PROJECT 3:WORKING WITH ETHEREUM BLOCKCHAIN

Authors: Sai Ram Prasanth Tatiraju
Sandeep Kothapally

Abstract

Real estate industry has traditionally been very slow to innovate. Real estate title varies county by county, with many localities still utilizing traditional paper records and filing cabinets rather than digital alternatives. Typically, real contracts are physically printed, read, amended and signed by all parties before anything can be finalized.

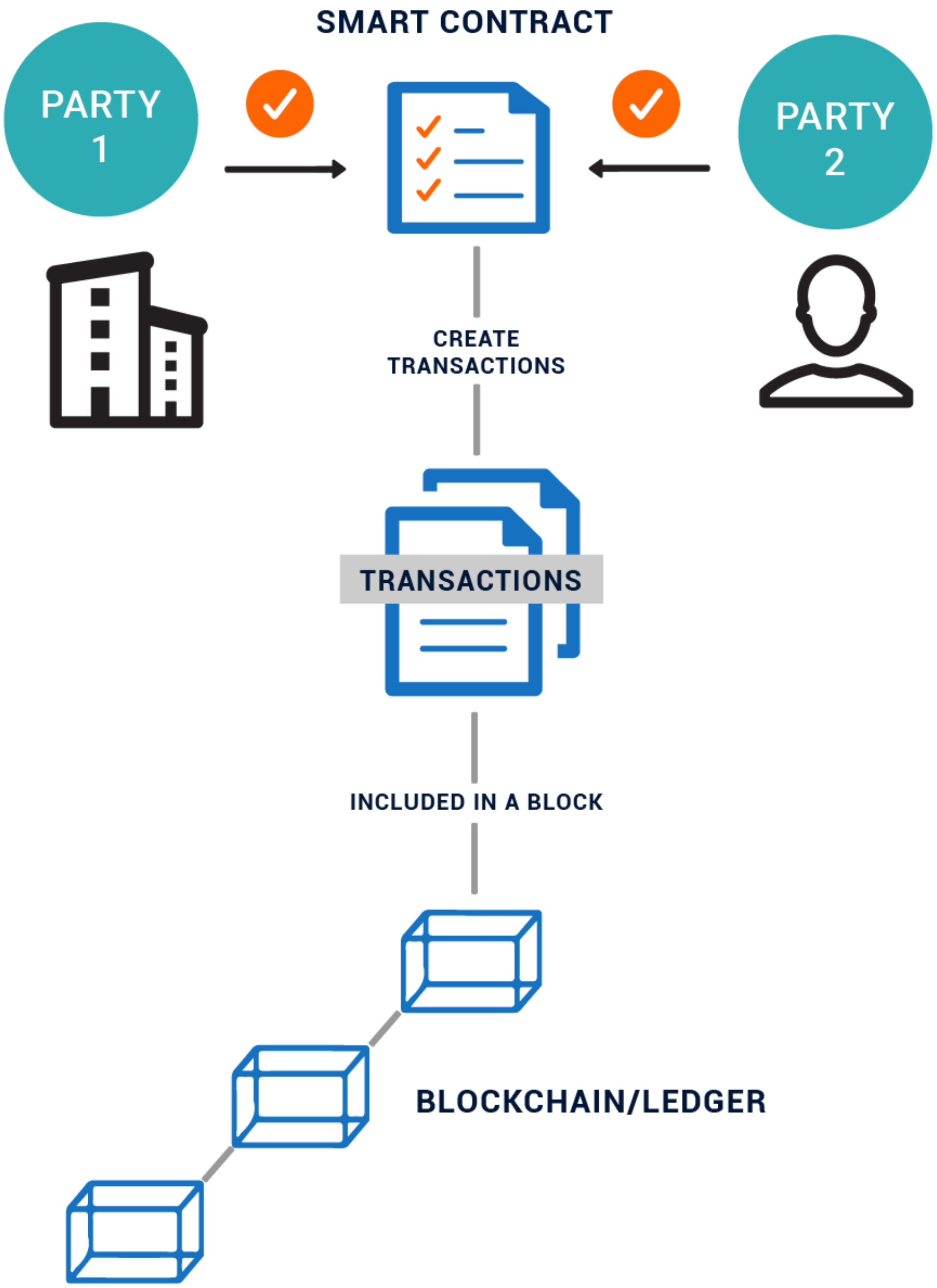
Consider the purchase of a residential property. One must account for state and county documents, mortgage agreements, title and deed papers, insurance policies, homeowners association forms, and more. This bundle of paperwork often totals some 100 pages for a single transaction.

The sheer amount of documents required in real estate transactions can easily lead to fraud or mistakes. Any mistake from one set of documents can lead to a cascade of problems in associated documents. There are often ownership disputes in the real estate industry from improper recordkeeping and if the building in which important documents are stored burns down or is broken into, valuable information could be forever lost.

Blockchain and smart contracts can revolutionize the real estate industry by creating a unique digital address for documents and events related to every property, including repairs, permits and ownership history.

The benefits smart contracts can offer are:
1)Enhance Speed of transactions
2)Protection against fraud
3)Significant reductions in real estate costs that lead to affordable investing
4)Cutting off the middle men

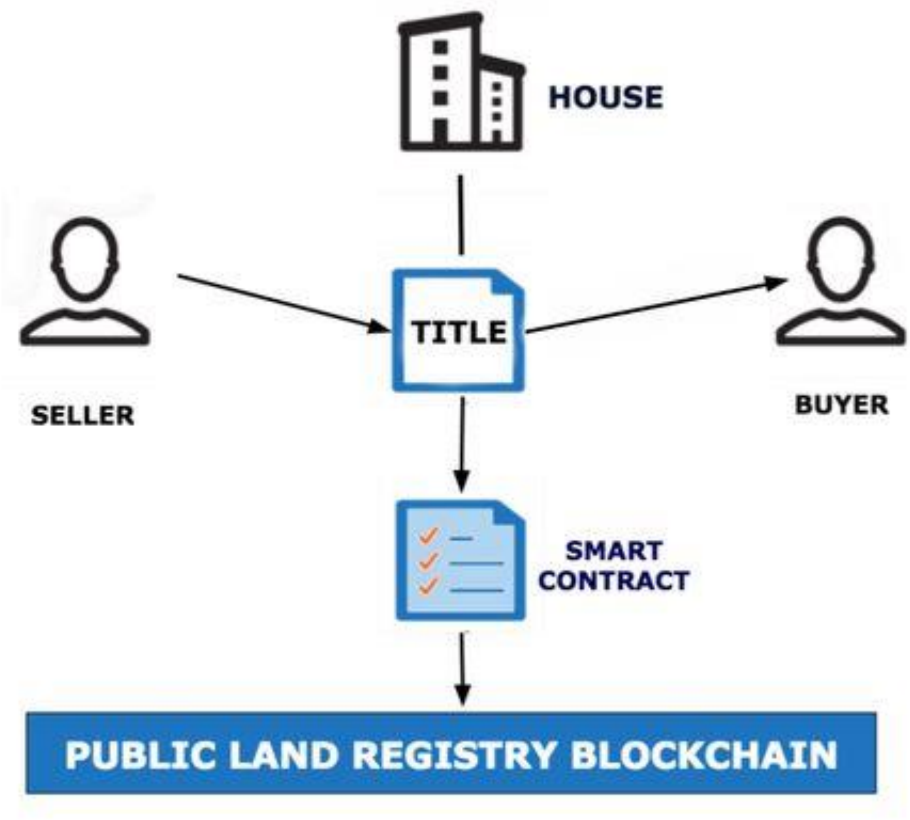
BLOCKCHAIN AND SMART CONTRACTS - FLOW DIAGRAM



Party

Attributes

- LandBalance
- WalletBalance

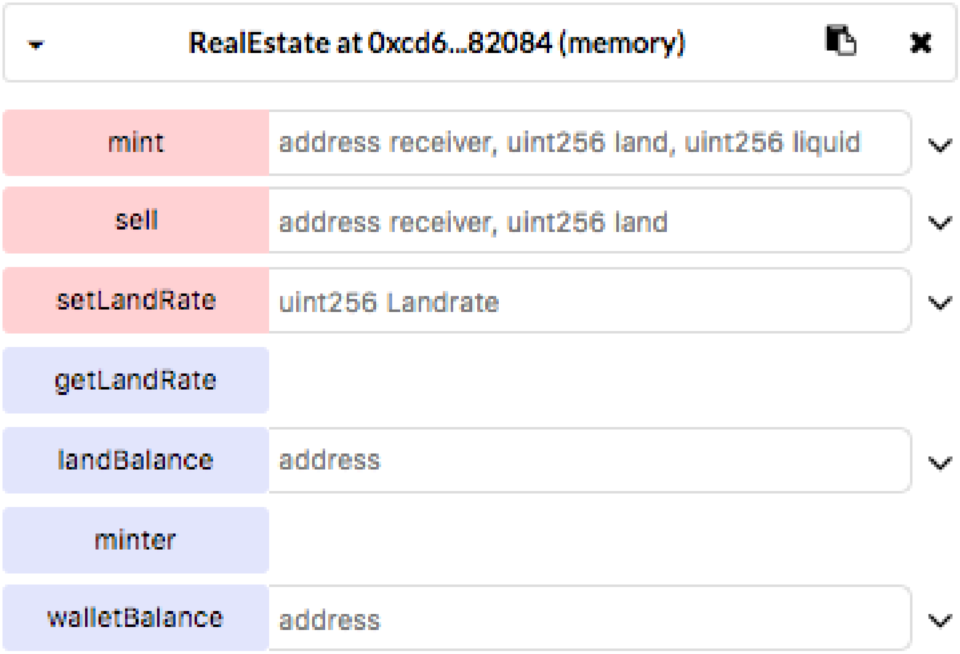


Working code of the Property Registration application

```
pragma solidity ^0.4.25;

contract RealEstate {
   // The keyword "public" makes those variables
   // readable from outside.
    uint storedData;

   address public minter;
   mapping (address => uint) public landBalance;
   mapping (address => uint) public walletBalance;
```

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

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

    function setLandRate(uint Landrate) public {
        storedData = Landrate;
    }

    function getLandRate() constant public returns (uint)  {
        return storedData;
    }
    function mint(address receiver, uint land, uint liquid) public  {
        if (msg.sender != minter) return;
        landBalance[receiver] += land;
        walletBalance[receiver]+= liquid;
    }

    function sell(address receiver, uint land) public {
        if (landBalance[msg.sender] < land) return;
        landBalance[msg.sender] -= land;
        walletBalance[msg.sender] += land*storedData;
        landBalance[receiver] += land;
        walletBalance[receiver] -= land*storedData;
        emit Sent(msg.sender, receiver, land);
    }
}
```

Remix UI after Deploying the contract

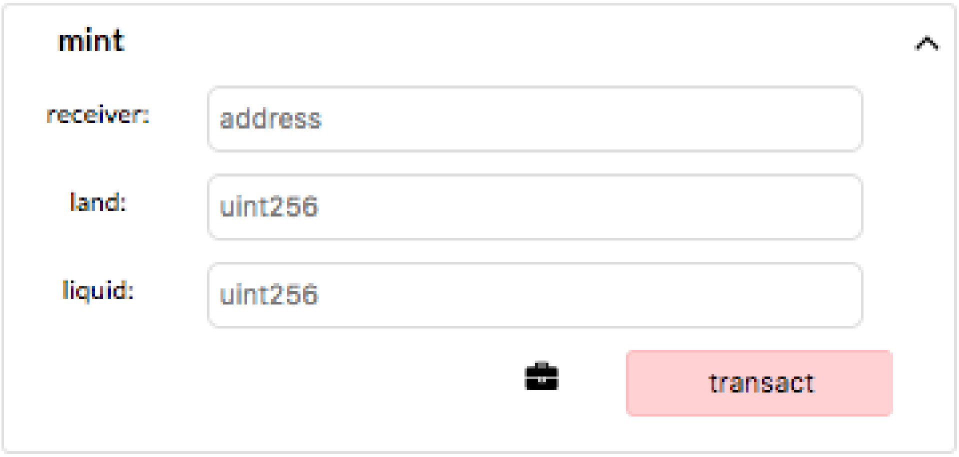

RealEstate at 0xcd6...82084 (memory)
mint
address receiver, uint256 land, uint256 liquid
sell
address receiver, uint256 land
setLandRate
uint256 Landrate
getLandRate
landBalance
address
minter
walletBalance
address

Our Implementation has the following methods

- Mint
- Sell
- setLandRate
- getLandRate
- landBalance
- walletBalance

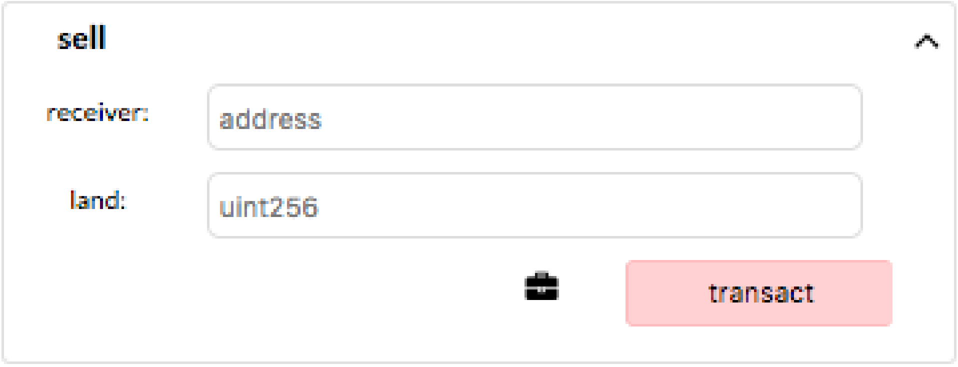
Mint:

Mint would create a new user (party) and allocate him some land and liquid cash. While creating we'll put the receiver's address, land and liquid cash in their respective input boxes.



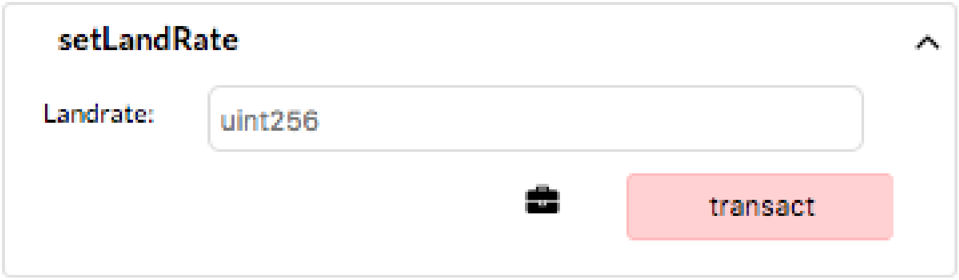
Sell:

Sell option is used by an user to sell his land to some other user. After transacting, the land his both land balance and walletbalance(Liquid cash) would automatically gets updated.



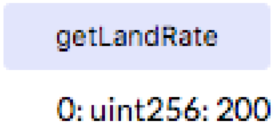
setLandRate:

The setLandRate option is used to define the Land rates at any point of time. We'll input the desired land rate and transact. By doing so the Land rate gets updated.



getLandRate:

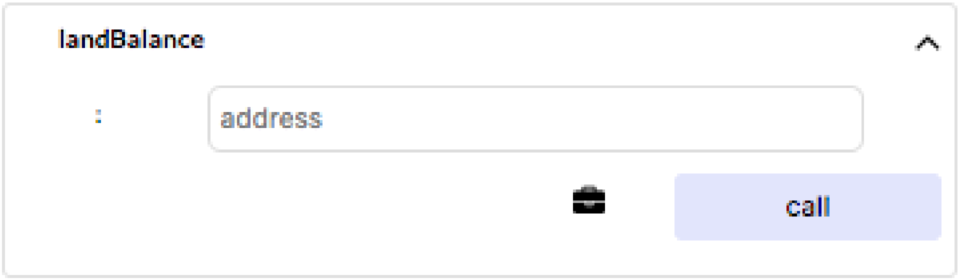
The getLandRate option is used to fetch the existing market rate of land.



```
0: uint256: 200
```

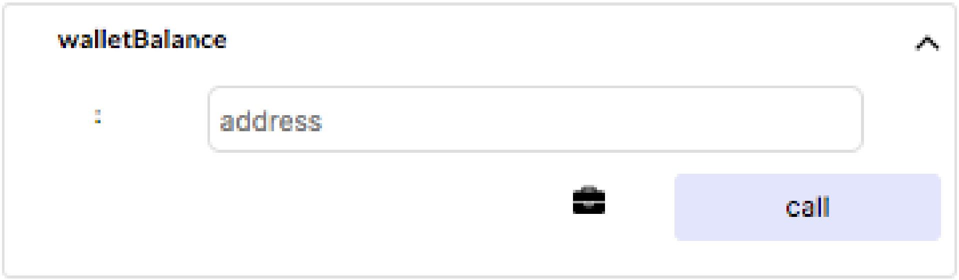
landBalance:

The landBalance option is used to check the amount of land any user has.



walletBalance:

The walletBalance option is used to check the amount of liquid cash any user has.



Conclusion:

Thus the Block chain Technology application for Property Registration would help efficiently to Enhance Speed of transactions and Protection against fraud in a robust way.