A PROJECT REPORT ON

DECENTRALIZED VOTING SYSTEM USING BLOCKCHAIN

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING
IN
COMPUTER ENGINEERING

SUBMITTED BY

ATHARVA JANGADA	**B150054220**
NIMISH DADLANI	**B150054261**
SANCHIT RAINA	**B150054448**
SOORAJ VS	**B150054479**

UNDER THE GUIDANCE OF
DR. A. R. BUCHADE
2021-2022

DEPARTMENT OF COMPUTER ENGINEERING
PUNE INSTITUTE OF COMPUTER TECHNOLOGY
DHANKAWADI, PUNE – 43

CERTIFICATE

This is to certify that the project report entitles
DECENTRALIZED VOTING SYSTEM USING BLOCKCHAIN

Submitted by

ATHARVA JANGADA	**B150054220**
NIMISH DADLANI	**B150054261**
SANCHIT RAINA	**B150054448**
SOORAJ VS	**B150054479**

are the bonafide students of this institute and the work has been carried out by them under the supervision of **Dr. A. R. Buchade** and it is approved for the fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering in Computer Engineering**.

Dr. A. R. Buchade
Guide
Dept. of Computer Engineering

Dr. G. V. Kale
Head,
Dept. of Computer Engineering

Dr. R. Sreemathy
Principal,
Pune Institute of Computer Technology

Place: Pune
Date:

ACKNOWLEDGEMENT

It gives us great pleasure in presenting the project report on 'De-Centralized Voting System using Blockchain'.

We would like to take this opportunity to thank **Dr. A. R. Buchade** for giving us all the help and guidance we needed. We are really grateful to him for his constant support, encouragement and motivation. His valuable suggestions were very helpful.

We are also grateful to **Dr. G. V. Kale**, Head, Department of Computer Engineering, PICT for her indispensable support and suggestions.

Atharva Jangada
Nimish Dadlani
Sanchit Raina
Sooraj VS

ABSTRACT

The revolutionary concept of blockchain, the technology behind the popular cryptocurrency Bitcoin and its successors, has ushered in a new era in the Internet and online services. While most people focus only on cryptocurrencies when they think about Blockchain, in fact many administrative activities and day-to-day services are managed offline and/or privately are now being securely transferred to the Internet as online services. Electronic Voting is one of the trending and important aspect of online services.

Traditional voting systems in our country include the use of ballots at polling stations or EVM's (Electronic Voting Machine). The ballot system is prone to tampering, whereas EVM's being a centralised system, are prone to hacking. We aim to offer a more secure, transparent, safer, cheaper and easier to use alternative to the above methods of voting, by making use of the revolutionary Blockchain technology.

Smart contracts are meaningful pieces of code that are incorporated into the Blockchain and implemented on schedule at each stage of the Blockchain update. Ethereum and its network is one of the most convenient due to its stability, wide usability and smart contract logic. In this work, we will be implementing and testing a prototype of an E-Voting application using Ethereum Wallet and Solidity language as a smart contract for the Ethereum network.

Keywords

Blockchain, Electronic voting, Decentralization, Transparency, Anonymity, Ethereum, EVM, Smart Contracts, Solidity

Contents

List of Abbreviations

Abbreviation	Full form
EVM	Ethereum Virtual Machine
PoW	Proof of Work
PoS	Proof of Stake
RPC	Remote Procedure Calls

List of Tables

List of Figures

CHAPTER 1

INTRODUCTION

1.1 Overview

E-voting systems based on blockchain have gained momentum and a line of research has recently been developed. Researchers consider these systems to be transparent, verifiable, scalable and open to the public.[1] Nevertheless, the lack of actual system structures makes it difficult to determine whether these systems actually have the described characteristics. Many influential studies have also proposed the use of a smart contract to create decentralized autonomous voting systems, which can eliminate as many human factors as possible.[2]

Blockchain provides all the characteristics that are needed for an e-voting system, which is arguably the most crucial part of a democratic society. A blockchain neither allows past events to be changed nor does it enable current events to be hacked.[5] In addition, the system does not tolerate changes. Every machine/node that has been granted access shows the same results, and every vote can be traced to its source without exposing the voter's identity.[3]

Furthermore, to guarantee consistency, when another block is created and appended to the previous block, a specific process is required to solve a computationally expensive puzzle known as Proof-of-Work(PoW). However, PoW is energy-inefficient and influences the throughput and transaction latency, which in turn impacts the scalability and performance.[4] Accordingly, Proof of Stake(PoS) has proved to be a strong competitor to overcome the problems associated with PoW. Expressly, a set of validators take turns to propose and vote on the next block in the PoS consensus mechanism, and the weight of the vote relies on the value of the staked tokens. Similar to PoW, PoS offers the network a convincing financial boost to operate effectively and reduce the risk of conspiracy or attack. However, unlike PoW, it is not computationally complex; therefore, it overcomes the problem of energy-inefficiency.[6] More recently, the hybrid consensus model has attracted considerable interest in an attempt to overcome the challenges faced by the PoW consensus method.

1.2 Motivation

The beating heart in democracy are elections. The approval of the people is the source of a government authority. Elections serve two purposes: to guarantee the authority of those in positions of power, and to empower citizens to remove unpopular rulers. Elections must be inclusive and participatory in order for this to happen. Regardless of the political sense, disenfranchisement and poor turnout threaten the legitimacy of election outcomes.

At the same time holding elections regularly is a costly affair. Holding the Indian Lok Sabha Elections (2019) cost the Government Rs. 50,000 crores ($7 Billion), which is a 40% increase from the 2014 LS elections. Moreover, there is always a complaint from the losing candidates about EVM hacks, voter manipulating, and security lapses. We have to solely trust the people who are involved with the process where lapses on even a small scale can jeopardize the essence of the Elections.

1.3 Problem Definition and Scope

1.3.1 Problem Definition

Designing a decentralised voting system to address the issue of maintaining anonymity of voters on a public blockchain network and achieving transparency, safety and security as compared to the existing models.

1.3.2 Scope

This report suggests methods to reduce latency and cost per vote by utilizing Proof-of-Stake(PoS) EVM compatible networks for the deployment of Election smart contract. It also addresses the issue of managing voters eligibility, uniqueness and anonymity using a hybrid architecture of central servers and blockchain network.

This study focuses on designing a smart contract to provide a trustworthy public bulletin board and a secure computing environment to keep track of election events and ensure the accuracy of the ballot outcome. Along with developing a Web3 application for interfacing with the deployed contracts.

All the experiments are carried out on 'Ethereum Rinkeby Test Network' with enpoints provided by Infura for testing and recording gas cost corresponding to each transaction. The application can be readily configured to operate on any EVM compatible Mainnets by modifying network RPC endpoint*(e.g. Polygon or Avalanche C-chain for achieving extremely low gas prices).*

CHAPTER 2

LITERATURE SURVEY

2.1 Literature Survey

Decentralized E-voting system based on Smart Contract by using Blockchain Technology

Ref: [1] Today, the use of the Internet is increasing. Electronic voting systems have been used in different countries to reduce the costs and time previously consumed by using traditional voting. If a voter wants to access the electronic voting system through a web application, there are requirements such as a web browser and a server. Voters use a web browser to access the central database. Using a central database for your voting system poses security issues: The results of data changes and adjustments by third parties in the network due to the use of the central database system are not displayed in real-time. However, this paper aims to provide a system that raises a high degree of security by using blockchain. Blockchain provides a decentralized model that makes your network more reliable, secure, flexible, and can support real-time services.

Decentralized Voting Platform Based on Ethereum Blockchain

Ref: [2] In a centralized environment, election results were always questionable and perceived differently by voters. Most existing electronic voting systems are based on centralized servers where voters must trust the organization for consistent results. This paper proposes a new approach for a decentralized and reliable voting platform based on blockchain technology to solve the trust problem. The main functions of this system include ensuring data integrity and transparency and conducting one vote for each mobile number for each survey that guarantees data protection. To achieve this, the Ethereum Virtual Machine (EVM) is used as the blockchain runtime environment. In this environment, the organizers of each voting event provide transparent, consistent, and deterministic smart contracts to enforce voting rules. Users are authenticated using their mobile number without the need for a third-party server. The results show that the system is viable and can provide a step towards an ideal environment for such an experience.

Blockchain based e-voting recording system design

Ref: [3] Today's increase in digital technology has made life easier for many. In contrast to the electoral system, there are many traditional uses of paper for its implementation. The security and transparency aspects are a threat from being still widely selected in traditional "offline" systems. Parliamentary elections still use a centralized system, and there are organizations that manage it. Some of the problems that can occur with traditional electoral systems are databases are very likely to be tampered within organizations that have complete control over the database and system. Blockchain technology is one of the solutions because it includes distributed systems and the entire database is owned by many users. The blockchain itself was used in the Bitcoin system known as the decentralized banking system. Introducing blockchain to database distribution in electronic voting systems can reduce one of the causes of database operations fraud. This survey describes recording voting results from each voting location using a blockchain algorithm. In contrast to Bitcoin with Proof of Work, this work proposed a method based on the predefined activation of the system of each node in the blockchain.

Votereum: An Ethereum-Based E-Voting System

Ref: [4] Blockchain, the foundation technology of the first cryptocurrency Bitcoin, has received a great deal of attention around the world in recent years. Its remarkable distributed ledger, reliable systems, and immutable properties enable disruptive innovation in the electronic payments industry as well as potential solutions in other areas that require trust-building. Electronic voting schemes (e-voting) are use cases where all attributes of the blockchain can provide an open, fair, and universally verifiable mechanism for the voting process. This white paper reviews the requirements and then proposes Votereum, an electronic voting system that uses blockchain technology. The proposed system is supported by the Ethereum platform, which includes one server that manages the entire system and another server that handles all blockchain-related requests. The implementation is also posted on the Rinkeby test network to assess its feasibility and discuss some of the security concerns mentioned in the conclusions of this document.

Blockchain-Based E-Voting System

Ref: [5] Building an electronic voting system that meets the legal requirements of the legislature has long been a challenge. Distributed ledger technology is an exciting technological advance in the world of information technology. Blockchain technology offers an endless range of applications that will benefit from the sharing economy. This paper aims to evaluate the application of blockchain as a service for implementing a distributed electronic voting system. This white paper clarifies the structural requirements of electronic voting systems and identifies legal and technical restrictions on using blockchain as a service to implement such systems. This white paper begins by assessing some of the popular blockchain frameworks that blockchain offers as a service. Next, we propose a new blockchain-based electronic voting system that addresses the limitations we discover. In a broader sense, this paper describes the process of implementing a case study, a blockchain-based application that conducts elections improves security, and reduces the cost of conducting state-wide elections and evaluates the potential of ledger technology.

