DOCUMENTATION ON

ELECTRONIC VOTING SYSTEM

USING HYPERLEDGER BLOCKCHAIN

Name - Saurabh Kumar Rai
Emp Id- 24085

Table of Contents

About The Project

Electronic Voting System (EVS) is used for election management for a country where it manage Constituency ,Party ,Candidate ,Voter and conduct an election and declare result.

a-***Current Scenario***

The traditional way to conduct an election is Ballot Paper method ,but now it is conducting by EVM.

b-***Problem***

There are many challenges in voting through traditional way ,some of them are following-

Paper ballots

- Voter fraud
- Time-consuming
- Ballot damaging

EVMs

- Tampering
- Lack of transparency

- Accessibility

c-**Solution(Advantages of introducing Fabric)**

- **Security:**
  1-Encryption
  2-Digital signatures
  3-Access control

- **Transparency:** Hyperledger Fabric is a permissioned blockchain, which means that only authorized parties can participate in the network.

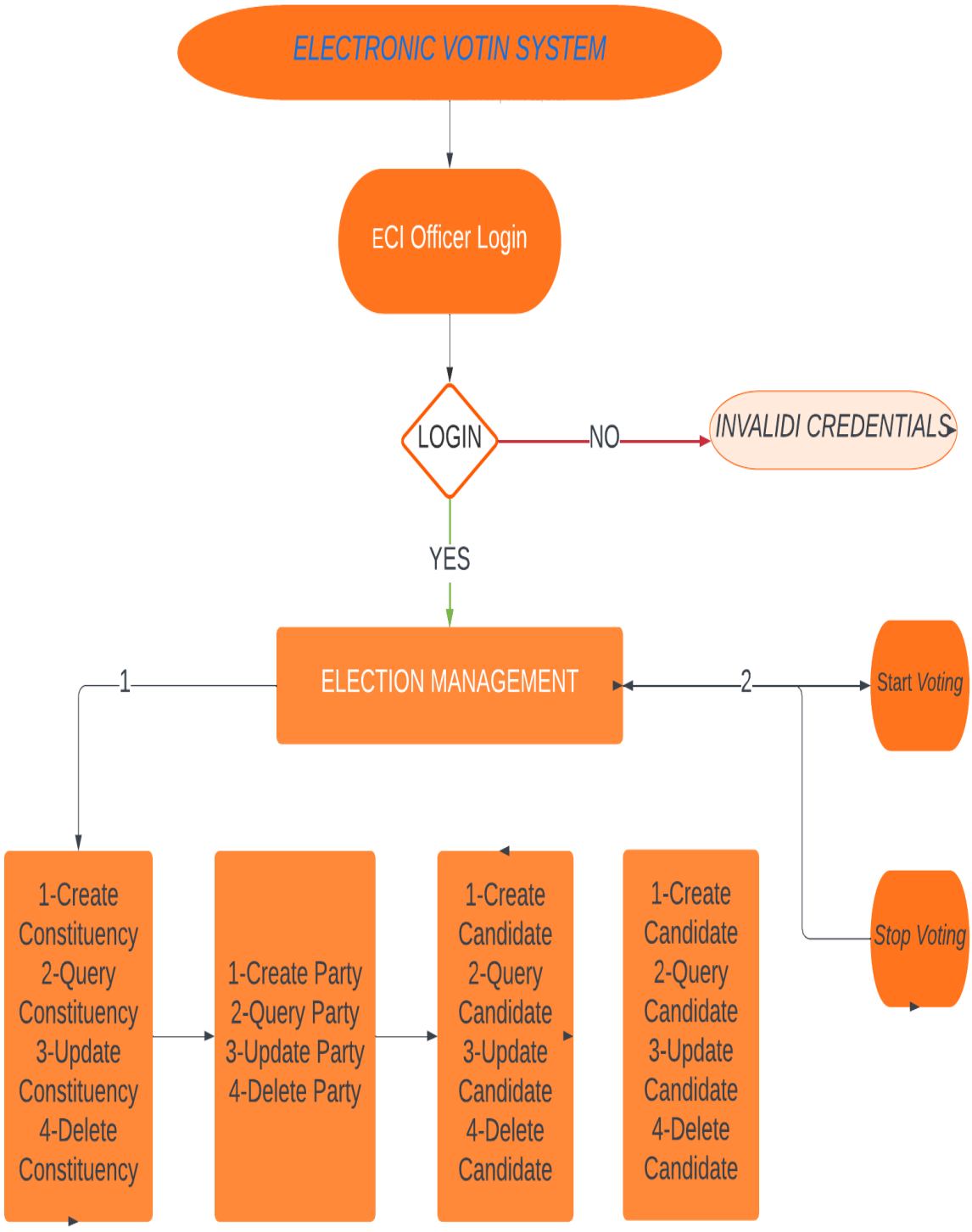
- **Scalability:** Hyperledger Fabric is a scalable platform that can be used to support large-scale elections.

- **Cost-effectiveness:** Hyperledger Fabric is a cost-effective platform that can be used to reduce the cost of elections.

Workflow

1- Workflow for an election commission officer:-

1. Create constituencies
2. Create parties
3. Register voters
4. Nominate candidates
5. Start voting
6. Stop voting
7. Declare the results

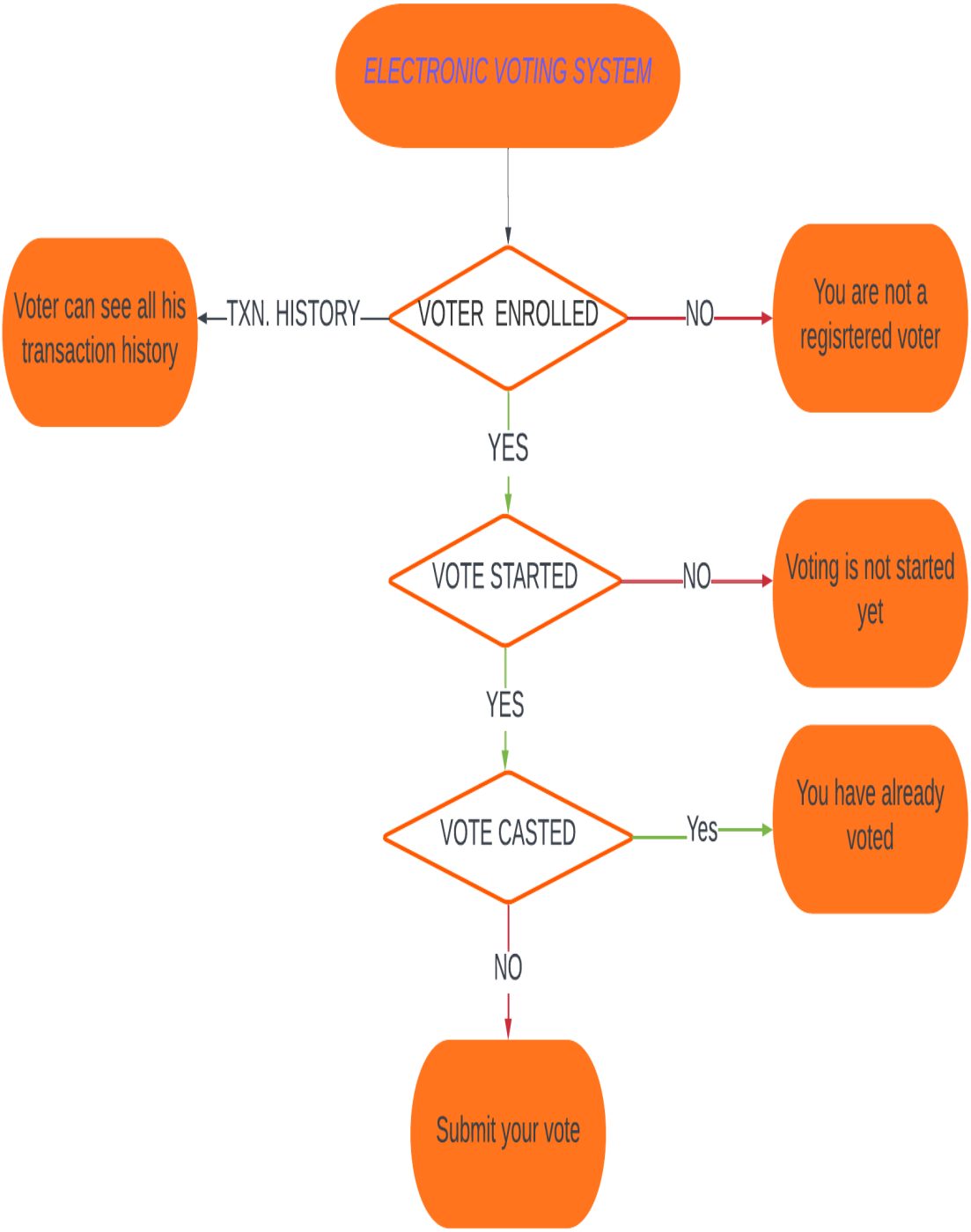

ELECTRONIC VOTIN SYSTEM
ECI Officer Login
LOGIN
NO
INVALIDI CREDENTIALS
YES
ELECTION MANAGEMENT
1
2
Start Voting
Stop Voting
1-Create Constituency
2-Query Constituency
3-Update Constituency
4-Delete Constituency
1-Create Party
2-Query Party
3-Update Party
4-Delete Party
1-Create Candidate
2-Query Candidate
3-Update Candidate
4-Delete Candidate
1-Create Candidate
2-Query Candidate
3-Update Candidate
4-Delete Candidate

2- Workflow for a Voter:-

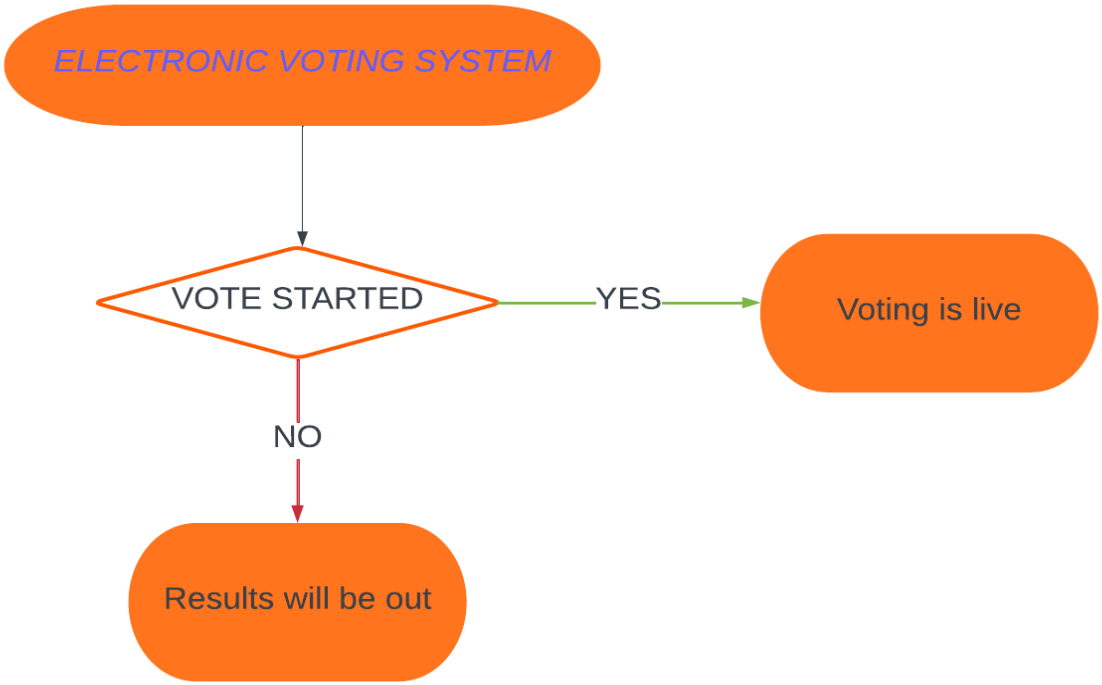
1. Enroll: The voter must first enroll to vote. It is a kind of verification process where voter Id, voter name ,dob verifies from voter list.

