BlockWager

The Future of Decentralized Betting is Here

Intro and Overview

BlockWager - Intro and Overview

Who we are

We are a group of professionals passionate about technology, the world of sports, and aiming to fill a gap in the sports betting market.

What we do

Our product is a user-friendly platform that leverages the security and transparency of blockchain and use of smart contracts.

How we do it

We execute on a strategy that user-friendly platform that combines the power of blockchain, smart contracts, and python.

Technologies Used

FinTech

- Odds Scraping
- Gambling Economics

Web3 and Blockchain

- Smart Contracts
- Tokenization

Coding Languages

- Python
- Solidity
- Streamlit/CSS
- SQL

Team

Pravin Patil

Stratis Gavnoudi

Esteban Lopez

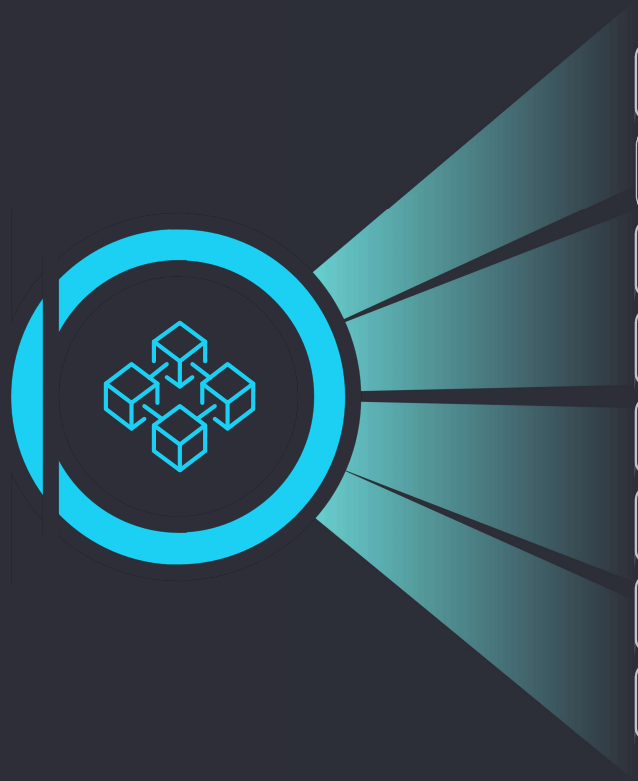
Liset Lop

GitHub: @prpercy/BlockWager

SCAN ME

Live Demo

Live Demo – full list of technologies and packages used



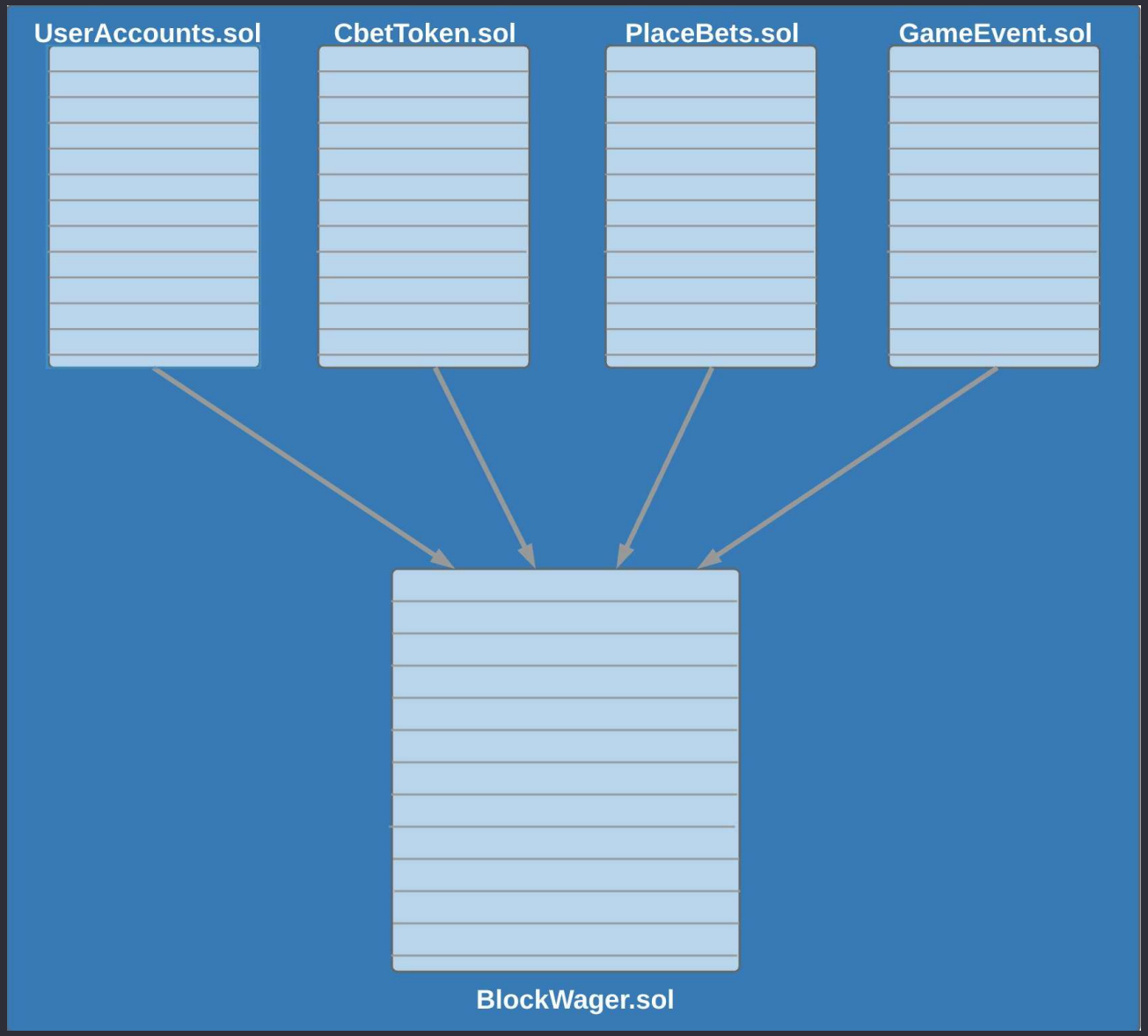
- Languages | Python, CSS, Solidity, SQL
- User Interface | Streamlit, streamlit_elements, colorama, emojis
- Odds-Scraping | sbrscrape, requests, SQLalchemy
- Smart Contracts | Remix
- Blockchain | Ganache, tokenization, MetaMask
- Sports Leagues* | MLB, MLS, NBA, NFL, NHL
- Cryptocurrencies | ETH, CBET**
- Publishing | GitHub

*Certain sports leagues are planned fo

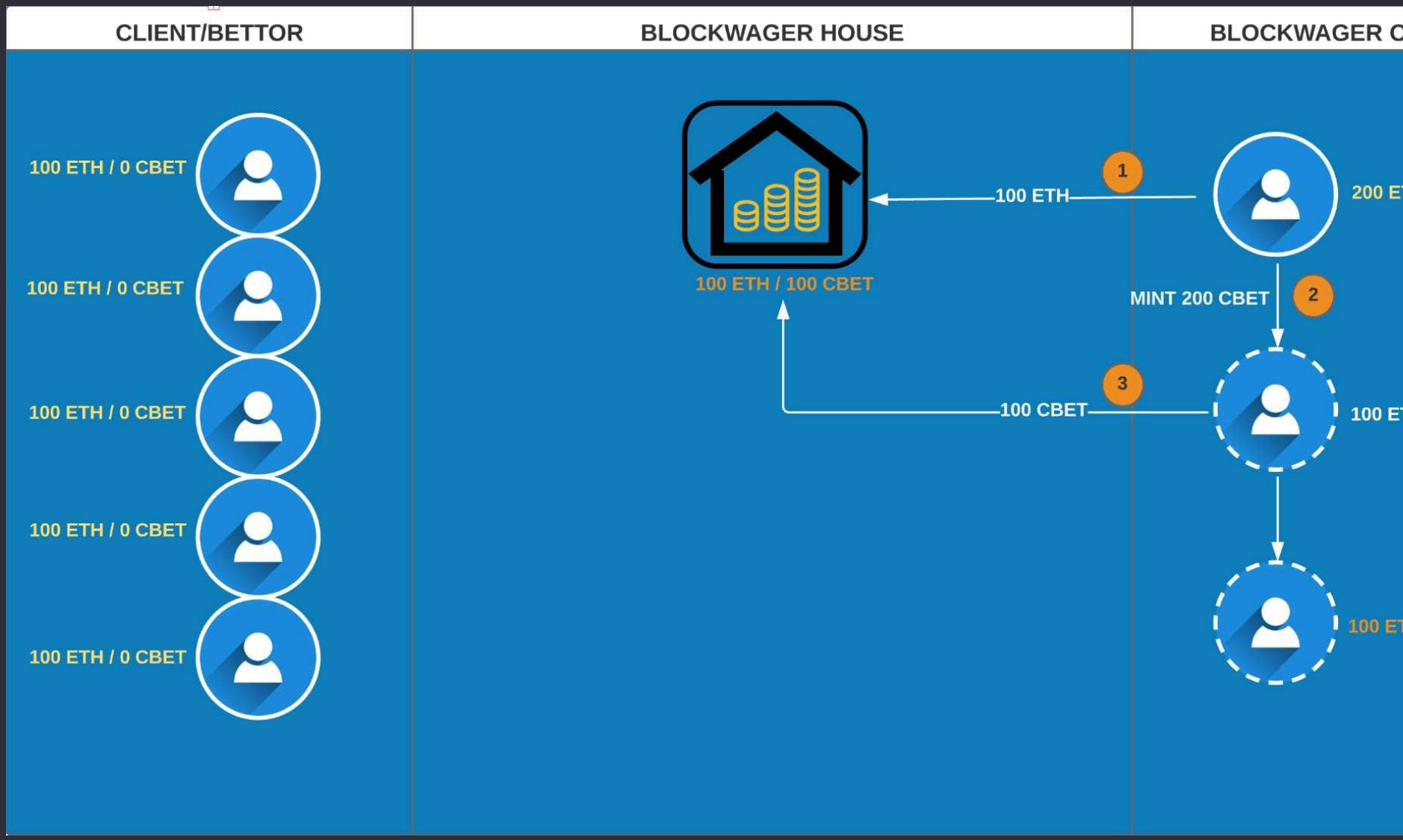
** Internally developed token, not traded in a

