

# 

# XCS508 - SOFTWARE ENGINEERING LABORATORY

# MINI PROJECT REPORT

# ON

ONLINE ELECTORAL SYSTEM

SUBMITTED BY

S AKSHAYA SARAVANAN -- 121011012719

M SABAREESWARAN -- 121012012759

C SRIHARIHARAN -- 121012012769

**Aim:**

The Online Electoral System aims to revolutionize the democratic process by offering citizens a convenient and secure platform for exercising their voting rights. This system strives to enhance accessibility, allowing voters to participate from any location with internet access, thereby fostering inclusivity and potentially increasing overall voter turnout. Emphasizing robust security measures, the aim is to safeguard the integrity of elections by ensuring the confidentiality of voter information and preventing unauthorized access. Additionally, the system aims for operational efficiency, streamlining the voting process for both citizens and election officials. Through a user-friendly interface and real-time result reporting, the aim is to promote transparency, trust.

**Software Requirements Specification (SRS) for Online Electoral System**

**1.Introduction**

Welcome to the Software Requirements Specification (SRS) document for the Online Electoral System. This document serves as a comprehensive guide outlining the requirements, functionalities, and specifications of the proposed system. It aims to provide a clear understanding of the project's objectives, ensuring a smooth development process and successful implementation.

* 1. **Purpose:**

The primary purpose of this SRS document is to define the requirements of the Online Electoral System, detailing the features and functionalities that will enable efficient and secure online voting processes. It serves as a foundation for developers, testers, and other stakeholders, ensuring a shared understanding of the system's goals and functionalities.

**1.2.Document Conventions:**

Standard conventions such as use of UML diagrams, requirement numbering, and consistent terminology will be employed. Acronyms and abbreviations will be defined for clarity, and each requirement will be categorized according to its type (functional, non-functional, etc.) to facilitate easy navigation and comprehension.

**1.3 Intended Audience:**

A diverse audience including developers, testers, project managers, and other stakeholders involved in the development and deployment of the Online Electoral System. It is assumed that the readers possess a basic understanding of software development and electoral processes.

**1.4 Reading Suggestions:**

Developers should focus on the detailed functional and non-functional requirements sections. Testers should refer to the testing requirements and scenarios outlined in the respective section. Project managers may find the project scope and timeline sections particularly relevant.

**1.5 Project Scope:**

The Online Electoral System aims to provide a secure, user-friendly platform for conducting elections in an online environment. It includes features such as voter registration, candidate nomination, secure authentication, and real-time result generation. The system will adhere to the electoral laws and regulations, ensuring transparency and integrity in the entire voting process.

**1.6 References:**

**-** Election Commission Guidelines

- Relevant Electoral Regulations

- Security Standards for Online Voting Systems

- Previous Successful Implementations of Online Voting Systems

**2. Overall Description**

**2.1 Product Perspective:**

The Online Electoral System operates as an independent and self-contained platform, interacting with existing electoral processes. It interfaces with voter databases, ensuring seamless integration while maintaining data integrity. The system is designed to enhance and digitize the traditional electoral workflow, providing a secure and efficient online voting experience.

**2.2 Product Features:**

**Voter Registration**: Allows citizens to register securely through the system.

**Candidate Nomination:** Enables political candidates to submit their candidacy online.

**Secure Authentication:** Implements robust authentication mechanisms to ensure voter identity verification.

**Ballot Casting**: Facilitates the electronic casting of votes with a user-friendly interface.

**Real-time Result Generation:** Provides instant and accurate results after the completion of voting.

**2.3 User Classes and Characteristics:**

* Voters: General citizens eligible to cast votes.
* Candidates: Individuals running for political office.
* Administrators: Election officials responsible for system management.
* Developers: Those involved in system maintenance and updates.

**2.4 Operating Environment:**

The Online Electoral System will operate in a secure server environment, accessible via standard web browsers. Compatibility will be maintained with common operating systems, ensuring widespread accessibility for voters and administrators alike.

**2.5 Design and Implementation Constraints:**

The system must comply with electoral regulations and data protection laws. Accessibility considerations for voters with diverse needs. Integration challenges with existing electoral databases and systems.

**2.6 Assumptions and Dependencies:**

**Assumption:** Voters have access to a reliable internet connection. Security measures, such as encryption, will be effective in safeguarding voter information.

**Dependency:** Compliance with electoral laws and regulations is essential for system deployment. Integration with existing electoral databases requires collaboration with relevant authorities.

**3. System Features**

**3.1 Functional Requirements:**

* User Registration:

- Users should be able to create accounts with accurate personal information.

- The system must validate and store user details securely.

* Candidate Nomination:

- Candidates must be able to submit their nomination forms electronically.

- The system should verify and record candidate information.

* Voter Authentication:

- Implement a secure authentication process for verifying voter identity.

- Ensure the confidentiality and integrity of user credentials.

* Ballot Casting:

- Provide an intuitive and user-friendly interface for casting votes.

- Ensure each voter can cast only one vote per election.

* Real-time Result Generation:

- Generate and display accurate election results in real-time.

- Results should be accessible to authorized individuals only.

**4. External Interface Requirements**

**4.1 User Interfaces:**

- Voter Interface:

- Intuitive dashboard for voter registration and ballot casting.

- Clear instructions and visual cues for ease of use.

- Candidate Interface:

- User-friendly nomination submission forms.

- Accessible platform for tracking campaign progress.

- Administrator Interface:

- Secure login for election officials.

- Comprehensive dashboard for system management.

**4.2 Hardware Interfaces:**

- The system should be compatible with standard web browsers.

- Ensure compatibility with diverse devices, including desktops, tablets, and smartphones.

**4.3 Software Interfaces:**

- Integration with existing voter databases for authentication.

- Compatibility with common operating systems (Windows, macOS, Linux).

- Security protocols for data encryption and protection.

**4.4 Communications Interfaces:**

- Secure communication channels for user data transmission.

- Integration with email or SMS services for user notifications.

- Compatibility with standard internet protocols (HTTP/HTTPS).

**5. Non-Functional Requirements**

**5.1 Performance Requirements:**

- Response Time:

- The system should respond to user interactions within 3 seconds.

- Real-time result generation should occur promptly after the completion of voting.

- Scalability:

- The system must handle an increased load during peak voting times.

- Scalability testing should ensure optimal performance under varying user loads.

**5.2 Safety Requirements:**

- Data Integrity:

- The system must maintain the integrity of voter and election data.

- Regular data backups should be performed to prevent data loss.

- Disaster Recovery:

- Implement a disaster recovery plan to restore the system in case of failures.

- Ensure minimal downtime and data loss during system recovery processes.

**5.3 Security Requirements:**

- Authentication:

- Use multi-factor authentication to enhance user identity verification.

- Employ encryption protocols to secure user data during transmission.

- Access Control:

- Define and enforce access controls for different user roles (voters, candidates, administrators).

- Implement role-based access to ensure authorized access to system functionalities.

- Data Privacy:

- Adhere to data protection regulations to safeguard user privacy.

- Regularly audit and monitor access to sensitive voter information.

