BCAD IA REPORT: E-voting System by 16010420062,16010420069 and 16010420073

1. INTRODUCTION

1.1 Problem Definition

When it comes to ensuring equitable and secure internal elections, private institutions such as corporations, universities, and non-profits frequently face obstacles. These obstacles include anonymity, confidentiality, security, and data manipulation issues.

To surmount these obstacles, a blockchain-based decentralized voting system can be implemented. The system should be designed to ensure that every vote is tabulated accurately and cannot be manipulated in any way. It should also guarantee voter anonymity so they can express their preferences without fear of reprisal. The proposed system should be user-friendly so that all participants can access and utilize it without difficulty.

Additionally, the system should be designed to prevent any single entity from controlling the election's outcome. This decentralized voting system aims to provide private institutions with a dependable and secure platform for conducting internal elections. This will assist institutions in ensuring the integrity of their elections and ensuring that the results accurately reflect the will of the electors, without the risk of deception or data manipulation.

1.2 Functional and Non-Functional Requirements

Functional Requirements:

- Account Verification: The admin and user should both need to possess and activate a valid MetaMask account on their system to activate the e-voting website.
- Voter Registration: The voter must have to first register as a voter himself/herself upon starting of the election by the admin in order to vote.
- Ballot creation: Admin should be able to create a ballot by adding the relevant election details such as ‘Details of the Admin’, ‘Title of Election’, and ‘Organization’, in order to carry out the elections.

- Voting phase control: Admin should have the control of changing the election phase (i.e. Registration phase, Voting phase and Result Phase)
- Publish Result: The admin should be able to end the voting phase and publish the results to the voters on the portal itself.
- Reset and start new election: Upon ending the previous election, the admin should be able to start a new election all over again using the CLI.

Non-functional Requirements:

- Security: The software should provide strong protection against unauthorized access, tampering, hacking attempts, and be highly secure overall.
- Confidentiality/Anonymity: The software should maintain the anonymity of the vote casted by the voter. The votes need to be anonymous to the admin as well in order to maintain the confidentiality.
- Integrity: The software should protect the integrity of the vote while still ensuring the privacy of the voter and the admin both.
- User Experience: The e-voting system should be accessible to all voters, regardless of their technical proficiency or physical abilities, it should be transparent, with clear and verifiable results, system should inspire trust in the voters, by providing a fair and reliable voting process, feedback must be provided to the voters, to let them know that their vote has been recorded and counted.

2. IMPLEMENTATION

2.1 Technologies Used

1. **HTML:** The Hypertext Markup Language or HTML is the standard mark-up language for documents designed to be displayed in a web browser. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

   Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.

2. **CSS:** Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of content and presentation, including layout, colours, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

3. **JavaScript:** Often abbreviated as JS, JavaScript is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98% of websites use JavaScript on the client side for webpage behaviour, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices.

   JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for

working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

4. **React JS:** ReactJS tutorial provides basic and advanced concepts of ReactJS. Currently, ReactJS is one of the most popular JavaScript front-end libraries which has a strong foundation and a large community.

   ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front-end library which is responsible only for the view layer of the application. It was initially developed and maintained by Facebook and later used in its products like WhatsApp & Instagram.

5. **NodeJS:** Node.js is a cross-platform, open-source server environment that can run on Windows, Linux, Unix, macOS, and more. Node.js is a back-end JavaScript runtime environment, runs on the V8 JavaScript Engine, and executes JavaScript code outside a web browser.

   Node.js lets developers use JavaScript to write command line tools and for server-side scripting. The ability to run JavaScript code on the server is often used to generate dynamic web page content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, as opposed to using different languages for the server- versus client-side programming.

6. **Ganache:** Ganache is a personal blockchain for rapid Ethereum and Filecoin distributed application development. You can use Ganache across the entire development cycle; enabling you to develop, deploy, and test your dApps in a safe and deterministic environment.

7. **Truffle:** Truffle is a world-class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier. Truffle is widely considered the most popular tool for blockchain application development with over 1.5 million lifetime downloads. Truffle supports developers across the full lifecycle of their projects, whether they are looking to build on

Ethereum, Hyperledger, Quorum, or one of an ever-growing list of other supported platforms. Paired with Ganache, a personal blockchain, and Drizzle, a front-end dApp development kit, the full Truffle suite of tools promises to be an end-to-end dApp development platform.

8. **Solidity:** Solidity is an object-oriented programming language created specifically by the Ethereum Network team for constructing and designing smart contracts on Blockchain platforms.

    - It's used to create smart contracts that implement business logic and generate a chain of transaction records in the blockchain system.

    - It acts as a tool for creating machine-level code and compiling it on the Ethereum Virtual Machine (EVM).

    - It has a lot of similarities with C and C++ and is pretty simple to learn and understand. For example, a “main” in C is equivalent to a “contract” in Solidity.

    Like other programming languages, Solidity programming also has variables, functions, classes, arithmetic operations, string manipulation, and many other concepts.

9. **VS Code:** Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace. Due to the availability of various extensions, in-built terminal, intelligence, etc. We have used VS Code to increase the speed of development.

10. **GitHub:** GitHub is a Git repository hosting service. GitHub also facilitates with many of its features, such as access control and collaboration. It provides a Web-based graphical interface.

GitHub is an American company. It hosts source code of your project in the form of different programming languages and keeps track of the various changes made by programmers.

It offers both distributed version control and source code management (SCM) functionality of Git. It also facilitates with some collaboration features such as bug tracking, feature requests, task management for every project.

11. **MetaMask:** It is a browser plugin that serves as an Ethereum wallet, and is installed like any other browser plugin. Once it's installed, it allows users to store Ether and other ERC-20 tokens, enabling them to transact with any Ethereum address.

    By connecting to MetaMask to Ethereum-based daps, users can spend their coins in games, stake tokens in gambling applications, and trade them on decentralized exchanges (DEXs). It also provides users with an entry point into the emerging world of decentralized finance, or DeFi, providing a way to access DeFi apps such as Compound and Pool Together.

2.2 Methodology

Algorithm for Workflow:

Step1: Open the Ganache and create a network and Login into the MetaMask wallet extension on the browser.

Step 2: The Admin has to fill the personal details as well as the details about the election being conducted on the main admin page.

Step 3: Now the Admin has to add the candidates that are to be elected in the election and start the Registration phase.

Step 4: The Voter has to login through the respective MetaMask wallet and register himself as a voter.

Step 5: The Voter will only be able to vote if he is approved as a legitimate voter by the Admin.

Step 6: After all voters have successfully voted, the admin will end the voting phase and declare the results.

Step 7: The result of the election can be viewed by both the admin and the voter.

2.3 Design Overview

- Simple and Intuitive: The user interface for the decentralised voting system is simple and intuitive, so that voters can easily understand how to participate in the voting process. A clear and concise set of instructions has been provided to guide voters through each step of the process.
- Secure Authentication: The authentication process is secure, with multiple layers of authentication to prevent unauthorized access to the voting system.
- Clear Ballot Information: The ballot information has been presented clearly, with concise descriptions of each candidate or option. This will make it easy for voters to make an informed decision.
- Minimalist Design: The interface has a minimalist design with a clean and uncluttered layout. This will make it easy for voters to focus on the voting process without being distracted by unnecessary information.
- User-friendly Navigation: The interface has user-friendly navigation with clear buttons and links. The navigation is intuitive, with easy-to-follow instructions for each step of the voting process.