BlockWager Smart Contracts

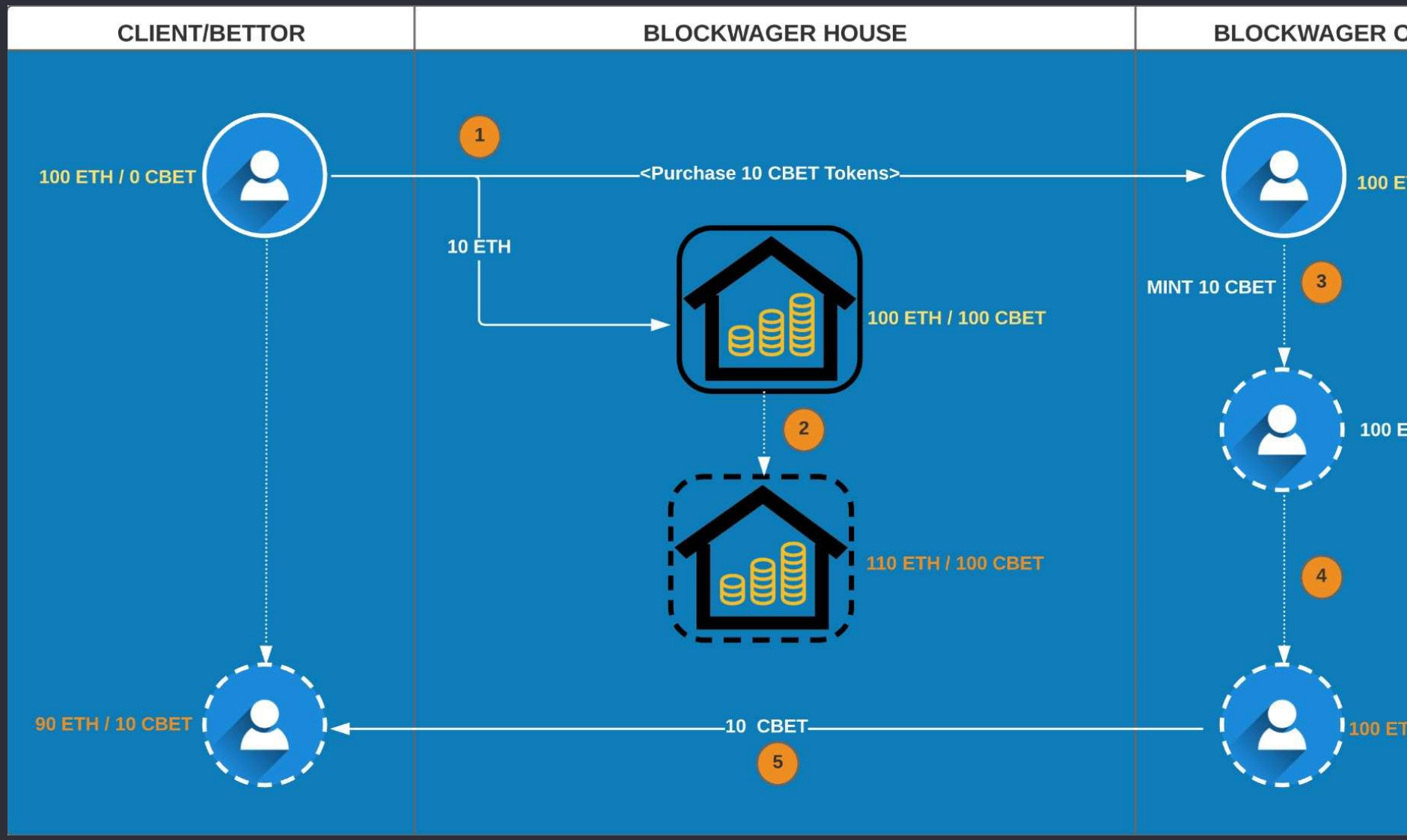
BlockWager Smart Contract – Contract / Solidity Code Structure