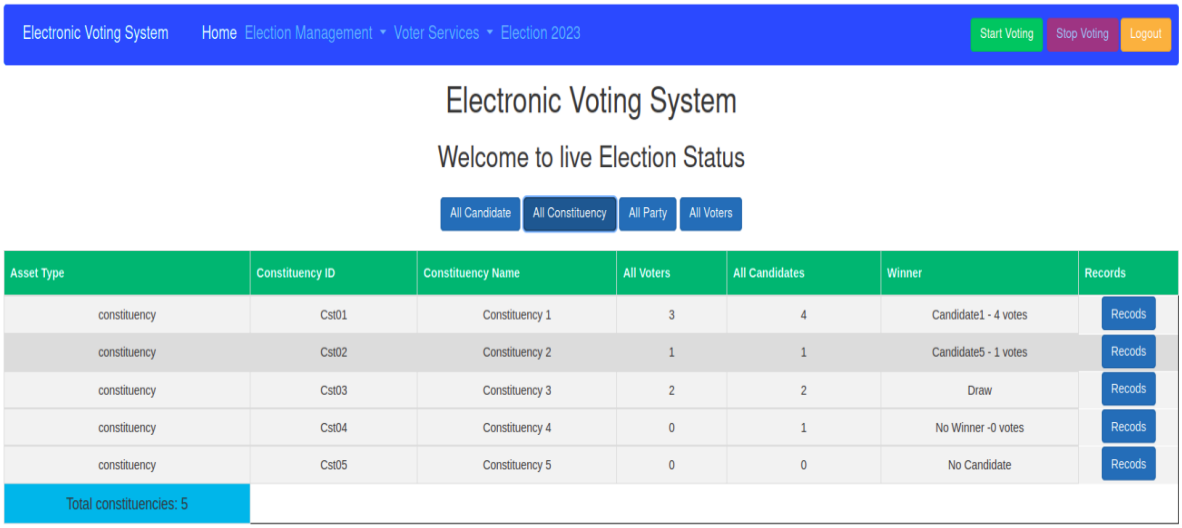
2. Cast vote: Once the voter is enrolled, they can cast their vote.

3. View transaction history: The voter can view their transaction history by logging into their voter account online. The transaction history will show following data
    a. Registered transaction history
    b. Enrolling transaction History
    c. Voting Tarnsaction History


ELECTRONIC VOTING SYSTEM
VOTER ENROLLED
TXN. HISTORY
Voter can see all his transaction history
NO
You are not a regisrtered voter
YES
VOTE STARTED
NO
Voting is not started yet
YES
VOTE CASTED
Yes
You have already voted
NO
Submit your vote

3- Results


ELECTRONIC VOTING SYSTEM
VOTE STARTED
YES
Voting is live
NO
Results will be out


Electronic Voting System
Home
Election Management
Voter Services
Election 2023
Start Voting
Stop Voting
Logout
Electronic Voting System
Welcome to live Election Status
All Candidate
All Constituency
All Party
All Voters
Asset Type
Constituency ID
Constituency Name
All Voters
All Candidates
Winner
Records
constituency
Cst01
Constituency 1
3
4
Candidate1 - 4 votes
Recods
constituency
Cst02
Constituency 2
1
1
Candidate5 - 1 votes
Recods
constituency
Cst03
Constituency 3
2
2
Draw
Recods
constituency
Cst04
Constituency 4
0
1
No Winner -0 votes
Recods
constituency
Cst05
Constituency 5
0
0
No Candidate
Recods
Total constituencies: 5

Running the Application

A-Prerequisites -

- Linux OS (Ex- Ubuntu)
- Good Internet connection

B-Installation -

A. Install git

sudo apt install git

B. Install vscode , download the .deb file for Ubuntu from https://code.visualstudio.com/download

sudo dpkg -i <file_name>

C. Create a folder as Blockchain-Project and clone the repo using vscode from https://gitlab.com/saurabhkumarr99/saurabhkumarrai_electronicvotingsystem

D.Download IBM Blockchain Extension from https://gitlab.com/CHF_KBA/kba_chf_ibmblockchain

platformextension_vscode/-/raw/main/ibm-blockchain-platform-2.0.8.vsix?inline=false

E. Add IBM extension in vscode

F. Install NPM packages

npm install

sudo npm install -g express-generator

G.Open folder Blockchain-Project terminal and execute

chmod +x installDependencies.sh

```
./installDependencies.sh
```

H.Reboot the system and execute

```
./installDependencies.sh bin
```

C- How to run the Application according to the workflow

1-Open the Electronic-Voting-System -> Network folder

2-Go to network folder terminal and execute
./startNetwork.sh

3-Open Chaincode->KBA-EVS folder in vscode

4-Go to IBM Blockchain Platform and add
a-All Wallets
b-Environment
c-Connect all Gateways
d-Package the project with .tar.gz file
e-Go to evsChannel add package,collections.json
f-Deploy smart contracts

5-Open Event folder in vscode and go to terminal and execute contractEventListener.js

