INTRODUCTION AND MOTIVATION

1. INTRODUCTION & MOTIVATION

1.1 INTRODUCTION

Counting hands was the first voting method, and it has since been replaced by paper, punch cards, mechanical levers, and optical scan devices. Modern electronic voting systems have a few characteristics that set them apart from more antiquated methods. They also offer better features than those methods, including mobility, accuracy, privacy, ease, adaptability, and verifiability. However, electronic voting methods have several problems, such being time-consuming, requiring a lot of paper labor, not involving senior officials directly, causing machine damage from neglect, preventing users from editing and updating many items at once, and so on. Thus, user can avoid data loss by putting in place a decentralized Blockchain-based server infrastructure. Using blockchain technology to hold a digital election lowers the possibility of voting-related unfairness while simultaneously saving costs. Modern technologies, like as blockchain technology, possess a high degree of security, and offer significant advantages when employed with caution. The implementation of this technology as the potential to enhance the transparency, reliability, and Monitoring of voting systems [1]. A current voting system involves a voting machine connected to a central database. This machine can be interfered with by anyone who has access to it. It may cause a single point failure in the entire voting system network; but an immutable blockchain cannot be altered by an individual traitor in the entire network.[8]

1.2 AIM AND OBJECTIVE

AIM

Introducing a new electronic voting system with face verification that will address the shortcomings of the country's current voting practices.[1]

OBJECTIVE

The key objectives of the project include:

- The election system must be openly verifiable and transparent.
- The election system must ensure that the vote cast by the voter has been recorded.
- Only eligible voters must be allowed to vote.
- The election system should be tamper-proof.
- No power-hungry organization must be able to manipulate and rig the election process.

1.3 EXISTING SYSTEM

This is the current voting system used in India. In this system vote is cast using electronics ballet. In this we cast our vote in an electronics machine. This is a group of some counter and registers. This voting system is quite easy, simple. It has advantage like mobility, secure, flexibility for election commission. But in today world all people are so much busy that they don’t have time to vote. This paper presents a perspective in the electronic voting process. That includes but not limited to identifying the polling process, The polling process the actual voting process used on the polling day.

Blockchain setup

To satisfy the privacy and security requirements for e-voting, and to ensure that the election system should not enable coerced voting, voters will have to vote in a supervised environment. In our work, we set up a Go-Ethereum permissioned Proof-of-Authority (POA) blockchain to achieve these goals. POA uses an algorithm that delivers comparatively fast transactions through a consensus mechanism based on identity as a stake. The reason for using Go-Ethereum for the blockchain infrastructure is explained in sub-section C. The structure of the blockchain is illustrated in Figure 1 and mainly consists of two types of nodes.

Verifying votes In the voting transaction, each voter receives the transaction ID of his vote. In this e-voting system, voters can use this transaction ID and go to an official election site (or authority) using a blockchain explorer and (after authenticating themselves using their electronic identification) locate the transaction with the corresponding transaction ID on the blockchain. Voters can, therefore, see their votes on the blockchain, and verify that the votes were listed and counted correctly. This type of verification satisfies the transparency requirements while preventing traceability of votes.

1.4 LIMITATIONS ON EXISTING SYSTEM

The problems of the existing manual system of voting include among others the following:

- **Expensive and Time consuming:** The process of collecting data and entering this data into the database takes too much time and is expensive to conduct, for example, time and money is spent in printing data capture forms, in preparing registration stations together with human resources, and there after advertising the days set for registration process including sensitizing voters on the need for registration, as well as time spent on entering this data to the database.
- **Too much paper work:** The process involves too much paper work and paper storage which is difficult as papers become bulky with the population size.
- **Short time provided to view the voter register:** This is a very big problem since not all people have free time during the given short period of time to check and update the voter register.

1.5 PROPOSED SYSTEM

The system that is suggested is the face verified online e-voting system with Face Verification using KNN algorithm and identification of voter using Block chain Address [2]. The Blockchain address is used to determine whether a particular voter is valid or not. It enables a particular voter to cast their ballot online. The polling procedure keeps going until the voting period is over, updating the server's database. Block chain addresses are used by the Face Verification online voting method to obtain all of the voter's personal information [3]. Additionally, the votes are publicly accessible and kept on a blockchain server, guaranteeing a reliable environment. When a voter inquiries to vote, the VMS, checks the voter's voting status on the blockchain through contrasting all existing transaction hashes with his or her computerized ID (Ethereum address). If a transaction's hash has been determined against the voter's ID, VMS rejects the request and logs the voter out of the system.[1]

ADVANTAGES OF PROPOSED SYSTEM:

- Voter can cast their votes from anywhere in the country without visiting to voting booths, in highly secured way.
- This will increase the voting percentage in India and reduces the cost of voting process.
- By using Face Verification it provides enough security which reduces the false votes.
- The collection of the results is done from the stored data on the blocks through the significant organization of the nodes in the block chain.

Algorithm: KNN, SHA-256

PROJECT OVERVIEW

2. PROJECT OVERVIEW

2.1 LITERATURE SURVEY

Paper 1

Paper Name: Secure E-Voting System using Block-chain technology and authentication via Face recognition and Mobile OTP

Authors Name: Q. Zhang, Palapye, Botswana

Year of Publish: 2022

Explanation:

A distributed ledger is used in blockchain which store data, making it an essential part of democracy. Web-based voting systems have evolved, making them available to all citizens, including those in rural areas. Voting ensures that each citizen has a say in a country's legislation, while decentralized systems such as blockchain ensure that transactions are linked to previous ones. The miners mine on the blockchain and get paid for their efforts. To ensure security, blockchain-based voting systems employ a decentralized chain of blocks linked by nodes, each containing a cryptographic hash, timestamp, and exchange information. A proposed solution is to store voting-related information on the blockchain and monitor users throughout the vote-taking process, with a Face Recognition System to ensure they are verified and eligible to vote.[2]

Paper 2

Paper Name: Blockchain-Based E-Voting System

Authors Name: Hjálmarsson, Gunnlaugur K. Hreidarsson, Mohammad Hamdaqa
Year of Publish : 2018

Explanation:

A secure electronic voting system that offers the fairness and privacy of current voting schemes, while providing the transparency and flexibility offered by electronic systems has been a challenge for a long time. In this work-in-progress paper, evaluate an application of blockchain as a service to implement distributed electronic voting systems. The paper proposes a novel electronic voting system based on blockchain that addresses some of the limitations in existing systems and evaluates some of the popular blockchain frameworks for the purpose of constructing a blockchain-based e-voting system. In particular, the paper evaluates the potential of distributed ledger technologies through the description of a case study; namely, the process of an election, and the implementation of a blockchain based application, which improves the security and decreases the cost of hosting a nationwide election[6].

Paper 3

Paper Name: Improved Face Recognition Rate Using HOG Features and SVM Classifier
Authors Name: Harihara Santosh Dadi, Gopala Krishna Mohan Pillutla

Year of Publish : 2019
Explanation:

A novel face recognition algorithm is presented in this paper. Histogram of Oriented Gradient features are extracted both for the test image and also for the training images and given to the Support Vector Machine classifier. The detailed steps of HOG feature extraction and the classification using SVM is presented. The algorithm is compared with the Eigen feature based face recognition algorithm. The proposed algorithm and PCA are verified using 8 different datasets. Results show that in all the face datasets the proposed algorithm shows higher face recognition rate when compared with the traditional Eigen feature based face recognition algorithm. There is an improvement of 8.75% face recognition rate when compared with PCA based face recognition algorithm. The experiment is conducted on ORL database with 2 face images for testing and 8 face images for training for each person. Three performance curves namely CMC, EPC and ROC are considered. The curves show that the proposed algorithm outperforms when compared with PCA algorithm[7].

2.1.1 Comparative Analysis

Table 2.1.1 Comparative Analysis of Existing System

Paper Title	Author	Year	Publication	Description
A Framework to Make Voting System Transparent Using Blockchain Technology	Muhammad Shoaib Farooq, Usman Iftikhar, Adel Khelif	2022	IEEE	Blockchain based e-voting system with Ethereum blockchain network,SHA-256 algorithm and face recognition with KNN.
Secure E-Voting System using Block-chain technology and authentication via Face recognition and Mobile OTP	A. Parmar, S. Gada, T. Loke, Y. Jain, S. Pathak and S. Patil	2021	12th International Conference on Computing Communication and Networking Technologies (ICCCNT)	Blockchain, a distributed ledger, is crucial for democracy and web-based voting systems, ensuring citizens have a say in legislation. Decentralized systems like blockchain link transactions and ensure security. A proposed solution is to store voting-related information on the blockchain and monitor users with a Face Recognition System..
E-Voting using Blockchain	Yash Dalvi, Shivam Jaiswal, Pawan Sharma	2021	International Journal of Engineering Research & Technology	An overview of the blockchain's fundamental properties and architecture in respect to electronic voting.
Improved Face Recognition Rate Using HOG Features and SVM Classifier	Harihara Santosh Dadi, Gopala Krishna Mohan Pillutla	2019	Iosr Journal Of Electronics And Communication Engineering (Iosr-Jece)	This paper presents an improved face recognition algorithm using HOG features and SVM classification, outperforming traditional methods by achieving an 8.75% higher recognition rat.

2.1.2 MATHEMATICAL MODEL

Creating a mathematical model for the K-Nearest Neighbors (KNN) algorithm in a face-verified voting system based on blockchain involves defining the key components of the system. Here's a simplified mathematical model:

1. **Data Representation**:

- Each voter's face data is represented as a feature vector, denoted as \(X_i\), where \(i\) represents the voter's index.

2. **Data Preprocessing**:

- Normalize and preprocess the feature vectors to ensure consistency. Let \(X_i\) be the preprocessed feature vector.

3. **Training Data:**

- The system uses a dataset of labeled facial biometric data to train the KNN algorithm.

4. **KNN Algorithm**:

- Given a new voter's feature vector \(X_{new}\) and a parameter \(K\) (number of nearest neighbors), the KNN algorithm finds the \(K\) nearest neighbors to \(X_{new\), denoted as \(N_{KNN}\).

5. **Similarity Measure**:

- Define a similarity measure, \(S(X_i, X_j)\), between two feature vectors \(X_i\) and \(X_j\). The choice of similarity measure depends on the nature of your data (e.g., Euclidean distance for numerical features, cosine similarity for embeddings).

6. **Nearest Neighbors**:

- Find the \(K\) nearest neighbors by ranking all voters based on the similarity measure:

\[N_{KNN} = \text{argmin}_{X_i} \sum_{X_j \in \text{Training Data}} S(X_{new}, X_j)\]

7. **Voting Decision**: - Aggregate the votes from the \(K\) nearest neighbors to determine the voting decision. For example, if the neighbors are classified as 'Verified' or 'Not Verified,' the majority vote among neighbors could be the decision.

