Fog Computing for Smart Services

Detailed application design - Peer to Peer Betting App

The application would facilitate the use of DID method and blockchain to allow users to bet in sporting events. Users would have their own DID and link it to a blockchain wallet address or multiple addresses. On the Dapp, users would be able to participate in betting pools where the bets could be placed on the performance of individual players or teams, in addition to the overall outcome of the event. When placing a bet and entering the betting pool for a certain event, users would be connected to a chatroom, where they could chat about the event. Here the DID would come in handy, because users would be able to choose who can see their information and also revoke all their information if they wanted.

Feature description

The application design contains features which are described below:

- The betting would be implemented using smart contracts that would run on the Ethereum network. Smart contracts would create betting pools where users would deposit cryptocurrencies as collateral when entering a bet. That would also enter them into the betting pool chatroom.
- The results of the events used for betting would be obtained using smart oracles (example Chainlink) and smart contracts would use the information to determine the users that won the bet and divide the winnings.
- DID's would be used when communicating with other users in the same betting pool using the chatroom feature. With DID's the app can ensure that only users that entered the betting pool would be able to chat. Users would be able to allow only certain amount of information about themselves be visible and revoke unwanted information. Also, because messages would be encrypted with their DID, user could determine who can see their messages and who can not see their messages.
- The app will have a general-purpose marketplace component that will allow users to create and discover betting opportunities. Users can create new bets by specifying the terms, conditions, and the amount of ETH they want to wager. Other users can discover these bets using various filters and see the betting history of other users and decide whether to participate or not.
- Each user's reputation would be tracked using their unique DID, which can help to promote fair play and prevent fraudulent behavior. Users with a high reputation score can be given priority access to certain betting opportunities or higher reputation could bring higher weights to bets for these users (higher rewards), which can help to incentivize good behavior. Other users could also rate other users which would change reputation score.

Wireframes and diagram

Figure 1 shows a wireframe for the home page of application where a user can browse bets and see their own bet information and join chatrooms. Figure 2 shows a wireframe for placing a bet on the selected sporting event. When a user confirms a bet, they would need to further confirm the transaction with Metamask. Figure 3 shows an example of betting pool group chatroom. Here users can select to chat to a specific user or groups of users. They also can set their DID settings, where they can select what information about them can be seen.

Figure 4 shows the diagram of the application. Data flow (list of APIs):

- Chainlink smart oracle (TheRundown or SportsdataIO nodes) would provide data on sporting events, which would be used by betting pool smart contract to settle bets.
- The betting pool smart contract would create and settle bets for users, send data of user bet results to the reputation system and user lists on bets to the chatroom functionality.
- The user reputation smart contract would track user reputation based on their betting results and other user voting. It would provide reputation data to the betting smart contract and get data from users that voted.
- DID smart contract would implement the digital user ID. Here users can change their DID settings or add additional information to their ID. The chatrooms and reputation smart contract would get DID data from this smart contract based on user permissions.
- The chatrooms would be implemented off-chain and would provide a social aspect of the application. It would gather data from betting pool smart contract to open multiple chatrooms and allow users from those betting pools to chat. It would also get DID data from DID smart contract to show user information.

