EX NO:1	WRITE THE COMPLETE PROBLEM STATEMENT
DATE	

AIM:

To prepare a PROBLEM STATEMENT for a e-voting system.

ALGORITHM:

- The problem statement is the initial starting point for a project.
- A problem statement describes what needs to be done without describing how.
- It is generally a one-to-three-page document that all project stakeholders agree upon, describing the goals of the project at a high level.
- The problem statement is intended for a broad audience and should be written in non-technical terms.
- It helps both technical and non-technical personnel communicate effectively by providing a clear description of the problem.
- The problem statement does not describe the solution to the problem.

INPUT:

- The input to requirement engineering is the problem statement prepared by the customer.
- It may include an overview of the existing system and the broad expectations from the new system.
- The first phase of requirements engineering begins with requirements elicitation, i.e., gathering information about the requirements.

  Here, requirements are identified with the help of the customer and existing system processes.

Problem:

The traditional voting system faces several challenges, including long wait times, logistical inefficiencies, and the requirement for physical presence, which often limits voter participation. This results in lower voter turnout, especially among remote or disabled individuals. Additionally, the manual vote counting process is prone to human error and delays.

Background:

The concept of electronic voting (e-voting) emerged as a solution to address the inefficiencies and challenges associated with traditional paper-based voting systems. With advancements in technology, especially in digital security and data encryption, e-voting systems have gained attention as a more efficient, secure, and accessible way to conduct elections.

Relevance:

The relevance of an e-voting system lies in its ability to enhance the efficiency, accessibility, and security of electoral processes. As the world moves toward digitalization, traditional voting methods are increasingly seen as outdated and prone to errors or fraud.

E-voting offers a way to streamline elections, reduce operational costs, and improve voter turnout, especially for remote or disabled populations.

Objectives:

The objective of the e-voting system is to enhance voter accessibility by enabling secure remote voting, ensure the integrity and confidentiality of votes through advanced encryption and authentication, and increase election efficiency by automating processes for faster and more accurate results.

1. Enhance Voter Accessibility: Provide a secure and convenient platform for citizens to vote remotely, eliminating barriers such as geographical location, time constraints, and physical limitations.

2. Ensure Security and Privacy: Implement advanced encryption and authentication methods to protect voter identities, ensure vote confidentiality, and prevent unauthorized access or manipulation.

3. Increase Efficiency and Speed: Automate the voting and counting processes to reduce delays, eliminate human errors, and deliver faster, more accurate election results.

4. Promote Transparency: Use blockchain or other transparent technologies to create an immutable and verifiable record of votes, ensuring trust in the election outcome.

5. Reduce Operational Costs: Lower the costs associated with traditional election methods, such as printing ballots, setting up polling booths, and hiring staff for manual counting.

6. Boost Voter Participation: Encourage higher voter turnout by making the voting process more accessible, user-friendly, and less time-consuming for eligible voters.

Result:

EX NO:2	WRITE THE SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT
DATE	

AIM:

To do requirement analysis and develop Software Requirement Specification Sheet(SRS) for E-voting system.

ALGORITHM:

SRS shall address are the following:

a) **Functionality.** What is the software supposed to do?

b) **External interfaces.** How does the software interact with people, the system‘s hardware, other hardware, and other software?

c) **Performance.** What is the speed, availability, response time, recovery time of various software functions, etc.?

d) **Attributes.** What is the portability, correctness, maintainability, security, etc. considerations?

e) **Design constraints imposed on an implementation.** Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s) etc.?

1. Introduction

- **1.1 Purpose:**
  Define the purpose of the SRS document, outlining its intended audience and the goals of the e-voting system.
- **1.2 Scope:**
  Describe the scope of the e-voting system, including the major functionalities such as voter registration, vote casting, result calculation, and security features.
- **1.3 Definitions, Acronyms, and Abbreviations:**
  List all key terms, abbreviations, and acronyms used throughout the document.
- **1.4 References:**
  Include any external documents, standards, or regulations that the system must comply with (e.g., legal standards for electronic voting).

---

2. Overall Description

- **2.1 Product Perspective:**
  Describe the overall system architecture, including how the e-voting system fits into the existing electoral framework (if applicable).

- **2.2 Product Features:**
  Provide an overview of the core features of the system, such as voter authentication, vote casting, result calculation, and audit logs.
- **2.3 User Classes and Characteristics:**
  Identify the different user types (e.g., voters, election administrators, system auditors) and their needs and permissions.
- **2.4 Operating Environment:**
  Describe the technical environment (e.g., web platform, mobile app, server specifications, operating systems) in which the e-voting system will operate.
- **2.5 Design and Implementation Constraints:**
  Identify any constraints on the system design, such as compliance with regulations, security standards, or technological limitations.
- **2.6 Assumptions and Dependencies:**
  List assumptions made during system development and any external dependencies (e.g., third-party APIs for weather data, election scheduling).

---

3. System Features

- **3.1 Feature 1: Voter Registration and Authentication**
  - **Description:** Detailed description of how users will register and authenticate their identity.
  - **Functional Requirements:** Registration process, authentication methods (password, two-factor authentication), and eligibility verification.
- **3.2 Feature 2: Vote Casting**
  - **Description:** Outline how voters will cast their votes electronically.
  - **Functional Requirements:** Vote selection, encryption, submission process, and confirmation.
- **3.3 Feature 3: Vote Tallying**
  - **Description:** Automated process of counting votes after the election period ends.
  - **Functional Requirements:** Vote aggregation, result generation, and real-time tracking.
- **3.4 Feature 4: Security and Privacy**
  - **Description:** Measures to protect voter data and prevent unauthorized access.
  - **Functional Requirements:** Data encryption, authentication protocols, secure communication channels, and data anonymization.

---

4. External Interface Requirements

- **4.1 User Interfaces:**
  Describe the design of the user interfaces for voters and administrators, including layout, navigation, and accessibility considerations.

- **4.2 Hardware Interfaces:**
  Specify any hardware that the system will interact with (e.g., biometric scanners for authentication, voting booths, etc.).
- **4.3 Software Interfaces:**
  Identify external software systems that the e-voting system will interact with (e.g., voter databases, third-party APIs for weather data).
- **4.4 Communication Interfaces:**
  Define how the system will communicate over networks (e.g., HTTP, SSL/TLS encryption for secure communication).

