**Zorawar khan**

**20PWBCS0737 / Sec ‘A’**

**Blockchain Assignment**

**README File**

Sir before this assignment I have done few steps already for learning Blockchain and Truffle out of my interest.

For example

Installing Node.Js,

Installing Ganache,

Installing Truffle,

and Metamask as well.

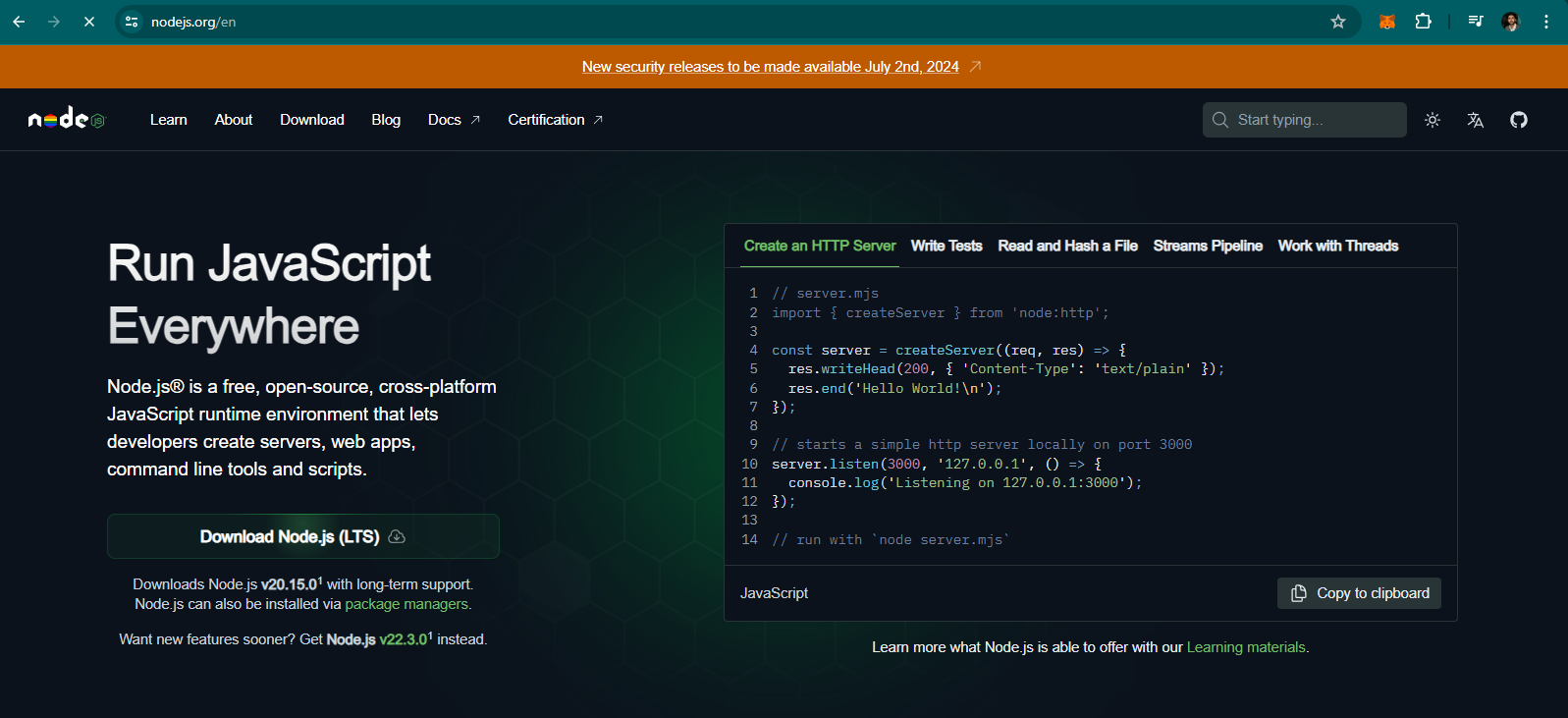
Mentioning **VScode** that I will be using for Truffle, code execution and interacting with the contract.

**I will show the files and apps that have already been installed.**

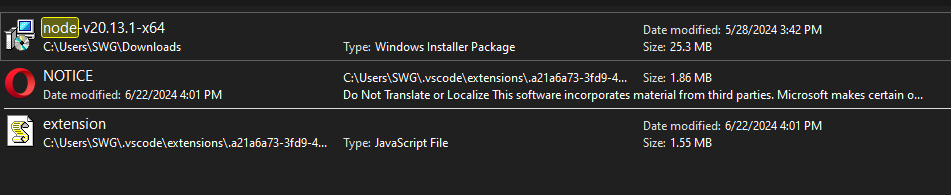
Have also done all the tasks of this contract on Remix, but didn’t mention because the required way of assignment have been used.

