ONLINE VOTING SYSTEM

A PROJECT REPORT

Submitted by

GV SANTOSH (2303811710421137)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

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

SAMAYAPURAM – 621 112

NOVEMBER- 2024

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 **GV SANTOSH (2303811710421137)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING Dr.A.DELPHIN CAROLINA RANI, M.E.,Ph.D., HEAD OF THE DEPARTMENT PROFESSOR	CGB1201-JAVA PROGRAMMING Mr.MALARMANNAN A, M.E., SUPERVISOR ASSISTANTP ROFESSOR
SIGNATURE	**SIGNATURE**
Mrs.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mr. A. Malarmannan, M.E.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held on 06-12-2024

CGB1201-JAVA PROGRAMMING Mr. M. SARAVANAN, M.E., INTERNAL EXAMINER ASSISTANT PROFESSOR	CGB1201-JAVA PROGRAMMING Mr.R. KARTHIK, M.E., EXTERNAL EXAMINER ASSISTANT PROFESSOR 8138-SCE, TRICHY. 06/12/2024
INTERNAL EXAMINER	EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **“ONLINE VOTING SYSTEM”** is the result of original work done by us and best of our 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 **CGB1201 - JAVA PROGRAMMING.**

.

Signature

GV SANTOSH

Place: Samayapuram

Date: 06-12-2024

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. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **MR. A. MALARMANNAN, M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his 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.

VISION OF THE INSTITUTION

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

MISSION OF THE INSTITUTION

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

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

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

The Online Voting System is a software application designed to digitize and streamline the voting process for various types of elections, including governmental, organizational, and educational elections. This project aims to replace traditional paper-based voting with a secure, transparent, and efficient electronic system. By leveraging modern technologies, the system ensures accessibility for all eligible voters while maintaining the confidentiality and integrity of the votes cast.

The system provides users with an intuitive interface developed using **Java AWT**, allowing voters to register, log in, and cast their votes in a seamless manner. It supports real-time vote tallying and displays election-related information, ensuring transparency throughout the voting process. Key features include user authentication, candidate selection, secure vote storage, and real-time results display. These functionalities are designed to prevent fraudulent activities and provide a tamper-proof voting experience.

The project is structured into distinct modules: **User Registration**, **User Login**, **Election Management**, and **Vote Casting**. The **User Registration** module enables eligible voters to create accounts, while the **User Login** module secures access to the system using credentials. The **Election Management** module allows administrators to set up and manage elections, including candidate lists. The **Vote Casting** module ensures each voter can cast their vote securely and prevents multiple voting attempts.

The implementation follows a robust architecture with a clear separation of concerns, ensuring scalability and maintainability. The use of **data structures** like HashMap for user and vote storage ensures fast and efficient data handling. Additionally, the graphical interface enhances user interaction and accessibility, making it easier for individuals to participate in the voting process, irrespective of their technical expertise.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Online Voting System project is a secure, efficient, and user-friendly platform designed to enable electronic voting for various elections, such as governmental, organizational, or educational. This system leverages Java AWT for an interactive graphical interface, allowing users to register, log in, and cast votes seamlessly. It ensures accessibility for all eligible voters while maintaining the confidentiality and integrity of their votes through robust authentication and data handling mechanisms. By eliminating the need for manual voting processes, this system enhances transparency, reduces errors, and accelerates the tallying process, making elections more convenient and reliable.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	viii
1	**INTRODUCTION**	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
2	**PROJECT METHODOLOGY**	3
	2.1 Proposed Work	3
	2.2 Block Diagram	4
3	**MODULE DESCRIPTION**	5
	3.1 User Registration	5
	3.2 User Authentication	5
	3.3 Election Management	5
	3.4 Voting System	6
	3.5 Result Display	6
4	**CONCLUSION & FUTURE SCOPE**	7
	4.1 Conclusion	7
	4.2 Future Scope	7
	APPENDIX A (SOURCE CODE)	8
	APPENDIX B (SCREENSHOTS)	13
	REFERENCES	17

CHAPTER 1

INTRODUCTION

1.1Objective

- The objective of this Online Voting System is to create a digital platform that facilitates secure and efficient voting for various types of elections, such as governmental, organizational, and educational.
- The system aims to eliminate the drawbacks of traditional paper-based voting, including logistical challenges, time constraints, and risks of tampering or fraud.
- It seeks to provide a user-friendly interface for voters to cast their votes conveniently and confidentially from anywhere, ensuring inclusivity for all eligible voters, including those with physical disabilities or residing in remote locations.
- By leveraging modern technologies, the system strives to maintain the integrity of the election process through robust authentication, data encryption, and secure vote storage. Ultimately, the system aspires to enhance trust in the electoral process while promoting higher voter participation and ensuring accurate and transparent results.