The Following table shows the literature survey by comparing techniques propose in various references:

Table 2.1: Literature survey

Paper	**Overview**	**Technology used**	**Feature**
Decentralized E-voting system based on Smart Contract by using Blockchain Technology [1]	This paper proposes a new approach for a decentralized and reliable voting platform based on blockchain technology to solve the trust problem.	Smart Contracts, 2-Factor Authentication	Provides a decentralized model that makes your network more reliable, secure, flexible, and can support real-time voting.
Blockchain Based E-Voting Recording System Design [3]	This survey describes recording voting results from each voting location. This work also proposed a method based on the predefined activation of the system of each node in the blockchain.	AES Encryption	Maintains a database of Voters using Advanced Encryption Standard(AES) encryption Algorithm
Votereum: An Ethereum-Based E-Voting System [4]	This paper reviews the requirements and how proposed system is supported one server that manages the entire system and another server that handles all blockchain-related requests.	Ethereum, Smart Contracts	Uses Ethereum Blockchain and the power of Smart Contracts to implement EthVote, a decentralized framework, that is easily scalable.
E-Voting using Blockchain [6]	This paper has extracted and reviewed multiple research papers that advise the blockchain framework for voting and indicates that there are different solutions than the traditional e-voting.	Firebase, React, LevelDB	An introduction to the basic structure and features of the blockchainin relation to e-voting alongwith a conceptual description of the desired blockchain-based e-voting application is provided

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Assumptions & Dependencies

3.1.1 Assumptions

1. Users have access to Javascript-enabled web browsers that support web3 extension 'MetaMask' (e.g. Google Chrome, Firefox).

2. The applications' ability to determine eligible voters assumes the election organizing committee will process the required details for authenticating participation in the voting event as per their needs and a list of eligible 'unique voter ids' are provided as input.

3. Users will comply to work with a dedicated/unused wallet, and never use this identity for performing transactions outside the application environment thus maintaining their anonymity while casting votes.

3.1.2 Dependencies

1. The development and testing of smart contracts require Rinkeby Test Network, Remix IDE, solidity compiler v0.8.7 accompanied with Javascript library solc v0.8.7

2. The frontend and backend for this application are written in Javascript and depend on an existing blockchain network, RPC endpoint, web3js library for interfacing with the network, and Metamask extension for injecting web3 window within the browser.

3.2 Functional Requirements

1. **Registration:** Users should have an account with a unique voter id for creating voting events, registering in elections, and casting votes.
2. **Election Deployment:** The application should allow organizers to create election events and deploy corresponding smart contracts for storing participants and their votes.
3. **User-Interface:** The system shall provide an easy-to-use user-interface. Also, it shall not disadvantage any candidate while displaying the choices.
4. **Transparency:** Voters should be able to possess a general knowledge and understanding of the voting process
5. **Accuracy:** The system shall record and count all the votes and shall do so correctly.
6. **Eligibility:** Only authorized voters, who are registered, should be able to vote.
7. **Uniqueness:** No voter should be able to vote more than once.
8. **Auditability:** It should be possible to verify that all votes have been correctly accounted for in the final election tally, and there should be reliable and demonstrably authentic election records, in terms of physical, permanent audit trail (which should not reveal the user's identity in any manner).
9. **Voter Confirmation:** The voter shall be able to confirm clearly how his vote is being cast, and shall be given a chance to modify his vote before he commits it.
10. **Immutability:** Once a vote is committed it must be unalterable for every user class including the system admins and guest users
11. **Over/Under Voting:** The voter shall be prevented from choosing more than one candidate/answer. The voter may receive a warning of not voting, but the system must not prevent under-voting.

3.3 External Interface Requirements

3.3.1 Software Interfaces

- The client-side software interacts with the user directly through an easy-to-use GUI, allowing the user to register on the platform, participate in elections, and organizers to create, manage election events. Thus, the UI itself is divided into two interfaces for these two user classes.
- Similarly, server-side application is also divided for election organizers, voters and will also include a full node to interface the blockchain network. This will be implemented using NodeJs and Express library which interacts with the UI, central databases, and blockchain.
- For storing user login, election details/status, and votes a combination of central database MongoDB, distributed blockchain ledger will be used.

3.3.2 Communication Interfaces

- The application will use a simple HTTP based interface between the client and server side. All server endpoints will be exposed as secure, authenticated REST endpoints, through which the front end application can communicate and access central express server.
- JSON-RPC endpoints will be used to communicate with the Rinkeby Test Network during the development phase, which will be later replaced with the RPC endpoint of a PoS Mainnet.

3.4 Non-Functional Requirements

3.4.1 Performance Requirements

- **Low Latency:** The application should commit the casted votes onto the smart contract instantly, with an acceptable latency of not more than 30 seconds.
- **Convenience:** The system shall allow the voters to cast their votes in one session, and should not require many special skills or intimidate the voter (to ensure Equality of Access to Voters).
- **Cost-effectiveness:** Election systems should be affordable and efficient, averaging less than 1$ per vote.

3.4.2 Security Requirements

- **Non-coercibility and No Vote-selling:** Voters should not be able to prove to others how they voted (which would facilitate vote selling or coercion).
- **System Disclosability:** The core of the system, especially the vote-casting equipment, shall be opensource, so that it can allow external inspection and auditing.
- Unauthorized access to data must not be allowed under any circumstances.
- Complete end-to-end encryption, in transit, as well as at rest is crucial.

3.4.3 Software Quality Attributes

The following are some key quality characteristics that will be important to objectively judge the end result and feasibility of the product:

- Extensibility
- Reliability
- Accuracy
- Robustness
- Maintainability
- Intuitive GUI
- Availability

3.5 System Requirements

3.5.1 Database Requirements

- To save the user records, we have chosen NoSQL Database.
- For performing the secret voting transaction, we have used Blockchain and smart contracts.

3.5.2 Software Requirements

1. Linux/Windows OS
2. Solidity
3. Web3js
4. ReactJs
5. Web Browser (preferably Google Chrome or Firefox)
6. VS Code (IDE)

3.5.3 Hardware Requirements

Sr. No.	Parameter	Minimum Requirements	Justification
1	CPU Speed	2 GHz	Support the operations of the UI and APIs
2	RAM	8GB	To speed up the execution of the blockchain transactions.
3	Processor	i5	For smooth operation of the website.

3.6 Analysis Models: SDLC Model to be applied

SDLC is a process which is followed for any software development project, within an organization. This project makes use of the Agile model. Agile model refers to a development process which is a combination of incremental and iterative models. This type of software development model is basically used for the project which has small incremental builds.

1. **Brainstorming on the Problem Statement**:
   In this phase, we understood and discussed the problem statement with our project guide. We studied the basics of blockchain and thoroughly analyzed our problem statement and it's scope. Requirements were revisited every month to ensure rechecking.

2. **Requirement Analysis**:
   During this phase, we gathered all the requirements of our system and decided to explore various tools, frameworks, softwares, API's and cloud providers we can use in our project. During this phase, the System Requirement Specification(SRS) was prepared as per the given format consisting of all the different requirements such as system requirement, functional requirements, system requirements(both H/W & S/W), system requirements,etc.

3. **Design**:
   In this phase, the technical details of the project were decided. We worked on the system design of our application and decided how various components and modules of the application would interact with each other. We designed the structure of the smart contracts which would be used to record the elections.
   Thus, the overall design of the system was done in this stage, which included Use Cases, class design, data flow design along with the system architecture.

4. **Implementation**:
   After the design stage, we started with the implementation stage, where the actual coding of the application was done. In this phase, coding of Decentralized Voting System was carried out by using the technology stack discussed in the previous phase.
   This phase consisted of coding of Smart Contract(using Solidity), Client Side application, and the Server Side application. First, the smart contracts were coded according to the requirements. Then, implementation of the Client and Server side application started.

5. **Testing**:
   This phase was carried out after coding out the respective modules. Unit tests were written to test the solidity smart contracts. Client and server of the application were tested manually individually as well as after their integration. Edge cases were also highlighted out and handled successfully.

6. **Deployment**:
   In the deployment phase, we made sure that the environment was setup correctly and we deployed the application modules to their respective environments.

   - Deployment of Client Side Reactjs App (UI) - GitHub Pages
   - Deployment of Server Side NodeJs App - Vercel
   - Deployment of Smart Contracts - Polygon(Matic) Testnet

   Once the application was deployed, any additional requirements or bugs experienced at run-time were also fixed.

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

The high level system design shows how the different computational components, APIs, RPCs and clients will be laid out to interact with blockchain.

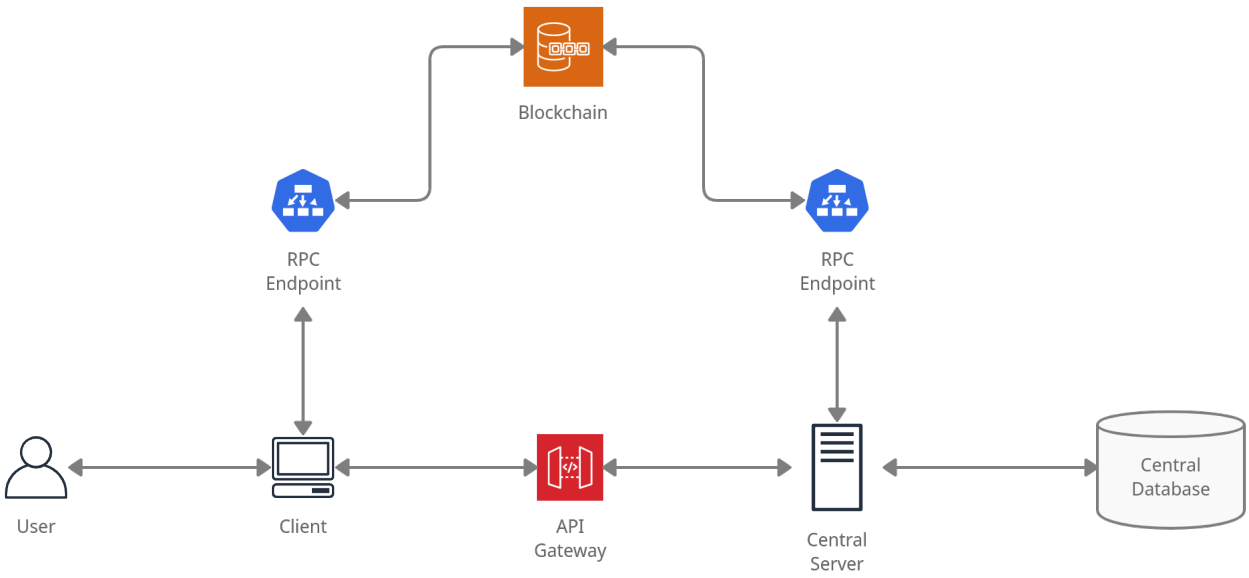


Figure 4.1: System Architecture Diagram

The end-user interacts with the blockchain via the RPC endpoints through the client application and the central database through the central server. The result after the voting phase ends are updated on the central database via the RPC endpoint through the central server.

