

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

**Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій**

Курсова робота

з дисципліни «Програмування»

на тему: «Бюро знахідок»

Виконав:

студент 1 курсу, групи ІА-33

Матвієнко Богдан Олексійович

(підпис)

Керівник:

асистент кафедри ІСТ

Мягкий Михайло Юрійович

(підпис)

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних
посилань.

Студент

(підпис)

Київ – 2024 року

ЗМІСТ

ВСТУП	3
1 ВИМОГИ ДО СИСТЕМИ	5
1.1 Функціональні вимоги до системи	5
1.2 Нефункціональні вимоги до системи	5
2 СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ	6
2.1 Діаграма прецедентів	6
2.2 Опис сценаріїв використання системи	7
3 АРХІТЕКТУРА СИСТЕМИ	10
4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ	12
4.1 Загальна структура проекту	12
4.2 Компоненти рівня доступу до даних	13
4.3 Компоненти рівня бізнес-логіки	16
4.4 Компоненти рівня інтерфейсу користувача	19
ВИСНОВКИ	22
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	24
ДОДАТОК А Лістинг програми	25

ВСТУП

Уявіть собі, як часто ми чуємо історії про загублені гаманці, ключі чи мобільні телефони. У стрімкому ритмі сучасного життя, де кожен з нас постійно поспішає, втрата особистих речей стає звичайним явищем. Але що, якби існувала система, яка б допомагала знайти ці речі швидко і без зайвих клопот?

Сьогодні, коли традиційні методи пошуку, такі як оголошення на стовпах чи пости в соціальних мережах, часто не дають бажаного результату, необхідність в інноваційних рішеннях стає особливо актуальною. "Бюро знахідок" — це не просто інформаційна система, це надія на повернення загублених речей, це шанс для кожного відчувати радість від знайдення дорогоцінного предмету.

Метою створення системи "Бюро знахідок" є розробка ефективного, інтуїтивного і зручного сервісу, який допоможе людям швидко знаходити свої загублені речі. Ми прагнемо створити платформу, де кожен користувач, будь то той, хто знайшов річ, чи той, хто її втратив, може легко і швидко знайти необхідну інформацію. Ця система повинна стати надійним помічником у розв'язанні проблеми втрат у повсякденному житті.

Для досягнення цієї амбітної мети, система "Бюро знахідок" має вирішити кілька ключових задач:

1. Реєстрація знахідок: Створення зручного інтерфейсу, який дозволить користувачам легко додавати інформацію про знайдені речі. Кожен запис включатиме детальний опис, ключові слова та контактну інформацію, щоб власники могли швидко ідентифікувати свою річ.
2. Видалення знахідок: Забезпечення можливості видалення записів, коли речі повертаються власникам або якщо інформація стає неактуальною. Це допоможе підтримувати базу даних в актуальному стані, роблячи пошук ще більш ефективним.
3. Пошук за ключовими словами: Розроблення потужного та швидкого механізму пошуку, який дозволить користувачам знаходити необхідну

інформацію за допомогою ключових слів. Це зменшить час на пошук і збільшить шанси на успішне повернення речей.

4. Перегляд інформації про знахідки: Створення зручного інтерфейсу для перегляду детальної інформації про знайдені речі, включаючи опис та контактні дані того, хто знайшов річ. Це зробить процес зв'язку між власником і знахідником максимально простим та ефективним.

Уявіть собі, скільки радості та полегшення може принести система "Бюро знахідок" людям, які втратили свої речі. Ця система — це не просто технологічне рішення, це крок назустріч людяності, взаємодопомозі та довірі. Вона допоможе повернути втрачені речі, зберегти спокій, і можливо, навіть зробити наш світ трохи добрішим.

1 ВИМОГИ ДО СИСТЕМИ

1.1 Функціональні вимоги до системи

Система має відповідати наступним функціональним вимогам:

- користувач повинен мати можливість переглядати знахідки та інформації про них;
- користувач повинен мати можливість здійснювати пошук по ключовим словам у назві та/або у описі знахідок;
- користувач повинен мати можливість створення та/або видалення знахідок.

1.2 Нефункціональні вимоги до системи

Система має відповідати наступним функціональним вимогам:

- система повинна мати відкриту архітектуру;
- система повинна мати веб-інтерфейс;
- інтерфейс користувача має бути зручним та інтуїтивно-зрозумілим;
- система повинна бути крос-платформною.

2 СЦЕНАРІЇ ВИКОРИСТАННЯ СИСТЕМИ

2.1 Діаграма прецедентів

Діаграма прецедентів системи представлена на рис. 2.1.

Акторами є користувачі системи: користувач.

Детально усі сценарії використання описані у наступному підрозділі.

