Practical Journal

BLOCKCHAIN
DEEP LEARNING
ROBOTIC PROCESS AUTOMATION

A Practical Report

Submitted in partial fulfilment of the
Requirements for the award of the Degree of

MASTER OF SCIENCE (INFORMATION TECHNOLOGY)
Part II – SEM IV

Submitted by

TEJAS GORULE

Department of Information Technology

Shri Vile Parle Kelavani Mandal's

USHA PRAVIN GANDHI COLLEGE OF ARTS, SCIENCE AND COMMERCE
(Autonomous Affiliated to University of Mumbai)
NAAC Reaccredited 'A+' Grade
MUMBAI, 400056

2024-25

Practical Journal

BLOCKCHAIN
DEEP LEARNING
ROBOTIC PROCESS AUTOMATION

A Practical Report

Submitted in partial fulfilment of the
Requirements for the award of the Degree
of

MASTER OF SCIENCE (INFORMATION TECHNOLOGY)
Part II – SEM IV

Submitted by

TEJAS GORULE– 53004230006

Department of Information Technology

Shri Vile Parle Kelavani Mandal's

USHA PRAVIN GANDHI COLLEGE OF ARTS, SCIENCE AND COMMERCE
(Autonomous Affiliated to University of Mumbai)
NAAC Reaccredited 'A+' Grade
MUMBAI,

400056

2024-25

INDEX

10	Compile and test smart contracts on a testing framework using the Ethereum Virtual Machine (EVM).	27/02/25	71	
11	Demonstrate the use of Bitcoin Core API.	03/03/25	72	
12	Create your own blockchain and demonstrate its use.	06/03/25	78	

PRACTICAL 1

Aim:-Write a simple client class that generates the private and public keys by using the built-in Python RSA algorithm and test it.

Code:-

```
import hashlib
import random
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5

class Client:
  def __init__(self):

    random = Random.new().read
    self._private_key = RSA.generate(1024, random)
    self._public_key = self._private_key.publickey()
    self._signer = PKCS1_v1_5.new(self._private_key)
  @property
  def identity(self):
    return binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')

client = Client()
print ("sender ",client.identity)
```

Output:-

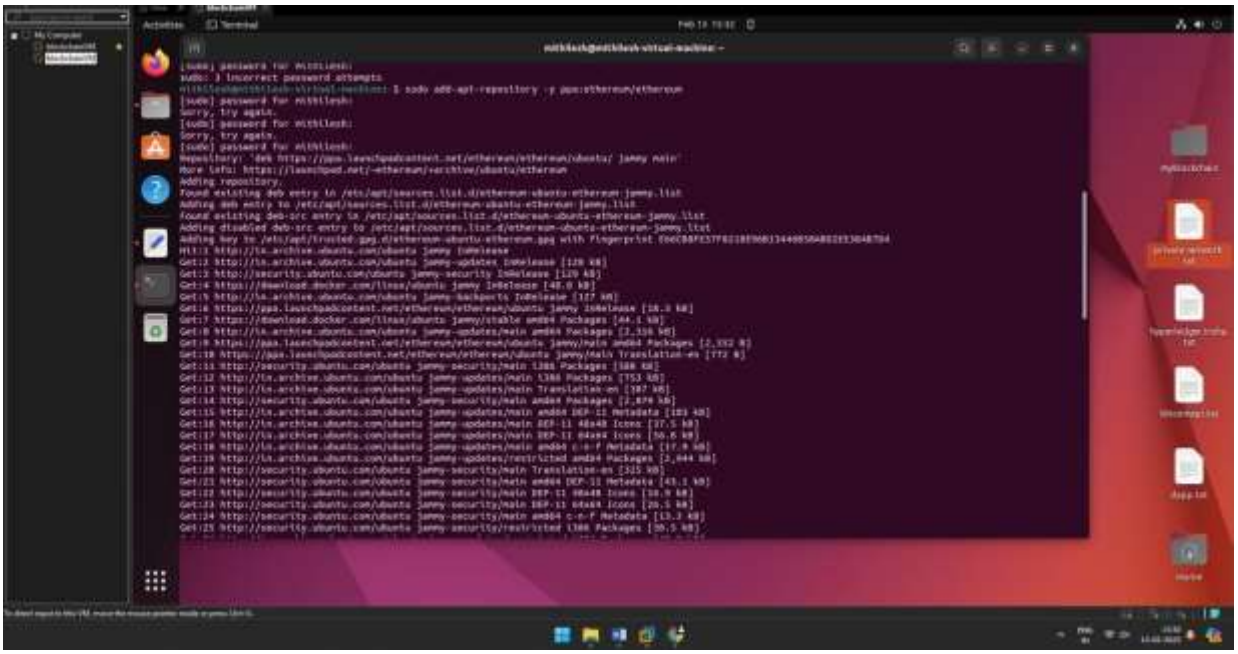
```
sender  30819f300d06092a864886f70d010101050003818d0030818902818100cb786eb7f9418677fcdcb7474d9444fdba6e51c172ee890f562e0d06e75b7054f28b06adbf45a62b52a5d94
a203e063ff48633edba8a7a3bafbf571603892d483d58650647ff8861fc58680782f1f9a428c3911c5b77b824ce2a484215371bf4567b53170a6f7f98010e067ad88cfda1650e63555da80984
8ab8284737863e37020301 0001
```

PRACTICAL 2

Aim:Setting up Ethereum network by using Geth command line interface.(INSTALLATION)
Code:-
Install on Ubuntu via PPAs
The easiest way to install go-ethereum on Ubuntu-based distributions is with the built-in launchpad PPAs (Personal Package Archives). We provide a single PPA repository that contains both our stable and development releases for Ubuntu versions trusty, xenial, zesty and artful.
linux:
To enable our launchpad repository run:
Step 1: open new terminal
Step 2: on terminal type this command
sudo add-apt-repository -y ppa:ethereum/ethereum
#if above command gives error then run
#sudo apt-get install --reinstall ca-certificates

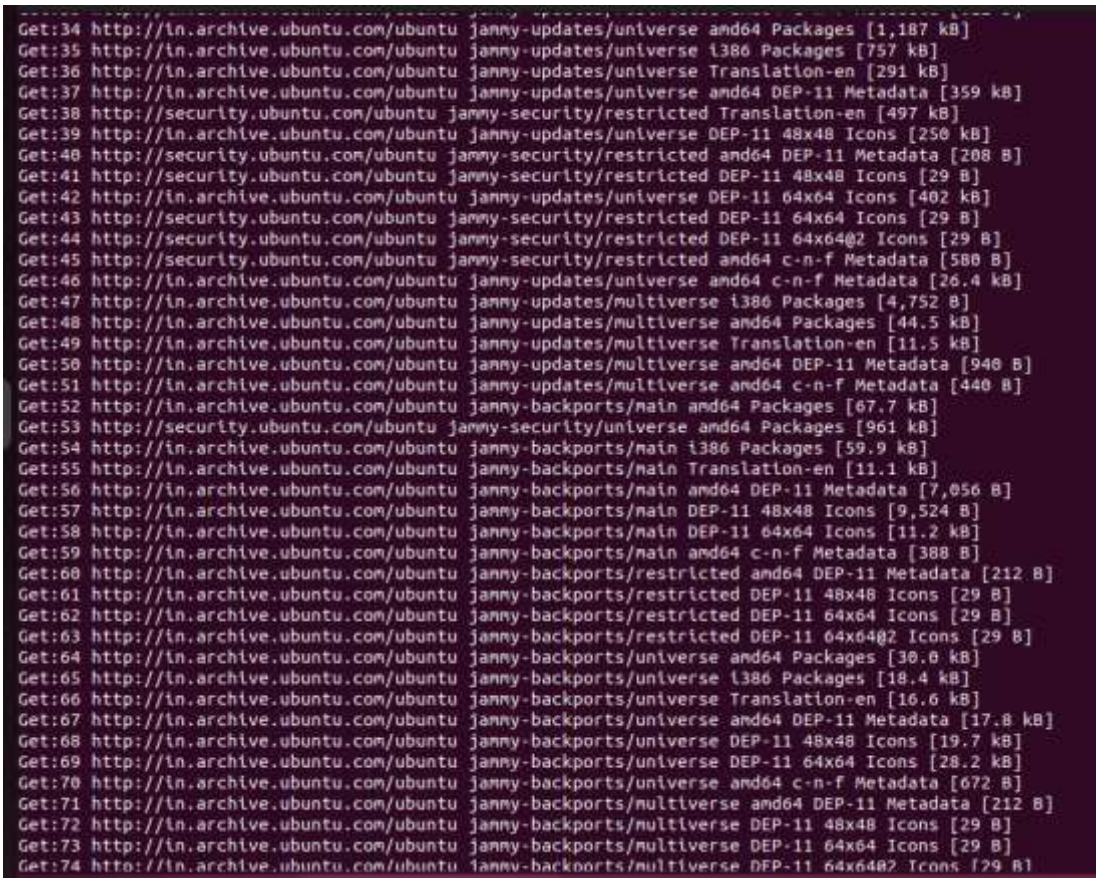
Step 3: install the stable version of go-ethereum:
sudo apt-get update
sudo apt-get install ethereum

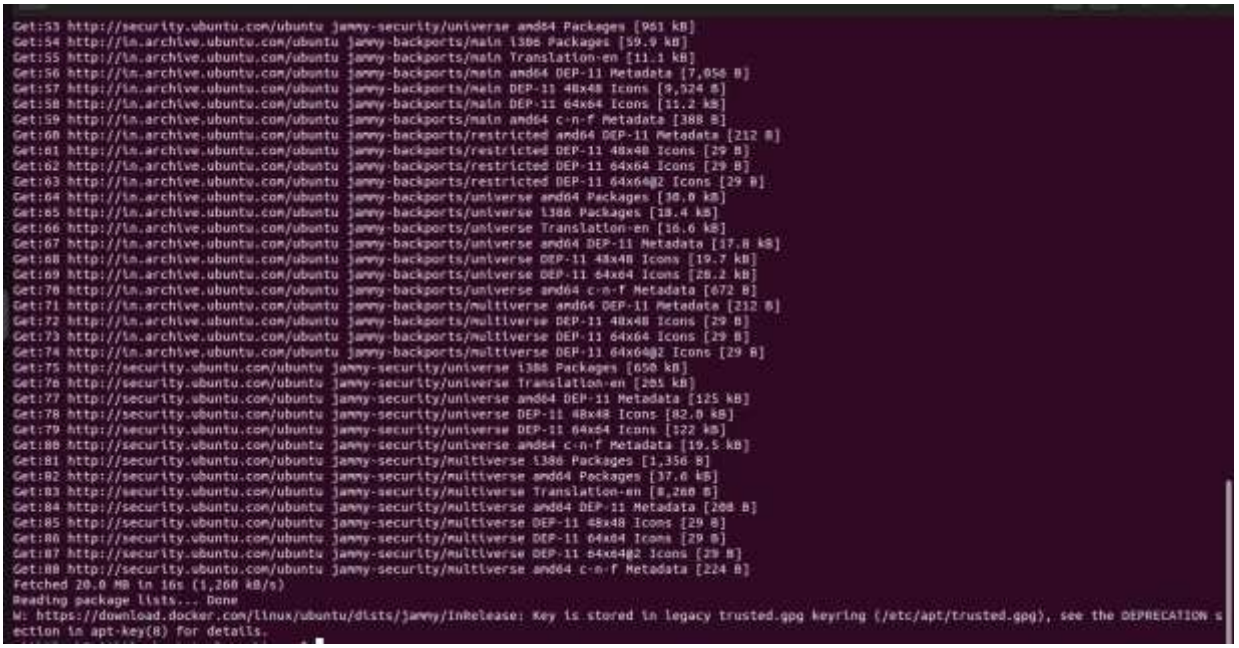
linux:
To enable our launchpad repository run:
Step 1: open new terminal
Step 2: on terminal type this command
sudo add-apt-repository -y ppa:ethereum/ethereum
#if above command gives error then run
#sudo apt-get install --reinstall ca-certificates



```
[sudo] password for mithilesh:
sudo: 3 incorrect password attempts
mithilesh@mithilesh-virtual-machine:~$ sudo add-apt-repository -y ppa:ethereum/ethereum
[sudo] password for mithilesh:
Sorry, try again.
[sudo] password for mithilesh:
Sorry, try again.
[sudo] password for mithilesh:
Repository: 'deb https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu/ jammy main'
More info: https://launchpad.net/~ethereum/+archive/ubuntu/ethereum
Adding repository.
Found existing deb entry in /etc/apt/sources.list.d/ethereum-ubuntu-ethereum-jammy.list
Adding deb entry to /etc/apt/sources.list.d/ethereum-ubuntu-ethereum-jammy.list
Found existing deb-src entry in /etc/apt/sources.list.d/ethereum-ubuntu-ethereum-jammy.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/ethereum-ubuntu-ethereum-jammy.list
Adding key to /etc/apt/trusted.gpg.d/ethereum-ubuntu-ethereum.gpg with fingerprint E66CB8FE57F8218E96B13446B50ABD2EE38487D4
Hit:1 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:4 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:6 https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu jammy InRelease [18.3 kB]
Get:7 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [44.1 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2,316 kB]
Get:9 https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu jammy/main amd64 Packages [2,552 B]
Get:10 https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu jammy/main Translation-en [772 B]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [588 kB]
Get:12 http://in.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [753 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [387 kB]
Get:14 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [2,079 kB]
Get:15 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [103 kB]
Get:16 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icons [37.5 kB]
Get:17 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 64x64 Icons [56.8 kB]
Get:18 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.9 kB]
Get:19 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2,944 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [325 kB]
Get:21 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43.1 kB]
Get:22 http://security.ubuntu.com/ubuntu jammy-security/main DEP-11 48x48 Icons [16.9 kB]
Get:23 http://security.ubuntu.com/ubuntu jammy-security/main DEP-11 64x64 Icons [26.5 kB]
Get:24 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.3 kB]
Get:25 http://security.ubuntu.com/ubuntu jammy-security/restricted i386 Packages [38.5 kB]
```

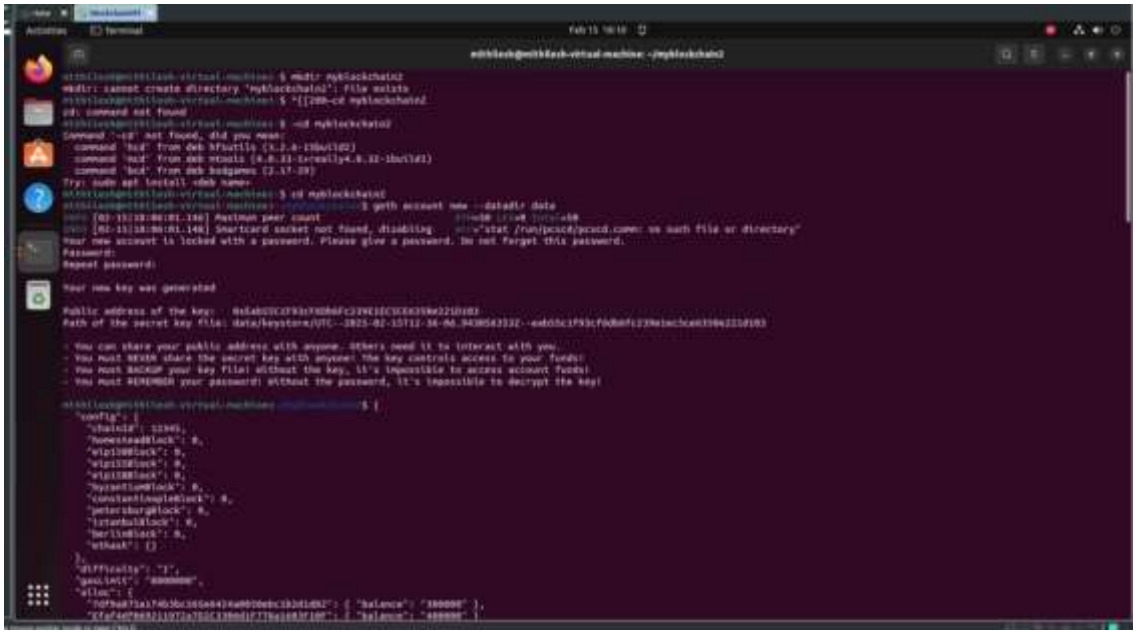
```
Found existing deb-src entry in /etc/apt/sources.list.d/ethereum-ubuntu-ethereum-jammy.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/ethereum-ubuntu-ethereum-jammy.list
Adding key to /etc/apt/trusted.gpg.d/ethereum-ubuntu-ethereum.gpg with fingerprint E66CB8FE57F8218E96B13446B50ABD2EE38487D4
Hit:1 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Get:4 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:6 https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu jammy InRelease [18.3 kB]
Get:7 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [44.1 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2,316 kB]
Get:9 https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu jammy/main amd64 Packages [2,552 B]
Get:10 https://ppa.launchpadcontent.net/ethereum/ethereum/ubuntu jammy/main Translation-en [772 B]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [588 kB]
Get:12 http://in.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [753 kB]
Get:13 http://in.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [387 kB]
Get:14 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [2,079 kB]
Get:15 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [103 kB]
Get:16 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icons [37.5 kB]
Get:17 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 64x64 Icons [56.8 kB]
Get:18 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.9 kB]
Get:19 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2,944 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [325 kB]
Get:21 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [43.1 kB]
Get:22 http://security.ubuntu.com/ubuntu jammy-security/main DEP-11 48x48 Icons [16.9 kB]
Get:23 http://security.ubuntu.com/ubuntu jammy-security/main DEP-11 64x64 Icons [26.5 kB]
Get:24 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.3 kB]
Get:25 http://security.ubuntu.com/ubuntu jammy-security/restricted i386 Packages [38.5 kB]
Get:26 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted i386 Packages [40.3 kB]
Get:27 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [2,839 kB]
Get:28 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [515 kB]
Get:29 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 DEP-11 Metadata [212 B]
Get:30 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted DEP-11 48x48 Icons [29 B]
Get:31 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted DEP-11 64x64 Icons [29 B]
Get:32 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted DEP-11 64x64@2 Icons [29 B]
Get:33 http://in.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [612 B]
Get:34 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1,187 kB]
Get:35 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [757 kB]
Get:36 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [291 kB]
Get:37 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [359 kB]
Get:38 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [497 kB]
```

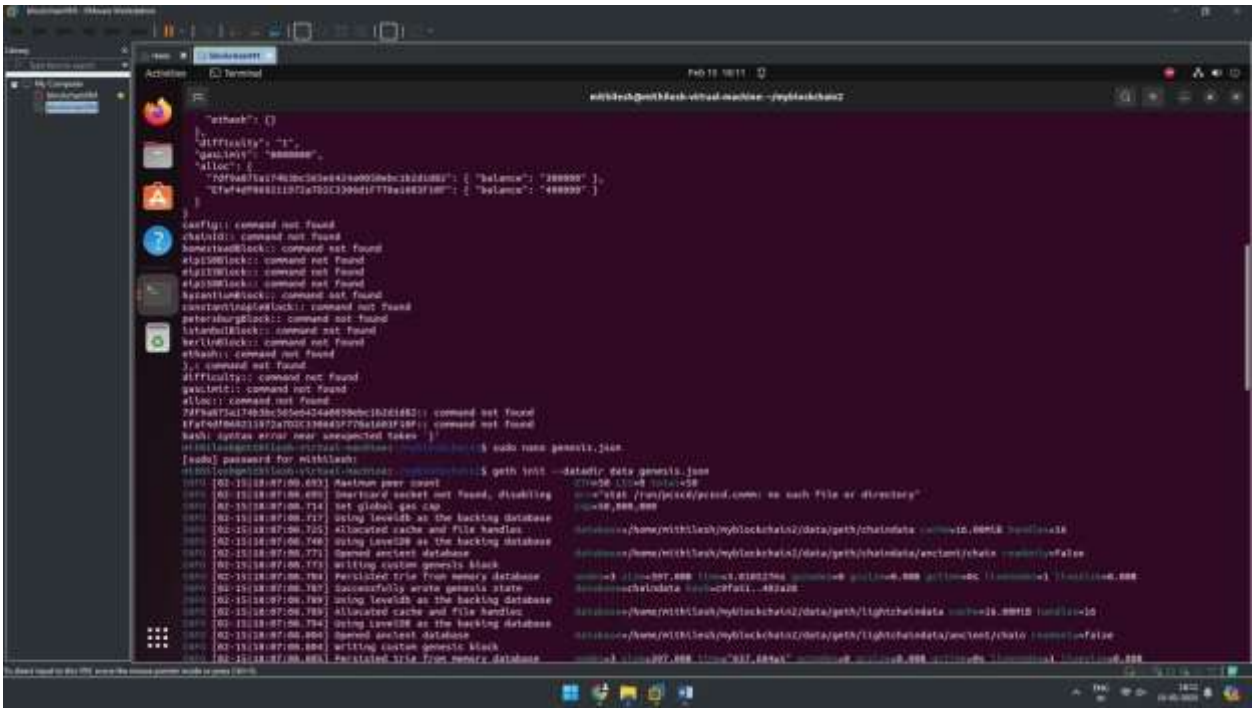

Get:34 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1,187 kB]
Get:35 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [757 kB]
Get:36 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [291 kB]
Get:37 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [359 kB]
Get:38 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [497 kB]
Get:39 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe DEP-11 48x48 Icons [250 kB]
Get:40 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 DEP-11 Metadata [208 B]
Get:41 http://security.ubuntu.com/ubuntu jammy-security/restricted DEP-11 48x48 Icons [29 B]
Get:42 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe DEP-11 64x64 Icons [402 kB]
Get:43 http://security.ubuntu.com/ubuntu jammy-security/restricted DEP-11 64x64 Icons [29 B]
Get:44 http://security.ubuntu.com/ubuntu jammy-security/restricted DEP-11 64x64@2 Icons [29 B]
Get:45 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [580 B]
Get:46 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [26.4 kB]
Get:47 http://in.archive.ubuntu.com/ubuntu jammy-updates/multiverse i386 Packages [4,752 B]
Get:48 http://in.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [44.5 kB]
Get:49 http://in.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [11.5 kB]
Get:50 http://in.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata [940 B]
Get:51 http://in.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [440 B]
Get:52 http://in.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [67.7 kB]
Get:53 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [961 kB]
Get:54 http://in.archive.ubuntu.com/ubuntu jammy-backports/main i386 Packages [59.9 kB]
Get:55 http://in.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [11.1 kB]
Get:56 http://in.archive.ubuntu.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata [7,056 B]
Get:57 http://in.archive.ubuntu.com/ubuntu jammy-backports/main DEP-11 48x48 Icons [9,524 B]
Get:58 http://in.archive.ubuntu.com/ubuntu jammy-backports/main DEP-11 64x64 Icons [11.2 kB]
Get:59 http://in.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [388 B]
Get:60 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 DEP-11 Metadata [212 B]
Get:61 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted DEP-11 48x48 Icons [29 B]
Get:62 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted DEP-11 64x64 Icons [29 B]
Get:63 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted DEP-11 64x64@2 Icons [29 B]
Get:64 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [30.0 kB]
Get:65 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe i386 Packages [18.4 kB]
Get:66 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [16.6 kB]
Get:67 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata [17.8 kB]
Get:68 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe DEP-11 48x48 Icons [19.7 kB]
Get:69 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe DEP-11 64x64 Icons [28.2 kB]
Get:70 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [672 B]
Get:71 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 DEP-11 Metadata [212 B]
Get:72 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse DEP-11 48x48 Icons [29 B]
Get:73 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse DEP-11 64x64 Icons [29 B]
Get:74 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse DEP-11 64x64@2 Icons [29 B]

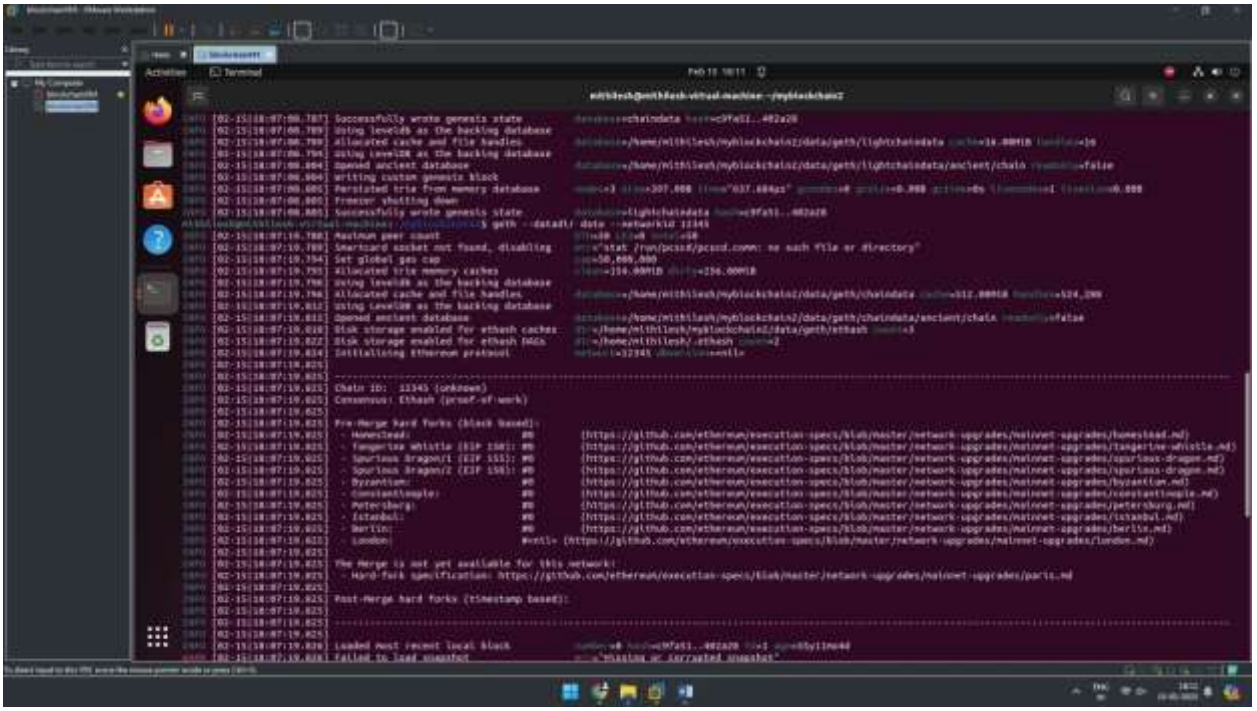

Get:53 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [961 kB]
Get:54 http://in.archive.ubuntu.com/ubuntu jammy-backports/main i386 Packages [59.9 kB]
Get:55 http://in.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [11.1 kB]
Get:56 http://in.archive.ubuntu.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata [7,056 B]
Get:57 http://in.archive.ubuntu.com/ubuntu jammy-backports/main DEP-11 48x48 Icons [9,524 B]
Get:58 http://in.archive.ubuntu.com/ubuntu jammy-backports/main DEP-11 64x64 Icons [11.2 kB]
Get:59 http://in.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [388 B]
Get:60 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 DEP-11 Metadata [212 B]
Get:61 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted DEP-11 48x48 Icons [29 B]
Get:62 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted DEP-11 64x64 Icons [29 B]
Get:63 http://in.archive.ubuntu.com/ubuntu jammy-backports/restricted DEP-11 64x64@2 Icons [29 B]
Get:64 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [30.0 kB]
Get:65 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe i386 Packages [18.4 kB]
Get:66 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [16.6 kB]
Get:67 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata [17.8 kB]
Get:68 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe DEP-11 48x48 Icons [19.7 kB]
Get:69 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe DEP-11 64x64 Icons [28.2 kB]
Get:70 http://in.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [672 B]
Get:71 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 DEP-11 Metadata [212 B]
Get:72 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse DEP-11 48x48 Icons [29 B]
Get:73 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse DEP-11 64x64 Icons [29 B]
Get:74 http://in.archive.ubuntu.com/ubuntu jammy-backports/multiverse DEP-11 64x64@2 Icons [29 B]
Get:75 http://security.ubuntu.com/ubuntu jammy-security/universe i386 Packages [650 kB]
Get:76 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [205 kB]
Get:77 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Metadata [125 kB]
Get:78 http://security.ubuntu.com/ubuntu jammy-security/universe DEP-11 48x48 Icons [82.0 kB]
Get:79 http://security.ubuntu.com/ubuntu jammy-security/universe DEP-11 64x64 Icons [122 kB]
Get:80 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [19.5 kB]
Get:81 http://security.ubuntu.com/ubuntu jammy-security/multiverse i386 Packages [1,356 B]
Get:82 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [17.6 kB]
Get:83 http://security.ubuntu.com/ubuntu jammy-security/multiverse Translation-en [8,288 B]
Get:84 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 DEP-11 Metadata [208 B]
Get:85 http://security.ubuntu.com/ubuntu jammy-security/multiverse DEP-11 48x48 Icons [29 B]
Get:86 http://security.ubuntu.com/ubuntu jammy-security/multiverse DEP-11 64x64 Icons [29 B]
Get:87 http://security.ubuntu.com/ubuntu jammy-security/multiverse DEP-11 64x64@2 Icons [29 B]
Get:88 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [224 B]
Fetched 20.0 MB in 16s (1,260 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/jammy/InRelease: key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.

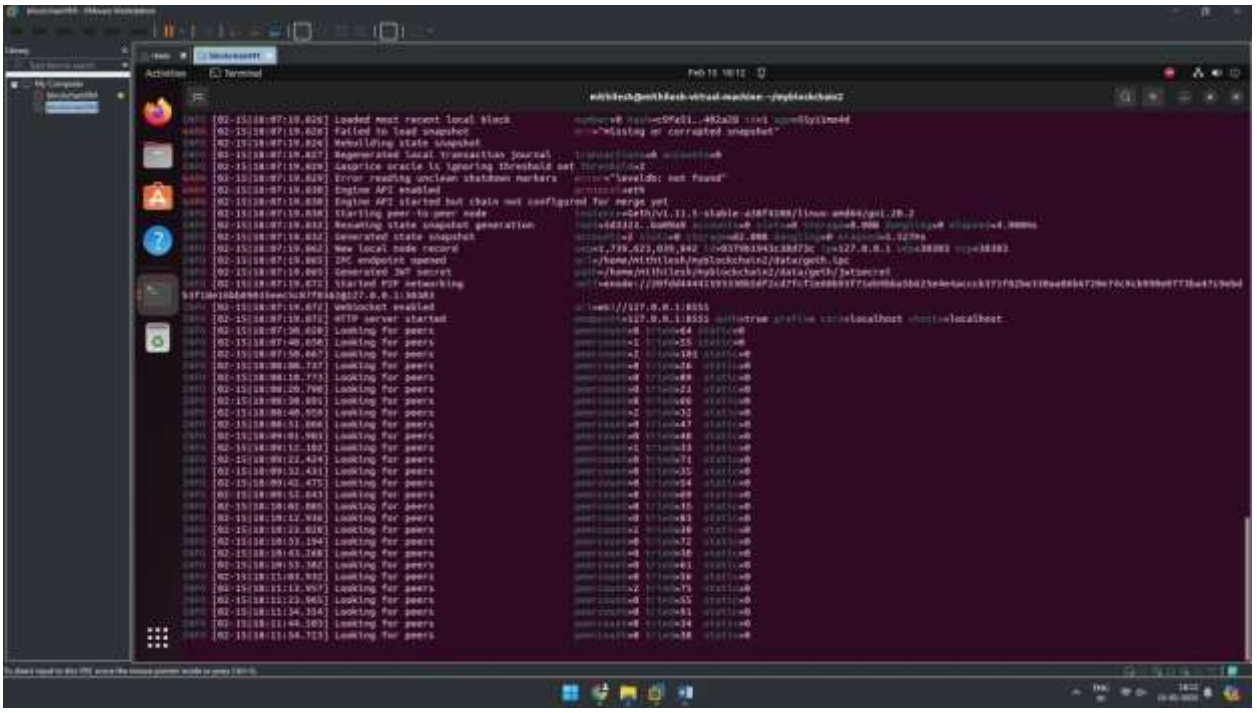
Step 3: install the stable version of go-ethereum:
sudo apt-get update

sudo apt-get install Ethereum



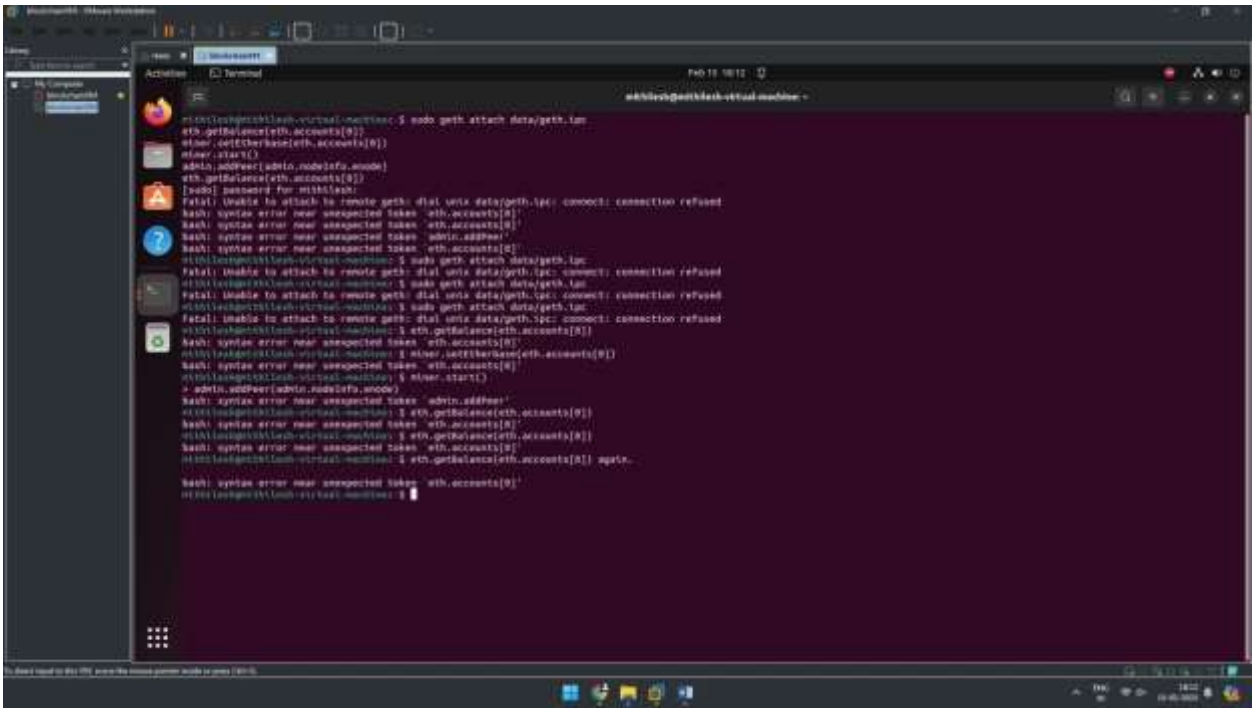






Terminal 2 :

Output:-



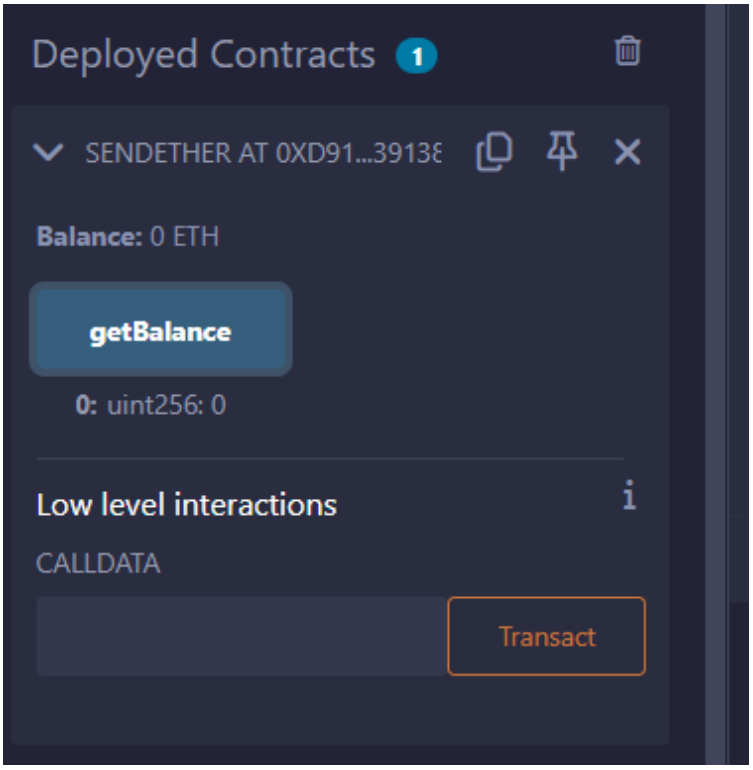
PRACTICAL 3

Aim:-Transfer ethers from one **contract** to another on an Ethereum testnet.

Code:-

```
 pragma solidity ^0.8.0;
contract sendEther{
function getBalance() external view returns(uint)
{
    return address(this).balance;
}
receive() external payable { }
}
```

Output:-



PRACTICAL 4

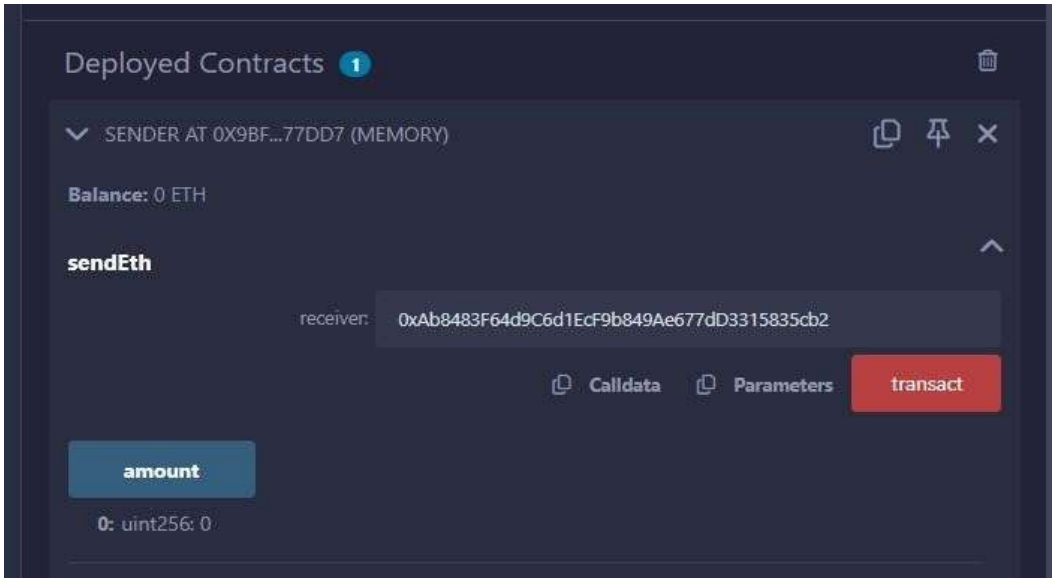
Aim:-Transfer ethers from one **account** to another on an Ethereum testnet.
Code:-

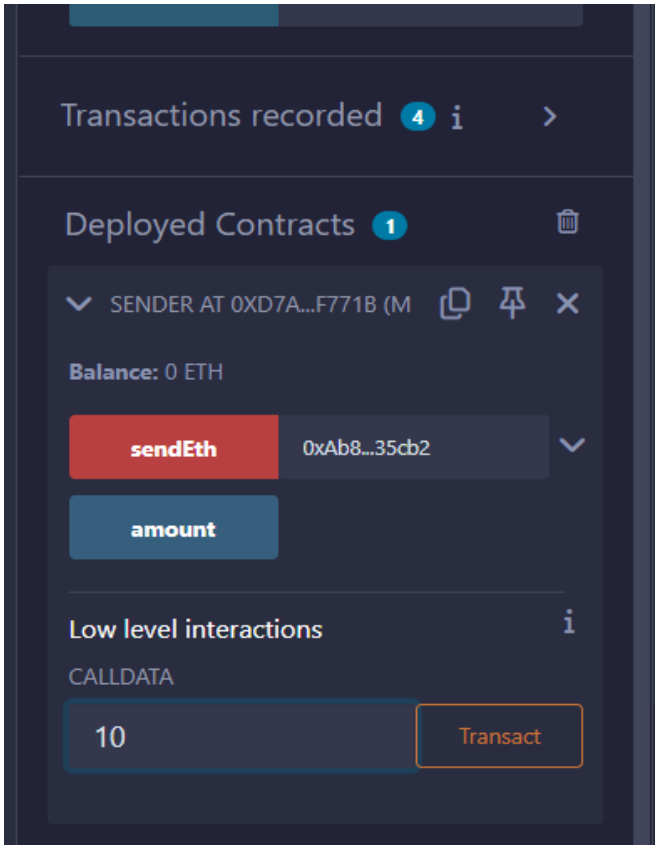
```
//https://dev.to/sparklesix/solidity-tutorial-how-to-build-and-deploy-a-smart-contract-to-send-ether-from-one-account-to-another-n54
pragma solidity ^0.8.11;
```

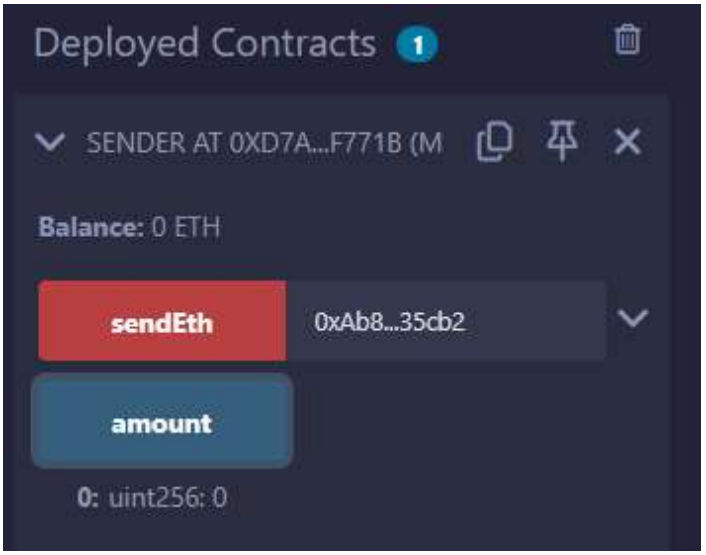
1)REMIX

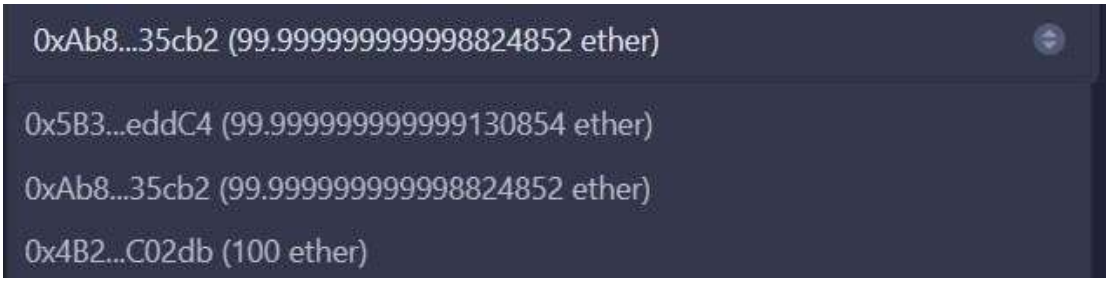
```
contract Sender {
  uint public amount;
  address payable owner;

  constructor (){
    owner = payable(msg.sender); // set the deployer of contract as the owner
  }
  function sendEth(address payable receiver) payable public{
    require(owner == msg.sender, "Only the owner can send funds");
    amount = msg.value;
    receiver.transfer(amount);
}
}
```

Output:



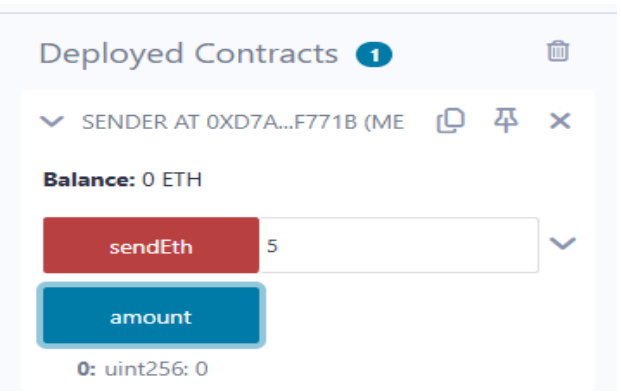


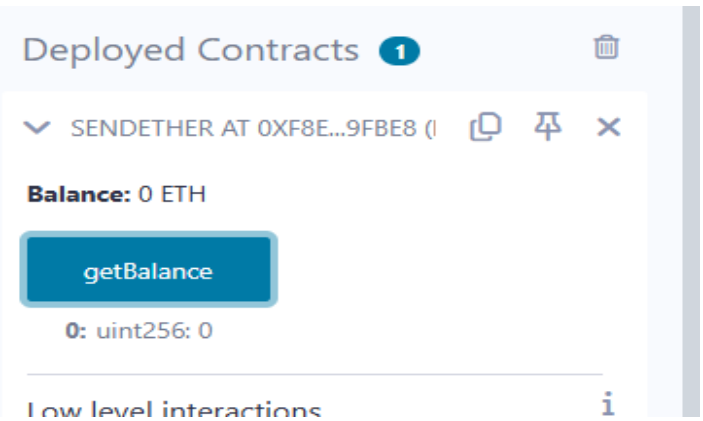




1. Transfer ethers from one **account** to another on an Ethereum testnet.

```
pragma solidity ^0.8.11;

contract Sender {
  uint public amount;
  address payable owner;

  constructor (){
    owner = payable(msg.sender); // set the deployer of contract as the owner
  }
  function sendEth(address payable receiver) payable public{
    require(owner == msg.sender, "Only the owner can send funds");
    amount = msg.value;
    receiver.transfer(amount);
}
}
```





2) Ganach

https://abhibvp003.medium.com/how-to-install-and-execute-truffle-on-an-ubuntu-16-04-7d0ff6458c9b

https://ethereum.stackexchange.com/questions/93533/call-an-existing-contract-function-from-truffle-console

sudo apt-get -y install curl git vim build-essential
sudo apt-get install curl software-properties-common

sudo apt install npm
sudo npm install -g web3
sudo apt-get install nodejs
sudo apt install python3.9
curl -sL https://deb.nodesource.com/setup_10.x | sudo bash -
sudo npm install --global node-sass@latest
sudo npm install -g truffle@latest
sudo npm install -g ganache-cli
export NODE_OPTIONS=--openssl-legacy-provider
//to update npm//
sudo npm cache clean -f
sudo npm install -g n

sudo n latest

////////////////////////////////////
Start from here!!!

mkdir upg1
cd upg1
truffle init

////////// create contract
nano contracts/HelloWorld.sol

```
pragma solidity ^0.5.0;
contract HelloWorld {
    function sayHello() public pure returns(string memory){
        return("hello world");
    }
```

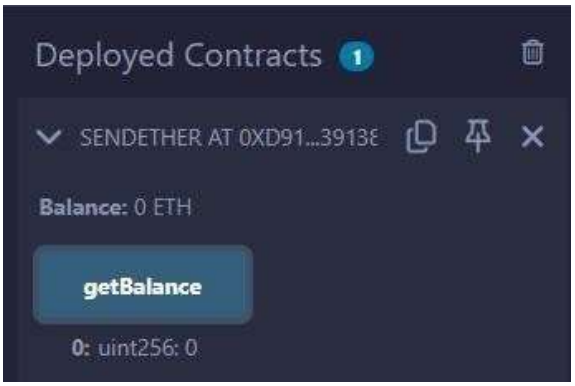
```
//////////////////create configuration
nano migrations/1_initial_migration.js
const Migrations = artifacts.require("HelloWorld");

module.exports = function (deployer) {
  deployer.deploy(Migrations,"hello");
};
///////////////network configuration
nano truffle-config.js
module.exports = {
  networks: {
    development: {
    host: "127.0.0.1",
      port: 8545,
      network_id: "*",
    }
  }
}
//////////////////start ganache-cli

ganache-cli

/////////////////
```

3) truffle migrate

```
truffle console
#replace contact address
contract = await HelloWorld.at('0x37354B83aadd35516c56f24b724228f29300be77')
a = await contract.sayHello()
a
```

2. Transfer ethers from one **contract** to another on an Ethereum testnet.

```
pragma solidity ^0.8.11;
contract sendEther{
 function getBalance() external view returns(uint)
          {
             return address(this).balance;
          }
          receive() external payable { }
          }
```



PRACTICAL 5

5)Implement and demonstrate the use of the following in Solidity:

PRACTICAL 5a

Aim:-**Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs**, Mappings, Conversions, Ether Units, Special Variables

Code:-

A)Variables:

supports three types of variables.

State Variables − Variables whose values are permanently stored in a contract storage.

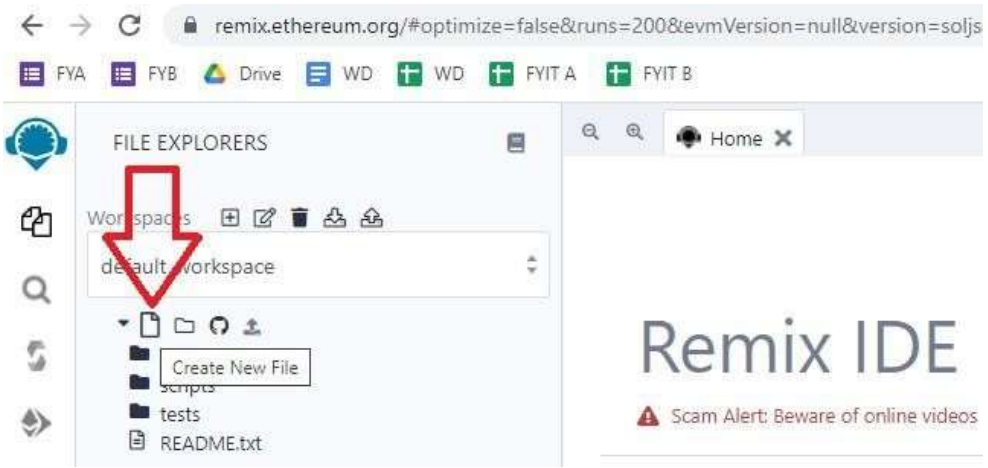
Local Variables − Variables whose values are present till function is executing.

Global Variables − Special variables exists in the global namespace used to get information about the blockchain.i.e. blockhash(uint blockNumber) returns (bytes32), block.coinbase (address payable), block.difficulty (uint)…..and many more

Step 1: Open this website

https://remix.ethereum.org/

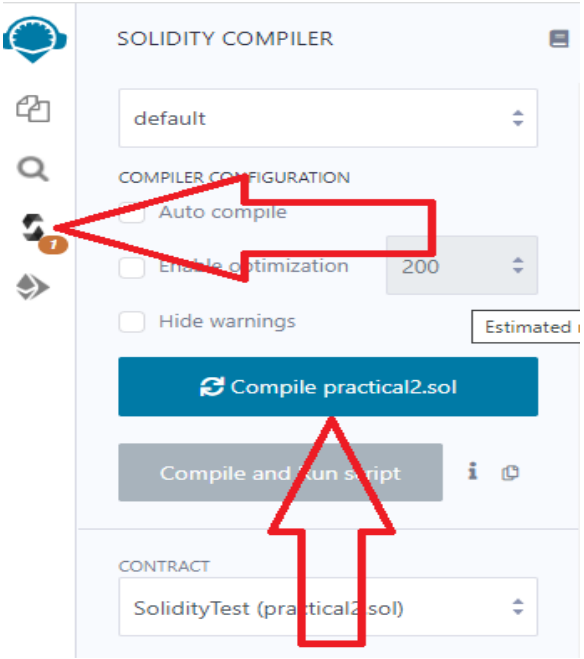
Step 2: Create new file – practical.sol



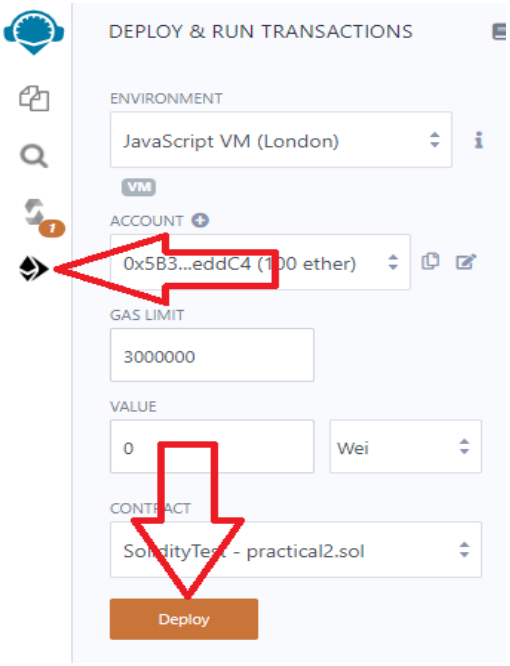
Step 3:Write the below program in new file pragma
solidity ^0.5.0;
contract SolidityTest {
uint storedData; // State variable
constructor() public { storedData
= 10;
}
function getResult() public view returns(uint){ uint
a = 1; // local variable
uint b = 2;
uint result = a + b;
return result; //access the state variable

```
}
}
```

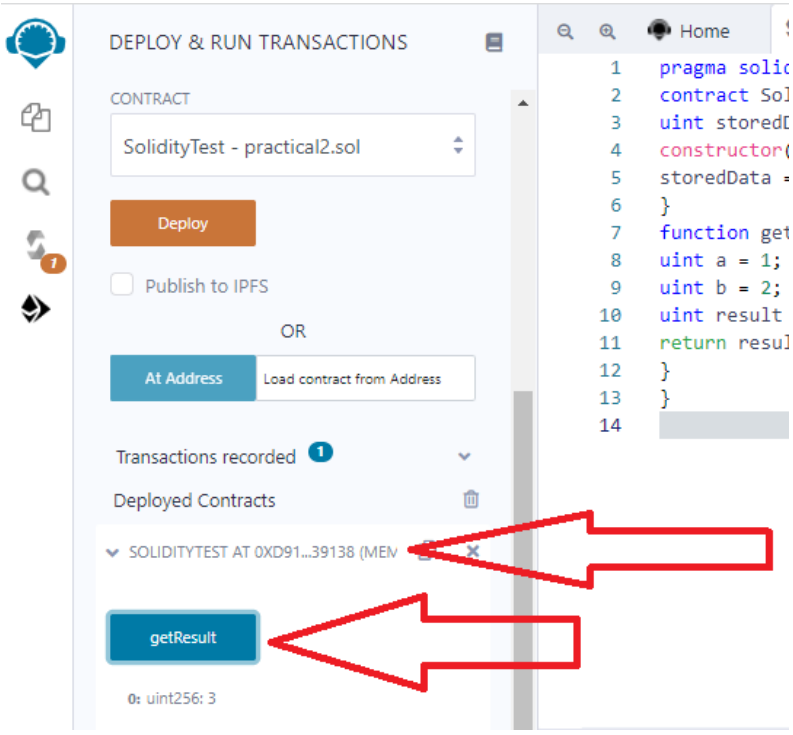
Step 4: Compile contract

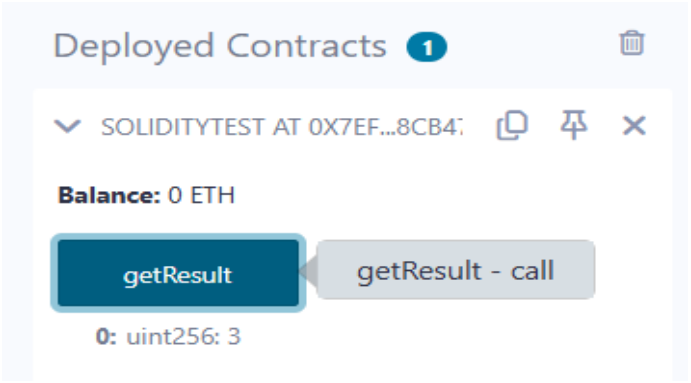


Step 5: Deploy contract



Step 6: Select the contract and click button

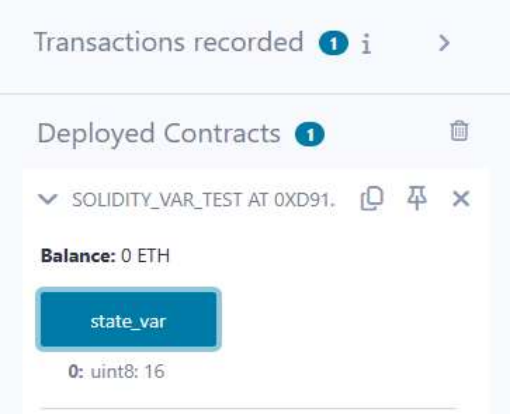




1. State Variable:

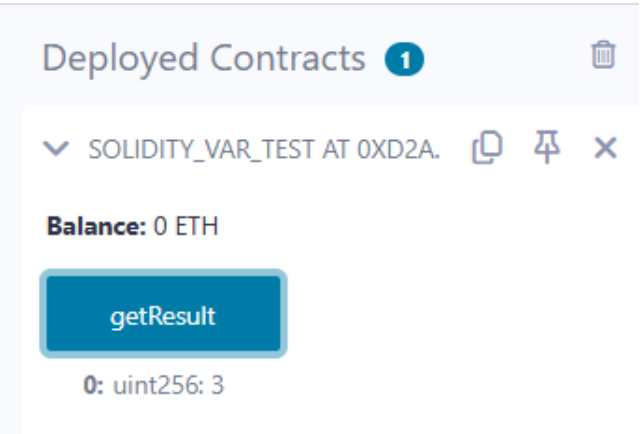
```
// Solidity program to
// demonstrate state
// variables
pragma solidity ^0.5.0;
```

```
// Creating a contract
contract Solidity_var_Test {
// Declaring a state variable
uint8 public state_var;
// Defining a constructor
constructor() public {
state_var = 16;
}
}
```



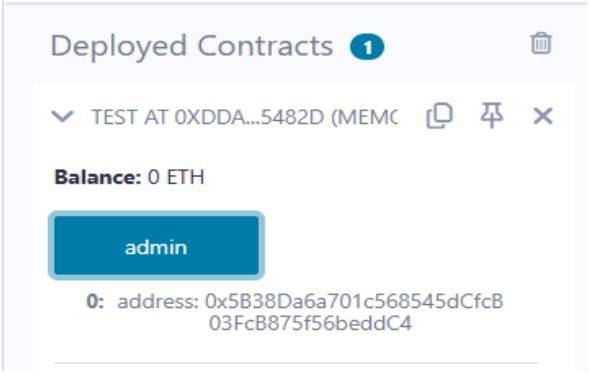
2. Local Variable:

```
// Solidity program to demonstrate
// local variables
pragma solidity ^0.5.0;
// Creating a contract
contract Solidity_var_Test {
// Defining function to show the declaration and
// scope of local variables
function getResult() public view returns(uint){
// Initializing local variables
uint local_var1 = 1;
uint local_var2 = 2;
uint result = local_var1 + local_var2;
// Access the local variable
return result;
}
}
```



3. Global variable:

```
// Solidity program to
// show Global variables
pragma solidity ^0.5.0;
// Creating a contract
contract Test {
// Defining a variable
address public admin;
// Creating a constructor to
// use Global variable
constructor() public {
admin = msg.sender;
}
}
```



Scope of local variables is limited to function in which they are defined but State variables can have three types of scopes.

Public − Public state variables can be accessed internally as well as via messages. For a public state variable, an automatic getter function is generated.

Internal − Internal state variables can be accessed only internally from the current contract or contract deriving from it without using this.

Private − Private state variables can be accessed only internally from the current contract they are defined not in the derived contract from it.

B) Operators

Solidity supports the following types of operators.

Arithmetic Operators

Comparison Operators

Logical (or Relational) Operators

Assignment Operators

Conditional (or ternary) Operators

1. Arithematic Operator

// Solidity contract to demonstrate

// Arithematic Operator

```
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
// Initializing variables
uint16 public a = 20;
uint16 public b = 10;
// Initializing a variable
// with sum
uint public sum = a + b;
// Initializing a variable
// with the difference
uint public diff = a - b;
// Initializing a variable
// with product
uint public mul = a * b;
// Initializing a variable
// with quotient
uint public div = a / b;
// Initializing a variable
// with modulus
uint public mod = a % b;
// Initializing a variable
// decrement value
uint public dec = --b;
// Initializing a variable
// with increment value
uint public inc = ++a;
}
```

2. Relational Operator

// Solidity program to demonstrate

// Relational Operator

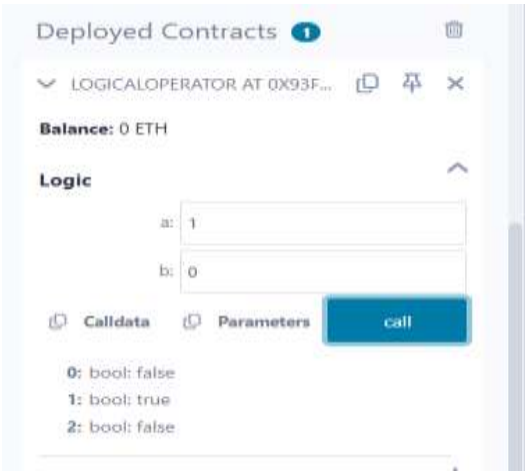
```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Declaring variables
uint16 public a = 20;
uint16 public b = 10;

// Initializing a variable
// with bool equal result
bool public eq = a == b;

// Initializing a variable
// with bool not equal result
bool public noteq = a != b;

// Initializing a variable
// with bool greater than result
bool public gtr = a > b;

// Initializing a variable
// with bool less than result
bool public les = a < b;

// Initializing a variable
// with bool greater than equal to result
bool public gtreq = a >= b;

// Initializing a variable
// bool less than equal to result
bool public leseq = a <= b;
}
```

3. <u>Logical Operators</u>

// Solidity program to demonstrate

// Logical Operators

```
pragma solidity ^0.5.0;

// Creating a contract
contract logicalOperator{

// Defining function to demonstrate
// Logical operator
function Logic(
bool a, bool b) public view returns(
bool, bool, bool){

// Logical AND operator
bool and = a&&b;

// Logical OR operator
bool or = a||b;

// Logical NOT operator
bool not = !a;
return (and, or, not);
}
}
```



4. <u>Bitwise Operators</u>

// Solidity program to demonstrate

// Bitwise Operator

```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Declaring variables
uint16 public a = 20;
uint16 public b = 10;

// Initializing a variable
// to '&' value
uint16 public and = a & b;

// Initializing a variable
// to '|' value
uint16 public or = a | b;

// Initializing a variable
// to '^' value
uint16 public xor = a ^ b;

// Initializing a variable
// to '<<' value
uint16 public leftshift = a << b;

// Initializing a variable
// to '>>' value
uint16 public rightshift = a >> b;

// Initializing a variable
// to '~' value
uint16 public not = ~a ;

}
```

1. **Assignment Operator**

// Solidity program to demonstrate

// Assignment Operator

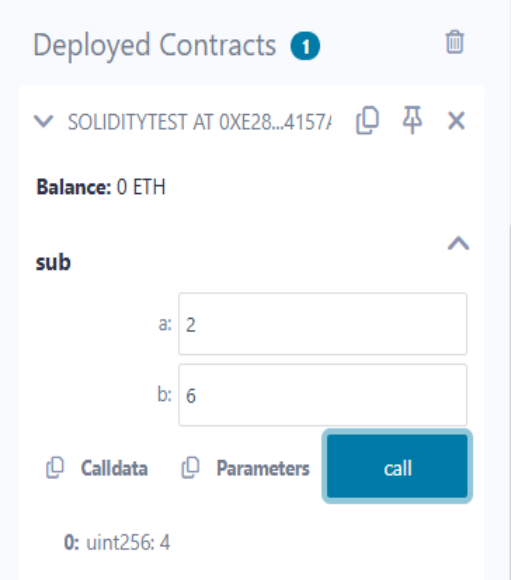
```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Declaring variables
uint16 public assignment = 20; uint
public assignment_add = 50; uint
public assign_sub = 50;
uint public assign_mul = 10;
uint public assign_div = 50;
uint public assign_mod = 32;

// Defining function to
// demonstrate Assignment Operator
function getResult() public{
assignment_add += 10;
assign_sub -= 20;
assign_mul *= 10;
assign_div /= 10;
assign_mod %= 20;
return ;
}
}
```

2. **Conditional Operators**

// Solidity program to demonstrate

// Conditional Operator

```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest{
// Defining function to demonstrate
// conditional operator
function sub(
uint a, uint b) public view returns(
uint){
uint result = (a > b? a-b : b-a);
return result;
}
}
```



C) Loops:

1. While loop: The most basic loop in Solidity is the **while** loop which would be discussed in this chapter. The purpose of a **while** loop is to execute a statement or code block repeatedly as long as an **expression** is true. Once the expression becomes **false,** the loop terminates.

2. do-while loop: The **do...while** loop is similar to the **while** loop except that the condition check happens at the end of the loop. This means that the loop will always be executed at least once, even if the condition is **false**.

3. for loop: The **for** loop is the most compact form of looping. It includes the following three important parts.

The **loop initialization** where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.

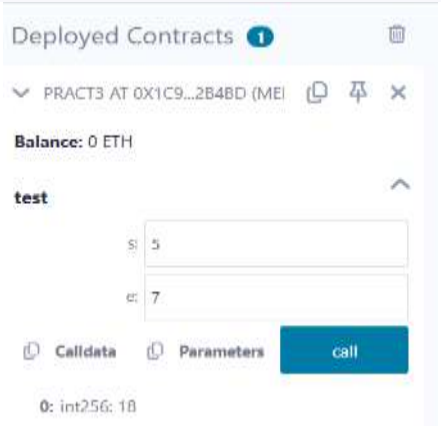
The **test statement** which will test if a given condition is true or not. If the condition is true, then the code given inside the loop will be executed, otherwise the control will come out of the loop.

The iteration statement where you can increase or decrease your counter.

1. loop control: Solidity provides full control to handle loops and switch statements. There may be a situation when you need to come out of a loop without reaching its bottom. There may also be a situation when you want to skip a part of your code block and start the next iteration of the loop.To handle all such situations, Solidity provides **break** and **continue** statements. These statements are used to immediately come out of any loop or to start the next iteration of any loop respectively.

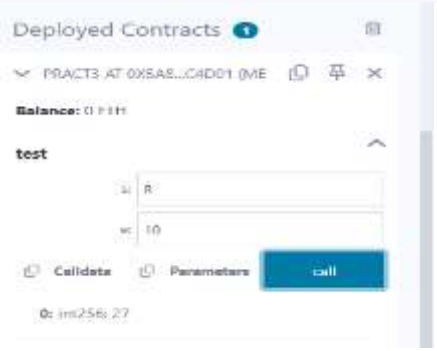
1. While Loop

```
pragma solidity ^0.5.0;
contract Pract3{
function test(int s, int e) public view returns(int)
{
int i;
int sum=0;
i=s;
while(i<=e)
{
sum+=i; //sum=sum+i;
i++;
}
return sum;
}
}
```



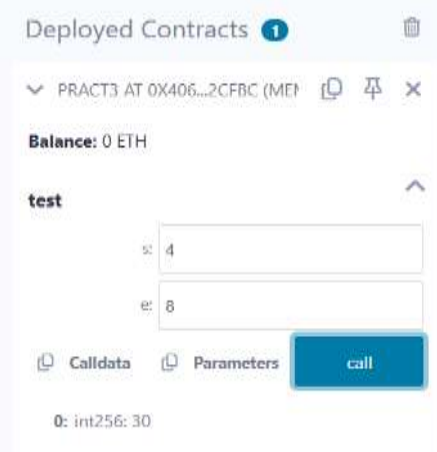
2. Do-while loop:

```
pragma solidity ^0.5.0;
contract Pract3{
function test(int s, int e) public view returns(int)
{
int i;
int sum=0;
i=s;
do
{
sum+=i; //sum=sum+i;
i++;
}while(i<=e);
return sum;
}
}
```



3. For Loop:

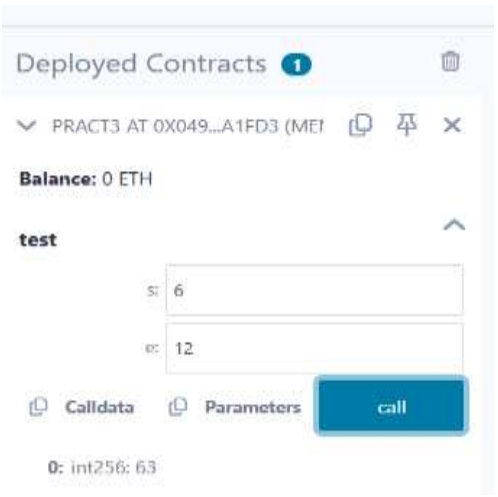
```
contract Pract3{
function test(int s, int e) public view returns(int)
{
int i;
int sum=0;
for(i=s;i<=e;i++)
{
sum+=i;  //sum=sum+i;
}
return sum;
}
}
```



4. loop Control: (Break statement)

```
pragma solidity ^0.5.0;

contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {

if (_i == 0) {
return "0";
}
uint j = _i;
uint len;

while (true) {
len++;
j /= 10;
if(j==0){
break;   //using break statement
}
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```

(continue statement)



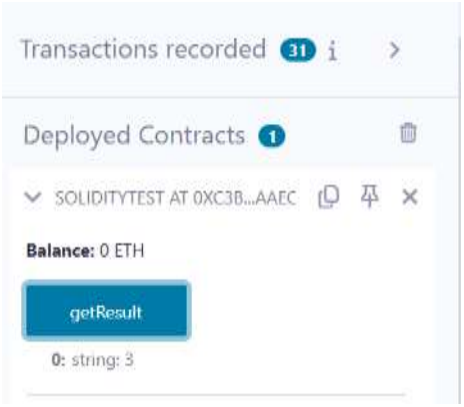
```
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint n = 1;
uint sum = 0;

while( n < 10){
n++;
if(n == 5){
continue; // skip n in sum when it is 5.
}
sum = sum + n;
}
return integerToString(sum);
}
function integerToString(uint _i) internal pure
returns (string memory) {

if (_i == 0) {
return "0";
}
uint j = _i;
uint len;

while (true) {
len++;
j /= 10;
if(j==0){
break;   //using break statement
}
}
bytes memory bstr = new bytes(len);

uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```

D) Decision Making:

While writing a program, there may be a situation when you need to adopt one out of a given set of paths. In such cases, you need to use conditional statements that allow your program to make correct decisions and perform right actions.Solidity supports conditional statements which are used to perform different actions based on different conditions. Here we will explain the **if..else** statement.

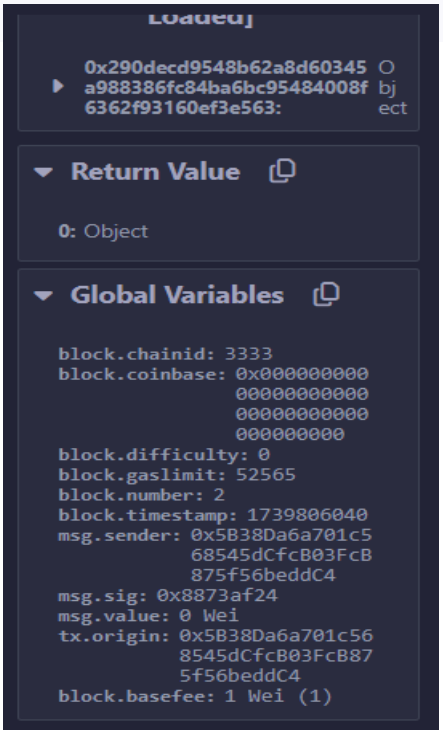
1. if statement: The **if** statement is the fundamental control statement that allows Solidity to make decisions and execute statements conditionally.

```
pragma solidity ^0.5.0;

contract SolidityTest {
uint storedData;
constructor() public {
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {
if (_i == 0) {   // if statement
return "0";
}
uint j = _i;
uint len;

while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);//access local variable
}}
```



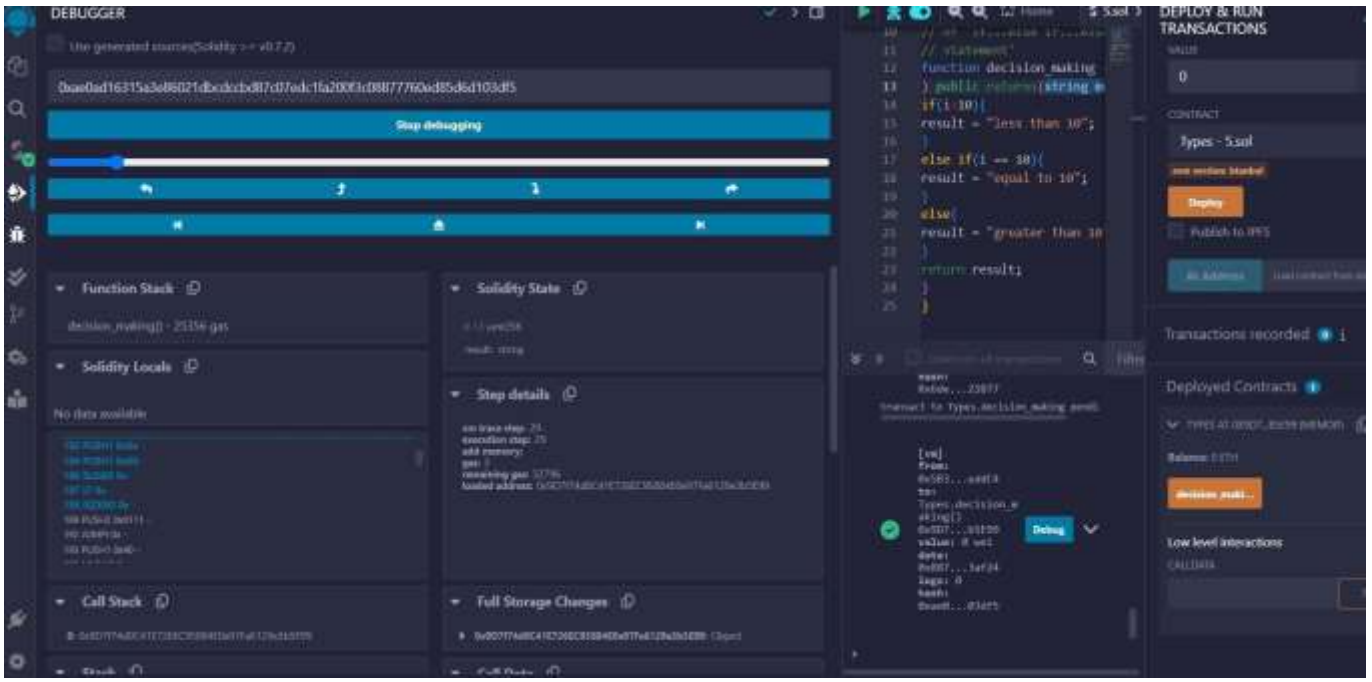
2.<u>if-else statement:</u> The **'if...else'** statement is the next form of control statement that allows Solidity to execute statements in a more controlled way.

