A Project Report

on

SECURE DIGITAL VOTING SYSTEM USING BLOCKCHAIN TECHNOLOGY

Submitted in partial fulfillment of the requirements

for the award of degree of

BACHELOR OF TECHNOLOGY

in

Information Technology

by

Kukka Supriya(20WH1A1225)

Katipally Anvitha(20WH1A1228)

Singari Ashwitha(20WH1A1235)

Eudulakanti Mamatha(21WH5A1206)

Under the esteemed guidance of

Ms. M. Sudha Rani

Assistant Professor

Department of Information Technology

BVRIT HYDERABAD College of Engineering for Women

Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)

(NAAC 'A' Grade & NBA Accredited- ECE, EEE, CSE & IT)

June, 2024

DECLARATION

We hereby declare that the work presented in this project entitled **"SECURE DIGITAL VOTING SYSTEM USING BLOCKCHAIN TECHNOLOGY"** submitted towards completion of in IV year II sem of B.Tech IT at "BVRIT HYDERABAD College of Engineering for Women",Hyderabd is an authentic record of our original work carried out under the esteemed guidance of Ms. M. Sudha Rani, Assistant Professor, Department of Information Technology.

Kukka Supriya(20WH1A1225)

Katipally Anvitha(20WH1A1228)

Singari Ashwitha(20WH1A1235)

Eudulakanti Mamatha(21WH5A1206)

VISHNU
UNIVERSAL LEARNING
BVRITH
Estd. 2012

BVRIT HYDERABAD

College of Engineering for Women

Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

(NAAC 'A' Grade & NBA Accredited- ECE, EEE, CSE & IT)

CERTIFICATE

This is to certify that the Project report on **"SECURE DIGITAL VOTING SYSTEM USING BLOCKCHAIN TECHNOLOGY"** is a bonafide work carried out by **Kukka Supriya (20WH1A1225), Katipally Anvitha (20WH1A1228), Singari Ashwitha (20WH1A1235)** and **Eudulakanti Mamatha (21WH5A1206)** in the partial fulfillment for the award of B.Tech degree in **Information Technology, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad** affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision. The results embodied in the project work have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide
Ms. M. Sudha Rani
Assistant Professor
Department of IT

Head of the Department
Dr. Aruna Rao S L
Professor & HOD
Department of IT

External Examiner

ACKNOWLEDGEMENT

We would like to express our profound gratitude and thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women** for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. Aruna Rao S L, Professor & Head, Department of IT, BVRIT HYDERABAD College of Engineering for Women** for all the timely support, constant guidance and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Ms. M. Sudha Rani, Assistant Professor, Department of IT, BVRIT HYDERABAD College of Engineering for Women** for her constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank our Project Coordinators **Dr. P. Kayal, Associate Professor and Ms. K. Bharathi, Associate Professor**, all the faculty and staff of Department of IT who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

Kukka Supriya(20WH1A1225)

Katipally Anvitha(20WH1A1228)

Singari Ashwitha(20WH1A1235)

Eudulakanti Mamatha(21WH5A1206)

ABSTRACT

Voting is a very important event organized in all countries by secret ballot as well as Electronic Voting Machines. Such processes have many drawbacks such as vote disruption, low turnout, time-consuming counting, hacking of electronic voting machines, and so on. To overcome this problem we will use blockchain technology which provides anonymity for voters. A blockchain is a distributed, digitized and consensus-based secure information storage mechanism. Blockchain technology moves in the direction of persistent revolution and change. A voting system that uses blockchain technology provides a secure and transparent environment for decisions, where voters can vote only once and vote will not be interrupted. Due to this online procedure percentage of voting will also increase. A distributed online electronic voting system implementation using the blockchain as a service is an innovative blockchain-based online e-voting system that tackles some of the limitations in the current systems and assesses some of the well-known blockchain frameworks in order to build such a system. In order to guarantee the validity and reliability of the voting process, smart contracts are used to control the voting process and a face detection system and cryptographic techniques are integrated to guarantee the confidentiality and privacy of voter data.A blockchain based online voting system is a promising approach to enable secure and transparent voting.

Keywords: Blockchain, Ethereum, Solidity, Ganache, SHA-256, File Storage, Face API(Open CV).

LIST OF FIGURES

LIST OF ABBREVATIONS

Abbrevation	Meaning
CNN	Convolutional Neural Network
OpenCV	Open Source Computer Vision
Web3	World Wide Web
SHA	Secure Hash Algorithm

CONTENTS

1. INTRODUCTION

A fair and transparent election is need for today's society according to the today's social environment. Currently, a lot of countries use the traditional ballot system, which requires centralized control with a trusted party for conduction of the voting process, and recording and counting of the vote ballots by the trusted third party. The democratic process lies at the heart of any society, serving as the cornerstone of governance and ensuring the representation of citizens' voices. However, traditional voting systems are often plagued by challenges such as fraud, tampering, and inefficiencies. In recent years, the emergence of blockchain technology has offered a potential solution to these issues by providing a secure, transparent, and decentralized framework for conducting elections.

Blockchain, originally conceived as the underlying technology behind cryptocurrencies, is essentially a distributed ledger that records transactions across a network of computers. Blockchain uses add and append only strategy. We cannot delete the existing data in blockchain. Blockchain uses peer to peer network systems. Each transaction, or "block,ïs cryptographically linked to the previous one, forming a chain of blocks that is immutable and transparent. This inherent transparency and security make blockchain an ideal candidate for revolutionizing the electoral process.

The secure digital voting system operates through a distributed network of nodes each maintain a copy of the blockchain ledger. When a voter casts their ballot, a unique cryptographic hash is generated and recorded on the blockchain using a smart contract. The smart contract ensures that the vote is securely stored and cannot be altered or manipulated without detection. Blockchain technology combined with smart contracts enables transparent and auditable elections by providing a verifiable trail of every vote cast.

Through cryptographic techniques, voters can verify that their vote has been recorded accurately without compromising their anonymity. This transparency fosters trust among participants and ensures the legitimacy of election outcomes. Blockchain based online voting application, utilizing decen-

tralization, cryptography, and consensus mechanisms to overcome traditional voting system limitations. It aims to enhance integrity, security, and accessibility in the voting process.

Furthermore, the project aligns with the Sustainable Development Goal(SDG) of creating Industry, Innovation increases transparency and contributes efficiency in electoral process. Online voting can encourage more people to participate in elections, potentially increasing voter turnout and make it easier for citizens who are geographically distant.

1.1. Objective

i) Utilizes blockchain's immutability to create a tamper-resistant environment, ensuring that once votes are recorded, they cannot be altered or manipulated.

ii) Provides a transparent voting process through an accessible ledger that allows the public to scrutinize the voting records. This transparency enhances trust in the electoral process.

iii) Prioritizes voter privacy through cryptographic techniques, ensuring that individual votes remain confidential and cannot be linked back to specific voters. language.

iv) Minimizes vulnerabilities by decentralizing the voting system, which reduces the risk of centralized points of failure or manipulation. This enhances overall security.

V) Promotes accessibility by offering remote voting options, making it easier for individuals to participate in the electoral process regardless of their location or mobility.

vi) Ensures inclusivity by providing remote voting options, which allows individuals who may face barriers to physical voting locations to still participate in elections.

vii) Enables swift and accurate processing of votes, leading to timely election results. Real-time processing enhances the efficiency of the electoral process.

viii) Utilizing Scikit-learn, post-election audits can leverage transparent blockchain records to verify result accuracy, ensuring electoral integrity. This fosters voter trust by addressing fraud concerns and bolstering cybersecurity, vital for legitimizing the electoral process.

1.2. Problem Definition

The existing voting system in India, primarily reliant on electronic voting machines (EVMs), faces significant challenges, including identity theft, potential for false votes, and concerns about corruption. Recent claims against the current system have raised doubts about its reliability and integrity. Moreover, the logistical complexities, long travel times to polling places,infrastructural costs,and security expenses associated with traditional voting methods present substantial hurdles. The aim of this project is to address these issues by developing a blockchain-based electronic voting system that ensures the security,transparency,and trustworthiness of the electoral process. The need for a more efficient and secure voting process, and the desire to reduce the time and costs associated with traditional voting methods.

1.3. Modules

1) Register
2) Face-Capture
3) Login
4) Vote Casting
5) Face Proctoring
6) Display Result

2. LITERATURE SURVEY

The rise of online voting has ignited a crucial debate in India, a nation brimming with aspirations for democratic participation. While the promise of accessibility and flexibility shines brightly, navigating this new technology comes with its own set of challenges. This research paper under consideration investigates the development and deployment of an online voting system leveraging blockchain technology.[1] The paper thoroughly explores the system's architecture, elucidating the role of smart contracts in orchestrating the voting process. Additionally, it highlights the integration of a face detection system and cryptographic techniques aimed at safe guarding voter data confidentiality and privacy.The focus is on the creation and implementation of an online voting system utilizing blockchain technology. The architecture of the system is thoroughly examined, with a specific emphasis on the use of smart contracts to manage the voting process. Furthermore, the paper discusses the incorporation of a face detection system and cryptographic techniques, which play a crucial role in ensuring the security and privacy of voter data. The integration of these components aims to enhance the overall integrity and confidentiality of the voting process. The comprehensive exploration of these aspects contributes to the paper's goal of establishing a secure and transparent online voting system.

