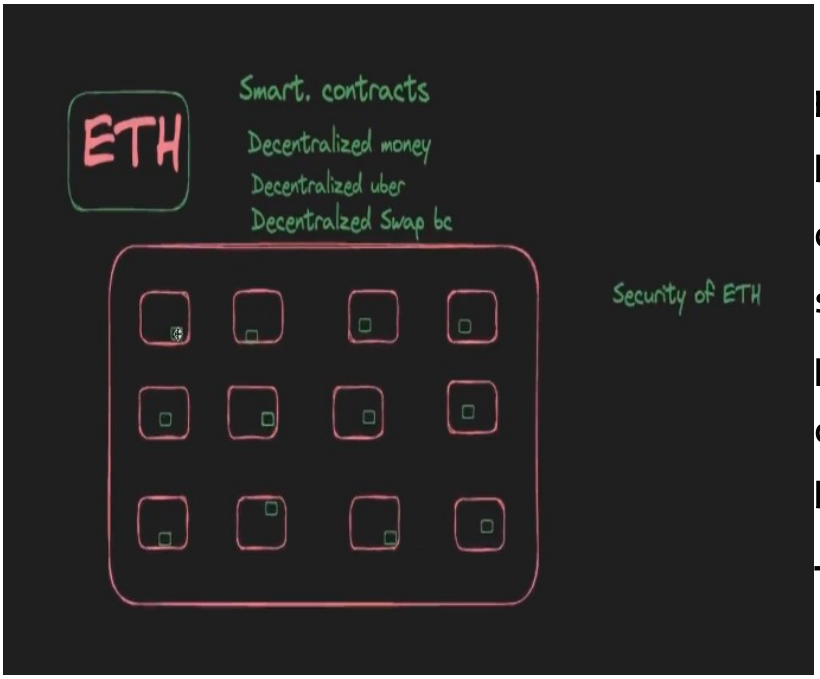
Solana is the one which introduced the smart contracts

Context??

**Why eth and sol are better than solana?**

When bitcoin came it just solved one usecase which was the decentralized currency during that time only many of the people started creating there own bitcoin eg for lending protocol = lending and borrowing bitcoin, decentrailized uber etc…

due to this some miners would come but not enough as bitcoin which lead to **cold startup problem** which means like less people less secure to fix that ETH came in and told you want to make these things come to ETH and make a smart contracts and make all your things and there will be enough miners to secure the transaction also everyone happy good.




**Programs/Smart contracts**

ETH was one of the first blockchains to introduce the concept of decentralized state / programs. These are popularly known as smart contracts on the ETH blockchain.

**THE COUNT VARIABLE IS A STATE**

- Here is a simple ETH smart contract

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract Counter {
    uint public count;

    // Constructor to initialize count
```

```
    constructor() {
        count = 0;
    }

    // Function to increment the count
    function increment() public {
        count += 1;
    }

    // Function to decrement the count
    function decrement() public {
        require(count > 0, "Count cannot be negative");
        count -= 1;
    }

