Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Пояснювальна записка

до магістерської кваліфікаційної роботи
магістр
(освітньо-кваліфікаційний рівень)

на тему: : Розробка методу та засобів побудови системи для проведення електронного голосування

Виконав: студент II курсу групи 2ПІ-18 м
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Коваль С. С.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. ПЗ Черноволик Г. О.
(прізвище та ініціали)

Рецензент: к.т.н., доц. каф. КН Арсенюк І. Р.

ВНТУ – 2019

Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії
Кафедра програмного забезпечення
Освітньо-кваліфікаційний рівень – магістр
Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
Романюк О. Н.
“____”_____2019 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Ковалю Сергію Сергійовичу

1. Тема роботи – Розробка методу та засобів побудови системи для проведення електронного голосування.

Керівник роботи: Черноволик Галина Олександрівна, к.т.н., доцент кафедри ПЗ, затверджені наказом вищого навчального закладу від “__”_____2019 року №___

2. Строк подання студентом роботи
_____

3. Вихідні дані до роботи: Мова програмування JavaScript, фреймворк Express.js.

4. Зміст розрахунково-пояснювальної записки: аналіз предметної галузі та постановка задач розробки; порівняльний аналіз аналогів; аналіз методів розв’язання задач, розробка загальної структури системи, розробка алгоритму роботи основних модулів; програмна реалізація додатку; тестування розробленої системи; економічна частина.

5. Перелік графічного матеріалу:

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	Черноволик Г. О., к.т.н., доцент кафедри ПЗ		
5	Бальзан М.В., к.е.н, доцент кафедри ЕПВМ		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної Роботи	Строк виконання етапів роботи	Примітка
1	Аналіз предметної галузі та постановка задач розробки		Вик.
2	Порівняльний аналіз аналогів		Вик.
3	Аналіз методів розв’язання задач		Вик.
4	Розробка загальної структури системи		Вик.
5	Розробка алгоритму роботи основних модулів		Вик.
6	Програмна реалізація додатку		
7	Тестування розробленої системи		
8	Економічна частина		

Студент _____ **Коваль С. С**

(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи _____ **Черноволик Г. О.**

(підпис) (прізвище та ініціали)

АНОТАЦІЯ

У магістерській кваліфікаційній роботі проведено аналіз методів та засобів побудови системи для проведення електронного голосування. Сформульовано мету досліджень – підвищення надійності захисту анонімності інформації та зменшення можливості фальсифікації.

Проведено аналіз існуючих методів і засобів систем електронного голосування. Удосконалено метод прийняття рішення по результатам голосування та методику авторизації виборця.

Розроблено алгоритми, програмні компоненти та систему на основі запропонованих теоретичних розробок та провести тестування розробленої системи.

Отримані в магістерській кваліфікаційній роботі наукові та практичні результати можна використати для проведення електронного голосування.

ABSTRACT

The master's qualification work analyzes the methods and means of building a system for electronic voting. The purpose of the research is formulated - to increase the reliability of protection of anonymity of information and to reduce the possibility of fraud.

An analysis of the existing methods and means of electronic voting systems. The method of decision-making on the results of voting and the method of voter authorization have been improved.

Algorithms, software components and system have been developed based on the proposed theoretical developments and testing of the developed system.

The scientific and practical results obtained in the master's qualification work can be used for electronic voting.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Особливою формою прояву політичної демократії є вибори, які формують центральні органи влади (президент, віце-президент, парламент) та органи місцевого самоврядування різних рівнів. Вони стимулюють політичну активність населення. Проте на сьогоднішній день є такі категорії населення, яким важко або неможливо прийти на виборчу дільницю, які знаходяться за межами держави тощо. Тому допомогти оптимізувати взаємодію громадян і влади, виборчий процес може е-демократія і, зокрема, е-врядування, електронне голосування.

Питання легітимності будь-якої влади безпосередньо залежить від довіри до виборчої системи держави. Сучасна Україна, протягом багатьох років страждає від низького ступеня довіри до державних інституцій, дуже тонко реагує на будь-які інсинуації, пов’язані з чесністю і прозорістю виборчого процесу. Причиною тому є і деякий історичний досвід, і гострота політичної боротьби. Так чи інакше, з наближенням чергових виборів в країні виникає питання легітимізації їх результатів.

Згідно з виборчим законодавством, ретельно коректованих кожен раз в залежності від змін в системі політичного устрою країни, офіційні результати голосування встановлюються на підставі паперових протоколів виборчих комісій «мокрими» печатками і підписами. Паралельно, починаючи з 2002-року, в тому чи іншому вигляді в Центральній виборчій комісії функціонує Єдина інформаційно-аналітична система «Вибори», покликана оперативно визначати попередні результати голосування. Роль цієї системи важлива, перш за все, в силу швидкості доставки і відображення інформації. Важливо відзначити, що забезпечення безпеки даних в цій системі завжди приділялася підвищена увага. При цьому виникає високий ступінь ризиків, відповідальність розробника і розпорядника системи в разі, якби з її допомогою держава визначала офіційні результати виборів. Проте, завдання організації електронного підрахунку голосів періодично актуалізується з проведенням чергового волевиявлення.

Електронне голосування (e-voting) – концепт, який передбачає об’єднання електронних технологій збору, передачі і підрахунку голосів і є природною складовою системи електронного уряду, широкого поширеною в розвинених країнах і все більше обговорюється в нашій країні.

Важливим елементом проведення електронних виборів також є повсюдне використання апаратного і програмного забезпечення. Відомі неодноразові випадки помилкового пропуску голосів і некоректного їх обліку. Програмне забезпечення обліку і підрахунку голосів може мати як елементарні помилки розробника, так і будь-якого роду незадекларовані можливості.

Розробка системи для проведення електронного голосування є актуальною, оскільки така система дає можливість ознайомитись з необхідною для вибору інформацією, дистанційно та цілком анонімно залишити свій голос за вибраного кандидата.

Зв’язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення

Мета та завдання дослідження. Метою роботи є підвищення надійності захисту анонімності інформації та зменшення можливості фальсифікації за рахунок розробки системи для проведення електронного голосування.

Основними задачами дослідження є:

– провести аналіз існуючих методів і засобів систем електронного голосування;

– удосконалити :

– метод прийняття рішення по результатам голосування;

– методику авторизації виборця;

– розробити алгоритми програмні компоненти та систему на основі запропонованих теоретичних розробок;

– провести тестування розробленої системи.

Об’єкт дослідження – процес проведення електронного голосування.

Предмет дослідження – методи та засоби програмної реалізації систем електронного голосування.

Методи дослідження. У процесі досліджень використовувались: теорія нечітких множин для прийняття рішення у випадку необхідності обробки даних суб'єктивного характеру; технологія блокчейн для підвищення рівню захисту результатів; методи аналізу даних для обробки результатів голосування.

Наукова новизна отриманих результатів.

–Удосконалено метод прийняття рішення по результатам голосування який відрізняється від існуючих застосуванням теорії нечітких множин що дозволяє врахувати дані суб'єктивного характеру.

– Удосконалено методику авторизації виборця, яка відрізняється від існуючих використанням технології блокчейн та дозволяє підвищити рівень захисту системи.

Практична цінність отриманих результатів. Практична цінність одержаних результатів полягає в тому, що на основі отриманих в магістерській кваліфікаційній роботі теоретичних положень запропоновано алгоритми та розроблено програмні засоби системи електронного голосування.

Особистий внесок здобувача. Усі наукові результати, викладені у магістерській кваліфікаційній роботі, отримані автором особисто.

Апробація матеріалів. Основні положення магістерської кваліфікаційної роботи доповідалися та обговорювалися на XII міжнародній науково-практичній конференції «Інформаційні технології і автоматизація –2019» (Одеса, 2019).

Результати дослідження подані до колективної монографії «Інформаційні технології та автоматизація» ОНАПТ, Одеса, 2019 р.

Публікації. Основні результати досліджень опубліковано в 2 наукових працях, у тому числі 1 – в матеріалах конференції й 1 – в монографічному вигляді.

Структура та обсяг роботи. Магістерська кваліфікаційна роботи складається зі вступу, п'яти розділів, висновків, списку використаних джерел, що містить 25 найменувань, додатків. Робота містить 15 ілюстрацій, 11 таблиць.

1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Аналіз стану проблеми.

З огляду на активний розвиток інформаційних технологій, велика кількість розвинених демократичних країн переходить до електронної політичної інфраструктури та інструментів електронної демократії, зокрема проводить електронні вибори та голосування. Електронне голосування – це безпечне, таємне голосування за допомогою інформаційно-комунікаційних технологій, результати якого передаються через мережу Інтернет. Електронні вибори, внаслідок своєї простоти та зручності у використанні, збільшують явку виборців, сприяють активному голосуванню молоді, ширшому представництву нацменшин (репрезентативність). Процес електронного голосування дозволяє практично усунути людський фактор, у наслідок чого результати стають максимально точними та надійними. На сьогодні інструмент електронних виборів широко використовується в США, Канаді, Австралії, Великобританії, Німеччині, Франції, Іспанії, Португалії, Італії, Норвегії, Швейцарії, Бельгії та Естонії.

Для української виборчої системи "електронне голосування" абсолютно нове, незвідане явище, тому необхідно зрозуміти його сутнісні характеристики. Електронне голосування охоплює кілька різних типів голосування. При цьому воно охоплює як сам процес голосування за допомогою електронних засобів, так і процес автоматичного підрахунку голосів за допомогою електронних пристроїв та спеціального програмного забезпечення. Електронні технології голосування – це перфокарти та системи оптичного сканування бюлетенів, "кіоски голосування". Виборчі бюлетені можуть передаватися за допомогою ізольованих комп'ютерних мереж, через Інтернет чи за допомогою телефонів.

Найбільше розповсюджене – Інтернет-голосування. Виборець (респондент) одержує на відповідному сайті електронний бюлетень (форму для

голосування з варіантами можливих відповідей) і голосує, позначаючи варіант відповіді. Підтвердження здійснення голосування за допомогою електронного бюлетеня здійснюється за рахунок, наприклад, приватних реєстраційних даних користувача. Ними можуть бути: ідентифікаційна мережева адреса та параметри пристрою і програмного забезпечення для голосування, приватний цифровий підпис, відбитки пальців.

Сьогодні хід і результати будь-яких виборів: від місцевих до виборів парламенту і президента не викликають довіри ні в Україні, ні за її межами, незважаючи на численних спостерігачів. Офіційні результати виборів не викликають ніякої довіри виборців, які все в більшій кількості не йдуть на виборчі дільниці [1]. Головними проблемами виборів в Україні є: низька явка населення на вибори; повільний підрахунок голосів та маніпулювання результатами; неможливість проведення голосування в містах, де відбуваються збройні конфлікти; на вибори йде багато державних коштів. Вирішити ці проблеми можна за допомогою електронних виборів. Завдяки такому голосуванню знизиться рівень фальсифікації, корупції під час виборів. Кожен громадянин матиме змогу самостійно простежити чи дійшов їхній голос до виборця.

В епоху формування ІТ-права та електронної демократії можливість голосування зі смартфона виглядає привабливою. Електронне голосування можна впроваджувати за умови високого рівня кіберграмотності населення та наявності у нього критичного мислення, яке дозволяє розмежувати якісні новини від фейкових. Останні президентські та парламентські вибори засвідчили високий ступінь маніпуляції виборцями через електронний протокол, обробку цільової аудиторії з метою маніпуляцій тощо. Електронне голосування привабливе нібито простотою, оскільки можна голосувати через електронну пошту або акаунт із соціальної мережі. Однак тут питання полягає у доступності. Для електронного голосування необхідний захист персональних даних громадян. Сьогодні він на низькому рівні. Хоча електронний протокол чітко відображає стороннє втручання в базу даних, але без належного захисту персональних даних

громадян електронне голосування буде під загрозою зовнішнього втручання, маніпулюванням метаданими виборців, що може суттєво спотворити результати виборів. Також канали зв'язку повинні бути якісно захищеними.

На нинішньому етапі виділяють позитивні та негативні аспекти цього впровадження.

Щодо позитивних чинників, то запровадження електронного голосування значно зменшить видатки державного бюджету, пов'язані з проведенням виборчого процесу. Щоправда, тоді доведеться інвестувати у створення та постійне оновлення програмного забезпечення. Також має мінімізуватися кількість помилок у процесі проведення виборів, які мають суто людський фактор.

Загалом, запровадження змін, що дозволять використання сучасних технологій, нададуть громадянам можливість зекономити час та виконати свій громадянський обов'язок, перебуваючи в будь-якому куточку світу та знаючи, що їхній голос буде врахований та важливий. В результаті якісного та обдуманого запровадження цієї системи відбудеться скорочення зловживань (підробка протоколів, підкуп тощо).

З приводу негативних чинників, то є сумніви щодо реальності запровадження електронного голосування найближчим часом. Питання полягає в багатьох технічних моментах, які виникатимуть, навіть якщо дивитися крізь призму кібербезпеки. Окрім того, зараз в Україні є чимало людей похилого віку, які далеко не завжди вміють користуватися сучасними технологіями. Однак небезпека зовнішніх втручань у такі вибори стала однією з причин, що змусила деякі країни повернутися до звичайного голосування за допомогою паперових бюлетенів.

Потрібно враховувати інтереси всіх верств населення, Можливо, не варто одразу виключати можливість паперового голосування, а спробувати поєднання двох видів (електронного і паперового) та впроваджувати зміни поетапно.

Існують різні типи електронного голосування. Насамперед, це може бути голосування як підтримка чи думка. Наприклад, вже зараз є голосування за

громадський бюджет у різних містах. Також можна назвати голосуванням, приміром, підписання петицій. Однак зовсім інші питання — це електронні вибори, тобто голосування за різні рівні влади: за міську владу, Верховну раду чи Президента. Адже це потребує зовсім іншого рівня безпеки ідентифікації та надійності.

Вже добре працюють і ще можна масштабувати та популяризувати такі формати голосування як опитування чи висловлення підтримки або не підтримки. Однак на рівні виборів ще не створений фундамент та немає впевненості в тому, що ці голоси не будуть передані та продані (якщо це запускати реально сьогодні).

Перевага електронного голосування на виборах – це можливість проголосувати для людей, які перебувають за межами України чи не можуть прийти на дільницю. Зазвичай це молодь, яка не встигає в неділю зайти на дільницю, а їхній голос дуже важливий.

Естонія перша держава у світі, де з 2005 р. проводяться електронні вибори. Згідно зі статистикою, участь у виборах там більша ніж в середньому в Європі, тому що громадяни, які перебувають поза межами держави чи не мають можливості прийти на вибори, можуть брати в них участь.

Однак дуже важливо врахувати, що для повноцінного електронного голосування на виборах має бути створений дуже потужний фундамент і побудована довіра до держави. Тобто має бути впевненість, що існує достатньо надійний інструмент електронної ідентифікації (наприклад, MobileID чи ID-картка, яку неможливо передати іншій людині). Це повний кіберзахист всіх інформаційних систем і баз даних. Також потрібні якісні реєстри виборців та мешканців, у тому числі демографічний реєстр, про створення якого вже давно йде мова.

До того ж ця система має бути побудована таким чином, щоб не було осіб, які мають доступ до результатів голосування або можуть будь-яким чином контролювати чи впливати на вхід.

Отже, електронне голосування на виборах є дуже делікатною темою, до якої потрібно дуже ретельно підходити та продумати всі можливі кроки. Довіра до виборів – це одна з фундаментальних засад. Однак, можливо, вже потрібно поширювати та популяризувати участь в опитуваннях і заохочувати висловлення своєї думки, таким чином поступово будуючи довіру та звички.

Тільки за умов дотримання всіх перерахованих вище вимог та рекомендацій можна досягти наступних переваг:

– можливість голосувати дистанційно, значно знижує ризик тиску голосувати та підвищує надійність вибору;
– можливість голосування протягом кількох днів у зручний для виборця час;
– зростання кількості молодих виборців, які готові брати участь у виборах з використанням інформаційних технологій;
– процедура голосування є більш зручною для людей з обмеженими фізичними можливостями;
– підрахунок голосів стає більш точним і швидким;
– за підрахунком голосів можна спостерігати в режимі реального часу.

1.2 Аналіз існуючих систем електронного голосування

У світовій практиці виділяють такі типи електронного голосування.

– Голосування на виборчій дільниці за допомогою електронної системи (Vote-recording Technologies) – виборець реєструє свою ID-картку в спеціальному зчитувачі, вводить свій пароль на сайті голосування, де отримує електронний бюлетень та відбиває свій вибір. Після завершення голосування такі електронні скриньки автоматично підраховують голоси.
– Оптичне сканування (Optical Scan Marksense) – виборець обирає кандидата шляхом відмічання на спеціальному бюлетені, який потім обробляє виборча машина, що за допомогою оптичних засобів підраховує голоси на дільниці.
– Голосування за допомогою перфорованих карт (Punched Card) – виборець використовує спеціальні картки, що зчитуються комп’ютером, відмічаючи

кандидата спеціальним кодом, який залишається на перфокарті; далі виборець опускає перфокарту до виборчої скриньки, яка автоматично здійснює підрахунок.

– Електронна виборча система прямого запису (Direct-recording Electronic Voting System – DRE) – виборець обирає кандидата на сенсорному екрані комп'ютера, після чого машина, за допомогою спеціальної програми підраховує голоси. Ідентифікація відбувається через відбитки пальців або ID-картку.

– Дистанційне голосування – виборець обирає кандидата через захищений комунікаційний канал (програмне забезпечення в Інтернет-мережі). Найчастіше це надсилання спеціального електронного листа на виборчу дільницю або голосування на спеціально створеному сайті. Процедура зарахування голосу відбувається лише після попередньої ідентифікації.

Системи електронного голосування можна поділити на два типи: ті, які потребують безпосередньої наявності виборця на виборчій дільниці та ті, які дозволяють проголосувати дистанційно [4]. Прикладами систем, які потребують наявності виборця на дільниці, є КОВБ-2003 та КОВБ-2010, які дозволяють автоматизувати процес підрахунку бюлетенів.

Такі системи як Hart eSlate DRE, UE 2000, ДАС "Вибори" відносяться до систем прямого запису. Вони зчитують голос виборця за допомогою електронно-оптичних або механічних компонентів та одразу записують голос виборця на електронний носій, завдяки чому забезпечують вищий рівень автоматизації виборчого процесу у порівнянні з системами КОВБ-2003 та КОВБ-2010.

Перша система електронних виборів з можливістю дистанційного голосування була застосована 16 жовтня 2005 року на муніципальних виборах в Естонії. Для того щоб проголосувати на виборах у Естонії через мережу Інтернет, у виборця має бути ID-паспорт громадянина Естонії, комп'ютер з підключенням до мережі Інтернет та прилад для зчитування інформації з ID-паспорту.

Окрім Естонії, досвід з проведення Інтернет-виборів різних рівнів (від місцевих до парламентських) мають Великобританія, Сполучені Штати Америки та Росія.

Розглянемо систему для електронного голосування «Єдність» – (рис. 1.1).

У цій системі є чотири головні компоненти: планшет (робоче місце), комп’ютер (який є сервером, де встановлена програма), телевізор (куди виводяться результати голосування) і є точка WI-FI, що представляє собою закриту мережу без доступу в Інтернет . Третя особа не зможе отримати доступ, нашкодити системі або змінити дані голосування.

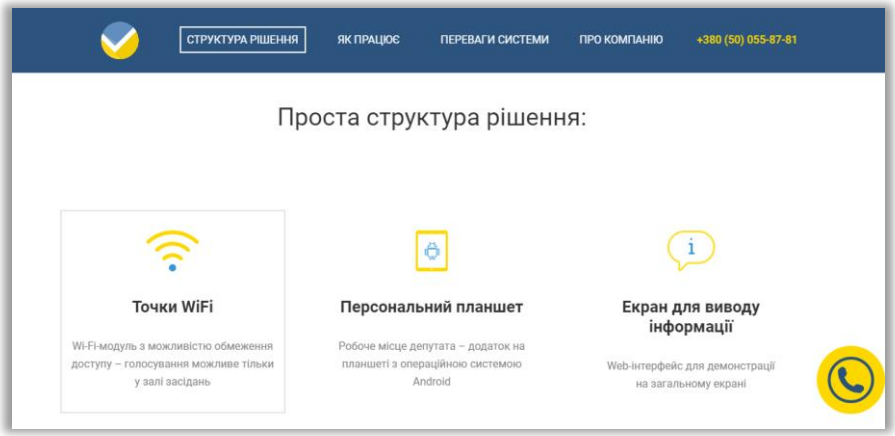


Рисунок 1.1 – Система електронного голосування «Єдність»

Система «ЕЛЕКТРОННЕ ГОЛОСУВАННЯ» призначена для автоматизованого проведення виборів, референдумів, опитувань, рейтингових виборів (primaries) і висунення кандидатів.

Технологія включає в себе наступні автоматизовані процедури:

– актуалізація електронних списків виборців по дільницях, друк індивідуальних запрошень з маркуванням у вигляді штрих-кодів (ШК) та інформаційних листків для комплектування планшетів (буклетів);

– контроль і реєстрація за разовими запрошеннями або постійним ID-картками з'явилися на ділянку виборців;

– фіксація в незалежній пам'яті кодів кандидатів, партій, відповідей на питання, які виділяються (вказуються) виборцями за допомогою портативного терміналу («електронної указки-бюлетеня»);

– автоматичний підрахунок розподілу голосів і оголошення результатів з використанням цифрового табло і синтезатора мови;

– автоматична принтерна друк зашифрованих протоколів голосування і оприлюднюються на ділянці контрольних протоколів;

– мережева передача «електронних протоколів» на сервер проводить і / або контролюючої організації;

– підведення підсумків організацією, яка проводить захід.

– Система ГОЛОС – це апаратно-інформаційний комплекс, котрий забезпечує можливість автоматизації повного циклу роботи ради в процесі підготовки та проведення засідань, публікації результатів та аналізу роботи депутатів і ради.

– Система електронного поіменного голосування ГОЛОС (рис.1.2) побудована за модульним принципом, тому є надзвичайно адаптивною та може налаштовуватись під індивідуальні потреби ради. Вона може працювати, як на основі бездротової мережі так і у дротовій конфігурації.

Рисунок 1.2 – система електронного голосування «Голос»

Виборці отримують повний доступ до баз даних проектів та результатів голосувань на своїх пристроях. В ході голосування виборцям та керівництву асистують модератор(и) системи. На центральне табло система виводить поточну інформацію в часі засідання.

1.3 Аналіз вимог до систем електронного голосування

Експерти виділяють наступні критерії електронного голосування:

- доступність (простота у використанні та наявність необхідних інструкцій);
- точність (голос має бути переданий у центральну мережу, при цьому він не може бути видалений або змінений);
- демократичність (відсутність дискримінації за статтю, расою, матеріальним положенням, фізичними можливостями тощо);
- таємність (ні система, ні уповноважені особи не мають можливості пов’язати голос з конкретною людиною);
- верифікація (можливість відкритої перевірки правильного підрахунку).

