

DOCUMENTATION ON

ELECTRONIC VOTING SYSTEM

USING HYPERLEDGER BLOCKCHAIN

Name - Saurabh Kumar Rai

Emp Id- 24085

Table of Contents

1-About The Project

- a-Current Scenario

- b-Problem

- c-Solution(Advantages of introducing Fabric).

2-Workflow

- a-ECI Officer workflow (Admin)

- b-Voters Workflow

- c-Results

3-Application

- a- Prerequisites

- b- Installation

- c- How to run the Application according to the workflow.

4- Chaincode Functions

- a-Candidate Contract b-Party Contract

- c-Constituency Contract d-Voter Contract

5- Shortcomings and future enhancements.

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

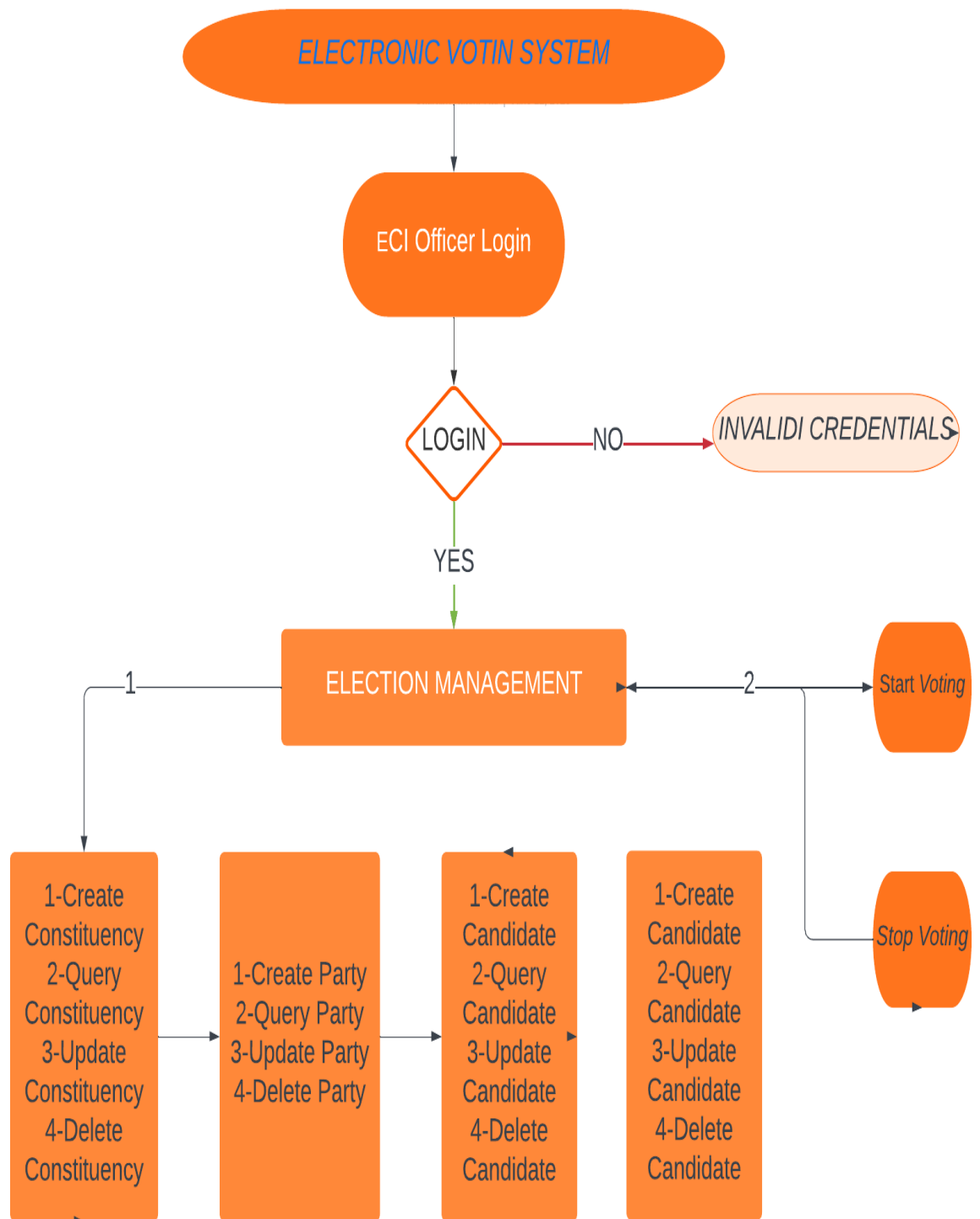
c-**Solution(Advantages of introducing Fabric)**

- **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

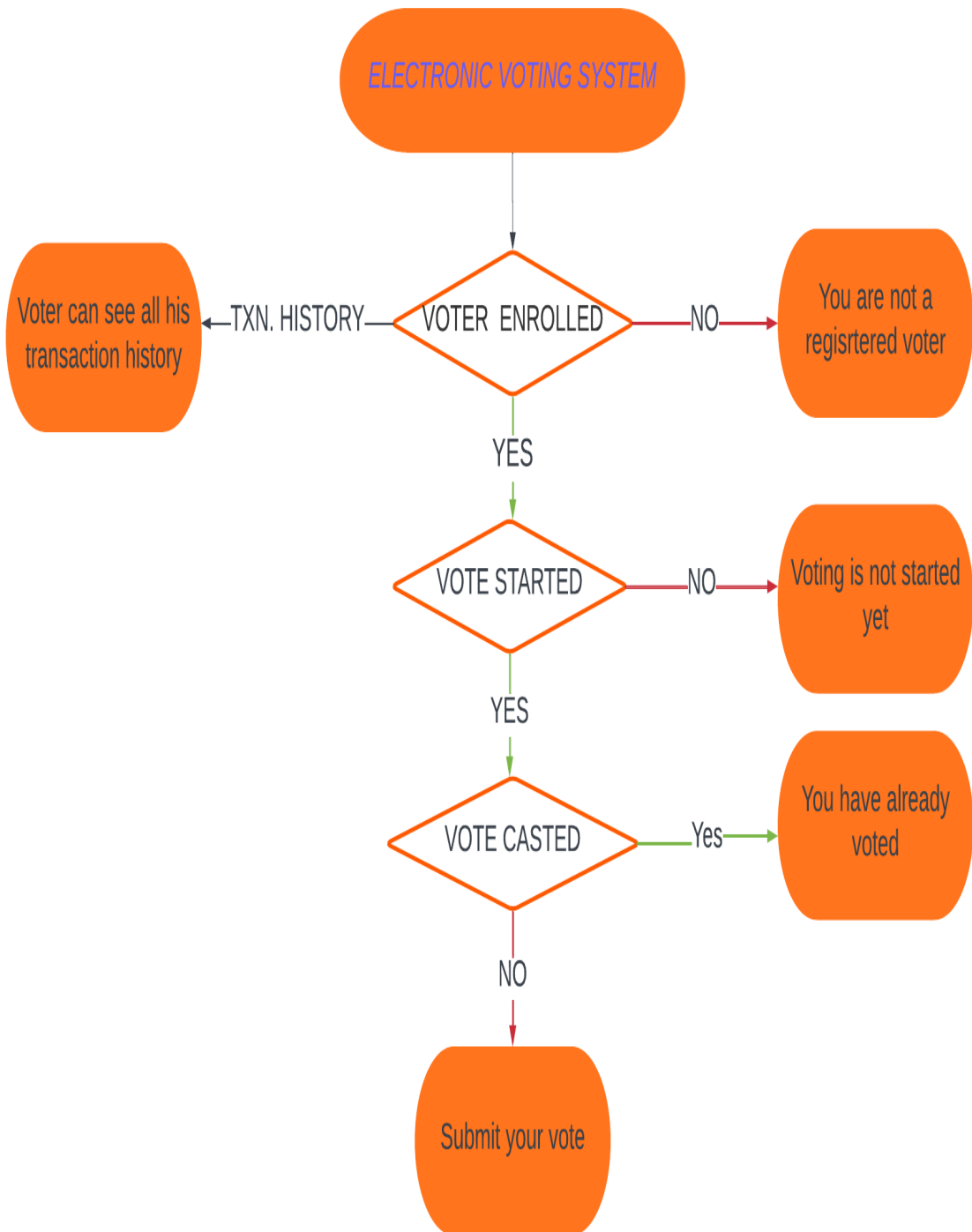
1- Workflow for an election commission officer:-

