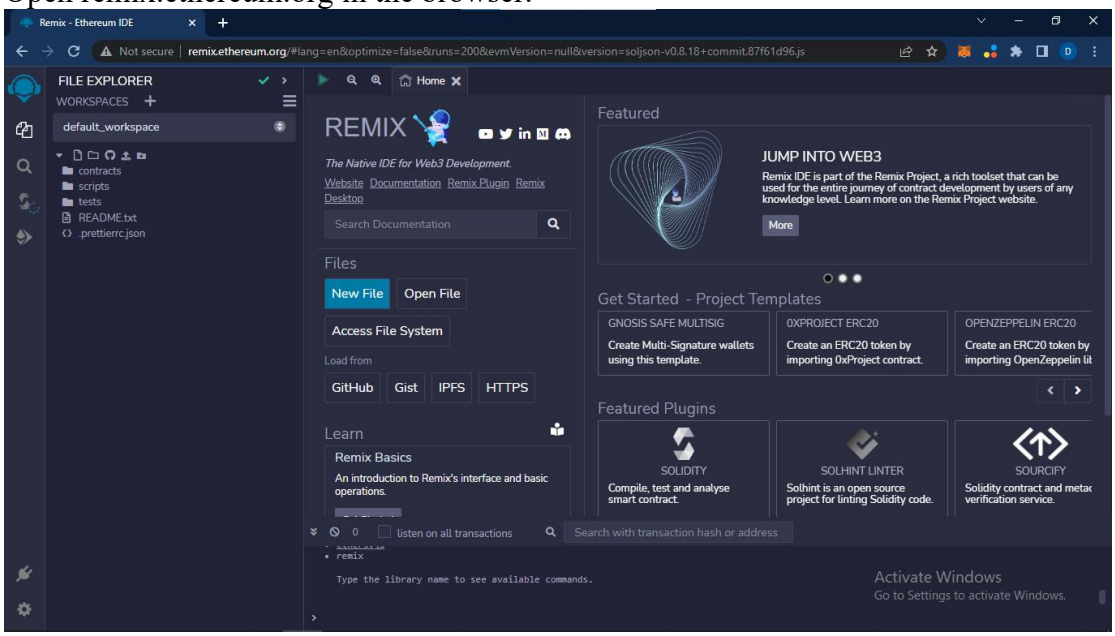
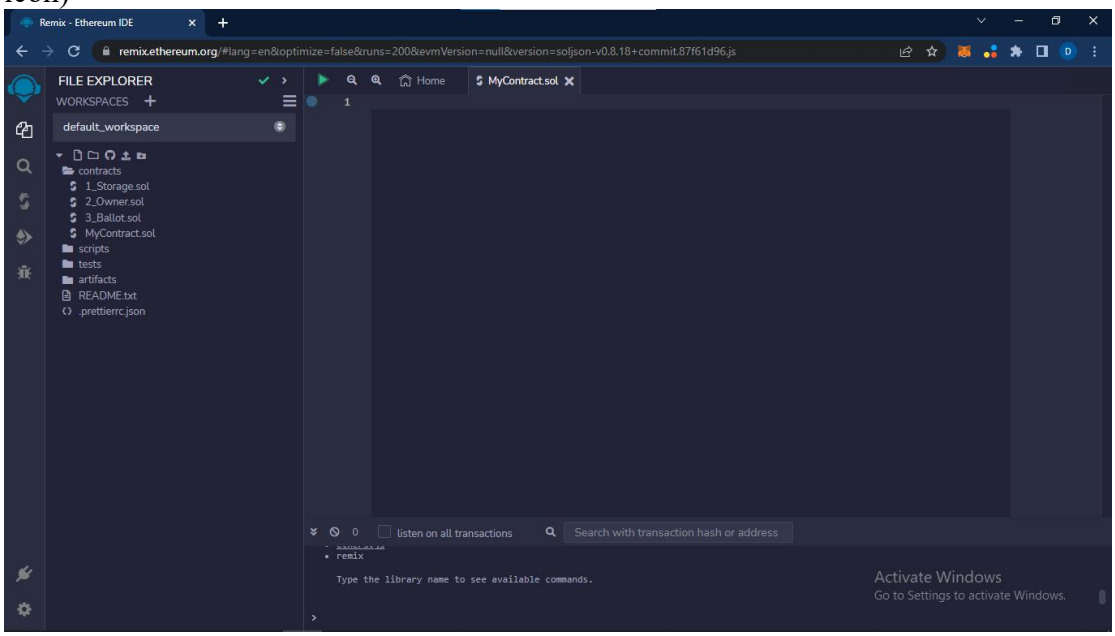
Task: Create a solidity based smart contract

Solidity programming:
Use remix IDE - Remix is a Solidity IDE that's used to write, compile and debug Solidity code. Solidity is a high-level, contract-oriented programming language for writing smart contracts.

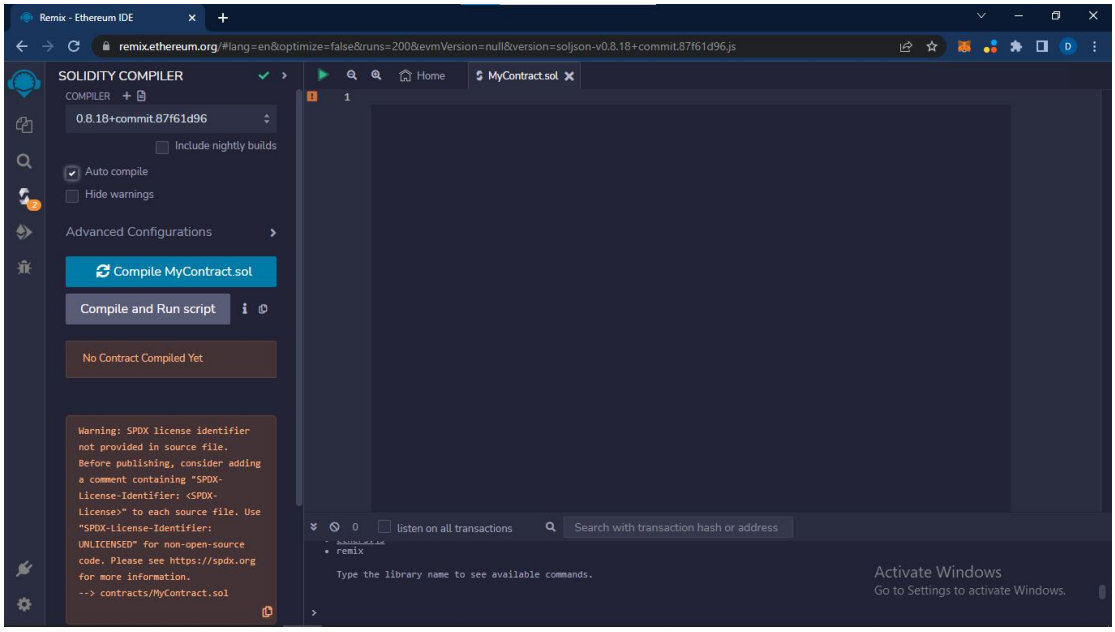
Open remix.ethereum.org in the browser.



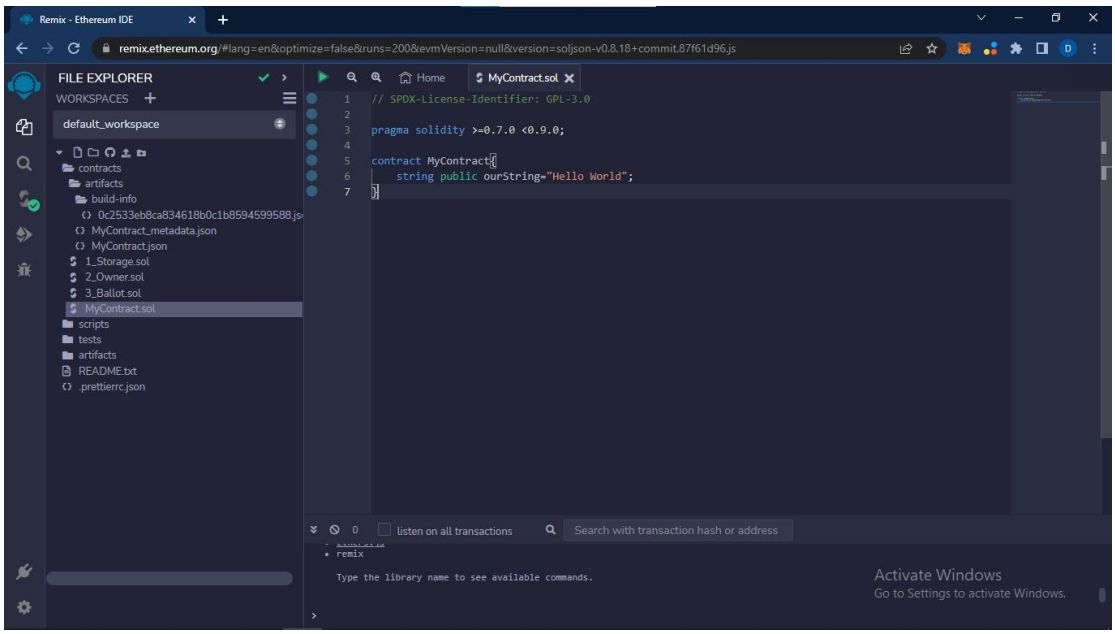
In default_workspace – contracts folder – create your file MyContract.sol (Use file icon)



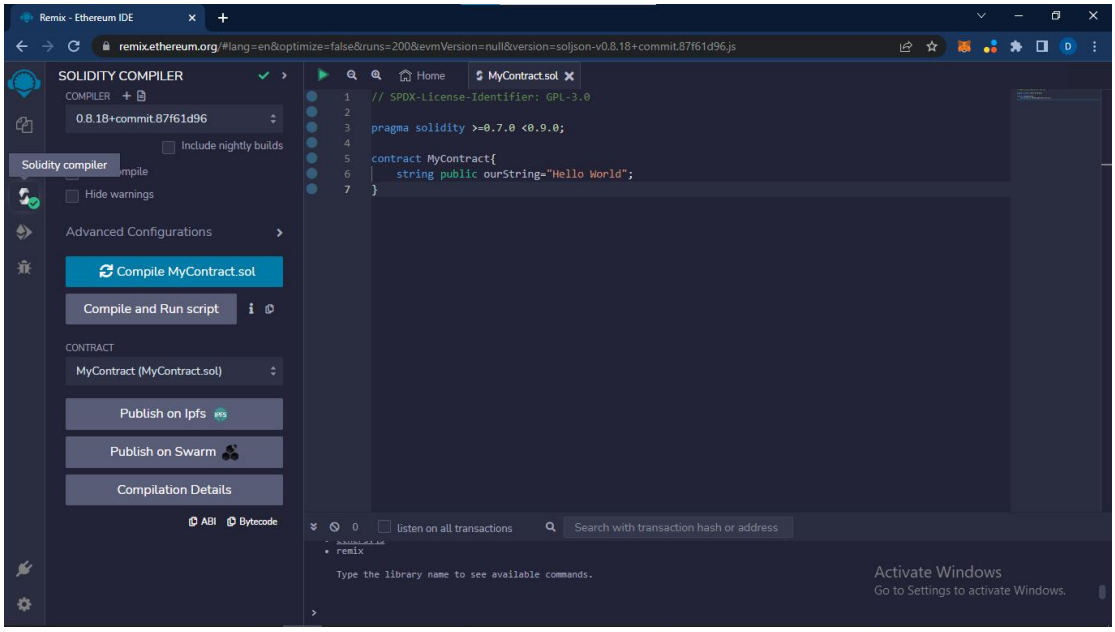
Ensure solidity compiler is in auto-compile mode as follows: If not select the compiler icon highlighted in red and check auto-compile



