**CSED490U Blockchain & Cryptocurrency**

**Assignment 9**

****

**Submitted by- Sajan Maharjan**

**POVIS id-** [**thesajan@postech.ac.kr**](mailto:thesajan@postech.ac.kr)

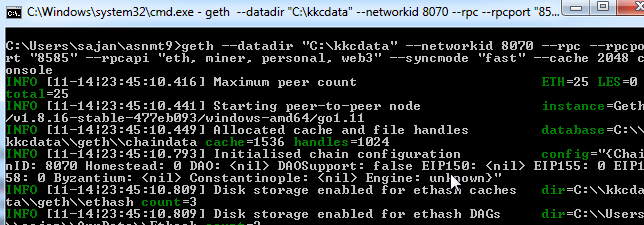
**Registration Number- 20182095**

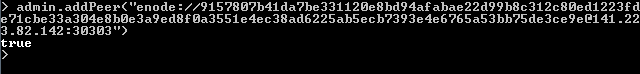
Assignment 9: Simple Ethereum Wallet

**<Step 0> Connect to private network**

> The blockchain node operating in my computer was connected to the given private network using geth command along with the same genesis file, network id/ chain id and peer was added using admin.addPeers() command.

|  |
| --- |
| *geth --datadir “C:\kkcdata” –networkid 8070 --rpc --rpcport “8585” --rpcapi “eth, miner, personal, web3” console* |



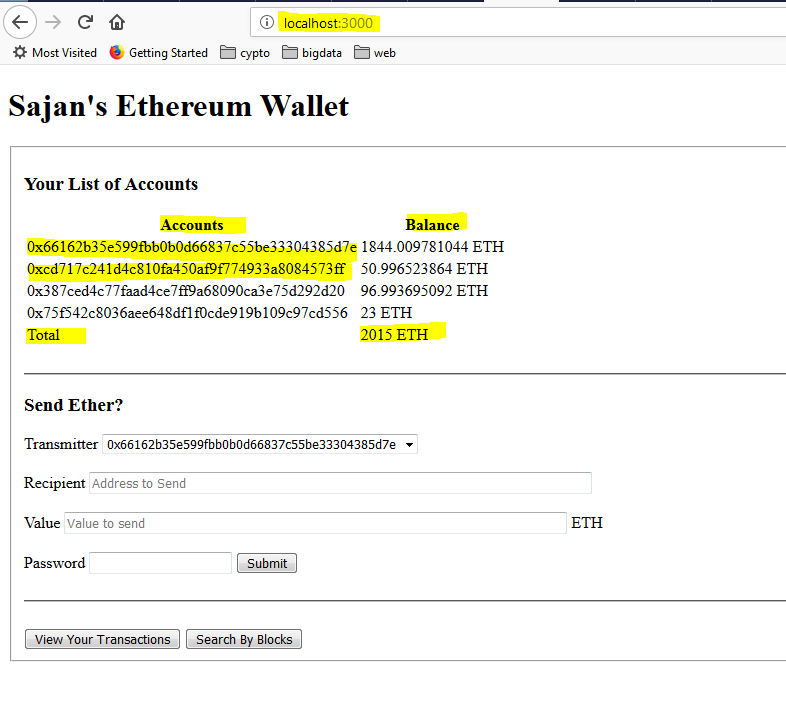


**<Step 1> Create a simple ethereum wallet**

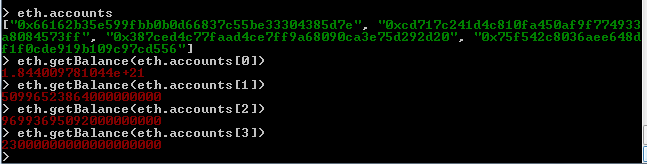
1) Show the code of your simple ethereum wallet and briefly describe it

