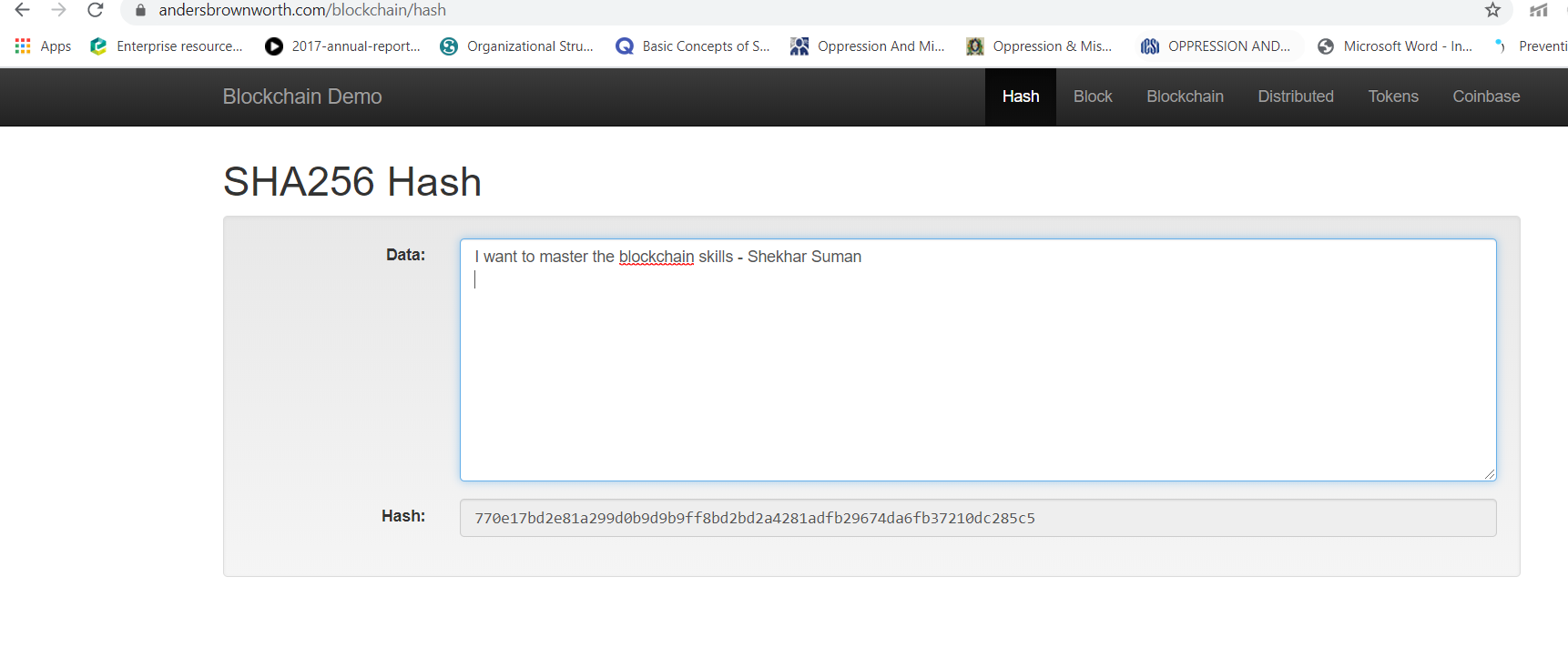
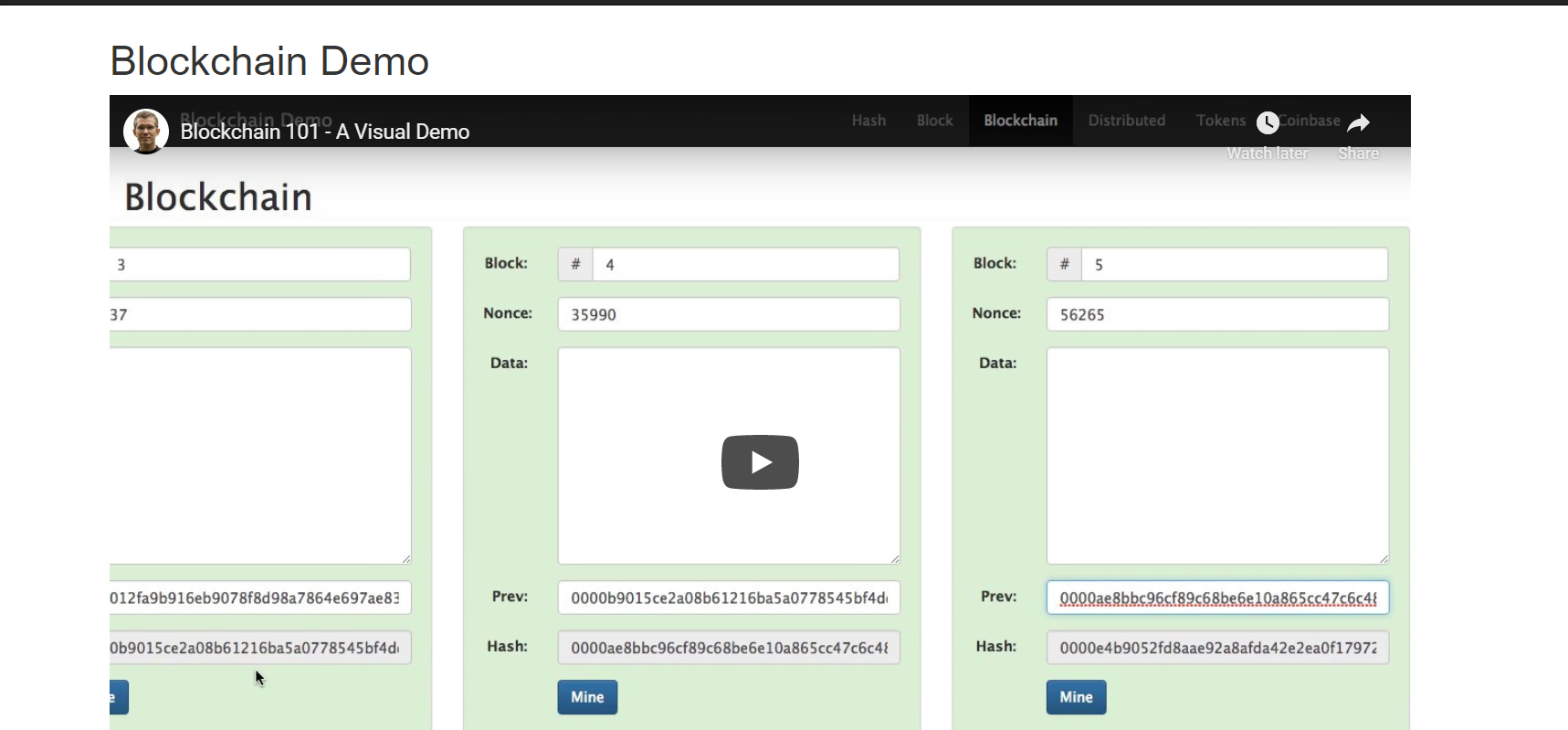
