Sportsbetting DApp

Ashrith, Harsha, Eleanor

Sportsbetting

- In 2021 the market for sports betting in the U.S. doubled in size
- Wagering was made legal in several more states in 2021 (including Michigan)
- Decentralization will allow for greater ease in transactions

The recent March Madness tournament, growth in the sports betting industry, and the potential for improvement through decentralizing sports betting inspired us to create our DApp

Solution Overview (Business):

- Smart Contracts allow for controlled transactions without the need for potentially untrustworthy intermediaries (bookmakers)
- Transaction speeds are fast and predictable.
- Reduces the juice from betting (the cut a sportsbook takes from every bet).
- Often bookmakers may ban gamblers who win via controversial methods (steam moves, arbitrage betting). With decentralization any strategy is valid.

Solution Overview (Tech): Structures Used

- 3 Structs to define what the bet stores and what to output and the information needed about the team
- 2 arrays used of the struct types to store user input

```
event NewBet(
    address addy,
    uint amount,
    Team teamBet
);


struct Bet {
    string name;
    address addy;
    uint bankAccount;
    uint teamID;
    uint amount;
    Team teamBet;
}

struct Team {
    string name;
    uint totalBetAmount;
}

Bet[] public bets;
Team[] public teams;

address conOwner;
uint public totalBetMoney = 0;
```

Solution Overview (Tech):

- Individuals bet on the available teams and then
- owner of the contract then declares the winner
- The individuals are then gain or lose money based on their bet

```
function teamWinDistribution(uint _teamId) public  {
    require (msg.sender == conOwner, "only owner can declare winner");


    uint div;
    uint winningTeamCount;

    if (_teamId == 0) {
        for (uint i = 0; i < bets.length; i++) {
            if ((bets[i].teamID) == 0) {
            winningTeamCount++;
            }
        }
        for (uint i = 0; i < bets.length; i++) {
            if ((bets[i].teamID) == 0) {
                div = teams[1].totalBetAmount/winningTeamCount;
                bets[i].bankAccount += div;
            }else{
                bets[i].bankAccount -= bets[i].amount;
            }
        }
    } else {
        for (uint i = 0; i < bets.length; i++) {
            if ((bets[i].teamID) == 1) {
            winningTeamCount++;
            }
        }
        for (uint i = 0; i < bets.length; i++) {
            if ((bets[i].teamID) == 1) {
                div = teams[0].totalBetAmount/winningTeamCount;
                bets[i].bankAccount += div;
            }else{
                bets[i].bankAccount -= bets[i].amount;
            }
        }
    }
```

Demonstration!