

A Project Report  
On  
**Electronic Voting Systems via Blockchain**

BY  
**RIJUL DHINGRA 2018B4A80807H**  
**HARDIK BHALLA 2018B4A30263H**

Under the supervision of  
**Prof. G GEETHAKUMARI**

**SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS OF**  
**BITS F463 CRYPTOGRAPHY – TERM PROJECT**



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)**  
**HYDERABAD CAMPUS**  
**(APRIL 2022)**

# PROBLEM STATEMENT

## 1.1 Introduction

There has been a lot of buzz around blockchains in recent years, ever since Bitcoin became popular. Blockchain technology offers new tools for authentication and authorization in the digital world that preclude the need for many centralized administrators. The scope of this project is to help you get accustomed to blockchain development. You are required to identify one problem that you feel can be solved with the help of blockchain and then implement your solution. You can work in teams which you have already formed earlier.

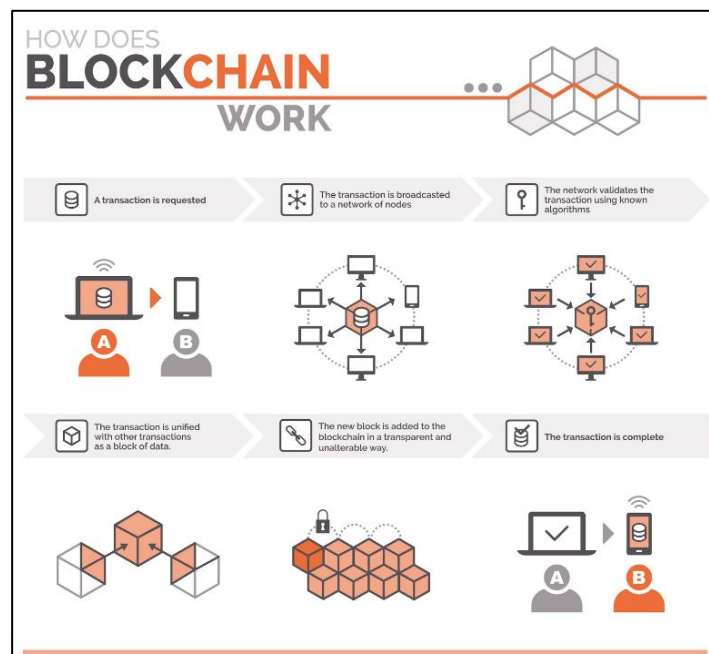
### 1.2.1 Blockchain Fundamentals

A blockchain is a digital and distributed ledger of transactions, recorded and replicated in real-time across a network of computers or nodes. Every transaction must be cryptographically validated via a consensus mechanism executed by the nodes before being permanently added as a new “block” at the end of the “chain.” There is no need for a central authority to approve the transaction, which is why blockchain is sometimes referred to as a peer-to-peer trust less mechanism.

Blockchain can be thought of as a linked list with each node containing multiple transactions. Each transaction has a hash that depends on the previous transactions hash as well. So, we can see that the order of transactions is important. If we were to change one transaction somewhere, it would have a ripple effect and change the hash of all subsequent transactions. This is one of the reasons why blockchain is a powerful medium for storing transactions.

The placing of a transaction in a block is called a successful conclusion to a proof of work challenge, and is carried out by special nodes called miners. Proof of Work is a system that requires some work from the service requester, usually meaning processing time by a computer. Producing a proof of work is a random process with low probability, so normally a lot of trial and error is required for a valid proof of work to be generated. When it comes to Bitcoins, hash is what serves as a proof of work. Miners on a Blockchain are nodes that produce blocks by solving proof of work problems. If a miner produces a block that is approved by an electronic consensus of nodes then the miner is rewarded with coins. This essentially is the crux of blockchain. Proof of Work is what is keeping all transactions on the blockchain secure and protecting it from malicious attempts to alter these transactions.

This is just a high level introduction. You are expected to dig deep and understand how the blockchain truly works. Different applications might implement it differently, but the core ideas remain the same.



