

Blockchain Assignment

Team Members:

Surendar S : 2020MT93162

Sunny Khanuja : 2020MT93110

E-Voting Distributed application (DAPP) Using Blockchain

Objectives of the Application

1

Make the Election process easy and secure.

2

Avoid the waiting for my turn to vote (avoid queue-based waiting system)

3

Make the vote counting process easy.

4

Avoid DOUBLE VOTING (casting invalid votes)

Scope of this document



Steps to get started: UI application (React application) and running smart contracts.



UI walkthrough and explanation
Note: Explanation Video link shared in this doc



Smart contracts explanation
Note: Explanation Video link shared in this doc



Links to access the source-code and Explanation video



Now, Lets get started.....

Getting started: Deploy smart contracts

- **To deploy the Smart Contract**

- Install Ganache and create a workspace.
- Install Truffle npm package globally by running `**npm install -g truffle**`
- Run `**truffle migrate --reset**` from the command line to deploy the smart contract to the blockchain.
- Download Metamask Chrome extension for the browser to help interaction between the application and the blockchain.

NOTE:

- make sure to run ganache an appropriate port to connect it with metamask
- Connect an account with local RPC network by using the private key of an account from ganache (this will grant 100ETH by default)

Getting started: Run react development server

- **To run the UI application locally**
 - Run the command `***npm install –no-optional***` (To download the node modules to your local working directory)
 - In order to start the UI application run the following command, ***'npm start'***
 - This will start the application in the localhost server.
 - NOTE: if the port is busy, then react will ask for confirmation to run in different port. The type in ‘y’ (Yes) to run the application.
 - This is start the application.

UI walkthrough – Explanation for different workflows

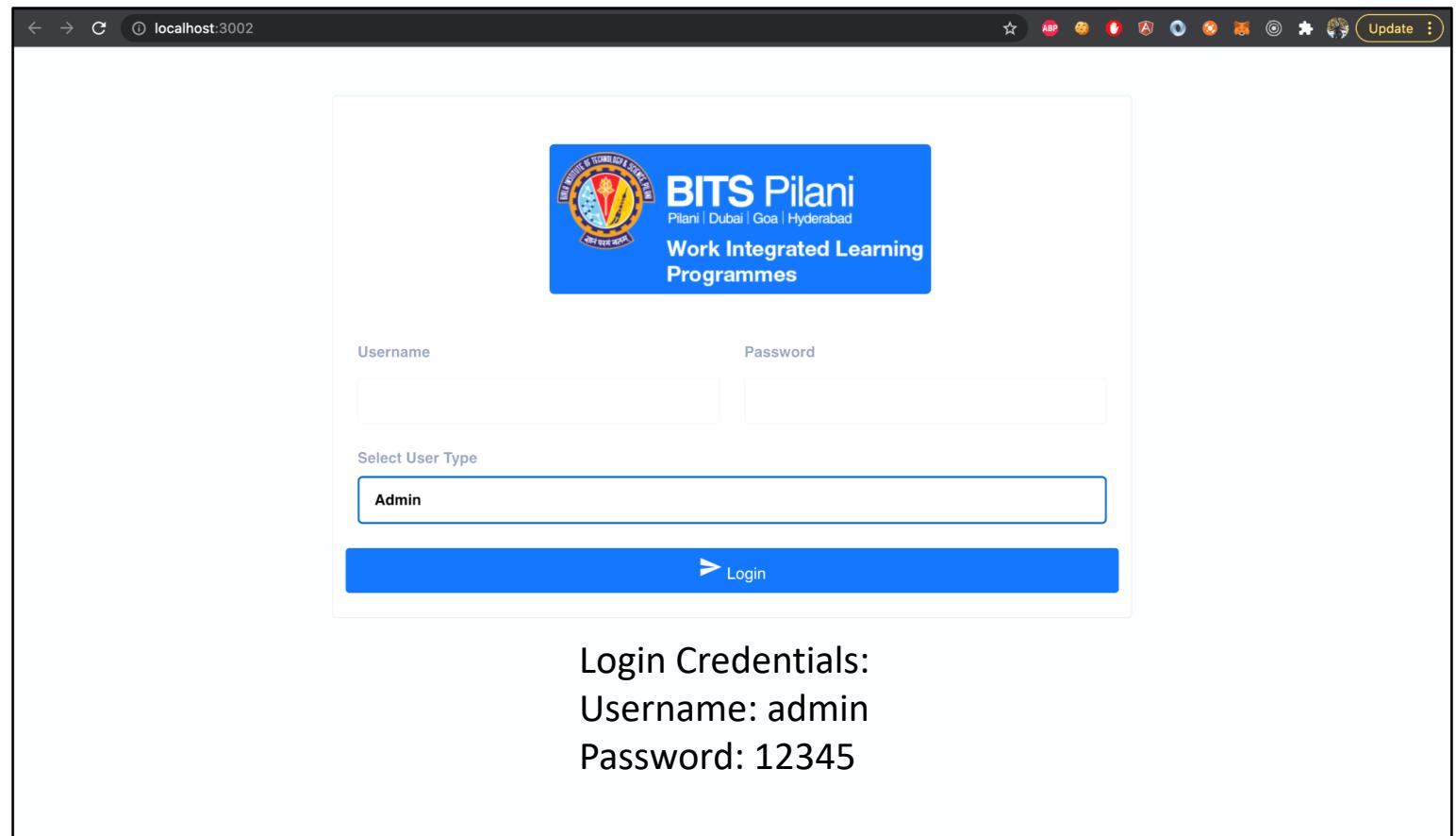
- - 1. Login as Admin
 - managing election, voters and candidates
 - 2. Login as Voter
 - for casting vote for a desired candidate

