Name of Student : Pushkar Sane		
Roll Number : 45		**LAB Assignment Number: 9**
Title of LAB Assignment : Creation of Dapp in Ethereum.		
DOP : 22-10-2024		**DOS : 25-09-2024**
CO Mapped : **CO5**	**PO Mapped:** **PO1,PO2, PO3, PO4, PO7, PO9, PSO1, PSO2**	**Signature:**

PRACTICAL 9

Aim: Creation of Dapp in Ethereum

Theory:

Decentralized Applications:

Decentralized Applications (or DApps) are applications that do not rely on a centralized backend running in AWS or Azure that power traditional web and mobile applications (outside of hosting the frontend code itself). Instead, the application interacts directly with a blockchain which can be thought of distributed cluster of nodes analogous to applications interacting directly with a "masterless" cluster of Cassandra nodes with full replication on every peer in an untrusted peer-to-peer network.

These blockchain nodes do not require a leader which would defeat the purpose of being truly decentralized. Unlike leader election in various consensus protocols like Raft and Paxos, blockchain transactions are sent to and processed by "random" nodes via Proof of Work or Proof of Stake. These nodes are untrusted nodes running in an arbitrary sized network on various computer devices around the world. Such technology can enable true decentralized ledgers and systems of records.

DApps are the frontend apps which interact with these blockchain over an API. For Ethereum, this API is a JSON-RPC layer called the Ethereum Web3 API which Moesif supports natively.

The fundamental requirements for DAPPS:

1. Open Source: dApps should be open source and its codebase should be freely available for all. Any changes in the structure or working of the app should only be taken by agreement of the majority.

2. Decentralized: dApps should be decentralized with all the information and operations stored on a public and decentralized Blockchain which would ensure security and transparency.

3. Incentive: dApps should offer some sort of incentives to their users in the form of cryptographic tokens. These are a sort of liquid assets and they provide incentives for users to support the Blockchain dApp ecosystem.

4. Protocol: dApps should have a particular protocol to demonstrate proof of value. This means showing the value of a particular process in a way that can be easily verified by others.

Ethereum is a cryptocurrency platform in the market just like Bitcoin. It is an open-source & decentralized blockchain featuring working on smart contracts. It has its own cryptocurrency known as ether. The smart contracts in Ethereum are written in solidity.
TestRPC The Ethereum TestRPC is like a manual emulator for blockchain. It provides blockchain interaction without running on an actual Ethereum node. It also provides all the features required for testing of your blockchain. It is based on Node.js.

Web3.js Web3.js is an Ethereum-based JavaScript API that allows us to interact with a local or remote Ethereum node using different servers such as HTTP. It interacts with the Ethereum blockchain and can retrieve data from the blockchain as required by the developer.

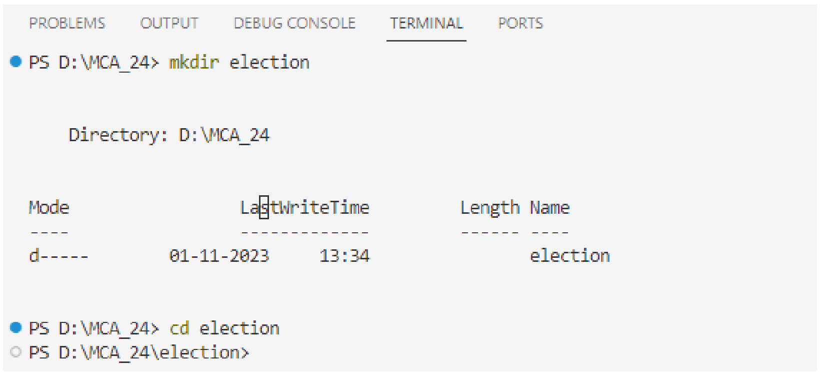
Advantages:

- Many of the advantages of dApps center around the program's ability to safeguard user privacy. With decentralized apps, users do not need to submit their personal information to use the function the app provides. DApps use smart contracts to complete the transaction between two anonymous parties without the need to rely on a central authority.
- Proponents interested in free speech point out that dApps can be developed as alternative social media platforms. A decentralized social media platform would be resistant to censorship because no single participant on the blockchain can delete messages or block messages from being posted.
- Ethereum is a flexible platform for creating new dApps, providing the infrastructure needed for developers to focus their efforts on finding innovative uses for digital applications. This could enable rapid deployment of dApps in a variety of industries including banking and finance, gaming, social media, and online shopping.

1. VOTING PROCESS

1.create a project directory for our dApp in the command line like this:

```
$ mkdir election
$ cd election
```