***Install Node.js: Download and install Node.js from nodejs.org.***

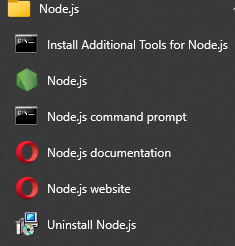
Here from the website

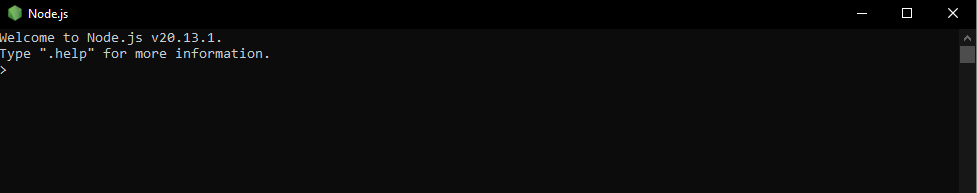


I downloaded Node.js



And here we got the installed node.js





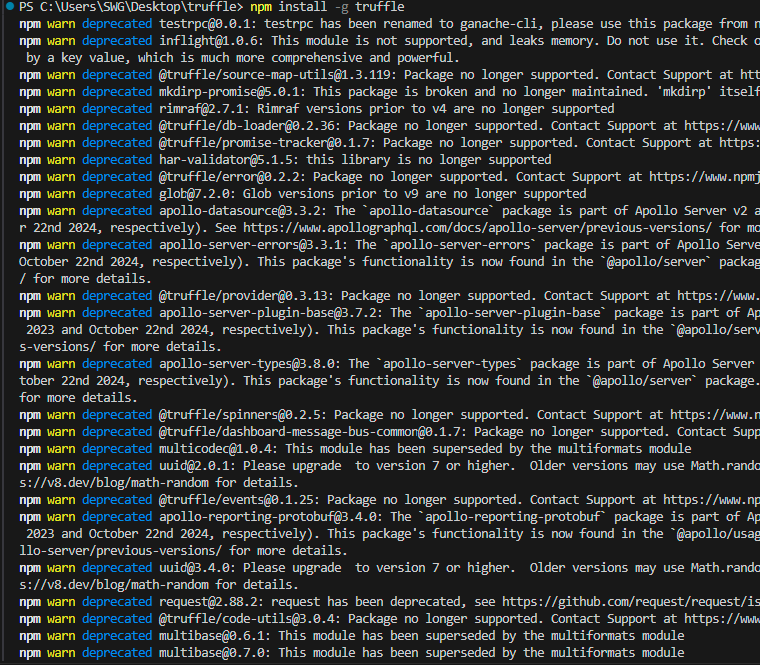
***Install Truffle: Use Node Package Manager (npm) to install Truffle by running***

