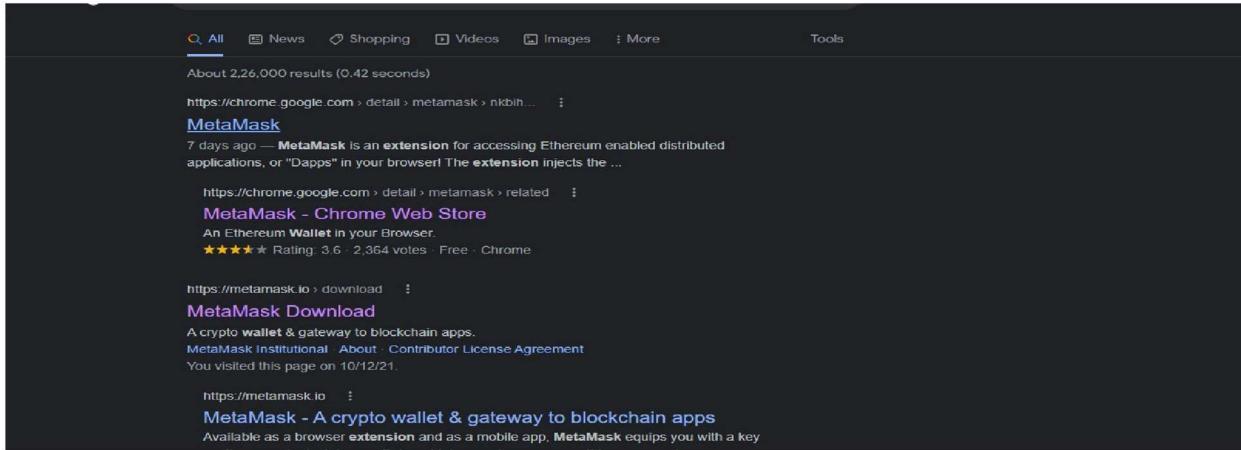
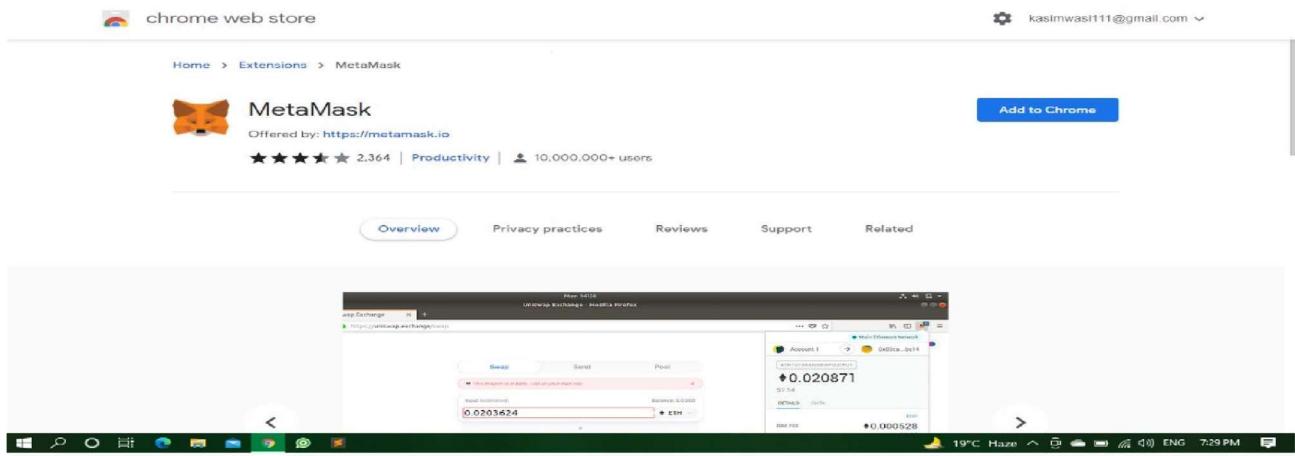


Practical 1-How to install, use and operate the MetaMask Wallet?

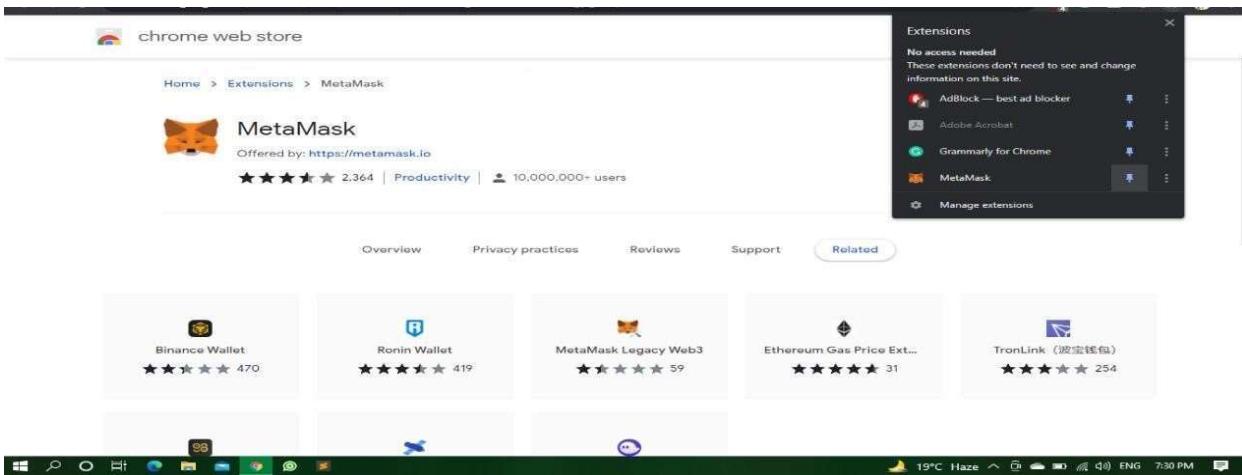
Step-1: Open Chrome Browser and search for “Metamask Wallet” and add extension download and open the first link.



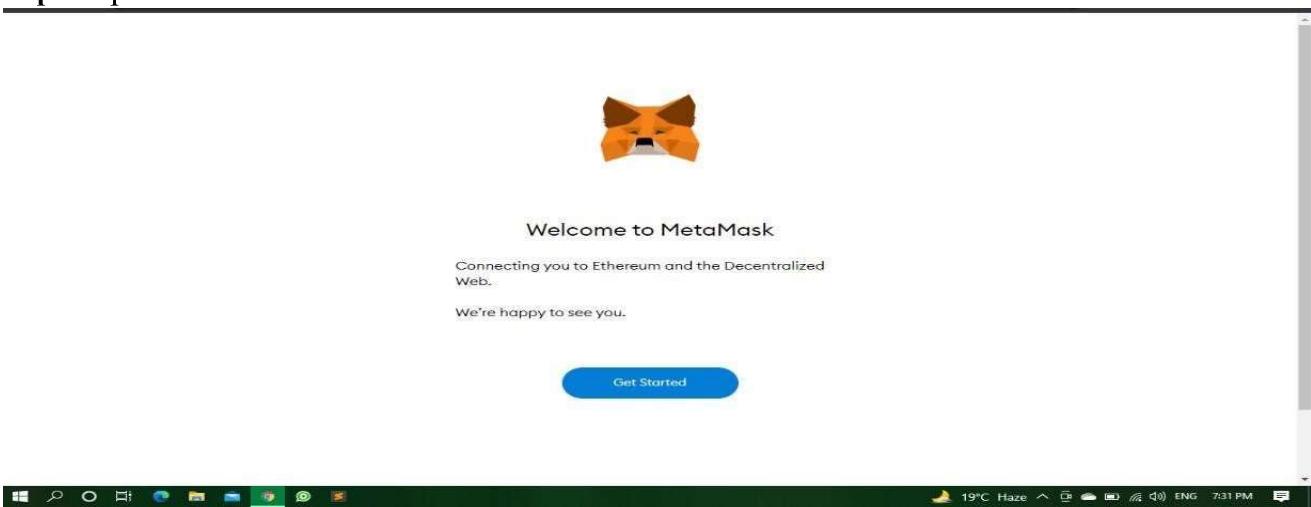
Step-2: Click on “Add to Chrome” and simply add it in your extension list.



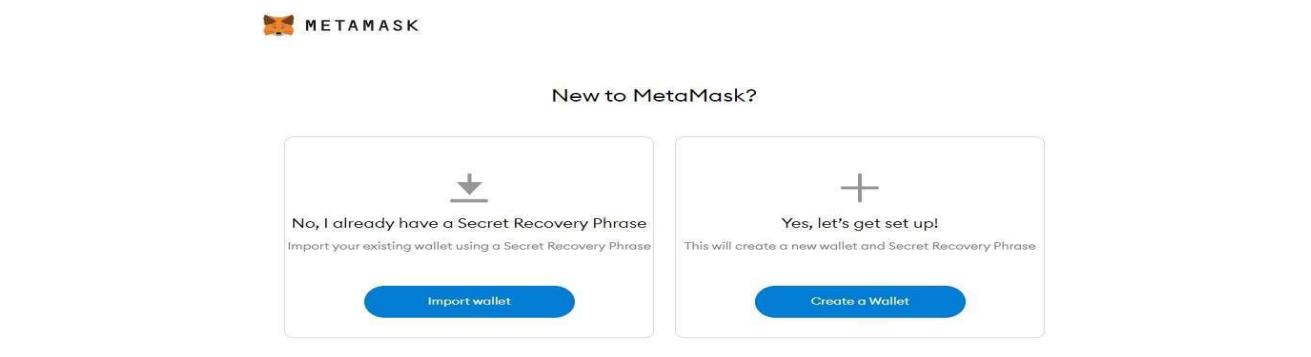
Step-3: click on extension option and pin the Metamask Wallet extension.



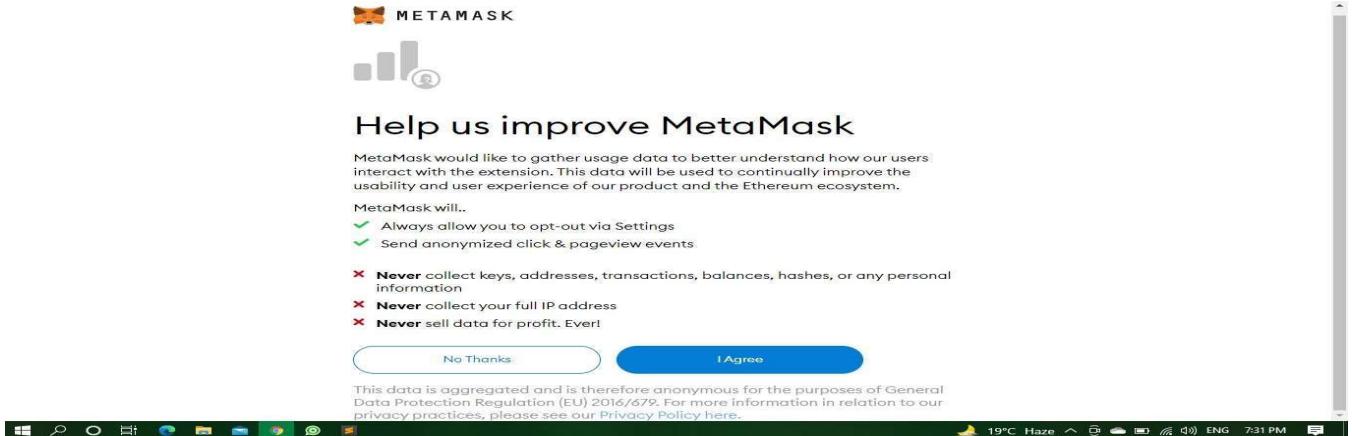
Step-4: Open Metamask Wallet and click “Get Started”.



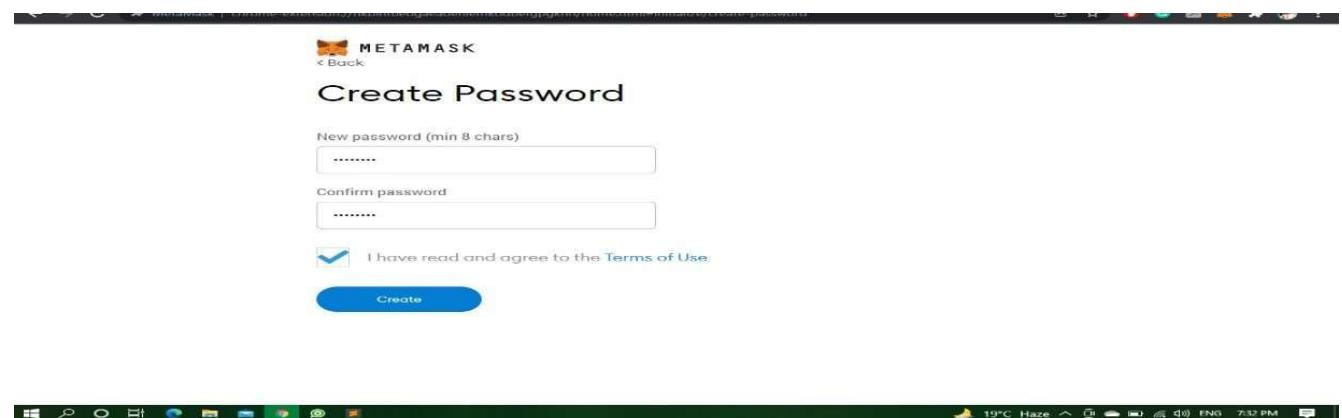
Step-5: Click on “Create a Wallet”.



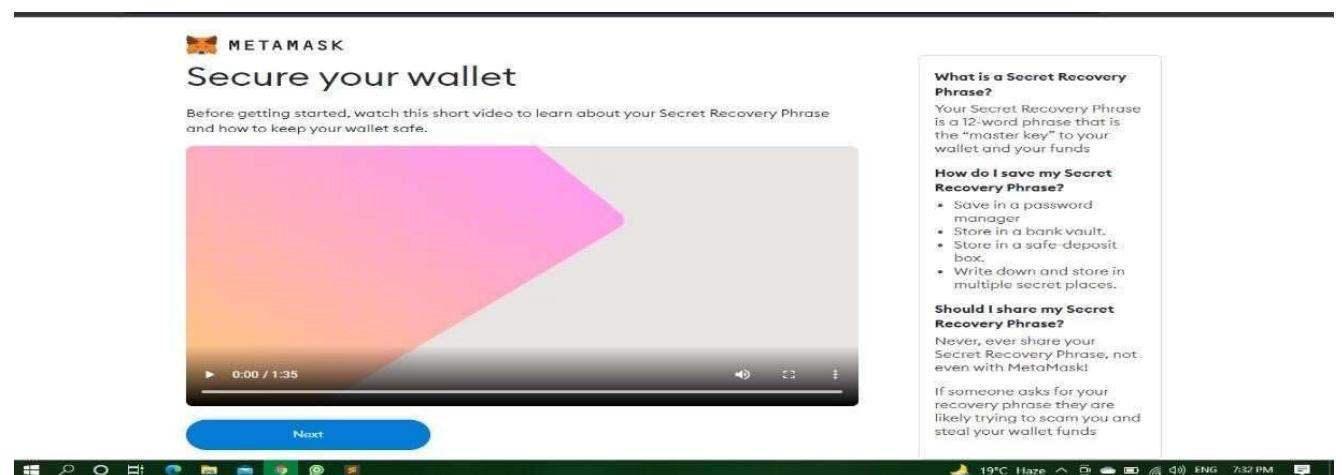
Step-6: Click on “I agree”.



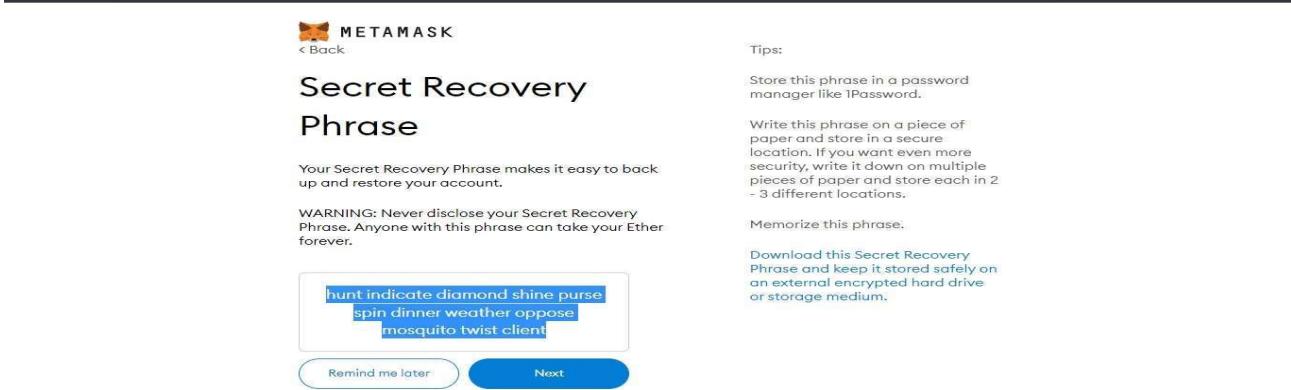
Step-7: now Create Password



Step-8: Click on "Next" tab.