**5.4 Software Quality Attributes:**

- Reliability:

- The system should be available 99.9% of the time during election periods.

- Minimize system failures and ensure quick recovery in case of errors.

- Usability:

- Provide an intuitive and accessible user interface for voters and candidates.

- Conduct user testing to ensure ease of use across diverse demographics.

- Maintainability:

- Design modular and well-documented code for ease of maintenance.

- Implement version control to track changes and updates.

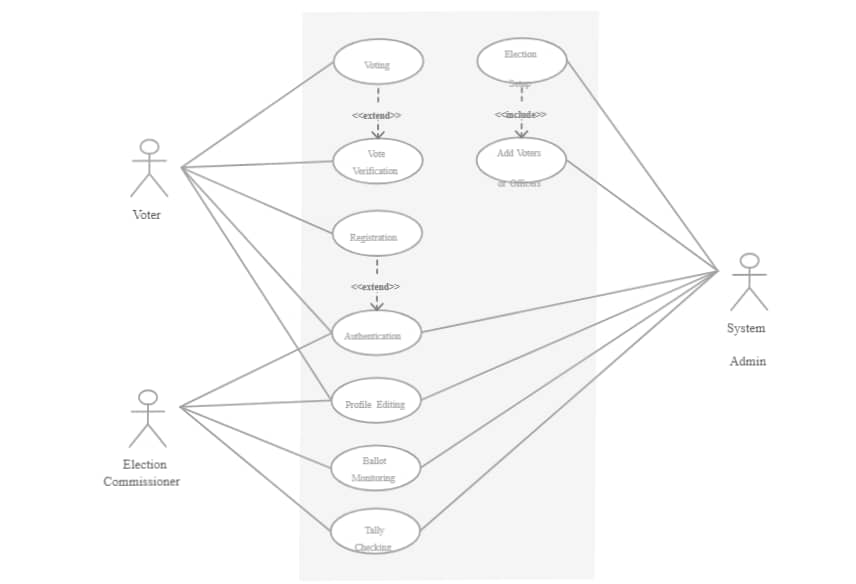
**Conclusion**

In conclusion, this Software Requirements Specification (SRS) document outlines the comprehensive framework for the development and implementation of the Online Electoral System. By addressing functional and nonfunctional requirements, external interfaces, and system features, this document serves as a vital guide for developers, testers, and stakeholders involved in the project. The system aims to revolutionize the electoral process by embracing modern technology to enhance accessibility, security, efficiency, and transparency. As we embark on the development journey, adherence to these requirements will ensure the creation of a robust, user-friendly, and trustworthy Online Electoral System that contributes to the democratic fabric of our society. Continuous collaboration, feedback, and adherence to industry standards will be pivotal in realizing the envisioned objectives of this transformative

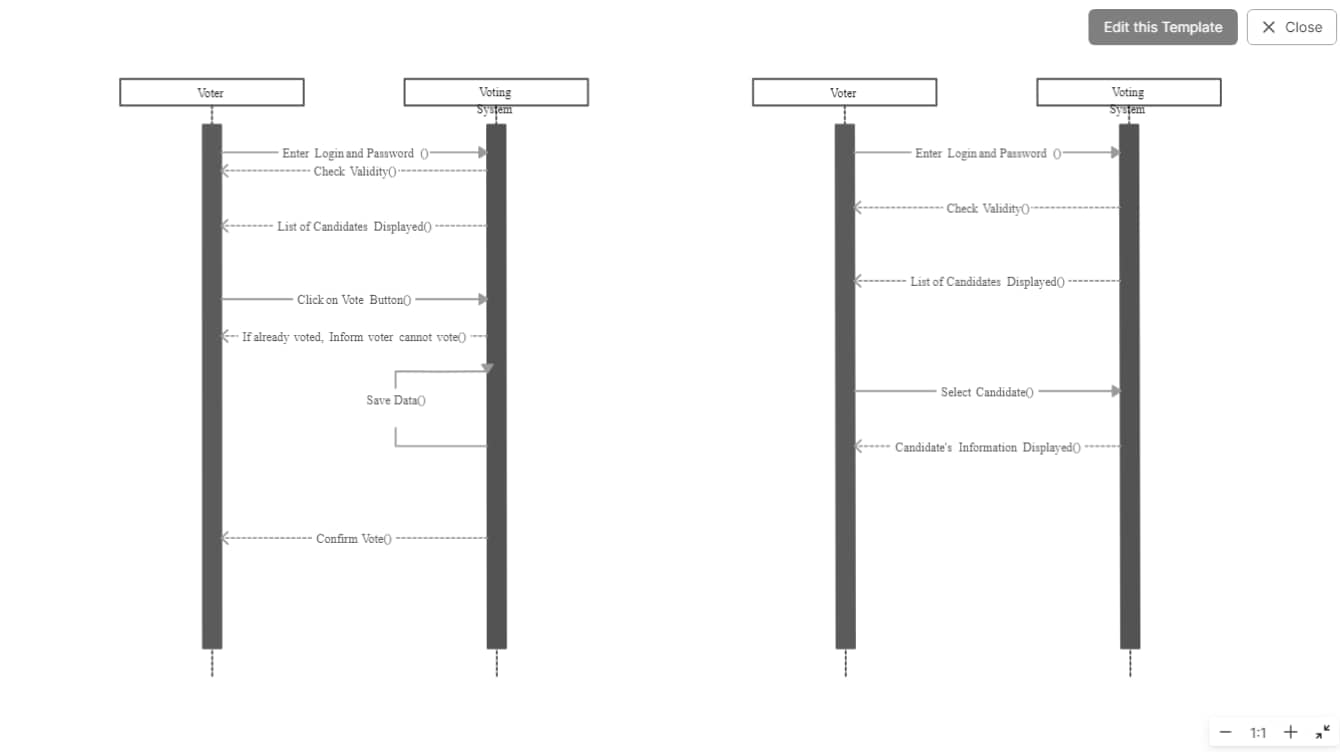
electoral platform.