This study suggests an innovative blockchain-based online e-voting system that tackles some of the limitations in the current systems and assesses some of the well-known blockchain frameworks in order to build such a system through the description of a case study, [2] specifically the process of an election time and the blockchain-based application system, which improves security and lowers the cost of hosting a national election and votes will be counted and the results will be reported immediately. The project suggests an online voting mechanism that makes use of the Ethereum Blockchain. It involves building a user wallet based on the Ethereum Blockchain, which will provide voters with a verified and highly secure personal ID. To ensure anonymity, voters can choose to cast their ballots by transferring tokens from their wallet to the candidate's wallet, thus maintaining the confidentiality of their choices.Any geographic location may vote for the voter's designated constituency. The anonymity of voters is further protected through blockchain, which is nevertheless accessible to the whole

public. The suggested voting mechanism is cost-effective and aims to utilize a tamper-proof blockchain to ensure that any modifications made to the voting process, stay unaltered and safe, regardless of voter intent or interference from other parties

This paper underscores the profound impact of blockchain technology on the con temporary professional landscape, characterizing it as a pivotal discovery with far-reaching implications. [3]Positioned as a distributed, digitized, and consensus based secure information storage mechanism, blockchain is depicted as a force propelling persistent revolution and change. The surge in blockchain technology over recent years has prompted a meticulous examination by scholars and spe cialists seeking innovative applications across diverse domains.Amid this tech nological upswing, the paper focuses on the burgeoning opportunities brought about by blockchain, particularly in the realm of e-voting applications. It con ducts a systematic review of emerging blockchain-based e-voting systems, delv ing into the details of open research matters within this rapidly evolving field.A key aspect highlighted is the necessity for frameworks to undergo enhancements to address existing reservations and challenges, especially concerning their appli cation in voting systems.[4] The conclusion drawn is that, despite the promise of blockchain in e-voting, further refinements are imperative to ensure the effective integration of these systems, emphasizing the dynamic nature of research and de velopment in this domain.

This paper addresses the formidable challenge of constructing a secure voting system with a focus on fairness and privacy, this implementation paper explores the application of blockchain as a service.[5] The primary aim is to assess the feasibility of employing a decentralized architecture for the implementation of distributed electronic voting systems. The objective is clear: to establish a voting scheme that is not only open, fair, and independently verifiable but also inher ently secure.The proposed solution hinges on implementing a protocol designed to achieve fundamental e-voting properties, coupled with a significant degree of decentralization. A noteworthy feature is the system's flexibility, allowing voters to change or update their votes. Through experimentation, the results affirm the benefits of this

proposed solution for both existing and future voting systems.In essence, the paper contributes to the ongoing discourse on the evolution of secure and fair voting systems by demonstrating the practical application of blockchain in the service of decentralized electronic voting. The positive experimental results underscore the potential effectiveness of this approach, signaling promise for the improvement of contemporary and upcoming voting processes.

This paper distinguishes identical twins using their face images in biometrics. The goal of this study is to construct a biometric system that is able to give the cor rect matching decision for the recognition of identical twins.[6] It proposed a method that uses feature-level fusion, score-level fusion, and decision-level fu sion with principal component analysis, histogram of oriented gradients, and lo cal binary patterns feature extractors. In the experiments, face images of identical twins from ND-TWINS-2009-2010 database were used. The results show that the proposed method is better than the state-of-the-art methods for distinguishing identical twins. Variations in illumination, expression, gender, and age of identi cal twins' faces were also considered in this study. The experimental results of all variation cases demonstrated that the most effective method to distinguish identi cal twins is the proposed method compared to the other approaches implemented in this study. The lowest equal error rates of identical twins recognition that are achieved using the proposed method are 2.07smiling expression, and 2.24.7ditions. Additionally, the proposed method is compared with the other methods for non-twins using the same database and standard FERET subsets. The results achieved by the proposed method for non-twins identification are also better than all the other methods under expression, illumination, and aging variations.

3. SYSTEM DESIGN

3.1 Existing System

The existing system for conducting elections typically relies on traditional methods, such as paper-based ballots or electronic voting machines. While these systems have been used for decades, they are not without their shortcomings. Paper-based systems are prone to errors in ballot counting, manual tampering, and logistical challenges in transporting and securing physical ballots. On the other hand, electronic voting machines, while offering some level of efficiency, have raised concerns over security vulnerabilities, potential manipulation of electronic data, and lack of transparency in the voting process.

Furthermore, traditional voting systems often face challenges in ensuring accessibility for all voters, particularly those with disabilities or living in remote areas. Additionally, the centralized nature of these systems can give rise to issues of trust and transparency, as the integrity of the electoral process relies heavily on the competence and impartiality of election authorities.

In recent years, there has been growing interest in exploring alternative approaches to conducting elections, with blockchain technology emerging as a promising solution. Blockchain offers a decentralized and immutable ledger that records transactions in a transparent and tamper-resistant manner. By leveraging cryptographic techniques and consensus mechanisms, blockchain-based voting systems aim to address many of the shortcomings of traditional voting systems.

These blockchain-based systems enable voters to cast their ballots securely and anonymously from any location with an internet connection. Each vote is recorded on the blockchain, ensuring transparency and integrity throughout the voting process. Furthermore, the decentralized nature of blockchain eliminates the need for a central authority, reducing risk of manipulation or interference.

Moreover, blockchain-based voting systems can enhance accessibility by providing alternative voting methods, such as mobile or online voting, thereby accommodating voters who may face challenges in accessing traditional polling stations. Additionally, blockchain technology can offer enhanced security measures, such as end-to-end encryption and multi-factor authentication, to safeguard against unauthorized access and tampering of voting data.

Despite the potential benefits, the adoption of blockchain-based voting systems still faces challenges, including scalability issues, regulatory hurdles, and public skepticism. However, ongoing research and pilot projects are demonstrating the feasibility and potential of blockchain technology to revolutionize the way elections are conducted, paving the way for more inclusive, transparent, and secure electoral processes.

3.2 Proposed System

The proposed blockchain-based online voting application presents a transformative approach to addressing the shortcomings of traditional voting systems. At its core, the system leverages blockchain technology to create a decentralized and immutable ledger that ensures the integrity and security of the voting process. By decentralizing the infrastructure and removing the reliance on a central authority, the proposed system mitigates the risk of manipulation, fraud, and coercion that often plague traditional voting systems.

One of the key features of the proposed system is transparency. Through the use of blockchain technology, the entire voting process becomes transparent and auditable in real-time. Each vote is recorded as a transaction on the blockchain, providing an immutable and tamper-resistant record of the election results. This transparency fosters trust among voters, election authorities, and other stakeholders, as anyone can verify the integrity of the electoral process and independently audit the results.

Moreover, the proposed system prioritizes security by leveraging the cryptographic features of blockchain technology. Each vote is encrypted and securely stored on the blockchain, safeguarding it against unauthorized access or tampering. The decentralized nature of the blockchain network further enhances security by eliminating single points of failure and reducing the risk of cyber attacks or data breaches. Additionally, the use of smart contracts ensures that the voting process is executed according to predefined rules and protocols, minimizing the potential for human error or manipulation.

In terms of accessibility, the proposed system offers a convenient and user-friendly voting experience. Through the use of an online platform, voters can securely cast their ballots from any location with an internet connection, eliminating the need to physically visit polling stations. This accessibility is par-

ticularly beneficial for individuals with mobility or transportation constraints, as well as those living in remote areas. Moreover, the system incorporates measures to ensure inclusivity, such as support for multiple languages and accommodations for voters with disabilities.

Privacy is another critical aspect of the proposed system. While maintaining transparency and integrity, the system also prioritizes voter privacy by employing cryptographic techniques and zero-knowledge proofs. These techniques allow voters to anonymously cast their ballots without revealing their identities or voting preferences, ensuring the confidentiality of their choices while still enabling the verification of the overall election outcome.