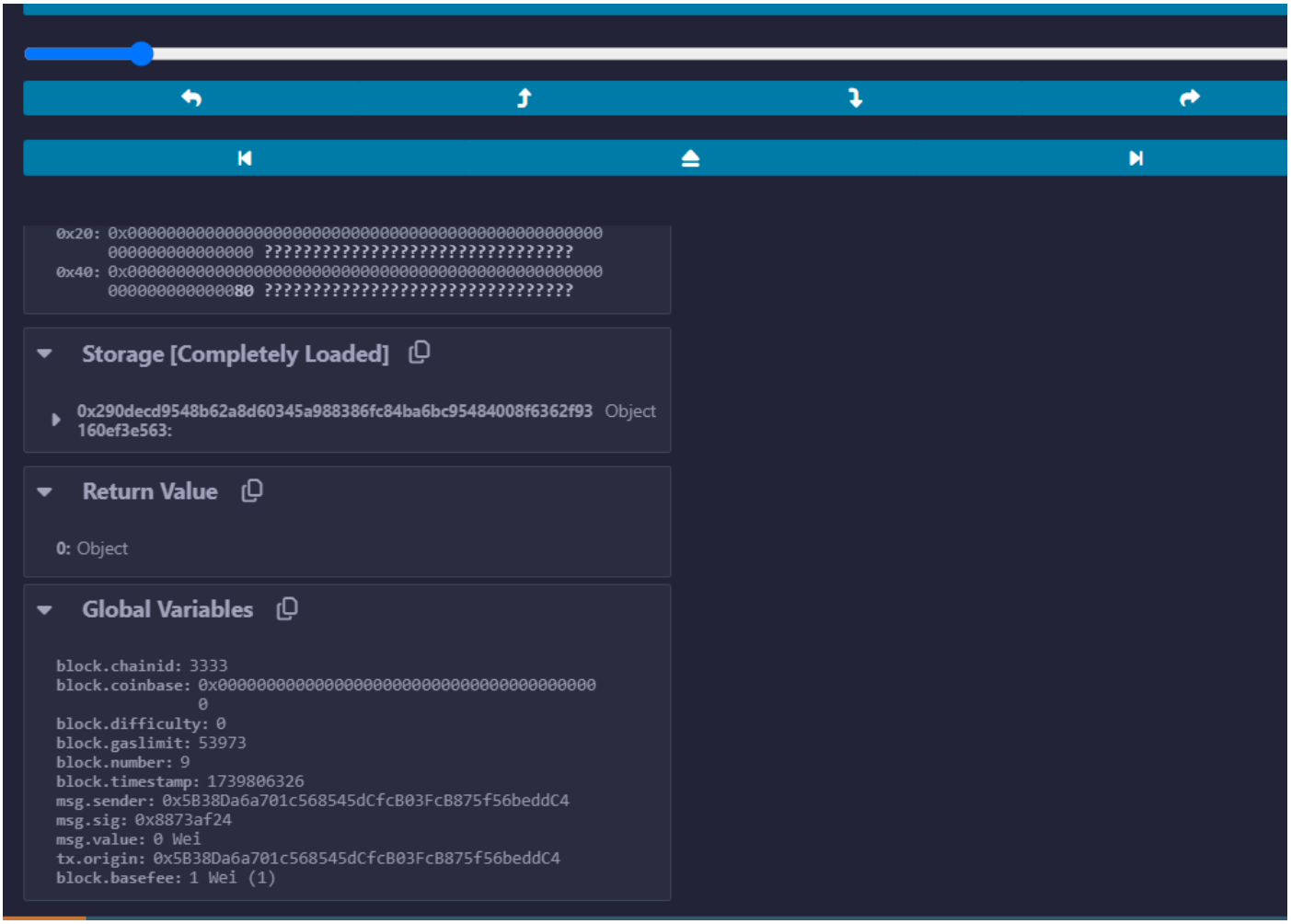
```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
// Declaring state variables
uint i = 10;
bool even;

// Defining function to
// demonstrate the use of
// 'if...else statement'
function decision_making(
) public payable returns(bool){
if (i%2 == 0){
even = true;
}
else{
even = false;
}
return even;
}
}
```



3.<u>if-else..if statement</u>: The **if...else if...** statement is an advanced form of **if...else** that allows Solidity to make a correct decision out of several conditions.

```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
// Declaring state variables
uint i = 12;
string result;
// Defining function to
// demonstrate the use
// of 'if...else if...else
// statement'
function decision_making (
) public returns(string memory){
if(i<10){
result = "less than 10";
}
else if(i == 10){
result = "equal to 10";
}
else{
result = "greater than 10";
}
return result;
}
}
```

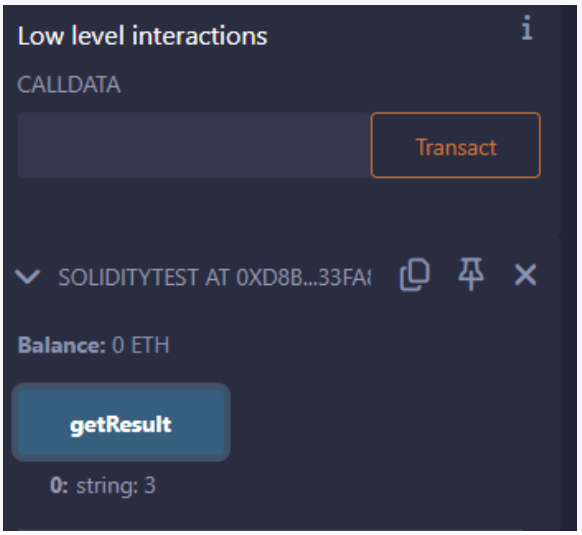



0x20: 0x0000000000000000000000000000000000000000000000000000000000000000
0x40: 0x0000000000000000000000000000000000000000000000000000000000000080
Storage [Completely Loaded]
0x290decd9548b62a8d60345a988386fc84ba6bc95484008f6362f93160ef3e563: Object
Return Value
0: Object
Global Variables
block.chainid: 3333
block.coinbase: 0x0000000000000000000000000000000000000000
block.difficulty: 0
block.gaslimit: 53973
block.number: 9
block.timestamp: 1739806326
msg.sender: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
msg.sig: 0x8873af24
msg.value: 0 Wei
tx.origin: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
block.basefee: 1 Wei (1)

B) String :PUBLIC FUNCTION

Solidity supports String literal using both double quote ('') and single quote ('). It provides string as a data type to declare a variable of type String.(Int to str)

```
pragma solidity ^0.5.0;

contract SolidityTest {
constructor() public{
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {

if (_i == 0) {
return "0";
}
uint j = _i;
uint len;

while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```



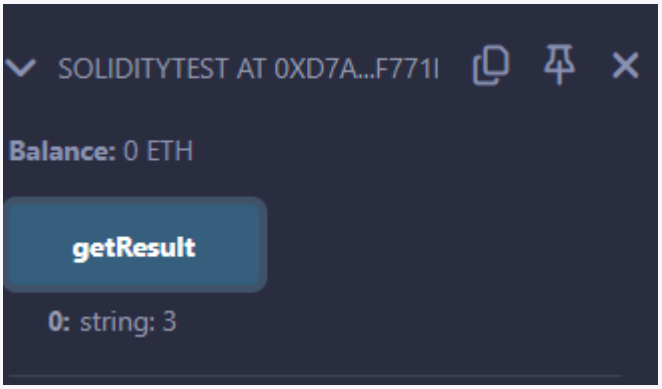
B) Array:

Array is a data structure, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

// Solidity program to demonstrate

// accessing elements of an array

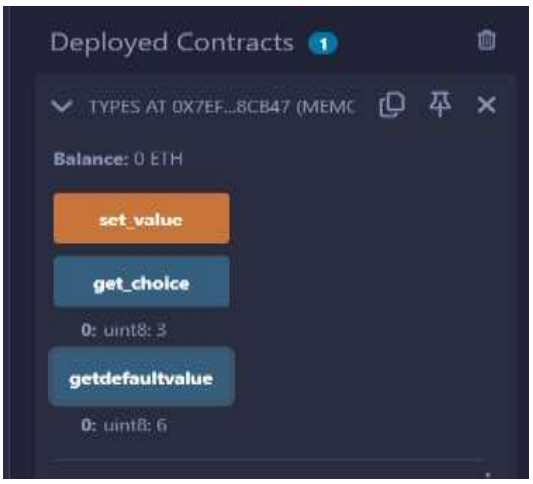
```
pragma solidity ^0.5.0;
function
// Creating a contract
contract Types {

// Declaring an array
uint[6] data;
uint x;

// Defining function to
// assign values to array
function array_example() public returns (uint[6] memory)
{

data  = [uint(10), 20, 30, 40, 50, 60];
}
function result() public view returns(uint[6] memory){
return data;
}
// Defining function to access
// values from the array
// from a specific index
function array_element() public view returns (uint){
uint x = data[2];
return x;
}
}
```



C) Enums:

Enums restrict a variable to have one of only a few predefined values. The values in this enumerated list are called enums. With the use of enums it is possible to reduce the number of bugs in your code.

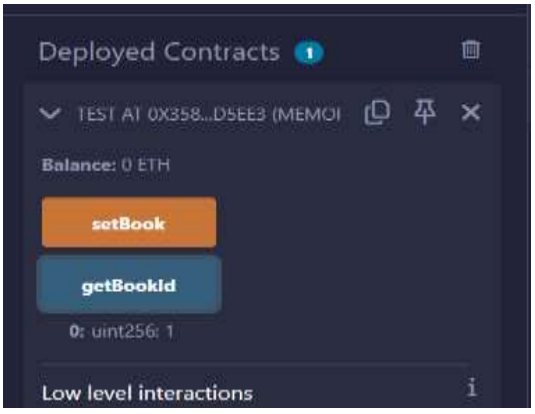
```
// Solidity program to demonstrate

// how to use 'enumerator'
pragma solidity ^0.5.0;

// Creating a contract
contract Types {

// Creating an enumerator
enum week_days
{
Monday,
Tuesday,
Wednesday,
Thursday,
Friday,
Saturday,
Sunday
}
// Declaring variables of
// type enumerator
week_days week;

week_days choice;

// Setting a default value
week_days constant default_value
= week_days.Sunday;

// Defining a function to
// set value of choice
function set_value() public {
choice = week_days.Thursday;
}
// Defining a function to
// return value of choice
function get_choice(
) public view returns (week_days) {
return choice;
}
// Defining function to
// return default value
function getdefaultvalue(
) public pure returns(week_days) {
return default_value;
}
}
```



D) Structure:

Struct types are used to represent a record.

```
pragma solidity ^0.5.0;

contract test {
struct Book {
string title;
string author;
uint book_id;
}
Book book;

function setBook() public {
book = Book('Learn Java', 'TP', 1);
}
function getBookId() public view returns (uint) {
return book.book_id;
}
}
```



E) Mappings:

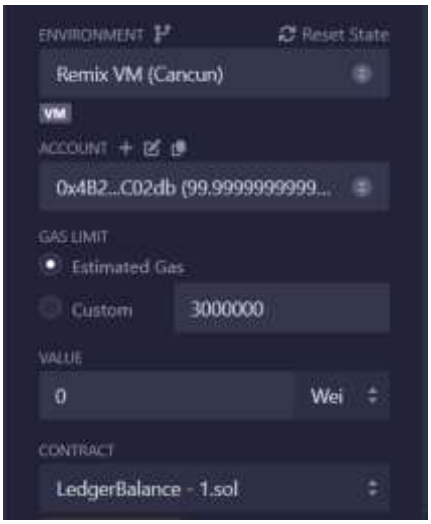
Mapping is a reference type as arrays and structs. Following is the syntax to declare a mapping type.

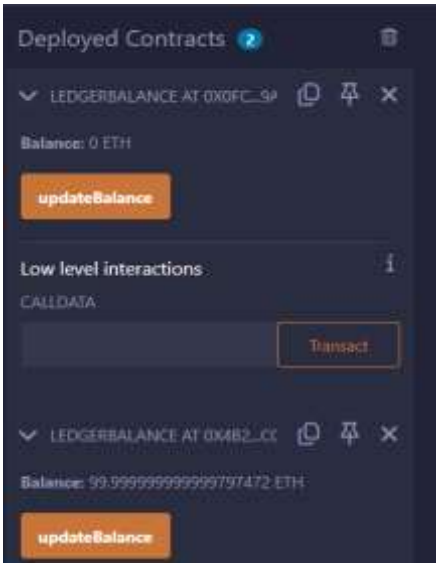
mapping(_KeyType => _ValueType) where ,

_KeyType − can be any built-in types plus bytes and string. No reference type or complex objects are allowed.

ValueType − can be any type.

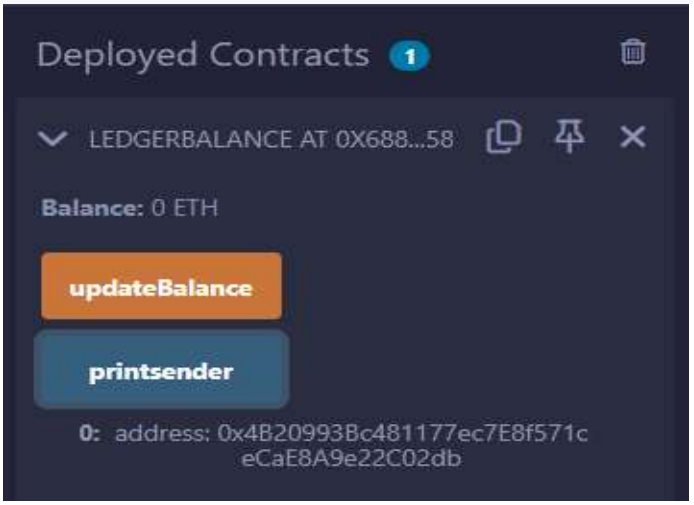
```
pragma solidity ^0.5.0;
contract LedgerBalance {
mapping(address => uint) balance;

function updateBalance() public  returns(uint) {
balance[msg.sender]=30;
return balance[msg.sender];
}
}
```





Mapping program for String.

```
pragma solidity ^0.5.0;

contract LedgerBalance {
mapping(address => string) name;

function updateBalance() public returns(string memory){
name[msg.sender] = "Mrunali";
return name[msg.sender];
}
function printsender() public view returns(address) {
return msg.sender;
}
}
```



PRACTICAL 5B

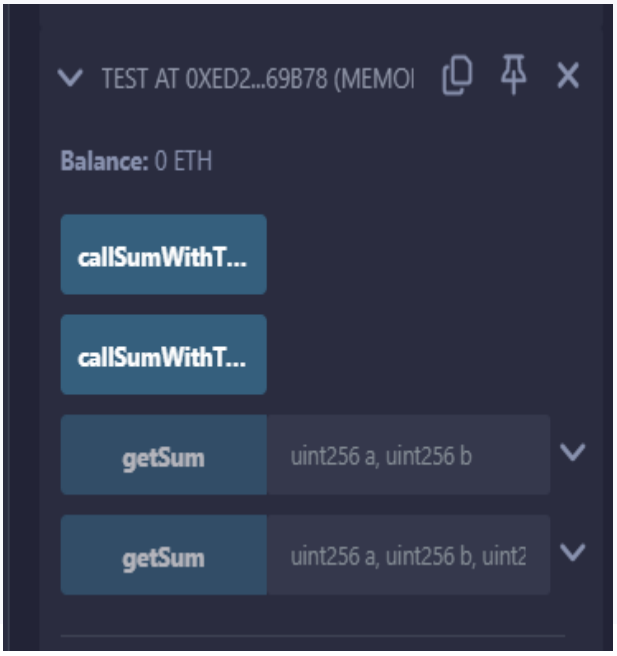
Aim: WRITE A SOLIDITY PROGRAM FOR FUNCTION OVERLOADING, MATHEMATICAL FUNCTION & CRYPTOGRAPHIC FUNCTIONS VIEW FUNCTION, PURE FUNCTION & FALLBACK FUNCTION.

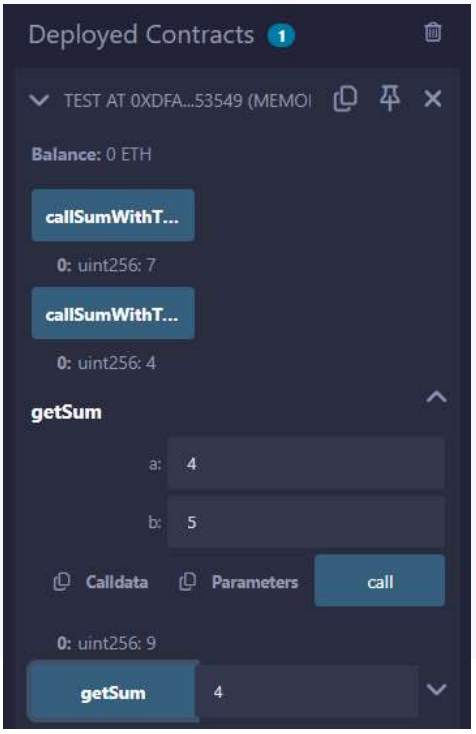
Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.

A) Function Overloading:

The definition of the function must differ from each other by the types and/or the number of arguments in the argument list. You cannot overload function declarations that differ only by return type.

```
pragma solidity ^0.5.0;

contract Test {
function getSum(uint a, uint b) public pure returns(uint){
return a + b;
}
function getSum(uint a, uint b, uint c ) public pure returns(uint){
return a + b + c;
}
function callSumWithTwoArguments() public pure returns(uint){
return getSum(2,2);
}
function callSumWithThreeArguments() public pure returns(uint){
return getSum(1,2,4);
}
```

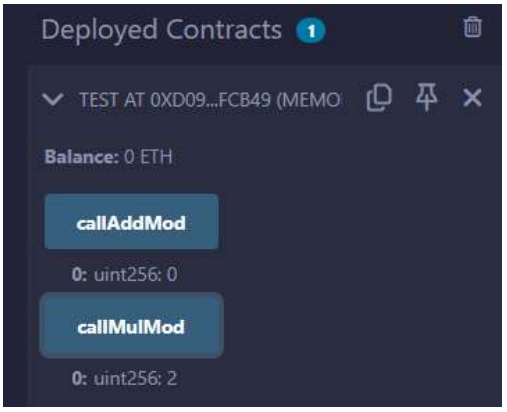




B) Mathematical Function:

Solidity provides inbuilt mathematical functions as well.

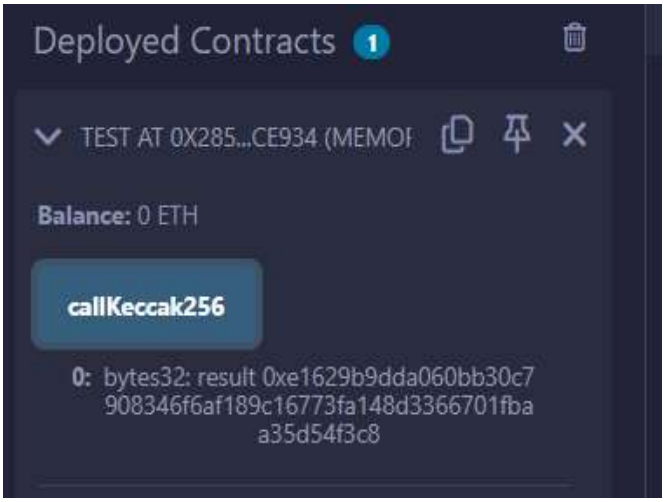
```
pragma solidity ^0.5.0;

contract Test {
function callAddMod() public pure returns(uint){
return addmod(4, 5, 3);
}
function callMulMod() public pure returns(uint){
return mulmod(4, 5, 3);
}
}
```



Cryptographic Function:

Solidity provides inbuilt cryptographic functions as well.

```
pragma solidity ^0.5.0;
contract Test {
function callKeccak256() public pure returns(bytes32
result){ return keccak 256("ABC");
}
}
```



C) Function:

A function is a group of reusable code which can be called anywhere in your program. This eliminates the need of writing the same code again and again. It helps programmers in writing modular codes. Functions allow a programmer to divide a big program into a number of small and manageable functions.

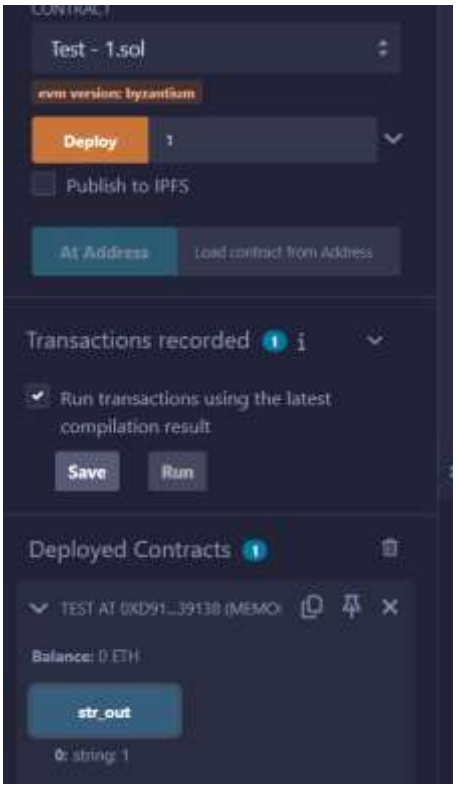
```
pragma solidity ^0.5.0;

contract SolidityTest {
constructor() public{
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {

if (_i == 0) {
return "0";
}
uint j = _i;
uint len;

while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);//access local variable

}
}
```

D) View Function:

View functions ensure that they will not modify the state. A function can be declared as **view**. Getter method are by default view functions.

```
// Solidity program to demonstrate

// how to create a contract

pragma solidity ^0.4.23;

// Creating a contract
contract Test {
// Declaring variable
string  str;

// Defining a constructor
constructor(string str_in){
str = str_in;
}
// Defining a function to
// return value of variable 'str'
function str_out() public view returns(string memory){
return str;
}
}
```

Note: after deploy it asked u to enter string then enter string over there and then see the output after clicking on str_out button



E) Pure Function:

Pure functions ensure that they not read or modify the state. A function can be declared as **pure**. Pure functions can use the revert() and require() functions to revert potential state changes if an error occurs.

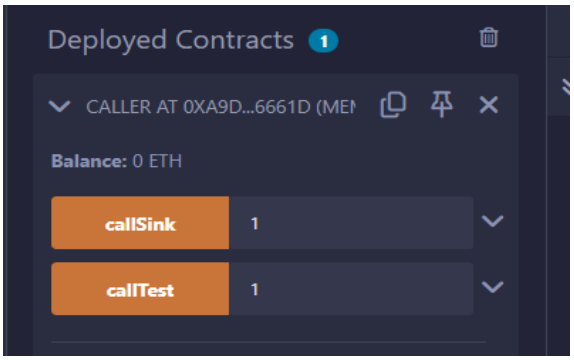
```
pragma solidity ^0.5.0;
contract Test {
int public x=10; //global
int y=90;//state
function f1() public returns(int){
    //read and update is allowed
    x=100;
return x;
}
function f2() public view returns(int){
  //  x=100; //erro beacuse x is global/state
  //we can access but we cannot update state or global variable int view function
return x;
}
function f3() public pure returns(int){
    //we cannot access or update state or global variable in pure function
    int z=80;
return z;
}
}
```

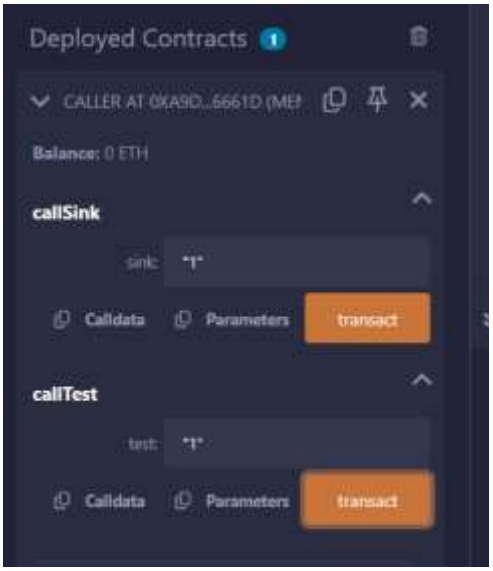
F) Fallback Function:

Fallback function is a special function available to a contract.

```
pragma solidity ^0.5.0;
contract Test {
uint public x ;
function() external { x = 1; }
}
contract Sink {
function() external payable { }
```

```
contract Caller {
function callTest(Test test) public returns (bool) {
(bool success,) = address(test).call(abi.encodeWithSignature("nonExistingFunction()"));
require(success);
// test.x is now 1
address payable testPayable = address(uint160(address(test)));
// Sending ether to Test contract,
// the transfer will fail, i.e. this returns false here.
return (testPayable.send(2 ether));
}
function callSink(Sink sink) public returns (bool) {
address payable sinkPayable = address(sink);
return (sinkPayable.send(2 ether));
}
}
```





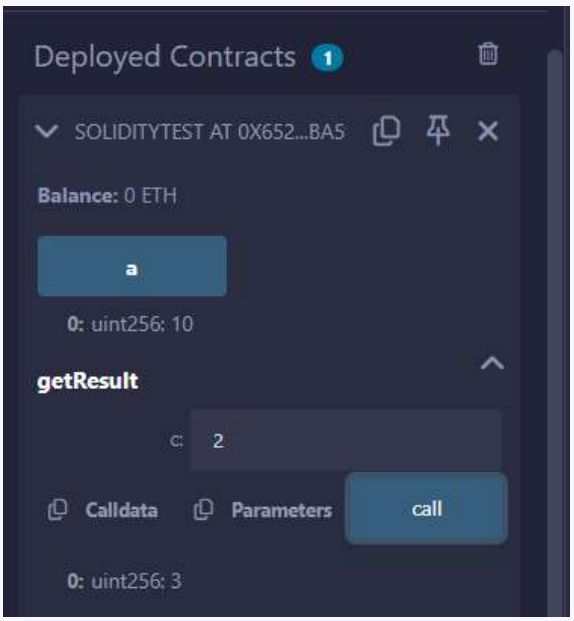
PRACTICAL 6

Aim:-Implement and demonstrate the use of the following in Solidity.

PRACTICAL 6a

6a. Withdrawal Pattern, Restricted Access.

```
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData; // State variable
uint public a=10;
constructor() public {
storedData = 10;
}
function getResult(uint c) public view returns(uint){
uint a = 1; // local variable
uint b = 2;
uint result = a + b;
return result; //access the state variable
}
}
```



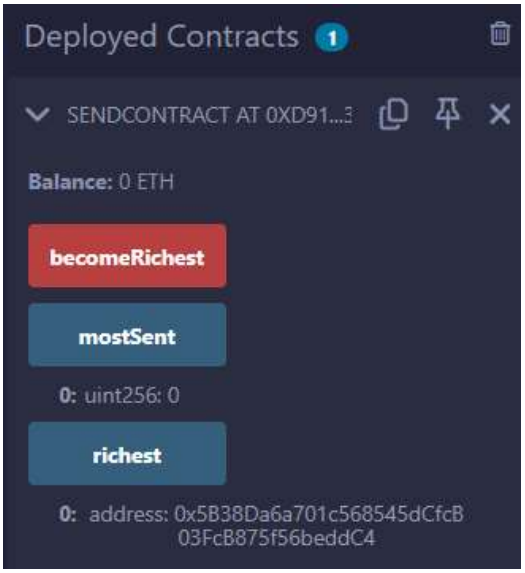
Withdraw Pattern:-

The recommended method of sending funds after an effect is using the withdrawal pattern. Although the most intuitive method of sending Ether, as a result of an effect, is a direct transfer call, this is not recommended as it introduces a potential security risk. You may read more about this on the Security Considerations page.

The following is an example of the withdrawal pattern in practice in a contract where the goal is to send the most of some compensation, e.g. Ether, to the contract in order to become the “richest”, inspired by King of the Ether.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.4;
```

```
contract SendContract {
    address payable public richest;
    uint public mostSent;

    /// The amount of Ether sent was not higher than
    /// the currently highest amount.
    error NotEnoughEther();

    constructor() payable {
        richest = payable(msg.sender);
        mostSent = msg.value;
    }

    function becomeRichest() public payable {
        if (msg.value <= mostSent) revert NotEnoughEther();
        // This line can cause problems (explained below).
        richest.transfer(msg.value);
        richest = payable(msg.sender);
        mostSent = msg.value;
    }
}
```



Restricted Access:-

Restricting access is a common pattern for contracts. Note that you can never restrict any human or computer from reading the content of your transactions or your contract's state. You can make it a bit harder by using encryption, but if your contract is supposed to read the data, so will everyone else.

You can restrict read access to your contract's state by other contracts. That is actually the default unless you declare your state variables public.

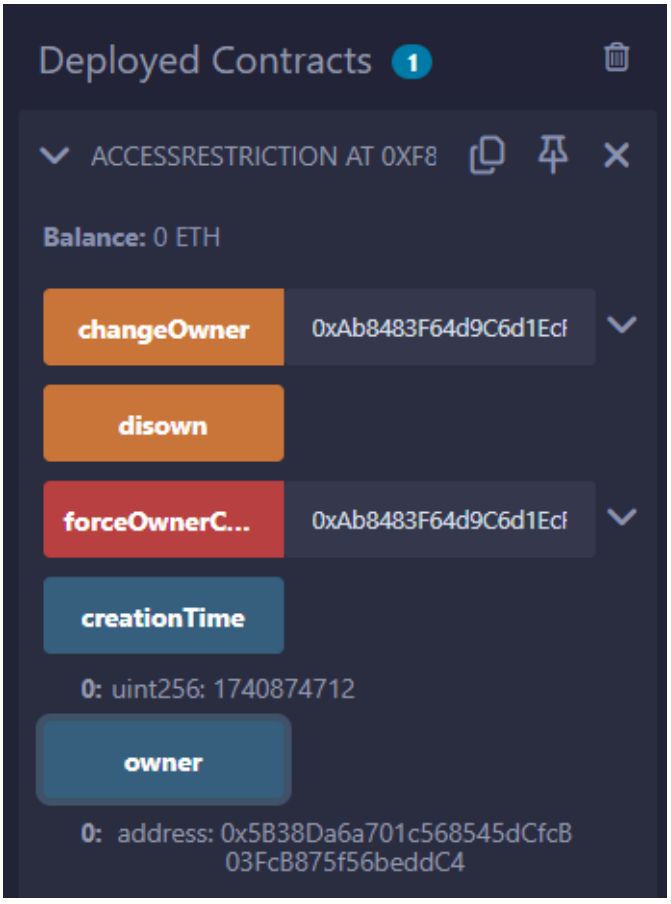
Furthermore, you can restrict who can make modifications to your contract's state or call your contract's functions and this is what this section is about.

The use of function modifiers makes these restrictions highly readable.

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.4;

contract AccessRestriction {
    // These will be assigned at the construction
    // phase, where `msg.sender` is the account
    // creating this contract.
    address public owner = msg.sender;
    uint public creationTime = block.timestamp;

    // Now follows a list of errors that
    // this contract can generate together
    // with a textual explanation in special
    // comments.

    /// Sender not authorized for this
    /// operation.
    error Unauthorized();

    /// Function called too early.
    error TooEarly();

    /// Not enough Ether sent with function call.
    error NotEnoughEther();

    // Modifiers can be used to change
    // the body of a function.
    // If this modifier is used, it will
    // prepend a check that only passes
    // if the function is called from
    // a certain address.
    modifier onlyBy(address account)
    {
        if (msg.sender != account)
            revert Unauthorized();
        // Do not forget the "_;"! It will
```

```
    // be replaced by the actual function
    // body when the modifier is used.
    _;
}

/// Make `newOwner` the new owner of this
/// contract.
function changeOwner(address newOwner)
    public
    onlyBy(owner)
{
    owner = newOwner;
}

modifier onlyAfter(uint time) {
    if (block.timestamp < time)
        revert TooEarly();
    _;
}

/// Erase ownership information.
/// May only be called 6 weeks after
/// the contract has been created.
function disown()
    public
    onlyBy(owner)
    onlyAfter(creationTime + 6 weeks)
{
    delete owner;
}

// This modifier requires a certain
// fee being associated with a function call.
// If the caller sent too much, he or she is
// refunded, but only after the function body.
// This was dangerous before Solidity version 0.4.0,
// where it was possible to skip the part after `_;`.
modifier costs(uint amount) {
    if (msg.value < amount)
```

```
            revert NotEnoughEther();

        _;
        if (msg.value > amount)
            payable(msg.sender).transfer(msg.value - amount);
    }

    function forceOwnerChange(address newOwner)
        public
        payable
        costs(200 ether)
    {
        owner = newOwner;
        // just some example condition
        if (uint160(owner) & 0 == 1)
            // This did not refund for Solidity
            // before version 0.4.0.
            return;
        // refund overpaid fees
    }
}
```



PRACTICAL 6B

Aim:- **WRITE A SOLIDITY PROGRAM FOR CONTRACT, INHERITANCE, CONSTRUCTORS, ABSTRACT CONTRACTS, INTERFACES, LIBRARIES, ASSEMBLY, EVENTS, ERROR HANDLING.**

A) Contract:

Contract in Solidity is similar to a Class in C++. A Contract have following properties.

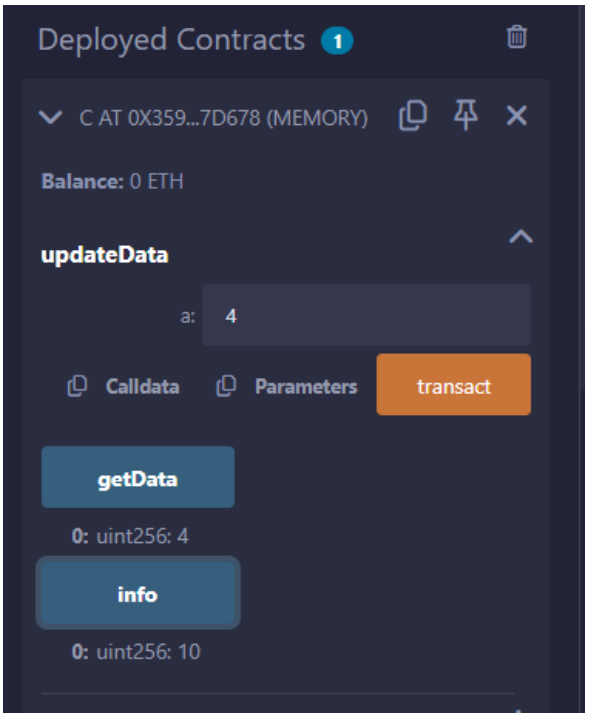
Constructor − A special function declared with constructor keyword which will be executed once per contract and is invoked when a contract is created.

State Variables − Variables per Contract to store the state of the contract.

Functions − Functions per Contract which can modify the state variables to alter the state of a contract.

// Calling function from external contract

```
pragma solidity ^0.5.0;
contract C {
//private state variable
uint private data;

//public state variable
uint public info;

//constructor
constructor() public {
info = 10;
}
//private function
function increment(uint a) private pure returns(uint) { return a + 1; }
//public function
function updateData(uint a) public { data = a; }
function getData() public view returns(uint) { return data; }
function compute(uint a, uint b) internal pure returns (uint) { return a + b; }
}
//Derived Contract
contract E is C {
uint private result;
C private c;

constructor() public {
c = new C();
}
function getComputedResult() public {
result = compute(3, 5);
}
function getResult() public view returns(uint) { return result; }
function getData() public view returns(uint) { return c.info(); }
}
```



B) Inheritance:

Inheritance is a way to extend functionality of a contract. Solidity supports both single as well as multiple inheritance.

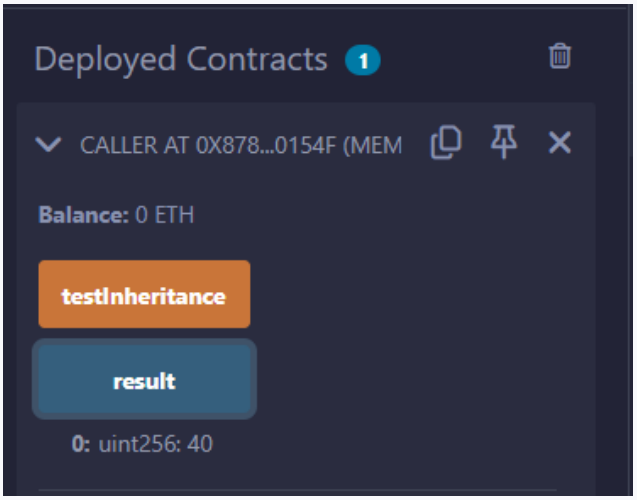
// Solidity program to

// demonstrate

// Single Inheritance

```
pragma solidity >=0.4.22 <0.6.0;

// Defining contract
contract parent{
// Declaring internal
// state variable
uint internal sum;

// Defining external function
// to set value of internal
// state variable sum
function setValue() external {
uint a = 20;
uint b = 20;
sum = a + b;
}
}
// Defining child contract
contract child is parent{

// Defining external function
// to return value of
// internal state variable sum
function getValue() external view returns(uint) {
```

```
return sum;
}
}
// Defining calling contract
contract caller {

// Creating child contract object
child cc = new child();

// Defining function to call
// setValue and getValue functions
function testInheritance() public {
cc.setValue();
}
function result() public view returns(uint ){
return cc.getValue();
}
}
```



C) Constructors:

Constructor is a special function declared using constructor keyword. It is an optional function and is used to initialize state variables of a contract. Following are the key characteristics of a constructor.

A contract can have only one constructor.

A constructor code is executed once when a contract is created and it is used to initialize contract state.

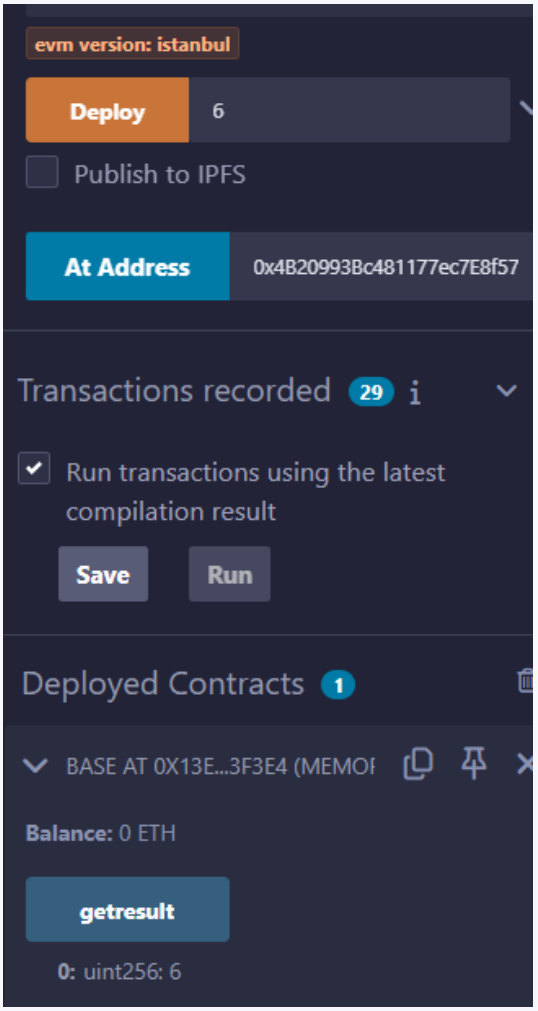
A constructor can be either public or internal.

An internal constructor marks the contract as abstract.

In case, no constructor is defined, a default constructor is present in the contract.

```
pragma solidity ^0.5.0;
contract Base {
uint data;
constructor(uint _data) public {
data = _data;
}
function getresult()public view returns(uint){
return data;
}
```

```
}
contract Derived is Base (5) {
constructor() public {}
}
```



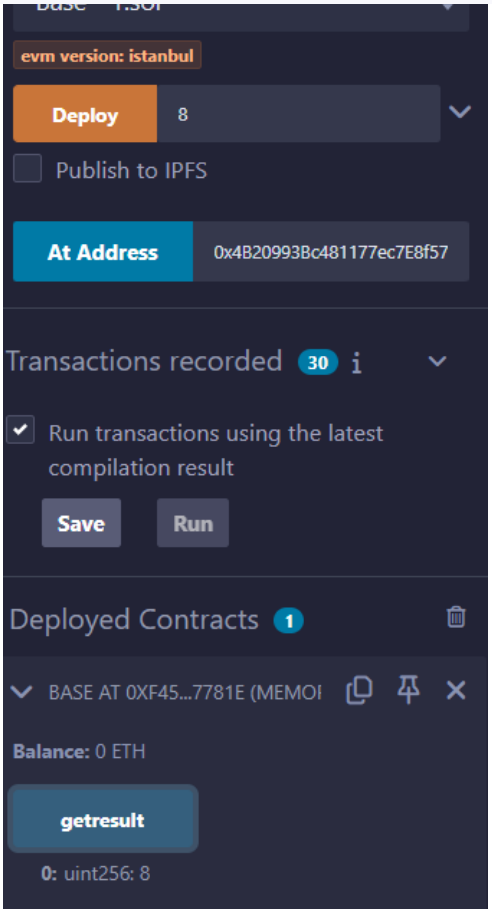
// Indirect Initialization of Base Constructor

```
pragma solidity ^0.5.0;

contract Base {
uint data;
constructor(uint _data) public {
data = _data;
}
function getresult()public view returns(uint){
return data;
}
}
contract Derived is Base {
constructor(uint _info) Base(_info * _info) public {}
}
```

D) Abstract Contracts:

Abstract Contract is one which contains at least one function without any implementation. Such a contract is used as a base contract. Generally an abstract contract contains both implemented as well as abstract functions. Derived contract will implement the abstract function and use the existing functions as and when required.

```
pragma solidity ^0.5.0;

contract Calculator {
function getResult() public view returns(uint);
}
contract Test is Calculator {
function getResult() public view returns(uint) {
uint a = 4;
uint b = 2;
uint result = a + b;
return result;
}
}
```



E) Interfaces:

Interfaces are similar to abstract contracts and are created using interface keyword. Following are the key characteristics of an interface.

Interface can not have any function with implementation.

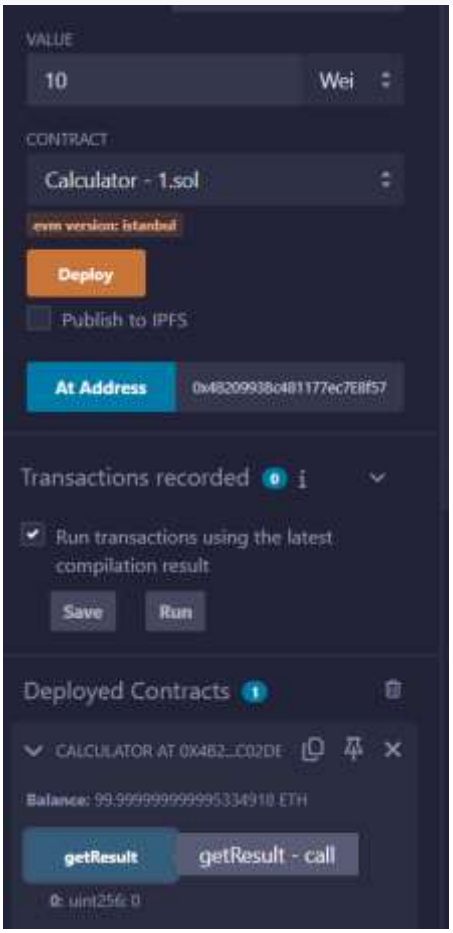
Functions of an interface can be only of type external.

Interface can not have constructor.

Interface can not have state variables.

```
pragma solidity ^0.5.0;

interface Calculator {
```

```
function getResult() external view returns(uint);
}
contract Test is Calculator {
constructor() public {}
function getResult() external view returns(uint){
uint a = 5;
uint b = 2;
uint result = a + b;
return result;
}
}
```



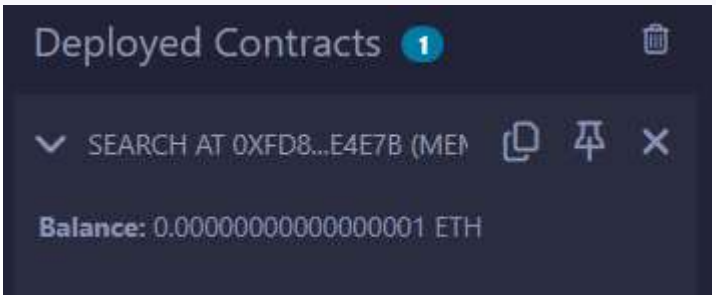
PRACTICAL 6c

Aim:-Libraries,Assembly, Events, Error handling.

Libraries:

Libraries are similar to Contracts but are mainly intended for reuse. A Library contains functions which other contracts can call. Solidity have certain restrictions on use of a Library.

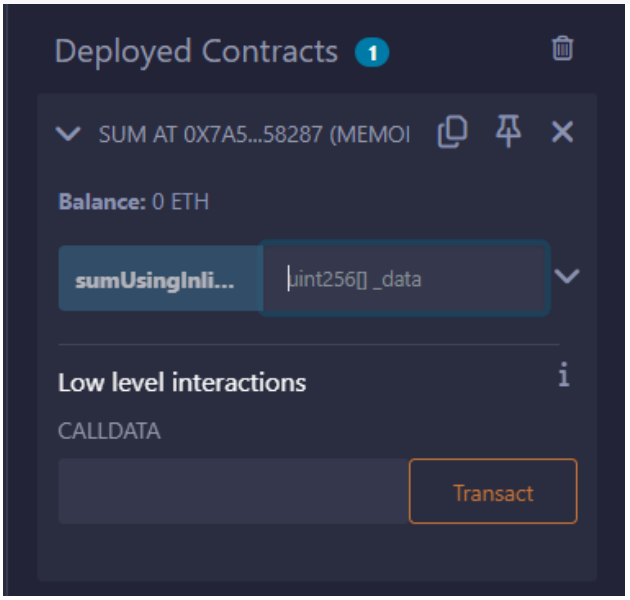
```
pragma solidity ^0.5.0;

library Search {
function indexOf(uint[] storage self, uint value) public view returns (uint) {
for (uint i = 0; i < self.length; i++)
if (self[i] == value) return i;
return uint(-1);}
}
contract Test {
uint[] data;
uint value;
uint index;
constructor() public {
data.push(6);
data.push(7);
data.push(8);
data.push(9);
data.push(10);
}
function isValuePresent() external {
value = 9;
//search if value is present in the array using Library function
index = Search.indexOf(data, value);
}
function getresult() public view returns(uint){
return index;
}}
```



Assembly:

Solidity provides an option to use assembly language to write inline assembly within Solidity source code. We can also write a standalone assembly code which then be converted to bytecode. Standalone Assembly is an intermediate language for a Solidity compiler and it converts the Solidity code into a Standalone Assembly and then to byte code. We can used the same language used in Inline Assembly to write code in a Standalone assembly.

```
pragma solidity ^0.5.0;

library Sum {
function sumUsingInlineAssembly(uint[] memory _data) public pure returns (uint o_sum) {
for (uint i = 0; i < _data.length; ++i) {
assembly {
o_sum := add(o_sum, mload(add(add(_data, 0x20), mul(i, 0x20))))
}}
}
}
contract Test {
uint[] data;
constructor() public {
data.push(1);
data.push(2);
data.push(3);
data.push(4);
data.push(5);
}
function sum() external view returns(uint){
return Sum.sumUsingInlineAssembly(data);
}
}
```



Events:

Event is an inheritable member of a contract. An event is emitted, it stores the arguments passed in transaction logs. These logs are stored on blockchain and are accessible using address of the contract till the contract is present on the blockchain. An event generated is not accessible from within contracts, not even the one which have created and emitted them.

// Solidity program to demonstrate

// creating an event

```
pragma solidity ^0.4.21;

// Creating a contract
contract eventExample {

// Declaring state variables
uint256 public value = 0;

// Declaring an event
event Increment(address owner);

// Defining a function for logging event
function getValue(uint _a, uint _b) public {
emit Increment(msg.sender);
value = _a + _b;
}
}
```

Error Handling:

Solidity provides various functions for error handling. Generally when an error occurs, the state is reverted back to its original state. Other checks are to prevent unauthorized code access.

Solidity program to demonstrate require statement.

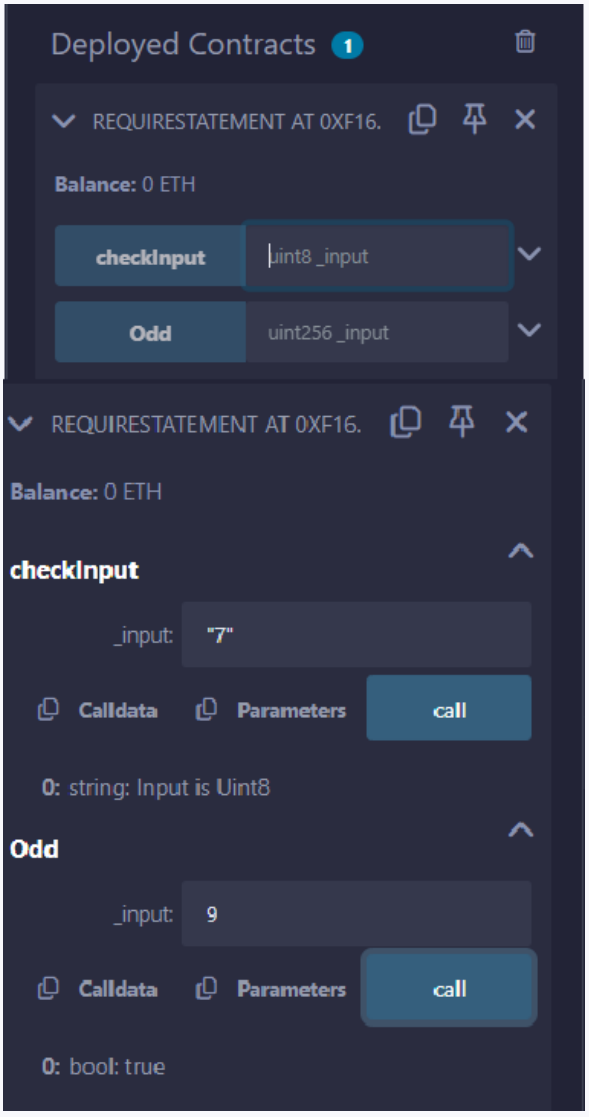
// Solidity program to

// demonstrate require

// statement

```
pragma solidity ^0.5.0;
// Creating a contract
contract requireStatement {
// Defining function to
// check input
function checkInput(uint8 _input) public view returns(string memory){
require(_input >= 0, "invalid uint");
require(_input <= 255, "invalid uint8");

return "Input is Uint8";
}
// Defining function to
// use require statement
function Odd(uint _input) public view returns(bool){
require(_input % 2 != 0);
```

```
return true;
}
}
```



Solidity program to demonstrate assert statement.

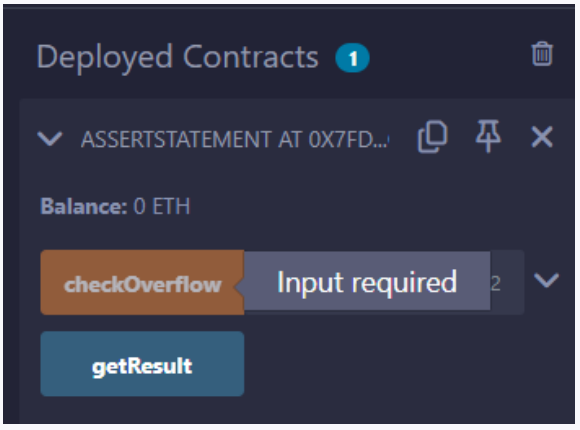
// Solidity program to

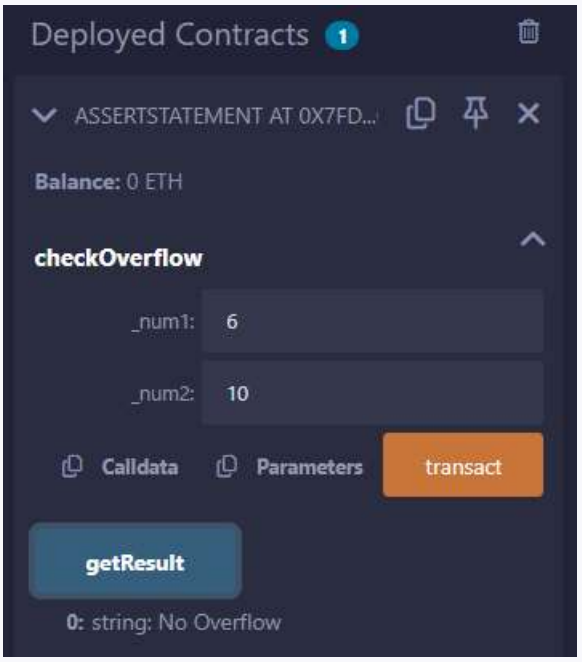
// demonstrate assert

// statement

```
pragma solidity ^0.5.0;

// Creating a contract
contract assertStatement {
// Defining a state variable
bool result;
// Defining a function
// to check condition
function checkOverflow(uint8 _num1, uint8 _num2) public {
uint8 sum = _num1 + _num2;
assert(sum<=255);
result = true;
}
```

```
// Defining a function to
// print result of assert
// statement
function getResult() public view returns(string memory){
if(result == true){
return "No Overflow";
}
else{
return "Overflow exist";
}
}
}
```





Solidity program to demonstrate revert statement.

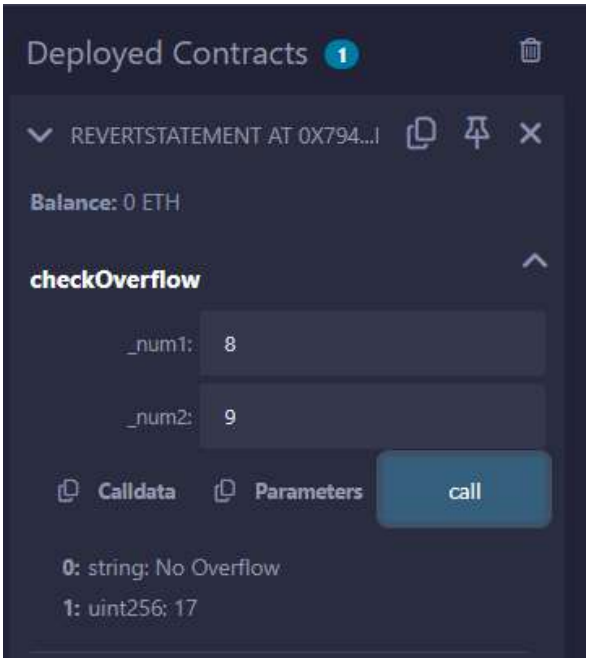
// Solidity program to

// demonstrate revert

```
pragma solidity ^0.5.0;
// Creating a contract
contract revertStatement {
// Defining a function
// to check condition
```

```
function checkOverflow(uint _num1, uint _num2) public view returns(
string memory, uint) {
uint sum = _num1 + _num2;
if(sum < 0 || sum > 255){
revert(" Overflow Exist");
}
else{
return ("No Overflow", sum);
}
}
}
```

https://www.tutorialspoint.com/solidity/index.htm



PRACTICAL 7

Aim:-Deploying a contracts on an external blockchain by using Ganache and/or MyEtherwalllet, Metamask

https://abhibvp003.medium.com/how-to-install-and-execute-truffle-on-an-ubuntu-16-04-7d0ff6458c9b

https://ethereum.stackexchange.com/questions/93533/call-an-existing-contract-function-from-truffle-

console

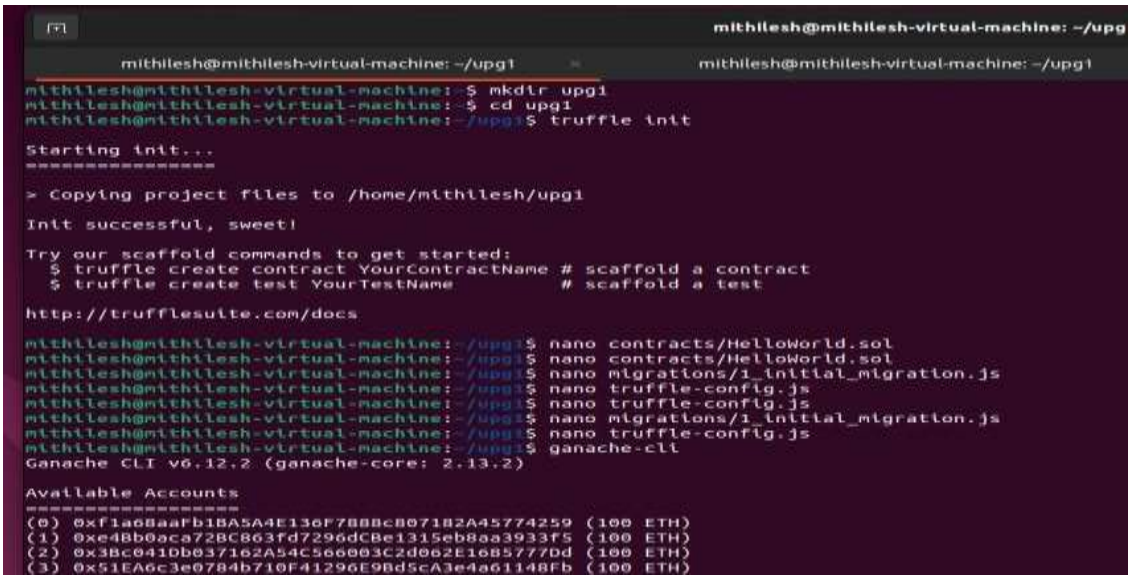
sudo apt-get -y install curl git vim build-essential
sudo apt-get install curl software-properties-
common

sudo apt install npm
sudo npm install -g
web3
sudo apt-get install nodejs
sudo apt install python3.9
curl -sL https://deb.nodesource.com/setup_10.x | sudo
bash - sudo npm install --global node-sass@latest
sudo npm install -g truffle@latest
sudo npm install -g ganache-cli
export NODE_OPTIONS=--openssl-legacy-provider

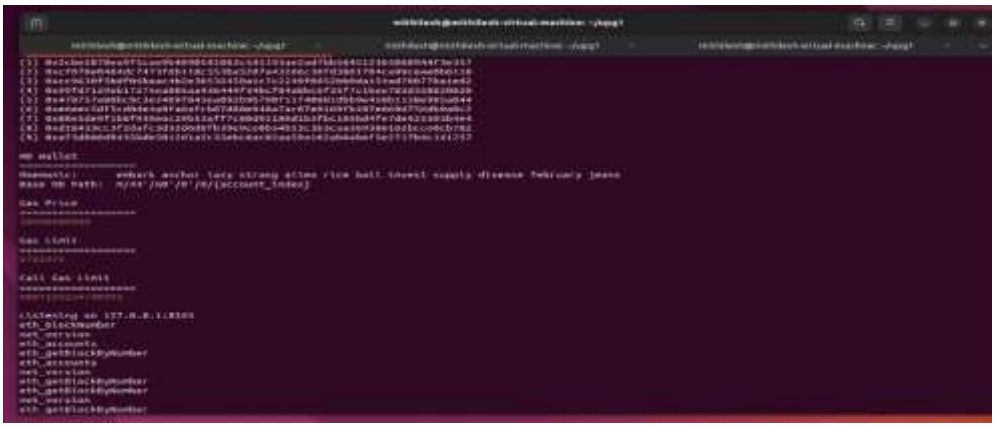
////to update npm//
sudo npm cache clean
-f sudo npm install -g
n
sudo n latest

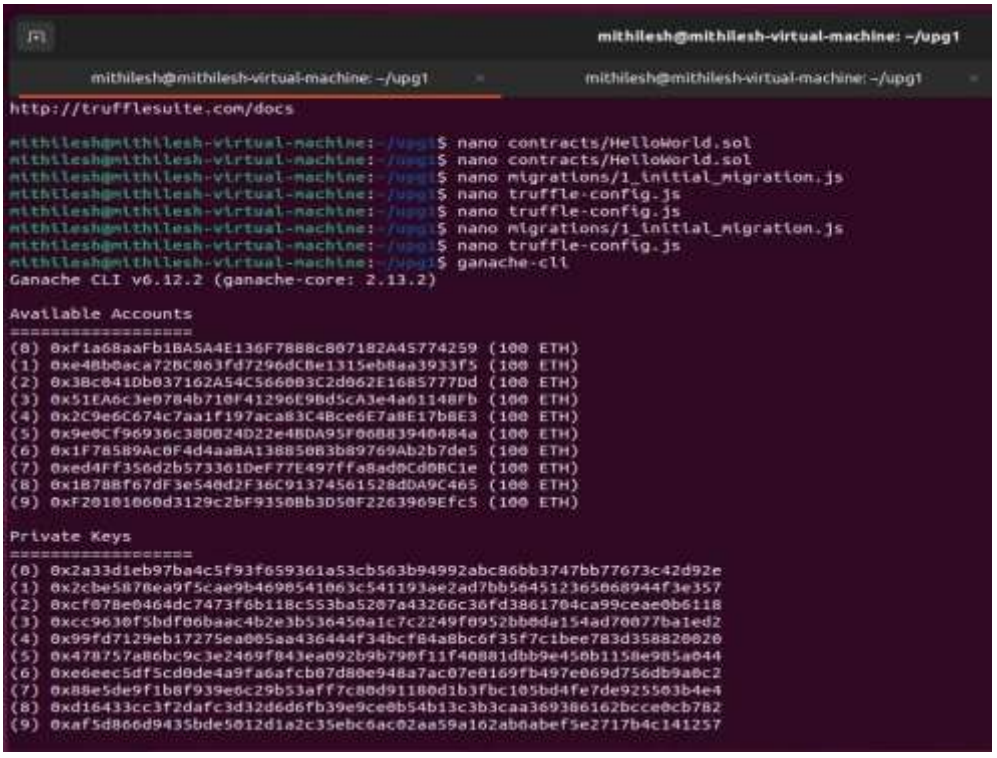
////////////////////////////////////
Start from here!!!

mkdir upg1
cd upg1
truffle init



////////// create contract
nano contracts/HelloWorld.sol





pragma solidity ^0.5.0; contract HelloWorld {
    function sayHello() public pure returns(string memory){
        return("hello world");
    }
}

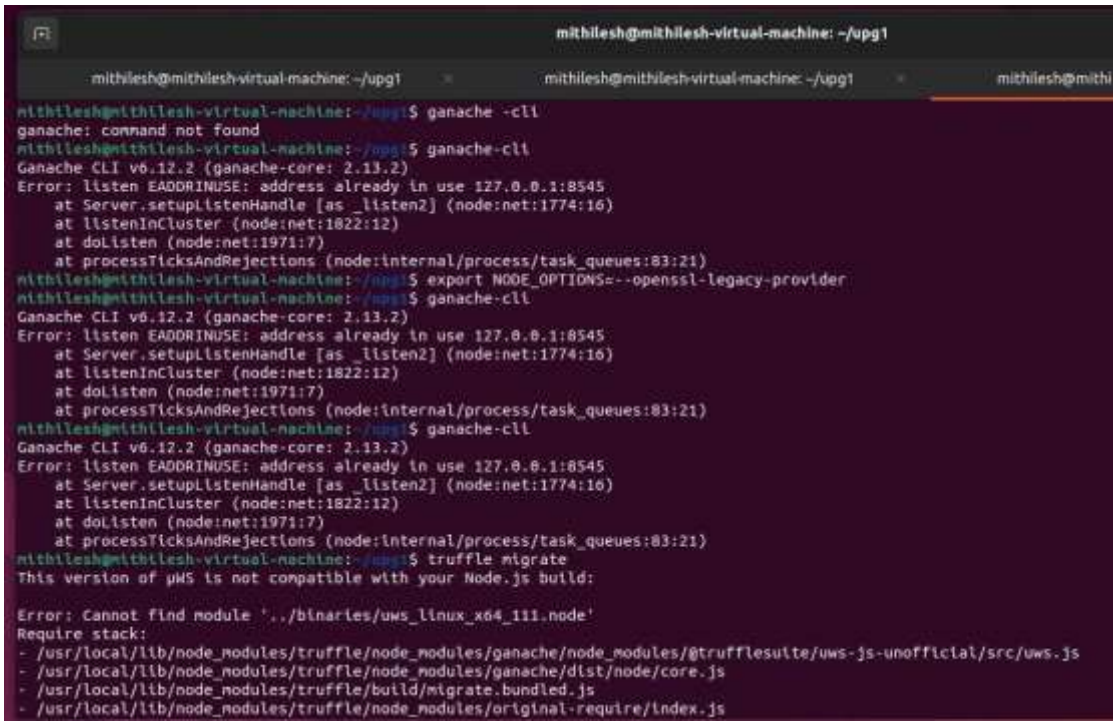
//////////////////create configuration
nano migrations/1_initial_migration.js
const Migrations = artifacts.require("HelloWorld");

module.exports = function (deployer) {
deployer.deploy(Migrations,"hello");
};

///////////////network configuration
nano truffle-config.js
module.exports = {
networks: {
development: {
host: "127.0.0.1",
port: 8545,
network_id: "*",
}
}
}

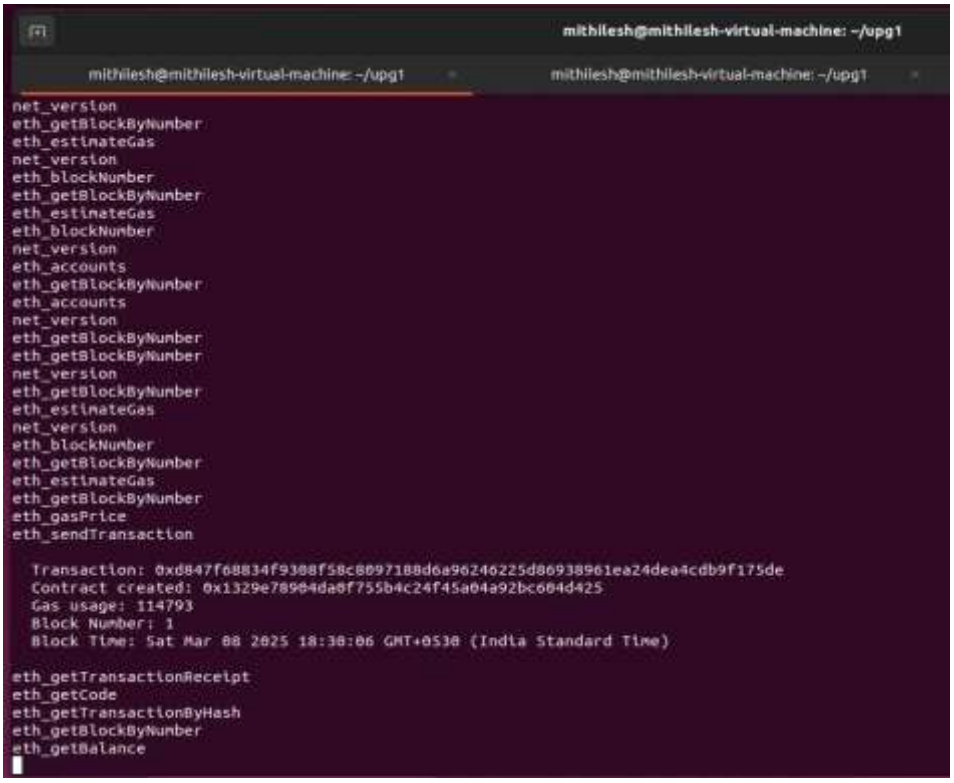
//////////////////start ganache-cli

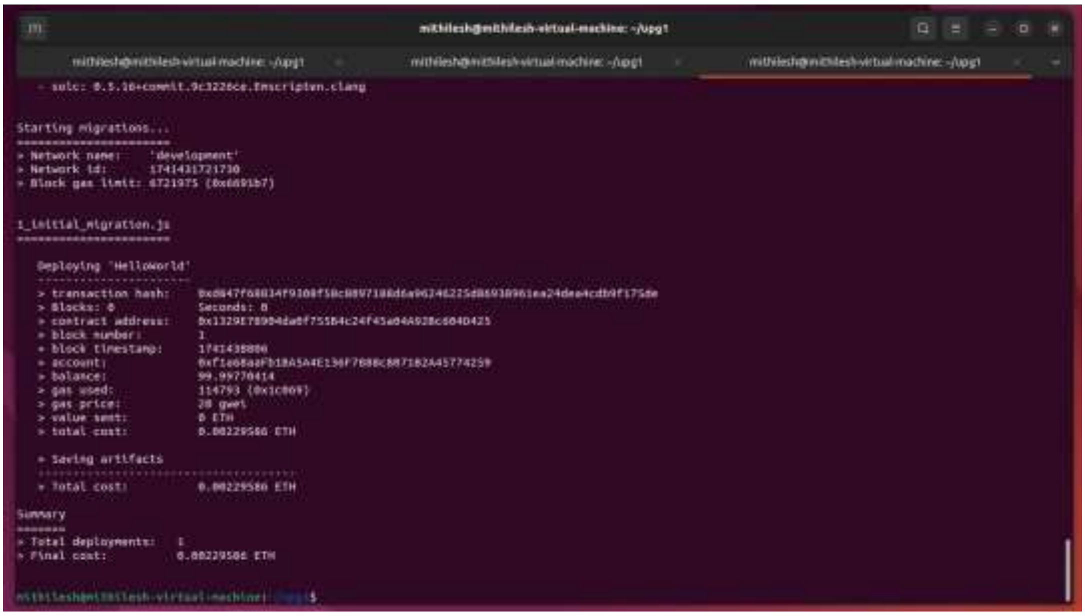
ganache-cli



////////////////
truffle migrate

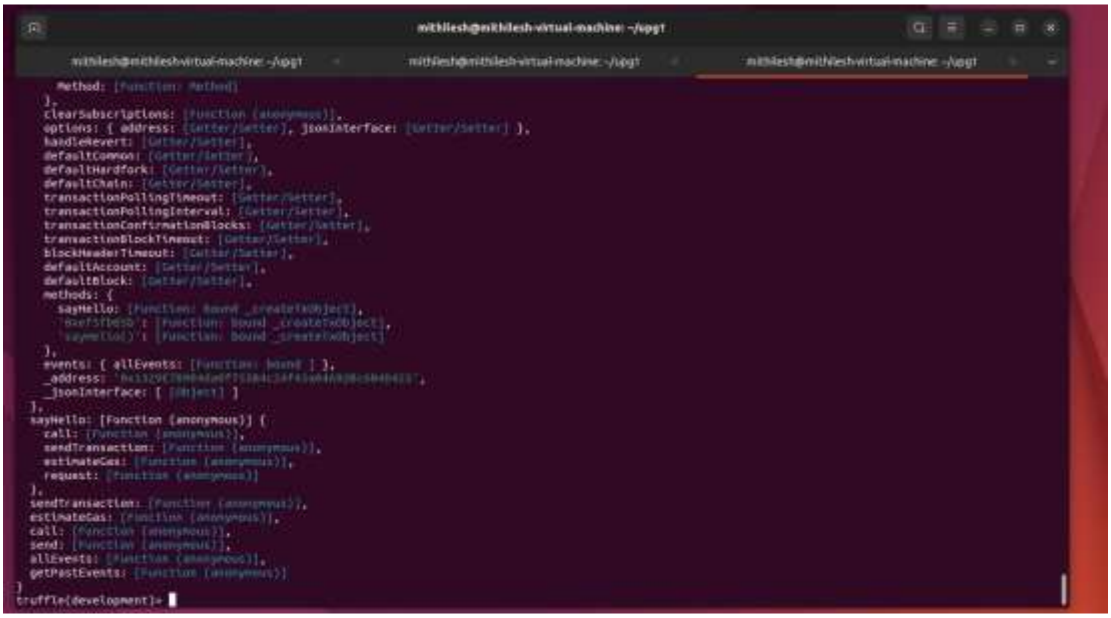
truffle console
#replace contact address
contract = await HelloWorld.at('0x37354B83aadd35516c56f24b724228f29300be77')
a = await contract.sayHello()
a

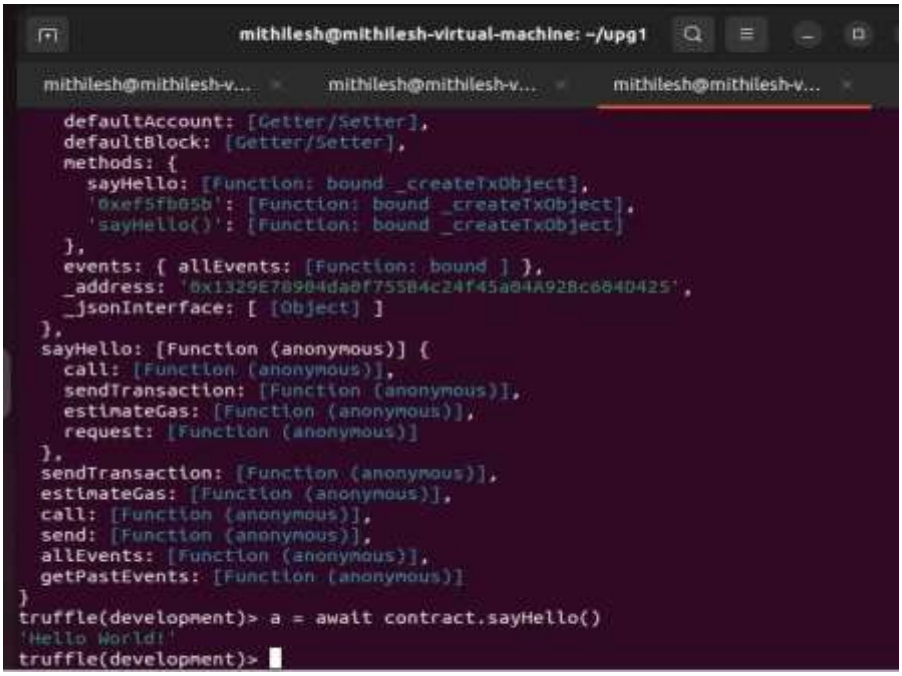



mithilesh@mithilesh-virtual-machine: ~/upg1
- solc: 0.5.16+commit.9c3226ce.Emscripten.clang
Starting migrations...
======================
> Network name: 'development'
> Network id: 1741431721730
> Block gas limit: 6721975 (0x6691b7)
1_initial_migration.js
======================
Deploying 'HelloWorld'

> transaction hash: 0xd847f68834f9308f58c8897188d6a96246225d8693896 1ea24dea4cdb9f175de
> Blocks: 0   Seconds: 0
> contract address: 0x1329E78904da0f755B4c24f45a04A928c6840425
> block number: 1
> block timestamp: 1741438800
> account: 0xf1e68aaFb18A5A4E136F7888c887182A45774259
> balance: 99.99770414
> gas used: 114793 (0x1c069)
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.00229586 ETH
> Saving artifacts

> Total cost: 0.00229586 ETH
Summary
=======
> Total deployments: 1
> Final cost: 0.00229586 ETH
mithilesh@mithilesh-virtual-machine: ~/upg1$


mithilesh@mithilesh-virtual-machine: ~/upg1
method: [Function: method]
clearSubscriptions: [Function (anonymous)],
options: { address: [Getter/Setter], jsonInterface: [Getter/Setter] },
handleRevert: [Getter/Setter],
defaultCommon: [Getter/Setter],
defaultHardfork: [Getter/Setter],
defaultChain: [Getter/Setter],
transactionPollingTimeout: [Getter/Setter],
transactionPollingInterval: [Getter/Setter],
transactionConfirmationBlocks: [Getter/Setter],
transactionBlockTimeout: [Getter/Setter],
blockHeaderTimeout: [Getter/Setter],
defaultAccount: [Getter/Setter],
defaultBlock: [Getter/Setter],
methods: {
sayHello: [Function: bound _createTxObject],
sayHello: [Function (anonymous)] {
call: [Function (anonymous)],
sendTransaction: [Function (anonymous)],
estimateGas: [Function (anonymous)],
request: [Function (anonymous)]
sendTransaction: [Function (anonymous)],
estimateGas: [Function (anonymous)],
call: [Function (anonymous)],
send: [Function (anonymous)],
allEvents: [Function (anonymous)],
getPastEvents: [Function (anonymous)]
truffle(development)>


mithilesh@mithilesh-virtual-machine: ~/upg1
defaultAccount: [Getter/Setter],
defaultBlock: [Getter/Setter],
methods: {
sayHello: [Function: bound _createTxObject],
'0xef5fb05b': [Function: bound _createTxObject],
'sayHello()': [Function: bound _createTxObject]
},
events: { allEvents: [Function: bound] },
_address: '0x1329E78904da0f755B4c24f45a04A928c6840425',
_jsonInterface: [[Object]]
},
sayHello: [Function (anonymous)] {
call: [Function (anonymous)],
sendTransaction: [Function (anonymous)],
estimateGas: [Function (anonymous)],
request: [Function (anonymous)]
},
sendTransaction: [Function (anonymous)],
estimateGas: [Function (anonymous)],
call: [Function (anonymous)],
send: [Function (anonymous)],
allEvents: [Function (anonymous)],
getPastEvents: [Function (anonymous)]
}
truffle(development)> a = await contract.sayHello()
'Hello World!'
truffle(development)>

PRACTICAL 8

Aim:-Create your own blockchain and demonstrate its use.Deploy a local private blockchain over a network with Ethereum or Rust (VM)

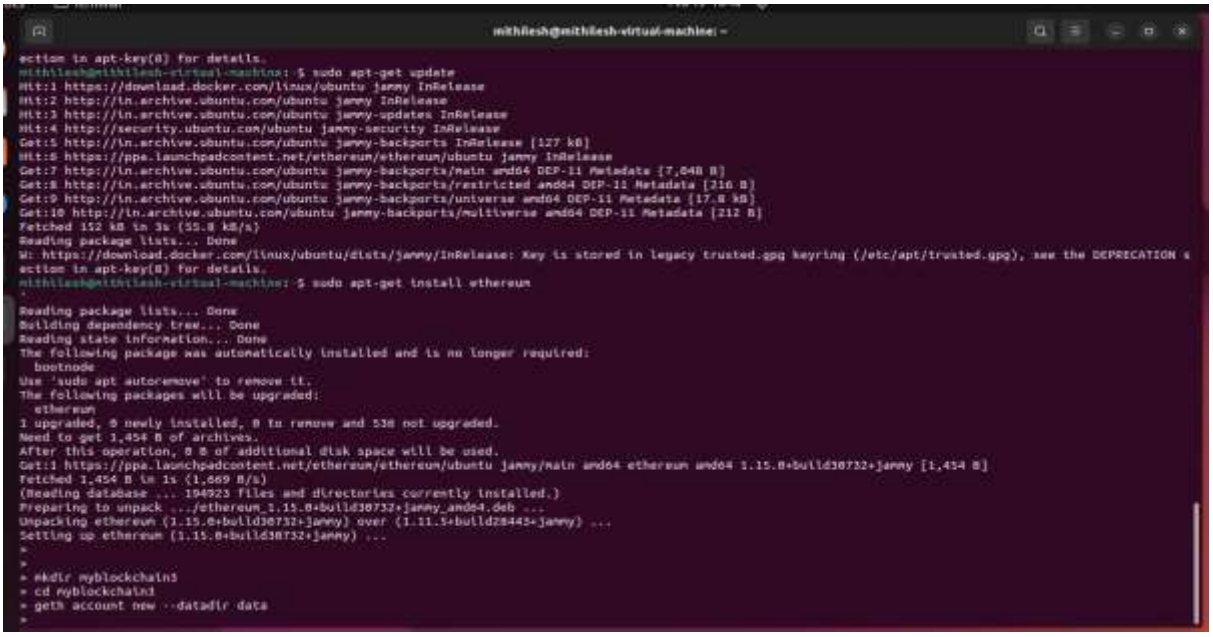
Install on Ubuntu via PPAs
The easiest way to install go-ethereum on Ubuntu-based distributions is with the built-in launchpad PPAs (Personal Package Archives). We provide a single PPA repository that contains both our stable and development releases for Ubuntu versions trusty, xenial, zesty and artful.

linux:
To enable our launchpad repository
run: Step 1: open new terminal
Step 2: on terminal type this command
sudo add-apt-repository -y
ppa:ethereum/ethereum #if above command
gives error then run
#sudo apt-get install --reinstall ca-certificates

Step 3: install the stable version of go-ethereum:
sudo apt-get update

sudo apt-get install Ethereum

Step 4: create new directory for storing blockchain data
mkdir myblockchain2
cd myblockchain2
geth account new --datadir data

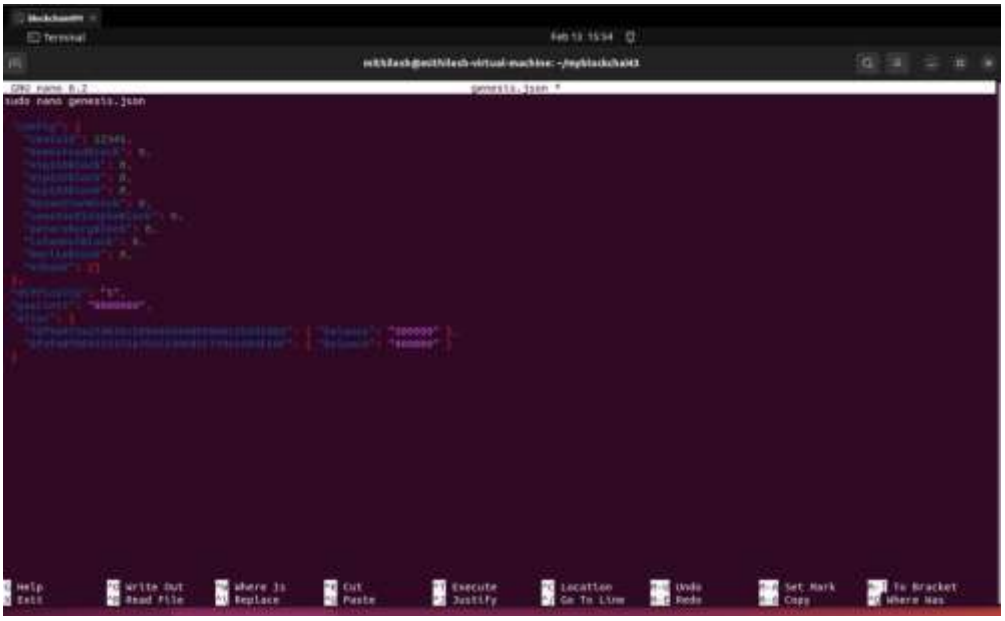


Step 5:Create genesis.json file

sudo nano genesis.json

