

A PROJECT REPORT

o

“Voting App using blockchain”

Submitted to
KIIT Deemed to be University

In Partial Fullfilment of the Requirement for the Award of

BACHELOR’S DEGREE IN
COMPUTER SCIENCE AND
COMMUNICATION TECHNOLOGY

BV

SRIJAN MAJUMDER	2029034
ABHISHEK KUMAR SINGH	2029043
AMAN SINGH	2029046
PRAKHAR BHARDWAJ	2029063

UNDER THE GUIDANCE OF
CHANDRA SEKHAR DUBEY



SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024
May 2023

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024



CERTIFICATE

This is to certify that the project entitled
“VOTING APP USING BLOCKCHAIN”
submitted by

SRIJAN MAJUMDER	2029034
ABHISHEK KR. SINGH	2029043
AMAN SINGH	2029046
PRAKHAR BHARDWAJ	2029063

This is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Communication Engineering) at KIIT Deemed to be University Bhubaneswar. This work is done during the year 2022-2023, under our guidance.

Date: 04/05/2023

Chandra Sekhar Dubey
Project Guide

Acknowledgments

We are profoundly grateful to Chandra Shekhar Dubey of Affiliation for his expert guidance and continuous encouragement throughout to see that this project rights its target from its commencement to its completion.

SRIJAN MAJUMDER
ABHISHEK Kr SINGH
AMAN SINGH
PRAKHAR BHARDWAJ

ABSTRACT

This project aimed to develop a secure and transparent voting application using blockchain technology. The application uses a decentralized network of nodes that are responsible for verifying and recording each vote on the blockchain, ensuring that the voting process is tamper-proof and transparent. The application provides a user-friendly interface for voters to cast their votes, and for election administrators to manage the voting process.

The application was built using the Ethereum blockchain, which provides a secure and reliable platform for smart contract development. The smart contracts used in the application were designed to enforce voting rules and ensure that each vote is counted accurately. The application was tested using a simulated election scenario, and the results showed that it is a reliable and efficient solution for conducting secure and transparent elections. Overall, the application has the potential to transform the way elections are conducted, providing a secure and transparent solution for voters and election administrators alike.

Contents

1	Introduction	1
2	Basic Concepts/ Literature Review	2-4
3	Problem Statement / Requirement Specifications	5-6
	3.1 Project Planning.....	5
	3.3 System Architecture (UML) / Block Diagram	6
4	Implementation	9-10
5	Standards Adopted	11-12
6	Conclusion and Future Scope	13
	References	14
	Individual Contribution	15-18
	Plagiarism Report	19

Chapter 1

Introduction

The use of blockchain technology has greatly increased in recent years, focusing on security, transparency, and decentralization. This technology has found many applications in various fields including finance, healthcare, logistics, etc. One such area where blockchain technology is not gaining significant attention is in the voting system. A voting system is an essential component of democracy. And transparency and security of the voting process are crucial for maintaining the integrity of the system. Blockchain technology provides a unique solution to the challenges faced by traditional voting systems by providing a decentralized, immutable, and transparent platform for voting.

The project "Voting System using Blockchain Technology" aims to design and develop a blockchain-based voting system that ensures the accuracy, transparency, and security of the voting process. This system employs the use of blockchain technology to create a tamper-proof record of each vote, which can be audited at any time. The use of blockchain technology also eliminates the need for central authorities or any human intervention(which can result in human error and personal bias), thus making the process more efficient and cost-effective and overall reliable.

The objective of this report is to provide an in-depth analysis of the design and implementation of the voting system using blockchain technology. This report has details of the technical aspects of the system, including the use of smart contracts and consensus algorithms, as well as the benefits and limitations of using blockchain technology in the voting system. Furthermore, the report will also discuss the potential use cases of the system and the challenges that need to be addressed before it can be adopted at scale.

The problem or the shortcoming of the current voting system is that we have to rely on a central authority to conduct the voting fairly and have no way to trace back the vote from where it was cast from. As we have seen an example where in the last Elections, the Congress Party accused the BJP of stealing the vote, but there was no way to verify the authentication of votes. Here Blockchain technology solves this problem by introducing a transparent system where each vote has a unique ID and can be traced back from its castle to its caster.

