МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

НА ТЕМУ

Система підтримки прийняття рішень проведення рейтингового голосування(праймеріз)

Галузь знань **12 «Інформаційні технології»**

Спеціальність **122 «Комп'ютерні науки»**

Освітня програма «**Комп'ютерні науки»**

Освітній рівень: бакалавр

Виконав: студент 4 курсу, групи КН- 42

Юраков Є.Г.
(прізвище та ініціали)

Керівник Гнатієнко Г. М.
(прізвище та ініціали)

к.т.н.
(науковий ступінь, звання)

Випускна кваліфікаційна робота бакалавра допущена до захисту
рішенням кафедри *інтелектуальних технологій*
Протокол № 11 від 06.06.2022 р.
зав. кафедри_____ доц. Іларіонов О.Є.

Київ - 2022

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ

Завідувач кафедри
інтелектуальних технологій

Іларіонов О.Є.

_____

"___"_____2022 р.

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Юракову Євгенію Геннадійовичу

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)

Система підтримки прийняття рішень проведення рейтингового голосування(праймеріз)

затверджена протоколом засідання кафедри від « 23 » грудня 2021 р. № 4

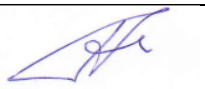
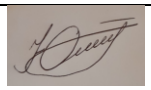
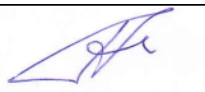
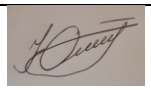
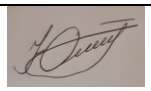
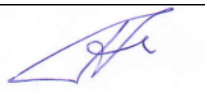
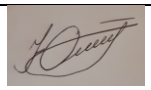
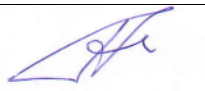
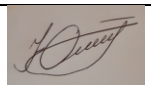
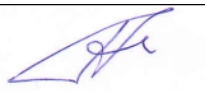
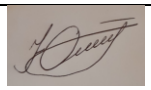
2. Термін здачі студентом закінченого проекту (роботи) 29 травня 2022 року

3. Вихідні дані до проекту (роботи)
Система підтримки прийняття рішень при проведенні рейтингового голосування(праймеріз) створюється для визначення лідерів у групі та ранжування відносного впливу цих лідерів

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
Вибір та аналіз предметної області, огляд задач голосування, праймеріз, задач ранжування та програмних систем, створених для підтримки цих задач, проектування архітектури системи, розробка системи, вибір тестових задач, візуалізація результатів, тестування роботи системи

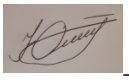
5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)
Огляд джерел інформації щодо задач рейтингового голосування, ілюстрація ситуації прийняття рішення, архітектура системи, програмна реалізація системи, презентація Power Point

6. Консультанти з випускної кваліфікаційної роботи із зазначенням розділів випускної кваліфікаційної роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Вибір та аналіз предметної області		
2	Постановка задачі рейтингового голосування		
2	Огляд підходів до визначення результуючого ранжування		
3	Проектування архітектури системи		
3	Програмна реалізація системи		
3	Вибір тестових задач, тестування роботи системи		

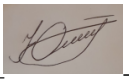
7. Дата видачі завдання 15 лютого 2022 року

Керівник __  ___ / Гнатієнко Г.М. /
(підпис) (ПІБ)

Завдання прийняв до виконання ___  ___ / Юраков Є.Г. /
(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

Пор. №	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів випускної кваліфікаційної роботи	Примітка
1.	Вибір та аналіз предметної області	15.02-22.02	
2.	Постановка задачі рейтингового голосування	23.02-25.02	
3.	Огляд підходів до визначення результуючого ранжування	26.02-03.03	
4.	Проектування архітектури системи	04.03-27.03	
5.	Програмна реалізація системи	28.03-14.04	
6.	Вибір тестових задач, тестування роботи системи	15.04-26.04	

Студент-дипломник  / Юраков Є.Г. /
(підпис) (ПІБ)

Керівник випускної кваліфікаційної роботи  / Гнатієнко Г.М. /
(підпис) (ПІБ)

Анотація

Юраков Євгеній Геннадійович виконав випускну кваліфікаційну роботу на тему «Система підтримки прийняття рішень проведення рейтингового голосування (праймеріз)» за напрямом підготовки «Комп'ютерні науки».

У випускній кваліфікаційній роботі наведено аналіз та результати досліджень у галузі систем підтримки прийняття рішень при проведенні рейтингового голосування (праймеріз), проаналізовано методи рейтингового голосування, виділено їх недоліки та переваги, розроблено систему підтримки прийняття рішень для проведення голосування та ранжування.

Ключові слова: система підтримки прийняття рішень, праймеріз, ранжування, рейтингове голосування, blockchain, розподілені системи, децентралізація, токен.

Summary

The degree project "The decision supporting system of ranked choice voting(primaries)" on the specialty "Computer science" has completed by **Yurakov Yevhenii**.

In the final qualification work the analysis of research results in the field of decision supporting systems of rating voting, there were analyzed rating voting methods with highlighted advantages and drawback and was developed the decision supporting system of rating voting.

Keywords: decision support system, primaries, ranking, rating voting, blockchain, distributes systems, decentralization, token.

ЗМІСТ

ВСТУП 8
РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ПРИ ПРОВЕДЕННІ РЕЙТИНГОВОГО ГОЛОСУВАННЯ 10
1.1. Вибір та аналіз предметної області 10
1.2. Постановка задачі рейтингового голосування 10
1.2.1. Задачі голосування 11
1.2.2. Рейтингове голосування (праймеріз) (алгоритми визначення переможців, способи задання вхідної інформації, пов'язані з ними проблеми) 12
1.2.2.1. Алгоритми визначення переможців 12
1.2.2.2. Порівняння методів рейтингового голосування 17
1.2.2.3. Способи задання вхідної інформації 18
1.3. Огляд підходів до визначення результуючого ранжування 19
1.3.1. Задачі ранжування та де воно застосовується 20
1.3.2. Варіанти задання початкових даних 21
1.3.3. Методи визначення розв'язку та їх складність 22
1.4. Програмні системи, створені для підтримки вказаних задач 27
РОЗДІЛ 2 ПРОЕКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ 33
2.1. Модель використання СППР 33
2.2. Побудова бізнес-процесів в роботі СППР 34
2.3. Алгоритм роботи СППР 36
2.4. Розробка моделі бази даних 41
РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ 44
3.2. Опис основних класів програми 47
3.3. Опис програмного інтерфейсу 48
Висновки 53
Список використаної літератури 54
Додатки 56

ВСТУП

У суспільстві вже з давніх часів важливі рішення приймаються шляхом голосування. За допомогою голосування обирають депутатів, директорів, президентів, також приймаються важливі постанови і закони в державних постановах чи університетах та школах. Людина своїм голосом часто може приймати важливі і звичайні рішення, від вибору президента країни до вибору фільму для перегляду.

Голосування може з першого погляду бути достатньо простою подією, але, якщо заглибитись у це питання, можна дізнатись, що це складний процес, адже голосування має дуже багато різних варіантів реалізації, кожен з яких повністю відрізняється своєю формулою та принципом проведення.

Одним з найбільш поширених варіантів голосування є рейтингове голосування. Це тип голосування при якому, маючи список всіх кандидатів, виборець створює свій рейтинг кандидатів, де на першому місці опиняється кандидат який йому імпонує найбільше і на останньому місці найгірший кандидат відповідно. Тобто, під час такого голосування людина віддає свій голос вже не одному кандидату, а може в своєму рейтингу на вищі позиції поставити кандидатів які на її погляд є найкращими. Саме такий вид голосування є дуже корисним у багатьох сферах, таких як вибори на посаду завідуючого кафедри, голосування на конкурсі Євробачення або голосування за найкращого футболіста року. Ця популярна виборча система дозволяє виборцям ранжувати кандидатів по вподобанням, тобто вони можуть подавати бюлетень в якому вказаний не тільки їх перший кандидат на посаду, але і другий, третій і так далі.

Як рейтингове голосування може відзначитись на виборах в Україні?

Якби рейтингове голосування було ухвалене по всій країні, це докорінно змінило б те, як відбуваються вибори в Україні. Прихильники рейтингового голосування стверджують, що плюралістична система не завжди відображає справжню волю народу. На думку експертів, це може призвести до поділу голосів між кандидатами зі схожими позиціями, внаслідок чого буде обрано

кандидата, який загалом менш популярний. Це називається ефектом "Надера" або ефектом "спойлера".

Метою даної роботи є створення програмного засобу у вигляді системи підтримки прийняття рішень при проведенні рейтингового голосування (праймеріз) яка визначає лідерів у групі та проводить ранжування відносного впливу цих лідерів

Поставлена мета досягається вирішенням наступних завдань:

- Огляд методів проведення рейтингового голосування;
- Визначення переваг і недоліків існуючих методів;
- Розробка алгоритмів для проведення рейтингового голосування;
- Програмна реалізація розробленого алгоритму з застосуванням існуючих бібліотек;
- Тестування розробленого програмного засобу.

Об’єкт дослідження – це процес проведення рейтингового голосування.

Предмет дослідження – це існуючі методи проведення праймеріз.

У першому розділі проведено огляд основних методів рейтингового голосування, розглянуто переваги і недоліки сучасних методів, обґрунтована актуальність проблеми проведення рейтингового голосування.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ПРИ ПРОВЕДЕННІ РЕЙТИНГОВОГО ГОЛОСУВАННЯ

1.1. Вибір та аналіз предметної області

В наш час звичайне голосування хоч і переважно справляється з поставленими задачами, але крім великих затрат ресурсів як людських, так і матеріальних, часто виникають великі суперечки з приводу виявленого переможця, так як таке голосування часто не демонструє волю виборців повною мірою. Саме для цього є методи рейтингового голосування. Навіть хоча його реалізація є більш складною, але це все компенсується результатом, таке голосування, складніше сфальсифікувати.

Зараз в багатьох країнах голосування переходить на систему рейтингового, створюється багато платформ завдяки яким можна легко контролювати легітимність виборів. Зрозуміло, що для України рейтингове голосування на онлайн платформі було б значним кроком вперед, і зараз вже робляться перші дії в цьому напрямку.

Але рейтингове голосування може застосовуватись не лише при виборах прозидента, а навіть при будь якому іншому голосуванні, яке людина може створити самостійно. Тобто, це може бути система підтримки прийняття рішень, в якій людина може як організатор самостійно на платформі створити власне голосування, задати кількість виборців і кандидатів, і провести його.

1.2. Постановка задачі рейтингового голосування

Рейтингове голосування – це виборча система, яка дозволяє людям голосувати за кількох кандидатів як перевагу. Замість того, об просто вибирати, кого ви хочете бачити переможцем, ви заповнюєте бюлетень, вказавши, хто є вашим першим вибором, другим вибором або третім вибором (або більше, якщо потрібно) для кожної позиції [5].

Кандидат, який набрав більшість (понад 50%) голосів першого вибору, перемагає беззастережно. Якщо жоден із кандидатів не отримує більшості голосів першого вибору, це запускає новий процес підрахунку голосів. Кандидат,

який зробив найгірший вибір, виключається, і бюлетені виборців цього кандидата перерозподіляються між його другим вибором. Іншими словами, якщо ви оцінили кандидата, що програв, як свого першого кандидата, а кандидат був виключений, то ваш голос все одно буде враховуватися: він просто перейде до кандидата, обраного вами другим. Цей процес триває доти, доки не з'явиться кандидат, який набрав більшість голосів. Для порівняння, федеральний уряд США і більшість американських штатів і міст в даний час використовують так звану систему множини: перемагає кандидат, який набрав найбільшу кількість голосів - і точка. Не має значення, чи набрав цей кандидат більшість голосів. У системі рейтингового голосування все працює інакше. Переможець кандидат майже завжди отримує більшість голосів, навіть якщо якась частина електорату обрала його чи її другим чи третім кандидатом [3].

1.2.1. Задачі голосування

Протягом життя ми отримуємо багато завдань, які вимагають від нас вибору певних варіантів дій. Людина завжди приймала рішення, покладаючись на свою інтуїцію, досвід та розум. Ухвалення рішення [10] з часом перетворилося в науку, яка для своєї реалізації використовує математичні методи дослідження. Звичайно, в найширшому обсязі ситуацій, де людям потрібно приймати рішення – від планування робочого дня до вибору свого життєвого шляху – важко передбачити, чи мають методи перевагу над інтуїцією.

Необхідність в використанні математичних методів з’явилась через наслідки прийняття рішень, які стосуються багатьох осіб і пов’язані з матеріальними витратами. Отже дуже зріз ступінь відповідальності людини за його схвалене рішення. Випадкове рішення може бути дуже збитковим та мати критичні наслідки [2].

1.2.2. Рейтингове голосування (праймеріз) (алгоритми визначення переможців, способи задання вхідної інформації, пов'язані з ними проблеми)

Праймеріз – це тип голосування, за яким виборці, партійці, обирають кандидатом одного представника від політичної партії. Голосування може бути відкритим, коли кожен громадянин може голосувати за кандидатів від різних партій, а буває і закритим – коли право голосу мають тільки члени політичної партії, яка проводить праймеріз, і коли голосують виключно за претендентів від цієї партії. У випадку, якщо два кандидати набрали найбільше голосів, можливий другий тур голосування за єдиного кандидата від політичної партії [5].

Праймеріз – це один зі способів, завдяки якому політичні партії чи блоки номінують кандидатів на майбутні вибори або ж довибори. Інший метод – це коли кандидатів номінують за зборах, з'їздах чи інших політичних зібраннях. Загалом праймеріз вважають прогресивним інструментом висування кандидатів від політичних партій, коли не керівництво партії, партійні функціонери визначають кандидатів на виборах, а самі виборці чи прості партійці [9].

1.2.2.1. Алгоритми визначення переможців

- Критерій Кондорсе

Деякі з концепцій, розроблених Маркізом де Кондорсе у вісімнадцятому столітті, досі відіграють центральну роль предметі.

Якщо є кандидат, якому більшість виборців віддає перевагу всім іншим кандидатам, цей кандидат відомий як переможець Кондорсе [2]. Метод голосування, який завжди вибирає переможця Кондорсе, якщо такий є, визначається як узгоджений з Кондорсе або (еквівалентно) задовольняючий критерію Кондорсе . Методи з цією властивістю відомі як методи Кондорсе.

Припустимо, що у виборах немає переможця Кондорсе [2]. І тут має бути цикл Кондорсе , який можна проілюструвати з прикладу. Припустимо, що є три кандидати, A, B і C, і 30 виборців, тому десять голосують C-B-A, десять голосують B-A-C і десять голосують A-C-B. Тоді немає переможця на Кондорсе.

Зокрема, ми бачимо, що A не може бути переможцем за Кондорсе, тому що 2⁄3 виборців вибрали B замість A. Проте B не може бути переможцем за Кондорсе, тому що 2 ⁄ 3 віддають перевагу C, а не B, а C не може бути переможцем за Кондорсе, тому що 2⁄3 віддають перевагу A C. Але A не може бути переможцем за Кондорсе. Таким чином пошук переможця Кондорсе заводить нас по колу, але так і не знаходить.

- Просторові моделі

Просторова модель [2] - це модель виборчого процесу, розроблена Дунканом Блеком і розширена Ентоні Даунсом. Передбачається, що кожен виборець і кожен кандидат займають становище в просторі думок, яке може мати один або кілька вимірів, і передбачається, що виборці віддають перевагу ближчому з двох кандидатів віддаленішому. Політичний спектр це проста просторова модель в одному вимірі.

Просторові моделі є важливими, тому що вони є природним способом візуалізації думок виборців. Вони призводять до важливої теореми, теореми про медіанного виборця, що також належить Блеку. Він стверджує, що для широкого класу просторових моделей, включаючи всі одновимірні моделі та всі симетричні моделі у вищих вимірах, переможець Кондорсе гарантовано існує і буде кандидатом, найближчим до медіани розподілу виборців.

Припустимо, ми застосовуємо ці ідеї до діаграми. У цьому випадку ми бачимо, що дійсно є переможець Кондорсе — B — якому віддають перевагу A на 64% і C на 66%, і що переможець Кондорсе дійсно є кандидатом, найближчим до медіани розподілу виборців.

- Інші теореми

Теорема Ерроу про неможливість проливає песимістичніше світло на рейтингове голосування. У той час як теорема про медіанного виборця говорить нам, що легко розробити метод голосування, який ідеально працює для багатьох наборів виборців, теорема Ерроу каже, що неможливо розробити метод, який ідеально працює у всіх випадках.

Питання, що більш точно описує поведінка під час виборів: песимізм Ерроу чи оптимізм Блека, необхідно визначити емпірично. Деякі дослідження, у тому числі стаття Тайдмана і Плассмана [2-3], припускають, що прості просторові моделі типу, що задовольняє теорему про медіанного виборця, дають точну відповідність поведінці виборця, що спостерігається.

Інший песимістичний результат, теорема Гіббарда (від Аллана Гіббарда), стверджує, що будь-яка система голосування має бути вразливою для тактичного голосування.

- Граф Борда

Підрахунок Борда [2] надає бал кожному кандидату шляхом додавання кількості балів, що присуджуються кожним бюлетенем. Якщо є m кандидатів, то кандидат, який посів перше місце в бюлетені, отримує m - 1 бал, другий - m - 2, і так далі, поки кандидат, який посів останнє місце, не отримає жодного балу. У наведеному прикладі обирається з 130 з 300 балів.

Кандидат	Результат
A	87
B	130
C	83

Підрахунок Борда простий у реалізації, але не відповідає критерію Кондорсе. Його особлива слабкість полягає в тому, що на результат може сильно вплинути висування кандидатів, які самі не мають жодних шансів бути обраними.

- Миттєве повторне голосування

Миттєве голосування (IRV) [5] виключає кандидатів у серії раундів, імітуючи вплив окремих бюлетенів на скорочення кількості кандидатів. Перший тур складається із поданих голосів. Оскільки ні в кого немає більшості голосів при першому підрахунку голосів, визначається кандидат із найменшою кількістю переваг першого місця (в даному випадку B) та виключається з

голосування у наступних раундах. Їхні голоси передаються відповідно до наступної зазначеної переваги, якщо така є. Таким чином, у другому турі голоси віддають перевагу лише двом кандидатам (у більш загальному випадку m – 1). Ми зупиняємось, тому що A є перевагою більшості виборців.

Підраховані бюлетені	1-ий раунд	2-ий раунд	3-ій раунд
36	A-B-C	A-C	A
15	B-A-C	A-C	A
15	B-C-A	C-A	A
34	C-B-A	C-A	A

Системи виключення вимагають повторної перевірки кожного голосу кожному раунді, а чи не дозволяють проводити обчислення з урахуванням простої таблиці похідних статистичних даних. IPB не відповідає критерію Кондорсе. На відміну від більшості рейтингових систем голосування, IRV не допускає однакових переваг, за винятком випадків, коли виборець обирає найменш бажаних кандидатів.

- Єдиний голос, що передається

Єдине голосування, що передається (STV) [5] - це пропорційна версія IRV з декількома переможцями. Як і в IRV, у STV вторинні переваги є умовними голосами і використовуються тільки тоді, коли перша перевага не може бути ефективно використана. Відповідно до STV кожен виборець має лише один голос (але може відзначати додаткові уподобання), і голос виборця спочатку розподіляється за найкращого кандидата. Після (або якщо) кандидата (ів) обрані (переможці) за квотою , зайві голоси передаються від переможців до кандидатів (які сподіваються) відповідно до замовних уподобань виборців. Якщо місця все ще необхідно заповнити, кандидат (и) виключаються, а їх виборці переходять до кандидатів, що залишилися. Різні форми STV можуть мати різні способи виключення кандидатів та проведення передачі голосів.

- Метод Коупленда

Метод Коупленда [2] надає кожному кандидату бал, отриманий з таблиці результатів, як показано вище для мінімаксу. Бал — це просто кількість сприятливих результатів серед кандидата, тобто кількість інших кандидатів, якому більшість виборців віддали перевагу конкретному кандидату. Кандидат із найбільшою кількістю балів перемагає.

Метод Коупленда сумісний з Кондорсе і прямолінійний, але для конкретних моделей переваг виборців (без переможця Кондорсе) він дасть нічию, хоч би яким великим був електорат. Тому його прихильники зазвичай рекомендують використовувати його у поєднанні з тай-брейком. Придатні правила цієї мети включають мінімакс, IRV і підрахунок Борда, останнє з яких дає метод Дасгупта-Маскіна.

Інші способи

- Рейтингове голосування відрізняється від кардинального голосування, за якого кандидати оцінюються незалежно, а не ранжуються.
- Завершення Кондорсе обирає переможця Кондорсе [3], якщо він є, і інакше вдається до окремої процедури визначення результату. Якщо підрахунок Борда є запасним варіантом, ми отримуємо метод Блека; якщо ми використовуємо IRV, ми отримуємо "Condorcet-Hare" Тайдмана.
- Метод Кумбса є простою модифікацією IPB. Кандидат, що вибув у кожному раунді, має більше переваг для останнього місця, ніж найменша кількість варіантів для першого місця (отже C, а не B вибуває в першому раунді прикладу, і B є переможцем). Метод Кумбса не є узгодженим з Кондорсом, але задовольняє теорему про медіанного виборця. У нього є недолік, який полягає в тому, що він спирається в основному на переваги виборців про останні місця, які можуть бути обрані з меншою обережністю, ніж їхні перші місця.
- У методах Болдуїна та Ненсона використовуються складніші правила виключення, засновані на підрахунку Борда. Вони узгоджуються з Кондорсом.
- Метод Кемені-Янг складний, але сумісний з Кондорсе.

- Метод Сміта зводить набір кандидатів до безлічі Сміта, що складається з одного елемента, що включає переможця Кондорсе, якщо він є, і в іншому випадку зазвичай менше вихідного набору. Зазвичай рекомендується використовувати його у поєднанні з тай-брейком, найбільш поширеними є IRV та мінімакс. Він простий у обчислювальному відношенні, хоч і не інтуїтивно зрозумілий більшості виборців.
- Умовне голосування є двоетапною версією IRV, а додаткове голосування є обмеженою формою умовного голосування.
- Метод Бакліна існує у кількох формах, деякі з яких узгоджуються з Кондорсом.
- Метод ранжованих пар, метод Шульце та метод розділеного циклу є узгодженими за Кондорсом методами середньої обчислювальної складності, заснованими на аналізі циклічної структури бюлетенів.
- Метод Доджсона відомий головним чином тим, що він був розроблений Льюїсом Керролом. Він узгоджений з Кондорсом, але складний у обчислювальному відношенні.
- Розширення правила затвердження

1.2.2.2. Порівняння методів рейтингового голосування

Найпростіша форма порівняння – аргументація на прикладі. Логічні критерії голосування можна представити як екстраполяцію характерних особливостей прикладів на нескінченні простору виборів. Наслідки часто важко передбачити: спочатку розумні заходи суперечать та відкидають задовільні в інших відносинах методи голосування.

Емпіричні порівняння можуть бути виконані з використанням моделюваних виборів. Популяції виборців та кандидатів будуються за просторовою (або іншою) моделлю. Точність кожного методу голосування, яка визначається як частота, з якою він обирає кандидата, найближчого до центру розподілу виборців, можна оцінити за допомогою випадкових випробувань. Способи Кондорсе (і спосіб Кумбса) дають кращі результати, за ними слід

підрахунок Борда, з деяким відставанням від IRV і найгіршим з усіх з першим минулим.

Математичні властивості методів голосування мають бути збалансовані із прагматичними характеристиками, такими як зрозумілість для середнього виборця.

1.2.2.3. Способи задання вхідної інформації

На основі традиційних підходів можна побудувати комбіновану експертизу, яку легко реалізувати, але її аналіз потребує адекватних та обгрунтованих методів обчислення результатів. Методи обробки експертної інформації [1] поділяються на три основні групи :

- статистичні методи;
- методи шкалювання;
- алгебраїчні методи.

Суть алгебраїчних методів полягає в тому, що на множині допустимих

оцінок задається відстань і результуюча оцінка визначається як така, відстань якої до оцінок експертів за певним вибраним критерієм є мінімальною. На основі алгебраїчного підходу будемо визначати коефіцієнти відносної компетентності експертів.

1.2.2.4. Проблеми пов’язані з голосування

Дослідження переваг рейтингового голосування

Рейтингове голосування [11-13] може призвести до менш негативної кампанії, каже Річард ДеЛеон, який досліджує рейтингове голосування у Державному університеті Сан-Франциско. Менш суперечливе політичне середовище також може допомогти кандидатам-жінкам, представникам меншин, центристським і стороннім кандидатам.

Дослідження показало, що рейтингове голосування призвело до невеликого відсоткового збільшення кількості кандидатів-жінок. Це може бути пов'язано з тим, що жінок можна «утримати від участі у виборах, якщо їм

доведеться вести негативну кампанію, щоб перемогти», — пояснює Сара Джон, автор дослідження та колишній директор із досліджень FairVote. «Згодом ми очікуємо, що кандидати навчаться проводити ефективну передвиборчу кампанію з рейтинговим голосуванням, яка рідше включатиме «вихід у мінус». Дослідження також показало, що жінки загалом і жінки з меншин з більшою ймовірністю перемагають у рейтингових системах голосування. Частково це відбувається через несвідоме «балансування» квитків, яке схильні практикувати багато виборців.

На думку експертів, системи ранжирування також мають тенденцію віддавати перевагу центристським кандидатам, оскільки система дозволяє виборцям віддавати перевагу одностороннім, упередженим кандидатам на свій вибір, а також помірним кандидатам, які мають ширшу привабливість. Кандидати-партійці мають вужчу привабливість, тому вони з меншою ймовірністю будуть другим та третім вибором для виборців, ніж кандидат-центрист. Це також може мотивувати партійних кандидатів уникати крайнощів, а також дати стороннім, центристським кандидатам більше стимулів для участі у виборах.

Недоліки рейтингового голосування

Рейтингове голосування [8] дозволяє виборцю обирати кандидатів, які більш точно відображають його переваги, ніж бюлетені за принципом «першим минулим». Проте вони складніші, а процедура підрахунку, якщо виконувати її вручну, складніше та повільніше.

1.3. Огляд підходів до визначення результуючого ранжування

Інформація експертів, що отримується шляхом одного методів потребує додаткового дослідження, обсяг якого зв’язаний з особливостями використаного методу. Зазвичай, метод обробки експертної інформації виконується особою-організатором.

Дані, що зібрані шляхом даних методів [12]: ситуаційний аналіз, рецензія, метод сценарію та інших, має вид закінченого висновку, або створеного самими

експертами тексту зафіксованого організатором використовуючи аудіо чи відео запис.

Дані, отримані під час соціологічних методів і схожих до них методів інформаційної взаємодії, потребують більшої роботи. Особливість даних методів є те, що кожен експерт діє самостійно, потім результати всіх експертів потрібно об’єднати. В залежності від типу інформації використовуються кількісні чи якісні методи обробки. Попередні кількісні дані зводять до загальної форми так, щоб їх можна було використати для повного аналізу, такими способами обробки інформації:

- упорядкування значень
- угруповання
- визначення медіани
- ранжування
- розрахунки середніх значень
- розрахунки індексів і т.д.

1.3.1. Задачі ранжування та де воно застосовується

Ранжування – це процес присвоєння рангів, відносних балів за якісними ознаками такими як розташування факторів у порядку суттєвості чи значимість в заданому дослідному контексті. Зазвичай ранжування використовується тоді, коли складно встановити безпосередньо оцінку. Також ранжування має інформацію тільки про те який об’єкт є кращим, але не має інформації про те наскільки різняться об’єкти.

Важливим способом представлення експертної інформації є ранжування альтернатив. У багатьох практичних задачах виникає необхідність використання множинних порівнянь або неповних ранжувань. До задач визначення узагальненого ранжування альтернатив застосовувалися класичні методи теорії голосувань. Однак, у таких випадках перспективним є використання алгебраїчних методи обчислення медіани, яка відображує колективну думку експертів.

1.3.2. Варіанти задання початкових даних

1.Класифікація за ступенем автоматизації

а) Не автоматизовані, діалогові методи [3]

- Фокусоване інтерв'ю з експертом (при заданому списку тем та відкритих запитань)
- Аналіз протоколів – розшифровка думок експертів, які прозвучали в ході роботи
- Генерування ідей – форсоване генерування великої кількості гіпотез групою фахівців
- Спостереження за співучасниками експертизи
- Психологічне шкалювання – введення шкал
- Структуроване інтерв'ю – інтерв'ю на основі запланованої структурованою програми діалогу
- Інтерв'ю з навчанням – інженер по знаннях демонструє розуміння експерта шляхом перефразування або розв'язання проблеми
- Одержання неточних знань, коли експерт висловлює неточні знання (гіпотези)
- Неструктуроване інтерв'ю – на основі загальних запитань
- Самоспостерігання, самозвітність
- Критичний огляд

б) Автоматизовані (інтерактивні) або частково автоматизовані [15] (змішані) методи

- Методи, які ґрунтуються на інтерв'ю
- Інтерв'ю із змінною ініціативою, коли дослідник бере інтерв'ю у експерта і навпаки
- Аналіз узгодженості експертної інформації
- Планування на основі діаграм впливу

- Моделювання – використання моделей предметної області при інтерв'ю (поглибленої моделі, причинної, когнітивної), можливо незалежно від засобів Одержання експертної інформації
- Використання багатьох джерел незалежно, а після цього їх порівняння та узагальнення
- Використання різних методів експертного оцінювання
- Аналіз протоколів
- Репертуарні решітки
- Аналіз текстів
- Методи, які ґрунтуються на навчанні
- Аналогія – застосування способів експертного оцінювання для аналогічних ситуацій
- Учнівство – навчання шляхом спостереження за роботою експерта
- Вибір зразків – вибір відповідної множини зразків для навчання
- Навчання на основі пояснення – вивід загального правила з прикладів
- Дерево розв'язків – створення та аналіз дерев розв'язків
- Індуктивний вивід моделі за прикладами
- Вивід правил з інших форм знань
- Вивід по аналогії – порівняння схожості з прикладами з позитивним результатом пошуку несправності на відміну від негативних
- Вивід часткових результатів на основі загальних законів

2.Класифікація за характером методології

- Керовані (прямі) методи (інтерв’ю; перелік запитань; аналіз розв'язків; аналіз протоколів; аналіз виводів)
- Некеровані (непрямі) (багатофункціональне шкалювання; ієрархічна кластеризація; мережі з узагальненою вагою; впорядковані дерева; репертуарні решітки)

1.3.3. Методи визначення розв’язку та їх складність

За постановкою питання та формою відповіді виділяють основні варіанти для проведення експертного оцінювання [3]:

- банальних оцінок
- абсолютних оцінок
- ранжування
- відносних оцінок
- попарного порівняння

Метод бального оцінювання базується на використанні шкали балів, її межі вказано та оприлюднено експертам.

Коли експерти мають однакові права, то використовується нацпростіша групова оцінка x_i , вона вираховується як середнє арифметичне бальних оцінок експертів по кожному і-му об’єкту за вказаною формулою:

$$x_i^{ca} \frac{1}{l} \sum_{j=1}^{l} x_{ij}$$

де x_{ij}– бальні оцінки і-их об'єктів j-ми експертами, m – загальна кількість об'єктів, l – загальна кількість експертів.

Якщо кожен експерт має різний вплив(відповідно до досвіду, ефективності проведення експертиз, компетентності тощо), то загальна групова бальна оцінка об’єктів буде обчислена як середньозважена:

$$x \ {}_i^{C3} = \sum_{j=1}^{l} \quad q_i x_{ij}; i = 1, \mathrm{m}; \sum_{j=1}^{l} \quad q_i = 1$$

де $q_i(x_{ij} = \frac{1}{p} \sum_{k=1}^{p} \quad x_{ijk})$ – ваговий коефіцієнт компетентності експерта (визначено суб'єктивно).

Якщо важливість ознак буде різною для досліджуваного об’єкта і експерти матимуть різну вагу, то групова бальна оцінка об’єкта вираховується за формулою:

$$x_{i_i}^{pb} = \frac{1}{l}\sum_{j=1}^{l} \quad q_i x_{ij} = \frac{1}{lp}\sum_{j=1}^{l} q_j \sum_{k=1}^{p} a_k \quad x_{ijk};\ i = \overline{1,m}$$

Де $x_{i_i}^{pb} = \frac{1}{p}\sum_{k=1}^{p} \quad a_k x_{ijk}\left(\sum_{k=1}^{p} \quad a_k = 1\right)$ – ваговий коефіцієнт ознаки об'єкта. Величина q_jта a_kзручно задати або визначити таким чином, щоб їх числове значення містилося в межах від 0 до 1.

Ваговий коефіцієнт компетентності експерта q_jта ознаки об'єкта a_k $\boldsymbol{a_k}$ можна визначити за допомогою диференціального самооцінюванням і взаємного оцінювання.

Під час **диференціального самооцінювання** [4] оцінку визначають за допомогою двох груп критеріїв: за критеріями, які показують ознайомлення експертів з основними джерелами даних в досліджуваній області, та за допомогою критеріїв, що характеризують ознайомлення з об'єктами експертизи.

Метод взаємооцінювання [4] характеризує будування матриці, в якій елементами будуть числа – взаємне оцінювання експертів (тобто, це може виявитись кількістю експертів, що ставлять і-го експерта більш компетентним, ніж j-го).

Головні переваги даного методу:

– просте визначення групової оцінки об'єктів після закінчення проведення експертизи;

– наявність можливості врахувати компетентність експерта;

– наявність можливості проаналізувати використовуючи як кількісні, так і якісні методи, це дає можливість порівняння результатів.

Коли висновки однакові, можна зробити висновок, що вони правильні та відповідають матеріалам експертизи, а не методам обробки даних.

Головні недоліки методу можуть бути пов'язані з проблемами отримання об'єктивної початкової оцінки $\boldsymbol{x_{ij}, q_j, x_{ijk}}$. На додачу, це дуже важка і довга робота.

Метод абсолютних оцінок [4] використовує числову шкалу оцінок, її межі визначені технічними властивостями об'єкта. Оцінка є фізичною величиною в певних одиницях вимірювання, тобто замість бальної оцінки (x_{ij})у наведених вище формулах використовується абсолютна оцінка.

Метод відносних оцінок [4] Експерти надають відносні оцінки якості об'єктів. Даний метод базується на використанні бальної або числової шкали відношень і застосовується, наприклад, в оцінюванні коефіцієнта відносної важливості цілі стратегії чи відносної важливості критеріїв. Враховуючи це, групова оцінка вираховується використовуючи формули середнього арифметичного та середньозваженого групових бальних оцінок. Сума всіх відносних оцінок повинна бути рівною 1.

Метод ранжування [4-5] Експерт оцінює якість об'єкта використовуючи встановлення їх рангу, а саме їх порядкового номеру, коли об’єкти розташувати у порядку підвищення їх якості. Таким чином, якщо об'єкт отримає більшу (меншу) суму 109 рангів від всіх експертів, то вища (нижча) буде його якість. Сума рангів об'єкта рахується за формулою:

$$r_i = \sum_{j=1}^{l} r_{ij}$$

де r_i– ранг об’єкта.

Середнє арифметичне рангу для кожного і-го об'єкта що проходить експертизу за оцінкою l експерта для матриці експертизи $\boldsymbol{R} = (\boldsymbol{r_{ij}})_{\boldsymbol{mxl}}$ рахується за формулою:

$$\overline{r_i} = \frac{1}{l} \sum_{j=1}^{l} r_{ij}\,, \qquad i = \overline{1,m}$$

Її використовують коли експерти мають можливість порівняння об'єктів між собою попарно і таким чином встановити найкращий об'єкт по кожній парі. Кожен з експертів має заповнити свою таблицю порівнянь. Порівнявши об'єкти кожної з пар, експерт виписує номер (і чи j) найкращого об'єкта з кожної пари в клітинку, що знаходиться в і-му рядку та j-му стовпці.

Взагалі, існують два методи: метод повних попарних порівнянь (коли заповнюється вся таблиця) та частинних попарних порівнянь (коли заповнюється половина таблиці).

Після того, як таблицю заповнено, методом частинних попарних порівнянь вираховується f_{ij}– частота переваг і-го об'єкта по оцінці j-го експерта (кількість чисел (і) в таблиці j-го експерта).

Середні частоти переваг f і для і-го об'єкта по усім експертам рахується за формулою:

$$f_i = \frac{1}{l}\sum_{j=1}^{l} f_{ij}\,, \qquad i = \overline{1,m}$$

Повна кількість порівнянь N, які виконано кожним експертом за методом частинних попарних порівнянь, вираховується за формулою:

Оцінка якості c_i і-го об'єкта обчислюється таким чином $N = \frac{m(m-1)}{2}$;

$$c_i = \frac{f_i}{N} = \frac{2f_i}{m(m-1)}; i = \overline{1,m}\ (\sum_{i=1}^{m} c_i = 1)$$

№ об’єкта	1	2	3	…	m
1	x	№	№	№	№
2		x	№	№	№
3			x	№	№
…				x	№
m					x

Метод повних попарних порівнянь [4-5] базується на повному заповненні таблиці, усі пари об'єктів потрібно порівняти два рази і використовується оцінювання якості роботи експертів, їх точність і запобігти можливій помилці. В ідеальному випадку матриця повинна бути симетрична відносно головної діагоналі.

Оцінка якості c_i і-го об'єкта:

$$c_i = \frac{f_i}{N} = \frac{f_i}{m(m-1)}; i = \overline{1,m} \left(\sum_{i=1}^{m} c_i = 1\right)$$

Головні переваги методів попарних порівнянь:

– просте у заповненні початкових матриць;

– чіткість математичного обґрунтування проведених операцій;

– наявність можливості для використання іншого подання експертної інформації (ранжування, бальні оцінки тощо).

1.4. Програмні системи, створені для підтримки вказаних задач

1)Платформа «Голос» у Білорусі

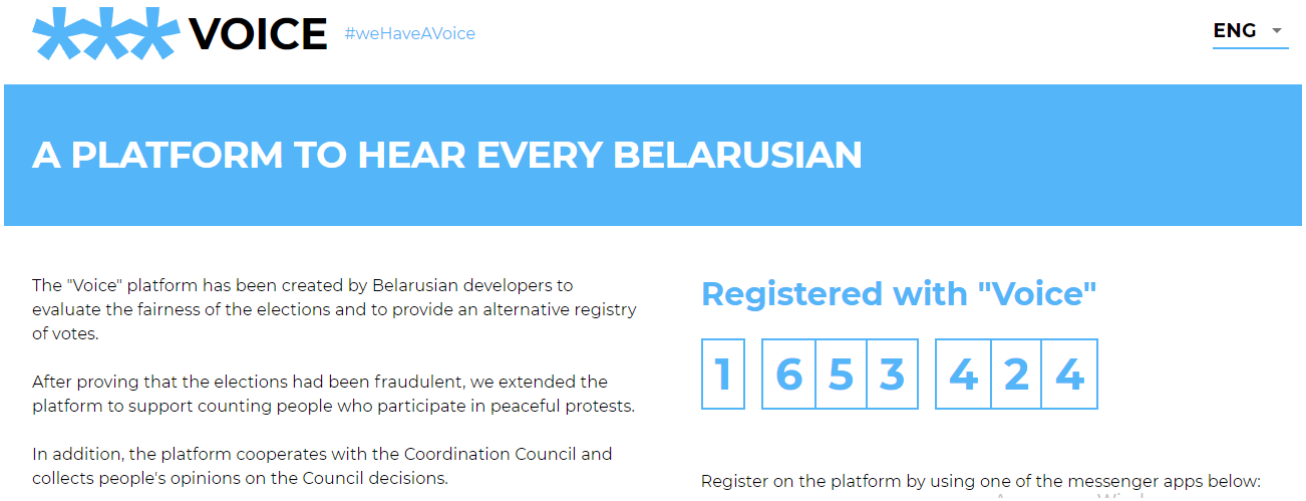


Рисунок 1.1 Платформа «Голос»

Платформа "Голос" була створена білоруськими програмістами для контролю за чесністю виборів та альтернативного підрахунку голосів.

Після доказу фальсифікацій на виборах було розширено функції платформи, щоб вона дозволяла підраховувати кількість людей, які виходять на мирний протест.

На додаток до цього платформа взаємодіє з Координаційною Радою для отримання думки білорусів про рішення, що приймаються радою.

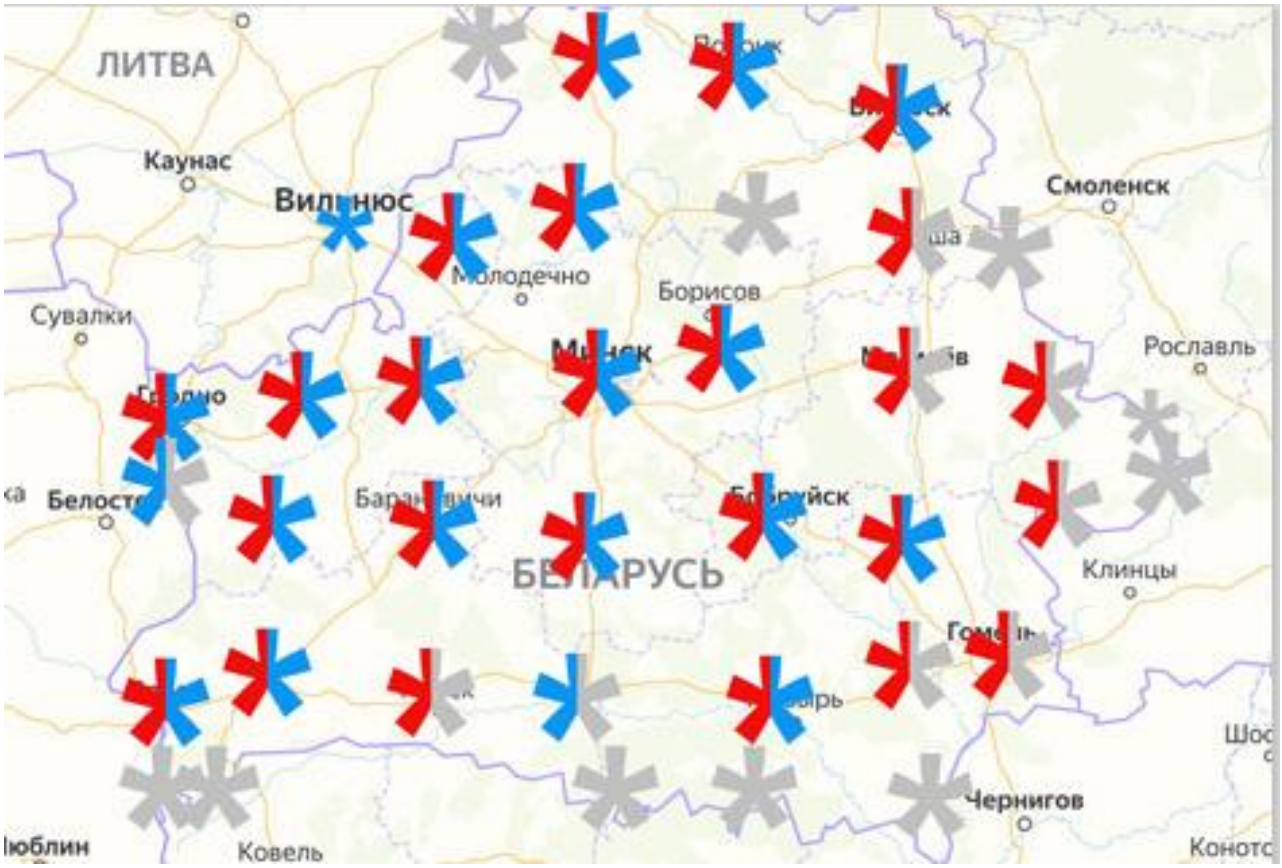


Рисунок 1.2 Платформа «Голос»

2) Українським прикладом є система голосування в Верховній Раді під назвою «Рада»

Рисунок 1.1 Система «Рада 4»

Під час роботи народних депутатів України використовується інформаційно-аналітична система голосування «Рада», вона полягає в системі поіменного електронного голосування .

Інститутами з проблем математичних машин та систем Національною академією наук України (ІПММС НАНУ) було створено та запроваджено декілька поколінь автоматичних систем інформаційної підтримки законодавчої діяльності народних депутатів України(«Рада-1», «Рада-2», «Рада-3»)

Система «Рада-1» перший раз працювала 15 травня 1990 року в парламенті УРСР. Використання системи «Рада-2» розпочалося восени 1993 року , тоді депутати починали використання мікрофонів, вбудованих у пульт на робочому столі. «Рада-3» стартувала 3 вересня 2002 року , і на депутатському столі з'явилися екрани.

Система «Рада-3», більш того, добре справляється з функціями супроводження голосування, супроводження виступів депутатів, інформаційного обслуговування депутатів за робочим місцем та контролем роботи голосування

Система починає отримувати голоси тільки після того, як стартує таймер. При цьому депутати можуть натиснути будь-які з кнопок (за, проти, утримався), але рішення фіксує лише ту кнопку, яку було натиснуто останньою .

Під час проведення голосування система вмикає 10-секундний таймер і відрахунок часу відбувається під супровід спадної послідовності десяти звуків: F5 (698,46 Гц), E♭5 (622,25 Гц), D5 (587,33 Гц), C5 (523,25 Гц), B♭4 (466,16 Гц), A♭4 (415,30 Гц), G4 (392,00 Гц), F4 (349,23 Гц), E♭4 (311,13 Гц), G6 (1567,98 Гц). Ці звуки є незмінними вже довго років і добре відомі кожному українцю .

2 березня 2021 року Верховна Рада почала використовувати в залі пленарних засідань сенсорну кнопку, яка зчитує відбиток пальця.

Система Рада-4 два роки безпомилково запобігає кнопкодавству у Київраді

Застосування системи для голосування «Рада-4» в сесійній залі Київської міської ради 2015 року було найбільшим кроком для подолання кнопкодавства. Це сказав керівник по справах секретаріату Київської міські ради Ігор Хацевич представникам виконавчого апарату Вінницької обласної ради під час обміну досвідом. Він говорив, що застосунок характеризує користування пультами, де

можна знайти та ознайомитись з порядком денним, а також отримати запис на виступ.

“Система «Рада-4» передбачає, що депутат Київської міської ради зареєструється використавши спеціальну персональну картку, а голосування передбачає натискання двох кнопок в один момент. При цьому тримати кнопки натиснутими необхідно весь час, поки голосування триває. Отже, це робить неможливим голосувати за іншу людину”, - зазначив керуючий справами.

Київська міська рада – орган місцевого самоврядування який став першим і зараз єдиним, що використовує нову систему голосування Рада-4. Саме тому досвід Київської міської ради може стати прикладом для запровадження практики викорінення кнопкодавства. Крім цього, Київська міська рада є зразковим органом місцевого самоврядування і в інших питаннях.

3) Нещодавно у додатку Дія з’явився сервіс для проведення опитуваннь, його можна знайти перейшовши в розділ Послуги.

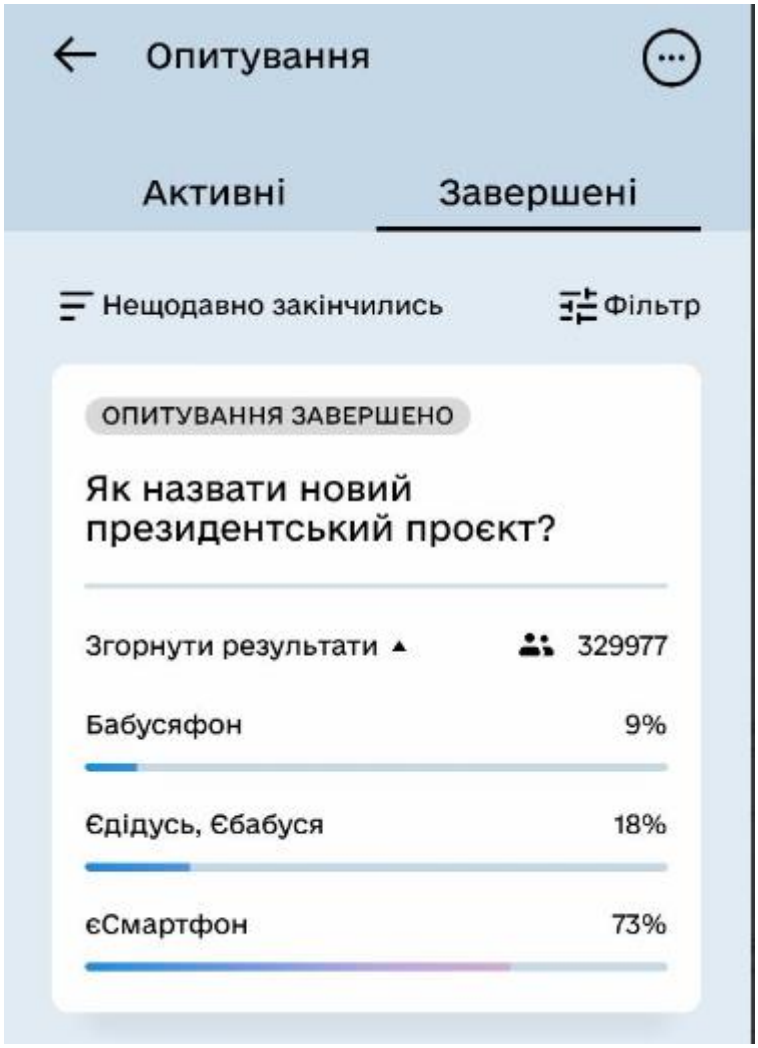


Рисунок 1.1 Додаток «Дія»

Зараз він використовується більше для тесту, для вирішення не надто важливих питань, як наприклад: назву для програми фонду боротьби з коронавірусом який забезпечує літніх людей смартфоном.

Можливо у найближчому майутньому завдяки таким опитуванням можна буде проводити референдуми, і ці опитування будуть законодавчо визнані.

Також є дуже багато веб-сайтів з допомогою яких можна створювати голосування та опитування для визначення думки людей які їх проходять. Серед таких платформ варто виділити такі:

- Testograf

Протягом тестового періоду можна створювати будь-яку кількість опитувань. Брендувати форми, додавати графіки. Налаштування логіки опитування з відповідей користувачів. Опитування можна створити на 20 типів питань. Є можливість упорядкованого збору статистики, перевірки коректності роботи респондентів. Шлях отримання відповідей — сайт, соціальні мережі, розсилка. Основний мінус — цінова політика, яка не відповідає чинним можливостям персоналізації анкет.

- Simpoll

Наявна безкоштовна версія, в якій можна створювити необмежену кількість опитувань з 10 питаннями в анкеті. Максимальна кількість респондентів — 100 осіб. Інтеграція з сервісами проводиться по API.

- Survio

Безкоштовна версія дозволяє створити всього 5 анкет, але з необмеженою кількістю питань в кожній. Приблизно 100 готових шаблонів питань, російськомовний інтерфейс. Є можливість розмістити опитування на сайті, в листі, соцмережах. Персоналізація — 70+ дизайнів, можливість зміни кнопок, відео та фото.

- Google Forms

Безкоштовний сервіс з доступним україномовним інтерфейсом для анкетування та опитувань від Google. Число анкет і питань необмежено. Шляхи

поширення стандартні — сайт, розсилка, соціальні мережі. Файли з результатами зберігаються на Диску або зводяться в таблиці даних.

Також є такі сервіси для проведення голосувань

- Typeform
- SurveyGizmo
- Anketolog
- Online Test Pad
- MySurveylab
- Surveynuts
- Oprosso
- SurveyMonkey
- Яндекс.Погляд
- Zoho Survey
- Survey Planet
- Та інші

РОЗДІЛ 2 ПРОЕКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ

2.1. Модель використання СППР

Якщо детально розглянути СППР, то можна побудувати дві діаграми прецедентів для користувача та адміністратора.

Проаналізувавши було виділено 3 акторів:

Користувач – автоизований користувач СППР, що може проходити голосування та переглядати доступну інформацію.

Адміністратор – також авторизований користувач СППР, який має всі можливості звичайного користувача, а також може виконувати всі можливі дії з голосуванням

Спостерігач – авторизований користувач СППР, що може спостерігати за процесом йому призначеного голосування

Рисунок 2.1 – Діаграма клієнтського сервісу

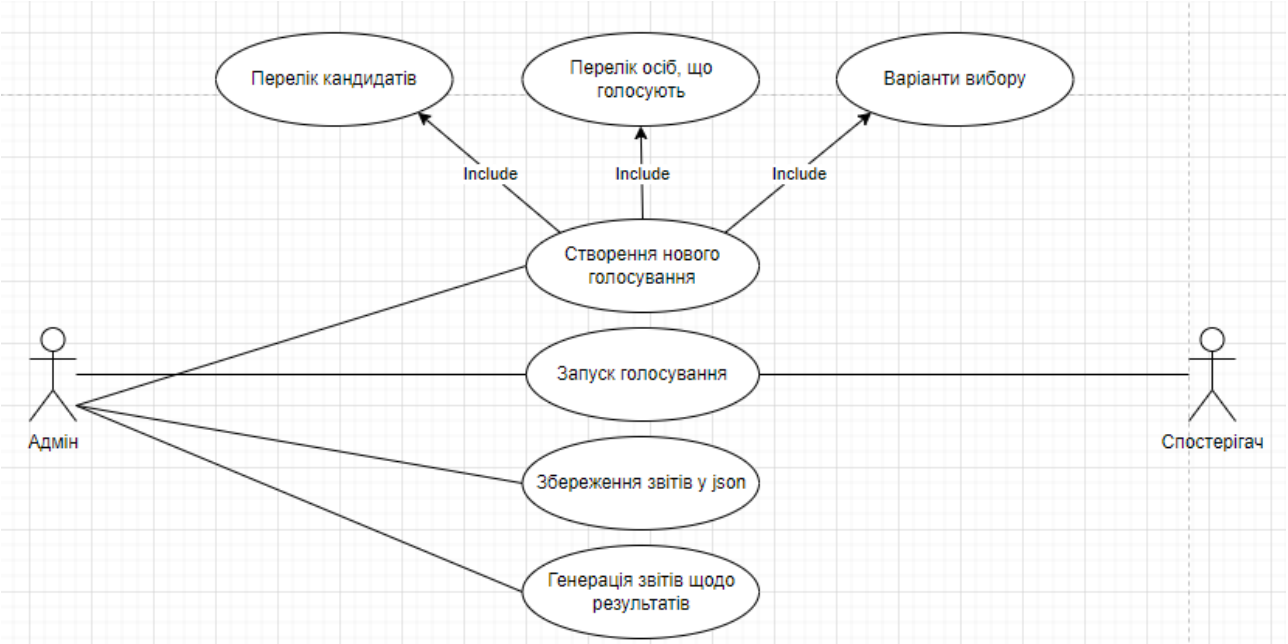


Рисунок 2.2 – діаграма адмін сервісу

2.2. Побудова бізнес-процесів в роботі СППР

В основному СППР представлено к вигляді діаграми IDEF0 (рис. 2.3)

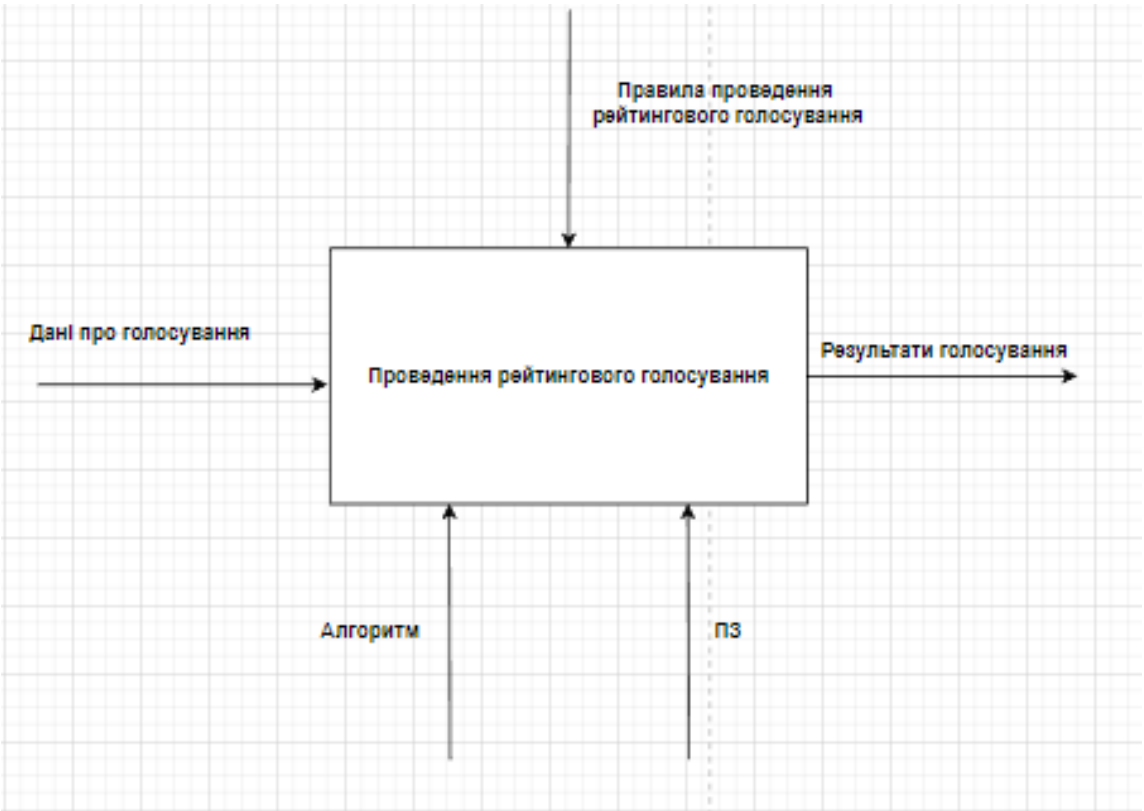


Рисунок 2.3 Діаграма IDEF0

Вхід: дані про голосування.

Управління: правила попередньої обробки.

Механізми: метод рейтингового голосування, програмне забезпечення.

Вихід: результати голосування.

На діаграмі показано діаграму IDEF0. На діаграмі показано головний процес в СППР проведення рейтингового голосування. Проаналізувавши діаграму, на вході ми отримуємо дані про нове голосування. Після отримання даних програмне забезпечення зберігає їх в базу даних. Наступним кроком система запускається у виконання. Після проведення голосування отримуємо його результати. Пройшовши алгоритм рейтингового голосування, видаливши найгірших учасників, система переходить до визначення переможця. Після цього дані записуються в базу даних для остаточного оформлення звітності. Основний процес проведення рейтингового голосування представлений у вигляді діаграми декомпозиції на рисунку 2.4:

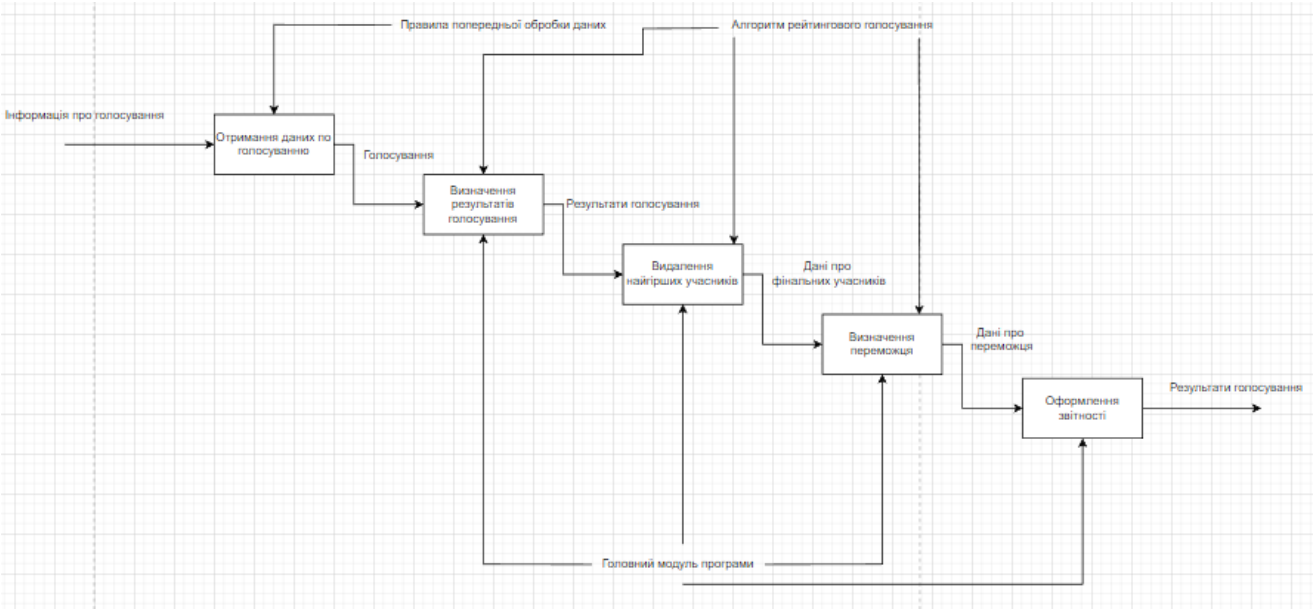


Рисунок 2.4 Діаграма декомпозиції IDEF0 проведення рейтингового голосування

На діаграмі видно, що після отримання інформації про голосування воно проходить попередню обробку. Слідом йде безпосереднє рейтингове голосування і виведення підсумкових результатів.

Кожен раз при отриманні даних про голосування їх потрібно привести в належний вигляд для можливості запуску. Відбувається сортування введених даних по таблицям в базі даних. Далі відбувається підрахунок голосів, видалення кандидатів з найменшою кількістю голосів та визначення переможця [14].

Якщо алгоритм знайшов переможця, то на виході маємо результат голосування, якщо ж переможця не знайдено – цикл повторюється. На рисунку 2.7 представлено опис алгоритму рейтингового голосування.

2.3. Алгоритм роботи СППР

Розроблену СППР для проведення рейтингового голосування можна поділити на такі етапи:

1) Користувач відкриває розроблений веб-додаток.

2) Користувач для отримання доступу повинен увійти за своїми даними, або за даними адміністратора.

3) Після входу в систему користувач переходить на головну сторінку, або на сторінку адміністратора.

4) На головній сторінці користувач може виконати дії такі як: проходження голосування, перегляду інформації та перегляду результатів. На сторінці адміністратора можна виконати всі дії пов’язані з голосування(створення, редагування, видалення, перегляд).

5) Після виконаних дій всі зміни зберігаються в базі даних.

Даний процес зображено на діаграмі (рис. 2.5) та діаграмі (рис. 2.6)

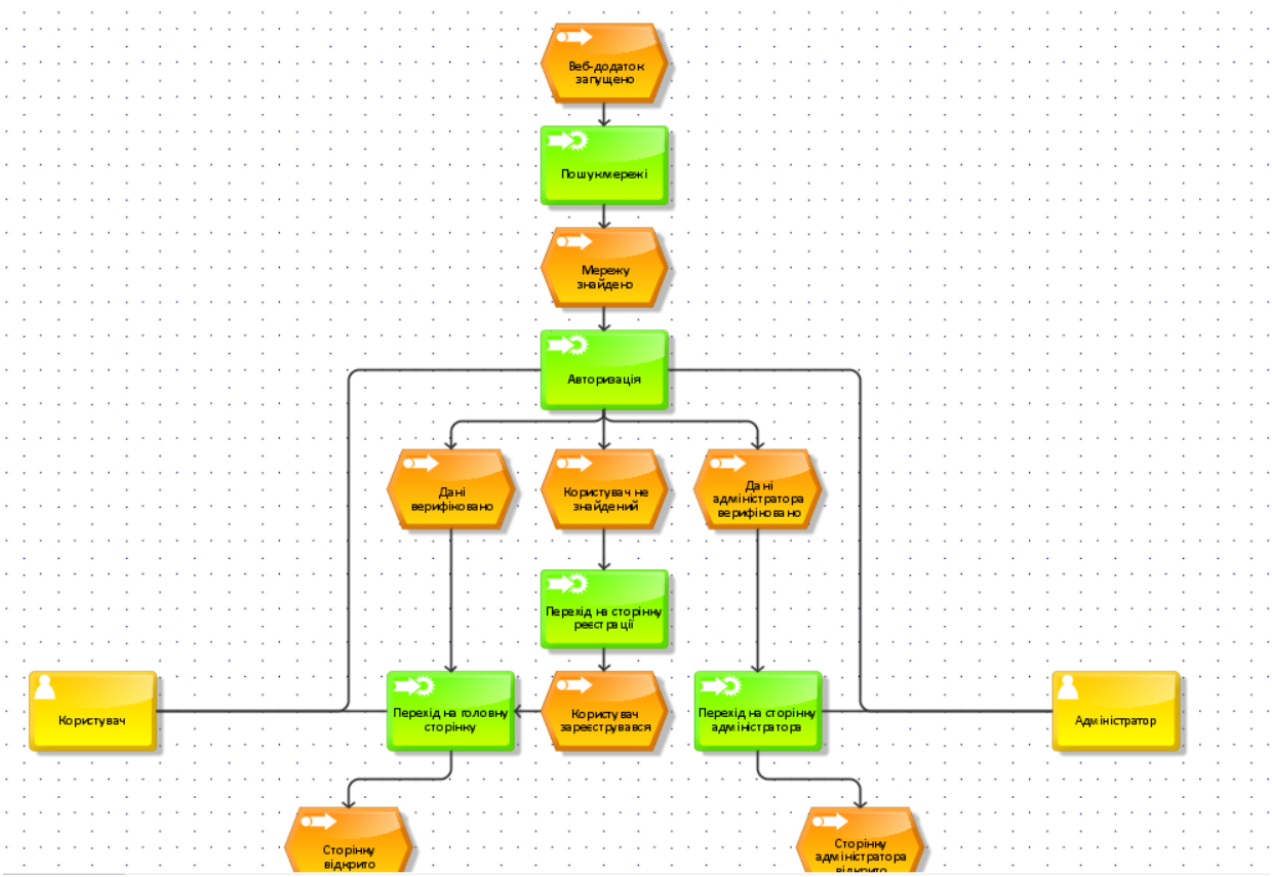


Рисунок 2.5 – Процес роботи СППР при проведенні рейтингового голосування(1)

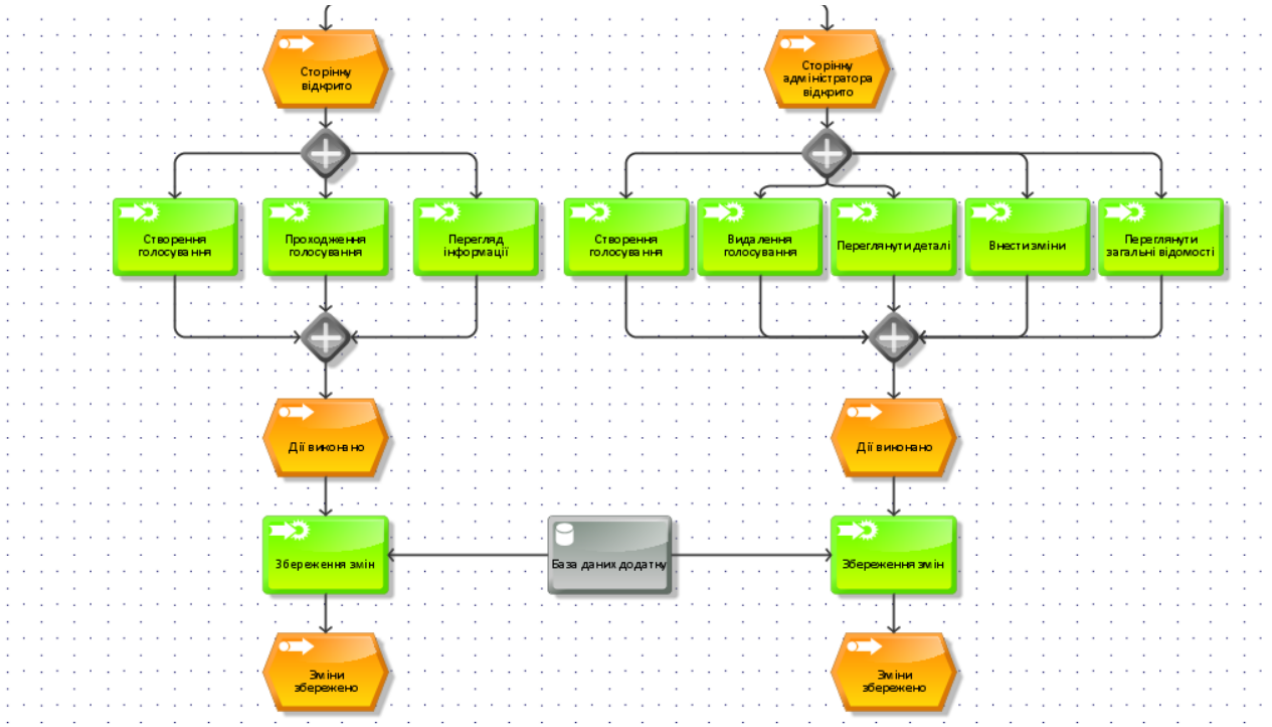


Рисунок 2.6 – Процес роботи СППР при проведенні рейтингового голосування(2)

Розглянемо основний алгоритм рейтингового голосування, його можна зобразити на діаграмі (рис. 2.7)

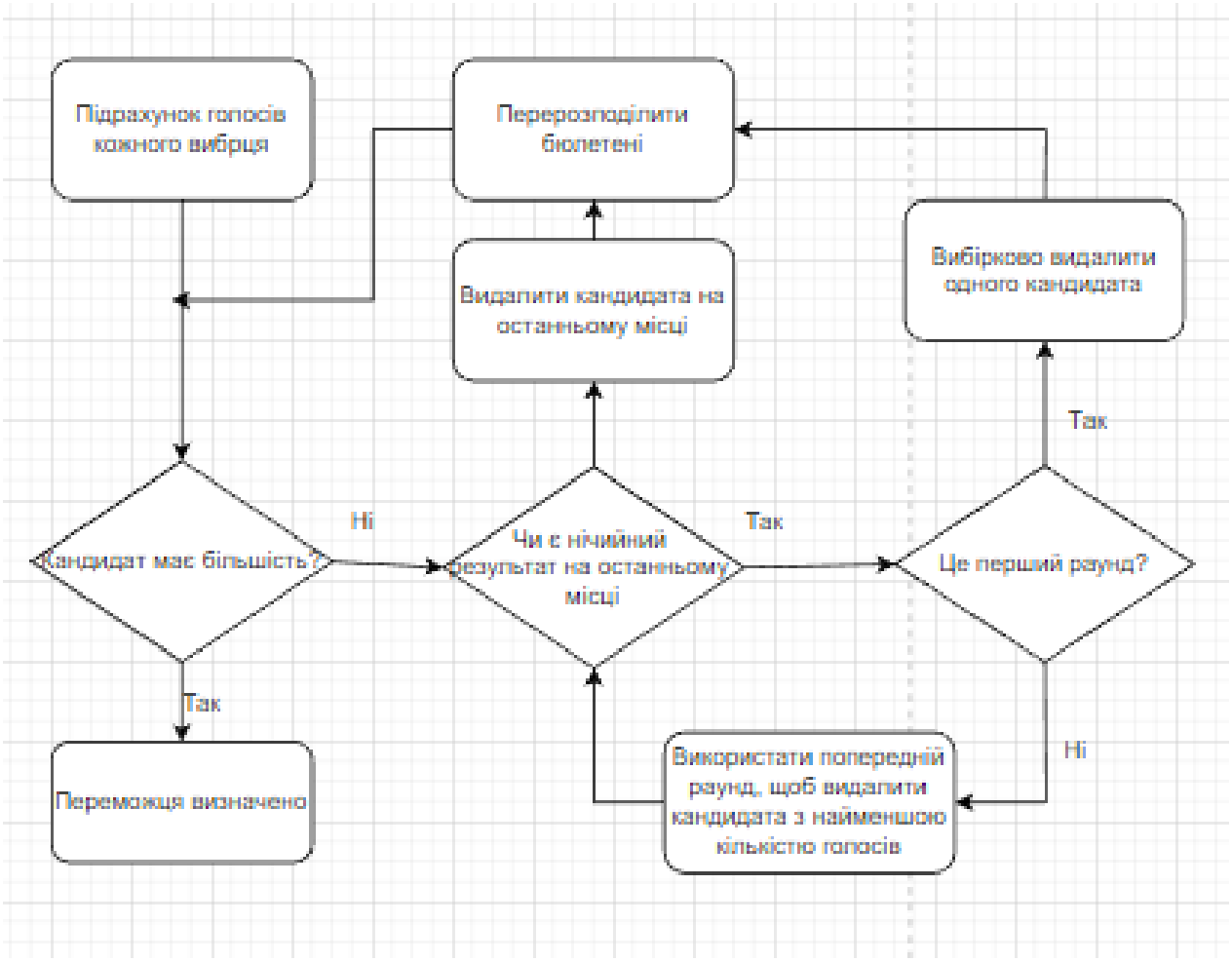


Рисунок 2.7 Алгоритм рейтингового голосування

З алгоритму робимо висновок, що після підрахунку голосів на першому турі алгоритм намагається визначити переможця який має більшість голосів(понад 50%), якщо такий є, то голосування завершується і результати зберігаються, якщо ні, алгоритм визначає кандидата який набрав найменшу кількість голосів, коли таких кандидатів декілька, програма вибірково видаляє одного кандидата, якщо один, програма також видаляє його і відбувається наступний тур, такий алгоритм проходить поки не буде визначено переможця.

Структура проекту

Розвиток технологій і суспільства підштовхує людство до пошуку більш дешевих і надійних методів голосування. Використання технології блокчейн [6] в голосуванні може вирішити більшу частину проблем, які існують в наявних виборчих системах.

В основі технології лежить транзакційна модель [6-7]: кожен користувач має гаманець з унікальним публічним і приватним ключем, якими він підтверджує будь-яку зміну даних. Уся інформація про транзакції зберігається в послідовно записаних блоках, таким чином, хеш даних попереднього блока входить в дані наступного. Так забезпечується незмінність даних – зміна будь-якого блока автоматично зробить невалідними всі наступні. Блокчейн зберігає всю інформацію про транзакції в повному обсязі одночасно на всіх вузлах, і вона не може бути змінена чи видалена.Найбільш широкого застосування блокчейн знайшов саме в сфері реєстрації даних про рух приватних прав на деякі цифрові об єкти – на даній ідеї створено сучасні криптовалюти. Такі об'єкти, що зазвичай називають токеном, можуть або створюватись автоматично, по завчасно створеному алгоритму, або випускатись учасником системи, який має на це права.

Ідея застосування блокчейну в системах голосування напрошувалась сама: блокчейн дозволяє замінити стару технологію шлхом передачі комусь свого голоса, вираженого фізичним об'єктом, на передачу цифрового токена. При цьому, як і в інших випадках переходу бізнес-процесу з фізичного світу в цифровий, різко скорочуються транзакційні затримки і півищується доступність системи. Крім цього, застосування блокчейну дає додаткові переваги:

1)**Непідробність результатів**. Результат голосування, організованого з застосуванням блокчейн технологій, неможливо підробити. Завжди можна сказати, скільки голосів було випущено спочатку голосування, як вони розподілялись по гаманцям і а який час проводились транзакції.

2) **Прозорість процесу**. Блокчейн дає можливість контролю за ходом голосування, так як будь-яка зацікавлена особа може розгорнути вузол з повною копією всіх даних і самостійно проаналізувати їх на рівні блокчейну.

3)**Анонімність**. Кожен учасник голосування має можливість створити пару з публічного і приватного ключа на локальній машині і ніхто крім нього не буде знати, що конкретний гаманець нажежить саме йогму. Таким чином ніхто не може знати, як проголосував даний учасник, за виключенням, коли учасник сам скаже, що гаманець належить йому.

4)**Швидкість обробки даних**. Голосування в рамках міста, регіону, країни чи корпорацій офісами в різних країнах може зіштовхнутись з великими фінансовими проблемами, організаційними питаннями і часовими втратами – як на проведенні голосування, так і на обробку даних. Децентралізація дозволить бачити результати голосування по всій країні вцілому, не дивлячись на те, що кожен регіон/місто/район може експлуатувати свій особистий вузол системи для роподілення нагрузки.

Процес голосування

В моїй системі голосування являється транзакцією. Кожен з варіантів, за яким можна проголосувати, має свій гаманець, на який учасники голосування(кожен з яких також має гаманець) переводять голосувальні токени . Для кожного голосування випускається унікальний токен, таким чином, токеном, випущеним, наприклад, для голосування про улюблений чай, не можна проголосувати за улюбленого футболіста.

Коли користувач хоче прийняти участь в голосуванні відбувається наступне:

1. Він обирає потрібне йому голосування з загального списку і відправляє запит на дозвіл участі в ньому.
2. Система перевіряє чи може користувач прийняти участь в голосуванні і,якщо так, начисляє йому унікальний токен даного голосування.

3. Користувач голосує за один з запропонованих варіантів, після чого система перевіряє, чи може користувач голосувати і, якщо так, голосувальний токен переводиться з його гаманця на гаманець обраного варіанту голосування.

Перевірка результатів голосування можлива декількома способами. По перше, в профілі кожного учасника є опція «My Transactions»: де можна подивитись всі транзакції, починаючи зі створення гаманця. По-друге, в списку голосувань можна перейти на екран конкретного голосування і в деталях голосування знайти транзакцію зі свого гаманця на гаманець обраного варіанту. По-третє, завжди є можливість розгорнути у себе Read only вузол і перевірити перерахунок результатів і роботу системи на рівні блокчейну.

2.4. Розробка моделі бази даних

Для мого застосунку була розроблена база даних, її можна подати у вигляді концептуальної моделі. На концептуальній моделі показано основні зв’язки та опис.

База даних має такі сутності:

- Голосування

- Користувач

- Кандидат

- Гаманець користувача

- Гаманець кандидата

- Токен голосування

База даних містить 6 таблиць які зв’язані за допомогою первинних ключів. Основною є таблиця даних голосування «Voting», вона містить у собі інформацію про назву голосування, назву його токена, їх кількість та кількість

голосуючих та має такі поля: Id, Name, Users, Type, CandidatesId . З нею пов’язано 3 таблиці: таблиця «VotingToken» в якій зберігаються токени створені для голосувань, вона в свою чергу має такі поля: VotingTokenId, VotingTokenName, VotingId, TokenAmount. Таблиця «User» яка містить інформацію про користувача та його баланс на гаманці та має поля Id, FirstName, LastName, Email, Password, EmailStatus. Таблиця «Candidate», що містить інформацію про кандидата та його гаманець, вона має такі поля: CandidateId, CandidateName, CandidateAge, CandidateWalletId, VotingId. Також дві таблиці «Wallet» для гаманця користувача з полями WalletId, WalletSecretWord, UserId, UserName та «CandidateWallet» для гаманця кандидата з полями CandidateWallet, CandidateName, CandidateId.

Проаналізувавши дані можна побудувати таку концептуальну модель бази даних (рис. 2.8).