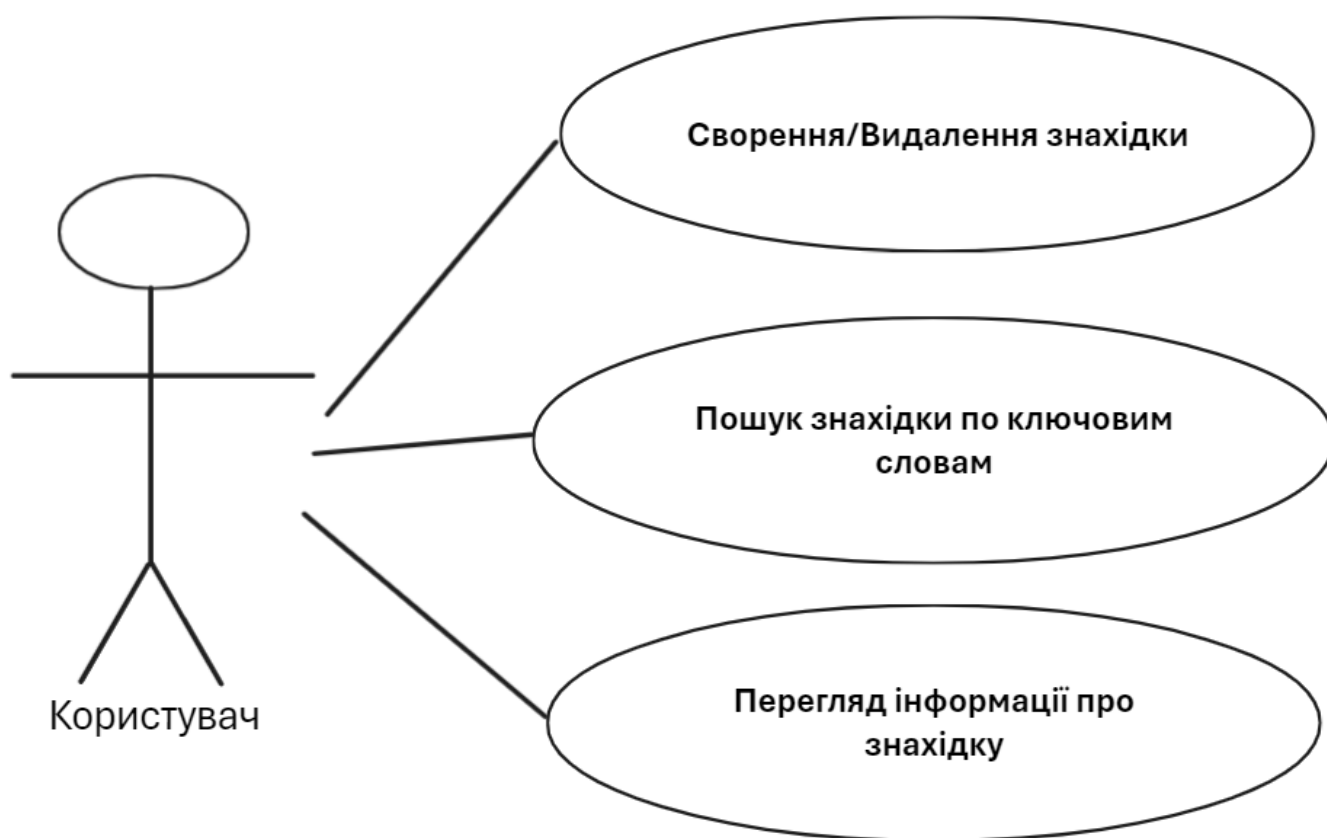


Рисунок 2.1 – Діаграма прецедентів

2.2 Опис сценаріїв використання системи

Детальні описи сценаріїв використання наведено у таблицях 2.1 – 2.3.

Таблиця 2.1 – Сценарій використання «Створення/Видалення знахідки»

Назва	Створення/Видалення знахідки
ID	1
Опис	Користувач, використовуючи кнопку «Додати знахідку» вказує дані про знахідку та створює її на сайті, також може натиснувши на кнопку в знахідці її видалити
Актори	Користувач
Вигоди компанії	Користувачі створюють активність сайту, легко створюють або видаляють нові знахідки та створюють контент для сайту
Частота користування	Часто
Тригери	Користувач натискає кнопку для створення знахідки або кнопку для видалення знахідки
Передумови	Кнопка створення або видалення знахідки
Постумови	Користувач створює або видаляє знахідку
Основний розвиток	Користувач натискає кнопку для створення знахідку на головній сторінці сайту, вказує дані знахідки та натискає кнопку створення знахідки
Альтернативні розвитку	Користувач натискає на знахідку, яку він хоче видалити, і натискає на кнопку видалення знахідки
Виняткові ситуації	—

В таблиці 2.2 представлений сценарій використання «Пошук знахідки по ключовим словам»

Таблиця 2.2 – Сценарій використання «Пошук знахідки по ключовим словам»

Назва	Пошук знахідки по ключовим словам
ID	2
Опис	Користувач, використовуючи поле для пошуку, шукає знахідку, що відповідає ключовим словам
Актори	Користувач
Вигоди компанії	Якщо користувачі не можуть шукати знахідки, то вони навряд чи продовжать користуватися сервісом
Частота користування	Часто
Тригери	Користувач вводить пошуковий запит у полі для пошуку
Передумови	Пошукове поле доступне на головній сторінці сайту
Постумови	Користувач бачить знизу результати пошуку
Основний розвиток	Користувач вводить запит у пошукову строку, натискає на кнопку пошуку
Альтернативні розвитки	—
Виняткові ситуації	—

В таблиці 2.3 представлений сценарій використання «Перегляд інформації про знахідку»

Таблиця 2.3 – Сценарій використання «Перегляд інформації про знахідку»

Назва	Перегляд інформації про знахідку
ID	3
Опис	Користувач переглядає інформацію по знахідку(заголовок, текст, контакти)
Актори	Користувач
Вигоди компанії	Сайт знахідок без можливості перегляду знахідок не буде користуватися попитом
Частота користування	Постійно
Тригери	Користувач натискає на знахідку
Передумови	Користувач бачить знахідку, яка його цікавить
Постумови	Користувач переглядає інформацію про знахідку
Основний розвиток	Користувач натискає на знахідку
Альтернативні розвитку	—
Виняткові ситуації	—

3 АРХІТЕКТУРА СИСТЕМИ

Загальна архітектура системи наведена на рис. 3.1.