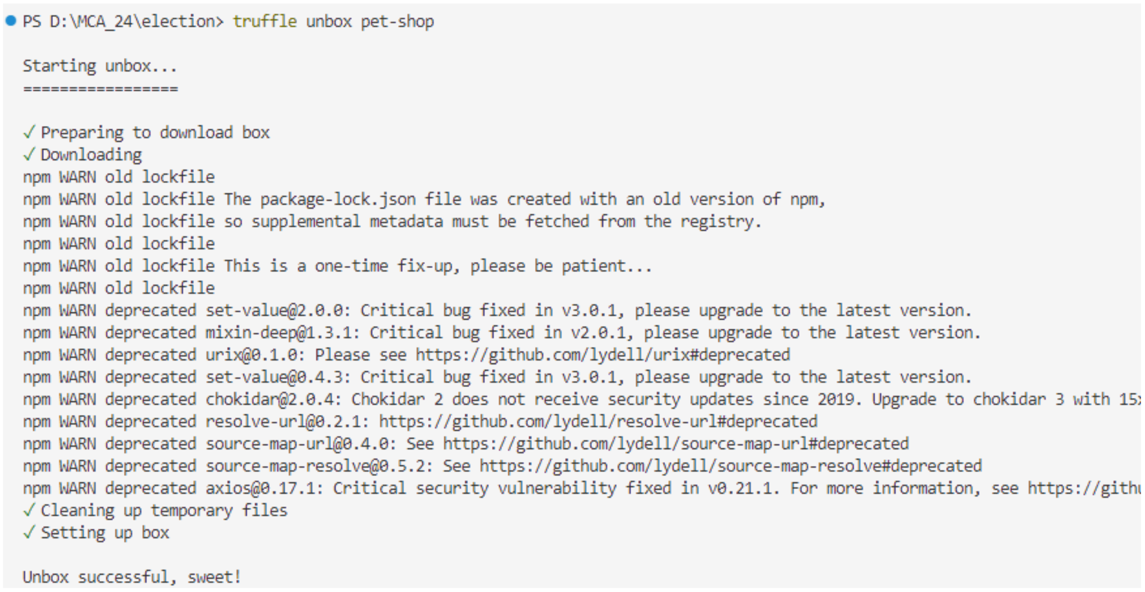


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

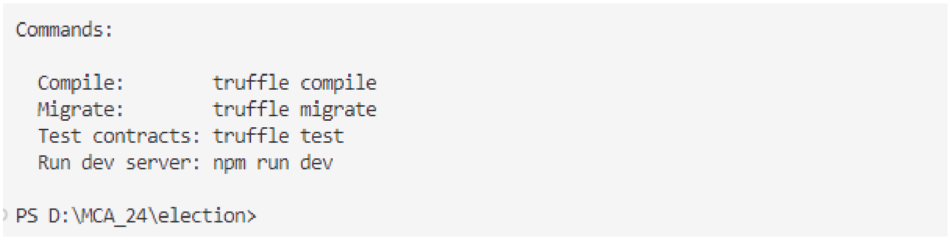
```
PS D:\MCA_24> mkdir election


    Directory: D:\MCA_24


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        01-11-2023     13:34                election


PS D:\MCA_24> cd election
PS D:\MCA_24\election>
```

2. From within your project directory, install the pet shop box from the command line like this:

$ truffle unbox pet-shop



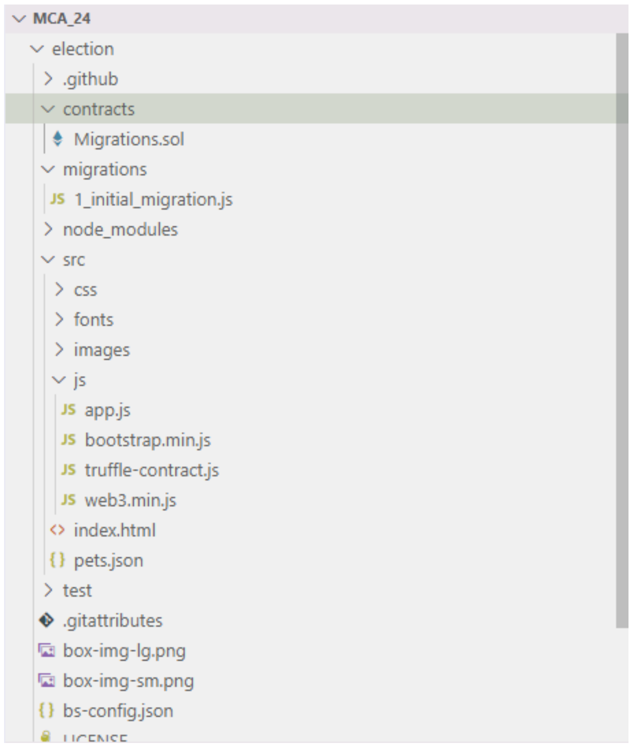
```
PS D:\MCA_24\election> truffle unbox pet-shop

Starting unbox...
=================

✓ Preparing to download box
✓ Downloading
npm WARN old lockfile
npm WARN old lockfile The package-lock.json file was created with an old version of npm,
npm WARN old lockfile so supplemental metadata must be fetched from the registry.
npm WARN old lockfile
npm WARN old lockfile This is a one-time fix-up, please be patient...
npm WARN old lockfile
npm WARN deprecated set-value@2.0.0: Critical bug fixed in v3.0.1, please upgrade to the latest version.
npm WARN deprecated mixin-deep@1.3.1: Critical bug fixed in v2.0.1, please upgrade to the latest version.
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated set-value@0.4.3: Critical bug fixed in v3.0.1, please upgrade to the latest version.
npm WARN deprecated chokidar@2.0.4: Chokidar 2 does not receive security updates since 2019. Upgrade to chokidar 3 with 15x
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated source-map-url@0.4.0: See https://github.com/lydell/source-map-url#deprecated
npm WARN deprecated source-map-resolve@0.5.2: See https://github.com/lydell/source-map-resolve#deprecated
npm WARN deprecated axios@0.17.1: Critical security vulnerability fixed in v0.21.1. For more information, see https://githu
✓ Cleaning up temporary files
✓ Setting up box

Unbox successful, sweet!
```



```
Commands:

  Compile:        truffle compile
  Migrate:        truffle migrate
  Test contracts: truffle test
  Run dev server: npm run dev

PS D:\MCA_24\election>
```

3.Open Ganache

4.Start developing the solidity code, test code and client-side code for the application:



```
MCA_24
  election
    .github
    contracts
      Migrations.sol
    migrations
      1_initial_migration.js
    node_modules
    src
      css
      fonts
      images
      js
        app.js
        bootstrap.min.js
        truffle-contract.js
        web3.min.js
      index.html
      pets.json
    test
    .gitattributes
    box-img-lg.png
    box-img-sm.png
    bs-config.json
```

Code:

Contracts-> Election.sol

```
pragma solidity >=0.4.2 <=0.8.0;
contract Election {
// Model a Candidate
struct Candidate {
uint id;
string name;
uint voteCount;
}
// Read/write candidates
mapping(uint => Candidate) public candidates;
// Store accounts that have voted
mapping(address => bool) public voters;
// Store Candidates Count
uint public candidatesCount;
constructor() public {
addCandidate("Atharva");
addCandidate("Rutik");
}
event votedEvent(uint indexed _candidateId);
function addCandidate (string memory _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);
}
}
```

migration-> 2_deploy_contracts.js

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

module.exports = function(deployer) {
  deployer.deploy(Election);
};
```

test-> election.js

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

