Blockchain based Land Registry and Land Transformation system using Ethereum Blockchain

[1]Dr. Prasanna B T, [2]Shobha G

[1]Associate Professor, [2]Student Department of CS&E
[1]Department of Computer Science and Engineering,
[1]Sri Jayachamarajendra college of Engineering, Mysuru, India.

Abstract: Land Registry is a simple Dapp based on the ethereum blockchain. It can be used as an alternative to the existing approach. Here the land owner registers the land details along with the land value by providing necessary proofs. Only a registrar or government authority who is registered as the superadmin can do the registration process. Lands coming under a particular area (eg. a village) can register to the system only through the superadmin assigned to that area. The smart contract has been written in such a way that the owner has to transfer his full asset to the buyer and no partial transaction of the asset is allowed. Even though the registration process requires a government authority, the entire process is transparent and the transaction happens between the two clients without any intermediaries.

IndexTerms **- Ethereum, Blockchain, Land Registration.**

I. INTRODUCTION

For land, being a high-valued asset, it is very important to have accurate records which identify the current owner and provide the proof that he is indeed an owner. These records can be used to:

a) Protect owner's rights
b) Prevent sale frauds
c) Resolve disputes
d) make sure ownership is correctly transferred to a new ownership

Thus, it is crucial to maintain correctness and completeness of this information, and prevent unauthorized, fraudulent changes.[5][8][12]

Currently people rely on third party, i.e., government agencies that are responsible for keeping track of ownership information. This third party keep all the records in the centralized database. Hence to transfer the ownership, it becomes difficult and slow to first find verify the land and then transfer the ownership.[16]

It is possible to keep track of the property ownership if we have a distributed system which stores all te land history and share it among the interested buyers. This would remove the intermediaries. And seller can directly contact the buyer. Thereby removing the extra cost and time that is needed to be spent on the intermediaries.

At minimum a blockchain based ledger is needed that stores the transactions done in the process of land ownership transfer. This problem is solved by Satoshi Nakamoto in his paper about bitcoins when he created[1][4]:

a) Storing the information in a blockchain,
b) For correctness protocol rules can be used,
c) For identifying the owner public key cryptography can be used.

Ethereum is a free open source platform which helps developers to build and deploy decentralized applications such as smart contracts and other complicated legal and financial applications [15]. Ethereum is kind of a programmable Bitcoin where developers can use the underlying blockchain to create markets, shared ledgers, digital organizations, and other endless solutions application to a problem that need immutable data and agreements, all without the need for a moderator or realtor. Released in 2015, Ethereum is the brainchild of the prodigious Vitalik Buterin, who saw the potential uses of Bitcoin's underlying blockchain technology as the next steps in speeding the expansion of the blockchain community. Ethereum is now currently the cryptocurrency with the second highest coin market cap and is expected by some to surpass Bitcoin as both a valued investment and as the world's most popular cryptocurrency. [2][9]

Hence Ethereum is best suited for creating a ledger that stores transactions during the land ownership transfer process. The aim is to create a ledger along with some smart contracts that will triggers the various events that are going to happen on the system during the process of ownership transfer.[10][11][17]

The roles in the system are:

- Buyer: Buyers needs to register himself by providing the documents issued by government. Then he can see the land which is available.
- Seller: seller needs to register himself as a seller, he will upload photos of the land, along with the documents of the land. Moreover, he needs to pin the land on the map.
- Land inspector: An official from land registration government agency, he inspects the documents once any seller approves the request of buyer to buy the land.

1.1 Blockchain

The concept of Blockchain technology can be divided into three different concepts: a. An organisational concept: blockchain technology is aimed to cut costs and to make the use of Trusted Third Parties, such as Notaries, Banks and Governmenatl organizations superfluous. It is meant to give individuals control on the processes in the blockchain, without the use of a man-in-the-middle. In well-functioning Land Registry systems there are reliable Trusted Third Parties involved in the transfer of ownership of immovable property. Because of the design of the process of transfer of ownership in these systems, the legal certainty is guaranteed and rightful claimants are protected. Apart from the registration of transactions, there is also a relationship between the transaction and the reality, the actual situation. It is questionable whether Blockchain technology can perform these elements as well or replace a well-functioning Land Registry, especially in cases where Land Registries are well-functioning and trusted by its users. In countries where there are no such (reliable) registries, the use of Blockchain technology perhaps seems more appropriate. This is why the use of blockchain is examined, amongst other countries, in Honduras6 and Ghana.

b. A concept of design: the creation of a reliable and accessible and/or public administration containing all kinds of transactions using the capabilities of the network organisation (just like the structure of the Internet, a network without a single point of failure). Blockchain technology offers a complete new perspective on how to keep a registration and to make information accessible (in a registration). This approach fits the self-reliance and it is interesting to take into account when designing new registrations, although there are not many Blockchain-based applications, especially in Land Registration matters, and little is known about potential drawbacks of this concept. In the Netherlands, Dutch Kadaster is doing some research on the use of Blockchain in cases of sharing specific data sets, concerning open data. If the technique seems fit for this purpose, these data sets will be put on blockchain, so everybody using these open data sets can see the dataset is put on the blockchain by (and therefore derived from the cadastral and land registry information from) Kadaster. c. A technological concept: a technical solution to situations where multiple parties can perform transactions. There is a need for a decentralized solution that ensures reliability and consistency of information. There are several implementations of blockchain that may offer useful functionality. These applications possibly can provide opportunities to simplify current. The possibility to use Blockchain for Land Registers It is clear that the Blockchain technology is (probably) fit for various purposes as mentioned in the previous paragraph.

The functionality of Blockchain can be described as a digital ledger. It serves the same functionalities as a sound Land Registry system: it knows who owns what at a certain time, it ensures single-ownership and it knows when a certain transaction took place. It is possible to 'track back' and therefore it should be possible to guarantee title. Compared with a 'classic land registration system', blockchain may even provide some additional certainty. Because of the shared databases there is security of back-ups. Trust is added by cryptographic proof and a decentralised database, especially in the case the current administrator (Registrar) is not trusted. It might safe costs because of remediation of intermediaries (Notaries or licensed conveyancers) or administrators (Registrars). It therefore can be judged as an alternative for the classical Land Registers. Because of its transaction dependency, in the Blockchain, it is not possible for a non-owner to transfer ownership. Checks on ownership using Blockchain technology are processed automatically, using transaction dependency and transaction rules, whereas in current Land Registry systems checks on ownership are executed by the Registrar, mostly by scrutinizing the deed and comparing this information to the content of the land register in person. That means that in the majority of cases the data of the seller mentioned in the deed is compared in person to the data of the current owner in the land register. One of the exceptions36 to this manual process is the computerized processing of deeds by using stylesheets, where the data with regard to the seller that is mentioned in the deed is automatically compared with the current owner as mentioned in the Land Register. The only possibility for a non-owner to transfer ownership, using the Blockchain technology, could be the case where someone other than the owner uses the private key of the owner and uploads a transaction. However, there is a possibility for the owner to use a back-up system for his/her private pin-code in case of a crash of the used device (smartphone, tablet or computer). The wallet can be accessed by using a back-up that has previously been installed, although this back-up is executed by an intermediate. The often proclaimed risk that hackers can steal bitcoins, depends more on weaknesses with company security than the underlying core protocol itself. This in fact is no different than the DigiNotar-case as described in part I of this paper.

Furthermore the register is public and not to be changed since recording of the data is time-stamped and therefore indisputable. In current Land Registry systems this is applied by using time-stamps and audit trails. In case of the Land Registry system in the Netherlands, the moment the deed is received by the Registrar is decisive.

