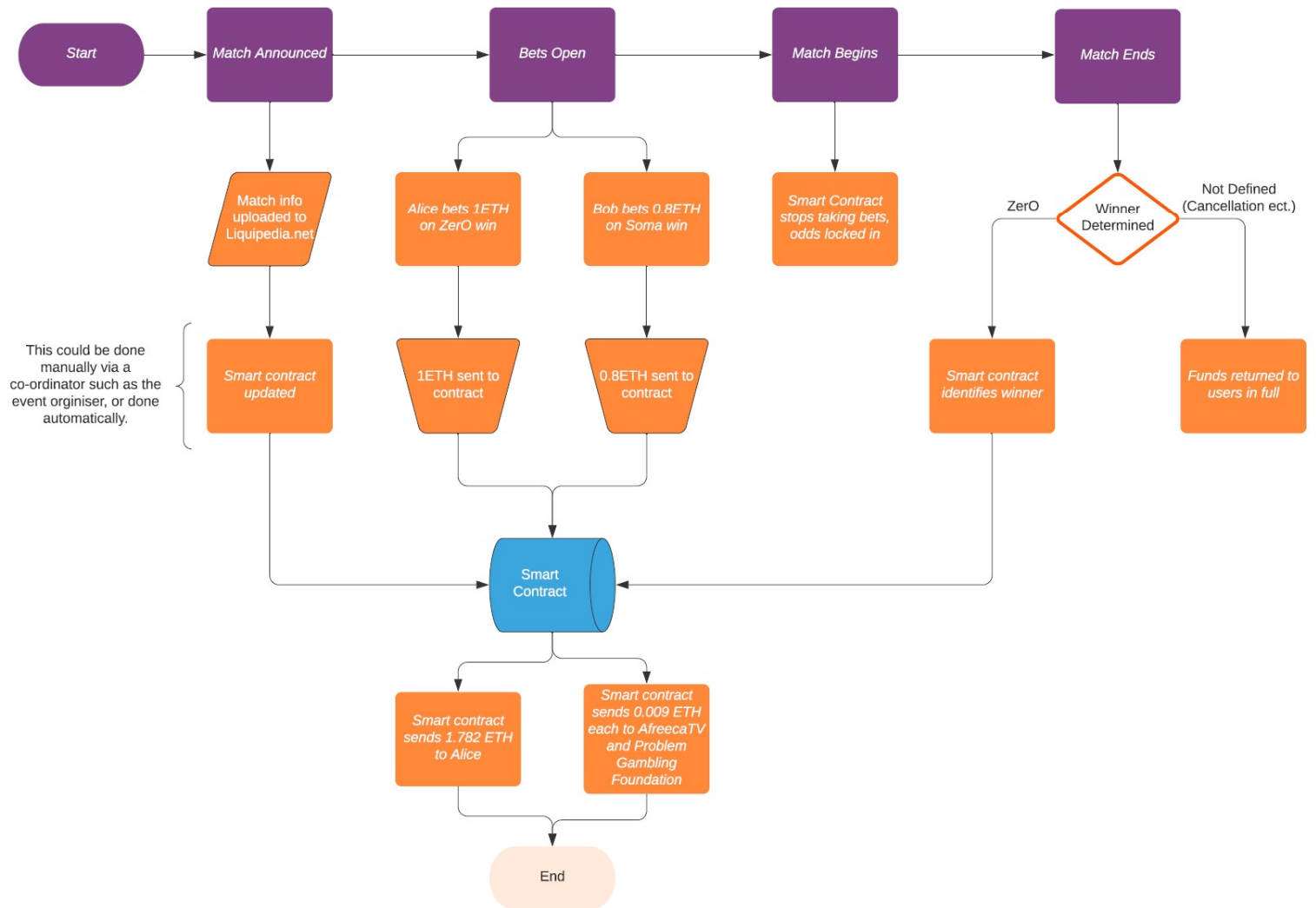
Pitch Presentation



[ESBN Pitch.pptx](#)