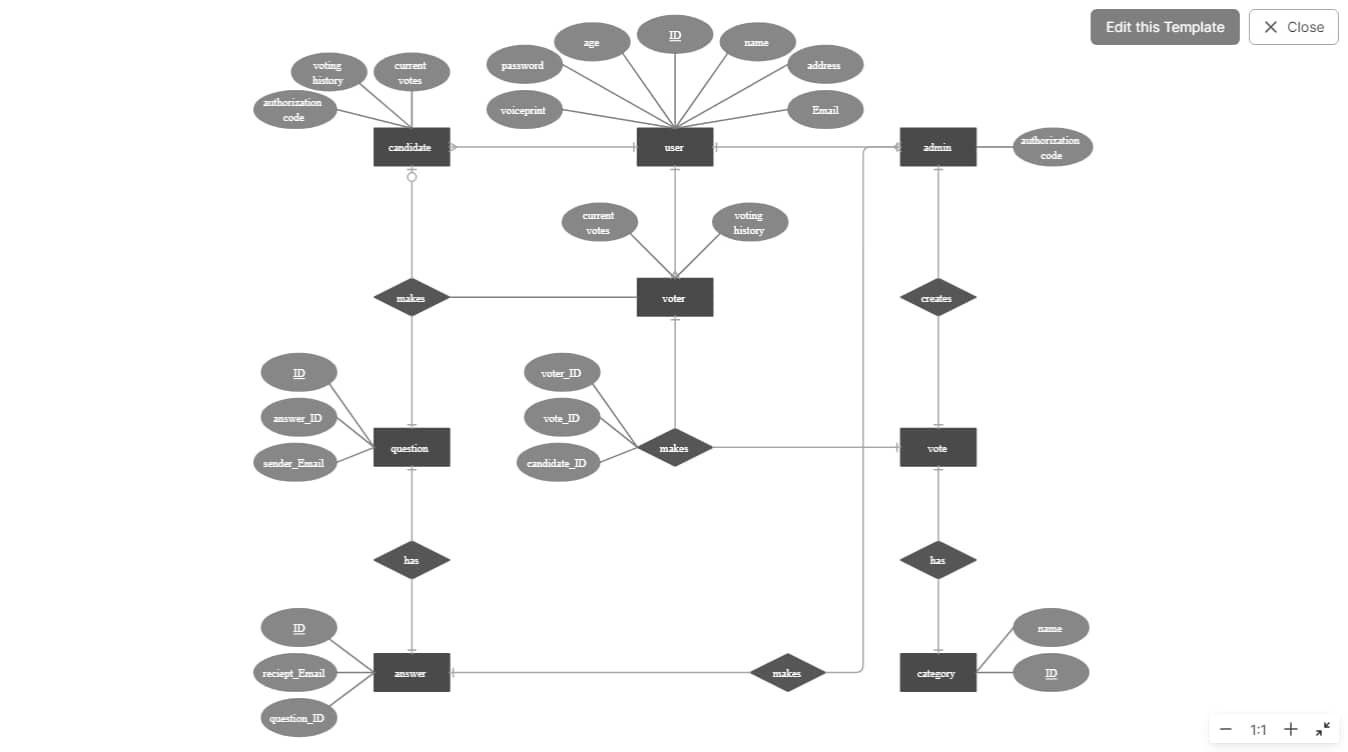
**Design**

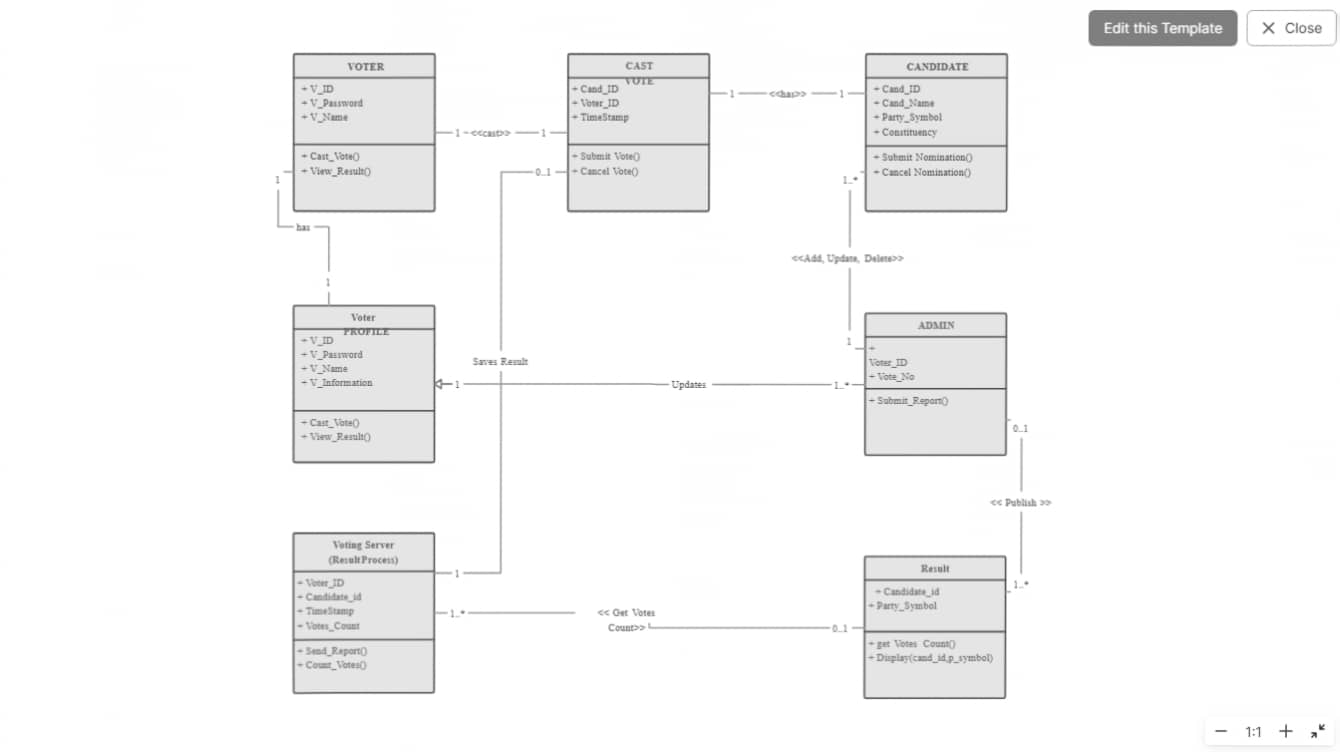
1. **Usecase Diagram**

****

1. **Sequence Diagram**



1. **ER Diagram**
2. **Class Diagram**

****

**Project Management Plan - Online Electoral System**

**1.Executive Summary**

* 1. **Project Overview:**

The Online Electoral System project endeavors to create a secure and accessible platform for conducting elections in an online environment. It aims to streamline the electoral process, ensuring transparency, integrity, and inclusivity.

* 1. **Project Manager:**

The project manager, Mr.Srihariharan , will be responsible for orchestrating project activities, resource management, and ensuring the project's alignment with its objectives.

1. **Project Objectives**

- Develop a robust online voting system with secure authentication and real-time result generation.

- Ensure compliance with electoral laws and regulations, prioritizing transparency and fairness.

- Enhance user experience for voters, candidates, and administrators through an intuitive interface.

1. **Project Scope**

The project encompasses the end-to-end development lifecycle of the online electoral system, covering planning, design, development, testing, deployment, and ongoing maintenance.

1. **Project Team**

**4.1 Roles and Responsibilities**

- Project Manager: Mr. Srihariharan

- Developer: Miss. S Akshaya Saravanan

- Electoral Process Consultant : Mr.Sabareeswaran

1. **Project Timeline**

**Milestones:**

1. Project Initiation and Planning:November 3rd, 2023 - November 7th, 2023 (5 working days)

2. System Design and Architecture: November 8th, 2023 - November 10th, 2023 (3 working days)

3. Development: November 13th, 2023 - November 15th, 2023 (3 working days)

4. Testing and Quality Assurance:November 16th, 2023 - November 17th, 2023 (2 working days)

5. Deployment: November 20th, 2023 - November 21st, 2023 (2 working day

1. **Risk Management**

**6.1 Identified Risks**

- Technical challenges in ensuring the security of the online voting system.