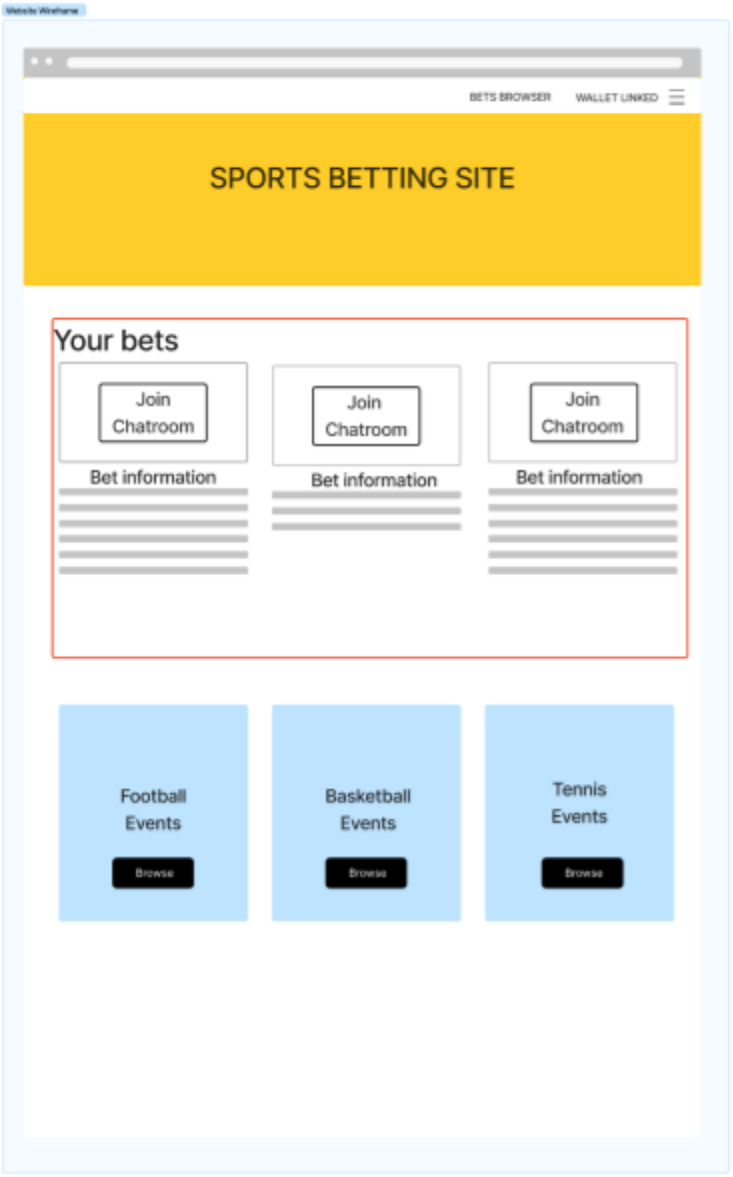

Figure 1: Home page of application

Figure 2: Placing a bet.

Figure 3: Betting pool chatroom.