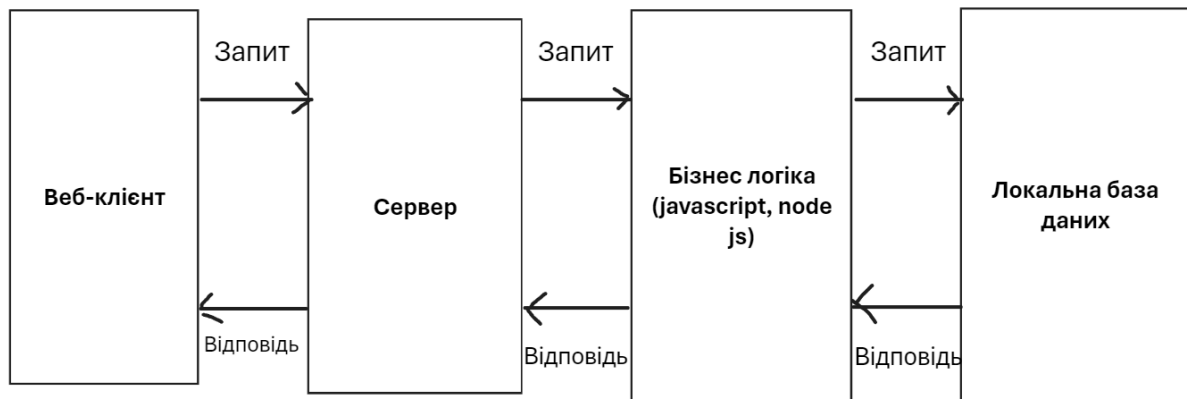


Рисунок 3.1 – Загальна архітектура системи

Система складається з наступних елементів:

- графічний інтерфейс;
- серверна частина;
- база даних.

Графічний інтерфейс необхідний для взаємодії з користувачем. HTTP запит надходить до серверної частини, де оброблюється і повертається відповідь. На серверній частині виконується основна логіка системи. Дані, отриманні з графічного інтерфейсу валідуються, конвертуються. Також, серверна частина формує запит до бази даних та оброблює відповідь і передає її до графічного інтерфейсу. База даних зберігає дані, які були сформовані на серверній частині та повертає їх у разі запиту.

До серверної частини належать наступні елементи:

- контролер;
- модель та вигляд;
- сервіс;

На контролер надходять дані з графічного інтерфейсу. З контролеру, дані формуються в сервісі для запиту в javascript програми. З них дані надсилаються до бази даних і зберігаються. Також в контролері формується вид, тобто об'єкт і його ім'я для відображення на графічному інтерфейсі.