6-Open the UI folder in vscode and go to terminal and execute the cmd -

npm start

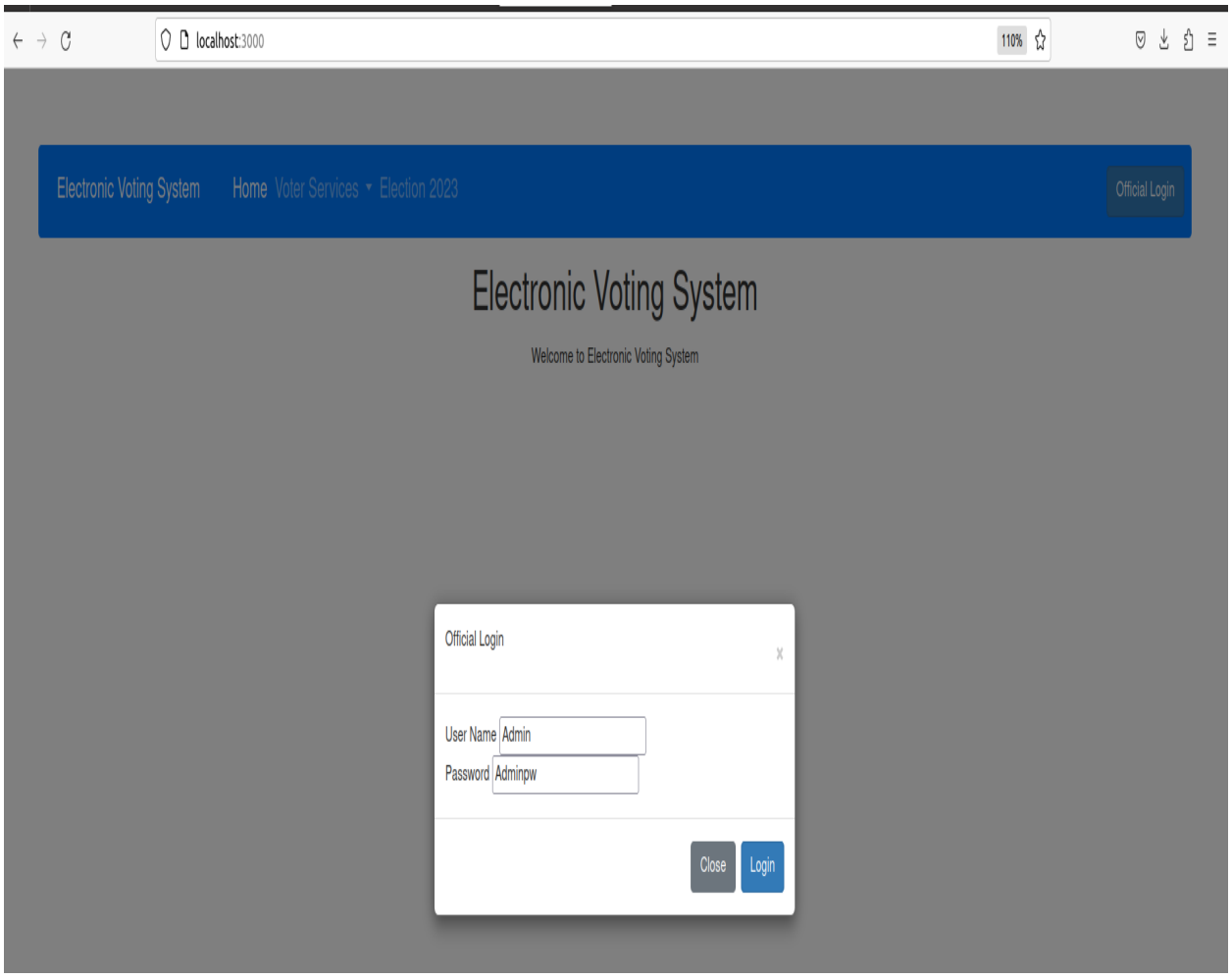
Click on http://localhost:3000/

ECI Officer workflow

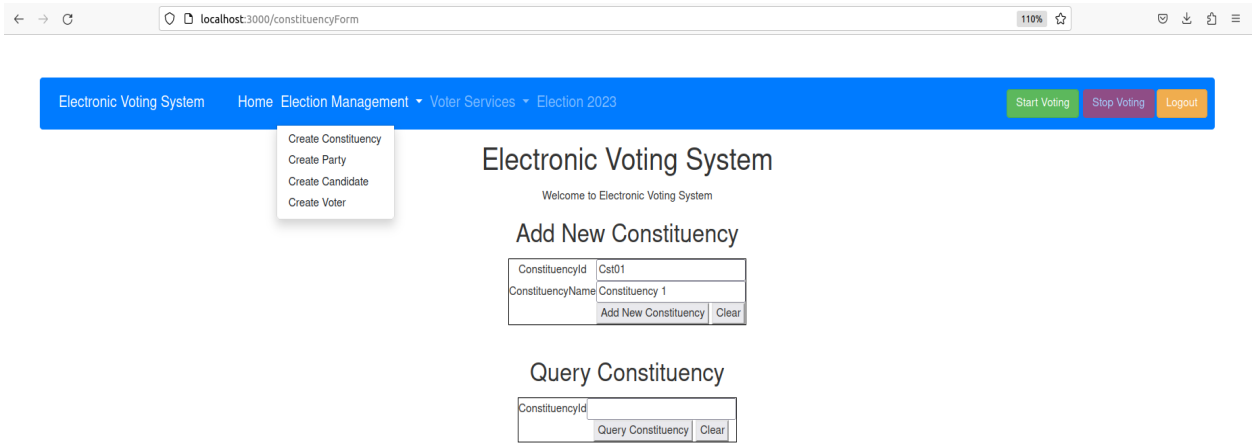
7.a-On the home page click to Official Login