|  |
| --- |
| //server.js file  //import necessary libraries  const express = require('express');  const app = express();  const fs = require('fs');  var bodyParser = require('body-parser');  var Web3 = require('web3');  var w3 = new Web3();  //logger for writing transaction hash to a file  var logger = fs.createWriteStream('txnHashList.txt', {  flags: 'a' // 'a' means appending (old data will be preserved)  })  //array containing txn hash, which is read from a file and used to listing transactions of wallet  var txnList = [];  //setting web3 object  w3.setProvider(new Web3.providers.HttpProvider('http://localhost:8585'));  //setting for body-parser and ejs engine and adding public folder to our path  app.use(bodyParser.urlencoded({ extended: true }));  app.use(express.static('public'));  app.set('view engine', 'ejs')  //handling GET requests  app.get('/', function (req, res) {  var accounts = w3.personal.listAccounts;  console.log(accounts)  var balArray = [];  for(var i=0; i < accounts.length; i++) {  balArray.push(w3.fromWei(w3.eth.getBalance(accounts[i]),'ether'));  }  console.log(balArray.toString());  var addressBalanceMap = {};  accounts.forEach((key, i) => addressBalanceMap[key] = balArray[i].toString());  console.log(addressBalanceMap);  res.render('index', { accounts: accounts, balArray: balArray });  })  //handling POST requests  app.post('/', function (req, res) {  var accounts = w3.personal.listAccounts;  console.log(accounts)  var balArray = [];  for(var i=0; i < accounts.length; i++) {  balArray.push(w3.fromWei(w3.eth.getBalance(accounts[i]),'ether'));  }  console.log(balArray.toString());  var addressBalanceMap = {};  accounts.forEach((key, i) => addressBalanceMap[key] = balArray[i].toString());  console.log(addressBalanceMap);  res.render('index', { accounts: accounts, balArray: balArray });  })  //custom defined function to unlock account in the wallet with password  function unlockAccountIfNeeded(account, password) {  console.log("Account " + account + " is locked. Unlocking ...")  w3.personal.unlockAccount(account, password, 300);  }  //function used for sending ether to another address  function sendAmt(sender, receiver, amt) {  var txHash = w3.eth.sendTransaction({  from: sender,  to: receiver,  value: w3.toWei(amt,'ether')  });  console.log("Current Transaction Hash: " + txHash);  logger.write(txHash);  logger.write('\n');  }  //function call made to unlockAccount and senderTransaction after getting form data  app.post('/processRequest', function(req, res) {  //print the POST variables in console  var obj = req.body.transmitter + " " + req.body.recipient + " " + req.body.sendAmount + " " + req.body.pwd;  console.log(obj.toString());  //store POST variable for send operation  var sender = req.body.transmitter;  var receiver = req.body.recipient;  var amt = parseFloat(req.body.sendAmount);  var pwd = req.body.pwd;  unlockAccountIfNeeded(sender, pwd, function(err, res) {  if(err) {  console.log("Password Error! Try Again");  res.send("Password Error! Try Again");  }  });  sendAmt(sender, receiver, amt);  res.send("Transaction Sent. Wait for block to be mined");  })  //handler for ‘View Transactions’ button  app.post('/viewTxn', function(req, res) {  txnList = fs.readFileSync('txnHashList.txt').toString().split("\n");  console.log(txnList);  res.render('viewtxn', { txnList : txnList });  })  //handler for ‘View Transaction Details’ button  app.post('/viewTxnDetails', function(req, res) {  var selectedTxHash = req.body.selectedTxn  var txnDetails = w3.eth.getTransaction(selectedTxHash);  res.send(txnDetails);  })  //handler for ‘Search blocks’ button  app.post('/searchBlock', function(req, res) {  res.render('block')  })  //handling GET with blockNumber requests  app.get('/block', callBlk);  function callBlk(req, res){  let number = req.query.blockNumber;  w3.eth.getBlock(number, function(err, Blk){  console.log('Block: '+ number + ' is sent ...');  console.log(Blk);  res.send(Blk);  });  }  //running application at PORT 3000  app.listen(3000, function () {  console.log('Example app listening on port 3000!')  }) |

