

A

Project Report on

# SPARKING LOKTANTRA

Submitted in partial fulfillment of completion of the course

Advanced Diploma in IT, Networking and Cloud

Submitted by:

**ANURAG KUMAR**

**RAJIV KUMR**

**MD GAFFAR**

**AMAN HATWAR**

Under Guidance of:

**Apoorva kr. (IBM)**

**Mr. GANAPAVARAPU NAVEEN ( Edunet )**



Year 2024

## Abstract

Sparking\_Loktantra is an innovative e-voting platform designed to solve the problem of voter disenfranchisement for citizens who are out-of-state, such as students and professionals, and are unable to vote in person during elections. This platform leverages web technologies such as CSS, Bootstrap, JavaScript, JSP, Java Servlets, and MySQL for secure data management and efficient voting operations. The system allows voters to register and log in, submit a valid reason for voting remotely, and receive a unique voting code from the admin after verification, enabling them to cast their vote securely online.

The platform integrates IBM Watson AI for real-time chatbot assistance, guiding voters through the registration and voting process. Security measures such as two-factor authentication (2FA) and geo-location tracking ensure that only verified voters can access the system, maintaining the integrity of the voting process and preventing fraud.

Sparking\_Loktantra is designed for flexible deployment across a variety of environments, providing scalability for larger elections. Future enhancements will include biometric verification and blockchain technology to further strengthen security. The platform aims to revolutionize the electoral process by making voting more accessible, secure, and efficient, particularly for those unable to return to their home state, ensuring that every citizen's voice is heard.

## Acknowledgement

We would like to express our deepest gratitude to everyone who supported and guided us throughout the development of Sparking\_Loktantra, our e-voting platform.

First and foremost, we extend our heartfelt thanks to our mentors and instructors for their valuable insights, constructive feedback, and continuous encouragement. Your expertise and guidance have been instrumental in shaping this project.

We are also grateful to our team members—Aman Hawar, Anurag Kumar, Rajiv Kumar, and Md Gaffar—for their dedication, hard work, and seamless collaboration throughout the development process. This project would not have been **possible** without the collective efforts and teamwork displayed by each member.

A special mention goes to the developers of the technologies we used, including CSS, Bootstrap, JavaScript, Java, JSP, Java Servlets, and MySQL, which formed the backbone of our solution. We would also like to acknowledge IBM Watson AI for its contribution to enhancing our platform's user experience with real-time assistance.

Lastly, we express our sincere thanks to our friends and family for their unwavering support and encouragement, which motivated us throughout this project.

## Team Composition and Workload Division

The development of Sparking\_Loktantra was a collaborative effort among four team members, each contributing their unique skills to different aspects of the project. The workload was divided as follows:

### Anurag Kumar

**Role:** Frontend Development

**Responsibilities:** Anurag designed and implemented the user interface using CSS, Bootstrap, and JavaScript, focusing on creating a seamless and responsive user experience.

### Aman Hatwar

**Role:** AI Integration Specialist

**Responsibilities:** Aman handled the integration of IBM Watson AI into the platform, providing users with real-time assistance through the chatbot during registration and voting.

### Md Gaffar

**Role:** Testing and Quality Assurance

**Responsibilities:** Gaffar was responsible for testing the platform for functionality, security, and usability, performing both manual and automated testing to ensure reliability and a smooth user experience.

### Rajiv Kumar

**Role:** Backend Development & Deployment

**Responsibilities:** Rajiv Kumar developed the backend using Java and Java Servlets, ensuring secure data handling and proper integration with the MySQL database. He also contributed to the deployment and optimization of the system's backend processes.

Each team member's role was essential to the project's success, ensuring a balance between frontend, backend, AI support, and quality testing to deliver a well-rounded and functional e-voting platform.

# **Table of Contents**

## **1. Introduction to Problem**

In democratic systems, voting is one of the most fundamental rights and responsibilities of citizens. However, a significant portion of eligible voters, especially those living away from their home states for work, education, or other reasons, face challenges in participating in elections. Many citizens are unable to travel back to their hometowns due to logistical and financial constraints, resulting in disenfranchisement.

This issue is particularly relevant in a diverse country like India, where a large number of students and professionals migrate between states. Despite their eagerness to participate in the electoral process, these individuals often miss out on exercising their voting rights, which undermines the inclusivity and fairness of elections.

Existing voting methods, such as physical ballots and early voting, fail to accommodate this group of people, leading to decreased voter turnout and weakening democratic representation. Additionally, manual voting systems are prone to fraud, logistical challenges, and inefficiencies.

Thus, there is a critical need for a secure, reliable, and accessible solution that enables out-of-state citizens to vote remotely, ensuring that every eligible voter has the opportunity to participate in elections. Sparking\_Loktantra is designed to address this problem by offering an online voting platform that allows citizens to vote from anywhere, with secure and efficient processes that uphold the integrity of elections.

## **2. Literature Review**

The concept of online voting has been a topic of considerable research and experimentation over the past few decades. Governments and organizations worldwide have explored various online voting systems in a bid to improve voter

participation, enhance convenience, and maintain election security. In this section, we review relevant literature and studies related to online voting systems, their challenges, and emerging technologies that can make such systems feasible.

### **E-Voting Systems**

The first significant experiment in electronic voting was conducted in the 1990s, with several countries like Estonia, Brazil, and Switzerland testing electronic voting machines and remote voting systems. Estonia is widely regarded as the pioneer of internet voting, having implemented a national system in 2005. According to research, Estonia's success in online voting is attributed to its strong digital infrastructure, secure identification systems, and a high level of public trust in e-governance.

### **Security and Privacy Concerns**

One of the most critical challenges in online voting systems is ensuring security and voter privacy. Studies have shown that online voting systems are prone to cyberattacks, voter impersonation, and hacking, which can undermine the integrity of elections. Researchers have proposed solutions such as two-factor authentication, end-to-end encryption, and blockchain technology to address these security concerns. Literature also highlights the importance of transparency in online voting systems to build public trust.

### **Blockchain in Voting**

Blockchain has emerged as a promising solution for enhancing the security and transparency of online voting. In recent years, research on blockchain-enabled voting systems has shown potential for tamper-proof, decentralized voting records that can be verified by multiple parties. However, studies also point out the scalability and resource constraints associated with blockchain systems, especially when applied to large-scale elections.

### **AI Integration in Voting**

Artificial intelligence (AI) is another area explored in the literature to assist voters and election authorities. AI-driven chatbots and virtual assistants, such as IBM Watson AI, have been tested to provide real-time support for voters during the registration and voting process. Literature suggests that AI can improve voter engagement, resolve technical queries, and enhance the overall user experience in online voting systems.

### **Existing Remote Voting Solutions**

There have been attempts to create remote voting solutions, primarily targeted at expatriates and overseas citizens. Many such systems rely on mail-in ballots or restricted access to online portals. However, these solutions have limitations in terms of accessibility, security, and real-time feedback, as highlighted by several studies.

### **Challenges in the Indian Context**

Research shows that implementing a remote voting system in India presents unique challenges due to the country's vast geography, diverse population, and varying levels of digital literacy. Studies suggest that for any e-voting platform to succeed in India, it must cater to a wide demographic, ensuring ease of use, security, and scalability. Moreover, concerns around electoral fraud and political manipulation remain significant hurdles to adoption.

## **3. Proposed Solution**

Sparking\_Loktantra is a web-based e-voting platform designed to address the problem of voter disenfranchisement among out-of-state citizens. The proposed solution provides a secure, accessible, and transparent system that allows eligible voters to participate in elections from any location, ensuring their right to vote is preserved regardless of their geographical constraints.

### **The solution is structured around two main phases:**

#### **Registration and Login:**

Voters must first create an account on the platform by providing their government-issued voter ID and other necessary credentials.

After registration, users log in to the system using their credentials and undergo two-factor authentication (2FA) to enhance security.

This phase ensures that only verified individuals can access the voting system, preventing unauthorized use.

### **Voting Request and Approval:**

In the second phase, voters who are unable to return to their home state must submit a request, providing a valid reason for remote voting (e.g., employment or educational commitments in another state).

The admin panel reviews and verifies the request, checking the voter's eligibility and the validity of the claim.

Upon approval, the admin sends a unique voting code to the voter's registered email. This code can be used to cast their vote securely.

### **Key Features of the Solution:**

#### **AI-Powered Chatbot Assistance:**

The platform integrates IBM Watson AI to offer real-time chatbot assistance for users during the registration and voting process. This feature ensures a smooth user experience by guiding voters through each step and answering any queries they may have.

#### **Security Measures:**

**Two-Factor Authentication (2FA):** Enhances voter security by requiring two levels of verification—first through login credentials and second through a one-time password (OTP) sent to the voter's registered mobile number or email.

**Geo-location Tracking:** Ensures that voters casting their ballots are within valid voting zones or eligible locations, further verifying their authenticity.

#### **Scalable and Accessible:**

The platform is designed using CSS, Bootstrap, JavaScript, Java, JSP, Java Servlets, and MySQL for a smooth, reliable experience. It is scalable to accommodate large numbers of users during major elections and is built to be responsive across devices.

**Benefits:**

Accessibility: Voters who cannot physically return to their home states can still participate in the voting process.

Security: Advanced security protocols, including 2FA and geo-tracking, ensure only eligible and verified voters participate.

Efficiency: The platform simplifies the electoral process for remote voters while maintaining transparency and accountability.

User-Friendly Interface: A simple and intuitive design ensures that even voters with limited digital literacy can easily navigate the system.

## **4. Requirements**

### **4.1 Technology Stack**

The technology stack for Sparking\_Loktantra has been carefully selected to ensure a robust, secure, and user-friendly e-voting platform. Each technology plays a crucial role in the development and functionality of the system.

**Frontend Technologies:**

HTML: The backbone of the web pages, used for structuring the content.

CSS: Used for styling the web pages, providing a visually appealing layout and design.

Bootstrap: A front-end framework that facilitates responsive design, ensuring that the application is accessible on various devices, including desktops, tablets, and smartphones.

JavaScript: Enables dynamic content and interactive features on the web pages, enhancing the user experience.

**Backend Technologies:**



Java: The primary programming language used for developing the server-side logic of the application, providing robustness and scalability.

Java Servlets: A component of Java EE used for handling HTTP requests and responses, enabling the server to process user input and manage session data.

JavaServer Pages (JSP): Allows for the creation of dynamic web content by embedding Java code directly into HTML, facilitating the generation of user-specific pages.

### **Database:**

MySQL: A relational database management system (RDBMS) used for storing user data, voting records, and session information securely. MySQL offers reliability, ease of use, and strong community support, making it a suitable choice for this application.

AI Integration:

IBM Watson AI: Utilized for integrating chatbot functionality into the platform, providing real-time assistance to users during the registration and voting process. This AI system enhances user interaction and helps answer common queries.

### **Security Measures:**

Two-Factor Authentication (2FA): Implemented to add an extra layer of security during the login process, ensuring that only verified users can access their accounts.

Geo-location Tracking: Used to verify the location of voters when they are casting their votes, ensuring that they are within valid voting zones.

Development Tools:

Integrated Development Environment (IDE): Tools such as Eclipse or IntelliJ IDEA for Java development.

Version Control: Git for source code management, allowing for collaborative development and version tracking.

## **4.2 Hardware**

To successfully develop, deploy, and run the Sparking\_Loktantra e-voting platform, the following hardware requirements are recommended:

### **Development Environment:**

Personal Computer (PC) or Laptop:

Processor: Intel i5 or higher (or equivalent AMD processor) for efficient processing and multitasking.

RAM: Minimum 8 GB (16 GB recommended) to support development tools, testing, and running local servers.

Storage: At least 256 GB SSD for faster read/write speeds, with additional HDD space for backups and larger databases if needed.

Graphics Card: Integrated graphics sufficient for development purposes; a dedicated graphics card is optional unless extensive graphical work is involved.

### **Server Environment:**

Web Server:

Processor: Intel Xeon or equivalent multi-core processor to handle concurrent user requests and ensure quick response times.

RAM: Minimum 16 GB (32 GB recommended) for optimal performance under load.

Storage: SSD with at least 512 GB capacity for fast data retrieval and better performance, especially during peak voting periods.

Network Interface: Gigabit Ethernet for high-speed internet connectivity, ensuring quick data transfer and reduced latency for users.

### **Database Server:**

Processor: Intel Xeon or equivalent with multi-core capabilities for handling complex queries and transactions.

RAM: Minimum 16 GB (32 GB recommended) to support efficient database operations and improve performance under load.

Storage: SSD with a capacity of at least 1 TB to accommodate the database size, including user information and voting records, along with redundancy measures for data safety.

### **Testing Environment:**

Testing Machines:

Similar to the development environment, testing machines should have comparable specifications to ensure that the application is tested under realistic conditions.

## **4.3 Software**

To effectively develop, deploy, and maintain the Sparking\_Loktantra e-voting platform, the following software requirements are necessary:

### **Development Tools:**

Integrated Development Environment (IDE):

Eclipse or IntelliJ IDEA: Java IDEs for coding, debugging, and testing the application.

**Version Control System:**

Git: A version control system for tracking changes in source code during development, enabling collaborative work among team members.

**Build Automation Tool:**

Apache Maven: For managing project dependencies and building the Java application efficiently.

Backend Technologies:

**Java Development Kit (JDK):**

JDK 22 or higher: Required for developing the Java components of the application.

**Java Server Technologies:**

Apache Tomcat: A servlet container used to run Java Servlets and JSP, serving as the web server for the application.

Frontend Technologies:

**Web Browsers:**

Latest versions of browsers like Google Chrome, Mozilla Firefox, or Microsoft Edge for testing and running the web application.

HTML/CSS/JavaScript Libraries:

Libraries such as jQuery (optional) for simplifying JavaScript programming, though pure JavaScript can also be used.

Database Management:

**MySQL Database Server:**

MySQL 8.0 or higher: The relational database management system used for storing user data, voting records, and managing application data securely.

AI Integration:

**IBM Watson API:**

The necessary SDK or API access for integrating IBM Watson's AI capabilities into the platform for chatbot functionalities.

**Deployment and Server Software:**

**Operating System:**

Linux (Ubuntu or CentOS): Recommended for the production server environment due to its stability, security, and performance.

**Containerization (Optional):**

Docker: For creating, deploying, and managing containers to ensure consistency across different environments.

Backup and Recovery:

**Backup Software:**

Tools for automating database backups and ensuring data integrity, such as MySQL Workbench or custom scripts.

#### **4.4 Deployment Environment**

The deployment environment for the Sparking\_Loktantra e-voting platform is designed to ensure reliability, scalability, and security. This environment encompasses the hardware, software, and network components necessary for hosting and running the application. Below are the key aspects of the deployment environment:

The deployment of the Sparking\_Loktantra e-voting platform is a crucial phase that transforms the application from a development environment into a publicly accessible platform. The deployment was carried out using GitHub as a repository for version control and collaboration. This section details the deployment strategy, steps involved, and post-deployment considerations.

**Deployment Strategy**

The deployment strategy for Sparking\_Loktantra was designed to ensure that the application is efficiently managed, easily updated, and accessible to users. Key components of the strategy included:

**Version Control with GitHub:**

Utilized GitHub for version control, enabling collaboration among team members and tracking changes to the codebase.

Maintained separate branches for development, testing, and production to streamline the workflow.

**Continuous Integration/Continuous Deployment (CI/CD):**

Implemented a CI/CD pipeline using GitHub Actions to automate the testing and deployment process.

This ensures that code changes are automatically tested, and upon successful completion, deployed to the live environment.

**Documentation:**

Maintained detailed documentation in the GitHub repository to facilitate onboarding and collaboration among team members.

## **5. User Requirements**

The user requirements for the Sparking\_Loktantra e-voting platform are categorized into various segments to ensure that the system meets the needs of its users effectively. The requirements are divided into functional, non-functional, and user interface requirements.

### **5.1 Functional Requirements**

User Registration:

Users must be able to register for the platform using a government-issued voter ID and other required personal information.

The system should send a confirmation email after successful registration.

User Authentication:

Users must log in to the platform using their registered email and password.

The system should implement Two-Factor Authentication (2FA) to enhance security during login.

Voting Request Submission:

Users must be able to submit a request for remote voting, including providing a valid reason (e.g., working or studying in another state).

The system should enable users to track the status of their voting request.

Admin Verification:

Admins should have the ability to review and verify voter requests for remote voting.

The system should notify users via email once their request has been approved or denied.

Voting Process:

Upon approval, users must receive a unique voting code via email to access the voting interface.

Users must be able to cast their votes securely using the provided code.

The system should record votes and ensure they are stored securely.

Chatbot Assistance:

The platform should offer real-time assistance through an AI-powered chatbot to help users with common queries and guide them through the voting process.

### **5.2 Non-Functional Requirements**

Performance:

The system should support a high volume of concurrent users during peak voting periods without degradation in performance.  
Response times for user actions (registration, login, submitting requests) should not exceed 3 seconds under normal load.

#### Security:

The platform must ensure data encryption for sensitive information such as user credentials and voting data.

The system should implement secure coding practices to prevent vulnerabilities such as SQL injection and cross-site scripting (XSS).

#### Usability:

The user interface should be intuitive and easy to navigate for users with varying levels of digital literacy.

The platform should be accessible across multiple devices, including desktops, tablets, and smartphones.

#### Scalability:

The system should be scalable to accommodate increased user loads during elections and other peak periods.

#### Availability:

The platform must be available 24/7, with minimal downtime for maintenance and updates.

### **5.3 User Interface Requirements**

#### Registration and Login Pages:

Clear and user-friendly forms for user registration and login, with appropriate input validation and error messages.

#### Voting Request Page:

A simple interface for users to submit their voting requests, including fields for required information and a submit button.

#### Admin Dashboard:

A comprehensive dashboard for admins to manage user requests, verify submissions, and send notifications to users.

#### Voting Interface:

An easy-to-use interface for casting votes, displaying the options clearly and allowing users to submit their choices with a single click.

#### Chatbot Interface:

A visible and accessible chatbot feature on all pages, providing users with immediate assistance.

## 6. Design Documentation

The design documentation for the Sparking\_Loktantra e-voting platform outlines the architectural and user interface design elements necessary for building a robust and user-friendly application. This section provides an overview of the system architecture, database design, and user interface design.

### 6.1 System Architecture

The architecture of Sparking\_Loktantra follows a three-tier structure, which includes:

#### Presentation Layer (Frontend):

Technologies: HTML, CSS, Bootstrap, JavaScript

Description: This layer is responsible for the user interface. It handles user interactions, displays information, and sends requests to the backend. The use of Bootstrap ensures a responsive design, making the platform accessible on various devices.

#### Business Logic Layer (Backend):

Technologies: Java, Java Servlets, JSP

Description: This layer processes user requests, performs the necessary business logic, and communicates with the database. It handles user authentication, voting requests, and interactions with the AI-powered chatbot.

Data Layer (Database):

Technologies: MySQL

Description: This layer is responsible for data storage and management. It stores user information, voting records, and system logs. The database design ensures data integrity and efficient data retrieval.

### 6.2 Database Design

The database design consists of the following key tables:

#### Registration Table

```
CREATE TABLE registration (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    uname VARCHAR(100) NOT NULL,  
    upwd VARCHAR(255) NOT NULL,  
    uemail VARCHAR(255) UNIQUE NOT NULL,  
    umobile VARCHAR(15) UNIQUE NOT NULL  
);
```

## Users Table

Fields:

```
CREATE TABLE users (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  title VARCHAR(10),  
  name VARCHAR(255) NOT NULL,  
  dob DATE NOT NULL,  
  gender ENUM('Male', 'Female', 'Other') NOT NULL,  
  email VARCHAR(255) NOT NULL,  
  mobile VARCHAR(15) NOT NULL,  
  aadhaar VARCHAR(12) UNIQUE NOT NULL,  
  voter_id VARCHAR(20) UNIQUE NOT NULL,  
  address TEXT NOT NULL,  
  ward VARCHAR(10),  
  pincode VARCHAR(6) NOT NULL,  
  state VARCHAR(50) NOT NULL,  
  district VARCHAR(50) NOT NULL,  
  verified BOOLEAN DEFAULT FALSE,  
  rejected BOOLEAN DEFAULT FALSE,  
  unique_key VARCHAR(255)  
);
```

## Admin Table

```
CREATE TABLE admins (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  username VARCHAR(100) NOT NULL,  
  password VARCHAR(255) NOT NULL  
);
```

```
INSERT INTO admins (username, password) VALUES ('admin', '123');
```

## Voting Requests Table

```
CREATE TABLE votes (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  aadhaar VARCHAR(12) NOT NULL,  
  unique_key VARCHAR(10) NOT NULL,  
  leader_name VARCHAR(255) NOT NULL,  
  party_name VARCHAR(255) NOT NULL,  
  timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);  
-- Optional: Add indexes for performance  
CREATE INDEX idx_aadhaar ON votes(aadhaar);  
CREATE INDEX idx_unique_key ON votes(unique_key);
```



### **6.3 User Interface Design**

The user interface (UI) design is structured to be intuitive and user-friendly, ensuring ease of navigation for all users. Key pages include:

Registration Page:

Input fields for name, email, password, voter ID, and a registration button.

Clear error messages for validation feedback.

Login Page:

Input fields for email and password, along with a login button.

Option for password recovery.

Voting Request Page:

Form for users to enter the reason for remote voting and submit the request.

Status update area to track the request.

Admin Dashboard:

Overview of pending requests, with options to approve or deny.

User management and reporting tools.

Voting Interface:

Display of candidates with radio buttons for selection.

Submit button for casting votes, along with a confirmation dialog.

Chatbot Interface:

Chat widget accessible from all pages to assist users in real-time.

## **7. Implementation Details**

The implementation details for the Sparking\_Loktantra e-voting platform encompass the actual coding practices, frameworks, and methodologies employed during the development process. This section outlines the steps taken in implementing the project, including coding standards, integration of technologies, and key functionalities.

### **7.1 Development Environment Setup**

Tools Used:

IDE: Eclipse or IntelliJ IDEA for Java development.

Version Control: Git and GitHub for code versioning and collaboration.

Build Tool: Apache Maven for managing dependencies and building the project.

Environment Configuration:

Web Server: Apache Tomcat is installed and configured to serve the Java Servlets and JSP pages.

Database: MySQL Server is set up to manage user data, voting records, and requests.

## **7.2 Coding Standards**

Java Coding Standards:

Followed Java naming conventions for classes, methods, and variables (CamelCase for classes, camelCase for methods and variables).

Used meaningful variable names to enhance code readability.

JavaScript and CSS Standards:

JavaScript functions were modularized to promote reusability.

CSS styling adhered to best practices, ensuring responsive design using Bootstrap classes.

## **7.3 Key Functionalities Implementation**

User Registration:

Implemented a registration form that captures user details and validates inputs.

Used hashing (e.g., BCrypt) to securely store passwords in the database.

User Authentication:

Developed a login system that verifies user credentials against stored data.

Implemented session management to maintain user login states.

Voting Request Submission:

Created a form for users to submit voting requests, including a reason for remote voting.

Used AJAX for real-time submission and feedback without refreshing the page.

Admin Verification:

Developed an admin dashboard for managing user requests.

Implemented functionalities to approve or deny requests, with notifications sent to users.

Voting Process:

After approval, users received a unique voting code via email.

Built the voting interface allowing users to cast their votes securely, ensuring data integrity during the submission process.

AI Chatbot Integration:

Integrated IBM Watson for chatbot functionalities, allowing users to get assistance with their queries.

Developed endpoints to connect the chatbot with the application for real-time responses.

## **8. Testing**

The testing phase of the Sparking\_Loktantra e-voting platform is critical to ensuring that the application functions as intended, meets user requirements, and provides a secure voting experience. This section outlines the testing strategies employed, the types of testing conducted, and the outcomes of these tests.

### **8.1 Testing Strategies**

The testing approach encompassed various strategies to cover different aspects of the application:

#### **Manual Testing:**

Conducted manual tests to assess the functionality of user interfaces, registration forms, and voting processes.

Engaged end-users in exploratory testing to identify usability issues.

#### **Automated Testing:**

Implemented automated tests using JUnit for Java components to validate business logic and ensure expected outcomes.

Used Selenium WebDriver for UI testing, simulating user interactions with the web application.

#### **Performance Testing:**

Conducted performance tests using tools like Apache JMeter to evaluate the application's response time and load handling capabilities under peak conditions.

#### **Security Testing:**

Performed security assessments to identify vulnerabilities in the application, including penetration testing and vulnerability scanning.

Employed tools like OWASP ZAP to scan for common security threats.

### **8.2 Testing Outcomes**

#### **Defect Resolution:**

Identified and resolved several bugs during the testing phase, including:

Input validation issues on registration forms.

Incorrect status notifications for voting requests.

Minor UI layout inconsistencies across different devices.

#### **Performance Metrics:**

Achieved an average response time of less than 2 seconds for user interactions under normal load.

The application successfully handled up to 1,000 concurrent users during stress tests without performance degradation.

#### **Security Findings:**

Resolved several security vulnerabilities identified during penetration testing.

Implemented additional security measures, including input sanitization and secure password storage practices.

#### User Feedback:

Gathered positive feedback from usability testing, with users appreciating the intuitive design and ease of navigation.

Addressed user-recommended improvements, such as clearer instructions on the voting process and enhanced error messaging.

## 9. Deployment

The deployment of the Sparking\_Loktantra e-voting platform is a crucial phase that transforms the application from a development environment into a publicly accessible platform. The deployment was carried out using GitHub as a repository for version control and collaboration. This section details the deployment strategy, steps involved, and post-deployment considerations.

### 9.1 Deployment Strategy

The deployment strategy for Sparking\_Loktantra was designed to ensure that the application is efficiently managed, easily updated, and accessible to users. Key components of the strategy included:

#### Version Control with GitHub:

Utilized GitHub for version control, enabling collaboration among team members and tracking changes to the codebase.

Maintained separate branches for development, testing, and production to streamline the workflow.

#### Continuous Integration/Continuous Deployment (CI/CD):

Implemented a CI/CD pipeline using GitHub Actions to automate the testing and deployment process.

This ensures that code changes are automatically tested, and upon successful completion, deployed to the live environment.

#### Documentation:

Maintained detailed documentation in the GitHub repository to facilitate onboarding and collaboration among team members.

## 10. Future Scope

The Sparking\_Loktantra e-voting platform is designed with a solid foundation, but there are numerous opportunities for future enhancements and expansions. The following outlines the potential areas for improvement and innovative features

that can be implemented to enhance the platform's functionality, usability, and security:

### 10.1 Enhanced User Experience

#### Mobile Application Development:

Develop a dedicated mobile application for Android and iOS platforms to facilitate easy access to the voting system for users on the go.

Implement push notifications to remind users of upcoming elections and voting deadlines.

#### Multi-Language Support:

Integrate support for multiple languages to cater to a diverse user base, ensuring accessibility for non-English speakers.

Allow users to select their preferred language upon registration or login.

#### Improved User Interface:

Continuously refine the user interface based on user feedback to enhance usability and accessibility.

Implement responsive design techniques to ensure seamless performance on various devices and screen sizes.

### 10.2 Advanced Security Features

#### Biometric Authentication:

Explore the integration of biometric authentication methods (e.g., fingerprint, facial recognition) to enhance user security during login and voting processes.

Utilize secure hardware-based solutions for biometric data processing.

#### Blockchain Technology:

Investigate the feasibility of using blockchain technology to provide a tamper-proof voting system, ensuring transparency and trust in the electoral process.

Implement smart contracts for automating election processes, such as vote tallying and result verification.

#### Enhanced Data Encryption:

Upgrade data encryption methods for both stored and transmitted data to further safeguard user information and voting choices.

Regularly audit security protocols and update them to counter emerging threats.

### 10.3 Feature Expansion

#### Real-Time Vote Tracking:

Implement a real-time voting status feature that allows voters to track the progress of their votes, providing transparency and reassurance.

Offer an option for users to receive confirmation notifications once their vote has been successfully recorded.

#### Election Analytics Dashboard:

Create an analytics dashboard for administrators to monitor voter participation rates, demographic insights, and other key performance indicators.

Use data analytics to identify trends and areas for improvement in future elections.

#### Social Media Integration:

Integrate social media sharing options to encourage users to promote the platform and increase awareness about upcoming elections.

Use social media campaigns to engage younger voters and increase overall participation.

#### 10.4 Community Engagement

##### User Feedback Mechanism:

Establish a robust feedback mechanism to gather user suggestions and experiences, helping the development team prioritize enhancements.

Conduct surveys and focus groups to better understand user needs and expectations.

##### Partnerships with NGOs and Educational Institutions:

Collaborate with non-governmental organizations and educational institutions to promote the platform and educate users about the voting process.

Host workshops and seminars to raise awareness about the importance of voting and how to use the platform effectively.

#### 10.5 Scalability

##### Infrastructure Scalability:

Plan for infrastructure scalability to handle increased traffic during peak voting periods, ensuring consistent performance and availability.

Explore options for auto-scaling on cloud services to dynamically adjust resources based on user demand.

##### Support for Additional Voting Scenarios:

Expand the platform's capabilities to support different types of elections, such as local, state, and national elections, as well as organizational voting.

Allow for custom voting scenarios, such as referendums and polls, to accommodate diverse electoral needs.

## 11. Conclusion

The Sparking\_Loktantra e-voting platform represents a significant advancement in the electoral process, addressing critical challenges faced by voters who are unable to physically attend polling stations due to various circumstances. By

leveraging modern web technologies and a user-centered design, the platform provides a secure, transparent, and accessible solution for online voting.

Through comprehensive features such as user registration, admin verification, and unique voting codes, Sparking\_Loktantra ensures that the voting process is not only streamlined but also robust against potential fraudulent activities. The integration of MySQL for database management and Java-based technologies (Java, JSP, and Java Servlets) supports a reliable and scalable backend, allowing for efficient data handling and processing.

Moreover, the deployment strategy utilizing GitHub for version control and CI/CD practices has facilitated seamless collaboration among team members, ensuring that the application remains up-to-date and functional in a live environment. Post-deployment considerations, including monitoring and user feedback mechanisms, further reinforce the platform's commitment to continuous improvement.

Looking ahead, the future scope of Sparking\_Loktantra is promising, with opportunities for enhancements in user experience, security, and scalability. The potential for mobile application development, advanced security features like biometric authentication, and community engagement initiatives will help broaden the platform's reach and effectiveness in serving voters.

In conclusion, Sparking\_Loktantra is poised to revolutionize the voting experience for users, fostering greater participation and engagement in the democratic process. By continually evolving and adapting to user needs and technological advancements, the platform aims to contribute to a more inclusive and accessible electoral system, ultimately empowering citizens to exercise their right to vote with confidence and ease.

## Appendix A Project Code

### Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="icon" href="Logo.png" type="image/x-icon" />
  <link rel="stylesheet" href="index.css">
  <link rel="stylesheet" href="styles.css">
  <title>E_voting</title>
  <!-- Bootstrap 5 CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
```

```

<!-- Header Section -->
<section>
  <nav class="navbar navbar-expand-lg navbar-light bg-primary">
    <div class="container-fluid">
      <a class="navbar-brand" href="index.html"></a>
      <div class="portal text-white">
        <p>E-मतदाता सेवा पोर्टल</p>
        <p>E-VOTER'S SERVICE PORTAL</p>
      </div>
      <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNavAltMarkup">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
        <div class="navbar-nav ms-auto text-center mx-5 fs-5">
          <!-- <a class="nav-link mx-3 text-white"
href="index.html">Home</a> -->
          <a class="nav-link mx-3 text-white"
href="Login.jsp">Login</a>
          <a class="nav-link mx-3 text-white"
href="userVerification.jsp">Vote</a>
          <a class="nav-link mx-3 text-white"
href="adminLogin.jsp">Admin</a>
        </div>
      </div>
    </div>
  </nav>
</section>

<!-- Carousel Section Start -->
<div id="demo" class="carousel slide" data-bs-ride="carousel">

  <!-- Indicators/dots -->
  <div class="carousel-indicators">
    <button type="button" data-bs-target="#demo" data-bs-slide-to="0"
class="active"></button>
    <button type="button" data-bs-target="#demo" data-bs-slide-to="1"></button>
    <button type="button" data-bs-target="#demo" data-bs-slide-to="2"></button>
  </div>

  <!-- The slideshow/carousel -->
  <div class="carousel-inner">
    <div class="carousel-item active">
      
      <div class="carousel-caption d-flex flex-column justify-content-center
text-center text-black poster-contant1" style="top: 50%; transform: translateY(-
50%);">
        <h1>Importance of E-voting</h1>
        <h3>Enhanced Efficiency and Accuracy:</h3>
        <p>E-voting streamlines the voting process, providing quicker and more
accurate results, reducing the potential for human error, and minimizing the need
for extensive manual counting.</p>
      </div>
    </div>
    <div class="carousel-item">
      

```



```

        <div class="carousel-caption d-flex flex-column justify-content-center
text-center text-black poster-contant1" style="top: 50%; transform: translateY(-
50%);">
            <h1>Importance of E-voting</h1>
            <h3>Remote Voting:</h3>
            <p>Allows voters to cast their ballots from anywhere, which is
particularly beneficial for those who are abroad, disabled, or unable to visit
polling stations</p>
        </div>
    </div>
    <div class="carousel-item">
        
        <div class="carousel-caption d-flex flex-column justify-content-center
text-center text-black poster-contant1" style="top: 50%; transform: translateY(-
50%);">
            <h1>Importance of E-voting</h1>
            <h3>Accessibility and Convenience</h3>
            <p>Increasing Voter Participation: E-voting can make the voting process
more accessible to people who might have difficulties voting in person, such as
those with disabilities, those living abroad, or those with time constraints.</p>
        </div>
    </div>
</div>

<!-- Left and right controls/icons -->
<button class="carousel-control-prev" type="button" data-bs-target="#demo"
data-bs-slide="prev">
    <span class="carousel-control-prev-icon"></span>
</button>
<button class="carousel-control-next" type="button" data-bs-target="#demo"
data-bs-slide="next">
    <span class="carousel-control-next-icon"></span>
</button>

</div>
<!-- Carousel Section End -->

<h1 class="text-center my-5">Our Features</h1>
<!-- our features start -->
    <!-- Card Section -->
    <div class="container my-5">
        <div class="row">
            <!-- Card 1 -->
            <div class="col-lg-3 col-md-6 mb-4">
                <div class="card h-100 text-center">
                    
                    <div class="card-body">
                        <h5 class="card-title">Transparency and Trust</h5>
                        <p class="card-text">Our e-voting system ensures transparency and
trust through secure, verifiable processes and clear audit trails.</p>
                    </div>
                </div>
            </div>
            <!-- Card 2 -->
            <div class="col-lg-3 col-md-6 mb-4">
                <div class="card h-100 text-center">

```

```

        
        <div class="card-body">
            <h5 class="card-title">Real Time Monitoring</h5>
            <p class="card-text">Our e-voting system enables real-time monitoring
to ensure immediate visibility and accountability throughout the voting
process.</p>
        </div>
    </div>
</div>
<!-- Card 3 -->
<div class="col-lg-3 col-md-6 mb-4">
    <div class="card h-100 text-center">
        
        <div class="card-body">
            <h5 class="card-title">public Trust</h5>
            <p class="card-text">Public trust is a fundamental idea in democracy,
which is based on the belief that the public holds the power and future of a
society.</p>
        </div>
    </div>
</div>
</div>
<!-- Card 4 -->
<div class="col-lg-3 col-md-6 mb-4">
    <div class="card h-100 text-center">
        
        <div class="card-body">
            <h5 class="card-title">Quick Results</h5>
            <p class="card-text">Our e-voting system delivers quick results while
maintaining accuracy and integrity.</p>
        </div>
    </div>
</div>
</div>
</div>
</div>
<!-- our features end -->

<h1 class="text-center my-5">Our Election Commission of India</h1>

<!-- review section start -->
<div class="clientss" id="clients">

    <div class="carousel slide container" id="my-clients" data-bs-ride="carousel"
data-bs-interval="2000">
        <div class="carousel-inner">
            <div class="carousel-item active text-center">
                
                <h3>Shri Rajiv Kumar</h3>
                <p>Chief Election Commissioner</p>

                <i class="fa-brands fa-facebook-f social-icon"></i>
                <i class="fab fa-twitter social-icon"></i>
                <i class="fab fa-google social-icon"></i>
            </div>

```

```

        <div class="carousel-item text-center">
            
            <h3>Shri Gyanesh Kumar</h3>
            <p>Election Commissioner</p>

            <i class="fa-brands fa-facebook-f social-icon"></i>
            <i class="fab fa-twitter social-icon"></i>
            <i class="fab fa-google social-icon"></i>
        </div>

        <div class="carousel-item text-center">
            
            <h3>Dr Sukhbir Singh Sandhu</h3>
            <p>Election Commissioner</p>

            <i class="fa-brands fa-facebook-f social-icon"></i>
            <i class="fab fa-twitter social-icon"></i>
            <i class="fab fa-google social-icon"></i>
        </div>
    </div>

