Online Election and Task Planner- Decentralized Apps

on Ethereum Blockchain

Project Report

Presented to

 Professor Guha Jaychandran

Department of Computer Science

San José State University

In Partial Fulfillment

CS-266

Topics in Information Security

By

Aditi Walia, Hardik Kumar, Harshit Trehan

December 2019

ABSTRACT

Blockchain, as we know it, is widely accepted as one of the landmark technologies introduced in the recent years. The main concepts embedded in the technology are cryptography and decentralization. Blockchain is like a distributed ledger system, where the records or assets are being distributed among the users or participants and can be verified and validated by any of the participants contributing in the blockchain. This decentralized and distributed aspect of the technology makes it distinctive and different from the traditional centralized data handling systems. Blockchain technology came to light when *Bitcoin* cryptocurrency was introduced in 2008, whose underlying principle is cryptographic hashing, consensus and decentralization. Similarly, other cryptocurrencies have also been introduced like Ethereum, Ripple etc. Ethereum works on the principle of Proof of Stake (PoS), which is different from Bitcoin’s Proof of Work (PoW). PoS is a type of consensus algorithm which states that a participant can mine or validate blocks according to the stake the participant has in the blockchain. It is seen as a modification to PoW, where the computing power decides the ability to mine a block. Ethereum gave rise to development of a lot of Decentralized applications, known as DApps, which can be programmed, making data stored on the blockchain immutable and almost immune to attacks.

TABLE OF CONTENTS

1. Introduction………………………………………………………………………………. 4
2. Tools and frameworks……...……………………………………………………………. 5
3. Methodology………………………………………………………………………………6
4. The apps…………………………………………………………………………………...8
5. Conclusion……………………………………………………………………………….13

References…………………………………………………………………………………….…14

INTRODUCTION

Blockchain technology has been successfully used in scenarios where there’s a dire need of cryptography, data hiding and immutability. The block data structure is the main constituent of the blockchain, and each block is added to the blockchain after going through a validation and verification process conducted by the participating nodes. Then, each block has a cryptographic hash generated for it, which is the linking factor between the blocks. Any change in the data of the block might result in the change of its hash value, resulting in the breakage of the link. This underlying principle helps keeping the data put up on the blockchain persistent and inflexible and protects the system from data breaches or attacks.

For our project, we have developed an Online Election System and also a mini Task Planner, which are decentralized applications on Ethereum using Smart Contracts. In the election system, two candidates are chosen to be elected. The voters, which are the participating nodes on the Ethereum blockchain can vote for their favored candidate only once, and ultimately the winner is decided based on the votes casted by the users having accounts on the blockchain.

For the Task Planner, the users participating can populate their own Task planner list keeping a record of the tasks to be done. New tasks can be added in the list according to the requirement, and the tasks which have been completed can be marked complete. The completed tasks will still be visible and present on the blockchain, as completed tasks.

TOOLS AND FRAMEWORKS

The tools used in both the applications are:

* **Ganache:** A personal blockchain for Ethereum development, which can be used to deploy smart contracts and developing local applications
* **Solidity:** An object-oriented high-level programming language for implementing smart contracts
* **Truffle:** A developing environment for Ethereum DApps with built-in features like smart contract compilation, linking, deployment and binary management
* **Node.js:** An open-source, cross-platform JavaScript runtime environment for server-side scripting and handling dynamic web-applications.
* **Metamask:** A web extension for accessing Ethereum enabled distributed applications.
* **Web3.js:**
* **HTML, JavaScript and CSS:** Standard web development tools used to make out apps interact with the user.

METHODOLOGY

1. *Online Election System*

We implemented this system with the help of smart contracts, which are the integral part of Ethereum. The smart contract has been coded using Solidity programming language, which inculcates the functionalities and features of our application.

The contract *Elections* keeps track of the user accounts, which act as the voters as well. *struct* Candidate defines the candidates having id, name and voteCount. In addition to it, few mappings have been created:

* + voters: maps the address of the user account to a Boolean value. This is to keep track of the voters who have casted, in order to prevent a user to vote again.
  + candidates: maps the id to a Candidate object.