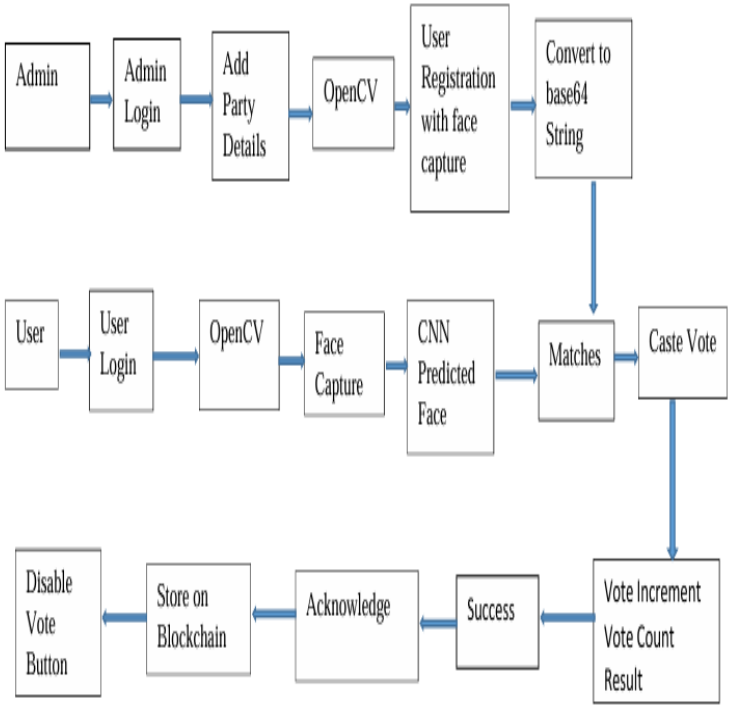

Figure 3.2.1: Proposed System Architecture

3.3 Technologies

Blockchain: A digital voting system leveraging blockchain technology represents a transformative approach to addressing longstanding issues in traditional voting systems. By harnessing the decentralized and secure nature of blockchain, this innovative solution aims to enhance the integrity, transparency, and efficiency of the electoral process.

Cryptography: In a blockchain-based digital voting system, cryptography is the cornerstone of a secure and trustworthy electoral process. Cryptographic techniques are strategically employed to address critical aspects such as voter authentication, ballot confidentiality, data integrity, and secure computation.

Smart Contracts: Smart contracts in a digital voting system using blockchain technology offer a transformative approach to conduct secure, transparent, and tamper-resistant elections. These self-executing contracts, powered by blockchain's decentralized and transparent nature, automate various aspects of the voting pro cess, providing benefits such as enhanced security, increased efficiency, and greater trust in electoral outcomes.

3.4 UML Diagram

UML is an acronym that stands for Unified Modeling Language. Simply put, UML is a modern approach to modeling and documenting software. In fact, it's one of the most popular business process modeling techniques. It is based on diagrammatic representations of software components. A UML diagram with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

3.4.1 Class Diagram:

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an ïs-a.ºr "has-arelationship. Each

class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed ”methods.of the class. Apart from this, each class may have certain .attributes”that uniquely.

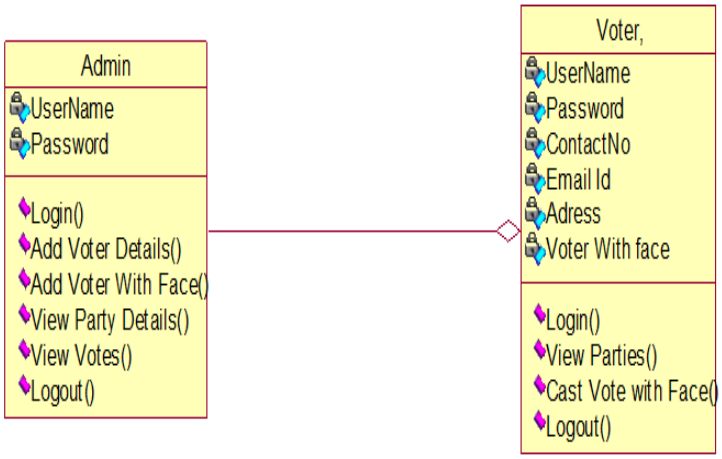

Figure 3.4.1: Class Diagram

3.4.2 Use Case Diagram:

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system’s functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system. The following picture depicts the Use case diagram of our model, ML Powered Text Auto-completion and Generation.

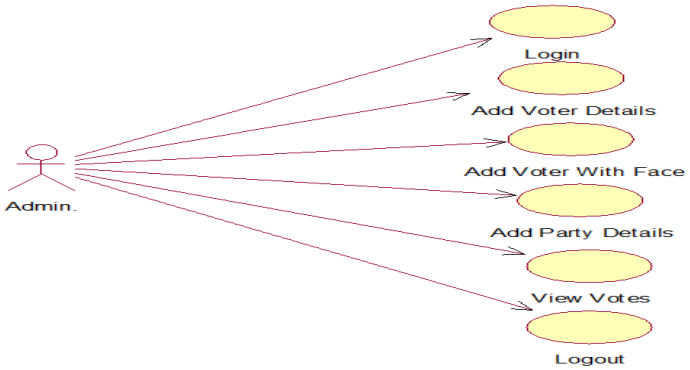

Figure 3.4.2: Use Case Diagram

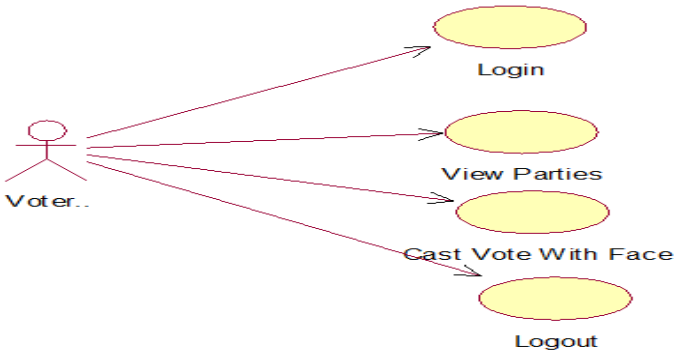

Figure 3.4.2: Use Case Diagram

3.4.3 Sequence Diagram:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing ”messages”.

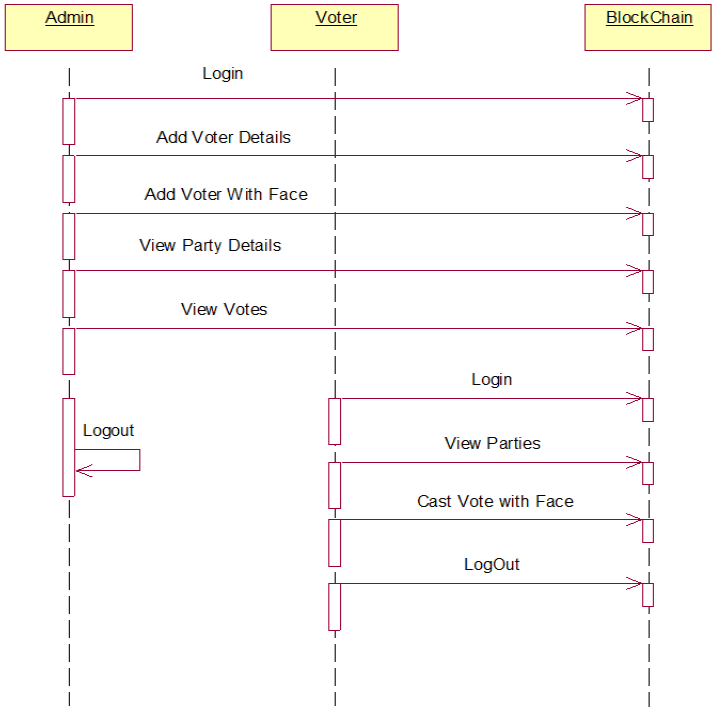

Figure 3.3.2.1 : Sequence Diagram

3.4.4 Collaborative Diagram:

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.

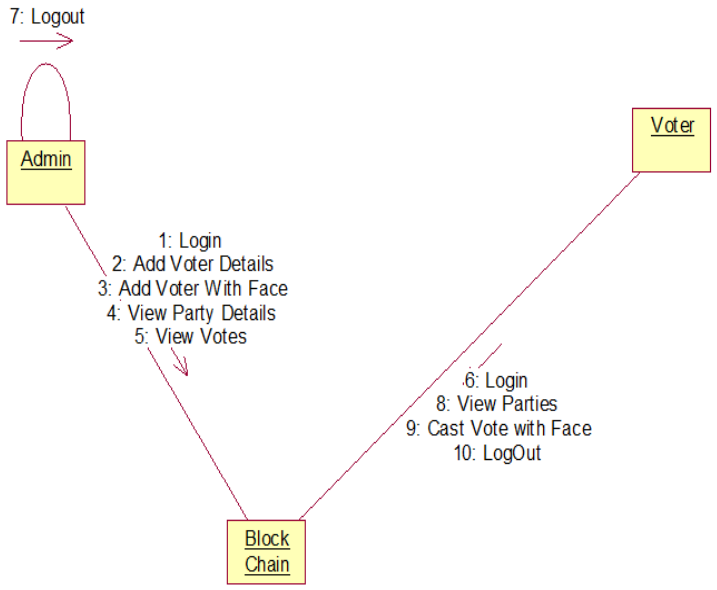

Figure 3.4.4: Collaborative Diagram

3.5 Software and Hardware Requirements

Hardware

i) System : Pentium IV 2.4 GHz.

ii) Hard Disk : 40 GB.

iii) Ram : 512 Mb.

