PIMPRI CHINCHWAD EDUCATION TRUST'S

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

Mini Project Report

TITLE : " E-Voting using Blockchain "

SUBJECT: Computer Networks

Year: TE Computer (2015 Course)

Submitted By:

Group Members

Pranav More (TECOB 215)

Mohit Nakhale (TECOB 221)

Rohit Nawale (TECOB 222)

Kaustubh Patil (TECOB 230)

ABSTRACT

The blockchain technology is presented as a game changer for many of the existing and emerging technologies & services. With its immutability property and decentralised architecture, it is taking centre stage in many services. One of such potential applications of the blockchain is in e-voting schemes.

An electronic voting protocol provides end-to-end verifiability if the voter can verify that their vote was counted and any party can verify the results of the election. The objective of such a scheme would be to provide a decentralised architecture to run and support a voting scheme that is open, fair and independently verifiable. We have tried to implement a new e-voting protocol that utilises the blockchain as a transparent ballot box.

All of this is achieved without sacrificing voter privacy or ballot integrity. The resulting system shows clear potential for Blockchain technology to become a central part of applications wishing to provide transparency and security in public scenarios.

a) Problem Statement:

The current voting system has many flaws such as vulnerability to fraud, susceptibility to hacking. We aim at reducing these flaws by implementing the concept of blockchain in e-voting.

b) Objectives of project:

The objective is to provide an alternate voting system which is fair, more secure and transparent. The system also looks forward to reduce the human efforts and make the process of voting hassle free and minimise human errors.

c) Scope:

The E-Voting system seeks to computerize the voting process through a convivial, efficient and easy-to-use graphical interface.The proposed system runs on the local network without internet connection. The system demanded the implementation of distributed database and distributed ledger. The result of the election is available everytime with all the clients.

LIST OF FIGURES & TABLES

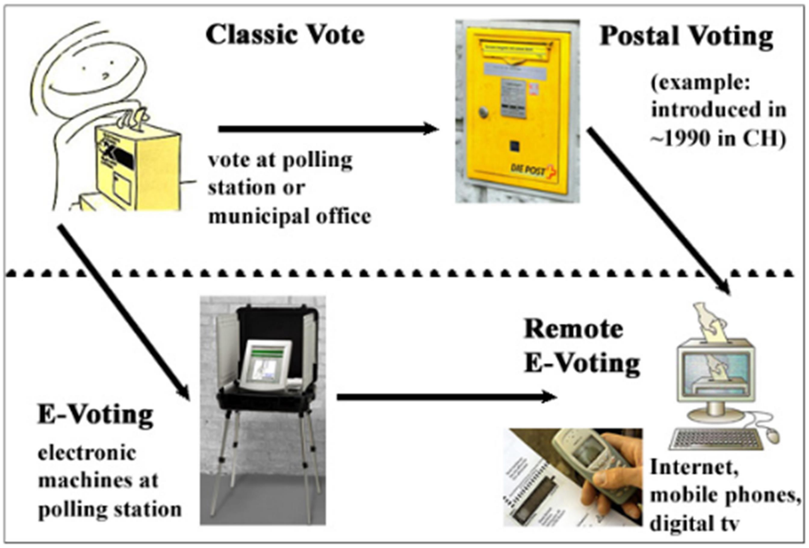


Figure 1: Example of voting evolution

PROJECT DESIGN

a) Technical Details:

Platform: PyCharm, Oracle 11g
Language: Python 3.7, SQL
API's Libraries: cx_Oracle, Tkinter
Packages:

b) Architecture diagrams / flow diagram:

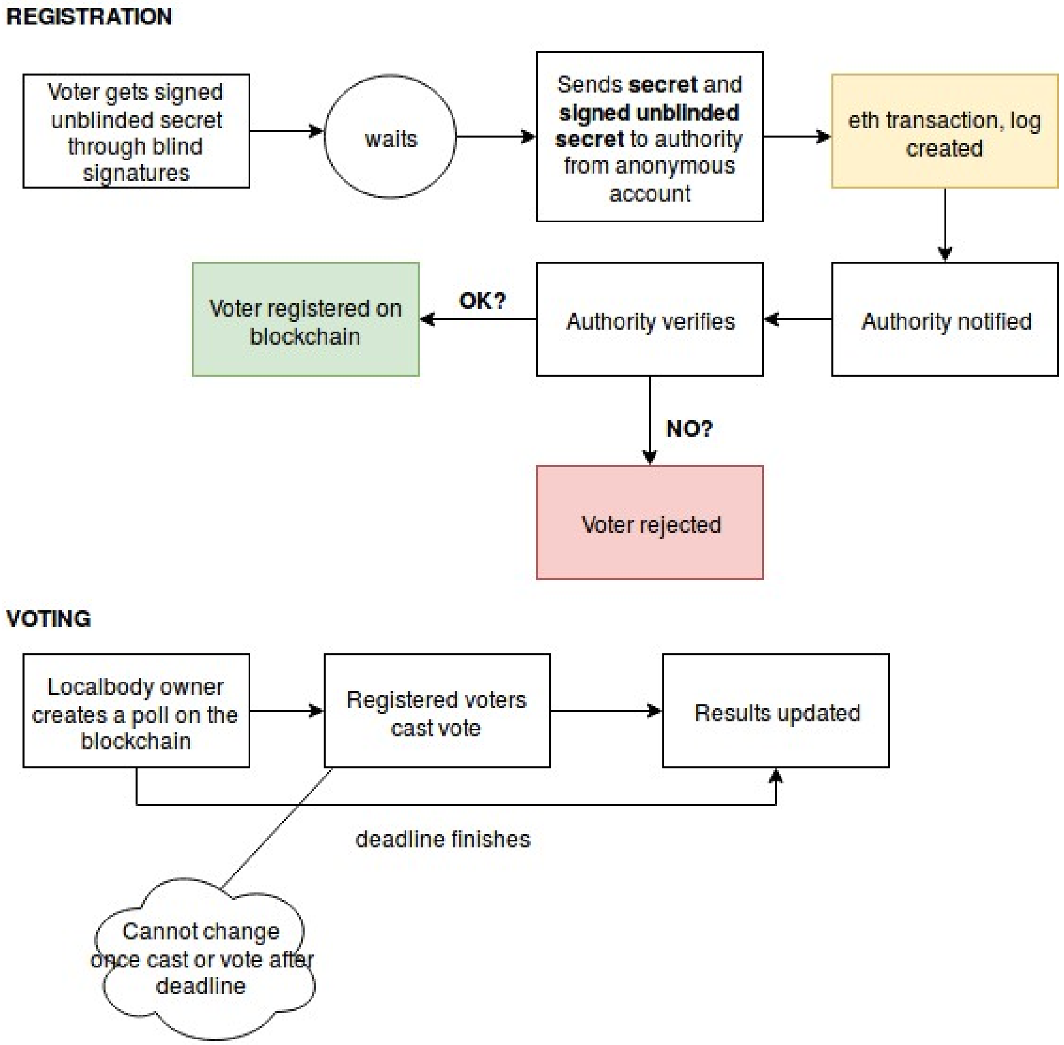

REGISTRATION
Voter gets signed unblinded secret through blind signatures
waits
Sends **secret** and **signed unblinded secret** to authority from anonymous account
eth transaction, log created
Authority notified
Authority verifies
OK?
Voter registered on blockchain
NO?
Voter rejected
VOTING
Localbody owner creates a poll on the blockchain
Registered voters cast vote
Results updated
deadline finishes
Cannot change once cast or vote after deadline

Figure 2: Flow Diagram

c) Detailed design of modules/functionality with explanation

Server: All the servers are always on and run simultaneously in order to support Distributed Database.

