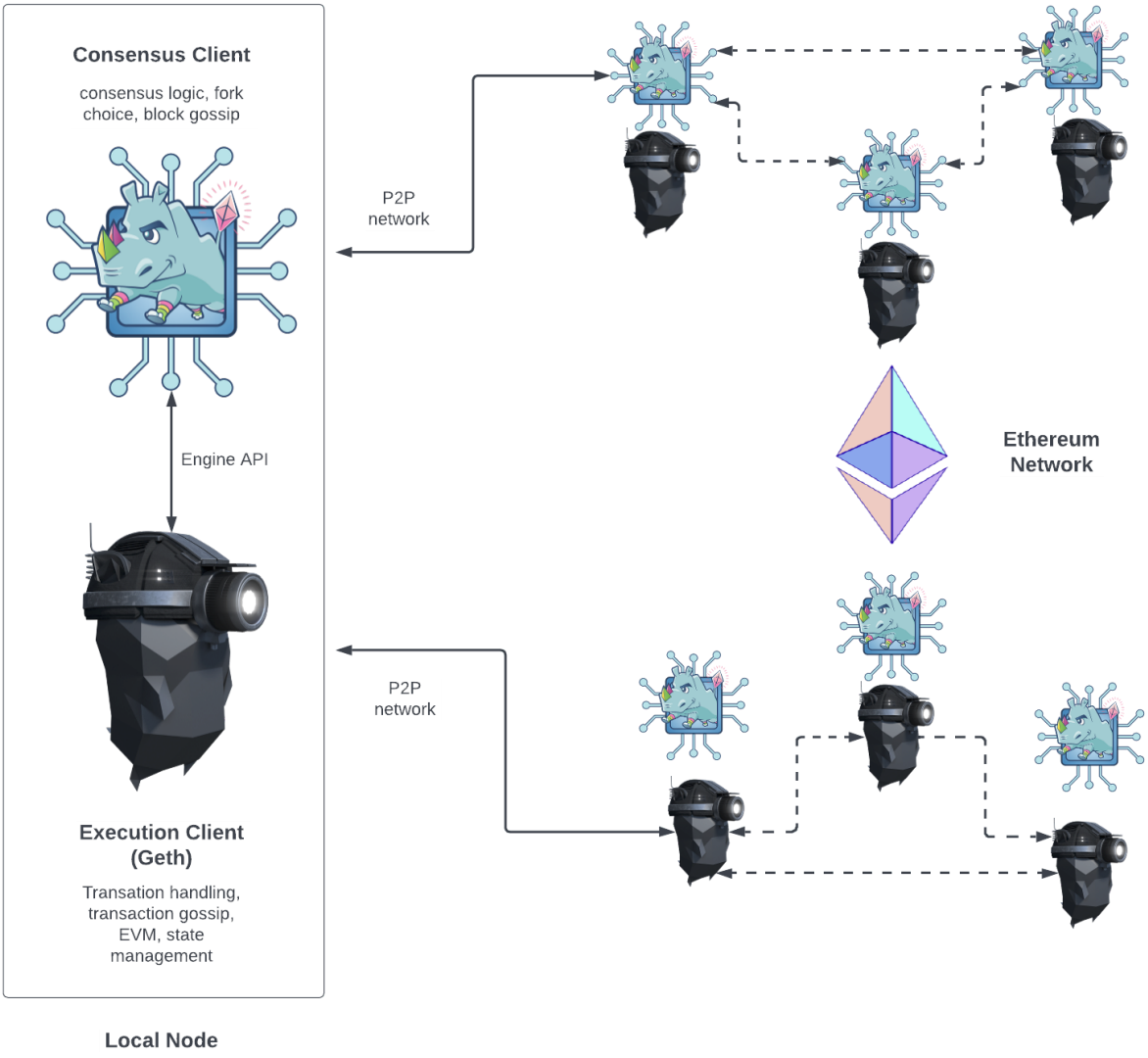
Name: Prasad Jawale **Class**: D16AD **Roll**: 20

Blockchain Lab 1

Aim: To install and set up an Ethereum network to create a private Ethereum blockchain for development and testing purposes.

