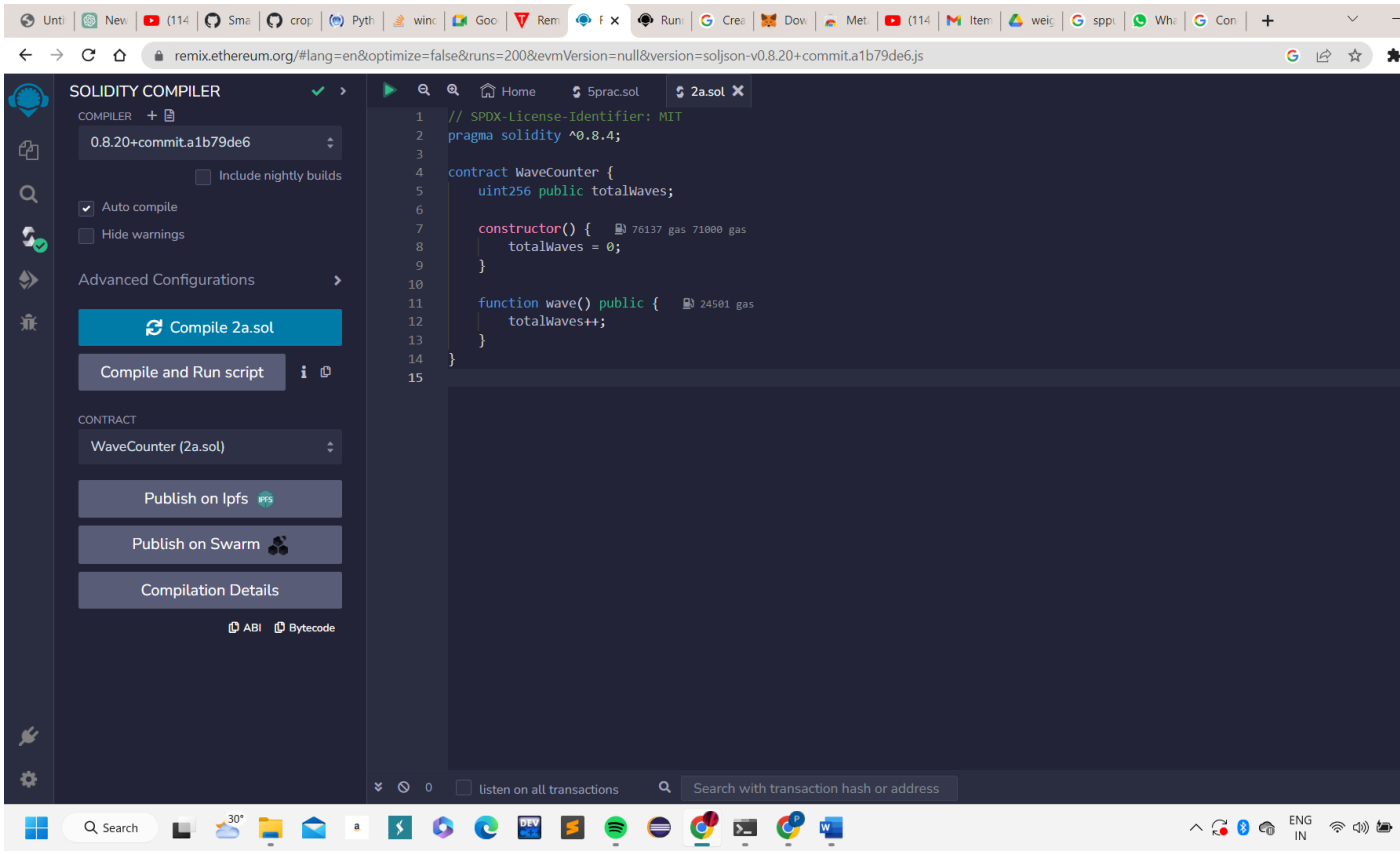
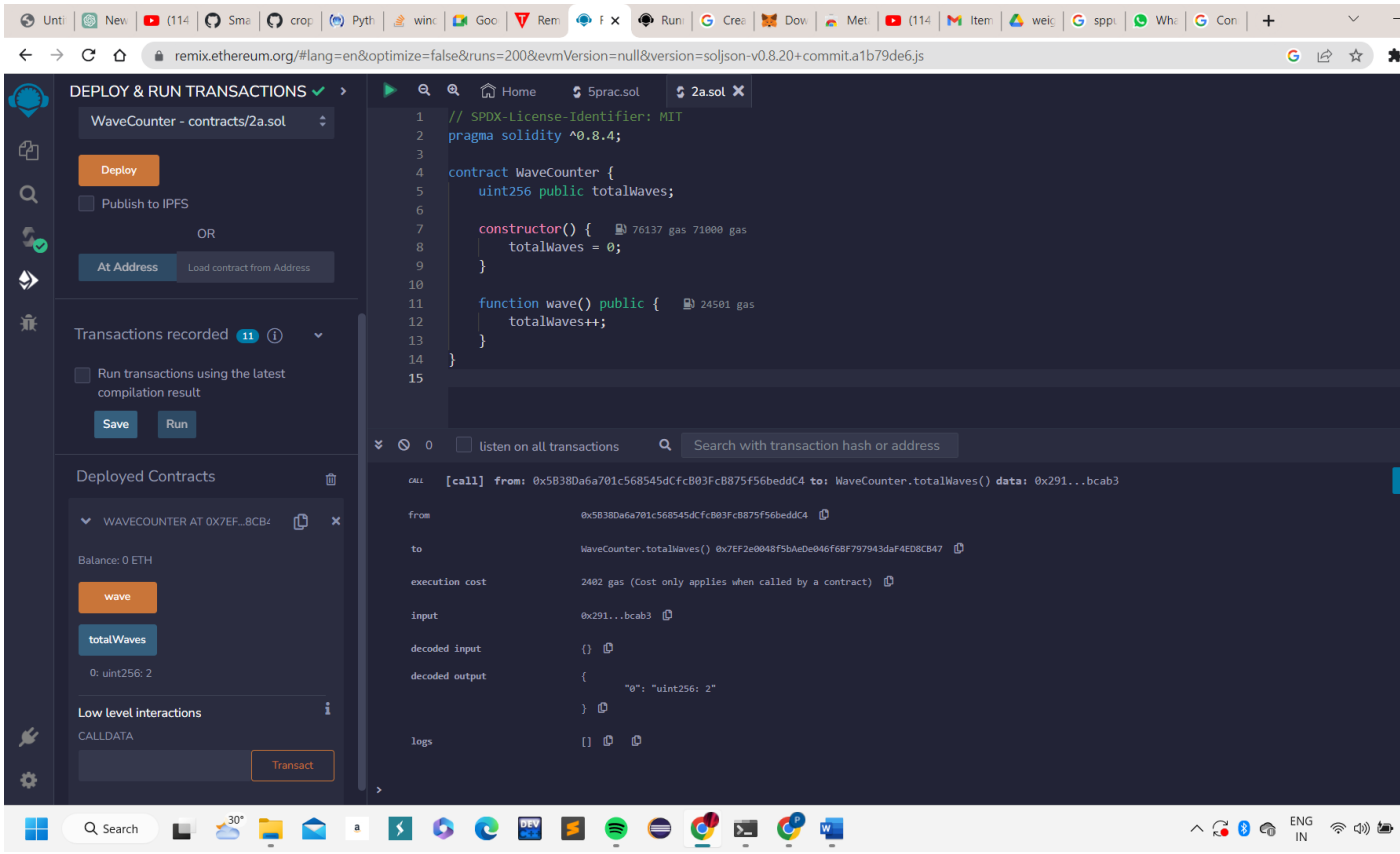
Assignment No.2