Step-9: Reveal the Secret Words then copy it and paste it in any document file and click on "Next".



METAMASK
Back

Secret Recovery Phrase

Your Secret Recovery Phrase makes it easy to back up and restore your account.

WARNING: Never disclose your Secret Recovery Phrase. Anyone with this phrase can take your Ether forever.

hunt indicate diamond shine purse
spin dinner weather oppose
mosquito twist client

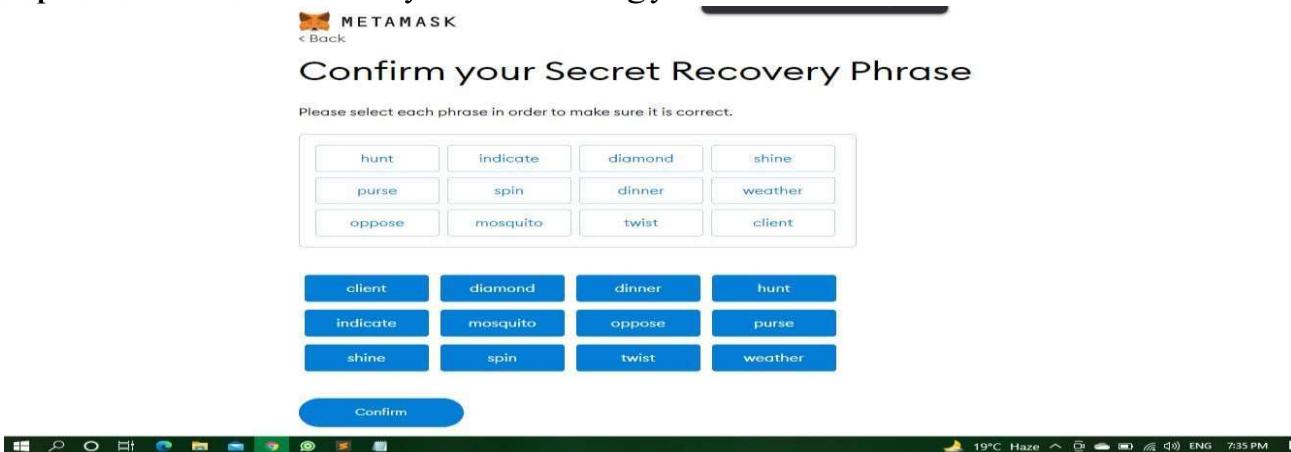
Remind me later Next

Tips:

- Store this phrase in a password manager like 1Password.
- Write this phrase on a piece of paper and store in a secure location. If you want even more security, write it down on multiple pieces of paper and store each in 2 - 3 different locations.
- Memorize this phrase.
- Download this Secret Recovery Phrase and keep it stored safely on an external encrypted hard drive or storage medium.



Step-10: Confirm Secret Recovery Phrase accordingly.



METAMASK
Back

Confirm your Secret Recovery Phrase

Please select each phrase in order to make sure it is correct.

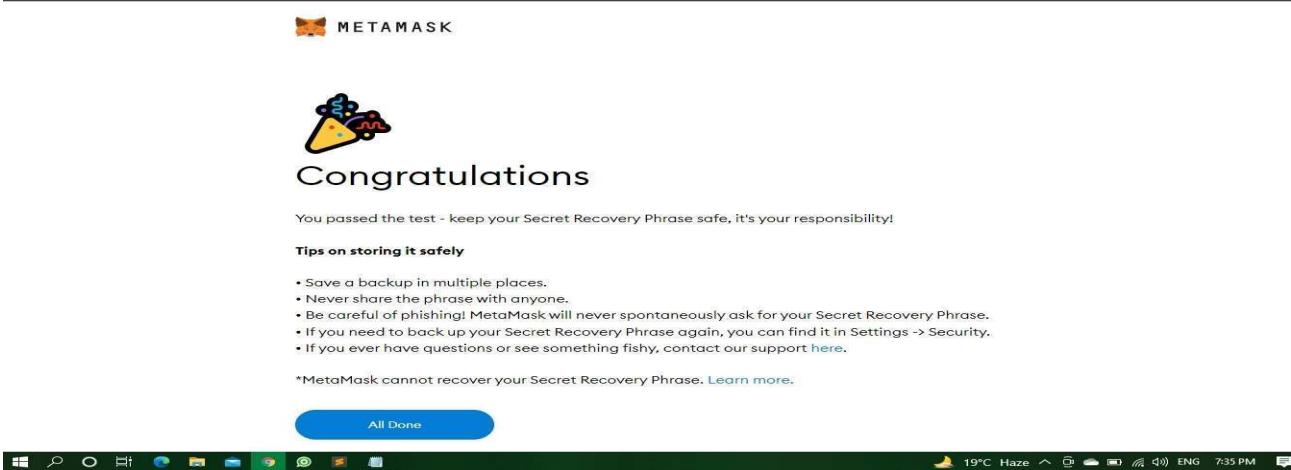
hunt	indicate	diamond	shine
purse	spin	dinner	weather
oppose	mosquito	twist	client

client	diamond	dinner	hunt
indicate	mosquito	oppose	purse
shine	spin	twist	weather

Confirm

19°C Haze ENG 7:35 PM

Step-11: After confirming the Secret Phrase simply click on “All Done”, and your account has been created.



METAMASK



Congratulations

You passed the test - keep your Secret Recovery Phrase safe, it's your responsibility!

Tips on storing it safely

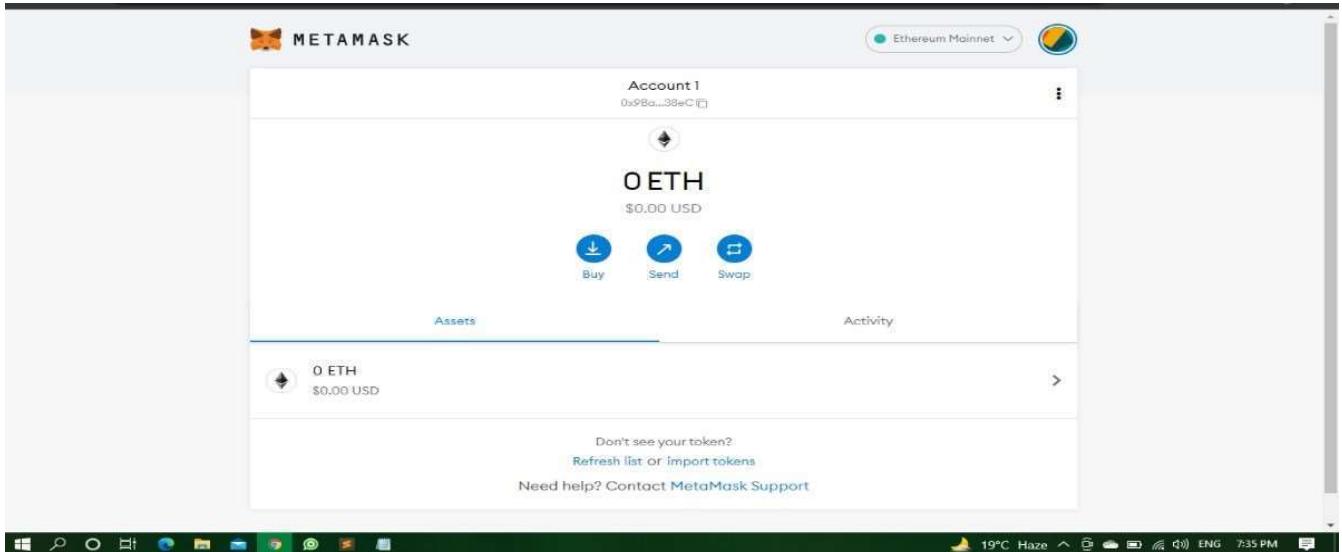
- Save a backup in multiple places.
- Never share the phrase with anyone.
- Be careful of phishing! MetaMask will never spontaneously ask for your Secret Recovery Phrase.
- If you need to back up your Secret Recovery Phrase again, you can find it in Settings → Security.
- If you ever have questions or see something fishy, contact our support [here](#).

*MetaMask cannot recover your Secret Recovery Phrase. [Learn more](#).

All Done

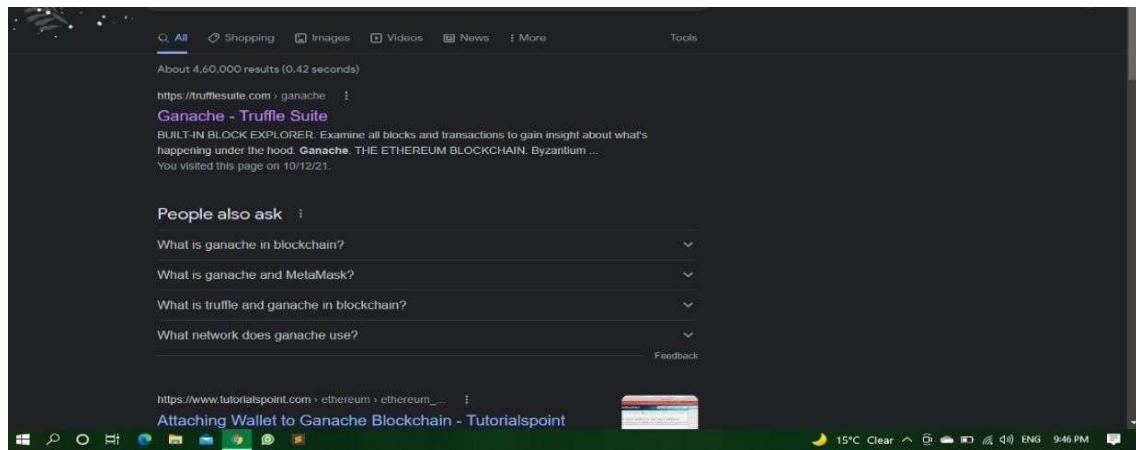
19°C Haze ENG 7:35 PM

Metamask Wallet will looks like.



Practical 2- How to install, use and operate the Ganache?

Step-1: Open your Chrome Browser and search for Ganache Wallet and Click on first link.



Step-2: Download Ganache for Windows.



Step-3: Open ganache and click on “Quickstart”



Step-4: Copy the “RPCServer” from Ganache and create a new Network in Metamask by using the RPC server.

Step-5: Click on Key symbol in Ganache and Copy the Private Key and Import a new account in Metamask by using this Private key, after click on “Import” you will see 100 ETH in your account.

Accounts Gas Limit Network ID RPC Server Mining Status Workspace

CURRENT BLOCK GAS PRICE: 20000000000 GAS LIMIT: 6721975 HARDORK: MUIRGLAICER NETWORK ID: 5777 RPC SERVER: HTTP://127.0.0.1:7545 MINING STATUS: AUTOMINING WORKSPACE: QUICKSTART SAVE SWITCH

MNEMONIC: tragic occur vapor pet hurdle express seek achieve heavy oyster primary scheme

ADDRESS: 0x81D48A732A6D97C0618D3A376E8E897D1F138dA6 **BALANCE:** 100.00 ETH TX COUNT: 0 INDEX: 0 **Show Ke**

ADDRESS: 0x0dE925a8CEA4E5Dea2D87b9e44aF2DB4d0231182 **BALANCE:** 100.00 ETH TX COUNT: 0 INDEX: 1 **Show Ke**

ADDRESS: 0xE9d78A71e1e81E8459Ac6f8e08B1F68160d8928C **BALANCE:** 100.00 ETH TX COUNT: 0 INDEX: 2 **Show Ke**

ADDRESS: 0xAdC287156Fe1a6b8B955Bc0d28CC6A51Cf255930 **BALANCE:** 100.00 ETH TX COUNT: 0 INDEX: 3 **Show Ke**

ADDRESS: 0xfc50Da28E7087121AFA2BacEf32be664457E3c0A **BALANCE:** 100.00 ETH TX COUNT: 0 INDEX: 4 **Show Ke**

HD PATH: m/44'/60'/0'/0/account_index

Buy Send Swap Import Account Connect Hardware Wallet Support -0 ETH Settings

"Contest" was successfully added! Help? Contact MetaMask Support

ACCOUNT INFORMATION

ACCOUNT ADDRESS: 0x81D48A732A6D97C0618D3A376E8E897D1F138dA6

PRIVATE KEY: 083c084bb27828d8b3ad4f629490a7007f34c859551512aa74977ece65a1251a
Do not use this private key on a public blockchain; use it for development purposes only!

DONE

Select Type: Private Key

Paste your private key string here: Import Cancel

HD PATH: m/44'/60'/0'/0/account_index

21°C Haze ENG 10:49 AM

ACCOUNT INFORMATION

ACCOUNT ADDRESS: 0x81D48A732A6D97C0618D3A376E8E897D1F138dA6

PRIVATE KEY: 083c084bb27828d8b3ad4f629490a7007f34c859551512aa74977ece65a1251a
Do not use this private key on a public blockchain; use it for development purposes only!

DONE

100 ETH

Buy Send Swap

Assets Activity

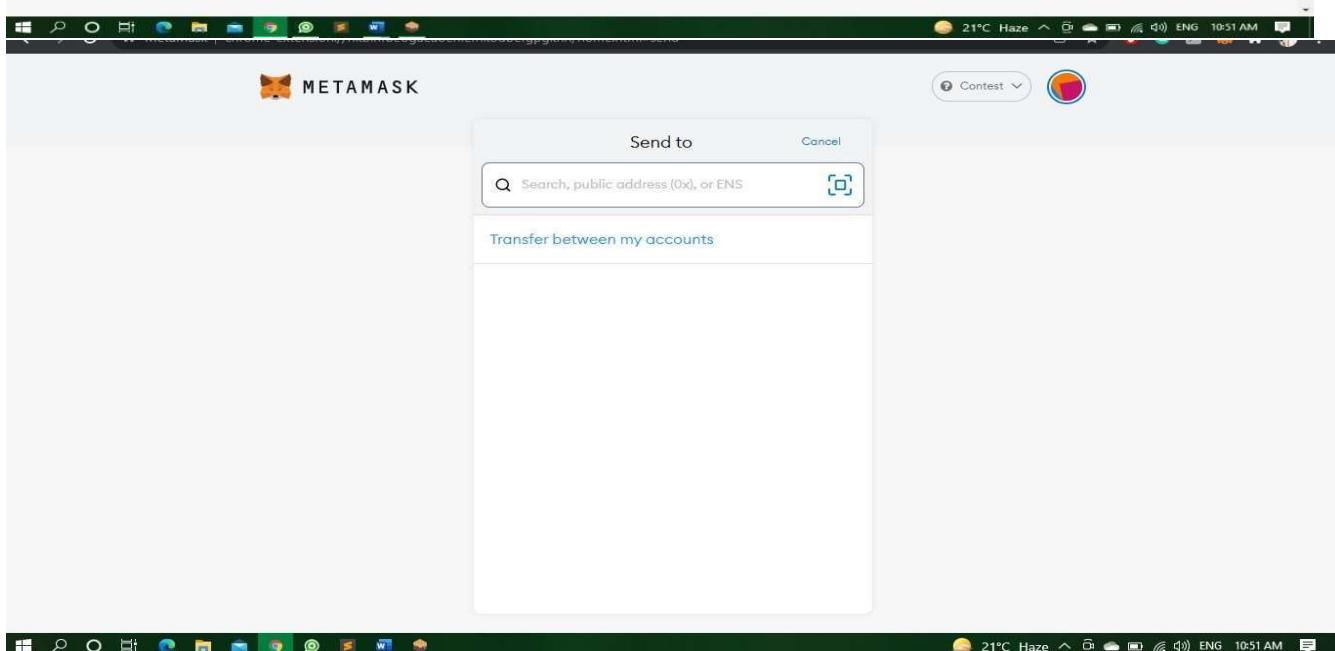
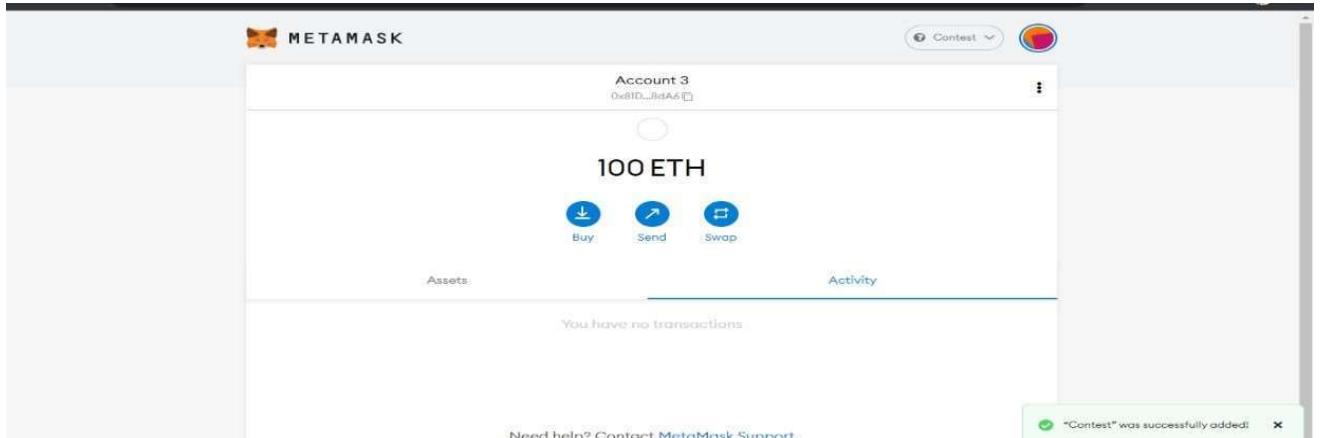
You have no transactions

"Contest" was successfully added! Need help? Contact MetaMask Support

HD PATH: m/44'/60'/0'/0/account_index

21°C Haze ENG 10:50 AM

Step-6: Now click on “Send” and transfer between account and choose an account to send some ether and confirm the transaction.



The image consists of three vertically stacked screenshots of the MetaMask wallet interface on a Windows desktop.

Screenshot 1: Send Transaction Step

This screenshot shows the "Send" dialog box. At the top, it says "Account 1" with the address "0x9ba578d574bcb8756db923532c094c8efd8238ec". Below that, "Asset:" is set to "100 ETH" with "Balance: 100 ETH". The "Amount:" field contains "15 ETH" with "Max" button. Under "Gas Price (GWEI)", the value is "20" with a dropdown menu showing "21000". Buttons at the bottom are "Cancel" and "Next".

Screenshot 2: Transaction Confirmation Step

This screenshot shows the "Edit" dialog box for a transaction from "Account 3" to "Account 1". It displays the amount "15" and the gas fee "0.00042 ETH". The total amount is "15.00042 ETH". Buttons at the bottom are "Reject" and "Confirm".

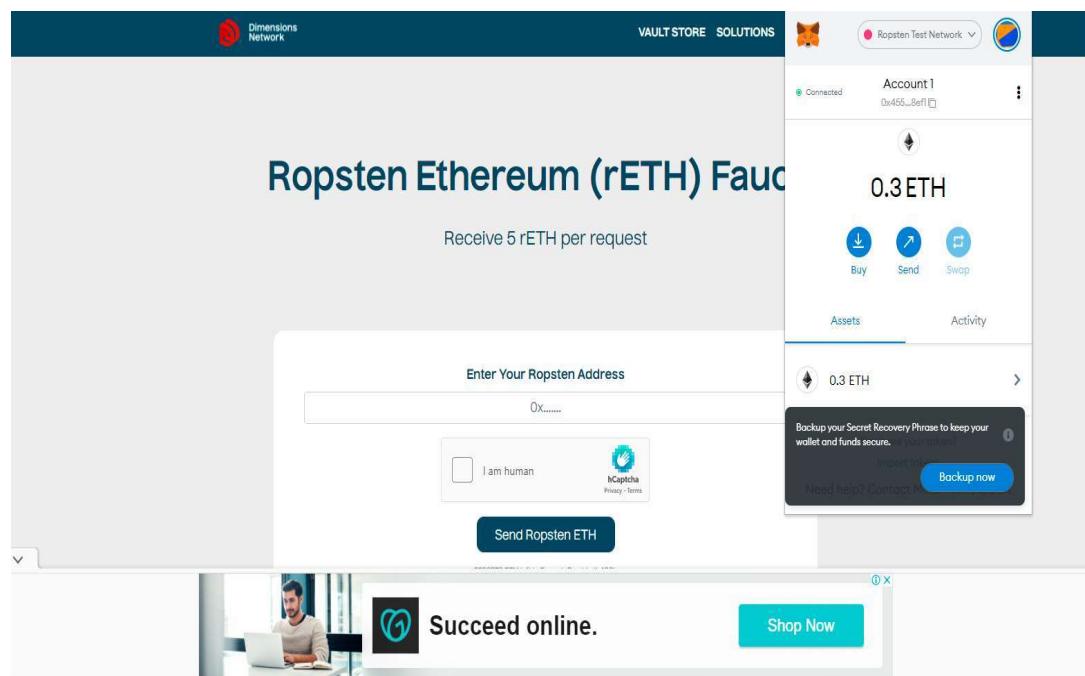
Screenshot 3: Transaction Confirmation and Activity

This screenshot shows the main MetaMask interface. At the top, it says "Account 3" with the address "0x81D...8dA6". Below that, the balance is "84.9996 ETH". There are three buttons: "Buy", "Send", and "Swap". The "Activity" tab is selected, showing a recent transaction: "Send" on Dec 22 to "0x9ba...38ec" with "-15 ETH". A message at the bottom says "Need help? Contact MetaMask Support". A green notification bar at the bottom right says "Google Chrome Confirmed transaction Transaction 0 confirmed!"

Transaction successfully completed

PRACTICAL -3

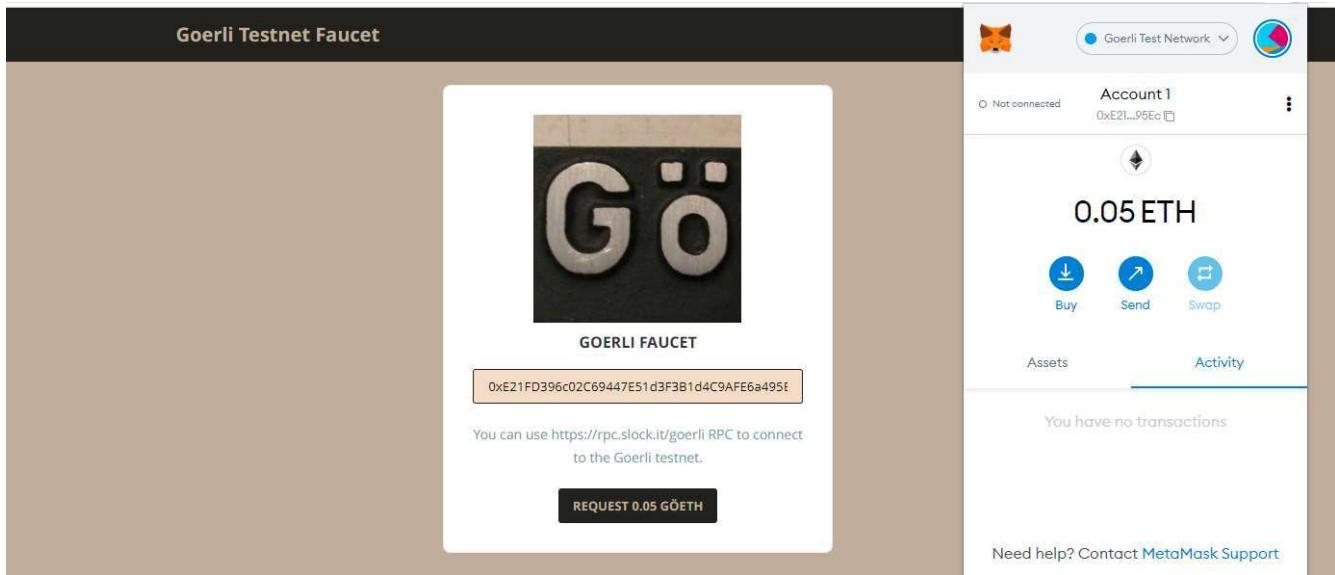
- ⑨ Firstly open your wallet and select ropston network .After that you click on buy ETH and then



click on get ether

- ⑨

→ AFTER ropston network ,open goerli network and click on buy button and after that you click on get ether



Practical No.5 Enable/Add the solidity compiler in Notepad++

Step-1: Download and Install Notepad++ .

Downloads

[Download Notepad++ v8.2](#)

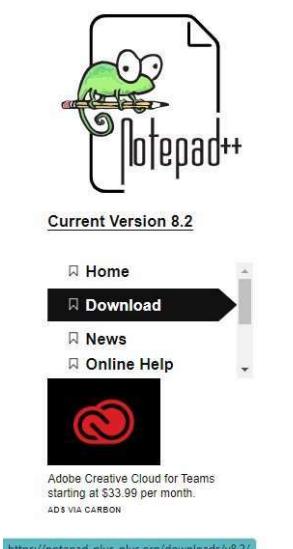
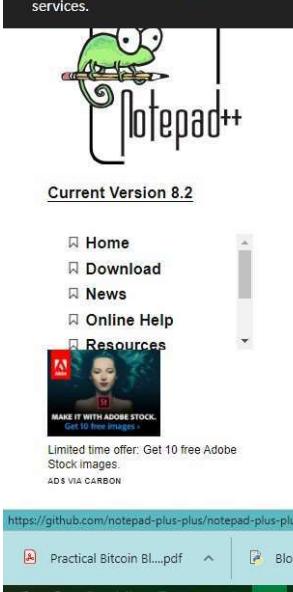
[Download Notepad++ v8.1.9.3](#)

[Download Notepad++ v8.1.9.2](#)

[Download Notepad++ v8.1.9.1](#)

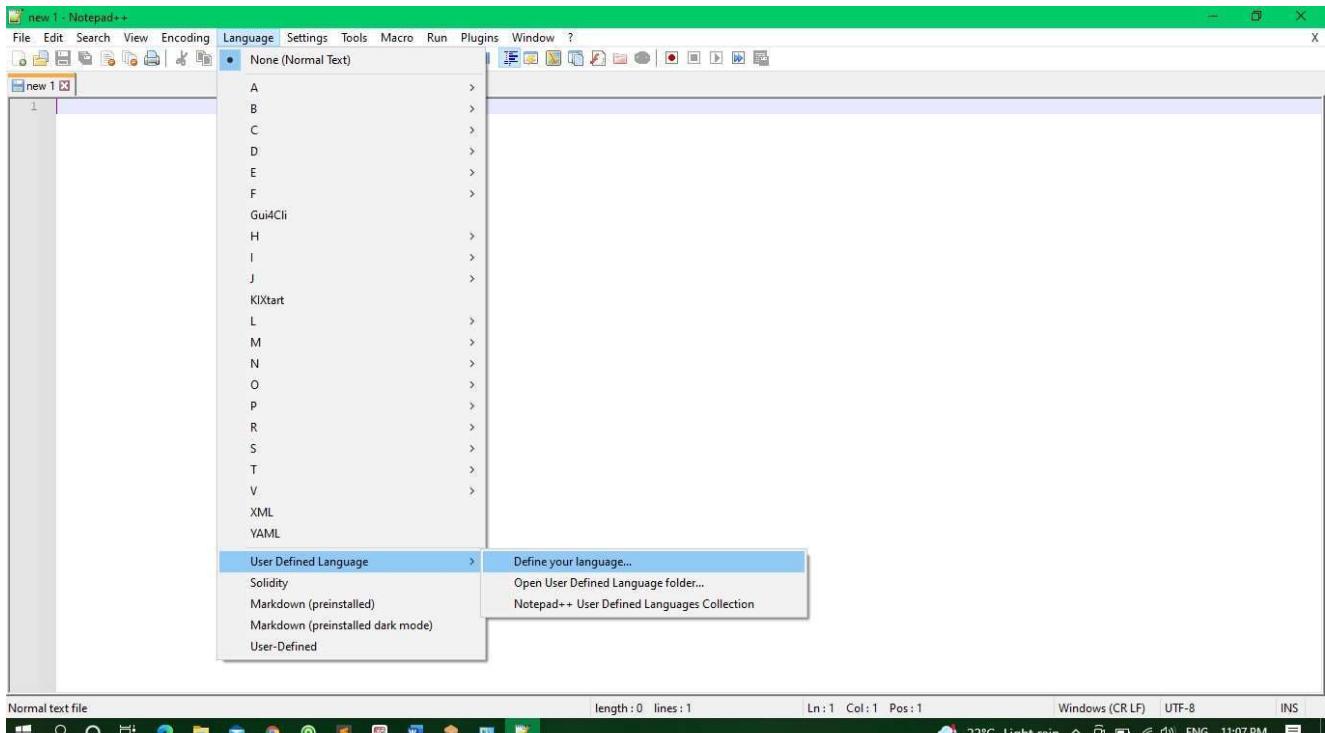
[Download Notepad++ v8.1.9](#)

[Download Notepad++ v8.1.8](#)