Software

i)Operating system : Windows.

ii) Coding Language : python.

4. METHODOLOGY

4.1 Tools and Libraries used

Blockchain : Blockchain revolutionizes the way transactions and assets are man aged by providing a shared, immutable ledger within business networks. This innovative technology facilitates the secure and transparent recording of transactions, allowing for the tracking of a diverse range of assets, both tangible (such as houses, cars, and land) and intangible (including intellectual property, patents, copyrights, and branding). Blockchain's inherent features make it a versatile platform capable of handling virtually anything of value, offering a comprehensive solution for various industries.

Solidity : Solidity is a high-level programming language designed for implementing smart contracts. It is a statically typed object oriented(contract-oriented)language.Solidity is highly influenced by Python, c++, and JavaScript which run on the Ethereum Virtual Machine(EVM).Solidity supports complex user-defined programming, libraries, and inheritance.Solidity is the primary language for blockchains running platforms.Solidity can be used to create contracts like voting, blind auctions, crowdfunding, multi-signature wallets.

Ganache : Ganache is a local blockchain tool that is used to develop and test Ethereum applications1. It is an in-memory blockchain server that emulates a full Ethereum client and allows developers to test and debug smart contracts without requiring a real blockchain1. Ganache serves a vital role in all stages of the development process with many plausible advantages2. It can be used throughout development, providing a secure and predictable environment for developing, deploying, and testing dApps.

File Storage : File storage in a secure digital voting system using blockchain technology ensures the confidentiality, integrity, and transparency of voting-related data, contributing to the credibility and trustworthiness of the electoral process.

OpenCV : OpenCV can be used to detect and recognize faces during the voter authentication process. By capturing and analyzing facial features, it helps ensure that only eligible voters participate in the election and that each voter's identity is verified accurately.OpenCV can compare a captured face with a reference image or database of faces to verify the identity of the individual. This is essential for ensuring that the person attempting to vote is the same as the registered voter, thereby preventing fraudulent voting.

Face-API : TheFace-API is a library used for face detection, face authentication, face recognition, etc. We used this library in our project to detect faces in the real time and get the count of the faces in real time. If the multiple faces are detected in the webcam then the system will show an alert to the user.One notable feature of the Face-API is its ability to handle scenarios involving multiple faces. In instances where the webcam captured multiple faces concurrently, the system triggered an alert to promptly notify the user. This alert mechanism served as an effective means of communication, ensuring that users were informed in real-time about the presence of multiple faces within the camera's field of view.

CNN : In a secure digital voting system using blockchain technology, Convolutional Neural Networks (CNNs) can be employed for various purposes such as face recognition, where they can learn hierarchical representations of facial features from raw image data. CNNs can verify the identity of voters by comparing their facial features with reference images stored in the system. This helps ensure that each voter is who they claim to be, mitigating the risk of impersonation or fraudulent voting.

Python : Python programming language can be utilized in various aspects of a secure digital voting system using blockchain technology to develop the backend of the voting system, including server-side logic, and API integration. Frameworks like Django provide robust tools for building scalable and secure web applications. Python libraries such as Web3.py can be employed to interact with

blockchain networks, enabling the integration of blockchain technology into the voting system. This includes deploying smart contracts, reading data from the blockchain, and sending transactions securely.

Tensor Flow : TensorFlow provides pre-trained models and tools for facial recognition tasks. It can be used to develop a facial recognition system that verifies the identity of voters during the registration and authentication process. This helps in preventing impersonation and ensuring that only eligible voters can participate in the voting process. TensorFlow can be used for analyzing voting data collected from the blockchain to identify voting trends, patterns, and correlations.

4.2 Algorithms Used

In a secure digital voting system using blockchain technology, the SHA-256 algorithm is employed to hash individual votes, creating unique faceprints for data integrity. Each vote, represented by its hash, is recorded as a transaction on the blockchain, forming an immutable and tamper-resistant ledger. The cryptographic properties of SHA-256 ensure that any attempt to alter vote data would result in a distinct hash value, providing a robust mechanism for detecting tampering. The blockchain's consensus mechanism and secure communication practices further enhance the overall security of the voting system, creating a transparent and trust worthy environment for recording and validating votes.

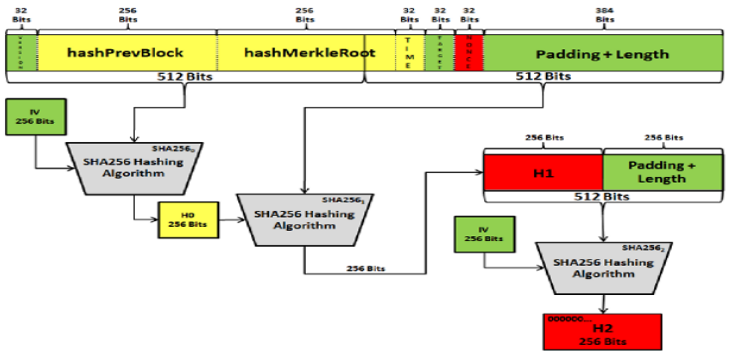

Figure 4.2: SHA-256 Algorithm Architecture

4.3 Modules

Admin Module– In admin module it is the responsibility of adding the party details and registering the voters and validating the vote count. Admin login to system by using username as 'admin' and password as 'admin'.

User Module– In user module the user has to sign up with the application by using username as his ID and then upload his face photo which capture from webcam. After registering user can go for login which validate user id and after successful login user can go for casting the vote.

Register– The voter must first run the application using blockchain then open the website. Register on the website using the user's basic information, including user name, password, address, contact number, email id, aadhar, profile image. The system would enable successful registration and would also encrypt the Aadhar number using SHA-256 encryption for security reasons.

Face-Capture– After Entering all the basic details for registration the user will be redirected to the face capture page where the user's face will be captured and then the captured image will be converted in base64 string and stored in the file storage along with the user's basic information. The face is captured for avoiding the risk of fake voters.

Login– Voting requires logging in, thus users must do so in order to cast a ballot. The user's username and password are required to login to the site, and the system will then encrypt using SHA-256 Encryption and determine whether or not the user is already registered by doing a check. Likewise upon a successful verification. A redirect will take the user to the voting page.

Vote Casting– Following a successful login, the user will be taken to the voting page, where they can select a candidate from a dropdown box that lists all the candidates. From where the users can

select the candidate whom they wants to vote and click the Vote button to cast their vote. After successfully casting the vote, the vote button will be disabled, making it difficult for the user to vote again.

Face Proctoring– The system will be able to detect faces by using the camera on the user's device. We are utilizing the Face-API library to detect faces. If the camera detects more than one face, the system will display an alert to the user and also sent the user back to the home page and stop their voting process.

Display Result– A real-time update of the vote total will be shown. Thus the transparency will be maintained and user can also verify their vote is getting counted.

5. Implementation & Results

In the implementation phase of the secure digital voting system using blockchain technology, the detailed design undergoes the transformation into functional code. This phase implements the web application platform in which it allows the user to register and cast vote with the blockchain ethereum platform techniquies.