Chapter 2

Basic Concepts/ Tools Used

React Js:-React JS is a popular JavaScript library for building user interfaces, and it can be used to create the front end of a voting system using blockchain technology. This is a type of javascript library that was used to design the user interface of the voting system. It provides us with a runtime environment as well as the user interface to develop and deploy the UI/UX design of the project

HTML, CSS, REACT-Bootstrap:-HTML, CSS, and REACT-Bootstrap are important technologies for building user interfaces in web development, including in React JS. Here is a brief explanation of each technology. Here HTML provides the skeleton of the UI and Bootstrap style is the skeleton in a manner that is presentable to the user and internally uses CSS for styling.

MDB UI KIT:- It is a comprehensive front-end toolkit for web developers that provides a collection of ready-to-use components, plugins, and templates. It is based on the latest version of Bootstrap and includes additional custom components and features. MDB UI Kit is designed to help developers create modern, responsive, and visually appealing web applications quickly and easily. The kit includes over 4,000 UI components, including buttons, forms, navigation menus, modals, cards, carousels, and many more.

Firestore Auth:-Firestore Authentication is a service provided by Google's Firebase platform that enables developers to easily integrate authentication into their web and mobile applications. With Firestore Authentication, users can sign up and sign in to your app using various authentication methods such as email and password, phone number, Google, Facebook, Twitter, GitHub, Apple, and others.

Firestore:-Firestore is a cloud-based NoSQL document database provided by Google's Firebase platform. It offers real-time synchronization and offline support, which enables developers to build responsive, collaborative applications that work seamlessly across web, mobile, and server platforms. Firestore stores data in documents, collections, and sub-collections, and allows developers to query, update, and delete data in real time with simple APIs. Firestore also offers powerful security rules that allow developers to control who has access to read and write data in their database. Additionally, Firestore provides features such as transactions, batched writes, and automatic scaling,

Firestore hosting: -Firestore Hosting is a service provided by Google's Firestore platform that enables developers to deploy and host web applications quickly and easily. Firestore Hosting uses a global content delivery network (CDN) to serve web content quickly from edge servers around the world. This results in faster page load times, improved user experience, and better search engine optimization (SEO) for web applications. Firestore Hosting supports a range of features including custom domains, SSL encryption, versioning, and rollbacks, which make it easy for developers to deploy, manage, and update their web applications with ease.

Solidity: -Solidity is a programming language used to develop smart contracts on the Ethereum blockchain. It is a high-level, contract-oriented, and statically typed language that is similar to JavaScript and C++. Solidity code is compiled into bytecode, which can be deployed to the Ethereum Virtual Machine (EVM) to run the smart contract. It was in the backend of the project to authenticate the wallet being used

Hadron.js: - Hadron.js is an open-source, cross-platform framework for building desktop applications using web technologies such as HTML, CSS, and JavaScript. It allows developers to use familiar web development tools and workflows to create desktop applications that can run on Windows, macOS, and Linux operating systems. Hadron.js provides a modular architecture that allows developers to easily add and remove features, as well as a powerful plugin system that enables developers to extend the functionality of their applications.

Ether.js: - Ether.js is a popular open-source library for interacting with the Ethereum blockchain using JavaScript. It provides a simple and intuitive API for sending and receiving transactions, querying contract states, and listening to events on the Ethereum blockchain. Ether.js supports both the Ethereum JSON-RPC API and Web3.js API, making it compatible with a wide range of Ethereum clients and dApp frameworks. It was used by the application to verify the transaction of a vote by a unique user

MetaMask: MetaMask is a popular browser extension and mobile application that enables users to interact with decentralized applications (dApps) on the Ethereum blockchain. It provides a simple and secure way to manage Ethereum accounts, including the ability to send and receive Ether and ERC-20 tokens, as well as the ability to sign transactions and messages with private keys. It was used to validate the wallet id in the application to cast a vote and process transaction


Sheet Js: Sheet.js is a popular open-source JavaScript library that provides powerful tools for working with spreadsheet data in various formats, including Excel, CSV, and HTML. It allows developers to read and write spreadsheet data programmatically, as well as to convert data to JSON format, which is later used by react js to display on the web application

Google Forms: Google Forms is a free online tool provided by Google that enables users to create surveys, questionnaires, and other forms quickly and easily. It was in the application to create a mock voting system to assert the importance of blockchain technology in this domain

Chapter 3

Problem Statement / Requirement Specifications

R1: Login

[SignUp](#) [login](#)

Login

Email


Enter your email

Password

Enter your password

Login

R1.1: Signup Page

[SignUp](#) [login](#)

SignUp

Name

Enter your name

Email

Enter your email

Password

Enter your password

Voter Id

Enter your voter id

Metamask Id

Enter your metamask id

SignUp

Description-

Case1: If User details are detected in DB,
Then he is redirected to metamask.

