mtech-bitcoin-elective

lecture 1

course code: 20cy712
text: mastering blockchain by imran basheer yr 2017
⇒ understanding blockchain basics

syllabus

1) introduction
• who created blockchain, who did what etc
2) specific concepts
3) bitcoin
1- architecture
2- crytpocurrency
3- challenges
4- transfer method
5- mining
5) ethereum(30 to 40 %, hands on lab etc)
6) introduction to hyperledger and some projects hands on
7) consesus
1- what is it
2- types
3- challenges
8) different use cases of blockchain
1- present and future of blockchain use cases
9) challenges
1- present and future challenges
2- realtime
3- compliance
4- legal

10) supporting tools
1- improving costs, ux
2- helping to solve challenges easier

introduction

• bitcoin and blockchain are not the same
◇ bitcoin is the first successful commercial application of blockchain
• blockchain is defined in two ways
◇ blockchain ledger (datastructure)
■ blockchain is consisting of many blocks
- each block is having transactions
■ every block is linked to each other via crypto hash
◇ technology
what
■ decentralized computation(workers), distributed ledger (data which is worked on by workers available to all workers) platform
- decision is made by some handful of computers
why
■ to immutably store transactions in a verifiable manner
how the why
■ through a rational decision making process (consensus) by multiple parties in open and public system in a efficient manner
• apply blockchain in multiple parties usecase, not for less number of people.

• characteristics of blockchain
◇ transparent

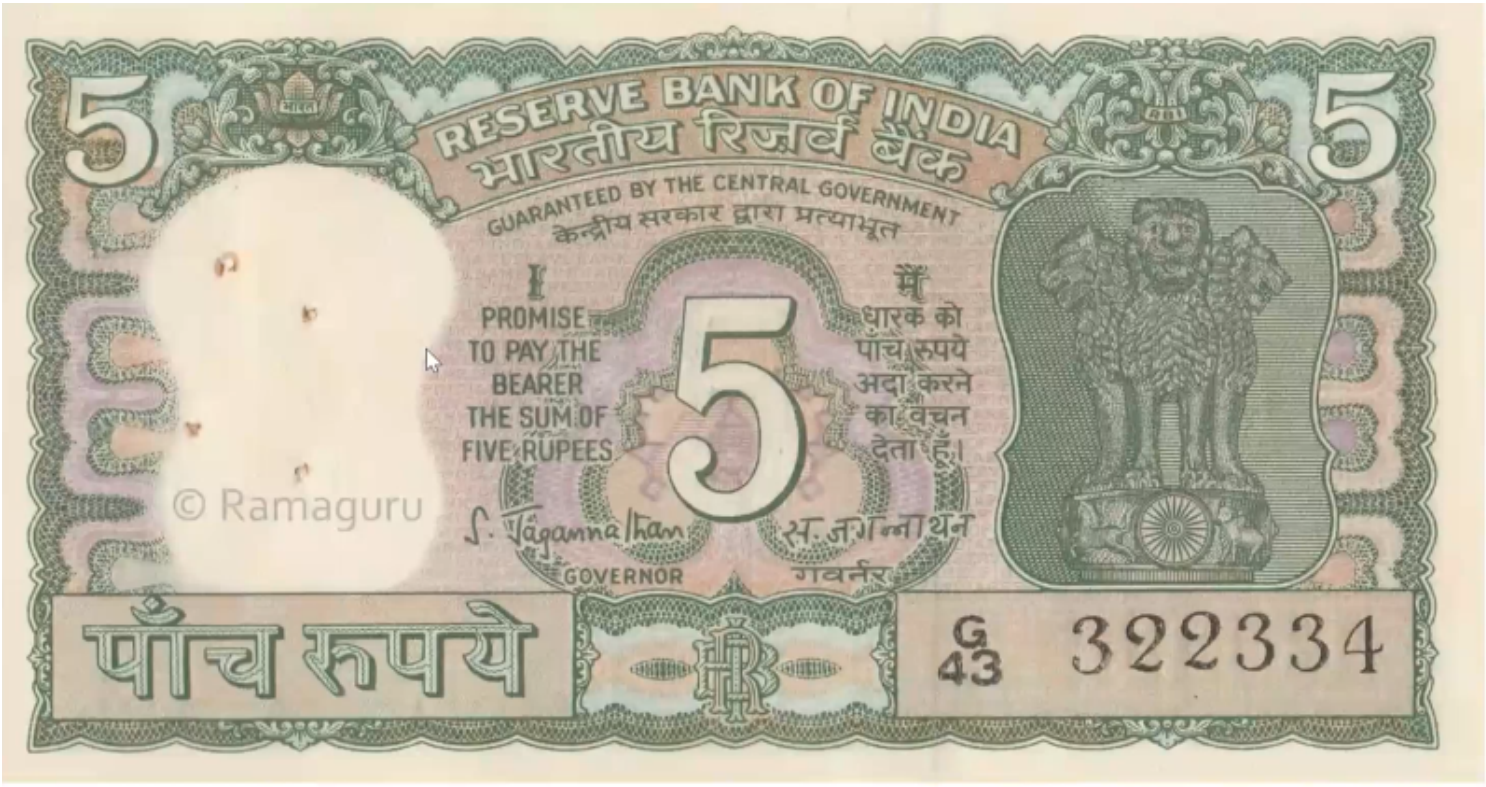
◇ distributed
◇ decentralized
◇ immutable
◇ consensus

lecture 2

why blockchain?

• there is no promissory note
• There is no reserve bank of india, there is only government of india
• Signed by finance secretary under ministry of finance
  ◇ only note which is having no promissory note and signed by ministry of finance
• there is a coin image
• when ever 1 rupee note is issued, it is issued like this,

substitute for one rupee coin
  ◇ hence the coin image
  ◇ coins wont have promissory note
  ◇ rupee note issued to reduce metal consumption required to mint the actual coin



• “I promise to pay the bearer the sum of five rupees” → promissory note by Governor of RBI
  ◇ can only be given by RBI and only be guaranteed by central government
  ◇ basically saying RBI is viable to pay the owner of the note, gold/services worth Rs 5
• The paper having five rupees worth is guaranteed by central government
• The paper is issued by reserve bank of india

commodity money
• value is there because of material has intrinsic value
  ◇ ie laptop “worth” 60,000

◇ gold, bronze etc coins

fiat money
• worthless stuff guranteed by certain organisation to have some money
◇ it is currency not a note
◇ material has extrinsic value
◇ sodexo pass
◇ amazon gift voucher is valid at amazon not flipkart
◇ it is at the end of day worthless paper
◇ value can be revoked at any stage

transaction:
• product or service getting transferred from one entity to other entity
• transactions need not be always financial
• ie exchanging books, giving one pen away for free
• single ended transaction
◇ giving book to person
• double ended transaction
◇ selling book to person
◇ in return person gives money due to first transaction

problem with traditional transaction?

• for accountability and auditing and future settlement you need ledger system
• account keeping for traditional transaction is very difficult
◇ keeping unpaid money details in a physical ledger is difficult
■ one customer needs one page or pages(physical constraint)

    ■ needs to constantly update manually details could be mistakenly made or forged
  ◇ scalability is very difficult

transaction fee:

• utilizing a resource
  ◇ for transferring from account A to account B, if using google pay
    ■ google needs to ensure account A exists, has the money and account B exists
    ■ account details of A and B are in different banks, google needs to check with them individually before transaction is initiated
    ■ x amount of value from account A decreases and x amount of value from account B increases
    ■ this amounts to computational resource and storage resource being used which we pay with service fee or personal data being extorted etc :P
• profit
  ◇ google needs to make profit so something extra they will charge
  ◇ problem is unregulated banking services might charge really high profit charges
    ■ quarterly charge
    ■ maintenance charge

social justice

• present banking system does not guarantee financial inclusivity
• KYC norms becoming too unneeded
• 2014 -> Jan Dhan Yojana
  ◇ zero balance account can be started

◇ but requires aadhar etc which a large section of poor dont have

bitcoin blockchain

• removes banks as financial intermediaries
• charge less transaction fees
◇ records are distributed to all people to reduce the transaction fees
◇ decentralize the network
◇ apply public consensus
■ out of n nodes some nodes will perform the computation to see if transaction is valid
■ split the transaction among nodes validating transaction
• people have more financial freedom
◇ money is in control of the people not the government
• problem is if you are using real identity to participate you are not having privacy
◇ entire node is able to see transaction
◇ so annon the network

ram -> 50rs -> sam
sameul -> 100rs -> ram
jack -> 900 -> sam

this became

erer -> 50rs -> trtr
adad -> 100rs -> erer
sdsd -> 900rs -> trtr

◇ overtime if you use single anon identity patterns can

be established
    ■ so one anon user can have multiple accounts (as much as they want)
    ■ multiple accounts meaning multiple identities
    ■ bitcoin do not care about identity of user only validity of transaction

Value of bitcoin

• the more bitcoins you have in general the more expensive it is
  ◇ this is because it takes more time and effort to mine the same number of bitcoins today than 5 years ago
  ◇ bitcoins have a upper limit (21 million) this is will reach in 2140
    ■ but 18 million bitcoins are mined by 2021
      - ie process is more complex every year
      - ie finite resource to mine
  ◇ reward to create a new bitcoin block gets halved every four years
  ◇ more people will use the bitcoin blockchain but less bitcoins per person
  ◇ this added complexity and rarity makes bitcoin expensive

lecture 3

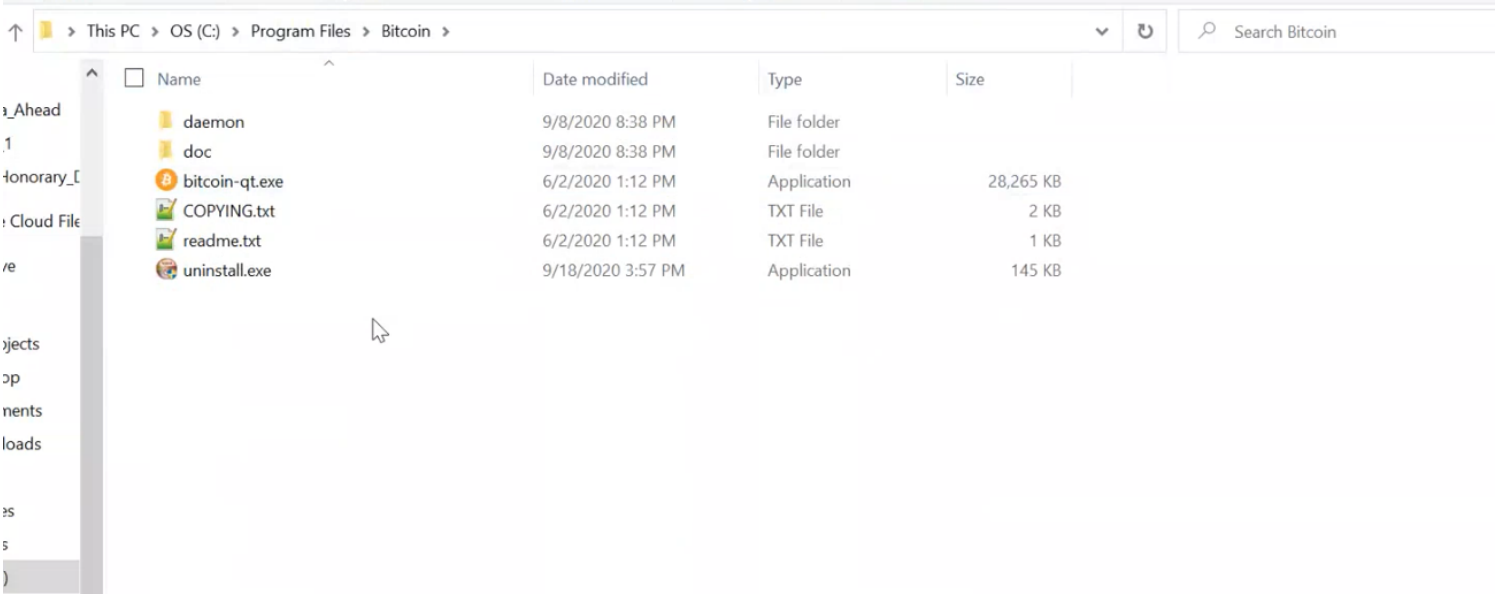
wallet

• container for money, documents etc
  ◇ physical wallet which can be carried with your self
  ◇ googlepay, paytm are digital wallet
    ■ whatever money u transfer from bank to digital wallet is there in ur digital wallet
    ■ if money is not there in wallet

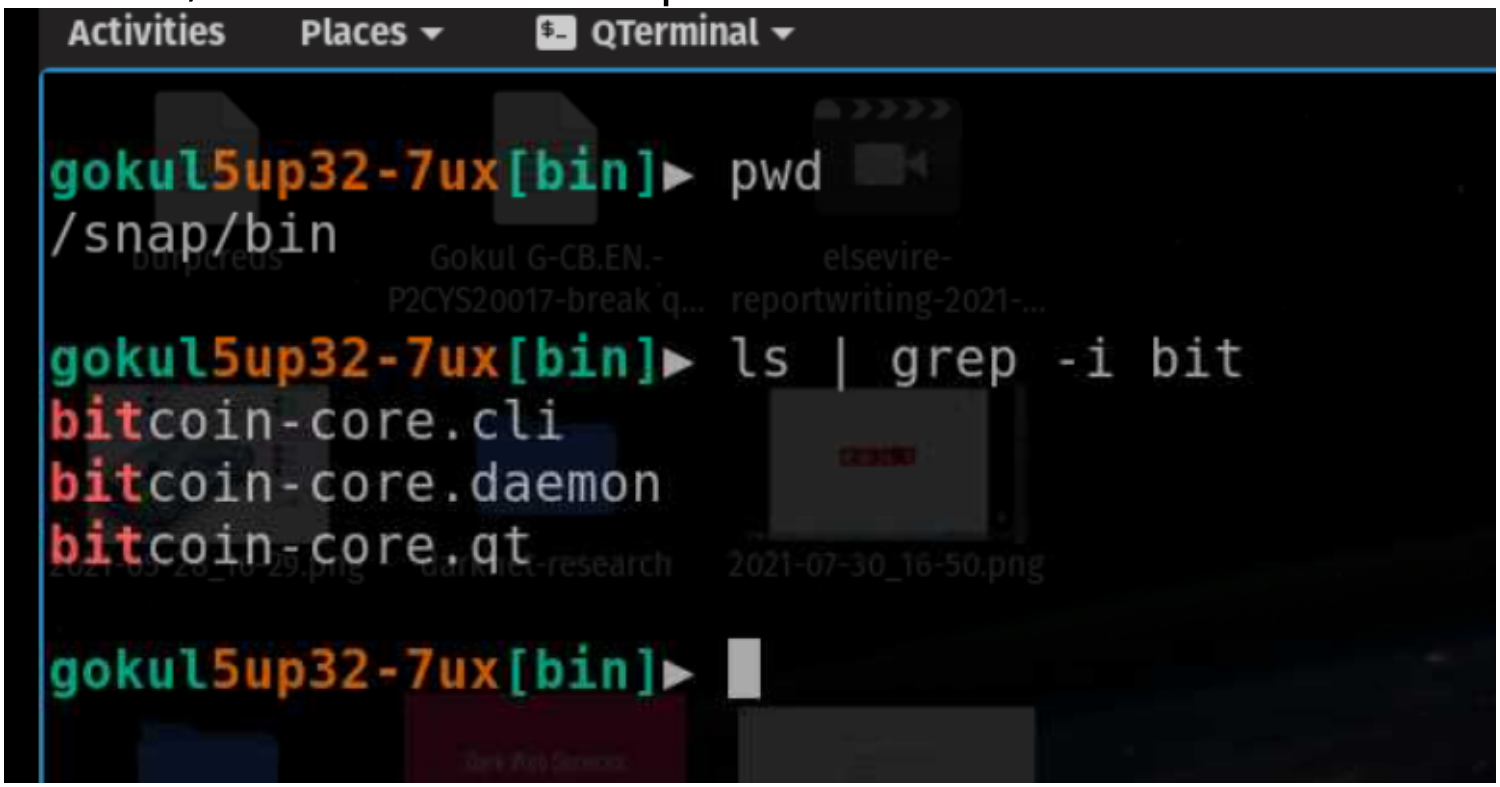
- then also we can do transactions
    - need not necessarily hold money
    - it is a place where we can have multiple accounts that we can use for digital transaction
    - it is a digital passbook, acting as payment gateway
    - mobile application or web mode
- crypto wallet
    - digital wallet for crypto assets
    - middle man which acts as passbook
    - multiple accounts possible for same or different blockchain
    - does not store the money as bitcoin etc are virtual
        - only have the private key and corresponding address used for signing the transaction
        - money model is UTXO → unspent transaction output (unspent bitcoin - value spend in tranasctions) and not a account balance model(data obtained from last transaction) like ethereum and this is a protocol token hence ethereum has crytpo called ether

bitcoin wallet

in windows:



in linux, installed with snap:



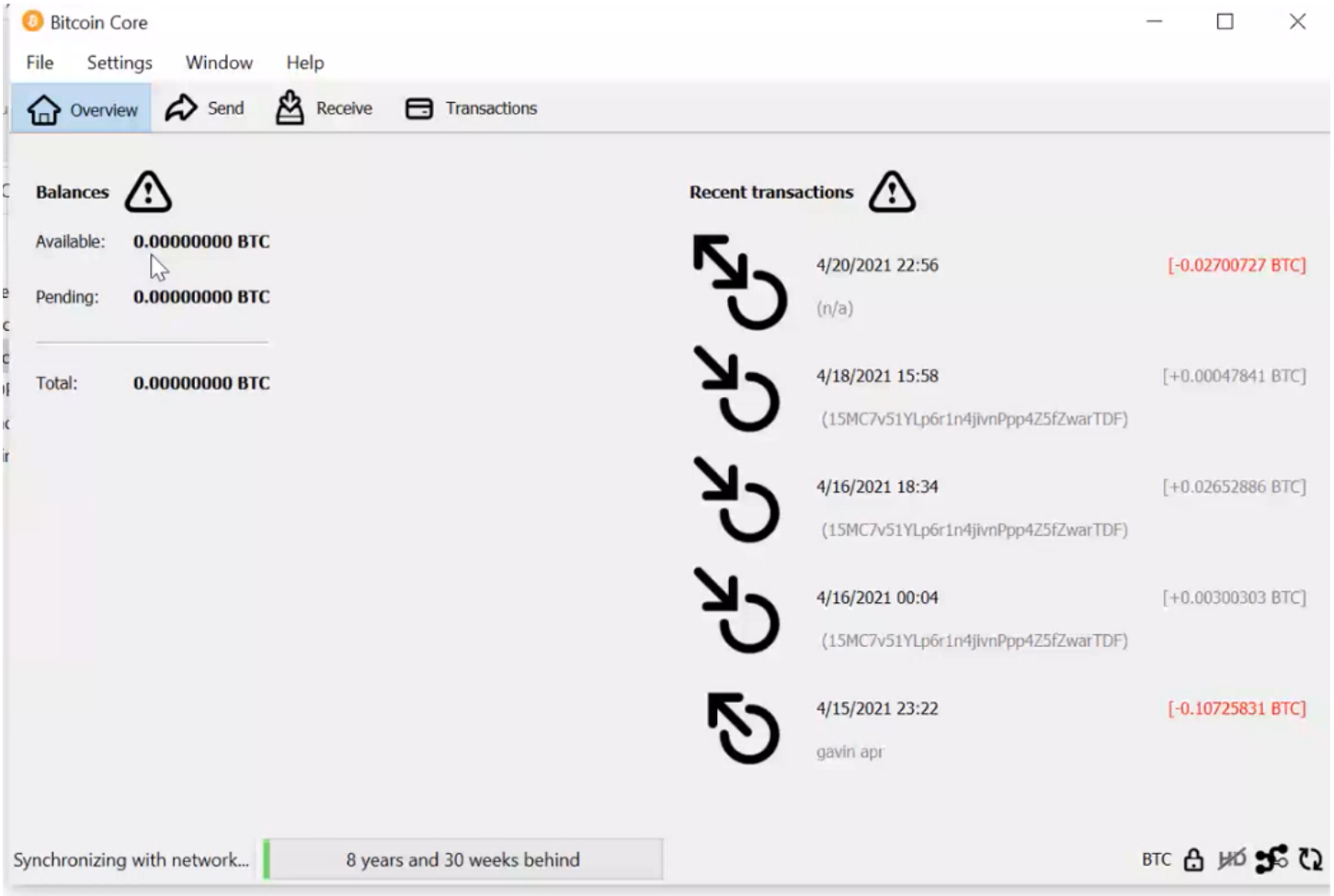
bitcoin-qt wallet is not only a wallet but also a bitcoin client

• meaning it is going to download the entire the bitcoin blockchain to the local node

◇ ie the ledger is going to be downloaded to your device

■ Client will download it because this is needed to participate in the blockchain

◇ if you run the bitcoin-qt, your computer is part of the blockchain ie a node



- we can have testing mode

◇ main net → everything is real, money and transaction is real

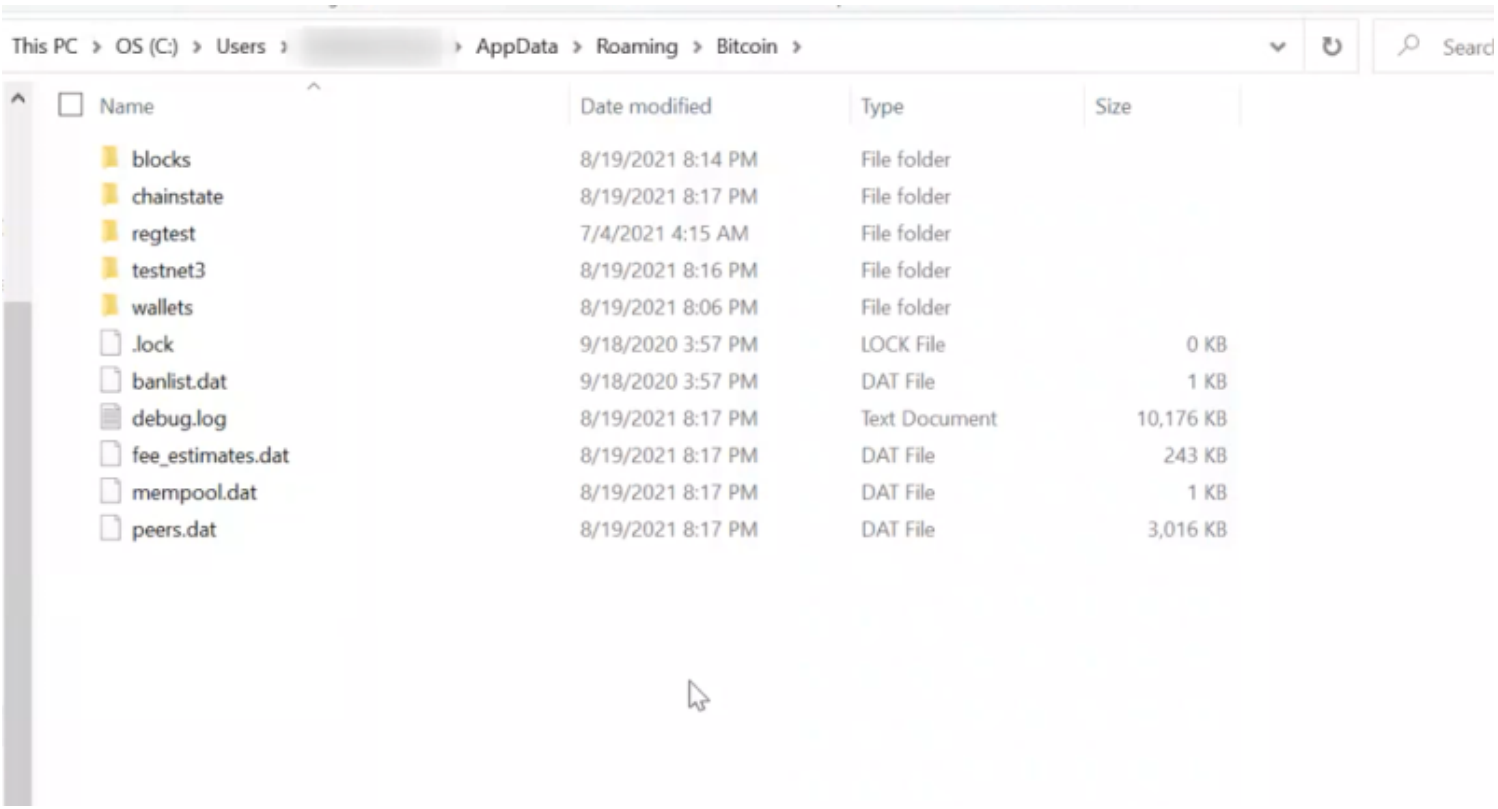
◇ test net → technology is real, money and transaction is dummy