5.1 Views.py

```
from django.template import RequestContext
from django.template import RequestContext
from django.contrib import messages
from django.http import HttpResponse
from django.core.files.storage import FileSystemStorage
import os
from datetime import date
import cv2
import numpy as np
import base64
import random
from datetime import datetime
from PIL import Image
import json
from web3 import Web3, HTTPProvider
import face_recognition
global username, password, contact, email, aadhar, address, names, encodings
global contract, web3
global usersList, partyList, voteList
#function to call contract
```

```
def getContract():
    global contract, web3
    blockchain_address = 'http://127.0.0.1:9545'
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'Voting.json' #voting contract file
    deployed_contract_address = '0xd8a8e9202C03A93Eb20fEd7d34b84eDDE128c532'
        #contract address
    with open(compiled_contract_path) as file:
        contract_json = json.load(file) # load contract info as JSON
        contract_abi = contract_json['abi'] # fetch contract's abi - necessary to call
            its functions
    file.close()
    contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi)
getContract()

def getUsersList():
    global usersList, contract
    usersList = []
    count = contract.functions.getUserCount().call()
    for i in range(0, count):
        user = contract.functions.getUsername(i).call()
        password = contract.functions.getPassword(i).call()
        email = contract.functions.getEmail(i).call()
        usersList.append([user, password, email,aadhar])

def getPartyList():
```

```
    global partyList, contract
    partyList = []
    count = contract.functions.getPartyCount().call()
    for i in range(0, count):
        cname = contract.functions.getCandidateName(i).call()
        pname = contract.functions.getPartyName(i).call()
        area = contract.functions.getArea(i).call()
        symbol = contract.functions.getSymbol(i).call()
        partyList.append([cname, pname, area, symbol])

def getVoteList():
    global voteList, contract
    voteList = []
    count = contract.functions.getVotingCount().call()
    for i in range(0, count):
        user = contract.functions.getUser(i).call()
        party = contract.functions.getParty(i).call()
        dd = contract.functions.getDate(i).call()
        candidate = contract.functions.getCandidate(i).call()
        voteList.append([user, party, dd, candidate])

def loadModel():
    global names, encodings
    if os.path.exists("model/encoding.npy"):
        encodings = np.load("model/encoding.npy")
        names = np.load("model/names.npy")
    else:
```

```
        encodings = []
        names = []


getUsersList()
getPartyList()
getVoteList()
loadModel()

face_detection = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

def alreadyCastVote(candidate):
    global voteList
    count = 0
    for i in range(len(voteList)):
        vl = voteList[i]
        if vl[3] == candidate:
            count = 1
    return count

def FinishVote(request):
    if request.method == 'GET':
        global username, voteList
        cname = request.GET.get('cname', False)
        pname = request.GET.get('pname', False)
        voter = ''
        today = date.today()
```

```
            status = 'Your vote casted to '+cname
            msg = contract.functions.createVote(username, pname, str(today),
                cname).transact()
            web3.eth.waitForTransactionReceipt(msg)
            voteList.append([username, pname, str(today), cname])
            context= {'data':'<font size=3 color=black>Your Vote Accepted for Candidate
                '+cname}
            return render(request, 'UserScreen.html', context)

def getOutput(status):
    global partyList
    output = '<h3><br/>'+status+'<br/><table border=1 align=center>'
    output+='<tr><th><font size=3 color=black>Candidate Name</font></th>'
    output+='<th><font size=3 color=black>Party Name</font></th>'
    output+='<th><font size=3 color=black>Area Name</font></th>'
    output+='<th><font size=3 color=black>Image</font></th>'
    output+='<th><font size=3 color=black>Cast Vote Here</font></th></tr>'
    for i in range(len(partyList)):
        pl = partyList[i]
        output+='<tr><td><font size=3 color=black>'+pl[0]+'</font></td>'
        output+='<td><font size=3 color=black>'+pl[1]+'</font></td>'
        output+='<td><font size=3 color=black>'+pl[2]+'</font></td>'
        output+='<td><img src="/static/parties/'+pl[3]+'" width=200
            height=200></img></td>'
        output+='<td><a href="FinishVote?cname='+pl[0]+'&pname='+pl[1]+'"><font size=3
            color=black>Click Here</font></a></td></tr>'
    output+="</table><br/><br/><br/><br/><br/><br/>"
```

```
    return output


def ValidateUser(request):
    if request.method == 'POST':
        global username, encodings, names
        predict = "none"
        page = "UserScreen.html"
        status = "unable to predict user"
        img = cv2.imread('VotingApp/static/photo/test.png')
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        face_component = None
        faces =
            face_detection.detectMultiScale(img, scaleFactor=1.1,minNeighbors=5,minSize=(30,30),flags=
        status = "Unable to predict. Please retry"
        faces = sorted(faces, reverse=True,key=lambda x: (x[2] - x[0]) * (x[3] -
            x[1]))[0]
        (fX, fY, fW, fH) = faces
        face_component = gray[fY:fY + fH, fX:fX + fW]
        if face_component is not None:
            img = cv2.resize(img, (600, 600))
            rgb_small_frame = cv2.cvtColor(img, cv2.COLOR_BGR2RGB) # Convert the
                frame to RGB color space
            face_locations = face_recognition.face_locations(rgb_small_frame) # Locate
                faces in the frame
            face_encodings = face_recognition.face_encodings(rgb_small_frame,
                face_locations) # Encode faces in the frame
            for face_encoding in face_encodings:
```

```
                matches = face_recognition.compare_faces(encodings, face_encoding) #
                    Compare face encodings
                face_distance = face_recognition.face_distance(encodings, face_encoding)
                     # Calculate face distance
                best_match_index = np.argmin(face_distance) # Get the index of the best
                    match
                print(best_match_index)
                if matches[best_match_index]: # If the face is a match
                    name = names[best_match_index] # Get the corresponding name
                    predict = name
                    break
            if predict == username:
                count = alreadyCastVote(username)
                if count == 0:
                    page = 'VotePage.html'
                    status = getOutput("User predicted as : "+predict+"<br/><br/>")
                else:
                    status = "You already casted your vote"
                    page = "UserScreen.html"
            else:
                page = "UserScreen.html"
                status = "unable to predict user"
            context= {'data': status}
            return render(request, page, context)

def UserLogin(request):
    if request.method == 'POST':
```

```
        global username, contract, usersList
        username = request.POST.get('username', False)
        password = request.POST.get('password', False)
        status = "Login.html"
        output = 'Invalid login details'
        for i in range(len(usersList)):
            ulist = usersList[i]
            user1 = ulist[0]
            pass1 = ulist[1]
            if user1 == username and pass1 == password:
                status = "UserScreen.html"
                output = 'Welcome '+username
                break
        context= {'data':output}
        return render(request, status, context)

def AdminLogin(request):
    if request.method == 'POST':
        global username
        username = request.POST.get('username', False)
        password = request.POST.get('password', False)
        if username == 'admin' and password == 'admin':
            context= {'data':'Welcome '+username}
            return render(request, 'AdminScreen.html', context)
        if status == 'none':
            context= {'data':'Invalid login details'}
            return render(request, 'Admin.html', context)
```

```
def AddParty(request):
    if request.method == 'GET':
        return render(request, 'AddParty.html', {})

def index(request):
    if request.method == 'GET':
        return render(request, 'index.html', {})

def Login(request):
    if request.method == 'GET':
        return render(request, 'Login.html', {})

def CastVote(request):
    if request.method == 'GET':
        return render(request, 'CastVote.html', {})

def AddVoter(request):
    if request.method == 'GET':
        return render(request, 'AddVoter.html', {})

def Admin(request):
    if request.method == 'GET':
        return render(request, 'Admin.html', {})

def AddVoterAction(request):
    if request.method == 'POST':
```

```
        global username, password, contact, email, aadhar, address, usersList
        username = request.POST.get('username', False)
        password = request.POST.get('password', False)
        contact = request.POST.get('contact', False)
        email = request.POST.get('email', False)
        aadhar=request.POST.get('aadhar',False)
        address = request.POST.get('address', False)
        status = "none"
        for i in range(len(usersList)):
            ul = usersList[i]
            if username == ul[0]:
                status = "exists"
                break
        if status == "none":
            context= {'data':'Capture Your face'}
            return render(request, 'CaptureFace.html', context)
        else:
            context= {'data':username+' Username already exists'}
            return render(request, 'AddVoter.html', context)


def WebCam(request):
    if request.method == 'GET':
        data = str(request)
        formats, imgstr = data.split(';base64,')
        imgstr = imgstr[0:(len(imgstr)-2)]
        data = base64.b64decode(imgstr)
        if os.path.exists("VotingApp/static/photo/test.png"):
```

```
            os.remove("VotingApp/static/photo/test.png")
        with open('VotingApp/static/photo/test.png', 'wb') as f:
            f.write(data)
        f.close()
        context={'data':"done"}
        return HttpResponse("Image saved")

def saveFace():
    global names, encodings
    encodings = np.asarray(encodings)
    names = np.asarray(names)
    np.save("model/encoding", encodings)
    np.save("model/names", names)

def saveUser(request):
    if request.method == 'POST':
        global username, password, contact, email, aadhar, address, usersList,
            encodings, names
        img = cv2.imread('VotingApp/static/photo/test.png')
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        face_component = None
        faces = face_detection.detectMultiScale(gray, 1.3,5)
        page = "AddVoter.html"
        status = 'Unable to detect face. Please retry'
        for (x, y, w, h) in faces:
            face_component = img[y:y+h, x:x+w]
        if face_component is not None:
```

```
        img = cv2.resize(img, (600, 600))
        if os.path.exists("VotingApp/static/photo/test.png"):
            os.remove("VotingApp/static/photo/test.png")
        cv2.imwrite("VotingApp/static/photo/test.png", img)
        image = face_recognition.load_image_file("VotingApp/static/photo/test.png")
        encoding = face_recognition.face_encodings(image)
        print("encoding "+str(encoding))
        if len(encoding) > 0 and username not in names:
            encoding = encoding[0]
            if len(encodings) == 0:
                encodings.append(encoding)
                names.append(username)
            else:
                encodings = encodings.tolist()
                names = names.tolist()
                encodings.append(encoding)
                names.append(username)
            saveFace()
            page = "AddVoter.html"
            status = 'User with Face Details added to Blockchain<br/><br/>'
            msg = contract.functions.createUser(username, email,password, aadhar,
                contact, address).transact()
            status += str(web3.eth.waitForTransactionReceipt(msg))
            usersList.append([username, password, email])
    context= {'data': status}
    return render(request, page, context)
```

```
def AddPartyAction(request):
    if request.method == 'POST':
        global partyList
        cname = request.POST.get('t1', False)
        pname = request.POST.get('t2', False)
        area = request.POST.get('t3', False)
        myfile = request.FILES['t4']
        imagename = request.FILES['t4'].name
        status = "none"
        page = "AddParty.html"
        for i in range(len(partyList)):
            pl = partyList[i]
            if cname == pl[0] and pname == pl[1]:
                status = "Candidate & Party Name Already Exists"
                break
        if status == "none":
            fs = FileSystemStorage()
            filename = fs.save('VotingApp/static/parties/'+imagename, myfile)
            status = 'Candidate details added to Blockchain<br/><br/>'
            msg = contract.functions.createParty(cname, pname, area,
                imagename).transact()
            status += str(web3.eth.waitForTransactionReceipt(msg))
            partyList.append([cname, pname, area, imagename])
        context= {'data': status}
        return render(request, page, context)

def getVoteCount(cname, pname):
```

```
    global voteList
    count = 0
    for i in range(len(voteList)):
        vl = voteList[i]
        if vl[1] == pname and vl[3] == cname:
            count += 1
    return count

def ViewVotes(request):
    if request.method == 'GET':
        output = '<table border=1 align=center>'
        output+='<tr><th><font size=3 color=black>Candidate Name</font></th>'
        output+='<th><font size=3 color=black>Party Name</font></th>'
        output+='<th><font size=3 color=black>Area Name</font></th>'
        output+='<th><font size=3 color=black>Image</font></th>'
        output+='<th><font size=3 color=black>Vote Count</font></th>'
        for i in range(len(partyList)):
            pl = partyList[i]
            count = getVoteCount(pl[0], pl[1])
            output+='<tr><td><font size=3 color=black>'+pl[0]+'</font></td>'
            output+='<td><font size=3 color=black>'+pl[1]+'</font></td>'
            output+='<td><font size=3 color=black>'+pl[2]+'</font></td>'
            output+='<td><img src="/static/parties/'+pl[3]+'" width=200
                height=200></img></td>'
            output+='<td><font size=3 color=black>'+str(count)+'</font></td></tr>'
        output+="</table><br/><br/><br/><br/><br/><br/>"
        context= {'data':output}
```

```
        return render(request, 'ViewVotes.html', context)

def ViewParty(request):
    if request.method == 'GET':
        output = '<table border=1 align=center>'
        output+='<tr><th><font size=3 color=black>Candidate Name</font></th>'
        output+='<th><font size=3 color=black>Party Name</font></th>'
        output+='<th><font size=3 color=black>Area Name</font></th>'
        output+='<th><font size=3 color=black>Image</font></th></tr>'
        for i in range(len(partyList)):
            pl = partyList[i]
            output+='<tr><td><font size=3 color=black>'+pl[0]+'</font></td>'
            output+='<td><font size=3 color=black>'+pl[1]+'</font></td>'
            output+='<td><font size=3 color=black>'+pl[2]+'</font></td>'
            output+='<td><img src="/static/parties/'+pl[3]+'" width=200
                height=200></img></td></tr>'
        output+="</table><br/><br/><br/><br/><br/><br/>"
        context= {'data':output}
        return render(request, 'ViewParty.html', context)
```