Admin workflow



Login as Admin

- This is how the landing page has been designed.
- There are two types of User, Admin and Voter.
- Admin has following privileges:
 - to create an Election (if not created before)
 - add candidates to election
 - Add voters
- **But they can't cast a vote**



The screenshot shows a web browser window with the URL `localhost:3002` in the address bar. The page title is "BITS Pilani Work Integrated Learning Programmes". The header includes the BITS Pilani logo and the text "BITS Pilani" followed by "Pilani | Dubai | Goa | Hyderabad". Below the header, there are two input fields: "Username" and "Password". Underneath these fields is a dropdown menu labeled "Select User Type" with the option "Admin" selected. At the bottom of the form is a large blue "Login" button with a white arrow icon. To the right of the form, there is a text box containing the login credentials: "Login Credentials: Username: admin Password: 12345".

Create Election (For first time)

- After a successful admin login, this is how the admin dashboard looks like.
- The admin need to create the election for the first time.
- With the help of “**web3.js**” we connect to the local Ethereum network (ganache) to store the election details.

BITS Pilani
Pilani | Dubai | Goa | Hyderabad
Work Integrated Learning
Programmes

Manage Election

No Election Exists: Create a new one

Election Name

BESCOM: Bengaluru

Create Election

Election BESCOM-Karnataka Added with ID ELECTION_5

Manage Election

- Once the Election is created, then the admin can start add candidates and voters.
- As of now, there is a one-to-one mapping for the Election organizer (Admin) and the election smart contract.
- This can be extended further to provide access to an organizer to handle multiple elections.

localhost:3000/elections

BITS Pilani
Pilani | Dubai | Goa | Hyderabad
Work Integrated Learning Programmes

Manage Election

Manage Election: Candidates & Voters

BESCOM: Bengaluru

Manage Candidates Manage Voters

Managing Candidates

- Once the Manage Candidates button is clicked, one can add the candidates under an election contract.
- As of now, we store the
 - Candidate ID
 - Candidate name
 - Description
 - Vote count

The screenshot shows a web browser window with the URL `localhost:3000/candidates/ELECTION_NO_12`. The page is titled "Manage Candidates for BESCOM: Bengaluru". It features a table with columns for Candidate ID, Candidate Name, Description, and Votes. There is also a header row with "Candidate Name" and "Candidate Description". A blue button labeled "+ Add" is visible at the top right of the table area. The table contains four rows of data:

Manage Candidates for BESCOM: Bengaluru			
	Candidate Name	Candidate Description	
ID	Candidate Name	Description	
1	Sunny	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	123
1	Rahil	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	25
1	Ramesh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	30
1235	Suresh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	45