    <a href="#my-clients" role="button" class="carousel-control-prev cl-btn"
data-bs-slide="prev">
        <i class="fas fa-chevron-left"></i>
    </a>

    <a href="#my-clients" role="button" class="carousel-control-next cl-btn"
data-bs-slide="next">
        <i class="fas fa-chevron-right"></i>
    </a>
</div>

</div>
<!-- review section end -->

<!-- Footer Start -->
<footer>
    <div class="container-fluid">
        <div class="row">
            <!-- Information Section -->
            <div class="col-md-4 footer-section">
                <h5>Digital Democracy</h5>
                <p>Digital Democracy refers to the use of digital technologies and the
internet to enhance or facilitate democratic processes. It allows citizens to
engage more directly in political decision-making through online voting,
petitions, and public forums.
                Digital democracy is an important part of the global democracy
movement.</p>
                <!-- <p><a href="#">Learn More</a></p> -->
            </div>

            <!-- About Us Section -->
            <div class="col-md-4 footer-section">
                <h5>Importance of E-voting</h5>
                <p>Increased Accessibility</p>
                <p>Faster Results:</p>

```

```

        <p>Improved Accuracy</p>
        <p>Cost-Effectiveness</p>
        <!-- <p><a href="#">Read Our Story</a></p> -->
    </div>

    <!-- Contact Section -->
    <div class="col-md-4 footer-section footer-contact">
        <h5>Contact</h5>
        <p><a href="mailto:info@example.com">1950</a> Toll Free</p>
        <p><a href="tel:+1234567890">23052220, 23052221</a></p>
        <p> Nirvachan Sadan, Ashoka Road, New Delhi 110001
            complaints[at]eci[dot]gov[dot]in</p>
    </div>
</div>

    <p>Copyright &copy; 2024. All Rights Reserved By Tech-Gang</p>
</footer>
<!-- Footer End -->
<!-- Bootstrap 5 JS (optional) -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"
></script>

<!-- CHAT BOOT OPENING -->
<script>
    window.watsonAssistantChatOptions = {
        integrationID: "85005c21-2dca-4da9-9f0c-7a15e460aa29", // The ID of this
integration.
        region: "au-syd", // The region your integration is hosted in.
        serviceInstanceID: "338fab45-f4d6-4e18-a00c-6bbd9fe521d2", // The ID of your
service instance.
        onLoad: async (instance) => { await instance.render(); }
    };
    setTimeout(function(){
        const t=document.createElement('script');
        t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
        document.head.appendChild(t);
    });
</script>

<!-- CHAT BOOT CLOSING -->
</body>
</html>

```

## Registration.jsp

```

<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-
8"%>
<%@ page import="jakarta.servlet.http.HttpSession" %>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" href="Logo.png" type="image/x-icon" />

```

```

<!-- Bootstrap CSS -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
<!-- Custom CSS -->
<link rel="stylesheet" href="styles.css">
<title>Register</title>
<style>
  html, body {
    height: 100%;
    margin: 0;
  }

  body {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    font-family: 'Roboto', sans-serif;
    background-image: url('background_img.jpg');
    background-position: center;
    background-size: cover;
  }

  .signup-section {
    margin-top: 50px;
  }

  .form-title {
    font-size: 28px;
    font-weight: 700;
    margin-bottom: 30px;
    text-align: center;
  }

  .form-group input {
    height: 45px;
    font-size: 16px;
  }

  .form-button input {
    width: 100%;
    height: 50px;
    font-size: 18px;
  }

  .signup-form {
    padding: 2rem;
    background-color: #e3f2fd; /* Light grey background */
    border-radius: 15px;
  }

  /* Footer Styling */
  footer {
    background-color: #333;
    color: #fff;
    padding: 1rem;
    text-align: center;
  }

  footer p {
    margin: 0;
    font-size: 1rem;
  }

```

```

</style>
</head>
<body>
    <!-- Header Section -->
    <section>
        <nav class="navbar navbar-expand-lg navbar-light bg-primary">
            <div class="container-fluid">
                <a class="navbar-brand" href="index.html">
                    
                </a>
                <div class="portal text-white">
                    <p>E-मतदाता सेवा पोर्टल</p>
                    <p>E-VOTER'S SERVICE PORTAL</p>
                </div>
                <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNavAltMarkup">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
                    <div class="navbar-nav ms-auto text-center mx-5 fs-5">
                        <a class="nav-link mx-3 text-white"
href="index.html">Home</a>
                        <a class="nav-link mx-3 text-white"
href="Login.jsp">Login</a>
                        <a class="nav-link mx-3 text-white"
href="userVerification.jsp">Vote</a>
                        <a class="nav-link mx-3 text-white"
href="adminLogin.jsp">Admin</a>
                    </div>
                </div>
            </div>
        </nav>
    </section>
    <!-- Hidden status for SweetAlert -->
    <input type="hidden" id="status" value="<%= request.getAttribute("status")
%>">

    <!-- Registration Form Section -->
    <div class="container signup-section">
        <div class="row justify-content-center">
            <div class="col-lg-6 mb-3">
                <div class="signup-form">
                    <h2 class="form-title">Sign Up</h2>
                    <!-- Main Registration Form -->
                    <form method="post" action="register">
                        <input type="hidden" name="action"
value="submitRegistration">
                        <div class="form-group mb-3">
                            <label for="name">Full Name</label>
                            <input type="text" name="name" id="name" class="form-
control" placeholder="Your Name" required style="text-transform: uppercase;">
                        </div>
                        <div class="form-group mb-3">
                            <label for="email">Email</label>
                            <input type="email" name="email" id="email"
class="form-control" placeholder="Email" required>
                        </div>
                        <div class="form-group mb-3">
                            <label for="contact">Mobile Number</label>

```

```

        <input type="text" name="contact" id="contact"
class="form-control" placeholder="Mobile" required>
    </div>
    <div class="form-group mb-3">
        <label for="pass">Password</label>
        <input type="password" name="pass" id="pass"
class="form-control" placeholder="Password" required>
    </div>
    <div class="form-group mb-3">
        <label for="re_pass">Confirm password</label>
        <input type="password" name="re_pass" id="re_pass"
class="form-control" placeholder="Confirm password" required>
    </div>
    <div class="form-group form-button">
        <input type="submit" name="register" id="register"
class="btn btn-primary btn-lg w-100" value="Submit">
    </div>
    </form>
</div>
</div>
</div>
</div>
<!-- Footer -->
<footer class="bg-dark text-white py-4">
    <div class="container text-center">
        <p>&copy; 2024 E-Voter Service Portal. All Rights Reserved.</p>
    </div>
</footer>

<!-- Bootstrap and JS -->
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>
<!-- SweetAlert JS -->
<script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>

<!-- SweetAlert for registration status -->
<script type="text/javascript">
var status = document.getElementById("status").value;
if (status === "success") {
    swal("Congrats", "Registration Successful", "success");
} else if (status === "emailExists") {
    swal("Oops", "This email is already registered!", "error");
} else if (status === "invalidOtp") {
    swal("Oops", "Invalid OTP, please try again.", "error");
} else if (status === "passwordMismatch") {
    swal("Oops", "Passwords do not match.", "error");
} else if (status === "failed") {
    swal("Oops", "Registration Failed. Please try again!", "error");
}
</script>
</body>
</html>

```

**RegistrationServlet.java**

```
package com.e_voting.registration;
```

```
import jakarta.servlet.RequestDispatcher;  
import jakarta.servlet.ServletException;  
import jakarta.servlet.annotation.WebServlet;  
import jakarta.servlet.http.HttpServlet;  
import jakarta.servlet.http.HttpServletRequest;  
import jakarta.servlet.http.HttpServletResponse;  
import jakarta.servlet.http.HttpSession;  
import org.mindrot.jbcrypt.BCrypt;
```

```
import java.io.IOException;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.PreparedStatement;  
import java.sql.ResultSet;  
import java.sql.SQLException;  
import java.util.Properties;  
import java.util.Random;  
import jakarta.mail.Message;  
import jakarta.mail.MessagingException;  
import jakarta.mail.PasswordAuthentication;  
import jakarta.mail.Session;  
import jakarta.mail.Transport;  
import jakarta.mail.internet.InternetAddress;  
import jakarta.mail.internet.MimeMessage;
```

```
@WebServlet("/register")
```

```
public class RegistrationServlet extends HttpServlet {  
    private static final long serialVersionUID = 1L;
```



```
protected void doPost(HttpServletRequest request, HttpServletResponse  
response) throws ServletException, IOException {
```

```
    HttpSession session = request.getSession();
```

```
    String action = request.getParameter("action");
```

```
    if ("submitRegistration".equals(action)) {
```

```
        handleRegistration(request, response, session);
```

```
    } else if ("verifyOtp".equals(action)) {
```

```
        verifyOtp(request, response, session);
```

```
    }
```

```
}
```

```
private void handleRegistration(HttpServletRequest request, HttpServletResponse response, HttpSession session) throws  
ServletException, IOException {
```

```
    String uname = request.getParameter("name").toUpperCase();
```

```
    String uemail = request.getParameter("email");
```

```
    String upwd = request.getParameter("pass");
```

```
    String re_upwd = request.getParameter("re_pass");
```

```
    String umobile = request.getParameter("contact");
```

```
    RequestDispatcher dispatcher = null;
```

```
    Connection con = null;
```

```
    if (!upwd.equals(re_upwd)) {
```

```
        request.setAttribute("status", "passwordMismatch");
```

```
        dispatcher = request.getRequestDispatcher("registration.jsp");
```

```
        dispatcher.forward(request, response);
```

```
        return;
```

```
    }
```

```

try {
    Class.forName("com.mysql.cj.jdbc.Driver");

    con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/E_voting?useSSL=
false", "root", "Rajiv@2023");

    // Check if the email is already registered
    PreparedStatement checkUserStmt = con.prepareStatement("SELECT *
FROM registration WHERE uemail = ?");
    checkUserStmt.setString(1, uemail);
    ResultSet rs = checkUserStmt.executeQuery();

    if (rs.next()) {
        request.setAttribute("status", "emailExists");
        dispatcher = request.getRequestDispatcher("registration.jsp");
    } else {
        // Generate OTP and send it to the user
        String otp = generateOtp();
        session.setAttribute("otp", otp);
        session.setAttribute("uname", uname);
        session.setAttribute("uemail", uemail);
        session.setAttribute("upwd", upwd); // Store the plain text password
for later use
        session.setAttribute("umobile", umobile);

        sendOtpEmail(uemail, otp);

        // Redirect to OTP verification page
        response.sendRedirect("regOtpVerify.jsp");
        return;
    }
    dispatcher.forward(request, response);
}

```

```

    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        if (con != null) {
            try {
                con.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

**private void verifyOtp(HttpServletRequest request, HttpServletResponse response, HttpSession session) throws ServletException, IOException {**

**String enteredOtp = request.getParameter("otp");**

**String sessionOtp = (String) session.getAttribute("otp");**

**// System.out.println("Entered OTP: " + enteredOtp);**

**// System.out.println("Session OTP: " + sessionOtp);**

**if (enteredOtp.equals(sessionOtp)) {**

**// If OTP is correct, save user to the database**

**//System.out.println("OTP Verified Successfully");**

**try {**

**Class.forName("com.mysql.cj.jdbc.Driver");**

**Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/E\_voting?useSSL=false", "root", "Rajiv@2023");**

**String uname = (String) session.getAttribute("uname");**

```
String upwd = BCrypt.hashpw((String) session.getAttribute("upwd"),  
BCrypt.gensalt()); // Hashed password
```

```
String originalPwd = (String) session.getAttribute("upwd"); // Plain  
text password
```

```
String uemail = (String) session.getAttribute("uemail");
```

```
String umobile = (String) session.getAttribute("umobile");
```

```
PreparedStatement pst = con.prepareStatement("INSERT INTO  
registration(uname, upwd, uemail, umobile) VALUES(?,?,?,?)");
```

```
pst.setString(1, uname);
```

```
pst.setString(2, upwd); // Store the hashed password in the database
```

```
pst.setString(3, uemail);
```

```
pst.setString(4, umobile);
```

```
int rowCount = pst.executeUpdate();
```

```
if (rowCount > 0) {
```

```
    // Registration successful, send confirmation email
```

```
    sendConfirmationEmail(uemail, originalPwd, uname);
```

```
    // Redirect to login.jsp with a success message
```

```
    session.setAttribute("isRegistered", true); // Set this attribute on  
successful registration
```

```
    response.sendRedirect("login.jsp?message=success");
```

```
    return;
```

```
} else {
```

```
    request.setAttribute("status", "failed");
```

```
    RequestDispatcher dispatcher =  
request.getRequestDispatcher("regOtpVerify.jsp");
```

```
    dispatcher.forward(request, response);
```

```
}
```

```
} catch (Exception e) {
```

```
    e.printStackTrace();
```

```

    }
} else {
    //System.out.println("OTP Mismatch");
    request.setAttribute("statusMessage", "Invalid OTP. Please try again.");

    RequestDispatcher dispatcher =
request.getRequestDispatcher("regOtpVerify.jsp");
    dispatcher.forward(request, response);
}
}

```

```

private String generateOtp() {
    Random random = new Random();
    int otp = 100000 + random.nextInt(900000); // Generate a 6-digit OTP
    return String.valueOf(otp);
}

```

```

private void sendOtpEmail(String recipientEmail, String otp) {
    final String senderEmail = "evotingproject2024@gmail.com";
    final String senderPassword = "osmn ioqe yavy wuqv";

```

```

    Properties props = new Properties();
    props.put("mail.smtp.host", "smtp.gmail.com");
    props.put("mail.smtp.port", "587");
    props.put("mail.smtp.auth", "true");
    props.put("mail.smtp.starttls.enable", "true");

```

```

    Session session = Session.getInstance(props, new
jakarta.mail.Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication() {
            return new PasswordAuthentication(senderEmail, senderPassword);
        }
    }

```

```

    });

    try {
        Message message = new MimeMessage(session);
        message.setFrom(new InternetAddress(senderEmail));
        message.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(recipientEmail));
        message.setSubject("Your OTP for Registration");
        message.setText("Dear User,\n\nYour OTP for registration is: " + otp +
"\n\nBest regards,\nE-Voting Team");

        Transport.send(message);
    } catch (MessagingException e) {
        e.printStackTrace();
    }
}

private void sendConfirmationEmail(String recipientEmail, String
originalPwd, String uname) {
    final String senderEmail = "evotingproject2024@gmail.com";
    final String senderPassword = "osmn ioqe yavy wuqv";

    Properties props = new Properties();
    props.put("mail.smtp.host", "smtp.gmail.com");
    props.put("mail.smtp.port", "587");
    props.put("mail.smtp.auth", "true");
    props.put("mail.smtp.starttls.enable", "true");

    Session session = Session.getInstance(props, new
jakarta.mail.Authenticator() {
        protected PasswordAuthentication getPasswordAuthentication() {
            return new PasswordAuthentication(senderEmail, senderPassword);
        }
    });

```

```

    }
});

try {
    Message message = new MimeMessage(session);
    message.setFrom(new InternetAddress(senderEmail));
    message.setRecipients(Message.RecipientType.TO,
InternetAddress.parse(recipientEmail));

    message.setSubject("Registration    Successful    -    Your    E-Voter
Credentials");

    // Construct the email body with login information (plain text password)
    String emailContent = "Dear " + uname + ",\n\n"
        + "Thank you for registering with the E-Voter Service Portal.\n"
        + "Your login details are as follows:\n"
        + "Email: " + recipientEmail + "\n"
        + "Password: " + originalPwd + "\n\n"

        + "Best regards,\nE-Voting Team";

    message.setText(emailContent);

    Transport.send(message);
} catch (MessagingException e) {
    e.printStackTrace();
}
}
}

```

## Login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%
    // Disable caching
    response.setHeader("Cache-Control", "no-cache, no-store, must-revalidate"); //
    HTTP 1.1.
    response.setHeader("Pragma", "no-cache"); // HTTP 1.0.
    response.setDateHeader("Expires", 0); // Proxies.
%>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" href="Logo.png" type="image/x-icon" />
    <!-- Bootstrap CSS -->
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet">
    <!-- Custom CSS -->
    <link rel="stylesheet" href="styles.css">

    <title>Login</title>
    <style>
        html, body {
            height: 100vh;
            margin: 0;
        }

        body {
            display: flex;
            flex-direction: column;
            justify-content: space-between;
            background-image: url(background_img.jpg);
            background-position: center;
            background-repeat: no-repeat;
            background-size: cover;

            font-family: 'Roboto', sans-serif;
        }

        .main {
            flex-grow: 1;
            display: flex;
            justify-content: center;
            border-radius: 15px;
            align-items: center;
        }

        .signin-form {
            max-width: 400px;
            width: 100%;
            background-color: #e3f2fd;
            border-radius: 15px;
        }

        h2 {
```



```

        font-weight: bold;
        font-size: 1.8rem;
    }

    .form-group input {
        height: 45px;
        font-size: 16px;
    }

    .form-button input {
        width: 100%;
        height: 50px;
        font-size: 18px;
    }
    form a {
        text-decoration: none;
        color: black;
    }

    footer {
        background-color: #333;
        color: #fff;
        padding: 1rem;
        text-align: center;
    }

    footer p {
        margin: 0;
        font-size: 1rem;
    }
</style>
</head>
<body>
<input type="hidden" id="status" value="<%= request.getAttribute("status") != null
? request.getAttribute("status") : "" %>">

    <!-- Header Section -->
    <section>
        <nav class="navbar navbar-expand-lg navbar-light bg-primary">
            <div class="container-fluid">
                <a class="navbar-brand" href="index.html"></a>
                <div class="portal text-white">
                    <p>E-मतदाता सेवा पोर्टल</p>
                    <p>E-VOTER'S SERVICE PORTAL</p>
                </div>
                <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNavAltMarkup">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
                    <div class="navbar-nav ms-auto text-center mx-5 fs-5">
                        <a class="nav-link mx-3 text-white"
href="index.html">Home</a>
                        <a class="nav-link mx-3 text-white"
href="registration.jsp">Register</a>
                        <a class="nav-link mx-3 text-white"
href="userVerification.jsp">Vote</a>

```

```

        <a class="nav-link mx-3 text-white"
href="adminLogin.jsp">Admin</a>
    </div>
</div>
</div>
</nav>
</section>
<!-- sign in form design -->
    <div class="main">
        <div class="signin-form p-5">
            <h2 class="text-center mb-4 text-title">Sign In</h2>