Esports Betting Network - ASL S10 Finals Example

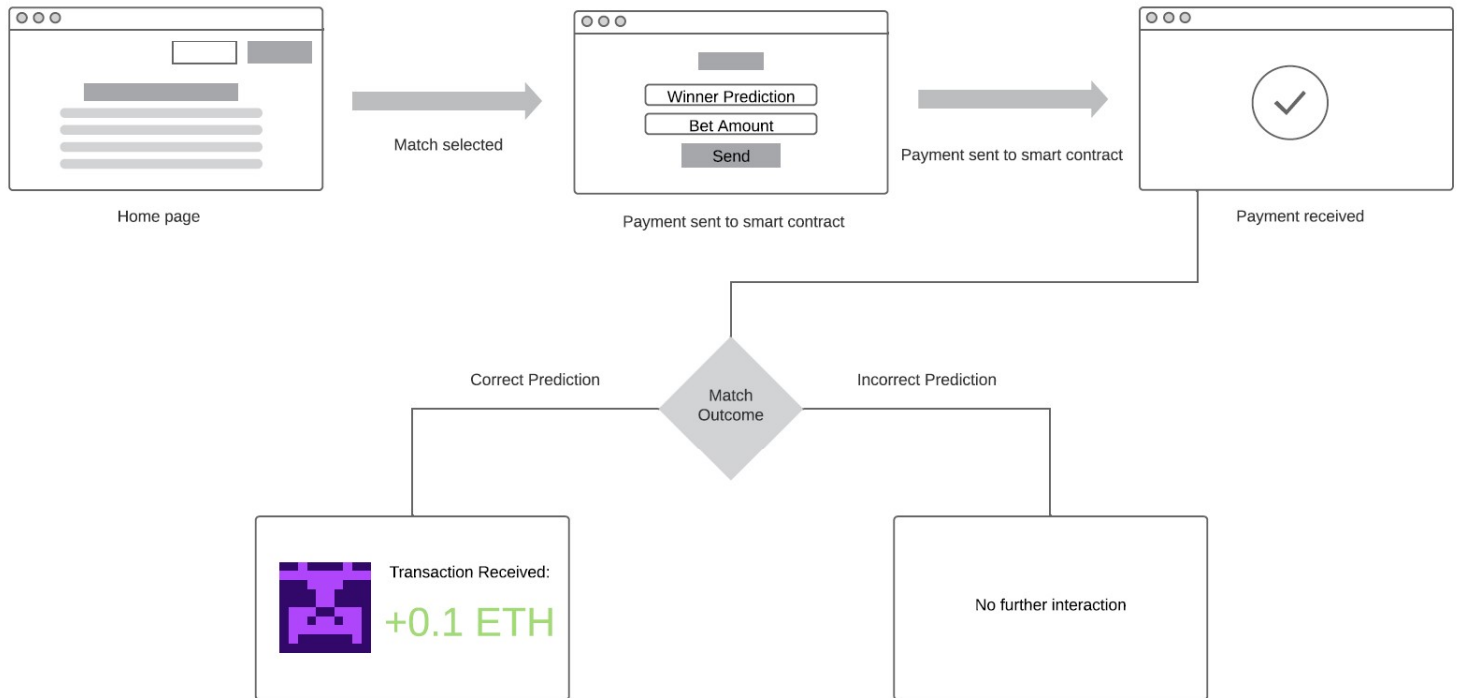
Matt Hill | November 30, 2020



Bet Placement Task Flow

Matt Hill | December 2, 2020

Goal: A user places a bet on a match



Frequently Asked Questions

What is the problem we are trying to solve?

Existing esports betting websites take a large cut of bets and offer poor odds for the end user, by operating as a decentralized autonomous organization we can eliminate most operating costs to provide better odds for users and by using open-source smart contracts we can provide peace of mind to users that their bets will be paid out in full, on time.

Is blockchain necessary?

Blockchain is an important tool to our approach, as the public ledger provides a way for users to verify that bets are being taken and paid out, and that the donations to event organizers and charities are also being paid.

How do I use the ESNB smart contract?

All that is needed to interact with the smart contract is an Ethereum wallet and Ether, for a full walkthrough check out our tutorials. We also currently plan to release both mobile and web applications which will simplify and streamline the entire process.

What functionality does the prototype have?

The current prototype of the ESNB currently has functionality for direct peer-to-peer bets of equal value, and it utilizes a coordinator instead of automated match upload and winner confirmation. In this regard it is not yet fully decentralized however we are able to see that this system works as a proof of concept.

What features can we expect from the smart contract in the future?

Some future features are the implementation of a betting pool to allow the market to decide odds for matches, refactoring the date system to operate on Unix time so that bets can be placed up until the time that a match starts, and removal of the coordinator in favor of further automation to allow us to run completely decentralized and autonomous.

Are there any other projects in development?

We are currently planning to perform a third-party security audit to ensure that the system is unable to be compromised and plan to provide mobile and web applications to simplify and streamline the process of interacting with the smart contract for our clients.

Prototype Tutorial

Setting up the smart contract.

Before we can begin interacting with the smart contract, we must first deploy it on the test-net, we will do this by utilizing features of the Ethereum studio web IDE.

Step 1: Open the web IDE

Go to <https://studio.ethereum.org/5fcad05926b473001237384f>

Step 2: Enable/Disable Anonymous Tracking

Either press save, or first disable anonymous tracking and then press save.

Enable Anonymous Tracking

By allowing us to track (anonymously) how you use Ethereum Studio, we can get valuable insights and better understand what the tool needs in order to make you a happier developer. By leaving the tracking on, you will really help us see what can be improved, iterated or removed.

We do not track any personal data, private keys or addresses. Ethereum Studio is Open Source, so you can either check out the [code](#) for yourself or read more on what we track and why in our [Help Center](#).

Thanks for your support and happy building!

Allow tracking (anonymously)

You can always change this from the Settings dialog.

