**Aim:** Implement the Blockchain platform Ganache

**Theory:**

**What is a Ganache?**

In the context of blockchain technology, "ganache" refers to a development tool that is used to create a local, private Ethereum blockchain for testing and development purposes. Ethereum is a blockchain platform that supports smart contracts and decentralized applications (DApps), and Ganache is a tool that helps developers build and test their DApps in a controlled and isolated environment.

Here are some key features and functions of Ganache in the context of blockchain development:

1. Local Blockchain: Ganache allows developers to set up a local Ethereum blockchain on their development machines. This local blockchain is not connected to the main Ethereum network (mainnet) and is used solely for development and testing.

2. Instant Mining: Ganache instantly mines new blocks, so developers don't have to wait for block confirmations as they would on the main Ethereum network. This accelerates the development and testing process.

3. Accounts and Ether: Ganache provides a set of pre-funded accounts with Ether (the cryptocurrency used on the Ethereum network) for testing purposes. Developers can use these accounts to simulate transactions and interactions within their DApps.

4. Gas Control: Gas is the computational cost associated with executing transactions and smart contracts on the Ethereum network. Ganache allows developers to control gas prices and limits, making it easier to test different scenarios and optimize gas usage.

5. Developer-Friendly Interface: Ganache typically comes with a user-friendly interface that displays information about the blockchain, including account balances, transactions, and smart contract deployments. This makes it easier for developers to monitor and debug their DApps.

6. Integration with Development Tools: Ganache can be integrated with various development tools and frameworks, such as Truffle and Remix, to streamline the development workflow and facilitate the testing of smart contracts.

Overall, Ganache is a valuable tool for Ethereum developers, as it provides a convenient and customizable environment for building and testing smart contracts and decentralized applications

without the need to interact with the main Ethereum network, which could involve real cryptocurrency and higher transaction costs.

**List the steps involved in connecting Ganache Environment with a Metamask and Remix IDE for performing transactions.**

Connecting a Ganache environment with MetaMask and Remix IDE for performing transactions involves a series of steps. Here's a step-by-step guide:

Prerequisites:
1. Install Ganache: Download and install Ganache, which provides a local Ethereum blockchain environment for development and testing.

2. Install MetaMask: Install the MetaMask browser extension, which allows you to interact with Ethereum-based applications and manage your Ethereum accounts.

3. Install Remix IDE: If you haven't already, set up Remix IDE, which is an online development environment for Ethereum smart contracts.

Steps:

1. Start Ganache:
- Launch the Ganache application on your computer.
- Ganache will create a local blockchain with a set of predefined accounts and private keys, along with a local Ethereum RPC (Remote Procedure Call) endpoint, typically at `http://localhost:7545` or `http://127.0.0.1:7545`.

2. Configure MetaMask:
- Open your MetaMask extension in your web browser.
- Click on the network selection dropdown (it may initially be set to "Main Ethereum Network").
- Choose "Custom RPC" to add a custom network.
- Enter the Ganache RPC endpoint (e.g., `http://localhost:7545`) as the "New RPC URL."
- Save the configuration.

3. Import Accounts to MetaMask:

- In Ganache, you will see a list of accounts with private keys. Import these accounts into MetaMask for testing purposes:
    - Click on an account in Ganache to reveal its private key.
    - In MetaMask, click on your account icon, then select "Import Account."
    - Paste the private key and import the account.

4. Connect Remix IDE to Ganache:
- Open the Remix IDE in your web browser.
- In the Remix interface, click on the "Settings" tab.
- Under the "General Settings" section, change the environment to "Web3 Provider."
- Enter the Ganache RPC endpoint (e.g., `http://localhost:7545`) as the "Web3 Provider Endpoint."
- Save the settings.

5. Compile and Deploy Smart Contracts:
- Write or import your smart contract code in Remix IDE.
- Compile the smart contract code using Remix.
- Deploy the smart contract to your Ganache network by selecting the appropriate account in Remix and clicking the "Deploy" button.

