National College of Ireland

Software Project

Business Information Systems

2021/2022

Constantin Maftean

x18493294

x18493294@studemt.ncirl.ie

Evote - Blockchain e-voting system

Technical Report

Contents

Executive Summary

The purpose of this report is to highlight every step I followed in the creation of the Evote application which is a general electronic voting system built with blockchain technology built by Near Protocol. The report also contains information about how I managed to achieve all the goals I set out at the beginning of the project, as well as testing methodologies I used to validate these goals.

First of all, I broke down all the application's features into use cases that contain a visualization and description of how the respective feature is intended to work, how the system manages the process, what errors may occur during the process, and how they (errors) are being handled.

Evote's main function is to allow a designated user to create a voting poll and post it on the main page where any users can vote. The application uses the NEAR blockchain technology to ensure that a poll can not be altered and under any circumstances a user can not vote more than once for the same poll. The application also provides a live voting results feature that will allow any users to see the results of a poll at any time during the voting phase.

In conclusion, I believe that even if I had to modify some concepts from my initial proposal, the final result of the project was a complete success and I would recommend the

blockchain technology to any enterprise that wants to upgrade the safety of their internal online infrastructure.

1.0 Introduction

1.1. Background

I chose to undertake this project because I wanted a challenge in a new software development field. I came up with the idea of a general e-vote system that can be reused in various domains when I noticed how the NCI student council elections have been undertaken. Blockchain technologies became more and more common in the current IT world as hundreds of organizations from different backgrounds started adopting them. In my opinion, an application that provides a secure and easy to access mode of voting and gathering data in the form of polls would have a huge potential market. My idea has slightly changed throughout this project. I started as creating a voting application that will have an election type voting system, but it changed when I realized that companies all around the world spend millions for developing such tools that are used in their daily routine. A potential customer will be willing to invest in a customizable system that empowers the key stakeholders of an organization to make all sorts of decisions.

1.2. Aims

My project aims to create a general e-vote system that allows the user to anonymously submit his vote for a poll without having his privacy at risk. Also the application aims to provide quick results unlike the "traditional way of voting" which might be very time consuming. Lastly and most importantly the vote must be 100% secured from all types of attacks. The main value of this application comes from the very high reusability of the system. Any organization could just simply assign a member to initiate the poll, then any stakeholder allowed to vote will be able to do so in a matter of seconds.

1.3. Technology

Some of the technologies I used in the development of this project are:

NEAR - Used this blockchain technology to create my smart contracts, to implement them and add functionality to the "NewPoll", and "PollingStation" sections of my project.

REACT - I used this technology mostly for my fron-end.

Node.js - Used multiple of its frameworks like AngularJS and ExpressJS to create some of the functionality in my project, also they came in very handy when creating the front-end of the application.

Yarn - Imported this package manager in Visual studio Code to act as my main project manager and handle several blockchain related tasks.

JSON. - I will use this file format for the database side of the application.

1.4. Structure

In this document I will outline the technologies I used in developing this project and break down all the main features of the application. I will also outline the testing methods I used to validate the functionality of the features. Lastly, I will provide my general thoughts on blockchain technologies and how this project has been executed.

2.0 System

2.1. Requirements

The main requirements of my system are:

- To provide the user with a way of creating a NEAR account in order to access the smart e-voting system
- To provide a list of all the active polls (polling station)
- To allow the moderator to delete a poll
- To allow the moderator to add a new poll
- To allow the moderator to provide different types of information for the polls
- To allow a registered member to vote in a poll
- To prevent an member voting more than once
- To encrypt the information in such a way that it can't be altered
- To allow an user to logout