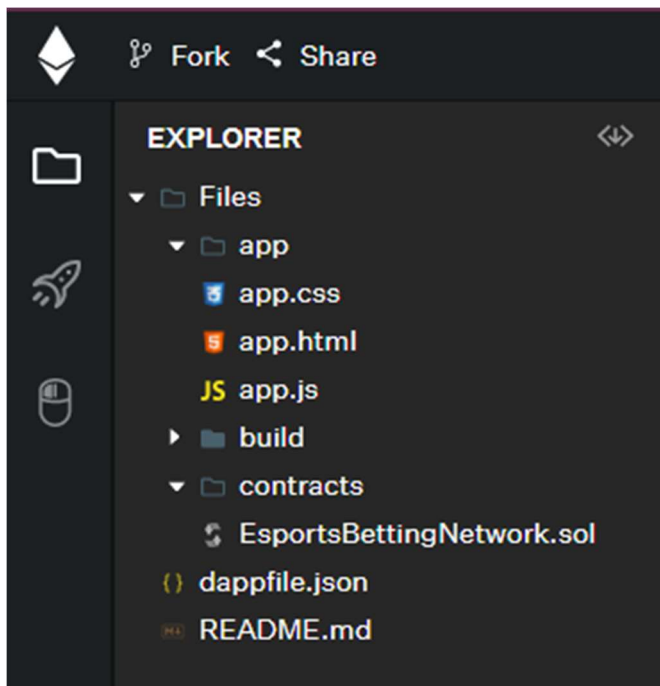
☒

Save

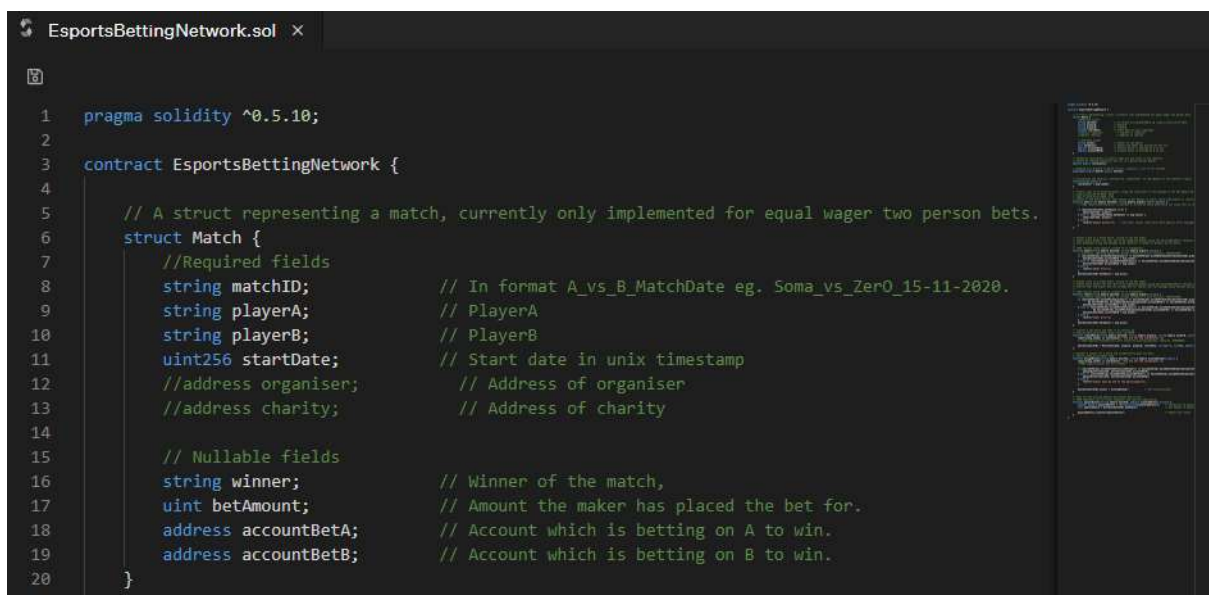
For the purpose of this tutorial anonymous tracking has no effect on results and is your decision whether to enable or disable.

Step 3: Viewing the Code

Left click **EsportsBettingNetwork.sol**

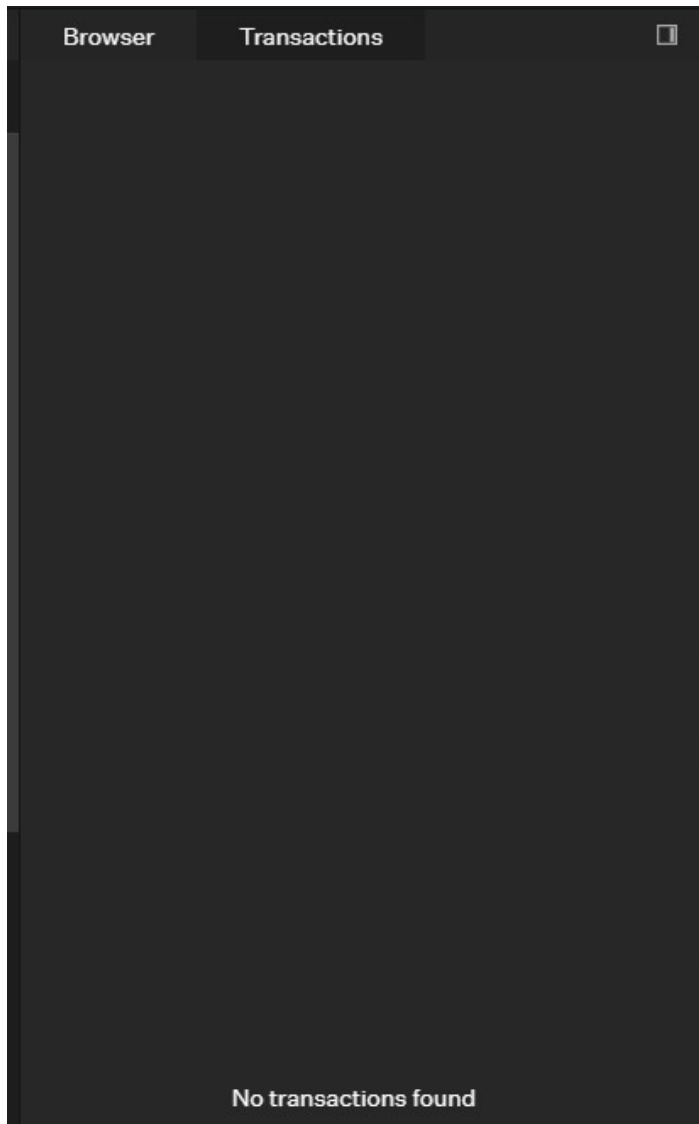


This will open the smart contract code in the explorer, allowing us to inspect the inner workings of the smart contract.



Step 4: Open the transaction viewer

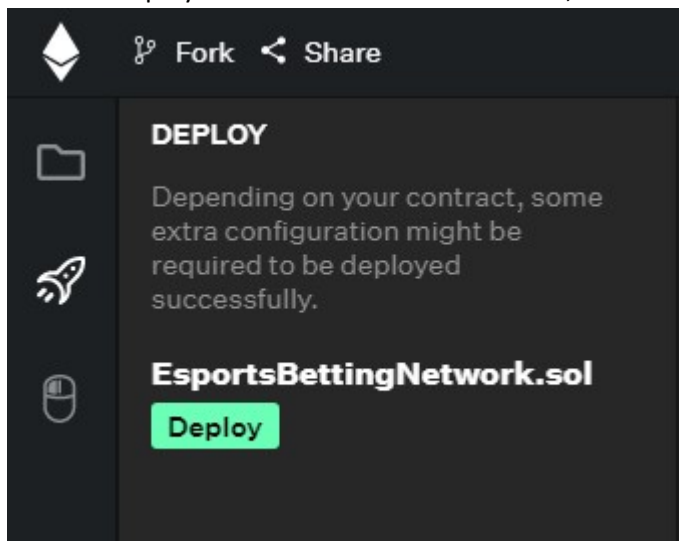
Left click the transaction tab on the right-hand side.