Adding Candidate to Election

- When the add button is clicked, then the candidate will be added to the blockchain data.
- The newly added candidate can be seen in the below list of candidate.

localhost:3000/candidates/ELECTION_NO_12

BITs Pilani
Pilani Dubai | Goa | Hyderabad
Work Integrated Learning Programmes

Manage Election

Manage Candidates for BESCOM: Bengaluru

Candidate ID	Name	Description	Votes
1	Sunny	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	123
2	Rahil	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	25
3	Ramesh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	30
4	Suresh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	45
5	Surendar	Some Description	0

Candidate Surendar added to election BESCOM: Bengaluru

Manage Voters

- When manage voter button is clicked, one can manage the list of voters by adding them.
- As of now, we store the
 - Voter ID
 - Voter name
 - Description
 - If voter Casted a vote or not

The screenshot shows a web browser window with the URL `localhost:3000/voters/ELECTION_NO_10`. The page title is "Manage Election". The main content is titled "Manage Voters for BESCOM: Bengaluru". It displays a table with four rows of voter information. The columns are "Voter ID", "Name", "Description", and "Casted Vote". The "Casted Vote" column contains icons: an "X" for the first three voters and a checkmark for the fourth. A blue "Add" button is located at the top right of the table area.

Voter ID	Name	Description	Casted Vote
1	Sunny	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	X
2	Rahil	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	✓
3	Ramesh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	X
4	Suresh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	✓

Manage Voters: Add new Voter

- When the add button is clicked, then the voter information will be added to the blockchain data under the current Election.
- The newly added candidate can be seen in the below list of candidate.

The screenshot shows a web browser window with the URL `localhost:3000/voters/ELECTION_NO_10`. The page is titled "Manage Election". On the left, there's a sidebar with the "Manage Election" icon. The main content area is titled "Manage Voters for BESCOM: Bengaluru". It displays a table with the following data:

Voter ID	Name	Description	Casted Vote
1	Sunny	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	X
2	Rahil	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	✓
3	Ramesh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	X
4	Suresh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	✓
5	Surendar	A new voter from India	X

A purple banner at the bottom right of the table area says "Voter Surendar added to election: BESCOM-Karnataka".

Voter workflow

Login as Voter

- A voter can login to the application by selection the user type as **Voter**
- The valid Username will be the same name the provided while creating Voter (by admin)
- As of now the default password is 12345.

localhost:3000



BITS Pilani
Pilani | Dubai | Goa | Hyderabad
Work Integrated Learning
Programmes

Username

Password

Select User Type

> Login

Voter Login: Vote for Candidate

- When the Voter logs in, then the list of candidates will be displayed.
- One can cast the vote for any set of the listed candidates.
- **'ONLY ONCE'.**
- The **Double voting** is restricted.

localhost:3000/candidates/ELECTION_NO_12

BITS Pilani
Pilani | Dubai | Goa | Hyderabad
Work Integrated Learning
Programmes

Sign Out

Vote for candidate in BESCOM: Bengaluru

Candidate ID	Name	Description	Actions
1	Sunny	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
2	Rahil	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
3	Ramesh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
4	Suresh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
5	Surendar	Some Description	<button>Vote</button>

Candidate Login: Voted for Candidate

- When a User tries to login again, then they will not be able to cast the vote.
- It is almost not possible to change the vote, because of the advantage of blockchain, **Data is Immutable**.
- **The vote button will be disable.**
- The total number of votes casted can be seen in admin page.

The screenshot shows a web browser window with the URL `localhost:3000/candidates/ELECTION_NO_12`. At the top, there is a header for 'BITS Pilani' with the subtext 'Pluri-Disciplinary Institute' and 'Work Integrated Learning Programmes'. On the right side of the header is a 'Sign Out' button. Below the header, the main content area has a title 'Vote for candidate in BESCOM: Bengaluru'. A table lists five candidates:

Candidate ID	Name	Description	Actions
1	Sunny	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
2	Rahil	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
3	Ramesh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
4	Suresh	Sed vitae nulla a est commodo vehicula. Morbi finibus malesuada maximus. Quisque non neque egestas erat scelerisque interdum eget id odio. Pellentesque vitae hendrerit orci, eu congue quam. Donec semper velit ut velit elementum aliquet. Aliquam tincidunt sem in pharetra rutrum. Donec placerat t	<button>Vote</button>
5	Surendar	Some Description	<button>Vote</button>

At the bottom center of the page, there is a message: 'Your vote is already recorded.'

