D Y PATIL
COLLEGE of
ENGINEERING & TECHNOLOGY
(AN AUTONOMOUS INSTITUTE)
KASABA BAWADA, KOLHAPUR

Computer Science & Engineering Department
D.Y. Patil College of Engineering & Technology, Kasaba Bawada Kolhapur

MINI PROJECT REPORT

SYA-CSE, AOOC, May 2025

Java-Based ONLINE VOTING SYSTEM

,

Batch	Group No.	Juno ID	Roll No.	Student Name	Sign
S4	G18	EN23178604	83	Sahil Shailesh Raje	
		EN23127155	92	Nihar Nitin Mane	
		EN23248292	79	Yadnesh Kiran Deshmukh	
		EN23166184	94	Saurabh Vaijinath Kumbhar	

Course Faculty: Prof S. B. Patil

1. PROBLEM STATEMENT & INTRODUCTION

In today's digital world, secure and accessible voting is a crucial requirement for various types of elections—whether it's for schools, organizations, communities, or even large-scale government processes. With the increasing shift toward online platforms, ensuring that only eligible voters can participate and that every vote is counted accurately has become more important than ever. However, many systems either fall short on security or lack basic features like voter authentication and result integrity, which can lead to distrust and misuse.

This project introduces a simple, Java-based **Online Voting System** designed to tackle these issues. It provides a basic yet functional framework for managing users, casting votes, and viewing results using the fundamental principles of Object-Oriented Programming (OOP). The system is modular and easy to expand, making it a great starting point for learning or building more advanced e-voting applications.

Some key features include:

• Voter Registration: Allows eligible users to sign up with validated credentials.

• Login System: Authenticates voters and grants access to the voting process.

• Voting Mechanism: Ensures each user can cast only one vote securely.

• Result Display: Shows real-time or final vote counts for each candidate.

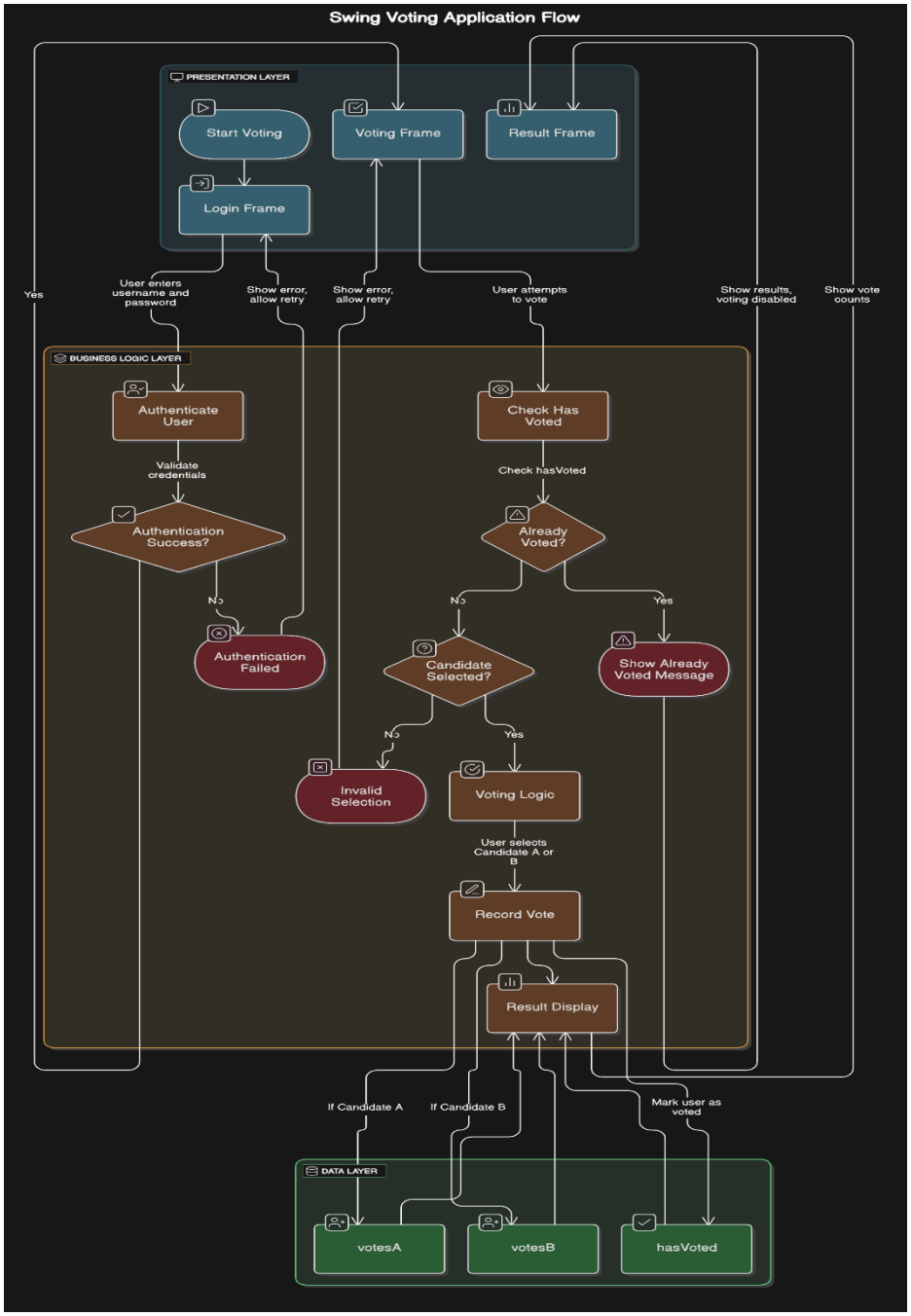
• Voter Listing: Displays all registered users—useful for admin oversight.