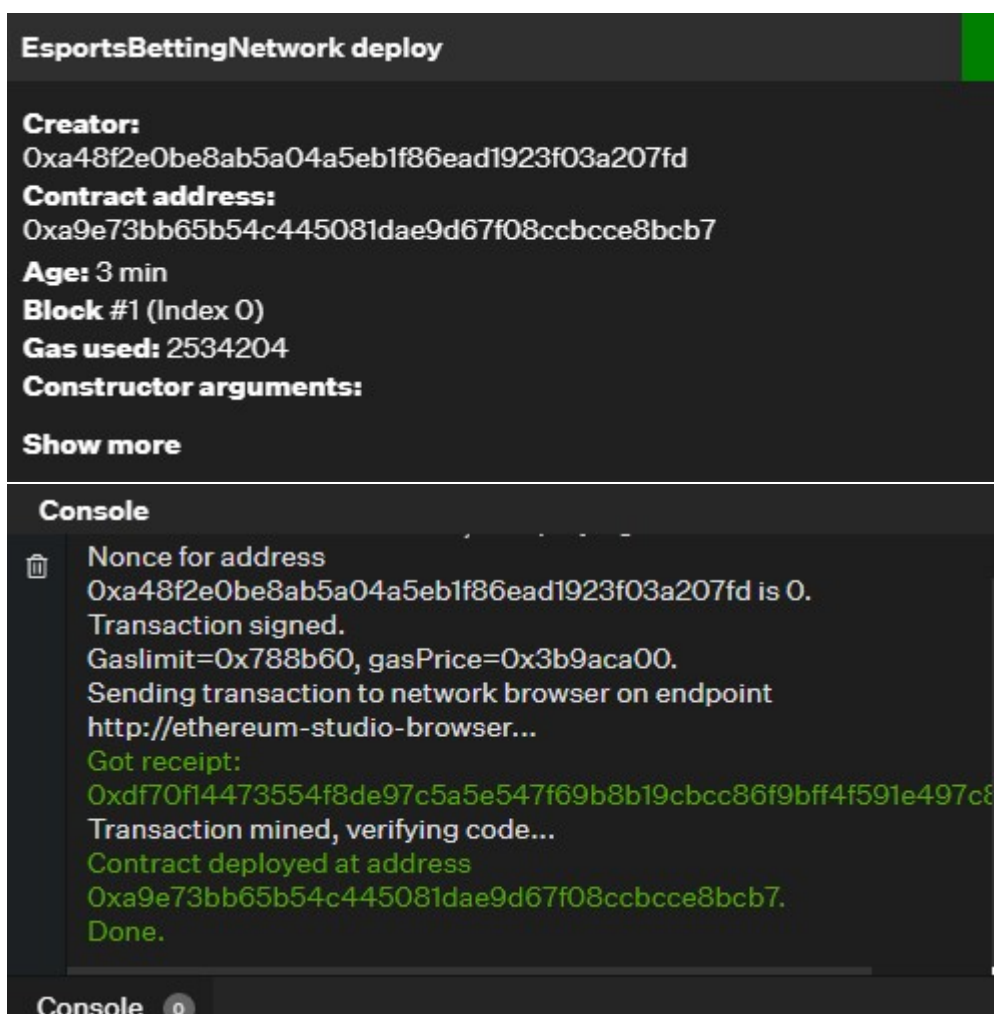
This will allow us to view the transactions taking place on the network once we deploy the contract on the test-net.

Step 5: Deploying the Smart Contract

Click the deploy menu on the left-hand sidebar, then select deploy.

