Report

on

“Online Voting System”

SUBMITTED BY

Prachi Rajendra Shedage
Swapnil Keshav Supekar

PROJECT GUIDE

Miss. Kajal Moghe

DEPARTMENT OF COMPUTER ENGINEERNG

D. Y. Patil Academic Education Excellence Federation’s

Dr. D. Y. PATIL TECHNICAL CAMPUS (Engg. & MCA)
VARALE-TALEGAON, PUNE–410507

Academic Year 2024-2025

DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

This is to certify that **Prachi Rajendra Shedage** and **Swapnil Keshav Supekar** of Computer Science of Engineering has successfully completed the Project titled **"Online voting system"** towards the partial fulfillment for the requirements of the Bachelor Degree in Computer Engineering course under the University of Pune during Academic year **2024-2025.**

Project Based Learning Guide
Miss. Kajal Moghe

Head of Department
Dr. S. P. Kosbatwar

Acknowledgement

I express my sincere gratitude towards the faculty members who made this project successful. I would like to express my thanks to my guide **Miss. Kajal Moghe** for his whole hearted co-operation and valuable suggestions, technical guidance throughout the seminar work.

Special thanks to Head of Department and Principal of Computer Science Engineering **Dr. S. P. Kosbatwar and Dr. S. D. Shirbahadurkar** for his kind official support and encouragement. Finally, I would like to thank to all my staff members of **Computer Science of Engineering** department who helped me directly or indirectly to complete this work successfully.

Student Name :
Prachi Rajendra Shedage
Swapnil Keshav Supekar
Department Name:
Computer Science of Engineering Batch 2024-25

TABLE OF CONTENTS

ABSTRACT

The **Online Voting System** is a Java-based web application designed to conduct elections securely and efficiently over the internet. It eliminates the need for traditional paper-based voting by allowing registered users to cast their votes online, ensuring a fast, reliable, and tamper-proof process.

This system is developed using Java (Servlets/JSP or JavaFX) for the front end and **MySQL** as the back-end database, with **JDBC** for connectivity. The core modules include Admin Management, Voter Registration, Secure Login, Voting Process, and Result Generation.

The admin can create elections, manage candidates, monitor votes, and view real-time results. Voters must register with valid credentials, and once approved, they can log in to securely cast their vote. The system ensures that each voter can vote only once and stores all votes in an encrypted format. Candidate lists are dynamically displayed, and results are generated instantly after voting ends.
The database consists of tables like voters, candidates, votes, and elections to handle all necessary records.

This system enhances election transparency, reduces human errors, and accelerates result processing. It is particularly useful for colleges, institutions, societies, or organizations to conduct internal elections. The platform supports secure authentication, real-time reporting, and remote accessibility, making it a practical solution for modern voting needs. Scalable and easy to use, the Online Voting System ensures integrity, convenience, and speed in the voting process, benefiting both voters and administrators alike

CHAPTER I

INTRODUCTION

The management of an election process involves several critical tasks, including voter registration, candidate management, vote casting, and result generation. Traditionally, these processes have been conducted manually or through paper-based systems, which are time-consuming, error-prone, and vulnerable to fraud. To address these challenges, the **Online Voting System** is designed as an automated and secure solution to streamline the voting process, enhance transparency, and ensure accurate results.

This system is developed using Java, a platform-independent and robust programming language, and employs MySQL as the backend database for storing voter, candidate, and election data securely. Java Database Connectivity (JDBC) facilitates seamless interaction between the application and the database, ensuring reliable data operations.

The system includes several core modules. The **Voter Registration** module allows new users to sign up with valid identification, and their records are securely stored and verified before allowing them to vote. The **Admin Module** enables election officers to manage elections, add candidates, activate or close voting sessions, and oversee the entire process. During the **Voting Phase**, authenticated voters can securely log in, view the list of candidates, and cast their vote—restricted to one per election. Votes are stored securely and counted automatically.

The **Result Generation** module displays real-time election results and generates reports for transparency and auditing. By automating these processes, the system significantly reduces manual effort, eliminates vote duplication, prevents tampering, and enhances trust in the electoral system.
Its user-friendly interface makes it accessible to both technical and non-technical users. Scalable and efficient, the Online Voting System is an ideal solution for institutions, colleges, organizations, or communities looking to conduct secure and efficient electronic voting.

CHAPTER II: BACKGROUND AND LITERATURE REVIEW

Background

Inventory and billing systems in pharmacies have undergone significant transformation due to the integration of software automation and relational databases. Traditional manual systems have been replaced with advanced, database-driven applications to streamline operations such as medicine tracking, sales processing, and reporting. These systems are now more reliable, accurate, and scalable, helping pharmacies to improve efficiency and reduce human errors.

Researchers and developers have proposed several solutions aimed at automating medical store operations. For instance, M. Deshmukh and A. Gupta developed a Java-based drug store management system using MySQL, which efficiently handles stock and sales transactions. Similarly, A. Patel and R. Shah proposed a web-based pharmacy management system that focuses on real-time billing and inventory control. Dr. S. Kumar's work highlighted the importance of managing expired medicines and optimizing inventory turnover through structured data handling techniques. Another significant contribution by T. Roy and N. Sharma explored a centralized hospital and pharmacy automation system, emphasizing comprehensive database integration.

These studies provide valuable insights and form the foundation for the development of the current Medical Store Management System, which aims to incorporate key features from previous works while enhancing usability and system scalability.

Literature Review

Paper No.	Title	Author(s)	Outcomes	Future Scope	Remarks
1	Pharmacy Management System	A. Patel, R. Shah	Designed a web-based system to manage pharmacy inventory and billing.	Integration with mobile apps and real-time notifications.	Focused on web, not console-based systems.
2	Inventory Management in Healthcare	Dr. S. Kumar	Improved medicine tracking and reduced expired stock issues.	Can add predictive analytics for demand forecasting.	Strong emphasis on expiry date management.
3	Design of Drug Store System Using Java	M. Deshmukh, A. Gupta	Java-based system with MySQL for drug sales and stock management.	Expand to support multiple store chains.	Very similar to current project; useful as reference.

Paper No.	Title	Author(s)	Outcomes	Future Scope	Remarks
4	A Study on Hospital Automation	T. Roy, N. Sharma	Centralized hospital and pharmacy system with full database linkage.	Cloud integration for remote access and backups.	More focused on hospital-wide systems.

CHAPTER III: AIM AND OBJECTIVES

Aim :

The primary aim of the **Medical Store Management System** is to design and develop a reliable, efficient, and user-friendly software application that automates the day-to-day operations of a medical store. This system is intended to enhance the accuracy, speed, and ease of managing critical activities such as inventory control, billing, purchases, and customer data handling.

The system is built using Java for the application layer and MySQL for the database, with JDBC facilitating seamless and secure communication between the two. It is specifically targeted at small to medium-sized medical stores to provide a cost-effective and scalable solution for modernizing their operational workflows.

Key Goals:

- Automate inventory management by maintaining real-time records of medicines, including quantity, expiry dates, and pricing.
- Enable fast and accurate billing during sales transactions through automated invoice generation.
- Provide timely alerts for low stock levels and expired medicines to ensure safety and continuous availability.
- Store detailed customer and supplier records for organized data management and better service delivery.
- Generate analytical reports on sales, purchases, and inventory status to support data-driven decision-making.
- Offer an intuitive, user-friendly interface that can be easily operated by non-technical users.
- Ensure secure, consistent, and efficient data operations using Java and JDBC with a robust backend.
- Deliver a scalable, cost-effective solution tailored for pharmacies aiming to digitize their workflow.

Objectives:

- To **automate inventory management** by accurately tracking medicine details such as name, quantity, expiry date, and pricing.
- To **simplify the sales process** by generating quick and precise bills with itemized invoices for customer purchases.
- To **issue real-time alerts** for low-stock and expired medicines, enabling timely replenishment and preventing unsafe sales.
- To **maintain organized records** of customers and suppliers for improved service and efficient purchase management.
- To **generate detailed reports** on inventory, sales, and purchase data, assisting store owners in making strategic business decisions.

CHAPTER IV: METHODOLOGY

1. System Overview

The **Online Voting System** is a Java-based console application integrated with a MySQL database, designed to facilitate secure and efficient electronic voting. It automates key election processes such as voter registration, candidate management, vote casting, and result generation. This system eliminates the need for manual or paper-based voting, ensuring better accuracy, speed, and transparency. Built with a user-friendly console interface, the system ensures that even users with limited technical skills can participate in the voting process. The application provides secure login for voters, prevents multiple voting, and allows the administrator to manage elections and view real-time results. It is suitable for educational institutions, organizations, or local communities seeking to conduct secure and paperless elections.

