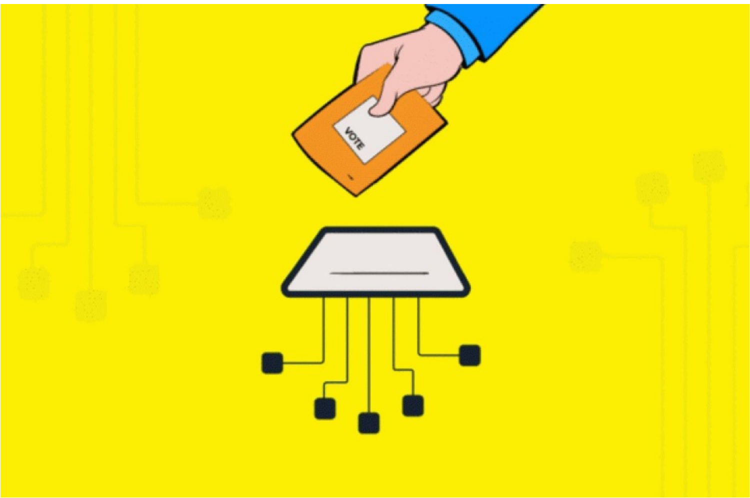
Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4316: Senior Design I
Summer 2022



The ChainVoters
Blockchain Voting

Ahmed Harun
Abdulla Sakallah
Seth Rodgers
Kelechi Egbuta

REVISION HISTORY

Revision	Date	Author(s)	Description
0.1	08.21.2022	AS	Document Creation
0.2	08.26.2022	AS	Introduction Review
0.2	08.26.2022	SR	Diagram Design
0.2	08.26.2022	KE	Diagram Design
0.3	08.30.2022	AH	Diagram Design Review
0.3	08.30.2022	AS	Diagram Design Review
1.0	09.3.2022	SR	Sub-System Modification
1.0	09.3.2022	SR	Sub-System Modification
1.0	09.3.2022	KE	Sub-System Modification
1.0	09.3.2022	AS	Description and Sub-System Modification

CONTENTS

List of Figures

List of Tables

1 Introduction

This project is a voting application that ensures votes transparency and privacy by using the Blockchain's functionality. The project is designed to match the needs of a business or government agency that must ensure voters' anonymity and transparency to help make decisions based on poll results. There are many key requirements that must exist to maintain a strong architectural design. First, an authentication system must exist to create and separate each voter's account when registering/logging in, this is important to verify the user's identity when they vote. Furthermore, users have an option to create a poll where they have to fill out the information that describes the poll and provides constraints on time and the number of people participating in the poll. In addition, the application will give access to the user administrating the poll to be able to manage and monitor the number of current votes with an option to terminate the poll. Also, a user should be able to join, based on the link they received from the host. Each vote that is placed will be verified using Solidity, to ensure transparency and fairness within results. User's should be able to customize their poll results page by updating the poll name or deleting the poll.

2 System Overview

The voting application consists of eight main layers that communicate across the application to provide the key requirements that a user can perform. The front-end layer which can also be known as the user layer is responsible for routing the expected front-end functionalities from other layers of the applications to the user. The user should be able to register and sign in to the application from the onboarding layer which provides that functionality. It is then directed to the user's choice of option to either create or join a poll which is controlled by the poll manager layer. Then, if the user places or participates in any vote or polls their information will be verified by accessing the voting layer and then getting redirected to the blockchain for validation. After that, these polls or votes will be stored in the application database layer for future use in another layer, for instance, the poll results layer. The application offers an interactive way for users to interact with developers and their votes by providing an FAQ and a notification system, and the user will be able to exit the application with all of their data stored securely.