■ we are gonna use only test network

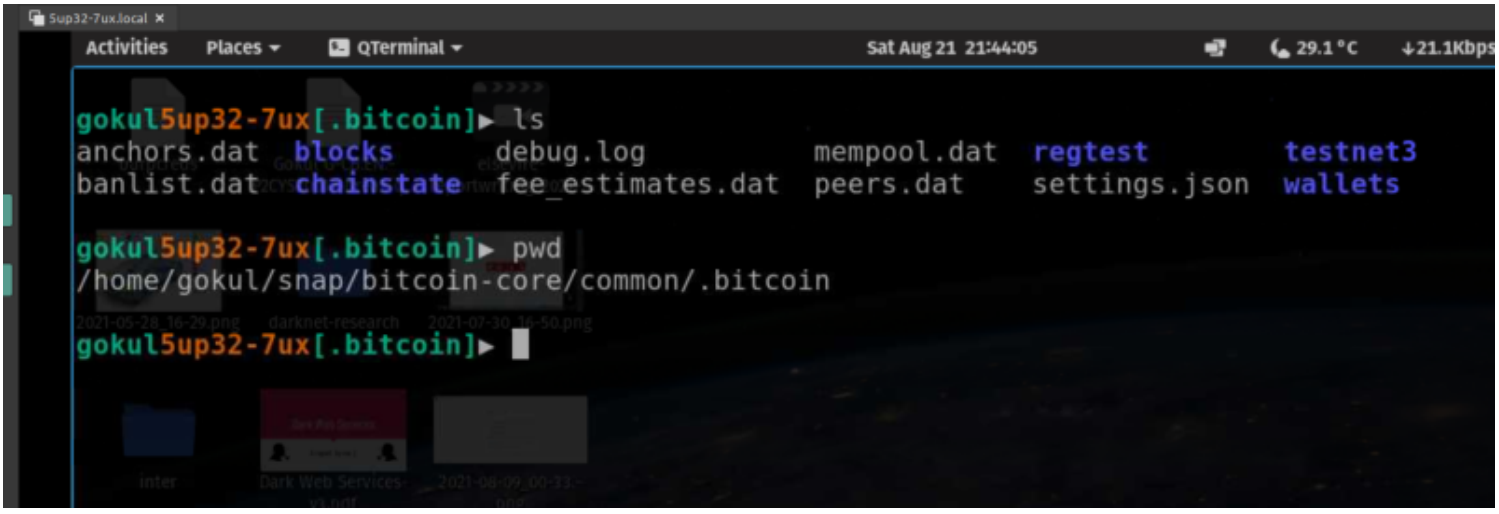
■ only less no of nodes when compared to main net

■ bitcoin.qt --testnet

in windows:



in linux:



blocks -> details about block stored as dat folder

- block details are stored in blkxxx.dat
  - ◇ one dat file has info about multiple blocks
  - ◇ max size is 128MB

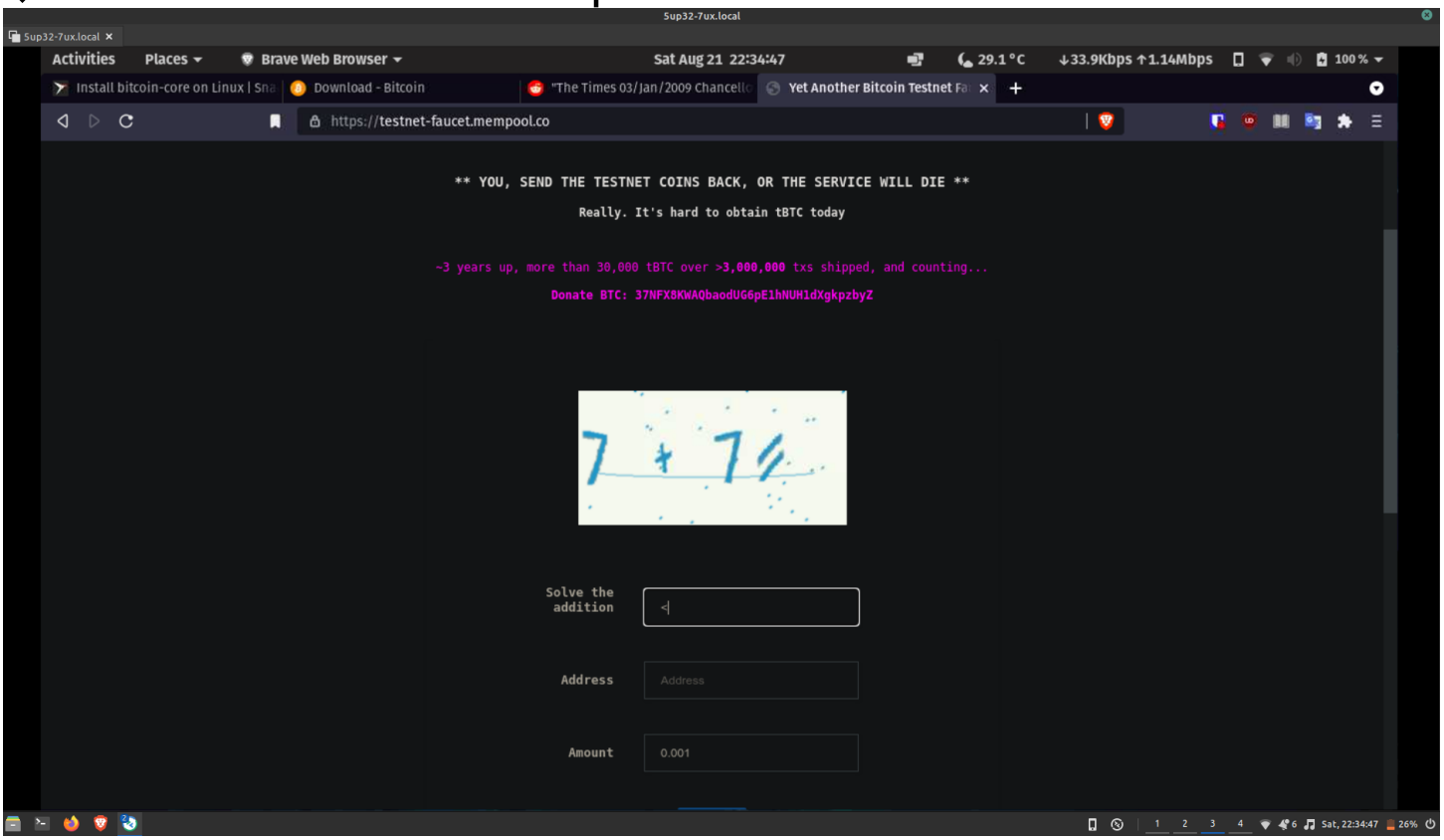
chainstate -> state of blockchain as database

wallets -> address and private key in a file called wallet.dat

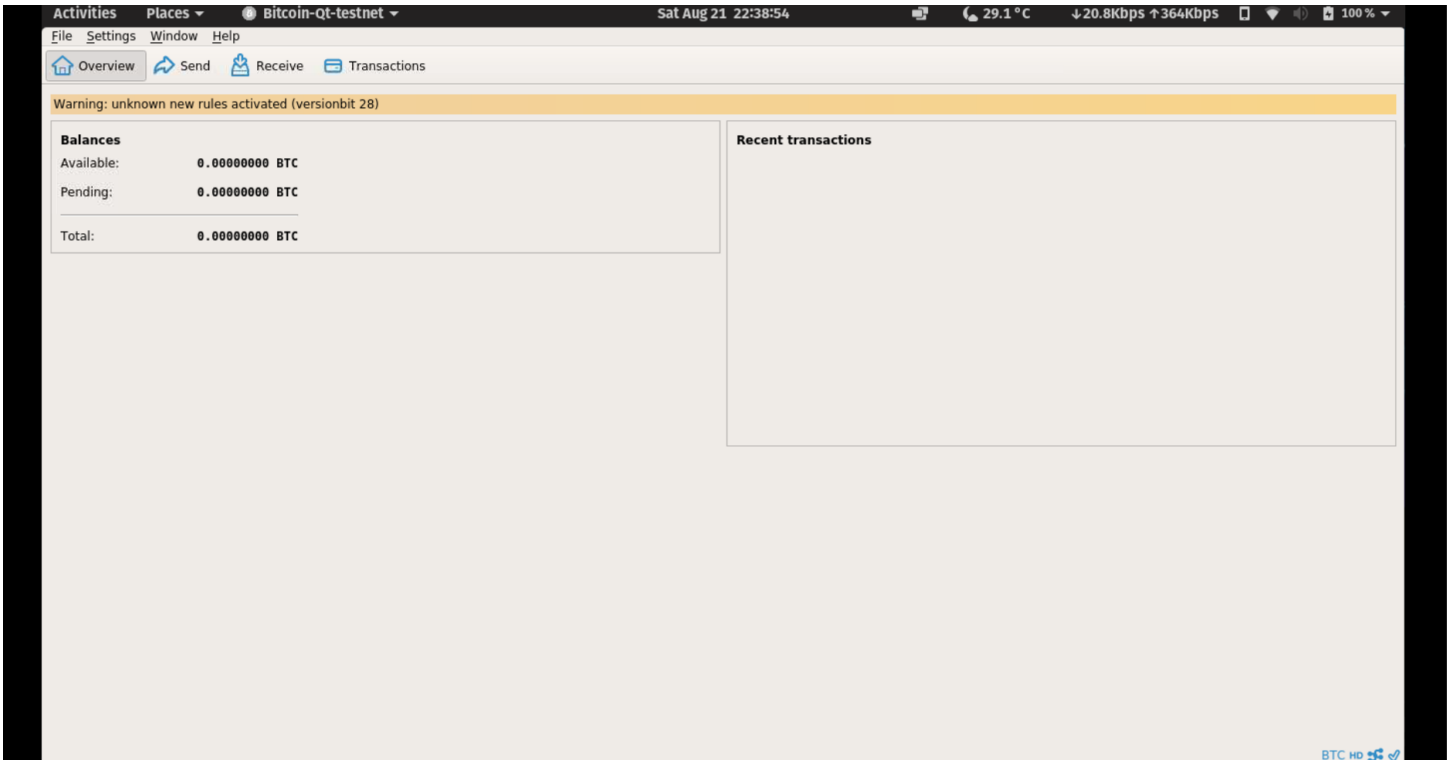
peers.dat -> connected peers
mempool.dat -> contain the transactions ur client has recieved to process
banlist -> block the contacts you dont want to communicate
feeestimate -> details regarding how transaction fees are kept
debug.log -> debugging info

getting test crypto to playwith

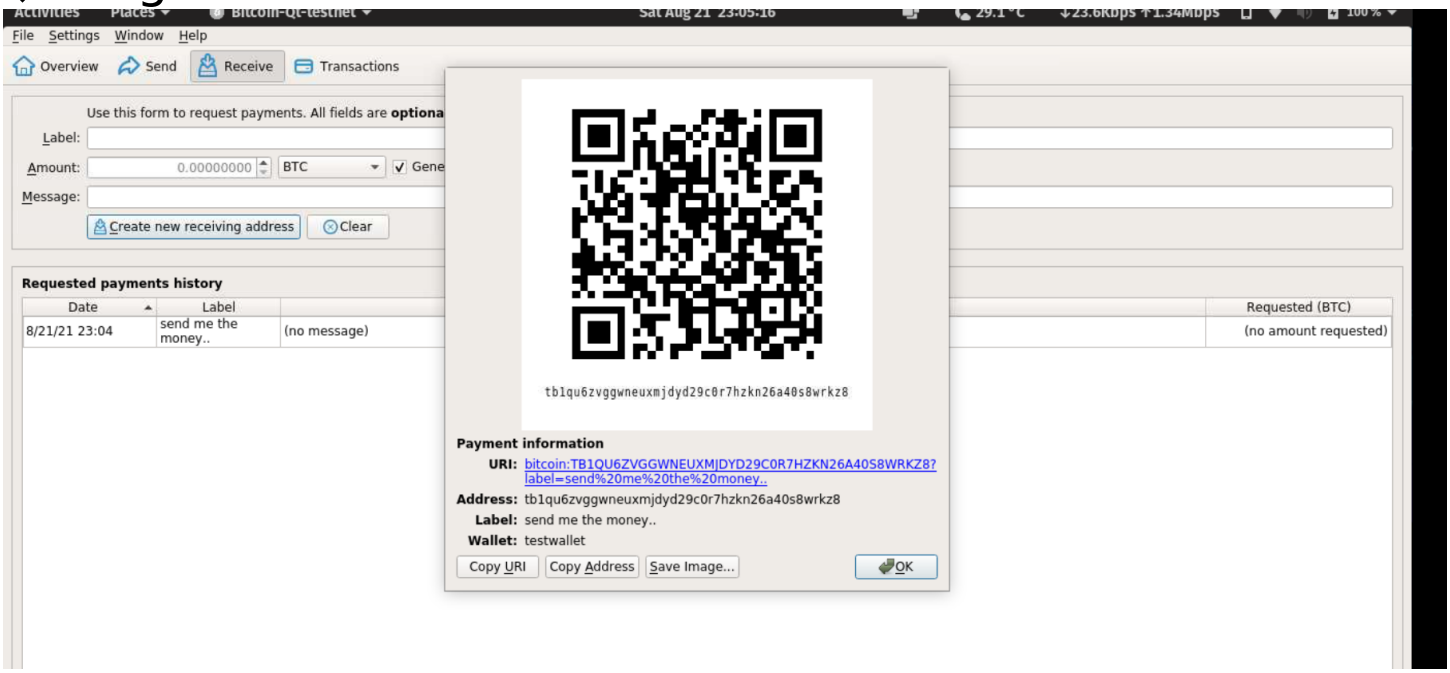
- use facuet
  - ◇ you can request test bitcoin to the site
  - ◇ this is not testnet but is an account with lot of “dummy coins”
  - ◇ only obligation is what ever you get you return it back
  - ◇ “testnet-faucet.mempool.co”



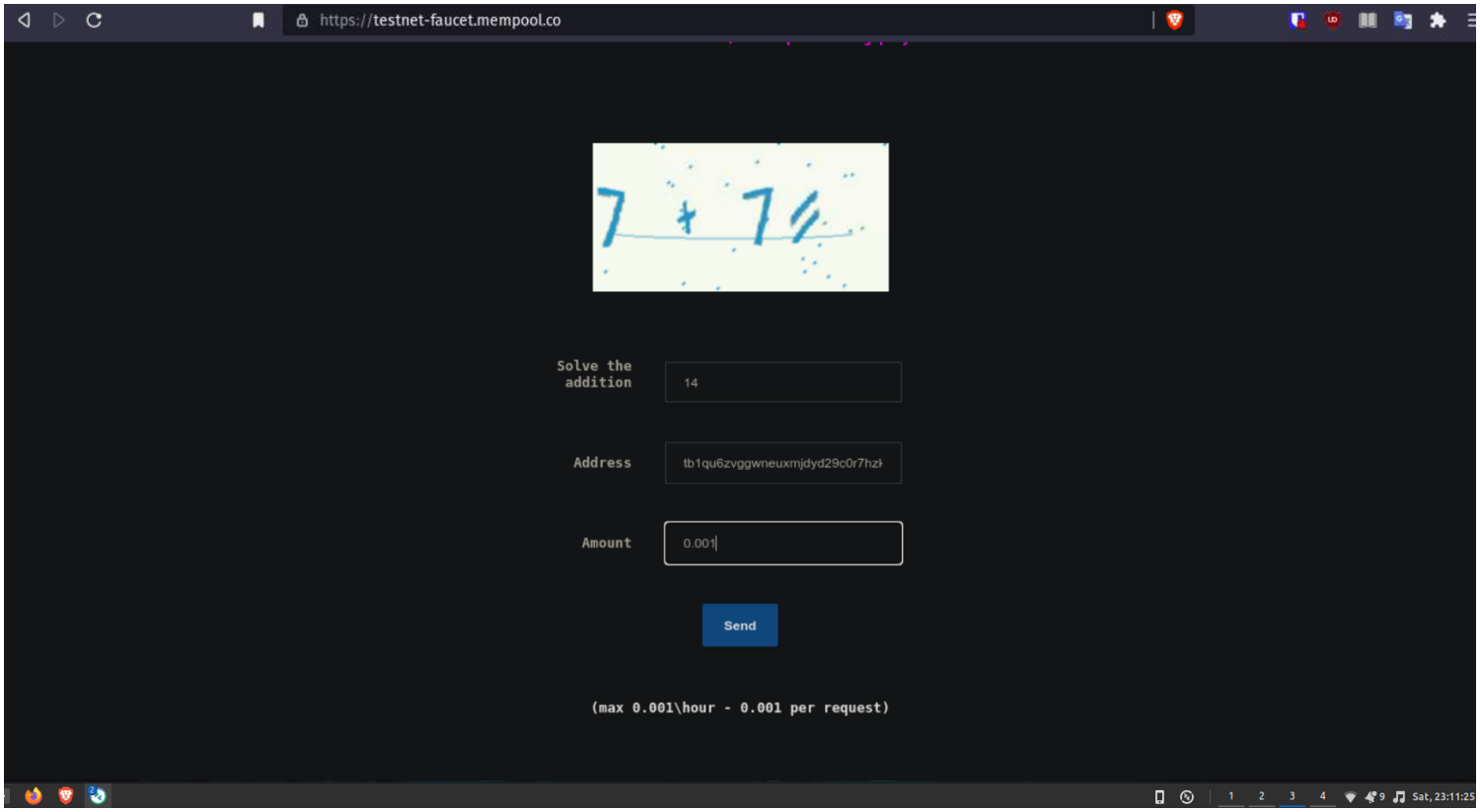
1) create a new wallet and address if you want

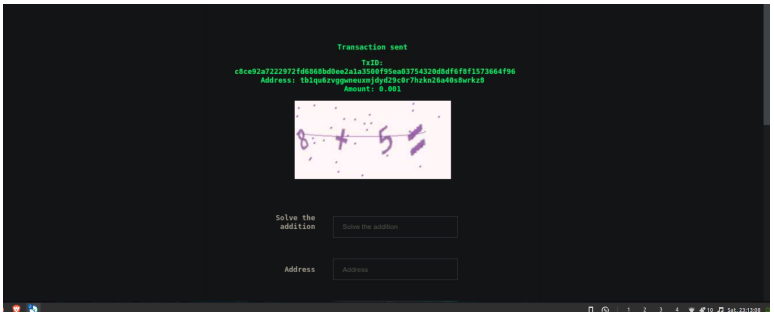


- now create a address
  ◇ length is 20-32 characters

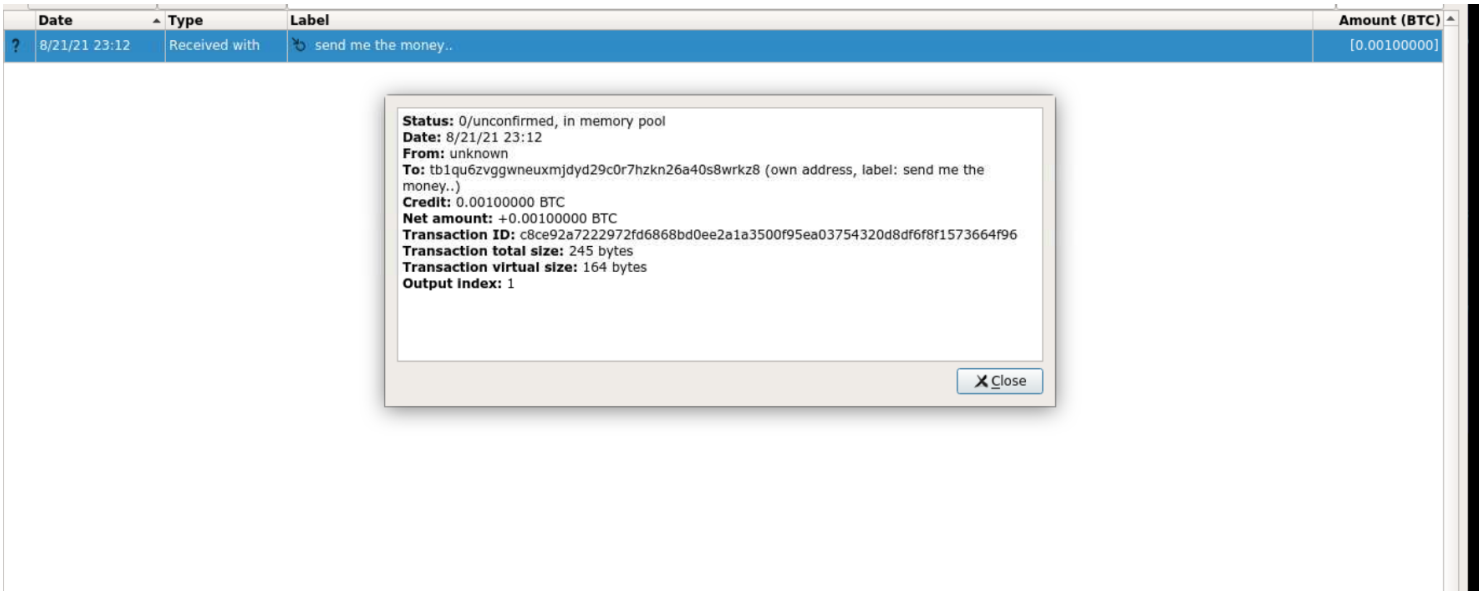


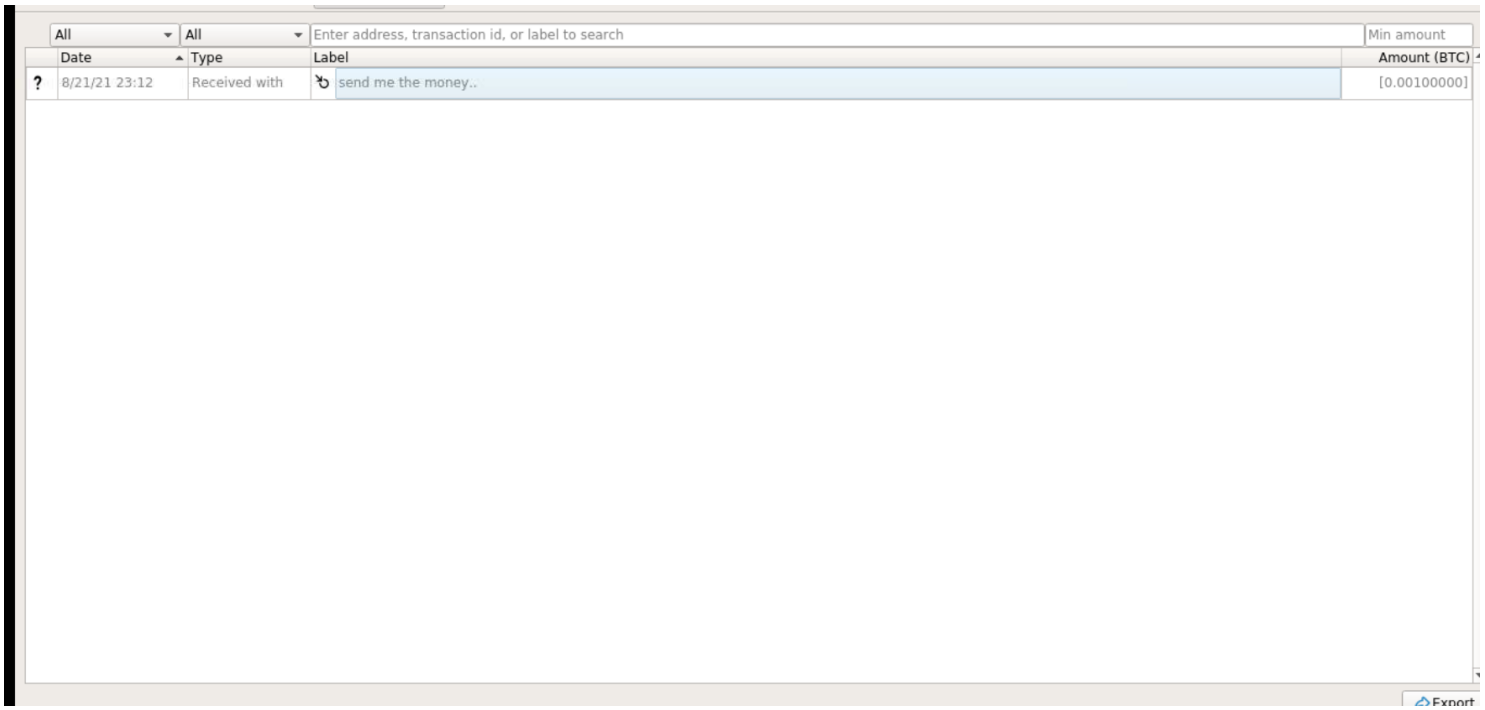
- copy the btc address and paste in faucet to get the money from it.