---

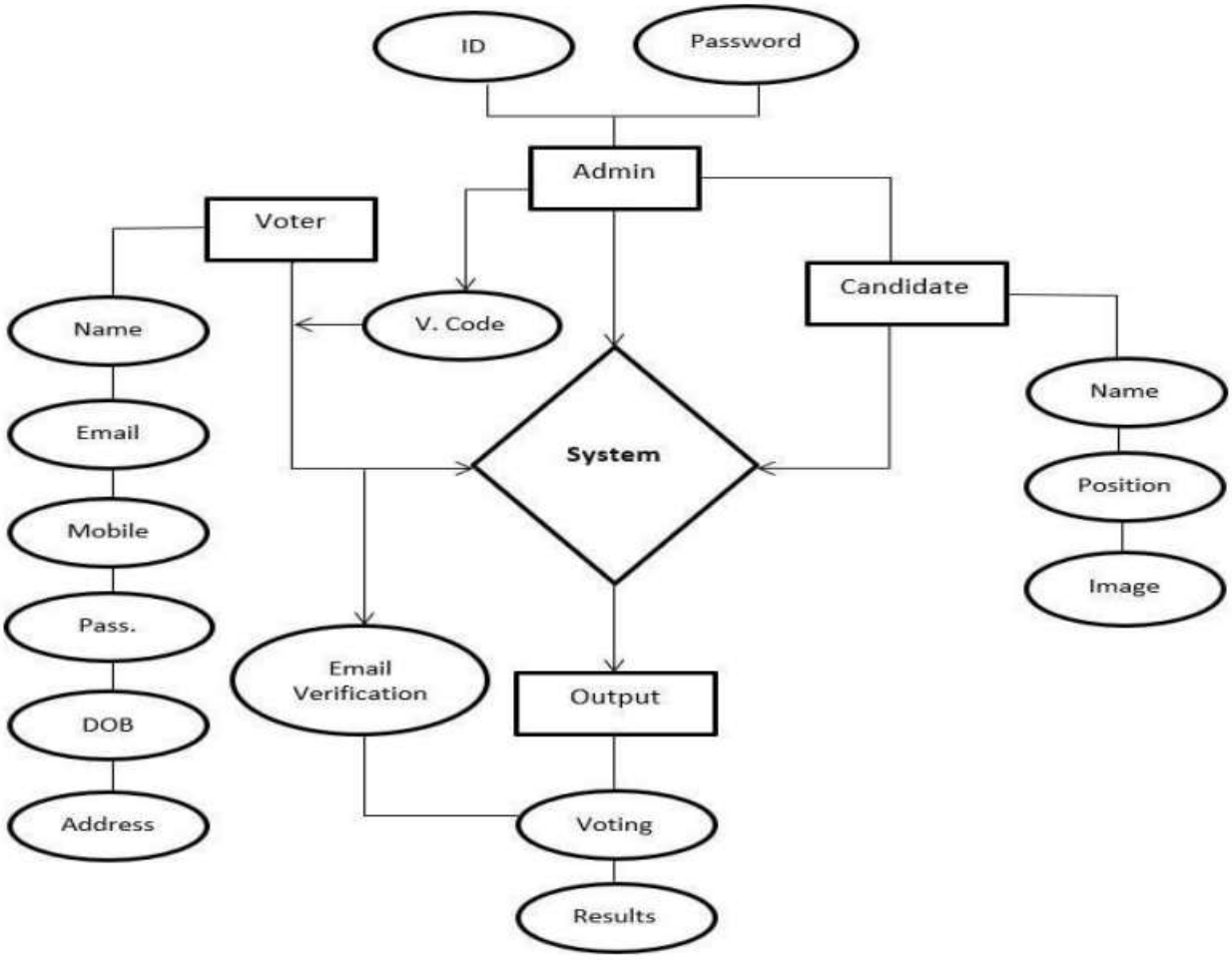
5. System Attributes

- **5.1 Performance Requirements:**
  Outline system performance criteria, such as response times, load handling (e.g., number of concurrent users during elections), and data processing speed.
- **5.2 Security Requirements:**
  Specify the security standards and features required for the system, including encryption, audit trails, and protection against cyberattacks (e.g., DDoS).
- **5.3 Reliability:**
  Define expected system uptime, fault tolerance, and backup/recovery requirements.
- **5.4 Availability:**
  Specify the availability requirements, particularly during election periods (e.g., 24/7 availability with minimal downtime).

Result:

SAMPLE OUTPUT:

ER DIAGRAM:



EX NO:3	DRAW THE ENTITY RELATIONSHIP DIAGRAM
DATE	

AIM:

To Draw the Entity Relationship Diagram for E-Voting system.

ALGORITHM:

Step 1: Mapping of Regular Entity Types

Step 2: Mapping of Weak Entity Types

Step 3: Mapping of Binary 1:1 Relation Types

Step 4: Mapping of Binary 1:N Relationship Types.

Step 5: Mapping of Binary M:N Relationship Types.

Step 6: Mapping of Multivalued attributes.

INPUT:

Entities

Entity Relationship Matrix

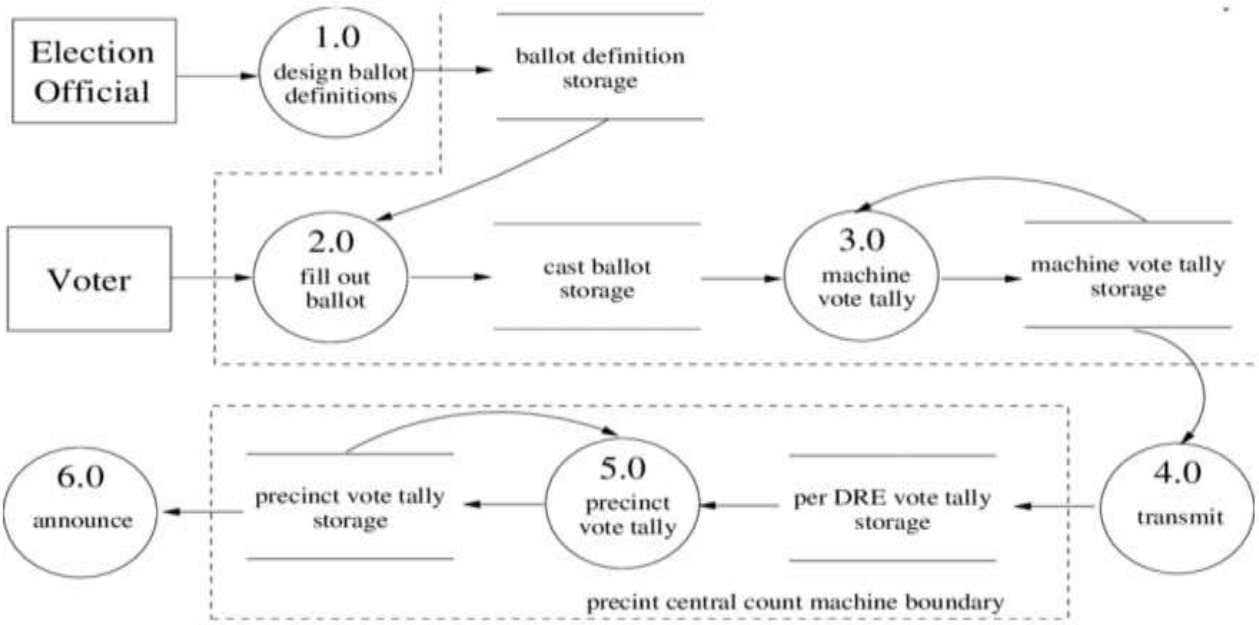
Primary Keys

Attributes

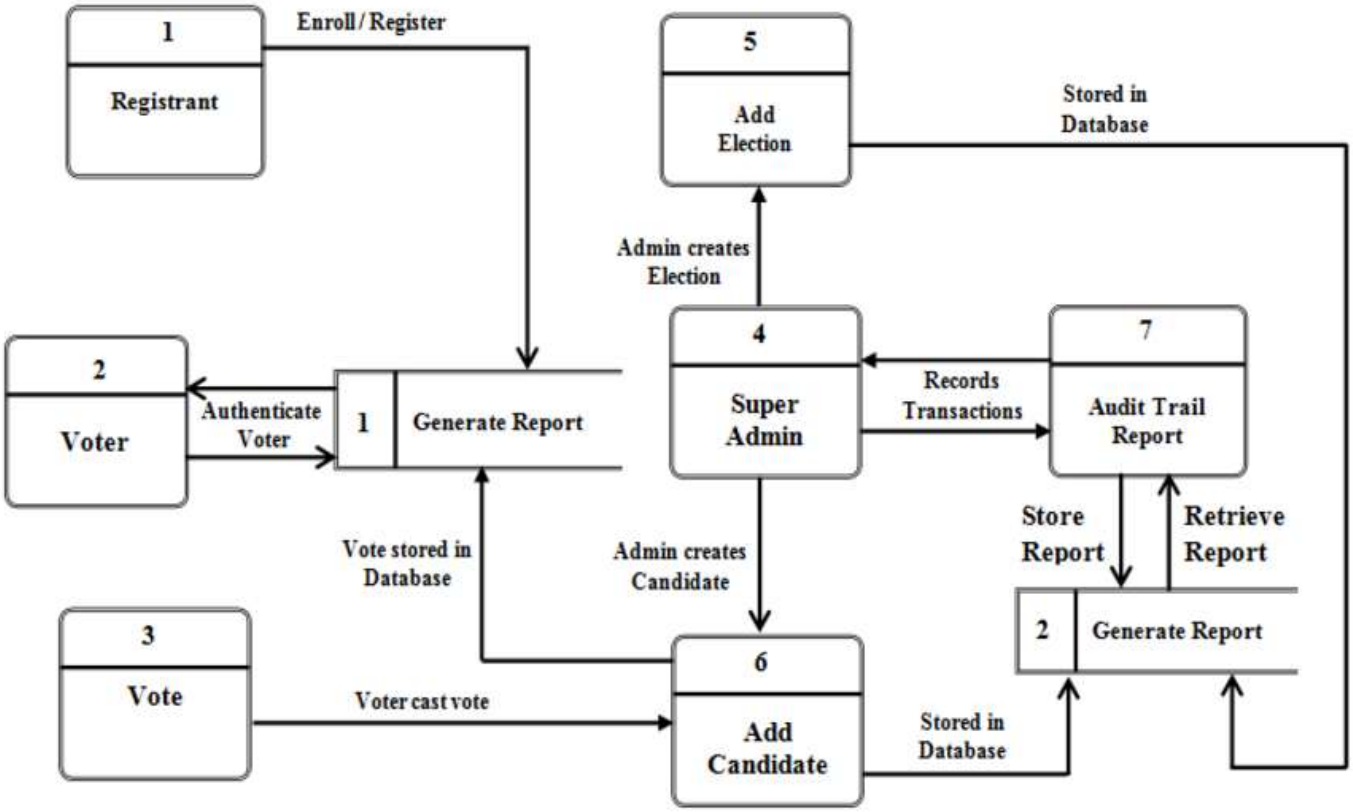
Mapping of Attributes with Entities

Result:

ZERO LEVEL:



FIRST LEVEL:



EX NO:4	DRAW THE DATA FLOW DIAGRAMS AT LEVEL 0 AND LEVEL 1
DATE	

AIM:

To Draw the Data Flow Diagram for E-Voting system and List the Modules in the Application.

ALGORITHM:

1. Open the Visual Paradigm to draw DFD (Ex.Lucidchart)
2. Select a data flow diagram template
3. Name the data flow diagram
4. Add an external entity that starts the process
5. Add a Process to the DFD
6. Add a data store to the diagram
7. Continue to add items to the DFD
8. Add data flow to the DFD
9. Name the data flow
10. Customize the DFD with colours and fonts
11. Add a title and share your data flow diagram

INPUT:

Processes

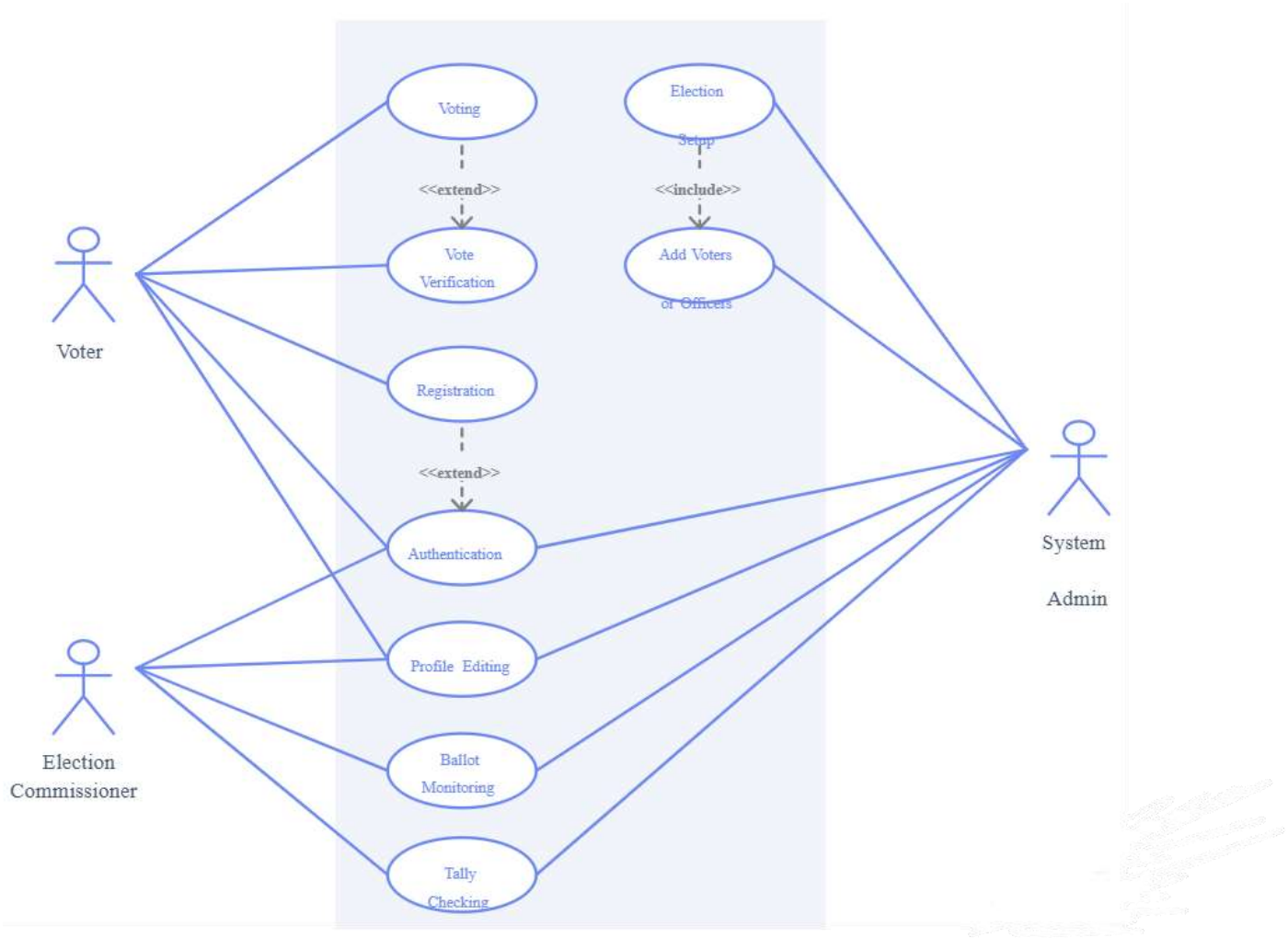
Datastores

External Entities

Result:

SAMPLE OUTPUT:

USE CASE DIAGRAM:



EX NO:5	DRAW USE CASE DIAGRAM
DATE	

AIM:

To Draw the Use Case Diagram for E-Voting system**.**

ALGORITHM:

Step 1: Identify Actors

Step 2: Identify Use Cases

Step 3: Connect Actors and Use Cases

Step 4: Add System Boundary

Step 5: Define Relationships

Step 6: Review and Refine

Step 7: Validate

INPUTS:

Actors

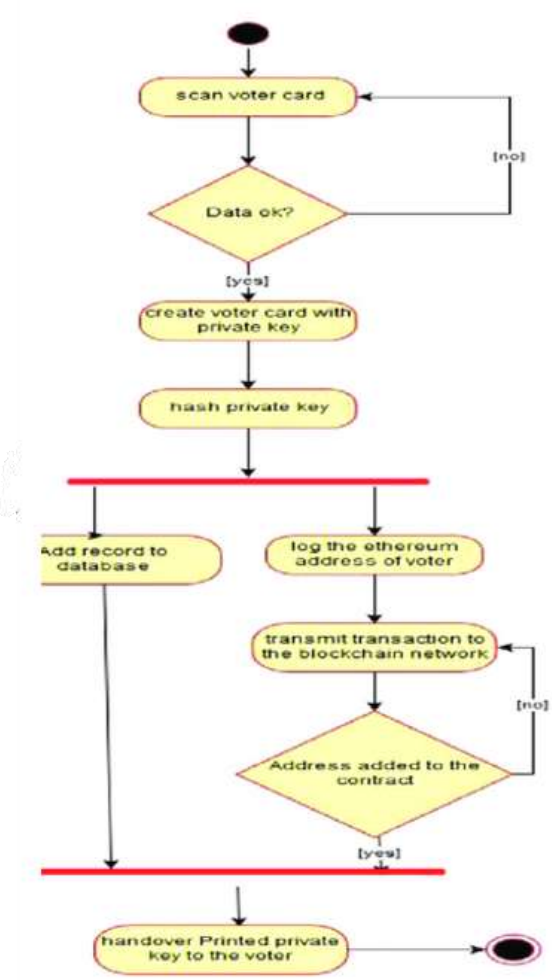
Use Cases

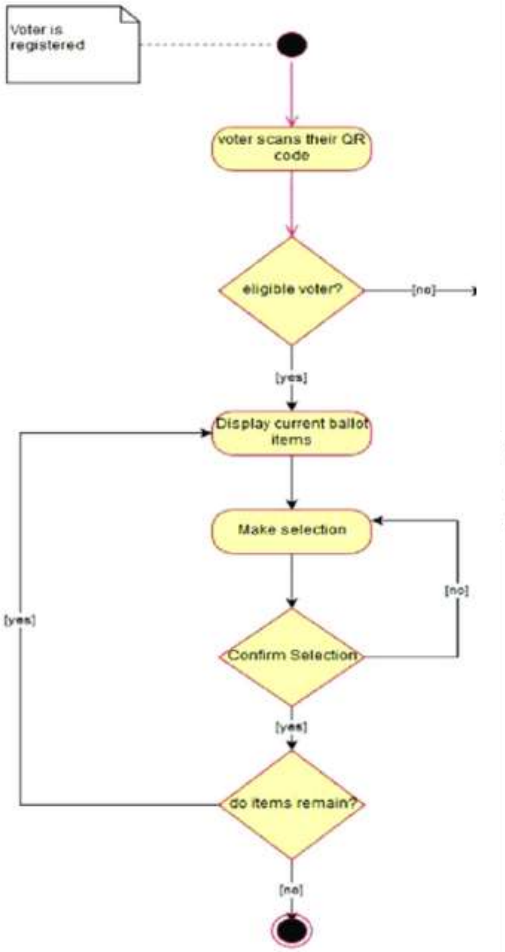
Relations

Result:

SAMPLE OUTPUT:

ACTIVITY DIAGRAM:





EX NO:6	DRAW ACTIVITY DIAGRAM OF ALL USE CASES.
DATE	

AIM:

To Draw the activity Diagram for E-Voting system.