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

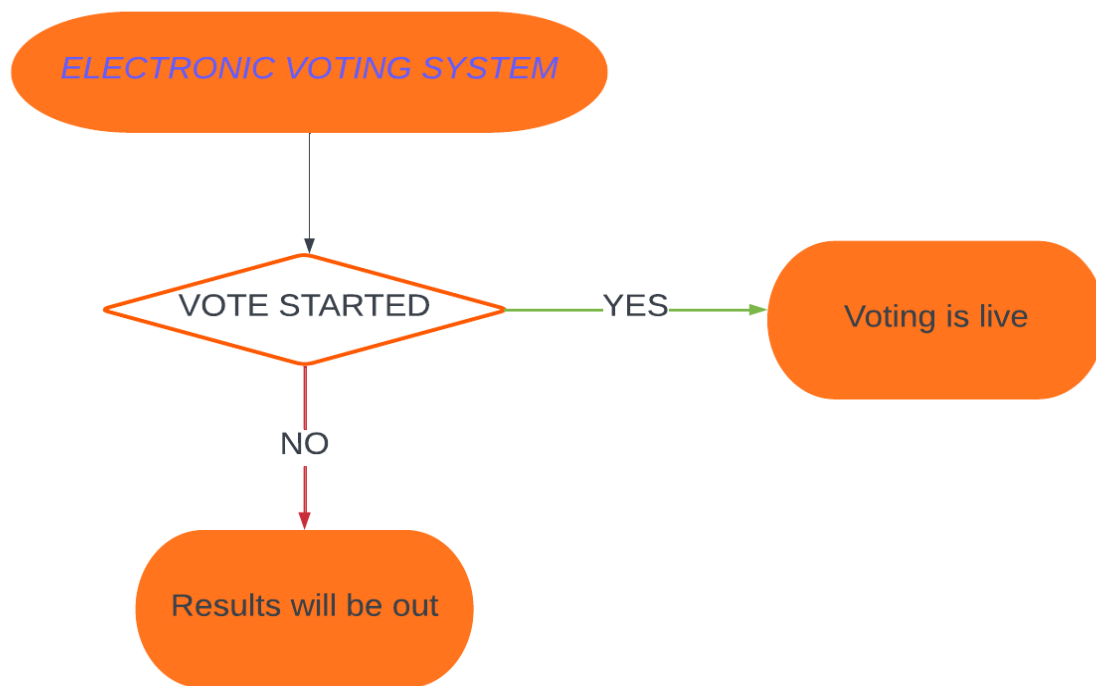


2- Workflow for a Voter:-

- 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



3- Results



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	Records
constituency	Cst02	Constituency 2	1	1	Candidate5 - 1 votes	Records
constituency	Cst03	Constituency 3	2	2	Draw	Records
constituency	Cst04	Constituency 4	0	1	No Winner -0 votes	Records
constituency	Cst05	Constituency 5	0	0	No Candidate	Records

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](https://raw.githubusercontent.com/ibm-blockchain/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