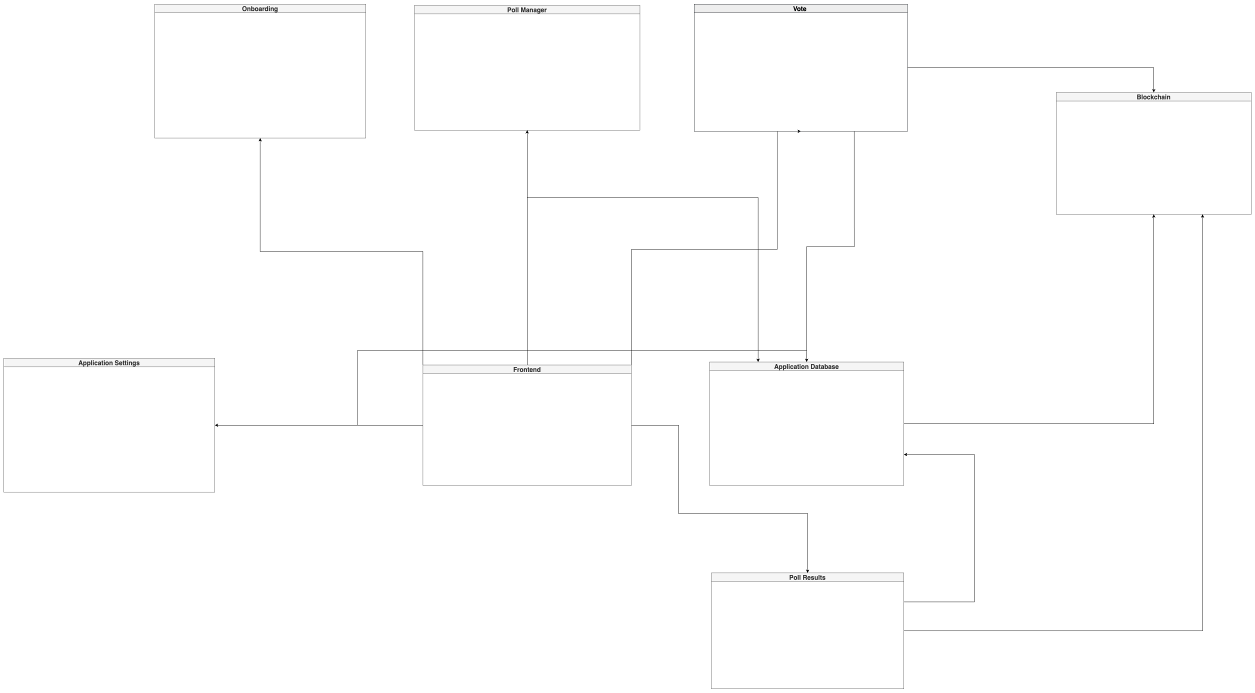


Figure 1: A simple architectural layer diagram

2.1 Front-end Layer

The front-end layer is responsible to create a connection between the user and the application by providing a friendly interface for the user to be able to perform operations in it. The first functionality from the front-end layer is that it takes the user inside the application from an onboarding screen. Then it directs the user to a poll interface to choose and decide which operations they want to do create/join and update the database by their selection. After that, the user will be directed to a voting interface to validate and update their votes concurrently. The user will have access to their poll results to review their results from previous polls. The user will also be able to access their account settings if they wanted to sign out or edit their information. The application router connects all these functionalities together by abstracted pages.

2.2 OnBoarding Layer

This layer handles the user's first interaction with an application. It does not consist of the voting functionality of the application; however, it is a major layer that needs to exist to verify the user's identity. This layer is responsible for the authentication operations handled by the database and performs any necessary recoveries for accounts.

2.3 Poll Manager Layer

This layer is responsible to initialize polls based on the user's input. The poll manager can create polls by directing the user to a page to fill out any information regarding the poll. Then, the poll manager generates an invitation to send to all targeted people that need to be assigned to vote. The user would have an option to also delete the poll. The database will be synced with each update made by any of the users.

2.4 Application Database Layer

This layer is responsible to perform CRUD operations to keep information logically synced and allow users to use it in different layers in the application. It contains a schema that gets mapped to other components to retrieve the values stored from the user when entering/registering information in any part of the application.

2.5 Vote Layer

This layer controls the status of each placed vote and ensures that each vote is being validated. After validation, the votes will be directed to check what each participant has locked their choice on. Then it will be uploaded to the Blockchain and would get stored in the database after getting processed in the blockchain.

2.6 Blockchain Layer

This layer is responsible to add the votes to the blockchain and then running through each vote and comparing the participants to see if there exists a match in the poll.

2.7 Poll Results Layer

The following layer contains abstract data and encapsulates it in a small section for the user to view on the poll results page. A query will be made to retrieve the data of the respected poll from the database so it gets matched with the respected poll. Validation will be performed by the blockchain then the results will be displayed on the screen for the user to view.

2.8 Application Settings Layer

The following layer contains information about the user and provides them with several other options to make the application more interactive and fun for the user. It does that by providing an FAQ section to allow the users to have their common questions answered and allow notifications to keep track of their most recent poll. with also allowing users to sign out from their applications and maintaining their data stored in a database.