Explanation for Smart contracts



Smart contracts

- In this DAPP we have used two smart contracts (*./src/contracts*)
 1. Election.sol
 2. Migration.sol
- **Election.sol**
 - This contract holds the model to store election details like
 - election name
 - List of candidates and their details (*id, name, desc, voteCount, candidatesCount*)
 - List of voters and their details (*id, name, desc, hasVoted, votersCount*)
 - Also, the methods to cast vote, add voter, add candidate
- **Migration.sol**
 - This is generated (automatically) when we do truffle migrate for the first time.

Revealing truffle-config.js

- The deployment is done in localhost, port 7545
- This is the same port where ganache RPC is running
- The built contracts are generated once contracts are migrated.

```
module.exports = {
...
networks: {
  development: {
    host: "127.0.0.1",
    port: 7545,
    network_id: "*" // Match any network id
  },
},
contracts_directory: './src/contracts/',
contracts_build_directory: './src/build/',
compilers: {
  solc: {
    optimizer: {
      enabled: true,
      runs: 200
    }
  }
}
}
```

How to run truffle contracts in console

- Considering the pre-requisites (truffle), one can use terminal to deploy, and make changes to blockchain data using smart contracts.
- In order to do so, please follow these steps:
 1. Make sure you are in the project directory (../Blockchain-EVoting-UI)
(Project Link available in the last slide)
 2. To deploy the smart contract, use the following command
`truffle migrate --reset`
 3. To start the truffle console, type in
`truffle console`
 4. Now, we are good to start with command-line instructions to deal with smart contracts

Let us consider a real time scenario

- For an election to be conducted,
 - We need 'n' number of candidates enrolled for the election.
 - We need 'n' number of valid voters to cast their vote for the enrolled candidates.
 - **Important:** A voter can cast their vote only once, **NO DUPLICATE VOTING**
 - The one with the Maximum number of votes will win the election.
- Let us consider a similar scenario.
- We will be creating an Election-block, to which we will add three voters and three candidates.
- Let's see who wins the election.

Ganache UI: Create workspace

The screenshot shows the Ganache UI interface. At the top, there are tabs for ACCOUNTS, BLOCKS, TRANSACTIONS, CONTRACTS, EVENTS, and LOGS. Below these are various configuration settings: CURRENT BLOCK (0), GAS PRICE (2000000000), GAS LIMIT (6721975), HARDFORK (MUIRGLACIER), NETWORK ID (5777), RPC SERVER (HTTP://127.0.0.1:7545), MINING STATUS (AUTOMINING), WORKSPACE (QUICKSTART), and buttons for SAVE, SWITCH, and a gear icon. A search bar at the top right allows searching for block numbers or tx hashes.

MNEMONIC: rescue fortune acid tourist pitch ask apart addict demand build brand royal

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

ADDRESS	BALANCE	TX COUNT	INDEX	Action
0x73518Ca1c4521B2d246dEBDD3bdEAb5EF5949881	100.00 ETH	0	0	🔗
0xC3bA84FfEbD3148AaaCD98CF0ec6742bA3F599BA	100.00 ETH	0	1	🔗
0xb7948A868f7f5D728cEcC659f34c5B5b2f9A124a	100.00 ETH	0	2	🔗
0x1530F003Bbb669b02B8DCd807f5855Ca87eF6E71	100.00 ETH	0	3	🔗
0x65D7E08f2f716f5697CE7F54b275AB5d1a6618B7	100.00 ETH	0	4	🔗
0x9D319a8860f64fa163d6Cd209b045629653A29B3	100.00 ETH	0	5	🔗
0xEc538cc884649CA1E0F116ada15c221423Cec354	100.00 ETH	0	6	🔗

- Make sure that ganache UI is up and running.
- We can also make use of ganache-cli. Use following command to install in local
`npm install ganache-cli`
- Once ganache is up we can proceed with truffle commands

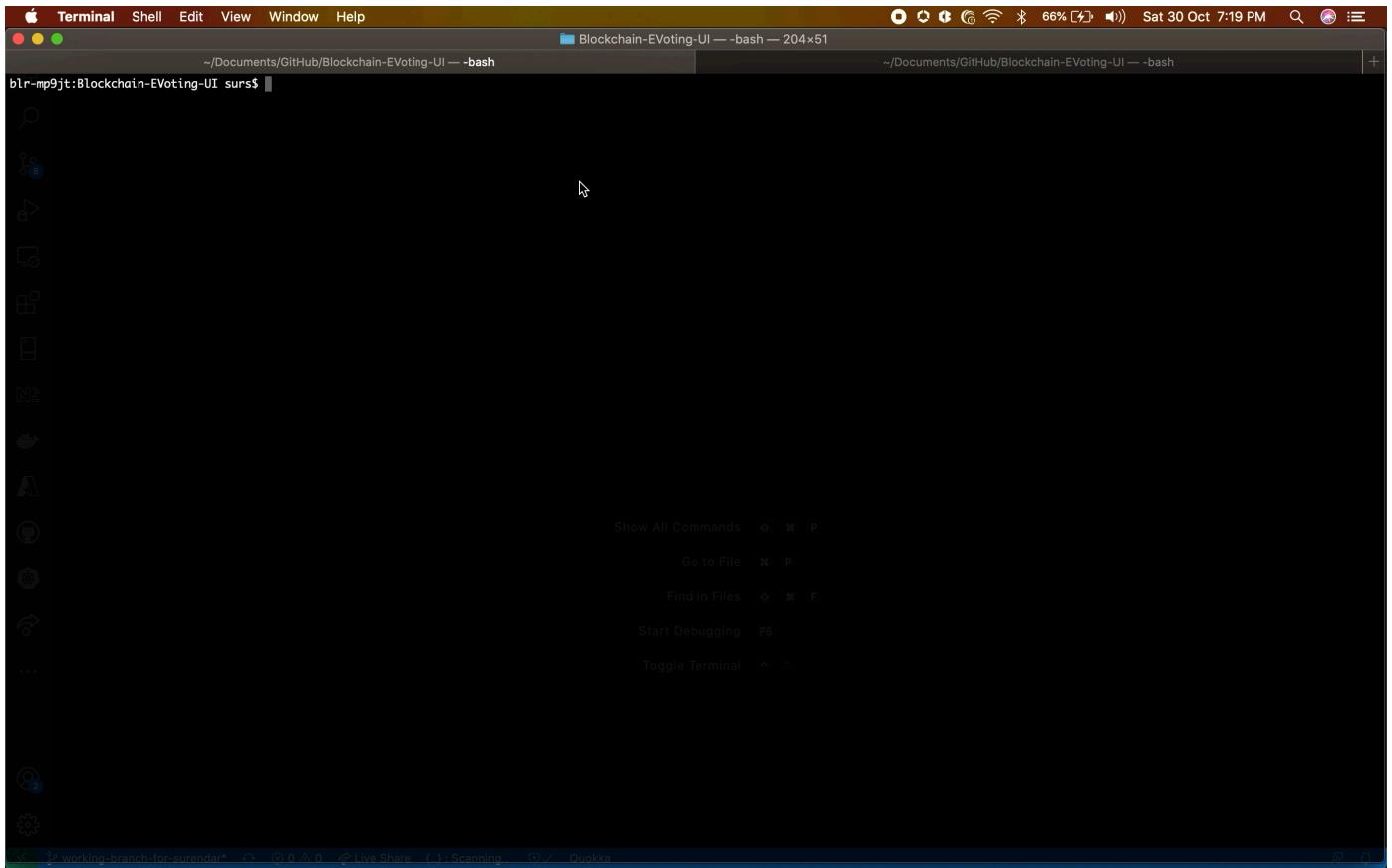
Truffle console commands

Step	Command	Description
1	truffle migrate --reset	Deploys the smart contract to Ethereum network
2	truffle console	To start the truffle console
3	Election.deployed().then((instance) => {app = instance})	To get the Election object (the deployed contract is a promise)
4	app.addCandidate('CANDIDATE_1', 'Name is Sunny')	To add a new candidate NOTE: Replace name and description appropriately
5	app.candidatesCount()	To get the count of candidates added
6	app.addVoter('VOTER_1', 'Name is Surendar')	To add a new voter NOTE: Replace name and description appropriately
7	app.votersCount()	To get the count of voters added
8	app.vote(1,1)	To cast a vote for a candidate by a voter NOTE: the arguments are func vote(voter_id, candidate_id) as stored in blockchain

Migration of smart contracts

Summary

- Total deployments: 2
- Final cost: 0.02391962 ETH



Add Candidate

- Start the Truffle console
 - Get the Election object by resolving the promise
 - Now add 3 candidates.
 - For each addition of candidates, a transaction will be made to the Election block address.
 - The data will be stored in the block.
 - This method is integrated in the UI using web3.js to add candidates for an election by admin.
 - The candidate voteCount is set to '0' initially.

Get the count of Candidates



The screenshot shows a macOS terminal window with two tabs. The active tab is titled 'Blockchain-EVoting-UI — node /usr/local/bin/truffle console — 82x49'. It displays the following Truffle console output:

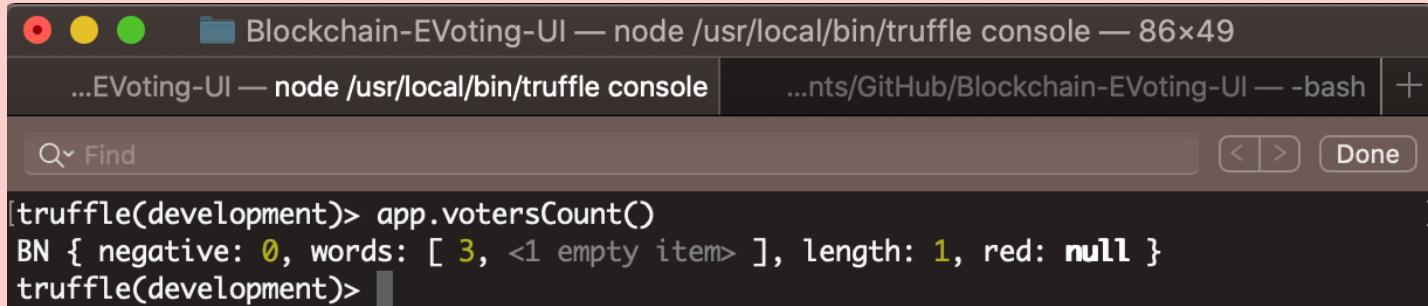
```
[truffle(development)> app.candidatesCount()
BN { negative: 0, words: [ 3, <1 empty item> ], length: 1, red: null }
truffle(development)>
=> {app = instance});
Name is Surendar'
'Name is Sunny')
```

- Once the candidates are added we can fetch the total number of candidates added
- This method is helpful in case of UI integration.
- The candidates are stored in (candidate id => Candidate struct)
`mapping uint => Candidate`
- This method has been used in UI to fetch and populate the Total candidates in Voters' page and admin page (manage candidates)
- In this case the Candidate count is 3, as per the considered scenario.

Add Voter

- Now add 3 Voters.
 - For each addition of voters, a transaction will be made to the Election block address.
 - The data will be stored in the block.
 - This method is integrated in the UI using web3.js to add Voters for an election by admin.
 - The voter's "hasVoted" field is set to false initially
 - This will be toggled when a vote is casted.

Get the count of Voters



```
[truffle(development)> app.votersCount()
BN { negative: 0, words: [ 3, <1 empty item> ], length: 1, red: null }
truffle(development)> ]
```

- Once the voters are added we can fetch the total number of candidates added
- This method is helpful in case of UI integration. (Admin page)
- The candidates are stored in mapping (voter id => Voter struct)`mapping uint => Voter`
- This method has been used in UI to fetch and populate the Total candidates in admin page (manage voters)
- Here the voters count is 3, as per the considered scenario.

Now lets cast
the vote...

- We can make use of the function `vote(voter_id, cand_id)` to cast the vote.
 - A voter can cast vote only once, as the hasVoted value will be set true after voting.
 - In this scenario, the voter with id 1 (VOTER1) is casting the vote for the candidate with id 1 (Surendar).
 - Similarly, voter 2 to candidate 1 and voter 3 to candidate 2, has casted their corresponding votes.

What happens when a voter tries to cast a vote again...?

- When a voter who already casted a vote tries to vote again, they will not be able to vote.
 - We have added a validation check

```
require(!voterList[_voterId].hasVoted, 'The Voter has already voted.');
```
 - Hence, ***'The Voter has already voted.'*** message will be thrown as an error and the transaction will be failed.

```
`require(!voterList[_voterId].hasVoted, 'The Voter has already voted.');
```

- Hence, `***The Voter has already voted.***` message will be thrown as an error and the transaction will be failed.

```
[truffle(development)> app.electionCandidates(1)
Result {
  '0': BN {
    negative: 0,
    words: [ 1, <1 empty item> ],
    length: 1,
    red: null
  },
  '1': 'CANDIDATE1',
  '2': BN {
    negative: 0,
    words: [ 2, <1 empty item> ],
    length: 1,
    red: null
  },
  '3': 'Name is Surendar',
  id: BN {
    negative: 0,
    words: [ 1, <1 empty item> ],
    length: 1,
    red: null
  },
  name: 'CANDIDATE1',
  voteCount: BN {
    negative: 0,
    words: [ 2, <1 empty item> ],
    length: 1,
    red: null
  },
  details: 'Name is Surendar'
}
truffle(development)>
```

The candidate 1 has
won the election... 

- The candidate 1 got 2 votes, candidate 2 got 1 vote and candidate 3 got no vote.
- The member var, **voteCount** has been used in UI to show the vote count for a candidate.
- And this is how the smart contracts work.



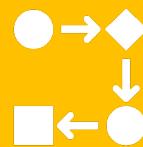
Objectives that we achieved



Made the vote counting easier.



Since we make use of blockchain to store the data, it is immutable hence it adds security to the process workflow.



Avoidance of waiting for my turn to cast my vote, since the process has been automated now.

Scope and Extensions

Scope/ Limitations

- The scope of this Application is restricted to one election per admin.
- An admin can organize only a single election.

Extensions

This can be extended in such a way that, the smart contracts can be split into three different entities

- Election (indexed with address)
- Candidates (referenced with election address)
- Voters (referenced with election address)

Contribution split

Surendar S (2020MT93162)