    // Function to get the current count
    function getCount() public view returns (uint) {
        return count;
    }
}
```

- Here is a simple Node.js HTTP server that does something similar

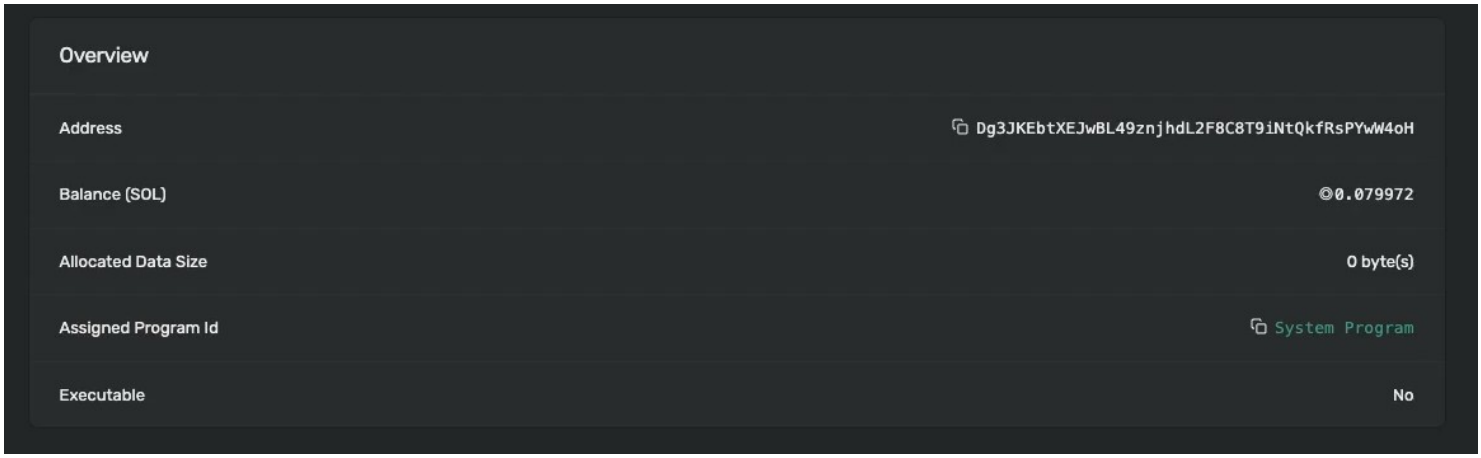
```
const express = require('express');
const app = express();
const port = 3000;
// Middleware to parse JSON bodies
app.use(express.json());
// Initialize count
let count = 0;
// Route to increment the count
app.post('/increment', (req, res) => {
  count += 1;
  res.json({ count });
});
// Route to decrement the count
app.post('/decrement', (req, res) => {
  if (count > 0) {
    count -= 1;
    res.json({ count });
  } else {
    res.status(400).json({ error: 'Count cannot be negative' });
  }
});
// Route to get the current count
app.get('/count', (req, res) => {
  res.json({ count });
});
// Start the server
app.listen(port, () => {
  console.log(`Server running at <http://localhost>:${port}`);
});
```

HTTP Servers are deployed on cloud providers like GCP, Azure

Smart contracts/programs are deployed on the blockchain

The way solana programs work is significantly different from other blockchains. Lets understand how.

## ACCOUNTS ON SOLANA




what all things a solana account can do:
can store data , can store code see the executable line
**1 SOL = 10^9 lamports**

## Accounts

On the Solana blockchain, an "account" is a fundamental data structure used to store various types of information.

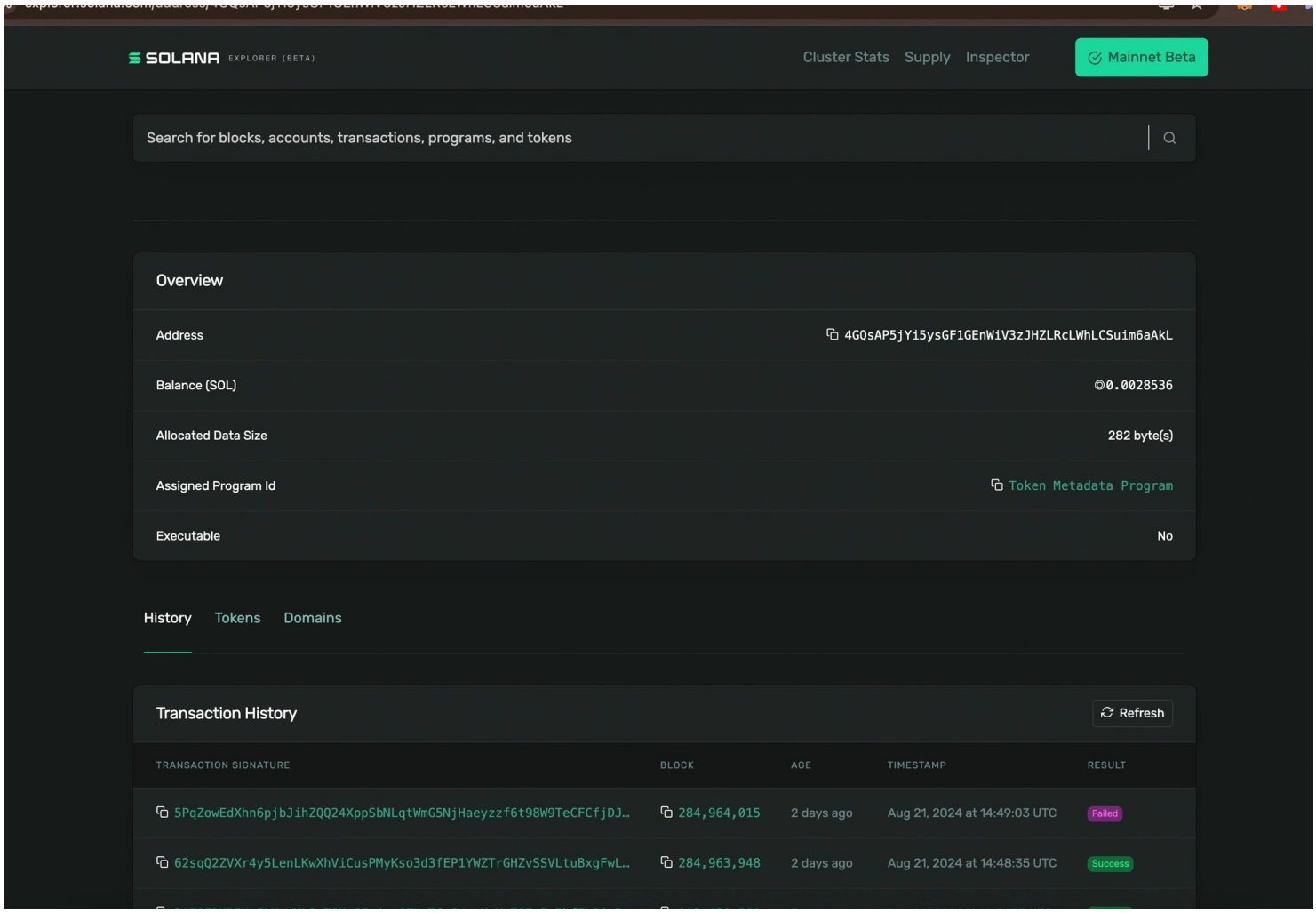
1. **Data Storage**: Accounts on Solana are used to store data required by programs (smart contracts) or to maintain state
