**Shashwat Tripathi**

**D20A Roll No: 64**

**Blockchain Lab**

**Experiment 1**

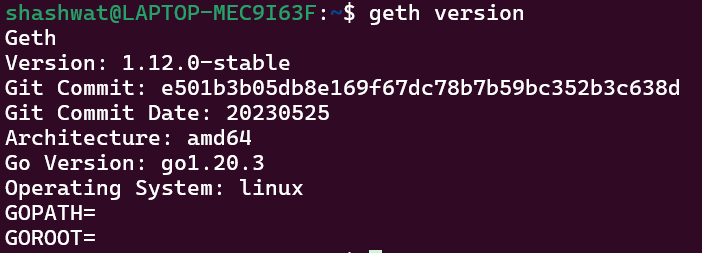
**Aim:** To install and set up an Ethereum network to create a private Ethereum blockchain for development and testing purposes.

**Steps:**

**1. Installing Geth on Ubuntu**



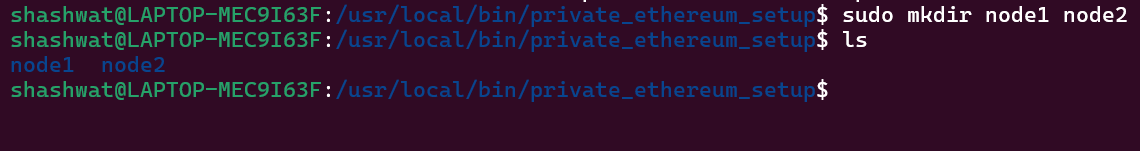
**2. Check the version of Geth on the Terminal**



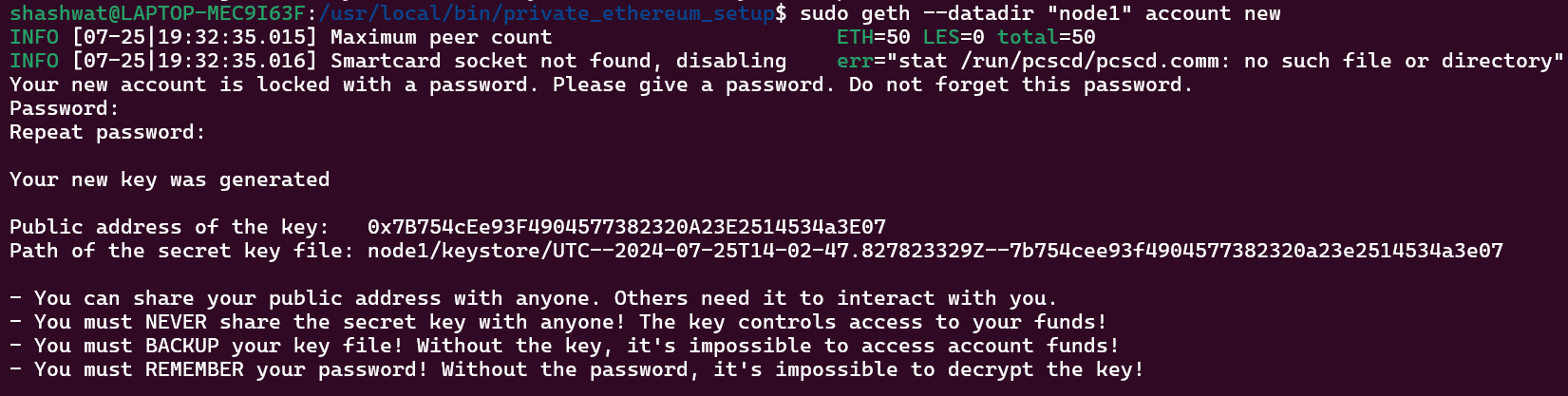
**3. Create a Private ethereum network**

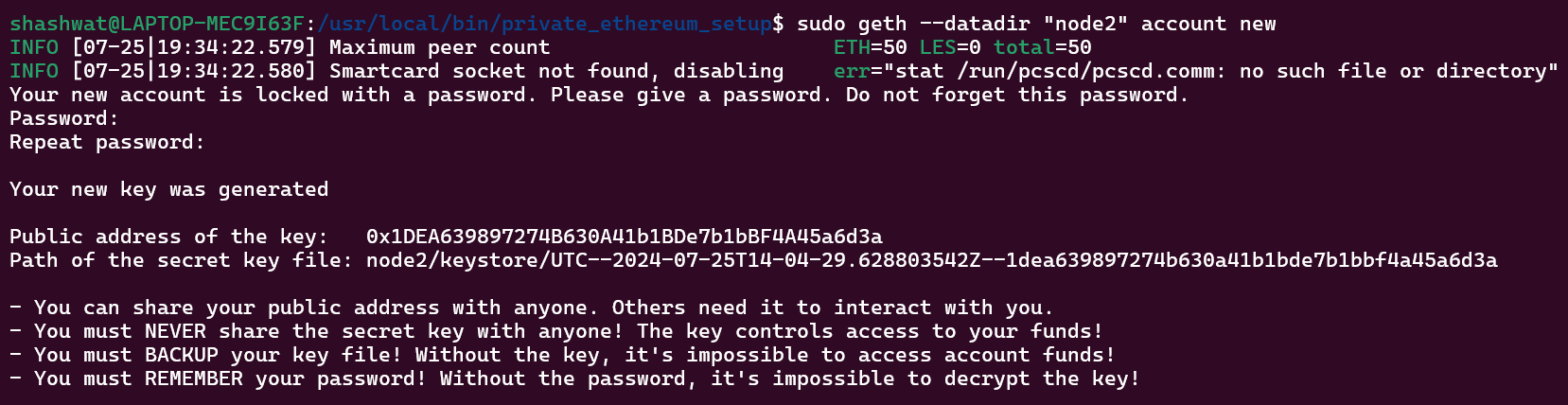
**1. Create a folder named, private\_ethereum\_setup**

**2. Create 2 subfolders named node1 and node2 in the folder private\_ethereum\_setup**



**3. Create 2 accounts in the folder corresponding to node1 and node2**

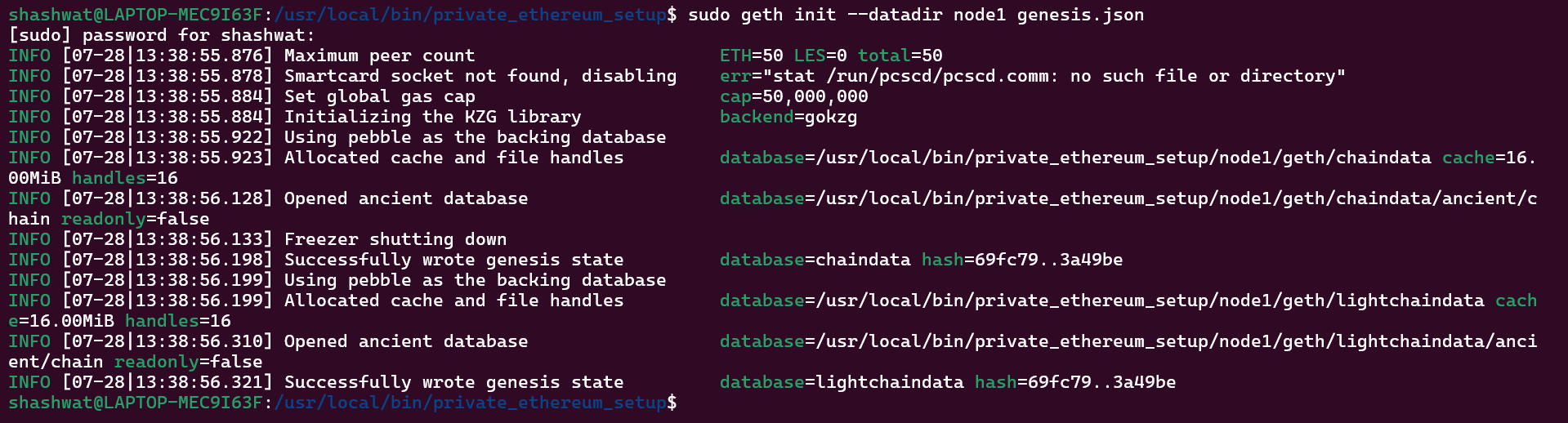


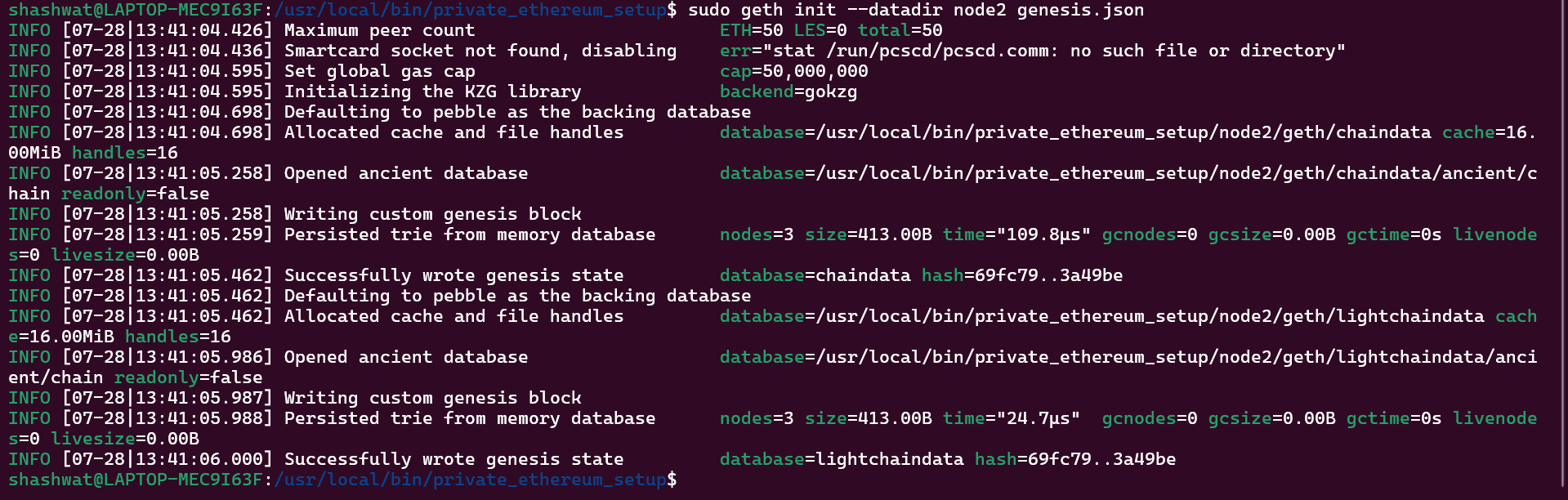


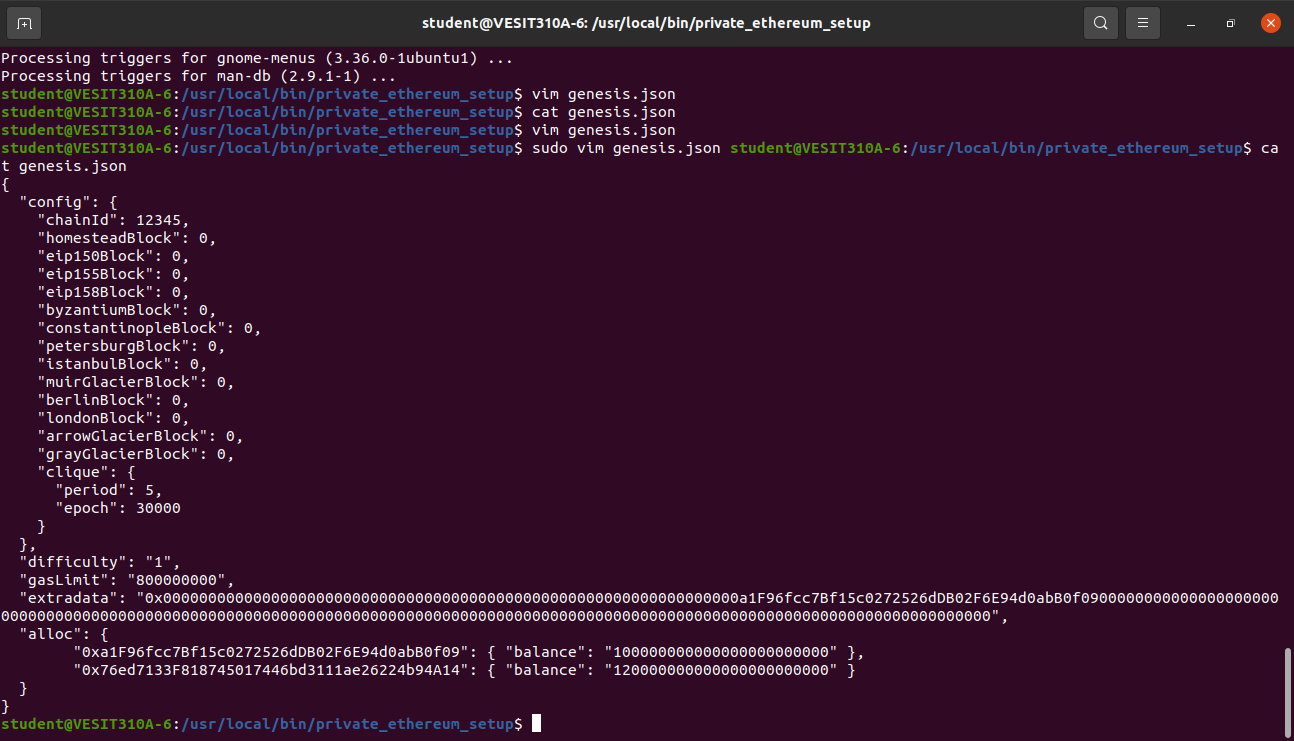
**4. Create a genesis.json file in the folder, private\_ethereum\_setup**

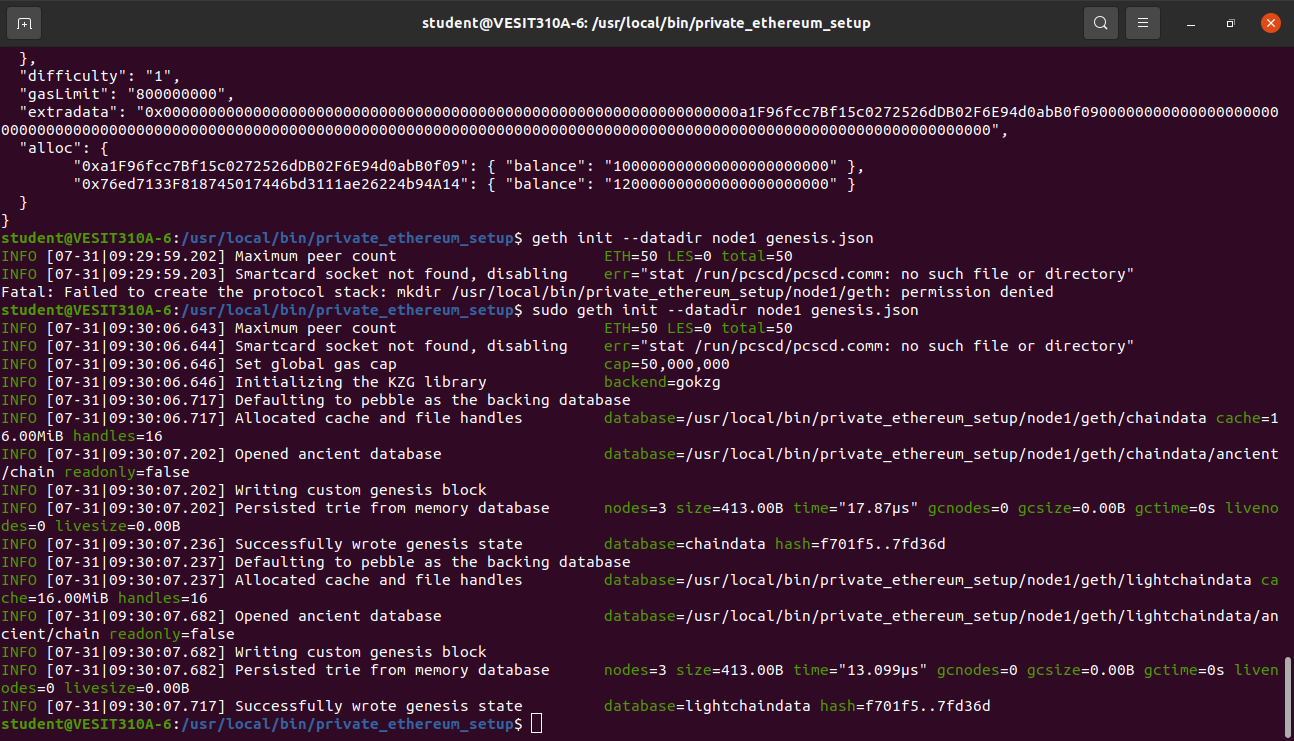


**5. Initialize the nodes with the genesis file**









Node 1

Public address of the key: 0x7584315bBdd1253A79E83E66388405cB714b54b9

Path of the secret key file: node1/keystore/UTC--2024-07-31T04-15-11.907190774Z--7584315bbdd1253a79e83e66388405cb714b54b9

Node 2

Public address of the key: 0x138F572b8A7F1326B3362ffA53338a1EA193b248

Path of the secret key file: node2/keystore/UTC--2024-07-31T04-15-58.836877163Z--138f572b8a7f1326b3362ffa53338a1ea193b248

Genesis.json

{

"config": {

"chainId": 12345,

"homesteadBlock": 0,

"eip150Block": 0,

"eip155Block": 0,

"eip158Block": 0,

"byzantiumBlock": 0,

"constantinopleBlock": 0,

"petersburgBlock": 0,

"istanbulBlock": 0,

"muirGlacierBlock": 0,

"berlinBlock": 0,

"londonBlock": 0,

"arrowGlacierBlock": 0,

"grayGlacierBlock": 0,

"clique": {

"period": 5,

"epoch": 30000

}

},

"difficulty": "1",

"gasLimit": "800000000",

"extradata": "0x00000000000000000000000000000000000000000000000000000000000000007584315bBdd1253A79E83E66388405cB714b54b90000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",

"alloc": {

"0x7584315bBdd1253A79E83E66388405cB714b54b9": { "balance": "100000000000000000000000" },

"0x138F572b8A7F1326B3362ffA53338a1EA193b248": { "balance": "120000000000000000000000" }

}

}

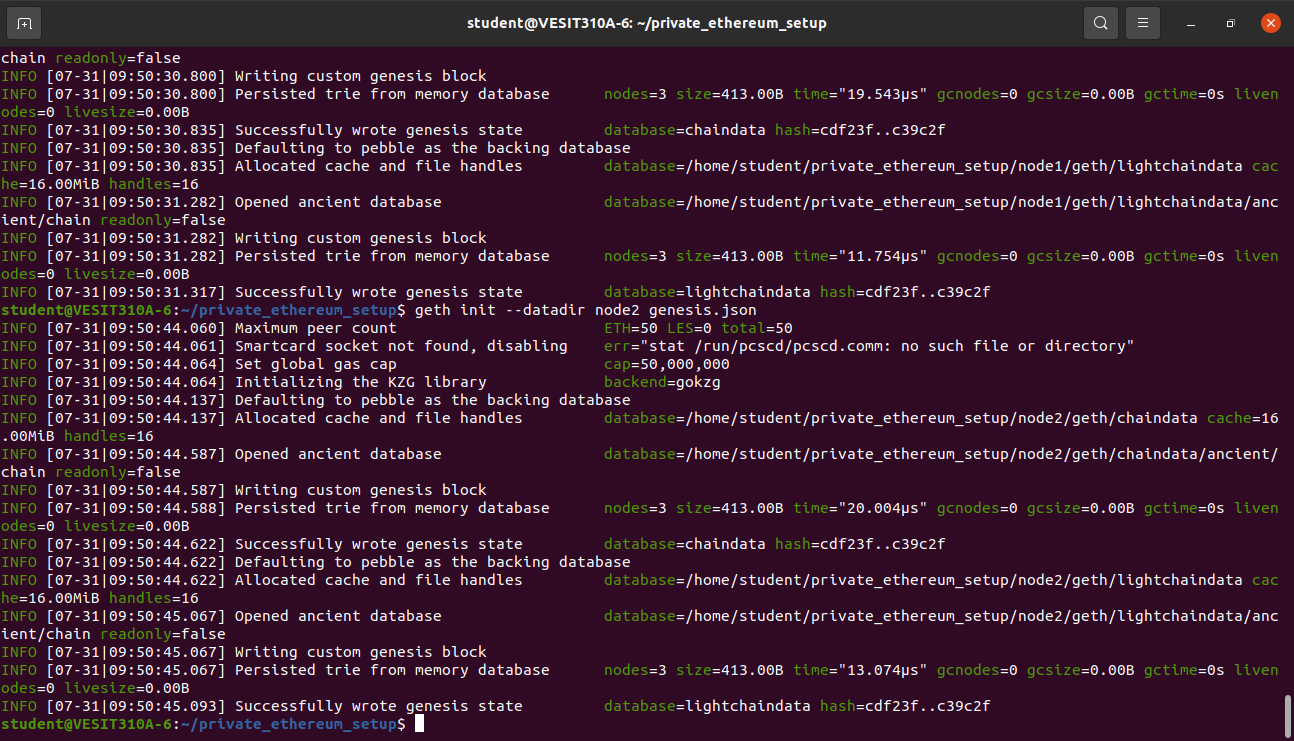
Run node1

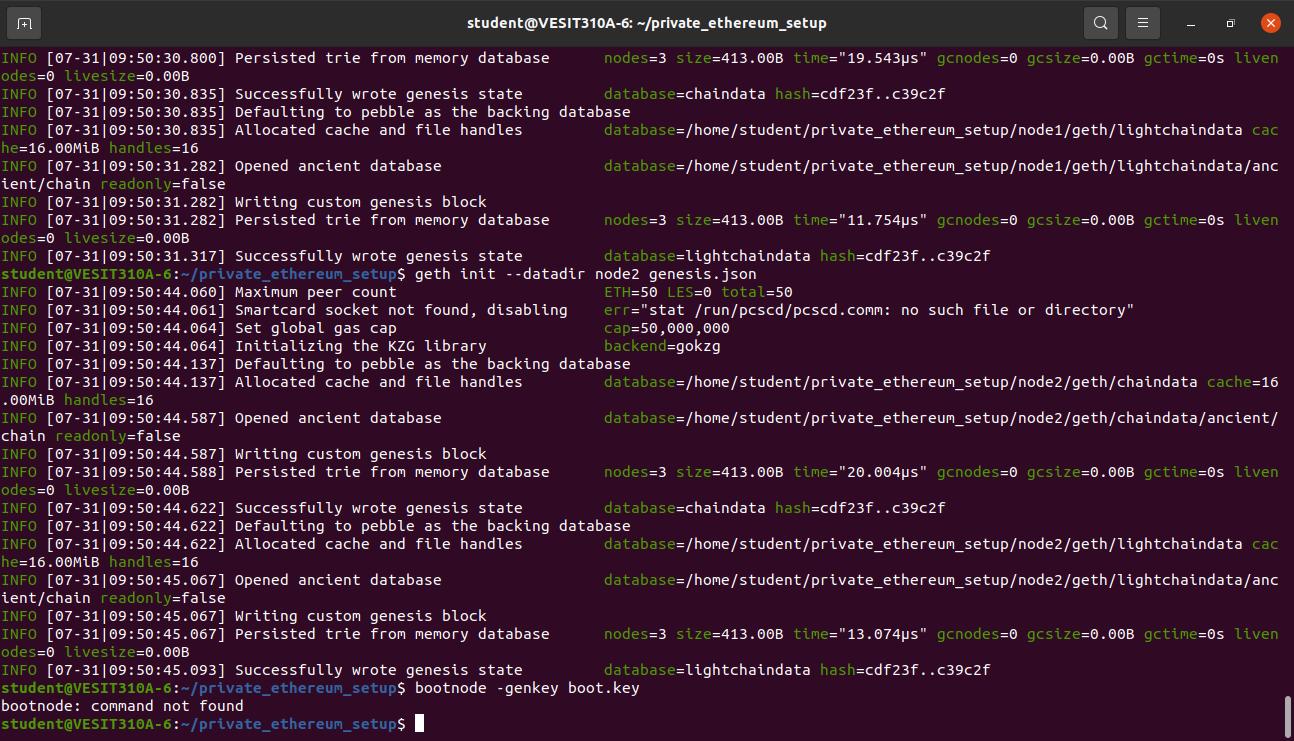
geth --datadir node1 --port 30306 --bootnodes enode://2bccaf4b4cf5d10f0e8b49cb68b3c3ad867b6cb40596c78a8b216ae8dd62a174457b9d8839364074047f749914e84b999b92485e988510edca153341a6f6107a@127.0.0.1:0?discport=30305 --networkid 123454321 --unlock 0x7584315bBdd1253A79E83E66388405cB714b54b9 --password node1/password.txt --authrpc.port 8551 --miner.etherbase 0x7584315bBdd1253A79E83E66388405cB714b54b9 --mine