8. **Blockchain Integration:**

- Integrate the voting decision into the blockchain-based voting system, recording the voter's decision and the verification process.

9. **Decision Threshold**:

- You may set a decision threshold for the KNN algorithm, indicating how many of the \(K\) neighbors must vote 'Verified' for the new voter to be 'Verified.'

Mathematical Formulation:

The mathematical model for the KNN algorithm in this context involves defining the similarity measure, which can vary based on the features used. For instance, if using the Euclidean distance as the similarity measure for numerical feature vectors, it can be defined as:

$$S(X_i, X_j) = \sqrt{\sum_{k=1}^{n} (X_{i_k} - X_{j_k})^2}$$

Where n is the dimensionality of the feature vectors, and X_{i_k} and X_{j_k} are the k-th components of the vectors X_i and X_j.

The decision threshold may be a value indicating the maximum acceptable distance or a specific number of neighbors required to vote 'Verified.'

This mathematical model serves as a conceptual framework.

In a face-verified voting system based on blockchain, the SHA-256 (Secure Hash Algorithm 256-bit) is used for securing and hashing critical data, such as voter identities and voting records. Here's a simplified mathematical model for the SHA-256 algorithm in this context:

SHA-256 algorithm

1. Data Representation:

Data, such as voter identities and voting records, is represented as a binary string or bytes. Let's denote this data as D.

2. SHA-256 Algorithm:

The SHA-256 algorithm takes the binary data D as input and computes a fixed-length 256-bit hash value H:

$H=\text{SHA-256}(D)$

3. Mathematical Formulation:

The SHA-256 algorithm processes the input data D in blocks and performs several bitwise operations, rotations, and modular additions. The details of the algorithm are quite complex but can be summarized mathematically as follows:

$H=\text{SHA-256}(D)=\text{SHA-256}(D0,D1,D2,\ldots,Dn)$

Where $D0,D1,D2,\ldots,Dn$ are the blocks of data.

The SHA-256 algorithm operates on each block using a series of logical functions, modular additions, bitwise shifts, and constants. It transforms the input data into the 256-bit hash H.

4. Security and Properties:

The SHA-256 algorithm is designed to have several important properties, including:

Deterministic: The same input data will always produce the same hash value.

Pre-image resistance: It should be computationally infeasible to reverse the hash function to find the original input data from its hash.

Collision resistance: It should be highly improbable that two different inputs produce the same hash value.

Avalanche effect: A small change in the input should result in a significantly different hash value.

Efficiency: The algorithm should be computationally efficient, making it difficult to find two different inputs with the same hash value (a collision).

These properties ensure that the SHA-256 hash is a secure and reliable way to represent data in a tamper-evident manner on a blockchain.

1. User Registration & Trained Voter Face:

User should register in our website *(User Voting Page Way)* Block chain as an initial step with their mobile name, email, aadhar id, Voter id , image Area, Block chain Address contact number to which an unique *USER-ID* register. Users who are all registered in this portal are also considered as voter. The voter image convert to trained image After registering successfully the admin verify the voter details, after user can login into their profile using their *USER-ID* and their registered password.

Admin:

Admin Login page with default user name and password. Admin can accept or reject an voter request by verifying the user detail and also admin can register another admin. User has to scan his aadhar card for verification process. After scanning he should enter his detail and send an request to the admin if the account get rejected due to some reason he will be intimated to register again by admin.

2. Create Election

The Admin can create an election with election type and election constituency. All the election gets triggered at the given date and time. And Verified user has to login and scan his Block chain Address if election and user constituency matches user can view Election details. And Block chain Address. To create Nominated account in block chain.

3. Voting

Voters must have access to any web browser to take part in voting. The voter's interface would be provided in English language to make it easy to use for all users. The proposed system can contain a large number of voters at the time of voting. A decentralized block chain system enables a voter to vote from any part of the world. A person can take part in voting from anywhere, even if he is in a foreign country, in this way his/her computerized National ID is verified from the national database so he can cast the vote. User has to face his registered finger during his registration process. In voting page voter has to scan his face if the User Face matches with registered Face , voter can cast his or her vote to the right candidate Source KNN an algorithm for recognition of human face is used to compare two Face. Voting transactions are sent to a pool from which miners analyze them and remove the malicious request by taking the consensus from the other nodes before adding it to the chain. The votes are fully secured using a cryptographic hash. Each vote cast adds a new block in the chain. When the transaction completes and a node is successfully added to Vote Chain, the voter of that particular voting transaction is notified through an SMS to his registered email. The voter has provided with a unique transaction hash by which he can verify his vote through a web portal and upon successfully completion of transaction the vote has been counted in the whole voting activity.

4. Publish Result

Smart contracts are providing a secure connection between the user and the network while executing a transaction in the chain. These are the rules that are implemented on the entire Block chain and cannot be neglected under any condition. All the nodes have to follow the smart contracts to save the vote in the system successfully. When user completes his or her voting process votes are stored in Block chain. So the voter can trust his votes stored in block chain cannot be changed. User can view his or her vote in a pie chart retrieved from block chain.SHA256 algorithm has been used to hash the data. Admin Can publish the result of each constituency after the election process is fully completed.

2.2 PROBLEM STATEMENT

Creating a robust facial recognition system that can accurately verify a voter's identity, ensuring that each eligible voter can only cast one vote and minimizing the potential for impersonation or fraud. Employing blockchain technology to establish an immutable and transparent ledger for recording votes. This ledger must be resistant to tampering and allow for public verification of the results, enhancing trust in the election process. Employing blockchain technology to establish an immutable and transparent ledger for recording votes. This ledger must be resistant to tampering and allow for public verification of the results, enhancing trust in the election process[1].

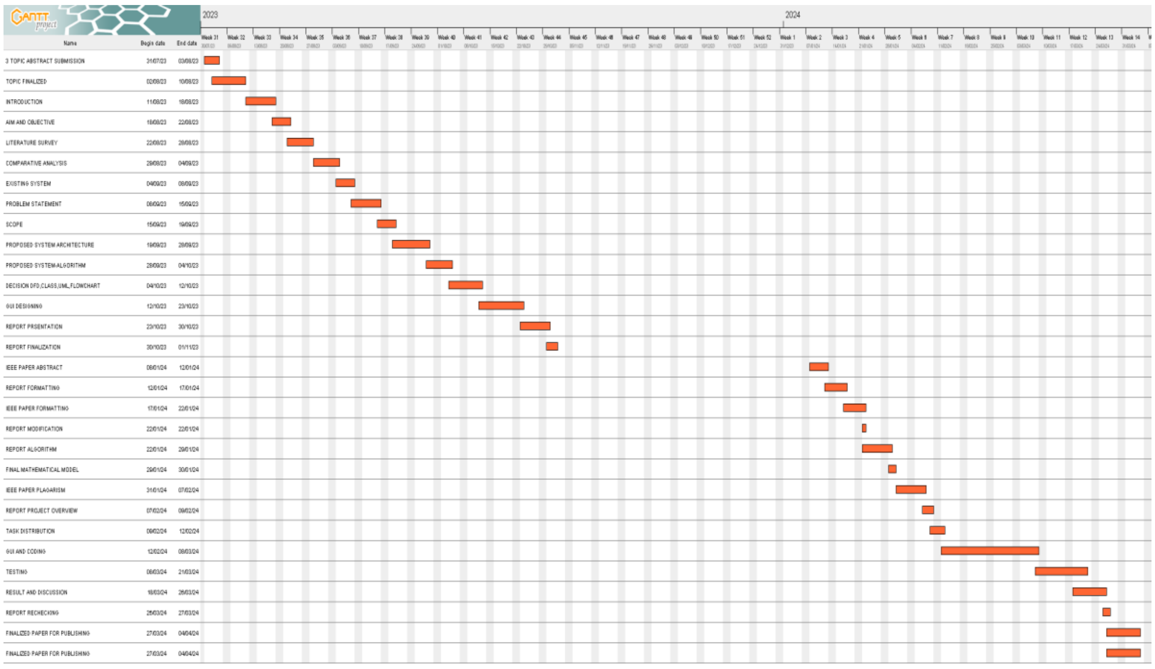
2.3 SYSTEM OVERVIEW

- **User** :
  User should register in our website *(User Voting Page Way)* Block chain as an initial step with their mobile name, email, aadhar id, Voter id , image Area, Block chain Address contact number to which an unique *USER-ID* register. Users who are all registered in this portal are also considered as voter. The voter image convert to trained image After registering successfully the admin verify the voter details, after user can login into their profile using their *USER-ID* and their registered password
- **Admin:**
  Admin Login page with default user name and password. Admin can accept or reject an voter request by verifying the user detail and also admin can register another admin. User has to scan his aadhar card for verification process. After scanning he should enter his detail and send an request to the admin if the account get rejected due to some reason he will be intimated to register again by admin.
- **Create Election**
  The Admin can create an election with election type and election constituency. All the election gets triggered at the given date and time. And Verified user has to login and scan his Block chain Address if election and user constituency matches user can view Election details. And Block chain Address. To create Nominated account in block chain.
- **Publish Result**

➢ Smart contracts are providing a secure connection between the user and the network while executing a transaction in the chain. These are the rules that are implemented on the entire

➢ Block chain and cannot be neglected under any condition. All the nodes have to follow the smart contracts to save the vote in the system successfully. When user completes his or her voting process votes are stored in Block chain. So the voter can trust his votes stored in block chain cannot be changed. User can view his or her vote in a pie chart retrieved from block chain.SHA256 algorithm has been used to hash the data. Admin Can publish the result of each constituency after the election process is fully completed[1].

