

SENG 3210– Applied Software Engineering

Online Voting System

Trevor Dayton (TRU ID)

Brendan Mierau (T00670407)

Date submitted

Table of Contents

[1 Introduction 5](#_Toc62394656)

[2 Design Problem 6](#_Toc62394657)

[2.1 Problem Definition 6](#_Toc62394658)

[2.2 Design Requirements 6](#_Toc62394659)

[2.2.1 Functions 6](#_Toc62394660)

[2.2.2 Non-functional requirements and constraints 6](#_Toc62394661)

[3 Solution 7](#_Toc62394662)

[3.1 Solution 1 7](#_Toc62394663)

[3.2 Solution 2 7](#_Toc62394664)

[3.3 Final Solution 7](#_Toc62394665)

[3.3.1 Features and the software architecture 7](#_Toc62394666)

[3.3.2 The system interfaces 7](#_Toc62394667)

[3.3.3 The user interface design 7](#_Toc62394668)

[3.3.4 The requirements tractability matrix 8](#_Toc62394669)

[3.3.5 Environmental, Societal, Safety, and Economic Considerations 8](#_Toc62394670)

[3.3.6 Limitations 8](#_Toc62394671)

[4 Teamwork 8](#_Toc62394672)

[4.1 Meeting 1 8](#_Toc62394673)

[4.2 Meeting 2 8](#_Toc62394674)

[4.3 Meeting 3 9](#_Toc62394675)

[4.4 Meeting 4 9](#_Toc62394676)

[5 Conclusion and Future Work 9](#_Toc62394677)

[6 References 10](#_Toc62394678)

[Appendix 11](#_Toc62394679)

* The table of contents should be automatically generated by selecting “References/ Table of Contents”. Remember that the table of contents should not have an entry of the “Table of Contents” itself.
* Proofread the text for typing and grammar mistakes.
* Follow the IEEE Bibliography style for the references by selecting "References/ Citations & Bibliography/ Style".

List of Figures

List of Tables

# Introduction

* Give a brief description of the design and summarize the relevant background information related to the topic. Give a rationale about what is needed and why.
* Give the reader an overview of what is in the next sections.
* Do not put any detailed results of your work here.

An Online Voting System is a system for managers to create topics to be stored in a database for users to vote on. When the users open the app they will have a different user experience where they can see a list of available topics to vote on. Once they select a topic they can pick an option to vote on and their feedback is sent back to the database and stored. The managers can then look at this data to get feedback from users on their topics to make more informed decisions with their users.

THIS PART NEEDS TO BE AN OVERVIEW

# Design Problem

## Problem Definition

There can be an exceedingly large amount of misunderstanding between management and users the same as between a government and its people. To gather feedback for what the users want, voting is one of the strongest strategies but the cost and time of the activity can make this strategy can be prohibitive.

Our project aims to make the process far more accessible and cost effective for business’s to implement using an android application. This allows everyone easy access to the system and eliminates the cost of posing new topics to the users. This allows a greater level of understanding about how users feel on certain topics and allows management to make more informed decisions.

## Design Requirements

### Functions

* An android application available to all desired users
* Users can cast votes on any topic that is created
* Managers can see the real-time statistics of every topics current polling
* Managers can create topics for users to vote on

### Objectives

* Allow for more understanding between users and managment
* Allow users to have a voice on topics within an organization
* Remove the high cost of in person voting systems
* Implement using firebase integration
* Have one app that operates different depending on who is using it (Voter or Manager)
* Provide objectives of the design project. Remember that the objectives are specified as adjectives.

### Non-functional requirements and constraints

* Users cannot vote multiple times on the same topic
* The app must operate differently depending on if a user logs in or a manager
* It must be a cheaper solution than in person voting
* This solution must be created in a very limited timeframe of only several weeks
* Data must be reliably stored in a firebase cloud database
* Environmentally friendly
* Provide non-functional requirements and constraints here.

# Solution

This section will provide an account of some solutions your team brainstormed to implement the project. Some solutions might not have all the desired features; some might not satisfy the constraints or both. These solutions come up in your mind while you brainstorm ways of implementing all the features while meeting the constraints. Towards this end, you select a solution that you think has all the features and satisfies all the constraints. Please remember that an engineering design is iterative!

## Solution 1

The first solution that was considered was integrating the OVS with a MYSQL database created in-house. This solution would allow more power on how to structure the database. This solution is more costly in the upfront capital into getting databases setup. This solution also has the advantage of being more environmentally friendly than in person voting as a small amount of servers are needed for data collection.

This solution does have some disadvantages. The team is responsible for the initial setup and maintenance of the database which increases production time and complexity. This also increases the workload as the team will have to setup and manage the hardware.

Write a brief description of your first solution and provide the reasons for not selecting this one.

You can use the component diagram, sequence diagram, and class diagram.

## Solution 2

This solution was our first working iteration of the OVS. Before we learned to structure our project we implemented all of the controller logic inside of the models. This allowed for the relevant code of each model to be integrated together and allowed for an easier initial implementation.

The problem with this solution is that it isn’t very modifiable, scalable or reliable. Since the models structure is tied with its controlling logic, changing anything to do with either the structure or the logic required a refactoring of both aspects. This isn’t ideal for future iterations and improvements as it more than doubles the time modifying or adding features. Creating an interconnected system like this with a large RFC adds increasing complexity due to the model logic and controller logic interconnected calling eachother across the same class for multiple instances.

This solution is an improved solution but might not be the final solution that you select. Give a brief description of this solution here.

You can use the component diagram, sequence diagram, and class diagram.

## Final Solution

The final solution is an iteration on solution 2. After consulting with the client the team decided to refactor the application so that the model and controller were separate. This architecture allows for greater modularization, modification and flexibility for implementing new features.

### Features and the software architecture

#### Single application for both user and manager.

Instead of having separate applications for the two different user types we are going to have one application that performs differently depending on the user who operates it. This is accomplished by the login activity. In the login controller when the user logs in, if they have a valid login it will extract their user object and inside the user object it has an attribute of the type of user they are. The controller uses this information to pull up the appropriate model for the user type, the voters get a voter dashboard while the managers get a manager dashboard.

#### Login Integrated with Firebase

The user models are saved within the firebase database. When a user logs in the controller will use the inputted login information to firebase to search for a user with the correct login information. Firebase then retrieves the user model and uses the users model information to operate within the application.

#### Dynamic Voting List

The list of topics in both the voter dashboard and the manager dashboard are generated dynamically based on how many topics there are to vote on within the database. This scales up and down as needed making the topics incredibly scalable. What happens in the controller pulls all of the available topics and then inputs each topic title into a button. Then the collection of buttons generated are put into a list for the user to pick to either vote on a topic or simply view its voting statistics. TREVOR PLZ EXPLAIN HOW TO REUSE REDUCE AND RECYCLE

#### Voting on Topics

The voters can select a topic from the voter dashboard to vote on. When they click on the topic they will be sent to a generated intent to display that topic and its voting options. This topic model is pulled from the database with the controller to then generate the view for the topic. The voter then selects a voting option from pulled topic and presses vote. This submits the vote and their user ID back into the database. The user ID is checked to disallow voting multiple times on the same topic. If the user ID hasn’t voted on the topic then the vote count in the database is incremented.

#### Viewing the statistics on available topics

The managers dashboard is how managers can view the statistics on all of the available topics. This is done by the same process as in the dynamic voting list up above except it pulls more information from the topic model such as how many votes and for what option are displayed beside each topic.

#### Creating Topics

The software architecture of our solution is based on MVC.

Discuss all the features of your final solution. Describe the functionalities of the top-level components and how they will be used for enabling those features. The product features may be tabulated (with a title) for improved comprehension. Use component diagrams to model the internal structures (i.e., sub-components or second-level components) of two major components. Describe the functionalities of the sub-components and the interactions (e.g., the interfaces) between the sub-components. Explain the interfaces between the top-level architecture and the internal structures (i.e., explaining how the internal structures interact with other top-level components).

### The system interfaces

Describe the temporal events (i.e., the time-triggered events) and the signal events (i.e., events received from external components) of the Online Voting System. Describe the expected response of the system to each event.

### The user interface design

Design the user interface components. Describe the user interface components, the possible business events, and the responses to the triggered events.

### The requirements traceability matrix

List the system’s requirements and map the requirements to the corresponding design component, code component (e.g., java class file or XML configuration file), and the required testing scenario.

### Environmental, Societal, Safety, and Economic Considerations

Explain how your design project considered environmental, societal, and economic considerations. It may include how your implementation has positive contributions to the environment and society. What type of financial decisions did you make? How did you make sure that the implementation is safe to use?

#### Environmental considerations

Explain how your design project considered environmental considerations.

#### Societal considerations

Explain how your design project considered societal considerations.

#### Safety considerations

Explain how your design project considered safety considerations.

#### Economic considerations

Explain how your design project considered economic considerations.

### Limitations

Every product has some limitations, so is the case with your design solution to the Online Voting System (OVS). Highlight some of the limitations of your implementation here.

# Teamwork

Since this is a group project, you must have a fair distribution of tasks among yourselves. To this end, you must hold meetings to discuss the distribution of tasks and keep track of the project progress.

## Meeting 1

Time: January 21, 2021, 10:00 am to 11:00 am

Agenda: Distribution of Project Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Previous Task | Completion State | Next Task |
| Team member 1 | N/A | N/A | Task 1 |
| Team member 2 | N/A | N/A | Task 2 |
| Team member 3 | N/A | N/A | Task 3 |
| Team member 4 | N/A | N/A | Task 4 |

## Meeting 2

Time: February 4, 2021, 10:00 am to 11:00 am

Agenda: Review of Individual Progress

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Previous Task | Completion State | Next Task |
| Team member 1 | Task 1 | 80% | Task 1, Task 5 |
| Team member 2 | Task 2 | 50% | Task 2 |
| Team member 3 | Task 3 | 100% | Task 6 |
| Team member 4 | Task 4 | 75% | Task 4, Task 7 |

## Meeting 3

Provide a similar description here.

## Meeting 4

Provide a similar description here.

# Conclusion and Future Work

* Describe a summary of what you achieved. Mention all the design functions and objectives that you achieved while satisfying the constraints?
* While keeping the limitations of the Online Voting System’s design, provide recommendations for future design improvements.

# References

* Use the IEEE reference style.
* Do not put any reference if it is not cited in the text.

# Appendix

If you want to provide additional information, use this appendix.