3 Subsystem Definitions & Data Flow

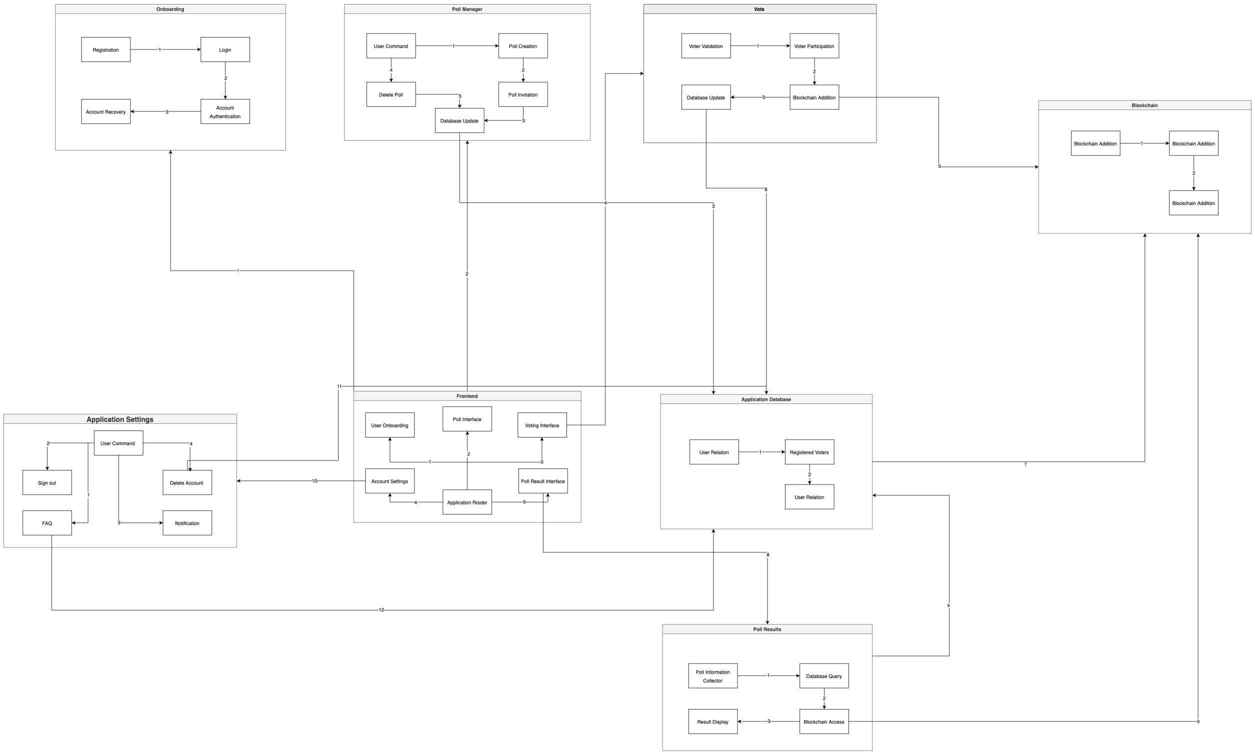


Figure 2: A simple data flow diagram

4 Block-chain Layer Subsystems

The blockchain layer is accountable for managing all of the polls and the votes that are made within them. This allows the front-end of the application to display poll and vote information to the user without having to dig through thousands of transactions on the blockchain.

4.1 Blockchain Addition

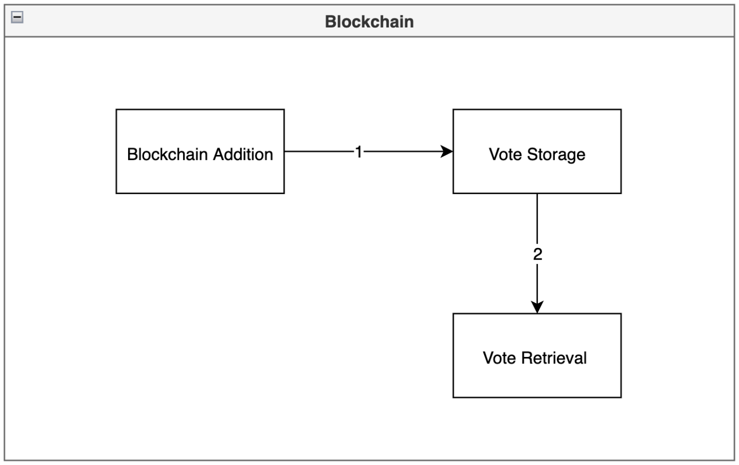


Figure 3: Blockchain Layer

4.1.1 Assumptions

- The blockchain subsystem is supplied with valid and correct data
- The Vote and Poll Manager layers have direct access to the subsystem
- The user has enough Ethereum to pay the gas fees and the transaction cost
- The gas fee is high enough that the miners are willing to perform the computation

4.1.2 Responsibilities

- Blockchain vote and poll transactions
- Security against double voting
- Ensuring that the user is authorized to vote on the poll
- Providing error code in the case of a failed transaction

4.1.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here. Create a table with an entry for each labelled interface that connects to this subsystem.

Table 2: Blockchain Addition Interfaces

ID	Description	Inputs	Outputs
#1	Voting Interface	Poll name Poll start date Poll end date User vote Voter's public key	Success code : 1 Failure code : 0
#2	Poll manager	Poll title Poll Start date Poll end date poll	Success code : 1 Failure code : 0
#3	Application Database	Poll title Public key	Voter data on the specified poll
#4	Poll Results	Poll title Public key	Voter data on the specified poll

4.2 Vote Storage

4.2.1 Assumptions

- Receives poll/vote transaction address
- The smart contract has enough memory to store thousands of transactions
- The cost to retrieve the transactions is free
- Vote storage directly is directly after a new vote or poll is made on the blockchain

4.2.2 Responsibilities

- Storing poll/vote hashes
- Retrieves poll/vote hashes
- validates that the data is unique within each poll and that the poll itself is unique (no duplicates)
- Free retrieval allowing the application to easily retrieve data from the blockchain

4.2.3 Vote Storage Interfaces

Table 3: Vote Storage interfaces

ID	Description	Inputs	Outputs
#1	Application Database	Poll title Public key	Voter data on the specified poll
#2	Poll Results	Poll title Public key	Voter data on the specified poll

4.3 Vote Retrieval

4.3.1 Assumptions

- Retrieval is of no cost to the requester
- Data is available and is retrievable
- Valid data is sent to the subsystem

4.3.2 Responsibilities

- Retrieve vote/poll from the smart contract memory
- Provide error code to user if the parameters are incorrect or not in memory
- validates that the data is unique within each poll and that the poll itself is unique (no duplicates)

4.3.3 Vote Retrieval Interfaces

Table 4: Vote Retrieval Interfaces

ID	Description	Inputs	Outputs
#1	Application Database	Poll title Public key	Voter data on the specified poll
#2	Poll Results	Poll title Public key	Voter data on the specified poll

5 Database Application Subsystems

The Database Application layer is responsible for maintaining either public or non-unique information that allows the application to run faster and also cheaper for the users. The main tradeoff of this layer is that it transforms the application to a hybrid of a centralized and decentralized application. There are some information that the developers and the application itself maintain in their database and some data that is directly stored on the blockchain. However the transparency and authenticity of the application is not lost due to the fact that only high level information related to the application's functionality for the user is stored on the applications database.

5.1 User relation

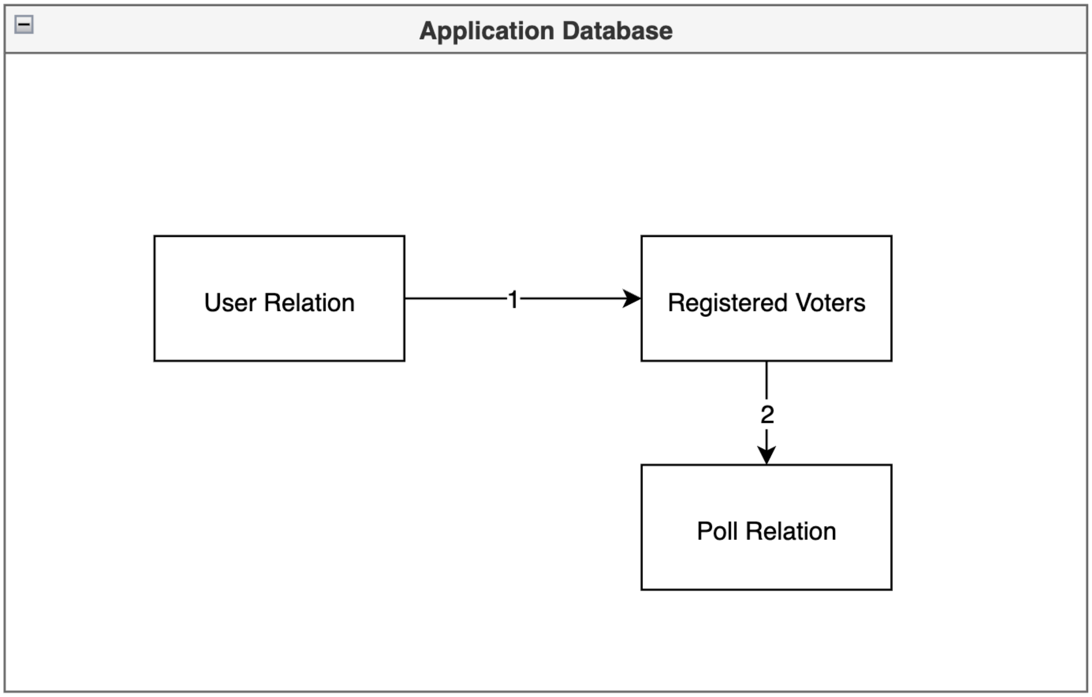


Figure 4: Application Database

5.1.1 Assumptions

- The relation has already been created and is ready to be used
- The quries to the relation utilize the correct data types and correct information otherwise no result can be returned
- The front-end can directly communicate with the relation and make queries based on the user's specifications

5.1.2 Responsibilities

- Store information related to the user and their account
- After registration add user to the registered users

- Retrieve user information during login to validate the user

5.1.3 User Relations Interfaces

Table 5: User Relations Interfaces

ID	Description	Inputs	Outputs
#1	On-boarding	Username password Email	N/A

5.2 Registered Voters Relation

5.2.1 Assumptions

- The relation is only queried after a poll administrator specifies the poll invitees
- The poll administrator will supply the necessary information to uniquely identify the individuals that should participate in the poll
- The relation is already be created and established

5.2.2 Responsibilities

- Maintain and manage the list of invitees
- Check if the user is in the relation prior to allowing the user to vote
- Add user to the relation after the administrator indicates that the person should be able to vote
- Be able to remove user from the relation if specified by the administrator

5.2.3 Registered Voters Relation Interfaces

Table 6: Registered Voters Relation Interfaces

ID	Description	Inputs	Outputs
#1	Poll manager	Poll address status public key	N/A

5.3 Poll Relation

5.3.1 Assumptions

- The relation is already be created and established
- There are existing polls in the database
- The user provides the poll information in order to retrieve the poll

5.3.2 Responsibilities

- Manages all created polls
- Contains unique information to distinguish b/w polls
- Allows users to validate poll authenticity and history
- Enhances speed and efficiency of the application by maintaining the address of each poll

5.3.3 Poll Relation Interfaces

Table 7: Poll Relation Interfaces

ID	Description	Inputs	Outputs
#1	Poll Results	Poll address	JSON containing queried data
#2	Vote	Poll address	N/A
#3	Application database	poll address	Relation containing queried data

6 Poll Manager Layer Subsystems

The Poll Manager layer is the layer of the application that manages the creation and deletion of the polls. This layer will include a user command, either to create or delete a poll. Depending on the command, the application will do the corresponding action and then update the database.

6.1 User Command

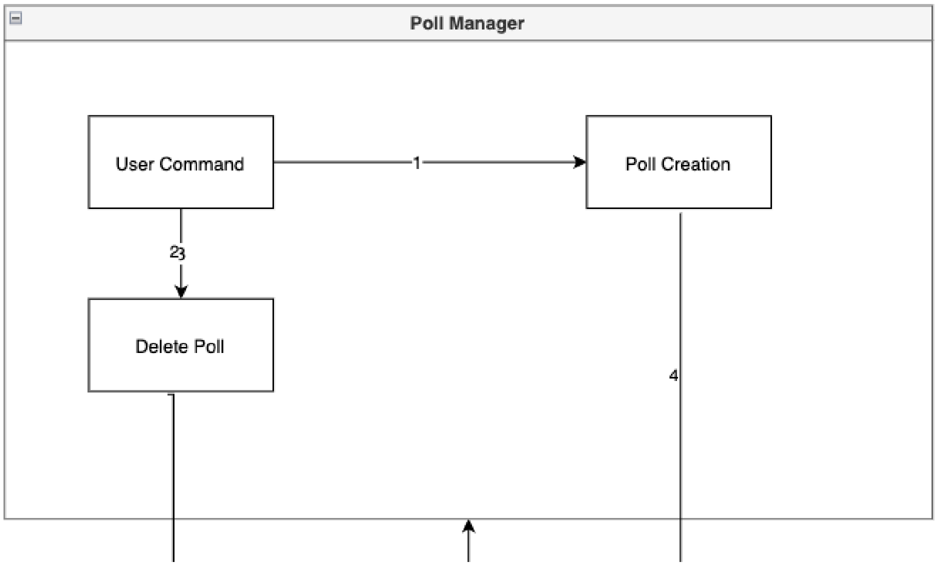


Figure 5: Poll Manager Layer

6.1.1 Assumptions

- We know the User ID making the command
- The command will be either create poll, or delete poll
- The user making the command will have permissions to do so

6.1.2 Responsibilities

- Allows users to create polls
- Allows users to delete polls
- Handle the user ID of the user making the command

Table 8: User Command Interfaces

ID	Description	Inputs	Outputs
#1	Poll Creation	Poll ID	New Poll ID
#2	Poll Deletion	Poll ID	N/A

6.2 Poll Creation

6.2.1 Assumptions

- The User ID is known
- The user making the command has the permissions to create
- The poll hasn't already been created

6.2.2 Responsibilities

- Handle all poll creation
- Update the database with the new poll
- Update the blockchain with the new poll

Table 9: Poll Creation Interfaces

ID	Description	Inputs	Outputs
#1	Database Update	Poll ID	Query w/ Poll Created
#2	Blockchain Update	Poll ID	Smart Contract Confirmation

6.3 Delete Poll

6.3.1 Assumptions

- The poll being deleted exists
- The user doing the deleting has permissions to do so

6.3.2 Responsibilities

- Manages all poll deletion
- Update the database with the given poll deleted
- Update the blockchain with the given poll terminated

Table 10: Delete Poll Interfaces

ID	Description	Inputs	Outputs
#1	Database Update	Poll ID	Query w/ poll deleted
#2	Blockchain Update	Poll ID	Smart Contract Confirmation

7 Vote Layer Subsystems

The vote layer is the section of the application that manages the poll votes. When users cast a vote, vote validation will happen to ensure the vote is valid and coming from a valid user. Once the vote is validated, the vote will be recorded on the blockchain and updated in the database.

7.1 Vote Validation

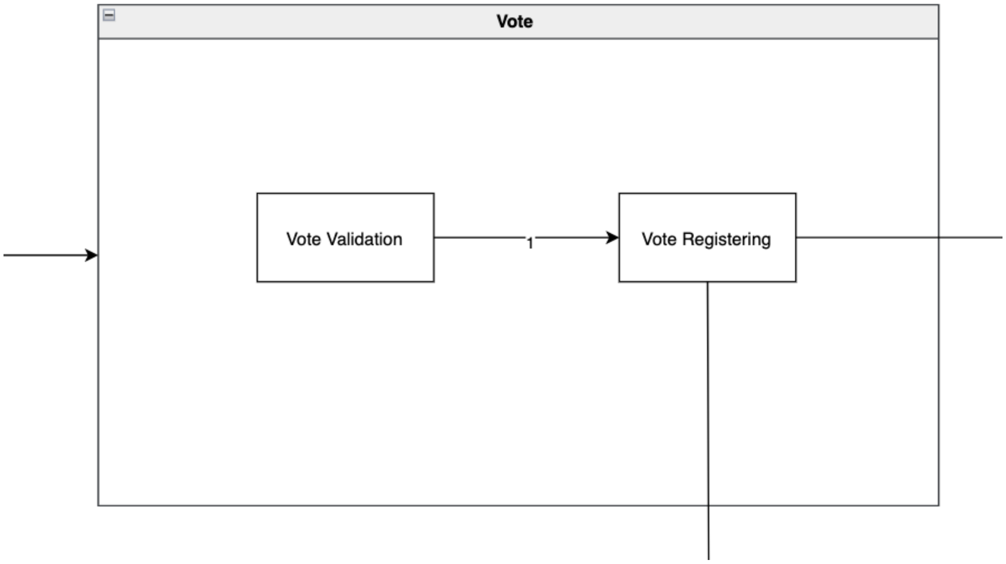


Figure 6: Vote Layer

7.1.1 Assumptions

- We know the user ID of the user placing the vote
- The vote is being placed for a poll that exists
- The user placing the vote has permission to do so