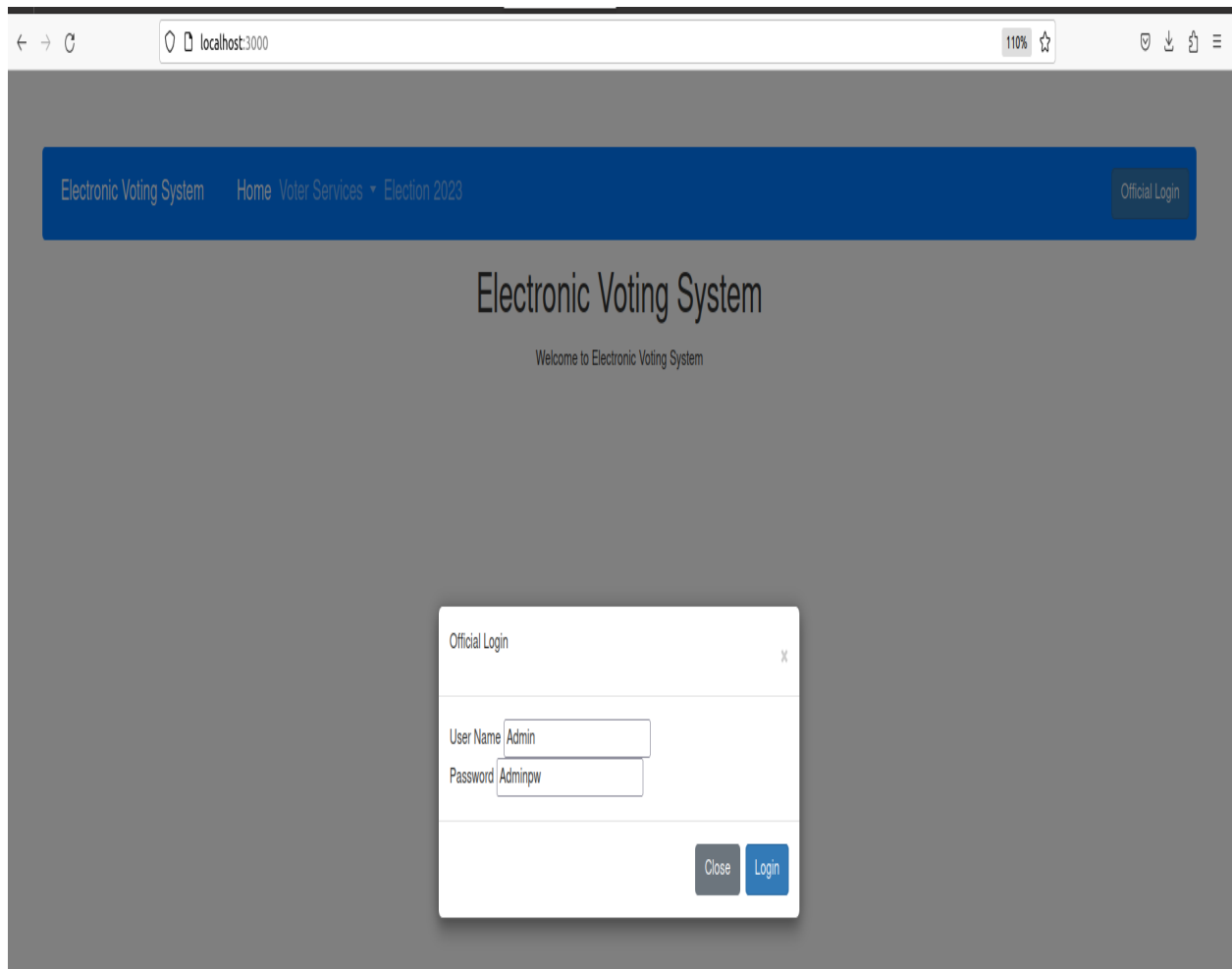
Click on <http://localhost:3000/>

ECI Officer workflow

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

7.b-Username-Admin

Password-Adminpw



The screenshot shows a web browser window at localhost:3000 displaying the 'Electronic Voting System' home page. The page has a dark blue header with the text 'Electronic Voting System', 'Home', 'Voter Services', and 'Election 2023'. A button labeled 'Official Login' is in the top right corner of the header. The main content area is gray and contains the title 'Electronic Voting System' and the subtitle 'Welcome to Electronic Voting System'. A modal window titled 'Official Login' is open in the center, featuring two input fields: 'User Name' with the value 'Admin' and 'Password' with the value 'Adminpw'. At the bottom of the modal are two buttons: 'Close' and 'Login'.

localhost:3000

Electronic Voting System Home Voter Services Election 2023 Official Login

Electronic Voting System

Welcome to Electronic Voting System

Official Login

User Name Admin

Password Adminpw

Close Login

8-Click Election Management and add Create constituency



- Create Constituency
- Create Party
- Create Candidate
- Create Voter

Electronic Voting System

Welcome to Electronic Voting System

Add New Constituency

ConstituencyId	Cst01
ConstituencyName	Constituency 1
Add New Constituency Clear	

Query Constituency

ConstituencyId	
Query Constituency Clear	



Electronic Voting System

Welcome to live Election Status

- [All Constituency](#) [All Party](#) [All Candidate](#) [All Voters](#)

Asset Type	Constituency ID	Constituency Name	All Voters	All Candidates	Winner	Records
constituency	Cst01	Constituency 1	0	0	No Candidate	Records
constituency	Cst02	Constituency 2	0	0	No Candidate	Records
constituency	Cst03	Constituency 3	0	0	No Candidate	Records
Total constituencies: 3						