Рекомендується використовувати систему електронного голосування лише за низки умов:

- запроваджена електронна система має бути безпечною та надійною;
- виборча система електронного голосування є прозорою та готовою до постійної перевірки її функціонування;
- громадяни мають володіти можливістю перевірки свого вибору та виправлення помилки, якщо така сталася.

Вибори здійснюються шляхом таємного голосування, тому перш за все система електронного голосування має забезпечувати збереження таємниці голосування. Не менш важливими вимогами, яким має задовольняти система електронного голосування, є неможливість змінити голоси виборців та провести вкидання бюлетенів. Для того щоб забезпечити високий рівень довіри виборців до нової системи електронного голосування, в ній повинен бути реалізований прозорий алгоритм підрахунку голосів та можливість перевірки виборцем стану свого голосу, а саме того, що голос був записаний у базу даних та не був змінений. Вище перелічені вимоги перш за все впливають на архітектурні рішення, які будуть застосовуватися у системі електронного голосування:

механізм автентифікації виборця, отримання цифрового бюлетеня та збереження голосів виборців. Але окрім відповідності вище перерахованим вимогам, система має дозволяти виборцям проголосувати дистанційно за допомогою персонального пристрою, оскільки це дозволить залучити до участі в виборах більшу кількість виборців і, як наслідок, отримати більш об’єктивні результати голосування. Вихідний код системи електронного голосування має знаходитися у відкритому доступі.

1.4 Аналіз методів та алгоритмів систем електронно голосування

У типовому сценарії процедури голосування виділяють п’ять основних етапів. Загальна схема цього процесу зображена на рисунку 1.3.

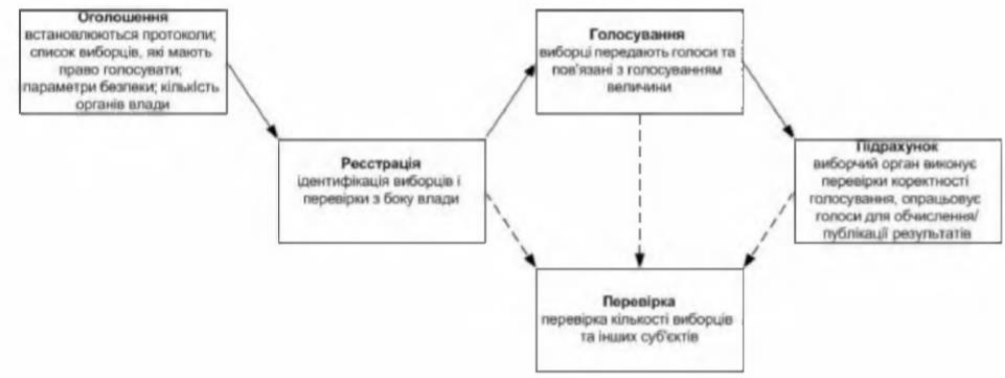


Рисунок 1.3 – Схема процедури голосування

На етапі налаштування (Set-up phase) встановлюються параметри голосування. До основних параметрів належать критерії прийняття кандидатів, виборців та органів влади; процедура голосування; критерії дійсності виборчого бюлетеня; правила підрахунку голосів (підбиття підсумків голосування). Кандидати реєструють себе в відповідальних органах, після чого чинні параметри голосування, кандидати та органи влади розголошуються.

На етапі реєстрації (Registration phase) виборцям необхідно зареєструватись в визначених органах реєстрації. Критерії прийняття визначаються з попереднього пункту. Особам, що не відповідають цим критеріям, заборонено брати участь у голосуванні. Список підтверджених виборців розголошується для забезпечення публічної верифікації. На етапі голосування (Voting phase) зареєстрованим виборцям дозволяється брати участь у голосуванні, дотримуючись наступних правил:

– Кожен виборець зобов'язаний бути автентифікованим відповідно до списку зареєстрованих виборців з етапу реєстрації. Особам, не зазначеним у цьому списку, заборонено брати участь у голосуванні.

– Кожен з автентифікованих виборців отримує порожній бюлетень і фіксує свій голос в бюлетені всередині фізично приватного та безпечного місця для уникнення примусу.

– Виборчий бюлетень повинен бути анонімним - відповідність виборця та його голосу засекречена.

На етапі обрахунку голосів (Tally phase) всі бюлетені оброблюються для визначення результатів голосування. Етап складається зі збору голосів, верифікації бюлетенів (відповідно до правил, встановлених на етапі налаштування), обрахунку голосів та оприлюднення отриманих результатів.

Окремим особливим етапом процедури голосування є перевірка (Check). Цей етап може бути використаний декілька разів після етапів реєстрації, голосування або підрахунку для здійснення перевірки необхідних параметрів відповідно до обраної схеми голосування.

Учасниками протоколу електронних виборів традиційно є виборці та органи влади. Усі учасники можуть контактувати один з одним за допомогою каналів зв'язку. Дії виборця під час електронного голосування повинні бути зведені до мінімуму, а протокол голосування повинен передбачати, що виборець володіє обмеженими часовими та обрахунковими потужностями.

Органи влади керують процесом вибрів, відповідно, потенційно володіють значними обчислювальними потужностями і здатні зберігати значну кількість

інформації. Кількість органів влади в різних протоколах може відрізнятись, однак зазвичай автори протоколів відмовляються від виділення єдиного такого органу, через ризикованість перенесення всієї відповідальності на єдиний центр. Протоколи електронного голосування також повинні враховувати можливу корумпованість певної кількості владних органів, кожен учасник повинен розраховувати, що принаймні певна кількість з них є гарантовано чесними. Структура голосу залежить від типу голосування, а точніше від питання, поставленого виборцям, та можливих варіантів відповіді на нього. Найбільш розповсюдженими є такі види голосів: так/ні (yes/no voting), вибір одного з L варіантів (1-out-of-L voting), вибір K з L варіантів (K-out-of-L voting), власний варіант (write-in voting).

1.5 Висновки

В першому розділі було здійснено аналіз стану проблеми, існуючих систем електронного голосування, аналіз вимог до систем електронного голосування та проаналізовано методи та алгоритми систем електронно голосування.

2. РОЗРОБКА СТРУКТУРИ ТА АЛГОРИТМІВ ОСНОВНИХ МОДУЛІВ СИСТЕМИ

2.1 Розробка загальної структури системи електронного голосування

Розглянемо саме поняття "електронне голосування". Отже, поняття "Електронне голосування", безпосередньо пов'язане із суміжними поняттями "Електронна демократія" і "Електронний Уряд" і тому потрібно зупинитися на них більш докладно. Так, поняття "Електронний Уряд" дослідник Шолль визначає, як "використання інформаційних технологій для підтримки діяльності органів державного управління, залучення до участі громадян та забезпечення виконання державних зобов'язань і послуг. Воно включає в себе не тільки електронне управління, а й електронне участь громадян " [5]. Інші дослідники, Рейнемен і фон Лак, пропонують розділяти поняття "Електронний уряд" і "Електронна демократія". Так, електронна демократія, на їх думку, означає "електронне подання демократичних процесів" [6]. А дослідники Парісек і Сібоек пропонують розділити ці демократичні процеси на наступні "підпроцеси":

а) збір інформації;

б) формування загальної думки;

в) прийняття рішення.

Поняття ж електронна демократія, на їх думку, складається з двох основних частин:

1) підготовка до прийняття рішення (тобто стадії "а" і "б");

2) область електронного голосування або прийняття рішень (тобто стадія "в") [10].

У цьому сенсі електронне голосування розглядається, не просто як фіксація волі виборців з використанням електронних технологій, а як процес прийняття політично, юридично і соціально значущих для держави і суспільства рішень за допомогою процедури виборів. Таким чином, можна прийти до

висновку, що електронне голосування потрібно розглядати і в матеріальному (змістовному) аспекті, тобто як процес прийняття політично і юридично значущих рішень, і у формальному аспекті - як процес фіксації волі виборців з використанням електронних технологій

Такий поділ науково виправданим, оскільки поняття електронний уряд, електронна демократія та електронне голосування мають різне смислове зміст (фактично поняття електронний уряд є похідним від поняття електронна демократія), а у зв'язку з цим потрібно їх доктринальне розмежування, варіант якого ми і запропонували. Більше того, саме в такому відношенні (розгляд електронного голосування, як складової частини електронної демократії) нам видається правильним будувати подальші міркування і висновки.

Схема електронного голосування представлена на рис. 2.1.

Рисунок 2.1 – Схема електронного голосування

Застосовується така класифікація форм електронного голосування.

1. Електронне голосування на виборчій дільниці. Причому, електронне голосування на виборчій дільниці можна розділити, на думку, наприклад, дослідника Рене Пералта на:

а) систему прямого запису (DRE) з використанням сенсорних екранів [13];

б) систему оптичного сканування, що являє собою пристрої введення, що використовують промені світла для сканування кодів (штрих-кодів), тексту або графічних зображень, при цьому отримані дані миттєво передаються в комп'ютер

або систему комп'ютерів; тут варто відзначити, що в межах системи оптичного сканування виділяються дві базові технологічних схеми

- штрих-код сканер;

- система ручного сканування.

Необхідно зауважити і те, що в деяких варіаціях цих двох технологічних схем заповнений виборчий бюлетень сканується за допомогою спеціального сертифікованого сканера, який повідомляє виборцю про допущені неточності в заповненні бюлетеня і дозволяє почати заповнення спочатку, що дозволяє уникнути подальшого відсіювання неправильно заповнених бюлетенів при підрахунку голосів.

2. Електронне голосування в неконтрольованій мережі Інтернет (Інтернет-голосування).

Що ж до типових (розповсюджених) технологічних рішень, то, припустимо, арабська дослідник Ноха Ель Мікава (Noha El-Mikawy) [4] виділяє в їх числі такі:

а) так звані перфокарти, які одночасно є і рахунковими машинами (особливо поширені в США);

б) оптичні машини, які при голосуванні сканують електронну карту для голосування (особливо поширені в США);

в) машини для голосування з сенсорним екраном, в яких вибір кандидата проводиться виборцями за допомогою використання екранних сенсорів (рідинних, ємнісних або органічних), а у випадках помилки у підрахунку голосів деякі моделі таких машин повідомляють про це (Бразилія, Голландія).

Варто відзначити, що такий специфічний вид електронного голосування, як Інтернет-голосування, поширений в значно меншій мірі, і суспільство ставиться до нього з великою побоюванням, ніж до традиційного голосуванню, оскільки виробляється воно в неконтрольованій середовищі Інтернету, причому повний контроль за цим процесом в рамках існуючої архітектури в принципі неможливий.

Саме тому основними проблемами Інтернет-голосування є вірусні і так звані "Denial of service attack" або "DOS" атаки.

Такого роду атаки можливі, коли передача даних здійснюється через комп'ютерну мережу, тобто мережа, в межах якої два або більше персональних комп'ютера з'єднані між собою для передачі даних в електронному вигляді, при цьому підключення між комп'ютерами не завжди буває фізичним. На практиці розрізняють два типи мереж - це локальні мережі (LAN) і глобальні мережі (WAN). Вважаємо за необхідне відзначити тут те, що сама архітектура Інтернету передбачає об'єднання безліч глобальних (WAN) мереж.

Однак, як перші (LAN), так і другі (WAN) використовуються в рамках технології Інтернет-голосування. Локальні мережі використовуються, як правило, для обробки голосів, а глобальні мережі для їх отримання.

Рисунок 2.2 – Структура адміністративної частини

Рисунок 2.3 – Структура клієнтської частини

2.2 Розробка методики авторизації виборця

Блокчейн – це розподілена база даних, у якій дані зберігаються у вигляді ланцюжку блоків, який постійно зростає, захищений від підробки та переробки даних. Кожен блок ланцюжку складається з заголовку та списку транзакцій. Заголовок блоку містить інформацію хеші попереднього блоку, хешсумах транзакцій, які увійшли у цей блок, свій хеш та час створення блоку. Перший блок в ланцюжку називають первинним блоком і розглядають як окремий випадок, оскільки він не має материнського блоку [10]. Для того щоб новий блок був прийнятим іншими користувачами мережі, він має задовольняти певним вимогам, які варіюються у залежності від обраного протоколу консенсусу. Найбільш поширеними протоколами консенсусу є доказ виконання роботи (Proof of Work (PoW)) та доказ долі власності (Proof of Stake (PoS)) [11]. Інші протоколи консенсусу застосовуються для вирішення вузького спектру задач [12].

Схема роботи технології блокчейн зображена на рисунку 2.4

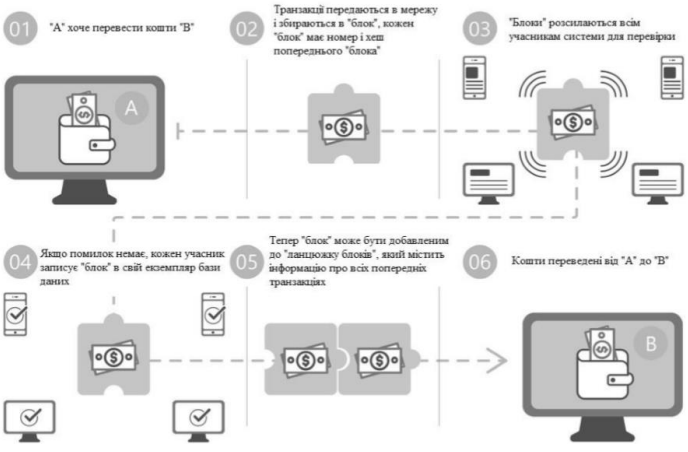


Рисунок 2.4 – Схема роботи технології блокчейн

Голосування за допомогою системи електронного голосування складається з наступних кроків:

1. Після перевірки особи виборця надається одноразовий ключ oneTimeKeyN, який необхідний для того, щоб виборець міг додати свій відкритий ключ pubKeyN у базу даних voterRegisterDB.

2. Виборець, використовуючи свій персональний пристрій, генерує електронний цифровий підпис, а саме відкритий ключ pubKeyN та особистий ключ prvKeyN. Авторизувавшись на сайті системи, за допомогою одноразового ключа oneTimeKeyN, виборець публікує свій відкритий ключ pubKeyN. Відкритий ключ pubKeyN публікується у списку виборців voterRegisterDB навпроти відповідного виборця. Генерація ключів є обов'язковою процедурою для усіх виборців, які бажають вперше прийняти участь у електронному голосуванні.

3. Виборець має завантажити та встановити програмне забезпечення для електронного голосування на свій персональний пристрій. Хеш-сума програмного забезпечення programHash має бути опублікована на сайті системи, для того щоб виборець мав змогу переконатися, що було завантажене та встановлене офіційне програмне завантаження та в нього не були внесені будь-які зміни.

4. Виборець має авторизуватися у системі електронного голосування за допомогою особистого ключа prvKeyN.

5. Якщо у поточний момент відбуваються вибори, то система автоматично розпочне завантаження копій баз даних bulletinDB та votesDB, побудованих на основі технології блокчейн, через мережу Інтернет. У іншому випадку завантаження баз даних bulletinDB та votesDB розпочнеться одразу ж після початку голосування, у якому виборець має право прийняти участь.

6. Після закінчення завантаження баз даних bulletinDB та votesDB система електронного голосування автоматично відправить запит на отримання випадкових вхідних даних. Вхідні данні автоматично, підписуються електронним цифровим підписом та публікуються у списку виборців навпроти відповідного виборця.

7. Генерується бюлетень bulletinN, який складається з унікальної послідовності символів, та публікується у списку виборців.

8. Виборець підписує свій бюлетень bulletinN особистим ключем prvKeyN та відправляє запит на додання підписаного бюлетеня у базу даних bulletinDB.

9. Після перевірки того, що цей бюлетень ще не був доданий до бази даних bulletinDB та підписаний особистим ключем виборця prvKeyN, він додається у мережу блокчейн bulletinDB, а виборцю надається можливість додати свій голос voteN у базу даних votesDB.

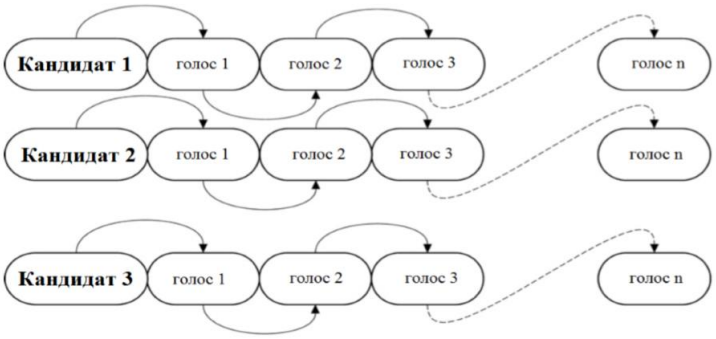
10. У залежності від того, які вибори проходять на даний момент, виборець віддає свій голос за одного з кандидатів, політичну партію або рішення референдуму. Після того, як виборець проголосував, йому пропонується впродовж 30 секунд намалювати на екрані смартфону або планшету, чи за допомогою комп’ютерної миші або тачпаду, у випадку голосування через персональний комп’ютер, будь-який випадковий малюнок. Цей малюнок та час голосування буде використаний для генерації хеш-суми hashN. Генерація хеш-суми hashN відбуватиметься на персональному пристрої виборця. Ця хеш-сума hashN буде додана до голосу виборця, який буде записаний у базу даних votesDB, а також збережена на персональному пристрої виборця. Це дозволить виборцю ідентифікувати свій голос серед інших, для того щоб перевірити те, що він записаний у базу даних votesDB та його значення не було змінено. Хеш-сума hashN не передається виборцем , тому ніхто окрім виборця не знатиме, кому належить голос виборця voteN, тому виборець не може бути ідентифікований ні адміністратором, ні іншими виборцями.

11. Голос виборця voteN записується у базу даних votesDB. Можливість виборця додати голос voteN до бази даних votesDB анулюється.

12. Виборець отримує змогу перевірити те, що його голос voteN був записаний до бази даних votesDB та не був змінений.

Безпека в технології блокчейн забезпечується через децентралізований сервер, проставляються мітки часу, і однорангові мережеві з'єднання. В результаті формується база даних, яка управляється автономно, без єдиного

центру [33]. Це робить ланцюжок блоків дуже зручними для реєстрації подій (наприклад, внесення медичних записів) та операцій з даними, управління ідентифікацією та перевірки походження. Перша транзакція, додана до блоку, буде спеціальною операцією, яка представляє кандидата, проілюстровано на рисунку 2.5. Коли ця транзакція буде створена, вона буде включати ім'я кандидата і буде служити основним блоком, причому кожний голос для цього конкретного кандидата покладеться на неї.



с

Рисунок 2.5 – Схема блокчейн структури кожного кандидата

На відміну від інших транзакцій, фонд не буде враховувати голосування, і буде містити лише ім'я кандидата. Система електронного голосування дозволить провести голосування за протест, де виборець може віддати порожній голос, щоб продемонструвати незадоволеність усіма кандидатами або відмову від нинішньої політичної системи або виборів. Кожного разу, коли людина оцінює транзакцію, вона буде записана, а блокчейн буде оновлено.

Щоб забезпечити безпеку системи, блок буде містити попередню інформацію про виборця. Якщо будь-який з блоків буде скомпрометовано, тоді це легко з'ясувати, оскільки всі блоки з'єднані один з одним. Блокчейн є децентралізований і не може бути пошкодженим, немає єдиної точки відмови. У блокчейні відбувається фактичне голосування. Голосування користувача надсилається на один з вузлів системи, а вузол потім додає голосування до блоку. Система голосування матиме вузол у кожному районі для забезпечення децентралізації системи.

2.3 Розробка методу та алгоритму прийняття рішення

Задача голосування належить до задач теорії прийняття рішень і досліджується вченими вже на протязі багатьох століть. Серед основних методів голосування можна виділити метод відносної більшості, правило Кондорсе, Борда, Компленда, Сімпсона [15]. Згадані методи базуються на різних підходах аналізу індивідуальних переваг виборців і, в загальному, передбачають можливість отримання різних результатів при розв'язуванні однієї і тої ж задачі. Аналіз методів голосування та їх застосування для різних практичних задач показали, що часто виникають випадки, коли при використанні того чи іншого методу переможцем в задачі голосування стає кандидат, який посідає останнє місце в індивідуальних перевагах більшості виборців, або випадки, в яких на результати голосування мають вирішальний вплив переваги виборців, які голосують «проти всіх». Таким чином, виникає необхідність розробки методів розв'язання задачі голосування, які б дозволили враховувати не тільки індивідуальні переваги виборців, а й такі їх суб'єктивні характеристики як негативізм чи надмірний оптимізм.

Суб'єктивний характер вхідних даних у задачі голосування можливо врахувати застосувавши теорію нечітких множин.

Розглянемо загальну задачу голосування . Нехай дано множину кандидатів та множину виборців. Кожен виборець задає індивідуальну перевагу на множині кандидатів у вигляді строгого ранжування, тобто задає лінійний порядок. Систему всіх індивідуальних переваг називають профілем голосування. Необхідно визначити найкращого в деякому смислі кандидата або ж колективний порядок. В такій постановці задача голосування може бути розв'язана відповідно до одного з наступних правил.

Правило відносної більшості. Відповідно до правила відносної більшості перемагає той кандидат, який набрав найбільшу кількість голосів, тобто який посів перше місце в індивідуальних перевагах найбільшої кількості виборців. Як

відомо, хоча і формально застосування такого підходу дозволяє врахувати волю більшості, проте в деяких випадках воно приводить до обрання кандидата, який при парному порівнянні програє буд-якому іншому кандидату з множини С. Правило відносної більшості з вибуванням забезпечує виграш кандидату, який набрав абсолютну більшість голосів, тобто кандидату, який є на першому місці в індивідуальних перевагах абсолютної більшості виборців. У випадку, коли з початкових умов переможця встановити неможливо, відповідно до даного правила проводиться другий тур, в якому виборцям пропонується здійснити вибір з двох «кращих» кандидатів, які набрали найбільшу кількість голосів в початковій задачі.

Правило Борда. У цьому випадку переможця визначають шляхом підрахунку очок таким чином: за останнє місце в індивідуальній перевазі виборця кандидату нараховують 0 очок, за передостаннє - 1, і так далі. Відповідно, за перше місце кандидат отримує М1 - бал. Переможцем визнається той кандидат, який в сумі отримав найбільшу кількість балів.

Правило Кондорсе. За Кондорсе переможцем є той кандидат, що перемагає всіх інших у парних порівняннях. Можливі випадки, коли переможця таким чином встановити неможливо. Існує ще багато інших методів голосування. Деякі з них враховують тільки те, кого виборець вважає найкращим кандидатом, інші беруть до уваги індивідуальну перевагу виборця на всій множині кандидатів. Проте, в деяких задачах голосування важливим є не тільки врахування індивідуального ранжування виборця, а й таких факторів як ставлення виборця до голосування загалом, здатність виборця виявити кращого на його думку кандидата тощо. Суб’єктивність таких факторів можна успішно використати шляхом введення нечітких оцінок кандидатів та їх врахування.

Таким чином, алгоритм визначення колективного порядку в нечіткій задачі голосування може бути представлений як послідовність таких кроків:

1. Формування множини кандидатів та множини виборців.
2. Побудова кандидатами власних нечітких множин «Переможець»

3. Вибір та застосування правил нечіткого логічного виведення для врахування суб’єктивності оцінок виборців.

4. Вибір та послідовне застосування одного з правил для визначення колективного порядку в задачі голосування.

2.4 Висновки

В другому розділі розроблено структуру та алгоритми основних модулів системи електронного голосування. Розроблено загальну структуру системи, методику авторизації виборця за допомогою технології блокчейн, метод та алгоритм прийняття рішення за допомогою теорії нечітких множин.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ЕЛЕКТРОННОГО ГОЛОСУВАННЯ

3.1 Обґрунтування вибору мови програмування

Ruby – високорівнева мова програмування для швидкого і зручного об'єктно-орієнтованого програмування. Мова володіє незалежною від операційної системи реалізацією багатопоточності, суворою динамічною типізацію, збирачем сміття і багатьма іншими можливостями. За особливостями синтаксису вона близька до мов Perl і Eiffel, з об'єктно-орієнтованого підходу - до Smalltalk. Також деякі риси мови взяті з Python, Lisp, Dylan і Клу [10].

Ruby on Rails - це повноцінний, багаторівневий фреймворк для побудови web-додатків, що використовують бази даних, який заснований на архітектурі Модель-Представлення-Контролер (Model-View-Controller, MVC).

Динамічний AJAX-інтерфейс, обробка запитів і видача даних в контролерах, предметна область, відображена в базі даних, - для всього цього Rails надає однорідне середовище розробки на Ruby. Все, що необхідно для початку – база даних і web-сервер.

Rails використовують всюди - від стартапів і некомерційних організацій до великого бізнесу. Rails - це перш за все інфраструктура, тому середовище чудово підходить для будь-якого типу web-додатків, будь це програми для організації спільної роботи, підтримки спільнот, електронного бізнесу, управління змістом, статистики, управління і так далі. Варто зазначити, що такі гіганти як Twitter і Github написані на Ruby [11].

PHP – це широко використовувана мова сценаріїв загального призначення з відкритим вихідним кодом.

PHP – мова програмування, спеціально розроблена для написання web-додатків (скриптів, сценаріїв), що виконуються на Web-сервері. Синтаксис мови багато в чому ґрунтується на синтаксисі C, Java і Perl. Він дуже схожий на C і на Perl, тому для професійного програміста не важко його вивчити. З іншого боку,

мова PHP простіше, ніж C, і її може освоїти web-майстер, який не знає поки інших мов програмування.

Плюсом PHP, на відміну від, наприклад, JavaScript, є те, що PHP-скрипти виконуються на стороні сервера. PHP не залежить від швидкості комп'ютера користувача або його браузера, він повністю працює на сервері. Користувач навіть може не знати, чи він отримує звичайний HTML-файл чи результат виконання скрипта. Сценарії на мові PHP можуть виконуватися на сервері у вигляді окремих файлів, а можуть інтегруватися в html сторінки.

PHP підтримує роботу з ODBC і великою кількістю баз даних: MySQL, MSQL, Oracle, PostgreSQL, SQLite і ін.

JavaScript спочатку створювався для того, щоб зробити web-сторінки "живими". Програми на цій мові називаються скриптами. У браузері вони підключаються безпосередньо до HTML і, як тільки завантажується сторінка - тут же виконуються. Сучасний JavaScript - це "безпечний" мова програмування загального призначення. Він не надає низькорівневих засобів роботи з пам'яттю, процесором, так як спочатку був орієнтований на браузери, в яких це не потрібно.

Що ж стосується інших можливостей - вони залежать від оточення, в якому запущений JavaScript. У браузері JavaScript вміє робити все, що відноситься до маніпуляції зі сторінкою, взаємодії з відвідувачем і, в якійсь мірі, з сервером:

- Створювати нові HTML-теги, видаляти існуючі, змінювати стилі елементів, ховати, показувати елементи і т.п.
- Реагувати на дії відвідувача, обробляти кліки миші, переміщення курсора, натискання на клавіатуру і т.п.
- Посилати запити на сервер і завантажувати дані без перезавантаження сторінки (ця технологія називається "AJAX").
- Отримувати і встановлювати cookie, запитувати дані, виводити повідомлення

JavaScript - швидкий і потужний мову, але браузер накладає на його виконання деякі обмеження ...

Це зроблено для безпеки користувачів, щоб зловмисник не міг за допомогою JavaScript отримати особисті дані або якось нашкодити комп'ютеру користувача.

Цих обмежень немає там, де JavaScript використовується поза браузера, наприклад на сервері. Крім того, сучасні браузери надають свої механізми по установці плагінів і розширень, які володіють розширеними можливостями, але вимагають спеціальних дій по установці від користувача JavaScript не може читати / записувати довільні файли на жорсткий диск, копіювати їх або викликати програми. Він не має прямого доступу до операційної системи.

Сучасні браузери можуть працювати з файлами, але ця можливість обмежена спеціально виділеної Директорією - "пісочницею". Можливості щодо доступу до пристроїв також опрацьовуються в сучасних стандартах і частково доступні в деяких браузерах.

JavaScript, що працює в одній вкладці, не може спілкуватися з іншими вкладками і вікнами, за винятком випадку, коли він сам відкрив це вікно або декілька вкладок з одного джерела (однаковий домен, порт, протокол).

Є способи це обійти, і вони розкриті в підручнику, але вони вимагають спеціального коду на обидва документи, які знаходяться в різних вкладках або вікнах. Без нього, з міркувань безпеки, залізти з однієї вкладки в іншу за допомогою JavaScript можна.

З JavaScript можна легко посилати запити на сервер, з якого прийшла сторінка. Запит на інший домен теж можливий, але менш зручний, І тут є обмеження безпеки.

Для вибору мови програмування виділимо відмінності трьох мов програмування PHP, Ruby, JavaScript.

Таблиця 3.1 – Порівняльні характеристики мов програмування

Критерій	PHP	Ruby	Javascript
Дата створення	1995	1995	1995
Зручність розробки	+	+++	++
Легкість вивчення	++	++	+++
Популярність	++++	+	++++
Необхідність серед роботодавців	++++	++	++++
Швидкість роботи	++	++	+++
Швидкість розробки	+	+++	+++
Масштабованість	++	+++	+++

Основними критеріями для вибору мови програмування були: зручність розробки, швидкість розробки, швидкість роботи та масштабованість, тому вибір – мова програмування Javascript, а саме на фреймворк – Express.js.

3.2 Розробка програмної реалізації модулів системи

Система розроблена із розділеною клієнтською та серверною частиною - клієнт-серверний додаток.

Серверна частина - набір REST API (ще посилання) для комунікації між клієнтом та базою даних, із відповідними перевірками і маніпуляціями даних.

Серверна сторона написана на node.js з використанням фреймворку express.js та не реляційної бази даних mongoDB.

Перелік бібліотек які використовувались на сервері:

- "bcryptjs": "^2.4.3"
- "cookie-parser": "~1.4.3"
- "debug": "~2.6.9",
- "express": "~4.16.0",
- "http-errors": "~1.6.2"
- "jade": "~1.11.0",

- "jsonwebtoken": "^8.2.1",
- "lodash": "^4.17.10",
- "mongodb": "^3.1.0-beta4"
- "mongoose": "^5.1.2",
- "morgan": "^1.9.0",
- "validator": "^10.2.0"

Серверний REST API має публічні API та API із аутентифікацією - для адмін панелі. Для створення функціоналу аутентифікації за основу було взято технологію json web token (JWT).

Перелік серверних REST API із їх параметрами знаходиться за наступним посиланням.

Клієнтська частина створення типу SPA(Single Page Application) із рендером даних на клієнті на основі front-end фреймворку VueJS. Фреймворк VueJS було розширено рективною моделлю даних за допомогою Vuex та додано роутинг(маршрутизацію) за допомогою Vue-router.

Перелік бібліотек які використовувались на клієнті:

"dependencies": {
  "axios": "^0.18.0",
  "bootstrap-vue": "^2.0.0-rc.11",
  "chart.js": "^2.7.2",
  "font-awesome": "^4.7.0",
  "material-design-icons-iconfont": "^3.0.3",
  "vue": "^2.5.11",
  "vue-chartjs": "^3.3.1",
  "vue-material-design-icons": "^1.5.1",
  "vue-router": "^3.0.1",
  "vuelidate": "^0.7.4",
  "vuetify": "^1.0.18",

```
  "vuex": "^3.0.1"
 },
 "devDependencies": {
  "babel-core": "^6.26.0",
  "babel-loader": "^7.1.2",
  "babel-preset-env": "^1.6.0",
  "babel-preset-stage-3": "^6.24.1",
  "cross-env": "^5.0.5",
  "css-loader": "^0.28.7",
  "file-loader": "^1.1.4",
  "stylus": "^0.54.5",
  "stylus-loader": "^3.0.2",
  "vue-cli-plugin-vuetify": "^0.1.3",
  "vue-loader": "^13.0.5",
  "vue-template-compiler": "^2.4.4",
  "webpack": "^3.6.0",
  "webpack-dev-server": "^2.9.1"
 }
```


Web-застосунок електронних президентських виборів створений для того, щоб виборці могли цілком анонімно залишити свій голос за вибраного кандидата.

Для того щоб розпочати процес голосування спочатку потрібно увійти в адмін панель, яка вимагає авторизації:

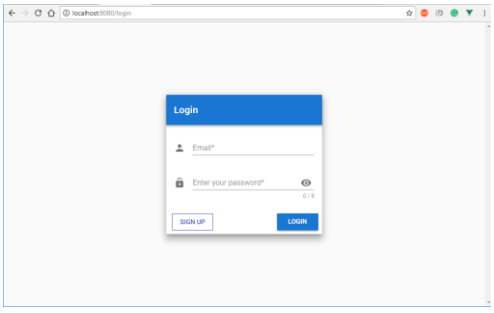


Рисунок 3.1 – Авторизація

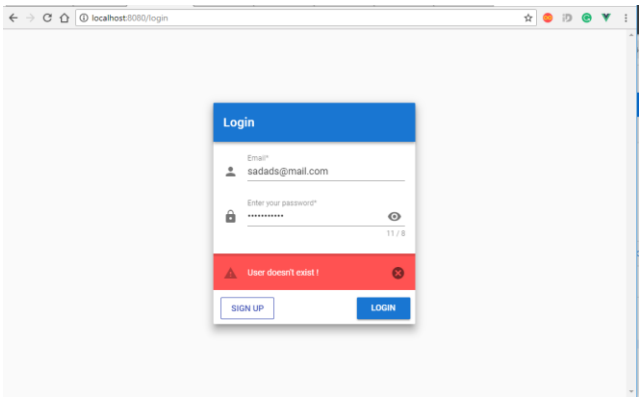


Рисунок 3. 2 – Валідація даних

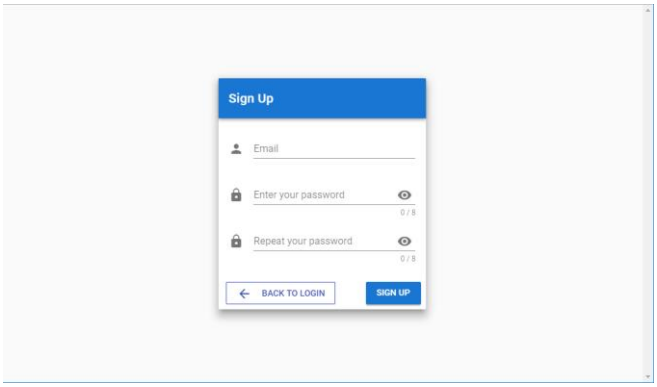


Рисунок 3.3 –Реєстрація

Після входу в адмін панель ми бачимо наступну сторінку:

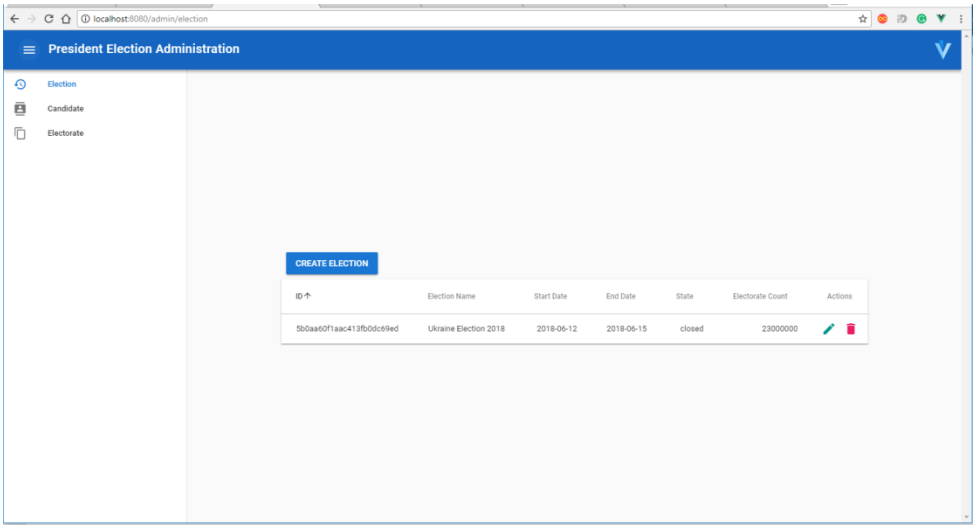


Рисунок 3.4 – Персональний кабінет

Зліва у нас знаходиться панель навігації яка містить 3 посилання:

1. Вибори
2. Кандидати
3. Виборці

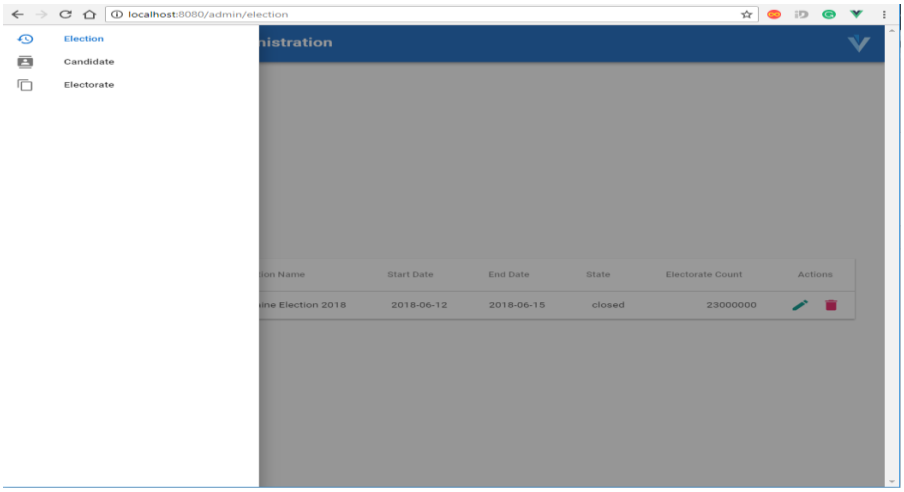


Рисунок 3.5 – Робота з даними

Всю праву частину займає панель для роботи із даними:
Основним елементом для роботи є табличка, яка відображає список існуючої інформації, та надає можливість редагувати, створювати, видаляти дані. Елементи редагування та видалення знаходяться справо від кожного запису. Елемент для створення нових записів знаходиться над табличкою.

Також усі таблички мають можливість сортування за будь-якою колонкою. Сортування здійснюється кліком по назві заголовку колонки.

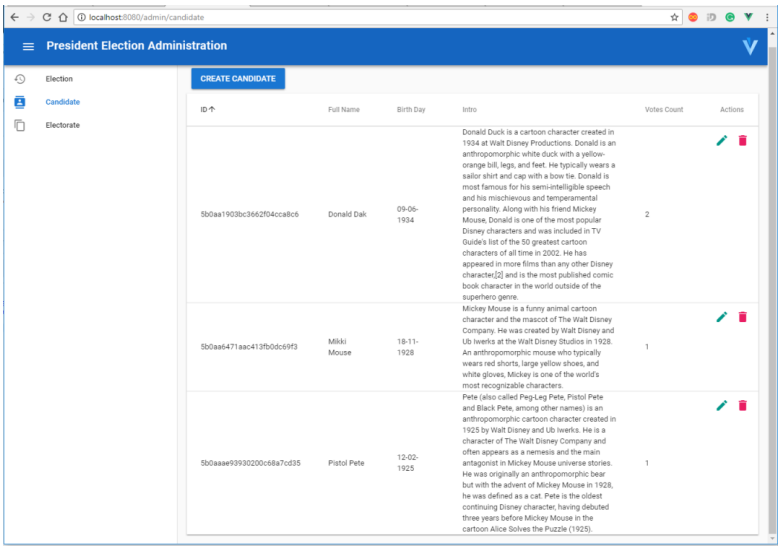


Рисунок 3.6 – Початковий вигляд

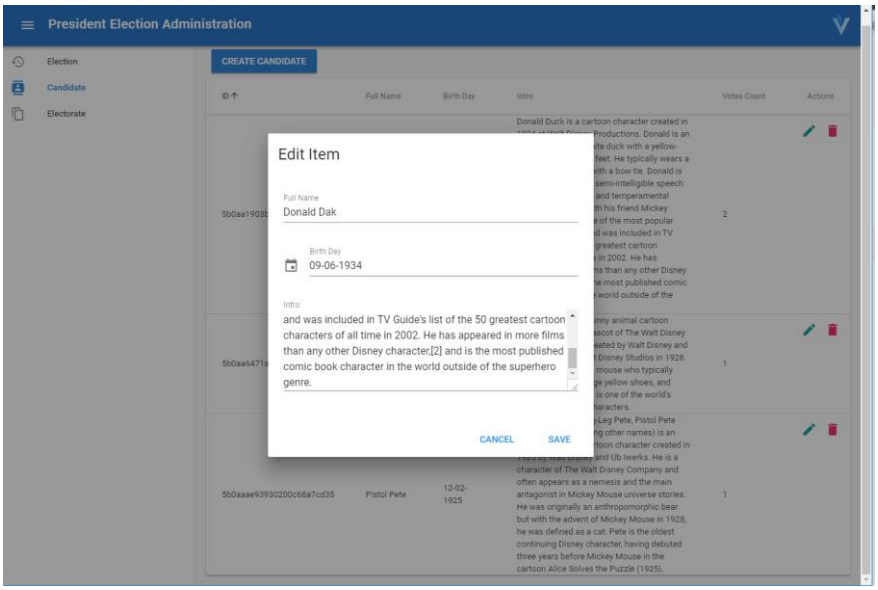


Рисунок 3.7 – Додавання кандидата

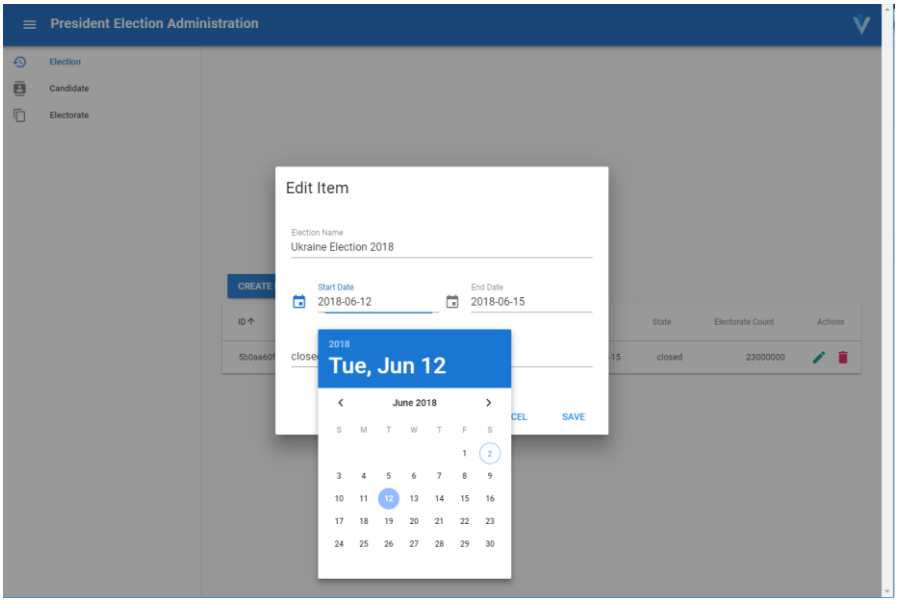


Рисунок 3.7 – Редагування

Під час створення виборів, потрібно вказати повну кількість тих, хто має можливість голосувати. Використовуючи отриману кількість буде згенеровано відповідний обсяг унікальних хеш-ключів для кожного виборця, для того щоб вони могли пройти процес підтвердження і здійснити свій вибір. Дані хеш-ключі можуть роздаватися на виборчих дільницях. Вони ніяк не прив’язані до особи,

тому весь процес голосування є цілком анонімним. Особисті дані виборців у системі не будуть зберігатися.

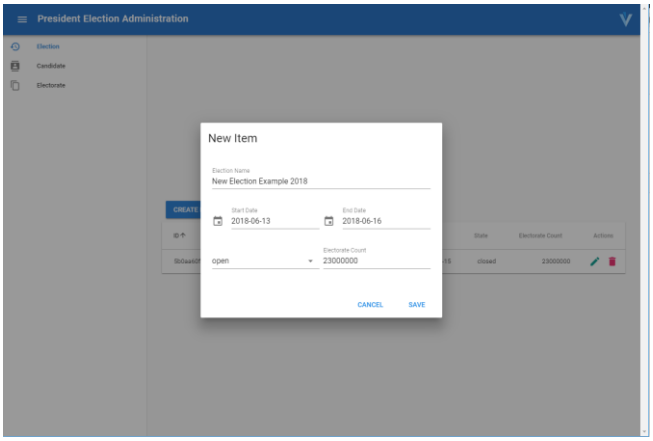


Рисунок 3.8 – Створення нових виборів

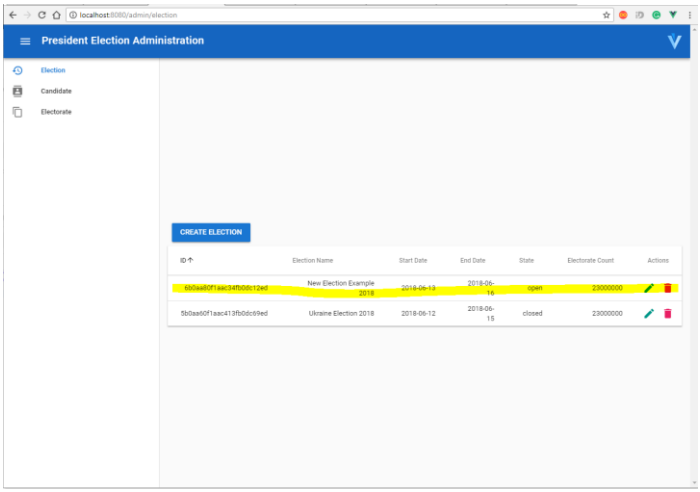


Рисунок 3.9 – Нові електронні вибори із вибраним статусом - Open(відкриті))

Натиснувши на елемент видалення ми отримуємо діалогове вікно із підтвердженням нашого вибору.