***npm install -g truffle.***

Using VScode for Installing truffle

Run the command **npm install –g truffle**

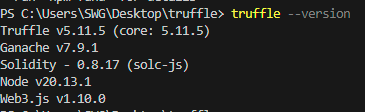
As it is already installed it shows



But when I check the Truffle version by running command

**Truffle –version**

Its shows the installed truffle version

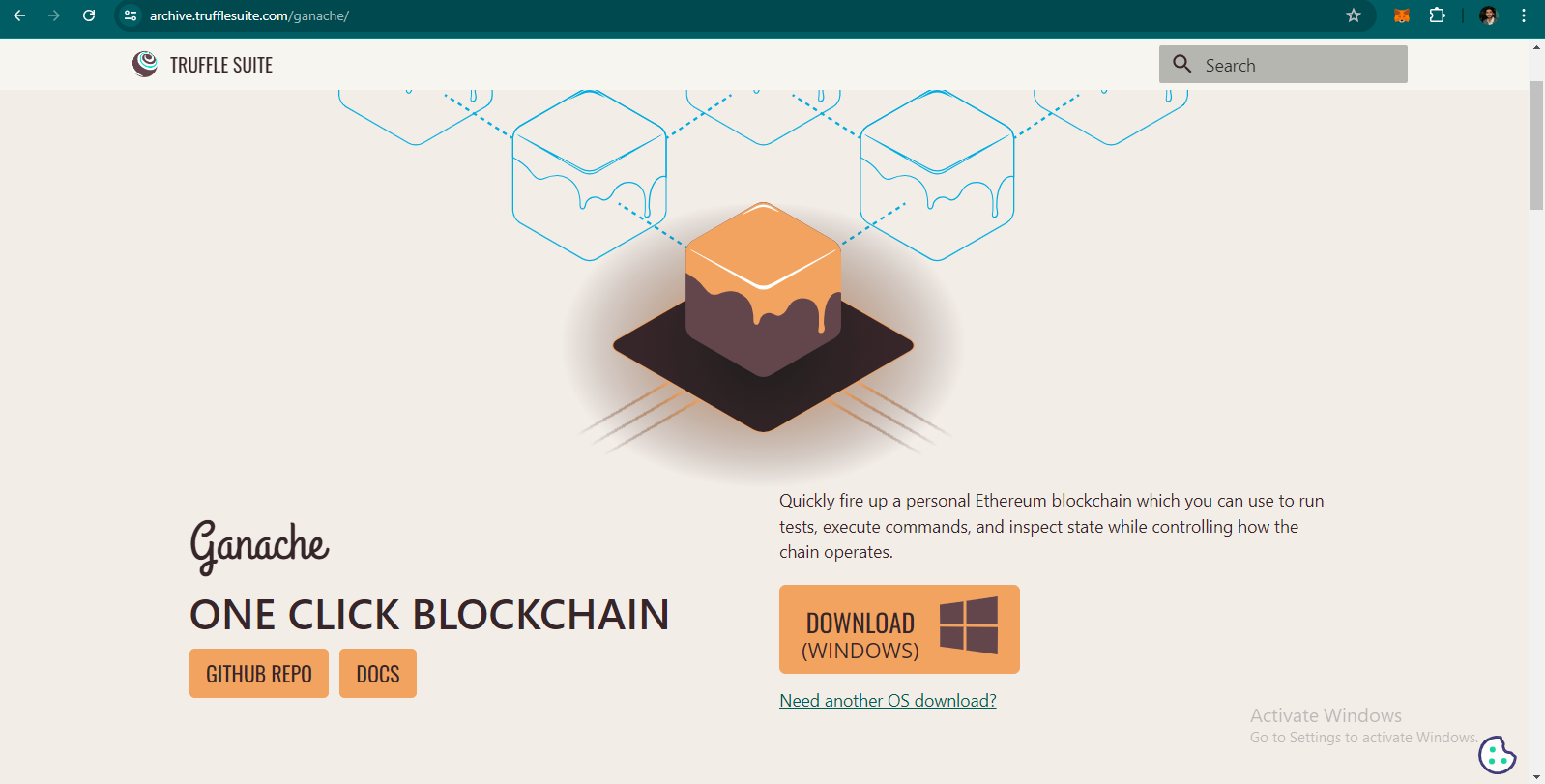


***Install Ganache: Download and install Ganache from trufflesuite.com/ganache.***

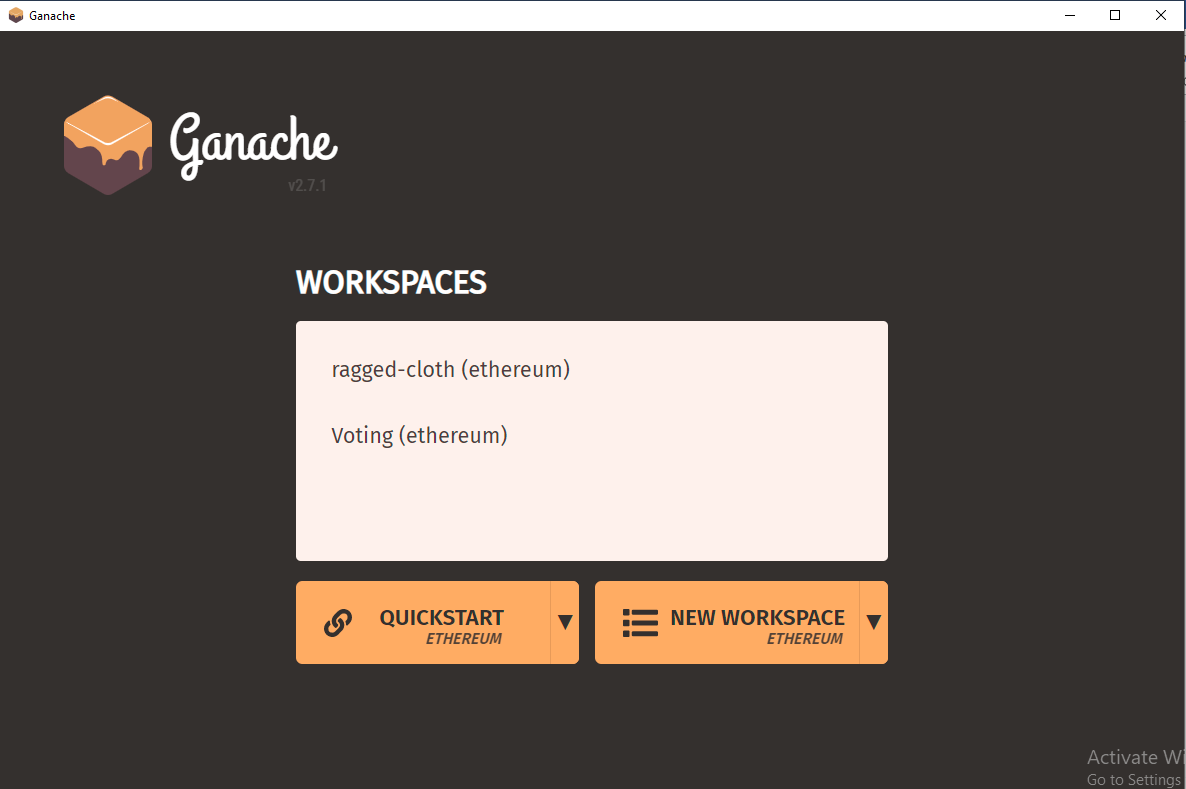
***Ganache is a personal blockchain for Ethereum development.***

From the website ***Trufflesuite.com/ganache***

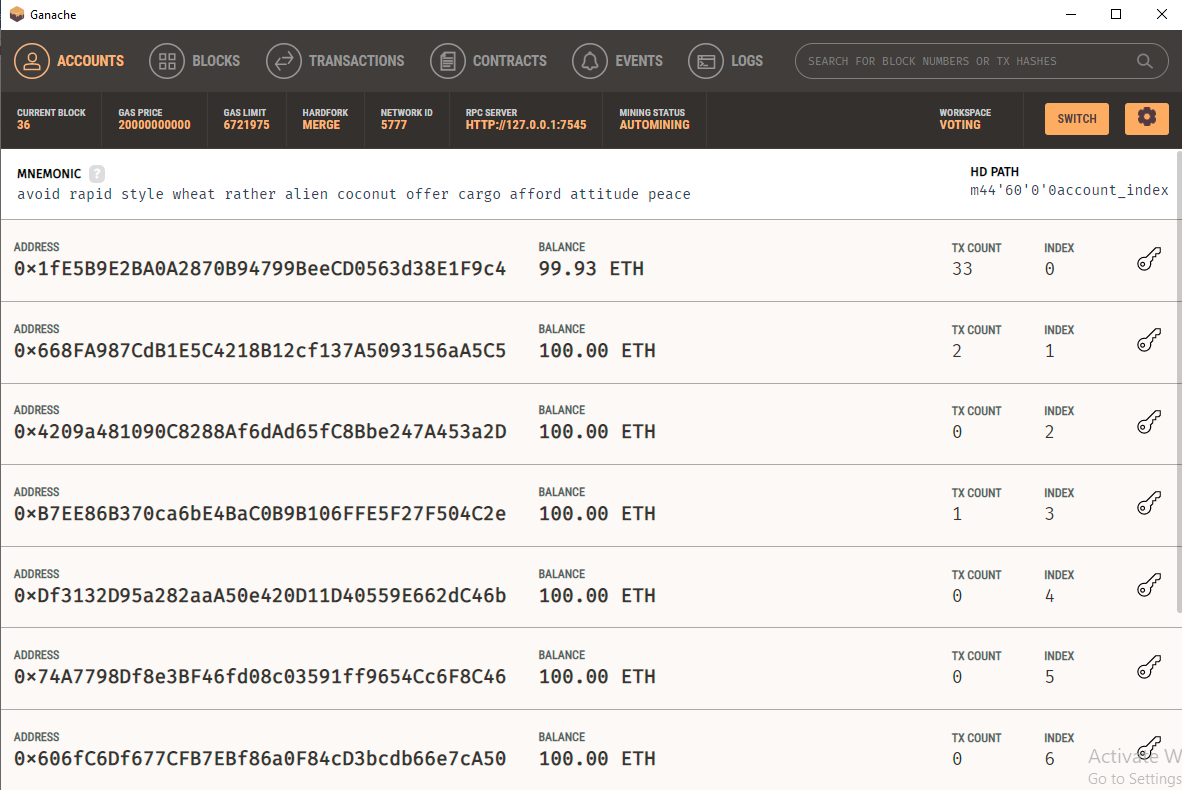
Downloaded ganache



Installed it



Used it for our voting contract as a local blockchain.



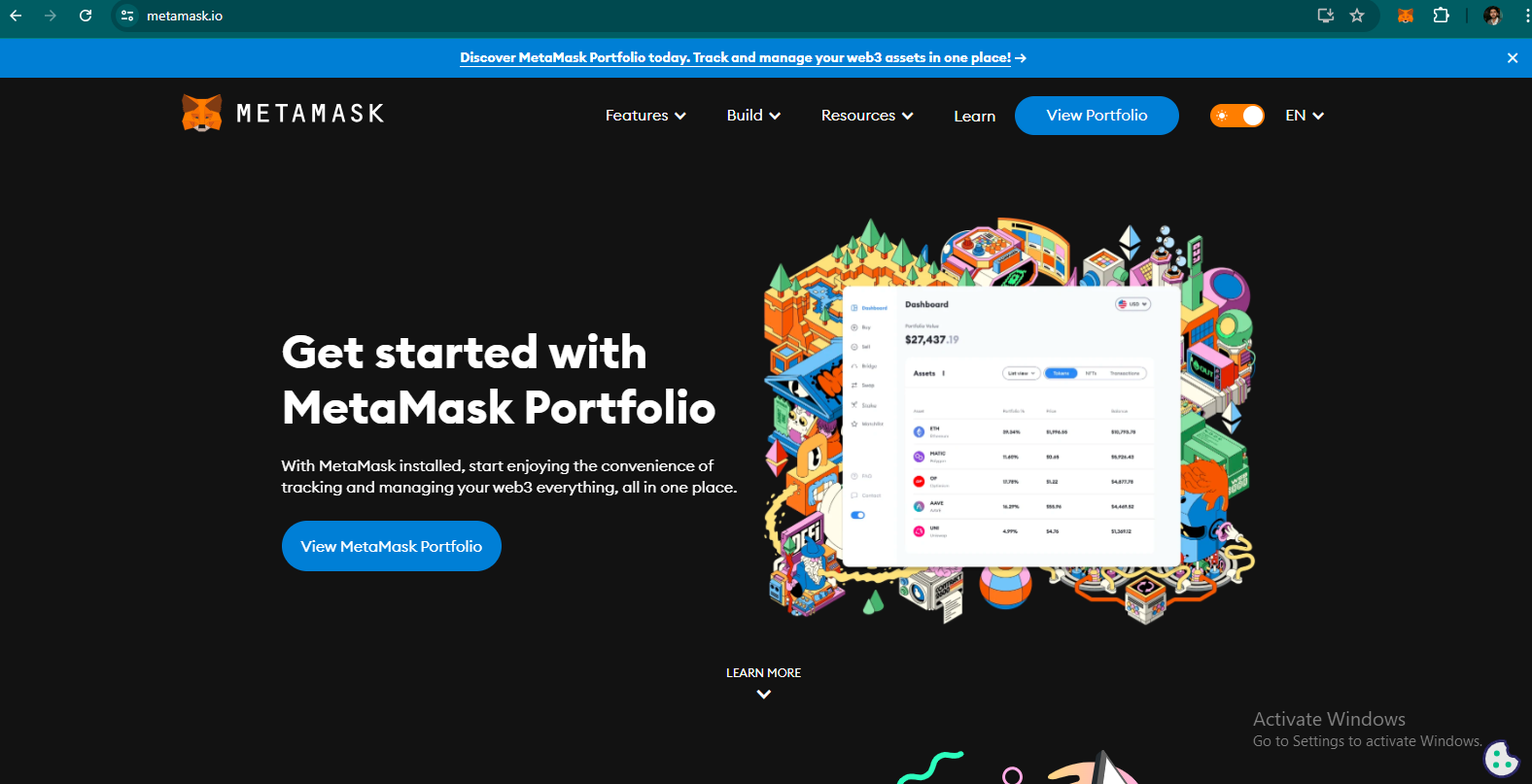
Transactions can be seen on first account due to deploying the contract many times and on other accounts as well for interacting with contract.

***Install Metamask: Install the Metamask browser extension from metamask.io.***

***Metamask is a cryptocurrency wallet used to interact with the Ethereum***

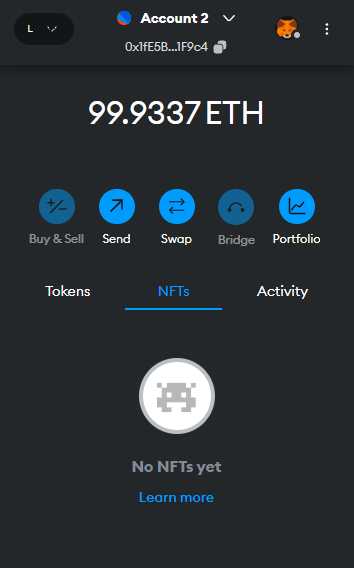
***blockchain.***

From the website

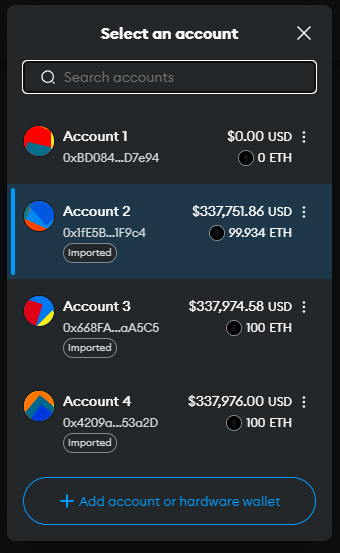


We added the Metamask extension to our browser.

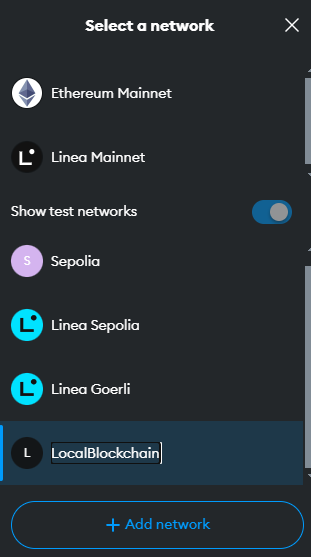
It helps us to keep account of the transactions and amount flowing around.



Different accounts have been added into metamask



Also connected to Ganache that is named as LocalBlockchain



***Initialize a Truffle Project:***

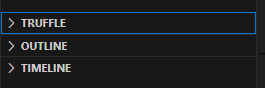
***Create a Project Directory: Create a new directory for your project and navigate***

***into it using the terminal:***

***mkdir VotingApp***

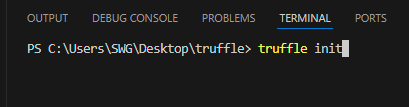
***cd VotingApp***

Have created a folder named as Truffle as directory and navigated to VScode.

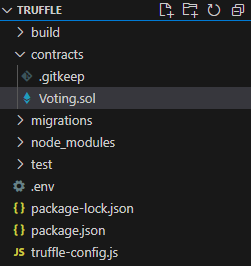


***Initialize Truffle: Initialize a new Truffle project by running:***

***truffle init***



It get us the Truffle project structure



***Add a Solidity File: Inside the contracts directory, create a new file named***

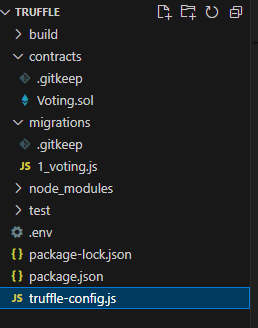
***Voting.sol.***

In the above snapshot File is added named as **voting.sol**

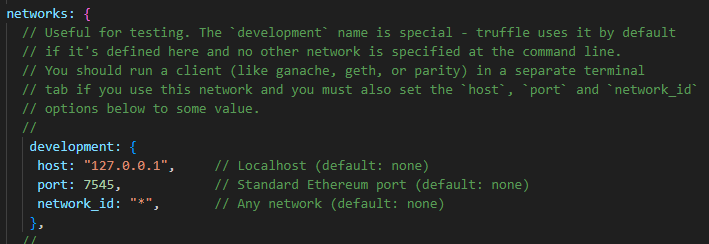
***Connecting ganache with Smart contract***

In truffle folder

Need to adjust something in config



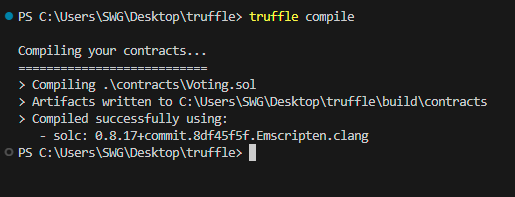
And that is the port number of networks by uncommenting it. We added port=7545 because it is the port number of Ganache

***Compile the Smart Contract:***

To compile the smart contract we need to run Command

**Truffle compile** in VScode terminal

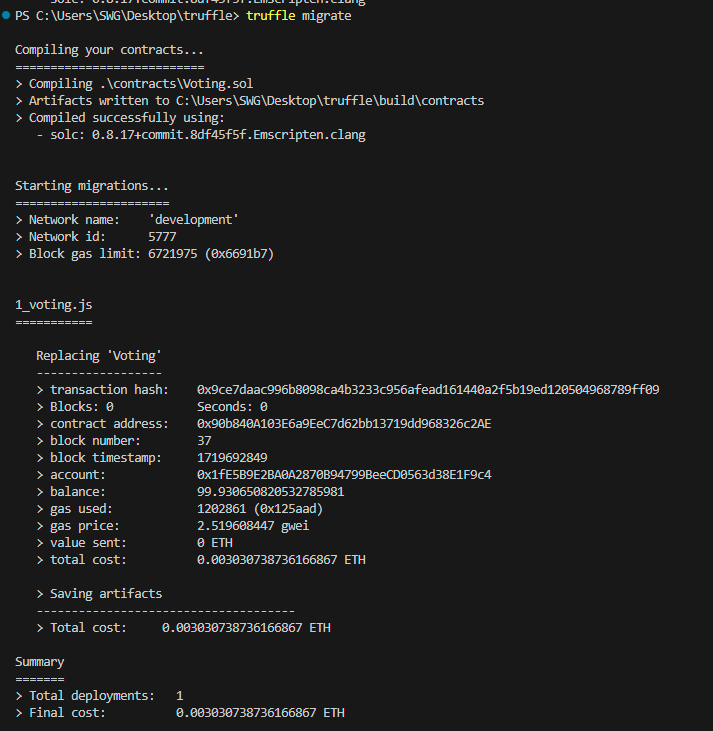


Project have been compiled

***Deploy the Smart Contract:***

To deploy the smart contract we run the command

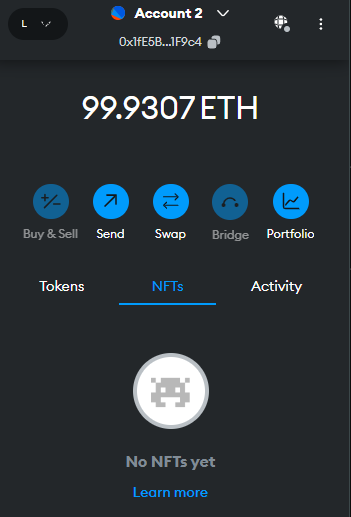
**Truffle migrate**



Contract have been deployed on the Ganache local blockchain.



This block shows the deployment of the contract. It is not the genesis block because I did it many times in order to learn.



Account have also been credited for deployment. Compare with the above amount.

***Interact with the Smart Contract:***

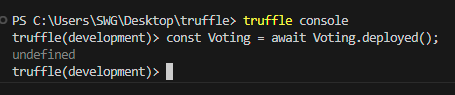
To interact with the smart contract we need to run JavaScript commands in the terminal.

***Retrieve Contract Instance: Get an instance of the deployed contract to interact***

***with it.***

**// Open truffle console and get deployed contract instance**

const Voting = await Voting.deployed();



***Add Candidates: Use the addCandidate function to add candidates to the***

***election.***

We need to define the contract owner

**// Get contract owner**

const owner = await Voting.owner();

console.log("Owner:", owner);

**// Add candidates**

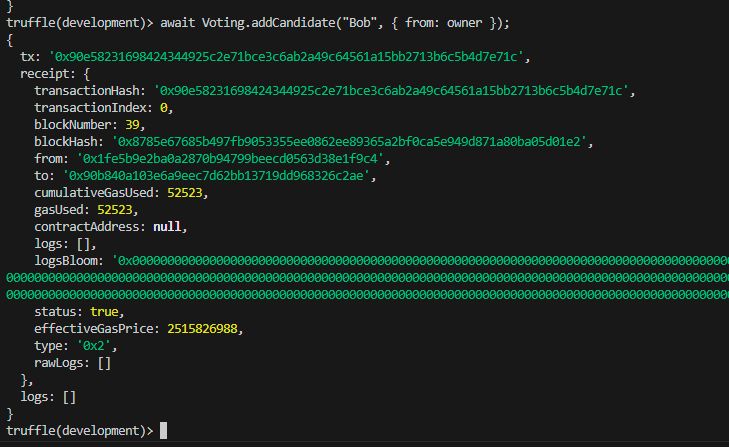
await Voting.addCandidate("Alice", { from: owner });

await Voting.addCandidate("Bob", { from: owner });

console.log("Added candidates: Alice, Bob");



In the above new candidate Alice have been added.