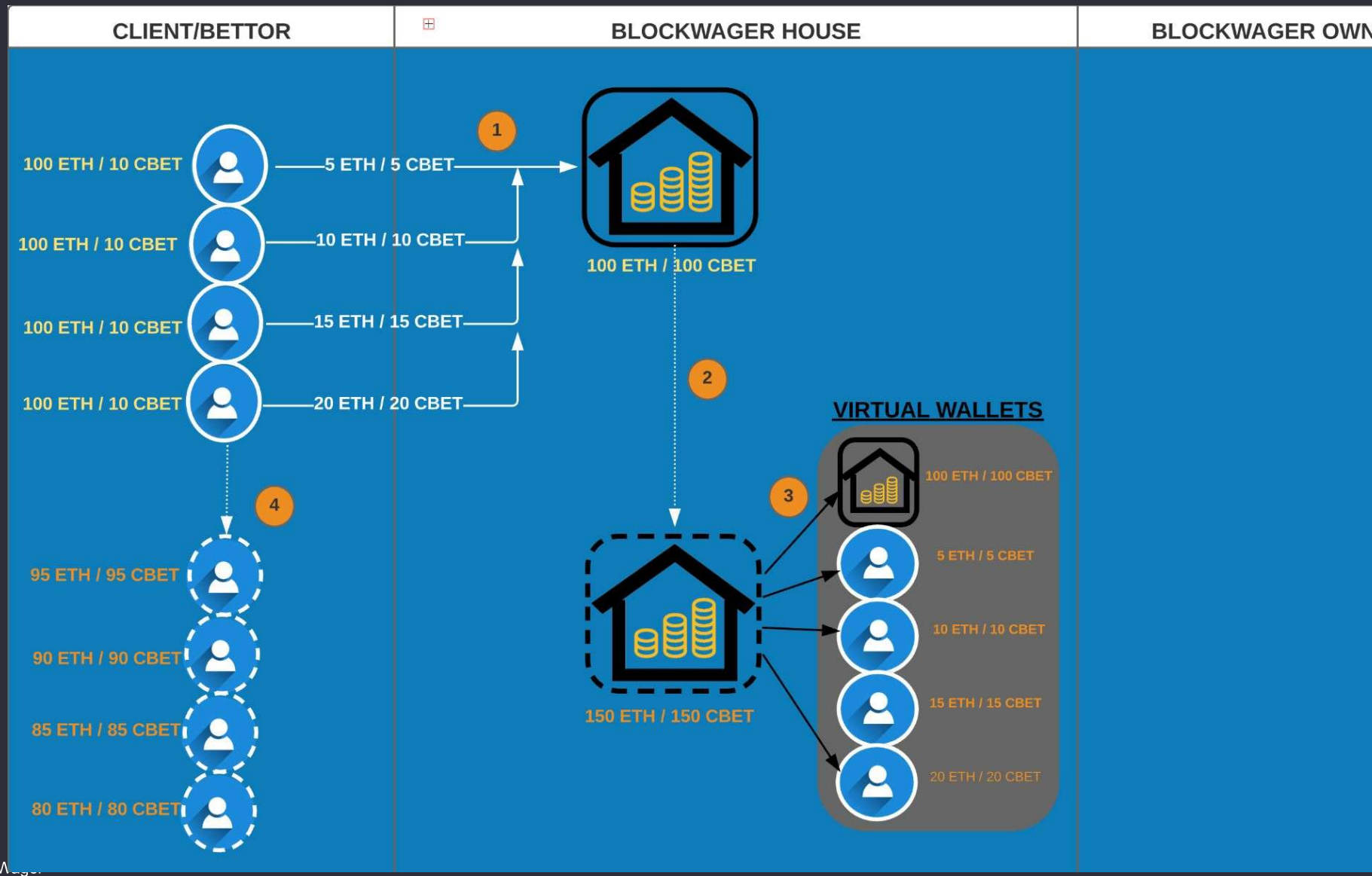
BlockWager Smart Contract – Initial Contract State



