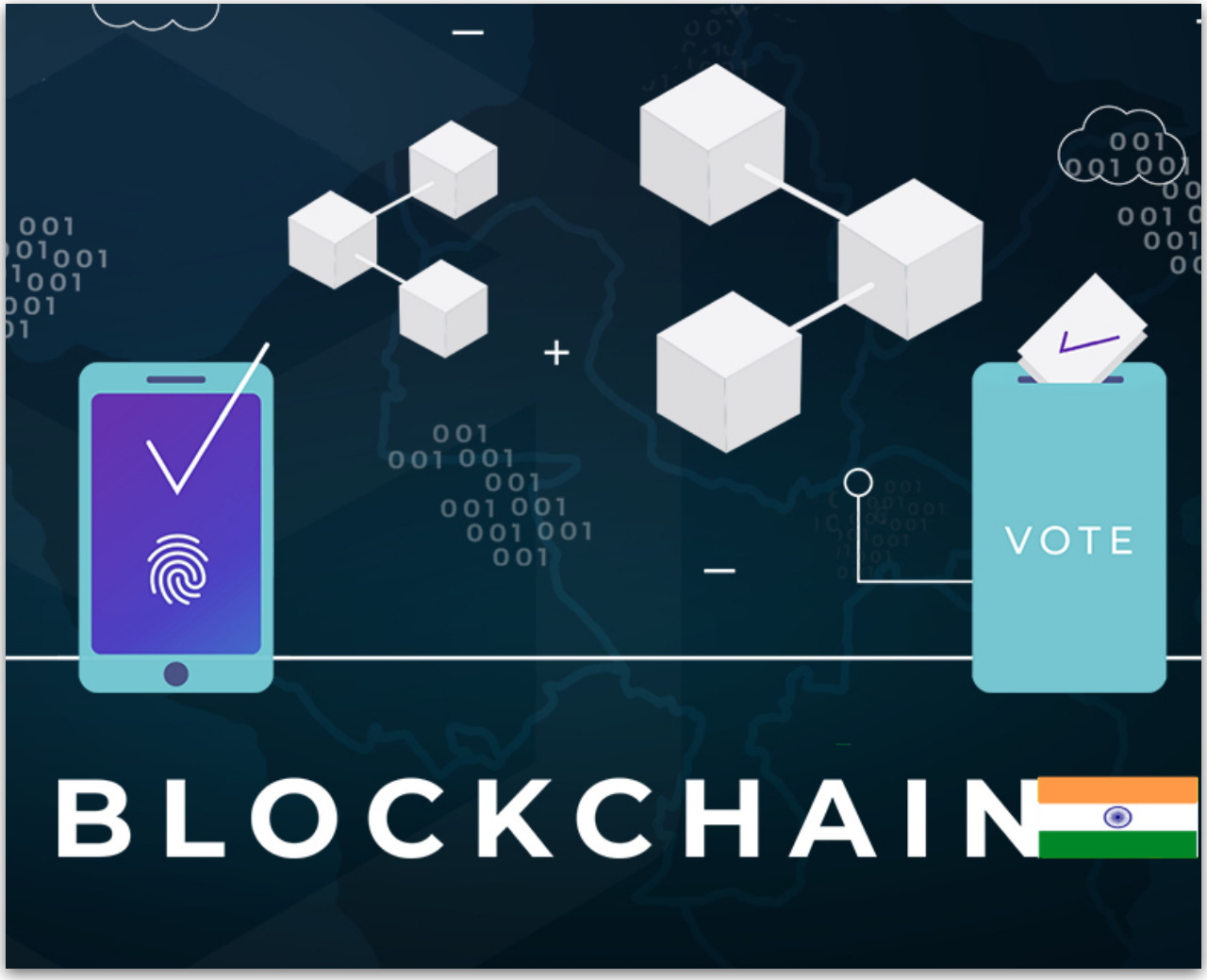


Blockchain Based Election

Prepared by: Manikya Bardhan
College: PES University

SUMMARY

Objective

To create a Decentralised Application which will make the entire process of voting more secure, transparent, and efficient.

Goals

- To allow people to vote securely.
- To expedite the process of vote counting.
- To ensure that every citizen over 18 can vote without applying for a voters id.
- To ensure that vote tampering is not possible and render all such claims invalid.
- To allow people to vote even from another constituency where they are registered as a voter.

Project Outline

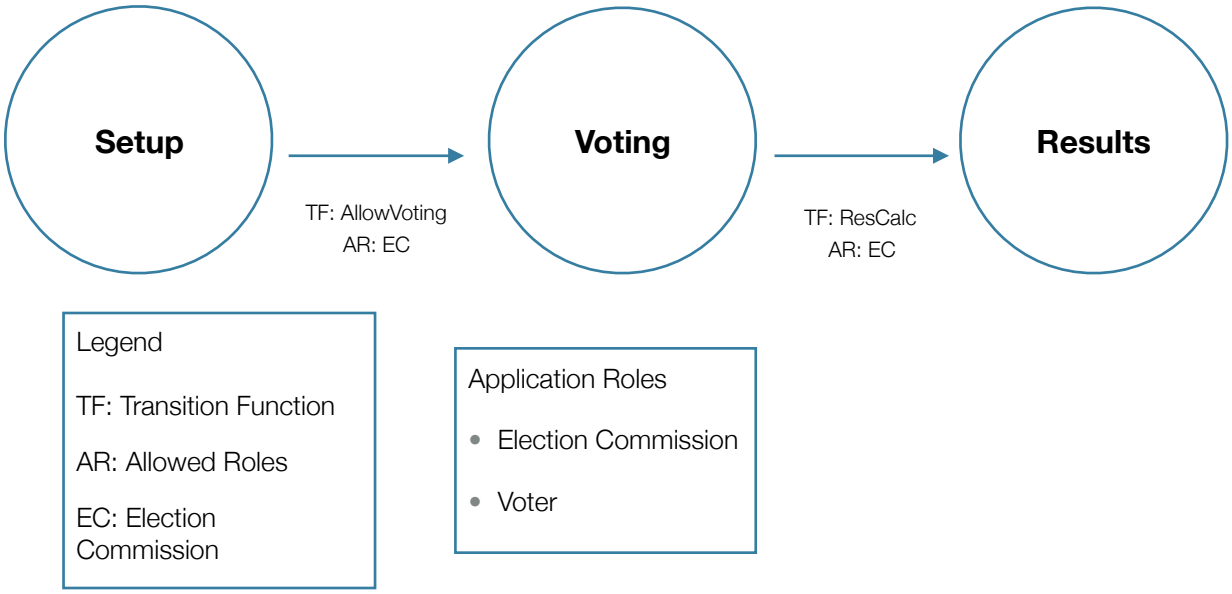
Application Roles

Name	Description
Election Commission	The party which initiates the election and declares the result
Voter	The person who votes for their favourite candidate

States

State	Description
Setup	In this, the election commission may add the candidates and setup the election
Voting	Indicates that the voter may vote for their favourite candidate
Results	In this, the election commission may declare the results

Workflow Details



A new Election smart contract may be created by the Election Commission (EC) and the name of the Election may be entered in the constructor. The initial state is the Setup State in which the EC has the permission to add candidates by invoking the function addCandidate. After the setup process is done, the EC may use the AllowVoting function to transition to the Voting State where the voter may vote for their favourite candidate. After voting, the voter will receive an email from the EC with the transaction id of the transaction which corresponds to their vote, which will allow them to be completely sure that their vote was properly registered. When the voting process is completed, the EC may calculate and declare the results by invoking the function ResCalc which expedites the process of vote counting. The winner of the election is shown in the table of the smart contract.

To ensure that every person who is 18 and above can vote, we can use the python script (People.ipynb) to filter out and create a csv with only people who are 18 years old. Now this csv is added directly to Azure Active Directory (AD) which assigns an ethereum id to them so that the new voters are able to vote on the blockchain without explicitly doing anything for it. The Aadhar database can be used for this purpose and this process will have to be repeated before the elections.

AZURE SERVICES USED:

Azure Blockchain Workbench

This has been used to create the decentralised application on the blockchain and contains the main logic for the election process. A configuration file (Election.json) and solidity file (Election.sol) was used to set up the Dapp.

Logic Apps

This has been used to fully take advantage of the messaging API of Azure Blockchain Workbench to link it such that whenever the voting function is invoked in the workbench application, the transaction id of that transaction is sent to the voter who cast that vote.
This transaction id can now be checked anytime to ensure that your vote is properly cast and thus render all claims of vote tampering invalid.
The code for the logic app is included in github (https://github.com/manikyabard/Blockchain-Election/blob/master/logic_app_code.json).

Azure Active Directory

This has been used to keep track of all the voters and assign an ethereum id as a key value pair with their email id. Also, those who turn 18 are added to the Azure Active Directory and in this way no prospective voters are left out. A python script has been used to automatically find all those who turned 18 before the election and create a csv file. This csv file is now added directly to the Azure AD.

Link to GitHub

https://github.com/manikyabard/Blockchain-Election

It contains all the code used in this project.