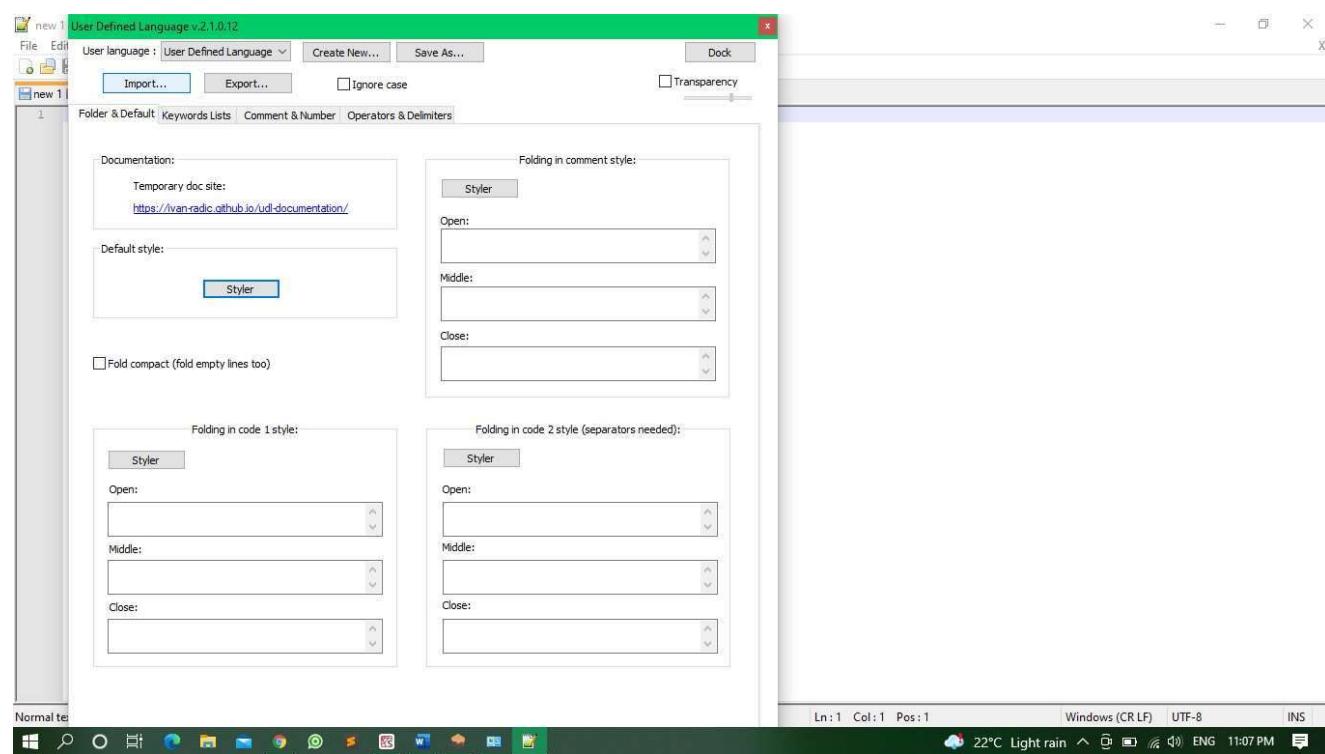



Step-2: Now open Notepad++

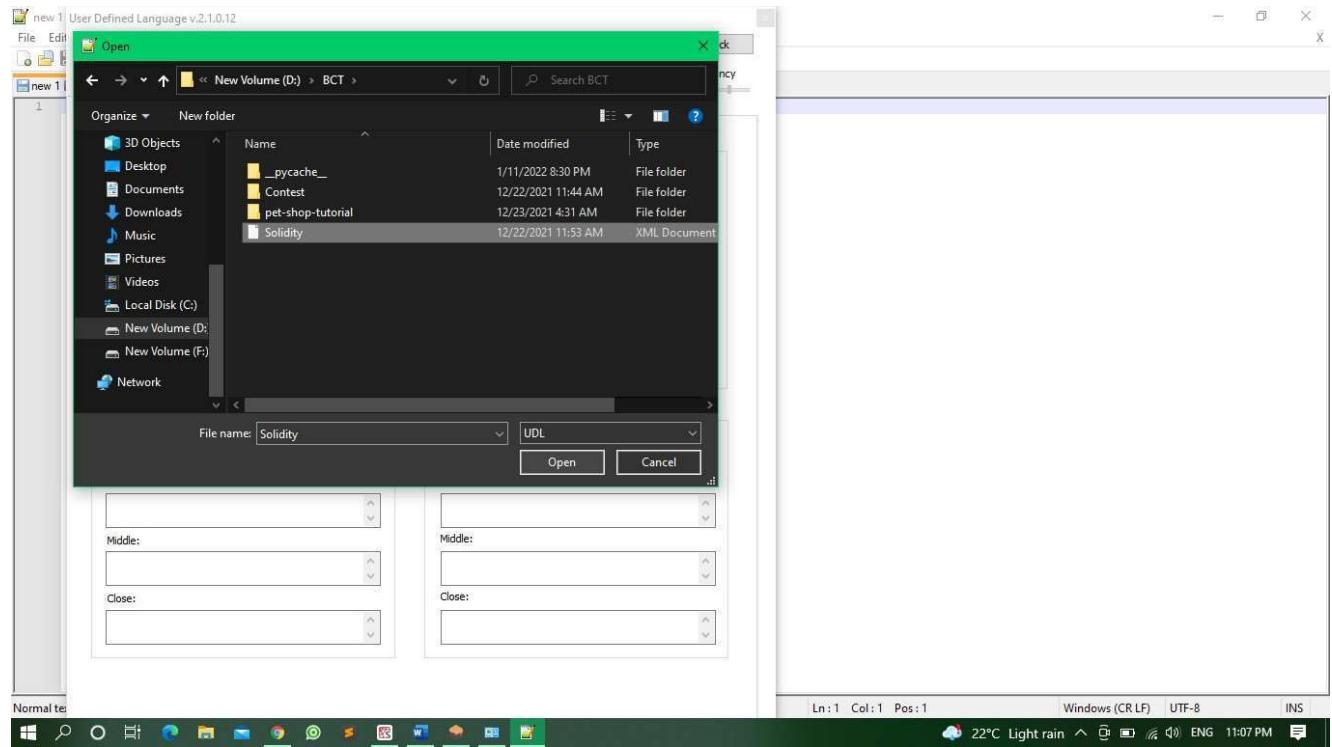
Step-3: Click on language >> User defined language >>define your language



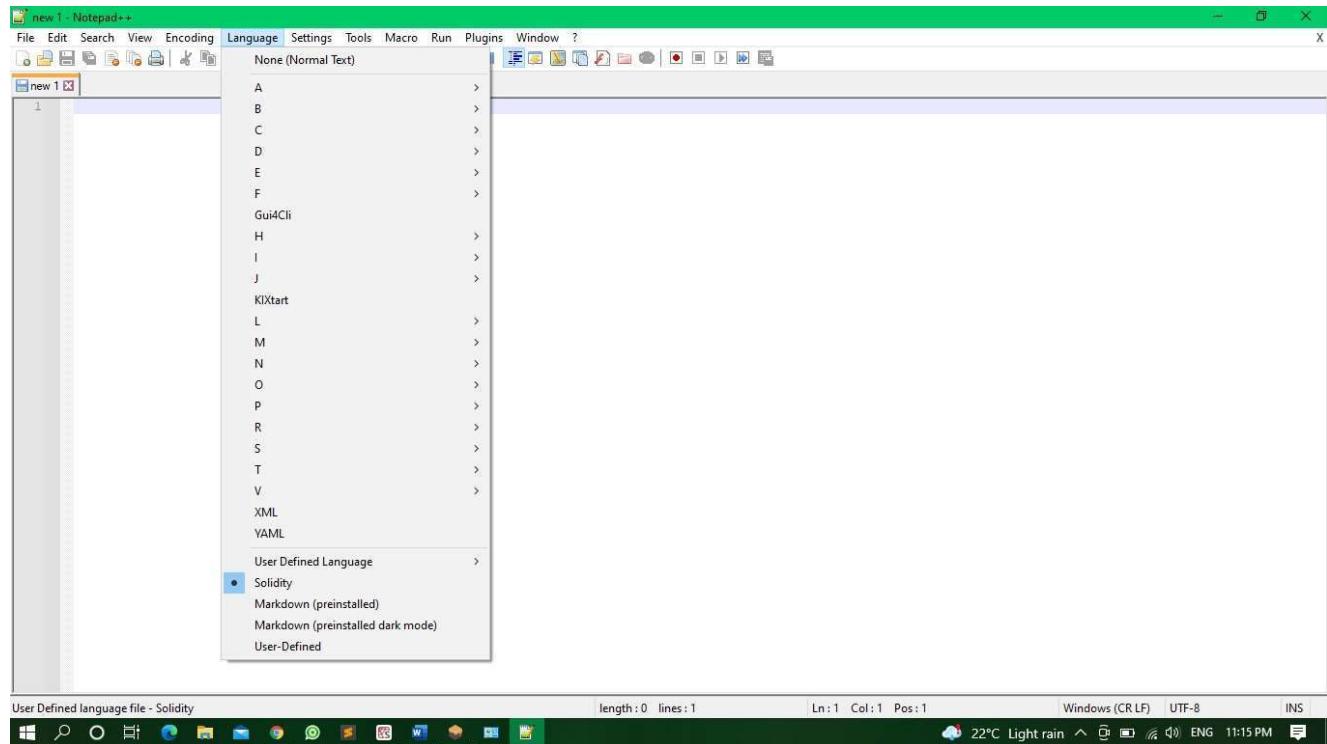
Step-4: Click on "Import"



Step-5: Click on BCT >> Solidity and open it



Step-6: Now you can see Solidity has been added in Notepad++.

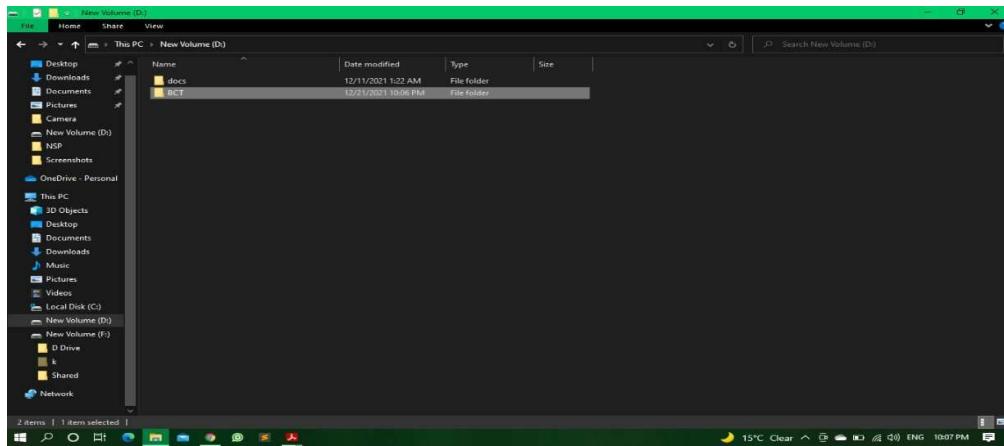


Practical No.6 E-Voting (Best Actor Contest)

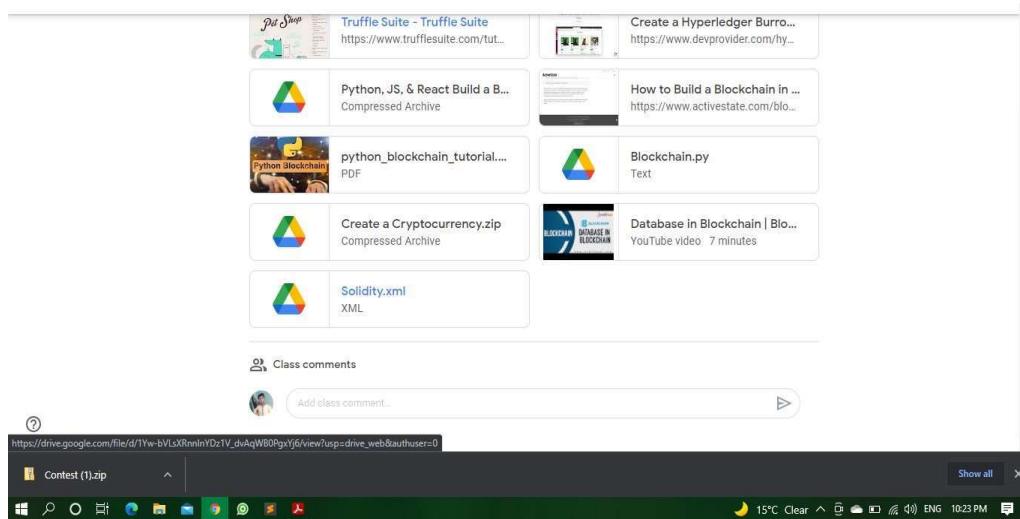
Step-1: Open Google classroom of Blockchain Technology open Practical folder and download "Contest" file and extract it into a new folder called "BCT".

The image consists of three vertically stacked screenshots of a Windows desktop environment, likely Windows 10, illustrating the process of extracting a ZIP file.

- Screenshot 1:** Shows a Google Classroom Stream for the "Blockchain Technology" class. It displays several posts from user "Khaleel Ahmad" and one post from "Mohammad Mubeen". One of Khaleel's posts includes a link to a Google Drive folder named "Practical".
- Screenshot 2:** Shows the "Practical" folder contents on Google Drive. It contains various files and folders related to blockchain technology, including "Contest.zip", "pet-shop-tutorial.zip", "Truffle Suite - Truffle Suite", "Create a Hyperledger Burro...", "Python, JS, & React Build a B...", "How to Build a Blockchain in ...", "Blockchain.py", "Create a Cryptocurrency.zip", and "Database in Blockchain | Blo...".
- Screenshot 3:** Shows the "Contest.zip" file being extracted. A progress bar indicates the extraction of "Contest" from the ZIP file. The extraction is completed, and the "Contest" folder is visible on the desktop.



Step-2: Download Notepad++, Solidity, NodeJS and Git from google classroom by clicking on truffle suite and add Solidity in Notepad++ .



Step-3: Open cmd and run “[npm install -g truffle](#)” command.

After installation simply copy the BCT folder path and open it in cmd and run following command one by one.

```
>>Truffle compile  
>>Truffle migrate  
>>Truffle test  
>>Npm run dev
```

After this there will be open a Best Actor Contest web page .

Now open metamask and connect this web page to an account which you have already imported from Ganache.

And Caste your vote by using some Ethereum.

```

Administrator: Command Prompt
lil receive updates in the future
npm WARN deprecated multicodec@2.1.3: This module has been superseded by the multiformats module
npm WARN deprecated multiformats@2.1.3: This module has been superseded by the multiformats module
npm WARN deprecated cidsp@0.7.5: This module has been superseded by the multiformats module
npm WARN deprecated graphql-tools@4.0.8: This package has been deprecated and now it only exports makeExecutableSchema. You can migrate to scoped packages such as @graphql-tools/schema, @graphql-tools/utils and etc. In check out https://www.graphql-tools.com to learn what package you should use instead
npm WARN deprecated core-js@2.6.12: core-js@<3 is no longer maintained and not recommended for usage due to the number of issues. Because of the V8 engine whims, feature detection in old core-js versions could cause a slowdown up to 100x even if nothing is polyfilled. Please, upgrade your dependencies to the actual version of core-js.
added 26 packages, removed 112 packages, changed 946 packages, and audited 955 packages in 3m
87 packages are looking for funding
  run `npm fund` for details
58 vulnerabilities (7 low, 33 moderate, 9 high, 7 critical)
To address issues that do not require attention, run:
  npm audit fix
To address all issues possible, run:
  npm audit fix --force
Some issues need review, and may require choosing
a different dependency.
Run 'npm audit' for details.
npm notice New minor version of npm available! 8.1.2 -> 8.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v8.3.0
npm notice Run npm install -g npm@8.3.0 to update!
npm notice

C:\Users\Qasim>
C:\Users\Qasim> -

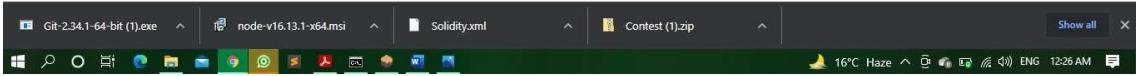
```

Best Actor Contest

Your Account: null

#	Name	Votes
1	Omy	0
2	Jammy	0

Select Contestant



Best Actor Contest

Your Account: 0xc62b307b455f94d5a2a63962611fcda76fcada7

#	Name	Votes
1	Omy	0
2	Jammy	0

Select Contestant

Account 2
0xc62...ADa

99.9715 FTH

Connected sites

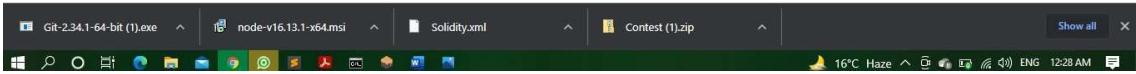
Account 2 is not connected to any sites.

Manually connect to current site

99.9715 ETH

Don't see your token?
Import tokens

Need help? Contact MetaMask Support



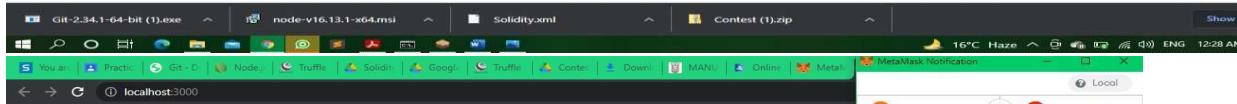
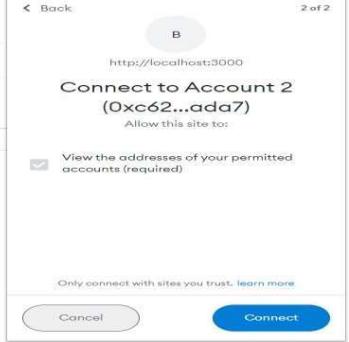


Best Actor Contest

Your Account: 0xc62b307b455f94d5a2a63962611fcda76fcada7

#	Name	Votes
1	Omy	0
2	Jammy	0

Select Contestant

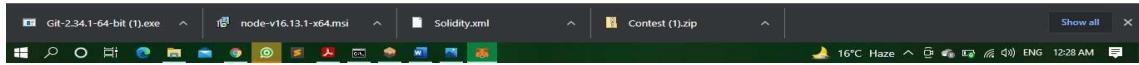
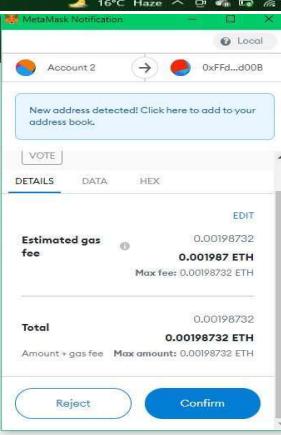


Best Actor Contest

Your Account: 0xc62b307b455f94d5a2a63962611fcda76fcada7

#	Name	Votes
1	Omy	0
2	Jammy	0

Select Contestant

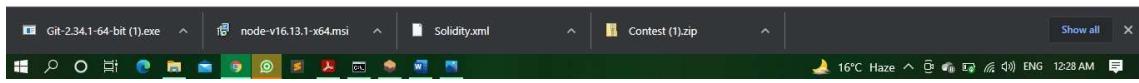


Best Actor Contest

Your Account: 0xc62b307b455f94d5a2a63962611fcda76fcada7

#	Name	Votes
1	Omy	1
2	Jammy	0

Select Contestant



Your E-voting has been done Successfully.