- Potential changes in project scope driven by regulatory updates.

- Risks associated with external factors affecting the online voting process.

**6.2 Risk Mitigation Strategies**

- Conduct rigorous security testing and implement encryption protocols.

- Establish a change control process to manage scope changes.

- Continuously monitor external factors and adapt the system accordingly.

1. **Communication Plan**

**7.1 Stakeholder Communication**

Stakeholders will receive regular updates through bi-weekly progress reports and monthly status meetings.

**7.2 Team Communication**

The project team will utilize collaboration tools for daily stand-ups and discussions, ensuring efficient communication.

1. **Quality Management**

**8.1 Testing Procedures**

Comprehensive testing, including unit testing and user acceptance testing, will be conducted throughout the development lifecycle.

**8.2 Acceptance Criteria**

The online electoral system must meet predefined criteria for functionality, security, and user experience to ensure a reliable and trustworthy voting platform.

**Implementation:**

**Source code :**

PYTHON  
import streamlit as st

import json

import pandas as pd

import plotly.express as px

# Initial votes (will be used only once at the start)

initial\_votes = {"Mr.Srihariharan": 0, "Mr.Sabareeswaran": 0, "Miss.Akshaya Saravanan": 0}

def save\_votes\_to\_file(votes\_data):

with open("votes.json", "w") as file:

json.dump(votes\_data, file)

def load\_votes\_from\_file():

try:

with open("votes.json", "r") as file:

return json.load(file)

except FileNotFoundError:

return {}

def register\_user(username, password):

users = st.session\_state.users

users[username] = password

def login\_user(username, password):

users = st.session\_state.users

return username in users and users[username] == password

def has\_user\_voted(username):

return username in st.session\_state.voted\_users

def set\_user\_voted(username):

st.session\_state.voted\_users.add(username)

def display\_home():

st.write("Welcome to the CEO Election System. Cast your vote for the next CEO of Margazhi Designs !")

def display\_registration():

st.title("Register")

username = st.text\_input("Username")

password = st.text\_input("Password", type="password")

if st.button("Register"):

register\_user(username, password)

st.success("Registration successful. You can now log in.")

def display\_login():

st.title("Login")

username = st.text\_input("Username")

password = st.text\_input("Password", type="password")

if st.button("Login"):

if login\_user(username, password):

st.success(f"Welcome, {username}!")

st.session\_state.logged\_in\_user = username

else:

st.error("Invalid username or password. Please try again.")

def display\_candidates():

st.title("About Candidates")

candidates\_info = [

("Mr. Srihariharan", "Mr. Srihariharan is a visionary leader with a proven track record..."),

("Mr. Sabareeswaran", "Mr. Sabareeswaran is a seasoned executive known for his strategic mindset..."),

("Miss. Akshaya Saravanan", "Miss. Akshaya Saravanan is a dynamic and forward-thinking professional..."),

]

for candidate in candidates\_info:

st.title(candidate[0])

st.write(candidate[1])

def display\_voting():

st.write("Voting Page")

if 'logged\_in\_user' not in st.session\_state:

st.warning("You need to log in to vote.")

else:

if has\_user\_voted(st.session\_state.logged\_in\_user):

st.warning("You have already voted. Cannot vote again.")

else:

candidate\_options = ["Mr.Srihariharan", "Mr.Sabareeswaran", "Miss.Akshaya Saravanan"]

selected\_candidate = st.selectbox("Select your preferred candidate:", ["Select Candidate"] + candidate\_options)

if selected\_candidate == "Select Candidate":

st.warning("Please select a candidate before submitting your vote.")

else:

if selected\_candidate not in votes:

votes[selected\_candidate] = 0

if st.button("Submit Vote"):

votes[selected\_candidate] += 1

save\_votes\_to\_file(votes)

set\_user\_voted(st.session\_state.logged\_in\_user)

st.success(f"You ({st.session\_state.logged\_in\_user}) voted for {selected\_candidate}. Thank you for participating!")

def display\_analytics():

st.write("Analytics Page")

st.write("Election Analytics")

if not votes:

st.warning("No votes recorded yet.")

else:

df\_votes = pd.DataFrame(list(votes.items()), columns=["Candidate", "Votes"])

fig = px.bar(df\_votes, x="Candidate", y="Votes", title="Vote Counts", labels={"Votes": "Number of Votes"})

st.plotly\_chart(fig)

# Load existing votes when the app starts

votes = load\_votes\_from\_file()

# Initialize session state

if 'users' not in st.session\_state:

st.session\_state.users = {}

if 'voted\_users' not in st.session\_state:

st.session\_state.voted\_users = set()

# Streamlit App

st.title("CEO Election System")

# Navigation Bar

page = st.sidebar.radio("Select a page", ["Home", "Register", "Login", "About Candidates", "Vote", "Analytics"])

# Display respective pages based on user's selection

if page == "Home":

display\_home()

elif page == "Register":

display\_registration()

elif page == "Login":

display\_login()

elif page == "About Candidates":

display\_candidates()

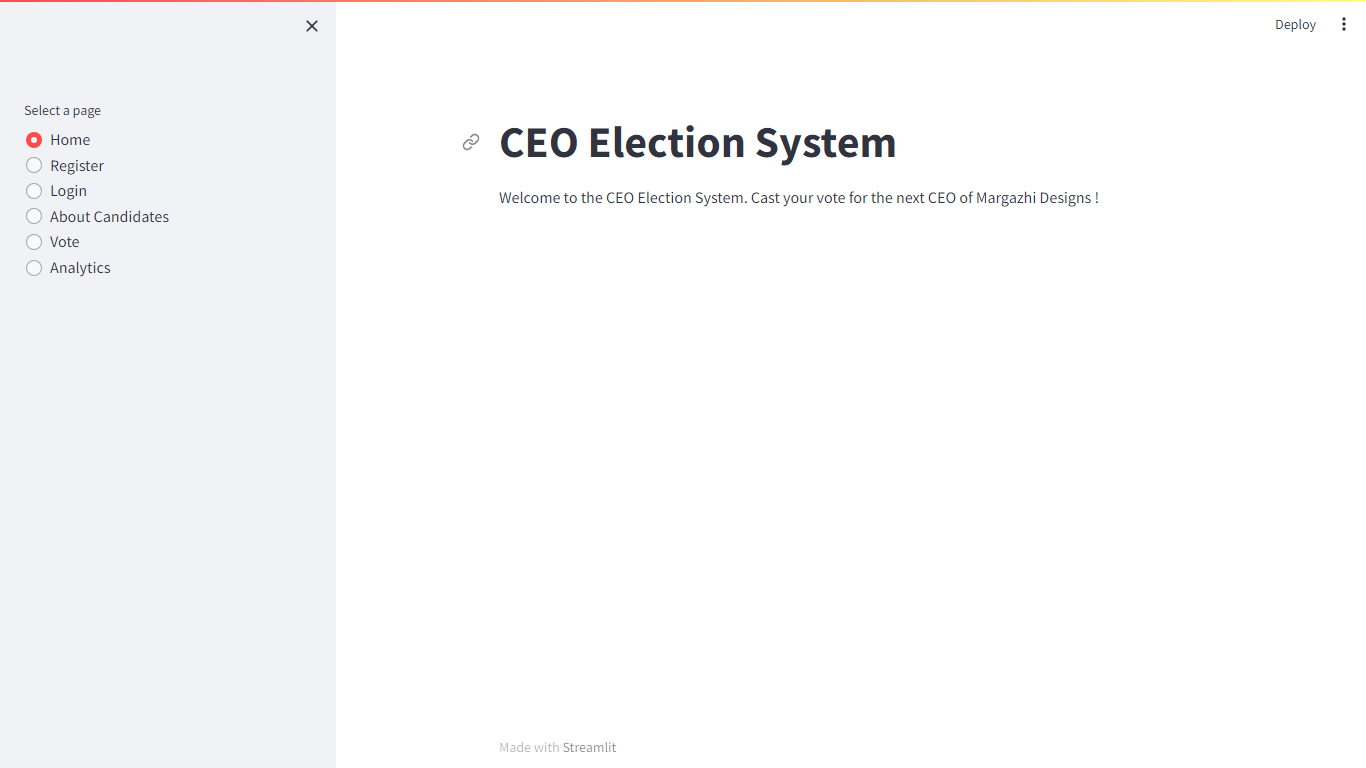
elif page == "Vote":

