Abstract

The traditional banking and financial systems that are controlled by a single authority such as a bank or government agency is termed as centralized banking. In centralized finance, the assets and services offered are managed by people, and users need to come to terms with that. They charge fees for using their services. A credit card charges from the merchant and sends to the acquiring bank and provides the card details to the card network. The network requests payment from the bank. Each entity in the chain gets paid for its services as merchants pay for the use of credit and debit cards. Based on the reports by public and private banks, fraud cases in ATM/Debit card, credit card and internet banking has increased from Rs 58.61 crore in financial year 19-20 to Rs 63.40 crore in financial year 20-21 due to the cybercrime. Some reasons include loss of proof of transaction, no capacity to store information in little places, complex network due to transfer of money in so many hands, vulnerable nature of centralized network, easy to access credit card credentials. Our project creates a decentralised banking website using blockchain.

Keywords Banking, Blockchain, Security

Table Of Content

Introduction

1. **Introduction:** Decentralized system allows people to engage in trade directly with each other. Instead of relying on the centrally-controlled servers of the companies, a decentralized market operates by employing its users' own computers as the infrastructure. Decentralized finance uses emerging technology to remove third parties and centralized institutions from financial transactions. It also uses security protocols, connectivity, software, and hardware advancements. When there is internet connection, transactions can occur using software that records and verifies financial actions in distributed financial databases or blockchains. Blockchain is used to protect against double spending and modification of previous transaction records. Transactions are recorded in blocks and verified by other users. Once the verifiers agree on a transaction, the block is closed and encrypted and another block is created that contains the information of the previous block. Information in the previous block cannot be modified without affecting the following blocks, so there is no way to alter a blockchain. This concept along with other security protocols ensures the secure nature of a blockchain. Anyone with an internet connection can access the platform and transactions without geographic restriction. Two Parties can directly negotiate interest rates and lend money via a P2P network.

2. **Data Set Description**: Our database is a distributed blockchain database environment known as Ganache.
   The data stored in the Ethereum workspace on Ganache.

- Accounts: The default view showcases accounts created and the balances associated with them.
- Blocks: The blocks page presents an illustration of the details of every block mined on the blockchain alongside the gas fees and transactions involved.
- Events: The page includes a list of all the events started since the creation of the workspace, and Ganache works on decoding the events facilitated by contracts in the project.
- Transactions: The transaction page, as the name implies, points at a list of all transactions operating against the blockchain network.
- Logs: The Ethereum workspace on Ganache smart contract testing blockchain would also feature an outline of the server logs, which help in debugging.
- Contracts: The Contracts page outlines all the contracts included in the Ethereum workspace.
  Initially, few accounts are pre-existing with 100 ETH in their bank account for each. But with creation of new accounts and transfer of money the data grows.

3. Data Flow Diagram:

Register for an Account

Every bank system has a unique account system. The first step of the proposed system is to create an accounting system for ether based on transactions. In this automated banking system, users will only give their ID when using a web banking system or mobile banking system to register for an account in the smart contract. The smart contract then creates an account for you. Now a user can use his/their account for any transaction with the bank or smart contract. After getting registered, the user can deposit ethers into their account and also withdraw money from their Ethereum wallet using the smart contract, which is illustrated using a flow in Fig 1.

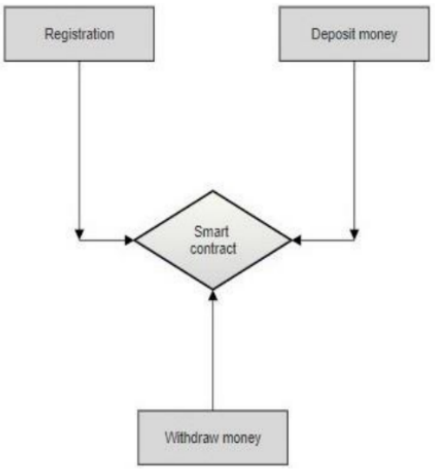


Fig 1. Functionalities of smart contract

Deposit or Transfer *Money*

Using a web banking system, users can log in to their Ethereum wallet account. After that, the user will deposit in their own bank or send a certain amount of ether to other user's bank by using the Ethereum network to the smart contract or bank. The bank or smart contract will calculate the Ether amount and transfer that amount of Ether to the user's bank account. The

workflow representing depositing money is depicted using Figure 2.

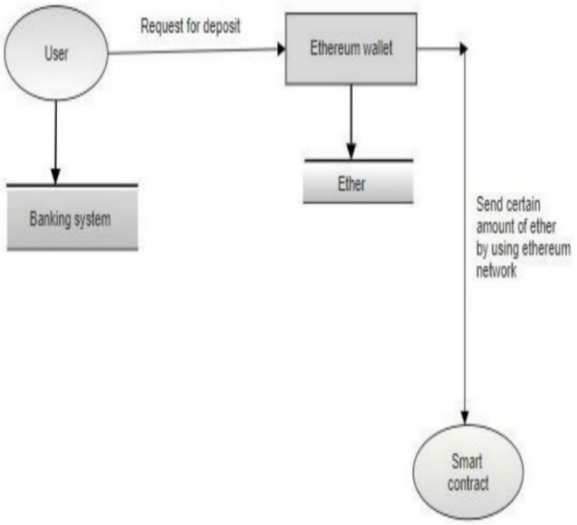


Fig 2. Depositing Money

Withdraw Money

A system user requests that the smart contract withdraw a certain amount of ethers from his/her wallet. Upon receiving the request, the smart contract will transfer the requested amount from the user's wallet to the smart contract's address. After successfully transferring the funds, the smart contract requests that the authorized personnel calculate the corresponding paper money of that ether amount and send it to the system user. The procedures for completing the process of money withdrawal are illustrated using Fig 3.

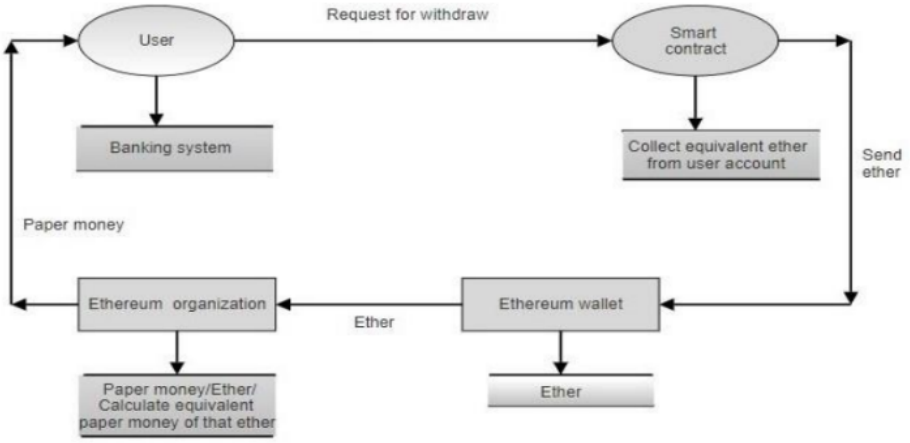


Fig 3. Withdrawing Money

Create or View Accounts

Using Internet-banking system, user can create multiple accounts and check account status. This service allows users to check their account balances. This is basically a user interface that works as a client, creating a link between the system's smart contract and the user

commands. The flow chart to demonstrate the internet banking app's workflow is represented using Fig 4.

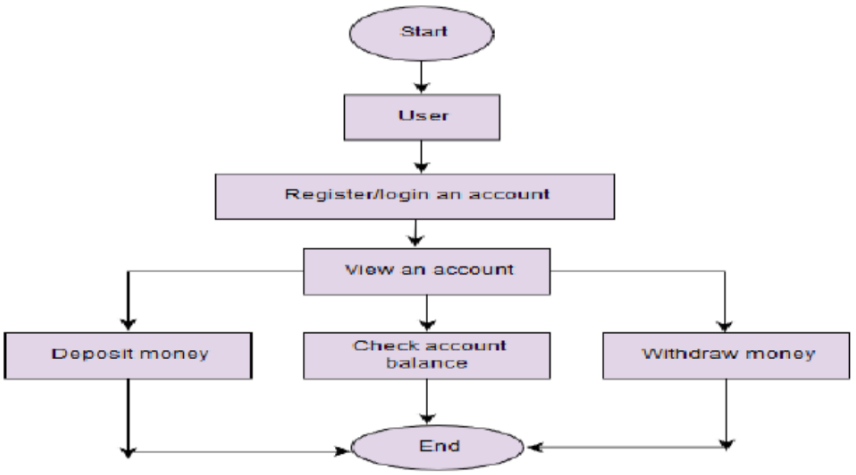


Fig 4. Flowchart of proposed banking system

Taking Loans

A system user can take loans from the bank which will be added to his wallet which is similar to withdrawing. Then he has to return the money back to the bank by depositing into the bank in instalments.

Literature Survey

In this paper [1] blockchain system is implemented by Ethereum that's why it is very effective for the transaction system than the other transactions (i.e Siacoin, Monero, Bitcoin) because it has smart contract technological features which have connected to the past transaction with present transaction. Besides, Ethereum can easily develop an application as well as is public, open-source and Blockchain-based software platform that allows developers to build and deploy decentralized applications. It has no central point of failure, as it is being run from thousands of volunteer computers around the globe, which means it can never go offline

METHODOLOGY:
In this, the authors propose the following implementation:

- The Back-end of the proposed system consists of a smart contract. Starting from registration, depositing money, sending money, earning interests and all of these tasks will be managed by the contracts.
- 'Mocha' testing framework is used to test whether smart contracts are functioning properly.
- 'Solc' compiler is conducted to compile the contracts and JSON files are created corresponding to that smart contracts.
- The system is firstly implemented on Ethereum Test Network which is known as Rinkeby. After signing up , The byte codes and ABIs of the smart contracts located in JSON files are then deployed to the URL of the Rinkeby Test Network by Truffle-HD-Wallet-Provider and web3.
- An interactive front-end (GUI) is developed using React.js. MetaMask Extension is used to access the implemented Ethereum enabled DAPP from the browser.

In this paper [2], they proposed a new banking system based on ethereum blockchain technology that is automated and has the total elimination of authoritarian interception. They also developed a smart contract that controls our system's procedures, and transaction details are stored inside the blocks. This system can perform four functions: account registration, depositing funds, withdrawing funds, and internet banking. This banking system has no room for fraudulent activities that cause financial loss for its users. They claim this automated banking system, no one can modify any financial transactions. Users can execute transactions from their homes and workplaces using the proposed system within moments.

Their system include:
1. Register for an Account: The first step is to develop an accounting system for ether based on transactions.

2. Deposit Money: Using a mobile banking system, the user will send a certain amount of ether to the smart contract or bank. A bank or smart contract will calculate the equivalent ether and transfer that ether to the user account.
3. Withdraw Money: The user will request a certain amount of ether to be withdrawn. The bank or smart contract will calculate the amount of ether anyone wants to withdraw and transfer those ethers from the user account to their Ethereum wallet using the Ethereum network.
4. Internet Banking: Using the internet banking system, the user will create an account and view the account. This service will enable users to view transaction history and updates.
In this automated banking system, no one can modify any financial transaction. Users will not have to wait a long time as they can perform any transaction from their home and workplace using this ethereum-based application.

Implementation

4. System Requirements:

The following programming languages were used to construct this financial system:

• Solidity in the design of smart contracts

• For communicating with the smart contract, use JavaScript

The following frameworks and libraries are used:

• Ganache library to build the local test network

• React.js is a front-end development framework

• Web3 collection for libraries

The following browser extension is used:

• For accessing ethereum-enabled distributed apps, we have used the MetaMask

Hardware requirements:-

- OS: Fedora linux
- Minimum 512 Mb RAM
- 1Gb hard disk space

5. Implementation

A Smart contract is included in the system's backend. This contract was created using the Solidity programming language. This system's front-end interfaces with smart contracts through contract addresses and bytecode. React.js is used to create an interactive front-end. The Solc compiler is used to compile the Smart Contract and JSON files are created corresponding to those Smart Contracts. The bytecodes and ABIs for the smart contract are located in the JSON file. The JSON file is then deployed to that URL on the Custom Banking test network by Truffle-HD-Wallet-Provider and web3. The Smart Contract address is returned and kept once it is deployed to the Banking test network. Web3 connects the Banking test network and the front-end. It acts as a link between the test network and the DApp established. The implemented ethereum-enabled DApp is accessed using the MetaMask browser. Manage the account and MetaMask does all the transactions. The overview and workflow of the proposed system maintained by the smart contract is illustrated using Fig 5.1.

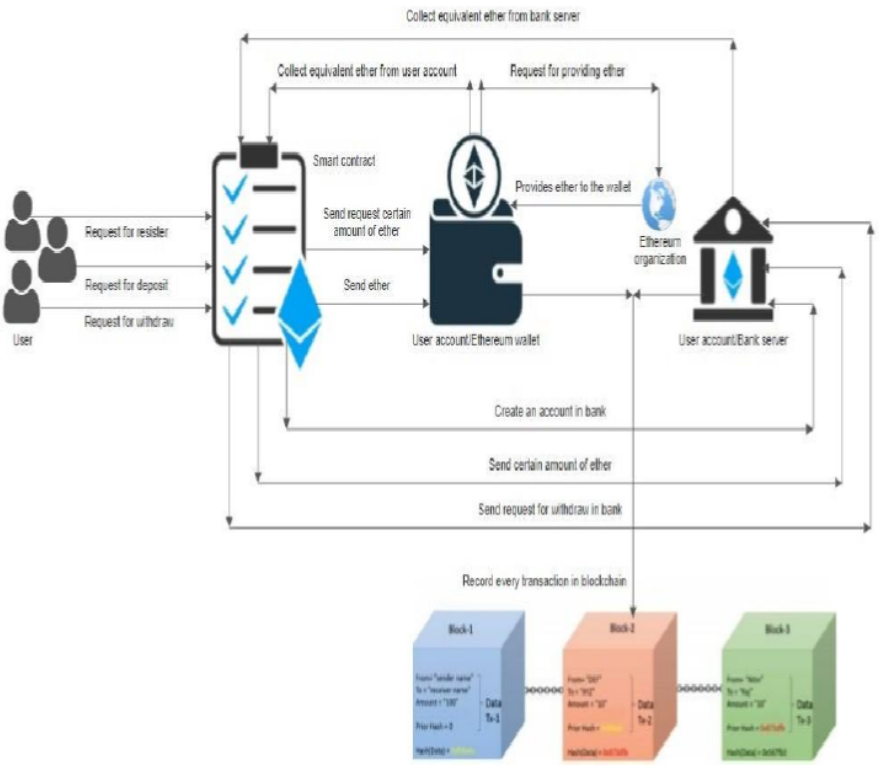


Fig 5.1. Architecture of the proposed system

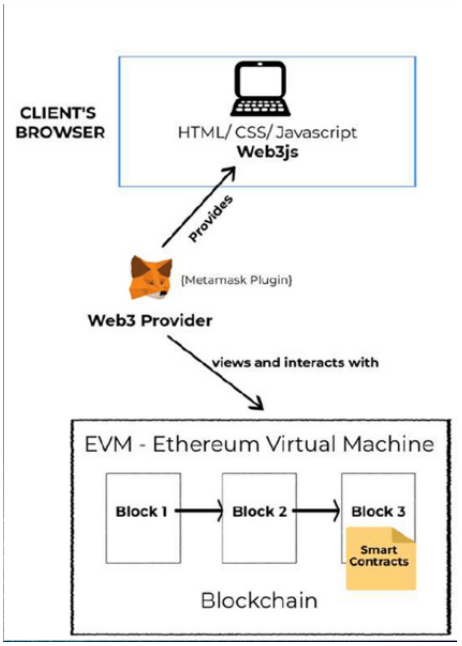


Fig 5.2. The proposed system implementation

Solidity Language for Backend

Our Smart Contract is written using the Solidity programming language. Most of

the algorithms of the system are in this contract. registration.sol is used to manage the banking system. This contract performs registration, depositing, and withdrawing money using an Ethereum wallet.

Front End using React js

React.js is a well-known JavaScript framework for creating dynamic front-ends.
The main reason for using React.js for the front end is that it makes creating complex web apps more accessible. The user interface (front-end) created by React.js for the system provides the ability to register an account, deposit, and withdraws money.

Metamask

MetaMask is a gateway that lets you experience the future's scattered web in your browser right now. It allows you to run ethereum DApps in your browser without running a full ethereum node. MetaMask has a secure identity vault that lets you save your identities across several sites and sign blockchain transactions using a user interface.

Smart Contract to process the event

The below contract function performs registration. It will create a new structure, for instance, for a bank user to get registered. To create an ethereum based bank account, users will require a minimum of 10 ether in their ethereum wallet. Registration will be done after meeting the criteria.
The function deposit will just transfer ether from the user's wallet to the bank wallet, and whenever a problem exists, the ether is refunded to the user's wallet. The function withdrawal will just check the amount of ether to be withdrawn and perform the withdraw operation. When this data changes, it will create a transaction, and all the functions can be accessed publicly. The functions getRegistered, getBanksAccount, and getDepositAmount keep records of the user's existence and changes in their profile. Fig 6 represents the algorithms for account registration, depositing ethers and withdrawing credits from the accounts.

Fig 6. Pseudocode for Registration, Deposit and Withdrawal process

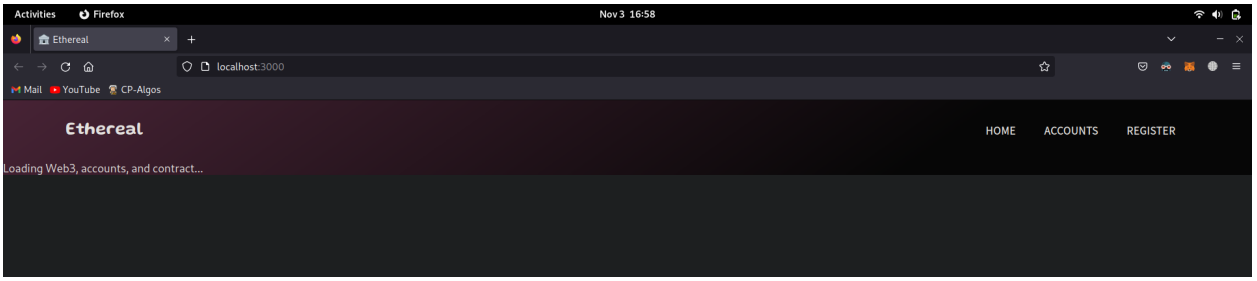
```
contract Registration
{
    address[] public registered;
    address public banksWallet;
    mapping(address => uint256) public deposit;

    procedure registration (address ba) public payable
    {
  banksWallet <= ba;
    }
    procedure performRegistration () public payable
    {
        require(msg.value is GREATER THAN EQUAL 10 ethers);
        registered.push(msg.sender);
        banksWallet.transfer(msg.value);
```

```
        deposit[msg.sender] <= msg.value;
    }
    procedure deposit () public payable
    {
        uint256 flag <= 0;
        FOR i <= 0; i is LESS THAN registered.length; i++ DO
          {
        IF msg.sender is EQUAL TO registered[i]
            THEN   flag <= 1
             }
            IF flag is EQUAL TO 1
           THEN
         banksWallet.transfer(msg.value);
                deposit[msg.sender] <= deposit[msg.sender] + msg.value;
            ELSE
                msg.sender.transfer(msg.value);
        }
       procedure withdrawal (uint256 amount) public payable
        {
            uint256 flag <= 0;
              FOR i <= 0; i is LESS THAN registered.length; i++ DO
                 {
                     IF msg.sender is EQUAL TO registered[i]
                     THEN
                           flag <= 1
                 }
                IF flag is EQUAL TO 1
                  THEN
                       amount <= amount*1000000000000000000;
                       IF deposit[msg.sender] is GREATER THAN amount
                          THEN
                           msg.sender.transfer(amount + msg.value);
                             deposit[msg.sender] <= deposit[msg.sender] - amount;
                            banksWallet.transfer(this.balance);
                   }
                   }
         }
}
```

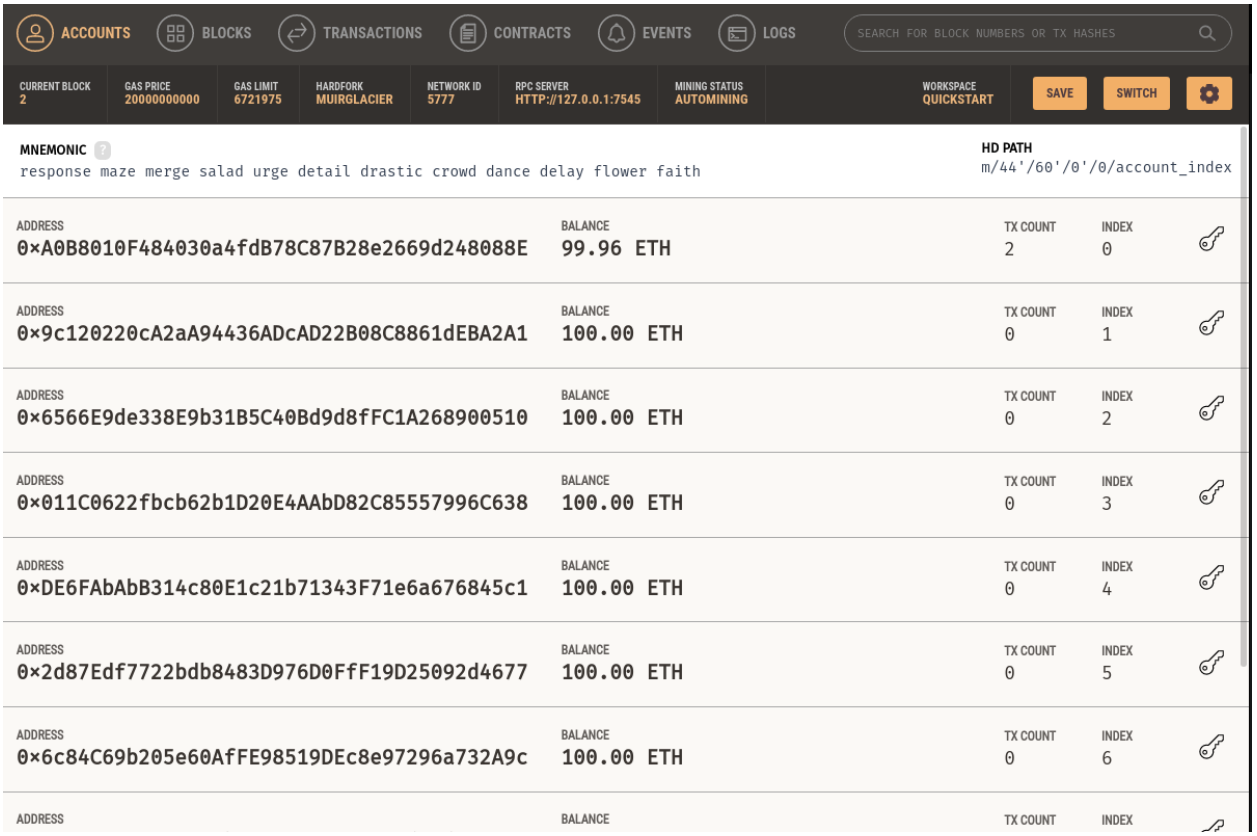
Results and Discussion

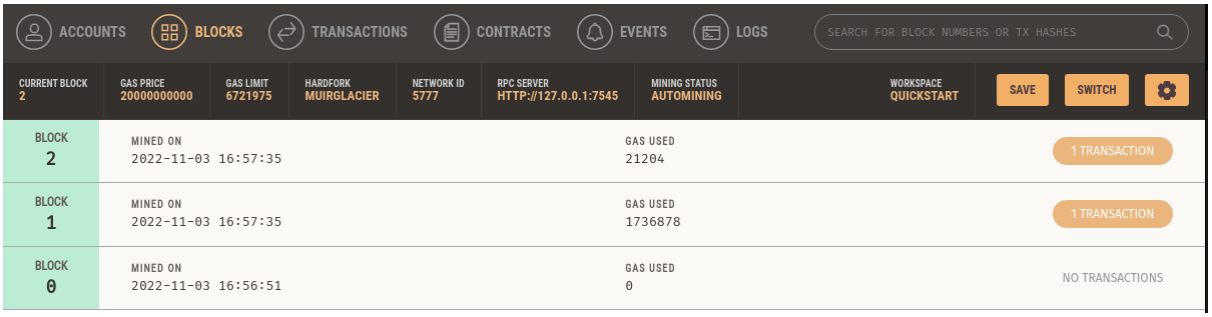
6. Results and Discussion

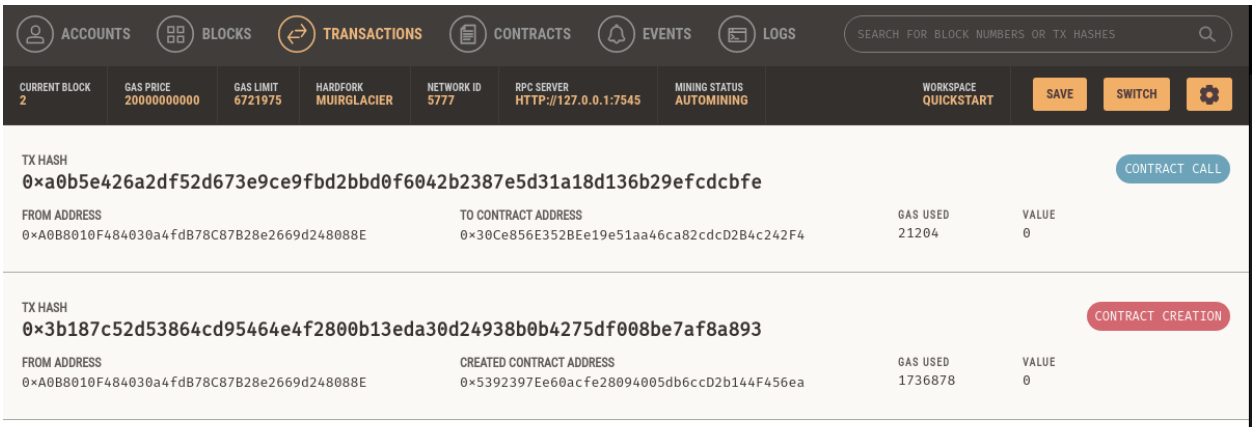
When no accounts are loaded:



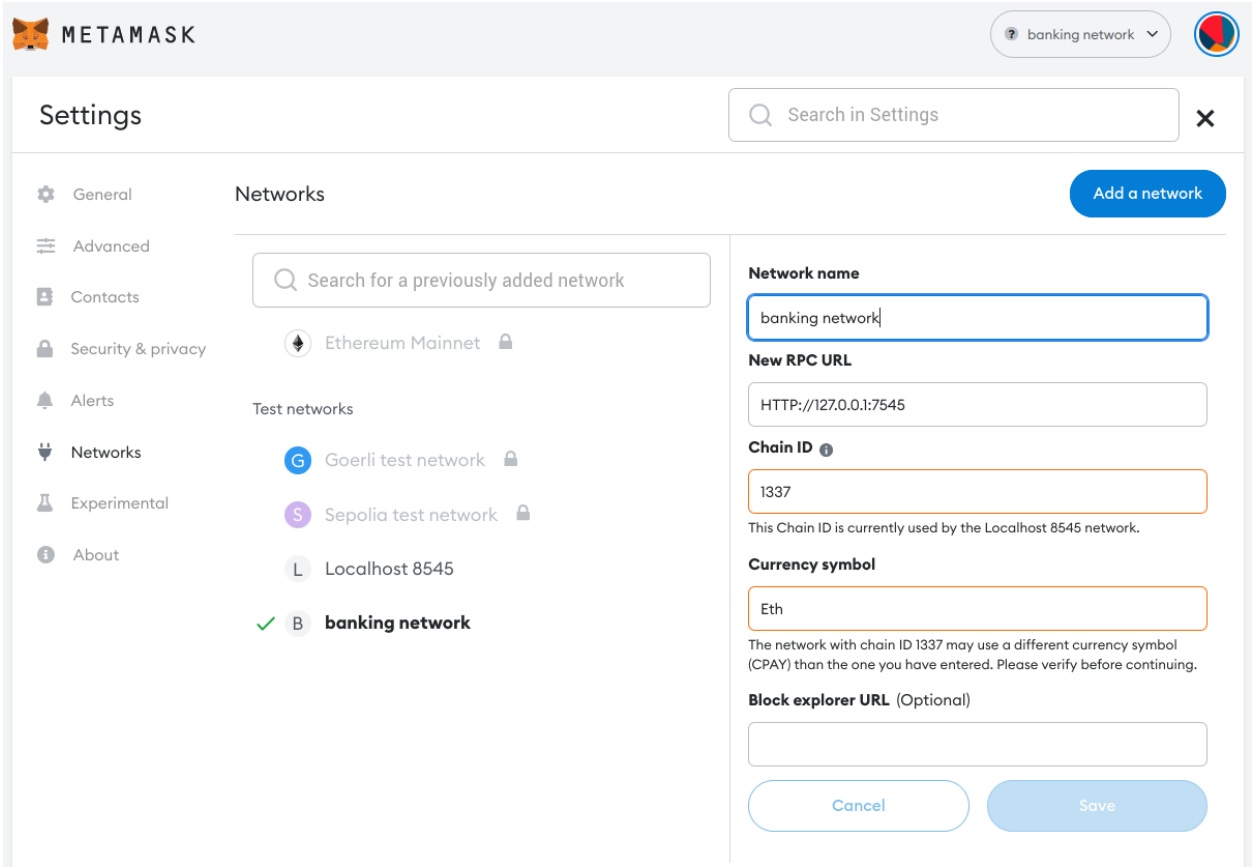
Ganache Before Connecting to an account :



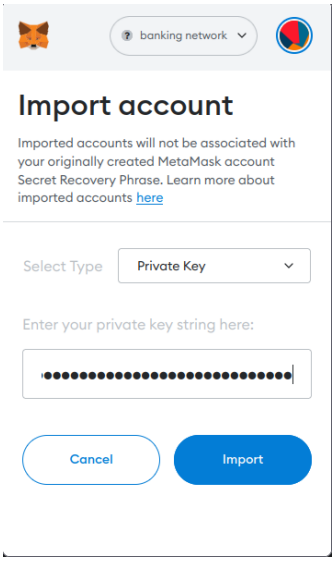


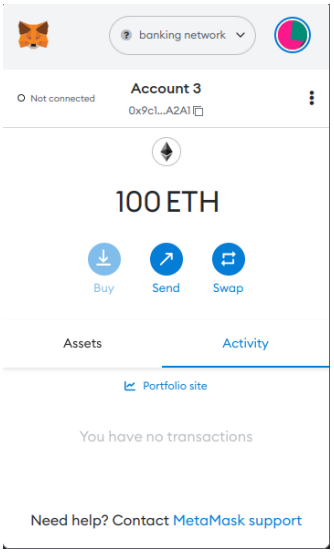


Local Banking Network Stats:

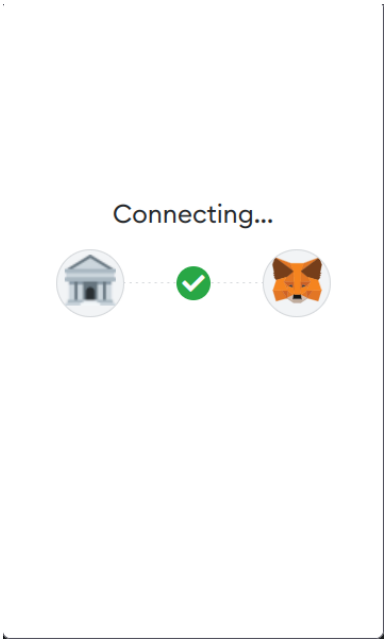


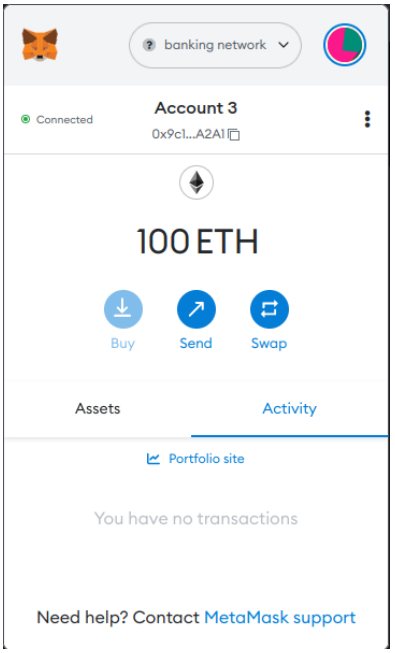
Import to an account to metamask:



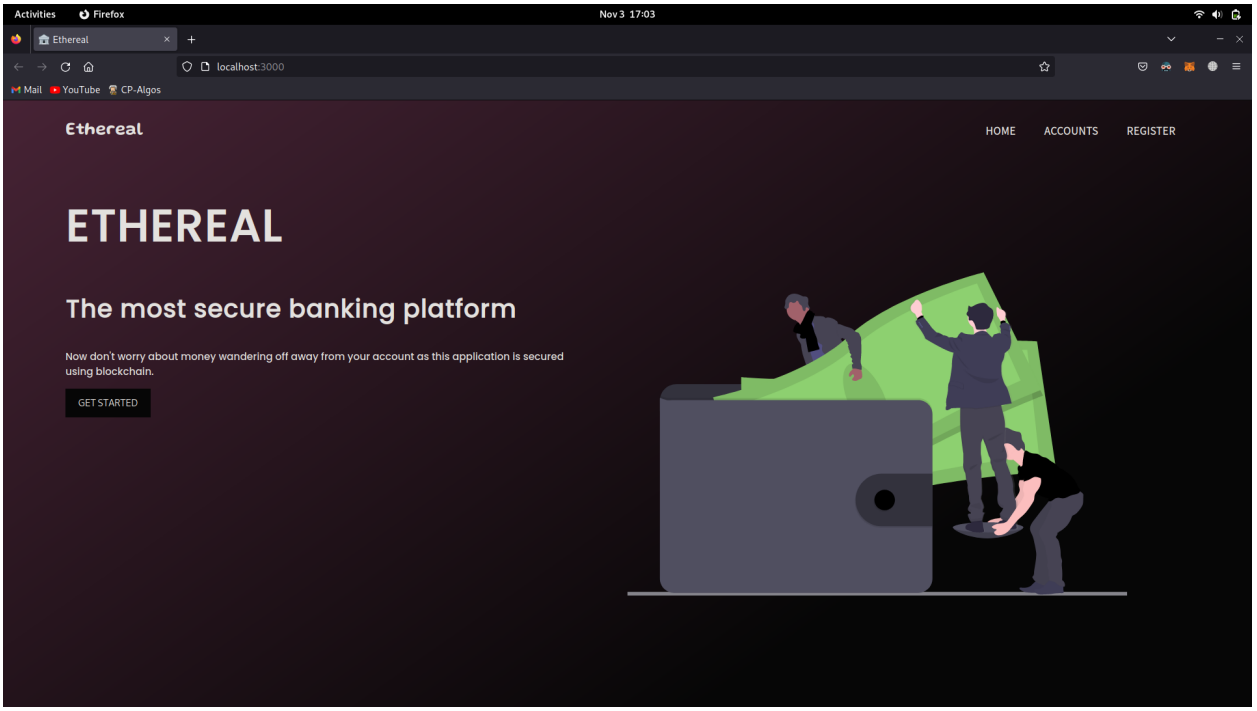


Connecting to the current website:

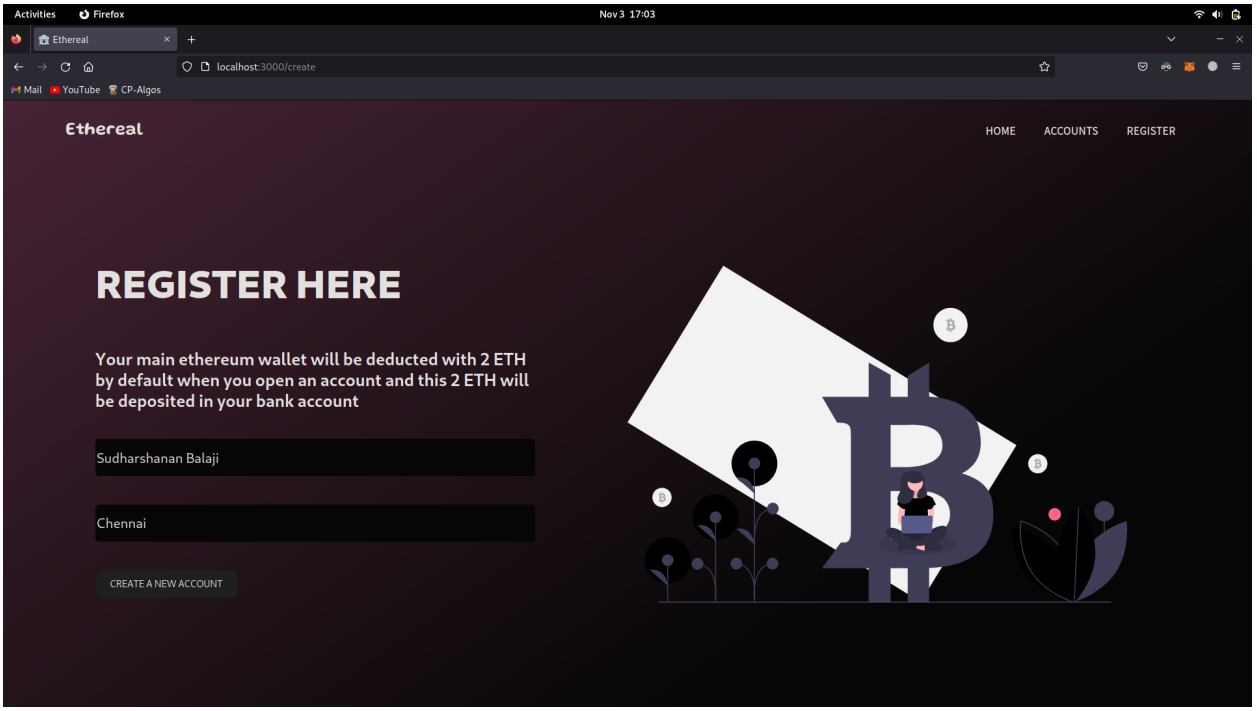




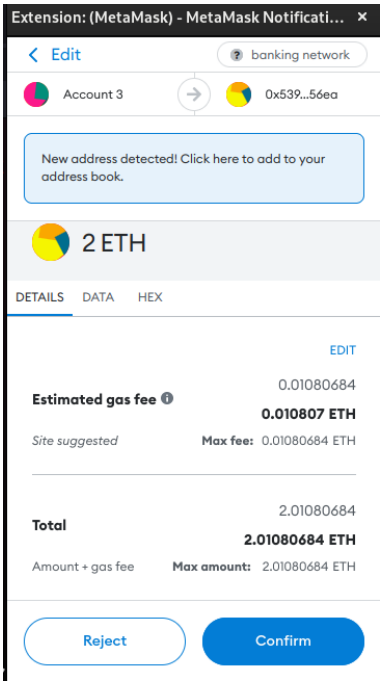
Website after connection:



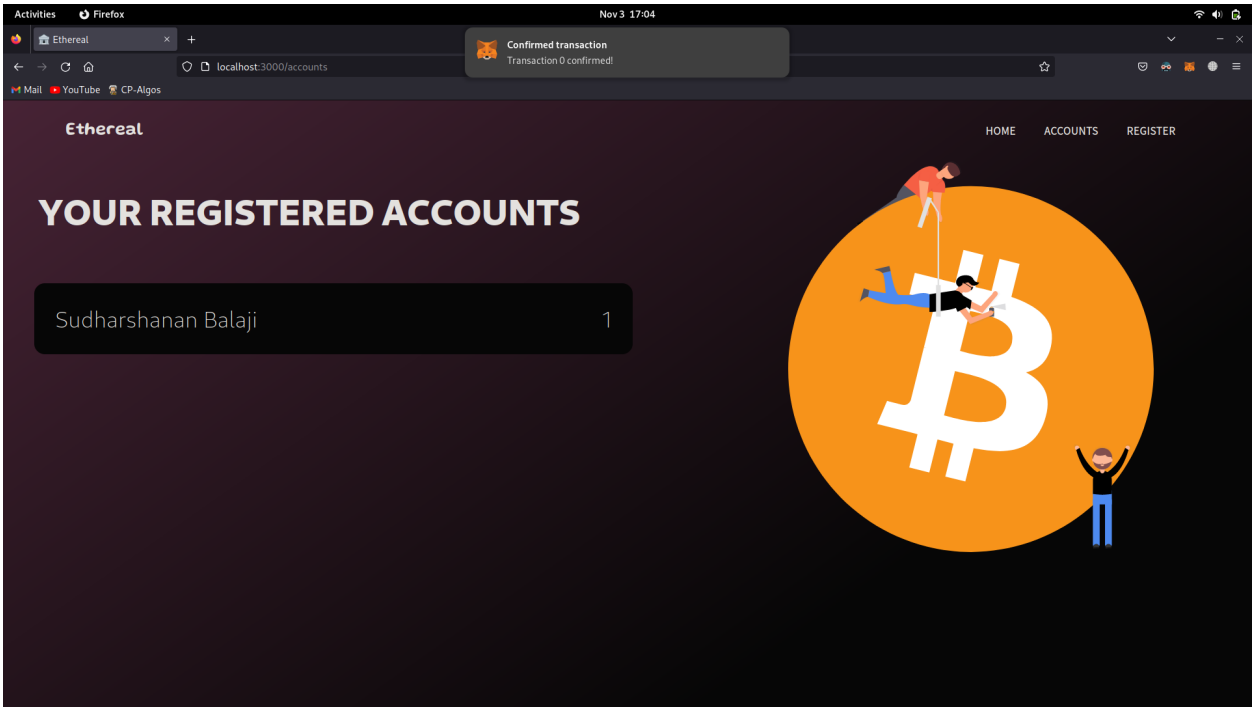
Registering Account:

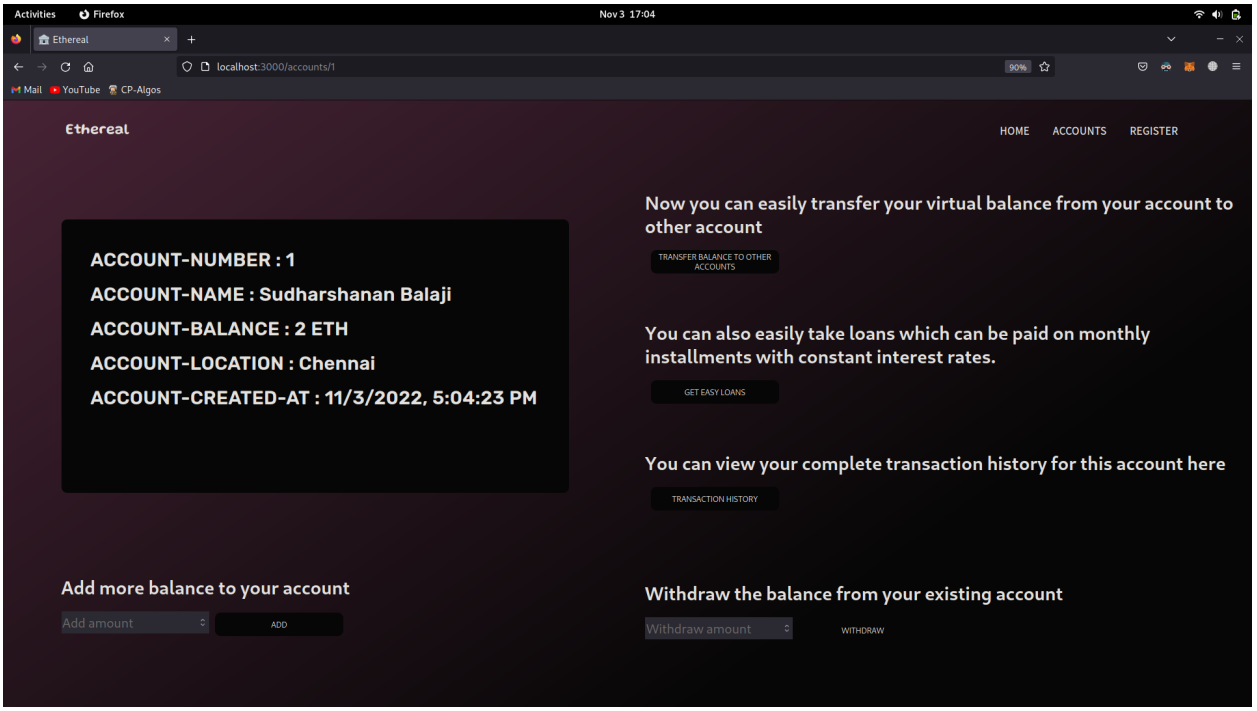


Metamask Prompt for 2ETH Approval from Personal Wallet:

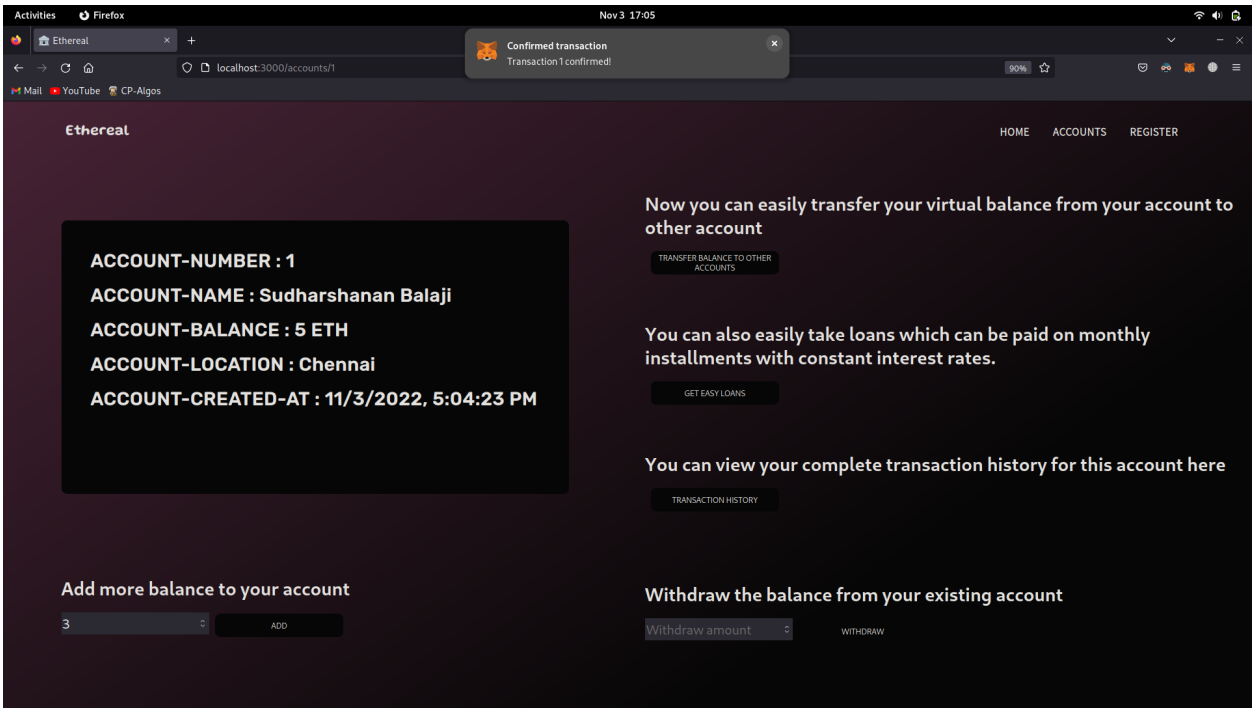


Displaying the Registered Account:

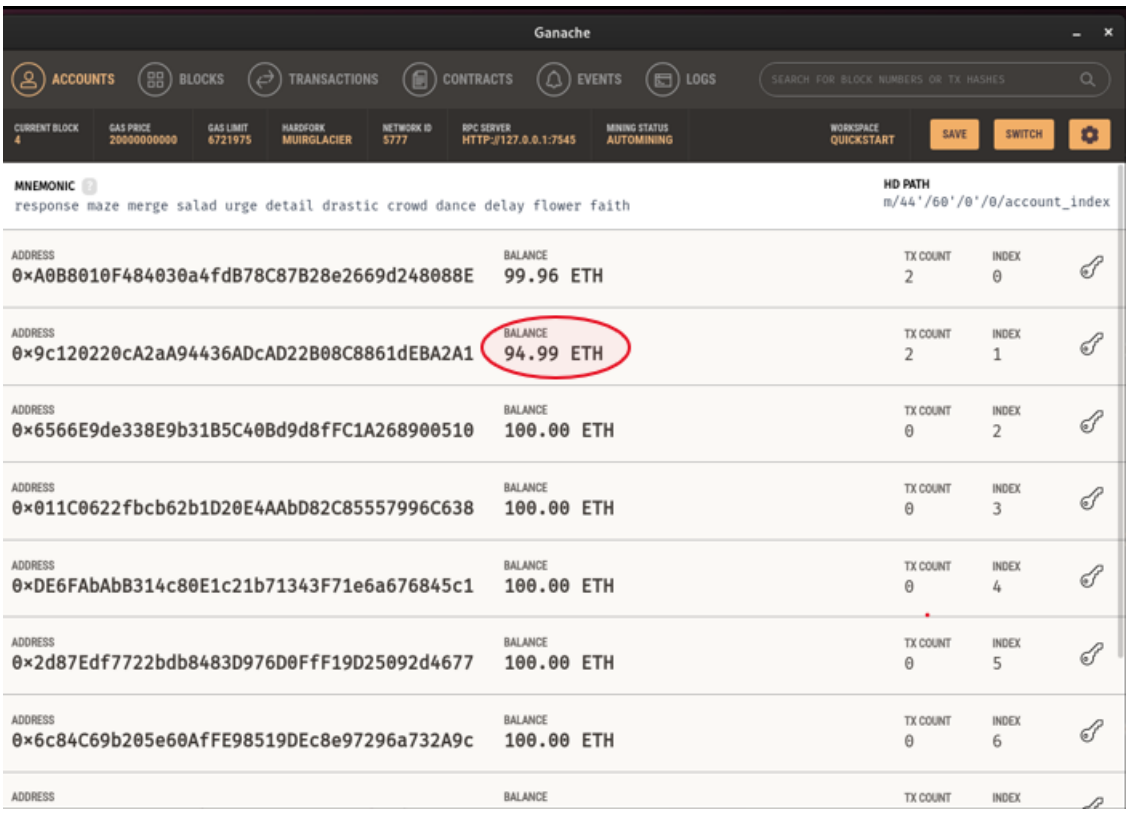




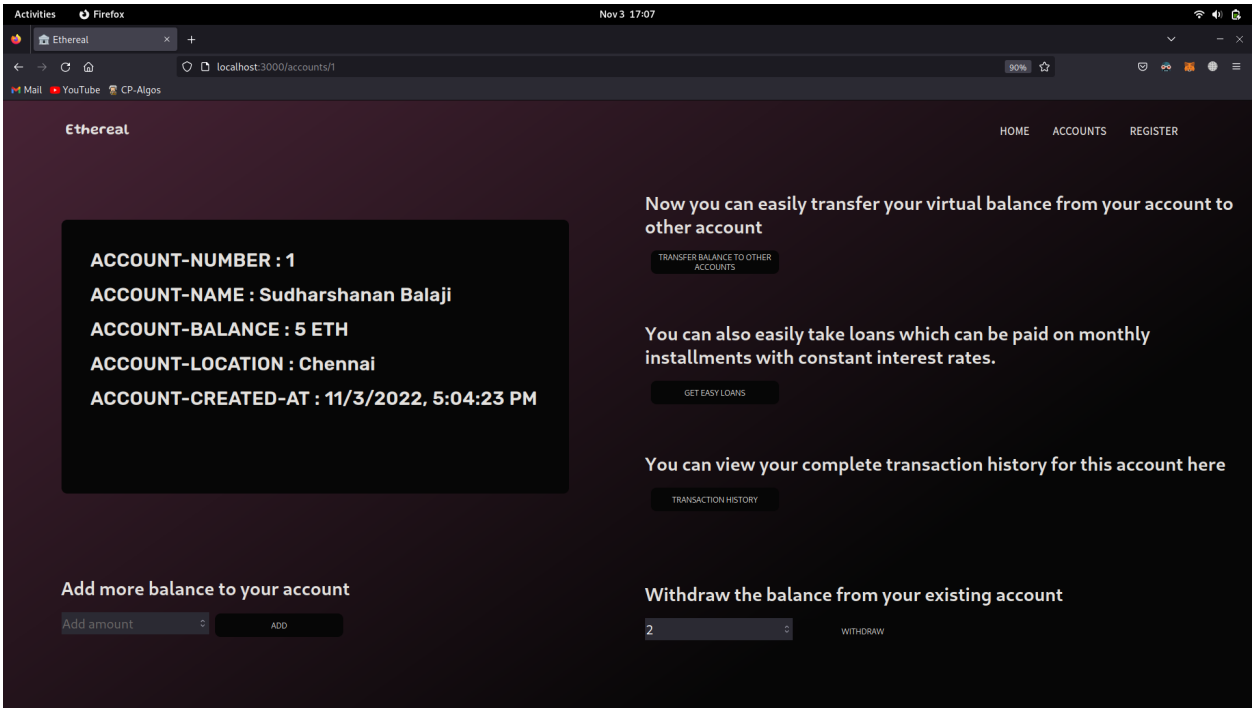
Adding An Amount to the bank:

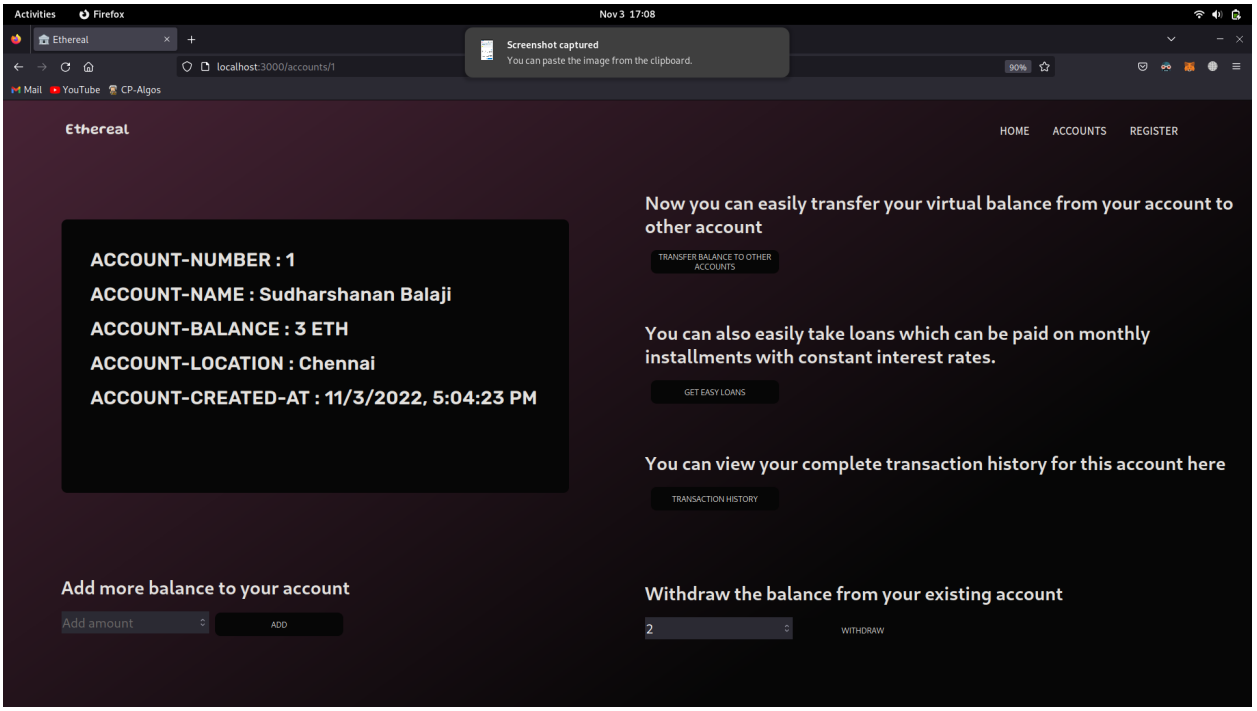


Reflection in the Ganache Wallet(5.01 Eth reduced) - Balance = 94.99 ETH

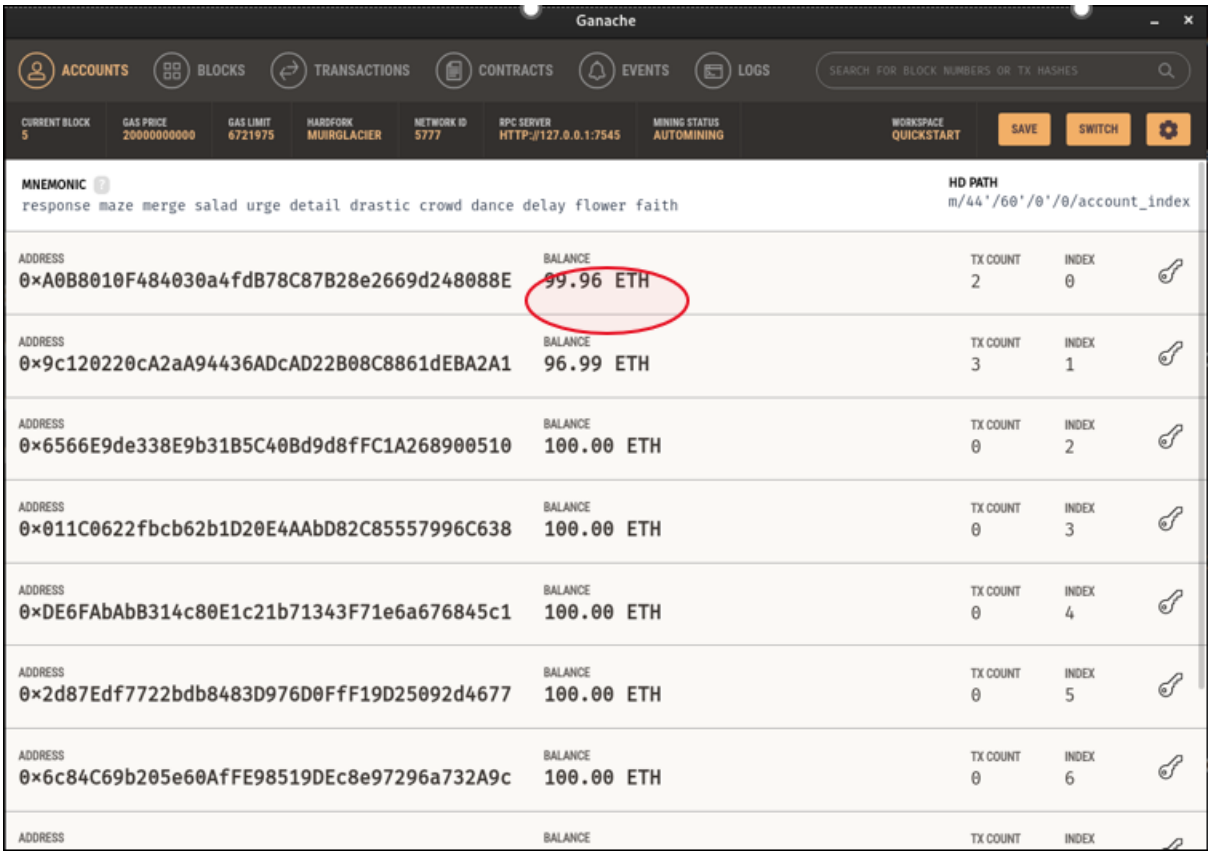


Withdrawing Amount:

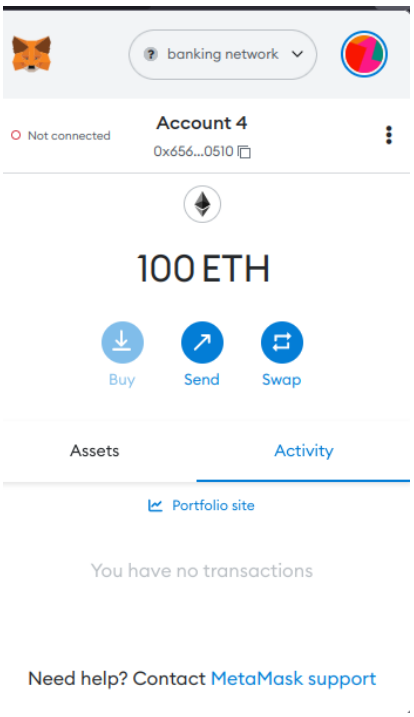


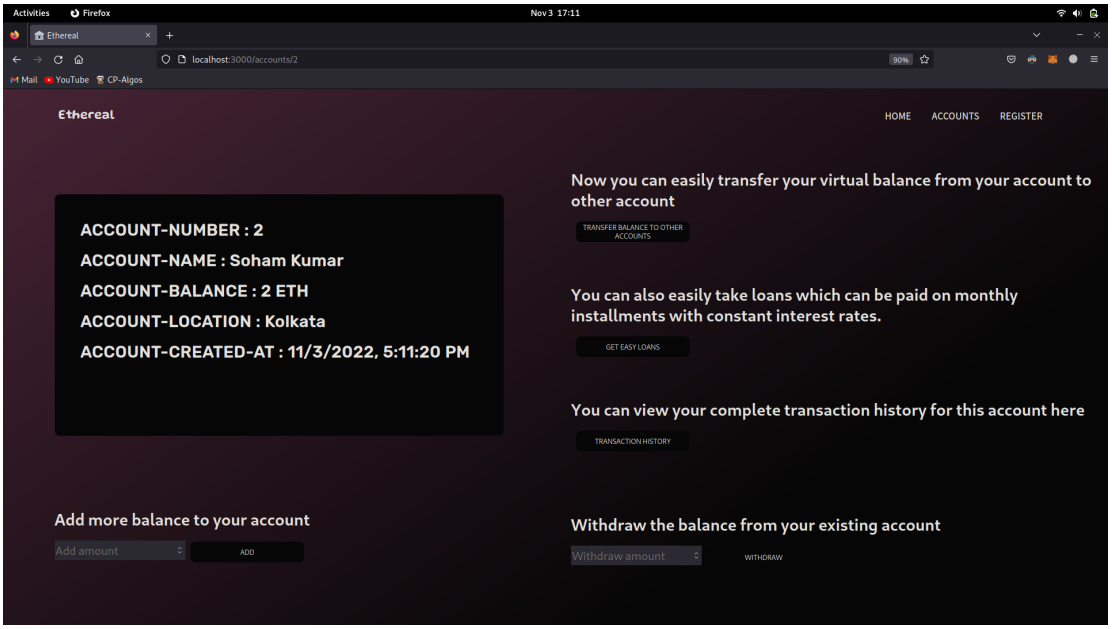


Reflection in the Ganache Wallet(2 Eth addedd) - Balance = 94.99 + 2 = 96.99 ETH

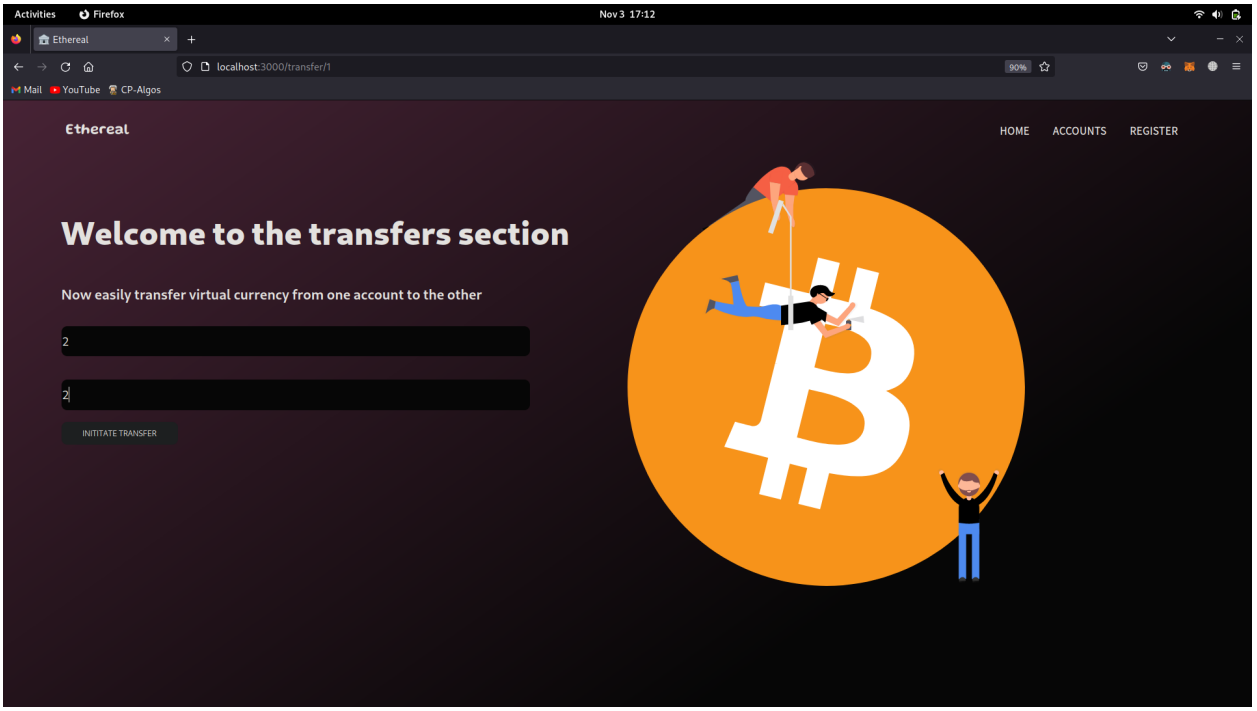


Creating One More Account to transfer Amount:

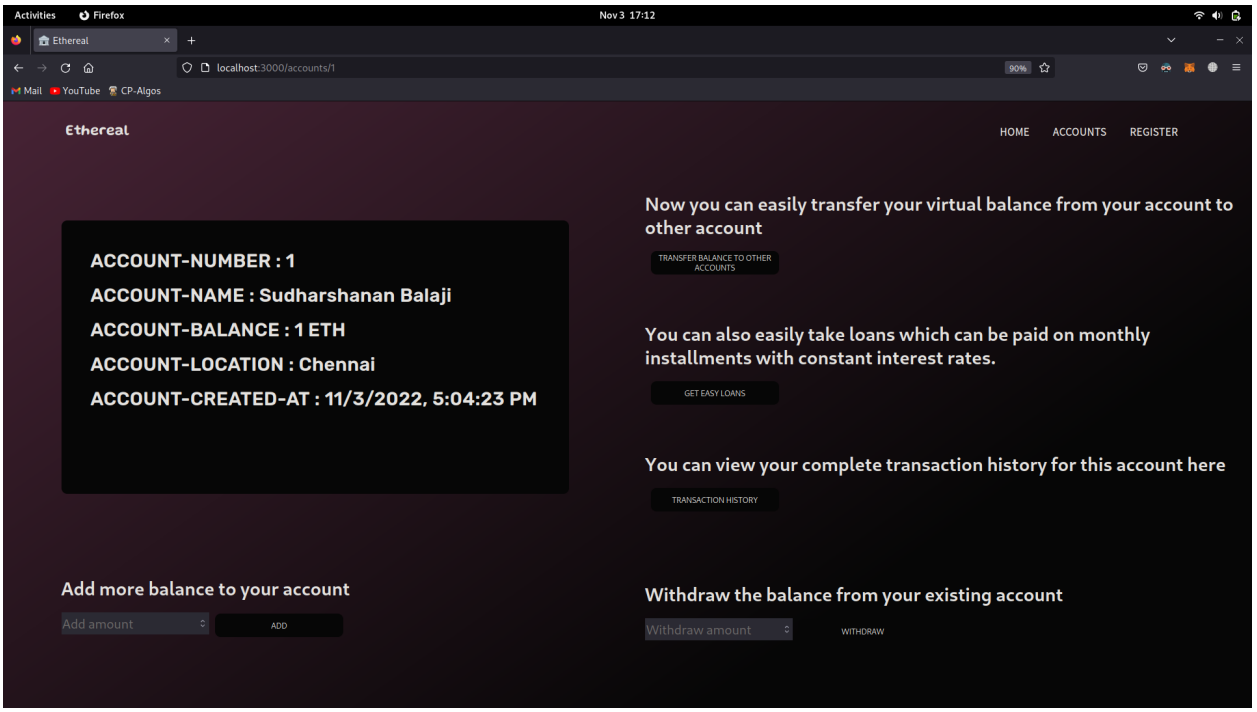




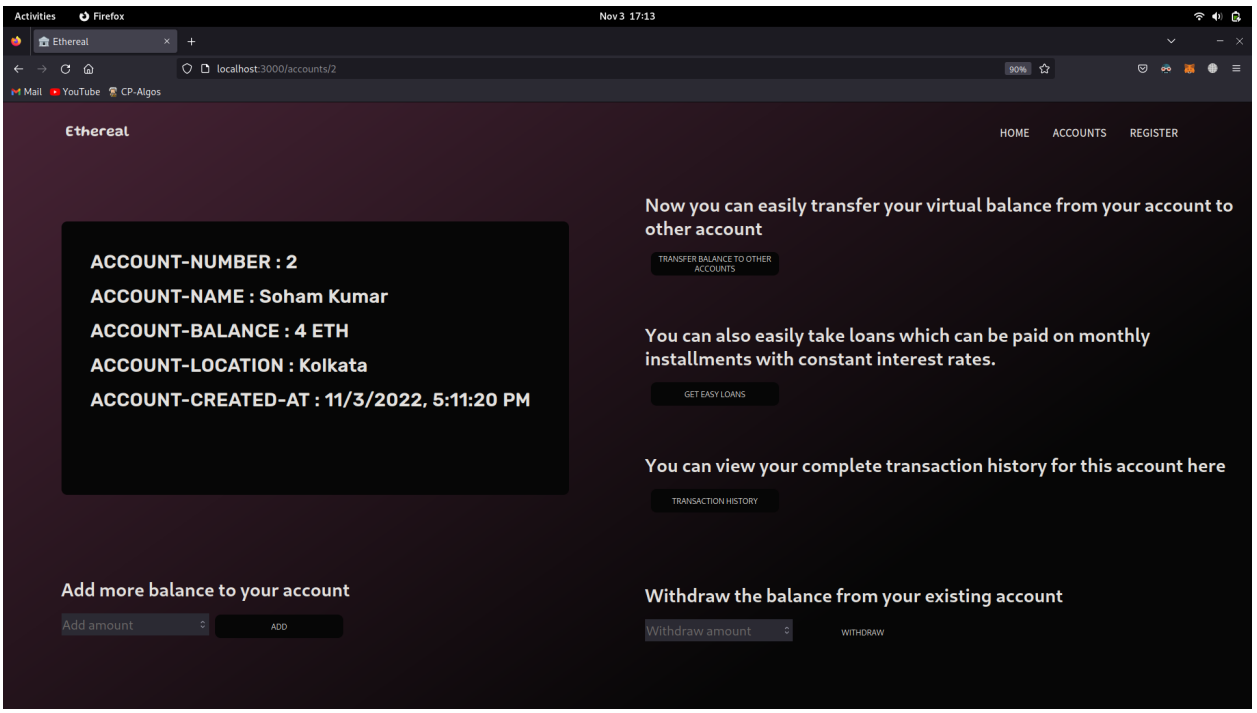
Transferring Funds from Account No 1 to Account No 2:



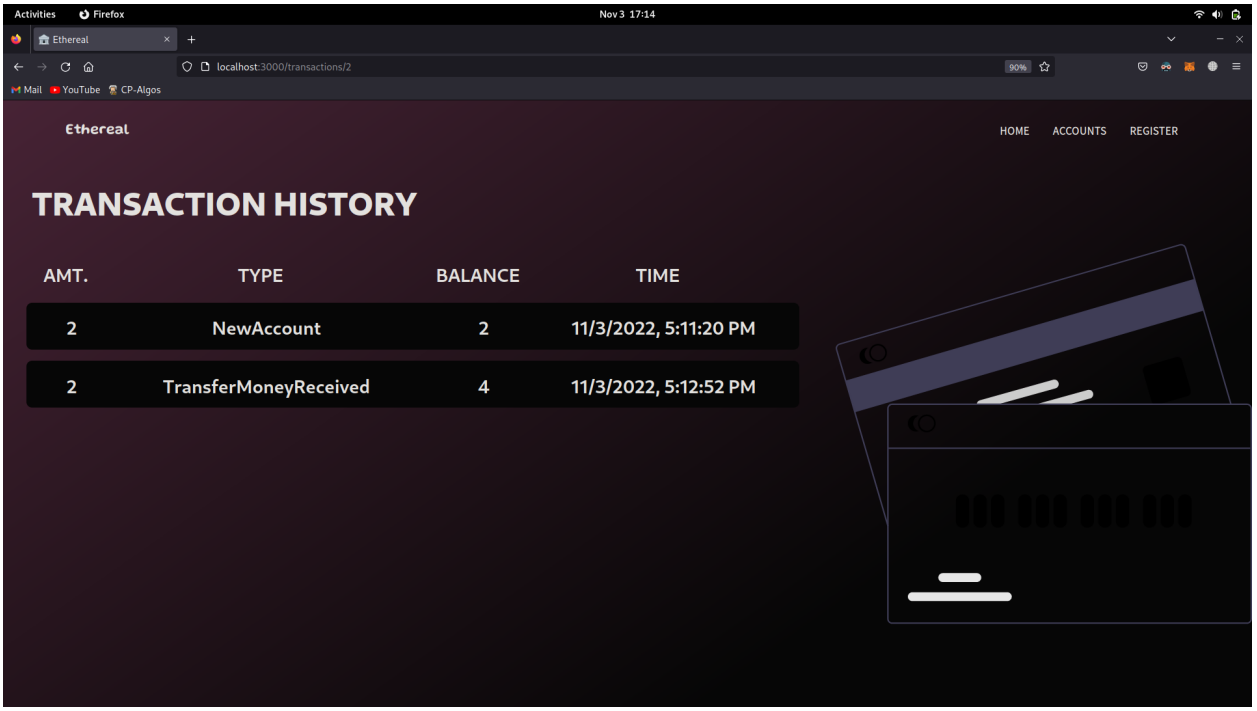
Reduction of 2ETH from account 1

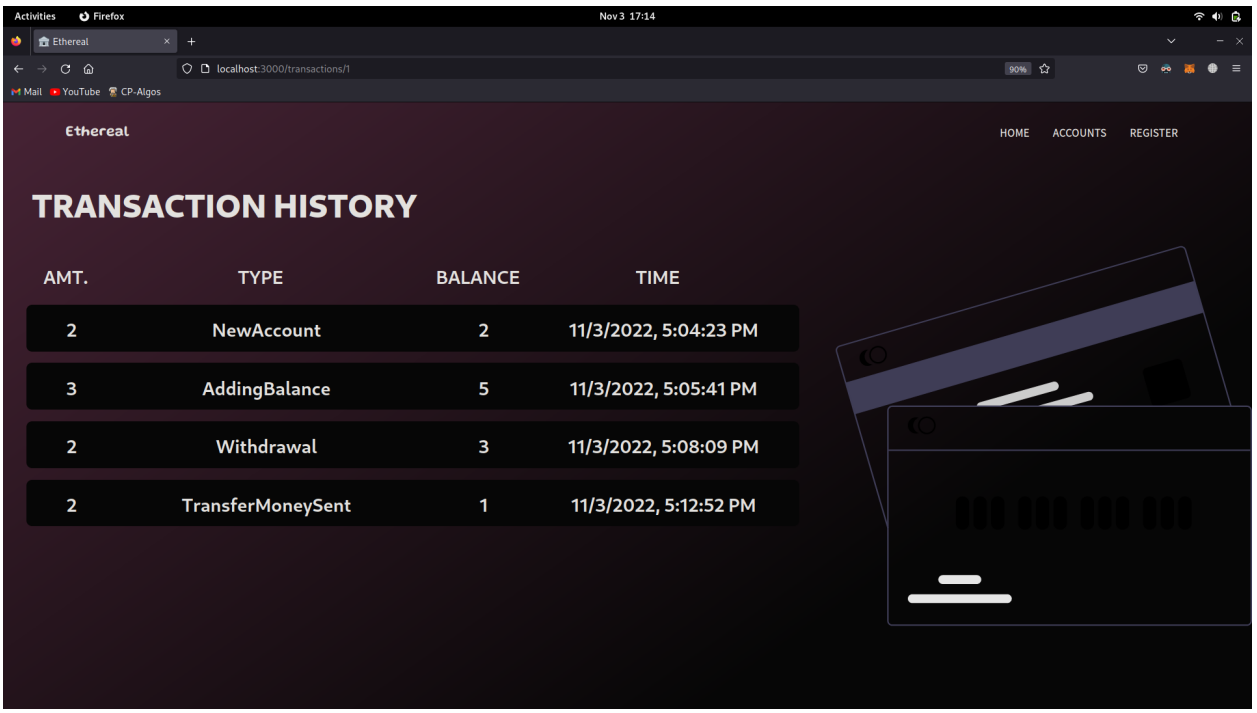


Addition in Account No.2

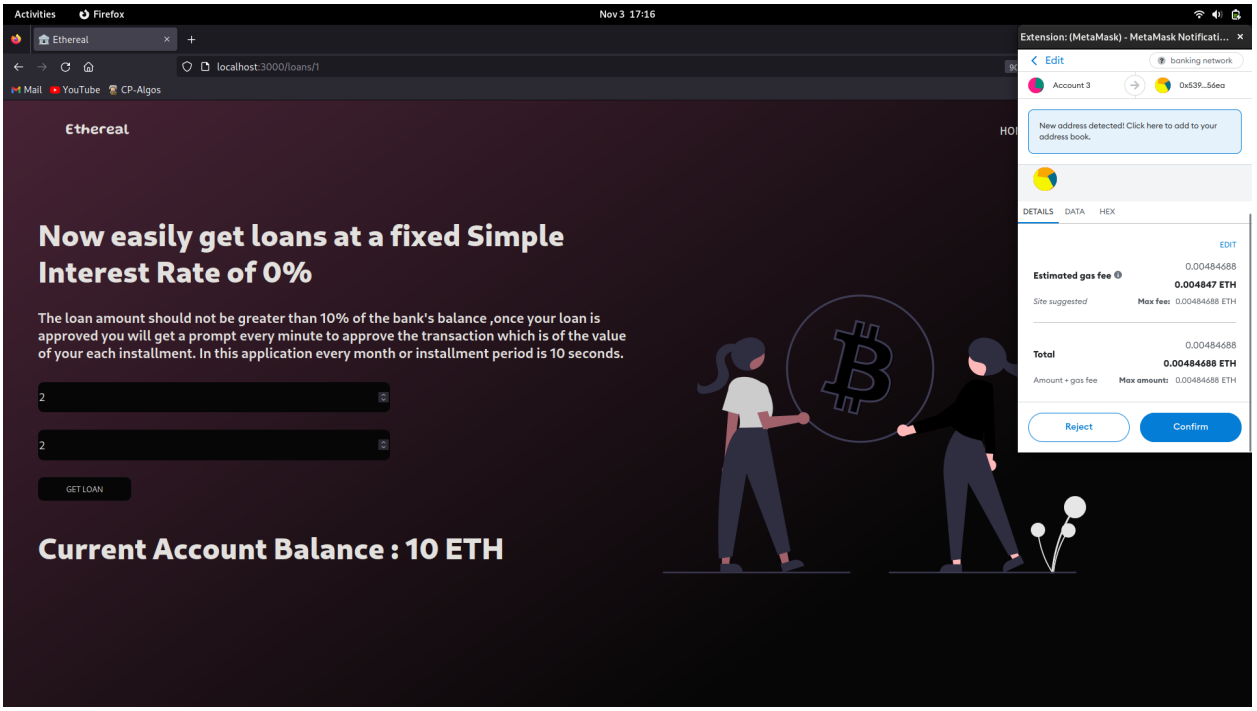


Displaying Transaction History:

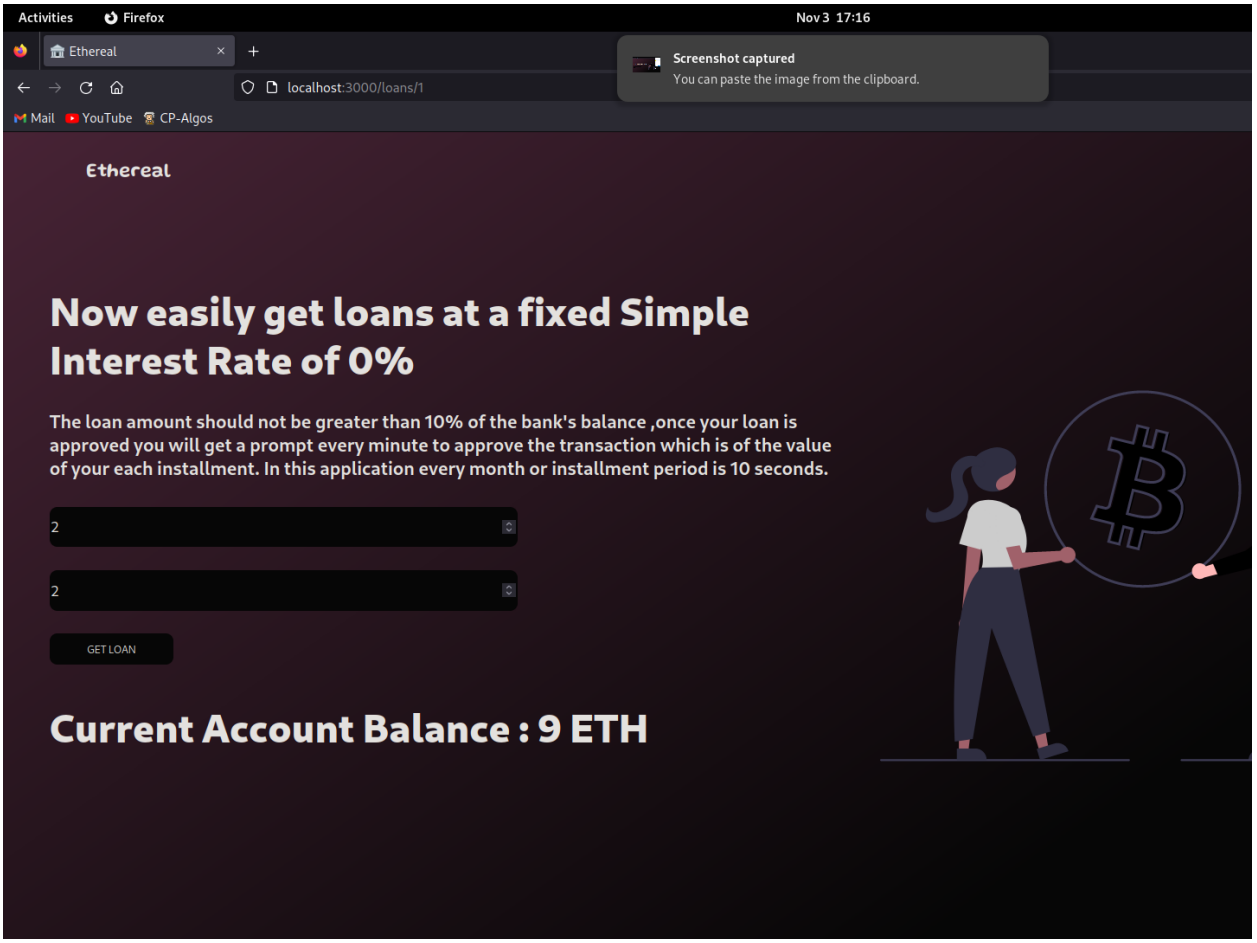




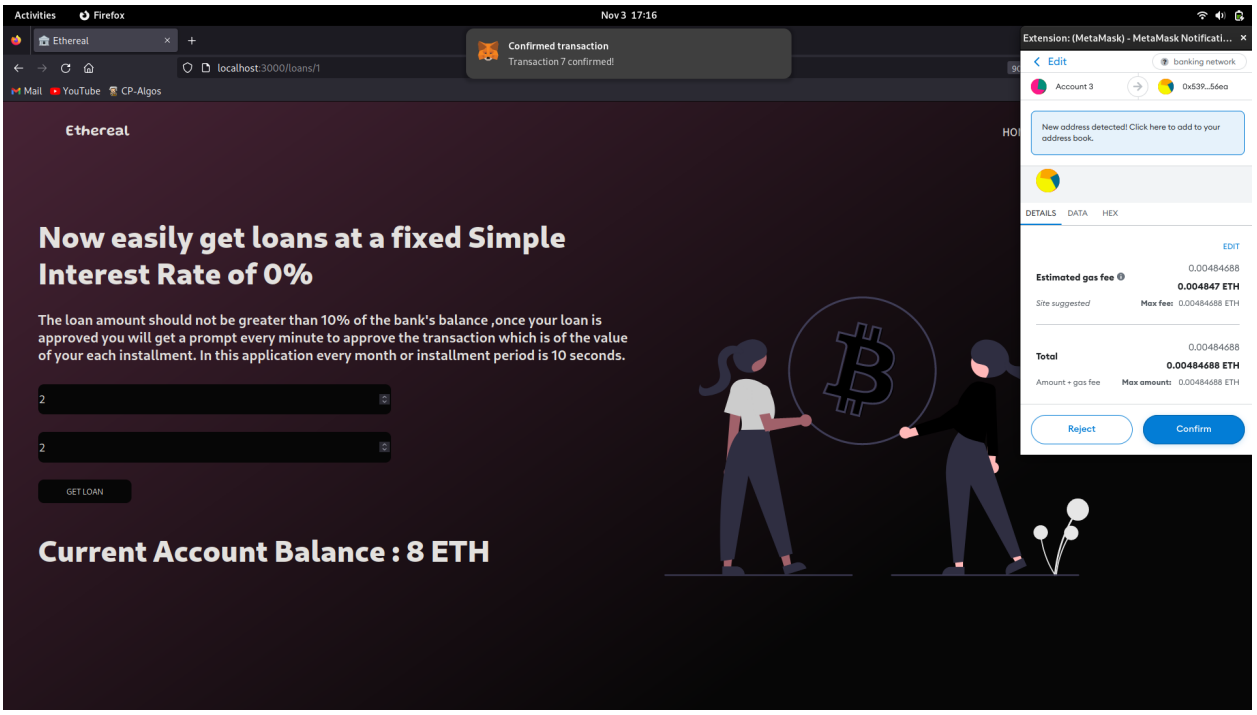
Getting Loans:



Installment 1:

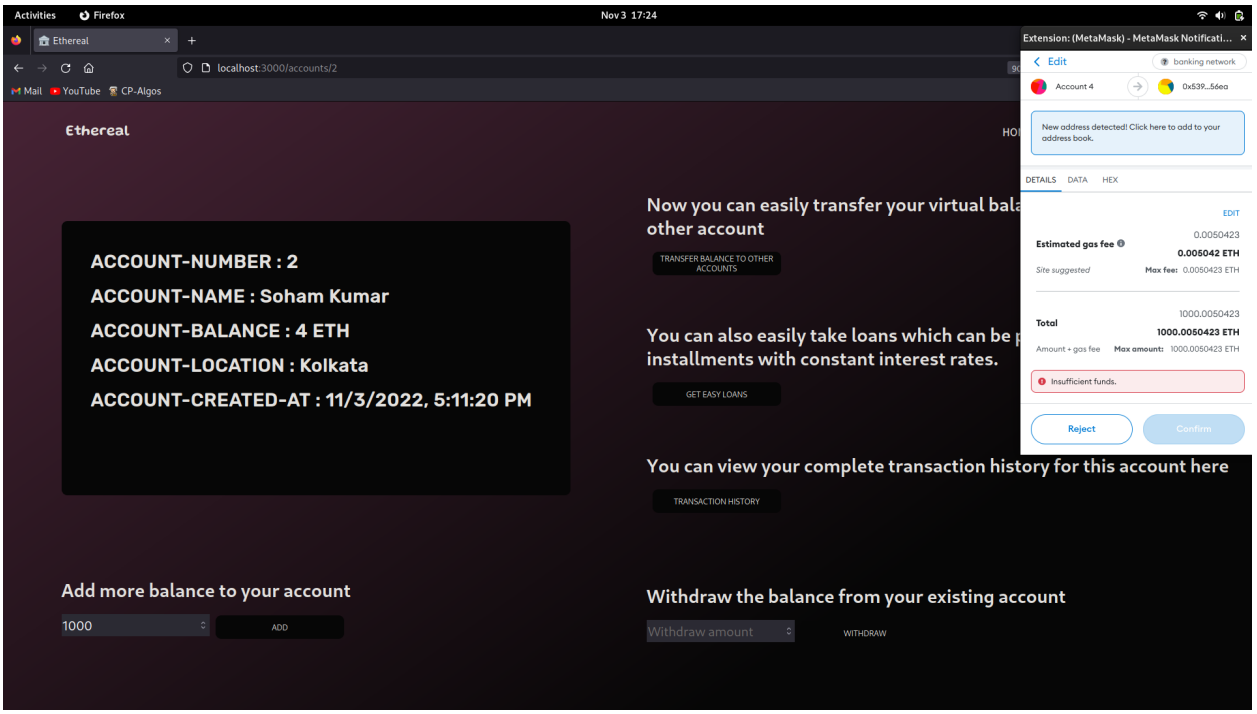


Installment 2:

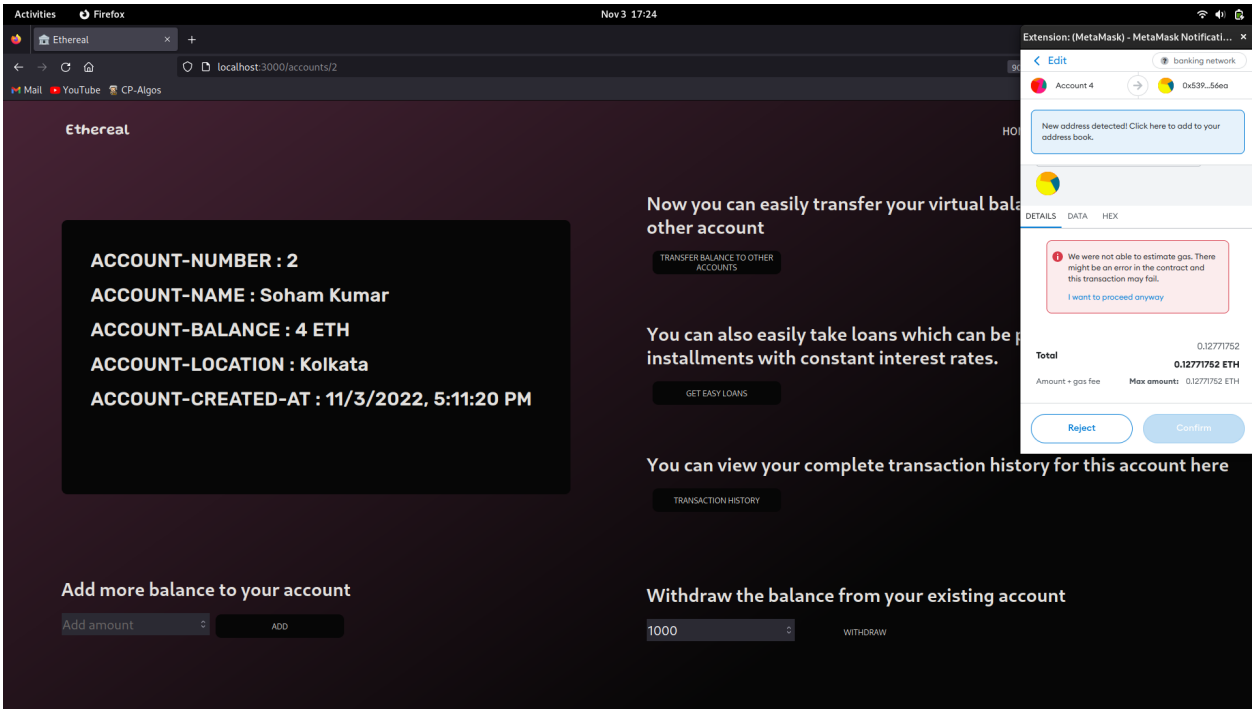


Exception Handling

Cannot Deposit More Eth than that present in wallet (must show Insufficient funds)

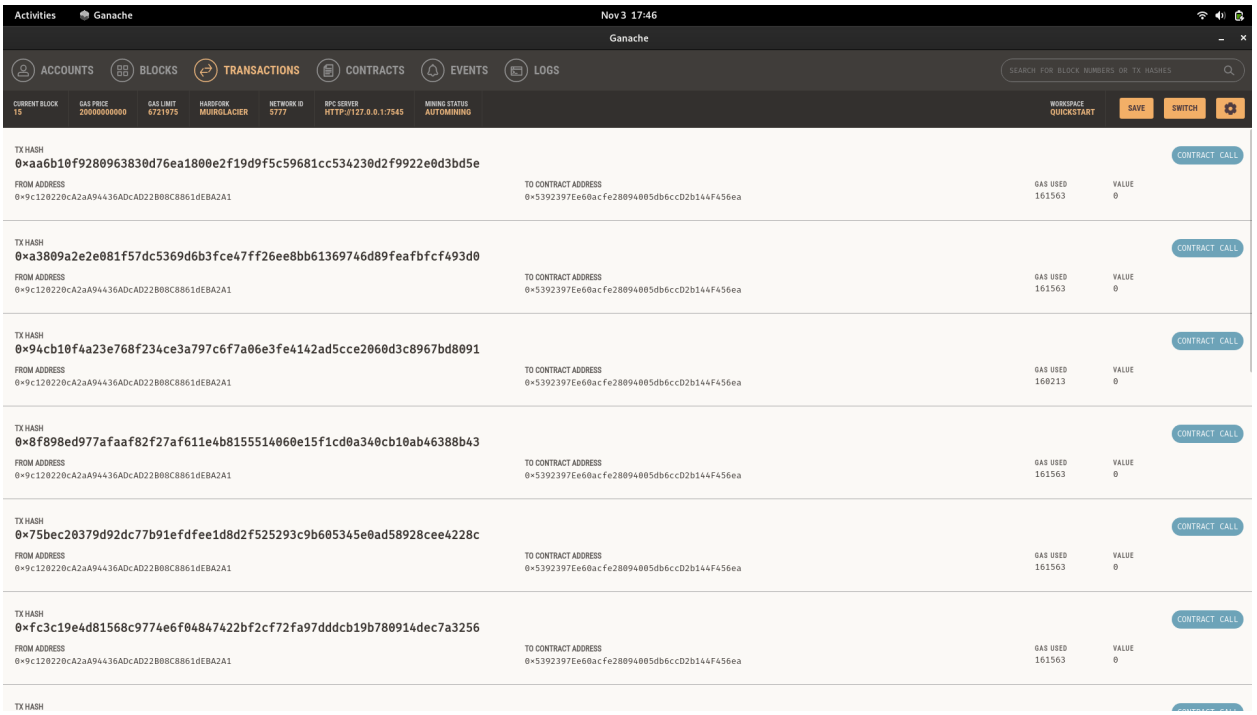


Cannot Withdraw More Than what is present in bank account:

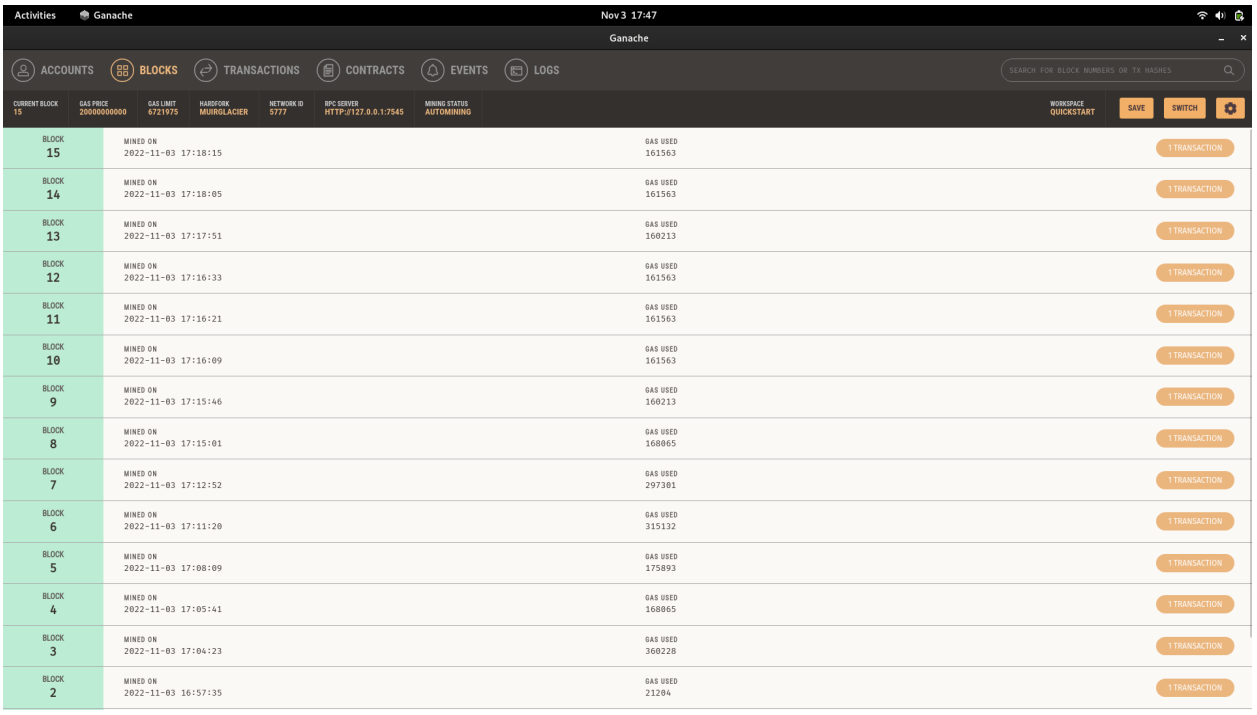


Final Database

Transactions in Blockchain:



Blocks in Blockchain:



It seems that the price of Bitcoin has been continuously declining, ranging from -5% to -29%. Whereas the price of ethereum escalated from 2% to 80% but faced a slight downgrade on May 17th, 2021, which was 44%. The continuous increment in the ethereum price change motivates users to invest more in ethereum-based decentralized applications compared to Bitcoin, as the return amount is far greater than the deposited amount. The statistic is represented in Table 1.

Date of Txns	Price of Per Bitcoin (in USD)	Price of Per Ethereum (in USD)	Change in Bitcoin Price (in percentage)	Change in Ethereum Price (in percentage)
April 19, 2021	55819.50	2273.56	-5%	2%
April 26, 2021	52971.92	2483.89	-7%	18%
May 3, 2021	57900	3083.42	-12%	40%
May 10, 2021	58849.82	3272.52	-5%	80%
May 17, 2021	44348.68	3400.87	-29%	44%

Table 1. Change in prices of bitcoin and ethereum

In developing our crypto banking system for blockchain technology, we focused on three core principles aimed at user information confidentiality, improving data security, and fast and reliable financial transactions. Another key benefit of our system is that it enhances transparency by establishing fund returns. All over the world, the banking industry is using this proposed banking system because it is based on the USD. It should be noted that our goal is to set up a secure and smooth transaction system that is limited to only bank based on the blockchain and not from one bank to another bank transactions. Once the terms of the contracts are fulfilled, the transactions will be executed and the transaction information will be stored in the blockchain.

Conclusion

The implemented system makes sure that the system will not face financial loss at any cost. It is a detailed improvement over the old manual banking system. The test result shows to make sure this system will not be crushed. Using USD as a transaction unit ensures that this system can be used worldwide in the banking industry. Banks can follow this ethereum-based banking system to make their banking systems more automated and secure for their users and shareholders.

In the future, other transaction functions can be developed. We only work on four individual sectors. There are many sectors in a bank. So, developers can work on that and make the banking system much more secure for their users. Making Smart Contracts smarter and richer all system-related functions will be performed if we build this system into a single Smart Contract. Developers can develop different Smart Contracts for every part of the system in the future. That is much more secure because if one smart contract's function is failed or is crushed, others are not affected by this. By improving the solidity language, we can mark untrustworthy transaction requests and avoid transferring, sending, or withdrawing them. Creating a new smart contract in Solidity blocks the untrustworthy transaction request and alerts the bank to it. In the future, blockchain-based government certificate verification can be developed that can access any blockchain user and no one can modify that information. Also, blockchain-based land record reservations
can be developed.

References

References

[1]. The Implementation of Blockchain in Banking System using Ethereum (2020), Masum Bakaul, Nipa Rani Das, Madhabi Akter Moni(Department of CSE Britannia University Cumilla, Bangladesh) DOI:10.5120/ijca2020919895

[2]. Decentralized Finance and Crypto Banking System Using Ethereum-based Blockchain Technology , Mamun Ahmed, Saha Reno* , Salma Akter and A. K. M. Abu Nowshad Chowdhury Department of Computer Science and Engineering, Bangladesh Army International University of Science and Technology, Cumilla, Bangladesh. https://journal.baiust.edu.bd/wp-content/uploads/2022/08/3.pdf

Appendix

CODE: https://github.com/Sudharshanan-B/ETHERAL

DEMO:https://drive.google.com/file/d/1snA9M-JYqGOfam45btbozTSOztfFrqUU/view?usp=share_link