Practical No.7

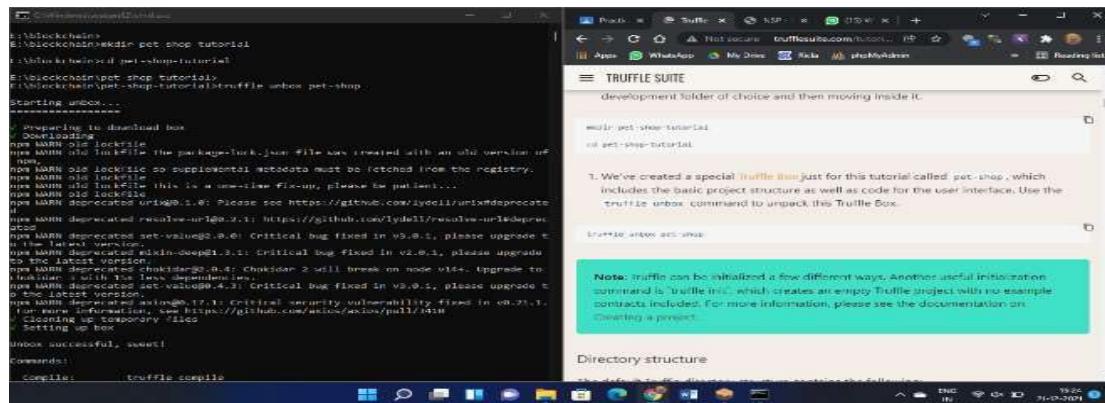
Pet-shop Tutorial

Step-1: Open cmd and run the following commands one by one,

```
>> mkdir pet-shop-tutorial
```

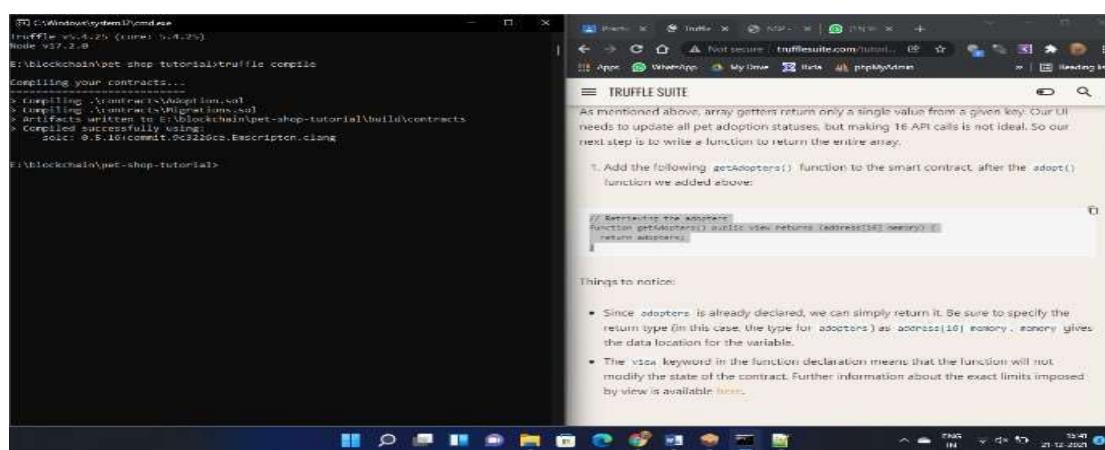
```
>> cd pet-shop-tutorial
```

```
>> truffle unbox pet-shop
```



```
C:\Windows\system32\cmd.exe
E:\blockchain>mkdir pet-shop-tutorial
E:\blockchain>cd pet-shop-tutorial
E:\blockchain\pet-shop-tutorial>
E:\blockchain\pet-shop-tutorial>truffle unbox pet-shop
Starting up...
-----
Preparing to download box...
Downloading...
Download complete!
The package-lock.json file was created with an old version of node. We're going to fix it.
node: warning: node-lockfile: no supplemental methods must be fetched from the registry.
node: warning: node-lockfile: no supplemental methods must be fetched from the registry.
node: warning: node-lockfile: This is a one-time fix-up, please be patient...
node: warning: node-lockfile: deprecated url=git@github.com:lydell/re-ecore@0.1.0: Please see https://github.com/lydell/re-ecore#deprecate
node: warning: node-lockfile: deprecated url=git@github.com:lydell/re-ecore@0.1.0: https://github.com/lydell/re-ecore#deprecate
node: warning: node-lockfile: deprecated set-valve@0.0.1: Critical bug fixed in v0.0.1, please upgrade to v0.0.2 or later.
node: warning: node-lockfile: deprecated set-valve@0.0.1: Critical bug fixed in v0.0.1, please upgrade to v0.0.2 or later.
node: warning: node-lockfile: deprecated set-valve@0.4.3: Critical bug fixed in v0.4.3, please upgrade to v0.4.4 or later.
node: warning: node-lockfile: deprecated set-valve@0.4.3: Critical bug fixed in v0.4.3, please upgrade to v0.4.4 or later.
node: warning: node-lockfile: deprecated set-valve@0.4.3: Critical security vulnerability fixed in v0.4.3.
node: warning: node-lockfile: deprecated set-valve@0.4.3: https://github.com/zeit/zeit/pull/454
Closing up temporary files...
Setting up box...
Unbox successful, sweet!
Commands:
  compiler: truffle compile
```

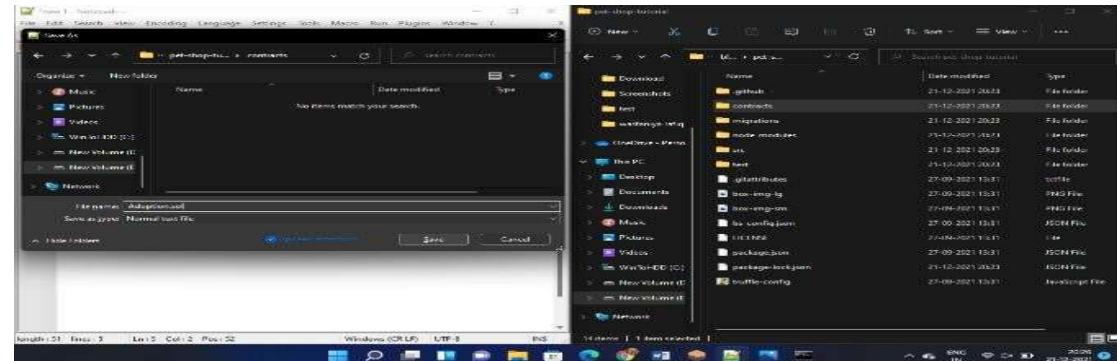
The Truffle Suite page displays the 'pet-shop' tutorial, which includes a note about Truffle's initialization command and a 'Directory structure' section.

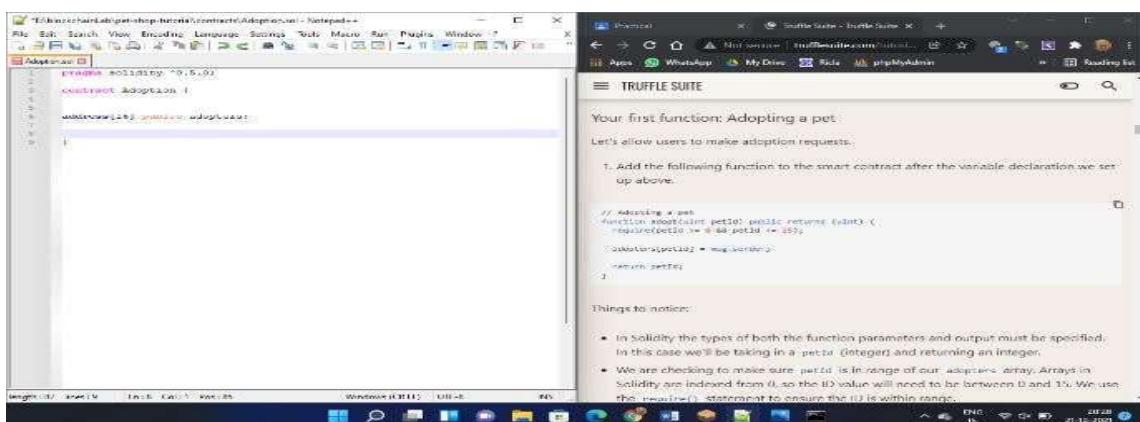


```
E:\blockchain>cd pet-shop-tutorial
E:\blockchain\pet-shop-tutorial>truffle compile
Compiling your contracts...
> Compiling ./contracts/Adoption.sol
> Compiling ./contracts/Vigilante.sol
> Compiling ./contracts/BlockchainPetShopTutorial/build/contracts
> Compiled successfully using:
  solc: 0.5.16+commit.0c3220cs.Emscripten clang
```

The Truffle Suite page displays the 'pet-shop' tutorial, which includes a note about array getters returning single values, a code snippet for the 'getAdopters()' function, and a 'Things to notice:' section with two bullet points.

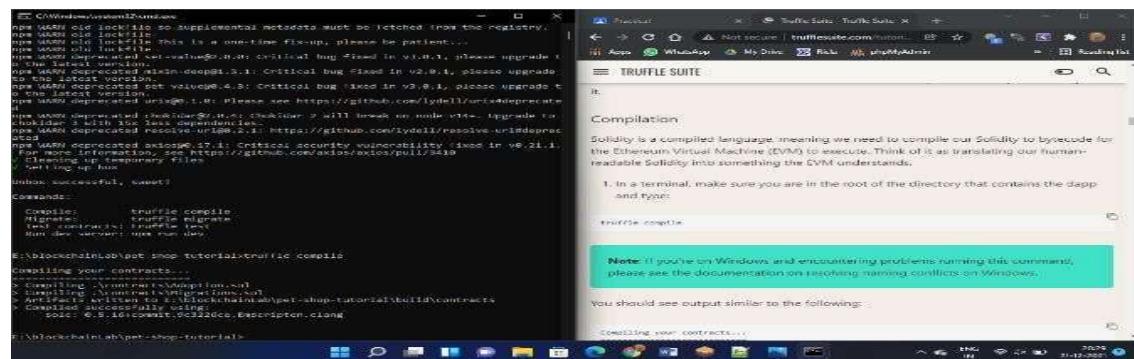
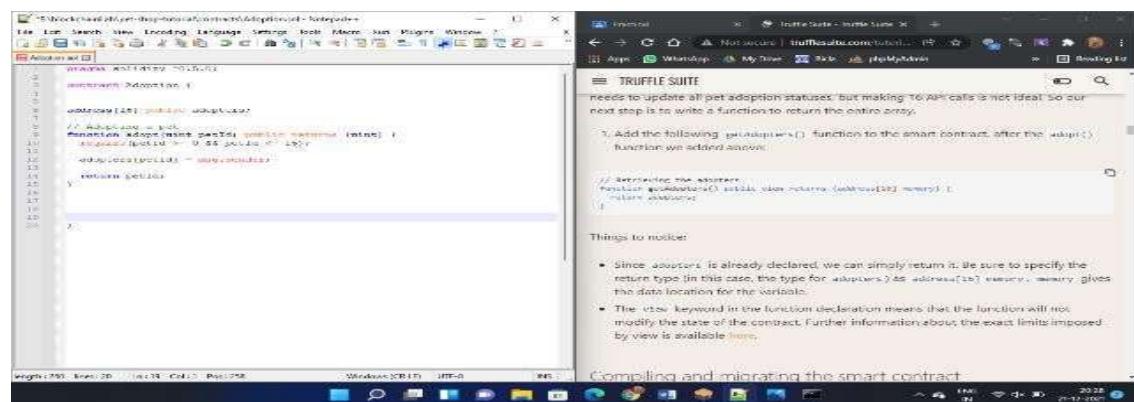
Step-2: Create a new file named Adoption.sol in the contracts/ directory, and add the following files.



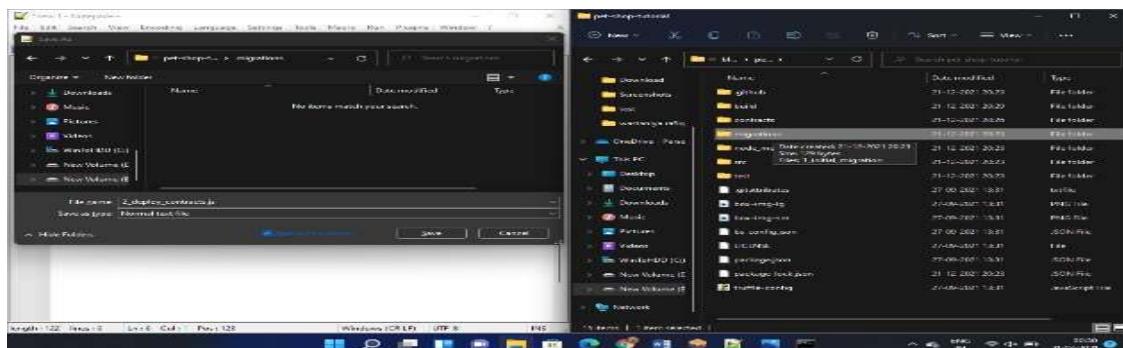


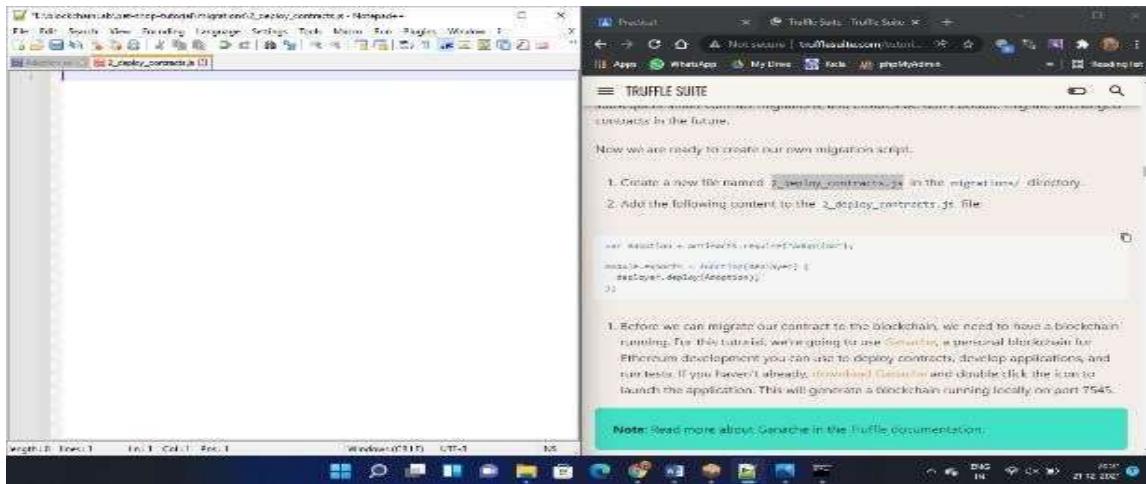
Step-3: Copy code and paste and run this command in

cmd >> truffle compile



Step-4: Create a new file named `2_deploy_contracts.js` in `migrations/` directory and write the codes.