3. RESULTS AND DISCUSSIONS

3.1 Results:

Log into MetaMask wallet browser extension:

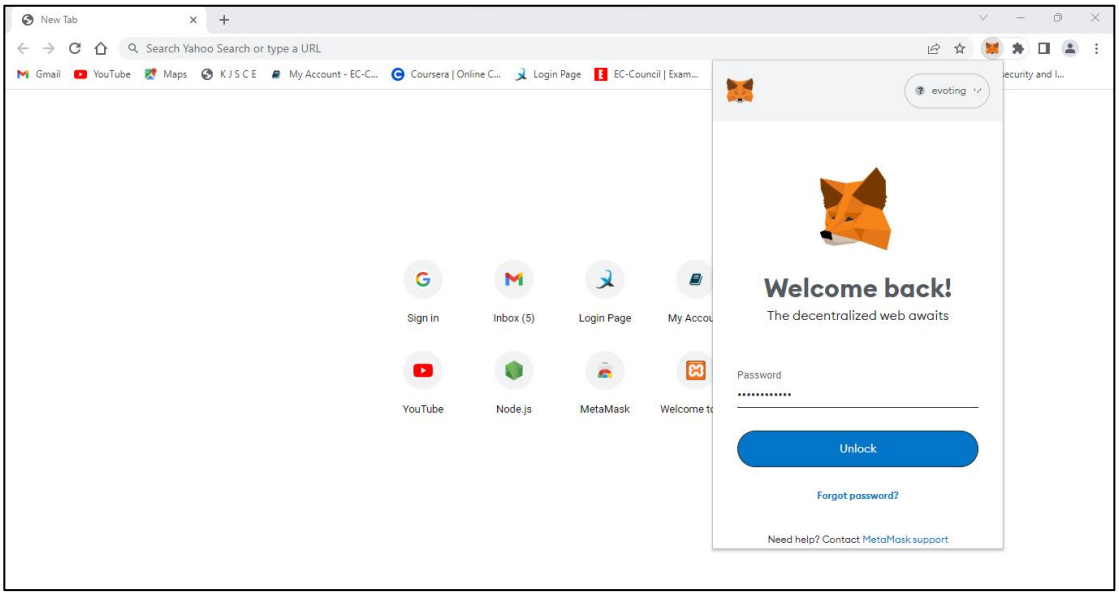

Fig-3.1 MetaMask Extension

Ganache screenshot displaying all the 10 accounts provided to us with default balance of 100 Ethereum for each account:

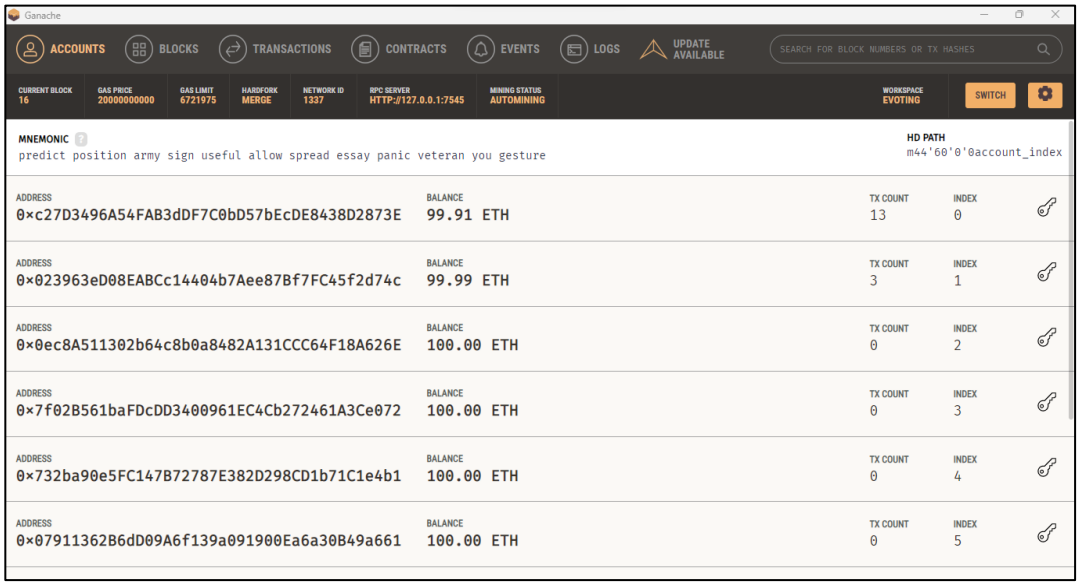

Fig-3.2 Ganache Account

After logging in the MetaMask extension we can see the balance in the MetaMask wallet:

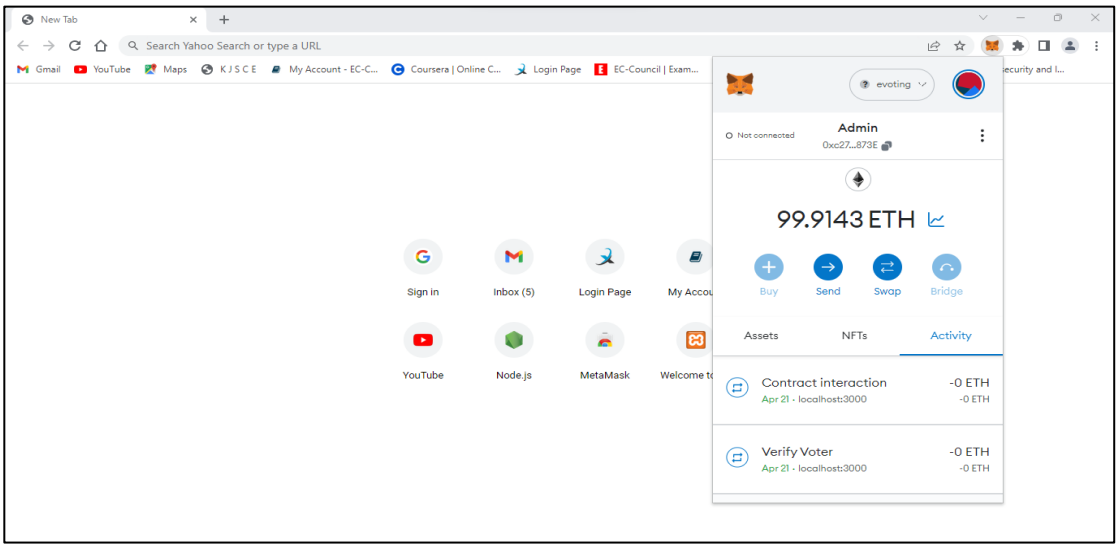

Fig-3.3 Admin's MetaMask Wallet

The "*npm start*" command which fires our development server for our voting system:

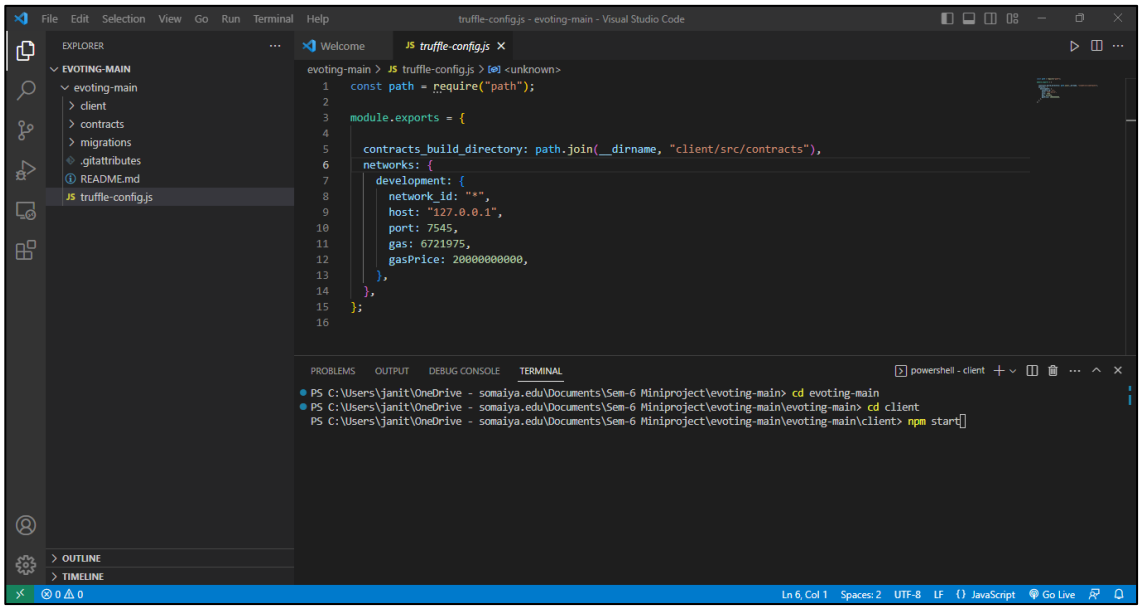

Fig-3.4 Truffle-config.js file

Admin Home Page pops up once we start our development server, it asks for the admin's information to be filled:

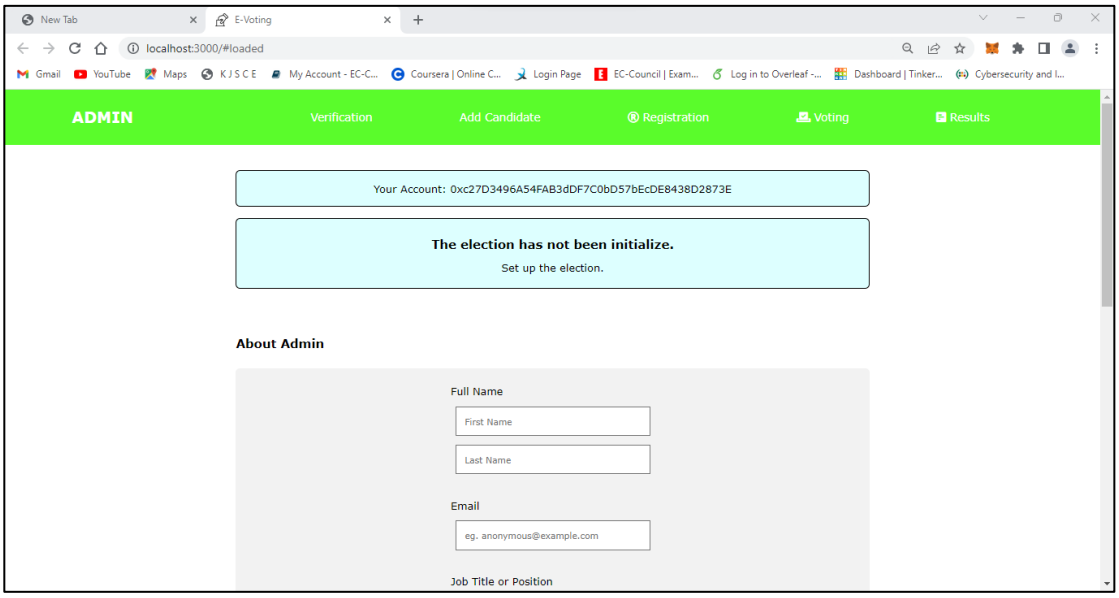

Fig-3.5 Home page

Filling in all the admin information such as Full Name, Email, and Job Title:

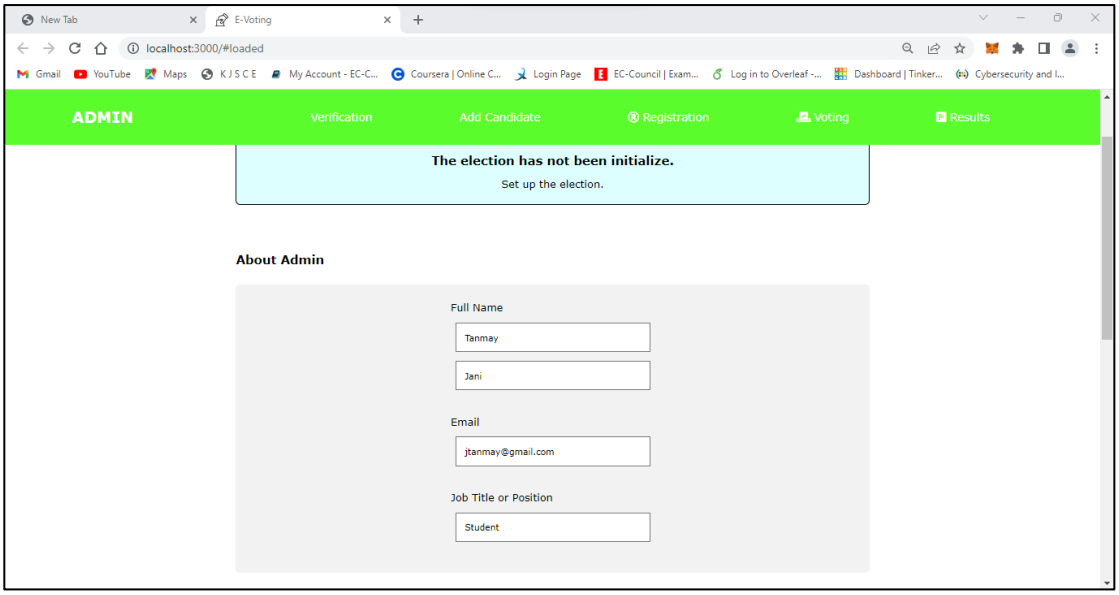

Fig-3.6 Admin's info

Filling in the election title and Organisation name on the same page:

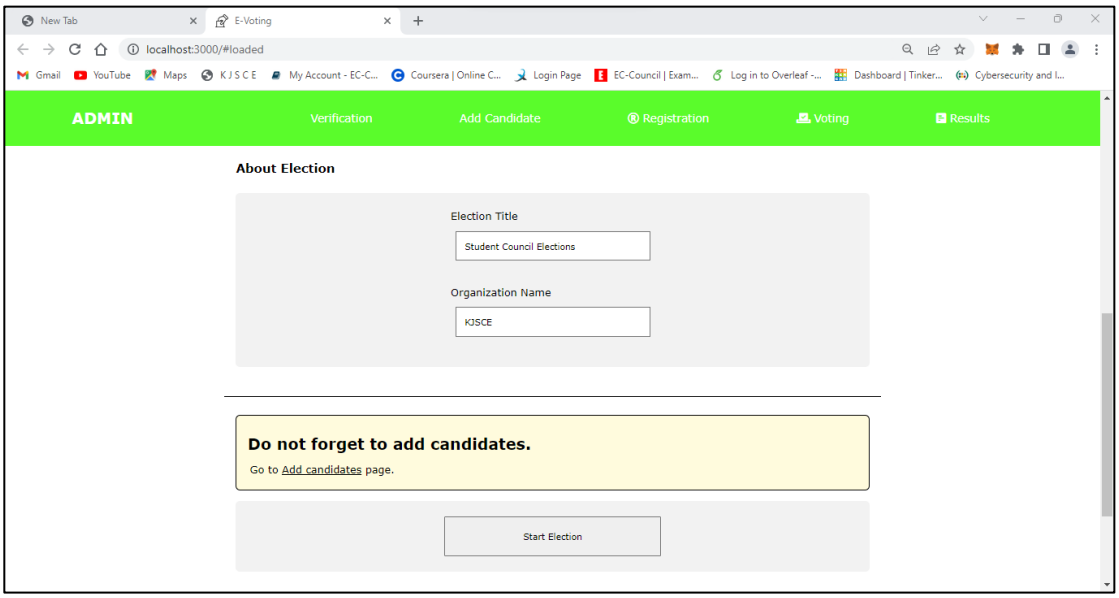

Fig-3.7 Election title and Organisation information

The admin has to start election after filling the above information we can see that election status is False (i.e.: it has not yet started):

