NAAN MUDHALVAN – BLOCKCHAIN

ASSIGNMENT-1

TEAM ID	**NM2023TMID09491**

TEAM LEAD	**AKASH RAJ K**
TEAM MEMBER 1	**LOGESH S**
TEAM MEMBER 2	**AFRIN S**
TEAM MEMBER 3	**ARAVINDAN P**

QUESTION:

Display a zone name as an output. Deploy the code by using remix platform.

Solution:

Source code:

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

contract MyContract {
function getZoneName() public pure returns (string memory) {
return "Zone - 4";
}
}
```

Code explanation:

1. **// SPDX-License-Identifier: MIT:** This is a comment at the beginning of the contract, indicating the license under which the code is released. In this case, it specifies the MIT license, which is a permissive open-source license

2. **pragma solidity ^0.8.0;:** This line specifies the Solidity compiler version to be used for compiling the contract. The caret (^) symbol (^0.8.0) indicates that any compiler version from 0.8.0 up to, but not including, 0.9.0 is acceptable. This ensures that the code is compiled with a compatible compiler version
3. **contract MyContract { ... }:** This defines a smart contract named `MyContract`. In Solidity, a contract is a fundamental building block of Ethereum-based applications.
4. **function getZoneName() public pure returns (string memory) { ... }:** This is a function within the `MyContract` contract. Let's break down the function:

• **function getZoneName():** This declares a function named `getZoneName`. It's a publicly accessible function, meaning it can be called from outside the contract.

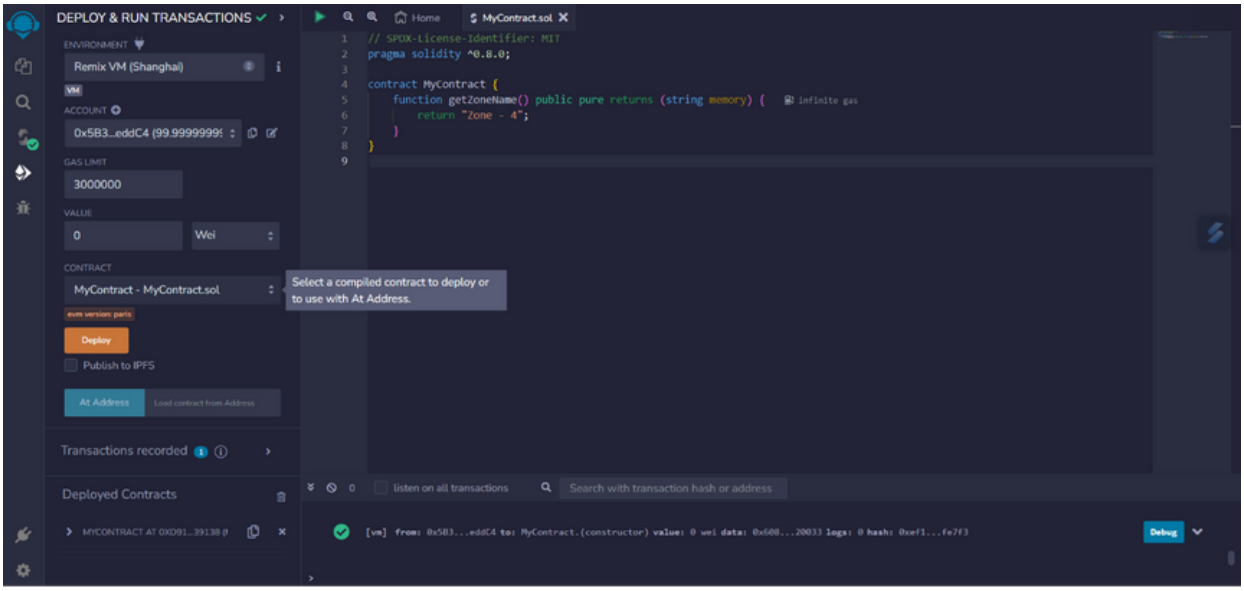
- **public:** This is a visibility specifier, indicating that the function is accessible from outside the contract. It can be called by anyone who interacts with the contract.

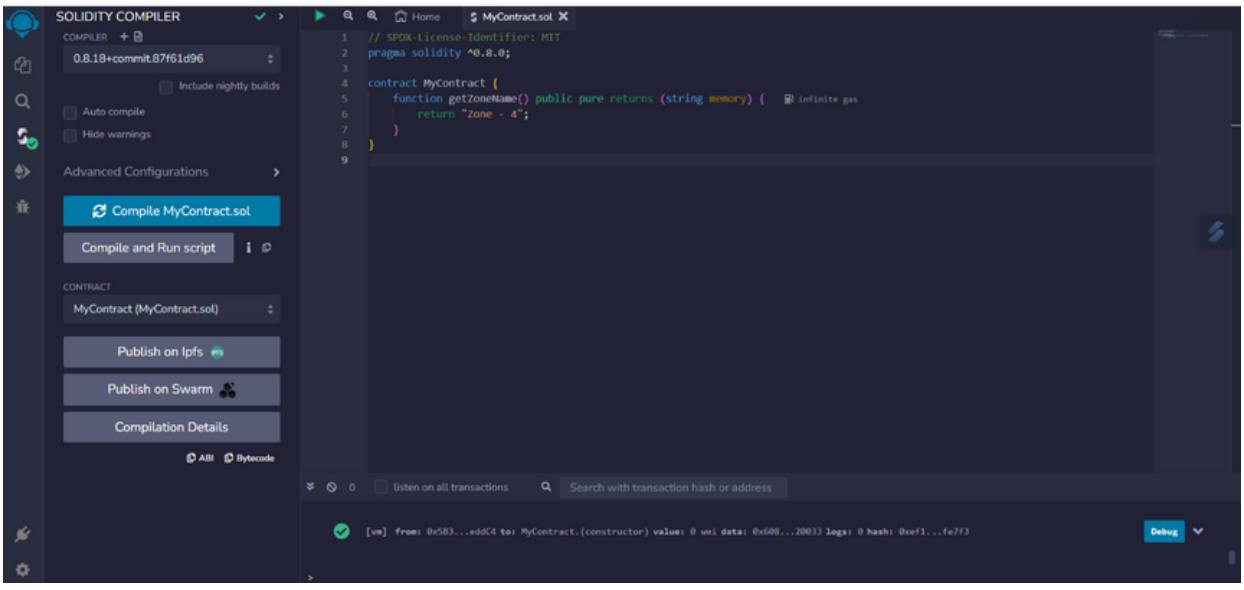
- **pure:** This is a function state mutability specifier. A `pure` function indicates that it doesn't modify the contract's state. It's used for functions that perform computations but don't change any data in the contract. In this case, the function simply returns a value and doesn't modify any contract state.

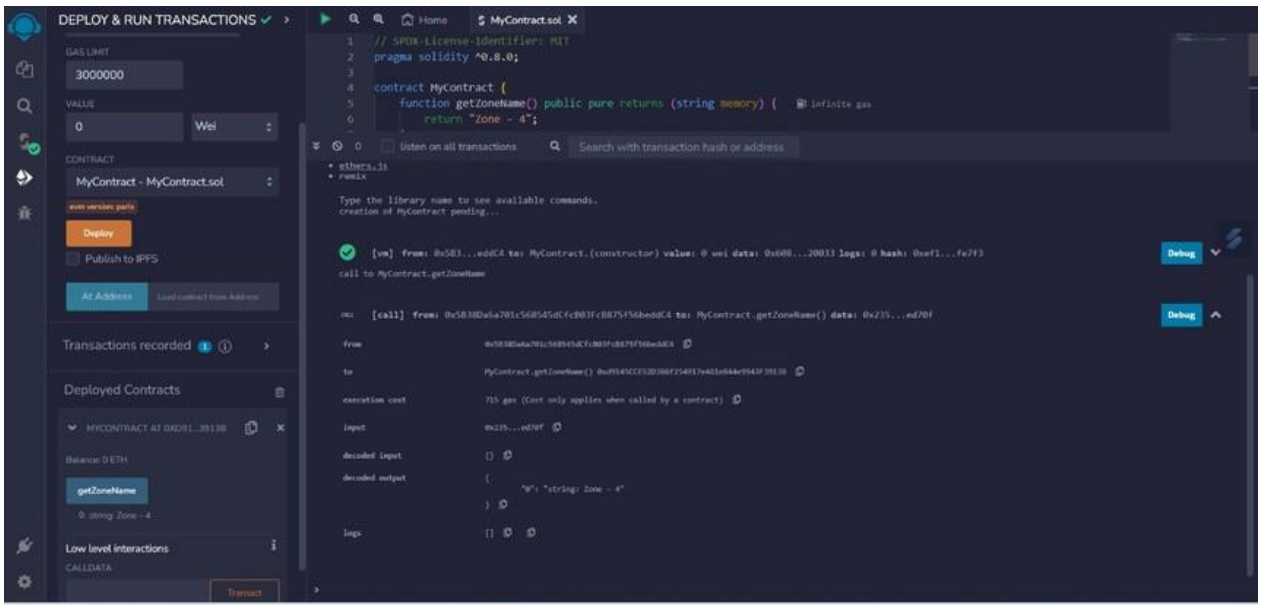
- **returns (string memory):** This specifies the return type of the function. The function returns a dynamic string stored in memory. Solidity functions can return different data types, and in this case, it's returning a string.

- **{ return "Zone - 4"; }:** This is the body of the function. It simply returns the string "Zone - 4."

Screenshots:






DEPLOY & RUN TRANSACTIONS
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
MyContract - MyContract.sol
Deploy
Publish to IPFS
At Address
Transactions recorded
Deployed Contracts
Balance: 0 ETH
getZoneName
0: string: Zone - 4
Low level interactions
CALLDATA
Transact
Home
MyContract.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract MyContract {
function getZoneName() public pure returns (string memory) {
return "Zone - 4";
Infinite gas
listen on all transactions
Search with transaction hash or address
ethers.js
remix
Type the library name to see available commands.
creation of MyContract pending...
[vm] from: 0x5B3...eddC4 to: MyContract.(constructor) value: 0 wei data: 0x608...20033 logs: 0 hash: 0xef1...fe7f3
Debug
call to MyContract.getZoneName
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: MyContract.getZoneName() data: 0x235...ed70f
Debug
from
to
MyContract.getZoneName() 0xef9545CCF52D306F254917e401e644e9943F39138
execution cost
715 gas (Cost only applies when called by a contract)
input
0x235...ed70f
decoded input
{}
decoded output
{
"0": "string: Zone - 4"
}
logs
[]

****