

Bharati Vidyapeeth’s

**Institute of Management & Information Technology**

C.B.D. Belapur, Navi Mumbai 400614

**Vision:**

Providing high quality, innovative and value-based education in information technology to build

competent professionals.

**Mission**

M1. Technical Skills: To provide solid technical foundation theoretically as well as practically

capable of providing quality services to industry.

M2. Development: Department caters to the needs of students through comprehensive educational

programs and promotes lifelong learning in the field of computer Applications.

M3. Ethical leadership: Department develops ethical leadership insight in the students to succeed in

industry, government and academia

**CERTIFICATE**

This is to certify that the journal is the work of

**Mr. Fardin Raje**  Roll No. **43** of MCA

(Sem: - \_3\_ Div: -\_B\_) For the Academic Year 2021-2023

Subject Code: - MCALE331

Subject Name: Blockchain Lab

Subject-in-charge Principal

Date:

External Examiner

Date:

# Bharati Vidyapeeth's Institute of Management & Information Technology

**Academic Year – 2021-23**

**MCA Sem III Division B**

**PRACTICAL INDEX**

**MCALE331 Blockchain LAB**

**Name: Fardin Raje Roll No.: 43 Div: B**

|  |  |  |
| --- | --- | --- |
| **Practical No** | **Problem Statement** | **Sign** |
| 1 | Implementation of Caeser Cipher and show the encryption as well as decryption process using JAVA or Python. (Symmetric Encryption) |  |
| 2 | Implementation of RSA Algorithm (Asymmetric Encryption) Encrypt and decrypt a string. |  |
| 3 | Implementation of SHA-256 (Use any programming Language) |  |
| 4 | Implementation of Binary Tree and to show all operations (Insert, Delete, Traversals, Display) |  |
| 5 | Blockchain creation program using Java |  |
| 6 | Install Ganache and metamask.  Compile and deploy an election smart contract in the personal blockchain using injected web 3 environments. Use Remix online IDE to compile and deploy the smart contract. Execute the smart contract and show the output. |  |
| 7 | Program using Solidity to check Balance |  |
| 8 | The use of GANACHE Truffle Suite to Deploy a Smart Contract in Solidity (Blockchain) |  |
| 9 | Write a program in solidity to create a structured student with Roll no, Name,Class, Department,Course enrolled as variables.  Add information of 5 students.  Search for a student using Roll no  Display all information |  |
| 10 | Create Daps Voting Process using Solidity smart contract and web3 (available on Classroom) |  |
| 11 | Mini Project |  |

**Practical No.: 01**

**Aim: Implementation of Caesar Cipher and show the encryption as well as decryption process using JAVA or Python. (Symmetric Encryption)**

**Program:**

def encrypt(text,s):

result = ""

# traverse text

for i in range(len(text)):

char = text[i]

# Encrypt uppercase characters

if (char.isupper()):

result += chr((ord(char) + s-65) % 26 + 65)

# Encrypt lowercase characters

else:

result += chr((ord(char) + s - 97) % 26 + 97)

return result

#check the above function

text = "ATTACKATONCE"

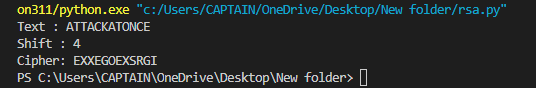
s = 4

print ("Text : " + text)

print ("Shift : " + str(s))

print ("Cipher: " + encrypt(text,s))

**Output:**

****

**Practical No.: 02**

**Aim: Implementation of RSA Algorithm (Asymmetric Encryption) Encrypt and decrypt a string.**

**Program:**

Pip install rsa

import rsa

public\_key, private\_key=rsa.newkeys (512)

def encrypt\_text (plain\_text):

plain\_text=plain\_text.encode('utf8')

encrypted\_text=rsa.encrypt (plain\_text, public\_key)

return encrypted\_text

def decrypt\_text (encrypted\_text) :

decrypted\_text=rsa.decrypt(encrypted\_text, private\_key)

return decrypted\_text.decode('utf-8')

# testing

plain\_text="Tayor Swift is the Music Industry"

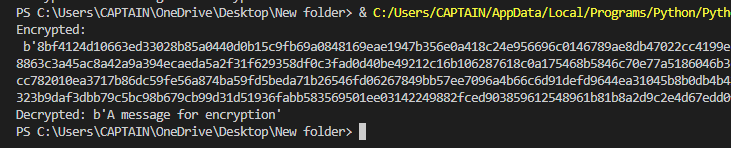
encrypted\_text=encrypt\_text (plain\_text)

print("Encrypted text is = %s" %(encrypted\_text))

decrypted\_text= decrypt\_text (encrypted\_text)

print ("Decrypted text is = %s" %(decrypted\_text))

**Output:**



**Practical No.: 03**

**Aim: Implementation of SHA-256 (Use any programming Language)**

**Program:**

import hashlib

string="Reputation is a great album"

encoded=string.encode()

result = hashlib.sha256(encoded)

print ("String :",end ="")

print (string)

print ("Hash Value : =",end="")

print (result)

print("Hexadecimal equivalent: ",result.hexdigest())

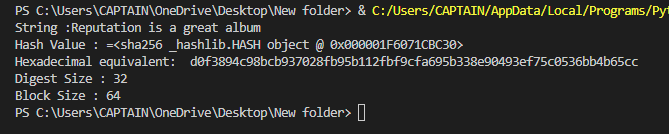
print ("Digest Size : ", end ="")

print (result.digest\_size)

print ("Block Size : ", end ="")

print (result.block\_size)

**Output:**



**Practical No.: 04**

**Aim: Implementation of Binary Tree and to show all operations (Insert, Delete, Traversals, Display)**

**Program:**

class Node:

def \_\_init\_\_(self, data):

self.left = None

self.right = None

self.data = data

**# Insert Node**

def insert(self, data):

if self.data:

if data < self.data:

if self.left is None:

self.left = Node(data)

else:

self.left.insert(data)

elif data > self.data:

if self.right is None:

self.right = Node(data)

else:

self.right.insert(data)

else:

self.data = data

**# Print the Tree**

def PrintTree(self):

if self.left:

self.left.PrintTree()

print( self.data),

if self.right:

self.right.PrintTree()

**# Preorder traversal**

**# Root -> Left ->Right**

def PreorderTraversal(self, root):

res = []

if root:

res.append(root.data)

res = res + self.PreorderTraversal(root.left)

res = res + self.PreorderTraversal(root.right)

return res

**# function to delete the given deepest node (d\_node) in binary tree**

def deleteDeepest(root, d\_node):

q = []

q.append(root)

while(len(q)):

temp = q.pop(0)

if temp is d\_node:

temp = None

return

if temp.right:

if temp.right is d\_node:

temp.right = None

return

else:

q.append(temp.right)

if temp.left:

if temp.left is d\_node:

temp.left = None

return

else:

q.append(temp.left)

**# function to delete element in binary tree**

def deletion(root, key):

if root == None:

return None

if root.left == None and root.right == None:

if root.key == key:

return None

else:

return root

key\_node = None

q = []

q.append(root)

temp = None

while(len(q)):

temp = q.pop(0)

if temp.data == key:

key\_node = temp

if temp.left:

q.append(temp.left)

if temp.right:

q.append(temp.right)

if key\_node:

x = temp.data

deleteDeepest(root, temp)

key\_node.data = x

return root

root = Node(27)

root.insert(14)

root.insert(35)

root.insert(10)

root.insert(19)

root.insert(31)

root.insert(42)

print(root.PreorderTraversal(root))

print("The tree before the deletion:")

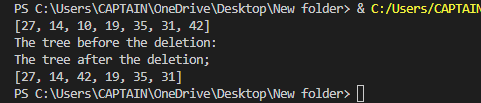
key = 10

root = deletion(root, key)

print("The tree after the deletion;")

print(root.PreorderTraversal(root))

**Output:**



**Practical No.: 05**

**Aim: Blockchain creation program using Java**

**Program:**

const SHA256=require("crypto-js/sha256");

class block

{

constructor(index,timestamp,data,previoushash="")

{

this.index=index;

this.timestamp=timestamp;

this.data=data;

this.previoushash=previoushash;

this.hash=this.calculateHash();

}

calculateHash()

{

return SHA256(this.index +this.timestamp+this.previoushash+JSON.stringify(this.data)).toString();

}

}

class Blockchain

{

constructor(index,timestamp,data,previoushash="")

{

this.index=index;

this.timestamp=timestamp;

this.data=data;

this.previoushash=previoushash;

this.chain=[this.createGenesisBlock()];

}

createGenesisBlock()

{

return new block(0,"23/11/2021","this is first program of blockchain creation","0");

}

addBlock(newBlock)

{

newBlock.previoushash=this.getLatestBlock().hash;

newBlock.hash=newBlock.calculateHash();

this.chain.push(newBlock);

}

getLatestBlock()

{

return this.chain[this.chain.length-1];

}

}

let block1 =new block(1,"22/11/2021","data1","o");

let block2 =new block(2,"21/11/2021","second block","");

let block3 =new block(2,"14/11/2021","third block","");

let myBlockchain =new Blockchain();

myBlockchain.addBlock(block1);

myBlockchain.addBlock(block2);

myBlockchain.addBlock(block3);

console.log(JSON.stringify(myBlockchain,null,4));