### 1.2.2 Quick Introduction to Zero-Knowledge Proofs

Zero-Knowledge Proofs (ZKP) refer to a proof construction that allows you to convince someone that you know something without revealing any information about what it is you know in the process. To explain ZKPs with the help of an example consider the following scenario:

#### Scenario:

Imagine two friends, Alice & Bob. Bob is color-blind. Alice has two identical balls that only differ in color: red and green. To Bob they seem completely identical and he is skeptical that they are actually distinguishable. Alice wants to prove to him they are in fact differently-colored, but nothing else, thus he does not reveal which one is red and which is green.

Alice gives the two balls to Bob and he hides it. Next, he takes out one ball and displays it. After that he hides the ball again and shows a ball. There is a 50% probability that he switched the balls. Alice is asked if the ball was switched. She could guess and answer correctly with a probability of 50% but if this exercise is repeated multiple times, we can see that this probability will eventually become negligible. So with 5 rounds, he will have a 1 in 32 chance of successfully faking. With 10 rounds, it is 1 in 1024, and with 20 rounds, it is about one in a million. This way one can reach any probabilistic level of proof that is desired, although an absolute certainty can never be achieved.

The above proof is zero-knowledge because Bob never learns which ball is green and which is red; but he can indeed verify that the balls differ in color.

As part of the assignment you will have to implement a similar proof when handling operations on data that you feel should remain secure (example your Aadhar Number when doing some sort of authentication). An algorithm for the same is outlined below.

Alice has sensitive data  $x$  for which she chooses two numbers  $p$  and  $g$ .  $p$  can be a large prime and  $g$  is a generator for  $p$  (the meaning of these terms will be covered as the course progresses. For now you can assume  $p=11$  &  $g=2$ ). She calculates  $y$  as  $y=g^{x \bmod (p-1)}$ . Now she performs the following steps to create a zero knowledge proof for  $x$ .

1. Alice chooses a random number  $0 \leq r < p-1$  and sends it to Bob as  $h=g^r \bmod (p)$
2. Bob receives  $h$  and sends back a random bit  $b$  (could be 0/1).
3. Alice sends  $s=(r+bx) \bmod (p-1)$  to Bob.
4. Bob computes  $g^s \bmod (p)$  which should equal  $hyb \bmod (p)$

Here Bob acts as a verifier and checks if Alice knows the value of  $x$  without actually getting to know what  $x$  is. You will have to implement the same in your blockchain application. While implementing the proof it will be up to you to decide on the number of rounds this should go on for.

### 1.3 Deliverables

(a) A brief presentation on the problem that you have identified and how it can be solved through the use of blockchain. Register your team and topic via the google sheet shared already. The presentation should also include details of all team members. The deadline for the same is 11:59 PM 13th April, 2022. From a team, only one team member needs to do the submission in the drive folder which we have earlier shared with you all [BITS F463 Assignment 2](#).

(b) You are free to code in any language but your solution must implement the following methods:

- createBlock()
- verifyTransaction()
- mineBlock() or something equivalent for proof of work
- viewUser()

(c) All transactions must be verified before they can be added to a block. As part of the verification process, you are required to use a zero-knowledge proof (as described in section 2.2) to verify at least one attribute.

(d) viewUser() should list all (successful) transactions against the user.

(e) For your final submission, submit your source code & readme as a single .zip or .tar file [here](#). Please name your file as bitsf463\_team99 (assuming your team number is 99. You can find your team number from the google sheet you filled earlier). The readme should contain a brief explanation of the project, steps to run the code, and the list of team members. The deadline for the same will be 11:59 PM 25th April 2022. The exact date and demo schedule will be intimated later. From a team, only one team member needs to do the submission.

## 1.4 Further Reading

- [What A Blockchain Actually Is, Written In Blockchain](#)
- [Bitcoin White Paper](#)
- [How Do Bitcoin Transactions Work?](#)
- [WTF is The Blockchain](#)
- [Creating Your First Blockchain](#)
- [Zero Knowledge Proofs](#)
- [Zero Knowledge Proofs using Discrete Log](#)
- [Lectures & Slides on Cryptocurrency from Princeton](#)

# HOW BLOCKCHAINS TAKE VOTING FORWARD?

## **History of Traditional Voting Methods:**

Voting & elections in general form the backbone of any democracy. 167 countries worldwide out of nearly 200 follow the system of democracy and we've seen great progress in terms of voting methodologies over time. However, none of the systems being followed for elections today are flawless.