Step-5: Run this command in cmd-

>> truffle migrate

```

E:\blockchain\abp\shop-tutorial\truffle> migrate
Compiling your contracts...
> Compiling ./contracts/Adoption.sol
> Compiling ./contracts/Migrations.sol
> Artifacts written to E:\blockchain\abp\shop-tutorial\build\contracts
> Compiled successfully using:
  - solc: 0.5.16+commit.0ec2a2cc.enscripten.clang

E:\blockchain\abp\shop-tutorial\truffle> migrate
Compiling your contracts...
> Compiling ./contracts/Adoption.sol
> Compiling ./contracts\Migrations.sol
> Artifacts written to E:\blockchain\abp\shop-tutorial\build\contracts
> Compiled successfully using:
  - solc: 0.5.16+commit.0ec2a2cc.enscripten.clang

Starting migrations...
> Network name: "development"
> Network id: 1337
> Block gas limit: 6720000 (maximator)

1. Initial migration.js

  Deploying 'Migration'
    > transaction hash: 0xb5905e195e2eccc61e70da20f86200249fae497cfbbdd3a2fcfccc01d0da
    > block: 8
    > contract address: 0xabc57057c01334c13a5117c58b609c6450002
    > block number: 34
    > block timestamp: 1464699018
    > balance: 99.04090038
    > gas used: 101.048 (0x26c7)
    > gas price: 0 gwei
    > value sent: 0 ETH

  1. Migrating 'Adoption'
    > transaction hash: 0xb58188764932a05f013695f5004f4af217170cf4bae9575bf1988a11252
    > block: 9
    > contract address: 0xcE20f8803857370920C15eE48Fc35EbF18Cc0f8
    > block number: 35
    > block timestamp: 1464699018
    > account: 0x0771dF0730f120890709b005880c83370C30c0d0
    > balance: 99.04081502
    > gas used: 60500 (0x18522)
    > gas price: 0 gwei
    > value sent: 0 ETH
    > total costs: 0.001337 ETH

  > Migrating 'Adoption'
    > transaction hash: 0xb58188764932a05f013695f5004f4af217170cf4bae9575bf1988a11252
    > block: 9
    > contract address: 0xcE20f8803857370920C15eE48Fc35EbF18Cc0f8
    > block number: 35
    > block timestamp: 1464699018
    > account: 0x0771dF0730f120890709b005880c83370C30c0d0
    > balance: 99.04081502
    > gas used: 60500 (0x18522)
    > gas price: 0 gwei
    > value sent: 0 ETH
    > total costs: 0.001337 ETH

Summary:
> Total deployments: 2
> Final cost: 0.0017580 ETH

```

```

E:\blockchain\abp\shop-tutorial> migrate
Compiling your contracts...
> Compiling ./contracts/Adoption.sol
> Compiling ./contracts\Migrations.sol
> Artifacts written to E:\blockchain\abp\shop-tutorial\build\contracts
> Compiled successfully using:
  - solc: 0.5.16+commit.0ec2a2cc.enscripten.clang

Starting migrations...
> Network name: "development"
> Network id: 1337
> Block gas limit: 6720000 (maximator)

1. Initial migration.js

  Deploying 'Migration'
    > transaction hash: 0xb5905e195e2eccc61e70da20f86200249fae497cfbbdd3a2fcfccc01d0da
    > block: 8
    > contract address: 0xabc57057c01334c13a5117c58b609c6450002
    > block number: 34
    > block timestamp: 1464699018
    > balance: 99.04090038
    > gas used: 101.048 (0x26c7)
    > gas price: 0 gwei
    > value sent: 0 ETH

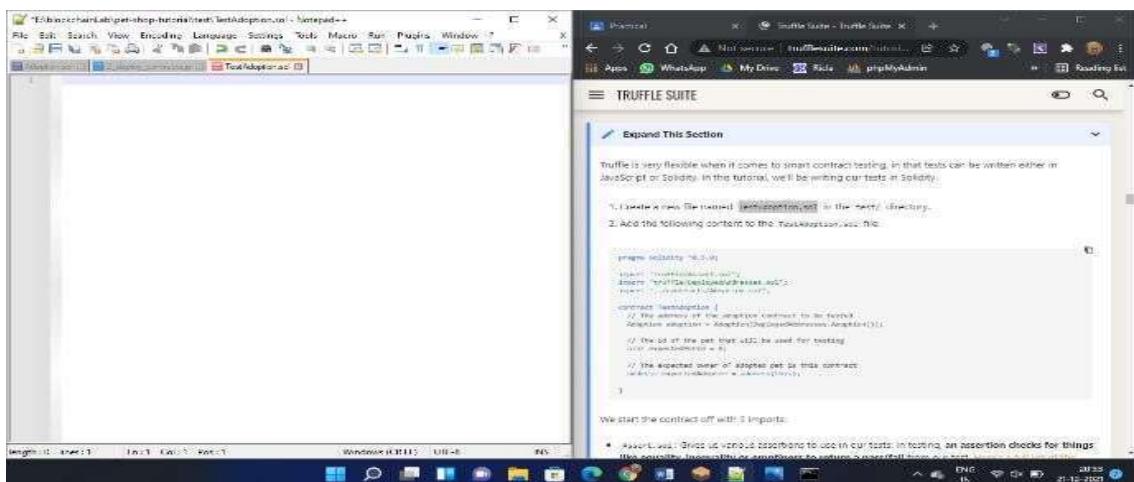
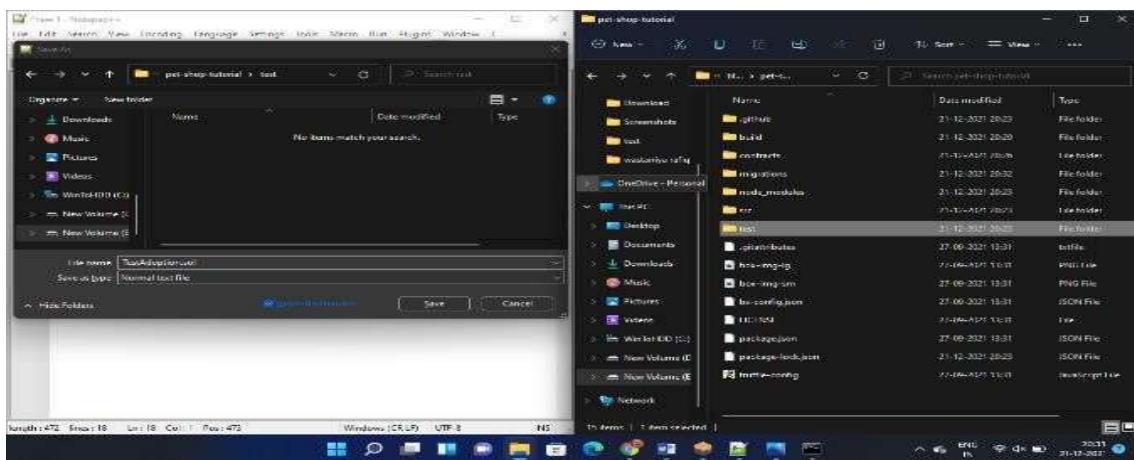
  1. Migrating 'Adoption'
    > transaction hash: 0xb58188764932a05f013695f5004f4af217170cf4bae9575bf1988a11252
    > block: 9
    > contract address: 0xcE20f8803857370920C15eE48Fc35EbF18Cc0f8
    > block number: 35
    > block timestamp: 1464699018
    > account: 0x0771dF0730f120890709b005880c83370C30c0d0
    > balance: 99.04081502
    > gas used: 60500 (0x18522)
    > gas price: 0 gwei
    > value sent: 0 ETH
    > total costs: 0.001337 ETH

  > Migrating 'Adoption'
    > transaction hash: 0xb58188764932a05f013695f5004f4af217170cf4bae9575bf1988a11252
    > block: 9
    > contract address: 0xcE20f8803857370920C15eE48Fc35EbF18Cc0f8
    > block number: 35
    > block timestamp: 1464699018
    > account: 0x0771dF0730f120890709b005880c83370C30c0d0
    > balance: 99.04081502
    > gas used: 60500 (0x18522)
    > gas price: 0 gwei
    > value sent: 0 ETH
    > total costs: 0.001337 ETH

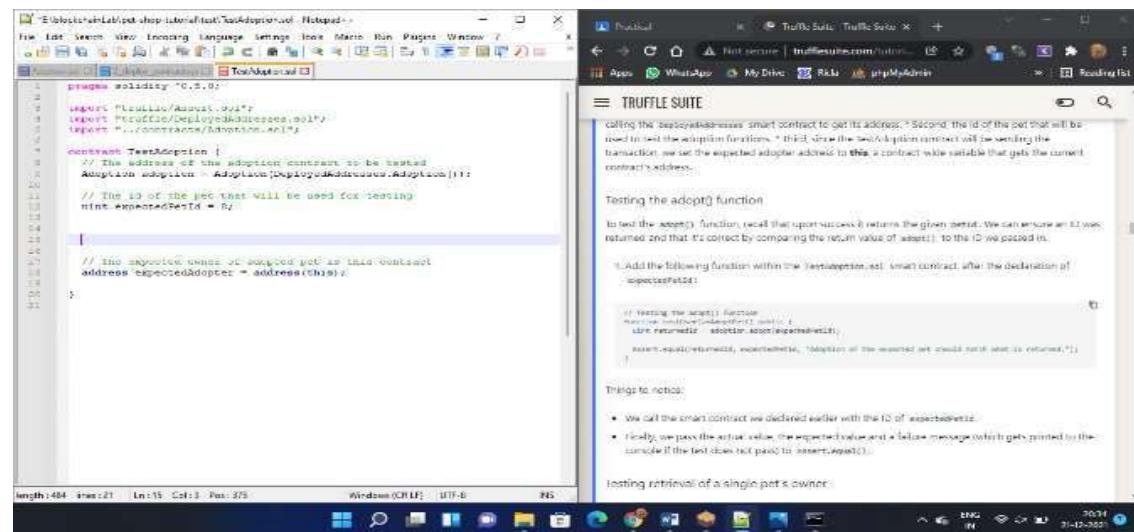
Summary:
> Total deployments: 2
> Final cost: 0.0017580 ETH

```

Step-6: Create a new file named TestAdoption.sol in test/ directory and write the code



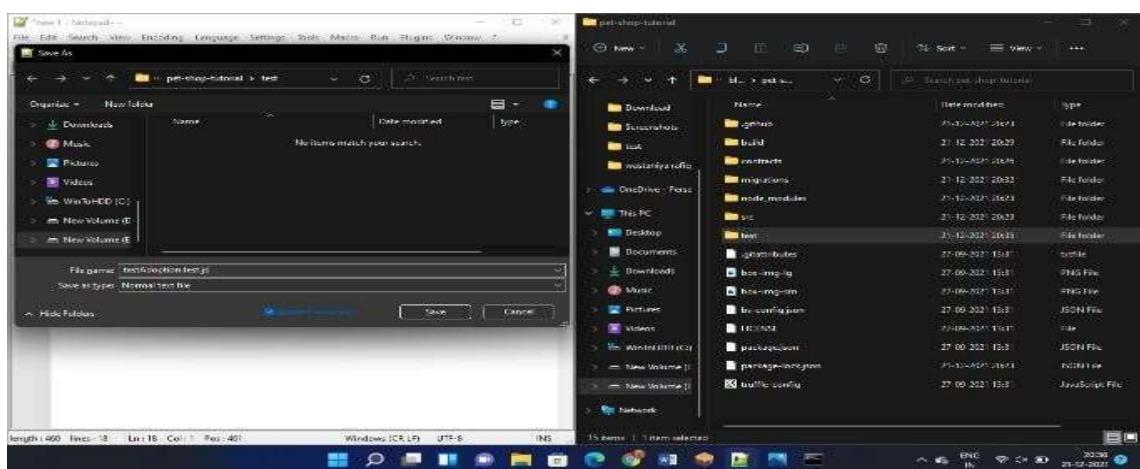
Step7: Add the following function within the `TestAdoption` after declaration of `expectedId` and following function previously added function

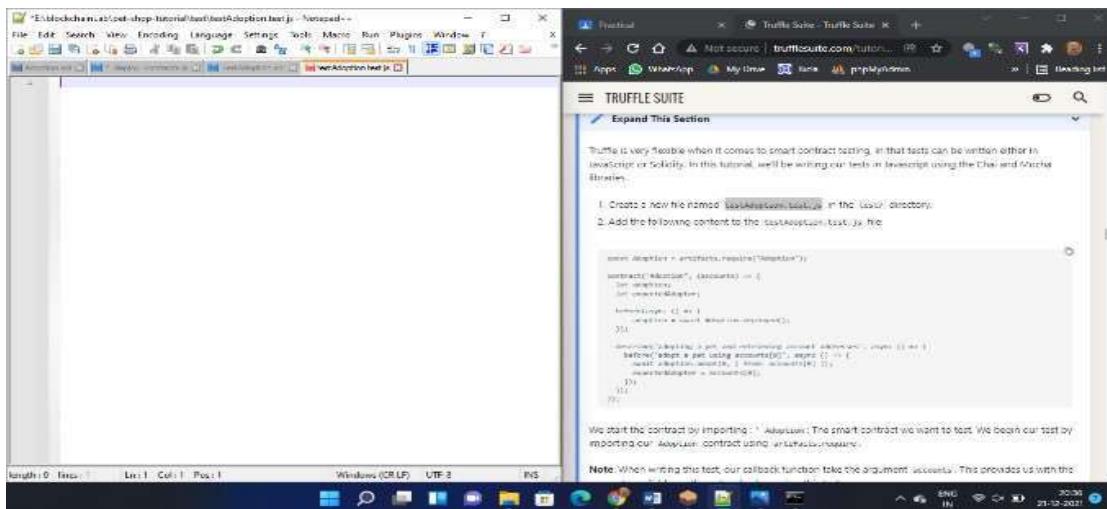


The screenshot shows a Windows desktop environment. On the left, a Notepad++ window displays Solidity code for a test contract named `TestAdoption.sol`. The code includes imports for `truffle/Assert.sol`, `truffle/DeployedAddresses.sol`, and `../contracts/Adoption.sol`. It contains a `TestAdoption` contract with a `testUserCanAdoptPet()` function and a `testGetAdopterAddressByPetId()` function. The `testGetAdopterAddressByPetId()` function uses `assertEqual` to check if the returned address matches the expected address. On the right, a Truffle Suite browser window is open, showing the Truffle Suite interface with documentation for the `testGetAdopterAddressByPetId()` function.

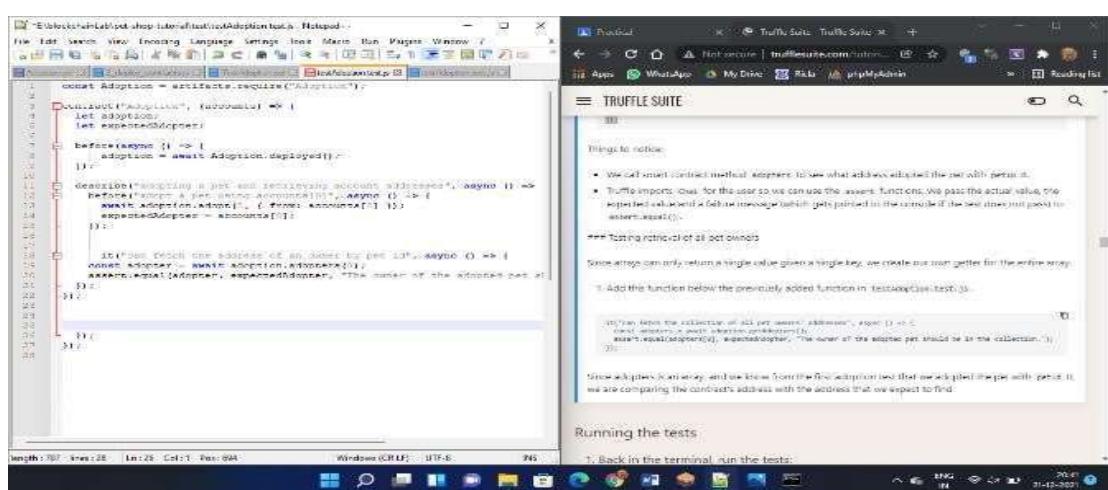
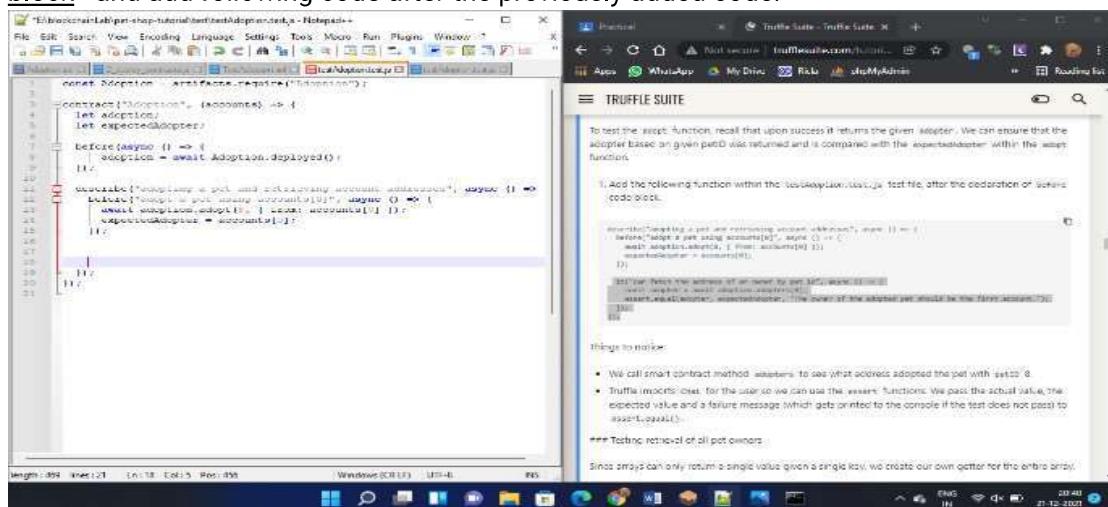
This screenshot is similar to the one above, showing the same Notepad++ window with the `TestAdoption.sol` code. The Truffle Suite browser window on the right now shows the documentation for the `testGetAdopterAddressByPetId()` function, which discusses the retrieval of a single pet's owner using a memory array.

Step-8: Create a new file named `testAdoption.test.js` in the `test/` directory and add following code.





Step-9: Add the following code within the testAdoption.test.js test file after the declaration of before code block and add following code after the previously added code.



Step-10: Run this command in cmd-

>> truffle test

The terminal window shows the command "truffle test" being run, outputting deployment logs for contracts "Adoption" and "PetShop". The browser window displays the Truffle Suite interface with the title "Running the tests". It includes a step-by-step guide: 1. Back in the terminal, run the tests: "truffle test". 2. If all the tests pass, you'll see console output similar to this: "Using network 'development'. Compiling your contracts...". 3. The test results show 3 passing tests (670ms). The status bar at the bottom right indicates the date and time: 21-12-2021 20:41.

```
> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.00383886 ETH

2.deploy_contracts.js
=====
Deploying 'Adoption'
-----
> transaction hash: 0x58168764932a05f013696f604b4af217170cf4be9575b1f88a11252
0x2000000000000000000000000000000000000000000000000000000000000000
  Blocks: 0           Seconds: 0
  contract address: 0xcE20f880385737902DC25aE480Fc35Eb18Cc9f8
  block number: 26
  block timestamp: 16400998016
  round: 0x000000000000000000000000000000000000000000000000000000000000000
  balance: 99.04681562
  gas used: 66850 (0x10522)
  gas price: 20 gwei
  value sent: 0 ETH
  total cost: 0.001337 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.001337 ETH

Summary
-----
Total deployments: 2
Final cost: 0.00517586 ETH

E:\blockchainLab\pet-shop-tutorial>truffle test
```

Step 11: Open `/src/js/app.js` and remove multiline comment from within `initWeb3` and replace it with the following code

The file explorer shows a folder structure for "pet-shop-tutorial > test" containing files "App.js", "getAdopter.sol", and "TestAdoption.js". The browser window shows the Truffle Suite interface with steps to create a test file and add content to the "testAdopter.sol" file. The code editor shows the "app.js" file with the `initWeb3` function uncommented and replaced with the provided code.

```
File Tab Search View Encoding Languages Settings Help Plugins Window T
Open
E:\blockchainLab\pet-shop-tutorial>test
Organizer New folder
Documents Downloads Music Pictures Videos
WnHDD (C) New Volume (E) New Volume (F)
File Name: testAdopter.sol | All types | Open Cancel
length: 3195 lines: 127 cols: 1 pos: 1 Windows (DE) - 100% File
E:\blockchainLab\pet-shop-tutorial\src\js - Notepad++
File Edit Search View Encoding Language Settings Icons Help Window T
App.js
1 // Main application entry point
2
3 return await App.main();
4
5
6 initWeb3: async () => {
7   // Modern dapp browsers...
8   if (window.ethereum) {
9     const web3Provider = window.ethereum;
10    try {
11      // Request account access
12      await window.ethereum.request({ method: "eth_requestAccounts" });
13    } catch (error) {
14      // User denied account access
15      console.error("User denied account access");
16    }
17  }
18  // Legacy dapp browsers...
19  else if (window.web3) {
20    // Web3 instance is detected, fall back to web3
21    const web3Provider = window.web3.currentProvider;
22    if (web3Provider.isMetaMask) {
23      // MetaMask instance is detected, use it
24      const web3 = new Web3(web3Provider);
25      return App.initContract();
26    }
27  }
28  // No injected web3 instance is detected, tell back to dapp
29  else {
30    // Create a local web3 instance
31    const web3Provider = new Web3.providers.HttpProvider("http://localhost:7545");
32    const web3 = new Web3(web3Provider);
33  }
34
35  return App.initContract();
36}
37
38 initContract: async() => {
39   // If the contract has already been deployed, skip deployment
40   // If the contract hasn't been deployed, deploy it with the given name
41 }
```

Step 12: Remove multiline comment from within `initContract` and replace it and markAdopted and replace

Step 12 : Remove multi-line comment from initContract and Run command npm run dev and click on next, and connect the opened web page with metamask and adopt a pet by using Ethereum transaction .

```

49     return App.initContracts();
50   }
51 
52   initContracts: function() {
53     // get the deployed artifact file and instantiate it with web3
54     var AdoptionArtifact = artifacts.require("Adoption");
55     App.contracts.Adoption = Adoption;
56     Adoption.setProvider(App.web3Provider);
57 
58     // Set the provider for our contract
59     App.contracts.Adoption.setProvider(App.web3Provider);
60 
61     // Use the contract to retrieve and save the adopted pets
62     return App.loadAdopted();
63   }
64 
65   return App.bindEvents();
66 }
67 
68 handleEvents: function() {
69   document.on("blockhash", App.handleAdopt);
70 }
71 
72 markAdopted: function() {
73   var adoptionInstance;
74 
75   App.contracts.Adoption.deployed().then(function(instance) {
76     adoptionInstance = instance;
77 
78     return adoptionInstance.getAdopters.call();
79   }).then(function(adopters) {
80     for (let i = 0; i < adopters.length; i++) {
81       if (adopters[i] === "0x0000000000000000000000000000000000000000") {
82         return;
83       }
84     }
85   });
86 }
87 
88 catchFunction(error) {
89   console.error(error);
90 }
91 
92 handleAdopt: function(event) {
93   const accounts = event.accounts;
94 }

```

Things to notice:

- We first retrieve the artifact file for our smart contract. **Artifacts are Information about our contract such as Its deployed address and Application Binary Interface (ABI)**. The ABI is a JavaScript object defining how to interact with the contract.

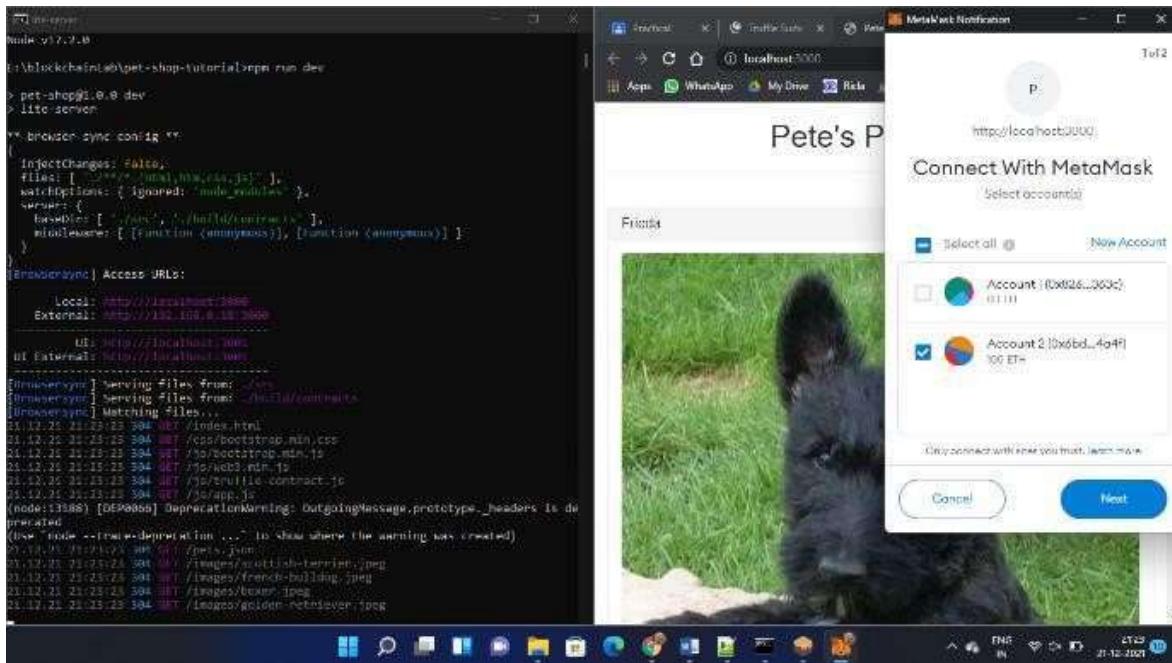
Step 13 : Remove multiline comment from handleAdopt and Run command npm run dev and click on next, and connect the opened web page with metamask and adopt a pet by using Ethereum transaction .

```

24   handleAdopt: function(event) {
25     event.preventDefault();
26 
27     var petId = parseInt(event.target.value);
28     var adoptionInstance;
29 
30     EthWeb3.eth.getAccounts(function(error, accounts) {
31       if (error) {
32         console.log(error);
33       }
34 
35       var accounts = accounts[0];
36 
37       App.contracts.Adoption.deployed().then(function(instance) {
38         adoptionInstance = instance;
39 
40         // Execute adopt as a transaction by sending account
41         adoptionInstance.adopt(petId, {from: accounts[0]});
42         console.log("Pet ID: " + petId);
43         console.log("Account: " + accounts[0]);
44       });
45     });
46   }
47 
48   EthContract: {
49     id: "0x0000000000000000000000000000000000000000"
50   }
51 }
52 
```

Things to notice:

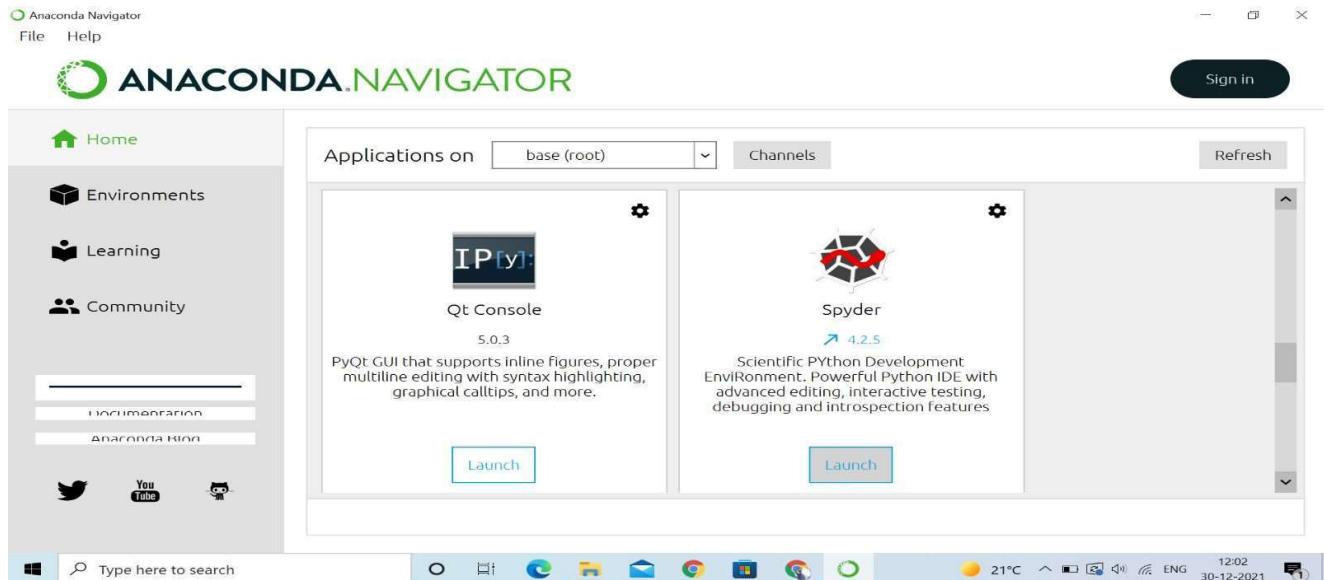
- We access the deployed Adoption contract, then call `getAdopters()` on that instance.



Practical No.8

Create the Blockchain using Python, Flask micro web framework and Postman.

Step 1 : Open Anaconda Navigator and Open Spyder.



Step 2 : Now open “Blockchain.py” file in spyder >> debug it and run it

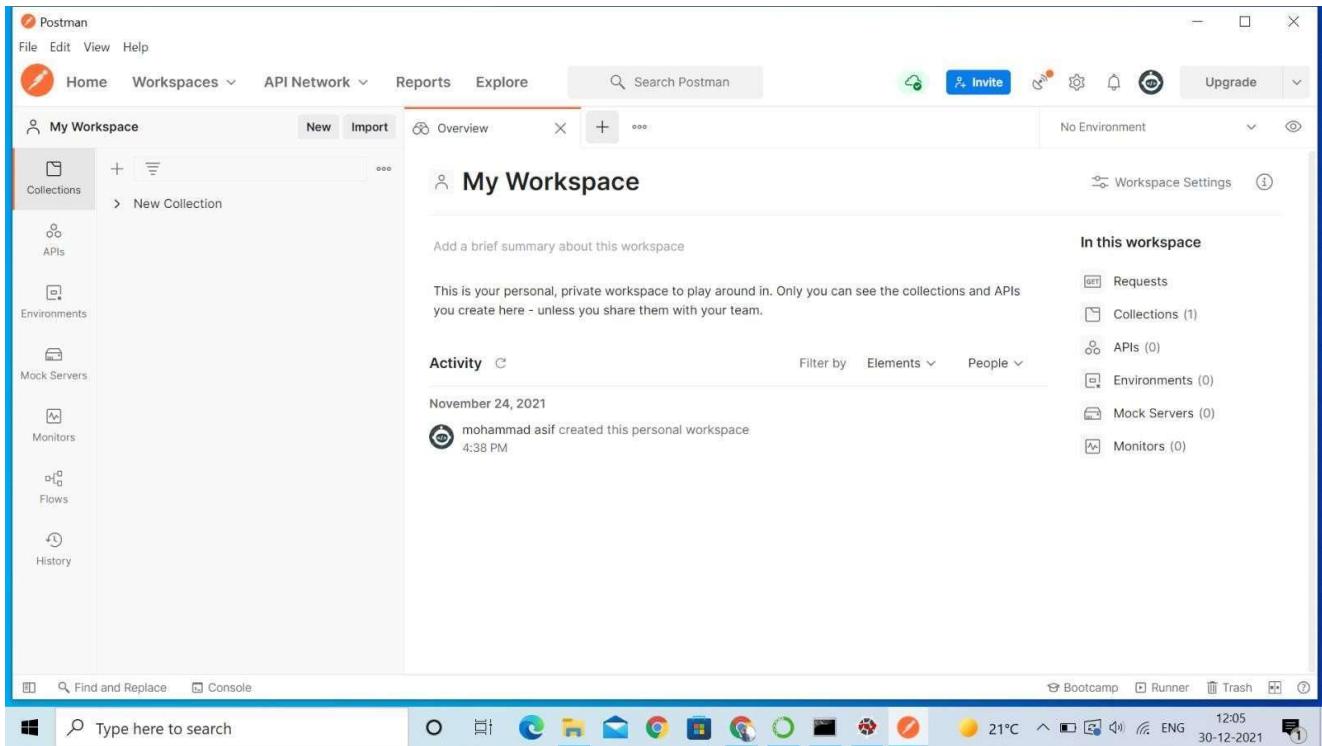
A screenshot of the Spyder IDE. The left pane shows the code editor with 'Blockchain.py' open, containing Python code for a blockchain. The right pane shows the IPython console with the code being run. The status bar at the bottom indicates the environment is 'conda: base (Python 3.8.8)'.

The screenshot shows the Spyder Python IDE interface. On the left, there are two tabs: 'Dictionary.py' and 'Blockchain.py'. The 'Blockchain.py' tab is active, displaying the following Python code:

```
1 # Module 1 - Create a Blockchain
2
3 # To be installed:
4 # Flask==0.12.2: pip install Flask==0.12.2
5 # Postman HTTP Client: https://www.getpostman.com/
6
7 # Importing the libraries
8 import datetime
9 import hashlib
10 import json
11 from flask import Flask, jsonify
12
13
14 # Part 1 - Building a Blockchain
15
16 class Blockchain:
17
18     def __init__(self):
19         self.chain = []
20         self.create_block(proof = 1, previous_hash = '0')
21
22     def create_block(self, proof, previous_hash):
23         block = {'index': len(self.chain) + 1,
24                  'timestamp': str(datetime.datetime.now()),
25                  'proof': proof,
26                  'previous_hash': previous_hash}
27         self.chain.append(block)
28         return block
29
30     def get_previous_block(self):
31         return self.chain[-1]
32
33     def proof_of_work(self, previous_proof):
34         new_proof = 1
```

The right side of the interface includes a 'Usage' help panel, a 'Console 1/A' window showing the execution of the script, and a system status bar at the bottom.

Step 3 : Open Postman and click on My Workspace.



Step 4 : Now write "http://127.0.0.1:5000/get_chain" in GET field and click on Send.

The screenshot shows the Postman application interface. In the top right, there's a search bar with "Search Postman". Below it, a button for "Invite" and other account settings. The main workspace shows a collection named "Media Authentication API" containing several requests like "Authentication", "AccessToken", etc. A specific request "GET http://10.10.193.234:5000/get_chain" is selected. The "Body" tab of the request details shows a JSON response:

```
1 "chain": [
2   {
3     "index": 1,
4     "previous_hash": "0",
5     "proof": 1,
6     "timestamp": "2022-01-11 23:57:43.153590"
7   }
8 ],
9 "length": 1
10
```

The status bar at the bottom indicates "200 OK 14 ms 252 B". The taskbar at the bottom of the screen shows various icons for system functions like battery, signal, and volume, along with the date and time "11-01-2022 23:58".