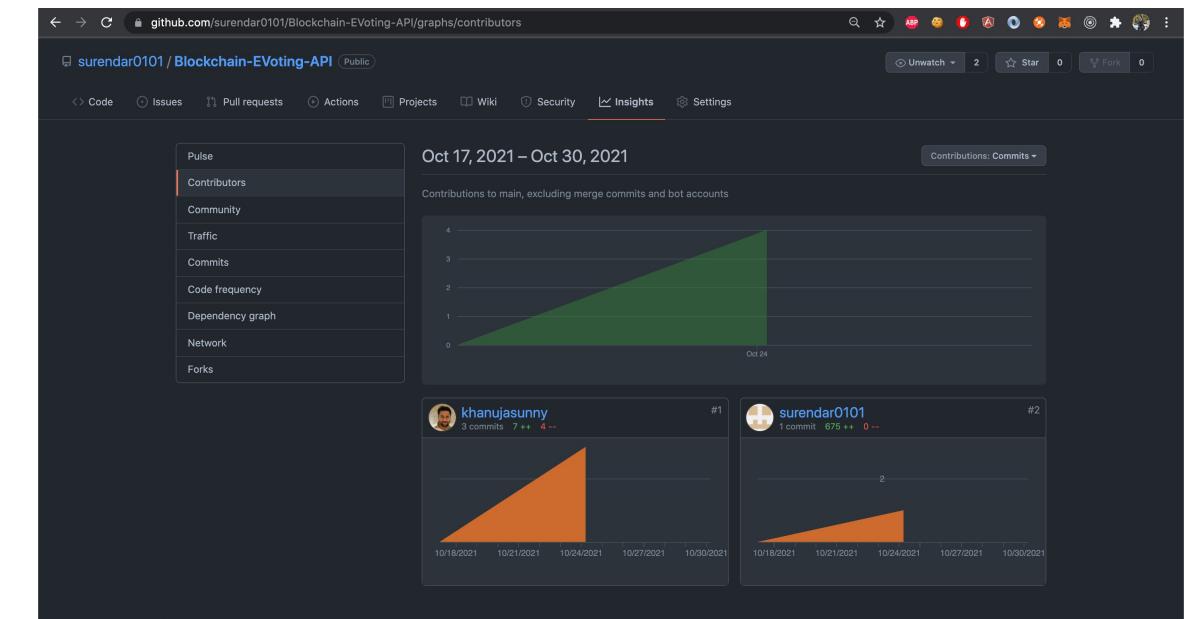
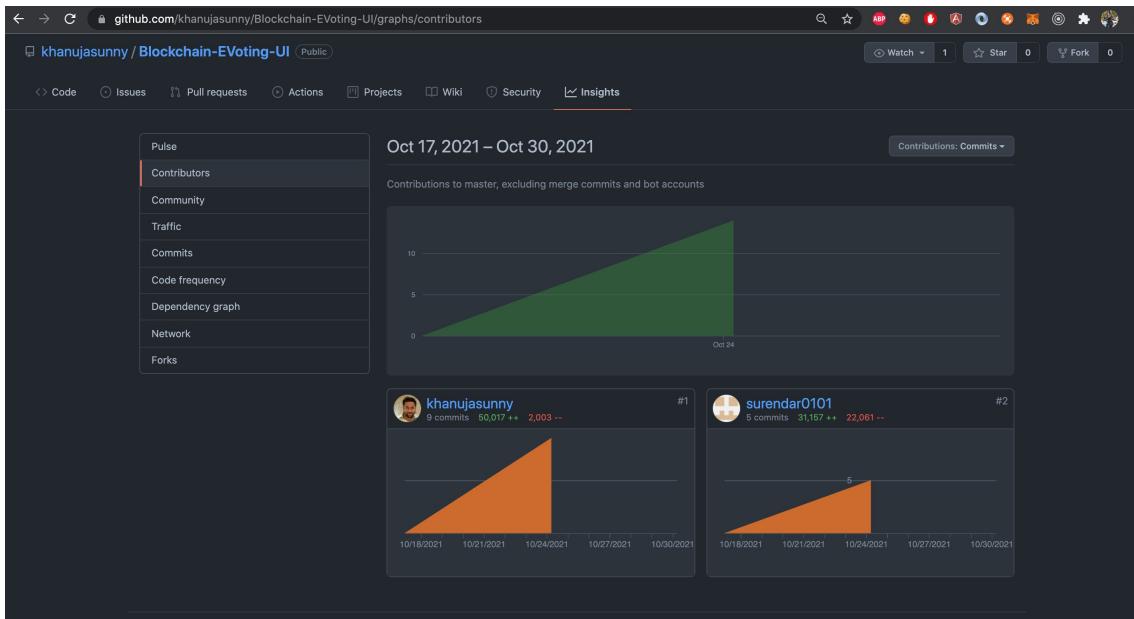
- API application For login
- Worked on smart contracts
- Worked on integrating UI application with web3.js
(Few modules: Like adding a new Candidate)

Sunny Khanuja (2020MT93110)

- Worked entirely setting up UI project
- Also, worked on Integrating the UI application

Overall, it was a collaborative work. We followed parallel coding.

Work contribution (UI and API)





Learnings

- Working on smart contracts was something which gave us a lot of learning.
- The assignments gave us a lot of learning when compared to the theoretical content.
- Since we got to work practically, this gave us more understanding.
- The online lectures to work with metamask, ganache, Ethereum solidity added more learning.
- Overall, it was something new that we learnt.

Links

Github:

1. UI Application:
<https://github.com/khanujasunny/Blockchain-EVoting-UI>
2. API Application:
<https://github.com/surendar0101/Blockchain-EVoting-API>

Explanation Video:

(contains detailed workflow and code explanation)

https://drive.google.com/drive/u/1/folders/1OjIGRGbXGS1qT2LIQEPOWEb1IZ8n_geh

Thank you