The deed will be registered, using the time the deed was received by the Registrar (priority). In theory it can happen that two deeds with regard to the same immovable property are received by the Registrar at the same time. In case this would happen, according to Dutch legislation, the time of execution of the deed is decisive. This time of execution is mentioned in the deed. In theory it could happen that two Notaries executed a deed with regard to the same object at the same time and sent is to the Registrar at the same time. Using the blockchain, one might think this is not possible. But in reality these situations happen quite a lot: a temporary situation of a so-called fork can occur in the Blockchain

II. Literature Review

Ethereum is an open source and decentralized network. The structure of Ethereum is more or less the same as the structure of Bitcoin. Ethereum is a Blockchain based distributed software platform which allows users to build and deploy decentralized applications. Ethereum is a distributed public block chain network focusing on running any decentralized application's programming code. The Ethereum network has its own crypto-currency 'heart.' Ethereum blockchain's Merkle Patricia tree structure [5] can also be emulated as IPFS artefacts. For larger pieces of data to be stored on the Ethereum blockchain, a larger amount of fee has to be paid, so only the hashes of files are stored on the Ethereum blockchain rather than storing the whole file on it. Further, this hash of the file can be linked with the file on the IPFS to access it. [6] A smart contract is used to verify the performance of the contract. The smart contract consists of rules in it which needs to be satisfied in order to successfully run a smart contract. Smart contracts enable valid transactions to be carried out without third parties. Smart contracts provide convenient access to the Ethereum blockchain. Solidity is a programming language which is used to write smart contracts. Ethereum smart contracts are written in a coding language of high level called solidity, inspired by coding languages such as C++, javascript and Python. Remix is used for building smart Ethereum contracts. Another is the Truffle architecture that supports streamlined compilation, linking, deployment and binary management of smart contracts. It facilitates ecosystem for implementation of both public and private networks. It is a framework to deploy smart contracts. AES (Advanced Encryption Standard) is an iterative algorithm and not Feistel algorithms. AES is very commonly used in cyber security to encrypt and decrypt messages. It involves a series of linked operations, some involving inputs being replaced by various outputs, while others involve shuffling bits around. In the case of AES the block size is 128 bits or 16 characters which means 16 characters can be encrypted at a time.

Lemieux argues that Blockchain is a new technology with the potential to radically change the record of land and real estate transactions. The author highlights Blockchain-based land registration practices developed by the requirements in Brazil, Georgia, Honduras, Ghana, India, Japan, and Sweden. Pilot applications in the regions determined within these countries are explained and it is predicted that the number of full-time uses will be increased [3].

The contribution of various scholars is studied for survey and analysing the merits and demerits in order to enhance the consequences for making the system work better. Vinay Thakur et al [1] proposed adopting blockchain for managing the records pertaining to land in India. It points out the various problems like unaccountability, less transparency and incongruous data sets with the various departments of the government related to the land record management. It also points out the tremendous delay in the existing system. He proposed the use of Blockchain for land titling, to provide right of ownership and make it fool proof.

Ingo Weber et al [2] proposed an architecture for multi-tenant blockchain based system to assure data integrity while preserving data privacy and performance isolation. Though there are difficulties in constructing the multi-tenant blockchain based architecture considering data and performance isolation. Firstly, the data of one tenant must be not available for another tenant to read, also that tenants having higher workload must not affect the read and write operation of the other tenants. Secondly the architecture must be scalable for each tenant and also with the number of tenants. This architecture maintains individual architecture for each tenant's data and smart contracts.

U.M.Ramya et al [10] proposed a system to reduce forgery in land registry using blockchain technology. The system makes use of a private permissioned blockchain called Multichain for land registration. Multichain provides a simple API, command line interface and supports Linux, Windows and Mac servers. As the system uses a private

permissioned blockchain it is not necessary to use an algorithm such as Proof-of-Work to show that transactions are taking place between nodes and new blocks are added to the chain. A drawback is that the technology is still developing and it's effectuation is not cheap. Also a minor change on the original document would change the hash which makes verification tedious.

Aravind Ramachandran et al [6] suggested that blockchain can be used to promote the collection of data provenance and verify them accordingly. The proposed system incorporates smart contracts and open provenance model (OPM) to record data transactions that are deemed as immutable. The privacy protection of the proposed system for the provenance data is accomplished by hashing and encryption. A user accepted by the owner can only see the changes made to an inferred document ID by accessing the event log. The major drawback faced here is the owner is able to give access to users who may misuse their access privilege and they might pose a threat to the credibility of the proposed system.

Shuai Wang et al [7] devised a Blockchain-Enabled Smart Contracts architecture, where smart contracts are pacts facilitate the negotiation and implementation of digital contracts without the need of a third party user. One of the key benefits of smart contracts is that the code of smart contracts is recorded on the blockchain making them immutable. But the proposed architecture faces a severe drawback, in the presence of irreversible bugs present in the smart contract that contains a bug which cannot be changed.

III. METHODOLOGY

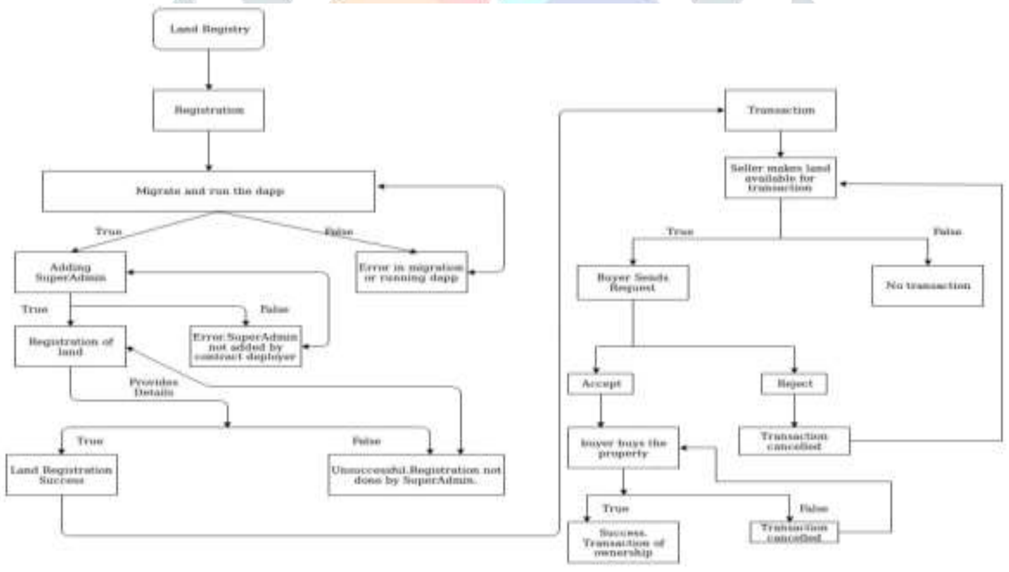


Fig 1: Code Flow Diagram

FUNCTIONS() :

1. addSuperAdmin() : Used for adding the super admin. This function can only be called by the owner of contract (person who deployed the contract). Address (ethereum) of super admin and the village to which the super admin is assigned for is provided as input. Each super admin's address is mapped to the corresponding village.

2. Registration() : Used for registering a land. Details of land such as state, district, village and survey number along with the owner's ethereum address, market value of the land and property ID is provided as the input. The details are registered mapped to the property ID. This function can only be called by the super admins and the registration of a land can only be done by the super admin of the village to which the land belongs to.

3. landInfoOwner() : Function to view the details of land for the owner. Input is property ID and returns state, district, village, survey number, availability to buy, address of the person who requested the land to buy, request status(accept, reject or pending) as the output.

4. landInfoUser() : Function to view the details of land for any user. Input is property ID and output is the current owner of the property, market value, availability of the land to buy, whether there are any requester for the land(address of requester) and request status.

