ONLINE VOTING SYSTEM

A PROJECT REPORT

Submitted by

MOHAMMED RAASITH H (2303811724321069)

in partial fulfillment of requirements for the award of the course

CGB1221-DATABASE MANAGEMENT SYSTEMS

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

JUNE- 2025

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"ONLINE VOTING SYSTEM"** is the bonafide work of **MOHAMMED RAASITH H (2303811724321069)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE

Dr.T. AVUDAIAPPAN, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

ASSOCIATE PROFESSOR

Department of Artificial Intelligence

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

SIGNATURE

Mrs.S. GEETHA, M.E.,

SUPERVISOR

ASSISTANT PROFESSOR

Department of Artificial Intelligence

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on …04.06.2025…….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

I declare that the project report on **"ONLINE VOTING SYSTEM "** is the result of original work done by me and best of my knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1221 – DATABASE MANAGEMENT SYSTEMS.**

.

Signature

MOHAMMED RAASITH H

Place: Samayapuram

Date: 04.06.2025

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. T. AVUDAIAPPAN, M.E.,Ph.D.,** Head of the department, **ARTIFICIAL INTELLIGENCE** for providing his encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs.S.GEETHA, M.E.,** Department of **ARTIFICIAL INTELLIGENCE,** for her incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

INSTITUTE

Vision:

- To serve the society by offering top-notch technical education on par with global standards.

Mission:

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all – round personalities respecting moral and ethical values.

DEPARTMENT

Vision:

- To excel in education, innovation, and research in Artificial Intelligence and Data Science to fulfil industrial demands and societal expectations.

Mission

- To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.
- To collaborate with industry and offer top-notch facilities in a conducive learning environment.
- To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.
- To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEO)

- **PEO1:** Compete on a global scale for a professional career in Artificial Intelligence and Data Science.
- **PEO2:** Provide industry-specific solutions for the society with effective communication and ethics.
- **PEO3** Enhance their professional skills through research and lifelong learning initiatives.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1:** Capable of finding the important factors in large datasets, simplify the data, and improve predictive model accuracy.
- **PSO2:** Capable of analyzing and providing a solution to a given real-world problem by designing an effective program.

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

In today's rapidly evolving digital governance ecosystem, managing elections manually is both time-consuming and prone to errors, highlighting the need for a secure and streamlined system—especially in schools, colleges, organizations, or community platforms. This project, titled "Online Voting System", offers a lightweight, user-friendly, and efficient interface for conducting online polls and secure voting processes. Developed using Python, it utilizes SQLite as the database backend and incorporates HTML and CSS for designing a responsive and accessible frontend. The system leverages the Gradio package to create an interactive browser-based UI, making it compatible with both Google Colab and VS Code. The application allows users to register, log in, and cast votes securely, while administrators can manage users and candidates, monitor vote counts, and view real-time results. All vote-related data is securely stored using a structured SQLite schema, ensuring consistency and relational integrity. Supporting full CRUD operations and authentication to ensure one vote per user, the system also provides raw table views for administrative audits. With multiple tabs like User Login, User Registration, and Admin Panel, the system ensures a seamless and engaging user experience. The modular architecture allows future enhancements like OTP authentication or real-time dashboards.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD DATABASES FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
Online Voting System is a secure, user-friendly application designed to simplify and streamline the election process in schools, colleges, organizations, and communities. Developed using Python with SQLite as the backend and HTML/CSS for a responsive frontend, the system uses the Gradio package to offer a browser-based interface compatible with Google Colab and VS Code. It allows users to register, log in, and securely cast a single vote, while administrators can manage users and candidates, monitor vote counts, and view real-time results.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -1** **PO5 -3** **PO6 -2** **PO7 -1** **PO8 -2** **PO9 -1** **PO10 -1** **PO11-2** **PO12-1**	**PSO1 -2** **PSO2 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

The primary objective of this project is to design and implement a database-centric online voting system using Python, SQLite, and Gradio that efficiently manages user registration, authentication, candidate management, voting, and real-time result tracking. The system ensures data consistency and integrity through a well-structured relational database with tables for users, admin, candidates, and voting records, while enforcing one-vote-per-user restrictions. By providing role-based access controls for voters and administrators, the system supports seamless operations including secure login, vote casting, administrative result monitoring, and direct inspection of database tables for transparency and efficiency analysis. The project aims to demonstrate the integration of backend database management, frontend interactive UI design, and secure voting logic to build a scalable and reliable online polling platform.

1.2 OVERVIEW

This project implements a robust online voting system that integrates a well-structured SQLite database to manage users, administrators, candidates, and voting records efficiently. The system supports essential features such as user registration, secure login, vote casting with a one-vote-per-user restriction, and administrative controls for monitoring election results and database contents. The database schema employs normalized tables with primary keys to maintain data integrity, while SQL queries facilitate reliable data operations including insertion, retrieval, and updates. A Gradio-based frontend provides an interactive and user-friendly interface with tabs for user login, registration, voting, and an admin panel for managing the election process

1.3 SQL AND DATABASE CONCEPTS USED

Structured Query Language (SQL) is fundamental to the functionality of this online voting system, facilitating seamless communication with the SQLite relational database. The project employs normalized tables such as users, admin, candidates, and voted_users to ensure data consistency and avoid redundancy. Primary keys uniquely identify records in each table, for example, username in the users table and candidate_id in the candidates table. Constraints ensure data integrity, such as preventing duplicate votes by tracking usernames in the voted_users table. SQL statements including CREATE TABLE, INSERT, SELECT, and UPDATE are extensively used to manage user registrations, authenticate logins, record votes, and update candidate vote counts. The database schema supports essential relational mappings reflecting real-world entities and interactions, ensuring accurate and secure vote processing.