2. System Architecture and Connections

a. Java Console Program to Database (JDBC)

The console-based Java application uses **JDBC (Java Database Connectivity)** to establish a connection with the MySQL database. This connection enables the execution of SQL queries such as `INSERT`, `SELECT`, `UPDATE`, and `DELETE` to manage voter records, candidate lists, election sessions, and voting results.

b. User Input to Program Logic

Users interact with the system by selecting options from the console-based menu. Voters can register, log in, and cast their vote, while the admin can manage candidates and view results. The program captures these inputs, processes them through logical modules, and performs the corresponding database operations.

c. Database Table Relationships

The MySQL database contains several interrelated tables that store and manage critical information:

- `Voters`: Stores registered voter details and login credentials.
- `Candidates`: Contains information about election candidates.
- `Votes`: Records each vote, linking voter IDs with candidate IDs.
- `Elections`: Manages the creation, status, and schedule of elections.

d. Module Interaction Within Code

The Java application is structured into modular classes, each responsible for a specific function:

- `VoterRegistration`: Handles new voter sign-ups and validation.
- `Login`: Manages voter and admin authentication.
- `Voting`: Allows authenticated users to cast a vote.
- `AdminPanel`: Provides options for election and candidate management.
- `ResultGenerator`: Retrieves and displays voting results

e. Output and Feedback to Console

All system interactions and operations provide immediate console output. Whether it's a successful registration, an error message during login, or the final vote tally, the system offers clear and real-time feedback. This ensures transparency and helps users understand the outcome of their actions.

4. Diagram :

1. Use Case Diagram

The **Use Case Diagram** for the **Online Voting System** illustrates the key interactions between the system and its primary users using UML notation. The diagram identifies three main actors: **Administrator**, **Voter**, and **System**. Each actor is associated with specific functionalities that represent their roles and permissions within the system.

- The **Administrator** can perform tasks such as managing elections, adding or removing candidates, viewing voting statistics, and publishing results.
- The **Voter** is responsible for registering on the system, logging in securely, and casting a vote for their preferred candidate during an active election.
- The **System** automates tasks such as validating votes, preventing multiple voting, storing data, and generating final results.

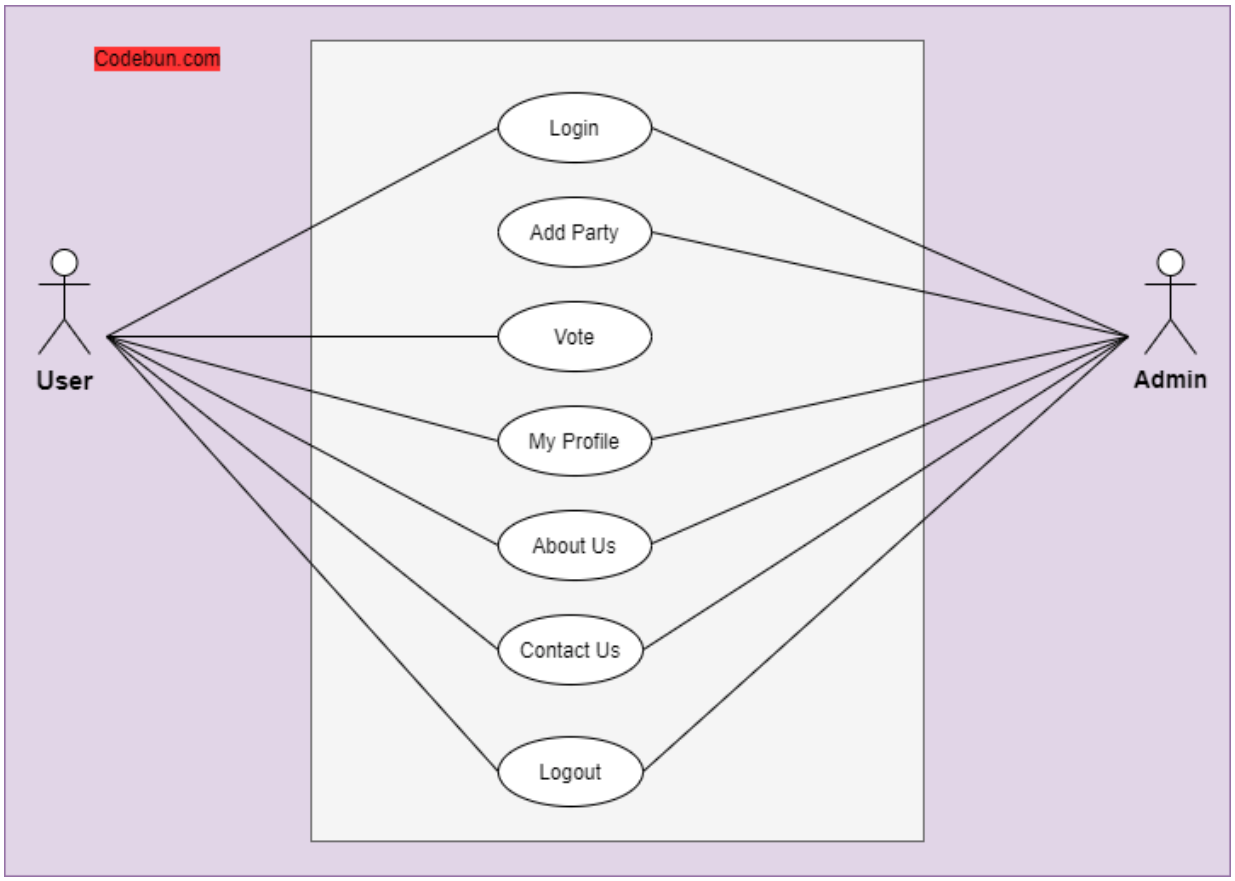


Fig 1 Use Case Diagram

2. Class Diagram

The **Online Voting System** is structured to efficiently manage the key components of an electronic voting platform, including user authentication, candidate management, vote casting, and result calculation. The system uses object-oriented design principles to organize core functionalities into distinct classes.

Key classes include:

- **User**: A base class that holds common attributes like user ID, name, and login credentials. It serves as a parent class for `Voter` and `Admin`.
- **Voter**: Inherits from `User`, and contains additional attributes like voter status (voted/not voted). It includes methods for registration, login, and casting a vote.
- **Admin**: Inherits from `User`, with privileges to create elections, manage candidates, and view results. It includes methods to activate/deactivate elections and generate reports.
- **Candidate**: Stores candidate information such as candidate ID, name, party, and election ID. This class is linked to the `Election` class.
- **Election**: Manages election-related data like election ID, title, status (active/inactive), and schedule. It connects to both `Candidate` and `Vote` classes.
- **Vote**: Represents a voting transaction, recording which voter cast a vote for which candidate in a specific election. This class ensures that a voter can only vote once per election

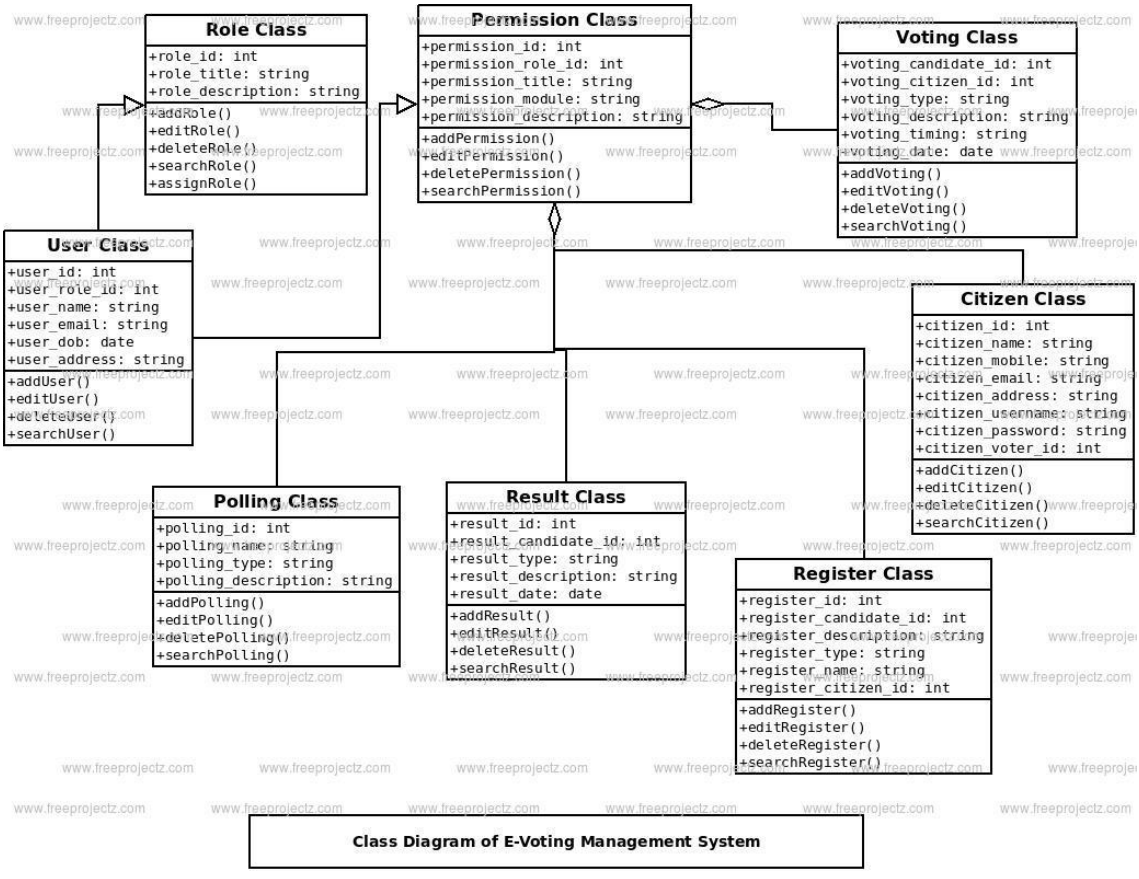


Fig 2 Class Diagram

3. Sequence Diagram

The **Sequence Diagram** for the **Online Voting System** illustrates the step-by-step interaction between a **voter** and the **system** during the **voting process**. It outlines how objects such as the voter, login module, election module, and vote handler interact over time to complete a secure and valid vote submission.

The process begins when the **voter logs in** to the system using their credentials. The **Login Module** validates the credentials against the database. Upon successful authentication, the system checks whether the voter has already voted. If not, the **Election Module** retrieves the list of active elections and associated candidates. The voter selects their preferred candidate, and the **Vote Handler** records the vote in the database while updating the voter's status to prevent duplicate voting. Finally, a confirmation message is displayed to the voter.

This interaction ensures secure authentication, accurate vote recording, and real-time feedback, maintaining the integrity and transparency of the election process.

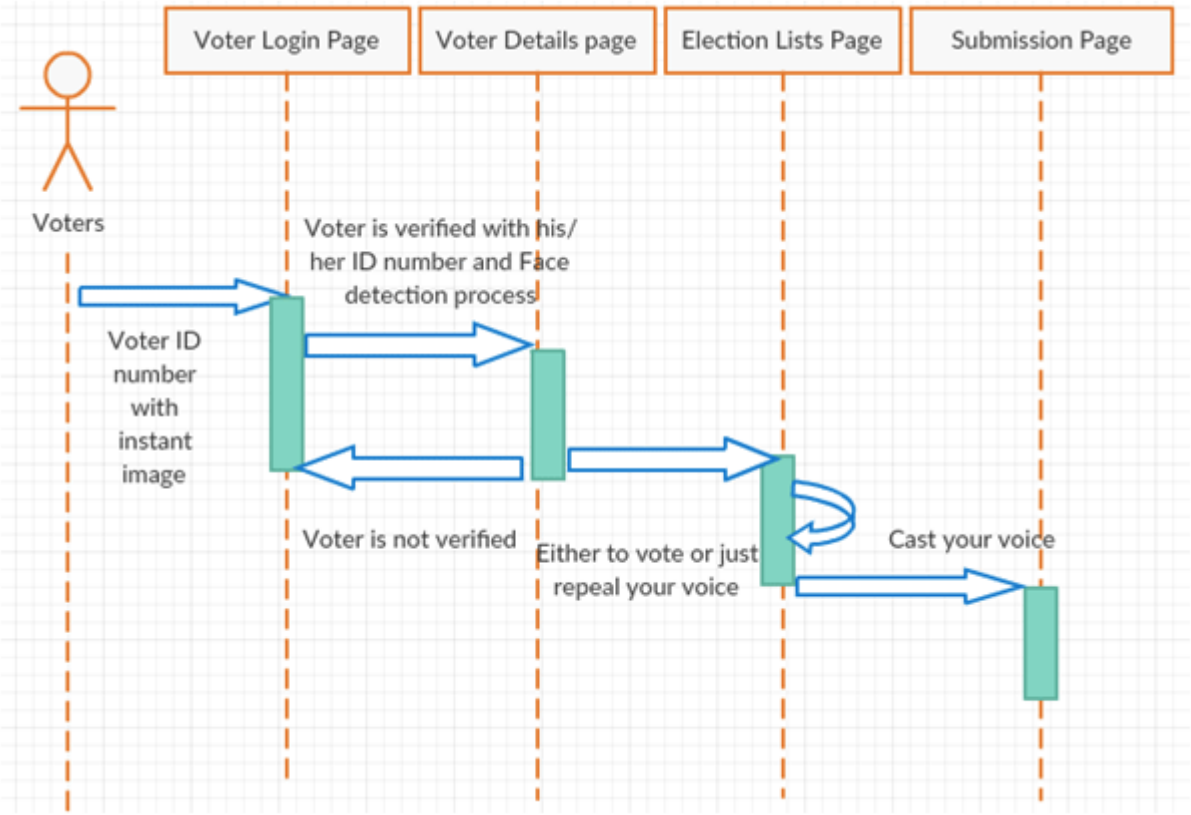


Fig 3 Sequence Diagram

4. Activity Diagram

The **Activity Diagram** for the **Online Voting System** illustrates the logical flow of activities in a console-based Java application connected to a MySQL database. The process begins with **establishing a database connection**, followed by **user authentication**, where the user may log in either as an **admin** or a **voter**.

If the user is a **voter**, the system checks whether they have already voted. If not, the system displays the list of active elections and candidates. The voter selects a candidate and submits their vote, after which the vote is recorded and a confirmation message is shown. The process then ends.

If the user logs in as an **admin**, the system displays administrative options such as managing elections, adding or removing candidates, and viewing results. The admin can perform any of these operations before choosing to exit the system.

This activity diagram clearly represents the workflow of both voter and admin interactions, emphasizing conditional logic, role-based access, and database-driven decision-making in a secure and structured manner.

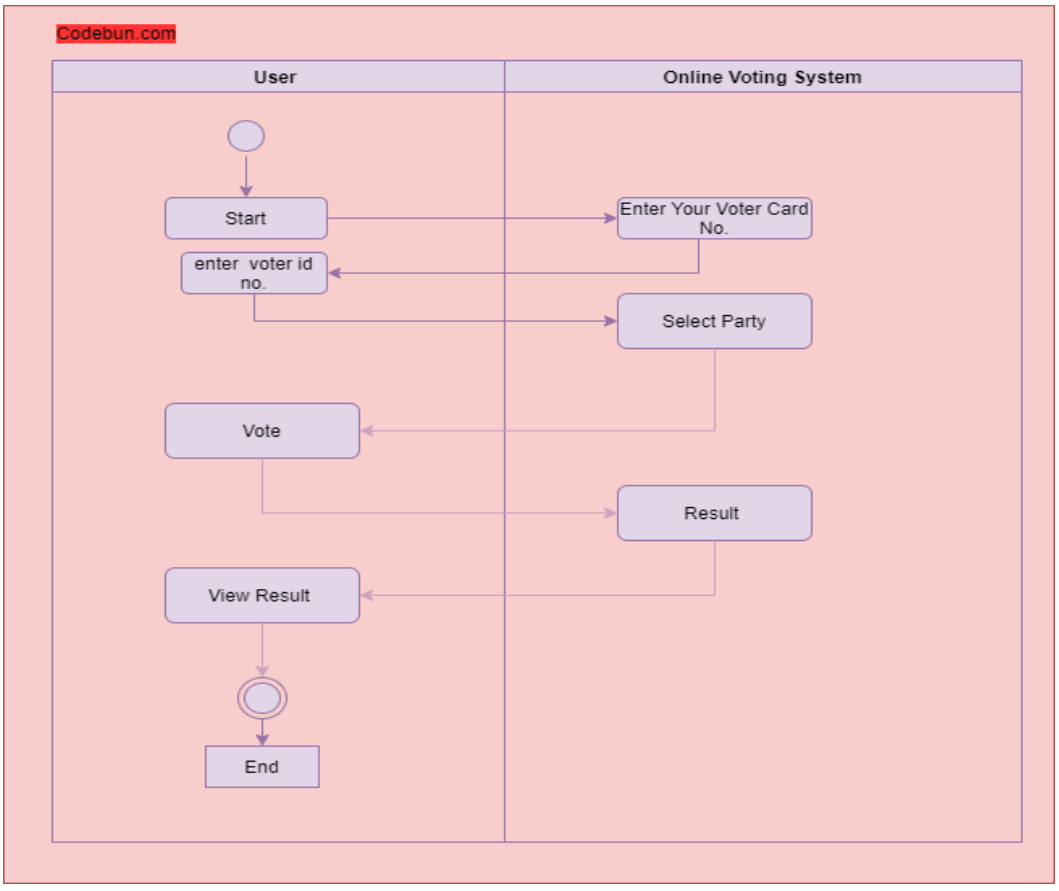
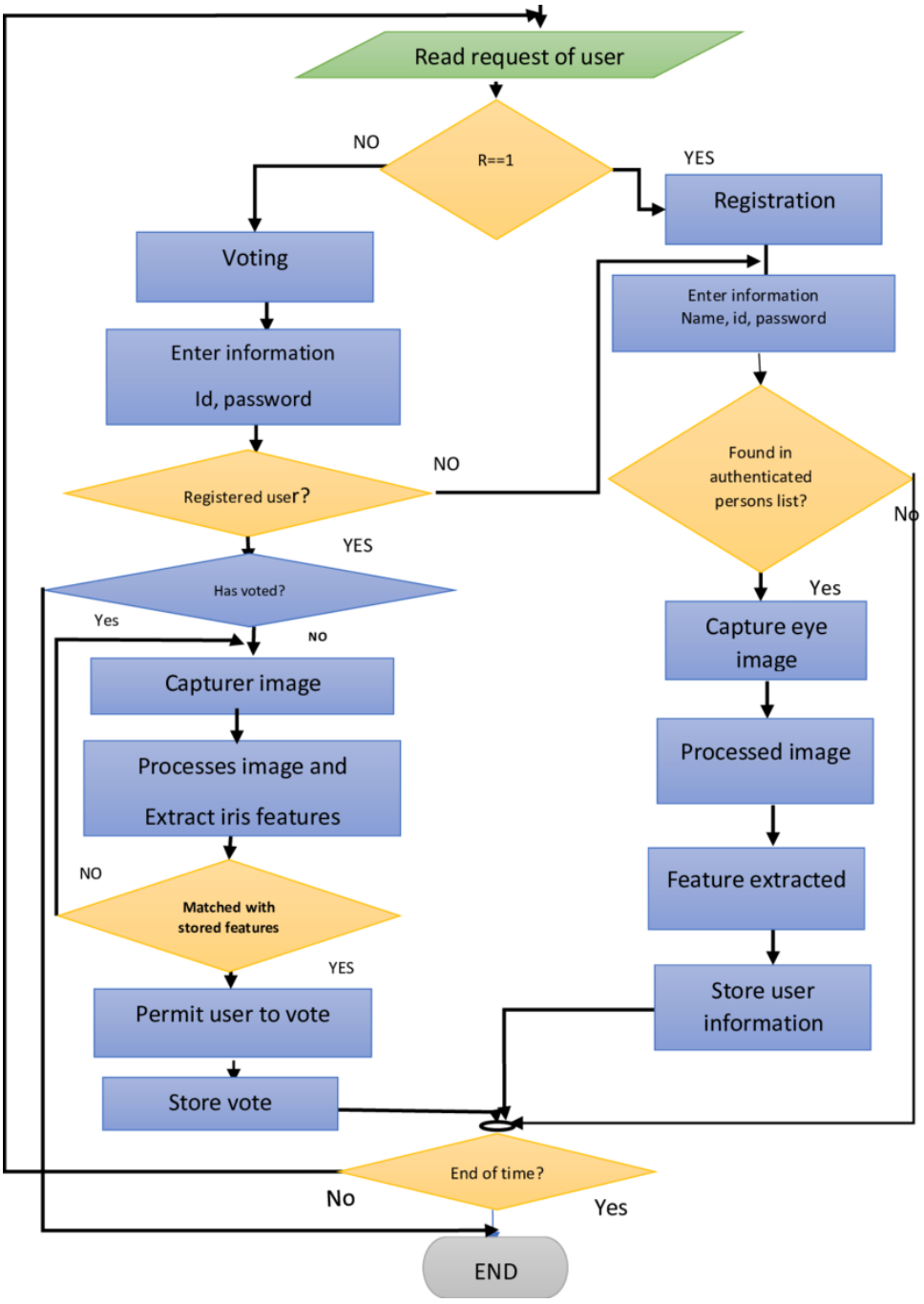
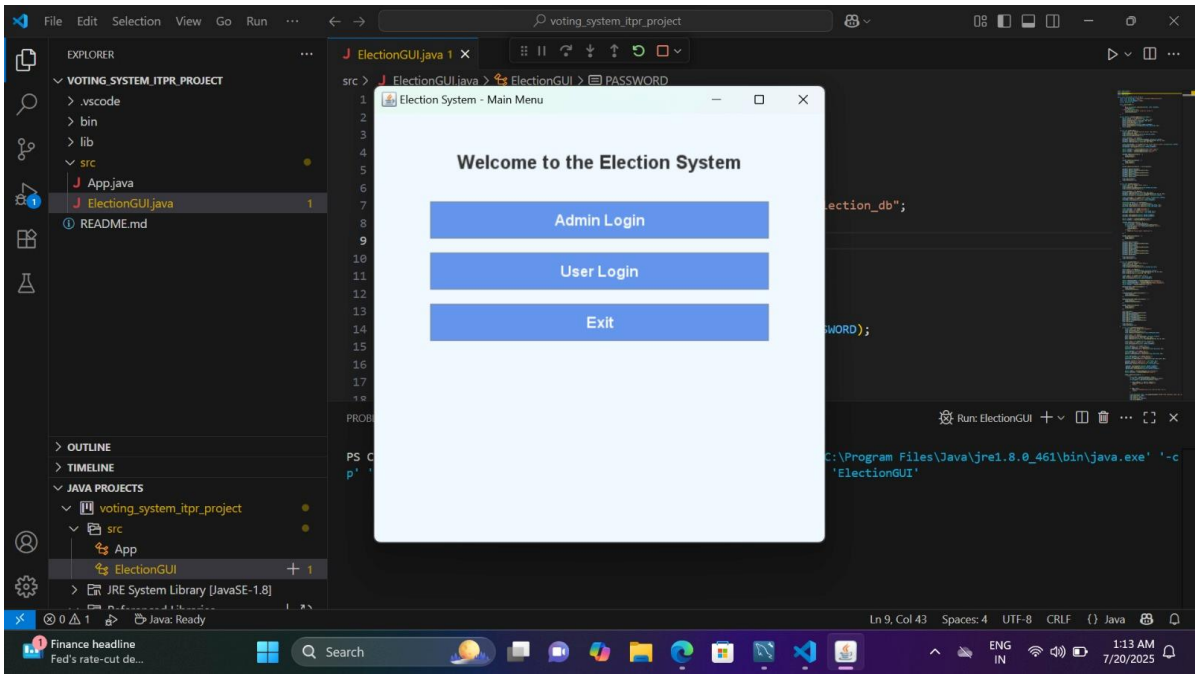