Case 2: If any login details are NOT valid,
Then User is redirected to the Login
page.

Case3: If the login details are valid,
The User is redirected to metamask.

R1.2: Signup Page

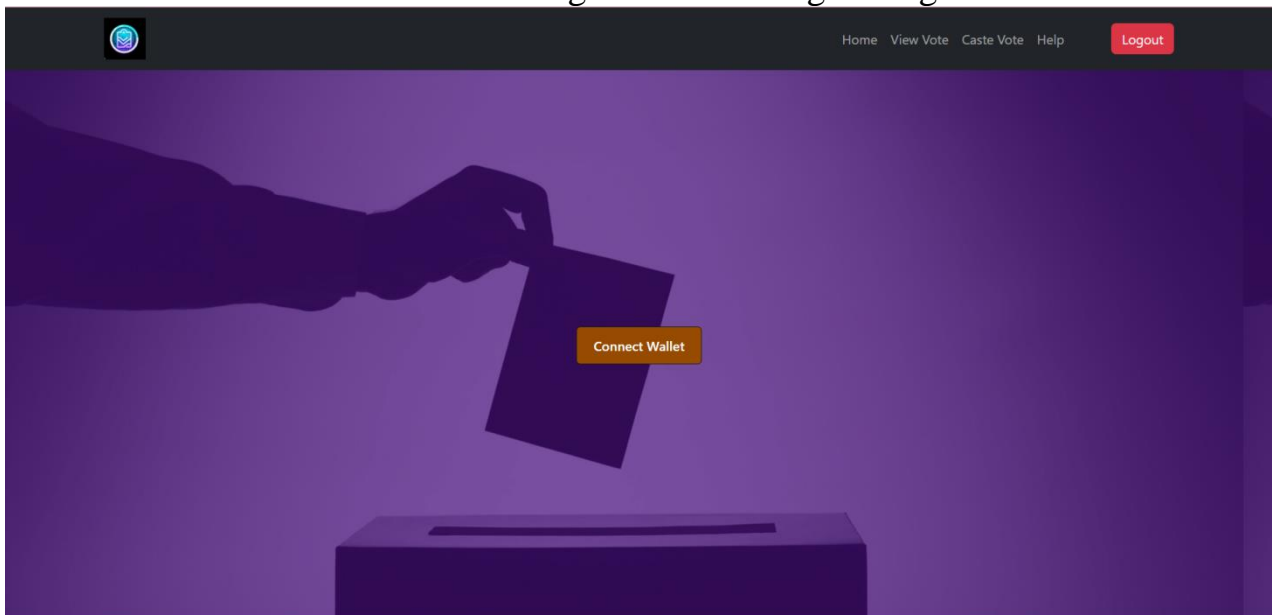
Casel: If User details are NOT valid,
The User is redirected to the Signup page.

Case2: If User details are valid,
The User is redirected to metamask.

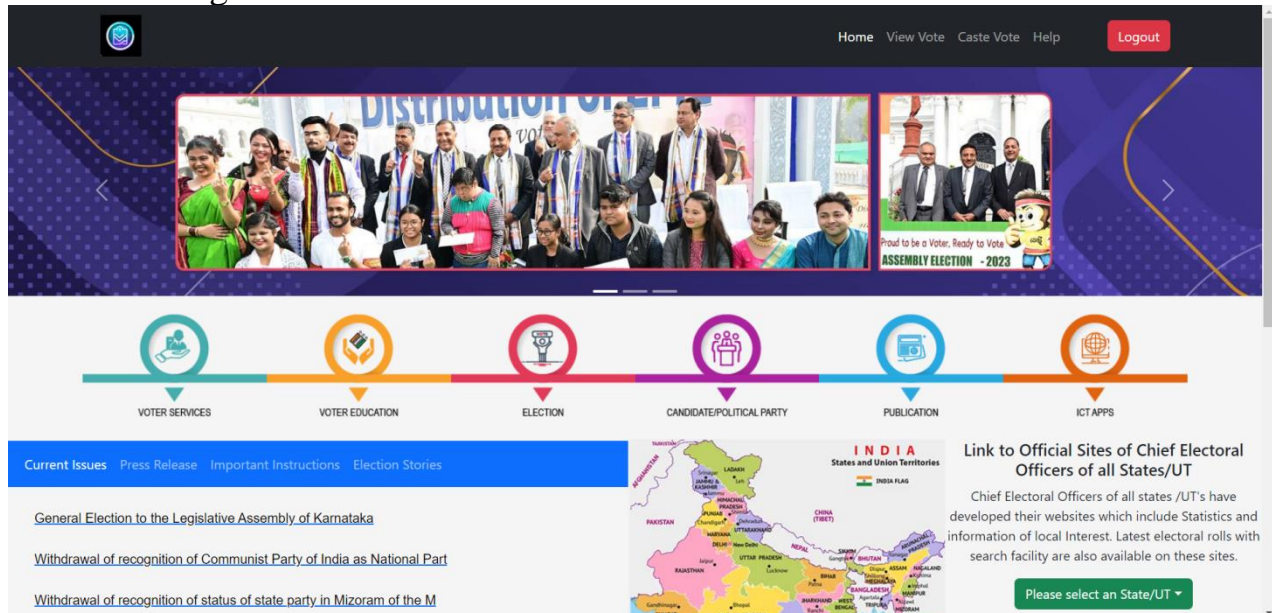
R2: MetaMask page

The user is requested to log in using MetaMask Wallet.

When the User logs in with the metamask wallet, the details are stored in DB and the User is sent to Home Page after checking the login details.



R3: Home Page



R3.1: Help page

The help page redirects Users to a help form they can fill up for any contacting helpdesk

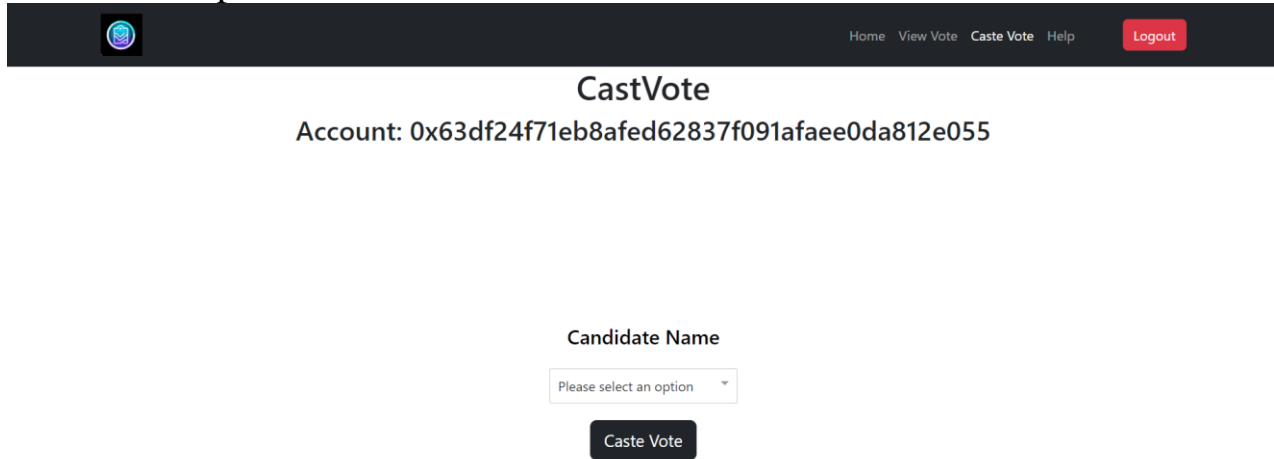
R3.2: ViewVote

This button calls the process GetVotes() in the backend which returns the names of all the candidates and their respective votes on a page.

Voting Table			
#	Candidate Name	Votes	
1	Narendra Modi	3	
2	Arvinda Kejriwal	1	

R3.3: CasteVote

This button calls the process CastVotes() in the backend which has different outputs-



CastVote

Account: 0x63df24f71eb8afed62837f091afae0da812e055

Candidate Name

Please select an option

Caste Vote

The user selects one of the csssssssss from the dropdown menu and select a candidate and then presses caste Vote to caste his/her vote.

R4: FireBase DB

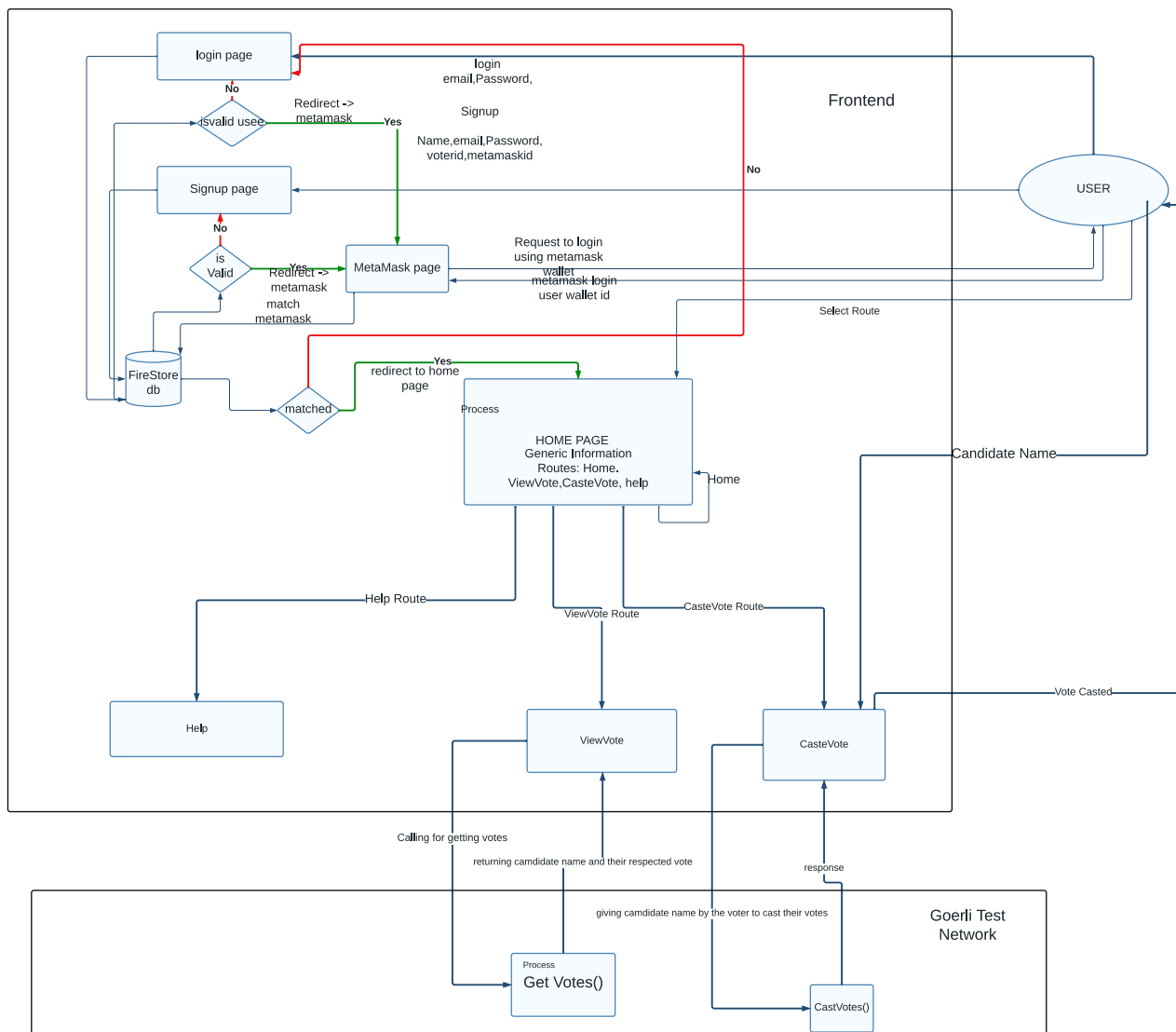
R4.1: Login Details

The Email and Password are stored in this DB and User doesn't need to log in every time.