**2) Show your simple ethereum wallet’s web page**



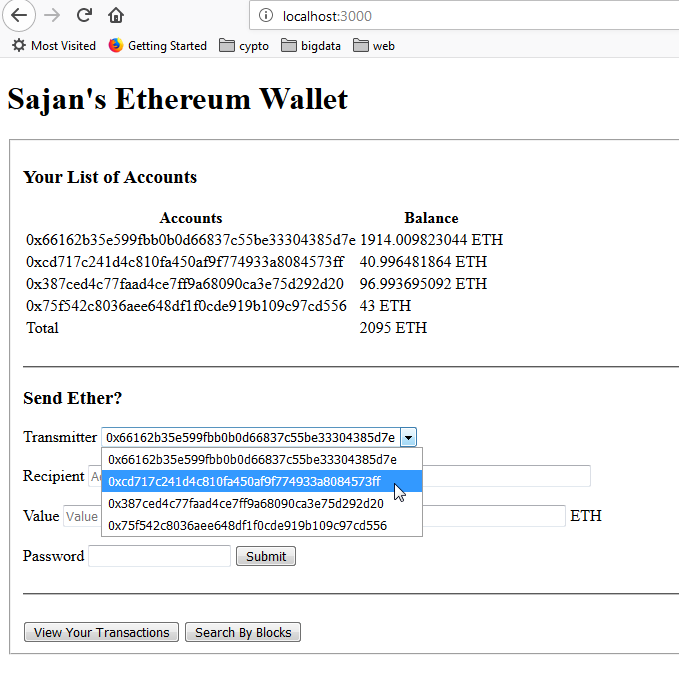
This detail matches with the response from geth command line interface and etherum wallet as-

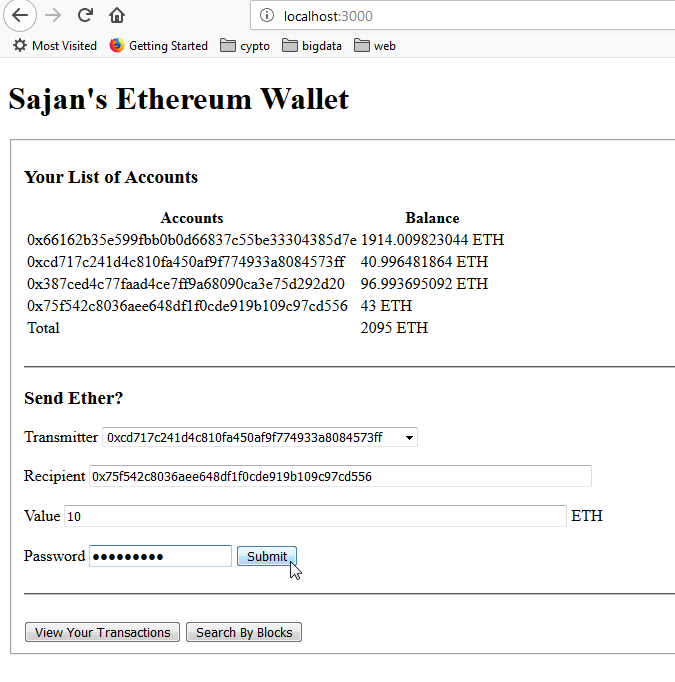




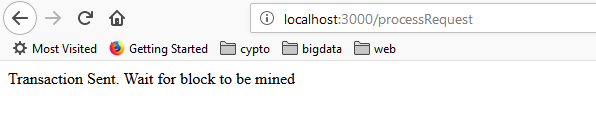
**3) Transfer Ether between accounts**

> Let us transfer 10 units of ether from the address ***0xcd717c241d4c810fa450af9f774933a8084573ff*** (currently holding 40.99ETH) to the address ***0x75f542c8036aee648df1f0cde919b109c97cd556*** (currently holding 43ETH) using our ethereum wallet web application. ***The units of ether shown above differs from here due to some test mining and transfer being done before taking snapshots.***