2.1.1. Functional Requirements

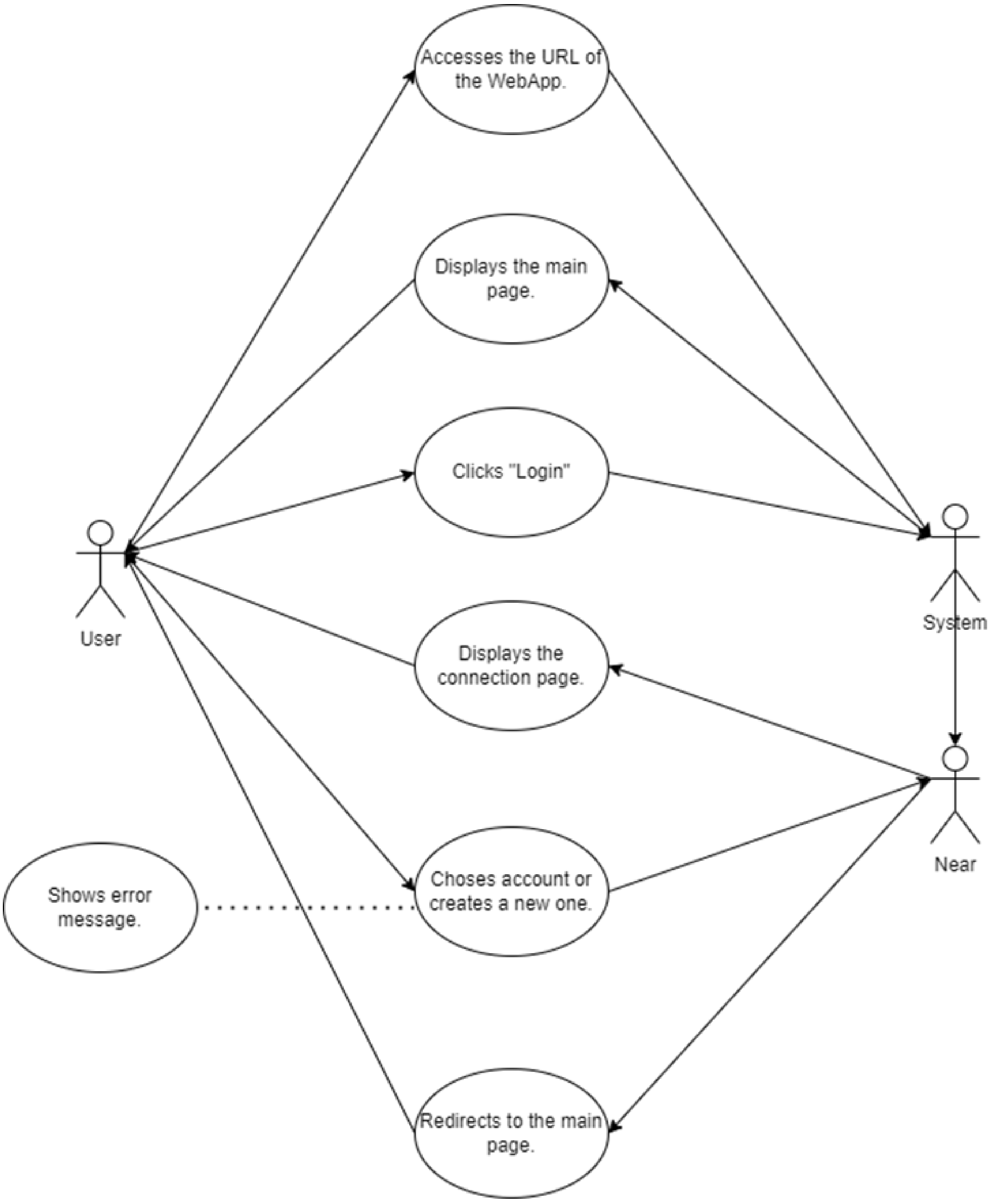
Requirement 1:User Login

Description & Priority: This requirement describes the process in which a user logs in using his near account. If he doesn't have an account, the user can choose to create one with his email address. This requirement is the most important one because in order for an user to access any of the functionality of the application, he must be logged in.

Scope: The scope of this use case is to register the user.

Description: This use case describes the process of registration.

Use Case Diagram


Accesses the URL of the WebApp.
Displays the main page.
Clicks "Login"
Displays the connection page.
Choses account or creates a new one.
Shows error message.
Redirects to the main page.
User
System
Near

Precondition: Users must have a valid email address.

Activation: This use case starts when an user accesses the application's URL

Main flow

1. The user accesses the URL.
2. The system displays the home page
3. The user clicks the "Login" button

4. The system send the information over to the NEAR platform/
5. Near displays the connection page.
6. The user selects the account.
7. Near redirects the user back to the main page.

Alternate flow

A1: User doesn’t have an account

1. The user clicks the register button.
2. Near displays the registration page.
3. The user creates a new account with his email address.
4. Near redirects the user back to the connection page.
5. The use case continues at position 5 of the main flow.

Exceptional flow

E1: Internet connection lost

1. Internet connection cuts out while a user is logging in
2. The system displays an error message
3. The login process is put on hold
4. When the internet connection comes back, the user will be redirected to the 2nd step of the main flow.

E2: User doesn’t have a valid email address.

1. Users must create an email address.

Termination

The use case is terminated after the user has successfully logged in and has been redirected to the main page.

Post condition

The local storage monitors every action performed by the user.

