

Online voting system



Uzair Hussain-057

Muhammad anees-033

hamza sattar-014

**Supervisor**

[Dr. Waqas]

**Course**

Software Construction and Development (SCD)

**Project Proposal – Online Voting System**

**1. Introduction**

Voting is a fundamental democratic process that allows citizens to choose their representatives and express their opinions. Traditional paper-based voting systems are time-consuming, prone to errors, and vulnerable to manipulation. In the digital age, web-based online voting systems provide a more reliable, efficient, and transparent method of conducting elections.

This project aims to develop a Java-based Online Voting System using JSP, Servlets, and JDBC (MySQL). The system will allow administrators to create and manage elections while enabling voters to securely cast their votes online.

**2. Problem Statement**

Traditional voting methods suffer from:

* Manual counting errors.
* High administrative costs (paper, printing, and staff).
* Long queues at polling stations.
* Lack of transparency and delayed results.
* Risk of duplicate or fraudulent voting.

There is a strong need for a **secure, transparent, and user-friendly web-based voting system** that eliminates these drawbacks.

**3. Objectives**

The main objectives of this project are:

1. To develop a **secure online platform** for conducting elections.
2. To enable **admins** to create, update, and manage elections.
3. To allow **voters** to register, log in, and cast their votes online.
4. To ensure that each voter can vote only once per election.
5. To provide **real-time vote counting** and display results instantly after polls close.
6. To implement **data security** measures for protecting sensitive information.

**4. Scope**

**In-Scope (What our project will cover):**

* User registration and authentication.
* Election creation and management (Admin side).
* Candidate management (add/update/remove).
* Vote casting by registered voters.
* Prevention of duplicate voting.
* Automatic vote counting and result display.
* Database management with MySQL.

**Out-of-Scope (Not included in this project):**

* Mobile application (only web-based system).
* Biometric authentication.
* Large-scale government elections (we will implement on small scale).

**2. Project Charter (Week 1)**

* **Problem:** Traditional voting requires physical presence, is time-consuming, and prone to errors.
* **Goal:** To build a secure, web-based voting platform using Java (JSP + JDBC) that allows voters to cast their votes online and enables administrators to manage elections.
* **Objectives:**
  + Implement secure login and registration for voters/admin.
  + Prevent duplicate voting.
  + Provide real-time result calculation.
  + Enable admin to manage elections, candidates, and voters.
* **Deliverables:** Web application (Java-based), UML diagrams, testing reports, deployment-ready package.

**3. GitHub Version Control (Week 2)**

* Repository will be created to manage source code.
* Branching strategy:
  + main branch → stable release.
  + dev branch → active development.
* Git commits will track progress (frontend, backend, database, testing).

**4. UML Modeling (Week 3)**

* **Use Case Diagram**: Voter (Register, Login, Vote, View Results), Admin (Login, Manage Elections, Manage Candidates, Manage Voters).
* **Class Diagram**: Classes for User, Voter, Admin, Election, Candidate, Vote, and DatabaseConnection (Singleton).
* **Sequence Diagram**: "Cast Vote" process showing interactions between Voter → System → Database.

**5. OOP Implementation (Week 4)**

* **Inheritance:** Voter and Admin inherit from abstract User.
* **Encapsulation:** User credentials stored privately with getters/setters.
* **Polymorphism:** Different login behaviors for Admin and Voter.
* **Abstraction:** Abstract class User defines common methods (login, logout).

**6. Debugging & Refactoring (Week 5)**

* Code will be tested and improved for readability, modularity, and maintainability.
* Redundant code removed, consistent naming conventions applied.

**7. Unit Testing – JUnit (Week 6)**

* Test cases:
  + Login validation.
  + Vote casting (valid/duplicate).
  + Result calculation.
* Expected: All critical functions pass before integration.

**8. Design Patterns (Week 7)**

* **Singleton:** Database connection.
* **Factory:** User creation (Admin or Voter).
* **Observer:** Update results dynamically when votes are cast.

**9. JSP + JDBC Integration (Week 8)**

* JSP pages for frontend: login, registration, dashboard, voting, results.
* JDBC for backend: database connection, queries for user authentication, vote storage, and results.

**10. Exception Handling (Week 9)**

* Handle login errors (invalid credentials).
* Handle duplicate voting attempts.
* Catch SQL/database exceptions.

**11. Software Configuration (Week 10)**

* Store configuration details in property files (DB credentials, app settings).
* Use environment variables for sensitive data.

**12. Integration Testing (Week 11)**

* Full workflow: Login → View Elections → Cast Vote → Store in DB → View Results.
* Ensure modules (frontend, backend, database) work together seamlessly.

**13. Code Reviews (Week 12)**

* Peer reviews for code readability, security, and maintainability.
* Apply feedback before final build.

**14. Deployment (Week 13)**

* Package application as **JAR/WAR file**.
* Deploy on **Apache Tomcat Server**.

**15. Final Demo (Week 14)**

* Live demonstration of the Online Voting System.
* Showcase modules: Admin election creation, Voter casting a vote, Real-time result generation.

**16. Requirements**

**Functional Requirements**

1. Voter registration/login.
2. Cast one vote per election.
3. Admin election and candidate management.
4. Automatic vote counting.
5. View results.

**Non-Functional Requirements**

1. Security (encrypted passwords, no duplicate votes).
2. Usability (easy interface, responsive design).
3. Performance (100+ concurrent voters).
4. Reliability (no data loss).
5. Portability (runs on Chrome/Firefox/Edge).

**17. Tools & Technologies**

* **Frontend:** JSP, HTML, CSS
* **Backend:** Java, JDBC
* **Database:** MySQL
* **IDE:** NetBeans / IntelliJ / Eclipse
* **Server:** Apache Tomcat
* **Testing:** JUnit
* **Version Control:** GitHub