Paper Ballot has been the traditional method used for vote counting. While it can be accessed by anyone and everyone regardless of their educational status, it brings along with it greater cons like time consumption for counting of votes and the need for reliability and trust during the vote counting procedures. And while the initial costs of Paper Ballots are low, in the long run it can be degrading for the environment.

The digital shift in voting came in terms of Electronic Voting Machines (EVMs) which uses the modern technologies to cast as well as tally votes. The primary advantage of using electronic systems is the response speed, with the long term costs and environmental factors also pitching in as significant pros. However, there are arguments against E-Voting that have prevented a worldwide shift. Unless this voting takes place in person – there is a risk of identity fraud. Secondly, the physical tampering of an EVM can influence millions of votes without getting detected effectively.

## **Requirement of a Central Trusting Authority:**

The current state of the art systems work solely with the presence of a trusting authority that acts in an unbiased manner. Whether it's the Election Commission of India acting for the interest of the nation or our very own College's Election Council – The responsibility of ensuring a impartial election for all candidates rests on the shoulders of a few do-good Samaritans.

However, as long as these processes require the public to "Trust" someone – there will always be a scope to question the results and the procedures of the elections.

Electronic voting protocols have a single authority that are responsible to oversee the complete voting process. However, this central authority's dishonesty can lead to erroneous elections and this fault cannot be rectified using the existing methods. Hence, the proposed solution of a decentralized network that removes the need for an overseeing authority appeals to be an ideal way forward.

The decentralization circumvents the need to have an authority with the power to influence elections while also providing certain benefits that add on to the existing methods or at least maintain the current standards of expectations.

## **Integrity**

Blockchains were made to be immutable, as in the data that once becomes a part of the blockchain cannot be modified or tampered with, thereby ensuring greater security. Hence, ensuring that the vote that gets cast goes to its intended target and not be manipulated. Such frauds could take place in paper ballot-based voting systems where a fraudulent counter could effectively change a vote and declare a biased false count as the result.

## **Anonymity**

Anonymity remains a very important part of free and fair elections where no voter can be held accountable for their choice of vote by any candidate. While the data storing capabilities of a blockchain is the necessity to ensure integrity of the votes cast, a blockchain can be easily configured to not store the voter details for the votes. Further, Zero-Knowledge proofs can be implied to ensure voter identity and the correctness of a vote.

## **Decentralization:**

Another added advantage of blockchains is its accessibility. Any transaction, or in this case vote that gets cast can be made visible to the general public/participants.

This decentralized nature ensures that the final vote count and details of the vote differences between each candidate are visible to everyone to verify the results. However, a caution to be noted here is that the results shouldn't be visible during the elections as they could influence the remaining vote.

## **Un-Reusability:**

Another aspect of a legitimate voting system is that each participant should get an equal vote, and no one person should be allowed to cast more than one vote. As all the details of votes cast are stored in the blockchain, it should be easy to prevent a person from voting for a second time.

## IMPLEMENTATION OF ZERO-KNOWLEDGE PROOFS

While dealing with voting systems in blockchains, the most sensitive piece of information is the vote cast by a user. The Vote should be kept secret for the elections to be conducted in a free and fair manner.

In our application, we have implemented two functions that use the concepts of Zero-Knowledge Proofs in order to verify if:

- 1) The vote cast has is what is being sent for the transaction into the blockchain.
- 2) The Vote is a valid vote, and doesn't vote for more than one candidate.

- **verify\_transaction() method:**

It uses the concept of a zero-knowledge transfer of information between the sender and the receiver about the sensitive information (x).