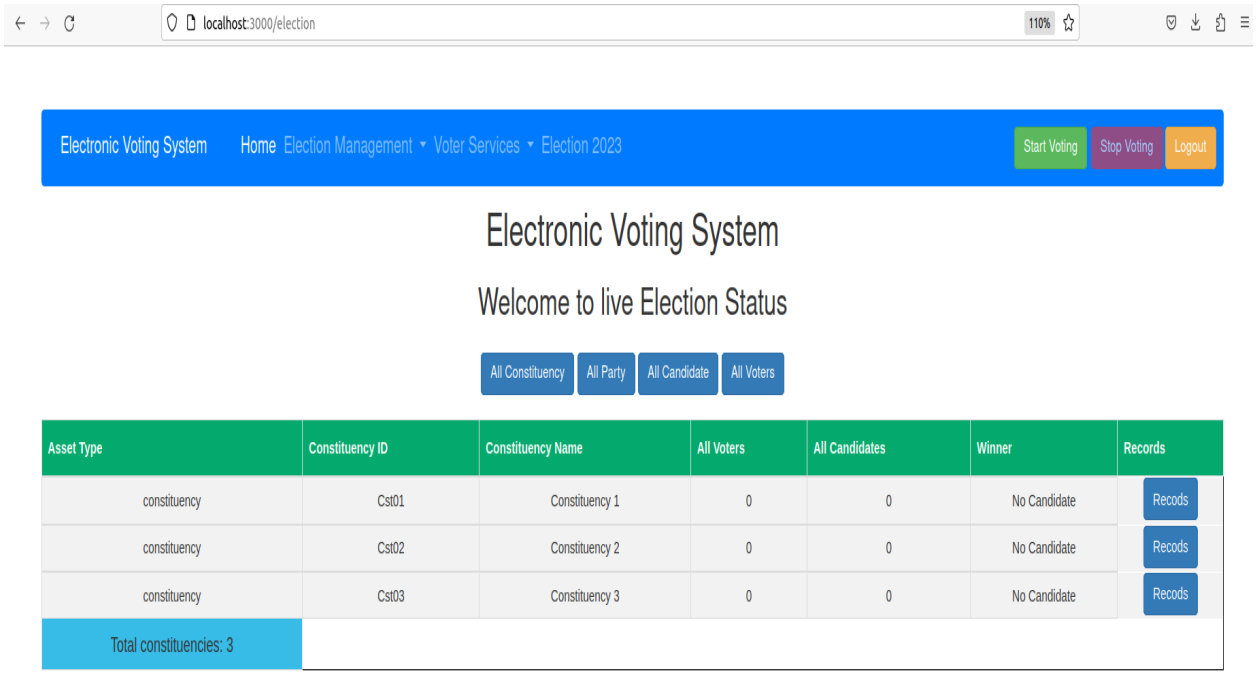
7.b-Username-Admin

Password-Adminpw

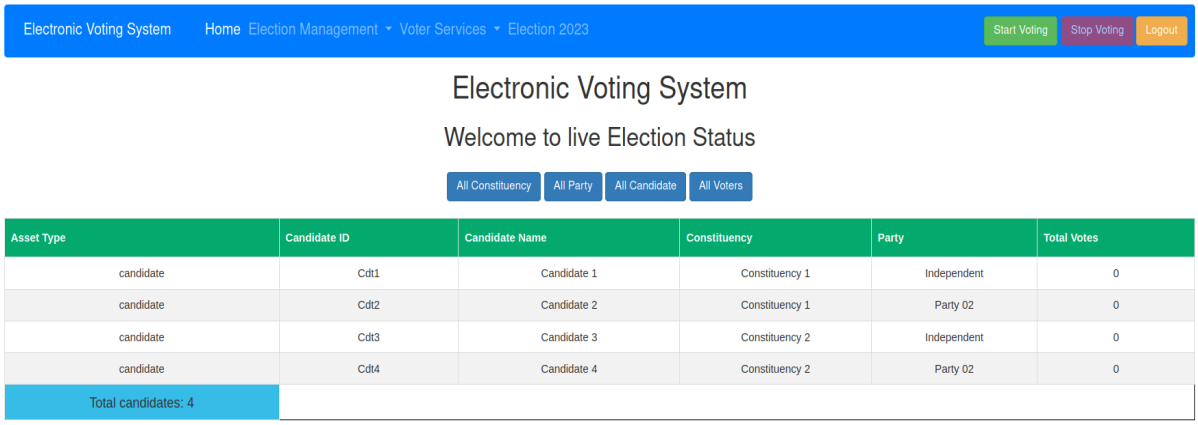


8-Click Election Management and add Create constituency



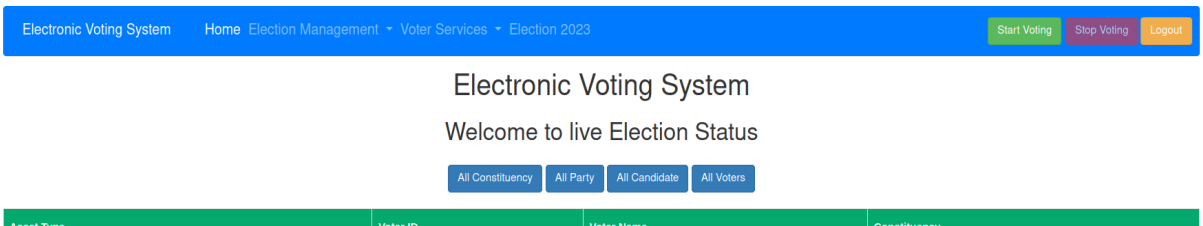


9-Similarly add Party, Candidate, Voters



Asset Type	Candidate ID	Candidate Name	Constituency	Party	Total Votes
candidate	Cdt1	Candidate 1	Constituency 1	Independent	0
candidate	Cdt2	Candidate 2	Constituency 1	Party 02	0
candidate	Cdt3	Candidate 3	Constituency 2	Independent	0
candidate	Cdt4	Candidate 4	Constituency 2	Party 02	0
Total candidates: 4					

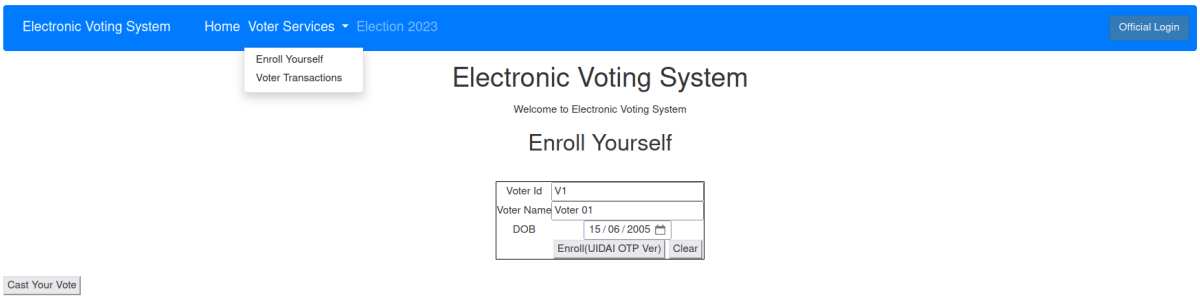
Asset Type	Party ID	Party Name
party	P1	Independent
party	P2	Party 02
party	P3	Party 03
Total parties: 3		