```
contract("Election", function(accounts) {
var electionInstance;
it("initializes with two candidates", function() {
return Election.deployed().then(function(instance) {
return instance.candidatesCount();
}).then(function(count) {
assert.equal(count, 2);
});
});
it("it initializes the candidates with the correct values", function() {
return Election.deployed().then(function(instance) {
electionInstance = instance;
return electionInstance.candidates(1);
}).then(function(candidate) {
assert.equal(candidate[0], 1, "contains the correct id");
assert.equal(candidate[1], "Candidate 1", "contains the correct name");
assert.equal(candidate[2], 0, "contains the correct votes count");
return electionInstance.candidates(2);
}).then(function(candidate) {
assert.equal(candidate[0], 2, "contains the correct id");
assert.equal(candidate[1], "Candidate 2", "contains the correct name");
assert.equal(candidate[2], 0, "contains the correct votes count");
});
});
it("allows a voter to cast a vote", function() {
  return Election.deployed().then(function(instance) {
  electionInstance = instance;
  candidateId = 1;
  return electionInstance.vote(candidateId, { from: accounts[0] });
  }).then(function(receipt) {
  assert.equal(receipt.logs.length, 1, "an event was triggered");
  assert.equal(receipt.logs[0].event, "votedEvent", "the event type is correct");
```

```
     assert.equal(receipt.logs[0].args._candidateId.toNumber(), candidateId, "the candidate id is
correct");
    return electionInstance.voters(accounts[0]);
    }).then(function(voted) {
    assert(voted, "the voter was marked as voted");
    return electionInstance.candidates(candidateId);
    }).then(function(candidate) {
    var voteCount = candidate[2];
    assert.equal(voteCount, 1, "increments the candidate's vote count");
    })
});
it("throws an exception for invalid candidates", function() {
return Election.deployed().then(function(instance) {
electionInstance = instance;
return electionInstance.vote(99, { from: accounts[1] })
}).then(assert.fail).catch(function(error) {
assert(error.message.indexOf('revert') >= 0, "error message must contain revert");
return electionInstance.candidates(1);
}).then(function(candidate1) {
var voteCount = candidate1[2];
assert.equal(voteCount, 1, "candidate 1 did not receive any votes");
return electionInstance.candidates(2);
}).then(function(candidate2) {
var voteCount = candidate2[2];
assert.equal(voteCount, 0, "candidate 2 did not receive any votes");
});
});
it("throws an exception for double voting", function() {
return Election.deployed().then(function(instance) {
electionInstance = instance;
candidateId = 2;
electionInstance.vote(candidateId, { from: accounts[1] });
return electionInstance.candidates(candidateId);
```

```
}).then(function(candidate) {
var voteCount = candidate[2];
assert.equal(voteCount, 1, "accepts first vote");
// Try to vote again
return electionInstance.vote(candidateId, { from: accounts[1] });
}).then(assert.fail).catch(function(error) {
assert(error.message.indexOf('revert') >= 0, "error message must contain revert");
return electionInstance.candidates(1);
}).then(function(candidate1) {
  var voteCount = candidate1[2];
assert.equal(voteCount, 1, "candidate 1 did not receive any votes");
return electionInstance.candidates(2);
}).then(function(candidate2) {
var voteCount = candidate2[2];
assert.equal(voteCount, 1, "candidate 2 did not receive any votes");
});
});
});
```

src-> 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">
   <!-- The above 3 meta tags *must* come first in the head; any other head content must come
*after* these tags -->
  <title>Pete's Pet Shop</title>
  <!-- Bootstrap -->
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
  <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
  <!--[if lt IE 9]>
    <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>
    <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
  <![endif]-->
</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>
```

src->js-> app.js

```
App = {
  web3Provider: null,
  contracts: {},
  account: '0x0',
  hasVoted: false,
  init: function() {
    return App.initWeb3();
},
initWeb3: function() {
// TODO: refactor conditional
if (typeof web3 !== 'undefined') {
```

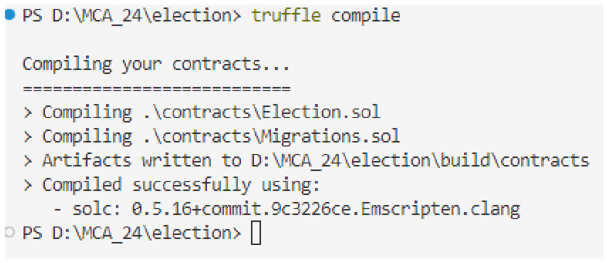
```
// If a web3 instance is already provided by Meta Mask.
App.web3Provider = web3.currentProvider;
web3 = new Web3(web3.currentProvider);
} else {
// Specify default instance if no web3 instance provided
App.web3Provider = new
Web3.providers.HttpProvider('http://localhost:7545');
web3 = new Web3(App.web3Provider);
}
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);
  });
  }
  };
```

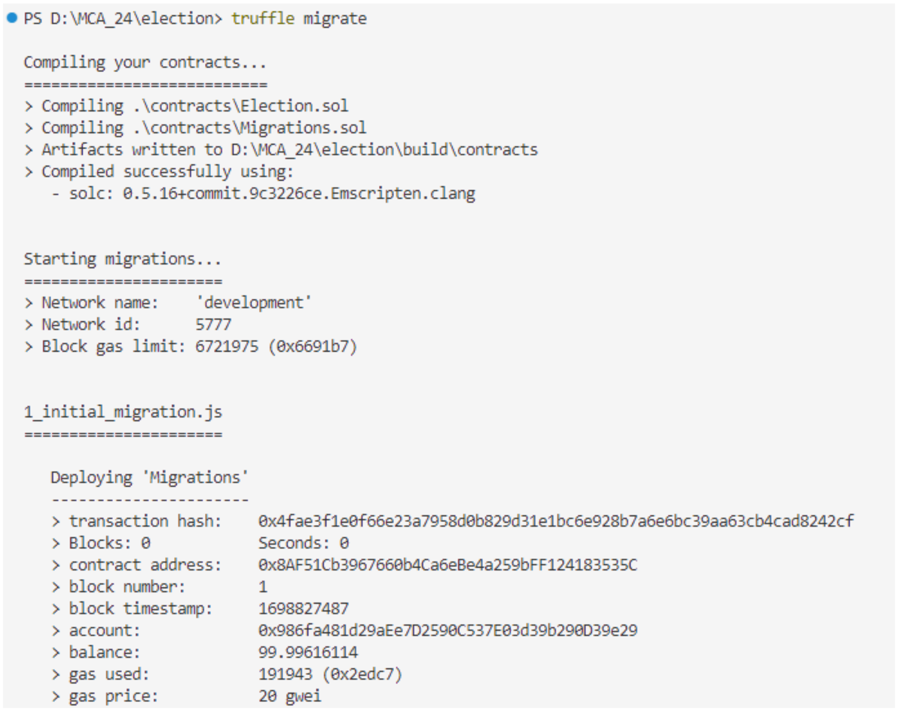
```
$(function() {
  $(window).load(function() {
  App.init();
  });
  });
```