4.2 Data Flow Diagram

The data flow diagram illustrates the flow of data throughout the life time of the application, for the three major use cases *(i.e. deploying election, voting and Result announcement).*

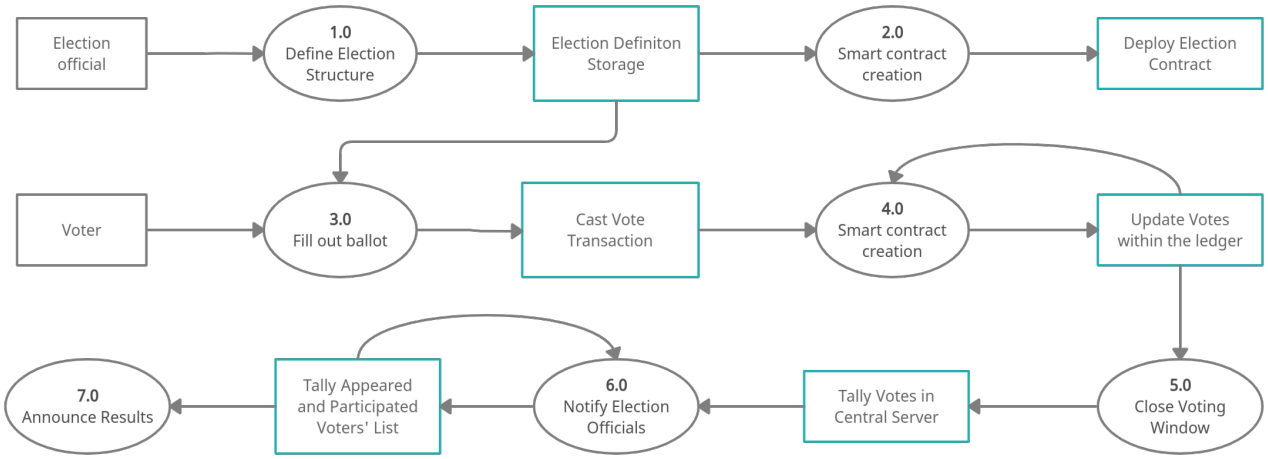


Figure 4.2: Dataflow Diagram for Major use cases

It shows the flow of data through the system. It also provides information about the outputs and inputs of each entity and the process itself. The detailed process is as follows:

a. The election official defines the election structure and creates & defines the smart contract for the said election.

b. The voter fills the ballot for voting and castes the vote which updates the votes within the ledger.

c. After the voting phase ends, the result is stored in the central database and is displayed on the results page of the said election.

4.3 Class Diagram

The components of the application can be divided into four main components and two contracts as illustrated below:

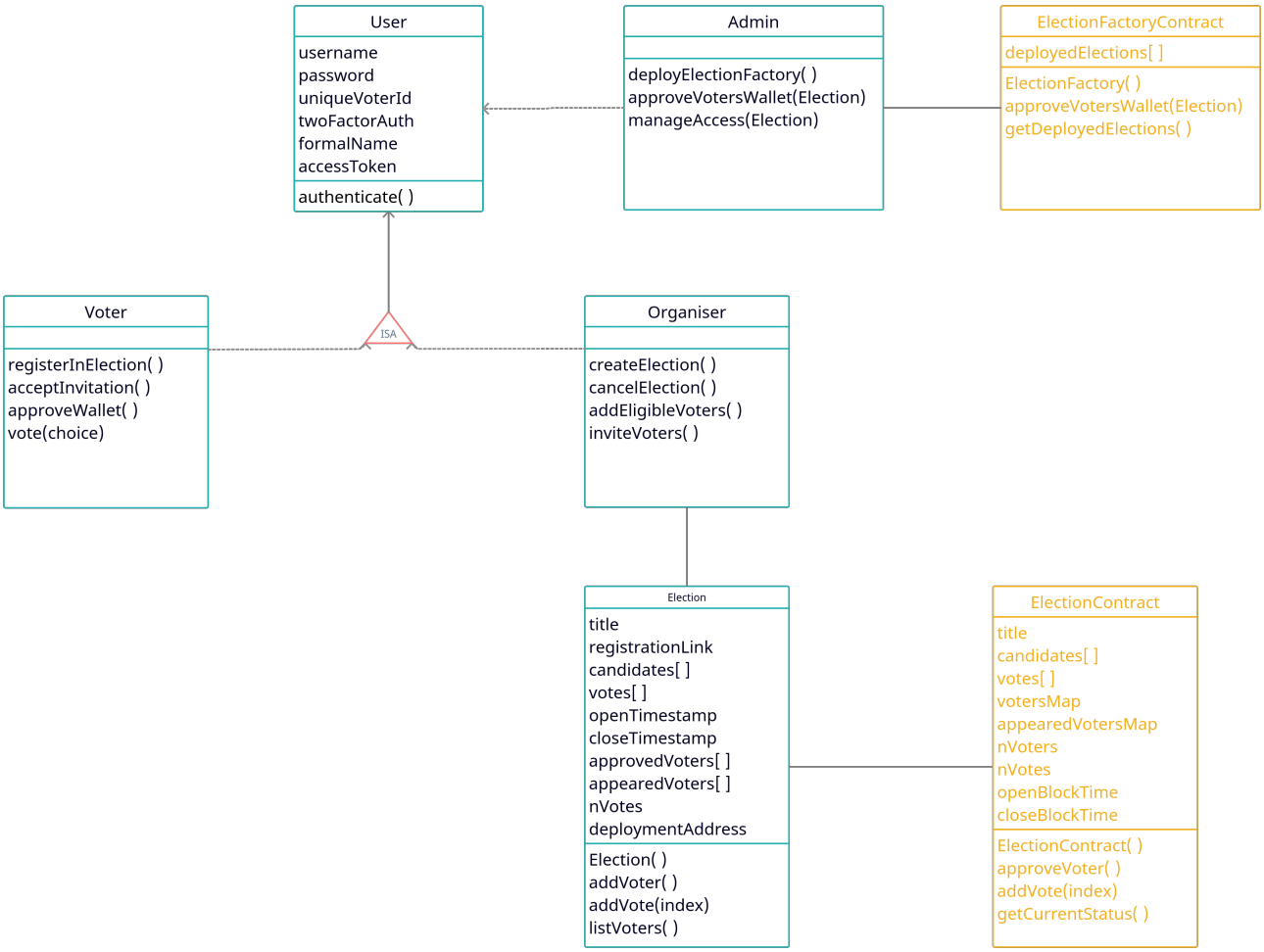


Figure 4.3: Class Diagram including smart contracts (in yellow)

This diagram describes the attributes and operations of class and the constraints imposed on the system. There are two subtypes of users(IS-A relationship) using this application: Organisers, and Voters. The organizers are allowed to create an election and voters are allowed to vote in an election if he/she is added to the list of approved voters for the said election.

Following are the enlisted use case for each type of user:

4.4 Use Case Diagram

The major use cases of the application are illustrated here, with users, server and blockchain as the actors interacting with different modules.

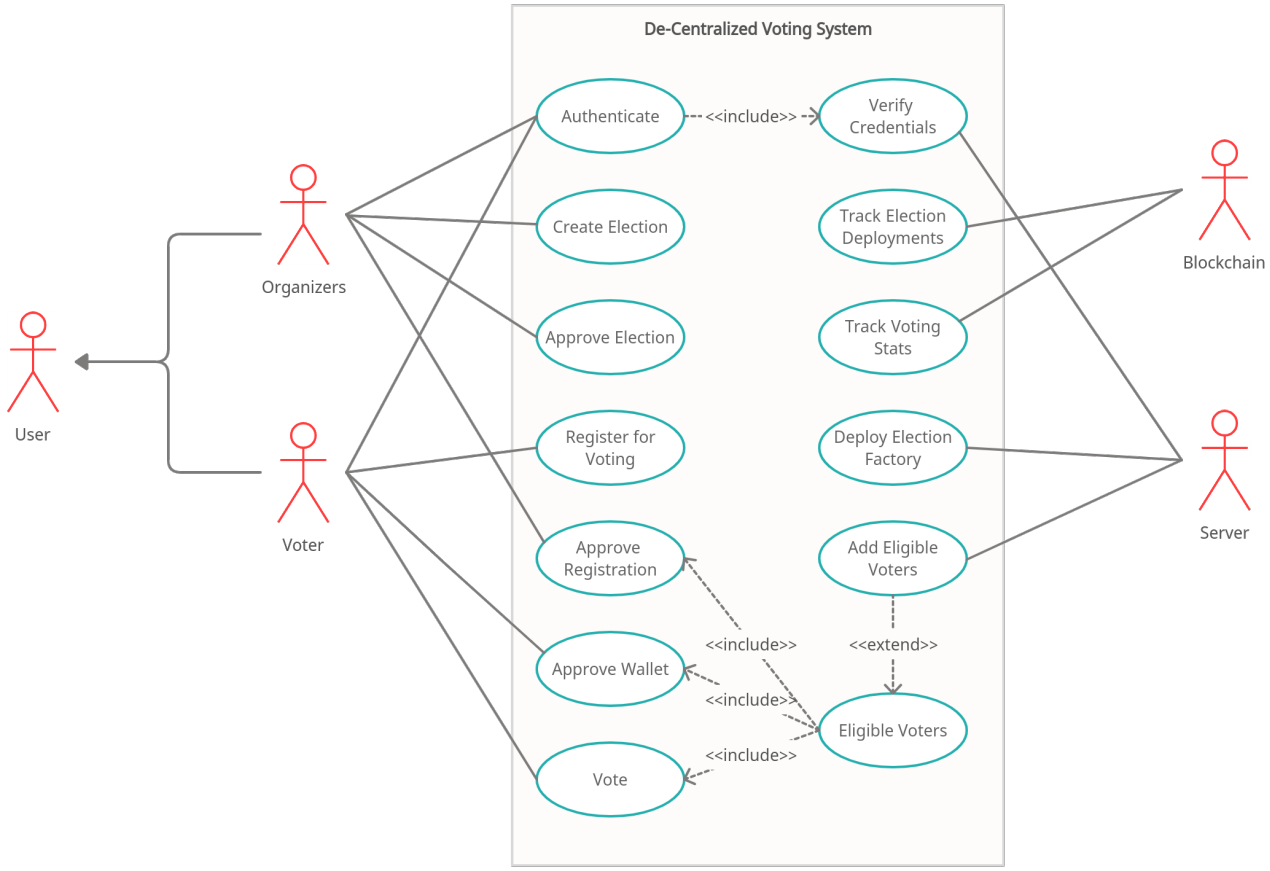


Figure 4.4: Use Case Diagram

There are two subtypes of users using this application, Organisers, and Voters, following are the enlisted use case for each type of user:

1. Organiser: Create, edit and delete Election, Approve voters for the election
2. Voter: Apply for voting, Cast vote
3. Common use cases: Authentication, Explore elections, View results

4.5 Sequence Diagram

The following diagram depicts the series of actions occurring for creating and voting in an election (top to down) and a subsequence of steps happening on different layers of the application viz. frontend, servers, and blockchain.