ALGORITHM:

Step 1: Identify the Initial State and Final States

Step 2: Identify the Intermediate Activities Needed

Step 3: Identify the Conditions or Constraints

Step 4: Draw the Diagram with Appropriate Notations

INPUTS:

Activities

Decision Points

Guards

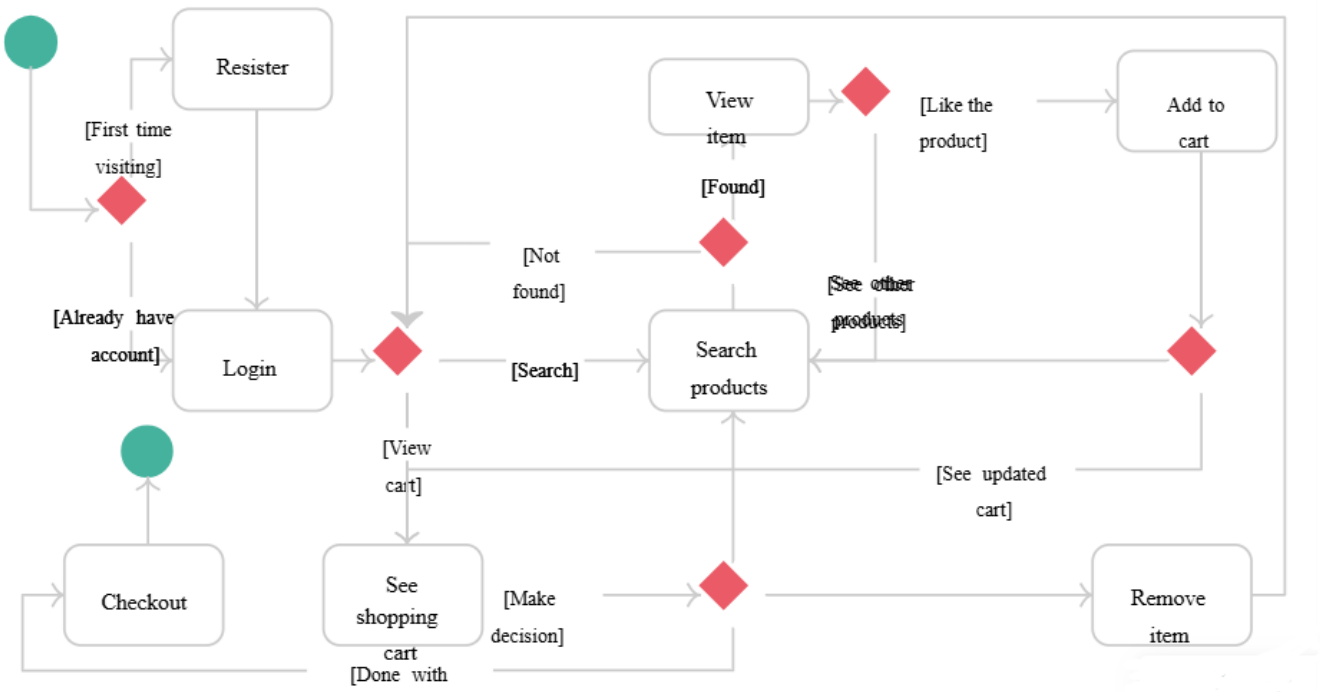
Parallel Activities

Conditions

Result:

SAMPLE OUTPUT:

STATE CHART DIAGRAM:



EX NO:7	DRAW STATE CHART DIAGRAM OF ALL USE CASES.
DATE	

AIM:

To Draw the State Chart Diagram for E-Voting system.

ALGORITHM:

STEP-1: Identify the important objects to be analysed.

STEP-2: Identify the states.

STEP-3: Identify the events.

INPUTS:

Objects

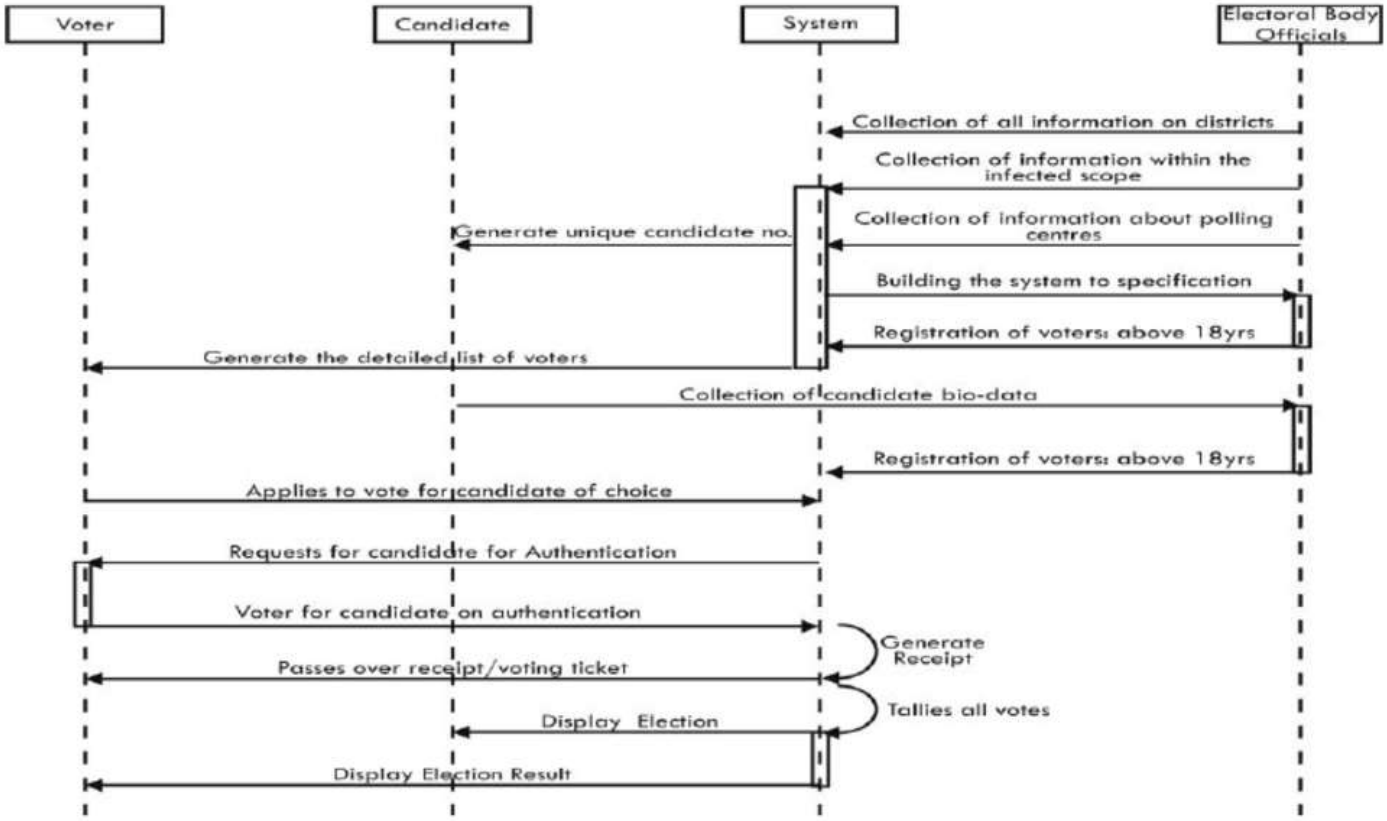
States

Events

Result:

SAMPLE OUTPUT:

SEQUENCE DIAGRAM:



EX NO:8	DRAW SEQUENCE DIAGRAM OF ALL USE CASES.
DATE	

AIM:

To Draw the Sequence Diagram for E-Voting system.