6. Interact with Smart Contracts:
- Once your smart contract is deployed, you can interact with it using Remix.
- You can use MetaMask to sign transactions and confirm interactions with the smart contract.

7. Test Transactions:
- Create transactions within Remix to test your smart contract's functionality.
- When prompted by Remix or MetaMask, confirm and sign the transactions using one of your imported Ganache accounts.

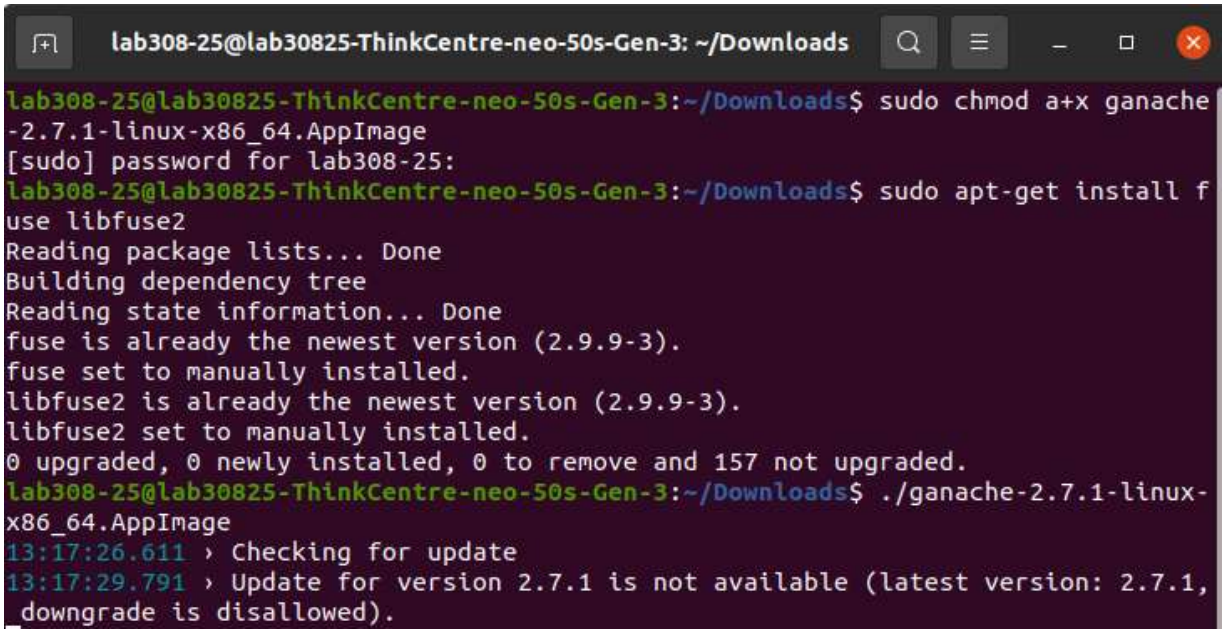
By following these steps, you can connect your Ganache environment with MetaMask and Remix IDE, enabling you to develop, deploy, and test Ethereum smart contracts on a local blockchain before deploying them to the main Ethereum network. This allows for safe and efficient development and testing of your blockchain applications.

**Conclusion:**

In conclusion, the implementation of the Ganache blockchain platform for our experiment has proven to be a valuable and essential tool for blockchain development and testing. Ganache provided a controlled and isolated environment that allowed us to simulate and experiment with various blockchain-related activities, without the complexities and costs associated with the main Ethereum network.