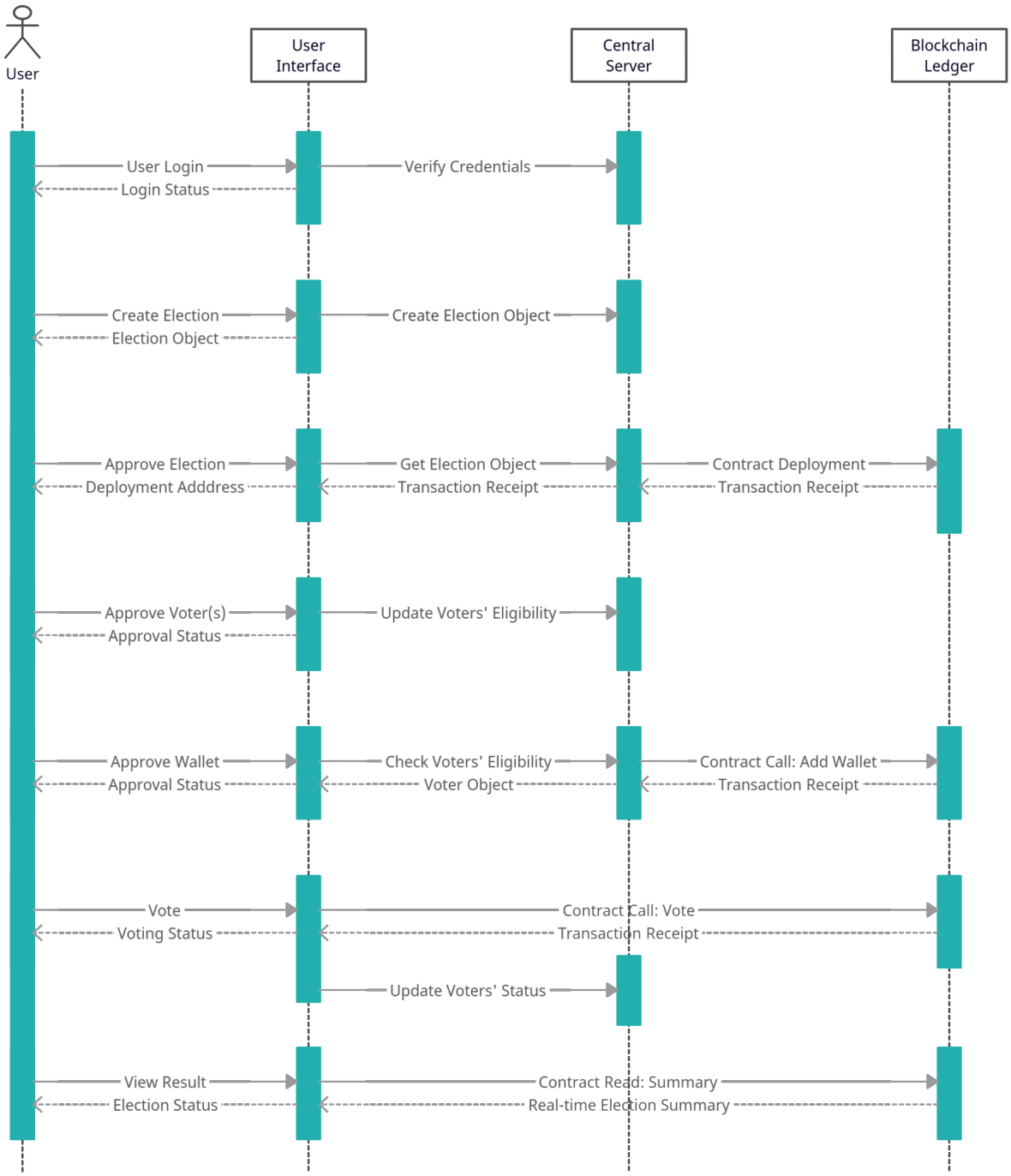


Figure 4.5: Sequence Diagram with Major Functionalities

E.g. a sequence of actions for the voting step of an election:

1. Vote for a candidate from the web application
2. A transaction occurs at the election contract
3. The transaction is confirmed and returns a receipt
4. Votes are updated on the server as per the transaction receipt
5. Voting status is conveyed to the end-user

CHAPTER 5

PROJECT PLAN

5.1 Project Estimate

5.1.1 Reconciled Estimates

- Cost Estimate: The software used for the application will incur no cost as it is all released under free open source licenses.

5.1.2 Human Resources

- Number of people: 4
- Skills: Blockchain, Solidity, Web3js, ReactJS, NodeJs.
- Client: In our project, the users can belong to any age group, or section
- Stakeholders: Stakeholders are often members of the project team. Considering the project, stakeholders will be the people who will be using the application, i.e, the clients along with project team.

5.1.3 Development Resources

- Hardware: i5 Processor, Minimum 8GB of RAM, At least 2GB of free disk space
- Software: Linux/Windows OS, Solidity, Web3js, ReactJs, Web Browser (preferably Google Chrome or Firefox)
- Database: To save the user records, we have chosen NoSQL Database. For performing the secret voting transaction, we have used Blockchain and smart contracts.

5.2 Risk Management

Risks are a part of developing any kind of applications. While risk is normally affiliated with negative connotations, a mature understanding of risk defines it as an essential element which helps the corporates to seize new opportunities and conquer new frontiers i.e. if the risk is handled properly.
Hence, risk management becomes a very important aspect of developing an application.

5.2.1 Risk Identification & Analysis

The risk analysis is performed using the following guidelines:

Risk Probability

a.	High Probability	$75\% \leq x \leq 100\%$
b.	Medium High Probability	$50\% \leq x \leq 75\%$
c.	Medium-Low Probability	$25\% \leq x \leq 50\%$
d.	Low Probability	$0\% \leq x \leq 25\%$

Table 5.1: Levels of Risk Probability

Risk Impact

a.	Very High	Catastrophic
b.	High	Critical
c.	Medium	Moderate
d.	Low	Marginal

Table 5.2: Severity of Risk Impact

5.2.2 Risk Identification

The risk identification can be achieved using the following risk items:

- **Process definition**: This type of risk is associated with the degree to which the software process has been defined i.e whether the process is efficient or not. In our case, it is related to development of our decentralized voting system.
- **Product size**: This type of risk is associated with the overall software and product size of the system. This risk is also associated with the scalability of our software i.e. larger the size of software, greater is the risk involved in it.
- **Development environment**: This type of risk is associated with the overall environment of development. It depends on quality of the tools being used during the development of the software.
- **Technology to be built**: This is the most crucial type of risk and is associated with the overall complexity of software we are developing. Complex applications lead to more risk and hence considering this risk is crucial. Also, latest technologies used during the software development are of concern while taking this risk into consideration.

5.2.3 Risk Mitigation

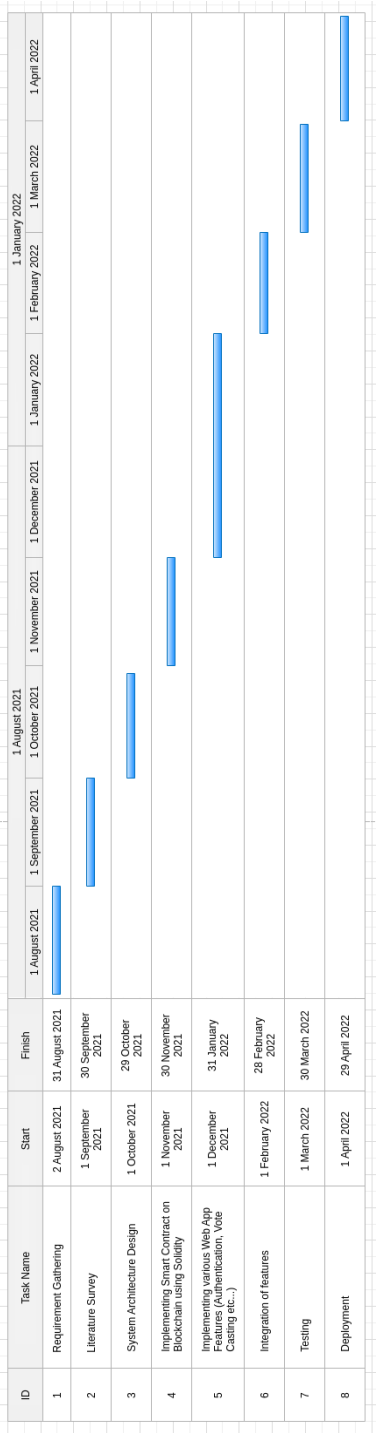
Risk avoidance is always the best strategy if the team adopts a proactive approach to mitigate it. Risk mitigation helps in avoiding the risks and thereby reducing the losses. The steps which can be taken are :

- Try to reduce the causes which are under control of the developers of the software.
- Follow up with the current staff to determine the cause for turnover.
- Effective Communication between the team members, so that the development is spread across the entire team.
- Define proper standards which will need to be followed by every member, while development.
- Have a backup version of the application for every release, in-case the new release crashes.

5.3 Project Schedule

5.3.1 Time Estimates

- Finalizing the Problem Statement: August 2021
- Designing the System Architecture: September 2021
- Implementation of the system: November 2021 - February 2022
- Testing: March 2022
- Deployment: April 2022



5.4 Team Organization

The manner in which a team is organized and the mechanisms for reporting have been noted. Updates regarding the improvement of project have been given to the project guide. A meet was held with the project guide regarding the updates, every month.

5.4.1 Team Structure

The team structure for the project has been identified. Roles are defined as follows -

- Atharva Jangada: Smart Contract, Client Side UI & Documentation
- Nimish Dadlani: Client Side UI, Smart Contract & Documentation
- Sanchit Raina: Smart Contract, Server Side application & Documentation
- Sooraj VS: Smart Contract, Server Side application & Documentation

5.4.2 Management reporting and Communication

- Team was always in contact with the project guide.
- Team members also collaborated in-person and worked together to improve the efficiency of the code.
- Chat groups, emails, online meetings(MS-Teams) & in-person(physically) meets have been used for communication with the Project Guide.

CHAPTER 6

PROJECT IMPLEMENTATION

6.1 Overview of Project Modules

The Project is divided into four main modules, enlisted below: -

1. Smart Contracts
2. Central Server
3. Application Frontend
4. Sample Organisation Server

6.2 Tools and Technologies Used

1. Smart Contracts
    - Solidity
    - Web3js for deployment
    - jest library for testing
2. Central Server
    - Nodejs, Express
    - MongoDB Atlas as Database
    - Web3js for interfacing with smart contracts
    - Vercel as hosting platform
3. Frontend
    - Reactjs
    - Web3js for interfacing with smart contracts
    - axios for api calls
    - Github pages for hosting
4. Sample Organisation Server
    - Django
    - SQLite as Database
    - RESTful APIs for intercating with Central Server
    - Deployed on Azure VM (Ubuntu 20.04)

CHAPTER 7

SOFTWARE TESTING

7.1 Types of Testing

- Alpha Testing : It is the most common type of testing used in the Software industry. The objective of this testing is to identify all possible issues or defects before releasing it into the market or to the user. It is generally carried out at the end of the software development phase but before the Beta Testing phase.
- Accepting Testing : An acceptance test is performed by the client and verifies whether the end to end the flow of the system is as per the needs of the end user and customer. Client accepts the software only when all the features and functionalities work as expected. It is generally the last phase of the testing, after which the software goes into the production stage.
- Ad-Hoc Testing: As the name suggests, this testing is performed on an ad-hoc basis i.e. with no reference to the test case and also without any plan or documentation in place for such type of testing. The objective of this testing is to find the bugs in the app and break the app by executing any flow.
- Beta Testing : Beta Testing is a formal type of software testing which is carried out by the customer. It is performed in the Real Environment before releasing the product to the market for the actual end users.
- Comparison Testing : Comparison of a product's strength and weaknesses with its previous versions or other similar products is termed as Comparison Testing.
- Compatibility Testing : It is a testing type in which it validates how software behaves and runs in a different environment, web servers, hardware, and network environment.