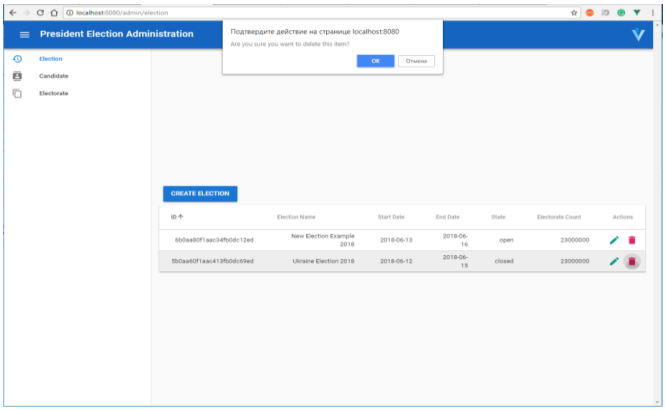


Рисунок 3.10 –Підтвердження вибору

Після підтвердження запис видаляється.

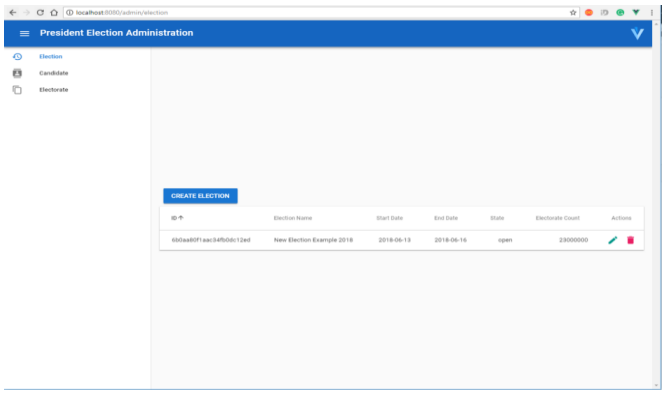


Рисунок 3.11 – Видалення записів

У вкладці Electorate(виборці) ми маємо список усіх доступних виборців. У даній таблиці ми можемо бачити хеш-ключ, статус голосування, та до якого виборчого процесу відносяться виборці.

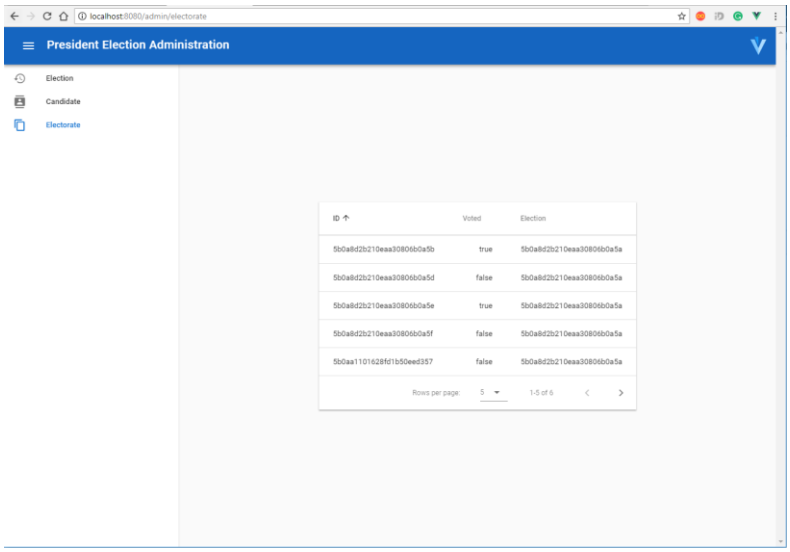


Рисунок 3.12 – Голосування

Для того щоб побачити застосунок для голосування, потрібно натиснути на іконку, яка знаходиться у правому верхньому кутку, або ж якщо ми не авторизовані у нашу адмін панель, то ми зможемо бачити тільки застосунок для голосування.

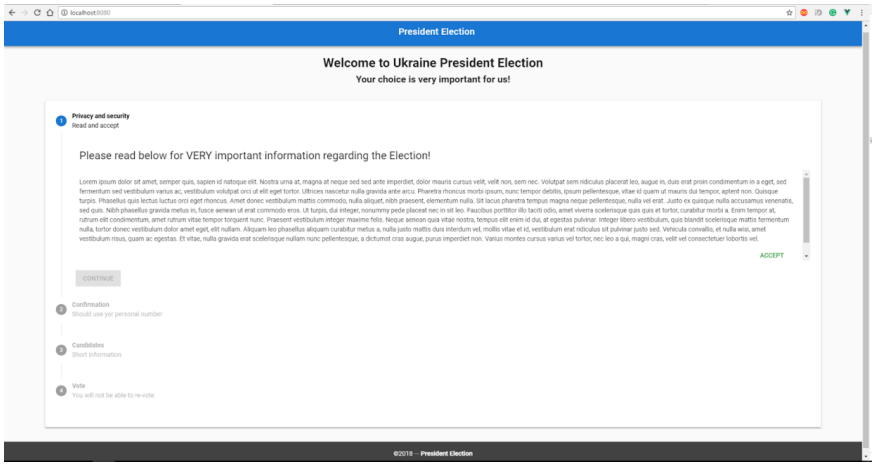


Рисунок 3.13 – Головна сторінка застосунку для голосування

Процес голосування розділений на 4 кроки:

1. Ознайомлення із Правами громадянина та процесом голосування. Після ознайомлення потрібно підтвердити те, що прочитали та ознайомилися. Без підтвердження неможливо перейти до наступного етапу.

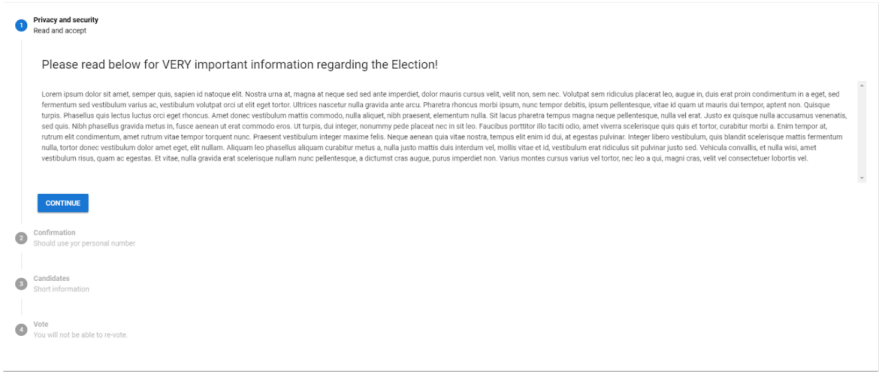


Рисунок 3. 14 – Початковий вигляд сторінки для голосування

2. Наступний крок підтвердження того, що виборець має право голосу. Підтвердження відбувається шляхом введення отриманого ключа.

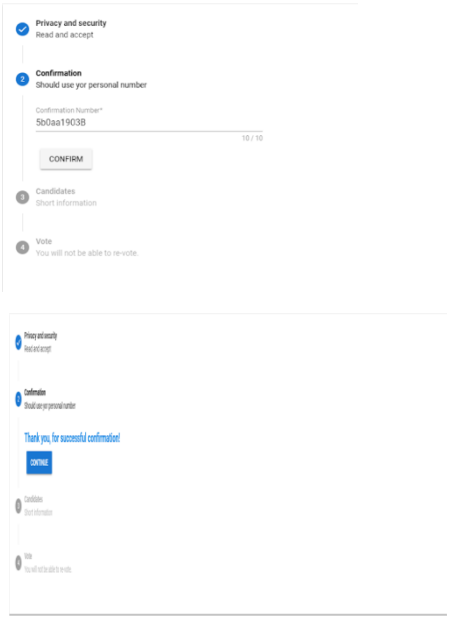


Рисунок 3.15 – Підтвердження права голосу

3. Третій крок призначений для ознайомлення із кандидатами. В даному кроці є список кандидатів, та коротка інформація про них:

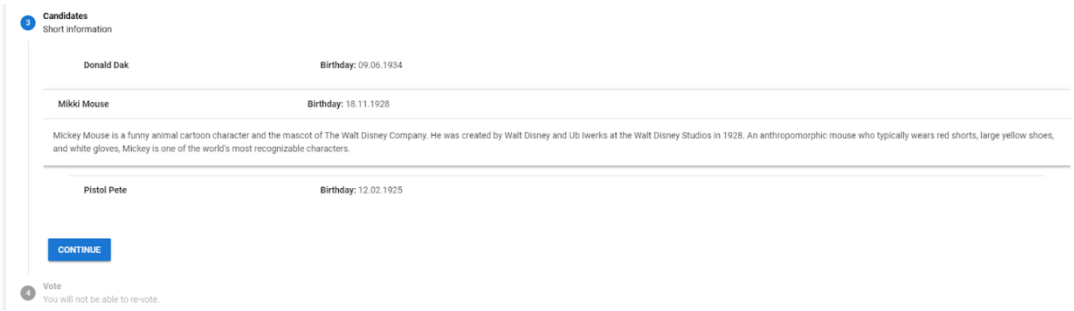


Рисунок 3.16 – Ознайомлення з кандидатами

4. Фінальний крок – здійснення голосування.

Рисунок 3. 17 – Голосування за обраного кандидата

5. Після голосування отримуємо повідомлення про успішність вибору.

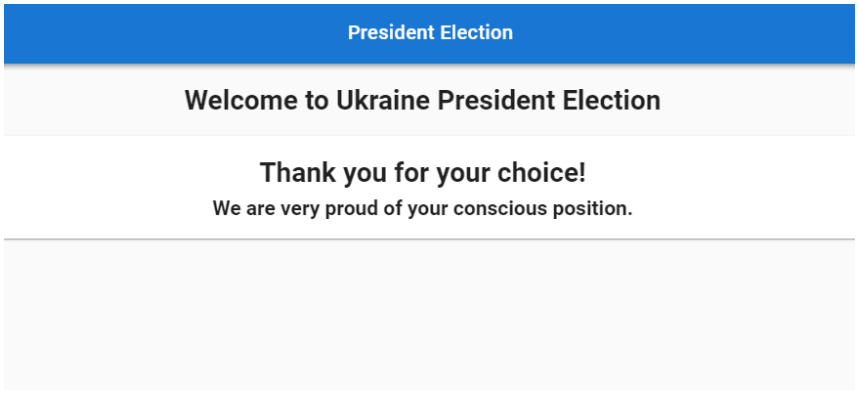


Рисунок 3. 18 – Підтвердження голосування

6. Після того як статус голосування буде змінено із open(відкрите) до closed(закрите) виборці на тій же сторінці зможуть ознайомитися із результатами голосування.

Рисунок 3. 19 – Результати голосування

На даній сторінці відображається переможець виборів, та інтерактивна діаграма, взаємодіючи з якою можна побачити і відсотковому відношенні кількість голосів за кожного кандидата.

3.3 Розробка бази даних системи гологування

Mongoose ODM у Node.js це M в MVC – модель яка є шаром в системі, відповідальним за подання бізнес-логіки і даних. MongoDB позиціонує себе як передова NoSQL база даних. Це твердження засноване на статистичних даних, так що довести зворотне досить важко.

MongoDB використовується зокрема в MEAN — "все в одному" JS фреймворку для розробки web-додатків. Перевага використання баз даних подібних до MongoDB полягає в тому, що вона надає можливість легкої роботи з даними у форматі JSON у будь-який частині вашої програми. Це значить, що ви можете передавати і зберігати дані у форматі JSON на на всіх рівнях: фронтенд (Angular), бекенд (Node) і, власне, база даних – MongoDB [22].

Express.js, або просто Express, каркас web-додатків для Node.js, реалізований як вільне і відкрите програмне забезпечення під ліцензією MIT. Він спроектований для створення web-додатків і API [3]. Де-факто є стандартним каркасом для Node.js. TJ Holowaychuk створений на основі написаного на мові Ruby каркаса Sinatra, він мінімалістичний і включає велику кількість додаткових плагінів. Express може бути backend'ом для програмного стека MEAN, разом з базою даних MongoDB і каркасом AngularJS для frontend'a. Можливості фреймворку:

- Помічники перенаправлення (redirect helpers)
- Динамічні помічники уявлень
- Опції уявлень рівня додатки
- Взаємодія з контентом
- Монтування додатків
- Зміни, швидко перемикаються під різні завдання
- Збережені в сесії спливаючі повідомлення (flash messages)
- Зроблено на основі Connect
- Скрипт express для швидкої генерації каркаса додатки
- Високе покриття тестами

3.4 Висновки

Отже, під час виконання третього розділу було проаналізовано основні засоби реалізації та обрано оптимальні засоби для розробки: мова програмування Javascript, а саме на фреймворк Express.js.

Розроблено модуль авторизації виборця із застосуванням технології блокчейн та модуль прийняття рішення .

Також було створено систему для електронного голосування.

4. ТЕСТУВАННЯ СИСТЕМИ

4.1 Аналіз методів тестування

Тестування програмного забезпечення[33] – процес технічного дослідження, що використовується для виміру якості виготовленого програмного продукту. Іншими словами тестування – процес, що виконується на вимогу замовників, і призначений для виявлення інформації про якість продукту відносно контексту, де він має використовуватись. В цей процес входить виконання програми з метою виявлення помилок у роботі програмного продукту. Тестування може оцінюватись[34]:

- відповідністю вимогам, якими керувалися розробники;
- відповідь усіх можливих вхідних даних;
- виконання функцій за прийнятний час;
- практичність;
- сумісність з програмним забезпеченням та операційними системами;
- відповідність задачам замовника.

Так як число можливих тестів для нескладних програм практично нескінченне, стратегія тестування полягає в тому, щоб провести всі можливі тести з урахуванням наявного часу і ресурсів. В результаті програмне забезпечення тестується стандартним виконанням програми з метою виявлення помилок. Тестування програмного забезпечення може надати об'єктивну, незалежну інформацію про якість ПЗ, ризики відмови, як для користувачів так і для замовників. Тестування може проводитись, як тільки створено виконуваний код[35]. Процес розробки зазвичай передбачає коли та і як буде відбуватися тестування. При поетапному процесі, більшість тестів відбувається після визначення системних вимог і реалізовуються в тестових програмах.

4.2 Тестування програмного продукту

Тестування, як завершальний етап розробки системи, грає життєво важливу роль в процесі створення якісного програмного забезпечення. Чим складніше система, тим більше часу потрібно на перевірку і налагодження. На жаль, існує безліч прикладів, коли розробники і замовники втрачають етап тестування, що практично завжди призводить до великих фінансових і тимчасових витратах надалі, невдоволення користувачів ресурсу, і в результаті, необхідність доопрацювання (або навіть повторної розробки) ресурсу. Залежно від специфіки проекту, на тестування може виділятися до 50% загального бюджету і часових ресурсів [17].

1) Починається все з підготовчих робіт. Тестувальник вивчає отриману документацію (аналізує функціонал по технічним завданням, вивчає кінцеві макети сайту і створює план тесту для подальшого тестування)

2) Функціональне тестування це найбільш тривалий етап перевірки ресурсу. Суть цього процесу полягає в перевірці всього описаного функціоналу:

- перевірки роботи всіх обов'язкових функцій;
- тестування працездатності призначених для користувача форм (наприклад, зворотний зв'язок, додавання коментаря і так далі);
- перевірки роботи пошуку;
- перевірки гіперпосилань, пошук непрацюючих посилань;
- перевірки під завантаження файлів на сервер;
- перевірки працездатності лічильників, встановлених на сторінках сайту;
- перегляд на відповідність вмісту сторінок сайту вихідного контенту, наданого замовником.

3) Тестування верстки. При перевірці верстки насамперед тестувальник перевіряє розташування елементів, відповідність їх позицій наданим макетам, а також перевіряє оптимізацію зображень і графіки. Далі здійснюється перевірка

валідності коду. У процесі верстки важливо дотримуватися коректної ієрархії об'єктів, і важливо впевнитися у її валідності за фактом завершення робіт. Браузери, незважаючи на явно невірний код, в будь-якому випадку спробують відобразити web-сторінку. Але оскільки не існує єдиного регламенту про те, як же повинен бути показаний "кривий" документ, кожний браузер намагається зробити це по-своєму. А це в свою чергу призводить до того, що один і той же документ може виглядати по-різному в різних браузерах. Виправлення явних промахів і систематизація коду призводить, як правило, до стабільного результату. Завершивши перевірку на валідність, фахівець приступає до перевірки на підтримку у багатьох браузерах, тобто перевіряє працездатність сайту в різних браузерах, а також при різних параметрах налаштувань екрану.

4) Usability тестування проводиться для оцінки зручності продукту у використанні. Метод заснований на залученні користувачів в якості тестувальників і аналізі отриманих результатів [18].

Незважаючи на той факт, що опрацювання зручності використання ресурсу здійснюється в процесі складання технічного завдання, розробки макетів, бувають ситуації, коли отриманий результат не є оптимальним. Хоча таке і відбувається досить рідко, оптимальне рішення в даному випадку - внести зміни в реалізований товар.

Тестування проводиться за участю декількох осіб з цільової аудиторії, так званих респондентів. Для проведення тестування досить 4-6 чоловік. Існує правило 80/20, яке свідчить, що 20% користувачів дають 80% результату. Тому така кількість респондентів максимально ефективно з точки зору економії часу і витрат [14].

5) Тестування безпеки. На цій стадії тестування фахівець перевіряє - чи немає у користувачів доступу до службових / закритих сторінок, а також проводить перевірку захисту всіх критично важливих сторінок (наприклад, розділу адміністрування сайту) від зовнішнього впливу.

6) Тестування продуктивності сайту проводиться з метою визначення швидкодії сайту або його частини під певним навантаженням. Тестування продуктивності включає в себе такі види тестування:

Тестування навантаженням. Найпростіша форма тестування продуктивності. Тестування навантаженням зазвичай проводиться для того, щоб оцінити поведінку сайту (або додатки) під заданої очікуваної навантаженням. Цим навантаженням може бути, наприклад, очікувана кількість одночасно працюючих користувачів на сайті, що здійснюють вказану кількість транзакцій за інтервал часу. Такий тип тестування зазвичай дозволяє отримати час відгуку всіх найважливіших бізнес-функцій.

Тестування швидкодії це перевірка швидкості завантаження сайту для визначення швидкості відпрацювання скриптів, завантаження зображень і контенту. Цей тест проводиться з метою оптимізації процесу завантаження сайту, а також визначення оптимальності налаштувань сервера [19].

4.3 Висновки

Проведено тестування системи. Під час проведення тестування не було виявлено, що сайт відображається однаково в найбільш популярних браузерах.

5. ЕКОНОМІЧНА ЧАСТИНА

5.1 Оцінювання комерційного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності. Результатом магістерської кваліфікаційної роботи «Розробка методу та засобів побудови системи для проведення електронного голосування» є розробка програмних засобів засоби системи електронного голосування. Для проведення технологічного аудиту залучено двох незалежних експертів. У нашому випадку такими експертами є: Черноволик Галина Олександрівна (к.т.н., доцент каф. ПЗ, ВНТУ), Катєльніков Денис Іванович (к.т.н., доцент каф. ПЗ, ВНТУ), Майданюк Володимир Павлович (к.т.н., доцент каф. ПЗ, ВНТУ).

Оцінювання комерційного потенціалу здійснено за критеріями, що наведені в таблиці 5.1.

Таблиця 5.1 – Критерії оцінювання комерційного потенціалу розробки

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів

Продовження таблиці 5.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років

Продовження таблиці 5.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки пові-домлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу експертами розробки зведено в таблицю 5.2.

Таблиця 5.2 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	1 – Черноволик	2 – Катєльніков	3 – Майданюк
	Бали, виставлені експертами:		
1	4	4	4
Ринкові переваги (недоліки):			
2	3	2	3
3	3	3	3
4	4	4	4
5	3	4	3
Ринкові перспективи			
6	3	3	3
7	4	4	3
Практична здійсненність			
8	4	3	4
9	3	3	3
10	3	3	3
11	4	4	4
12	4	4	4
Сума балів	$СБ_1$=42	$СБ_2$=41	$СБ_3$=41
Середньоарифметична сума балів $\overline{СБ}$	41, 3		

За даними таблиці 5.2 можна зробити висновок, щодо рівня комерційного потенціалу розробки. Зважимо на результат й порівняємо його з рівнями комерційного потенціалу розробки, що представлено в таблиці 5.3.

Таблиця 5.3 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів $\overline{СБ}$, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0 – 10	Низький
11 – 20	Нижче середнього
21 – 30	Середній
31 – 40	Вище середнього
41 – 48	Високий

Рівень комерційного потенціалу розробки, становить 41,3 балів, що відповідає рівню «високий».

В ході проведення порівняльного аналізу існуючих аналогів, який було проведено в першому розділі магістерської роботи, було виявлено, що швидкість роботи з базою даних більша ніж в аналога, за рахунок прямого виконання SQL запитів, а не використання сторонніх моделей для доступу до баз даних виборців. Швидкість реагування на дії користувача швидше ніж в аналога, за рахунок того, що використовується асинхронний операцій з даними.

Для роботи з файлами у власній розробці використовуються бібліотеки, які не вимагають наявності встановленого програмного забезпечення (Microsoft Office), для перегляду та редагування файлів, тобто власна розробка працює відразу з файлом, а не відкриваючи додаткове програмне забезпечення, що дає можливість зменшити кількість затрачуваного часу на операціях імпорту та експорту

У таблиці 5.4 наведено основні технічні показники аналога і нового програмного продукту.

Таблиця 5.4 – Основні технічні показники аналога і нової розробки

Критерій	«ВТБ-регистратор»	Власний програмний продукт	Відношення параметрів нової розробки до параметрів аналога
Швидкість роботи з базою даних	0,9	1	1,11
Економія ресурсів і часу	0,8	1	1,25
Функціональність	0,6	1	1,66
Захист	0,1	1	10
Швидкість реагування на дії користувача	0,8	1	1,25
Швидкість роботи з файлами	0,8	1	1,25

Нова розробка покриває недоліки існуючих рішень і забезпечує більшу зручність та функціональність при використанні програми. Економія ресурсів та часу нової розробки виявляється в одночасній синхронізації декількох баз даних. Функціональність розробки в 1,25 раз перевищує функціональність розробки аналогу, що зазначено у таблиці 5.4.

Отже, основними перевагами створеного продукту є:

- Можливість врахування даних суб’єктивного характеру.
- Можливість перевірки виборцем врахування свого голосу.
- Підвищений рівень захисту системи за рахунок застосування технології блокчейн.

Розроблений продукт не можна назвати зовсім новим. Дане рішення є модифікацією продуктів, які вже існують на ринку.

Основні потреби, які задовольняє розробка, це забезпечення можливості електронного голосування, захист від фальсиікації.

Прогнозується поступове збільшення попиту на розробку, оскільки її ціна є нижчою за ціни конкурентів, а функціональність більша.

