Software Requirements Specification

for

Decentralized Voting System Using Blockchain

Version 1.0

Prepared by Haseeb Gul

Department of Computer Science
Abbottabad University of Science & Technology

01/01/2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This SRS document details the requirements of a voting application to be used in elections. This application is developed to enhance the integrity, confidentiality and accessibility of the voting process. This document captures the project from user registration to vote tabulation for the first pilot implementation targeted at local government elections.

1.2 Definitions, Acronyms and Abbreviations

S No.	Acronyms/Abbreviations	Definitions
1	SRS	Software Requirements Specification
2	GUI	Graphical User Interface
3	OTP	One Time Password
4	E-Voting	Electronic Voting
5	FAQ	Frequently Asked Questions
6	SDKs	Software Development Kits

Table 1: Abbreviations

1.3 Product Scope

1.3.1 Deliverable

A mobile application that will available on both iOS as well as Android.

A user authentication System that provides an effective means for voters to register and login to a system.

Provide an easy to navigate interface for voting.

An admin dashboard system that allows election administrators to perform management functions during voting.

Automated collection of results and reporting, means that once the voting period end the system automatically collect votes, count them and make results available to public.

Helping information and instructions are also available for users of the app.

1.3.2 Features

Confidential and easy process of signing up through verification.

Simple steps that users will have to follow in order to vote.

To make available in clear and bold letters the names of the candidates along with their respective photos and the party affiliations to whom the votes will be cast.

All election data, from vote count and voter choices are kept safe and unseen.

Once the voting period starts, no one-not even an administrator can see or access the results.

User will receive real time updates of the status of their casting or any other important announcements.

To maintain a complete record of every voting or attempted voting activity.

1.3.3 Limitations

Needs a proper internet connection for proper functionality

Less technical friendly for some demographics.

There is a potential occurrence of cyber-attacks that could lead to loss of data.

The first stage of implementation can be limited to a local government election.

1.3.4 Acceptance Criteria

Users must be able to log in, register, cast their votes without critical errors during the testing of the app.

All users will be properly authenticated with strict password requirements to prevent unauthorized access.

The application must have more than 80% on user testing. In other words, it should be easy to navigate and understand.

Technical as well as user documentation should be prepared to explain all the features and functionalities of the apps clearly.

The application must be in compliance with local election laws and regulations specifically regarding voter privacy and data protection.

1.3.5 Goals and Objectives

Improve election processes that every voter, especially the disabled can use for voting.

Provide easier method for voting exercise in order to increase the number of people that vote.

Secures personal data of the voters and ensure that appropriate security controls are in place to ensure election security.

Enable the counting and reporting of election outcomes in the fastest time possible.

Create easy, functional and inclusive mechanism for its target users.

Ensure that voters are identifiable and secured through trusted identification methods to avoid any cons.

2. Overall Description

2.1 Product Perspective

This is a new, self-contained system designed application (not an extension of any existing system). It is developed from scratch with latest technologies to meet the modern security standards. The system will consist of the following components:

Mobile Application: For voter interaction like (registration, authentication, voting).

Admin Dashboard: Admin Dashboard is an interface that contains comprehensive tools for management operations like to create and manage election process, voter verification etc.

Blockchain System: Ethereum-based (Sepolia Test Net) for secure voting recording.

Result Tabulation System: Record Confidential results and displays after voting process ends.

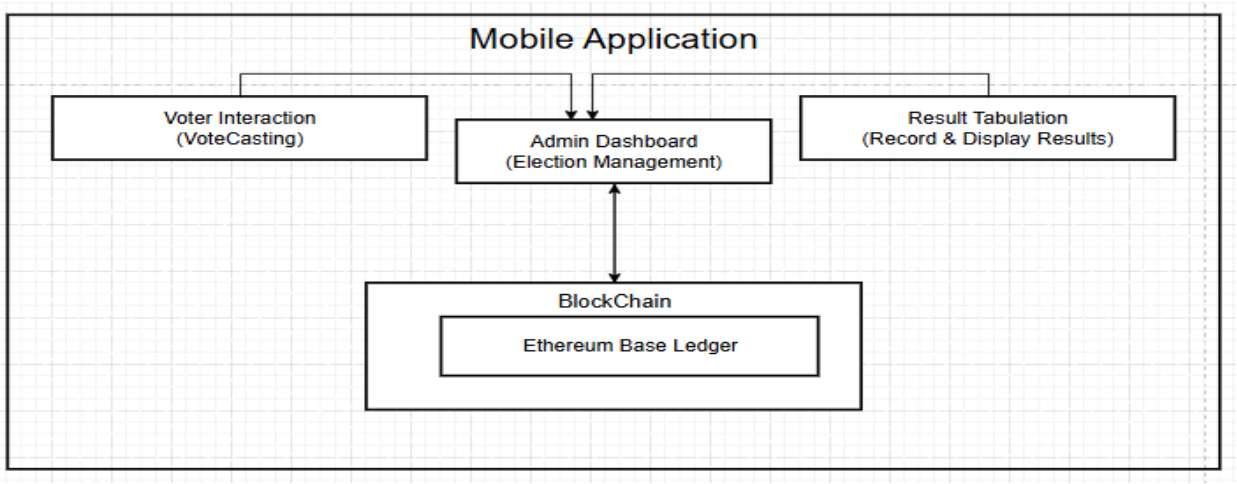


Figure 1: Product Prospective

2.2 Product Functions

This mobile application will have the following major functions listed below:

User Registration & Authentication

Ballot Box Display

Voting Casting & Vote Confirmation

Data Computation & Confidential Results

User Support & Documentation

Audit Trails

Security

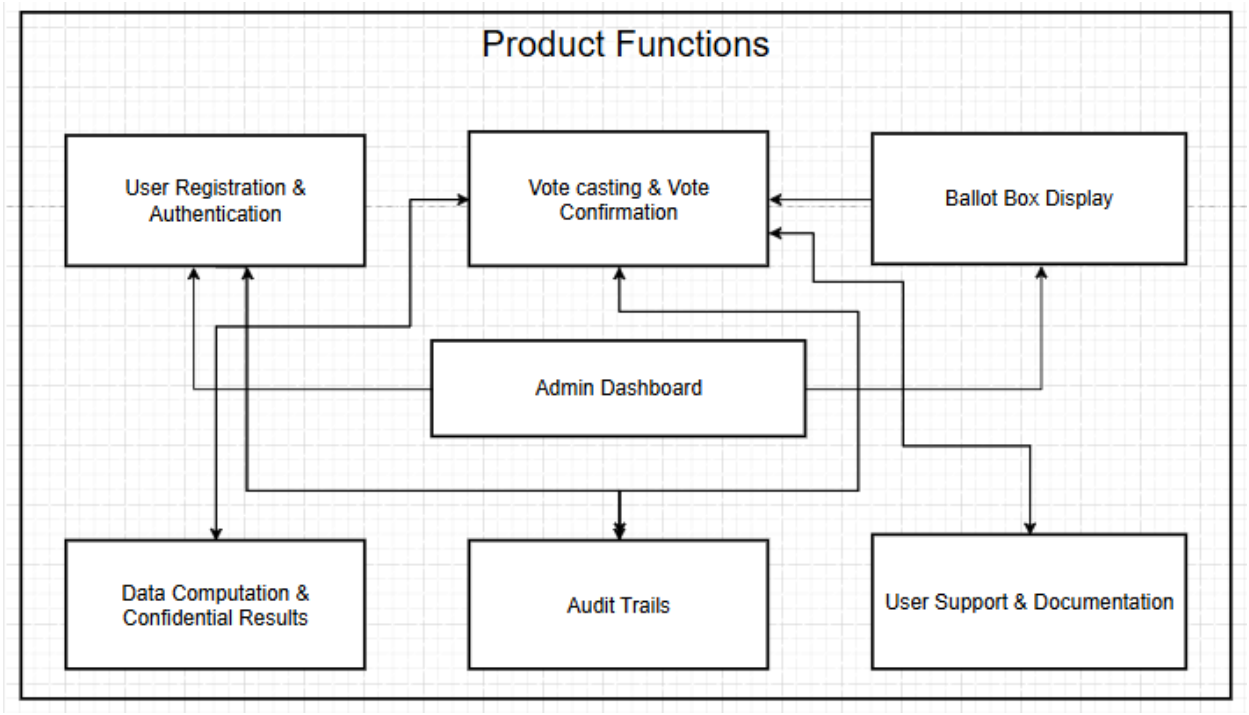


Figure 2: Product functions

2.3 User Classes and Characteristics

2.3.1 Voters:

These are the individuals who are eligible to vote in the election. They are the local citizens that are registered for voting purposes, before casting their votes, voters are required to register and authenticate their identity using OTP.

2.3.2 Admins:

Admins are the administrators of the voting process, they are responsible for managing the voting process, candidate management and election process through admin dashboard. The admin dashboard contains comprehensive tools for management operatios.

2.4 Operating Environment

2.4.1 Operating System:

This application will be developed to run on both Android and iOS platforms so the hardware will need to support these operating systems.

Android devices: Devices should be had at least 2GB RAM and Android version 7.0 or above.

iOS devices: Minimum iPhone 6 with iOS 11 or above.

The mobile devices will have sufficient processing power and storage to install and run the application smoothly.

2.4.2 Backend Servers (Firebase Backend):

While Firebase is itself a cloud-based database so the app's backend does not require dedicated server for hosting, as Firebase provides managed infrastructure.

Firebase runs over Google-provided infrastructure so it depends upon the availability of stable internet access to smoothly handle user-data management, authentications as well as voting process.

2.4.3 Network Environment:

Both mobile application and backend need stable internet connectivity in order to function properly. The user needs internet for validating their credentials., cast votes in real-time and connect to the backend server for data synchronization.

The system interacts with a blockchain, that is, Ethereum based (Sepolia Test Network) it needs good internet access to connect to the blockchain for the verification of transaction.

Firebase shall take care the live updates of data and also authentication for users, so the application's backend depends on Firebase's cloud infrastructure, that automatically takes care of loading and scalability.

2.4.4 Software Dependencies:

React Native will be utilized for native cross-platform mobile application for both Android and iOS

The server-side logic will utilize Node.js integrated with Express.js, where each connection from the mobile application to the database will be dealt.
The smart contracts shall be deployed using Solidity for the Ethereum blockchain based ledger (Sepolia Test Network).
Firebase SDKs and services will be utilized for data storage (Firestore), authentication, and real-time communication.

