KIET Group of Institutions, Ghaziabad

COMPUTER SCIENCE

Testing Report

for

Major Project

E-Voting using Blockchain

2023-2024

Mentor: Mr. Akash Goel (Asst. Professor CS)

Ashish Kumar Gupta 2000290120043 CS 7A
Saurabh Pundir 2000290110149 CS 7A
Aditya Aggarwal 2000290120013 CS 7A

TEST PLAN FOR
E-VOTING USING BLOCKCHAIN

ChangeLog

Version	Change Date	By	Description
version number	Date of Change	Name of person who made changes	Description of the changes made
1	November 1 2023	Ashish Kumar Gupta	Performed test cases, added test delieverables

1 Introduction

Building a secure electronic voting system that offers the fairness and privacy of current voting schemes, while providing the transparency and flexibility offered by electronic systems has been a challenge for a long time. In this work-in-progress paper, we evaluate an application of blockchain as a service to implement distributed electronic voting systems. The paper proposes a novel electronic voting system based on blockchain that addresses some of the limitations in existing systems and evaluates some of the popular blockchain frameworks for the purpose of constructing a blockchain-based e-voting system In particular, we evaluate the potential of distributed ledger technologies through the description of a case study; namely, the process of an election, and the implementation of a blockchain-based application, which improves the security and decreases the cost of hosting a nationwide election. This technology will improve the trust of voters that there action is secure.

1.1 Scope

1.1.1 In Scope

This shows the aspects of the E-Voting System that are within the scope of our testing efforts. It includes the following:

- **Voting Interface**: Testing the user interface to ensure that it is user-friendly, accessible, and accurately captures voter choices.
- **Vote Encryption and Decryption**: Ensuring the cryptographic processes used to secure votes are functioning correctly.
- **Vote Recording**: Validating that each vote is accurately recorded on the blockchain ledger.
- **Security Measures**: Verifying the effectiveness of security measures to protect against unauthorized access.
- **Performance**: Assessing the system's responsiveness and scalability to handle a significant volume of votes.
- **Usability:** Evaluating the overall user experience and accessibility of the system.

1.1.2 Out of Scope

For our E-Voting System, the following are out of scope:

- No. of voters: Only limited number of entities can be tested.
- **Blockchain Technology:** The core blockchain technology itself, as it should have undergone extensive testing during its development.
- **Hardware Infrastructure:** The physical hardware infrastructure supporting the system is out of scope.
- **Network Infrastructure:** The broader network infrastructure is not within our testing purview.

1.2 Quality Objective

Our quality objectives are:

- **Accuracy**: Ensuring that each vote is recorded and counted accurately.
- **Security**: Guaranteeing the integrity, confidentiality, and availability of the voting data.
- **Usability**: Creating a user-friendly system that is accessible to a broad range of voters.
- **Performance**: Ensuring the system can handle a significant load without degradation.

1.3 Roles and Responsibilities

Detail description of the Roles and responsibilities of team members

- Test Manager (Ashish Kumar Gupta)
- Tester (Ashish Kumar Gupta) – Tested the test cases.
- Developers (Ashish Kumar Gupta, Saurabh Pundir, Aditya Aggarwal) - Addressed and resolved issues identified during testing.
- Project Manager (Mr. Akash Goel) - Oversee the project's progress and ensure alignment with testing efforts.

2 Test Methodology

2.1 Overview

For our E-Voting System project, we have adopted the Waterfall test methodology. The choice of this methodology is driven by several key factors specific to our project's requirements and characteristics:

- As our requirements are clearly defined and stable we are using Waterfall methodology.
- The voting process is subject to strict regulations, and a Waterfall approach allows for thorough planning and documentation to ensure compliance with legal and security requirements.
- Waterfall provides a structured approach.

2.2 Test Levels

Following are the testing levels that are defined based on the case of the E-Voting System:

- **Unit Testing**: Ensuring that the all the modules are working correctly.
- **Integration Testing**: Testing that all the integrated modules are working as expected.
- **System Testing**: Evaluating the entire system to verify its correctness and compliance with election regulations.
- **User Acceptance Testing (UAT)**: Allowing end-users, including election officials and voters, to validate the system for usability and suitability.