Moreover, functions have been incorporated to add the candidates for the election, and for the user to cast the vote.

**Ganache** is used to simulate a local blockchain having 10 user accounts with default private keys. These accounts can cast the vote which will result in updation of the voteCount of the respective candidates.

**Truffle** provides the environment to work on the application. It compiles the contracts in our project and formulates the required migrations folder and further deploys the contract on the blockchain utilizing a particular amount of gas. Now, any write operation on the blockchain initiated by any user account(Eg casting vote) will utilize gas, which can be visible on the Ganache GUI.

1. *Task Planner*

This is a mini application implemented similarly like the online election application, having the same underlying principles.

The contract *TodoList.sol* handles all the tasks entered in the Task planner list. Here, *struct* Task has parameters id, name and Boolean value completed, which depicts the completion status of the task. Mapping tasks have been created to map the task id to the Task object. Furthermore, functions *createTask* and *toggleCompleted* are for creation of a new task and toggling a task to completed status respectively.

**Ganache** and **Truffle** have been used in a way like the previous application. Ganache provides 10 accounts for the local Ethereum blockchain. Truffle is used to compile, migrate and deploy the contract on the blockchain. Additionally, we have used **Web3.js**, a JavaScript library which has JSON-RPC protocol embedded, to provide the UI and interaction with our DApp.

**MetaMask** is used to run the Ethereum DApp on the browser without running a full Ethereum node besides providing a identity vault to manage the accounts and sign transactions. Entering a task to the list as well as toggling a task to complete signifies a write operation on the blockchain and hence will use up gas related to an account.

THE APPS

Code for both the Dapps can be found on the following github link:

<https://github.com/harshittrehan/BlockchainProject>

1. *Online Election System:* Due to some ongoing issue with metamask and ganache, we were unable to connect this with the front end, but the smart contract is working fine, and we are attaching screenshots of a simulation of the election between 2 candidates run in the truffle console.

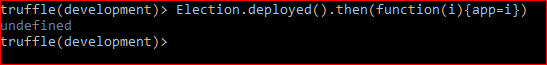


Fig. 1: The contract has been deployed successfully

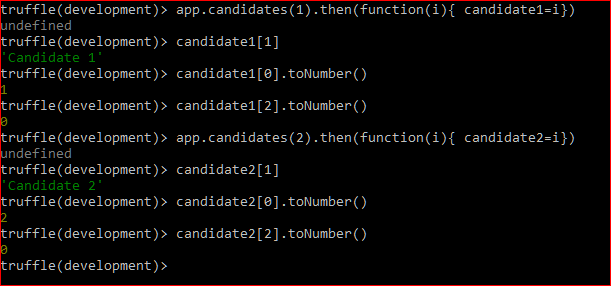


Fig. 2: Information about the 2 candidates. Field ‘1’ is the candidate name. Field ‘0’ is the candidate ID. Field ‘2’ is the candidate current vote count.

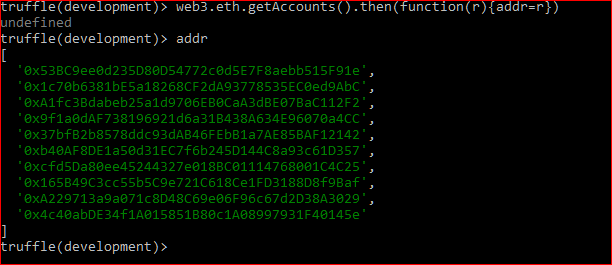


Fig. 3: These are the 10 available user addresses, provided by Ganache.