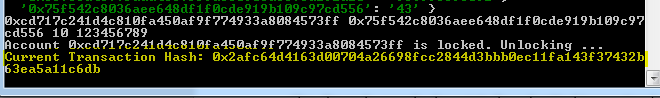
On successfully entering password, the transaction is submitted to the blockchain and waits for miners to commit, which is shown by the wallet application as-



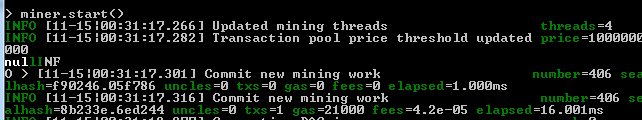
This can be confirmed from the geth terminal which shows new submitted transaction as-

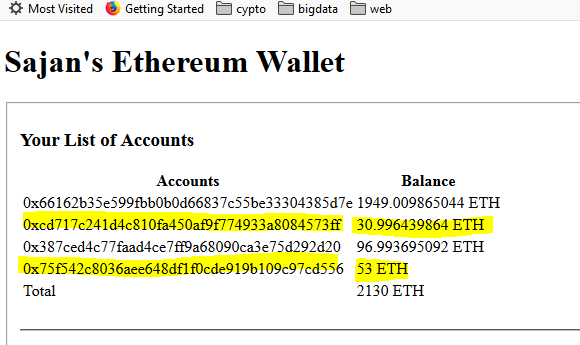


Also, the node console also prints the submitted transaction hash as-

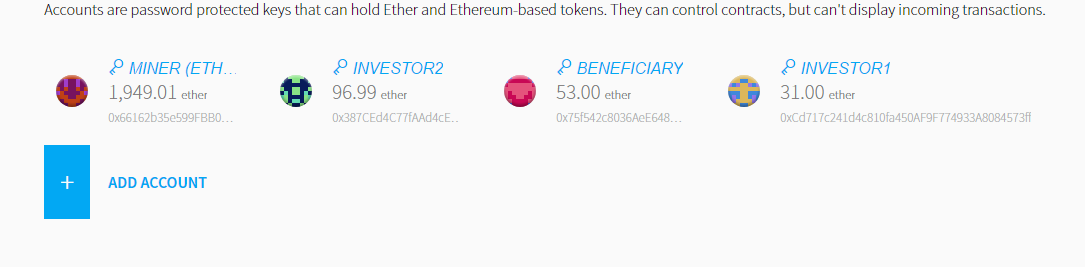


Now, we start mining and confirm the transfer of ether as-



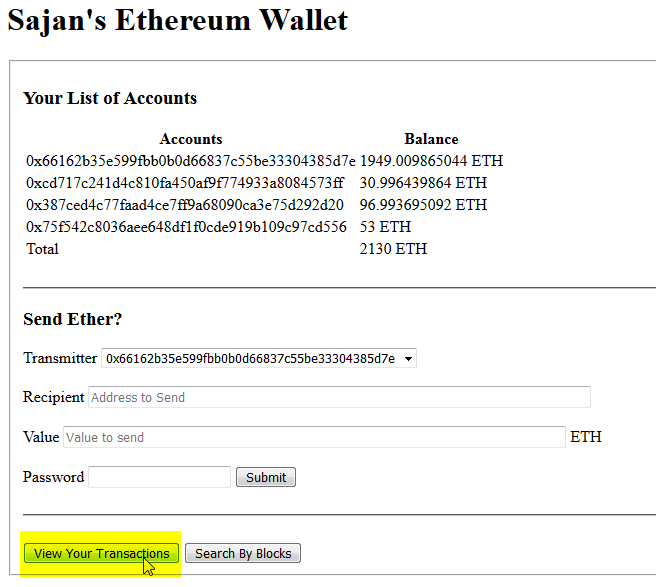