7.1.2 Responsibilities

- Handle and validate the User ID of the user placing the vote
- Handle and validate the Poll ID that the vote is being placed for
- Pass the vote on to the vote registration subsystem to be recorded

Table 11: Vote Validation Interfaces

ID	Description	Inputs	Outputs
#1	Front End	User ID Poll ID Vote	Vote Registration

7.2 Vote Registration

7.2.1 Assumptions

- We have access to the blockchain object to register the vote
- We have access to the database to register the vote in the correct poll
- The poll we are placing the vote in exists in both the blockchain and the application database

7.2.2 Responsibilities

- Update the database with the new vote
- Update the blockchain with the new vote
- Ensure both the database and blockchain were correctly updated with the new vote

Table 12: Vote Registration interfaces

ID	Description	Inputs	Outputs
#1	Database Update	Vote Poll ID	Query w/ New Vote
#2	Blockchain Update	Vote Poll ID	Smart Contract Confirmation

8 Application Settings Layer Subsystems

The application setting layer is responsible to make the application more interactive with the user. A user can access any of the sign-out, FAQ, delete account, or/and notifications functionalities. The layer allows users to sign out or delete their account, once they use these two functionalities they have to either log in/register again to access other functionalities in the layer.

8.1 User Command

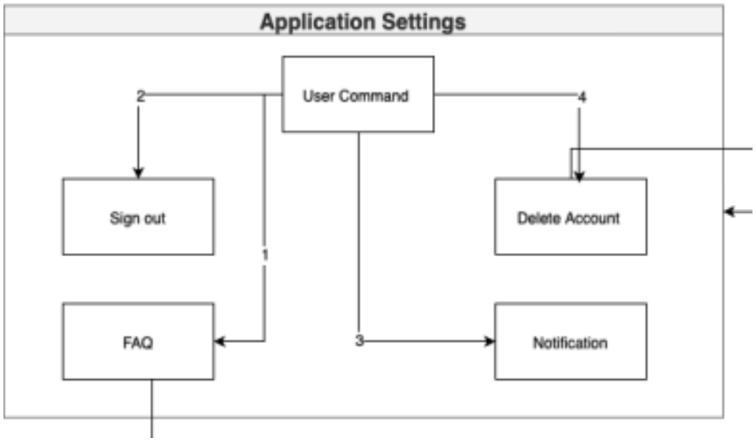


Figure 7: Application Settings Layer

8.1.1 Assumptions

- We know the User ID making the command
- The command will be either delete the account, FAQ, notification, or sign out
- The user making the command will have permission to do so

8.1.2 Responsibilities

- Allows users to sign out
- Allows users to delete their account
- Allows users to access FAQs
- Allows users to enable notifications
- Handle the user ID of the user making the command

Table 13: User Command Interfaces

ID	Description	Inputs	Outputs
#1	Sign Out	User ID	Unlink Account
#2	Delete Account	User ID	Delete User ID

8.2 Sign Out

8.2.1 Assumptions

- The user ID is known
- The user making the command exits the application

8.2.2 Responsibilities

- Unsync the user's data from the application

Table 14: Sign Out Interfaces

ID	Description	Inputs	Outputs
#1	Sign Out	User ID	UnSync Account

8.3 Delete Account

8.3.1 Assumptions

- The user ID is known
- The user selecting the delete account option

8.3.2 Responsibilities

- Unable future access to the account deleted
- Remove all the data for the deleted account

Table 15: Delete Account Interfaces

ID	Description	Inputs	Outputs
#1	Database Update	User ID	Query w/ Account deleted

8.4 FAQ

8.4.1 Assumptions

- User does not know how to use the application

8.4.2 Responsibilities

- Provides the user with an overview of how to use the application
- Answers common asked questions by other users

Table 16: FAQ Interfaces

ID	Description	Inputs	Outputs
#1	User Help	Question	Answer

8.5 Notifications

8.5.1 Assumptions

- The user ID is known
- The user has a poll scheduled

8.5.2 Responsibilities

- Get notified when the poll is about to start

Table 17: Notifications Interfaces

ID	Description	Inputs	Outputs
#1	Notify User	User ID/ Poll ID	Reminder/Pop up message

9 Poll Results Layer Subsystems

The poll results layer is responsible to display the user's poll information by having them encapsulated as a collector. If a user wishes to retrieve or update values to them, they can. This layer also performs a database query that gets validated by blockchain access to display the poll's results.