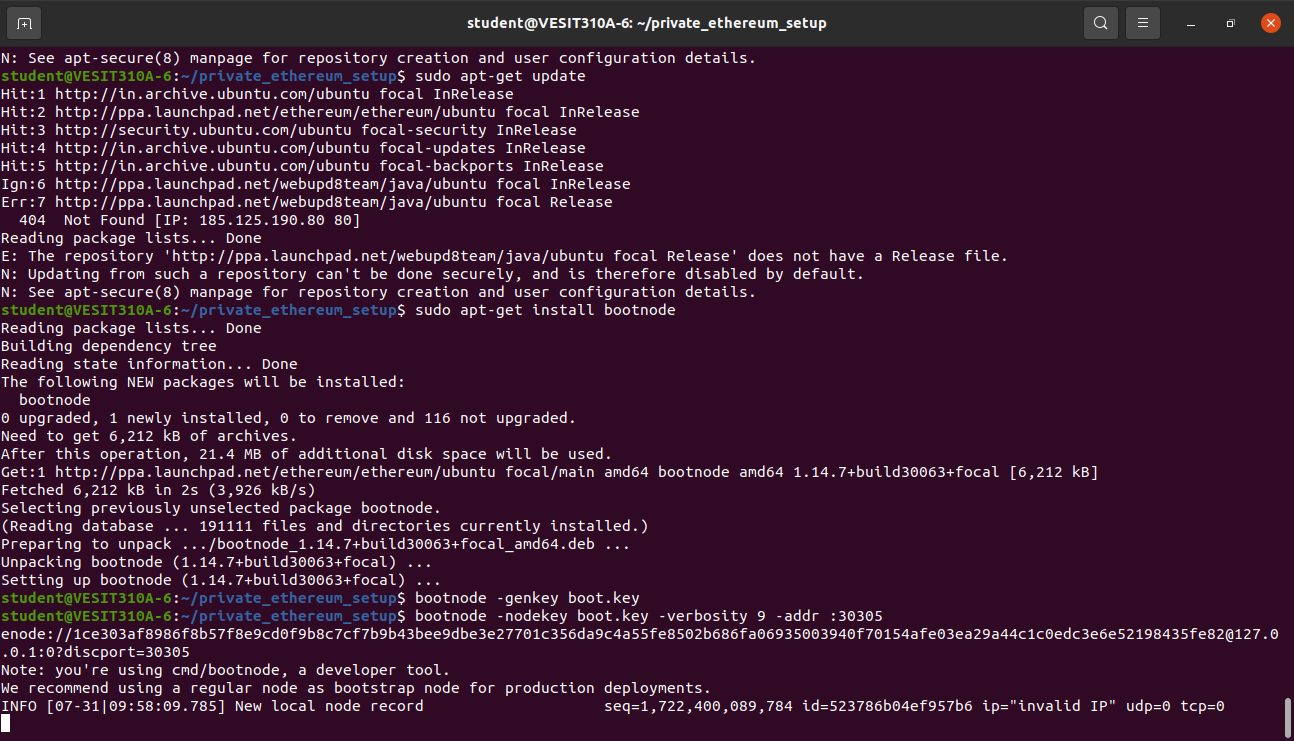
Run node 2

geth --datadir node2 --port 30307 --bootnodes enode://2bccaf4b4cf5d10f0e8b49cb68b3c3ad867b6cb40596c78a8b216ae8dd62a174457b9d8839364074047f749914e84b999b92485e988510edca153341a6f6107a@127.0.0.1:0?discport=30305 --networkid 123454321 --unlock 0x138F572b8A7F1326B3362ffA53338a1EA193b248 --password node2/password --authrpc.port 8552

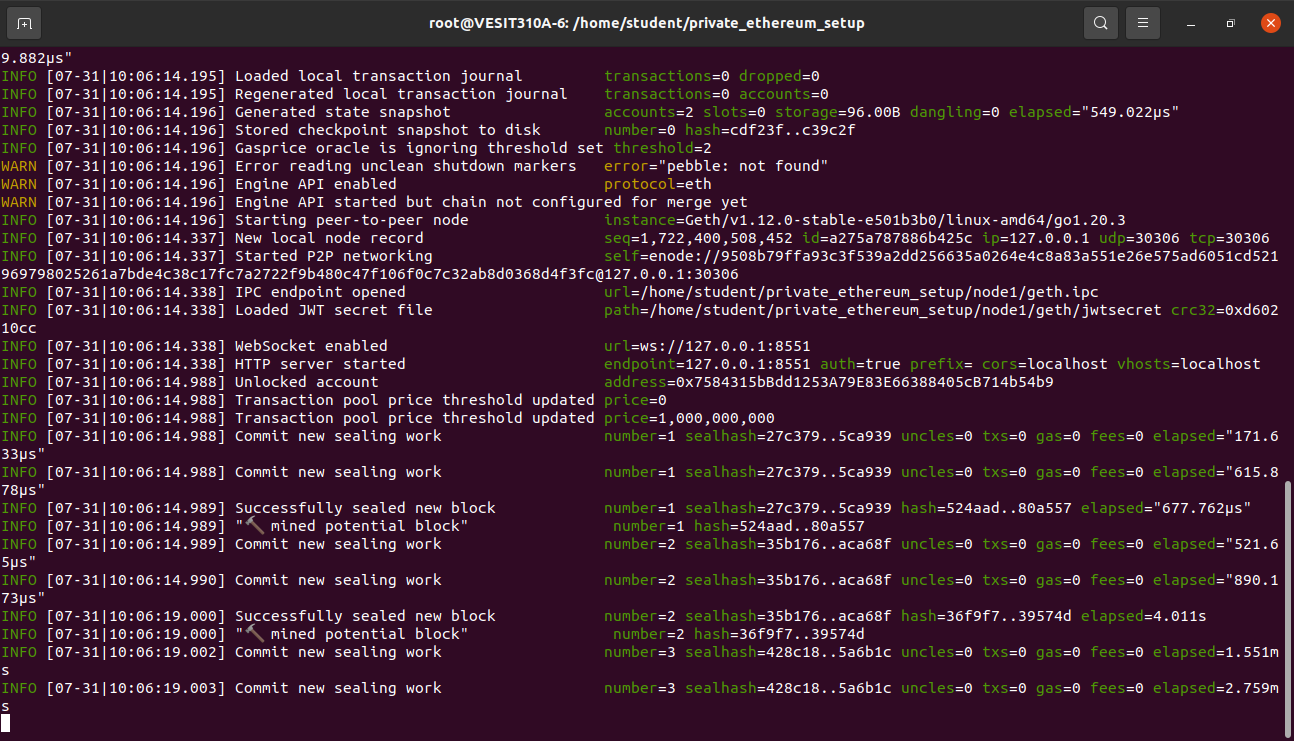
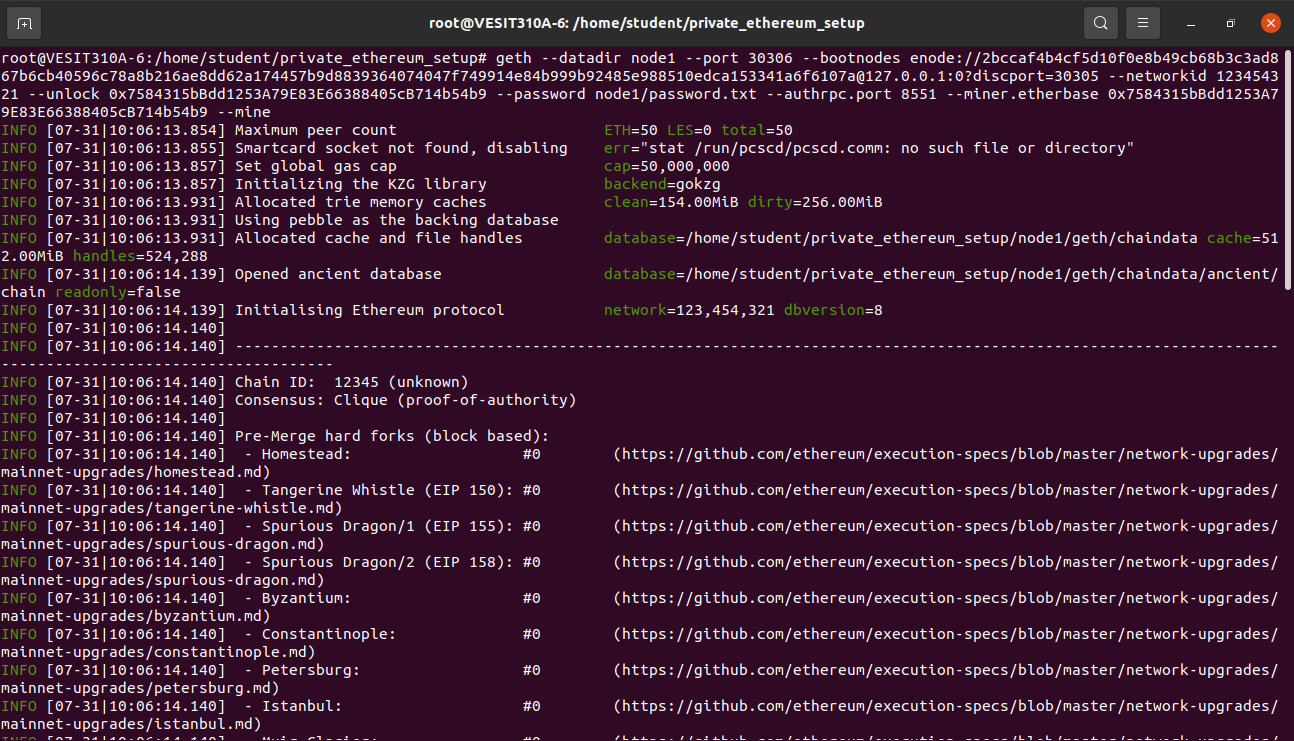
**Step 4: Establish a Peer-Peer Connection between the nodes along with the bootnode**