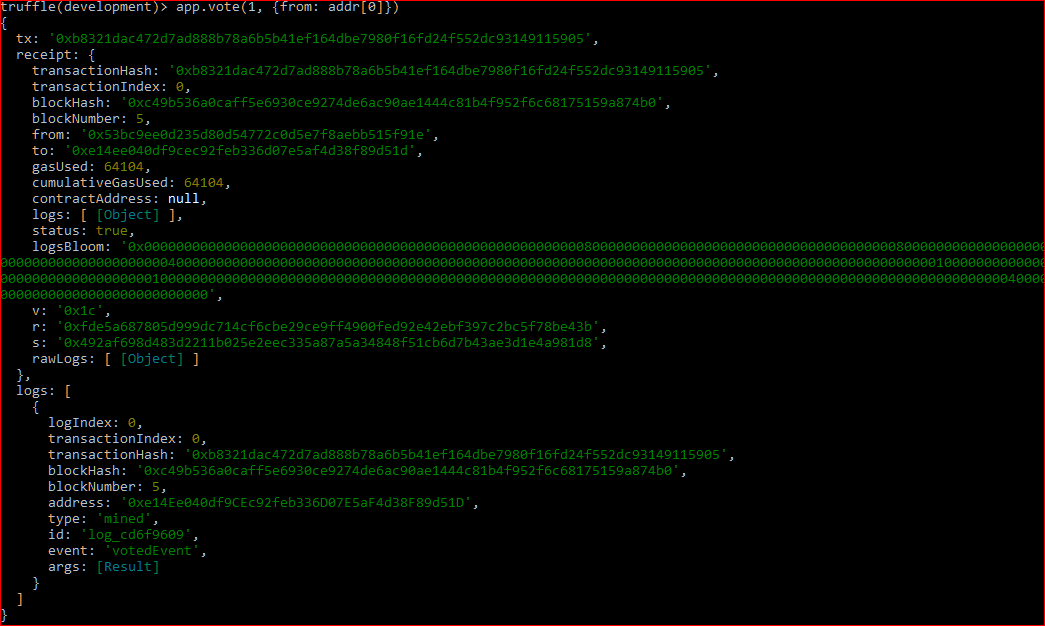


Fig. 4: Account ‘0’ has voted for candidate ID: 1 and this transaction has been recorded on the blockchain, we have the block number as well as the transaction hash.

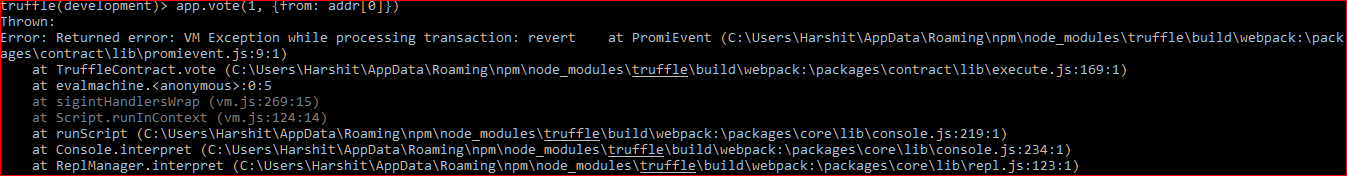


Fig. 5: If the user account ‘0’ tried to vote again, an exception is thrown. This ensures that a single user can’t vote more than once.

1. *Task Planner:* This app was running fine and we’re attaching the screenshots of the front end of the app running on a localhost using ganache.

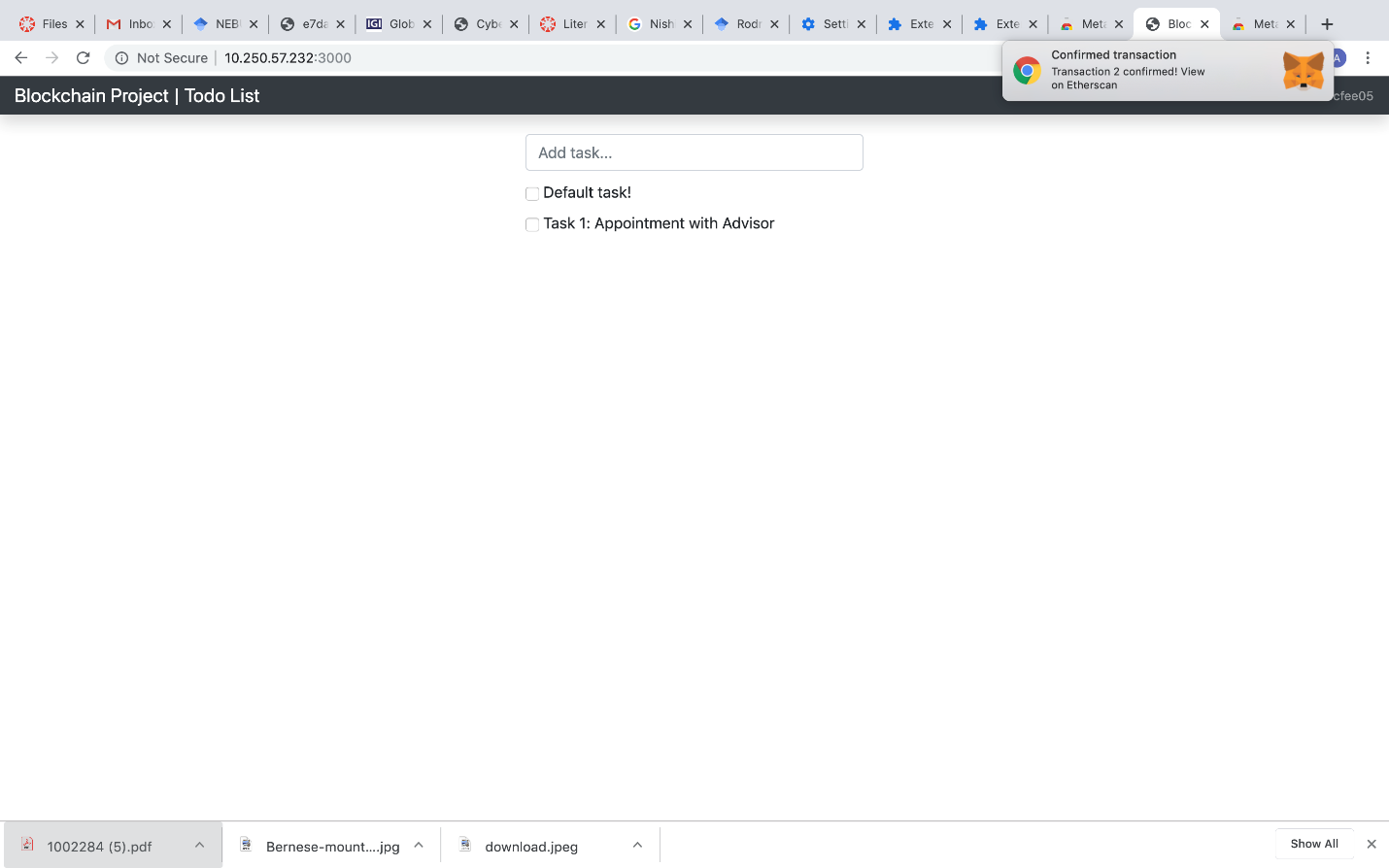


Fig. 6: We can add tasks on the task planner. For example, adding: Task 1: Appointment with Advisor.