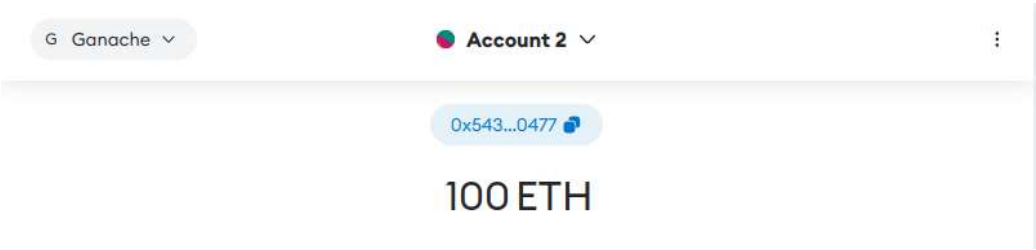
**Output:**

**Ganache installation and setup:**

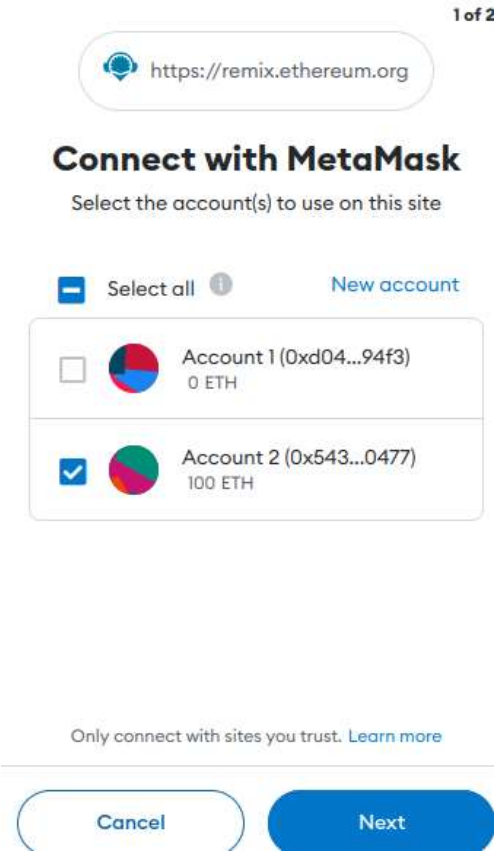



**Importing account in metamask using private key from Ganache:**

**Account balance:**

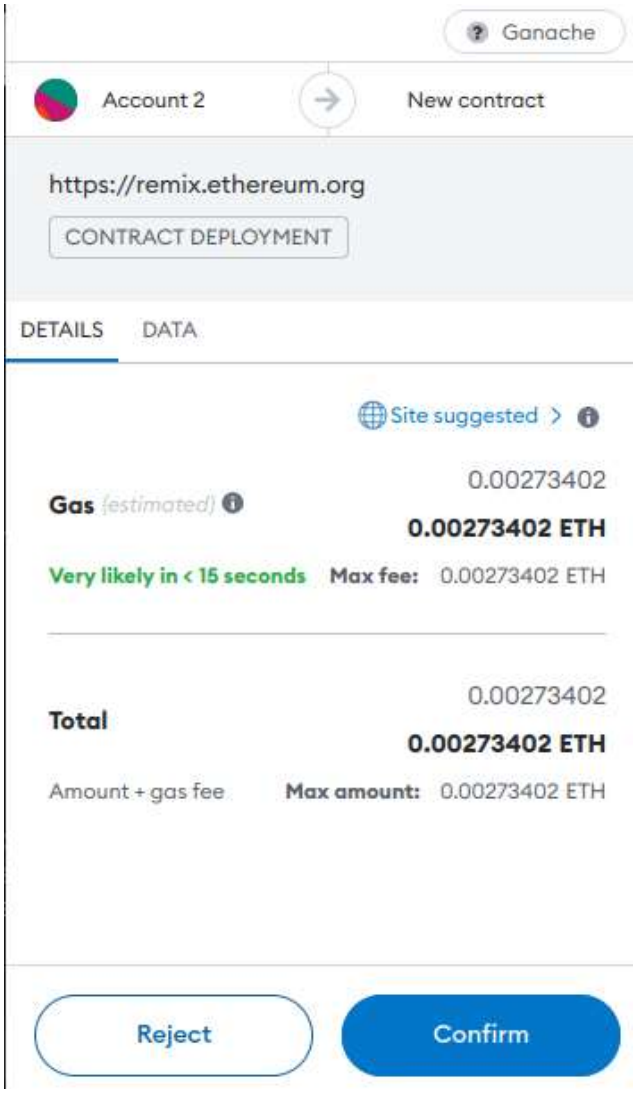



**Connecting with metamask:**

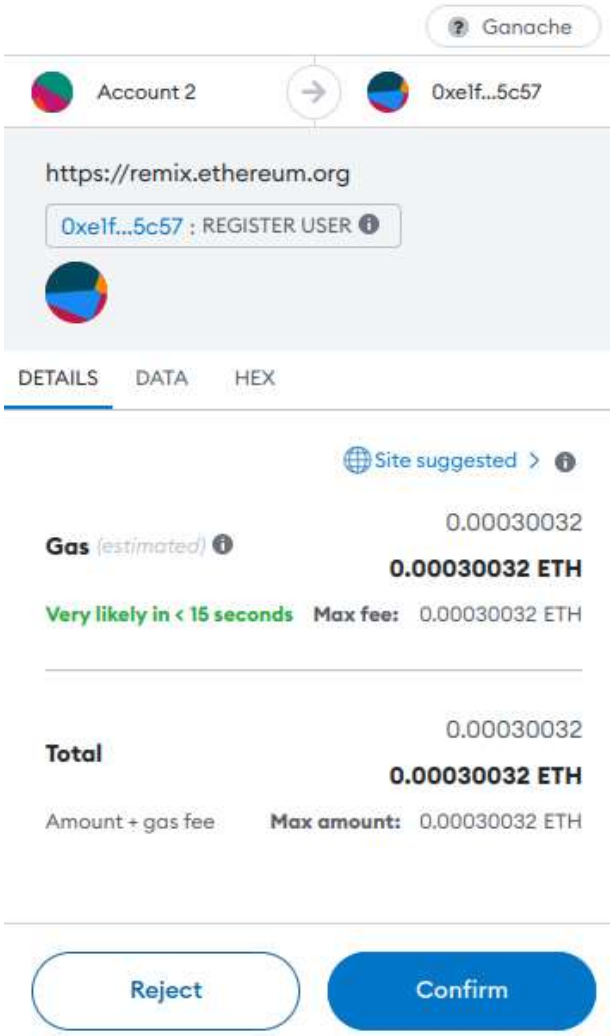



**Account balance in Remix IDE:**

**Transaction request for Contract Deployment:**



**Transaction request for Registering user details:**



**Contract Deployment Transaction details:**

[block:1 txIndex:0] from: 0x543...D0477 to: UserAccount.(constructor) value: 0 wei
data: 0x608...20033 logs: 0 hash: 0x5b3...7de14

| | |
|---|---|
| status | true Transaction mined and execution succeed |
| transaction hash | 0x5f9b1c1705ea991d0a522df6412b7a87d518ee7c57f489fdb85427a35a8875ab |
| block hash | 0x5b384aa21a08584adf37d0760a3ba80f2c0f9526c55925c14e25748447b7de14 |
| block number | 1 |
| contract address | 0xe1fdf98df02699644C925DB0d175269111DA5c57 |
| from | 0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477 |
| to | UserAccount.(constructor) |
| gas | 607561 gas |
| transaction cost | 607561 gas |
| input | 0x608...20033 |
| decoded input | {} |
| decoded output | - |
| logs | [] |

logs


```
[
        {
                "from": "0xe1fdf98df02699644C925DB0d175269111DA5c57",
                "topic":
"0x48cac28ad4dc618e15f4c2dd5e97751182f166de97b25618318b2112aa951a2f",
                "event": "UserRegistered",
                "args": {
                        "0": "0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477",
                        "1": "pushkaraj",
                        "userAddress": "0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477",
                        "username": "pushkaraj"
                }
        }
]
```