Here, the function acts as a verifier and checks if the vote that is being passed into the blockchain is actually the vote that was cast by the user. The function here does not know what the value of the vote is, but is still able to verify if the intended value of vote input by the user is being passed on further to the blockchain.

```
def verify_transaction(p,g,y,s,b,r):  
    #y = pow(g,x,p) Sender's Public Value  
    #s = (r+b*x)%(p-1) Calculated by Sender  
    #b = int.from_bytes(random.randbytes(1),"big")  
    #r = random.randint(0,p-1)  
    h = pow(g,r,p)  
    if pow(g,s,p) == (h*pow(y,b,p))%p:  
        return True  
    else:  
        return False
```

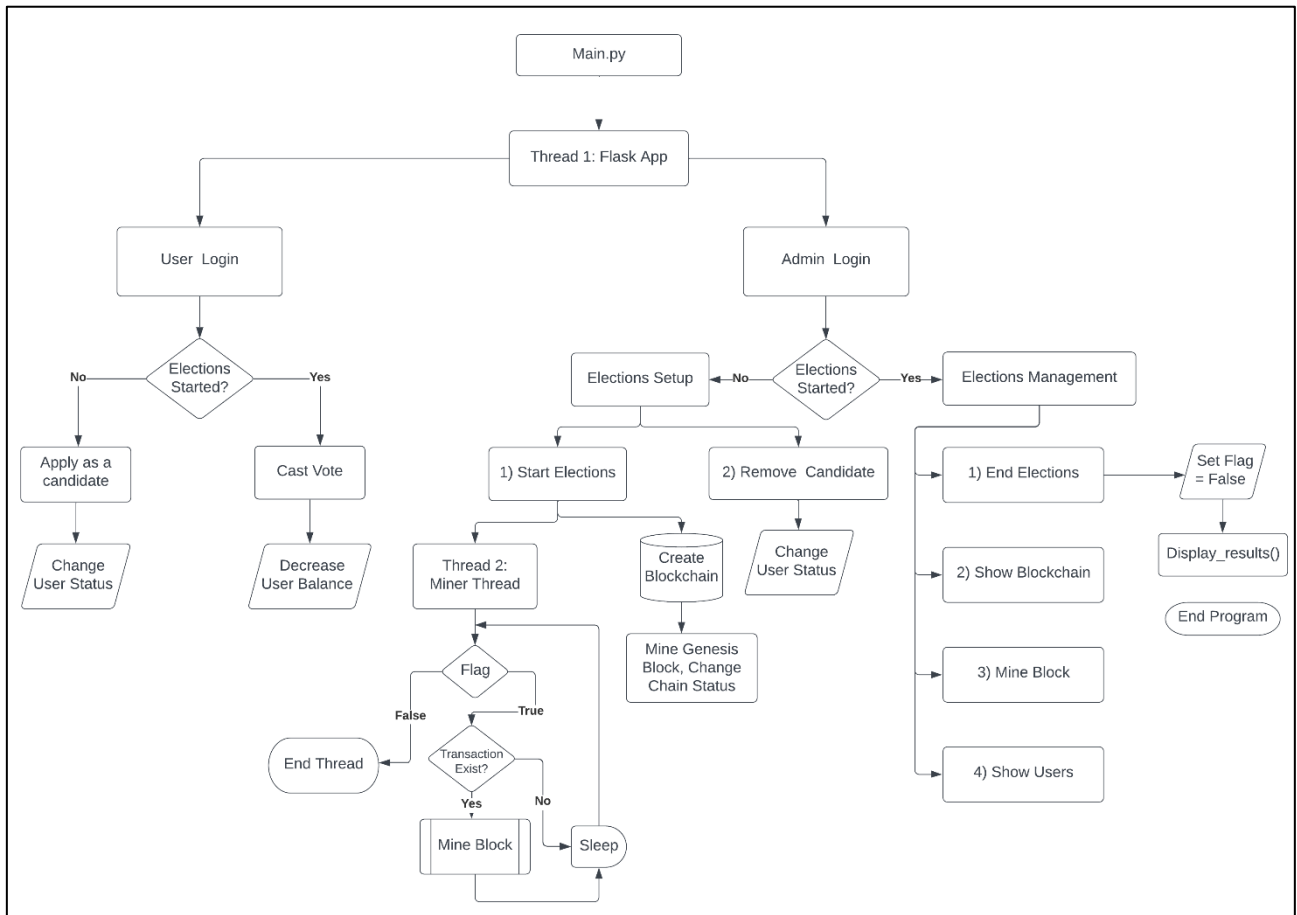
- **check\_vote() Method:**

Another implementation of a zero-knowledge proof is to check if the vote that has been cast is a valid vote.