5.2 Voting.sol

```
pragma solidity >=0.8.11 <=0.8.11;
pragma experimental ABIEncoderV2;
//cheque solidity code
contract Voting {
    uint public partyCount = 0;
    mapping(uint => party) public partyList;
```

```
    struct party
    {
        string candidate;
        string partyname;
        string areaname;
        string symbol;
    }

// events
event partyCreated(uint indexed _partyId);
//function to save party details
function createParty(string memory cn, string memory pn, string memory an, string
    memory sy) public {
    partyList[partyCount] = party(cn, pn, an, sy);
    emit partyCreated(partyCount);
    partyCount++;
}

  //get Party count
 function getPartyCount() public view returns (uint) {
        return partyCount;
 }

 function getCandidateName(uint i) public view returns (string memory) {
      party memory chq = partyList[i];
return chq.candidate;
 }
```

```
 function getPartyName(uint i) public view returns (string memory) {
      party memory chq = partyList[i];
return chq.partyname;
 }

 function getArea(uint i) public view returns (string memory) {
      party memory chq = partyList[i];
return chq.areaname;
 }

 function getSymbol(uint i) public view returns (string memory) {
      party memory chq = partyList[i];
return chq.symbol;
 }

 uint public votingCount = 0;
 mapping(uint => voting) public votingList ;
   struct voting
  {
     string user;
     string party;
     string date;
     string candidate;
  }

// events
```

```
event voteCreated(uint indexed _voteId);

//function to save vote details
function createVote( string memory usr, string memory party, string memory dd, string
    memory candidate) public {
    votingList[votingCount] = voting(usr, party, dd, candidate);
   emit voteCreated(votingCount);
   votingCount++;
 }

  //get Vote count
 function getVotingCount() public view returns (uint) {
        return votingCount;
 }

 function getUser(uint i) public view returns ( string memory) {
      voting memory chq = votingList[i];
return chq.user;
 }

 function getParty(uint i) public view returns ( string memory) {
      voting memory chq = votingList[i];
return chq.party;
 }

 function getDate(uint i) public view returns ( string memory) {
      voting memory chq = votingList[i];
```

```
return chq.date;
}

function getCandidate(uint i) public view returns (string memory) {
    voting memory chq = votingList[i];
return chq.candidate;
}

uint public userCount = 0;
mapping(uint => user) public usersList;
  struct user
  {
    string username;
    string email;
    string password;
    string aadhar;
    string phone;
    string home_address;
  }

// events

event userCreated(uint indexed _userId);
function createUser(string memory _username, string memory _email, string memory
    _password, string memory _aadhar, string memory _phone, string memory _address)
    public {
     usersList[userCount] = user(_username, _email, _password, _aadhar, _phone,
```

```
            _address);
        emit userCreated(userCount);
        userCount++;
    }

     //get user count
    function getUserCount() public view returns (uint) {
            return userCount;
    }
    function getUsername(uint i) public view returns (string memory) {
        user memory usr = usersList[i];
    return usr.username;
    }
    function getPassword(uint i) public view returns (string memory) {
        user memory usr = usersList[i];
    return usr.password;
    }
    function getEmail(uint i) public view returns (string memory) {
        user memory usr = usersList[i];
    return usr.email;
    }
    function getAadhar(uint i) public view returns (string memory) {
        user memory usr = usersList[i];
    return usr.aadhar;
    }
}
```

6. RESULTS & DISCUSSIONS

Our proposed system introduces a dynamic website for online voting using Blockchain Technology where user can register into the website using the basic user credentials with face capturing then the details are stored into the blockchain in the encrypted form. So that there is no chance of manipulating the data. After successful registration the user can sign into the user module page where the face Proctoring takes for validating the voter. If the voter matches then the party details are listed so the voter can choose the candidate whom they wants to vote and click the Vote button to cast their vote. After successfully casting the vote, the vote button will be disabled, making it difficult for the user to vote again.

6.1 Deploying Blockchain Contract

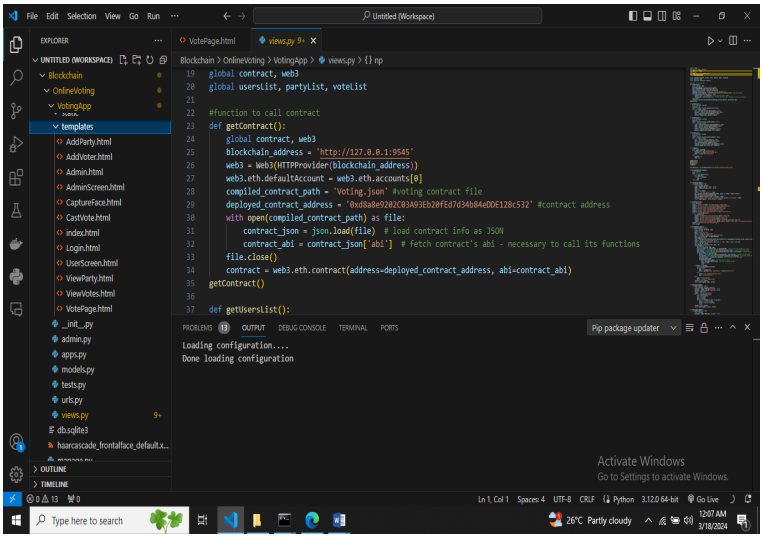

Figure 6.1.1 : Deploying blockchain Contract

In above contract we have defined function to manage all Voting system like voter details, party details and user registration details. Now we need to deploy above contract in Blockchain Ethereum using below steps.

1) First go inside ‘hello-eth/node-modules/bin’ folder and then double click on ‘runBlockchain.bat’ file to get below screen.