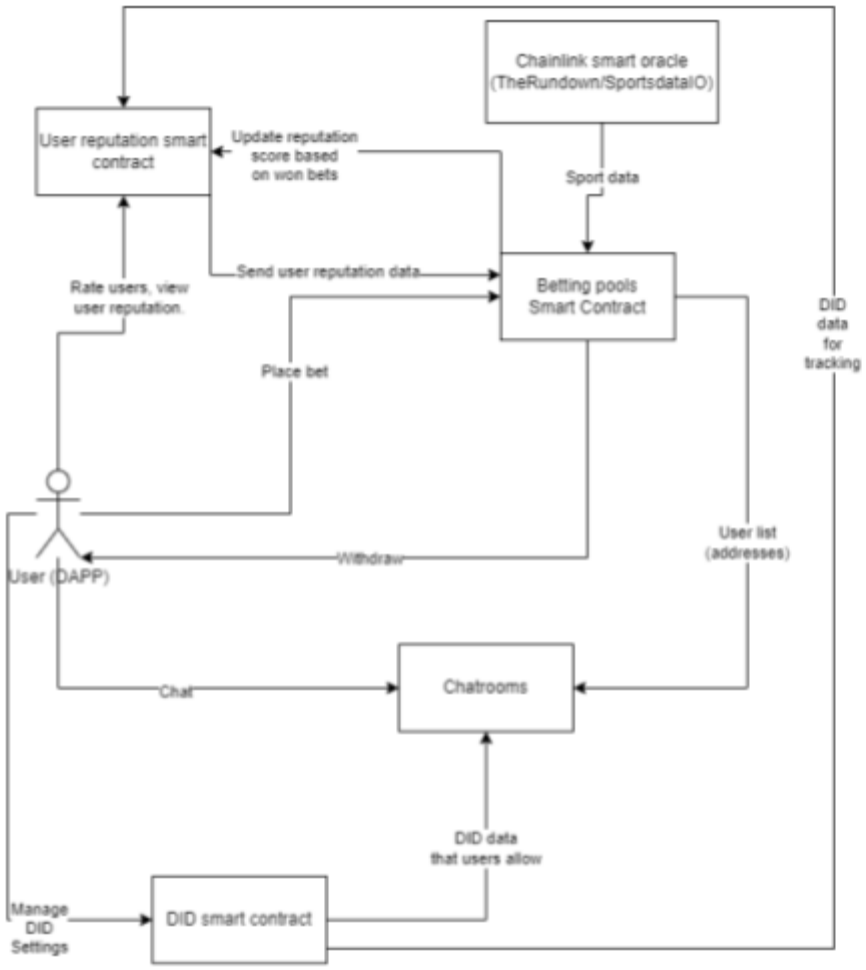

Figure 4: Diagram of the application.

Detailed description of the smart contracts

The following list contains functions which will be implemented for every smart contract:

- **Betting smart contract**
  - createBet: function to create a new betting pool with specified parameters, such as event name, participants, and minimum/maximum bet amount.
  - jointBet: function for users to join a betting pool by placing a bet and locking funds in the contract.
  - resolveBet: function to resolve a betting pool by triggering the smart oracle to obtain the outcome of the event and distributing funds to the winners based on the rules specified in the contract.
  - cancelBet: function to cancel a betting pool if there are not enough participants or if the event is cancelled.

    - getUserList: return list of users for a specific betting pool (list of addresses), to be used for chatrooms.
- **User reputation smart contract**
    - addRating: function for users to rate other participants in a betting pool based on their behavior, such as timely payments or disputes.
    - getRating: function to retrieve the current reputation rating of a user in the system.
    - updateRating: function to update the reputation rating of a user based on new ratings received.
- **DID smart contract**
    - createDID: function to create a new DID for a user, including generating a private/public key pair and storing the public key on the blockchain.
    - revokeDID: function for a user to revoke their DID and associated public key.
    - verifyDID: function to verify the authenticity of a DID for a user in the system.
    - updateDID: function for a user to update (add/delete) their DID data. (the added data could be anything about themselves)
    - getDIDData: function to get the DID data that a specific user made public.

Detailed description of Chatrooms

Chatrooms will be implemented with use of Docker containers. Each chatroom will be one docker container. Docker has good scalability which we could use in certain cases.

List of functions which will be implemented:

- createChatroom: function that will create and set up new docker container resulting in new chatroom
- updateChatroom: function updating chatroom settings
- addUser: function that adds new user (using DID)
- deleteUser: function that deletes existing user (using DID)
- deleteChatroom: function that deletes entire chatroom, also destroys docker container

Frontend:

The frontend of the dapp would be built using modern web technologies such as HTML, CSS, and JavaScript, and would be designed to be responsive and user-friendly. The frontend would interact with the smart contracts on the blockchain through a web3.js library, which would allow for secure communication with the blockchain. Users would sign their transactions using metamask.

The frontend would consist of several key components, including:

- Landing Page: The landing page would be the first page that users see when they visit the dapp. It would provide an overview of the dapp's features and allow users to sign in using their wallet.

- Dashboard: The dashboard would be the main hub of the dapp, where users can view their current bets, check their balances, and access other features such as chatrooms and user settings.
- Betting Interface: The betting interface would allow users to place bets on various events and view the current odds for each bet. The interface would be designed to be intuitive and easy to use, with clear instructions and visual cues to help guide users through the betting process.
- Chatroom hub: The chatroom would be a key feature of the dapp, allowing users to communicate with other users in their betting pool. Users would be able to see all the chatrooms of the betting pools they are in and they would be able to join the chatrooms by clicking on the room.
- User Settings: The user settings page would allow users to manage their account information, update their profile, and configure their privacy settings. This would include options for controlling which data is shared with other users or smart contracts, as well as options for managing their reputation score.

Implementation plan

The implementation of the application will be done in stages. First stage will consist of implementing the smart contracts of the application. The second stage will be implementation of the chatroom component. The last stage will consist of the frontend implementation.

References:

https://github.com/OwnYourData/oydid

https://ownyourdata.github.io/oydid/

https://www.figma.com/

https://market.link/nodes/TheRundown/integrations?searchView=false

https://www.docker.com/