1. **vserver.py**

   The code snippet uses the TCP IP Protocol and accepts the “username” and “password” entered by the client at client side. The function named **se()** then validates the credentials received by searching in the database stored in the server side. If the credentials are valid and user has not voted earlier then server sends an encoded msg **“yes”** to the client side which grants user the further permission to vote, otherwise an error msg is displayed as msg **“no”** is sent from server.

2. **a_server.py**

   Once the credentials are verified the program controls switches to this server as it updates the voted status of the client to “**YES**” which does not allows the same voter to vote again for the same election.

3. **messenger_server.py**

   This server is responsible for collecting the votes from all the eligible clients via TCP IP protocol and increments the count of votes for respective party in the server side database. It also maintains the record of all the connections established during the voting.

4. **sendData.py**

   This server tries to update the databases of all the clients by sending the data to all the clients every second.

5. **servergui.py**

   This file deals with the GUI of the server side. The server side GUI has 3 main functions which are as follows:

   - Sign Up: The sign up attribute allows the server to add a new person into the voting list database on it’s side.
   - Analysis: This attribute brings up the statistical analysis of the voting.
   - Result: This attribute reveals the results of the election.

Client:

1. **Firstpage.py**

   This program handles the GUI on the client side and allows the user to login and cast vote in the favor of the party. It is the 1st page to start execution.

2. **getData.py**

   It timely receives the data from the server votes table and updates the received data in the client votes table.

3. **vclient.py**

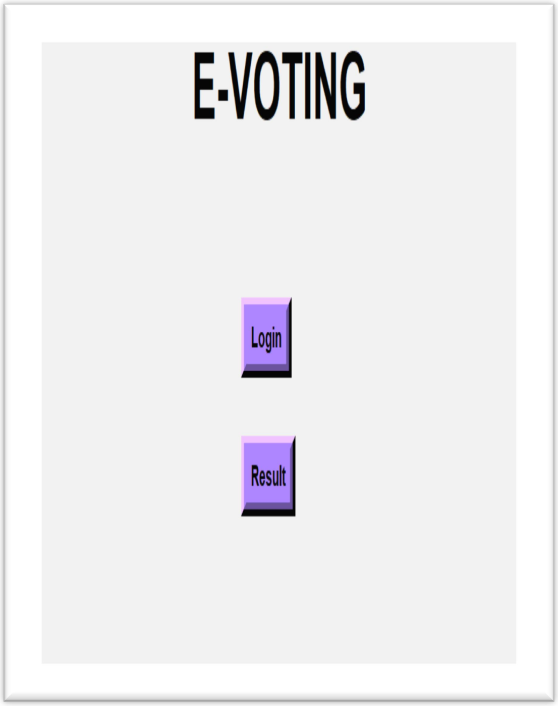
   The code snippet uses the TCP IP Protocol and send the “username” and “password” entered by the client at client side. The function named **se()** then validates the credentials received by searching in the database stored in the server side. If the credentials are valid and user has not voted earlier then client receives an encoded msg **“yes”** to the client side which grants user the further permission to vote, otherwise an error msg is displayed as msg **“no”** is sent from server.

4. **messenger_client.py**

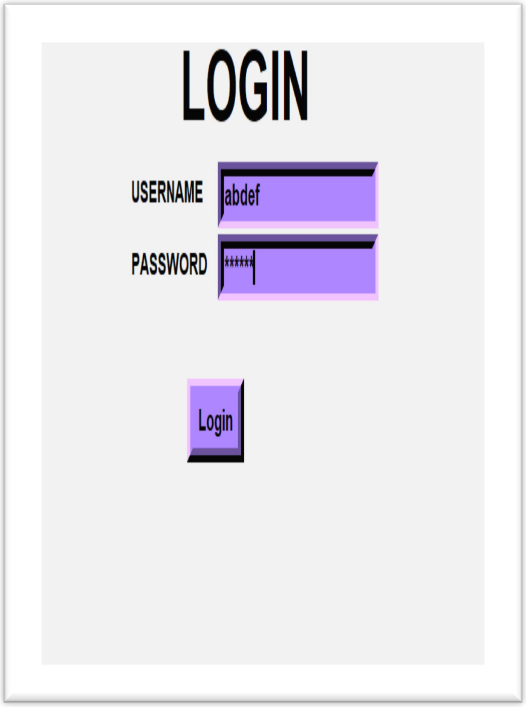
   This client is responsible for sending the votes from all the eligible clients via TCP IP protocol and increments the count of votes for respective party in the client side database. It also maintains the record of all the connections established during the voting.