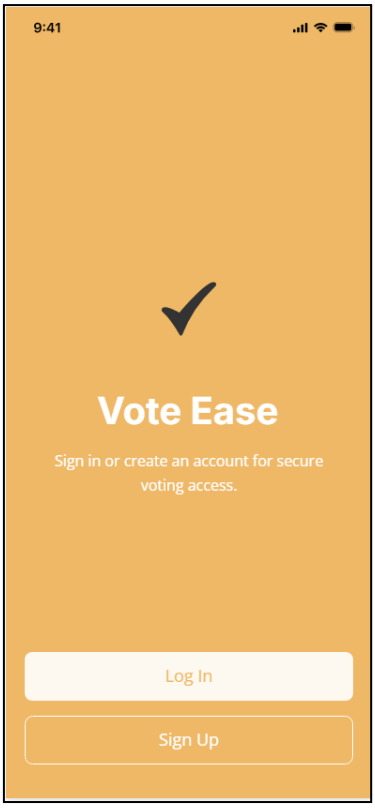
2.5 Design and Implementation Constraints

The system must operate with the local election legislation and policies, particularly aspects regarding the privacy of voters, data protection, and security.
The application will be deployed on both iOS and Android devices, meaning the design has to consider the differences in hardware capabilities between different smartphones. It should be optimized for performance on different processing powers, memory, and screen sizes in order to function properly.
It uses the Ethereum based ledger, which adds constraints such as amount of time it takes for a transaction to be confirmed, and potential scalability limitations. Developers will have to craft smart contracts in solidity with a keen focus on efficiency, security, and reliability under high-load conditions.
This system is considered to be very sensitive and so requires strong security mechanisms, which will be able to prevent cyber-attacks or unauthorized access, as well as data manipulation. Therefore, proper encryption techniques that secure the votes, multi-factor authentication mechanism for user login, as well as the blockchain's innate security features will be fundamental.
The application relies on a stable internet connection for the voting process to work efficiently. In areas with weak internet connectivity, the application may not function as desired, thereby reducing its usability.

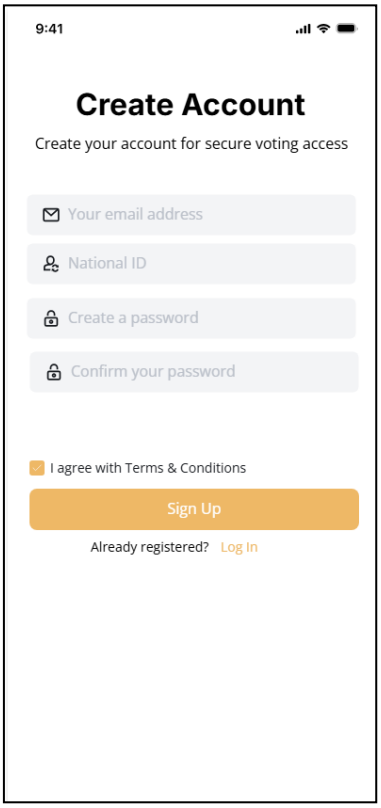
3. External Interface Requirements

3.1 User Interfaces

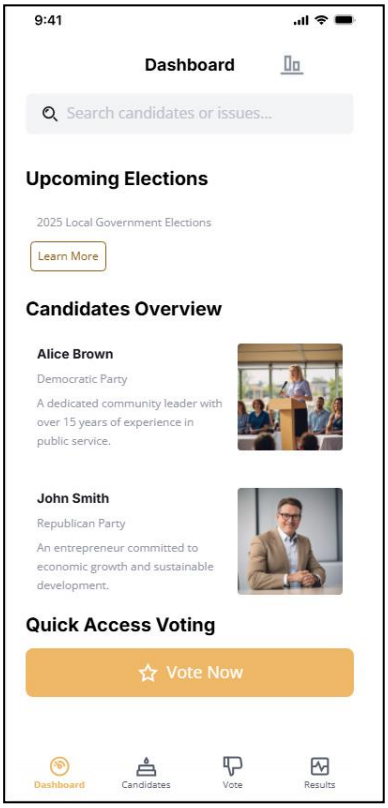
User Interfaces (UI) are the mean of communication through with user interact with the software/system. Some of the **UI** Relevant to our application are:



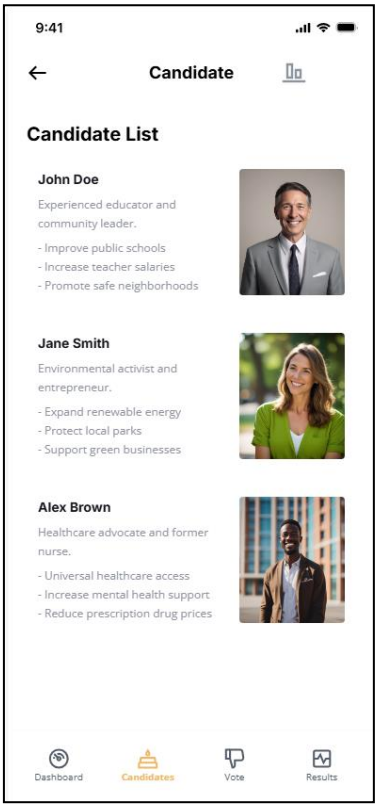
Launch Screen



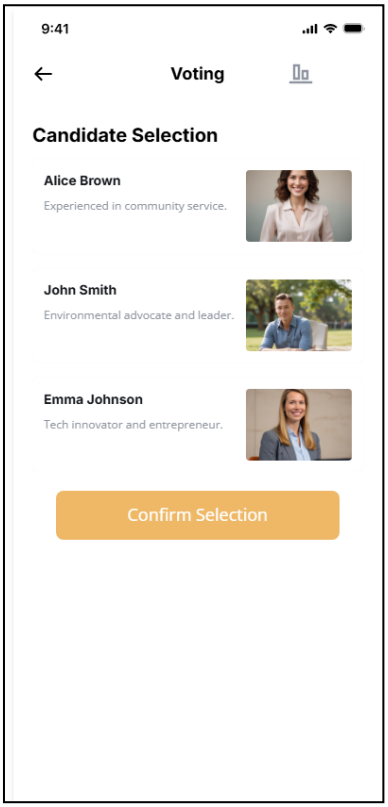
Sign Up Screen



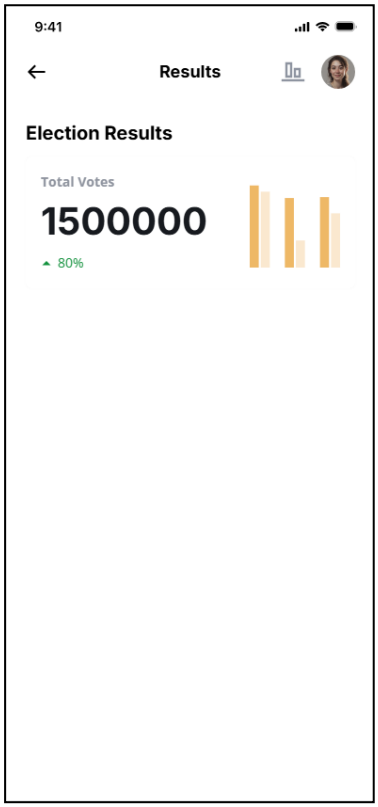
Dashboard Screen



Candidate Screen



Voting Screen



Result Screen

3.2 Hardware Interfaces

No specific hardware interfaces are applicable, as the product is designed to operate on standard mobile devices.

3.3 Software Interfaces

This section details the interactions among various software components this includes databases, operating systems, tools and libraries also identifies data flow, required services, and nature of communication.

3.3.1 Interaction Between Components:

3.3.1.1 Mobile App (Front-end + Backend)

Interaction with Blockchain (Smart Contracts): It interacts with the blockchain by means of smart contracts. Once the user registers or once he has cast the vote, it communicates to the blockchain so that it stores vote data. This way votes will be tamper-proof and secured. It's also interacting with the blockchain after getting the voting period ended, for getting results.

Interaction with Database: The non-sensitive information such as preferences of users, login status, and some basic information during registration are stored locally within the application's database like Firebase. This also serves to handle the user session to keep the application running. On the other hand, data that is sensitive, like records of votes, is being managed using blockchain.

Interaction with User Interface: the frontend provides an interface, which allows users to login, register, browse, and vote for candidates. These interfaces directly hook into an app's backend to serve, for example, registration/authentication processes and vote submissions and show real updates to the user in real time.

Interaction with Authentication System: This mobile application would monitor the registration and login system of the user based on basic verification, which includes email confirmation and multi-factor authentication. The local backend of the application verifies credentials before loading up the voting section.

3.3.1.2 Blockchain (Smart Contracts)

Interaction with the Mobile App: The blockchain is a secure and transparent backbone of voting. It does not store the sensitive user data, including the usernames or passwords but instead records every vote in an immutable ledger. Vote data will be forwarded from the mobile app to the blockchain using smart contracts for integrity and confidentiality of votes.
Interaction with Database: The blockchain itself will not interact with the database; however, the mobile app may store some auxiliary, non-sensitive information in the database, for example, the status of user authentication. But the actual data of votes and results shall be stored exclusively on the blockchain

3.3.1.3 Database (Firebase)

Interaction with Mobile App: This application keeps user registration and session information and non-sensitive information locally in the database. This, therefore, allows users to continue their interaction with this application without requiring a server-side connection.
Interaction with blockchain: On-chain does not interact with the blockchain but has local information used to track any voting process or user's registration status. Core vote data is held within the blockchain.

3.3.1.4 Authentication System

Interaction with Mobile App: The authentication system verifies user credentials during login, ensuring only authorized voters can participate.
Interaction with Backend: The authentication system communicates with the backend to store and validate user credentials. It ensures that only eligible users can vote by checking voter registration details before allowing access to the voting process.

3.3.1.5 OTP Service

Interaction with Mobile App: The OTP service sends one-time password codes to users for verification during login or registration, ensuring secure user identification.
I**nteraction with Backend:** The OTP service interacts with the backend to handle verification processes. Once the user enters the OTP, the backend confirms the validity and grants access to the system.

3.3.2 Ingoing & Outgoing Data / Messages:

3.3.2.1 Incoming Data/ Messages:

Data Item	Purpose
The user's name, email, phone number, address, and identification document (for verification).	This data is used to register a new voter in the system. It ensures the identity of the voter before they can participate in the election.
User login credentials (username/email and password, or multi-factor authentication details such as OTP).	To authenticate the identity of the user when logging into the system. This is to ensure that only registered users can access the voting system
User's vote, i.e. choice of candidate or preference.	This is the vote by the user. It is forwarded to the blockchain for an immutable record of the vote in a secure way.
User identity verification request (using multi-factor authentication such as OTP).	To verify the voter and ensure he is eligible to vote
Login credentials of admin, election monitoring request or result data of election	In order to allow the admin to log in into the voting system and supervise the process of the election, monitor the votes, and see results

Table 3: Ingoing Data Messages

3.3.2.2 Outgoing Data Messages:

Data Items	Purpose
A confirmation message or OTP (One-Time Password) sent to the user.	Authenticate the identity of the user and check if he is registered.
Authentication success or failure, session token of user	To authorize access to the voting system by the user.
A success message or receipt for submitting the vote.	To let the voter know that his vote has been casted and recorded successfully.
OTP code sent to the user's registered phone number or email.	To authenticate users using a one-time password to ensure safe access.
A success message or receipt for submitting the vote.	To let the voter know that his vote has been casted and recorded successfully.

Table 4: Outgoing Data Messages

4. Specific Requirements

This part of SRS illustrates organizing the specific requirements for the e-voting system by system features or functional requirements, the major services provided by the product are:

4.1 User Registration and Authentication

Users must be able to register in the system by providing basic information such as their name, email, and phone number, and subsequently go through a verification process.
A one-time password will be sent to the user's registered phone number or email ID for confirmation of identity.
The system must check on voter eligibility through predefined rules such as age, citizenship, and voter registration.
After registering, users should be allowed to log into the system via a username and password or through multi-factor authentication

4.2 Ballot Box Display

The Ballot Box Display is one of the most crucial components because the actual voting occurs in it.
It displays a list of all the candidates or voting options, along with relevant information such as their names, party affiliations, and photos (if applicable), to help voters make an informed decision.
The system must validate the selection to ensure that votes are cast accurately, and no invalid votes (e.g., for unlisted candidates) are submitted.

4.3 Vote Casting & Vote Confirmation

The Voting Casting & Vote Confirmation feature ensures that after voters select their preferred candidate(s), they will be allowed to confirm the choices before submitting the vote. This step reduces the risk of accidental submission and allows voters to check their choices.
Voters may change their choice if they decide to change their mind at confirmation. Once the vote is confirmed and casted, the system will ensure that the vote is recorded securely in the blockchain so that it cannot be changed.

4.4 Data Computation & Confidential Result

This is very crucial in maintaining the integrity and confidentiality of the voting process. After the closing of the voting period, the system will calculate the outcome and declare results.
The system will use smart contracts on the blockchain to automatically compute the results based on the votes cast.
Results will be kept private during the voting period and will only be made public after the voting is officially concluded.
The process must be secure and tampered-proof, using the unchanged nature of the blockchain technology to guarantee that votes cannot be changed.

4.5 Security and Data Protection

No one, including administrators, shall access the votes during voting period. The votes will be kept confidential until the election period is officially closed.
The system will keep logs of any voting-related action (for instance, registration, voting action) for auditing and accountability purposes.

4.6 Results Tabulation and Reporting

After the voting period ends, the system will automatically tally up the votes, thereby producing a correct result.
After that users will be able to view the election results

4.7 User Support and Documentation

The application must have an FAQ section that addresses the frequently occurring problems for users.
User can write to the support team by email for technical or voting issues.
Support must have technical as well as user documentation, step-by-step instructions on how to register, vote, and check the results.

4.8 Functional Requirements

4.8.1 User Registration & Authentication

This feature is vital to ensure that only registered users can register and vote in the system, and thereby maintain the integrity and security of the election. The system must registration the users

through required details, including their names and contact information such as email or phone number and possibly other identification details like government-issued ID numbers.
At login, users registered here must authenticate with their chosen usernames and passwords. Strong password policies are supported in order to provide security for all account access. OTP verification can be integrated for an additional layer of protection. Only authenticated users will be allowed to access the voting interface.

4.8.2 Eligibility Check

This step ensures that only eligible voters can vote in the election, preventing unauthorized users from casting a vote. This is crucial for maintaining the integrity of the election.
The system must verify the eligibility of the voter based on certain criteria such as age, nationality, or voter registration status.
This is achievable through cross-referencing of user registration details with an authorized voter list that is held by the election authority.
When the voter is not eligible, a clear message from the system will be presented to state the reason why the voter is not allowed to vote-for instance.

4.8.3 Voting Interface & Ballot Design

This requirement will ensure that the voting process is user-friendly, intuitive, and accessible to all voters, including those who are not technologically advanced.
The voting interface should show a plain, simple layout and display a list of candidates available with any other choices relevant for the election (e.g., political party, propositions).

4.8.4 Voting & Vote Confirmation

This allows voters to cast their votes and be sure that their vote is recorded correctly. It also makes sure that the voting process is confidential and accurate.
After the choices have been selected, the voter should be shown a confirmation screen that shows the selected candidates or choices this step allows the voter to review their selections before finalizing their vote.
Once the voter confirms their choices, the system will cast the vote, storing it securely and immutably on the system (e.g., using blockchain for integrity and security).
After the vote is cast, the system should provide an acknowledgment message, ensuring the voter that their vote has been successfully submitted and recorded.

4.8.5 Vote Calculation & Confidential Results

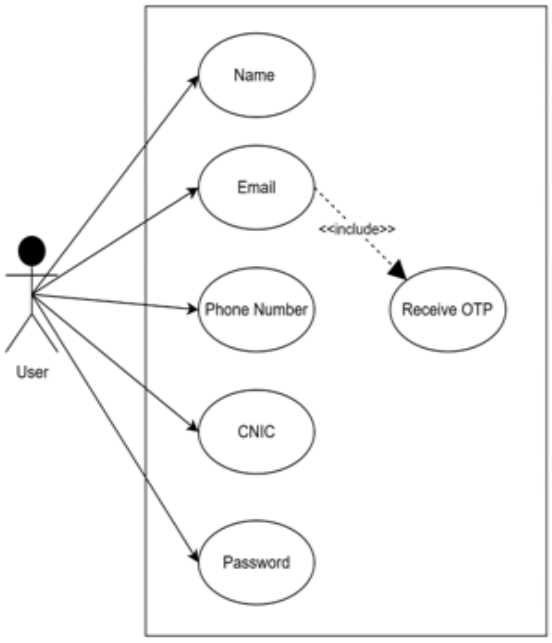
This feature ensures that the results are calculated in secure and confidential manner.
Once the voting period ends, the system automatically computes the results of the election on the basis of the votes cast by registered users. The calculations should be conducted securely, so that vote data remains private during this process.
It must ensure that the results are confidential till published and disclosed officially. The integrity of the results should be maintained using blockchain so that no one can manipulate it.

4.8.6 User Supports (FAQs and Email support)

This module is designed to provide support for users during the process of registration, voting, and checking of results. The goal is to clear out any questions or concerns of the users.
FAQs (Frequently Asked Questions): The system should have an accessible FAQ section that provides clear answers to common questions such as how to register, how to cast a vote, how to change a vote, and what to do if something goes wrong.
Email Support: The system needs to provide a means for a user to contact support, in case of issues that they cannot solve using the FAQs.

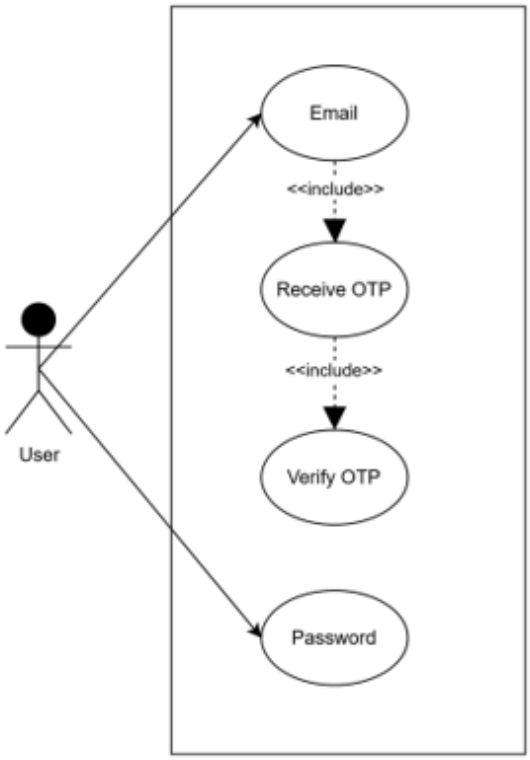
4.8.7 Use Case Diagrams

4.8.7.1 User Sign Up



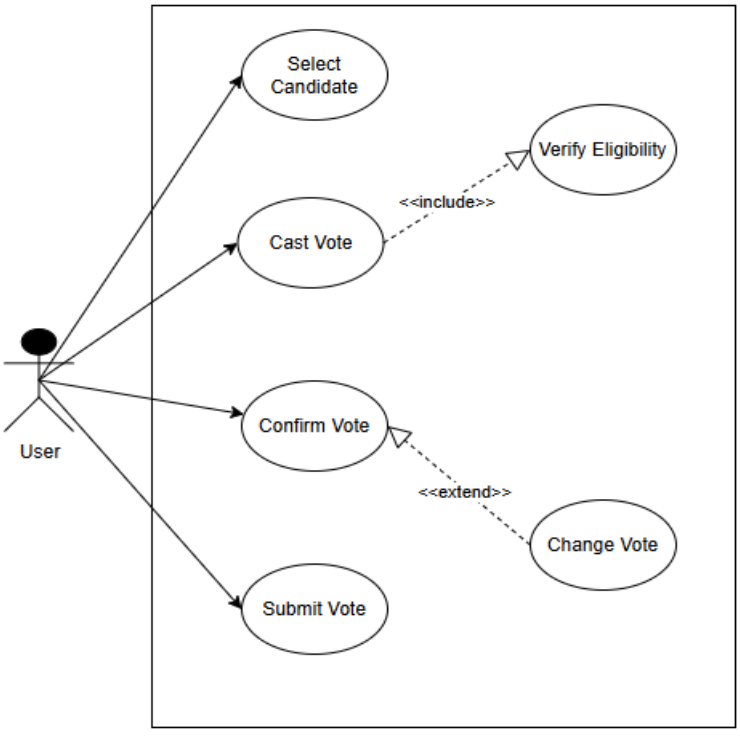
Use Case 1: User Sign Up

4.8.7.2 User Login



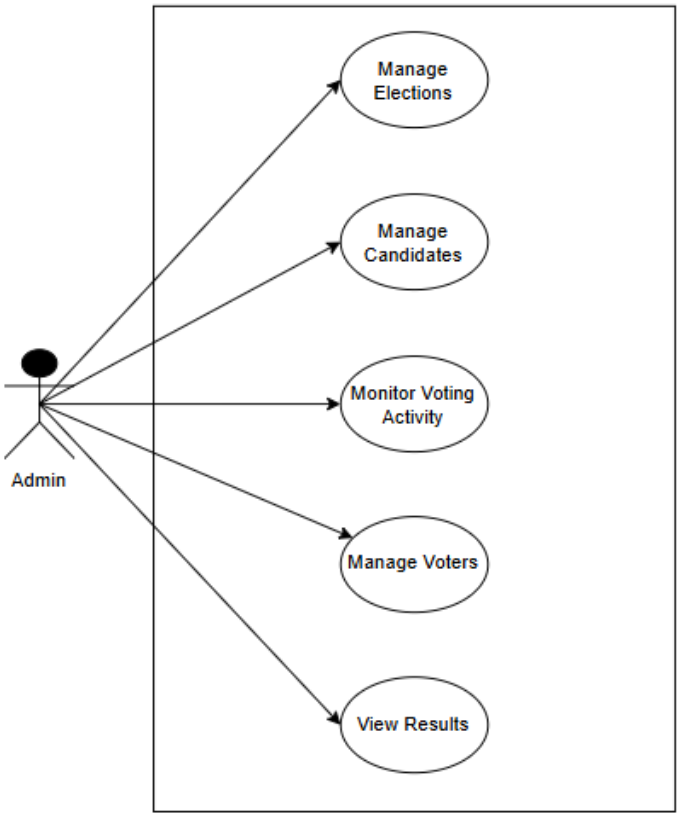
Use Case 2: User login

4.8.7.3 Vote Casting & Vote Confirmation



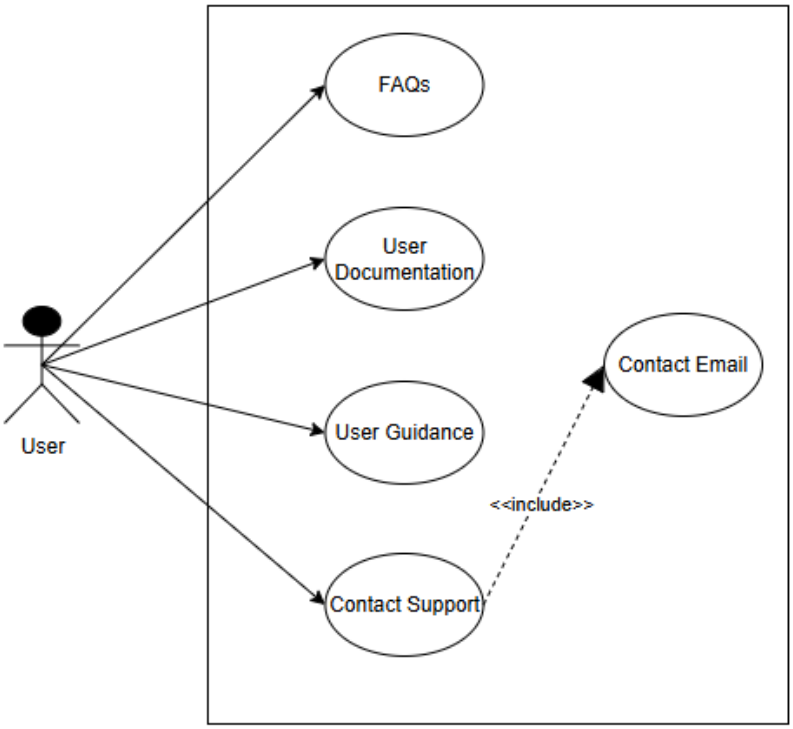
Use Case 3: Vote Casting & Vote Confirmation

4.8.7.4 Admin Management



Use Case 3: Admin Dashboard

4.8.7.5 User Support & Documentation



Use Case 5: User Support & Documentation

5. Nonfunctional Requirements

5.1 Performance Requirements

Performance requirements are the standards defining the operational conditions under which a system should be equipped to work efficiently and effectively. This will ensure the responsiveness, scalability, reliability, and optimal user experience at any given time, especially at election times when the demand is high. The following performance requirements are outlined to ensure the system operates smoothly under various conditions:

Response time to the users: The system must be responsive to the user input actions, such as logging in, registering, and voting. This way, users will have an easy experience without allowing delays that would demote participation, especially when elections are approaching.

Quick Result Tabulation: This ensures that there is no lag in counting votes submitted votes counting so that results of an election are as fresh and reliable as possible, presenting them as they are received.

Latency in Blockchain Transactions: Blockchain transactions, like recording votes, should be confirmed within a reasonable time. While blockchain networks may introduce delays, it is critical to minimize them to maintain a responsive system and give voters confidence that their votes are being accurately recorded.

System downtime and availability: The system must be highly available during the voting period. It should be designed in such a way that minimizes or avoids downtime; otherwise, outages might disrupt election processes, prevent voting, or even cause distrust in the system.

Scalability: The system must be able to efficiently handle a moderate number of users, ensuring smooth performance even as the number of voters increases. It should be able to support peak usage during voting periods, handling multiple concurrent users without slowing down or crashing.

Database Performance: The database of this system must support regular read and writes. Data access and retrieval, such as vote confirmation and election results, must be fast even in peak usage.

Concurrency: The system should be able to handle a huge number of concurrent users. The system should allow all these concurrent users to cast their votes simultaneously without causing any performance degradation.

Data Synchronization: The system should make sure all the voting-related data is in sync with the application user interface and the back end, like blockchain. At the time when a vote has been cast by the user, it is imperative for the system to have reflected this change in blockchain, also provide instant feedback to the user about it on the user interface.

5.2 Safety Requirements

Risk Mitigation Measures: This section details the identification and minimization of potential risks involved in the development and usage of the product. For instance, ensuring that the e-voting application works securely under different conditions, such as high server traffic, without compromising user safety.

Protecting against Unauthorized Access: Preventive measures must be taken to exclude unauthorized users or hackers from gaining improper access to this voting system. Voter and administrator systems must be strongly authenticated using at least one multi-factor authentication mechanisms. Security procedures should ensure that no unwanted interference or attempts to tinker with the voting take place.

Countering Voting Manipulation: To prevent voter manipulation, the system must ensure that votes are cast securely and remain confidential. This can be achieved through strong authentication methods. Additionally, the system should include mechanisms like vote confirmation screens to ensure that each vote is cast accurately and cannot be altered once submitted.

5.3 Security Requirements

Security requirements define the measures that must be implemented to protect the system, data, and users from unauthorized access, threats, or breaches. Below are some key security requirements for our system:

Data Encryption and Storage Security: All sensitive data, including voter identities and votes, will be encrypted with advanced encryption protocols such as AES-256. Data will be stored securely, and only authorized personnel will access it.

Secure smart contracts execution: Smart contracts should be thoroughly tested to avoid vulnerabilities that can compromise the voting process. Any faults in the smart contract code can allow attackers to change votes or information theft. Secure deployment best practices, such as utilizing test networks, are fundamental to validate contract functionality before live use.

Secure Communication Channels: All communication between the voter, the voting system, and administrators must be encrypted ensuring that no unauthorized party can access or tamper with sensitive data, such as votes or personal information, during transmission.
User Access Control: Implement role-based access to restrict access to sensitive data of the system.

5.4 Software Quality Attributes

Software Quality Attributes, are the qualities that are fundamental to the software product in meeting the expectations of the customers and developers. They ensure that the software is of high quality and functions under different conditions. Some key quality characteristics that could be considered for the for our project are:

Adaptability: The system should evolve with new election requirements or changes in technology. For example, it could easily integrate future advancements in blockchain or voting standards.
Availability: The e-voting system should be highly available, means that they are accessible to voters especially during election periods. High availability minimizes downtime and ensures that voters can cast their votes without disruptions.
Correctness: The system must work as expected like counting votes correctly, and aggregate results, and to process voter data properly. Correctness is essential for building the credibility and integrity of the system.
Flexibility: The software should allow change or customization in accordance with different election requirements (candidate information, voter categories, or election laws) without requiring full redevelopment.
Maintainability: The code should be written in a way that is easy to understand and modify. Regular updates, bug fixes, and scalability improvements should be manageable. Given that blockchain and security protocols are central, keeping the system maintainable is important for long-term success.
Portability: The system should be deployable across different platforms, including iOS and Androids. This ensures that voters can access the voting system regardless of their preferred device.

Reliability: The e-voting system should operate consistently and handle peak loads, such as during the voting period. It should recover from errors quickly, ensuring that the election process is not disputed.

Testability: The system must be testable, with defined components and interfaces for unit testing and system testing. Proper testing of smart contracts and backend services is crucial in order to avoid any vulnerability.

Usability: The e-voting system should be easy to use, enabling people of different ages and technical abilities to register, log in, cast their votes, and receive confirmations. A good user interface will help ensure higher voter participation and satisfaction.

5.5 Business Rules

5.5.1 Voter Authentication and Eligibility:

Only registered and verified individuals can cast votes.

Voters must be authenticated using a secure multi-factor authentication process, such as OTP to verify their identity.

Only voters who have satisfied the eligibility criteria, such as legal age and voter registration, are allowed to access the voting interface.

Once registered and authenticated, a voter can only vote once during an election cycle. A voter cannot vote more than once during an election cycle.

If voter is not eligible a clear message from the system will be shown with a reason that why he/she is not eligible to vote.

5.5.2 Roles and Permissions:

5.5.2.1 Voter Role:

Register their details securely.

Log into the voting application during the voting period

Cast and view their ballots before casting them

Have a vote confirmation receipt generated for their vote

5.5.2.2 Administrative Roles:

Create and manage elections, defining candidates and election rules.

Track registration and voting levels without being able to access any ballot cast during an election.
Ensure that the system is safe and managed, for example by updating smart contracts when necessary.
Review User activities and report problems to the appropriate authorities.
Prove a user support through email in which admin dealt with the user complications and solve those complications.
System maintenance is also a part of administrative role.

5.5.3 Vote Privacy:

The votes are secret, and none, including administrators, may access them.
At the close of the election, votes are automatically computed, and results are made available to the public.
Information regarding the voters must remain encrypted, and thus not accessible to unauthorized parties or individuals.

5.5.4 Election Period:

Only allow voting during the official election period. Once the voting period ends, no new votes can be cast.
Time for start and end the voting would be determined by the admin of the site. The strict rules of system will strictly not accept the votes of persons before and after the specific election timing is scheduled.

5.5.5 Voting Interface Restrictions:

Once the vote is cast and confirmed, voters cannot change it.
Voters can view and edit their ballot until the confirmation step.

5.5.6 Tallying Results and Declaration

The results are automatically tabulated after the election period ends and cannot be changed by anyone, including administrators, during the voting period.
Once Voting period ends the system automatically collect votes, tabulate them, generate report and make available to public.
The results will be displayed through the system’s results tabulation interface.

6. References

Guidelines For Developing a Good GUI by Antcheva, R. Brun, F. Rademakers, CERN, Geneva, Switzerland
https://cds.cern.ch/record/865663/files/p613.pdf?form=MG0AV3

Graphical user interface (GUI) testing: Systematic mapping and repository by Ishan Banerjeea, Bao Nguyena, Vahid Garousib, Atif Memona.
https://www.cs.umd.edu/~atif/papers/ISTSurvey2013.pdf?form=MG0AV3

Blockchain smart contracts: Applications, challenges and future trends by Shafaq Naheed Khan, Faiza Loukil, Chirine Ghedira-Guegan, Elhadj Benkhelifa & Anoud Bani-Hani
https://link.springer.com/article/10.1007/s12083-021-01127-0?form=MG0AV3

Smart Contracts in Blockchain Technology: A Critical Review by by Hamed Taherdoost
https://www.mdpi.com/2078-2489/14/2/117?form=MG0AV3

IEEE Recommended Practice for Software Requirements Specifications (IEEE Std 830-1998)
https://cse.msu.edu/~cse870/IEEEXplore-SRS-template.pdf?form=MG0AV3

Automatic Generation of a Software Requirements Specification (SRS) Document by Marinos Georgiades
https://www.academia.edu/15276356/Automatic_generation_of_a_Software_Requirements_Specification_SRS_document?form=MG0AV3

Blockchain-Based E-Voting Systems: A Technology Review by Mohammad Hajian Berenjestanaki, Hamid R. Barzegar and Nabil El Ioini.
https://www.mdpi.com/2079-9292/13/1/17

Enhancing Security and Transparency in Online Voting through Blockchain Decentralization by Inderpreet Singh, Inderpreet Singh, Amandeep Kaur, Parul Agarwal and Sheikh Idrees.
https://www.researchgate.net/publication/384465973_Enhancing_Security_and_Transparency_in_Online_Voting_through_Blockchain_Decentralization

A Voting System Using Blockchain a Deep Survey by Rajesh Gupta, Rajesh Gupta, Aditi Sharma, Vk Satya Prasad, Sura Rahim Alatba**.**
https://www.researchgate.net/publication/372619119_A_Voting_System_Using_Block_Chain_A_Deep_Survey

Appendix A: Client Contracts

Appendix B: Questionnaires