

National College of Information Technology (NACIT)

(Blantyre Campus)

Advanced Diploma in Computing (Level5DC)

Computing Project Final Report

**Title:** Student’s Council Voting System for St John the Baptist University Blantyre

**Name:** Tupochele Mlanjira

**ID:** P00200696

**Email:** [tupomlanjira@gmail.com](mailto:tupomlanjira@gmail.com)

**Phone Number:** +265 885 862 376

**Project Coordinator**: Madam Mwale

**Project Supervisor**: Madam Mwausegha

**Date**: 2nd May, 2025.

# **Abstract**

The present report fully describes the benefits of a student’s council for St John the Baptist University over the traditional manual student council elections. This document contains all features that will help in automating the current manual election processes namely the candidate manifesto chat, timely election reminder notifications, user registrations, election timer, election tabulations from all numeric data, and overall authenticating the running elections. These features will be individually highlighted together with information gathered and methods used to come up with such a system.

# **Overview**

St John the Baptist University Blantyre is one of the three campuses that is also under the flagship of the Daughters and Mary Immaculate (DMI), with the vision of spreading education in the southern parts of Malawi. This campus is located opposite the Airtel Limbe headquarters along Raynor Avenue, and this campus just like the other 2 contributes all its financial earnings to the DMI trust fund that helps in assisting DMI affiliated countries when affected by disasters.

The current system makes sure that three weeks before the elections a committee consisting of an equal number of male and female students is chosen with the rule that they don’t run for office themselves, along with supervising teachers. These committee members produce printed forms that allow interested students to apply for the position they desire, then the committee lets the supervising teachers to review the submitted application forms by checking all the students’ academic performances and conducts. The following week, the candidates set the tone by creating social media forums, and visiting classes after learning hours to get their fellow students to vote for them. The committee takes advantage of the campaign period to prepare the printed ballots and wait for the Elections day. The elections are held in a big room on campus where candidates deliver speeches before the casting of votes, the candidates then cast their votes before the rest of the students. The ballots are collected after the elections for calculations, and this process is monitored and managed by supervising teachers. The results are released after the vote counts leading to the warm welcomes of the newly elected winners.

These are the issues the current manual system faces: winner announcements are delayed mostly because of the result of manual counting which also involves re-counting, mismanagement of paper ballots that leads to damage and loss of the paper ballots, high resource costs which is caused by the need of paper products like posters, application forms, and ballots, lastly is a problem that would breach the confidentiality of the elections by letting outsiders participate while thinking all voters are students from the campus.

Student Council Voting System for St John the Baptist University Blantyre is the proposed system that will be accessed on multiple devices since the system will be a web based system to allow a responsive design that will help students to cast votes without the worry of the viewport. The proposed system will ensure will ensure that all voters are authenticated and registered using their student IDs, the system will automatically tabulate all vote counts, the system will help the university eliminate the damage and loss of ballots problem while reducing the costs of resources.

# **AIMS AND OBJECTIVES**

## **AIM**

The student’s council voting system for St John the Baptist University aim is to allow students with active student IDs to create voting accounts which will help the student in participating in running elections. The system will generate reports of winners in form of podiums while also generating bar graphs for all numeric data; for example the number that registered for the elections.

## **Objectives**

## **Investigate and Analyze the Current System**

* Conducting literature review on the voting system.
* Analyzing and conducting research on similar voting systems.
* Reviewing Technologies needed for coming up with the voting system.

## **Design the System**

* Coming up with the structural model of the system.
* Highlighting how different parts of the system interact with the use of behavioral models.
* Identifying the entities needed in the database schema.

## **Develop the System**

* Creating the database together with the required tables.
* Coming up with frontend of the voting system.
* Developing the system by integrating the database and then implementing the logic through the backend.

## **Test and Implement the System**

* Creating test cases to see if the system functions as expected.
* Conducting black box tests to ensure the system produces the expected outputs.
* Testing all functionalities to check if they are performing as required by the system requirements.

## **Documentation**

* Capturing system processes through screenshots.
* Coming up with a user guide.
* Producing a final report.

# **Analysis**

Analysis helps in meeting the needs of system by gathering and understanding information corresponding to the system as stated by (Namacha, 2024). Analysis allows developers in assessing technical feasibilities of all requirements helping the developers in reviewing issues that are not apparent at that time but later down the development of the system as highlighted by (Brandoburová, 2023). Analysis helps in setting up the logical reasoning of the developer(s) analyzing a particular system which allows the developer(s) to find patterns and trends in the gathered data as noted by (Karl, 2024).

