Вищий навчальний заклад

«Університет економіки та права «КРОК»
Фаховий коледж

Циклова комісія з інформаційних технологій

Кваліфікаційна робота фахового молодшого бакалавра

на тему Інформаційна система дистанційного голосування

Виконав _____
(Підпис)

__Воскобоєв_Андрій_Олександрович_____
(прізвище, ім'я, по батькові)

_____

Науковий керівник
____Кириченко _Віктор_Вікторович___
(прізвище, ім'я, по батькові)

_____

_____
(Резолюція «До захисту»)

Попередній захист:

_____
(Висновок: "До захисту в екзаменаційній комісії ")

Голова циклової комісії _____ _____ _____
(Підпис) (Прізвище, ініціали) (Дата)

Київ – 2025 року

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

Фаховий коледж

Циклова комісія з інформаційних технологій

Спеціальність 121 інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Голова циклової комісії _____ Леонід УВАРОВ

(підпис)

«____» _____ 2025 року

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувач освіти **Воскобоєв Андрій Олександрович**

1. **Тема роботи: Інформаційна система дистанційного голосування** затверджена наказом по університету від «_» **_____ 202** р. № _____
2. **Термін здачі закінченої роботи:** «30» травня 2025 року
3. **Вихідні дані до роботи:**
    1. **Цільова аудиторія:** Користувачі, що потребують проведення або участі в дистанційних опитуваннях та голосуваннях у закритих групах (наприклад, колективи, невеликі спільноти, освітні заклади для внутрішніх потреб).
    2. **Функціональність:**
        - Реєстрація та авторизація користувачів (з ролями "Користувач" та "Адміністратор").

- Керування опитуваннями (створення, активація/завершення, видалення) для адміністратора.
- Участь у голосуванні для зареєстрованих користувачів.
- Фіксація голосів та запобігання повторному голосуванню одного користувача в одному опитуванні.
- Відображення узагальнених результатів голосування.

3. **Технічні вимоги:**
    - Клієнтська веб-розробка: HTML5, CSS3, JavaScript (ES6+).
    - Зберігання даних: localStorage браузера.
    - Архітектура: Односторінковий веб-додаток (SPA).
    - Безпека: Базове хешування паролів (MD5 для демонстрації).
4. **Можливість подальшого розвитку:** Перехід на сервер-клієнт архітектуру, інтеграція з базою даних, розширення функціоналу опитувань.
5. **Існуючі рішення:** Аналіз сервісів для онлайн-опитувань та голосувань (наприклад, Google Forms, SurveyMonkey та ін.).

4. **Зміст пояснювальної записки:**
    1. **Розділ 1. Теоретичні основи та аналіз предметної області.**
        - Визначення та класифікація інформаційних систем дистанційного голосування.

- Огляд сучасних веб-технологій для клієнтської розробки (HTML5, CSS3, JavaScript) та їх можливостей.
- Аналіз існуючих рішень для дистанційного голосування, їх переваг та недоліків.
- Обґрунтування вибору архітектури односторінкового веб-додатку та використання localStorage для поточного проекту.

2. **Розділ 2. Проєктування інформаційної системи дистанційного голосування.**
    - Визначення функціональних та нефункціональних вимог до системи.
    - Проєктування архітектури системи та основних компонентів.
    - Моделювання варіантів використання системи.
    - Проектування структури зберігання даних в localStorage (схеми користувачів та опитувань).
    - Проєктування користувацьких інтерфейсів (UI/UX) для різних ролей.
3. **Розділ 3. Реалізація інформаційної системи дистанційного голосування.**
    - Загальна характеристика програмної реалізації, використані технології та інструменти.

- Опис основних програмних модулів та їх функціоналу (модуль керування даними, модуль автентифікації/авторизації, модуль керування опитуваннями, модуль голосування та відображення результатів).
- Детальний опис алгоритмів роботи ключових компонентів (наприклад, алгоритм реєстрації, алгоритм голосування, алгоритм розрахунку результатів).
- Тестування та налагодження роботи платформи, виявлені недоліки та шляхи їх усунення.

5. **Перелік графічного матеріалу:**
   - Діаграми.
   - Скріншоти інтерфейсу розробленої системи (екрани авторизації, панелі користувача/адміністратора, сторінки голосування, результатів тощо).
   - Порівняльні таблиці аналізу існуючих рішень.

Дата видачі завдання «15» лютого 2025 року

Науковий керівник _____ ___Кириченко В. В.

(підпис) (прізвище, ім'я, по батькові)

Завдання прийняла до виконання _____ ___Воскобоєв А. О.

(підпис) (прізвище, ім'я, по батькові)

РЕФЕРАТ

Пояснювальна записка: 77 сторінок, 10 рисунків, 1 таблиць, 6 додатків, 17 джерел.

Об'єкт дослідження – інформаційна система дистанційного голосування.

Мета роботи – Розробка інтуїтивно зрозумілої веборієнтованої системи для електронного голосування з локальним збереженням даних та базовою авторизацією.

У кваліфікаційній роботі проведено аналіз сучасних електронних систем голосування, визначено вимоги до дистанційного волевиявлення. Розроблено веборієнтовану інформаційну систему, що дозволяє створювати та проводити онлайн-голосування з поділом на ролі користувачів. Система реалізована як вебзастосунок на HTML, CSS та JavaScript, дані зберігаються локально в localStorage браузера у форматі JSON. Підтримано реєстрацію, автентифікацію, адміністрування та відображення результатів голосувань. Результати можуть бути використані в навчальних закладах або організаціях, що потребують простої системи електронного голосування.

Ключові слова: електронне голосування, вебзастосунок, JavaScript, HTML, CSS, локальне збереження, авторизація, адміністрування, JSON, користувацький інтерфейс.

ABSTRACT

Explanatory Note: 77 pages, 10 figures, 1 tables, 6 appendices, 17 sources.

Object of Study: Information system for remote voting.

Goal of the Work: Development of an intuitive web-oriented system allowing electronic voting with local data storage and basic authentication.

This qualification paper analyzes modern electronic voting systems, identifies key requirements for remote voting, and presents a developed web-oriented information system. The system enables online poll creation and conduct with user role

differentiation. Implemented as a web application using HTML, CSS, and JavaScript, it stores data locally in the browser's localStorage in JSON format. The system supports user registration and authentication, poll administration, and the processing and display of voting results. A dedicated administrator interface is provided for creating, closing, and deleting polls. The development can be utilized in educational or public organizations requiring a simple electronic voting system.

Keywords: electronic voting, web application, JavaScript, HTML, CSS, local storage, authentication, administration, JSON, user interface.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

1. **API** — Application Programming Interface (Інтерфейс програмування застосунків)
2. **CSS** — Cascading Style Sheets (Каскадні таблиці стилів)
3. **CRUD** — Create, Read, Update, Delete (Створити, Прочитати, Оновити, Видалити)
4. **DOM** — Document Object Model (Об'єктна модель документа)
5. **ДСТУ** — Державний стандарт України
6. **ES6+** — ECMAScript 2015 (6-та версія стандарту JavaScript) і новіші
7. **HTML** — HyperText Markup Language (Мова гіпертекстової розмітки)
8. **HTTP** — HyperText Transfer Protocol (Протокол передачі гіпертексту)
9. **ID** — Identifier (Ідентифікатор)
10. **JS** — JavaScript (Мова програмування JavaScript)
11. **JSON** — JavaScript Object Notation (Об'єктна нотація JavaScript)
12. **MD5** — Message-Digest Algorithm 5 (Алгоритм хешування повідомлень версії 5)
13. **SPA** — Single Page Application (Односторінковий веб-застосунок)
14. **UI** — User Interface (Користувацький інтерфейс)
15. **URL** — Uniform Resource Locator (Уніфікований покажчик ресурсу)
16. **UX** — User Experience (Користувацький досвід)

ВСТУП

Актуальність завдання. Електронне голосування - термін, який охоплює декілька різновидів застосування електронних засобів для безпосередньо голосування та/або підрахунку голосів. Зазвичай це можуть бути електронні машини для голосуваннячиінші електронні способи реєстрації голосів на виборчій дільниці. Онлайн-голосування або Інтернет-голосування - це версії електронного голосування, де виборці маютьзмогувіддати свій голос, не відвідуючи нічого, використовуючи комп'ютер, планшет або смартфон. Як і інші підходи до електронного голосування, ці системи також полегшують підрахунок голосів. Дистанційне голосування є важливим у сучасному світі через зростання діджиталізації, потребу в швидких, прозорих і доступних опитуваннях. Воно дозволяє людям голосувати з будь-якого місця, економить ресурси, забезпечує оперативний підрахунок голосів і може гарантувати безпеку та конфіденційність при правильному впровадженні. Такі системи актуальні для виборів, корпоративних рішень, роботи громадських організацій, освітніх установ і децентралізованих спільнот.

Мета роботи. Розробити веб-додаток для створення та проведення дистанційних голосувань з підрахунком голосів під час і після завершення голосування.

Завдання роботи. Завданням роботи є проведення аналізу існуючих аналогічних рішень у сфері систем дистанційного голосування та опитувань, а також визначення їхніх переваг і недоліків; Проєктування логічної структури зберігання даних для інформації про користувачів та опитування, з обґрунтуванням вибору механізму клієнтського зберігання даних (localStorage) для реалізації функціональності демонстраційного прототипу; Розробка механізму реєстрації та авторизації користувача й адміністратора, включаючи імітацію хешування паролів для забезпечення базової безпеки облікових записів.; Реалізація функціоналу обробки голосів користувачів та відображення узагальнених результатів голосування адміністраторові, із візуалізацією даних;

Впровадження можливості створення нових опитувань адміністратором, а також управління активними та завершеними опитуваннями (включаючи їхнє завершення та видалення); Проведення тестування та налагодження роботи клієнтської платформи для забезпечення її стабільної та ефективної роботи в межах архітектури.

Об’єкт дослідження. Процеси дистанційного голосування та інформаційні системи, що забезпечують їхню реалізацію.

Предмет дослідження. Методи та засоби розробки клієнтської інформаційної системи дистанційного голосування на основі сучасних веб-технологій.

Методи дослідження. Під час виконання роботи були використані такі методи: системний аналіз для вивчення вимог до системи; об'єктно-орієнтоване проектування для побудови логічної архітектури; програмна реалізація з використанням мови JavaScript та технологій HTML5, CSS3; функціональне та інтеграційне тестування для перевірки працездатності системи.

Практичне значення одержаних результатів. Розроблена система має практичне значення як демонстраційний прототип для проведення внутрішніх опитувань у невеликих колективах, освітніх установах або для навчальних цілей. Вона демонструє можливості створення інтерактивних веб-додатків з використанням виключно клієнтських технологій. Результати роботи були апробовані шляхом демонстрації функціональності системи та її тестування в умовах, наближених до реального використання.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

1.1 Поняття та призначення інформаційних систем дистанційного голосування:

Інформаційна система дистанційного голосування (ІСДГ) – це програмно-технічний комплекс, розроблений для організації та проведення опитувань або голосувань через комп'ютерні мережі. Він дозволяє користувачам брати участь без необхідності фізичної присутності на виборчій дільниці. Головна мета ІСДГ – надати прозорий, зручний та швидкий інструмент для збору голосів виборців. Цей механізм активно використовується, наприклад, у студентських опитуваннях, корпоративних голосуваннях або при прийнятті рішень у різних спільнотах. Значною перевагою таких систем є можливість залучення великої кількості учасників, незалежно від їхнього місця розташування, а також автоматизація підрахунку результатів голосування. Окрім зручності та автоматизму, інформаційна система дистанційного голосування (ІСДГ) має ще декілька вагомих переваг.

Передусім, вона суттєво зменшує витрати ресурсів – як грошових, так і людських. Відпадає необхідність у друкуванні паперових бюлетенів, оренді приміщень або залученні великої кількості працівників. По-друге, завдяки використанню цифрових технологій, стає можливим швидке оброблення великого обсягу даних, що робить процес голосування майже миттєвим – від моменту віддання голосу до отримання результату. Сучасні системи електронного голосування можуть бути втілені за допомогою веб-інтерфейсів, мобільних додатків чи навіть шляхом інтеграції з системами електронного документообігу. Це забезпечує легкість їхнього впровадження у широкому спектрі сфер: від муніципалітетів до внутрішніх корпоративних процесів і навчальних закладів.

Під час розробки таких систем ключовим аспектом є безпека та довіра. Це передбачає захист особистої інформації користувачів, забезпечення таємності голосування, уникнення можливості подвійного голосування, а також забезпечення прозорості підрахунку голосів. Для цього можуть застосовуватись передові технології шифрування, автентифікації, імітації та реалізація механізмів хешування та інших методів.

Отже, системи електронного голосування не лише адаптуються до вимог цифрової ери, але й відкривають нові можливості для активної участі громадян у процесах прийняття рішень — зручно, швидко, надійно та результативно.

1.2 Основні вимоги до інформаційних систем голосування (функціональні, нефункціональні, безпекові):

Для ефективного функціонування та довіри з боку користувачів, інформаційні системи дистанційного голосування повинні відповідати ряду специфічних вимог, які можна класифікувати як функціональні, нефункціональні та безпекові.

Функціональні вимоги: Це вимоги, що визначають, що саме система повинна робити. Для розробленої системи "Інформаційна система дистанційного голосування" можна виділити наступні ключові функціональні вимоги:

Управління користувачами:

- Можливість реєстрації нових користувачів із унікальним електронним листом та паролем.
- Функціонал авторизації зареєстрованих користувачів та адміністратора.
- Забезпечення унікальності імені користувача та електронної пошти під час реєстрації.

Управління опитуваннями (для адміністратора):

- Створення нових опитувань з можливістю введення назви, кількох варіантів відповідей (мінімум дві) та встановлення терміну завершення (дедлайну).
- Перегляд списку всіх створених опитувань (активних та завершених).
- Можливість примусового завершення активного опитування.
- Можливість видалення будь-якого опитування.

Процес голосування (для користувача):

- Перегляд списку активних опитувань.
- Можливість вибору одного варіанта відповіді в активному опитуванні.
- Заборона повторного голосування одного користувача в одному опитуванні.
- Відображення статусу опитування (активне/завершене).

Перегляд результатів:

- Відображення узагальнених результатів голосування для кожного опитування у кількісному та відсотковому співвідношенні.
- Візуалізація результатів (наприклад, у вигляді прогрес-барів).
- Можливість повернення до списку опитувань після перегляду результатів.

Нефункціональні вимоги: Це вимоги, що визначають якість системи, її властивості та обмеження.

- Зручність використання (Usability): Інтуїтивно зрозумілий та простий у навігації інтерфейс як для звичайних користувачів, так і для адміністратора.
- Продуктивність (Performance): Швидкий відгук системи на дії користувача, оперативне завантаження даних та відображення результатів.

- Надійність (Reliability): Стабільна робота системи без збоїв, коректне збереження та обробка даних навіть при великій кількості одночасних операцій (у межах клієнтської моделі).
- Адаптивність (Responsiveness): Коректне відображення інтерфейсу на екранах різного розміру (десктоп, планшет, мобільний телефон) завдяки використанню адаптивного дизайну.
- Підтримуваність (Maintainability): Чітка структура коду, що полегшує його модифікацію та подальший розвиток.

Безпекові вимоги: Це вимоги, що стосуються захисту інформації та системи від несанкціонованого доступу, модифікації або знищення.

- Автентифікація: Перевірка ідентичності користувачів (реєстрація/вхід) та адміністратора.
- Авторизація: Надання доступу до певних функцій системи (наприклад, створення/редагування опитувань лише адміністратору) відповідно до ролі користувача.
- Цілісність даних: Забезпечення того, що дані опитувань та результати голосування не будуть випадково чи навмисно змінені. Захист від повторного голосування одного і того ж користувача в одному опитуванні.
- Конфіденційність: Забезпечення захисту персональних даних користувачів. (Тут ВАЖЛИВО) Зазначити, що у демонстраційному проекті з використанням localStorage конфіденційність та захист від несанкціонованого доступу до даних, що зберігаються на клієнтській стороні, є обмеженими. Паролі 'хешуються' простим методом, що є прийнятним для демонстрації концепції, але недостатнім для реального застосування, оскільки localStorage не призначений для зберігання чутливої інформації та легко доступний для перегляду та модифікації з боку користувача."

1.3 Огляд існуючих аналогів та платформ для онлайн-голосування

На сучасному ринку інформаційних технологій представлено велика кількість рішень для проведення онлайн-опитувань та голосувань, які можна розділити на кілька категорій: від простих сервісів для створення анкет до складних систем електронного голосування державного рівня. Для аналізу було обрано декілька типових прикладів, що демонструють різні підходи до реалізації функціоналу.

1. Google Forms:

- Опис: Один з найпопулярніших і найдоступніших інструментів для створення опитувань та анкет. Інтегрований з екосистемою Google.
- Переваги: Простота у використанні, безкоштовність, широкий набір типів питань, автоматичний збір та візуалізація результатів, легка інтеграція з Google Sheets.
- Недоліки: Обмежений функціонал саме для *голосування* (наприклад, складно реалізувати захист від повторного голосування без авторизації, немає поняття 'активних'/'завершених' опитувань як у системі голосування), відсутність ролей адміністратора/користувача в класичному розумінні системи голосування, не призначений для конфіденційного або офіційного голосування.

2. SurveyMonkey:

- Опис: Професійна платформа для проведення опитувань, маркетингових досліджень та збору зворотного зв'язку. Пропонує розширений функціонал для аналізу даних.
- Переваги: Потужні інструменти аналізу даних, різні типи питань, можливість створення складної логіки опитувань, професійні шаблони, масштабованість.

- Недоліки: Переважно платний сервіс (хоча є обмежена безкоштовна версія), орієнтований більше на дослідження, ніж на просте голосування, може бути надмірним для малих та простих завдань голосування.

3. Loomio (або інші платформи для прийняття рішень/співпраці):

- Опис: Платформи, що фокусуються на груповому прийнятті рішень та обговореннях, часто включають функціонал голосування.
- Переваги: Інтеграція з інструментами для обговорення, підтримка складних сценаріїв голосування (наприклад, з кількома варіантами або пріоритезацією), орієнтованість на колегіальні рішення.
- Недоліки: Часто складніші для освоєння, можуть бути орієнтовані на нішеві потреби, можуть вимагати підписки.

1.3.1. Переваги та недоліки існуючих рішень.

На основі аналізу можна узагальнити, що більшість загальнодоступних рішень є або надто простими і не забезпечують достатньої контрольованості та захисту від повторного голосування (як Google Forms), або є надмірно складними та дорогими для потреб простого дистанційного голосування (як SurveyMonkey). Професійні системи голосування зазвичай вимагають значних інвестицій та наявності серверної інфраструктури.

1.3.2. Вибір та обґрунтування підходу до розробки.

Враховуючи поставлену мету – розробка демонстраційної інформаційної системи дистанційного голосування – було обрано підхід створення односторінкового веб-додатку (Single Page Application, SPA), що функціонує виключно на клієнтській стороні. Це дозволило:

- Зосередитися на логіці frontend-розробки: Відпрацювати навички роботи з HTML, CSS та JavaScript для створення динамічного інтерфейсу та взаємодії з користувачем.

- Уникнути складнощів із серверною інфраструктурою: Для цілей демонстрації немає необхідності розгортати та підтримувати сервер, базу даних, що значно спрощує процес розробки та розгортання проекту.
- Швидкість прототипування: Можливість швидко реалізувати ключовий функціонал та перевірити концепцію.

Для збереження даних користувачів та інформації про опитування було обрано механізм localStorage браузера. Цей вибір обґрунтовується простотою реалізації, доступністю даних без серверного запиту та достатністю для демонстрації функціональності системи в умовах, коли не висуваються високі вимоги до безпеки та масштабованості, що є типовим для навчальних проектів.

1.4. Огляд технологій для розробки веб-додатків

Для розробки сучасної клієнтської інформаційної системи дистанційного голосування було використано фундаментальні веб-технології: HTML5, CSS3 та JavaScript.

1.4.1. HTML5, CSS3, JavaScript: огляд можливостей

- **HTML5 (HyperText Markup Language 5):** "Є п'ятою і поточною основною версією мови розмітки HTML, на якій базується більшість веб-сторінок. В контексті даного проекту HTML5 використовується для створення базової структури веб-сторінки (catalog.html), визначення всіх її елементів (форми, кнопки, списки, контейнери для відображення даних). Важливими нововведеннями HTML5, що були застосовані, є семантичні теги (<header>, <nav>, <main>, <footer>), які покращують структуру сторінки та її доступність, а також нові типи елементів форм (<input type="email">, <input type="datetime-local">), які спрощують валідацію вхідних даних."
- **CSS3 (Cascading Style Sheets Level 3):** "Мова стилів, що використовується для візуального оформлення веб-сторінок. У проекті CSS3 застосовується для визначення зовнішнього вигляду всіх елементів інтерфейсу:

кольорової схеми, шрифтів, відступів, позиціонування елементів, а також для створення адаптивного дизайну, що забезпечує коректне відображення системи на пристроях з різним розміром екрана. Використання властивостей Flexbox та Grid (якщо ви їх використовували, інакше просто згадайте про адаптивний дизайн), а також трансформацій та переходів (наприклад, для кнопок, як transition: background-color 0.3s ease;) дозволило створити сучасний та привабливий користувацький інтерфейс."

- **JavaScript (JS):** "Динамічна мова програмування, що виконується на стороні клієнта (в браузері) і є основою інтерактивності веб-додатків. В даному проекті JavaScript виконує всі основні функції:
    - **Маніпуляції з DOM (Document Object Model):** Динамічне створення, зміна та видалення HTML-елементів для відображення списків опитувань, форм, результатів.
    - **Обробка подій:** Реакція на дії користувача (натискання кнопок, введення даних у форми).
    - **Управління станом додатка:** Відстеження поточного користувача, активних опитувань, вибраних варіантів голосування.
    - **Взаємодія з localStorage:** Збереження та зчитування даних користувачів та опитувань.
    - **Реалізація бізнес-логіки:** Валідація форм, обробка реєстрації/авторизації, логіка голосування, підрахунок та візуалізація результатів опитувань."

1.4.2. Механізми клієнтського зберігання даних (localStorage)

- localStorage є одним з механізмів веб-сховища (Web Storage API), який дозволяє веб-додаткам зберігати дані безпосередньо в браузері користувача. Ці дані зберігаються у вигляді пар ключ/значення і залишаються доступними навіть після закриття браузера або

перезавантаження сторінки, на відміну від sessionStorage, дані якого видаляються при закритті вкладки.

Принцип роботи: localStorage оперує лише рядками, тому для збереження складних об'єктів (таких як об'єкти користувачів або опитувань у даному проекті) необхідно використовувати JSON.stringify() для перетворення об'єкта в рядок перед збереженням та JSON.parse() для зворотного перетворення при отриманні даних.

Переваги localStorage для даного проекту:

- **Простота використання:** Легкий API для збереження та отримання даних.
- **Стійкість даних:** Зберігаються між сесіями браузера, що дозволяє зберегти стан системи.
- **Відсутність серверної залежності:** Дозволяє реалізувати функціонал збереження даних без необхідності розгортати сервер та базу даних, що ідеально підходить для демонстраційного та навчального проекту.
- **Значний обсяг зберігання:** До 5-10 МБ на домен, що є більш ніж достатнім для даних даної системи.

Недоліки та обмеження localStorage (особливо важливі для дипломної роботи):

- **Безпека:** localStorage є небезпечним для зберігання конфіденційної інформації (паролі, особисті дані), оскільки дані зберігаються у відкритому вигляді (хоч і "хешуються" у цьому проекті, але метод хешування простий) і легко можуть бути переглянуті або змінені користувачем через інструменти розробника браузера. Це робить його непридатним для систем, що потребують високого рівня безпеки даних.

- **Відсутність синхронізації:** Дані зберігаються лише на одному пристрої та в одному браузері. Зміни, внесені на одному пристрої, не будуть відображені на іншому, що робить його непридатним для багатокористувацьких систем з централізованим зберіганням даних.
- **Неблокуючий доступ:** Операції з localStorage є синхронними, що може призвести до "зависання" інтерфейсу при роботі з дуже великими обсягами даних (хоча для даного проекту це не критично).
- **Відсутність підтримки складних запитів:** Немає можливості виконувати складні SQL-подібні запити чи індексування даних. Усі операції з пошуку та фільтрації потрібно реалізовувати вручну за допомогою JavaScript.

Саме через ці обмеження localStorage було обрано як інструмент для демонстрації концепції роботи системи, а не як рішення для промислової експлуатації.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДИСТАНЦІЙНОГО ГОЛОСУВАННЯ

2.1. Постановка задачі та визначення функціональних вимог

Проектування будь-якої інформаційної системи розпочинається з чіткої постановки задачі та детального визначення функціональних вимог. Цей етап є критично важливим, оскільки він закладає основу для подальшої розробки, забезпечуючи відповідність кінцевого продукту потребам користувачів та поставленим цілям.

2.1.1. Постановка задачі

Головною задачею, що стоїть перед розроблюваною системою, є створення зручного та простого у використанні інструменту для проведення дистанційних голосувань та опитувань. Система має забезпечувати розмежування ролей користувачів: звичайний користувач (виборець) може переглядати активні опитування та голосувати, а адміністратор має розширені можливості для створення, управління та моніторингу опитувань. Ключовою особливістю даного проекту є його реалізація як повністю клієнтського веб-додатку з локальним збереженням даних у браузері користувача. Це спрощує розгортання та демонстрацію системи, роблячи її доступною для використання у навчальних закладах, невеликих громадських ініціативах або для швидких внутрішніх опитувань, де не потрібна складна серверна інфраструктура чи високий рівень захисту конфіденційних даних.

