Develop Leave Application using Blockchain Smart Contract

Vinayak Singla*, Indra Kumar Malav†, Jaspreet Kaur‡ and Sumit Kalra§
Deptartment of Computer Science & Engineering
Indian Institute of Technology Jodhpur
Jodhpur, India
Email: *singla.1@iitj.ac.in, †malav.1@iitj.ac.in, ‡kaur.3@iitj.ac.in, §sumitk@iitj.ac.in

***Abstract*—In today's time, we use Leave Application System managed by some central authority. This can lead to corruption, misuse by a single authority or unnecessary time for approval. So, in this paper, we have developed smart contract architecture for Leave Management System using Solidity and Ethereum, which is a decentralized app architecture leveraged from Blockchain Technology. We also include smartphones as IoT devices by which any client can use this app. Due to the resource-constrained nature of these devices, the smart contract app is coupled with an alternative centralized architecture which is a classic client/server architecture with an underlying Blockchain backend.**

***Index Terms*—EVM (Ethereum Virtual Machine), Smart Contract, Blockchain, Ethereum, Solidity**

I. Introduction

In today's time, leave management application work as a centralized client-server system. it's take a lots of time and human labor for approval. Due to centralization, users must have to bind their trust at that server. There is a chance, server data may get damaged or misused for someone's profit. So, we need some new technology such as smart contract decentralized application which solves all of the above limitations.

When we consider the contracts in the real world, we always need someone as an intermediary who verifies that the contract is signed by the two or multiple parties involved and this intermediary also has the job of enforcing involved parties to fulfill the contracts condition but a smart contract is smart as it is a self-executing contract which doesn't need an intermediary. This is made possible with the use of cryptographic techniques which allows digital signing along with Blockchain which can be thought of as an immutable distributed ledger.

We create a smart contract for Leave Management System(LMS). Further, As the Blockchain requires high computational power and lot of I/O operations thus it makes it impossible to run a Blockchain node on mobile devices(IoT devices) which generally have less memory and comparatively less computing power. But one should be able to apply for leave using his phone, also not everyone would want to have a complete Blockchain node running on their laptops just to apply for a leave. To solve this problem, we propose a decentralized app architecture coupled with an alternative centralized architecture for smart contract applications(along with a working demo) which allow the users to make transactions on a smart contract deployed in Blockchain irrespective of the fact whether they are running a Blockchain node on their devices or not.

In section 2, we describe some related technologies or terms. Then we describe our proposed architecture followed by some preliminary results. Finally, the conclusion is given along with future directions.

II. Some related technologies or terms

Here, we briefly describe some related technologies used in this application as:

A. Private Blockchain

We created the smart contract application on private Ethereum network as in this network only we are making transactions on Blockchain so the rate of the morphing of Blockchain was relatively very less as compared to the case of public Blockchain and thus syncing was possible even with an HDD node easily. For Setting a Blockchain in private Ethereum network, we have to create a genesis file. As we know that genesis block is the first block in the Blockchain and genesis file is used to create it. This file includes configuration features(such as Difficulty level, Gas Limit and many more) of your Blockchain. After genesis file was created it was used to create genesis block for both Blockchains. Connecting two nodes in private Ethereum network was done explicitly. To verify that both chains are communicating properly we started two node instances on different ports and started mining on just one instance then we checked the block number on the 1st node and then on the 2nd node. Both were same which ensured that both nodes were able to get in sync. Finally, we were able to run multiple nodes in a private Ethereum network.

B. Smart Contract in Solidity Language

Solidity is a Contract Oriented Language, used for writing smart contracts which can be deployed on EVM. It follows an object-oriented approach and support features like inheritance, complex data types among many. We have created a simple leave management system with the following functionality:

- A user can apply for leave and he has to just give the start date and end date of the leave.

- There is one admin account who can get all the applied leaves and then approve or reject them (verified by multiple miners or nodes). This admin account is kind of like master account which was used to deploy the contract.
- Any user can check the status of his/her leaves.
- Admin can check the status of any leave.

Current Contract Design

- Following are the Complex data types created:

```
1. struct user {
        string id;
        address account;
          }
2. struct leaveApplication  {
        address applicant;
        uint startDate;
        uint endDate;
        statusChoices status;
          }
```

- There is a array of leaves to store leave applications.
- A map to map each address of account to a user.

C. *Deploying and Testing Contract using Truffle*

Truffle is a development environment and a testing framework. This helps in the automatic compilation and deploying of contract on Blockchain without much problem. It also helps while making migration in Blockchain. Finally using truffle we deployed the contract on a private Ethereum Blockchain. Deploying the contract on Blockchain is also a transaction and hence required ether which can be obtained by simply doing miner.start(1) in your geth console. (Note: Remember the difficulty parameter we talked about in Genesis file, the value of the parameter should be less or else it will become a pain for you to mine ether).

III. Proposed Architecture

The decentralized architecture of the app formed by deploying smart contract in Blockchain requires the user to have a fully synced instance of Blockchain node running on his device for him to make any transaction. On mobile devices, this becomes a problem because of computational and storage resources. Moreover, it will be fair to say that many users wont want to set up and run a Blockchain node on their device just to apply for a leave.

The architecture of the application constitutes of a decentralized architecture of Blockchain and a centralized client/server architecture which constitutes of a NodeJS server with all APIs written in REST Architecture allowing to create multiple interfaces like an android app, ios app, web interface all having the same backend with a SQL Database. We have X Blockchain nodes which will always be running, in between these nodes and NodeJS server there will be a Load Balancer which will divide the incoming request traffic among various nodes.

Case1: Client uses web interface to make transaction. No need to run an Ethereum node as shown in Fig1.

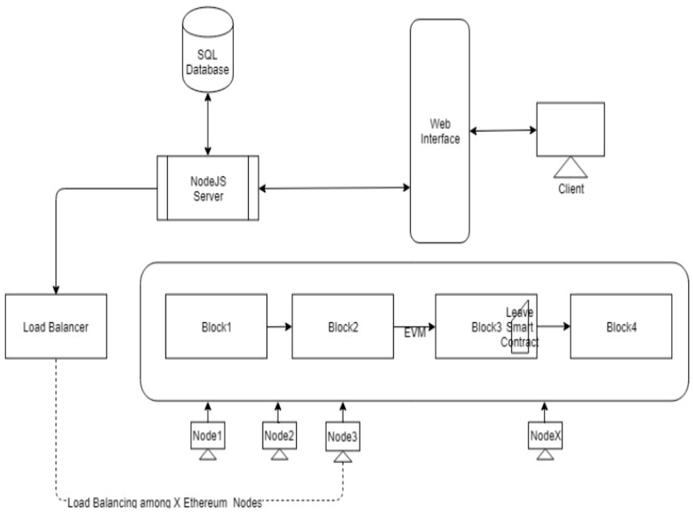


Fig. 1. Architecture of the Smart Contract App Case1: Client interacts with Blockchain without running Blockchain Node via web interface

Case2: Client directly makes transaction on Blockchain. Client has to run an Ethereum node as shown in Fig2.

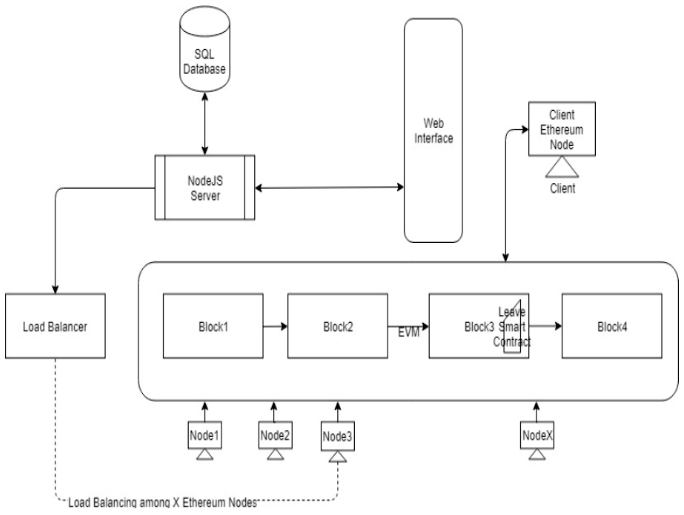
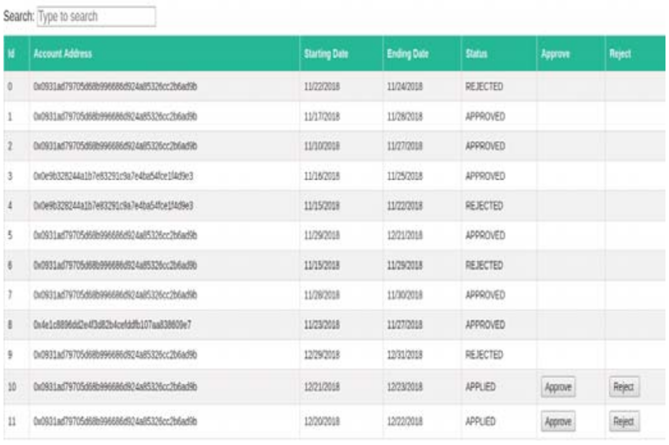


Fig. 2. Architecture of the Smart Contract App Case2: Client directly interacts with Blockchain

IV. Observations & Results

Finally, the end product is Leave Management System, with a login page(for admin & users). After login, another web page opens, in which users apply to leave or see leave approval status or admin can accept or reject leave requests. For more visibility, the screenshot of admin page is shown below in figure 3.


Search: Type to search

Id	Account Address	Starting Date	Ending Date	Status	Approve	Reject
0	0x0931ad79705d68b996686d924a85326cc2b6ad9b	11/22/2018	11/24/2018	REJECTED		
1	0x0931ad79705d68b996686d924a85326cc2b6ad9b	11/17/2018	11/28/2018	APPROVED		
2	0x0931ad79705d68b996686d924a85326cc2b6ad9b	11/10/2018	11/27/2018	APPROVED		
3	0x0e9b328244a1b7e83291c9a7e4ba54fce1f4d9e3	11/16/2018	11/25/2018	APPROVED		
4	0x0e9b328244a1b7e83291c9a7e4ba54fce1f4d9e3	11/15/2018	11/22/2018	REJECTED		
5	0x0931ad79705d68b996686d924a85326cc2b6ad9b	11/29/2018	12/21/2018	APPROVED		
6	0x0931ad79705d68b996686d924a85326cc2b6ad9b	11/15/2018	11/29/2018	REJECTED		
7	0x0931ad79705d68b996686d924a85326cc2b6ad9b	11/28/2018	11/30/2018	APPROVED		
8	0x4e1c8896dd2e4f3d82b4cefddfb107aa838609e7	11/23/2018	11/27/2018	APPROVED		
9	0x0931ad79705d68b996686d924a85326cc2b6ad9b	12/29/2018	12/31/2018	REJECTED		
10	0x0931ad79705d68b996686d924a85326cc2b6ad9b	12/21/2018	12/23/2018	APPLIED	Approve	Reject
11	0x0931ad79705d68b996686d924a85326cc2b6ad9b	12/20/2018	12/22/2018	APPLIED	Approve	Reject

Fig. 3. Screenshot of Admin Page

To make transactions we provide each user with some ethers at the signup by doing mining on one of the x nodes. These ether can be considered to be infinite for the transactions in our app as shown by the example in Fig3.
Initial Balance = 2.9499991e+20 wei
Cost of one transaction = 9.9122e+13 wei
Total Transactions that can be made assuming that user uses the system for 50 years he can apply for more than 50 + leaves in a day on average.

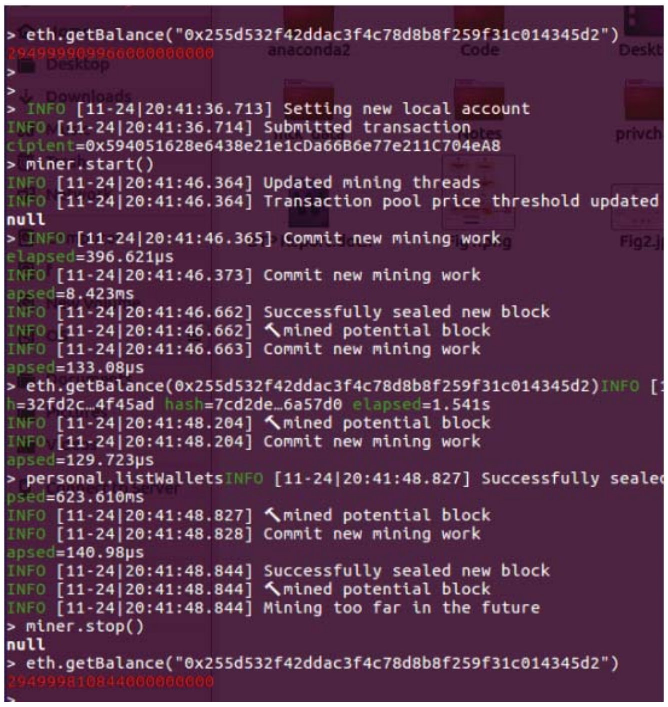

```
> eth.getBalance("0x255d532f42ddac3f4c78d8b8f259f31c014345d2")
294999909966000000000
>
>
> INFO [11-24|20:41:36.713] Setting new local account
INFO [11-24|20:41:36.714] Submitted transaction
cipient=0x594051628e6438e21e1cDa66B6e77e211C704eA8
> miner.start()
INFO [11-24|20:41:46.364] Updated mining threads
INFO [11-24|20:41:46.364] Transaction pool price threshold updated
null
> INFO [11-24|20:41:46.365] Commit new mining work
elapsed=396.621µs
INFO [11-24|20:41:46.373] Commit new mining work
apsed=8.423ms
INFO [11-24|20:41:46.662] Successfully sealed new block
INFO [11-24|20:41:46.662] ⛏mined potential block
INFO [11-24|20:41:46.663] Commit new mining work
apsed=133.08µs
> eth.getBalance(0x255d532f42ddac3f4c78d8b8f259f31c014345d2)INFO [
h=32fd2c…4f45ad hash=7cd2de…6a57d0 elapsed=1.541s
INFO [11-24|20:41:48.204] ⛏mined potential block
INFO [11-24|20:41:48.204] Commit new mining work
apsed=129.723µs
> personal.listWalletsINFO [11-24|20:41:48.827] Successfully seale
psed=623.610ms
INFO [11-24|20:41:48.827] ⛏mined potential block
INFO [11-24|20:41:48.828] Commit new mining work
apsed=140.98µs
INFO [11-24|20:41:48.844] Successfully sealed new block
INFO [11-24|20:41:48.844] ⛏mined potential block
INFO [11-24|20:41:48.844] Mining too far in the future
> miner.stop()
null
> eth.getBalance("0x255d532f42ddac3f4c78d8b8f259f31c014345d2")
294999810844000000000
>
```