9.1 Poll Information Collector

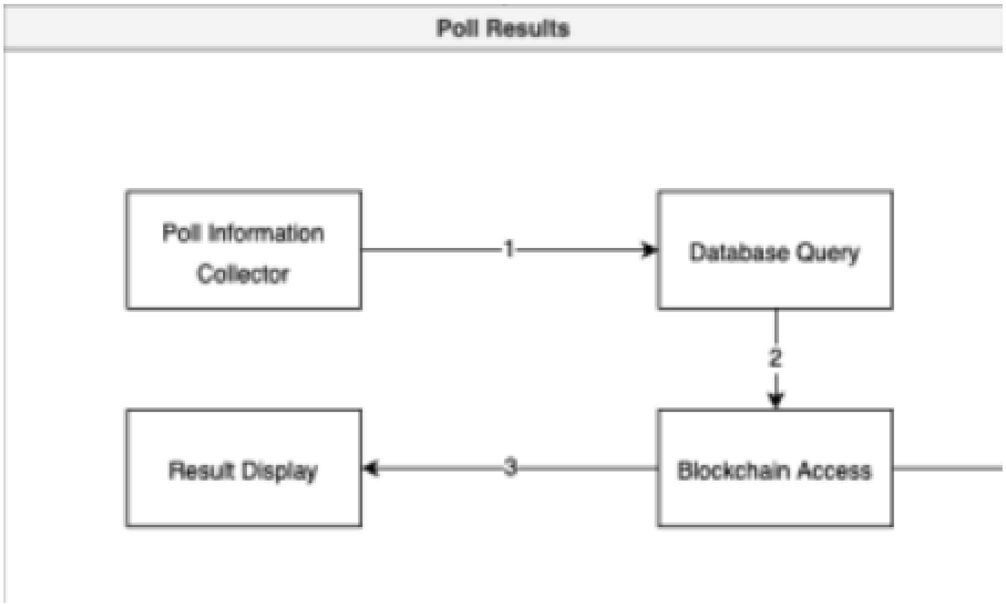


Figure 8: Poll Results Layer

9.1.1 Assumptions

- We know the user ID making the command
- We know the poll ID that the user participated in

9.1.2 Responsibilities

- Obtain all information regarding the poll

Table 18: Poll Information Collector Interfaces

ID	Description	Inputs	Outputs
#1	Poll Information Collector	User ID/Poll ID	Poll Information

9.2 Database Query

9.2.1 Assumptions

- Have access and connected to the database
- Information of the poll is stored on the database

9.2.2 Responsibilities

- Update the database with the collected information about the poll's result
- Update the blockchain with the new poll

Table 19: Database Query Interfaces

ID	Description	Inputs	Outputs
#1	Database Update	Poll ID	Query w/ Poll Update

9.3 Blockchain Access

9.3.1 Assumptions

- The poll information exists

9.3.2 Responsibilities

- validate the poll's results
- Update the blockchain with the given poll results

Table 20: Blockchain Access Interfaces

ID	Description	Inputs	Outputs
#2	Blockchain Update	Poll ID	Smart Contract Confirmation

9.4 Result Display

9.4.1 Assumptions

- The poll information exists
- The user has already participated in a poll

9.4.2 Responsibilities

- Show the poll's results to the user
- Allow the user to navigate to see the choices made for the poll

Table 21: Results Display Interfaces

ID	Description	Inputs	Outputs
#2	Display Poll Results	Poll ID	Poll Results

10 Onboarding Layer Subsystems

This serves the purpose of familiarizing the user with the application. The onboarding process happens after a user logs on for the first time. It gives a mini tutorial of all the functionalities available on the app.

10.1 Registration

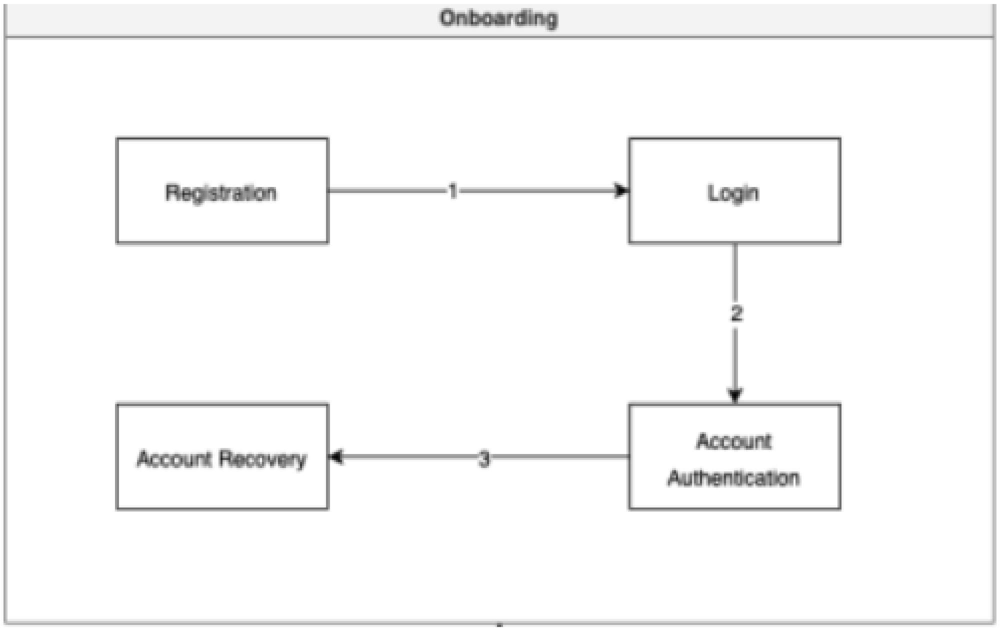


Figure 9: Onboarding Layer

10.1.1 Assumptions

- The user's credentials do not match a pre-existing account in the database

10.1.2 Responsibilities

- Allow users the capacity to use the app

Table 22: Registration Interfaces

ID	Description	Inputs	Outputs
#1	User registration	User credentials	Created user account in database