5. computeId() : Used to compute a unique ID for a property. State, district, village and survey number is provided as input and a unique ID is returned as the output.

6. requestToLandOwner() : Used to send a request to buy a property. Property ID is provided as the input and requester's address is sent to the owner.

7. viewAssets() : Function to view the assets of a person. When the function is called, the asset list of the person who called the function is returned.

8. viewRequest() : Function to view the address of the requester for a particular land, if there are any. Input is the property ID and the output is the address of the requester.

9. processRequest() : Used to process the request came to buy the land. This function can only be used by the owner of the land. Property ID and request status(accept or reject) is provided as the input.

10. makeAvailbale() : Function to make a property available to buy. This function is also restricted to the land owner. Property ID is provided as the input. If the address matches the land owner the land is made available to buy.

11. buyProperty() : Function to buy the property which has already been approved to buy by the owner. This function can only be called by the requester whose request has been approved. The property ID is provided as the input and the transaction of ether corresponding to the market value and land tax is taken in exchange for the ownership of the property.

12. removeOwnership() : Function is called inside buyProperty(). Used to remove ownership of the previous owner of land. The property ID and the previous owner address is passed from buyProperty().

13. findId() : Function to find the index of the property in the asset list of the previous owner. This function is used inside removeOwnership(). Property ID and the previous owner address is passed from the removeOwnership().

3.1 Algorithm Used : Ethhash algorithm

Ethash is the planned PoW algorithm for Ethereum 1.0. It is the latest version of Dagger-Hashimoto, although it can no longer appropriately be called that since many of the original features of both algorithms have been drastically changed in the last month of research and development.

The general route that the algorithm takes is as follows:

1. There exists a seed which can be computed for each block by scanning through the block headers up until that point.
2. From the seed, one can compute a 16 MB pseudorandom cache. Light clients store the cache.
3. From the cache, we can generate a 1 GB dataset, with the property that each item in the dataset depends on only a small number of items from the cache. Full clients and miners store the dataset. The dataset grows linearly with time.
4. Mining involves grabbing random slices of the dataset and hashing them together. Verification can be done with low memory by using the cache to regenerate the specific pieces of the dataset that you need, so you only need to store the cache.

The large dataset is updated once every 30000 blocks, so the vast majority of a miner's effort will be reading the dataset, not making changes to it.

Definitions

We employ the following definitions:

```
WORD_BYTES = 4                    # bytes in word
DATASET_BYTES_INIT = 2**30        # bytes in dataset at genesis
DATASET_BYTES_GROWTH = 2**23      # dataset growth per epoch
CACHE_BYTES_INIT = 2**24          # bytes in cache at genesis
CACHE_BYTES_GROWTH = 2**17        # cache growth per epoch
CACHE_MULTIPLIER=1024             # Size of the DAG relative to the cache
EPOCH_LENGTH = 30000              # blocks per epoch
MIX_BYTES = 128                   # width of mix
HASH_BYTES = 64                   # hash length in bytes
DATASET_PARENTS = 256             # number of parents of each dataset element
CACHE_ROUNDS = 3                  # number of rounds in cache production
ACCESSES = 64                     # number of accesses in hashimoto loop
```

Copy

A note regarding "SHA3" hashes described in this specification

Ethereum's development coincided with the development of the SHA3 standard, and the
standards process made a late change in the padding of the finalized hash algorithm, so that Ethereum's
"sha3_256" and "sha3_512" hashes are not standard sha3 hashes, but a variant often referred
to as "Keccak-256" and "Keccak-512" in other contexts.

Parameters

The parameters for Ethash's cache and dataset depend on the block number. The cache size and dataset size both grow linearly; however, we always take the highest prime below the linearly growing threshold in order to reduce the risk of accidental regularities leading to cyclic behavior.

```
def get_cache_size(block_number):
    sz = CACHE_BYTES_INIT + CACHE_BYTES_GROWTH * (block_number // EPOCH_LENGTH)
    sz -= HASH_BYTES
    while not isprime(sz / HASH_BYTES):
        sz -= 2 * HASH_BYTES
    return sz

def get_full_size(block_number):
    sz = DATASET_BYTES_INIT + DATASET_BYTES_GROWTH * (block_number // EPOCH_LENGTH)
    sz -= MIX_BYTES
    while not isprime(sz / MIX_BYTES):
        sz -= 2 * MIX_BYTES
    return sz
```

Copy

Tables of dataset and cache size values are provided in the appendix.

Cache Generation

Now, we specify the function for producing a cache:

```
def mkcache(cache_size, seed):
    n = cache_size // HASH_BYTES

    # Sequentially produce the initial dataset
    o = [sha3_512(seed)]
    for i in range(1, n):
        o.append(sha3_512(o[-1]))

    # Use a low-round version of randmemohash
    for _ in range(CACHE_ROUNDS):
        for i in range(n):
            v = o[i][0] % n
            o[i] = sha3_512(map(xor, o[(i-1+n) % n], o[v]))

    return o
```

Copy

The cache production process involves first sequentially filling up 32 MB of memory, then performing two passes of Sergio Demian Lerner's RandMemoHash algorithm from Strict Memory Hard Hashing Functions (2014). The output is a set of 524288 64-byte values.

Data aggregation function

We use an algorithm inspired by the FNV hash in some cases as a non-associative substitute for XOR. Note that we multiply the prime with the full 32-bit input, in contrast with the FNV-1 spec which multiplies the prime with one byte (octet) in turn.

```
FNV_PRIME = 0x01000193

def fnv(v1, v2):
    return ((v1 * FNV_PRIME) ^ v2) % 2**32
```

Copy

Please note, even the yellow paper specifies fnv as v1*(FNV_PRIME ^ v2), all current implementations consistently use the above definition.

Full dataset calculation

Each 64-byte item in the full 1 GB dataset is computed as follows:

```
def calc_dataset_item(cache, i):
    n = len(cache)
    r = HASH_BYTES // WORD_BYTES
    # initialize the mix
    mix = copy.copy(cache[i % n])
    mix[0] ^= i
    mix = sha3_512(mix)
    # fnv it with a lot of random cache nodes based on i
    for j in range(DATASET_PARENTS):
        cache_index = fnv(i ^ j, mix[j % r])
        mix = map(fnv, mix, cache[cache_index % n])
    return sha3_512(mix)
```

Copy

Essentially, we combine data from 256 pseudorandomly selected cache nodes, and hash that to compute the dataset node. The entire dataset is then generated by:

```
def calc_dataset(full_size, cache):
    return [calc_dataset_item(cache, i) for i in range(full_size // HASH_BYTES)]
```

Copy

Main Loop

Now, we specify the main "hashimoto"-like loop, where we aggregate data from the full dataset in order to produce our final value for a particular header and nonce. In the code below, header represents the SHA3-256 hash of the RLP representation of a truncated block header, that is, of a header excluding the fields mixHash and nonce. nonce is the eight bytes of a 64 bit unsigned integer in big-endian order. So nonce[::-1] is the eight-byte little-endian representation of that value:

```
def hashimoto(header, nonce, full_size, dataset_lookup):
    n = full_size / HASH_BYTES
    w = MIX_BYTES // WORD_BYTES
    mixhashes = MIX_BYTES / HASH_BYTES
    # combine header+nonce into a 64 byte seed
    s = sha3_512(header + nonce[::-1])
    # start the mix with replicated s
    mix = []
    for _ in range(MIX_BYTES / HASH_BYTES):
        mix.extend(s)
    # mix in random dataset nodes
    for i in range(ACCESSES):
        p = fnv(i ^ s[0], mix[i % w]) % (n // mixhashes) * mixhashes
        newdata = []
        for j in range(MIX_BYTES / HASH_BYTES):
            newdata.extend(dataset_lookup(p + j))
        mix = map(fnv, mix, newdata)
    # compress mix
    cmix = []
    for i in range(0, len(mix), 4):
        cmix.append(fnv(fnv(fnv(mix[i], mix[i+1]), mix[i+2]), mix[i+3]))
    return {
        "mix digest": serialize_hash(cmix),
        "result": serialize_hash(sha3_256(s+cmix))
    }

def hashimoto_light(full_size, cache, header, nonce):
    return hashimoto(header, nonce, full_size, lambda x: calc_dataset_item(cache, x))

def hashimoto_full(full_size, dataset, header, nonce):
    return hashimoto(header, nonce, full_size, lambda x: dataset[x])
Copy
```

Essentially, we maintain a “mix” 128 bytes wide, and repeatedly sequentially fetch 128 bytes from the full dataset and use the fnv function to combine it with the mix. 128 bytes of sequential access are used so that each round of the algorithm always fetches a full page from RAM, minimizing translation lookaside buffer misses which ASICs would theoretically be able to avoid.