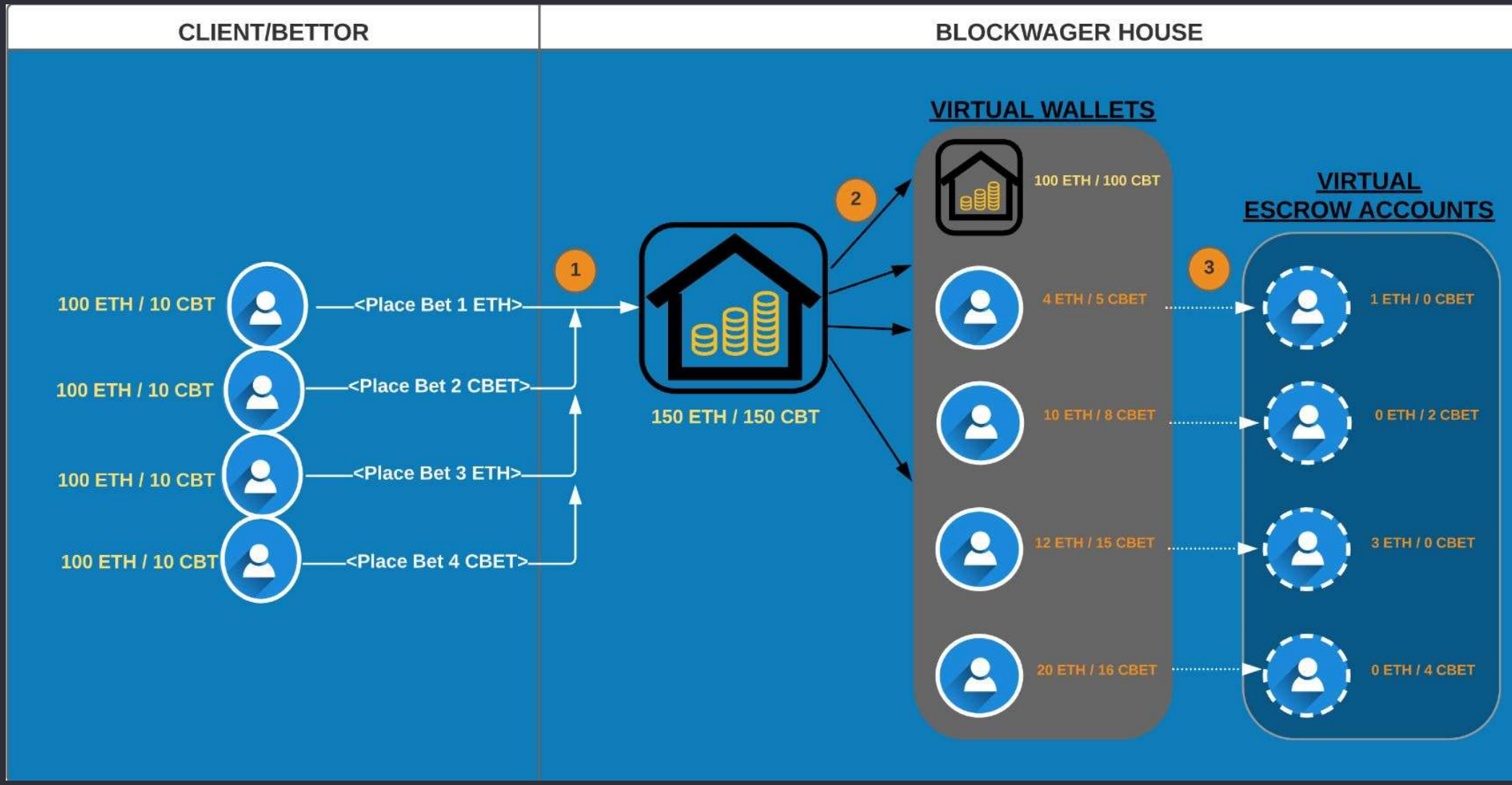
BlockWager Smart Contract – Purchase CBET Tokens



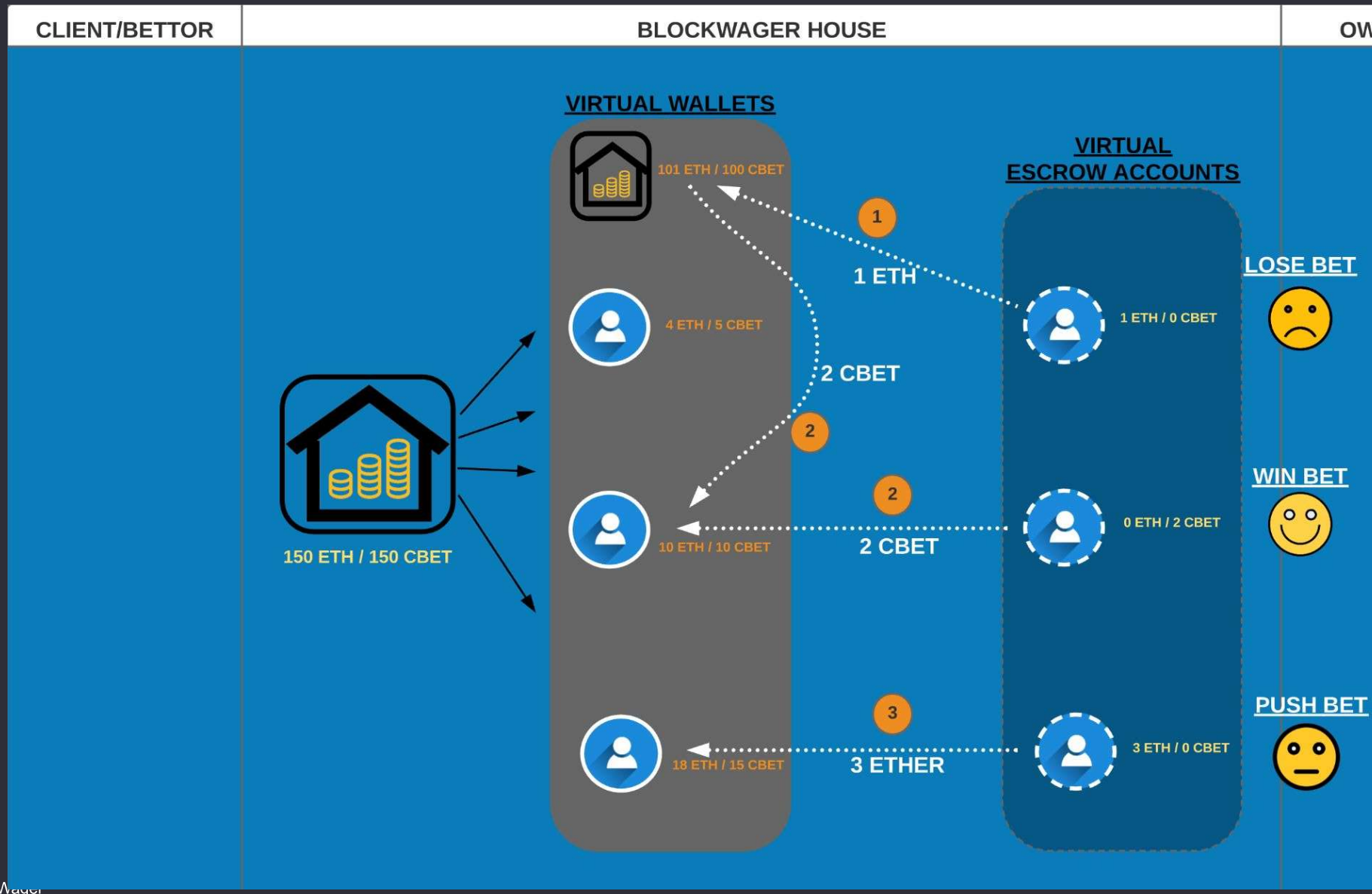
BlockWager Smart Contract – Deposit Ether/Tokens Into Betting Accou



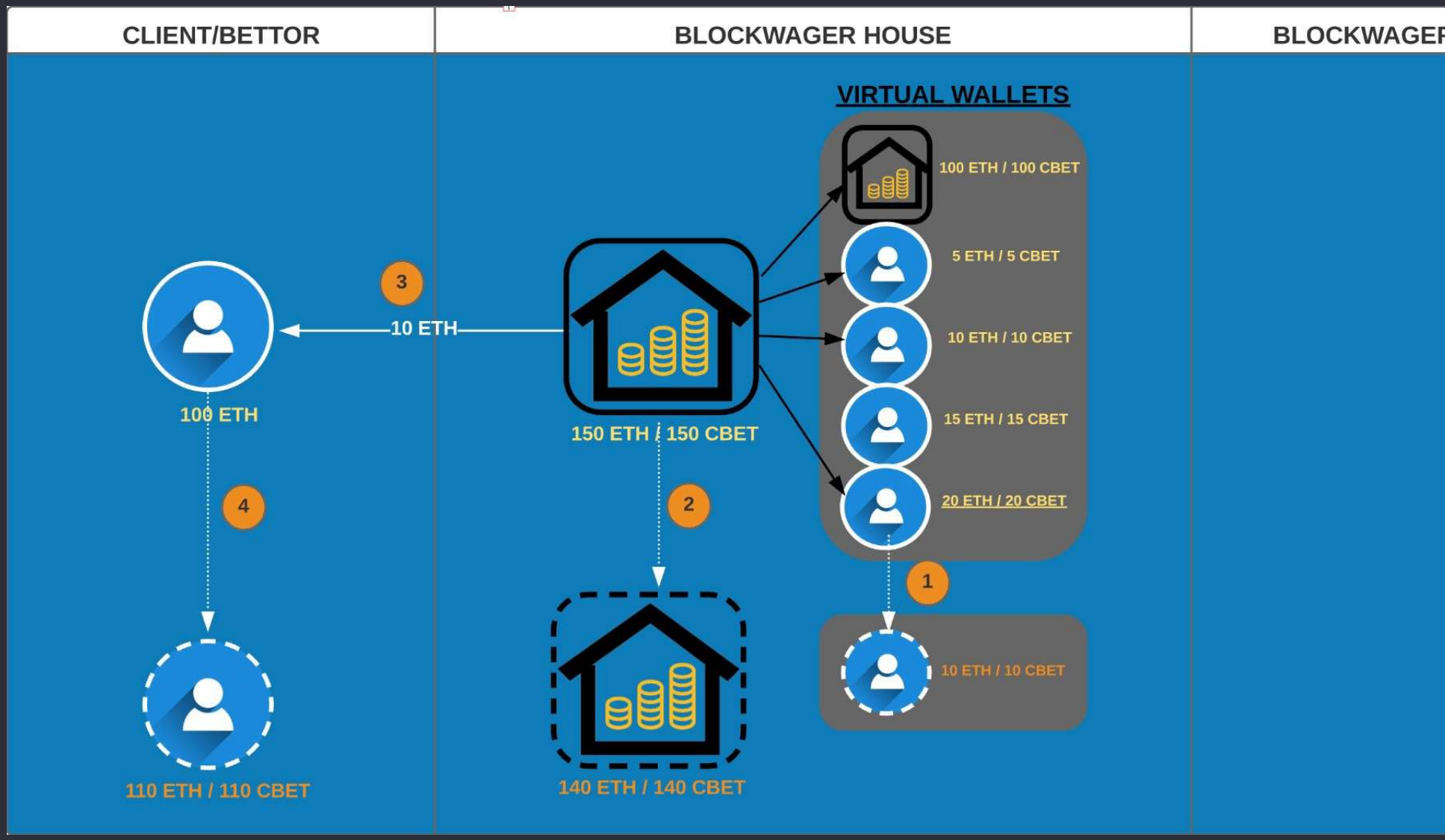
BlockWager Smart Contract – Placing Bets



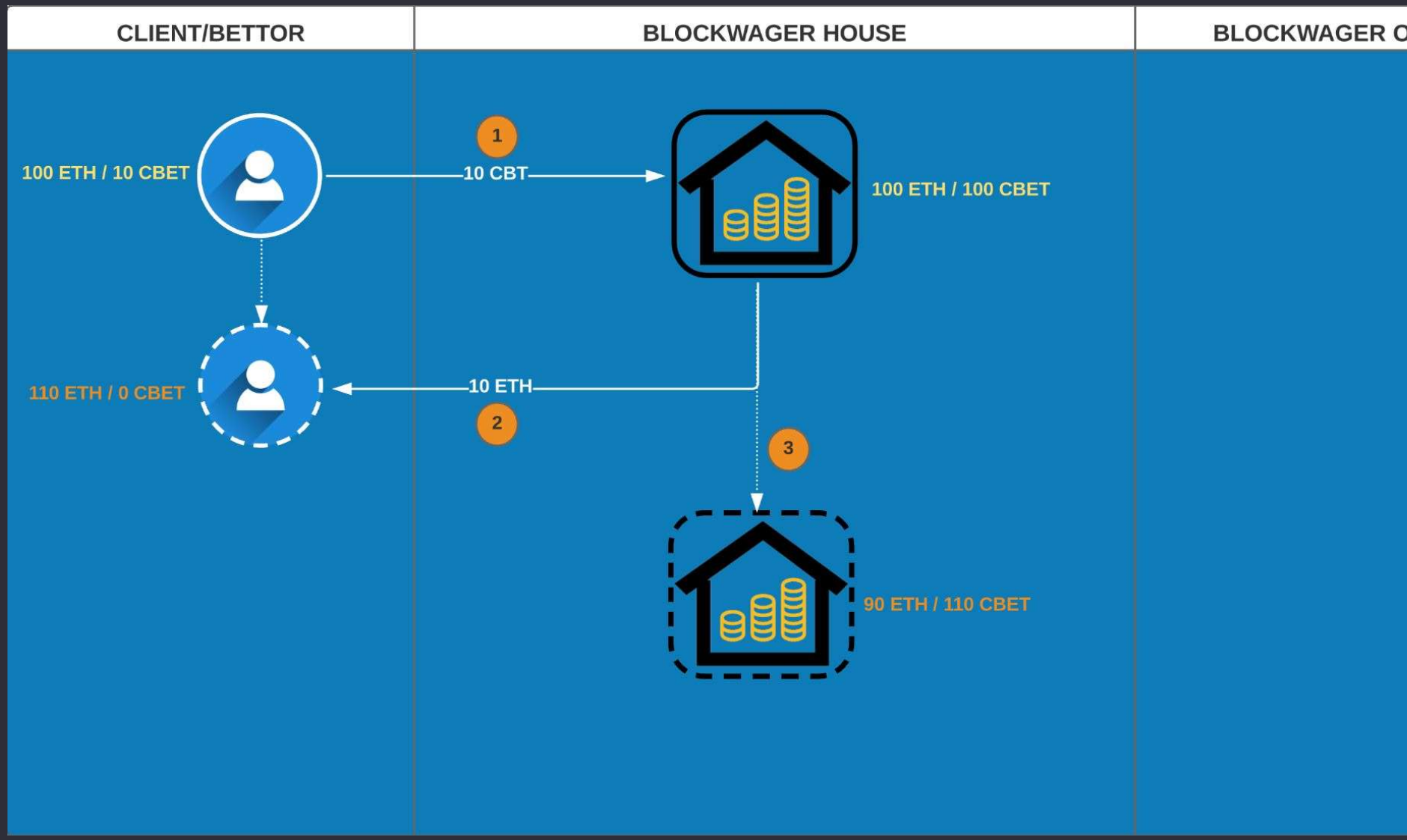
BlockWager Smart Contract – Betting Results



BlockWager Smart Contract – Withdrawal Ether From Betting Account



BlockWager Smart Contract –Sell CBET Tokens



Challenges Faced

BlockWager – Challenges Faced

- Ran into frequent contract size limitations (of 24 KBytes) when trying to deploy our contract to the blockchain. Had make quick adjustments such as:
  - Removing all strings and replacing with ID's (e.g. instead of a team name string such as "Philadelphia Eagles replaced with TeamID='X', and letting the front end application maintain the mapping).
  - Optimizing all data types in the contract from using int/unit --> int8/uint8, int16/uint16, int32/uint32, etc, on a c case basis.
  - In certain scenarios, the contract size limitation prevented us from inheriting standard contract packages (whi grabs all components of the inherited contract regardless if we only needed a subset of the features). In this had to write custom code/functions ourselves.
- No float operations. Operations like multiplication and division required alternate techniques with use of integers
- Much more difficult to debug than your typical software application (e.g. python or C++)
- Testing was time consuming, many permutations of different scenarios and state the application can be in.
- Streamlit has very restricting multi-page features. It was particularly hard to work with smart contract event listener rendering of the page and navigating to one page within the other page. Used html, css hacks to work with UI.

Future Developments and Next Steps

BlockWager – Future Developments and Next Steps

Next Steps

- Peer-to-peer betting
- Contract optimization
- Access to all major US sports leagues and the largest international sport
- Improved quality assurance and internal controls
- Regulatory Compliance
- Cloud deployment

BlockWager

Feeling lucky?