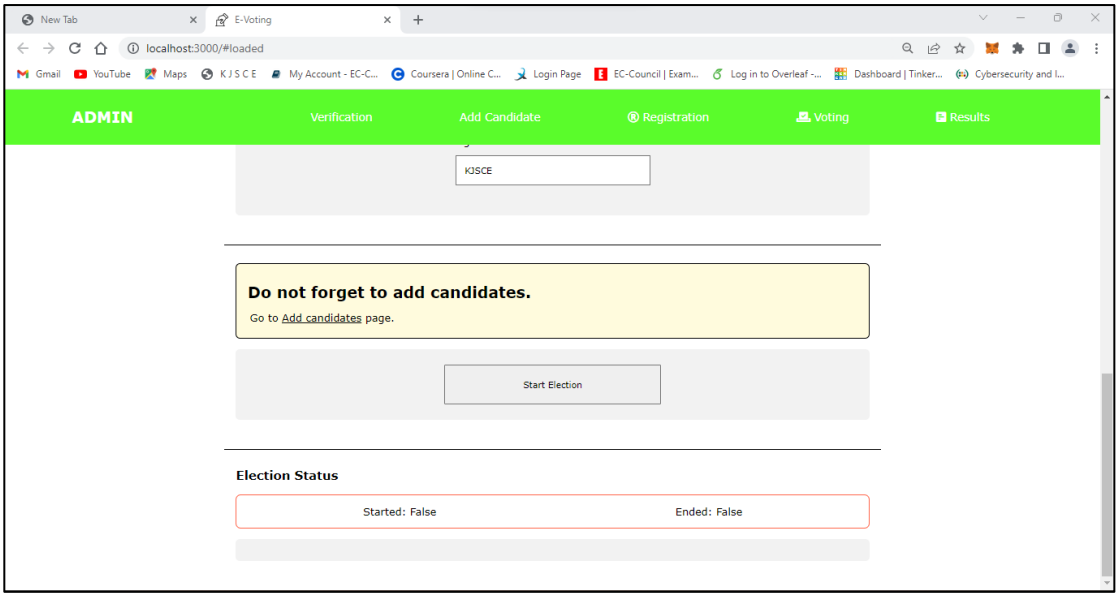

Fig-3.8 Start Election

It is admin's job to add candidates for voting in the election:

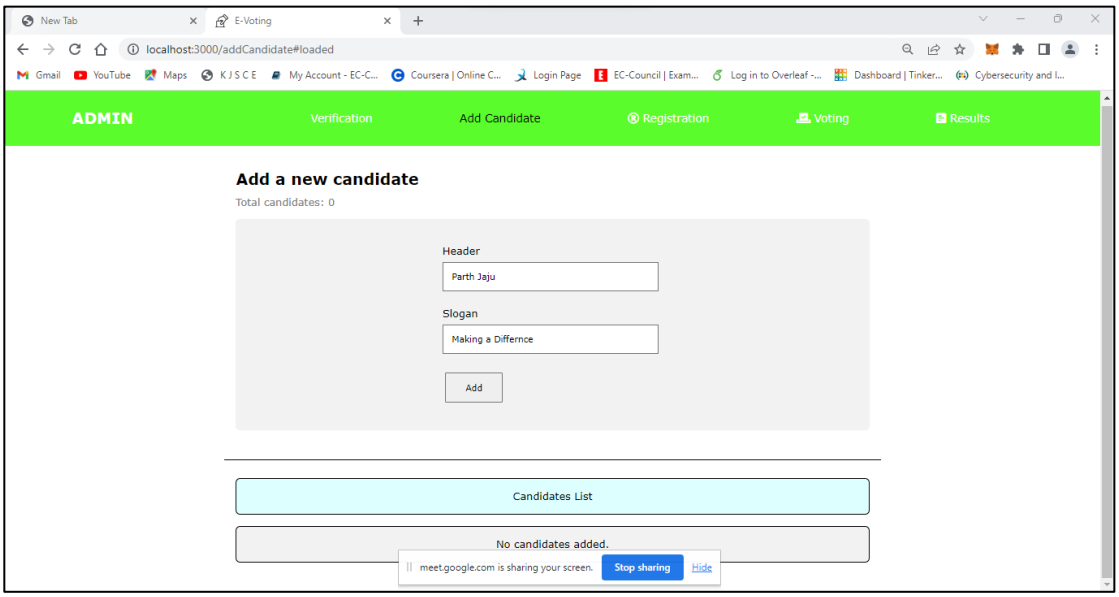

Fig-3.9 Adding First Candidate

After the admin adds the candidate, we can see that the candidate has been added to the candidate list:

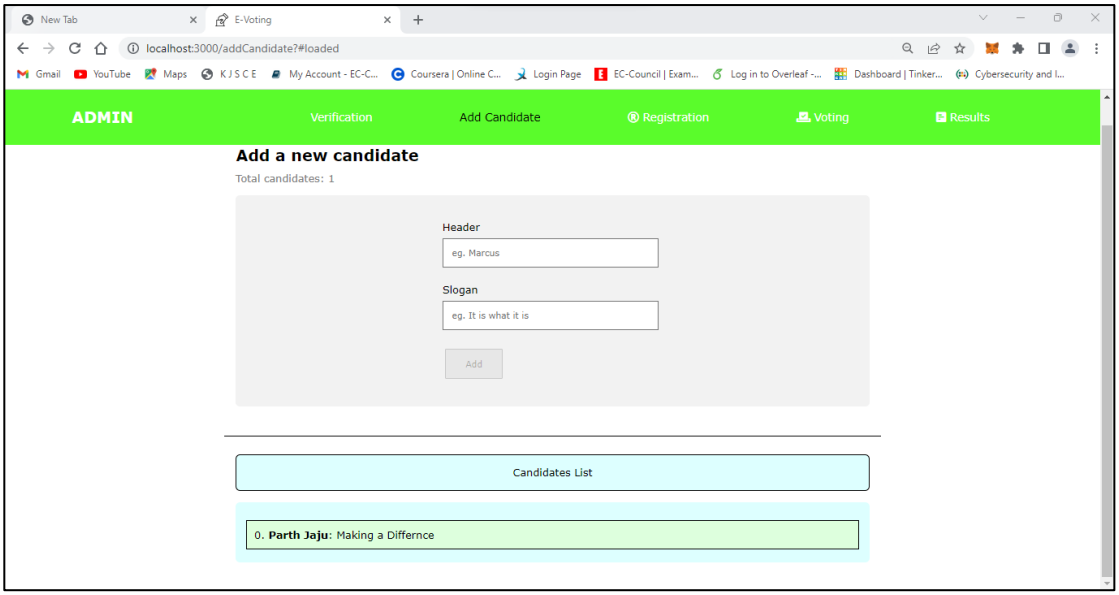

Fig-3.10 Updated Candidates list

First candidate being added with their name and preferred slogan:

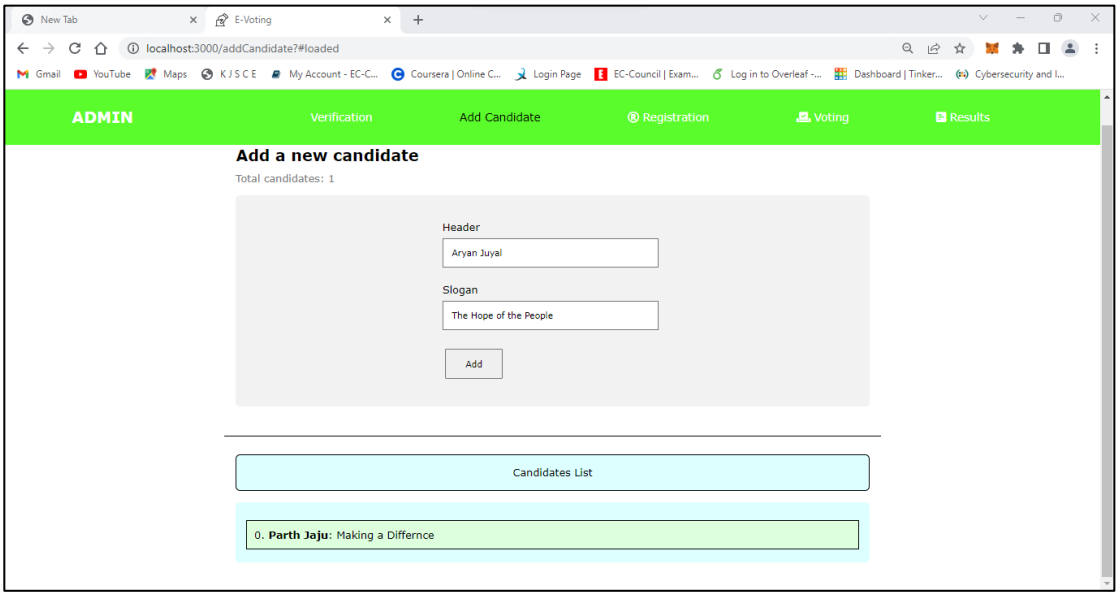

Fig-3.11 Adding Second Candidate

Both the candidates have been added to the list:

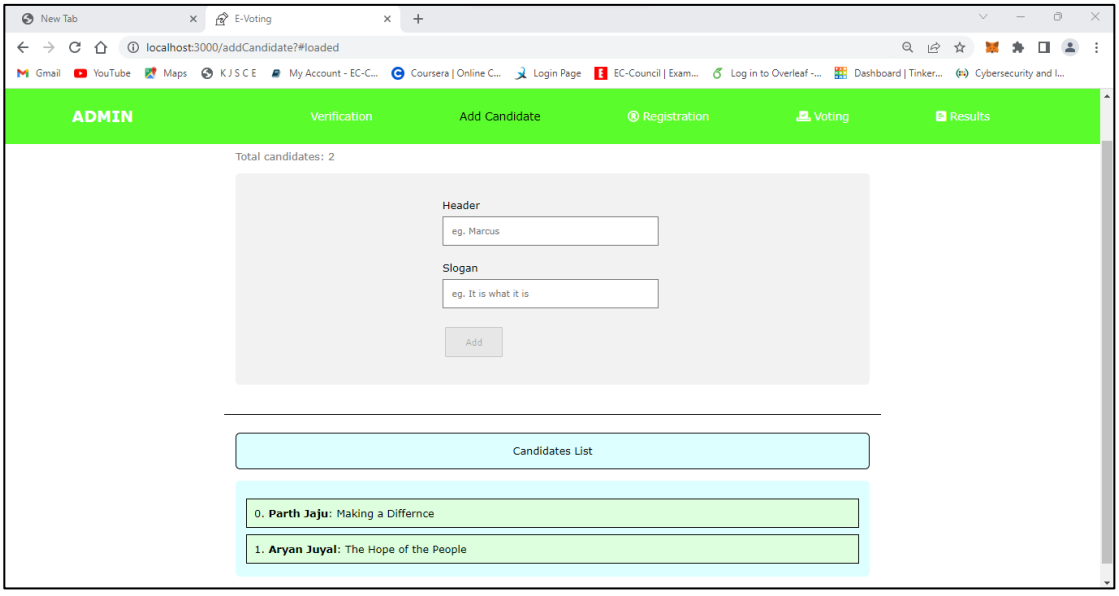

Fig-3.12 Update Candidates list

Screenshot showing the admin page which has election information that the admin had filled earlier:

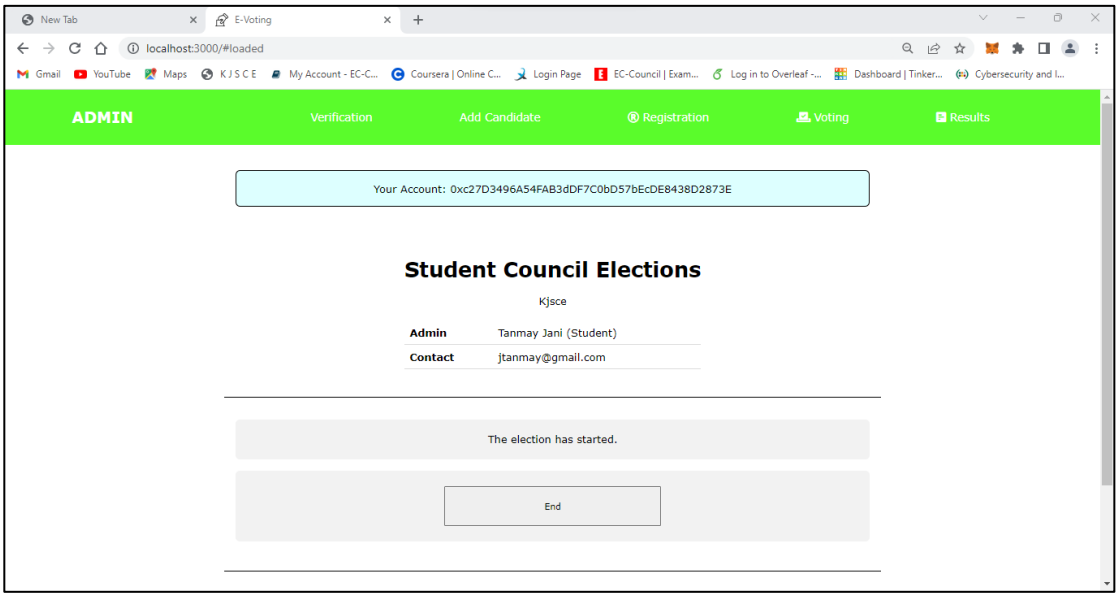

Fig-3.13 Election Information

The user registration phase in which the user is registering himself so that he can vote by adding his valid Name and 10-digit Mobile Number. The Account-Address is pre-filled by default and cannot be edited:

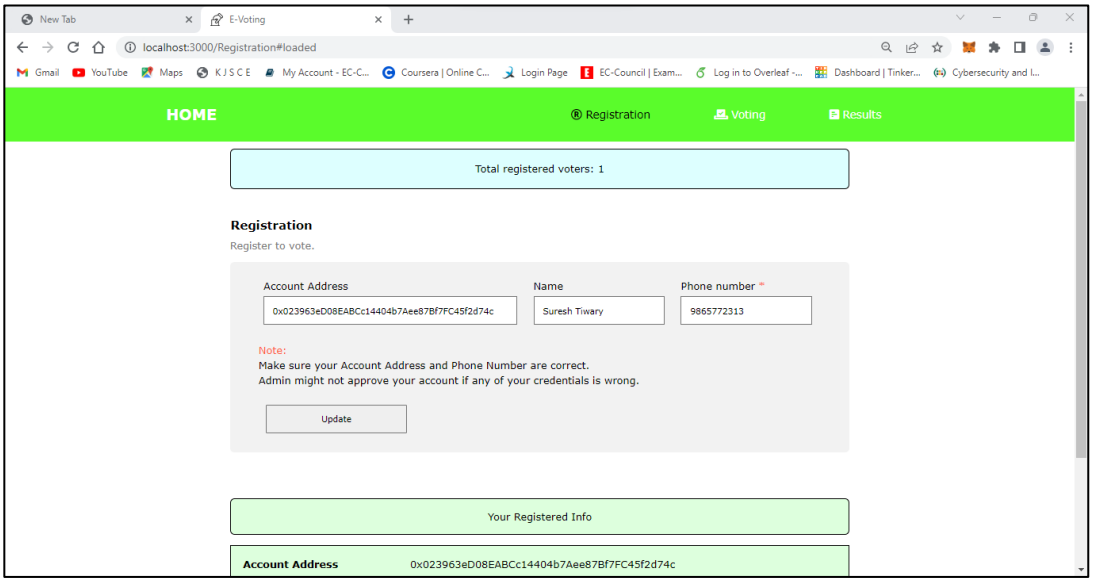

Fig-3.14 User Registration

After the user updates his information, he can see his registered information:

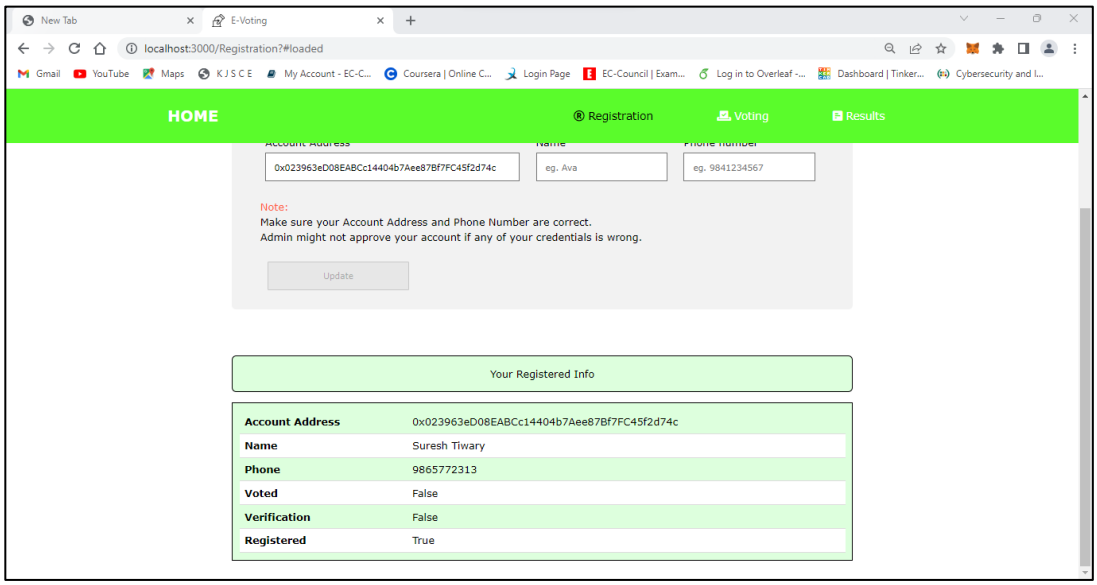

Fig-3.15 Confirming the added information Part 2

Second user registering himself by adding his information:

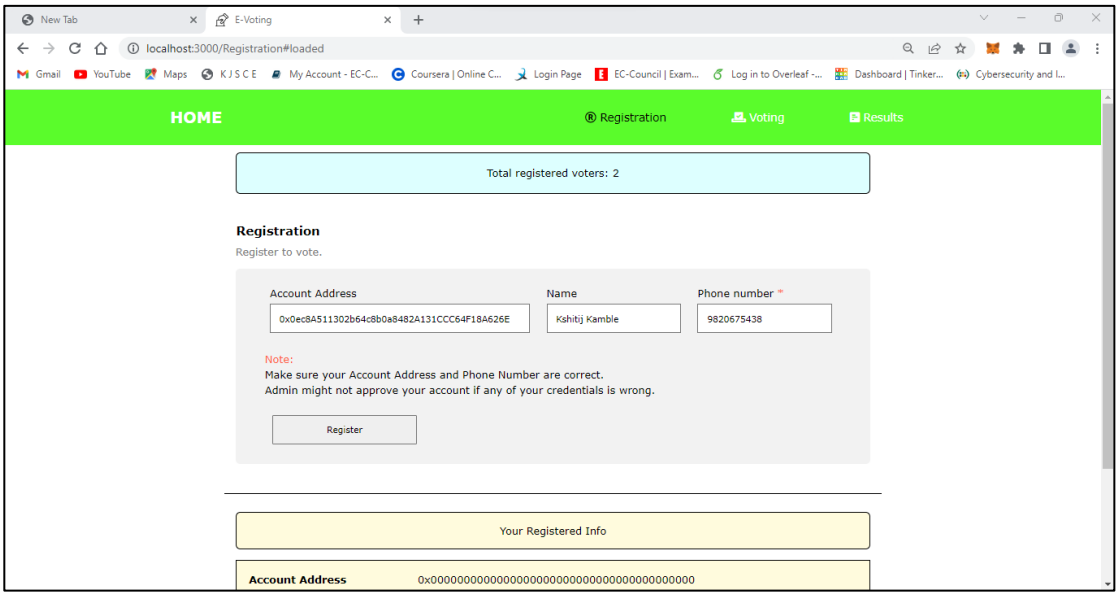

Fig-3.16 User Registration Part 2

Second user will be able to see his information after the registration:

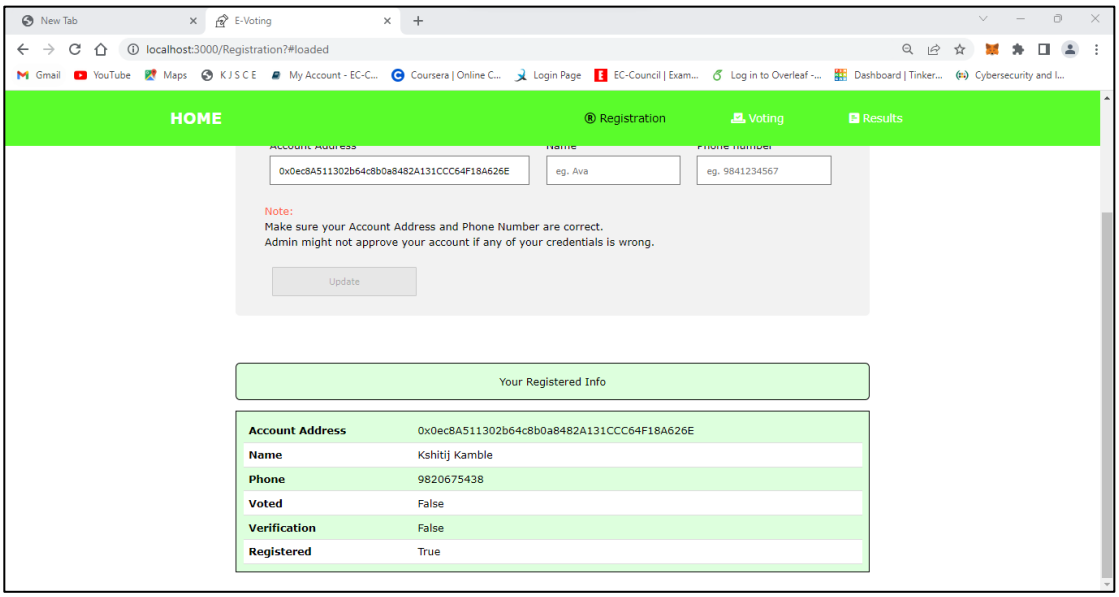

Fig-3.17 Confirming the added information Part 2

After registration phase is over the Admin then has to verify whether the registered voters are valid or not and approve them if the information filled by them is correct and valid:

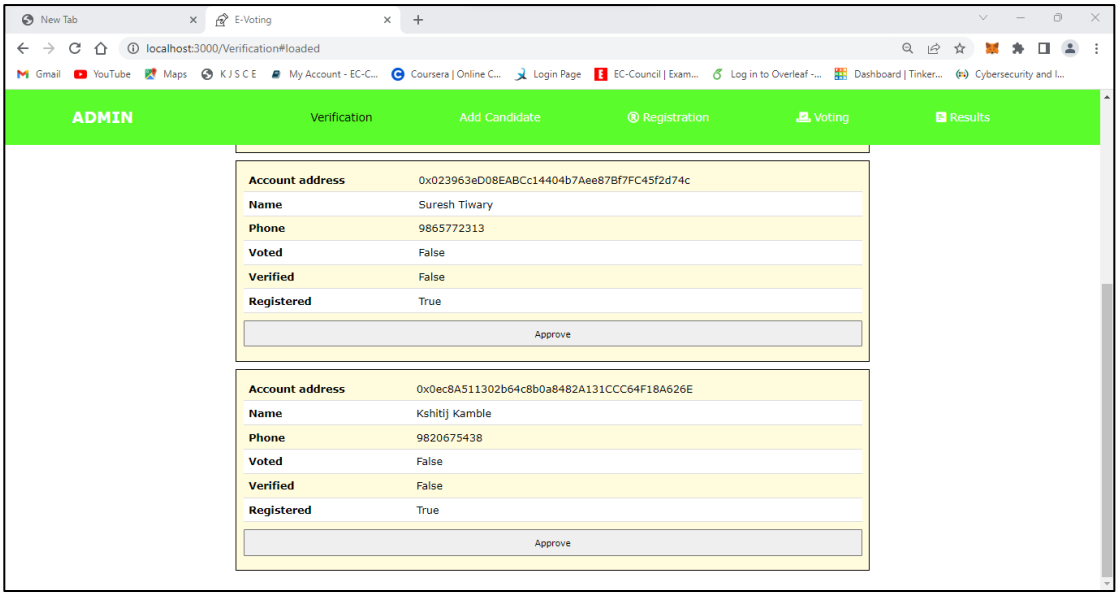

Fig-3.18 User Verification by Admin

After the admin has verified all users the voting phase begins where the users will vote for the candidate of their choice:

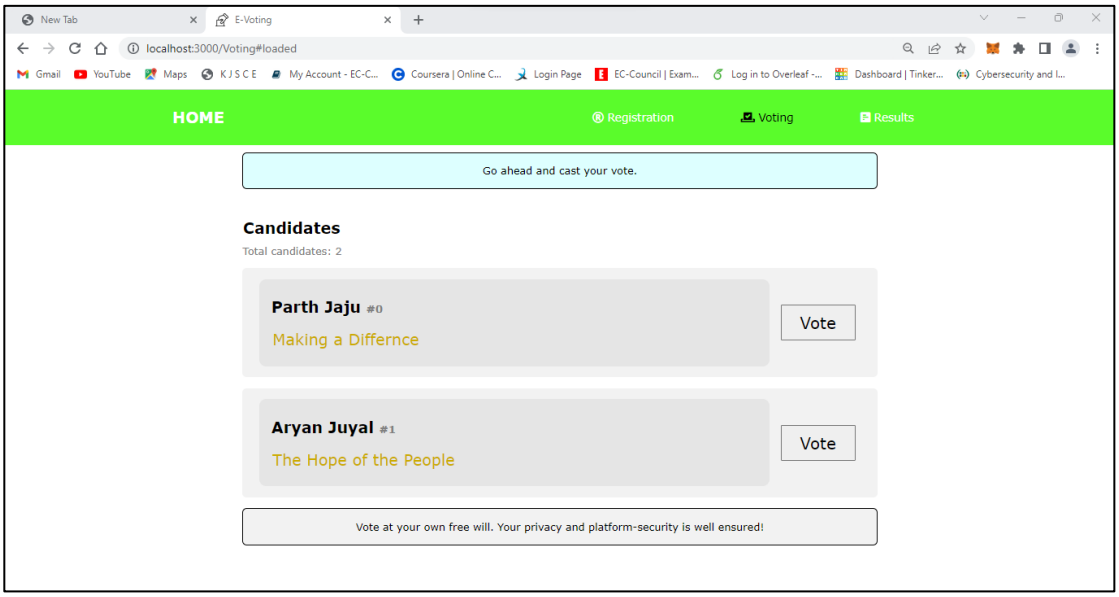

Fig-3.19 Users casting their votes.

After users cast their votes, a "You've casted your vote." message pops up on the screen to signify successful voting and the voting options are henceforth unaccessible:

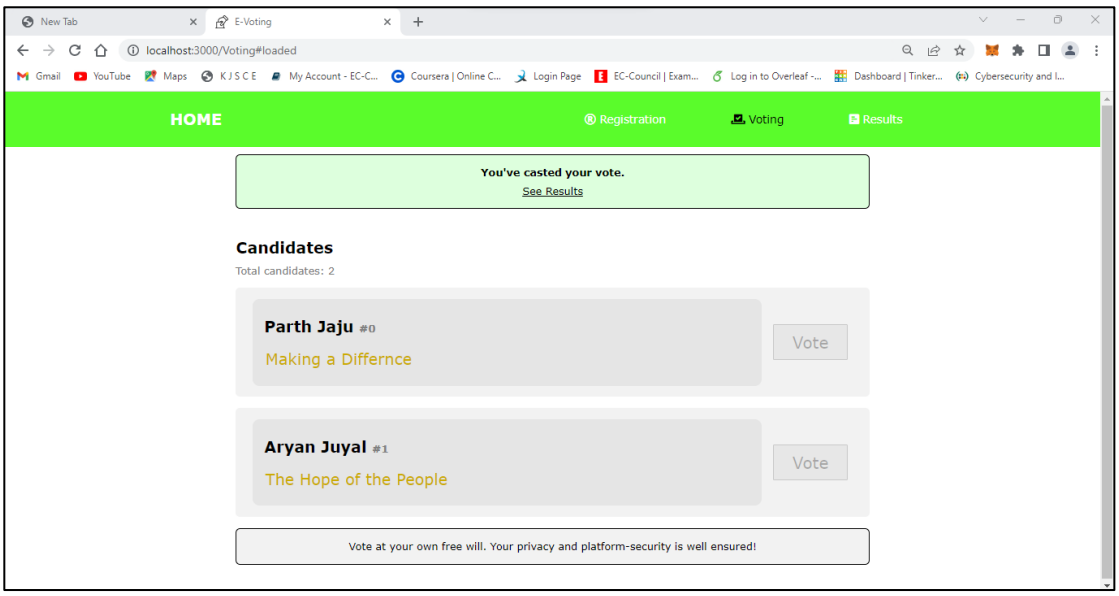

Fig-3.20 Vote casting Confirmation

After all users have voted, the admin has to end the election so that the users can see the election results:

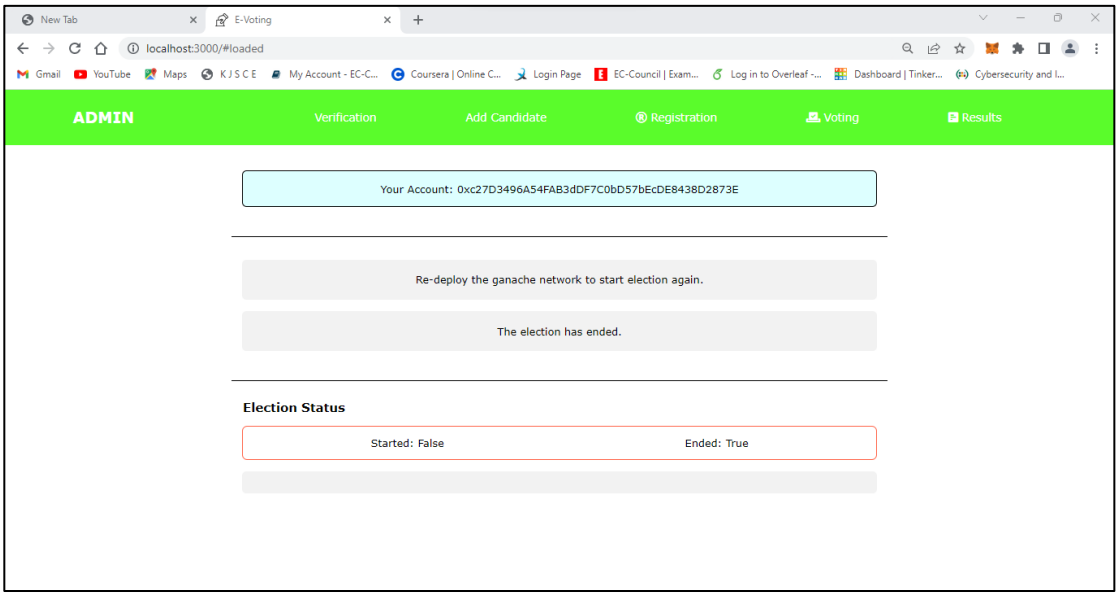

Fig-3.21 Admin ending the election.