7.2 Test cases & Test Results

1. Functional Test 1
    - Title: **Signup**
    - Input: Name, Username, Email Id, Password
    - Expected Output: Create a user's account, which can be used for voting as well as creating elections.
    - Test Result: Positive
2. Functional Test 2
    - Title: **Login**
    - Input: Email Id, Password
    - Expected Output: Login to the user's account and fetch the data.
    - Test Result: Positive
3. Functional Test 3
    - Title: **Create an Organization**
    - Input: Organization Name

- Expected Output: User should be able to create a new organization.
- Test Result: Positive

4. Functional Test 4

   - Title: **Create Election**
   - Input: Election Name, Registration Link, Start Time, End Time
   - Expected Output: A new election should be created and stored into the database.
   - Test Result: Positive

5. Functional Test 5

   - Title: **Deploy Election**
   - Input: Same as above
   - Expected Output: A new election should be deployed onto the blockchain network, after receiving a confirmation from the user.
   - Test Result: Positive

6. Functional Test 6

   - Title: **Register to Vote in an Election**
   - Input: Unique Voter Id
   - Expected Output: A request should be sent to the organization for approving the voter.
   - Test Result: Positive

7. Functional Test 7

   - Title: **Vote in an Election**
   - Input: A new Signed Metamask Wallet, Approval from the organization.
   - Expected Output: The voter should be able to cast his/her vote to the candidate of his/her choice.
   - Test Result: Positive

8. Functional Test 8

   - Title: **View Election result**
   - Input: None
   - Expected Output: The users should be able to view the result of the election immediately after the election is over.
   - Test Result: Positive

CHAPTER 8

RESULTS

8.1 Outcomes

This project presents a framework that would be utilizing both blockchain technology and multi-factor authentication to provide an easily accessible voting system while also securely protecting the casted votes and verifying the voter's eligibility to cast their votes. The implementation of this proposed framework would offer multiple benefits in the electoral sector and some of these benefits would be -

- A polling station would not need to be set up due to the fact that the voters can make use of their mobile/laptop device to partake in the voting procedure which will help to reduce the high cost that is required in setting up polling stations.
- A public holiday need not be declared by the government as the voters shall be given access to the system only on the day of voting. This will not halt the nation's economy for the day.
- Voters will not be required to leave their homes or place of work in order to cast their votes.
- Problems associated with the double casting of votes can be mitigated.
- This could also provide easier accessibility of the voting procedure to multiple individuals including the elderly and those with disabilities.
- The casted votes would be safely and securely stored in the blockchain database which would mitigate any form of manipulation or tampering like the deletion of legitimate votes or the addition of illegitimate votes.
- The casting of votes and tallying of the votes would be a lot quicker while also providing better transparency and accuracy.
- There will be a progressive increment in the number of youths taking part in the voting procedure.

Hence, our system takes advantage of the transparency of smart contracts to allow all voters to participate in both the recording and verification of ballots. It will enhance the voters' confidence and also reduce the wastage of election resources.