2.4 PROJECT TIMELINE CHART



Name	Begin date	End date
3 TOPIC ABSTRACT SUBMISSION	31/07/23	03/08/23
TOPIC FINALIZED	02/08/23	10/08/23
INTRODUCTION	11/08/23	18/08/23
AIM AND OBJECTIVE	18/08/23	22/08/23
LITERATURE SURVEY	22/08/23	28/08/23
COMPARATIVE ANALYSIS	29/08/23	04/09/23
EXISTING SYSTEM	04/09/23	08/09/23
PROBLEM STATEMENT	08/09/23	15/09/23
SCOPE	15/09/23	19/09/23
PROPOSED SYSTEM ARCHITECTURE	19/09/23	28/09/23
PROPOSED SYSTEM-ALGORITHM	28/09/23	04/10/23
DECISION DFD,CLASS,UML,FLOWCHART	04/10/23	12/10/23
GUI DESIGNING	12/10/23	23/10/23
REPORT PRSENTATION	23/10/23	30/10/23
REPORT FINALIZATION	30/10/23	01/11/23
IEEE PAPER ABSTRACT	08/01/24	12/01/24
REPORT FORMATTING	12/01/24	17/01/24
IEEE PAPER FORMATTING	17/01/24	22/01/24
REPORT MODIFICATION	22/01/24	22/01/24
REPORT ALGORITHM	22/01/24	29/01/24
FINAL MATHEMATICAL MODEL	29/01/24	30/01/24
IEEE PAPER PLAGARISM	31/01/24	07/02/24
REPORT PROJECT OVERVIEW	07/02/24	09/02/24
TASK DISTRIBUTION	09/02/24	12/02/24
GUI AND CODING	12/02/24	08/03/24
TESTING	08/03/24	21/03/24
RESULT AND DISCUSSION	18/03/24	26/03/24
REPORT RECHECKING	26/03/24	27/03/24
FINALIZED PAPER FOR PUBLISHING	27/03/24	04/04/24
FINALIZED PAPER FOR PUBLISHING	27/03/24	04/04/24

Fig.2.4 Project Timeline Chart

2.5 TASK DISTRIBUTION

2.5.1 DESIGN PHASE

Table No 2.5.1: Design Phase

Name of Student	**Task Performed**	**Result**
Gaurav V. Jadhav	Project Design	Completed Successfully
Aakash L. Desale	Suggested Appropriate contents	Completed Successfully
Nitesh N. Sawardekar	Documentation	Completed Successfully

2.5.2 IMPLEMENTATION PHASE

Table No 2.5.2: Implementation Phase

Name of Student	**Implementation Task**	**Result**
Gaurav V. Jadhav	Project Implementation	Completed Successfully
Aakash L. Desale	Implemented Project Algorithm.	Completed Successfully
Nitesh N. Sawardekar	Documentation	Completed Successfully

SOFTWARE REQUIRMENT SPECIFICATION

3. SOFTWARE REQUIREMENT SPECIFICATION

3.1 HARDWARE REQUIREMENTS

System	:	Intel Core i3 2.00 GHz.
Hard Disk	:	1 TB.
Monitor	:	14’ Color Monitor.
Mouse	:	Optical Mouse.
Ram	:	4 GB.
Keyboard	:	101 Keyboard Keys.

3.2 SOFTWARE REQUIREMENTS

Operating system	:	Windows 10 and above.
Coding Language	:	Python
Software’s used	:	eclipse IDE, Python 3.6.3
XAMPP, Ganache.		
Languages Used	:	JavaScript, HTML, CSS, PHP, Python, Solidity
Technology Used	:	Blockchain
Algorithms	:	SHA-256,
		KNN (K-NEAREST NEIGHBOURS)

SYSTEM DESIGN

4. SYSTEM DESIGN

4.1 DESIGN SPECIFICATION

4.1.1 ALGORITHM

Step 1: Start

Step.2: Face Recognition using k-nearest neighbors

```
def predict(self, X):
y_pred = [self._predict(x) for x in X]
return np.array(y_pred)
def _predict(self, x):
distances = [euclidean_distance(x, x_train) for x_train in self.X_train]
k_indices = np.argsort(distance)[:self.k]
k_nearst_labels = [self.y-train[i] for i in k_indies]
most_common= np.bincount(k_nearest_labels).argmax()
return most_common
```

Step 3: Voter Registration using Blockchain address

```
Require: Initialization of parameters
Initialize voter id = this voter_id
Initialize voter name = this voter_name
Func (Register Voter)
Input: voter id
Require: voter_id =! Null
If voter_id exist
then revert back to voter id else
if voter_age < 18
then revert back to voter id else
Add Voter successfully
End Func
End Smart Contract
```

Step 4: Voter identification Vote casting using SHA-256:

```
mapping(address => Voter) public voters;
event Voted(adress indexed voter, uint256 vote);
function vote(uint256 _vote) public {
require(!voters[msg.sender].hasVoted, "Voter has already voted");
voters[msg.sender].hasVoted = true;
voters[msg.sender].vote = _vote;
emit Voted(msg.sender, _vote);}
```

Step 5 : Stop

4.1.2 WORKING OF ALGORITHM

The general idea of working of proposed system algorithm is given as bellows:

Step 1: Start

This is just the beginning of the algorithm, indicating the start of the process.

Step 2: Face Recognition using k-nearest neighbors

This step involves face recognition using the k-nearest neighbors algorithm. The algorithm predicts the label of a given input image by finding the k nearest images in the training dataset and taking a majority vote among their labels.

Step 3: Voter Registration using Blockchain address

This step involves registering a voter using a blockchain address. It seems to define a smart contract that initializes parameters such as voter ID and name, and then provides a function to register a voter by checking if the ID exists and if the age is above 18. If the conditions are met, the voter is successfully added to the registry.

Step 4: Voter identification and Vote casting using SHA-256

This step utilizes blockchain technology for voter identification and vote casting. It defines a mapping between addresses and voters in a smart contract. The vote function allows a voter to cast their vote by specifying their choice. It checks if the voter has already voted, records their vote, and emits an event indicating the vote.

Step 5: Stop

This marks the end of the algorithm.