9-Similarly add Party, Candidate, Voters

Electronic Voting System Home Election Management ▾ Voter Services ▾ Election 2023						Start Voting	Stop Voting	Logout
Electronic Voting System								
Welcome to live Election Status								
All Constituency All Party All Candidate All 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								

Electronic Voting System Home Election Management ▾ Voter Services ▾ Election 2023						Start Voting	Stop Voting	Logout
Electronic Voting System								
Welcome to live Election Status								
All Constituency All Party All Candidate All Voters								
Asset Type	Party ID	Party Name						
party	P1	Independent						
party	P2	Party 02						
party	P3	Party 03						
Total parties: 3								

Electronic Voting System Home Election Management ▾ Voter Services ▾ Election 2023						Start Voting	Stop Voting	Logout
Electronic Voting System								
Welcome to live Election Status								
All Constituency All Party All Candidate All Voters								
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

The screenshot shows the 'Electronic Voting System' interface. The top navigation bar is blue with links for 'Electronic Voting System', 'Home', 'Voter Services' (with a dropdown arrow), and 'Election 2023'. An 'Official Login' button is on the right. A dropdown menu under 'Voter Services' shows 'Enroll Yourself' and 'Voter Transactions'. The main heading is 'Electronic Voting System' with a sub-heading 'Welcome to Electronic Voting System'. Below this is the 'Enroll Yourself' section. It contains a form with fields for 'Voter Id' (V1), 'Voter Name' (Voter 01), and 'DOB' (15/06/2005). There are buttons for 'Enroll(UIDAI OTP Ver)' and 'Clear'. A 'Cast Your Vote' button is located at the bottom left of the form area.

Voter Id	V1
Voter Name	Voter 01
DOB	15/06/2005
<input type="button" value="Enroll(UIDAI OTP Ver)"/> <input type="button" value="Clear"/>	

11-After successful enrolling , cast your vote

The screenshot shows the 'Electronic Voting System' interface after successful enrollment. The top navigation bar is the same. The main heading is 'Electronic Voting System' with a sub-heading 'Welcome to Electronic Voting System'. Below this is the 'Cast Your Vote' section. It contains a form with fields for 'Voter Id' (V1), 'Voter Name' (Voter 01), and 'DOB' (15/06/2005). There are additional fields for 'Constituency' (Constituency 1) and 'Candidates' (Candidate 1 with a dropdown arrow). There are buttons for 'Vote' and 'Clear'.

Voter Id	V1
Voter Name	Voter 01
DOB	15/06/2005
Constituency	Constituency 1
Candidates	Candidate 1
<input type="button" value="Vote"/> <input type="button" value="Clear"/>	

12- Select a candidate and vote

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

Voter History

VoterId	<input type="text" value="V1"/>
Voter Name	<input type="text" value="Voter 01"/>
DOB	<input type="text" value="15/06/2005"/>
<input type="button" value="Voter Transactions (UIDAI OTP Ver)"/> <input type="button" value="Clear"/>	

All Transactions History

TxId: 02cf3190c56e52313bf0e2a63ac783051e32672a2d5d51c1afee7f06e58e295a

Timestamp: 1686594479.0.981

Record: { "Id": "V1", "assetType": "voter", "candidateName": "Candidate 1", "constituency": "Constituency 1", "dob": "2005-06-15", "enrolledStatus": "Yes", "voteCasted": "Yes", "voterName": "Voter 01" }

TxId: 9dd9e95e5983c67ed45006fb8e2b600752313814799c0bfd913d4641902430c3

Timestamp: 1686594416.0.242

Record: { "Id": "V1", "assetType": "voter", "candidateName": "", "constituency": "Constituency 1", "dob": "2005-06-15", "enrolledStatus": "Yes", "voteCasted": "No", "voterName": "Voter 01" }

TxId: c3dba2661c1c3bf23aa30524e800bbffef6702f56487ecf88c6e12e4cec752b4

Timestamp: 1686593876.0.266

Record: { "assetType": "voter", "Id": "V1", "voterName": "Voter 01", "dob": "2005-06-15", "constituency": "Constituency 1", "enrolledStatus": "No", "voteCasted": "No", "candidateName": "" }