Fig. 4. Screenshot Showing Cost of Transaction for Applying Leave

V. CONCLUSION & FUTURE DIRECTION

In this paper, we provide a new secure reliable leave management system through blockchain smart contract handled via mobile or IoT devices. At this time this app has limited functionality such as leave application is applicable only at educational institutes but in future, we will extend this industry as well as government level. We will also include some additional features as:

- **Blockchain at IoT Devices:** In the current scenario, we do not run blockchain on the mobile devices due to the resource-constrained nature of these. But we will add some techniques by which users have to manage blockchain at their end. We will also include some other IoT devices for login facilities such as smart watches.
- **Mining Resource Distribution Algorithm:** When the user is making a transaction without running an Ethereum node even then he needs ether in his account but he cant mine ether without running Ethereum node. So obviously this mining has to be done on the server side. But how do we distribute this mining resource among clients? Do we charge them for ethers? If not then what is the incentive for other to contribute to mining? These all are some more design questions which need more thought. Currently, we provide every user with some ethers at signup but this can be improved.
- **Software Based Load Balancing:** Though many Load Balancing solutions are readily available from companies like CISCO, AVI. AVI networks, in particular, provide a software Load Balancer which can be programmed according to the requirements. When the system becomes more complex and you need to do load balancing between mining requests then to realize your algorithm for mining resource distribution, a software-based load balancer will be required.

REFERENCES

[1] Mahesh Murthy,'Full Stack Hello World Voting Ethereum Dapp Tutorial Part 2'.2017[Online]. Available:https://medium.com/@mvmurthy/full-stack-hello-world-voting-ethereum-dapp-tutorial-part-2-30b3d335aa1f.[Accessed: 8- Sep- 2018]

[2] Prashant Ram,'How to set up a multi-node private Ethereum blockchain on your Mac'.2018[Online]. Available:https://medium.freecodecamp.org/how-to-set-up-a-multi-node-private-ethereum-blockchain-from-scratch-in-20-mins-or-less-e0d7e091e062.[Accessed: 15- Sep- 2018]

[3] Viktor Trn,Hudson Jameson "Homestead Documentation Initiative", Ethereum community,2016.[Online]. Available: http://www.ethdocs.org/en/latest/introduction/index.html.

[4] G. Wood, C. Reitwiessner, A. Beregszaszi, L. Husikyan, Y. Hirai, "Solidity Documentation", Ethereum Corp.,2016.[Online]. Available: https://solidity.readthedocs.io/en/v0.4.25/

[5] F. Vogelsteller, M. Kotewicz, J. Wilcke, M. Oance, "web3.js Documentation", Ethereum Corp.,2016.[Online]. Available: https://web3js.readthedocs.io/en/1.0/web3-eth.html

[6] Prashant Ram,'How to set up a private Ethereum blockchain and deploy a Solidity Smart Contract on the blockchainin less than 20 mins!'.2018[Online]. Available:https://hackernoon.com/set-up-a-private-ethereum-blockchain-and-deploy-your-first-solidity-smart-contract-on-the-caa8334c343d[Accessed: 17- Oct- 2018]