CHAPTER 2

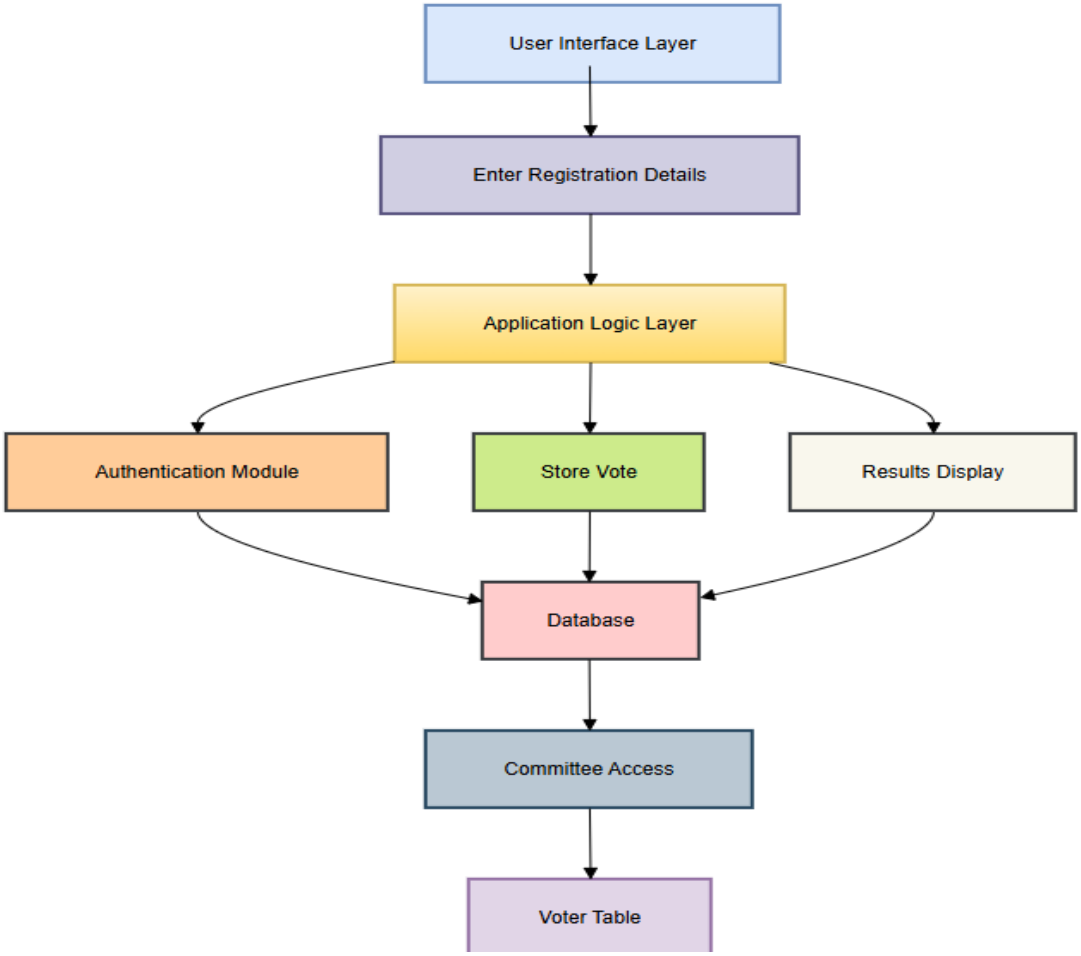
PROJECT METHODOLOGY

2.1 PROPOSED WORK

The Online Voting System is developed to provide a streamlined and efficient platform for conducting online polls and surveys, with a primary focus on the underlying database architecture. The system is structured into distinct layers including the user interface, application logic, authentication, and a relational database that manages all core data related to users, polls, and votes. The database schema is designed using relational concepts such as primary and foreign keys, indexes, and constraints to ensure data integrity, consistency, and optimized query performance. Key entities include polls with associated options, user profiles with defined roles, and voting records that track participation, selected choices, and timestamps.

This design enables dynamic management of polls through CRUD functionalities, allowing real-time updates between the frontend and backend systems. Authentication mechanisms secure user access, with administrative controls facilitating the creation of new polls, monitoring of voting activity, and management of user permissions. By centralizing data storage and operations, the system supports concurrent voting sessions and real-time result updates, providing a scalable and reliable solution. The project highlights the practical application of database management principles such as normalization, referential integrity, and transaction management, ensuring efficient data handling in a real-world polling environment.

