

Institiúid Teicneolaíochta Cheatharlach



At the Heart of South Leinster

Online Voting System

Functional Specification

Name: Mark Kelly

Student Number: C00198041

Course: Bachelor of Science (Honours) Software Development

Supervisor: Dr. Lei Shi

Date: 18th April 2018

Abstract

The purpose of this document is to provide an overview of the functional and nonfunctional requirements of the suVote online voting system. The core functionality is outlined for all the main components of the system. The non functional requirements are also outlined. An iterative plan detailing the implementation of the functionalities over three iterations is also provided.

Table of Contents

1. Introduction	3
2. Project Concept	4
3. Target Market	4
4. System Architecture Diagram	5
5. Use Case Diagram	5
6. Brief Use Cases	6
6.1 Brief use case - CRUD Election (Election Admin)	6
6.2 Brief use case - CRUD Ballot (Election Admin)	6
6.3 Brief use case - CRUD Candidate (Election Admin)	7
6.4 Brief use case - Preview Election (Election Admin)	7
6.5 Brief use case - View Results (Election Admin)	8
6.6 Brief use case - View Results (Voter)	8
6.7 Brief use case - Login (Election Admin)	8
6.8 Brief use case - Login (Voter)	9
6.9 Brief use case - Logout (Election Admin)	9
6.10 Brief use case - Logout (Voter)	9
6.11 Brief use case - Vote	10
6.12 Brief use case - Register (Voter)	10
7. Supplementary Specification	10
7.1 Functionality	10
7.2 Usability	11
7.3 Reliability	11
7.4 Performance	11
7.5 Supportability	11
7.6 +	11
8. Iteration Plan	11
8.1 Iteration one	11
8.2 Iteration two	12
8.3 Iteration three	12
9. Conclusion	12

1. Introduction

This functional specification document details the functional and nonfunctional requirements for our online voting system, a web application called suVote. The following document is detailed under the following headings:

- Project concept
 - In this chapter we will discuss the main concept behind this project. We will explain what the project is and what it's supposed to do.
- Target market
 - This chapter will identify the target market for our proposed system. We will outline how our web application fits into the current market.
- System architecture diagram
 - This chapter provides an overview of the system architecture.
- Use case diagram
 - In this chapter we provide a use case diagram showing the main functionality of our system.
- Brief use cases
 - This section will provide a brief use case for each of the main functionalities of our system.
- Supplementary specification
 - Here we will detail the functional and nonfunctional requirements of our system.
- Conclusion
 - The final chapter will make a conclusion based on the functional and nonfunctional requirements for our system.

2. Project Concept

suVote will be an online voting platform for IT Carlow students to use to vote in IT Carlow Students Union elections and referenda. suVote will provide the opportunity for students who are not on campus on the day of voting, for example students who are out on placement, to register and vote online. There will be two types of users able to interact with the web application, election administrators and voters (students).

Election administrators will be able to login to the web application, create and manage elections, ballots, candidates. The election administrator will be able to upload an Excel file with the email addresses of the allowed voters on it. The election administrator will also be able to view past results of elections. Graphs and charts will also be provided to allow the election administrator to view and verify the results of past elections. Election administrators will be able to print off a list of all the voters that have registered to vote online on the day of the election.

Voters will be allowed to register with the web application by the election administrator and be linked to an election to allow them to vote on that particular election. Registered voters will then be able to login and view information on the candidates for the upcoming election. On the day of the election, voters will receive a reminder email and be able to login and cast their vote. After the election is over, voters will be able to login so that they can view and verify the results of the election. The only way a voter can register to vote for an election is if they are on the list of allowed voters. Voters will have to register again for future elections.

3. Target Market

We send emails instead of a handwritten letter, we read Kindles instead of books, we use tablets instead of pen and paper: and yet, voting is still mostly left to the old-fashioned paper ballot system. Voting technology has essentially remained at a standstill for decades.

The current target market for our proposed system will be third level institutions and small organisations. If we manage to implement our proposed online voting system using blockchain technology to provide ultimate transparency, accountability and trust, our target market will greatly increase to national and perhaps international organisations.

4. System Architecture Diagram

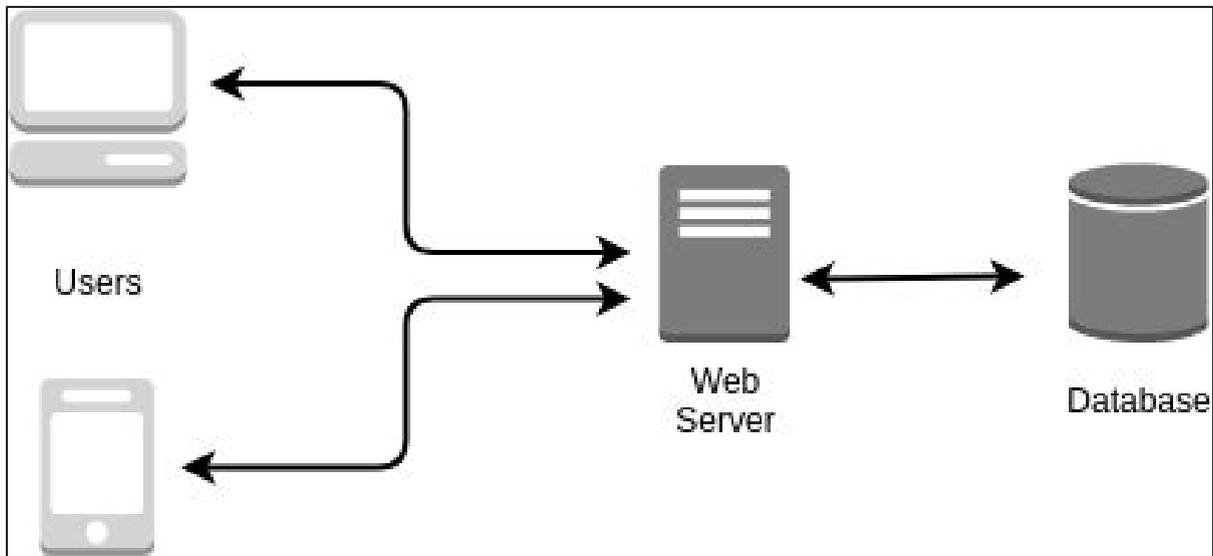


Fig. 4.1 System Architecture Diagram

5. Use Case Diagram

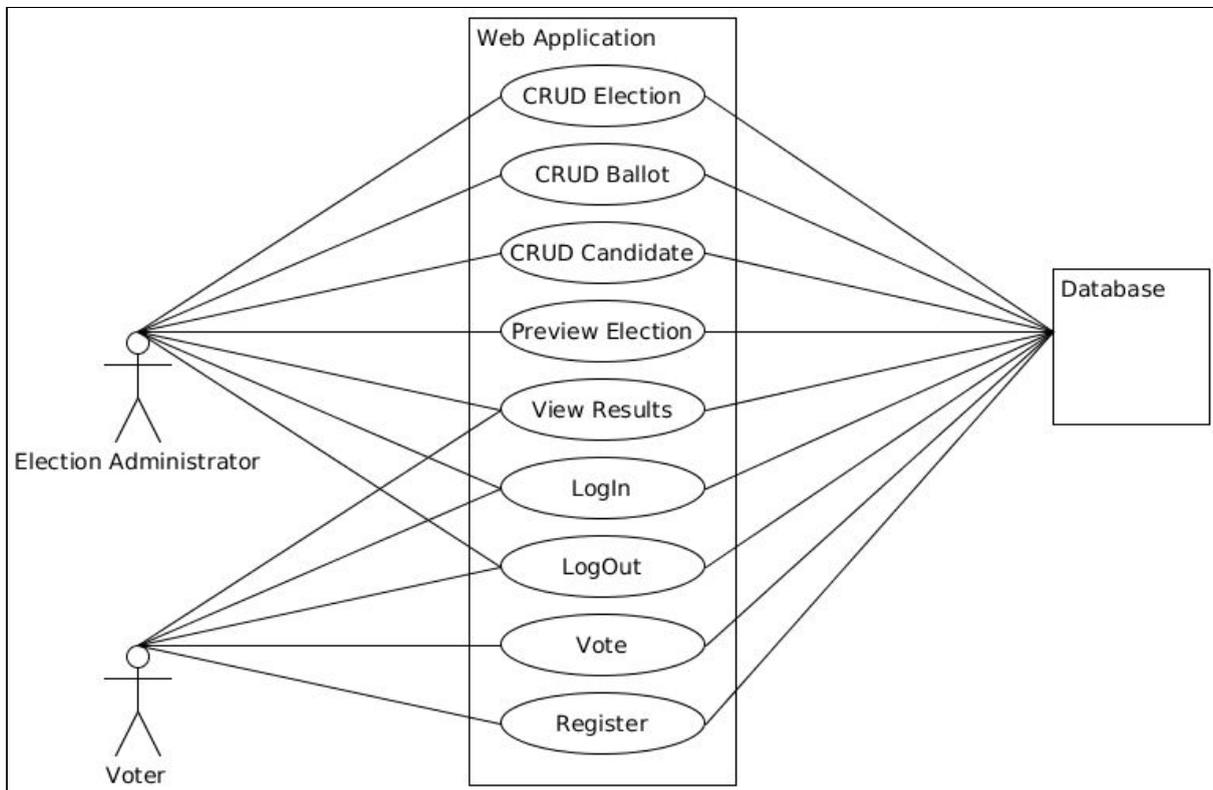


Fig. 5.1 Use Case Diagram

6. Brief Use Cases

6.1 Brief use case - CRUD Election (Election Admin)

Actors	Election Administrator, Database.
Preconditions	The election administrator has successfully logged into the system and is on the admin page.
Activity	This use case begins when the election administrator wishes to create an election. The election administrator selects "Add Election" button. The web application displays the "Election" page. The election administrator enters the details of the election and clicks "Save". The details of an election include a name, introduction, start date, end date, and allowed voters. The web application validates the details.
Consequence	Upon validation, the system stores the details of the election in the database. The system indicates to the election administrator that the election was created successfully. The system displays the "Election" page. An election has been created.

Fig. 6.1 CRUD Election

6.2 Brief use case - CRUD Ballot (Election Admin)

Actors	Election Administrator, Database.
Preconditions	The election administrator has successfully logged into the system. An election has been created and the election administrator is on the "Ballot" page.
Activity	This use case begins when the election administrator wishes to create a ballot. The election administrator selects the "Add Ballot" button. The system displays the "Add Ballot" page. The election administrator enters the details of the ballot and selects the "Save" button. The details of a ballot include the election for the ballot, a description, an introduction. The system validates the details.
Consequence	Upon validation, the system stores the details of the ballot in the database. The system indicates to the election administrator that the question was added successfully. The system displays the "Ballot" page. A ballot has been created.

Fig. 6.2 CRUD Ballot

6.3 Brief use case - CRUD Candidate (Election Admin)

Actors	Election Administrator, Database.
Preconditions	The election administrator has successfully logged into the system. An election has been created. A ballot has been created and the election administrator is on the "Admin" page.
Activity	This use case begins when the election administrator wishes to add a candidate for an election.. The election administrator selects the "Add Candidate" button. The system displays the "Add Candidate" page. The election administrator enters the details of the candidate and selects the "Save" button. The details of an option include, the ballot, first name, last name, institution and profile picture. The system validates the details.
Consequence	Upon validation, the system stores the details of the candidate in the database. The system indicates to the election administrator that the option was added successfully. The system displays the "Ballot" page. A candidate has been created.

Fig. 6.3 CRUD Candidate

6.4 Brief use case - Preview Election (Election Admin)

Actors	Election Administrator, Database.
Preconditions	The election administrator has successfully logged into the system and is on the "Admin" page. An election has been created. At least one ballot with at least one candidate has been created.
Activity	This use case begins when the election administrator selects the "Preview Election" button. The system retrieves the details of the election from the database and displays them on the screen.
Consequence	The web administrator can now view the preview of the election.

Fig. 6.4 Preview Election

6.5 Brief use case - View Results (Election Admin)

Actors	Election Administrator, Database.
Preconditions	The election administrator has successfully logged into the system and is on the "Admin" page. The election to view the results on has ended.
Activity	This use case begins when the election administrator selects the "View Results" button. The system retrieves the results of the election from the database. The system displays the results on the screen.
Consequence	The election administrator can now view the results of the election.

Fig. 6.5 View Results

6.6 Brief use case - View Results (Voter)

Actors	Voter, Database.
Preconditions	The voter has successfully logged into the system. The election to view the results on has ended.
Activity	This use case begins when the voter selects the "View Results" button. The system retrieves the results of the election from the database. The system displays the results on the screen.
Consequence	The voter can now view the results of the election.

Fig. 6.6 View Results (Voter)

6.7 Brief use case - Login (Election Admin)

Actors	Election Administrator, Database.
Preconditions	An account has been created for the election administrator. The system is running and the "login" page has been loaded.
Activity	This use case begins when the election administrator wishes to login to the system. The election administrator enters their login credentials. Credentials include email and password. The system validates the credentials with the database.
Consequence	Upon validation, the system displays the election administrator "home" page.

Fig. 6.7 Login (Election Admin)

6.8 Brief use case - Login (Voter)

Actors	Voter, Database.
Preconditions	The voter is registered with the web application. The system is running and the login page has been loaded.
Activity	This use case begins when the voter wishes to login to the web application. The voter enters their login credentials. Credentials include email and password. The system validates the credentials with the database.
Consequence	Upon validation, the system displays the voter "home" page.

Fig. 6.8 Login (Voter)

6.9 Brief use case - Logout (Election Admin)

Actors	Election Administrator, Database.
Preconditions	The election administrator has successfully logged into the system.
Activity	The use case begins when the election administrator wishes to logout of the system. The election administrator clicks the "logout" button. The web application logs the election administrator out.
Consequence	The election administrator has successfully logged out. The system displays the "Home" page.

Fig. 6.9 Logout (Election Admin)

6.10 Brief use case - Logout (Voter)

Actors	Voter, Database.
Preconditions	The voter has successfully logged into the system.
Activity	The use case begins when the voter wishes to logout of the web application. The voter clicks the logout button. The system logs the voter out.
Consequence	The voter has successfully logged out. The system displays the "Home" page.

Fig. 6.10 Logout (Voter)

6.11 Brief use case - Vote

Actors	Voter, Database.
Preconditions	An election has been created. At least one ballot question with at least one candidate has been created. Allowed voters have been added. The election is between the start and end date. The voter has successfully logged into the system and is on the "Vote" page.
Activity	This use case begins when a voter wishes to vote on an election. The system displays the "Vote" page. The voter fills in the ballot. The system validates the ballot.
Consequence	Upon validation, the system saves the vote details to the database. The system displays a thank you message and displays the "Home" page.

Fig. 6.11 Vote

6.12 Brief use case - Register (Voter)

Actors	Voter, Database.
Preconditions	An election has been created. At least one ballot question with at least one candidate has been created. Allowed voters have been added.
Activity	This use case starts when the voter clicks on the registration button on the "Login" page. The system displays the registration page. The voter enters their details. Details for the registration include a password. The system checks if voter is on allowed voters list. The system validates the details. A confirmation email is sent to the voter. The voter confirms the email.
Consequence	Upon validation and confirmation, the system stores the voters login details in the database. The system displays the login page.

Fig. 6.12 Register

7. Supplementary Specification

The purpose of this chapter is to define the supplementary requirements of the suVote system. This chapter goes on to list the requirements that are not captured in the use cases.

7.1 Functionality

The web application must have access to the internet to interact with the database and allow users to use the system.

7.2 Usability

It must be simple for a user to navigate through the web application and find their way around. It must be easy for a voter to fill out and submit a ballot. The web application must be responsive to accommodate for users with smartphones and tablets. The web application must provide the election administrator with clear instructions on how to create a ballot and add voters.

7.3 Reliability

In the event of the database failing to retrieve or validate the data, exceptions must be caught and handled without the web application crashing and displaying built in error pages. The web application must be available 99.9% of the time, especially when an election is live.

7.4 Performance

The web application must be able to handle thousands of request at a given time. The response time from the web application to the database must be fast, so as users are not waiting prolonged periods of time. The web application must be able to recover from a crash automatically, without human interaction.

7.5 Supportability

The web application must be simple to maintain and service. The web application must be modular so as new features can be easily added later. The web application must not be a niche system so as to allow for it to be used for other institutions and organizations.

7.6 +

The web application must allow voters to vote anonymously while providing traceability for each vote. Voters must be able to check that their vote went where they wanted it to go. All personal data should be encrypted and stored within the EU.

8. Iteration Plan

This chapter will break the functionalities to be implemented in this project into three iterations. Each iteration is six weeks long, which gives a total of eighteen weeks to complete the project. The following sub headings will outline what is hoped to be achieved in each individual iteration.

8.1 Iteration one

The first thing we will do in the first iteration is create a Django project. Inside the Django project we will create our suVote application. The next thing we want to do is setup version

control, so we can keep our code safe and be able to roll back versions if we run into any problems while developing. Next we will setup a couple of things on our backend. Firstly we will setup a MySQL database on localhost, create our tables and connect it to the web application. An election administrator with the appropriate permissions will be created at the backend, as election administrators will not be allowed to register themselves. Secondly we will setup SMTP email service for password reset, reminding voters to register, confirmation emails, and sending general notifications related to our web application.

It is hoped to implement the following use cases in the first iteration:

- CRUD Election
- CRUD Ballot
- CRUD Candidate

8.2 Iteration two

The first plan for iteration two is to deploy the web app to pythonanywhere. This may seem a bit early to be deploying, but in my past experience, deploying web apps can be troublesome. I have had to abandon past projects that I was unable to deploy. So I want to get the web app deployed as soon as possible, for peace of mind.

It is hoped to implement the following use cases in the second iteration:

- Vote
- View Results
- Register

8.3 Iteration three

In the final iteration the first plan is to implement the last few use cases. Secondly, the plan is to do add any finishing touches to the GUI. It would also be nice to carry out some penetration testing and time permitting, test the code. I also hope to test a mock election with the help of my classmates.

It is hoped to implement the following use cases in the third and final iteration:

- Preview Election
- Login
- Logout

9. Conclusion

In conclusion, this document has described the core functionalities of the suVote system. A use case diagram identifies the main functionality of the system. The functionalities were highlighted through the means of brief use cases and detailed descriptions of non-functional requirements. The iterative plan has outlined what will be hoped to be developed and achieved in each of the three iterations.