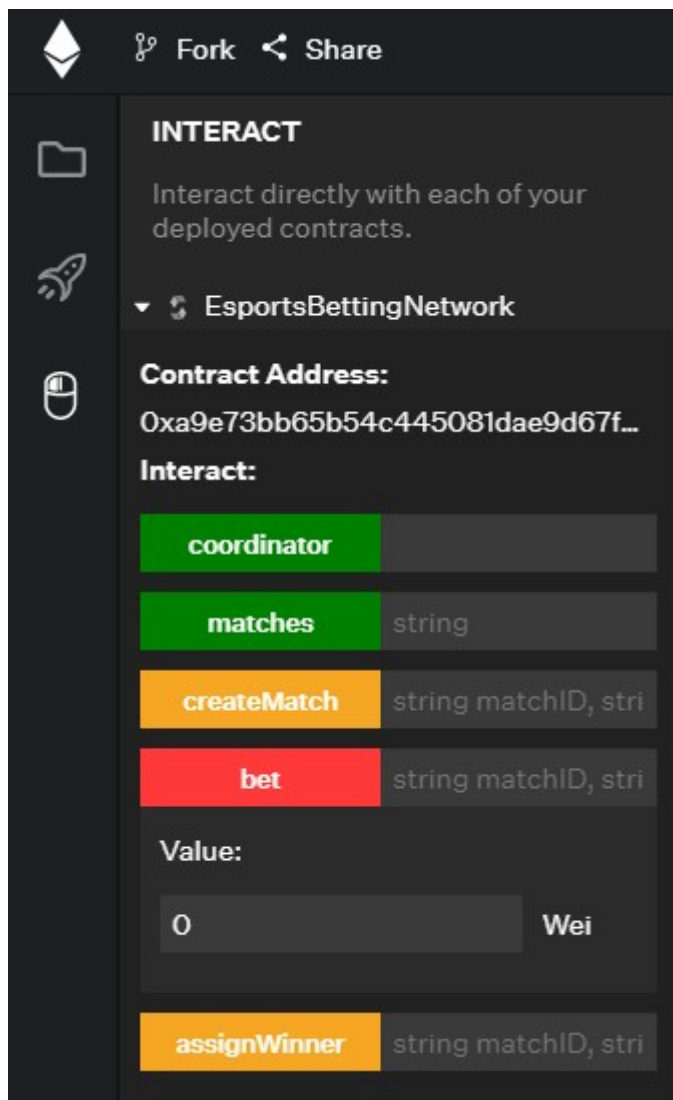


We can see the transaction on the test-net which deployed the smart contract, we are able to see the address of the deployer, in this case this is the coordinator. This means that special functions will be available for this account and only this account, such as creating matches and assigning winners.



Step 6: Opening the interaction console

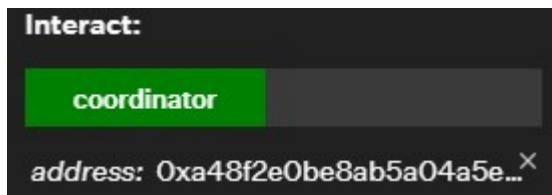
Click the interact menu on the left-hand sidebar, then select EsportsBettingNetwork.



This is the interaction menu, here is where we will be directly calling methods contained within the smart contract to interact with it. In a later build, this would instead be called via a web app utilising an existing framework such as JavaScript for ease of accessibility.

Step 7: Checking the contracts coordinator's address

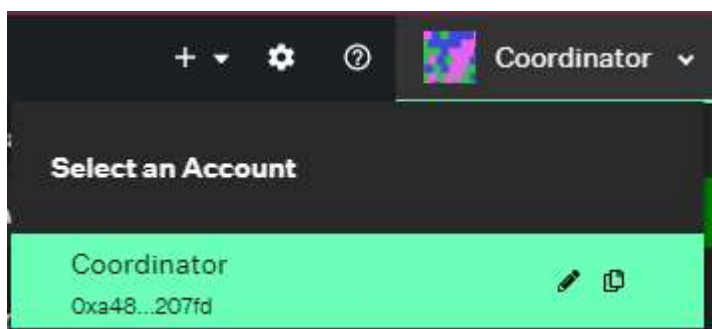
Click the green coordinator button to see the coordinator's address



This is the address of the deployer, referred to as the coordinator, they will have greater access to the smart contract but will be removed in favour of autonomous methods in a later build.

Step 8: Checking that the coordinator address matches the contracts coordinator address

Click the account selection dropdown in the top right, and check that the address of the coordinator matches the coordinator of the currently deployed smart contract.



Here we can see that the addresses do match, therefore the account named coordinator is the coordinator for the deployed contract. This gives this address extra functionality available.

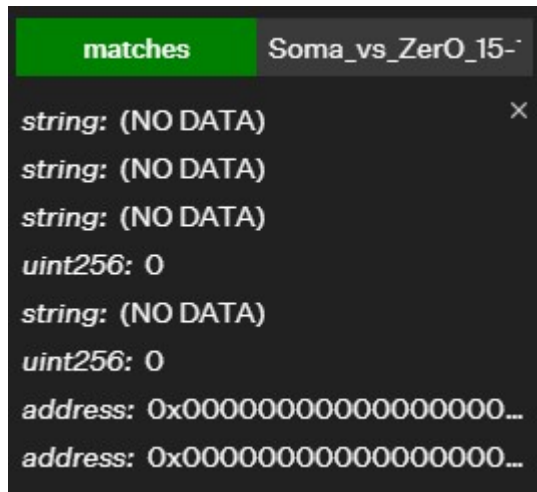
Creating a match.

The first step in our betting procedure is for a match to be uploaded to the system, in this section we will learn how to query the network for an existing match, and how to create a new match.

Step 1: Checking for an existing match

Enter the following in the matches text field and click the green button.

Soma_vs_ZerO_15-11-2020



The parameter used for searching for matches is the match ID, which consists of the participants in alphabetical order, followed by the match date.

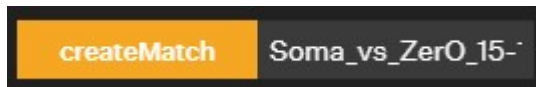
Here we can see that if we search for a match which does not exist all fields display as their default values (NO DATA) for strings, 0 for integers, and 0x0 for addresses.

This function is available to all users and as it does not cause any internal data to change it does not cost any ether to perform.

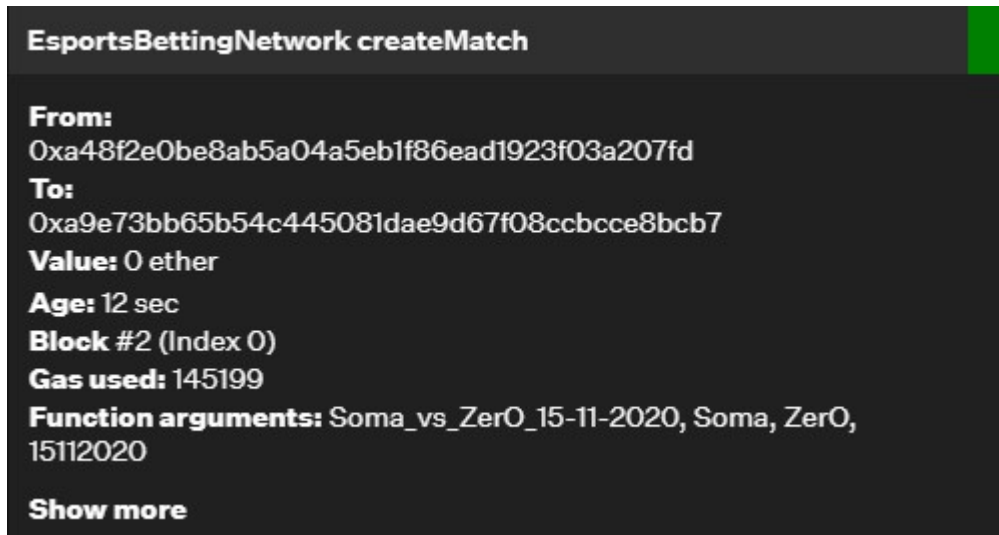
Step 2: Creating a new match

Enter the following in the createMatch text field and click the orange button.