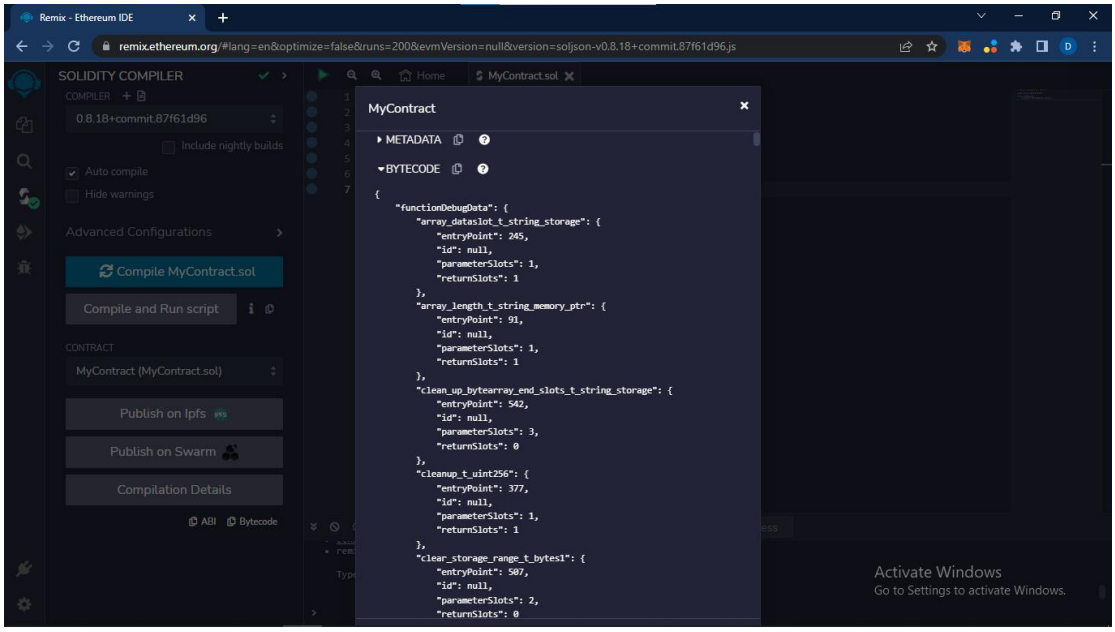
Creating smart contract:



Select Solidity Compiler: (icon with green tick and has tooltip solidity compiler in the leftmost)



Click Compilation details button and observe generated bytecodes of smart contract



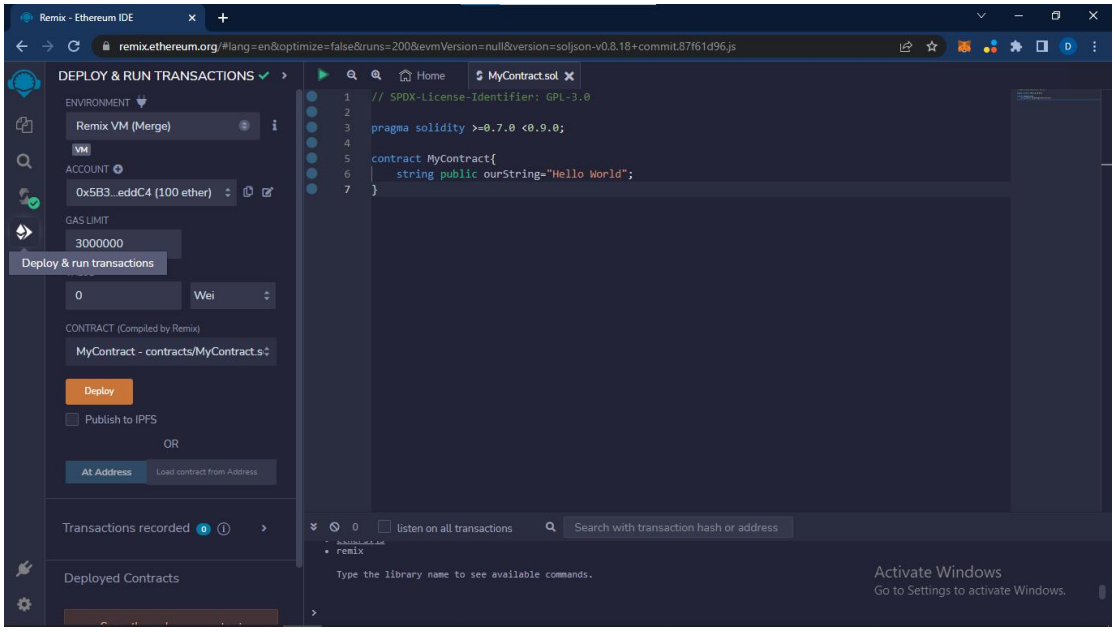
Click other links and study

……..

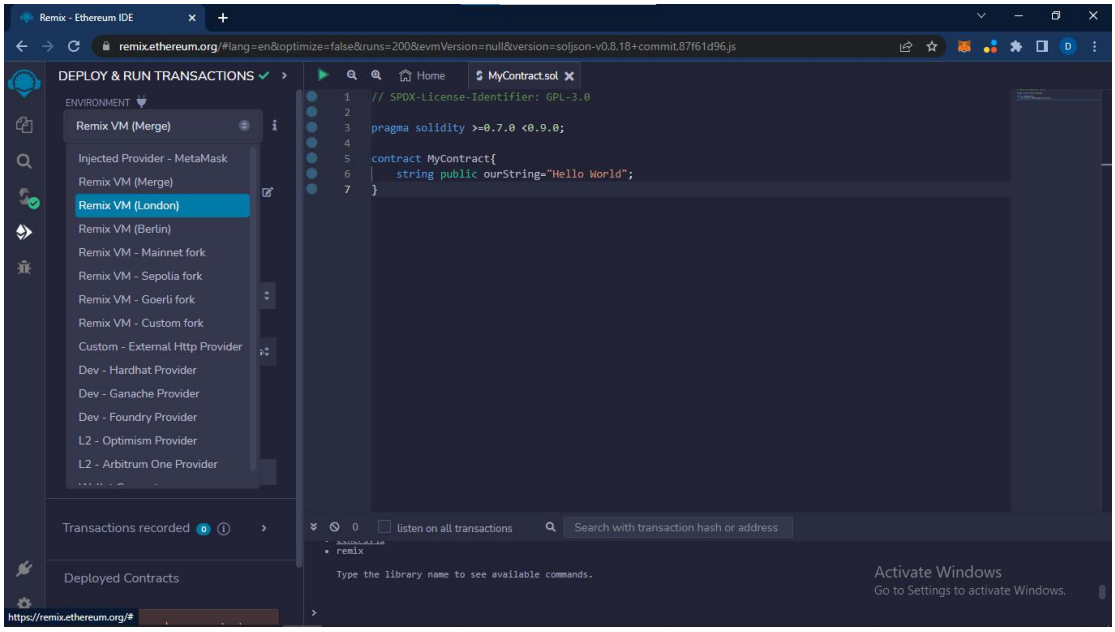
The smart contract you developed is auto-compiled and the bytecodes are generated.

……..

Now we need to interact with the smart contract. So normally we need cryptowallet like metamask. However, remix provides an inprocess and free deployment. This is the icon below solidity compiler.



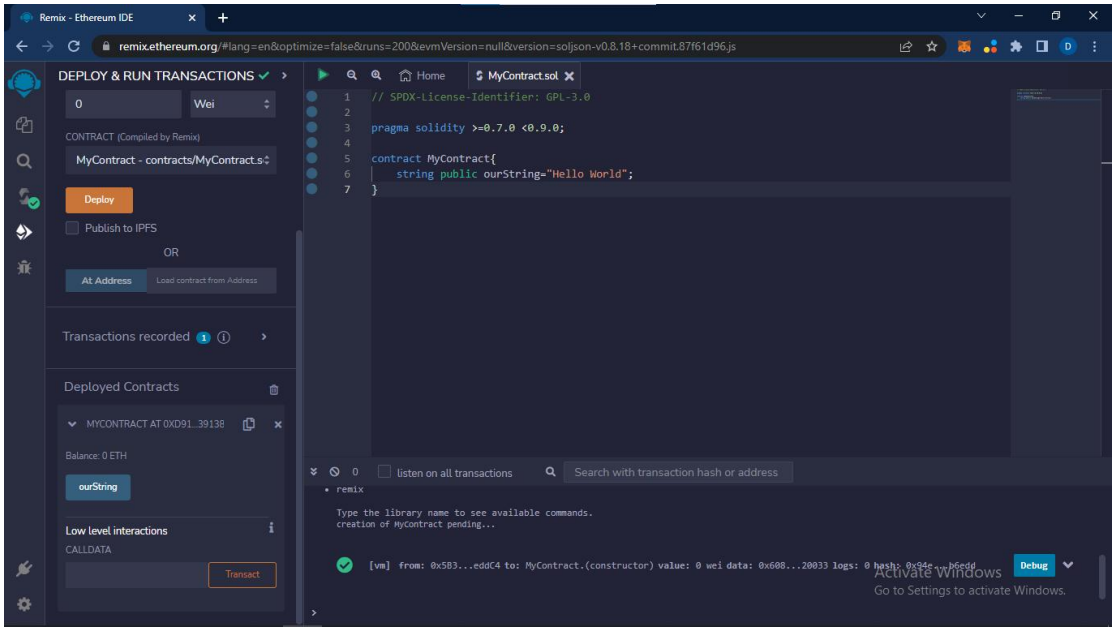
Observe the environment: Remix VM



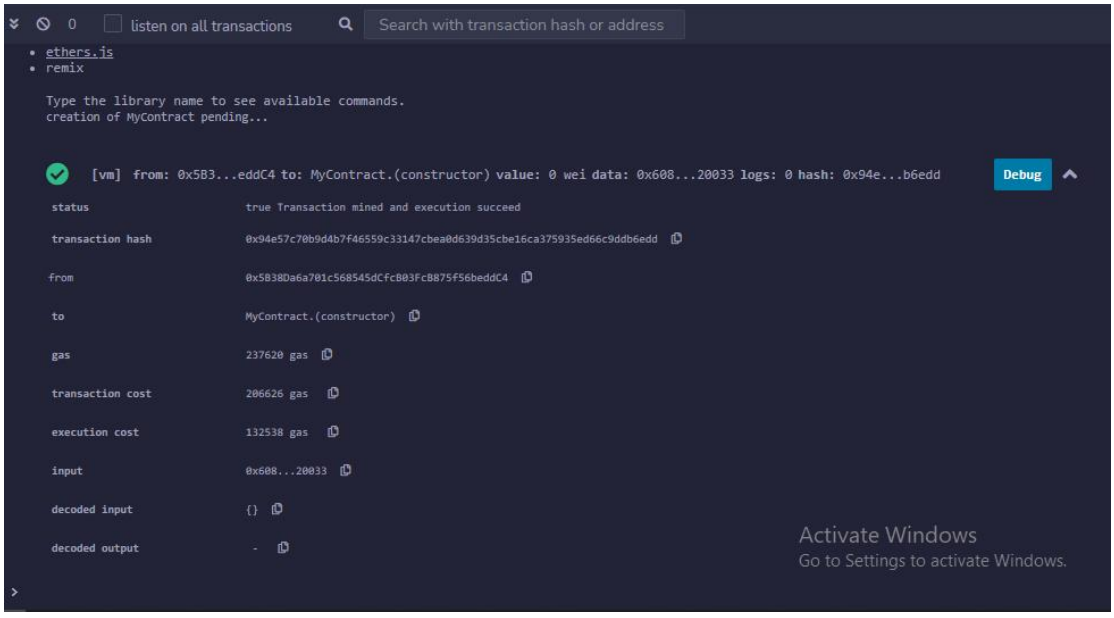
There is account, gas limit and ether value to work with.

……..