10.2 Login

10.2.1 Assumptions

- The user's credentials do not match a pre-existing account in the database

10.2.2 Responsibilities

- Allow users the capacity to use the app

Table 23: Login Interfaces

ID	Description	Inputs	Outputs
#1	User login	User credentials	Access to user account

10.3 Account Authentication

10.3.1 Assumptions

- The database is up to date with all the necessary credentials

10.3.2 Responsibilities

- Ensuring that everyone using the app is who they claim they are

Table 24: Account Authentication Interfaces

ID	Description	Inputs	Outputs
#1	Database double-check	User credentials	Access to user account

10.4 Account Recovery

10.4.1 Assumptions

- User no longer has access to their account

10.4.2 Responsibilities

- Provides the user a method to recover their account

Table 25: Account Authentication Interfaces

ID	Description	Inputs	Outputs
#1	Database double-check	User credentials	Code to reset password

11 Frontend Layer Subsystems

This is the user layer where most interactions and services will be provided to the user. It contains an application router that is used to navigate the user through the application. A poll and voting interface allows the user to create and participate in a poll. Also, an onboarding functionality that registers new users and authenticates existing ones.

11.1 User Onboarding

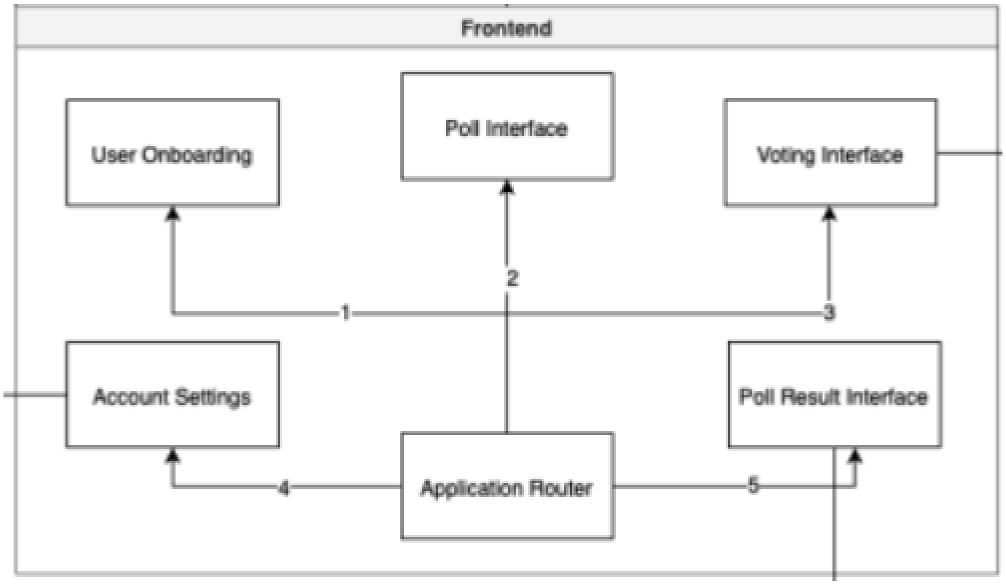


Figure 10: Frontend Layer

11.1.1 Assumptions

- The user has no prior experience with our app
- The user has experience with basic terminologies

11.1.2 Responsibilities

- Ensure the new user is proficient enough to use the app to its fullest extent

Table 26: User onboarding Interfaces

ID	Description	Inputs	Outputs
#1	User tutorial	N/A	Information on voting app

11.2 Poll Interface

11.2.1 Assumptions

- The user has done the onboarding and understands how to use the different filters

11.2.2 Responsibilities

- Ensure ease of access for the user
- Ensure the user can ask exactly what they want

Table 27: Poll interface Interfaces

ID	Description	Inputs	Outputs
#1	Poll creation	Title Start time End time Filters	Created poll

11.3 Voting Interface

11.3.1 Assumptions

- The user has completed the onboarding and understands how the page works
- User has polls they qualify for

11.3.2 Responsibilities

- Provide a place for the user where all the polls they are allowed to vote on are located

Table 28: Voting interface Interfaces

ID	Description	Inputs	Outputs
#1	User voting	Voting choice	Single registered vote

11.4 Account settings

11.4.1 Assumptions

- The user wanted to sign out of the application
- The user wanted to delete their account
- The user wanted to enable their notifications

11.4.2 Responsibilities

- Allow the user to easily change settings as they see fit

Table 29: Account settings Interfaces

ID	Description	Inputs	Outputs
#1	Edit account	N/A	Applied settings

11.5 Poll result interface

11.5.1 Assumptions

- The user ran a poll
- The user specified who could vote on the poll

11.5.2 Responsibilities

- Provide an easy and efficient method to look at the polling results

Table 30: Poll result Interfaces

ID	Description	Inputs	Outputs
#1	Poll results	User credentials Poll title	Poll winner
#2	Poll analytics	User credentials Poll ID Poll title	Voters analytics

11.6 Application Router

11.6.1 Assumptions

- The user wanted to navigate through the application

11.6.2 Responsibilities

- Ensure that the applications point of access to the cloud is uncompromised

Table 31: Application router Interfaces

ID	Description	Inputs	Outputs
#1	App's link to the cloud	App credentials	User authentication Forward requests Proxy requests

References