Soma_vs_ZerO_15-11-2020, Soma, ZerO, 15112020



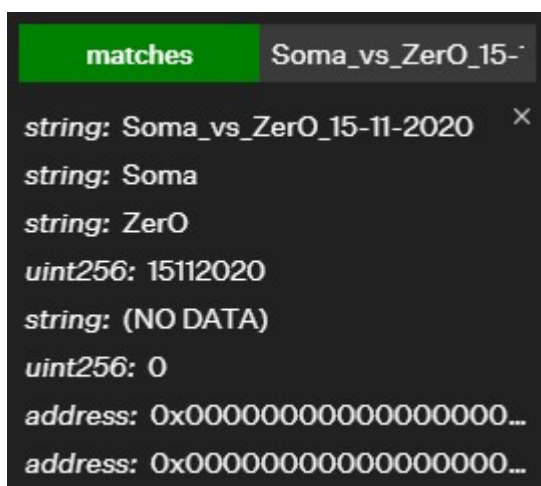
The parameters used for creating a match are the match ID, followed by the participants in alphabetical order, and finally the match date.



Step 3: Checking for an existing match

Ensure the following text is still contained within the matches text field and click the green button.

Soma_vs_ZerO_15-11-2020



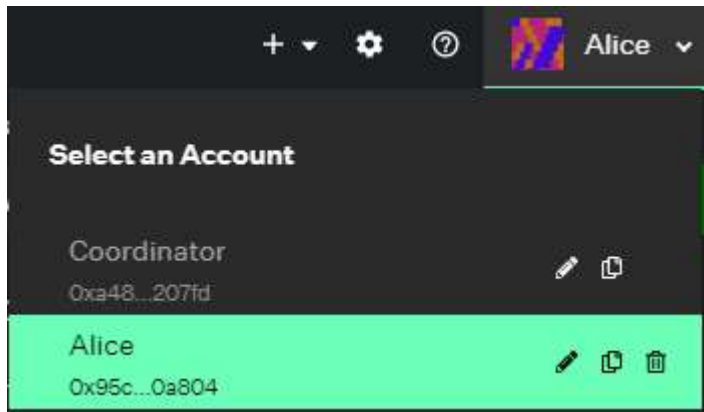
Repeating the process of checking for a match lets us see there is now an entry with the non-null fields entered. We have now successfully created a new match for the ESBN smart contract.

Placing bets.

Now that we have learned how to upload new matches to the system we will now try and place bets on the existing match.

Step 1: Changing account to Alice

In the top right-hand corner, click on the account selection button and select Alice from the dropdown menu.

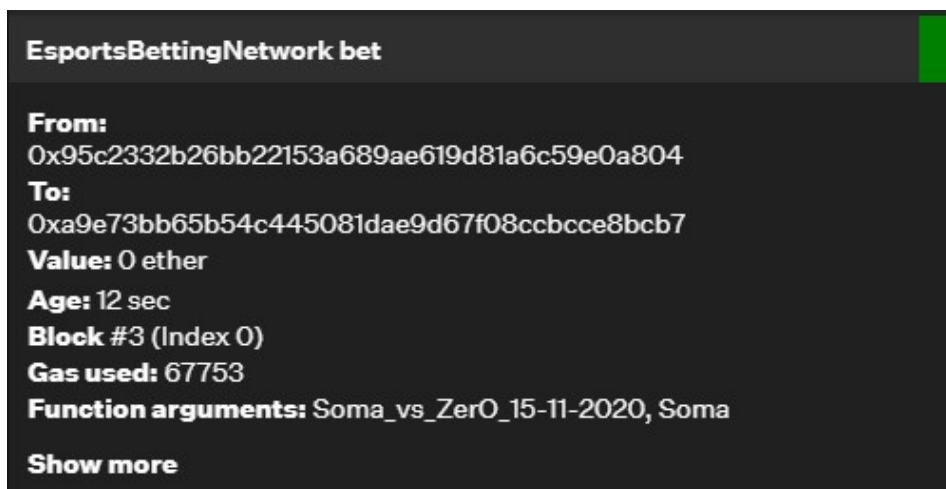


This changes the current address to a new one, that of Alice's, therefore we lose access to the coordinator restricted methods.

Step 2: Placing a bet on Soma

Enter the following in the bet text field, set the value to 2000 Wei and click the red button

Soma_vs_ZerO_15-11-2020, Soma

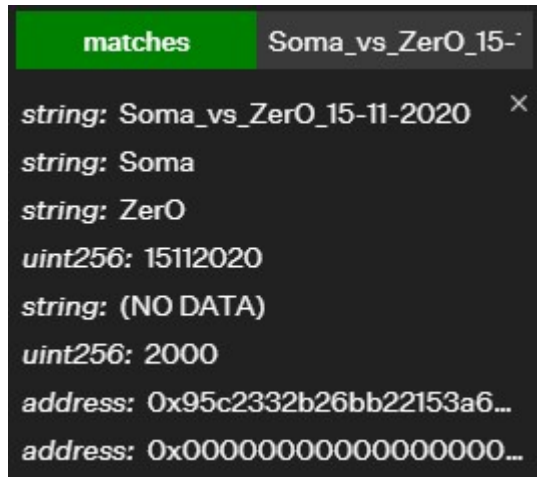
A screenshot of a bet placement form. At the top, there is a red button labeled "bet" and a text field containing "Soma_vs_ZerO_15-". Below this, the label "Value:" is followed by a text input field containing "2000" and a unit selector dropdown menu currently set to "Wei".

The parameters for the bet function are the match ID, followed by the participant that the user is placing a bet on winning. The bet amount is the amount of ether sent in the transaction.

Note: There seems to be a bug causing the value to display 0 when viewed in the transaction viewer.

Step 3: Checking the bet in the system

Click the matches button again to refresh and see the betAmount has been set, and one of the address fields now contains Alice's address.



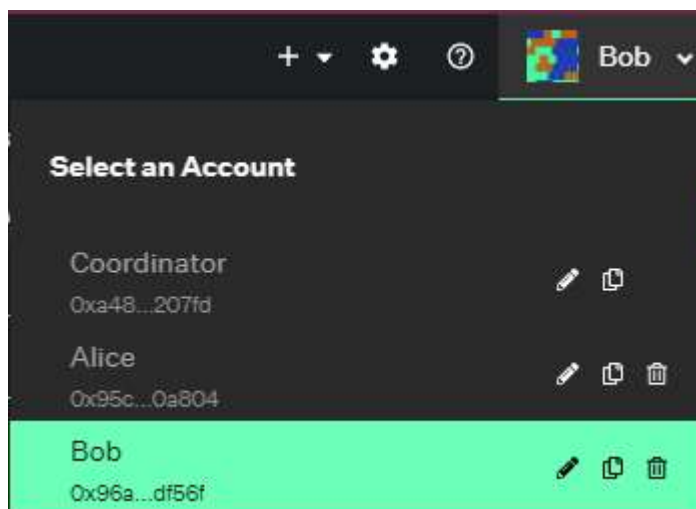
Even though we received a receipt in the console and can see the transaction which took place on the network, we still want to make sure that the system has been updated correctly. Here we can see that our address is correct, and the amount that we sent in the transaction matches that stored in the amount in the smart contract's records.

We have now successfully placed a bet on a player, from here Alice's bet will either be taken up by another user, in which case she will receive her pay-out if she wins or if no one takes up the bet she will be returned in full her initial amount.

NOTE: Automatic fund return not yet implemented.

Step 4: Changing account to Bob

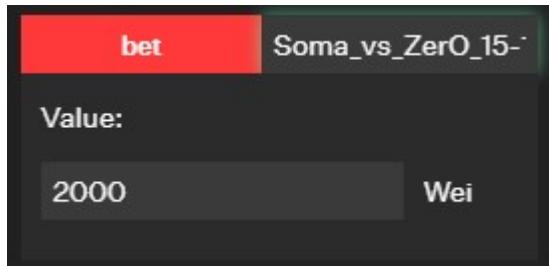
In the top right-hand corner, click on the account selection button and select Bob.



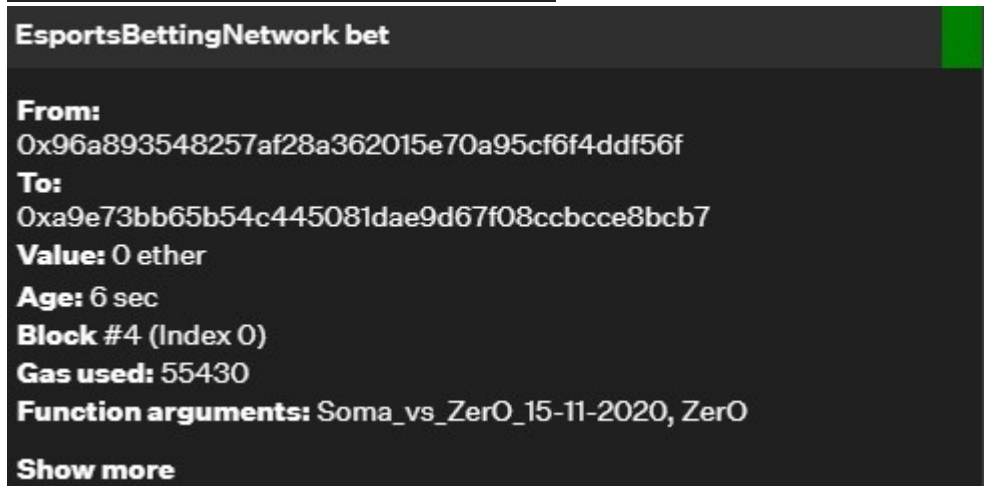
Step 5: Placing a bet on ZerO

Enter the following in the bet text field, set the value to 2000 Wei and click the red button

Soma_vs_ZerO_15-11-2020, ZerO



A screenshot of a web interface showing a red button labeled 'bet' and a text input field containing 'Soma_vs_ZerO_15-'. Below the input field, there is a label 'Value:' followed by a text box containing '2000' and a unit selector dropdown menu currently set to 'Wei'.

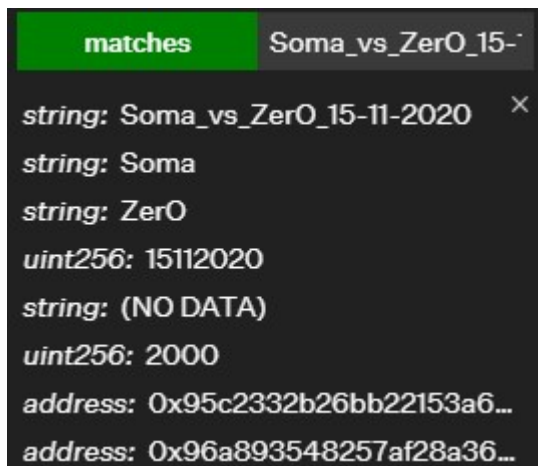


A screenshot of a transaction confirmation window titled 'EsportsBettingNetwork bet'. It displays the following details: 'From:' followed by a long hexadecimal address, 'To:' followed by another long hexadecimal address, 'Value: 0 ether', 'Age: 6 sec', 'Block #4 (Index 0)', 'Gas used: 55430', and 'Function arguments: Soma_vs_ZerO_15-11-2020, ZerO'. At the bottom, there is a 'Show more' link.