## **Requirements**

## **Functional Requirements**

Functional requirements are expected features that the system will perform as expected according to (Rebrova, 2024). Functional requirements helps people work towards one goal hence making everyone present to be on the same page as stated by (Watson, 2024). The system will use the MOSCOW approach for effective implementation.

### **User Registration**

The system **must** allow users to successfully create accounts.

### **Timely Election Reminders**

The system **should** remind all voters on elections updates via email and in system notifications.

### **Candidate Manifesto Chat**

The system **must** provide a chat-like platform for candidates to share manifestos and for voters to view all manifestos.

### **Election Authentication**

The system **should** prevent duplicate votes, late votes while maintaining the secrecy of all voters.

### **Election Timer**

The system **must** display a timer counting down to a specified time on the vote and manifesto pages.

### **Calculating Final Tabulations**

The system **should** count total vote counts on each candidate, the number of voters that registered, and the registered voters that actually casted votes.

### **Report Generation**

The system **should** create winner podiums for all running office positions and bar graphs for all the numeric data.

## **Non-Functional Requirements**

Non-functional requirements are the conditions that help the features that fit the system requirements to perform s expected perfectly as highlighted by (Krüger, 2024).

### **Performance**

The system **should** handle concurrency issues, such as allowing multiple users to register while others are voting without breaking or lagging.

### **Security**

The system **must** prevent SQL injection attacks by using parameterized SQL queries and prepared SQL statements to protect the system’s database.

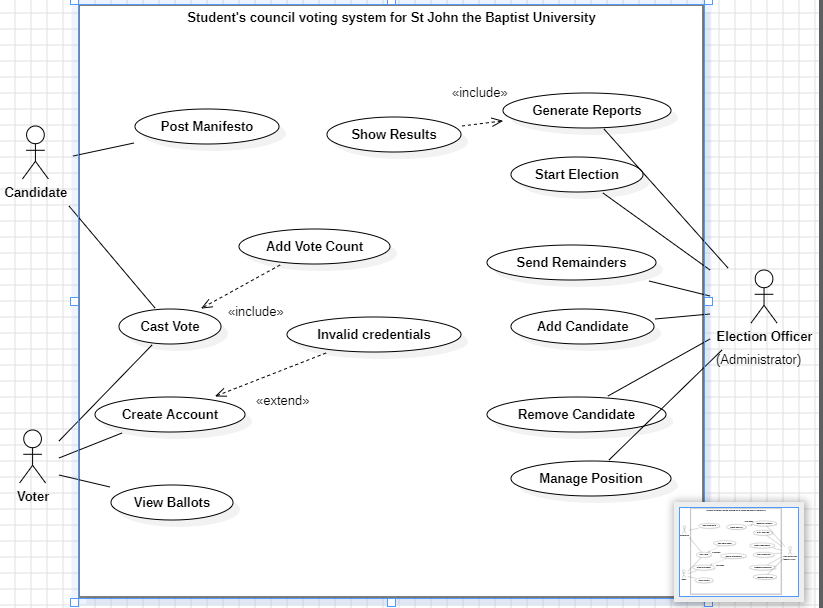
### **Usability**

The system **should** have an easy to use user-interface by making the manifesto and vote page simple and straightforward while maintaining minimal steps and words.

### **Maintainability**

The system **should** be easily updated by making use of C# classes that easily help in finding bugs and issues in code snippets.

## **Use Case Diagram**



## **System Architecture**

JAVASCRIPT

HTML & CSS

**PRESENTATION TIER**

C#

ASP.NET

**LOGIC TIER**

SQL

DATABASE (SQLite)

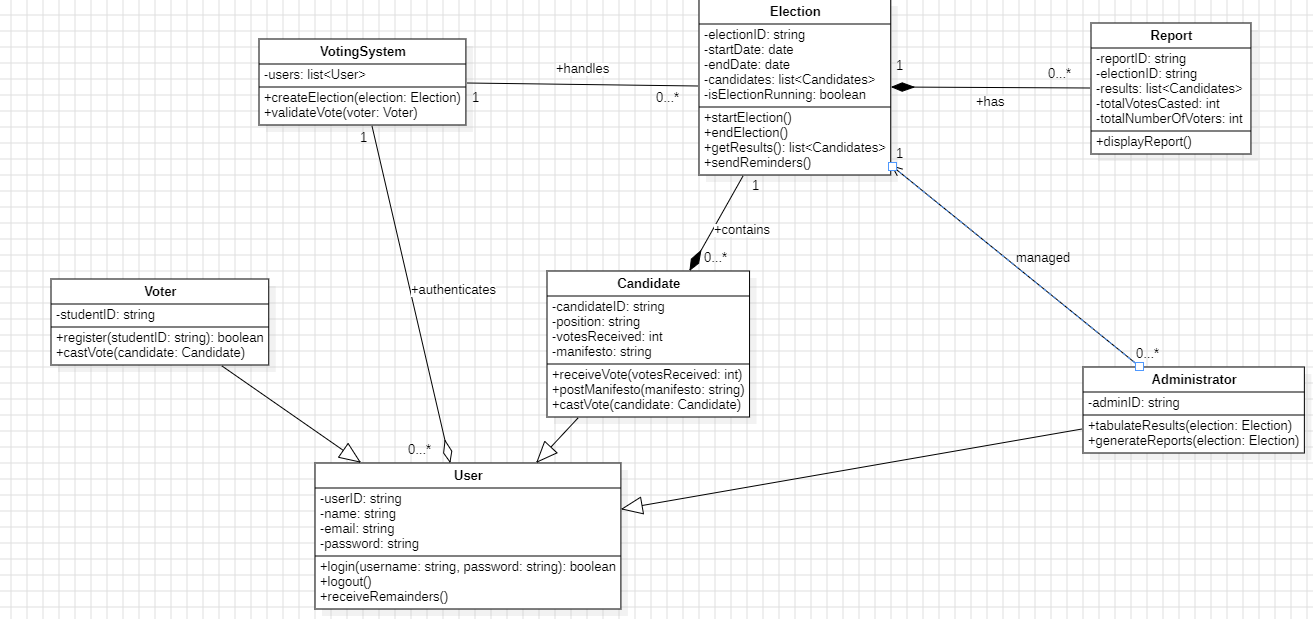
**DATA TIER**

# **Design**

This section will highlight the structural model of the system together with the behavioral models of the system processes illustrated in the use case diagram.

## **Structural Model**

A structural model helps better understand a system since the aim of such a model is to show components namely classes and modules that make up the system as stated by (Pandey, 2023).

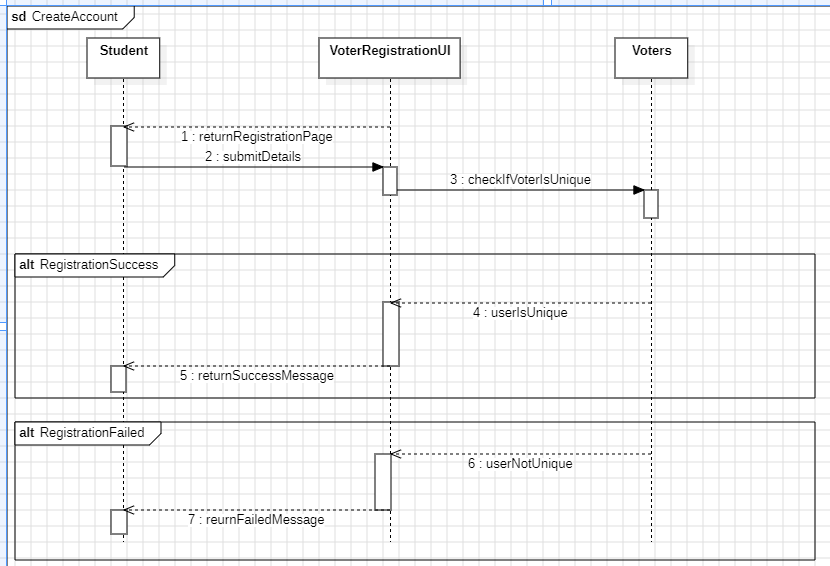


The above diagram shows the class diagram for the Student’s Council voting system for St John the Baptist University. The Voter, Candidate, and Administrator classes are all child classes of the abstract class; a class that acts as template and will never be instantiated as highlighted by (Gillis, 2025). The class User will help the 3 different roles to perform different functionalities within the system, where voters can cast votes after creating accounts, candidates can both post manifesto’s and cast votes, and the administrator who is the election officer can generate election reports together with managing the elections. The VotingSystem class authenticates all users and also handles the election processes.

## **Behavioral Models**

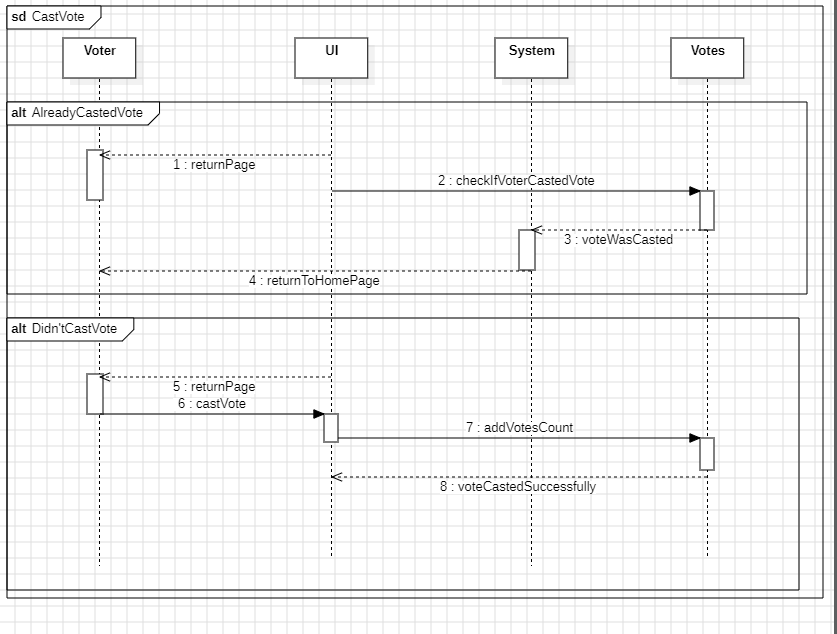
Behavioral models visualize how different parts of a system communicate by focusing on functionalities rather than models of the system as highlighted by (Dam, 2022).

### **Create Account**



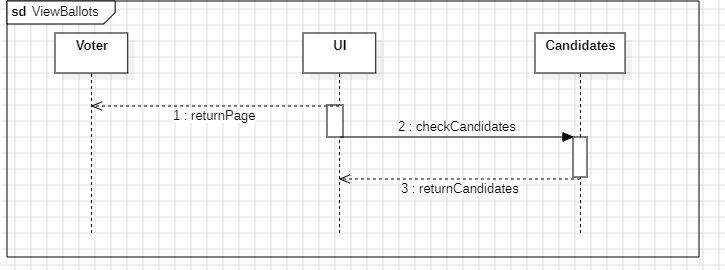
The voter (student) will be introduced to the create account page where they will provide details needed by the form before submitting, the system will check if the user details provided are unique and if the details are indeed unique the account will be created and the voter will be eligible to vote and if the user is not unique the system will be prevent the voter from successfully creating an account.

### **Cast Vote**



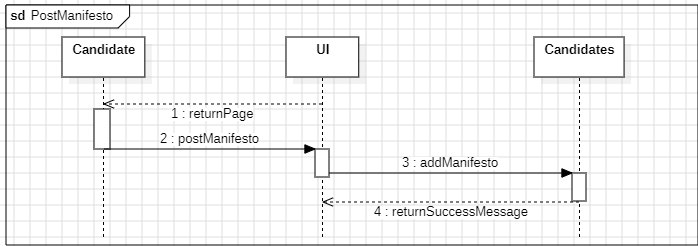
Before showing the page the system will check if the voter already voted by checking votes linked to that particular voter and if the system returns that the voter did vote the voter will be redirected to the home page of the system and if the voter didn’t cast any vote the system will allow the interested voter to view all the ballots and cast votes.

### **View Ballots**



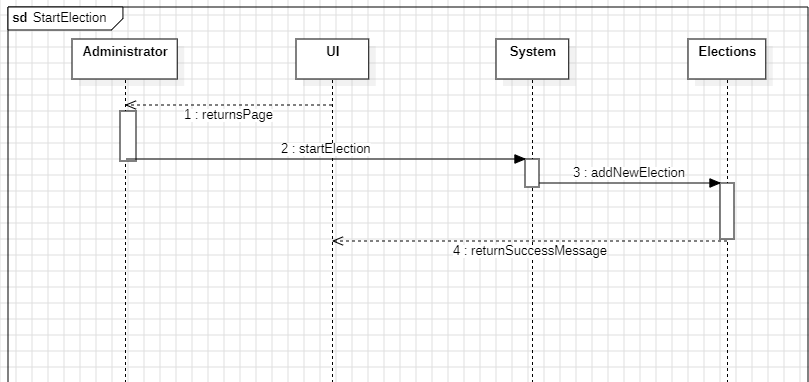
This use case allows voters to view ballots. A ballot is a paper or a digital page that has names of all candidates and their positions in the running election, according to (gravesham borough council, 2025). Viewing ballots will help voters in their casting votes.

### **Post Manifesto**



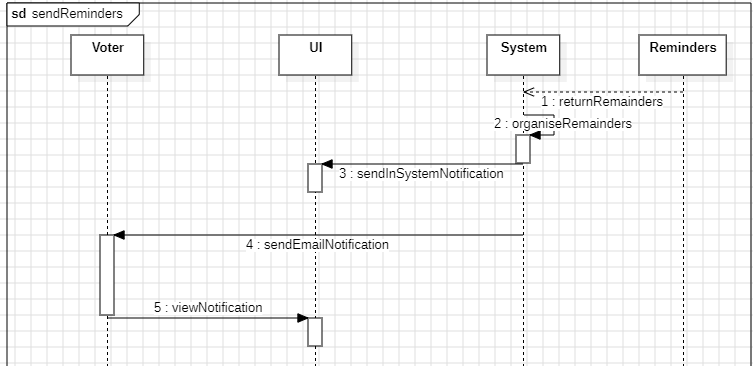
Candidates will be able to post manifesto’s which are defined as policies delivered by candidates after an election is won (Ben Paxton, 2024). This will help candidates catch the eyes of voters and help in determining the will and wish of voters as argued by (Ahmed, 2013).

### **Start Election**



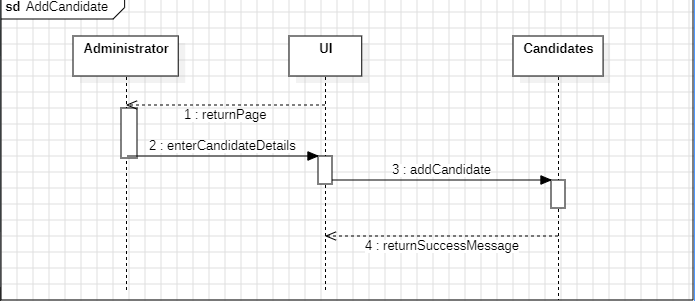
The election will be handled by the administrator, and one way of managing it is by activating the start election use case which sets the election running and open to the students who are our voters.

### **Send Reminders**



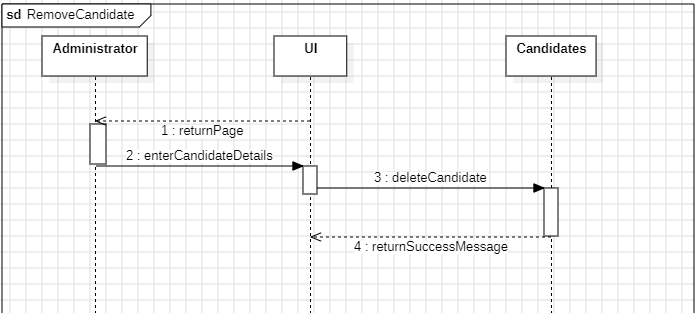
This use case will be automatically run by the system based off of the timer that will be within the system causing voters to receive email notifications and in-system notifications; which are short messages sent to users within the application grabbing the attention of the target audience (AppsFlyer, 2025).

### **Add Candidate**



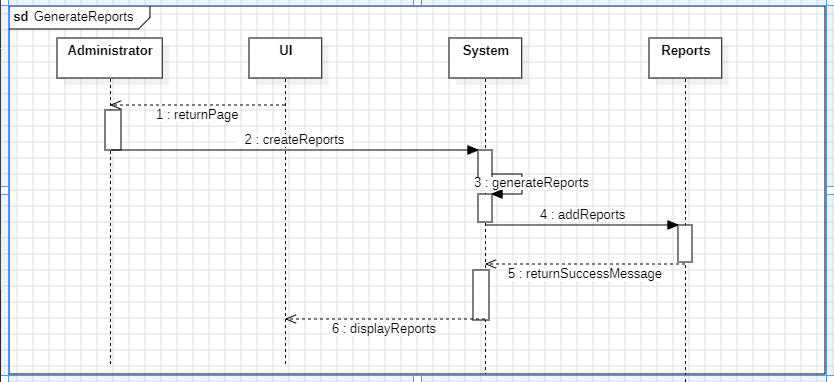
Administrators add all the candidates that will be involved in the election by adding all their details that will be processed and then later saved to the system’s database.

### **Remove Candidate**



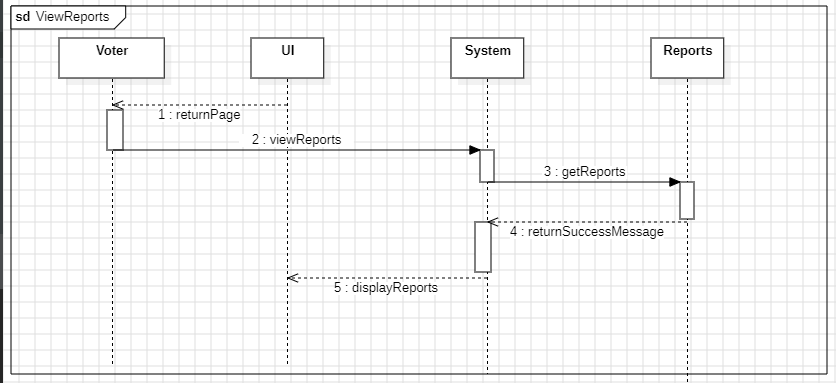
This use case will also be managed by the administrator allowing the administrator to remove the candidate from both the election and the system’s database.

### **Generate Reports**



Election reports involves displaying the number of people that voted, number of ballots casted in the election, and the candidates that were elected (ACE, 2025). The reports will be generated by the administrators and then later displayed to all the voters.

### **View Reports**



This action will be done by voters to check the winners of the election and other insights of the student’s council elections.

# **Conclusion**

The Student Council Voting System is a great milestone that eliminates the manual process by making the voting experience automatic while providing a user-friendly interface, and ensuring everyone can use it by implementing minimal and easy to understand text, the problems of the current system have been solved by this student’s council voting system. The constant testing and code improvements also displays that the system is reliable and easy to maintain. This system makes voting easier, transparent, secure, and efficient, making it a useful tool for any school’s student council to say the least.

|  |  |
| --- | --- |
| System Requirements | Outcome |
| User Registration | **Success** |
| Timely Election Reminders | **Success** |
| Candidate Manifesto Chat | **Success** |
| Election Authentication | **Success** |
| Election Timer | **Success** |
| Calculating Final Tabulations | **Success** |
| Report Generation | **Success** |

## **Future Development**

The following are future developments that could improve the student’s council voting system:

1. **Calendar integration:** letting users add highlighted elections day to their personal gadgets directly from the voting system.
2. **Live Tabulations:** providing real-time voting progress on which candidate has the highest vote count from each category.
3. **Voice assistance:** implementing a voice command feature that helps users with motor disabilities.
4. **Integration with Student Information System:** integrating with this system will allow the voting system to make predictions on who will likely be the best fits in the upcoming elections based on school conduct and grades, increasing student engagement.
5. **Feedback feature:** allowing voters to provide reviews and feedback on the elections and how they think the system could be better.

# **Evaluation of Work**

## **What Went Well?**

The system efficiently managed to house the mentioned requirements namely: user registration, timely election reminders, candidate manifesto chat, election authentication, final tabulation calculations, and report generation. The integration of the SQLite database to the voting system was successful allowing secure and better management of data to and from the database. The system also achieved scalability with the use of C# classes that are provided by ASP.NET.

## **What Did Not?**

In terms of issues, combining JavaScript and C# was difficult since these programming have different aims making it difficult to use them together. The solution was to use one **Cshtml** file that allowed to combine both programming languages causing the file to have a lot of unmanageable code which is bad practice for any developer.

## **What Was Learned?**

A project needs a lot of planning and analysis because implementation is just acting upon already analyzed information. Coding is also not about writing the best manageable code snippets but rather dissolving problems to smaller solvable chunks that lead to having the best manageable and readable code snippets. Project reports help in making changes in the future when better features are integrated in the future and reports also allow developers to recall on how they solved a particular problem.

## **What Will Be Done Differently?**

The use of design patterns will strategically help to better improve the quality of code snippets as highlighted by (Cogtix, 2023). Time will not be allocated evenly since some functionalities require less time compared to others. Future projects will be divided into more but smaller achievable problem chunks allowing less stress and more efficiency.

# **References**

ACE, 2025. *aceProject.* [Online]   
Available at: https://aceproject.org/main/english/em/eme11.htm  
[Accessed 14 February 2025].

Ahmed, R. I., 2013. *ResearchGate.* [Online]   
Available at: https://www.researchgate.net/publication/326635518\_Manifesto's\_importance\_in\_general\_elections\_2013  
[Accessed 14 February 2025].

AppsFlyer, 2025. *AppsFlyer.* [Online]   
Available at: https://www.appsflyer.com/glossary/in-app-notifications/  
[Accessed 14 February 2025].

Ben Paxton, C. H., 2024. *Institute of Government.* [Online]   
Available at: https://www.instituteforgovernment.org.uk/explainer/manifesto  
[Accessed 14 February 2025].

Brandoburová, M., 2023. *Goodrequest.* [Online]   
Available at: https://www.goodrequest.com/blog/how-tech-specification-helps-in-softver-development-and-why-it-is-important-for-clients  
[Accessed 24 April 2025].

Cogtix, 2023. *LinkedIn.* [Online]   
Available at: https://www.linkedin.com/pulse/importance-design-patterns-software-development-cogtix  
[Accessed 25 April 2025].

Dam, S., 2022. *SPEC innovations.* [Online]   
Available at: https://specinnovations.com/blog/what-is-a-behavior-model  
[Accessed 24 April 2025].

Gillis, A. S., 2025. *TheServerSide.* [Online]   
Available at: https://www.theserverside.com/definition/abstract-class  
[Accessed 14 February 2025].

gravesham borough council, 2025. *gravesham.* [Online]   
Available at: https://www.gravesham.gov.uk/elections-voting/vote/2  
[Accessed 14 February 2025].

Karl, T., 2024. *newhorizons.* [Online]   
Available at: https://www.newhorizons.com/resources/blog/data-analysis-for-research  
[Accessed 24 April 2025].

Krüger, G., 2024. *perforce.* [Online]   
Available at: https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples  
[Accessed 24 April 2025].

Namacha, N., 2024. *Medium.* [Online]   
Available at: https://medium.com/@nnamacha/a-beginners-guide-to-software-engineering-systems-analysis-and-design-228b22cbc755#:~:text=What%20is%20Systems%20Analysis%20and,How%20will%20the%20software%20work?%E2%80%9D  
[Accessed 24 April 2025].

Pandey, P., 2023. *Medium.* [Online]   
Available at: https://medium.com/@publicapplicationcenter/understanding-structural-models-in-software-architecture-5d9318497865  
[Accessed 24 April 2025].

Rebrova, A., 2024. *maddevs.* [Online]   
Available at: https://maddevs.io/blog/functional-vs-non-functional-requirements/  
[Accessed 24 April 2025].

Watson, M., 2024. *fullscale.* [Online]   
Available at: https://fullscale.io/blog/functional-vs-non-functional-requirements/  
[Accessed 24 April 2025].

# **APPENDIX**

## **Requirements Collection Catalogue**

### MoSCoW Prioritization

#### Must-Have

* User Registration.
* Election Authentication.
* Calculating Final Tabulations.

#### Should-Have

* Timely Election Reminder.
* Candidate Manifesto Chat.
* Election Timer.
* Report Generation.

#### Could-Have

* Real-Time Vote Count Updates.
* Voice Command Feature.

#### Won’t-Have (this time)

* Potential Candidate Suggestion.
* Social Media Account Logins.

### Sample Questionnaire

1. How do you inform students about upcoming student council elections?

- [ ] Posters and Flyers

- [ ] School Announcements

- [ ] Social Media

2. How do students register or sign up to vote in the current system?

- [ ] In-person Registration

- [ ] Submitting a Form

- [ ] No Registration Required

3. Where and how do students cast their votes?

- [ ] Dedicated External Voting area

- [ ] Classrooms

4. How are candidates for the student council nominated?

- [ ] Self-Nomination

- [ ] Teacher Nomination

- [ ] Peer Nomination

5. What is the process for counting and verifying votes?

- [ ] Manual Count by Appointed school officials

- [ ] Volunteering students

- [ ] Automated Counting

6. How are the election results communicated to the students?

- [ ] School Assembly

- [ ] Posters and Flyers

- [ ] Emails

- [ ] Social Media

7. How satisfied are students with the current manual voting process?

- [ ] Very Satisfied

- [ ] Satisfied

- [ ] Neutral

- [ ] Unsatisfied

- [ ] Very Unsatisfied

## **Use Case Descriptions**

1. **Create Account**: This use case action allows all users who are the university students to create accounts that will require them to input personal details about them.
2. **Cast Vote**: Allows eligible voters for cast votes on their candidate of choice under a chosen position.
3. **View Ballots**: A use case designed to allow voters to view candidates of different positions before they cast their votes.
4. **Post Manifesto**: An action performed only by candidates to catch the eye of voters before the election’s day.
5. **Add Vote Count**: An action that is performed automatically when a vote is casted to add to the vote counts.
6. **Invalid Credentials**: An action that is triggered when a user hasn’t met the account creation requirements.
7. **Generate Reports**: A use case performed only by the administrator to produce reports on the election process.
8. **Show Results**: Automatically shows results when the generate reports use case is performed.
9. **Start Election:** This action is started and maintained by the election administrator till the election is closed.
10. **Send Remainders:**  Remainders that in the form of emails are sent to all users after the cause of this use case.
11. **Add Candidate:** This use case involves adding candidates in the election database and this action is performed by the election officer.
12. **Remove Candidate:** This involves the election officer removing candidates that have withdrawn from the election.
13. **Manage Position:** This use case enables the election officer to manipulate all positions which might involve removing or adding election positions.

## **Detailed Class Descriptions**

## **User class**

This parent class encapsulates the user information and operations while being inherited by the **Voter, Candidate,** and **Administrator** classes.

## **Voter class**

This class handles registration of voters and allows the voters to cast votes.

## **Candidate class**

The Candidate class receives votes while allowing the posting of manifestos and casting of votes.

## **Administrator class**

This Administrator class handles the tabulating of election results and the generation of reports.

## **VotingSystem class**

This class manages the voting process and this same class is involved in managing users and starting of elections.

## **Election class**

The election class manages candidates while allowing the ending and restarting of elections.

## **Report class**

This class generates the reports of the election process acting upon numeric data.

## **Test Scripts**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case** | **Functionality** | **Scenario** | **Data Value** | **Expected Results** | **Actual Results** |
| 1 | User Registration | Users registers with valid credentials | Student Id: L5BIM/60  Email: [tupomlanjira@gmail.com](mailto:tupomlanjira@gmail.com)  Username: TTSOT2  Password: Tupo@2025P | User registered successfully. | TTSOT2 registered successfully. |
| User registers with already used Student Id | Student Id: L5BIM/60  Email: [joshuaadams@gmail.com](mailto:joshuaadams@gmail.com)  Username: JoshyA20  Password: Josh@5220 | User is not registered and modal window is triggered with informative text. | Student Id or Username or Email already registered with!! |
| 2 | Timely Election Reminders | Reminders are sent to all registered users | Reminder: Don’t forget to cast your votes tomorrow!! | Reminder sent and received Successfully. | Voter receives an Email and in system notification. |
| Unregistered students | Reminder: Don’t forget to cast your votes tomorrow!! | Reminder is not received. | Unregistered Students don’t receive any Emails. |
| 3 | Candidate Manifesto Chat | Candidates post and manage all their manifestos | Posted By: TTSOT2  Manifesto: quizzes at least twice a week | Manifesto will be posted and linked to the candidate and shown under the candidate’s real name using their student id. | All TTSOT2 manifestos are linked to his name using his student Id and highlighted in a unique sender color. |
| Voters only view the manifestos | Posted By: Tupo272  Manifesto: more filed trips  Posted By: Francis22  Manifesto: school entertainment shows every Thursday and Saturday. | Manifestos will be viewed by all voters denying them of editing the manifestos. | All Manifestos are not editable and have one uniform color. |
| 4 | Election Authentication | Voters can cast a vote once | Voter: TTSOT2 | Voter will successfully cast a vote. | You’ve successfully casted your vote! |
|  |  | Voters tries to cast twice | Voter: TTSOT2 | Voter will not be allowed to cast twice. | You already Voted! |
| 5 | Election Timer | Pages are hidden | Timer < 40 seconds | Vote Page contents will be hidden. | Ballots will be not be viewed by voters and a decreasing timer will be shown. |
| Pages are shown | Timer > 40 seconds | Vote page contents will be shown. | Ballots will now be viewed by the voters and the timer will now be hidden. |
| 6 | Calculating Final Tabulations | Number of votes, Number of empty ballots, Number of Voters > 0 | Number of Voters: 12  Number of Votes: 8  Number of empty ballots: 3 | System makes calculations | Display of calculated totals for each numeric filed. |
| Number of votes, Number of empty ballots, Number of Voters < 0 | Number of Voters: 0  Number of Votes: 0  Number of empty ballots: 0 | No calculations are made | No data currently!! |
| 7 | Report Generation | Admin reports | Number of Voters: 12  Number of Votes: 8  Number of empty ballots: 3 | Bar graphs are generated for each calculation | A display of bar graphs. |
| Voter reports | Position: President  Candidate: Rachel  Vote Count: 6 | A podium is generated and shows top 3 candidates. | Voters view podiums for all fields with the candidates that have high vote counts being on top. |