Theory: Ethereum node is composed of two clients: an execution client and a consensus client. Geth is an execution client. Originally, an execution client alone was enough to run a full Ethereum node. However, ever since Ethereum turned off proof-of-work and implemented proof-of-stake, Geth has needed to be coupled to another piece of software called a “consensus client” in order to keep track of the Ethereum blockchain.The execution client (Geth) is responsible for transaction handling, transaction gossip, state management and supporting the Ethereum Virtual Machine EVM. However, Geth is not responsible for block building, block gossiping or handling consensus logic. These are in the remit of the consensus client.The relationship between the two Ethereum clients is shown in the schematic below. The two clients each connect to their own respective peer-to-peer (P2P) networks. This is because the execution clients gossip transactions over their P2P network enabling them to manage their local transaction pool. The consensus clients gossip blocks over their P2P network, enabling consensus and chain growth.

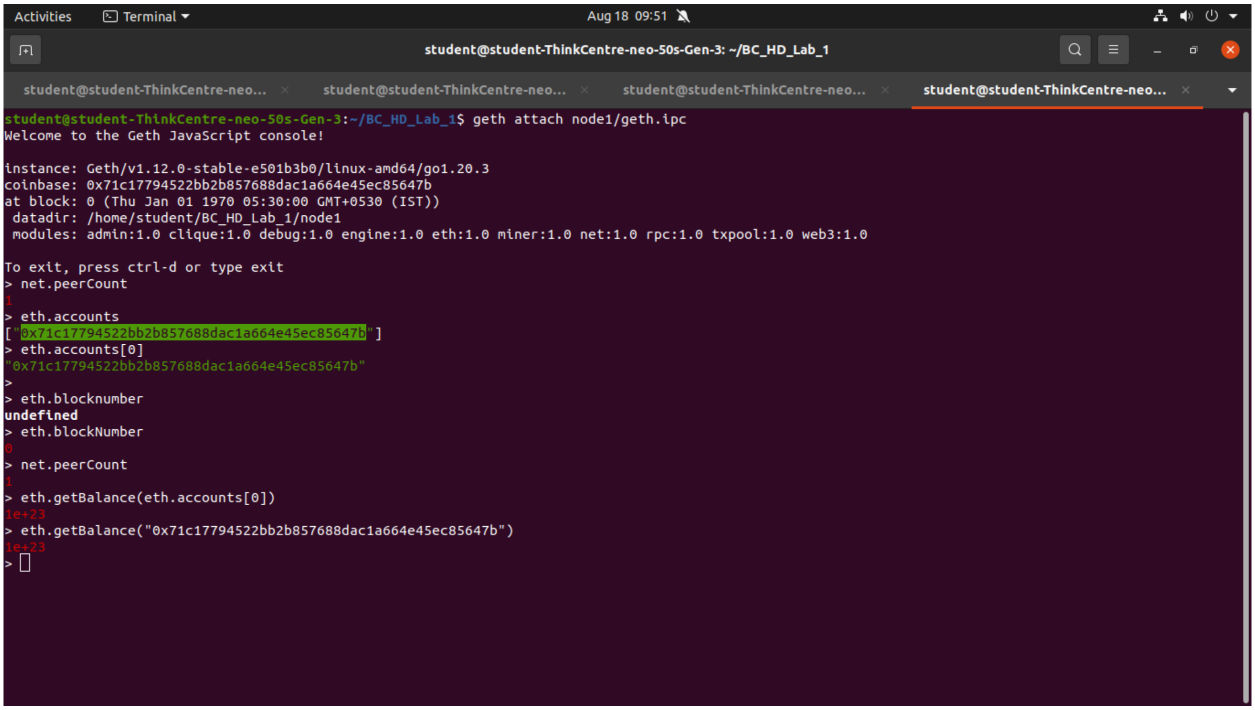
Node-architecture:

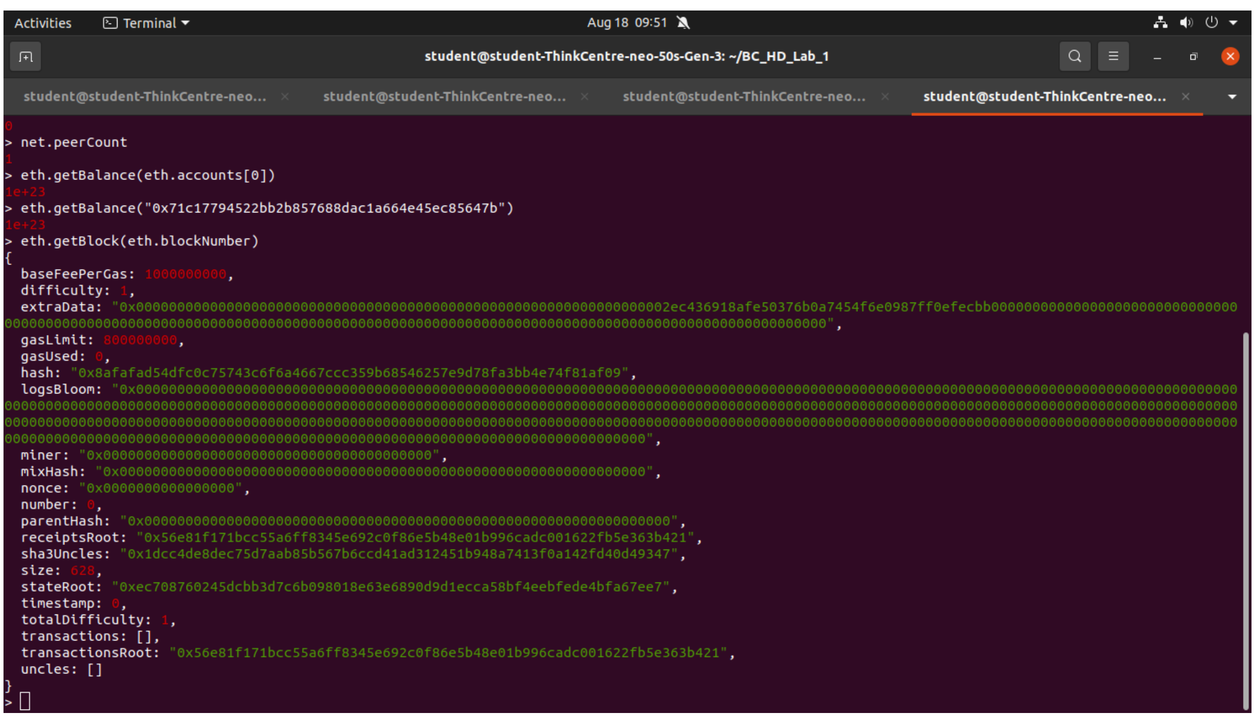


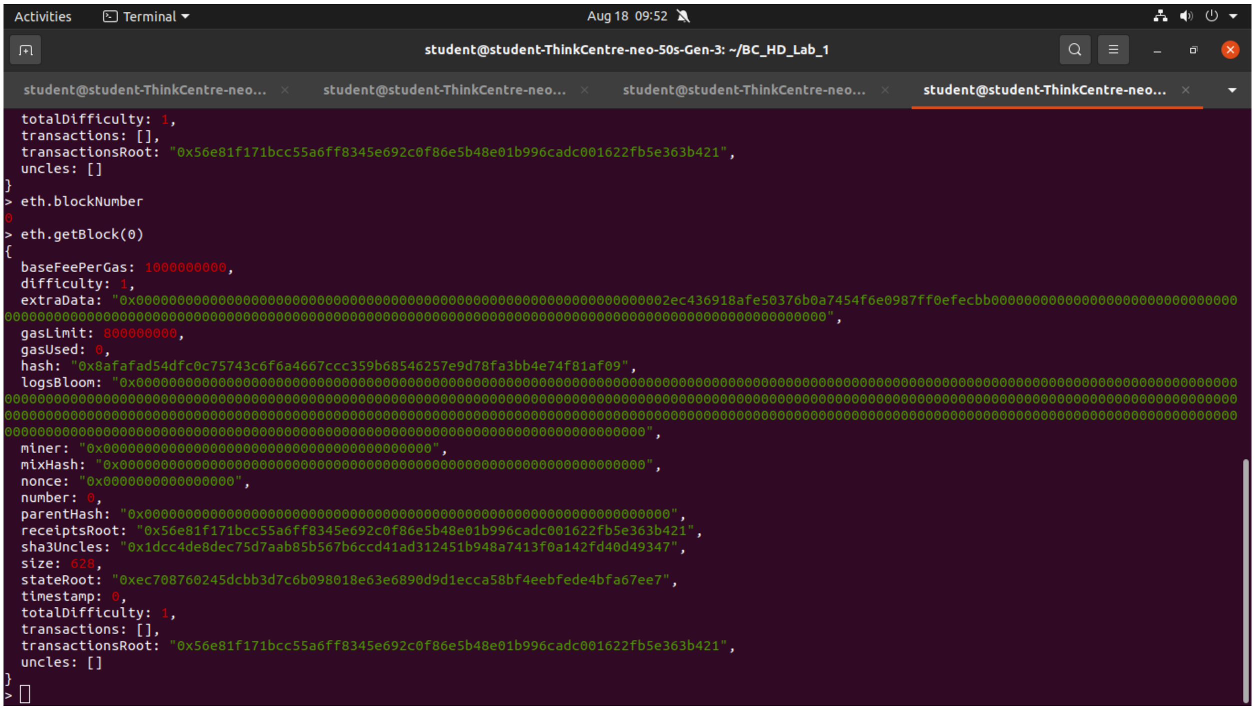
For this two-client structure to work, consensus clients must be able to pass bundles of transactions to Geth to be executed. Executing the transactions locally is how the client validates that the transactions do not violate any Ethereum rules and that the proposed update to Ethereum's state is correct. Likewise, when the node is selected to be a block producer the consensus client must be able to request bundles of transactions from Geth to include in the new block. This inter-client communication is handled by a local RPC connection using the engine API.