```
{
  "config": {
    "chainId": 12345,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip155Block": 0,
    "eip158Block": 0,
    "byzantiumBlock": 0,
    "constantinopleBlock": 0,
    "petersburgBlock": 0,
    "istanbulBlock": 0,
    "berlinBlock": 0,
    "ethash": {}
  },
  "difficulty": "1",
  "gasLimit": "8000000",
  "alloc": {
    "7df9a875a174b3bc565e6424a0050ebc1b2d1d82": { "balance": "300000" },
    "Efaf4df069211972a7D2C3306d1F778a1603F10F": { "balance": "400000" }
  }
}
```

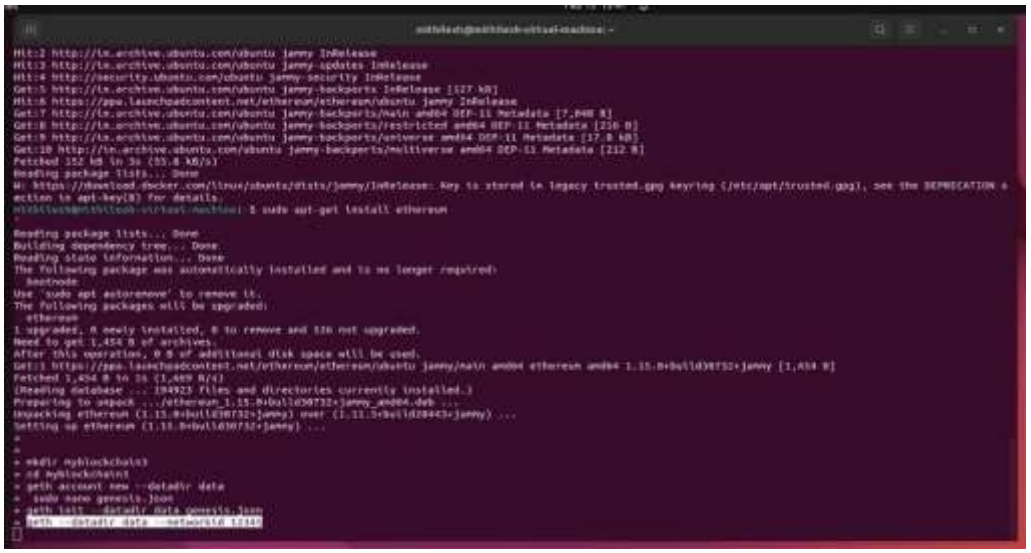
save the file -> ctrl +o to write -> {enter} save -> ctrl +x exit



Step 6: initialize the block
geth init --datadir data genesis.json
Step 7: create network
geth --datadir data --networkid 12345

[do not close this terminal]

//

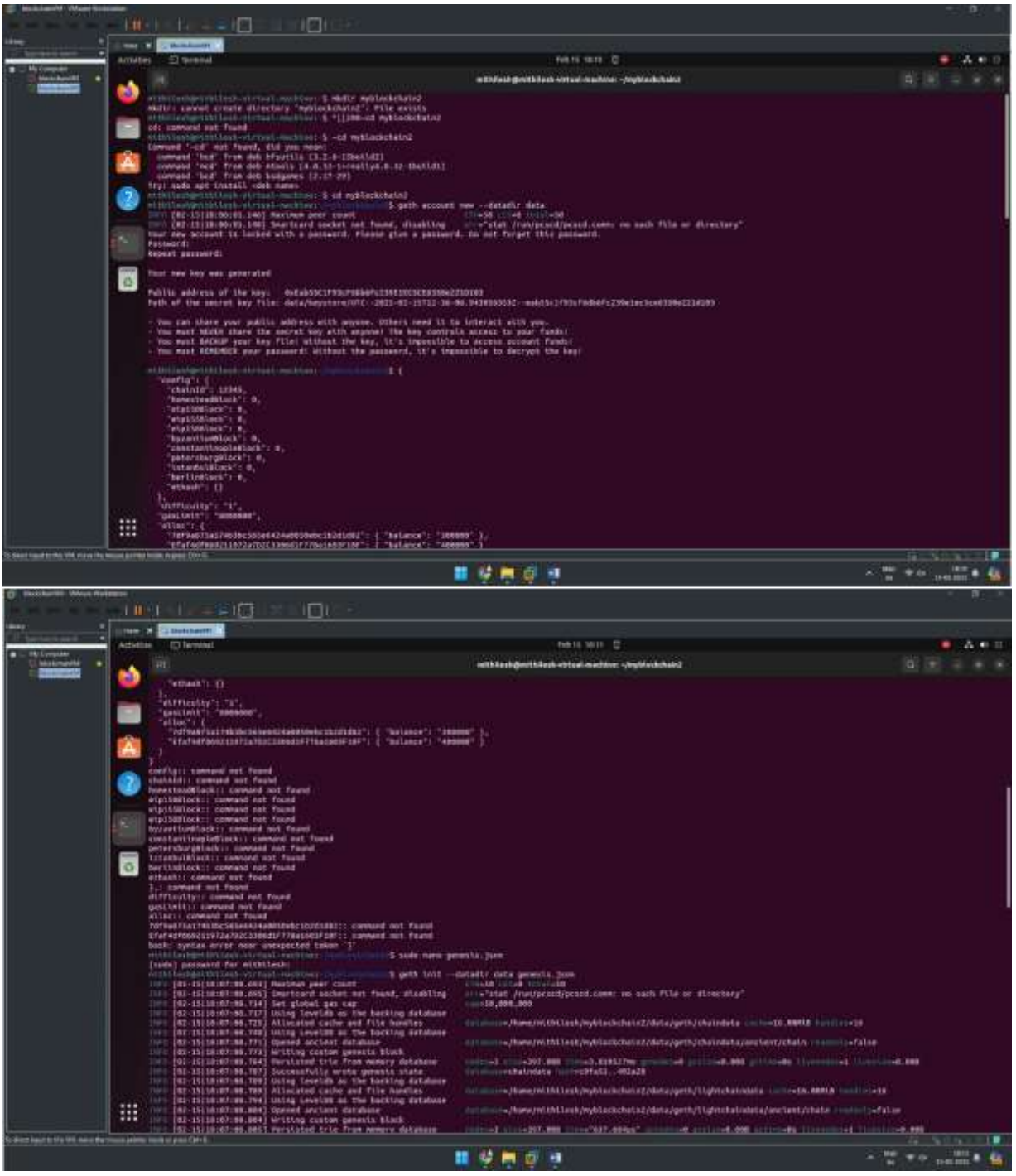


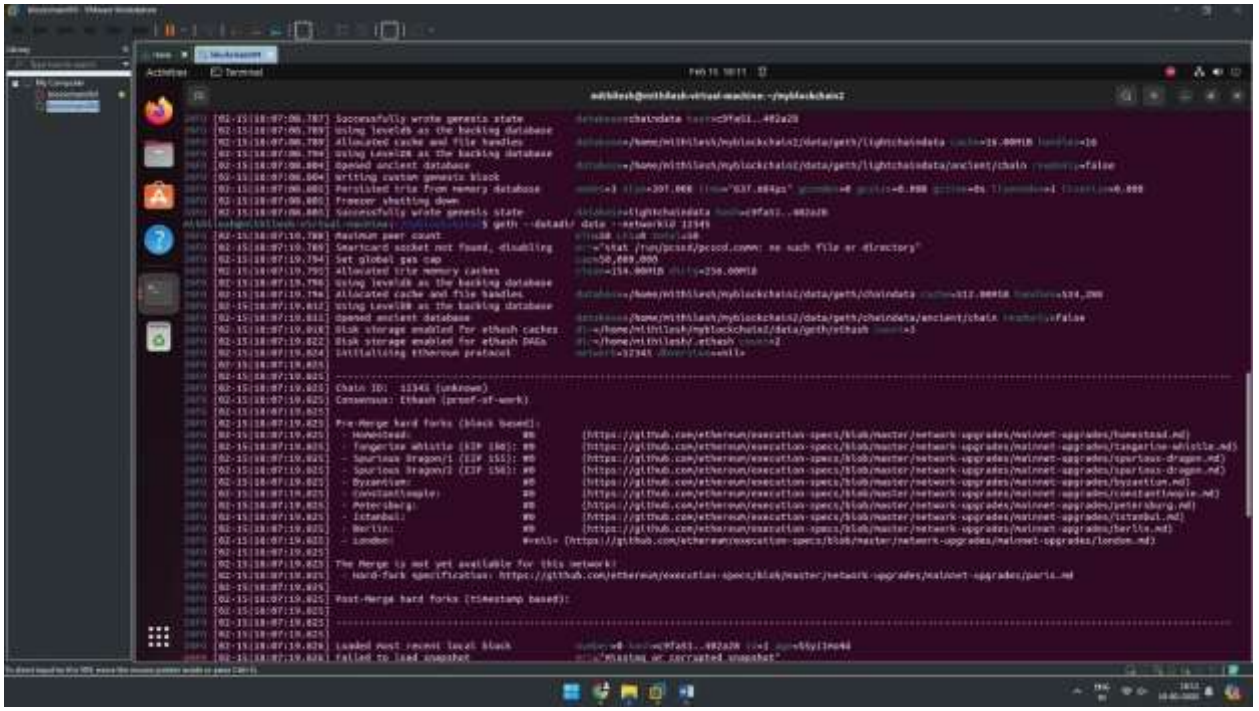
Step 8: open new tab/terminal 2:
sudo geth attach data/geth.ipc
eth.getBalance(eth.accounts[0])
miner.setEtherbase(eth.accounts[0])
miner.start()
admin.addPeer(admin.nodeInfo.enode)
eth.getBalance(eth.accounts[0])

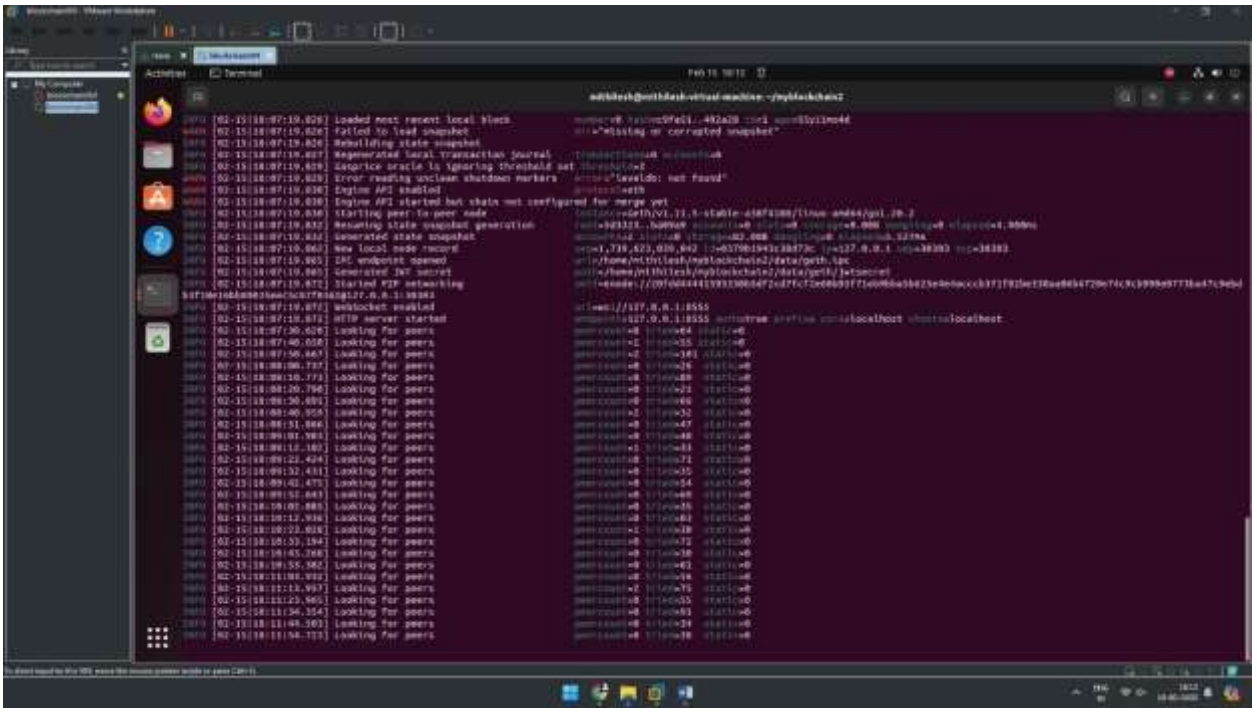
Step 10: Wait for 10-20 minutes and check balance
eth.getBalance(eth.accounts[0])

if ether balance is 0 wait for 10-20minutes for mining process to get complete and run eth.getBalance(eth.accounts[0]) again.

After balance is updated you can check current block height
eth.blockNumber

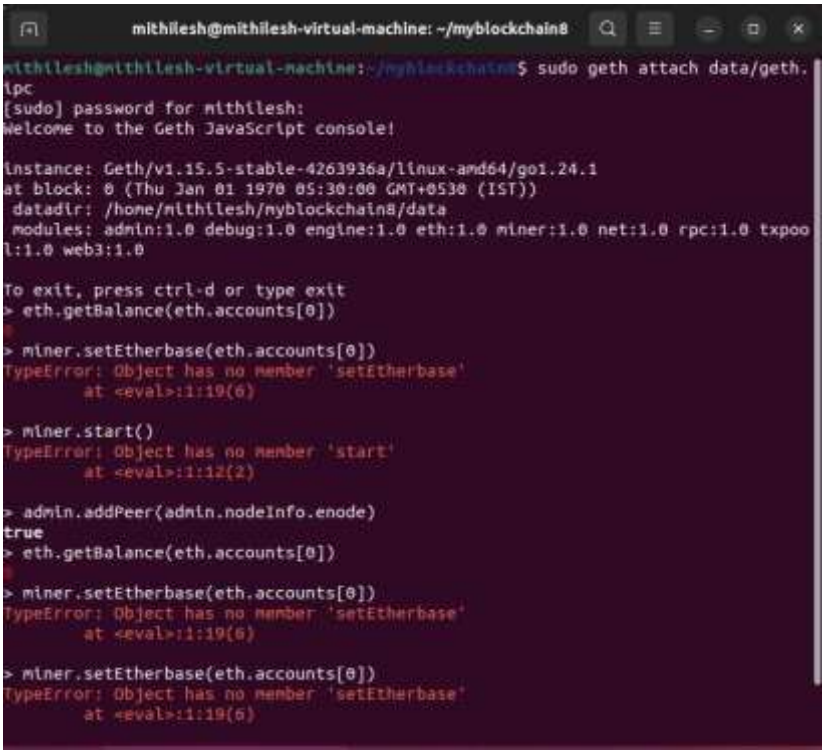


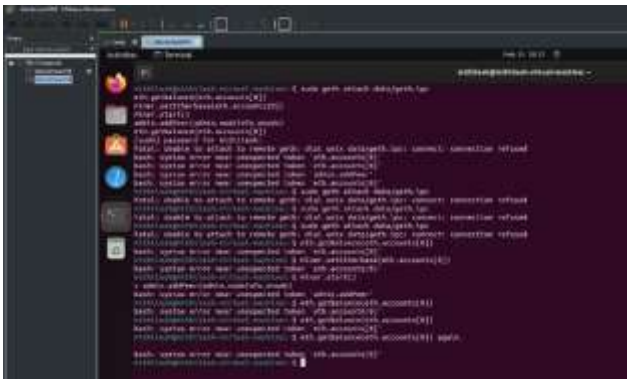




Terminal 2 :

Output:-





Geth has transitioned to Proof-of-Stake (PoS), so mining is disabled.The miner.setEtherbase() and miner.start() commands are only available in older Proof-of-Work (PoW) versions of Geth.In the latest Geth versions (v1.14+), you cannot mine new blocks with geth

PRACTICAL 9

Aim:- Implement the mining module of Bitcoin client . The mining module, or miner, should produce blocks that solve proof-of-work puzzle
Code:-Open Python IDLE and create new Script.

##########################

```
from bitcoinlib.wallets import Wallet
w = Wallet.create('Wallet1')
key1 = w.get_key()
print('Wallet Address:',key1.address)
w.scan()
print(w.info())
```

```
Wallet Address: bc1qppnqpg9quay7qf5hzh2ekx2cu0g8tmky666u5n
=== WALLET ===
 ID                          1
 Name                        Wallet1
 Owner
 Scheme                      bip32
 Multisig                    False
 Witness type                segwit
 Main network                bitcoin
 Latest update               2025-02-17 04:17:53.572643+00:00

= Wallet Master Key =
 ID                          1
 Private                     True
 Depth                       0

- NETWORK: bitcoin -
- - Keys
     6 m/84'/0'/0'/0/0         bc1qppnqpg9quay7qf5hzh2ekx2cu0g8tmky666u5n    address index 0                0.00000000 ฿
     7 m/84'/0'/0'/0/1         bc1qy762wx0jqp9y6psekuhg6yn2h0z2ry7gavymra    address index 1                0.00000000 ฿
     9 m/84'/0'/0'/0/2         bc1q0lvwve045pl0q9hdvucwk8h5dmvu88ggrdevyj    address index 2                0.00000000 ฿
    10 m/84'/0'/0'/0/3         bc1ql33pg2mjzempsyzxme740xzzsln7fc7hhry4fl    address index 3                0.00000000 ฿
    11 m/84'/0'/0'/0/4         bc1q4p3fhm8r7sjwzhhq9lxrp5wg0thqpn2k3xj5fe    address index 4                0.00000000 ฿
    13 m/84'/0'/0'/1/0         bc1qkcuslwjmd2uhgs9auwua0xgydt4sc6jrjlqesx    address index 0                0.00000000 ฿
    15 m/84'/0'/0'/1/1         bc1qx29svsg7he7l2nqpfk6y4k0x5hk5zev9r0nvv5    address index 1                0.00000000 ฿
    16 m/84'/0'/0'/1/2         bc1qeh7dpxunzd55rc35tgg7qr2c4krwhn70pvg2wv    address index 2                0.00000000 ฿
    17 m/84'/0'/0'/1/3         bc1q3cwgrkmrclprvdk9caqpfjhnyszinhhw0pag3    address index 3                0.00000000 ฿
    18 m/84'/0'/0'/1/4         bc1q3wd0tnqsy7a5u7rugz07de54cwxd8vdswn2fxr    address index 4                0.00000000 ฿

- - Transactions Account 0 (0)

= Balance Totals (includes unconfirmed) =


None
```

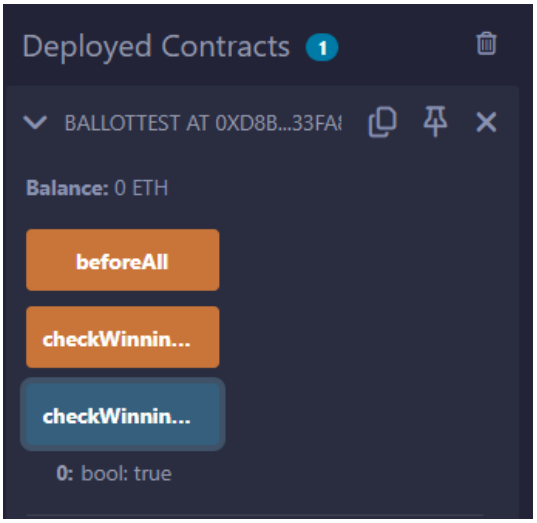
Open CMD and install bitcoinlib package
pip install bitcoinlib

PRACTICAL 10

Aim:-Compile and test smart contracts on a testing framework using the Ethereum Virtual Machine (EVM).
Code:-

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;
import "remix_tests.sol"; // this import is automatically injected by Remix.
import "hardhat/console.sol";
import "../contracts/3_Ballot.sol";

contract BallotTest {

    bytes32[] proposalNames;

    Ballot ballotToTest;
    function beforeAll () public {
        proposalNames.push(bytes32("candidate1"));
        ballotToTest = new Ballot(proposalNames);
    }

    function checkWinningProposal () public {
        console.log("Running checkWinningProposal");
        ballotToTest.vote(0);
        Assert.equal(ballotToTest.winningProposal(), uint(0), "proposal at index 0 should be the winning
proposal");
        Assert.equal(ballotToTest.winnerName(), bytes32("candidate1"), "candidate1 should be the winner
name");
    }

    function checkWinninProposalWithReturnValue () public view returns (bool) {
        return ballotToTest.winningProposal() == 0;
    }
}
```

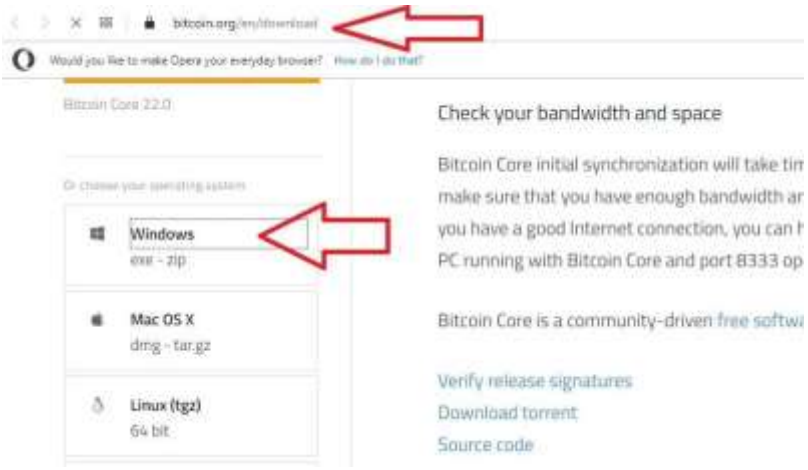
Output:-



PRACTICAL 11

Aim:-**Demonstrate the use of Bitcoin Core API.**
Step 1: Visit: https://bitcoin.org/en/download

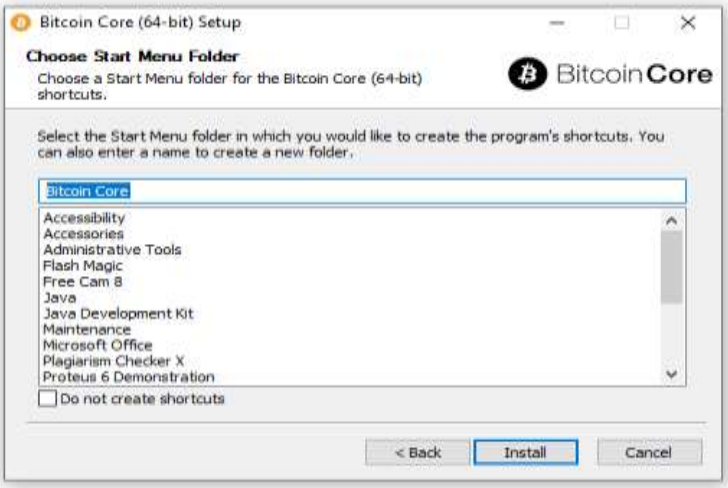
Step 2: Download windows setup [use and try with Linux version as well]

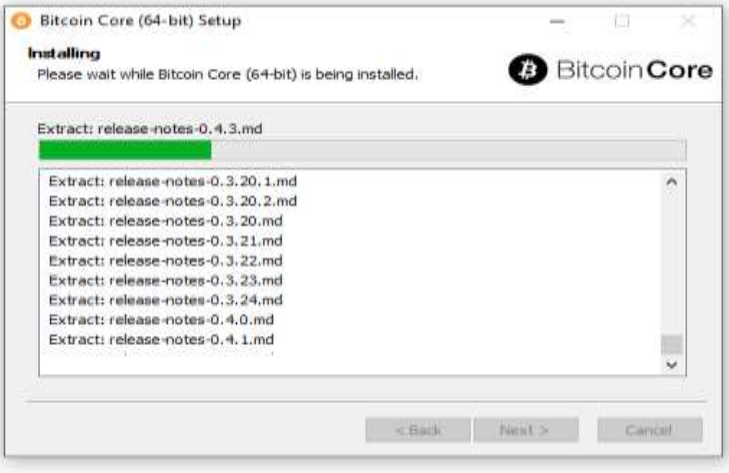


Step 3: Run the setup file-> click next

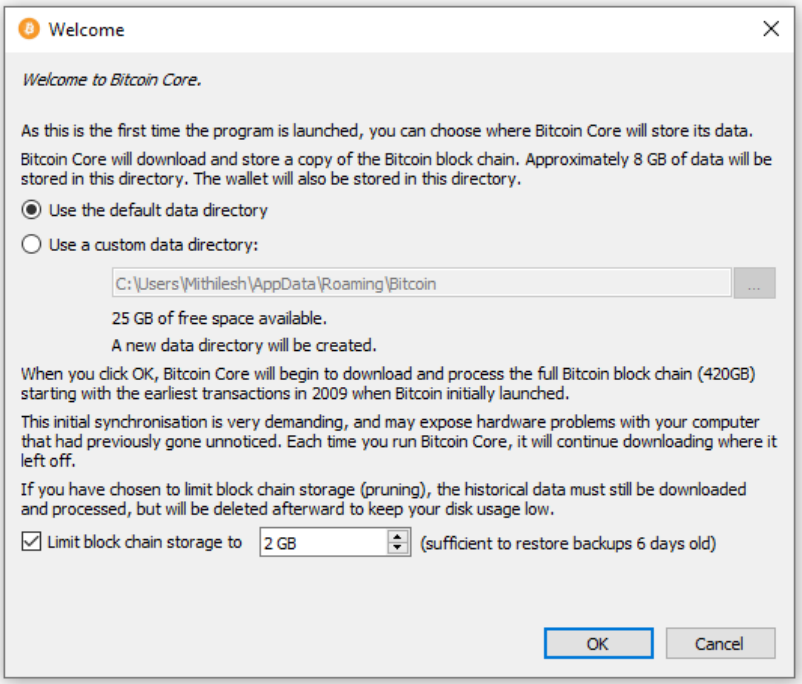
Step 4: Click Next

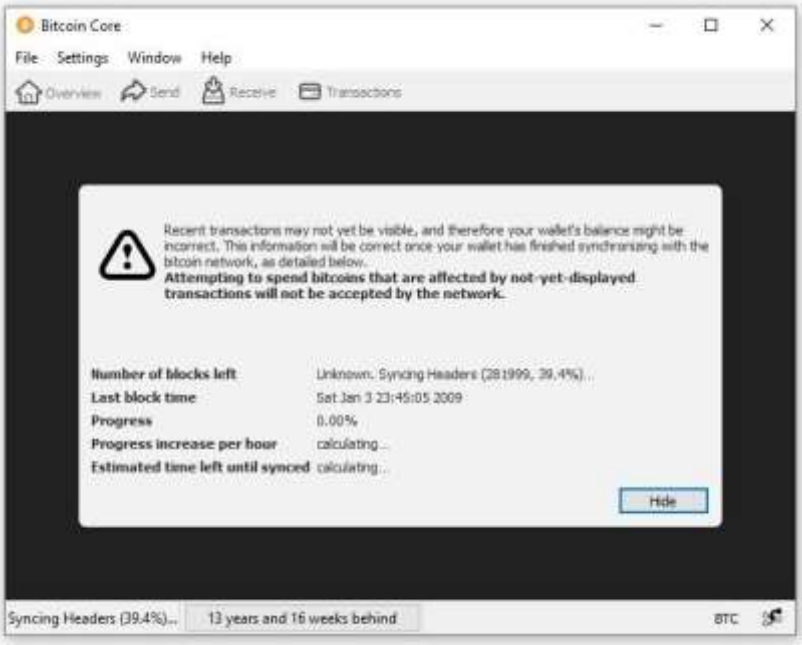
Step 5: Finally click on Install



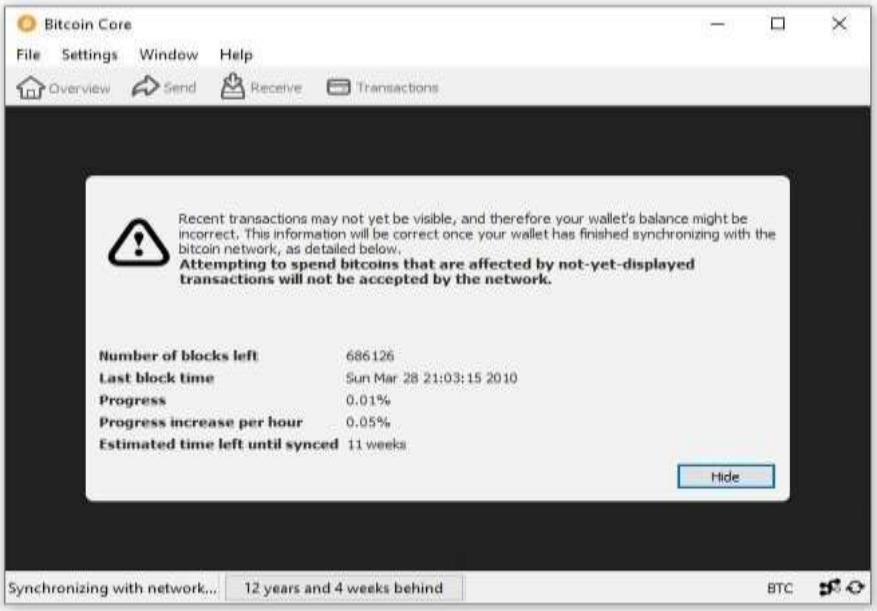


Launch Bitcoin Core-> Click OK.

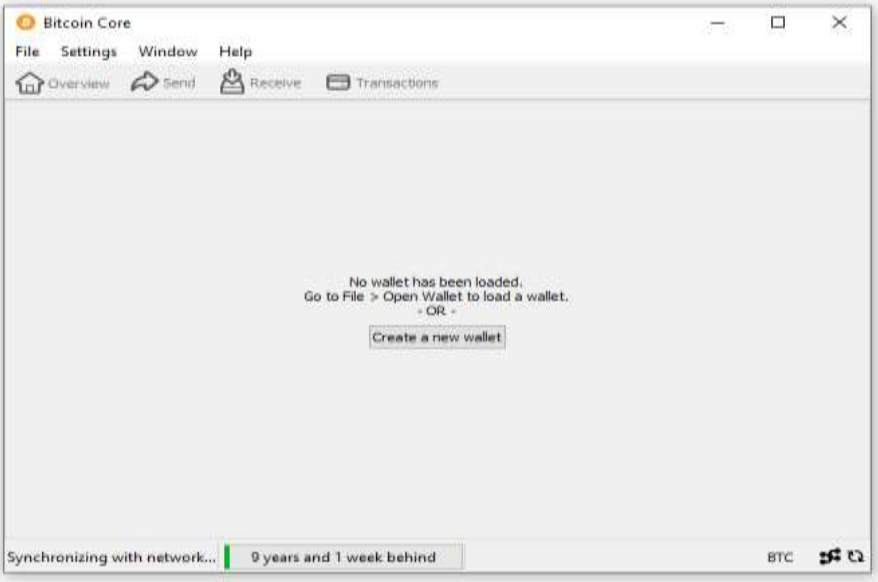




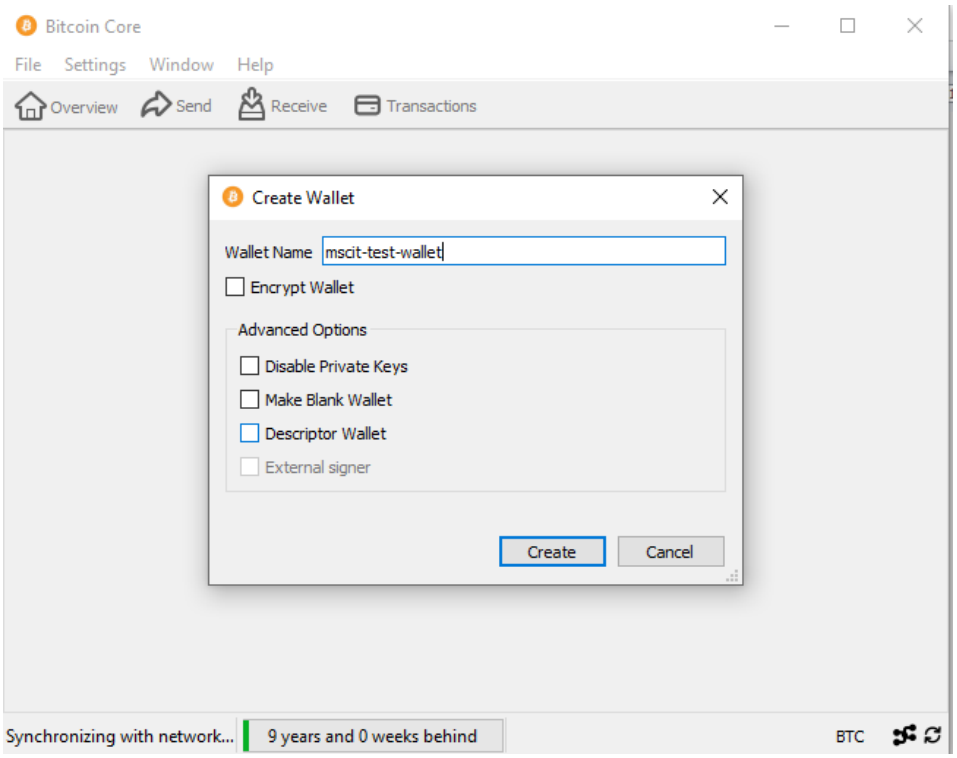
Click on Hide button [Synchronization take place in background]



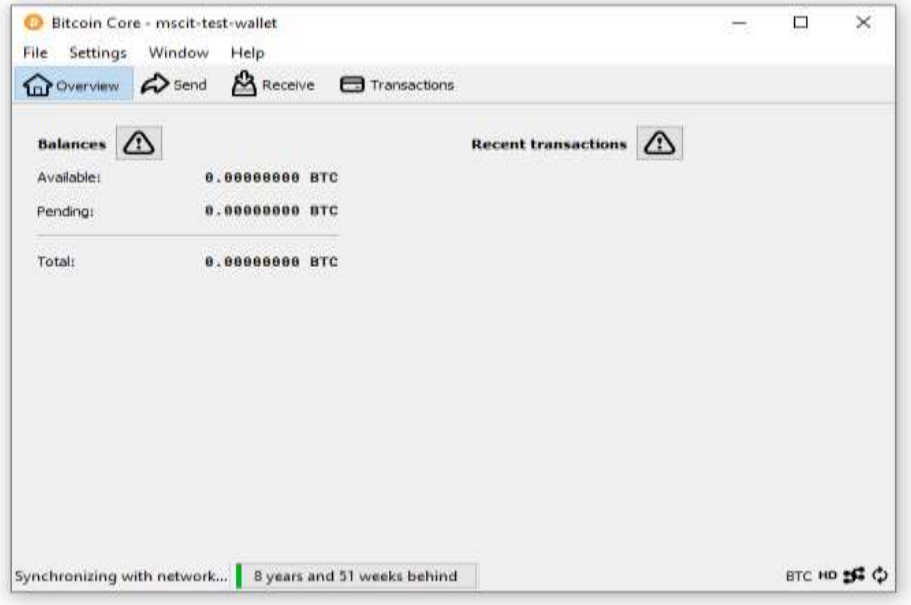
You can create a wallet -> Create a new wallet

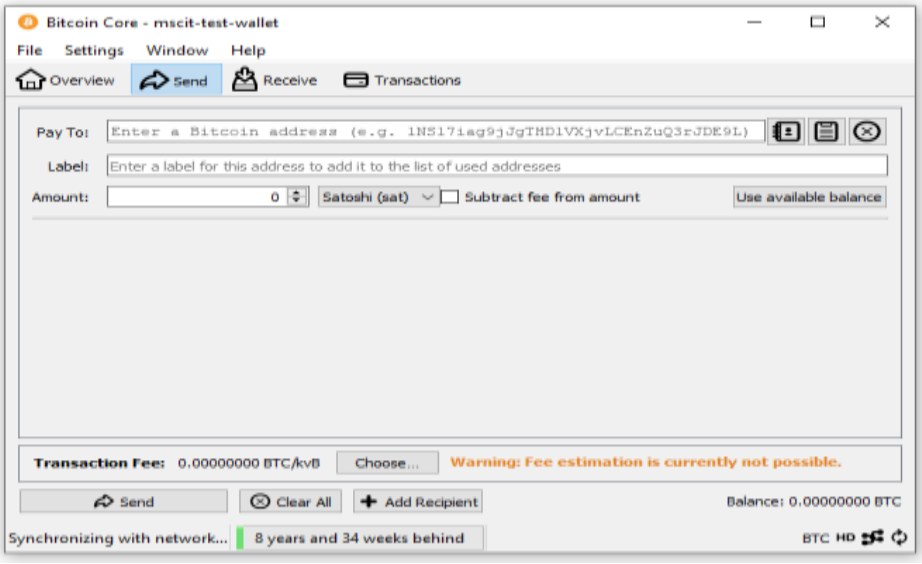


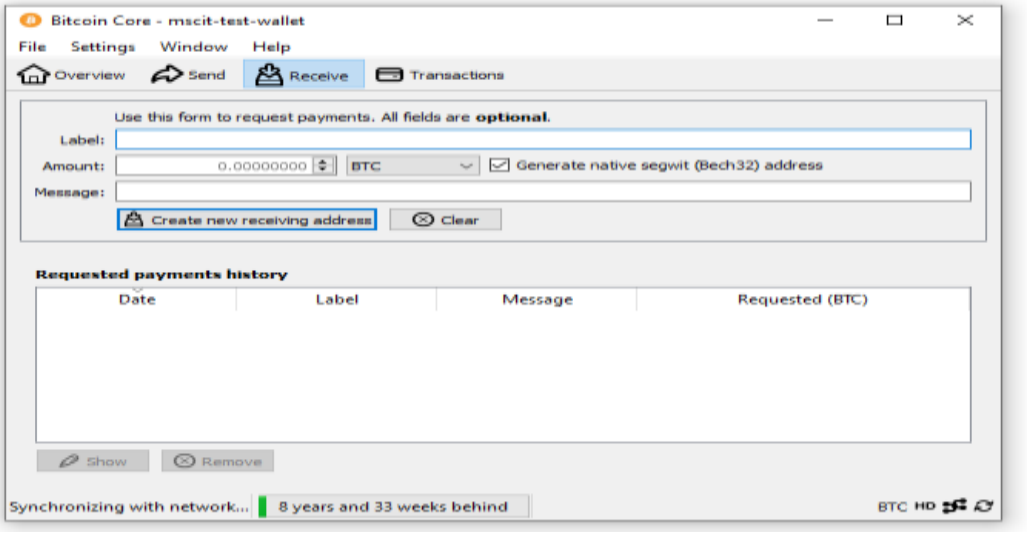
Enter Wallet name

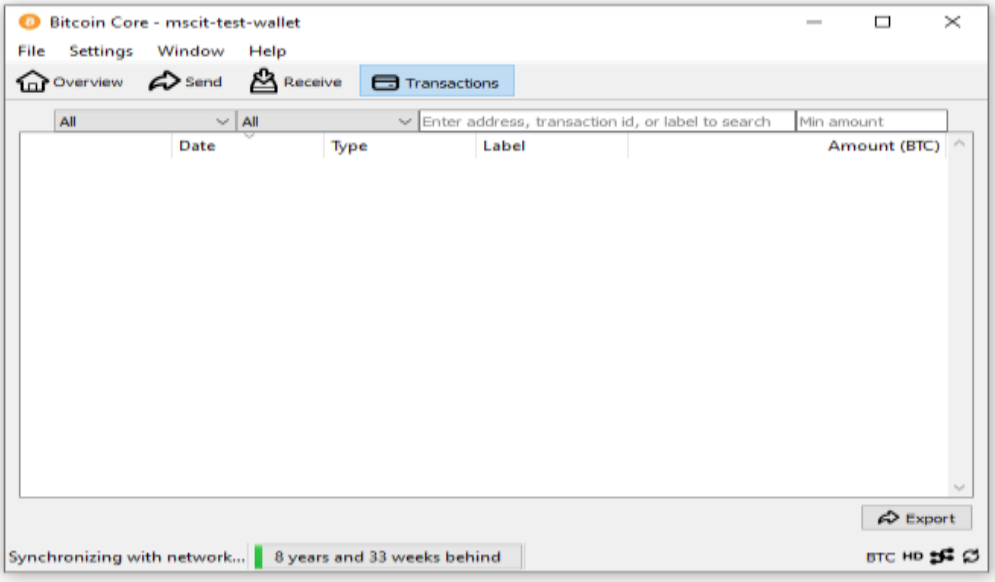


Finally Account is setup




Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
Pay To: Enter a Bitcoin address (e.g. 1NS17iag9jJgTHD1VXjvLCEnZuQ3rJDE9L)
Label: Enter a label for this address to add it to the list of used addresses
Amount: 0 Satoshi (sat) Subtract fee from amount Use available balance
Transaction Fee: 0.00000000 BTC/kvB Choose... Warning: Fee estimation is currently not possible.
Send Clear All Add Recipient Balance: 0.00000000 BTC
Synchronizing with network... 8 years and 34 weeks behind
BTC HD


Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
Use this form to request payments. All fields are optional.
Label:
Amount: 0.00000000 BTC Generate native segwit (Bech32) address
Message:
Create new receiving address Clear
Requested payments history
Date Label Message Requested (BTC)
Show Remove
Synchronizing with network... 8 years and 33 weeks behind
BTC HD


Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
All All Enter address, transaction id, or label to search Min amount
Date Type Label Amount (BTC)
Export
Synchronizing with network... 8 years and 33 weeks behind
BTC HD

PRACTICAL 12

Aim:-Create your own blockchain and demonstrate its use.

Code:-

```
# following imports are required by PKI
import hashlib
import random
import binascii
import datetime
import collections

from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5

class Client:
    def __init__(self):
      random = Random.new().read
      self._private_key = RSA.generate(1024, random)
    self._public_key = self._private_key.publickey()
    self._signer = PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
      return binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')

class Transaction:
    def _init_(self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()

    def to_dict(self):
        if self.sender == "Genesis":
            identity = "Genesis"
        else:
            identity = self.sender.identity

        return collections.OrderedDict({
            'sender': identity,
            'recipient': self.recipient,
            'value': self.value,
            'time' : self.time})

    def sign_transaction(self):
```

```
        private_key = self.sender._private_key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode('utf8'))
        return binascii.hexlify(signer.sign(h)).decode('ascii')

def display_transaction(transaction):
        #for transaction in transactions:
        dict = transaction.to_dict()
        print ("sender: " + dict['sender'])
        print (' ----- ')
        print ("recipient: " + dict['recipient'])
        print (' ----- ')
        print ("value: " + str(dict['value']))
        print (' ----- ')
        print ("time: " + str(dict['time']))
        print (' ----- ')

def dump_blockchain (self):
    print ("Number of blocks in the chain: " + str(len (self)))
    for x in range (len(TPCoins)):
        block_temp = TPCoins[x]
        print ("block # " + str(x))
        for transaction in block_temp.verified_transactions:
            display_transaction (transaction)
            print ('-----------------')
        print ('====================================')

class Block:
    def _init_(self):
        self.verified_transactions = []
        self.previous_block_hash = ""
        self.Nonce = ""

def sha256(message):
        return hashlib.sha256(message.encode('ascii')).hexdigest()

def mine(message, difficulty=1):
    assert difficulty >= 1
    #if(difficulty <1):
    #       return
    #'1'*3=> '111'
    prefix = '1' * difficulty
    for i in range(1000):
        digest = sha256(str(hash(message)) + str(i))
        if digest.startswith(prefix):
            return i #i= nonce value

A = Client()
B =Client()
C =Client()
t0 = Transaction (
    "Genesis",
    A.identity,
```

```
   500.0
)

t1 = Transaction (
   A,
   B.identity,
   40.0
)
t2 = Transaction (
   A,
   C.identity,
   70.0
)
t3 = Transaction (
   B,
   C.identity,
   700.0
)
#blockchain
TPCoins = []

block0 = Block()
block0.previous_block_hash = None
Nonce = None
block0.verified_transactions.append (t0)
digest = hash (block0)
last_block_hash = digest #last_block_hash it is hash of block0
TPCoins.append (block0)

block1 = Block()
block1.previous_block_hash = last_block_hash
block1.verified_transactions.append (t1)
block1.verified_transactions.append (t2)
block1.Nonce=mine (block1, 2)
digest = hash (block1)
last_block_hash = digest
TPCoins.append (block1)

block2 = Block()
block2.previous_block_hash = last_block_hash
block2.verified_transactions.append (t3)
Nonce = mine (block2, 2)
block2.Nonce=mine (block2, 2)
digest = hash (block2)
last_block_hash = digest
TPCoins.append (block2)

dump_blockchain(TPCoins)


#########################################################################

save the file -> ctrl +O to write ->  {enter} save -> ctrl +x exit
```

Run this file

Output:

```
Number of blocks in the chain: 3
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d01010105000381d0030818902818100b1558beccb109cbf1c223ebf6e791ea734e20c03dc8bce926f3b28c9e2a2383cf4ae87b6431211299b11
7d6e143373a6682a64e0c7de6955fbd9dc8806103d4c6a738d92d7511112e944c9d6eb51730b76e2b0b6a48069b6bd90c52549832429cbf8ba7ade362d4f3b04a5d568f54d30d6e3cb57ceaf1
8e7e7b2fb2df2d8d2530203010001
-----
value: 500.0
-----
time: 2025-02-17 10:34:26.627963
-----
--------------
====================================
```

```
------------------------------------
block # 1
sender: 30819f300d06092a864886f70d01010105000381d0030818902818100b1558beccb109cbf1c223ebf6e791ea734e20c03dc8bce926f3b28c9e2a2383cf4ae87b6431211299b117d6
e143373a6682a64e0c7de6955fbd9dc8806103d4c6a738d92d7511112e944c9d6eb51730b76e2b0b6a48069b6bd90c52549832429cbf8ba7ade362d4f3b04a5d568f54d30d6e3cb57ceaf18e7
e7b2fb2df2d8d2530203010001
-----
recipient: 30819f300d06092a864886f70d01010105000381d0030818902818100bbcf8e76057d12120ce04cdd708cc9dca881bca54eaf520584faadeaa97a6b90de616b04036b465e8f1b
bb2b79c3236f59d83c818830280b108c1a114026eb93cff404b9b78d757147b83cc259e099ae9166a0afde96a06f6fdcd1c8d99aab21ded8bf89ea7eab67efdbd52370daf555eee8ceb7e56a2
ba7c37f441c6ac00aa70203010001
-----
value: 40.0
-----
time: 2025-02-17 10:34:26.627963
-----
--------------
sender: 30819f300d06092a864886f70d01010105000381d0030818902818100b1558beccb109cbf1c223ebf6e791ea734e20c03dc8bce926f3b28c9e2a2383cf4ae87b6431211299b117d6
e143373a6682a64e0c7de6955fbd9dc8806103d4c6a738d92d7511112e944c9d6eb51730b76e2b0b6a48069b6bd90c52549832429cbf8ba7ade362d4f3b04a5d568f54d30d6e3cb57ceaf18e7
e7b2fb2df2d8d2530203010001
-----
recipient: 30819f300d06092a864886f70d01010105000381d0030818902818100bcfacd1614746e02a0ab42490f4932f55c8a339b2a2c4661c74950d9a1a22c3217e56525342d57bc51f2
c5a97fe5bd0601c8e6cc30b5752c3c6c927add09703245b10ee469ae427d4929d947e51d4e1b8921cf2fb83f6d20aef8be123f9a30ac2d8eb602a8fdcf2dfb8150db16a31196af78de1b09028
022fd48e63f476916d50203010001
-----
value: 70.0
-----
time: 2025-02-17 10:34:26.627963
-----
--------------
====================================
```

```
------------------------------------
block # 2
sender: 30819f300d06092a864886f70d01010105000381d0030818902818100bbcf8e76057d12120ce04cdd708cc9dca881bca54eaf520584faadeaa97a6b90de616b04036b465e8f1bbb2
b79c3236f59d83c818830280b108c1a114026eb93cff404b9b78d757147b83cc259e099ae9166a0afde96a06f6fdcd1c8d99aab21ded8bf89ea7eab67efdbd52370daf555eee8ceb7e56a2ba7
c37f441c6ac00aa70203010001
-----
recipient: 30819f300d06092a864886f70d01010105000381d0030818902818100bcfacd1614746e02a0ab42490f4932f55c8a339b2e2c4661c74950d9a1a22c3217e56525342d57bc51f2
c5a97fe5bd0601c8e6cc30b5752c3c6c927add09703245b10ee469ae427d4929d947e51d4e1b8921cf2fb83f6d20aef8be123f9a30ac2d8eb602a8fdcf2dfb8150db16a31196af78de1b09028
022fd48e63f476916d50203010001
-----
value: 780.0
-----
time: 2025-02-17 10:34:26.627963
-----
--------------
====================================
```

INDEX

Practical: 1

Aim:- Implement and visualize different activation functions by plotting their graphs to understand their behavior and impact on neural networks.

Theory:-

1. **Sigmoid function:**

The Sigmoid function is a type of mathematical function that maps any real-valued number to a value between 0 and 1. It is commonly used in machine learning, especially in logistic regression and neural networks, to introduce non-linearity into models. The function is defined as:

$$S(x) = \frac{1}{1 + e^{-x}}$$

Where:

- S(x) is the output of the sigmoid function,
- e is the base of the natural logarithm, and
- x is the input.

This function has the property of being differentiable, and its output is always between 0 and 1, which makes it useful for probabilities and binary classification tasks.
As $x \to \infty$, $S(x) \to 1$, and as $x \to -\infty$, $S(x) \to 0$. The function is symmetric around x=0, where S (0) = 0.5.
In the code provided, we visualize both the sigmoid function and its derivative using Matplotlib**.** The sigmoid function and its derivative are computed for a range of x-values from -6 to 6.

Here's the step-by-step explanation:

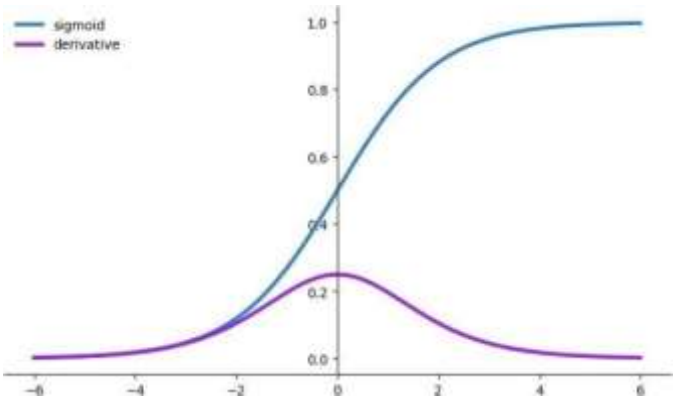
1. **Sigmoid Function Plot**: The blue curve represents the sigmoid function itself. It starts near 0 for very negative x, increases smoothly, and approaches 1 as x becomes more positive.
2. **Derivative Plot**: The purple curve represents the derivative of the sigmoid function. It shows how the rate of change of the sigmoid function behaves. As shown, the derivative has its peak near x=0and decreases as x moves away from the origin.

Code:-

```
# Sigmoid Function

import matplotlib.pyplot as plt

import numpy as np

def sigmoid(x):

    s = 1/(1+np.exp(-x))

    ds = s*(1-s)
```

```
    return s, ds


x = np.arange(-6,6,0.01)
sigmoid(x)
fig, ax = plt.subplots(figsize=(9,5))
ax.spines['left'].set_position('center')
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.plot(x,sigmoid(x)[0], color="#307EC7", linewidth=3, label="sigmoid")
ax.plot(x,sigmoid(x)[1], color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



2. TanH (Hyperbolic Tangent) Function:

The TanH (hyperbolic tangent) function is another activation function commonly used in neural networks and machine learning. It is similar to the sigmoid function, but it outputs values between -1 and 1 instead of 0 and 1. The TanH function is defined as:

$$\text{TanH}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Where:

- e is the base of the natural logarithm,
- x is the input.

The TanH function transforms input values into a range from -1 to 1, which can be useful when we need both negative and positive outputs. It is widely used in neural networks, particularly in hidden layers, because it provides a symmetric range and has a strong derivative for learning.

In the provided code, the TanH function and its derivative are visualized using Matplotlib. Here's a detailed explanation of the steps:

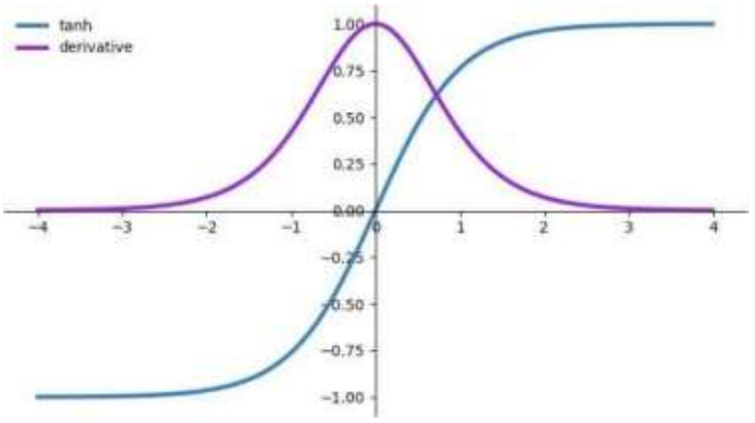
1. **TanH Function Plot**: The blue curve represents the TanH function. It is symmetric around the origin and smoothly maps input values into the range of -1 to 1. As $x \to \infty$, the output approaches 1, and as $x \to -\infty$, the output approaches -1. At x=0, the output is 0.
2. **Derivative Plot:** The purple curve represents the derivative of the TanH function. This curve shows how the rate of change of the TanH function behaves. The derivative is largest at x=0, where the slope of the TanH function is the steepest. As x moves away from 0 in either direction, the derivative becomes smaller, approaching 0 as x goes to infinity or negative infinity.

The plot is created using Matplotlib where the TanH function is plotted in blue, and the derivative is plotted in purple. The axes are adjusted to center at (0, 0), making the symmetry of the TanH function more visually apparent.

```
# TanH fucntion

import matplotlib.pyplot as plt

import numpy as np

def tanh(x):

    t = (np.exp(x)-np.exp(-x))/(np.exp(x)+np.exp(-x))

    dt = 1-t**2

    return t, dt


z = np.arange(-4,4,0.01)

tanh(z)[0].size, tanh(z)[1].size

fig, ax = plt.subplots(figsize=(9,5))

ax.spines['left'].set_position('center')

ax.spines['bottom'].set_position('center')

ax.spines['right'].set_color('none')

ax.spines['top'].set_color('none')

ax.xaxis.set_ticks_position('bottom')

ax.yaxis.set_ticks_position('left')

ax.plot(z,tanh(z)[0], color="#307EC7", linewidth=3, label="tanh")
```

```
ax.plot(z,tanh(z)[1], color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



3. ReLU (Rectified Linear Unit) Function:

The **ReLU** (Rectified Linear Unit) function is one of the most widely used activation functions in deep learning models, especially in convolutional and fully connected layers. It is preferred due to its simplicity and the fact that it helps mitigate some of the issues seen with other activation functions like **Sigmoid** or **TanH** (such as the vanishing gradient problem). The ReLU function is defined as:

$$\text{ReLU}(x) = \max(0, x)$$

Where:

- x is the input value to the function, and
- The function outputs x if x is greater than 0, otherwise, it outputs 0.

This means that for any positive input, the output is the same as the input, while for negative inputs, the output is 0. The function is non-linear and piecewise linear at x=0, making it computationally efficient.

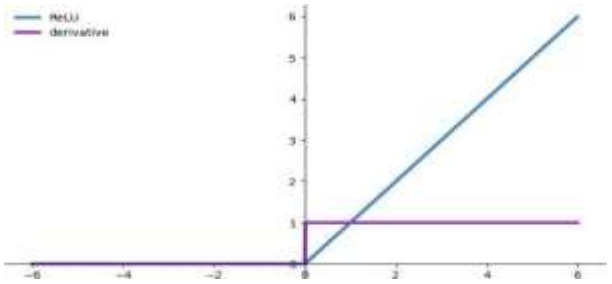
1. **ReLU Function**: The blue curve represents the ReLU activation function. It starts at 0 for negative values of x and increases linearly with a slope of 1 for positive values of x. This indicates that all negative inputs are squashed to 0, while positive inputs pass through unchanged.
2. **ReLU Derivative**: The purple curve represents the derivative of the ReLU function. For values of x greater than 0, the derivative is 1, indicating a constant slope of 1 in the positive region of the ReLU curve. For values of x less than or equal to 0, the derivative is 0, meaning that the function is flat, and no change occurs in this region.

Limitations of ReLU:

- Dying ReLU Problem**:** ReLU units can sometimes "die" during training, meaning that they stop learning entirely if they get stuck in the inactive region ($x \leq 0$), where the derivative is 0. Variants like Leaky ReLU and Parametric ReLU are introduced to address this issue.

```
# relu activation and its derivative
import matplotlib.pyplot as plt
import numpy as np
def relu(x):
    r = np.maximum(0, x)
    dr = np.where(x <= 0, 0, 1)
    return r, dr
x = np.arange(-6, 6, 0.01)
relu_values, relu_derivatives = relu(x)
fig, ax = plt.subplots(figsize=(9,5))
ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position(('data', 0))
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.plot(x, relu_values, color="#307EC7", linewidth=3, label="ReLU")
ax.plot(x,relu_derivatives, color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



4. **Leaky ReLU Activation Function:**

The Leaky ReLU (Leaky Rectified Linear Unit) function is a variation of the traditional ReLU function designed to address the problem of "dead neurons" encountered with ReLU. The problem with standard ReLU is that when the input is less than or equal to zero, the derivative is zero, and neurons can "die" and stop learning altogether. Leaky ReLU introduces a small slope for negative input values to prevent neurons from becoming inactive.
The Leaky ReLU function is defined as:

$$\text{Leaky ReLU}(x) = \max(\alpha x, x)$$

Where:

- x is the input value.
- α is a small positive constant (usually set to 0.01), which controls the slope for negative input values.
- For x≥0, the function behaves the same as ReLU, i.e., it outputs x.
- For x<0, the function outputs αx, where α is typically a small positive value.

This modification ensures that the function does not completely "saturate" for negative values, which prevents the "dying neuron" problem in neural networks.

1. **Leaky ReLU Function**: The blue curve represents the Leaky ReLU function. For values of x≥0, the function behaves just like the ReLU function, outputting the input value. However, for x<0, the function has a small slope defined by α (default is 0.01), ensuring that negative values do not cause the function to "die" (i.e., it does not output zero for negative inputs).
2. **Derivative of the Leaky ReLU Function**: The purple curve represents the derivative of the Leaky ReLU function. For x>0, the derivative is 1, indicating a constant slope. For x<0, the derivative is α\alphaα (default is 0.01), which ensures that the gradient is non-zero even for negative values of x, preventing dead neurons.

Advantages of Leaky ReLU:

- **Prevents Dying Neurons**: Unlike the standard ReLU, Leaky ReLU allows for a small gradient when x is negative, which helps avoid the "dying ReLU" problem where neurons stop learning.
- **Simplicity**: Leaky ReLU is still simple to compute and does not introduce significant computational overhead.
- **Improved Learning**: It allows the network to continue learning even when the input is negative, which can improve performance in some cases.

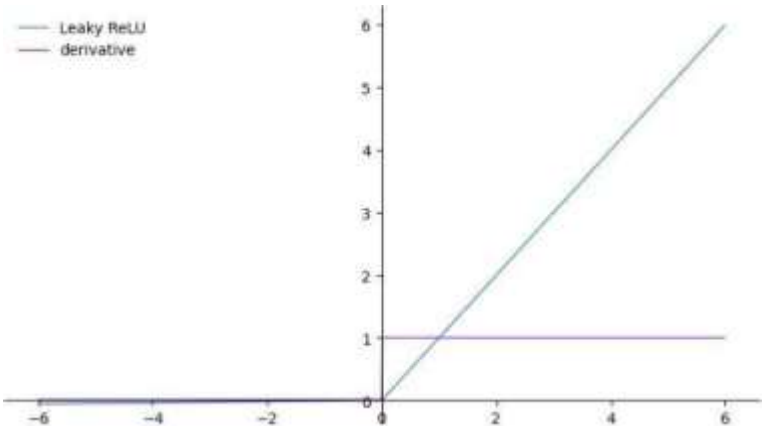
Disadvantages of Leaky ReLU:

- **Choosing the Best *α***: The choice of α\alphaα is important. If α is too large, it may lead to a function that behaves too similarly to a linear activation, potentially losing the benefits of non-linearity.
- **Still prone to some issues**: While Leaky ReLU mitigates the "dying neuron" issue, it can still suffer from other problems, such as being overly sensitive to negative inputs.

```
# Leaky ReLU

import matplotlib.pyplot as plt

import numpy as np

def leaky_relu(x, alpha=0.01):
```

```
    r = np.maximum(alpha*x, x)
    dr = np.where(x < 0, alpha, 1)
    return r, dr

x = np.arange(-6, 6, 0.01)
leaky_relu_values, leaky_relu_derivatives = leaky_relu(x)
fig, ax = plt.subplots(figsize=(9,5))
ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position(('data', 0))
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.plot(x, leaky_relu_values, color="#307EC7", linewidth=1, label="Leaky ReLU")
ax.plot(x,leaky_relu_derivatives, color="#9621E2", linewidth=1, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



5. Parametric Rectified Linear Unit (PReLU) Activation Function:

The Parametric Rectified Linear Unit (PReLU) is a variation of the Leaky ReLU function that introduces an additional learnable parameter, α, which controls the slope for negative inputs. Unlike Leaky ReLU, where α is a fixed constant, PReLU allows the model to learn the best value for α during training. This makes it more flexible and potentially more effective for different types of data.

The PReLU function is defined as:

$$\text{PReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha x, & \text{if } x < 0 \end{cases}$$

Where:

- x is the input to the function.
- α is a learnable parameter that controls the slope of the function for negative values of x.
- For x≥0, the function behaves like the standard ReLU, i.e., it outputs x.
- For x<0, the function outputs αx, with α being a trainable parameter.

The advantage of this flexibility is that it allows the network to learn the best value of α\alphaα, improving the model's performance on tasks where different features may require different levels of activation for negative inputs.

1. **PReLU Function**: The blue curve represents the PReLU activation function. For x≥0, it behaves like a standard ReLU function, where the output is the same as the input. For x<0, the output is scaled by a factor of α, meaning that the function still provides a non-zero output even for negative values.
2. **PReLU Derivative**: The purple curve represents the derivative of the PReLU function. For positive values of x, the derivative is 1, just like ReLU. For negative values of x, the derivative is α, ensuring that the network can continue learning even for negative input values.

Advantages of PReLU

- **Learnable Slope**: Unlike Leaky ReLU, where α is fixed, PReLU allows the model to learn the optimal value of α during training, improving flexibility and adaptability.
- **Prevents Dead Neurons**: Just like Leaky ReLU, PReLU avoids the "dying neuron" problem by allowing negative inputs to contribute to the learning process.
- **Increased Flexibility**: The learnable parameter α allows the model to adjust how much negative information is allowed to propagate through the network, which can lead to better performance.

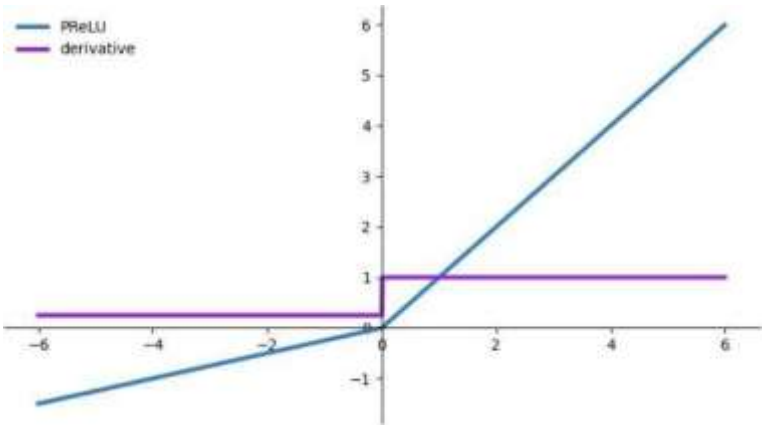
Disadvantages of PReLU

- **Risk of Overfitting**: Since α is a learnable parameter, it introduces more parameters into the model, which increases the risk of overfitting, especially in smaller datasets.
- **More Computationally Expensive**: Learning the value of α introduces additional computational complexity during training.

```
# Parametric Rectified Linear Unit (PReLU)

import matplotlib.pyplot as plt

import numpy as np

def prelu(x, alpha=0.25):

    r = np.maximum(alpha*x, x)

    dr = np.where(x < 0, alpha, 1)

    return r, dr
```

```
x = np.arange(-6, 6, 0.01)
prelu_values, prelu_derivatives = prelu(x)
fig, ax = plt.subplots(figsize=(9,5))
ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position(('data', 0))
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.plot(x, prelu_values, color="#307EC7", linewidth=3, label="PReLU")
ax.plot(x, prelu_derivatives, color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



6. ELU Activation Function:

The **Exponential Linear Unit (ELU)** is a type of activation function that is used in neural networks to introduce non-linearity. Unlike traditional activation functions like ReLU, which outputs zero for negative inputs, ELU has a smoother transition for negative values, which helps avoid issues such as "dying neurons" and allows for better optimization. The mathematical form of the ELU function is given as:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(e^x - 1) & \text{if } x < 0 \end{cases}$$

Where α is a constant that controls the smoothness of the function for negative values. By default, α=1.0.

The code defines a function elu(x, alpha=1.0) that computes both the ELU function and its derivative for a given input x.

- elu(x, alpha=1.0): This function accepts an array x and a parameter alpha to calculate the ELU output and its derivative.
- The ELU function (r) is computed using np.where(x >= 0, x, alpha*(np.exp(x) - 1)), which returns x for values of x >= 0 and alpha * (e^x - 1) for values of x < 0.
- The derivative (dr) is computed using np.where(x >= 0, 1, alpha * np.exp(x)), returning 1 for x >= 0 and alpha * e^x for x < 0.
- x = np.arange(-6, 6, 0.01): This line generates an array of x values from -6 to 6 with a step size of 0.01.

- **ELU Function**: The ELU curve (in blue) has a linear relationship for positive values of x, and for negative values of x, it follows an exponential curve, which smoothly saturates and avoids the flat zero region seen in ReLU.
- **Derivative**: The derivative curve (in purple) is 1 for positive values of x (as the ELU function is linear there), and for negative values, it follows α * exp(x) (exponential behavior). The derivative curve's behavior is significant for optimization as it provides the gradient for backpropagation.

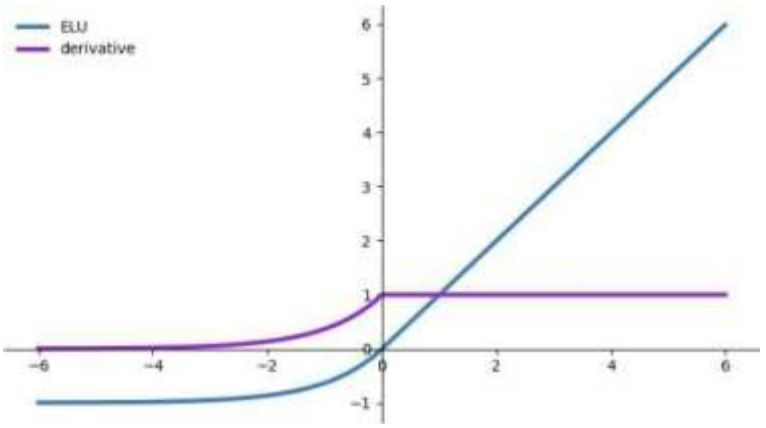
The ELU activation function is considered beneficial because:

- It reduces the likelihood of "dead neurons" (common in ReLU, where neurons output zero and stop learning).
- It allows the network to have a smoother gradient flow, leading to potentially faster convergence during training.

```
# Exponential Linear Unit (ELU)

import matplotlib.pyplot as plt

import numpy as np

def elu(x, alpha=1.0):

    r = np.where(x >= 0, x, alpha*(np.exp(x)-1))

    dr = np.where(x >= 0, 1, alpha*np.exp(x))

    return r, dr
```

```
x = np.arange(-6, 6, 0.01)
elu_values, elu_derivatives = elu(x)
fig, ax = plt.subplots(figsize=(9,5))
ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position(('data', 0))
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.plot(x, elu_values, color="#307EC7", linewidth=3, label="ELU")
ax.plot(x, elu_derivatives, color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



7. SoftPlus Activation Function:

The **SoftPlus** activation function is a smooth approximation of the ReLU (Rectified Linear Unit) function. It can be described by the equation:

$$f(x) = \log(1 + e^x)$$

The function is differentiable and smooth, unlike ReLU, which has a sharp transition at x=0. This smooth nature makes it less likely to cause issues like "dead neurons" that can occur with ReLU.

- For large positive values of x, SoftPlus behaves similarly to ReLU, since $\log(1+e^x) \approx x$ for large x.
- For large negative values of x, the SoftPlus function approaches zero, but smoothly, as opposed to the hard zero cut-off of ReLU.

- SoftPlus(x): This function takes an array x and computes both the SoftPlus function (r) and its derivative (dr).
  - The SoftPlus function is computed as np.log(1 + np.exp(x)), which applies the logarithmic function to the exponential of x and adds 1.
  - The derivative of SoftPlus is computed as 1 / (1 + np.exp(-x)), which is the sigmoid function.

- x = np.arange(-6, 6, 0.01): This creates an array of x values ranging from -6 to 6 with a step size of 0.01, providing the input for the SoftPlus function and its derivative.

- **SoftPlus Function:** The SoftPlus curve (in blue) starts from approximately zero for negative values of x and smoothly increases for positive x. As x grows larger, the curve approaches a straight line with a slope of 1 (similar to the ReLU function).
- **Derivative (Sigmoid):** The derivative (in purple) follows the sigmoid curve, with values between 0 and 1. It approaches 0 for large negative x and approaches 1 for large positive x. This derivative describes the rate of change of the SoftPlus function at each point.

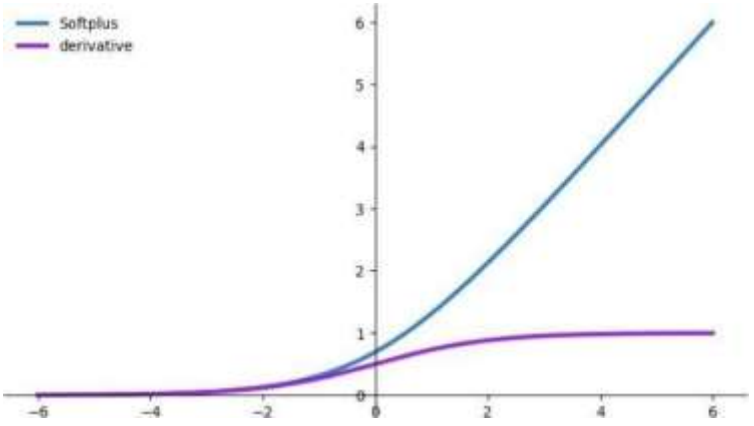
The SoftPlus function is commonly used as an activation function in neural networks because:

- It is a smooth, continuous function, which makes it differentiable everywhere, preventing problems with optimization, such as issues that arise with non-differentiable functions like ReLU at x=0.
- The derivative of SoftPlus is the sigmoid function, which can help stabilize the gradients during training by avoiding extremely small or large gradient values.
- For large positive values of x, it behaves similarly to ReLU, providing a non-zero output, while for negative values of x, the output is closer to zero, maintaining a smooth gradient.

```
# SoftPlus

import matplotlib.pyplot as plt

import numpy as np

def softplus(x):

    r = np.log(1+np.exp(x))

    dr = 1/(1+np.exp(-x))

    return r, dr


x = np.arange(-6, 6, 0.01)
```

```
softplus_values, softplus_derivatives = softplus(x)
fig, ax = plt.subplots(figsize=(9,5))
ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position(('data', 0))
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
ax.xaxis.set_ticks_position('bottom')
ax.yaxis.set_ticks_position('left')
ax.plot(x, softplus_values, color="#307EC7", linewidth=3, label="Softplus")
ax.plot(x, softplus_derivatives, color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-



8. ArcTan (Arctangent) Activation Function:

The **ArcTan (arctangent)** function is the inverse of the tangent function. It maps real values to an output in the range of $-\pi/2 \leq \text{ArcTan}(x) \leq \pi/2$. The function is defined mathematically as:

$$f(x) = \arctan(x)$$

It is used as an activation function in some neural networks due to its smooth and continuous nature. The ArcTan function:

- Has an output that is bounded, with asymptotes at $\pm\pi/2$ as x approaches infinity.

- The output smoothly transitions from negative to positive values, which is desirable for many optimization tasks in neural networks.

The code defines a function arctan(x) that computes both the ArcTan function and its derivative.

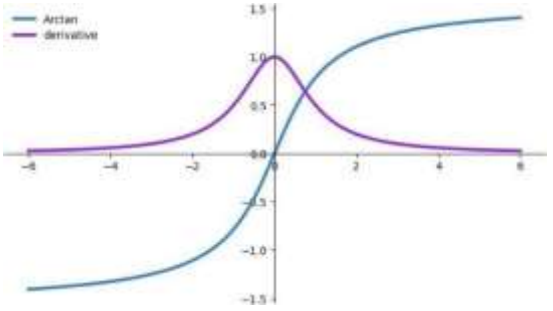
- **arctan(x)**: This function takes an array x and computes both the ArcTan function (r) and its derivative (dr):
- The ArcTan function is computed using np.arctan(x), which calculates the arctangent of each element in the input array x.
- The derivative of ArcTan is computed using the formula d/dx arctan(x)=1/1+x^2 which is implemented as 1 / (1 + x**2).
- **x = np.arange(-6, 6, 0.01)**: This creates an array of x values ranging from -6 to 6, with a step size of 0.01. This array serves as the input to the ArcTan function and its derivative.

- **ArcTan Function**: The ArcTan curve (in blue) smoothly transitions from negative to positive values. As x approaches large positive or negative values, the function asymptotically approaches $\pm\pi/2$. For x=0, the ArcTan function crosses through 0, with a smooth curve across the entire range.
- **Derivative**: The derivative (in purple) is given by 1/ {1 + x^2} . It is a smooth curve that starts at 1 when x=0, and as x becomes large (positive or negative), the derivative approaches 0. The slope of the ArcTan function is steepest around x=0 and becomes flatter as |x| increases.

```
# ArcTan

import matplotlib.pyplot as plt

import numpy as np

def arctan(x):

    r = np.arctan(x)

    dr = 1/(1+x**2)

    return r, dr


x = np.arange(-6, 6, 0.01)

arctan_values, arctan_derivatives = arctan(x)

fig, ax = plt.subplots(figsize=(9,5))

ax.spines['left'].set_position('center')

ax.spines['bottom'].set_position(('data', 0))

ax.spines['right'].set_color('none')

ax.spines['top'].set_color('none')

ax.xaxis.set_ticks_position('bottom')
```

```
ax.yaxis.set_ticks_position('left')
ax.plot(x, arctan_values, color="#307EC7", linewidth=3, label="Arctan")
ax.plot(x, arctan_derivatives, color="#9621E2", linewidth=3, label="derivative")
ax.legend(loc="upper left", frameon=False)
plt.show()
```

Output:-

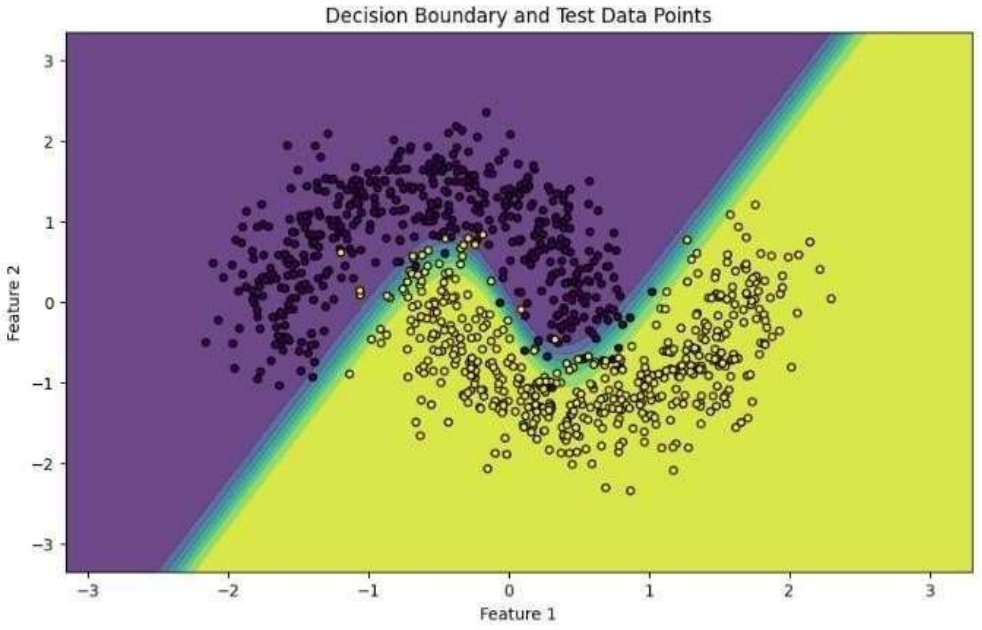



```
import tensorflow as tf
  from tensorflow.keras.models import Sequential
  from tensorflow.keras.layers import Dense
  from tensorflow.keras.optimizers import Adam
  from sklearn.datasets import make_moons
  from sklearn.model_selection import train_test_split
  from sklearn.preprocessing import StandardScaler
  import numpy as np
  import matplotlib.pyplot as plt

  X, y =make_moons(n_samples=1000, noise=0.2, random_state=42)
  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
  scalar = StandardScaler()
  X_train = scalar.fit_transform(X_train)
  X_test = scalar.transform(X_test)

model = Sequential([
  Dense(10, input_shape=(2,), activation='tanh'),
  Dense(10, activation='elu'),
```

```
  Dense(1, activation='sigmoid')
  ])
  2024-25
  model.compile(optimizer=Adam(learning_rate=0.01), loss='binary_crossentropy',
metrics=['accuracy'])
  history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50,
verbose=1)
  def plot_decision_boundary(model, X, y):
        x_min, x_max = X[:, 0].min()- 1, X[:, 0].max() + 1
        y_min, y_max = X[:, 1].min()- 1, X[:, 1].max() + 1
        xx, yy = np.meshgrid(np.linspace(x_min, x_max, 100), np.linspace(y_min, y_max,
100))
        Z =model.predict(np.c_[xx.ravel(), yy.ravel()])
        Z =Z.reshape(xx.shape)
        plt.contourf(xx, yy, Z, alpha=0.8)
        plt.scatter(X[:, 0], X[:, 1], c=y, s=20, edgecolor='k')
        plt.xlabel('Feature 1')
        plt.ylabel('Feature 2')
        plt.title('Decision Boundary and Test Data Points')
  plt.figure(figsize=(10, 6))
  plot_decision_boundary(model, np.vstack((X_train, X_test)), np.hstack((y_train, y_test)))
  plt.show()
  plt.plot(history.history['accuracy'], label='Training Accuracy')
  plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
  plt.title('Model Training Performance')
  plt.ylabel('Accuracy')
  plt.xlabel('Epoch')
  plt.legend()
  plt.show()
```

Output:-

```
Epoch 1/50
C:\Users\DELL\anaconda3\envs\python39\lib\site-packages\keras\src\layers\core\dense.py:87: UserWarning: Do not pass a
nt to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model i
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
22/22 ━━━━━━━━━━━━━━━━━━━━ 3s 27ms/step - accuracy: 0.8032 - loss: 0.4778 - val_accuracy: 0.8733 - val_loss: 0.2804
Epoch 2/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8591 - loss: 0.3185 - val_accuracy: 0.8800 - val_loss: 0.2744
Epoch 3/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8651 - loss: 0.3043 - val_accuracy: 0.8767 - val_loss: 0.2749
Epoch 4/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8658 - loss: 0.3079 - val_accuracy: 0.8833 - val_loss: 0.2706
Epoch 5/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8624 - loss: 0.2733 - val_accuracy: 0.8733 - val_loss: 0.2718
Epoch 6/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - accuracy: 0.8753 - loss: 0.2925 - val_accuracy: 0.8800 - val_loss: 0.2694
Epoch 7/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8549 - loss: 0.3122 - val_accuracy: 0.8733 - val_loss: 0.2686
Epoch 8/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8568 - loss: 0.3088 - val_accuracy: 0.8833 - val_loss: 0.2674
Epoch 9/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8695 - loss: 0.2907 - val_accuracy: 0.8767 - val_loss: 0.2681
Epoch 10/50
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8614 - loss: 0.2992 - val_accuracy: 0.8900 - val_loss: 0.2633
```



Practical: 2

Aim:- Implement gradient-based optimization to minimize the loss function and visualize loss reduction over iterations.

Theory:-

Gradient descent optimization process to minimize a simple quadratic loss function, (x−3) ^ 2, using TensorFlow.

Gradient descent works by iteratively adjusting the value of x to minimize the given loss function. The loss function here is a simple quadratic function, (x - 3) ^ 2, which has a global minimum at x=3.

- **TensorFlow** is used for automatic differentiation and optimization.
- **Matplotlib** is used for plotting the loss reduction curve over time.
- The **loss function** is a simple quadratic function: f(x)=(x−3) ^ 2.
- The function calculates the square of the difference between x and 3. The minimum value of this function occurs when x=3, where the loss is 0.
- x is initialized as a TensorFlow **trainable variable** with an initial value of 5.0.
- **learning_rate** is set to 0.1, which determines the size of the steps we take in each iteration towards the minimum.
- steps and **loss_values** are lists to store the values of the step index and corresponding loss values for plotting.
- Gradient **Tape**: TensorFlow's **GradientTape** records the operations on the TensorFlow variables to compute the gradients (derivatives) during backpropagation.
- **tape.gradient(loss, [x])** computes the derivative of the loss function with respect to x.
- Since the loss function is f(x) = (x - 3) ^ 2, its derivative is 2(x−3). This gradient is used to update x towards the minimum.
- Gradient **Update**: The **assign_sub** method is used to update x by subtracting the gradient scaled by the **learning rate** (0.1). This is a typical gradient descent update rule:

$$x = x - \eta \cdot \frac{d}{dx}\text{Loss}(x)$$

where η is the learning rate, and d/dx Loss(x) is the gradient.

The updated value of x is printed at each step along with the corresponding loss.

- After the optimization loop, a plot is generated that shows how the loss decreases over the 20 steps. The x-axis represents the **steps** (iterations), and the y-axis shows the **loss values** at each step.
- The plot demonstrates how the optimization process gradually reduces the loss value over time, converging towards the minimum at x=3.

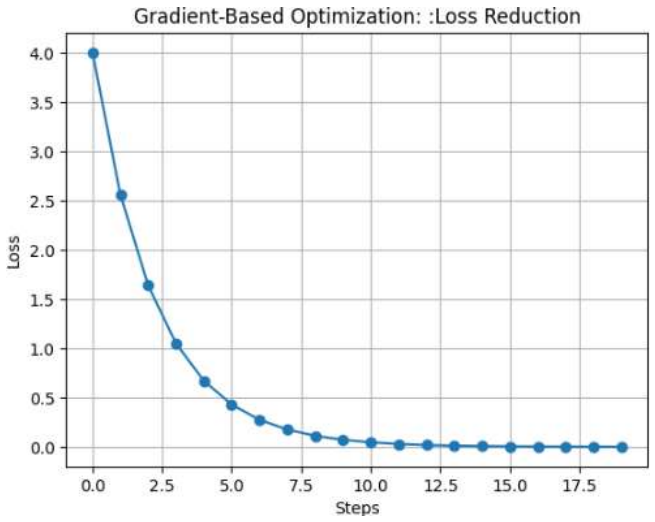
Code:-

```
import tensorflow as tf

import matplotlib.pyplot as plt

def loss_function(x):

    return (x-3)**2
```

```
x = tf.Variable(initial_value = 5.0, trainable = True, dtype = tf.float32)
learning_rate = 0.1
steps = []
loss_values = []
for step in range(20):
    with tf.GradientTape() as tape:
        loss = loss_function(x)
    gradients = tape.gradient(loss, [x])
    x.assign_sub(learning_rate*gradients[0])
    steps.append(step)
    loss_values.append(loss.numpy())
    print(f"Step {step+1}: x = {x.numpy():.4f}, Loss = {loss.numpy():.4f}")

plt.plot(steps, loss_values, marker='o')
plt.title('Gradient-Based Optimization: :Loss Reduction')
plt.xlabel('Steps')
plt.ylabel('Loss')
plt.grid(True)
plt.show()
```

Output:-

```
Step 1: x = 4.6000, Loss = 4.0000
Step 2: x = 4.2800, Loss = 2.5600
Step 3: x = 4.0240, Loss = 1.6384
Step 4: x = 3.8192, Loss = 1.0486
Step 5: x = 3.6554, Loss = 0.6711
Step 6: x = 3.5243, Loss = 0.4295
Step 7: x = 3.4194, Loss = 0.2749
Step 8: x = 3.3355, Loss = 0.1759
Step 9: x = 3.2684, Loss = 0.1126
Step 10: x = 3.2147, Loss = 0.0721
Step 11: x = 3.1718, Loss = 0.0461
Step 12: x = 3.1374, Loss = 0.0295
Step 13: x = 3.1100, Loss = 0.0189
Step 14: x = 3.0880, Loss = 0.0121
Step 15: x = 3.0704, Loss = 0.0077
Step 16: x = 3.0563, Loss = 0.0050
Step 17: x = 3.0450, Loss = 0.0032
Step 18: x = 3.0360, Loss = 0.0020
Step 19: x = 3.0288, Loss = 0.0013
Step 20: x = 3.0231, Loss = 0.0008
```



The code demonstrates the use of the **Adam optimizer** in TensorFlow for minimizing the same quadratic loss function (x - 3) ^ 2 through optimization. The Adam optimizer is an advanced gradient descent technique that adjusts the learning rate for each parameter dynamically.

The Adam (short for Adaptive Moment Estimation) optimizer combines ideas from two other popular methods:

- **Momentum:** Uses a moving average of past gradients to smooth updates.
- **RMSprop**: Uses a moving average of the squared gradients to scale the learning rate

Adam computes adaptive learning rates for each parameter based on both first-order (mean) and second-order (variance) moments. It is generally faster and more efficient compared to traditional gradient descent and is particularly well-suited for training deep learning models.

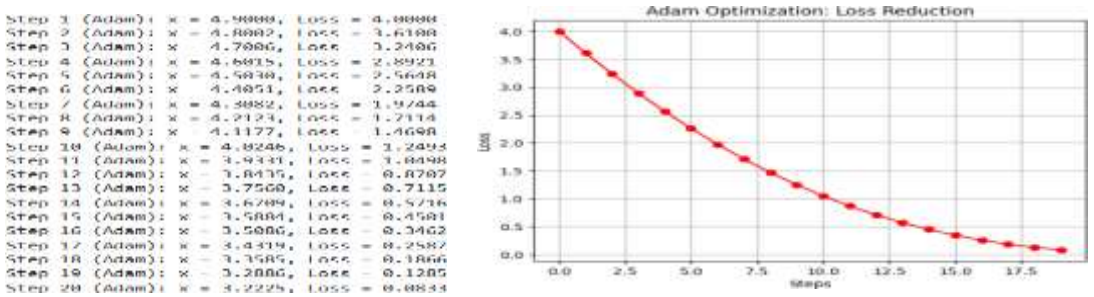
- x: This is a TensorFlow variable initialized at x=5.0. This variable will be optimized to minimize the loss function.
- optimizer = **tf.keras.optimizers.Adam(learning_rate=0.1)**: This creates an **Adam optimizer** with a learning rate of 0.1. The Adam optimizer will be responsible for applying gradients to the variable x during optimization.
- tf**.GradientTape()**: TensorFlow's GradientTape is used to record operations on the x variable, which will allow us to compute gradients later. It watches the operations that happen inside the with block.
- loss = loss_function(x): The loss function (x - 3) ^ 2is computed based on the current value of x.

- gradients = tape.gradient(loss, [x]): The gradient of the loss function with respect to x is computed. The gradient of (x - 3) ^ 2 is 2(x−3), which tells us how to adjust x to reduce the loss.
- optimizer.apply_gradients(zip(gradients, [x])): The Adam optimizer is used to update x by applying the gradients. The optimizer uses both the first-order moment (mean) and second-order moment (variance) to adjust x more effectively than traditional gradient descent.

- Storing Loss and Step: The current step number and the corresponding loss value are stored in adam_steps and adam_loss_values for later plotting.
- print(): After each step, the current value of x and the loss are printed, showing how the optimization progresses.

```
# Using Tensorflow Optimizers

x = tf.Variable(initial_value = 5.0, trainable = True, dtype = tf.float32)

optimizer = tf.keras.optimizers.Adam(learning_rate=0.1)

adam_steps = []

adam_loss_values = []

for step in range(20):
    with tf.GradientTape() as tape:
        loss = loss_function(x)
    gradients = tape.gradient(loss, [x])
    optimizer.apply_gradients(zip(gradients, [x]))
    adam_steps.append(step)
    adam_loss_values.append(loss.numpy())
    print(f"Step {step+1} (Adam): x = {x.numpy():.4f}, Loss = {loss.numpy():.4f}")


plt.plot(adam_steps, adam_loss_values, marker='o', color='red')

plt.title('Adam Optimization: Loss Reduction')

plt.xlabel('Steps')

plt.ylabel('Loss')

plt.grid('True')

plt.show()
```

Output:-



Practical: 3

Aim:- Implement a CNN for MNIST digit classification and analyze its performance using accuracy and loss visualization.

Theory:-

Applying convolutional neural network (CNN) on the MNIST dataset for handwritten digit classification using TensorFlow. It also performs some data augmentation and visualizes the model's performance during training.

- **MNIST Dataset**: This code loads the MNIST dataset using TensorFlow's tf.keras.datasets.mnist.load_data() function. The dataset contains 60,000 training images and 10,000 test images of handwritten digits (0-9).
- **Normalization**: The pixel values are in the range 0 to 255. The code normalizes the pixel values by dividing by 255, so the values are now between 0 and 1.
- **Adding Channel Dimension**: The MNIST images are grayscale, and their shape is initially (28, 28). To work with convolutional layers, we add a new axis at the end to represent the channel (which is 1 for grayscale images). This results in shapes (28, 28, 1).
- **Printing Shape**: This line prints the shape of the train and test datasets. After adding the channel dimension, each image has the shape (28, 28, 1).
- **Creating TensorFlow Datasets**:
    - tf.data.Dataset.from_tensor_slices() is used to convert the mnist_train and mnist_test data into tf.data.Dataset objects. This provides an efficient way to handle data in batches.
    - **Shuffling**: The training dataset is shuffled with a buffer size of 10,000. Shuffling helps prevent the model from learning the order of the data and improves generalization.
    - **Batching**: Both the training and test datasets are batched into chunks of 32 samples (batch_size=32), which is a standard size for training in deep learning.
- **Data Augmentation**: The augment() function is defined to perform random data augmentation on the images during training:
    - **Random Horizontal Flip**: The image is randomly flipped horizontally, which helps the model learn more robust features.
    - **Random Brightness**: Randomly adjusts the brightness of the image by a small value (up to 0.1).
    - These transformations are applied only to the training data, and they help increase the diversity of the training dataset and improve the model's ability to generalize.

- **Augmented Dataset**: The training dataset is augmented using map(augment), meaning each image in the training set will undergo the transformations defined in augment()
- **Sequential Model**: A Sequential model is created where layers are stacked one after another. The model consists of:
    - **Conv2D Layer**: The first convolutional layer applies 32 filters of size 3×33 \times 33×3 to the input image, followed by ReLU activation.
    - **MaxPooling2D Layer**: The first pooling layer reduces the spatial dimensions by applying a 2×22 \times 22×2 pooling window.
    - **Second Conv2D Layer**: The second convolutional layer applies 64 filters of size 3×33 \times 33×3, again followed by ReLU activation.
    - **Second MaxPooling2D Layer**: Another pooling layer reduces the spatial dimensions.

    - **Flatten Layer**: The output of the convolutional layers is flattened into a 1D vector to be passed to fully connected layers.
    - **Dense Layer**: A fully connected layer with 128 neurons, using ReLU activation.
    - **Final Dense Layer**: The output layer has 10 neurons (one for each digit from 0 to 9), using the softmax activation function to produce a probability distribution over the classes.

- **Optimizer**: The Adam optimizer is used for training. Adam adapts the learning rate dynamically and is widely used in deep learning for faster convergence.
- **Loss Function**: The loss function used is sparse_categorical_crossentropy, which is appropriate for multi-class classification where the target labels are integers (0-9 in this case).
- **Metrics**: Accuracy is tracked as a performance metric.
- **Training**: The model is trained for 10 epochs using the training dataset. The validation dataset is passed during training so that the model's performance on unseen data can be evaluated after each epoch.
- **Training Curves**: After training, the accuracy and loss for both the training and validation datasets are plotted for each epoch.
    - **Left Plot (Accuracy)**: Shows how the model's accuracy on both the training and validation sets evolves over time.
    - **Right Plot (Loss)**: Shows the model's loss for both datasets during training.

Code:-

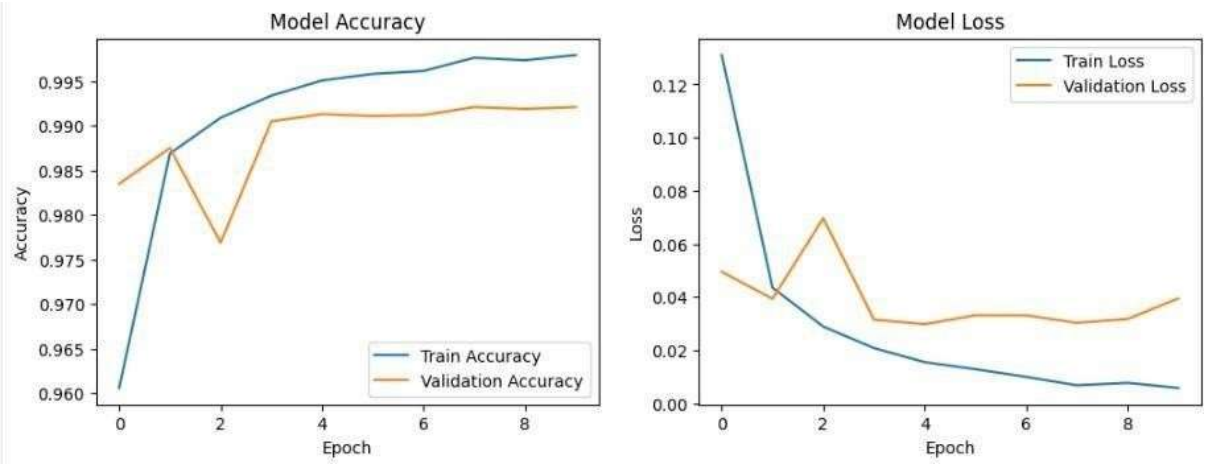
```
import tensorflow as tf

from tensorflow.keras import layers, models

import matplotlib.pyplot as plt

(mnist_train, mnist_train_labels), (mnist_test, mnist_test_labels) =
tf.keras.datasets.mnist.load_data()

mnist_train = mnist_train/255.0

mnist_test = mnist_test/255.0

mnist_train = mnist_train[..., tf.newaxis]

mnist_test = mnist_test[..., tf.newaxis]

print(f"MNIST Train Shape: {mnist_train.shape}, MNIST Test Shape: {mnist_test.shape}")

batch_size = 32

train_dataset = tf.data.Dataset.from_tensor_slices((mnist_train, mnist_train_labels))

train_dataset = train_dataset.shuffle(buffer_size=10000).batch(batch_size)

test_dataset = tf.data.Dataset.from_tensor_slices((mnist_test, mnist_test_labels))

test_dataset = test_dataset.batch(batch_size)
```

```
def augment(image, label):
    image = tf.image.random_flip_left_right(image)
    image = tf.image.random_brightness(image, max_delta=0.1)
    return image, label

augmented_train_dataset = train_dataset.map(augment)
model = models.Sequential([
    layers.Input(shape=(28,28,1)),
    layers.Conv2D(32, kernel_size=(3,3), activation='relu'),
    layers.MaxPooling2D(pool_size=(2,2)),
    layers.Conv2D(64, kernel_size=(3,3), activation='relu'),
    layers.MaxPooling2D(pool_size=(2,2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(train_dataset, epochs=10, validation_data=test_dataset)
plt.figure(figsize=(12,4))
plt.subplot(1,2,1)
plt.plot(history.history['accuracy'], label=['Train Accuracy'])
plt.plot(history.history['val_accuracy'], label=['Validation Accuracy'])
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
```

```
plt.legend()
plt.subplot(1,2,2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

Output:-

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 10s 1us/step
MNIST Train Shape: (60000, 28, 28, 1), MNIST Test Shape: (10000, 28, 28, 1)
Epoch 1/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 49s 24ms/step - accuracy: 0.9086 - loss: 0.3016 - val_accuracy: 0.9835 - val_loss: 0.0496
Epoch 2/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 68s 36ms/step - accuracy: 0.9859 - loss: 0.0465 - val_accuracy: 0.9875 - val_loss: 0.0394
Epoch 3/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 61s 32ms/step - accuracy: 0.9898 - loss: 0.0315 - val_accuracy: 0.9769 - val_loss: 0.0697
Epoch 4/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 62s 33ms/step - accuracy: 0.9933 - loss: 0.0218 - val_accuracy: 0.9905 - val_loss: 0.0316
Epoch 5/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 66s 35ms/step - accuracy: 0.9954 - loss: 0.0157 - val_accuracy: 0.9913 - val_loss: 0.0299
Epoch 6/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 56s 30ms/step - accuracy: 0.9957 - loss: 0.0129 - val_accuracy: 0.9911 - val_loss: 0.0332
Epoch 7/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 52s 28ms/step - accuracy: 0.9963 - loss: 0.0094 - val_accuracy: 0.9912 - val_loss: 0.0332
Epoch 8/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 63s 34ms/step - accuracy: 0.9979 - loss: 0.0064 - val_accuracy: 0.9921 - val_loss: 0.0304
Epoch 9/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 56s 30ms/step - accuracy: 0.9973 - loss: 0.0072 - val_accuracy: 0.9919 - val_loss: 0.0318
Epoch 10/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 48s 26ms/step - accuracy: 0.9979 - loss: 0.0058 - val_accuracy: 0.9921 - val_loss: 0.0395
```



Practical: 4

Aim:- Perform regularization to prevent model from over-fitting.

Theory:-

Regularization is a technique used in machine learning to prevent overfitting by adding some form of penalty to the loss function. The goal is to reduce the complexity of the model, ensuring that it generalizes well to new, unseen data rather than fitting too closely to the training data. Regularization helps combat overfitting, which occurs when a model learns to perform very well on the training data but fails to generalize to unseen data (test data). Regularization techniques like **L2 regularization (Ridge regression)** penalize the size of the weights, which prevents the model from becoming too complex and overfitting the training data.

In this case, **L2 regularization** is applied to the first two Dense layers of the model:

- The regularization term added to the loss function is proportional to the square of the weights. By penalizing large weights, the model is encouraged to learn simpler, more generalized patterns rather than memorizing the training data.
- **Impact on Training and Validation Loss**: With regularization, you might observe that the gap between the training loss and validation loss is smaller, and the model is less likely to overfit, especially if you train it for many epochs.

- **Loading the MNIST dataset**: The MNIST dataset contains images of handwritten digits. The data is split into training (60,000 images) and test (10,000 images) sets.
- **Normalization**: The pixel values range from 0 to 255. By dividing by 255.0, the pixel values are normalized to the range [0, 1], which is a common preprocessing step in neural networks to improve convergence.
- **Reshaping**: The images, originally 28x28 pixels, are flattened into 784-dimensional vectors (28 x 28 = 784). This flattening is required for fully connected (Dense) layers in a feed-forward neural network.
- **One-Hot Encoding**: The MNIST labels are integers (0-9), but neural networks require categorical labels as a one-hot encoded vector (i.e., a vector where only the index corresponding to the class is 1, and all others are 0). For example, the label 2 is converted to [0, 0, 1, 0, 0, 0, 0, 0, 0, 0].
- **Model Architecture**: A simple feed-forward neural network is built using the Sequential model:
    - **First Dense Layer**: A fully connected layer with 128 units and ReLU activation. The input_shape=(784,) indicates the input dimension (784 features from the flattened MNIST image). The kernel_regularizer=tf.keras.regularizers.l2(0.01) applies L2 regularization with a weight decay factor of 0.01. This penalizes large weights during training and helps prevent overfitting by adding a term proportional to the sum of the squared weights to the loss function.
    - **Second Dense Layer**: Similar to the first layer, but with 64 units and the same L2 regularization.
    - **Output Layer**: A softmax output layer with 10 units (one for each digit class).
- **Optimizer:** The Adam optimizer is used with a learning rate of 0.001. Adam is an adaptive optimization algorithm that adjusts the learning rate during training based on the gradient information.

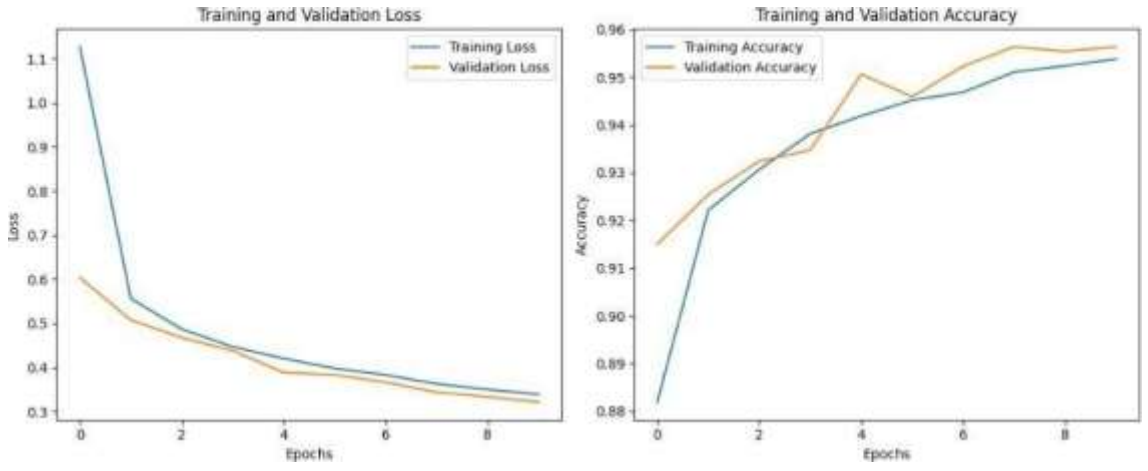
Code:-

```
import tensorflow as tf

# Load MNIST data

(train_data, train_labels), (test_data, test_labels) = tf.keras.datasets.mnist.load_data()

# Normalize and flatten the images

train_data = train_data.reshape((60000, 784)) / 255.0

test_data = test_data.reshape((10000, 784)) / 255.0

# One-hot encode labels

train_labels = tf.keras.utils.to_categorical(train_labels, 10)

test_labels = tf.keras.utils.to_categorical(test_labels, 10)

# Build the model

model = tf.keras.models.Sequential([

    tf.keras.layers.Dense(128, activation='relu', input_shape=(784,),
kernel_regularizer=tf.keras.regularizers.l2(0.01)),

    tf.keras.layers.Dense(64, activation='relu',
kernel_regularizer=tf.keras.regularizers.l2(0.01)),

    tf.keras.layers.Dense(10, activation='softmax')

])

# Compile the model

model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),

                loss='categorical_crossentropy',

                metrics=['accuracy'])

# Train the model

history = model.fit(train_data, train_labels, epochs=10, batch_size=128,
validation_data=(test_data, test_labels))

import matplotlib.pyplot as plt

plt.figure(figsize=(12,5))

plt.subplot(1,2,1)

plt.plot(history.history['loss'], label='Training Loss')

plt.plot(history.history['val_loss'], label='Validation Loss')

plt.title('Training and Validation Loss')
```

```
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.subplot(1,2,2)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.tight_layout()
plt.show()
```

Output:-

```
Epoch 1/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 6s 8ms/step - accuracy: 0.8095 - loss: 1.9077 - val_accuracy: 0.9150 - val_loss: 0.6038
Epoch 2/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9175 - loss: 0.5843 - val_accuracy: 0.9254 - val_loss: 0.5074
Epoch 3/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9286 - loss: 0.4989 - val_accuracy: 0.9324 - val_loss: 0.4672
Epoch 4/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9365 - loss: 0.4567 - val_accuracy: 0.9347 - val_loss: 0.4382
Epoch 5/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9407 - loss: 0.4263 - val_accuracy: 0.9506 - val_loss: 0.3883
Epoch 6/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9434 - loss: 0.4043 - val_accuracy: 0.9459 - val_loss: 0.3830
Epoch 7/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 4s 7ms/step - accuracy: 0.9460 - loss: 0.3887 - val_accuracy: 0.9523 - val_loss: 0.3662
Epoch 8/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 5s 7ms/step - accuracy: 0.9498 - loss: 0.3667 - val_accuracy: 0.9564 - val_loss: 0.3435
Epoch 9/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9536 - loss: 0.3477 - val_accuracy: 0.9554 - val_loss: 0.3334
Epoch 10/10
469/469 ━━━━━━━━━━━━━━━━━━━━ 3s 7ms/step - accuracy: 0.9543 - loss: 0.3384 - val_accuracy: 0.9564 - val_loss: 0.3215
```



Practical: 5

Aim:- Implement a CNN for MNIST digit classification, train and evaluate the model, and predict sample images.

Theory:-

The task at hand is to implement a Convolutional Neural Network (CNN) for recognizing handwritten digits from the **MNIST dataset**. The MNIST (Modified National Institute of Standards and Technology) dataset is a large collection of 28x28 grayscale images of handwritten digits (0 through 9). This dataset is often used as a benchmark for evaluating machine learning models, especially for image classification tasks.

In this case, we are using **Convolutional Neural Networks (CNNs)**, which are a class of deep neural networks that are particularly effective for processing images due to their ability to detect patterns such as edges, textures, and shapes through convolution operations.

- **Loading the MNIST Dataset:**
  - The MNIST dataset is loaded using keras.datasets.mnist.load_data(). This function returns training and test data as tuples of image arrays and corresponding labels.
  - x_train and x_test contain the images of the digits (28x28 pixel grayscale images).
  - y_train and y_test contain the corresponding labels (the actual digit in each image).

- **Preprocessing the Data:**
  - The pixel values of the images are normalized to the range [0, 1] by dividing by 255. This is common practice to ensure that the model learns efficiently without large scale values.
  - The images are also reshaped to have an additional channel dimension, as CNNs expect 3D input: height, width, and channels (grayscale has 1 channel). So, the shape of the images becomes (28, 28, 1) instead of just (28, 28).

- **Building the CNN Model:**
  - **Conv2D Layer**: A 2D convolution layer with 32 filters of size (3, 3) is applied. This layer is responsible for extracting features from the input image. The relu activation function introduces non-linearity to the model.
  - **MaxPooling2D Layer**: After each convolution layer, max pooling is applied with a pool size of (2, 2). This helps reduce the spatial dimensions (height and width) of the feature maps while retaining the important information.
  - **Conv2D and MaxPooling2D Layers**: A second set of convolution and pooling layers is applied with 64 filters. This further extracts high-level features from the image.
  - **Flatten Layer**: After feature extraction, the 2D feature maps are flattened into a 1D vector to feed into the fully connected layers.
  - **Dense Layers**: These are fully connected layers. The first dense layer has 64 neurons with relu activation, and the second dense layer has 10 neurons with softmax activation. The 10 output neurons correspond to the 10 digit classes (0-9), and the softmax activation ensures the output is a probability distribution over the classes.

- **Compiling the Model:**

- The model is compiled with the **Adam optimizer**, which is an adaptive learning rate optimization algorithm.
- The loss function used is **sparse categorical crossentropy**, which is appropriate for multi-class classification problems where the target labels are integers (as in the case of the MNIST labels).
- We track accuracy as a performance metric.

➢ **Training the Model:**
- The model is trained using the training data (x_train and y_train) for **15 epochs** with a batch size of 128.
- The validation data (x_test and y_test) is used to evaluate the model's performance after each epoch.
- The model's training history, including loss and accuracy over epochs, is saved in history.

➢ **Evaluating the Model:**
- After training, the model is evaluated on the test data (x_test and y_test) to check its accuracy on unseen data.
- The accuracy is printed out.

➢ **Making Predictions on a Sample Image:**
- A sample image from the test set is chosen (x_test[0]), and its true label is obtained (y_test[0]).
- The model makes a prediction on this sample image, and the predicted label is extracted by taking the index of the highest probability (using np.argmax).

➢ **Visualizing the Sample Image:**
- The sample image is displayed using Matplotlib. The. squeeze() method is used to remove the extra dimensions added earlier (such as the batch dimension).

Key Concepts in CNNs:

- **Convolution:** This is the process where the network scans the image with a filter (kernel) to extract important features such as edges and textures. The output of the convolution layer is a feature map that contains learned patterns from the image.
- **Max Pooling:** This operation reduces the spatial dimensions (height and width) of the feature map, reducing the number of parameters and computations while retaining the most important information.
- **ReLU Activation:** The **Rectified Linear Unit (ReLU)** is a non-linear activation function that introduces non-linearity into the network, allowing it to learn complex patterns.
- **Softmax Activation:** The **softmax function** is used in the final layer of classification models to output probabilities for each class, ensuring the predicted values sum to 1.

Code:-

```
import tensorflow as tf
```

```
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
x_train = x_train.astype("float32")/255.0
x_test = x_test.astype("float32")/255.0
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)
model = keras.models.Sequential([
    keras.layers.Conv2D(32, (3,3), activation="relu", input_shape=(28,28,1)),
    keras.layers.MaxPooling2D((2,2)),
    keras.layers.Conv2D(64, (3,3), activation="relu"),
    keras.layers.MaxPooling2D((2,2)),
    keras.layers.Flatten(),
    keras.layers.Dense(64, activation="relu"),
    keras.layers.Dense(10, activation="softmax")
])
model.compile(optimizer="adam", loss="sparse_categorical_crossentropy",
metrics=['accuracy'])
history = model.fit(x_train, y_train, epochs=15, batch_size=128, validation_data=(x_test,
y_test))
test_loss, test_acc = model.evaluate(x_test, y_test)
print("Test accuracy:", test_acc)
sample_img = x_test[0]
sample_label = y_test[0]
sample_img = np.expand_dims(sample_img, 0)
pred = model.predict(sample_img)
pred_label = np.argmax(pred)
print("Sample image true label:", sample_label)
print("Sample image predicted label:", pred_label)
```

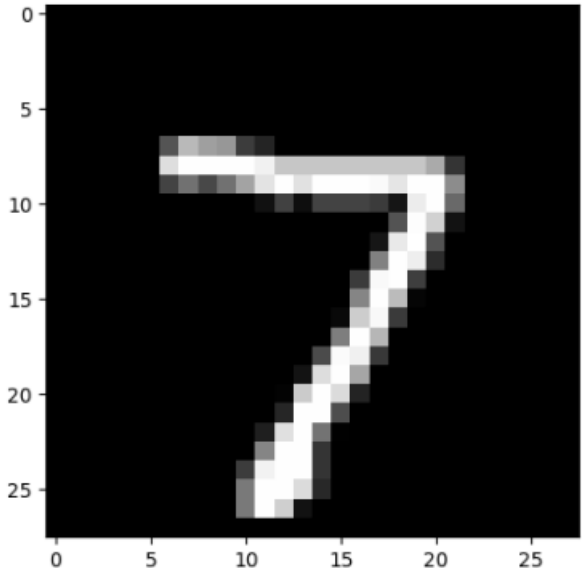
plt.imshow(sample_img.squeeze(), cmap='gray')

plt.show()

Output:-

```
Epoch 1/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 34s 68ms/step - accuracy: 0.8465 - loss: 0.5369 - val_accuracy: 0.9804 - val_loss: 0.0679
Epoch 2/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 31s 66ms/step - accuracy: 0.9811 - loss: 0.0622 - val_accuracy: 0.9859 - val_loss: 0.0448
Epoch 3/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 31s 66ms/step - accuracy: 0.9865 - loss: 0.0440 - val_accuracy: 0.9887 - val_loss: 0.0349
Epoch 4/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 31s 67ms/step - accuracy: 0.9901 - loss: 0.0334 - val_accuracy: 0.9841 - val_loss: 0.0482
Epoch 5/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 32s 68ms/step - accuracy: 0.9914 - loss: 0.0272 - val_accuracy: 0.9910 - val_loss: 0.0275
Epoch 6/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 41s 67ms/step - accuracy: 0.9927 - loss: 0.0245 - val_accuracy: 0.9909 - val_loss: 0.0293
Epoch 7/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 32s 68ms/step - accuracy: 0.9950 - loss: 0.0179 - val_accuracy: 0.9909 - val_loss: 0.0287
Epoch 8/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 36s 77ms/step - accuracy: 0.9954 - loss: 0.0148 - val_accuracy: 0.9912 - val_loss: 0.0272
Epoch 9/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 37s 78ms/step - accuracy: 0.9963 - loss: 0.0117 - val_accuracy: 0.9901 - val_loss: 0.0316
Epoch 10/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 38s 81ms/step - accuracy: 0.9971 - loss: 0.0104 - val_accuracy: 0.9912 - val_loss: 0.0298
Epoch 11/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 36s 77ms/step - accuracy: 0.9976 - loss: 0.0074 - val_accuracy: 0.9907 - val_loss: 0.0326
Epoch 12/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 38s 81ms/step - accuracy: 0.9982 - loss: 0.0060 - val_accuracy: 0.9906 - val_loss: 0.0311
Epoch 13/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 33s 71ms/step - accuracy: 0.9979 - loss: 0.0069 - val_accuracy: 0.9911 - val_loss: 0.0356
Epoch 14/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 34s 72ms/step - accuracy: 0.9986 - loss: 0.0056 - val_accuracy: 0.9921 - val_loss: 0.0326
Epoch 15/15
469/469 ━━━━━━━━━━━━━━━━━━━━ 33s 70ms/step - accuracy: 0.9987 - loss: 0.0042 - val_accuracy: 0.9902 - val_loss: 0.0382
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step - accuracy: 0.9872 - loss: 0.0471
Test accuracy: 0.9901999831199646
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 178ms/step
Sample image true label: 7
Sample image predicted label: 7
```



Practical: 6

Aim:- Implement time series forecasting using RNN and ARIMA models to predict airline passenger trends and compare their performance.

Theory:-

Implementing **Recurrent Neural Network (RNN)** for predicting airline passenger traffic based on historical data. The dataset we are using contains monthly airline passenger counts from 1949 to 1960. We will compare the performance of the RNN model with the **ARIMA** (AutoRegressive Integrated Moving Average) model, which is a traditional statistical model for time series forecasting. Time series forecasting refers to predicting future values based on previously observed values in the series. **RNNs**, particularly variants like **LSTM (Long Short-Term Memory)**, are very effective for time series data because they have the ability to learn patterns over time by maintaining memory of previous states.

- **RNNs for Time Series Forecasting**: RNNs are used for sequential data and time series problems due to their ability to "remember" past inputs. They are suitable for tasks like predicting future values based on historical data.
- **ARIMA Model**: ARIMA is a classical statistical model used for time series forecasting. It combines three components:

  - **AR (AutoRegressive)**: Uses the relationship between an observation and a number of lagged observations.
  - **I (Integrated)**: Represents the differencing of raw observations to make the time series stationary.
  - **MA (Moving Average)**: Models the relationship between an observation and a residual error from a moving average model applied to lagged observations.
  - **RMSE**: A measure of the differences between predicted and actual values, giving an indication of the model's prediction accuracy.

- **Building the RNN Model:**
  - **SimpleRNN Layers**: A simple RNN model is built using two RNN layers, each with 50 units. The return_sequences=True argument in the first RNN layer ensures that the output of each time step is passed on to the next layer. The second RNN layer doesn't return sequences (return_sequences=False), which means it will output the final state after processing all time steps.
  - **Dense Layers**: A dense layer with 25 neurons and another dense output layer with a single neuron that predicts the future value (passenger count for the next month).
  - **Compilation**: The model is compiled using the **Adam optimizer** and **mean squared error** as the loss function, which is appropriate for regression tasks.

- **Making Predictions and Inverse Transforming the Data:**
  - After training, predictions are made on both the training and test datasets.
  - The predictions are then **inverse transformed** to the original scale of the passenger data using scaler.inverse_transform. This step is necessary since the data was scaled earlier.

- **Calculating RMSE (Root Mean Squared Error):**

- **RMSE** is calculated for both the training and test predictions. RMSE is a common metric to evaluate regression models, as it represents the square root of the average squared differences between predicted and actual values. A lower RMSE indicates better performance.

➢ **Comparing ARIMA Model:**

- An **ARIMA model** is fit to the original dataset (df) with parameters (5,1,0). These parameters represent the order of the AR, I, and MA components, respectively.
- Predictions are made for the same length as the test set and RMSE is calculated to evaluate the ARIMA model's performance.

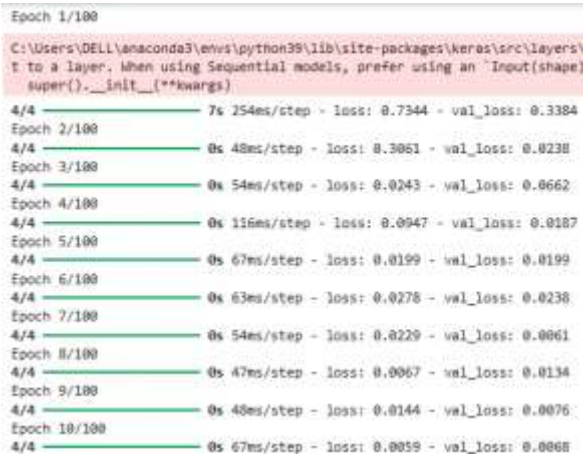
Code:-

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, SimpleRNN, LSTM
from sklearn.model_selection import train_test_split
from statsmodels.tsa.arima.model import ARIMA
import seaborn as sns
url= 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/airline-passengers.csv'
df = pd.read_csv(url,usecols=['Passengers'])
df.head()
scaler = MinMaxScaler(feature_range=(0,1))
scaled_data = scaler.fit_transform(df)


def create_dataset(data, time_step=1):
    X, Y = [], []
    for i in range(len(data)-time_step-1):
        a = data[i:(i+time_step),0]
        X.append(a)
        Y.append(data[i+time_step,0])
    return np.array(X), np.array(Y)
```

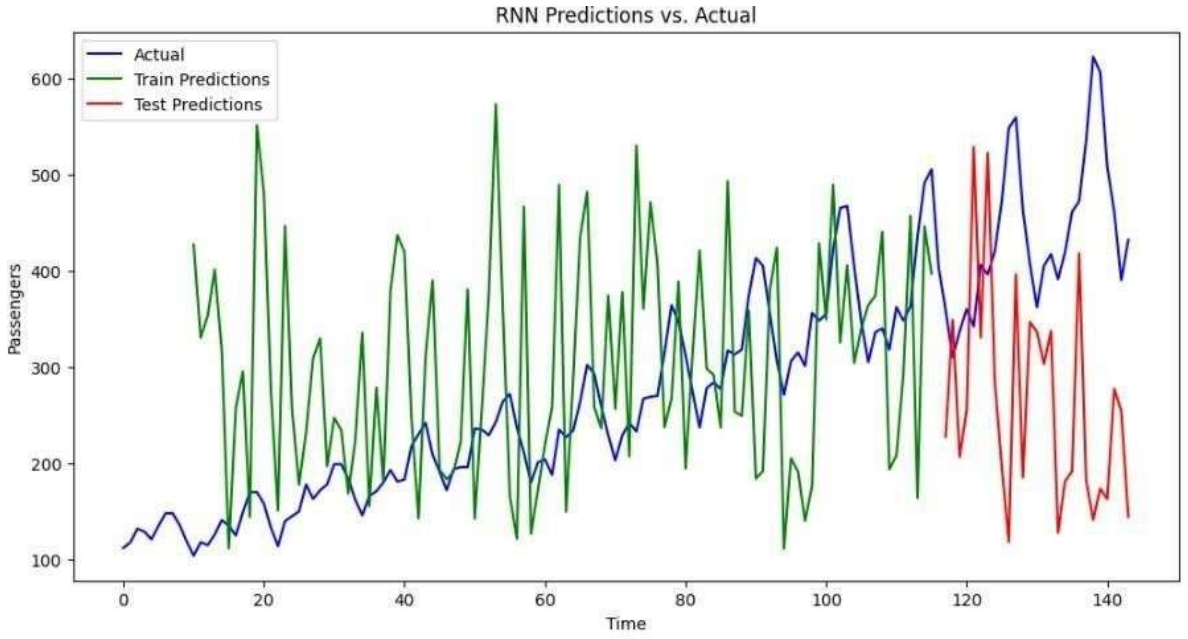
```
time_step=10
X, Y = create_dataset(scaled_data, time_step)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
X_train = X_train.reshape(X_train.shape[0], X_train.shape[1],1)
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1],1)

model = Sequential()
model.add(SimpleRNN(50, return_sequences=True, input_shape=(time_step,1)))
model.add(SimpleRNN(50, return_sequences=False))
model.add(Dense(25))
model.add(Dense(1))
model.compile(optimizer='adam',  loss='mean_squared_error')
model.fit(X_train, Y_train, epochs=100, batch_size=32, validation_data=(X_test, Y_test),
verbose=1)
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)
train_predict = scaler.inverse_transform(train_predict)
test_predict = scaler.inverse_transform(test_predict)
Y_train = scaler.inverse_transform([Y_train])
Y_test = scaler.inverse_transform([Y_test])
train_rmse = np.sqrt(mean_squared_error(Y_train[0], train_predict[:,0]))
test_rmse = np.sqrt(mean_squared_error(Y_test[0], test_predict[:,0]))
print(f'Train RMSE: {train_rmse}')
print(f'Test RMSE: {test_rmse}')
model_arima = ARIMA(df, order=(5,1,0))
model_arima_fit = model_arima.fit()
arima_pred = model_arima_fit.forecast(steps=len(X_test))
arima_rmse = np.sqrt(mean_squared_error(df[-len(arima_pred):], arima_pred))
print(f'ARIMA RMSE: {arima_rmse}')
plt.figure(figsize=(12,6))
```

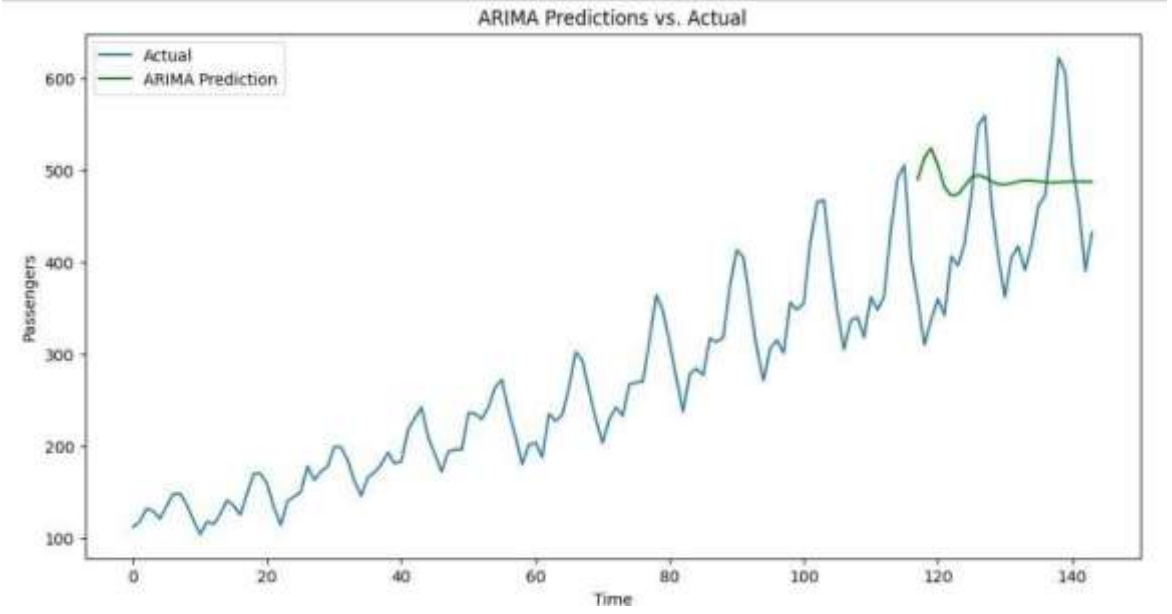
```
plt.plot(df, label='Actual', color='blue')

plt.plot(range(time_step, time_step+len(train_predict)), train_predict, color='green',
label='Train Predictions')

plt.plot(range(len(df)-len(test_predict), len(df)), test_predict, color='red', label='Test
Predictions')

plt.legend()

plt.title('RNN Predictions vs. Actual')

plt.xlabel('Time')

plt.ylabel('Passengers')

plt.show()


plt.figure(figsize=(12,6))

plt.plot(df, label='Actual')

plt.plot(range(len(df)-len(arima_pred), len(df)), arima_pred, color='green', label='ARIMA
Prediction')

plt.legend()

plt.title('ARIMA Predictions vs. Actual')

plt.xlabel('Time')

plt.ylabel('Passengers')

plt.show()
```

Output:-



```
Epoch 95/100
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step - loss: 0.0020 - val_loss: 0.0019
Epoch 96/100
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 67ms/step - loss: 0.0023 - val_loss: 0.0019
Epoch 97/100
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 91ms/step - loss: 0.0023 - val_loss: 0.0018
Epoch 98/100
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step - loss: 0.0023 - val_loss: 0.0017
Epoch 99/100
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 93ms/step - loss: 0.0021 - val_loss: 0.0017
Epoch 100/100
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 115ms/step - loss: 0.0020 - val_loss: 0.0018
4/4 ━━━━━━━━━━━━━━━━━━━━ 2s 355ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 157ms/step
Train RMSE: 23.23939814270194
Test RMSE: 21.82204568843397
ARIMA RMSE: 96.63626313278345
```





Practical: 7

Aim:- Perform stock price prediction using LSTM and GRU models, compare their performance, and forecast future prices.

Theory:-

Two types of **Recurrent Neural Networks (RNNs)** — **LSTM (Long Short-Term Memory)** and **GRU (Gated Recurrent Units)** — to predict the stock prices of **Apple Inc. (AAPL)**. These models are popular for time series forecasting because they can handle sequential data, which makes them suitable for stock price predictions, weather forecasting, etc.

1. **LSTM (Long Short-Term Memory):**
   - **LSTM** is a type of RNN designed to solve the problem of vanishing gradients, which can occur in traditional RNNs when dealing with long sequences of data.
   - LSTM units have memory cells that can store information over long periods. They utilize a combination of **gates** (input, output, and forget) that control the flow of information. This makes LSTM particularly suitable for problems where long-term dependencies exist, such as time series forecasting.
   - The gates in LSTM regulate how information is remembered and how much influence past states should have on future predictions.

2. **GRU (Gated Recurrent Units):**
   - **GRU** is a simpler variant of LSTM and also solves the vanishing gradient problem. However, it combines the input and forget gates into a single update gate and merges the cell state and hidden state.
   - GRUs are computationally more efficient than LSTMs since they use fewer parameters, while still achieving similar performance for many tasks.

Both models are used for sequential tasks, but LSTMs are generally more powerful for learning complex patterns in long sequences, whereas GRUs can be faster and more efficient in practice.

- **Building the LSTM and GRU Models:**
  - **LSTM Model:**
    - A **Sequential** model is created, and an **LSTM layer** with 50 units is added.
    - The output of the LSTM is connected to a **Dense layer** with a single neuron that outputs the predicted stock price.
    - The model is compiled using the **Adam optimizer** and **mean squared error** as the loss function, which is typical for regression problems.
  - **GRU Model:**
    - The **GRU layer** works similarly to the LSTM layer but uses fewer parameters. It is also followed by a **Dense layer** to output the predicted stock price.

Code:-

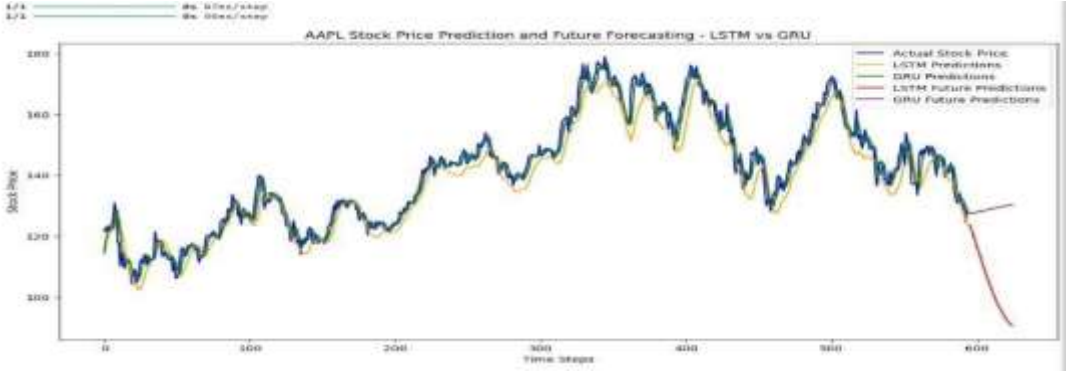
```
import yfinance as yf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, GRU, Dense
stock_symbol = 'AAPL'
df = yf.download(stock_symbol, start='2010-01-01', end='2023-01-01')
data = df[['Close']].values
scaler = MinMaxScaler(feature_range=(0,1))
data_scaled = scaler.fit_transform(data)
train_size = int(len(data_scaled)*0.8)
train, test = data_scaled[0:train_size], data_scaled[train_size:]

def create_dataset(dataset, look_back=60):
    X, Y = [], []
    for i in range(len(dataset)-look_back):
        X.append(dataset[i:i+look_back, 0])
        Y.append(dataset[i + look_back, 0])
    return np.array(X), np.array(Y)

look_back = 60
X_train, Y_train = create_dataset(train, look_back)
X_test, Y_test = create_dataset(test, look_back)
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
model_lstm = Sequential()
model_lstm.add(LSTM(50, input_shape=(look_back, 1)))
```

```
model_lstm.add(Dense(1))
model_lstm.compile(loss='mean_squared_error', optimizer='adam')
model_gru = Sequential()
model_gru.add(GRU(50, input_shape=(look_back, 1)))
model_gru.add(Dense(1))
model_gru.compile(loss='mean_squared_error', optimizer='adam')
model_lstm.fit(X_train, Y_train, epochs=20, batch_size=32, verbose=1)
model_gru.fit(X_train, Y_train, epochs=20, batch_size=32, verbose=1)
lstm_predictions = model_lstm.predict(X_test)
gru_predictions = model_gru.predict(X_test)
lstm_predictions = scaler.inverse_transform(lstm_predictions)
gru_predictions = scaler.inverse_transform(gru_predictions)
Y_test_actual = scaler.inverse_transform([Y_test])
future_steps = 30
last_sequence = X_test[-1]
lstm_future_predictions = []
for _ in range(future_steps):
    prediction = model_lstm.predict(np.reshape(last_sequence, (1, look_back, 1)))
    lstm_future_predictions.append(prediction[0][0])
    last_sequence = np.append(last_sequence[1:], prediction[0])

last_sequence = X_test[-1]
gru_future_predictions = []
for _ in range(future_steps):
    prediction = model_gru.predict(np.reshape(last_sequence, (1, look_back, 1)))
    gru_future_predictions.append(prediction[0][0])
    last_sequence = np.append(last_sequence[1:], prediction[0])

lstm_future_predictions =
scaler.inverse_transform(np.array(lstm_future_predictions).reshape(-1, 1))
```

```
gru_future_predictions = scaler.inverse_transform(np.array(gru_future_predictions).reshape(-1, 1))

future_range = np.arange(len(Y_test_actual[0]), len(Y_test_actual[0])+ future_steps)


plt.figure(figsize=(14,7))

plt.plot(Y_test_actual[0], label='Actual Stock Price', color='blue')

plt.plot(lstm_predictions, label='LSTM Predictions', color='orange')

plt.plot(gru_predictions, label='GRU Predictions', color='green')

plt.plot(future_range, lstm_future_predictions, label='LSTM Future Predictions', color='red')

plt.plot(future_range, gru_future_predictions, label='GRU Future Predictions', color='purple')

plt.title(f'{stock_symbol} Stock Price Prediction and Future Forecasting - LSTM vs GRU')

plt.xlabel('Time Steps')

plt.ylabel('Stock Price')

plt.legend()

plt.show()
```

Output:-

```
Epoch 1/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 5s 31ms/step - loss: 0.0043
Epoch 2/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 32ms/step - loss: 6.3292e-05
Epoch 3/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 31ms/step - loss: 5.0747e-05
Epoch 4/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 31ms/step - loss: 4.8310e-05
Epoch 5/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 32ms/step - loss: 5.7124e-05
Epoch 6/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 32ms/step - loss: 4.0317e-05
Epoch 7/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 32ms/step - loss: 3.7738e-05
Epoch 8/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 31ms/step - loss: 3.8731e-05
Epoch 9/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 32ms/step - loss: 3.4381e-05
Epoch 10/20
80/80 ━━━━━━━━━━━━━━━━━━━━ 3s 31ms/step - loss: 3.9180e-05
```



Practical: 8

Aim:- Perform classification of cats and dogs using a pre-trained VGG16 model with transfer learning, data augmentation, and evaluation on validation data.

Theory:-

Transfer Learning is a deep learning technique where a pre-trained model (trained on a large dataset) is reused for a new, but related, task. Instead of training a model from scratch, TL allows leveraging learned features from a powerful neural network, reducing training time and improving accuracy with limited data.

Key Concepts:

1. **Feature Extraction** – Use the pre-trained model as a fixed feature extractor by removing its top layers and adding new task-specific layers. The pre-trained weights remain unchanged.
2. **Fine-Tuning** – Unfreeze some layers of the pre-trained model and retrain them on the new dataset, adapting to specific features of the new task.
3. **Benefits:**
    - Faster convergence and reduced computational cost.
    - Requires less labelled data.
    - Improves accuracy by utilizing knowledge from large datasets.

Common Pre-Trained Models:

- **VGG16/VGG19** – Simple but large models.
- **ResNet** – Efficient deep networks with skip connections.
- **Inception, EfficientNet** – Optimized for high accuracy with fewer parameters.

In **image classification**, TL is widely used, where models trained on **ImageNet** (millions of images) are adapted for tasks like medical imaging, object detection, and more.

Code:-

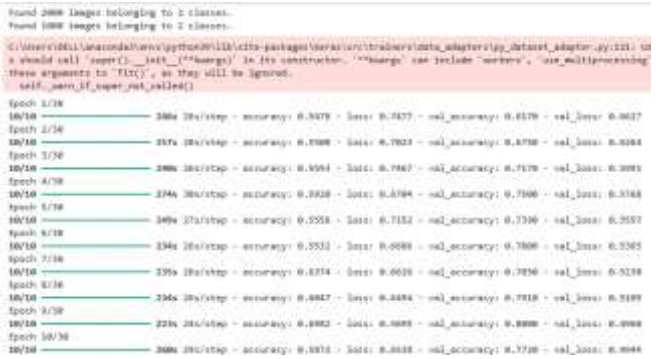
```
import tensorflow as tf

import numpy as np

import matplotlib.pyplot as plt

import os

import zipfile

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.applications import VGG16
```

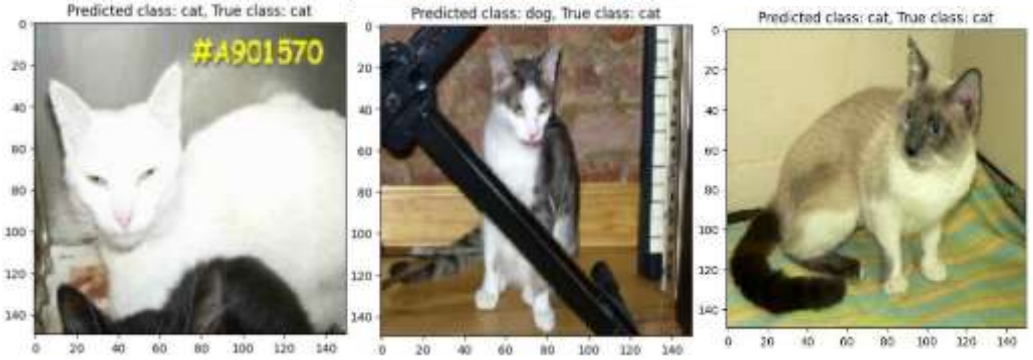
```
# Define cache directory and filename
cache_dir = os.getcwd()  # Set the current working directory as the cache directory
filename = "cats_and_dogs_filtered.zip"  # Name of the file to be downloaded
url = "https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip"
# Download the file
file_path = tf.keras.utils.get_file(filename, url, cache_dir=cache_dir)
# Extract the file
with zipfile.ZipFile(file_path, "r") as zip_ref:
    zip_ref.extractall()
# Define directories
train_dir = os.path.join(os.getcwd(), "cats_and_dogs_filtered", "train")
validation_dir = os.path.join(os.getcwd(), "cats_and_dogs_filtered", "validation")
# Data augmentation and rescaling for training and validation data
train_datagen = ImageDataGenerator(rescale=1./255, rotation_range=20,
width_shift_range=0.2, height_shift_range=0.2, shear_range=0.2, zoom_range=0.2,
horizontal_flip=True)
validation_datagen = ImageDataGenerator(rescale=1./255)
# Image generators
train_generator = train_datagen.flow_from_directory(train_dir, target_size=(150,150),
batch_size=20, class_mode="binary")
validation_generator = validation_datagen.flow_from_directory(validation_dir,
target_size=(150,150), batch_size=20, class_mode="binary")
# Load pre-trained VGG16 model
conv_base = VGG16(weights="imagenet", include_top=False, input_shape=(150, 150, 3))
# Freeze the convolutional base
conv_base.trainable = False
# Build the model
model = tf.keras.models.Sequential()
model.add(conv_base)
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(256, activation="relu"))
```

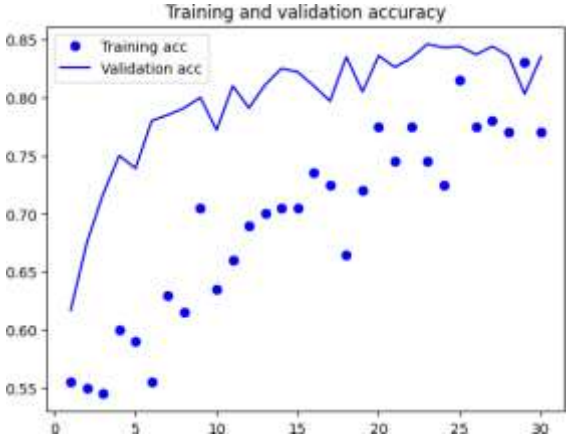
```
model.add(tf.keras.layers.Dropout(0.5))

model.add(tf.keras.layers.Dense(1, activation="sigmoid"))

# Compile the model

model.compile(loss="binary_crossentropy",optimizer=tf.keras.optimizers.RMSprop(learning
_rate=2e-5), metrics=["accuracy"])

# Train the model

history = model.fit(train_generator, steps_per_epoch=10, epochs=30,
validation_data=validation_generator, validation_steps=50)

# Display predictions for some validation images

x, y_true = next(validation_generator)

y_pred = model.predict(x)

class_names = ['cat', 'dog']

for i in range(len(x)):

    plt.imshow(x[i])

    plt.title(f'Predicted class: {class_names[int(round(y_pred[i][0]))]}, True class:
{class_names[int(y_true[i])]}')

    plt.show()

# Plot training and validation accuracy

acc = history.history["accuracy"]

val_acc = history.history["val_accuracy"]

loss = history.history["loss"]

val_loss = history.history["val_loss"]

epochs = range(1, len(acc) + 1)

# Accuracy plot

plt.plot(epochs, acc, "bo", label="Training acc")

plt.plot(epochs, val_acc, "b", label="Validation acc")

plt.title("Training and validation accuracy")

plt.legend()

# Loss plot

plt.figure()

plt.plot(epochs, loss, "bo", label="Training loss")
```

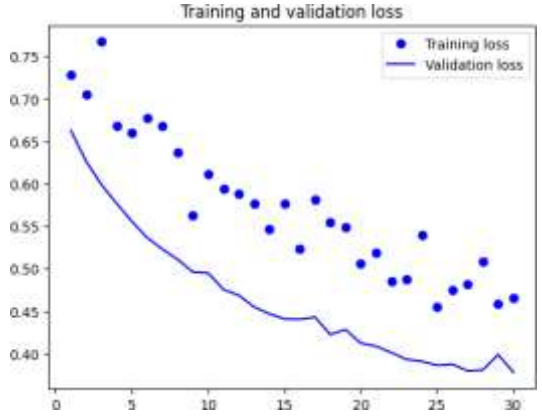
```
plt.plot(epochs, val_loss, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.legend()
plt.show()
```

Output:-




Predicted class: cat, True class: cat
#A901570
Predicted class: dog, True class: cat
Predicted class: cat, True class: cat


Training and validation accuracy
Training acc
Validation acc


Training and validation loss
Training loss
Validation loss

Practical: 9

Aim:- Build an autoencoder for the MNIST dataset to encode and decode digit images, reducing dimensionality while reconstructing the original data.

Theory:-

An **autoencoder** is a type of neural network used for **unsupervised learning** that learns to encode input data into a compressed representation and then reconstruct it back to the original form. It consists of two main parts:

1. **Encoder** – Compresses input data into a lower-dimensional latent space.
2. **Decoder** – Reconstructs the input from this compressed representation.

Key Concepts:

- **Dimensionality Reduction** – Learns efficient representations, similar to PCA but non-linear.
- **Noise Reduction (Denoising Autoencoders)** – Learns to remove noise from data.
- **Anomaly Detection** – Identifies unusual patterns by detecting high reconstruction errors.

Types of Autoencoders:

- **Vanilla Autoencoder** – Basic encoder-decoder structure.
- **Denoising Autoencoder** – Trained to remove noise from input.
- **Sparse Autoencoder** – Uses sparsity constraints for efficient feature learning.
- **Variational Autoencoder (VAE)** – Learns probabilistic latent space for generating new data.

Autoencoders are widely used in **image compression, feature extraction, anomaly detection, and generative modeling**.

Code:-

```
import numpy as np

import matplotlib.pyplot as plt

import tensorflow as tf

from tensorflow.keras.layers import Input, Dense

from tensorflow.keras.models import Model

from tensorflow.keras.datasets import mnist

#load

(x_train, _), (x_test, _) = mnist.load_data()
```

```
#normalize
x_train = x_train.astype('float32') / 255.
x_test = x_test.astype('float32') / 255.
x_train = x_train.reshape((len(x_train),np.prod(x_train.shape[1:])))
x_test = x_test.reshape((len(x_test),np.prod(x_test.shape[1:])))
#size
encoding_dim = 32
#input
input_img = Input(shape=(784,))
#encoded
encoded = Dense(encoding_dim, activation='relu')(input_img)
#decoded
decoded = Dense(784, activation='sigmoid')(encoded)
#this model maps its reconstruction
autoencoder = Model(input_img, decoded)
#this model maps its    encoded representation
encoder = Model(input_img, encoded)
#create
encoded_input = Input(shape=(encoding_dim,))
#retrive
decoder_layer =autoencoder.layers[-1]
#create
decoder = Model(encoded_input, decoder_layer(encoded_input))
autoencoder.compile(optimizer='adam',  loss='binary_crossentropy')
autoencoder.fit(x_train, x_train, epochs=50, batch_size=256, shuffle=True,
validation_data=(x_test, x_test))
#encode
encoded_imgs = encoder.predict(x_test)
decoded_imgs = decoder.predict(encoded_imgs)
#use
n=10
```

```
plt.figure(figsize=(20,4))
for i in range(n):
    ax = plt.subplot(2, n, i + 1)
    plt.imshow(x_test[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

Output:-

```
Epoch 1/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 6s 16ms/step - loss: 0.3806 - val_loss: 0.1893
Epoch 2/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.1791 - val_loss: 0.1523
Epoch 3/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.1486 - val_loss: 0.1341
Epoch 4/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.1324 - val_loss: 0.1220
Epoch 5/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 5s 13ms/step - loss: 0.1213 - val_loss: 0.1140
Epoch 6/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.1139 - val_loss: 0.1082
Epoch 7/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.1084 - val_loss: 0.1037
Epoch 8/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 14ms/step - loss: 0.1044 - val_loss: 0.1003
Epoch 9/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.1011 - val_loss: 0.0977
Epoch 10/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.0985 - val_loss: 0.0958
Epoch 11/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 14ms/step - loss: 0.0968 - val_loss: 0.0946
Epoch 12/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.0955 - val_loss: 0.0939
Epoch 13/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - loss: 0.0952 - val_loss: 0.0934
```

```
Epoch 46/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 15ms/step - loss: 0.0927 - val_loss: 0.0915
Epoch 47/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 16ms/step - loss: 0.0925 - val_loss: 0.0916
Epoch 48/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 17ms/step - loss: 0.0925 - val_loss: 0.0915
Epoch 49/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 16ms/step - loss: 0.0927 - val_loss: 0.0915
Epoch 50/50
235/235 ━━━━━━━━━━━━━━━━━━━━ 4s 16ms/step - loss: 0.0927 - val_loss: 0.0915
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
```

Practical: 10

Aim:- Implement a Generative Adversarial Network (GAN) to generate realistic handwritten digits based on the MNIST dataset.

Theory:-

A **Generative Adversarial Network (GAN)** is a deep learning framework that consists of two neural networks competing against each other:

1. **Generator (G)** – Creates synthetic data resembling real data.
2. **Discriminator (D)** – Distinguishes between real and fake data.

These networks are trained in an **adversarial process**, where the **generator tries to fool the discriminator**, and the **discriminator improves at detecting fakes**. Over time, the generator produces highly realistic data.

Key Concepts:

- **Adversarial Training** – Both networks improve iteratively through competition.
- **Loss Function** – Uses a minimax game where the generator minimizes the discriminator's accuracy while the discriminator maximizes it.
- **Latent Space** – Random noise is transformed into realistic outputs.

Types of GANs:

- **Vanilla GAN** – Basic GAN architecture.
- **DCGAN (Deep Convolutional GAN)** – Uses CNNs for better image generation.
- **WGAN (Wasserstein GAN)** – Improves training stability with Wasserstein loss.
- **StyleGAN** – Generates high-quality images with style control.

Applications:

- Image Generation (e.g., DeepFake, AI Art)
- Super-Resolution (Enhancing low-res images)
- Data Augmentation
- Drug Discovery

GANs are powerful for generating realistic data but are challenging to train due to issues like **mode collapse** and **instability**.

Code:-

```
import tensorflow as tf

import numpy as np
```

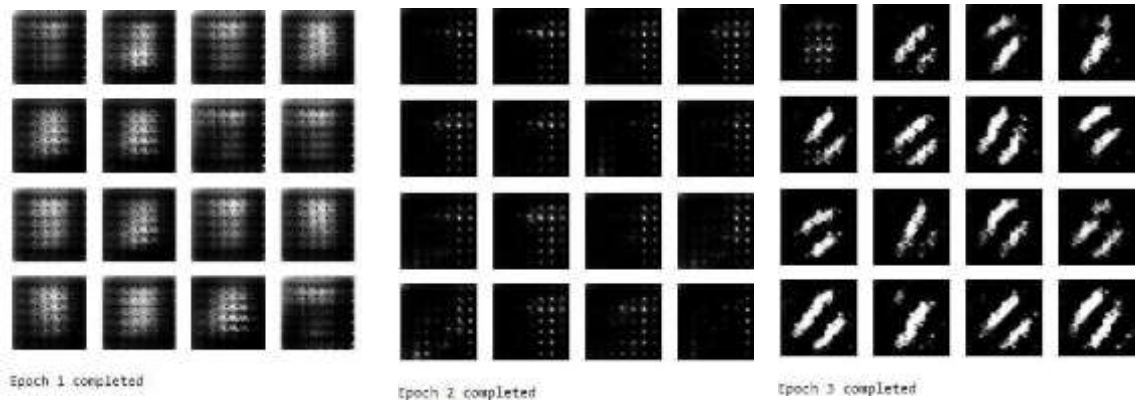
```
import matplotlib.pyplot as plt

# Load the MNIST dataset

mnist = tf.keras.datasets.mnist

(train_images, train_labels), (_, _) = mnist.load_data()

# Normalize the images to [-1, 1]

train_images = (train_images.astype('float32') - 127.5) / 127.5

# Reshape the images to (28, 28, 1) and add a channel dimension

train_images = np.expand_dims(train_images, axis=-1)

BUFFER_SIZE = 60000

BATCH_SIZE = 256

train_dataset =
tf.data.Dataset.from_tensor_slices(train_images).shuffle(BUFFER_SIZE).batch(BATCH_SI
ZE)

# Create the generator model

def make_generator_model():

    model = tf.keras.Sequential()

    model.add(tf.keras.layers.Dense(7*7*256, use_bias=False, input_shape=(100,)))

    model.add(tf.keras.layers.BatchNormalization())

    model.add(tf.keras.layers.LeakyReLU())

    model.add(tf.keras.layers.Reshape((7, 7, 256)))

    assert model.output_shape == (None, 7, 7, 256)

    model.add(tf.keras.layers.Conv2DTranspose(128, (5, 5), strides=(1, 1), padding='same',
use_bias=False))

    assert model.output_shape == (None, 7, 7, 128)

    model.add(tf.keras.layers.BatchNormalization())

    model.add(tf.keras.layers.LeakyReLU())


    model.add(tf.keras.layers.Conv2DTranspose(64, (5, 5), strides=(2, 2), padding='same',
use_bias=False))

    assert model.output_shape == (None, 14, 14, 64)

    model.add(tf.keras.layers.BatchNormalization())

    model.add(tf.keras.layers.LeakyReLU())
```

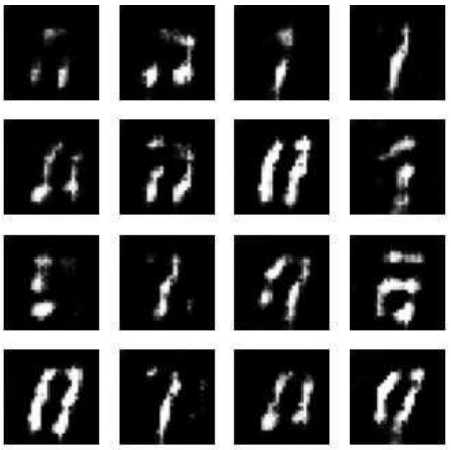
```
        model.add(tf.keras.layers.Conv2DTranspose(1, (5, 5), strides=(2, 2), padding='same',
use_bias=False, activation="tanh"))
        assert model.output_shape == (None, 28, 28, 1)
        return model
  # Create the discriminator model
 def make_discriminator_model():
        model = tf.keras.Sequential()
        model.add(tf.keras.layers.Conv2D(64, (5, 5), strides=(2, 2), padding='same',
input_shape=[28, 28, 1]))
        model.add(tf.keras.layers.LeakyReLU())
        model.add(tf.keras.layers.Dropout(0.3))
        model.add(tf.keras.layers.Conv2D(128, (5, 5), strides=(2, 2), padding='same'))
        model.add(tf.keras.layers.LeakyReLU())
        model.add(tf.keras.layers.Dropout(0.3))
        model.add(tf.keras.layers.Flatten())
        model.add(tf.keras.layers.Dense(1))
        return model
# Loss functions
cross_entropy = tf.keras.losses.BinaryCrossentropy(from_logits=True)


def discriminator_loss(real_output, fake_output):
        real_loss = cross_entropy(tf.ones_like(real_output), real_output)
        fake_loss = cross_entropy(tf.zeros_like(fake_output), fake_output)
        total_loss = real_loss + fake_loss
        return total_loss
def generator_loss(fake_output):
        return cross_entropy(tf.ones_like(fake_output), fake_output)
# Define the models
generator = make_generator_model()
discriminator = make_discriminator_model()
# Define the optimizers
```

```
generator_optimizer = tf.keras.optimizers.Adam(1e-4)
discriminator_optimizer = tf.keras.optimizers.Adam(1e-4)
# Training loop settings
EPOCHS = 5
noise_dim = 100
num_examples_to_generate = 16
# Training step function
@tf.function
def train_step(images):
    noise = tf.random.normal([BATCH_SIZE, noise_dim])
    with tf.GradientTape() as gen_tape, tf.GradientTape() as disc_tape:
        generated_images = generator(noise, training=True)

        real_output = discriminator(images, training=True)
        fake_output = discriminator(generated_images, training=True)

        gen_loss = generator_loss(fake_output)
        disc_loss = discriminator_loss(real_output, fake_output)

    gradients_of_generator = gen_tape.gradient(gen_loss, generator.trainable_variables)
    gradients_of_discriminator = disc_tape.gradient(disc_loss,
discriminator.trainable_variables)

    generator_optimizer.apply_gradients(zip(gradients_of_generator,
generator.trainable_variables))
    discriminator_optimizer.apply_gradients(zip(gradients_of_discriminator,
discriminator.trainable_variables))
# Image generation function
def generate_and_save_images(model, epoch, test_input):
    predictions = model(test_input, training=False)
    predictions = (predictions + 1) / 2.0    # Rescale to [0, 1]
```

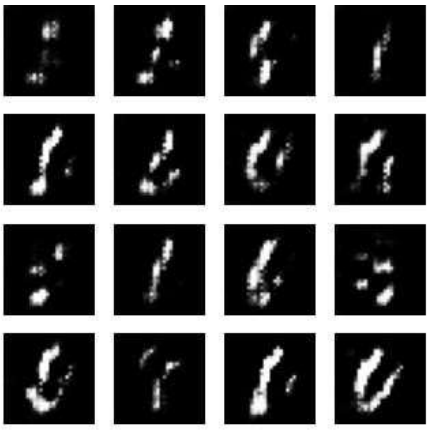
```
    fig = plt.figure(figsize=(4, 4))
    for i in range(predictions.shape[0]):
        plt.subplot(4, 4, i+1)
        plt.imshow(predictions[i, :, :, 0], cmap='gray')
        plt.axis('off')