The Votes are being cast in the form of a string of 0s and 1s. Hence, a simple check to see if the string sums up to a total of 1 can verify that the particular vote is only voting for one person and not more than one person. Also, in order to prevent a string that could be made up of negative numbers to make the total equal one, a check to see if the string is binary or not makes sure that the check is Sound, Complete as well as Zero-knowledge in nature.

```
def check_vote(self):  
    sum=0  
    if len(self.vote) != Vote.candidates:  
        return False  
    for i in range(Vote.candidates):  
        if int(self.vote[i]) not in [0,1]:  
            return False  
        else:  
            sum+=int(self.vote[i])  
    if sum!=1:  
        return False  
    return True
```

## KEY FUNCTIONALITITES OF THE CODE



The application flow is mainly running in two threads:

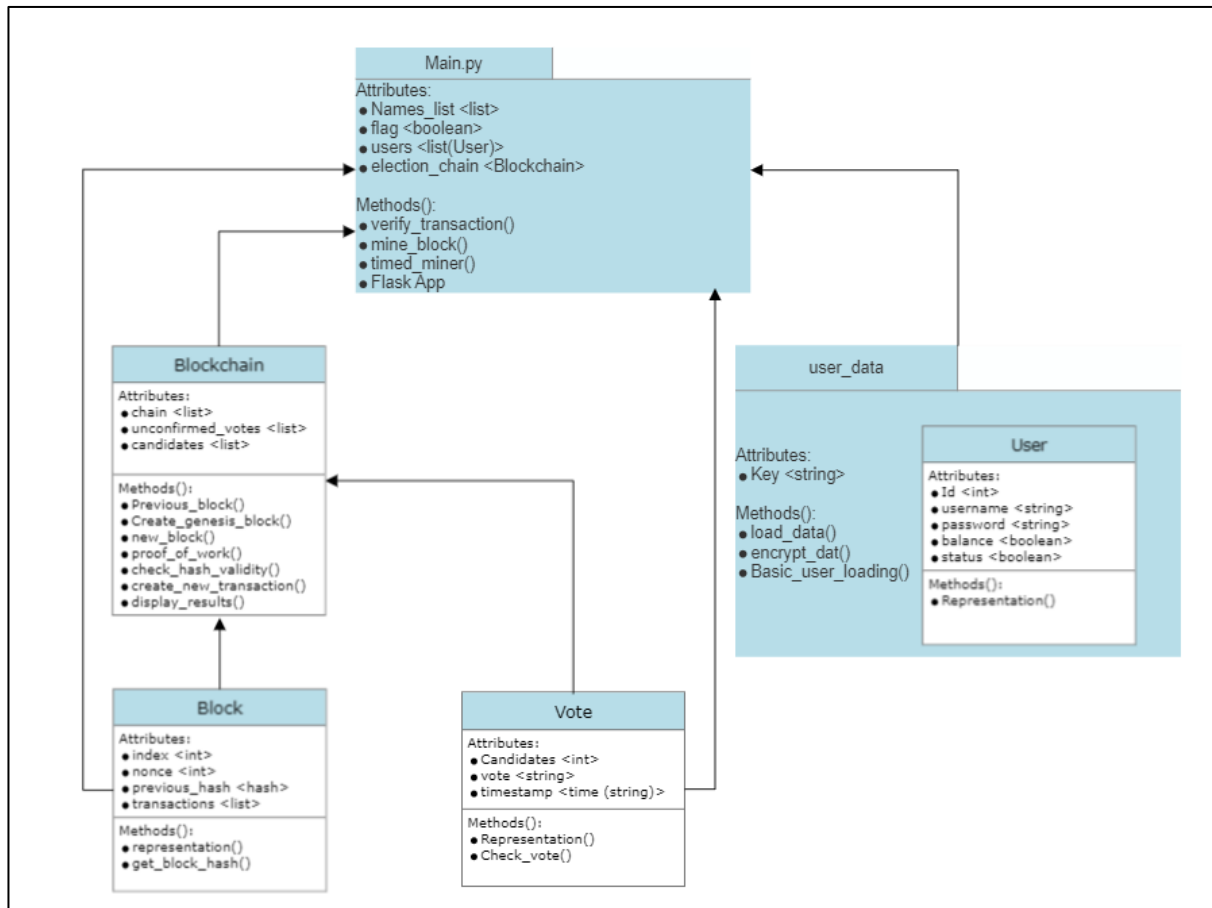
**Flask Thread:** This thread is responsible for the webapp to function properly and interact and implement the various necessities of an election while using a blockchain as a ledger.