1) Create a local Ethereum network using Hardhat or any other tool, build a smart contract that lets you send a 👋(wave) to your contract and keep track of the total # of waves. Compile it to run locally.

2) Connect to any Ethereum walleteg.Metamask. Deploy the contract with testnet. Connect wallet with your webapp. Call the deployed contract through your web app. Then store the wave messages from users in arrays using structs.




remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.20+commit.a1b79de6.js
DEPLOY & RUN TRANSACTIONS
WaveCounter - contracts/2a.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 11
Run transactions using the latest compilation result
Save
Run
Deployed Contracts
WAVECOUNTER AT 0X7EF...8CB4
Balance: 0 ETH
wave
totalWaves
0: uint256: 2
Low level interactions
CALLDATA
Transact
Home
5prac.sol
2a.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.4;
contract WaveCounter {
uint256 public totalWaves;
constructor() { 76137 gas 71000 gas
totalWaves = 0;
}
function wave() public { 24501 gas
totalWaves++;
}
}
listen on all transactions
Search with transaction hash or address
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: WaveCounter.totalWaves() data: 0x291...bcab3
from 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to WaveCounter.totalWaves() 0x7EF2e0048f5bAeDe046f6BF797943daF4ED8CB47
execution cost 2402 gas (Cost only applies when called by a contract)
input 0x291...bcab3
decoded input {}
decoded output { "0": "uint256: 2" }
logs []
Search
ENG
IN