    plt.savefig('image_at_epoch_{:04d}.png'.format(epoch))
    plt.show()
# Generate a fixed set of noise for evaluating the model during training
fixed_noise = tf.random.normal([num_examples_to_generate, noise_dim])
# Train the model
for epoch in range(EPOCHS):
    for image_batch in train_dataset:
        train_step(image_batch)
    generate_and_save_images(generator, epoch + 1, fixed_noise)    # Now saving images at every epoch
    print(f'Epoch {epoch + 1} completed')
# Generate final images after training
print("Training completed. Generating final images...")
generate_and_save_images(generator, EPOCHS, fixed_noise)
```

Output:-

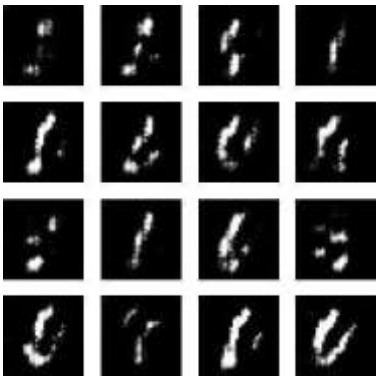




Epoch 4 completed



Epoch 5 completed
Training completed. Generating final images...



INDEX

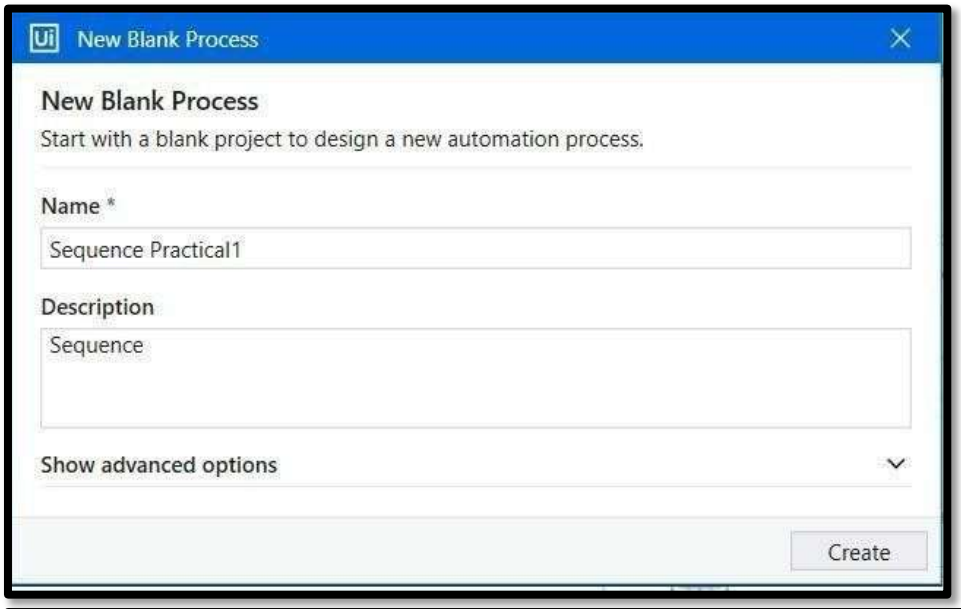
	f	Automate the process of launching an assistant bot on a keyboard event	18/02/25	45	
4		Exception Handling, Debugging, and Logging			
	a	Automate the process of send mail event (on any email).	22/02/25	48	
	b	Demonstrate the Exception handing in UiPath.	01/03/25	49	
	c	Demonstrate the use of config files in UiPath.	03/03/25	52	
	d	Install and automate any process using UiPath with plug-ins:	07/03/25	53	

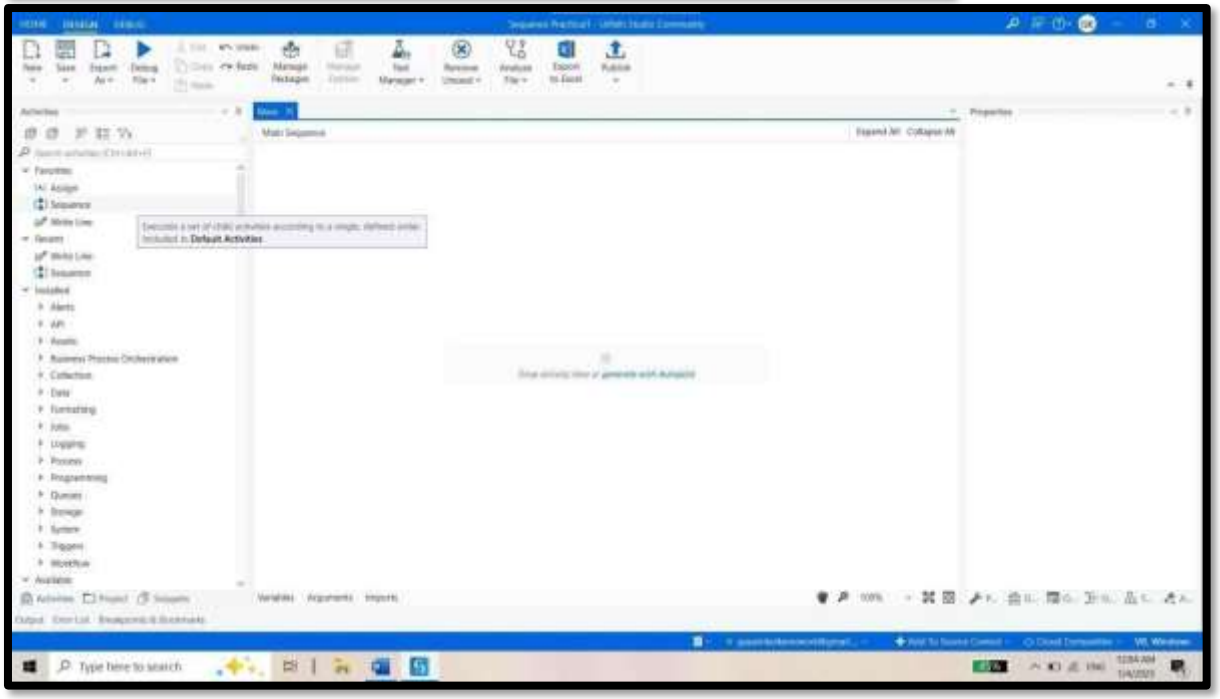
Module 1

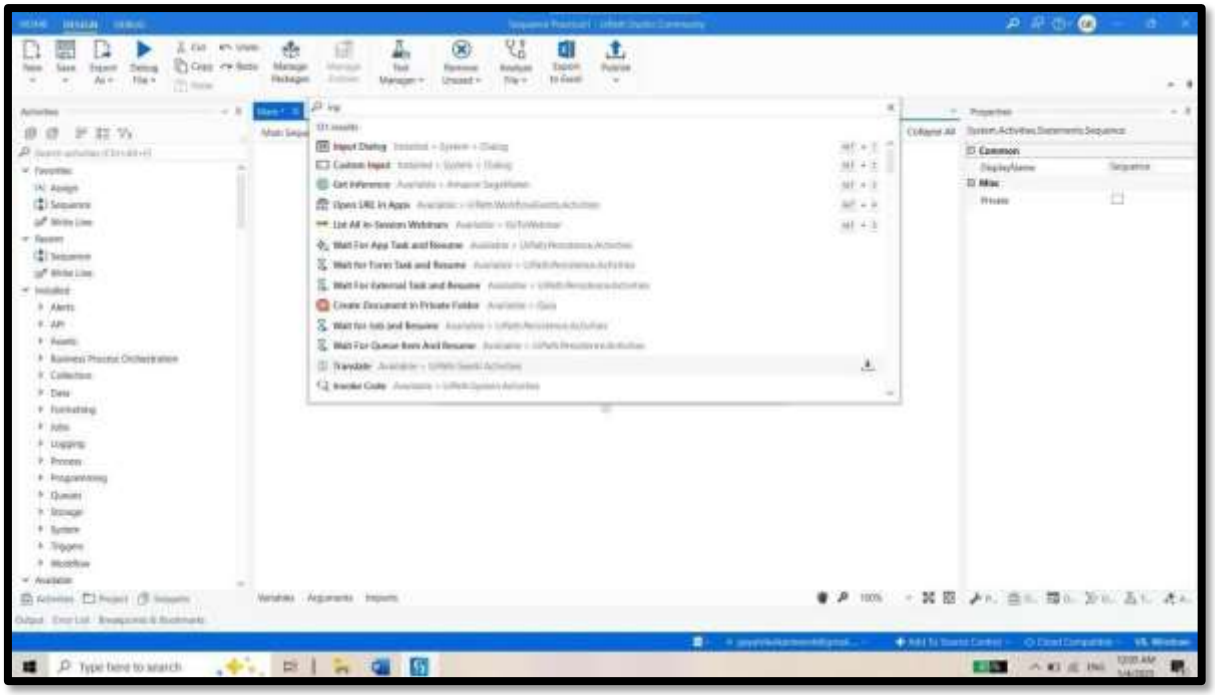
Contents:- UiPath stack, Downloading and installing UiPath Studio, Learning UiPath Studio, Task recorder, Step-by-step examples using the recorder. Sequence, Flowchart, and Control Flow: Sequencing the workflow, Activities, Control flow, various types of loops, and decision making, Step-by-step example using Sequence and Flowchart, Step-by-step example using Sequence and Control flow, Step-by-step examples

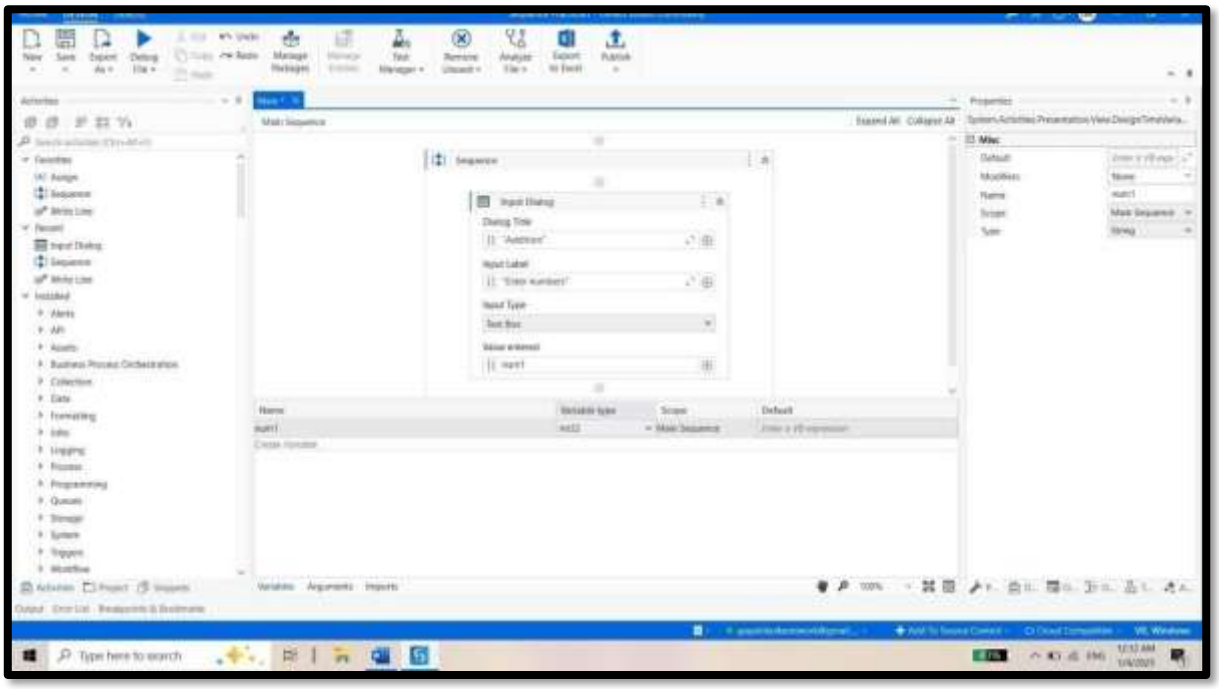
using the recorder.

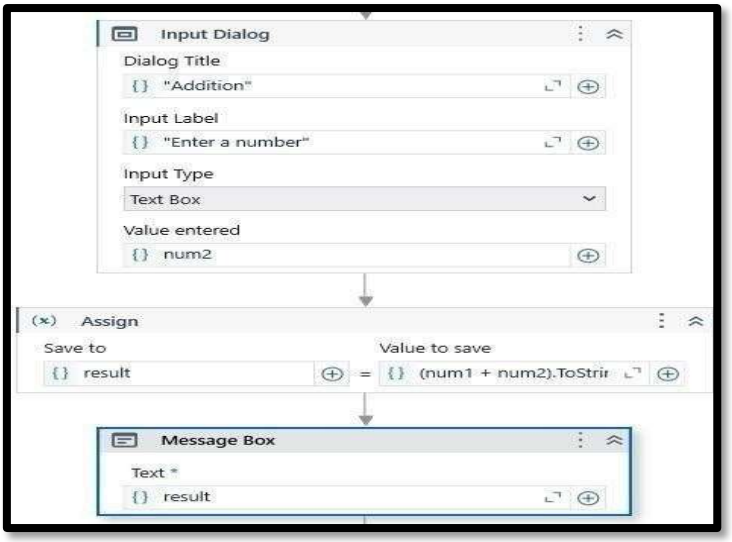
a. Create a simple sequence based project.

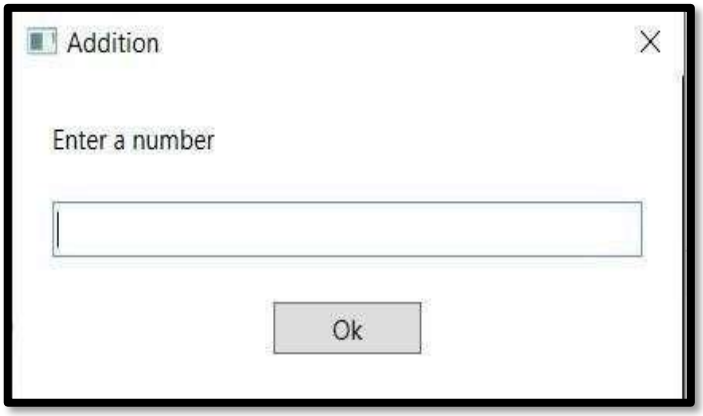


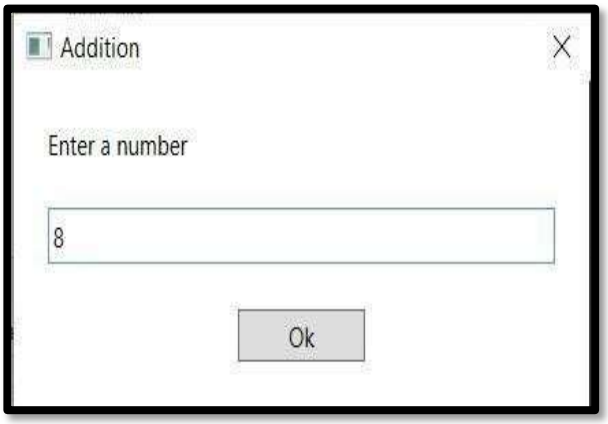







Input Dialog
Dialog Title
{} "Addition"
Input Label
{} "Enter a number"
Input Type
Text Box
Value entered
{} num2
Assign
Save to
{} result
Value to save
{} (num1 + num2).ToStri
Message Box
Text *
{} result

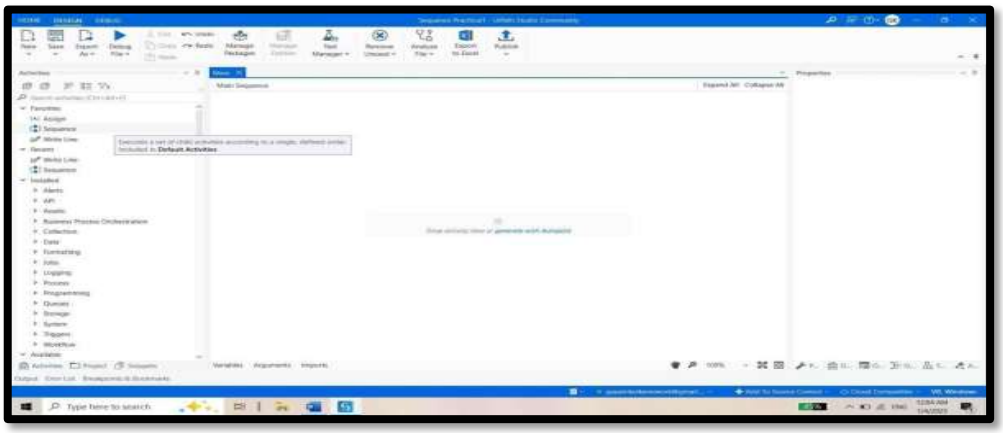

Addition
Enter a number
Ok

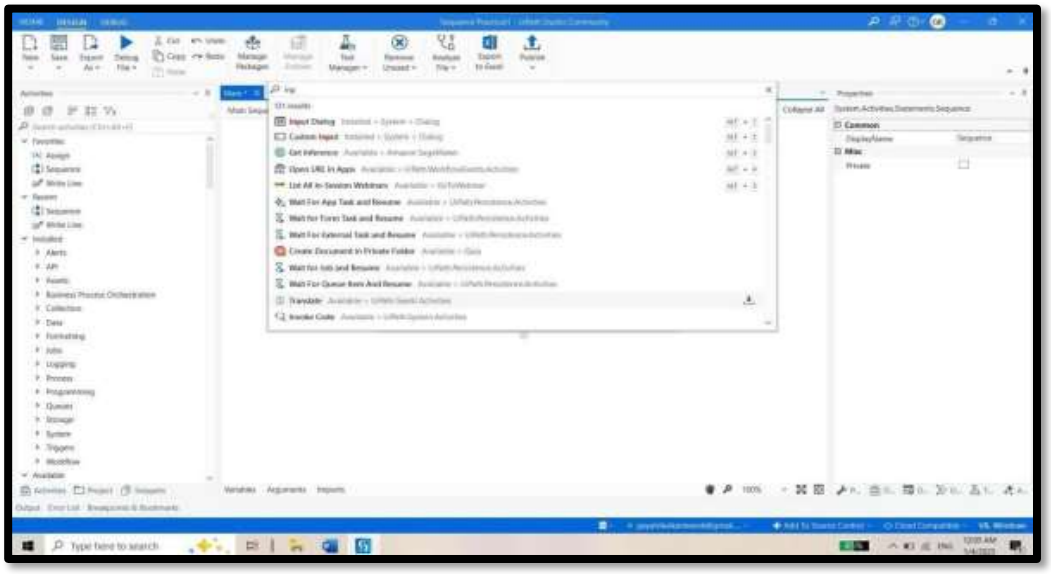

Addition
Enter a number
8
Ok

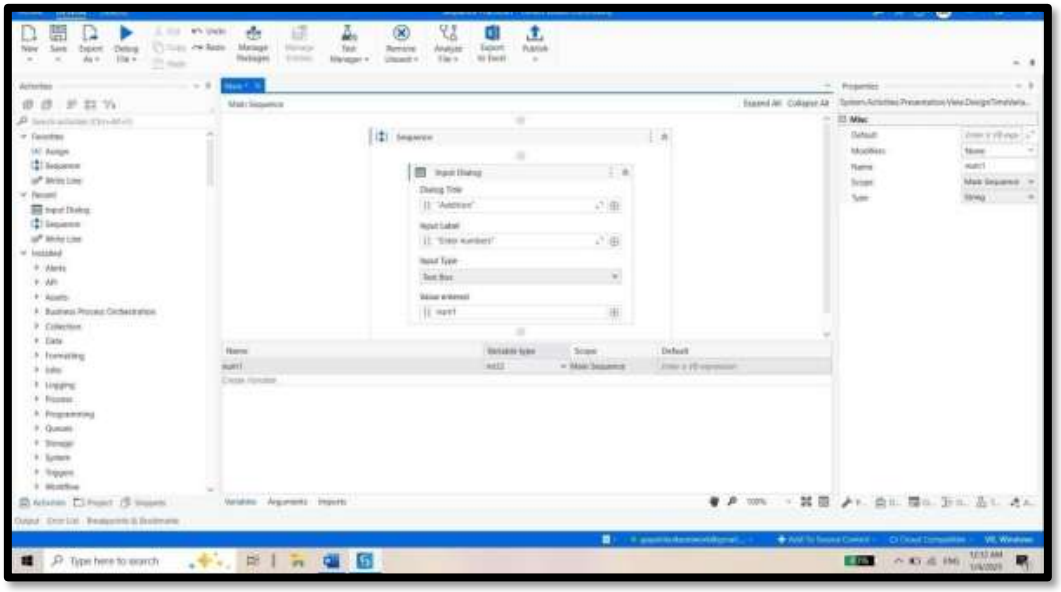

Addition
Enter a number
4
Ok

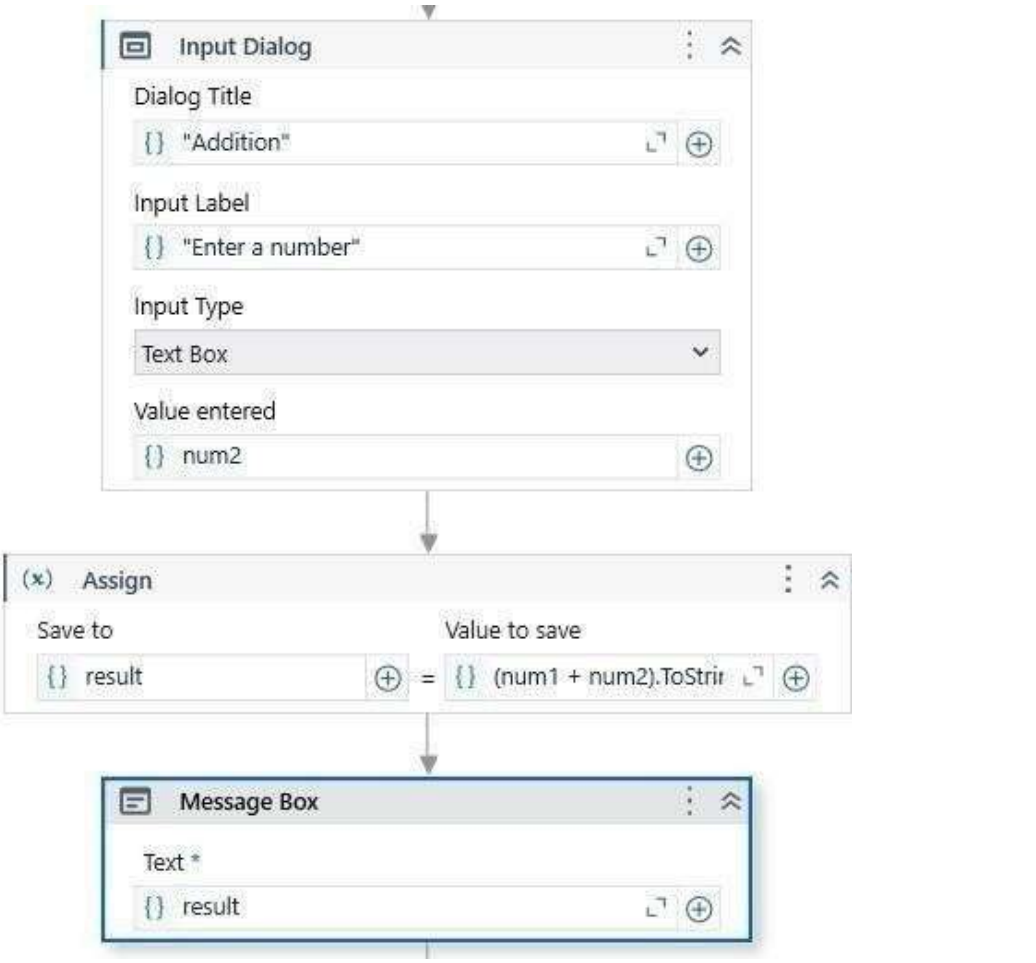

Message ...
12
OK

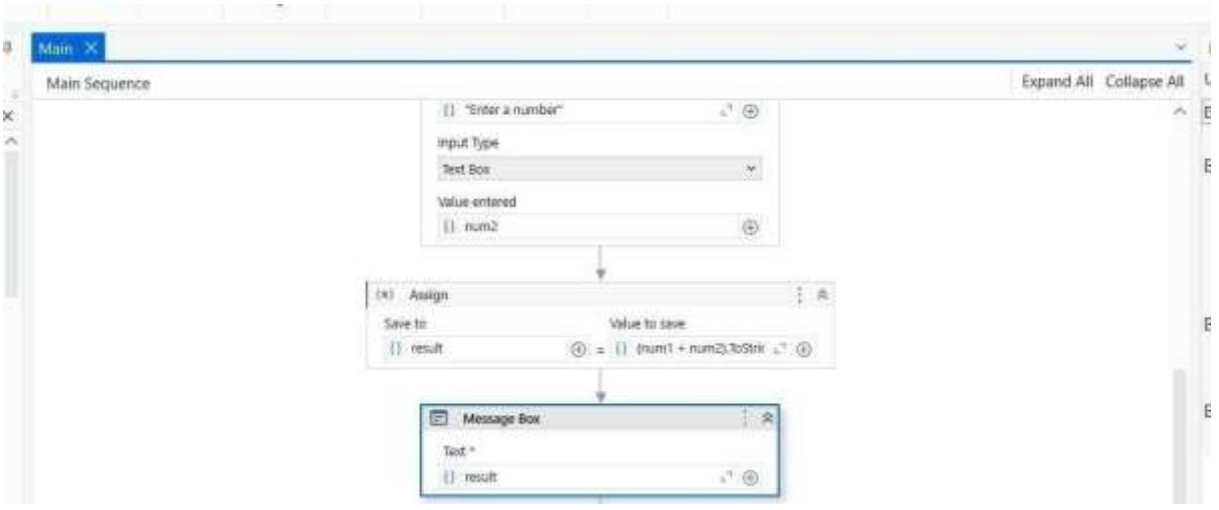
a) Create a flowchart-based project.

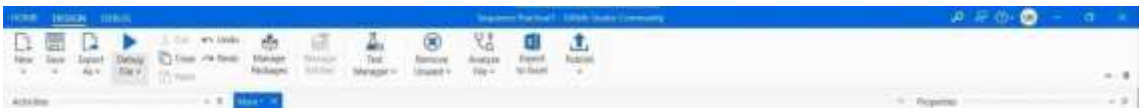


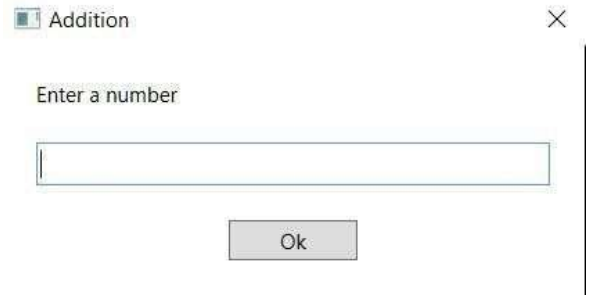


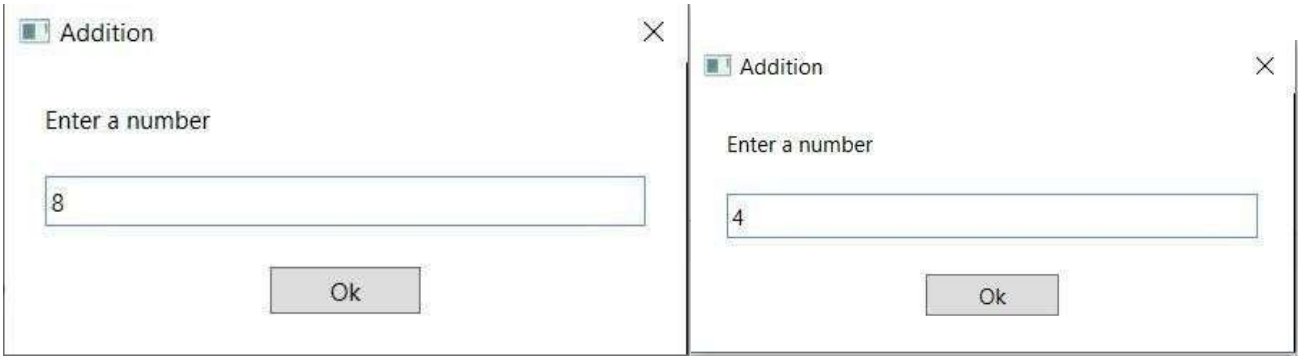



Input Dialog
Dialog Title
"Addition"
Input Label
"Enter a number"
Input Type
Text Box
Value entered
num2
Assign
Save to
result
Value to save
(num1 + num2).ToStrir
Message Box
Text *
result

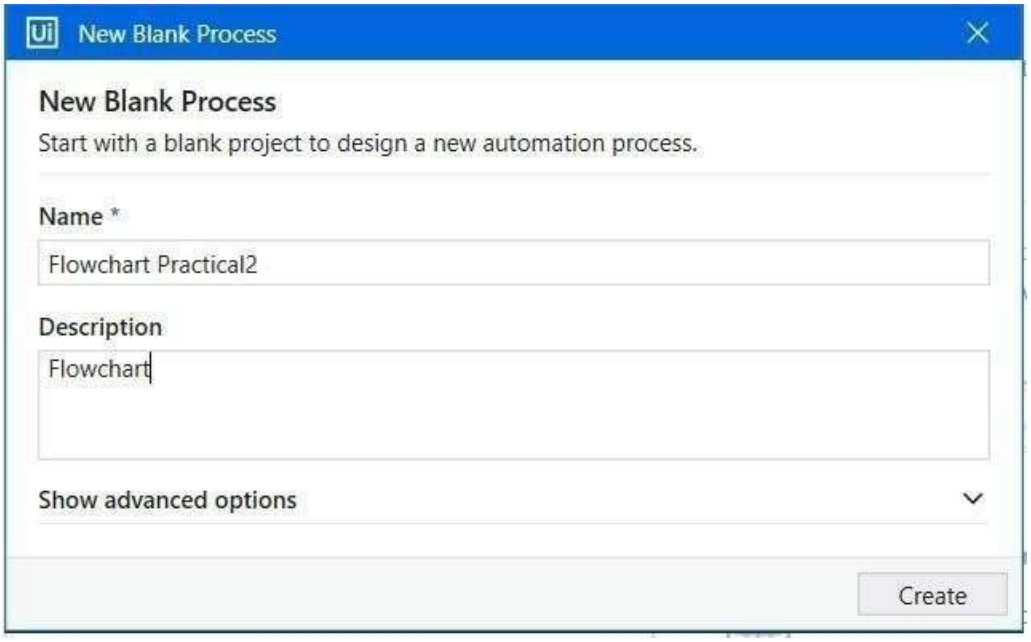

Main
Main Sequence
Expand All
Collapse All
"Enter a number"
Input Type
Text Box
Value entered
num2
Assign
Save to
Value to save
result
(num1 + num2).ToStrir
Message Box
Text *
result

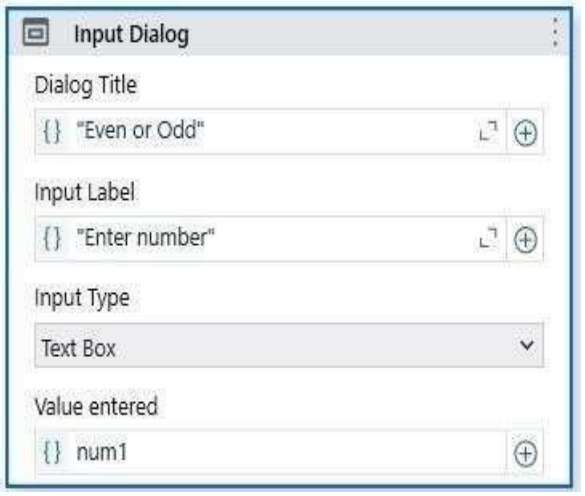


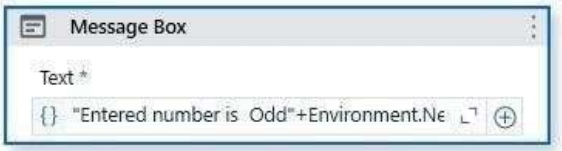


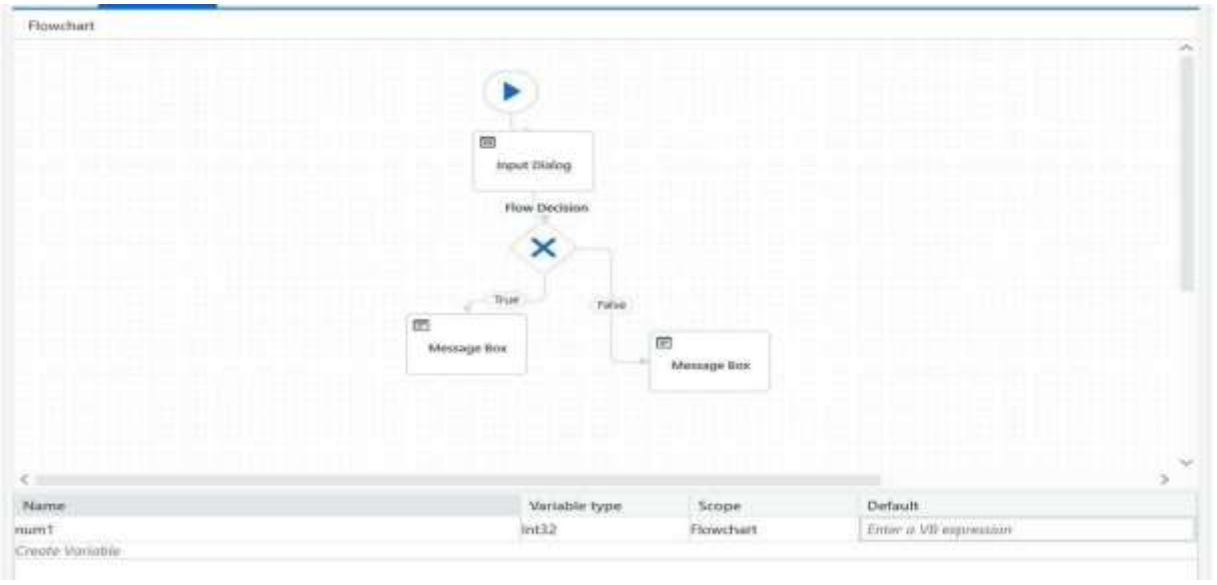


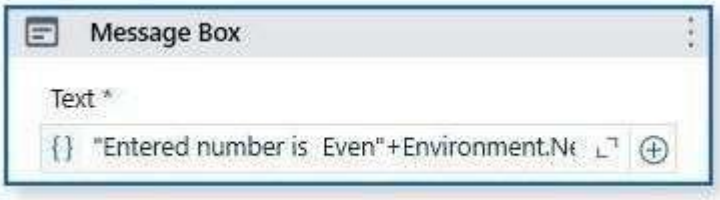
b) Create an automation UiPath Project using decision statements.



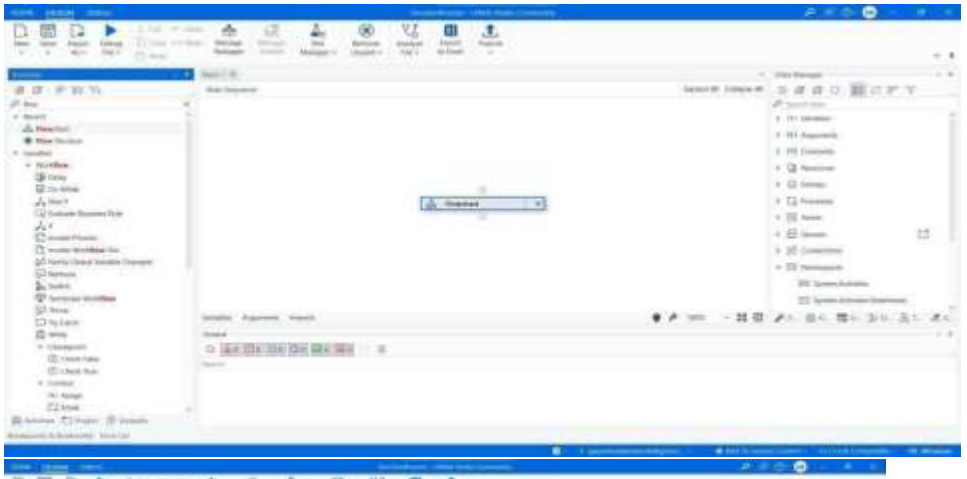

Input Dialog
Dialog Title
{} "Even or Odd"
Input Label
{} "Enter number"
Input Type
Text Box
Value entered
{} num1

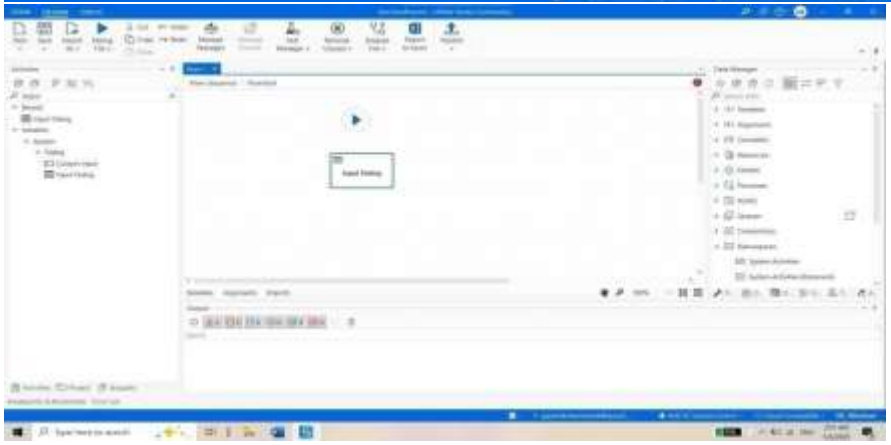

Message Box
Text *
{} "Entered number is Odd"+Environment.Ne

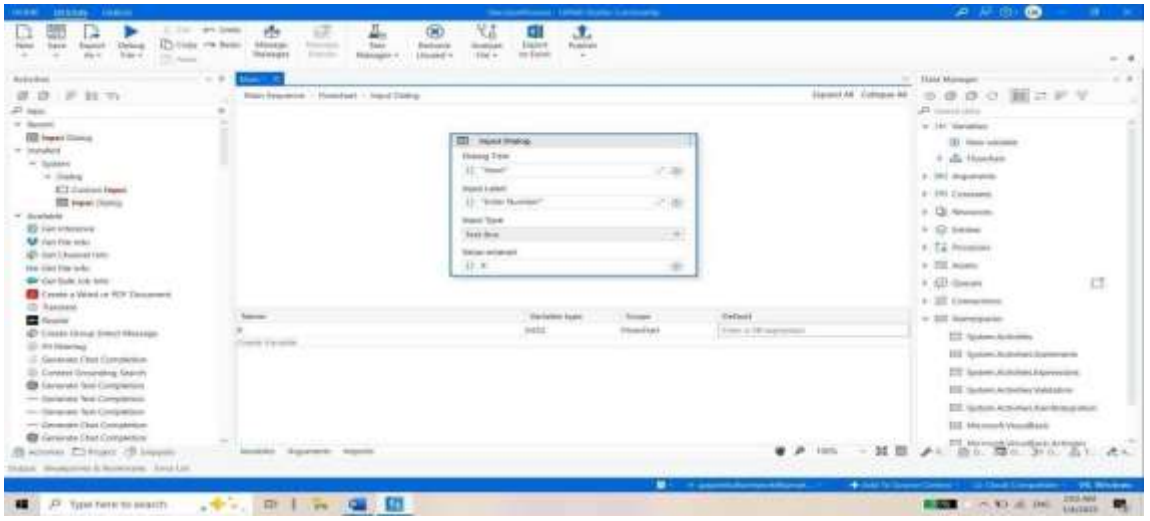

Flowchart
Input Dialog
Flow Decision
True
False
Message Box
Message Box
Name
Variable type
Scope
Default
num1
Int32
Flowchart
Create Variable

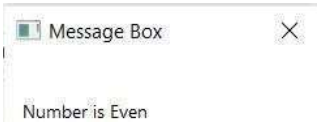

Message Box
Text *
{} "Entered number is Even"+Environment.Ne

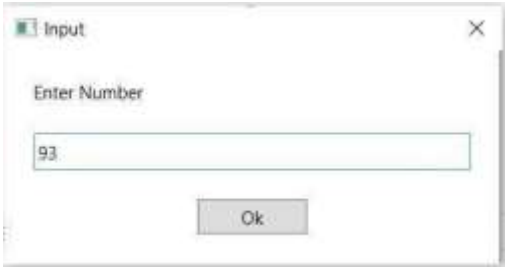
If else





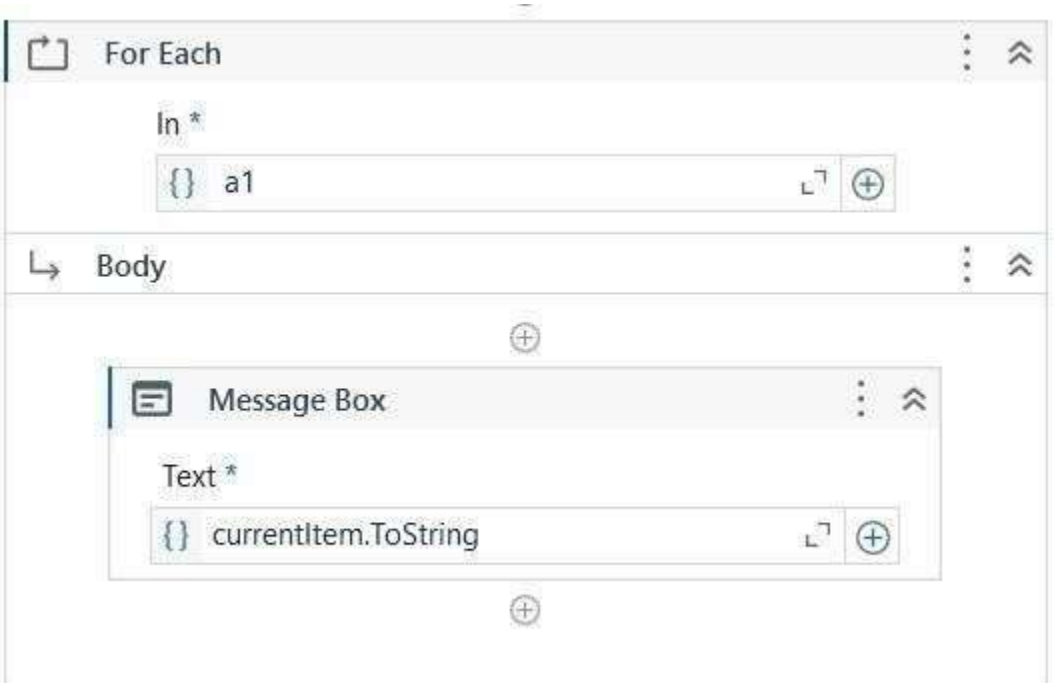


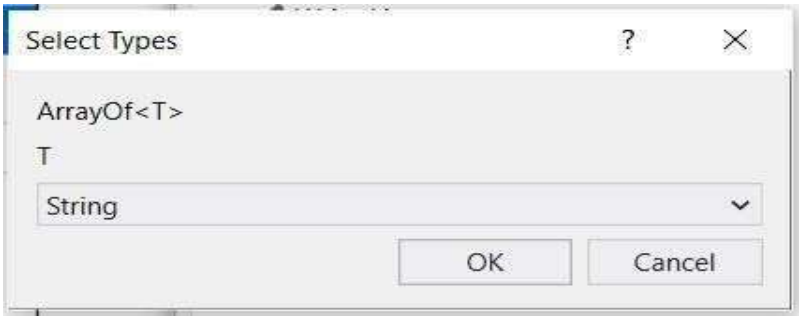


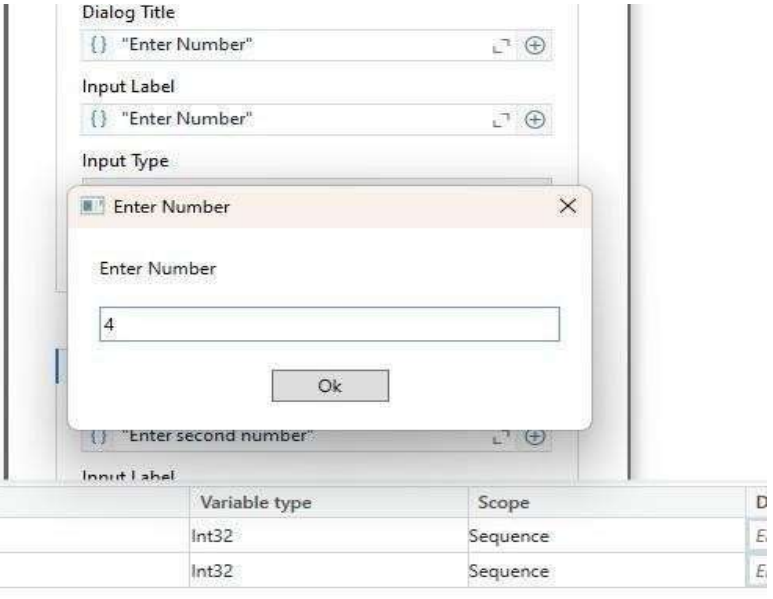


c) Consider an array of names. We have to find out how many of them start with the letter "a". Create an automation where the number of names starting with "a" is counted and the result is displayed.

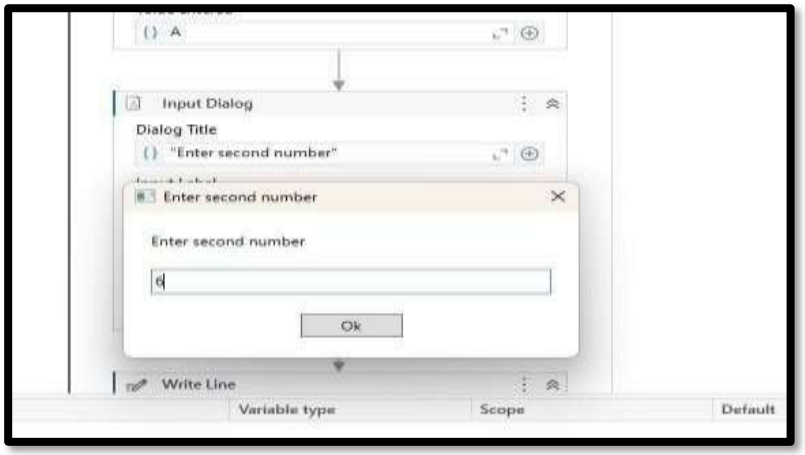
Name	Variable type	Scope	Default
Names	String[]	Flowchart	{"Aarti", "Akashnsha","aparna"}
count	Int32	Main Sequence	Enter a VB expression

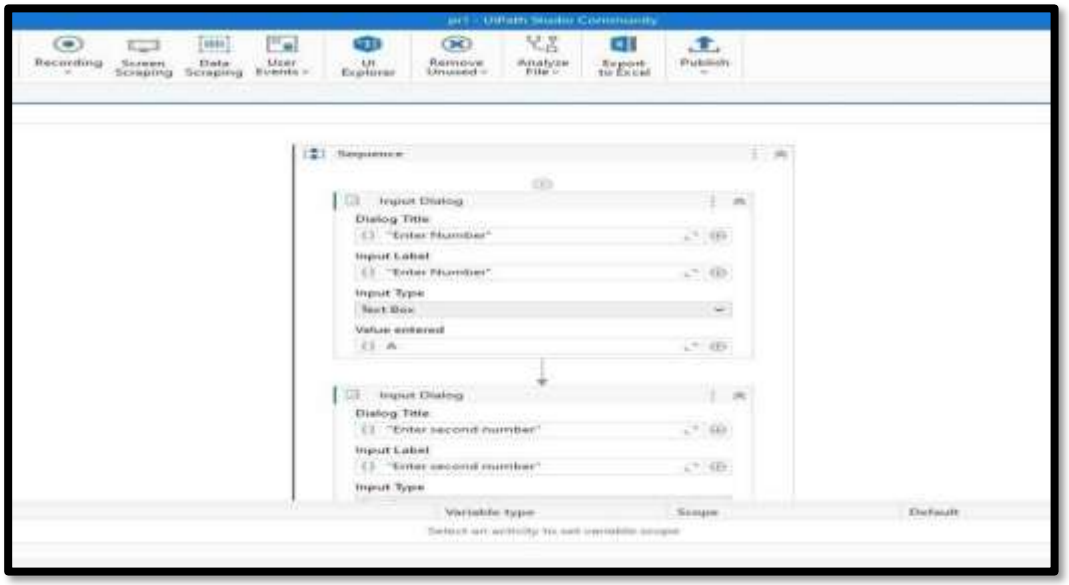


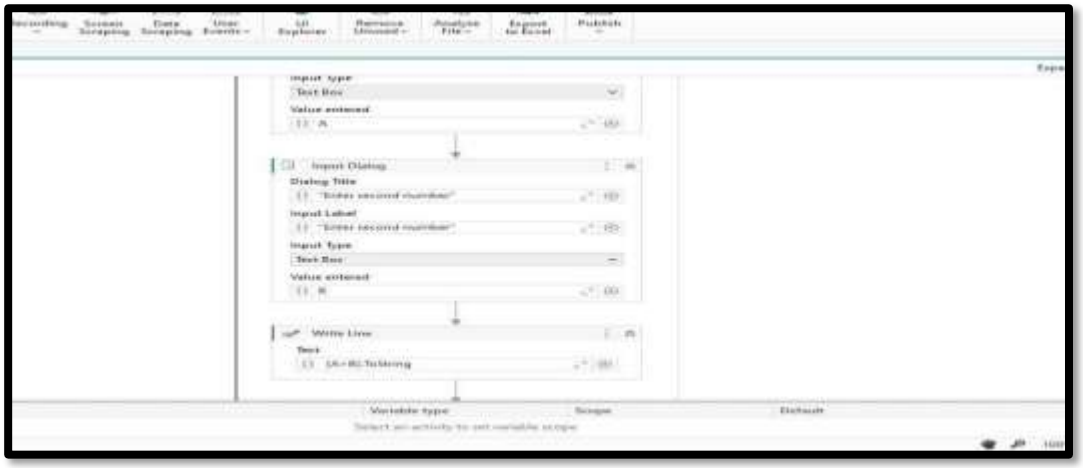


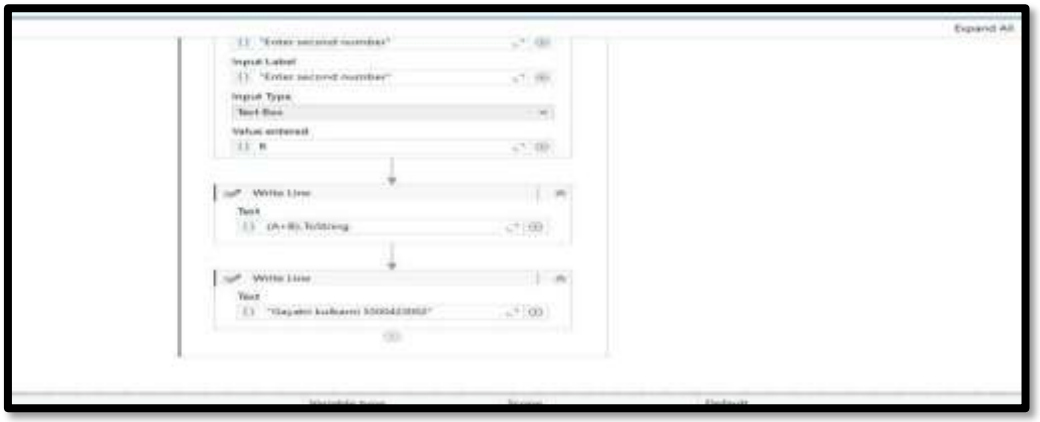


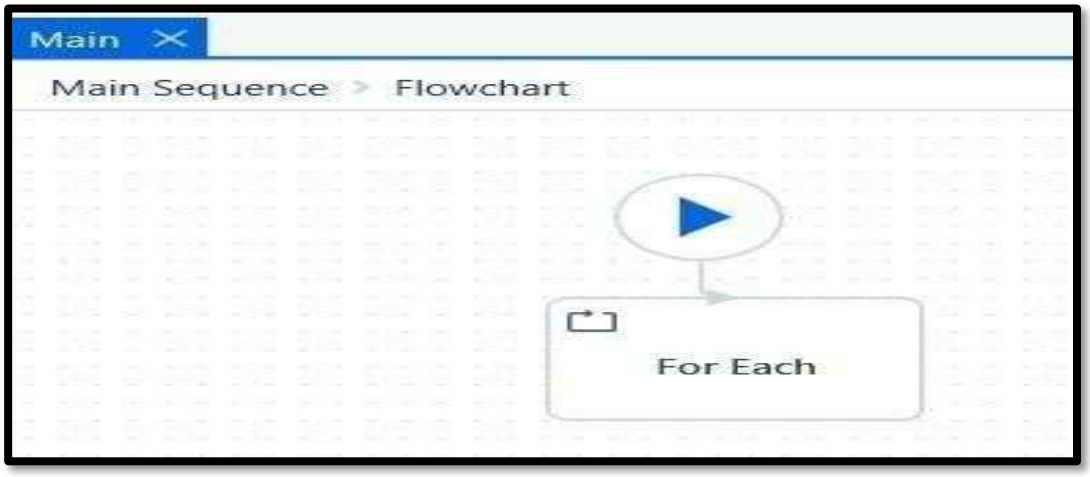
	Variable type	Scope	D
	Int32	Sequence	E
	Int32	Sequence	E

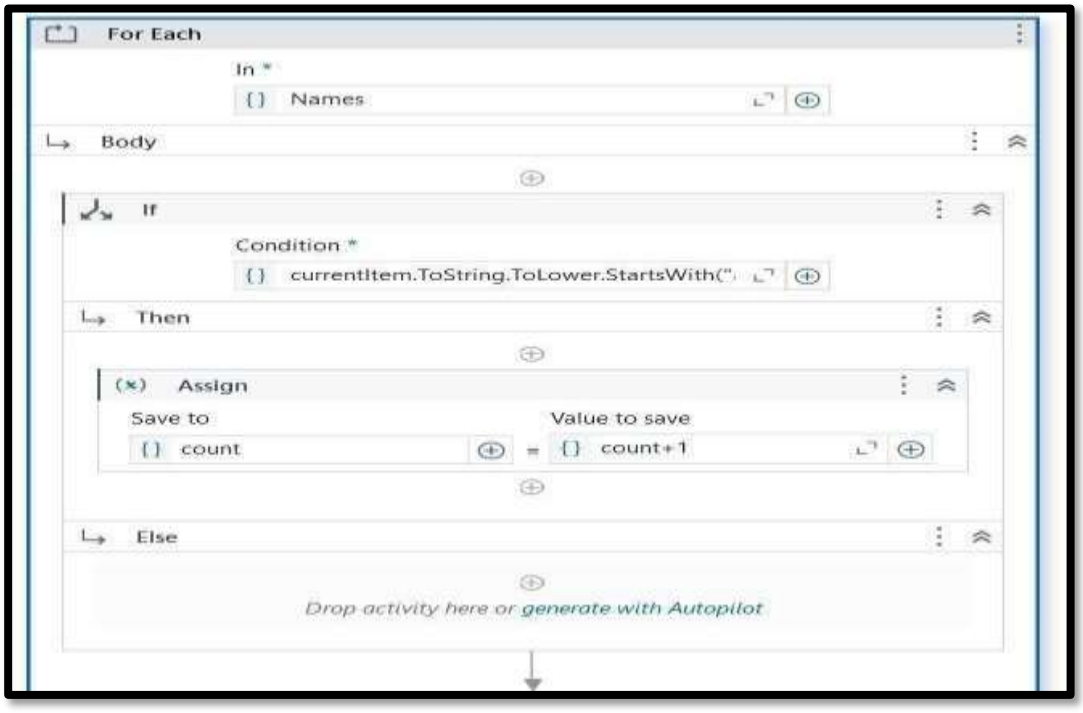

A
Input Dialog
Dialog Title
"Enter second number"
Enter second number
Enter second number
6
Ok
Write Line
Variable type
Scope
Default


Recording
Screen Scraping
Data Scraping
User Events
UI Explorer
Remove Unused
Analyze File
Export to Excel
Publish
Sequence
Input Dialog
Dialog Title
"Enter Number"
Input Label
"Enter Number"
Input Type
Text Box
Value entered
A
Input Dialog
Dialog Title
"Enter second number"
Input Label
"Enter second number"
Input Type
Variable type
Scope
Default



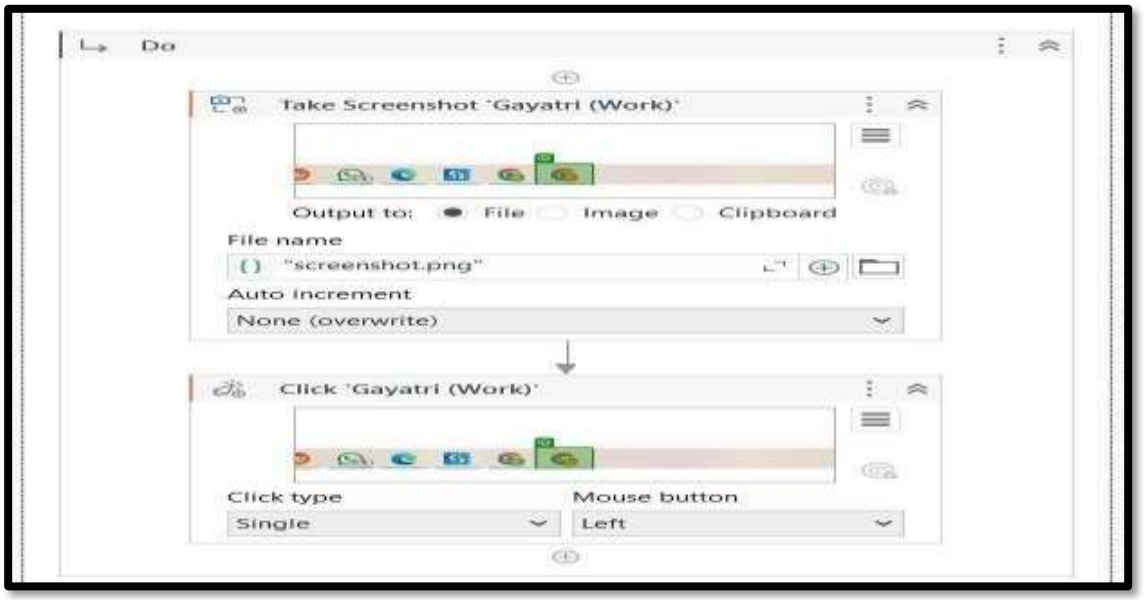

Expand All
Write Line
Write Line

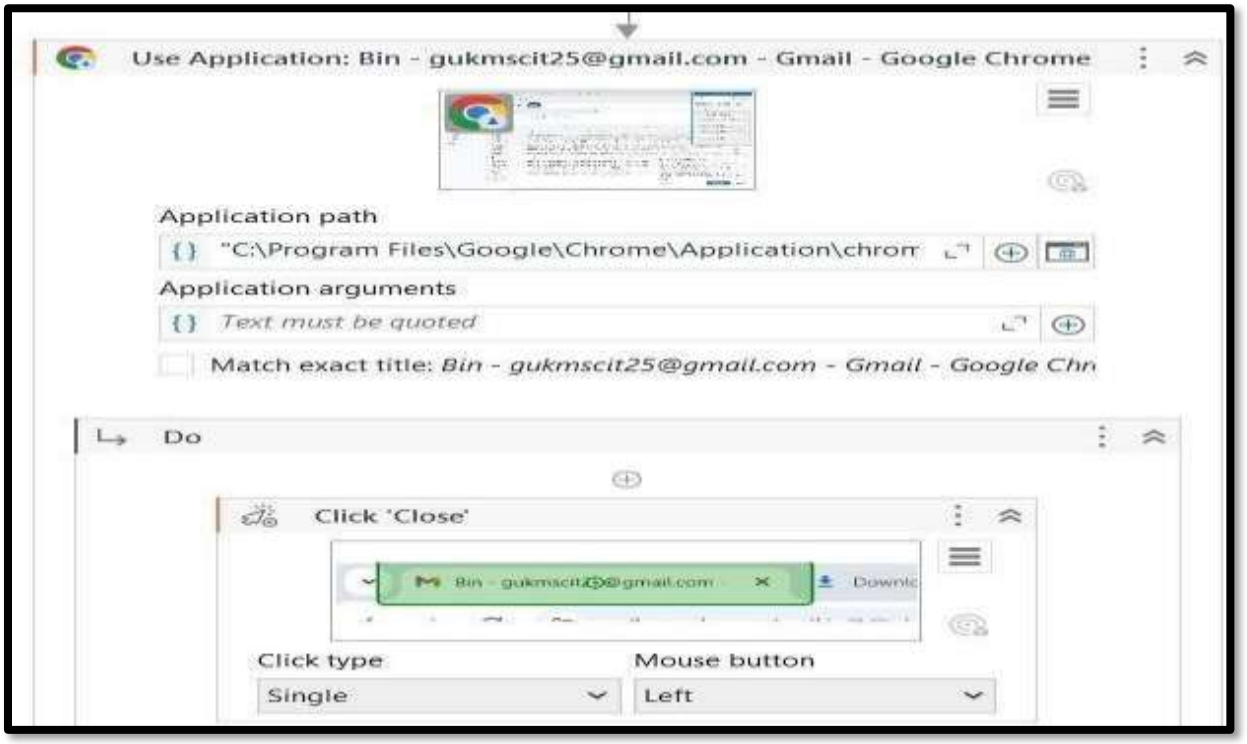

Main
Main Sequence > Flowchart
For Each

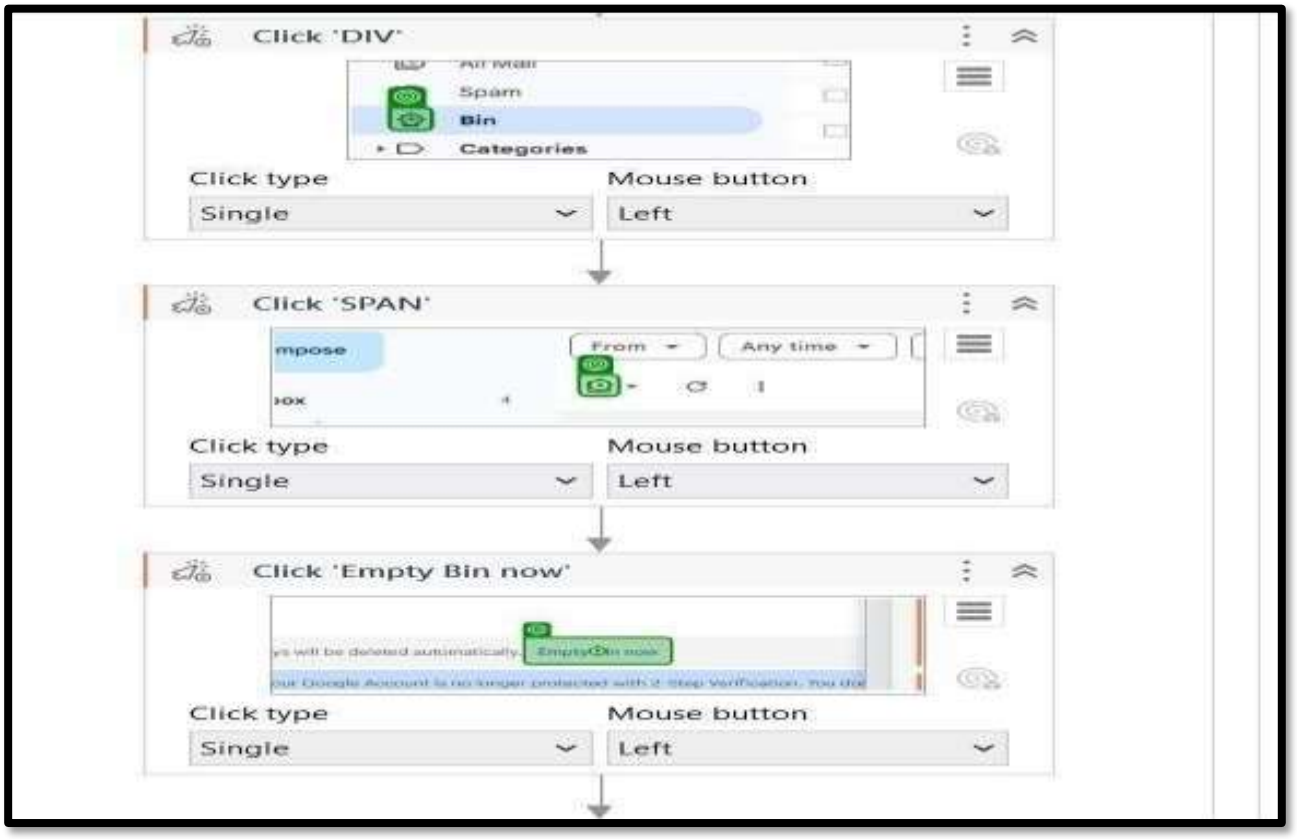

For Each
In *
Names
Body
If
Condition *
currentItem.ToString.ToLower.StartsWith("
Then
Assign
Save to
count
Value to save
count+1
Else
Drop activity here or generate with Autopilot

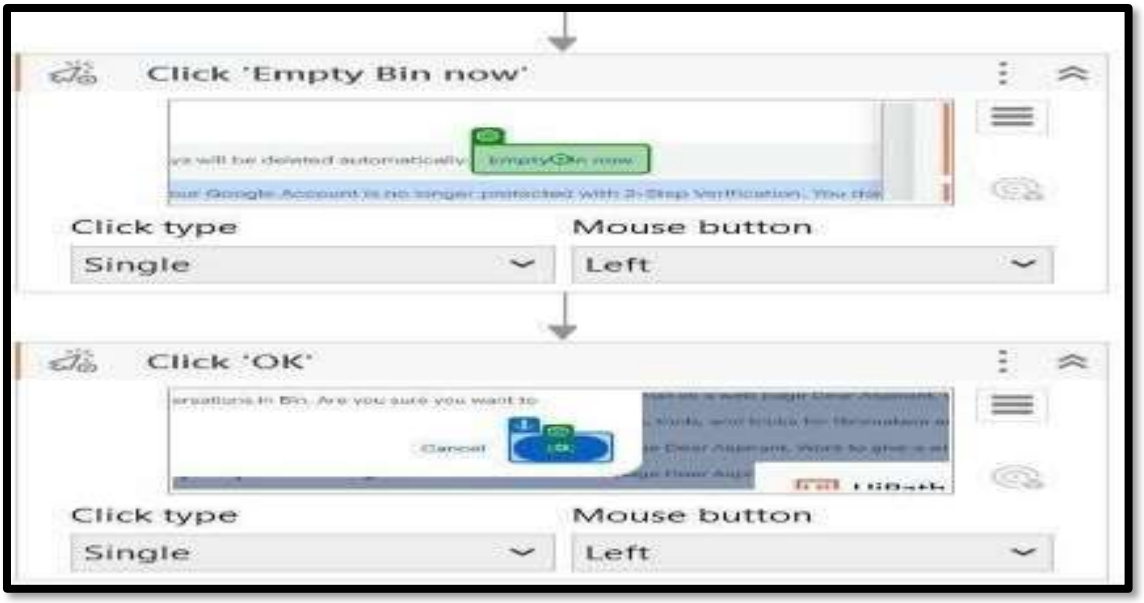

Message Box
Text *
"Count the names starting with a:"+count.T

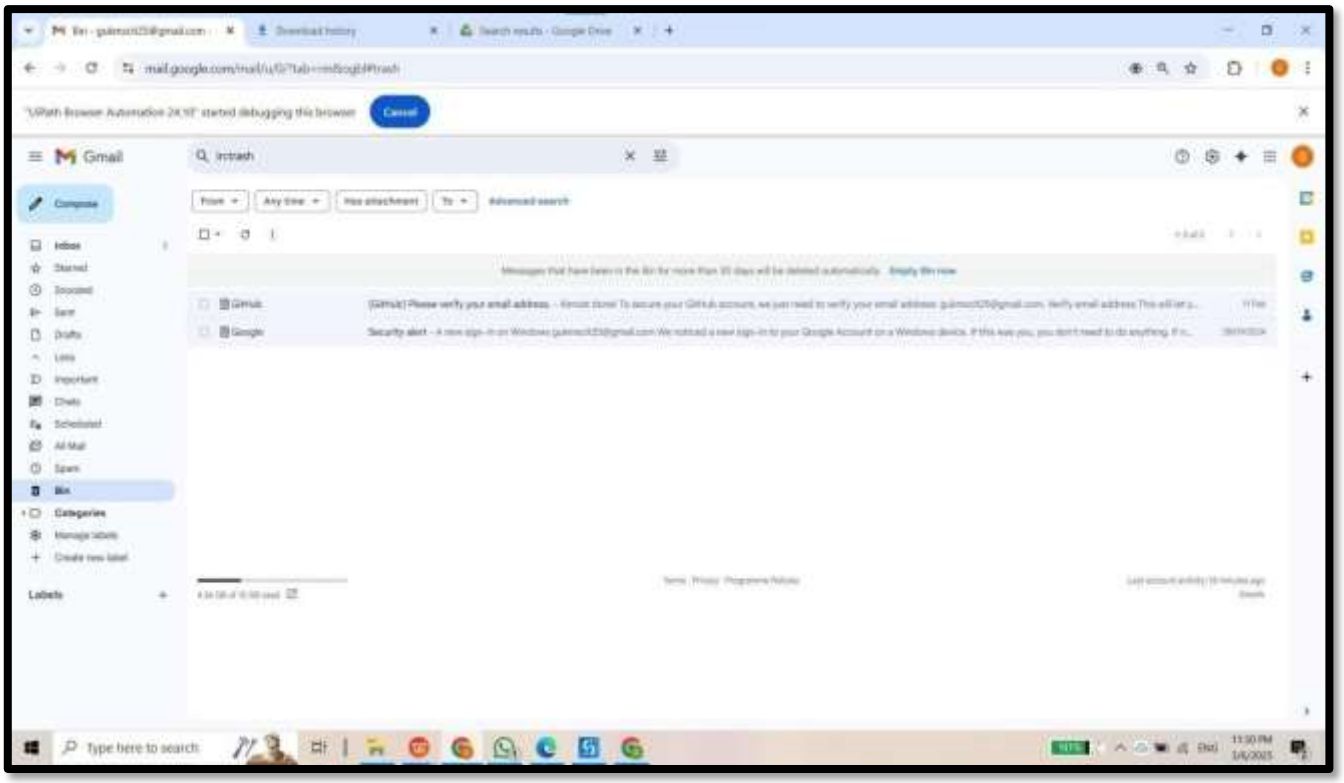
d) Create an UiPath Robot which can empty a folder in Gmail solely on basis of recording . Automate any process using basic record.

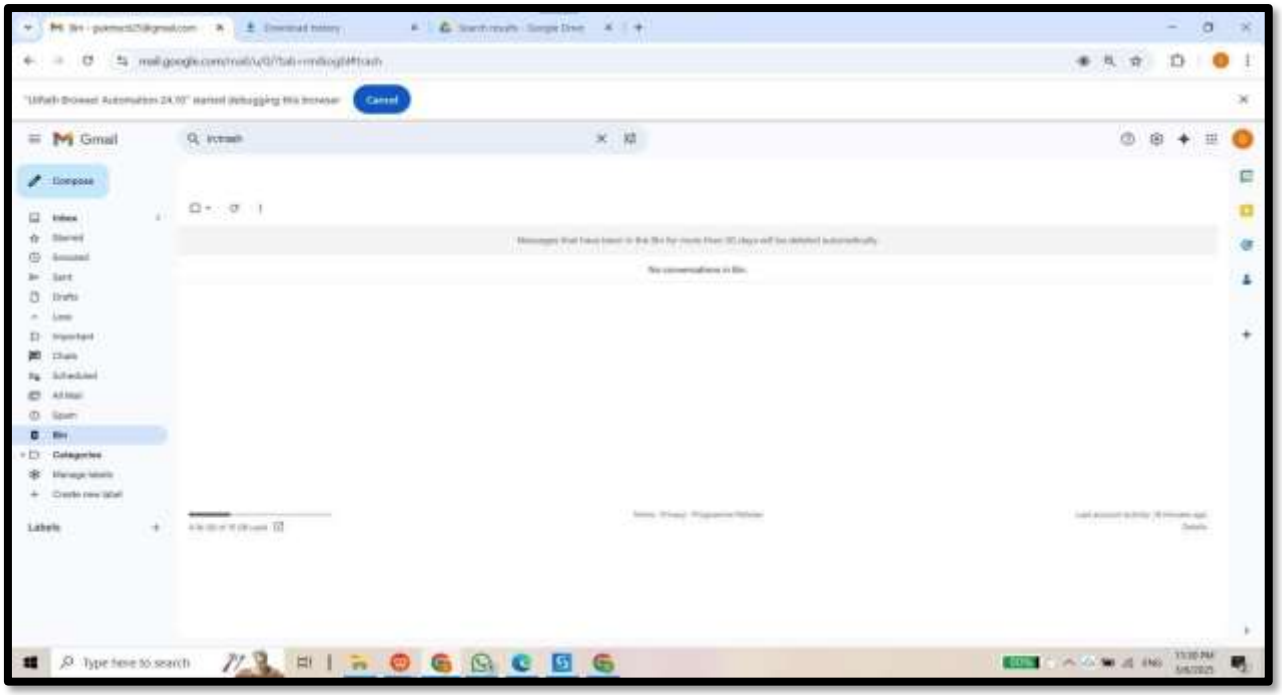



Use Application: Bin - gukmscit25@gmail.com - Gmail - Google Chrome
Application path
"C:\Program Files\Google\Chrome\Application\chrom
Application arguments
Text must be quoted
Match exact title: Bin - gukmscit25@gmail.com - Gmail - Google Chr
Do
Click 'Close'
Click type
Single
Mouse button
Left

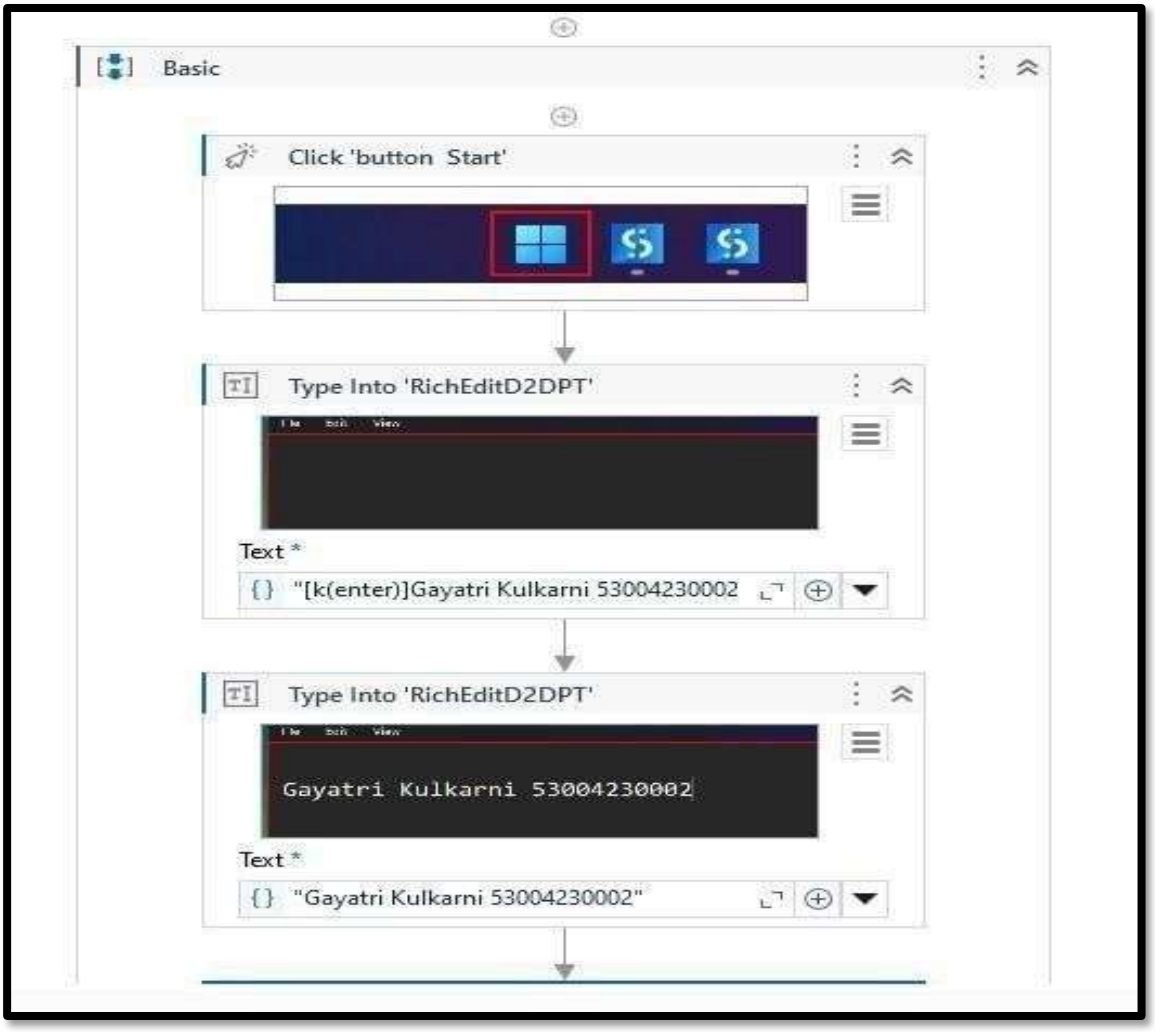

Click 'DIV'
Spam
Bin
Categories
Click type
Single
Mouse button
Left
Click 'SPAN'
From
Any time
Click type
Single
Mouse button
Left
Click 'Empty Bin now'
Click type
Single
Mouse button
Left

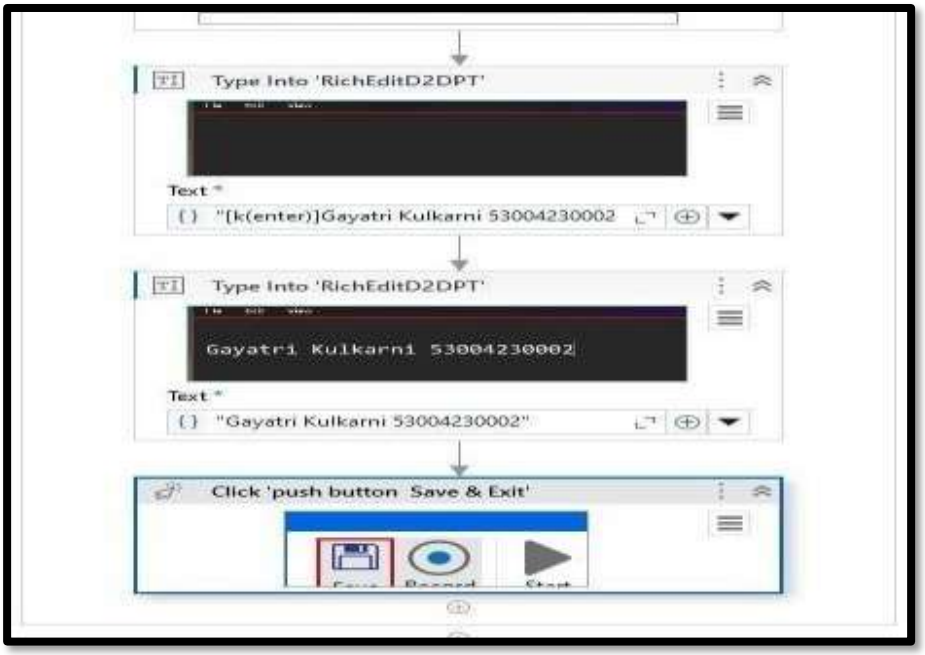

Click 'Empty Bin now'
Click type
Single
Mouse button
Left
Click 'OK'
Click type
Single
Mouse button
Left

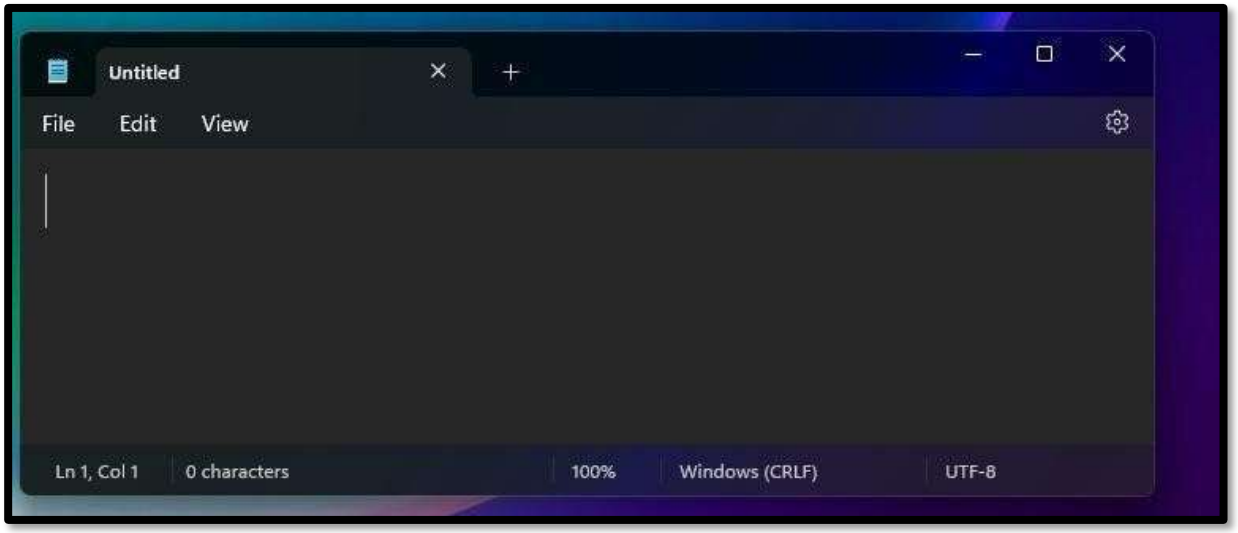

Gmail
Inbox
Starred
Snoozed
Sent
Drafts
Important
Chats
Scheduled
All Mail
Spam
Bin
Categories
Labels
Type here to search

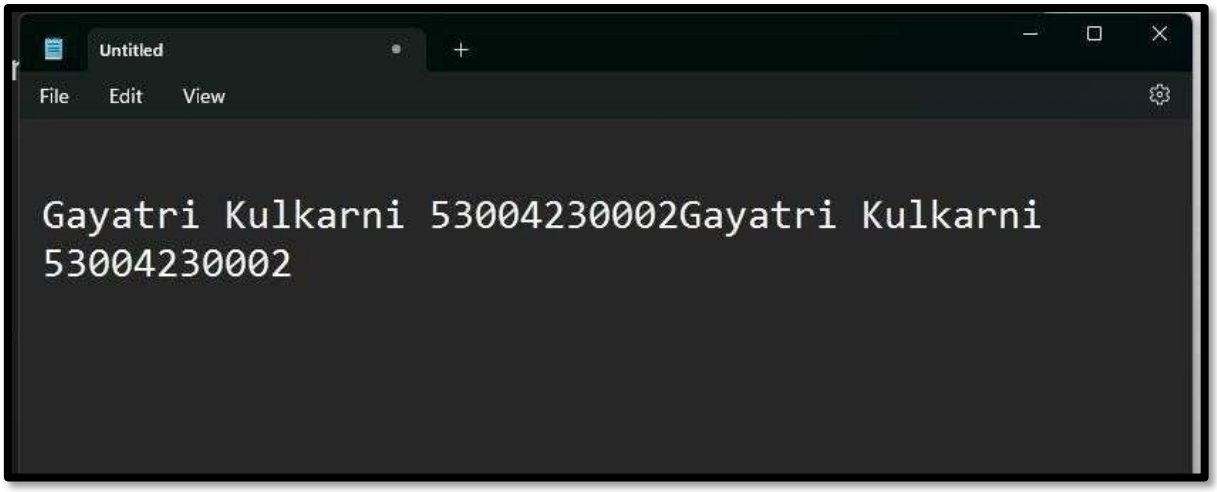


e) Automate any process using basic recording.

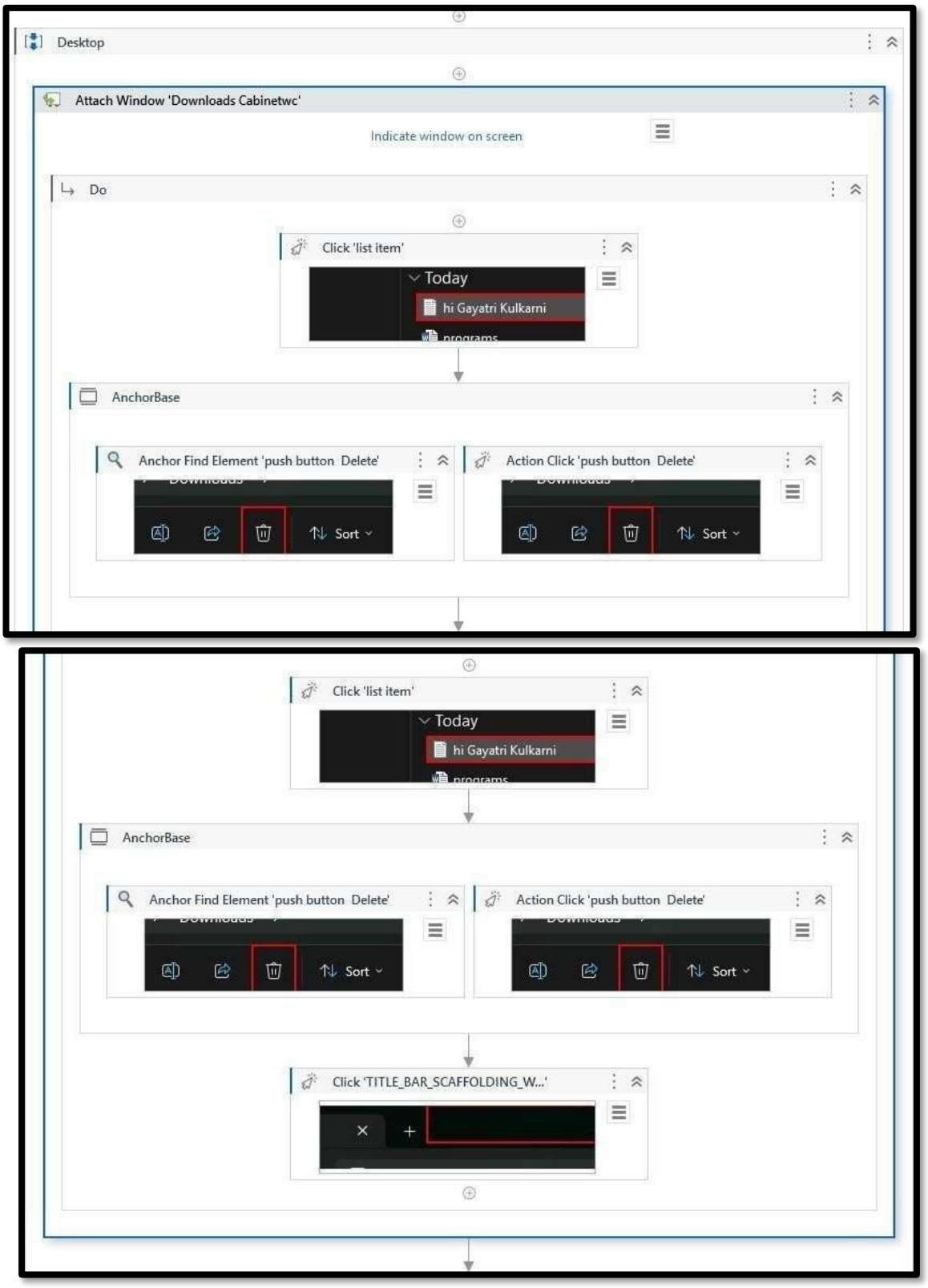


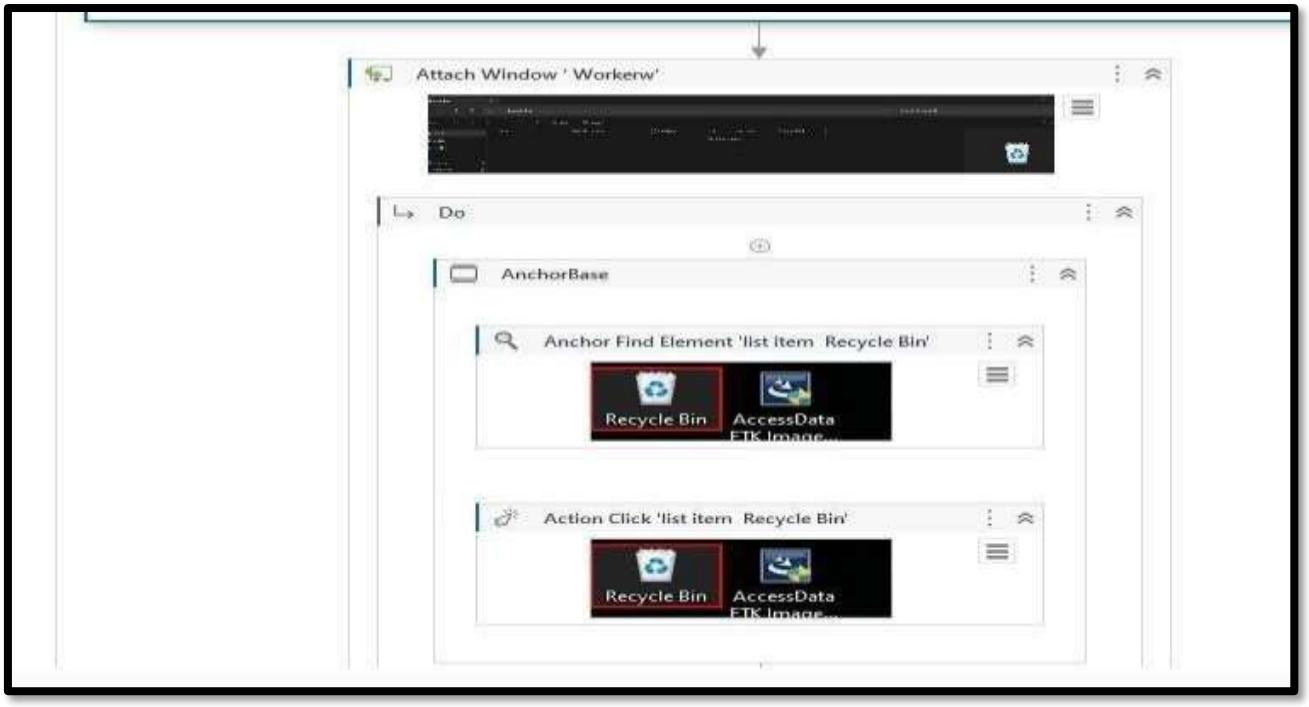

Type Into 'RichEditD2DPT'
Text *
{} "[k(enter)]Gayatri Kulkarni 53004230002
Type Into 'RichEditD2DPT'
Gayatri Kulkarni 53004230002
Text *
{} "Gayatri Kulkarni 53004230002"
Click 'push button Save & Exit'


Untitled
File
Edit
View
Ln 1, Col 1
0 characters
100%
Windows (CRLF)
UTF-8

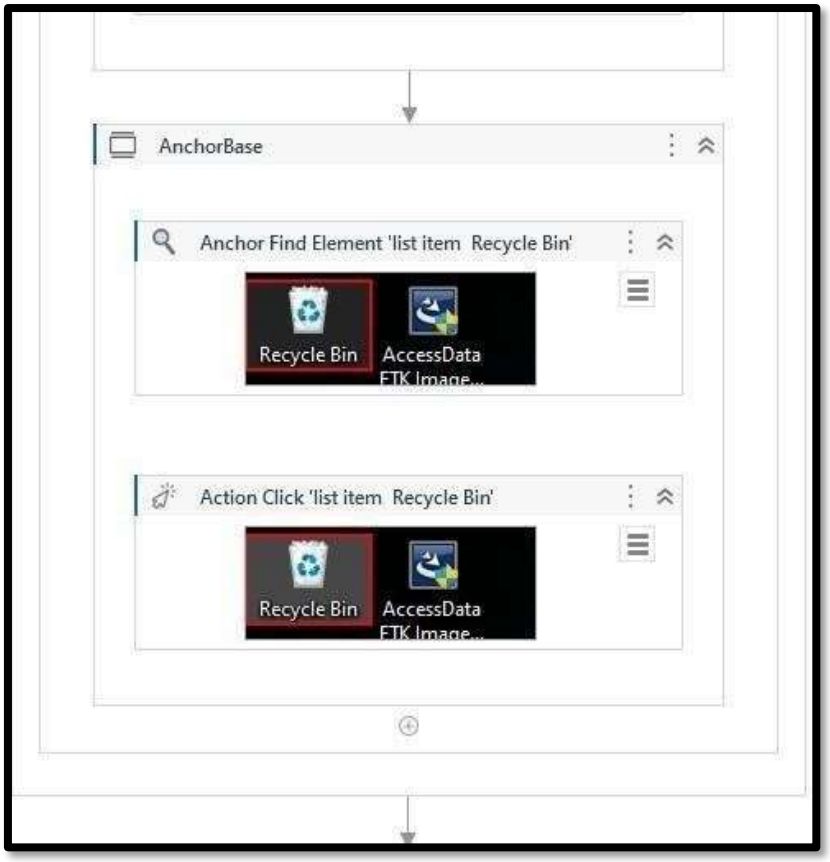

Untitled
File
Edit
View
Gayatri Kulkarni 53004230002Gayatri Kulkarni
53004230002

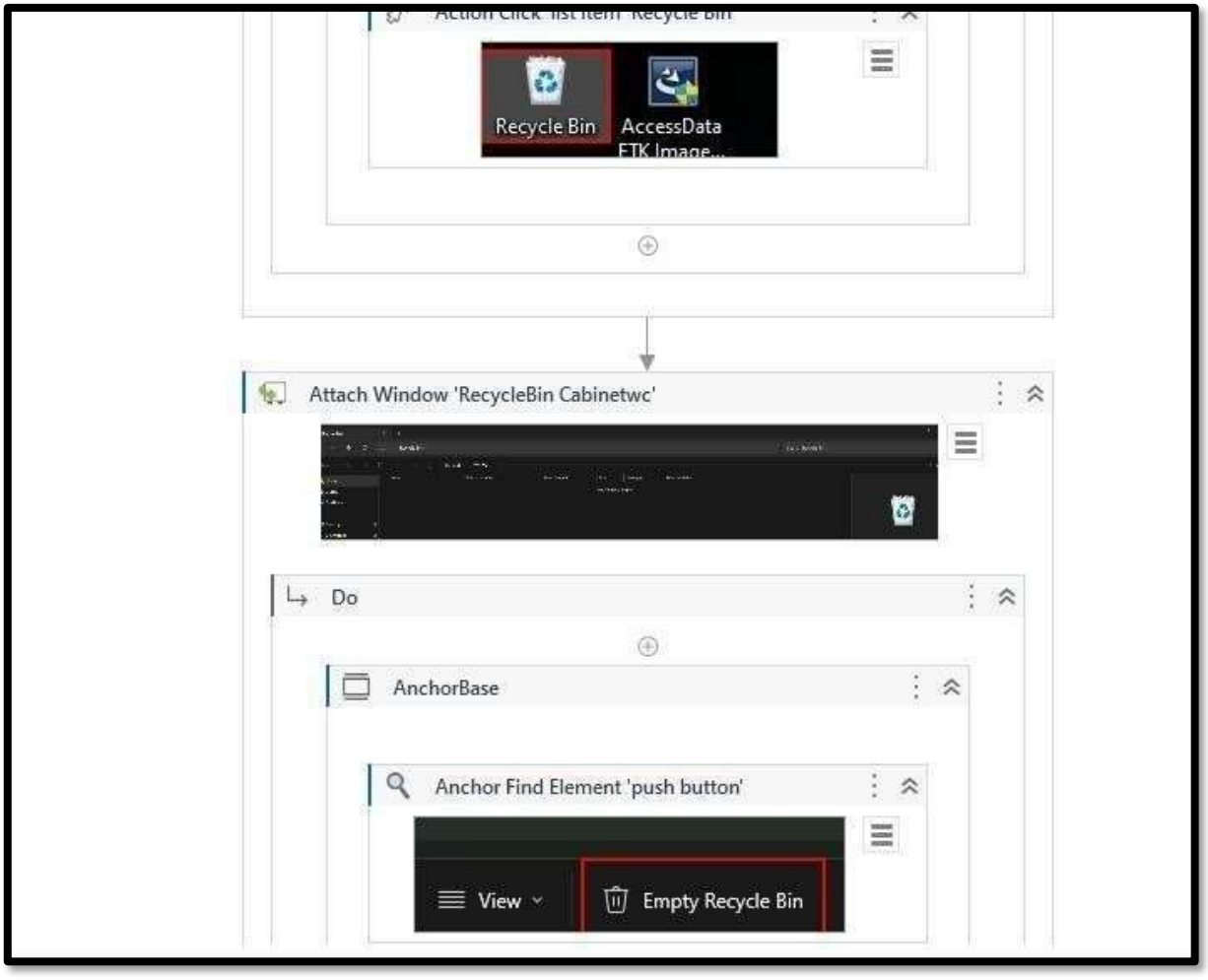
f) Automate any process using desktop recording.

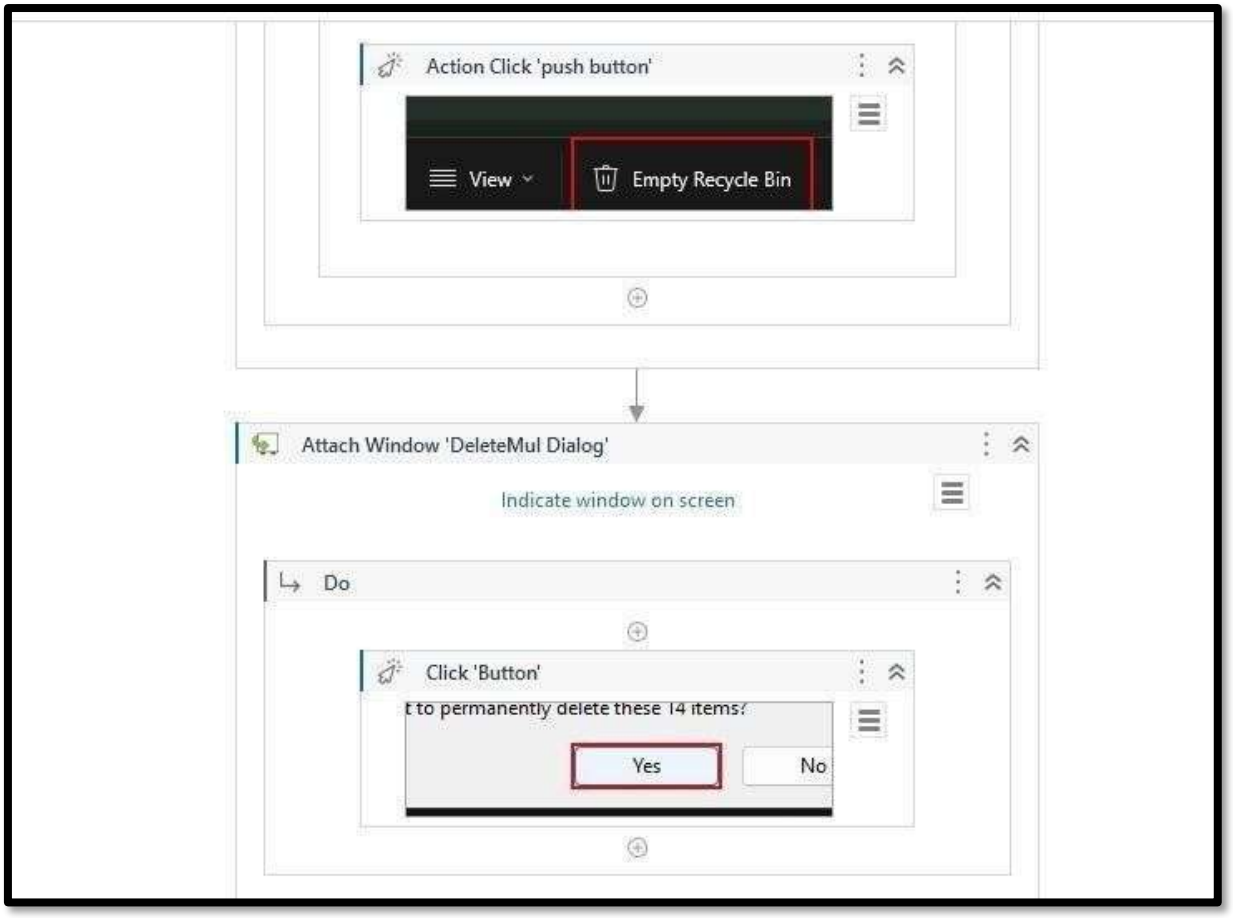



Attach Window ' Workerw'
Do
AnchorBase
Anchor Find Element 'list item Recycle Bin'
Recycle Bin
AccessData
Action Click 'list item Recycle Bin'
Recycle Bin
AccessData

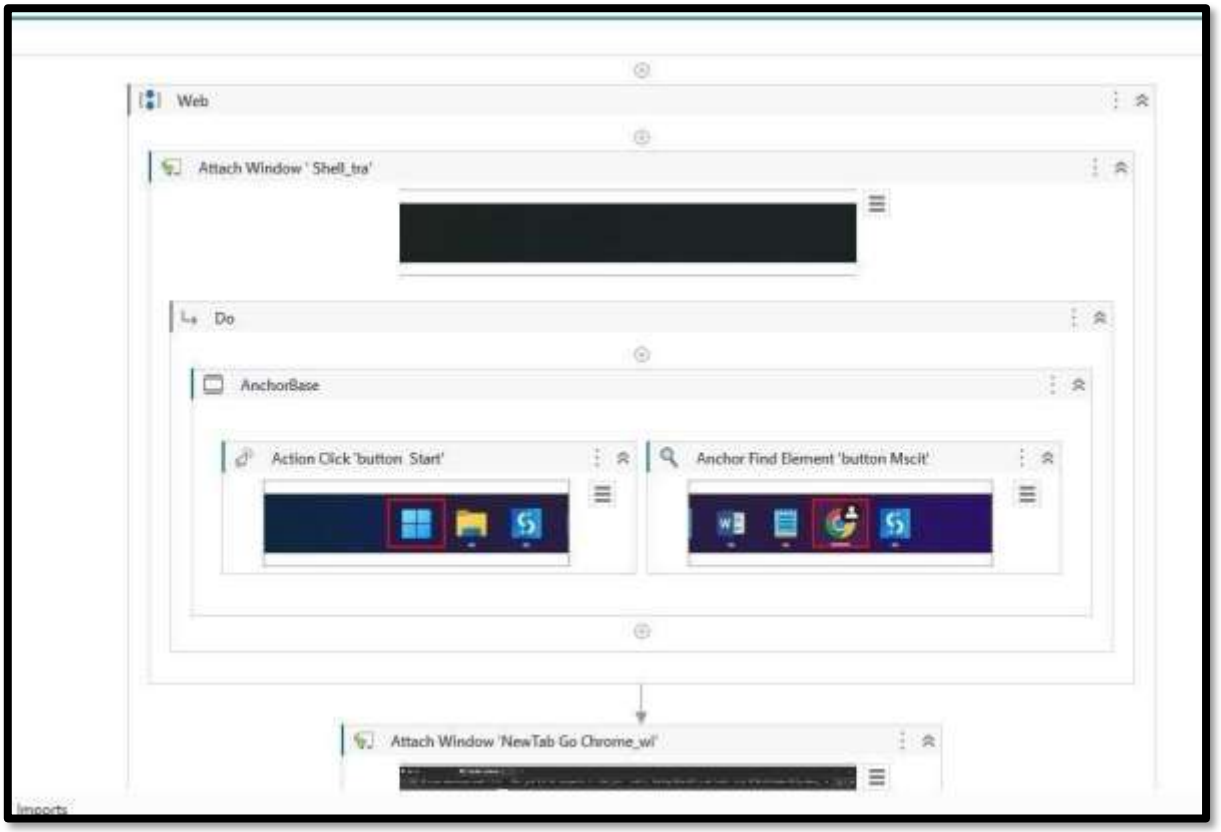

AnchorBase
Anchor Find Element 'list item Recycle Bin'
Recycle Bin
AccessData
Action Click 'list item Recycle Bin'
Recycle Bin
AccessData
AnchorBase
Anchor Find Element 'list item Recycle Bin'

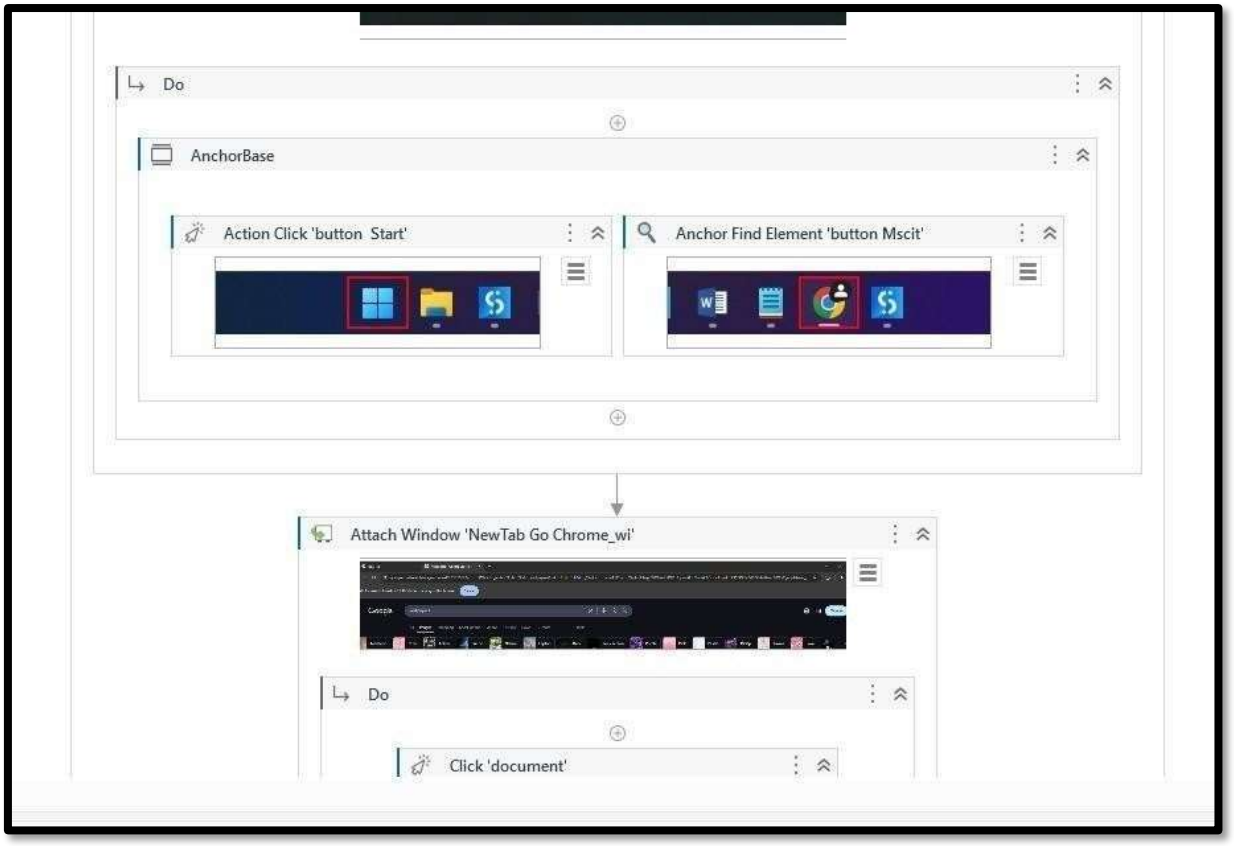

AnchorBase
Anchor Find Element 'list item Recycle Bin'
Recycle Bin
AccessData FTK Image...
Action Click 'list item Recycle Bin'
Recycle Bin
AccessData FTK Image...

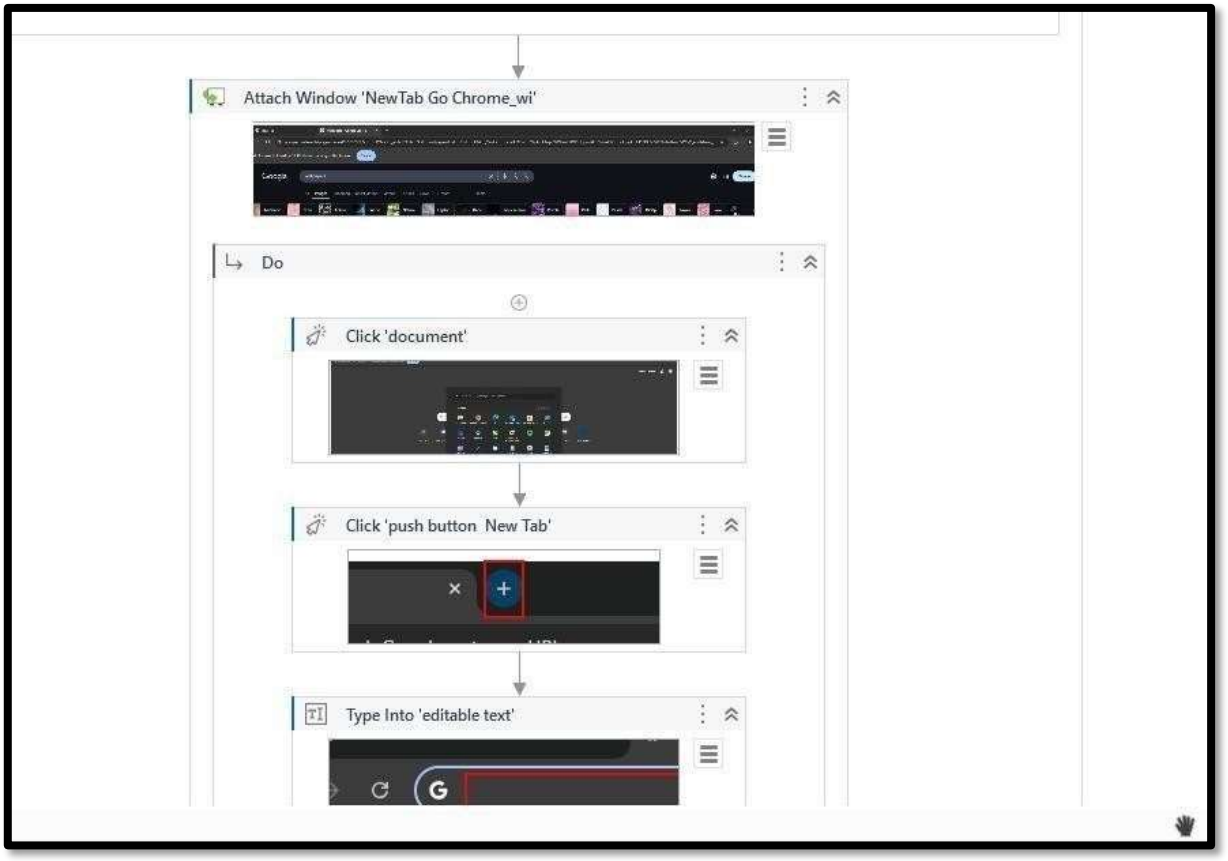

Recycle Bin
AccessData FTK Image...
Attach Window 'RecycleBin Cabinetwc'
Do
AnchorBase
Anchor Find Element 'push button'
View
Empty Recycle Bin

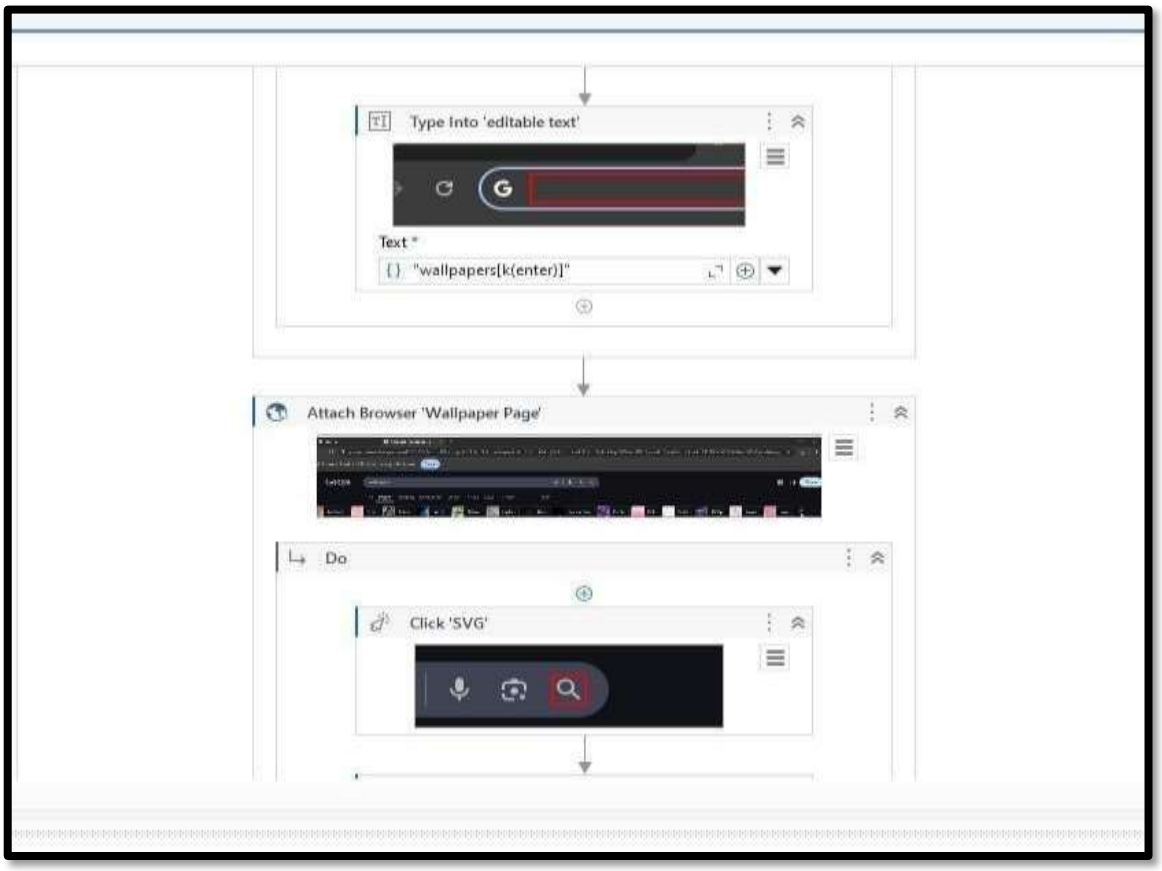


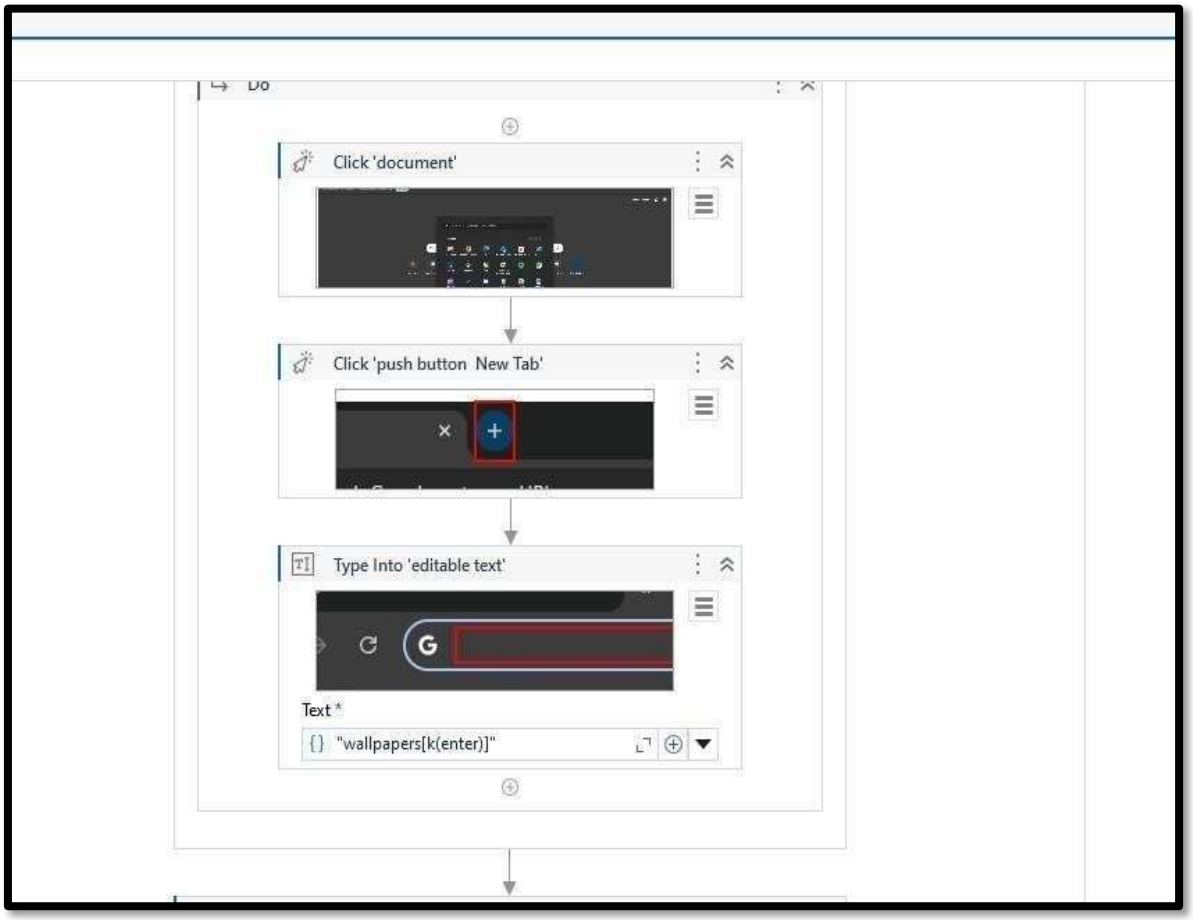
g) Automate any process using web recording:

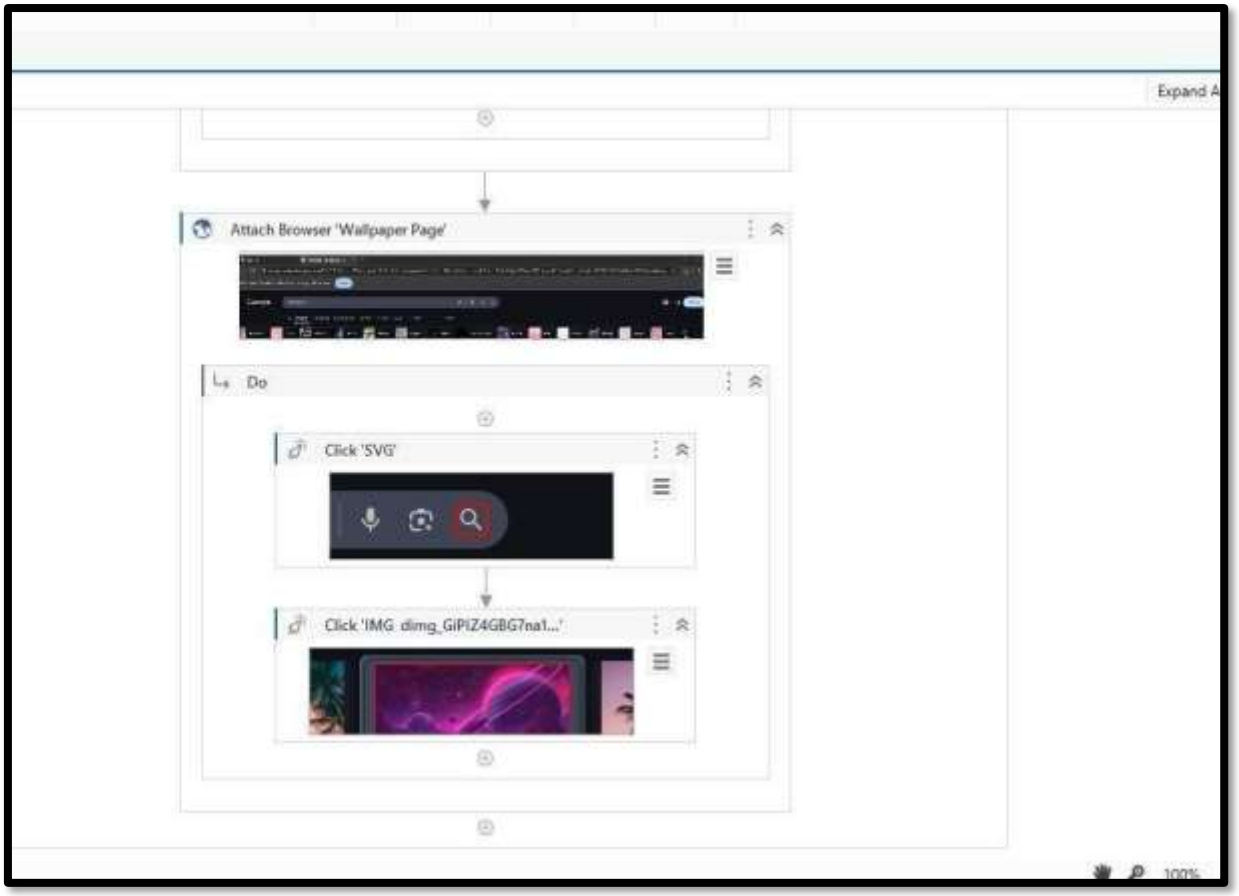


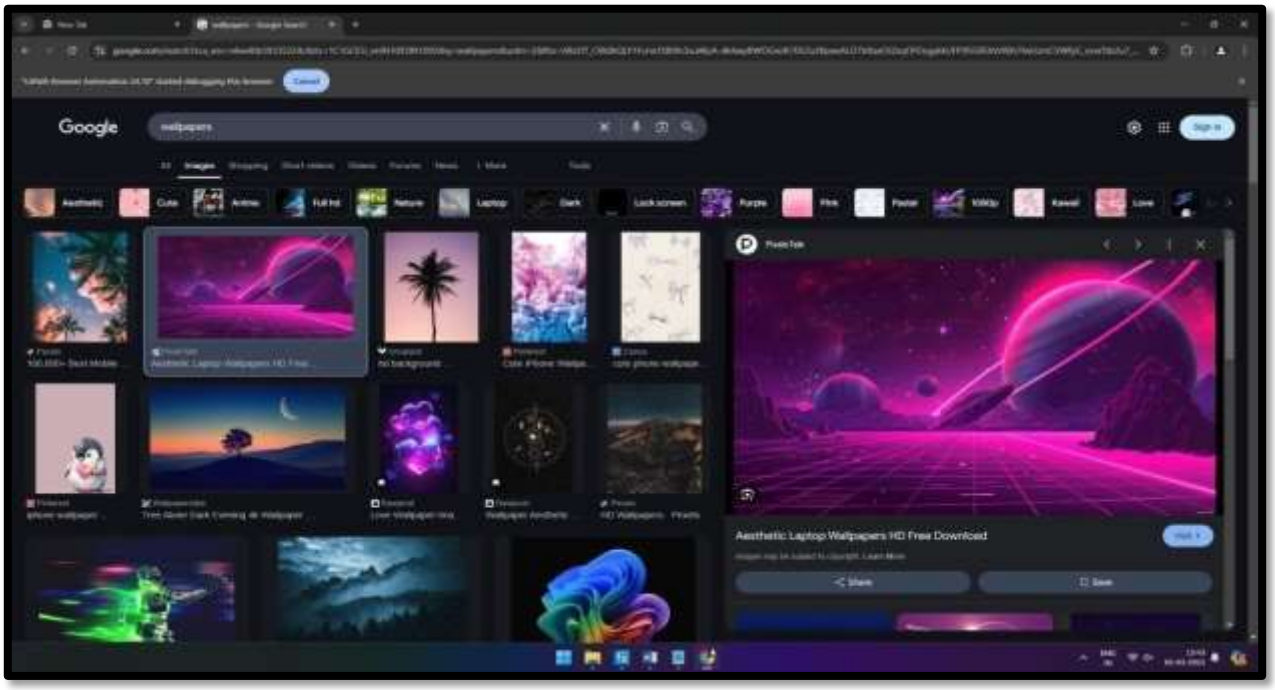

Do
AnchorBase
Action Click 'button Start'