8.2 Screen Shots

Figure 8.1: Landing Page

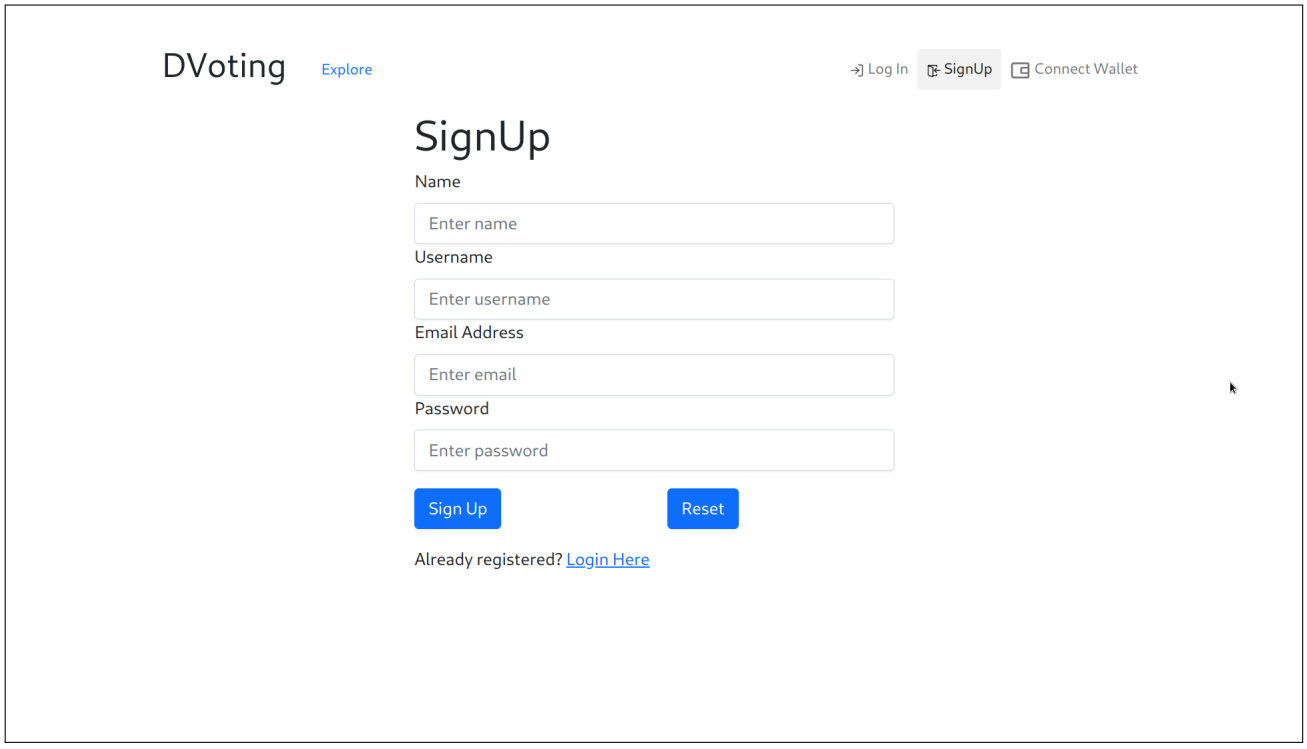


Figure 8.2: Signup Page

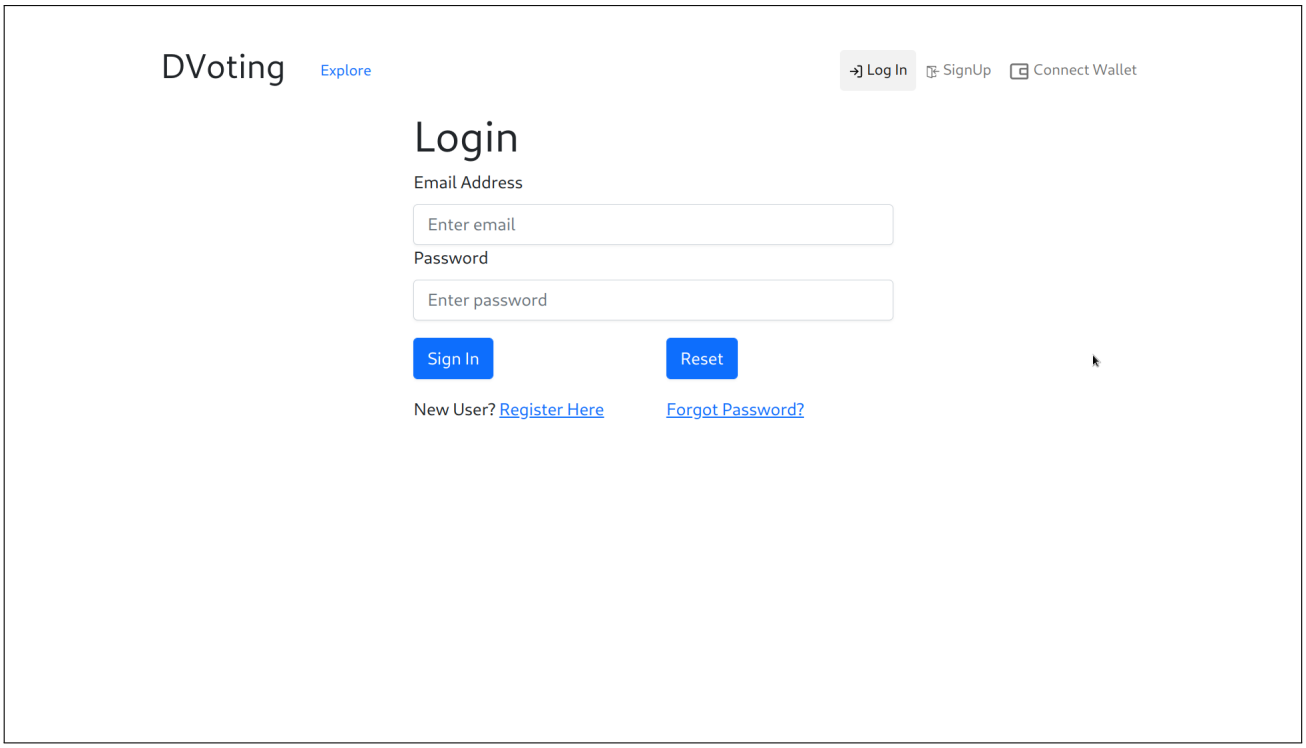


Figure 8.3: Login Page

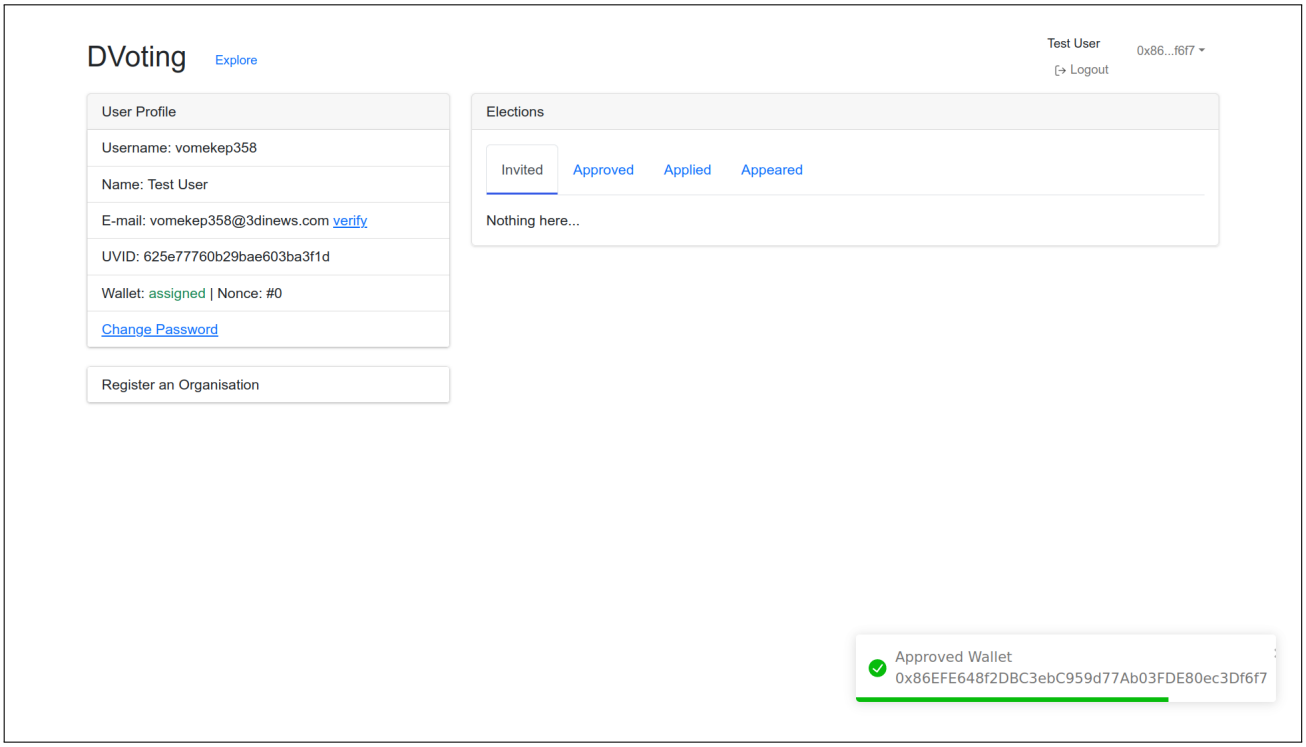


Figure 8.4: User Dashboard

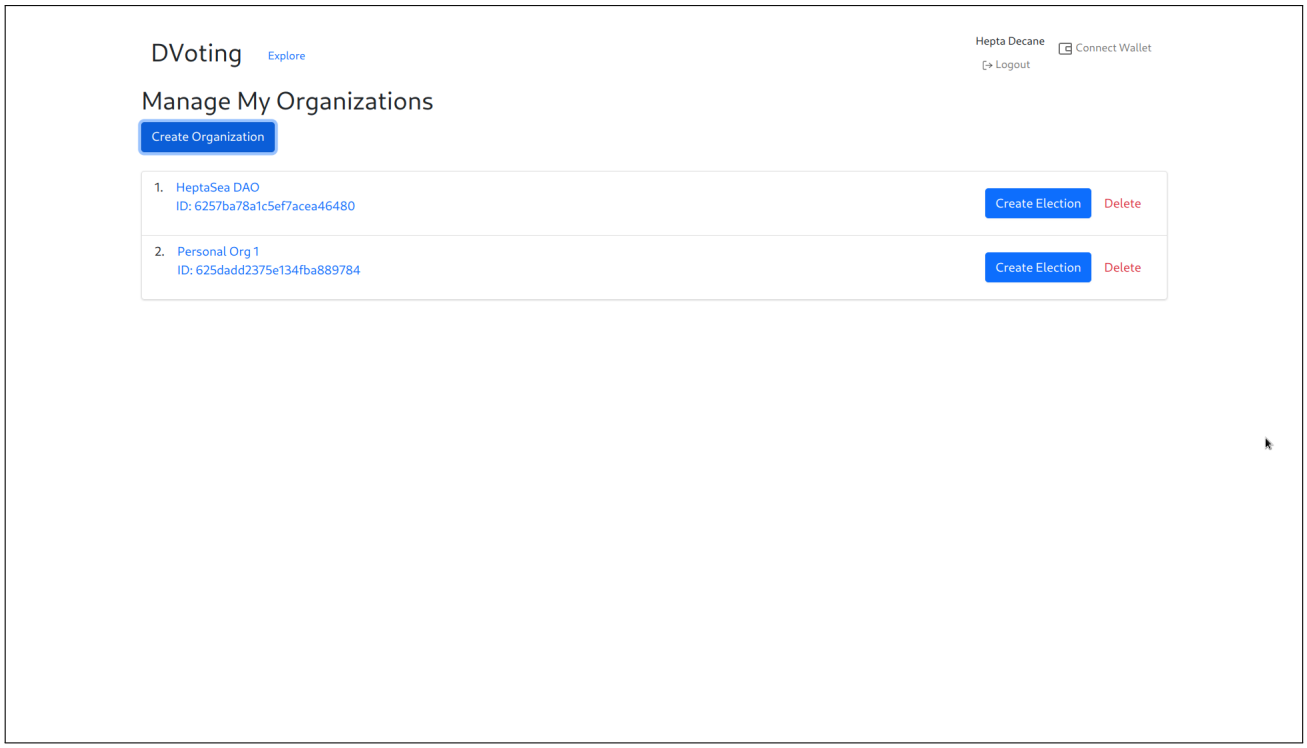


Figure 8.5: Manage Organizations

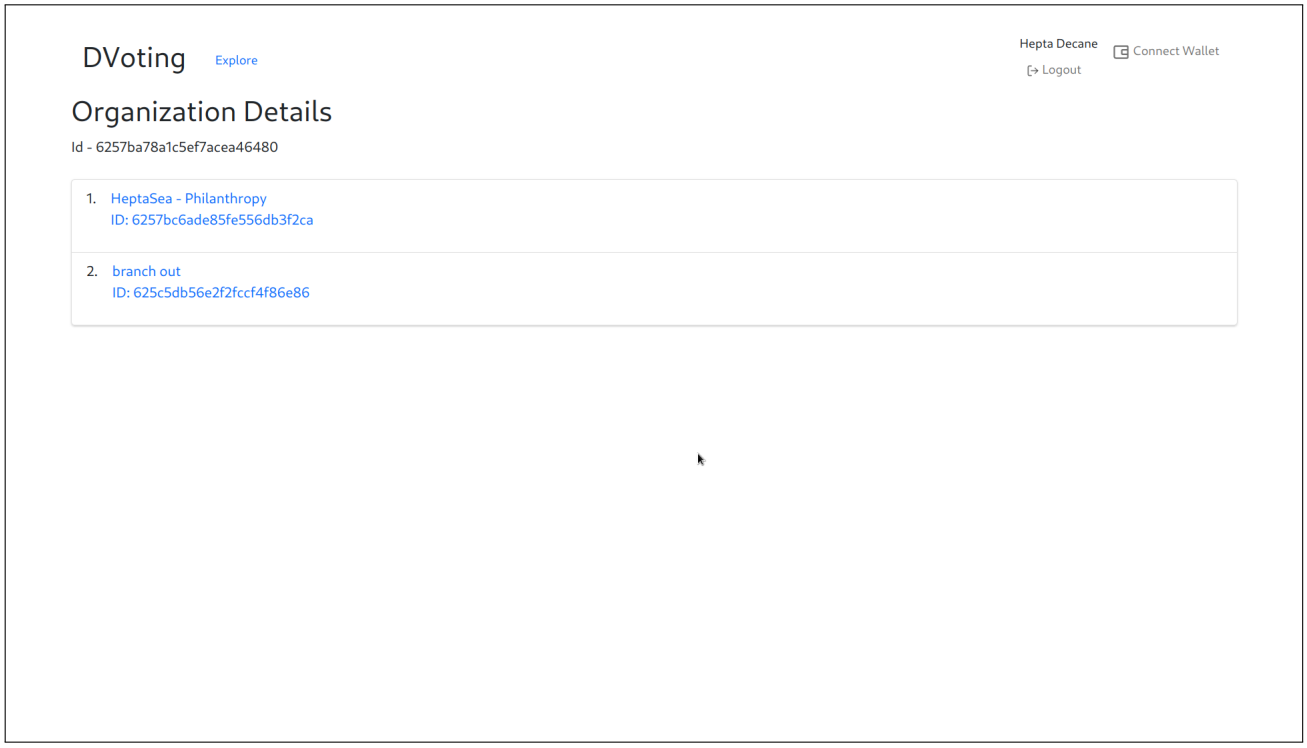


Figure 8.6: My Elections

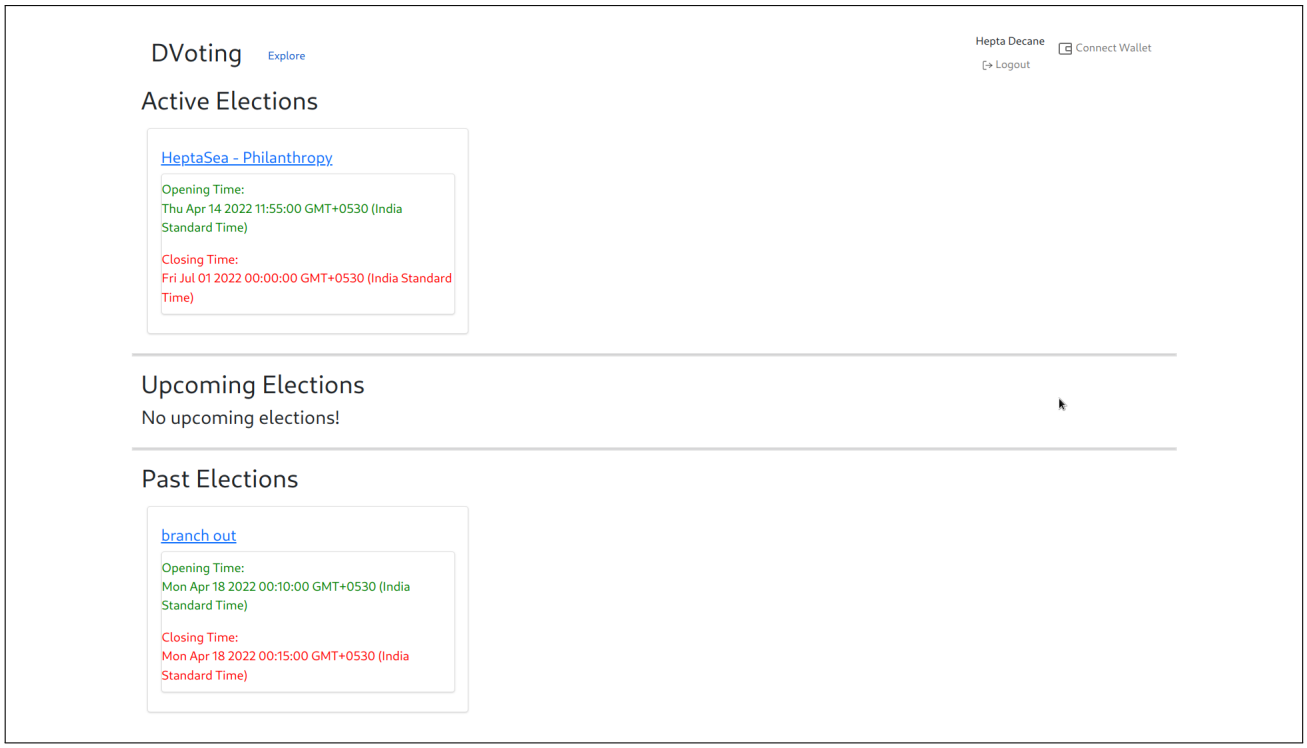


Figure 8.7: Explore Elections

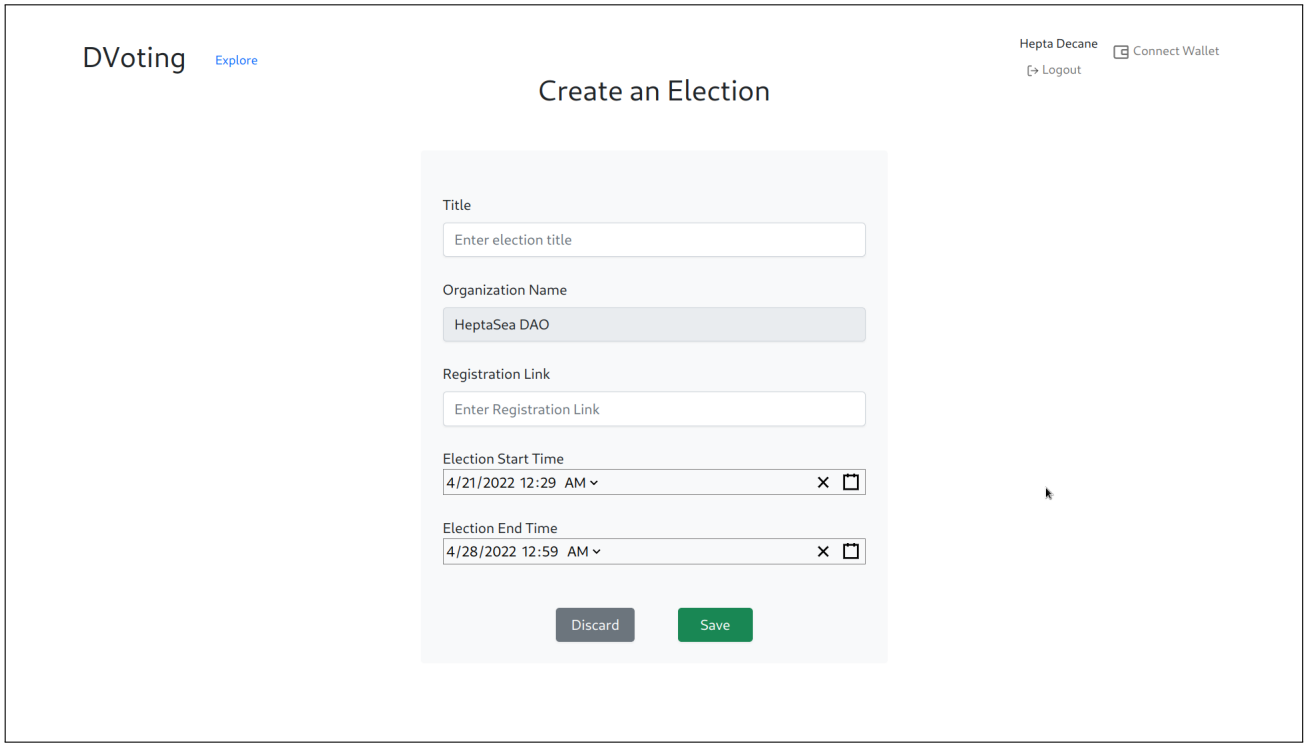


Figure 8.8: Create Election - 1

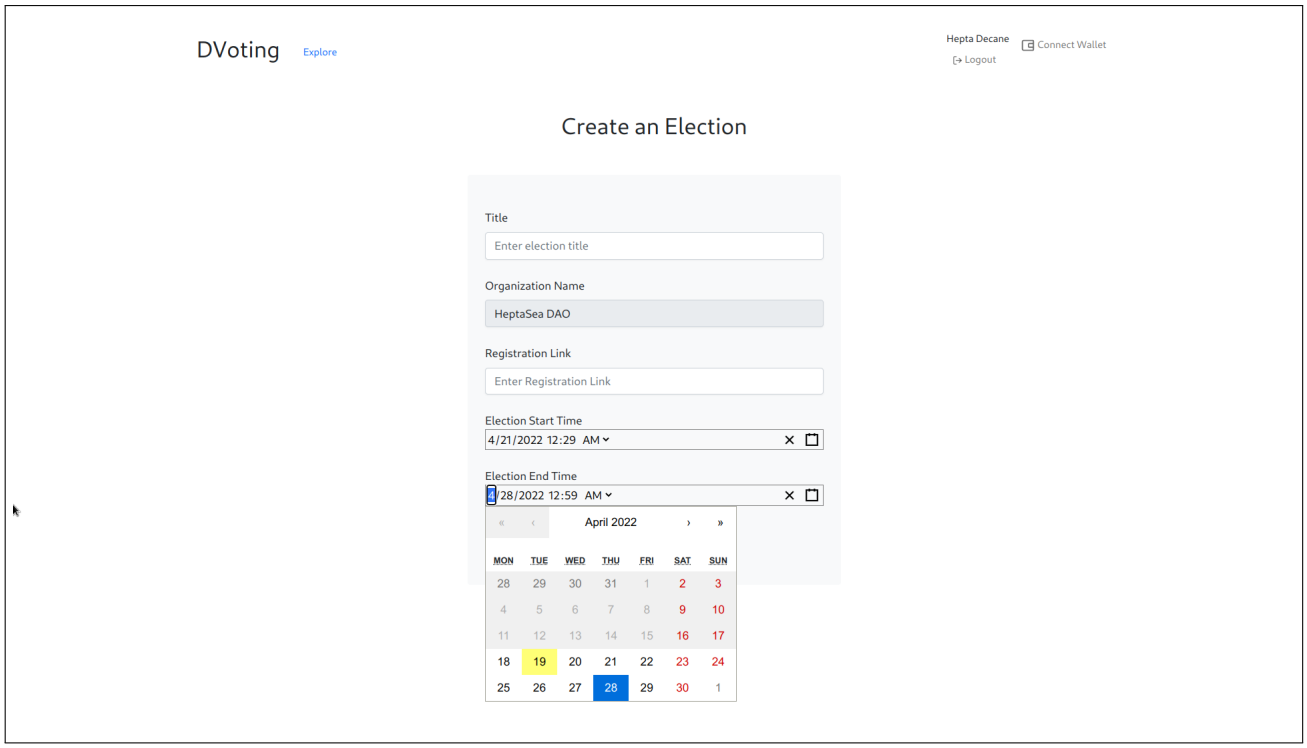


Figure 8.9: Create Election - 2

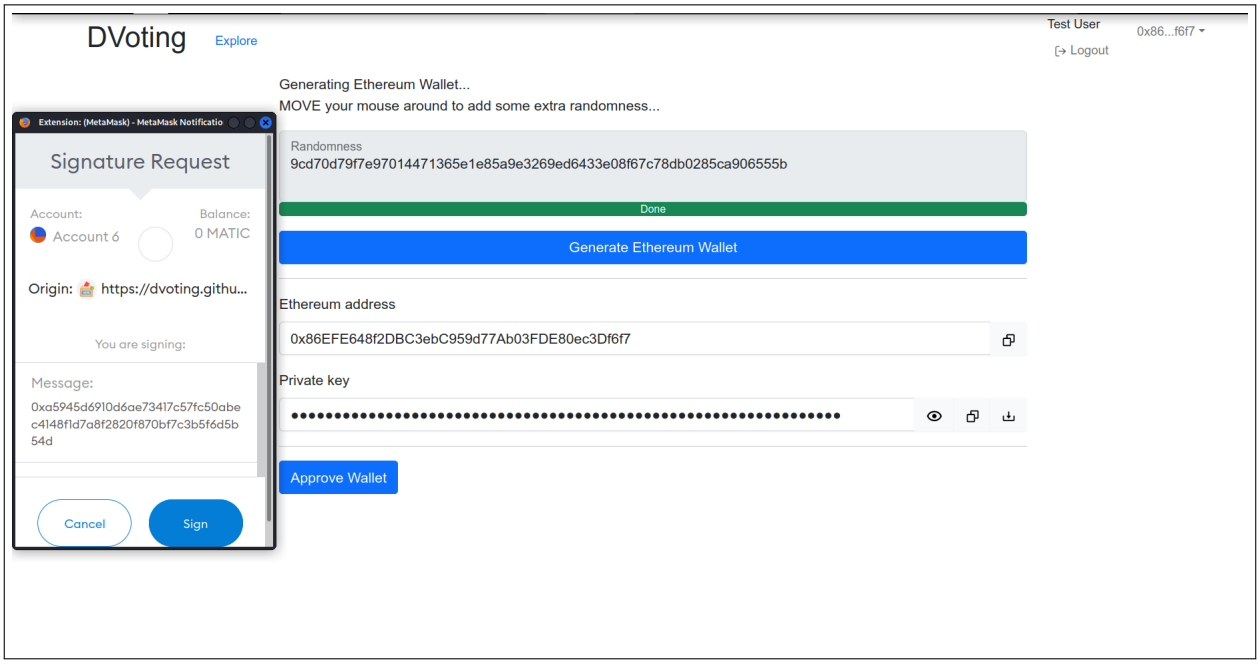


Figure 8.10: Generate & Approve Wallet

Figure 8.11: Election Page

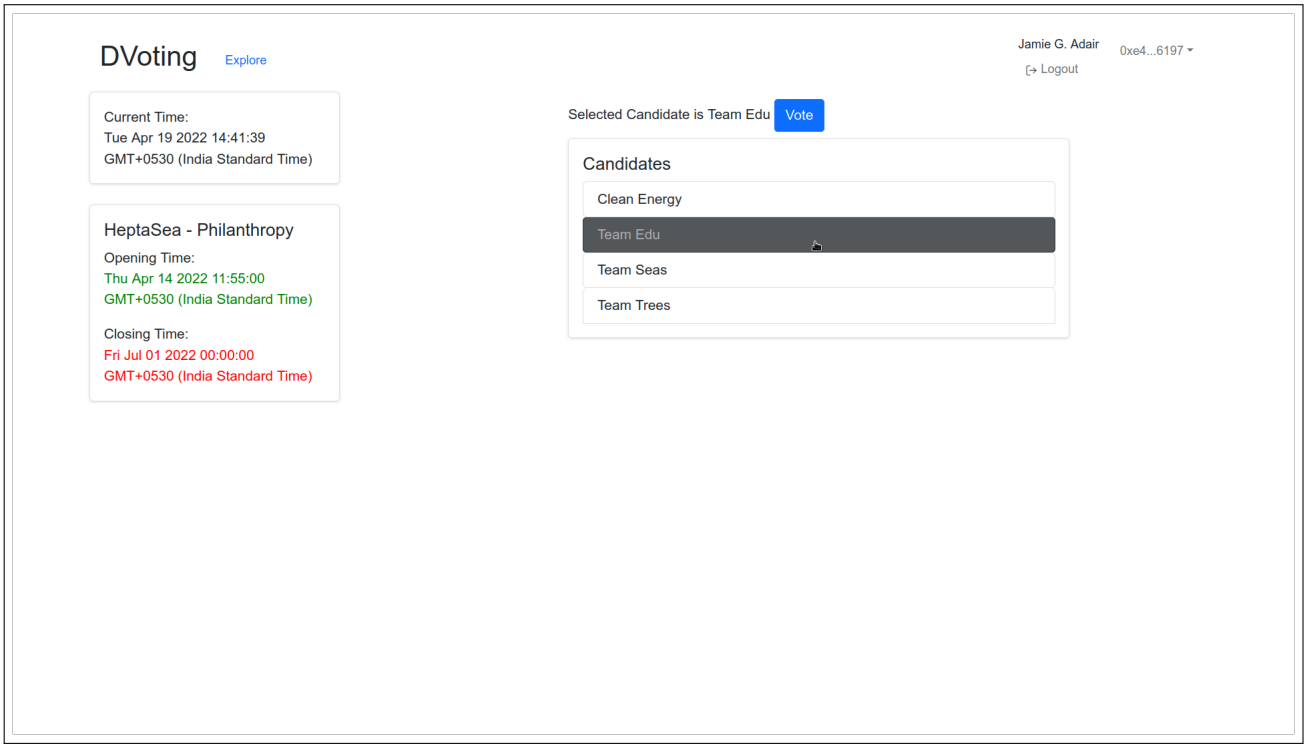


Figure 8.12: Cast Vote Page - 1

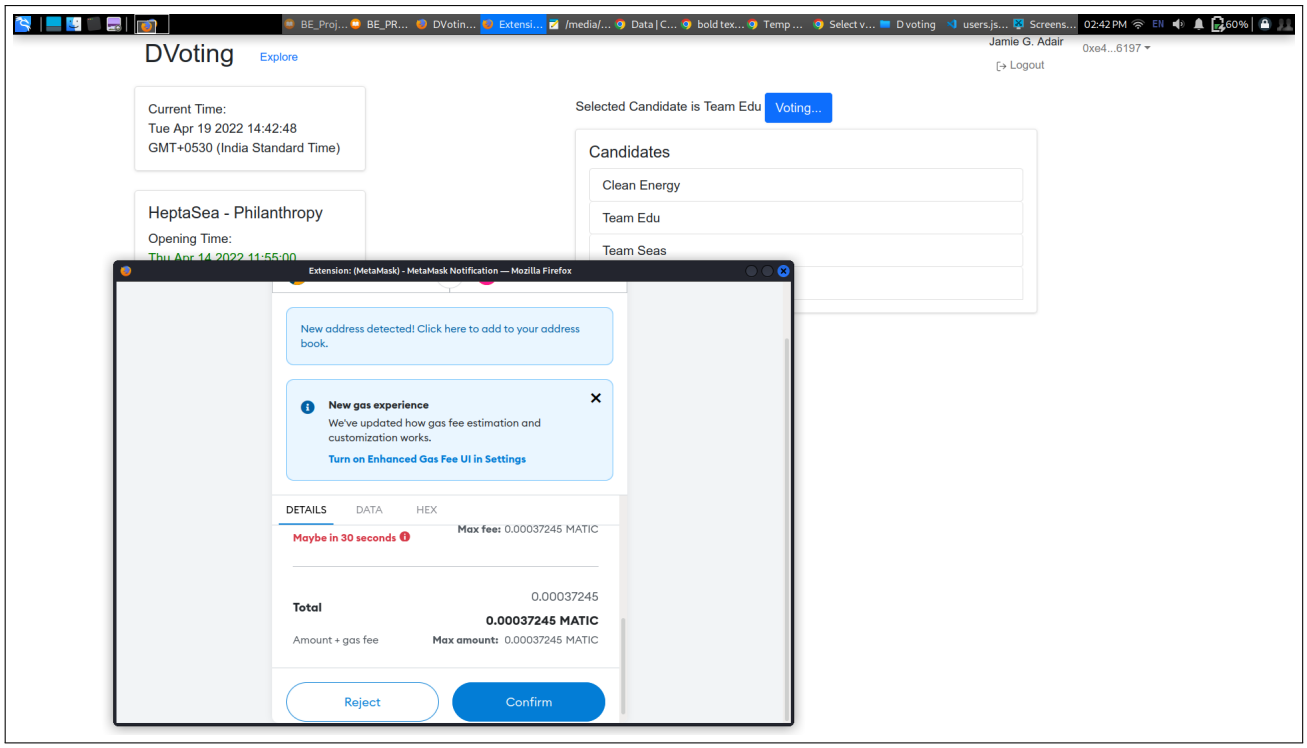


Figure 8.13: Cast Vote Page - 2

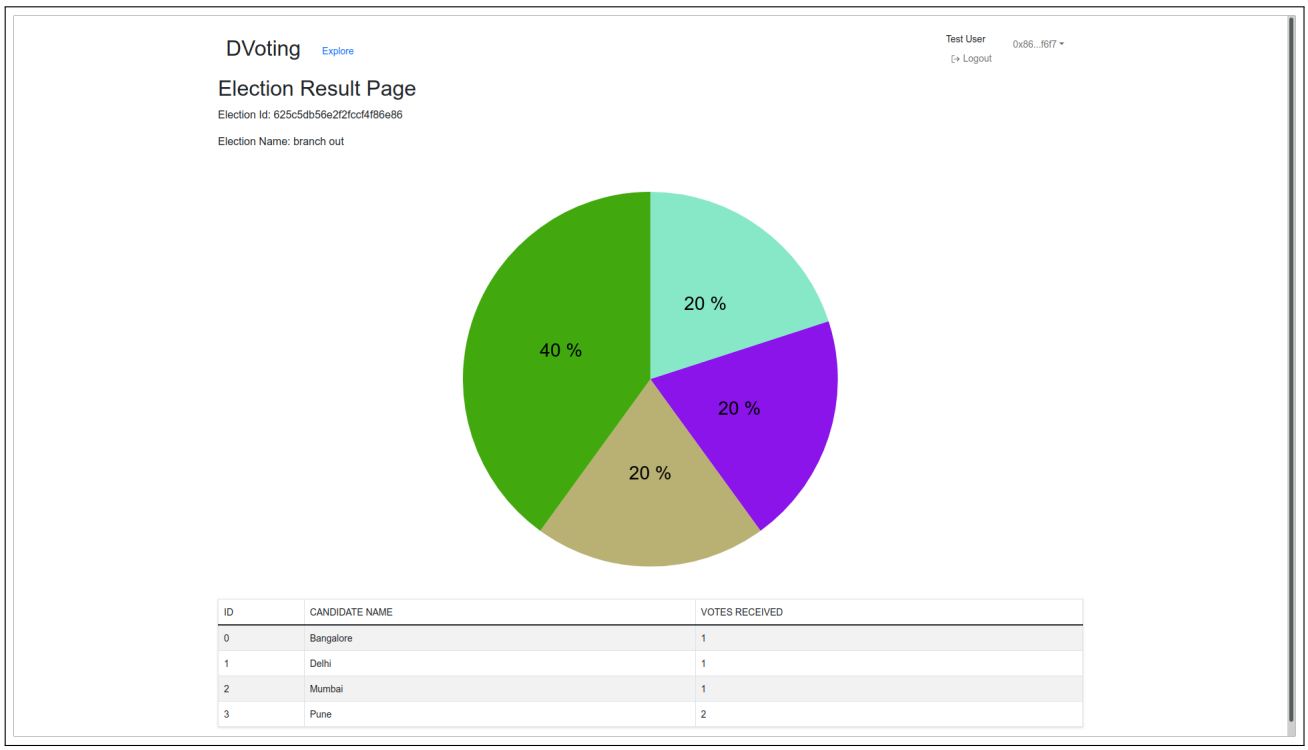


Figure 8.14: Election Result

CHAPTER 9

CONCLUSIONS

9.1 Conclusion

Problem statement was analyzed and a software solution has been presented using Blockchain technologies and proposes a plan for developing a secure, safe and transparent Decentralized Voting System. First, the current Electoral system's flaws were determined and new e-voting schemes are discussed using the Blockchain principles and its implementations. This application tries to put forward a strong case for a new and secure Blockchain based E-Voting system. The detailed system architecture has also been provided.

9.2 Future Work

For the future scope, we would like to optimize the website for small screen devices, develop a native mobile application for decentralized voting as it would make our application accessible to a larger audience, as many users today prefer smart phones over large screen desktops.
We would also look further into this domain to increase the efficiency and accuracy of this software.