transaction sent to account 

status



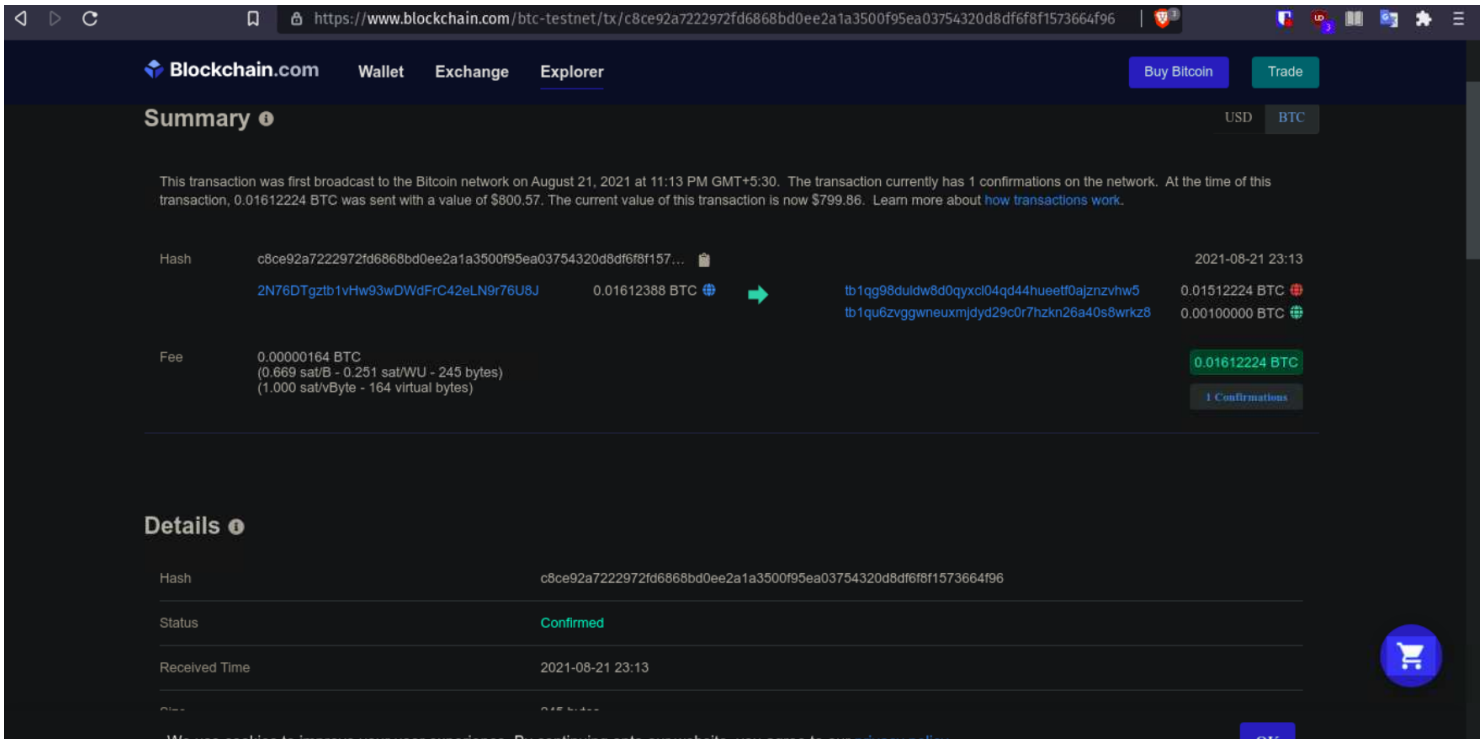


transaction is pending, because it is verified by the network ie miner

- recvier and sender account is valid?
- sender is having enough btc

but since it is peer to peer network, the moment the transaction is started, all clients will get the information about it loaded on its mempool.dat

- minimum 6 people need to verify that the transaction is valid(this is for mainnet)
- for testnet only one node needs to verify
  - ◇ process takes min 10 mins
- verified transactions gets written on the blockchain
- bitcoin is a digital and virtual currency ie it is just recorded in transaction and you cannot physically see it



lecture 04

- BTC, HD
  - ◇ HD -> hierachical determinsitc wallet
    - ■ hierarchical -> has well defined inheritance
    - ■ deterministic -> defined output - input pairs
    - ■ most wallets are HD wallet

how btc public-private key is generated:
when new wallet option is clicked:

private and public key points are generated on elliptic curve
pub key -> sdasda (in hex)
priv key -> dsadd (in hex)

operations performed on pub key to make it a bitcoin address:

1) pub key **-SHA256->** sha256_output1
2) sha256_output1 **-RIPEMD-160->** r160_output1

3) make a exact copy of r160_output1 ie r160_output1copy

4) r160_output1copy **-SHA256->** sha256_output2 **-SHA256->** sha256_output3

5) take first few bytes of sha256_output3 and append that to r160_output1 to get publicaddr_raw

6) publicaddr_raw **-Base58CheckEncoding->** publicaddr_final (20-32 alphanumeric characters)
   1- before encoding, a hex value is prefixed which represents type of address (mainnet or testnet), compressed or uncompressed, private or public key etc

metamask - browser based extension simple node + mobile app

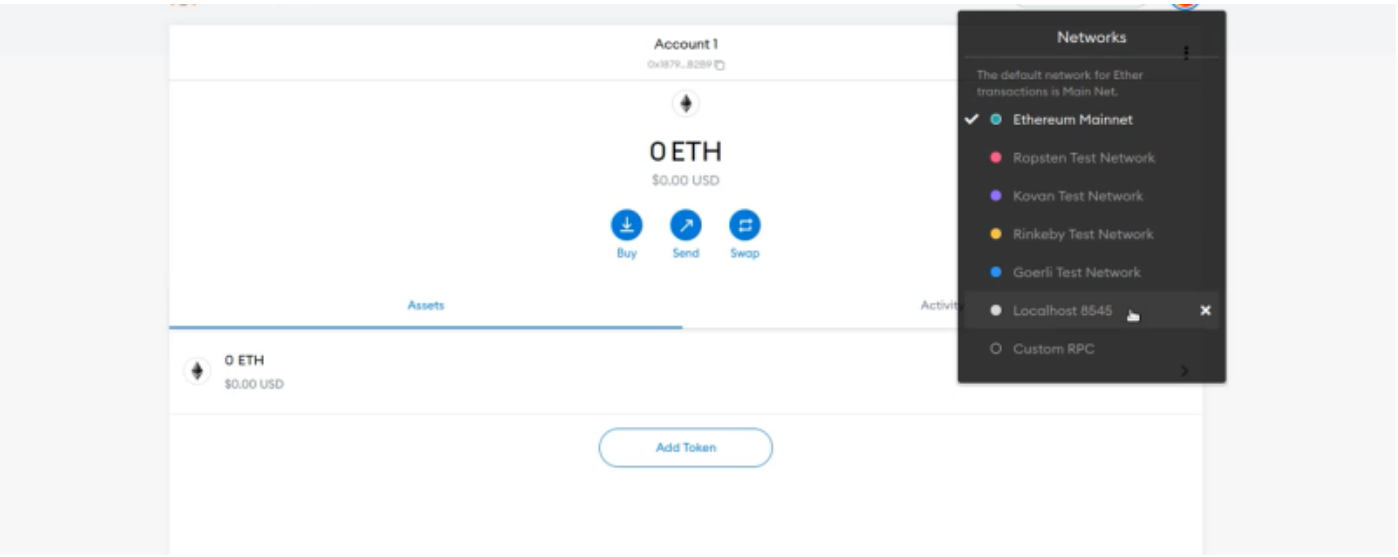
• we can use this to connect to ethereum blockchain and any decentralized web as well
   ◇ ex: binance smart chain (BSC mainet)
   ◇ mumbai test net etc
• we can either import or set up a new wallet
   ◇ if set up a new wallet create password and a secret code (for account recovery they will give 12 random english words)

- aka secret phrase
- aka mnemonic
- this is for wallet password not blockchain
- blockchain has no password only private key

• HD wallet

◇ for pseudo annoymity each user can have multiple addresses

- for different types of ethererum nets in case of metamask
  - like ropsten
  - koven
  - rinkeby
- we also local client or remote client etc
- mainnet is there as well but easier to switch between nets



◇ each address is a private key

◇ when ever you want to regenerate addresses

- addresses are stored in a hiearchical way
- if you want to regenerate, plug in the secret code
- all addresses are obtained in a deterministic way
  - “for these 12 english words for a particular currency” -> will always generate the hierarchical addresses

    ■ this is called BIP32 standard

• ether -> ETH
• the address you have in mainnet will be there in all the testnets
  ◇ but purpose is different
  ◇ same account number is there in multiple nets but different balances are there in diff nets based on transactions
    ■ no encoding is there in eth for addresses directly hexa they will use
    ■ in bitcoin we have for each net we have different account number
      - for purpose of distinction
depends on gas fees
      transaction fees is calculated
      more gas fees less time to complete

• for small transactions you need to pay higher transaction fees
  ◇ so for everyday paytm like usage ethereum is not good
  ◇ for size of data stored ie for smart contracts, NFT etc then alos storage fees is very high

faucet for metamask -> facuet.metamask.io
crypto wallet can also store crypto assets.

lecture 05

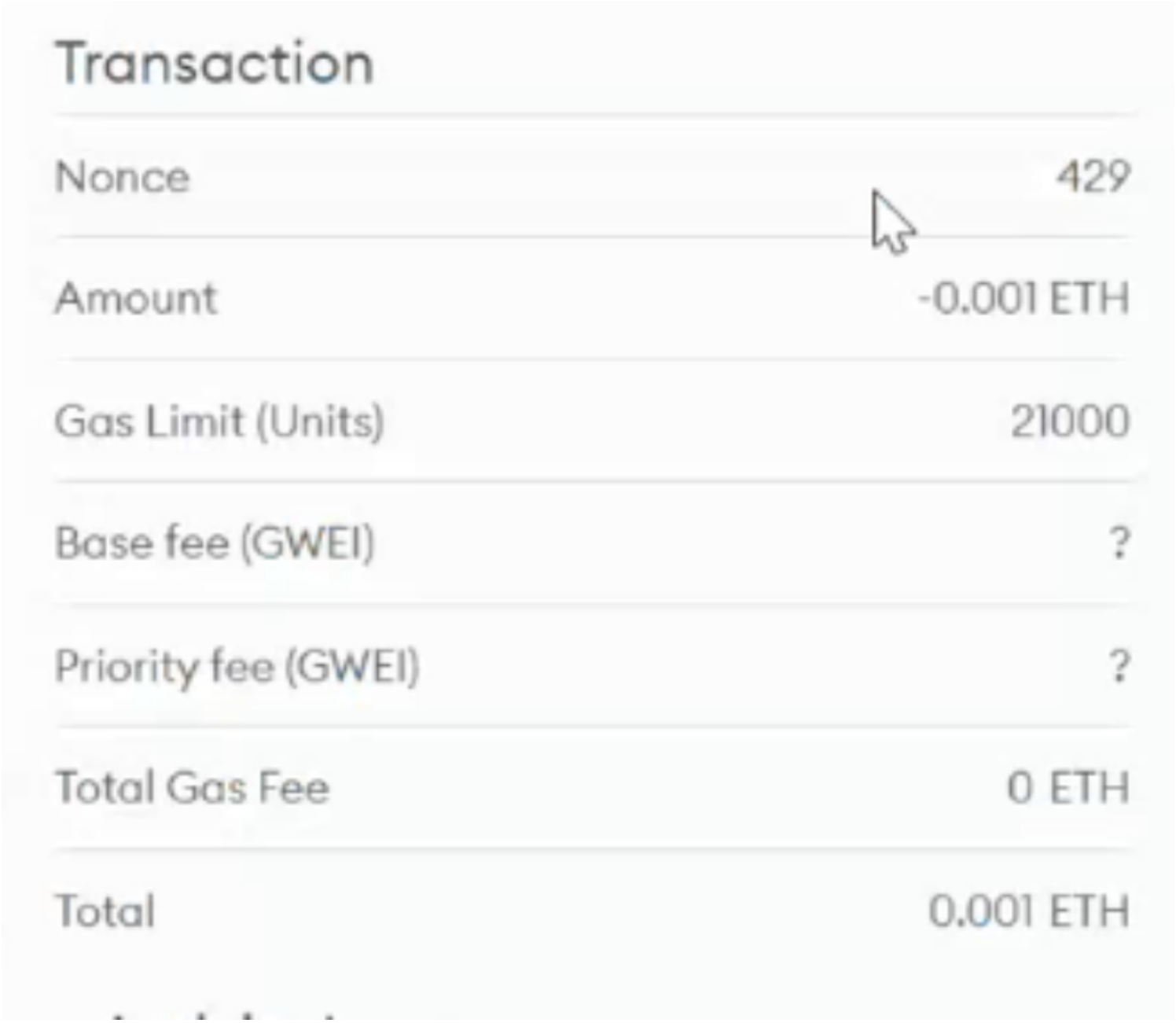
• prerequisite
  ◇ cryptography

  ◇ mathematics
  ◇ databases
  ◇ how internet and distributed systems works
  ◇ networking basics
  ◇ programming
• public key
  ◇ address you want to transact with
• private key
  ◇ hash of the message your sigining with private key
    ■ aka this is the signature
• cryptographic hash functions are one way mathematical functions used to provide immutablity to data
• properties of hash
  ◇ preimage resistance
    ■ if i have input very easy to find output
    ■ if i have output practically difficult to find input
  ◇ second preimage resistance
    ■ for m1 input we have output H
    ■ for m2 input we should not have same output H
  ◇ collision resistance
    ■ it is not possible to find any two different input which gives same output.
  ◇ avalanche effect
    ■ small change in input gives huge change in output
    ■ ie deterministic attacks fails
    ■ puzzle friendliness
  ◇ MD (message digest)
    ■ MD2(dead)
    ■ MD3(dead)
    ■ MD4(dead)
    ■ MD5(dead)
  ◇ SHA (secure hash algorithm)

- SHA1 160bytes
- SHA2 224,256,384,512bytes
  - blockchain uses SHA2-256
- SHA3 224,256,384,512 bytes
  - ethereum using keccack-256

◇ RIPEMD 160

• transaction ID → hash of transaction details



all the details are hashed and called as transaction ID

• computed by client and attach it to the transaction
• this details is send across all nodes
  ◇ transaction details are verified in public blockchain
  ◇ integrity of the detail is verified by digital signature
• as we have multiple transactions at the same time or similar time
  ◇ transactions in mempool are verified and are put in block
    ■ block is broadcasted to other nodes to be verified by consensus algo

minning:
• node will group all the transactions together add some data to it to form a block(including hash of previous block)
• block is verified and crypto puzzle is solved to add proof that he did the work correctly
  ◇ correct block means, the node gets the coins coming from new block as well as transaction fees
  ◇ transaction is verified and then money from ur account gets reduced only if transaction ends up on block ie transaction is complete
    ■ this is called finality of transaction
    ■ a block to be created requires 10 mins on avg in bitcoin
      - this does not mean ur transaction gets confirmed in 10 mins
      - depends on whether ur transaction is in the next block to be created or not
      - this consensus is bitcoin is called Proof of Work
        → to reduce congestion
        → to verify integrity of transaction
        → to scale difficulty based on no of people in blockchain and tech used

→ PoW is cryptohash
⇒ solution is complex
⇒ verification of solution is easy

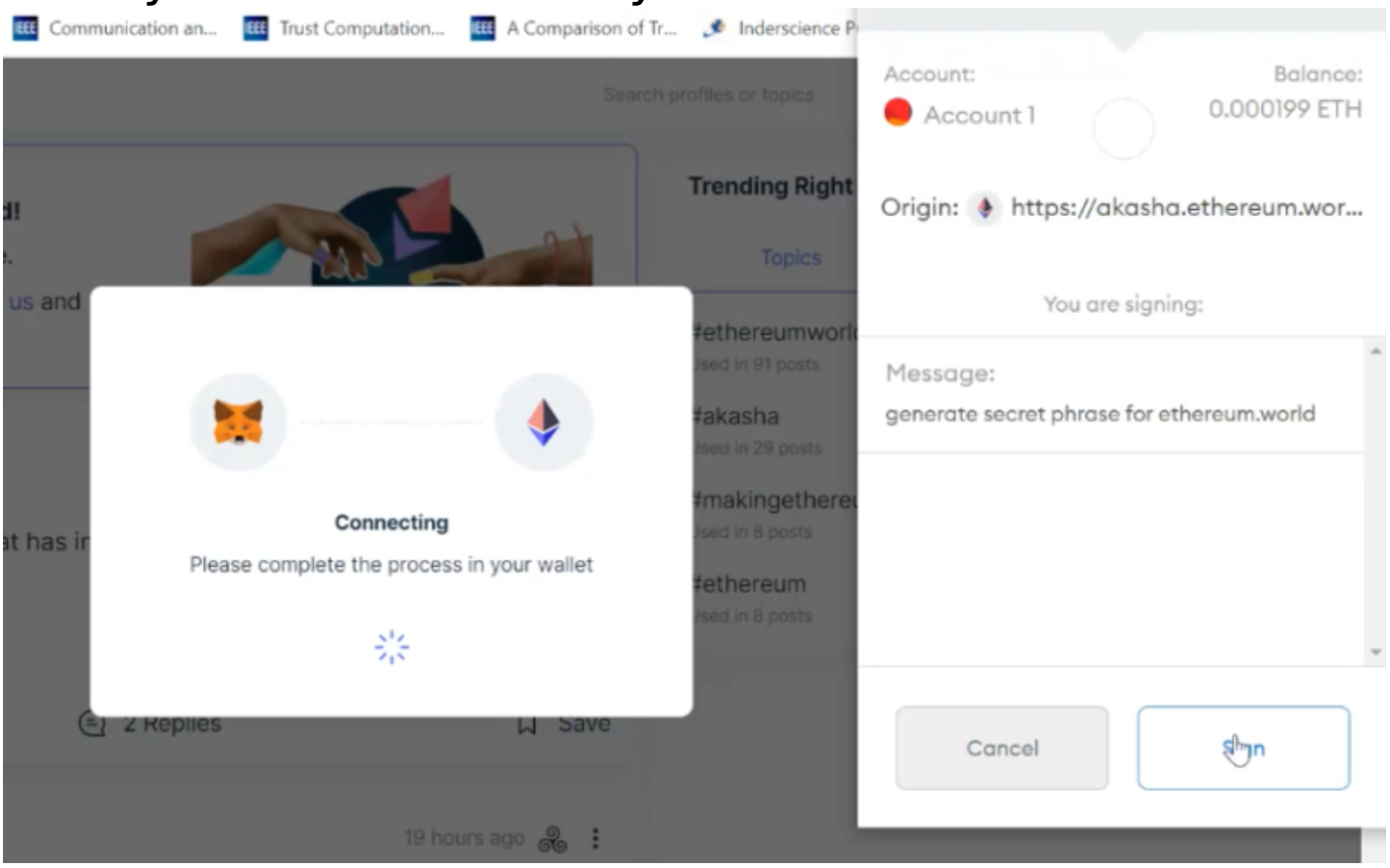
• difficulty parameter → measure of how difficult to mine a block
◇ reverse computing of nonce from hash value of block is difficult
◇ will say the first x characters of hash value is zero
◇ you have to bruteforce values starting with a nonce to figure out what is the nonce that causes this hashvalue
◇ the hashvalue, the number of x characters and difficulty parmater are all fixed by blockchain algorithm
◇ difficulty changes in bitcoin blockchain every 2016 blocks

• this nonce and the rest all are included in proposed block and send to others
• others will calculate the hash of the block and verify if the hash obtained is matching the difficulty to verify and confirm the block
• immutable in the sense all the blocks are connected via previous block hash

Lecture 06

• cryptokitty NFT
■ indivisble
- we can use “ethitem” to “fractionalize” NFT
→ wrap erc20, erc721 etc to mask NFT behavior and then fractionalize it
→ the main NFT in blockchain is burnt ie no one