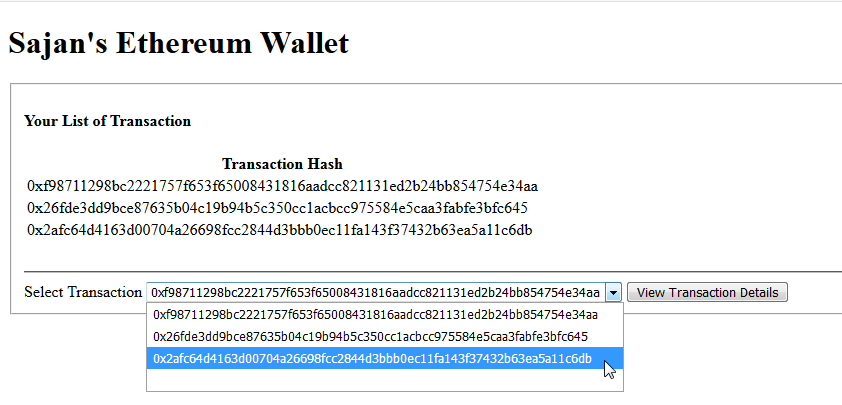
10 units of ether was deducted from ***0xcd717c241d4c810fa450af9f774933a8084573ff*** (now holding 30.99ETH) account while corresponding 10 units were added to ***0x75f542c8036aee648df1f0cde919b109c97cd556*** (now holding 53ETH) through the above transaction.



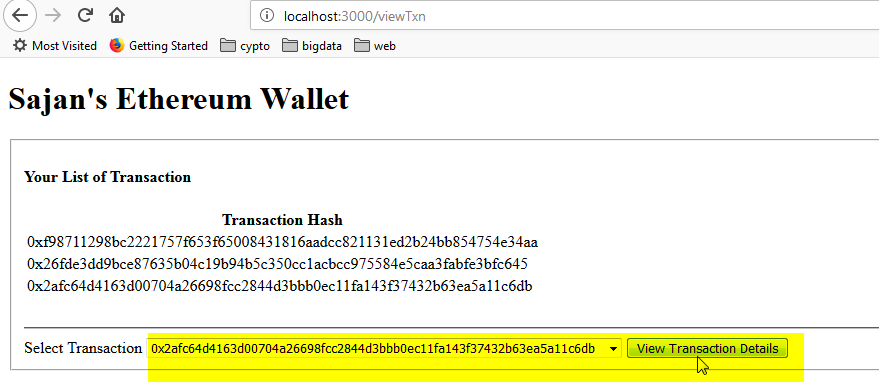
4) Check the transaction that send 10 units of ether

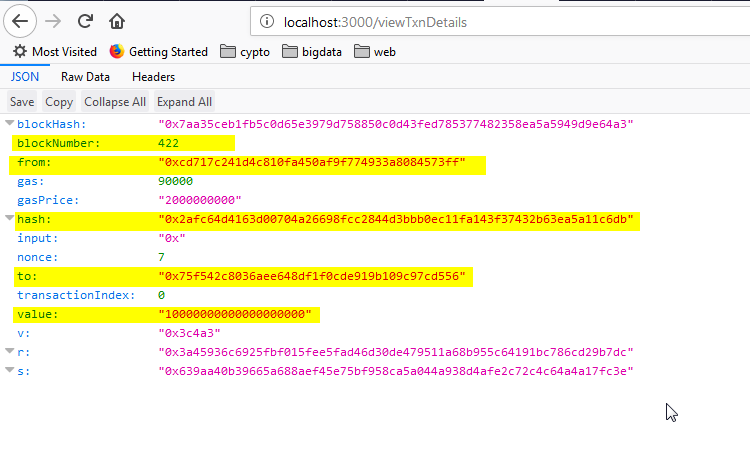
> Our wallet application GUI facilitates checking all the transaction details corresponding to the wallet as-