4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

4.1 Загальна структура проекту

Загальна структура проекту представлена на рис.4.1

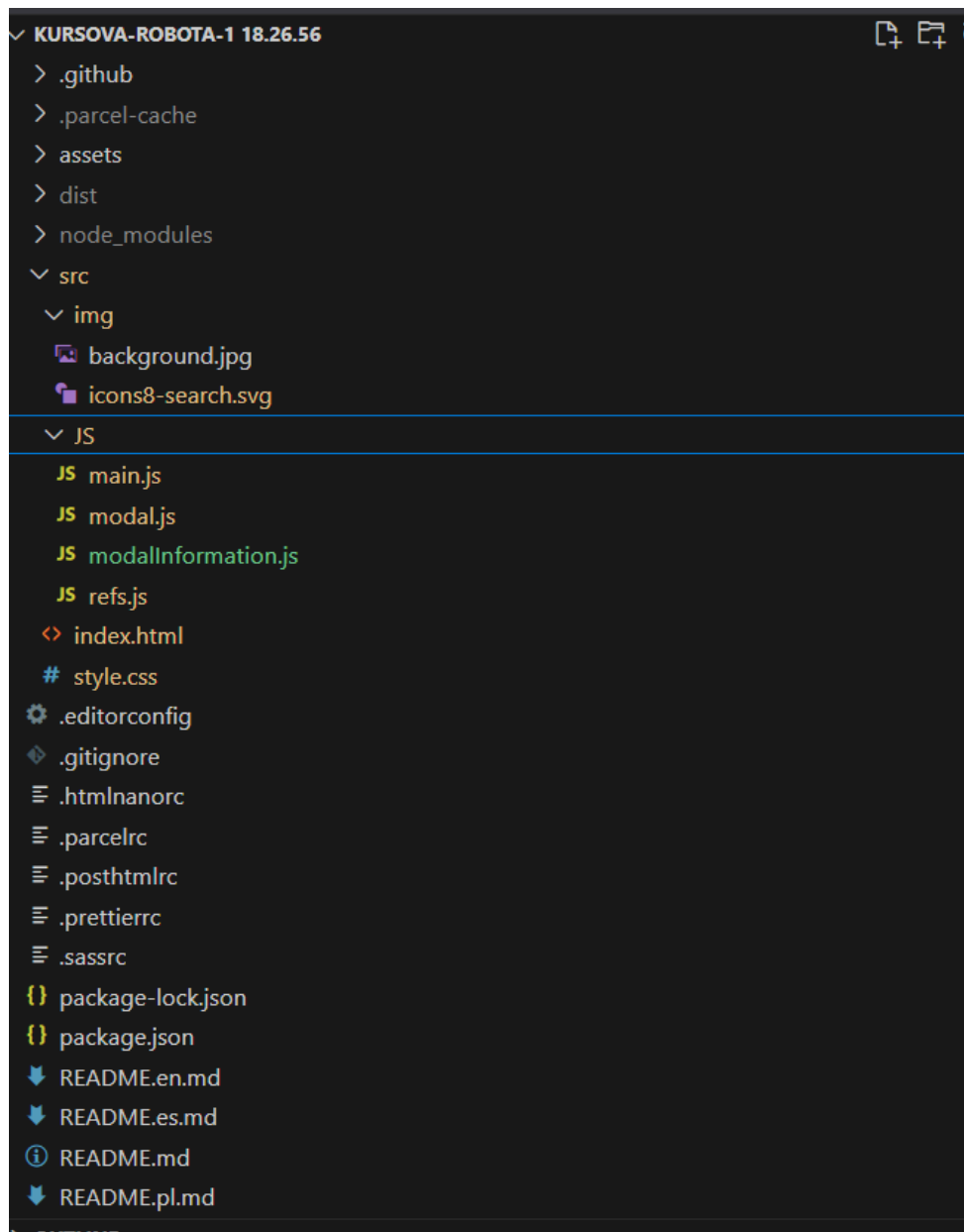


Рисунок 4.1 – Загальна структура проекту

Проект складається з веб-ресурсів, бібліотек, та вихідного коду, який в свою чергу можна поділити на компоненти рівня доступу до даних, компоненти бізнес-логіки та веб-компоненти.

4.2 Компоненти рівня доступу до даних

Основні сутності та інтерфейси рівня доступу до даних наведені на рис. 4.2

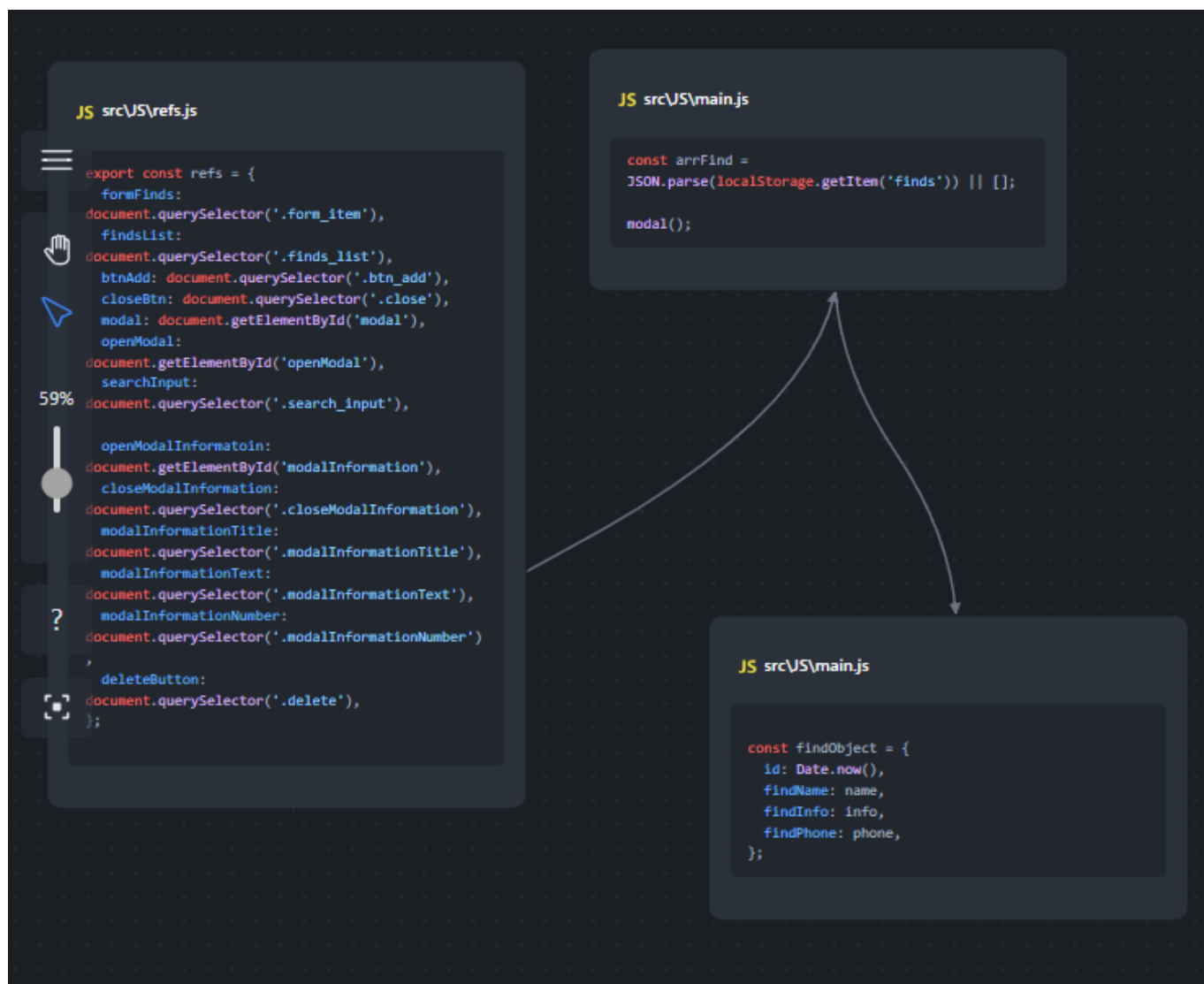


Рисунок 4.2 – Основні сутності та інтерфейси рівня доступу до даних

Сутності

1. Find (Знахідка)

id: Унікальний ідентифікатор (ціле число, генерується за допомогою `Date.now()`).

findName: Назва знахідки (рядок).

findInfo: Опис знахідки (рядок).

findPhone: Контактний номер телефону (рядок, з валідацією).

2. Refs (Посилання)

Містить посилання на різні елементи, такі як форми, кнопки, модальні вікна та інші компоненти, які використовуються для маніпуляцій та взаємодії з користувачем.

Зв'язки

1. Знахідка - Зберігання в локальному сховищі

Дані про знахідки зберігаються в локальному сховищі браузера у вигляді JSON-об'єкта. Це забезпечує постійний доступ до даних навіть після перезавантаження сторінки.

2. Знахідка - Візуалізація

Знахідки відображаються у вигляді карток на сторінці. Кожна картка містить інформацію про назву, опис та контактний номер знахідки.

3. Знахідка - Модальне вікно інформації

При натисканні на картку знахідки відкривається модальне вікно з детальною інформацією про знахідку. Це дозволяє користувачу переглядати деталі та видаляти знахідку.

Способи використання

1. Додавання знахідки

Користувач може додати нову знахідку, заповнивши форму та натиснувши кнопку "Опублікувати". Дані форми зберігаються в локальному сховищі, і нова знахідка відображається на сторінці.

2. Пошук знахідок

Користувач може шукати знахідки за назвою або описом, використовуючи пошукове поле. Введений запит фільтрує список відображених знахідок у реальному часі.

3. Перегляд інформації про знахідку

Натискання на картку знахідки відкриває модальне вікно з детальною інформацією про цю знахідку.

4. Видалення знахідки

У модальному вікні інформації про знахідку є кнопка "Видалити", яка дозволяє видалити знахідку з локального сховища та списку відображених знахідок.

4.3 Компоненти рівня бізнес-логіки

Основні сутності та інтерфейси рівня бізнес-логіки наведені на рис. 4.3

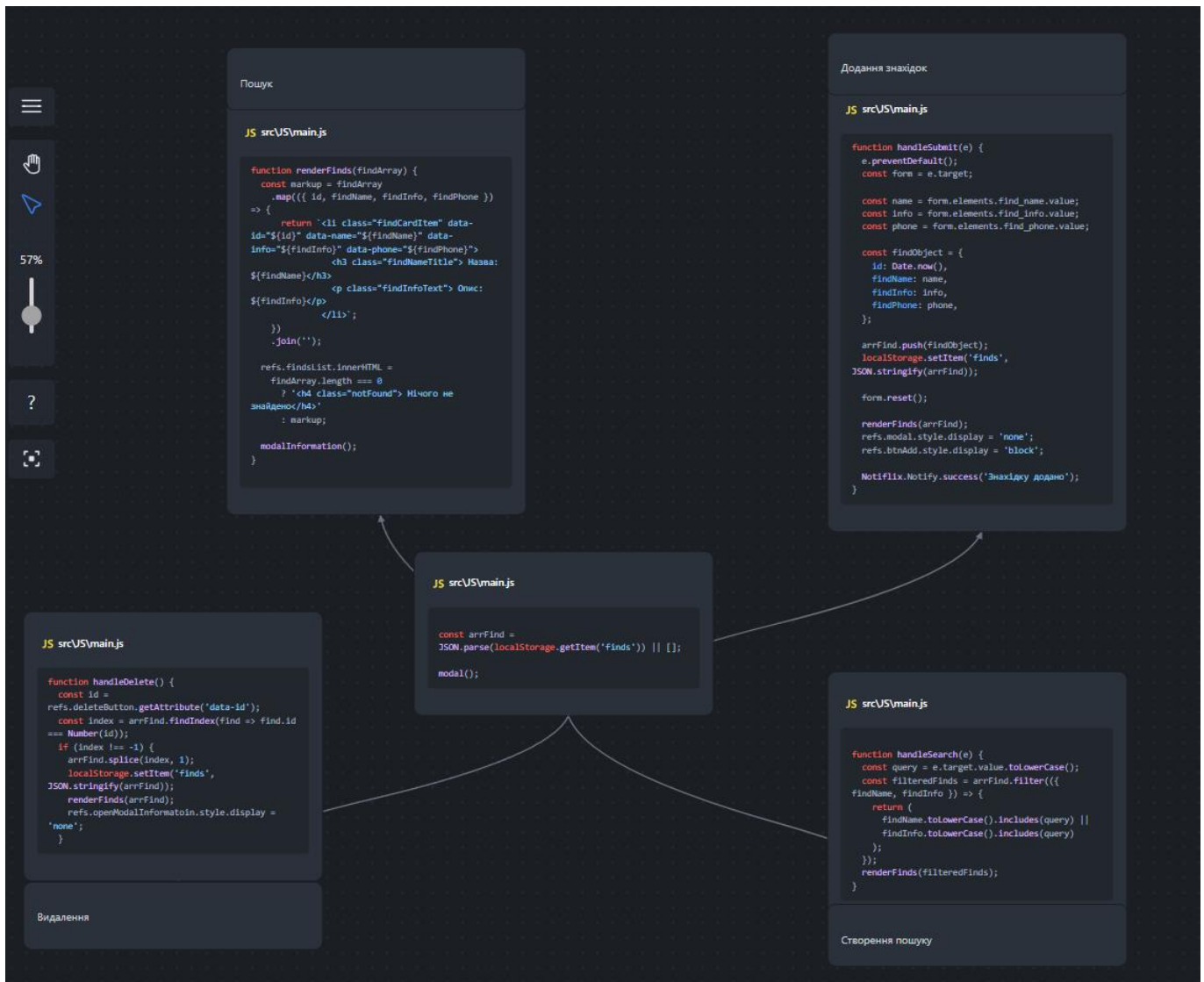


Рисунок 4.3 – Основні сутності та інтерфейси рівня бізнес-логіки

Сутності

1. FindService

Відповідає за управління даними знахідок, зокрема за створення, читання, оновлення та видалення знахідок у локальному сховищі.

2. ModalService

Відповідає за управління станом модальних вікон, зокрема за їх відкриття та закриття, а також за передачу необхідних даних між компонентами.

3. NotificationService

Відповідає за відображення повідомлень користувачу, наприклад, про успішне додавання або видалення знахідки.

Зв'язки

1. FindService взаємодіє з локальним сховищем для збереження та отримання даних про знахідки.
2. ModalService взаємодіє з компонентами UI для відображення та закриття модальних вікон.
3. NotificationService взаємодіє з компонентами UI для відображення повідомлень користувачу.

Способи використання

1. Додавання знахідки

FindService: Додає нову знахідку до локального сховища та повертає оновлений список знахідок.

NotificationService: Відображає повідомлення про успішне додавання знахідки.

2. Пошук знахідок

FindService: Фільтрує знахідки за заданим запитом та повертає відфільтрований список знахідок.

Відображення інформації про знахідку

FindService: Повертає дані про конкретну знахідку для відображення у модальному вікні.

3. Видалення знахідки

FindService: Видаляє знахідку з локального сховища та повертає оновлений список знахідок.

NotificationService: Відображає повідомлення про успішне видалення знахідки.

Сутності:

1. FormComponent

Відповідає за відображення та обробку форми додавання нової знахідки.

2. SearchComponent

Відповідає за пошук знахідок за назвою або описом.

3. FindListComponent

Відповідає за відображення списку знахідок та ініціацію модального вікна з інформацією про вибрану знахідку.

4. ModalComponent

Відповідає за відображення та управління модальними вікнами для додавання та перегляду інформації про знахідку.

5. NotificationComponent

Відповідає за відображення повідомлень користувачу.

Зв'язки

1. FormComponent передає дані до FindService для додавання нової знахідки.
2. SearchComponent використовує FindService для пошуку знахідок.
3. FindListComponent використовує ModalService для відкриття модального вікна з детальною інформацією про знахідку.
4. ModalComponent використовує FindService для видалення знахідки та NotificationService для відображення повідомлень.
5. NotificationComponent взаємодіє з NotificationService для відображення повідомлень.

Способи використання

1. Додавання знахідки

Користувач заповнює форму додавання знахідки, після чого викликається FindService для збереження даних та NotificationService для відображення повідомлення про успіх.

2. Пошук знахідок

Користувач вводить запит у пошукове поле, після чого викликається FindService для фільтрації знахідок та FindListComponent для відображення результатів.

3. Перегляд інформації про знахідку

Користувач натискає на картку знахідки, після чого викликається ModalService для відображення модального вікна з детальною інформацією.

4. Видалення знахідки

Користувач натискає кнопку видалення в модальному вікні, після чого викликається FindService для видалення знахідки та NotificationService для відображення повідомлення про успіх.

ВИСНОВКИ

У процесі створення курсової роботи було розроблено інформаційну систему "Бюро знахідок", що стала надійним помічником для тих, хто загубив або знайшов речі. Це система, яка з самого початку ставила амбітні цілі, успішно вирішила всі поставлені задачі, створивши зручний, інтуїтивний та ефективний сервіс.

Основною метою нашої роботи було створення платформи, яка допоможе людям швидко і легко знаходити загублені речі та зв'язуватися з тими, хто їх знайшов. Ця мета була досягнута завдяки реалізації наступних ключових можливостей:

- 1 Реєстрація знахідок: Інтуїтивно зрозумілий інтерфейс дозволяє користувачам легко додавати інформацію про знайдені речі.
- 2 Видалення знахідок: Користувачі можуть видаляти записи про знайдені речі, коли вони повертаються власникам або якщо інформація стає неактуальною.
- 3 Пошук за ключовими словами: Механізм пошуку дозволяє користувачам швидко знаходити необхідну інформацію за допомогою ключових слів.
4. Перегляд інформації про знахідки: Зручний інтерфейс для перегляду детальної інформації про знайдені речі, включаючи опис та контактні дані того, хто знайшов річ.

Результати роботи показали, що система "Бюро знахідок" є дієвим та надійним інструментом для вирішення проблеми втрати особистих речей. Вона зручна у використанні, інтуїтивно зрозуміла та функціональна. Реалізовані можливості дозволяють користувачам легко додавати, видаляти та шукати інформацію про загублені речі, що значно підвищує шанси на їх успішне повернення.

Система "Бюро знахідок" вже зараз є незамінним інструментом для вирішення проблеми втрати особистих речей. Вона не тільки допомагає людям знайти загублене, але й приносить радість від повернення цінних предметів. Подальший розвиток системи зробить її ще більш потужною та ефективною, що сприятиме зменшенню кількості загублених речей та допоможе людям зберігати спокій і довіру в нашому суспільстві.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. npm: notiflix: <https://www.npmjs.com/package/notiflix>
2. Node.js: <https://nodejs.org/en/>
3. Документація JavaScript: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
4. Документація CSS: <https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>