If the output of this algorithm is below the desired target, then the nonce is valid. Note that the extra application of sha3_256 at the end ensures that there exists an intermediate nonce which can be provided to prove that at least a small amount of work was done; this quick outer PoW verification can be used for anti-DDoS purposes. It also serves to provide statistical assurance that the result is an unbiased, 256 bit number.

3.2 Land Registry- Design Decisions

The dApp is created to make the land registration and land transaction transparent and decentralised. Angular is used to implement the front-end and

ethereum solidity contract is used in the back-end. Angular is a JavaScript

framework which is used to build single page applications. It’s main main feature is that we can build our applications in a modular fashion which helps in

reducing code repetition and makes it easier to debug. It also gives a way to

modify the HTML elements dynamically which makes it easier to create realtime interactive pages. The ethereum solidity contract in the back-end makes the dApp decentralised and transparent. The contract consists of mainly two fuctionalities.

1. Registration

Here the user provides the land details to the government authority who is registered as the superadmin. The land which is going to be registered should be in the same area as the superadmin who is going to register the land. The superadmin varifies the details with the existing records and enters into the dApp.

The details that are enrolled into the dApp are:

• state
• district
• village
• survey number
• owner address
• market value.

Along with these, an ID generated from the first four details of the land is also passed in. This ID is generated in the function “computeId()” using SHA256. The values entered in the registration form of the UI is passed in to the function “Registration()” and the details are mapped using the ID generated from above. Later on this mapping allows searching for a land easier.

There is another field in the registration page, which is, adding the superadmin. There the address of the superuser and the village in which he/she is working is provided. The village is mapped in to the address so that it becomes easier to check that only the superadmin assigned to a village is able to register the details of a land in that village.

2. Transaction: The transaction of a property has several stages involved. The algorithm is designed in such a way that there is no need for any central authority to varify the transaction process. It is important to note that the owner of a property can sell the land as a whole, i.e, there is no partial transaction of the property. This is just to simplify the problem in hand.

Later on, while improving the dApp, more of these functionalities can be added. The following are the steps involved:

• Making the land available: Once the buyer and seller agrees to make the transaction, the seller should make the land availble to buy. The land owner passes the property ID to the function “makeAvailbale()” and the function varifies the account of owner and changes the value of “isAvailable” to true which implies that the land is open to buy.

• Sending request to land owner: When the land is available to buy, the buyer sends a request to the land owner to buy the property. The ID of the land is feeded into the function “requestToLandOwner()”.The function varifies whether the land is availble to buy by checking the value of “isAvailable”. If the value is true, the buyer’s address is stored inside “requester” which was initailly 0 address. The value of “isAvailable” is then set to false so that no more request can be sent and request status is changed from “default” to “pending”.

The above two functions are important in the transaction process because if there is no “makeAvailbale” function then any one can send request to the land owner which will overwrite the request of the original buyer. If there is no request function then anyone who sent the exact amount to buy the property can actually get the property.

• Viewing the request: The function “viewRequest” takes the property ID as the input and returns the address of the requester. The function is for checking the address of the buyer.

• Processing the request: Once the seller views the requester address and if it is the right one, then the seller can process the request by inputting property ID, request status to the function “processRequest”.

The function, as usual, verifies whether the input is done by the owner of the land and process the request. If the requester address is not of the original buyer then the seller can reject the request and the function changes the value inside requester to 0 address and request status as default. This is for reverting the states to the original and starting the transaction process from zero.

• Buying the property: Once the request is approved the buyer can buy the property. The buyer enters the land ID to the function “buyProperty”. The function check whether the request status is approved or not and if it is approved, then it checks if the amount given is greater than the sum of market value and 10% of market value which goes as the land tax. If the conditions are satisfied then the amount is transferred to the land owner’s account. The functions then changes the ownership of the land to the buyer. Removing the ownership of the previous owner is done by calling another function “removeOwnership”. This function is called after the transfer of amount is complete. “removeOwnership” removes the property from the last owner’s asset list.