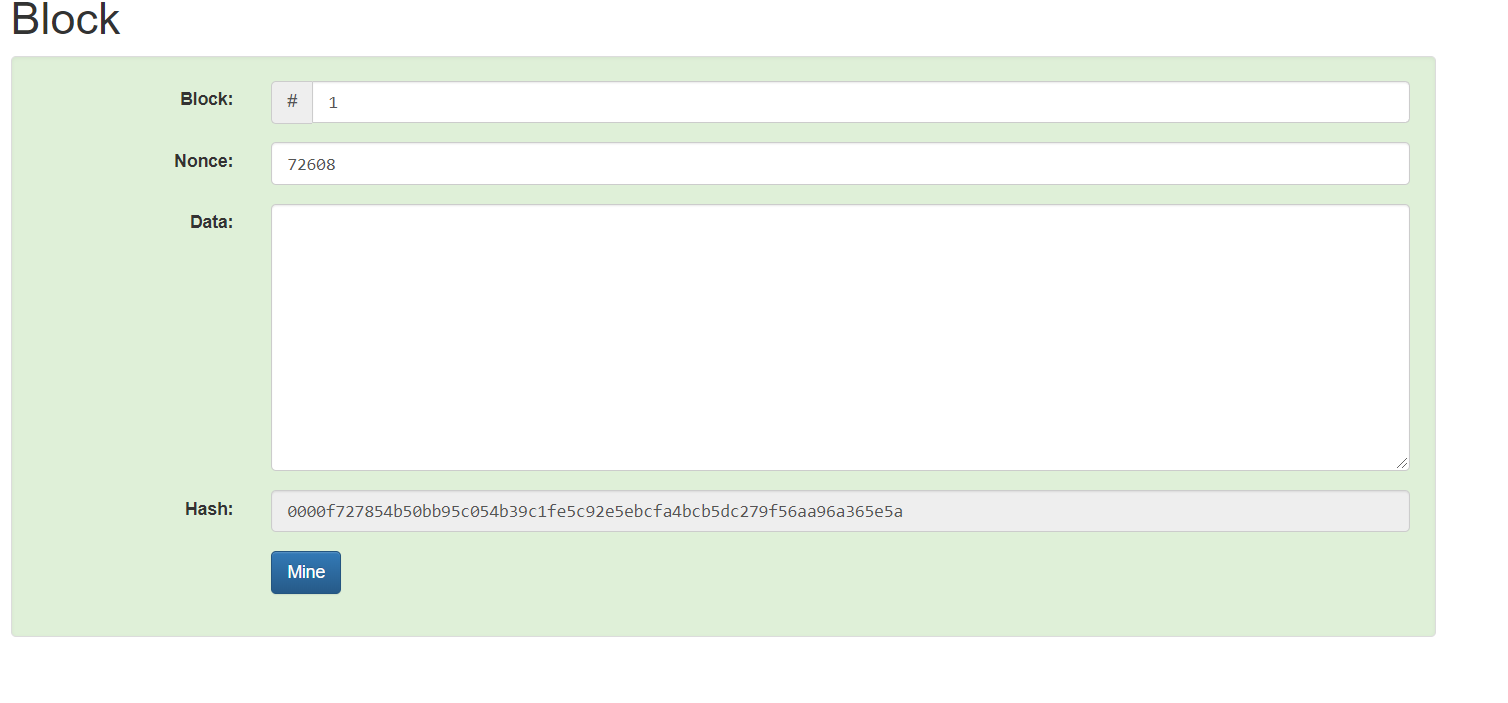
**DAY-3 Assignment**