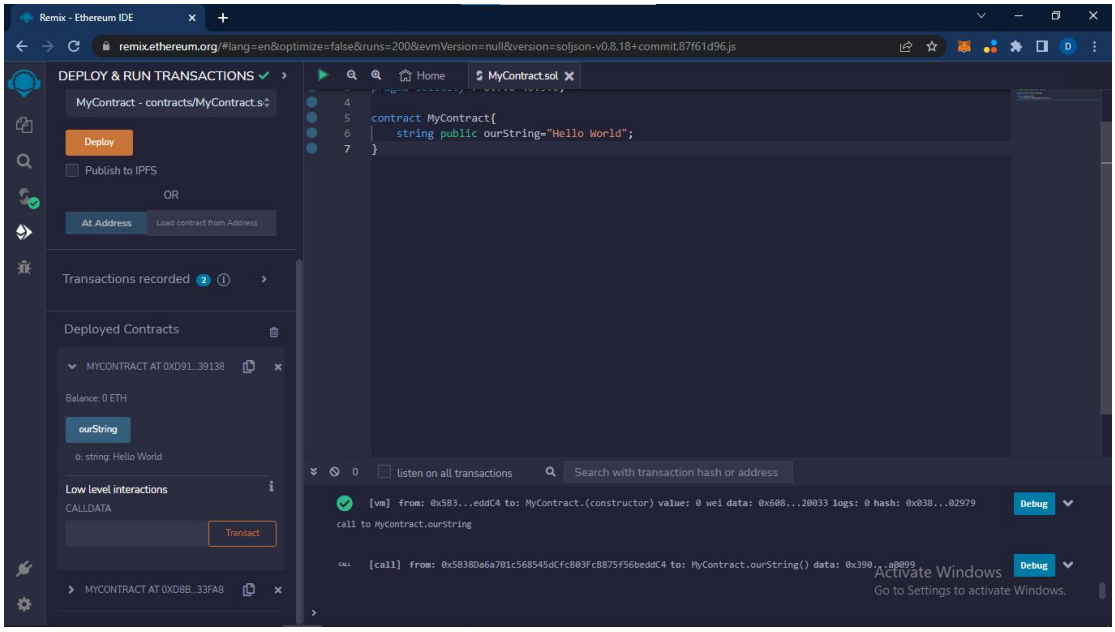
Now select Remix VM London and deploy. Observe Deployed Contracts:



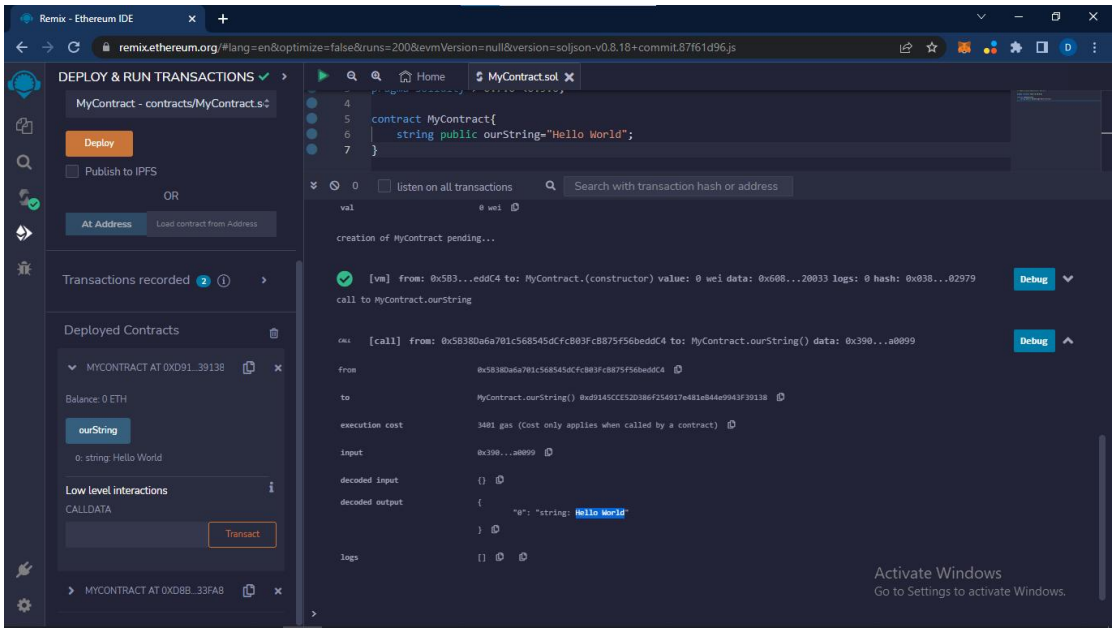
Click on the transaction in the transaction log to see details:



Now let us run the Smart Contract:
Select an account.
Deploy - Click
ourString – Click

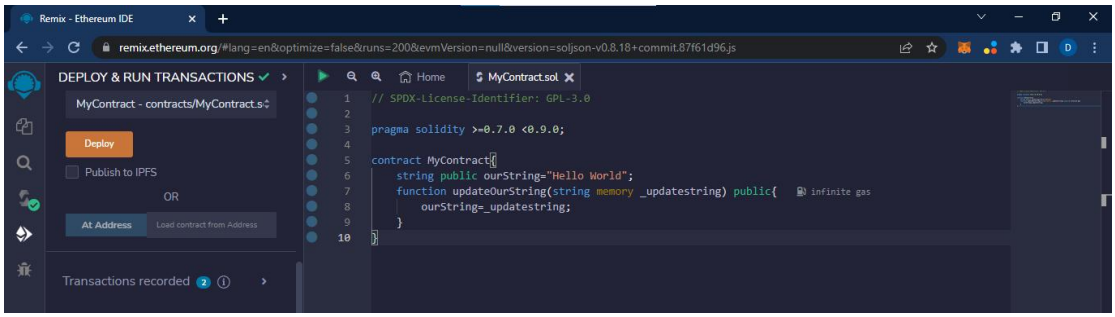


Observe below ourString- Hello World is displayed i.e Smart Contract is running
Observe transaction log that suggests Smart contract created and called:

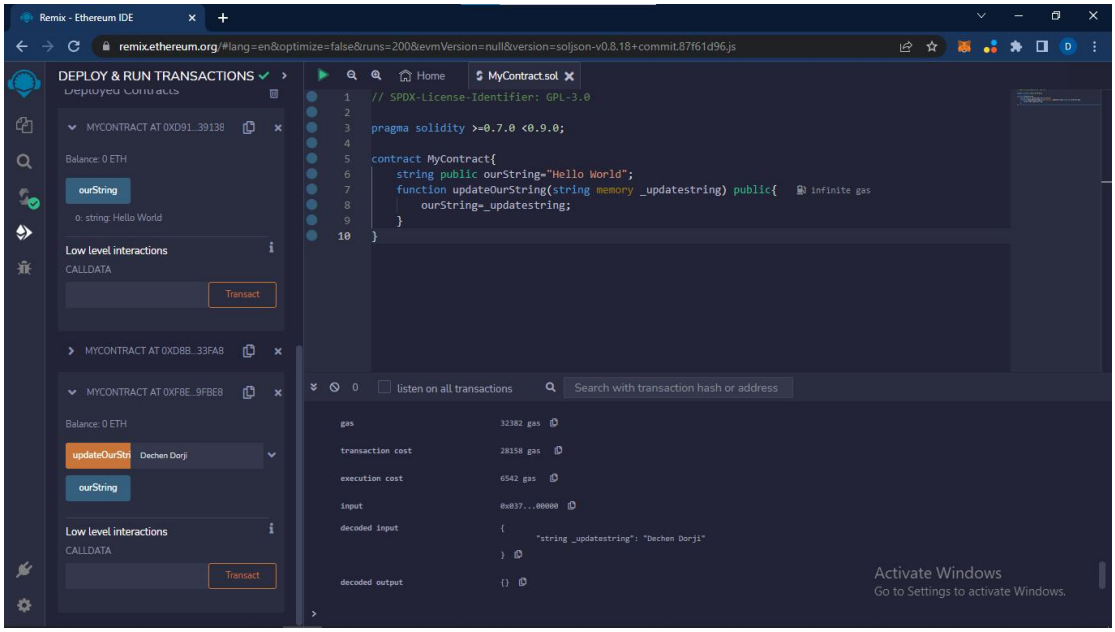


……..END……..……..

Adding a function in smart contract:



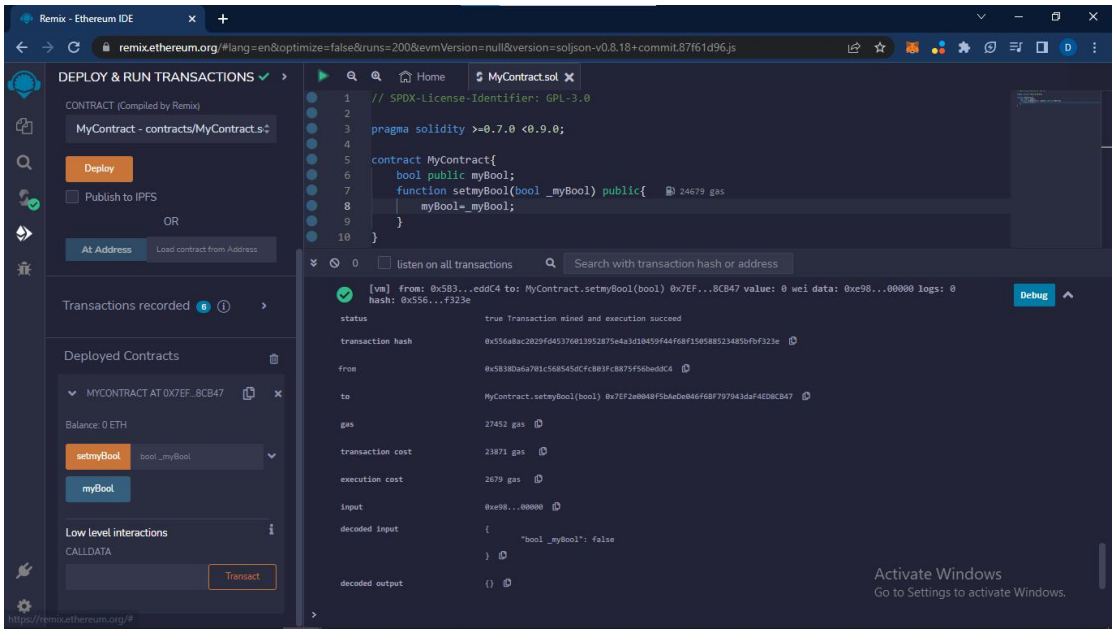
Deploy:

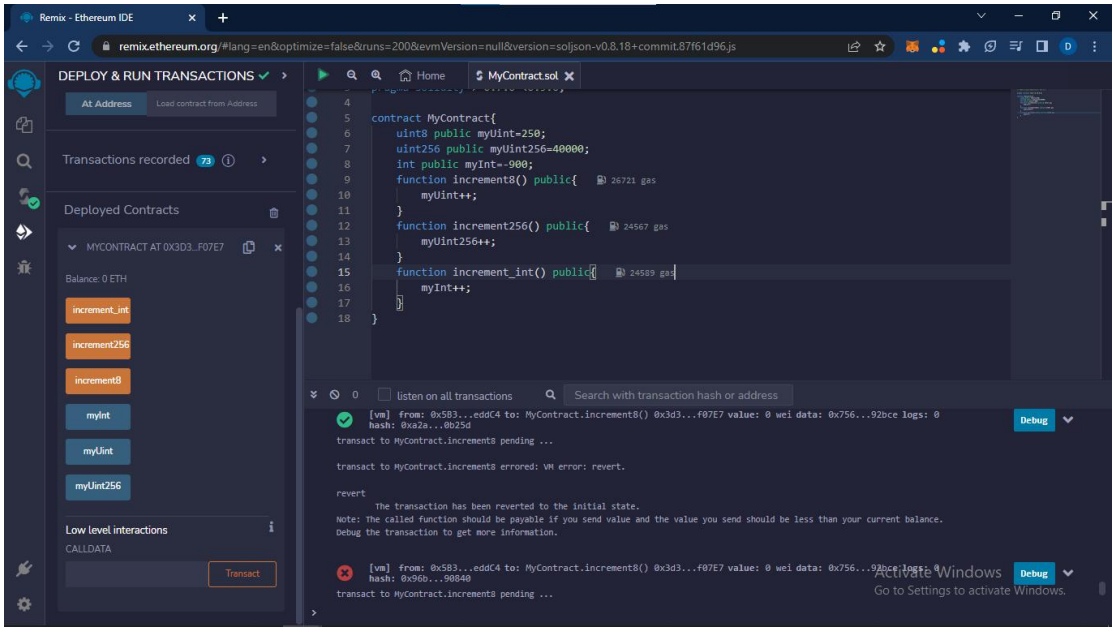


Check transaction log for the function being called:

……..END……..……..