2.3 Suspension Criteria and Resumption Requirements

- **Suspension Criteria**: If a critical security vulnerability is discovered during requirements testing, further testing may be suspended until the issue is resolved.
- **Resumption Criteria**: In the case of the security vulnerability, testing can resume when the issue is fixed, and the system is confirmed secure.

2.4 Test Completeness

- All requirements are met.
- Compatibility: Software is compatible with other platforms, browsers, devices, OS.
- 100% test coverage
- All Test cases executed
- All open bugs are fixed or will be fixed in next release.

3 Test Deliverables

3.1 Manual Testing

Test_Case_ID	Test Case Objective	Pre Requisite	Input Data	Expexted Output	Actual Output	Status
TC_01	Sign in using MetaMask	MetaMask wallet account	Account Password	Logged in	logged in	PASS
TC_02	Test Image upload to IPFS using API	System must be connected to internet	Image	Image Uploaded	Image Uploaded	PASS
TC_03	Retrieving uploaded image from IPFS	System must be connected to internet		Image Retrieved	Image Retrieved	PASS
TC_04	Registering Candidate & Connecting with Smart Contract	User must be logged in with admin account	Details of candidate & unique address	Candidate registered	Candidate registered	PASS
TC_05	Registering Voter	User must be logged in with admin account	Details of voter & unique address	Voter registered	Voter registered	PASS
TC_06	Voting as Voter	Must be registered by admin		Voted	Voted	PASS
TC_07	Voting again	Must be voted already		You have already voted	You have already voted	PASS

3.2 Automation Testing

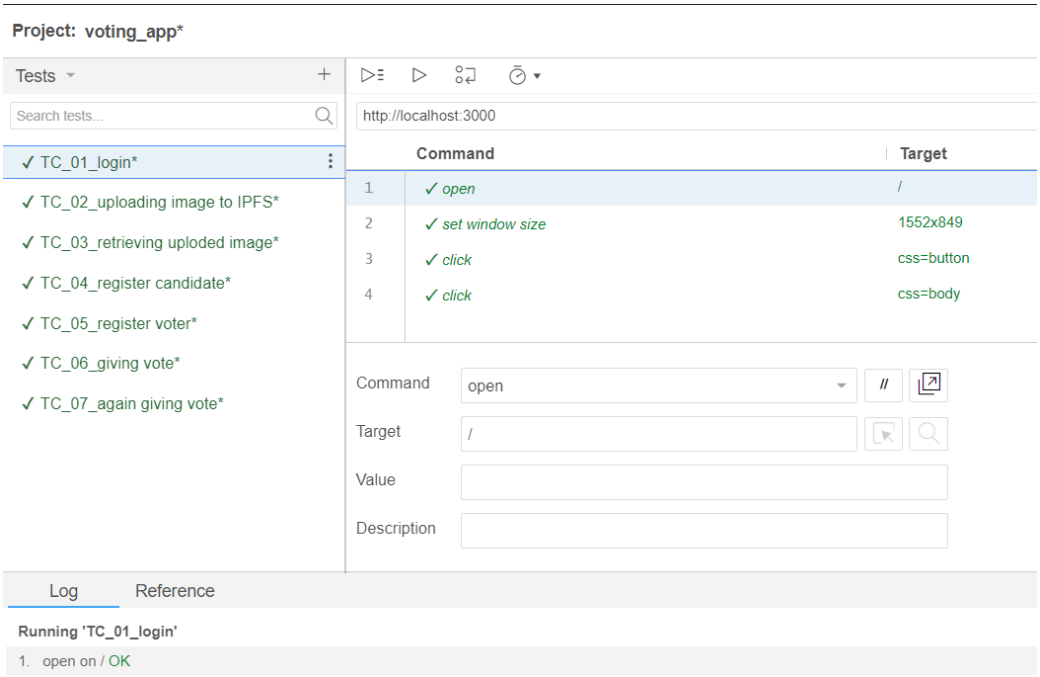
Test Case 1: Sign In Using Metamask

Test Description: This scenario evaluates the functionality of signing into the E-Voting System using MetaMask, a popular Ethereum wallet and gateway to blockchain applications.

Steps:

1. Open the E-Voting System application.
2. Click on the "Sign In" option.
3. Select "Sign in using MetaMask."
4. Connect the MetaMask extension.
5. Verify successful sign-in and access to the user's account.

Expected Result: The user can sign in using MetaMask, and their account is accessible within system.



Logs:

- **Running 'TC_01_login'20:47:36**
- 1.open on / OK20:47:36
- 2.setWindowSize on 1552x849 OK20:47:37
- 3.click on css=button OK20:47:37
- 4.click on css=body OK20:47:39
- **'TC_01_login' completed successfully20:47:39**

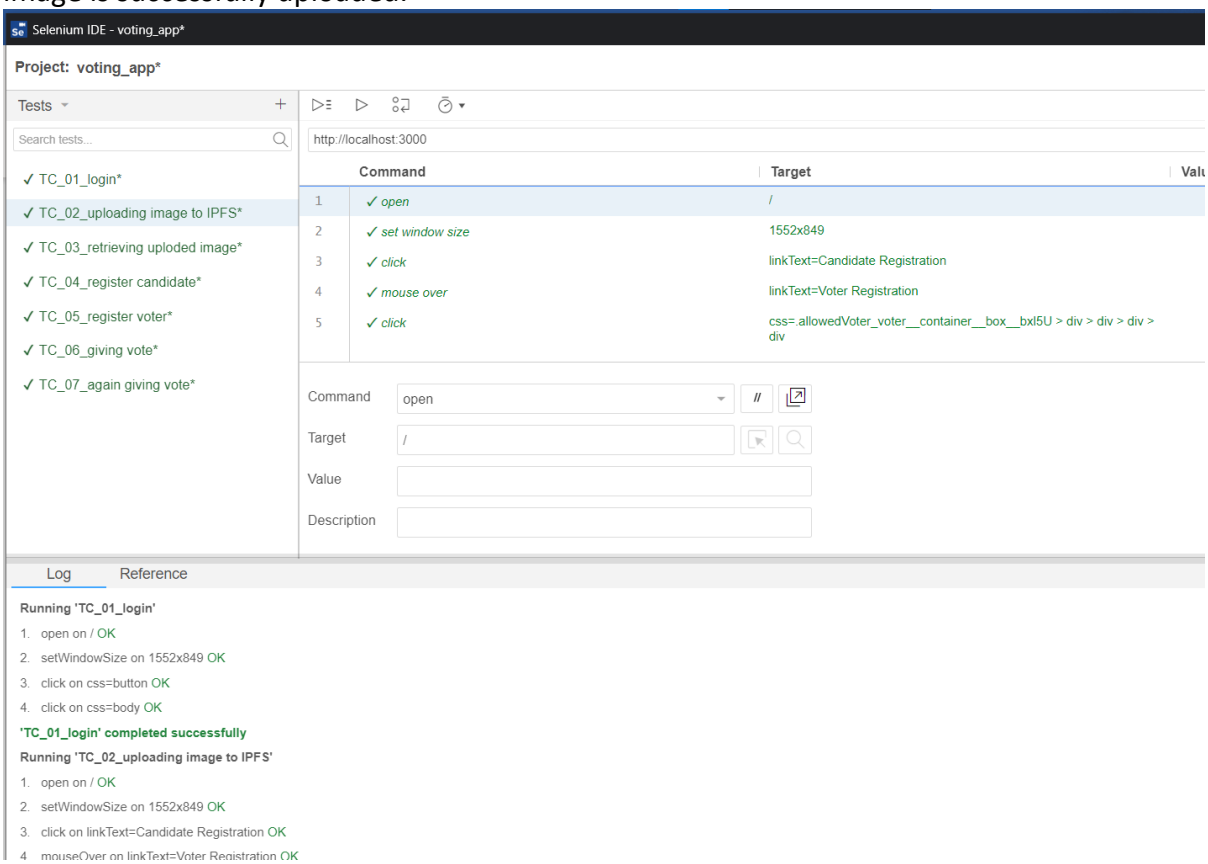
Test Case 2: Test Image Upload to IPFS Using API:

Test Description: This scenario examines the capability of the system to upload an image to IPFS via an API.

Test Steps:

1. Log in to the E-Voting System.
2. Navigate to the "Candidate Registration" section.
3. Upload an image using the upload button.
4. Confirm successful image upload.

Expected Result: The system allows users to upload images to IPFS through the IPFS API, and the image is successfully uploaded.



Test Case 3: Retrieving Uploaded Image from IPFS (after upload automatically)

Test Description: This scenario tests the system's ability to retrieve an image previously uploaded to IPFS.

Test Steps:

1. Upload the image to the IPFS.
2. It will automatically retrieve and show the image.
3. Verify that the correct image is displayed.

Expected Result: The system successfully retrieves and displays the uploaded image from IPFS.

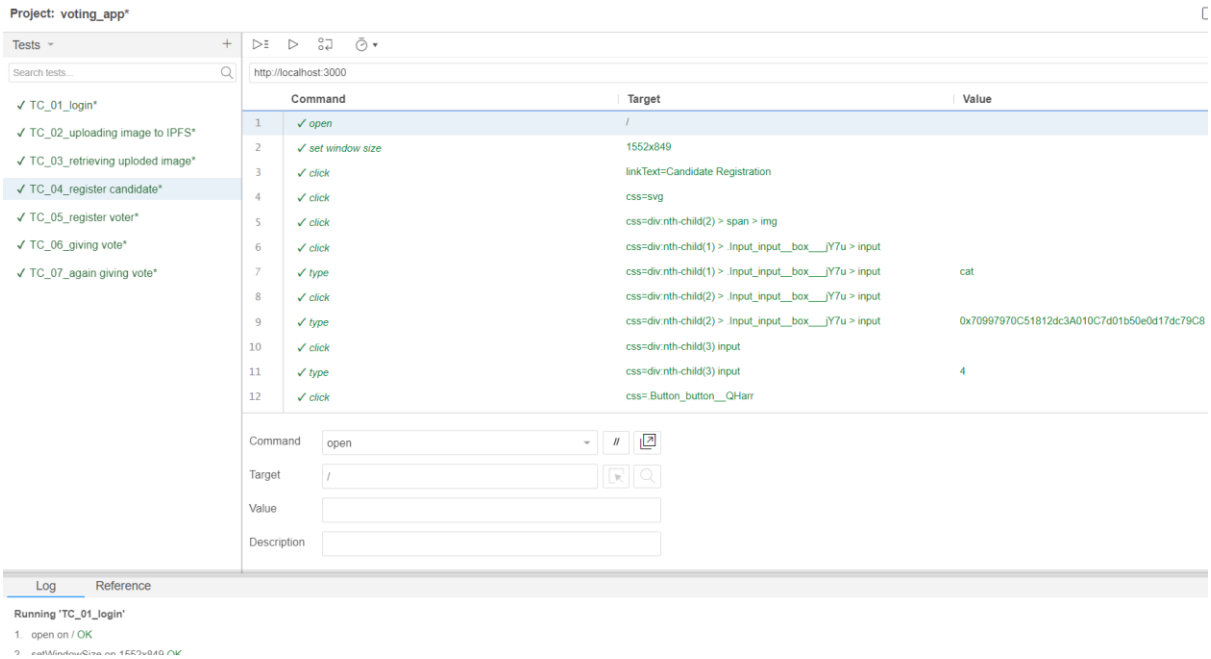
Test Case 4: Registering Candidate & Connecting with Smart Contract

Test Description: This scenario assesses the system's functionality to register a candidate and establish a connection with the underlying Smart Contract.

Test Steps:

1. Log in to the E-Voting System as an administrator.
2. Access the "Candidate Registration" section.
3. Register a candidate, providing relevant details and uploading image.
4. Confirm the transaction process.

Expected Result: The system successfully registers the candidate and establishes a connection with the Smart Contract, allowing the candidate to participate in the election.



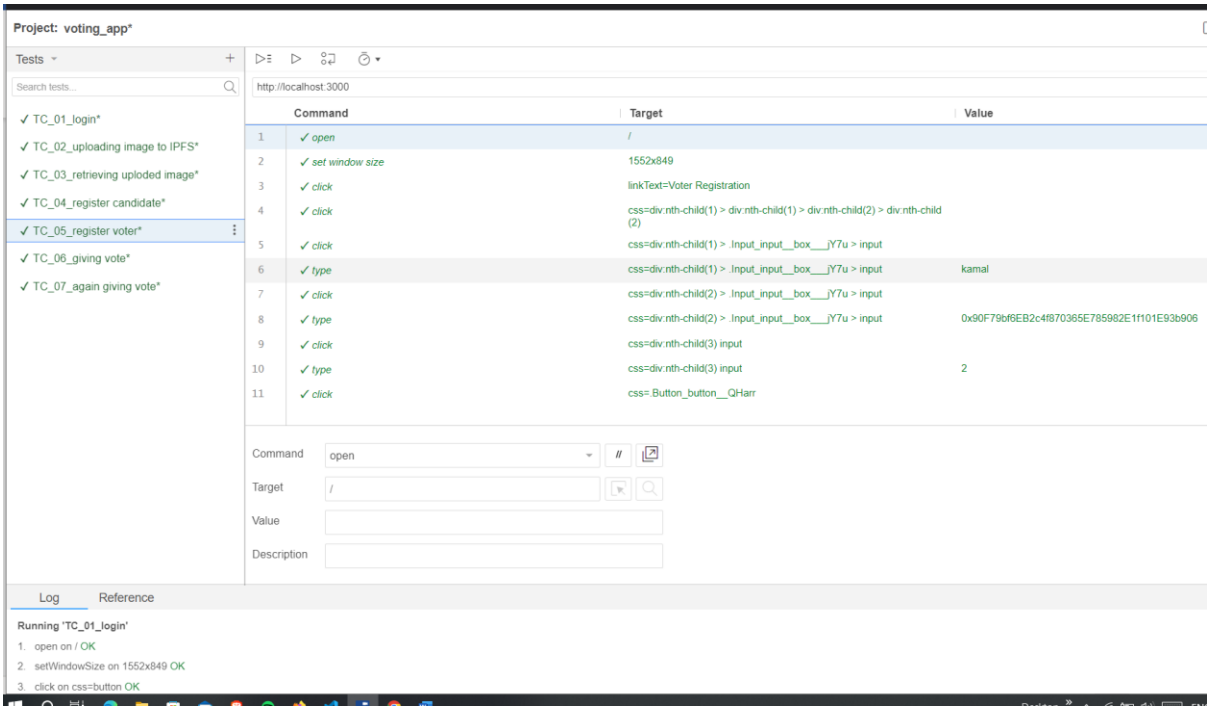
Test Case 5: Registering Voter

Test Description: This scenario evaluates the system's capability to register voters for the election.

Test Steps:

1. Log in to the E-Voting System as an administrator.
2. Navigate to the "Voter Registration" section.
3. Register a voter by entering their details.
4. Verify that the voter's registration is recorded in the system.

Expected Result: The system registers the voter, making them eligible to participate in the election.

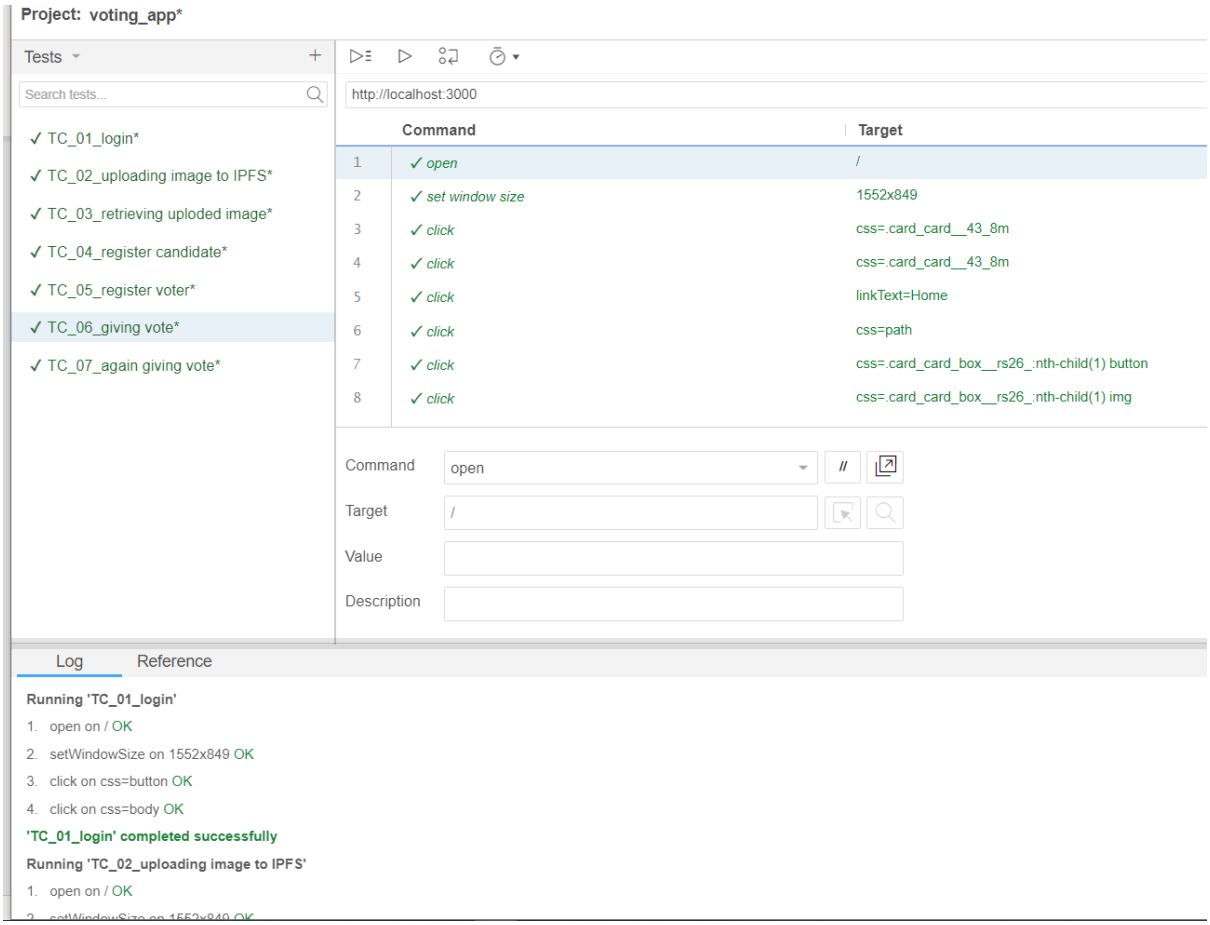


Test Case 6: Voting as Voter

Test Description: This scenario evaluates the voting process for a registered voter.
Test Steps:

1. Log in as a registered voter.
2. Go to homepage of web application.
3. Cast a vote for a candidate by clicking vote button.
4. Verify that the vote is recorded in the system.

Expected Result: Registered voters can cast their votes successfully, and the system records their choices.



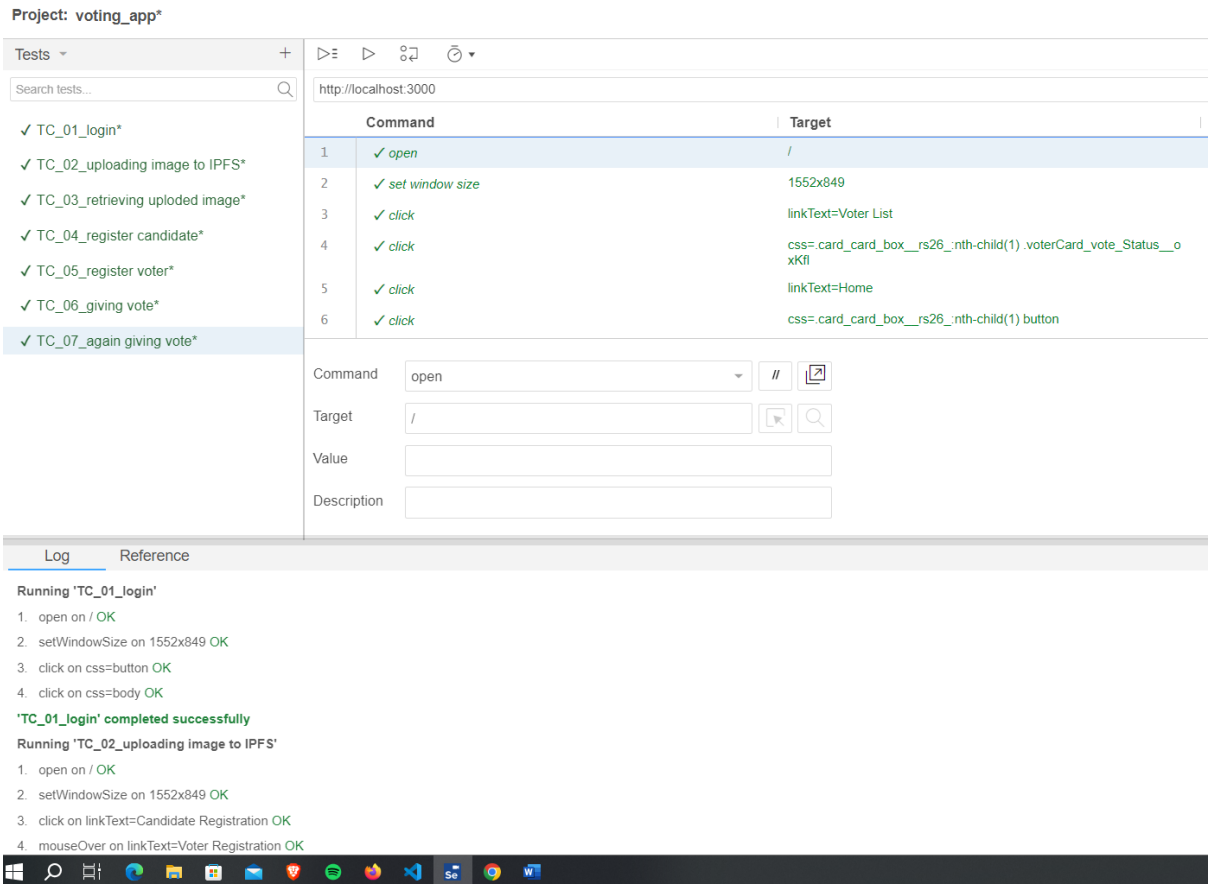
Test Case 7: Voting Again

Test Description: This scenario verifies the system's ability to prevent a voter from casting multiple votes.

Test Steps:

1. Log in as a registered voter.
2. Cast a vote for a candidate.
3. Attempt to vote again using the same voter account.
4. Confirm that the system prevents the voter from casting multiple votes.

Expected Result: The system should restrict voters from casting multiple votes, ensuring the integrity of the election process.



4 Resource & Environment Needs

4.1 Testing Tools

Following Tools like

- Selenium IDE
- Browser (Firefox or chrome)

4.2 Test Environment

Following **software's** are required in addition to client-specific software.

- Windows 10 and above
- Browser
- Microsoft Visual Studio

5 Terms/Acronyms

Make a mention of any terms or acronyms used in the project

TERM/ACRONYM	DEFINITION
API	Application Program Interface
AUT	Application Under Test
IPFS	Inter Planetary File System
IDE	Integrated Development Environment