ДОДАТОК А

Лістинг програми

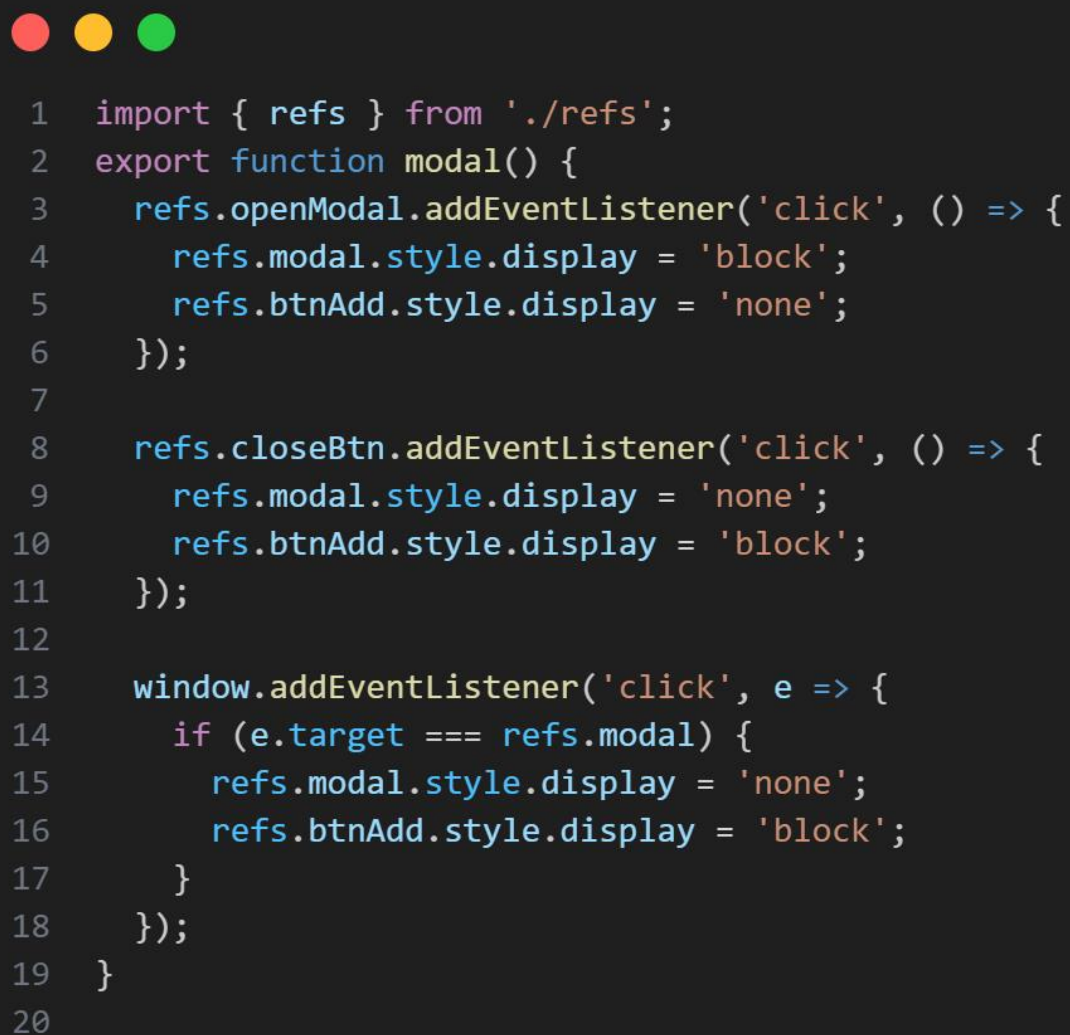
Посилання на github – <https://github.com/dajind/Kursova-robota-1>

```

1 import { refs } from './refs';
2 import { modal } from './modal';
3 import { modalInformation } from './modalInformation';
4
5 const arrFind = JSON.parse(localStorage.getItem('finds')) || [];
6
7 modal();
8
9 refs.formFinds.addEventListener('submit', handleSubmit);
10 refs.searchInput = document.querySelector('.search_input');
11 refs.searchInput.addEventListener('input', handleSearch);
12
13 function handleSubmit(e) {
14   e.preventDefault();
15   const form = e.target;
16
17   const name = form.elements.find_name.value;
18   const info = form.elements.find_info.value;
19   const phone = form.elements.find_phone.value;
20
21   const findObject = {
22     id: Date.now(),
23     findName: name,
24     findInfo: info,
25     findPhone: phone,
26   };
27
28   arrFind.push(findObject);
29   localStorage.setItem('finds', JSON.stringify(arrFind));
30
31   form.reset();
32
33   renderFinds(arrFind);
34   refs.modal.style.display = 'none';
35   refs.btnAdd.style.display = 'block';
36
37   Notiflix.Notify.success("Знахідку додано");
38 }
39
40 function handleSearch(e) {
41   const query = e.target.value.toLowerCase();
42   const filteredFinds = arrFind.filter(({ findName, findInfo }) => {
43     return (
44       findName.toLowerCase().includes(query) ||
45       findInfo.toLowerCase().includes(query)
46     );
47   });
48   renderFinds(filteredFinds);
49 }
50
51 function renderFinds(findArray) {
52   const markup = findArray
53     .map(({ id, findName, findInfo, findPhone }) => {
54       return `<li class="findCardItem" data-id="${id}" data-name="${findName}" data-info="${findInfo}" data-phone="${findPhone}">
55         <h3 class="findNameTitle"> Назва: ${findName}</h3>
56         <p class="findInfoText"> Опис: ${findInfo}</p>
57       </li>`;
58     })
59     .join('');
60
61   refs.findsList.innerHTML =
62     findArray.length === 0
63       ? `<h4 class="notFound"> Нічого не знайдено</h4>`
64       : markup;
65
66   modalInformation();
67 }
68
69 document.addEventListener('DOMContentLoaded', () => {
70   renderFinds(arrFind);
71 });
72
73 export function handleDelete() {
74   const id = refs.deleteButton.getAttribute('data-id');
75   const index = arrFind.findIndex(find => find.id === Number(id));
76   if (index !== -1) {
77     arrFind.splice(index, 1);
78     localStorage.setItem('finds', JSON.stringify(arrFind));
79     renderFinds(arrFind);
80     refs.openModalInformation.style.display = 'none';
81   }
82 }
83