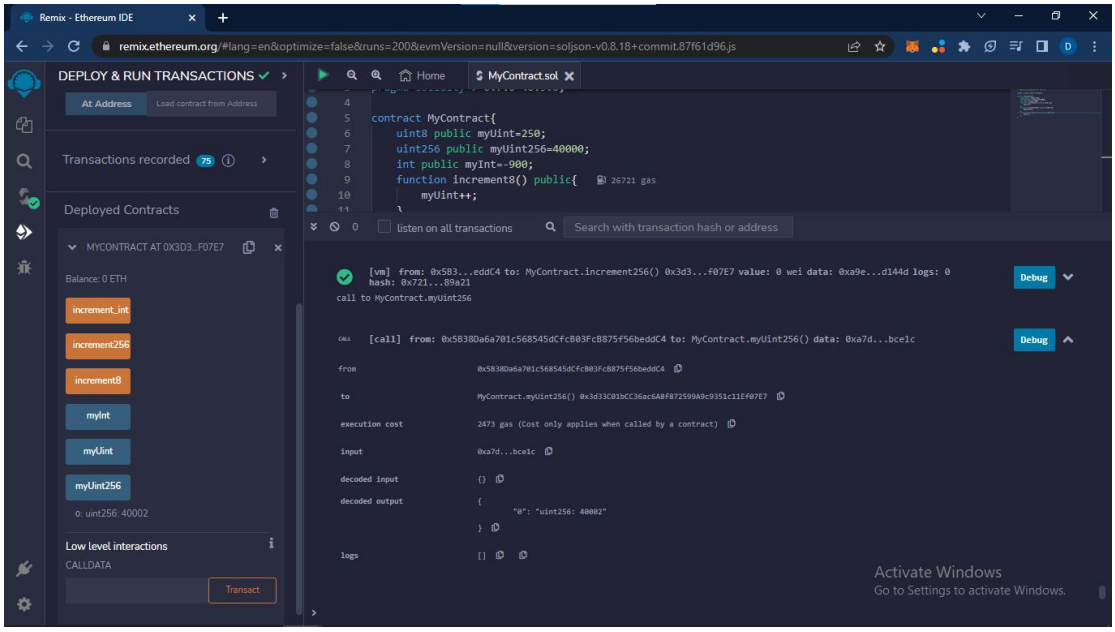
Datatypes:



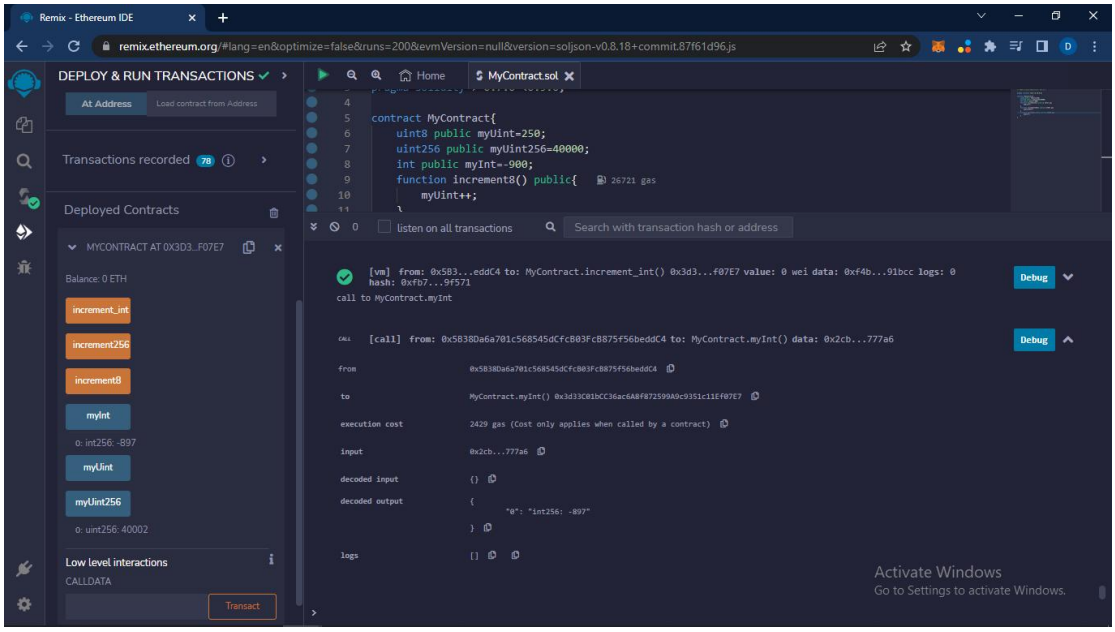


Observe increment8 was clicked 6 times and hence a range error is displayed.

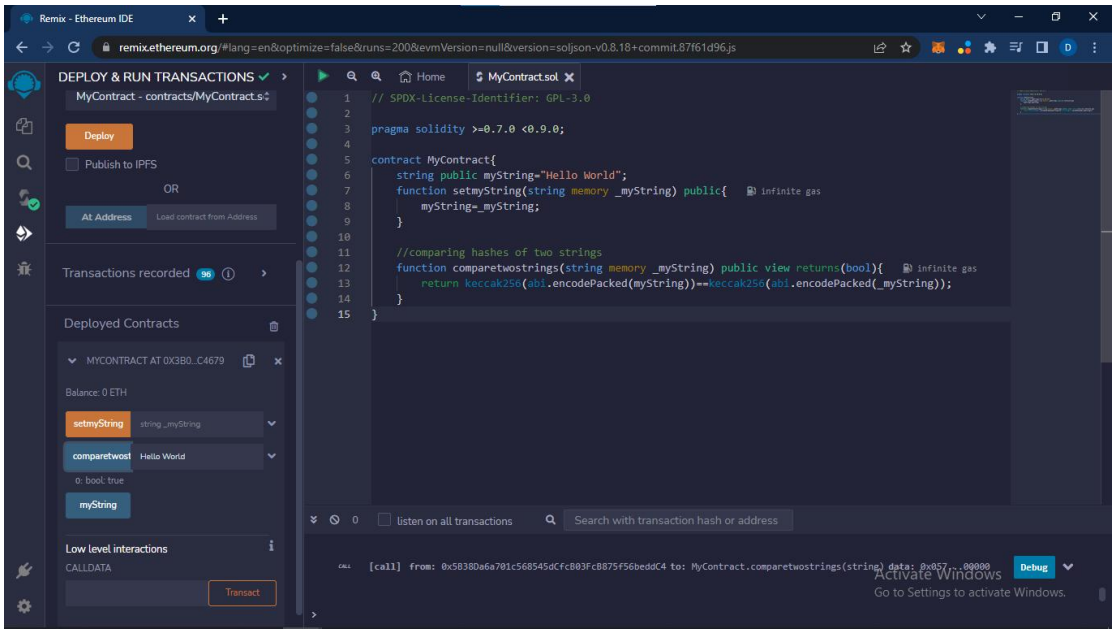
Called increment256 twice then call myUint256

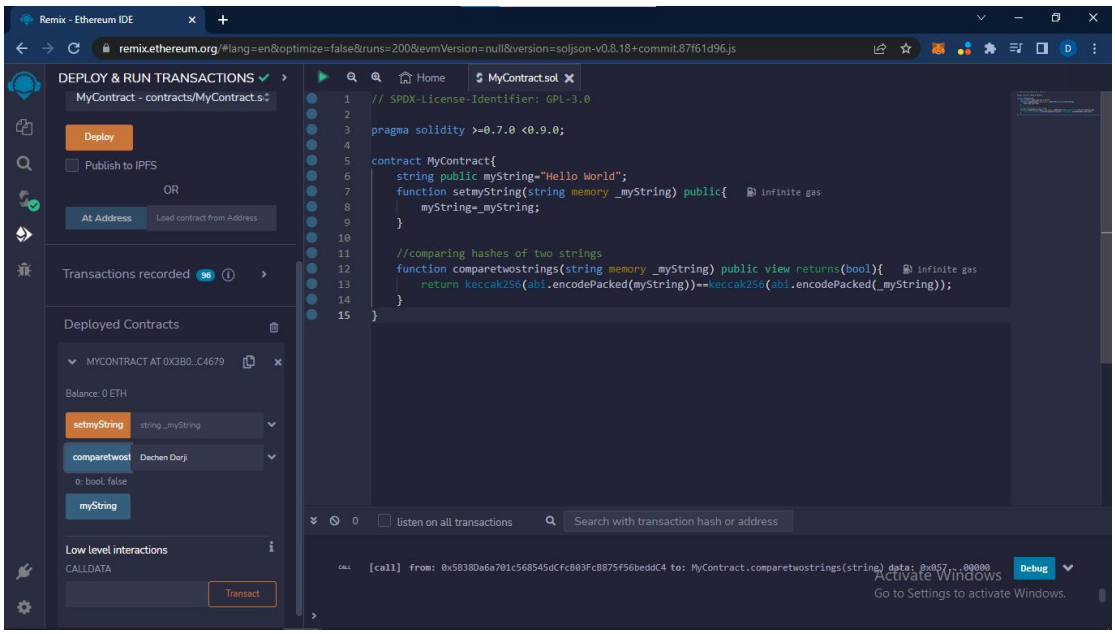


Similarly for int called increment_int thrice and then myInt

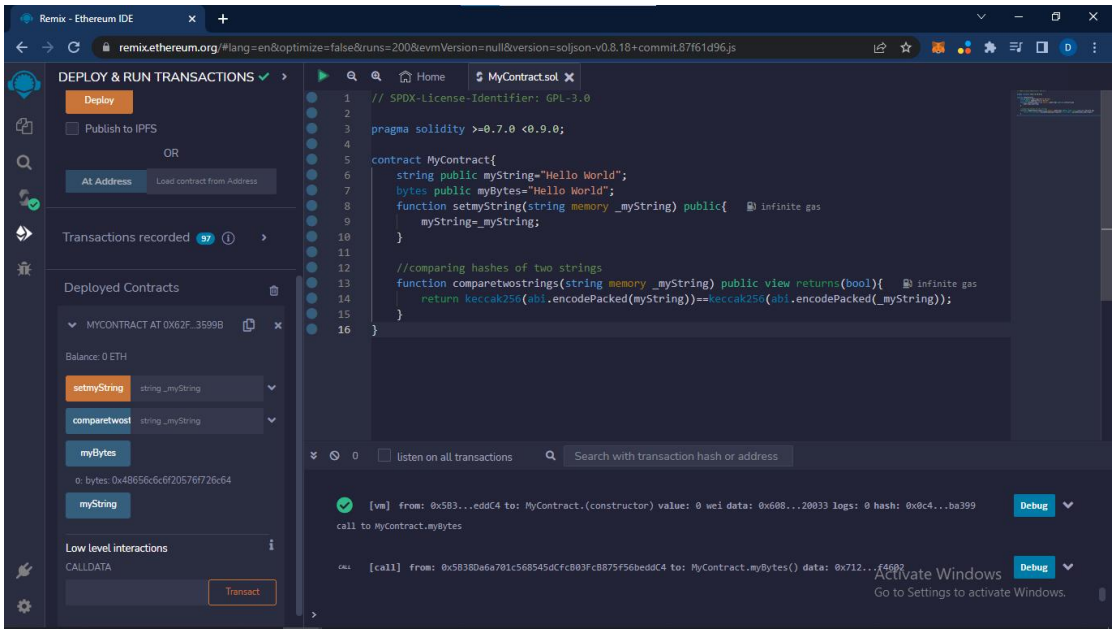


String manipulation:

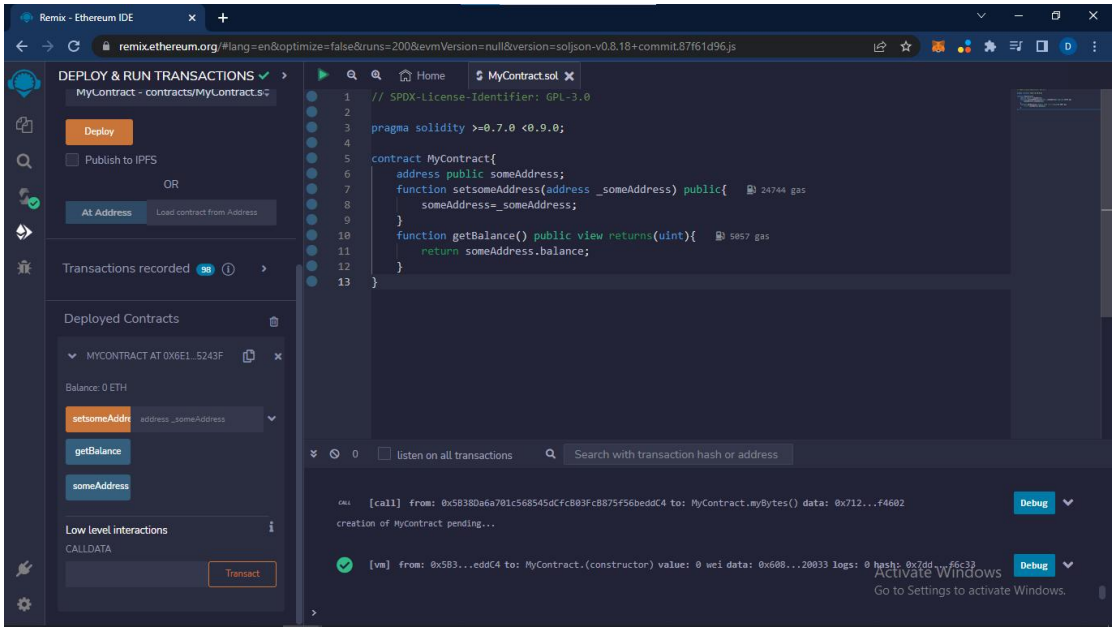




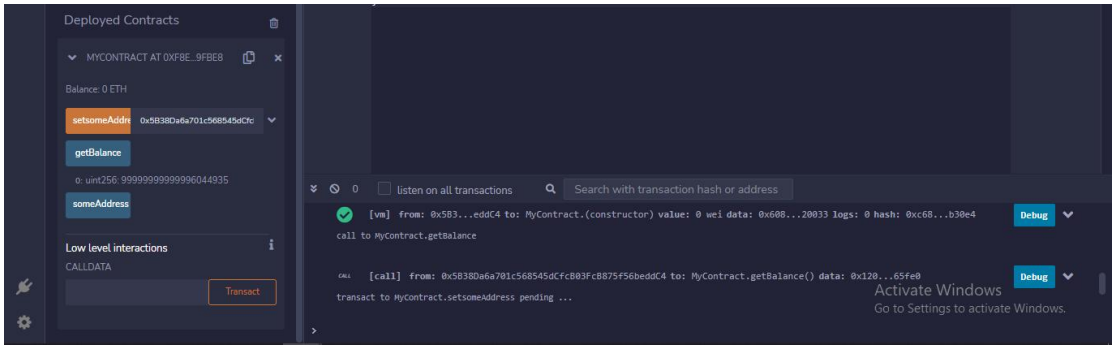
Bytes:



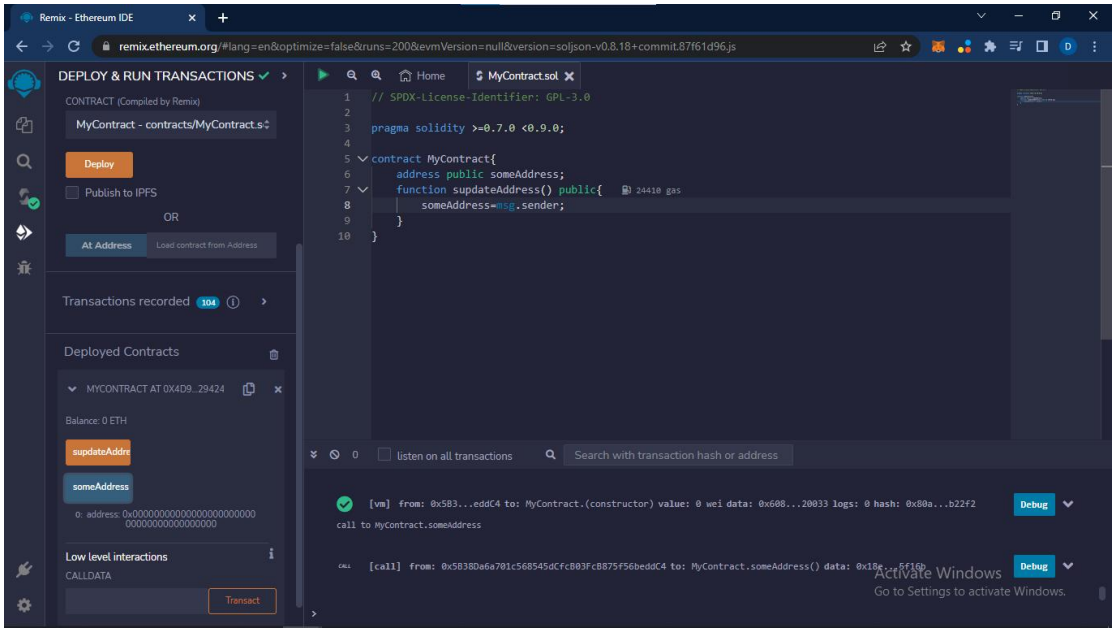
Address-stores 20bytes of Ethereum address



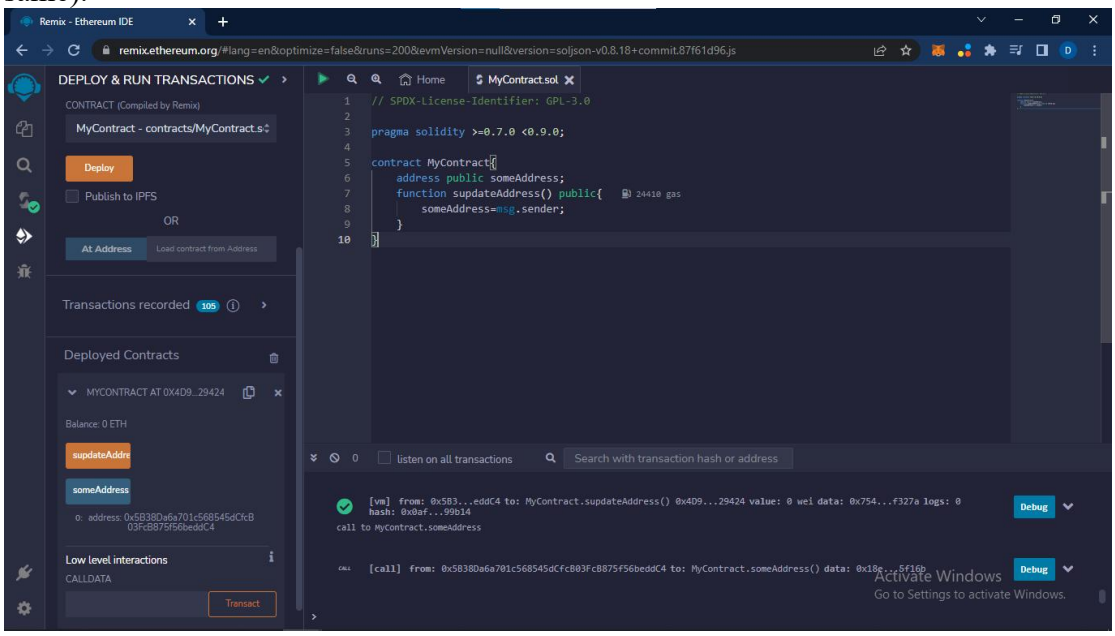
- ✓ Copy the address on deploy-account
- ✓ Let this copied address be the argument of setsomeAddress
- ✓ Then click getBalance()



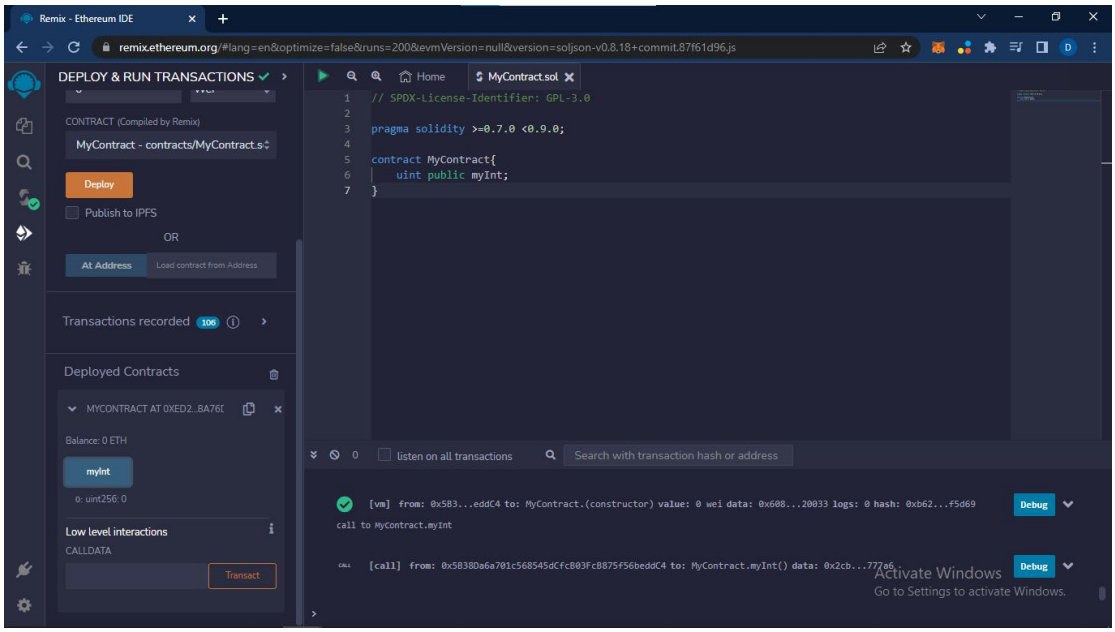
msg.sender will have the account address that interacted with the contract



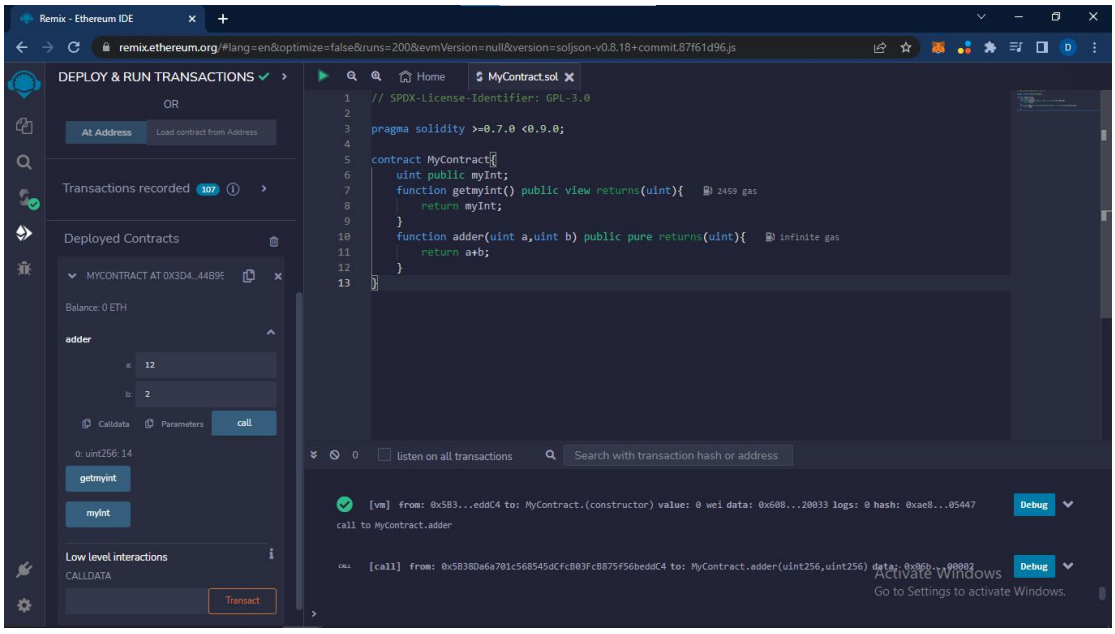
Now click updatesomeAddress and then someAddress msg.sender picks the account address associated with this contract (see the account address above it will be the same).