ALGORITHM:

1. Identify the Scenario
2. List the Participants
3. Define Lifelines
4. Arrange Lifelines
5. Add Activation Bars
6. Draw Messages
7. Include Return Messages
8. Indicate Timing and Order
9. Include Conditions and Loops
10. Consider Parallel Execution
11. Review and Refine
12. Add Annotations and Comments
13. Document Assumptions and Constraints
14. Use a Tool to create a neat sequence diagram

INPUTS:

Objects taking part in the interaction.

Message flows among the objects.

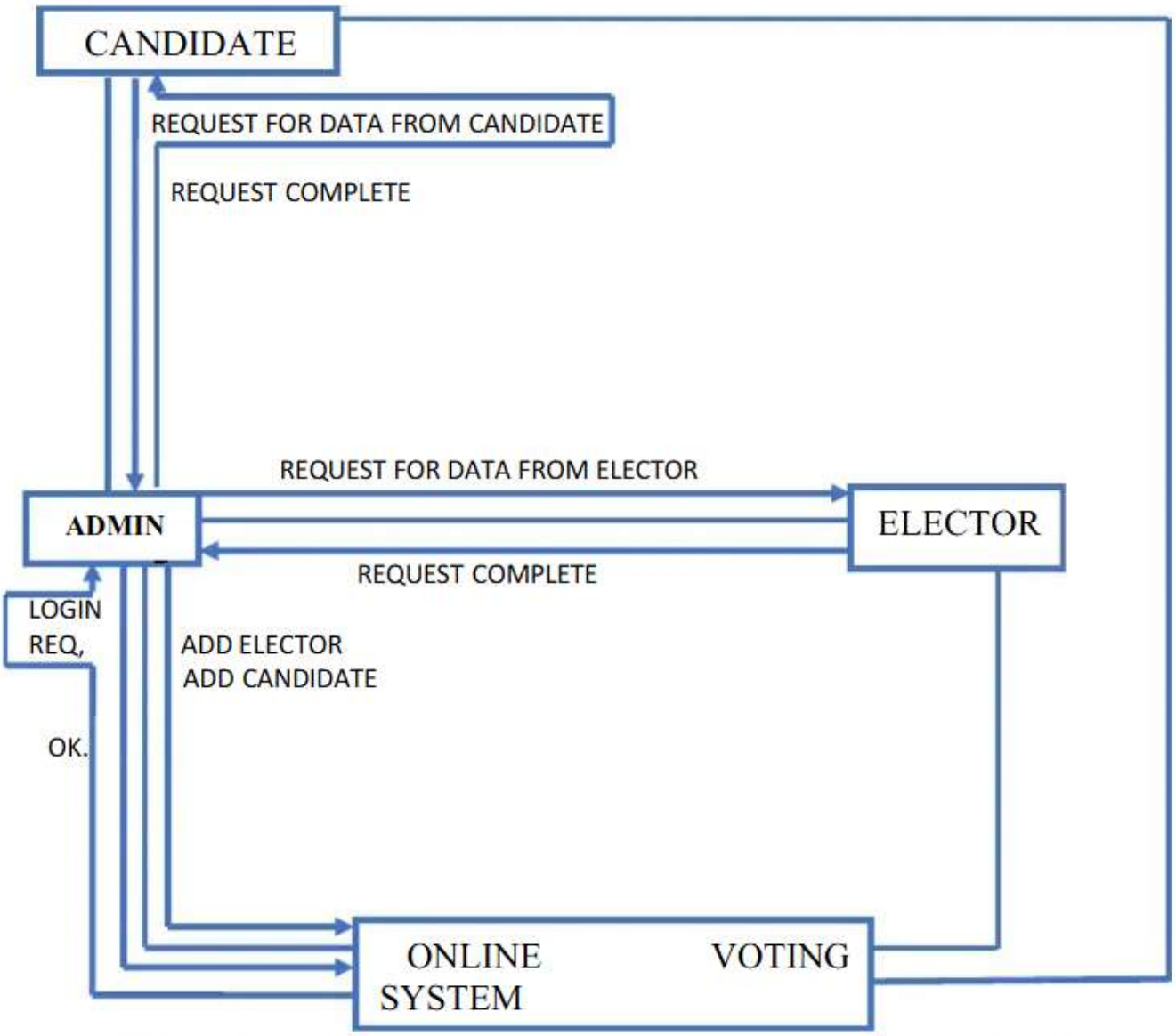
The sequence in which the messages are flowing.

Object organization.

Result:

SAMPLE OUTPUT:

COLLABORATION DIGRAM:



ADDING ELECTOR AND CANDIDATE INFORMATION (A)

EX NO:9	DRAW COLLABORATION DIAGRAM OF ALL USE CASES
DATE	

AIM:

To Draw the Collaboration Diagram for E-Voting system.

ALGORITHM:

Step 1: Identify Objects/Participants

Step 2: Define Interactions

Step 3: Add Messages

Step 4: Consider Relationships

Step 5: Document the collaboration diagram along with any relevant explanations or annotations.

INPUTS:

Objects taking part in the interaction.

Message flows among the objects.

