**SRS Outline**

**1. Introduction**

* **Project Overview:** The project involves developing a secure and efficient online voting system using blockchain technology.
* **Scope:** The system will allow voters to securely register, authenticate, and cast their votes. The blockchain ensures transparency and immutability in vote recording and result declaration.

**2. Functional Requirements**

These requirements specify what the system should do.

* **Voter Registration:**
  + Voters must register with valid identification details.
  + The system generates a unique voter ID linked to their blockchain account.
* **Voter Authentication:**
  + Implement multi-factor authentication (MFA) using passwords and one-time passwords (OTPs).
  + Ensure the system verifies voter identity before allowing access.
* **Voting Process:**
  + Voters can securely cast their votes from the web or mobile app.
  + The system should prevent double voting.
* **Blockchain Integration:**
  + Votes are recorded as immutable transactions on the blockchain.
  + Smart contracts handle vote counting and result declaration.
* **Result Declaration:**
  + Automatic real-time result processing and display after the voting ends.
  + The system should allow public verification without exposing voter identities.

**3. Non-Functional Requirements**

These requirements outline the system’s performance, security, and usability standards.

* **Security:**
  + Implement end-to-end encryption for all data exchanges.
  + The blockchain must ensure immutability and trust in the voting data.
* **Performance:**
  + The system should handle thousands of simultaneous voters with low latency.
  + Results should be processed in real-time without performance degradation.
* **Scalability:**
  + The system should support scaling to accommodate high user loads.
  + Efficient management of blockchain nodes and data processing.
* **Usability:**
  + Provide a simple and intuitive interface for voters with minimal technical skills.
  + Ensure compatibility across various devices (desktop, mobile, tablets).
* **Reliability:**
  + 99.9% uptime during voting periods.
  + Failover mechanisms in place to ensure high availability.
* **Compliance:**
  + Adhere to election laws and data privacy regulations.

**4. Software Engineering Model**

The **Agile Development Model** is appropriate given the iterative nature of development and the need for continuous feedback.

* **Requirement Analysis:**
  + Collaborate with stakeholders to refine requirements in iterations.
* **Design Phase:**
  + Develop architectural designs including smart contract frameworks, user interfaces, and backend infrastructure.
* **Development Phase:**
  + Implement the system in sprints, focusing on modules such as registration, voting, and result processing.
  + Integrate blockchain functionality progressively, ensuring smart contracts are robust.
* **Testing Phase:**
  + Conduct rigorous unit, integration, and user acceptance testing.
  + Test for security vulnerabilities and ensure all data remains confidential.
* **Deployment Phase:**
  + Deploy the application on cloud infrastructure.
  + Monitor performance during a simulated election and gather feedback for improvements.
* **Maintenance & Future Enhancements:**
  + Post-deployment support includes fixing bugs and rolling out enhancements based on user feedback.