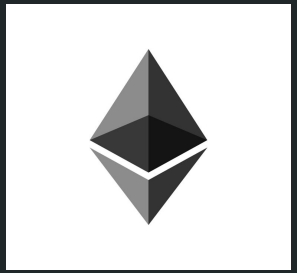
Developing an Ethereum DApp Using React



Jonathan Land (Graduate Project [CPSC 5900])
December 5, 2018
UTC

Presentation Outline:

I. Project relevance and use case
II. Dependencies
    A. Node.js and NPM
    B. Truffle > Ganache
    C. Metamask
    D. IPFS
    E. Solidity, React JS, etc.

III. Demo of project

I. Project Relevance and Use Case

Developing an Ethereum DApp Using React

I. Project Relevance and Use Case

Project relevance:

A. Re-envision how day-to-day transactions take place.
B. Provides quality assurance
C. Speeds up the accessing of important personal documents, bypassing third-party red-tape situations
D. Safer in terms of no single point of failure (i.e. “decentralization!”).

Obviously, utilizing blockchain technology is not a technological panacea for every situation.

I. Project Relevance and Use Case

Use Case:

A. Contractors working on building project need to communicate with customers
B. A private blockchain could be utilized so that customers could sign in to see what is/has been done without having to visit the office or possibly distract workers in progress.
C. A contract could be created in a very short amount of time, migrated to the blockchain, and the work process officially begin
D. A record is kept if there are any questions about timeline or what was done and by whom.
E. Everyone has up-to-date information.

II. Dependencies

Developing Blockchain Dapps

II. A. Node.js and NPM

II. A. Node.js and NPM

1. Node.js is a JS runtime environment that provides the capability to execute anything and everything related to JS and its frameworks (e.g., Angular, React, etc.). Without Node.js we could not run a standalone JS application.
2. Encapsulated within Node.js is its package management ecosystem, **NPM**.

   NPM is a massive open source collection of open source libraries that can be used for any kind of project and/or programming idea.

II. B. Truffle > Ganache

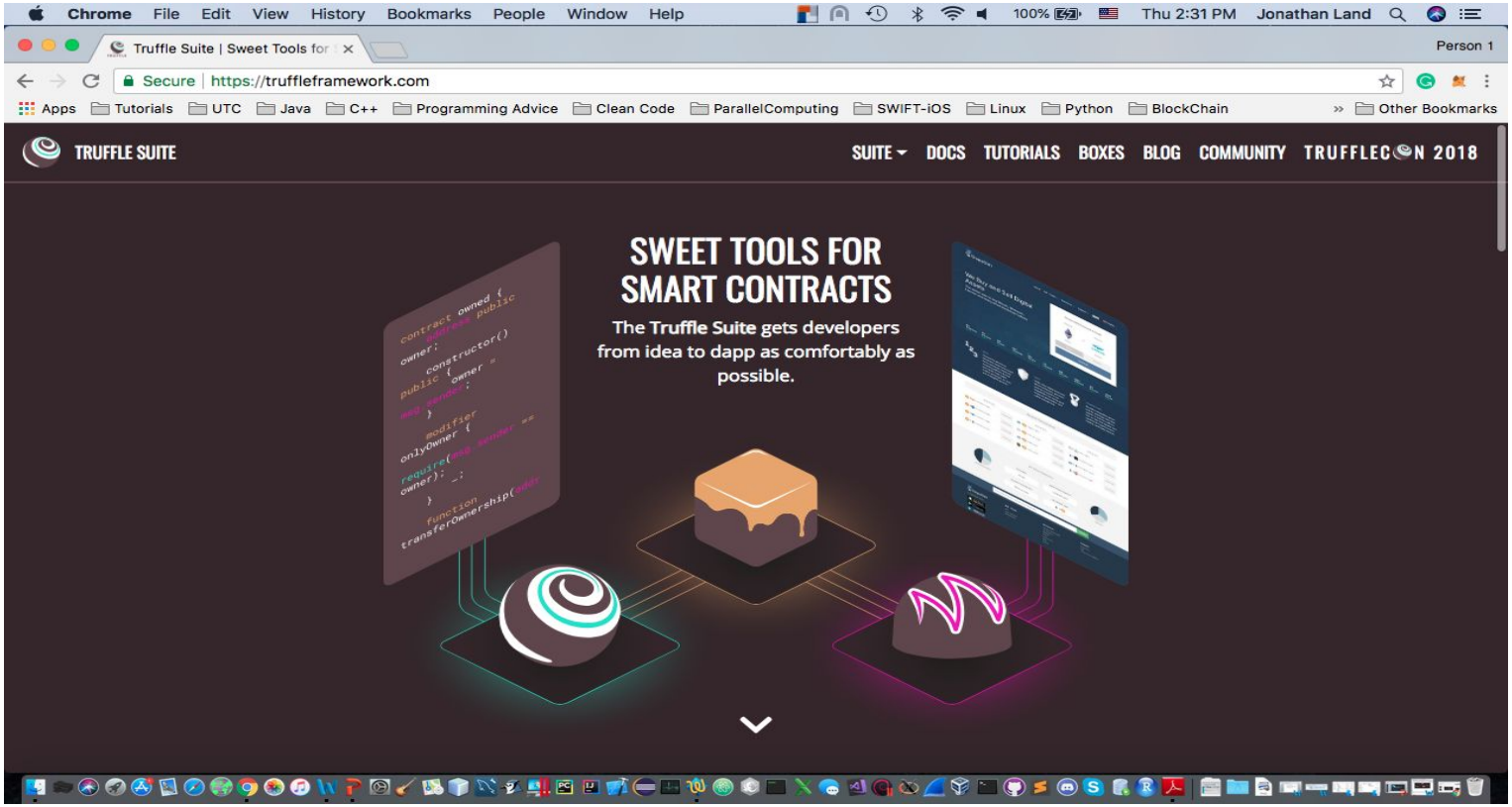
II. B. Truffle > Ganache

Truffle is a development environment that provides library guides and tools for creating, testing, and publishing smart contracts using the Ethereum Virtual Machine (EVM) (supports JS, JSX [used with React for templating, language extension to JS])

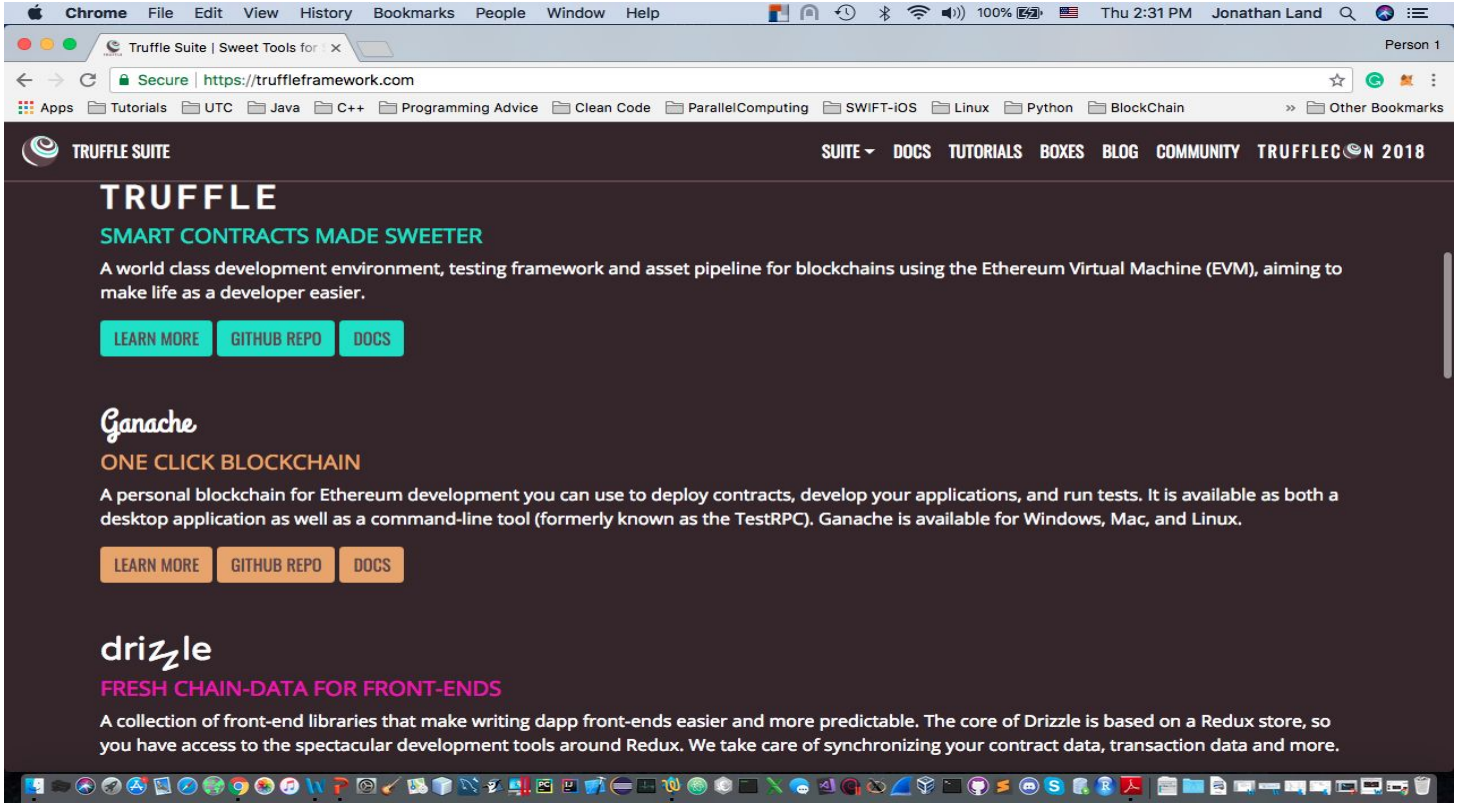
- **Built-in smart contract compilation, linking, deployment and binary management.**
- Automated contract testing for rapid development.
- Scriptable, extensible deployment & migrations framework.
- Network management for deploying to any number of public & private networks.
- **Package management with EthPM (package registry for Ethereum) & NPM (the best!), using the ERC190 standard.**
- Interactive console for direct contract communication.
- Configurable build pipeline with support for tight integration.
- External script runner that executes scripts within a Truffle environment.
- **Provides migrations, which are JS files that help to deploy smart contracts to Ethereum network.**

https://github.com/trufflesuite/truffle

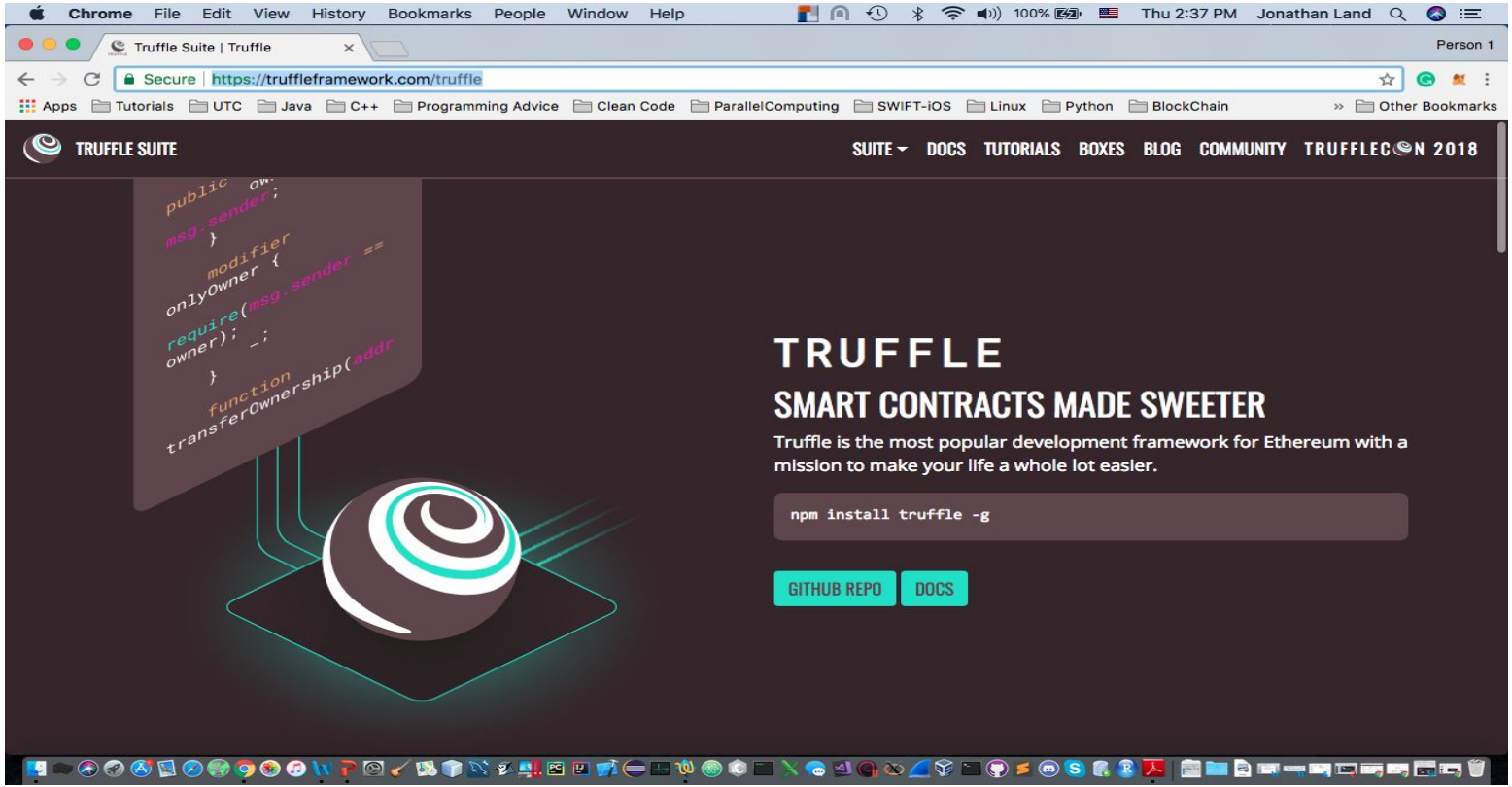
II. B. Truffle > Ganache



II. B. Truffle > Ganache

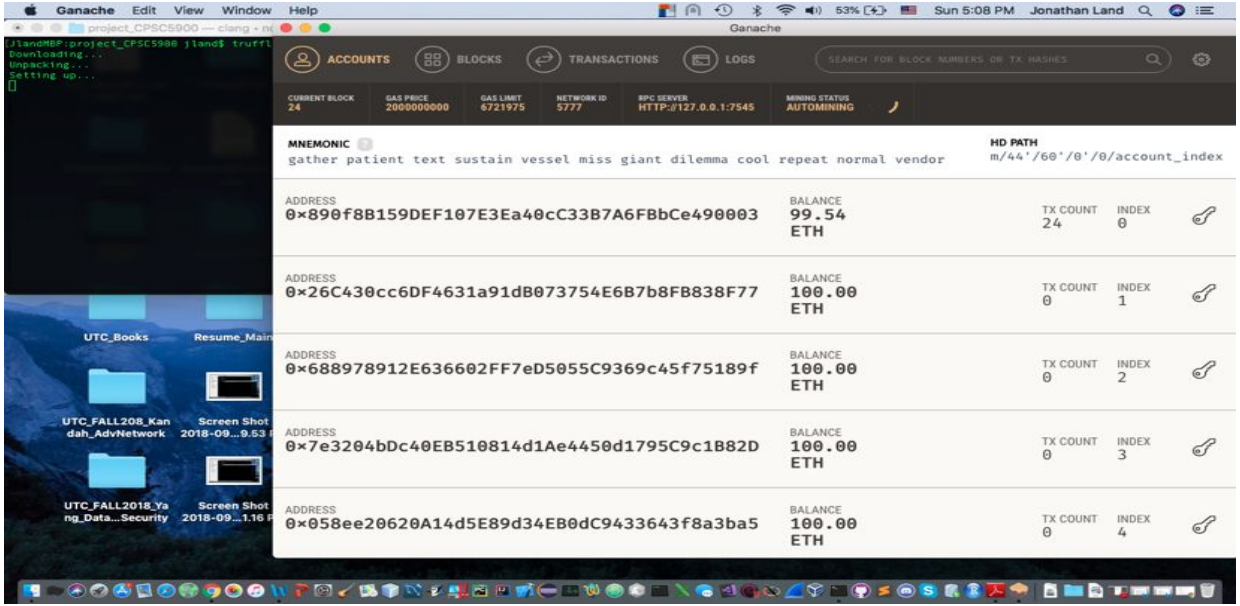


II. B. Truffle > Ganache



II. B. Truffle > Ganache

Ganache is a part of the Truffle suite (Truffle, Ganache, Drizzle). This creates a virtual blockchain for Ethereum development. After downloading, Ganache will create a private blockchain with ten free accounts and some ether. `npm install -g ganache-cli`



II. C. Metamask

II. C. Metamask

Metamask is a chrome browser plug-in that allows Dapps to be accessed on a normal web-browser. Without this, web-browsers do not know how to interact with the blockchain. The plugin injects a JS library (web3js) that provides methods for normal websites to use in order to complete read/write requests to Ethereum blockchain.

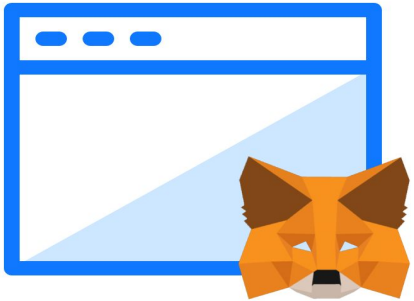
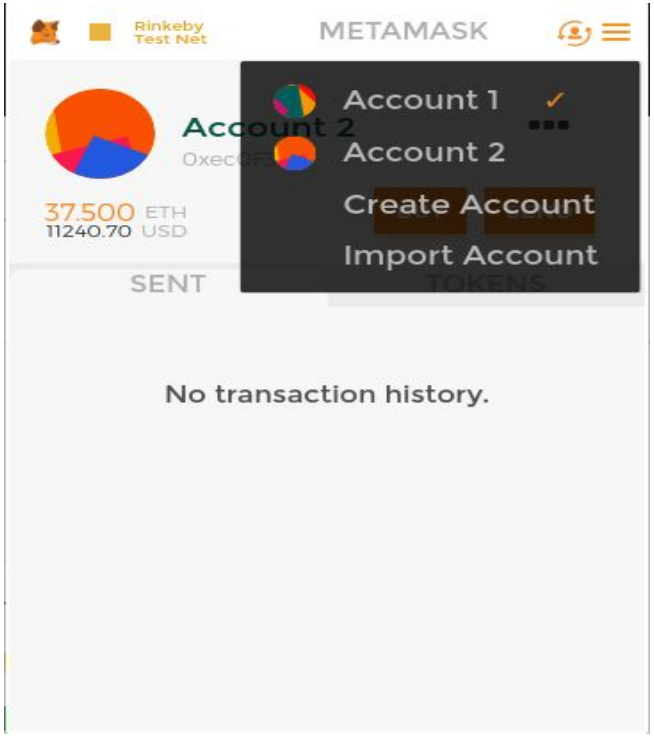
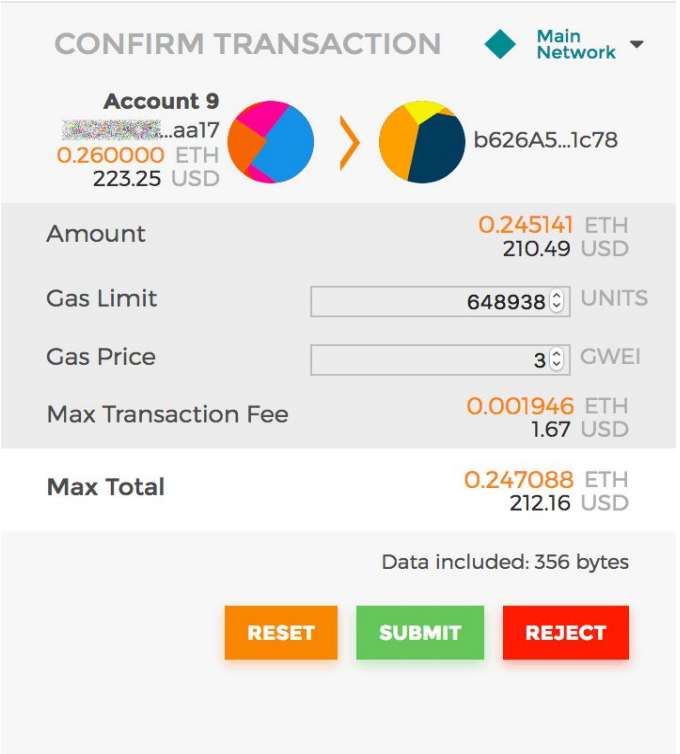


Image: https://blockonomi.com/metamask-guide/

II. C. Metamask



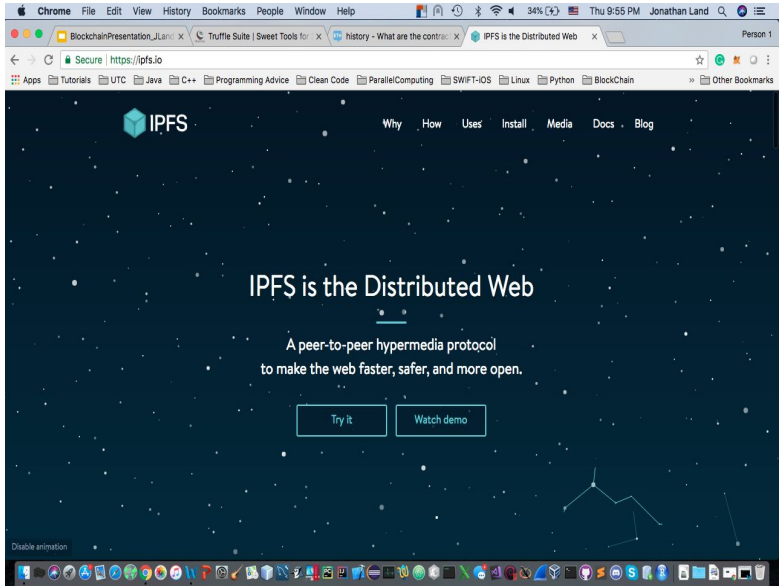


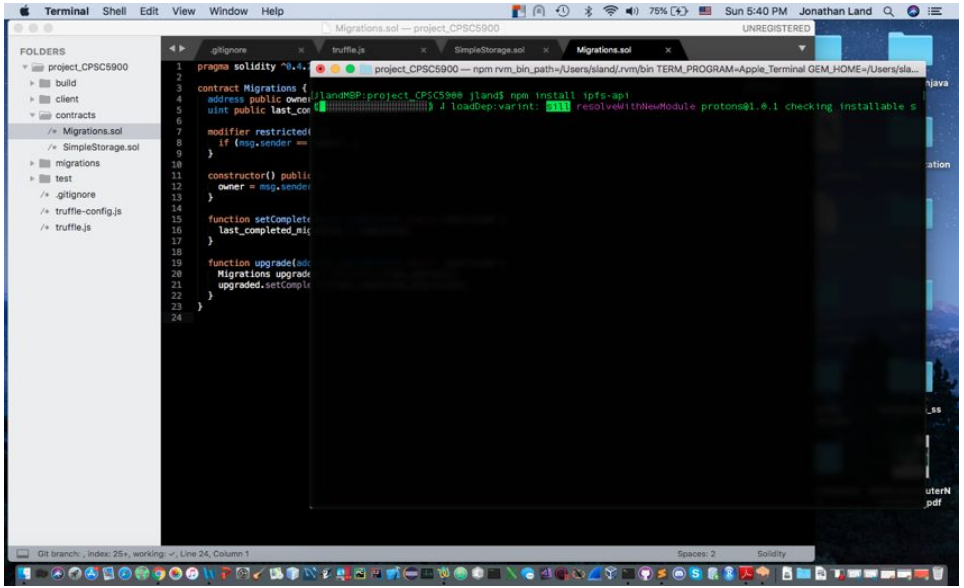
Images: https://steemit.com/ethereum/@tomshwom

II. D. IPFS

II. D. IPFS

IPFS (Interplanetary File System) functions a lot like Git (store files and track different versions/additions over time) and BitTorrent. It defines how files move across a network, making IPFS a distributed file system (uses distributed hash table [DHT]). Once we have a hash, we ask the peer network who has the content located at that hash and then download the content directly from the node that has the data.





II. E. Solidity

II. E. Solidity and React

Ethereum smart contracts are written in a programming language. One of the most popular languages for creating smart contracts is Solidity (basically like JS). Solidity is often used in conjunction with ReactJS, which is an open-source JavaScript library that is used for building user interfaces specifically for single page applications. React uses a special syntax called JSX which allows you to integrate HTML with JavaScript.

```
pragma solidity ^0.4.21;
contract Coin {
    // The keyword "public" makes those variables
    // readable from outside.
    address public minter;
    mapping (address => uint) public balances;

    // Events allow light clients to react on
    // changes efficiently.
    event Sent(address from, address to, uint amount);

    // This is the constructor whose code is
    // run only when the contract is created.
    function Coin() public {
        minter = msg.sender;
    }

    function mint(address receiver, uint amount) public {
        if (msg.sender != minter) return;
        balances[receiver] += amount;
    }

    function send(address receiver, uint amount) public {
        if (balances[msg.sender] < amount) return;
        balances[msg.sender] -= amount;
        balances[receiver] += amount;
        emit Sent(msg.sender, receiver, amount);
    }
```

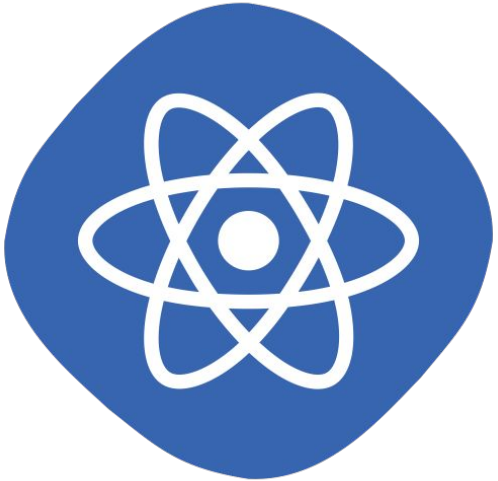


Image: https://marketplace.visualstudio.com/items?itemName=beaugunderson.solidity-extended

II. Demo...