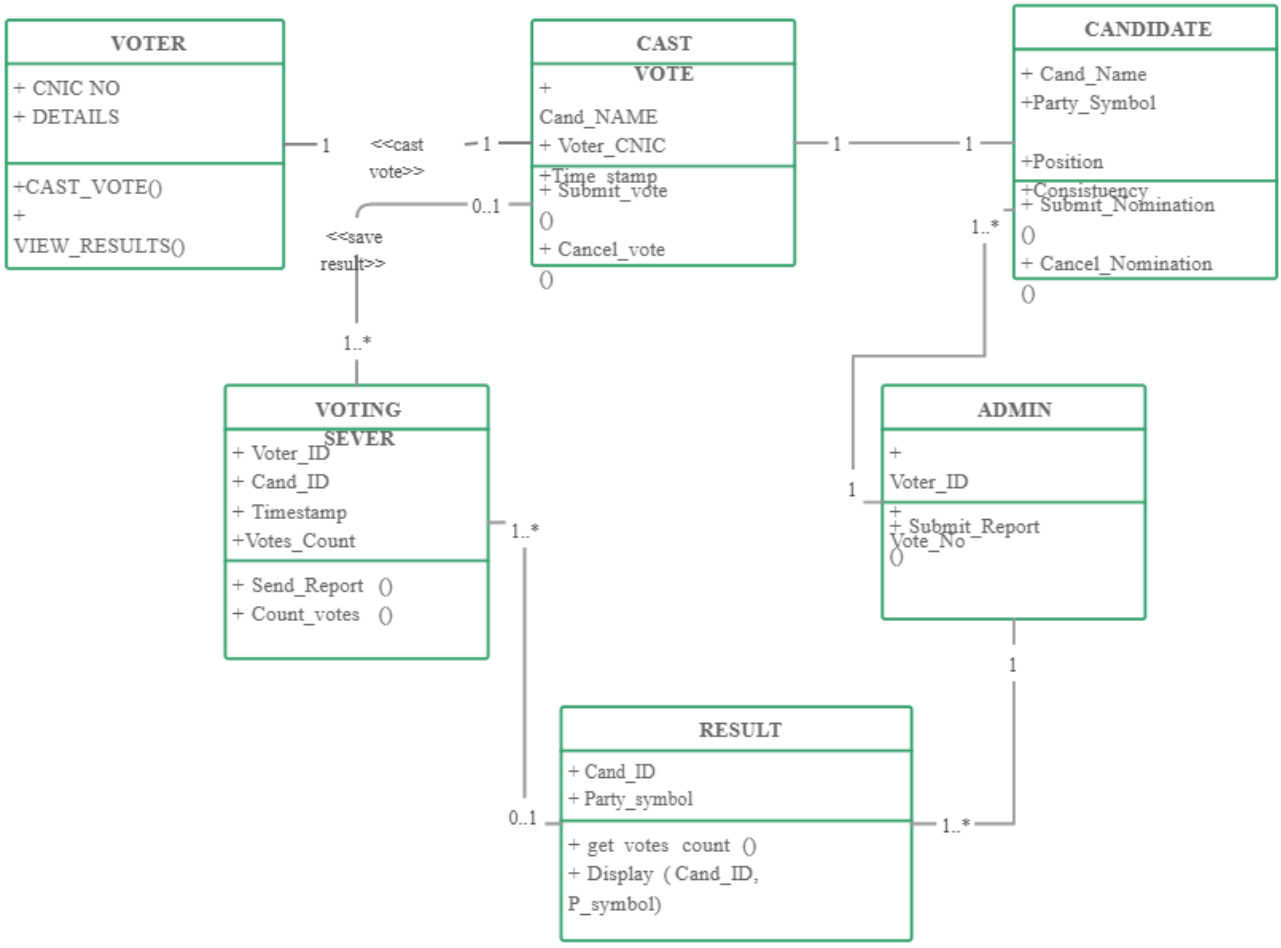
The sequence in which the messages are flowing.

Object organization.

Result:

SAMPLE OUTPUT:

CLASS DIAGRAM:



EX NO:10	ASSIGN OBJECTS IN SEQUENCE DIAGRAM TO CLASSES AND MAKE CLASS DIAGRAM.
DATE	

AIM:

To Draw the Class Diagram for E-Voting system.

ALGORITHM:

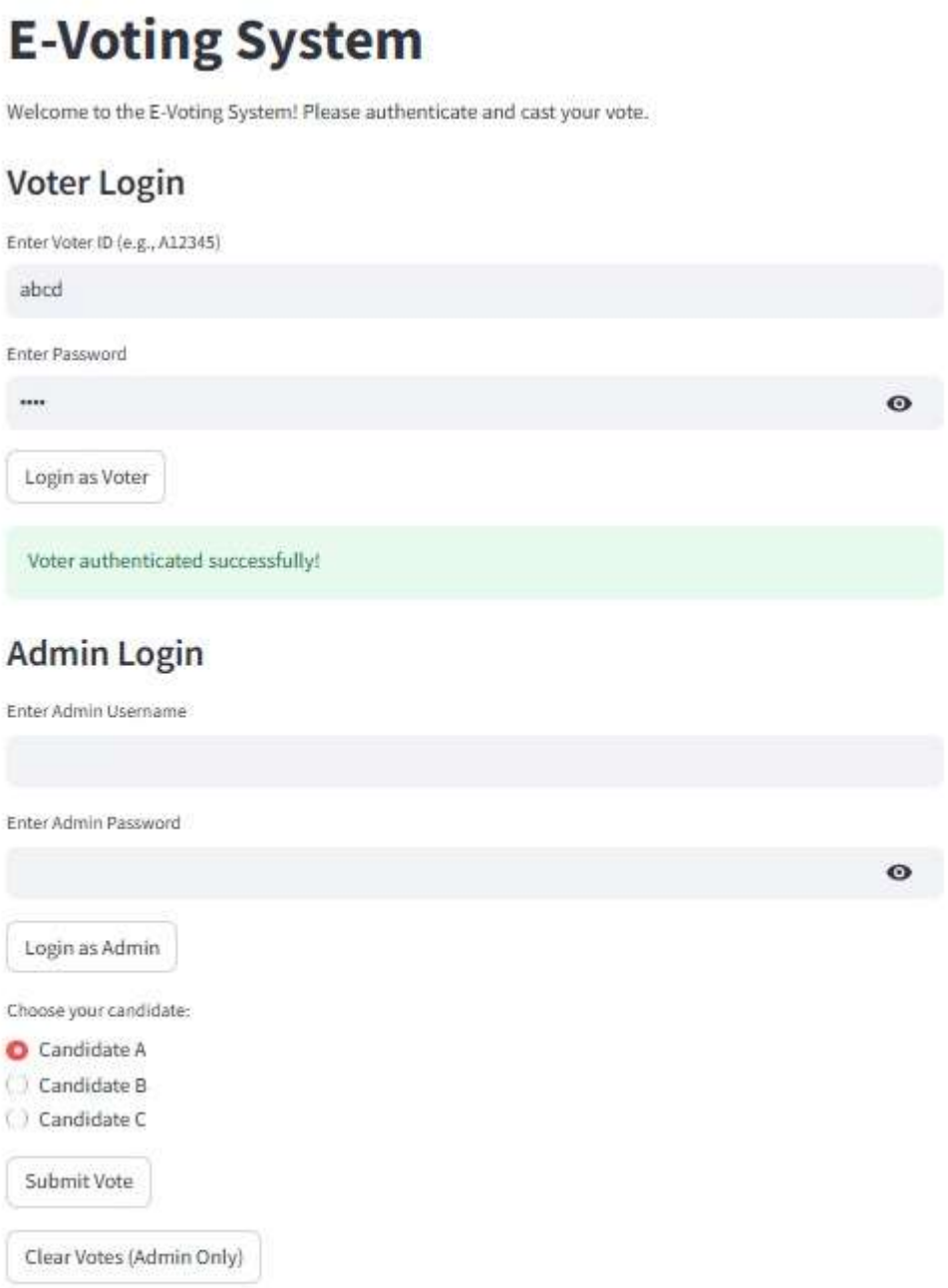
1. Identify Classes
2. List Attributes and Methods
3. Identify Relationships
4. Create Class Boxes
5. Add Attributes and Methods
6. Draw Relationships
7. Label Relationships
8. Review and Refine
9. Use Tools for Digital Drawing

INPUTS:

1. Class Name
2. Attributes
3. Methods
4. Visibility Notation

RESULT:

OUTPUT:


E-Voting System
Welcome to the E-Voting System! Please authenticate and cast your vote.
Voter Login
Enter Voter ID (e.g., A12345)
abcd
Enter Password
Login as Voter
Voter authenticated successfully!
Admin Login
Enter Admin Username
Enter Admin Password
Login as Admin
Choose your candidate:
Candidate A
Candidate B
Candidate C
Submit Vote
Clear Votes (Admin Only)

EX NO:11	MINI PROJECT- E-VOTING SYSTEM
DATE	

AIM:

The primary aim of this mini-project is to develop a secure and user-friendly E-Voting system. By utilizing MySQL for robust data storage and Streamlit for a seamless user interface, we aim to enhance the voting process, ensuring transparency, efficiency, and voter confidentiality.

ALGORITHM:

1. User registers with valid credentials.
2. User logs in using their credentials.
3. System displays a list of candidates.
4. User selects their preferred candidate.
5. User's vote is encrypted and stored securely.
6. System verifies the integrity of the voting process.
7. Votes are decrypted and counted to determine the winner.
8. Results are published transparently.

PROGRAM:

```
import streamlit as st

# Initialize session state to store vote data and authentication status

if 'votes' not in st.session_state:

    st.session_state.votes = {'Candidate A': 0, 'Candidate B': 0, 'Candidate C': 0}

if 'voted' not in st.session_state:

    st.session_state.voted = False  # To track if the user has voted

if 'voter_authenticated' not in st.session_state:

    st.session_state.voter_authenticated = False  # To track if the user is authenticated

if 'admin_authenticated' not in st.session_state:

    st.session_state.admin_authenticated = False  # For admin authentication
```


E-Voting System
Welcome to the E-Voting System! Please authenticate and cast your vote.
Admin Login
Enter Admin Username
Enter Admin Password
Login as Admin
Choose your candidate:
Candidate A
Candidate B
Candidate C
Submit Vote
Clear Votes (Admin Only)

```
# Admin credentials (for simplicity, using hardcoded values)
ADMIN_USERNAME = "admin"
ADMIN_PASSWORD = "admin123"
# Voting System Title
st.title("E-Voting System")
st.write("Welcome to the E-Voting System! Please authenticate and cast your vote.")
# Authentication - Voter login
if not st.session_state.voter_authenticated:
    st.subheader("Voter Login")
    voter_id = st.text_input("Enter Voter ID (e.g., A12345)", max_chars=10)
    voter_password = st.text_input("Enter Password", type="password")
    if st.button("Login as Voter"):
        if voter_id and voter_password:
            # In a real system, authenticate using a database or external service
            st.session_state.voter_authenticated = True
            st.success("Voter authenticated successfully!")
        else:
            st.error("Please enter valid credentials.")
# Admin authentication section
if not st.session_state.admin_authenticated:
    st.subheader("Admin Login")
    admin_username = st.text_input("Enter Admin Username", max_chars=10)
    admin_password = st.text_input("Enter Admin Password", type="password")
    if st.button("Login as Admin"):
        if admin_username == ADMIN_USERNAME and admin_password == ADMIN_PASSWORD:
            st.session_state.admin_authenticated = True
            st.success("Admin authenticated successfully!")
        else:
```

E-Voting System

Welcome to the E-Voting System! Please authenticate and cast your vote.

Admin Login

Enter Admin Username

Enter Admin Password

Login as Admin

Choose your candidate:

- ○ Candidate A
- ◉ Candidate B
- ○ Candidate C

Submit Vote

Your vote for Candidate B has been recorded!

Thank you for voting!

Voting Results:

Total Votes: 1

Candidate A: 0 votes (0.00%)

Candidate B: 1 votes (100.00%)

Candidate C: 0 votes (0.00%)

You have already voted. Thank you for participating.

Clear Votes (Admin Only)

```
st.error("Invalid admin credentials.")
# Voting Process
if st.session_state.voter_authenticated and not st.session_state.voted:
    # Voting form
    candidate = st.radio("Choose your candidate:", ['Candidate A', 'Candidate B', 'Candidate C'])
    if st.button("Submit Vote"):
        # Prevent multiple votes by the same user
        st.session_state.votes[candidate] += 1
        st.session_state.voted = True
        st.success(f"Your vote for {candidate} has been recorded!")
        st.write("Thank you for voting!")
# Display results only if voting is closed or admin is authenticated
if st.session_state.admin_authenticated or st.session_state.voted:
    st.subheader("Voting Results:")
    total_votes = sum(st.session_state.votes.values())
    if total_votes > 0:
        st.write(f"Total Votes: {total_votes}")
        for candidate, vote_count in st.session_state.votes.items():
            percentage = (vote_count / total_votes) * 100
            st.write(f"{candidate}: {vote_count} votes ({percentage:.2f}%)")
    else:
        st.write("No votes yet.")
# Admin Feature - Close voting
if st.session_state.admin_authenticated:
    st.subheader("Admin Options")
    if st.button("End Voting"):
        st.session_state.votes = {'Candidate A': 0, 'Candidate B': 0, 'Candidate C': 0}
        st.session_state.voted = False
```

E-Voting System

Welcome to the E-Voting System! Please authenticate and cast your vote.

Admin Login

Enter Admin Username

Enter Admin Password

Login as Admin

Voting Results:

Total Votes: 1

Candidate A: 0 votes (0.00%)

Candidate B: 1 votes (100.00%)

Candidate C: 0 votes (0.00%)

You have already voted. Thank you for participating.

Clear Votes (Admin Only)

You must be an admin to clear votes.

```
  st.session_state.voter_authenticated = False  # Reset voter authentication
        st.success("Voting has been ended and results have been reset.")
# Display a message if the user has already voted
if st.session_state.voted:
    st.warning("You have already voted. Thank you for participating.")
# Clear Votes (for testing)
if st.button("Clear Votes (Admin Only)"):
    if st.session_state.admin_authenticated:
        st.session_state.votes = {'Candidate A': 0, 'Candidate B': 0, 'Candidate C': 0}
        st.session_state.voted = False
        st.session_state.voter_authenticated = False
        st.success("Votes have been cleared.")
    else:
        st.error("You must be an admin to clear votes.")
```

`

Conclusion:

The E-Voting system offers a secure, efficient, and transparent solution to traditional voting methods. By leveraging technology, it enhances voter participation, reduces the risk of fraud, and expedites the result declaration process. This project demonstrates the potential of technology to revolutionize the electoral process and strengthen democratic principles.