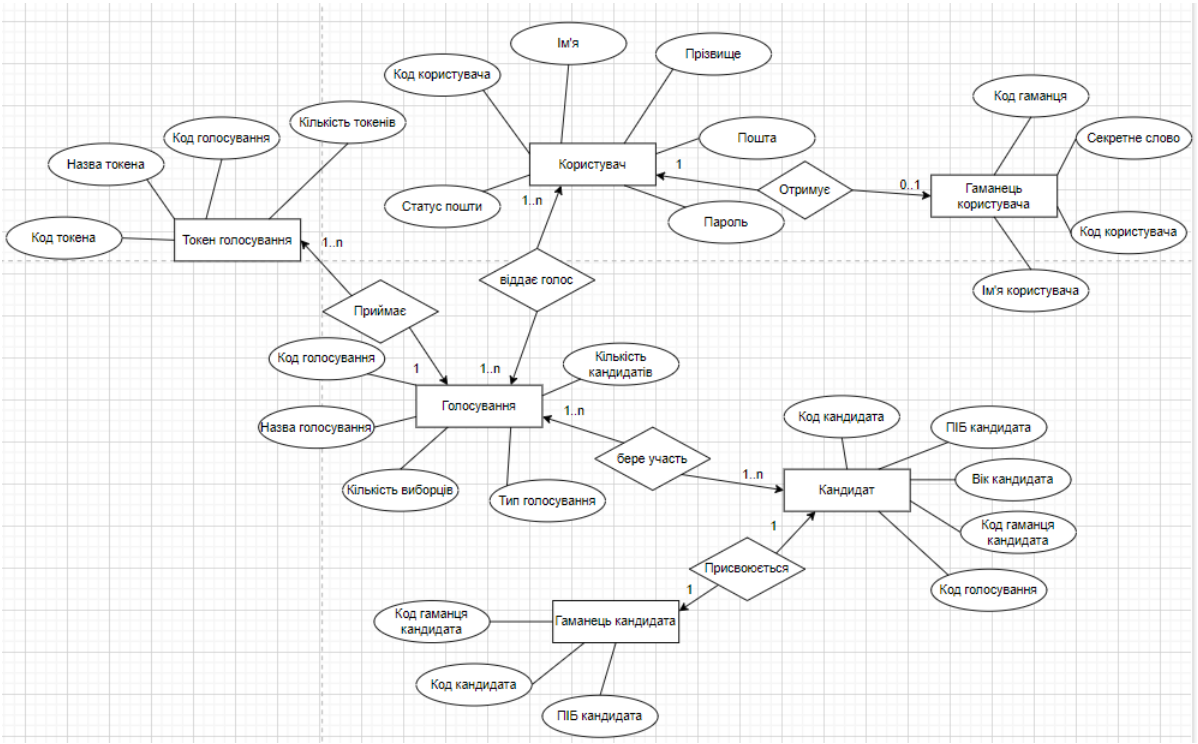


Рисунок 2.8 – Концептуальна модель бази даних

При створенні голосування, в таблиці VotingToken створюється токен створеного голосування, його кількість відповідає тому, скільки буде вказано виборців та кандидатів. Щоб проголосувати, виборець має увійти в свій

особистий кабінет та знайти потрібне голосування, якщо він є серед учасників даного голосування, він може перейти на сторінку голосування та пройти його.

Для створення голосування є таблиця Voting, в неї може вносити зміни тільки адміністратор, перейшовши на відповідну сторінку веб-додатку. Коли всі етапи голосування пройдено, його результат записується в базу даних, та користувачі, що брали в ньому участь можуть переглянути результати.

Логічна модель бази даних матиме вигляд (рис. 2.9) детальнішої інформації по базі даних, вона містить дані про первинні та зовнішні ключі.

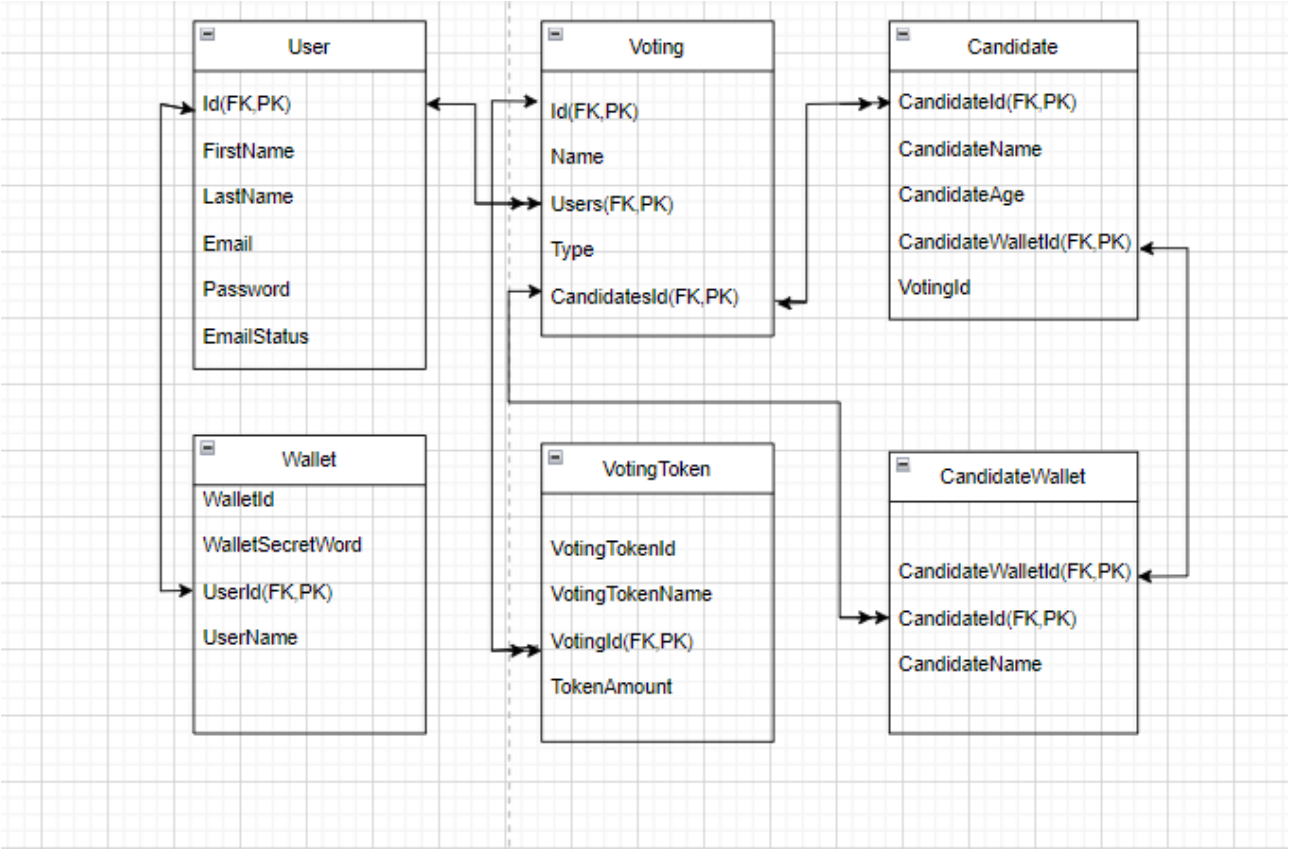


Рисунок 2.9 – Логічна модель бази даних

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1. Вибір інструментарію розробки

Проектування системи здійснювалася використовуючи об'єктно-орієнтовану мову програмування C# в середовищі для розробки Microsoft Visual Studio 2017 [6]. Microsoft Visual Studio - продукт компанії Microsoft, який включає в себе інтегроване середовище розробки ПЗ та багато різних інструментів та засобів. В якості основної було обрано мову C#, так як вона має необхідні параметри для реалізації програмного продукту, має вбудовану підтримку узагальнення, делегати і події, це дуже полегшує реалізацію.

C# можна віднести до сім'ї з C-подібним синтаксисом, вона має постійну типізацію, має підтримку поліморфізму, може перевантажувати операторів (операторів і явного і неявного приведення типу такожі), мова підтимує використання делегатів, атрибутів, подій, властивостей, узагальнених типів і методів, ітераторів, анонімних функцій які підтримують замикання, LINQ, виключеннь, коментарів у форматі XML. Також важливим критерієм для вибору саме цієї мови мови програмування був досвід розробки на ньому, отриманий за час навчання та роботи.

Для розробки проекту однією з основних була використана технологія Entity Framework. Завдяки даній технологію було можливо не виходячи з Visual Studio створити базу даних для проекту. База даних створюється на основі створених моделей, які з використанням інструменту міграцій створюють таблиці. Варіант роботи з базою даних по принципу Code First є набагато надійнішим і швидшим, зменшується ризик допущення помилок при додаванні деталей до бази даних.

Перед .NET 3.5 розробники постійно користувалися ADO.NET для написання коду або блоки доступів до даних компанії для збереження або використання даних застосунку з початкової бази даних. Раніше розробники відкривали зв'язок з базою даних, розробляли DataSet для того, щоб отримати дані або для надсилання даних у базу даних, переробляли дані з DataSet в об'єкти

.NET чи навпаки, для того аби використати бізнес-правила. Це був трудомісткий і небезпечний процес. Корпорація Microsoft створила фреймворк з назвою "Entity Framework", для того аби зробити автоматичними всі дії, які пов'язані з базою даних для розробленої програми.

Entity Framework Core [6]— це платформа ORM з доступним вихідним кодом для додатків .NET, яка підтримується Microsoft. Таким чином, це дає змогу розробнику дозволяє розробникам працювати з інформацією, застосовуючи об’єкти належачих для домену класів, не звертаючи особливу увагу на базові таблиці та стовпці бази даних, де ці дані зюережено. Використовуючи Entity Framework розробник може працювати на вищому рівні абстрактності, коли він використовує дані, а також може створити та підтримувати програми з кількістю коду набагато меншою ніж в традиційних програмах.

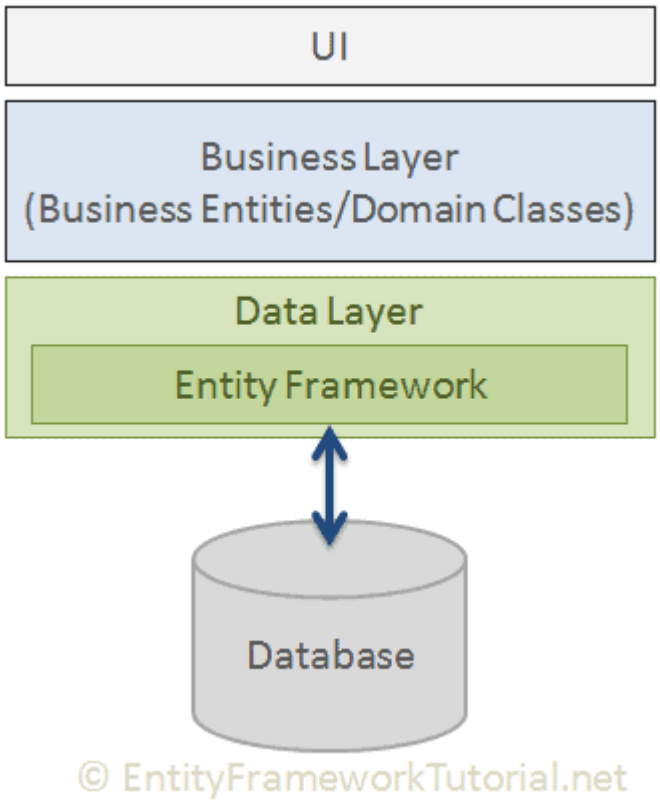


Рисунок 3.1 Принцип Entity Framework

Entity Framework — це об’єктно-реляційний картограф (O/RM), що дає змогу розробнику .NET виконувати роботу з базою даних використовуючи

принципи .NET. Це нівелює потребу в більшості коду доступу до даних, який треба написати розробнику».

Наступний рисунок показує, як Entity Framework працює з вашою програмою.

Основні переваги:

- Кросплатформенність: Entity Framework Core – це кросплатформний пакет, що може працювати на різних ОС.
- Моделювання об’єктів: Entity Framework Core будує EDM (модель даних) базуючись на об’єктах POCO (об’єкт CLR) із технологіями отримання/встановлення різноманітних типів даних. Вона користується цією моделлю при запиті або при збереженні даних сутності в початковій базі даних.
- Запити LINQ: Entity Framework Core дає нам можливість користуватися запитами LINQ (C#/VB.NET) якщо потрібно отримати дані з початкової бази даних. Запити LINQ перекладуться на мову запитів, притаманну для бази даних (наприклад, SQL для реляційної бази даних). Entity Framework Core також дає змогу відправляти необроблені запити SQL прямо до бази даних.
- Стеження за змінами: Entity Framework Core стежить за змінами, які відбуваються в екземплярах сутностей, які треба надсилати до бази даних.
- Збереження даних: Entity Framework Core дає змогу виконувати команди INSERT, UPDATE та DELETE на основі внесених змін, які з’явилися у об’єктах при викликанні SaveChanges(). Entity Framework Core також забезпечить асинхронність при SaveChangesAsync().
- Паралельність: Entity Framework Core користується Optimistic Concurrency по замовчуванню для того, щоб захиститися від перезаписування внесених іншим користувачем змін, після отримання даних з бази даних.

- Керування транзакціями: Entity Framework Core керує транзакціями автоматично,під час відправки запиту або рід час збереження даних. Він також дає змогу налаштувати керування транзакціями.
- Кешування даних: Entity Framework Core включає початковий рівень кешування. Тому, при повторному запиті дані будуть повертатися з кешу замість того, щоб відправляти запит в базу даних.
- Вбудовані правила: Entity Framework Core притримується конвенції про шаблон програмування конфігурацій та містить правила по замовчуванню, вони автоматично конфігурують модель Entity Framework Core.
- Налаштування: Entity Framework Core дає змогу налаштовувати модель Entity Framework Core використовуючи атрибути анотації даних, або Fluent API для зміни необов'язкових умов.
- Міграції до бази даних: Entity Framework Core дає можливість використовувати команди міграцій, це дає змогу виконувати їх використавши консоль менеджера пакетів NuGet або в інтерфейсі командного рядка для створення або редагування бази даних.

3.2. Опис основних класів програми

Головний клас додатки HomeController містить представлення усіх сторінок веб-додатку, крім цього в ньому реалізовано всі головні методи проекту. Кожне голосування при створенні відправляє запит до бази даних саме з цього класу.

Клас головної форми додатка містить елементи інтерфейсу програми, а також реалізує роботу з базою даних та навігацію по сайту. Крім цього даний клас реалізує графічне відображення результатів роботи системи.

Клас CreateVoting визначає створення нового голосування. Елементи даного класу зберігаються в обновлюваному списку створених голосувань, на основі даних якого проводиться відображення інформації на формі.

Клас AddCandidate визначає додавання нового кандидата до голосування. Елементи класу Зберігаються в таблиці кандидатів, та використовуються при голосуванні

Клас AddUser визначає створення нового користувача, та додавання його в базу даних в таблицю користувачів. Користувач авторизувавшись в системі може користуватись додатком.

Клас CreateWallet визначає створення гаманця для користувача. Дані гаманця зберігаються в окремій таблиці в базі даних. Ці дані використовуються для того, щоб користувач мав можливість проголосувати.

3.3. Опис програмного інтерфейсу

При розробці програмного інтерфейсу було реалізовано стартову сторінку, яка містить дві кнопки «Зареєструватися» та «Увійти» залежно від того чи є користувач в базі даних можна обрати один з двох варіантів.