Upon successful ending of election:

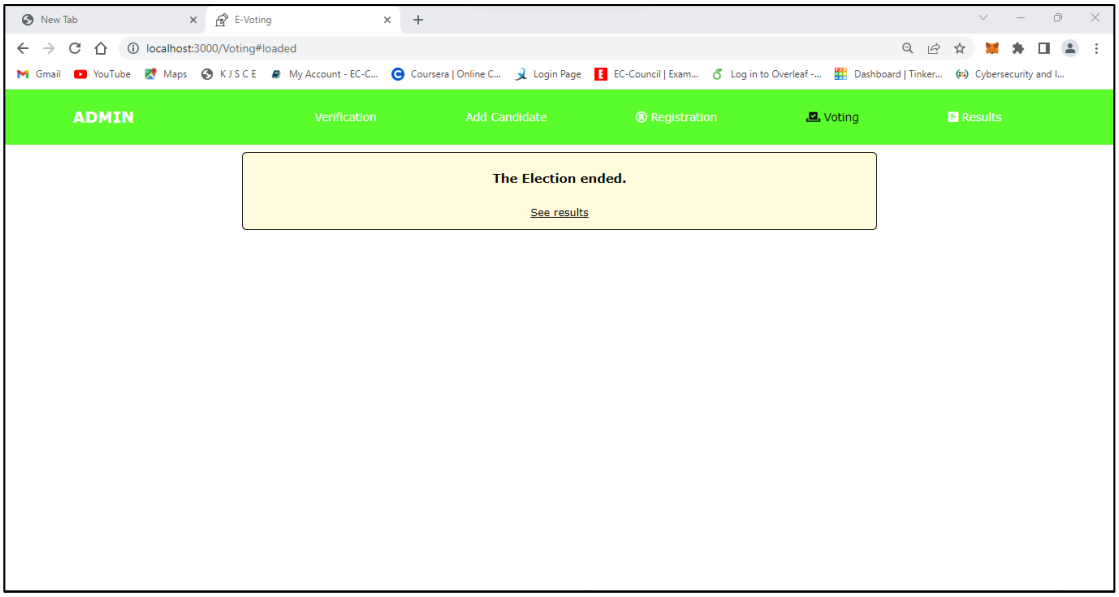

Fig-3.22 User trying to vote after election has ended.

Election results displaying the winner(s) and candidate-wise vote count table:

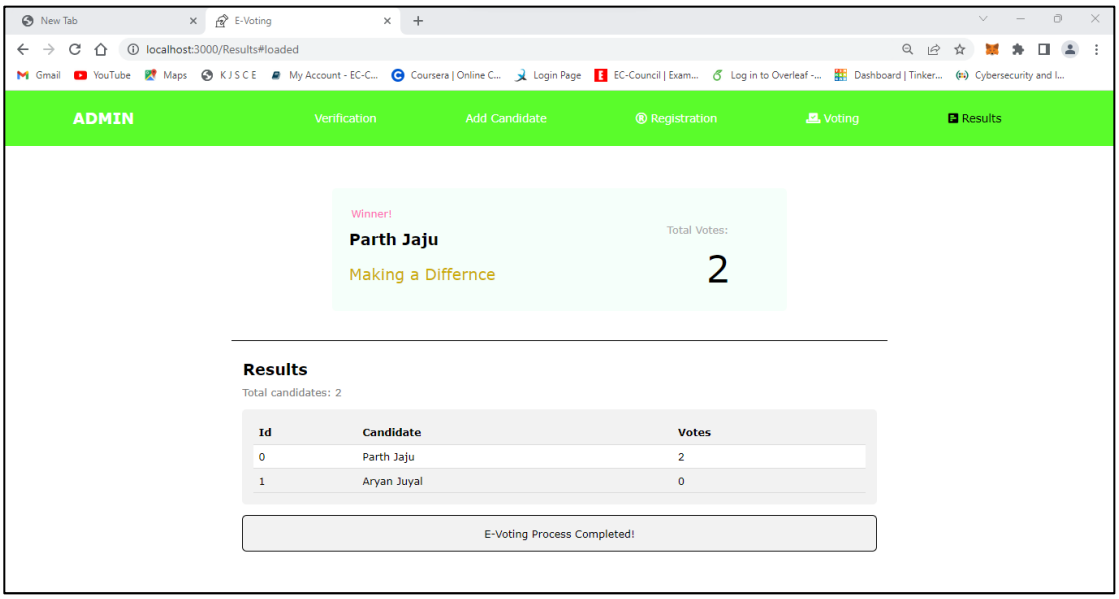

Fig-3.23 Election Results

4. CONCLUSIONS AND LEARNINGS

4.1 Conclusion

In conclusion, the proposed Decentralized Voting System using Blockchain has the potential to overcome the challenges of anonymity, confidentiality, security, and data manipulation during private internal elections.

The system aims to ensure the accuracy of ballots, prevent manipulation and control, and protect the anonymity of voters. The system is user-friendly, making it accessible to all eligible voters, and enabling private institutions to conduct internal elections with confidence in the accuracy and fairness of the results.

The implementation of this decentralized blockchain-based e-voting system could significantly improve the security and reliability of internal elections, ensuring their impartiality and fairness. With proper planning and implementation, a decentralized voting system using blockchain technology can revolutionize the way we vote and ensure a more democratic and transparent process.

4.2 Scope for future Works

Listed below are some of the points that can be taken into consideration in future for improving the project and expanding the scope of the project:

1. Email facility for user verification OTP, vote confirmation, and result publishing: As of yet user verification in our website is done by admin of the election and the vote confirmation is done using a pop-up notification that appears once a user has voted also results are displayed on the website by admin by ending election as such there is no use of email involved in above points but to improve the authenticity and security of the system we can use email verification on the website in which user can be verified by sending an email which will contain an OTP for the user to verify and after he has casted his vote he will get an email confirmation saying you have successfully casted your vote also once the admin has ended the elections and published the results the user will get an email saying the election has ended and all the results of the elections will be

attached in the email so user can access the results through his email also he does not have to open the website every time for the results. The results would be available any time in the user's email.

2. Able to manage more than one election at a time: As of yet on our system there can only be a single election being held at one time so we can introduce the provision of multiple election being held at a time so that the efficiency of the system is improved and user can choose in which election he wants to participate in our website and can vote accordingly**.**

3. Enhanced security using biometric: As of yet in our website the admin has all the information of the user before he registers but to enhance the security aspect of our system, we can introduce biometric authentication of each user maybe by adding facial recognition of each user in our system. Each user before he votes has to undergo facial recognition compulsorily so that we get know that the user casting his vote is authentic and that he is not scamming his way into voting in the election.

4.3 Learnings

We discovered how blockchain technology may improve private institution's internal elections' security, accuracy, and integrity. The Blockchain-based Decentralized Voting System solves anonymity, confidentiality, security, and data manipulation issues these organizations confront during internal elections.

Blockchain technology can secure and anonymize private company elections, limiting manipulation and control. We also found that a user-friendly approach may improve voter turnout and ensure accurate outcomes. We learned Node.js, React, Ganache, Ethereum, Solidity, and MetaMask while building a blockchain-based e-voting system. We implemented smart contracts, created a user interface, and used a local blockchain network. We overcome system correctness, security, and component integration throughout the project.

Trial and error taught us how to use our technology. The initiative shed light on establishing a blockchain-based e-voting system and how blockchain technology may improve election processes. Our insights and expertise will help create decentralized blockchain-based e-voting systems and other blockchain applications.