```

Рисунок 1 – Файл main.js



```
1  import { refs } from './refs';
2  export function modal() {
3    refs.openModal.addEventListener('click', () => {
4      refs.modal.style.display = 'block';
5      refs.btnAdd.style.display = 'none';
6    });
7
8    refs.closeBtn.addEventListener('click', () => {
9      refs.modal.style.display = 'none';
10     refs.btnAdd.style.display = 'block';
11   });
12
13   window.addEventListener('click', e => {
14     if (e.target === refs.modal) {
15       refs.modal.style.display = 'none';
16       refs.btnAdd.style.display = 'block';
17     }
18   });
19 }
20
```

Рисунок 2 – Файл modal.js

```

1  import { refs } from './refs';
2  import { handleDelete } from './main';
3
4  export function modalInformation() {
5      const cards = document.querySelectorAll('.findCardItem');
6
7      cards.forEach(card => {
8          card.addEventListener('click', () => {
9              const id = card.dataset.id;
10             const name = card.dataset.name;
11             const info = card.dataset.info;
12             const phone = card.dataset.phone;
13
14             refs.modalInformationTitle.textContent = `Назва: ${name}`;
15             refs.modalInformationText.textContent = `Опис: ${info}`;
16             refs.modalInformationNumber.textContent = `Номер: ${phone}`;
17
18             if (refs.deleteButton) {
19                 refs.deleteButton.setAttribute('data-id', id);
20             } else {
21                 console.error('Delete button not found in DOM');
22             }
23
24             refs.openModalInformatoin.style.display = 'block';
25         });
26     });
27
28     window.addEventListener('click', e => {
29         if (e.target === refs.openModalInformatoin) {
30             refs.openModalInformatoin.style.display = 'none';
31         }
32     });
33
34     refs.closeModalInformation.addEventListener('click', () => {
35         refs.openModalInformatoin.style.display = 'none';
36     });
37
38     if (refs.deleteButton) {
39         refs.deleteButton.addEventListener('click', handleDelete);
40     } else {
41         console.error('Delete button not found in DOM');
42     }
43 }
44

```

Рисунок 3 – Файл modalInformation.js



```
1  export const refs = {  
2    formFinds: document.querySelector('.form_item'),  
3    findsList: document.querySelector('.finds_list'),  
4    btnAdd: document.querySelector('.btn_add'),  
5    closeBtn: document.querySelector('.close'),  
6    modal: document.getElementById('modal'),  
7    openModal: document.getElementById('openModal'),  
8    searchInput: document.querySelector('.search_input'),  
9  
10   openModalInformatoin: document.getElementById('modalInformation'),  
11   closeModalInformation: document.querySelector('.closeModalInformation'),  
12   modalInformationTitle: document.querySelector('.modalInformationTitle'),  
13   modalInformationText: document.querySelector('.modalInformationText'),  
14   modalInformationNumber: document.querySelector('.modalInformationNumber'),  
15   deleteButton: document.querySelector('.delete'),  
16 };  
17
```

Рисунок 4 – Файл refs.js

```

1 <!DOCTYPE html>
2 <html lang="uk">
3 <head>
4 <meta charset="UTF-8" />
5 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
6 <title>Бюро знахідок</title>
7 <link rel="stylesheet" href="./style.css" />
8 </head>
9 <body>
10 <section class="home">
11 <div class="container">
12 <br />
13 <div class="menu">
14 <h1>Вас вітає бюро знахідок!</h1>
15 <ul class="form_list">
16 <li>
17 <div class="search_find">
18 <div class="search_box">
19 <input
20 autocomplete="off"
21 class="search_input"
22 type="text"
23 placeholder="Пошук"
24 />
25 <button class="btn_search">
26 <svg width="18" height="18">
27 <use href="./img/icons8-search.svg#icon-search"></use>
28 </svg>
29 </button>
30 </div>
31 <ul></ul>
32 </div>
33 </li>
34 <li>
35 <button id="openModal" class="btn_add">Додати знахідку</button>
36 </li>
37 </ul>
38 </div>
39 </div>
40 </section>
41
42 <section class="render_finds">
43 <ul class="finds_list"></ul>
44 </section>
45
46 <div id="modal" class="modal">
47 <div class="modal-form-content">
48 <span class="close">&times;</span>
49 <form class="form_item" type="submit">
50 <input
51 autocomplete="off"
52 id="input_name"
53 type="text"
54 placeholder="Назва"
55 name="find_name"
56 required
57 />
58 <textarea
59 id="input_info"
60 placeholder="Опис"
61 name="find_info"
62 required
63 ></textarea>
64 <input
65 autocomplete="off"
66 id="input_phone"
67 type="tel"
68 placeholder="Контакти"
69 name="find_phone"
70 pattern="[0-9]{10,15}"
71 title="Введіть дійсний номер телефону (від 10 до 15 цифр)"
72 required
73 />
74 <button class="btn">Опублікувати</button>
75 </form>
76 </div>
77 </div>
78
79 <div id="modalInformation" class="modalInformation">
80 <div class="modal-content">
81 <span class="closeModalInformation">&times;</span>
82 <h4 class="modalInformationTitle"></h4>
83 <p class="modalInformationText"></p>
84 <p class="modalInformationNumber"></p>
85 <button class="delete" data-id="">Видалити</button>
86 </div>
87 </div>
88
89 <script src="./JS/main.js" type="module"></script>
90 </body>
91 </html>
92

```

Рисунок 5 – Файл index.html