2. **Lamports**: Accounts hold a balance of Solana's native cryptocurrency, lamports. Lamports are used to pay for transaction fees and to rent the space that the account occupies on the blockchain.
3. **Programs:** On Solana, programs are special accounts that contain executable code. These accounts are

distinct from regular data accounts in that they are designed to be executed by the blockchain when triggered by a transaction.

## TYPES OF ACCOUNTS IN SOLANA

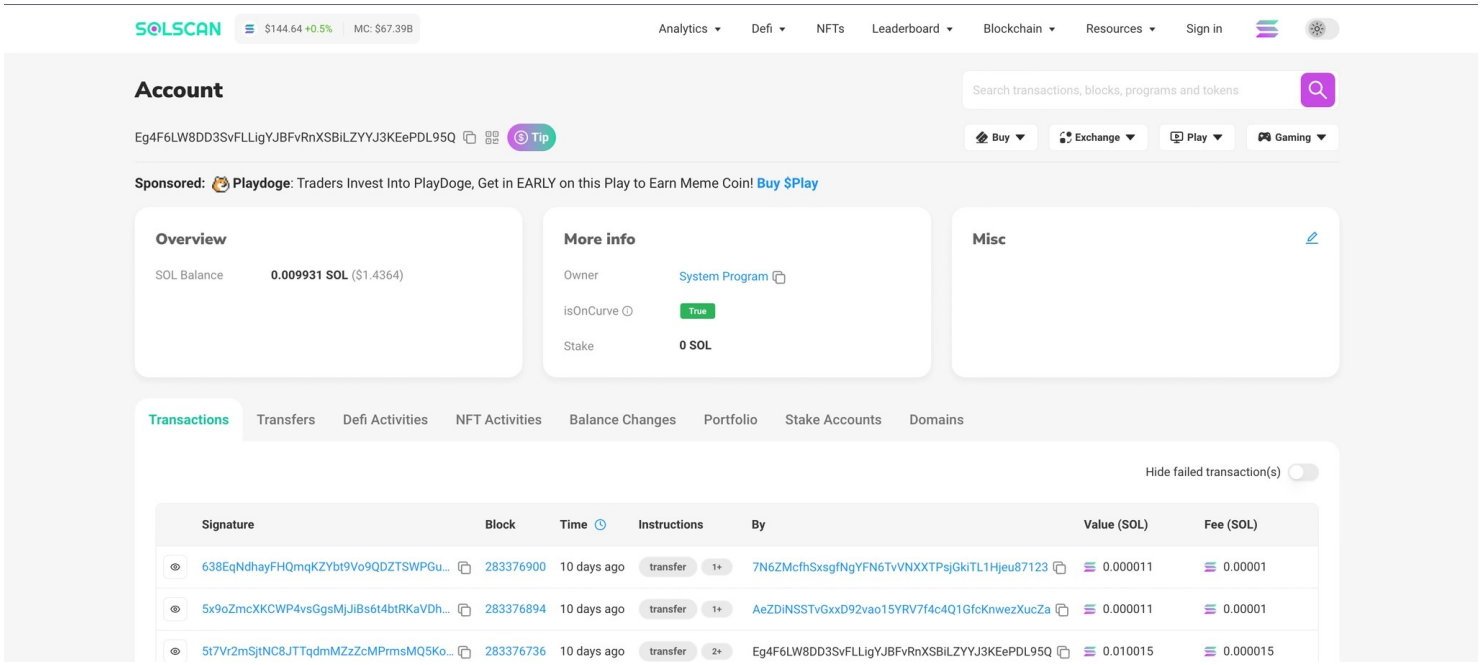
**Account with data and lamports but no data -**

https://explorer.solana.com/address/4GQsAP5jYi5ysGF1GEnWiV3zJHZLRcLWhLCSuim6aAkL



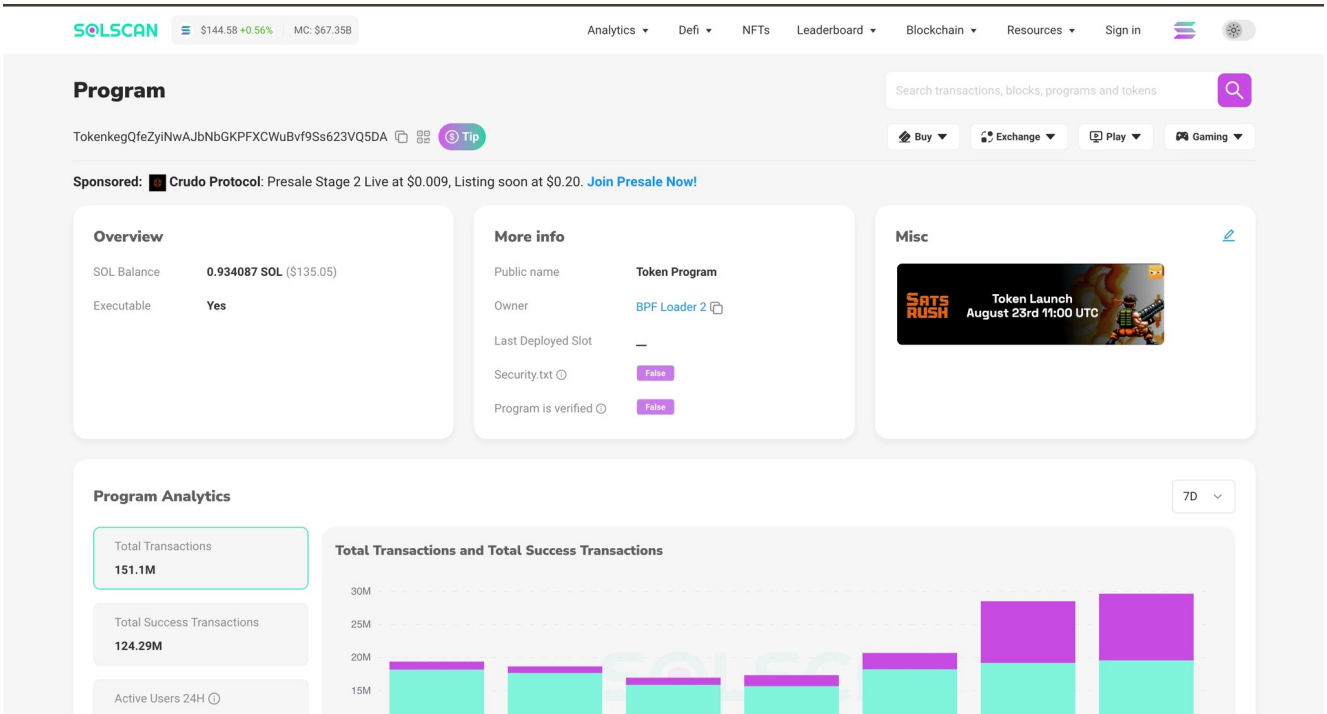

**Account with lamports but no data -

https://solscan.io/account/Eg4F6LW8DD3SvFLLigYJBFvRnXSBiLZYYJ3KEePDL95Q**




# Program

https://solscan.io/account/TokenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA




## What is rent and on solana and how to calculate it?

The code we have has to be stored on the blockchain which has some bytes of data size and to store that data we have

to maintain a minimum balance of money which is called as **rent exemption amount(this is the min amount)**.
Rent is given when you don’t have minimum balance just like a actual bank account if you maintain the minimum balance then no issue if you don’t maintain then pay the rent.
Rent is a mechanism on the Solana blockchain that ensures efficient usage of the blockchain's resources. It requires accounts to maintain a minimum balance proportional to the amount of data they store on the network. Accounts that fail to maintain this minimum balance will be removed from the ledger to free up storage. You can think of it as similar to a bank account. For many accounts, the bank will charge you a fee if you do not maintain a minimum balance. If your balance falls below the required minimum, the bank may charge you a fee or close your account.
Solana rent is like the minimum balance requirement, ensuring that accounts on the network have enough lamports (a lamport is one one-billionth of a SOL) to cover network storage costs. If an account's balance falls below the rent-exempt threshold, the account may be removed from the network. Rent is refundable. If an account is closed (removed from the network), the data is deleted from the chain and the rent is refunded to the account owner (or other defined account).
If you are keeping the minimum balance required for the data size you will be exempted from the rent otherwise money will be deducted.

If you don’t have the enough money then to clear the space the account will be deleted.

## SOLANA CLI


1. Download this for windows from github solana-release-x86_64-pc-windows-msvc.tar.bz2
2. you can try many commands and the private key is present in PS C:\Users\Admin\.config\solana\id.json
3. the private key pair can be taken out by going to the directory and using cat id.json.
4. this private key can be used in phantom and backpack also

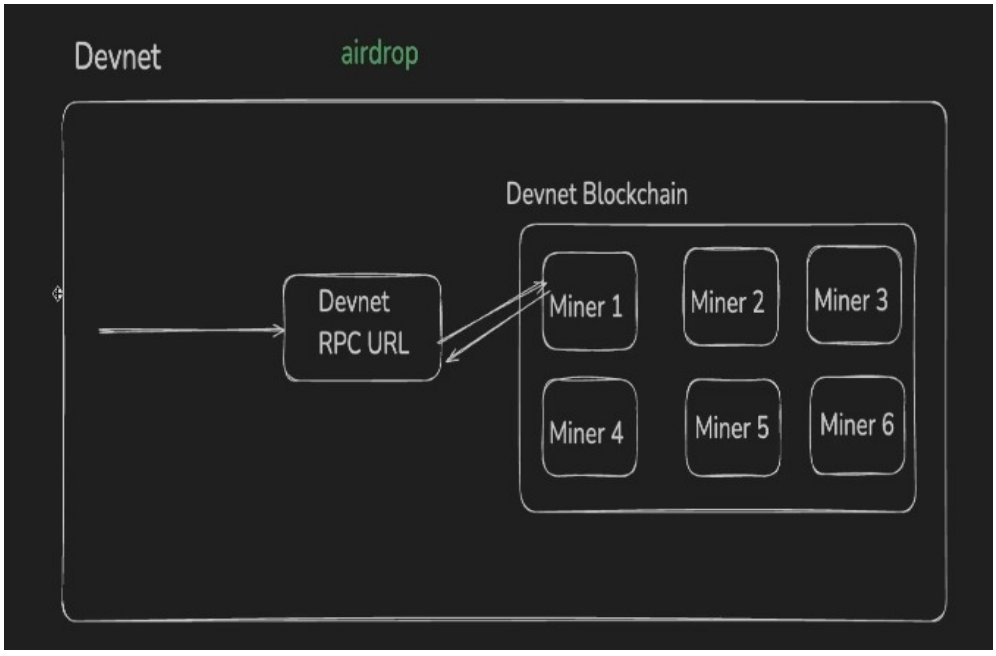
```
C:\Windows\System32\cmd.e
D:\Data\Programs\Solana\bin>solana-keygen new
Generating a new keypair