The transaction hash of the last transaction was- ***0x2afc64d4163d00704a26698fcc2844d3bbb0ec11fa143f37432b63ea5a11c6db***. We can view the details of this transactions by clicking on View Transaction Details button as-

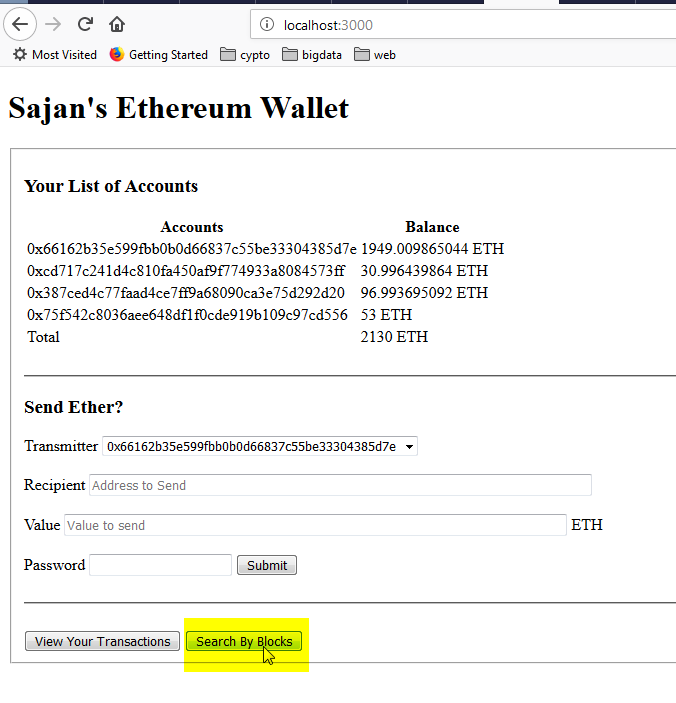


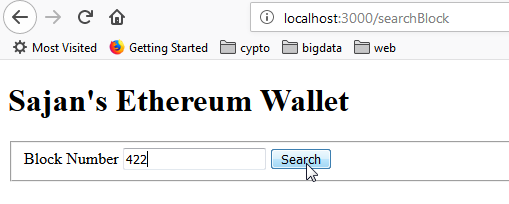


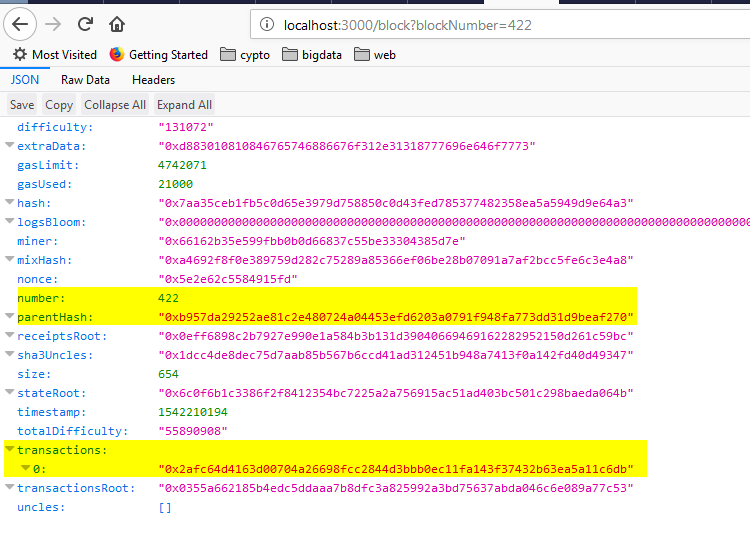
We can confirm the details of the transactions such as from, to, value, blockNumber containing the transaction and more.

**5) Check a block that includes the transaction**

> We can search for a particular block in our ethereum wallet application by using block number. This is shown as-







This will return the block details such as- transaction merkle root, list of transactions (our previous transaction is also included as shown above), parent block hash and more.