2.2 BLOCK DIAGRAM



CHAPTER 3

MODULE DESCRIPTION

The system architecture is divided into multiple interdependent modules, each dedicated to handling a specific segment of the voting workflow. This modular structure ensures the platform remains scalable, easy to manage, and efficient in terms of database integration, user interaction, and administrative control. Each module interacts with the centralized database to ensure real-time data consistency, transactional integrity, and seamless user experience.

3.1 USER REGISTRATION

This module allows new users to register and existing users to log in using their credentials. Registration details such as name, contact information and password are stored securely in the database. The login system verifies these credentials to authenticate users. Once authenticated, users are granted access to participate in polls. This module ensures session validation and acts as a gateway to the other system features.

3.2USER LOGIN

This module provides tools for administrators to create and manage polls. Each poll includes details such as poll title, description, start and end dates, and multiple voting options. Admins can add, edit, or delete polls and options through a backend interface. Poll status is dynamically updated based on timing and user participation, ensuring that real-time poll data is accurately reflected on the user interface.Booking System

3.3CASTING A VOTE

This core module allows users to cast votes in active polls. Upon selecting a poll and an option, the system records the vote in the database, linking it to the user ID and poll option. The module ensures that users cannot vote multiple times in the same poll by enforcing constraints or conditional logic in the backend

3.4 VOTE RECORDED – CONFIRMATION

After votes are cast, this module calculates the results by aggregating votes per option. The calculated results and poll summaries are displayed to users in real-time, with charts or tables showing voting percentages and counts. This module ensures accurate result calculation and timely updates while maintaining the integrity of the voting process.Admin Overview Panel

3.5 ADMIN VIEWING THE RESULTS

This module is accessible only to authorized administrators and provides tools for managing user accounts, polls, votes, and results. It allows for manual updates, poll closures, or modifications of records. The panel presents a structured overview of all key operations, including user activity, active polls, and voting statistics. All information is fetched directly from the database using well-structured SQL queries, ensuring up-to-date data visibility and administrative control.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

CONCLUSION

The Online Voting System was successfully developed as a database-centric application aimed at automating and streamlining the process of conducting online polls and managing vote collection. The system ensures efficient management of user data, poll details, voting records, and real-time result display, all integrated through a structured and relational database schema. Through proper implementation of CRUD operations, data validation, and constraints, the project has demonstrated the practical application of database management concepts in a real-world scenario. The centralized database not only supports smooth front-end operations but also enhances backend consistency and reliability. The modular design of the system allows for clear separation of concerns between voters and administrators, making the platform scalable and maintainable.

FURURE SCOPE

Looking ahead, the project has significant potential for future enhancement. Features such as anonymous voting, multi-option polls, result analytics dashboards, real-time notifications, and user feedback modules can be added to further improve functionality and user experience. Additionally, the system could be extended into a mobile application or cloud-based service, enabling broader accessibility and real-time synchronization across devices. These improvements would transform the platform into a comprehensive, enterprise-grade online voting solution.

APPENDIX A – SOURCE CODE

SQLITE CODE FOR DATABASE INITIALIZATION

```
# Create users table
CREATE TABLE IF NOT EXISTS users (
    username TEXT PRIMARY KEY,
    password TEXT NOT NULL
);


#Create admin table
CREATE TABLE IF NOT EXISTS admin (
    adminname TEXT PRIMARY KEY,
    password TEXT NOT NULL
);


#Create candidates table
CREATE TABLE IF NOT EXISTS candidates (
    candidate_id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    votes INTEGER DEFAULT 0
);


#Create voted_users table
CREATE TABLE IF NOT EXISTS voted_users (
    username TEXT PRIMARY KEY
);


#Insert default admin
INSERT OR IGNORE INTO admin (adminname, password)
VALUES ('admin', 'admin123');


# Insert default candidates
INSERT OR IGNORE INTO candidates (name, votes) VALUES ('Alice', 0);
INSERT OR IGNORE INTO candidates (name, votes) VALUES ('Bob', 0);
```

SQL QUERIES TO VIEW TABLE DATA

```
#View all users
SELECT * FROM users;


# View admin credentials
SELECT * FROM admin;


# View candidates and vote counts
SELECT * FROM candidates;


# View who has voted
SELECT * FROM voted_users;
```

CONNECTION TO CREATE AND INTERFACE OF THE MODEL

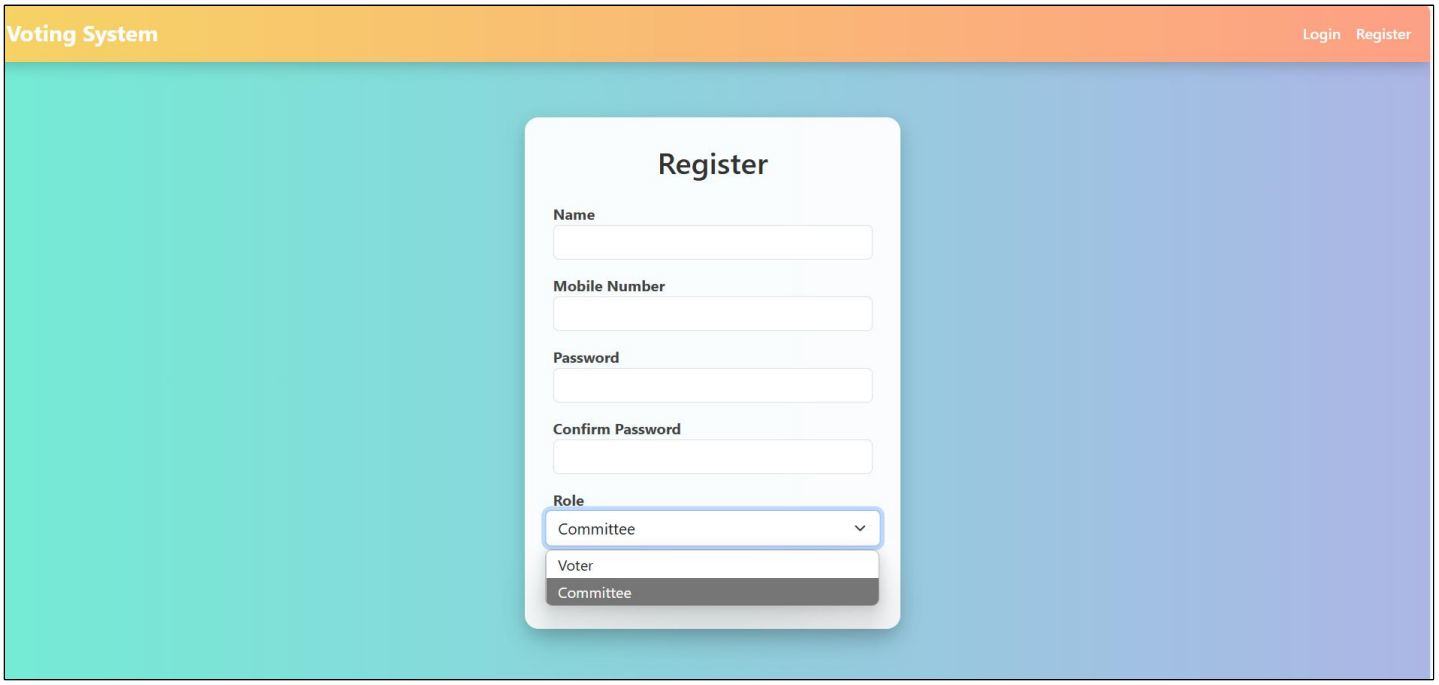
```
import sqlite3
import gradio as gr

# Connect to or create the database
conn = sqlite3.connect('voting_system.db', check_same_thread=False)
cursor = conn.cursor()

# Initialize database tables
def init_db():
    cursor.execute('''CREATE TABLE IF NOT EXISTS users (
                      username TEXT PRIMARY KEY,
                      password TEXT NOT NULL)''')

    cursor.execute('''CREATE TABLE IF NOT EXISTS admin (
                      adminname TEXT PRIMARY KEY,
                      password TEXT NOT NULL)''')

    cursor.execute('''CREATE TABLE IF NOT EXISTS candidates (
                      candidate_id INTEGER PRIMARY KEY AUTOINCREMENT,
                      name TEXT NOT NULL,
                      votes INTEGER DEFAULT 0)''')
```