val 0 wei

## Registering User Transaction details:

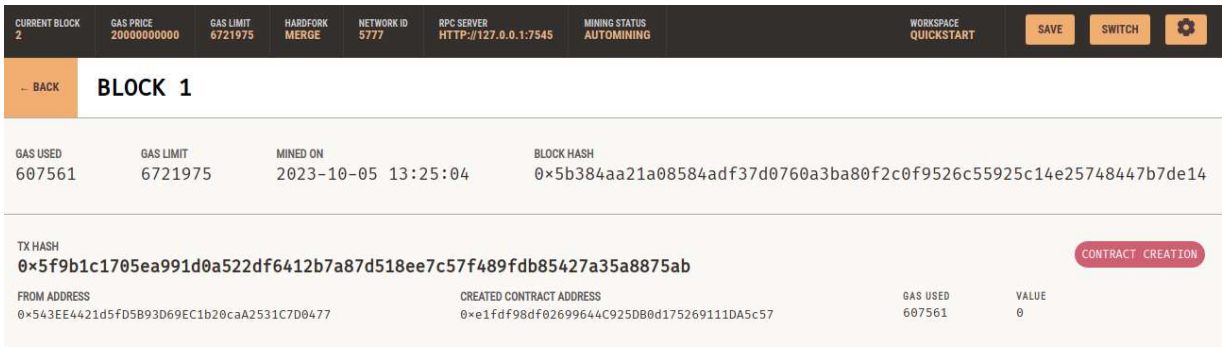
**[block:2 txIndex:0] from:** 0x543...D0477 **to:** UserAccount.registerUser(string) 0xe1f...A5c57
**value:** 0 wei **data:** 0x704...00000 **logs:** 1 **hash:** 0xf86...c0e8c

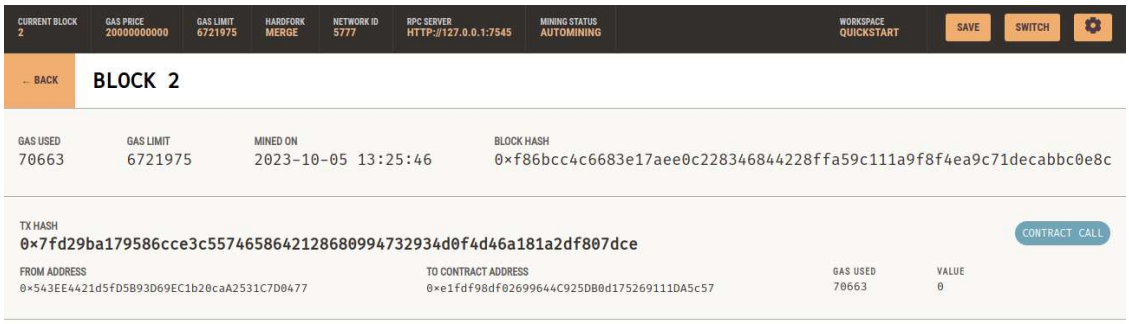
| | |
|---|---|
| **status** | true Transaction mined and execution succeed |
| **transaction hash** | 0x7fd29ba179586cce3c557465864212868099473293 4d0f4d46a181a2df807dce |
| **block hash** | 0xf86bcc4c6683e17aee0c228346844228ffa59c111a9f8f4ea9c71decabbc0e8c |
| **block number** | 2 |
| **from** | 0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477 |
| **to** | UserAccount.registerUser(string) 0xe1fdf98df02699644C925DB0d175269111DA5c57 |
| **gas** | 70663 gas |
| **transaction cost** | 70663 gas |
| **input** | 0x704...00000 |
| **decoded input** | { "string _username": "pushkaraj" } |
| **decoded output** | - |
| **logs** | [ |

CALL **[call] from:** 0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477 **to:** UserAccount.users(address)
**data:** 0xa87...d0477