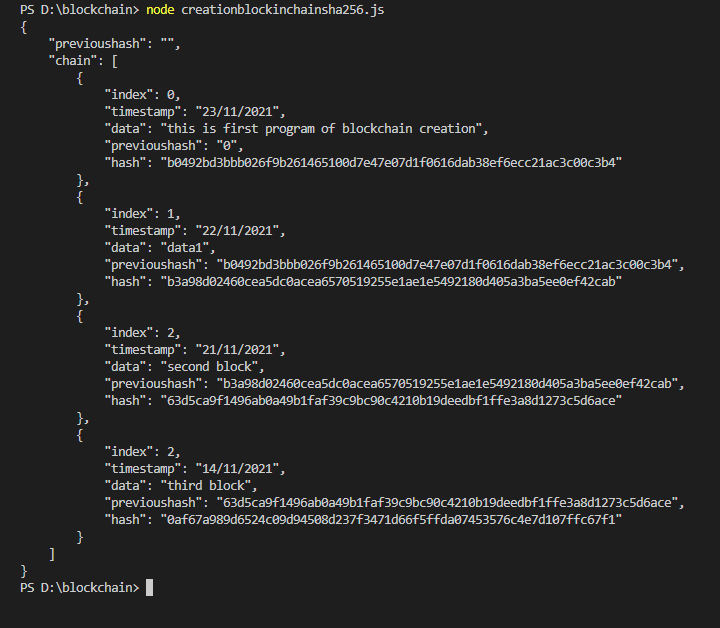
**Output:**

open terminal=

Go in the folder where you have saved folder(path)

->npm install crypto-js

->node creationblockinchainsha256.js

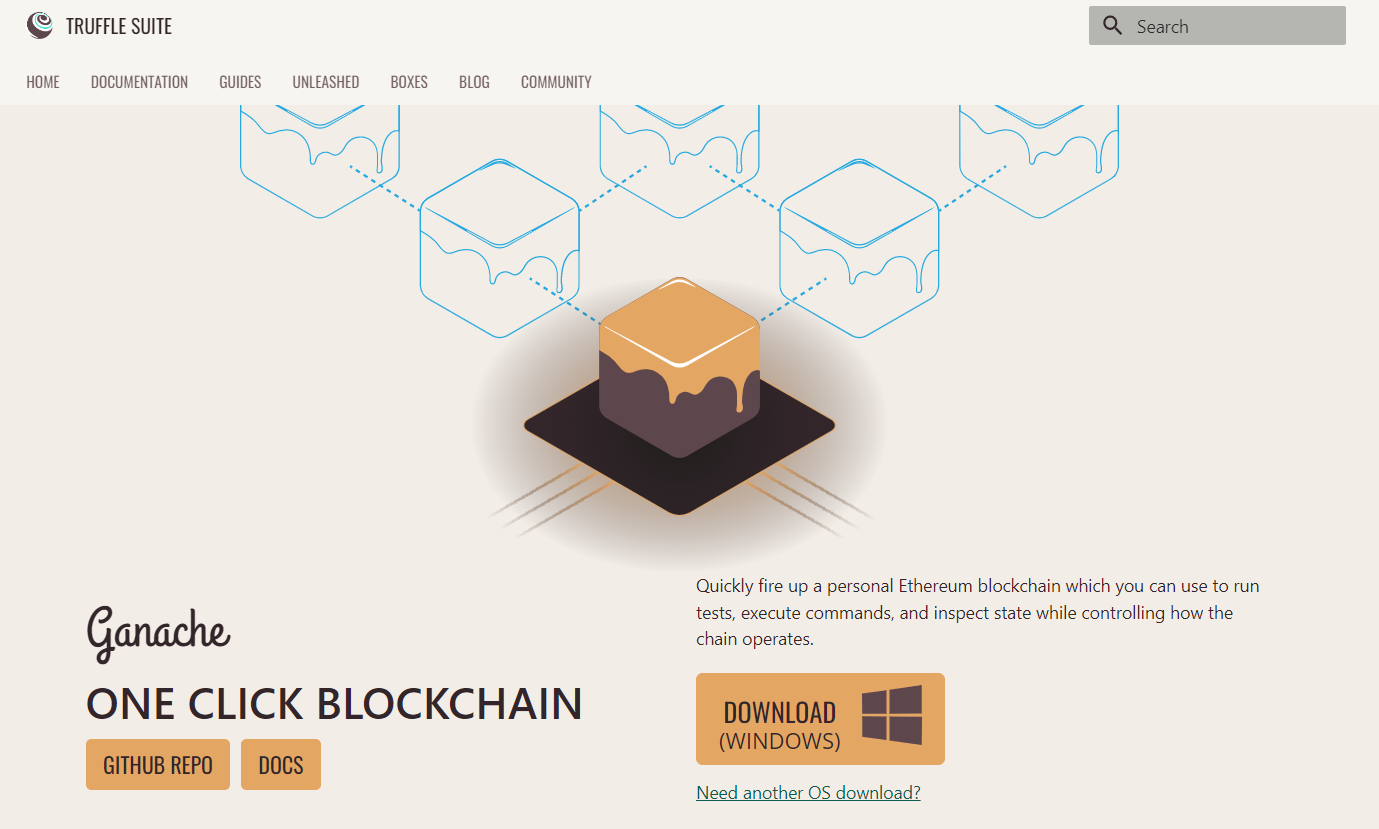


**Practical No.: 06**

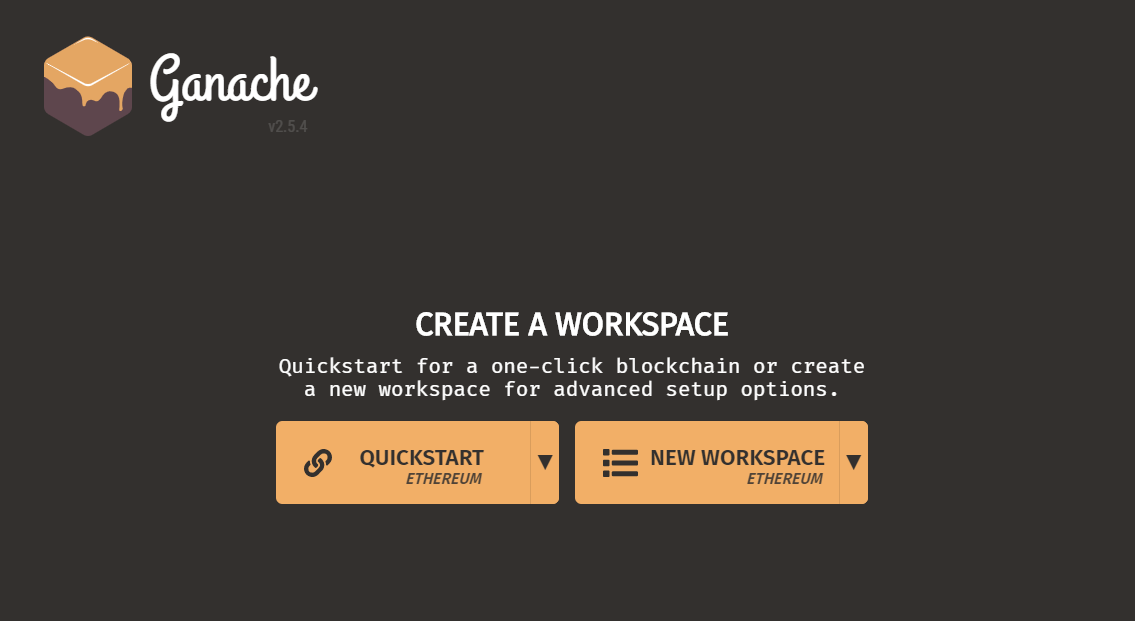
# Aim: To implement the installation of Ganache, Metamask and Remix IDE and deploy smart contract using injected web 3 environment.

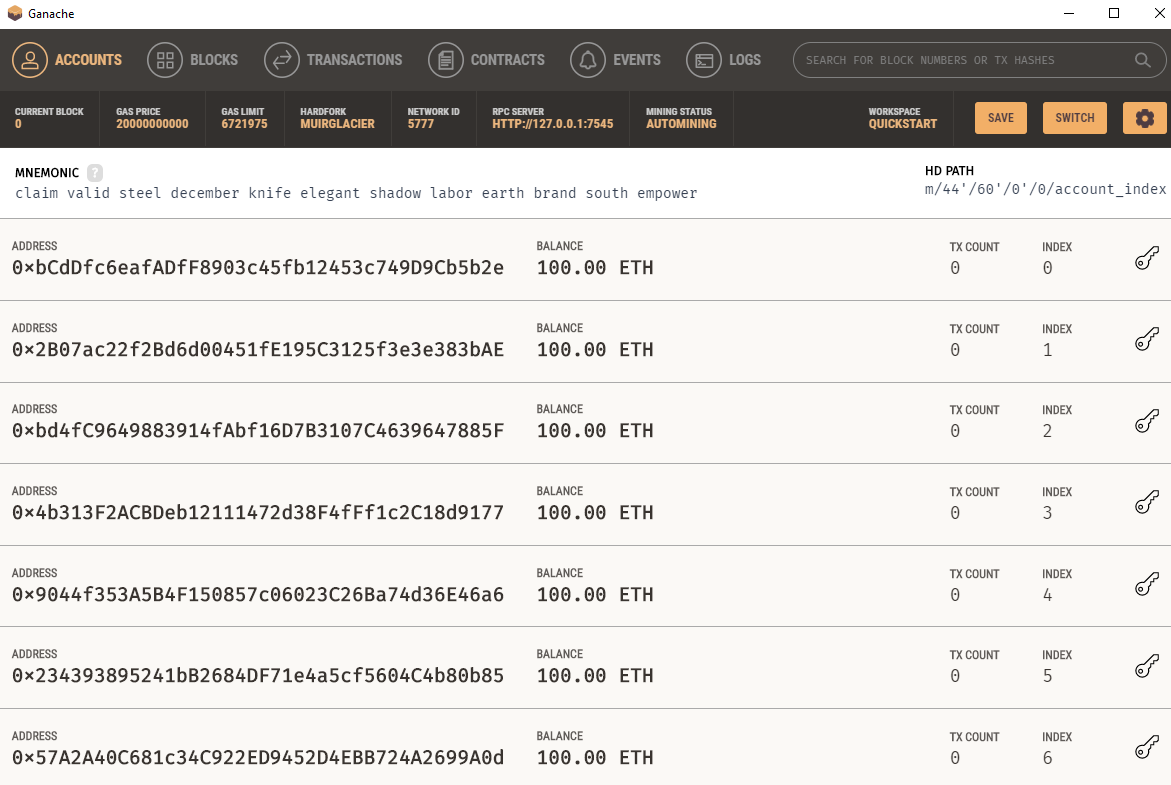
## Ganache:

1. Download Ganache from <https://www.trufflesuite.com/ganache>



1. Install Ganache



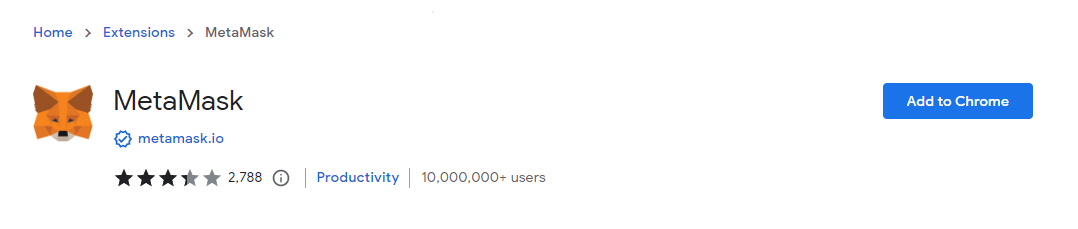


The console in the above screenshot shows user accounts with balance of 100 ETH (Ether - a currency for transaction on Ethereum platform). It shows a transaction count of zero for each account. As the user has not performed any transactions so far, this count is obviously zero.

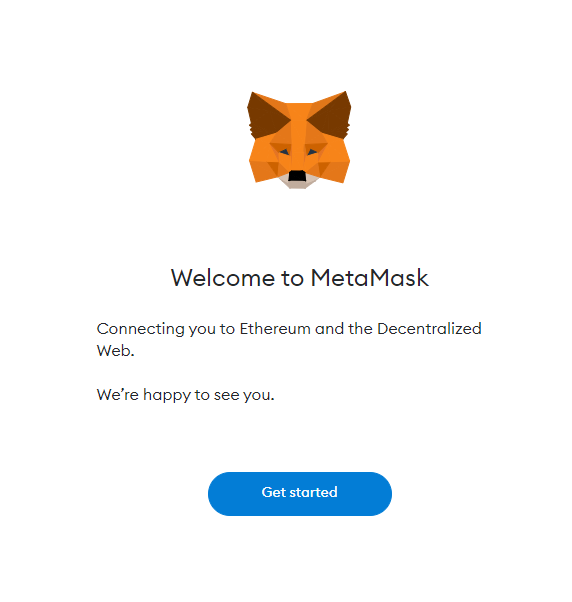
**Metamask**: Installation

1) Go to Chrome Web Store Extensions Section.

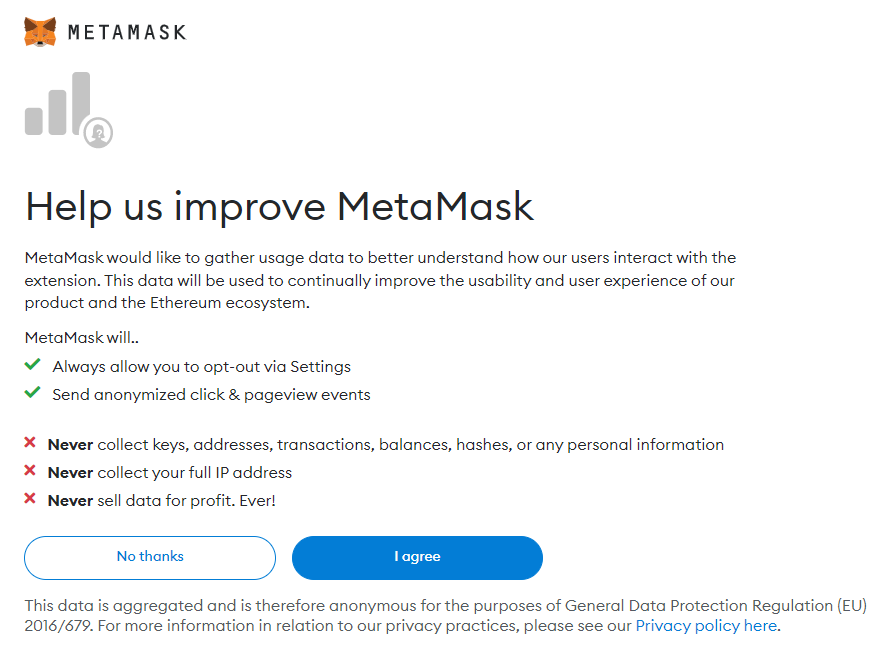
2) Search MetaMask and click add to chrome



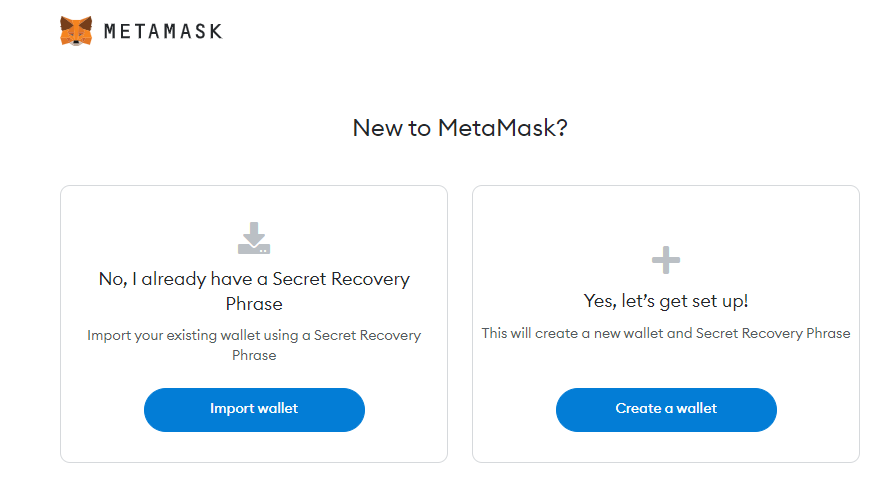
3)Once installation is complete this page will be displayed. Click on the Get Started button



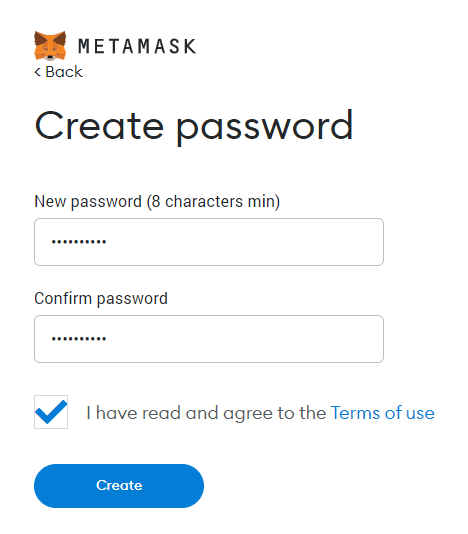
5)Click I Agree button to allow data to be collected to help improve MetaMask or else click the No Thanks button. The wallet can still be created even if the user will click on the No Thanks button



5) This is the first time creating a wallet, so click the Create a Wallet button. If there is already a wallet then import the already created using the Import Wallet button



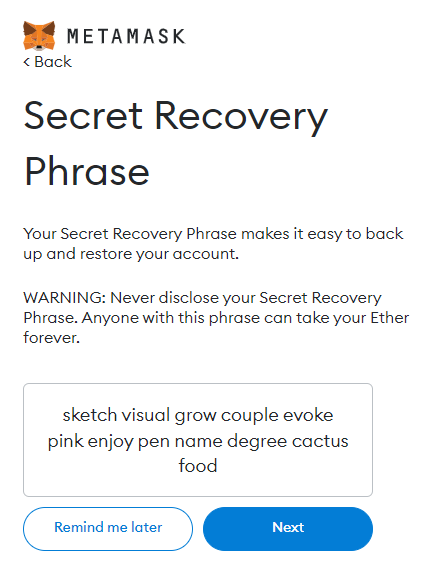
6) Create a password for your wallet. This password is to be entered every time the browser is launched and wants to use MetaMask. A new password needs to be created if chrome is uninstalled or if there is a switching of browsers. In that case, go through the Import Wallet button. This is because MetaMask stores the keys in the browser. Agree to Terms of Use



8) Click on the dark area which says Click here to reveal secret words to get your secret phrase.