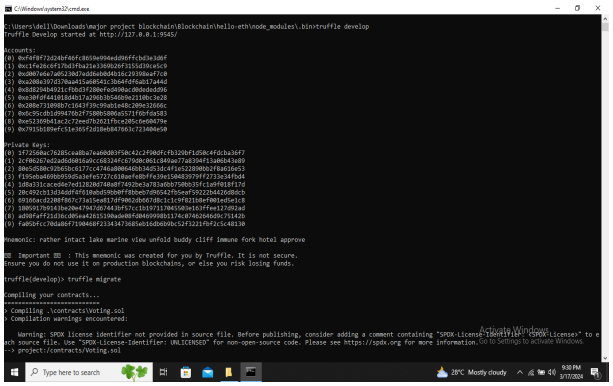

Figure 6.1.2: Running Blockchain

2) In above screen Blockchain started with default private keys and account and now type command as ‘migrate’ and press enter key to deploy contract and get below output.

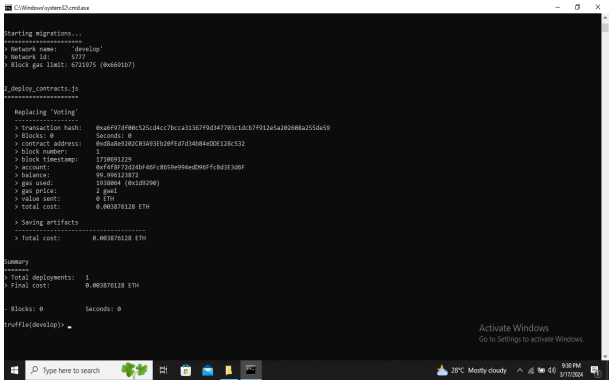

Figure 6.1.3: Blockchain Truffle Suite Output

3) In above screen Voting contract deployed and got contract address and this address need to specify in python code to call functions from Blockchain. Click on ‘run.bat’ file to start python web server.

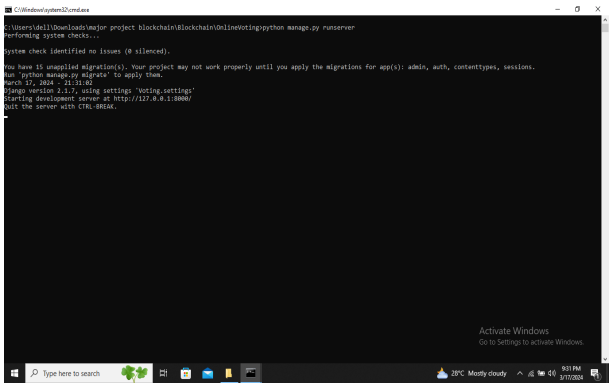

Figure 6.1.3: Run Server Output

6.2 Home Page

Home page consists of user login and admin login. In which user can register by entering his or her details. Admin login using username as ‘admin’ and password as ‘admin’.

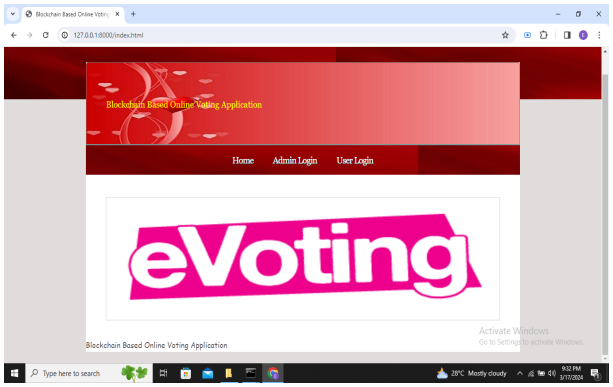

Figure 6.2.1: Home Page

Admin Login: The application prompts the admin to enter their login details. Admin login to system by using username as ’admin’ and password as ’admin’.

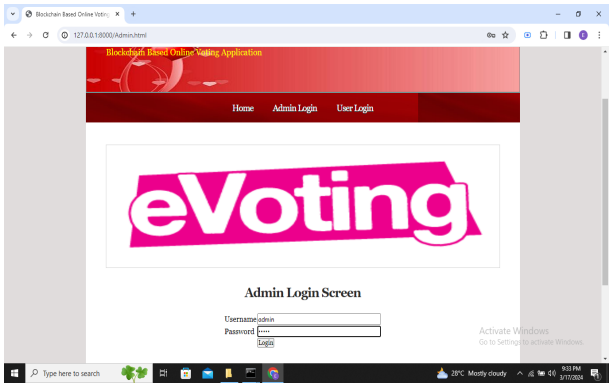

Figure 6.2.2: Admin Login Page

Add Party Details: The admin will add the party details like Candidate name, party area, party name, party image for the particular area.

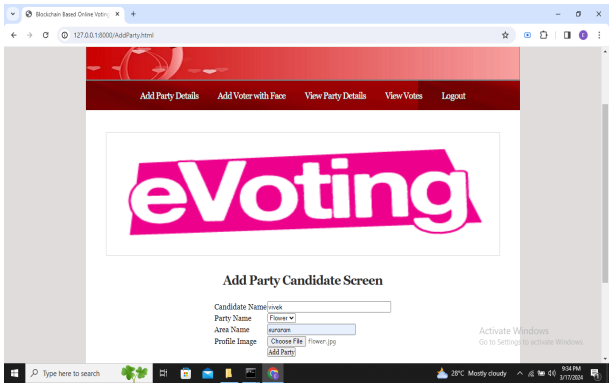

Figure 6.2.3: Adding Parties

Party Details: In this page the party Details of each area are listed.

Figure 6.2.4: Party Details

Initial Vote Count: In this page the intial vote count details are shown.

Figure 6.2.5: Initial Vote Count

Face-Capture: After Entering all the basic details for registration the user will be redirected to the face capture page where the user's face will be captured and then the captured image will be converted in base64 string and stored in the file storage.

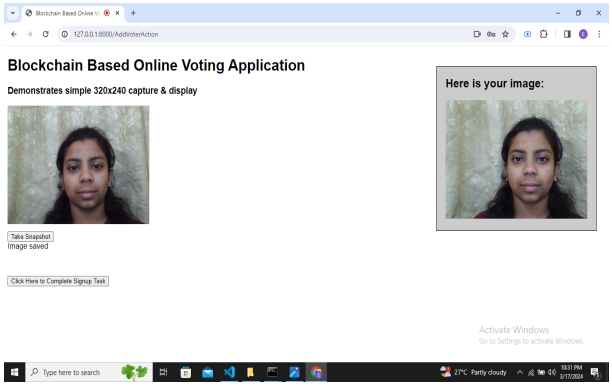

Figure 6.2.6: Face Capture

Login: Voting requires logging in, thus users must do so in order to cast a ballot. The user's username and password are required to login to the site.

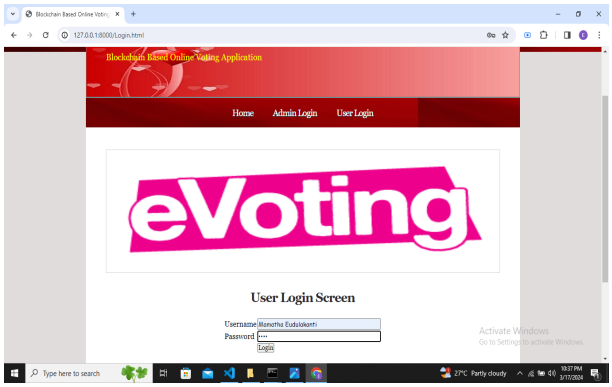

Figure 6.2.7: User Login

Validate User: In this page the user is validated by the CNN predicted face with the matching criteria.

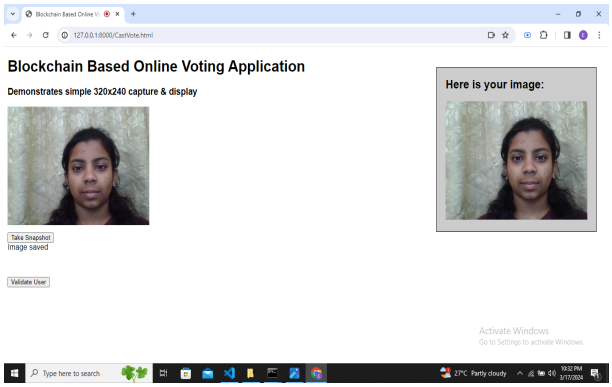

Figure 6.2.8: Validate User

Vote Casting: In this page the user has to select the candidate whom they wants to vote and click the Vote button to cast their vote.