Розроблений програмний продукт планується реалізовувати шляхом ліцензійної угоди. Придбавши програму, покупець (організатор виборів) зможе відразу завантажити її на свій комп’ютер та отримає ліцензійний ключ для її активації. Також можливим є розповсюдження безкоштовної версії продукту з обмеженим функціоналом. По закінченню терміну користувачеві буде запропоновано придбати повну версію програмного продукту. Такі заходи забезпечують додаткове збільшення продажів.

Впровадження розробки зможе покращити умови праці, оскільки за рахунок зручності, автоматизованості та зменшення трудомісткості процесу вона знизить рівень психологічного та фізичного навантаження працівників.

Крім того розробка є програмним продуктом, що не впливає негативно на навколишнє середовище.

Розробка технічно готова та розміщена на сайті клієнта.

Виходячи зі специфіки нової розробки, здійснювати просування її на ринок доцільно використовуючи метод прямого продажу, який дозволить працювати індивідуально з кожним покупцем. Характерними рисами прямого продажу є безпосередній контакт зі споживачем, демонстрація продукції та вичерпна консультація продавця.

Крім того, після реалізації розробки взаємодія з клієнтом буде продовжуватись у вигляді послуг з технічної підтримки та супроводження експлуатації продукту, зокрема:

- поточна підтримка – виправлення дефектів програмного забезпечення та технічна підтримка;
- розширення функціональних можливостей програмного продукту;
- модифікація програмного продукту у відповідності зі змінами законодавства та умов експлуатації.

Працюватиме зворотній зв'язок з клієнтами через електронну пошту та портал технічної підтримки.

5.2 Прогнозування витрат на виконання наукової роботи та впровадження результатів

Проведемо прогнозування витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи для розробки програмного забезпечення, яке складається з таких етапів:

1-й етап: розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи;

2-й етап: розрахунок загальних витрат на виконання даної роботи;

3-й етап: прогнозування загальних витрат на виконання та впровадження результатів даної роботи.

Виконаємо розрахунок витрат приймаючи до уваги те, що розробкою займався один розробник програмного забезпечення.

Основна заробітна розробника-дослідника $З_о$:

$$З_о = \frac{М}{Т_р} \cdot t \text{ [грн]}, \quad (5.1)$$

де М – місячний посадовий оклад – 7500 грн;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р$ = (22) дні;

t – число робочих днів роботи розробника-дослідника – 42.

$$Зо = (7500/22)\cdot 42 = 14318{,}18 \text{ (грн)}.$$

Додаткова заробітна плата $З_д$ розробника розраховується як 10% від основної заробітної плати:

$$З_д = 0{,}10 \cdot 14318{,}18 = 1431{,}81 \text{ (грн)}.$$

Нарахування на заробітну плату $Н_{зп}$ розробника становить:

$$H_{зп} = (З_о + З_д) \cdot \frac{\beta}{100} \text{ [грн]}, \quad (5.2)$$

де $З_о$ – основна заробітна плата розробника;

$З_д$ – додаткова заробітна плата розробника;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування – 22%.

$$Н_{зп} = (14318,18 + 1431,81) \cdot 0,22 = 3464,99 \text{ (грн)}.$$

Амортизація обладнання, комп'ютерів та приміщень, які використовувались під час виконання даного етапу роботи. Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо.

У спрощеному вигляді амортизаційні відрахування розраховуємо за формулою:

$$А = (Ц \cdot Т)/(12 \cdot Тв) \text{ [грн]}, \quad (5.3)$$

де Ц – загальна балансова вартість обладнання, приміщення тощо, грн;

Т – фактична тривалість використання, міс;

$Т_В$ – термін використання обладнання, приміщень тощо, роки.

Розробка програмного забезпечення проводилася протягом 2 місяців.

Зроблені розрахунки зведено до таблиці 5.5.

Таблиця 5.5 – Амортизаційні відрахування

Найменування	Балансова вартість, грн	Термін використання, роки	Фактична трив. використання, міс.	Величина амортизаційних відрахувань, грн
Офісне приміщення	100000	25	2	66,70
Комп'ютер	15000	5	2	500,00
Монітор	3700	6	2	102,70
Всього				669,40

Інформацію про матеріали, що використовуються при виготовленні даного інноваційного продукту внесено до таблиці 5.6.

Таблиця 5.6 – Матеріали, що використовуються при виготовленні даного продукту

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено, шт.	Вартість витраченого матеріалу, грн
Папір (пачка)	78,00	1	78,00
Ручка	6,00	2	12,00
Всього			90,00

Під час розробки програмного продукту використовувались лише безкоштовні програмні засоби.

Витрати на енергію визначаються на основі витрат на одиницю продукції та тарифів на енергію за допомогою формули:

$$B_e = B \cdot П \cdot Ф \cdot K_n \, [грн], \tag{5.4}$$

де В – вартість 1кВт електроенергії;

П – установлена потужність обладнання, кВт;

Ф – фактична кількість годин роботи при створенні програмного продукту, годин;

$К_п$ – коефіцієнт використання потужності.

Отже, витрати на енергію становлять:

$$B_e = 1{,}68 \cdot 0{,}5 \cdot 352 \cdot 0{,}4 = 132{,}35 \, (грн).$$

Також потрібно врахувати витрати на доступ до мережі Інтернет, що використовувався під час виконання роботи.

Витрати за доступ до Інтернет можна розрухувати за формулою:

$$B_{дi} = Ц_{дi} \cdot T \, [грн], \tag{5.5}$$

де $Ц_{ді}$ – це ціна доступу за місяць;

Т – кількість місяців використання доступу до мережі.

$$В_{ді} = 130 \cdot 2 = 260{,}00 \text{ (грн)}.$$

Інші витрати $В_{ін}$ охоплюють: витрати на управління організацією, оплату службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо. Інші витрати $І_в$ можна прийняти як 150% від суми основної заробітної плати розробника:

$$В_{ін} = 1,5 \cdot 14318,18 = 21477,27 \text{ (грн)}.$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи – В.

$$В = З_о + З_д + Н_{зп} + А + В_{мат} + В_е + В_{ін} \text{ [грн]},$$
$$В = 14318,18 + 1431,81 + 3464,96 + 669,40 + 90 + 132,35 + 260 + 21477,27 = 41843,97 \text{ (грн)}.$$

2-й етап. Розрахунок загальних витрат на виконання даної роботи. Загальна вартість всієї наукової роботи визначається за $В_{заг}$ формулою:

$$В_{заг} = \frac{В}{\alpha} \text{ [грн]}, \quad (5.6)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях.

Так, як над роботою задіяна одна людина, якою виконується вся робота, то α становить 1. Підставивши дані у формулу, отримуємо:

$$В_{заг} = 41843,97 \text{ (грн)}.$$

3-й етап. Прогнозування загальних витрат на виконання та впровадження результатів виконаної роботи. Прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної роботи здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta} \text{ [грн]}, \quad (5.7)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи. Так, якщо розробка знаходиться:

- на стадії науково-дослідних робіт, то $\beta \approx 0{,}1$;
- на стадії технічного проектування, то $\beta \approx 0{,}2$;
- на стадії розробки конструкторської документації, то $\beta \approx 0{,}3$;
- на стадії розробки технологій, то $\beta \approx 0{,}4$;
- на стадії розробки дослідного зразка, то $\beta \approx 0{,}5$;
- на стадії розробки промислового зразка, $\beta \approx 0{,}7$;
- на стадії впровадження, то $\beta \approx 0{,}9$.

Отже, підставимо дані в формулу й отримаємо результат:

$$ЗВ=41843{,}97/0{,}9=46493{,}30 \text{ (грн)}.$$

5.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Спрогнозуємо вигоду яку можна отримати у майбутньому від впровадження результатів виконаної наукової роботи.

Всі зроблені тут розрахунки будуть приблизними і не передбачають деталізації.

В умовах ринку, узагальнюючим позитивним результатом, який планує отримати підприємець від впровадження результатів нової розробки, є збільшення чистого прибутку. Зростання чистого прибутку ми можемо оцінити у теперішній вартості грошей.

Зростання чистого прибутку забезпечить надходження додаткових коштів, які дозволять покращити фінансові результати діяльності та виплатити кредити (якщо вони потрібні для впровадження результатів розробки).

Оцінити збільшення чистого прибутку підприємства $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки можливо використовуючи формулу:

$$\Delta П_i = \sum_i^n \left(\Delta П_o \cdot N + Ц_o \cdot \Delta N\right) \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\upsilon}{100}\right) [грн], \qquad (5.8)$$

де ΔПо – покращення основного оціночного показника від впровадження результатів розробки у даному році. Зазвичай таким показником може бути ціна одиниці нової розробки;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки;

λ – коефіцієнт, який враховує сплату податку на додану вартість.

ρ – коефіцієнт, який враховує рентабельність продукту. Рекомендується приймати ρ = 0,25;

υ – ставка податку на прибуток (18%).

В результаті впровадження результатів наукової розробки покращується якість кінцевого продукту, що дозволяє підвищити ціну його реалізації на 200 грн. Кількість одиниць реалізованої продукції збільшиться: протягом першого року – 50 шт., протягом другого року – 80 шт., протягом третього року –100 шт.

Орієнтовно: реалізація продукції до впровадження результатів наукової розробки складала 1 шт., а ціна становила 4000.

Спрогнозуємо збільшення чистого прибутку підприємства від впровадження результатів наукової розробки у кожному році відносно базового.

Збільшення чистого прибутку підприємства Δ Пі протягом першого року складе:

$$\Delta П_1 = [4000 \cdot 1 + 50 \cdot 4200] \cdot 0{,}88 \cdot 0{,}25 \cdot \left(1 - \frac{18}{100}\right) = 38605{,}60\ (грн).$$

Збільшення чистого прибутку підприємства Δ Пі протягом другого року (відносно базового року, тобто року до впровадження результатів наукової розробки) складе:

$$\Delta П_2 = [4000 \cdot 1 + 80 \cdot 4200] \cdot 0{,}88 \cdot 0{,}25 \cdot \left(1 - \frac{18}{100}\right) = 61336{,}00\ (грн).$$

Збільшення чистого прибутку підприємства Δ Пі протягом третього року (відносно базового року, тобто року до впровадження результатів наукової розробки) складе:

$$\Delta П_3 = [4000 \cdot 1 + 100 \cdot 4200] \cdot 0{,}88 \cdot 0{,}25 \cdot \left(1 - \frac{18}{100}\right) = 76489{,}60\ (грн).$$

5.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Основними показниками, які визначають доцільність фінансування наукової розробки інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності. Розрахунок ефективності вкладених інвестицій передбачає проведення таких робіт:

1-й крок. Розраховується теперішня вартість інвестицій PV, що вкладаються в наукову розробку. Такою вартістю можемо вважати прогнозовану величину загальних витрат ЗВ на виконання та впровадження результатів НДДКР, розраховану за формулою, тобто будемо вважати, що ЗВ = PV=46493,30 грн.

2-й крок. Розраховується очікуване збільшення прибутку $\Delta П_i$, що його отримає підприємство (організація) від впровадження результатів наукової розробки, для кожного із років, починаючи з першого року впровадження. Таке збільшення прибутку також було розраховане раніше.

3-й крок. Будуємо вісь часу, на яку наносимо всі платежі (інвестиції та прибутки), що мають місце під час виконання науково-дослідної роботи та впровадження її результатів.

Припустимо, що загальні витрати ЗВ на виконання та впровадження результатів НДДКР (або теперішня вартість інвестицій PV) дорівнює 46493,30 грн. Результати вкладених у наукову розробку інвестицій почнуть виявлятись протягом трьох років. У першому році підприємство отримає збільшення чистого прибутку на 38605,60 грн відносно базового року, в другому році – збільшення чистого прибутку на 61336,00 грн (відносно базового року), у третьому році – збільшення чистого прибутку на 76489,60 грн (відносно базового року).

Тоді рух платежів (інвестицій та додаткових прибутків) буде мати вигляд, наведений на рис. 5.1.

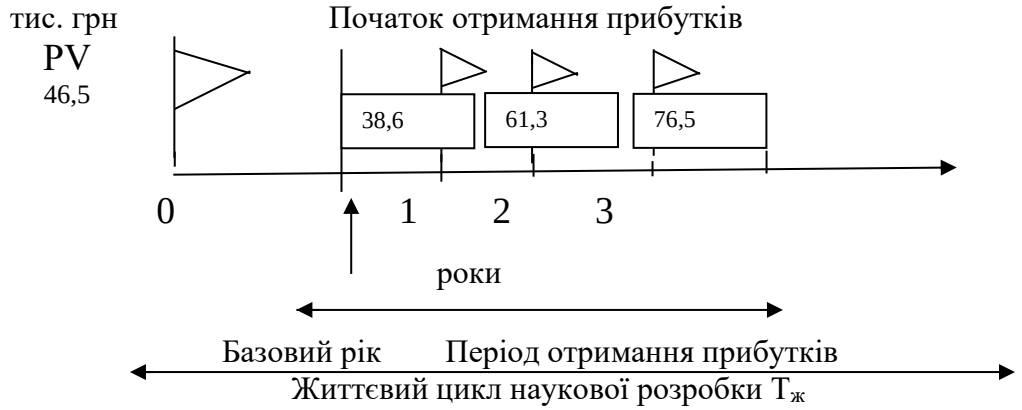


Рисунок 5.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

4-й крок. Розраховується абсолютна ефективність вкладених інвестицій $Е_{абс}$.

Для цього використовується формула:

$$Е_{абс} = (ПП – PV)\ [грн], \quad (5.9)$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство (організація) від реалізації результатів наукової розробки, грн.;

PV – теперішня вартість інвестицій PV = ЗВ, грн.

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_{1}^{m} \frac{\Delta П_i}{(1+\tau)^t} [грн], \quad (5.10)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні;

t – період часу (в роках) від моменту отримання чистого прибутку до точки „0”.

Якщо $Е_{абс} > 0$, то результат від проведення наукових досліджень та їх впровадження принесе прибуток, але це ще не свідчить про те, що інвестор буде зацікавлений у фінансуванні даного проекту (роботи).

Розрахуємо абсолютну ефективність інвестицій, вкладених у реалізацію проекту. Ставка дисконтування τ дорівнює 0,1. Отримаємо:

$$ПП = \frac{38605{,}60}{(1+0{,}1)^1} + \frac{61336{,}00}{(1+0{,}1)^2} + \frac{76489{,}60}{(1+0{,}1)^3} = 138030{,}34\ (грн).$$

Тоді,

$$Е_{абс} = 138030{,}34 – 46493{,}30 = 91537{,}04\ (грн).$$

Оскільки $Е_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР є доцільним.

5-й крок. Розраховуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$. Для цього використовуємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1 [грн], \quad (5.11)$$

де $Е_{абс}$ – абсолютна ефективність вкладених інвестицій, грн;

PV –теперішня вартість інвестицій PV = ЗВ, грн;

$Т_ж$ – життєвий цикл наукової розробки, роки.

Далі, розрахована величина $Е_в$ порівнюється з мінімальною (бар'єрною) ставкою дисконтування τмін, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть.

У загальному вигляді мінімальна (бар'єрна) ставка дисконтування τмін визначається за формулою:

$$\tau = d + f, \quad (5.12)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; d = 0,18;

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = 0,05.

Якщо величина $Е_в > \tau$мін, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки. В іншому випадку фінансування наукової розробки здійснюватися не буде. Спочатку спрогнозуємо величину τмін. Припустимо, що за даних умов τмін = 0,18 + 0,05 = 0,23. Тоді відносна (щорічна) ефективність вкладних інвестицій в проведення наукових досліджень та впровадження їх результатів складе:

$$E_{\text{в}} = \sqrt[3]{1 + \frac{91537,04,}{46493,30}} - 1 = \sqrt[3]{2,96} - 1 = 0,43 \text{ або } 43\%$$

Оскільки $E_{\text{в}} = 43\% > \tau_{\text{мін}} = 0,23 = 23\%$, то у інвестора буде зацікавленість вкладати гроші в дану розробку, оскільки значно більші прибутки він отримає від того, що інвестує кошти в розробку, а не розмістить гроші на депозиті у комерційному банку.

6-й крок. Розраховуємо термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{\text{ок}}$ можна розрахувати за формулою:

$$T_{\text{ок}} = \frac{1}{E_{\text{в}}} \text{ [грн]}. \quad (4.13)$$

Для нашої розробки термін окупності вкладених у реалізацію проекту інвестицій $T_{\text{ок}}$ складе:

$$T_{ок} = \frac{1}{0,43} = 2,32 \text{ (}року\text{)},$$

$T_{\text{ок}}$ є менше трьох років, тому фінансування даної наукової розробки вважається доцільним.

5.5 Висновки

В даному розділі було виконано оцінювання комерційного потенціалу розробки системи електронних виборів.

Проведено технологічний аудит з залученням трьох незалежних експертів. Визначено, що рівень комерційного потенціалу розробки високий.

Аналіз комерційного потенціалу розробки показав, що програмний продукт за своїми характеристиками випереджає аналогічні програмні продукти і є перспективною розробкою. Він має кращі функціональні показники, а тому є

конкурентоспроможним товаром. Існуючі переваги нової розробки дозволять швидко її поширити на ринку.

Згідно із розрахунками всіх статей витрат на виконання науково-дослідної, дослідно-конструкторської та конструкторсько-технологічної роботи загальні витрати на розробку складають 46493,30 грн.

Розрахована абсолютна ефективність вкладених інвестицій в сумі 91537,04 грн свідчить про отримання прибутку інвестором від комерціалізації програмного продукту.

Щорічна ефективність вкладених в наукову розробку інвестицій складає 43 %, що вище за мінімальну бар'єрну ставку дисконтування, яка складає 23%. Це означає потенційну зацікавленість інвесторів у фінансуванні розробки.

Термін окупності вкладених у реалізацію проекту інвестицій становить 2,32 року, що також свідчить про доцільність фінансування нової розробки.

ВИСНОВКИ

Проведено аналіз існуючих методів і засобів систем електронного голосування.

Удосконалено метод прийняття рішення по результатам голосування та методику авторизації виборця.

Розроблено програмні компоненти та систему на основі запропонованих теоретичних розробок.

Проведено тестування розробленої системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Береза А. Сучасні технології голосування. Наук. зап. ІПіЕНД ім. І. Ф. Кураса НАН України. – Вип. 50. – С. 155–160. – URL: http://www.ipiend.gov.ua/uploads/nz/nz_50/bereza_suchasni.pdf.

2. Глущенко Н. Блокчейн в Україні: Що це за технологія і чим вона корисна . –URL: https://ua.112.ua/statji/blokchein-v-ukraini-shcho-tse-za-tekhnolohiiai-chym-vona-korysna-417161.html.

3. Інтернет для громадськості: Рек. Ком. міністрів державам – членам Ради Європи CM/Rec(2016) 2 від 10.02.2016. – URL: https://minjust.gov.ua/m/rekomendatsiiparlamentskoi-asamblei-ta-komitetu-ministriv-radi-evropi.

4. Кохалик Х. Світовий досвід впровадження електронної демократії: проблеми та досягнення. Ефек тивність держ. упр. – 2015. – Вип. 42. – С. 169–174. – URL: http://www.lvivacademy.com/vidavnitstvo_1/edu_42/fail/ 20.pdf.

5. Малиновська О. Зарубіжний досвід реалізації виборчого права громадян за кордоном та можливості його застосування в Україні. Віче. – 2016. – № 7–8. – С. 39–43. 9. Наконечний А. Й. Інтернет речей і сучасних технологій. А. Й. Наконечний, З. Є. Верес. – С. 3–9. – URL: http://ena.lp.edu.ua/bitstream/ntb/36449/1/2_3-9.pdf.

6. Оніпко О. Шляхи вдосконалення технології виборчого процесу. Вісн. Центр. вибор. коміс. – 2006. – Вип. № 4 (6). – С. 67–70. – URL: http://www.cvk.gov.ua/visnyk/pdf/2006_4/ visnyk_st_22.pdf.

7. Мулен, Э. Кооперативное принятие решений: Аксиомы и модели – М.: Мир, 1991. – 464 с.

8. Nakamoto, S., 2008. Bitcoin: a peer-to-peer electronic cash system, November 2008 [Електронний ресурс]. – Режим доступу: https://bitcoin.org/bitcoin.pdf

9. Niemoller K. Experience with Voting Machines in the Netherlands and Germany.bAppendix 2K. [Електронний ресурс]. – Режим доступу: http://www.umic.pt/images/stories/publicacoes1/Appendix%202K.pdf

10. Борисов И.Б., Журавлев В.П. Развитие электронного голосования // Журнал о выборах. – 2011. – № 3. – С. 38–43.

11. Volkamer M. Evaluation of Electronic Voting: Requirements and Evaluation Procedures to Support Responsible Election Authorities. Springer Science & Business Media. – 2009. – P. 39

12. Скальский А. Электронная демократия. [Електронний ресурс]. – Режим доступу: www.newsbabr.com/irk/?IDE=77528.\\ 6. Кукушкин С.Н. Кибервалюта – миф или реальность // Экономика знаний: теория и практика. – 2017. – №1. – С.16

13. Рудина М. Итоги Blockchain & Bitcoin Conference 2016 // Опубликованные материалы конференции Blockchain & Bitcoin Conference, 2016. [Електронний ресурс]. – Режим доступу: https://moscow.blockchainconf.world/ru

14. Hans B. Technology Tipping Points and Societal Impact // Опубликованные материалы Всемирного экономического форума, 2015. [Електронний ресурс]. – Режим доступу: http:// www.weforum.org/docs/WEF_GAC15_Technological_Tipping_Points_report_ 2015.pdf

15. Свон М. Блокчейн: Схема новой экономики: пер. с англ. М.: Издательство «Олимп-Бизнес». – 2017. – 240 с.

16. Волошин, О. Ф. Теорія прийняття рішень: навч. посібн. Ф. Волошин, С. О. Мащенко. – К.: Видавничо-поліграфічний центр «Київський університет», 2010. – 366 с.

17. Ларичев, О. И. Объективные модели и субъективные решения. О. И. Ларичев – М.: Наука, 1987. – 143 с.

18. Тоценко, В. Г. Методы и системы поддержки принятия решений. Алгоритмический аспект. В. Г. Тоценко. – К.: Наук. думка, 2002. – 382 с.

19. Орловский, С. А. Проблемы принятия решений при нечеткой исходной информации [Текст] / С. А. Орловский. – М.: Наука. Главная редакция физико-математической литературы, 1981. – 208 с.

20. Штовба, С. Д. Введение в теорию нечетких множеств и нечеткую логику [Електронний ресурс] / С. Д. Штовба. – Режим доступу: http://matlab.exponenta.ru/fuzzylogic/book1/index.php/

21. Коваль С. С., Черноволик Г. О. Система для проведення електронного голосування. Міжнарод. наук.-практ. конференції «Інформаційні технології і автоматизація», м. Одеса, с. 120-121, 2019.

22. Тестування програмного забезпечення. URL: https://uk.wikipedia.org/wiki/Тестування_програмного_забезпечення (дата звернення: 6.10.2019).

23. White-box testing. URL: https://en.wikipedia.org/wiki/White-box_testing (Last accessed: 5.11.2019).

24. Black-box testing. URL: https://en.wikipedia.org/wiki/Black-box_testing (Last accessed: 5.11.2019).

25. Test case. URL: https://en.wikipedia.org/wiki/Test_case (Last accessed: 5.11.2019).

26. Козловський В.О. Техніко-економічні обґрунтування та економічні розрахунки в дипломних проектах та роботах. Навчальний посібник. / В.О. Козловський – Вінниця: ВДТУ, 2003. – 75 с.

27. Козловський В.О. Методичні вказівки до виконання студентамимагістрантами економічної частини магістерських кваліфікаційних робіт / В.О. Козловський – Вінниця: ВНТУ, 2012. – 22 с.

ДОДАТКИ

Додаток А. Технічне завдання

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
Романюк О. Н.
«____» _____2019 року

Технічне завдання
на магістерську кваліфікаційну роботу
«Розробка методу та засобів побудови системи для проведення електронного голосування»
за спеціальністю 121 – Інженерія програмного забезпечення

Керівник магістерської кваліфікаційної роботи:
_____ к.т.н., доц. Г.О. Черноволик
"____"_____2019 р.
Виконав:
_____студент гр. 2ПІ-18м С. С. Коваль
"____"_____2019 р.

Вінниця — 2019 року

1. Найменування та галузь застосування

Магістерська кваліфікаційна робота: «Розробка методу та засобів побудови системи для проведення електронного голосування».

Галузь застосування – системи електронного голосування.

2. Підстава для розробки.

Підставою для виконання магістерської кваліфікаційної роботи (МКР) є індивідуальне завдання на МКР та наказ №___ ректора по ВНТУ про закріплення тем МКР.

3. Мета та призначення розробки.

Метою роботи є підвищення надійності захисту анонімності інформації та зменшення можливості фальсифікації за рахунок розробки системи для проведення електронного голосування.

Призначення роботи – розробка методу та засобів побудови системи проведення електронного голосування.

3 Вихідні дані для проведення НДР

Перелік основних літературних джерел, на основі яких буде виконуватись МКР.

1. Конахович Г.Ф. Захист інформації в мережах передачі даних: підручник / Г.Ф. Конахович, О.Г. Корченко, О.К. Юдін. – К.: Видавництво ТОВ НВП «ІНТЕРСЕРВІС», 2009. – 714 с.

2. Оніпко О. Шляхи вдосконалення технології виборчого процесу. Вісн. Центр. вибор. коміс. – 2006. – Вип. № 4 (6). – С. 67–70. – URL: http://www.cvk.gov.ua/visnyk/pdf/2006_4/ visnyk_st_22.pdf.

4. Технічні вимоги

Мова програмування – JavaScript, фреймворк Express.js.

5. Конструктивні вимоги.

Конструкція пристрою повинна відповідати естетичним та ергономічним вимогам, повинна бути зручною в обслуговуванні та керуванні.

Графічна та текстова документація повинна відповідати діючим стандартам України.

6. Перелік технічної документації, що пред'являється по закінченню робіт:

– пояснювальна записка до МКР;

– технічне завдання;

– лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії та етапи розробки:

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи
1	Аналіз предметної галузі та постановка задач розробки	
2	Порівняльний аналіз аналогів	
3	Аналіз методів розв'язання задач	
4	Розробка загальної структури системи	
5	Розробка алгоритму роботи основних модулів	
6	Програмна реалізація додатку	
7	Тестування розробленої системи	
8	Економічна частина	

10. Порядок контролю та прийняття.

Виконання етапів магістерської кваліфікаційної роботи контролюється керівником згідно з графіком виконання роботи.

Прийняття магістерської кваліфікаційної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком.

Додаток Б. Лістинг програми

1) Серверна частина.

a. Rest API

```
require('./config/config');
const _   = require('lodash');
const      express  = require('express');
const   body_parser  = require('body-parser');
const        morgan = require('morgan');
const         {db}  = require('./db/mongoose');
const    {ObjectID}  = require('mongodb');
const   {Candidate}  = require('./models/candidate');
const    {Election}  = require('./models/election');
const  {Electorate}  = require('./models/electorate');
const       {User}  = require('./models/user');
const {authenticate} = require('./middleware/authenticate');
const {electorate_auth} = require('./middleware/electorate_auth');
const          app  = express();
const         port  = process.env.PORT;
app.use(body_parser.json());
app.use(morgan('dev'));
app.use(function(req, res, next) {
  res.header("Access-Control-Allow-Origin", "*");
  res.header("Access-Control-Allow-Headers", "*");
  res.header("Access-Control-Allow-Methods", "*");
  res.header("Accept-Encoding", "gzip, deflate, br");
  res.header("Accept", "*");
  next();
});
app.post('/users', (req, res) => {
  let body = _.pick(req.body, ['email', 'password', 'password_again']);
  if(_.isEmpty(body)){
```

```
return res.status(403).send({message: `The fields are not suitable to process`, success: false});
  }
  if(body.password !== body["password_again"]){
     return res.status(403).send({message: `Password doesn't match`, success: false});
  }
  let user = new User(body);
  user.save().then(() => {
     return user.generate_auth_token();
  }).then((token) => {
     res.header('x-auth', token).status(200).send({user, token});
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false});
  })
});
app.get('/users/me', authenticate, (req, res) => {
  res.send({user: req.user, success: true});
});
app.post('/users/login', (req, res) => {
  let body = _.pick(req.body, ['email', 'password']);
  if(_.isEmpty(body)){
     return res.status(403).send({message: `The fields are not suitable to process`, success: false});
  }
  User.find_by_credentials(body.email, body.password).then((user) => {
     user.generate_auth_token().then((token) => {
        user.token = token;
        res.header('x-auth', token).status(200).send({user, token, success: true});
     });
  }).catch((err) => {
     res.status(401).send(err);
  });
});
app.delete("/users/me/token", authenticate, (req, res) => {
```

```
  req.user.remove_token(req.token).then(() => {
    res.status(200).send({message: "The token was deleted", success: true})
  }).catch((err) => {
    res.status(400).send({message: err.message, success: false});
  });
});
app.post('/election', authenticate, (req, res) => {
  let body = _.pick(req.body, ['name', 'start_date', 'end_date', 'state', 'electorate_count', 'tour']);
  if(_.isEmpty(body)){
    return res.status(403).send({message: `The fields are not suitable to process`, success: false});
  }
  if(body['end_date'] <= body['start_date']){
    res.status(400).send({'message': 'The end date should be more than start date', success: false});
  } else {
    const election = new Election(body);
    election.save().then((doc) => {
      Electorate.create_electorates(doc.electorate_count, doc._id).then((electorates) => {
        res.status(200).send({doc, success: true});
      });
    }).catch((err) => {
      res.status(400).send({message:err.message, success: false});
    });
  }
});
app.get('/election', (req, res) => {
        Election.find().sort('start_date').then((election) => {
                if(election.length === 0){
                        throw new Error('There is not available election');
                }
                res.status(200).send({election, success: true});
        }).catch((err) => {
                res.status(400).send({success: false, message: err.message});
```

```
            });
});
app.get('/election/id/:id', (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Election.findOne({_id: id}).then((election) => {
     if(!election){
        return res.status(404).send({message: `There are not found election by id <${id}>`, success: false});
     }
     res.status(200).send({election, success: true});
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false});
  })
});
app.get('/election/state/:state', (req, res) => {
  let state = req.params && req.params.state;
  if(!state && !_.contains(state, [ 'Open', 'Closed', 'Finished' ])) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Election.find({state}).then((election) => {
     if(!election){
        return res.status(404).send({message: `There are not found election by state <${state}>`, success:
false});
     }
     res.status(200).send({election, success: true});
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false});
  });
});
```

```
app.delete('/election/:id', authenticate, (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Election.findOneAndRemove({_id: id}).then((election) => {
     if(!election){
        return res.status(404).send({message: `There are not found election by id <${id}>`, success: false});
     }
     res.status(200).send({election, success: true});
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false});
  });
});
app.patch('/election/:id', authenticate, (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  //Filter props from body based on provided in arr
  let body = _.pick(req.body, ['state', 'name']);
  if(_.isEmpty(body)){
     return res.status(403).send({message: `The fields are not suitable for updates`, success: false});
  }
  Election.findOneAndUpdate({_id: id}, {$set: body}, {new: true}).then((election) => {
     if(!election){
        return res.status(404).send({message: `There is not found election by id <${id}>`, success: false});
     }
     res.status(200).send({election, success: true});
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false}){
  });
```

```
});
app.get('/electorate/login/:id', (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Electorate.findOne({_id: id}).then((electorate) => {
     if(!electorate){
       return res.status(401).send({message:"You don't have access to current election", success: false});
     }
     res.header('x-elec-id', id).status(200).send(electorate);
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false}){
  });
});
app.post('/candidate', authenticate, (req, res) => {
  let body = _.pick(req.body, ['full_name', 'birth_day', 'intro', '_election']);
  if(_.isEmpty(body)){
     return res.status(403).send({message: `The fields are not suitable to process`, success: false});
  }
  const candidate = new Candidate(body);
  candidate.save().then((doc) => {
     res.status(200).send(doc);
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false});
  });
});
app.get('/candidate', (req, res) => {
          Candidate.find().sort('full_name').select('full_name birth_day intro _election
votes_count').then((candidate) => {
                    if(candidate.length === 0){
                              throw new Error('There is not available candidate');
```

```
                }
                res.status(200).send({candidate, success: true});
        }).catch((err) => {
                res.status(400).send({success: false, message: err.message});
        });
});

app.get('/candidate/id/:id', (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Candidate.findOne({_id: id}).select('full_name birth_day intro _election').then((candidate) => {
     if(!candidate){
        return res.status(404).send({message: `There is not found candidate by id <${id}>`, success: false});
     }
                res.status(200).send({candidate, success: true});
  }).catch((err) => {
     res.status(400).send({message: err.message, success: false});
  });
});
app.get('/candidate/election/:id', (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
     return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Candidate.find({_election: id}).select('full_name birth_day intro _election').then((candidate) => {
     if(!candidate){
        return res.status(404).send({message: `There is not found candidate by election _id <${id}>`, success:
false});
     }
     res.status(200).send({candidate, success: true});
```

```
  }).catch((err) => {
    res.status(400).send({message: err.message, success: false});
  });
});
app.delete('/candidate/:id', authenticate, (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
    return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  Candidate.findOneAndRemove({_id: id}).select('full_name birth_day intro _election').then((candidate) =>
{
    if(!candidate){
      return res.status(404).send({message: `There are not found candidate by id <${id}>`, success: false});
    }
    res.status(200).send({candidate, success: true});
  }).catch((err) => {
    res.status(400).send({message: err.message, success: false});
  });
});

app.patch('/candidate/:id', authenticate, (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
    return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  //Filter props from body based on provided in arr
  let body = _.pick(req.body, ['intro', 'full_name', 'birth_day']);
  if(_.isEmpty(body)){
    return res.status(403).send({message: `The fields are not suitable for updates`, success: false});
  }
  Candidate.findOneAndUpdate({_id: id}, {$set: body}, {new: true}).select('full_name birth_day intro
_election').then((candidate) => {
    if(!candidate){
```

```
        return res.status(404).send({message: `There is not found candidate by id <${id}>`, success: false});
      }
      res.status(200).send({candidate, success: true});
  }).catch((err) => {
      res.status(400).send({message: err.message, success: false});
  });
});

app.patch('/candidate/add_vote/:id', electorate_auth, (req, res) => {
  let id = req.params && req.params.id;
  if(id && !ObjectID.isValid(id)) {
      return res.status(404).send({message: `The id: <${id}> is not valid!`, success: false})
  }
  //Filter props from body based on provided in arr
  let body = _.pick(req.body, ['add_vote']);
  if(!body.add_vote && !body.add_vote === true){
      return res.status(403).send({message: `The fields are not suitable to operate`, success: false});
  }
  Candidate.findOneAndUpdate({_id: id}, {$inc: {votes_count:1}}, {new: true}).select('full_name birth_day
intro _election').then((candidate) => {
      if(!candidate){
          return res.status(404).send({message: `There is not found candidate by id <${id}>`, success: false});
      }
      Electorate.made_vote(req.electorate._id).then((doc) => {
          res.status(200).send({candidate, success: true});
      });
  }).catch((err) => {
      res.status(400).send({message: err.message, success: false});
  });
});

app.get('*', function(req, res){
```

```
  res.status(404).send({message: "The route is not found", success: false});
});

app.listen(port, () => {
  console.log(`Server is up, and listening on port: ${port}`);
});
module.exports = {app};
```

b) MongoDb схема для екземпляру User

```
const mongoose = require('mongoose');
const validator = require('validator');
const Schema = mongoose.Schema;
const jwt = require('jsonwebtoken');
const _ = require('lodash');
const bcrypt = require('bcryptjs');
const User_Schema = new Schema({
  email: {
     type: String,
     validate: {
        validator: (value) => validator.isEmail(value),
        message: '{VALUE} is not a valid email value!'
     },
     required: true,
     trim: true,
     //unique: true,
     dropDups: true
  },
  password: {
     type: String,
     required:true,
     minlength: 6
  },
```

```
  tokens: [{
    access: {
      type: String,
      required: true
    },
    token: {
      type: String,
      required: true
    }
  }]
});
User_Schema.methods.generate_auth_token = function() {
  const User = this,
    access = 'auth',
    token = jwt.sign({_id: User._id.toHexString(), access}, process.env.JWT_SECRET).toString();
  User.tokens.push({access, token});
  return User.save().then(() => token);
};

User_Schema.methods.toJSON = function() {
  const user = this;
  let user_object = user.toObject();
  return _.pick(user_object, ['_id', 'email']);
};

User_Schema.methods.remove_token = function(token){
  const user = this;
  return user.update({
    $pull: {
      tokens: {
        token: token
      }
```

```
    }
  });
};

User_Schema.statics.find_by_token = function(token) {
  const User = this;
  let decoded;
  try {
    decoded = jwt.verify(token, process.env.JWT_SECRET);
  } catch(e) {
    return Promise.reject('The provided token is incorrect!');
  }
  return User.findOne({
    _id: decoded._id,
    'tokens.token': token,
    'tokens.access': 'auth'
  });
};

User_Schema.statics.find_by_credentials = function(email, password) {
  const User = this;
  return User.findOne({email}).then((user) => {
    if(!user){
      return Promise.reject({message: "User doesn\'t exist !", success: false});
    }
    return new Promise((resolve, reject) => {
      bcrypt.compare(password, user.password, (err, res) => {
        if(res){
          resolve(user);
        }else{
          reject({message:(err && err.message) || "The password doesn\'t matches", success: false});
        }
```

```
        });
    });
  });
};
User_Schema.pre('save', function(next){
  const User = this;
  if(User.isModified('password')){
    bcrypt.genSalt(10, (err, salt) => {
      bcrypt.hash(User.password, salt, (err, hash) => {
        User.password = hash;
        next();
      });
    });
  }else{
    next();
  }
});
const User = mongoose.model('User', User_Schema);
module.exports = {
 User
};
```

c) MongoDb схема для екземпляру Election

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const electionSchema = new Schema({
  name: {
    type: String,
    required: true,
    minlength: [6, 'Text should contains at least 6 characters'],
    trim: true
  },
```

```
  start_date: {
     type: Date,
     required: true,
  },
  end_date: {
     type: Date,
     required: true,
  },
  state: {
     type: String,
     enum : [ 'open', 'closed', 'finished' ],
     required: true,
     lowercase: true
  },
  electorate_count: {
     type: Number,
     minlength: 1,
     required: true
  },
  tour:{
     type: Number,
     required: true,
     default: 1
  }
});

const Election = mongoose.model('Election', electionSchema);
module.exports = {
  Election
};
```

d) MongoDb схема для екземпляру Electorate

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const electorateSchema = new Schema({
  voted: {
     type: Boolean,
     required: true,
     default: false
  },
  _election: {
     type: mongoose.Schema.Types.ObjectId,
     required: true
  }
});
electorateSchema.statics.create_electorates = function(count, election_id){
  const electorate = this;
  let arr_electorate = [];
  for(var i = 0; i < count; i++){
     arr_electorate.push({_election: election_id});
  }
  return electorate.insertMany(arr_electorate);
};
electorateSchema.statics.made_vote = function(id){
  const electorate = this;
  return electorate.findOneAndUpdate({_id: id}, {$set: {"voted": true}}, {new: true});
};

const Electorate = mongoose.model('Electorate', electorateSchema);
module.exports = {
  Electorate
};
```

е) MongoDb схема для екземпляру Candidate

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const candidateSchema = new Schema({
  full_name: {
     type: String,
     required: true,
     minlength: [6, 'Text should contains at least 6 characters'],
     trim: true
  },
  birth_day: {
     type: Date,
     required: true,
  },
  intro: {
     type: String,
     required: true,
  },
  votes_count: {
     type: Number,
     required: true,
     default: 0
  },
  _election: {
     type: mongoose.Schema.Types.ObjectId,
     required: true
  }
});

const Candidate = mongoose.model('Candidate', candidateSchema);
module.exports = {
  Candidate
};
```

f) Middleware для аутентифікації у адмін панель

```
const {User} = require('./../models/user');
const authenticate = (req, res, next) => {
  const token = req.header('x-auth');
  if(!token){
     return res.status(401).send({message: "You are not authorised!", success: false});
  }
  User.find_by_token(token).then((user) => {
     if(!user){
        return Promise.reject('You are not authorised!');
     }
     req.user = user;

     req.token = token;

     next();
  }).catch((err) => {
     res.status(401).send({message: err, success: false});
  });
};
module.exports = {authenticate};
```

g) Middleware для підтвердження права виборців на голосування

```
const {Electorate} = require('./../models/electorate');

const  {ObjectID}  = require('mongodb');
const electorate_auth = (req, res, next) => {
  const id = req.header('x-elec-id');
  if(id && !ObjectID.isValid(id)) {
     return res.status(401).send({message: "You are not authorised!", success: false});
  }
```

```
  Electorate.findOne({_id: id, voted: false}).then((electorate) => {
    if(!electorate){
       return Promise.reject('You can not make a vote or already did it!');
    }
    req.electorate = electorate;
    req.token = id;
    next();
  }).catch((err) => {
    res.status(401).send({message: err, success: false});
  });
};
module.exports = {electorate_auth};
```

2) Клієнтська частина

а) Головний компонент клієнтської частини

```
<template>
  <router-view></router-view>
</template>
<script>
 export default {
  name: 'app',
  data() {
   return {
    msg: 'Welcome to Your Vue.js App',
    links: ['Home', 'About Us', 'Team', 'Services', 'Blog', 'Contact Us']
   }
  },
  created() {
   this.$store.dispatch('AUTH_CHECK_REQUEST').then().catch((e) => {
    console.log(JSON.stringify(e))
   });
  }
```

```
 }
</script>
```

b) Головний компонент системи управління

```
<template>
 <v-app id="inspire">
  <v-navigation-drawer
   :clipped="$vuetify.breakpoint.lgAndUp"
   v-model="drawer"
   fixed
   app
  >
   <v-list dense>
    <template v-for="item in items">
     <v-list-tile :key="item.text" :to="item.to">
      <v-list-tile-action>
       <v-icon>{{ item.icon }}</v-icon>
      </v-list-tile-action>
      <v-list-tile-content>
       <v-list-tile-title>
        {{ item.text }}
       </v-list-tile-title>
      </v-list-tile-content>
     </v-list-tile>
    </template>
   </v-list>
  </v-navigation-drawer>
  <v-toolbar
   :clipped-left="$vuetify.breakpoint.lgAndUp"
   color="blue darken-3"
   dark
   app
   fixed
```

```
    >
      <v-toolbar-title style="width: 500px" class="ml-0 pl-3">
        <v-toolbar-side-icon @click.stop="drawer = !drawer"></v-toolbar-side-icon>
        <span class="hidden-sm-and-down">President Election Administration</span>
      </v-toolbar-title>
      <v-spacer></v-spacer>
      <v-btn icon large :to="home">
        <v-avatar size="32px" tile>
          <img
            src="https://vuetifyjs.com/static/doc-images/logo.svg"
            alt="Vuetify"
          >
        </v-avatar>
      </v-btn>
    </v-toolbar>
    <v-content>
      <v-container fluid fill-height>
        <v-layout justify-center align-center>
          <router-view></router-view>
        </v-layout>
      </v-container>
    </v-content>
  </v-app>
</template>

<script>
  export default {
    data: () => ({
      dialog: false,
      drawer: null,
      home: '/',
      items: [
```

```
      { icon: 'history', text: 'Election', to: 'election' },
      { icon: 'contacts', text: 'Candidate', to: 'candidate' },
      { icon: 'content_copy', text: 'Electorate', to: 'electorate' }
    ]
  }),
  props: {
    source: String
  }
 }
</script>
```

c) Головний компонент системи голосування

```
<template>
 <v-app>
  <customer-header></customer-header>
  <v-content>
   <v-container v-if="!election_finished">
    <h1 class="text-xs-center">Welcome to Ukraine President Election</h1>
    <h2 class="text-xs-center mb-3">Your choice is very important for us!</h2>
    <stepper></stepper>
   </v-container>
  </v-content>
  <v-container v-if="election_finished">
   <h1 class="text-xs-center">Ukraine Election 2018 is finished</h1>
   <h3 class="text-xs-center">The winner of 2018's election is Donald Dak</h3>
   <h3 class="text-xs-center mb-5">You can see the election results on the chart below: </h3>
   <results></results>
  </v-container>
  <customer-footer></customer-footer>
 </v-app>
</template>


<script>
```

```
import footer from './election/footer';
import header from './election/header';
import stepper from './election/stepper';
import results from './election/results';
export default {
  name: 'Election',
  components: {
    'customer-footer': footer,
    'customer-header': header,
    'stepper': stepper,
    'results': results
  },
  data() {
    return {
      msg: 'Welcome to Your Vue.js App',
      election_finished: localStorage.getItem("election_finished") == "true",
    }
  }
}
</script>
```

d) Головний компонент авторизації

```
<template>
  <v-app id="inspire">
    <v-content>
      <v-container fluid fill-height>
        <v-layout align-center justify-center>
          <v-flex xs12 sm8 md4>
            <v-card class="elevation-12">
              <v-toolbar dark color="primary">
                <v-toolbar-title>Login</v-toolbar-title>
                <v-spacer></v-spacer>
              </v-toolbar>
```

```
<v-card-text>
  <v-form>
    <v-text-field
    v-model="email"
    prepend-icon="person"
    name="Email"
    label="Email"
    type="email"
    :error-messages="email_errors"
    required
    @input="$v.email.$touch()"
    @blur="$v.email.$touch()"
    ></v-text-field>
    <v-text-field
      v-model="password"
      :append-icon="show_pass ? 'visibility' : 'visibility_off'"
      :append-icon-cb="() => (show_pass = !show_pass)"
      :counter="8"
      :type="show_pass ? 'password' : 'text'"
      :error-messages="password_errors"
      name="password"
      prepend-icon="lock"
      label="Enter your password"
      hint="At least 8 characters"
      min="8"
      required
      @input="$v.password.$touch()"
      @blur="$v.password.$touch()"
    ></v-text-field>
  </v-form>
</v-card-text>
<v-alert v-model="error.show" type="error" dismissible>
```

```
            {{error.message}}
          </v-alert>
          <v-card-actions>
            <v-spacer><v-btn outline @click="transfer('signup')" color="indigo">Sign Up</v-btn></v-spacer>
            <v-btn  color="primary" @click="submit">Login</v-btn>
          </v-card-actions>
        </v-card>
      </v-flex>
    </v-layout>
  </v-container>
 </v-content>
</v-app>
</template>

<script>
 import { validationMixin } from 'vuelidate'
 import { required, minLength, email } from 'vuelidate/lib/validators'

 export default {
  mixins: [validationMixin],
  data: () => ({
   email: '',
   password: '',
   show_pass: true,
   error: {
    show: false,
    message: ''
   }
  }),
  validations: {
   email: { required, email },
   password: { required, minLength: minLength(8) }
```

```
},
 computed: {
  password_errors () {
    const errors = []
    if (!this.$v.password.$dirty) return errors
    !this.$v.password.minLength && errors.push('Password must be at least 8 characters long')
    !this.$v.password.required && errors.push('Password is required')
    return errors
  },
  email_errors () {
    const errors = []
    if (!this.$v.email.$dirty) return errors
    !this.$v.email.email && errors.push('Must be valid e-mail')
    !this.$v.email.required && errors.push('E-mail is required')
    return errors
  }
},
methods: {
  submit: function () {
    this.$v.$touch();
    if (this.$v.$invalid) {
      return false;
    }
    const payload = {email: this.email, password:this.password};
    this.$store.dispatch('AUTH_REQUEST', payload).then(() => {
      this.$router.push('/admin/election');
    }).catch((e) => {
      this.error.show = true;
      if (e.response) {
        this.error.message = e.response.data.message;
      } else {
        this.error.message = e.message;
```

```
        }
       })
     },
     transfer: function (route) {
       this.$router.push('/' + route);
     }
   }
 }
</script>
```

е) Головний компонент реєстрації

```
<template>
  <v-app id="inspire">
    <v-content>
      <v-container fluid fill-height>
        <v-layout align-center justify-center>
          <v-flex xs12 sm8 md4>
            <v-card class="elevation-12">
              <v-toolbar dark color="primary">
                <v-toolbar-title>Sign Up</v-toolbar-title>
                <v-spacer></v-spacer>
              </v-toolbar>
              <v-card-text>
                <v-form>
                  <v-text-field
                  v-model="email"
                  prepend-icon="person"
                  name="Email"
                  label="Email"
                  type="email"
                  :error-messages="email_errors"
                  @input="$v.email.$touch()"
                  @blur="$v.email.$touch()"
```

```
    ></v-text-field>
    <v-text-field
      v-model="password"
      :append-icon="show_pass ? 'visibility' : 'visibility_off'"
      :append-icon-cb="() => (show_pass = !show_pass)"
      :counter="8"
      :type="show_pass ? 'password' : 'text'"
      :error-messages="password_errors"
      name="password"
      prepend-icon="lock"
      label="Enter your password"
      hint="At least 8 characters"
      min="8"
      @input="$v.password.$touch()"
      @blur="$v.password.$touch()"
    ></v-text-field>
    <v-text-field
      v-model="repeat_password"
      :append-icon="show_second_pass ? 'visibility' : 'visibility_off'"
      :append-icon-cb="() => (show_second_pass = !show_second_pass)"
      :counter="8"
      :type="show_second_pass ? 'password' : 'text'"
      :error-messages="second_password_errors"
      name="repeat_password"
      prepend-icon="lock"
      label="Repeat your password"
      hint="Password should match"
      @input="$v.repeat_password.$touch()"
      @blur="$v.repeat_password.$touch()"
    ></v-text-field>
  </v-form>
</v-card-text>
```

```
          <v-alert v-model="error.show" type="error" dismissible>
            {{error.message}}
          </v-alert>
          <v-card-actions>
            <v-spacer><v-spacer><v-btn outline @click="transfer('login')" color="indigo"><v-icon dark left>arrow_back</v-icon>Back to login</v-btn></v-spacer></v-spacer>
            <v-btn  color="primary" @click="submit">Sign Up</v-btn>
          </v-card-actions>
        </v-card>
      </v-flex>
    </v-layout>
  </v-container>
 </v-content>
</v-app>
</template>

<script>
 import { validationMixin } from 'vuelidate'
 import { required, minLength, email, sameAs } from 'vuelidate/lib/validators'

 export default {
  mixins: [validationMixin],
  data: () => ({
   email: '',
   password: '',
   repeat_password: '',
   show_pass: true,
   show_second_pass: true,
   error: {
    show: false,
    message: ''
   }
```

```
}),
validations: {
 email: { required, email },
 password: { required, minLength: minLength(8) },
 repeat_password: { sameAsPassword: sameAs('password') }
},
 computed: {
 password_errors () {
  const errors = []
  if (!this.$v.password.$dirty) return errors
  !this.$v.password.minLength && errors.push('Password must be at least 8 characters long')
  !this.$v.password.required && errors.push('Password is required')
  return errors
 },
 email_errors () {
  const errors = []
  if (!this.$v.email.$dirty) return errors
  !this.$v.email.email && errors.push('Must be valid e-mail')
  !this.$v.email.required && errors.push('E-mail is required')
  return errors
 },
 second_password_errors () {
  const errors = []
  if (!this.$v.repeat_password.$dirty) return errors
  !this.$v.repeat_password.sameAsPassword && errors.push('Passwords must be identical.')
  return errors
 }
},
methods: {
 submit: function () {
  this.$v.$touch();
  if (this.$v.$invalid) {
```

```
        return false;
      }
      const payload = {
        email: this.email,
        password: this.password,
        password_again: this.repeat_password
      }

      this.$store.dispatch('SIGNUP_REQUEST', payload).then(() => {
         this.$router.push('/admin')
      }).catch((e) => {
        this.error.show = true;
        if (e.response) {
          this.error.message = e.response.data.message;
        } else {
          this.error.message = e.message;
        }
      })
    },
    transfer: function (route) {
      this.$router.push('/' + route);
    }
  }
 }
</script>
```

f) Vuex Store для авторизації

```
import userService from '../../services/UserService'

const AUTH_REQUEST = "AUTH_REQUEST",
  SIGNUP_REQUEST = "SIGNUP_REQUEST",
  AUTH_CHECK_REQUEST = "AUTH_CHECK_REQUEST",
  AUTH_SUCCESS = "AUTH_SUCCESS",
```

```
  AUTH_ERROR = "AUTH_ERROR",
  AUTH_LOGOUT = "AUTH_LOGOUT";

const mutations = {
  [AUTH_REQUEST]: (state) => {
    state.status = 'loading'
  },
  [AUTH_SUCCESS]: (state, token) => {
    state.status = 'success'
    state.token = token
  },
  [AUTH_ERROR]: (state) => {
    state.status = 'error'
  },
  [AUTH_LOGOUT]: (state) => {
    state.status = 'logged_out',
    state.token = ''
  },
}

const actions = {
  [AUTH_REQUEST]: ({
    commit,
    dispatch
  }, user) => {
    return new Promise((resolve, reject) => { // The Promise used for router redirect in login
      commit(AUTH_REQUEST)
      userService.login(user)
        .then(resp => {
          const token = resp.data.token;
          if(!token){
            return reject(resp.data);
```

```
      }
      localStorage.setItem('x-auth', token) // store the token in localstorage
      commit(AUTH_SUCCESS, token)
      resolve(resp)
    })
    .catch(err => {
      commit(AUTH_ERROR, err)
      localStorage.removeItem('x-auth') // if the request fails, remove any possible user token if possible
      reject(err)
    })
 })
},
[AUTH_CHECK_REQUEST]: ({
 commit,
 dispatch
}) => {
 return new Promise((resolve, reject) => { // The Promise used for router redirect in login
  commit(AUTH_REQUEST);
  var token = localStorage.getItem('x-auth')
  userService.check(token)
   .then(resp => {
    commit(AUTH_SUCCESS, localStorage.getItem('x-auth'))
    resolve(resp)
   })
   .catch(err => {
    commit(AUTH_ERROR, err)
    localStorage.removeItem('x-auth') // if the request fails, remove any possible user token if possible
    reject(err)
   })
 })
},
[SIGNUP_REQUEST]: ({
```

```
  commit,
  dispatch
}, user) => {
  return new Promise((resolve, reject) => { // The Promise used for router redirect in login
    commit(AUTH_REQUEST)
    userService.sign_up(user)
      .then(resp => {
        const token = resp.data.token;
        localStorage.setItem('x-auth', token) // store the token in localstorage
        commit(AUTH_SUCCESS, token)
        // you have your token, now log in your user :)
        //dispatch(USER_REQUEST)
        resolve(resp)
      })
      .catch(err => {
        commit(AUTH_ERROR, err)
        localStorage.removeItem('x-auth') // if the request fails, remove any possible user token if possible
        reject(err)
      })
  })
},
[AUTH_LOGOUT]: ({
  commit,
  dispatch
}) => {
  return new Promise((resolve, reject) => {
    var token = localStorage.getItem('x-auth')
    userService.logout(token)
    .then(resp => {
      localStorage.removeItem('x-auth') // clear your user's token from localstorage
      commit(AUTH_LOGOUT)
      resolve()
```

```
    })
   })
  }
}

const getters = {
  isAuthenticated: state => !!state.token,
  authStatus: state => state.status,
}

export default {
  state: {
    token: localStorage.getItem('x-auth') || '',
    status: ''
  },
  mutations,
  actions,
  getters
}
```

g) Налаштування API сервісу

```
import axios from 'axios'
//const token = localStorage.getItem('admin-token') || '';

export default (token) => {
  return axios.create({
    baseURL: 'http://localhost:3000',
    validateStatus: function (status) {
      return status <= 402;
    },
    headers: {'x-auth': token || ''},
    proxy: {
      host: 'http://localhost',
```

```
    port: 3000
  }
 });
}
```

g) User API service

```
import api from './api'
export default {
 sign_up(payload) {
  return api().post('users', payload);
 },
 check(token) {
  return api(token).get('/users/me');
 },
 login(payload) {
  return api().post('/users/login', payload);
 },
 logout(token) {
  return api(token).delete('/users/me/token');
 }
}
```

h) Налаштування роутингу

```
import Vue from 'vue'
import Router from 'vue-router'
import Auth from '../components/Login'
import Signup from '../components/Signup'
import Election from '../components/Election'
// Start Admin
import Admin from '../components/Admin'
import admin_candidate from '../components/admin/admin_candidate'
import admin_election from '../components/admin/admin_election'
import admin_electorate from '../components/admin/admin_electorate'
// End Admin
```

```
import store from '../store'
Vue.use(Router)

const ifNotAuthenticated = (to, from, next) => {
  if (!store.getters.isAuthenticated) {
    next()
    return
  }
  next('/admin')
}

const ifAuthenticated = (to, from, next) => {
  if (store.getters.isAuthenticated) {
    next()
    return
  }
  next('/login')
}

export default new Router({
  mode: 'history',
  routes: [{
      path: '/',
      name: 'Election',
      component: Election
    },
    {
      path: '/admin',
      name: 'Admin',
      component: Admin,
      beforeEnter: ifAuthenticated,
      redirect: to => ({
```

```
    name: 'election',
    params: {
      section: 'election'
    }
  }),
  children: [{

      path: 'candidate',
      component: admin_candidate,
      beforeEnter: ifAuthenticated
    },
    {
      name: 'election',
      path: 'election',
      component: admin_election,
      beforeEnter: ifAuthenticated
    },
    {
      path: 'electorate',
      component: admin_electorate,
      beforeEnter: ifAuthenticated
    }
  ]
},
{
  path: '/login',
  name: 'Login',
  component: Auth,
  beforeEnter: ifNotAuthenticated
},
{
  path: '/login',
```

```
    name: 'Login',
    beforeEnter: ifAuthenticated
  },
  {
    path: '/signup',
    name: 'Signup',
    component: Signup,
    beforeEnter: ifNotAuthenticated
  }
 ]
})
```

3. Авторизація за технологією блокчейн

```
pragma solidity ^0.4.4;

/*
ERROR CODES

Error 1 = NO_PERMISSION
Error 2 = ALREADY_VOTED
Error 3 = INVALID_TOKEN
Error 99 = WRONG_STAGE
*/

contract Election {

// ############### EVENTS ###############

  event error(uint);
  event log(string);
  event voteEvent(uint indexed _currentRound, string _token, uint _nrVotes);
  event resultPublishedEvent(uint indexed _currentRound, string _result);
```

```
    event electionStatusEvent(uint indexed _currentRound, uint _status);

// ############## STRUCTS ##############

    struct Vote {
        address addr;
        string token;
        string candidate;
    }

    enum Stage {
        PRE_VOTING,
        VOTING,
        PROCESSING,
        RESULT
    }

// ############## FIELDS ##############

    // contract owner becomes admin
    address public admin;
    Stage public currentStage = Stage.PRE_VOTING;

    // name of the election, e.g. "BP 2016"
    string public name;

    uint public currentRound = 1; // incremented for every election round
    Vote[] public votes;
    uint public nrVotes = 0;

    string public lastResult = "";
    string public privateKey = "";
```

```
// ############### PUBLIC FUNCTIONS ###############

    // Constructor of the contract
    function Election(string _name) {
        admin = msg.sender;
        name = _name;
    }

    function reset() requiresAdmin {
        currentStage = Stage.PRE_VOTING;
        votes.length = 0;
        nrVotes = 0;
        lastResult = "";
        privateKey = "";
    }

    function startElection() requiresAdmin preVoting {
        currentStage = Stage.VOTING;
    }

    function vote(string _token, string _candidate) returns(uint) {
        if(currentStage != Stage.VOTING) { error(99); return 99; } // WRONG_STAGE
        if(alreadyVoted(_token)) { error(2); return 2; } // ALREADY_VOTED
        if(! isTokenValid(_token)) { error(3); return 3; } // INVALID_TOKEN

        if(nrVotes == votes.length) {
            votes.length += 1;
        }
        votes[nrVotes++] = Vote({
            addr: msg.sender,
            token: _token,
```

```
        candidate: _candidate
    });

    voteEvent(currentRound, _token, nrVotes);
    return 0;
}

function getNrVotes() returns(uint) {
    return nrVotes;
}

function getCurrentElectionRoundAndStatus() returns(uint round, uint status) {
    return (currentRound, uint(currentStage));
}

function stopElection() requiresAdmin voting {
    currentStage = Stage.PROCESSING;
    electionStatusEvent(currentRound, uint(currentStage));
}

function publishResult(string _result, string _privateKey) requiresAdmin {
    lastResult = _result;
    privateKey = _privateKey;

    resultPublishedEvent(currentRound, _result);
    currentStage = Stage.RESULT;
    electionStatusEvent(currentRound, uint(currentStage));
}

function startNewRound() requiresAdmin {
    currentStage = Stage.VOTING;
    // votes.length = 0; // disabled because it easily runs out of gas
```

```
        nrVotes = 0;
        currentRound++;
        electionStatusEvent(currentRound, uint(currentStage));
    }
// ############### MODIFIERS ###############
modifier requiresAdmin {
    if(msg.sender != admin) throw;
    _;
}
modifier preVoting {
    if(currentStage != Stage.PRE_VOTING) throw;
    _;
}
modifier voting {
    if(currentStage != Stage.VOTING) throw;
    _;
}
// ############### PRIVATE FUNCTIONS ###############
    function alreadyVoted(string _token) returns(bool) {
        for(var i=0; i<nrVotes; i++) {
            if(compareStrings(votes[i].token, _token) == 0) {
                return true;
            }
        }
        return false;
    }

    function isTokenValid(string _token) returns (bool) {
        return true;
    }

    function compareStrings(string _a, string _b) returns (int) {
```

```
        bytes memory a = bytes(_a);
        bytes memory b = bytes(_b);
        uint minLength = a.length;
        if (b.length < minLength) minLength = b.length;
        for (uint i = 0; i < minLength; i ++)
            if (a[i] < b[i])
                return -1;
            else if (a[i] > b[i])
                return 1;
        if (a.length < b.length)
            return -1;
        else if (a.length > b.length)
            return 1;
        else
            return 0;
    }
}

export class Crypto {
    pubKey = `-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtKwVvZBCxI24bfuKg5yU
g5vY+vm8nhukVdU400w5DjmiolMFrcCyDh2IjEjL23fXO8J5M9SMKA9QOJ9zLKGO
bzIdwdZPLc8F+p5UkA25kyLL46BSKkbrhNRYe+yG544qElbZcRhnhuc6WSCq6T0m
4rQMtnpfHZfSyvx4YUIM6pUgT/Pxehcj9hj45ppsi3QLGnFP6vHCDzeGnJt8QIdJ
hTSlSIROvkcH+aa6sj7Ekc5oNt7KHbzh+ZnWd7jHSYyGOCVRp/fuIUvENcOuvLP1
zFtvb8KLJ8fzQZNImzBOxo2tsGNR8PdeNtFb1iosrI+5XPKLBt89acLyGpYJ4OXz
WwIDAQAB
-----END PUBLIC KEY-----`

    constructor() {
        if(! crypto.subtle) {
            console.error('Crypto API not supported!')
```

```
    } else {
      var self = this
      this.importPublicKey(this.pubKey).then((key) => {
        this.importedPubKey = key
        console.log('public key imported')
      })
    }
  }

  encryptionPromise(cleartext) {
    if(! this.importedPubKey) {
      console.error('no pubkey imported')
      return null
    }

    var promise = crypto.subtle.encrypt(this.encryptAlgorithm, this.importedPubKey,
this.textToArrayBuffer(cleartext))
    return promise
  }



  crypto = window.crypto || window.msCrypto
  encryptAlgorithm = {
    name: "RSA-OAEP",
    hash: {
      name: "SHA-1"
    }
  };

  arrayBufferToBase64String(arrayBuffer) {
    var byteArray = new Uint8Array(arrayBuffer)
    var byteString = ''
```

```
    for (var i = 0; i < byteArray.byteLength; i++) {
        byteString += String.fromCharCode(byteArray[i])
    }
    return btoa(byteString)
}

base64StringToArrayBuffer(b64str) {
    var byteStr = atob(b64str)
    var bytes = new Uint8Array(byteStr.length)
    for (var i = 0; i < byteStr.length; i++) {
        bytes[i] = byteStr.charCodeAt(i)
    }
    return bytes.buffer
}

textToArrayBuffer(str) {
    var buf = unescape(encodeURIComponent(str))
    var bufView = new Uint8Array(buf.length)
    for (var i = 0; i < buf.length; i++) {
        bufView[i] = buf.charCodeAt(i)
    }
    return bufView
}

convertPemToBinary(pem) {
    var lines = pem.split('\n')
    var encoded = ''
    for (var i = 0; i < lines.length; i++) {
        if (lines[i].trim().length > 0 &&
            lines[i].indexOf('-BEGIN RSA PRIVATE KEY-') < 0 &&
            lines[i].indexOf('-BEGIN RSA PUBLIC KEY-') < 0 &&
            lines[i].indexOf('-BEGIN PUBLIC KEY-') < 0 &&
```

```
            lines[i].indexOf('-END PUBLIC KEY-') < 0 &&
            lines[i].indexOf('-END RSA PRIVATE KEY-') < 0 &&
            lines[i].indexOf('-END RSA PUBLIC KEY-') < 0) {
            encoded += lines[i].trim()
          }
        }
        return this.base64StringToArrayBuffer(encoded)
    }

    importPublicKey(pemKey) {
        var self = this
        return new Promise( (resolve) => {
          var importer = crypto.subtle.importKey("spki", self.convertPemToBinary(pemKey),
self.encryptAlgorithm, false, ["encrypt"])
          importer.then(function (key) {
            resolve(key)
          })
        })
    }
}

export class Crypto {
    pubKey = `-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtKwVvZBCxI24bfuKg5yU
g5vY+vm8nhukVdU400w5DjmiolMFrcCyDh2IjEjL23fXO8J5M9SMKA9QOJ9zLKGO
bzIdwdZPLc8F+p5UkA25kyLL46BSKkbrhNRYe+yG544qElbZcRhnhuc6WSCq6T0m
4rQMtnpfHZfSyvx4YUIM6pUgT/Pxehcj9hj45ppsi3QLGnFP6vHCDzeGnJt8QIdJ
hTSlSIROvkcH+aa6sj7Ekc5oNt7KHbzh+ZnWd7jHSYyGOCVRp/fuIUvENcOuvLP1
zFtvb8KLJ8fzQZNImzBOxo2tsGNR8PdeNtFb1iosrI+5XPKLBt89acLyGpYJ4OXz
WwIDAQAB
-----END PUBLIC KEY-----`
```

```
  constructor() {
    if(! crypto.subtle) {
      console.error('Crypto API not supported!')
    } else {
      var self = this
      this.importPublicKey(this.pubKey).then((key) => {
        this.importedPubKey = key
        console.log('public key imported')
      })
    }
  }

  encryptionPromise(cleartext) {
    if(! this.importedPubKey) {
      console.error('no pubkey imported')
      return null
    }

    var promise = crypto.subtle.encrypt(this.encryptAlgorithm, this.importedPubKey,
this.textToArrayBuffer(cleartext))
    return promise
  }



  crypto = window.crypto || window.msCrypto
  encryptAlgorithm = {
    name: "RSA-OAEP",
    hash: {
      name: "SHA-1"
    }
  };
```

```
arrayBufferToBase64String(arrayBuffer) {
   var byteArray = new Uint8Array(arrayBuffer)
   var byteString = ''
   for (var i = 0; i < byteArray.byteLength; i++) {
      byteString += String.fromCharCode(byteArray[i])
   }
   return btoa(byteString)
}

base64StringToArrayBuffer(b64str) {
   var byteStr = atob(b64str)
   var bytes = new Uint8Array(byteStr.length)
   for (var i = 0; i < byteStr.length; i++) {
      bytes[i] = byteStr.charCodeAt(i)
   }
   return bytes.buffer
}

textToArrayBuffer(str) {
   var buf = unescape(encodeURIComponent(str))
   var bufView = new Uint8Array(buf.length)
   for (var i = 0; i < buf.length; i++) {
      bufView[i] = buf.charCodeAt(i)
   }
   return bufView
}

convertPemToBinary(pem) {
   var lines = pem.split('\n')
   var encoded = ''
   for (var i = 0; i < lines.length; i++) {
      if (lines[i].trim().length > 0 &&
```

```
            lines[i].indexOf('-BEGIN RSA PRIVATE KEY-') < 0 &&
            lines[i].indexOf('-BEGIN RSA PUBLIC KEY-') < 0 &&
            lines[i].indexOf('-BEGIN PUBLIC KEY-') < 0 &&
            lines[i].indexOf('-END PUBLIC KEY-') < 0 &&
            lines[i].indexOf('-END RSA PRIVATE KEY-') < 0 &&
            lines[i].indexOf('-END RSA PUBLIC KEY-') < 0) {
            encoded += lines[i].trim()
        }
      }
      return this.base64StringToArrayBuffer(encoded)
    }

    importPublicKey(pemKey) {
      var self = this
      return new Promise( (resolve) => {
        var importer = crypto.subtle.importKey("spki", self.convertPemToBinary(pemKey),
self.encryptAlgorithm, false, ["encrypt"])
        importer.then(function (key) {
          resolve(key)
        })
      })
    }
}
```

Додаток В. Ілюстративний матеріал

ІЛЮСТРАТИВНИЙ МАТЕРІАЛ ДО ЗАХИСТУ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Завідувач кафедри ПЗ, д. т. н., професор _____ О. Н. Романюк

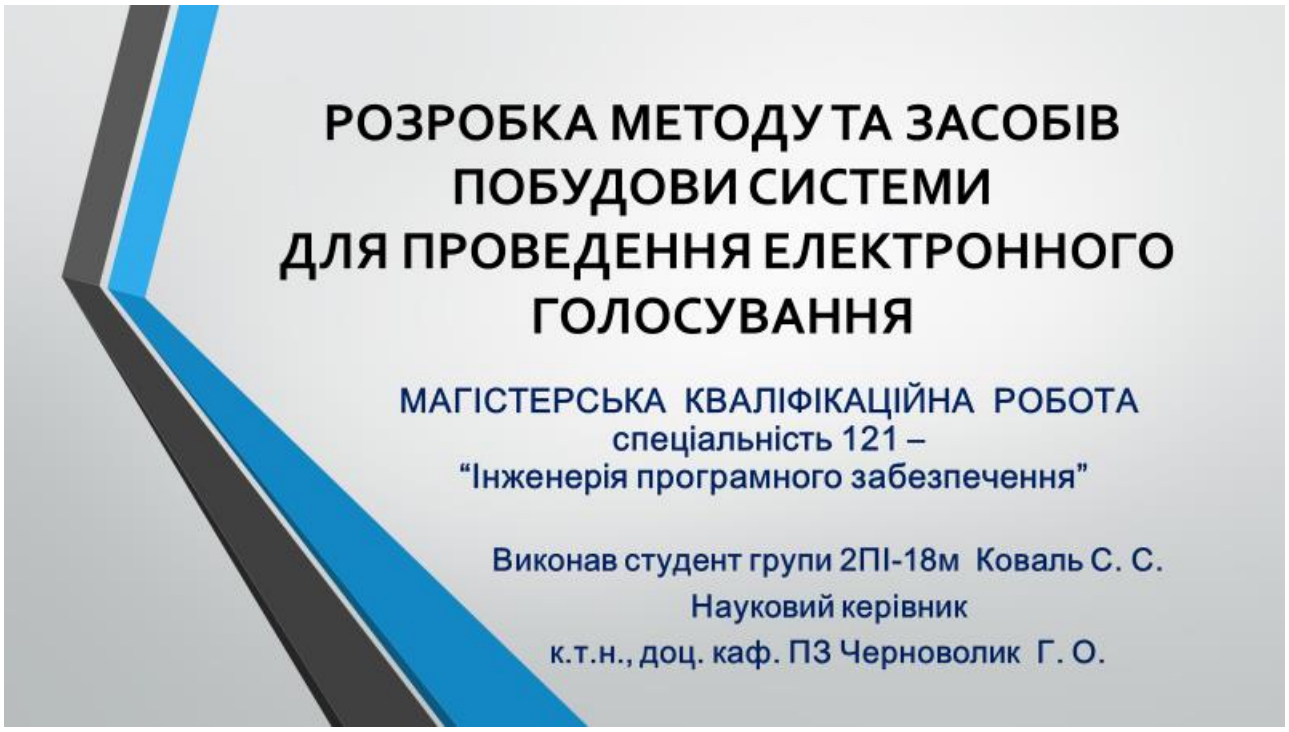
Науковий керівник, к. т. н., доцент кафедри ПЗ_____ Г. О. Черноволик

Рецензент, к. т. н., доцент кафедри КН_____ І. Р. Арсенюк

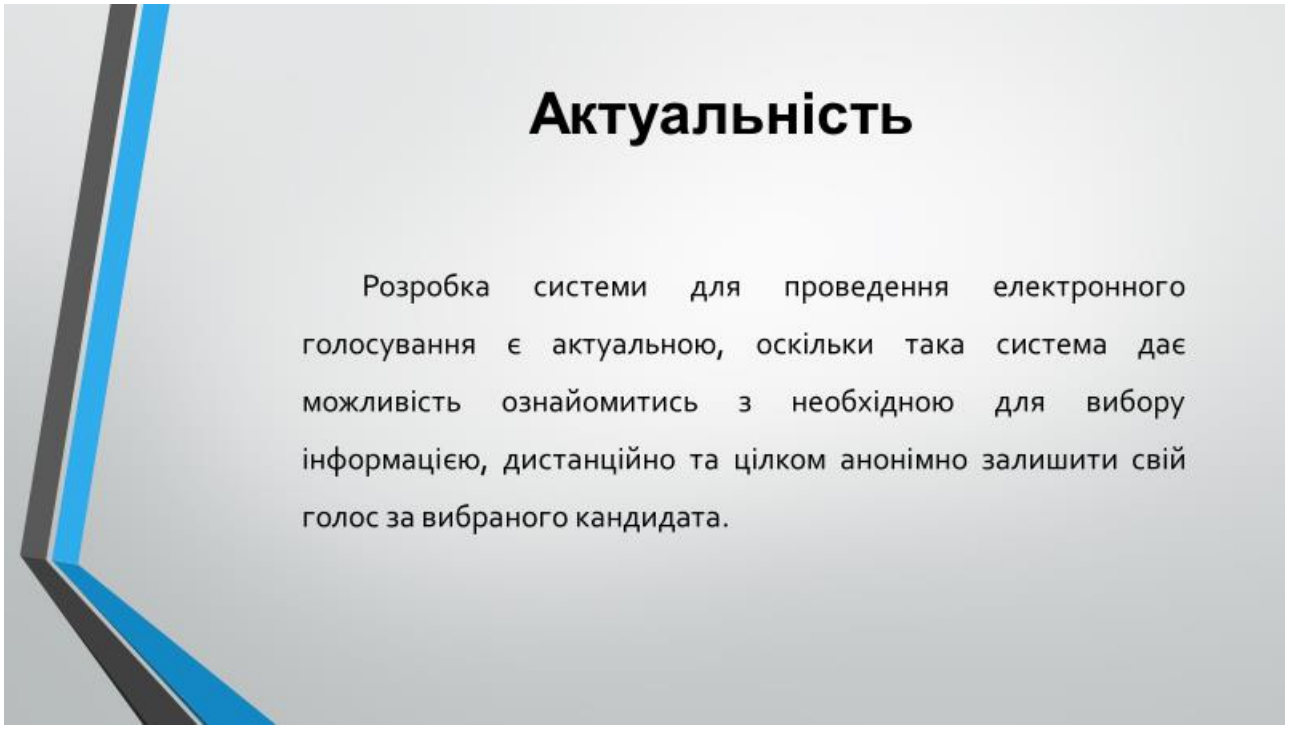
Нормоконтроль, к. т. н., доцент кафедри ПЗ _____ Г. О. Черноволик

Виконавець, студент групи 2ПІ-18м _____ С. С. Коваль

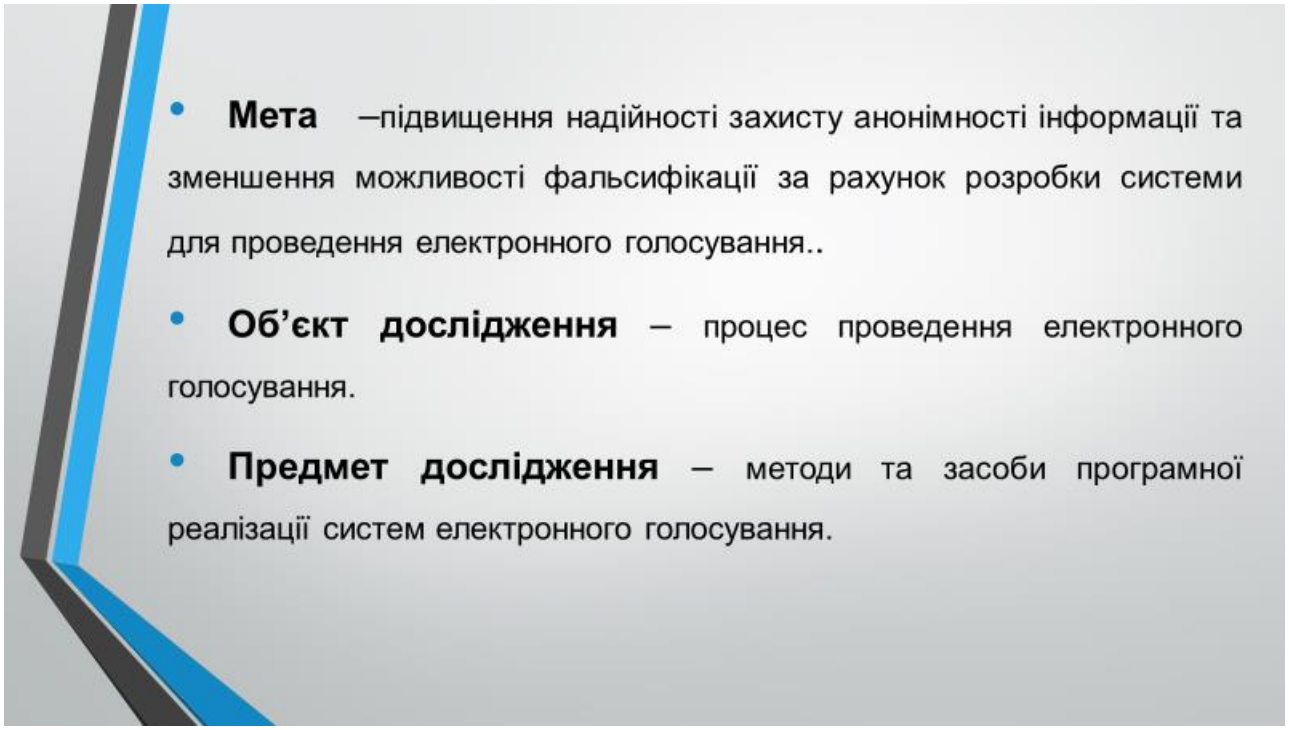
Слайд 1 – тема роботи



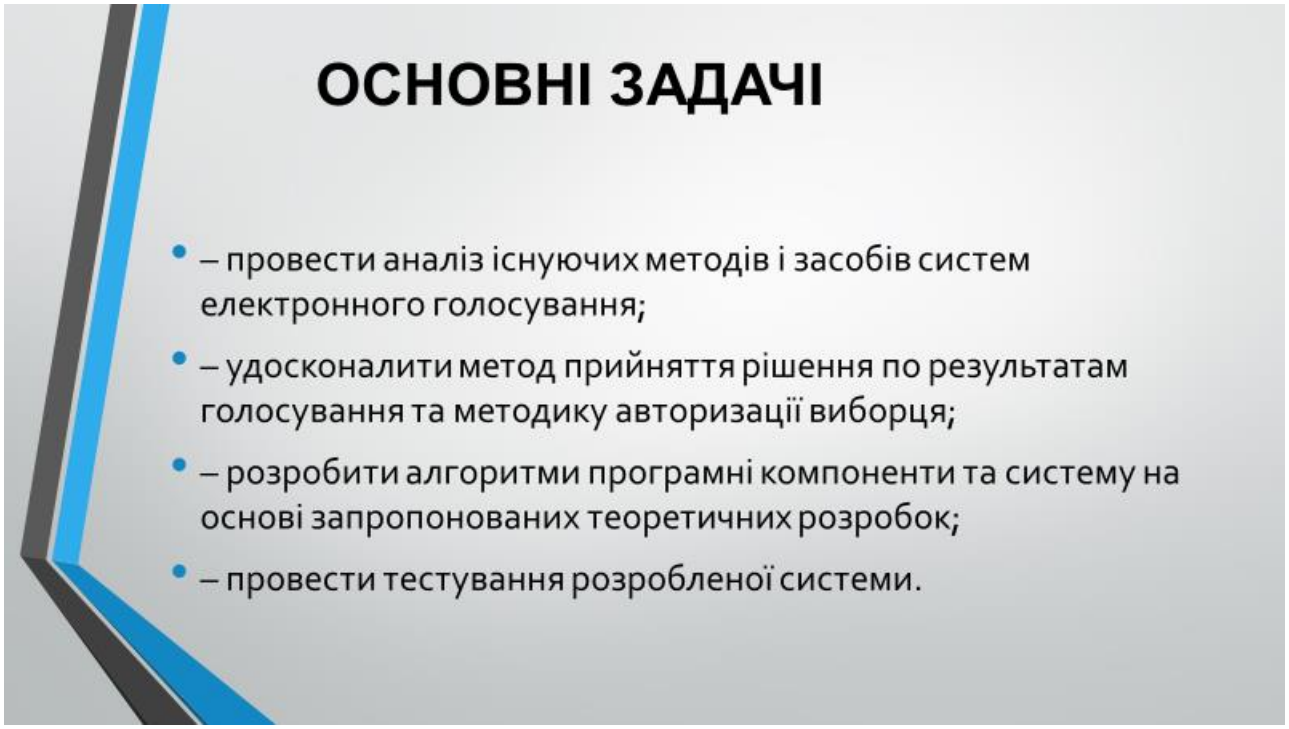
Слайд 2 – Актуальність дослідження



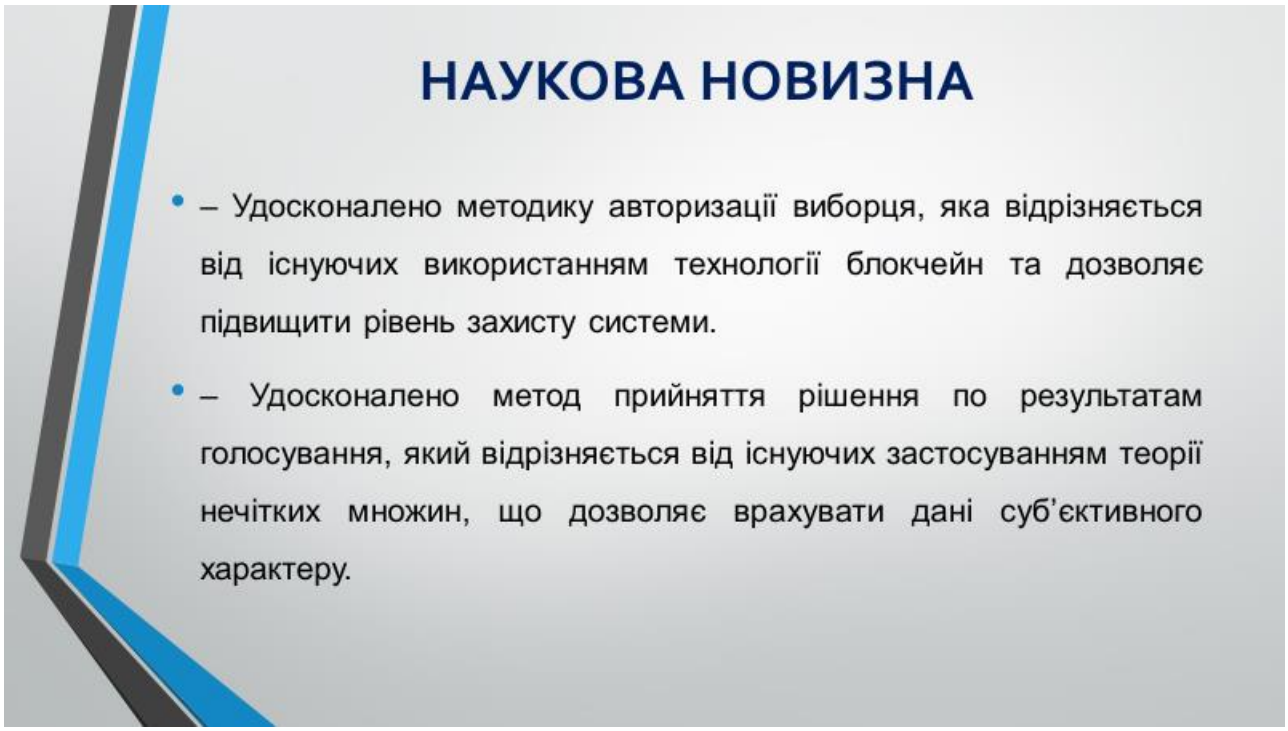
Слайд 3 – Мета, об’єкт, предмет дослідження



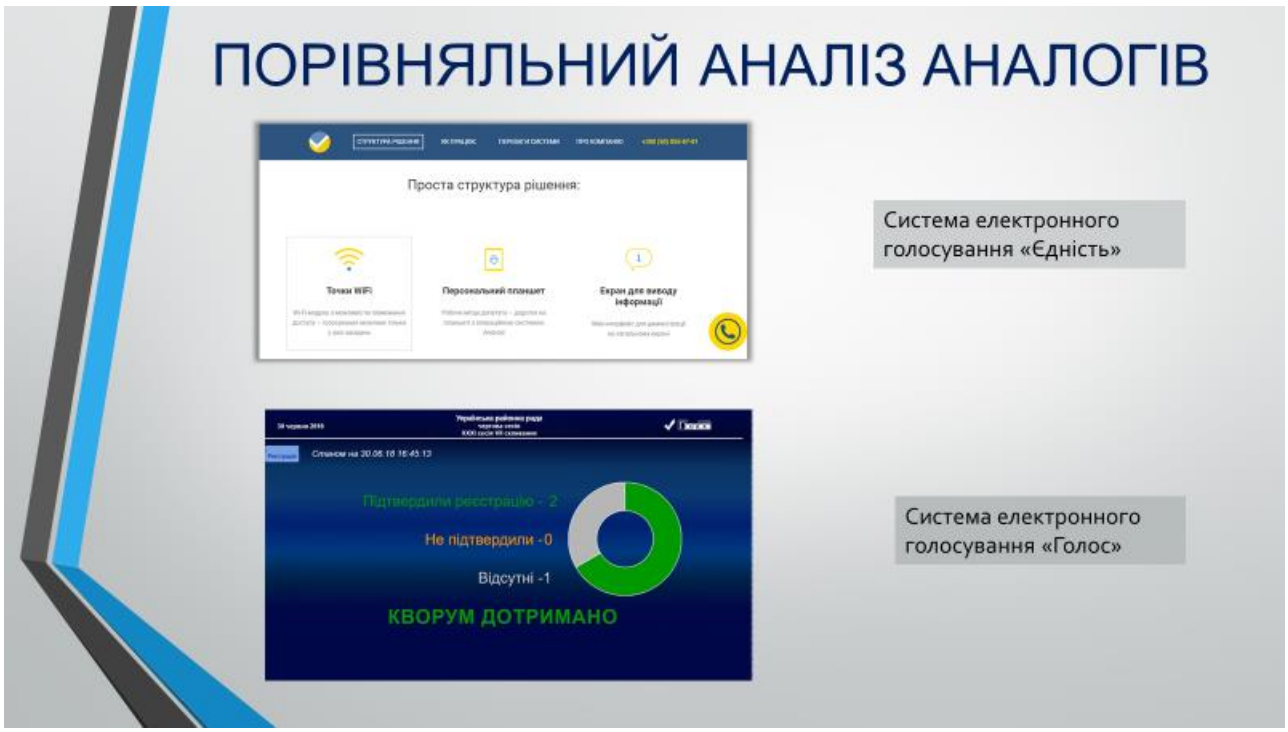
Слайд 4 – Основні задачі дослідження



Слайд 5 – Наукова новизна



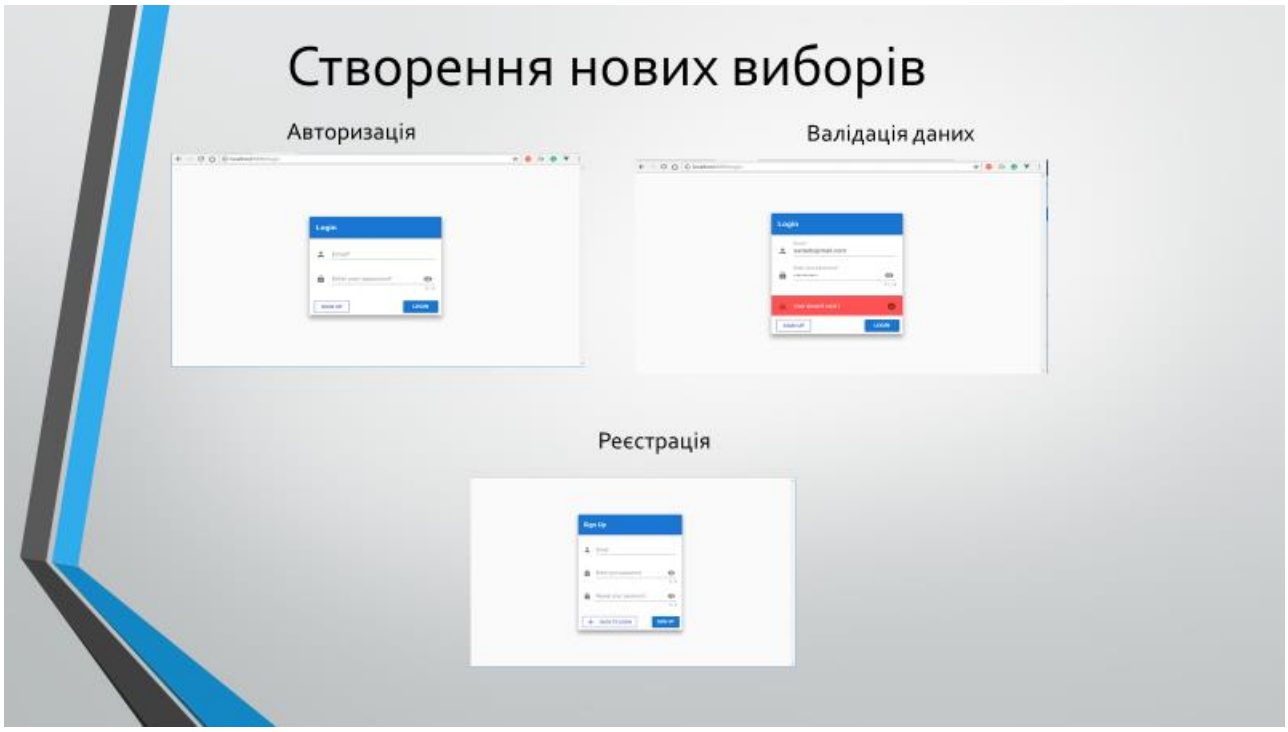
Слайд 6 – Порівняльний аналіз аналогів



Слайд 7 – Структура адміністративної частини

Слайд 8 –Структура клієнтської частини

Слайд 9 – Створення нових виборів



Слайд 10 – Висновки