Output:

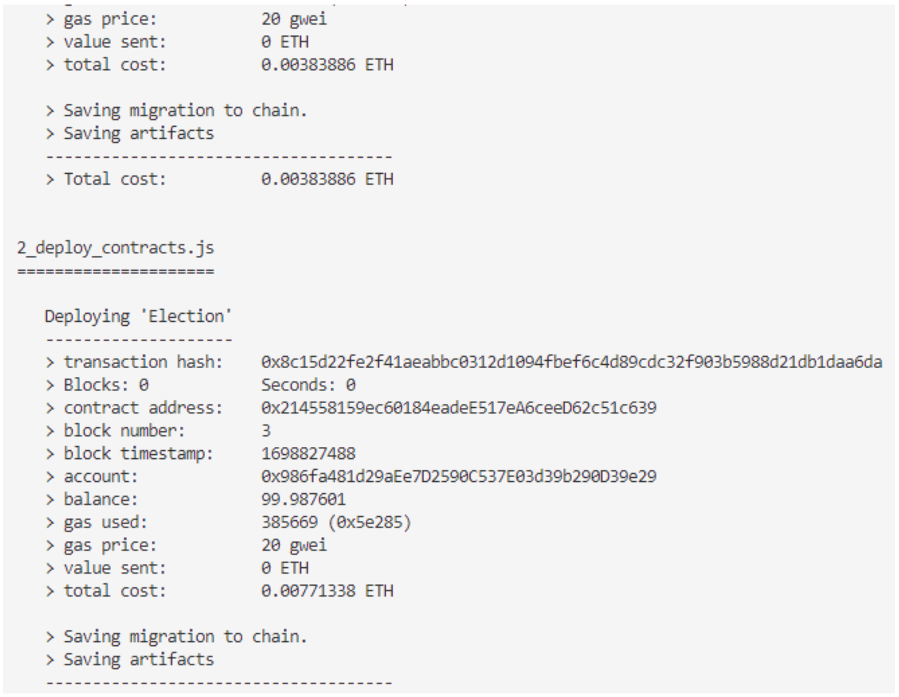
- **truffle compile**
- **truffle migrate**
- **truffle console**
- **Election.deployed().then(function(instance) { app = instance })**
- **Truffle test**
- **npm run dev**



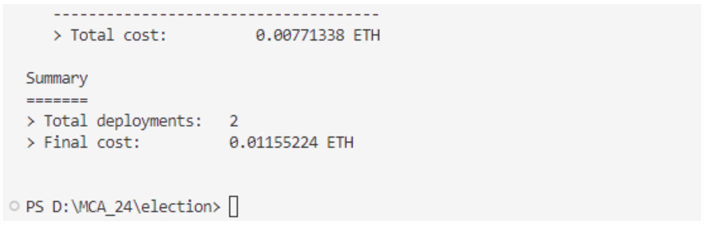
```
PS D:\MCA_24\election> truffle compile

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to D:\MCA_24\election\build\contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang
PS D:\MCA_24\election> 
```



```
● PS D:\MCA_24\election> truffle migrate

Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to D:\MCA_24\election\build\contracts
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang


Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)


1_initial_migration.js
======================

   Deploying 'Migrations'
   ----------------------
   > transaction hash:    0x4fae3f1e0f66e23a7958d0b829d31e1bc6e928b7a6e6bc39aa63cb4cad8242cf
   > Blocks: 0            Seconds: 0
   > contract address:    0x8AF51Cb3967660b4Ca6eBe4a259bFF124183535C
   > block number:        1
   > block timestamp:     1698827487
   > account:             0x986fa481d29aEe7D2590C537E03d39b290D39e29
   > balance:             99.99616114
   > gas used:            191943 (0x2edc7)
   > gas price:           20 gwei
```

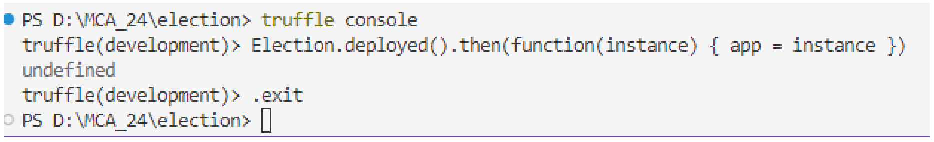


```
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00383886 ETH

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


2_deploy_contracts.js
=====================

   Deploying 'Election'
   --------------------
   > transaction hash:    0x8c15d22fe2f41aeabbc0312d1094fbef6c4d89cdc32f903b5988d21db1daa6da
   > Blocks: 0            Seconds: 0
   > contract address:    0x214558159ec60184eadeE517eA6ceeD62c51c639
   > block number:        3
   > block timestamp:     1698827488
   > account:             0x986fa481d29aEe7D2590C537E03d39b290D39e29
   > balance:             99.987601
   > gas used:            385669 (0x5e285)
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00771338 ETH

   > Saving migration to chain.
   > Saving artifacts
   -------------------------------------
```