### Primary Functions:

- **Admin\_home():** This function creates a blockchain and is responsible for the genesis block of the chain to be mined. It also passes on the names of the candidates to the Blockchain class to be written into the genesis block. A chain gets created only if there are 2 or more candidates that have registered for the elections.
- **Admin\_home2():** This function is responsible to end the elections when called, as well as displaying the results for the elections by calling the **display\_results()** function of the blockchain. It can also display the current status of the chain, mine a block and show users.
- **Mine\_block():** This function acquires the shared memory space and locks it to ensure the blockchain doesn't get manipulated while a block is being mined. The function then calls the **new\_block()** method of the blockchain to add a new block into the chain. In the background, the **proof\_of\_work()** method also gets called to create an acceptable block hash with the votes as contents of the block.



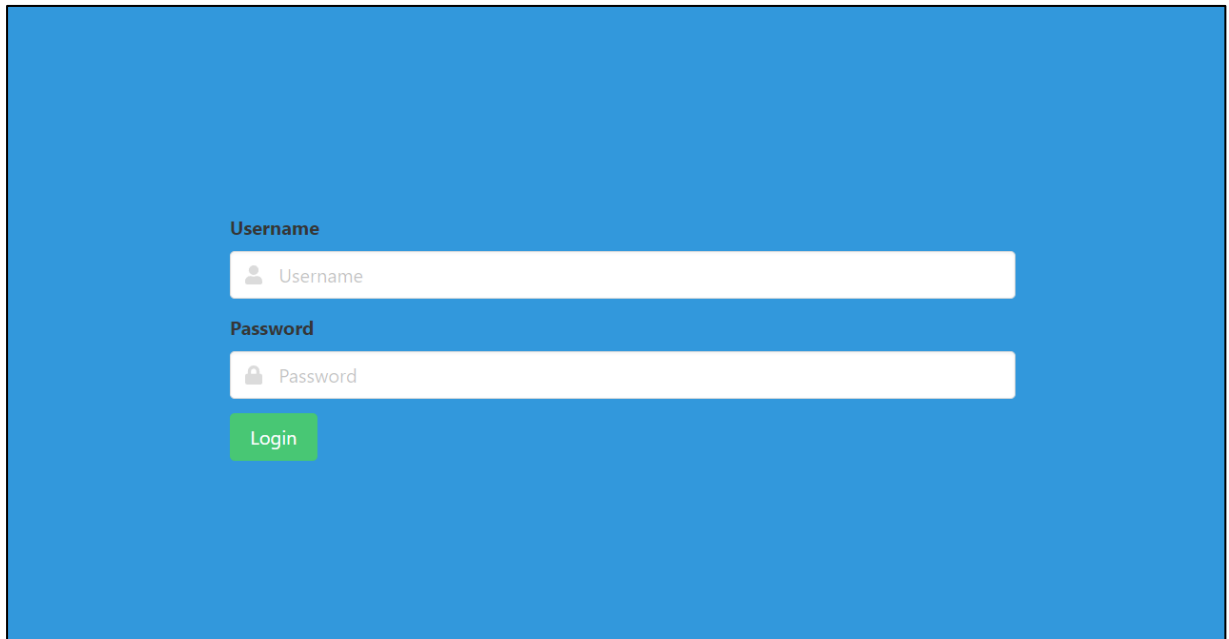
- **Timed\_miner():** This function is responsible to call the Mine\_block() function described above every 30 seconds if there are pending transactions in the blockchain.
- **Login():** Responsible to handle the authentication and creating a session as per the status of the user (or admin).
- **User\_home():** Enables a user to apply as a candidate if the voting has not yet started. In case the voting has started, the function then enables a User to cast their vote if they are eligible to do so. This function calls the ZKP functions, **check\_vote()** & **verify\_transaction()** in order to verify the validity of the vote cast.