9.3 Applications

- This system will be useful for conducting any kind of election procedure, be it national elections or even college society elections.
- This system will provide easier accessibility of the voting procedure to multiple individuals including the elderly and those with disabilities.

APPENDIX A: MATHEMATICAL MODEL

Mathematical Model :-

- System S is defined as S = LP, V, Cname, Vid, Cid, Su, F.
- Input
  - Login Process LP = lp1, lp2, ... , lpn.
    where, LP is the set of login voters and lp1, lp2, ... , lpn are the number of voters.
  - V = Set of voters
  - Cname = Candidate's Name
  - Vid = Voter's unique Id
  - Cid = Candidate's unique Id
- Process
  - Vote to be casted.
  - Check validity of vote casted with nodes in the blockchain.
  - If valid vote, execute the vote.
- Output
  - Su = Successful Vote
  - F = Failed Vote
  - Success Condition: The user voted should have voted only once in the current election and vote should be successfully marked.
  - Failure Condition: The user voted would be either trying to vote more than once in the current election or the vote would not be successfully marked.

APPENDIX B:
PLAGIARISM REPORT

Curiginal

Document Information

Analyzed document	BE_PROJECT_REPORT - Group 71.pdf (D136579626)
Submitted	2022-05-15T06:12:00.0000000
Submitted by	Amar
Submitter email	arbuchade@pict.edu
Similarity	5%
Analysis address	arbuchade.pict@analysis.urkund.com

Sources included in the report

W URL: https://www.ijsdr.org/papers/IJSDR1905104.pdf
Fetched: 2021-10-11T06:07:58.2630000

W URL: https://www.researchgate.net/publication/342932007_E-Voting_using_Blockchain_Technology
Fetched: 2020-12-18T03:27:01.7400000

W URL: https://ieeexplore.ieee.org/document/9299581
Fetched: 2021-03-24T10:58:26.8100000

W URL: https://www.sciencegate.app/document/10.35940/ijitee.h6515.079920