display\_voting()

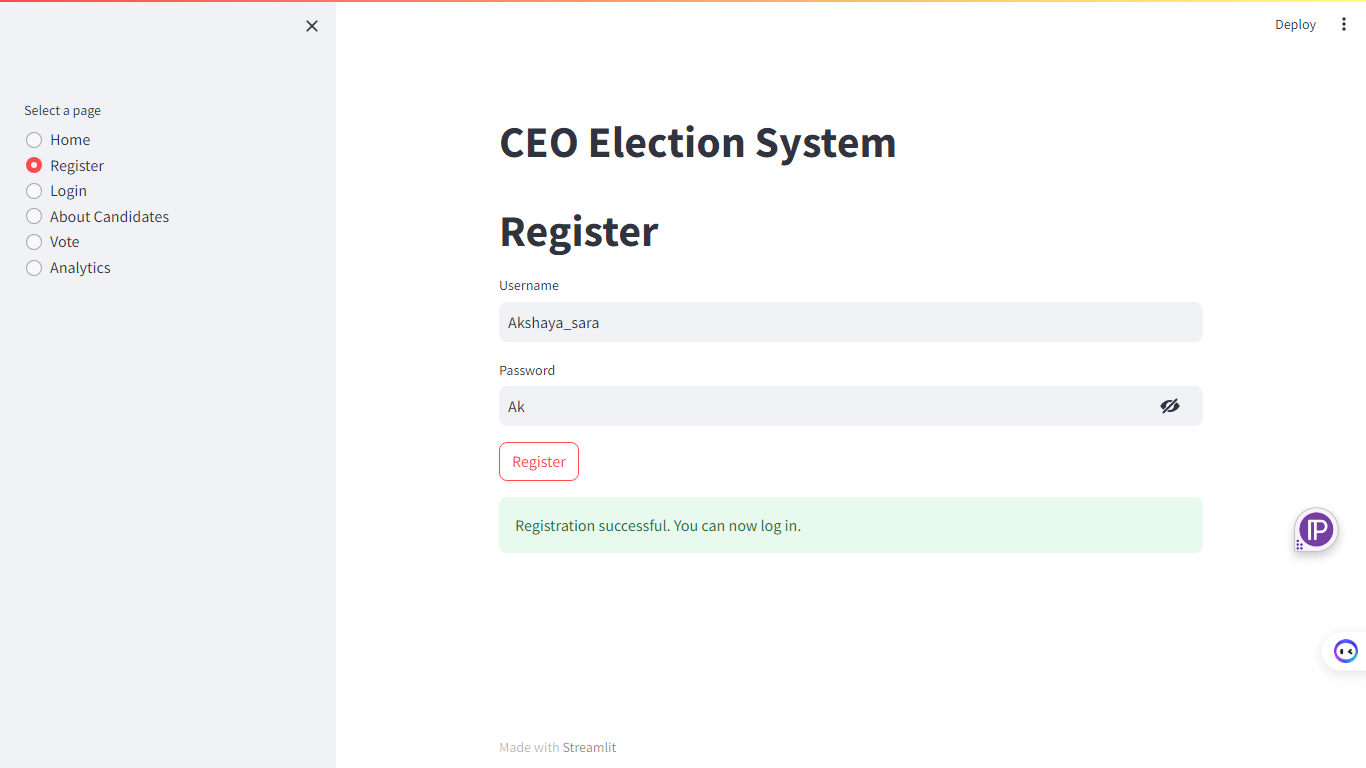
elif page == "Analytics":

display\_analytics()