4.2 SYSTEM ARCHITECTURE

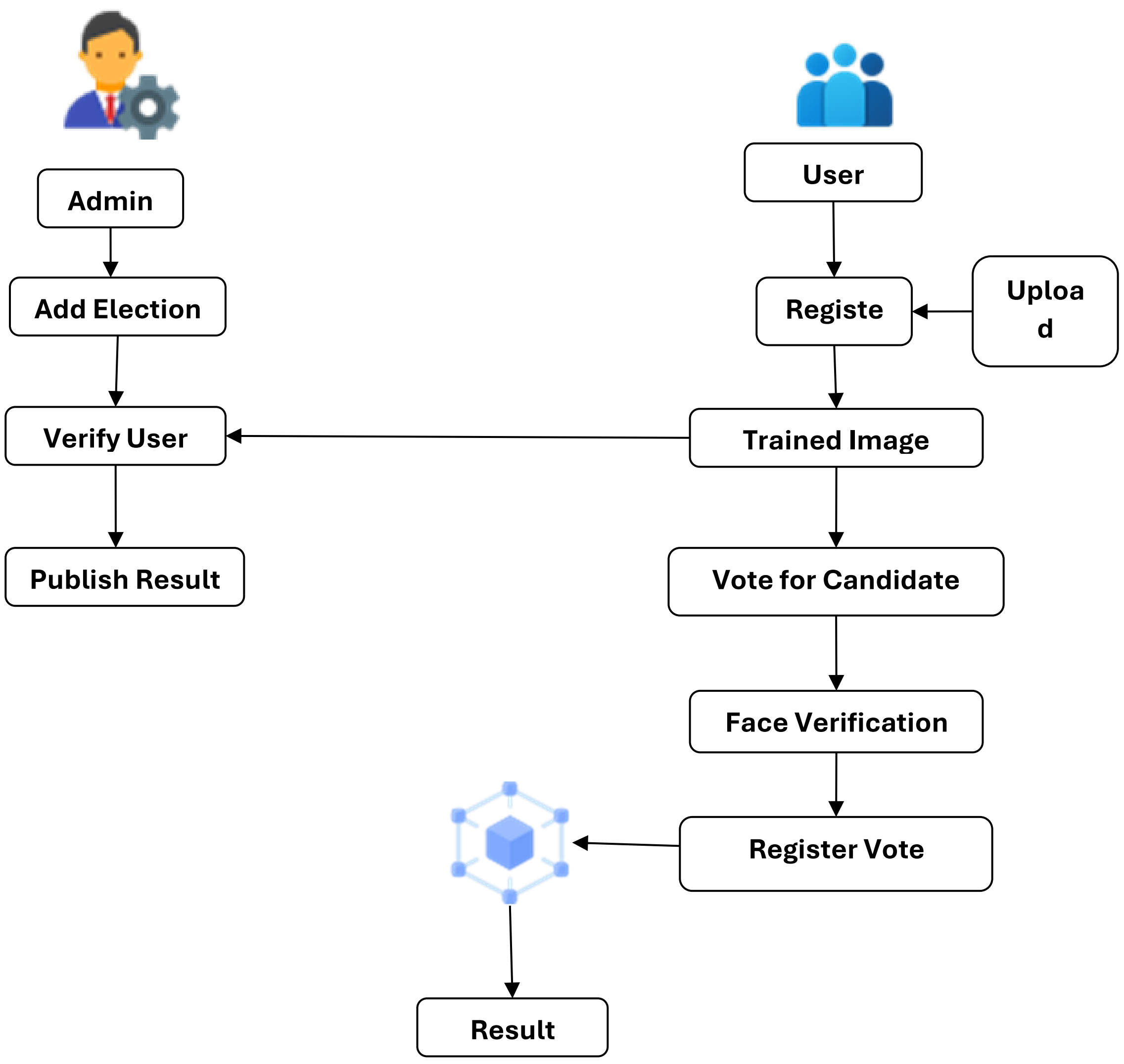


Fig.4.2: System Architecture for e-Voting System

4.3 DATA FLOW DIAGRAM

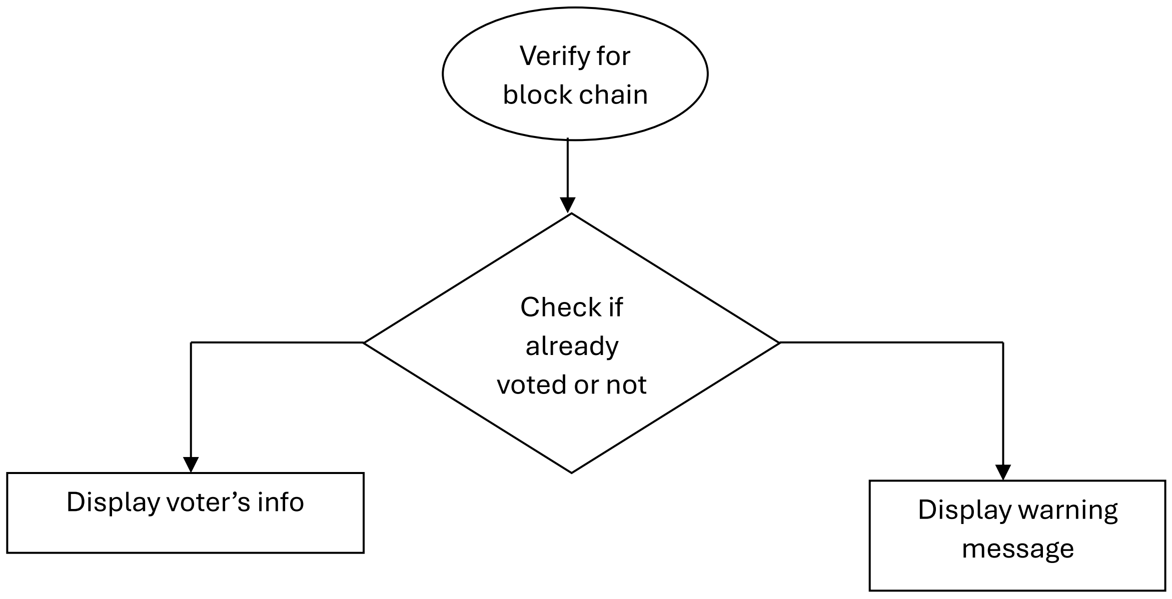


Fig. 4.3.1: DFD Level 0 for e-Voting System

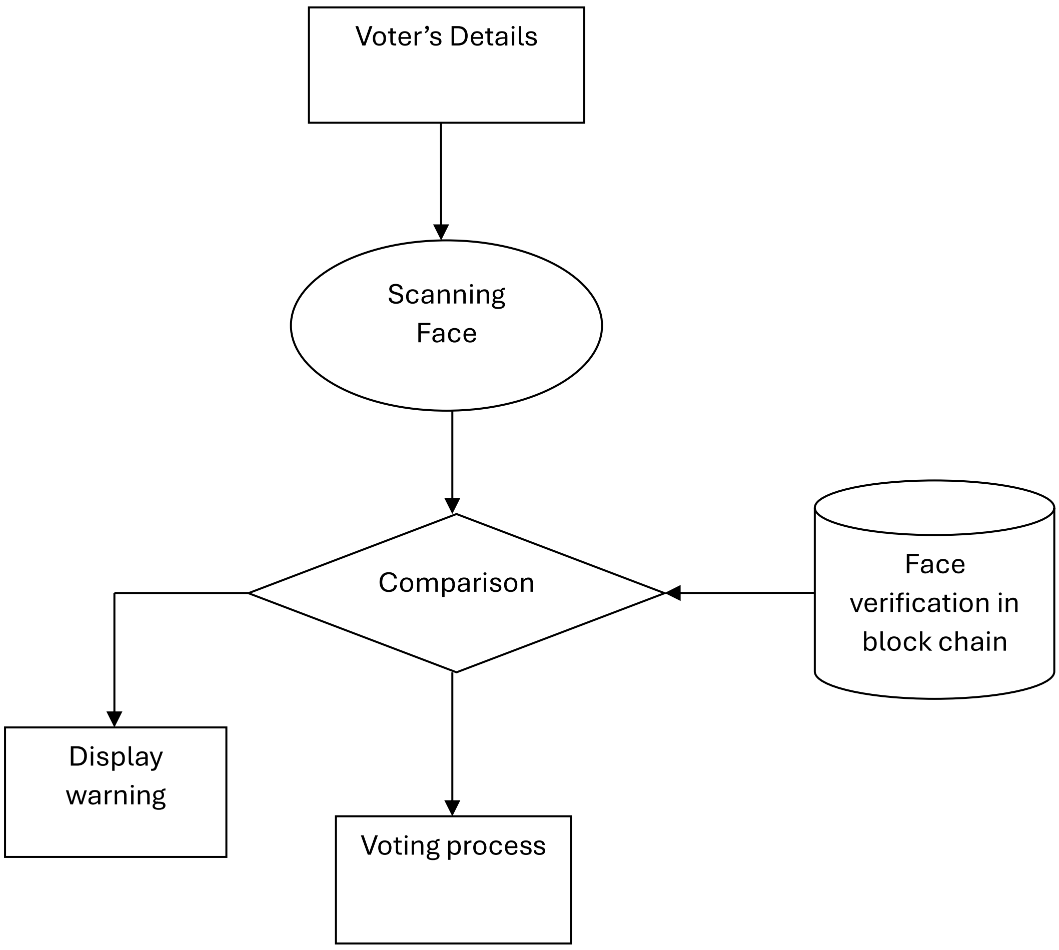


Fig. 4.3.2: DFD Level 1 for e-Voting System

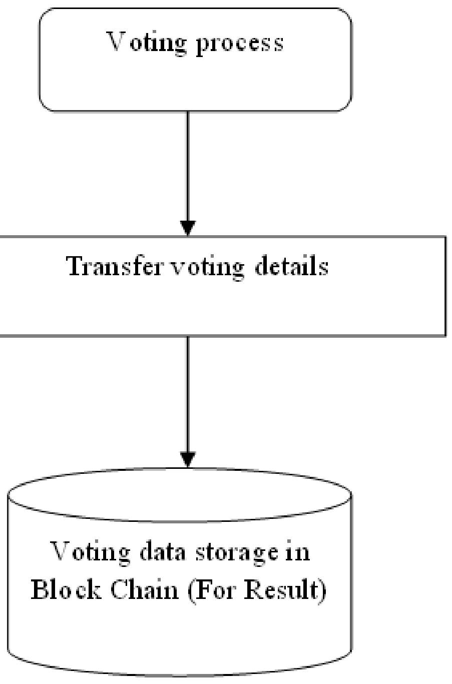


Fig. 4.3.3: DFD Level 2 for e-Voting System

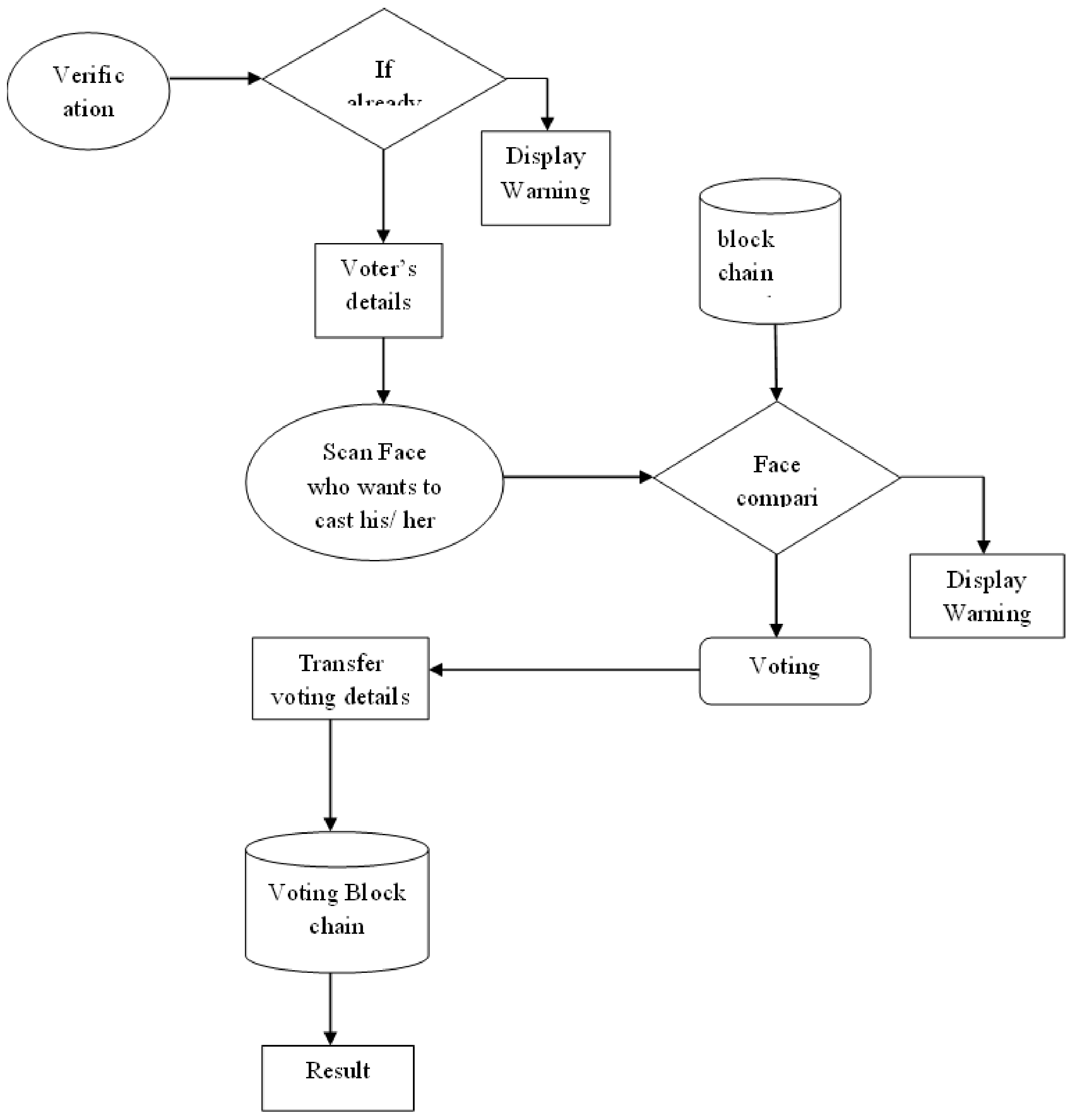


Fig. 4.3.4: DFD Level 3 for e-Voting System

4.4 FLOW CHART

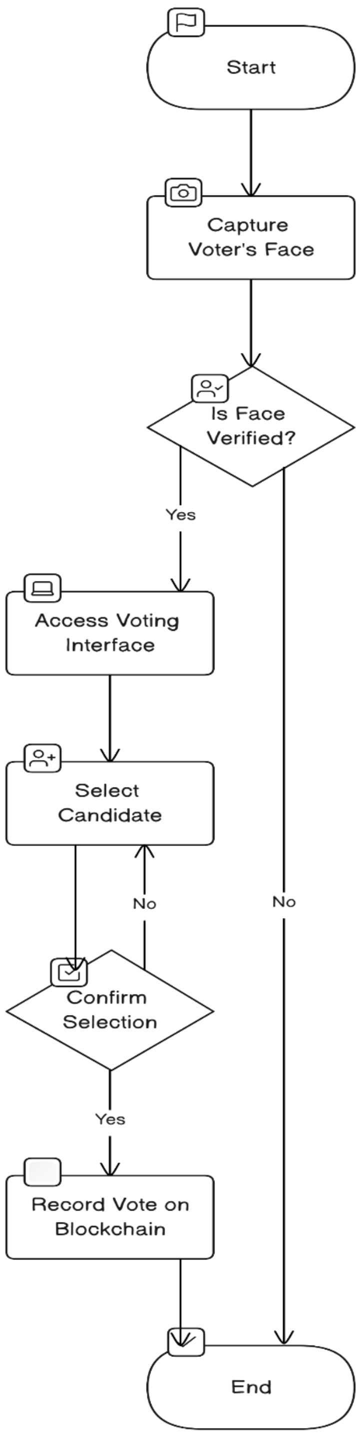


Fig. 4.4: Flowchart of e-Voting System

4.5 UML DIAGRAMS

4.5.1 CLASS DIAGRAM

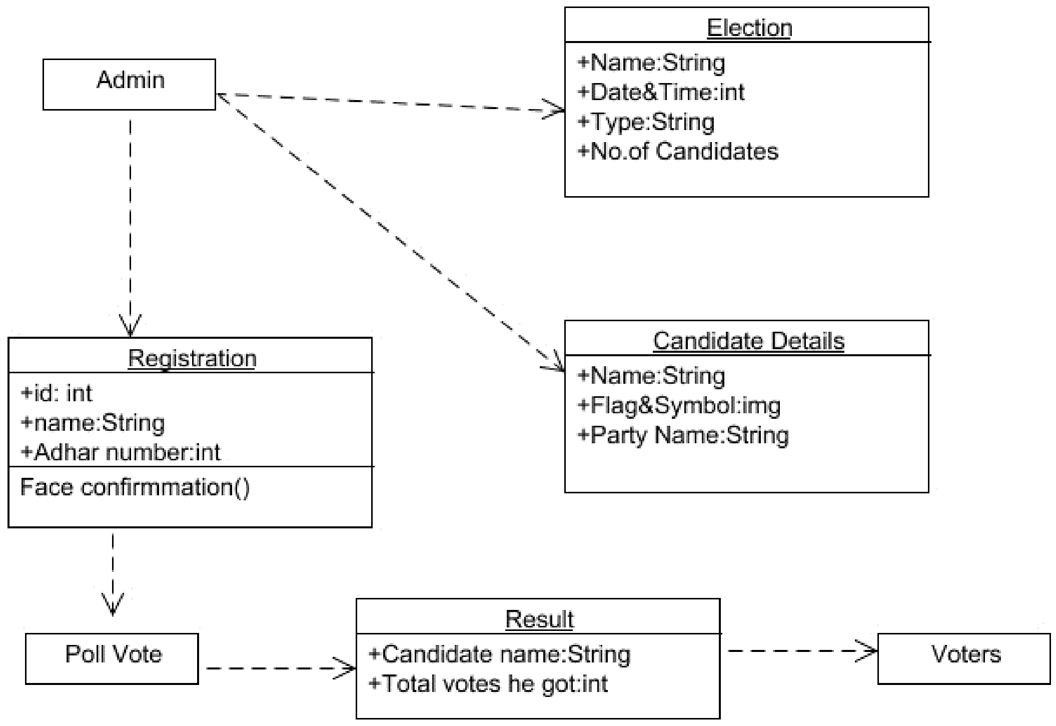


Fig. 4.5.1 Class Diagram for e-Voting System

4.5.2 SEQUENCE DIAGRAM

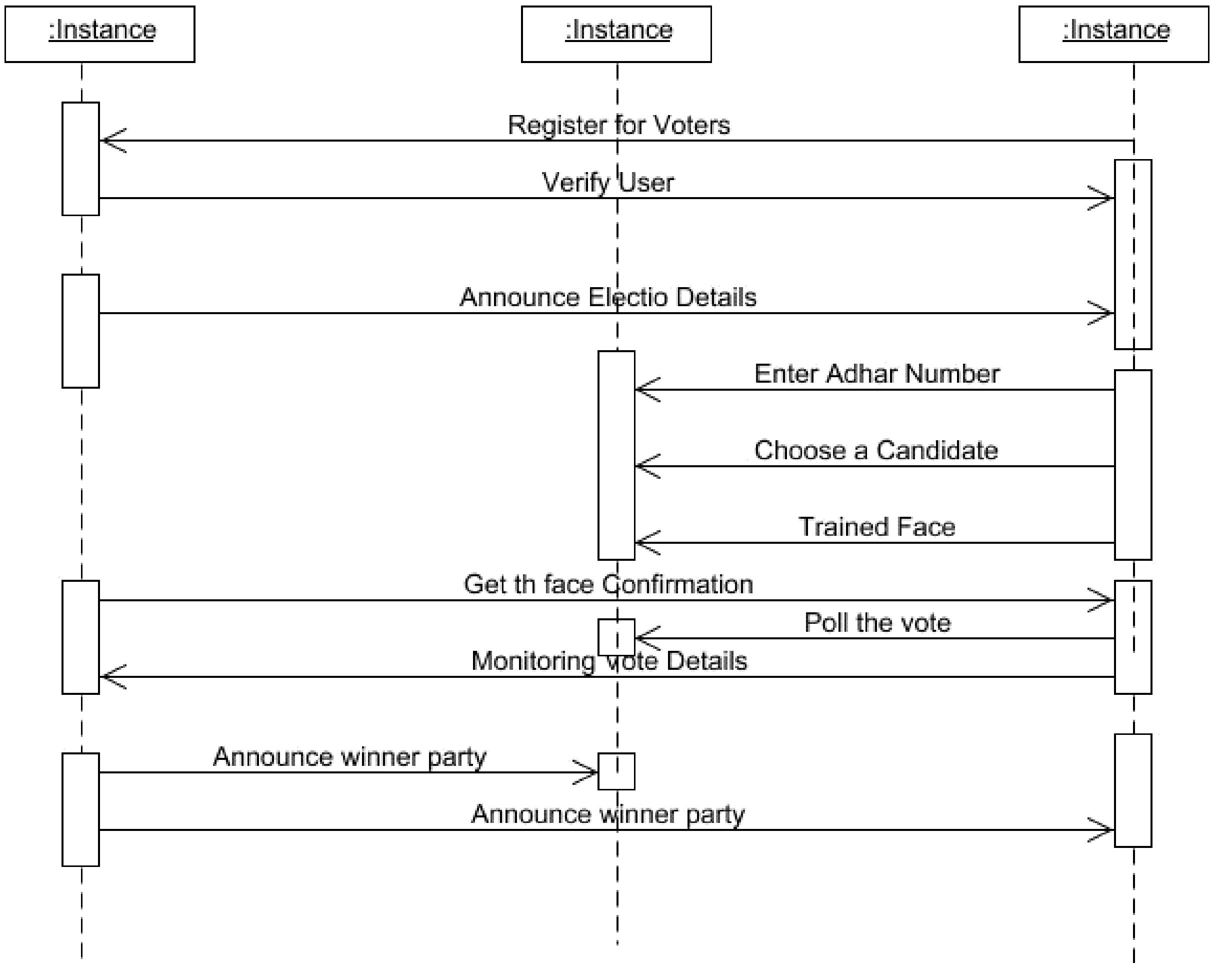


Fig. 4.5.2 Sequence Diagram for e-Voting System

4.5.3 Use Case Diagram

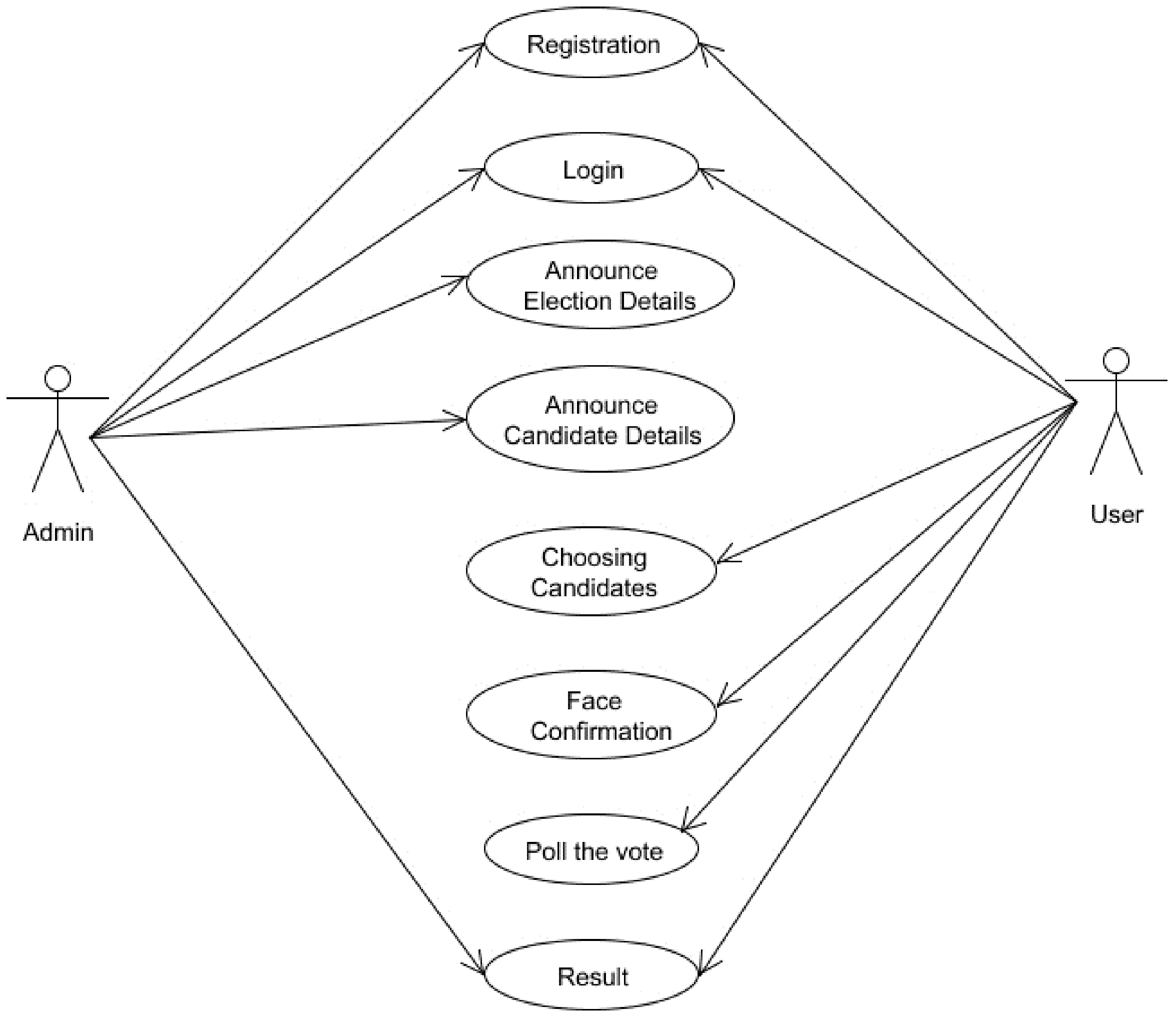


Fig. 4.5.3 Use Case Diagram for e-Voting System

4.5.4 ACTIVITY DIAGRAM

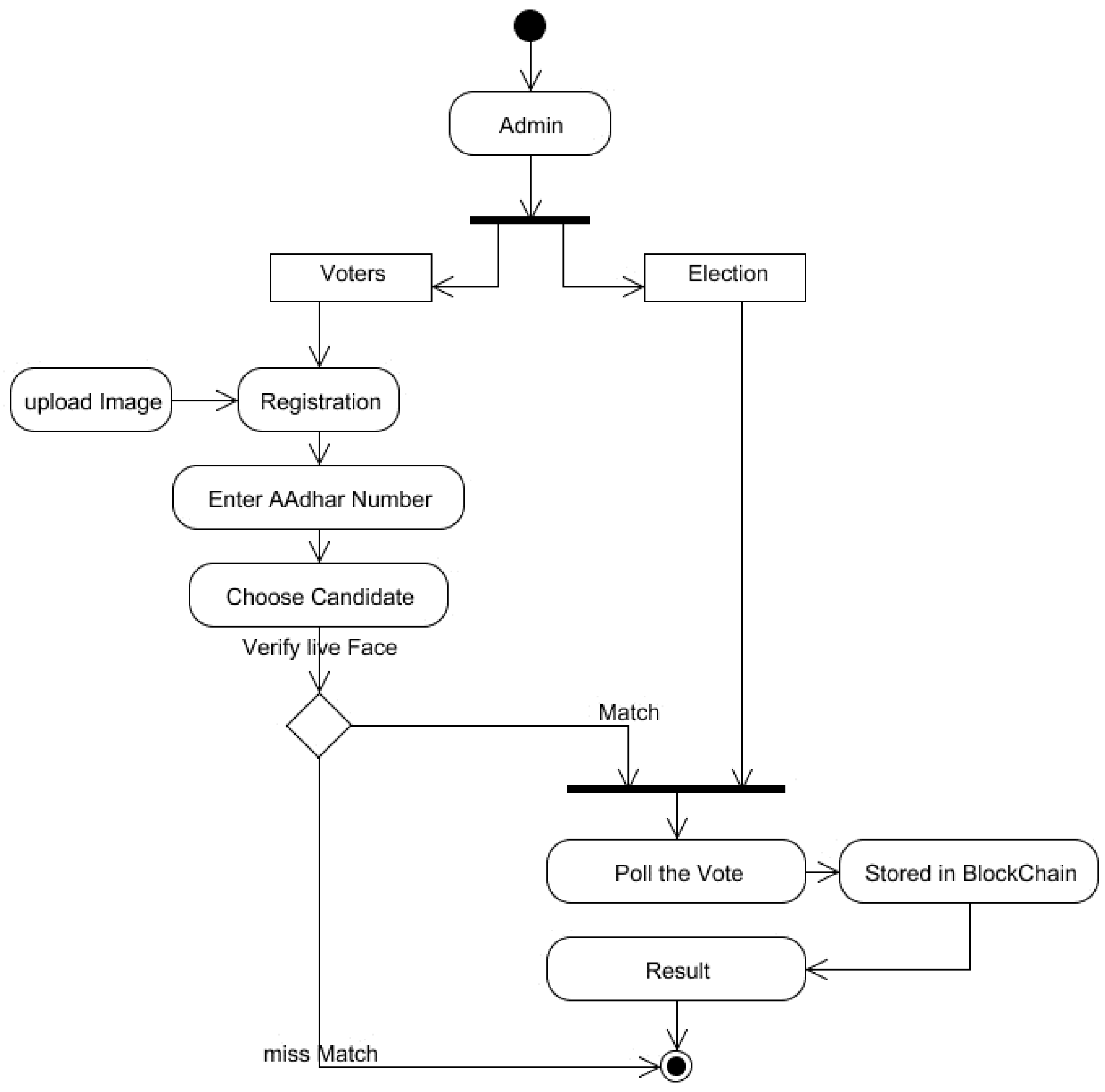


Fig. 4.5.4 Activity Diagram e-Voting System

4.5.5 COLLABORATION DIAGRAM

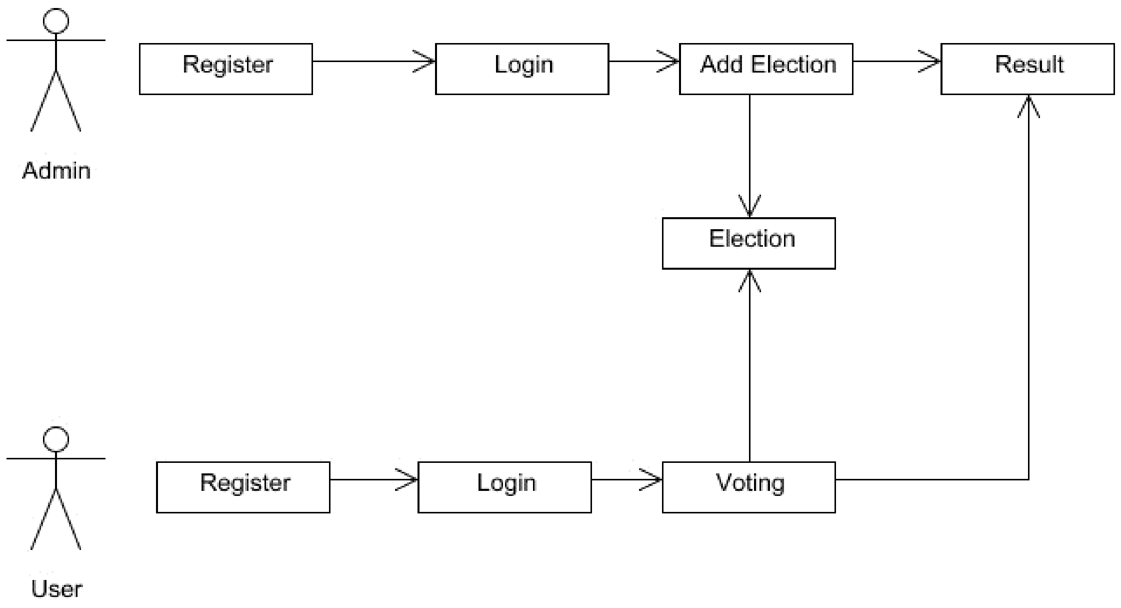


Fig. 4.5.5 Collaboration Diagram for e-Voting System

PROJECT IMPLEMENTATION

5. PROJECT IMPLEMENTATION