5. **Node_server.py**

   It creates the blockchain and adds the blocks to the chain.It uses SHA256 algorithm to calculate the 256bits hash value which is unique for every block in the chain.



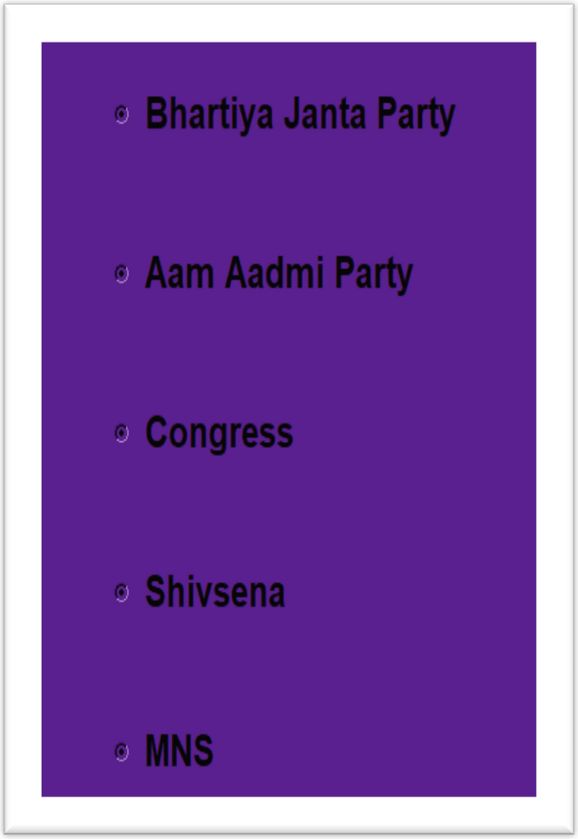
E-VOTING