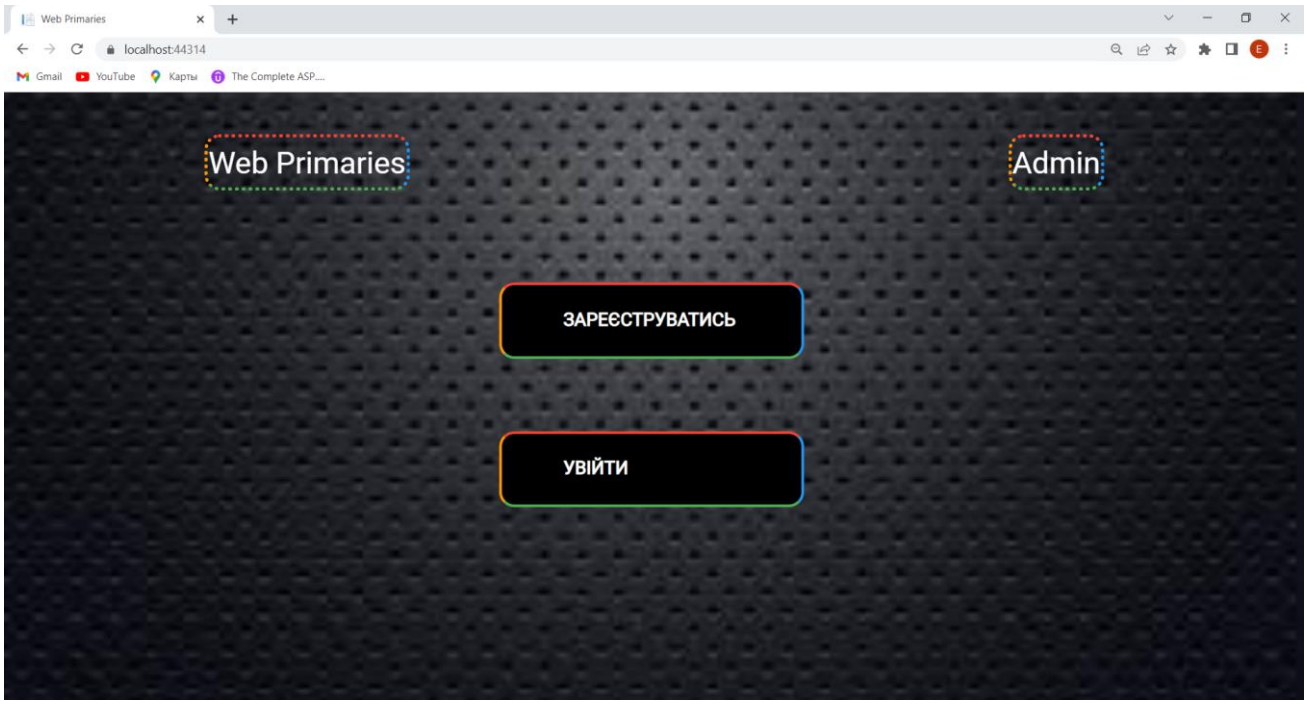


Рисунок 3.2 Стартова сторінка

Якщо користувача немає в базі даних, він переходить по кнопці реєстрації та заповнює необхідні поля, після чого потрібно натиснути на кнопку«Save» і в таблиці «User» з’явиться запис з інформацією про нового користувача

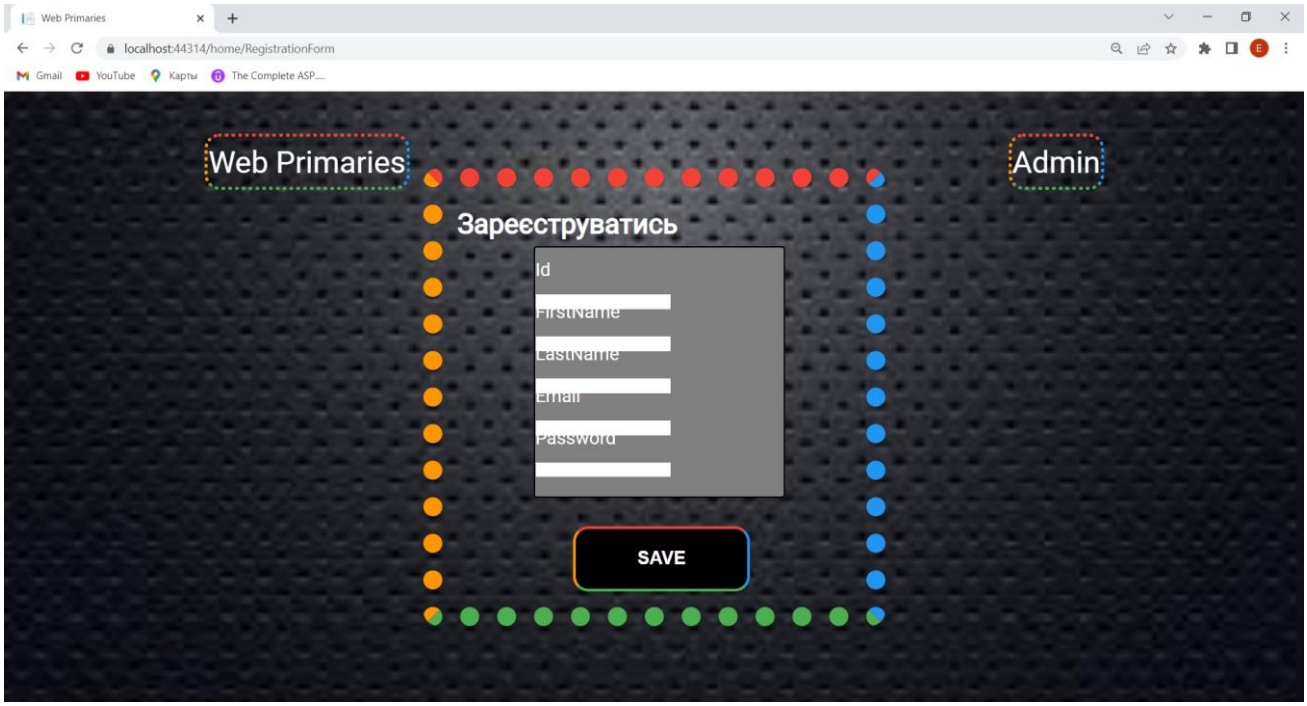


Рисунок 3.3 Форма реєстрації

Якщо ж користувач був зареєстрований раніше, то йому потрібно натиснути на кнопку «Увійти», після цього відкриється форма авторизації. Користувачу потрібно ввести адресу електронної пошти та пароль, які він зазначив при реєстрації. Після того як користувач натисне кнопку «Увійти» введені дані перевіряться на наявність такого запису в таблиці User, якщо такий запис існує то користувач перейде на головну сторінку сайту, якщо ні, то потрібно перевірити правильність введених даних.

Після успішної реєстрації або авторизації, користувач потрапляє на головну сторінку веб-додатку, на ній знаходиться вся інформація про даний сервіс. Сторінка має блок навігації по сайту, блок роботи з базою даних та блоки іформації.

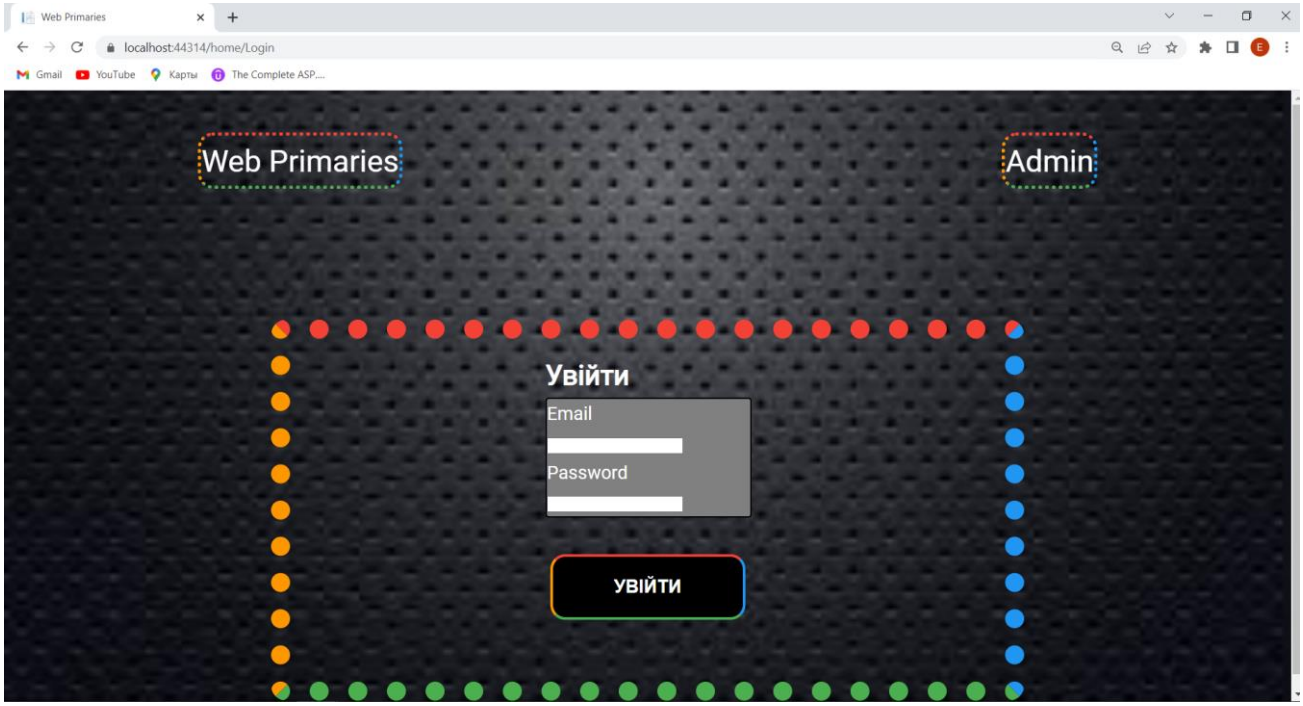


Рисунок 3.4 Форма авторизації

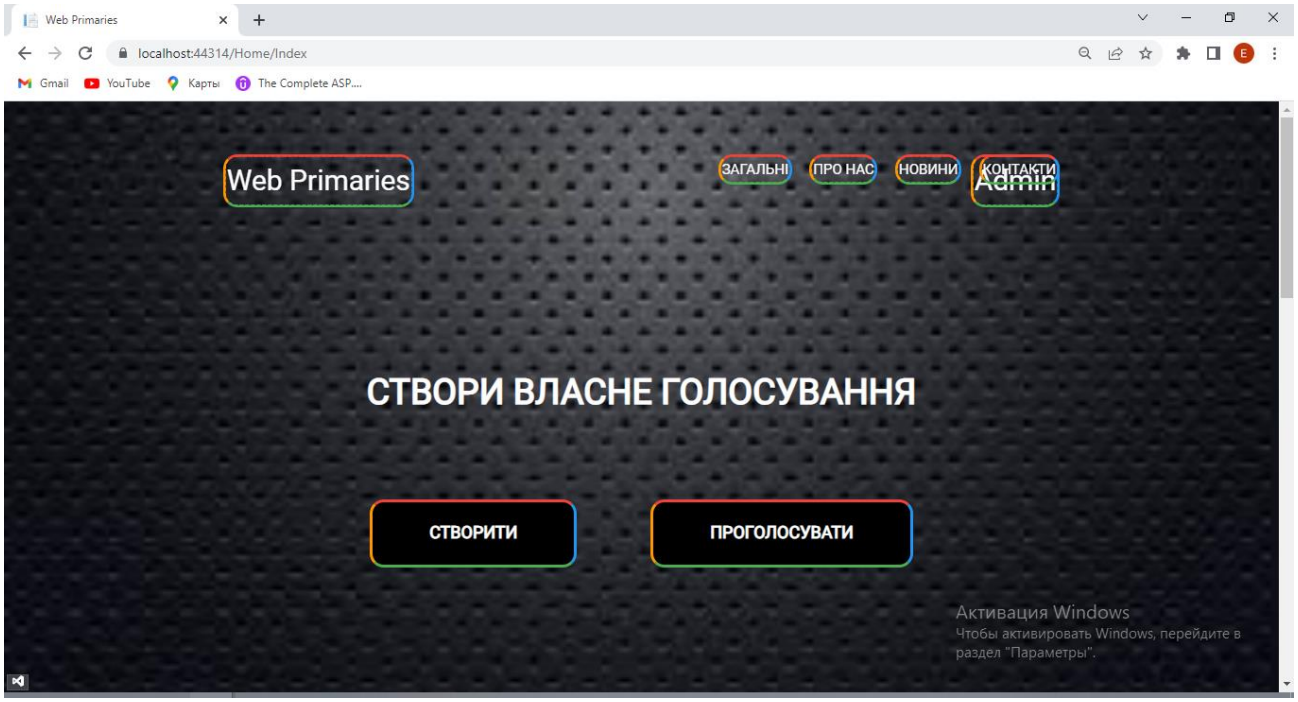


Рисунок 3.5 Головна сторінка

З даної сторінки можна перейти на панель адміністратора за допомогою кнопки «Адмін». В адмін-панелі користувач може отримати інформацію з бази даних, редагувати, додавати та видаляти дані. На головній сторінці додатку також є ще дві кнопки: «Створити» голосування та «Проголосувати». Кнопка

«Створити» переводить користувача на форму для створення нового голосування.

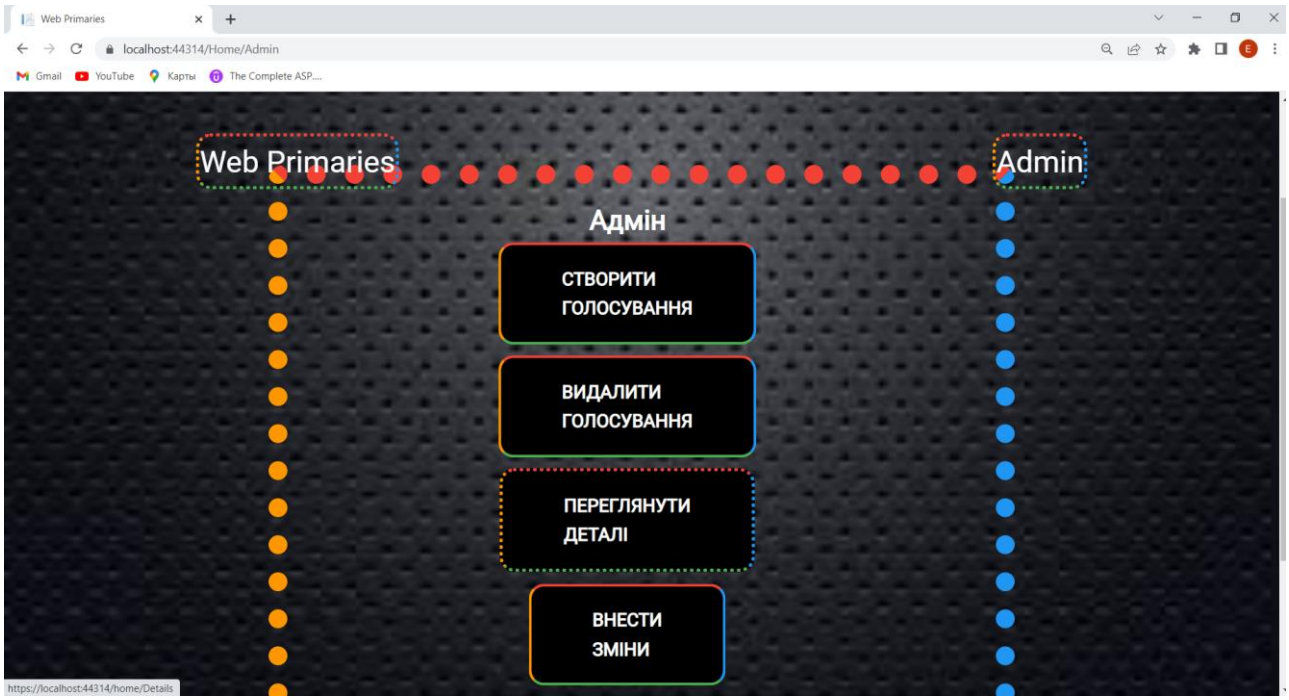


Рисунок 3.6 Панель адміністратора

Користувачу потрібно заповнити необхідні поля та натиснути кнопку «Save», після цього дані будуть записані в таблицю «Voting», а голосування буде створеним.