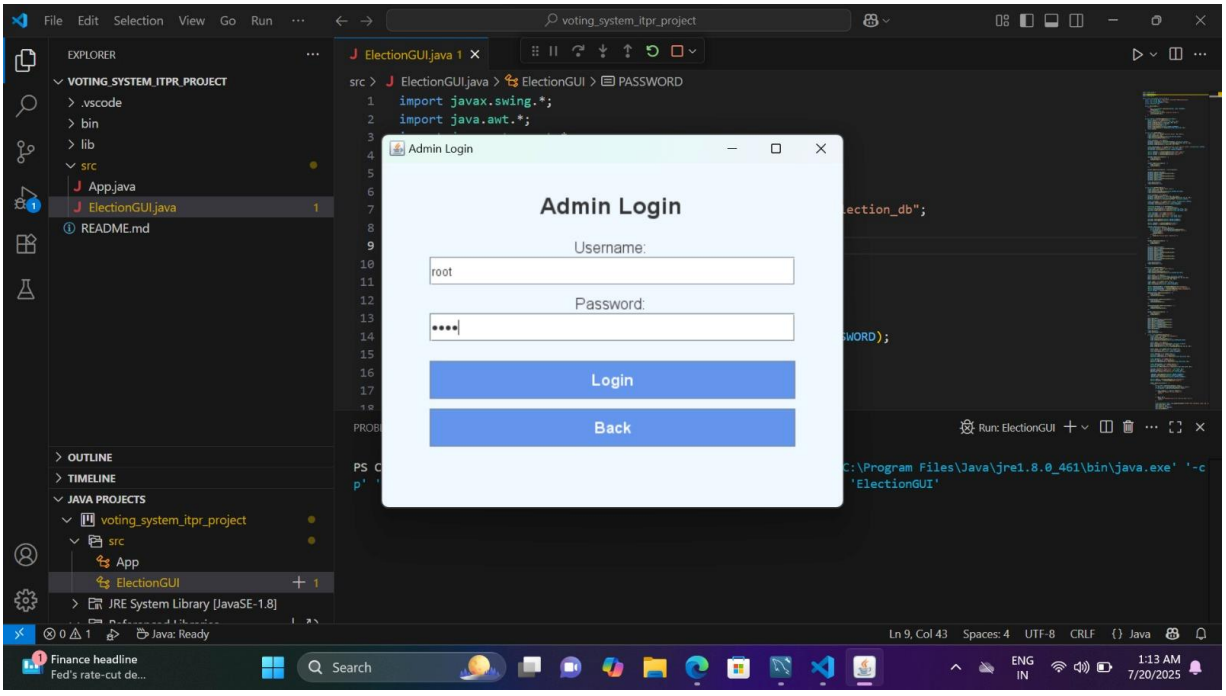
Fig 4 Activity Diagram

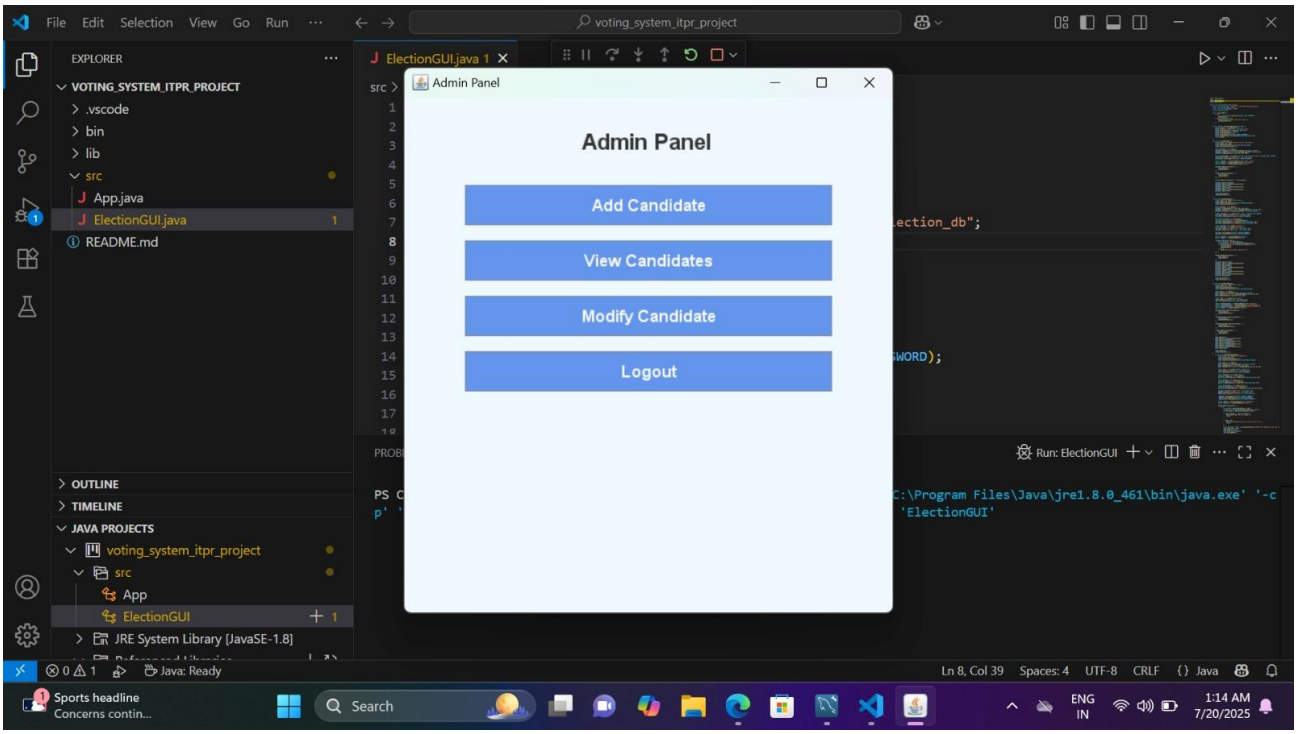
5. Flowchart

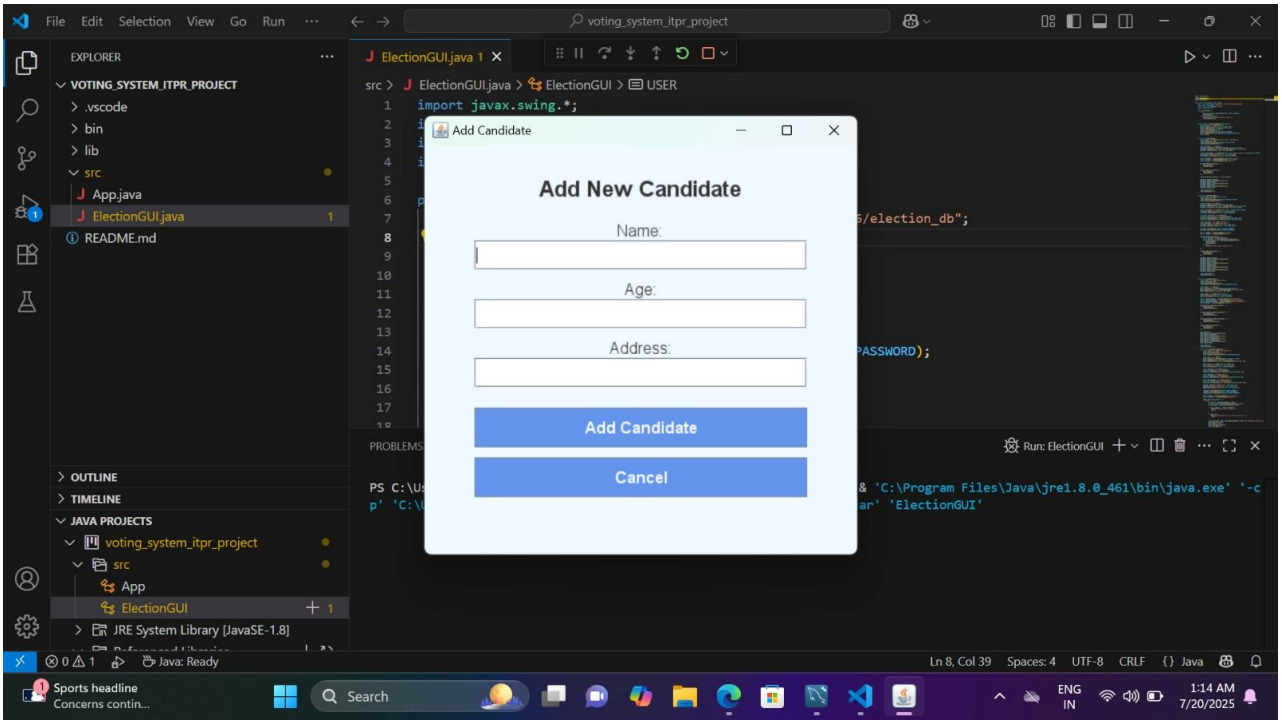

Read request of user
R==1
NO
YES
Registration
Voting
Enter information
Name, id, password
Enter information
Id, password
Found in
authenticated
persons list?
NO
Registered user?
No
YES
Has voted?
Yes
Yes
NO
Capture eye
image
Capturer image
Processed image
Processes image and
Extract iris features
Feature extracted
NO
Matched with
stored features
YES
Store user
information
Permit user to vote
Store vote
End of time?
No
Yes
END

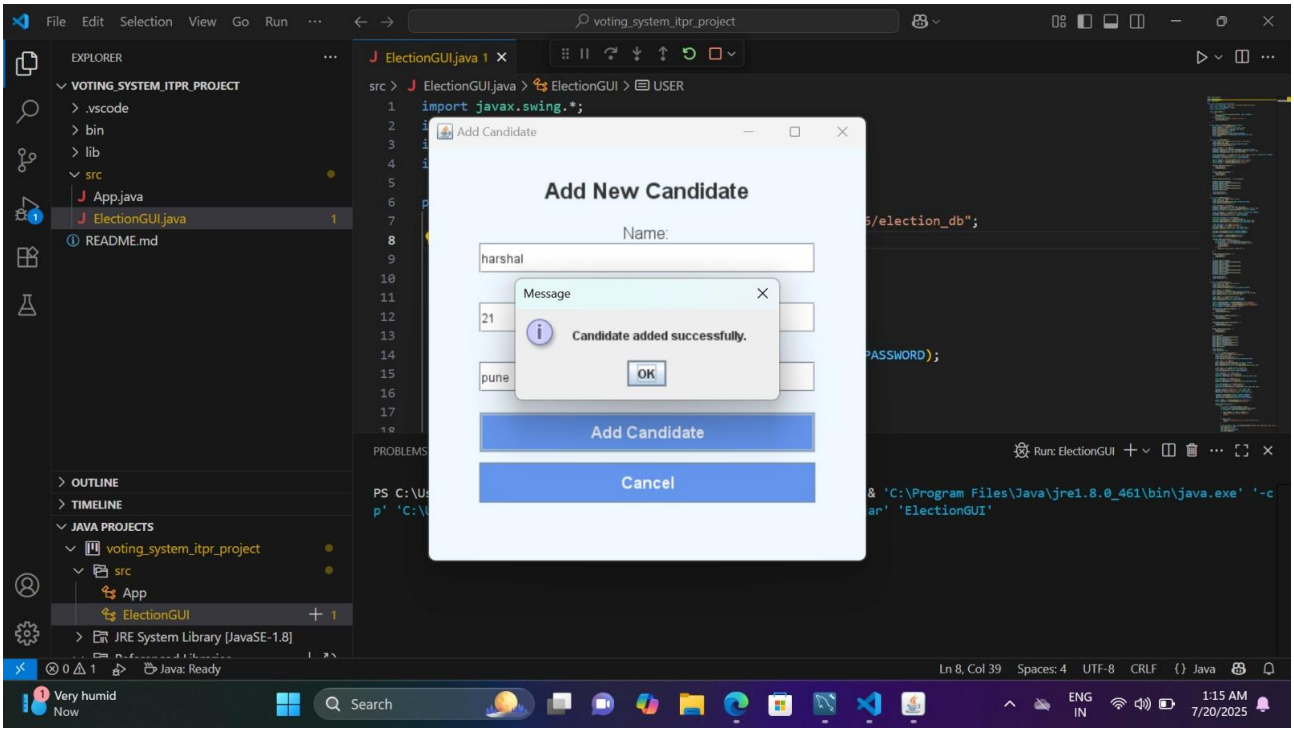
output:

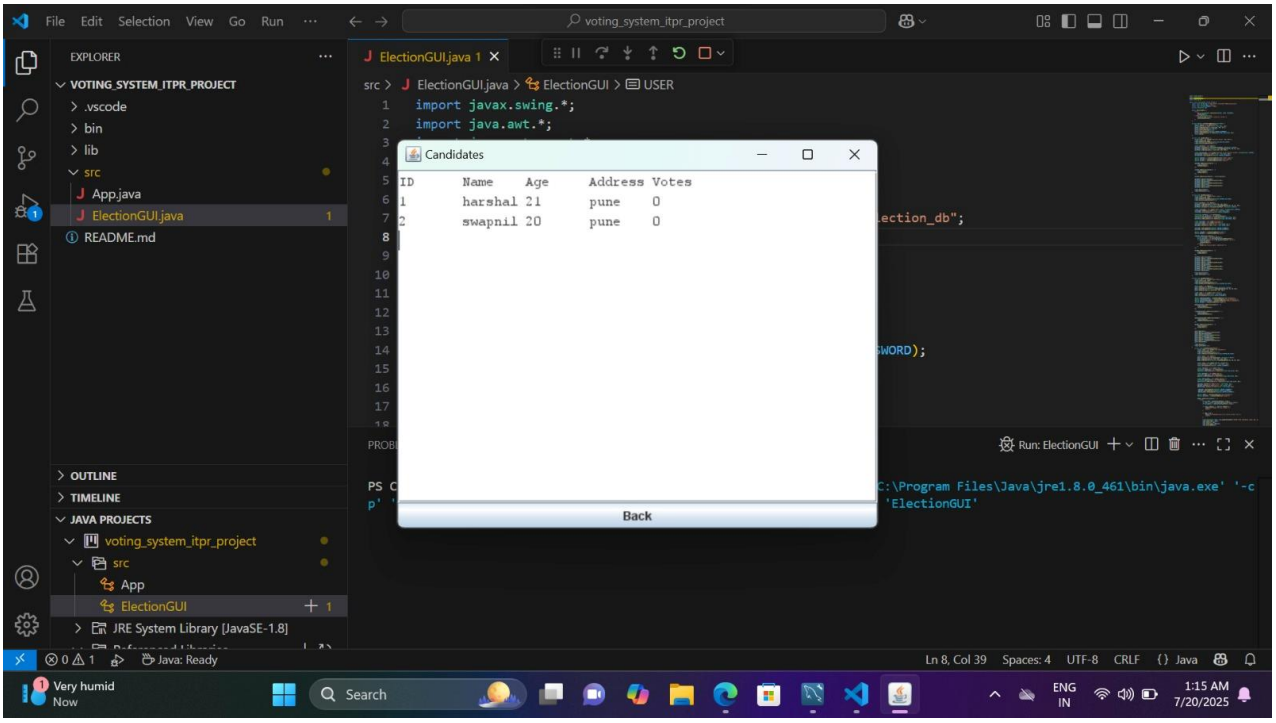

Election System - Main Menu
Welcome to the Election System
Admin Login
User Login
Exit