IV. RESULTS AND DISCUSSION

4.1 Gui Screenshots

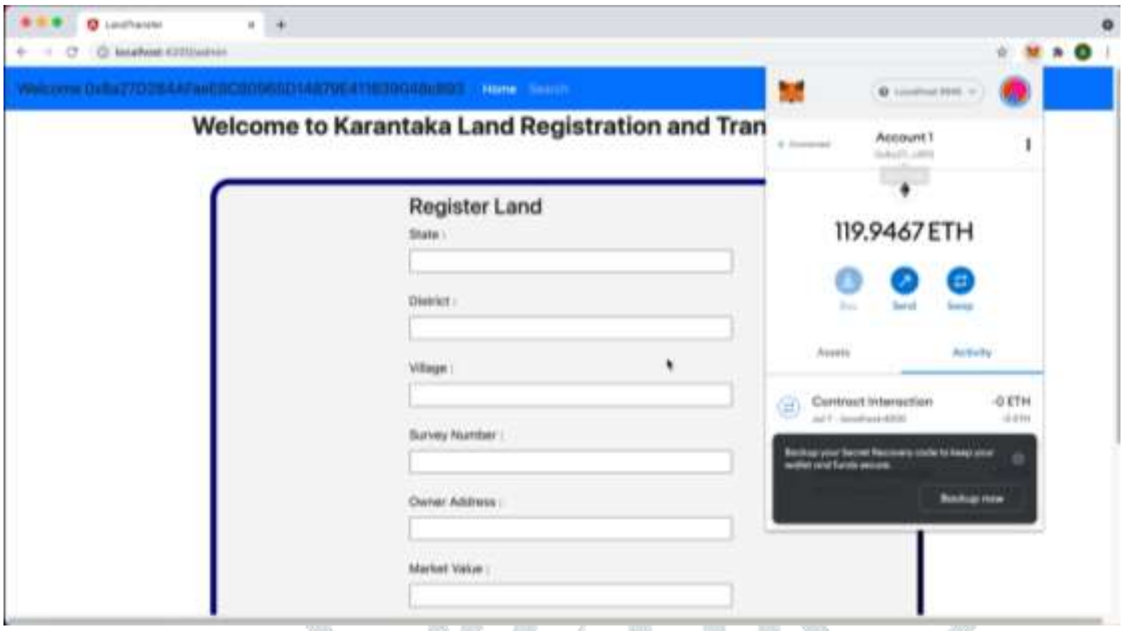


Fig 2: Admin login page

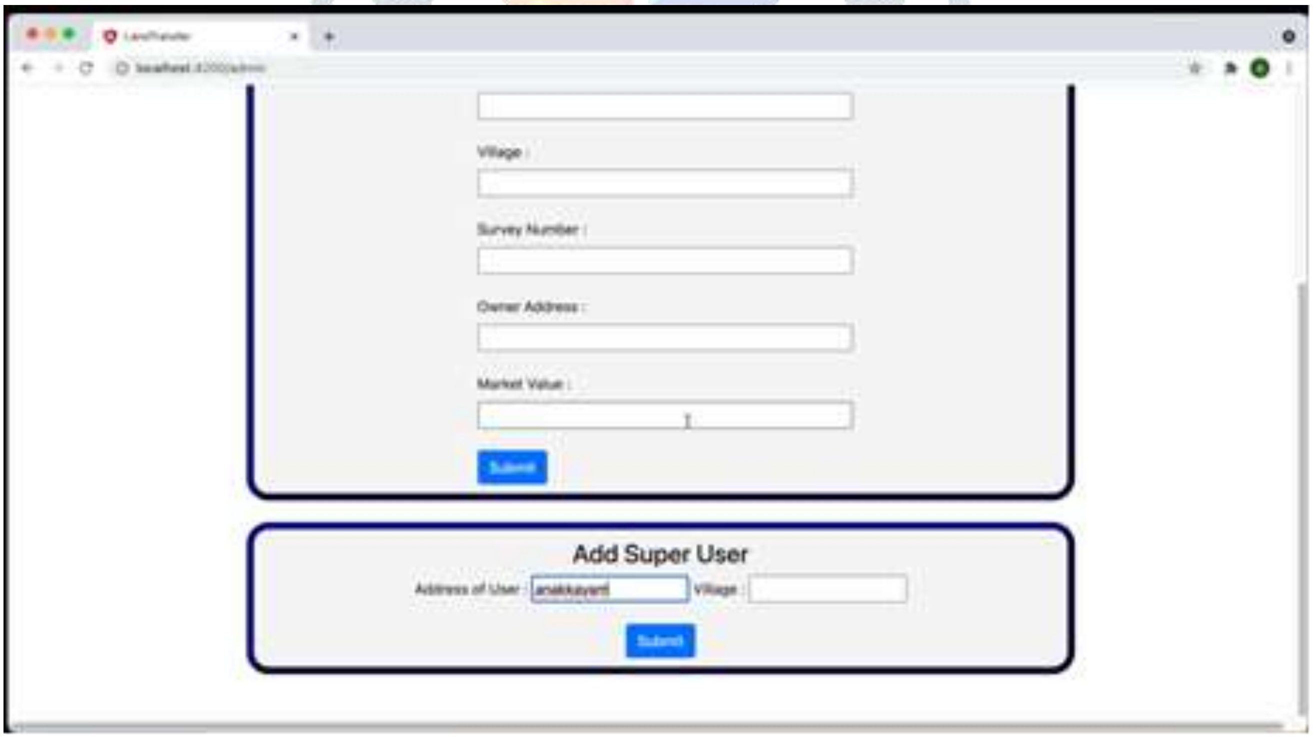
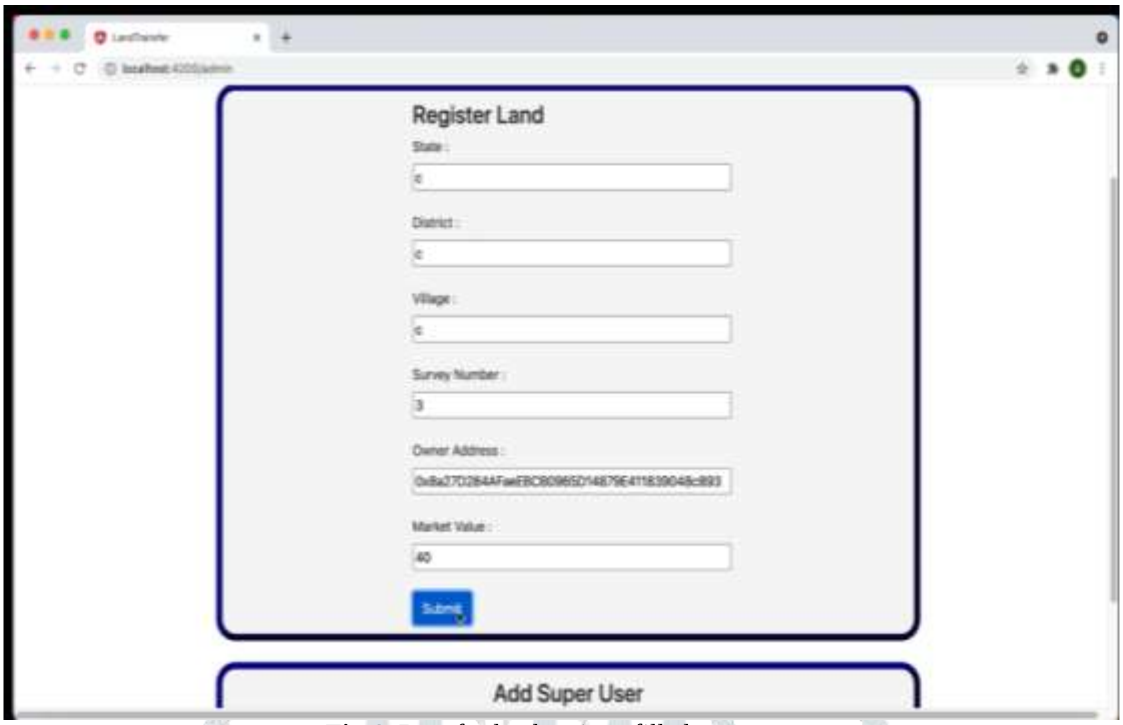


Fig 3: login using metascape



Register Land

State : c

District : c

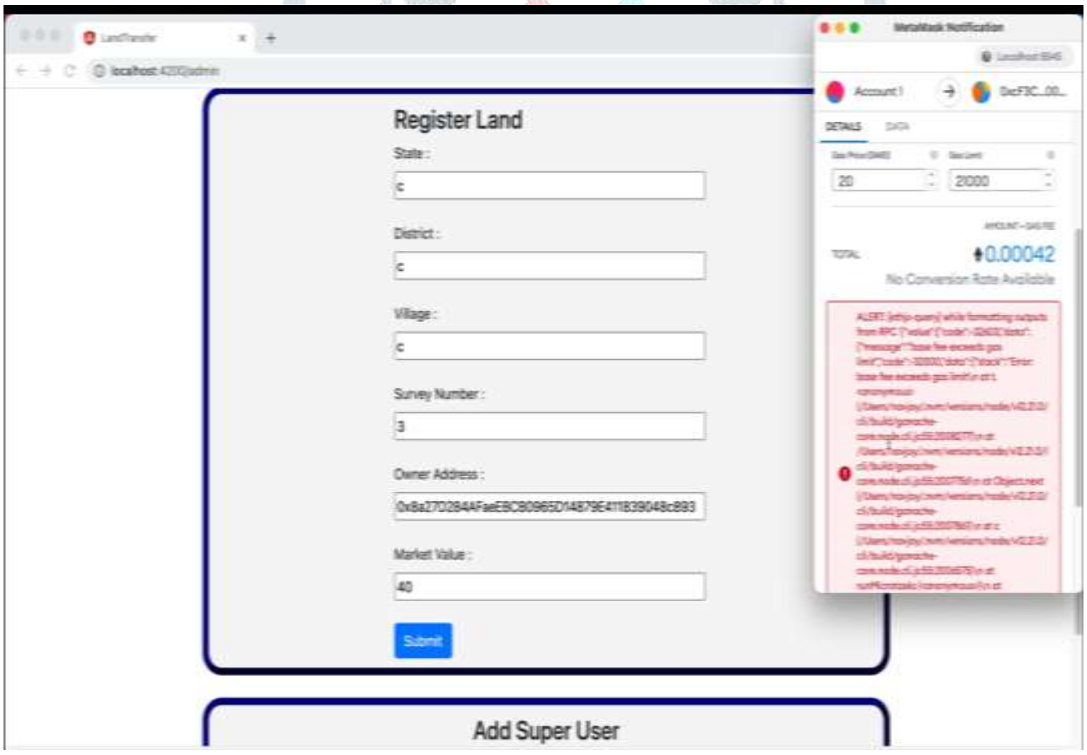
Village : c

Survey Number : 3

Owner Address : 0x8a27D284AFaeEBCB0965D14879E411839048c893

Market Value : 40

Fig 4: Data for land registry filled



Register Land

State : c

District : c

Village : c

Survey Number : 3

Owner Address : 0x8a27D284AFaeEBCB0965D14879E411839048c893

Market Value : 40

MetaMask Notification

Account 1 → 0xcF3C...00...