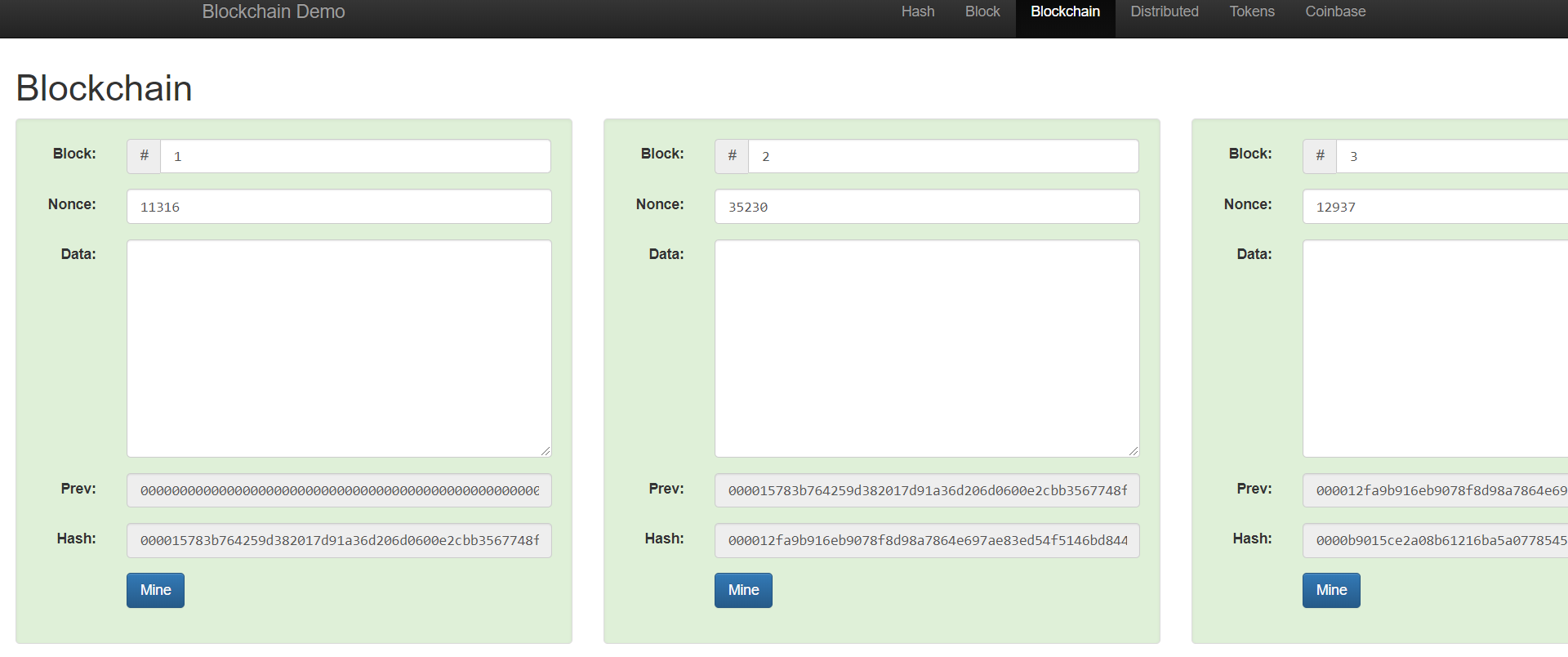
**Q1) Experiment on Andersbrownworth**

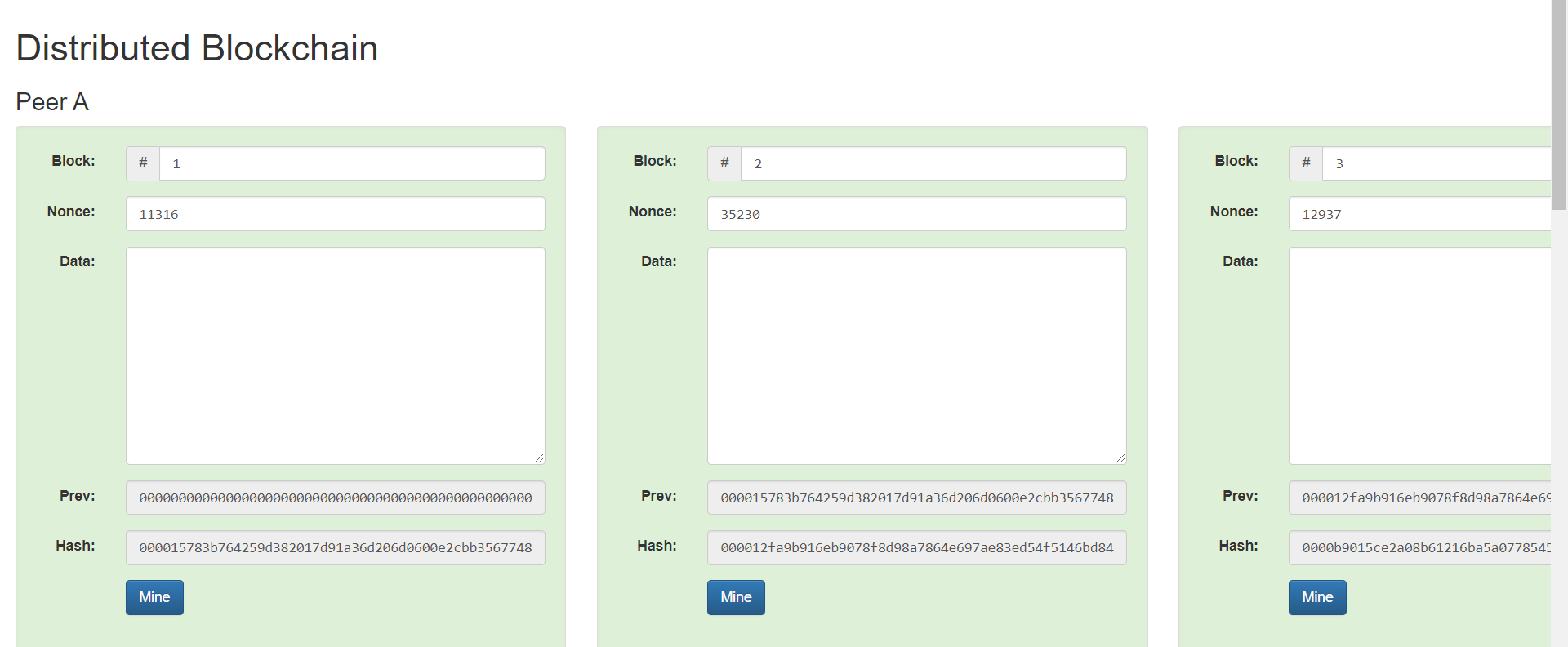
**Screenshot**

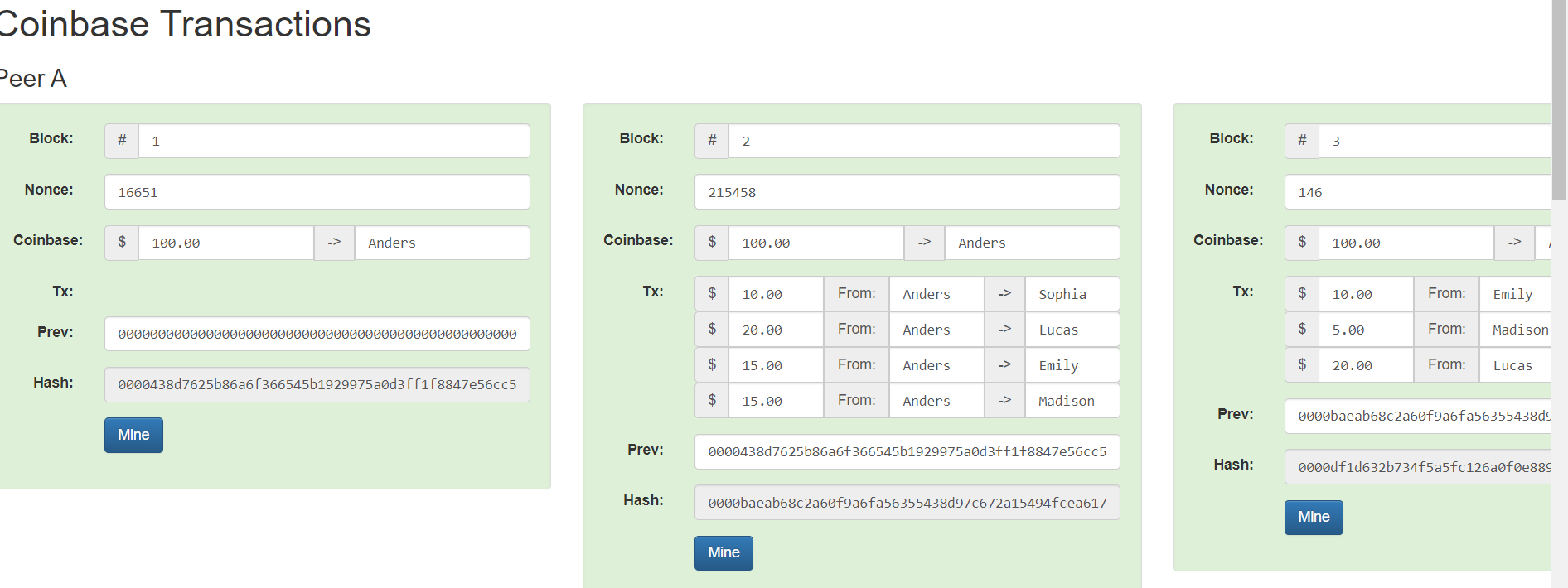












**Q2) Sample Contract example of Owner.sol**

pragma solidity >=0.4.22 <0.7.0;

/\*\*

\* @title Owner

\* @dev Set & change owner

\*/

contract Owner {

address private owner;

// event for EVM logging

event OwnerSet(address indexed oldOwner, address indexed newOwner);

// modifier to check if caller is owner

modifier isOwner() {

// If the first argument of 'require' evaluates to 'false', execution terminates and all

// changes to the state and to Ether balances are reverted.

// This used to consume all gas in old EVM versions, but not anymore.

// It is often a good idea to use 'require' to check if functions are called correctly.

// As a second argument, you can also provide an explanation about what went wrong.

require(msg.sender == owner, "Caller is not owner");

\_;

}

/\*\*

\* @dev Set contract deployer as owner

\*/

constructor() public {

owner = msg.sender; // 'msg.sender' is sender of current call, contract deployer for a constructor

emit OwnerSet(address(0), owner);

}

/\*\*

\* @dev Change owner

\* @param newOwner address of new owner

\*/

function changeOwner(address newOwner) public isOwner {

emit OwnerSet(owner, newOwner);

owner = newOwner;

}

/\*\*

\* @dev Return owner address

\* @return address of owner

\*/

function getOwner() external view returns (address) {

return owner;

}

}

**Screenshot**