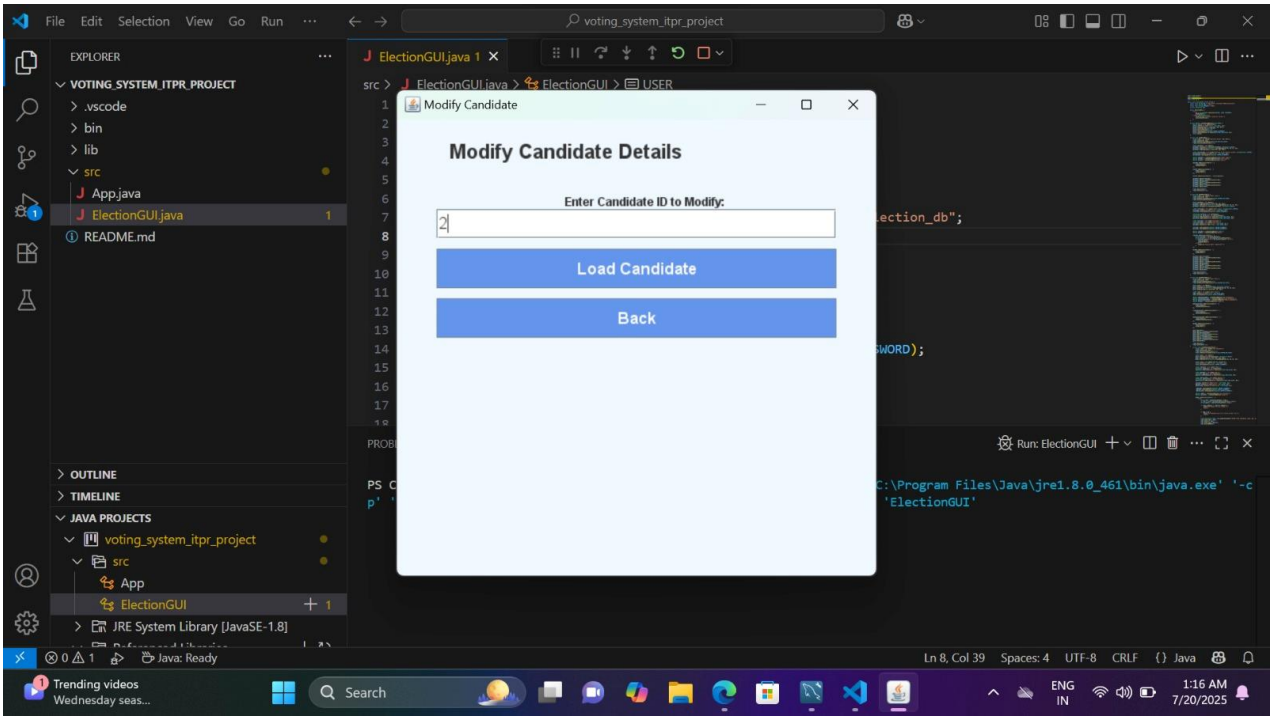

Admin Login
Admin Login
Username:
root
Password:
Login
Back

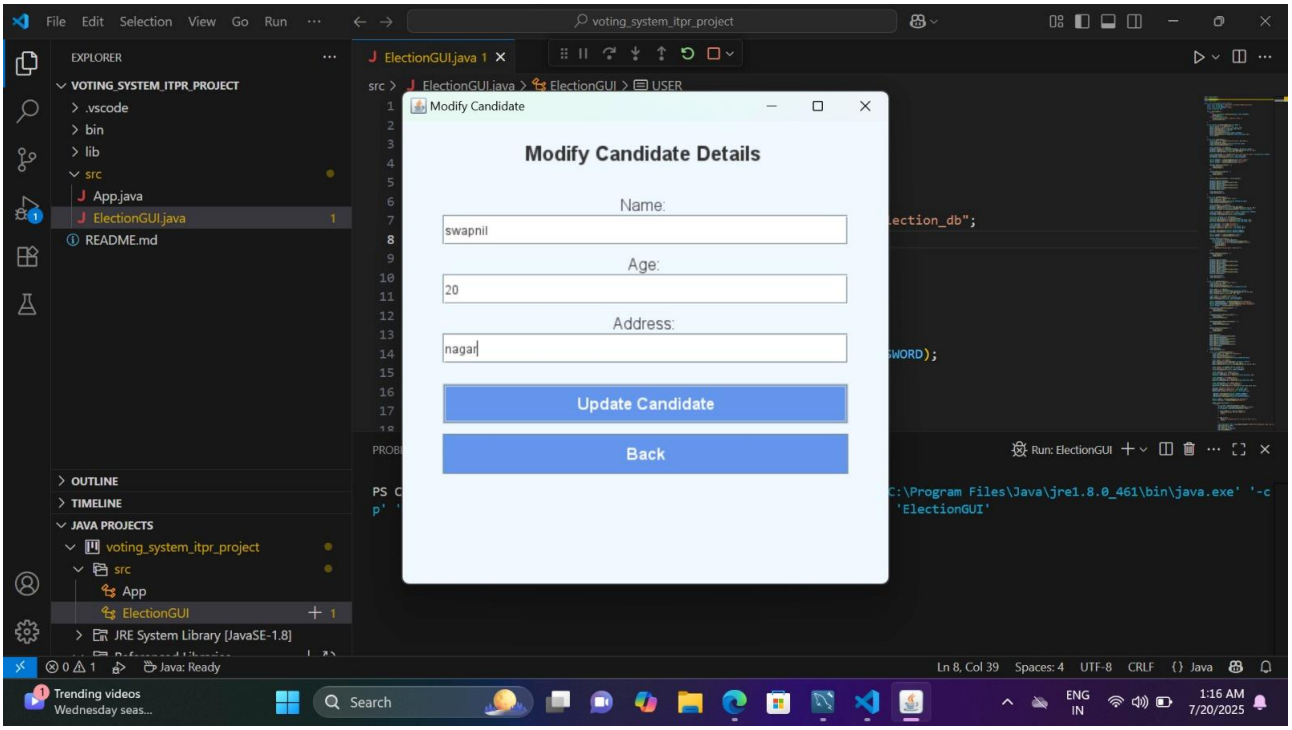

Admin Panel
Admin Panel
Add Candidate
View Candidates
Modify Candidate
Logout

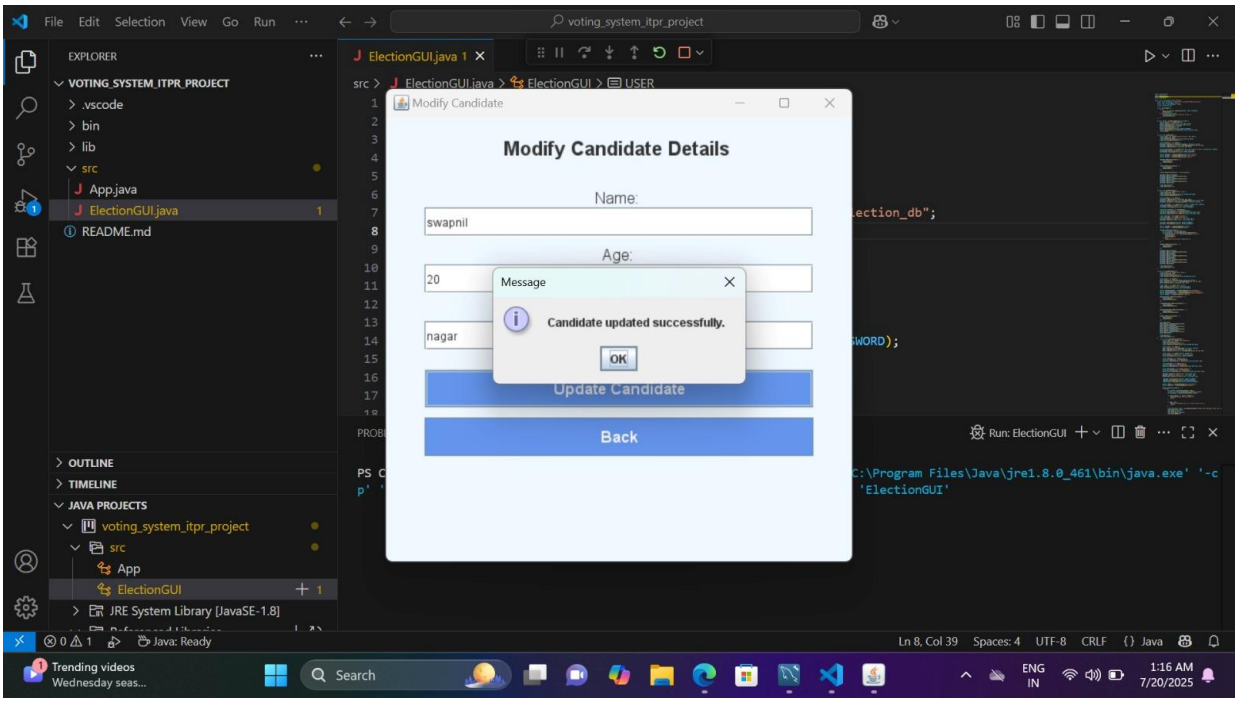

Add Candidate
Add New Candidate
Name:
Age:
Address:
Add Candidate
Cancel

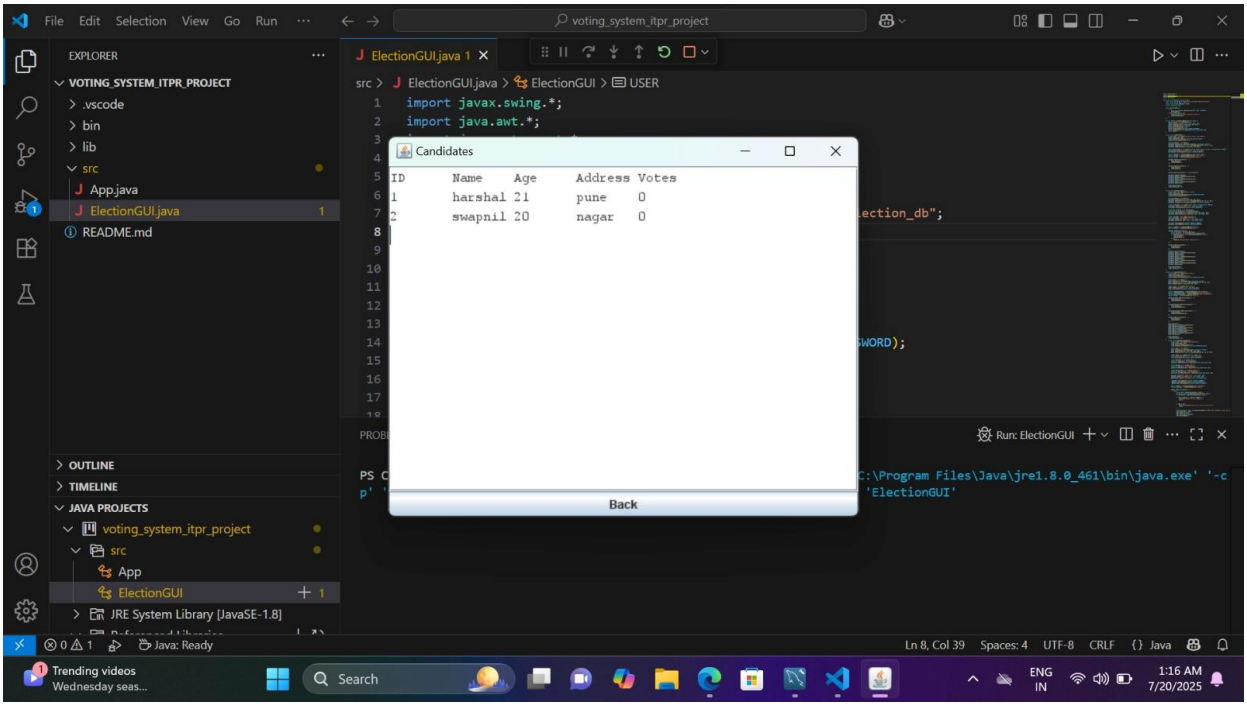

Add Candidate
Add New Candidate
Name:
harshal
21
pune
Message
Candidate added successfully.
OK
Add Candidate
Cancel

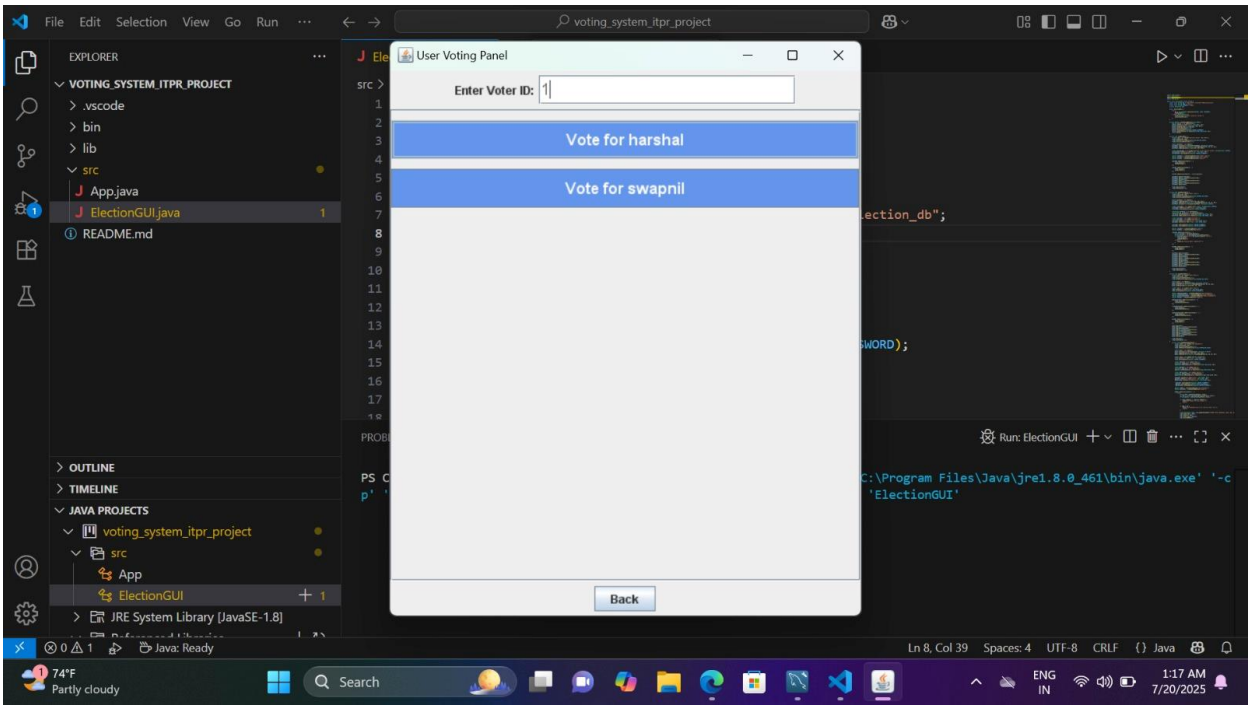

Candidates
ID Name Age Address Votes
1 harshal 21 pune 0
2 swapnil 20 pune 0
Back