Gas Price (GWEI): 20

Gas Limit: 21000

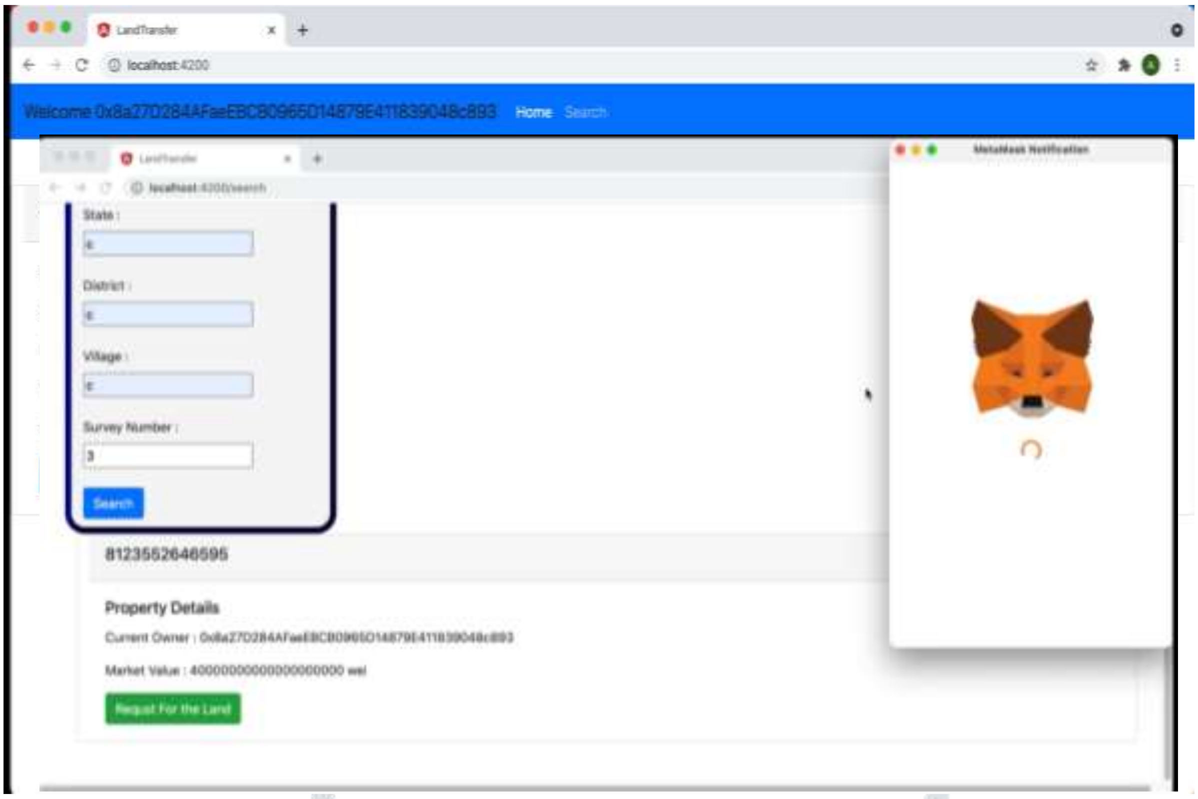
AMOUNT + GAS FEE

TOTAL ⧫0.00042

No Conversion Rate Available

```
ALERT: [ethjs-query] while formatting outputs
from RPC '{"value":{"code":-32603,"data":
{"message":"base fee exceeds gas
limit","code":-32000,"data":{"stack":"Error:
base fee exceeds gas limit\n at t.
<anonymous>
[illegible]
```

Fig 5: Notification for wrong details that has to accepted and confirmed


LandTransfer

localhost:4200

Welcome 0x8a27D284AFaeEBC80965D14879E411839048c893 Home Search

localhost:4200/search

State : c

District : c

Village : c

Survey Number : 3

Search

8123552646595

Property Details

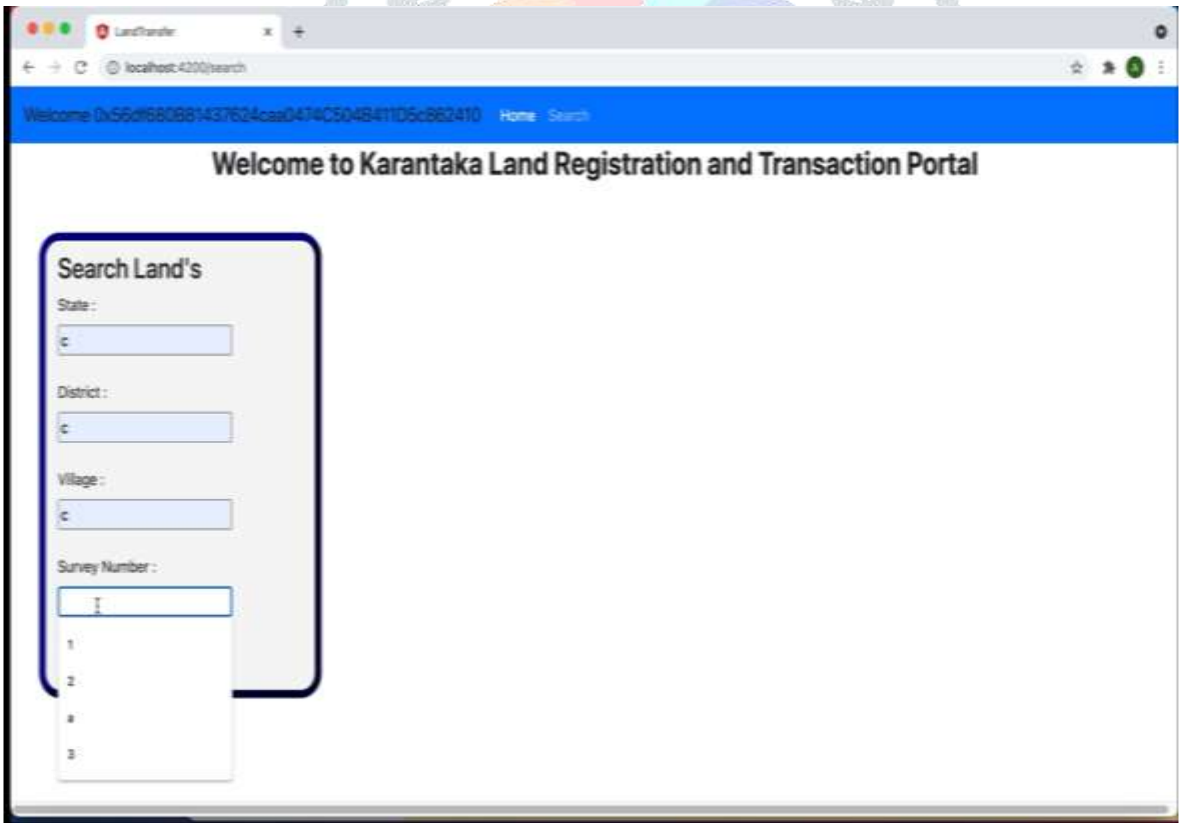
Current Owner : 0x8a27D284AFaeEBC80965D14879E411839048c893