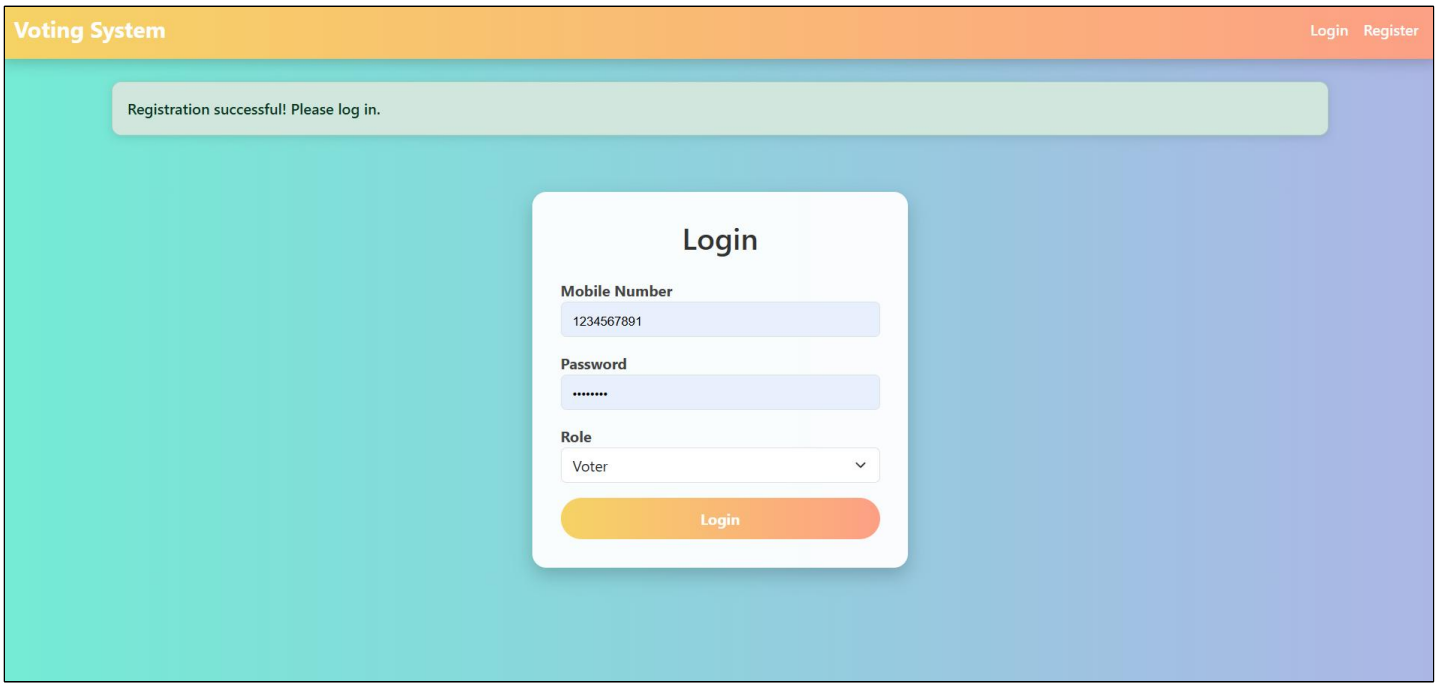
```
    cursor.execute('''CREATE TABLE IF NOT EXISTS voted_users (
                      username TEXT PRIMARY KEY)''')

    # Insert a default admin and candidates if not exist
    cursor.execute("INSERT OR IGNORE INTO admin VALUES ('admin', 'admin123')")
    cursor.execute("INSERT OR IGNORE INTO candidates (name, votes) VALUES ('Alice', 0)")
    cursor.execute("INSERT OR IGNORE INTO candidates (name, votes) VALUES ('Bob', 0)")
    conn.commit()

init_db()

# USER Login
def user_login(username, password):
    cursor.execute("SELECT * FROM users WHERE username=? AND password=?", (username, password))
    if cursor.fetchone():
        return gr.update(visible=False), gr.update(visible=True), f"Welcome, {username}!"
    return gr.update(), gr.update(), "Invalid User Credentials."

# ADMIN Login
def admin_login(username, password):
    cursor.execute("SELECT * FROM admin WHERE adminname=? AND password=?", (username, password))
    if cursor.fetchone():
        return gr.update(visible=False), gr.update(visible=True), f"Admin logged in!"
    return gr.update(), gr.update(), "Invalid Admin Credentials."

# Registration
def register_user(username, password):
    try:
        cursor.execute("INSERT INTO users VALUES (?, ?)", (username, password))
        conn.commit()
        return "User registered successfully."
    except:
        return "Username already exists."

# Voting function
def vote(candidate_name, username):
    cursor.execute("SELECT * FROM voted_users WHERE username=?", (username,))
    if cursor.fetchone():
        return "You have already voted."

    cursor.execute("UPDATE candidates SET votes = votes + 1 WHERE name=?",
```

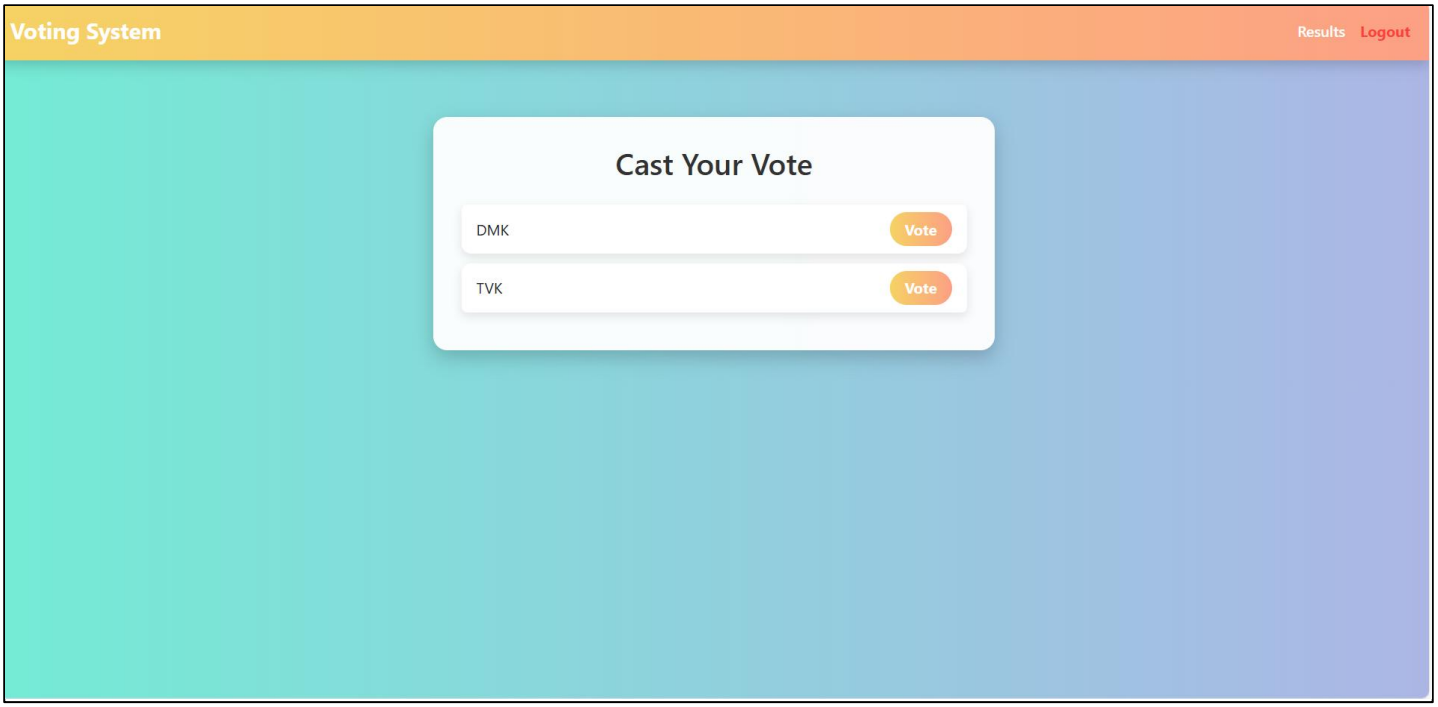
```
(candidate_name,))
    cursor.execute("INSERT INTO voted_users VALUES (?)", (username,))
    conn.commit()
    return f"Thank you for voting for {candidate_name}!"

# View Results (Admin)
def view_results():
    cursor.execute("SELECT * FROM candidates")
    results = cursor.fetchall()
    result_str = "Candidate Vote Count:\n"
    for cand in results:
        result_str += f"{cand[1]}: {cand[2]} votes\n"
    return result_str

# View Tables
def view_table(table_name):
    try:
        cursor.execute(f"SELECT * FROM {table_name}")
        data = cursor.fetchall()
        return f"{table_name}:\n" + "\n".join(str(row) for row in data)
    except Exception as e:
        return f"Error: {e}"

# UI with Gradio
with gr.Blocks(theme=gr.themes.Base(primary_hue="blue", secondary_hue="gray")) as demo:
    gr.Markdown("# 🗳️ ONLINE VOTING SYSTEM\n### Please Login/Register to Continue")

    with gr.Tab("User Login"):
        username = gr.Textbox(label="Username")
        password = gr.Textbox(label="Password", type="password")
        login_btn = gr.Button("Login")
        user_status = gr.Textbox(label="Status")
        user_panel = gr.Group(visible=False)

        login_btn.click(user_login, inputs=[username, password],
                        outputs=[login_btn, user_panel, user_status])

        with user_panel:
            gr.Markdown("### Vote for a Candidate")
            candidate = gr.Dropdown(choices=["Alice", "Bob"], label="Select Candidate")
            vote_btn = gr.Button("Vote")
            vote_result = gr.Textbox(label="Result")
            vote_btn.click(vote, inputs=[candidate, username], outputs=vote_result)
```

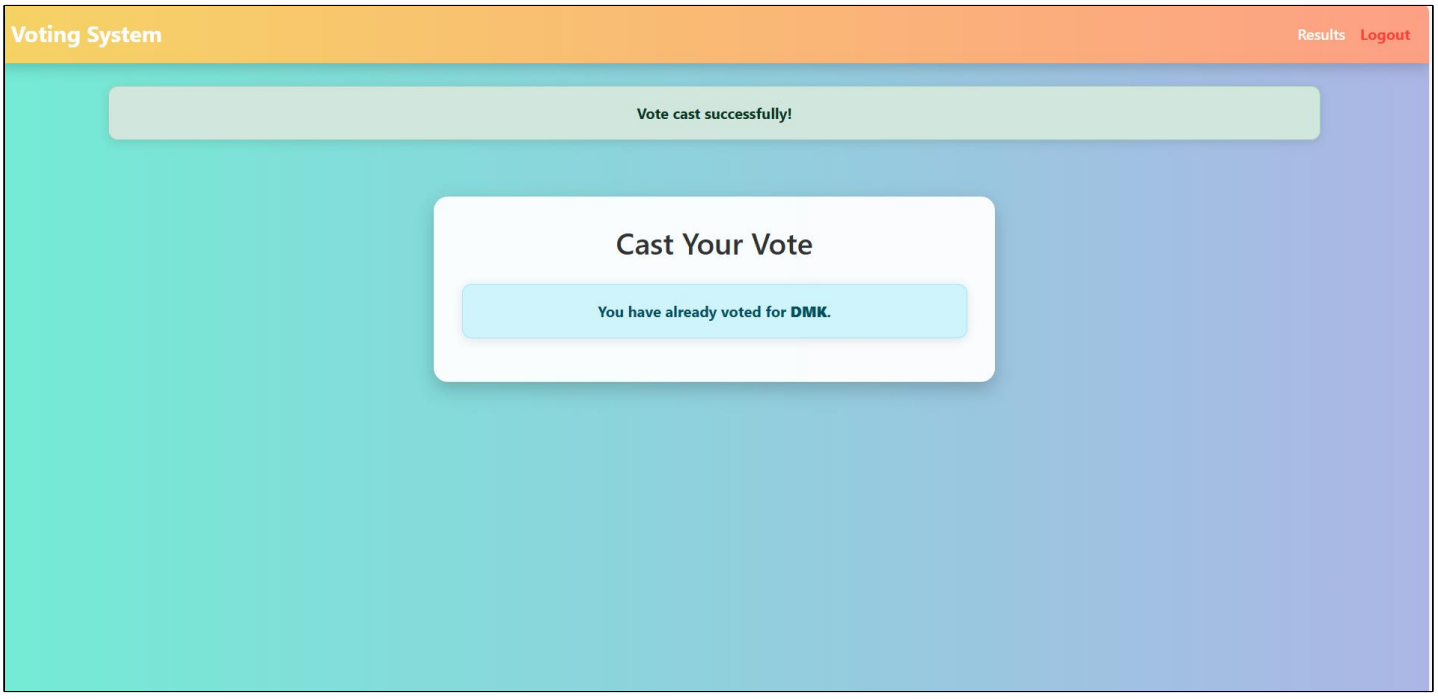
```
    with gr.Tab("User Registration"):
        reg_username = gr.Textbox(label="Choose Username")
        reg_password = gr.Textbox(label="Choose Password", type="password")
        reg_btn = gr.Button("Register")
        reg_status = gr.Textbox(label="Status")
        reg_btn.click(register_user, inputs=[reg_username, reg_password],
outputs=reg_status)

    with gr.Tab("Admin Panel"):
        admin_name = gr.Textbox(label="Admin Username")
        admin_pass = gr.Textbox(label="Admin Password", type="password")
        admin_btn = gr.Button("Admin Login")
        admin_status = gr.Textbox(label="Status")
        admin_panel = gr.Group(visible=False)

        admin_btn.click(admin_login, inputs=[admin_name, admin_pass],
                        outputs=[admin_btn, admin_panel, admin_status])

        with admin_panel:
            gr.Markdown("### View Voting Results")
            result_btn = gr.Button("Show Results")
            result_out = gr.Textbox(label="Voting Results")
            result_btn.click(view_results, outputs=result_out)

            gr.Markdown("### View Table Data (for Efficiency Analysis)")
            table_name = gr.Dropdown(choices=["users", "admin", "candidates",
"voted_users"], label="Select Table")
            table_btn = gr.Button("Show Table Data")
            table_out = gr.Textbox(label="Table Data", lines=10)
            table_btn.click(view_table, inputs=table_name, outputs=table_out)

demo.launch()
```

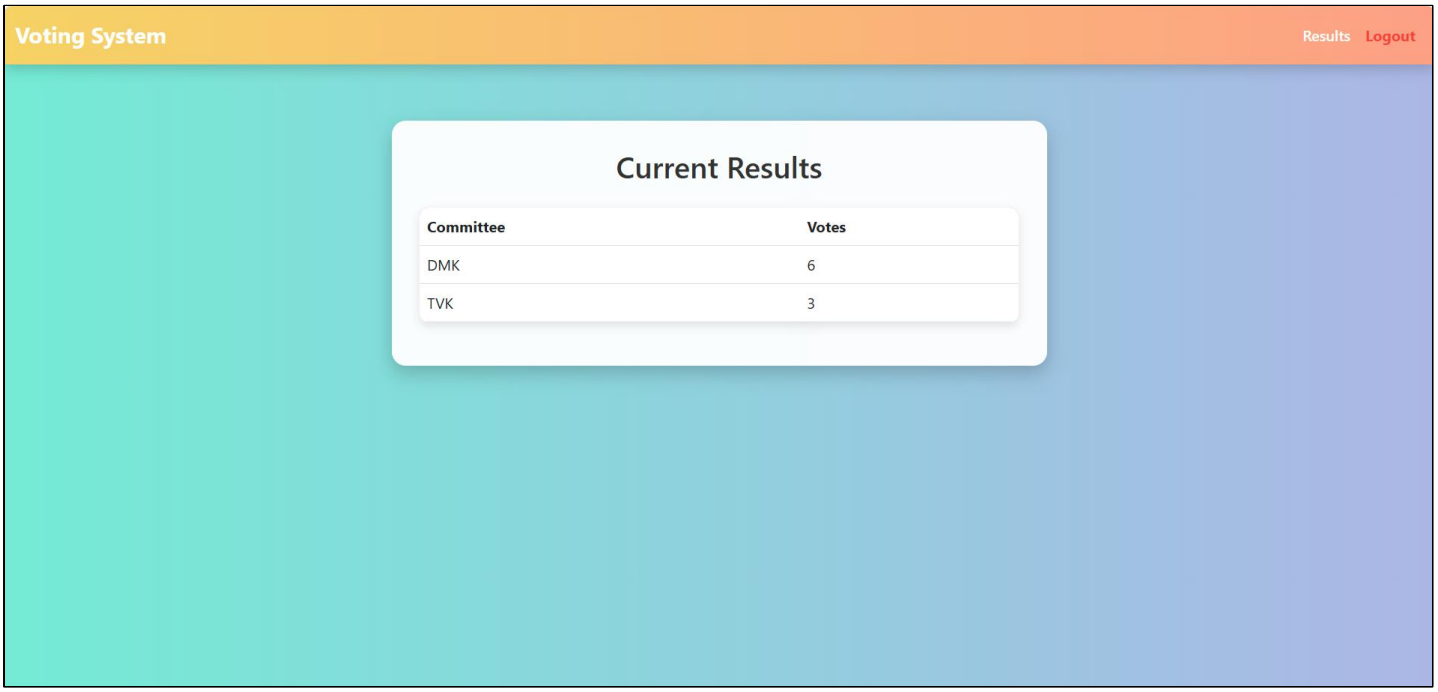
APPENDIX B – SCREENSHOTS

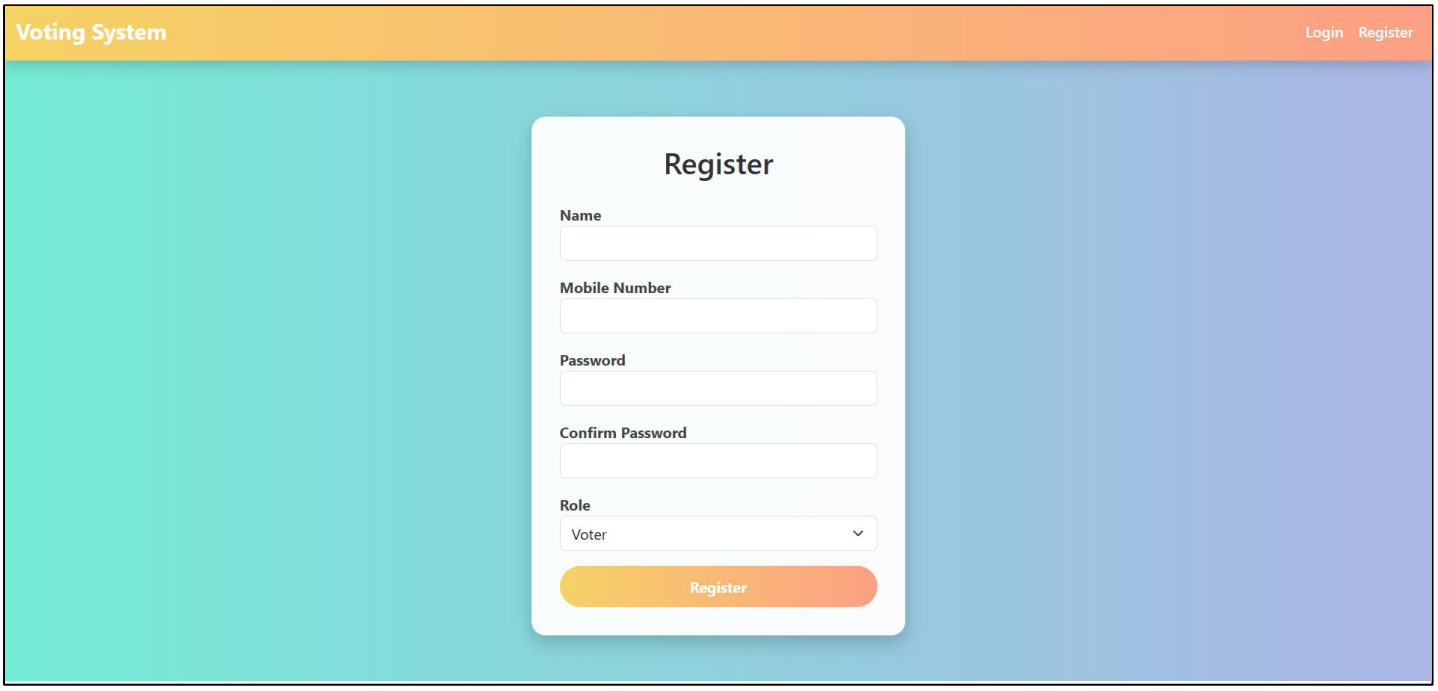





Voting System
Results
Logout
Cast Your Vote
DMK
Vote
TVK
Vote


Voting System
Results
Logout
Vote cast successfully!
Cast Your Vote
You have already voted for DMK.


Voting System
Results
Logout
Current Results
Committee
Votes
DMK
6
TVK
3


Voting System
Login
Register
Register
Name
Mobile Number
Password
Confirm Password
Role
Voter
Register

REFERENCES

1. **Elmasri, R., & Navathe, S. B.** (2016). *Fundamentals of Database Systems* (7th Edition). Pearson Education.Covers entity-relationship modeling and normalization concepts used in structuring the voting database.
2. **Flask Documentation** – https://flask.palletsprojects.com/ Official documentation for Flask, the Python web framework used for building backend server logic.
3. **MySQL Documentation** – https://dev.mysql.com/doc/ Useful for understanding how to store and retrieve votes securely in a relational database.
4. **Silberschatz, A., Korth, H. F., & Sudarshan, S.** (2010). *Database System Concepts* (6th Edition). McGraw-Hill Education. Provides foundational knowledge on database design, which is crucial for managing voter and poll data.

5. **W3Schools – HTML and CSS Reference** – https://www.w3schools.com/
   Resource for front-end development used to build the user interface for login, vote casting, and result viewing.