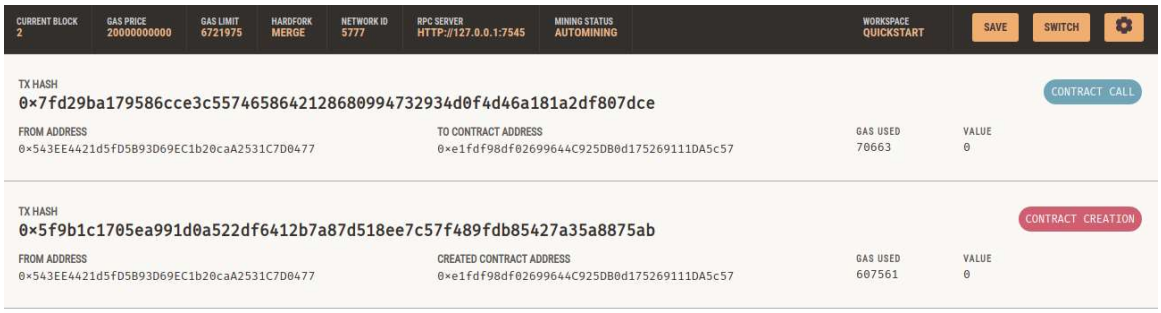
| | |
|---|---|
| **from** | 0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477 |
| **to** | UserAccount.users(address) 0xe1fdf98df02699644C925DB0d175269111DA5c57 |
| **input** | 0xa87...d0477 |
| **decoded input** | { "address ": "0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477" } |
| **decoded output** | { "0": "string: username pushkaraj", "1": "address: wallet 0x543EE4421d5fD5B93D69EC1b20caA2531C7D0477", "2": "bool: isRegistered true" } |
| **logs** | [] |

## Block Details on Ganache:

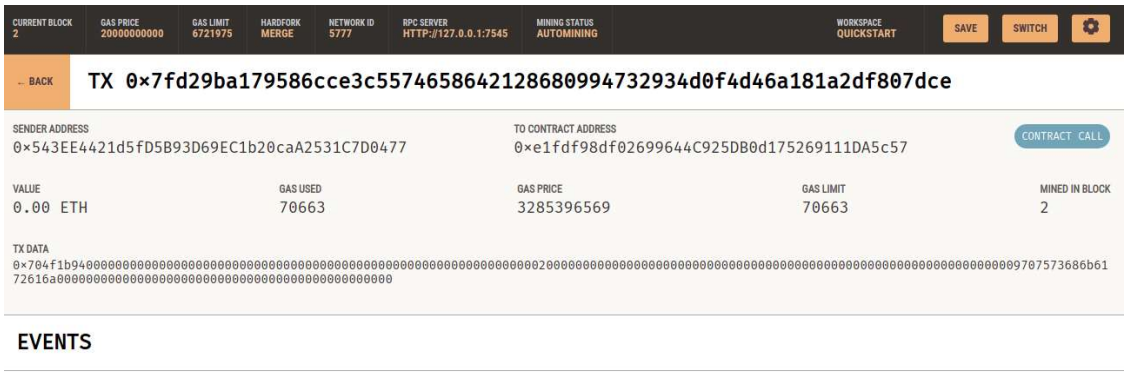



## Transaction details on Ganache:



CURRENT BLOCK 2
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MERGE
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
← BACK
TX 0×5f9b1c1705ea991d0a522df6412b7a87d518ee7c57f489fdb85427a35a8875ab
SENDER ADDRESS
0×543EE4421d5fD5B93D69EC1b20caA2531C7D0477
CREATED CONTRACT ADDRESS
0×e1fdf98df02699644C925DB0d175269111DA5c57
CONTRACT CREATION
VALUE
0.00 ETH
GAS USED
607561
GAS PRICE
3375000000
GAS LIMIT
607561
MINED IN BLOCK
1
TX DATA
0×6080604052348015610010576000080fd5b50610a0d80610020600039600f3fe6080604052348015610010576000080fd5b506004361061003657600035560e01c8063704f1b941461003b
578063a87430ba14610057575b600080fd5b6100556004803603810190610050919061049456 5b610089565b005b6100716004803603810190610 06c919061053b565b61025b565b604051
6100809392919061062356 5b604051809103 90f35b6000803373ffffffffffffffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffffffffffffff16815260200190
81526020016000206001016014905490610100 0a900460ff16156101185760405 17f08c379a0000000000000000000000000000000000000000000000000000000081526004161010f90
6106ad565b604051809103 90fd5b604051806060016040528082815260200133 73ffffffffffffffffffffffffffffffffffffffff16815260200160011515815250600080337 3ffffffff
ffffffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffffffffffffff16815260200190815260200160002060008201518160000190816101 9f91906108e3565b50
6020820151816001016000610100 0a81548173ffffffffffffffffffffffffffffffffffffffff021916908373ffffffffffffffffffffffffffffffffffffffff16021790555060408201
5181600101601461010000a81548160ff0219169083151502179055509050503373ffffffffffffffffffffffffffffffffffffffff167f48cac28ad4dc618e15f4c2dd5e97751182f166de
97b25618318b2112aa951a2f8260405161025091906109b5565b6040518091039 0a250565b6000602052806000526040600020600091509050806000018054 61027e906106fc565b80601f
0160208091040260200160405190810160405280929190818152602001828054 6102aa906106fc565b80156102f75780601f106102cc576101008083540402835291602001916102f7565b
8201919060005260206000020905b81548152906001019060200180831161 02da57829003601f168201915b5050505050908060010160009054906101000a900473ffffffffffffffffffff
ffffffffffffffff1690806001016014905490610100 0a900460ff16905083565b6000604051905090565b6000080fd5b600080fd5b600080fd5b600080fd5b6000601f19601f830116
9050919050565b7f4e487b7100000000000000000000000000000000000000000000000000000000000000 0052604160045260246000fd5b6103a182610358565b810181811067ffffffffffff
ffff82111715610 3c0576103bf610369565b5b80604052505050565b60006103d361033a565b90506103df828261039856 5b919050565b600067ffffffffffffffff8211156103ff576103
fe610369565b5b6104082610358565b9050602081019050919050565b828183376000838015250505056 5b60006104376104328461 03e4565b6103c9565b9050828152602081018484 84
01111561045357610452610353565b5b6104 5e8482856104155656 5b5093925050505 65b600082601f83011261047b5761047a61034e565b5b813561048b8482602086016104245656 b915050
92915050565b60006020828403121561 04aa576104a9610344565b5b600082013567ffffffffffffffff8111156104c8576104c7610349565b5b6104d484828501610466565b9150509291
5050565b600073ffffffffffffffffffffffffffffffffffffffff8216905091905056 5b60006105088261 04dd565b905091905056 5b610518816104fd565b81146105235760008 0fd5b50
565b60008135905061053581610 50f565b92915050565b60006020828403121561055157610550610344565b5b600061055f8482850161052656 5b915050929150505 65b60008151905091
9050565b60008282526020820190509291505056 5b60005b83811015610 5a2578082015181840152602081019050610587565b60008484015250505050565b60006105b982610568565b61
05c38185610573565b93506105d38185602086016105845 65b6105dc8161035856 5b8401915050929150505 65b60006105f2826104dd565b905091905056 5b610602816105e7565b825250
50565b60008115905091905056 5b61061d8161060856 5b82525050565b60006060820190508181036000830152610 63d81866105ae565b905061064c602083018561 05f9565b61065960


CURRENT BLOCK 2
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MERGE
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
← BACK
TX 0×7fd29ba179586cce3c5574658642128680994732934d0f4d46a181a2df807dce
SENDER ADDRESS
0×543EE4421d5fD5B93D69EC1b20caA2531C7D0477
TO CONTRACT ADDRESS
0×e1fdf98df02699644C925DB0d175269111DA5c57
CONTRACT CALL
VALUE
0.00 ETH
GAS USED
70663
GAS PRICE
3285396569
GAS LIMIT
70663
MINED IN BLOCK
2
TX DATA
0×704f1b940000000000000000000000000000000000000000000000000000000000000000200000000000000000000000000000000000000000000000000000000000000009707573686b61
72616a0000000000000000000000000000000000000000000000000000

## EVENTS