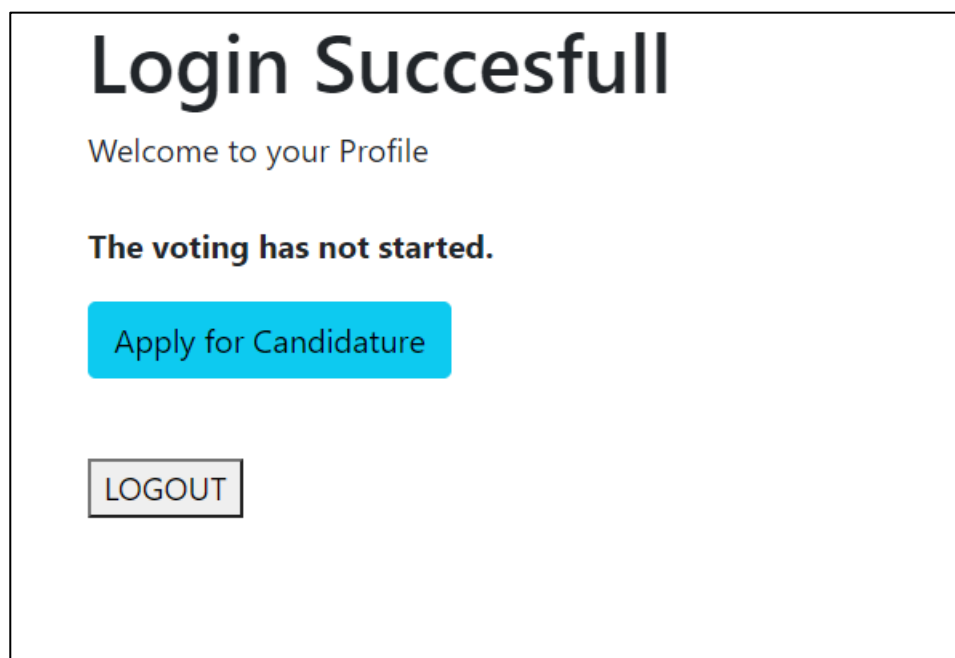
The above ULM Diagram describes the classes used and their relationships.

- The main.py package runs the flask app which utilises the features of the classes defined below.
- The **Blockchain class** contains an attribute “*chain*” which stores the **Block objects** created.
- The Blockchain class contains an attribute “*transactions*” which stores the Vote objects created.
- The **main.py** package creates instances of **Vote class** to verify votes by calling `check_vote()` method of the class.

## SCREENSHOTS OF WORKING APPLICATION

A screenshot of a login page with a solid blue background. In the center, there is a white login form. The form has two input fields: the top one is labeled 'Username' and contains a user icon and the placeholder text 'Username'; the bottom one is labeled 'Password' and contains a lock icon and the placeholder text 'Password'. Below these fields is a green button with the text 'Login' in white.

Initial Login Page for users and admin

A screenshot of a user's profile page after a successful login. The page has a white background. At the top, it says 'Login Succesfull' in a large, bold, black font. Below this, it says 'Welcome to your Profile' in a smaller, grey font. Further down, there is a bold black statement: 'The voting has not started.' Below this statement are two buttons: a bright blue button with the text 'Apply for Candidature' and a grey button with the text 'LOGOUT'.

User homepage before voting/elections start

# Login Succesfull

Welcome to your Profile

**You've registered as a candidate for the elections.**

## Election Candidates:

1) Naman

**Note:** Your voting string must be of length = Number of candidates

For example, if there are three candidates, your vote string must be like: **001** where the 1 denotes a vote for candidate 3.

User homepage after applying for candidature for elections

# Admin Login Succesfull

## Election Setup

### Election Candidates:

☐ Naman

☐ Akshay

I

Admin home page before elections start

# Admin Login Successful

Elections are Underway!