Step 5 : Now write "http://127.0.0.1:5000/mine_block" in GET field and click on Send and repeat it for three four times.

The screenshot shows the Postman application interface again. A collection named "My Workspace" is selected. A request "GET http://127.0.0.1:5000/mine_block" is selected. The "Body" tab of the request details shows a JSON response:

```
1 "index": 2,
2 "message": "Congratulations, you just mined a block!",
3 "previous_hash": "7011fb74e4ecb3290051ac37adc5dfc160306b2985ad4f4a483f79b93f408923",
4 "proof": 533,
5 "timestamp": "2021-12-30 12:05:39.700283"
```

The status bar at the bottom indicates "200 OK 22 ms 347 B". The taskbar at the bottom of the screen shows various icons for system functions like battery, signal, and volume, along with the date and time "30-12-2021 12:05".

Postman

File Edit View Help

Home Workspaces API Network Reports Explore

Search Postman

Overview GET http://127.0.0.1:50... + ... No Environment

Save Send

My Workspace New Import Overview

GET http://127.0.0.1:5000/mine_block

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description	...	

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 "index": 3,
2 "message": "Congratulations, you just mined a block!",
3 "previous_hash": "691f027f1b0aaf810ecf5242766b11bee4041e325db8974fb5aa0b6a3fc17547",
4 "proof": 45293,
5 "timestamp": "2021-12-30 12:06:02.268888"
```

200 OK 131 ms 349 B Save Response

Find and Replace Console

Bootcamp Runner Trash

Type here to search

12:06 30-12-2021

Postman

File Edit View Help

Home Workspaces API Network Reports Explore

Search Postman

Overview GET http://127.0.0.1:50... + ... No Environment

Save Send

My Workspace New Import Overview

GET http://127.0.0.1:5000/mine_block

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description	...	

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 "index": 4,
2 "message": "Congratulations, you just mined a block!",
3 "previous_hash": "48848053556f405d9253225f95200f0997aaa4c8e0b173747d0901983397fa34",
4 "proof": 21391,
5 "timestamp": "2021-12-30 12:06:15.259601"
```

200 OK 71 ms 349 B Save Response

Find and Replace Console

Bootcamp Runner Trash

Type here to search

12:06 30-12-2021

Postman

File Edit View Help

Home Workspaces API Network Reports Explore

Search Postman

Overview GET http://127.0.0.1:50... +

No Environment

My Workspace

Collections + New Import Overview http://127.0.0.1:5000/get_chain

APIs Environments

Mock Servers

Monitors Flows History

Find and Replace Console

Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
"chain": [
  {
    "index": 1,
    "previous_hash": "0",
    "proof": 1,
    "timestamp": "2021-12-30 12:03:27.963689"
  },
  {
    "index": 2,
    "previous_hash": "7011fb74e4ecb3290051ac37adc5dfc160306b2985ad4f4a483f79b93f408923",
    "proof": 533,
    "timestamp": "2021-12-30 12:05:39.700283"
  },
  {
    "index": 3,
```

Postman

File Edit View Help

Home Workspaces API Network Reports Explore

Search Postman

Overview GET http://127.0.0.1:50... +

No Environment

My Workspace

Collections + New Import Overview http://127.0.0.1:5000/get_chain

APIs Environments

Mock Servers

Monitors Flows History

Find and Replace Console

Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
      ],
      "length": 4
    },
```

Practical 9

**Read the real-time blocks (.dat files) of Bitcoin using Python.
Write at least five programs to read the blocks.**

Program 1

```
sample 1.py
from bitcoinlib.blocks import BlockFile, RawBlock
from bitcoinlib.core import Block

def read_blocks_bitcoinlib():
    block_file = BlockFile("/path/to/blocks")
    for raw_block in block_file.read_raw():
        block = Block(raw_block)
        print("Block Height:", block.height)
        # Additional block information can be extracted as needed

read_blocks_bitcoinlib()
```

Program 2

```
Example 1.py
1  import binascii
2  from pybitcointools.main import deserialize
3
4  def read_blocks_pybitcointools():
5      with open("/path/to/blocks", "rb") as file:
6          while True:
7              raw_block = file.read(80)
8              if not raw_block:
9                  break
10             block_data = deserialize(binascii.hexlify(raw_block).decode())
11             print("Block Hash:", block_data["hash"])
12
13 # Make sure to install the pybitcointools library:
14 # pip install bitcoin
15 read_blocks_pybitcointools()
```

Program 3

```
Example 1.py
1  from blocktools import Block
2
3  def read_blocks_blocktools():
4      with open("/path/to/blocks", "rb") as file:
5          while True:
6              raw_block = file.read(80)
7              if not raw_block:
8                  break
9              block = Block(raw_block)
10             print("Block Hash:", block.hash)
11
12
13     read_blocks_blocktools()
```

Program 4

```
Example 1.py > ...
    Click here to ask Blackbox to help you code faster
1  import subprocess
2  import json
3
4  def read_blocks_blockparser():
5      command = "blockparser --read-headers --silent /path/to/blocks"
6      process = subprocess.Popen(command, stdout=subprocess.PIPE, shell=True)
7      while True:
8          line = process.stdout.readline()
9          if not line:
10              break
11          block_data = json.loads(line)
12          print("Block Hash:", block_data["hash"])
13
14 # Make sure to install blockparser: https://github.com/znort987/blockparser
15     read_blocks_blockparser()
```

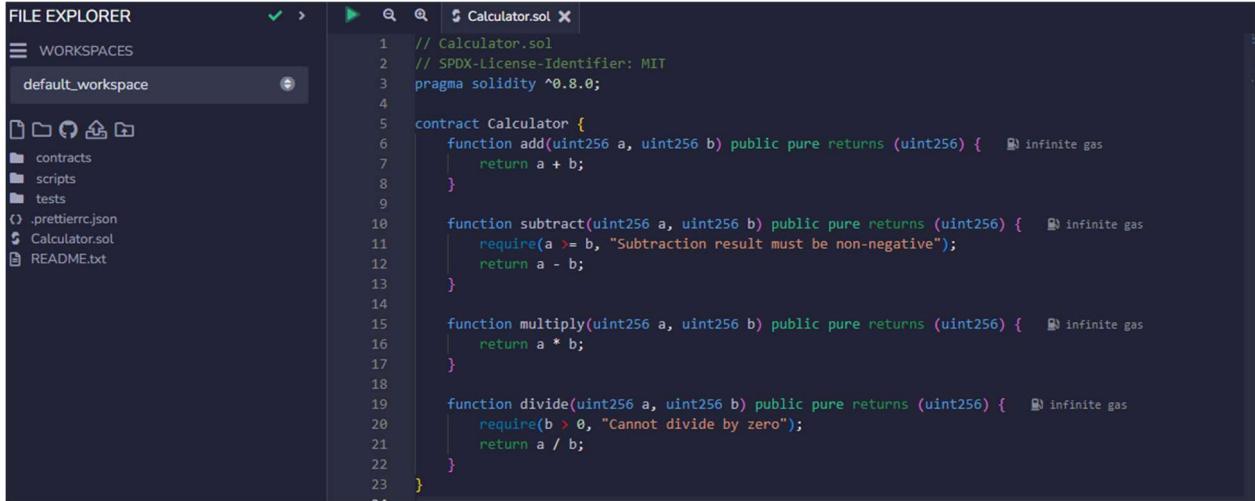
Program 5

```
Example 1.py
1  from bitcoin_block_explorer import BitcoinBlockExplorer
2
3  def read_blocks_block_explorer():
4      explorer = BitcoinBlockExplorer("/path/to/blocks")
5      for block in explorer.get_blocks():
6          print("Block Hash:", block.hash)
7
8 # Make sure to install bitcoin-block-explorer library:
9 # pip install bitcoin-block-explorer
0     read_blocks_block_explorer()
```

Practical 10

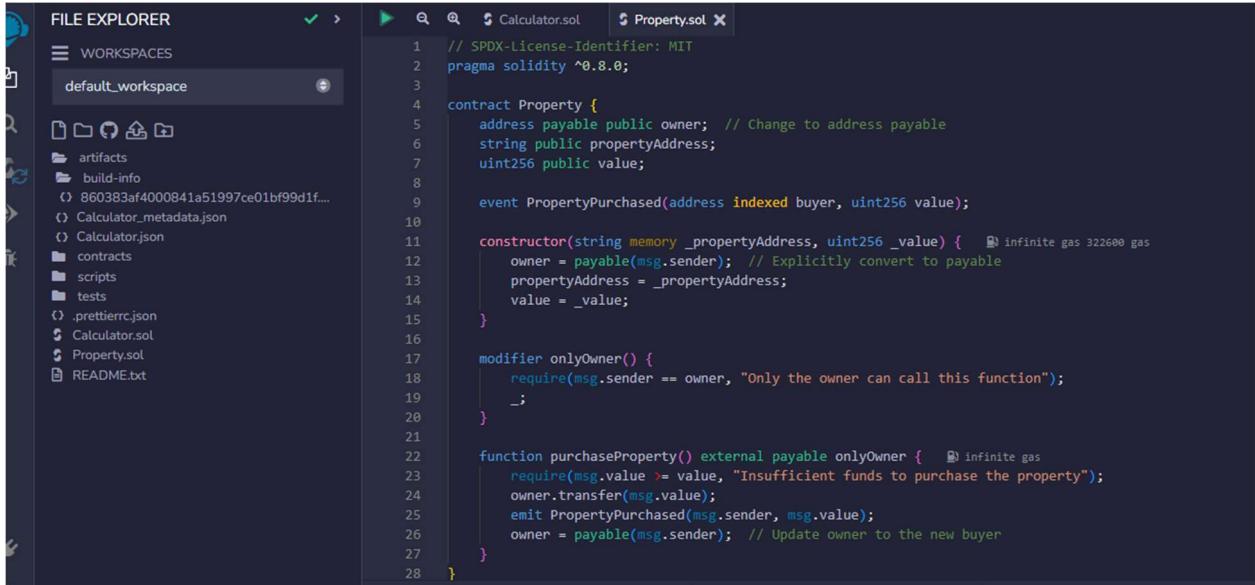
Create the following Contracts using Solidity Programming and compile these programs on Remix using Ropston Test Network, Goerli Test Network and Ganache network. a. Calculator b. Property c. Lottery

Calculator



```
FILE EXPLORER ▾ > ▶ Q ⓘ Calculator.sol X
WORKSPACES
default_workspace ⚙
contracts scripts tests .prettierrc.json Calculator.sol README.txt
1 // Calculator.sol
2 // SPDX-License-Identifier: MIT
3 pragma solidity ^0.8.0;
4
5 contract Calculator {
6     function add(uint256 a, uint256 b) public pure returns (uint256) {    infinite gas
7         return a + b;
8     }
9
10    function subtract(uint256 a, uint256 b) public pure returns (uint256) {   infinite gas
11        require(a >= b, "Subtraction result must be non-negative");
12        return a - b;
13    }
14
15    function multiply(uint256 a, uint256 b) public pure returns (uint256) {   infinite gas
16        return a * b;
17    }
18
19    function divide(uint256 a, uint256 b) public pure returns (uint256) {   infinite gas
20        require(b > 0, "Cannot divide by zero");
21        return a / b;
22    }
23 }
```

Property



```
FILE EXPLORER ▾ > ▶ Q ⓘ Calculator.sol ⓘ Property.sol X
WORKSPACES
default_workspace ⚙
artifacts build-info
860383af4000841a51997ce01bf99d1f... Calculator.metadata.json
Calculator.json contracts scripts tests .prettierrc.json Calculator.sol Property.sol README.txt
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract Property {
5     address payable public owner; // Change to address payable
6     string public propertyAddress;
7     uint256 public value;
8
9     event PropertyPurchased(address indexed buyer, uint256 value);
10
11     constructor(string memory _propertyAddress, uint256 _value) {    infinite gas 322600 gas
12         owner = payable(msg.sender); // Explicitly convert to payable
13         propertyAddress = _propertyAddress;
14         value = _value;
15     }
16
17     modifier onlyOwner() {
18         require(msg.sender == owner, "Only the owner can call this function");
19         ;
20     }
21
22     function purchaseProperty() external payable onlyOwner {    infinite gas
23         require(msg.value >= value, "Insufficient funds to purchase the property");
24         owner.transfer(msg.value);
25         emit PropertyPurchased(msg.sender, msg.value);
26         owner = payable(msg.sender); // Update owner to the new buyer
27     }
28 }
```

Lottery

The screenshot shows a code editor interface with two tabs open: `Lottery.sol` and `Property.sol`. The `Lottery.sol` tab is the active one, displaying the following Solidity code:

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

contract Lottery {
    address public manager;
    address[] public players;

    event WinnerSelected(address indexed winner, uint256 prize);

    constructor() {
        manager = msg.sender;
    }

    modifier onlyManager() {
        require(msg.sender == manager, "Only the manager can call this function");
        _;
    }

    function enter() external payable {
        require(msg.value > 0.01 ether, "Minimum entry fee is 0.01 ether");
        players.push(msg.sender);
    }

    function pickWinner() external onlyManager {
        require(players.length > 0, "No players in the lottery");
        uint256 index = random() % players.length;
        address winner = players[index];
        address winner = players[index];
        uint256 prize = address(this).balance;
        payable(winner).transfer(prize);
        emit WinnerSelected(winner, prize);
        players = new address[](0); // Reset the players array for the next round
    }

    function getPlayers() external view returns (address[] memory) {
        return players;
    }

    function random() private view returns (uint256) {
        return uint256(keccak256(abi.encodePacked(block.difficulty, block.timestamp, players)));
    }
}
```

The left sidebar, titled "FILE EXPLORER", shows the project structure for the `default_workspace`:

- artifacts
- build-info
- 860383af4000841a51997ce01bf99d1f...
- Calculator_metadata.json
- Calculator.json
- contracts
- scripts
- tests
- .prettierrc.json
- Calculator.sol
- Lottery.sol
- Property.sol
- README.txt

Practical 11

Supply chain management using python in blockchain technology

The screenshot shows a blockchain development interface with two tabs of Solidity code for a supply chain contract.

Top Tab Content:

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract SupplyChain {
5     address public owner;
6
7     enum Role {Manufacturer, Supplier, Distributor, Retailer, Consumer}
8
9     struct Product {
10         uint256 productId;
11         address owner;
12         Role currentRole;
13         string productName;
14     }
15
16     mapping(uint256 => Product) public products;
17     uint256 public productCounter;
18
19     event ProductCreated(uint256 productId, address owner, Role currentRole, string productName);
20     event RoleUpdated(uint256 productId, Role newRole);
21
22     modifier onlyOwner() {
23         require(msg.sender == owner, "Only the owner can call this function");
24         _;
25     }
26
27     constructor() { }
```

Bottom Tab Content:

```
26
27 constructor() { 752426 gas 727400 gas
28     owner = msg.sender;
29 }
30
31 function createProduct(string memory _productName) public onlyOwner {  infinite gas
32     productCounter++;
33     products[productCounter] = Product(productCounter, msg.sender, Role.Manufacturer, _productName);
34     emit ProductCreated(productCounter, msg.sender, Role.Manufacturer, _productName);
35 }
36
37 function updateRole(uint256 _productId, Role _newRole) public onlyOwner {  undefined gas
38     require(_productId <= productCounter && _productId > 0, "Invalid product ID");
39     products[_productId].currentRole = _newRole;
40     emit RoleUpdated(_productId, _newRole);
41 }
42 }
```

```
Supply_chain.py > ...
    Click here to ask Blackbox to help you code faster
1  from web3 import Web3
2  web3 = Web3(Web3.HTTPProvider('http://localhost:8545'))
3
4  # Load contract ABI and bytecode
5  with open('SupplyChain.abi', 'r') as file:
6      abi = file.read()
7
8  with open('SupplyChain.bin', 'r') as file:
9      bytecode = file.read()
10
11 # Deploy the contract
12 supply_chain = web3.eth.contract(abi=abi, bytecode=bytecode)
13 tx_hash = supply_chain.constructor().transact({'from': web3.eth.accounts[0]})
14 tx_receipt = web3.eth.waitForTransactionReceipt(tx_hash)
15
16 # Interact with the contract
17 contract_address = tx_receipt['contractAddress']
18 supply_chain_instance = web3.eth.contract(address=contract_address, abi=abi)
19
20 # Create a product
21 supply_chain_instance.functions.createProduct("ProductA").transact({'from': web3.eth.accounts[0]})
22
23 # Update role of the product
24 supply_chain_instance.functions.updateRole(1, 2).transact({'from': web3.eth.accounts[0]})
25
26 # Get product details
27 product_details = supply_chain_instance.functions.products(1).call()
28 print("Product Details:", product_details)
29
```