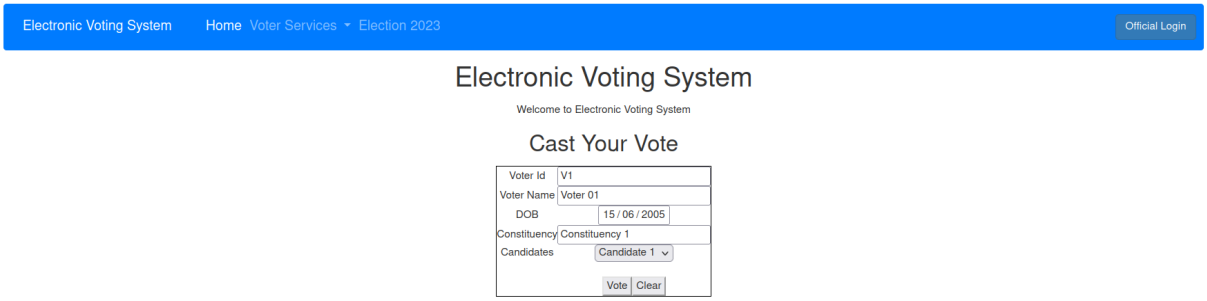
Asset Type	Voter ID	Voter Name	Constituency
voter	V1	Voter 01	Constituency 1
voter	V2	Voter 02	Constituency 1
voter	V3	Voter 03	Constituency 2
voter	V4	Voter 04	Constituency 1
voter	V5	Voter 05	Constituency 2
Total voters: 5			

Voter workflow

10-Go to Voter Services and enroll a voter

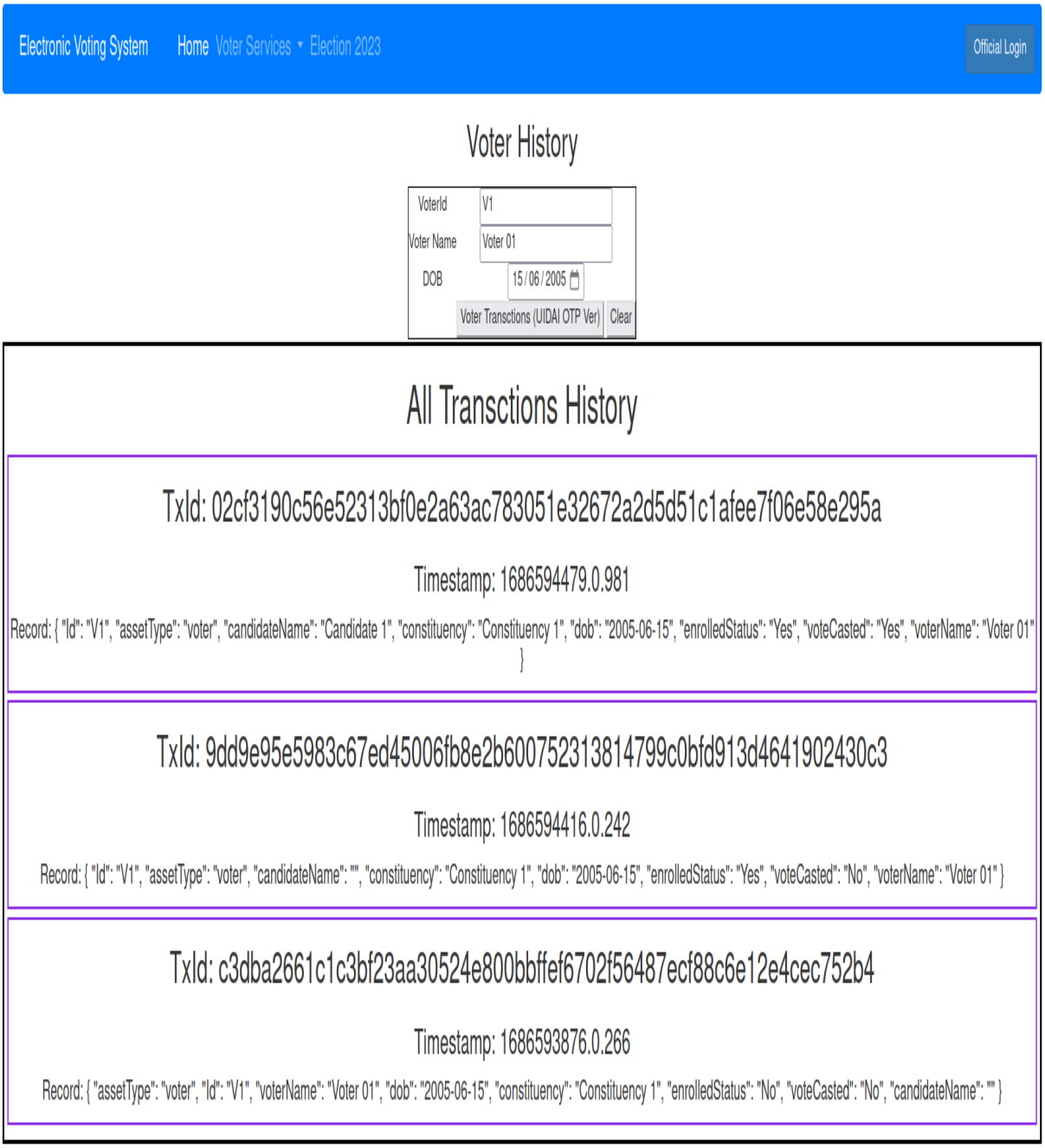


11-After successful enrolling , cast your vote



12- Select a candidate and vote

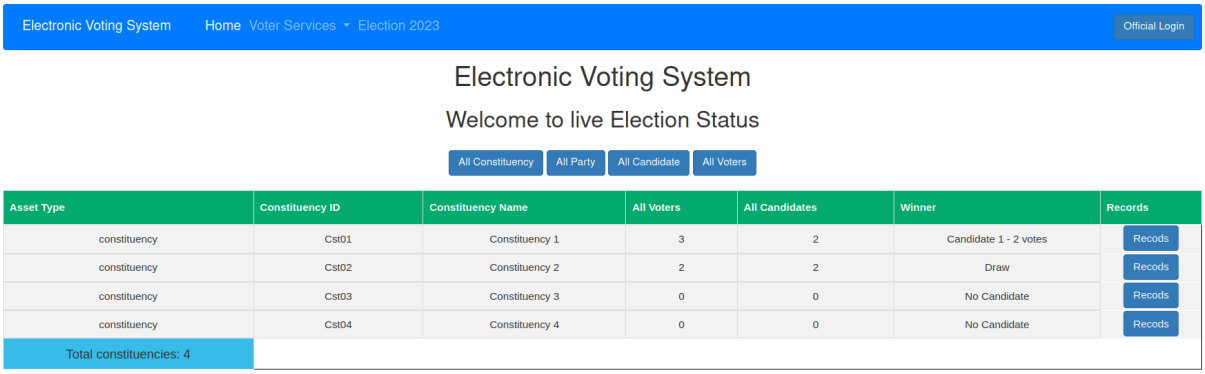
13-For Txn History click Voter Transactions and fill data of a voter and submit