2.1.2. Визначення функціональних вимог

На основі поставленої задачі та попереднього аналізу існуючих рішень (див. підрозділ 1.3), були сформовані детальні функціональні вимоги до системи. Ці вимоги стали основою для подальшого проектування логіки роботи системи, структури даних та користувацького інтерфейсу:

Для всіх користувачів (незалежно від ролі):

- Можливість реєстрації нового облікового запису, що включає унікальний email, ім'я користувача та пароль.

- Можливість авторизації в системі за допомогою зареєстрованих облікових даних (email та пароль).
- Можливість коректного виходу із системи, що призводить до очищення сесії користувача.

Для звичайного користувача (виборця):

- Перегляд списку доступних для голосування активних опитувань із зазначенням їхньої назви та терміну завершення.
- Вибір одного варіанту відповіді у вибраному активному опитуванні.
- Система повинна автоматично запобігати повторному голосуванню одного користувача в межах одного й того ж опитування.
- Перегляд узагальнених результатів голосування для завершених опитувань або для опитувань, в яких користувач вже проголосував.

Для адміністратора:

- Можливість авторизації з адміністративними правами за допомогою спеціальних облікових даних.
- Функціонал створення нових опитувань, що передбачає введення назви, додавання від двох до десяти варіантів відповідей, а також встановлення точної дати та часу завершення (дедлайну).
- Перегляд повного списку всіх створених опитувань, з відображенням їхнього поточного статусу (активне/завершене).
- Можливість примусового завершення будь-якого активного опитування адміністратором.
- Можливість повного видалення будь-якого опитування із системи, включаючи всі пов'язані з ним голоси.

- Перегляд детальних узагальнених результатів будь-якого опитування, включаючи кількісні та відсоткові дані для кожного варіанта відповіді, з візуалізацією.

2.2. Моделювання функціональності системи

Моделювання функціональності є ключовим етапом проектування, що дозволяє візуалізувати взаємодію користувачів з системою та послідовність виконання ключових процесів. Для цього використовуються діаграми мови UML (Unified Modeling Language), які надають стандартизований спосіб графічного представлення архітектури та поведінки програмного забезпечення.

2.2.1. Діаграма варіантів використання

Діаграма варіантів використання є ефективним інструментом для візуалізації взаємодії акторів із системою та основними функціями, які вона надає. На рис. 2.1 представлено діаграму варіантів використання розробленої Інформаційної системи дистанційного голосування.

Акторами даної системи є:

- **Користувач (Виборець):** Звичайна особа, яка взаємодіє з системою для реєстрації, авторизації, перегляду опитувань та участі в голосуванні.
- **Адміністратор:** Користувач з розширеними правами, який відповідає за створення, управління та моніторинг опитувань.

Обидва актори можуть виконувати базові функції, такі як реєстрація, авторизація та вихід із системи. 'Користувач (Виборець)' має можливість переглядати список активних опитувань, брати участь у голосуванні та ознайомлюватися з результатами. 'Адміністратор', окрім загальних функцій, володіє повним контролем над опитуваннями: він може створювати нові опитування, переглядати їхній повний список, примусово завершувати або видаляти, а також детально аналізувати результати голосувань.

Дивіться Рис. Г.1.

2.2.2. Опис процесів діяльності системи

Для розуміння послідовності кроків та логіки виконання ключових операцій у системі, були детально описані основні процеси діяльності.

2.2.2.1. Опис процесу реєстрації користувача

Процес реєстрації нового користувача включає наступні послідовні кроки:

1. **Введення облікових даних:** Користувач вводить у відповідні поля форми електронну пошту, ім'я користувача, пароль та підтвердження пароля.
2. **Валідація даних:** Система перевіряє коректність введених даних (наприклад, відповідність формату email, мінімальну довжину пароля, співпадіння пароля та його підтвердження). У разі виявлення помилок, користувачеві виводиться відповідне повідомлення, і процес переривається до виправлення помилок.
3. **Перевірка унікальності email:** Система перевіряє, чи не існує вже облікового запису з введеною електронною поштою у локальному сховищі даних (localStorage). Якщо email вже зареєстровано, користувач отримує повідомлення, і процес реєстрації припиняється.
4. **Імітація хешування пароля:** Для забезпечення базової безпеки, введений пароль перетворюється на простий хеш-рядок перед збереженням.
5. **Збереження користувача:** Дані нового користувача (email, ім'я користувача, хешований пароль) зберігаються в localStorage як новий об'єкт у масиві користувачів.
6. **Підтвердження реєстрації:** Користувачеві виводиться повідомлення про успішну реєстрацію, і він може перейти до авторизації.

2.2.2.2. Опис процесу створення опитування адміністратором

Процес створення нового опитування адміністратором виконується за такою логікою:

1. **Авторизація адміністратора:** Адміністратор входить у систему під своїми обліковими даними, після чого отримує доступ до адміністративної панелі.
2. **Вибір функції "Створити опитування":** Адміністратор активує форму для створення нового опитування.
3. **Введення даних опитування:** Адміністратор вводить назву опитування, а також додає щонайменше два варіанти відповідей. Можливість додавання до десяти варіантів реалізована динамічно. Також встановлюється дата та час завершення опитування (дедлайн).
4. **Валідація даних опитування:** Система перевіряє, чи всі обов'язкові поля заповнені, чи коректно введена дата дедлайну, і чи є мінімальна кількість варіантів відповідей. При виявленні помилок виводиться повідомлення.
5. **Генерація ID:** Для нового опитування генерується унікальний ідентифікатор, зазвичай на основі поточної позначки часу.
6. **Збереження опитування:** Сформований об'єкт опитування (з назвою, варіантами, дедлайном, порожнім списком голосів та статусом "активне") зберігається в localStorage.
7. **Підтвердження:** Адміністратор отримує повідомлення про успішне створення нового опитування, і воно з'являється у загальному списку.

2.2.2.3. Опис процесу голосування користувачем

Процес участі користувача в голосуванні виглядає наступним чином:

1. **Авторизація користувача:** Користувач входить у систему під своїм обліковим записом.
2. **Перегляд активних опитувань:** Система відображає користувачеві список усіх опитувань, які наразі є активними та доступними для голосування.

3. **Вибір опитування:** Користувач обирає конкретне опитування, в якому бажає проголосувати.
4. **Перевірка на повторне голосування:** Система перевіряє, чи не був голос цього користувача вже зафіксований у вибраному опитуванні. Якщо так, користувачеві виводиться повідомлення про заборону повторного голосування, і можливість проголосувати відсутня.
5. **Вибір варіанта:** Якщо користувач ще не голосував, йому надається можливість обрати один з варіантів відповіді.
6. **Фіксація голосу:** Вибір користувача фіксується у структурі даних опитування в localStorage (додається запис, що цей користувач проголосував за певний варіант).
7. **Оновлення стану:** Система оновлює відображення опитування, щоб показати, що користувач вже проголосував, або переходить на сторінку результатів, якщо це передбачено логікою.
8. **Підтвердження:** Користувач отримує повідомлення про успішне голосування.

2.3. Проектування архітектури системи

Проектування архітектури системи визначає її загальну структуру, ключові компоненти та взаємозв'язки між ними. Враховуючи обраний підхід створення клієнтського веб-додатку, архітектура системи є відносно простою, зосередженою на взаємодії браузера з локальним сховищем даних.

2.3.1. Загальна архітектура системи

Інформаційна система дистанційного голосування побудована за **клієнтською архітектурою** без використання традиційної серверної частини для обробки та зберігання даних. Це означає, що весь програмний код та логіка системи (HTML, CSS, JavaScript) виконуються безпосередньо у веб-браузері користувача. Взаємодія з даними відбувається за допомогою вбудованого механізму localStorage браузера.

Схема взаємодії компонентів:

1. **Користувач/Адміністратор:** Здійснює взаємодію із системою через користувацький інтерфейс, що відображається у веб-браузері.
2. **Веб-браузер:** Є середовищем виконання для всього додатка. Він завантажує catalog.html, style.css та script.js (або весь код знаходиться в одному HTML-файлі).
3. **Клієнтський веб-додаток (HTML, CSS, JavaScript):**
    - **HTML:** Відповідає за структурну розмітку сторінки, розміщення всіх елементів інтерфейсу (форм, кнопок, списків тощо).
    - **CSS:** Визначає візуальне оформлення та стиль елементів інтерфейсу, забезпечуючи привабливий та адаптивний дизайн.
    - **JavaScript:** Є ядром логіки системи. Він обробляє взаємодію користувача (натискання кнопок, введення даних), динамічно змінює вміст сторінки (додає/видаляє опитування, оновлює результати), керує автентифікацією/авторизацією та взаємодіє з localStorage для збереження та отримання даних.
4. **localStorage браузера:** Виступає у ролі сховища даних. Всі відомості про користувачів, опитування та голоси зберігаються тут у вигляді JSON-об'єктів. localStorage забезпечує стійке зберігання даних, які залишаються доступними навіть після закриття браузера.

Ця архітектура забезпечує простоту розгортання та незалежність від серверної інфраструктури, що є оптимальним для демонстраційного та навчального проекту. Однак, вона накладає обмеження щодо масштабованості та безпеки, оскільки дані зберігаються на клієнтській стороні та не є захищеними від прямого доступу чи модифікації з боку досвідченого користувача.

2.3.2. Структура даних

Для ефективного зберігання та управління інформацією про користувачів та опитування в localStorage застосовується структурований підхід. Дані зберігаються у вигляді JSON (JavaScript Object Notation) об'єктів, що дозволяє легко серіалізувати їх у рядки для збереження за допомогою JSON.stringify() та десеріалізувати назад в об'єкти за допомогою JSON.parse() при отриманні даних.

2.3.2.1. Структура даних користувача

Дані про кожного зареєстрованого користувача зберігаються у масиві об'єктів під ключем 'users' у localStorage. Кожен об'єкт користувача містить наступні поля:

- email (String): Унікальний ідентифікатор користувача, що використовується для входу в систему.
- username (String): Ім'я користувача, що відображається в інтерфейсі.
- passwordHash (String): Рядок, що імітує хешований пароль. **Важливо зазначити, що це є спрощеною реалізацією для демонстраційних цілей і не забезпечує високого рівня криптографічної безпеки для реального застосування.**
- role (String, опціонально): Для звичайних користувачів це значення може бути відсутнім або мати значення 'user', для адміністратора – 'admin'.

JSON-структура для даних користувачів: дивіться додаток В.1.

2.3.2.2. Структура даних опитування

Дані про всі опитування (голосування) зберігаються у масиві об'єктів під ключем 'polls' у localStorage. Кожен об'єкт опитування містить наступні поля:

- id (String): Унікальний ідентифікатор опитування, що генерується, наприклад, на основі позначки часу (Unix timestamp).
- title (String): Назва або тема опитування.

- options (Array of Strings): Масив, що містить текстові варіанти відповідей для даного опитування.
- deadline (String): Дата та час завершення опитування у форматі ISO 8601 (наприклад, 'YYYY-MM-DDTHH:mm').
- votes (Object): Об'єкт, що зберігає інформацію про голоси. Ключами цього об'єкта є email користувача, який проголосував, а значеннями – індекс (числовий) обраного варіанту відповіді з масиву options. Це забезпечує унікальність голосу від кожного користувача в рамках одного опитування.
- active (Boolean): Логічне значення, що вказує поточний статус опитування (true, якщо опитування активне і доступне для голосування; false, якщо воно завершене).

JSON-структура для даних опитувань: дивіться додаток В.2.

Ці структури даних забезпечують логічне та послідовне зберігання інформації, що дозволяє ефективно керувати функціоналом системи та відображати актуальні дані користувачам та адміністратору

2.4 Проектування інтерфейсу користувача (UI/UX)

Проектування користувацького інтерфейсу (UI) та досвіду взаємодії (UX) є ключовим етапом розробки, оскільки від нього безпосередньо залежить зручність, ефективність та задоволення користувача від роботи із системою. Метою проектування інтерфейсу Інформаційної системи дистанційного голосування було створення інтуїтивно зрозумілого, візуально привабливого та функціонального середовища для обох категорій користувачів – виборців та адміністраторів.

2.4.1. Основні принципи дизайну інтерфейсу

При розробці користувацького інтерфейсу системи “E-Vote” були застосовані наступні принципи, що забезпечують її зручність та ефективність:

- **Простота та мінімалізм:** Інтерфейс максимально спрощено, щоб уникнути перевантаження інформацією та зайвими елементами. Фокус зроблено на основному функціоналі, що дозволяє користувачам швидко орієнтуватися та виконувати необхідні дії без відволікань.

