Blockchain Optimization & Applications

CIS 6930

Home work 1

Spring 2020

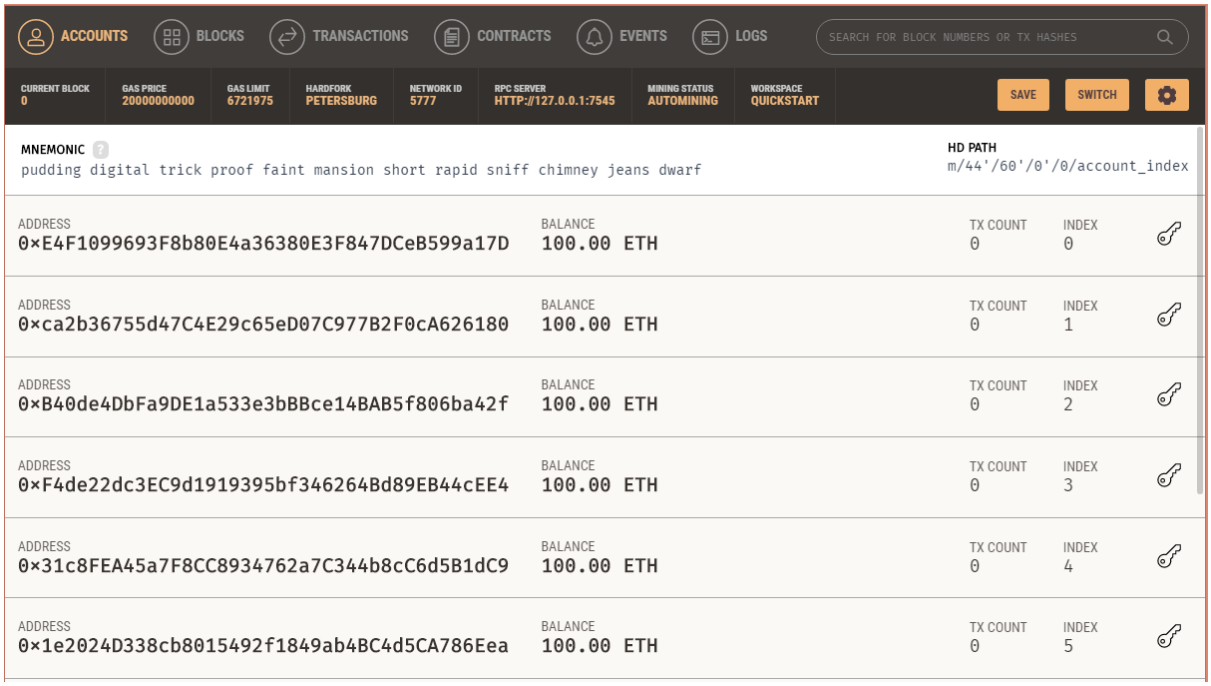
Name	Aniket Dash
UFID	7549-9549
Email	aniket.dash@ufl.edu

Function Description and Transactions

`function bid() public payable :`

The function does not require any parameters as all the required parameters are globally set .The payable keyword is used to make the method accept ethers. We first check using the require method that is the auction still active, if inactive the sent ethers are reverted and method call terminates. Next we check if the amount is greater than the current highest bid if not we revert back the amount that is bid . If the amount bid is greater we store the previous bidders address and the amount he bid and store the current highest bidder and his bid. The money of the previous highest bidder is not sent back directly as it might trigger unknown contracts so we allow the bidder to withdraw his money.

The account address and balances before the migration and deployment of the contracts.



The benificary account here by default the account[0] pays for the migration and deployment of the contract for the auction

Migration cost= gas used* gas price

=188483wei*20gwei

=188483*20000000000

=3769660000000000wei

Deploy cost= gas used* gas price

=534340wei*20gwei

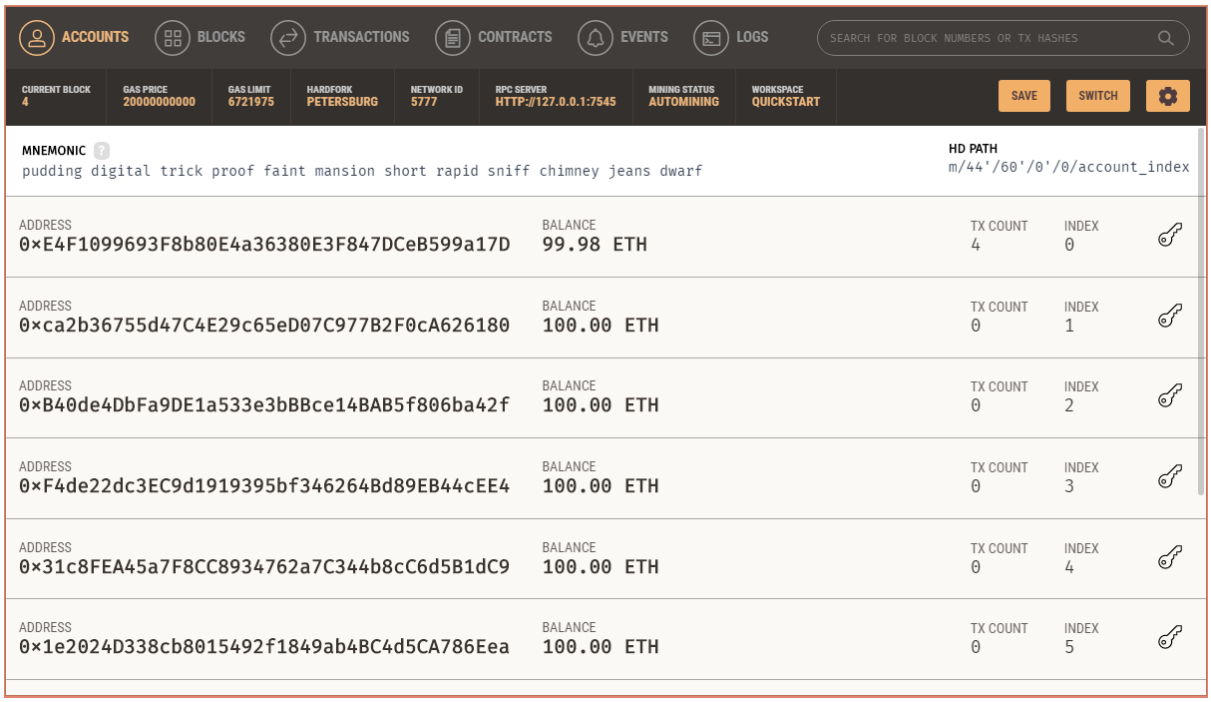
=534340*20000000000

=10686800000000000wei

Total Cost

144564600000000000wei

Before bidding begins:



The method bid () is called from account 2 and a bid of 5 ether is placed

The Transaction fee for bid: gas used* gas price

=gas used*gas price

=62350wei*20gwei

=62350*20000000000

=1247000000000000wei

Final account 2 balance after bidding and paying transaction fee

Another bid is placed from account 3 and a bid of 6 ether is placed

```
truffle(ganache)> instance.bid({from:accounts[3],value:web3.utils.toWei('6','ether')})
{
  tx: '0x65e58b22cb46807d4b87dccefca569c4e0a6ce4e41dec74048ac90bbdc0618e9',
  receipt: {
    transactionHash: '0x65e58b22cb46807d4b87dccefca569c4e0a6ce4e41dec74048ac90bbdc0618e9',
    transactionIndex: 0,
    blockHash: '0x1f57633ceb44d09bc5c3cdec8a08996a1c6dad96c01eaf3c6ca095da10ce7669',
    blockNumber: 6,
    from: '0xf4de22dc3ec9d1919395bf346264bd89eb44cee4',
    to: '0x3528563440831le14995b5561db3c424c662eaf0',
    gasUsed: 53109,
    cumulativeGasUsed: 53109,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    v: '0x1b',
    r: '0xfa22942f840403542395b731d52603bc81ac01ccfa6b1edca2a3a1b9e0f2b395',
    s: '0x3fbb83c19a2884c13305c310bcff451f062375d18bc5aa5dcbc53c22f85c3e50',
    rawLogs: []
  },
  logs: []
}
```

Transaction fee : gas used*gas price

=53109wei*20gwei

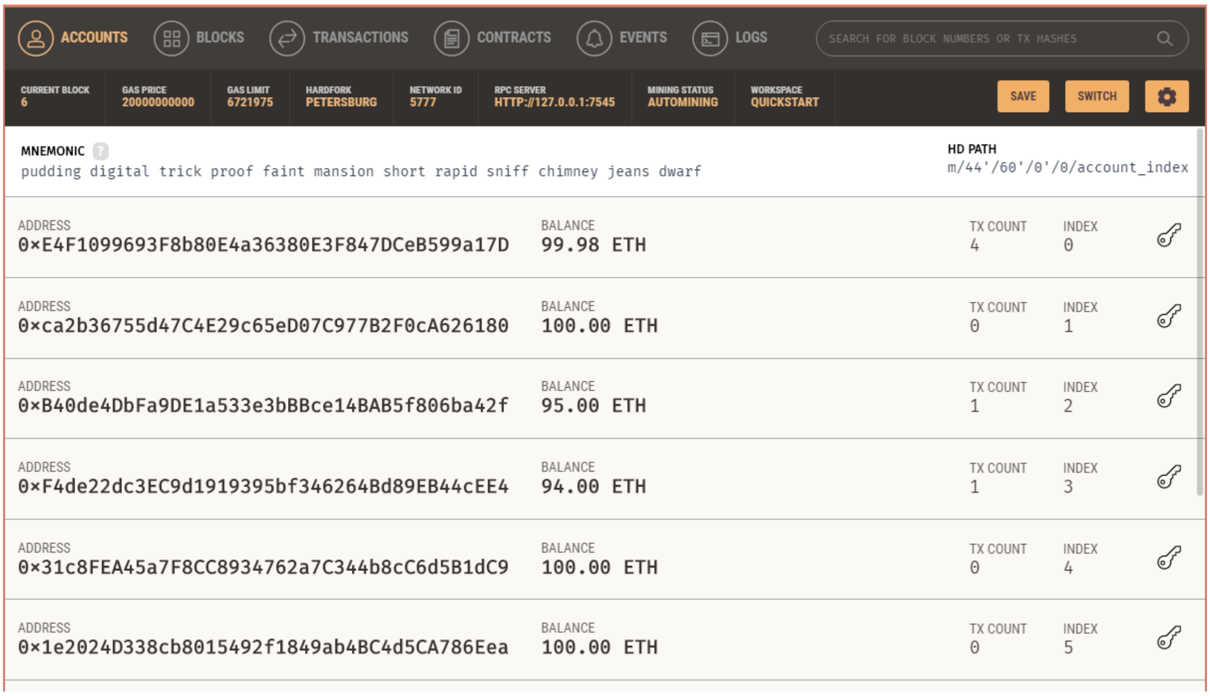
=53109*20000000000

=1062180000000000wei

Final account 3 balance after bidding and paying transaction fee

```
truffle(ganache)> web3.eth.getBalance(accounts[3])
'93998937820000000000'
```

Balance as reflected in ganache:



Function withdraw ():

We first store the amount that is pending for the function caller in a separate variable for future reference. If there is pending amount for the method caller then we first make the pending amount of that caller as zero to prevent re-entry attack and claiming the same amount again. If the amount cannot be sent back due to error we again update the pending amount for the caller by using the reference variable we had used earlier and return false that withdraw could not happen.

Here in my example account 2 wants to withdraw its bid of 5 ethers

Transaction fee: gas used*gas price

=19465wei*20gwei

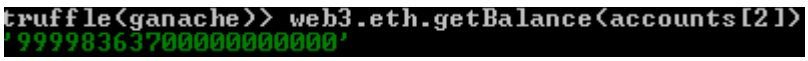
=19465*20000000000

=389300000000000wei

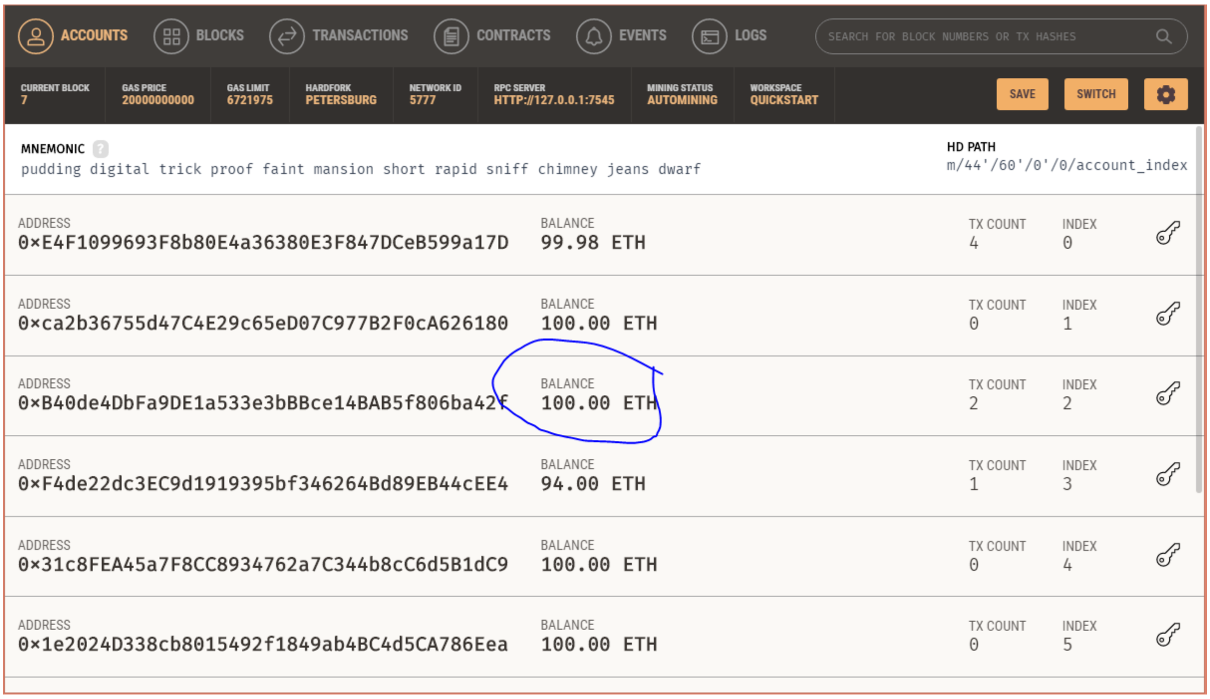
New balance = Previous balance + returned amount -transaction fees

=94998753000000000000 + 5*(Math.pow(10,18) – 389300000000000

=99998363700000000000



The balance updated for account 2 after withdraw



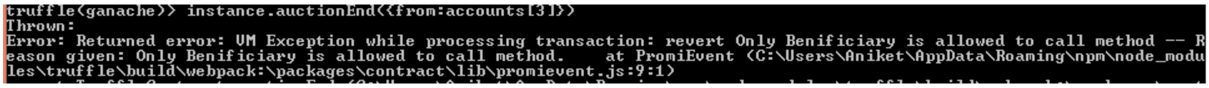
If the same account tries to withdraw again the account balance des not increase. the account pays for the gas fee of executing the method and method call ends

```
truffle(ganache)> instance.withdraw({from:accounts[2]})
{
  tx: '0x4f83cdce5495c708b581c8cfc6aca2356257cd882e5802e6789c81735304ce96',
  receipt: {
    transactionHash: '0x4f83cdce5495c708b581c8cfc6aca2356257cd882e5802e6789c81735304ce96',
    transactionIndex: 0,
    blockHash: '0xfbc9eeaa251f071b242e26dc10e7f00f75f5d636ceae175a513e48c01880f661',
    blockNumber: 8,
    from: '0xb40de4dbfa9de1a533e3bbbce14bab5f806ba42f',
    to: '0x35285634408311e14995b5561db3c424c662eaf0',
    gasUsed: 21870,
    cumulativeGasUsed: 21870,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    v: '0x1b',
    r: '0xb538f6080554d29b64d9c5aeba91d0e7fc41e0ccc1a17a68934f34d9c80df279',
    s: '0x2027500acb0a276c7ae7c94219f614b0f343c586bead7da72f96f57c9e50b4c7',
    rawLogs: []
  },
  logs: []
}
truffle(ganache)> web3.eth.getBalance(accounts[2])
'99997926300000000000'
truffle(ganache)>
```

Function auctionEnd ():

The auction end method can only be invoked by the beneficiary account. We check for this first and revert the transaction if that is not the case. We also check if the auction is active or not and if the beneficiary is trying to make multiple calls. If all the require methods do not revert the calls then we first set the auctionEnded Boolean variable as true and then transfer the highest bid amount to the beneficiary.