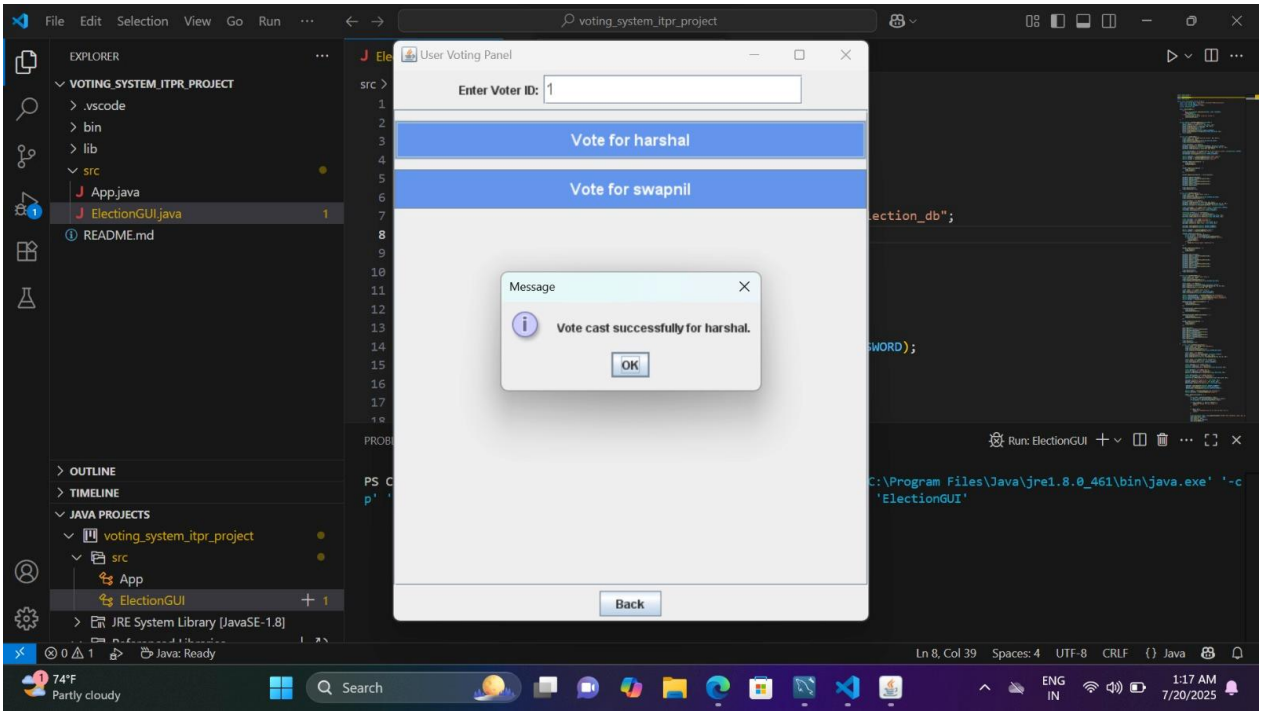

Modify Candidate
Modify Candidate Details
Enter Candidate ID to Modify:
2
Load Candidate
Back

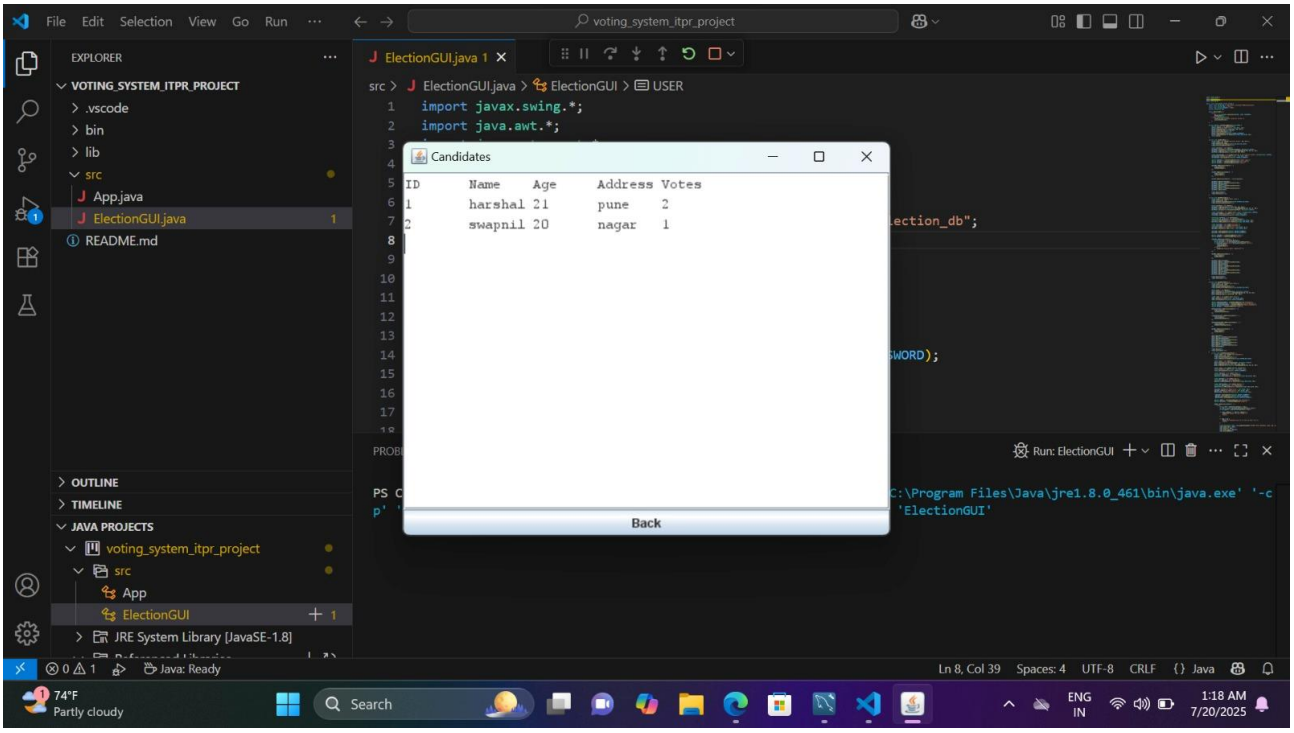

Modify Candidate
Modify Candidate Details
Name:
swapnil
Age:
20
Address:
nagar
Update Candidate
Back


Modify Candidate
Modify Candidate Details
Name:
swapnil
Age:
20
nagar
Message
Candidate updated successfully.
OK
Update Candidate
Back


Candidates
ID Name Age Address Votes
1 harshal 21 pune 0
2 swapnil 20 nagar 0
Back


User Voting Panel
Enter Voter ID: 1
Vote for harshal
Vote for swapnil
Back


User Voting Panel
Enter Voter ID: 1
Vote for harshal
Vote for swapnil
Message
Vote cast successfully for harshal.
OK
Back


File Edit Selection View Go Run
voting_system_itpr_project
EXPLORER
VOTING_SYSTEM_ITPR_PROJECT
.vscode
bin
lib
src
App.java
ElectionGUI.java
README.md
ElectionGUI.java 1
src > ElectionGUI.java > ElectionGUI > USER
1 import javax.swing.*;
2 import java.awt.*;
Candidates
ID Name Age Address Votes
1 harshal 21 pune 2
2 swapnil 20 nagar 1
Back
ection_db";
WORD);
Run: ElectionGUI
C:\Program Files\Java\jre1.8.0_461\bin\java.exe' '-c
'ElectionGUI'
OUTLINE
TIMELINE
JAVA PROJECTS
voting_system_itpr_project
src
App
ElectionGUI
JRE System Library [JavaSE-1.8]
Java: Ready
Ln 8, Col 39 Spaces: 4 UTF-8 CRLF Java
74°F
Partly cloudy
Search
ENG
IN
1:18 AM
7/20/2025

CHAPTER V: CONCLUSION

The **Online Voting System** is a console-based application developed using Java and integrated with a MySQL database through JDBC. It offers a secure, efficient, and reliable solution for managing the core processes of an electronic voting platform, including voter registration, candidate management, vote casting, and result generation. The system is designed to eliminate the inefficiencies and vulnerabilities of traditional paper-based elections by introducing a structured, automated approach to the voting process.

By automating these essential tasks, the system ensures accurate vote tracking, prevents duplicate voting, and delivers real-time election results. The use of Java ensures platform independence, while MySQL provides a secure and organized way to manage critical data such as user credentials, election configurations, and vote records. JDBC acts as the communication layer, enabling smooth interaction between the Java application and the database.

This project is well-suited for small to medium-scale elections such as in schools, colleges, organizations, and communities. Its modular structure allows for easy maintenance and future scalability, including potential upgrades like GUI integration, biometric voter validation, or web-based and mobile app versions. Overall, the system not only fulfills the fundamental requirements of a digital voting platform but also serves as a practical and educational foundation for developing more advanced and large-scale election management systems in the future.

References:

[1] Dr. Z.A. Usmani, Kaif Patanwala, Mukesh Panigrahi, Ajay Nair, “Multi-Purpose Platform Independent Online Voting System”, International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), 2017

[2] Cesar R. K, et.al., “Web 2.0 E-Voting System Using Android Platform”, IEEE International Conference on Progress in Informatics and Computing, pp. 1138-1142, 2010

[3] Mohammad Hosam Sedky, Essam M. Ramzy Hamed, “A Secure e-Government's e-Voting System”, Science and Information Conference, pp. 1365-1373, 2015

[4] Ramya Govindaraj, Kumaresan P, K.Sree harshitha, “Online Voting System using Cloud”, International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), pp. 1-4, 2020