Results

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

15- Click All Constituency ,Results will be display -

Electronic Voting System Home Voter Services Election 2023						Official Login
Electronic Voting System						
Welcome to live Election Status						
All Constituency All Party All Candidate All Voters						
Asset Type	Constituency ID	Constituency Name	All Voters	All Candidates	Winner	Records
constituency	Cst01	Constituency 1	3	2	Candidate 1 - 2 votes	Records
constituency	Cst02	Constituency 2	2	2	Draw	Records
constituency	Cst03	Constituency 3	0	0	No Candidate	Records
constituency	Cst04	Constituency 4	0	0	No Candidate	Records
Total constituencies: 4						

16-Click on records of any constituency

Electronic Voting System Home Voter Services Election 2023

Official Login

Electronic Voting System

Welcome to live Election Status

[All Constituency](#) [All Party](#) [All Candidate](#) [All Voters](#)

Asset Type	Constituency ID	Constituency Name	All Voters	All Candidates	Winner	Records
constituency	Cst01	Constituency 1	3	2	Candidate 1 - 2 votes	Records
constituency	Cst02	Constituency 2	2	2	Draw	Records
constituency	Cst03	Constituency 3	0	0	No Candidate	Records
constituency	Cst04	Constituency 4	0	0	No Candidate	Records
Total constituencies: 4						

Constituency : Constituency 1

Total Voters : 3

Total Candidates :2

Winner: Candidate 1 - 2 votes

Candidate ID	Candidate Name	Party	Total Votes
Cdt1	Candidate 1	Independent	2
Cdt2	Candidate 2	Party 02	1

Close

17- Events generated in event terminal

The screenshot shows a Visual Studio Code editor with the following components:

- EXPLORER:** A file tree on the left showing the project structure. The file `contractEventListeners.js` is selected.
- EDITOR:** The main workspace showing the content of `contractEventListeners.js`. The code defines an `EciEvent` class that inherits from `EventEmitter` and registers several event listeners for various events like "Constitution creation", "Party creation", "Candidate creation", and "Party creation".
- TERMINAL:** A terminal window at the bottom showing the output of running `node contractEventListeners.js`. The output displays a series of events generated by the application, each with a specific type and data.

```
JS contractEventListeners.js > ...
1  const { EventEmitter } = require('events')
2
3  let EciEvent = new EventEmitter();
4
5  EciEvent.contractEventListener("eci", "Admin", "evschannel",
6    "KBA-EVS", "CandidateContract", "addCandidateEvent");
7
8  EciEvent.contractEventListener("eci", "Admin", "evschannel",
9    "KBA-EVS", "ConstituencyContract", "addConstituencyEvent");
10
11
12
13  EciEvent.contractEventListener("eci", "Admin", "evschannel",
14    "KBA-EVS", "PartyContract", "addPartyEvent");
15
16
17  EciEvent.contractEventListener("eci", "Admin", "evschannel",
18    "KBA-EVS", "VoterContract", "addVoterEvent");
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
npici-test@Training:~/BlockChain-Project/Electronic-Voting-System/Event$ node contractEventListeners.js
(node:91658) [DEP0123] DeprecationWarning: Setting the TLS ServerName to an IP address is not permitted by RFC 6066. This will be ignored in a
future version.
(Use `node --trace-deprecation ...` to show where the warning was created)
Event: {"Type":"Constitution creation","constituencyName":"Constituency 1"}
Event: {"Type":"Constitution creation","constituencyName":"Constituency 2"}
Event: {"Type":"Constitution creation","constituencyName":"Constituency 3"}
Event: {"Type":"Party creation","partyName":"Independent"}
Event: {"Type":"Party creation","partyName":"Party 02"}
Event: {"Type":"Party creation","partyName":"Party 03"}
Event: {"Type":"Candidate creation","candidateName":"Candidate 1"}
Event: {"Type":"Candidate creation","candidateName":"Candidate 2"}
Event: {"Type":"Candidate creation","candidateName":"Candidate 3"}
Event: {"Type":"Candidate creation","candidateName":"Candidate 4"}
Event: {"Type":"Party creation","voterName":"Voter 01"}
Event: {"Type":"Party creation","voterName":"Voter 02"}
Event: {"Type":"Party creation","voterName":"Voter 03"}
Event: {"Type":"Party creation","voterName":"Voter 04"}
Event: {"Type":"Party creation","voterName":"Voter 05"}
Event: {"Type":"Constitution creation","constituencyName":"Constituency 4"}
```

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, candidateld)**: This function gets the history of a candidate.
10. **addVote(ctx, candidateld)**: 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.