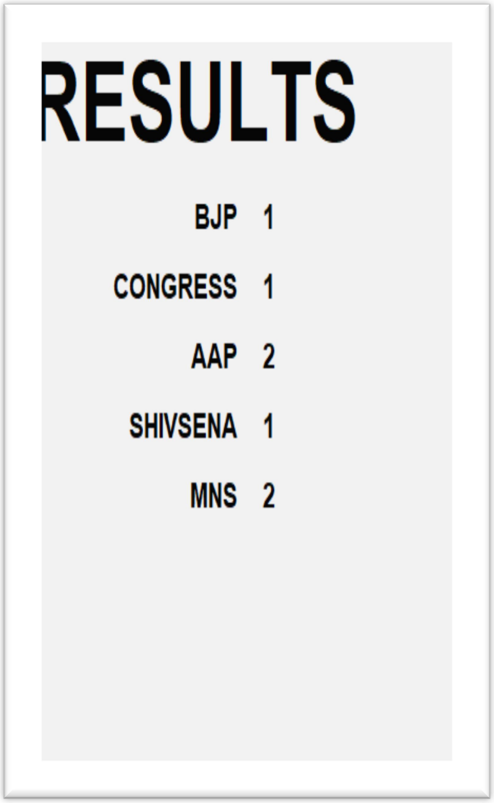
LOGIN

USERNAME abdef

PASSWORD *****

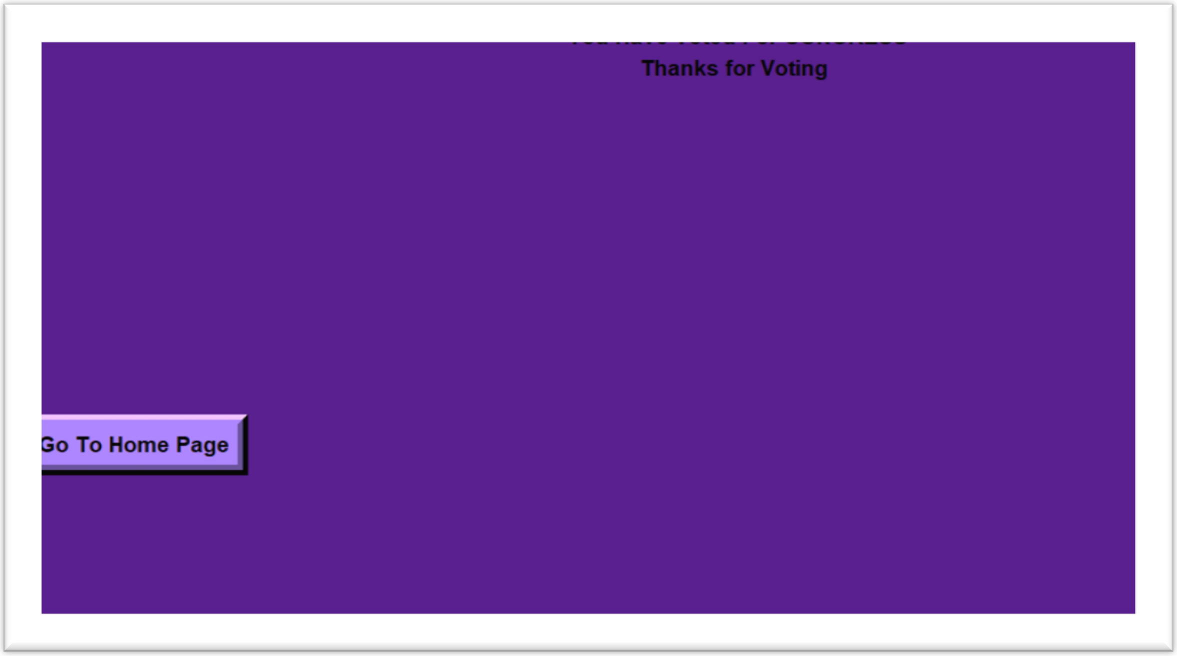


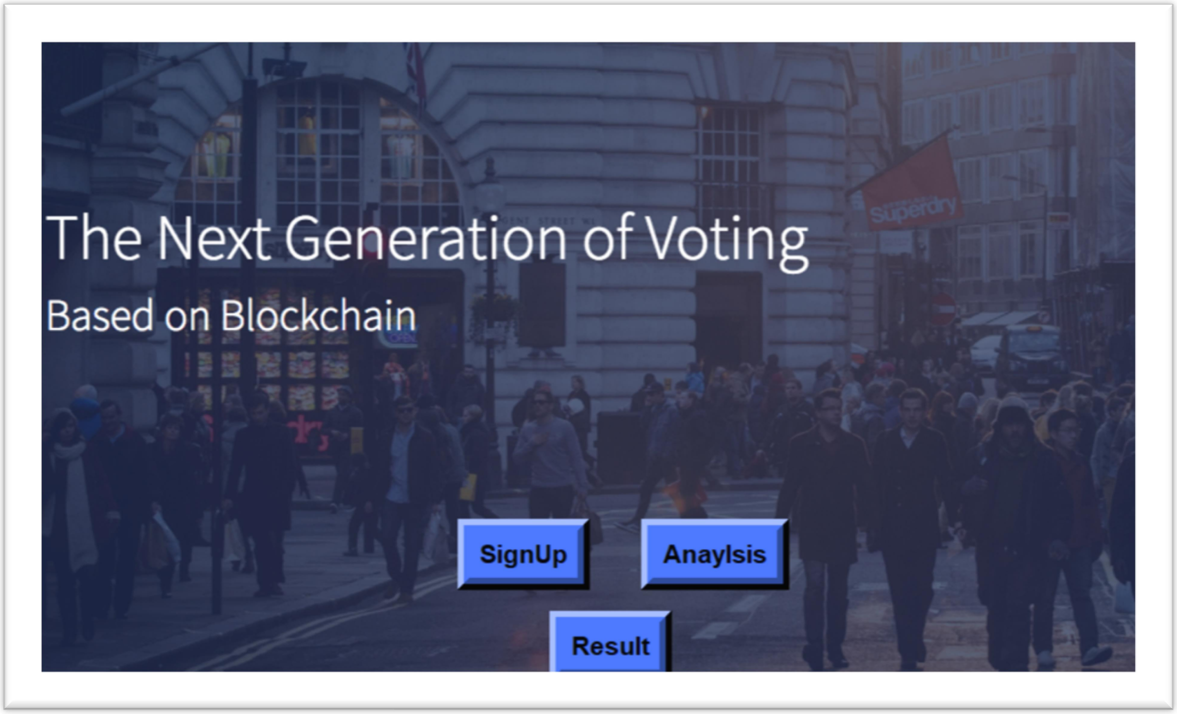
- ○ Bhartiya Janta Party
- ○ Aam Aadmi Party
- ○ Congress
- ○ Shivsena
- ○ MNS



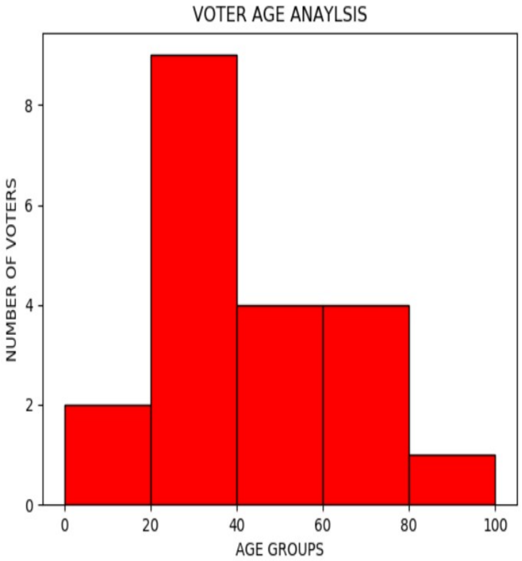
RESULTS

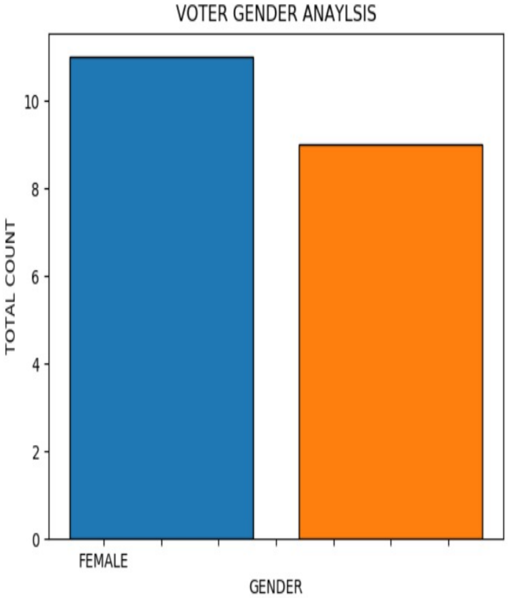
BJP	1
CONGRESS	1
AAP	2
SHIVSENA	1
MNS	2

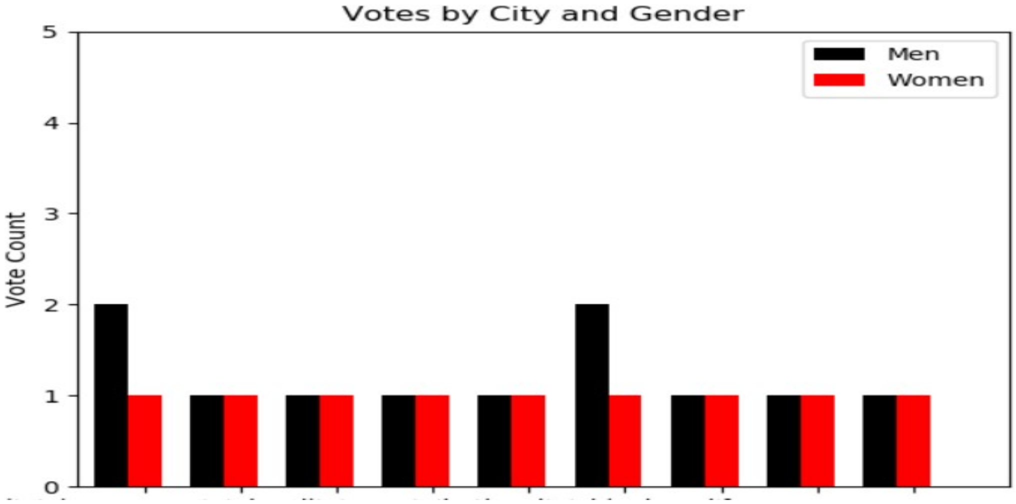

Thanks for Voting

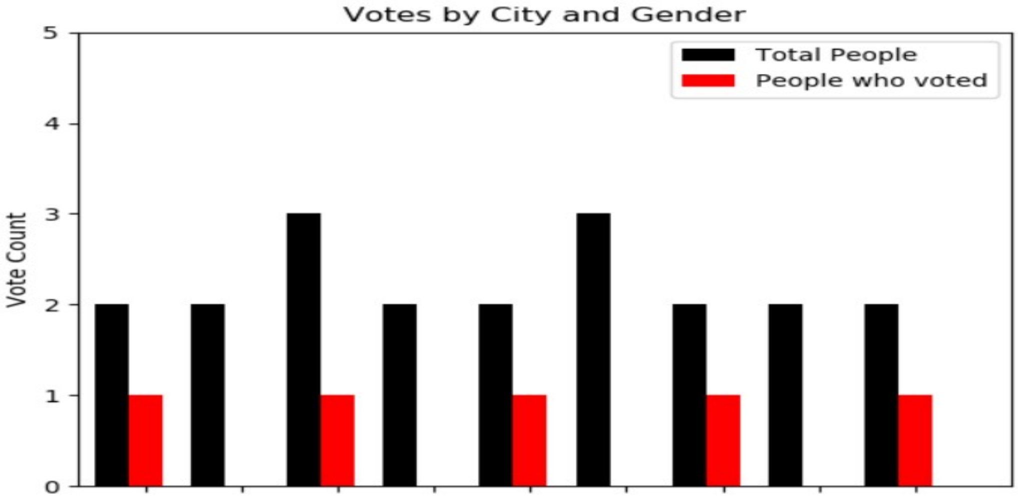

The Next Generation of Voting

Based on Blockchain


VOTER AGE ANAYLSIS
NUMBER OF VOTERS
0
2
4
6
8
0
20
40
60
80
100
AGE GROUPS


VOTER GENDER ANAYLSIS
TOTAL COUNT
0
2
4
6
8
10
FEMALE
GENDER


Votes by City and Gender
Men
Women
Vote Count
0
1
2
3
4
5


Votes by City and Gender
Total People
People who voted
Vote Count
0
1
2
3
4
5

Conclusion:

Thus we implemented E-Voting with the help of blockchain. The client is able to see the result on its side and server performs the validation of data and the user and timely maintains the votes count at all the clients with the help of distributed database. The server performs various analysis on the data ie. Age, gender, city wise analysis of voters and plots the graphs .