[End Elections](#)[Show Blockchain](#)[Mine Block](#)[View Users](#)[LOGOUT](#)

## Blockchain:

<Block index: 0 Nonce: 226 Transactions: ['Naman', 'Akshay'] Timestamp: 2022-04-25 12:42:43.635966 Previous\_hash: 628051bc2461053bdc67aba82077489ff70a69be4ecda5eedac0a1236bf3aad5>

Admin home page after elections start

# Congratulations!

Your vote has been succesfully registered.

[LOGOUT](#)

User home after casting vote

End Elections

Show Blockchain

Mine Block

View Users

LOGOUT

**Blockchain:**

<Block index: 0 Nonce: 81 Transactions: ['Naman', 'Akshay'] Timestamp: 2022-04-25 03:30:57.298145 Previous\_hash: 628051bc2461053bdc67aba82077489ff70a69be4ecda5eedac0a1236bf3aad5>

<Block index: 1 Nonce: 428 Transactions: [<Vote String: 10 Voting Time: 2022-04-25 03:33:07.331375>] Timestamp: 2022-04-25 03:33:27.303901 Previous\_hash: 001b760d02bbf3afe7411b64b16b983ace3236883a75d334f9679910c141c5ee>

<Block index: 2 Nonce: 19 Transactions: [<Vote String: 10 Voting Time: 2022-04-25 03:34:15.607714>] Timestamp: 2022-04-25 03:34:27.311783 Previous\_hash: 00c044aebcf843f784123f616e7be6bdc682a0986ee11e3e937132ed0ea06464>

<Block index: 3 Nonce: 287 Transactions: [<Vote String: 01 Voting Time: 2022-04-25 03:34:30.924167>, <Vote String: 10 Voting Time: 2022-04-25 03:34:56.454452>] Timestamp: 2022-04-25 03:34:57.304231 Previous\_hash: 00acc2ce16dedad154d193ed550211e65ad77efb8563739fd6ce02c22dd98435>

<Block index: 4 Nonce: 171 Transactions: [<Vote String: 10 Voting Time: 2022-04-25 03:35:07.100180>] Timestamp: 2022-04-25 03:35:27.300993 Previous\_hash: 0029156b3368eb01a3ea4686393fe094e0e9ad5c6746cbfa58f192d941cc8bd5>

<Block index: 5 Nonce: 15 Transactions: ['Election over at:2022-04-25 03:35:38.300783'] Timestamp: 2022-04-25 03:35:38.300783 Previous\_hash: 00834f3fa5b8436c89c54a00993980ca0b80b30d84c38e21a459c27f152e09d8>

**Elections Stopped! No more blocks will be mined**

**Results:**

Naman has recieved 4 votes.

Akshay has recieved 1 votes.

Admin home after Ending Elections

```

Miner Started in Background.
127.0.0.1 - - [25/Apr/2022 03:38:09] "
POST /admin_setup HTTP/1.1" 302 -
127.0.0.1 - - [25/Apr/2022 03:38:09] "GET /admin_stop HTTP/1.1" 200 -
127.0.0.1 - - [25/Apr/2022 03:38:09] "GET /static/bootstrap.min.css HTTP/1.1" 304 -
127.0.0.1 - - [25/Apr/2022 03:38:26] "GET / HTTP/1.1" 200 -
Block Mined
127.0.0.1 - - [25/Apr/2022 03:39:14] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [25/Apr/2022 03:39:16] "POST / HTTP/1.1" 302 -
127.0.0.1 - - [25/Apr/2022 03:39:16] "GET /admin_stop HTTP/1.1" 200 -
127.0.0.1 - - [25/Apr/2022 03:39:16] "GET /static/bootstrap.min.css HTTP/1.1" 304 -
127.0.0.1 - - [25/Apr/2022 03:39:18] "POST /admin_stop HTTP/1.1" 200 -
127.0.0.1 - - [25/Apr/2022 03:39:18] "GET /static/bootstrap.min.css HTTP/1.1" 304 -
Breaking due to flag variable.
Miner Stopped.

```

Background Miner Thread logs in the console