- **Інтуїтивність:** Всі елементи управління (кнопки, поля вводу, списки) є зрозумілими за своїм призначенням, а навігація між екранами – логічною та передбачуваною, не вимагаючи додаткових інструкцій.
- **Узгодженість:** Застосовано єдиний візуальний стиль (кольорова палітра, шрифти, розміри елементів) та послідовне розташування ключових елементів на всіх екранах системи. Це забезпечує цілісність сприйняття та зменшує когнітивне навантаження на користувача.
- **Зворотний зв'язок:** Система надає чіткий та своєчасний зворотний зв'язок на дії користувача (наприклад, повідомлення про успішну реєстрацію, помилки валідації, підтвердження голосування).
- **Адаптивність (Responsive Design):** Завдяки використанню сучасних CSS-технологій, інтерфейс системи коректно відображається та функціонує на пристроях з різним розміром екрана – від великих моніторів до мобільних телефонів. Це досягається за рахунок гнучких макетів (Flexbox, Grid) та медіа-запитів.
- **Доступність:** Були враховані базові аспекти доступності, такі як достатній контраст кольорів тексту та фону, використання семантичних HTML-елементів для покращення навігації для користувачів допоміжних технологій.

2.4.2. Опис основних екранів системи

Для демонстрації ключових аспектів користувацького досвіду та візуальної структури системи, нижче наведено описи основних екранів. Детальні скріншоти реалізованого інтерфейсу, що повністю відображають фінальний дизайн, представлені у Додатку Б.

- Екран "Авторизація / Реєстрація":
    - Цей екран є початковою точкою взаємодії з системою. Він містить дві основні форми: для входу існуючих користувачів та для реєстрації нових.

- **Форма авторизації** включає поля для введення електронної пошти та пароля, а також кнопку "Увійти".
- **Форма реєстрації** передбачає поля для введення електронної пошти, бажаного імені користувача, пароля та його підтвердження, а також кнопку "Зареєструватися". Присутня навігація між цими двома формами.
- На цьому екрані також можуть відображатися повідомлення про помилки (наприклад, "Невірний логін або пароль", "Користувач з таким email вже існує").

- **Екран "Панель користувача (Список активних опитувань)":**
  - Цей екран є основним для зареєстрованого користувача.
  - Він відображає перелік всіх активних опитувань, доступних для голосування. Кожен елемент списку чітко показує назву опитування, його термін завершення.
  - Для кожного опитування є кнопка "Проголосувати", яка веде до детальної сторінки голосування.
  - Опитування, в яких користувач вже проголосував, можуть бути візуально позначені або мати деактивовану кнопку "Проголосувати".
- **Екран "Сторінка голосування":**
  - Цей екран відображається після вибору конкретного опитування зі списку.
  - Він містить назву опитування та список його варіантів відповідей, представлених у вигляді радіокнопок, що дозволяють вибрати лише один варіант.
  - Присутня кнопка "Проголосувати" для фіксації вибору.

- Якщо користувач вже голосував у цьому опитуванні, відображається відповідне повідомлення, а можливість повторного голосування блокується.
- Є кнопка "Назад до списку", яка дозволяє повернутися до переліку опитувань.

- **Екран "Адміністративна панель (Список всіх опитувань)":**
  - Цей екран доступний лише для адміністратора системи.
  - Він відображає повний список усіх опитувань, створених у системі, включаючи як активні, так і вже завершені.
  - Для кожного опитування відображається його назва, дедлайн, поточний статус ("Активне", "Завершене").
  - Доступні кнопки для управління кожним опитуванням: "Завершити" (для активних), "Видалити" та "Результати" (для перегляду детальної статистики).
  - Присутня кнопка "Створити нове опитування", яка веде до відповідної форми.
- **Екран "Створення нового опитування (для адміністратора)":**
  - Це форма, призначена для адміністратора.
  - Містить поля для введення назви опитування, динамічно додаються поля для введення варіантів відповідей (з кнопками "Додати варіант" та "Видалити варіант").
  - Присутнє поле для вибору дати та часу завершення опитування (дедлайну) за допомогою спеціального елемента керування датою та часом (datetime-local).
  - Форма завершується кнопками "Створити опитування" та "Скасувати".

- **Екран "Результати голосування":**
    - Цей екран відображає підсумки голосування для конкретного опитування.
    - На ньому показується назва опитування, а для кожного варіанта відповіді – кількість голосів та їх відсоткове співвідношення до загальної кількості голосів.
    - Результати візуалізовані у вигляді горизонтальних прогрес-барів, що наочно демонструють розподіл голосів.
    - Є кнопка "Закрити результати" для повернення до попереднього екрану

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДИСТАНЦІЙНОГО ГОЛОСУВАННЯ

3.1. Загальна характеристика програмної реалізації

Програмна реалізація Інформаційної системи дистанційного голосування здійснювалася як односторінковий веб-додаток (Single Page Application, SPA), що повністю функціонує на клієнтській стороні. Цей підхід дозволив зосередитися на розробці інтерактивного інтерфейсу та логіки взаємодії без необхідності розгортання та підтримки серверної інфраструктури, що є оптимальним для демонстраційного та навчального проекту.

Основними технологіями, використаними для реалізації, є:

- **HTML5:** Використаний для створення базової структури веб-сторінки (catalog.html) та організації всіх статичних елементів користувацького інтерфейсу, таких як форми, кнопки, та контейнери для динамічного контенту.
- **CSS3:** Застосований для візуального оформлення елементів сторінки, забезпечення адаптивного дизайну, що коректно відображається на різних пристроях, та створення привабливого зовнішнього вигляду. Зокрема, використані сучасні техніки Flexbox та Grid для гнучкого компонування елементів, а також медіа-запити для адаптації під різні розміри екранів.
- **JavaScript (ES6+):** Є основною мовою для реалізації всієї динамічної логіки системи. Він відповідає за обробку подій користувача (натискання кнопок, введення даних), динамічну маніпуляцію з DOM (Document Object Model) для оновлення вмісту сторінки без перезавантаження, управління станом користувача та системи, валідацію даних, а також збереження та отримання інформації з localStorage.

Весь програмний код системи, включаючи HTML-розмітку, CSS-стилі та JavaScript-логіку, інтегрований в єдиний файл catalog.html. Це рішення було прийняте для максимального спрощення розгортання та запуску демонстраційного проекту. Зберігання даних реалізовано за допомогою

localStorage браузера, що дозволяє зберігати інформацію про користувачів, опитування та голоси без постійного з'єднання з базою даних. Це ідеально підходить для автономної роботи та швидкої демонстрації функціоналу, але накладає обмеження щодо масштабованості та захисту даних у реальних умовах.

3.2. Опис основних програмних модулів та їх функціоналу

Для забезпечення модульності та зручності розробки, логіка системи була розділена на кілька функціональних блоків (модулів). Хоча фізично весь JavaScript-код розташований в одному файлі, логічно він розділений на секції, які керують відповідними частинами системи, що покращує читабельність та підтримуваність коду.

3.2.1. Модуль керування даними (localStorage API)

Цей модуль відповідає за взаємодію з localStorage для збереження та отримання всіх даних системи. Він забезпечує абстракцію від прямої роботи з localStorage.setItem() та localStorage.getItem(), інкапсулюючи логіку серіалізації/десеріалізації JSON-даних.

Основні функції:

- loadData(key): Завантажує дані за вказаним ключем з localStorage та парсить їх з JSON-рядка в JavaScript-об'єкт (масив). Якщо даних за ключем немає, повертає порожній масив.
- saveData(key, data): Зберігає JavaScript-об'єкт (масив) у localStorage під вказаним ключем, попередньо серіалізуючи його в JSON-рядок.
- Ініціалізація початкових даних: При першому запуску системи (або якщо дані відсутні) створюються порожні масиви для користувачів (users) та опитувань (polls). Також реалізована логіка створення облікового запису адміністратора за замовчуванням, якщо він ще не існує.

Фрагмента коду: дивіться додаток В.3.

Цей фрагмент коду демонструє реалізацію основних функцій для взаємодії з localStorage. Функції loadData та saveData є універсальними для читання та

запису будь-яких JSON-сутностей. Показано ініціалізацію глобальних змінних users та polls, які зберігають усі дані системи. Особлива увага приділена ініціалізації облікового запису адміністратора: система перевіряє його наявність і автоматично створює, якщо він відсутній, забезпечуючи початковий доступ до адміністративних функцій.

3.2.2. Модуль автентифікації та авторизації

Цей модуль відповідає за процеси реєстрації нових користувачів, входу в систему (автентифікація) та визначення їхніх прав доступу (авторизація).

Основні функції та обробники подій:

- **Реєстрація користувача:** Обробляє дані форми реєстрації (registrationForm), включаючи валідацію полів (перевірка на порожність, співпадіння паролів) та унікальність email. Перед збереженням у localStorage пароль імітується хешуванням за допомогою простої функції simpleHash().
- **Вхід користувача:** Обробляє дані форми авторизації (loginForm). Перевіряє надані email та хешований пароль на відповідність існуючим записам у users. У разі успіху, об'єкт поточного користувача (currentUser) зберігається в localStorage для підтримки сесії, і користувач перенаправляється на відповідну сторінку (панель користувача або адміністратора).
- **Вихід із системи:** Обробник кнопки logoutBtn очищує інформацію про поточного користувача з localStorage та глобальної змінної currentUser, повертаючи інтерфейс до екрану авторизації.

Фрагмент коду: дивіться додаток В.4.

Даний фрагмент коду демонструє ключові механізми автентифікації та авторизації. При реєстрації (registrationForm.addEventListener) виконуються перевірки на заповненість полів, співпадіння паролів та унікальність email. Пароль трансформується за допомогою simpleHash перед збереженням.

Обробник входу (loginForm.addEventListener) здійснює пошук користувача за email та хешованим паролем. У разі успішної автентифікації, об'єкт currentUser зберігається в localStorage, і система перенаправляє користувача до відповідної панелі залежно від його ролі (user або admin). Функція виходу (logoutBtn.addEventListener) обнуляє стан сесії та повертає інтерфейс до початкового екрану входу/реєстрації.

3.2.3. Модуль керування опитуваннями (для адміністратора)

Цей модуль реалізує повний життєвий цикл управління опитуваннями, доступний виключно для користувачів з роллю адміністратора.

Основні функції та обробники подій:

- **Створення опитування:** Обробник форми createPollForm збирає назву, варіанти відповідей та дедлайн. Виконується валідація даних (мінімум 2 варіанти, наявність назви та дедлайну). Для нового опитування генерується унікальний id на основі Date.now(), і воно додається до масиву polls, після чого зберігається.
- **Відображення списку опитувань:** Функція renderAdminPollList() динамічно генерує HTML-розмітку для всіх опитувань. Вона відображає назву, статус (активне/завершене, з урахуванням дедлайну), дедлайн та надає кнопки для керування: "Завершити", "Видалити", "Результати".
- **Завершення опитування:** Обробник кнопки "Завершити" змінює статус active відповідного опитування на false та оновлює дані в localStorage.
- **Видалення опитування:** Обробник кнопки "Видалити" видаляє об'єкт опитування з масиву polls після підтвердження користувача.
- **Перегляд результатів:** Обробник кнопки "Результати" викликає функцію renderPollResults для відображення детальної статистики голосування.

Фрагмент коду: дивіться додаток В.5.

Тут відображено основні аспекти управління опитуваннями адміністратором. Обробник форми createPollForm.addEventListener відповідає за збір даних нового опитування, їх валідацію та збереження унікального об'єкта опитування до масиву polls. Функція renderAdminPollList динамічно формує HTML-структуру для відображення всіх опитувань у зручному форматі, включаючи їхній поточний статус (який автоматично оновлюється на "Завершене" за дедлайном). Вона також прикріплює обробники подій до кнопок "Завершити", "Видалити" та "Результати", забезпечуючи можливість виконання відповідних адміністративних дій над опитуваннями.

3.2.4. Модуль голосування (для користувача) та відображення результатів

Цей модуль реалізує основний функціонал системи для звичайних користувачів, дозволяючи їм переглядати активні опитування, голосувати та переглядати узагальнені результати.

Основні функції та обробники подій:

- **Відображення активних опитувань:** Функція renderPollsForUser() фільтрує опитування, залишаючи лише ті, що активні та не завершилися за дедлайном. Для кожного такого опитування динамічно генерується елемент списку з назвою, дедлайном та кнопкою "Проголосувати". Кнопка деактивується, якщо поточний користувач вже проголосував у цьому опитуванні.
- **Сторінка голосування:** Функція showPollDetails(pollId) відображає деталі конкретного опитування, включаючи його назву та варіанти відповідей у вигляді радіокнопок.
- **Фіксація голосу:** Обробник кнопки "Проголосувати" на сторінці деталей опитування перевіряє, чи вибрав користувач варіант, та чи не голосував він уже в цьому опитуванні. Якщо всі умови виконані, голос фіксується у відповідному об'єкті опитування в votes та зберігається.

- **Відображення результатів:** Функція renderPollResults(poll) обчислює загальну кількість голосів та кількість голосів за кожен варіант. Вона динамічно генерує візуалізацію результатів, використовуючи прогрес-бари, що відображають відсоткове співвідношення голосів для кожного варіанта.

Фрагмент коду: дивіться додаток В.6.

Цей фрагмент коду реалізує основну функціональність для користувачів та відображення результатів. Функція renderPollsForUser відповідає за динамічне створення списку активних опитувань, надаючи користувачеві можливість проголосувати або переглянути результати, якщо він вже голосував. showPollDetails відображає обране опитування, дозволяючи користувачеві вибрати варіант відповіді; вона також містить логіку запобігання повторному голосуванню та обробку самого акту голосування.