5.1 TECHNOLOGY OVERVIEW

PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating system. CPython, the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit python software foundation. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object oriented , imperative, functional and procedural, and has a large and comprehensive standard library.

Bloch-chain

Blockchain is a distributed ledger technology that enables secure, transparent, and immutable record-keeping of transactions across a network of computers. Each block in the chain contains a number of transactions, and every time a new transaction occurs, a record of that transaction is added to every participant's ledger.
Key Features:
Decentralization: Unlike traditional centralized systems where a single authority controls the data, blockchain operates on a decentralized network of computers (nodes). This eliminates the need for a central authority, making the system more resilient and less prone to manipulation.
Transparency: The entire transaction history is recorded on a public ledger that is accessible to all participants. This transparency helps to build trust among users as they can verify the authenticity of transactions.

Immutability: Once a transaction is recorded on the blockchain, it cannot be altered or deleted. Each block contains a unique cryptographic hash of the previous block, making it extremely difficult to tamper with historical data without the consensus of the majority of participants.

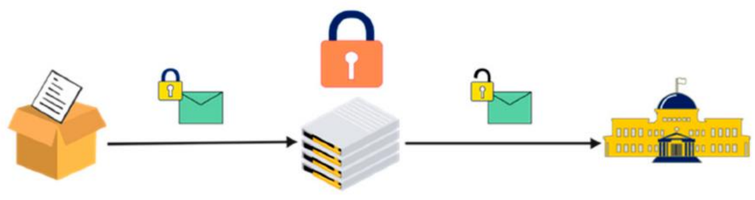
Security: Blockchain uses cryptographic techniques to secure transactions and protect the integrity of the network. Each participant has a private key to sign transactions, and consensus mechanisms ensure that only valid transactions are added to the ledger.

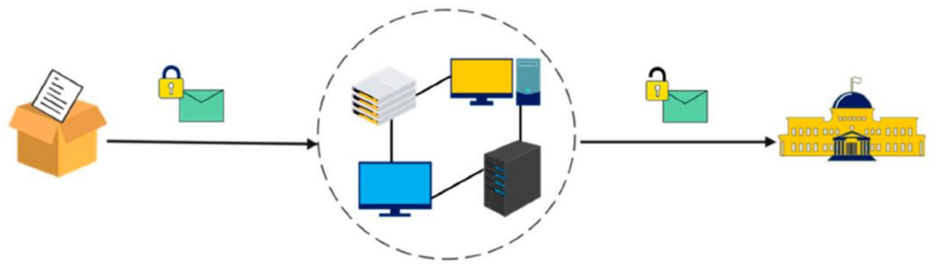
Types of Blockchain:

Public Blockchain: Anyone can participate in a public blockchain network, and the data is accessible to anyone. Bitcoin and Ethereum are examples of public blockchains.

Private Blockchain: In a private blockchain, access is restricted to a specific group of participants. It is often used by businesses and organizations to maintain privacy and control over the network.

Consortium Blockchain: A consortium blockchain is controlled by a group of organizations rather than a single entity. It offers a balance between the openness of public blockchains and the control of private blockchains.


Traditional Voting System


Blockchain Voting System

5.2 CODING

face_recognition_knn_web.py

```
import os
import json
from math import sqrt
#from sklearn import neighbors
from sklearn.neighbors import KNeighborsClassifier
from os import listdir
from os.path import isdir, join, isfile, splitext
import pickle
from PIL import Image, ImageFont, ImageDraw, ImageEnhance
import face_recognition
from face_recognition import face_locations
from face_recognition.cli import image_files_in_folder
#This is not working for me
#from face_recognition.face_recognition_cli import  image_files_in_folder
#comment this line if you got error
from flask import Flask, jsonify, request, redirect,render_template
from flask_cors import CORS
from os import path
 ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg'}
REGISTER_PATH="./Register/"
TEMP_PATH="./Temp/"
if not os.path.exists(REGISTER_PATH):
   os.makedirs(REGISTER_PATH )
app = Flask(__name__)
CORS(app)
def allowed_file(filename):
   return '.' in filename and \
          filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
@app.route('/register', methods=['GET', 'POST'])
```

```
def upload_register():
    # Check if a valid image file was uploaded
    if request.method == 'POST':
        if 'file' not in request.files:
            return json.dumps({"status": "Error", "msg": "Image cannot be empty "})
        name = request.form.get('name')
        email = request.form.get('email')
        if(name ==''):
            return json.dumps({"status": "Error", "msg": "Name cannot be empty "})

        file = request.files['file']
       # src = path.realpath(REGISTER_PATH+"/"+email+"/"+name+"/"+file.filename)

        print(name)
        print(file)
        if file.filename == '':
            return json.dumps({"status": "Error", "msg": "Image cannot be empty "})

        if file and allowed_file(file.filename):
            if os.path.exists(REGISTER_PATH+email):

                #file.save(REGISTER_PATH+email+"/"+file.filename)
                foo = Image.open(file)
                #foo = foo.rotate(90, PIL.Image.NEAREST, expand = 1)

foo.save(REGISTER_PATH+email+"/"+file.filename.strip(),optimize=True,quality=85)

os.rename(REGISTER_PATH+email+"/"+file.filename,REGISTER_PATH+email+"/"+na
me+".jpg")

                #return json.dumps({"status": "Error", "msg": "User Already Registered"})
            else:
```

```
 os.makedirs(REGISTER_PATH+email)
               foo = Image.open(file)
               #file.save(REGISTER_PATH+email+"/"+file.filename)
               #foo = foo.rotate(90, PIL.Image.NEAREST, expand = 1)

foo.save(REGISTER_PATH+email+"/"+file.filename.strip(),optimize=True,quality=85)

os.rename(REGISTER_PATH+email+"/"+file.filename,REGISTER_PATH+email+"/"+na
me+".jpg")

print(file)
            return json.dumps(train(REGISTER_PATH))
        else:
            return json.dumps({"status": "Error", "msg": "Image Format not supported
<png,jpg,jpeg> "})
    # If no valid image file was uploaded, show the file upload form:
    return json.dumps ({"status":"Error","msg":"GET Not Allowed "})
@app.route('/', methods=['GET', 'POST'])
def upload_image():
    # Check if a valid image file was uploaded
    if request.method == 'POST':
        if 'file' not in request.files:
            return redirect(request.url)

        file = request.files['file']

        if file.filename == '':
            return redirect(request.url)
        foo = Image.open(file)

   if file and allowed_file(file.filename):
            # The image file seems valid! Detect faces and return the result.
```

```
  print(file)
          #foo = foo.rotate(90, PIL.Image.NEAREST, expand = 1)
          foo.save(TEMP_PATH+"/"+file.filename.strip(),optimize=True,quality=85)
          return json.dumps(predict(file))

    # If no valid image file was uploaded, show the file upload form:
    return render_template('index.html')
def predict(X_img_path, knn_clf = None, model_save_path ="train", DIST_THRESH = .5):
    # if not isfile(X_img_path) or splitext(X_img_path)[1][1:] not in
ALLOWED_EXTENSIONS:
    #    raise Exception("invalid image path: {}".format(X_img_path))
    if knn_clf is None:
        with open(model_save_path, 'rb') as f:
            knn_clf = pickle.load(f)
    # X_img = face_recognition.load_image_file(X_img_path)
    #X_img = face_recognition.load_image_file(X_img_path)
    X_img = face_recognition.load_image_file(TEMP_PATH+X_img_path.filename)
    X_faces_loc = face_locations(X_img)
    if len(X_faces_loc) == 0:
        return []
    faces_encodings = face_recognition.face_encodings(X_img,
known_face_locations=X_faces_loc)
    closest_distances = knn_clf.kneighbors(faces_encodings, n_neighbors=1)
    is_recognized = [closest_distances[0][i][0] <= DIST_THRESH for i in
range(len(X_faces_loc))]
    return [{"result":pred,"status":str(rec)} if rec else {"result":"N/A","status":str(rec)} for
pred, loc, rec in zip(knn_clf.predict(faces_encodings), X_faces_loc, is_recognized)]
def train(train_dir, model_save_path="train", n_neighbors=None, knn_algo='ball_tree',
verbose=False):
    X = []
    y = []
    for class_dir in listdir(train_dir):
        if not isdir(join(train_dir, class_dir)):
            continue
        for img_path in image_files_in_folder(join(train_dir, class_dir)):
            image = face_recognition.load_image_file(img_path)
            faces_bboxes = face_locations(image)
            if len(faces_bboxes) != 1:
                if verbose:
                    print("image {} not fit for training: {}".format(img_path, "didn't find a face" if
len(
                        faces_bboxes) < 1 else "found more than one face"))
                continue
            X.append(face_recognition.face_encodings(image,
known_face_locations=faces_bboxes)[0])
            y.append(class_dir)

    if n_neighbors is None:
        n_neighbors = int(round(sqrt(len(X))))
        if verbose:
            print("Chose n_neighbors automatically as:", n_neighbors)
    #knn_clf = neighbors.KNeighborsClassifier(n_neighbors=n_neighbors,
algorithm=knn_algo, weights='distance')
    knn_clf = KNeighborsClassifier(n_neighbors=n_neighbors, algorithm=knn_algo,
weights='distance')
    knn_clf.fit(X, y)

    if model_save_path != "":
        with open(model_save_path, 'wb') as f:
            pickle.dump(knn_clf, f)
    return {"status":"Success","msg":"Trained successfully"}
if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5001, debug=True)
```

TESTING

6. TESTING

6.1 SYSTEM TESTING

Table no. 6.1: Test Cases

Sr no.	Test Case	Expected Result	Result	Remark (if fail)
1	User Register	If User registration successfully.	Pass	If already user email exist, then it fails.
2	User Login	If Username and password is correct, then it will getting valid page.	Pass	Un Register Users will not logged in.
3	Admin login	Admin can login with his login credential. If success he get his home page	Pass	Invalid login details will not allowed here
4	Admin can activate the register users	Admin can activate the register user id	Pass	If user id not found then it won’t login
7	Blockchain public Address	Obtaining Unique Hash value	Pass	Receiving Unique Hash Value using Blockchain from Ganache Server
5	Face Recognition	If face is Recognised	Pass	If Face is not Valid then voter cannot cast vote
6	Input Aadhar Feed by the user	Valid Aadhar number is accepted as enrolled in Database	Pass	Invalid User Not Allowed

RESULTS AND DISCUSSIONS

7. RESULTS AND DISCUSSIONS-

7.1 RESULTS SETS

TABLE 7.1.1 TESTING PERFORMANCE ON TRAIN SET

Testing with k = 1

Test	Precision	Recall	Fl-score	Support
A	1.0	0.50	0.67	2.00
B	0.67	1.00	0.80	2.00
C	1.0	1.00	1.0	2.00
D	0.67	1.00	0.80	2.00
E	1.0	0.50	0.67	2.00
F	0.67	1.00	0.80	2.00
G	1.0	1.00	1.0	2.00
H	1.00	0.50	0.67	2.00
I	1.00	1.0	1.0	2.00
H	0.50	1.00	0.67	2.00
K	1.0	0.50	0.67	2.00
L	1.0	1.00	1.0	2.00
M	0.50	0.50	0.50	2.00
N	1.0	1.00	1.0	2.00
O	1.0	1.00	1.0	2.00
P	0.33	0.50	0.40	2.00
Q	0.67	1.0	0.80	2.00
R	1.0	1.0	1.0	2.00
S	1.0	1.0	1.0	2.00
T	0.50	1.0	0.67	2.00
Average	**0.83**	**0.85**	**0.81**	**2.00**

TABLE7.1.2 TESTING PERFORMANCE ON VALIDATION SET

Testing with k = 2

Test	**Precision**	**Recall**	**Fl-score**	**Support**
A	0.33	0.5	0.4	2
B	0.5	0.5	0.5	2
C	0.5	1	0.67	2
D	0.33	0.5	0.4	2
E	0.33	0.5	0.4	2
F	0.5	1.0	0.67	2
G	1.0	1.0	1.0	2
H	1.0	1.0	1.0	2
I	0.67	1.0	0.8	2
H	0.5	1.0	0.67	2
K	0	0	0	2
L	0.25	0.5	0.33	2
M	1.0	0.5	0.67	2
N	1.0	0.5	0.67	2
O	1.0	0.5	0.67	2
P	0.5	1.0	0.67	2
Q	0.67	1.0	0.8	2
R	0	0	0	2
S	0	0	0	2
T	0	0	0	2
Average	**0.504**	**0.6**	**0.516**	**2**

TABLE7.1.3 TESTING PERFORMANCE ON VALIDATION SET

Testing with k = 3

Test	**Precision**	**Recall**	**Fl-score**	**Support**
A	0.67	1.0	0.8	2
B	1.0	1.0	1.0	2
C	0.5	1.0	0.67	2
D	0	0	0	2
E	0.33	0.5	0.4	2
F	0.33	1.0	0.5	2
G	1.0	0.5	0.67	2
H	1.0	0.5	0.67	2
I	0.5	1.0	0.67	2
H	0.25	0.5	0.33	2
K	0	0	0	2
L	0.33	0.5	0.4	2
M	1.0	0.5	0.67	2
N	1.0	0.5	0.67	2
O	1.0	1.0	1.0	2
P	0.33	0.5	0.4	2
Q	0.33	0.5	0.4	2
R	0	0	0	2
S	0	0	0	2
T	0.5	0.5	0.5	2
Average	**0.5035**	**0.55**	**0.4875**	**2**