Anchor Find Element 'button Mscit'
Attach Window 'NewTab Go Chrome_wi'
Do
Click 'document'


Attach Window 'NewTab Go Chrome_wi'
Do
Click 'document'
Click 'push button New Tab'
Type Into 'editable text'


Type Into 'editable text'
Text *
{} "wallpapers[k(enter)]"
Attach Browser 'Wallpaper Page'
Do
Click 'SVG'


Do
Click 'document'
Click 'push button New Tab'
Type Into 'editable text'
Text *
{} "wallpapers[k(enter)]"

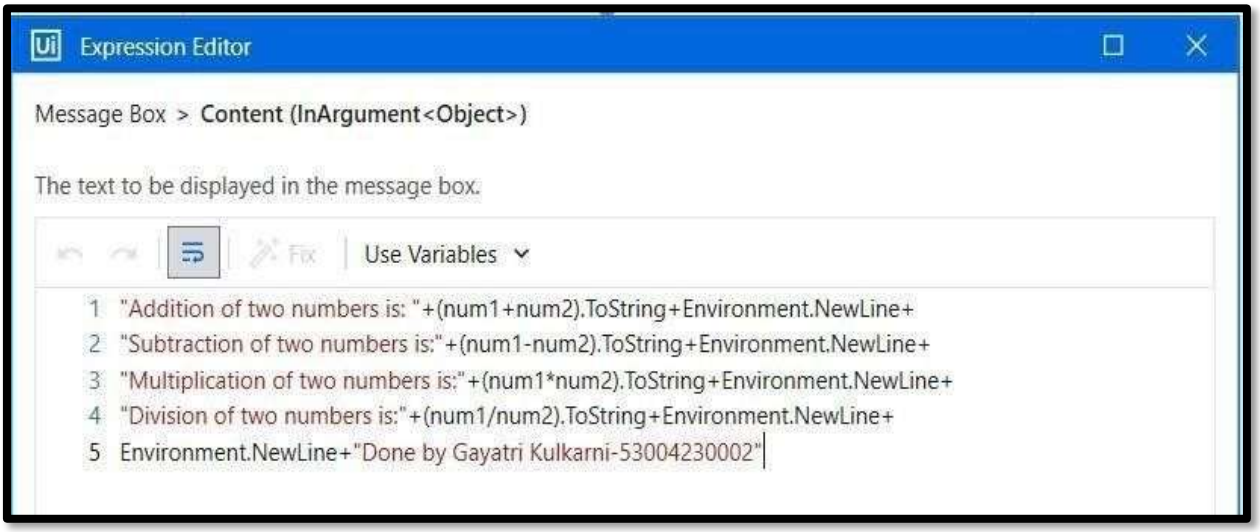

Expand A
Attach Browser 'Wallpaper Page'
Do
Click 'SVG'
Click 'IMG dimg_GiPiZ4GBG7na1...'

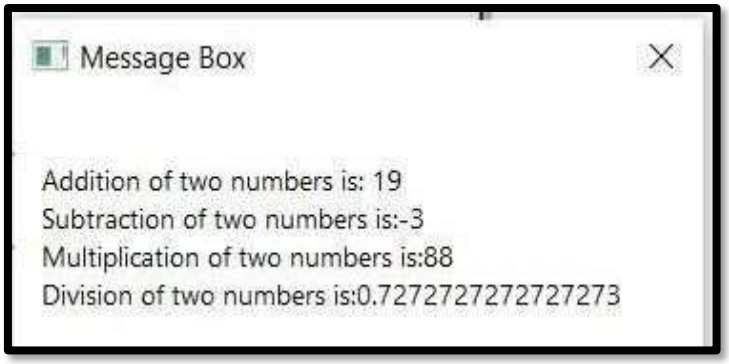

Google
wallpapers
Aesthetic: Laptop Wallpapers HD Free Download

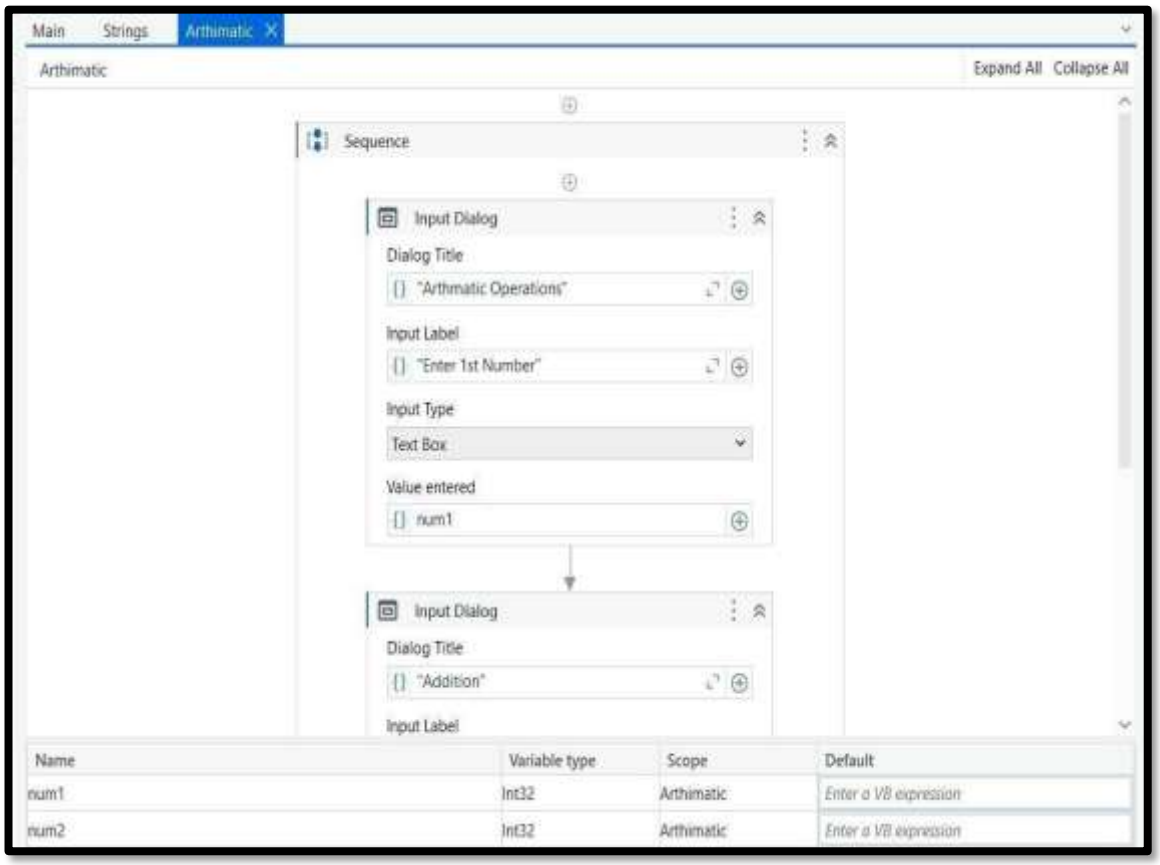
Module 2

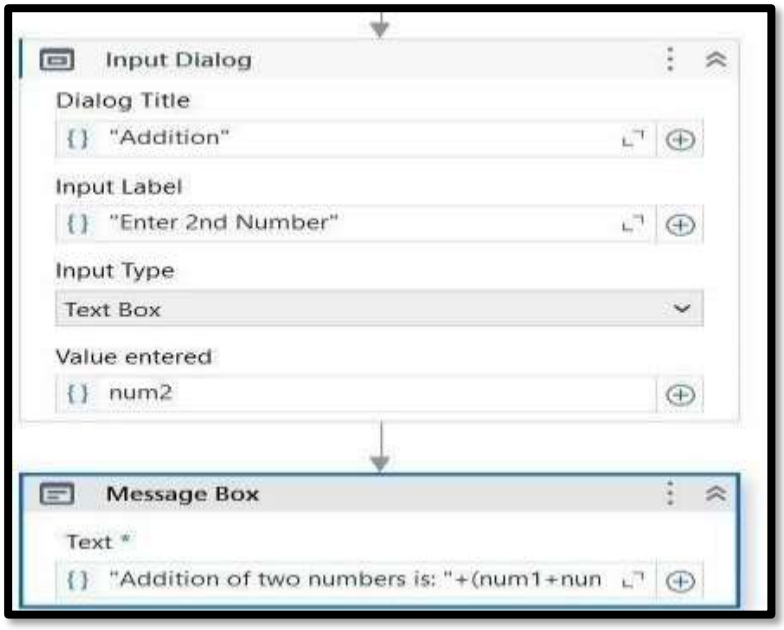
Data Manipulation: Variables and scope, Collections, Arguments –Purpose and use, Data table usage with examples, Clipboard management, File operation with step-by-step example, CSV/Excel to data table and vice versa (with a step-by-step example)

a) Automate UiPath Number Calculation (Subtraction,Multiplication, Division of numbers).

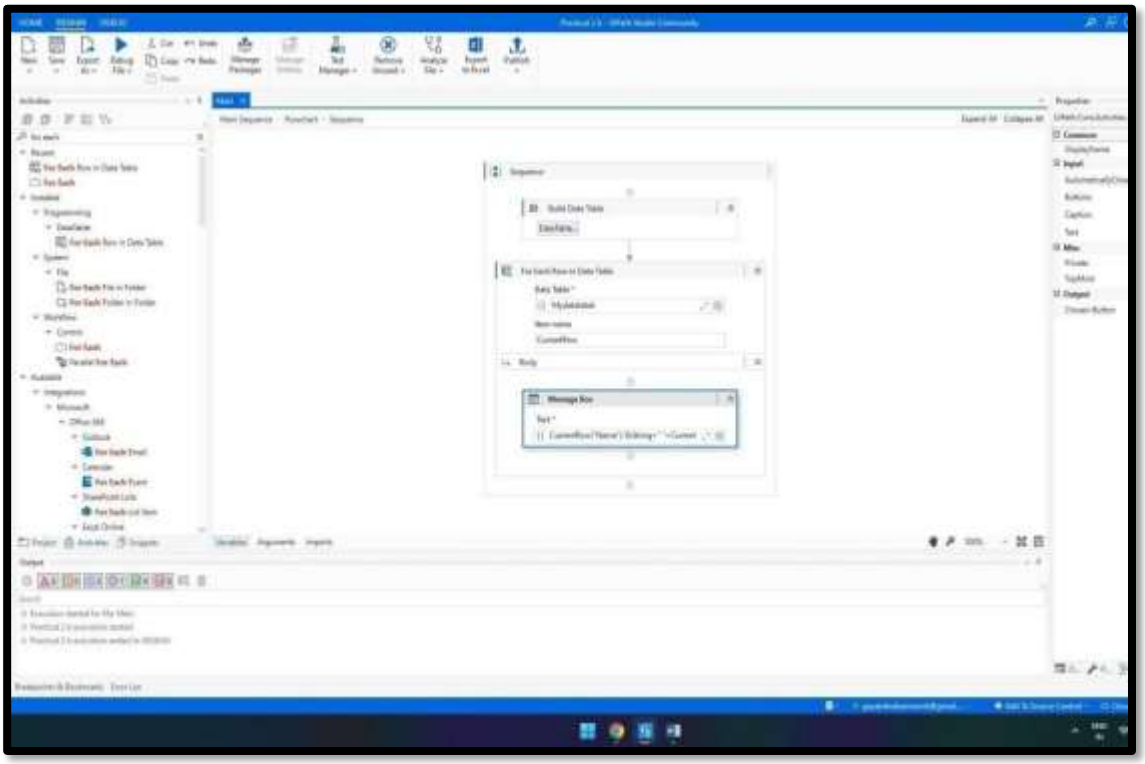


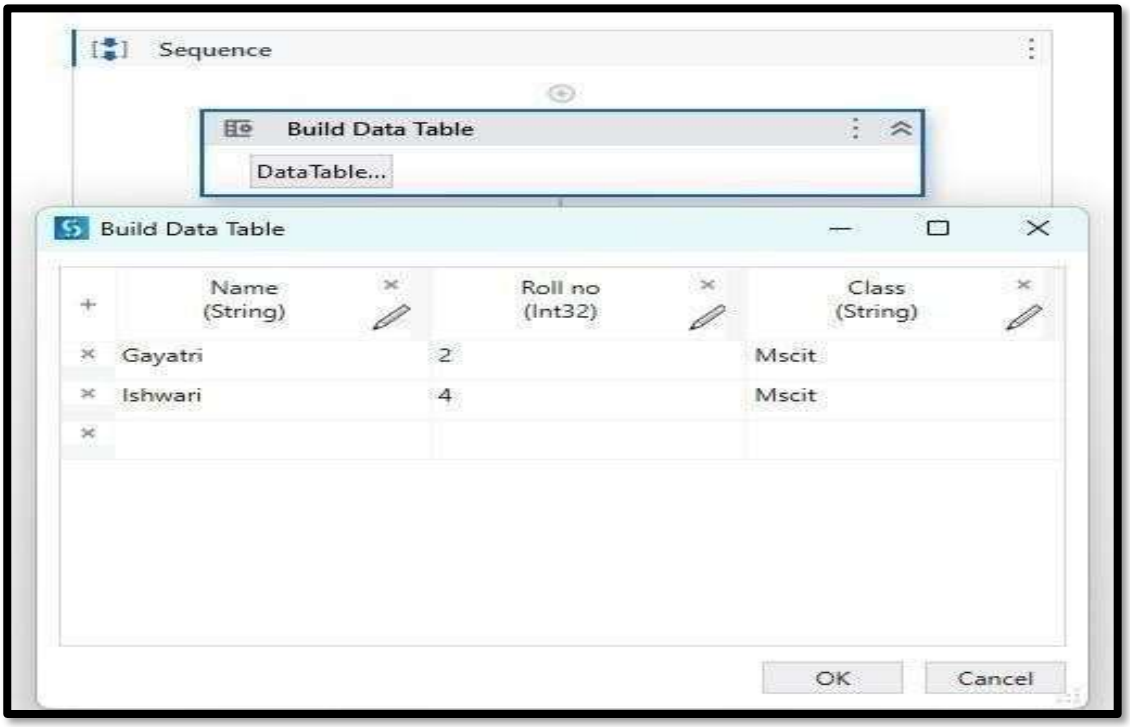

Message Box
Addition of two numbers is: 19
Subtraction of two numbers is:-3
Multiplication of two numbers is:88
Division of two numbers is:0.7272727272727273


Main
Strings
Arthimatic
Arthimatic
Expand All
Collapse All
Sequence
Input Dialog
Dialog Title
"Arthmatic Operations"
Input Label
"Enter 1st Number"
Input Type
Text Box
Value entered
num1
Input Dialog
Dialog Title
"Addition"
Input Label
Name
Variable type
Scope
Default
num1
Int32
Arthimatic
Enter a VB expression
num2
Int32
Arthimatic
Enter a VB expression

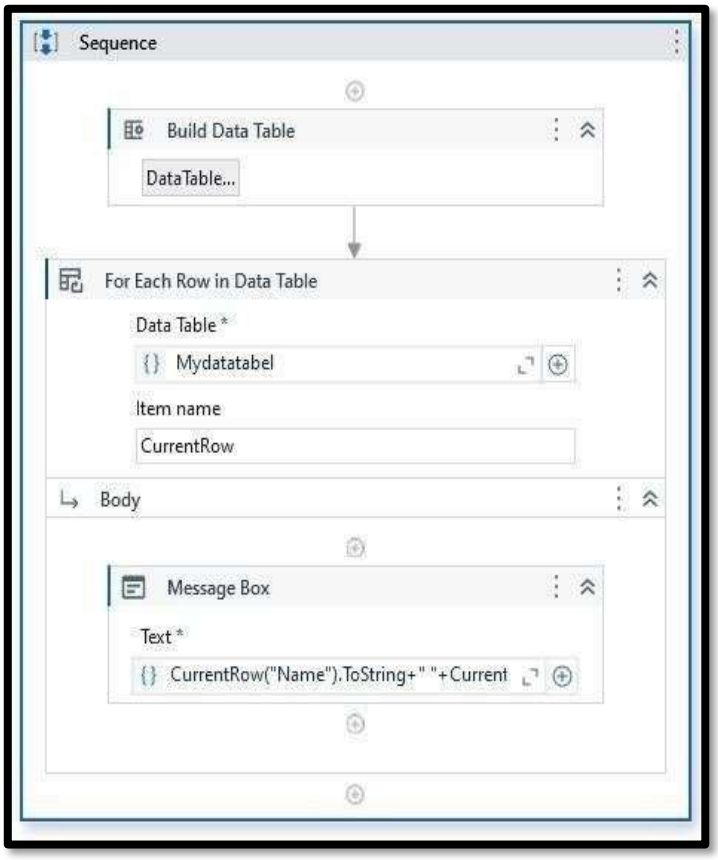

Input Dialog
Dialog Title
"Addition"
Input Label
"Enter 2nd Number"
Input Type
Text Box
Value entered
num2
Message Box
Text *
"Addition of two numbers is: "+(num1+nun

b) Create an automation UiPath project using different types of variables (number, datetime, Boolean, generic, array, data table):



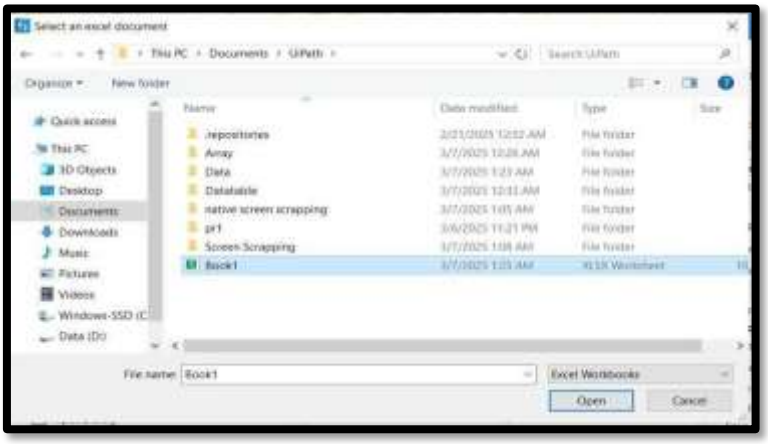


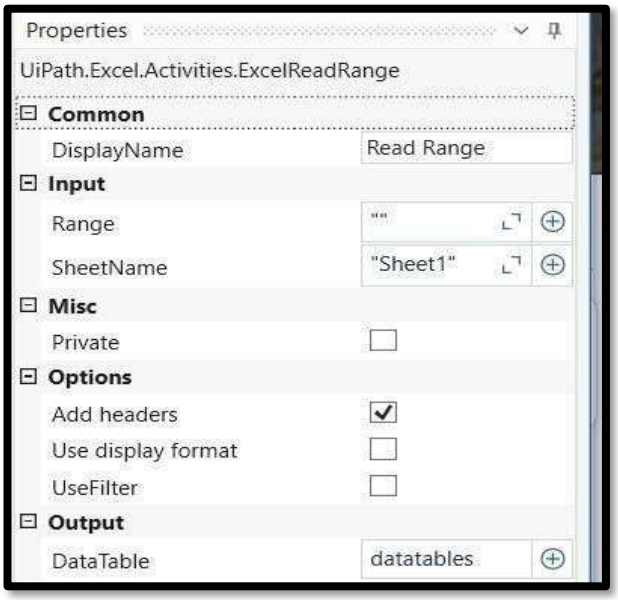
CurrentRow("Name").ToString+" "+CurrentRow("Roll no").ToString+" "+CurrentRow("Class").ToString

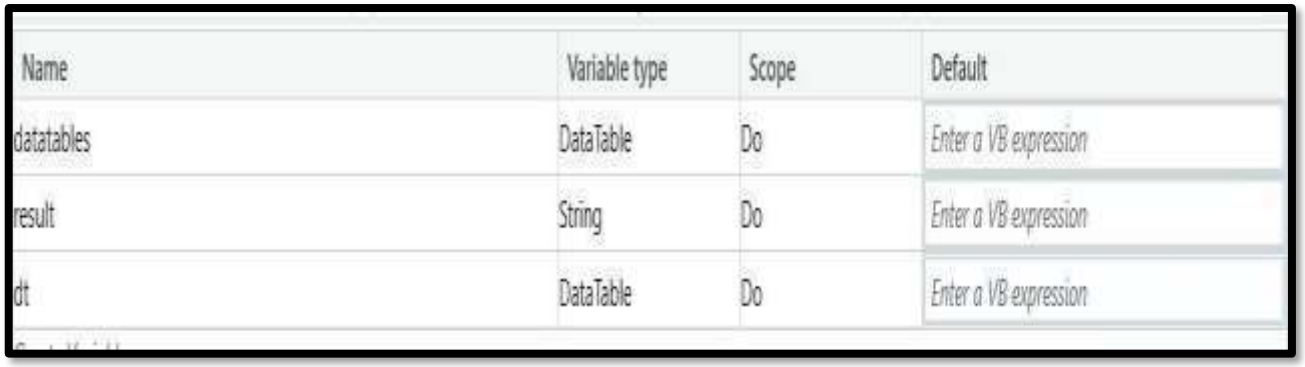


c) Create an application automating the read, write and append operation on excel file.

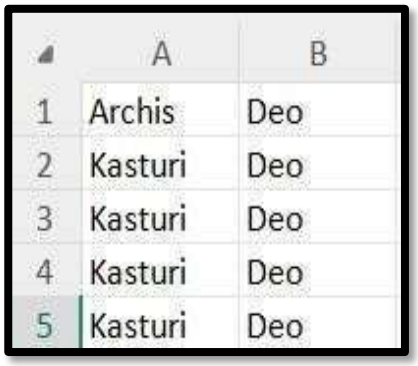
	A	B
1	Name	Surname
2	Gayatri	Kulkarni
3	Ishwari	Kulkarni
4	Archis	Deo
5	Kasturi	Deo



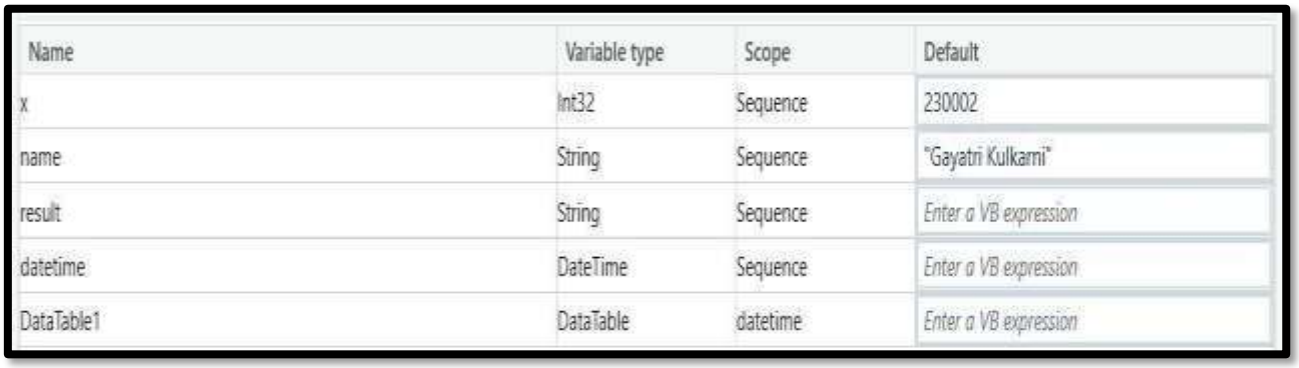




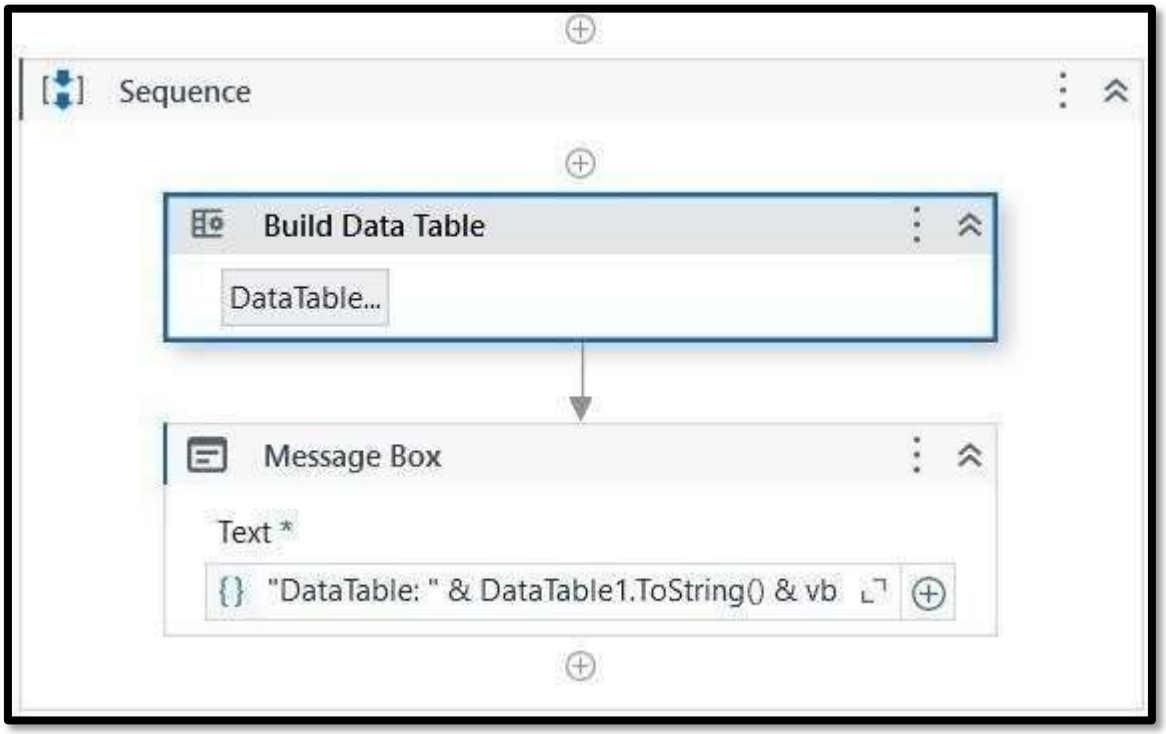
Name	Variable type	Scope	Default
datatables	DataTable	Do	Enter a VB expression
result	String	Do	Enter a VB expression
dt	DataTable	Do	Enter a VB expression

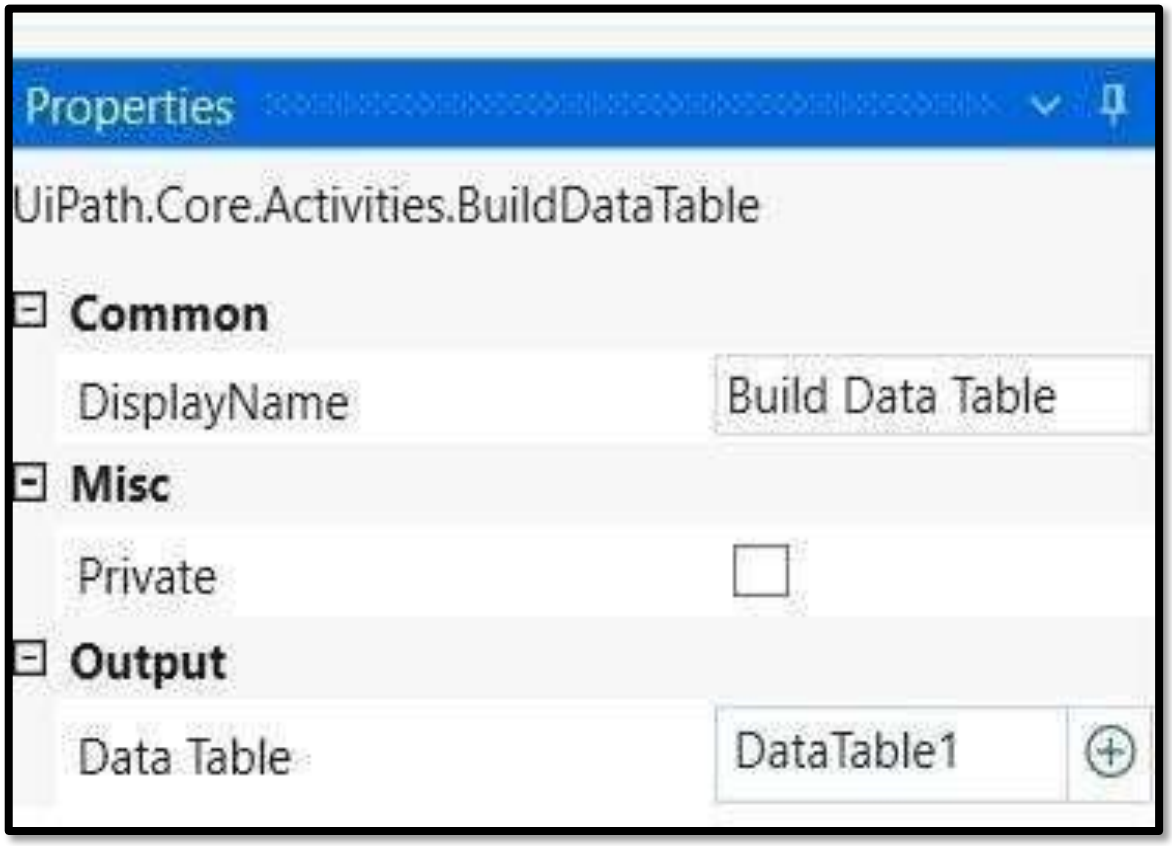


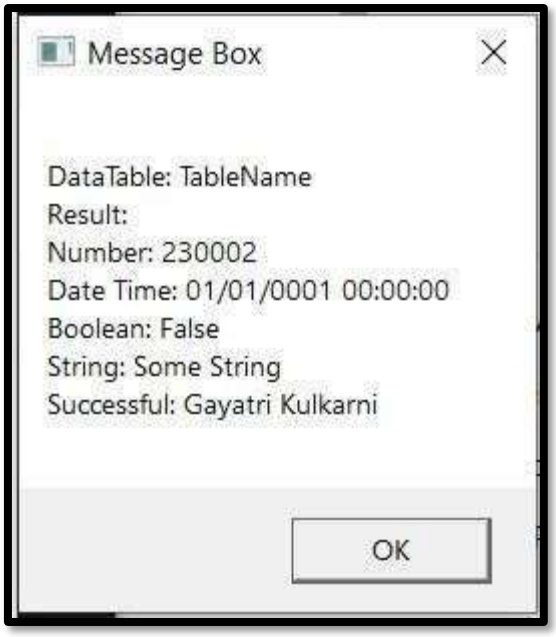
	A	B
1	Archis	Deo
2	Kasturi	Deo
3	Kasturi	Deo
4	Kasturi	Deo
5	Kasturi	Deo



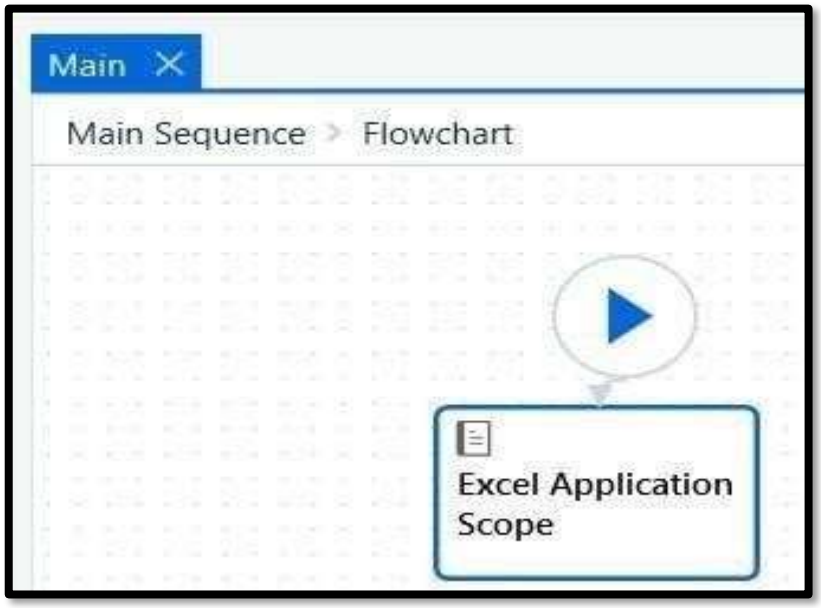
Name	Variable type	Scope	Default
x	Int32	Sequence	230002
name	String	Sequence	"Gayatri Kulkarni"
result	String	Sequence	Enter a VB expression
datetime	DateTime	Sequence	Enter a VB expression
DataTable1	DataTable	datetime	Enter a VB expression

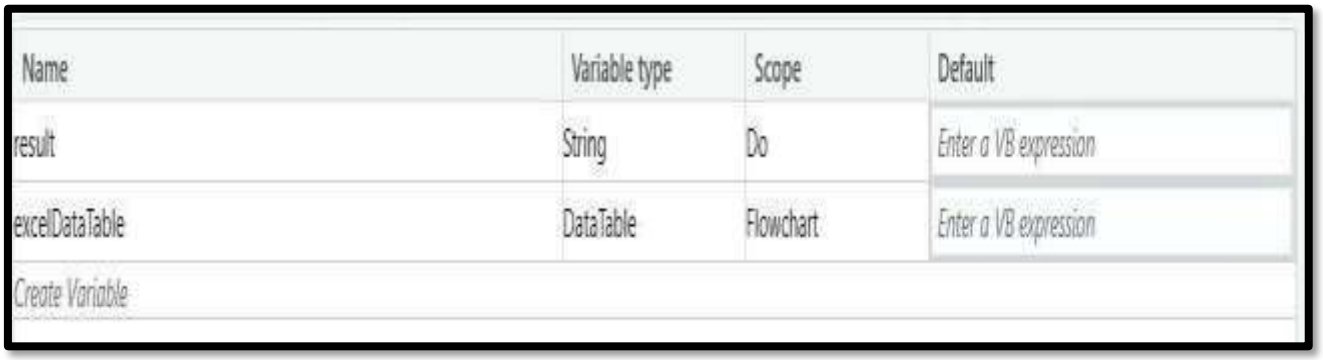

Sequence
Build Data Table
DataTable...
Message Box
Text *
{} "DataTable: " & DataTable1.ToString() & vb


Properties
UiPath.Core.Activities.BuildDataTable
Common
DisplayName
Build Data Table
Misc
Private
Output
Data Table
DataTable1



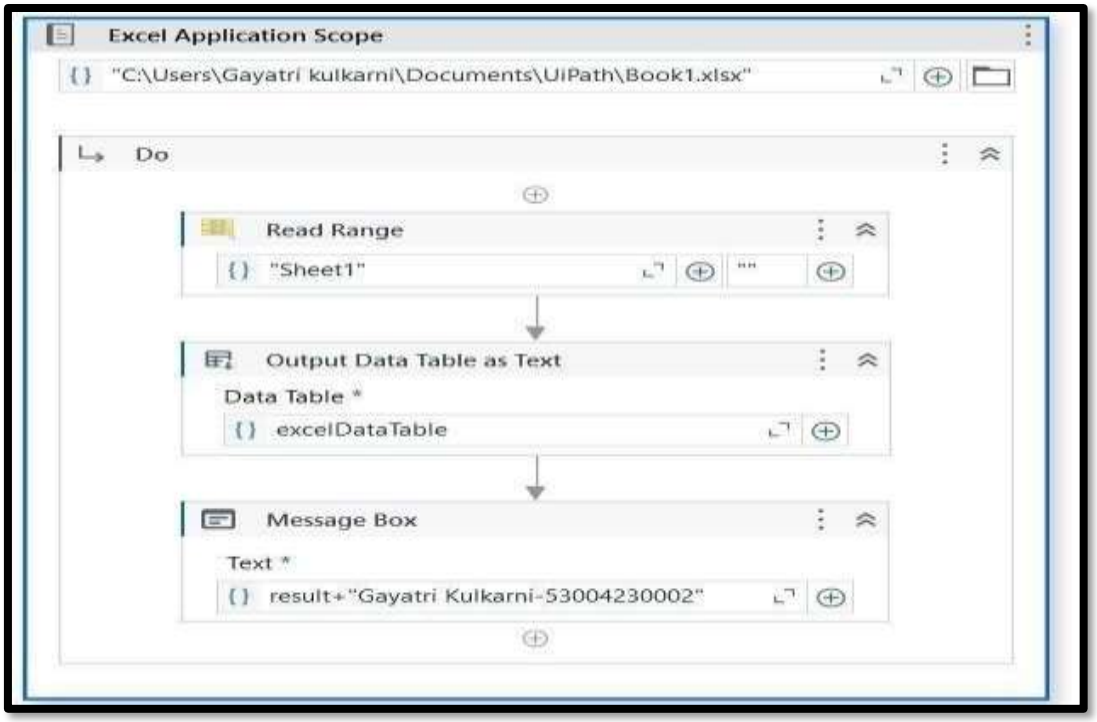
d) Automate the process to extract data from an excel file into a data table and vice versa:





Name	Variable type	Scope	Default
result	String	Do	Enter a VB expression
excelDataTable	DataTable	Flowchart	Enter a VB expression
Create Variable			


Properties
UiPath.Excel.Activities.ExcelReadRange
Common
DisplayName
Read Range
Input
Range
""
SheetName
"Sheet1"
Misc
Private
Options
Add headers
Use display format
UseFilter
Output
DataTable
excelDataTable


Excel Application Scope
"C:\Users\Gayatri kulkarni\Documents\UiPath\Book1.xlsx"
Do
Read Range
"Sheet1"
""
Output Data Table as Text
Data Table *
excelDataTable
Message Box
Text *
result+"Gayatri Kulkarni-53004230002"

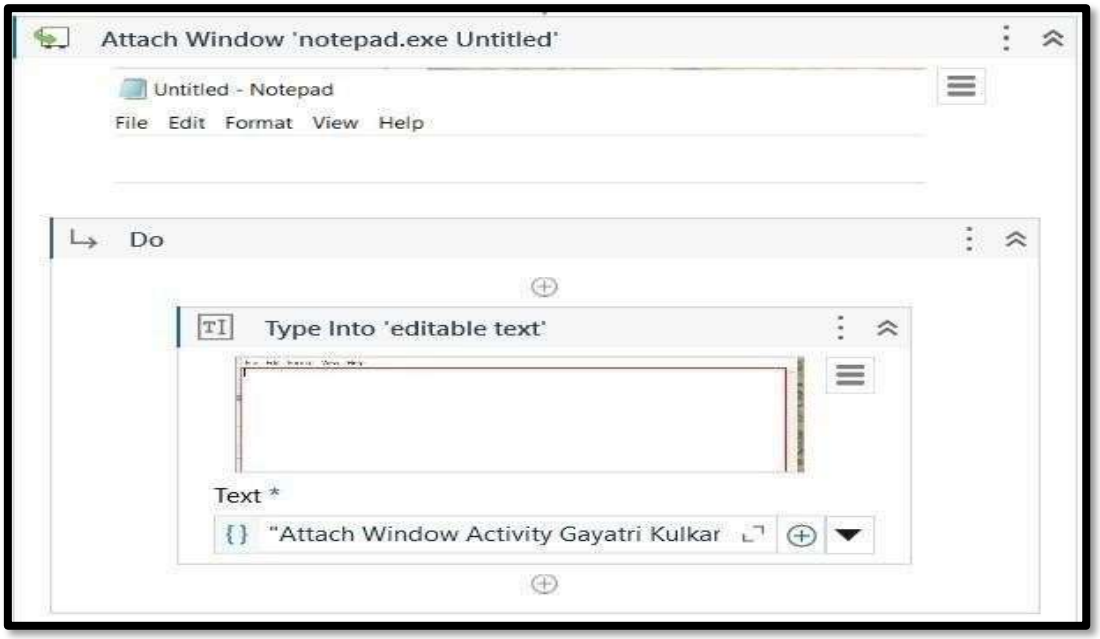

Expression Editor
Message Box > Content (InArgument<Object>)
The text to be displayed in the message box.
Use Variables
1 result+"Gayatri Kulkarni-53004230002"


Message Box
Kasturi,Deo
Kasturi,Deo
Kasturi,Deo
Kasturi,Deo
Kasturi,Deo
Gayatri Kulkarni-53004230002
OK

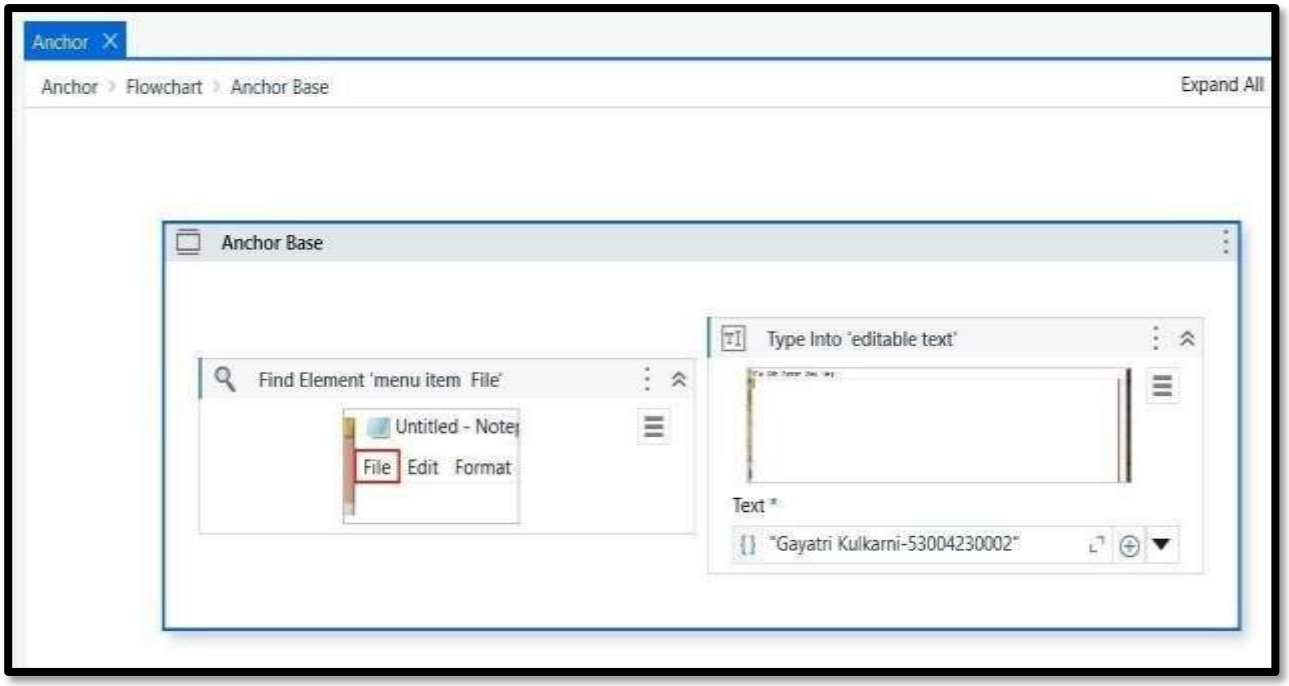
Module 3

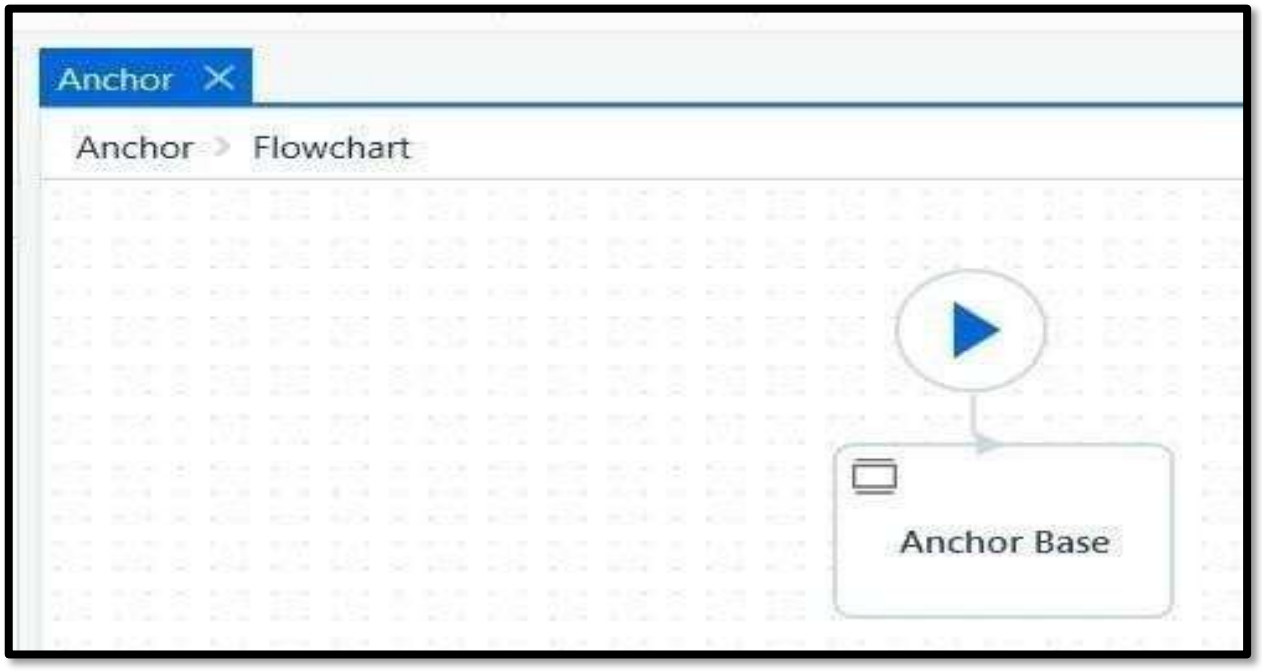
Taking Control of the Controls: Finding and attaching windows,Finding the control, Techniques for waiting for a control, Act on controls – mouse and keyboard activities, Working with UiExplorer,Handling events, Revisit recorder, Screen Scraping, When to use OCR, Types of OCR available, How to use OCR.Handling User Events and Assistant Bots: What are assistant bots?Monitoring system event triggers, Hotkey trigger, Mouse trigger, System trigger ,Monitoring image and element triggers, An example of monitoring email, Example of monitoring a copying event and blocking it, Launching an assistant bot on a keyboard event.

a) Implement the attach window activity.

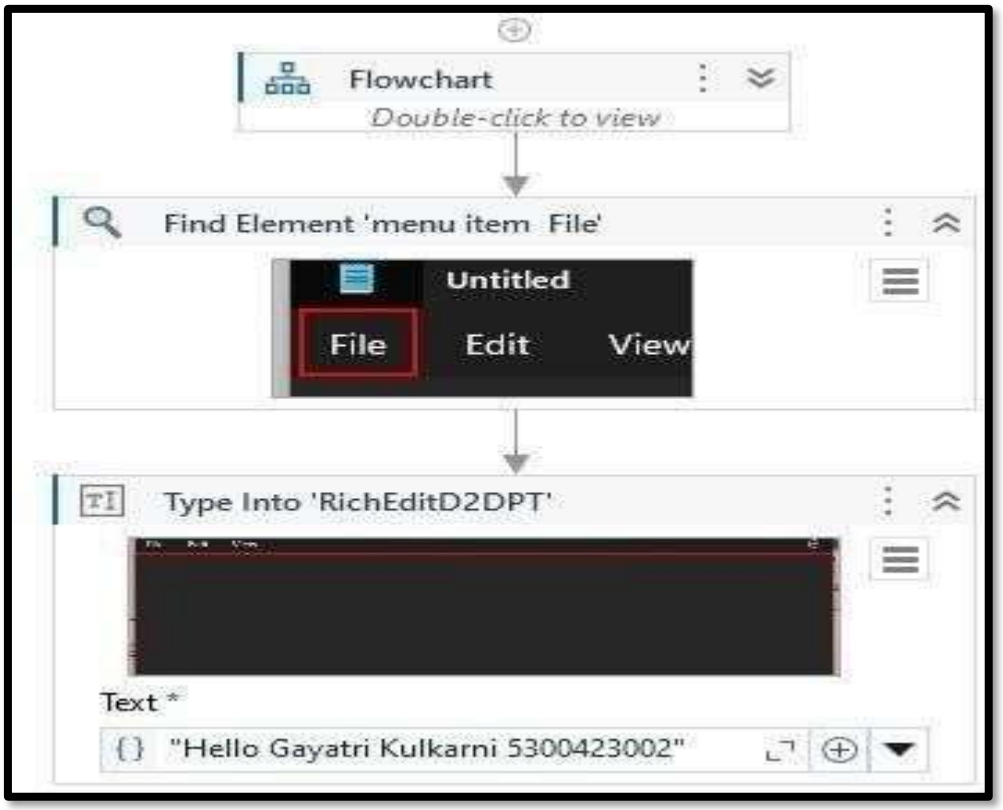


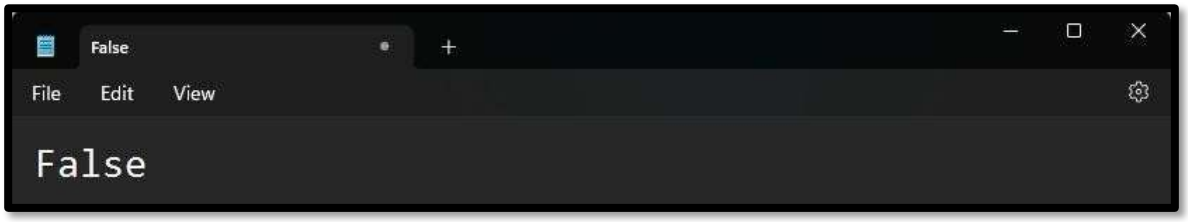
b) Find different controls using UiPath :



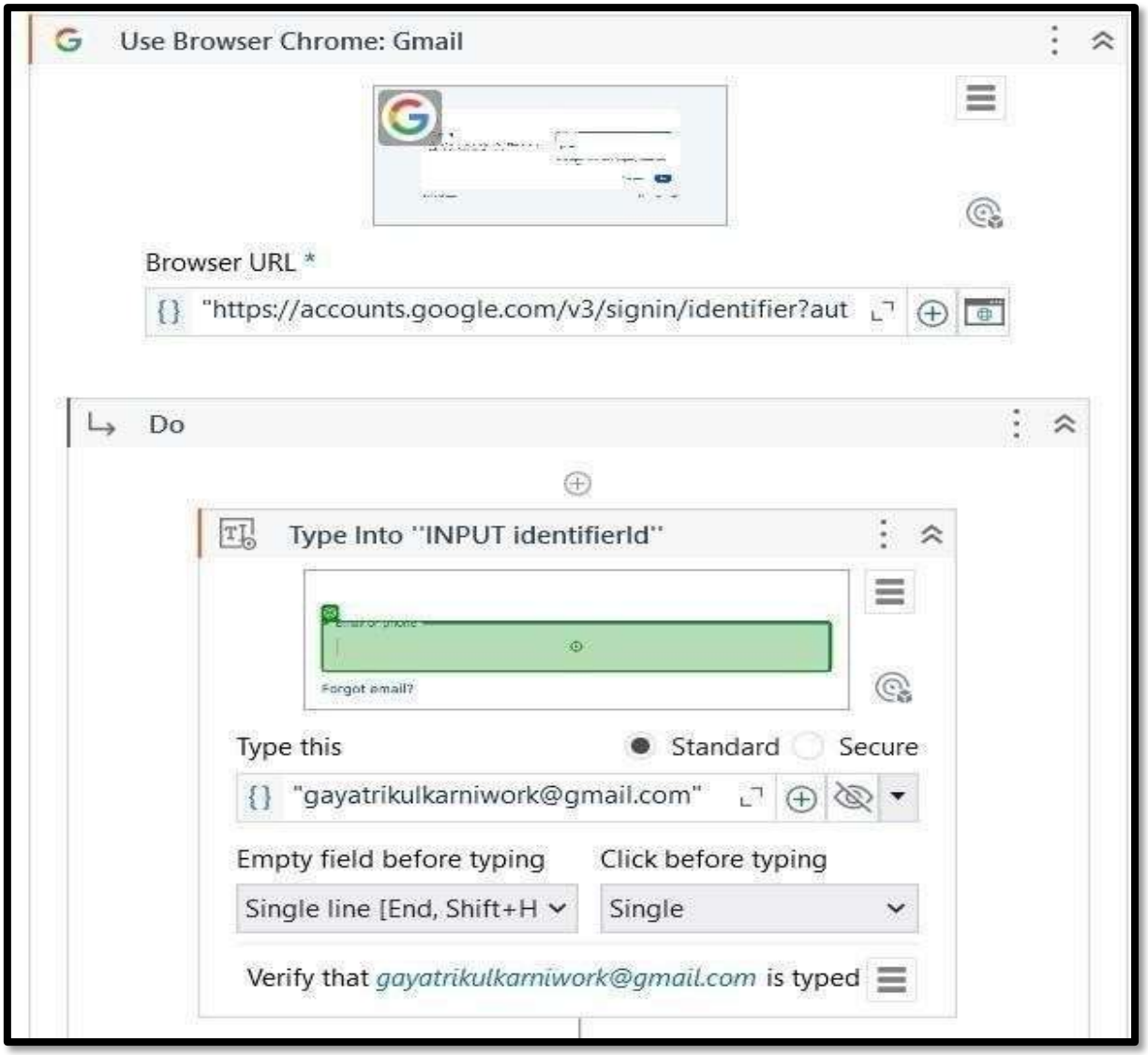


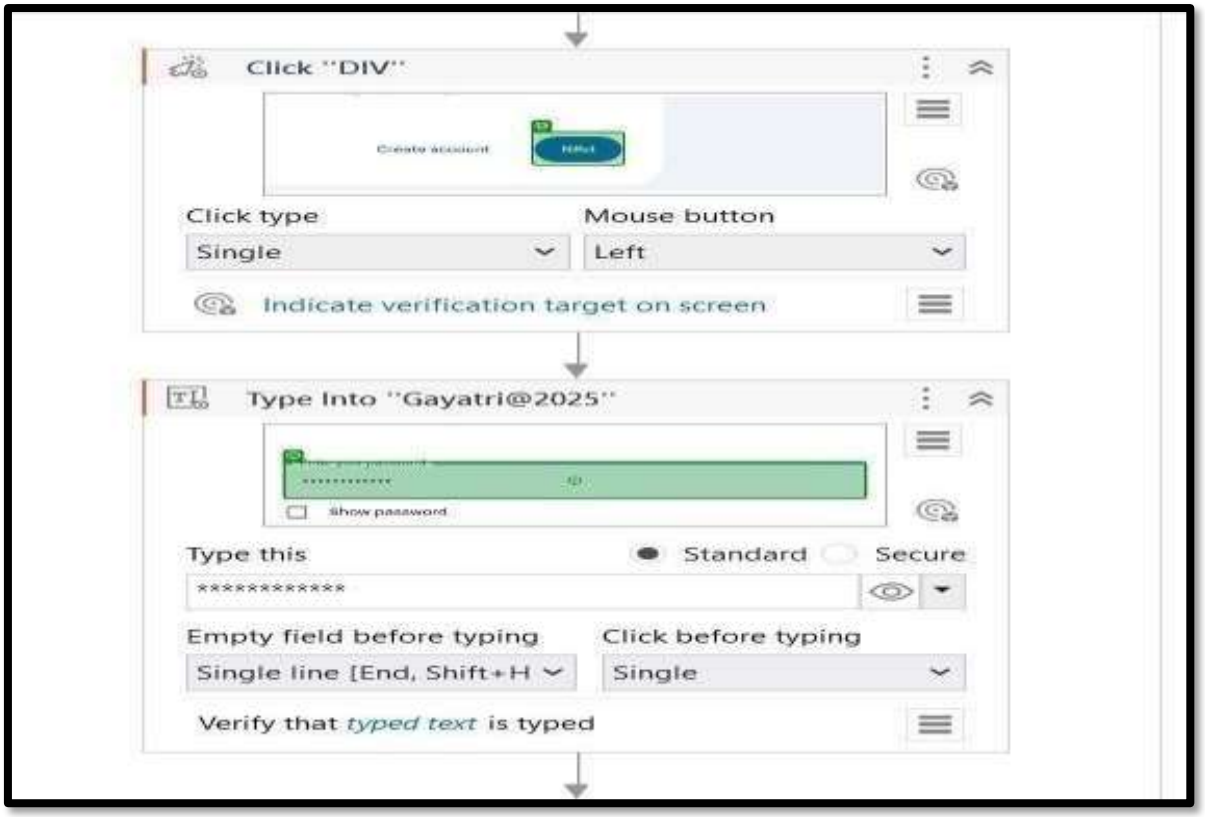
c) Demonstrate the following activities in UiPath: i. Mouse(click, double click and hover) ii. Type into:

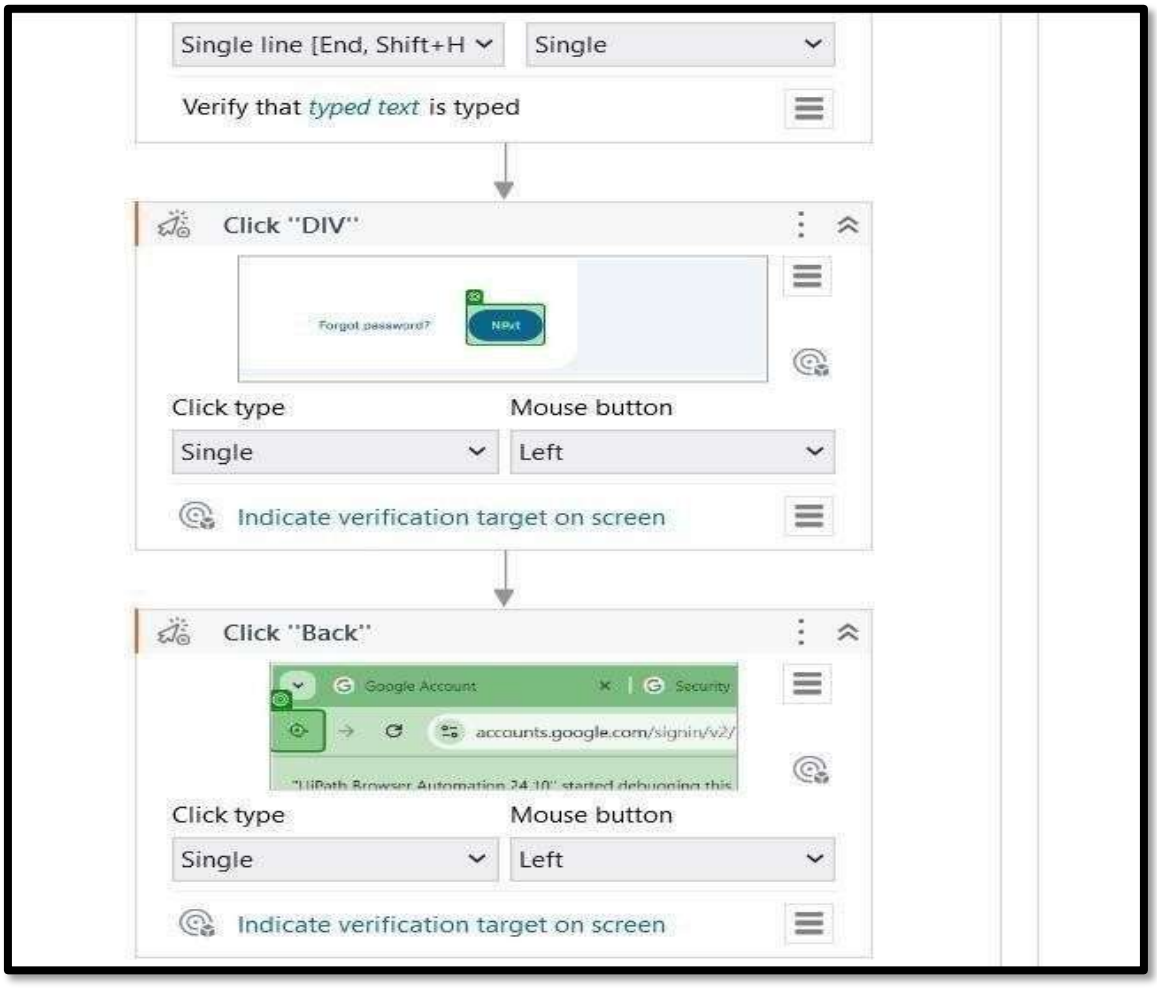




iii. **Type Secure Text:**

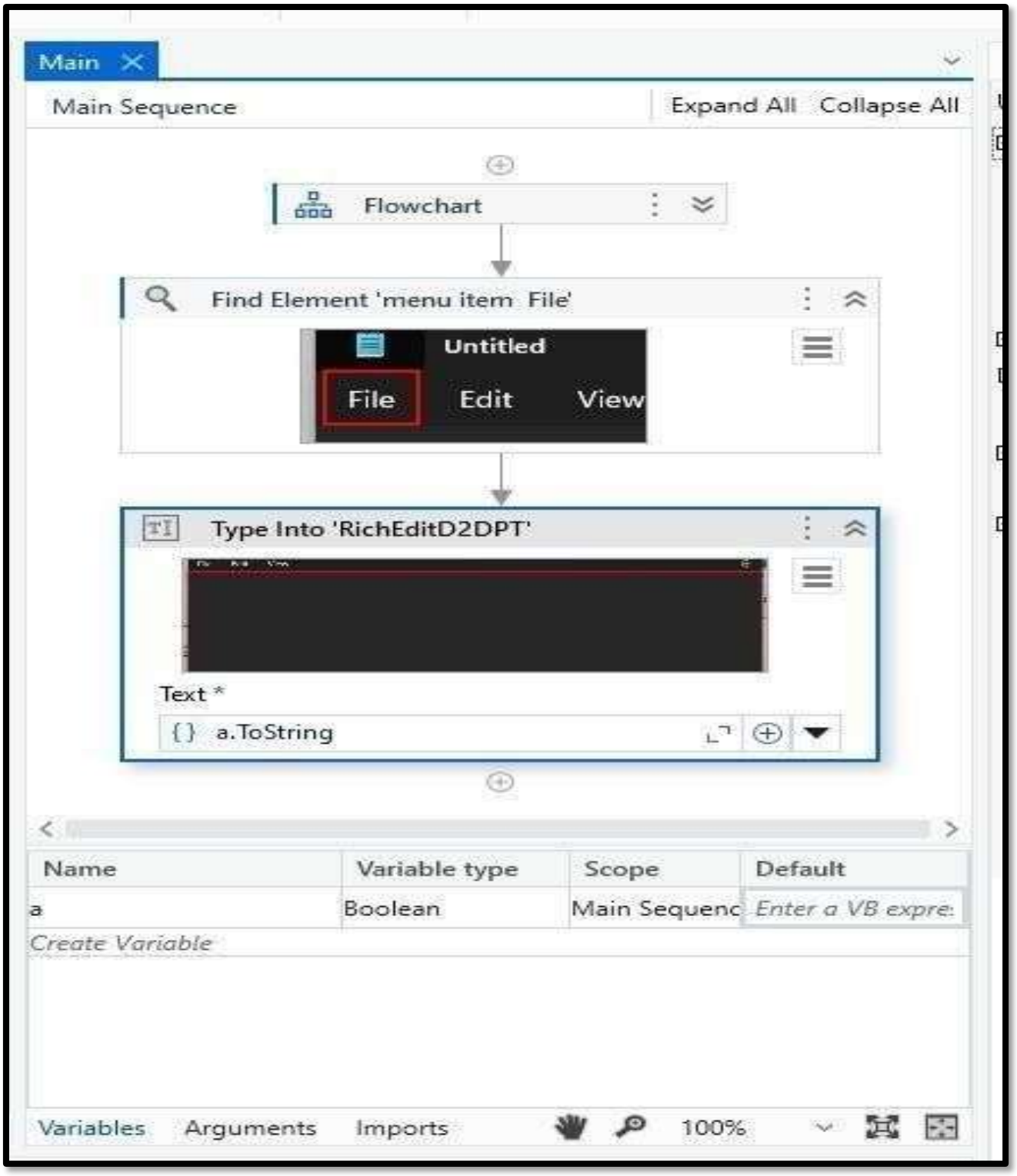



Click "DIV"
Click type
Single
Mouse button
Left
Indicate verification target on screen
Type Into "Gayatri@2025"
Show password
Type this
Standard
Secure
Empty field before typing
Single line [End, Shift+H
Click before typing
Single
Verify that typed text is typed


Single line [End, Shift+H
Single
Verify that typed text is typed
Click "DIV"
Forgot password?
Click type
Single
Mouse button
Left
Indicate verification target on screen
Click "Back"
Google Account
Security
accounts.google.com/signin/v2/
Click type
Single
Mouse button
Left
Indicate verification target on screen

c) **Demonstrate the following events in UiPath:**

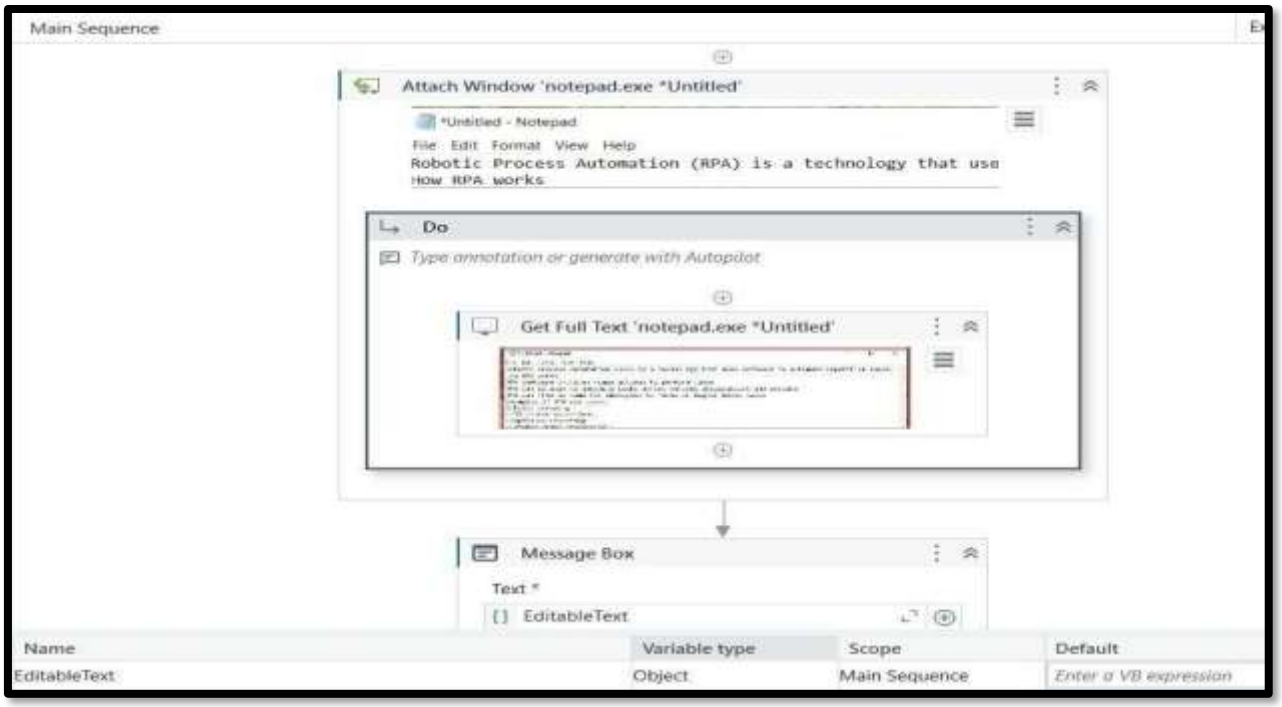
i. **Element triggering event:**



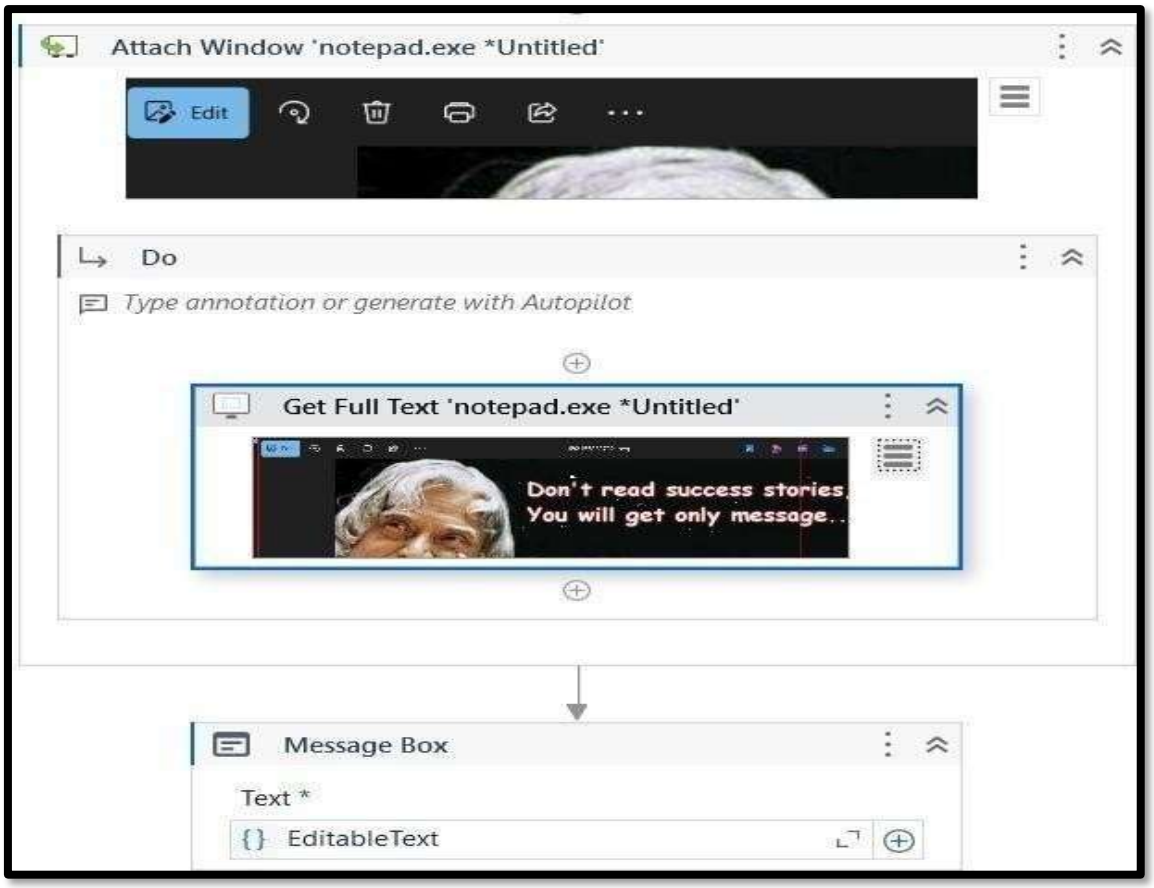
d) **Automate the following screen scraping methods using UiPath**

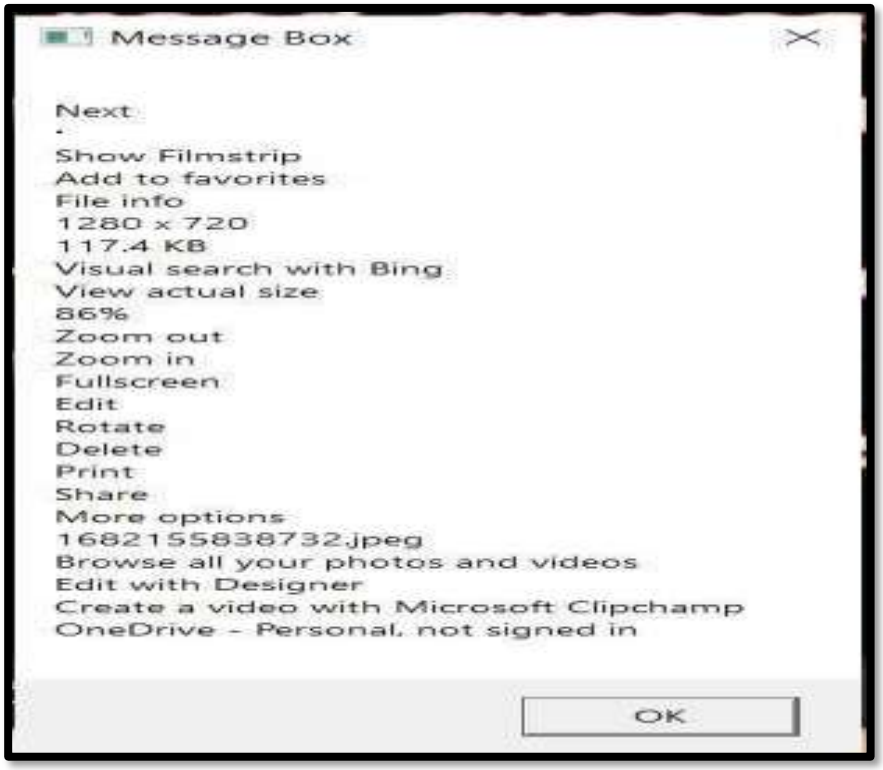
i. **Full Test**

ii. **OCR**

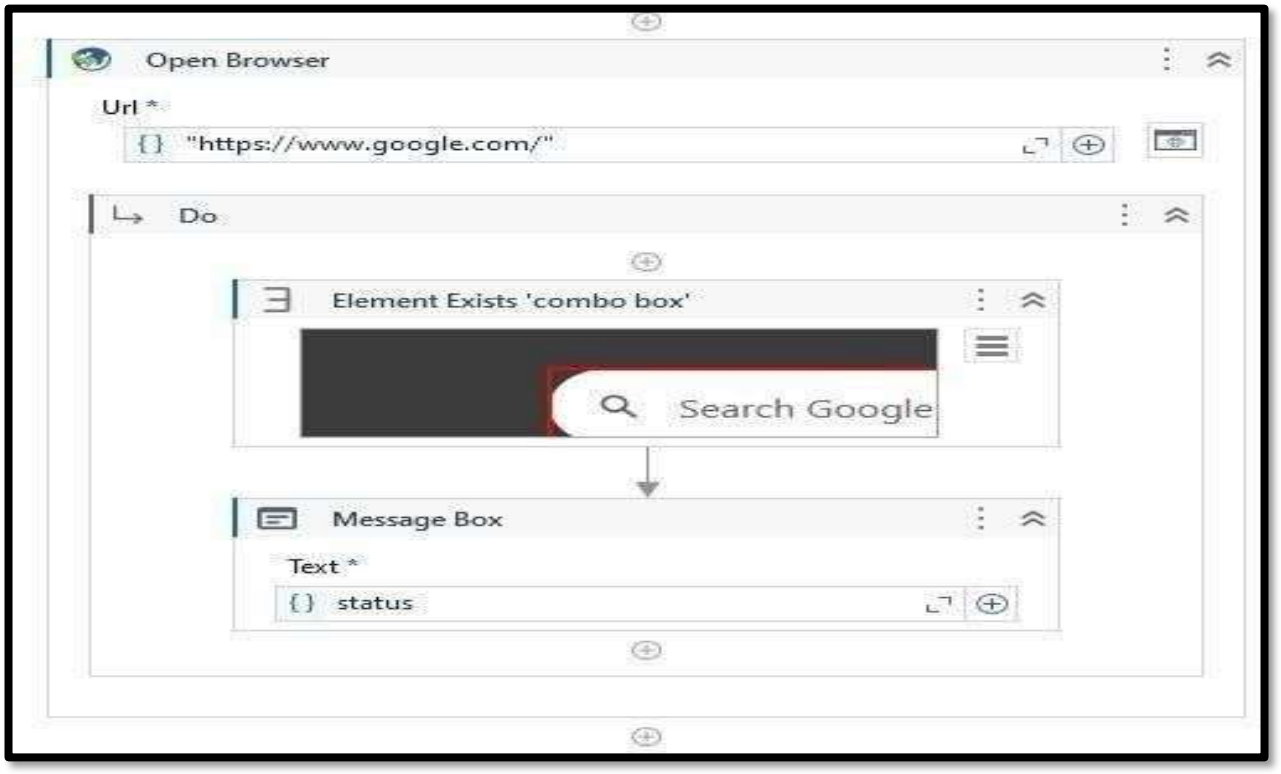


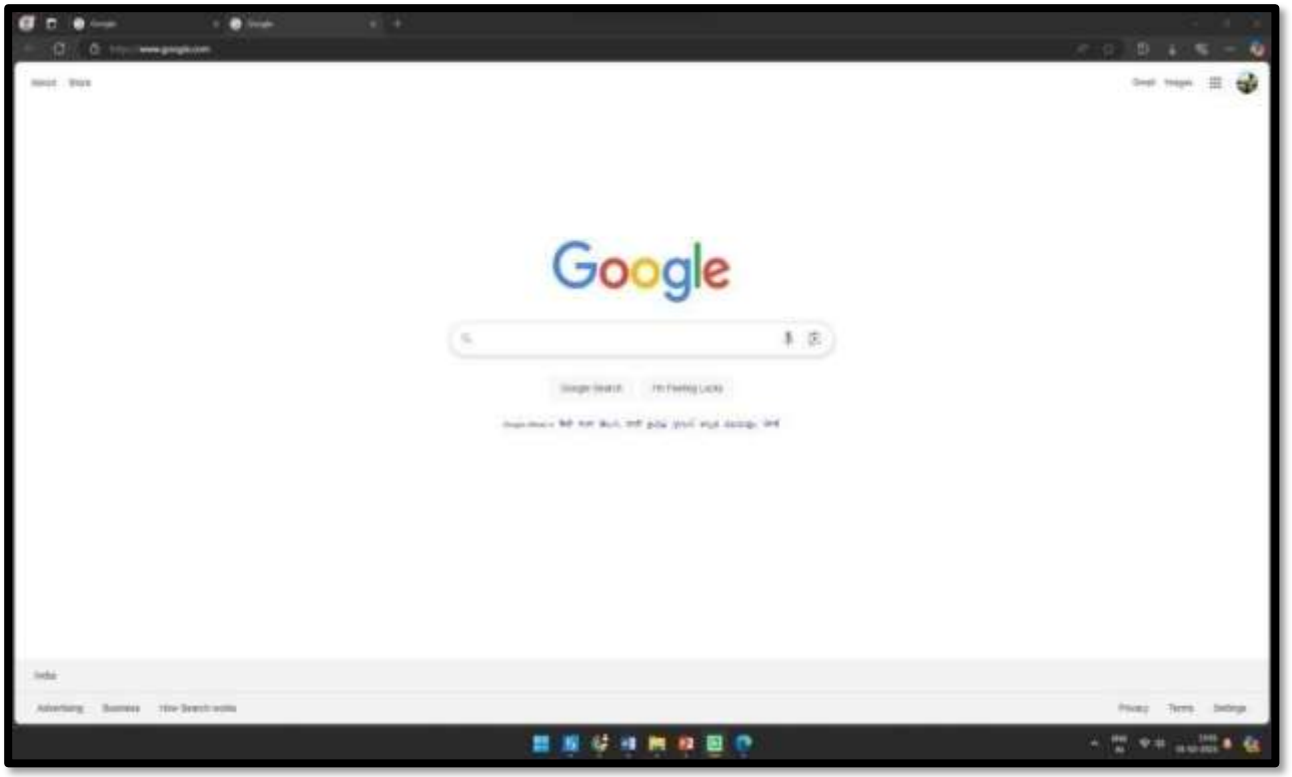
iii. Native


Attach Window 'notepad.exe *Untitled'
Edit
Do
Type annotation or generate with Autopilot
Get Full Text 'notepad.exe *Untitled'
Don't read success stories
You will get only message..
Message Box
Text *
{} EditableText

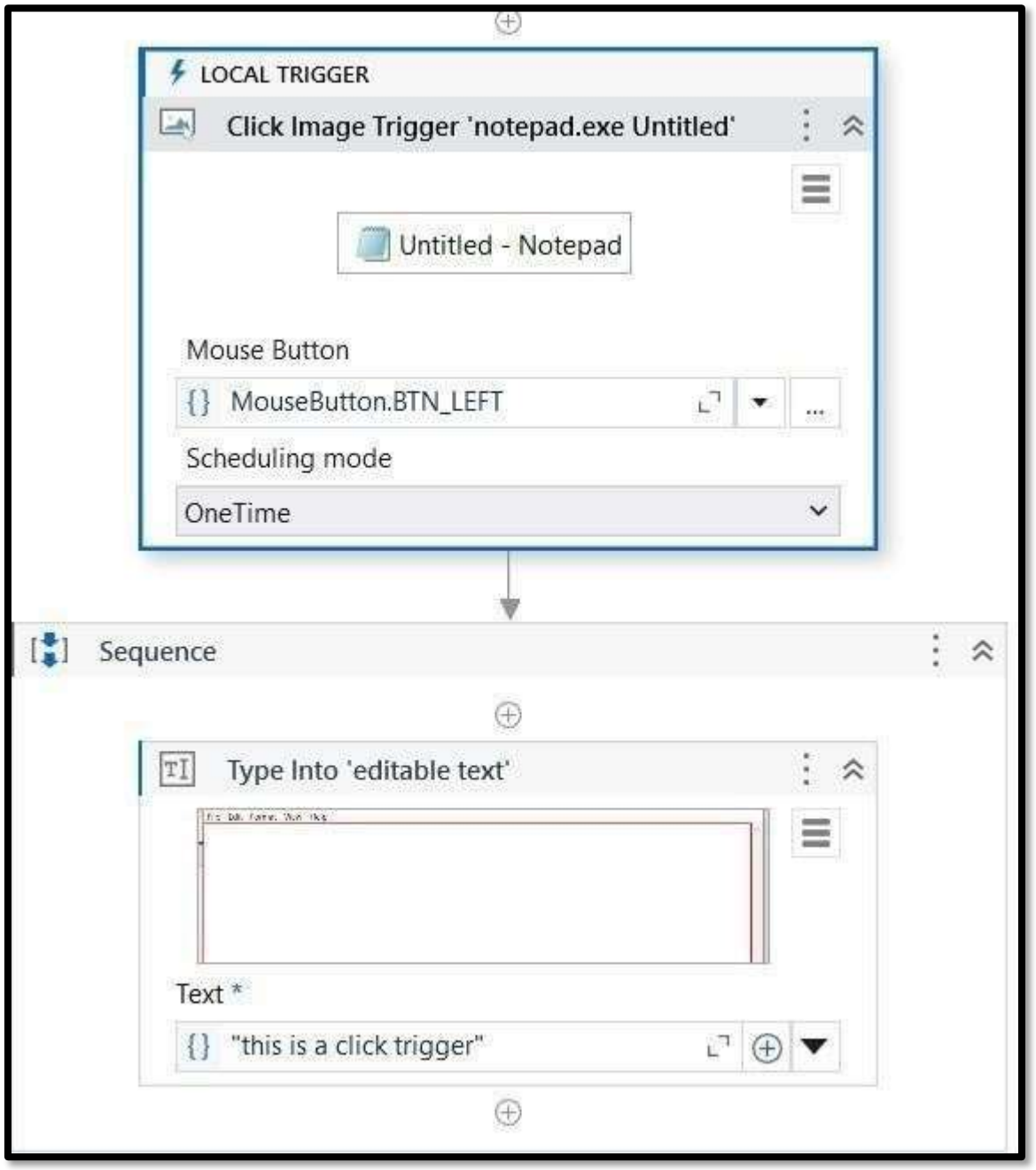

Message Box
Next
Show Filmstrip
Add to favorites
File info
1280 x 720
117.4 KB
Visual search with Bing
View actual size
86%
Zoom out
Zoom in
Fullscreen
Edit
Rotate
Delete
Print
Share
More options
1682155838732.jpeg
Browse all your photos and videos
Edit with Designer
Create a video with Microsoft Clipchamp
OneDrive - Personal, not signed in
OK

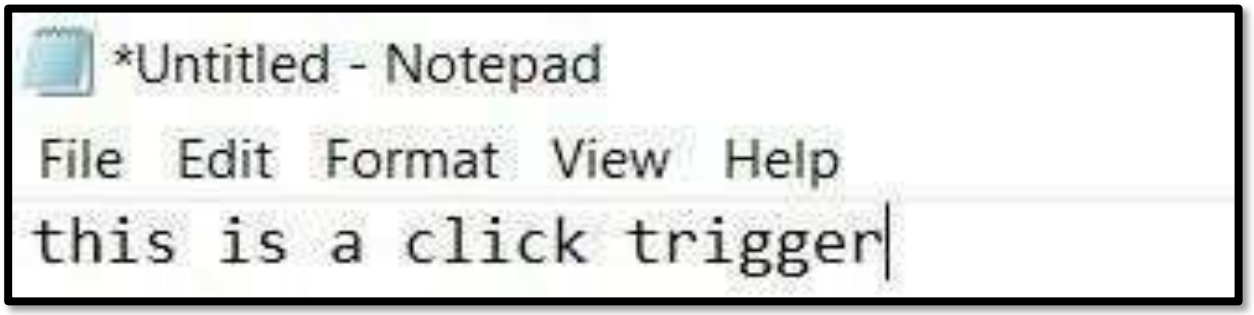
e) **Automate the process of launching an assistant bot on a keyboard event:**



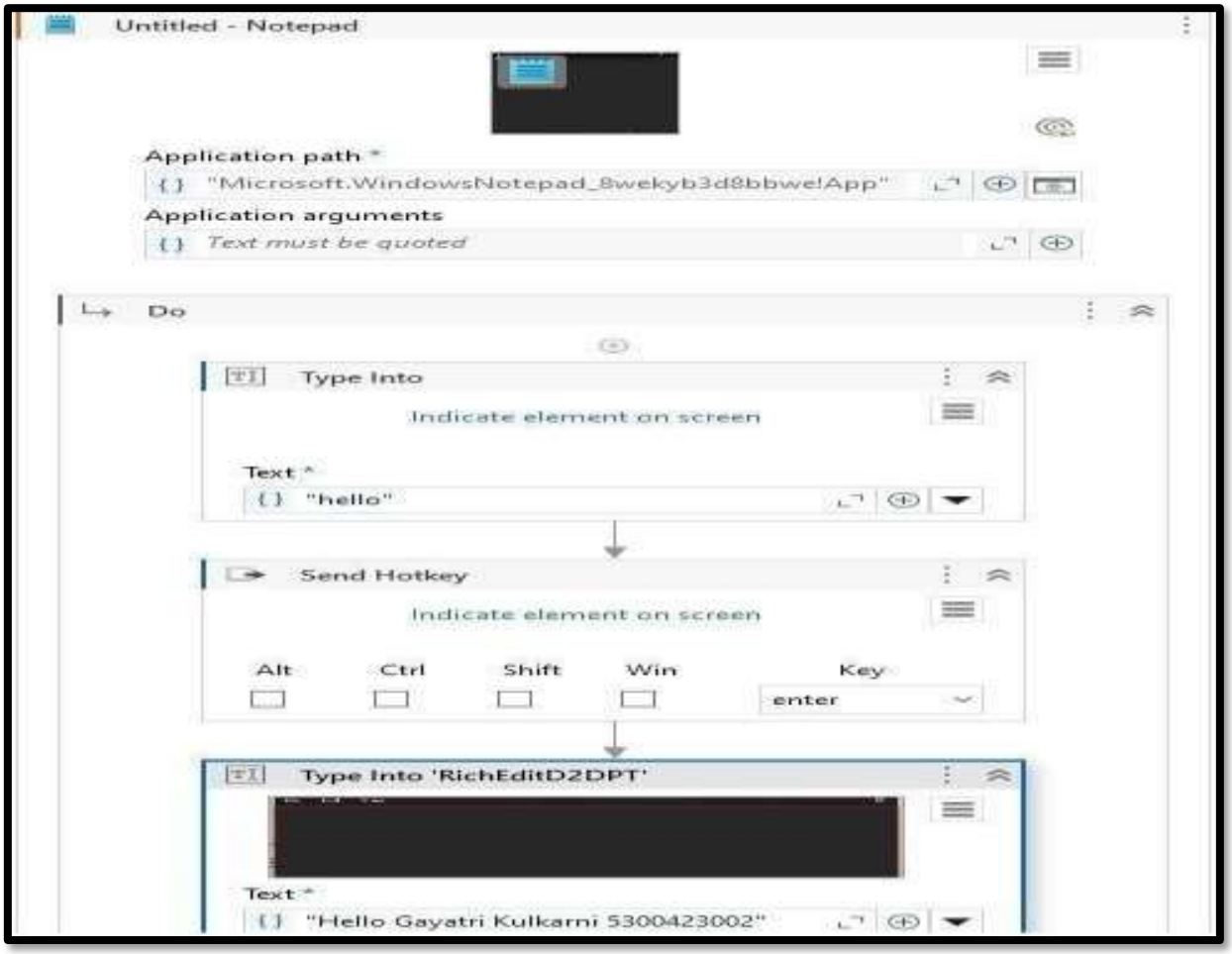


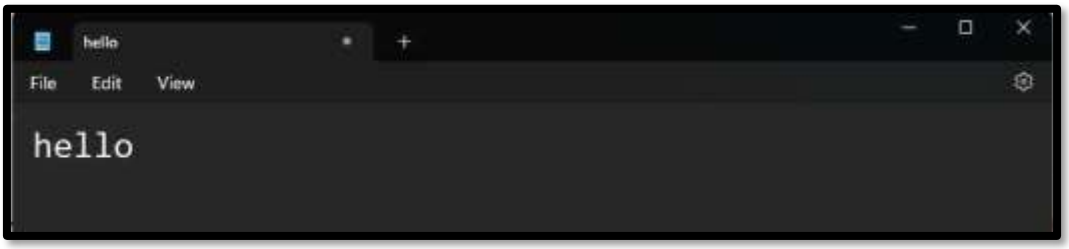
i. **Image triggering event**





iii. **System Triggering Event:**

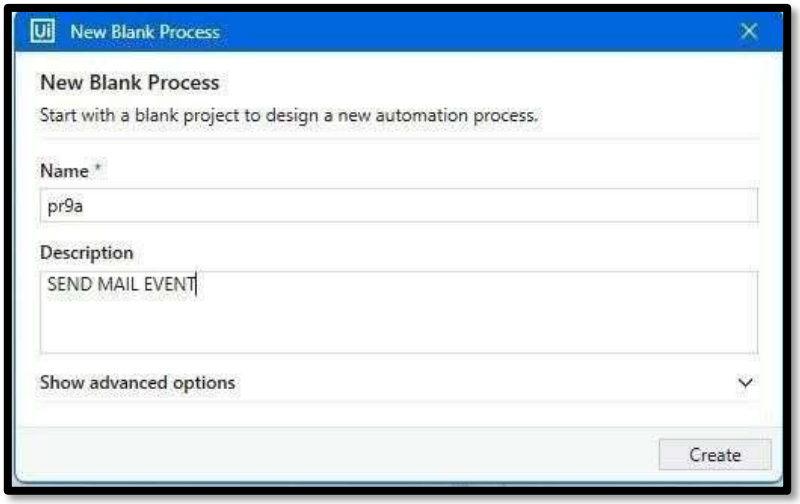


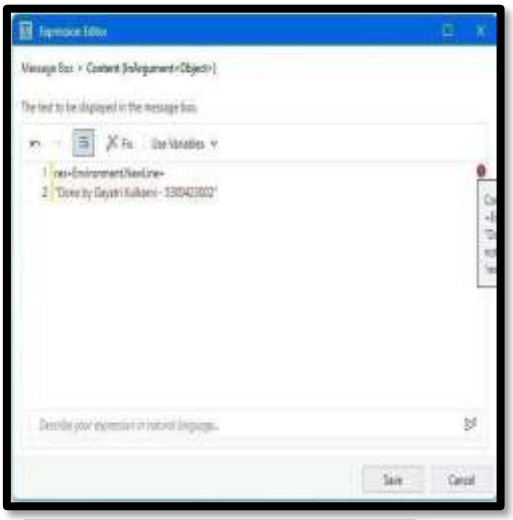


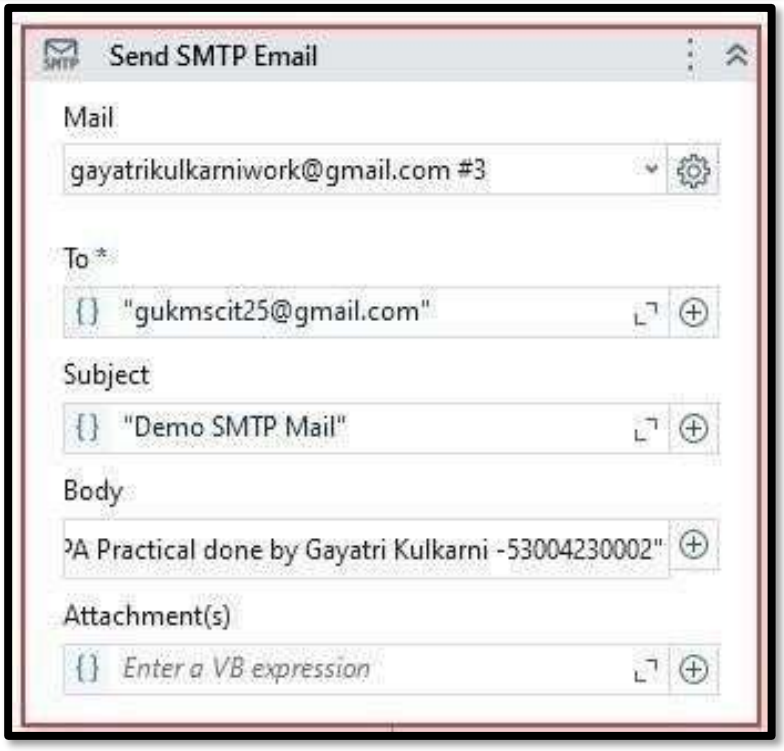
Module 4

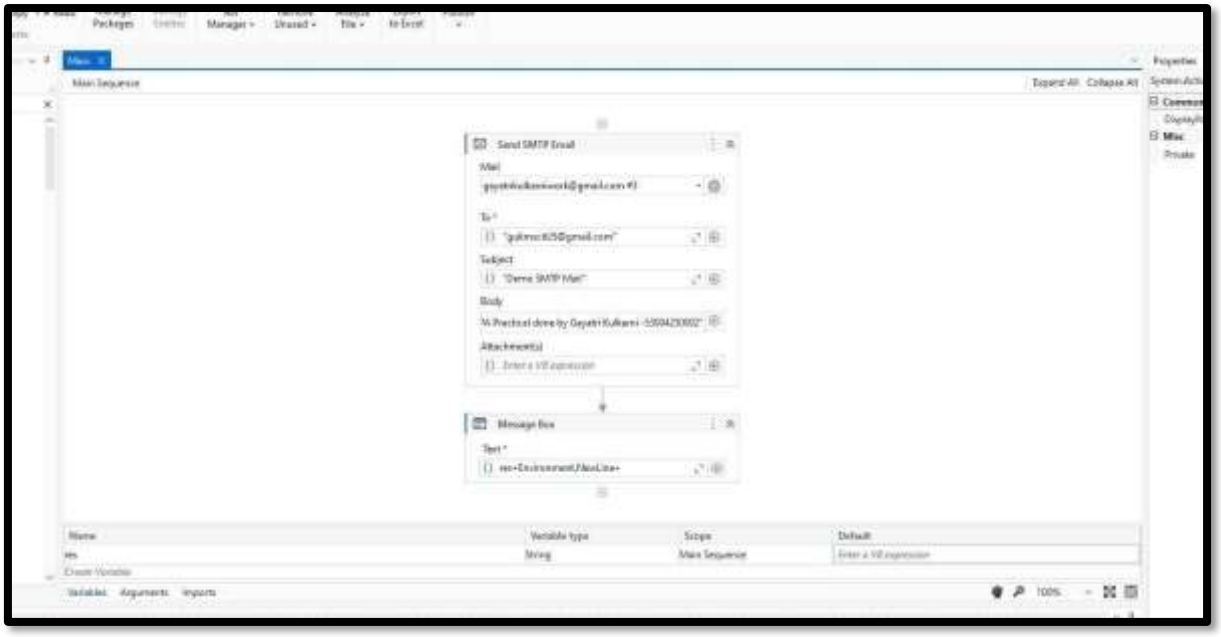
Exception Handling, Debugging, and Logging: Exception handling,Common exceptions and ways to handle them, Logging and taking screenshots, Debugging techniques, Collecting crash dumps, Error reporting

a) Automate the process of send mail event (on any email).

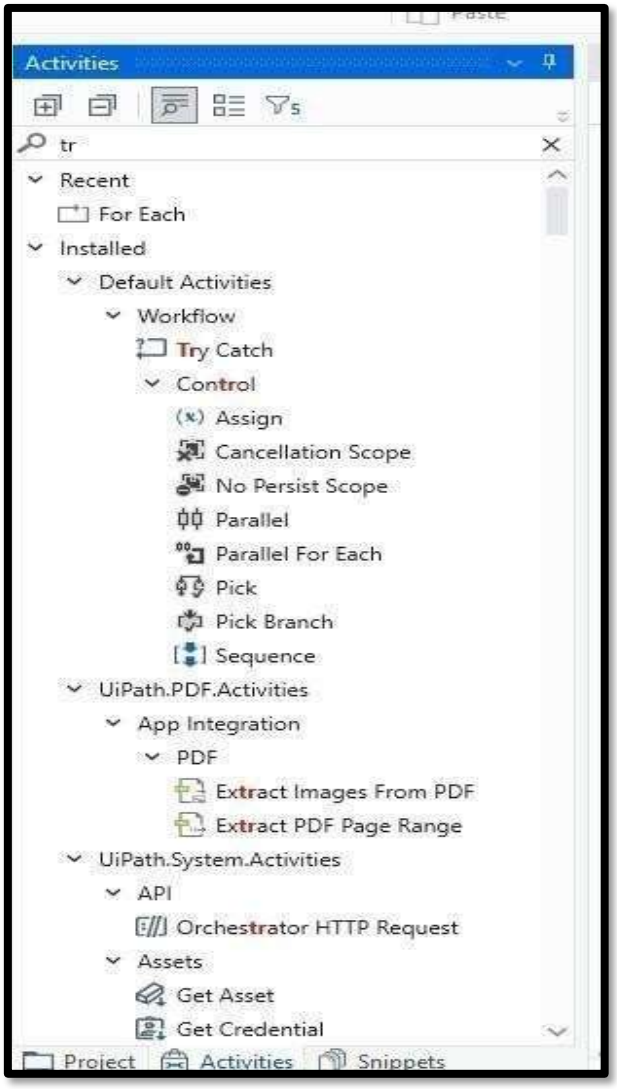


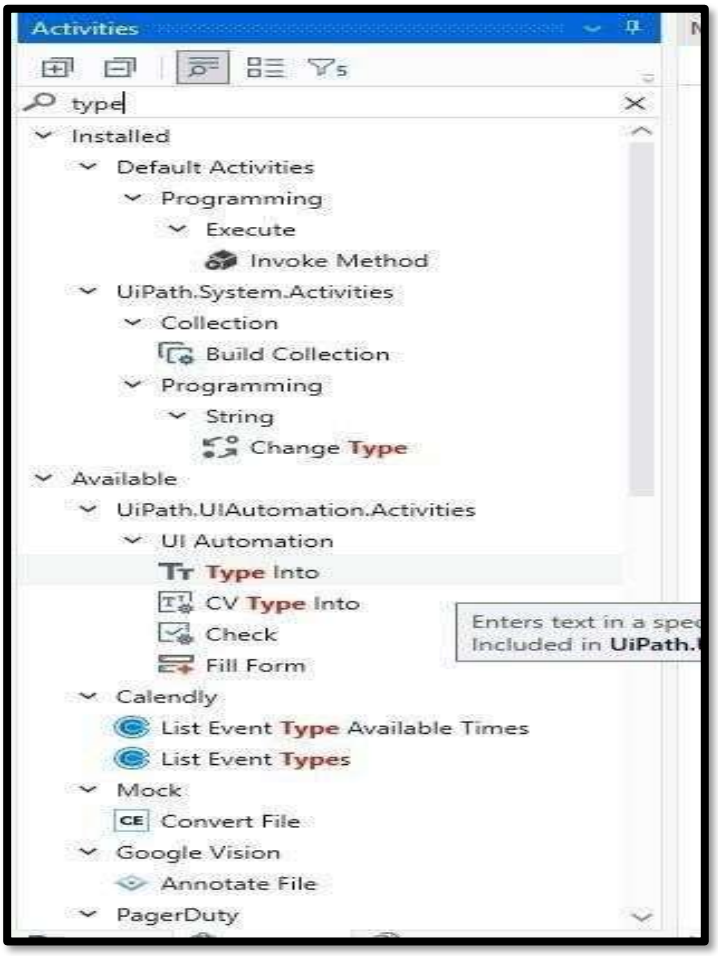


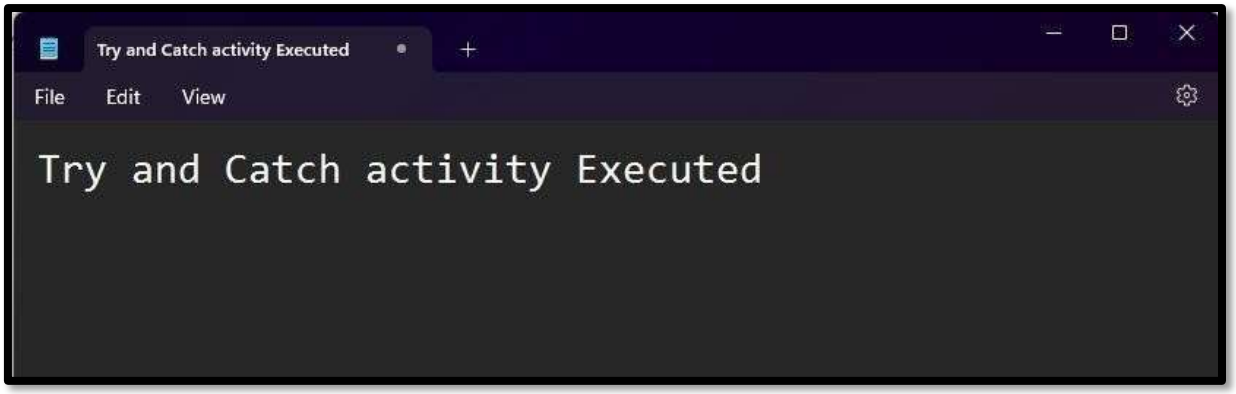




b) Demonstrate the Exception handing in UiPath.



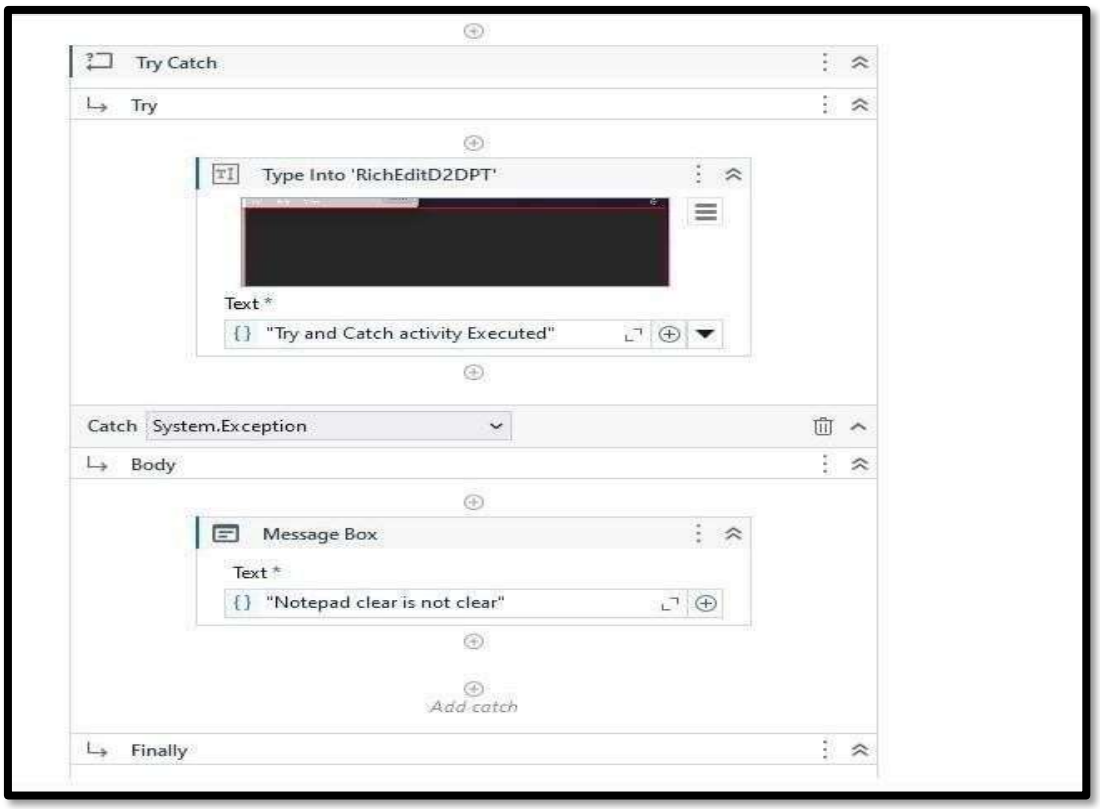


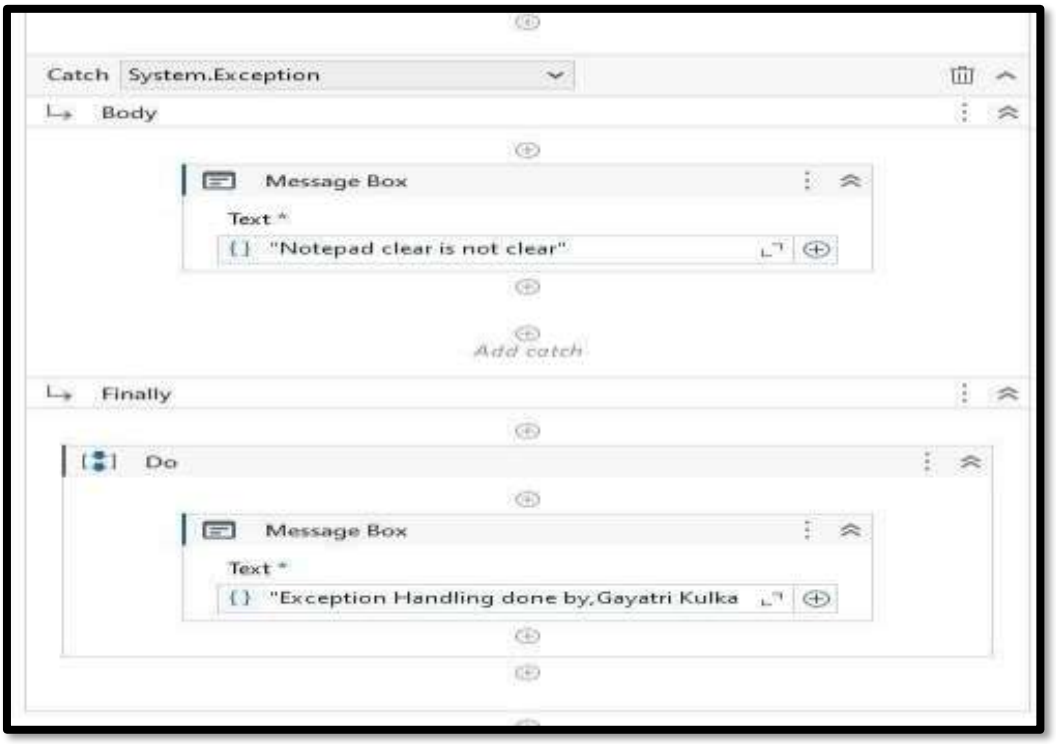

Try and Catch activity Executed
File
Edit
View
Try and Catch activity Executed


Message Box
Exception Handling done by,Gayatri Kulkarni-5300423002
OK

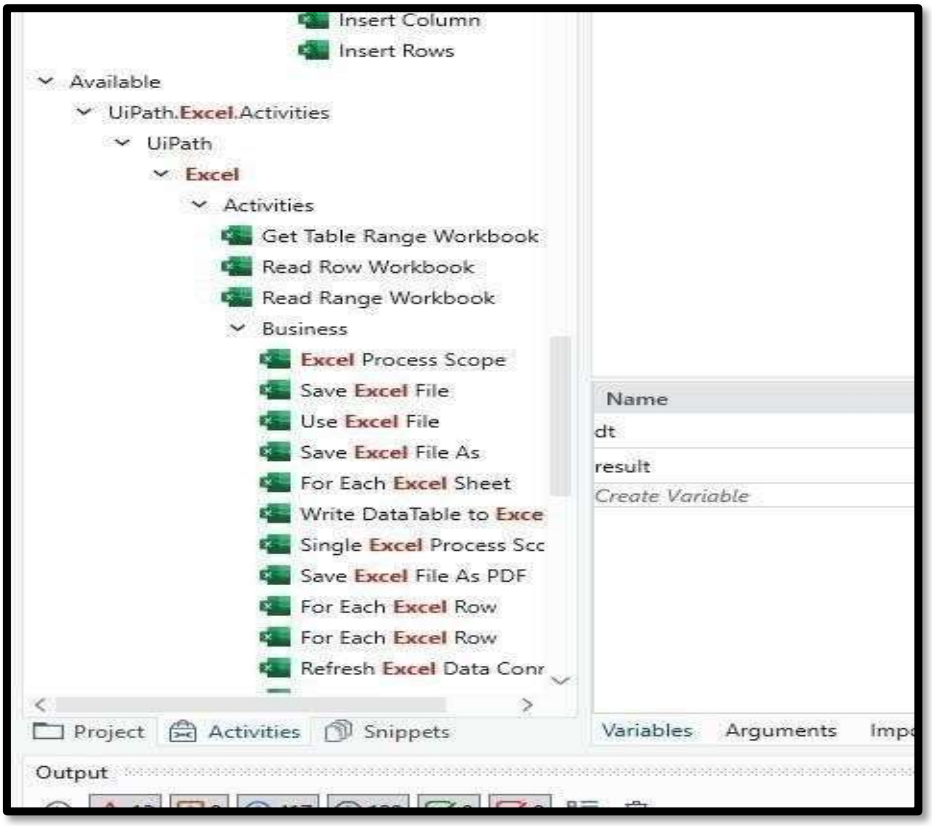

Message Box