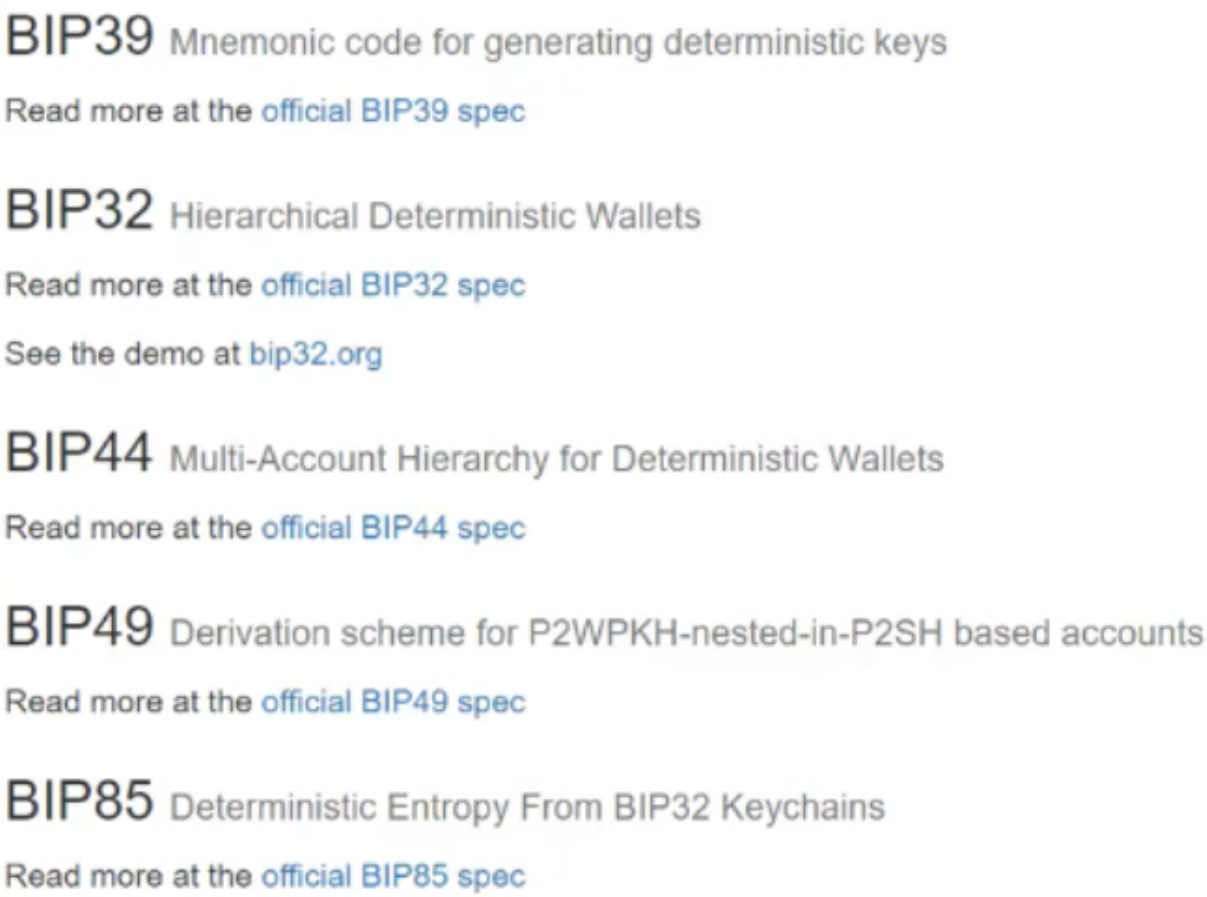
can spent or own the NFT
    - cannot be exchanged with other NFT of same chain
    - globally unique
  - ◇ first token based on ERC721 standard
  - ◇ value based on uniqueness and rarity
  - ◇ tokenization
    - digital conversion of asset and put that in blockchain
    - diginoor → fan movie clipping is billed as NFT
    - cryptopunks
    - mooncat rescue
    - pixelchain art
    - marble card → polygon matic
    - unlandme → virtual realestate
- decentralized social media applications
  - ◇ akasha.ethereum.world
    - preview ie alpha phase in rinkeyby test net
      - signin process
        ⇒ generate digital signature using user privatekey to validate identity



• Ad rewarding the user to click via the brave basic attention token issued on top of ethereum
  ◇ all the ppl who are using brave browser will get the ad
    ■ will reward the person who saw the ad with BAT
    ■ can also directly support the person who created the ad
  ◇ anyone can post ad
    ■ will get higher value to the one who posted the ad
      - as no middle men like fb ads or google ads or apple ads
    ■ so it is anti monopoly

Lecture 07

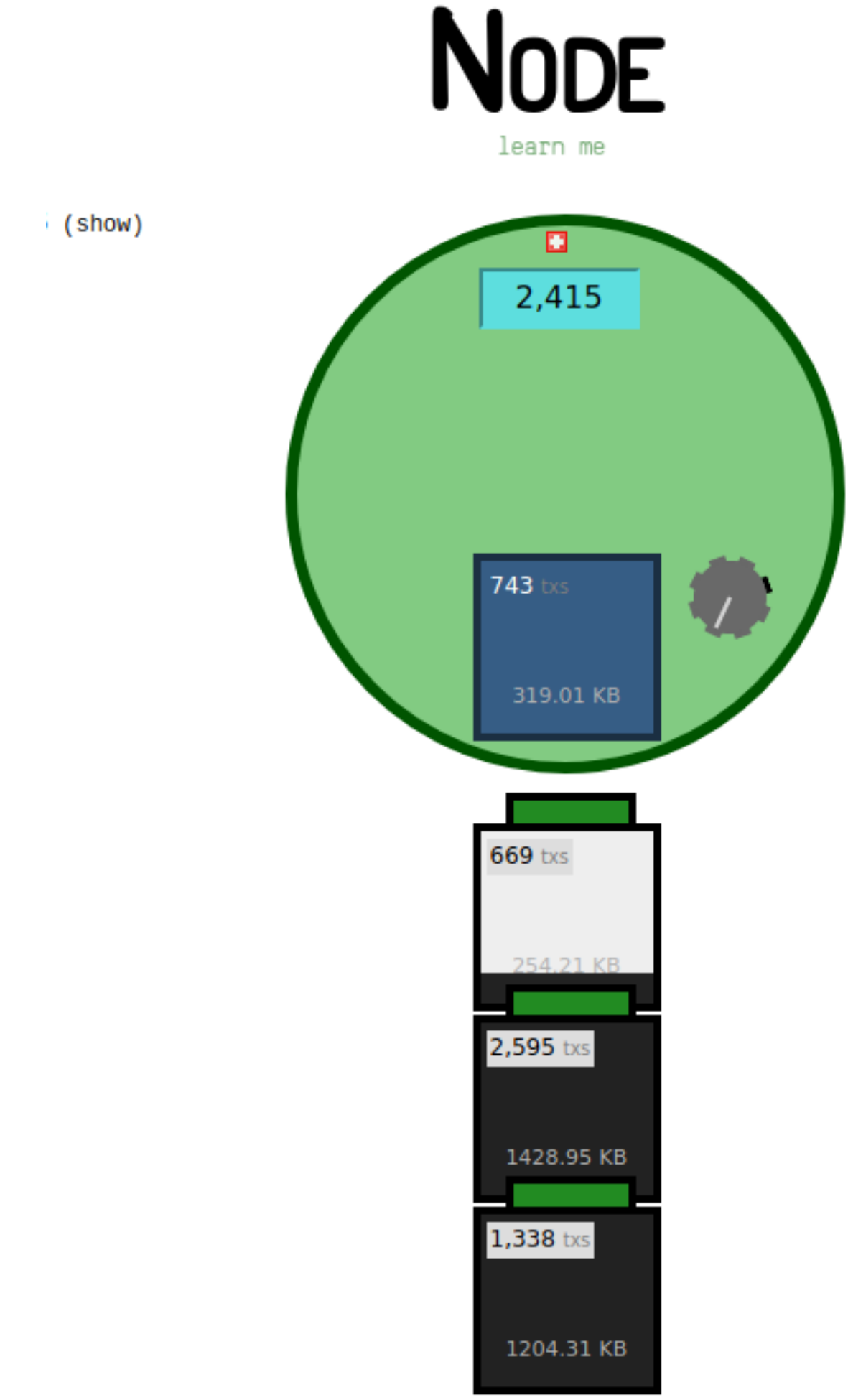
• HD wallet is proposed by bitcoin, this is BIP32,BIP44,BIP39 etc



◇ metamask using BIP44(EIP44) for multi account HD wallet
◇ BIP32 is the normal single account HD wallet
◇ secret phrase can be of any size(default 12) in any language
◇ so it is part of BIP (bitcoin improvement proposal)
◇ ie subsequent public private key pairs are dervied from single public-private key pair

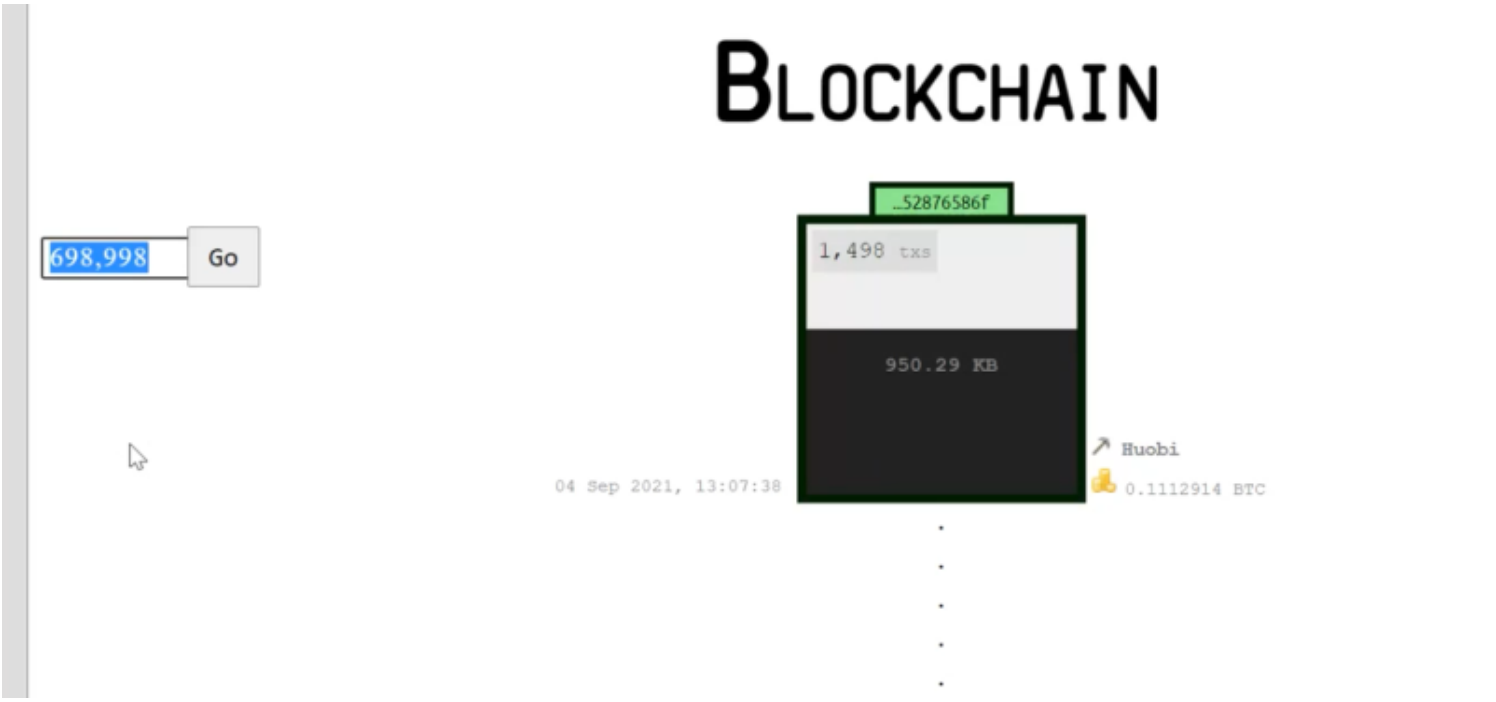
- fork
  - ◇ hardfork ie create a new chain
    - ■ from Bitcoin (BTC)
      - Bitcoin cash (BCH)
      - Bitcoin Cash SV (BCHSV)
      - Bitcoin Gold (BTG)
    - ■ from ethereum classic(ETC) (old version)
      - → Ether (ETH) (new version of ethereum)
- mutable “blockchains”
  - ◇ ie redactable blockchains where we can modify/-delete past transactions

Lecture 08

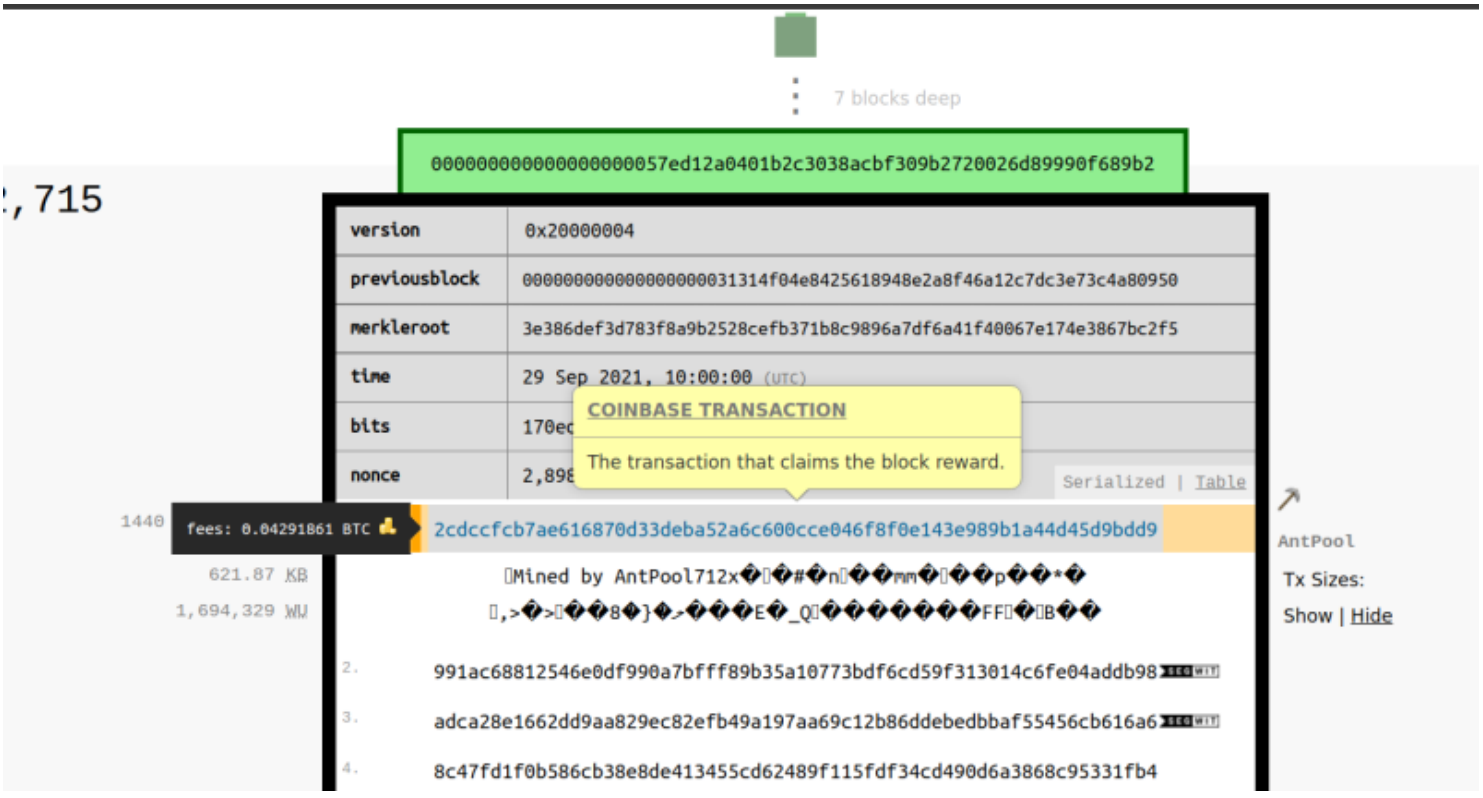


• 2415 confirmed transactions in mempool, with 743 in new proposed block by node, with gear reprsenting

difficulty
• new proposed block is the candidate block as other miners also will compete for bitcoins
  ◇ validity vs completeness or finality
• all confirmed blocks are having different number of transactions ie max 1 to 4MB like that
• block number is same as block height
  ◇ 698998 is blockheight
• hash of the block is not stored within the block as indicated by green color box and is only part of the next block
• the current block does not know its hash



block height 7027115 mined at 10 am, 29 sep 2021 by antpool got transaction fess of 1440 transactions plus of 04291861 BTC



the first transction is block is the coinbase transaction which is automatically generated and is the transaction reward of the miner.

the block hash is basically the hash of the block header.

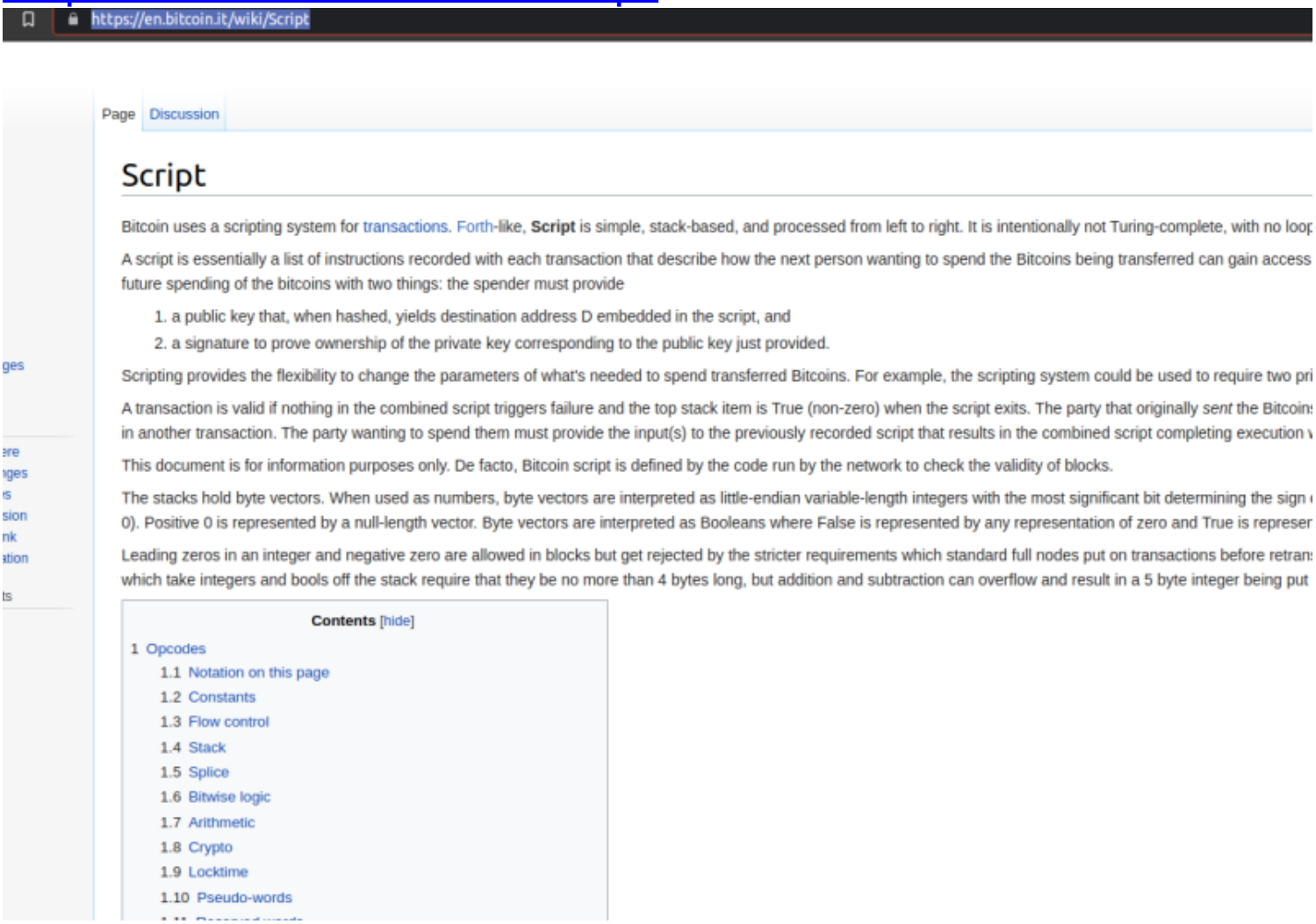
• transactions are secured via merkle tree
  ◇ we can which transaction got tampered by the merkle root hash
  ◇ the tampered transaction will affect all the parent hashes above it till the merkle root of the tree
  ◇ we can reconstruct the entire tree from non tampered hashes of other nodes to tell ok this transaction is wrong
  ◇ ethereum uses Patricia merkle tree which is an efficient form of merkle tree

public --> no sensitive data stored
private → sensitive data stored
permissoned → regardless of data stored you need permission to access

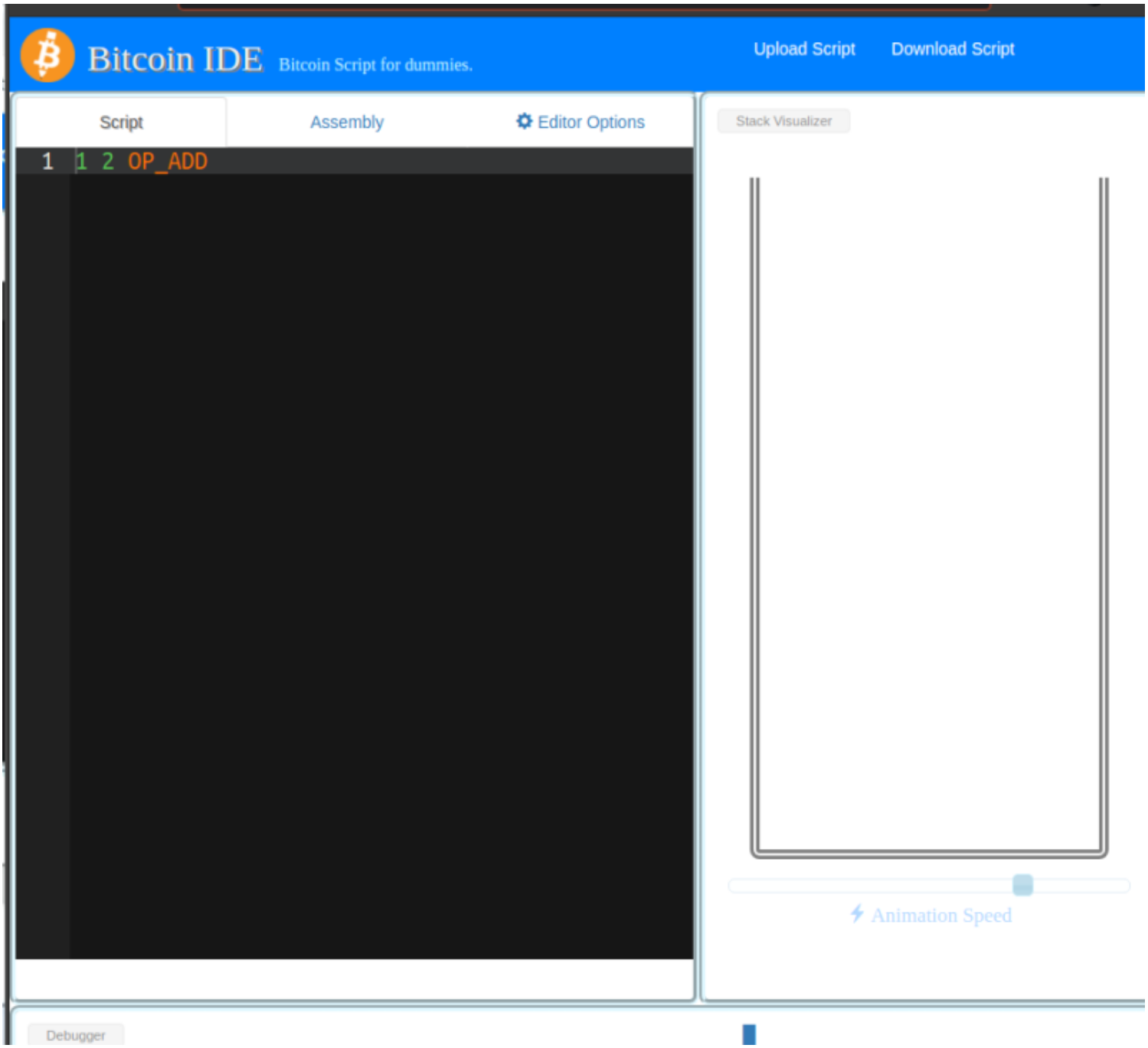
lecture 9

• in bitcoin, each address is like one bank account
• so we can have as many bank accounts as we want for all unique transactions, this aids in psuedo privacy
  ◇ this is part of UTXO
  ◇ we can have multiple accounts sending to single/-mutliple accounts or viceversa
• in ethereum, we are sending all stuff to single wallet address
  ◇ we can have as many wallets as we want
  ◇ this is acccount based model

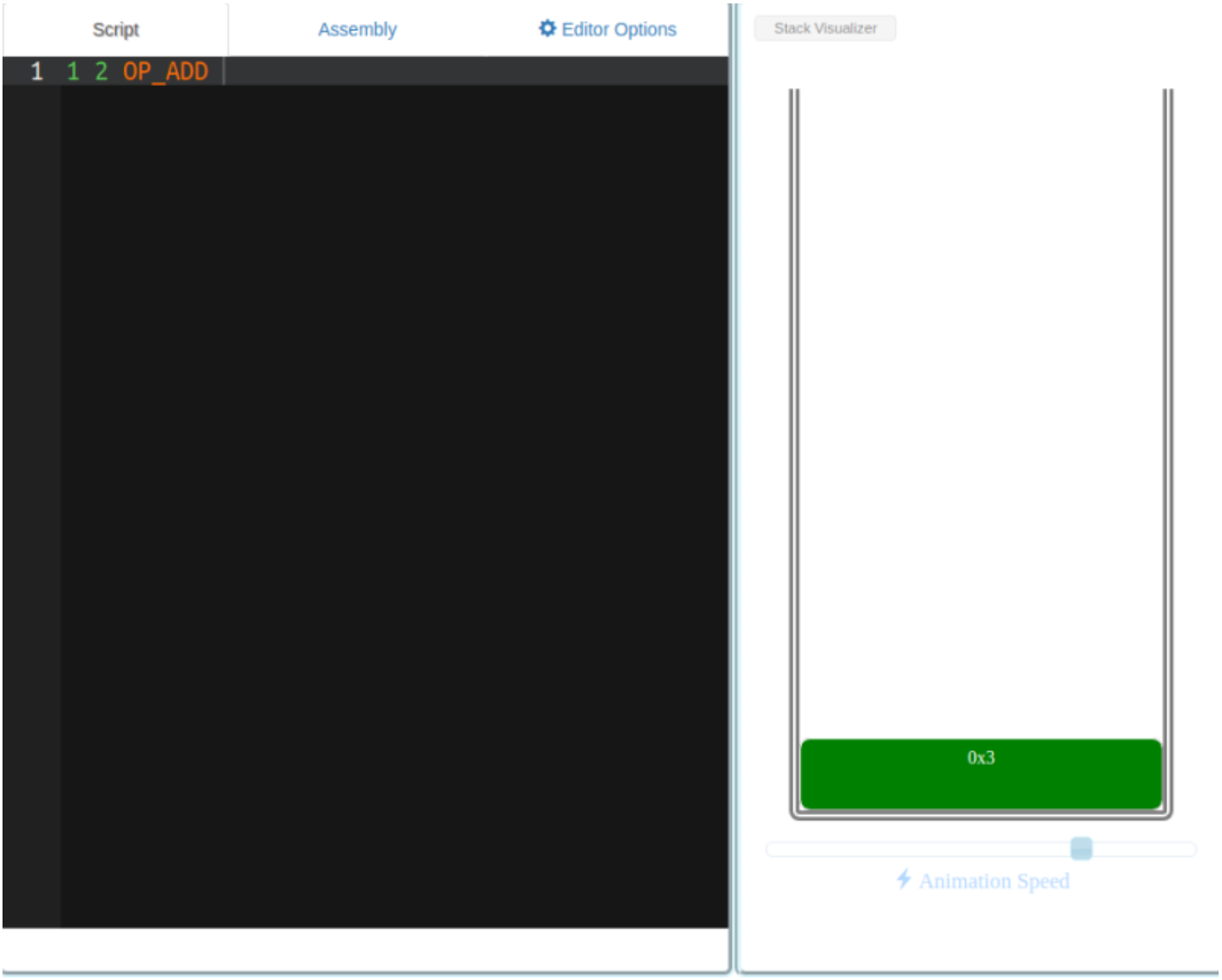
bitcoin script:
https://en.bitcoin.it/wiki/Script



https://siminchen.github.io/bitcoinIDE/build/editor.html



- similiar to assembly language code
  - ◇ stack based execution
    - ■ LIFO
    - ■ pop and push based operation
- pay this tx to this person after some time like this we can “code”
  - ◇ this is not as easy or feasible as smart contracts
  - ◇ this is one of the reasons for blockchain 2.0 and application based blockchain



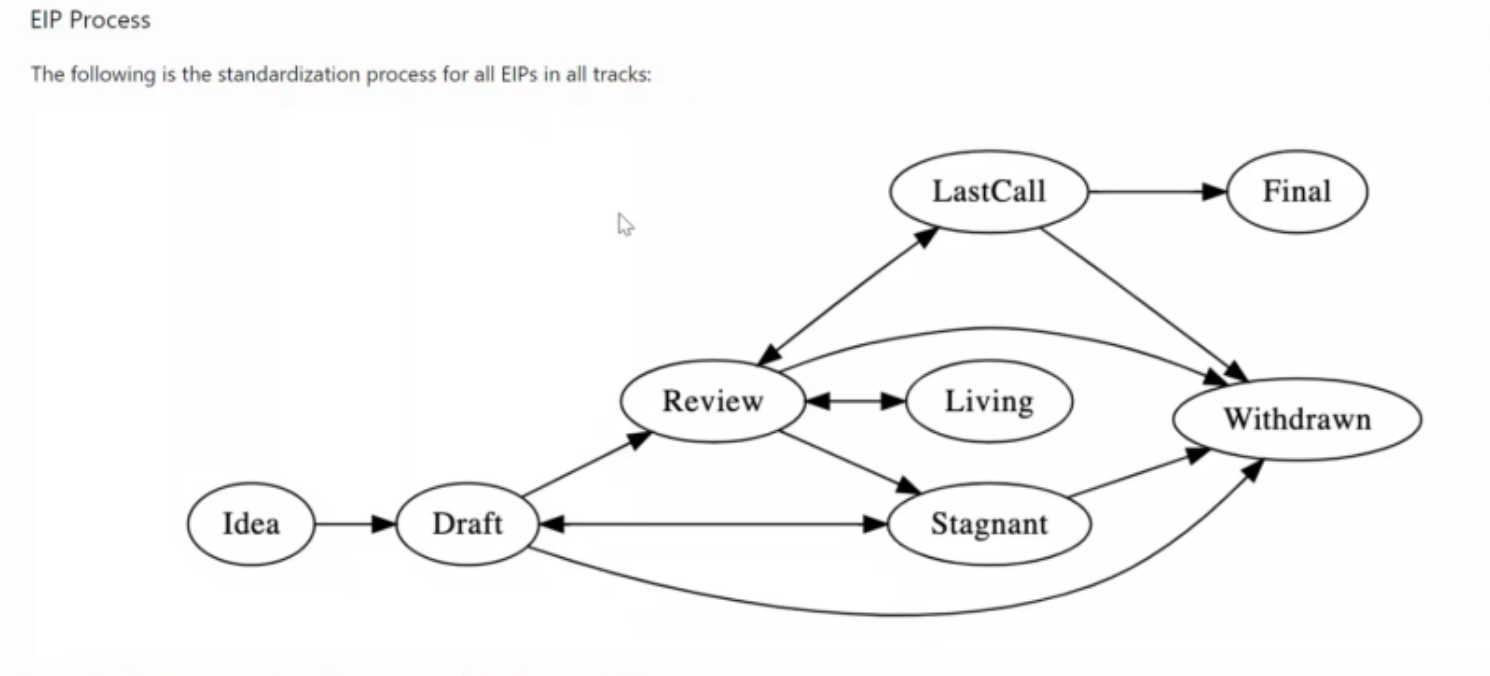
for : 1+2 = 3

	Script
	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

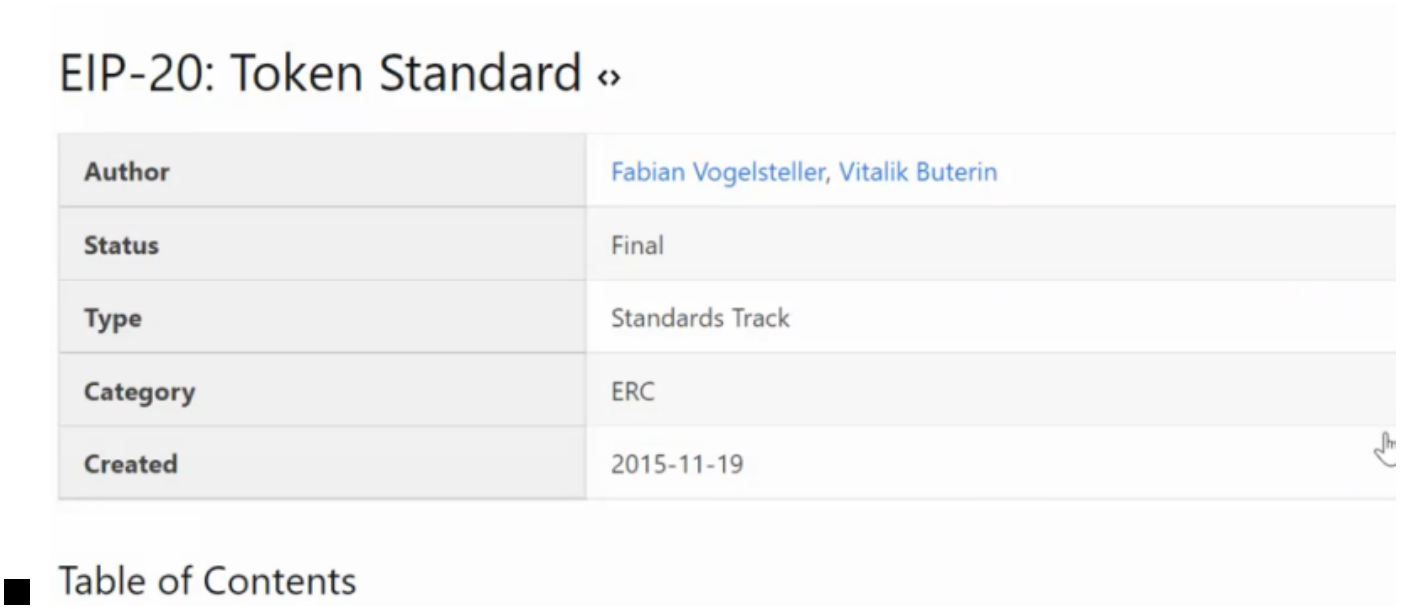
script to check if authorized user is only sending

lecture 10

- BIP
  - ◇ bitcoin improvement proposal
  - ◇ if it is implemented status is final
  - ◇ if it is replaced something, that something status is replace
  - ◇ even if it is implemented or dead, bip is always named as bip
  - ◇ bip-1 → bitcoin introduction and scope work etc
- EIP
  - ◇ EIP → ethereum improvement proposal



  - ◇ ie centralized decision are made as to how this is implemented
  - ◇ a idea which is finalized is called ERC
    - ■ ethereum request for comments
    - ■ EIP20 became ERC20 (for fungible tokens)
    - ■



block 478558 is the hard fork point from which BCH came from BTC

- uptill this point BTC and BCH blockchain are identical
- from BCH, we got BSV
- from BTC we got bitcoin Gold

lecture 11,12

why ethereum

- what i need is not provided by bitcoin
  - ◇ layer to write smart contract, dapps
  - ◇ easy scripting language integration
- instant payment not possible because of time and transaction payment issues
- what is in bitcoin I dont need
  - ◇ only fungible token support
    - ■ I need non fungible tokens
  - ◇ regardless of transaction everything is average of 10 mins for confirmation
    - ■ consensus for some time critical applications is not covered

◇ PoW is good for ensuring no one can manipulate
■ but is slowly computionally infeasible to invest in a bitcoin operation nowadays

smart contract

• A consensus protocol for correct execution of a publicly specified computer program.
◇ executes terms based on some time/event based conditions
• the code is avaliable in all nodes of network via blockchain transaction
◇ when a smart contract is triggered, it gets triggered in all nodes of the blockchain
◇ once result is generated, it is publicly verified and agreed upon
■ the result is stored as a transaction

distributed applications

• set of smart contracts running on every node
• smart contract is not legally valid outside of ethereum
• ricardian contract
◇ form which has printable in a human readable format
■ for court solving , we can use this as this is english format
◇ using this as input a template code uses this and deploys as a smart contract
• ETH is 10^-18 divisible max
◇ more transactions are there
■ transactions in applications
■ applications
■ crytpo etc
◇ so we need to pump more money
◇ ethereum has no defined money upper limit

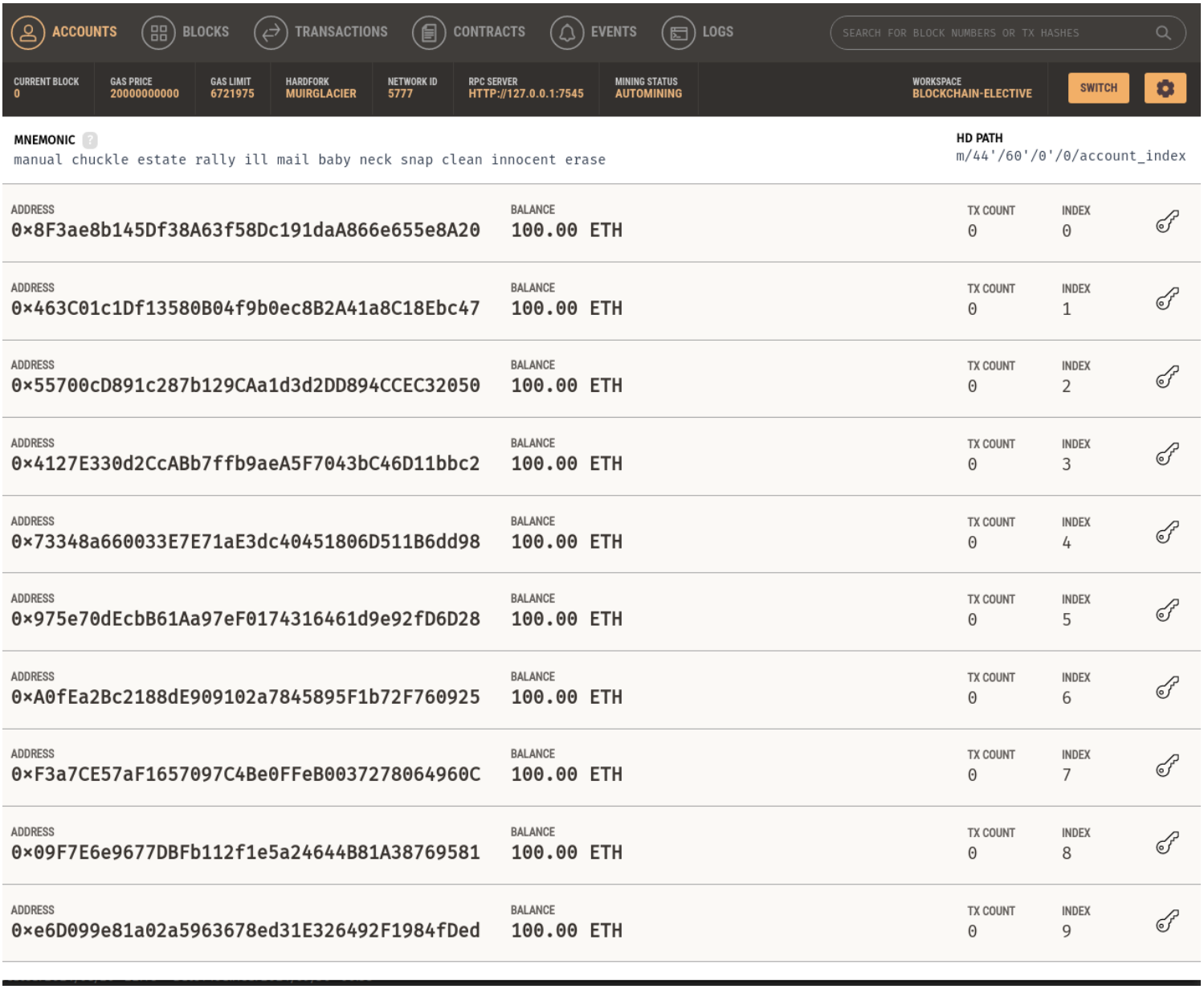
Unit	Wei Value	Wei
wei	1 wei	1
Kwei (babbage)	1e3 wei	1,000
Mwei (lovelace)	1e6 wei	1,000,000
Gwei (shannon)	1e9 wei	1,000,000,000
microether (szabo)	1e12 wei	1,000,000,000,000
milliether (finney)	1e15 wei	1,000,000,000,000,000
ether	1e18 wei	1,000,000,000,000,000,000

- we need some dev tools for writing and deploying smart contracts on blockchain
  - ◇ IDE
    - ■ compiler
    - ■ deployer
    - ■ helpful for debugging,performance etc for debugging
  - ◇ local support for development
    - ■ inbuilt wallet
    - ■ inbuilt blockchain explorer
    - ■ local blockchain
    - ■ default is preloaded with currency
    - ■ private blochain for development and testing
      - truffle framework(for smart contract developement) + ganache(private self contained blockchain)

→ **ganaches use port number 7545**,

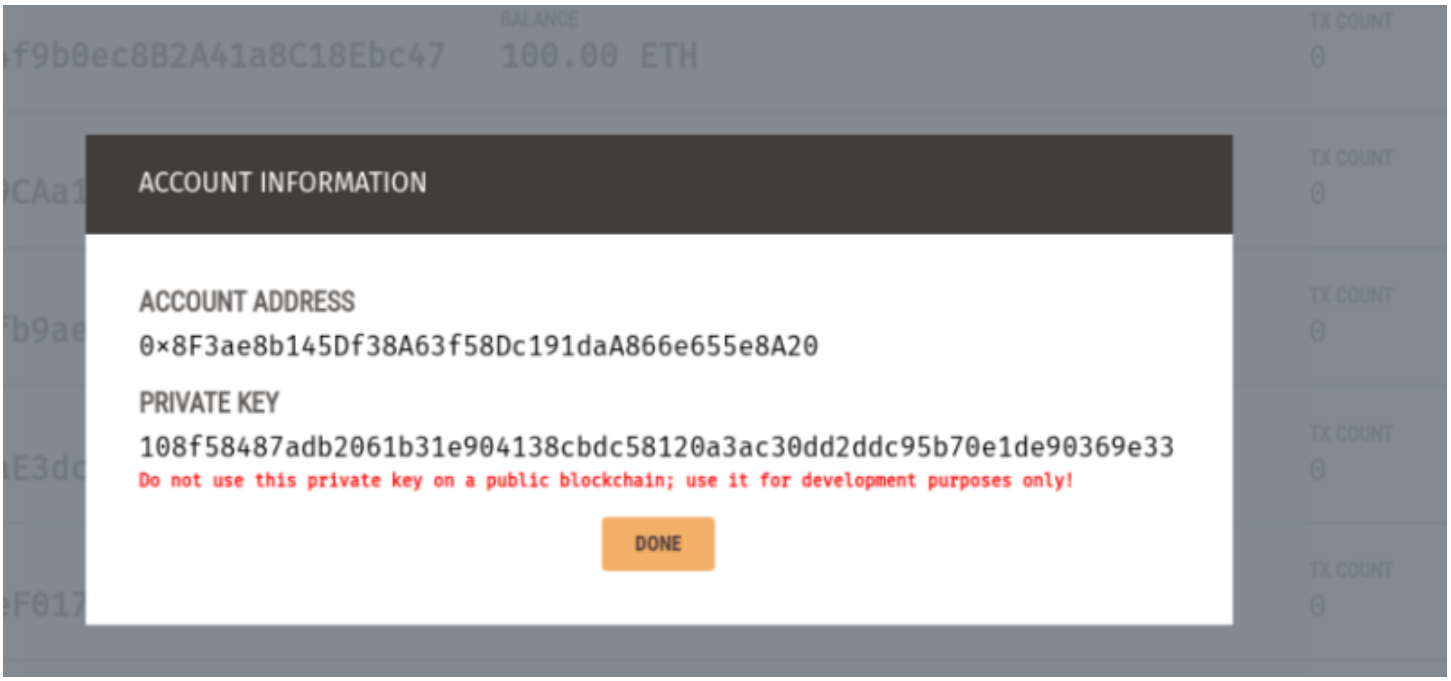
ganche:

click new workspace to save ur settings after u quit



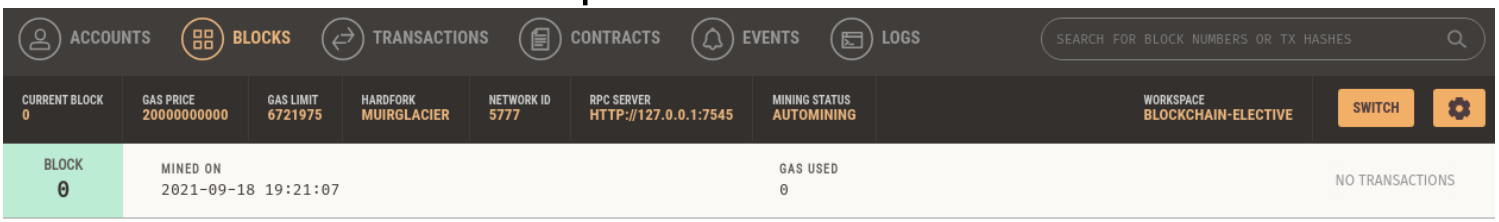
you have on opening

- 10 wallets with each 100 eth
  ◇ private keys are hidden behind the key icon next to the index



