**Test plan for**

**BLOCKCHAIN BASED**

**E-Voting System**

Made by:

Rhythm Garg (2100290120141 – CS-7C)

Test Manager:

Prof. Rishabh Chakraborty

*ChangeLog*

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Change Date** | **By** | **Description** |
| 001 | 09.12.2024 | Rhythm Garg | Intitial Draft |
|  |  |  |  |
|  |  |  |  |

1 Introduction 2

1.1 Scope 2

1.1.1 In Scope 2

1.2 Quality Objective 2

1.3 Roles and Responsibilities 2

2 Test Methodology Error! Bookmark not defined.

2.1 Overview 2

2.2 Test Levels **Error! Bookmark not defined.**

2.3 Test Completeness 3

3 Test Deliverables 3

4 Resource & Environment Needs 3

4.1 Testing Tools 3

4.2 Test Environment 3

5 Terms/Acronyms 4

# Introduction

SecureVote is a blockchain-based e-voting system built on the Ethereum blockchain, ensuring secure, transparent, and tamper-proof elections. The system incorporates Aadhar-based biometric authentication for voter verification, ensuring each vote is uniquely and securely cast. The test plan includes strategies for verifying smart contract functionality, system integration, and security measures.

## Scope

### In Scope

Features to be tested:  
- Smart contract functionality in Solidity.  
- Aadhar-based biometric authentication.  
- Vote casting, validation, and result computation.  
- System integration and blockchain node communication.  
- Usability and accessibility of the voting interface.

## Quality Objective

Some objectives of your testing project could be

* Ensure all smart contracts meet functional and non-functional requirements.
* Validate the security of the blockchain network.
* Confirm voter data integrity and authenticity.

## Roles and Responsibilities

Roles and Responsibilities:

* QA Analyst: Rhythm Garg
* Test Manager: Prof. Rishabh Chakraborty
* Configuration Manager: Rhythm Garg
* Developer: Rhythm Garg
* Installation Team: Prof. Arti Sharma, Rhythm Garg.

# Test Methodology

## Overview

The testing methodology for SecureVote adopts an Agile approach, ensuring iterative and continuous testing during development. Each feature is tested incrementally, allowing for timely identification and resolution of issues. This approach ensures that all components function as intended, both individually and when integrated.

## Test Levels

1. **Unit Testing**: Validating the functionality of individual smart contract functions.  
2. **Integration Testing**: Ensuring seamless interaction between the authentication system, blockchain network, and user interface.  
3. **System Testing**: Verifying the entire system, including end-to-end voting, authentication, and result calculation.  
4. **Security Testing**: Assessing the system's resistance to vulnerabilities such as double-voting, unauthorized access, and data breaches.

## Test Completeness

Test Completeness  
Testing will be deemed complete when:  
- All smart contract functions achieve 100% code coverage.  
- Security vulnerabilities are resolved.  
- All critical bugs are fixed, and non-critical bugs are documented for future releases.  
- The voting system passes performance benchmarks for up to 1 million users.

# Test Deliverables

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Comprehensive test cases for smart contract operations.Bug reports and resolution logs.Performance metrics for voter throughput and system latency.Security audit report for the blockchain and authentication system. Test Cases for Aadhar Authentication:  Boundary Value Analysis (BVA) is a testing technique used to validate inputs at the edge of acceptable ranges, ensuring the system works correctly for both valid and invalid boundaries.  An Aadhar Card number is a 12-digit numeric identifier issued in India. The valid range of digits is 0000 0000 0001 to 9999 9999 9999.   |  |  |  |  | | --- | --- | --- | --- | | Test Case ID | Test Input | Expected Result | Reason | | TC1 | 000000000001 | Valid | Minimum Valid Value | | TC2 | 999999999999 | Valid | Maximum Valid Value | | TC3 | 000000000000 | Invalid | Below Lower Boundary | | TC4 | 1000000000000 | Invalid | Above upper boundary | | TC5 | 12345678901 | Invalid | Less than 12 digits | | TC6 | 1234567890123 | Invalid | More than 12 digits | | TC7 | 1234A5678901 | Invalid | Alphanumeric Input | | TC8 | 12@#$5678901 | Invalid | Special Characters Input | | TC9 | “” | Invalid | Empty Input | | TC10 | “ “ | Invalid | Input with spaces only |   Decision Table to allow/reject voting based on several conditions:   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Rule ID | C1: 12 Digits | C2: Numeric Only | C3: Valid Range | C4: Already Voted | Action | | R1 | Yes | Yes | Yes | No | Allow Voting | | R2 | No | - | - | - | Reject Due to Invalid Aadhar | | R3 | Yes | No | - | - | Reject Due to Invalid Aadhar | | R4 | Yes | Yes | No | - | Reject Due to Invalid Aadhar | | R5 | Yes | Yes | Yes | Yes | Reject Due to Duplicate Vote | | R6 | No | No | - | - | Reject Due to Invalid Aadhar | |

# Resource & Environment Needs

## Testing Tools

Ganache: For local blockchain simulation.  
Truffle: For deploying and testing smart contracts.  
Postman: For API testing of authentication endpoints.  
Selenium: For automated testing of the voting interface.  
OWASP ZAP: For security testing and vulnerability analysis.

## Test Environment

Hardware Requirements:  
- Minimum: 8 GB RAM, Quad-core CPU, 256 GB SSD.  
- Recommended: 16 GB RAM, Octa-core CPU, 512 GB SSD.

Software Requirements:  
- Operating System: Windows 10 or Ubuntu 20.04.  
- Blockchain Network: Ethereum (Rinkeby Testnet).  
- Authentication System: Aadhar-based biometric verification modules.  
- Tools: Ganache, Truffle, Visual Studio Code.

# Terms/Acronyms

API: Application Programming Interface.  
AUT: Application Under Test.  
ETH: Ethereum cryptocurrency.  
PoW: Proof of Work (Ethereum's consensus mechanism).  
Rinkeby: Ethereum's test network.