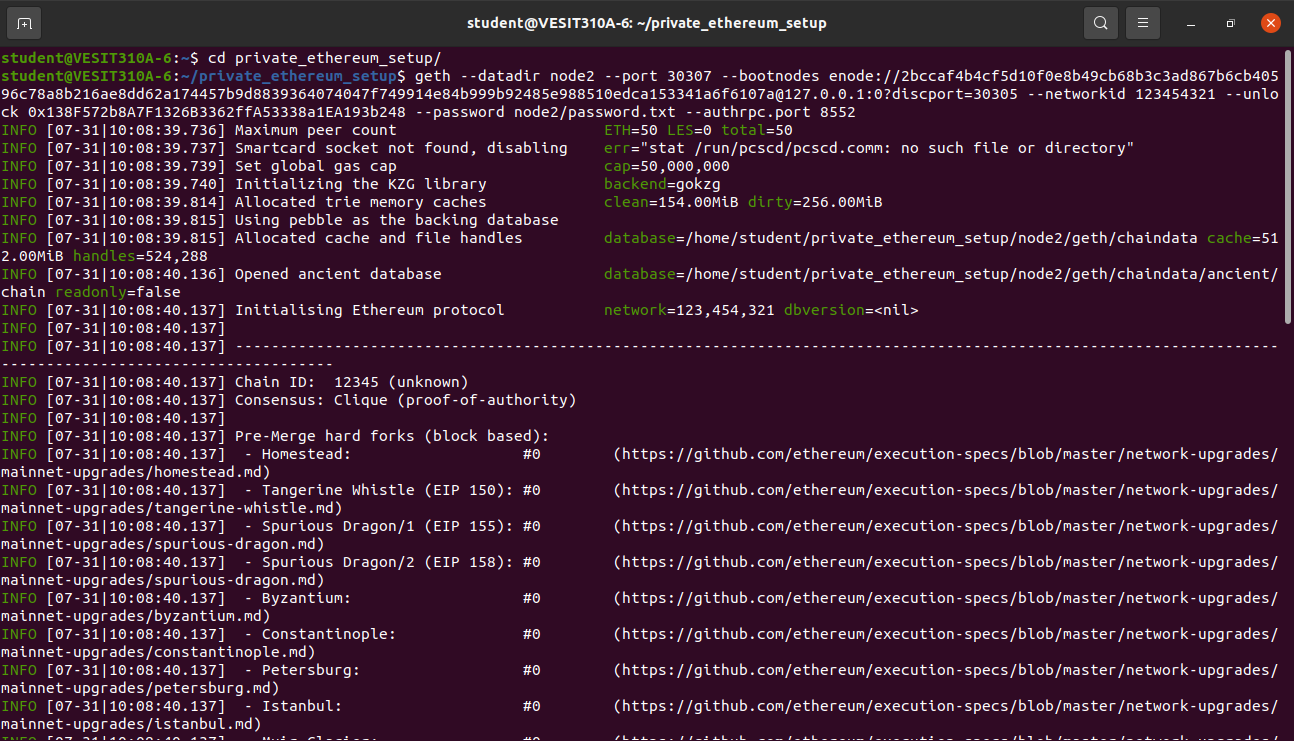


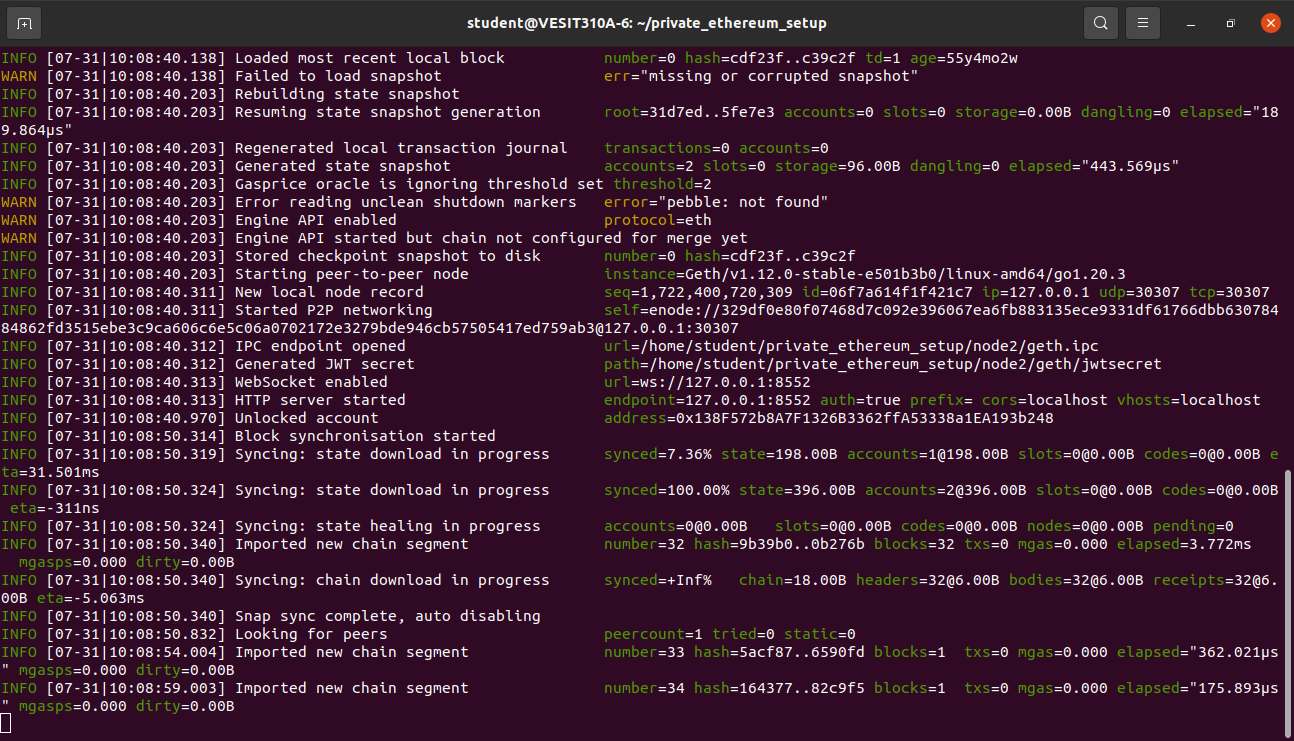


run the bootnode



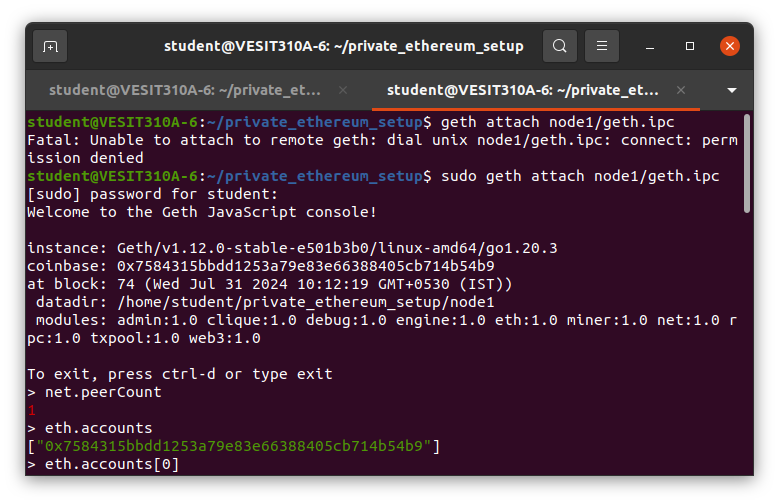
Node 1 starts mining



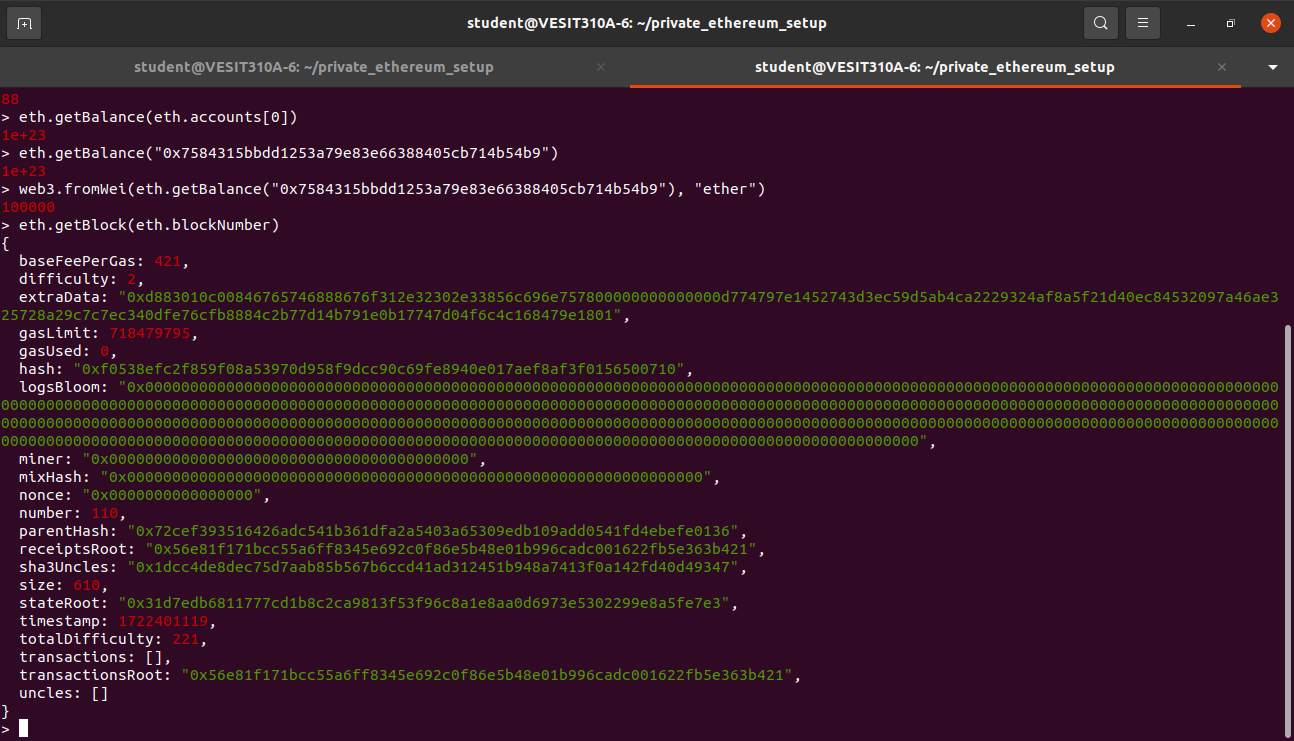