Функція renderPollResults відповідає за збір та візуалізацію статистики голосування для будь-якого опитування, розраховуючи відсотки та генеруючи прогрес-бари для наочності. Обробник closeResultsBtn дозволяє повернутися до попереднього інтерфейсу (список опитувань для користувача або адмін-панель).

3.3. Тестування та налагодження роботи платформи

Після завершення етапу реалізації, критично важливим є проведення тестування та налагодження розробленої системи. Цей процес дозволяє виявити помилки, переконатися у відповідності системи визначеним вимогам та забезпечити її стабільну та ефективну роботу.

3.3.1. Методика тестування

Тестування проводилося за методикою чорної скриньки (Black-box testing), що означає перевірку функціональності системи з точки зору кінцевого користувача, без доступу до внутрішньої структури коду. Основна увага приділялася функціональному тестуванню.

Етапи тестування включали:

1. **Тестування реєстрації та авторизації:**

- Перевірка коректної реєстрації нових користувачів та адміністратора за замовчуванням.
- Тестування входу в систему з правильними та невірними обліковими даними (неіснуючий email, неправильний пароль).
- Перевірка коректного розмежування доступу за ролями (звичайний користувач не може бачити адмін-функціонал, і навпаки).
- Тестування коректного виходу із системи та очищення даних сесії.

2. **Тестування функціоналу адміністратора:**
   - Створення нових опитувань з різною кількістю варіантів (від 2 до 10) та встановленням дедлайнів.
   - Перевірка валідації полів при створенні опитування (наприклад, наявність назви, мінімум 2 варіанти, коректна дата дедлайну).
   - Тестування можливості примусового завершення активного опитування та перевірка зміни його статусу.
   - Тестування можливості повного видалення опитування.
   - Перевірка коректного відображення повного списку опитувань в адмін-панелі з актуальним статусом.
3. **Тестування функціоналу користувача:**
   - Перегляд списку активних опитувань та їх коректне відображення.
   - Голосування в різних активних опитуваннях.
   - Тестування механізму запобігання повторному голосуванню одного користувача в одному опитуванні – спроби проголосувати двічі мають блокуватися.
   - Перевірка коректного відображення узагальнених результатів голосування, включаючи кількісні та відсоткові показники.

4. **Тестування збереження даних:**
    - Перевірка збереження всіх даних (users, polls) у localStorage після виконання операцій (реєстрація, голосування, створення/видалення опитувань) та після перезавантаження сторінки або закриття/відкриття браузера.
    - Перевірка цілісності даних та їх відповідності очікуванням після багаторазових змін.
5. **Тестування інтерфейсу та адаптивності:**
    - Візуальна перевірка коректності відображення всіх елементів інтерфейсу на різних екранах (наприклад, реєстрація, авторизація, список опитувань, форма створення, результати).
    - Тестування адаптивності на пристроях з різним розміром екрана (використання інструментів розробника браузера для емуляції мобільних пристроїв або на реальних мобільних пристроях).

3.3.2. Виявлені недоліки та шляхи їх усунення (Налагодження)

Під час тестування були виявлені незначні недоліки та баги, які були оперативно усунені. Приклади таких недоліків та способів їх усунення включають:

- **Недолік 1:** Після створення нового опитування адміністратором, список опитувань в адмін-панелі не оновлювався автоматично, вимагаючи ручного перезавантаження сторінки для відображення змін.
    - **Шлях усунення:** Впровадження повторного виклику функції renderAdminPollList() одразу після успішного збереження нового опитування в localStorage. Це забезпечило негайне оновлення інтерфейсу та відображення новоствореного опитування.
- **Недолік 2:** Існувала можливість голосування користувачем в опитуванні, яке вже завершилося за встановленим дедлайном, або в якому він уже проголосував, що призводило до некоректних даних.

- **Шлях усунення:** Додано додаткові перевірки статусу active опитування та порівняння поточної дати з deadline при рендерингу списку опитувань для користувача (renderPollsForUser). Кнопки "Проголосувати" деактивуються (disabled атрибут) та змінюється їхній текст, якщо опитування неактивне, завершилося, або користувач вже проголосував. Аналогічні перевірки додані і в логіку функції showPollDetails та submitVoteBtn для остаточного запобігання невірним діям.

- **Недолік 3:** При виході з системи або переході між режимами користувача/адміністратора, вміст деяких контейнерів (наприклад, форма створення опитування) залишався видимим, створюючи візуальний безлад.
    - **Шлях усунення:** Вдосконалено функції управління відображенням секцій інтерфейсу (showAuthForm, showAdminPanel, showUserPanel). Додано явне приховування всіх неактивних контейнерів за допомогою classList.add('hidden') перед відображенням потрібної секції, що забезпечило чистий перехід між станами інтерфейсу.

3.3.3. Результати тестування

За результатами проведеного всебічного тестування та налагодження, Інформаційна система дистанційного голосування демонструє стабільну роботу та повністю відповідає заявленим функціональним вимогам, що були визначені на етапі проектування.

Всі ключові сценарії використання системи, включаючи реєстрацію нових користувачів, авторизацію під різними ролями, створення та управління опитуваннями адміністратором, а також участь у голосуванні та перегляд результатів звичайним користувачем, працюють коректно.

Забезпечено базову стійкість до помилок введення даних та логіки взаємодії. Система готова до демонстрації та подальшого потенційного розвитку.

ВИСНОВКИ

У кваліфікаційній роботі було виконано комплексне дослідження та розробку інформаційної системи дистанційного голосування, яка забезпечує можливість проведення опитувань та фіксації голосів користувачів.

В ході виконання роботи було **досягнуто поставленої мети** – розроблено функціональну веб-систему для проведення голосувань з використанням клієнтських веб-технологій та локального сховища даних.

Для досягнення цієї мети було **виконано наступні завдання**:

1. **Проаналізовано існуючі системи дистанційного голосування** та визначено їхні ключові особливості, переваги та недоліки. Це дозволило сформувати розуміння основних функціональних вимог та архітектурних підходів до подібних систем.
2. **Визначено функціональні та нефункціональні вимоги** до розроблюваної системи. Були деталізовані вимоги до ролей користувачів (Користувач/Виборець та Адміністратор), до механізмів реєстрації, авторизації, створення та управління опитуваннями, а також до процесу голосування та відображення результатів.
3. **Розроблено проект системи**, включаючи моделювання функціональності та проектування архітектури. Були визначені основні варіанти використання системи, описані ключові процеси її діяльності (реєстрація, створення опитування, голосування), а також розроблено структуру зберігання даних для localStorage (опис об'єктів користувачів та опитувань).

4. **Реалізовано програмне забезпечення** інформаційної системи дистанційного голосування з використанням HTML5, CSS3 та JavaScript. Вся логіка системи була втілена на клієнтській стороні, а для зберігання даних використано localStorage браузера. Реалізовано функціонал реєстрації/авторизації, панелі адміністратора для створення та управління опитуваннями, та панелі користувача для участі в голосуваннях та перегляду результатів.
5. **Проведено тестування та налагодження** розробленої платформи. Була застосована методика чорної скриньки для функціонального тестування, виявлені та усунені недоліки, що забезпечило стабільну та коректну роботу системи.

Практичне значення отриманих результатів полягає в створенні готового до використання програмного продукту, який може бути застосований для:

- Проведення неформальних опитувань та зборів думок у невеликих групах або спільнотах.
- Використання як навчального інструменту для демонстрації принципів клієнтської веб-розробки та взаємодії з localStorage.
- Старту для подальшого розвитку у повноцінну систему з розширеним функціоналом та серверною частиною.

Перспективи подальшого розвитку системи включають:

- **Перехід на повноцінну сервер-клієнт архітектуру** з використанням бази даних (наприклад, SQL або NoSQL) для забезпечення масштабованості, централізованого зберігання даних та підвищення безпеки.
- **Впровадження більш надійних механізмів автентифікації та хешування паролів**, таких як bcrypt або PBKDF2.

- **Розширення функціоналу опитувань**, додавання можливості багатоваріантного вибору, анонімного голосування, різних типів питань (текстові, з оцінкою).
- **Удосконалення інтерфейсу користувача**, включення анімацій, покращення візуалізації результатів (діаграми, графіки).
- **Реалізація можливості управління користувачами** з боку адміністратора (блокування, редагування ролей).
- **Впровадження системи сповіщень** про нові опитування або завершення поточних.

Таким чином, розроблена інформаційна система дистанційного голосування є успішним прикладом застосування сучасних веб-технологій для вирішення практичної задачі, і вона має значний потенціал для подальшого розвитку та розширення її можливостей.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Берк, Д. HTML, CSS, JavaScript для веб-розробників. — Київ: BHV, 2023. — 650 с. — С. 45–52.
2. Васильєв, О. С. Системи керування базами даних: теорія та практика. — Харків: Фоліо, 2022. — 320 с. — С. 110.
3. Григоренко, П. В. Основи веб-програмування з JavaScript. — Львів: Магнолія, 2021. — 280 с. — С. 78, 85.
4. Державна служба спеціального зв'язку та захисту інформації України. Закони України про захист інформації [Електронний ресурс]. — URL: https://www.sssc.gov.ua/ (дата звернення: 22.05.2025).
5. Ковальчук, А. П. Проектування інформаційних систем. — Київ: Центр учбової літератури, 2020. — 410 с. — С. 30–35.
6. Мельник, І. М. Безпека веб-застосувань: основи та кращі практики. — Дніпро: Акцент, 2023. — 250 с. — С. 120–125.
7. Офіційна документація HTML5 [Електронний ресурс]. — URL: https://html.spec.whatwg.org/ (дата звернення: 22.05.2025).
8. Офіційна документація JavaScript (MDN Web Docs) [Електронний ресурс]. — URL: https://developer.mozilla.org/uk/docs/Web/JavaScript (дата звернення: 22.05.2025).
9. Пасічник, В. В. Алгоритми та структури даних. — Львів: Новий Світ-2000, 2021. — 390 с. — С. 60–62.
10. Сміт, Дж. Клієнтська веб-розробка: сучасні підходи. — Київ: Основа, 2022. — 300 с. — С. 90.
11. Тихонов, В. О. Архітектура програмного забезпечення. — Одеса: Латстар, 2020. — 270 с. — С. 150–155.
12. Хайнес, Р. Проектування інтерфейсів користувача. — Київ: БукРі, 2023. — 220 с. — С. 25–28.
13. Щербак, М. Л. Тестування програмного забезпечення. — Харків: Торнадо, 2022. — 210 с. — С. 70.

14. W3C. Cascading Style Sheets (CSS) Current Work [Електронний ресурс]. — URL: https://www.w3.org/Style/CSS/current-work (дата звернення: 22.05.2025).
15. Stack Overflow. Questions & Answers for Developers [Електронний ресурс]. — URL: https://stackoverflow.com/ (дата звернення: 22.05.2025)
16. OpenAI. ChatGPT. [Електронний ресурс]. — OpenAI, 2025. — URL: https://openai.com/chatgpt (дата звернення: 22.05.2025).
17. Google. Gemini. [Електронний ресурс]. — Google, 2025. — URL: https://gemini.google.com/ (дата звернення: 22.05.2025).

ДОДАТКИ

ДОДАТОК А. АНАЛІЗ КОНКУРЕНТНИХ РІШЕНЬ ДЛЯ ДИСТАНЦІЙНОГО ГОЛОСУВАННЯ

У таблиці нижчє наведено порівняльний аналіз ключових характеристик розробленої Інформаційної системи дистанційного голосування з існуючими популярними сервісами, які пропонують функціонал опитувань та голосувань. Цей аналіз дозволяє виділити переваги та особливості власного рішення.