If some other account invokes auctionEnd method then there is an error and transaction is reverted.



```
truffle(ganache)> instance.auctionEnd({from:accounts[3]})
Thrown:
Error: Returned error: VM Exception while processing transaction: revert Only Benificiary is allowed to call method -- R
eason given: Only Benificiary is allowed to call method.    at PromiEvent (C:\Users\Aniket\AppData\Roaming\npm\node_modu
les\truffle\build\webpack:\packages\contract\lib\promievent.js:9:1)
```

When auctionEnd is invoked by the beneficiary which is account 0 :

```
truffle(ganache)> instance.auctionEnd({from:accounts[0]})
{
  tx: '0x1046063ef6408b97d01accb5d496fddad14fc403b900f4462211eea134ce2e79',
  receipt: {
    transactionHash: '0x1046063ef6408b97d01accb5d496fddad14fc403b900f4462211eea134ce2e79',
    transactionIndex: 0,
    blockHash: '0x1925ea640f53e3fddbfeff571ce083a3a5bec8fe5cdb43c79e54090c63424b1a',
    blockNumber: 10,
    from: '0xe4f1099693f8b80e4a36380e3f847dceb599a17d',
    to: '0x35285634408311e14995b5561db3c424c662eaf0',
    gasUsed: 50149,
    cumulativeGasUsed: 50149,
    contractAddress: null,
    logs: [],
    status: true,
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000',
    v: '0x1b',
    r: '0xd04d382da9775fecb60c6ece730d62e12dfa3b061c0b879e4f5967880e43e4d9',
    s: '0x278fb92b100f86dbbfa47aba9f63dac26c64346687642b8956580fc55d2fd883',
    rawLogs: []
  },
  logs: []
}
```

Transaction fee: gas used* gas Price

=50149wei*20gwei

=50149*2

=1002980000000000

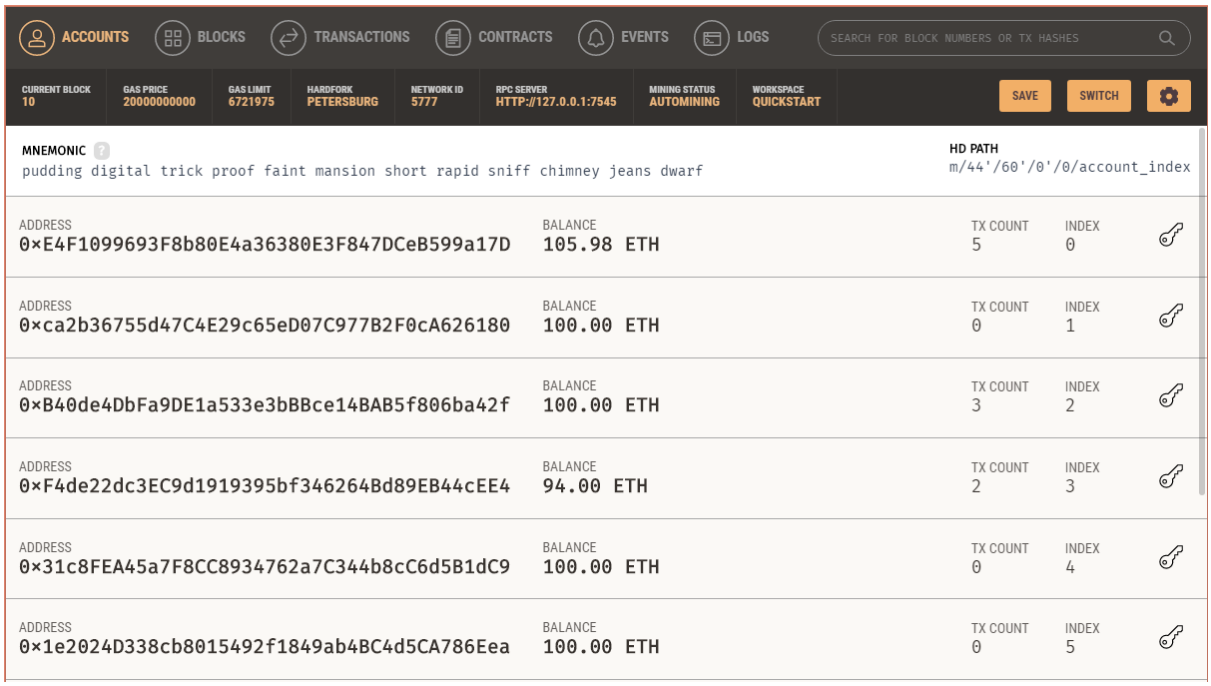
Total Benificairy balance = Previous balance + highest bid – transaction fee

=99983160520000000000 + 6*(Math.pow(10,18)) - 1002980000000000

=105983160520000000000

```
truffle(ganache)> web3.eth.getBalance(accounts[0])
'105983160520000000000'
```

The ganache interface showing the updated balances:



If the beneficiary tries to call auctionEnd again after receiving the amount then we get an error.