1.2Overview

The Online Voting System is a robust and efficient platform designed to facilitate secure and transparent electronic voting processes for various types of elections, including governmental, organizational, and educational contexts. This system leverages Java's Abstract Window Toolkit (AWT) to provide an interactive and user-friendly graphical interface, ensuring a seamless experience for voters and administrators alike.

The platform encompasses essential functionalities such as user registration, authentication, election management, and vote casting, all implemented with a focus on data integrity and confidentiality.

It incorporates safeguards to prevent unauthorized access and double voting, thus maintaining the credibility of the election process. By digitizing voting operations, this system aims to enhance accessibility, eliminate logistical challenges associated with traditional voting methods, and deliver quick and accurate results.

1.3 Java Programming Concepts

1)Object-Oriented Programming (OOP): Encapsulation of logic within a class (OnlineVotingSystem).

2) Inheritance: Extends Frame class for GUI creation.

3) Event Handling: Implements ActionListener and handles actions like button clicks.

4) AWT Framework: Utilizes Frame, Panel, TextField, Button, Choice, etc., for GUI.

5) Collections Framework: Uses HashMap and HashSet for managing data like users and votes.

6) Control Statements: Employs if-else and loops for logic control.

7) Exception Handling: Validates user input to avoid runtime errors.

8) String Manipulation: Processes text input from the user.

9) CardLayout: Manages switching between different UI panels.

10) Polymorphism: Overrides actionPerformed for event handling.

11) Abstraction: Hides implementation details of vote recording and user validation.

12) Encapsulation: Groups related data (voter info, election type) into cohesive methods.

13) Dynamic Binding: Calls methods (addActionListener) at runtime based on events.

14) Data Structures: Organizes and processes vote counts and voter records.

15) Thread Safety: Maintains isolated state for voters within the GUI flow.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work

1. Requirement Analysis

- Identify the stakeholders (voters, administrators, candidates).
- Define the types of elections (governmental, organizational, or educational).
- Establish the system requirements for security, accessibility, and functionality.

2. System Design

- **Architecture Design:** Design a modular system with separate components for registration, login, election type selection, and voting.
- **Database Structure:** Use HashMap and HashSet for managing user details, voter IDs, and vote counts efficiently.
- **User Interface (UI):** Develop a user-friendly GUI using AWT with clear navigation between panels.

3. Implementation

- **User Registration:** Allow new users to register with unique voter IDs and credentials.
- **Login and Authentication:** Implement secure login using voter ID, username, and password validation.
- **Election Type Selection:** Enable users to select the type of election they wish to participate in.
- **Voting Mechanism:**
  - Restrict voting to eligible voters.
  - Prevent multiple votes from the same voter ID in the same election.
- **Results Display:** Show live voting results for transparency while ensuring voter anonymity.

4. Security Measures

- Validate user input to prevent incorrect data entry.
- Ensure that each voter ID can vote only once per election.
- Maintain confidentiality of user credentials and votes.

5. Testing and Validation

- **Unit Testing:** Test individual modules such as registration, login, and voting.
- **Integration Testing:** Ensure smooth interaction between panels and data structures.
- **User Testing:** Collect feedback from a small group of users to improve usability.