For added security, enter a BIP39 passphrase

NOTE! This passphrase improves security of the recovery seed phrase NOT the
keypair file itself, which is stored as insecure plain text

BIP39 Passphrase (empty for none):

Enter same passphrase again:


Wrote new keypair to C:\Users\Admin\.config\solana\id.json
==========================================================================
pubkey: 948xxGpw8PCYtuW2svtixPZgNjQU5pdrFYRurPjYK9GW
==========================================================================
Save this seed phrase and your BIP39 passphrase to recover your new keypair:
zero shop decide age gather soul video swear twist make radar quick
==========================================================================

D:\Data\Programs\Solana\bin>solana address
948xxGpw8PCYtuW2svtixPZgNjQU5pdrFYRurPjYK9GW

D:\Data\Programs\Solana\bin>solana config get
Config File: C:\Users\Admin\.config\solana\cli\config.yml
RPC URL: https://api.mainnet-beta.solana.com
WebSocket URL: wss://api.mainnet-beta.solana.com/ (computed)
Keypair Path: C:\Users\Admin\.config\solana\id.json
Commitment: confirmed

D:\Data\Programs\Solana\bin>
```

# RPC URL, TESTNET, DEVNET, MAINNET

RPC URL is way of getting info about anything like balance, transactions etc we did this last week using **alchemy** and this can be used by the user.

**IN COMPANY THERE IS STAGING,DEV AND PROB ENV**

**Devnet** is like dev space in which we can airdrop some sol coins and try our codes and all



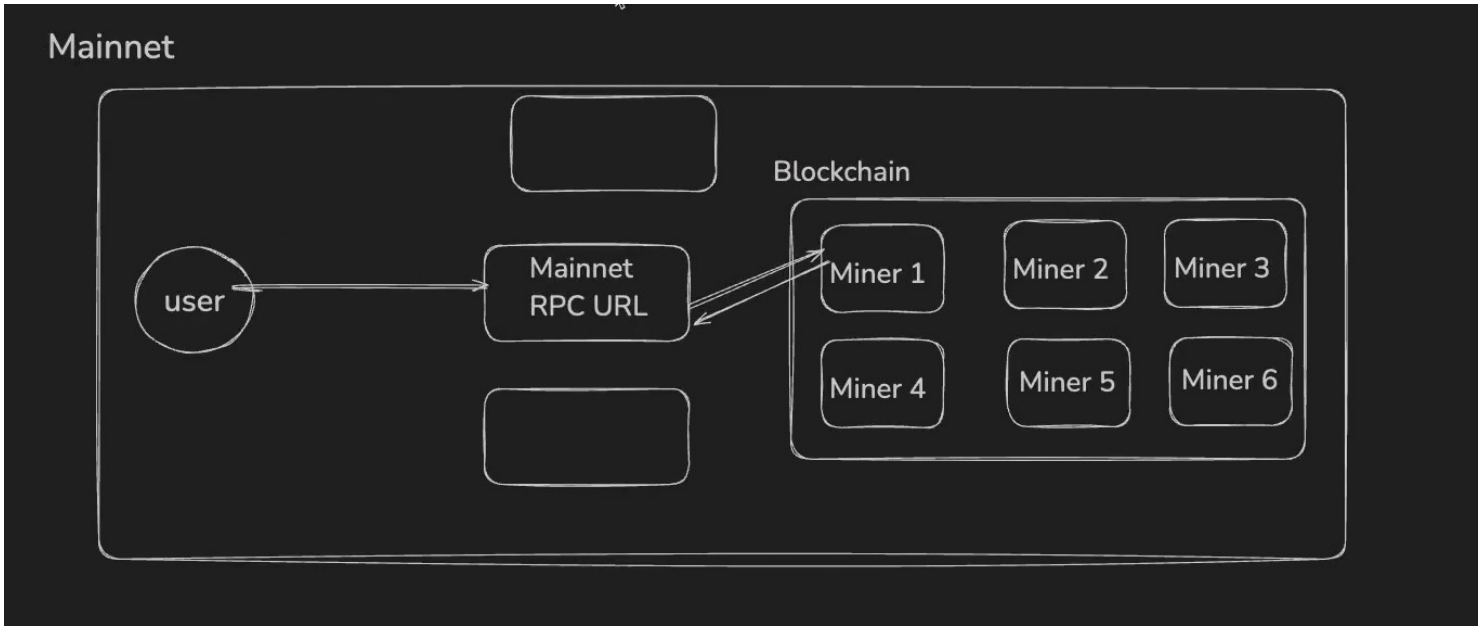
solana config set --url https://api.devnet.solana.com/

solana airdrop 1

you can airdrop yourself sol and also to check if you have you can use solana balance

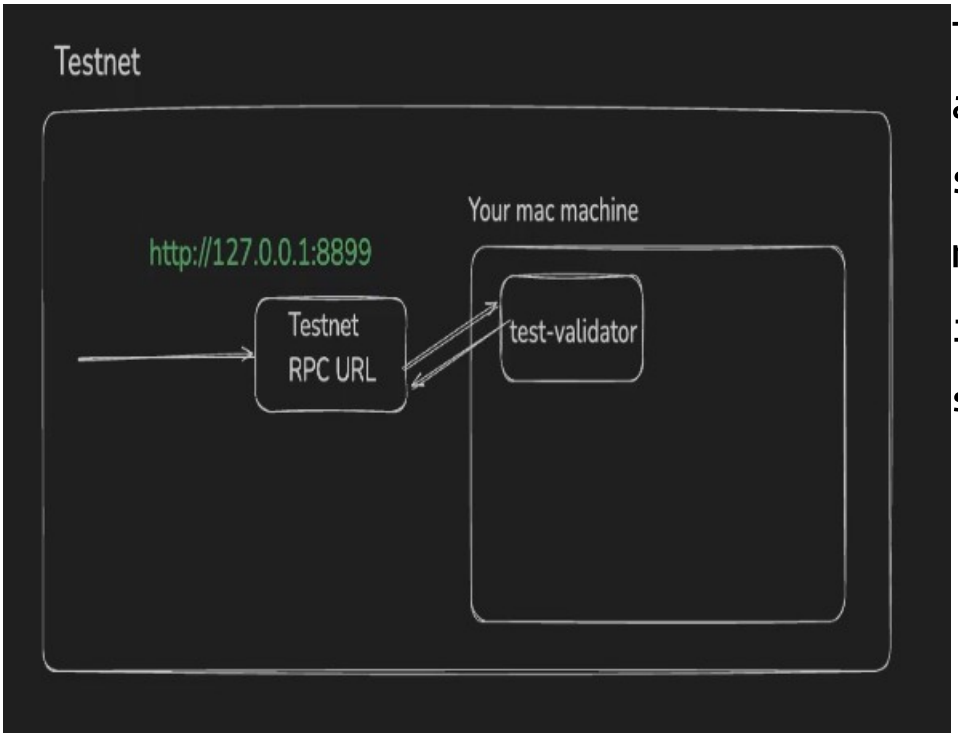
or go to website https://explorer.solana.com

## WHAT IS MAINNET?




In this chain the main transaction happens