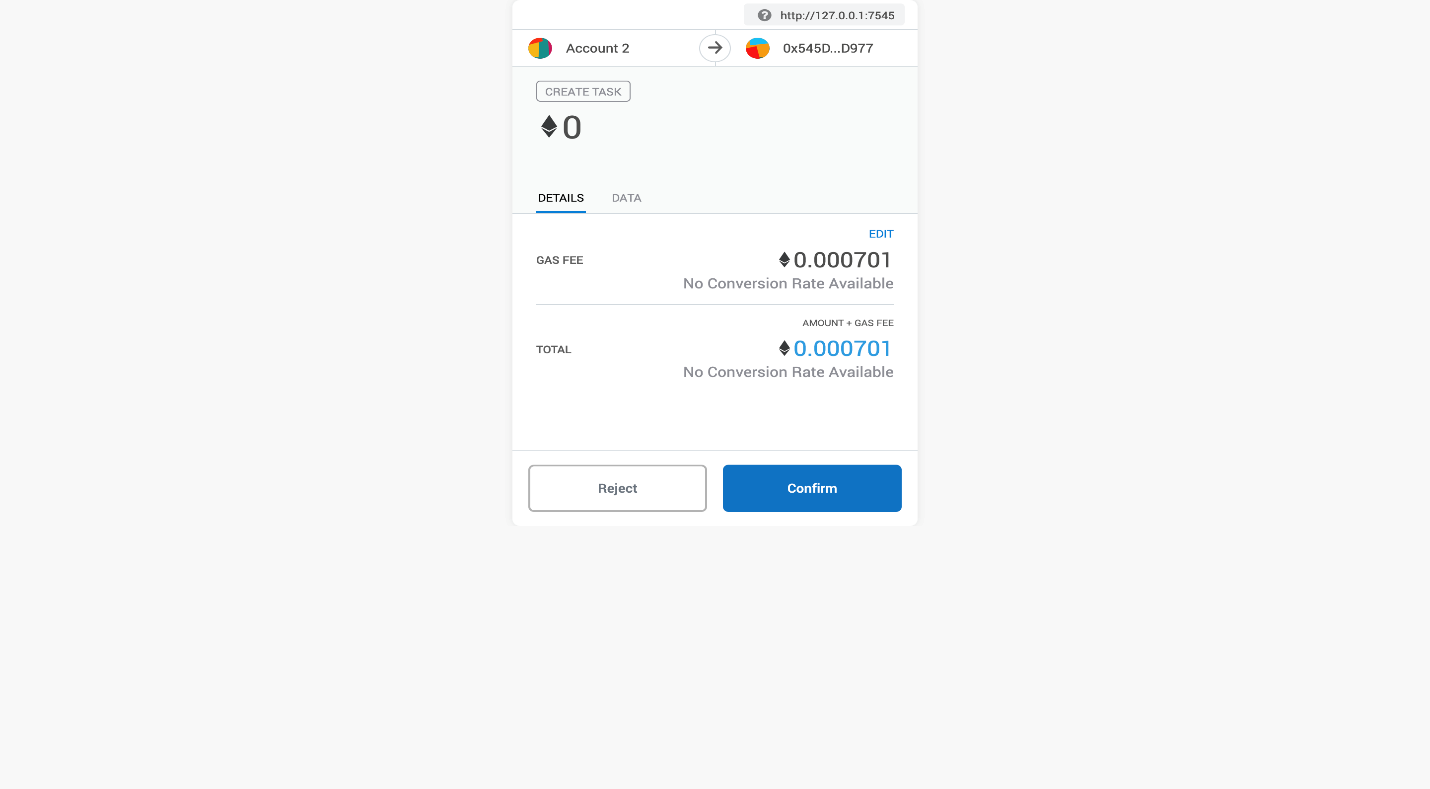


Fig. 7: Adding the task incurred some cost (write operation on the blockchain) which can be seen on MetaMask, hence it was written on the blockchain.

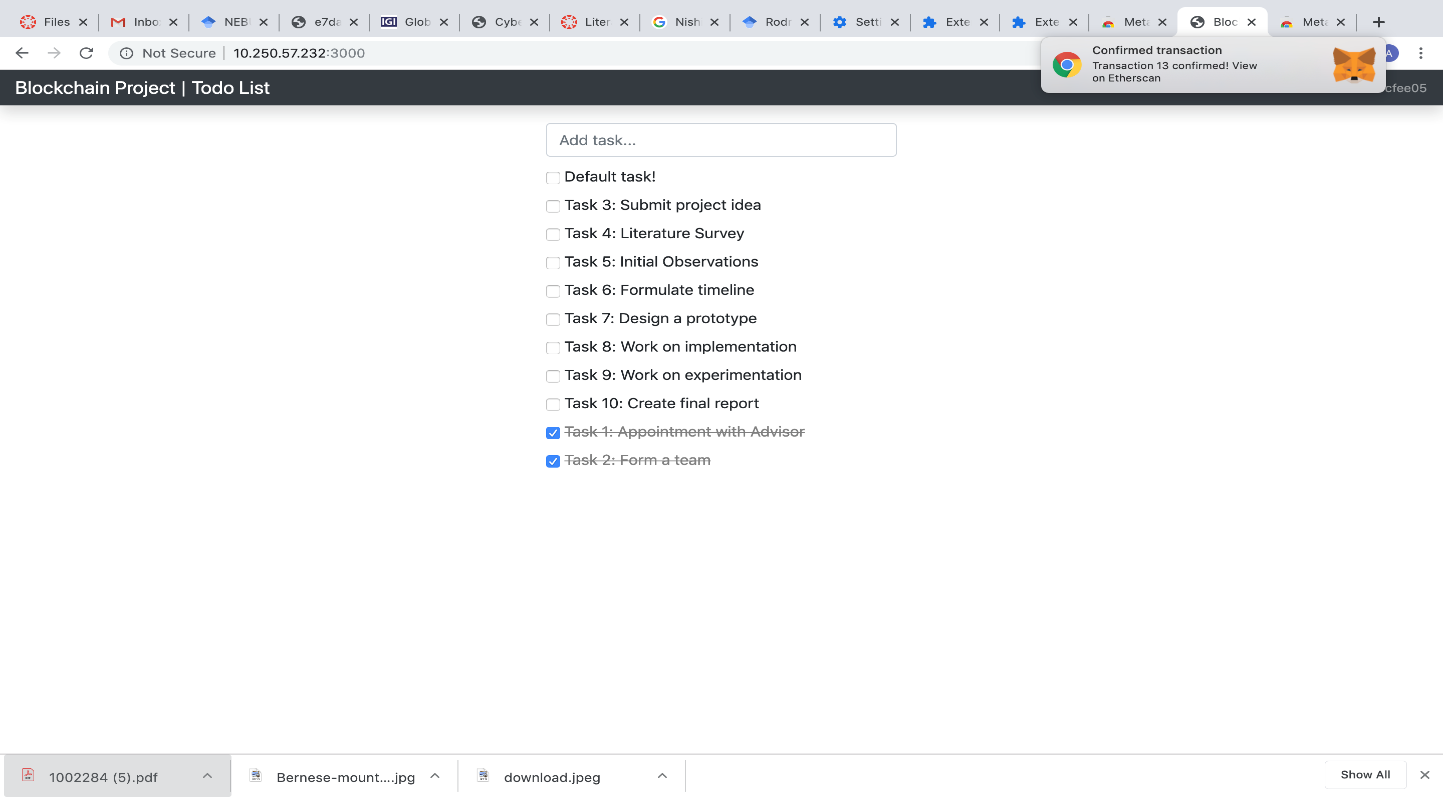


Fig. 8: We can also delete tasks. For example, deleting tasks 1 and 2 is possible.

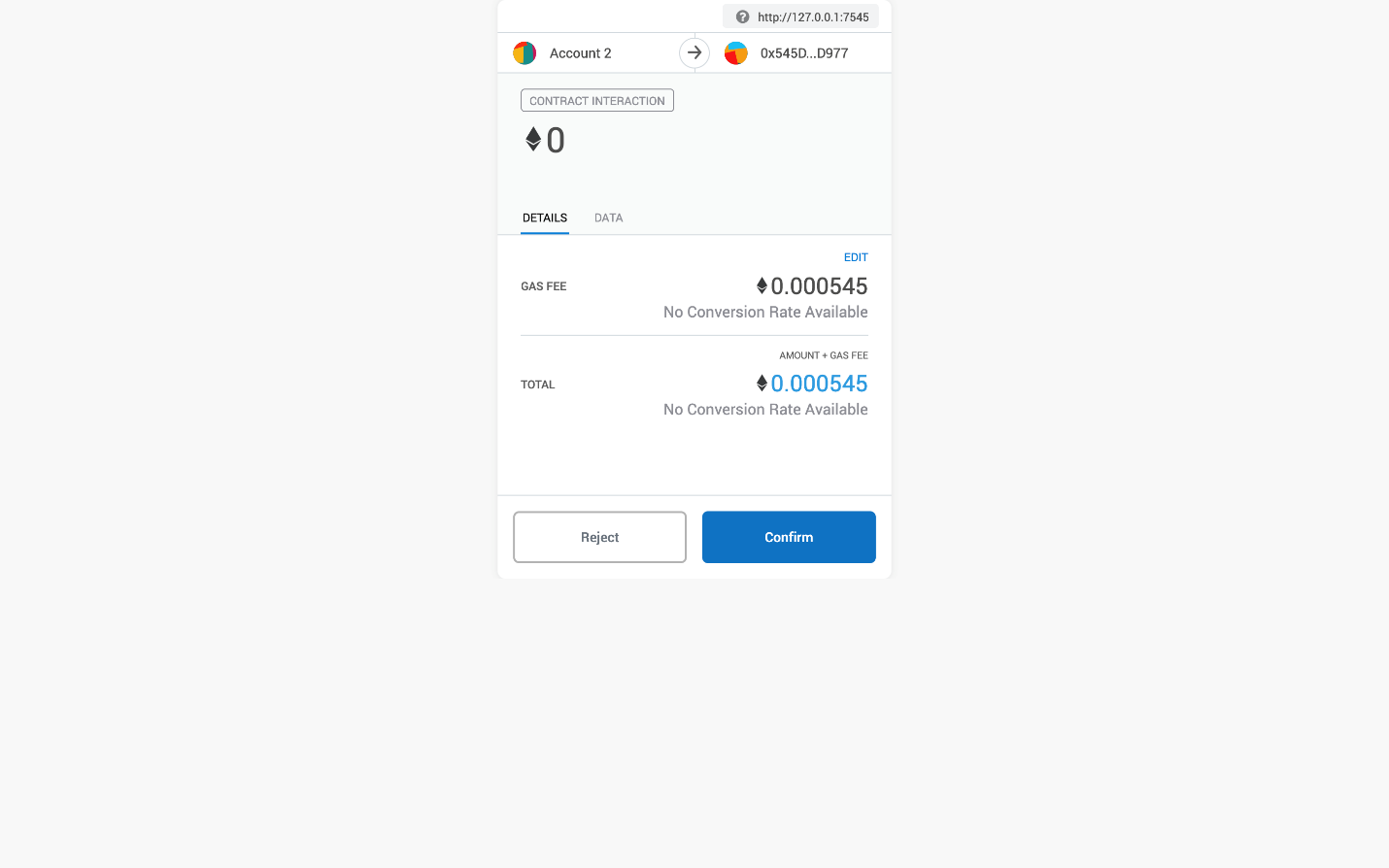


Fig. 9: Deleting a task also incurs some cost since it is a write operation on the blockchain.

CONCLUSION

The report has discussed the development of two decentralized applications on Ethereum blockchain, an online election system and a Task planner list. The objective of the project was to implement the DApps, keeping in mind that the inherent properties of a smart contract and distributed blockchain technology are also effectuated. For future work and enhancements, re-election can be invoked in case of a tie between the candidates. Also, a timestamp can be integrated in the system to introduce the constraint of time limit to the voting process. Regarding the Task planner list DApp, tasks can be placed under categories and also can be segregated according to the priority.

REFERENCES

1. <https://github.com/ethereum/wiki/wiki/White-Paper>
2. <https://github.com/ethereum/wiki/wiki/Decentralized-apps-(dapps)>
3. <https://blockgeeks.com/guides/solidity/>
4. <https://metamask.io/>
5. <https://www.trufflesuite.com/ganache>
6. <https://medium.com/rocknnull/a-glimpse-into-how-to-build-an-ethereum-dapp-89be65f1e578>