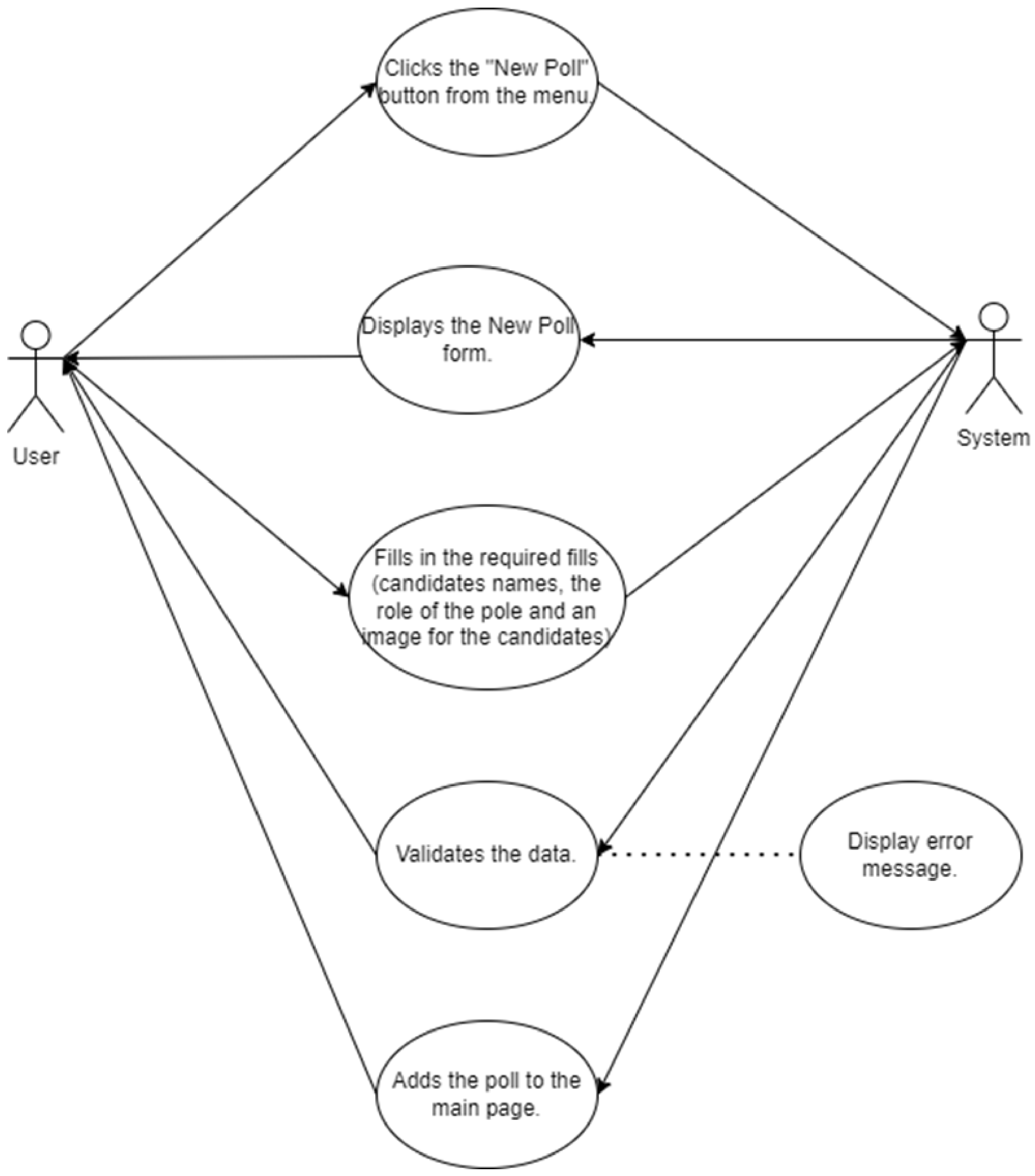
Requirement 2: Add Poll

Description & Priority: This requirement describes the process in which a user adds a poll for everyone to see. In the future this feature is going to be restricted to only administrator accounts. The user must fill in some details such as: the candidates names, provide an image for the two candidates (optional), and the role (short description) of the poll.

Scope: The scope of this use case is to add a new poll on the main page.

Description: This use case describes the process of adding a new poll.

Use Case Diagram


Clicks the "New Poll" button from the menu.
Displays the New Poll form.
Fills in the required fills (candidates names, the role of the pole and an image for the candidates)
Validates the data.
Display error message.
Adds the poll to the main page.
User
System

Precondition: Users must be already logged in.

Activation: This use case starts when the user clicks the "New Poll" button from the main menu.

Main flow

1. The user clicks the "New Poll" button.
2. The system displays the "New Poll" form.
3. The user types in the required information (candidates names and the role of the pool)
4. The system validates the information.
5. The system adds the new poll to the main page.

Alternate flow

A1: User doesn't provide the required information.

1. The user doesn't fill in the form.
2. The system displays an error message.
3. The user is redirected back to the "New Poll" form.
4. The use case continues at position 3 of the main flow.

Exceptional flow

E1: Internet connection lost

1. Internet connection cuts out while a user is creating the poll/
2. The system displays an error message
3. The poll creation process is put on hold
4. When the internet connection comes back, the user will be redirected to the main page of the application.

E2: User doesn't have the permission to add the poll.

1. The user doesn't see the "Create" poll option.
2. The user must contact a moderator.
3. The moderator will assign the user with the option of adding a new poll

E3: User doesn't have an account.

1. Users must create an account following the A1 flow from the first Use Case.

Termination: The use case is terminated when the new poll has been successfully added to the main page.

Post condition: The new poll is saved in the local storage and it keeps track of every vote received.

Requirement 3: Vote

Description & Priority: This requirement describes the process in which a user can Vote. Firstly he needs to locate the poll he/she would like to vote from the main page list. Then the user must click the "Go to Poll" button beside the poll's name. Once the polling station page is loaded, the user can see the candidates competing for the respective role and he can choose to vote for one of them. This is one of the most important features of the application, because it handles most of the functionality.

Scope: The scope of this use case is to allow a user to vote in a poll.

Description: This use case describes the process in which a user selects a poll from the list, then votes for one of the candidates.

Use Case Diagram

Precondition: Users must have an account and there must be at least one poll in the list.

Activation: This use case starts when a user clicks the "Go to Poll" button for one of the Polls from the list.

Main flow

1. The user picks a poll.
2. The user clicks the "Go to Poll" button/
3. The system displays the Polling station interface.
4. The user picks a candidate.
5. The user clicks the "Vote" button underneath the candidate.
6. The system updates the vote count.
7. The system displays the new vote count.

Exceptional flow

E1: Internet connection lost

1. Internet connection cuts out before the user can submit his/her vote.

2. The system displays an error message

3. The voting process is put on hold

4. When the internet connection comes back, the user will be redirected to the main page of the application and he needs to start over from the step 1 of the main flow.

E2: User has already voted in that poll.

1. If the user voted already, the vote buttons are deactivated.

Termination: The use case is terminated after the new vote count is displayed on the screen.

Post condition: The local storage updates the information of the voter, so he/she would not be able to vote again in the same poll.

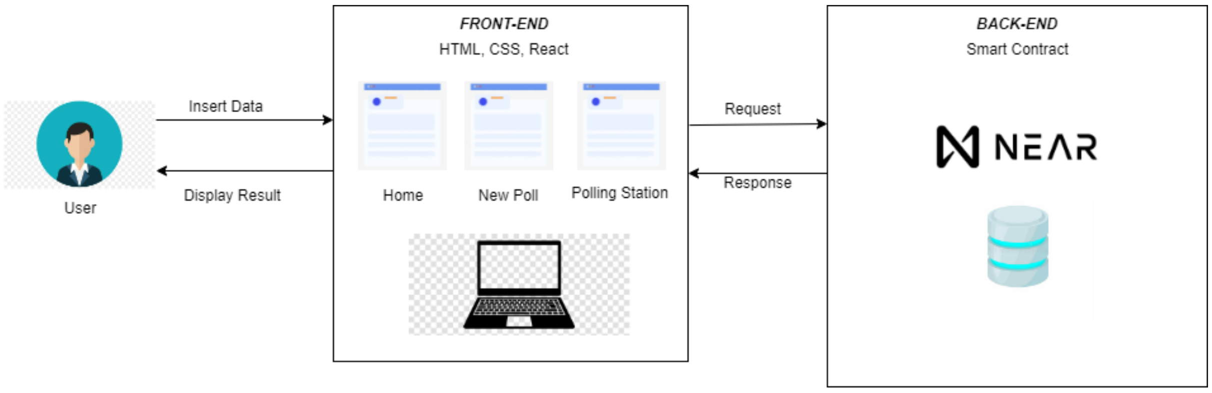
2.2. Design & Architecture

The system is formed from a front-end I created using my JavaScript knowledge and a smart contract I created using a near template which I imported using the Yarn package manager. For the moment I have 3 main components: the Home component which represents a list of all the polls added to the system, the NewPoll component which is the component that contains the function which adds a new poll to the system, and the PollingStation component which contains the voting function. My smart contract is formed of:

- View methods which do not change the state of the blockchain and have the purpose of pulling and reading information from the blockchain.

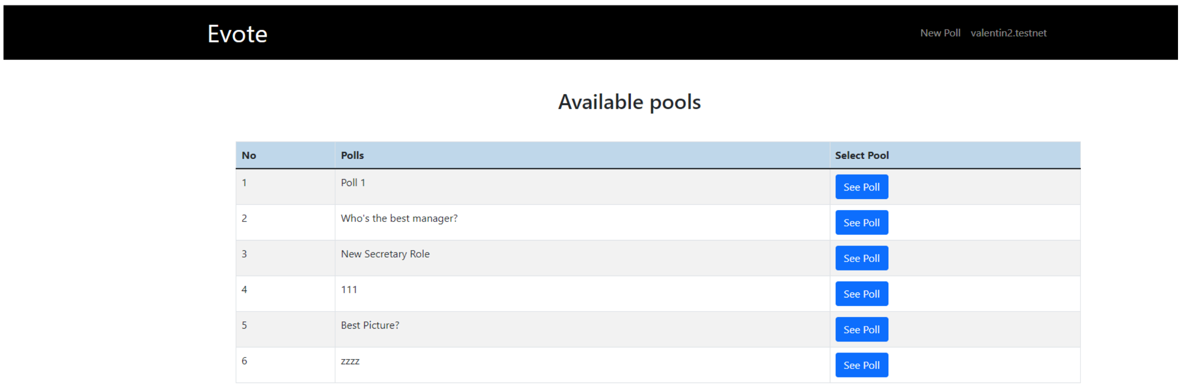
- Change methods which can add or modify information to the blockchain and because of that the blockchain changes its state everytime one of those methods is being used.

One of the first steps I had to undertake was to initialize the smart contract by establishing a connection to the NEAR testnet and wallet based account.



2.3. Graphical User Interface (GUI)

The list of polls:

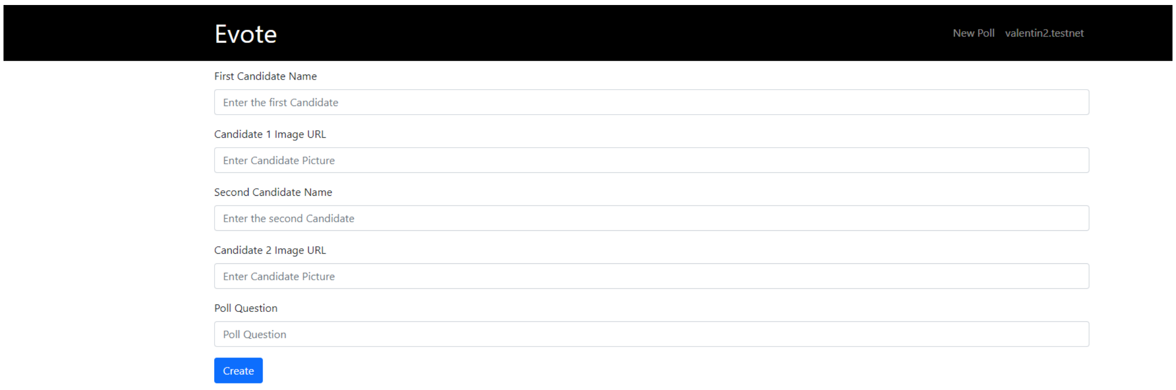

Evote

New Poll valentin2.testnet

Available pools

No	Polls	Select Pool
1	Poll 1	See Poll
2	Who's the best manager?	See Poll
3	New Secretary Role	See Poll
4	111	See Poll
5	Best Picture?	See Poll
6	zzzz	See Poll

New Poll:


Evote

New Poll valentin2.testnet

First Candidate Name

Enter the first Candidate

Candidate 1 Image URL

Enter Candidate Picture

Second Candidate Name

Enter the second Candidate

Candidate 2 Image URL

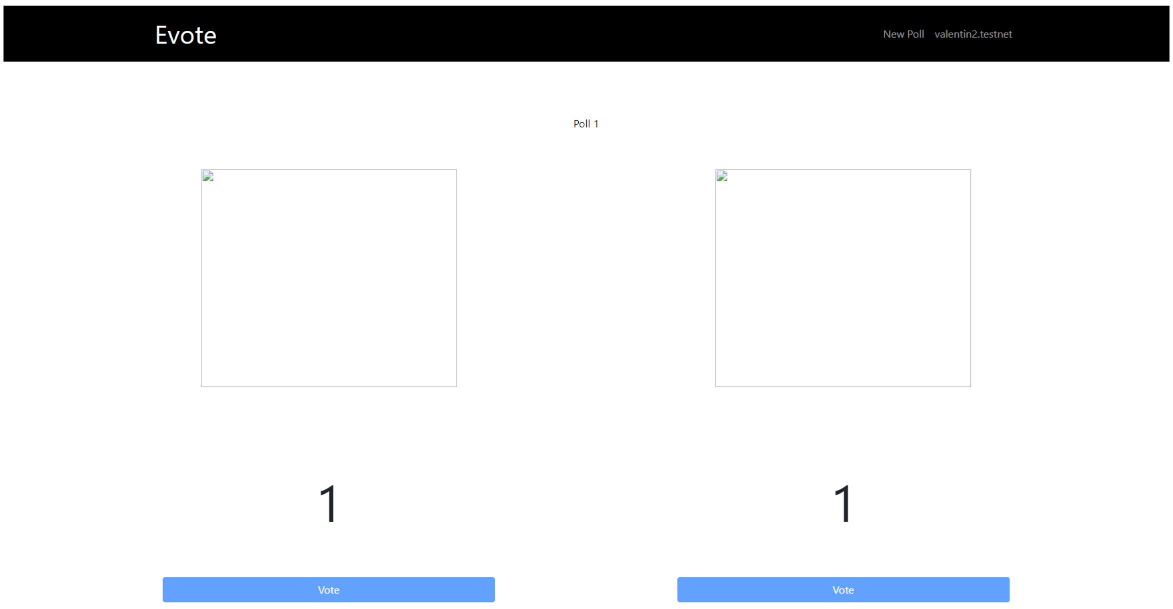
Enter Candidate Picture

Poll Question

Poll Question

Create

Example of a poll:


Evote

New Poll valentin2.testnet

Poll 1

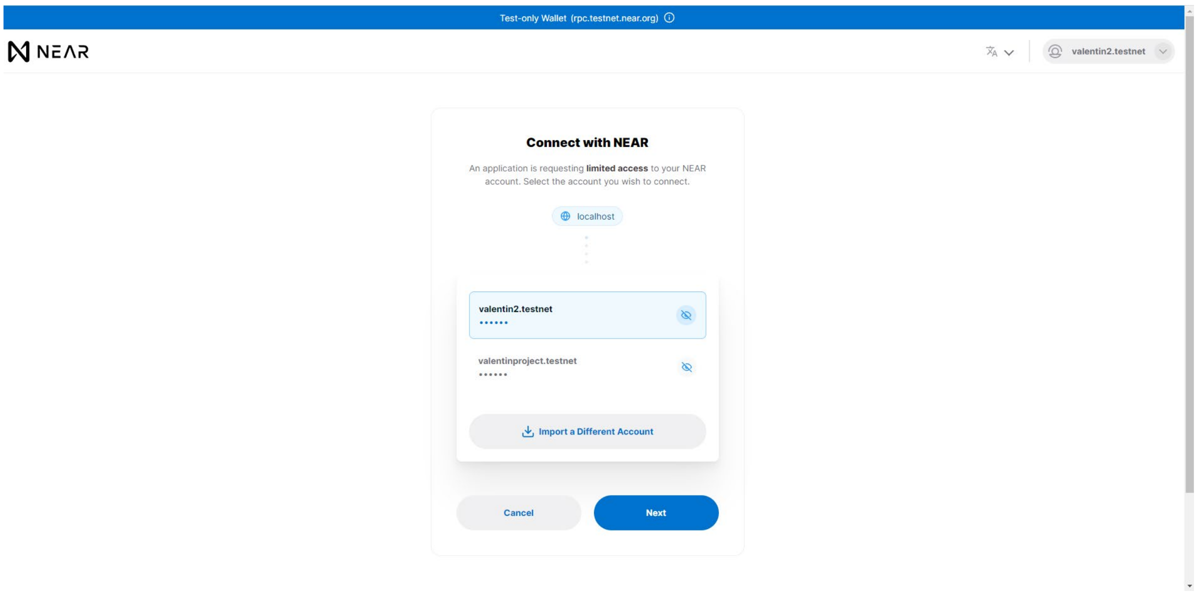
1

Vote

1

Vote

LogIn screen:



Connect with NEAR

An application is requesting **limited access** to your NEAR account. Select the account you wish to connect.

2.4. Testing

The two main testing methods I have used for validating my projects are Unit Testing and End-To-End Testing.

Unit Testing:

Test Case ID	T1
Teste Component	Add Poll
Test Description	Test the ability of a user to create a new poll.
Test Priority	Very Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Test Browser**	**Comments**
Step1 : Click the “Add Poll” button.	Click	The "New Poll" page successfully loads.	The "New Poll" page successfully loads.	Google Chrome	The loading time was in between normal parameters.
Step 2: Fill in the required fields.	Text	The user will be able to type in the requested	The user was able to type in the requested	Google Chrome	There were no issues during this step.

		data.	data.		
Step 3: Click the “Create poll” button.	Click	The user will receive the confirmation that the poll has been added to the polling list.	The user received the confirmation that the poll has been added to the polling list.	Google Chrome	The loading time was between 5 to 7 seconds longer than expected.

Test Case ID	T2
Teste Component	Login
Test Description	Test the ability of a user to login using a Near account.
Test Priority	Very Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Test Browser**	**Comments**
Step1 : Click the Login button from the main page.	Click	The near login page is displayed.	The near login page is displayed.	Google Chrome	The loading time was in between normal parameters.
Step 2: Provide the account details.	Text	The account selection page will be displayed.	The account selection page will be displayed.	Google Chrome	There were no issues encountered during this step.
Step 3: Select account and press “Connect”.	Clicks	The user will be successfully logged into the application.	The user will be successfully logged into the application.	Google Chrome	The loading time was between 5 to 7 seconds longer than expected.

Test Case ID	T3
Teste Component	Vote
Test Description	Test the ability of a user to vote in a chosen vote if he/she hasn’t already done so.
Test Priority	Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Test Browser**	**Comments**
Step1 : Select a poll by clicking the “Vote” button beside it.	Click	The polling station page is displayed.	The polling station page is displayed.	Google Chrome	The loading time was in between normal parameters.
Step 2: Click the vote button underneath the chosen candidate	Click	The vote is counted and the live voting results are being displayed.	The vote is counted and the live voting results are being displayed.	Google Chrome	The loading time was over 10 seconds more than expected.

End-To-End Testing:

Test Case ID	Step	Description	Status	Expected Result	Actual Result	Comments
FR-01	1	User accesses the URL of the web application	Success	The main page loads successfully.	The main page loads successfully.	There were no issues encountered performing

						this action.
FR-02	2	User clicks the “login” button.	Success	The near login page is displayed.	The near login page is displayed.	There were no issues encountered performing this action.
FR-03	3	User provides the wrong details.	Success	The system displays an error message.	The system displays an error message.	There were no issues encountered performing this action.
FR-04	4	User provides the correct login details.	Success	The user has been successfully logged in the application.	The user has been successfully logged in the application.	There were no issues encountered performing this action.
FR-05	5	User clicks the “Add Poll” button,	Success	The “New Poll” page is being displayed.	The “New Poll” page is being displayed.	There were no issues encountered performing this action.
FR-06	6	User doesn’t fill in all the empty fields.	Success	The system displays an error message.	The system displays an error message.	There were no issues encountered performing this action.
FR-07	7	User fills in all the required fields.	Fail	The new poll is being added to the “Polls list” and all the details provided by the user are being displayed in	The new poll has been created but the candidate images are not loaded in the new polling	The only pieces of data that are not being loaded on to the polling station are the pictures of the two candidates.

				the new poll.	station.	
FR-08	8	User accesses a poll by clicking the “Vote” button beside it.	Success	The polling station for the respective poll is being displayed.	The polling station for the respective poll is being displayed.	There were no issues encountered performing this action.
FR-09	9	User clicks the vote button underneath the chosen candidate	Success	The voting count is updated and displayed.	The voting count is updated and displayed.	The loading time was 10 seconds longer than the expected loading one.
FR-10	10	User clicks on the logout button.	Success	The user has been successfully logged out.	The user has been successfully logged out.	There were no issues encountered performing this action.

2.5. Evaluation

After I conducted all my tests I came to the conclusion that all the features are working as they are supposed to with the exception of one (loading the candidate pictures). The loading time has been between normal parameters with the exception of two instances: when adding a new poll which is about 7 seconds longer and when voting for a poll which is 10 seconds longer than the expected loading time. After I researched this issue, I came to the conclusion that the delays are caused by the slow time in which the application establishes the connection with the NEAR servers due to my free to use account.

3.0 Conclusions

The application E-Vote is a blockchain based voting system that can ensure full security to the integrity of the vote as well as the users taking part. The first primary strength of E-Vote is that it is blockchain based. Blockchain has become a staple in the information security industry for its robustness, reliability and security. By centering my application behind blockchain any vote that takes place on my application will be 100% trustworthy and safe from any potential cyber attacks. Another strength of my application is transparency. Users can see live polling results. This can ensure all users of the app of a fair and impartial voting system. The benefits of the transparency is that it shows a degree of efficiency as well as providing trust between the students and the admin who is in control of the app.

A very big strength of my application is that voter fraud can also be mitigated. A single user cannot vote more than once. This can stop users' attempts to tamper with votes that can be very important to many users. This aspect is important because without it a voting system becomes meaningless.

I believe that companies in the future will see my Application “E-Vote” as a tool to provide safe and secure polling within large organisations as well as providing anonymity to its many users. This application could also have a future in government for voting going forward.

The limitations that I have identified was time that I could spend on developing this project. With many other assignments going on it was hard to be able to allocate time to develop this project at certain points throughout the year.

4.0 References

Binance Academy. 2022. What Is NEAR Protocol (NEAR)? | Binance Academy. [online] Available at: <https://academy.binance.com/en/articles/what-is-near-protocol-near>

Gondek, C., 2022. How Blockchain Technology Transforms Voting. [online] Originstamp.com. Available at: <https://originstamp.com/blog/how-blockchain-voting-systems-transform-the-way-we-vote/

Other coding Links I use in the development of my project:

https://docs.near.org/docs/develop/contracts/sandbox

https://www.youtube.com/watch?v=sbcuseMGw9Q&t=6s&ab_channel=NEAR

https://www.youtube.com/watch?v=Uuj5eWaeAxI&ab_channel=NEAR

https://near.github.io/near-api-js/classes/contract.contract-1.html

5.0 Appendices

5.1. Project Proposal

1.0 Introduction

1.1. Background

I chose to undertake this project because I wanted a challenge in a new software field. My goal is to get a career in software engineering and learning something new was also one of my priorities when picking this type of project. Another reason is that blockchain technologies became more and more common in the current IT world and because this type of technology is relatively new, there are not too many people who can master one of its fields. I came up with this idea after watching how the NCI student council elections have been undertaken. Because we live in a world of covid, I consider that a fair and secured e-vote system is a must for every big institution or organization.

1.2. Aim

My project aims to create a general e-vote system that allows the user to anonymously submit his vote for a poll without having his privacy at risk. Also the application aims to provide quick results unlike the “traditional way of voting” which might be very time consuming. Lastly and most importantly the vote must be 100% secured from all types of attacks. Even if my goal is to create a general system that can be reused for most of the projects, to highlight how this system would work in the real world, I picked the student council voting in NCI as my study case. The administrator of the application must be able to create a new poll for the specific role, the applicants must be able to enroll themselves as candidates and any NCI student must be able to submit a vote for each poll. In my proposal, I also stated some side goals which have the role to provide an idea of how this e-vote system might be incorporated inside of another application. Some of the side tasks are: create a newsfeed for students and a sign-up system for the clubs/societies in NCI. After I spent several weeks researching these topics, I decided to focus on the main feature of the application (the blockchain e-vote system) and for the other side tasks complete as much as time allows me to.

1.3. Technology

Some of the technologies I used or I am going to use throughout this project are:

NEAR - Used this blockchain technology to create my smart contracts, to implement them and add functionality to the “NewPoll”, and “PollingStation” sections of my project.

Node js - Used multiple of its frameworks like AngularJS and ExpressJS to create some of the functionality in my project, also they came in very handy when creating the front-end of the application.

Yarn - Imported this package manager in Visual studio Code to act as my main project manager and handle several blockchain related tasks.

JSON. - I will use this file format in the future when I am going to implement my own database, so I don’t have to rely on the local storage to store my data.

1.4. Structure

In this document I am going to talk about what technologies I have used to create some of the features within my project. I will also detail every feature and the reason I choose those approaches instead of the other options. Another focus point in this report is going to highlight the methods I used to gather my requirements. Lastly, I am going to give some testing options for my system.

2.0 System

2.1. Requirements

The first requirement needed for the user to access the system is to have a NEAR account. In case he doesn’t have one, he/she can create one for free in a matter of seconds, everything needed for that is a valid email address and an internet connection. The system itself is pretty simple, so an experienced user should get a solid grasp on how to use the polls functions without committing any errors in less than half of an hour.

2.1.1. Functional Requirements

2.1.1.1. Requirement 1:User Login

2.1.1.2. Description & Priority

This requirement describes the process in which a user logs in using his near account. If he doesn’t have an account, the user can choose to create one with his email address. This requirement is the most important one because in order for an user to access any of the functionality of the application, he must be logged in.

2.1.1.3. Use Case

Scope

The scope of this use case is to register the user.

Description

This use case describes the process of registration.

Use Case Diagram

Precondition

Users must have a valid email address.

Activation

This use case starts when an user accesses the application’s URL

Main flow

1. The user accesses the URL.
2. The system displays the home page
3. The user clicks the “Login” button
4. The system send the information over to the NEAR platform/
5. Near displays the connection page.
6. The user selects the account.
7. Near redirects the user back to the main page.

Alternate flow

A1: User doesn’t have an account

1. The user clicks the register button.
2. Near displays the registration page.
3. The user creates a new account with his email address.
4. Near redirects the user back to the connection page.
5. The use case continues at position 5 of the main flow.

Exceptional flow

E1: Internet connection lost

1. Internet connection cuts out while a user is logging in

2. The system displays an error message

3. The login process is put on hold

4. When the internet connection comes back, the user will be redirected to the 2nd step of the main flow.

E2: User doesn't have a valid email address.

1. Users must create an email address.

Termination

The use case is terminated after the user has successfully logged in and has been redirected to the main page.

Post condition

The local storage monitors every action performed by the user.

2.1.1.4. Requirement 2:Add Poll

2.1.1.5. Description & Priority

This requirement describes the process in which a user adds a poll for everyone to see. In the future this feature is going to be restricted to only administrator accounts. The user must fill in some details such as: the candidates names, provide an image for the two candidates (optional), and the role (short description) of the poll.

2.1.1.6. Use Case

Scope

The scope of this use case is to add a new poll on the main page.

Description

This use case describes the process of adding a new poll.

Use Case Diagram

Precondition

Users must be already logged in.

Activation

This use case starts when the user clicks the "New Poll" button from the main menu.

Main flow

1. The user clicks the "New Poll" button.

2. The system displays the “New Poll” form.
3. The user types in the required information (candidates names and the role of the pool)
4. The system validates the information.
5. The system adds the new poll to the main page.

Alternate flow

A1: User doesn’t provide the required information.

1. The user doesn’t fill in the form.
2. The system displays an error message.
3. The user is redirected back to the “New Poll” form.
4. The use case continues at position 3 of the main flow.

Exceptional flow

E1: Internet connection lost

1. Internet connection cuts out while a user is creating the poll/
2. The system displays an error message
3. The poll creation process is put on hold
4. When the internet connection comes back, the user will be redirected to the main page of the application.

E2: User doesn’t have an account.

1. Users must create an account following the A1 flow from the first Use Case.

Termination

The use case is terminated when the new poll has been successfully added to the main page.

Post condition

The new poll is saved in the local storage and it keeps track of every vote received.

2.1.1.7. Requirement 1:Vote

2.1.1.8. Description & Priority

This requirement describes the process in which a user can Vote. Firstly he needs to locate the poll he/she would like to vote from the main page list. Then the user must click the "Go to Poll" button beside the poll's name. Once the polling station page is loaded, the user can see the candidates competing for the respective role and he can choose to vote for one of them. This is one of the most important features of the application, because it handles most of the functionality.

2.1.1.9. Use Case

Scope

The scope of this use case is to allow a user to vote in a poll.

Description

This use case describes the process in which a user selects a poll from the list, then votes for one of the candidates.

Use Case Diagram

Precondition

Users must have an account and there must be at least one poll in the list.

Activation

This use case starts when a user clicks the "Go to Poll" button for one of the Polls from the list.

Main flow

1. The user picks a poll.
2. The user clicks the "Go to Poll" button/
3. The system displays the Polling station interface.
4. The user picks a candidate.
5. The user clicks the "Vote" button underneath the candidate.
6. The system updates the vote count.
7. The system displays the new vote count.

Exceptional flow

E1: Internet connection lost

1. Internet connection cuts out before the user can submit his/her vote.

2. The system displays an error message

3. The voting process is put on hold

4. When the internet connection comes back, the user will be redirected to the main page of the application and he needs to start over from the step 1 of the main flow.

E2: User has already voted in that poll.

1. If the user voted already, the vote buttons are deactivated.

Termination

The use case is terminated after the new vote count is displayed on the screen.

Post condition

The local storage updates the information of the voter, so he/she would not be able to vote again in the same poll.

2.2. Design & Architecture

The system is formed from a front-end I created using my JavaScript knowledge and a smart contract I created using a near template which I imported using the Yarn package manager. For the moment I have 3 main components: the Home component which represents a list of all the polls added to the system, the NewPoll component which is the component that contains the function which adds a new poll to the system, and the PollingStation component which contains the voting function. My smart contract is formed of:

- View methods which do not change the state of the blockchain and have the purpose of pulling and reading information from the blockchain.

- Change methods which can add or modify information to the blockchain and because of that the blockchain changes its state everytime one of those methods is being used.

2.3. Implementation

An example of a change method that modifies the state of the blockchain I used in my project is:

```
export function addVote(prompt:string,index:i32):void{

  if(VoteArray.contains(prompt)){

   let tempArray=VoteArray.getSome(prompt)

   let tempVal=tempArray[index];

   let newVal=tempVal+1;

   tempArray[index]=newVal;

   VoteArray.set(prompt,tempArray);

  }else{

   let newArray=[0,0];

   newArray[index]=1;

   VoteArray.set(prompt,newArray);

  }

}
```

In this function if there are existing values for the prompt, the system is going to retrieve one value, add 1 to it, replace the index with the new values and save it to the blockchain. In the “else” statement I specified that if the prompt doesn’t exist, the system must create a new instance of that prompt.

2.4. Graphical User Interface (GUI)

Home page which contains the list of polls. As you can see I currently have only one poll in my list.

The New poll form that a user must complete in order to create a new poll.

The Polling station feature that allows an user to vote for a particular pool. As you can see my pool has the role: “NCI student council president” and currently has only one vote for candidate 1.

2.5. Testing

Unit Testing - Smoke Testing

Because this project is still a prototype, I believe that the ebay testing approach is running weekly smoke tests. This testing method helped me to determine if the main features of the application were working as expected and if there were no showstoppers inside the project.

Integration Testing - Because the project is locally hosted for the moment, I could not run any integration test so far. But next year when the project is deployed, I am planning to use Selenium WebDrive to run a series of tests to check the performance of the web application.

End User testing - Because this project is in the prototype phase, I could not run any end user testing.

2.6. Evaluation

Most of the features of the prototype seem to behave as expected. The user is able to create a poll, to vote, and to view the live results of that particular pool. Also the system doesn’t allow more than one vote per account. The only problem I currently have is to display the candidates name on the Polling Station page. Because the prototype is located on a localhost server right now I can not perform too many evaluations. But from a quick smoke test I got to the conclusion that the system is stable.

3.0 Conclusions

With online voting becoming more and more popular these days, there are also a number of issues raised regarding the security and privacy of the votes. I believe that blockchain technology can fix most of these problems. In order for an e-vote system to be usable, the system should check for voter’s details such as: are the voters the person they claim to be, has the user the right to vote and also check if the respective user hasn’t already voted. The accessibility of a blockchain e-vote system can be considered both an advantage and disadvantage. Because of its nature, the voter can submit his vote from anywhere as long as he has a device with an internet connection, but there could be cases where an eligible person for the vote doesn't have access to those requirements. Some other pros for the blockchain evote system are: the security, the privacy of the voters, it is very transparent and everyone can see how the election went. One of the big disadvantages of this project is the technology required in this type of system. For example, if a user loses his private key during the process, his vote can be denied by the software. In conclusion, I believe that this type of technology is an improved version of the traditional way of voting and even if it is not perfect at this very moment in time, with the fast growth of technology, there are going to be ways to upgrade and polish it to a much better state.

4.0 Further Development or Research

During the next semester I am planning to fix some of the features and upgrade them. The first step would be to implement different types of user accounts, so only the "administrator" users would have the option to create a poll and everybody would have the right to vote. Another improvement that I consider a must is an account validation which involves the prevention of an user holding multiple accounts with different student emails. Lastly, I am planning to create a JSON database to manage my data instead of relying on the local storage.