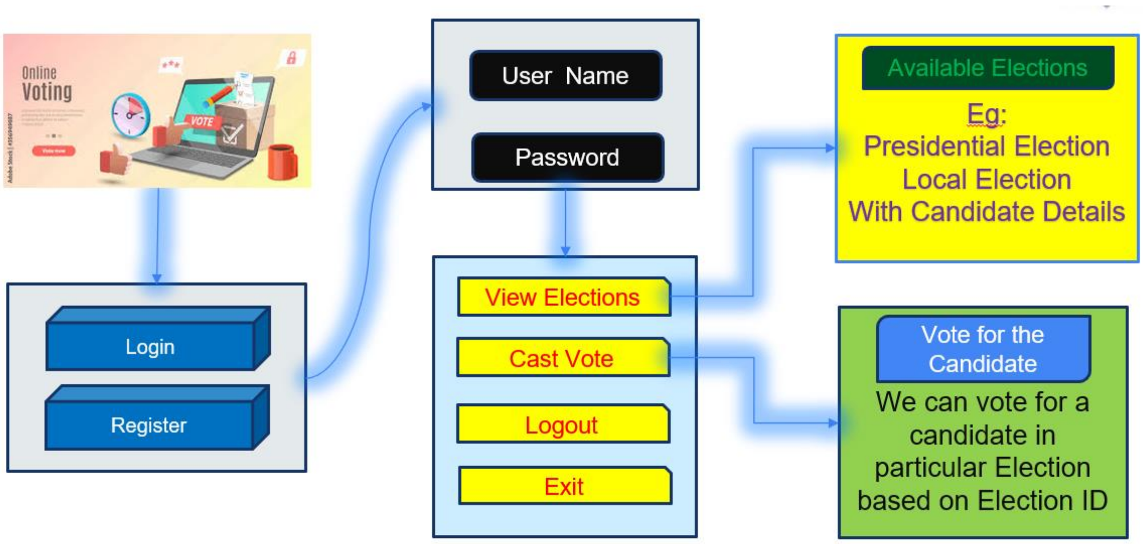
6. Deployment and Usage

- **Unit Testing:** Test individual modules such as registration, login, and voting.
- **Integration Testing:** Ensure smooth interaction between panels and data structures.
- **User Testing:** Collect feedback from a small group of users to improve usability.

7. Maintenance and Future Enhancements

- **Unit Testing:** Test individual modules such as registration, login, and voting.
- **Integration Testing:** Ensure smooth interaction between panels and data structures.
- **User Testing:** Collect feedback from a small group of users to improve usability.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Module 1 User Registration

This module handles the registration of voters to ensure only eligible participants can access the voting system.

- **Features:**
  - Collects voter ID, username, and password during registration.
  - Validates inputs to ensure uniqueness of voter IDs and usernames.
  - Stores user details securely for authentication purposes.
- **Objective:** To enable users to register securely and create their credentials.

3.2 Module 2 User Authentication

This module verifies the identity of users before granting access to the voting system.

- **Features:**
  - Validates voter ID, username, and password against the stored data.
  - Displays error messages for invalid credentials.
  - Redirects authenticated users to the election selection interface.
- **Objective:** To ensure secure access to the voting system and prevent unauthorized access.

3.3 Module 3 Election Management

This module enables users to select the type of election they wish to participate in.

- **Features:**
  - Provides options for different election types: Governmental, Organizational, and Educational.
  - Dynamically updates voting options and data based on the selected election type.
- **Objective:** To support multiple election types in a single system, catering to various scenarios.

3.4 Module 4 Voting System

This is the core module responsible for managing the voting process.

- **Features:**
    - Displays candidates dynamically based on the selected election type.
    - Ensures each voter ID can vote only once per election.
    - Updates the vote count securely for the selected candidate.
    - Prevents tampering or multiple voting attempts.
- **Objective:** To allow users to cast their votes efficiently while ensuring fairness and security.

3.5 Module 5 Result Display

This module displays the results of the election in real-time.

- **Features:**
    - Shows vote counts for each candidate in the selected election type.
    - Updates the result display dynamically as votes are cast.
    - Ensures voter anonymity by not revealing individual votes.
- **Objective:** To provide transparency in the voting process while maintaining confidentiality.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The **Online Voting System** project successfully demonstrates the development of a secure, user-friendly, and efficient electronic voting platform. By incorporating modules for **User Registration**, **Authentication**, **Election Management**, **Voting**, and **Result Display**, the system ensures that elections are conducted transparently while maintaining the confidentiality and integrity of voter data.

The system caters to various types of elections, such as governmental, organizational, and educational, making it versatile and applicable in diverse scenarios. Features like **voter ID validation**, **one vote per election**, and **real-time result display** enhance its reliability and trustworthiness.

This project leverages fundamental Java programming concepts, such as object-oriented design, AWT-based GUI, and secure data handling, to provide a robust and scalable solution. It addresses challenges in traditional voting systems, such as accessibility, efficiency, and security, making it a significant step towards modernizing election processes.

4.2 FUTURE SCOPE

The **Online Voting System** has significant potential for future expansion and improvement. One of the key areas of growth is scalability, where the system can be enhanced to handle large-scale elections by integrating robust databases like MySQL or MongoDB, enabling it to manage millions of voters efficiently. The system can also be expanded for multi-platform accessibility, with web and mobile applications, ensuring compatibility across various devices and operating systems. Security can be further strengthened by incorporating encryption techniques and multi-factor authentication, alongside using blockchain technology for transparency and tamper-proof voting records. Additionally, the system can be made more inclusive by supporting multiple languages and adding accessibility features for differently-abled individuals. Future versions can offer support for more election types, with customizable options to meet the needs of different organizations. Real-time analytics and monitoring features would allow election administrators to track voter turnout and voting patterns, ensuring the system's effectiveness and security.

APPENDIX A

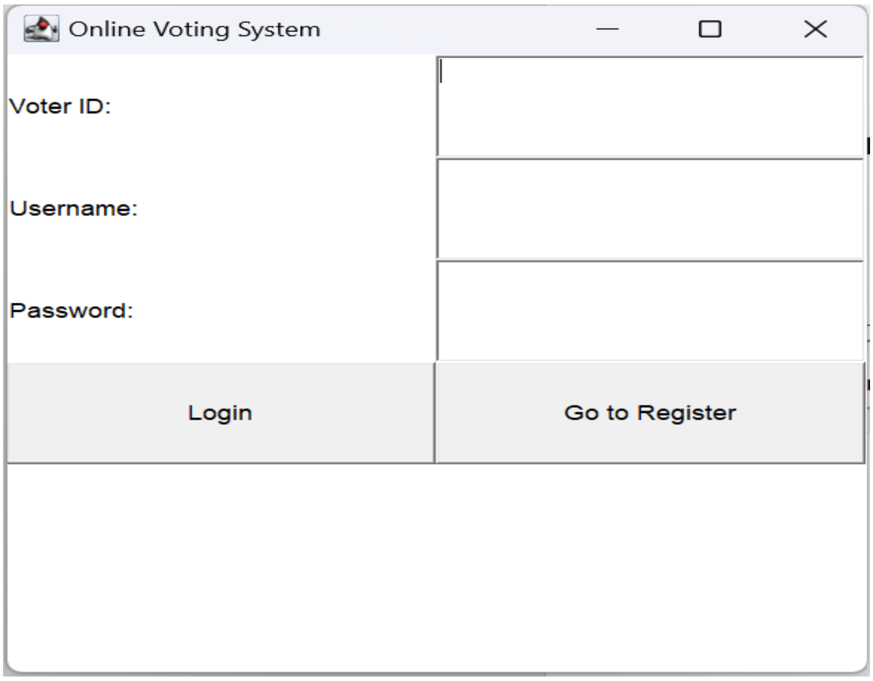
(SOURCE CODE)

```
import java.awt.*;
import java.awt.event.*;
import java.util.HashMap;
import java.util.HashSet;

public class OnlineVotingSystem extends Frame implements ActionListener {
    private CardLayout cardLayout;

    private Panel loginPanel, registerPanel, votingPanel, electionTypePanel;
    private TextField voterIdField, usernameField, passwordField;
    private TextArea resultsArea;
    private Label statusLabel;
    private Choice candidateDropdown, electionTypeDropdown;

    // Data structures to hold user data and vote counts
    private HashMap<String, String> userDatabase; // Maps username to password
    private HashMap<String, String> voterIds; // Maps username to voter ID
    private HashMap<String, HashMap<String, Integer>> electionVoteCounts; // Election
type -> (Candidate -> Vote Count)
    private HashMap<String, HashSet<String>> electionVoters; // Election type -> Set of
voter IDs who have voted

    private String currentVoterId;

    public OnlineVotingSystem() {
        // Initialize user data storage
        userDatabase = new HashMap<>();
        voterIds = new HashMap<>();
        electionVoteCounts = new HashMap<>();
        electionVoters = new HashMap<>();

        // Initialize vote counts and voter records for each election type
        for (String electionType : new String[]{"Governmental", "Organizational",
"Educational"}) {
            HashMap<String, Integer> voteCount = new HashMap<>();
            voteCount.put("Candidate A", 0);
            voteCount.put("Candidate B", 0);
            voteCount.put("Candidate C", 0);
            electionVoteCounts.put(electionType, voteCount);
            electionVoters.put(electionType, new HashSet<>());
        }

        // Set up the frame and layout
        setSize(400, 400);
        setTitle("Online Voting System");
        cardLayout = new CardLayout();
        setLayout(cardLayout);
        // Initialize panels
        createLoginPanel();
        createRegisterPanel();
        createElectionTypePanel();
        createVotingPanel();

        // Add panels to the card layout
        add(loginPanel, "Login");
        add(registerPanel, "Register");
        add(electionTypePanel, "ElectionType");
        add(votingPanel, "Voting");

        // Show the login panel by default
        showLoginPanel();

        // Close operation
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent we) {
                System.exit(0);
            }
        });
    }

    private void createLoginPanel() {
        loginPanel = new Panel(new GridLayout(6, 2));
        loginPanel.add(new Label("Voter ID:"));
        voterIdField = new TextField();
        loginPanel.add(voterIdField);
        loginPanel.add(new Label("Username:"));
        usernameField = new TextField();
        loginPanel.add(usernameField);
        loginPanel.add(new Label("Password:"));
        passwordField = new TextField();
        passwordField.setEchoChar('*');
        loginPanel.add(passwordField);

        Button loginButton = new Button("Login");
        loginButton.addActionListener(this);
        loginPanel.add(loginButton);

        Button registerButton = new Button("Go to Register");
        registerButton.addActionListener(e -> showRegisterPanel());
        loginPanel.add(registerButton);

        statusLabel = new Label("");
        loginPanel.add(statusLabel);
    }

    private void createRegisterPanel() {
        registerPanel = new Panel(new GridLayout(5, 2));
        TextField regVoterIdField = new TextField();
        TextField regUsernameField = new TextField();
        TextField regPasswordField = new TextField();
        regPasswordField.setEchoChar('*');

        registerPanel.add(new Label("Voter ID:"));
        registerPanel.add(regVoterIdField);
        registerPanel.add(new Label("Username:"));
        registerPanel.add(regUsernameField);
        registerPanel.add(new Label("Password:"));
        registerPanel.add(regPasswordField);

        Button submitRegisterButton = new Button("Register");
        submitRegisterButton.addActionListener(e -> {
            registerUser(regVoterIdField.getText(), regUsernameField.getText(),
regPasswordField.getText());
        });
        registerPanel.add(submitRegisterButton);

        Button goToLoginButton = new Button("Already Registered? Go to Login");
        goToLoginButton.addActionListener(e -> showLoginPanel());
        registerPanel.add(goToLoginButton);
    }

    private void createElectionTypePanel() {
        electionTypePanel = new Panel(new GridLayout(3, 1));
        electionTypeDropdown = new Choice();
        electionTypeDropdown.add("Governmental");
        electionTypeDropdown.add("Organizational");
        electionTypeDropdown.add("Educational");
        electionTypePanel.add(new Label("Select the type of election:"));
        electionTypePanel.add(electionTypeDropdown);

        Button proceedButton = new Button("Proceed to Voting");
        proceedButton.addActionListener(e -> showVotingPanel());
        electionTypePanel.add(proceedButton);
    }

    private void createVotingPanel() {
        votingPanel = new Panel(new BorderLayout());
        votingPanel.add(new Label("Vote for Your Candidate"), BorderLayout.NORTH);

        // Panel for candidate selection
        Panel voteInputPanel = new Panel(new FlowLayout());
        candidateDropdown = new Choice();
        candidateDropdown.add("Candidate A");
        candidateDropdown.add("Candidate B");
        candidateDropdown.add("Candidate C");
        voteInputPanel.add(new Label("Select Candidate:"));
        voteInputPanel.add(candidateDropdown);

        Button voteButton = new Button("Vote");
        voteButton.addActionListener(e ->
submitVote(candidateDropdown.getSelectedItem()));
        voteInputPanel.add(voteButton);

        votingPanel.add(voteInputPanel, BorderLayout.CENTER);

        // Results display area
        resultsArea = new TextArea(10, 30);
        resultsArea.setEditable(false);
        votingPanel.add(resultsArea, BorderLayout.SOUTH);
    }

    private void showLoginPanel() {
        cardLayout.show(this, "Login");
        clearFields();
        statusLabel.setText("");
    }

    private void showRegisterPanel() {
        cardLayout.show(this, "Register");
    }

    private void showElectionTypePanel() {
        cardLayout.show(this, "ElectionType");
    }

    private void showVotingPanel() {
        cardLayout.show(this, "Voting");
        updateResults();
    }

    private void clearFields() {
        voterIdField.setText("");
        usernameField.setText("");
        passwordField.setText("");
    }

    private void registerUser(String voterId, String username, String password) {
        if (voterId.isEmpty() || username.isEmpty() || password.isEmpty()) {
            statusLabel.setText("All fields are required!");
            return;
        }
        if (userDatabase.containsKey(username)) {
            statusLabel.setText("Username already exists!");
            return;
        }
        userDatabase.put(username, password);
        voterIds.put(username, voterId);
        statusLabel.setText("Registration successful!");
    }

    private void submitVote(String candidate) {
        String electionType = electionTypeDropdown.getSelectedItem();

        // Check if the voter has already voted in this election
        if (electionVoters.get(electionType).contains(currentVoterId)) {
            resultsArea.setText("You have already voted in this election!");
            return;
        }

        // Record the vote
        HashMap<String, Integer> voteCount = electionVoteCounts.get(electionType);
        voteCount.put(candidate, voteCount.get(candidate) + 1);

        // Mark the voter as having voted
        electionVoters.get(electionType).add(currentVoterId);

        resultsArea.setText("Vote cast successfully!\n");
        updateResults();
    }
    private void updateResults() {
        String electionType = electionTypeDropdown.getSelectedItem();
        HashMap<String, Integer> voteCount = electionVoteCounts.get(electionType);
        resultsArea.append("Current Results for " + electionType + " Election:\n");
        voteCount.forEach((c, count) -> resultsArea.append(c + ": " + count + "\n"));
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        String voterId = voterIdField.getText();
        String username = usernameField.getText();
        String password = passwordField.getText();

        if (!voterIds.containsKey(username) || !userDatabase.get(username).equals(password) ||
!voterIds.get(username).equals(voterId)) {
            statusLabel.setText("Invalid credentials!");
            return;
        }
        currentVoterId = voterId; // Save current voter ID
        statusLabel.setText("Login successful!");
        showElectionTypePanel();
    }

    public static void main(String[] args) {
        new OnlineVotingSystem().setVisible(true);
    }
}
```

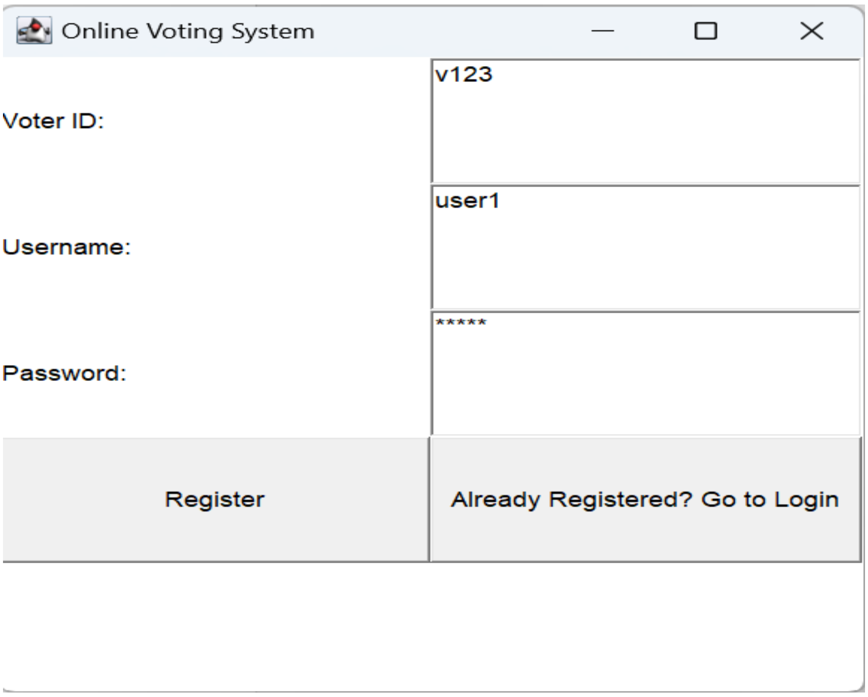
APPENDIX B

(SCREENSHOTS)

1) OUTPUT INTERFACE



2) REGISTER PAGE

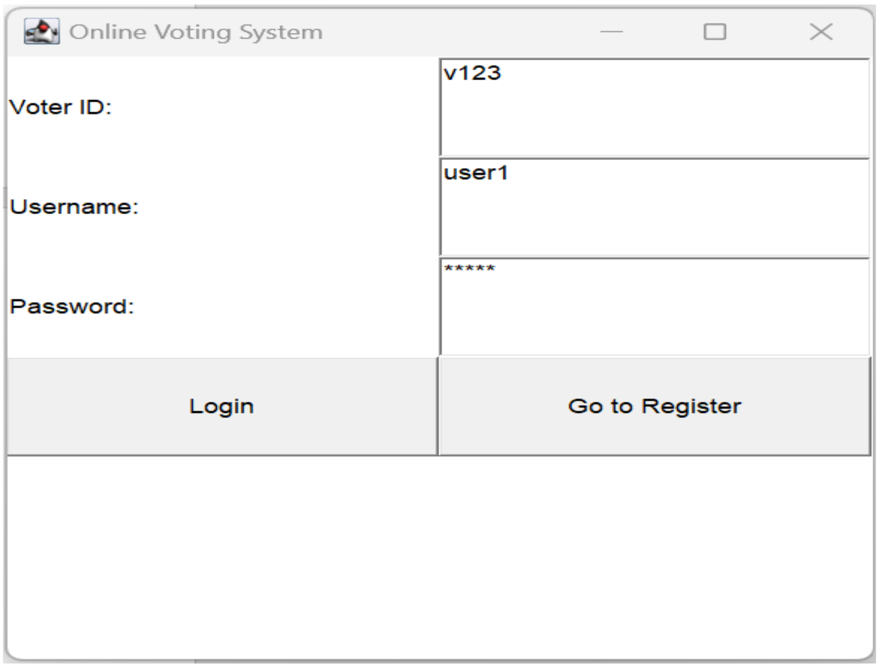


Voter ID: v123

Username: user1

Password: *****

3) LOGIN PAGE



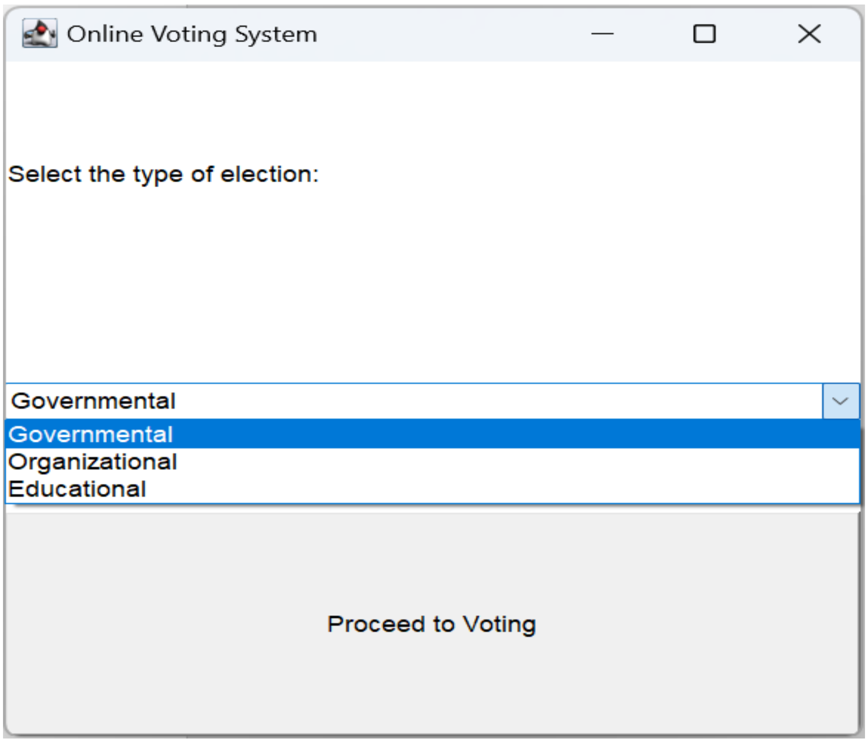
Online Voting System

Voter ID: v123

Username: user1

Password: *****

4) SELECTING THE TYPE OF ELECTION

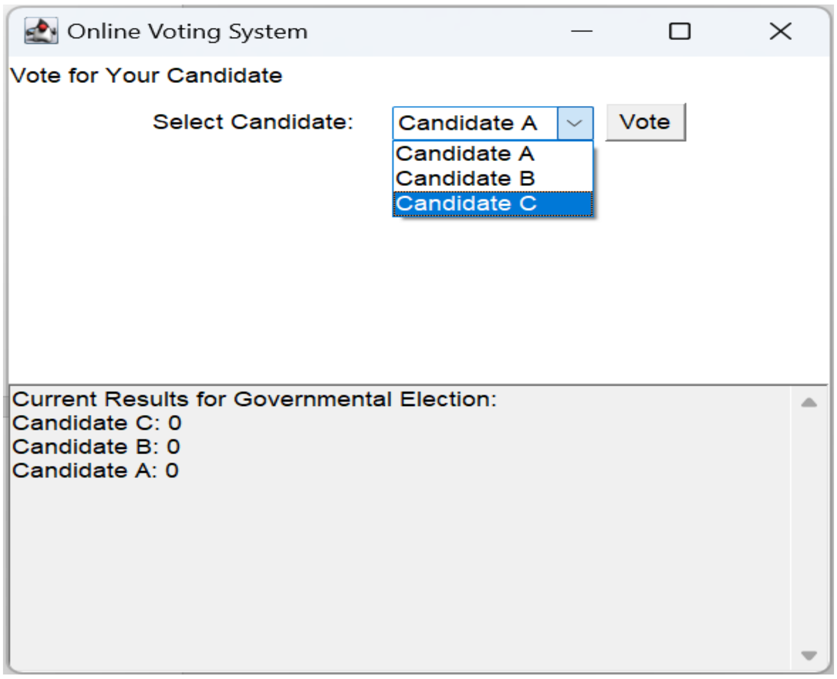


Online Voting System

Select the type of election:

Governmental

5) VOTE FOR YOUR CANDIDATE

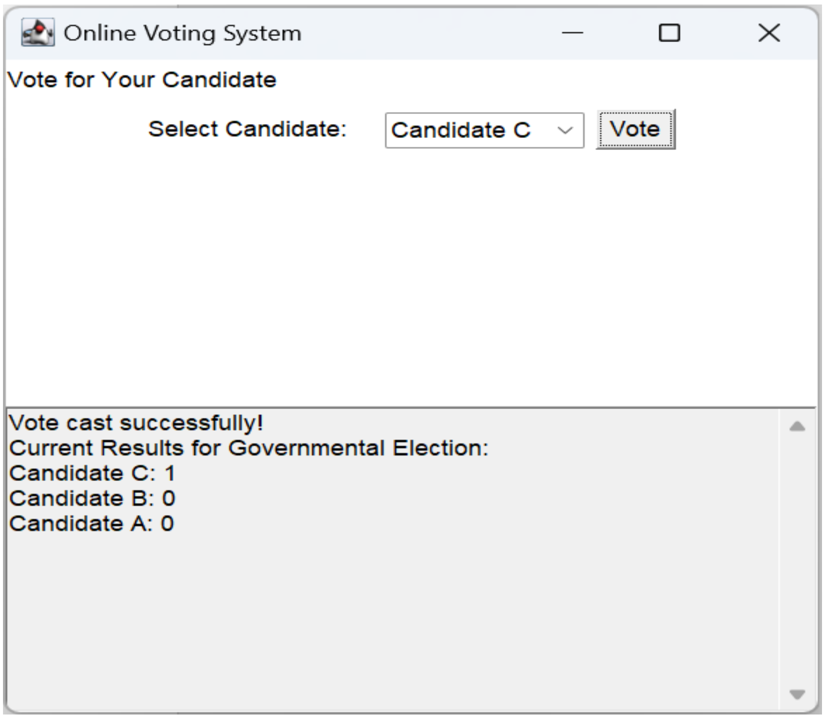


Vote for Your Candidate

Select Candidate: Candidate A

```
Current Results for Governmental Election:
Candidate C: 0
Candidate B: 0
Candidate A: 0
```

6) RESULT

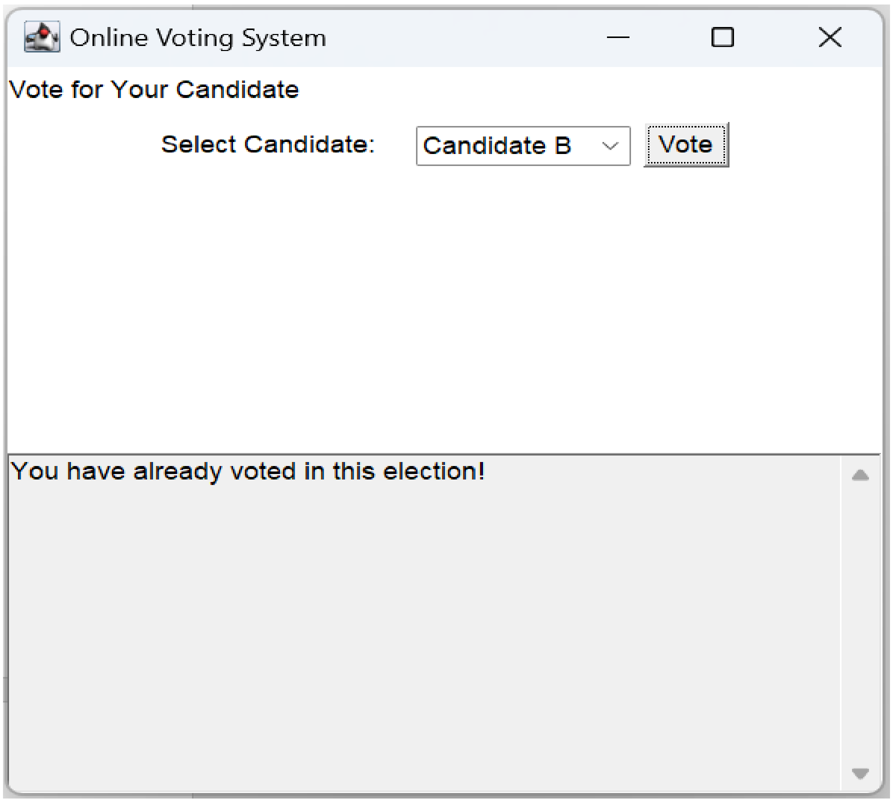


Vote for Your Candidate

Select Candidate: Candidate C

```
Vote cast successfully!
Current Results for Governmental Election:
Candidate C: 1
Candidate B: 0
Candidate A: 0
```

7) RETRYING TO VOTE



You have already voted in this election!

REFERENCES

1. Sharma, Rahul, and Suresh Kumar. "Design and Implementation of Online Voting System Using Java and MySQL." International Journal of Computer Science and Information Technologies (IJCSIT), vol. 11, no. 4, 2023. **Design of Secure Online Voting Systems Using Java and MySQL**.
2. Gupta, Richa, and Amit Sharma. "E-Voting System: Secure and Scalable Web-Based Election Solution." International Journal of Engineering and Technology (IJET), vol. 7, no. 5, 2021. **E-Voting System Implementation and Security Considerations**.
3. Patel, Kiran, et al. "A Study of E-Voting Systems and Their Development for Transparent and Secure Elections." Journal of Information Security and Applications, vol. 28, 2020. **Secure Online Voting Systems for Digital Elections**.