The interface is built using a command-line environment, making it lightweight and beginner-friendly. All data is stored temporarily in memory using Java's built-in structures like arrays and objects—no external databases are required. This approach keeps the focus on Java basics like encapsulation, class structures, and user-defined methods.

Even though this is a simple system, it introduces important concepts like data validation, error handling, and session control. It also lays the groundwork for future improvements such as persistent data storage, encryption, multi-role access (admin vs. voter), and web-based or graphical interfaces.

Whether you're just starting with Java or looking to prototype an electronic voting system, this project offers a solid foundation that balances functionality with simplicity.

2. SYSTEM ARCHITECTURE



3. CLASS DIAGRAM

4. MODULE DESCRIPTION

a. User.java — The User Blueprint (Model Class)

User.java is the foundational model class that represents a single user in the system. It defines the structure and behavior of what a "user" means in this application.

Key Features:

• User Storage:

- **Maintains a list (e.g., ArrayList<User>) to store all registered users in memory.**

• Sign-Up Method:

- **Allows new users to register by entering a unique email and password.**
- **Includes checks to prevent duplicate email registrations.**
- **Validates email format and password strength before creating the account.**

• Login Method:

- **Verifies the entered email and password against stored users.**
- **Returns a User object if authentication is successful; otherwise, returns null.**

• Password Update Method:

- **Allows an authenticated user to securely change their password.**
- **Uses the setPassword() method from the User class for encapsulation.**

• View User Password:

- **Enables a logged-in user to view their current password (in a real system, this would be avoided or encrypted, but it's useful for learning purposes).**

• User Listing:

- **Displays the emails of all registered users—useful for admin-like features.**

• Helper Methods:

- **Includes utility methods for checking if an email is already registered, validating inputs, and managing user sessions.**
- **Purpose:**

  The UserManager class acts as the **controller** in this system. It doesn't directly handle user input/output (that's handled in the main driver class), but it encapsulates all the logic needed to manage users effectively. It keeps the code organized by separating responsibilities between data (User.java) and logic (UserManager.java).
- This modular structure makes it easier to expand the system later with features like persistent storage, encryption, or GUI-based input handling.

b. UserManager.java — The Brain of the System (Logic & Data Handler)

UserManager.java is the heart of the application, responsible for managing the entire lifecycle of user accounts. It contains the logic that handles authentication, user tracking, and input validation.

Key Responsibilities:

1. Voter and Candidate Storage

- **Voter List:**
  Stores all registered voters as Voter objects in a list or array.
- **Candidate List:**
  Maintains a list of Candidate objects representing those eligible for votes.
- **Current Session:**
  Tracks the currently logged-in voter to simulate a secure voting session.

2. Core Functionalities

- **Voter Registration:**
  Validates unique Voter ID, email, and password. Adds new voters after successful checks.
- **Voter Login:**
  Authenticates voter credentials and grants access to voting options.
- **Cast Vote:**
  Allows a logged-in voter to vote for a candidate. Ensures only one vote per voter.
- **View Vote Status:**
  Lets the voter confirm whether they have voted, and view their selected candidate (optional).
- **Add Candidate (Admin):**
  Enables administrators to register new candidates for the election.
- **View Results (Admin):**
  Displays live or final voting results, restricted to administrators.
- **List Candidates:**
  Shows the list of all registered candidates available for voting.

3. Validation Methods

- **Email Validation:**
  Checks that email inputs are correctly formatted.
- **Password Strength Validation:**
  Ensures passwords meet defined criteria:
  - **Minimum character length**
  - **Includes special characters, numbers, and varied cases**
- **Voting Eligibility Validation:**
  Verifies that a user is registered and has not already voted.

Purpose

The VotingManager acts as the core controller of the online voting system, managing logic, enforcing rules, and facilitating secure interactions. It separates application logic from the user interface (CLI, web, or mobile), enabling clean, testable, and maintainable code.

c. Main.java — The Entry Point & User Interface

Main.java contains the main() method, which is where the program starts. It provides the interface through which users interact with the system, though in this case, it's a text-based interface using the command line.

Key Responsibilities:

1. Menu Display

- **Initial Menu:**
  Presents options such as Sign Up, Login, and Exit to unauthenticated users.
- **Post-Login Menu:**
  Displays additional options based on the login role:
  - **Standard User: View Password, Update Password, Logout**
  - **Admin: List Users, Logout**

2. User Interaction

- **Input Handling:**
  Collects user input through a Scanner for menu navigation and data entry.
- **Command Routing:**
  Interprets user choices and invokes the corresponding methods in UserManager.

3. Control Flow

- **Role-Based Navigation:**
  Dynamically switches between user and admin menus based on the logged-in user's role.
- **Looping Structure:**
  Keeps the application running in a loop, repeatedly showing menus until the user chooses to exit.

Purpose

Main.java acts as the interface controller, managing user interactions and orchestrating the application's flow. It serves as the bridge between the user interface (CLI) and the logic layer (UserManager), ensuring a smooth and intuitive experience.

1. SCREEN-SHOTS

1. Login :

2. Cast Your Vote :

3. Voting Results:


Voting Results
Voting Results
Candidate A: 0 votes
Candidate B: 1 votes