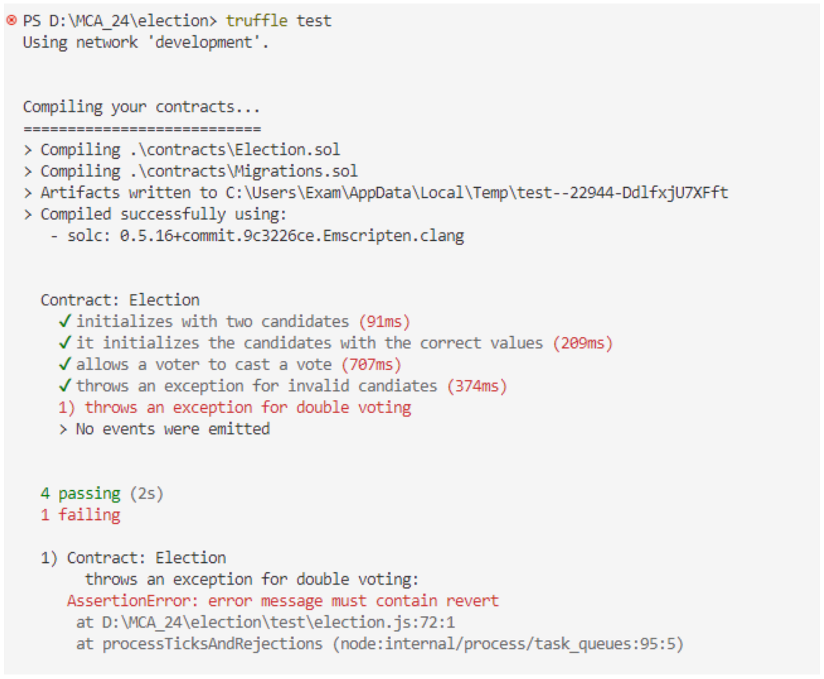


```
   -------------------------------------
   > Total cost:          0.00771338 ETH

Summary
=======
> Total deployments:   2
> Final cost:          0.01155224 ETH


○ PS D:\MCA_24\election> []
```

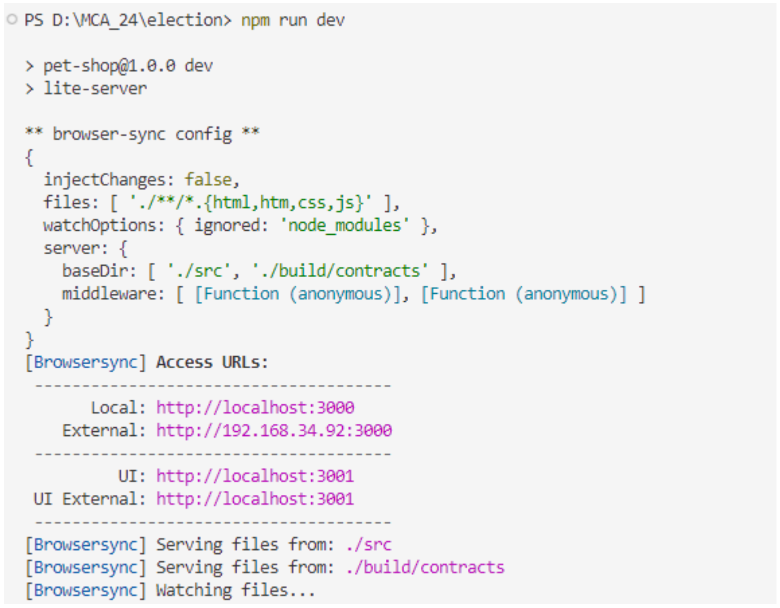
Election.deployed().then(function(instance) { app = instance })



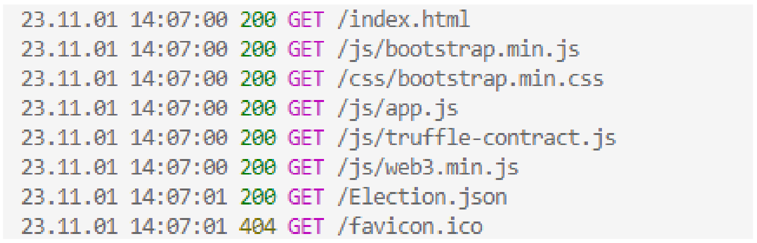
```
PS D:\MCA_24\election> truffle console
truffle(development)> Election.deployed().then(function(instance) { app = instance })
undefined
truffle(development)> .exit
PS D:\MCA_24\election> 
```



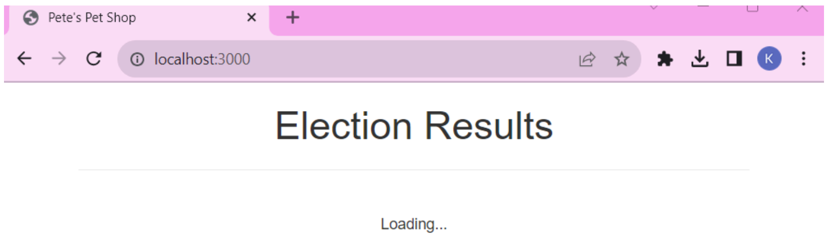
```
PS D:\MCA_24\election> truffle test
Using network 'development'.


Compiling your contracts...
===========================
> Compiling .\contracts\Election.sol
> Compiling .\contracts\Migrations.sol
> Artifacts written to C:\Users\Exam\AppData\Local\Temp\test--22944-DdlfxjU7XFft
> Compiled successfully using:
   - solc: 0.5.16+commit.9c3226ce.Emscripten.clang


  Contract: Election
    ✓ initializes with two candidates (91ms)
    ✓ it initializes the candidates with the correct values (209ms)
    ✓ allows a voter to cast a vote (707ms)
    ✓ throws an exception for invalid candiates (374ms)
    1) throws an exception for double voting
    > No events were emitted


  4 passing (2s)
  1 failing

  1) Contract: Election
       throws an exception for double voting:
     AssertionError: error message must contain revert
      at D:\MCA_24\election\test\election.js:72:1
      at processTicksAndRejections (node:internal/process/task_queues:95:5)
```



```
PS D:\MCA_24\election> npm run dev

> pet-shop@1.0.0 dev
> lite-server

** browser-sync config **
{
  injectChanges: false,
  files: [ './**/*.{html,htm,css,js}' ],
  watchOptions: { ignored: 'node_modules' },
  server: {
    baseDir: [ './src', './build/contracts' ],
    middleware: [ [Function (anonymous)], [Function (anonymous)] ]
  }
}
[Browsersync] Access URLs:
 --------------------------------------
       Local: http://localhost:3000
    External: http://192.168.34.92:3000
 --------------------------------------
          UI: http://localhost:3001
 UI External: http://localhost:3001
 --------------------------------------
[Browsersync] Serving files from: ./src
[Browsersync] Serving files from: ./build/contracts
[Browsersync] Watching files...
```

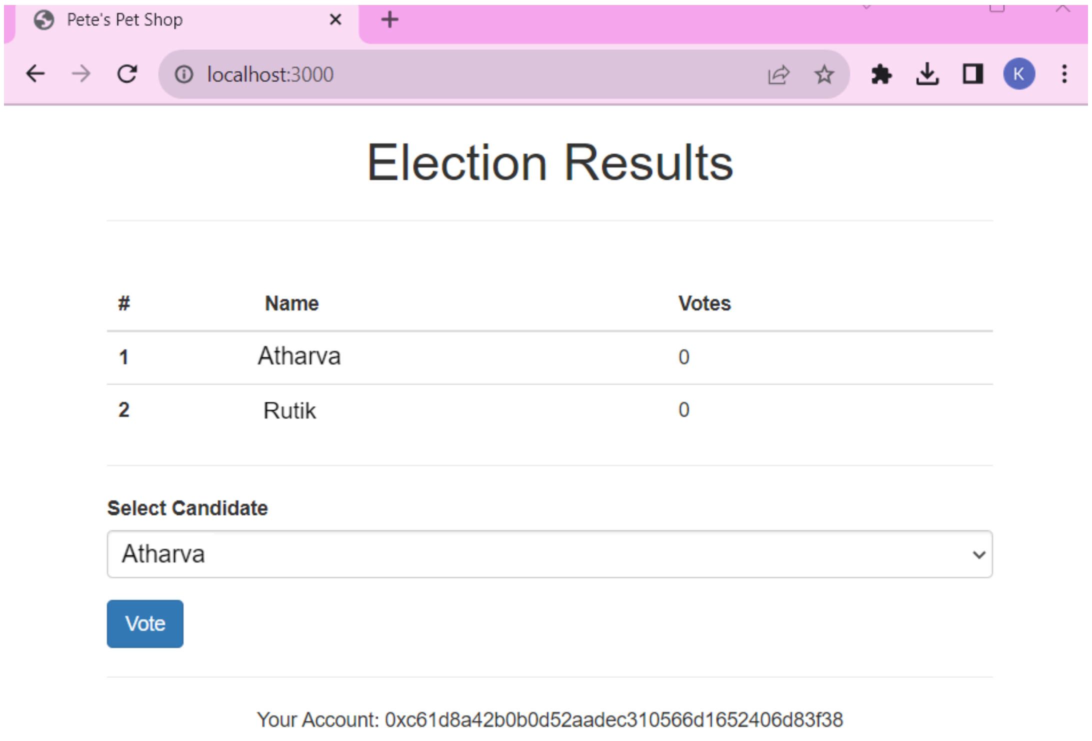


```
23.11.01 14:07:00 200 GET /index.html
23.11.01 14:07:00 200 GET /js/bootstrap.min.js
23.11.01 14:07:00 200 GET /css/bootstrap.min.css
23.11.01 14:07:00 200 GET /js/app.js
23.11.01 14:07:00 200 GET /js/truffle-contract.js
23.11.01 14:07:00 200 GET /js/web3.min.js
23.11.01 14:07:01 200 GET /Election.json
23.11.01 14:07:01 404 GET /favicon.ico
```



Election Results

Loading...



Election Results

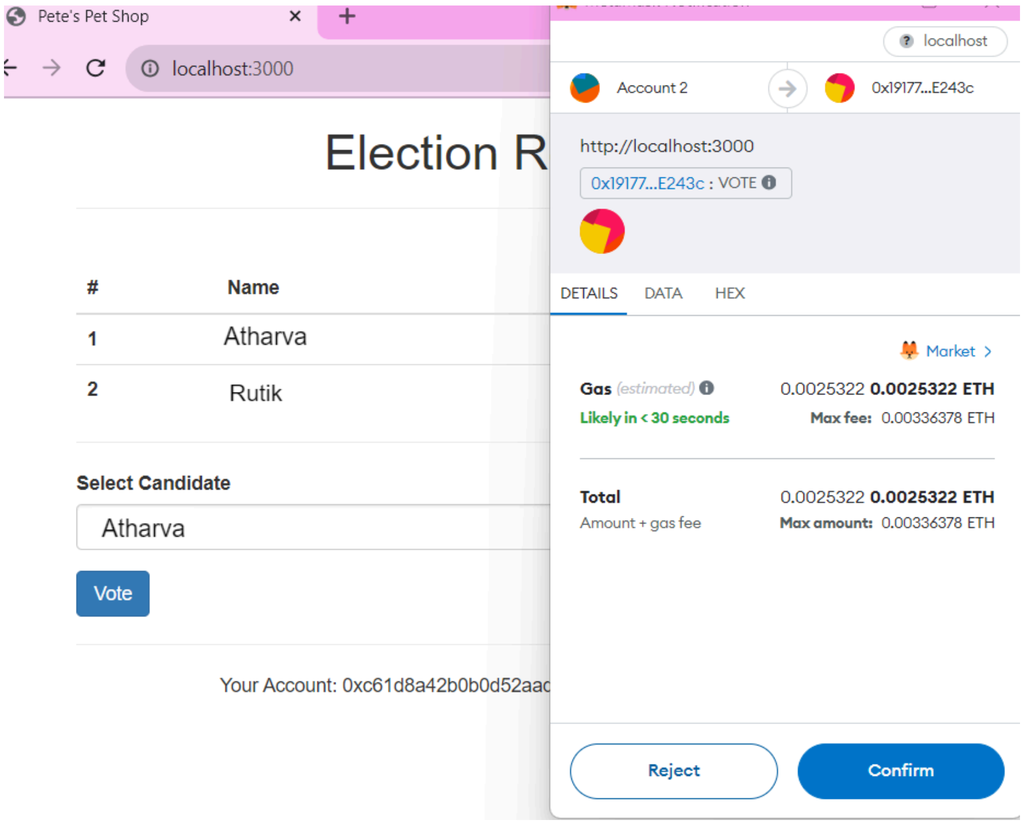
#	Name	Votes
1	Atharva	0
2	Rutik	0

Select Candidate

Atharva

Vote

Your Account: 0xc61d8a42b0b0d52aadec310566d1652406d83f38



Pete's Pet Shop

localhost:3000

Election R

#	Name
1	Atharva
2	Rutik

Select Candidate

Atharva

Vote

Your Account: 0xc61d8a42b0b0d52aac

localhost

Account 2 → 0x19177...E243c

http://localhost:3000

0x19177...E243c : VOTE

DETAILS DATA HEX

Market >

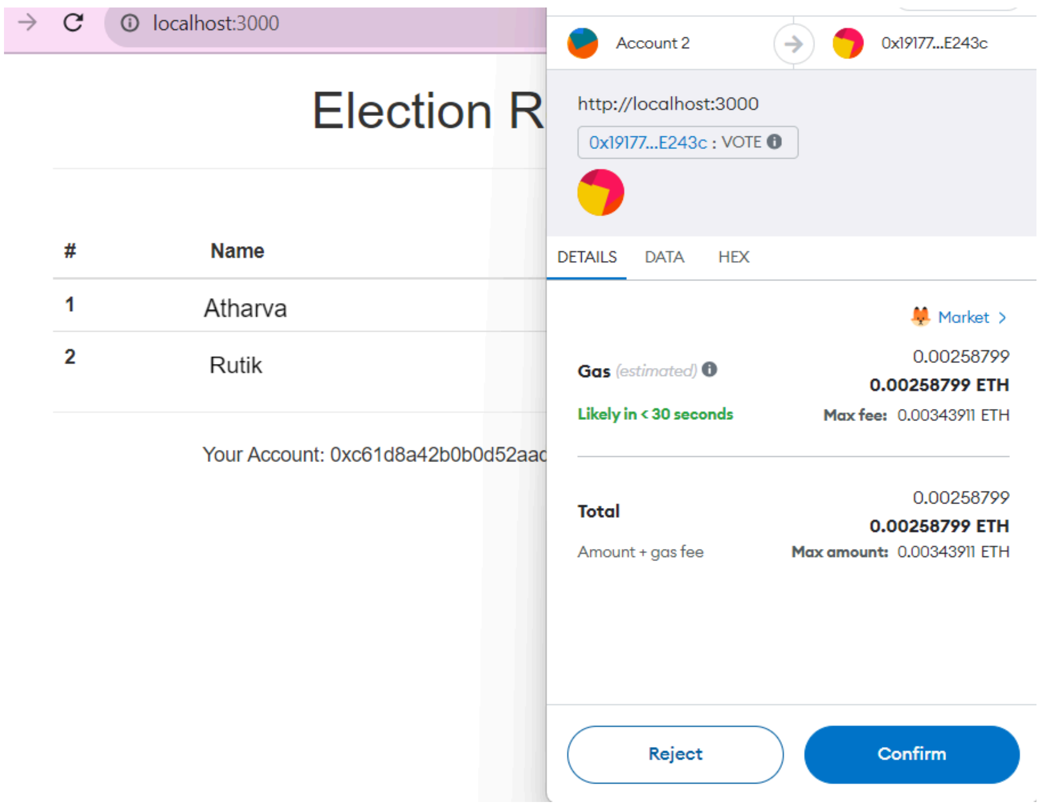
Gas (estimated) 0.0025322 **0.0025322 ETH**

Likely in < 30 seconds **Max fee:** 0.00336378 ETH

Total 0.0025322 **0.0025322 ETH**

Amount + gas fee **Max amount:** 0.00336378 ETH

Reject Confirm



localhost:3000

Election R

#	Name
1	Atharva
2	Rutik

Your Account: 0xc61d8a42b0b0d52aac

Account 2 → 0x19177...E243c

http://localhost:3000

0x19177...E243c : VOTE

DETAILS DATA HEX

Market >

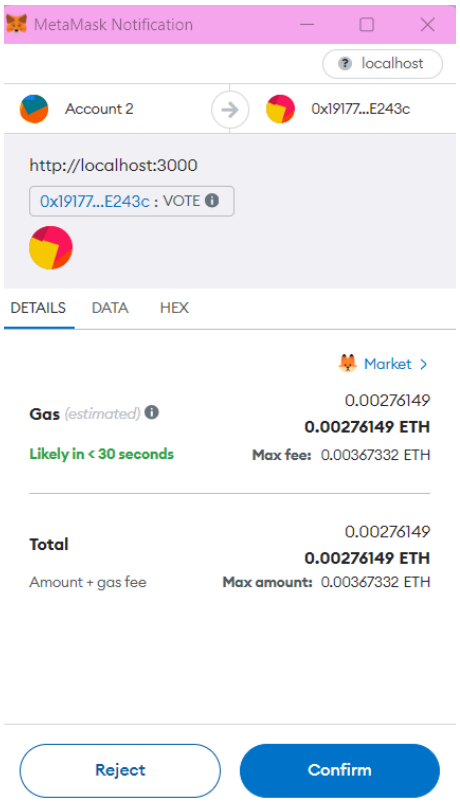
Gas (estimated) 0.00258799 **0.00258799 ETH**

Likely in < 30 seconds **Max fee:** 0.00343911 ETH

Total 0.00258799 **0.00258799 ETH**

Amount + gas fee **Max amount:** 0.00343911 ETH

Reject Confirm

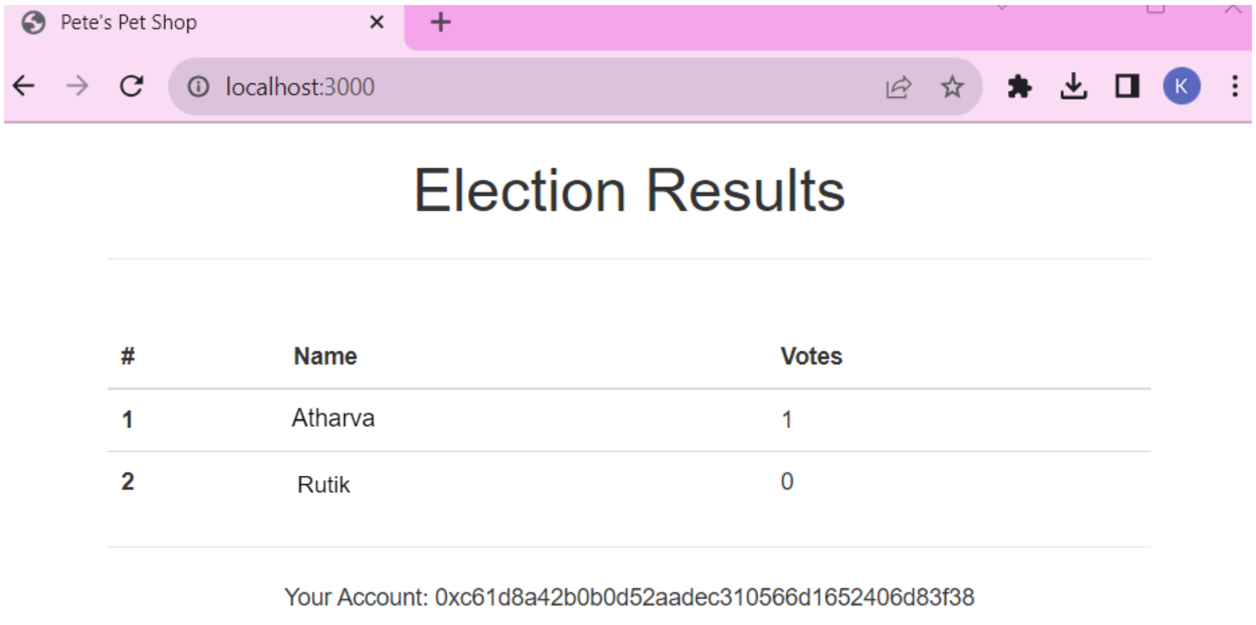


http://localhost:3000

0x19177...E243c : VOTE

Gas (estimated) 0.00276149
0.00276149 ETH
Likely in < 30 seconds Max fee: 0.00367332 ETH

Total 0.00276149
0.00276149 ETH
Amount + gas fee Max amount: 0.00367332 ETH

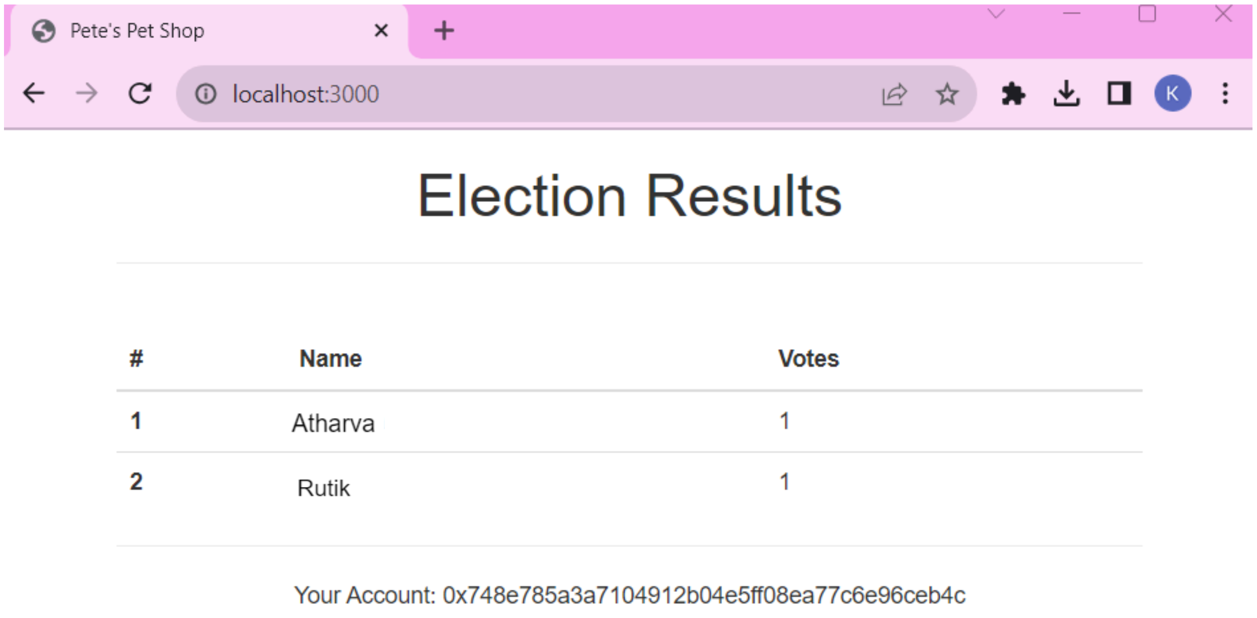


Election Results

#	Name	Votes
1	Atharva	1
2	Rutik	0

Your Account: 0xc61d8a42b0b0d52aadec310566d1652406d83f38

We can also create another account and vote again:



Election Results

#	Name	Votes
1	Atharva	1
2	Rutik	1

Your Account: 0x748e785a3a7104912b04e5ff08ea77c6e96ceb4c

Conclusion: I have successfully built DAPPS for the voting process .