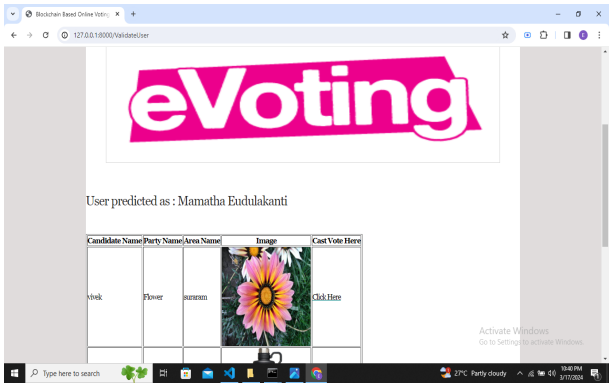

Figure 6.2.9: Vote Casting

Vote Casted : In this page the vote to the selected candidate will be accepted.

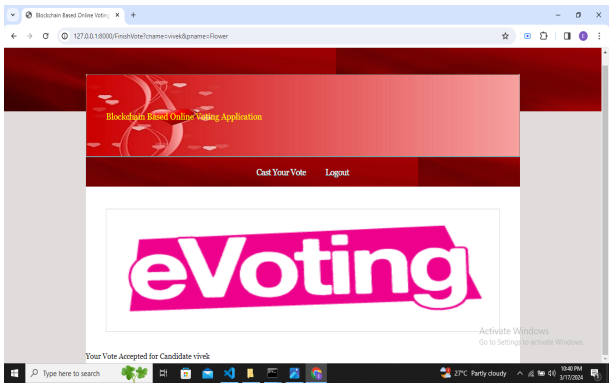

Figure 6.2.10: Vote Casted

Final Vote Count: In this page the final count of votes for each candidate will be displayed.

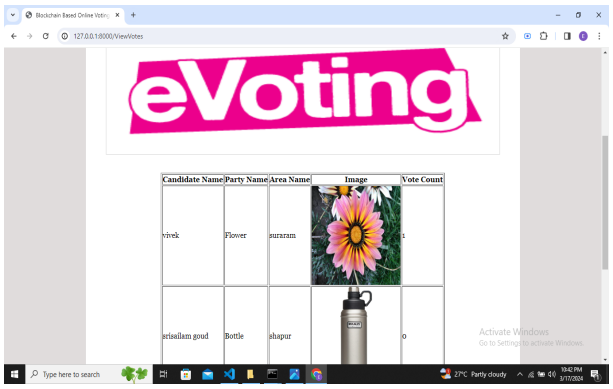

Figure 6.2.11: Final Vote Count

7. CONCLUSIONS & FUTURE SCOPE

Blockchain technology offers a decentralized and immutable ledger that ensures the integrity, security, and transparency of the voting process. By leveraging cryptographic techniques, smart contracts, and consensus mechanisms, the proposed application provides a secure, transparent, and accessible platform for conducting elections in the digital age.

The advantages of the proposed system are manifold. It offers transparency by recording all votes on the blockchain, ensuring an auditable and tamper-resistant record of the election results. Security measures, such as encryption, digital signatures, and multi-factor authentication, protect against unauthorized access and tampering of voting data. The system also prioritizes accessibility and inclusivity by enabling voters to cast their ballots online from any location with an internet connection.

However, the implementation of a blockchain-based online voting application is not without challenges. Scalability, regulatory compliance, and public trust are among the key challenges that need to be addressed. Additionally, ongoing research and development are required to overcome technical limitations and ensure the reliability and usability of the voting application.

Despite these challenges, the proposed application has the potential to revolutionize the way elections are conducted, fostering greater trust, transparency, and participation in the democratic process. By harnessing the power of blockchain technology, we can create a more inclusive, secure, and democratic electoral system for the benefit of society as a whole.

Future work in the development and enhancement of blockchain-based online voting applications holds great promise for further improving the integrity, security, and accessibility of electoral processes. One avenue for future work is the continued research and development of scalability solutions for blockchain networks. As the demand for online voting applications grows, scalability becomes increasingly important to ensure that the system can handle large volumes of transactions and users

without compromising performance or security. Solutions such as sharding, sidechains, and layer 2 protocols offer promising approaches to scaling blockchain networks while maintaining decentralization and security.

Another area for future work is the exploration of advanced cryptographic techniques and privacy-preserving mechanisms to further enhance the privacy and anonymity of voters. Techniques such as homomorphic encryption, secure multi-party computation, and privacy-enhancing protocols like zero-knowledge proofs can provide stronger guarantees of voter privacy while still enabling the verification of election integrity.

Moreover, research into the integration of blockchain technology with other emerging technologies, such as biometrics, artificial intelligence, and internet of things (IoT), could yield innovative solutions for enhancing the security and usability of online voting applications. For example, biometric authentication methods could strengthen voter identification processes, while AI algorithms could help detect and prevent fraud or manipulation in real-time.

Furthermore, future work should also focus on addressing regulatory and legal challenges surrounding the adoption of blockchain-based online voting applications. Collaborations between technology developers, policymakers, election authorities, and civil society organizations are essential to establish regulatory frameworks, standards, and best practices for ensuring the integrity, security, and legality of blockchain-based voting systems.

REFERENCES

[1] S. A. P. S. Sonali Ridhorkar, Monali kamlesh Wanjari, "My vote blockchain based online voting system", International Research Journal of Innovation in Engineering an Technology(IRJIET), vol.7, 2023.

[2] V. Sathya Preiya, V. D. Ambeth Kumar, R. Vijay, Vijay K, N. Kirubakaran, "Blockchain-Based E-Voting System with Face Recognition", American Scientific Publishing Group, Vol. 12, No. 01. PP. 53-63, 2023, DOI: https://doi.org/10.54216/FPA.120104.

[3] J. M. R. R.-C. Maria-Victoria Vladucu, Ziqian Dong, "E-voting meets blockchain: A survey," IEEE Access, vol. 11, 2023, DOI: 10.1109/ACCESS.2023.3253682.

[4] N. S. S. K. L. i. j. a. J. Farzana Khareem, Muhammad Safeer, "Online vot ing system using blockchain technology", International Journal of Computer Science Trends and Technology (IJCST), vol. 10, 2022, ISSN: 2347-8578.

[5] Mr. Shreeyash Pednekar, Mr. Bhushan Halasagi, "BLOCKCHAIN BASED E-VOTING SYSTEM", "International Journal of Creative Research Thoughts (IJCRT)", ISSN: 2320-2882, Vol.10, Issue 5, May 2022, ISSN: 2320-2882.

[6] Z. S. Uzma Jafar, Mohd Juzaiddin Ab Aziz, "Blockchain for electronic voting system review and open research challenges," MDPI Access, vol. 21, 2021, DOI:10.3390/s21175874.

[7] S. S. A. Y. C. Dhiraj Amrutkar, Gaurav Dongare, "E-voting systems using blockchain: A systematic review and future research direction", International Journal of Research and Development (IJRD), vol. 6, 2021, DOI:10.36713/epra7157.

[8] C. Mehta, A. Mehta, S. Gada and N. Kadukar, ”Demystifying Democracy: Incentivizing Blockchain Voting Technology for an enriched Electoral System”, International Conference on Communication information and Computing Technology (ICCICT), Mumbai, India, 2021, DOI: 10.1109/ICCICT50803.2021.9510158.

[9] Pradeep Katta, Ovaiz Mohammed, Prabaakaran Kandasamy, M Divya, “Smart voting using Fingerprint, Face and OTP Technology with Blockchain”, Journal of Physics Conference Series, May 2021, DOI:10.1088/1742-6596/1916/1/012139.

[10] Mrunal Parthak, Amol Suradkar, Ajinkya Kadam, Akanaha Ghodeswar, Prashant Parde, “Blockchain based e-voting system”, International Journal of Scientific Research in Science and Technology, vol. 8, 2021, DOI:10.32628/IJSRST2182120.

[11] S. S. P. T. Dipali Pawar, Pooja Sarode, “Implementation of secure voting system using blockchain”, International Journal of Engineering Research Technology (IJERT), vol. 9, 2020, ISSN: 2278-0181.

[12] O. T. Ayman Afaneh, Fatemeh Noroozi, “Recognition of identical twins using fusion of various facial feature extractors”, EURASIP Journal on Image and Video Processing, vol. 8, 2020.

[13] Muhammad Asaad Cheema, Nouman Ashraf, Asad Aftab, Hassaan Khaliq Qureshi, Muhamm Kazim, Ahmed Taher Azar, “Machine Learning with Blockchain for Secure E-Voting System”, IEEE Access, November 2020.

[14] A. U. T. A. A. P. Abhishek Subhash Yadav, Yash Vandesh Urade, “E voting using blockchain technology”, International Journal of Engineering Research Technology (IJERT), vol. 9, 2020.

[15] Kashif Mehboob Khan, Junaid Arshad, Muhammad Mubashir Khan, " Secure Digital Voting System Using Blockchain", International Journal of Electronic Government Research, January 2018.

[16] M. M.K.Kashif Mehboob Khan, Junaid Arshad, "Secure digital voting system based on blockchain technology", International Journal of Electronic Government Research, vol. 14, 2018.