Node 2, receives the mined details on its terminal

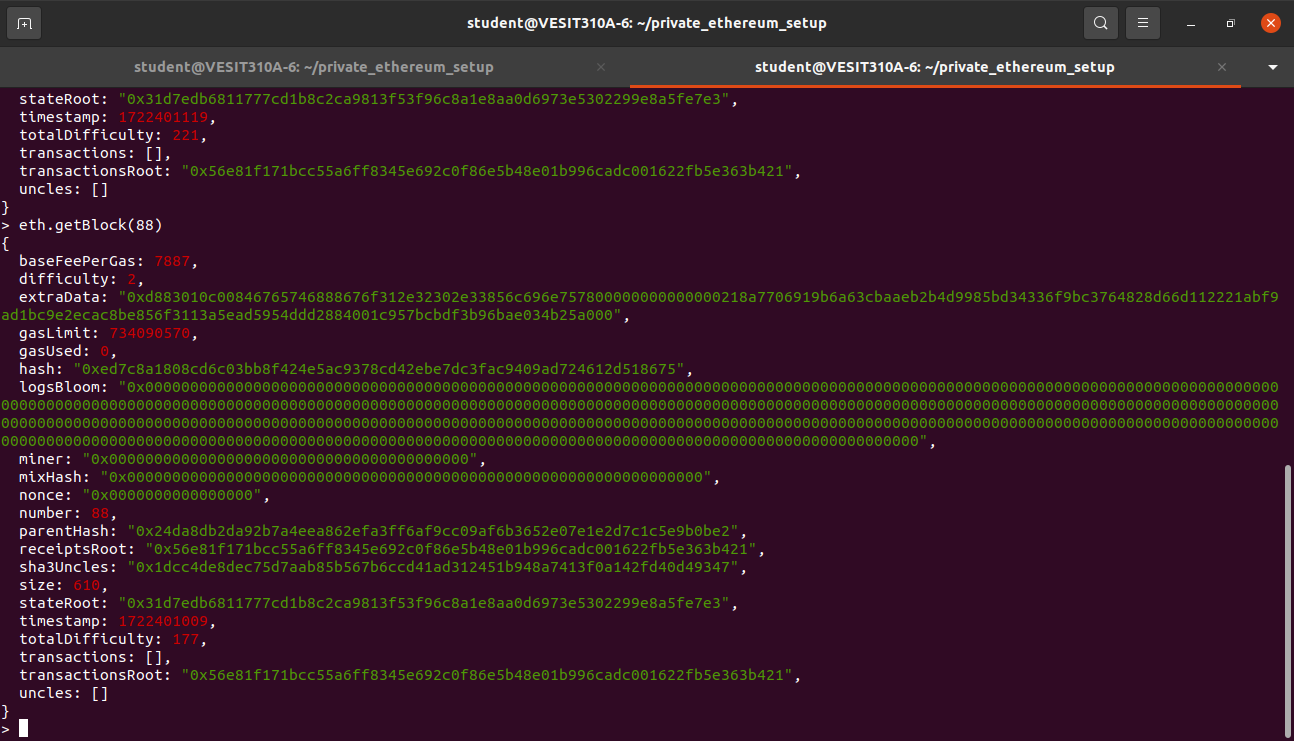
**Step - 5 : Exploring the network by attaching Javascript console to Node 1**

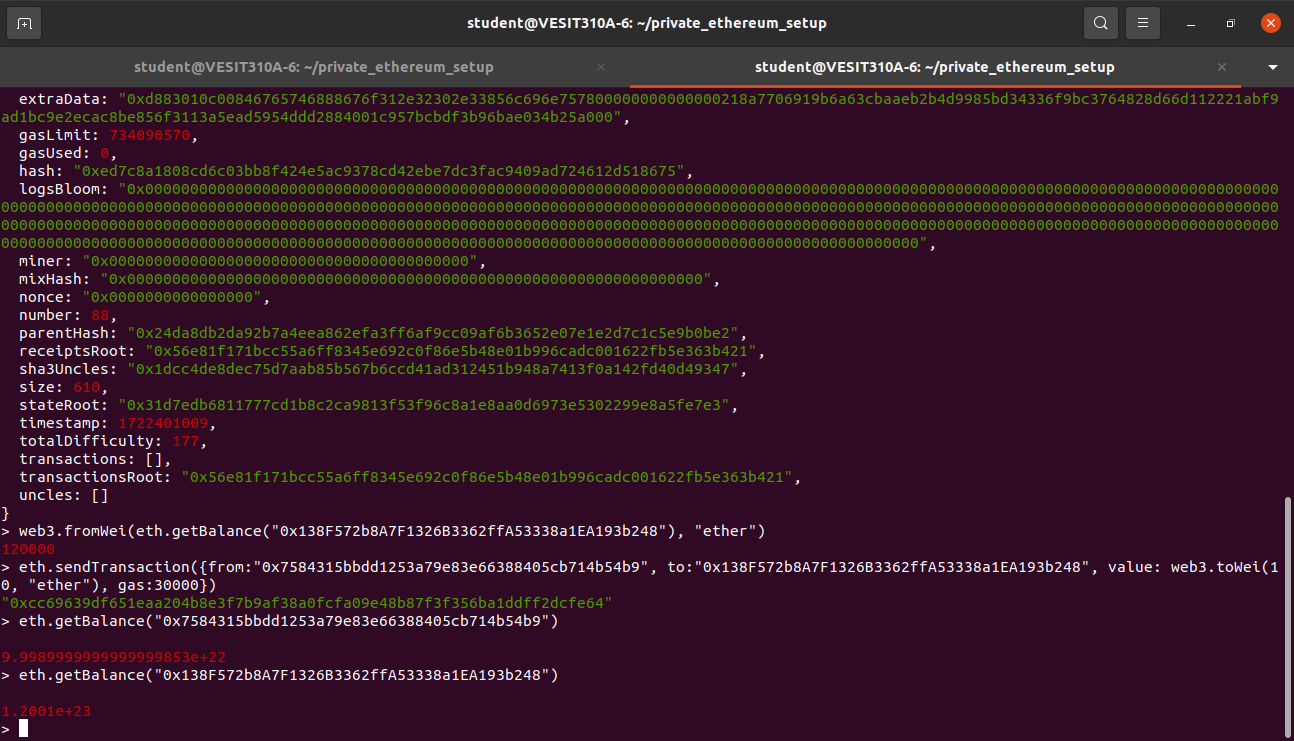


Attach the JavaScript Console with Node 1

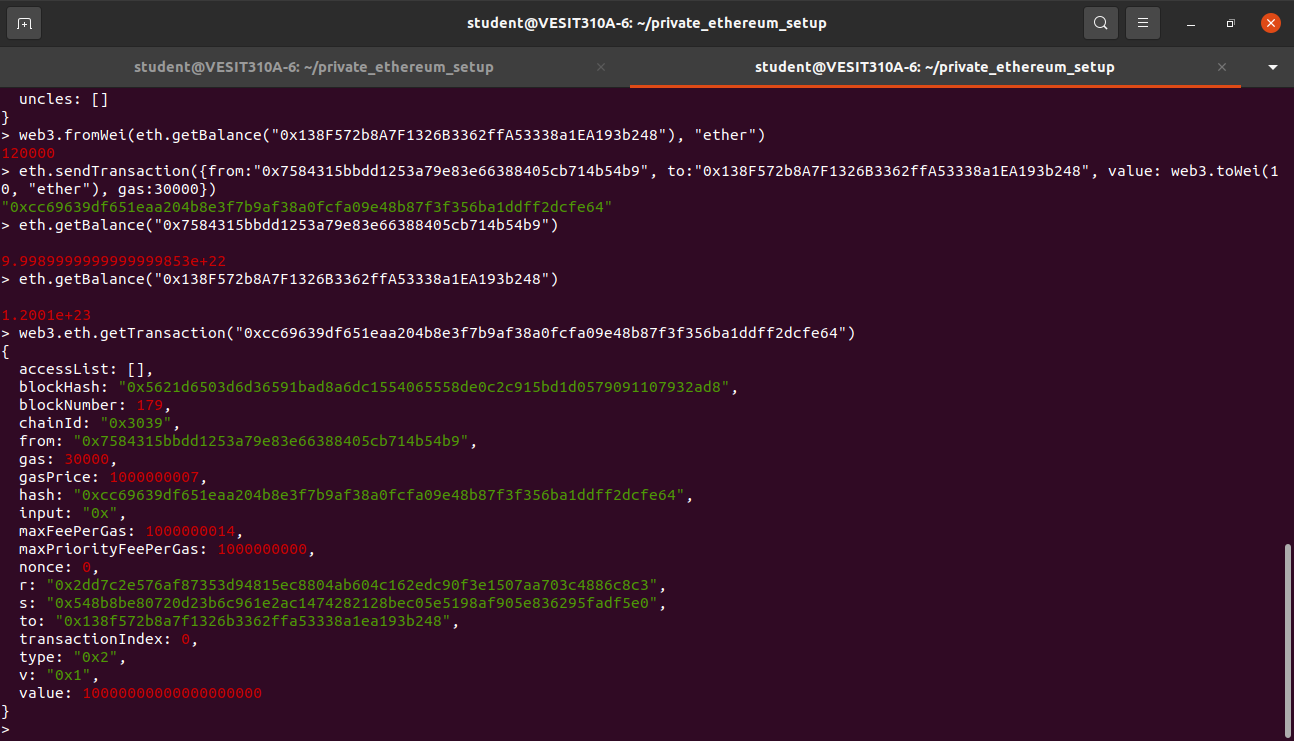


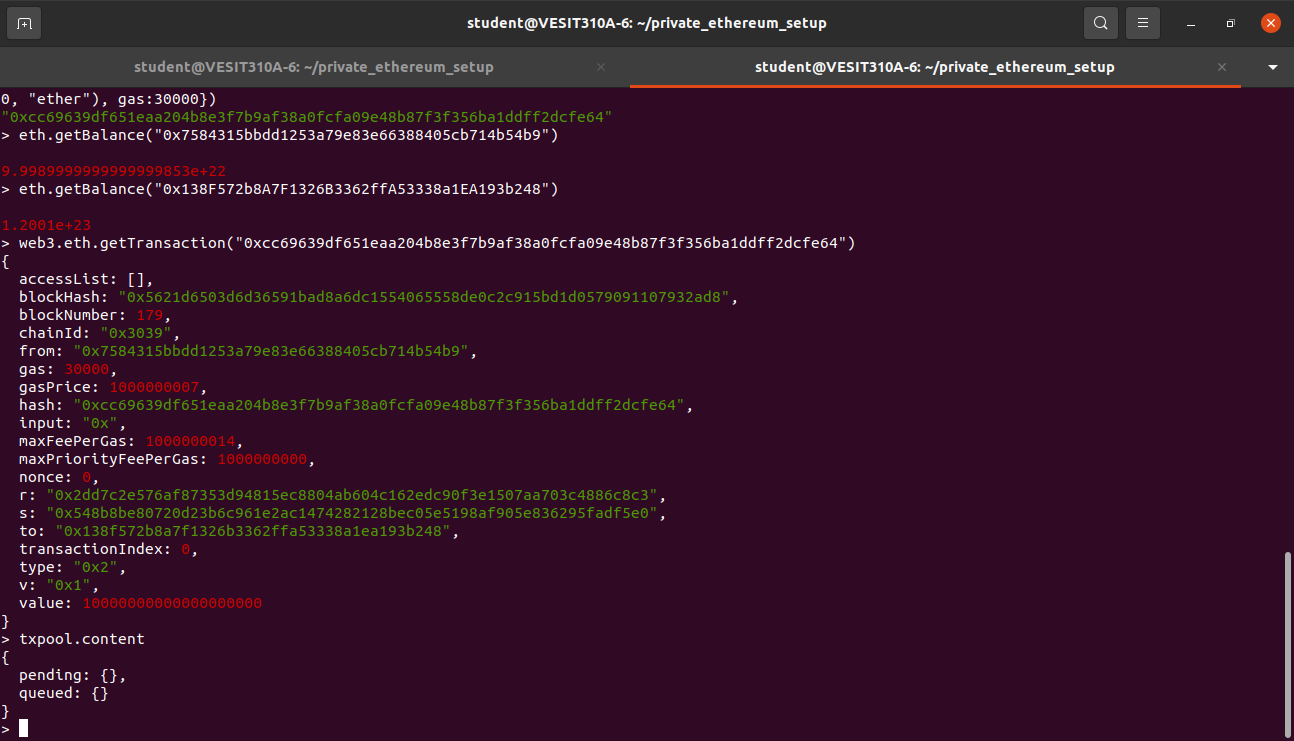
Fetch network status, To list the nodes in the network, To list the nodes in the network , To fetch the number of blocks mined :



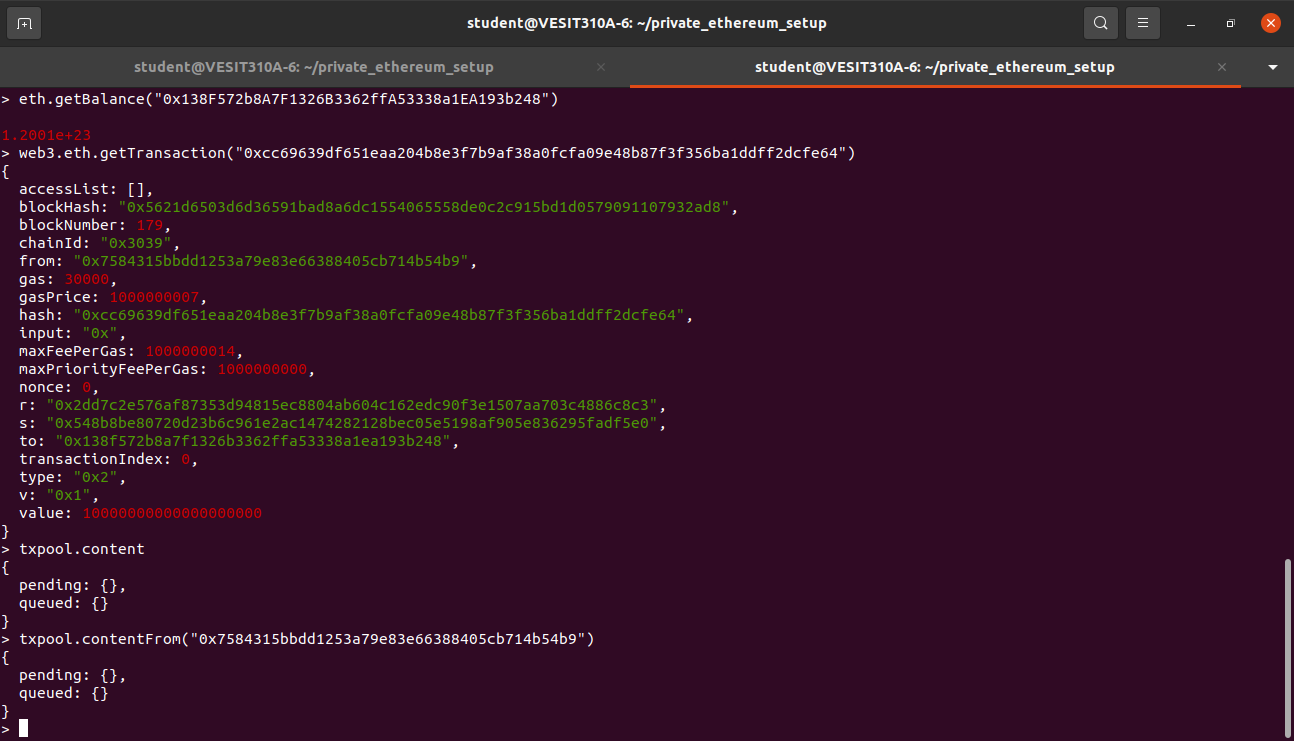


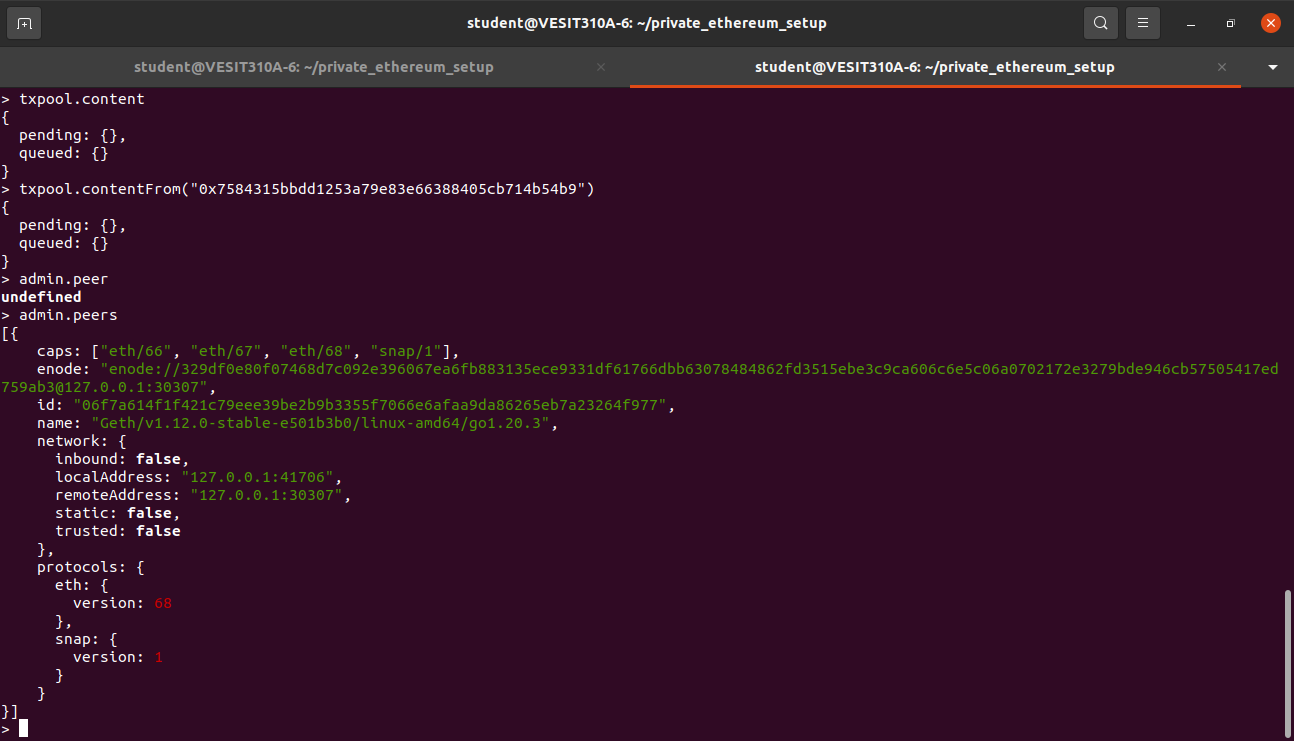
To check the balance of the accounts in wei





To check the balance of the accounts in ether, get the details of the block in which the transaction is added





To check the contents in the Mempool - Transaction Pool

**Conclusion:** Therefore, we studied and understood how to setup a private ethereum network using geth and implemented the same.

6. For configuring the bootnode

4. Establish a Peer-Peer Connection between the nodes along with the bootnode

1. On the first Terminal, Use boot.key to run the bootnode :

2. On the second Terminal, Run Node 1

3. On the third Terminal, Run Node 2

5. Step - 5 : Exploring the network by attaching Javascript console to Node 1

1. Fetch network status :

2. To list the nodes in the network :

3. To list the nodes in the network :

4. To fetch the number of blocks mined :

5. To check the balance of the accounts in wei :

6. To check the balance of the accounts in ether :

7. To fetch the details of the lastest mined block OR To fetch the details of a specific block :

8. To check the account balance of the peer machine, provide their Public Key :

9. Fetch the details of the peers in the network : admin.peers

10. Perform Transactions between peers in the network :

11. Check the balances of sender and receiver

12. To check the details of the transaction on Node 1 Terminal

13. To get the details of the block in which the transaction is added :

14. To check the contents in the Mempool - Transaction Pool :

15. To check the status of the Mempool - Transaction Pool :

16. To check the transactions initiated by a client, which are in the pool :

Node 1 public-private key details

Public address of the key: 0x7B754cEe93F4904577382320A23E2514534a3E07

Path of the secret key file: node1/keystore/UTC--2024-07-25T14-02-47.827823329Z--7b754cee93f4904577382320a23e2514534a3e07

Node 2 public-private key details

Public address of the key: 0x1DEA639897274B630A41b1BDe7b1bBF4A45a6d3a

Path of the secret key file: node2/keystore/UTC--2024-07-25T14-04-29.628803542Z--1dea639897274b630a41b1bde7b1bbf4a45a6d3a

{

"config": {

"chainId": 12345,

"homesteadBlock": 0,

"eip150Block": 0,

"eip155Block": 0,

"eip158Block": 0,

"byzantiumBlock": 0,

"constantinopleBlock": 0,

"petersburgBlock": 0,

"istanbulBlock": 0,

"muirGlacierBlock": 0,

"berlinBlock": 0,

"londonBlock": 0,

"arrowGlacierBlock": 0,

"grayGlacierBlock": 0,

"clique": {

"period": 5,

"epoch": 30000

}

},

"difficulty": "1",

"gasLimit": "800000000",

"extradata": "0x00000000000000000000000000000000000000000000000000000000000000000x7B754cEe93F4904577382320A23E2514534a3E070000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",

"alloc": {

"0x7B754cEe93F4904577382320A23E2514534a3E07": { "balance": "100000000000000000000000" },

"0x1DEA639897274B630A41b1BDe7b1bBF4A45a6d3a": { "balance": "120000000000000000000000" }

}

}