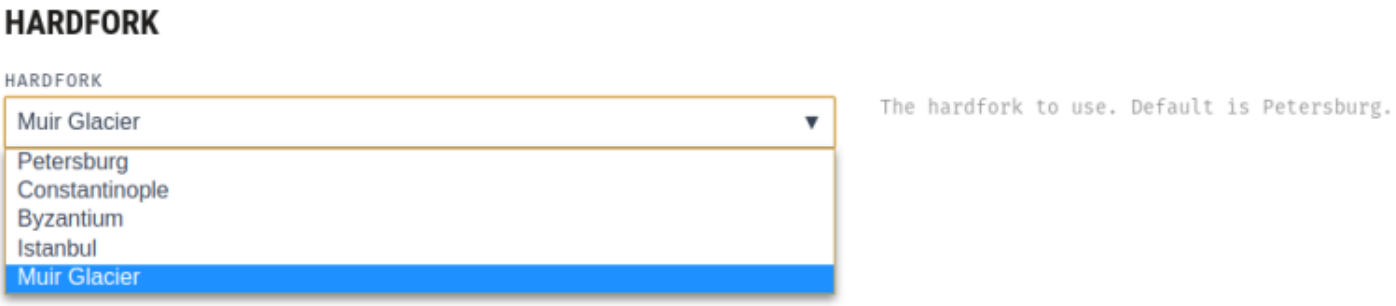
- HD wallet with eip44
- a miner which is on automine
- connection to remote process server

- in built blockchain explorer



•workflow
use truffle framework to write smart contract, compile on inbuilt solidity compiler and deploy it on ganache
- names of ethereum hard forks ie different incompatible

versions, latest hardfork is muir glacier



- has two transaction types
  - ◇ contract call
  - ◇ contract deployment

installing and working with truffle

- install nvm, use it to update to latest stable version of npm and use that version
- then install truffle globally, npm install truffle --global
- then create a blank project
  - ◇ truffle init
  - ◇ in that truffe configjs file is there which helps to talk with ganache
  - ◇ test folder contains your custom test cases
  - ◇ migration folder is where you write code to deploy to ganache
  - ◇ contract folder is for smart contracts

we can download a preconfigured and working project called a box to test out application

- the box we use is called pet-shop
- command truffle unbox pet-shop

```
$ ls
gokul ~/Software/custom/mtech-blockchain/truffle
$ truffle unbox pet-shop

Starting unbox...
==================

✔ Preparing to download box
✔ Downloading
```

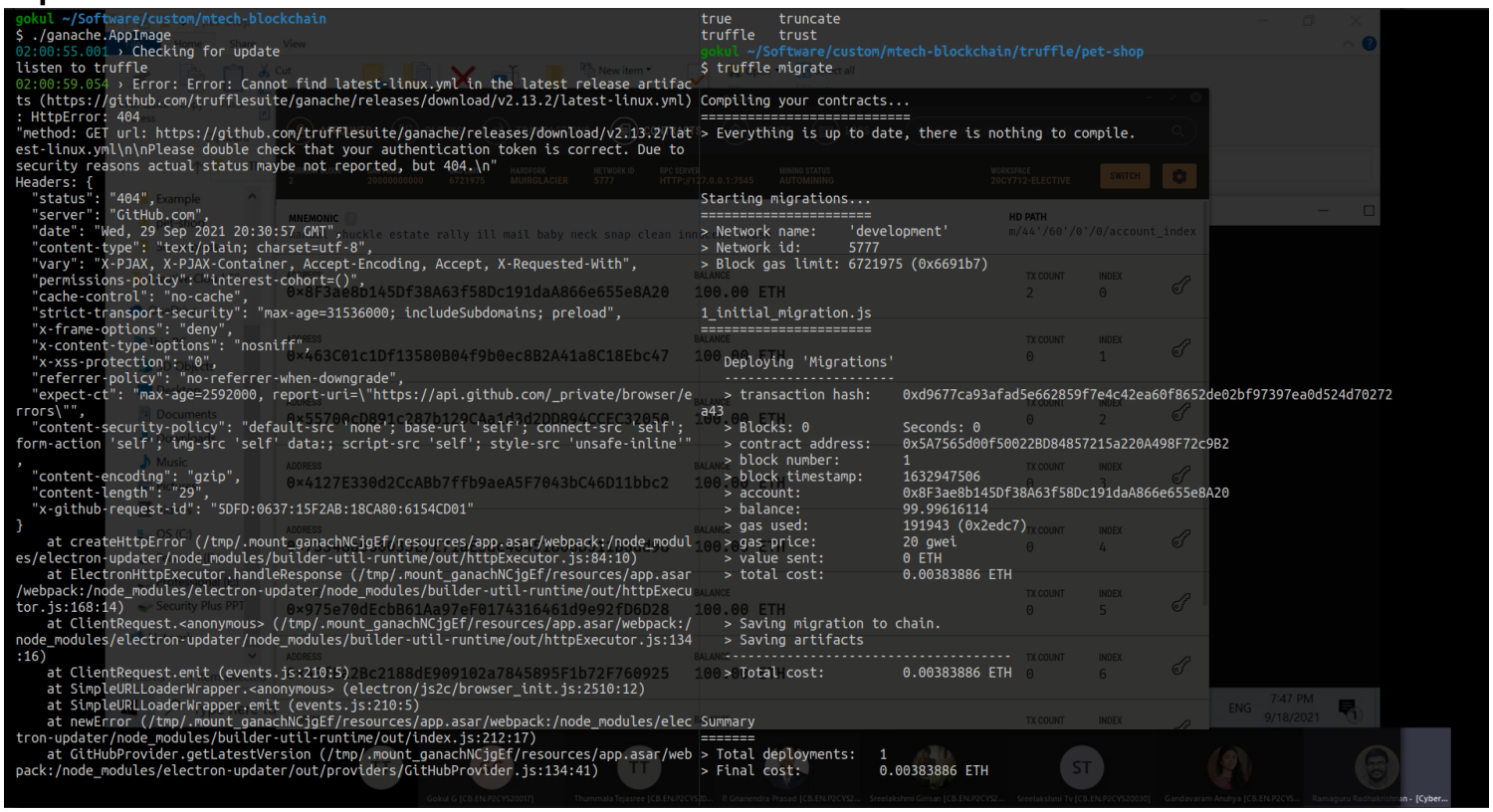
to complie all contracts within contract folder
truffle compile

```
$ truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Migrations.sol
> Artifacts written to /home/gokul/Software/custom/mtech-blockchain/truffle/pet-shop/build/contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang

gokul ~/Software/custom/mtech-blockchain/truffle/pet-shop
$
```

and compiled output is written to build folder into name-of-contract.json, as we had only migrations.sol, the compiled code is migrations.json

```
gokul ~/Software/custom/mtech-blockchain/truffle/pet-shop/build/contracts
$ pwd
/home/gokul/Software/custom/mtech-blockchain/truffle/pet-shop/build/contracts
gokul ~/Software/custom/mtech-blockchain/truffle/pet-shop/build/contracts
$ ls
Migrations.json
gokul ~/Software/custom/mtech-blockchain/truffle/pet-shop/build/contracts
$
```

now we can migrate/deploy the json file using truffle migrate, we should have ganache (mine is an appImage) open for this

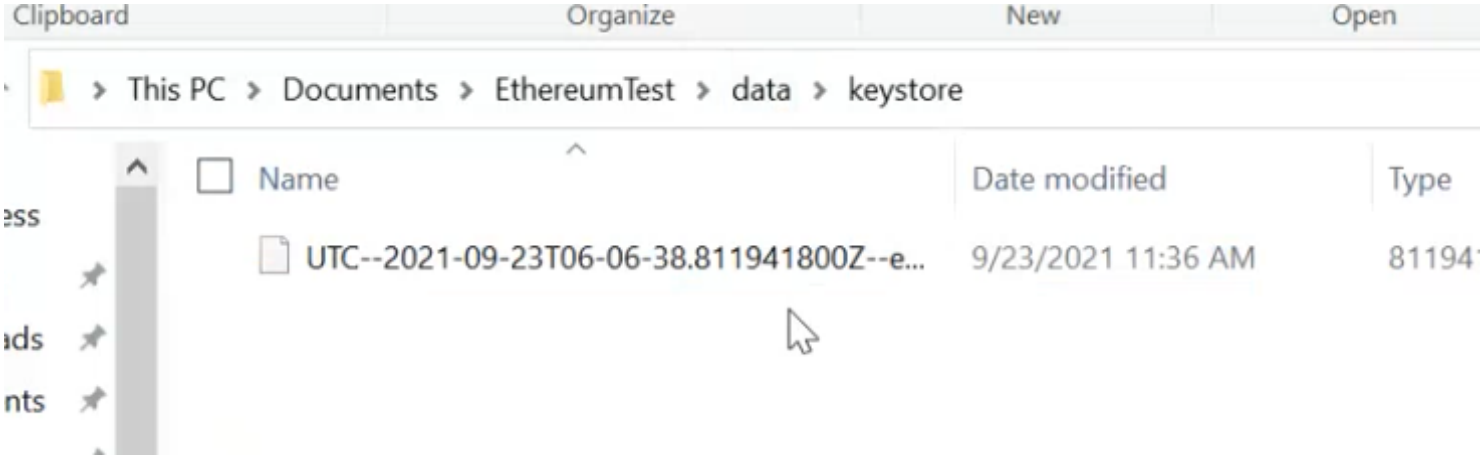


for unit testing
truffle test

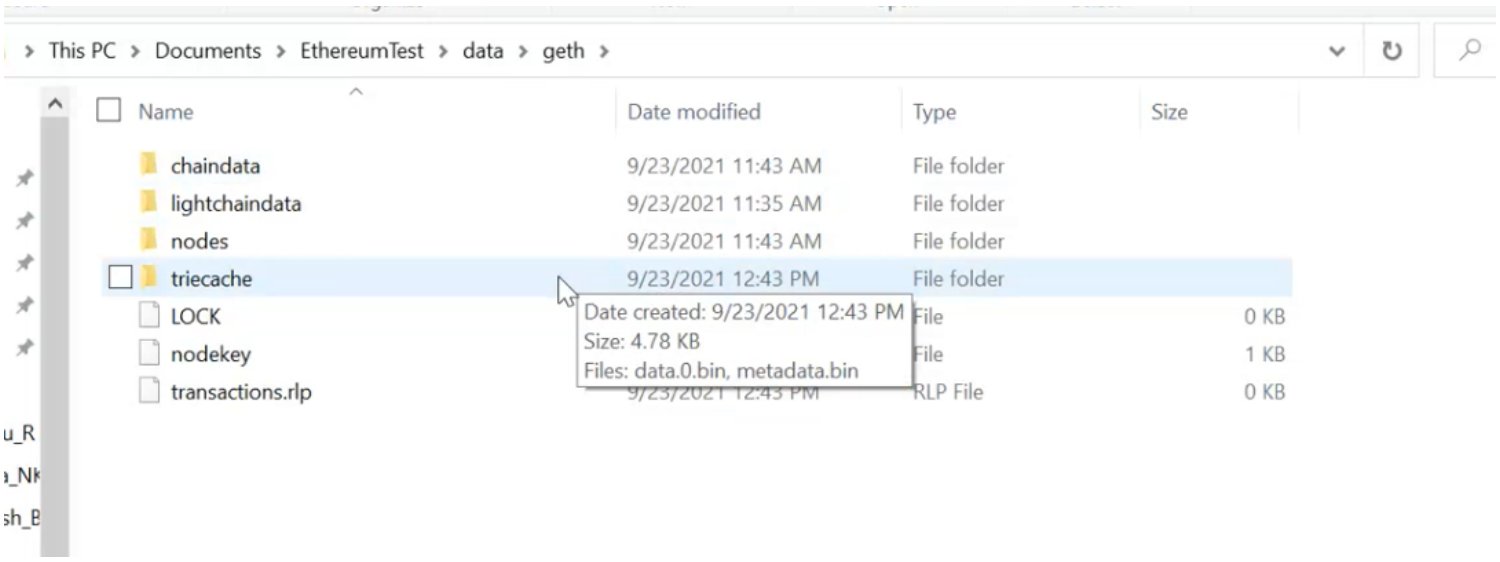
for deploying for testing locally:
npm install //for install all dependencies
npm run dev //running on local host

```
npm install -g truffle
truffle unbox
truffle init
truffle compile
truffle migrate (deploy)
truffle develop
truffle console
truffle debug
truffle test
truffle publish
```

lecture -13



keystore for account related data

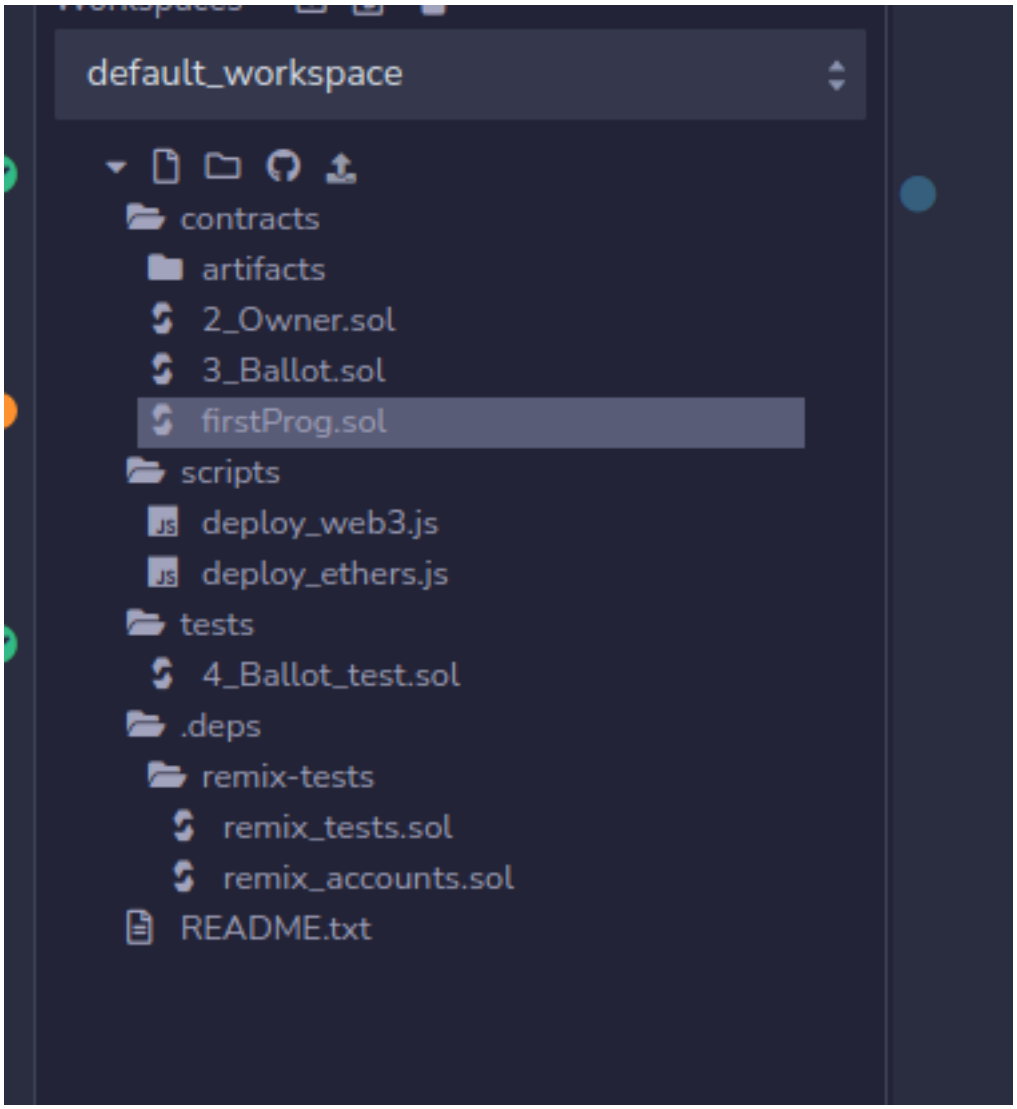


data folder for geth
<need to finish, not completed.>

lec-14 pending

lec-15

ganache: A private blockchain which has built in wallet and stuff
geth : A low level client to access ethereum blockchain
DApp: smart contract when it gets deployed on blockchain we call it DApp



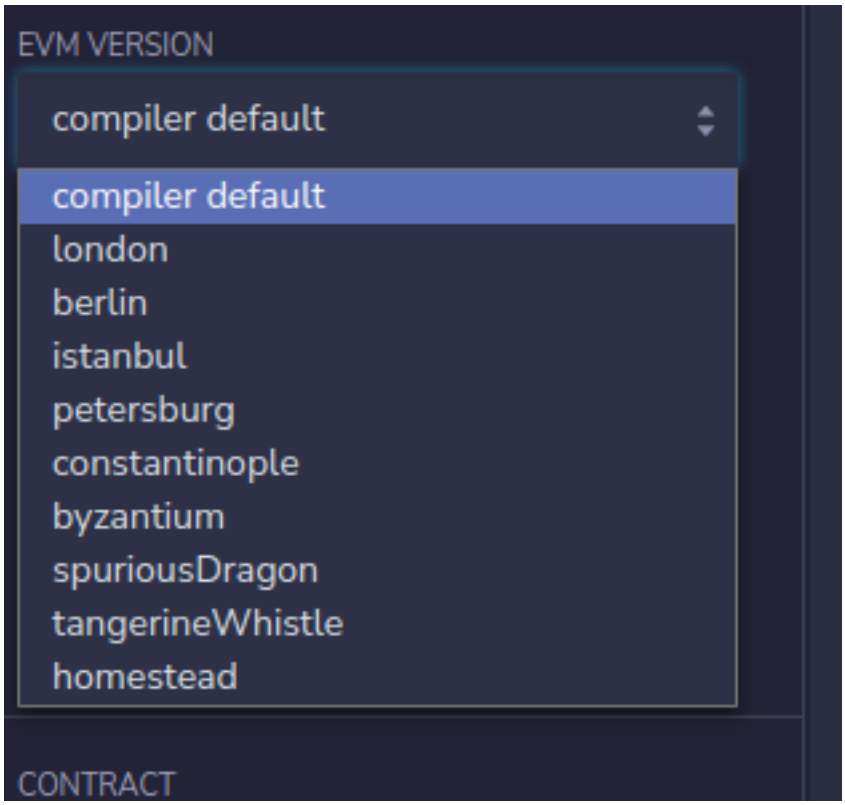
\

in remix ide, we always have three folders

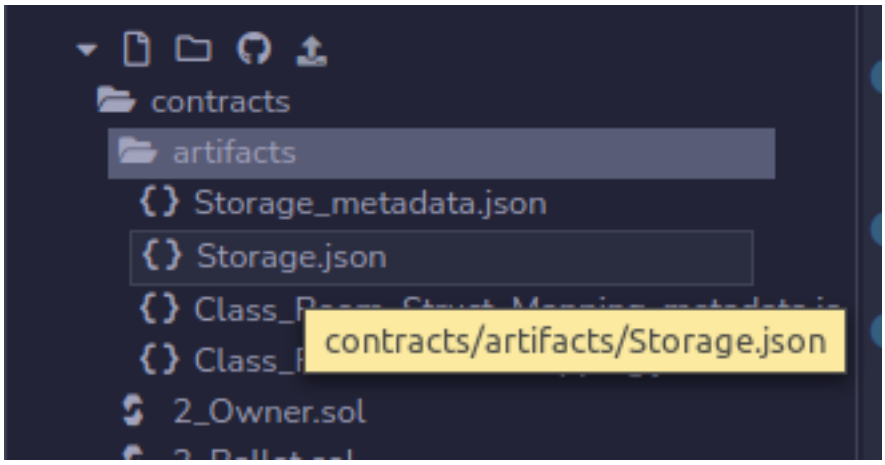
- contracts
  - ◇ write your SC here
- scripts
- tests

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

/**
 * @author Ramaguru Radhakrishnan
 * @title Class Room with Structure & Mapping
 * @dev Store & retrieve a student details
 */
```

pragma solidity tells what lang version should the choose, ie between 7 and 9 anything is fine



EVM machines:
homestead is oldest version
london is newest version
in truffle when we compile we have build folder
in remix we have artifact under contract folder



- byte code is basically the compiled output of solidity execution
  - ◇ byte code is machine independent
  - ◇ this is stored in blockchain as transaction
- ABI
  - ◇ this code is generated in *_metadata.json file under artifact folder

• in other programs, you have to compile, run test and debug then only can you deploy the program
• in solidity you have to deploy (in blockchain)first then you can run

how to write solidity programming:

this is a mandatory comment:
// SPDX-License-Identifier: GPL-3.0

pragma solidity to specify version

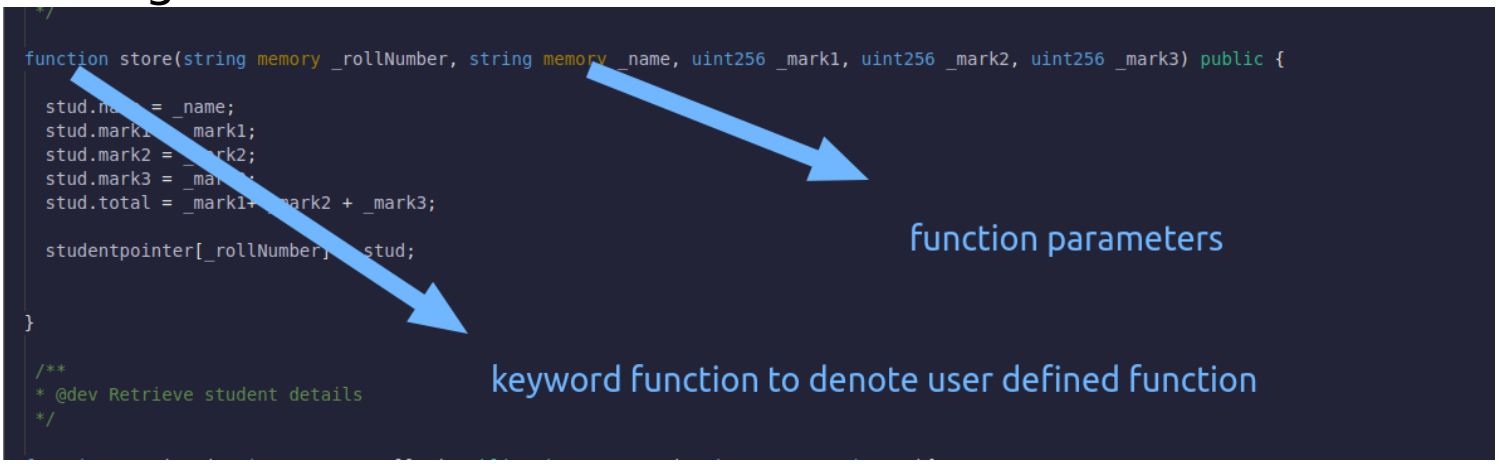
contract:
• equivalent of class in other OOPs languages
• this is the smart contract

all programs end with .sol

how to store variables in block chain?
• define a state variable in a contract and assign value to it
  ◇ assign value by putting direct equal to
    ■ string name = “gokul”
    ■ or write a dedicated function for it

```
 */
contract Class_Room_Struct_Mapping {

    mapping(string=>student) studentpointer;
    //Key - uint256 Roll No
    //  studentpointer[_rollNumber] = stud;


    struct student {
    //State Variable
    string name;
    uint256 mark1;
    uint256 mark2;
    uint256 mark3;
    uint256 total;

    }

    student stud;
```

writing functions

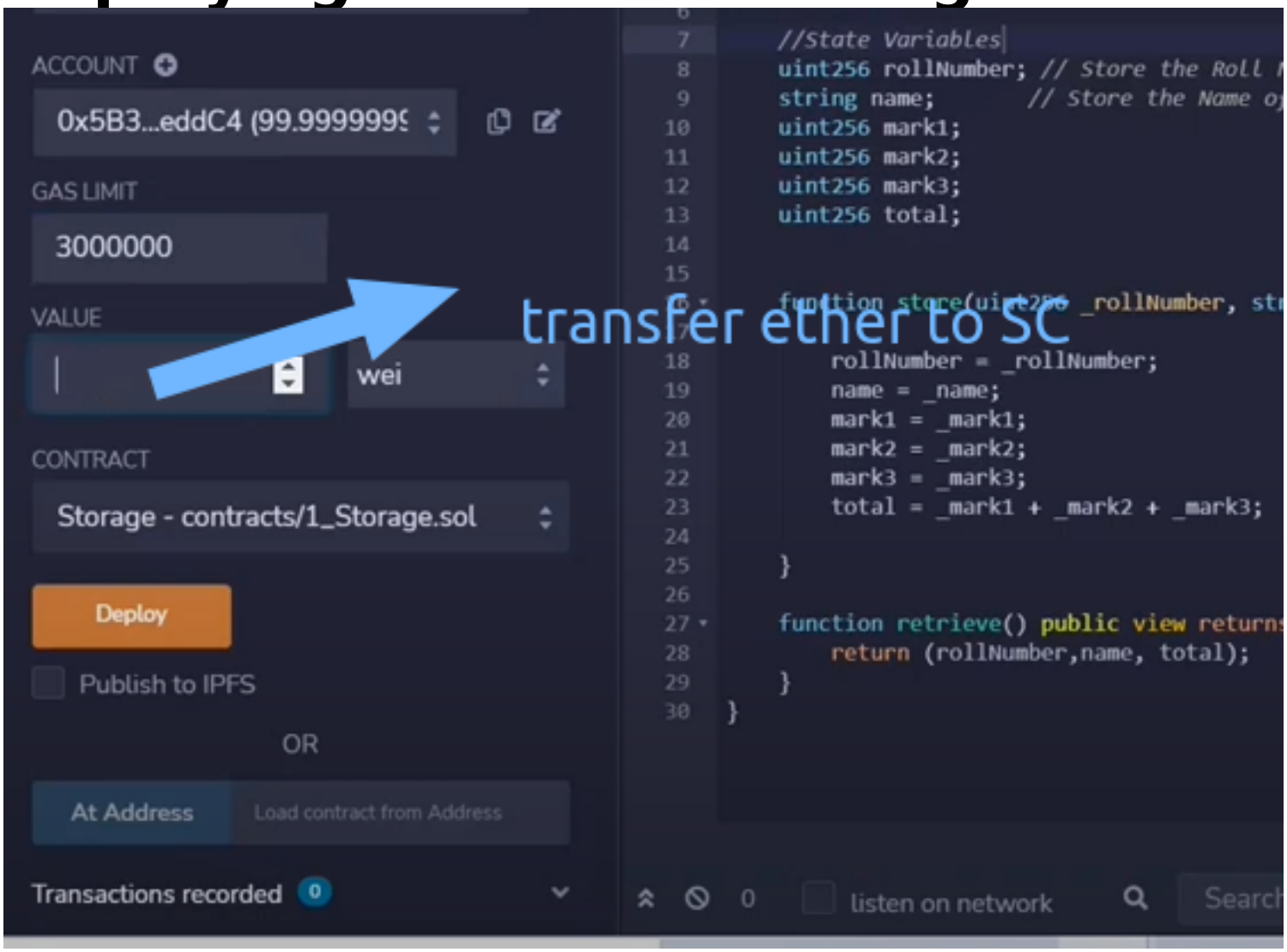



```
function retrieve(string memory _rollno) public view returns (string memory, uint256){
    student memory stud1 = studentpointer[_rollno];
    return (stud1.name, stud1.total);
}
```

view keyword is used so that end user can view the

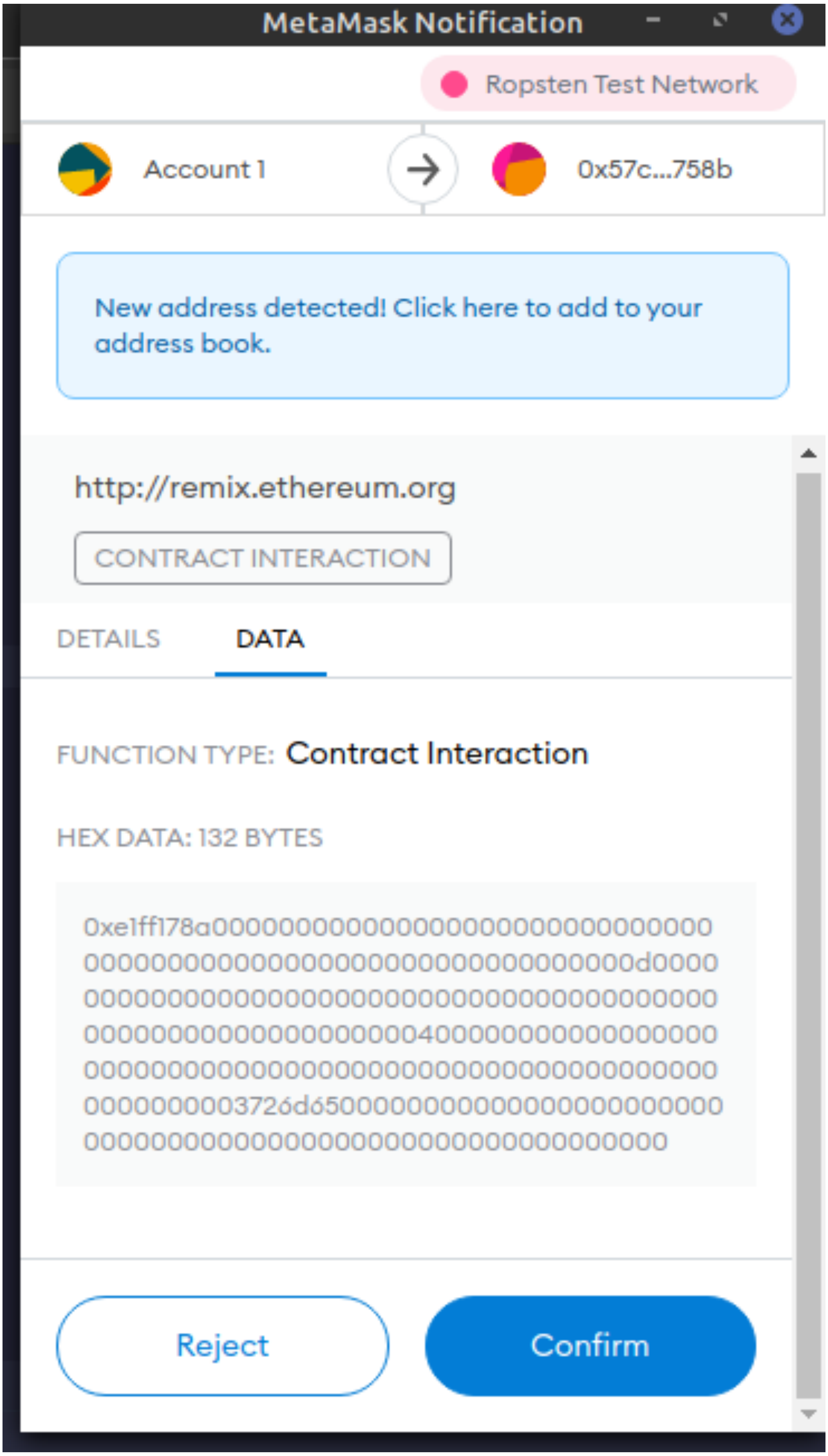
function result in Dapp
function maybe public ie can be interacted with
if u put private then nothing is visible
• other stuff we will see later are payable, internal and external
• keyword memory is used to mean vairables are only to be stored in program temp and not in blockchain
• you need not do that for integer datatypes

deploying in non EVM things



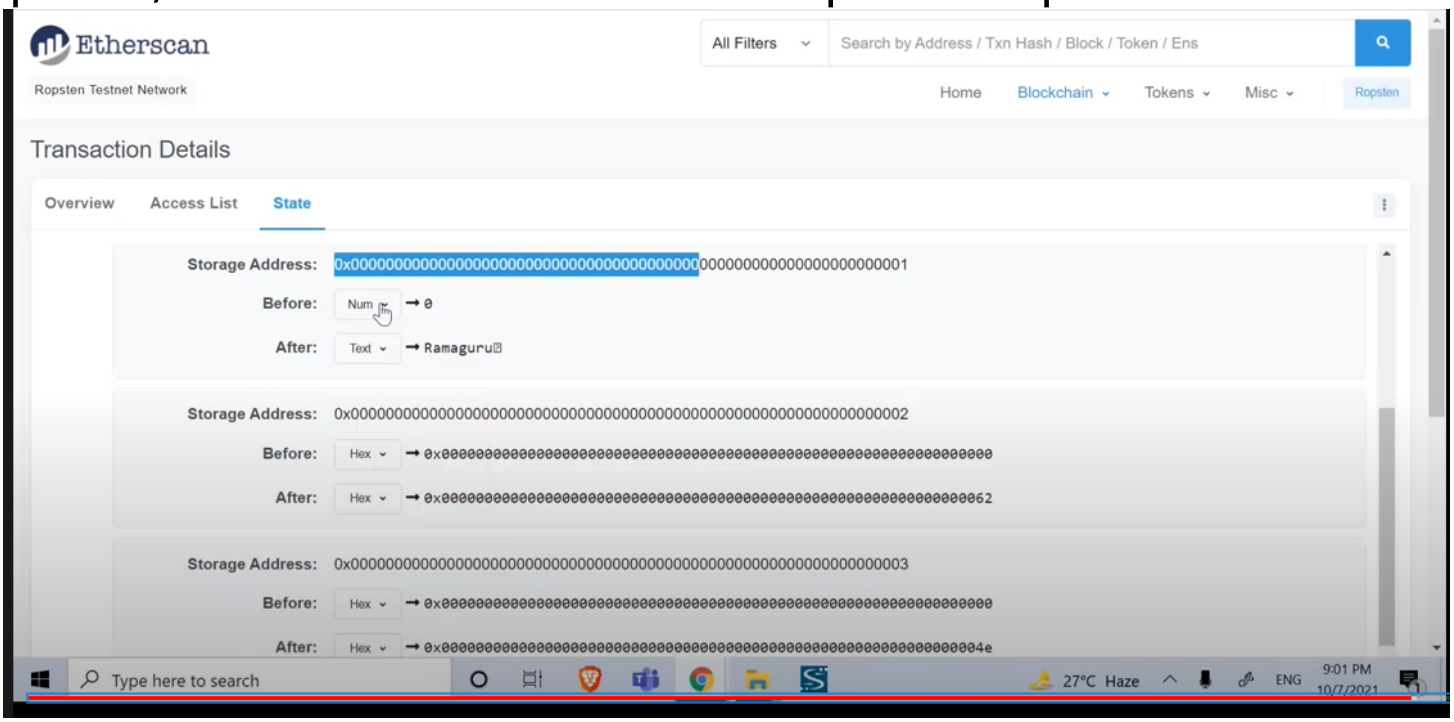
• java script vm is getting deployed in browser
• injected web3
  ◇ deploy it in testnet

- ▪ automatically your metamask will open
- ▪ when u deploy or interact with SC you have put to put in ether



- to store you are pefroming a consensus which requires money
  - ◇ all members verify the transaction

◇ and store it among themselves
• while retrieving you are retrieving it from one of the nodes so no money required
• since this is a sc deployed in blockchain for example ropsten, this can be seen in the ropsten explorer
•

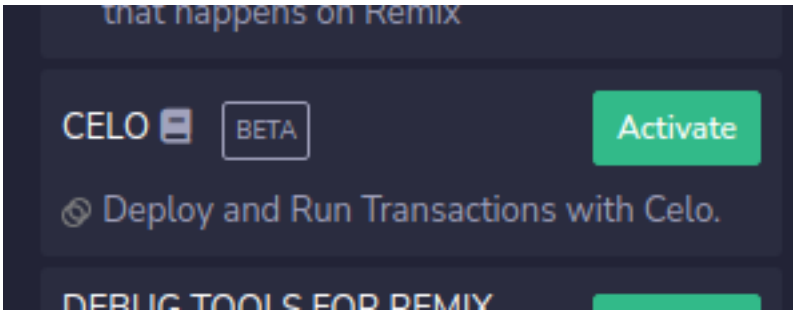


lec 16

ABI -> Application Programming Interface
• in order to talk to a SC deployed on blockchain you need ABI
• similiar to API in centralized web
• include this in ur front end and using this talk to smart contract

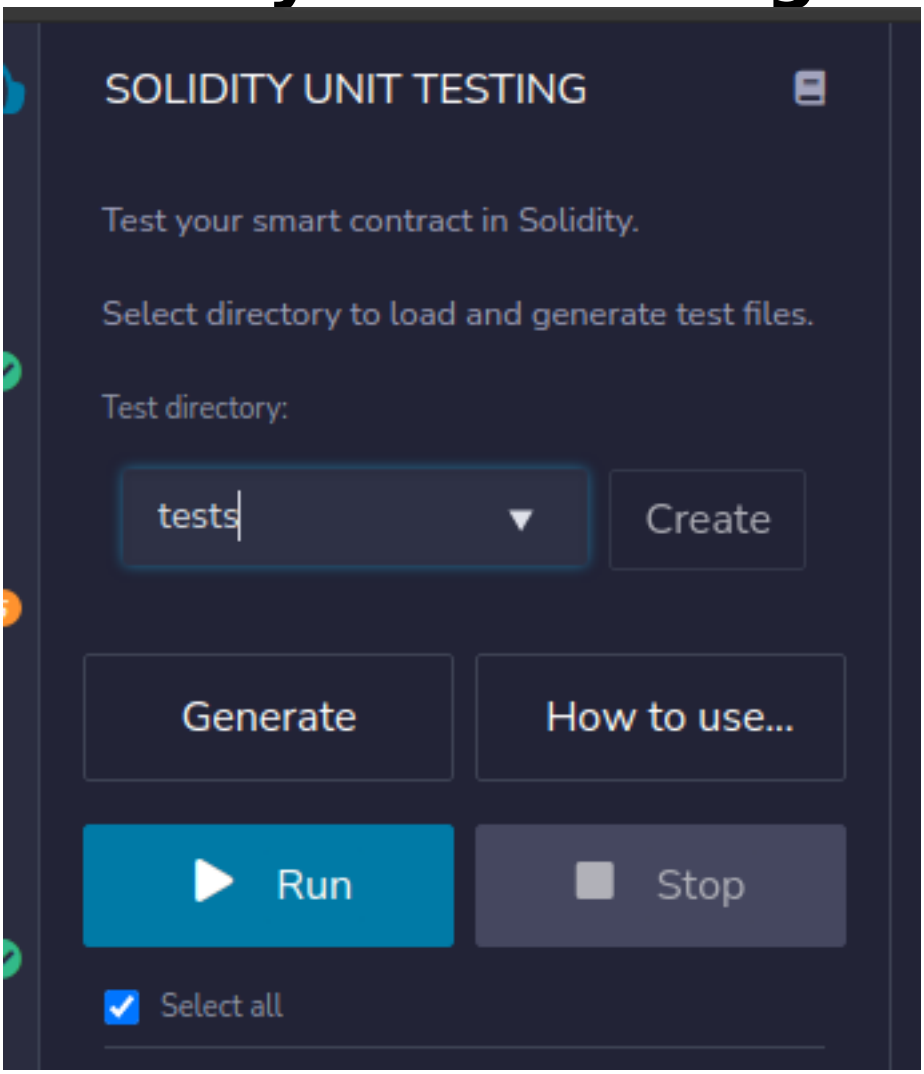
plugins

• celo
◇ hyperledger project
◇ write an SC and deploy on celo

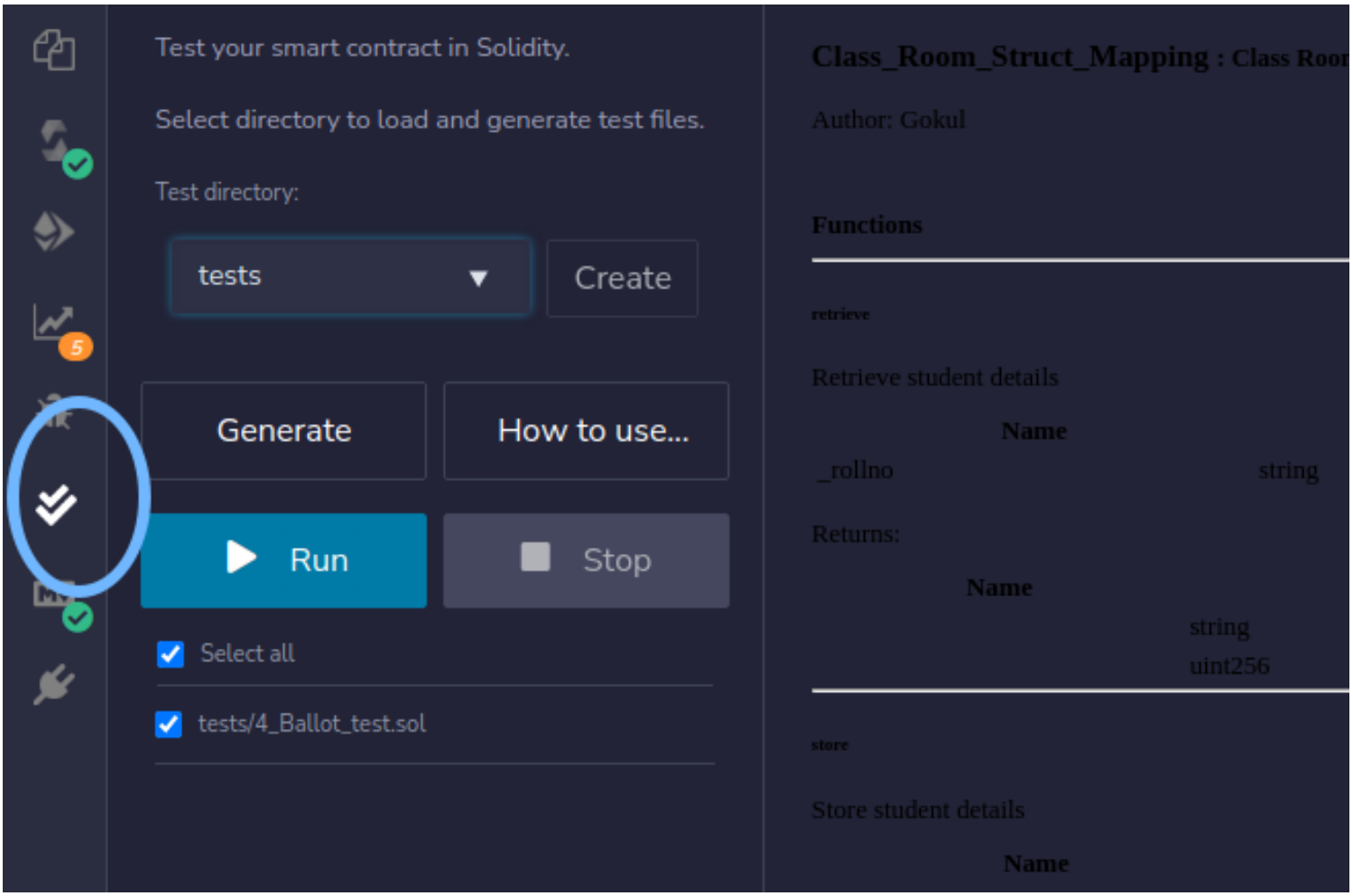


- ethdoc- documentation generator
  - ◇ generate documentation for your program
  - ◇ and publish to IPFS
- solidity to UML
  - ◇ generates UML class diagram for SC

Solidity unit testing

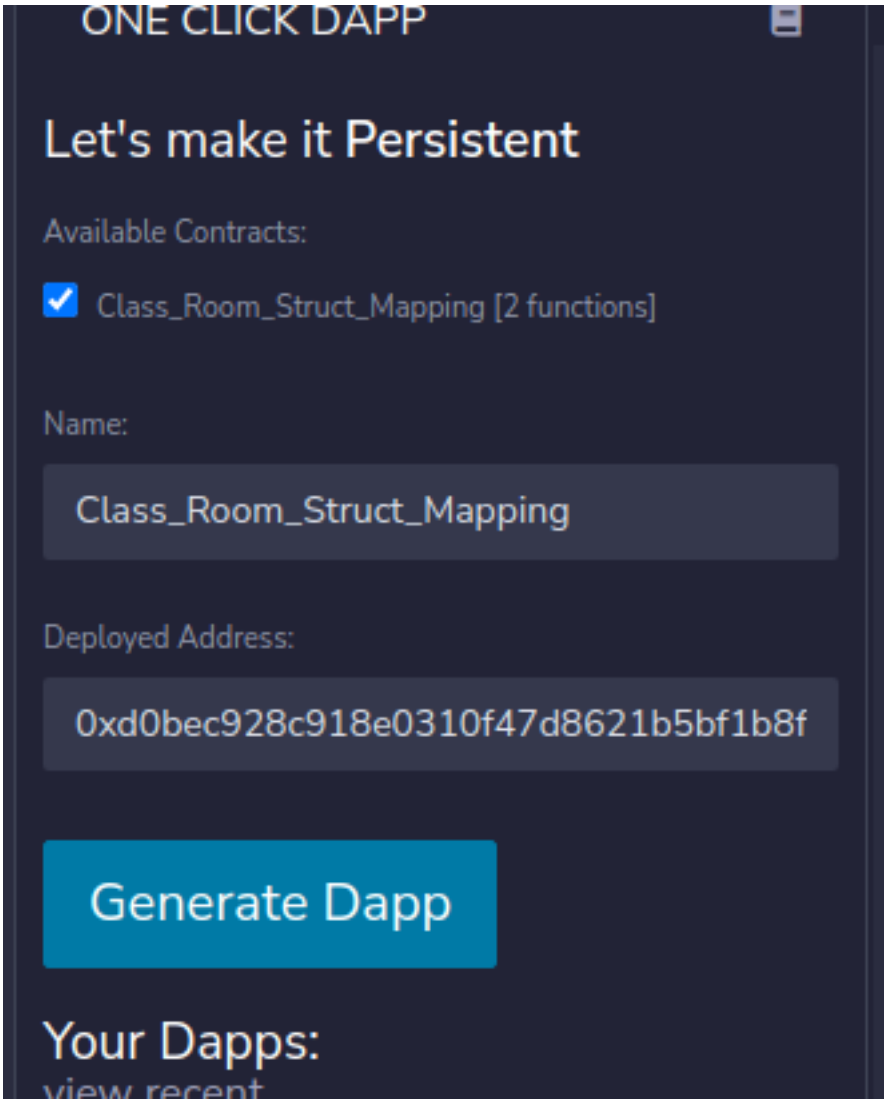


- you can run your test cases here by clicking run

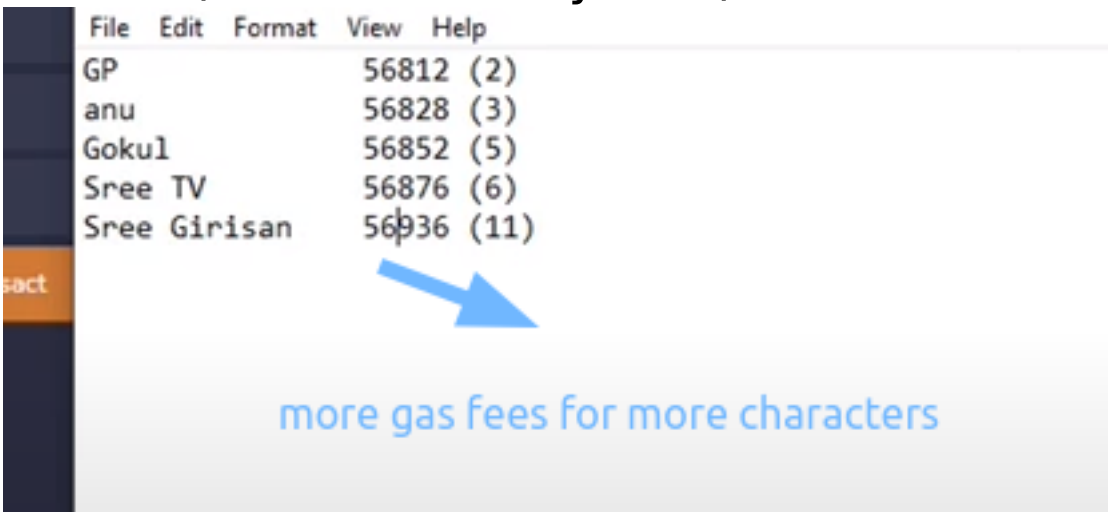


lec 17

- whenever you need to talk to a SC
  - ◇ you need smart contract address
  - ◇ ABI
- you can “publish” your code with minimal UI elements to IPFS using one click DApp
  - ◇ this is a plugin
  - ◇ deploy it on testnet/mainnet
  - ◇ copy the smart contract address, and paste

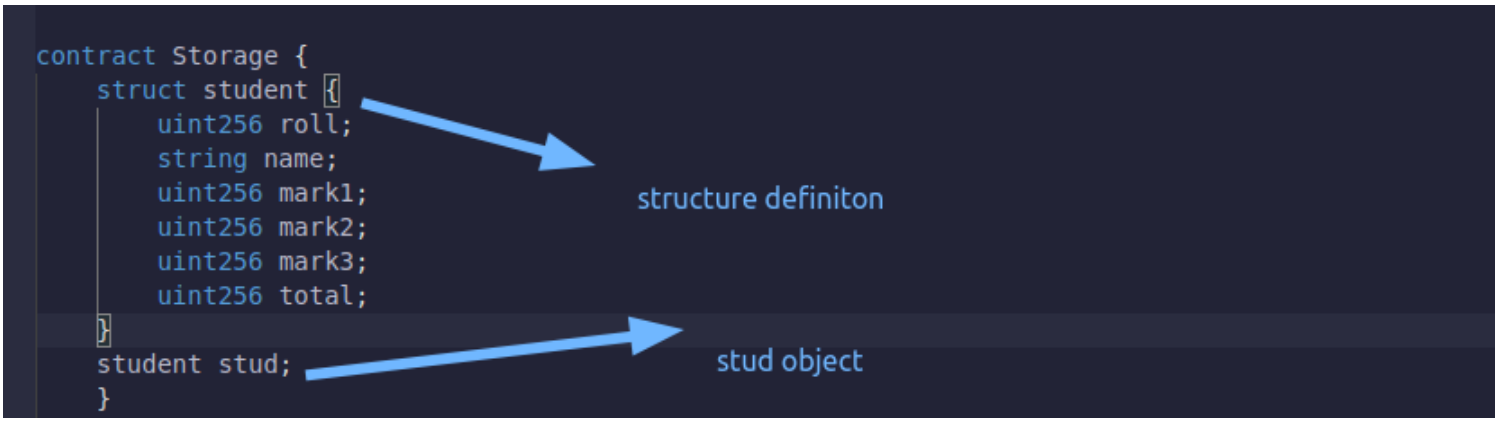


• you can see decompiled(non user friendly) or sometimes actual code(pulled from github) for SC in DEDAUB (contract-library.com)



•

• if u want to store and retrieve multiple objects of different data types use structure



we have to refer to arguments in get and store function with object name

• to reduce computational costs associated with storing and searching for items we use mapping

mapping (<primary key> => <value you want to reference>) variable_name;
eg: mapping (uint256 => student) studPointer; ie maps a integer to a object of type student
maps a integer data type to whole of student which in this case is a struct

to assign it
studPointer[_rollno] = studentObject;

to call

return(studPointer[12].name)

lec 18

IPFS
• Interplanetary file system

- it is a protocol
  - ◇ ipfs://
- aims to provide decentralized file storage and retrieval service across large distances

why IPFS:
- with current tech, web history cannot be preserved
  - ◇ centralized control coming from GAFAM companies, IETF, ICANN etc means censorship of certain sites
  - ◇ IPFS is decentralized, meaning content cannot be censored
- we use location based addressing today
  - ◇ http://webserver.mainfolder/subfolder1/subfolder2/.../-needed_file
  - ◇ to get the needed file u need to known the whole path
  - ◇ this also allows for file duplication
- current web is central backbone based
  - ◇ with ipfs you can have connectivity in a distributed manner
- IPFS uses content based addressing
  - ◇ you tell the hash of the file, IPFS will retrieve the content matching that hash
  - ◇ if you change the file, when you upload to IPFS, the hash will also change
  - ◇ you cannot upload the same file more than once
    - ■ any duplicate request to upload the same file will allow user to link to that original master copy in IPFS
    - ■ but wont create any duplicates
- you cannot store large files in blockchain
  - ◇ you can use IPFS to store huge files in a distributed way immutably

◇ you cannot use blockchain and non-IPFS based distributed systems because

■ stuff like torrents use location based addressing and therefore

- subject to change location periodically thus immutable property of blockchain is preventing correct linking of blockchain to file

- people can take the same file name and put it another location of non IPFS file system

→ violates integrity of file

◇ just put the file hash in blockchain while the file itself in IPFS

◇ the hash is content ID or CID

◇ CID v0: starts with Qm

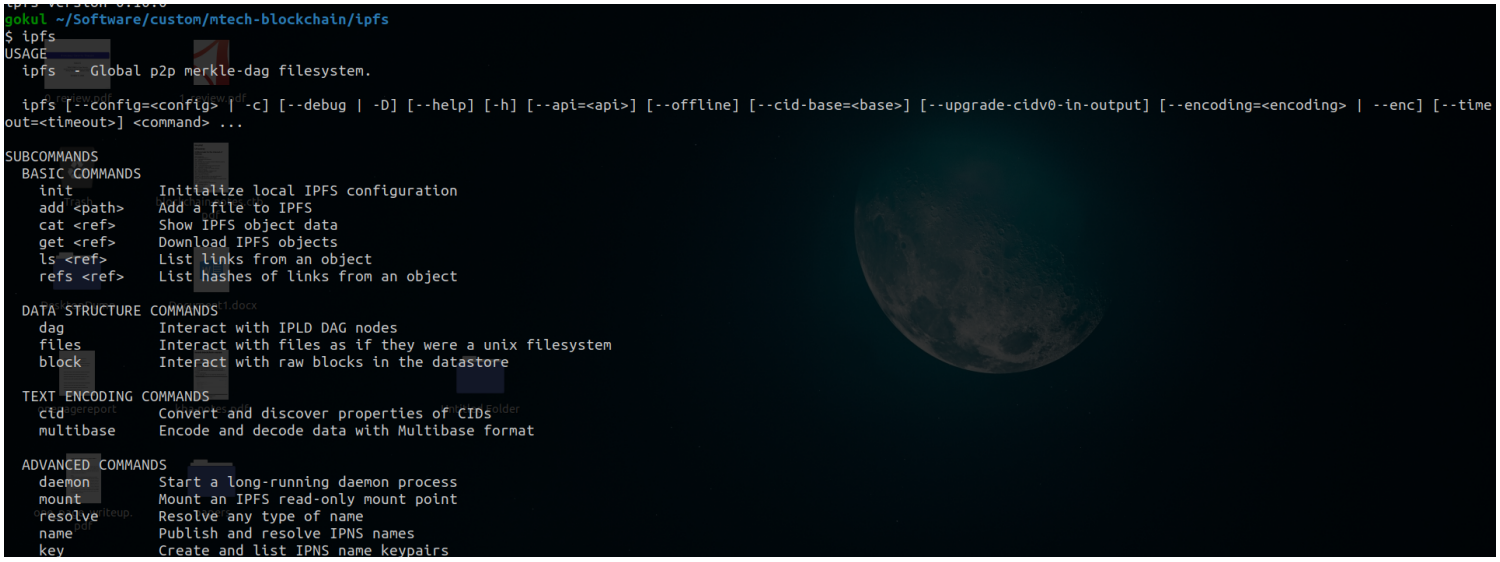
◇ CID v1: starts with ba

• entire file is divided into small chunks and scattered across nodes

■ use Directed Acyclic Graph (DAG) to tell which chunk is where

ipfs command line version

• based on go

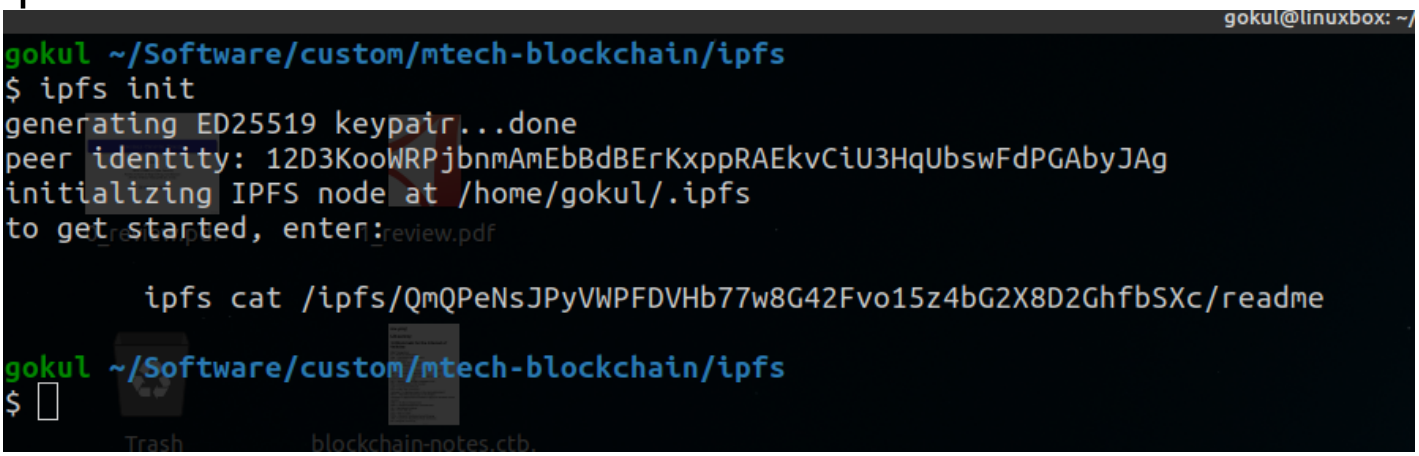


advantages:

- you can host your own website in IPFS
  - ◇ D.tube
    - ■ all videos on this site and coming from IPFS
  - ◇ ipfs-search.com
    - ■ search engine for ipfs

- each ipfs node is identified by node id
  - ◇ hash value computed from some parameters of your local system used as unique identifier
  - ◇ to create a node id

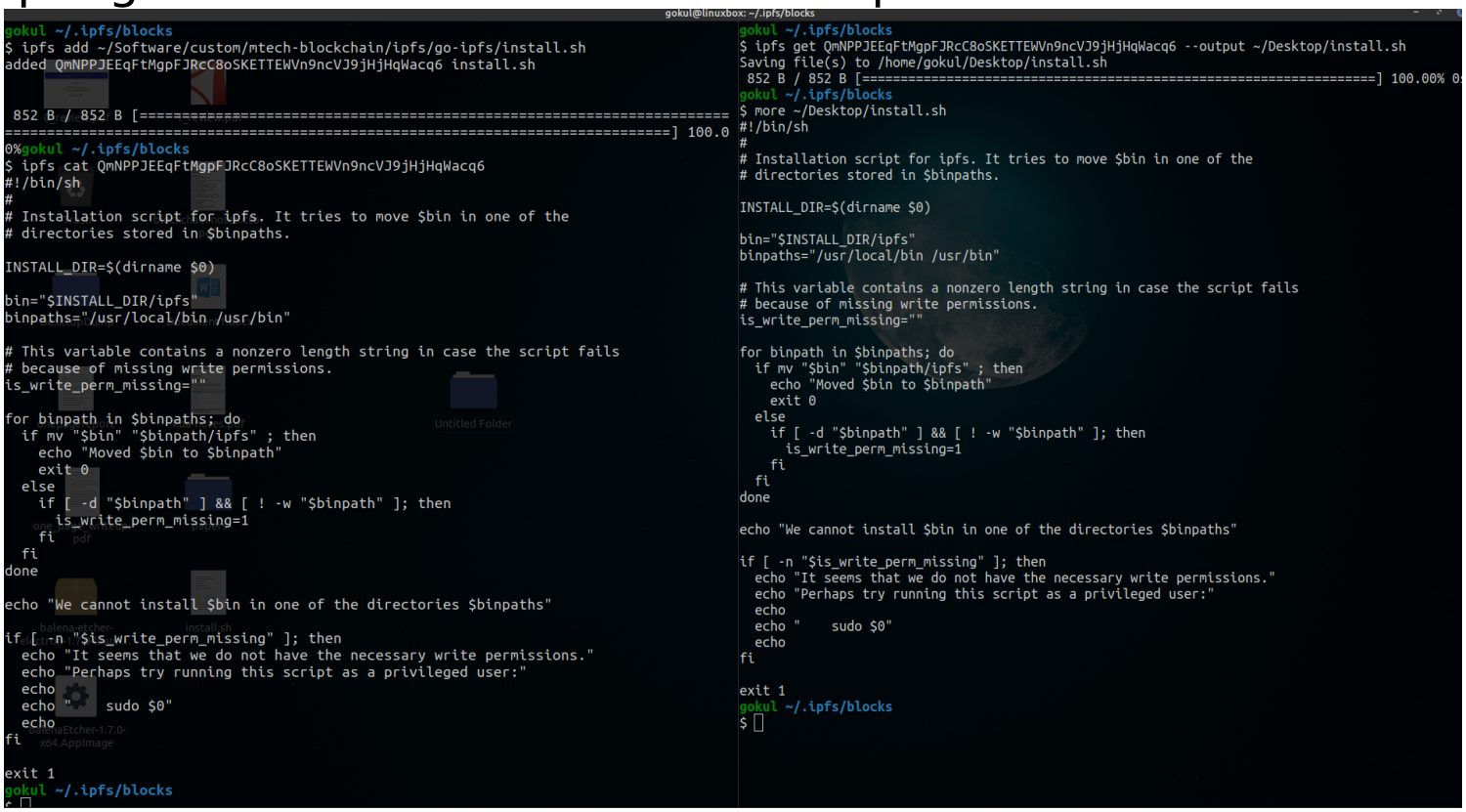
    ipfs init



  - ◇ if you do it again and again it would say ur node id is

already created and would not let u proceed

ipfs add: adds a file to IPFS and returns the hash
ipfs cat: print the content on screen if u provide the hash
ipfs get: download the content if u provide the hash



if you dont specify output flag, then in get command file name is the hash of the file generated

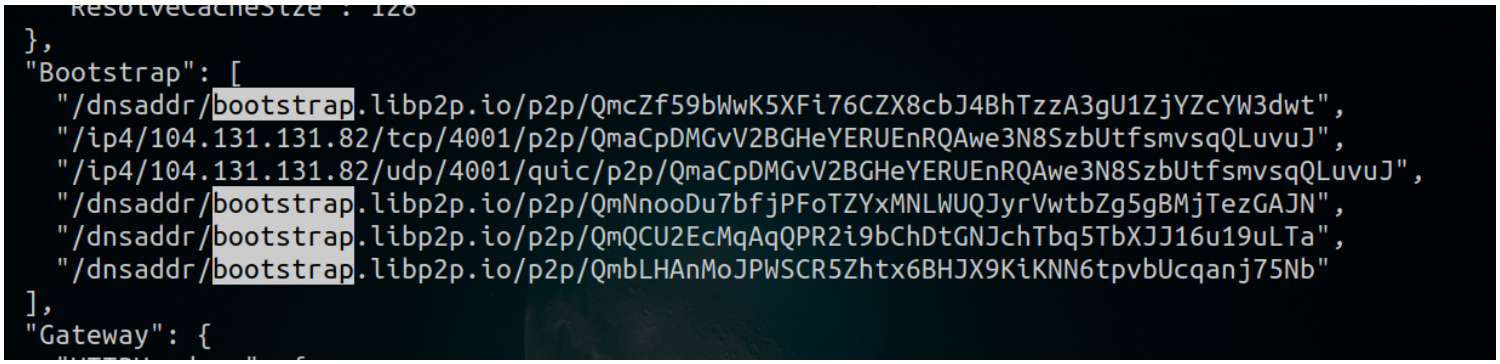
```
gokul ~/Desktop
$ ipfs add --only-hash Document1.docx
added QmcQmstsqtccdJNthYBYTU87QCGep5juzquJrTNLiqMWcx Document1.docx
 21.60 KiB / 21.60 KiB [============================================================] 100.00%
gokul ~/Desktop
$ ipfs get QmcQmstsqtccdJNthYBYTU87QCGep5juzquJrTNLiqMWcx
Error: merkledag: not found
gokul ~/Desktop
$
```

- ipfs add --only-hash conducts a dry run

ipfs config file

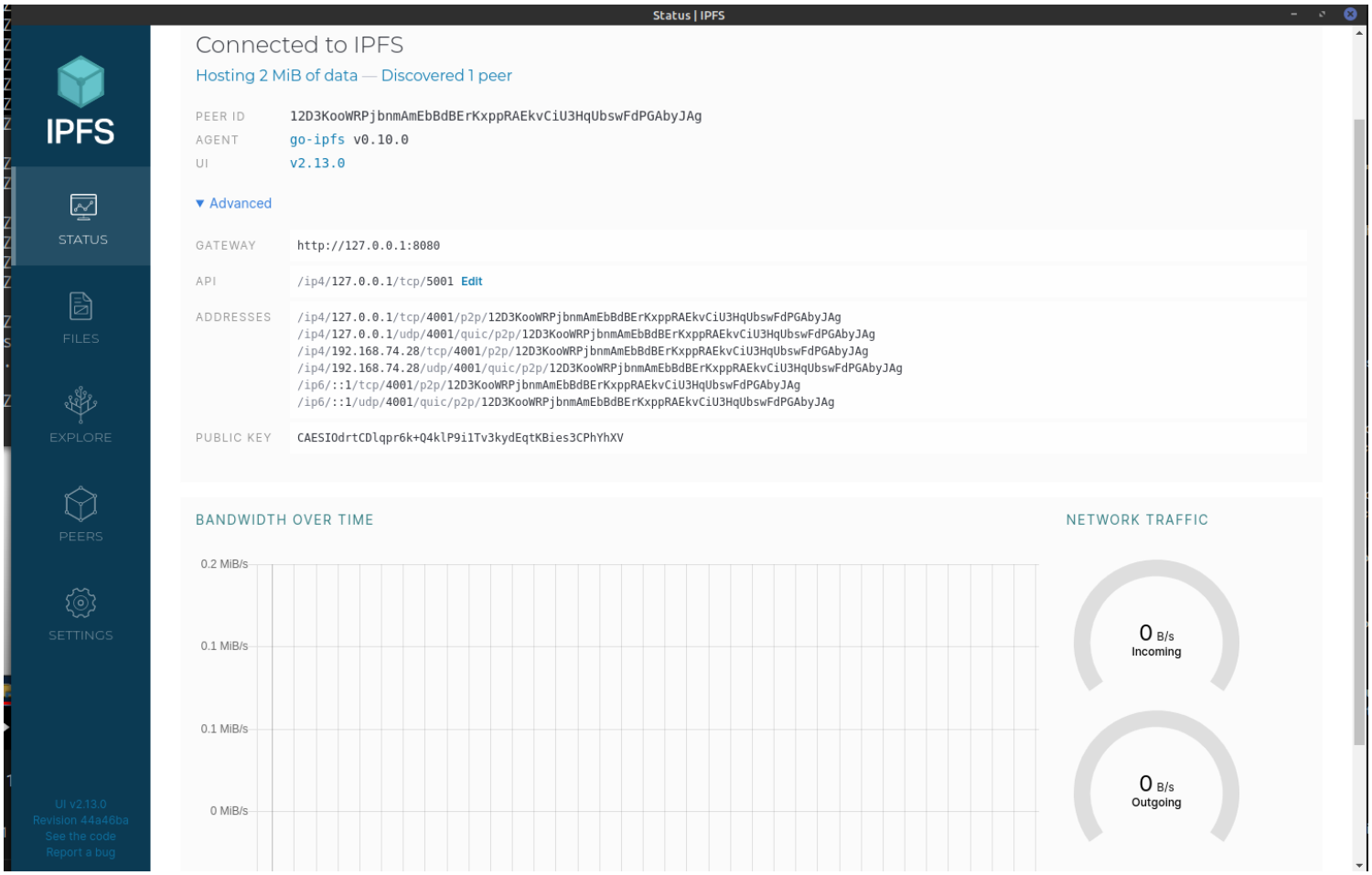
```
gokul ~/.ipfs
$ head config
{
  "Identity": {
    "PeerID": "12D3KooWRPjbnmAmEbBdBErKxppRAEkvCiU3HqUbswFdPGAbyJAg",
    "PrivKey": "CAESQEJC8C8uZj4K8LrYS2m55ixDaCCbEY7vHeULjpmazL/q52u0IOWqm
vqT5DiSU/2LVO/eTJ0Sq0oGJ6zcI+FiFdU="
  },
  "Datastore": {
    "StorageMax": "10GB",
    "StorageGCWatermark": 90,
    "GCPeriod": "1h",
    "Spec": {
gokul ~/.ipfs
$
```

```
gokul@linuxbox: ~/.ipfs/blocks
{
  "Identity": {
    "PeerID": "12D3KooWRPjbnmAmEbBdBErKxppRAEkvCiU3HqUbswFdPGAbyJAg",
    "PrivKey":
  },
  "Datastore": {
    "StorageMax": "10GB",
    "StorageGCWatermark": 90,
    "GCPeriod": "1h",
    "Spec": {
      "mounts": [
        {
          "child": {
            "path": "blocks",
            "shardFunc": "/repo/flatfs/shard/v1/next-to-last/2",
```



• bootstrap nodes allow my node to xonnect to rest of ipfs when you run IPFS deamon command
  ◇ you can actively share around file chunks now

ipfs pin add <hashvalue>
• by default the file you added to IPFS will be GCed(Garbage collected)
• if you pin the file it will stay there for ever



ipfs desktop app

here the pdf will be divided into chunks and each chunk is send to one of the nodes. Over time the nodes will get a lot of chunks, and will not know what data it got from where.
if we want to recover a file from IPFS, we give CID using which DAG algorithm collects all chunks and passes it to the intended recipient.

lec 18