R4.2: Metamask Details

When the metamask wallet is connected to the System its ID is saved in DB to avoid connecting the wallet every time the user logs in.

UML DIAGRAM



Shortcomings / Limitations:

Only a meta mask wallet can be connected to the system. No other wallet is recognized.

The vote is cast by using cryptocurrency. If there is no currency in the wallet, the vote cannot be cast.

Integrating blockchain technology into existing voting laws and regulations requires careful consideration and collaboration with legal and regulatory bodies to ensure compliance and acceptance.

The digital divide may limit access to blockchain-based voting systems, as not all individuals have equal access to technology or the necessary digital literacy.

Chapter 4

Implementation

Main Voting app

Creating Smart Contracts using Solidity:

Solidity is a contract-oriented programming language used for creating smart contracts on the Ethereum blockchain. We started by creating smart contracts for our voting app using Solidity. We defined the contract structure, added necessary functions such as creating, updating, and deleting proposals, and tested them using a local blockchain network.

Frontend Development with React.js:

React.js are popular frontend development framework used for building web applications. We used React.js for frontend development, including creating pages and components, implementing the necessary functionality and user interfaces, and integrating the app with the backend. React especially provided many components as well as an environment for developing the front of the voting app

Styling with CSS and React-Bootstrap:

Styling our app was important for providing a user-friendly and visually appealing experience. We used CSS and React-Bootstrap for styling, making our app responsive and visually appealing. We used Bootstrap's pre-designed components and customized them as per our requirements. We implemented Bootstrap by installing its modules to the react app and using its pre-defined XML code from the website to design the components such as the navbar, and buttons accordingly

Authentication and Database Management with Firebase:

A well-liked platform for creating online and mobile applications is Firebase. To handle database operations and authentication, we integrated Firebase into our app. Both user authentication and data storage were handled by Firebase Authentication and Firebase Firestore, respectively.

Wallet Integration with Metamask:

For interfacing with the Ethereum blockchain, a well-known wallet is MetaMask. To enable users to engage with smart contracts and access their blockchain wallets, we integrated MetaMask into our app. Voting was done through an Ethereum wallet.

Markup with HTML:

The fundamental structure and layout of web pages are created using the markup language HTML. Our app's core markup, including the design and organization of our web pages, was created using HTML and a combination of CSS and Bootstrap which was integrated to make a presentable application.

Deployment of a smart contract on Goerli with Hardhat and Metamask::

Hardhat can be used to set up a local environment from which we can develop, compile, and deploy the Solidity code to the Goerli test net. Then, we can communicate with the contract and carry out transactions using MetaMask. Hardhat and MetaMask offer crucial development and deployment tools, and by deploying on Goerli, we guarantee the app is secure and decentralized.

Deploying the App on Firestore

Web application deployment can be done using Firestore, a cloud platform. We made our app publicly accessible by deploying it to Firestore. Before deploying, we made sure that all dependencies were properly set up and that our app worked as intended.

Mock Voting app

Creating a Google Form:

To begin, we developed a Google Form with a question and many voting alternatives for the users. After then, an Excel spreadsheet was used to store the form responses.

Data Management with Excel:

The vote data was managed in Excel. The Excel spreadsheet was immediately updated with the results from the Google Form. The data was manipulated using Excel's sorting and filtering features to make it suitable for the online app.

Displaying Data on React App with Sheet JS::

To read and show the vote data on a React app, we used Sheet JS. A JavaScript library called Sheet JS enables the reading and writing of Excel files. To read the Excel file and display the voting information on a React app, we used the Sheet JS package. To read and show the vote data on a React app, we used Sheet JS. A JavaScript library called Sheet JS enables the reading and writing of Excel files. To read the Excel file and display the voting information on a React app, we used the Sheet JS package.

Chapter 5

Standards Adopted

5.1 Design Standards

1. User interface (UI) and user experience (UX) design: The UI design is intuitive and easy to navigate, while the UX design is to provide a positive user experience.
2. Responsive design: The app is optimized for use on all devices, including smartphones, tablets, and desktops, and should be able to adjust to different screen sizes.
3. Security: The app should be designed with security in mind, including features such as two-factor authentication (Google Auth), encryption, and secure data storage.
4. Performance: The app should be designed to be fast and responsive, with minimal loading times and smooth animations.
5. Content management: The app should provide users with the ability to manage their content, including the ability to delete or edit posts, control who can see their content, and report inappropriate content.

5.2 Coding Standards

1. Use consistent coding conventions: Use consistent coding conventions throughout the app to ensure readability and maintainability.
2. Use version control: Use a version control system Git to manage code changes and collaborate with other developers.
3. Follow a modular design pattern: Use a modular design pattern to break the app down into smaller, reusable components that can be easily maintained and tested.

4. Optimize performance: Optimize the app's performance by minimizing database queries, reducing server requests, and optimizing images and media.
5. Use error handling: Use error handling to prevent crashes and provide meaningful error messages to users.
6. Plan for scalability: Plan for scalability by designing the app to handle large amounts of data and traffic. Use techniques like caching.
7. Follow coding best practices: Follow coding best practices such as writing small, reusable functions, using descriptive variable names, and avoiding complex code that is hard to maintain.

5.3 Testing Standards

1. Test in different environments: Test the app in different environments, including different browsers, devices, and operating systems, to ensure that it works consistently across all platforms.
2. Performance test: Test the app for performance by measuring load times, response times, and other performance metrics. This can help to identify performance bottlenecks and optimize the app for better performance.
3. Use real user data: Use real user data in testing to ensure that the app functions as expected with realistic data. This can help to identify issues that may not be apparent with test data.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

In conclusion, the "Voting System using Blockchain Technology" project has shown the potential of blockchain to enhance the accuracy, transparency, and security of the voting process. Blockchain provides a decentralized and immutable platform, addressing the shortcomings of traditional voting systems. It offers transparency, traceability, and tamper-proof records of votes, eliminating the need for central authorities and reducing the risk of fraud.

6.2 Future Scope

The future scope of this project includes refining the system based on real-world data, piloting the technology, and addressing the digital divide to ensure inclusivity. Additionally, efforts should be made to enhance privacy measures while maintaining transparency and auditability.

Overall, blockchain technology has the potential to transform the voting process, providing a robust and trusted platform for conducting elections and promoting democratic values.

References

- [1] <https://legacy.reactjs.org/docs/getting-started.html>.
- [2] <https://www.w3schools.com/cssrefindex.php>
- [3] https://www.w3schools.com/html/html_intro.asp
- [4] <https://docs.soliditylang.org/en/v0.8.19/index.html>
- [5] <https://hardhat.org/docs>
- [6] <https://docs.ethers.org/v5/>
- [7] <https://react-bootstrap.github.io/>
- [8] <https://mdbootstrap.com/docs/react/getting-started/installation>

INDIVIDUAL CONTRIBUTION REPORT:

Voting DApp
Srijan Majumdar
2029034

Abstract: This project involved creating a smart contract using Hardhat and deploying it on the Goerli test network. The contract was connected to Ether.js and Metamask for easy interaction with the network. The objective of the project was to demonstrate the creation and deployment of a functional smart contract using popular blockchain development tools.

Individual contribution and findings: As a member of the project group, my primary responsibility was to design the user interface of the voting application using CSS and React Bootstrap. I worked closely with the front-end developers to ensure that the application had a user-friendly and visually appealing interface. My role involved creating mockups, wireframes, and design assets, as well as implementing the final design using CSS and React Bootstrap. In terms of planning, I first analyzed the project requirements and user needs to come up with an appropriate design strategy. I then created a detailed design plan

Individual contribution to project report preparation: In the project group, I was responsible for creating the abstract section and outlining the tools used in developing the voting application. In the abstract, I summarized the main objective of the project and provided an overview of the blockchain-based voting application we developed. I ensured that the abstract was concise and informative, and highlighted the unique features of the application.

Full Signature of Supervisor:

Full signature of the student:

VOTING DAPP

Abhishek Kumar Singh

2029043

Abstract: This project involved creating a smart contract using Hardhat and deploying it on the Goerli test network. The contract was connected to Ether.js and Metamask for easy interaction with the network. The objective of the project was to demonstrate the creation and deployment of a functional smart contract using popular blockchain development tools.

Individual contribution and findings: As part of the project team, my contribution focused on creating all front-end functions and ensuring seamless communication between the frontend and Blockchain. Design and implement user-friendly interfaces that seamlessly integrated it with blockchain. Work involved creating pages, and routing them correction. Also creating a function to fetch votes from the blockchain. Form submission, handling voting to Blockchain.

kkkkkkk was analyzing the project requirements and creating a detailed plan for the front-end development process. I then created wireframes and mockups to visualize the user interface and ensure that all the necessary functionality was accounted for. Once the design was finalized, I began coding the frontend using HTML, CSS, and JavaScript frameworks such as React and Angular.

Individual contribution to project report preparation:

As a member of the project group, my contribution to the project report was focused on providing the backend report and UML diagram. In addition to creating all frontend functions and connecting the backend to the frontend, I also played a crucial role in preparing the project report. This involved creating detailed documentation of the frontend design, including UML diagrams and images for all pages.

To begin, I created a UML diagram that provided a high-level overview of the frontend architecture. This diagram helped to visualize the relationships between different frontend components, such as the user interface, data models, and APIs. It also helped to identify potential design issues and optimize the frontend architecture for scalability and maintainability.

Full Signature of Supervisor:

.....

Full signature of the student:

.....

Voting App using Blockchain

Aman Singh

2029046

Abstract: This project involved creating a smart contract using Hardhat and deploying it on the Goerli test network. The contract was connected to Ether.js and Metamask for easy interaction with the network. The objective of the project was to demonstrate the creation and deployment of a functional smart contract using popular blockchain development tools.

Individual contribution and findings: As a member of the project group, my primary contribution was to developing the backend of the smart contract and deploying it on the Goerli test network. I also played a key role in connecting the contract to both the frontend and backend of the project.

To start, I worked closely with other members of the group to plan out the structure and logic of the smart contract. This involved defining the functions and variables that would be needed to meet the project requirements, as well as determining the overall architecture of the contract.

Once the initial planning phase was complete, I worked independently to implement and test the backend code for the contract. This involved writing Solidity code and using Hardhat to test and debug the contract on the local development environment.

This involved configuring the necessary settings to ensure that the contract was properly connected to the network and could be accessed by the frontend.

Finally, I worked closely with other members of the group to connect the backend of the project to the deployed contract. This involved using Ether.js to interact with the contract functions and ensure that all the necessary data was being passed between the contract and the frontend.

Individual contribution to project report preparation: As a member of the project group, my contribution to the project report was focused on providing the backend report. Specifically, I was responsible for outlining the architecture and logic of the smart contract, as well as providing detailed descriptions of the functions and variables used in the contract.

After the initial planning phase was complete, I created a detailed report outlining the back end of the project. This report included a detailed description of each of the functions and variables used in the smart contract, as well as a step-by-step explanation of how they worked together to achieve the desired outcome.

Full Signature of Supervisor:
student:

Full signature of the

Voting App
PRAKHAR BHARDWAJ
2029063

Abstract: This project involved creating a smart contract using Hardhat and deploying it on the Goerli test network. The contract was connected to Ether.js and Metamask for easy interaction with the network. The objective of the project was to demonstrate the creation and deployment of a functional smart contract using popular blockchain development tools.

Individual contribution and findings: I integrated Firebase into the project, configuring authentication and establishing the database structure for optimal organization of voting-related data. Utilizing React hooks such as useState and useEffect, I streamlined development and improved the user interface. For instance, I managed the user authentication state efficiently using the useState hook and fetched real-time data from Firebase using the useeffect hook. Additionally, I implemented dynamic form handling using React hooks, managing form inputs and ensuring data integrity through validation logic. These contributions enhanced the responsiveness, user experience, and intuitive nature of the voting system.

Individual contribution to project report preparation: I played a role in creating the System Requirements Specification (SRS) section of the report. I actively participated in requirement-gathering sessions and collaborated with team members to define the system's functionalities. I also identified and documented the limitations and shortcomings of the voting system

Full Signature of Supervisor:

Full signature of the student:

TURNITIN PLAGIARISM REPORT
(This report is mandatory for all the projects and plagiarism
must be below 25%)