## TESTNET




this is like when we are running our own server on our machine the command is

solana-test-validator

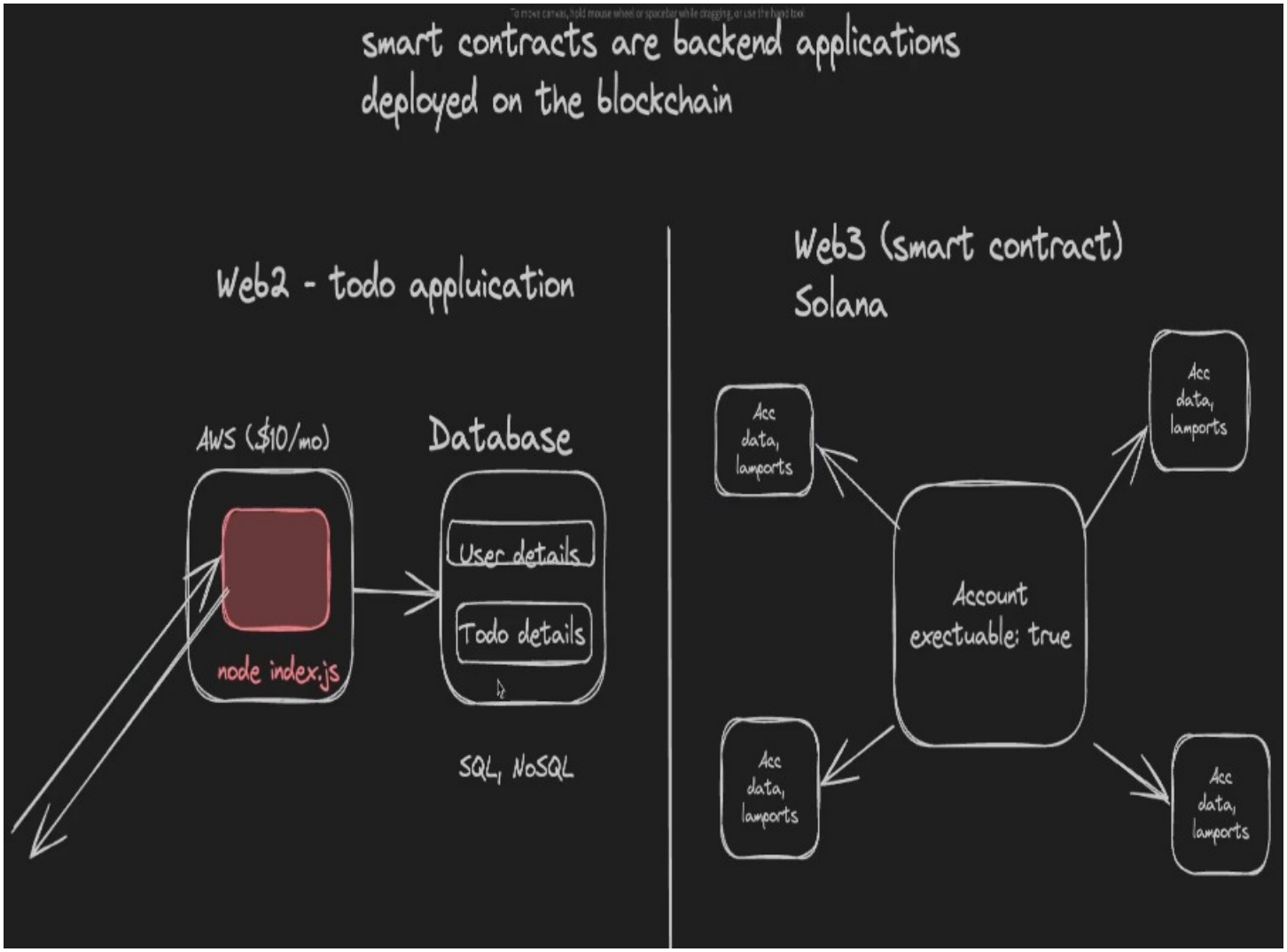
get ip from this and then use

solana config set --url http://127.0.0.1:8899

solana config get

solana balance

**SMART CONTRACTS ARE THE BACKEND APPLICATIONS DEPLOYED ON THE BLOCKCHAIN**

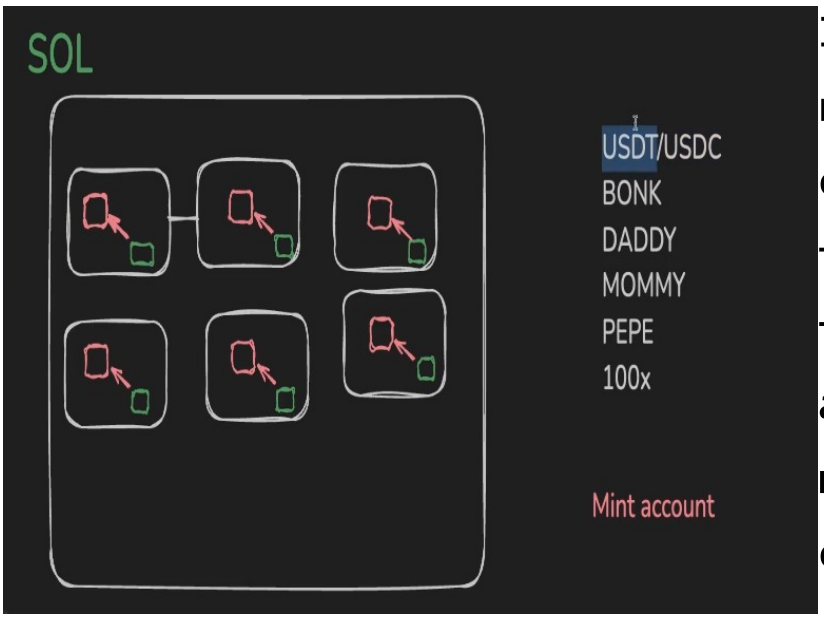



in solana how it stores data for a user it will create a new account with its data and lamports and the data will be stored in the data(bytes wala), program is stored in a single account and users data in another account not like the web2 which has a sql, nosql database.

WHY DO WE CREATE INDIVIDUAL ACCOUNT FOR THE USERS?
Bcoz is the user creates the account he will pay the sol money not the programmer as per his requirement he will pay and store the data.

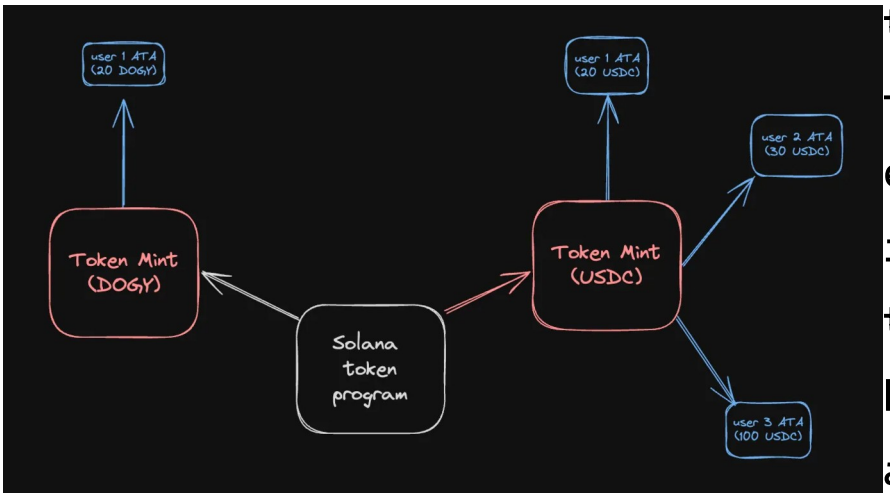
## TOKEN PROGRAM

**USDC** AND **USDT** ARE NOT A BLOCKCHAIN DOES NOT RUN CONSENSUS




If you want to create a new token you just have to create a account on the top of a token program and that new account is called as MINT ACCOUNT (mint means a like you have created a bank)

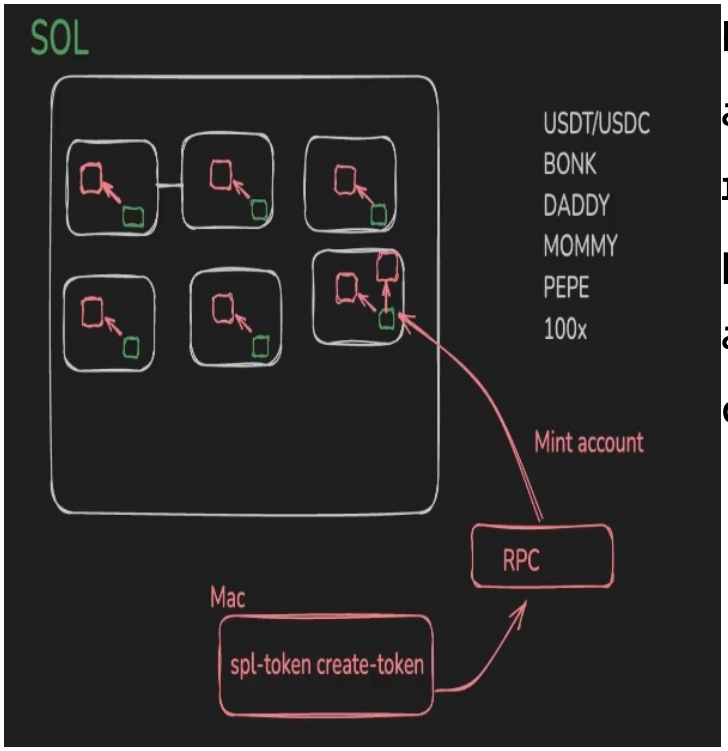
minting a token = creating new tokens to let other users access them




the coins are derived from the token program eg tether
in this the solana token program is exec but the mint account are not exec

SPL – solana programming library

HOW IS THE TOKEN CREATED?




Firstly you use the command and then the rpc sends the req to the solana token program which creates a mint account and the new token is created

first setup a new token

1. solana-keygen new -force
2. solana config get (to check the url is devnet or not)
3. airdrop some solana and then check balance
4. spl-token create-token