            <!-- Success Message -->
            <% if
("loggedout".equals(request.getParameter("message")))> { %>
                <div class="alert alert-success text-center">
                    You have successfully logged out.
                </div>
            <% } %>

            <!-- Error Message -->
            <% if
("failed".equals(request.getAttribute("status")))> { %>
                <div class="alert alert-danger text-center">
                    Invalid username or password. Please try
again.
                </div>
            <% } %>

            <form method="post" action="login" class="register-
form" id="login-form">
                <div class="form-group mb-3">
                    <label for="username">Email Address</label>
                    <input type="email" name="username"
id="username" class="form-control" placeholder="Enter Your Email" required>
                </div>
                <div class="form-group mb-3">
                    <label for="password">Password</label>
                    <input type="password" name="password"
id="password" class="form-control" placeholder="Password" required>
                </div>
                <a href="forgotPassword.jsp" class="d-block mb-
3">Forgot Password</a>
                <div class="form-group form-button">
                    <input type="submit" name="signin" id="signin"
class="btn btn-primary btn-lg w-100" value="Log in">
                </div>
                <a href="registration.jsp" class="d-block mt-3
text-end">Create new account </a>
            </form>
        </div>
    </div>

    <!-- Footer -->
    <footer class="bg-dark text-white py-4">
        <div class="container text-center">
            <p>&copy; 2024 E-Voter Service Portal. All Rights Reserved.</p>
        </div>
    </footer>

```

```

    <!-- JS Scripts -->
    <script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
></script>

    <!-- SweetAlert Script -->
    <script type="text/javascript">
        document.addEventListener('DOMContentLoaded', function() {
            var status = document.getElementById("status").value;
            var urlParams = new URLSearchParams(window.location.search);
            var message = urlParams.get('message');

            if (status === "resetSuccess") {
                swal("Congrats", "Password reset successfully. You can now log in
with your new password!", "success");
            } else if (status === "resetFailed") {
                swal("Oops", "Password reset failed. Please try again!", "error");
            } else if (status === "passwordMismatch") {
                swal("Oops", "Passwords do not match. Please try again!", "error");
            } else if (message === "success") {
                swal("Congrats", "Registration successful! Please log in.",
"success");
            }
        });
    </script>
</body>
</html>

```

## Login.java

```
package com.e_voting.registration;
```

```

import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
import org.mindrot.jbcrypt.BCrypt;

import java.io.IOException;
import java.sql.Connection;

```

```

import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

@WebServlet("/login")
public class Login extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
        String uemail = request.getParameter("username");
        String upwd = request.getParameter("password");
        HttpSession session = request.getSession(); // Get the session (create if
not exists)

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");

            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/E_voting?useSSL=
false", "root", "Rajiv@2023");

            // Fetch the hashed password and username from the database based
on the provided email

            PreparedStatement pst = con.prepareStatement("SELECT uname, upwd
FROM registration WHERE uemail = ?");
            pst.setString(1, uemail);

            ResultSet rs = pst.executeQuery();

            if (rs.next()) {

```

```

String storedHashedPwd = rs.getString("upwd");
String uname = rs.getString("uname");

// Compare the entered password with the stored hashed password
if (BCrypt.checkpw(upwd, storedHashedPwd)) {
    // Password match, login successful
    session.setAttribute("name", uname);

    // Prevent caching
    response.setHeader("Cache-Control", "no-cache, no-store, must-revalidate"); // HTTP 1.1
    response.setHeader("Pragma", "no-cache"); // HTTP 1.0
    response.setDateHeader("Expires", 0); // Proxies

    // Use redirect instead of forward to implement PRG pattern
    response.sendRedirect("welcome.jsp");
} else {
    // Password does not match
    request.setAttribute("status", "failed");

    RequestDispatcher dispatcher =
request.getRequestDispatcher("login.jsp");
    dispatcher.forward(request, response);
}
} else {
    // Email not found
    request.setAttribute("status", "failed");