9) This is the most important step. Back up your secret phrase properly. Do not store your secret phrase on your computer. Please read everything on this screen until you understand it completely before proceeding. The secret phrase is the only way to access your wallet if you forget your password.

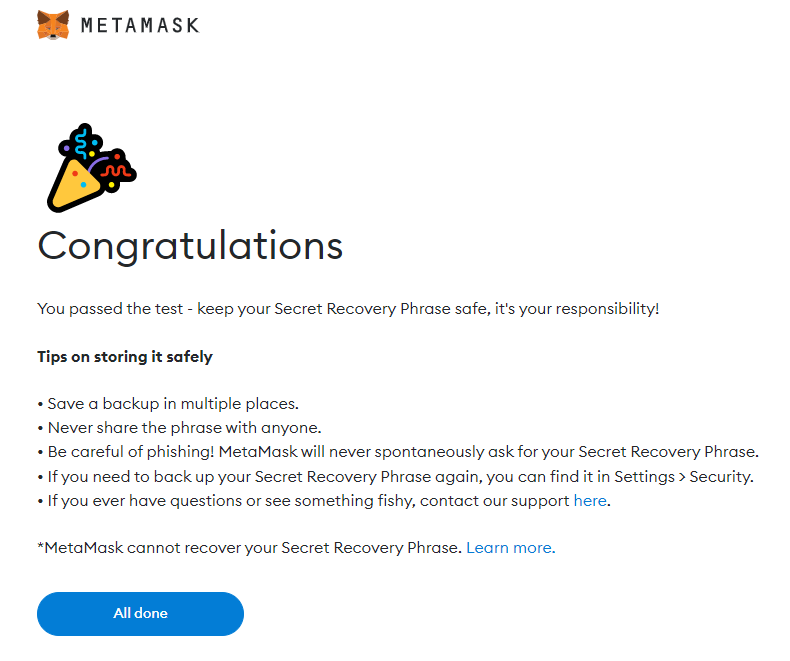
Once done click the Next button.



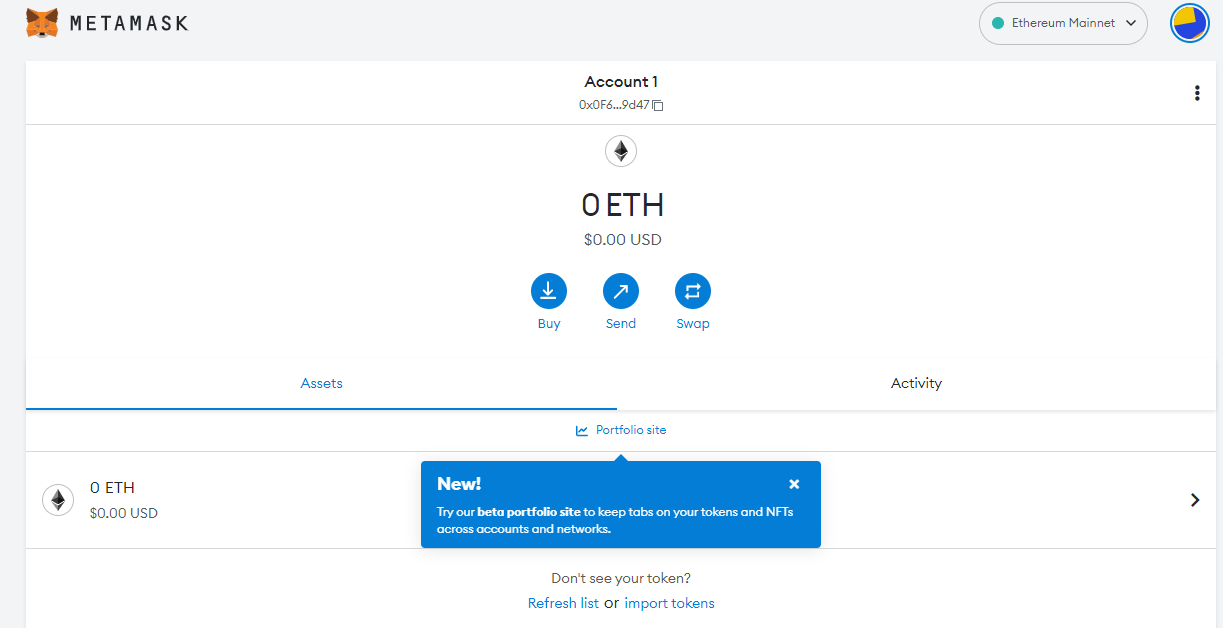
10) Click the buttons respective to the order of the words in your seed phrase. In other words, type the seed phrase using the button on the screen. If done correctly the Confirm button should turn blue.



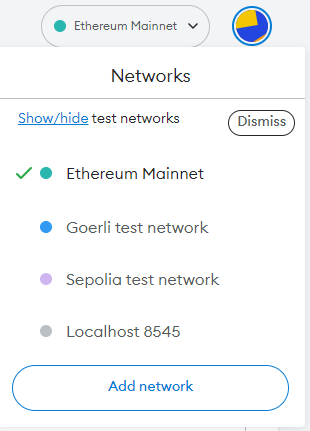
11) Click the All Done button. Please follow the tips mentioned.



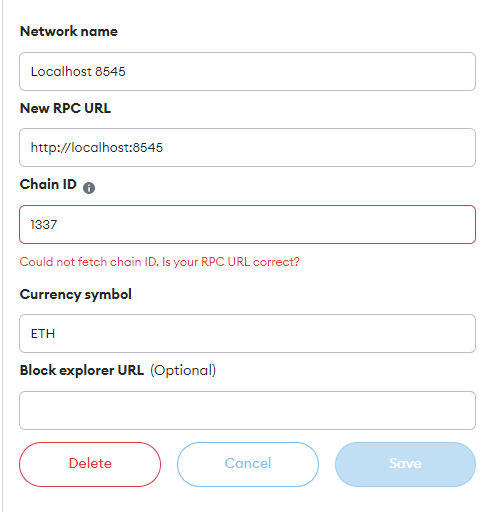
12) One can see the balance and copy the address of the account by clicking on the Account 1 area.



13) One can access MetaMask in the browser by clicking the MetaMask extension icon on the top right.

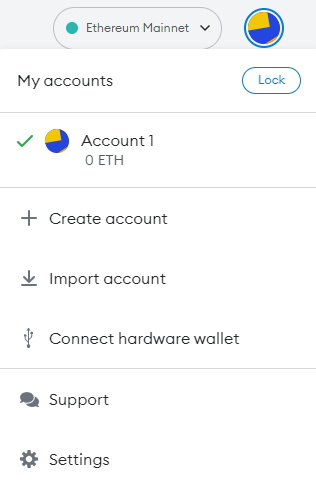


14) Adding Ganache Network to MetaMask.

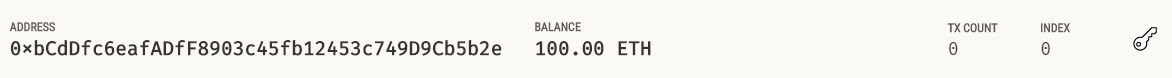


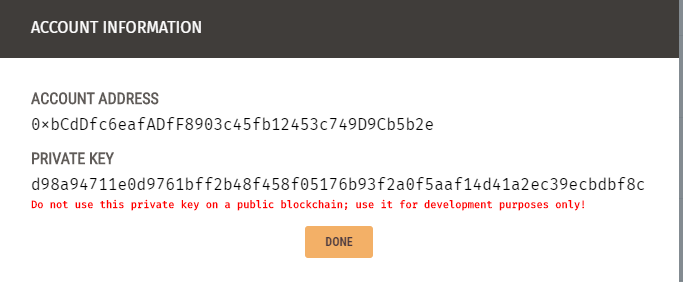
15: Importing Accounts.

* To Import an account, click on the circular icon at the top – right of your MetaMask Extension and select Import accounts. Copy the private key of your ganache account by clicking on key icon of particular account.

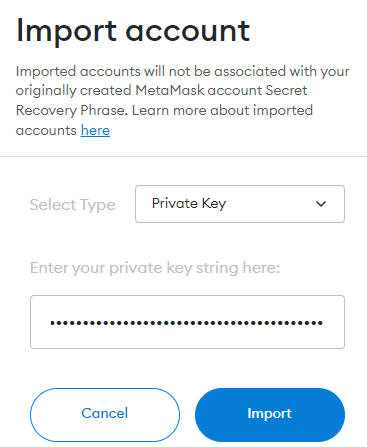


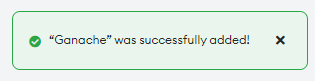
* You will need to copy the private key of your ganache account by clicking on key icon of particular account.

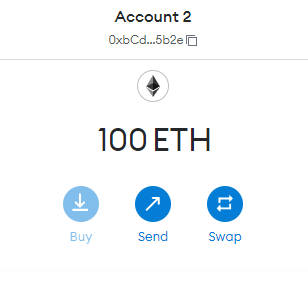


****

- Click on import button once private key string is pasted.