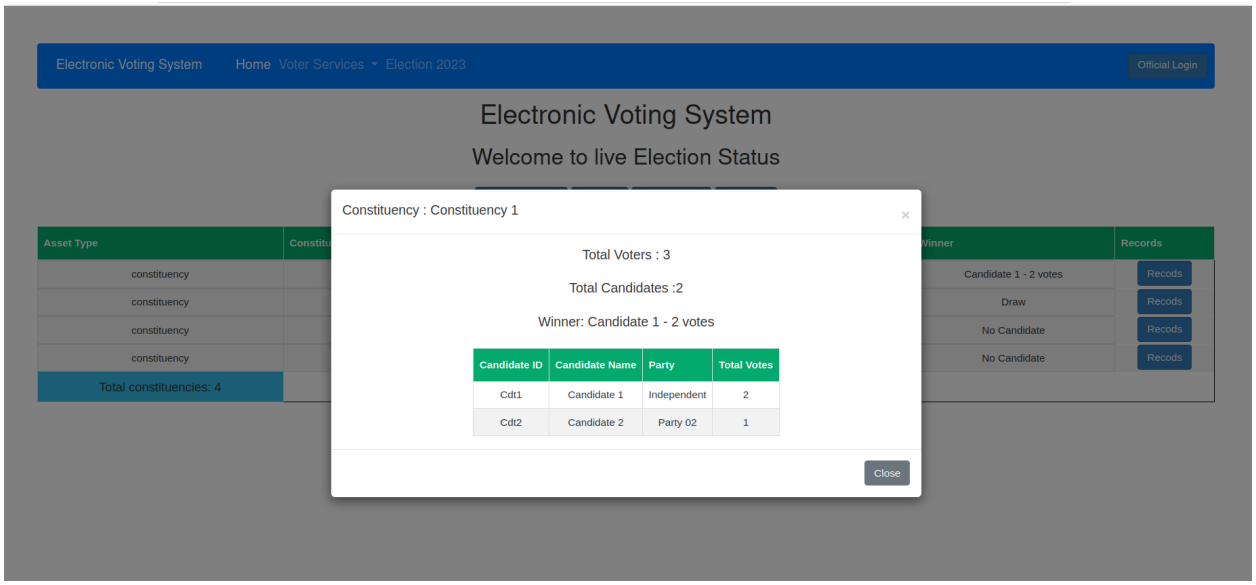
Results

14-Login as ECI Officer and click on Stop Voting and then logout.

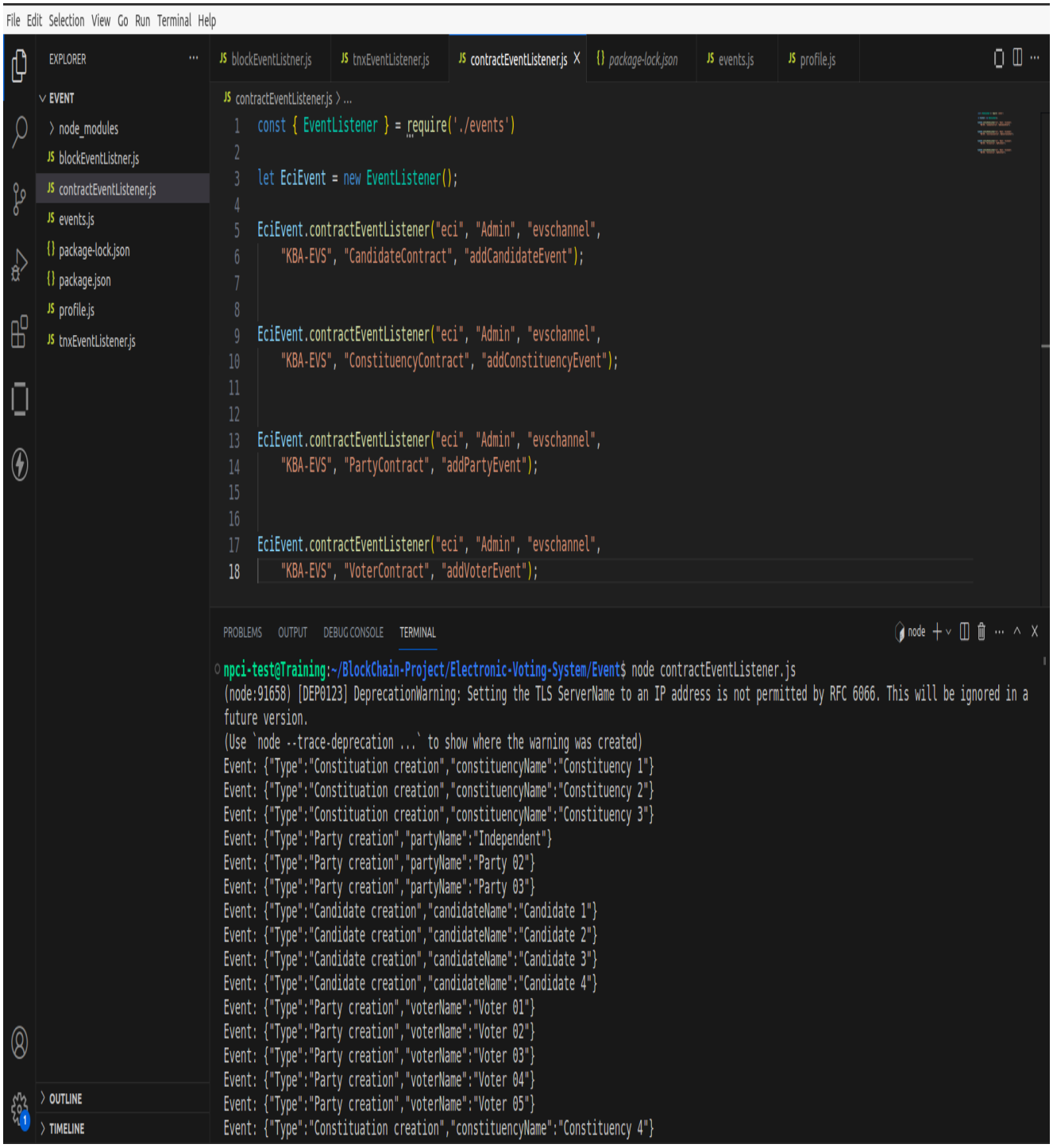
15- Click All Constituency ,Results will be display -