    RequestDispatcher dispatcher =
request.getRequestDispatcher("login.jsp");
    dispatcher.forward(request, response);
}
} catch (Exception e) {

```

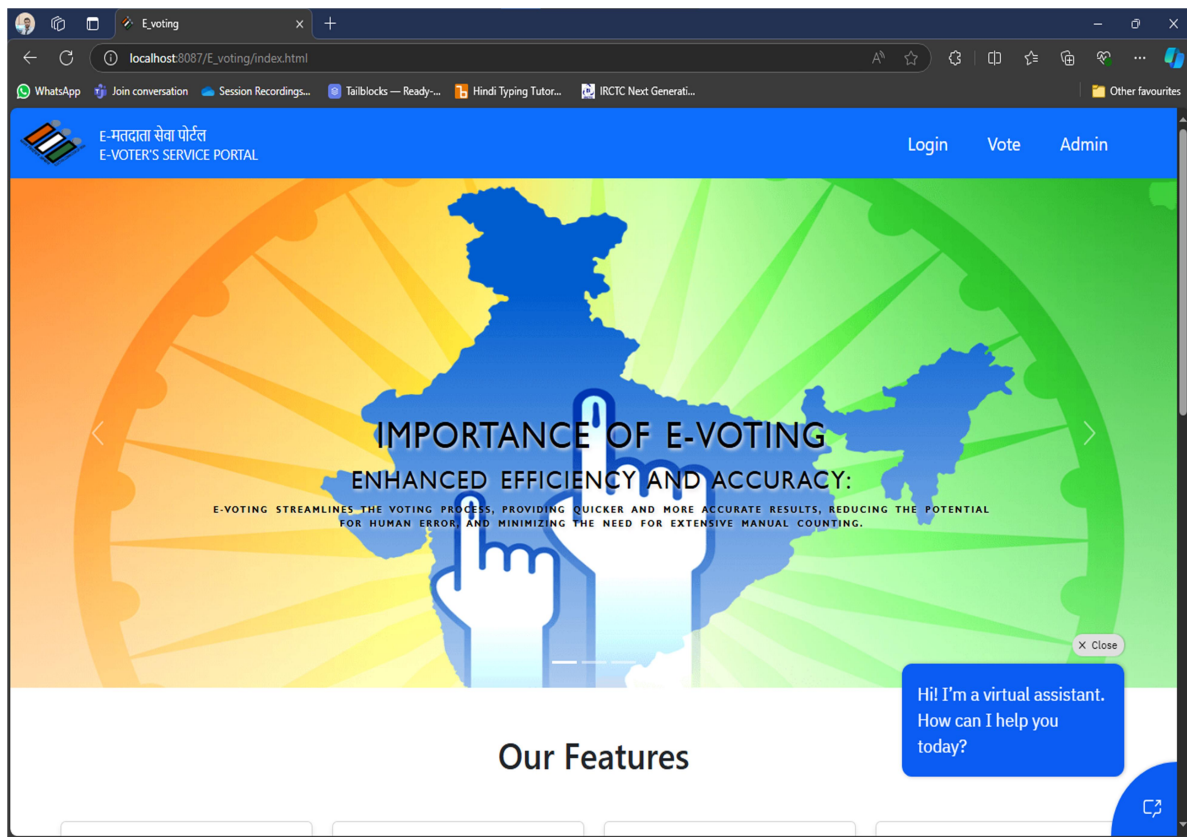
```
e.printStackTrace();  
}  
}  
}
```

For more source code visit the github link :-

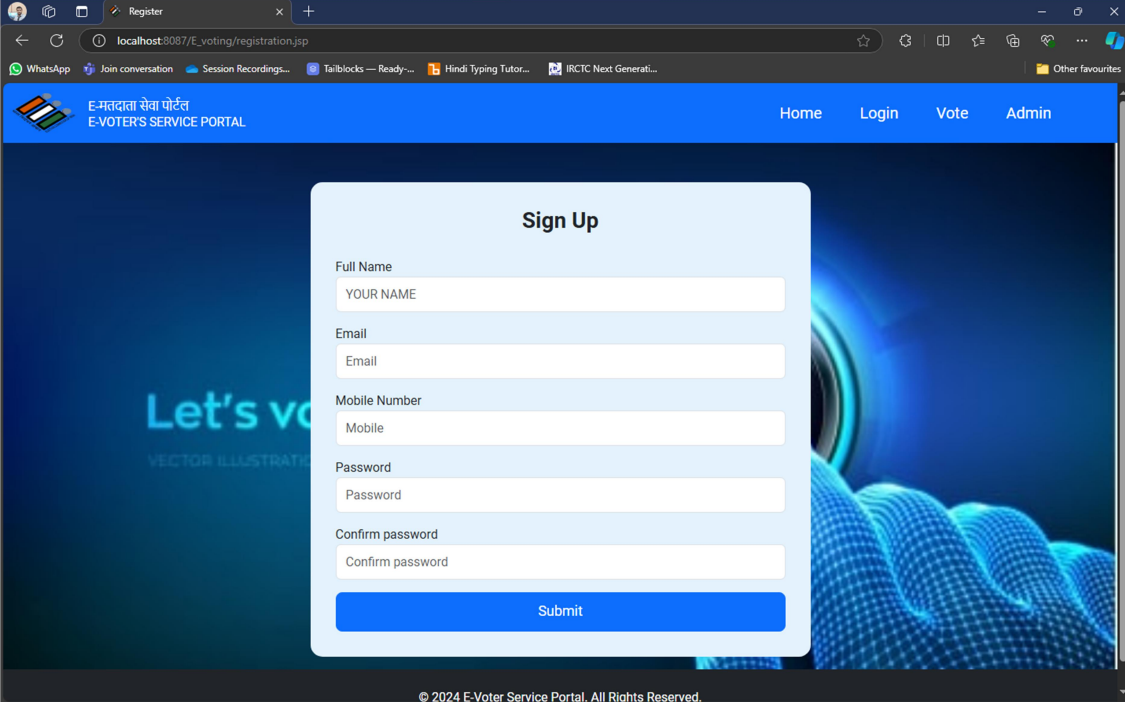
<https://github.com/Rajivkumar2023/E-voting.git>

## Appendix B Screenshot of Project

### Home page



## Registration page



The screenshot shows a web browser window with the URL `localhost:8087/E_voting/registration.jsp`. The page features a blue header with the text "E-मतदाता सेवा पोर्टल" and "E-VOTER'S SERVICE PORTAL" on the left, and navigation links "Home", "Login", "Vote", and "Admin" on the right. The main content area has a dark blue background with a glowing blue hand reaching towards a circular button labeled "VOTE". In the center, there is a white "Sign Up" form with the following fields: "Full Name" (placeholder: YOUR NAME), "Email" (placeholder: Email), "Mobile Number" (placeholder: Mobile), "Password" (placeholder: Password), and "Confirm password" (placeholder: Confirm password). A blue "Submit" button is at the bottom of the form. The footer contains the text "© 2024 E-Voter Service Portal. All Rights Reserved."

Register

localhost:8087/E\_voting/registration.jsp

WhatsApp Join conversation Session Recordings... Tailblocks — Ready... Hindi Typing Tutor... IRCTC Next Generati...

E-मतदाता सेवा पोर्टल  
E-VOTER'S SERVICE PORTAL

Home Login Vote Admin

### Sign Up

Full Name  
YOUR NAME

Email  
Email

Mobile Number  
Mobile

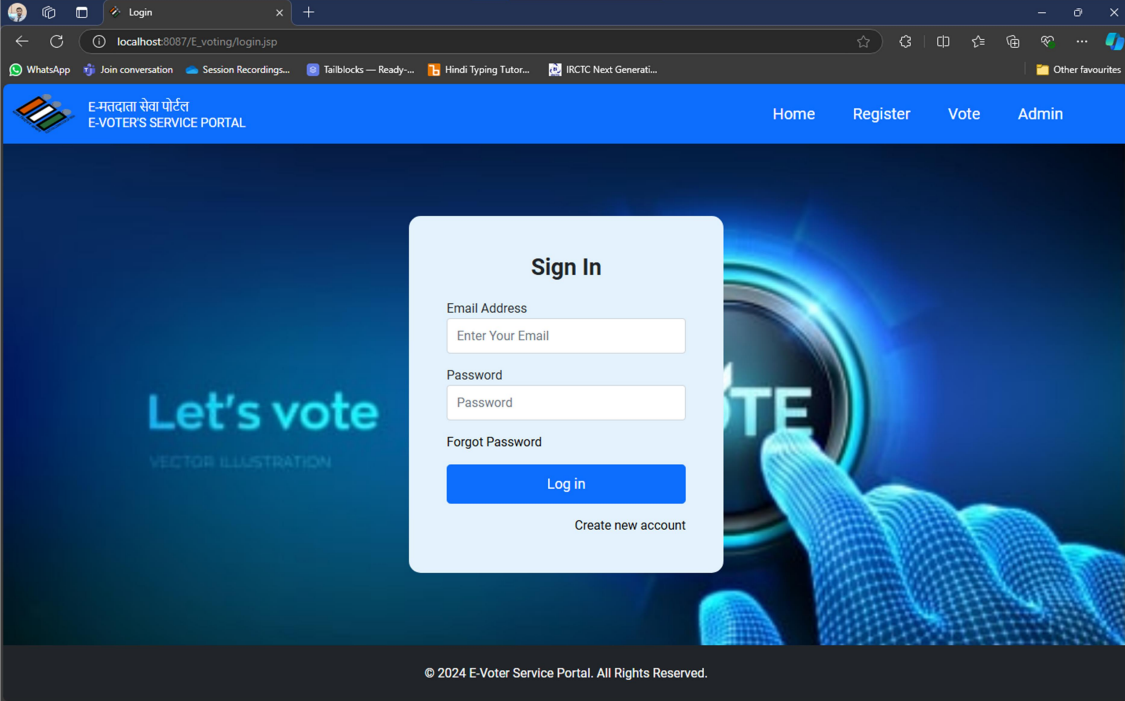
Password  
Password

Confirm password  
Confirm password

Submit

© 2024 E-Voter Service Portal. All Rights Reserved.

## Login page



The screenshot shows a web browser window with the URL `localhost:8087/E_voting/login.jsp`. The page features a blue header with the text "E-मतदाता सेवा पोर्टल" and "E-VOTER'S SERVICE PORTAL" on the left, and navigation links "Home", "Register", "Vote", and "Admin" on the right. The main content area has a dark blue background with a glowing blue hand reaching towards a circular button labeled "VOTE". In the center, there is a white "Sign In" form with the following fields: "Email Address" (placeholder: Enter Your Email) and "Password" (placeholder: Password). Below the password field is a link "Forgot Password". A blue "Log in" button is at the bottom of the form, and a link "Create new account" is below it. The footer contains the text "© 2024 E-Voter Service Portal. All Rights Reserved."

Login

localhost:8087/E\_voting/login.jsp

WhatsApp Join conversation Session Recordings... Tailblocks — Ready... Hindi Typing Tutor... IRCTC Next Generati...

E-मतदाता सेवा पोर्टल  
E-VOTER'S SERVICE PORTAL

Home Register Vote Admin

### Sign In

Email Address  
Enter Your Email

Password  
Password

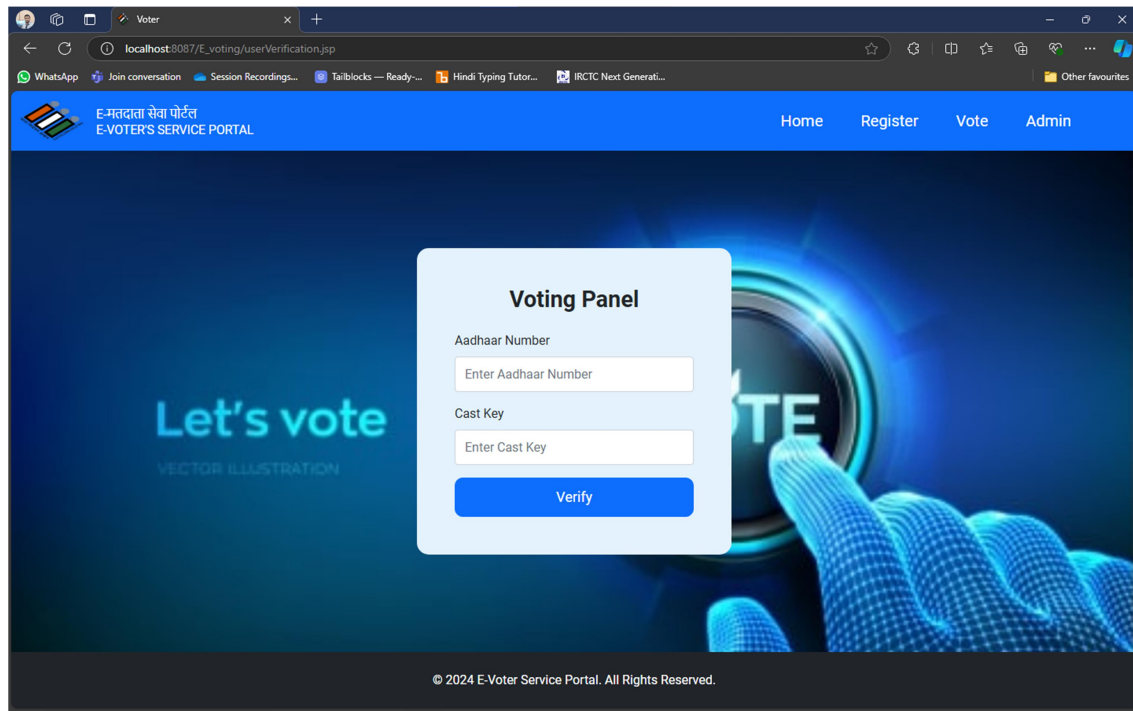
[Forgot Password](#)

Log in

[Create new account](#)

© 2024 E-Voter Service Portal. All Rights Reserved.

## Voting page



## Appendix C abbreviation

Here is a list of commonly used abbreviations relevant to the Sparking\_Loktantra e-voting platform project:

Abbreviation	Full Form
API	Application Programming Interface
CI	Continuous Integration
CD	Continuous Deployment
DB	Database
EC2	Elastic Compute Cloud (AWS)
GUI	Graphical User Interface
HTTPS	Hypertext Transfer Protocol Secure
JS	JavaScript
JSP	JavaServer Pages
MySQL	My Structured Query Language
RDS	Relational Database Service



Abbreviation	Full Form
SQL	Structured Query Language
UI	User Interface
WAR	Web Application Archive
AWS	Amazon Web Services
OOP	Object-Oriented Programming
SDK	Software Development Kit
SSL	Secure Sockets Layer
JVM	Java Virtual Machine
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets