Exercise 5 - Group S

Filippo Boiani (387680)
Riccardo Sibani (382708)
Stefan Stojkovski (387529)
Gabriel Vilén (387555)

Theory Question

1. Why is it impossible to call external web service APIs from a Smart Contract?

The Ethereum blockchain was designed to be entirely deterministic. This means, that if I took the whole history of the network, then replayed it on my computer, I should always end up with the correct state.
Since the internet is non-deterministic and changes over time, then every time I replayed all of the transactions on the network, I would receive a different answer.
Determinism is important so that nodes can come to a consensus. If there were a contract that required the number of votes on some question, the value could differ from time to time or even place to place, causing nodes in the future or without access to this site to reach different conclusions about the state of the network, thus breaking the consensus.
By requiring that every data input is initiated through an external transaction, we can be sure that the blockchain itself contains all of the information required to verify itself. By using a single contract-level oracle (https://oraclize.it) instead of a network or consensus-level feature, we ensure that there is only one canonical result.

Reference:
- https://ethereum.stackexchange.com/questions/301/why-cant-contracts-make-api-calls

2. What is the difference between Message Calls and Transactions in Ethereum?

Message Calls
A call is a local invocation of a contract function that does not broadcast or publish anything on the blockchain.
It is a read-only operation and will not consume any Ether. It simulates what would happen in a transaction, but discards all the state changes when it is done.
It is synchronous and the return value of the contract function is returned immediately.

Transactions
A transaction is broadcasted to the network, processed by miners, and if valid, is published on the blockchain.
It is a write-operation that will affect other accounts, update the state of the blockchain, and consume Ether (unless a miner accepts it with a gas price of zero).
It is asynchronous, because it is possible that no miners will include the transaction in a block (for example, the gas price for the transaction may be too low). Since it is asynchronous, the immediate return value of a transaction is always the transaction's hash. To get the "return value" of a transaction to a function, Events need to be used.

Reference:
- https://ethereum.stackexchange.com/questions/765/what-is-the-difference-between-a-transaction-and-a-call

3. What is the difference between UTXO and Account Transaction models?

UTXO system characteristics are:
An individuals 'balance' is the value of UTXO he can unlock with his private keys

The value distribution is defined by the rights to use the transaction outputs as inputs for new transactions
The overall value in the system is defined as the value of all unspent transaction outputs, i.e., they are not inputs of other transactions
A transaction always refers to previous transactions as inputs
For validation of the transaction the following must be done:
every referenced input must be valid and not yet spent
the transaction must have a signature matching the owner of the input for every input
the total value of the inputs must equal or exceed the total value of the outputs

Account-based system characteristics are:

a global state stores a list of accounts with balances, code, and internal storage
a transaction is valid if the sending account has enough balance to pay for it, in which case the sending account is debited and the receiving account is credited with the value
if the receiving account has code, the code runs, and internal storage may also be changed, or the code may even create additional messages to other accounts which lead to further debits and credits
A transaction specifies sender and receiver accounts and does not refer to previous transactions
An individuals overall balance is the sum of balances of all accounts he owns

For validation of the transaction the following must be done:
Check whether signatures match with sender's address
Check that current nonce matches the nonce in sender's account. If true, increment sender's nonce
Ensure the sender's balance is larger or equal than the transaction value

In general, the benefits of UTXOs are:
Higher degree of privacy: if a user uses a new address for each transaction that they receive then it can be difficult to link accounts to each other
Potential scalability paradigms: UTXOs are more theoretically compatible with certain kinds of scalability paradigms
The benefits of accounts are:
Large space savings: because every transaction need only make one reference and one signature and produces one output
Greater fungibility: because there is no blockchain-level concept of the source of a specific set of coins, it becomes less practical to institute a redlist/blacklisting scheme
Simplicity: easier to code and understand, especially once more complex scripts become involved
Constant light client reference: light clients can at any point access all data related to an account by scanning down the state tree in a specific direction

References:
- https://ethereum.stackexchange.com/questions/326/what-are-the-pros-and-cons-of-ethereum-balances-vs-utxos
- And the slides provided in the lectures.