UiPath.Core.SelectorNotFoundException: Could not find the UI element corresponding to this selector:
[1] <wnd app='notepad.exe' appid='Microsoft.WindowsNotepad_8wekyb3d8bbwe!App' title='Untitled - Notepad' />
[2] <wnd cls='RichEditD2DPT' />
Search failed at selector tag:
[1] <wnd app='notepad.exe' appid='Microsoft.WindowsNotepad_8wekyb3d8bbwe!App' title='Untitled - Notepad' />
The closest matches found are:
[85%] <wnd app='notepad.exe' appid='Microsoft.WindowsNotepad_8wekyb3d8bbwe!App' title='*Try and Catch activity Executed - Notepad'/>
at UiPath.Core.Activities.ObsoleteTaskAsyncCodeActivity`1.EndExecute(AsyncCodeActivityContext context, IAsyncResult result)
at System.Activities.AsyncCodeActivity.CompleteAsyncCodeActivityData.CompleteAsyncCodeActivityWorkItem.Execute(ActivityExecutor executor, BookmarkManager bookmarkManager)
OK

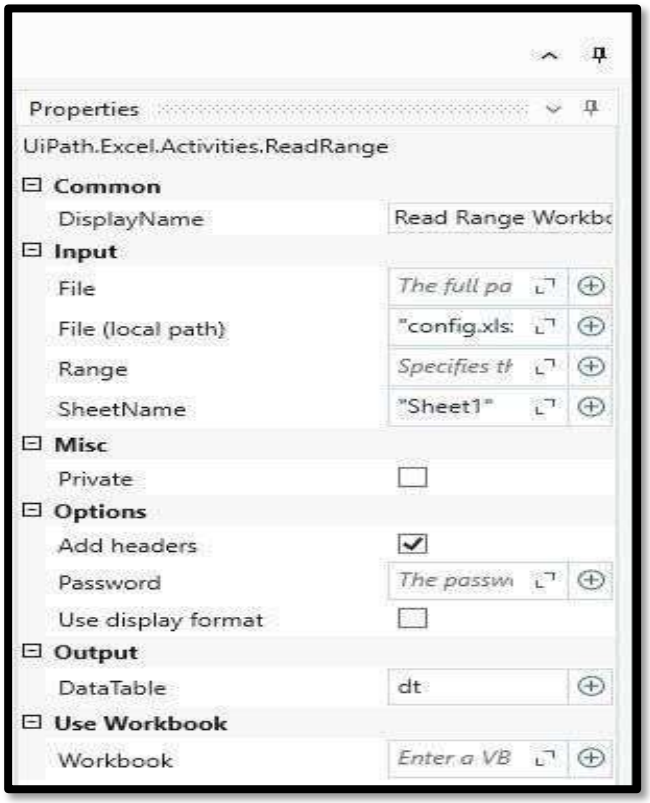

Message Box
Notepad clear is not clear
OK

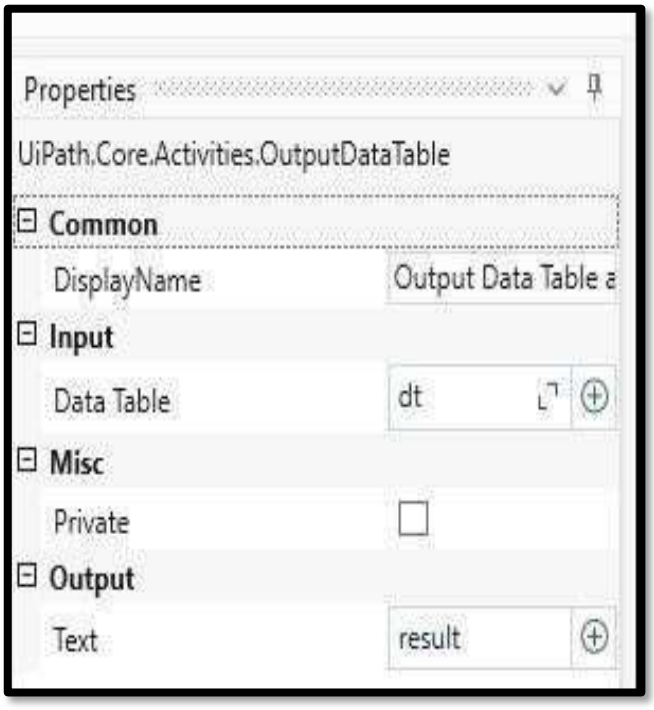

Try Catch
Try
Type Into 'RichEditD2DPT'
Text *
{} "Try and Catch activity Executed"
Catch System.Exception
Body
Message Box
Text *
{} "Notepad clear is not clear"
Add catch
Finally

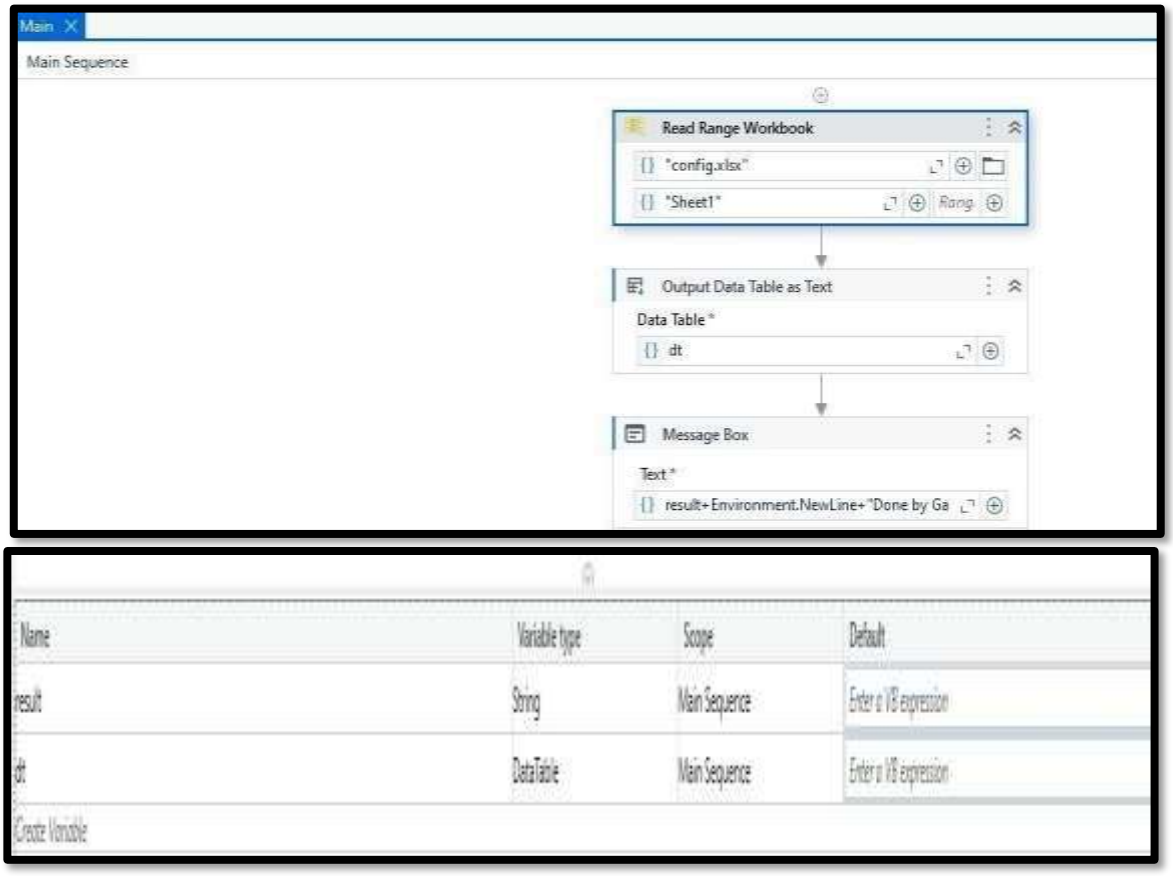

Catch System.Exception
Body
Message Box
Text *
{} "Notepad clear is not clear"
Add catch
Finally
Do
Message Box
Text *
{} "Exception Handling done by,Gayatri Kulka

c) Demonstrate the use of config files in UiPath:









d) Install and automate any process using

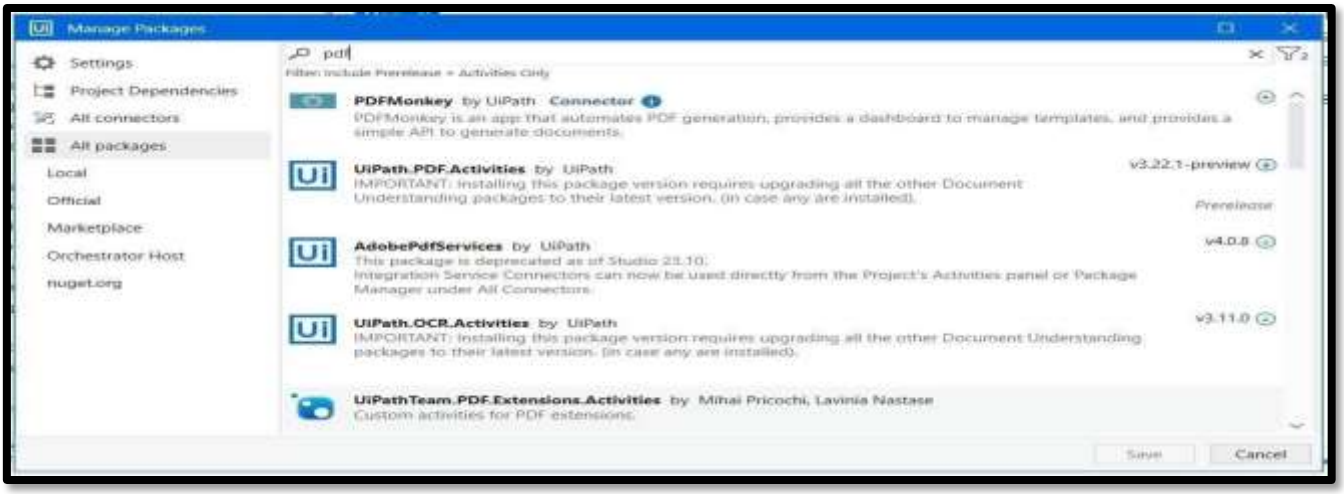
UiPath with the following plug-ins:

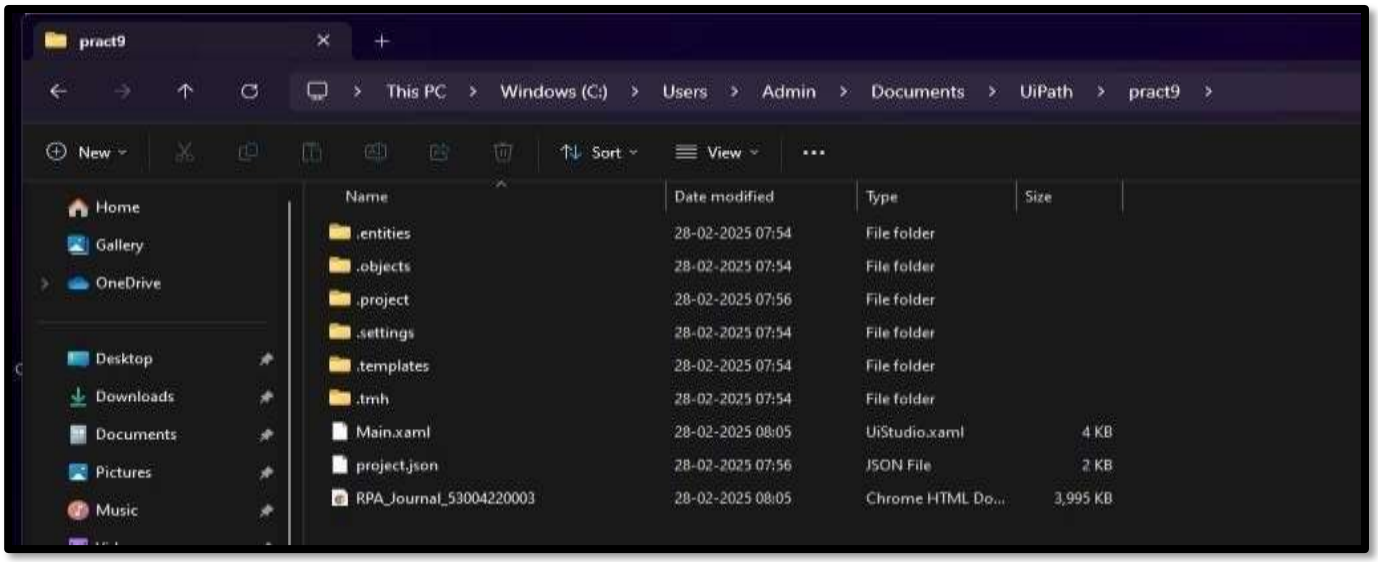
i. **Java Plugin**

ii. **Mail Plugin**

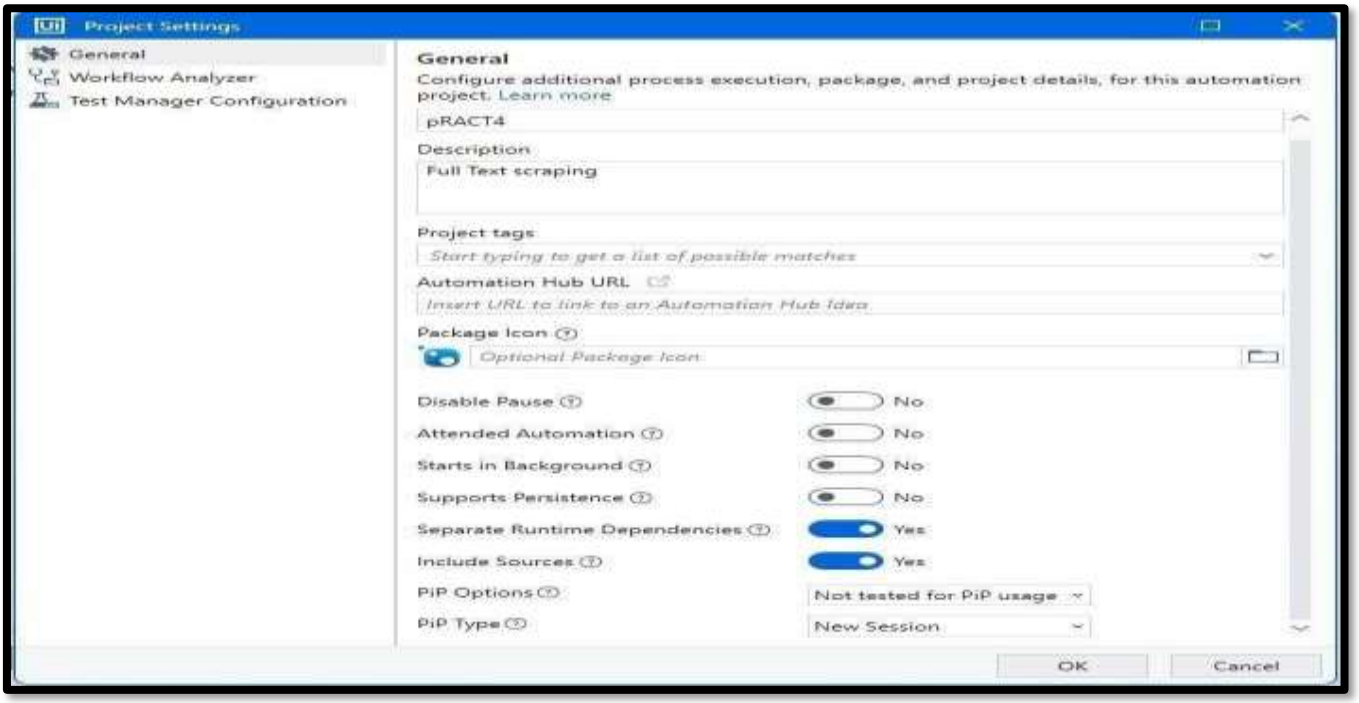
iii. **PDF Plugin**


HOME
DESIGN
DEBUG
Plugins - UiPath Studio Community
New
Save
Export As
Debug File
Cut
Copy
Paste
Undo
Redo
Manage Packages
Manage Entities
Test Manager
Remove Unused
Analyze File
Export to Excel
Publish


Manage Packages
Settings
Project Dependencies
All connectors
All packages
Local
Official
Marketplace
Orchestrator Host
nuget.org
pdf
PDFMonkey by UiPath Connector
UiPath.PDF.Activities by UiPath
v3.22.1-preview
Prerelease
AdobePdfServices by UiPath
v4.0.0
UiPath.OCR.Activities by UiPath
v3.11.0
UiPathTeam.PDF.Extensions.Activities by Mihai Pricochi, Lavinia Nastase
Save
Cancel

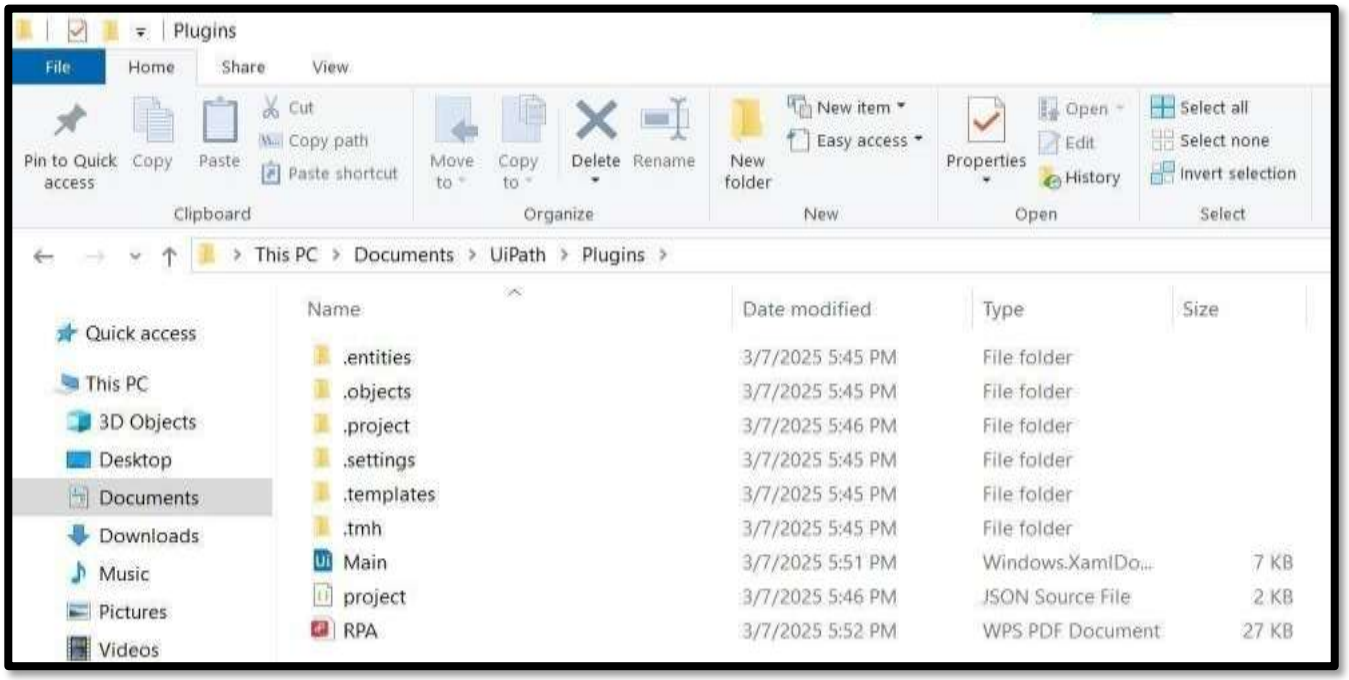

pract9
This PC
Windows (C:)
Users
Admin
Documents
UiPath
pract9
New
Sort
View
Home
Gallery
OneDrive
Desktop
Downloads
Documents
Pictures
Music
Name
Date modified
Type
Size
.entities 28-02-2025 07:54 File folder
.objects 28-02-2025 07:54 File folder
.project 28-02-2025 07:56 File folder
.settings 28-02-2025 07:54 File folder
.templates 28-02-2025 07:54 File folder
.tmh 28-02-2025 07:54 File folder
Main.xaml 28-02-2025 08:05 UiStudio.xaml 4 KB
project.json 28-02-2025 07:56 JSON File 2 KB
RPA_Journal_53004220003 28-02-2025 08:05 Chrome HTML Do... 3,995 KB

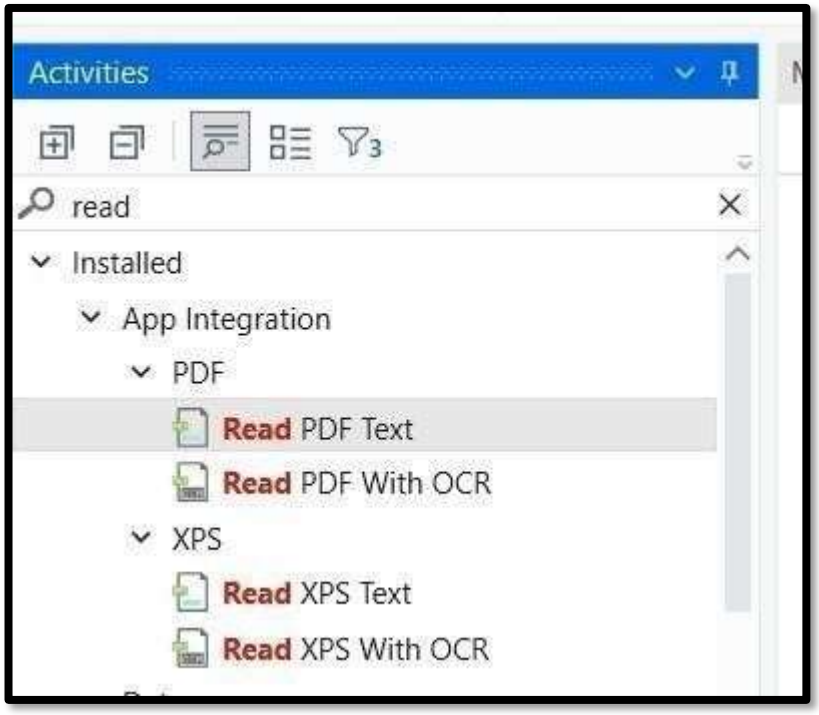

Manage Packages
Settings
Project Dependencies
All connectors
All packages
Local
Official
Marketplace
Microsoft Visual Studio Offlin
Orchestrator Host
nuget.org
Search
UiPath.PDF.Activities by UiPath
v3.21.0-preview
Prerelease
UiPath.System.Activities by UiPath
v25.2.1
Save
Cancel

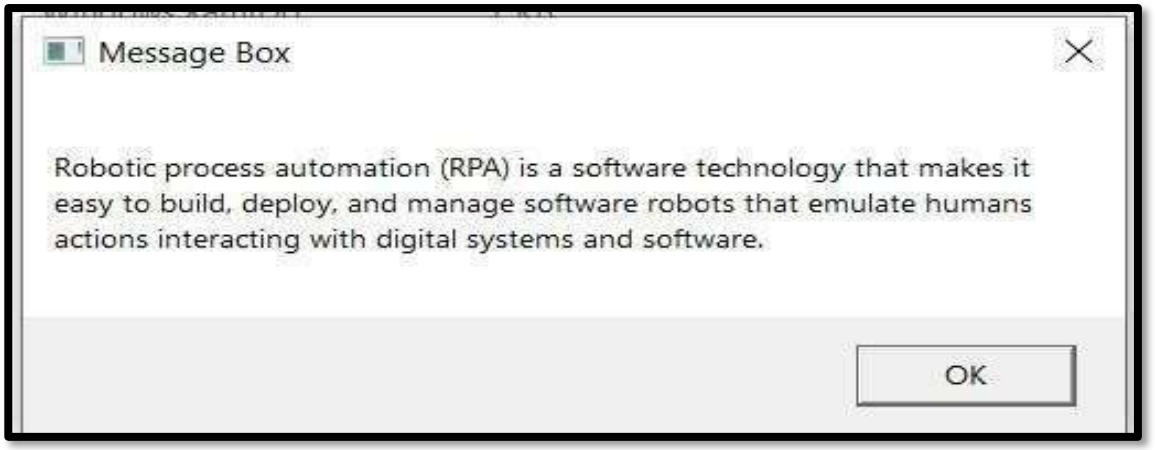

Project Settings
General
Workflow Analyzer
Test Manager Configuration
General
Configure additional process execution, package, and project details, for this automation project. Learn more
pRACT4
Description
Full Text scraping
Project tags
Start typing to get a list of possible matches
Automation Hub URL
Insert URL to link to an Automation Hub Idea
Package Icon
Optional Package Icon
Disable Pause
Attended Automation
Starts in Background
Supports Persistence
Separate Runtime Dependencies
Include Sources
PiP Options
PiP Type
No
No
No
No
Yes
Yes
Not tested for PiP usage
New Session
OK
Cancel

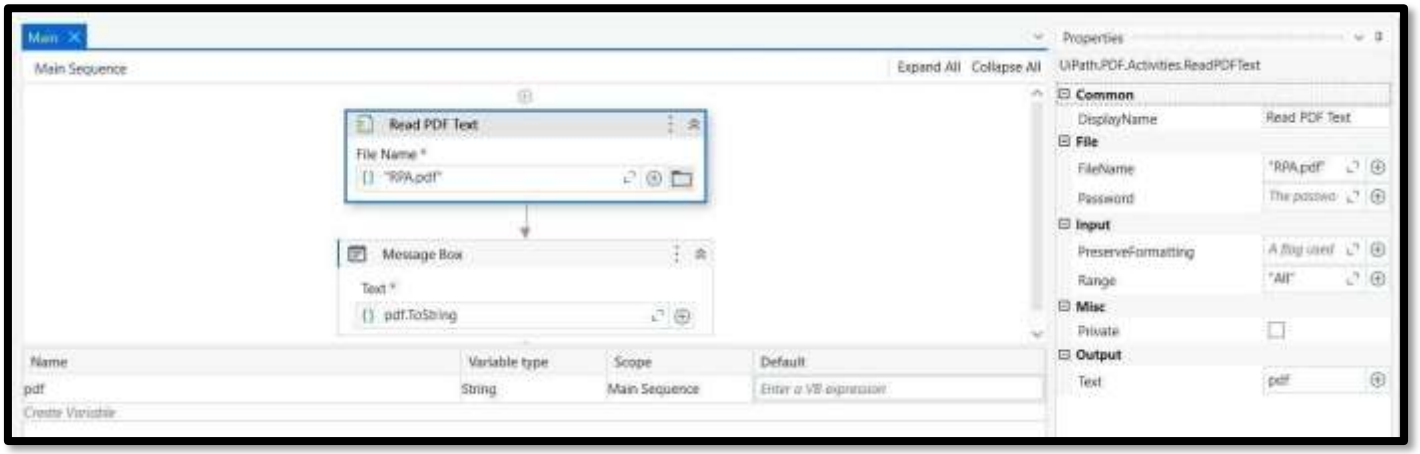

Manage Packages
Settings
Project Dependencies
All connectors
All packages
Local
Official
Marketplace
Microsoft Visual Studio Offlin
Orchestrator Host
nuget.org
pdf
PDFMonkey by UiPath Connector
UiPath.PDF.Activities by UiPath
v3.21.0-preview
Prerelease
AdobePdfServices by UiPath
v4.0.8
qBotica.PdfGeneration.Activities by qBotica
ConvertEncryptedPDF.Activities by Anoop AP
UiPath.PDF.Activities
Version:
3.21.0-preview
Install
Runtime Rule:
Strict
Install this package
Description:
Save
Cancel


Manage Packages
Settings
Project Dependencies
All connectors
All packages
Local
Official
Marketplace
Orchestrator Host
nuget.org
Search
Filter: Include Prerelease + Activities Only
UiPath.System.Activities by UiPath
IMPORTANT: This package version requires Studio v23.4 or above. Installing it on versions lower than the recommended ones may cause unexpected changes in your workflows.
Core activities which enable the robots to manipulate data tables and collections, work with files and...
v25.2.1
v25.2.2
UiPath.PDF.Activities by UiPath
IMPORTANT: Installing this package version requires upgrading all the other Document Understanding packages to their latest version. (in case any are installed).
v3.22.1-preview
Prerelease


Plugins
File
Home
Share
View
Pin to Quick access
Copy
Paste
Cut
Copy path
Paste shortcut
Clipboard
Move to
Copy to
Delete
Rename
Organize
New folder
New item
Easy access
New
Properties
Open
Edit
History
Open
Select all
Select none
Invert selection
Select
This PC > Documents > UiPath > Plugins >
Quick access
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Name
Date modified
Type
Size
.entities 3/7/2025 5:45 PM File folder
.objects 3/7/2025 5:45 PM File folder
.project 3/7/2025 5:46 PM File folder
.settings 3/7/2025 5:45 PM File folder
.templates 3/7/2025 5:45 PM File folder
.tmh 3/7/2025 5:45 PM File folder
Main 3/7/2025 5:51 PM Windows.XamlDo... 7 KB
project 3/7/2025 5:46 PM JSON Source File 2 KB
RPA 3/7/2025 5:52 PM WPS PDF Document 27 KB


Activities
read
Installed
App Integration
PDF
Read PDF Text
Read PDF With OCR
XPS
Read XPS Text
Read XPS With OCR

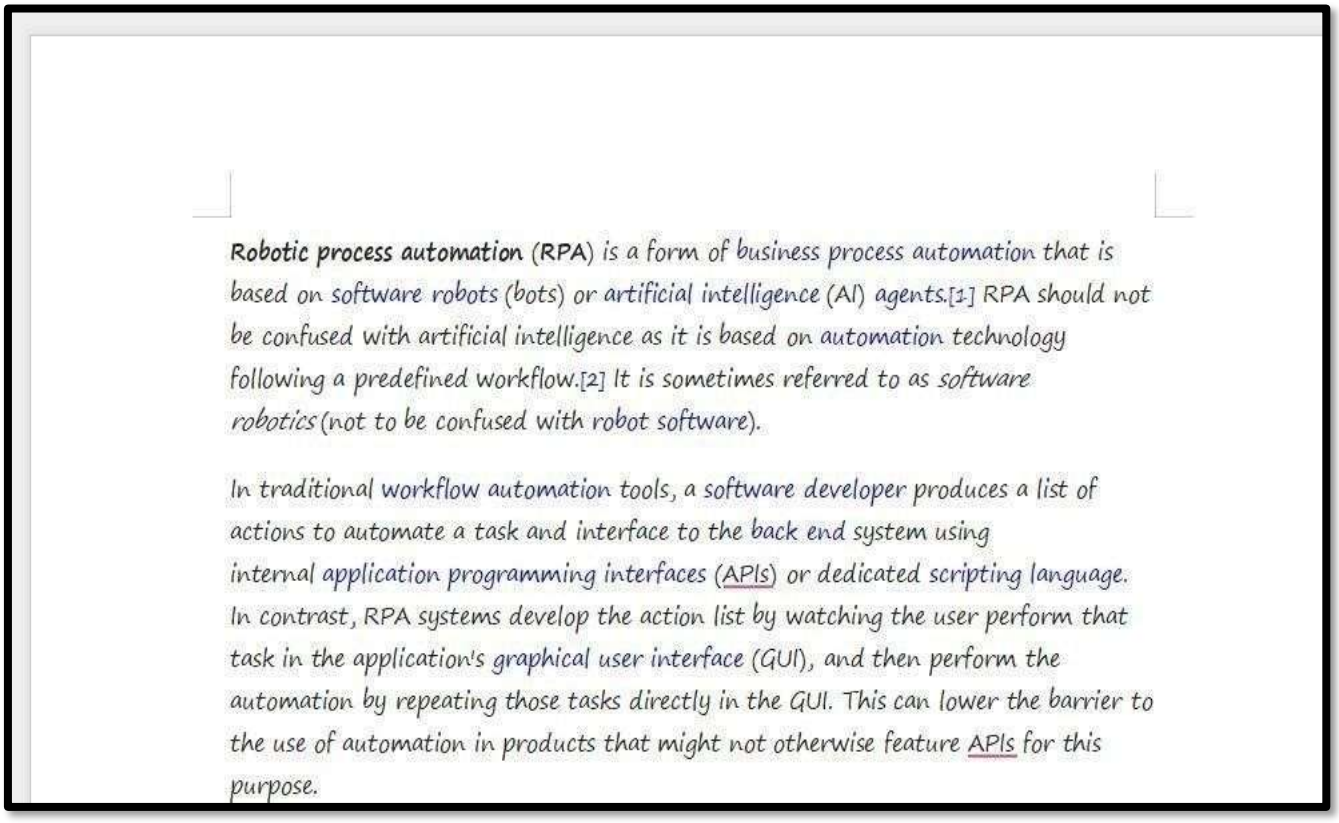

Message Box
Robotic process automation (RPA) is a software technology that makes it easy to build, deploy, and manage software robots that emulate humans actions interacting with digital systems and software.
OK



iv. **Web Integration**

v. **Excel Plugin**

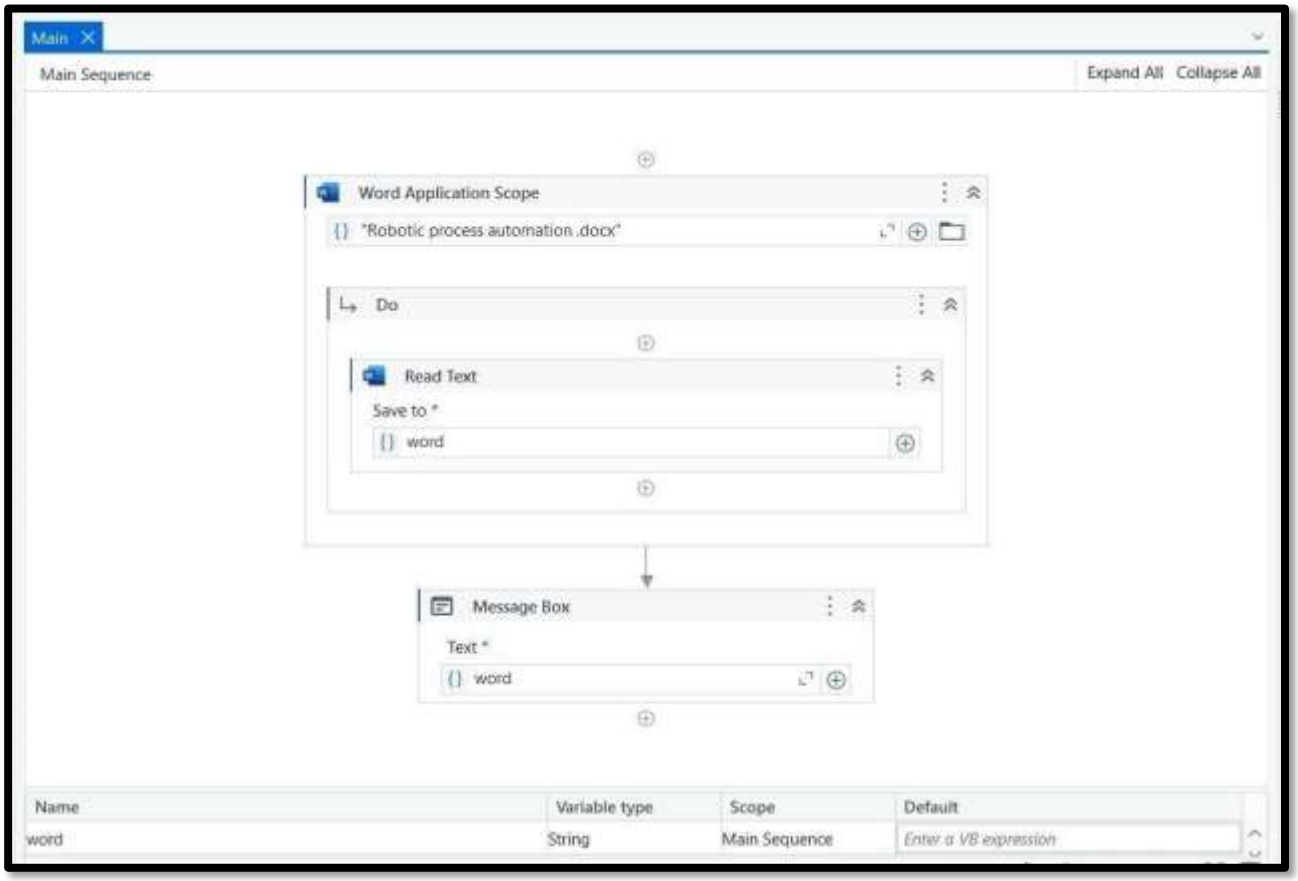
vi. **Word, Plugin**



Robotic process automation (RPA) is a form of business process automation that is based on software robots (bots) or artificial intelligence (AI) agents.[1] RPA should not be confused with artificial intelligence as it is based on automation technology following a predefined workflow.[2] It is sometimes referred to as *software robotics* (not to be confused with robot software).

In traditional workflow automation tools, a software developer produces a list of actions to automate a task and interface to the back end system using internal application programming interfaces (APIs) or dedicated scripting language. In contrast, RPA systems develop the action list by watching the user perform that task in the application's graphical user interface (GUI), and then perform the automation by repeating those tasks directly in the GUI. This can lower the barrier to the use of automation in products that might not otherwise feature APIs for this purpose.


Message Box
Robotic process "automation (RPA)" is a form "of business" process "automation that" is based "on software" "robots (bots)" "or artificial" "intelligence (AI) agents.[1] RPA" should not be confused with artificial intelligence as it is based "on automation technology" following a predefined "workflow.[2] It" is sometimes referred to "as software" "robotics (not" to be confused "with robot" software).
In "traditional workflow automation tools," "a software" "developer produces" a list of actions to automate a task and interface to "the back" "end system" using "internal application" programming "interfaces (APIs)" or "dedicated scripting" language. In contrast, RPA systems develop the action list by watching the user perform that task in the "application's graphical" user "interface (GUI)," and then perform the automation by repeating those tasks directly in the GUI. This can lower the barrier to the use of automation in products that might not otherwise feature APIs for this purpose.
OK


Main
Main Sequence
Expand All
Collapse All
Word Application Scope
"Robotic process automation .docx"
Do
Read Text
Save to *
word
Message Box
Text *
word
Name
Variable type
Scope
Default
word
String
Main Sequence
Enter a VB expression