****

****

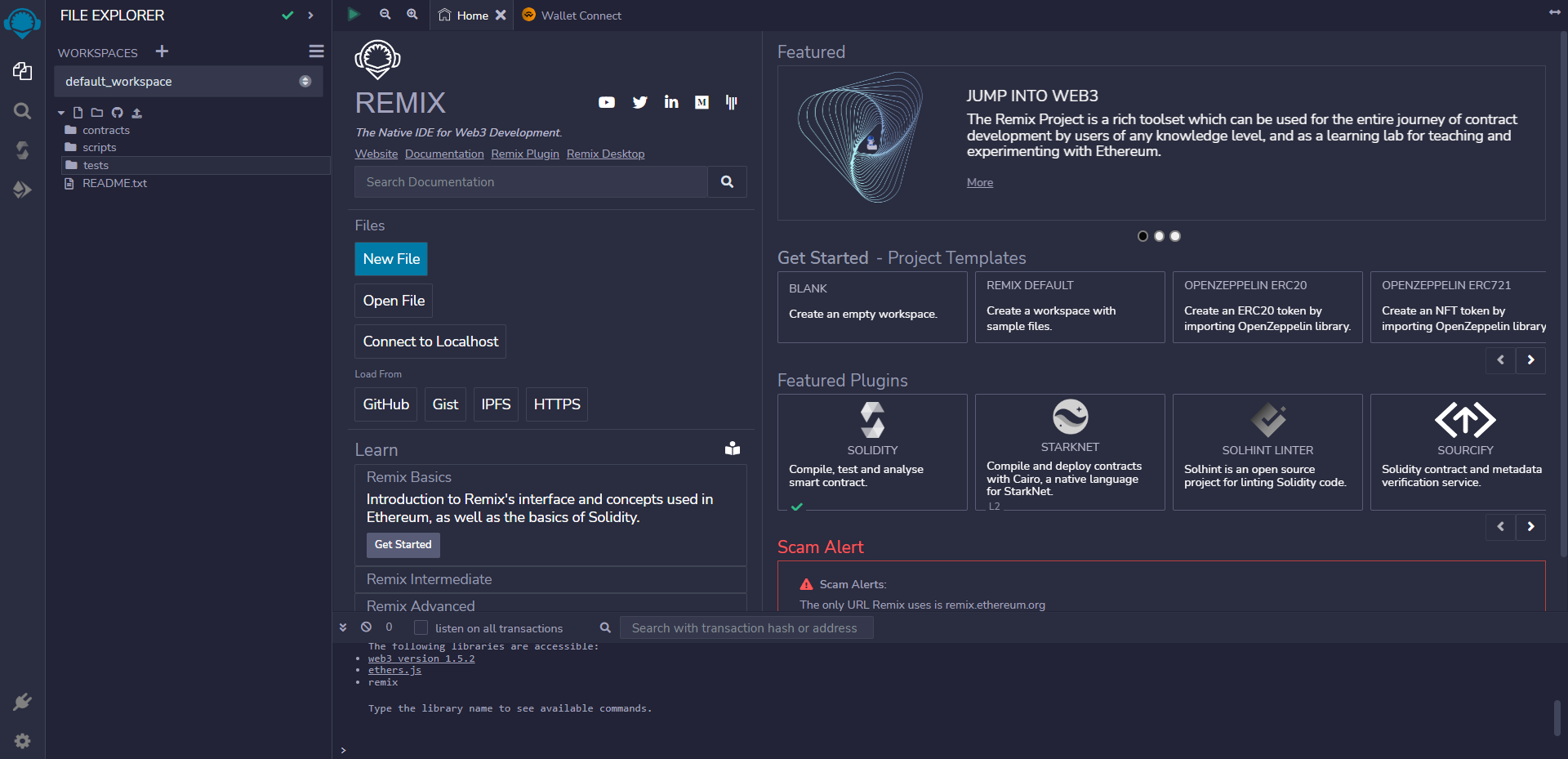
****

## Remix IDE:

1. Go to <https://remix.ethereum.org/>

As we use Solidity to write our smart contracts, .sol extension is used.

Let’s now create a new contract. For that, right-click on the workspace and select New File. Name our file.

****

2) Select your newly created file and type the following code.

## Program:

pragma solidity ^0.4.2;

contract Election {

// Model a Candidate structCandidate {

uint id;

string name;

uint voteCount;

}

// Store accounts that have voted

mapping(address => bool) public voters;

// Store Candidates

// Fetch Candidate

mapping(uint => Candidate) public candidates;

// Store Candidates

Count uint public candidatesCount;

// voted event

eventvotedEvent (

uint indexed \_candidateId

);

function Election() public {

addCandidate("N MODI, BJP"); addCandidate("A kejriwal, AAP"); addCandidate("Rahul G, Congress"); addCandidate("Nikhil, JDS");

}

function addCandidate (string \_name) private

{ candidatesCount ++;

candidates[candidatesCount] = Candidate(candidatesCount, \_name, 0);

}

function vote (uint \_candidateId) public {

// require that they haven't voted before require(!voters[msg.sender]);

// require a valid candidate

require(\_candidateId > 0 && \_candidateId <= candidatesCount);

// record that voter has voted voters[msg.sender] = true;

// update candidate vote Count candidates[\_candidateId].voteCount ++;

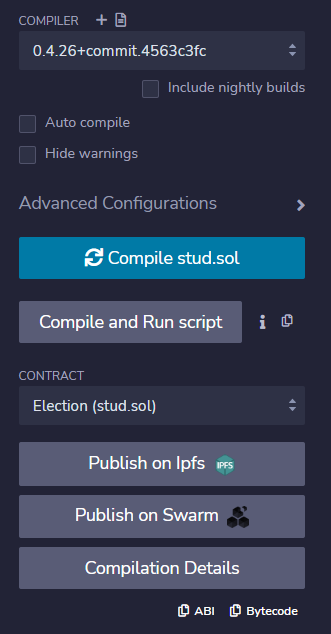
// trigger voted event votedEvent(\_candidateId);

}

}

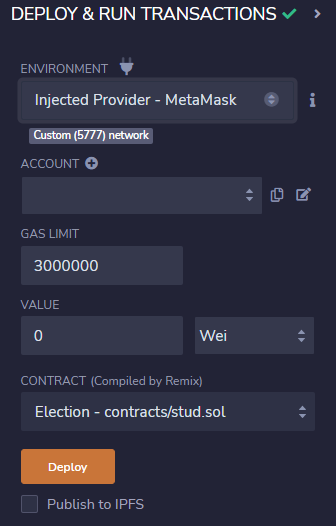
## 3) Click on the solidity compiler present on the left.

Select Auto-compile so our contract automatically compiles when we do some changes.

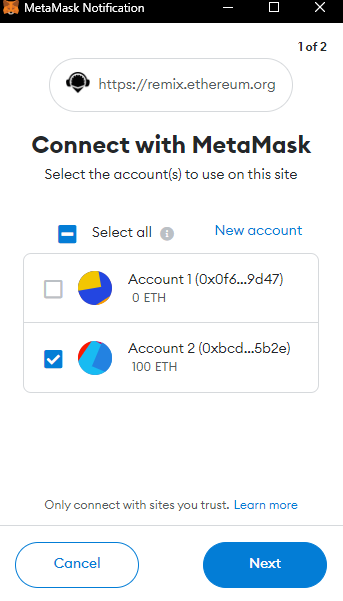


4) Click on Deploy and Run Transactions Button.

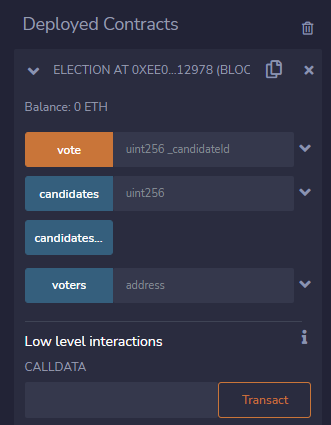
Set Environment to “Injected Web3”. (Make sure you are connected to website with Metamask).



Click on Deploy Button and you will see a pop up for confirmation. Once confirm your contract is deployed

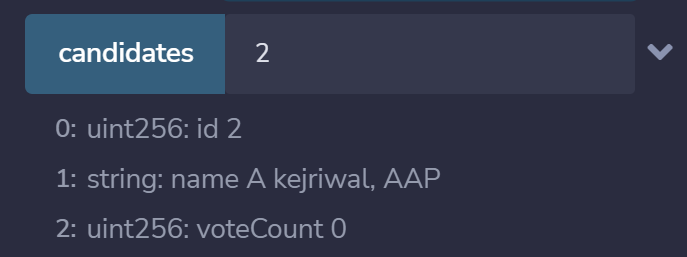


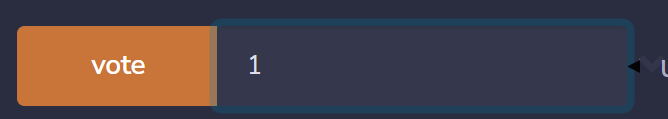
After your contract is successfully deployed, you will able to see your contract under the deployed contracts.



If you give the input as 1 in Candidate’s column, you will be to see the details of our first candidate. In our case, the first candidate is Modi.

If you give the input as 1 in Candidate’s column, you will be to see the details of our first candidate. In our case, the first candidate is Modi.

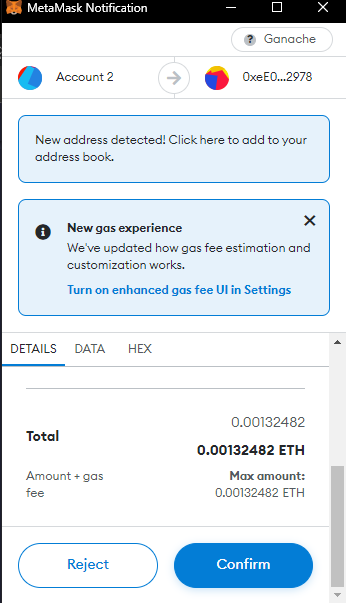




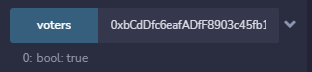
If you click on votes button with any account address as input you can see whether the person has voted or not.