And Bob is added as a candidate here.

***To register a voter***

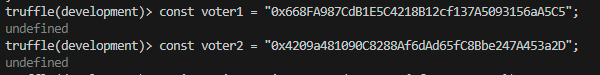
**// Register voters**

const voter1 = "0xYourVoter1Address";

const voter2 = "0xYourVoter2Address";

await Voting.registerVoter(voter1, { from: owner });

await Voting.registerVoter(voter2, { from: owner });



Two voters have been registered



These two addresses have been used to register as voters.

After it, got the instances of these voters by command

await Voting.registerVoter(voter1, { from: owner });

await Voting.registerVoter(voter2, { from: owner });

to check the voter’s addresses

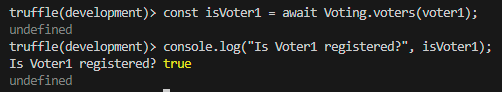


***Vote for Candidates: Use the vote function to cast votes for candidates.***

**// Check if addresses are registered voters**

const isVoter1 = await Voting.voters(voter1);

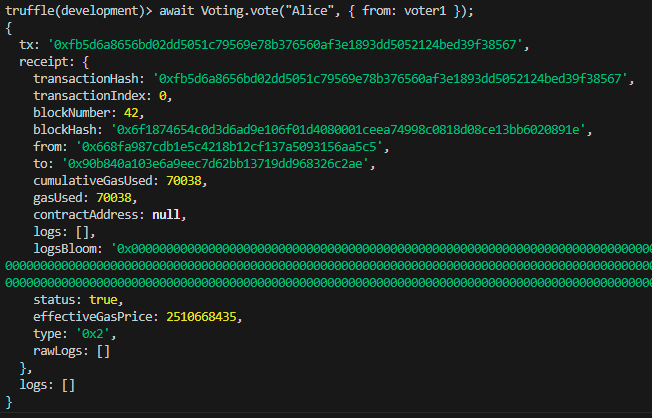
console.log("Is Voter1 registered?", isVoter1);



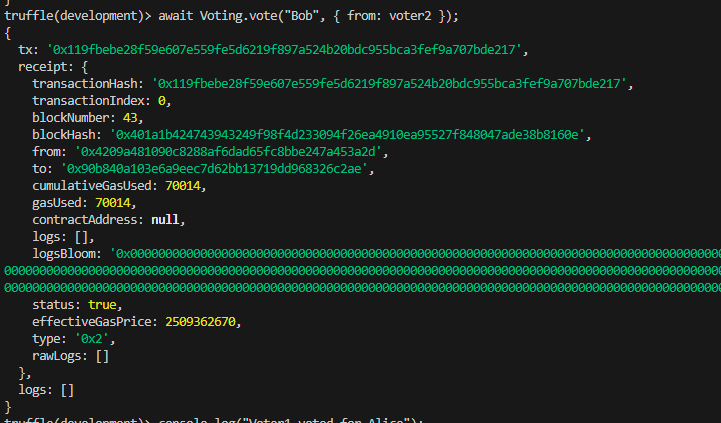
It means that the voter is registered.

**Now voting for candidate**

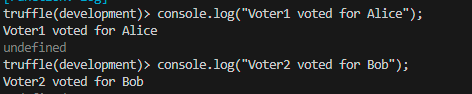
await Voting.vote("Alice", { from: voter1 });



await Voting.vote("Bob", { from: voter2 });

****

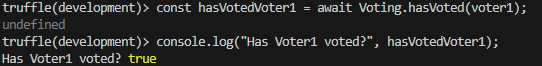
Check who voted for whom



***// Check if voters have voted***

const hasVotedVoter1 = await Voting.hasVoted(voter1);

console.log("Has Voter1 voted?", hasVotedVoter1);



Shows that the voter one has voted and it implies that this address voter cannot vote again.

***View Results: Use the getCandidate function to view the current vote count for***

***each candidate.***

**// Get results**

const results = await Voting.result();

results.forEach(result => {console.log(result.name, result.voteCount.toString());

});



Both got 1 vote each.

**// Get details of a specific candidate**

const aliceDetails = await Voting.getCandidate("Alice");

console.log("Alice details:", aliceDetails.name, aliceDetails.voteCount.toString());