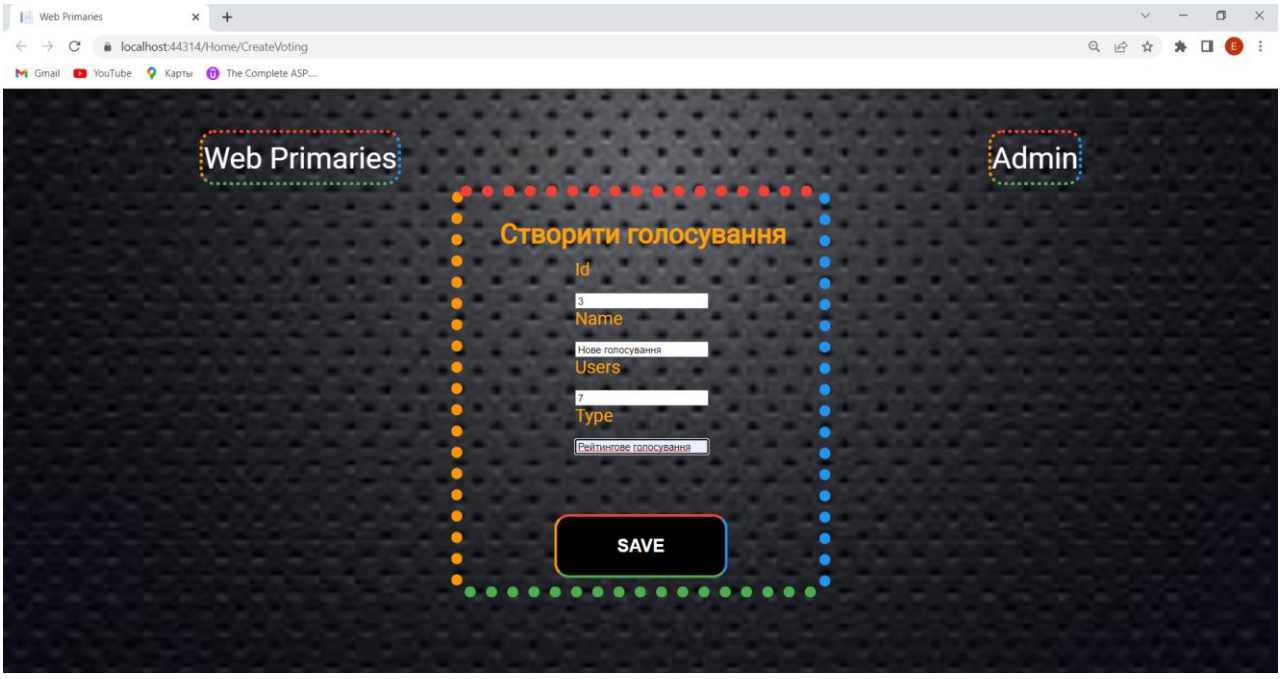


Рисунок 3.7 Форма створення голосування

Кнопка «Проголосувати» переводить користувача на форму для голосування. В ній користувач серед кандидатів обирає трьох найкращих, та віддає 3 голоси кандидату якого він вважає переможцем, 2 голоси кандидату на другому місці і 1 голос кандидату на третьому місці. Після того як усі користувачі проголосують, перший тур голосування буде завершено, і якщо в першому турі переможця не виявлено, кандидат який набрав найменше голосів видаляється з голосування, і проводиться другий тур.

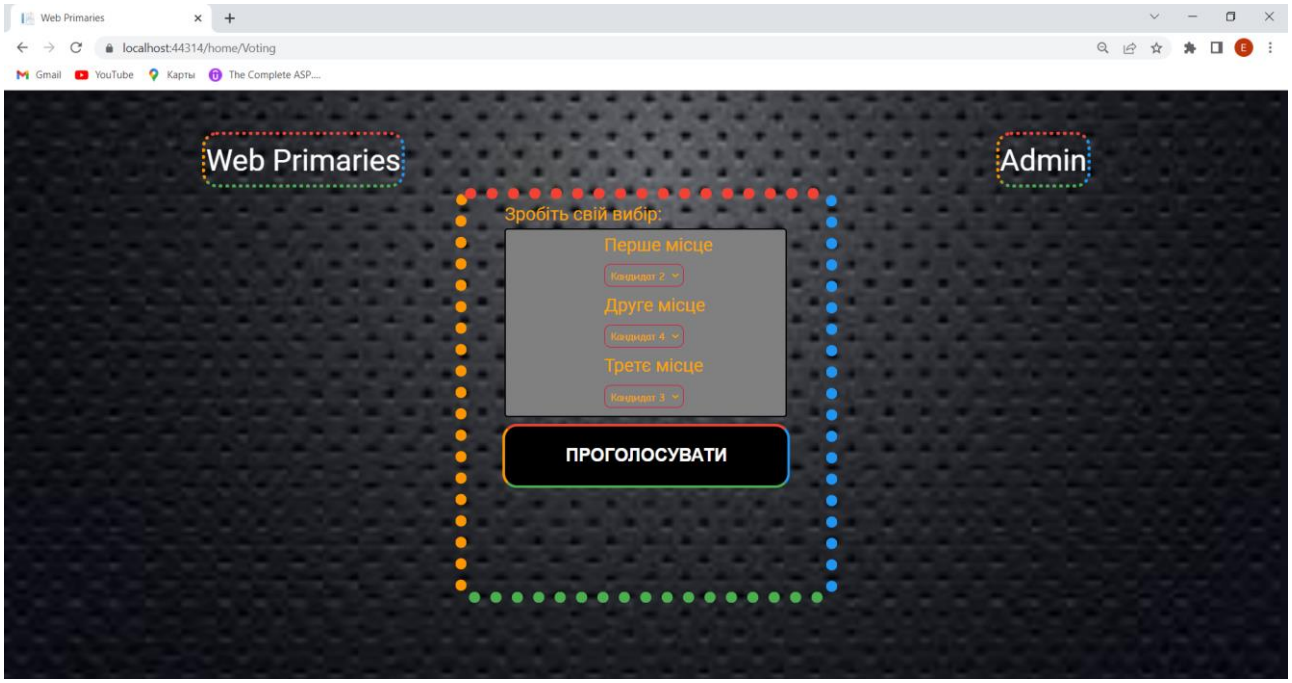


Рисунок 3.8 Форма голосування

Висновки

За результатами випускної кваліфікаційної роботи було розроблено систему підтримки прийняття рішень для проведення рейтингового голосування. Також було розглянуто взаємодію між мовою програмування C# та технологією Entity Framework, також було розроблено інтерфейс взаємодії веб додатку з базою даних.

Програмний модуль було розроблено для подальшої розробки та удосконалення у майбутньому. Майбутні удосконалення включають покращення користувацького інтерфейсу, поліпшення швидкості роботи програми та рефакторинг програмного модуля з потенціалом до майбутнього удосконалення та розширення функціоналу.

Список використаної літератури

1. Гнатієнко Г.М.; Власенко О.М. Математичне забезпечення процедури рейтингового голосування.

2. Гнатієнко Г.М.; Снитюк В.Є. Експертні технології прийняття рішень: Монографія. – К.: ТОВ «Маклаут». -2008. - С.444

3. Петруня Ю.Є. Прийняття управлінських рішень / Петруня Ю.Є.; Говоруха В.Б.; Осадча Н.В.; Літовченко Б.В.; Петруня В.Ю.; Мормуль М.Ф.; Ткачова О.К. // Навчальний посібник. – 2020.-С.124-132

4. Формування експертних оцінок та оцінка узгодженості експертів. - [Електронний ресурс]. – Режим доступу: https://pidru4niki.com/17190512/menedzhment/

5. Розробка системи голосування. - [Електронний ресурс]. – Режим доступу: https://habr.com/ru/post/340342/

6. Рейтингове голосування. - [Електронний ресурс]. – Режим доступу: https://lyubarev.livejournal.com/67928.html

7. Розробка веб-додатку. - [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com

8. How to tabulate ranked choice voting system. - [Електронний ресурс]. – Режим доступу: http://www.csharphelper.com/howtos/howto_ny_rank_voting.html

9. Ranked Choice Voting /Instant Runoff Voting systems. - [Електронний ресурс]. – Режим доступу: https://electionrunner.com/support/kb/ballot/ranked-choice-voting-instant-runoff-voting/

10. Праймеріз це? . - [Електронний ресурс]. – Режим доступу: https://economic-definition.com/

11. Рейтингове голосування. - [Електронний ресурс]. – Режим доступу: https://artsandculture.google.com/entity/

12. Ranked-Choice Voting resource center. - [Електронний ресурс]. – Режим доступу: https://www.rcvresources.org/how-it-works

13. Ranked Voting. - [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Ranked_voting.

14. Ranked Choise Voting system. - [Електронний ресурс]. – Режим доступу: https://ballotpedia.org/Ranked-choice_voting_(RCV)

15. RCV. - [Електронний ресурс]. – Режим доступу: https://www.fairvote.org/how_rcv_works

Додатки

HomeController:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using WebPrimaries.Models.DB;
using WebPrimaries.Models;
using FakeItEasy.Sdk;
using WebPrimaries.ViewModels;

namespace WebPrimaries.Controllers
{
   public class HomeController : Controller
   {
      private readonly WebPrimariesContext _context;
      public HomeController()
      {
         _context = new WebPrimariesContext();
      }
      public ActionResult IndexHome()
      {

         return View();
      }
      public ActionResult Index()
      {

         return View();
      }
      public ActionResult Admin()
      {


         return View();
      }
      public ActionResult Voting()
      {
         var candidates = _context.Candidate.ToList();
         var votings = _context.Voting.ToList();
         var users = _context.User.ToList();
         var viewModel = new VotingViewModel

         {
            Voting = new Voting(),
            Candidates = candidates,
            Votings = votings,
            Users = users
         };
```

```
    return View();
}
public ActionResult RegistrationForm()
{
    var user = new User();
    return View();
}
public ActionResult CreateVoting()
{

    var voting = new Voting();
    return View("CreateVoting", voting);
}
public ActionResult AddCandidate()
{

    var candidate = new Candidate();
    return View("AddCandidate", candidate);
}
public ActionResult CreateWallet()
{
    var wallet = new Wallet();
    return View("CreateWallet", wallet);
}

public ActionResult Create()
{

    var voting = new Voting();
    return View("Create", voting);
}
public ActionResult Delete()
{

    return View();
}
public ActionResult Details()
{

    return View();
}
public ActionResult Login()
{

    return View();
}
public ActionResult Edit()
{

    return View();
}
public ActionResult Index1()
```

```
        {

            return View();
        }
        [HttpPost]
        public ActionResult CreateVoting(Voting voting)
        {
            if (!ModelState.IsValid)
            {

                var viewModel = new VotingViewModel()
                {
                    Voting = voting,
                    Candidates = _context.Candidate.ToList(),
                    Votings = _context.Voting.ToList(),
                    Users = _context.User.ToList()
                };
            }
            if (voting.Id == 0)
                _context.Voting.Add(voting);
            else
            {
                var votingInDb = _context.Voting.Single(v => v.Id == voting.Id);
                votingInDb.Id = voting.Id;
                votingInDb.Name = voting.Name;
                votingInDb.CandidatesId = voting.CandidatesId;
                votingInDb.Users = voting.Users;
                votingInDb.Type = voting.Type;
            }
            _context.SaveChanges();
            return RedirectToAction("Admin", "Home");
        }
        [HttpPost]
        public ActionResult AddCandidate(Candidate candidate)
        {
            if (!ModelState.IsValid)
            {

                var viewModel = new VotingViewModel()
                {
                    Candidates = _context.Candidate.ToList(),
                    Votings = _context.Voting.ToList(),
                    Users = _context.User.ToList()
                };
            }
            if (candidate.CandidateId == 0)
                _context.Candidate.Add(candidate);
            else
            {
                var candidateInDb = _context.Candidate.Single(c => c.CandidateId ==
candidate.CandidateId);
                candidateInDb.CandidateId = candidate.CandidateId;
```

```
            candidateInDb.CandidateName = candidate.CandidateName;
            candidateInDb.CandidateAge = candidate.CandidateAge;
            candidateInDb.CandidateWalletId = candidate.CandidateWalletId;
            candidateInDb.VotingId = candidate.VotingId;
        }
        _context.SaveChanges();
        return RedirectToAction("CreateVoting", "Home");
    }
    [HttpPost]
    public ActionResult CreateWallet(Wallet wallet)
    {
        if (!ModelState.IsValid)
        {

            return View("CreateWallet", wallet);
        }
        if (wallet.WalletId == 0)
            _context.Wallet.Add(wallet);
        else
        {
            var walletInDb = _context.Wallet.Single(w => w.WalletId == wallet.WalletId);
            walletInDb.UserName = wallet.UserName;
            walletInDb.WalletSecretWord = wallet.WalletSecretWord;
        }
        _context.SaveChanges();
        return RedirectToAction("Admin", "Home");
    }
    [HttpPost]
    public ActionResult AddUser(User user)
    {
        if (!ModelState.IsValid)
        {

            return View("RegistrationForm", user);
        }
        if (user.Id == 0)
            _context.User.Add(user);
        else
        {
            var userInDb = _context.User.Single(u => u.Id == user.Id);
            userInDb.Id = user.Id;
            userInDb.FirstName = user.FirstName;
            userInDb.LastName = user.LastName;
            userInDb.Email = user.Email;
            userInDb.Password = user.Password;
        }

        _context.SaveChanges();
        return RedirectToAction("Admin", "Home");
    }
    [HttpPost]
    public ActionResult CheckLogin(User user)
```

```
            {
                var userInDb = _context.User.FirstOrDefault(u=>u.Email==user.Email &&
        u.Password==user.Password);
                if (userInDb == null)
                    return RedirectToAction("Login", "Home");
                return RedirectToAction("Index", "Home");
            }
        }
    }
```

RCVMethod:

```
        var ballots = BallotsToTally.Select(x => x.Entries.OrderBy(y =>
        y.OrderedPreferenceForContestant).Select(y => y.ContestantID).ToArray()).ToArray();
        var active = Enumerable.Range(1, thingsToVoteOn).ToList();
        var top = (IGrouping<int, int>)null;

        while (active.Count() > 0)
        {
            var votes =
                from a in active
                join v in ballots.Select(x => x.Where(y => active.Contains(y)).FirstOrDefault()).Where(x => x
        != 0) on a equals v into gvs
                select new { candidate = a, votes = gvs.Count() };

            var ranks =
                votes
                    .OrderByDescending(x => x.votes)
                    .ThenBy(x => x.candidate)
                    .GroupBy(x => x.votes, x => x.candidate)
                    .ToArray();

            top = ranks.First();

            Console.WriteLine(String.Join(Environment.NewLine, ranks.Select(x => $"{x.Key} votes for
        {String.Join(", ", x)}")));
            Console.WriteLine();

            foreach (var candidate in ranks.Last())
            {
                active.Remove(candidate);
            }
        }

        Console.WriteLine($"Winner(s) {top.Key} votes for {String.Join(", ", top)}");
```

Model Voting:

```
        using System.Web;
        using System.ComponentModel.DataAnnotations;

        namespace WebPrimaries.Models
        {
            public class Wallet
```

```
        {
            [ScaffoldColumn(false)]
            public int WalletId { get; set; }
            [Required]
            public string WalletSecretWord { get; set; }
            public int UserId { get; set; }
            public string UserName { get; set; }
        }
}
```

Model User:

```
        using System;
        using System.Collections.Generic;
        using System.Linq;
        using System.Web;

        namespace WebPrimaries.Models
        {
            using System.ComponentModel.DataAnnotations;

            public class User
            {
                [ScaffoldColumn(false)]
                public int Id { get; set; }
                [Required]
                public string FirstName { get; set; }
                [Required]
                public string LastName { get; set; }
                [Required]
                public string Email { get; set; }
                [Required]
                public string Password { get; set; }
                public decimal EmailStatus { get; set; }
            }
}
```

Model Candidate:

```
        using System.Web;
        using System.ComponentModel.DataAnnotations;

        namespace WebPrimaries.Models
        {
            public class Candidate
            {
                [ScaffoldColumn(false)]
                public int CandidateId { get; set; }
                [Required]
                public string CandidateName { get; set; }
                public int CandidateAge { get; set; }
                public int CandidateWalletId { get; set; }
                public int VotingId { get; set; }
```

```
        }
}
```

Model VotingToken:

```
        using System.Web;
        using System.ComponentModel.DataAnnotations;

        namespace WebPrimaries.Models
        {
            public class VotingToken
            {
                [ScaffoldColumn(false)]
                public int VotingTokenId { get; set; }
                [Required]
                public string VotingTokenName { get; set; }
                public int VotingId { get; set; }
                public int TokenAmount { get; set; }
            }
}
```

Model Wallet:

```
        using System.Web;
        using System.ComponentModel.DataAnnotations;

        namespace WebPrimaries.Models
        {
            public class Wallet
            {
                [ScaffoldColumn(false)]
                public int WalletId { get; set; }
                [Required]
                public string WalletSecretWord { get; set; }
                public int UserId { get; set; }
                public string UserName { get; set; }
            }
}
```

Model VandidateWallet:

```
        using System.Web;
        using System.ComponentModel.DataAnnotations;

        namespace WebPrimaries.Models
        {
            public class CandidateWallet
            {
                [ScaffoldColumn(false)]
                public int CandidateWalletId { get; set; }
                [Required]
                public string CandidateName { get; set; }
                public int CandidateId { get; set; }
            }
```

```
}
```