```
D:\Data\Programs\Solana\bin>spl-token create-token
Creating token 5masCSRxmE1SEXPzxQJeD6Pg651aqXZV6Jr6LrYS5Vzj under program To
kenkegQfeZyiNwAJbNbGKPFXCWuBvf9Ss623VQ5DA

Address:  5masCSRxmE1SEXPzxQJeD6Pg651aqXZV6Jr6LrYS5Vzj
Decimals:  9

Signature: 2FJW7ExutmP9tbDJx3SBtCbcjjuUHHtuoGv2zpoF6oBMKH4KZ6NgAyDnbPVC1gWbW
CGJiDp5gKwiLGRqwy6CMZjF
```

on the postman you can check its size by putting the devnet url and some json data

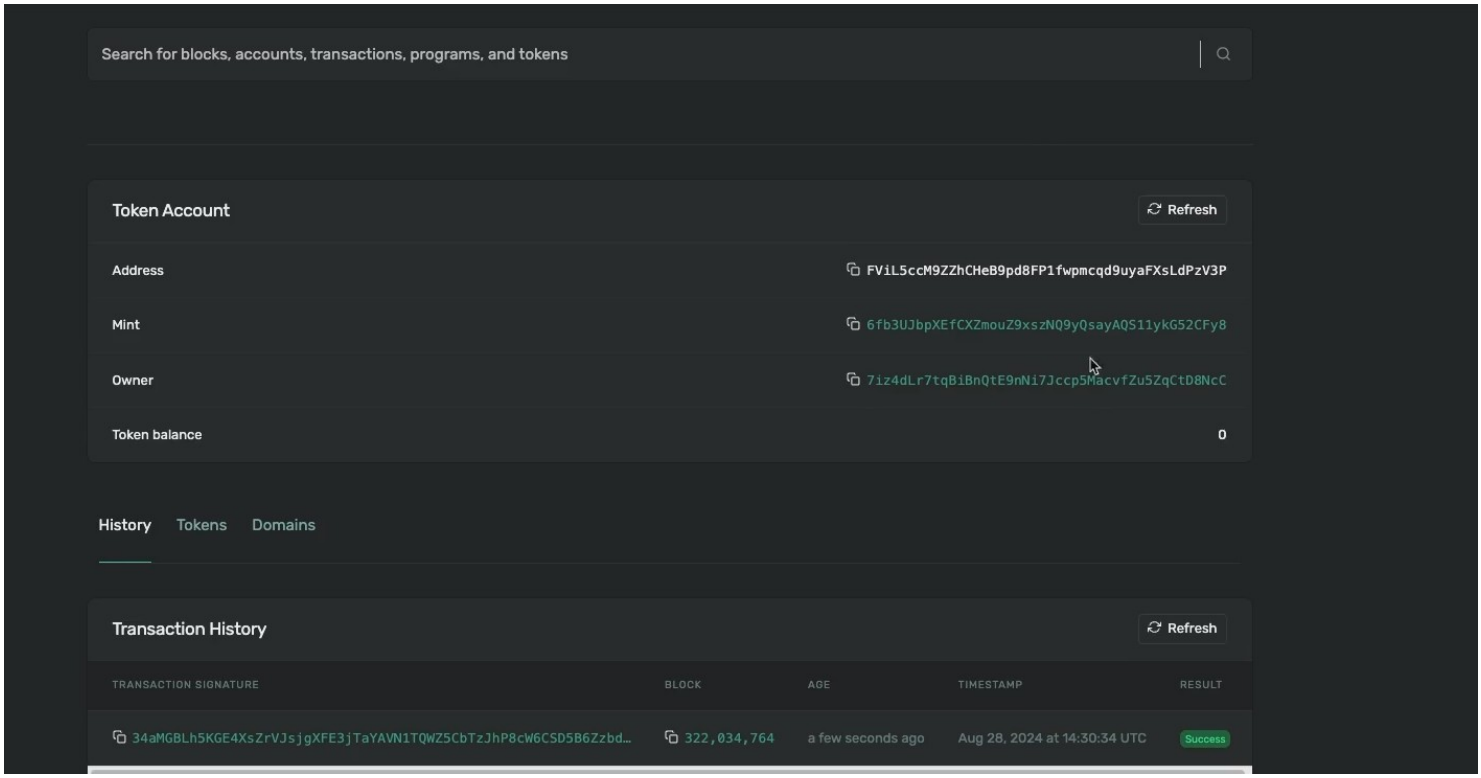
space is same 82 for usdc,doge,for my token etc

**MINT AUTHORITY = TOKEN CREATOR ADDRESS**

→ now to own a token you should have a associated account for the token

spl-token create-account **tokenAddress**

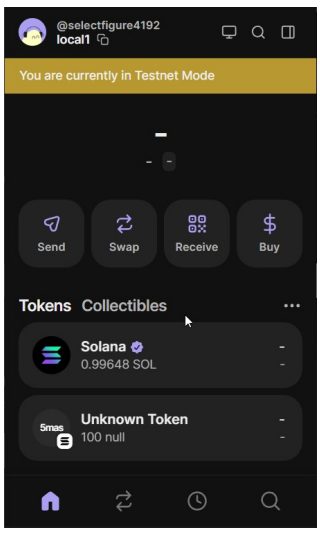
this associated account is created on top of the own address (its like kiska account kholna hai konsi currency par kholna hai)

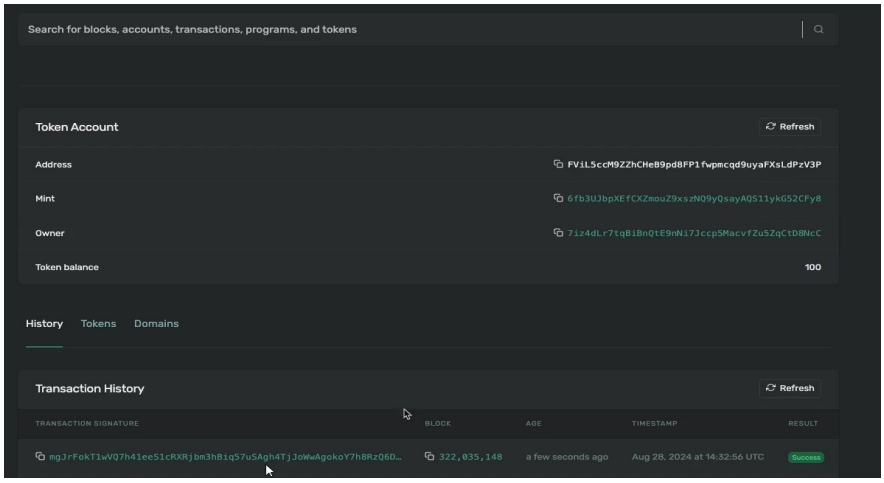



in this the mint is the token address, owner is the token creator address and this is a token account(Associated Account)

→ now to transfer the data use

spl-token mint **tokenID** 100(NUMBER OF TOKENS)



→ btw whenever new account is created a sol is deducted form the creator of the token account(as a gas) and then the new associated account is created.
→ for creation of the account the amount of sol required is high as compared to sending the normal sol the gas required is less

## what is decmials here?

Its just the lease number of decimal you can send means minimum you can send is 0.000000001

**you can change this using**

spl-token create-token –decimals 4

**NFT** is when the decimal is 0 you have to send natural numbers only no decimals allowed

Deanonymize transfers = using Anonymizers/mixers

eth = tornado cache

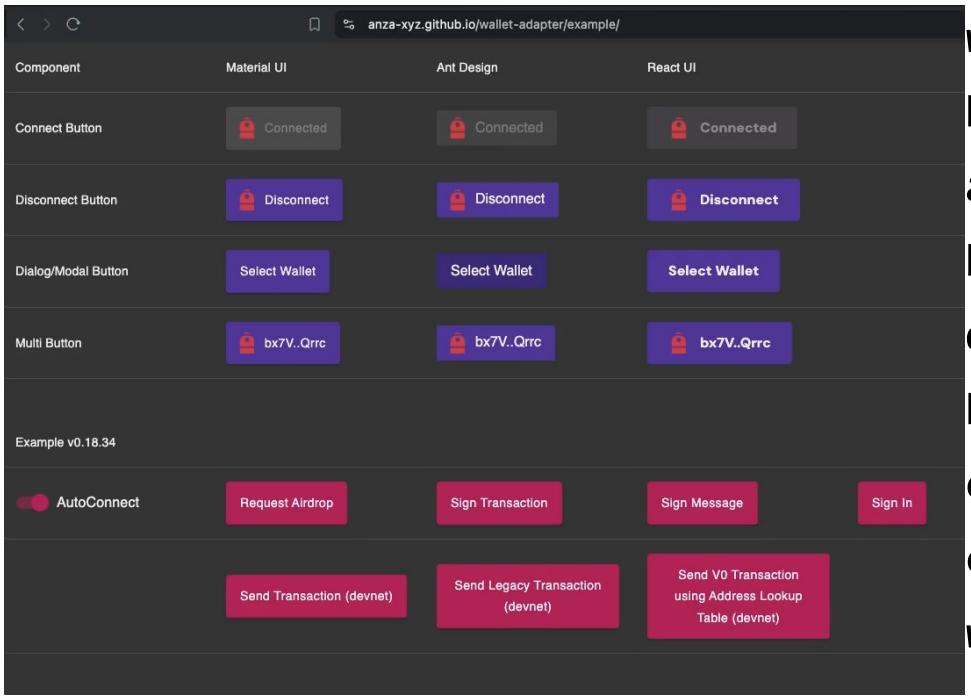
bitcoin = coin join

**creating a dapp**