Blockchain Programming Task

HOW TO RUN

1- Run the following command in order to start the go ethereum client

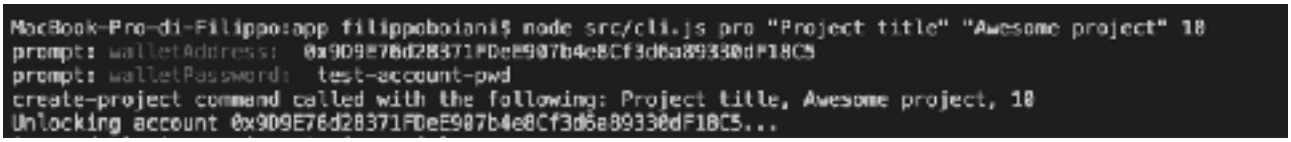
```
$ geth --fast --cache=512 --rpcapi personal,db,eth,net,web3 --rpc --
testnet
```

2- Go to the project folder **`ex5-group-s/app`**

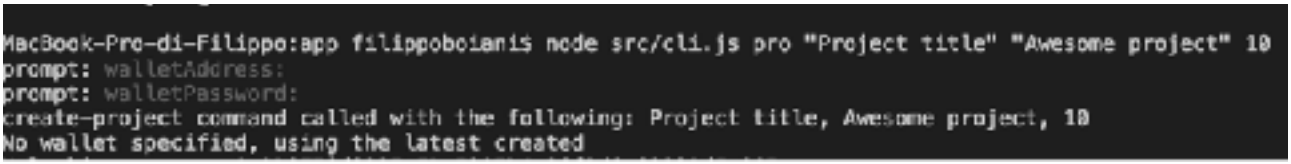
DEPLOY A CONTRACT
Run the following command

```
$ node src/cli.js create <title> <desc> <amount>
```

in order to deploy a project contract with the specified title, description and funding goal. Once you have run the command, the program will ask for wallet address and password. You can decide either to specify them



or press enter key and use the default account already used in the config file (could be either an account set up from us or the last account to used to deploy)



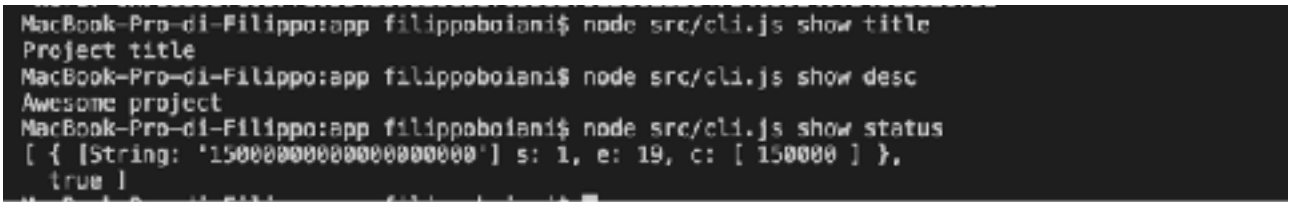
Either way, if you specified the correct credentials (or none at all) the output should be something like this:

3 - Show status
Run the following command:

```
$ node src/cli.js show <status | title | desc | goal | fundings>
[project]
```

in order to have the status (amount of fundings and a boolean **`true`** it the goal has been reached) or the title, description, goal or amount funded of the latest deployed project.

The output should be like this:

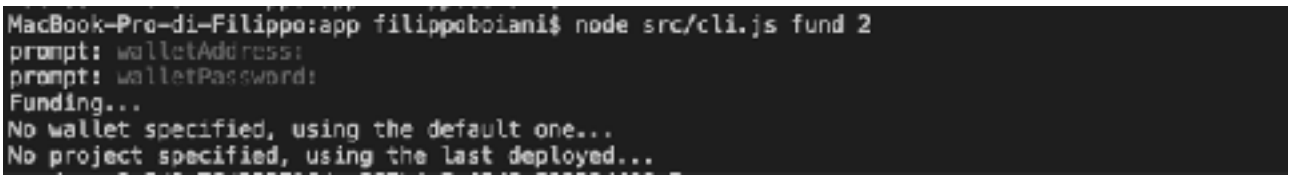


optionally it is also possible to specify the project address to have the information regarding the one specified.

4 - fund the project
running the following (**note**: the amount is in ether)

```
$ node src/cli.js fund <amount> [project]
```

You can specify the amount and (optionally) the project



you will asked for wallet credential like the deploy command and you can either specify them or press enter (output for the latter above).

Below, the output for the former case: