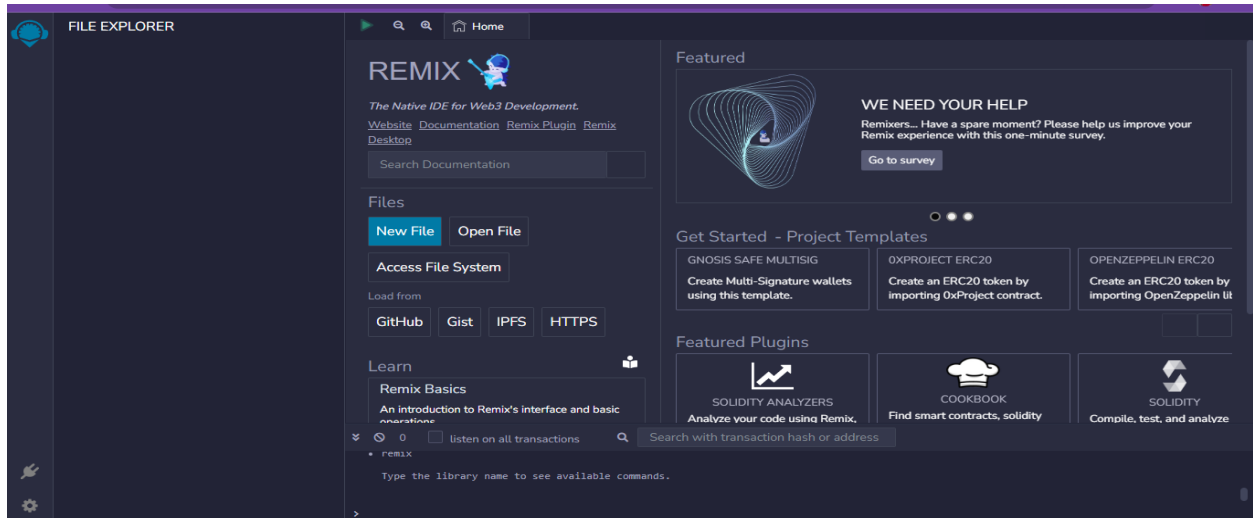
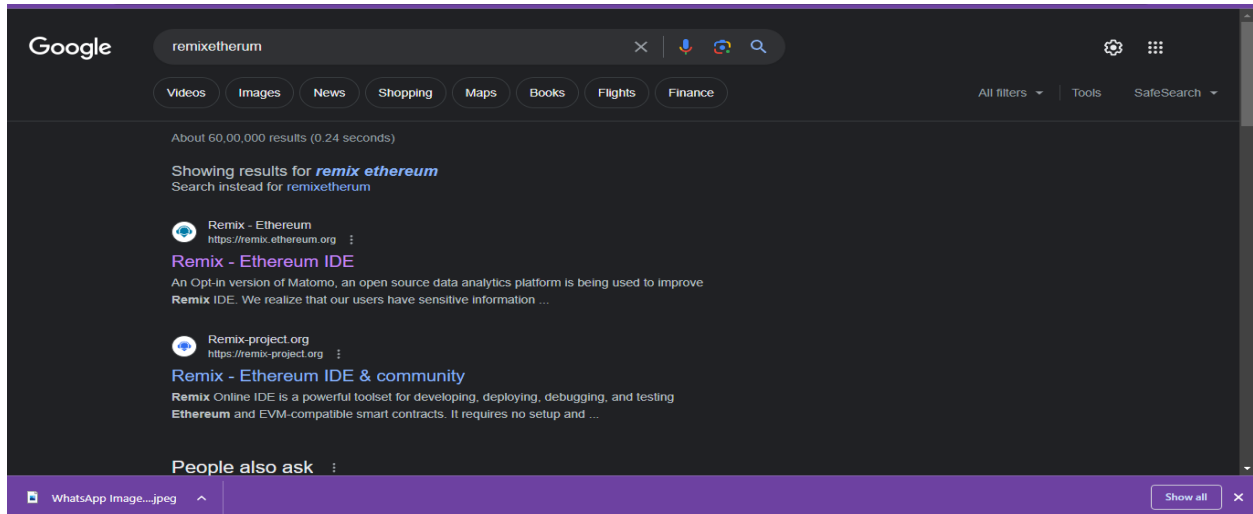
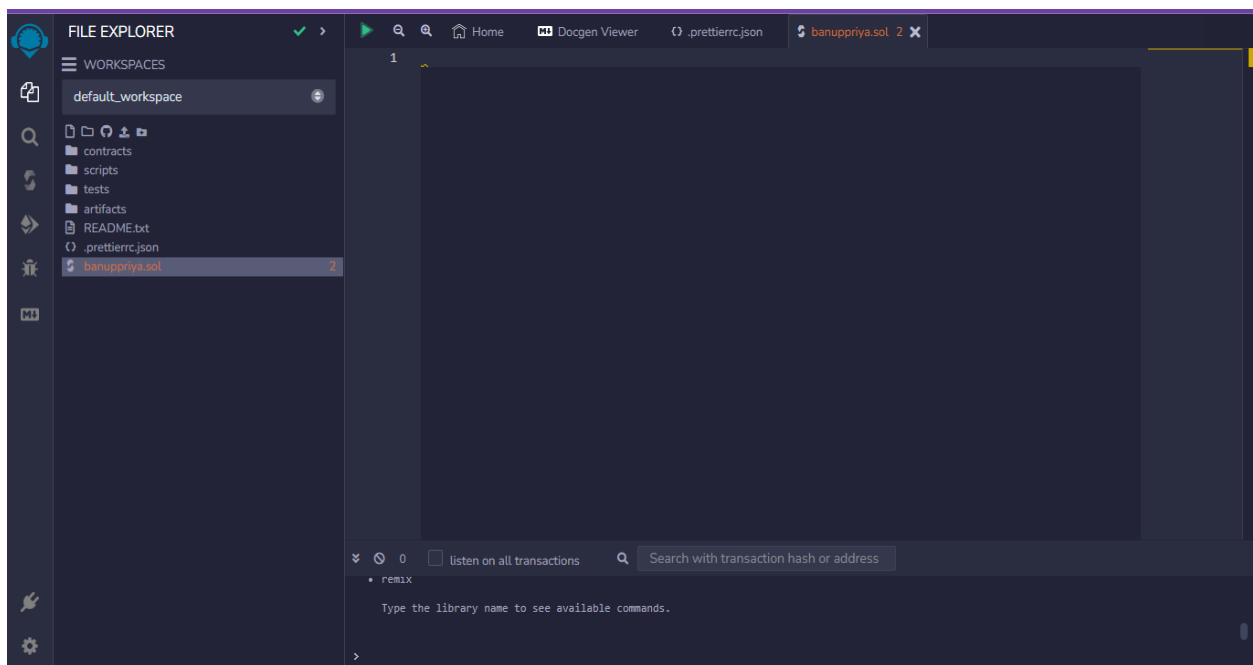
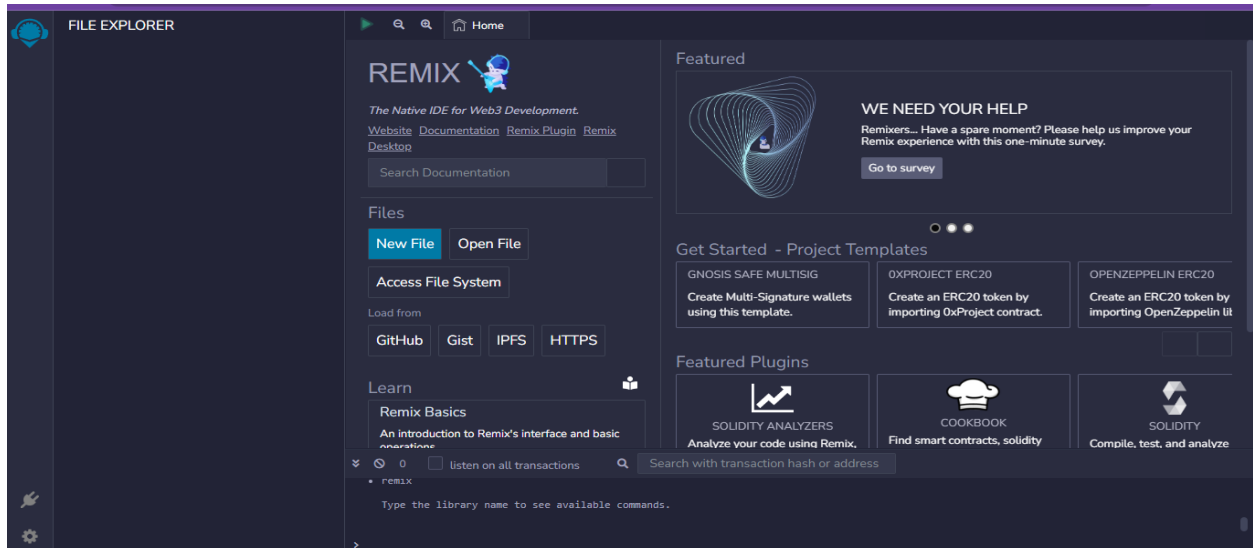


# ASSIGNMENT -1

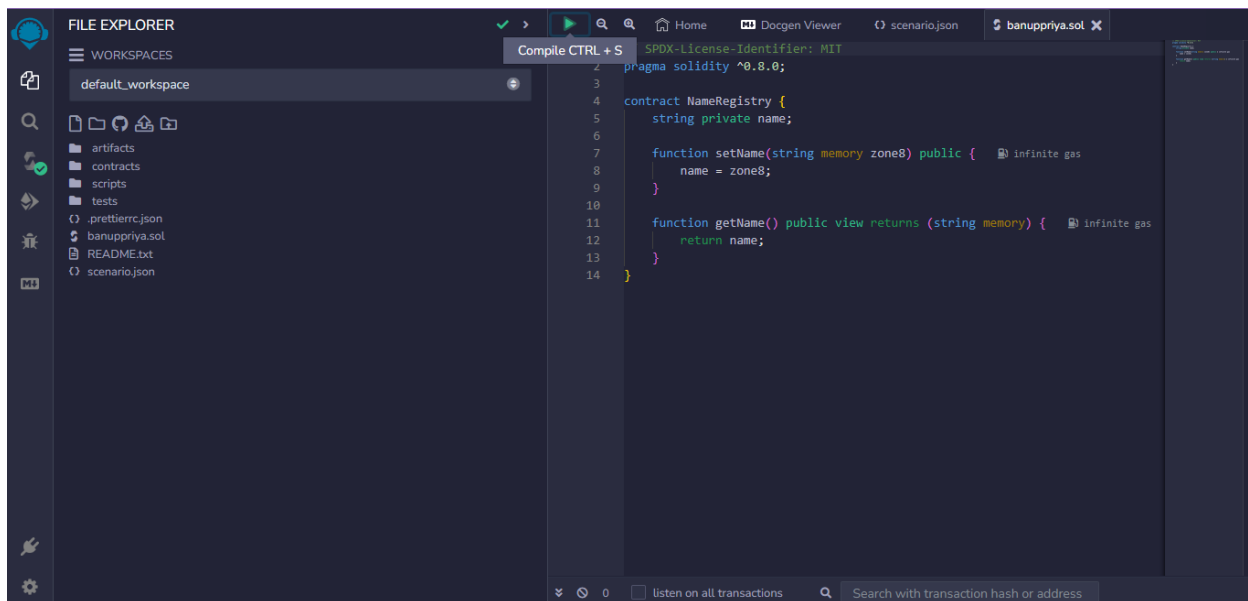
1.Go to the chrome and open remix platform



## 2. Open the remix page and create a new file



3. In that newly created file, create a program to return your string, "Zone name"



The screenshot shows the Visual Studio Code (VS Code) editor interface. On the left, the 'FILE EXPLORER' sidebar displays the 'default\_workspace' with a file tree containing: 'artifacts', 'contracts', 'scripts', 'tests', '.prettierrc.json', 'banupriya.sol', 'README.txt', and 'scenario.json'. The main editor area shows the 'banupriya.sol' file, which contains a Solidity contract named 'NameRegistry'. The contract is written in Solidity 0.8.0 and includes a private string variable 'name' and two public functions: 'setName' and 'getName'. The 'setName' function takes a string parameter 'zone8' and assigns it to 'name'. The 'getName' function returns the value of 'name'. The bottom status bar shows '0' transactions and a search bar for transaction hashes or addresses.

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract NameRegistry {
5     string private name;
6
7     function setName(string memory zone8) public { infinite gas
8         name = zone8;
9     }
10
11     function getName() public view returns (string memory) { infinite gas
12         return name;
13     }
14 }
```

[illegible]

ABI:

```
[
  {
    "inputs": [],
    "name": "getName",
    "outputs": [
      {
        "internalType": "string",
        "name": "",
        "type": "string"
      }
    ],
    "stateMutability": "view",
    "type": "function"
  },
  {
    "inputs": [
      {
        "internalType": "string",
        "name": "zone8",
        "type": "string"
      }
    ]
  }
]
```

```

    }

],

"name": "setName",

"outputs": [],

"stateMutability": "nonpayable",

"type": "function"

}

]]

```

## 5.Finally Deploy it to display the output

The screenshot displays the Remix IDE interface during the deployment of a Solidity contract. The left sidebar shows the 'DEPLOY & RUN TRANSACTIONS' panel. The 'VALUE' field is set to '0' with the unit 'Wei'. The 'CONTRACT' dropdown is set to 'NameRegistry - banupriya.sol'. The 'evm version' is set to 'paris'. The 'Deploy' button is highlighted. Below it, there is a 'Publish to IPFS' checkbox and an 'At Address' button. The 'Transactions recorded' section shows one transaction. The 'Deployed Contracts' section shows the 'NAMERegistry AT 0xD91...39138 (MEMORY)' contract. The 'Low level interactions' section shows the 'CALLDATA' field. The main workspace displays the transaction details for the deployment, including the transaction hash, block hash, block number, contract address, from address, to address, gas, transaction cost, execution cost, input, and decoded input.

**DEPLOY & RUN TRANSACTIONS**

VALUE: 0 Wei

CONTRACT: NameRegistry - banupriya.sol

evm version: paris

Deploy

Publish to IPFS

At Address Load contract from Address

Transactions recorded: 1

Deployed Contracts

NAMERegistry AT 0xD91...39138 (MEMORY)

Balance: 0 ETH

setName string zone8

getName

Low level interactions

CALLDATA

Transaction details:

- status: true Transaction mined and execution succeed
- transaction hash: 0x3d5346fe68fa6f642e36bc47f077bc02112f01fca818af2f736f1a7382947a
- block hash: 0xd19b383064153666c46b3f1d664bc8072e1a5f28d0f7c1978326089ae886f2b
- block number: 2
- contract address: 0xd8b934580fce35a11b58c6073a0e468a2833fab
- from: 0x5838D6a701c568545dcfb03fc8875f56beddC4
- to: NameRegistry.(constructor)
- gas: 472485 gas
- transaction cost: 419962 gas
- execution cost: 332368 gas
- input: 0x688...28033
- decoded input: {}