```
npm create vite@latest
npm init -y
npm install -g yarn
npm install @solana/wallet-adapter-base \
    @solana/wallet-adapter-react \
    @solana/wallet-adapter-react-ui \
    @solana/wallet-adapter-wallets \
@solana/web3.js
```

WHAT IS A WALLET ADAPTER?

Its a thing which helps to connect the wallet to the website so that it can peform transactions and all , it can also detects what all type of wallets do you have and while doing transcation it first ask for a permission then only it works




wallet adapter just know you have three adapters in the browser and it can detect that

how it detect like each wallet has a code in the console

window.backpack

window.phantom

through this it try to find the wallet and then use that

Dapp is like any website which will ask for approval from the websites


Wallet adapter

means which helps you to connect to a wallet

metamask is a eth based wallet

one of the first wallet

uniswap is also a wallet adapter

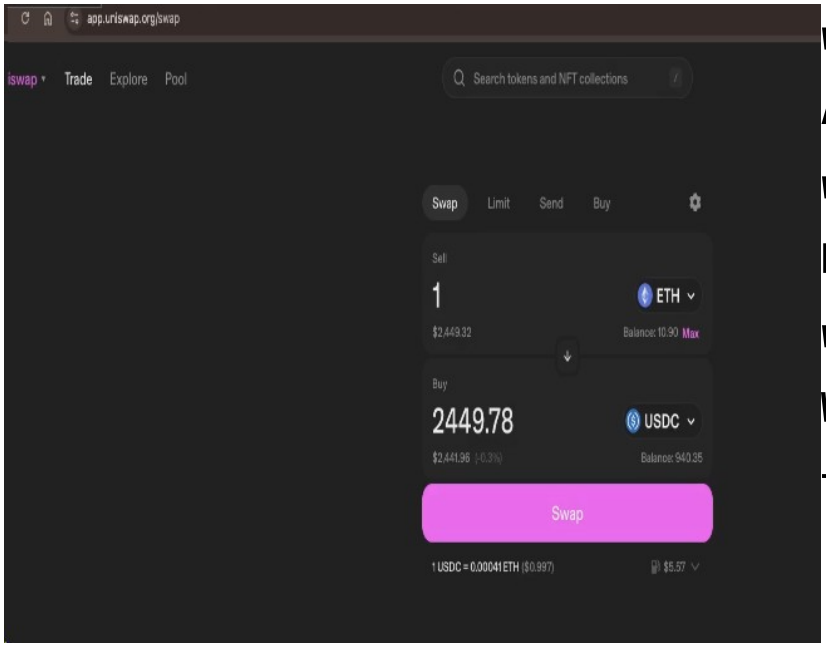
what does dap do?

Ask the wallet for approval for something

what does wallet do?

Ask the user for approval

if you press yes then using your private key it will sign the trasaction and then send the transaction to blockchain




what is a dap?

A uniswap website

what is a wallet?

Metamask

what is a wallet adapter?

Which helps a dap connect to its wallet

TODAY WHAT WE ARE CREATING WILL JUST HELP ASK THE USER FOR SOMETHING

we can use multiple types of adapter like unified jupyter

```
npm install @solana/wallet-adapter-base @solana/wallet-adapter-react @solana/wallet-adapter-react-ui @solana/wallet-adapter-wallets @solana/web3.js --legacy-peer-deps --ignore-scripts
```

bring all the dependencies required for the project

Things to do

Once a user connects to your dapp, you usually ask the user to do a few things with their wallet balance -

1. Request Airdrop
2. Show SOL balances (GET data from the blockchain)
3. Send a transaction (Send a transaction to the blockchain)
4. Sign a message (Verify the ownership of the wallet)

Creating a airdrop website

in this we create a website which helps us to send some sol on the devnet channel like eg solfaucet ,https://faucet.solana.com/


what we are building is like it will automatically fetch the wallet id and then we just have to enter the amount to send some airdrop

now to setup a project install this

from the github page of wallet adapter use the typescript docs or the solana wallet option

https://github.com/anza-xyz/wallet-adapter/blob/master/APP.md

three react things are required for this

context api, prop drilling,

```jsx
function App() {
    //create own rpc url? using alchemy
  return (
    <ConnectionProvider endpoint={"https://api.devnet.solana.com"}>
            <WalletProvider wallets={[]} autoConnect>
                <WalletModalProvider>
                    <div>
                        hi there <b> hello</b>
                    </div>
                </WalletModalProvider>
            </WalletProvider>
        </ConnectionProvider>
  )
```

this is the code we are using we just wrapped our code inside some providers

in this the wallet array will be empty as the wallet provider can itself detect all the wallets which follows the wallet rules and are autodetected and if some does not then add to the array

https://github.com/anza-xyz/wallet-adapter/blob/master/APP.md

copying the code from the above website

now for the button for connecting the solana accout we can just use the button from the library

Walletmultibutton, Walletdisconnectbutton from the library from the wallet adapter providers

it only works for original account not the devnet but work for mainnet

1 sol = 10^9 lamports and in the  await connection.requestAirdrop(wallet.publicKey,10) the last parameter the the lamports not the sol

App.jsx Airdrop.jsx

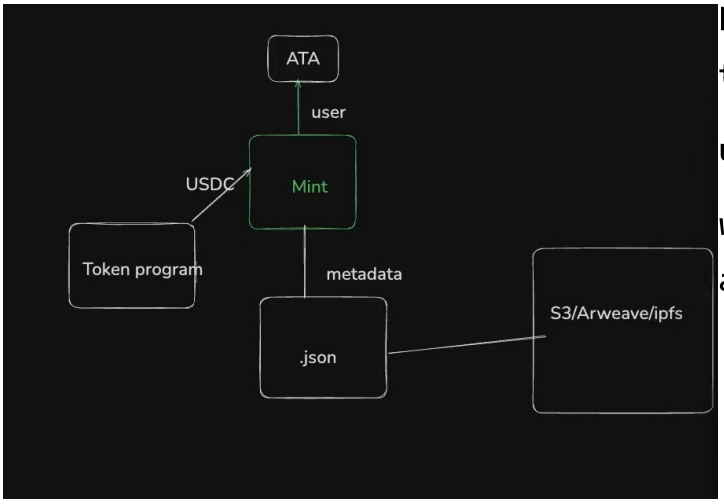
```jsx
import { useConnection, useWallet } from "@solana/wallet-adapter-react"

//hooks in react
//the wallet provider provides this use wallet and now we can access the
//user details
//The useWallet hook provies the wallet variable inside the Airdrop component
// because i wrapped the Airdrop component inside the WalletProvider
export function Airdrop(){
    const wallet = useWallet();
    const {connection} = useConnection();
    async function sendAirdropToUser(){
        const amount = document.getElementById("publicKey").value
        await connection.requestAirdrop(wallet.publicKey,amount*1000000000)
        alert("airdropped sol")
    }
    return <div>
        <input id="publicKey" type="text" placeholder="Amount"></input>
        <button onClick={sendAirdropToUser}>Send Airdrop</button>
    </div>
}

```

you can also check the sol in explorer solana

pretty easy shit like just taking the amount and then using the function to send the airdrop using a function only and defining the url in the app.jsx using the alchemy or the original website

next task is to add a behaviour of showing the balance in the account

third usecase is the signing the message on the behave of the user




how the metadata is stored and used in the system

using the token 22 program

wormhole in which many currencies exist at once