**Blockchain-DPOS-Land-Management System:**

This project aims to develop a system for supply chain management with the following characteristics:

To register new distributors to the system to buy products from the manufacturer.

Register new clients to the system to buy products from the distributor if they can sell that product.

Delegated Proof of Stake (DPOS) consensus algorithm is incorporated to improve the security of the blockchain.

Implementing Merkle tree to calculate the root hash of all the transactions inside a block.

Users can view the transaction history that is related to a product.

Technology Used:

* JavaScript
* Node.js

**Working of Every Module:**

**blockchain.js file:**

Blockchain (): This function initializes an empty array called blockchain, which will contain all the blocks in sequential order. It will also initialize an empty array called newTransaction, which will contain all the transactions in chronological order. It will also create the genesis block.

addBlock (): It will have parameters of transaction data, hash of the previous block, timestamp, and hash of the current block. It will create a new block with all its data and add it to the array "blockchain".

getLastBlock (): This will return the last block in the array "blockchain".

createNewTrans(): This will take property Id, seller Id & buyer Id as parameters. It will create a new structure called 'newTransactions' which will have seller, buyer, and property Id. This transaction will be added into the array 'newTransaction'.

calculateHash (): This will take the hash of the previous block, Merkel root & and data as parameters. This will use the sha256 algorithm to calculate the hash of the block using all the parameters and current time.

DPOS (): Delegated Proof of Stake is a blockchain consensus mechanism where network users vote and elect delegates to validate the next block. Like a traditional proof-of-stake mechanism, DPoS uses a collateral staking system. However, it also uses a specific democratic process designed to address POS's limitations. This function returns the calculated hash of the current block, timestamp, and also the Merkel root.

merkelRoot(): It takes number of transaction as a parameter. It implements the merkel tree to calculate the merkel root .

registration.js file

Registration(): It initializes the userId array which will contains the ids of all the registered users.It will also initialize the propertyId array which will contain Ids from 1 to 10.

addUser(): This function will check if the property Id requested by the user is available in the propertyId array and if it is present the it will register the user by adding his Id in userId array, If not then it will throw the error ‘Property doesn’t exist in the system’.

test.js:

It will include all the files in the project and give them object name.

It will create instances of all the js files which then will be used to access all the functions from these files.

This file creates a menu which displays all the options to be used to use all the features of the project like buying, selling and mining the block. This file is to be run to use this project.

To run this file use command “node test.js” in the terminal.

transaction.js file

buySell(): It initializes the available property array called availableProp which has values from 1 to 10. It also initializes array called usersRegistered.

buy():It takes userId and propertyId as parameters and checks if property is owned by someone, If yes then it will throw the error ‘Property already owned by someone’. If no, then it assigns the property to that user.

sell(): It takes userId and propertyId as parameters and checks if userId is assigned to that property. If it is, then it returns ‘success’ and removes the link between the user and property.If it is not assigned then it throws the error ‘This is not your property’.