Market Value : 4000000000000000000 wei

Request For the Land

MetaMask Notification

Fig 6: Property details obtained


LandTransfer

localhost:4200/search

Welcome 0x56df680B81437624caa0474C504B411D5c862410 Home Search

Welcome to Karantaka Land Registration and Transaction Portal

Search Land's

State : c

District : c

Village : c

Survey Number :

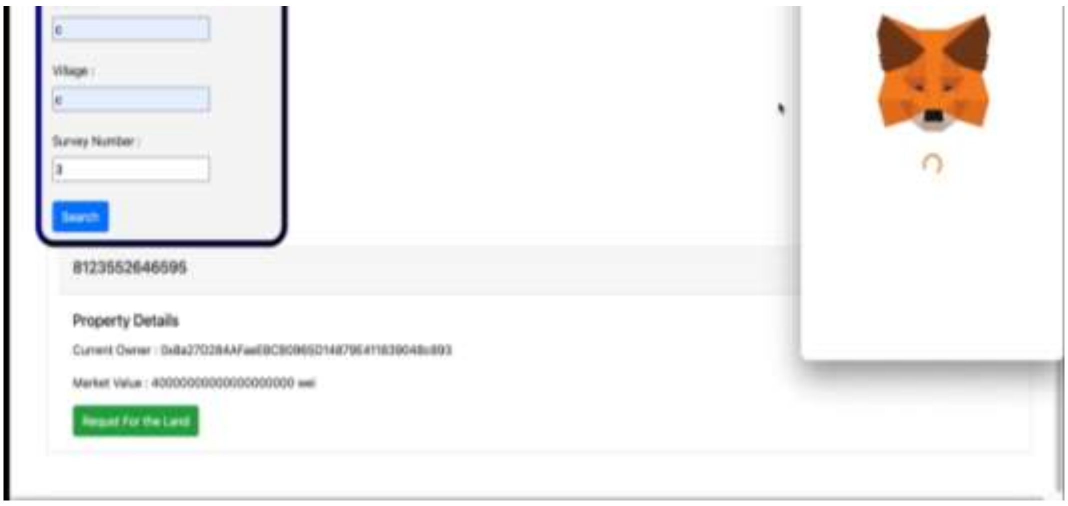
1

2

4

3

Fig 7: Buyer searching land details



c

Village :

c

Survey Number :

3

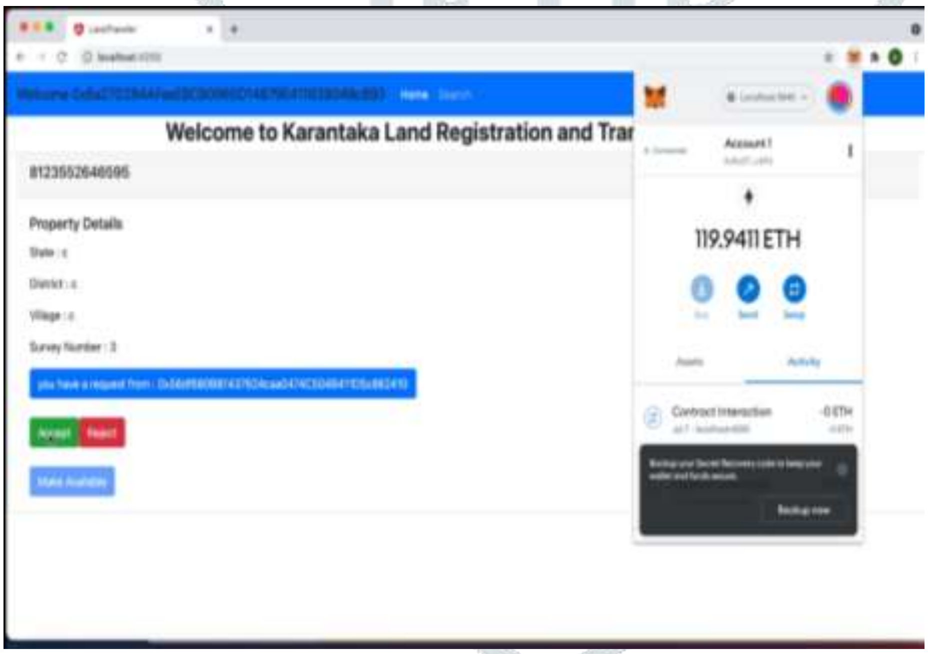
8123552646595

Property Details

Current Owner : 0x8a27D284AFaeEBC80965D14879E411839048c893

Market Value : 40000000000000000000 wei

Fig 8: Request for the land for buying



Welcome to Karantaka Land Registration and Trar

8123552646595

Property Details

State : c

District : c

Village : c

Survey Number : 3

you have a request from : [illegible]

119.9411 ETH

Contract Interaction -0 ETH

Fig 9: Land owner sees the request of buying land

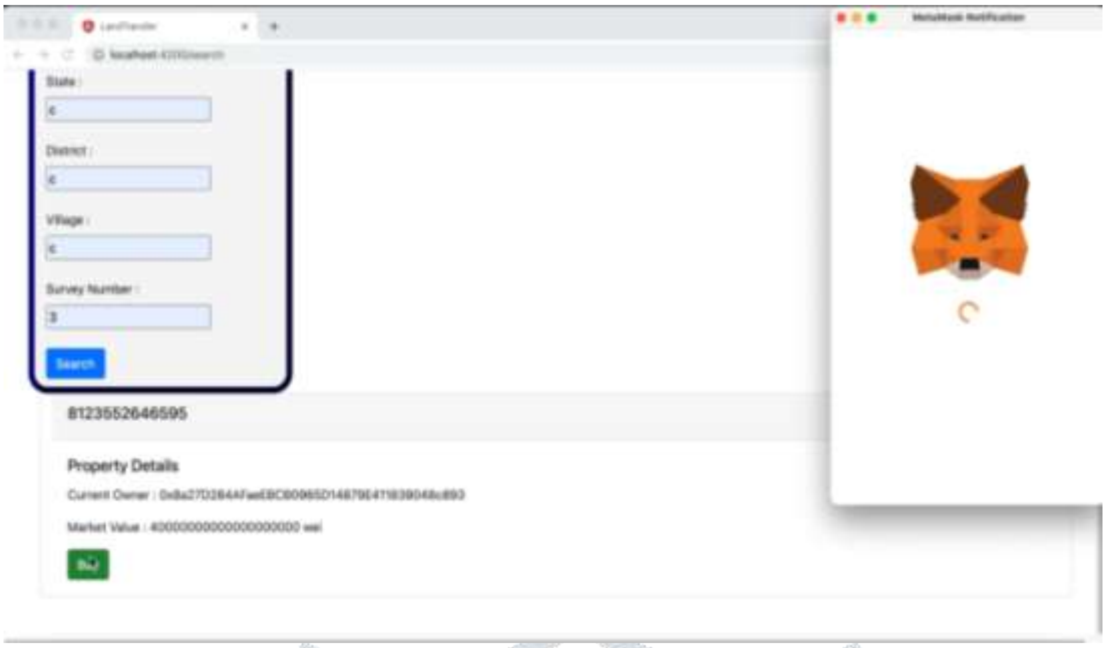

State :
c
District :
c
Village :
c
Survey Number :
3
Search
8123552646595
Property Details
Current Owner : 0x8a27D264AFaeEBC60965D14679E471839048c893
Market Value : 4000000000000000000 wei