The parameters for the bet function are the same when taking up a bet which has already been placed, however the function will revert if the bet amount and the transaction value differ.

Step 6: Checking the bet in the system

Click the matches button again to refresh and see both addresses are populated.



A screenshot of a web interface showing a green button labeled 'matches' and a text input field containing 'Soma_vs_ZerO_15-'. Below the input field, there is a list of data items: 'string: Soma_vs_ZerO_15-11-2020' (with a close icon), 'string: Soma', 'string: ZerO', 'uint256: 15112020', 'string: (NO DATA)', 'uint256: 2000', 'address: 0x95c2332b26bb22153a6...', and 'address: 0x96a893548257af28a36...'.

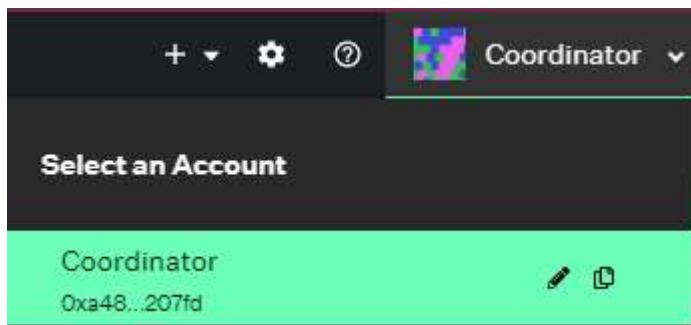
We have now seen the process of placing bets in full and can see that upon the development of a web application our user experience will be very intuitive.

Assigning a Winner.

Now that both accounts have placed their bets the only remaining step is for the coordinator to assign the winner of the match. Then the smart contract will automatically calculate the pay-out and distribute it to the proper addresses.

Step 1: Changing account to the Coordinator

In the top right-hand corner, click on the account selection button and select the Coordinator.

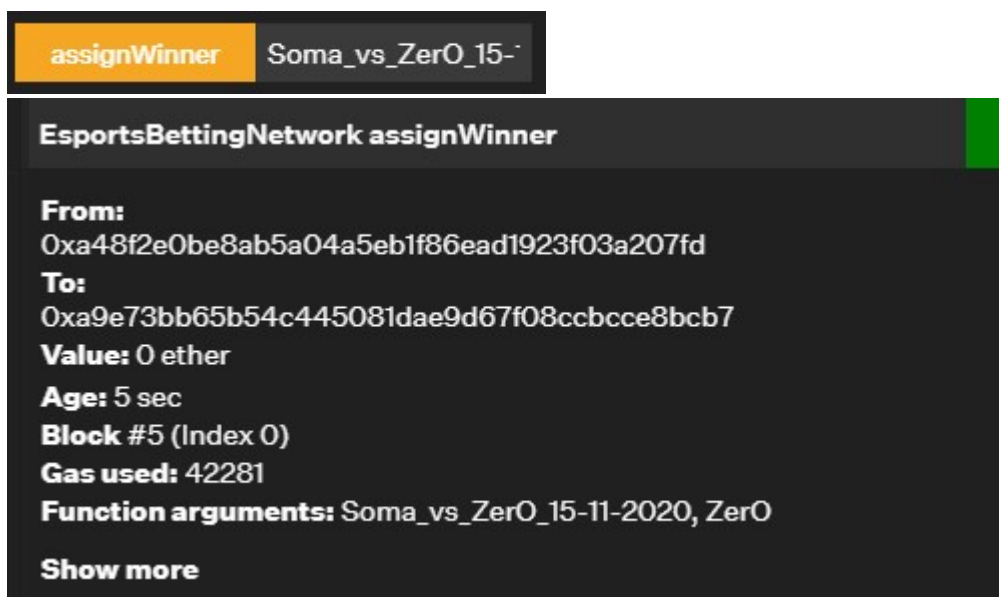


To determine the winner of a match we must have coordinator privileges available.

Step 2: Assign a winner

Enter the following in the assignWinner text field and click the orange button.

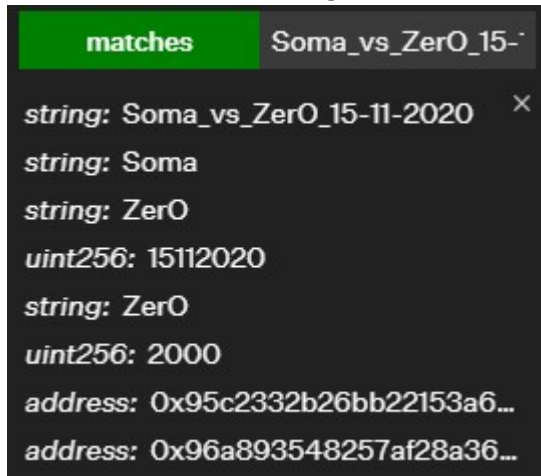
Soma_vs_ZerO_15-11-2020, ZerO



The parameters for the assignWinner function are the match ID, followed by the winning player.

Step 3: Check the winner has been updated

Click the matches button again to refresh and see the winner field has been populated.



Part of the method call for the assignWinner method transfers funds back to the account which bet on the winning player. Therefore, if this method completes without reverting the winner has been paid out.

This concludes the tutorial for the Esports Betting Network prototype, we hope you have enjoyed learning how to interact with the inner workings of the smart contract and we look forward to bringing more tutorials for future features as the project's development continues.

Thank you and have a great day.