Fetched: 2021-06-09T13:09:01.2500000

W URL: https://www.hindawi.com/journals/sp/2022/1383007/
Fetched: 2022-01-21T04:33:45.9600000

W URL: https://www.ijert.org/research/e-voting-using-blockchain-IJERTV10IS030138.pdf
Fetched: 2021-11-22T16:43:32.5200000

W URL: https://www.ijert.org/research/implementation-of-secure-voting-system-using-blockchain-IJERTV9IS060974.pdf
Fetched: 2020-12-18T03:27:00.3100000

W URL: https://www.hindawi.com/journals/scn/2021/6673691/
Fetched: 2021-05-06T04:50:47.6330000

W URL: https://www.mdpi.com/2073-8994/12/8/1328/htm
Fetched: 2020-12-18T03:27:00.1930000

REFERENCES

References

[1] A. M. Al-madani, A. T. Gaikwad, V. Mahale and Z. A. T. Ahmed, "Decentralized E-voting system based on Smart Contract by using Blockchain Technology" 2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC), 2020, doi: 10.1109/ICSIDEMPC49020.2020.9299581.

[2] D. Khoury, E. F. Kfoury, A. Kassem and H. Harb, "Decentralized Voting Platform Based on Ethereum Blockchain" 2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET), 2018, doi: 10.1109/IMCET.2018.8603050.

[3] Hanifatunnisa, Rifa & Rahardjo, Budi. (2017). Blockchain based e-voting recording system design. 1-6. 10.1109/TSSA.2017.8272896.

[4] L. V. Thuy, K. Cao-Minh, C. Dang-Le-Bao and T. A. Nguyen, "Votereum: An Ethereum-Based E-Voting System" 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF), 2019, doi: 10.1109/RIVF.2019.8713661.

[5] F. Þ. Hjálmarsson, G. K. Hreiðarsson, M. Hamdaqa and G. Hjálmtýsson, "Blockchain-Based E-Voting System," 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), 2018, doi: 10.1109/CLOUD.2018.00151.

[6] Shivam Jaiswal , Yash Dalvi , Pawan Sharma, 2021, "E-Voting using Blockchain", INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 03 (March 2021), https://www.ijert.org/e-voting-using-blockchain.