Fig 10: The buyer moves ahead for buying after the seller accepts his request

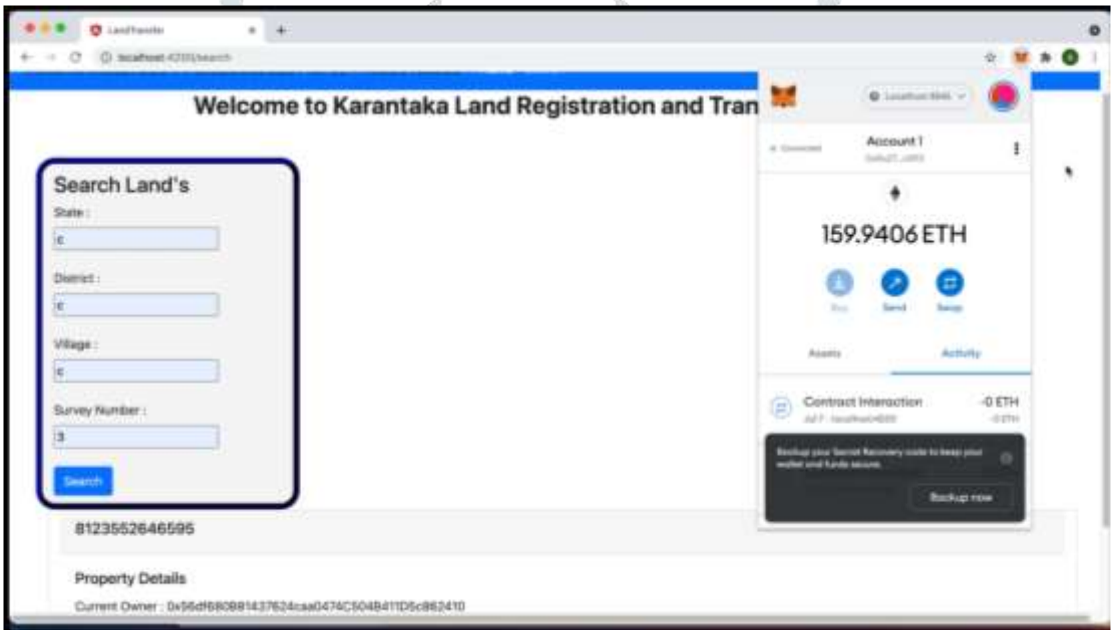

Welcome to Karantaka Land Registration and Tran
Search Land's
State :
c
District :
c
Village :
c
Survey Number :
3
Search
8123552646595
Property Details
Current Owner : 0x56df680B81437624caa0474C504B411D5c862410
Account 1
159.9406 ETH
Contract Interaction -0 ETH

Fig 11: Seller pays the landowner using Ethereum

IV. Conclusion

In this paper the authors proposed a seamless, easy to use and hustle-free platform which can be used for making the land registration easy. There are many problems such as involvement of brokers or middleman, time delays, etc. This platform will eliminate the problems associated with land registration in India as well as in many parts of the world. The steps involved in the process of land registration are discussed in details in the paper. Making land registration paperless will not only make the process easier but also secure the papers of ownership of land from various man-made and natural disasters. The blockchain technology is emerging very rapidly due to secure features it offer. Hence using blockchain to save the land record transaction is the way to create the immutable records. There are many additional features that can be added to the platform of land registry. Nowadays, land is not a liquidated asset. By using the platform land assets can also be liquidated using the cryptocurrency, that maps with the land record created by a seller on the platform. Hence the scope is wide and there can be many use cases of the platform created.

V. FUTURE SCOPE

The information related to land records, registration and surveying is maintained at different levels like Tehsils and Blocks. All the data is not available centrally, rather it is maintained in isolation at different levels. Further, the boundary updating of land parcels are still not updated and not clearly demarked for individual users. Still the shares are maintained among the group of owners. The individual transactions should lead to clear, identified entity or object. Hence all the parcel boundaries and shares should be clearly identified by boundary coordinates. Before initiation of the Blockchain implementation, the States should focus on this point. Further the data should be maintained in central servers at appropriate hierarchy and it should be for all the Departments like Land Revenue Department, Registration and Survey and Settlement Department etc. Since the data is maintained at different Departments business process engineering is required for each State to have a standard operating procedure. But land being a State subject in our legislation, the State has the exclusive power on this subject. Government of India should make standard operating procedures for implementation of Blockchain based systems applicable to all the States. The current process of designing of a Blockchain system is based on the subject or context. There is no generic and uniform design of Blockchain available. The research should be carried out in these directions. The load on the Blockchain system will gradually increase and the transactions will increase because of population growth. The legal requirements to meet the need of any disputes in case of the Blockchain Implementation also need to be notified by the Government. In future it is also very essential to integrate the Blockchain Technology with Artificial Intelligence (AI) for making the complete land management eco-system safer, faster, transparent and responsive.

REFERENCES

[1] Baliga, Arati. "Understanding blockchain consensus models." Persistent 2017.4 (2017): 1-14.
[2] Vos J. Blockchain-based land registry: panacea illusion or something in between?. InIPRA/CINDER Congress, Dubai. European Land Registry Association (ELRA) 2017 Oct.
[3] Anand A, McKibbin M, Pichel F. Colored coins: Bitcoin, blockchain, and land administration. InAnnual World Bank Conference on Land and Poverty 2016.
[4] Oprunenco A, Akmeemana C. Using blockchain to make land registry more reliable in India. LSE Business Review. 2018 Apr 13.
[5] Alketbi A, Nasir Q, Talib MA. Blockchain for government services—Use cases, security benefits and challenges. In2018 15th Learning and Technology Conference (L&T) 2018 Feb 25 (pp. 112-119). IEEE.
[6] Barbieri M, Gassen D. Blockchain–can this new technology really revolutionize the land registry system. InLand and Poverty Conference 2017: Responsible Land Governance 2017.
[7] Graglia JM, Mellon C. Blockchain and Property in 2018: At the End of the Beginning. Innovations: Technology, Governance, Globalization. 2018 Jul;12(1-2):90-116.
[8] Benbunan-Fich R, Castellanos A. Digitization of Land Records: From Paper to Blockchain.
[9] Valenta, Martin, and Philipp Sandner. "Comparison of Ethereum, hyperledger fabric and corda." [ebook] Frankfurt School, Blockchain Center (2017).
[10] Gencer AE, Basu S, Eyal I, Van Renesse R, Sirer EG. Decentralization in bitcoin and Ethereum networks. arXiv preprint arXiv:1801.03998. 2018 Jan 11.
[11] Deininger, K., & Feder, G. (2009). Land registration, governance, and development: Evidence and implications for policy. The World Bank Research Observer, 24(2), 233-266.
[12] Zevenbergen, Jaap. "Systems of land registration aspects and effects." Publications on Geodesy, 51 (2002).
[13] Hanstad, Tim. "Designing land registration systems for developing countries." Am. U. Int'l L. Rev. 13 (1997): 647.
[14] Bogner, Andreas, Mathieu Chanson, and Arne Meeuw. "A decentralised sharing app running a smart contract on the Ethereum blockchain." Proceedings of the 6th International Conference on the Internet of Things. ACM, 2016.
[15] Wüst, Karl, and Arthur Gervais. "Do you need a Blockchain?." 2018 Crypto Valley Conference on Blockchain Technology (CVCBT). IEEE, 2018.
[16] Vujičić, Dejan, Dijana Jagodić, and Siniša Ranđić. "Blockchain technology, bitcoin, and Ethereum: A brief overview." 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH). IEEE, 2018.