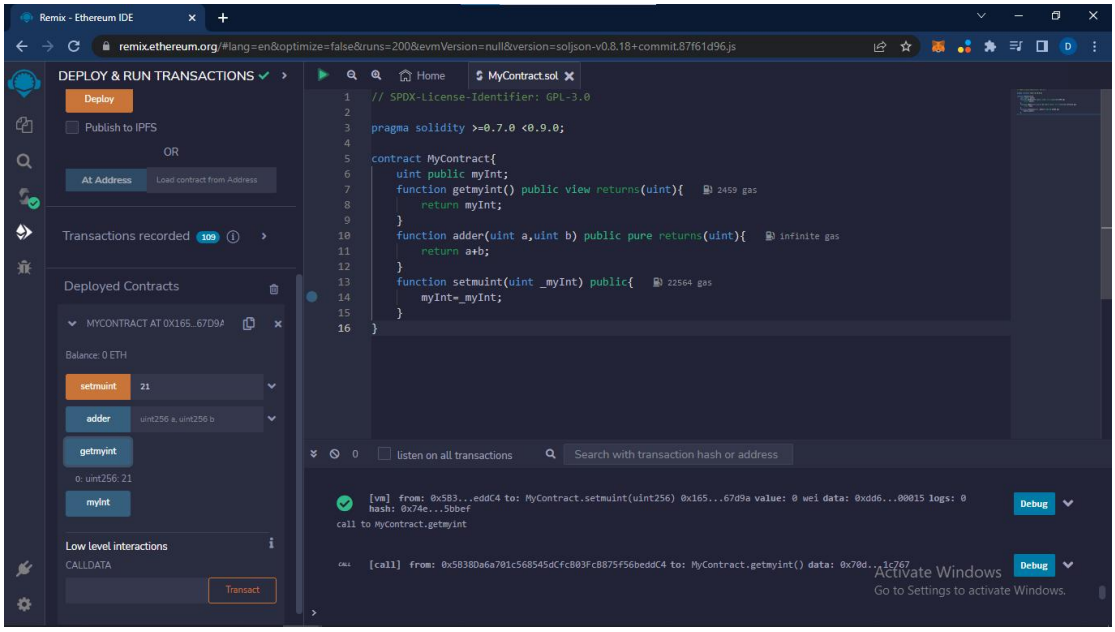
Reading and writing functions:
getter function



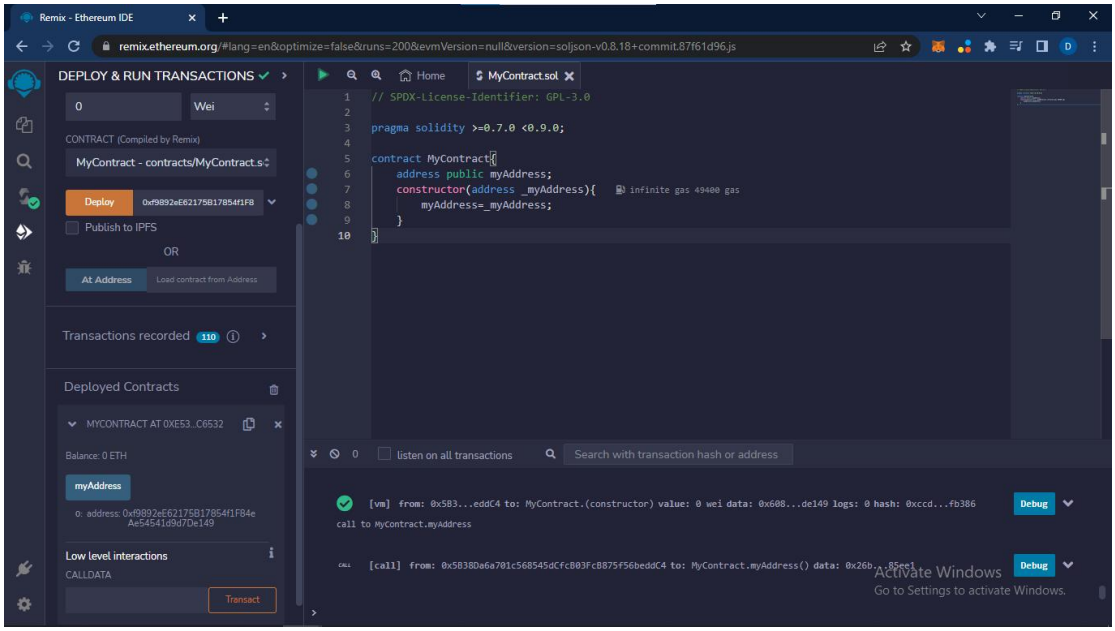
Another 2 types of reading functions are view and pure
view can read storage variable or a variable with global scope. Pure function can read only local variable.



Function for writing. This function will be in orange color.

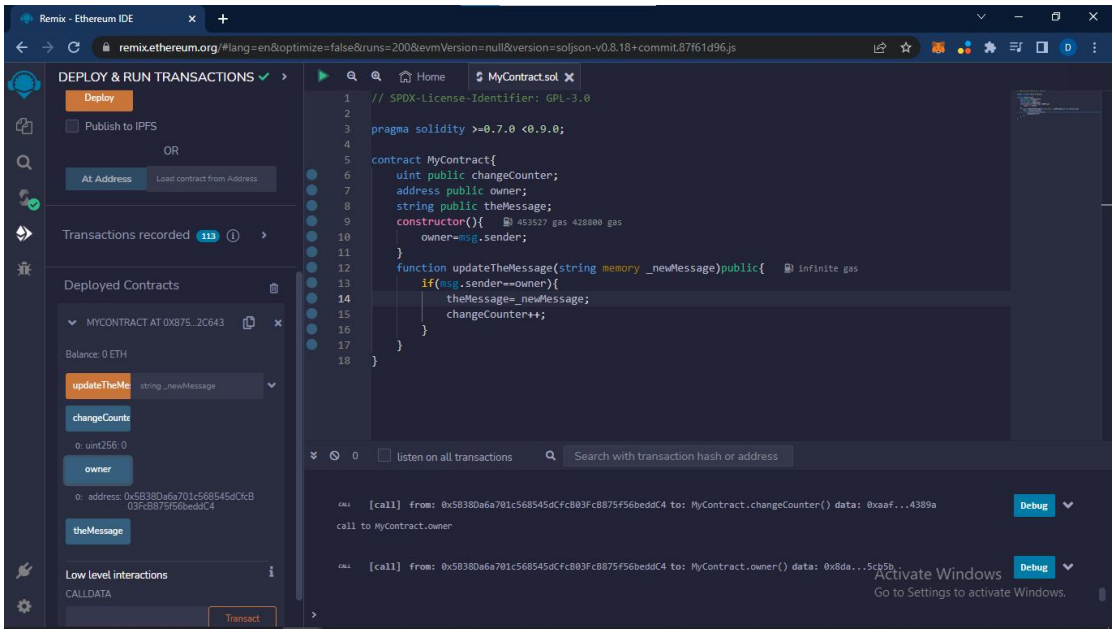


Constructor – will be called only once
Normally first address will be 00000 and then you assign. But constructor will assign at creation. So copy paste address near deploy it self. Then use myAddress.

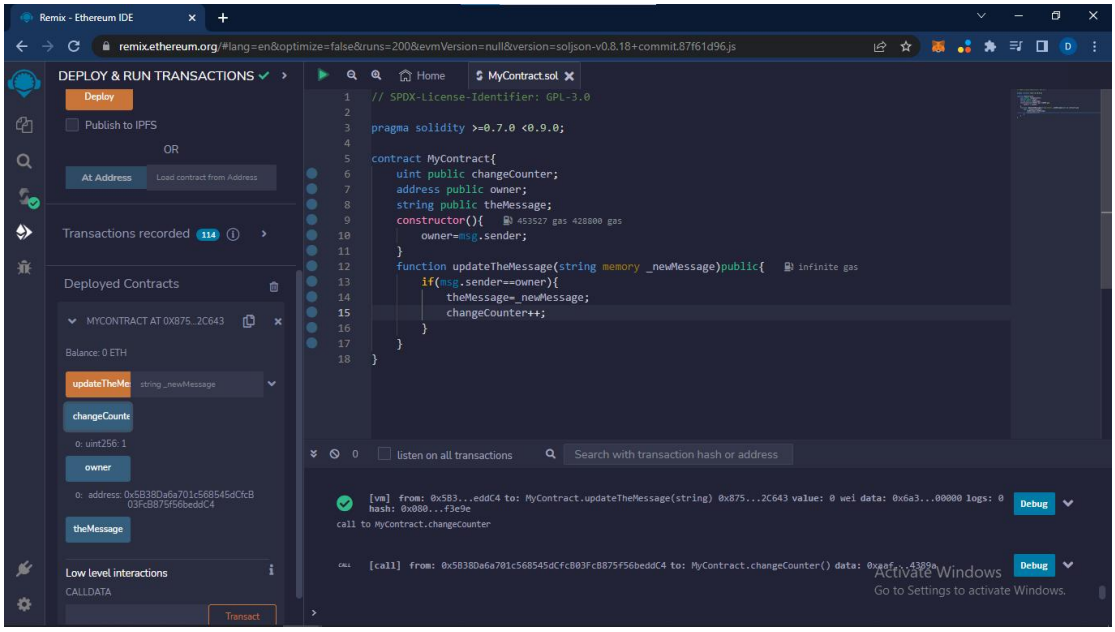


……END……

Blockchain messenger:

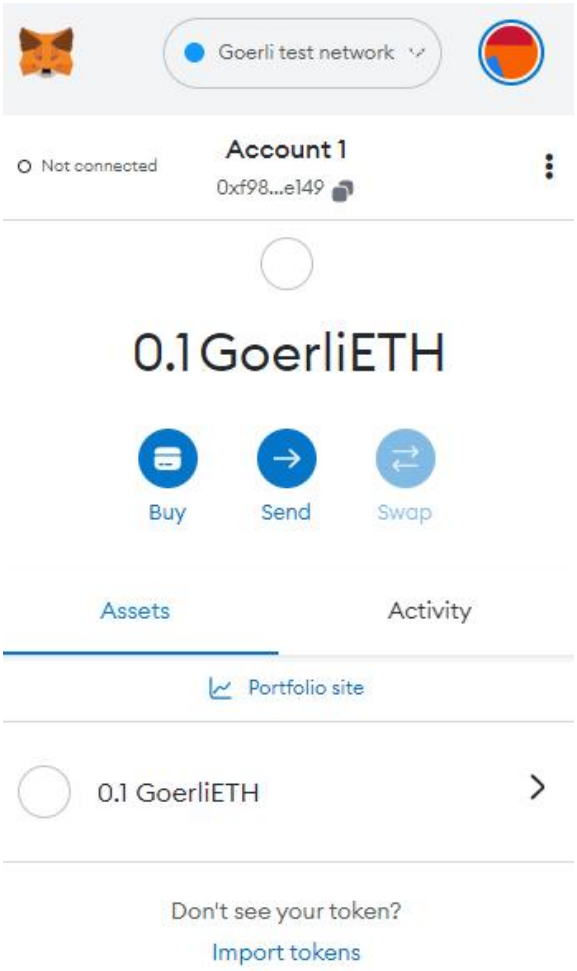


updateTheMessage then see the other values.

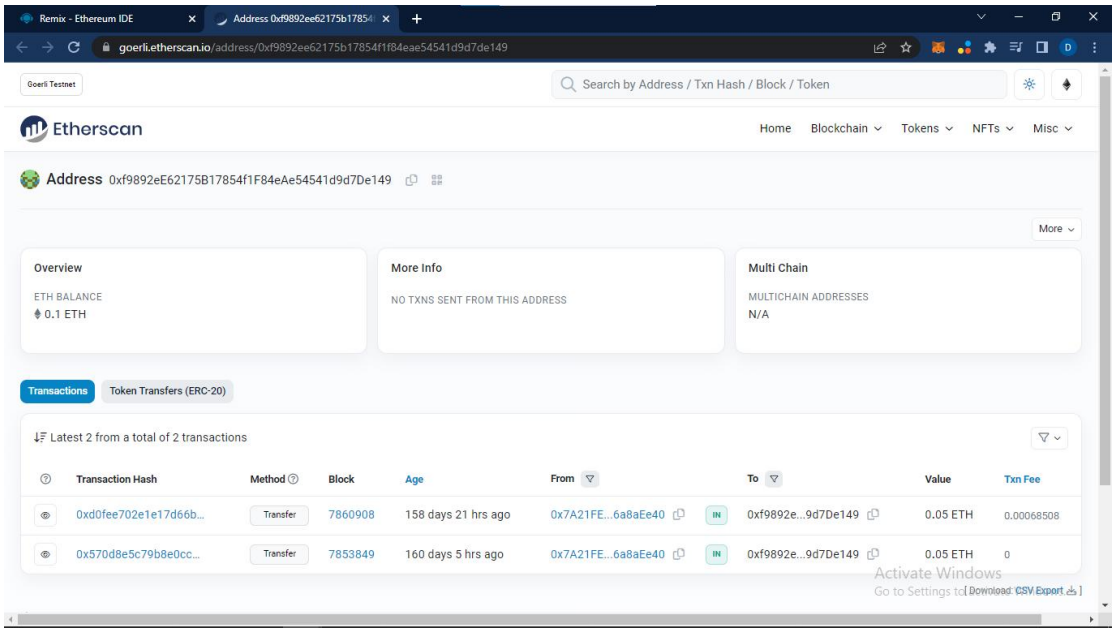


……END……

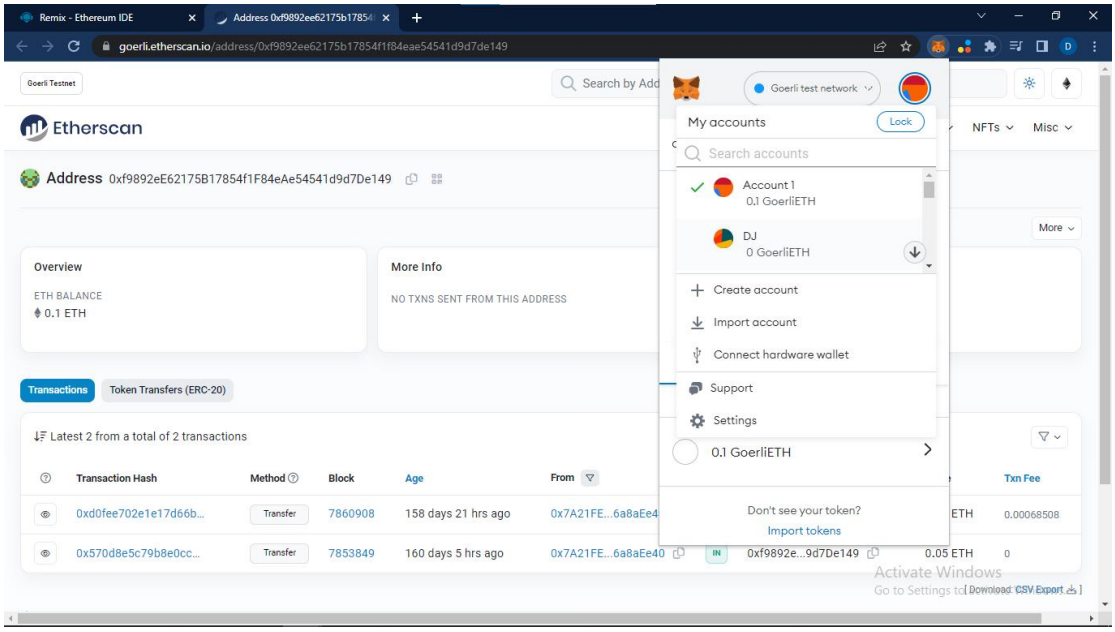
Smart Money deposit and withdrawals:



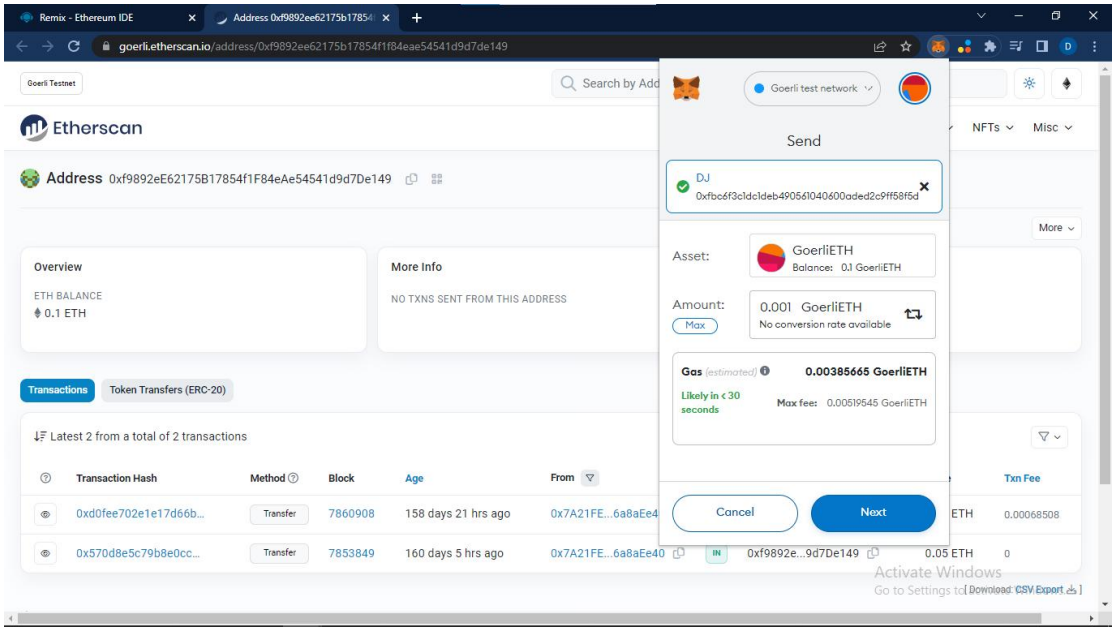
Click sandwich icon in metamask and check the transaction on etherscan



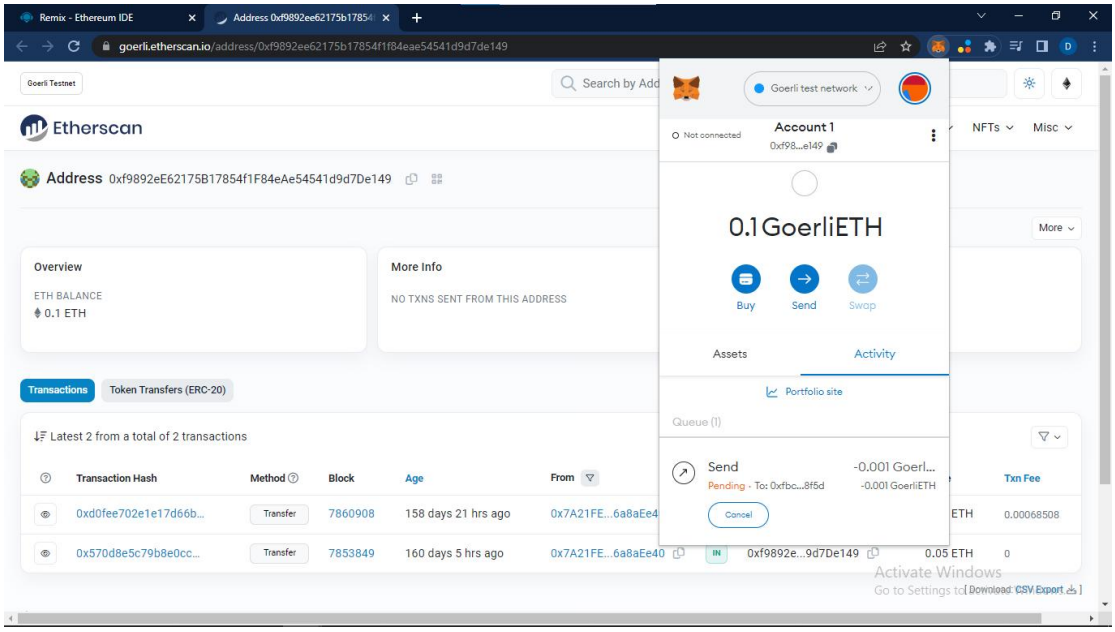
Now to transact from one account to another:



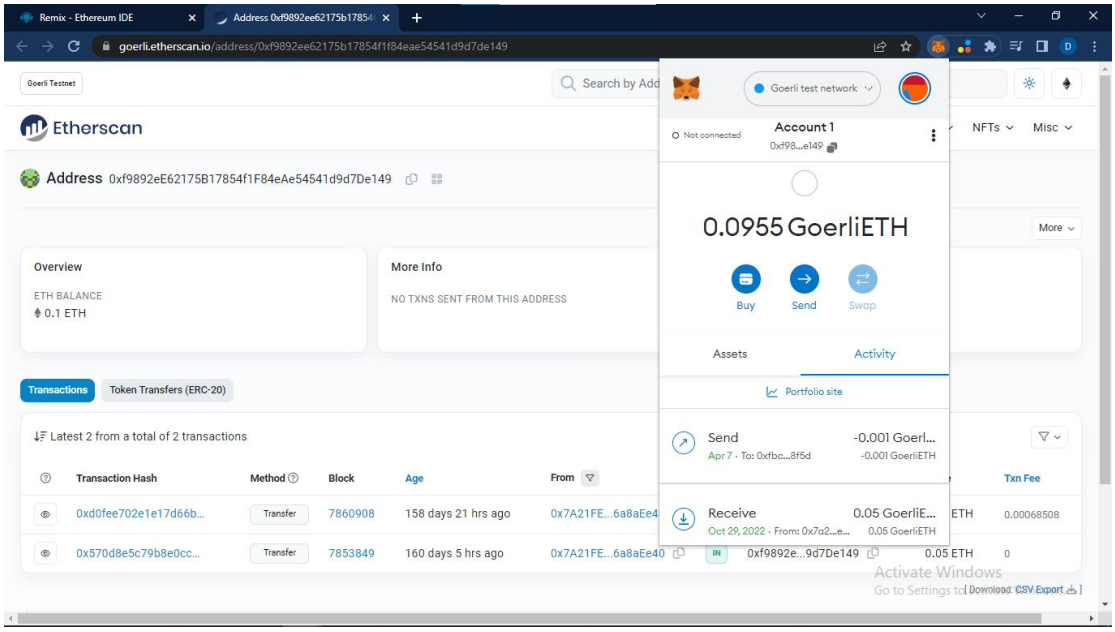
click send button-add account2 address and transact .001 ethers



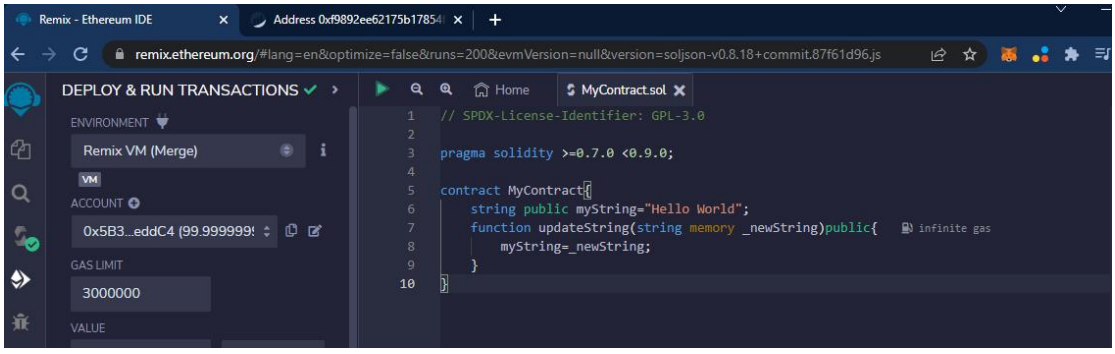
Transaction pending to be added in blockchain:

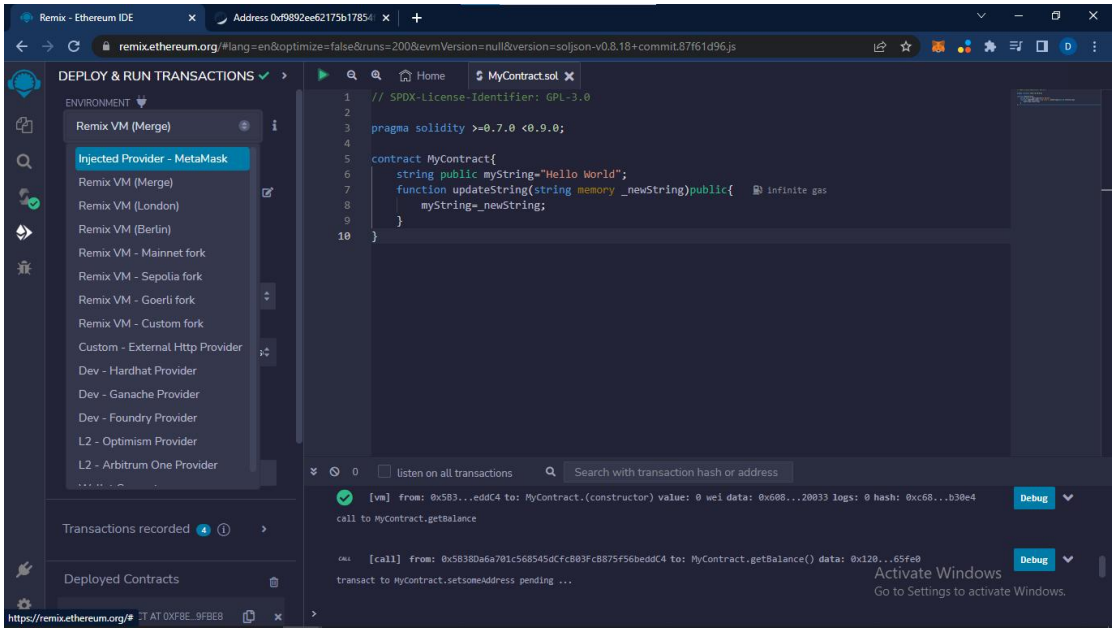


Now added to blockchain

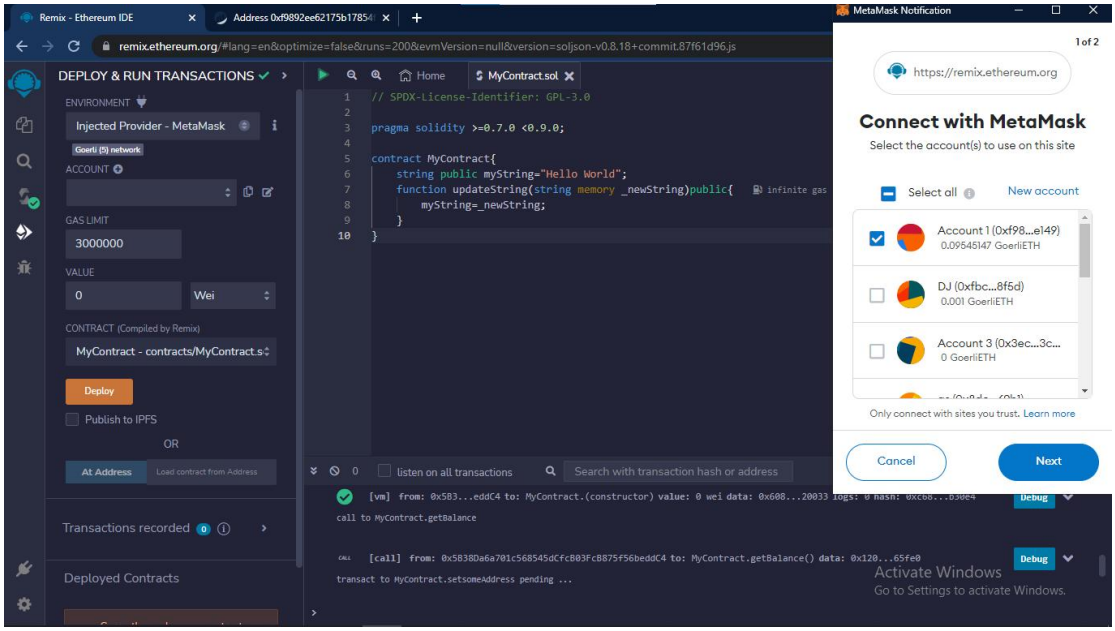


Remix and the Injected Web3 Provider:
Create a new smart contract:



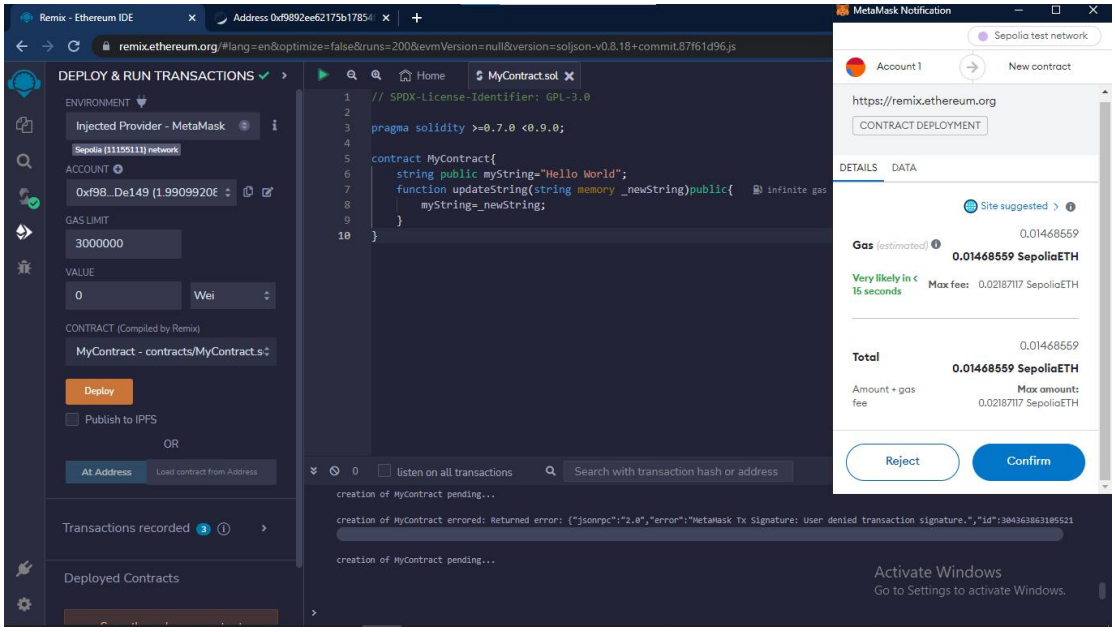


- Rightclick metamask- manage extension – developer mode on – update
- Reload
- Now you can connect with injected web3.

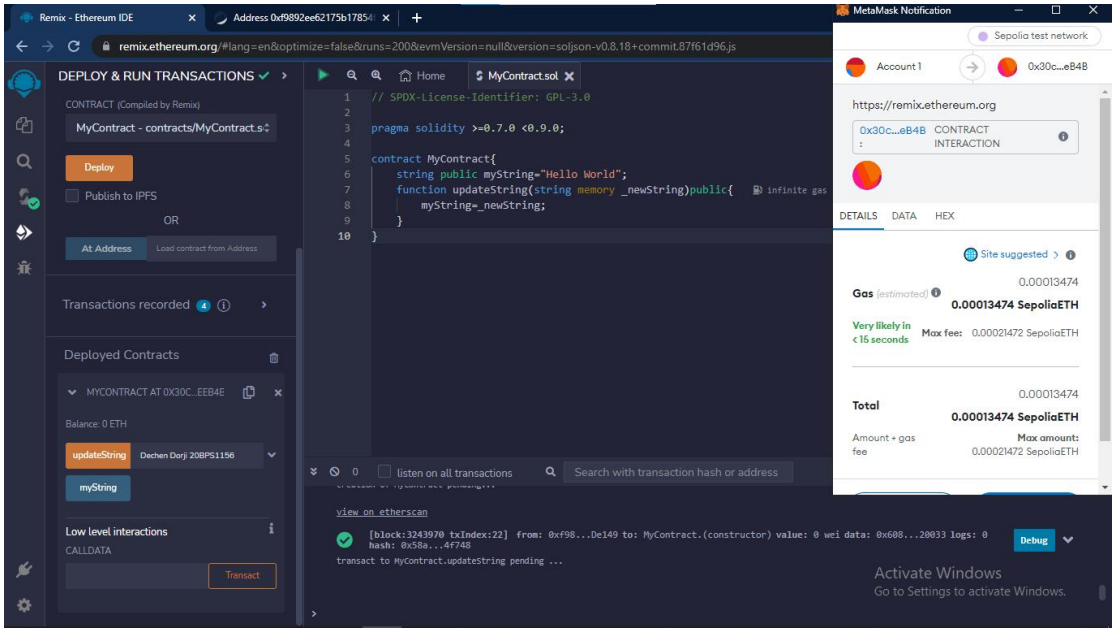


Select account:

- ✓ Next connect
- ✓ Now account address will reflect in metamask
- ✓ Deploy

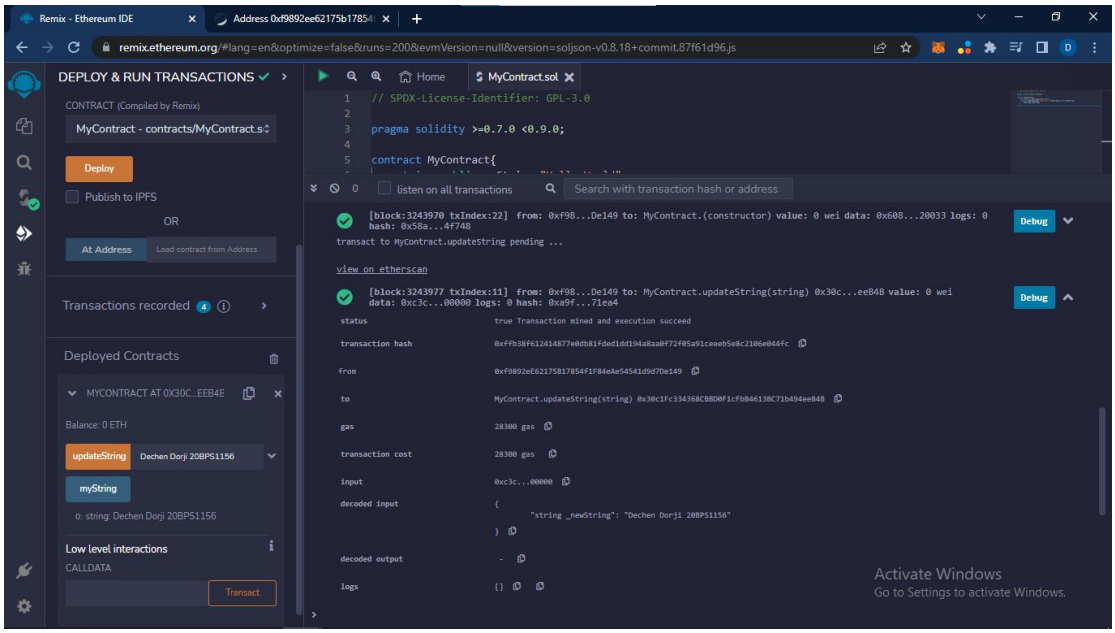


Confirm

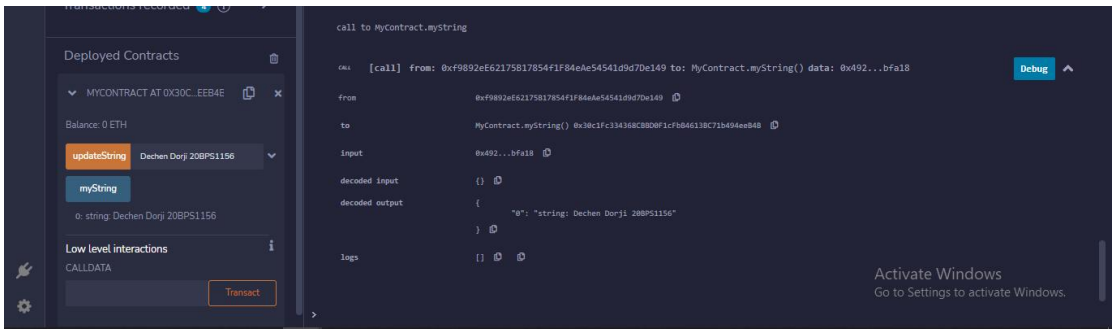


When you click updateString a call is made and the data - Dechen Dorji 20BPS1156 is maintained in the transaction

Block created



Click myString it will not call the metamask as it is read only and not write



…….END…….