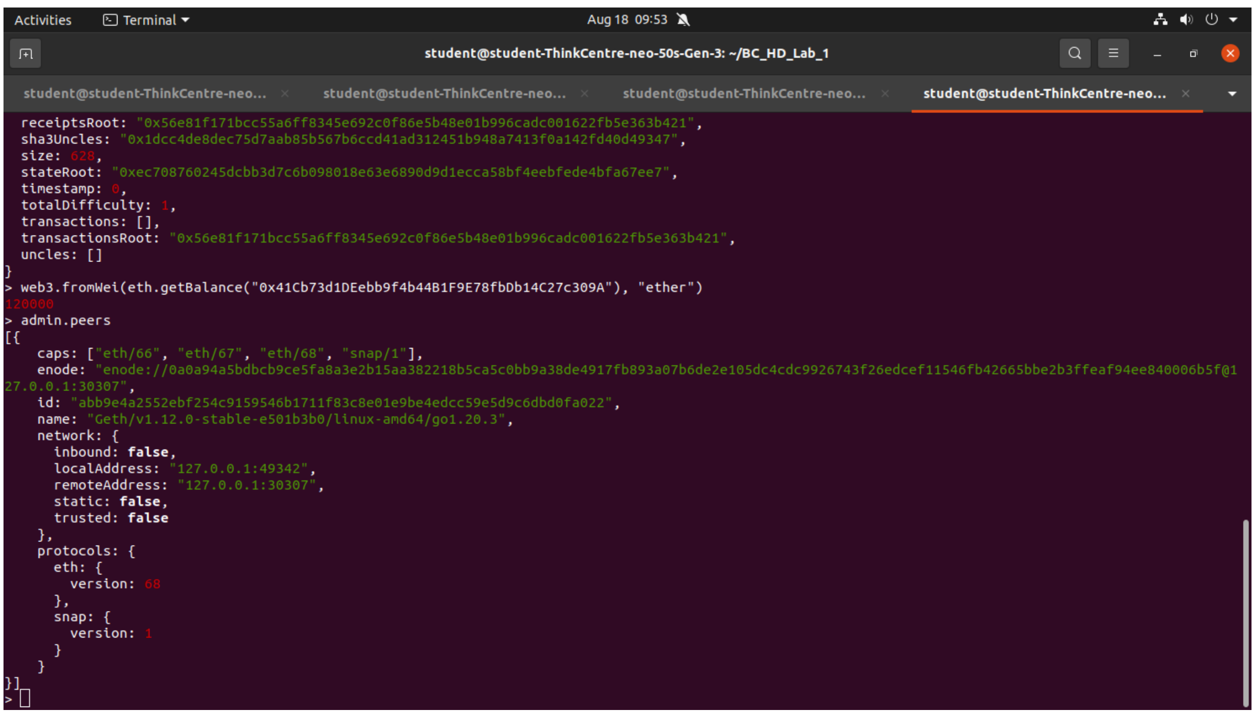
Program:

1. Geth commands

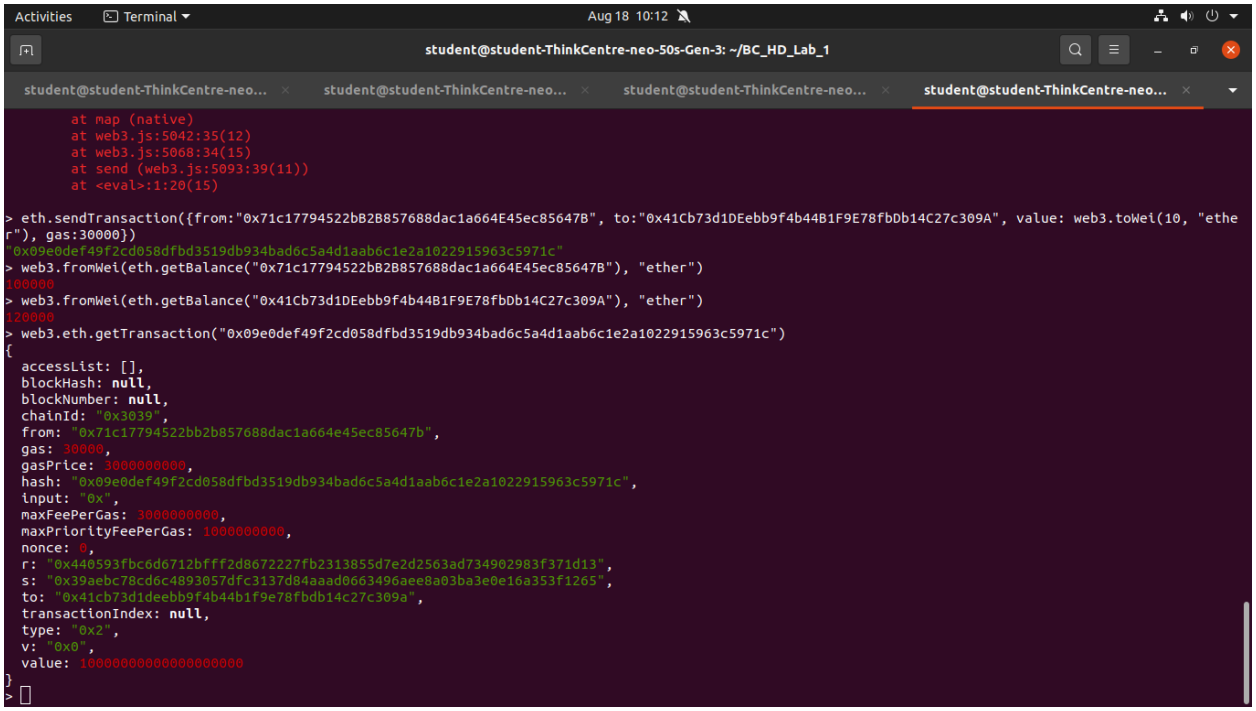




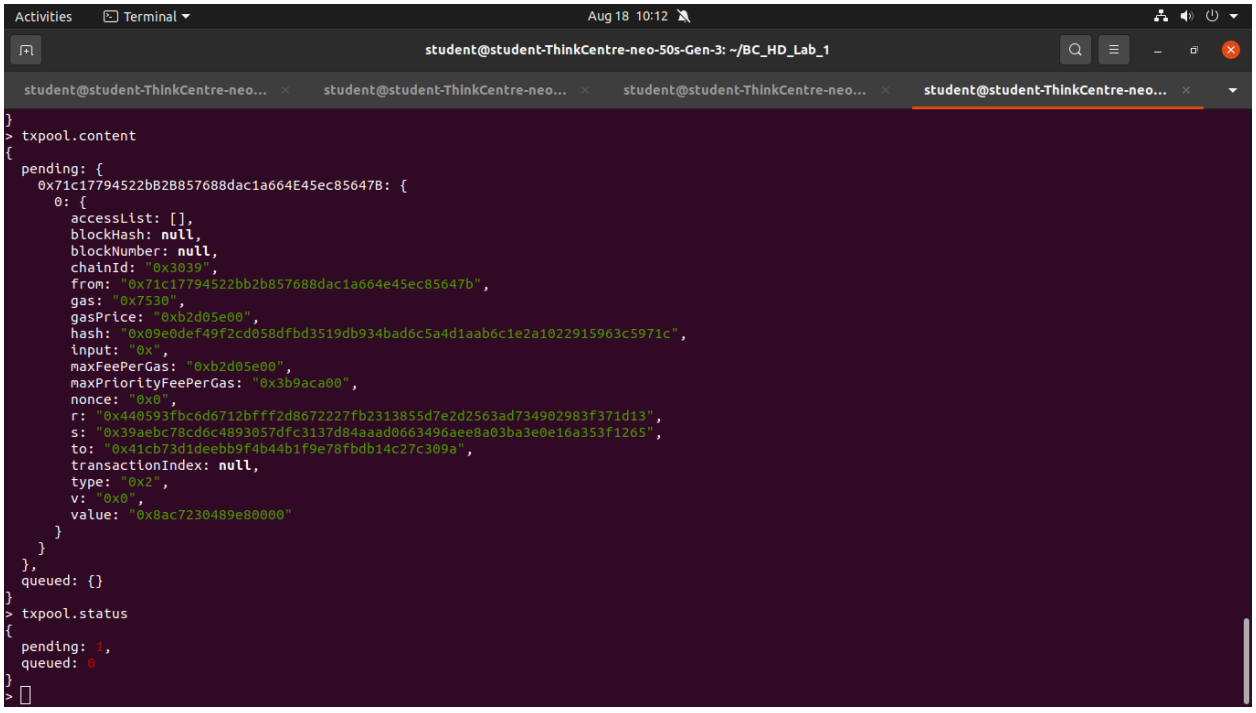

Activities
Terminal
Aug 18 09:52
student@student-ThinkCentre-neo-50s-Gen-3: ~/BC_HD_Lab_1
student@student-ThinkCentre-neo...
student@student-ThinkCentre-neo...
student@student-ThinkCentre-neo...
student@student-ThinkCentre-neo...
totalDifficulty: 1,
transactions: [],
transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
uncles: []
}
> eth.blockNumber
0
> eth.getBlock(0)
{
baseFeePerGas: 1000000000,
difficulty: 1,
extraData: "0x00000000000000000000000000000000000000000000000000000000000000002ec436918afe50376b0a7454f6e0987ff0efecbb0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
gasLimit: 800000000,
gasUsed: 0,
hash: "0x8afafad54dfc0c75743c6f6a4667ccc359b68546257e9d78fa3bb4e74f81af09",
logsBloom: "0x00",
miner: "0x0000000000000000000000000000000000000000",
mixHash: "0x0000000000000000000000000000000000000000000000000000000000000000",
nonce: "0x0000000000000000",
number: 0,
parentHash: "0x0000000000000000000000000000000000000000000000000000000000000000",
receiptsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
size: 628,
stateRoot: "0xec708760245dcbb3d7c6b098018e63e6890d9d1ecca58bf4eebfede4bfa67ee7",
timestamp: 0,
totalDifficulty: 1,
transactions: [],
transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
uncles: []
}
>

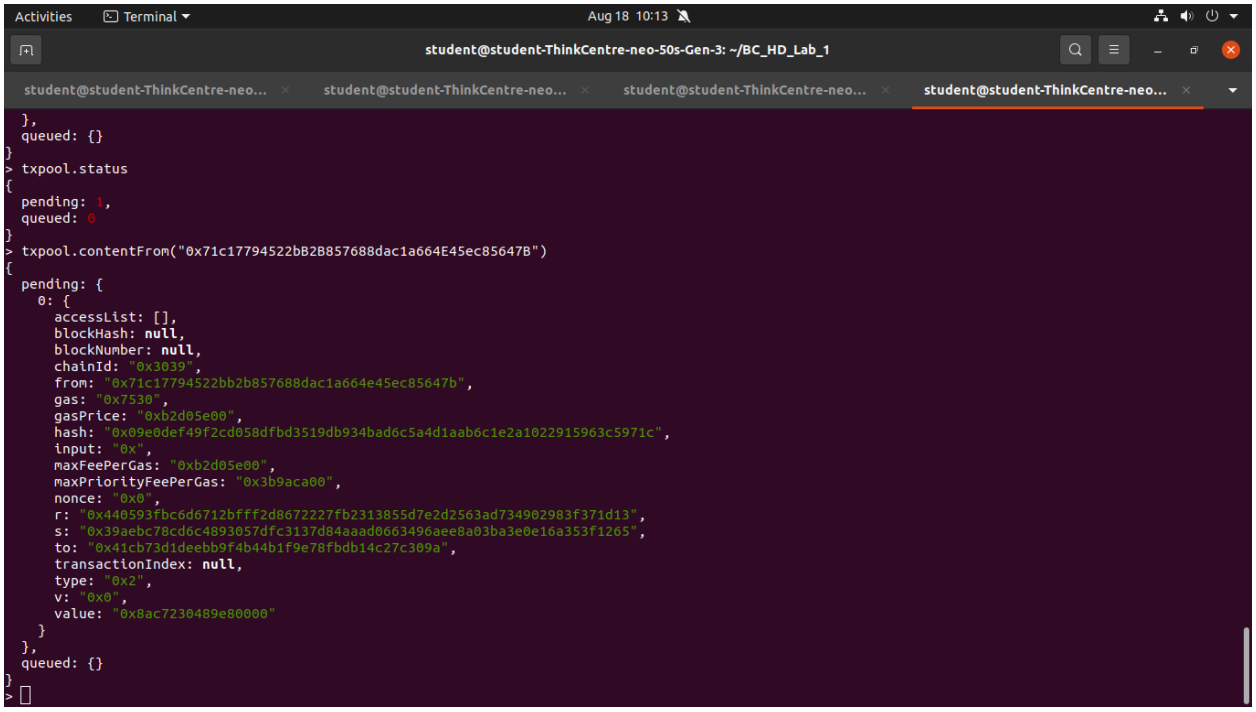

Activities
Terminal
Aug 18 09:53
student@student-ThinkCentre-neo-50s-Gen-3: ~/BC_HD_Lab_1
student@student-ThinkCentre-neo...
student@student-ThinkCentre-neo...
student@student-ThinkCentre-neo...
student@student-ThinkCentre-neo...
receiptsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347",
size: 628,
stateRoot: "0xec708760245dcbb3d7c6b098018e63e6890d9d1ecca58bf4eebfede4bfa67ee7",
timestamp: 0,
totalDifficulty: 1,
transactions: [],
transactionsRoot: "0x56e81f171bcc55a6ff8345e692c0f86e5b48e01b996cadc001622fb5e363b421",
uncles: []
}
> web3.fromWei(eth.getBalance("0x41Cb73d1DEebb9f4b44B1F9E78fbDb14C27c309A"), "ether")
120000
> admin.peers
[{
caps: ["eth/66", "eth/67", "eth/68", "snap/1"],
enode: "enode://0a0a94a5bdbcb9ce5fa8a3e2b15aa382218b5ca5c0bb9a38de4917fb893a07b6de2e105dc4cdc9926743f26edcef11546fb42665bbe2b3ffeaf94ee840006b5f@127.0.0.1:30307",
id: "abb9e4a2552ebf254c9159546b1711f83c8e01e9be4edcc59e5d9c6dbd0fa022",
name: "Geth/v1.12.0-stable-e501b3b0/linux-amd64/go1.20.3",
network: {
inbound: false,
localAddress: "127.0.0.1:49342",
remoteAddress: "127.0.0.1:30307",
static: false,
trusted: false
},
protocols: {
eth: {
version: 68
},
snap: {
version: 1
}
}
}]
>

2. Transaction from Node1 to Node2



3. Content of transaction pool





Conclusion : Hence we have successfully implemented the Ethereum network for testing and development purpose .