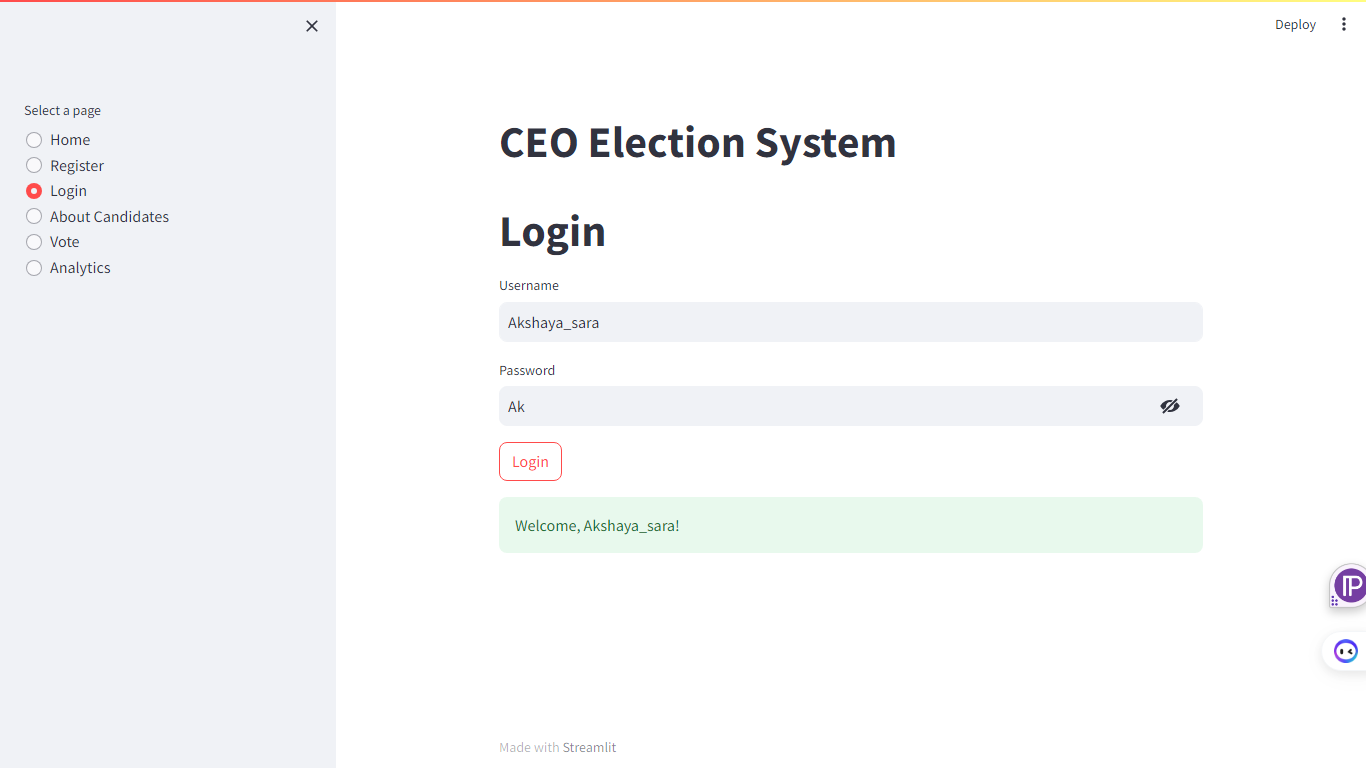
**OUTPUT :**

**HOME PAGE**

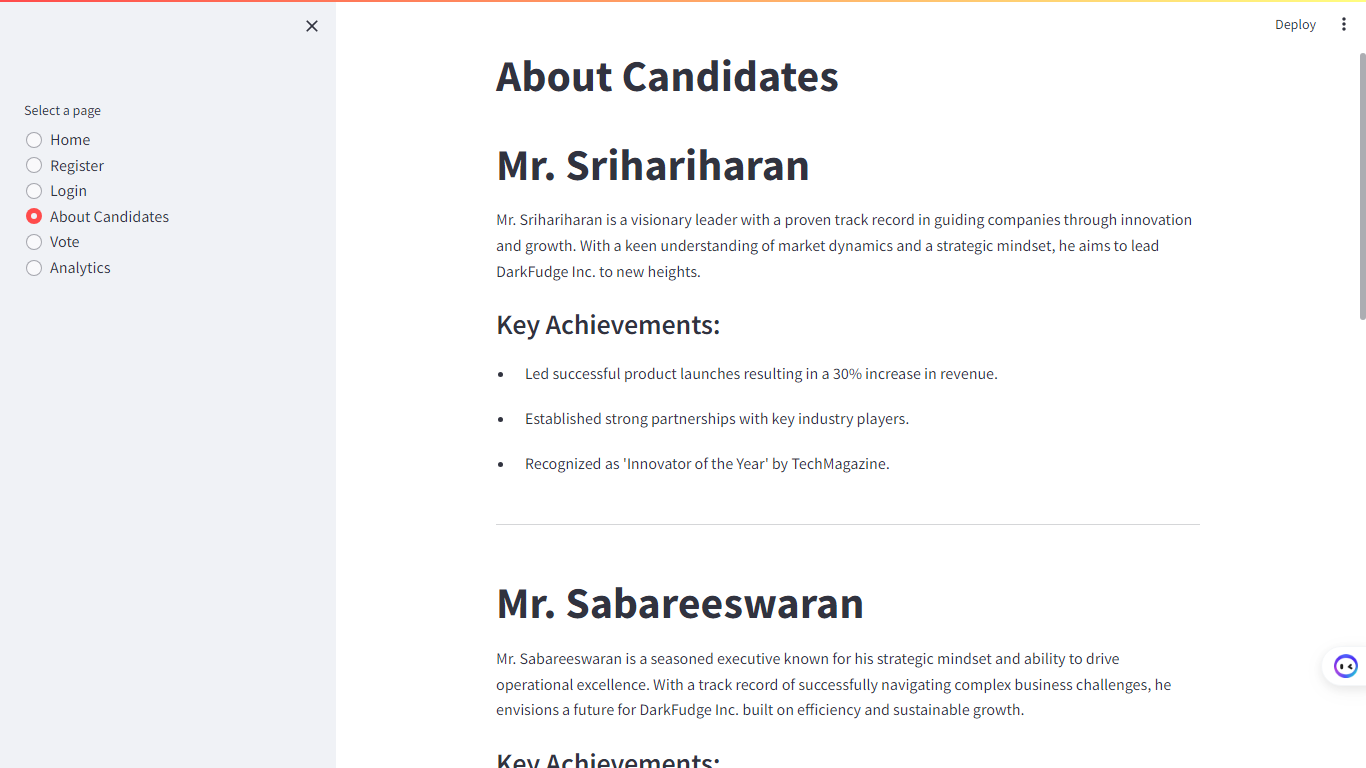
**REGISTRATION PAGE**

****

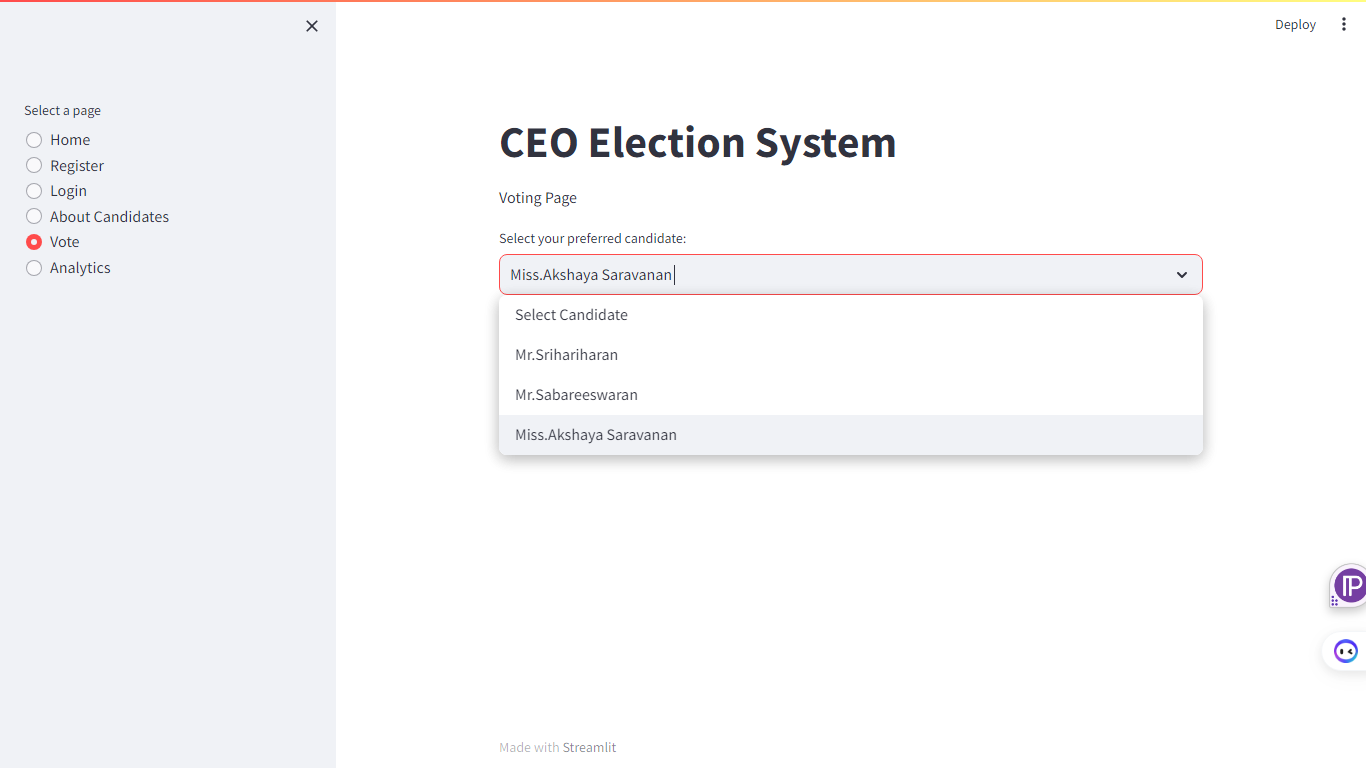
**LOGIN PAGE**

****

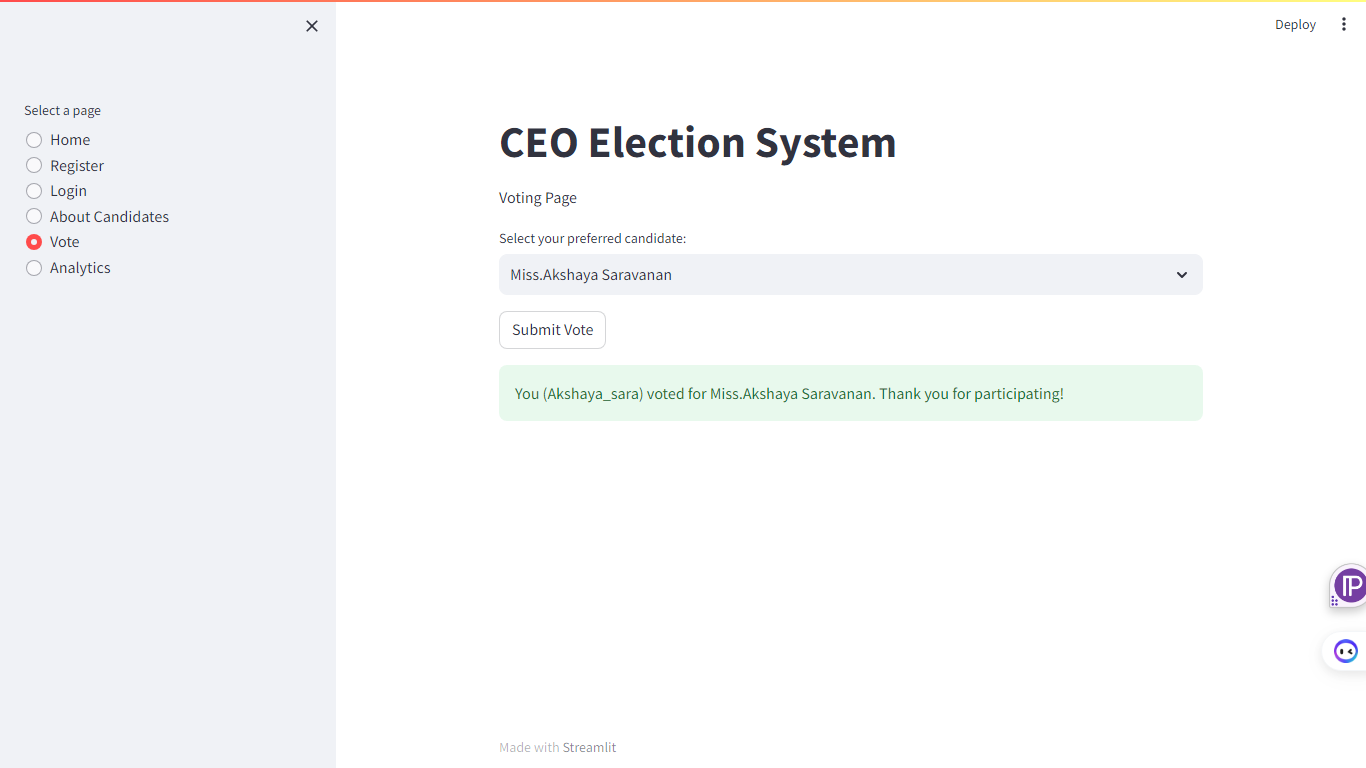
**ABOUT CANDIDATES PAGE**

****

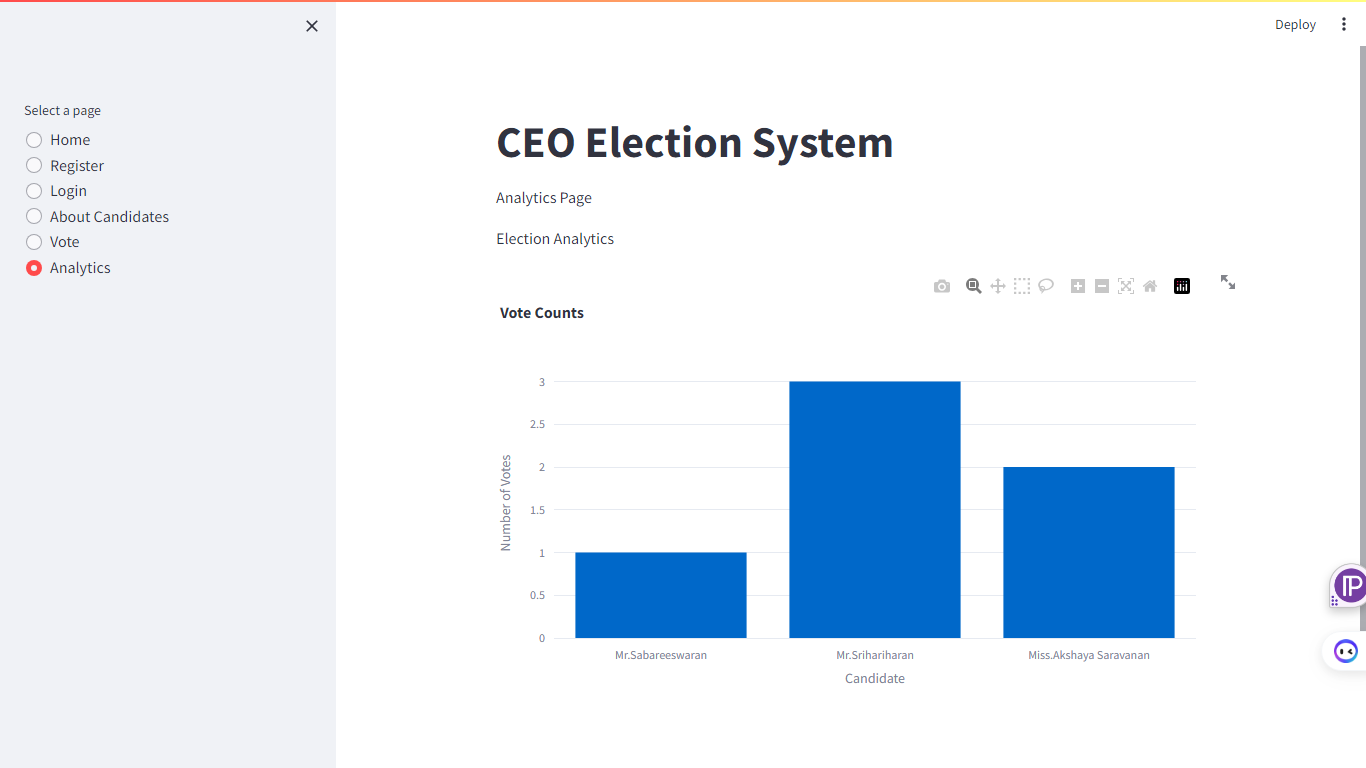
**CASTING VOTE PAGE**

****

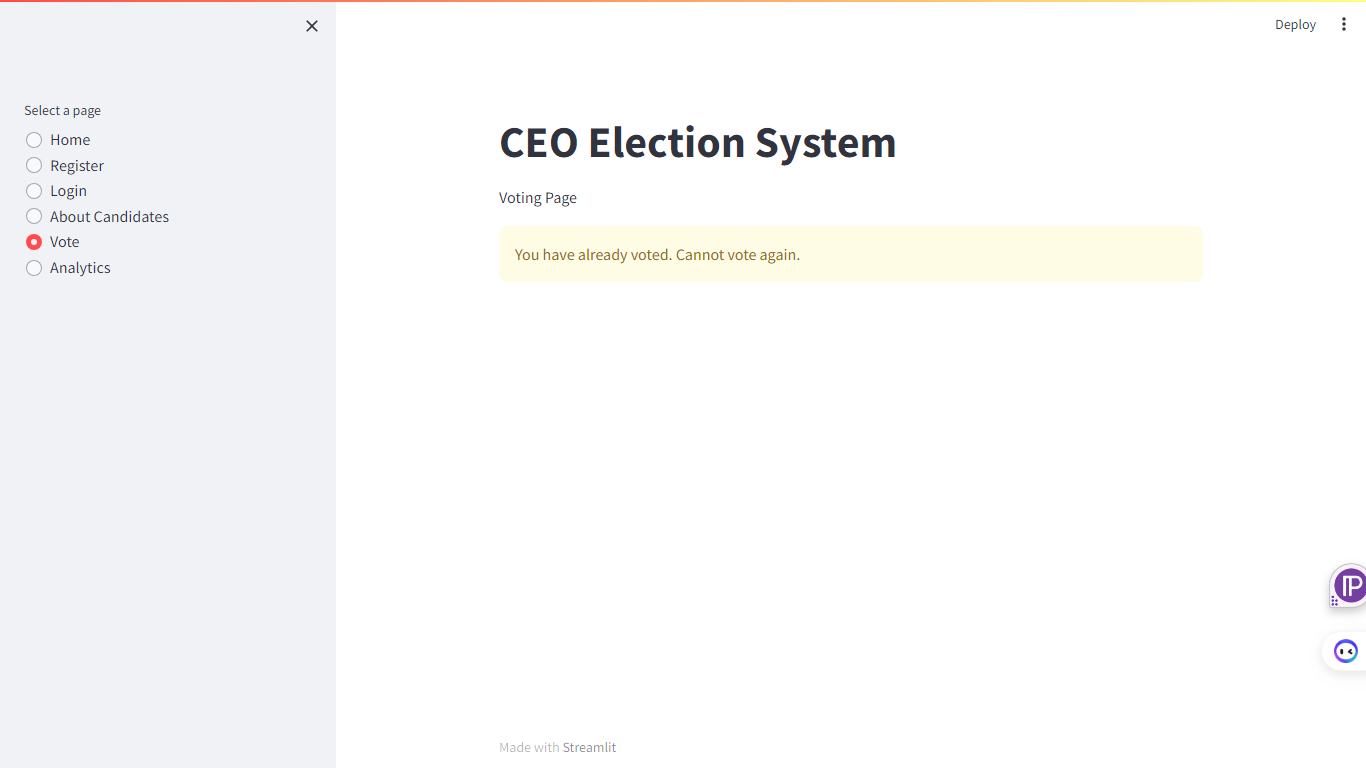
**AFTER CASTING THE VOTE**

****

**ANALYSIS PAGE**

****

**VOTE CASTED USERS CANNOT VOTE AGAIN**

****

**Implementation Of Test Case For Online Electoral System**

1. Test Case ID: TC001

- Input:register\_user("user1", "password1")

- Expected Output:{"user1": "password1"}

- Actual Output:{"user1": "password1"}

- Status:Pass

2. Test Case ID: TC002

- Input:register\_user("user1", "password2")

- Expected Output: {"user1": "password1"}

- Actual Output:{"user1": "password1"}

- Status: Pass

3. Test Case ID: TC003

- Input:\* login\_user("user1", "password1")

- Expected Output:True

- Actual Output:True

- Status:Pass

4. Test Case ID: TC004

- Input:login\_user("nonexistent\_user", "password")

- Expected Output: False

- Actual Output: False

- Status: Pass

5. Test Case ID: TC005

- Input:login\_user("user2", "incorrect\_password")

- Expected Output: False

- Actual Output:False

- Status: Pass

6. Test Case ID: TC006

- Input: has\_user\_voted("user1")

- Expected Output: False

- Actual Output:False

- Status: Pass

7. Test Case ID: TC007\*

- Input: set\_user\_voted("user1")

- Expected Output:True

- Actual Output:True

- Status: Pass

8. Test Case ID: TC008

- Input: has\_user\_voted("user1")

- Expected Output: True

- Actual Output: True

- Status: Pass

9. Test Case ID: TC009

- \*Input:\* has\_user\_voted("user2")

- \*Expected Output:\* False

- \*Actual Output:\* False

- \*Status:\* Pass

10. Test Case ID: TC010

- \*Input:\* display\_voting() (first vote)

- \*Expected Output:\* "You (user1) voted for candidate1."

- \*Actual Output:\* "You (user1) voted for candidate1."

- \*Status:\* Pass

11. Test Case ID: TC011

- \*Input:\* display\_voting() (attempt to vote twice)

- \*Expected Output:\* "You have already voted. Cannot vote again."

- \*Actual Output:\* "You have already voted. Cannot vote again."

- \*Status:\* Pass

**Conclusion**

In conclusion, All test cases passed successfully, indicating that the CEO Election System modules are functioning as intended. The registration, login, and voting processes have been validated, and the system behaves appropriately under various scenarios. It is recommended to continue testing with additional cases and scale the testing process as the project evolves. Additionally, integration testing should be conducted to ensure the seamless interaction of different modules.

**References:**

* **Official International Electoral Organizations**
* **Government Websites**
* **Journals and Publications**
* **News articles and reports**