If you click on vote button with input as your candidate’s id for e.g., 1, a pop up will appear to confirm the transaction.

Once confirmed the voted will be registered



It gives the Boolean result meaning false as not voted and true as voted



If you click on candidatesCount Button you will get the count of total candidates standing for election



**Practical No.: 07**

**Aim: Program using Solidity to check Balance**

**Program:**

pragma solidity >=0.7.0 <0.9.0;

// Creating a contract

contract MyContract

{

// Private state variable

address private owner;

// Defining a constructor

constructor() public{

owner=msg.sender;

}

// Function to get

// address of owner

function getOwner(

) public view returns (address) {

return owner;

}

// Function to return

// current balance of owner

function getBalance(

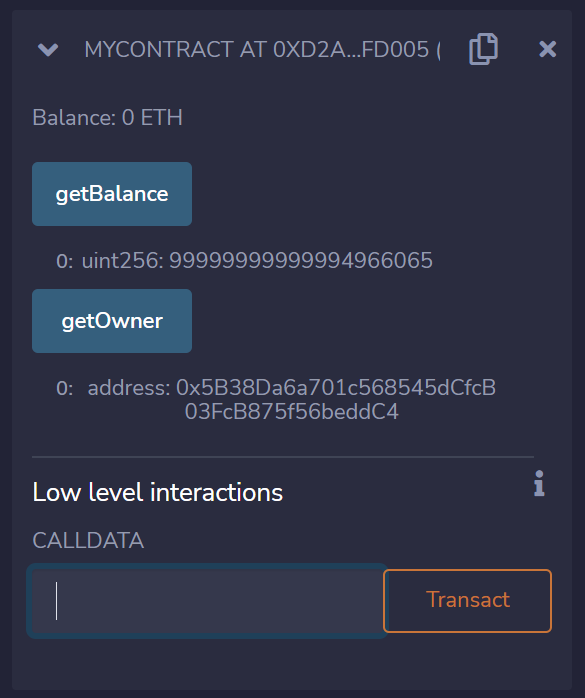
) public view returns(uint256){

return owner.balance;

}

}

**Output:**



**Practical No.: 08**

**Aim: The use of GANACHE Truffle Suite to Deploy a Smart Contract in Solidity (Blockchain)**

**Program:**

// SPDX-License-Identifier: MIT

pragma solidity >=0.4.0 <0.7.0;

contract SimpleStorage {

uint storedData;

function set(uint x) public {

storedData = x;

}

function get() public view returns (uint) {

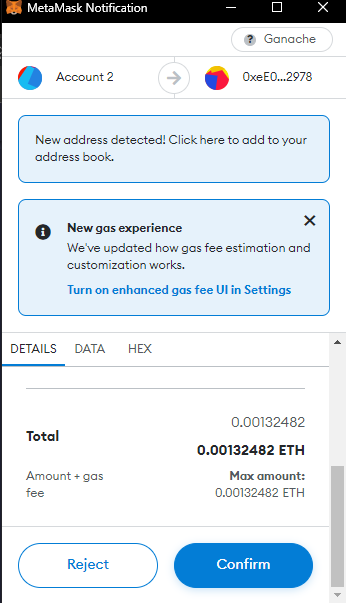
return storedData;

}

}

**Output:**





**Practical No.: 09**

**Aim: Write a program in solidity to create a structured student with Roll no, Name, Class, Department, Course enrolled as variables.**

1. **Add information of 5 students.**
2. **Search for a student using Roll no**
3. **Display all information**

**Program:**

pragma solidity >=0.7.0 <0.9.0;

pragma experimental ABIEncoderV2;

contract students

{

struct Student

{

uint rn;

string name;

string class;

string department;

string course;

}

Student[] student;

uint count;

constructor()

{

count=0;

}

function addstudentInfo(uint rollNumber, string memory name, string memory class, string memory dept, string memory course )public

{

student.push(Student(rollNumber,name,class,dept, course));

}

function getstudent(uint rollNumber ) public view returns (uint, string memory)

{

uint i =0;

for (i=0;i<student.length;i++)

{

if (student[i].rn == rollNumber)

{

return(student[i].rn, student[i].name);

}

}

return(student[0].rn, student[0].name);

}

function displayAllInfo() public view returns (Student[]memory)

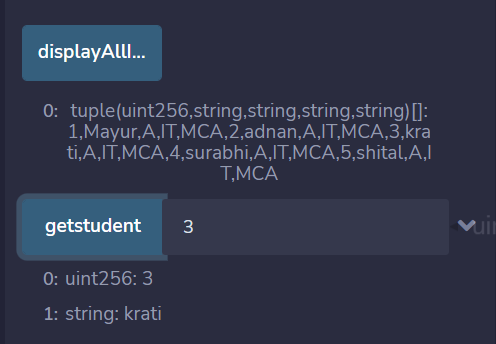
{

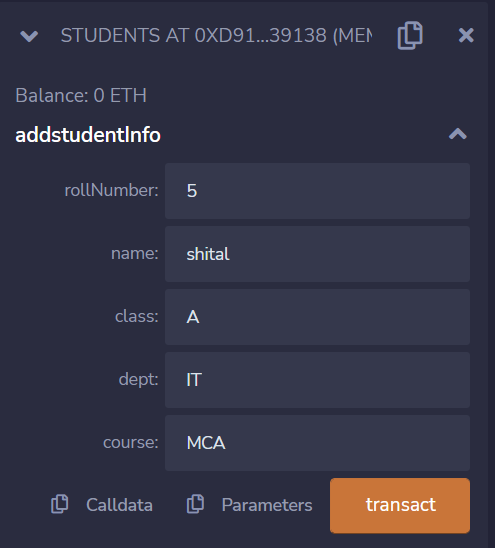
return student;

}

}

**Output:**





**Practical No.: 10**

**Aim: Create Daps Voting Process using Solidity smart contract and web3**

# Program:

## - Election.sol

pragma solidity 0.4.25;

contract Election {

// Model a Candidate

struct Candidate {

uint id;

string name;

uint voteCount;

}

// Store accounts that have voted

mapping(address => bool) public voters;

// Store Candidates // Fetch Candidate

mapping(uint => Candidate) public candidates;

// Store Candidates Count

uint public candidatesCount;

// voted event

event votedEvent (uint indexed \_candidateId); constructor () public {

addCandidate("Candidate 1");

addCandidate("Candidate 2");

}

function addCandidate (string \_name) private { candidatesCount ++;

candidates[candidatesCount] = Candidate(candidatesCount, \_name, 0);

}

function vote (uint \_candidateId) public {

// require that they haven't voted before require(!voters[msg.sender]);

// require a valid candidate

require(\_candidateId > 0 && \_candidateId <= candidatesCount);

// record that voter has voted voters[msg.sender] = true;

// update candidate vote Count candidates[\_candidateId].voteCount ++;

// trigger voted event

emit votedEvent(\_candidateId);

}}

#### **index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Election Results</title>

<!-- Bootstrap -->

<link href="css/bootstrap.min.css" rel="stylesheet">

</head>

<body>

<div class="container" style="width: 650px;">

<div class="row">

<div class="col-lg-12">

<h1 class="text-center">Election Results</h1>

<hr/>

<br/>

<div id="loader">

<p class="text-center">Loading...</p>

</div>

<div id="content" style="display: none;">

<table class="table">

<thead>

<tr>

<th scope="col">#</th>

<th scope="col">Name</th>

<th scope="col">Votes</th>

</tr>

</thead>

<tbody id="candidatesResults">

</tbody>

</table>

<hr/>

<form onSubmit="App.castVote(); return false;">

<div class="form-group">

<label for="candidatesSelect">Select Candidate</label>

<select class="form-control" id="candidatesSelect">

</select>

</div>

<button type="submit" class="btn btn-primary">Vote</button>

<hr />

</form>

<p id="accountAddress" class="text-center"></p>

</div>

</div>

</div>

</div>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"><

/script>

<!-- Include all compiled plugins (below), or include individual files as needed

-->

<script src="js/bootstrap.min.js"></script>

<script src="js/web3.min.js"></script>

<script src="js/truffle-contract.js"></script>

<script src="js/app.js"></script>

</body>

</html>

#### **app.js**

App = { web3Provider: null, contracts: {}, account: '0x0', hasVoted: false, init: function() {

return App.initWeb3();

},

initWeb3: function() {

// TODO: refactor conditional if (typeof web3 !== 'undefined') {

// If a web3 instance is already provided by Meta Mask. App.web3Provider = web3.currentProvider;

web3 = new Web3(web3.currentProvider);

} else {

// Specify default instance if no web3 instance provided App.web3Provider = new

Web3.providers.HttpProvider('http://localhost:7545'); web3

= new Web3(App.web3Provider);

}

return App.initContract();

},

initContract: function() {

$.getJSON("Election.json", function(election) {

// Instantiate a new truffle contract from the artifact App.contracts.Election = TruffleContract(election);

// Connect provider to interact with contract App.contracts.Election.setProvider(App.web3Provider); App.listenForEvents(); return App.render();

});