16-Click on records of any constituency



17- Events generated in event terminal



Chaincode Functions

A- Candidate-contract Functions

1. **candidateExists(ctx, candidateId)**: This function checks if a candidate with the given ID exists.
2. **createCandidate(ctx, candidateId, candidateName, constituency, party)**: This function creates a new candidate with the given information.
3. **readCandidate(ctx, candidateId)**: This function reads the information for a candidate with the given ID.
4. **updateCandidate(ctx, candidateId, candidateName, constituency, party)**: This function updates the information for a candidate with the given ID.
5. **deleteCandidate(ctx, candidateId)**: This function deletes a candidate with the given ID.
6. **getAllResults(iterator, isHistory)**: This function gets all the results from an iterator. The `isHistory` parameter indicates whether the results should be from the history of the asset.
7. **queryAllCandidate(ctx)**: This function gets all the candidates.

8. **totalCandidate(ctx, constituency)**: This function gets the total number of candidates in a constituency.
9. **getCandidateHistory(ctx, candidateId)**: This function gets the history of a candidate.
10. **addVote(ctx, candidateId)**: This function adds a vote to a candidate.
11. **getCandidateWithPagination(ctx, pageSize, bookMark)**: This function gets candidates with pagination. The `pageSize` parameter specifies the number of candidates to return per page. The `bookMark` parameter specifies the bookmark for the next page.

B- Constituency-contract Functions

1. **constituencyExists** checks if a constituency with the given ID already exists.
2. **createConstituency** creates a new constituency with the given ID and name.
3. **readConstituency** reads the details of a constituency with the given ID.
4. **updateConstituency** updates the details of a constituency with the given ID.
5. **deleteConstituency** deletes a constituency with the given ID.
6. **getAllResults** gets all the constituencies that match the given query.
7. **getConstituencyHistory** gets the history of changes to a constituency with the given ID.
8. **getConstituencyWithPagination** gets constituencies with the given pagination parameters.

C- Party-contract Functions

1.**partyExists():** This function checks if a party with the given ID already exists in the ledger.

2.**createParty():** This function creates a new party with the given ID and name.

3.**readParty():** This function reads the party with the given ID from the ledger.

4.**updateParty():** This function updates the party with the given ID in the ledger.

5.**deleteParty():** This function deletes the party with the given ID from the ledger.

6.**getAllResults():** This function gets all the parties in the ledger.

7.**queryAllParty():** This function gets all the parties that match the given query criteria.

8.**getPartyHistory():** This function gets the history of changes to the party with the given ID.

9.**getPartyWithPagination():** This function gets the parties with the given pagination criteria.

D- ***Vote-contract Functions***

(Private Data Collection -pdc)

1. **voteExists()** checks to see if a vote with the specified ID already exists. This is done by querying the blockchain for the private data hash for the specified ID. If the hash exists, then a vote with the specified ID already exists.
2. **createVote()** creates a new vote with the specified ID, voter ID, and candidate name. The vote is stored in the private collection on the blockchain. The vote is also hashed using a cryptographic hash function. The hash is then stored on the public ledger.
3. **readVote()** reads the vote with the specified ID from the private collection on the blockchain. The vote is decrypted and returned to the caller.
4. **updateVote()** updates the vote with the specified ID in the private collection on the blockchain. The vote is encrypted and then stored in the private collection. The hash of the vote is also updated on the public ledger.

5. **deleteVote()** deletes the vote with the specified ID from the private collection on the blockchain. The hash of the vote is also deleted from the public ledger.
6. **verifyVote()** verifies that the vote with the specified ID was cast by the voter with the specified MSP ID. This is done by comparing the hash of the vote with the hash stored on the public ledger. If the hashes match, then the vote was cast by the voter with the specified MSP ID.

E- Voter-contract Functions

1. **voterExists():** This function checks if a voter with a given ID exists.
2. **createVoter():** This function creates a new voter with the given ID, name, date of birth, and constituency.
3. **readVoter():** This function reads the voter data for a given ID.
4. **updateVoter():** This function updates the voter data for a given ID.
5. **deleteVoter():** This function deletes the voter data for a given ID.
6. **getAllResults():** This function gets all the voter data.
7. **queryAllVoter():** This function gets all the voter data in a paginated format.
8. **getVoterHistory():** This function gets the voter history for a given ID.
9. **enrollVoter():** This function enrolls a voter by setting their enrolled status to "Yes".
10. **castVote():** This function casts a vote by setting the voter's voteCasted status to "Yes" and the candidateName to the name of the candidate they voted for

Shortcomings and future enhancements

Shortcomings

- **Security:** Blockchain technology is still in its early stages of development, and there are concerns about its security. For example, blockchains can be hacked, and it is possible for malicious actors to alter or delete data.
- **Privacy:** Blockchain is a public ledger, which means that all transactions are visible to everyone on the network. This could raise privacy concerns, especially for voters who want to keep their vote confidential.
- **Accessibility:** Not everyone has access to the internet or a computer, which could make it difficult for some people to vote electronically.
- **Cost:** Developing and implementing an electronic voting system can be expensive. This could make it difficult for smaller organizations or governments to adopt this technology.

Future Enhancements

- **Improved security:** Researchers are working on ways to improve the security of blockchain technology. For example, they are developing new encryption techniques and ways to detect and prevent hacking.
- **Enhanced privacy:** Researchers are also working on ways to enhance the privacy of blockchain data. For example, they are developing ways to encrypt data so that it is only visible to authorized users.
- **Increased accessibility:** Researchers are working on ways to make electronic voting more accessible to people who do not have access to the internet or a computer. For example, they are developing ways to allow people to vote by phone or text message.
- **Reduced cost:** The cost of developing and implementing an electronic voting system is expected to decrease as the technology matures. This could make it more affordable for smaller organizations and governments to adopt this technology.