Критерій	Google Forms	SurveyMonkey	Мій веб-додаток
Тип застосунку	Загальний сервіс для опитувань та форм	Професійний сервіс для маркетингових та соціологічних досліджень	Клієнтська інформаційна система дистанційного голосування (SPA)
Призначення	Збір інформації, прості опитування, вікторини	Детальні опитування, збір даних, аналіз ринку	Проведення голосувань та опитувань з автентифікацією учасників
Вимога до реєстрації	Для учасників: Ні (якщо не встановлено обмежень)	Для учасників: Залежить від налаштувань, часто ні	Для учасників: Так (обов'язкова реєстрація/авторизація)

Ролі користувачів	Власник форми, редактори, відповідачі	Створювач опитувань, адміністратори акаунтів, респонденти	Адміністратор, Користувач (Виборець)
Вартість	Безкоштовно	Є безкоштовний тариф з обмеженнями, платні професійні тарифи	Безкоштовно (локальна розробка, відкритий код)
Зберігання даних	Хмарне сховище (Google Drive)	Хмарна база даних сервісу	localStorage браузера (локальне зберігання на пристрої користувача)
Масштабованість	Висока (хмарна інфраструктура Google)	Висока (професійна хмарна платформа)	Низька (обмеження localStorage, дані не синхронізуються між пристроями)
Контроль унікальності голосу	Обмежений (може бути обійдений через багаторазові відповіді з різних	Є опції (залежить від тарифу та	Є (прив'язка голосу до email

	пристроїв/браузерів)	налаштувань: IP-адреси, cookies)	зареєстрованого користувача)
Механізм автентифікації	Залежить від облікового запису Google	Власні механізми, часто через email/пароль	Просте хешування MD5 (для демонстрації)
Адміністративні функції	Створення/редагування форм, перегляд агрегованих відповідей	Розширене управління опитуваннями, користувачами, аналітика	Створення, завершення, видалення опитувань, перегляд детальних результатів
Візуалізація результатів	Автоматичні діаграми та графіки	Розширені можливості візуалізації, звітність	Прогрес-бари з відсотковим співвідношенням голосів
Адаптивний дизайн	Так	Так	Так
Складність розробки	Низька (для використання)	Низька (для використання)	Середня (для розробки клієнтського SPA)
Безпека даних/автентифікації	Висока (стандарти Google)	Висока (професійні стандарти)	Базова (для демонстрації, MD5 не є надійним для реальних систем)

Передача даних	HTTP(S) запити до серверів Google	HTTP(S) запити до серверів SurveyMonkey	Немає (всі дані обробляються та зберігаються локально)

Табл. А.1. Порівняльна таблиця

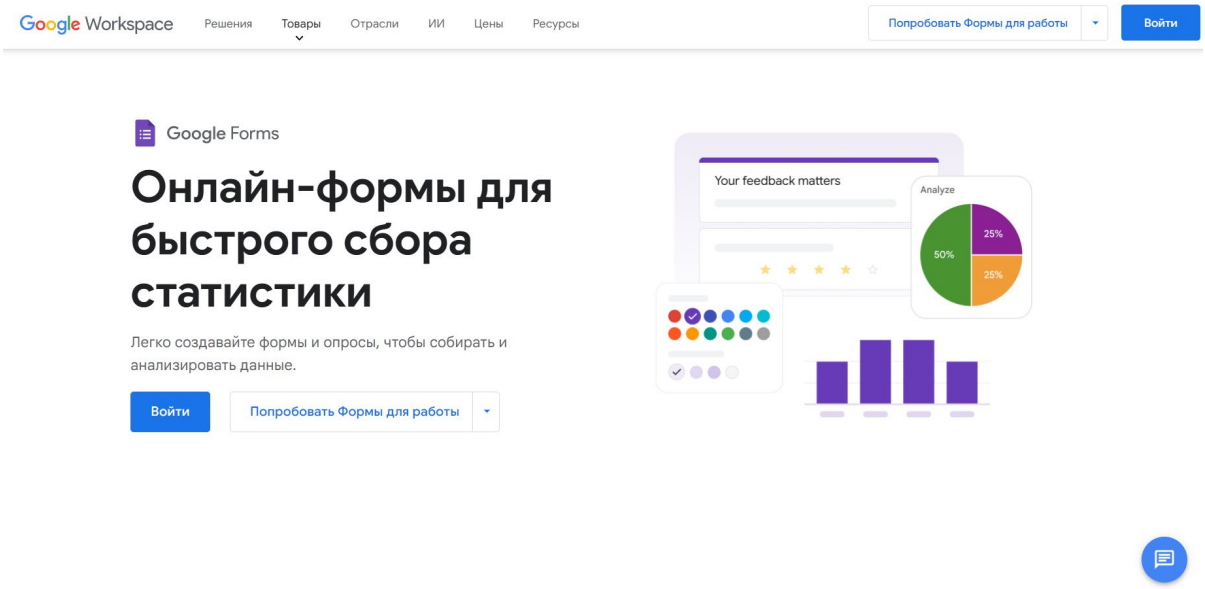

Рис. А.1. Головна сторінка Google Forms

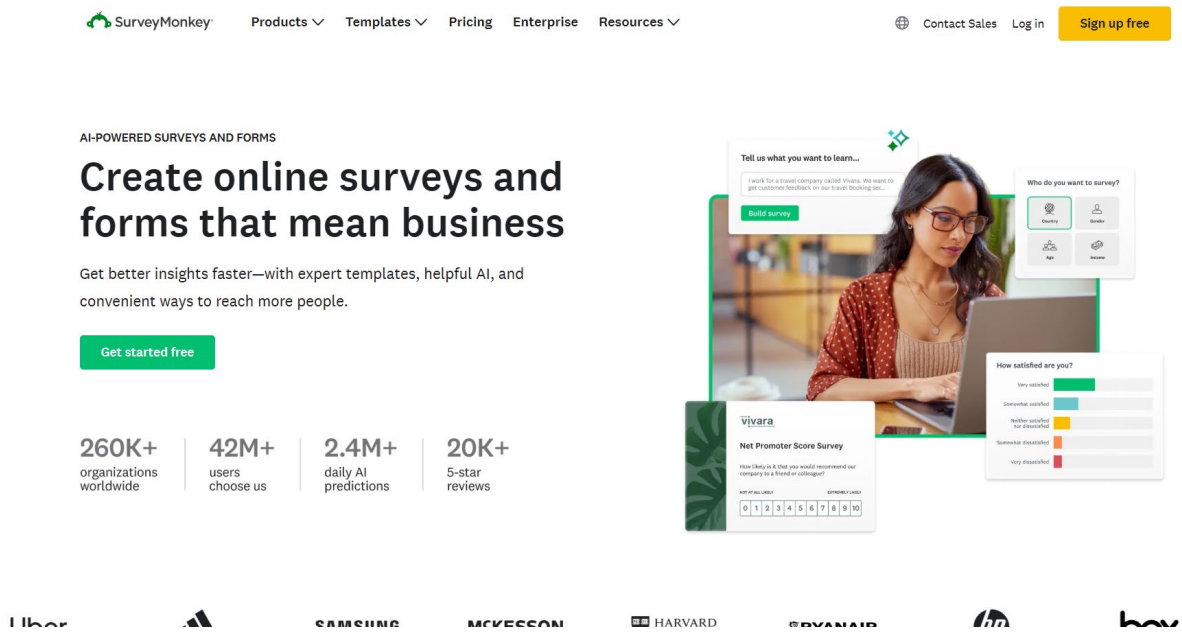

Рис. А.2. Головна сторінка SurveyMonkey

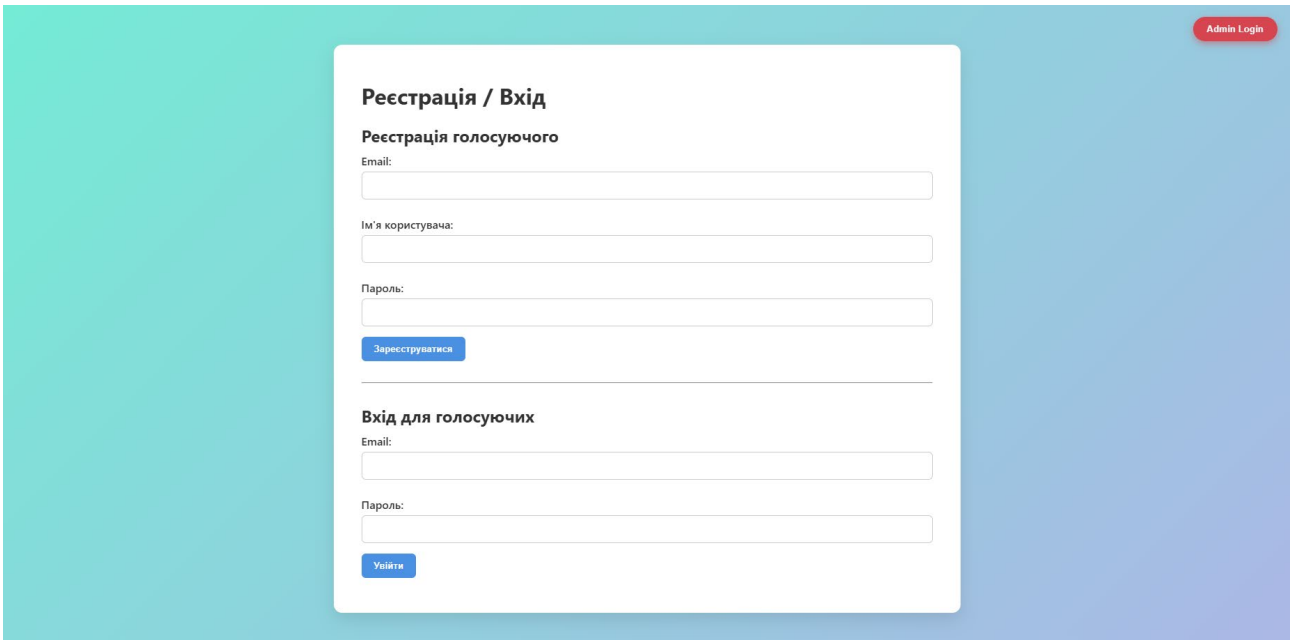

Рис. А.3. Головна сторінка моєї системи

ДОДАТОК Б. СКРІНШОТИ РОБОТИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДИСТАНЦІЙНОГО ГОЛОСУВАННЯ

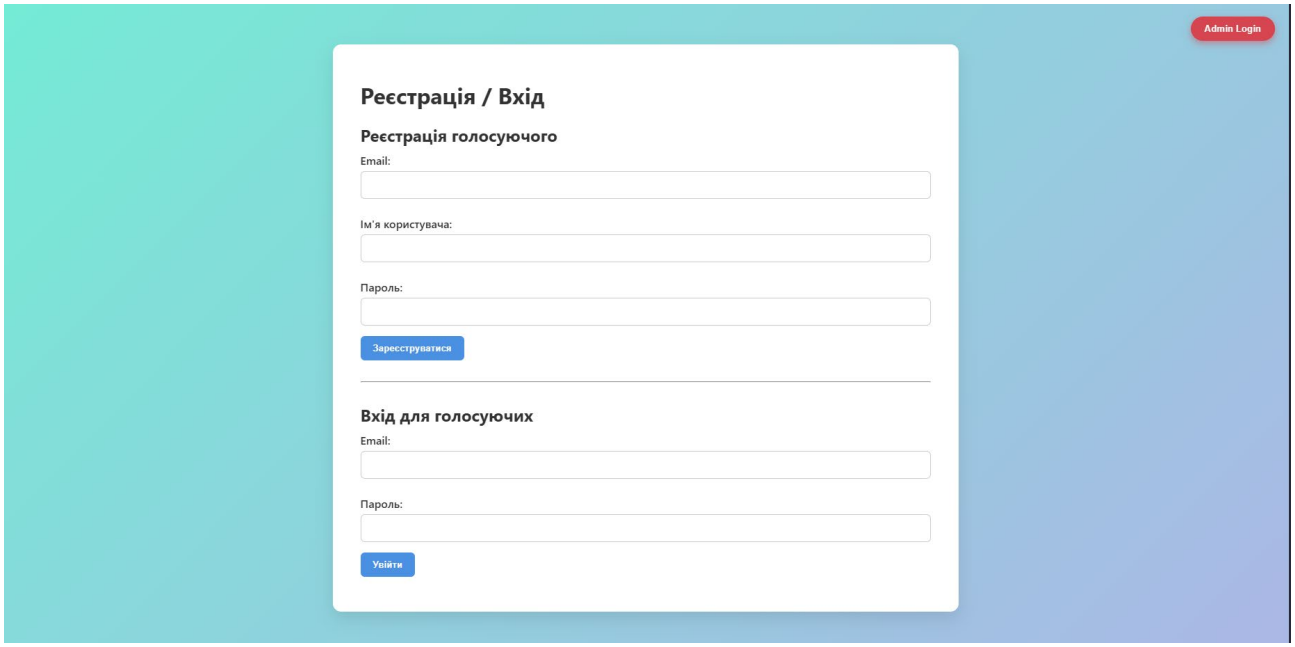

Рис. Б.1. Екран авторизації/реєстрації

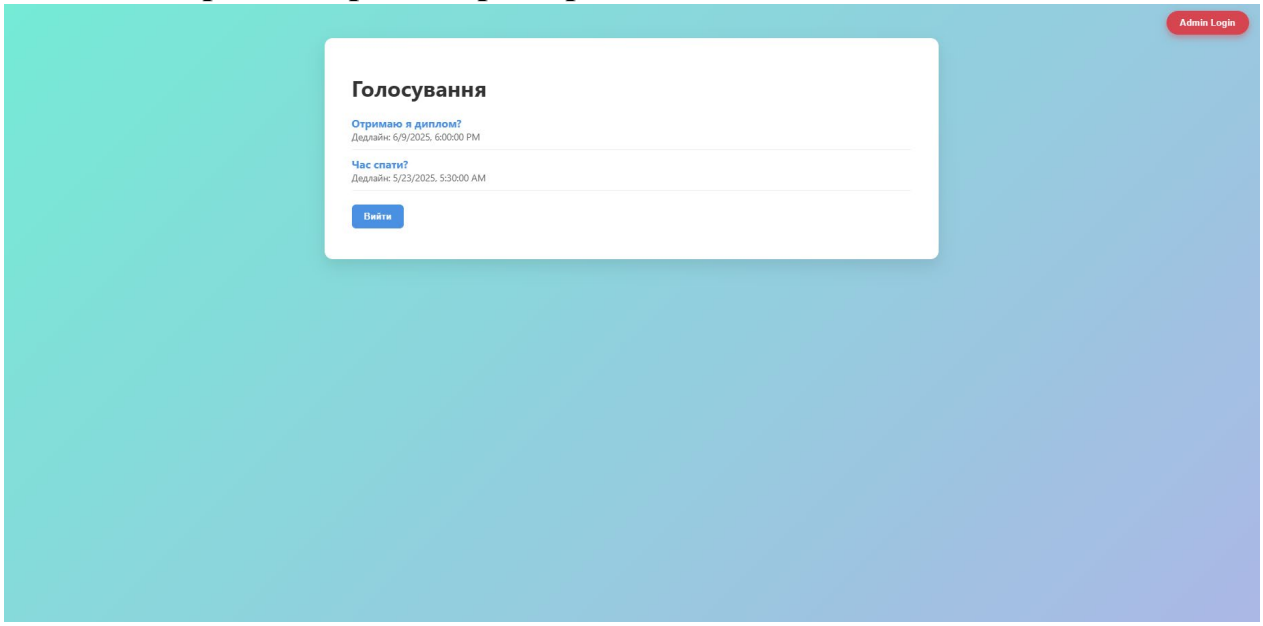

Рис. Б.2. Панель користувача: список активних опитувань

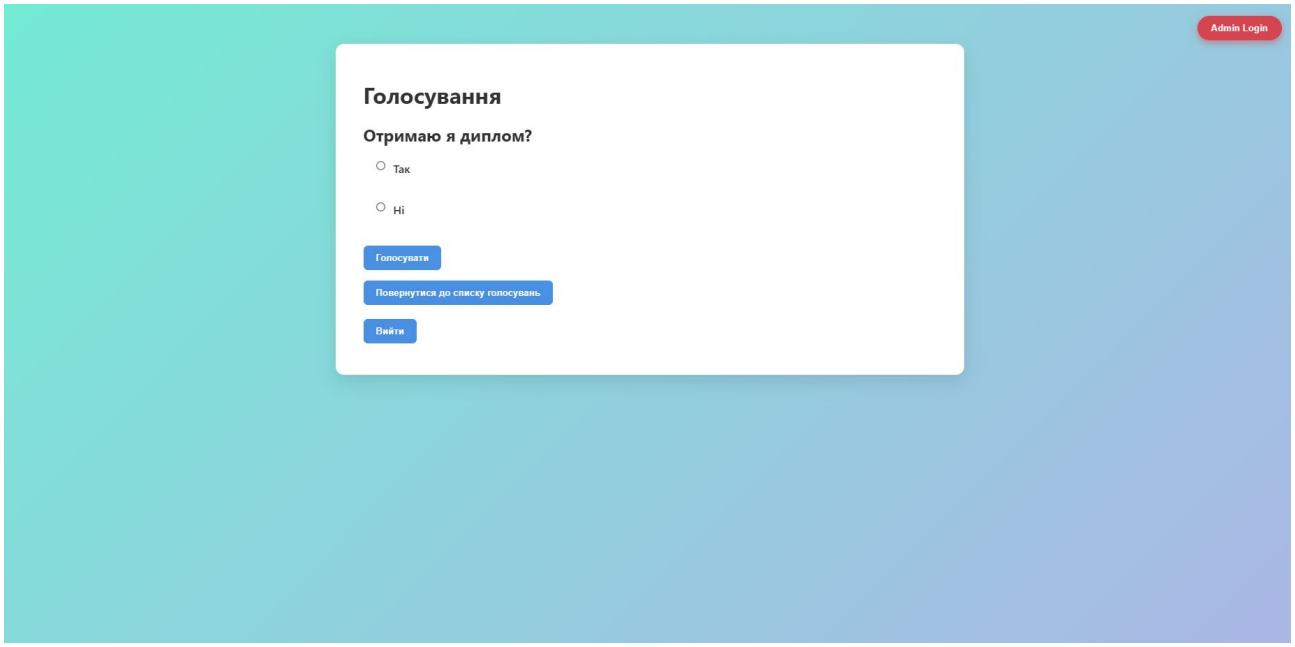

Рис. Б.3. Сторінка голосування за опитуванням

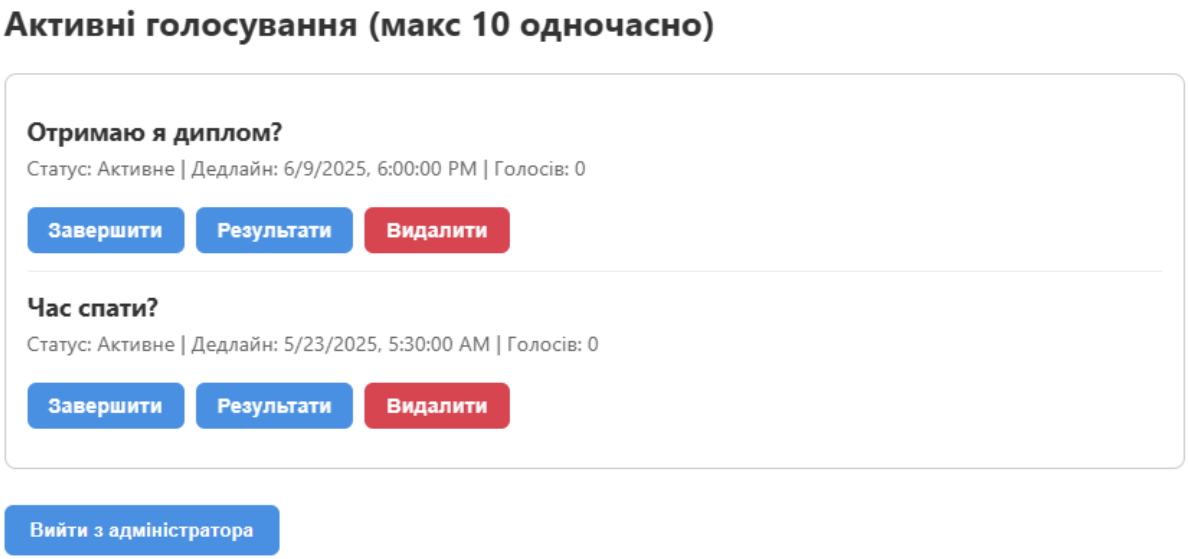

Рис. Б.4 Панель адміністратора зі списком всіх опитувань.

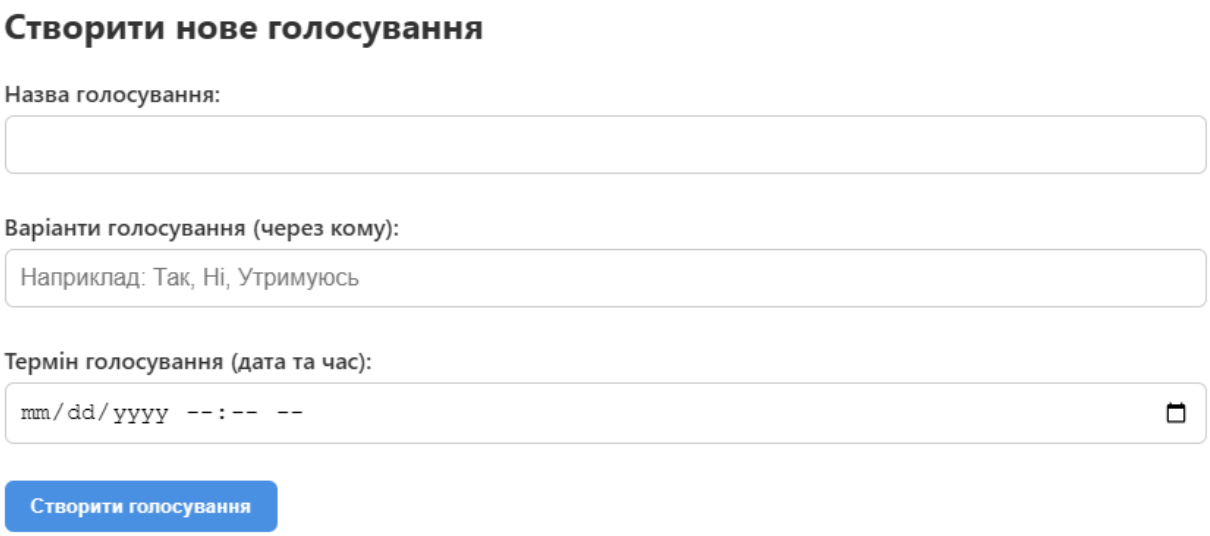

Рис. Б.5 Форма створення нового опитування (для адміністратора)

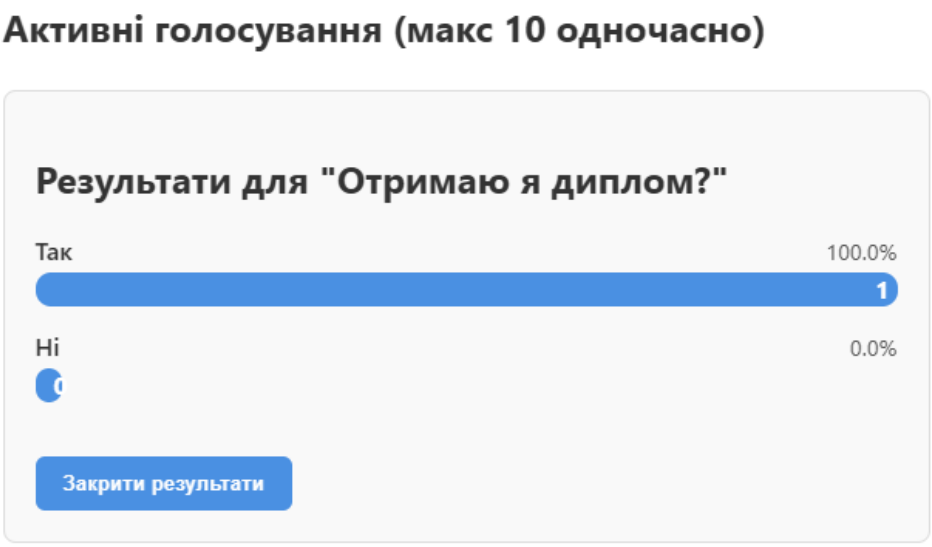

Рис. Б.6 Екран результатів голосування

ДОДАТОК В. ПРОГРАМНИЙ КОД ІНФОРМАЦІЙНОЇ СИСТЕМИ


```
{
  "email": "user1@example.com",
  "username": "Звичайний Користувач",
  "passwordHash": "e10adc3949ba59abbe56e057f20f883e",
  "role": "user"
 },
 {
  "email": "admin@example.com",
  "username": "Адміністратор Системи",
  "passwordHash": "21232f297a57a5a743894a0e4a801fc3",
  "role": "admin"
 }
```


Додаток В.1. JSON-структура для даних користувачів


```
[
 {
  "id": "1716382000000",
  "title": "Вибір програмної мови для наступного проекту",
  "options": ["JavaScript", "Python", "Java", "C#"],
  "deadline": "2025-06-15T23:59",
  "votes": {
   "user1@example.com": 0,
   "user2@example.com": 1,
   "user3@example.com": 0
  },
```

```
    "active": true
  },
  {
    "id": "1716382500000",
    "title": "Оцінка якості навчання",
    "options": ["Висока", "Середня", "Низька"],
    "deadline": "2025-05-10T12:00",
    "votes": {
      "user1@example.com": 0,
      "admin@example.com": 0,
      "user2@example.com": 1
    },
    "active": false
  }
]
```

Додаток В.2. JSON-структура для даних опитувань

```
<script>
    // Функції для роботи з localStorage
    const loadData = (key) => {
        const data = localStorage.getItem(key);
        return data ? JSON.parse(data) : [];
    };

    const saveData = (key, data) => {
        localStorage.setItem(key, JSON.stringify(data));
    };

    // Ініціалізація даних
```

```
    let users = loadData('users');
    let polls = loadData('polls');
    let currentUser = JSON.parse(localStorage.getItem('currentUser'));

    // Якщо адмін не існує, створити його
    if (!users.find(u => u.role === 'admin')) {
        users.push({
            email: 'admin@example.com',
            username: 'Admin',
            passwordHash: '21232f297a57a5a743894a0e4a801fc3', // MD5 hash of 'admin'
            role: 'admin'
        });
        saveData('users', users);
    }
</script>
```

Додадток В.3. Реалізація основних функцій для взаємодії з localStorage

```
<script>
    // ... (попередні модулі та оголошення елементів DOM) ...

    // Функція для простого хешування пароля (для демонстраційних цілей)
    const simpleHash = (str) => {
        let hash = 0;
        for (let i = 0; i < str.length; i++) {
            const char = str.charCodeAt(i);
            hash = ((hash << 5) - hash) + char;
            hash |= 0; // Convert to 32bit integer
        }
```

```
    return hash.toString();
  };

  // Обробка реєстрації
  registrationForm.addEventListener('submit', (e) => {
    e.preventDefault();
    const email = regEmailInput.value.trim();
    const username = regUsernameInput.value.trim();
    const password = regPasswordInput.value;
    const confirmPassword = regConfirmPasswordInput.value;

    if (!email || !username || !password || !confirmPassword) {
      alert('Будь ласка, заповніть всі поля!');
      return;
    }
    if (password !== confirmPassword) {
      alert('Паролі не співпадають!');
      return;
    }
    if (users.some(u => u.email === email)) {
      alert('Користувач з таким email вже зареєстрований!');
      return;
    }

    const hashedPassword = simpleHash(password);
    users.push({ email, username, passwordHash: hashedPassword, role: 'user'
});
    saveData('users', users);
    alert('Реєстрація успішна! Тепер увійдіть.');
```

```
    showAuthForm('login'); // Переключитися на форму входу
    registrationForm.reset();
  });

  // Обробка входу
  loginForm.addEventListener('submit', (e) => {
    e.preventDefault();
    const email = loginEmailInput.value.trim();
    const password = loginPasswordInput.value;
    const hashedPassword = simpleHash(password);

    const foundUser = users.find(u => u.email === email && u.passwordHash
=== hashedPassword);

    if (foundUser) {
      currentUser = foundUser;
      localStorage.setItem('currentUser', JSON.stringify(currentUser));
      alert(`Ласкаво просимо, ${currentUser.username}!`);
      if (currentUser.role === 'admin') {
        authContainer.classList.add('hidden');
        adminPanel.classList.remove('hidden');
        renderAdminPollList();
      } else {
        authContainer.classList.add('hidden');
        votingContainer.classList.remove('hidden');
        renderPollsForUser();
      }
    } else {
      alert('Невірний email або пароль.');
```

```
    }
  });

  // Обробка виходу
  logoutBtn.addEventListener('click', () => {
    currentUser = null;
    localStorage.removeItem('currentUser');
    authContainer.classList.remove('hidden');
    votingContainer.classList.add('hidden');
    adminPanel.classList.add('hidden');
    // Очистити вміст відображення, якщо потрібно
    pollList.innerHTML = '';
    createPollFormContainer.classList.add('hidden'); // Додатково приховати форму створення опитування
    resultsArea.classList.add('hidden'); // Приховати результати
    alert('Ви успішно вийшли з системи.');
  });
</script>
```

Додаток В.4. Ключові механізми автентифікації та авторизації

```
<script>
  // ... (попередні модулі) ...

  // Обробка створення опитування
  createPollForm.addEventListener('submit', (e) => {
    e.preventDefault();
    const title = document.getElementById('pollTitle').value.trim();
    const optionInputs = document.querySelectorAll('.option-input');
    const options = Array.from(optionInputs).map(input => input.value.trim()).filter(val => val !== '');
```

```
    const deadline = document.getElementById('pollDeadline').value;

    if (!title || options.length < 2 || !deadline) {
        alert('Будь ласка, заповніть всі обов\'язкові поля (мінімум 2 варіанти відповідей).');
        return;
    }

    const newPoll = {
        id: Date.now().toString(), // Унікальний ID на основі мітки часу
        title,
        options,
        deadline,
        votes: {}, // Об'єкт для зберігання голосів користувачів
        active: true
    };
    polls.push(newPoll);
    saveData('polls', polls);
    alert('Опитування успішно створено!');
    createPollForm.reset();
    clearOptionInputs(); // Очистити додаткові поля опцій
    renderAdminPollList(); // Оновити список
    showAdminPanel('pollList'); // Повернутись до списку опитувань
});

// Функція для відображення списку опитувань в адмін-панелі
const renderAdminPollList = () => {
    adminPollList.innerHTML = '';
    const now = new Date();
```

```
    if (polls.length === 0) {
        adminPollList.innerHTML = '<p class="no-items-message">Немає створених опитувань.</p>';
        return;
    }

    polls.forEach(poll => {
        // Оновлення статусу опитування за дедлайном
        if (poll.active && new Date(poll.deadline) < now) {
            poll.active = false;
            saveData('polls', polls);
        }

        const pollItem = document.createElement('div');
        pollItem.classList.add('poll-item');
        pollItem.innerHTML = `
            <h3>${poll.title}</h3>
            <p>Статус: <span class="${poll.active ? 'active-status' : 'inactive-status'}">${poll.active ? 'Активне' : 'Завершене'}</span></p>
            <p>Дедлайн: ${new Date(poll.deadline).toLocaleString('uk-UA')}</p>
            <div class="poll-actions">
                ${poll.active ? `<button class="finish-poll-btn" data-id="${poll.id}">Завершити</button>` : ''}
                <button class="delete-poll-btn" data-id="${poll.id}">Видалити</button>
                <button class="view-results-btn" data-id="${poll.id}">Результати</button>
            </div>
        `;
```

```
    adminPollList.appendChild(pollItem);
});

// Додаємо обробники подій для кнопок
adminPollList.querySelectorAll('.finish-poll-btn').forEach(button => {
    button.onclick = (e) => {
        const pollId = e.target.dataset.id;
        const pollIndex = polls.findIndex(p => p.id === pollId);
        if (pollIndex !== -1 && polls[pollIndex].active) {
            polls[pollIndex].active = false;
            saveData('polls', polls);
            alert('Опитування успішно завершено!');
            renderAdminPollList(); // Перерендерити список
        }
    };
});

adminPollList.querySelectorAll('.delete-poll-btn').forEach(button => {
    button.onclick = (e) => {
        const pollId = e.target.dataset.id;
        if (confirm('Ви впевнені, що хочете видалити це опитування?')) {
            polls = polls.filter(p => p.id !== pollId);
            saveData('polls', polls);
            alert('Опитування успішно видалено!');
            renderAdminPollList();
        }
    };
});
```

```
    adminPollList.querySelectorAll('.view-results-btn').forEach(button => {
        button.onclick = (e) => {
            const pollId = e.target.dataset.id;
            const poll = polls.find(p => p.id === pollId);
            if (poll) {
                renderPollResults(poll); // Виклик функції для відображення результатів
            }
        };
    });
};
</script>
```

Додаток В.5. Основні аспекти управління опитуваннями адміністратором

```
<script>
// ... (попередні модулі) ...


// Функція для відображення опитувань для звичайного користувача
const renderPollsForUser = () => {
    pollList.innerHTML = ''; // Очистити попередній список
    const now = new Date();
    // Фільтруємо лише активні опитування, що ще не закінчились
    const activePolls = polls.filter(p => p.active && new Date(p.deadline) >= now);


    if (activePolls.length === 0) {
```

```
        pollList.innerHTML = '<p class="no-items-message">Наразі немає активних опитувань.</p>';
        return;
    }

    activePolls.forEach(poll => {
        const pollItem = document.createElement('div');
        pollItem.classList.add('poll-item');
        // Перевіряємо, чи поточний користувач вже проголосував в цьому опитуванні
        const hasVoted = poll.votes && poll.votes[currentUser.email];
        pollItem.innerHTML = `
            <h3>${poll.title}</h3>
            <p>Дедлайн: ${new Date(poll.deadline).toLocaleString('uk-UA')}</p>
            <div class="poll-actions">
                <button class="vote-btn" data-id="${poll.id}" ${hasVoted ? 'disabled' : ''}>
                    ${hasVoted ? 'Ви вже проголосували' : 'Проголосувати'}
                </button>
                ${hasVoted || !poll.active ? `<button class="view-results-user-btn" data-id="${poll.id}">Переглянути результати</button>` : ''}
            </div>
        `;
        pollList.appendChild(pollItem);
```

```
    });

    // Додаємо обробники для кнопок
    pollList.querySelectorAll('.vote-btn').forEach(button => {
        button.onclick = (e) => {
            const pollId = e.target.dataset.id;
            showPollDetails(pollId); // Показати деталі опитування для голосування
        };
    });

    pollList.querySelectorAll('.view-results-user-btn').forEach(button => {
        button.onclick = (e) => {
            const pollId = e.target.dataset.id;
            const poll = polls.find(p => p.id === pollId);
            if (poll) {
                renderPollResults(poll);
            }
        };
    });
};

// Функція відображення деталей опитування та голосування
```

```
    const showPollDetails = (pollId) => {
        const poll = polls.find(p => p.id === pollId);
        if (!poll) return;

        // Перевірка, чи опитування ще активне і чи користувач не голосував
        const alreadyVoted = poll.votes[currentUser.email] !== undefined;
        const isDisabled = !poll.active || alreadyVoted || (new Date(poll.deadline) < new Date());

        pollDetailsContent.innerHTML = `
            <h2>${poll.title}</h2>
            <p>Дедлайн: ${new Date(poll.deadline).toLocaleString('uk-UA')}</p>
            <div id="optionsContainer">
                ${poll.options.map((option, index) => `
                    <label class="option-label">
                        <input type="radio" name="voteOption" value="${index}" ${isDisabled ? 'disabled' : ''}>
                        ${option}
                    </label>
                `).join('')}
            </div>
            <button id="submitVoteBtn" ${isDisabled ? 'disabled' : ''}>Проголосувати</button>
```

```
        <button id="backToPollsBtn">Назад до списку</button>
        ${alreadyVoted ? '<p class="info-message">Ви вже проголосували в цьому опитуванні.</p>' : ''}
        ${!poll.active || (new Date(poll.deadline) < new Date()) ? '<p class="info-message">Опитування завершене або неактивне.</p>' : ''}
    `;

    // Перемикання видимості контейнерів
    votingContainer.classList.add('hidden'); // Приховати список
    pollDetails.classList.remove('hidden'); // Показати деталі

    // Обробник для кнопки голосування
    if (!isDisabled) { // Прикріплюємо обробник тільки якщо кнопка активна
        document.getElementById('submitVoteBtn').onclick = () => {
            const selectedOption = document.querySelector('input[name="voteOption"]:checked');
            if (selectedOption) {
                const optionIndex = parseInt(selectedOption.value);
                const pollIndex = polls.findIndex(p => p.id === pollId);
                if (pollIndex !== -1) {
                    polls[pollIndex].votes[currentUser.email] = optionIndex; // Фіксація голосу
                    saveData('polls', polls);
                    alert('Ваш голос успішно прийнято!');
```

```
                pollDetails.classList.add('hidden'); // Сховати деталі голосування
                renderPollResults(polls[pollIndex]); // Показати результати
            }
        } else {
            alert('Будь ласка, оберіть варіант.');
        }
    };
}

// Обробник для кнопки "Назад до списку"
document.getElementById('backToPollsBtn').onclick = () => {
    pollDetails.classList.add('hidden');
    votingContainer.classList.remove('hidden');
    renderPollsForUser(); // Перерендерити список на випадок змін
};
};

// Функція для візуалізації результатів голосування
const renderPollResults = (poll) => {
    resultsTitle.textContent = `Результати опитування: "${poll.title}"`;
    resultsContent.innerHTML = '';
```

```
    const totalVotes = Object.keys(poll.votes).length;
    const counts = poll.options.map((_, index) =>
      Object.values(poll.votes).filter(voteIndex => voteIndex === index).length
    );

    if (totalVotes === 0) {
      resultsContent.innerHTML = '<p class="no-items-message">Ще ніхто не проголосував у цьому опитуванні.</p>';
    } else {
      poll.options.forEach((option, i) => {
        const percent = totalVotes > 0 ? (counts[i] / totalVotes * 100).toFixed(2) : 0;
        const resultBarContainer = document.createElement('div');
        resultBarContainer.classList.add('result-bar-container');
        resultBarContainer.innerHTML = `
          <div class="result-label">${option}</div>
          <div class="result-bar-wrapper">
            <div class="result-bar" style="width: ${percent}%"></div>
            <div class="result-percentage">${percent}% (${counts[i]} голосів)</div>
          </div>
        `;
        resultsContent.appendChild(resultBarContainer);
```

```
        });
    }
    pollList.classList.add('hidden');
    createPollFormContainer.classList.add('hidden'); // Приховати форму створення
    adminPanel.classList.add('hidden'); // Приховати адмін панель, якщо відображаються результати
    resultsArea.classList.remove('hidden'); // Показати область результатів
};

closeResultsBtn.onclick = () => {
    resultsArea.classList.add('hidden');
    if (currentUser && currentUser.role === 'admin') {
        adminPanel.classList.remove('hidden');
        renderAdminPollList();
    } else {
        votingContainer.classList.remove('hidden');
        renderPollsForUser();
    }
};
</script>
```

Додаток В.6. Основна функціональність для користувачів та відображення результатів.

ДОДАТОК Г. ІЛЮСТРАТИВНІ МАТЕРІАЛИ

Рис. Г.1. Діаграма варіантів використання Інформаційної системи дистанційного голосування