},

// Listen for events emitted from the contract listenForEvents: function() {

App.contracts.Election.deployed().then(function(instance) {

// Restart Chrome if you are unable to receive this event

// This is a known issue with Metamask

// https://github.com/MetaMask/metamask-extension/issues/2393

instance.votedEvent({}, { fromBlock: 0, toBlock: 'latest'

}).watch(function(error, event) { console.log("event triggered", event)

// Reload when a new vote is recorded App.render();

});

});

},

render: function() {

var electionInstance;

var loader = $("#loader"); var content = $("#content"); loader.show();

content.hide();

// Load account data web3.eth.getCoinbase(function(err,

account) {

if (err === null) { App.account = account;

$("#accountAddress").html("Your Account: " + account);

}

});

// Load contract data App.contracts.Election.deployed().then(function(instance)

{

electionInstance = instance;

return electionInstance.candidatesCount();

}).then(function(candidatesCount) {

var candidatesResults = $("#candidatesResults"); candidatesResults.empty();

var candidatesSelect = $('#candidatesSelect'); candidatesSelect.empty();

for (var i = 1; i <= candidatesCount; i++) { electionInstance.candidates(i).then(function(candidate) {

var id = candidate[0];

var name = candidate[1];

var voteCount = candidate[2];

// Render candidate Result

var candidateTemplate = "<tr><th>" + id + "</th><td>" + name + "</td><td>" + voteCount + "</td></tr>"

candidatesResults.append(candidateTemplate);

// Render candidate ballot option

var candidateOption = "<option value='" + id + "' >" + name + "</ option>"

candidatesSelect.append(candidateOption);

});

}

return electionInstance.voters(App.account);

}).then(function(hasVoted) {

// Do not allow a user to vote if(hasVoted)

{

$('form').hide();

}

loader.hide();

content.show();

}).catch(function(error) { console.warn(error);

});

},

castVote: function() {

var candidateId = $('#candidatesSelect').val(); App.contracts.Election.deployed().then(function(instance) {

return instance.vote(candidateId, { from: App.account });

}).then(function(result) {

// Wait for votes to update

$("#content").hide();

$("#loader").show();

}).catch(function(err) { console.error(err);

});

}

};

$(function() {

$(window).load(function() { App.init();

});

});

#### **truffle-config.js**

module.exports = {

// See [<http://truffleframework.com/docs/advanced/configuration>](http://truffleframework.com/docs/advanced/configuration)

// for more about customizing your Truffle configuration! networks: { development: {

host: "127.0.0.1",

port: 7545,

network\_id: "\*" // Match any network id

}

},

compilers: { solc: {

version: '0.4.25', optimizer: {

enabled: true, runs: 200

}

}

}

};

#### **bs-config.js**

{

"server": {

"baseDir": ["./src", "./build/contracts"]

}

}

#### **package.json**

{

"name": "election",

"version": "1.0.0",

"description": "",

"main": "truffle.js",

"directories": {

"test": "test"

},

"scripts": {

"dev": "lite-server",

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "",

"license": "ISC",

"devDependencies": {

"lite-server": "^2.3.0",

"truffle": "5.0.0-beta.0"

}

}

#### **2\_deploy\_contracts.js**

var Election = artifacts.require("./Election.sol");

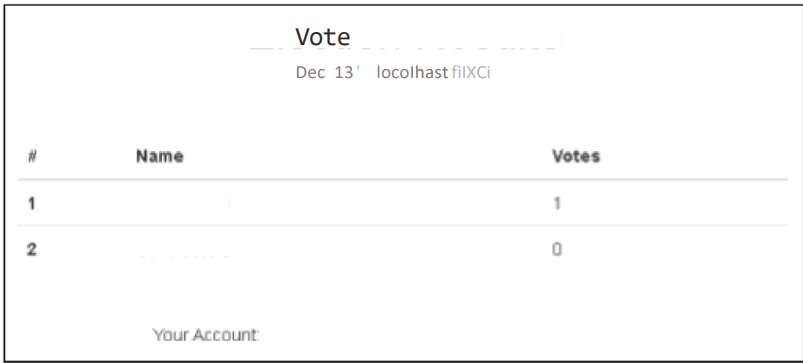
module.exports = function(deployer) {

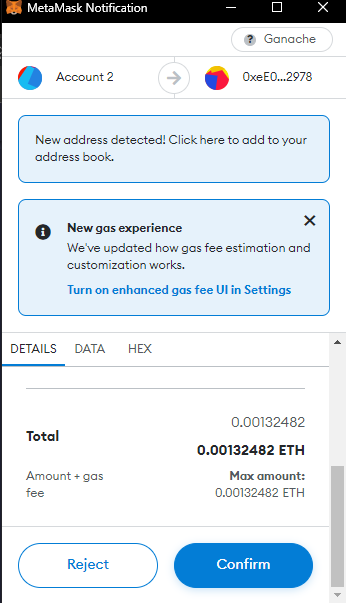
deployer.deploy(Election);

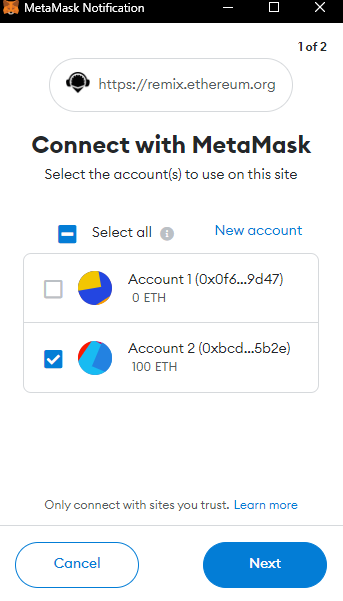
};

**Output:**









**Practical No.: 11**

**Mini Project**

**App.js**

import React, {useEffect,useState } from 'react';

import './App.css';

import Axios from 'axios';

import Coin from './components/Coin';

function App() {

const [listOfCoins,setListOfCoins]=useState([]);

const [searchWord, setSearchWord] = useState("");

useEffect(()=>{

Axios.get("https://api.coinstats.app/public/v1/coins?skip=0&limit=10").then(

(Response)=>{

setListOfCoins(Response.data.coins);

}

);

},[]);

const filteredCoins = listOfCoins.filter((coin) => {

return coin.name.toLowerCase().includes(searchWord.toLowerCase());

});

return (

<div className="App">

<div className="cryptoHeader">

<input

type="text"

placeholder="Bitcoin..."

onChange={(event) => {

setSearchWord(event.target.value);

}}

/>

</div>

<div className="cryptoDisplay">

{filteredCoins.map((coin) => {

return (

<Coin

name={coin.name}

icon={coin.icon}

price={coin.price}

symbol={coin.symbol}

/>

);

})}

</div>

</div>

);

}

export default App;

**App.css**

.App {

height: auto;

width: 100vw;

font-family: Arial, Helvetica, sans-serif;

}

body {

padding: 0%;

margin: 0%;

}

.cryptoHeader {

width: 100%;

height: 200px;

background-color: rgb(255, 196, 0);

display: flex;

justify-content: center;

align-items: center;

}

.cryptoHeader input {

width: 50%;

height: 50px;

border: none;

border-radius: 5px;

background-color: rgb(255, 255, 255);

font-size: 20px;

font-weight: bold;

color: rgb(0, 0, 0);

text-align: center;

margin-top: 10px;

}

.cryptoDisplay {

margin-top: 20px;

display: flex;

justify-content: center;

align-items: center;

flex-direction: column;

}

.coin {

width: 400px;

height: 300px;

background-color: rgb(36, 36, 36);

color: white;

box-shadow: rgba(0, 0, 0, 0.24) 0px 3px 8px;

border-radius: 10px;

margin: 20px;

text-align: center;

}

.coin img {

height: 100px;

}

**Coin.js**

import React from 'react'

function Coin({name,icon,price,symbol}) {

return (

<div className="coin">

<h1> Name: {name}</h1>

<img src={icon}/>

<h3> Price: {price}</h3>

<h3> Symbol: {symbol}</h3>

</div>

);

}

export default Coin;

**index.js**

import React from "react";

import ReactDOM from "react-dom/client";

import App from "./App";

import reportWebVitals from "./reportWebVitals"

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

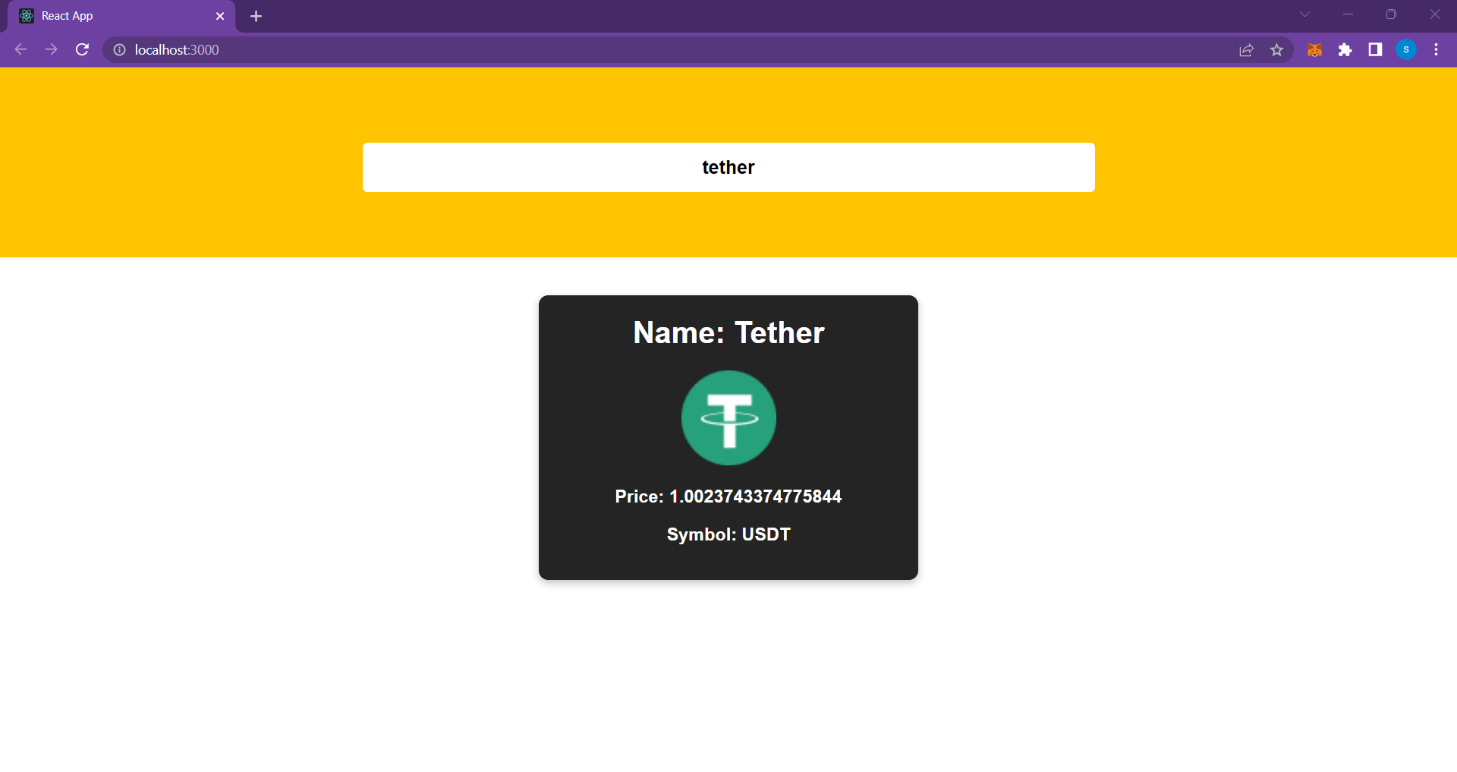
// If you want to start measuring performance in your app, pass a function

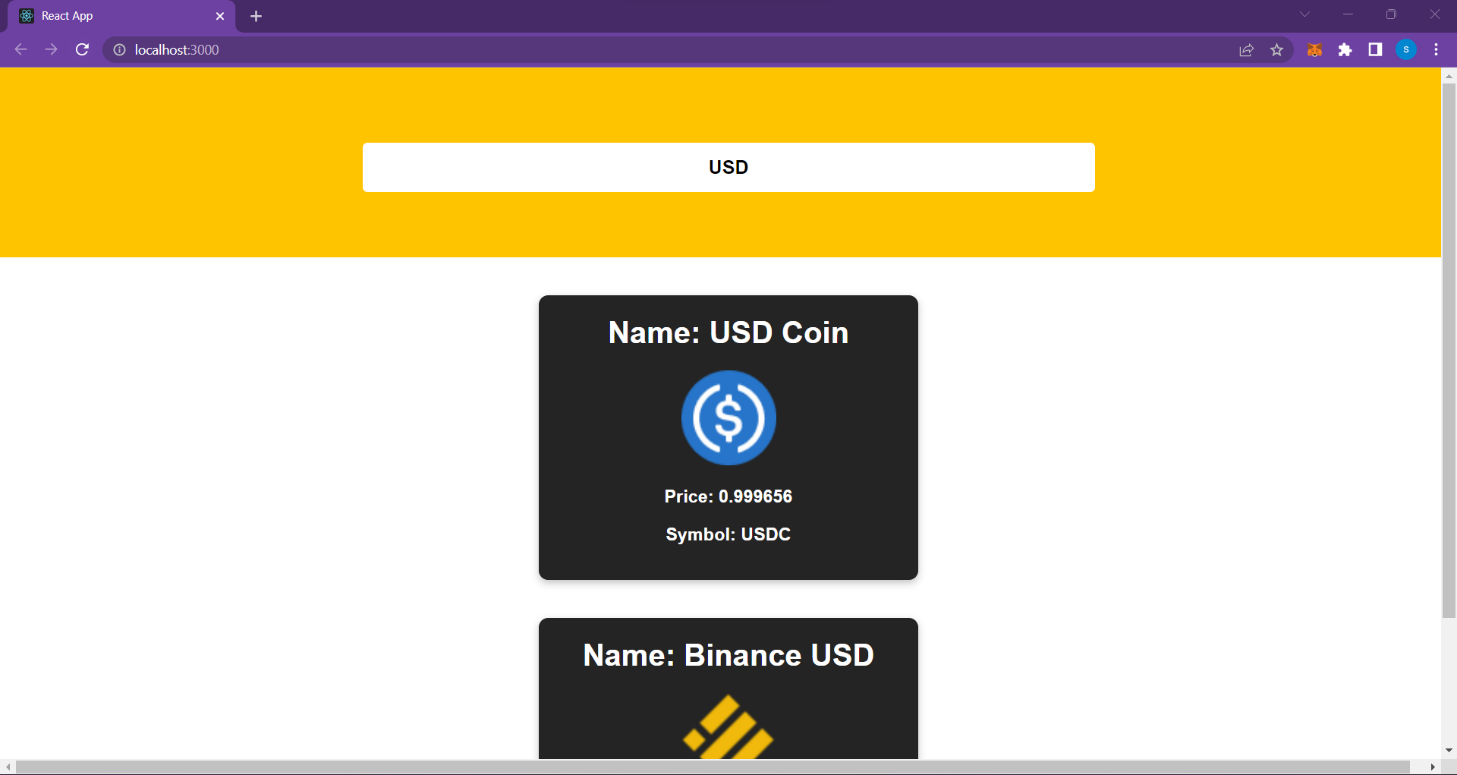
// to log results (for example: reportWebVitals(console.log))

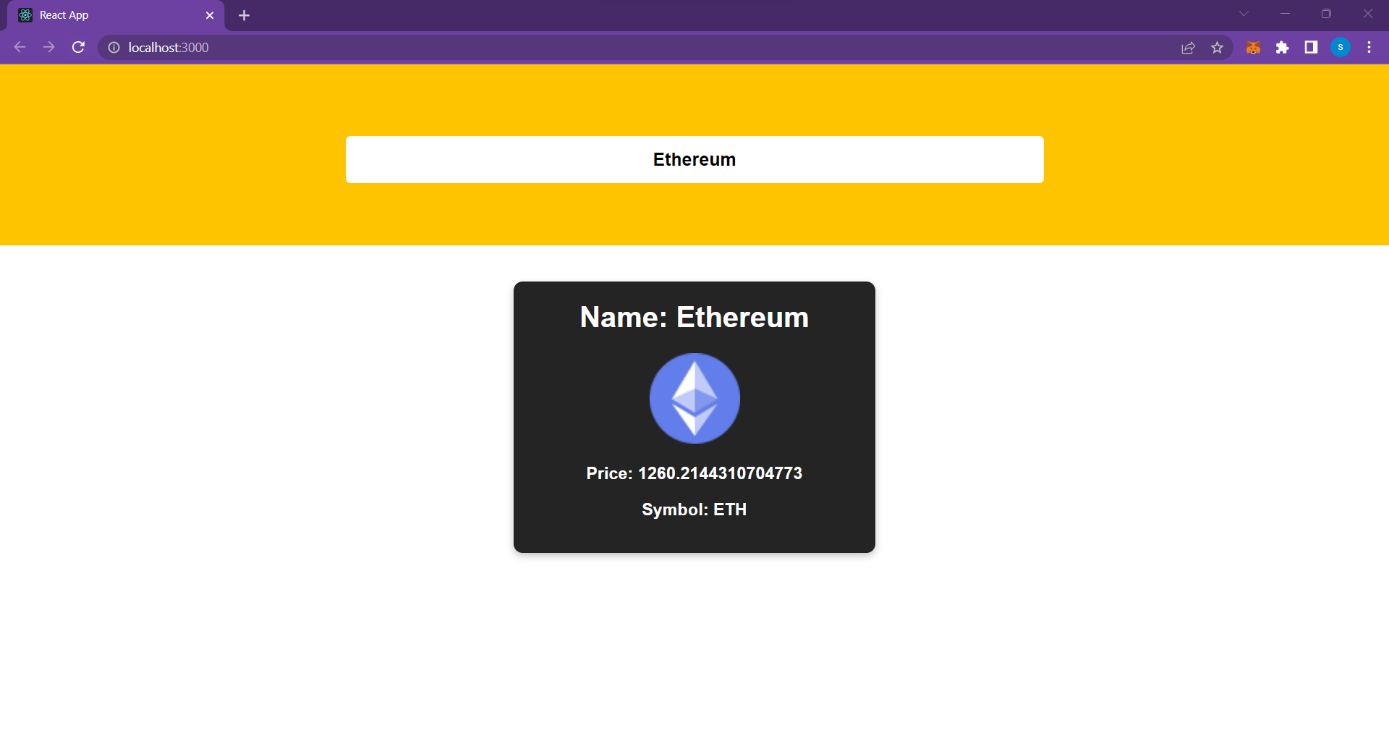
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**SCREENSHOTS:**

****

****

****