7.2 SCREENSHOTS

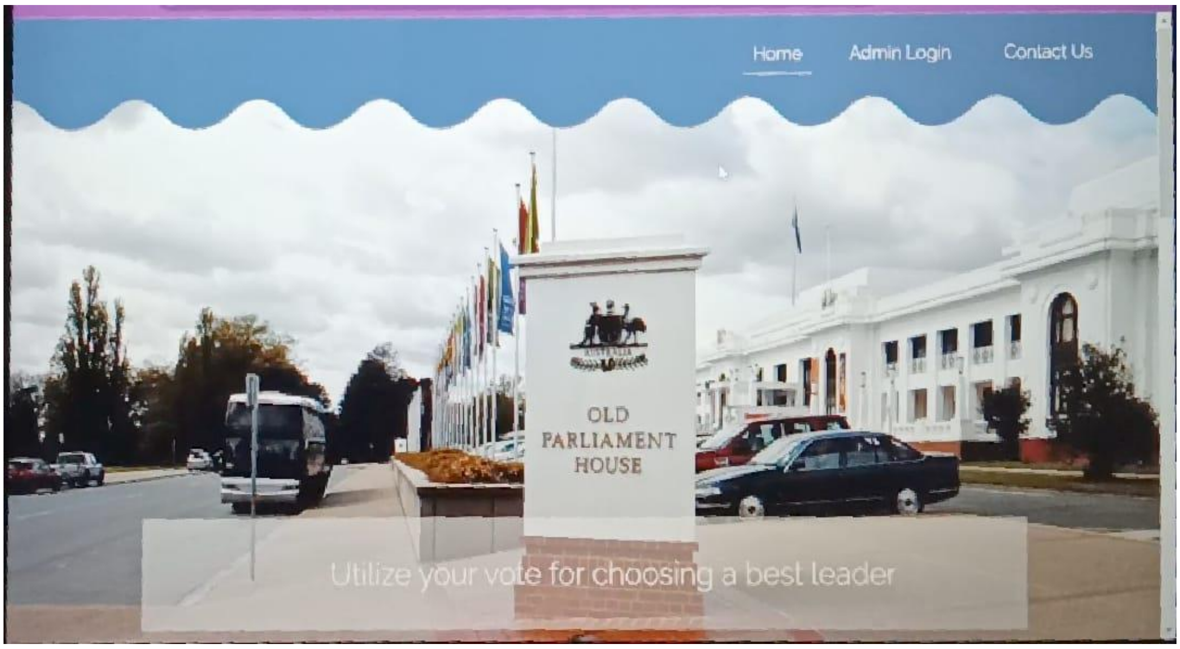
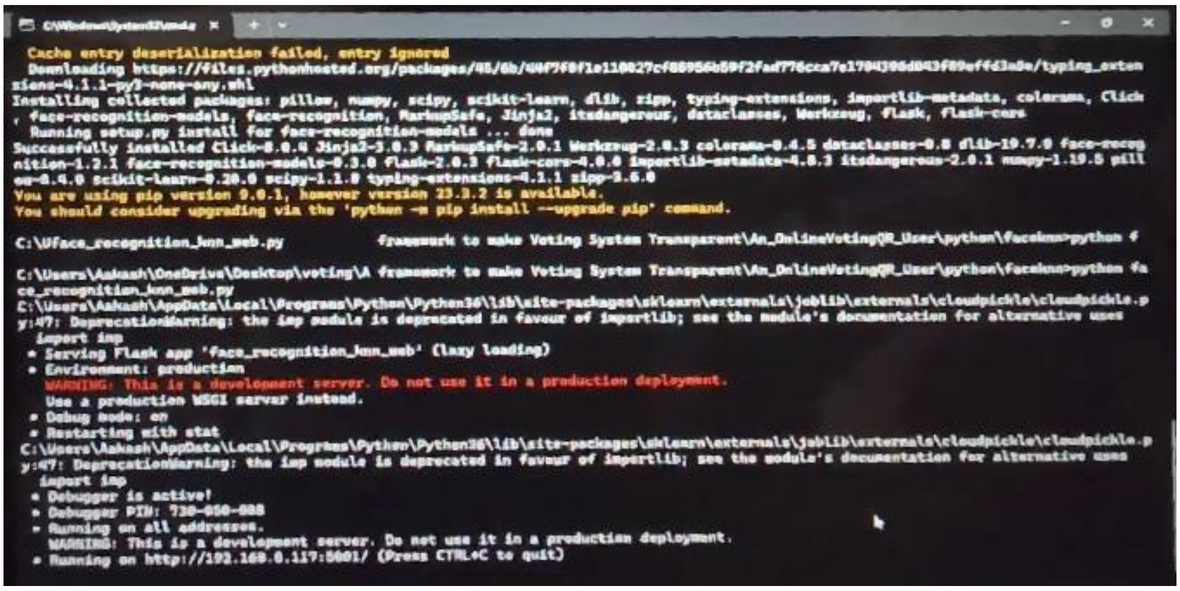


Fig.7.2.1: Home Page

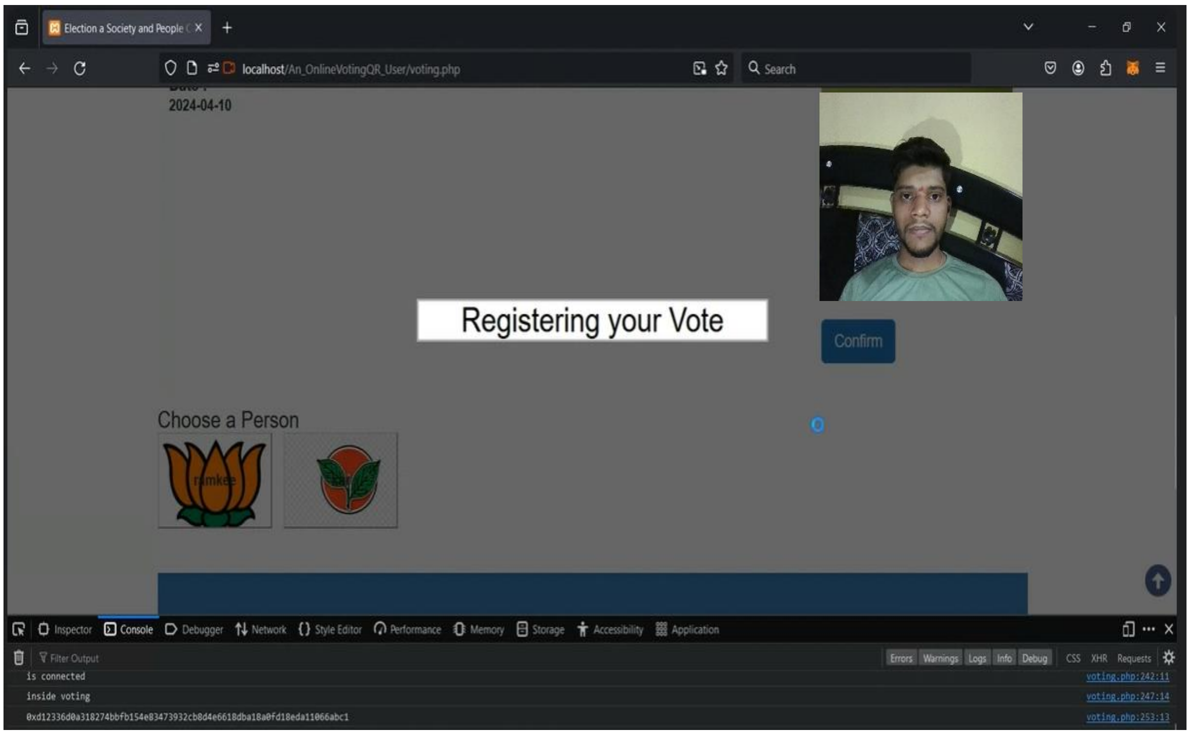


```
C:\Windows\System32\cmd.e
  Cache entry deserialization failed, entry ignored
  Downloading https://files.pythonhosted.org/packages/45/6b/44f7f8f1e110827cf88956b59f2fad776cca7e1704396d043f89eff4d3a0e/typing_exten
sions-4.1.1-py3-none-any.whl
Installing collected packages: pillow, numpy, scipy, scikit-learn, dlib, zipp, typing-extensions, importlib-metadata, colorama, Click
, face-recognition-models, face-recognition, MarkupSafe, Jinja2, itsdangerous, dataclasses, Werkzeug, flask, flask-cors
  Running setup.py install for face-recognition-models ... done
Successfully installed Click-8.0.4 Jinja2-3.0.3 MarkupSafe-2.0.1 Werkzeug-2.0.3 colorama-0.4.5 dataclasses-0.8 dlib-19.7.0 face-recog
nition-1.2.1 face-recognition-models-0.3.0 flask-2.0.3 flask-cors-4.0.0 importlib-metadata-4.8.3 itsdangerous-2.0.1 numpy-1.19.5 pill
ow-8.4.0 scikit-learn-0.20.0 scipy-1.1.0 typing-extensions-4.1.1 zipp-3.6.0
You are using pip version 9.0.1, however version 23.3.2 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Uface_recognition_knn_web.py          framework to make Voting System Transparent\An_OnlineVotingQR_User\python\faceknn>python f

C:\Users\Aakash\OneDrive\Desktop\voting\A framework to make Voting System Transparent\An_OnlineVotingQR_User\python\faceknn>python fa
ce_recognition_knn_web.py
C:\Users\Aakash\AppData\Local\Programs\Python\Python36\lib\site-packages\sklearn\externals\joblib\externals\cloudpickle\cloudpickle.p
y:47: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses
  import imp
 * Serving Flask app 'face_recognition_knn_web' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
C:\Users\Aakash\AppData\Local\Programs\Python\Python36\lib\site-packages\sklearn\externals\joblib\externals\cloudpickle\cloudpickle.p
y:47: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses
  import imp
 * Debugger is active!
 * Debugger PIN: 730-050-088
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://192.168.0.117:5001/ (Press CTRL+C to quit)
```

Fig.7.2.2: Face verification



```
is connected                                                  voting.php:242:11
inside voting                                                 voting.php:247:14
0xd12336d0a318274bbfb154e83473932cb8d4e6618dba18a0fd18eda11066abc1    voting.php:253:13
```

Fig.7.2.3: Caste Vote

ADVANTAGES

8. ADVANTAGES

- Helps solve advanced real-world issues with many constraints.
- Voters can cast their votes from anywhere in the country without visiting to voting booths, in highly secured way.
- This will increase the voting percentage in India and reduces the cost of voting process.
- Provides a path towards achieving Artificial General Intelligence sometime within the future.
- By using Face Verification, it provides enough security which reduces the false votes.
- The collection of the results is done from the stored data on the blocks through the significant organization of the nodes in the block chain.

CONCLUSION

CONCLUSION

Thus, we have tried to implement the paper “Muhammad Shoaib Farooq, Usman Iftikhar, And Adel Khelifi" A Framework to Make Voting System Transparent Using Blockchain Technology " 2022 IEEE Access and the conclusion is as follows While e-voting on the blockchain with KNN and SHA-256 offers significant advantages in terms of security and transparency the fusion of blockchain technology, KNN, and SHA-256 encryption holds great promise for revolutionizing the electoral process, making it more reliable, resilient, translucent, and digitally relevant.

FUTURE WORK

FUTURE WORK

Block chain-based voting, which relies on a decentralized, distributed digital ledger is vulnerable to many of the security flaws inherent in internet voting, such as the potential for malware to alter votes on a voter’s local device before the ballot is transmitted and the lack of secret ballots.

REFERENCES

APPENDIX