Міністерство освіти і науки України
Криворізький національний університет
Кафедра моделювання і програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавра

зі спеціальності 121 − Інженерія програмного забезпечення

На тему: Веб-застосунок для управління заходами екологічного спрямування

Засвідчую, що в цій кваліфікаційній роботі немає запозичень із праць інших авторів без відповідних посилань.

Студент групи ІПЗ-22-1

_____ / А. А. Безщасна /

Керівник кваліфікаційної роботи _____ / О. Г. Рибальченко /

Завідувач кафедри _____ / А. М. Стрюк /

Кривий Ріг
2026

Криворізький національний університет

Факультет Інформаційних технологій

Кафедра Моделювання та програмного забезпечення

Ступінь вищої освіти бакалавр

Спеціальність 121 − Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Зав. кафедрою
к.п.н., доц. А.М. Стрюк
“____”_____20___р.

ЗАВДАННЯ

на кваліфікаційну роботу

студентці групи ІПЗ-22-1 Безщасній Анастасії Андріївни

1. Тема: Розробка веб-застосунку для управління заходами екологічного спрямування наказом по КНУ №284-с від «28» травня 2026 р.
2. Термін подання студентом закінченої роботи: «10» червня 2026 р.
3. Вихідні дані по роботі: Результати аналізу існуючих екологічних платформ (Eco-Cycle A-Z Recycling Guide, Freegle, RecycleBank, Україна без сміття, Ecola, Let's Do It Ukraine) технічна документація використаних технологій (Node.js, TypeScript, React, MySQL).
4. Зміст пояснювальної записки (перелік питань, що їх треба розробити): Вступ. Розділ 1 – Аналіз сучасного стану екологічних платформ та обґрунтування актуальності роботи. Розділ 2 – Проєктування вебзастосунку. Розділ 3 – Реалізація та тестування вебзастосунку. Висновки. Список використаних джерел.
5. Перелік ілюстративного матеріалу: структурна схема, діаграма варіантів використання (прецедентів), контекстна діаграма, діаграма послідовностей, схема бази даних, знімки екрану розробленого веб-застосунку.

Календарний план:

№	Найменування етапів кваліфікаційної роботи	Термін виконання етапів роботи
1	Огляд літературних джерел відповідно до теми кваліфікаційної роботи	31.03.2026 – 04.04.2026
2	Пошук та проведення аналізу існуючих у предметній області аналогів програмних продуктів	05.04.2026 – 10.04.2026
3	Формулювання актуальності та мети і визначення завдань роботи	11.04.2026 – 13.04.2026
4	Підготовка матеріалів першого розділу роботи	14.04.2026 – 16.04.2026
5	Розробка математичного апарату, діаграм архітектури програмного продукту, проєктування бази даних	17.04.2026 – 19.04.2026
6	Створення вайрфреймів UI-компонентів вебзастосунку	20.04.2026 – 21.04.2026
7	Підготовка матеріалів другого розділу роботи	22.04.2026 – 25.04.2026
8	Створення і наповнення бази даних, розробка back-end частини вебзастосунку	26.04.2026 – 05.05.2026
9	Вибір технологій реалізації проєкту, розробка front-end частини вебзастосунку	06.05.2026 – 17.05.2026
10	Тестування розробленого програмного продукту, усунення виявлених дефектів	18.05.2026 – 23.05.2026
11	Підготовка матеріалів третього розділу роботи	23.05.2026 – 26.05.2026
12	Оформлення пояснювальної записки	27.05.2026 – 02.06.2026

Дата видачі завдання: «30» березня 2026р.

Студент: _____ / А. А. Безщасна /

Керівник роботи: _____ / О. Г. Рибальченко /

РЕФЕРАТ

ВЕБЗАСТОСУНОК, ЕКОЛОГІЯ, ПЕРЕРОБКА ВІДХОДІВ, ДРУГЕ ЖИТТЯ РЕЧЕЙ, МАЙСТЕР-КЛАСИ, ПОДІЇ, КОРИСТУВАЧІ, МОДЕРАЦІЯ, REACT, NODE.JS, БАЗА ДАНИХ

Пояснювальна записка: 75 с., 2 табл., 10 дод., 51 рис., 22 джерел.

Метою роботи є створення вебзастосунку екологічного спрямування, який надає користувачам інформацію про можливості переробки різних видів матеріалів, способи їх повторного використання, а також забезпечує доступ до екологічних подій та заходів. Розроблена система спрямована на підвищення рівня екологічної свідомості населення та популяризацію ідей сталого розвитку.

Об’єкт проєктування – вебзастосунок, що забезпечує зручний, інтуїтивно зрозумілий та адаптивний інтерфейс для взаємодії користувачів з інформацією про переробку відходів, екологічні ініціативи та освітній контент у вигляді статей і майстер-класів.

У даній кваліфікаційній роботі проведено аналіз сучасних екологічних вебресурсів і платформ, що підтримують концепцію повторного використання матеріалів та сортування відходів. Визначено їх основні переваги та недоліки, на основі чого сформовано вимоги до розроблюваної системи.

Розроблений вебзастосунок включає декілька функціональних модулів:

- модуль довідника переробки, що дозволяє користувачам перевіряти, чи підлягає певний продукт переробці та отримувати інформацію про місця його здачі;
- модуль статей і майстер-класів, де зареєстровані користувачі можуть публікувати матеріали щодо надання предметам другого життя;
- модуль подій, який відображає актуальні екологічні заходи та дозволяє користувачам реєструватися для участі в них.

У системі передбачено розмежування прав доступу: незареєстровані користувачі можуть переглядати інформацію про переробку, статті та події;

зареєстровані користувачі отримують можливість створювати власні статті, залишати коментарі, зберігати матеріали та реєструватися на заходи; модератори здійснюють перевірку та керування контентом; адміністратори відповідають за управління користувачами, ролями та подіями.

Реалізація програмного продукту охоплює розробку серверної логіки для керування контентом і автентифікації користувачів, а також створення клієнтського інтерфейсу. Останній забезпечує безперешкодний доступ до інформаційних ресурсів системи, включаючи модулі пошуку продуктів для переробки та інтерактивні інструменти взаємодії із спільнотою в реальному часі.

Розроблений вебзастосунок може бути використаний як інформаційна платформа для популяризації екологічного способу життя, підвищення обізнаності щодо переробки відходів та залучення користувачів до екологічних ініціатив.

ABSTRACT

WEB APPLICATION, ECOLOGY, WASTE RECYCLING, SECOND LIFE OF ITEMS, WORKSHOPS, EVENTS, USERS, MODERATION, REACT, NODE.JS, DATABASE

Explanatory note: 75 p., 2 tab., 10 app., 51 fig., 22 sources.

The aim of this work is to develop an environmentally oriented web application that provides users with information about the recyclability of various products, ways of their reuse, as well as access to environmental events and activities. The developed system is aimed at increasing environmental awareness and promoting the principles of sustainable development.

The object of the study is a web application that provides a convenient, intuitive, and adaptive interface for user interaction with information about waste recycling, environmental initiatives, and educational content in the form of articles and workshops.

In this qualification work, an analysis of modern environmental web resources and platforms supporting the concept of reuse and waste sorting has been carried out. Their main advantages and disadvantages have been identified, and based on this analysis, the requirements for the developed system have been formed.

The developed web application includes several functional modules:

- a recycling reference module that allows users to check whether a product can be recycled and obtain information about where it can be delivered;
- an articles and workshops module where registered users can publish materials on giving items a second life;
- an events module that displays current environmental activities and allows users to register for participation.

The system provides role-based access control: unregistered users can view recycling information, articles, and events; registered users can create their own articles, leave comments, save materials, and register for events; moderators are

responsible for content verification and management; administrators manage users, assign roles, and create, edit, or delete events.

The implementation of the software product includes the development of server logic for content management and user authentication, as well as the creation of a client interface. The latter provides seamless access to the system's information resources, including modules for searching for products for recycling and interactive tools for interacting with the community in real time.

The developed web application can be used as an informational platform to promote an environmentally friendly lifestyle, increase awareness of waste recycling, and engage users in environmental initiatives.

ЗМІСТ

ВСТУП

У сучасному світі проблема забруднення навколишнього середовища та нераціонального використання природних ресурсів набуває все більшої актуальності. Зростання обсягів відходів, недостатній рівень їх переробки та низька екологічна обізнаність населення призводять до негативних наслідків для довкілля та якості життя людей. У зв'язку з цим особливо важливим є впровадження інформаційних технологій, які сприяють популяризації екологічного способу життя та формуванню відповідального ставлення до природних ресурсів.

Одним із перспективних напрямів вирішення цієї проблеми є створення спеціалізованих вебзастосунків, що надають користувачам доступ до інформації про можливості переробки відходів, способи повторного використання речей, а також про екологічні заходи та ініціативи. Такі системи дозволяють не лише отримувати необхідну інформацію, але й активно взаємодіяти з екологічною спільнотою, обмінюватися досвідом та долучатися до корисних заходів.

Актуальність теми зумовлена необхідністю підвищення рівня екологічної свідомості населення та створенням зручних інструментів для доступу до інформації про переробку відходів і повторне використання матеріалів. Використання сучасних вебтехнологій дозволяє реалізувати інтерактивні платформи, які поєднують інформаційні, освітні та соціальні функції.

Метою даної роботи є розробка вебзастосунку для управління заходами екологічного спрямування, який забезпечує користувачів інформацією про переробку продуктів, можливості їх здавання, а також надає платформу для публікації матеріалів щодо надання речам другого життя та організації екологічних заходів.

Об'єктом дослідження є процеси інформування користувачів про переробку відходів та організацію екологічної взаємодії в інформаційних системах.

Предметом дослідження є методи та засоби розробки вебзастосунків для підтримки екологічних ініціатив та управління відповідним контентом.

Практичне значення отриманих результатів полягає у створенні функціонального вебзастосунку, який може бути використаний як інформаційна та комунікаційна платформа для популяризації екологічного способу життя, підвищення обізнаності щодо переробки відходів та залучення користувачів до екологічних заходів.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОБҐРУНТУВАННЯ АКТУАЛЬНОСТІ РОБОТИ

1.1. Актуальний стан професійної області

У сучасних умовах стрімкого розвитку інформаційних технологій та зростання екологічних проблем питання ефективного управління відходами та популяризації екологічного способу життя набувають особливої актуальності. Значне збільшення обсягів побутових і промислових відходів, а також недостатній рівень їх переробки створюють серйозне навантаження на навколишнє середовище. У зв’язку з цим виникає потреба у впровадженні інноваційних підходів до інформування населення та організації екологічної діяльності.

Сучасна професійна область, що поєднує інформаційні технології та екологію, активно розвивається у напрямку створення цифрових платформ і вебзастосунків, спрямованих на підвищення екологічної обізнаності користувачів. Такі системи забезпечують доступ до інформації про сортування та переробку відходів, дозволяють знаходити пункти прийому вторинної сировини, а також сприяють популяризації повторного використання матеріалів.

На сьогодні існує значна кількість вебресурсів та мобільних додатків екологічного спрямування. Вони виконують різні функції: від довідників із сортування сміття до платформ для організації екологічних заходів. Однак більшість із них мають обмежений функціонал, зосереджуючись лише на одному аспекті, наприклад, інформуванні про переробку або публікації екологічних новин.

Важливим напрямом розвитку є інтеграція освітнього контенту, зокрема матеріалів щодо надання речам другого життя. Такий підхід дозволяє не лише інформувати користувачів, але й формувати практичні навички екологічної поведінки. Водночас не всі існуючі рішення надають

можливість користувачам самостійно створювати та поширювати подібний контент, що обмежує розвиток спільноти та обмін досвідом.

Окрему увагу в сучасних системах приділяють соціальній взаємодії користувачів. Реалізація функцій коментування, збереження матеріалів та участі в подіях сприяє формуванню активної екологічної спільноти. Проте питання ефективного управління контентом і контролю його якості залишаються актуальними, що обумовлює необхідність впровадження механізмів модерації та розмежування прав доступу.

Сучасні вебтехнології, такі як клієнт-серверна архітектура, використання REST API, а також фреймворки для розробки інтерфейсів і серверної логіки, дозволяють створювати масштабовані, зручні та функціональні системи. Це відкриває широкі можливості для реалізації комплексних вебзастосунків, які об’єднують різні аспекти екологічної діяльності в єдиній платформі.

Таким чином, аналіз актуального стану професійної області показує, що існує потреба у створенні інтегрованих вебзастосунків, які поєднують інформаційні, освітні та соціальні функції. Розробка такої системи дозволить підвищити ефективність інформування користувачів, сприятиме розвитку екологічної культури та забезпечить зручні інструменти для взаємодії в межах екологічних ініціатив.

1.2. Аналіз існуючих аналогів

У сучасному цифровому середовищі існує значна кількість вебресурсів та мобільних застосунків, спрямованих на підтримку екологічного способу життя, сортування відходів та повторне використання матеріалів. Проведений аналіз дозволяє виділити основні типи таких систем та оцінити їх функціональні можливості.

Одним із поширених типів є довідкові системи з переробки відходів. Прикладом є платформа «Eco-Cycle A-Z Recycling Guide» (Рисунок 1.1), яка дозволяє користувачам здійснювати пошук інформації про можливість

переробки різних предметів та отримувати рекомендації щодо їх утилізації. Основною перевагою таких систем є простота використання та швидкий доступ до інформації. Однак їх функціональність обмежується лише довідковими можливостями без елементів соціальної взаємодії чи створення контенту користувачами [1].

Рисунок 1.1 – Інтерфейс «Eco-Cycle A-Z Recycling Guide»

Окрему групу становлять платформи, спрямовані на повторне використання речей. Наприклад, «Freegle» (Рисунок 1.2) є онлайн-мережею, що дозволяє користувачам безкоштовно обмінюватися речами, зменшуючи кількість відходів. Такі системи сприяють розвитку концепції циркулярної економіки, однак не містять освітнього контенту або структурованої інформації про переробку матеріалів [2].

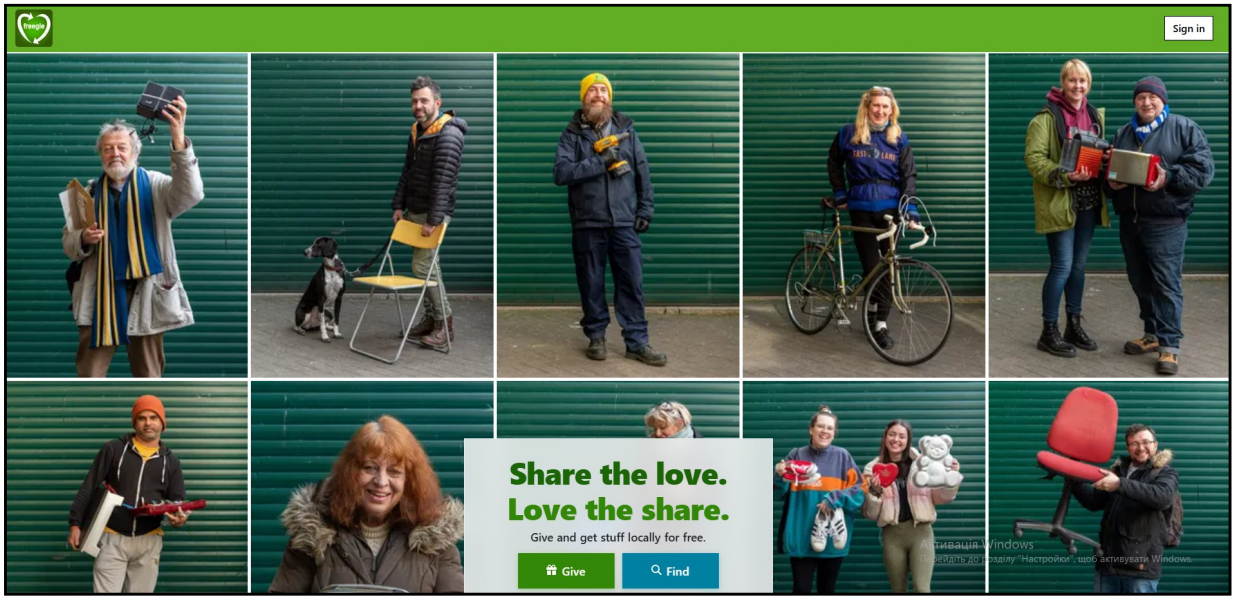

Рисунок 1.2 – «Інтерфейс Freegle»

Існують також платформи з елементами економічної вигоди, такі як «Compare and Recycle» (Рисунок 1.3), які дозволяють користувачам порівнювати пропозиції щодо здачі електроніки та отримувати за це винагороду. Перевагою є стимулювання користувачів до здачі відходів, однак функціонал обмежується лише певними типами продукції та не передбачає освітньої чи соціальної складової [3].

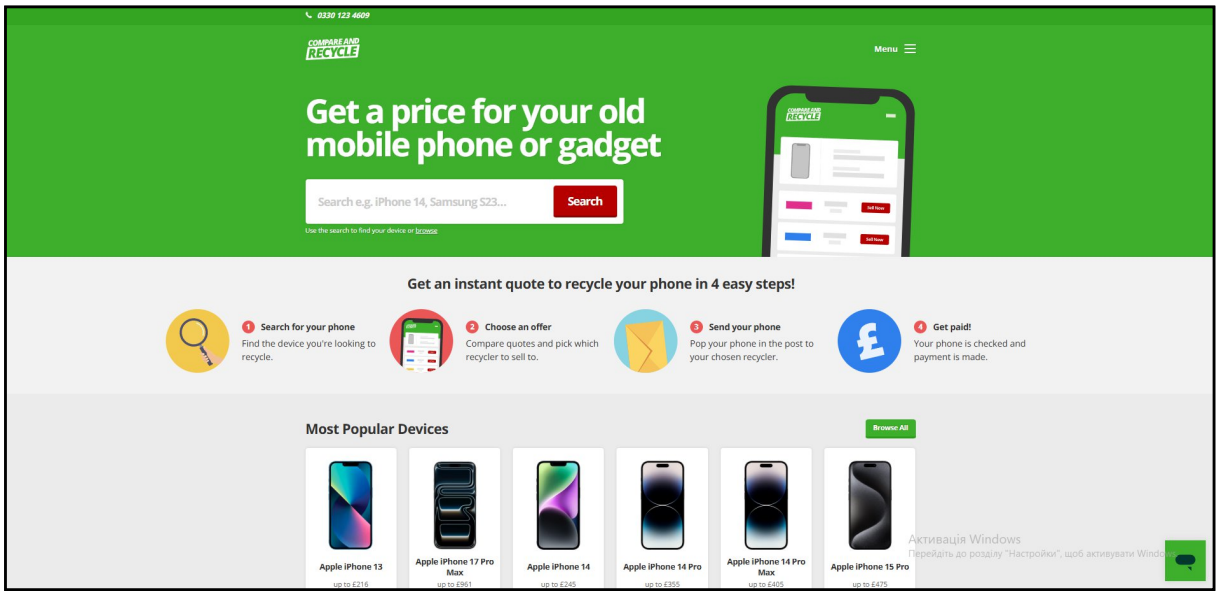

Рисунок 1.3 – Інтерфейс «Compare and Recycle»

Серед сучасних тенденцій варто відзначити використання інноваційних технологій у сфері управління відходами. Наприклад, компанія

«Recycle Track Systems» (Рисунок 1.4) використовує цифрові платформи, штучний інтелект та системи відстеження для контролю процесів переробки. Такі рішення є ефективними для бізнесу та муніципальних структур, проте вони є складними та недоступними для широкого кола користувачів [4].

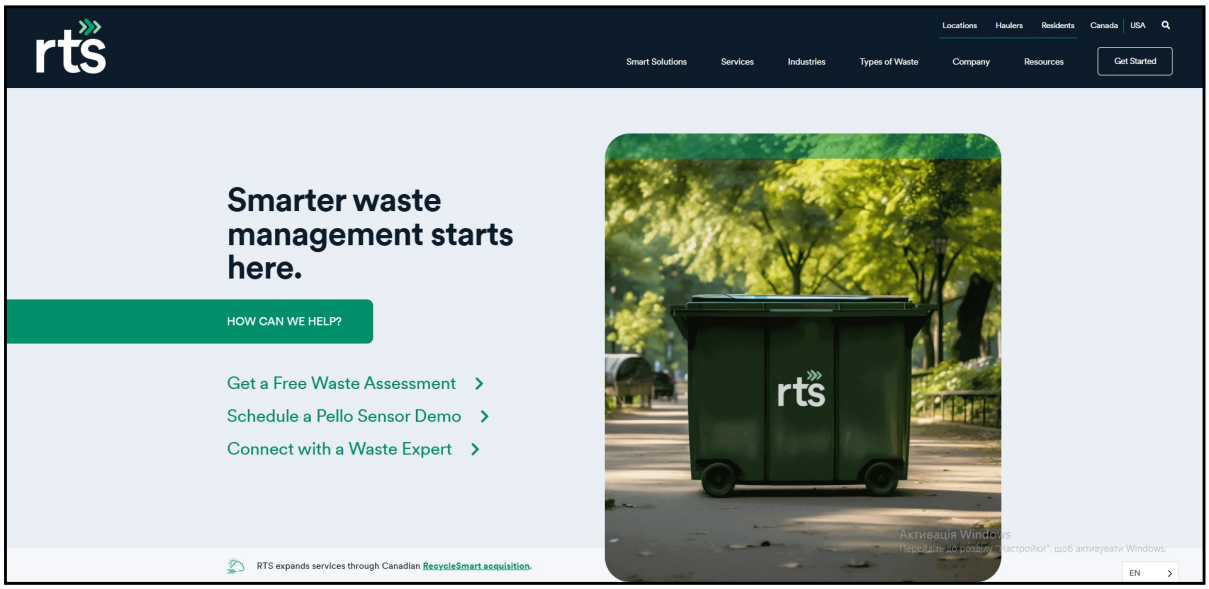

Рисунок 1.4 – «Інтерфейс Recycle Track Systems»

Крім того, існують застосунки, які поєднують інформаційні та освітні функції, наприклад «Horizon» (Рисунок 1.5) або «RecycleBank» (Рисунок 1.6), що дозволяють користувачам дізнаватися про можливості переробки та навіть отримувати бонуси за екологічну поведінку. Проте такі системи часто мають обмеження за регіоном або не забезпечують повноцінної взаємодії користувачів між собою [5,6].

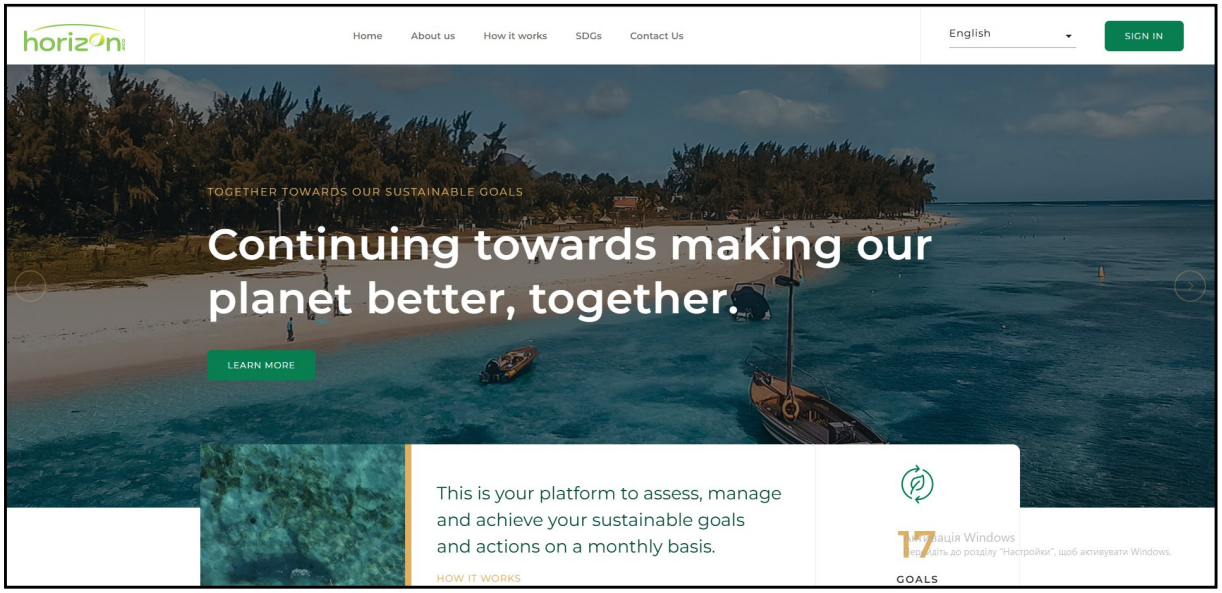

Рисунок 1.5 – Інтерфейс «Horizon»

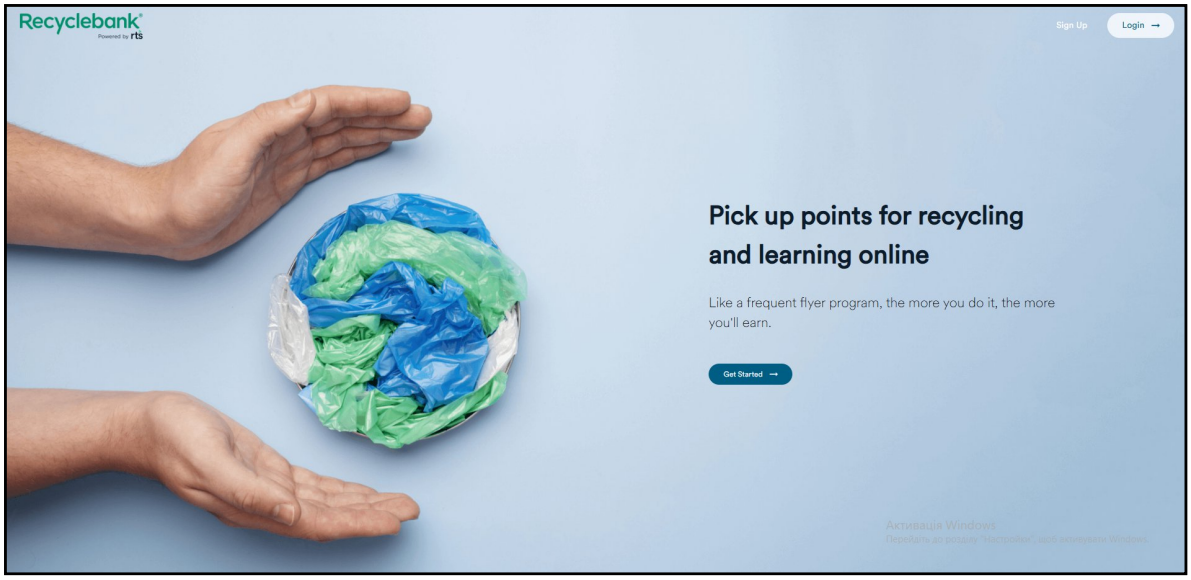

Рисунок 1.6 – Інтерфейс «RecycleBank»

Аналіз відгуків користувачів у мережі Reddit показує, що однією з основних проблем існуючих рішень є їх локальність та фрагментованість. [7]

Це свідчить про потребу в універсальній платформі, яка поєднує різні функції в одному середовищі.

Таблиця 1.1 – Порівняльний аналіз іноземних аналогів

Критерій		Назва вебзастосунку			
		Eco-Cycle A–Z Recycling Guide	Freegle	RecycleBank	Розроблюваний вебзастосунок «EcoWay»
Функціональні можливості	Перевірка переробки продуктів	Так	Ні	Так	Так
	Інформація про пункти здачі	Так	Ні	Так (обмежено)	Так
	Підтримка різних типів відходів	Так	Так	Так	Так
Контент	Освітні статті / майстер-класи	Ні	Ні	Так	Так
	Можливість створення контенту користувачами	Ні	Так (оголошення)	Ні	Так
Соц. взаємодія	Система коментарів	Ні	Ні	Ні	Так
	Збереження матеріалів	Ні	Ні	Так	Так
Події	Події та заходи	Ні	Ні	Ні	Так
	Реєстрація на події	Ні	Ні	Ні	Так

Проаналізувавши не тільки іноземні аналоги, а й українські можна дійти висновку, що одним із найбільш відомих українських екологічних сервісів є «Україна без сміття» (Рисунок 1.7). Проєкт надає інформацію щодо сортування відходів, правил переробки різних матеріалів та пунктів прийому вторинної сировини. Основною перевагою сервісу є наявність великої кількості довідкової інформації та популяризація культури сортування сміття. Водночас ресурс не підтримує функціональність створення користувацького контенту, систему ролей та модуль організації екологічних заходів [8].

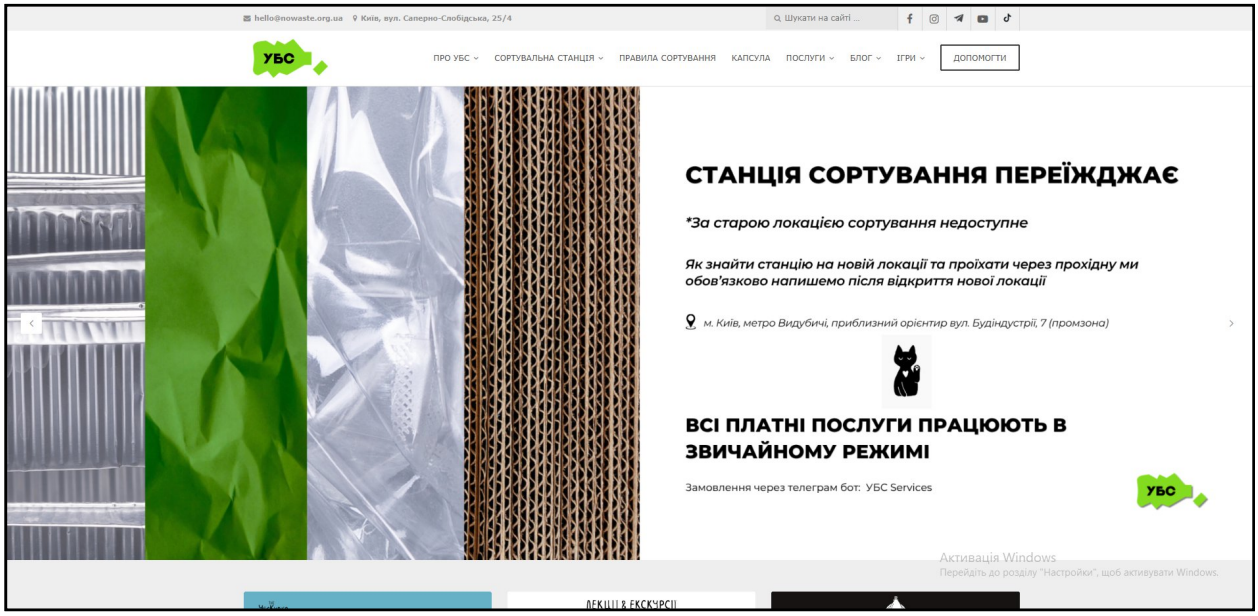

Рисунок 1.7 – Інтерфейс «Україна без сміття»

Також було розглянуто сервіс «Ecola» (Рисунок 1.8), який надає інформацію про пункти прийому вторинної сировини та екологічні ініціативи. Основною перевагою є зручний пошук місць здачі відходів. Недоліком є обмежений функціонал щодо створення освітнього контенту та відсутність системи модерації користувацьких матеріалів [9].

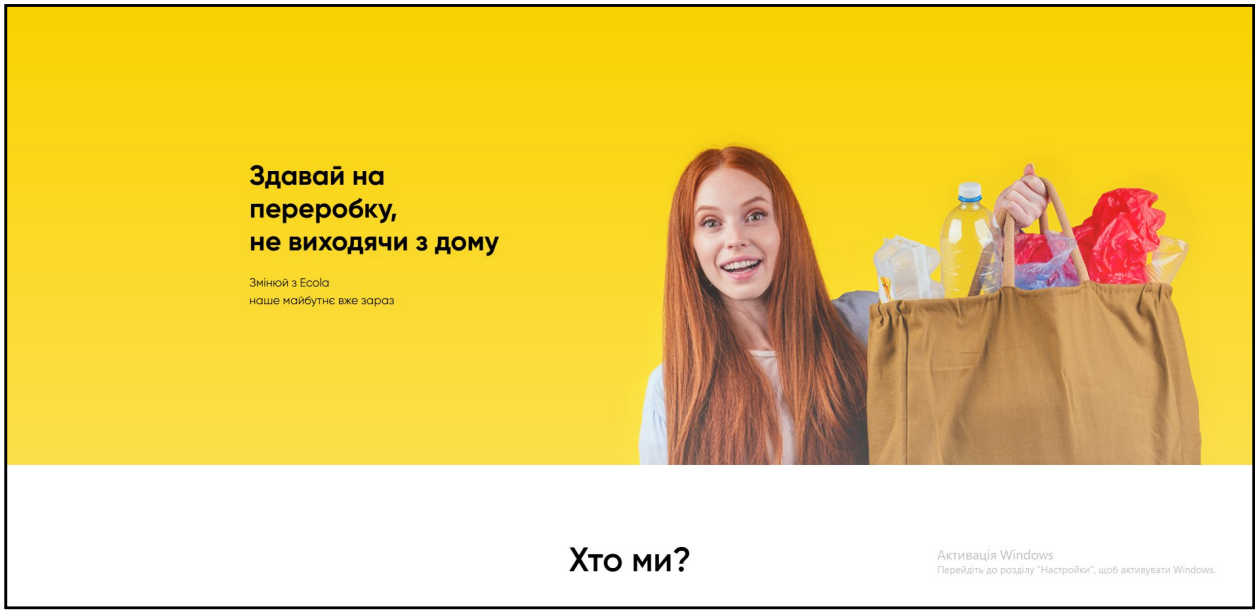

Рисунок 1.8 – Інтерфейс «Ecola»

Окрему увагу заслуговує платформа «Let's Do It Ukraine» (Рисунок 1.9), яка спеціалізується на організації екологічних заходів та волонтерських ініціатив. Сервіс дозволяє користувачам переглядати інформацію про

екологічні акції та брати участь у них. Однак система не містить довідника переробки та можливості створення майстер-класів щодо повторного використання речей [10].

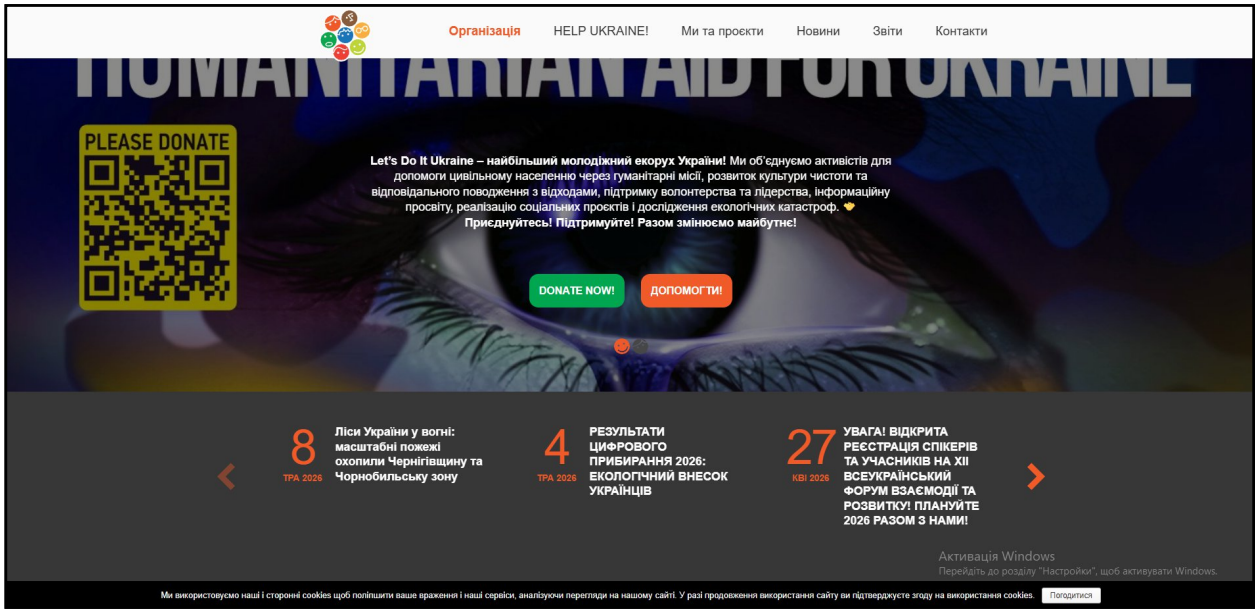

Рисунок 1.9 – Інтерфейс «Let's Do It Ukraine»

Таблиця 1.2 – Порівняльний аналіз українських аналогів

Критерій		Назва вебзастосунку			
		Україна без сміття	Ecola	Let's Do It Ukraine	Розроблюваний вебзастосунок «EcoWay»
1		2	3	4	5
Функціональні можливості	Перевірка переробки продуктів	Так	Так	Ні	Так
	Інформація про пункти здачі	Так	Так	Ні	Так
	Підтримка різних типів відходів	Так	Ні	Ні	Так
Контент	Освітні статті / майстер-класи	Так	Ні	Так	Так
	Можливість створення контенту користувачами	Ні	Ні	Ні	Так

Продовження таблиці 1.2

1		2	3	4	5
Соц. взаємодія	Система коментарів	Ні	Ні	Ні	Так
	Збереження матеріалів	Ні	Ні	Ні	Так
Події	Події та заходи	Так	Ні	Так	Так
	Реєстрація на події	Ні	Ні	Так	Так

Таким чином, аналіз існуючих аналогів дозволяє зробити такі висновки:

- більшість систем реалізують лише одну функцію (довідник, обмін речами або пошук пунктів прийому);
- відсутня інтеграція освітнього контенту та можливість створення матеріалів користувачами;
- обмежена соціальна взаємодія між користувачами;
- значна частина сервісів має регіональні обмеження;
- недостатньо реалізовані механізми модерації та управління контентом.

На відміну від аналогів, розроблюваний вебзастосунок поєднує всі ключові функції в єдиній системі: довідник переробки, платформу для створення освітнього контенту, систему подій, а також механізми взаємодії користувачів і модерації. Це забезпечує більш високий рівень зручності, функціональності та ефективності використання.

1.3. Формулювання актуальності та мети роботи

У сучасному світі проблема раціонального поводження з відходами та переходу до економіки замкненого циклу набуває критичного значення. Зростання обсягів побутового сміття та низький рівень обізнаності населення щодо правил сортування призводять до погіршення екологічної ситуації. Попри наявність загальної інформації в мережі, користувачі часто стикаються з відсутністю єдиного зручного інструменту, який би поєднував у

собі чіткі інструкції щодо того, які продукти підлягають переробці та куди їх можна здати, популяризацію концепції "другого життя" речей (upcycling) через користувацький контент, координацію екологічних заходів та подій.

Розробка спеціалізованого вебзастосунку з розгалуженою системою прав доступу (користувач, модератор, адміністратор) дозволить не лише систематизувати екологічні дані, а й створити активну спільноту свідомих споживачів. Це робить тему дослідження актуальною як з точки зору екологічного менеджменту, так і з позиції розробки сучасних багаторівневих вебсистем.

Метою дипломної роботи є проєктування та розробка багатофункціонального вебзастосунку для управління екологічними ініціативами, який сприятиме підвищенню рівня екологічної грамотності населення та стимулюватиме повторне використання ресурсів.

1.4. Визначення завдань кваліфікаційної роботи

Для досягнення поставленої мети кваліфікаційної роботи, що полягає у розробці вебзастосунку екологічного спрямування, необхідно вирішити комплекс взаємопов’язаних завдань:

- проаналізувати існуючі рішення у сфері екологічних застосунків та визначити необхідний функціонал;
- спроектувати архітектуру бази даних для збереження інформації про продукти, майстер-класи, події та користувачів із різними ролями;
- реалізувати інтерфейс для перегляду бази даних вторинної сировини;
- розробити систему створення та модерації авторського контенту (статей про "друге життя" речей) з можливістю коментування та збереження в обране;
- впровадити модуль управління екологічними подіями та реєстрації учасників на заходи.

Виконання зазначених завдань дозволить створити повнофункціональний вебзастосунок, який відповідатиме сучасним вимогам до програмних систем екологічного спрямування та забезпечить ефективну взаємодію користувачів із екологічним контентом.

2. ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ

2.1. Розробка математичної моделі

Для забезпечення ефективного функціонування вебзастосунку екологічного спрямування необхідно формалізувати основні процеси системи у вигляді математичної моделі. Така модель дозволяє описати структуру системи, взаємозв'язки між її елементами, а також логіку обробки даних і взаємодії користувачів.

Розглянемо вебзастосунок як множину взаємопов'язаних компонентів:

$$S = \{U, A, E, R, C\} \tag{2.1}$$

де:

U – множина користувачів;

A – множина статей (майстер-класів);

E – множина подій;

R – довідник переробки (множина продуктів);

C – множина коментарів.

Кожен користувач $u \in U$ характеризується набором атрибутів:

$$u = (id, name, email, role) \tag{2.2}$$

де $role \in \{guest, user, user-pro, moderator, admin\}$.

Множина статей описується як:

$$A = \{a_i\}, \tag{2.3}$$

$$a_i = \{id, title, content, author\_id, status\} \tag{2.4}$$

де status відображає стан модерації.

Множина подій:

$$E = \{e_i\}, \tag{2.5}$$

$$e_i = (\text{id,title,description,date, location}) \tag{2.6}$$

Довідник переробки продуктів:

$$R = \{r_i\}, \tag{2.7}$$

$$r_i = (id,product\_name,recyclable,disposal\_info) \tag{2.8}$$

де recyclable ознака можливості переробки.

Множина коментарів:

$$C = \{c_i\}, \tag{2.9}$$

$$c_i = (id,user\_id,article\_id,text,date) \tag{2.10}$$

Для опису системи доступу введемо функцію авторизації:

$$auth(u) = role \tag{2.11}$$

яка визначає права користувача. Доступ до функцій системи задається умовами:

- якщо role = guest , дозволено перегляд даних;
- якщо role = user, дозволено створення контенту та взаємодія з ним;
- якщо role = user-pro, дозволено створення контенту та заходу, взаємодія з ним;
- якщо role = moderator, дозволено модерацію статей;
- якщо role = admin, дозволено повне управління системою.

Ефективність функціонування системи можна оцінити за показником активності користувачів:

$$Q = \alpha * |A| + \beta * |C| + \gamma * \left|E_{reg}\right| \quad (2.12)$$

де:

|A| – кількість статей;

|C| – кількість коментарів;

$|E_{reg}|$ – кількість реєстрацій на події;

α, β, γ – вагові коефіцієнти.

Таким чином, розроблена математична модель дозволяє формалізувати основні елементи вебзастосунку, їх взаємозв'язки та логіку роботи. Вона є основою для подальшого проєктування архітектури системи, структури бази даних і реалізації програмного забезпечення.

2.2. Визначення користувацьких ролей і рівнів доступу до системи

Одним із ключових аспектів розробки вебзастосунку є забезпечення безпеки, цілісності даних та коректної взаємодії користувачів із системою. Для цього реалізується механізм розмежування прав доступу на основі ролей користувачів, який дозволяє визначити дозволені дії для кожної категорії користувачів.

У межах розроблюваного вебзастосунку передбачено чотири основні ролі: незареєстрований користувач (guest), зареєстрований користувач (user), модератор (moderator) та адміністратор (admin).

Незареєстрований користувач (guest) має базовий рівень доступу до системи. Йому доступні лише функції перегляду інформації без можливості взаємодії з контентом. Зокрема, такий користувач може:

- здійснювати пошук інформації про можливість переробки продуктів;
- переглядати статті (майстер-класи) щодо повторного використання речей;

- переглядати список екологічних подій та заходів.

Зареєстрований користувач (user) отримує розширений функціонал після проходження процедури автентифікації. До його можливостей належать:

- створення та редагування власних статей (майстер-класів);
- перегляд і коментування статей інших користувачів;
- збереження обраних матеріалів;
- реєстрація на події та заходи;
- управління власним профілем.

Користувач-pro (user-pro) – це зареєстрований користувач із розширеними правами, який, окрім базового функціоналу, має можливість створювати власні екологічні події та заходи. Дана роль призначена для активних учасників екологічної спільноти, організаторів ініціатив або представників організацій. До функціональних можливостей користувача-про належать:

- створення, редагування та видалення власних подій і заходів;
- перегляд списку створених подій;
- управління інформацією про події (опис, дата, місце проведення);
- взаємодія з учасниками заходів (перегляд зареєстрованих користувачів за потреби);
- усі можливості стандартного зареєстрованого користувача (створення статей, коментування, збереження матеріалів, реєстрація на події).

Водночас створені користувачем-pro події можуть підлягати перевірці або редагуванню з боку адміністратора, що забезпечує контроль якості контенту та достовірність інформації.

Модератор (moderator) відповідає за контроль якості контенту, що публікується в системі. Його основні функції:

- перевірка статей перед публікацією;
- схвалення або відхилення матеріалів;
- видалення некоректного або недопустимого контенту;

- контроль дотримання правил користування системою.

Адміністратор (admin) має найвищий рівень доступу та повний контроль над системою. До його функцій належать:

- управління користувачами (створення, блокування, видалення);
- призначення та зміна ролей;
- додавання, редагування та видалення подій і заходів;
- контроль загального функціонування системи.

Реалізація такої моделі забезпечує:

- захист даних від несанкціонованого доступу;
- розмежування відповідальності між користувачами;
- ефективне управління контентом;
- масштабованість системи при додаванні нових функцій.

Таким чином, впровадження ролей і рівнів доступу є необхідною умовою для забезпечення безпечної, стабільної та зручної роботи вебзастосунку екологічного спрямування.

2.3. Створення структурної схеми та діаграми прецедентів

На рисунку 2.1 представлено структурну діаграму вебзастосунку для управління екологічного спрямування, яка відображає основні компоненти системи та взаємозв’язки між ними. Архітектура системи побудована за принципом поділу на три ключові рівні: інтерфейс користувача, серверна логіка та рівень зберігання даних.

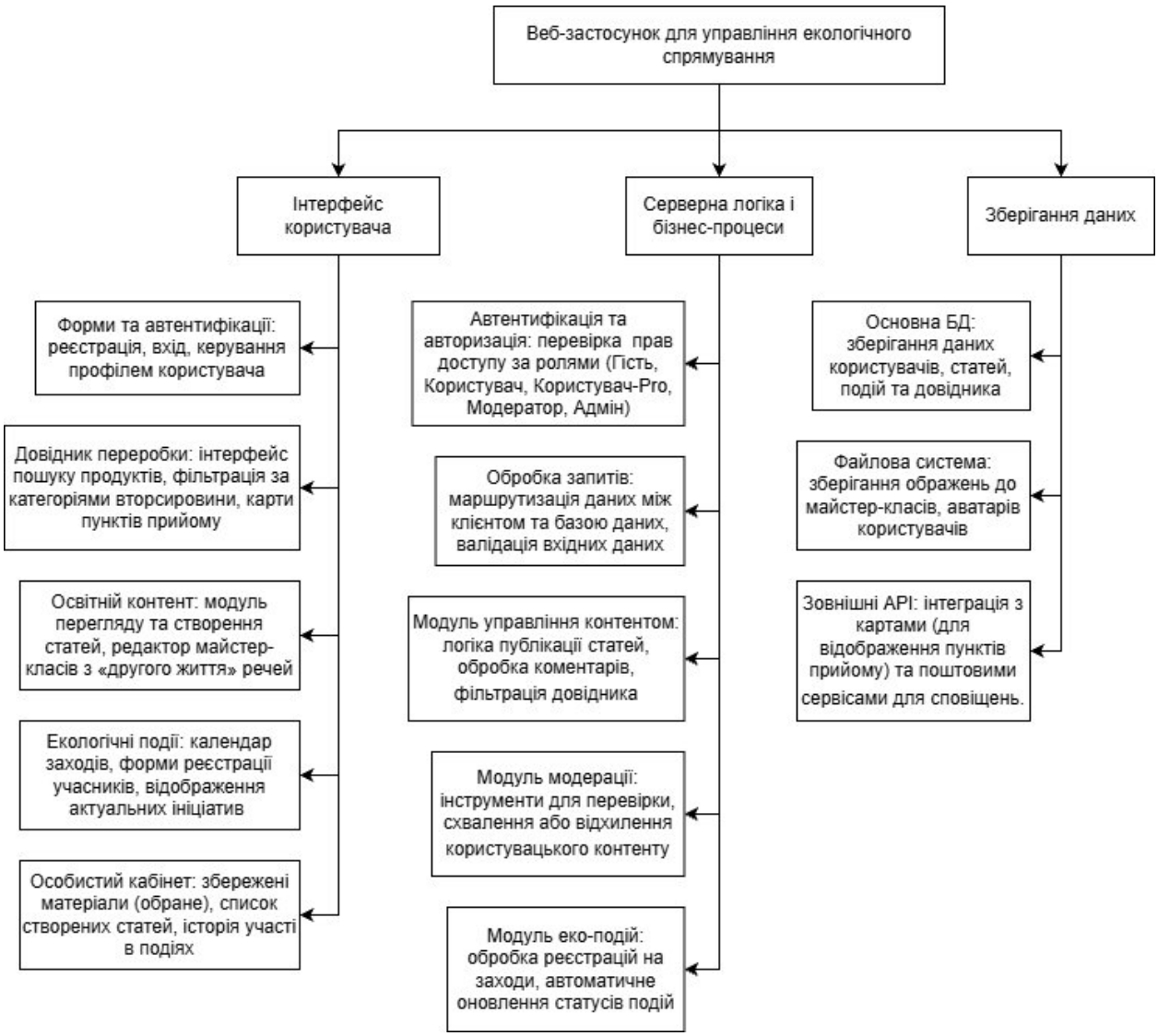

Рисунок 2.1 – Структурна схема вебзастосунку

Інтерфейс користувача забезпечує взаємодію користувача із системою та включає такі основні модулі:

- модуль автентифікації, що реалізує реєстрацію, вхід у систему та керування профілем користувача;
- довідник переробки, який надає можливість пошуку продуктів, фільтрації за категоріями вторинної сировини та перегляду пунктів прийому;
- модуль освітнього контенту, що дозволяє переглядати та створювати статті й майстер-класи з повторного використання речей;
- модуль екологічних подій, який відображає календар заходів та забезпечує реєстрацію користувачів;

- особистий кабінет, де користувач може переглядати збережені матеріали, власні публікації та історію участі в подіях.

Серверна логіка та бізнес-процеси відповідають за обробку запитів, реалізацію функціональності системи та взаємодію з базою даних. До цього рівня належать:

- модуль автентифікації та авторизації, який перевіряє права доступу відповідно до ролей користувачів (гість, користувач, модератор, адміністратор);
- модуль обробки запитів, що забезпечує маршрутизацію даних між клієнтською частиною та базою даних, а також валідацію введених даних;
- модуль управління контентом, який реалізує логіку створення, редагування та відображення статей, а також обробку коментарів;
- модуль модерації, що забезпечує перевірку, схвалення або відхилення користувацького контенту;
- модуль управління подіями, який відповідає за реєстрацію користувачів на заходи та оновлення статусів подій.

Рівень зберігання даних призначений для збереження та управління інформацією. Він включає:

- основну базу даних, у якій зберігаються дані про користувачів, статті, події та довідник переробки;
- файлову систему для зберігання мультимедійних ресурсів, таких як зображення для статей і аватари користувачів;
- інтеграцію із зовнішніми API, зокрема картографічними сервісами для відображення пунктів прийому вторинної сировини та поштовими сервісами для надсилання сповіщень.

Взаємодія між компонентами системи відбувається за клієнт-серверною моделлю: користувач через інтерфейс формує запит, який обробляється серверною частиною, після чого здійснюється звернення до бази даних або зовнішніх сервісів, і результат повертається користувачу у зручному вигляді.

Таким чином, представлена діаграма відображає логічну структуру вебзастосунку та демонструє розподіл функціональності між його основними складовими, що забезпечує масштабованість, зручність використання та ефективне управління екологічним контентом.

На рисунку 2.2 представлено діаграму прецедентів створюваного вебзастосунку для управління екологічного спрямування. Дана діаграма відображає взаємодію користувачів різних ролей із функціональними можливостями системи [11].

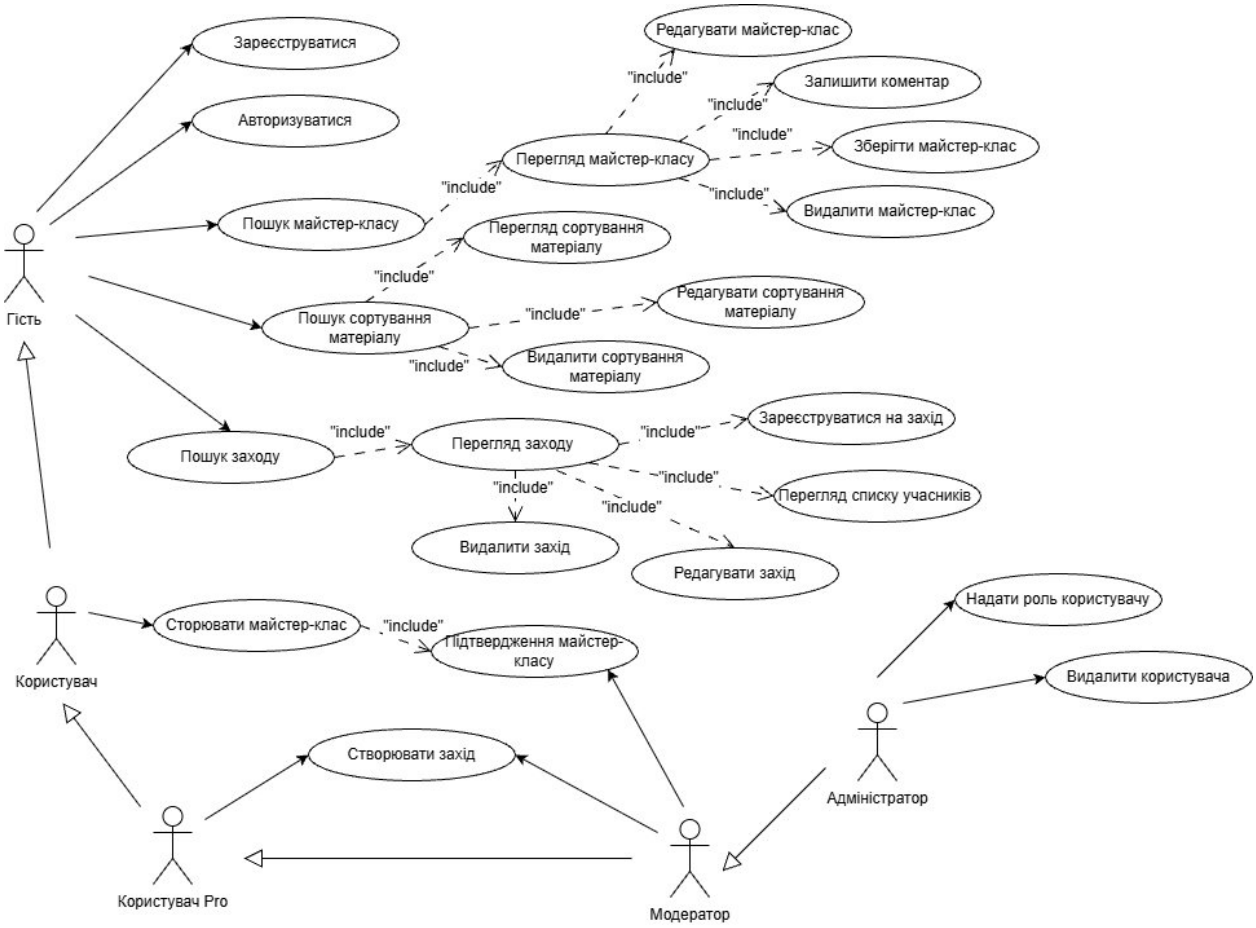

Рисунок 2.2 – Діаграма прецедентів створюваного вебзастосунку

У системі визначено п’ять основних акторів: гість, користувач, користувач-pro, модератор та адміністратор, кожен із яких має відповідний рівень доступу до функцій системи.

Гість (неавторизований користувач) має базові можливості взаємодії із системою. Він може:

- зареєструватися та авторизуватися в системі;

- здійснювати пошук майстер-класів;
- переглядати матеріали щодо сортування відходів;
- здійснювати пошук екологічних заходів і переглядати інформацію про них.

Доступ до детального перегляду контенту реалізується через зв’язки типу include, що означає використання базових сценаріїв у більш складних.

Користувач (зареєстрований) має розширені можливості порівняно з гостем. Він може:

- створювати майстер-класи;
- проходити процес підтвердження створених матеріалів;
- взаємодіяти з контентом після його перегляду (зокрема через включені сценарії).

Користувач-pro є розширеною роллю користувача та додатково має можливість:

- створювати екологічні заходи;
- брати участь в управлінні подіями.

Ця роль успадковує всі можливості звичайного користувача, що відображено відповідним зв’язком узагальнення на діаграмі.

Модератор відповідає за контроль якості контенту. Його функції включають:

- підтвердження або відхилення майстер-класів;
- участь у процесі перевірки користувацького контенту;
- часткове управління подіями та матеріалами.

Модератор взаємодіє з процесом публікації через сценарій підтвердження контенту, що є обов’язковим етапом перед його відображенням у системі.

Адміністратор має найвищий рівень доступу та виконує функції управління системою, зокрема:

- надання ролей користувачам;
- видалення користувачів;

~ загальний контроль функціонування системи.

Окрему групу варіантів використання становлять сценарії, пов’язані з контентом і подіями. Після перегляду майстер-класу користувач може виконувати додаткові дії, такі як: редагування, коментування, збереження або видалення матеріалу. Аналогічно, при роботі з подіями доступні функції реєстрації на захід, перегляду списку учасників, редагування або видалення заходу.

Зв’язки типу include, використані на діаграмі, відображають залежність між сценаріями, коли виконання одного варіанту використання включає інший як обов’язкову складову.

Таким чином, дана діаграма демонструє розподіл функціональних можливостей між ролями користувачів, логіку їх взаємодії із системою та структуру основних сценаріїв використання вебзастосунку.

2.4. Розробка функціональної діаграми

У межах проєктування вебзастосунку для управління екологічного спрямування було побудовано функціональну модель системи за методологією IDEF0 [12], яка дозволяє формалізувати основні бізнес-процеси, інформаційні потоки та взаємодію між компонентами системи.

Дана модель складається з контекстної діаграми, що відображає систему в цілому, та діаграм декомпозиції, які деталізують її внутрішню структуру.

Контекстна діаграма (Рисунок 2.3) представляє вебзастосунок як єдиний функціональний блок, що взаємодіє із зовнішнім середовищем через вхідні, вихідні потоки, керуючі впливи та механізми реалізації.

Рисунок 2.3 – Контекстна діаграма

До системи надходять такі основні типи даних:

- дані користувачів, що включають персональну інформацію (ПІБ, електронну пошту, пароль), необхідну для автентифікації та створення профілю;
- інформація про продукти, яка використовується в довіднику переробки (назва продукту, тип матеріалу, категорія відходів);
- екологічний контент, що включає тексти статей, інструкції, зображення та матеріали майстер-класів, створені користувачами.

Функціонування системи регламентується наступними факторами:

- бізнес-правилами, які визначають ролі користувачів (гість, користувач, користувач-про, модератор, адміністратор) та їх права доступу;
- екологічним законодавством і правилами сортування, що визначають коректність інформації у довіднику переробки;
- системними параметрами, зокрема конфігурацією серверного середовища та налаштуваннями механізмів автентифікації (JWT).

Реалізація функціональності системи здійснюється за допомогою:

- користувачів, які взаємодіють із системою;

- технологічної платформи, що включає клієнтську частину (React), серверну логіку (Node.js) та систему управління базою даних.

Результатом роботи системи є:

- токени автентифікації, які визначають рівень доступу користувача;
- інформаційні ресурси, зокрема опубліковані статті, довідник переробки;
- дані про події, включаючи списки учасників та розклад заходів.

Таким чином, контекстна діаграма відображає загальну архітектуру системи, її взаємодію із зовнішніми сутностями та основні потоки даних.

Для детальнішого аналізу функціонування вебзастосунку було виконано декомпозицію основного процесу на чотири взаємопов’язані підсистеми A1–A4.

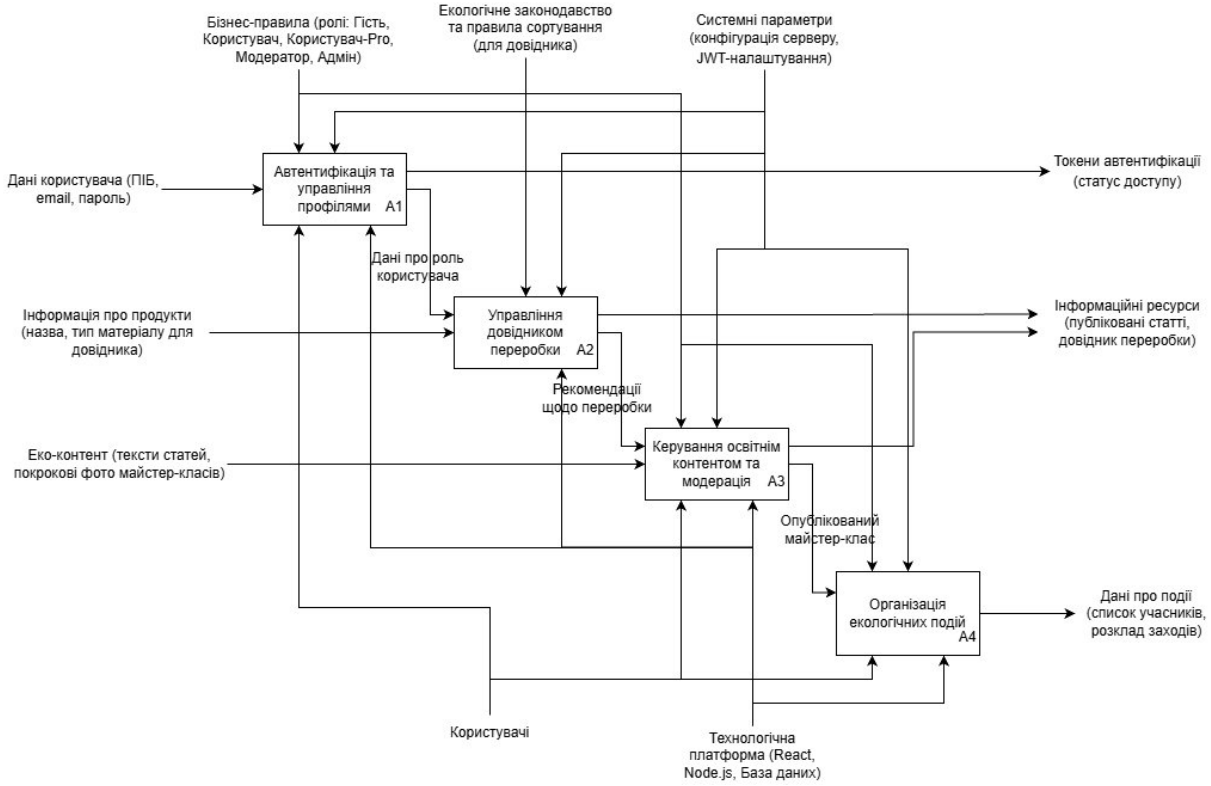

Рисунок 2.4 – Діаграма першого рівня декомпозиції контекстного представлення

На рисунку 2.5 зображено декомпозиція рівня A1 – автентифікація та управління профілями. Цей підпроцес є базовим для функціонування всієї системи, оскільки забезпечує ідентифікацію користувачів і контроль доступу.

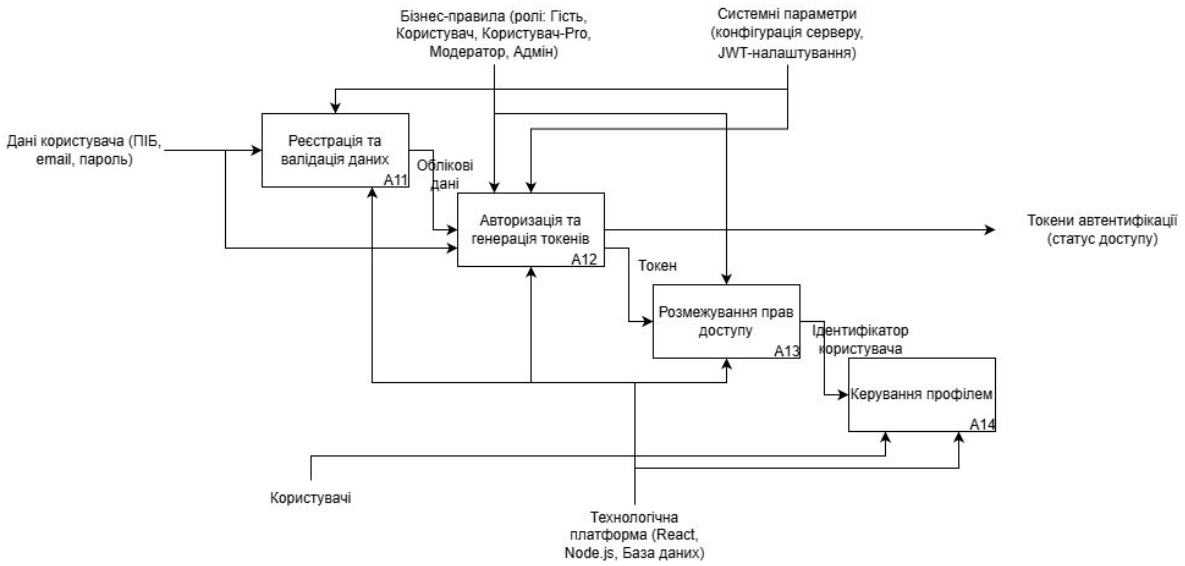

Рисунок 2.5 – Діаграма другого рівня декомпозиції автентифікація та управління профілями

Основні функції процесу:

- реєстрація користувачів, яка включає введення даних та їх валідацію;
- авторизація, під час якої перевіряються облікові дані та генеруються токени доступу;
- розмежування прав доступу, що визначає доступні функції залежно від ролі користувача;
- керування профілем, включаючи зміну особистих даних.

Взаємодія цього процесу з іншими підсистемами забезпечує безпечний доступ до функцій системи.

На рисунку 2.6 зображено декомпозиція рівня А2 – управління довідником переробки. Даний процес реалізує ключову функціональність системи, пов’язану з інформуванням користувачів щодо поводження з відходами.

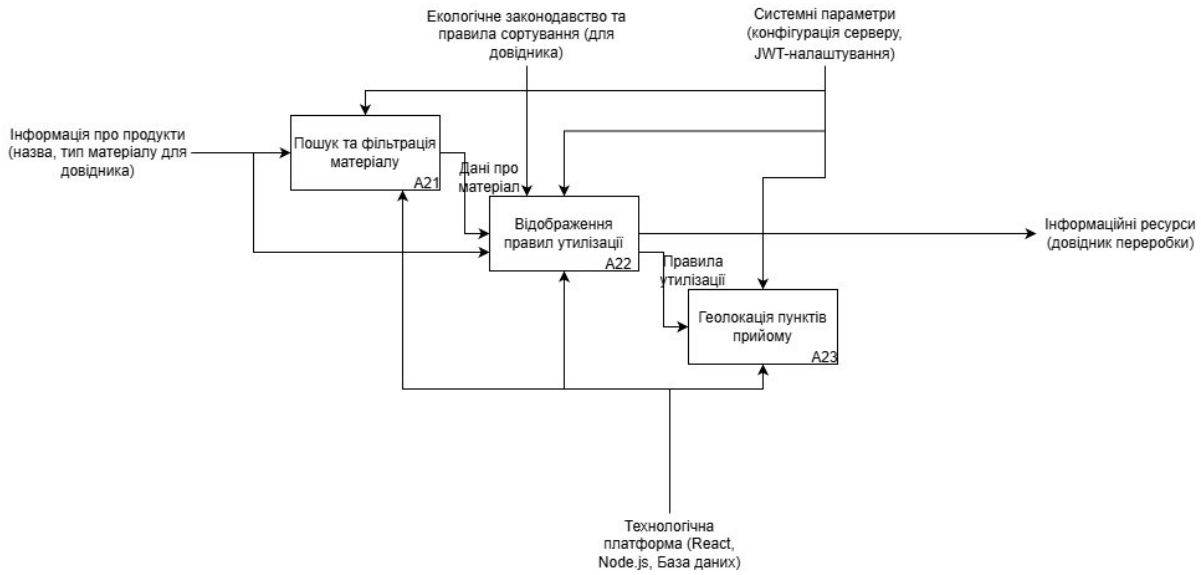

Рисунок 2.6 – Діаграма другого рівня декомпозиції управління довідником переробки

Основні функції:

- пошук та фільтрація матеріалів за категоріями;
- відображення правил утилізації, сформованих на основі екологічних норм;
- визначення геолокації пунктів прийому, що дозволяє користувачам знаходити найближчі місця здачі відходів;
- формування рекомендацій, які допомагають користувачам приймати екологічно правильні рішення.

Цей модуль відіграє важливу роль у підвищенні екологічної обізнаності користувачів.

На рисунку 2.7 зображено декомпозиція рівня А3 – керування освітнім контентом та модерація. Ця підсистема відповідає за створення, перевірку та публікацію освітнього контенту.

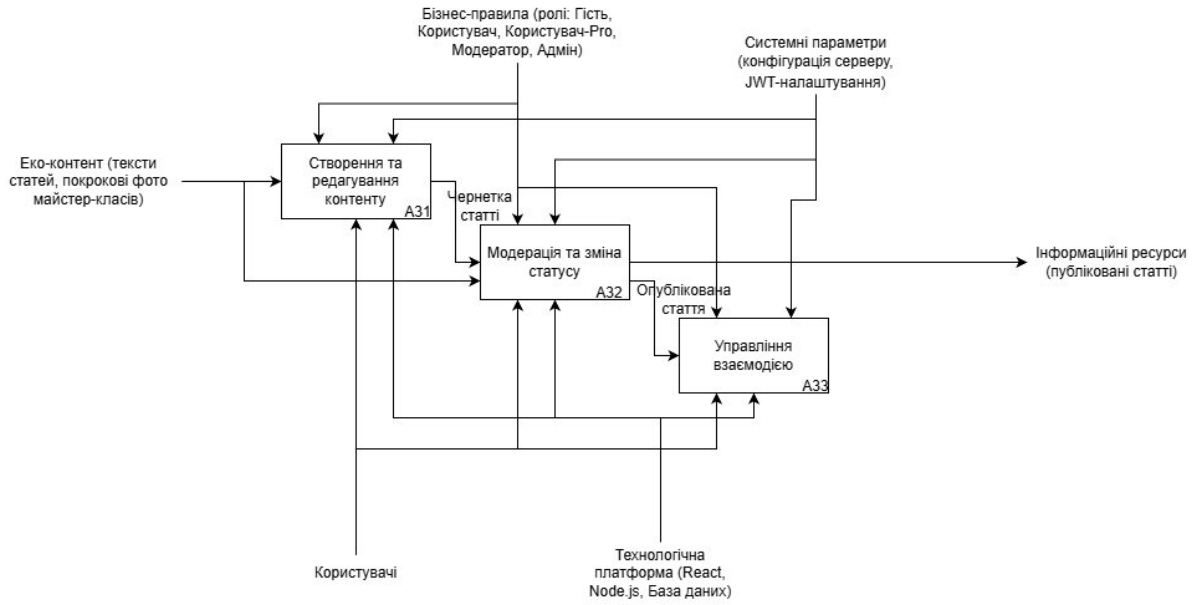

Рисунок 2.7 – Діаграма другого рівня декомпозиції керування освітнім контентом та модерація

Функціональні можливості:

- створення та редагування статей і майстер-класів користувачами;
- модерація контенту, яка включає перевірку та зміну статусу матеріалів;
- публікація статей, що стають доступними для всіх користувачів;
- управління взаємодією, включаючи коментування, збереження та оцінювання контенту.

Наявність модерації забезпечує якість і достовірність інформації.

На рисунку 2.8 зображено декомпозиція рівня А4 – організація екологічних подій. Цей підпроцес забезпечує функціональність, пов’язану з організацією та проведенням заходів.

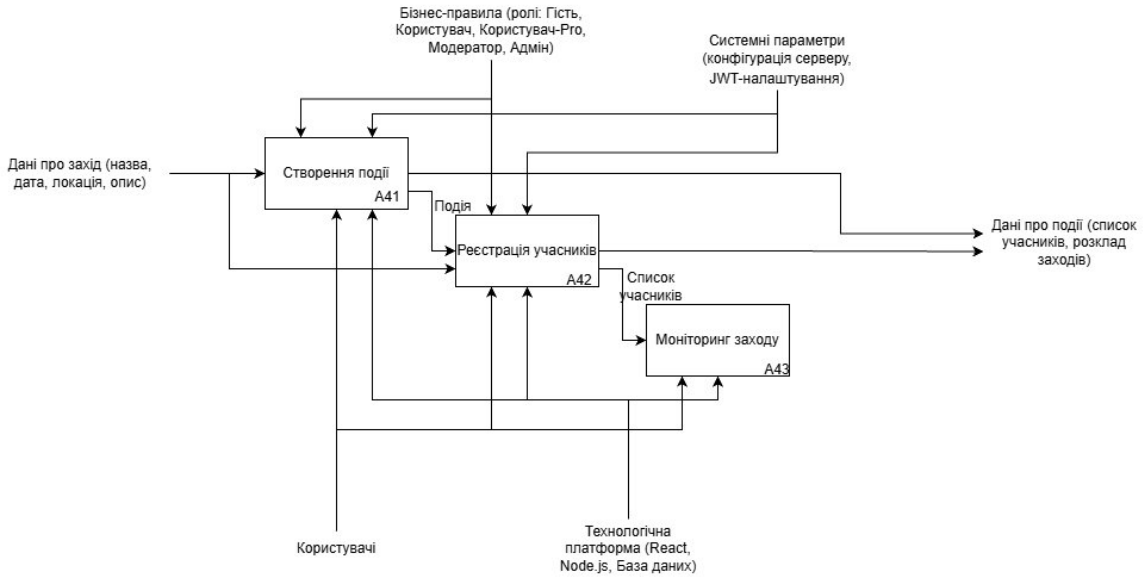

Рисунок 2.8 – Діаграма другого рівня декомпозиції організація екологічних подій організація екологічних подій

Основні функції:

- створення подій, що доступне, зокрема, користувачам-про;
- реєстрація учасників, яка дозволяє користувачам долучатися до заходів;
- моніторинг заходів, що включає відстеження активності та статусу події;
- формування списків учасників і розкладу.

Цей модуль сприяє розвитку екологічної спільноти та залученню користувачів до реальних ініціатив.

Запропонована декомпозиція демонструє, що система має модульну структуру, у якій кожен компонент виконує чітко визначені функції, а їх взаємодія забезпечує комплексне вирішення задач екологічного інформування, взаємодії користувачів та організації подій.

2.5. Створення діаграми послідовностей

На рисунку 2.9 представлено діаграму послідовностей, яка відображає процес створення, перевірки та публікації статті (майстер-класу) у вебзастосунку екологічного спрямування. Діаграма демонструє взаємодію

між основними учасниками системи: користувачем, модератором, інтерфейсом (клієнтською частиною), сервером та базою даних [13].

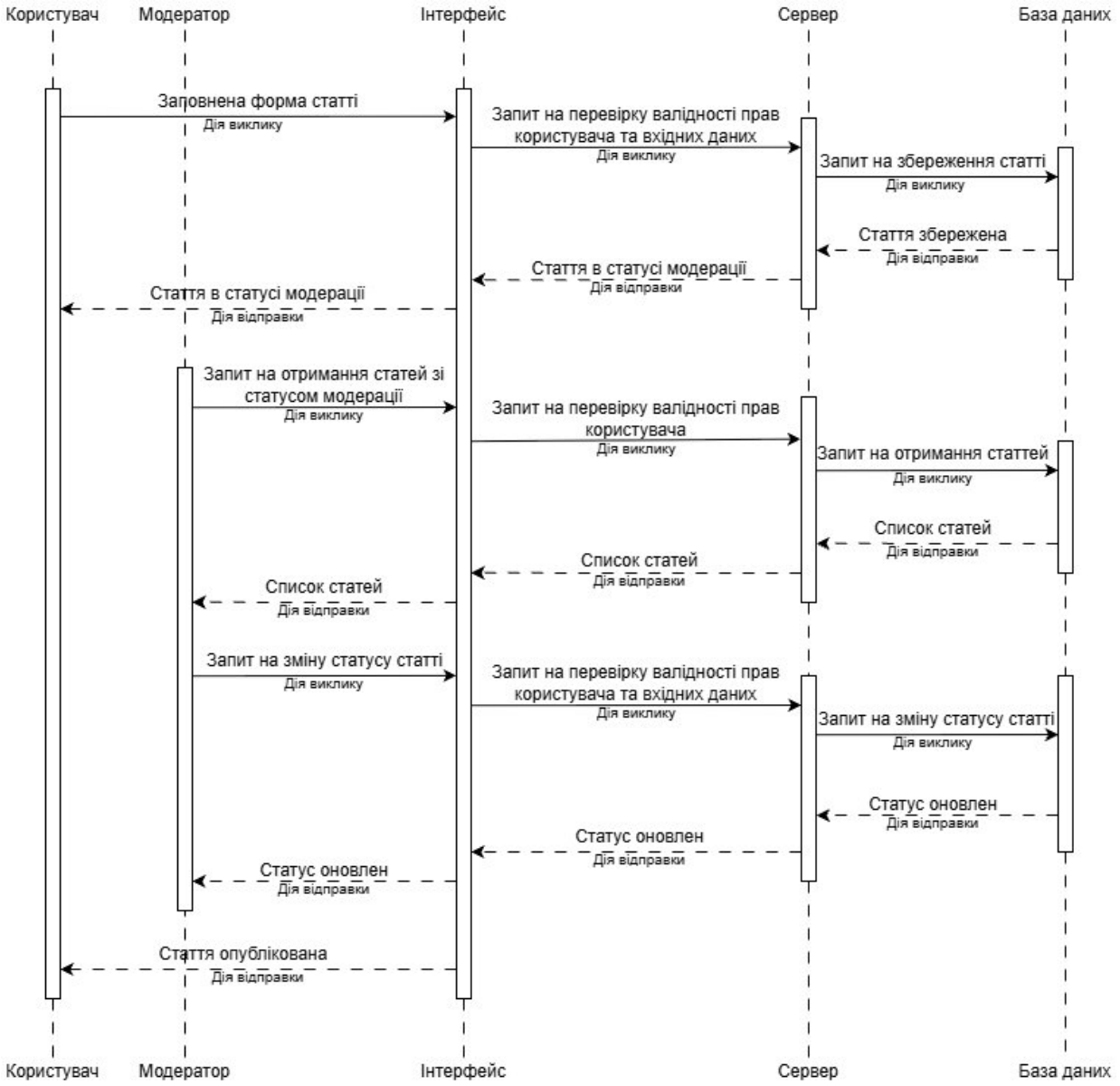

Рисунок 2.9 – Діаграма послідовностей

Процес починається з того, що користувач заповнює форму створення статті та надсилає її через інтерфейс системи. Інтерфейс передає запит на сервер для перевірки:

- коректності введених даних;
- прав доступу користувача.

Після успішної перевірки сервер формує запит до бази даних на збереження статті. У результаті стаття записується в базу даних зі статусом «на модерації».

Далі відповідь передається назад через сервер і інтерфейс до користувача, який отримує повідомлення про те, що його стаття перебуває на етапі перевірки.

Модератор ініціює запит на отримання списку статей, що знаходяться у статусі модерації. Інтерфейс передає цей запит на сервер, де відбувається перевірка прав доступу модератора. Після підтвердження прав сервер звертається до бази даних для отримання відповідних записів. База даних повертає список статей, який передається назад через сервер та інтерфейс до модератора. Таким чином, модератор отримує перелік матеріалів, що потребують перевірки.

Після перегляду матеріалу модератор може змінити його статус (наприклад, на «опубліковано» або «відхилено»). Для цього він надсилає відповідний запит через інтерфейс. Інтерфейс передає запит на сервер, де знову виконується:

- перевірка прав доступу модератора;
- валідація переданих даних.

Після цього сервер формує запит до бази даних для оновлення статусу статті. База даних змінює статус і повертає підтвердження виконання операції. Отриманий результат передається назад через сервер та інтерфейс до модератора.

У випадку схвалення статті її статус змінюється на «опублікована», після чого вона стає доступною для перегляду іншими користувачами системи. Цей етап завершує життєвий цикл обробки статті – від створення до публікації.

На діаграмі чітко простежується клієнт-серверна архітектура:

- усі запити ініціюються користувачами через інтерфейс;
- сервер виконує перевірку прав доступу та обробку логіки;
- база даних відповідає за збереження та отримання інформації.

Використання таких механізмів забезпечує:

- безпеку (через перевірку прав доступу);

- цілісність даних (через валідацію);
- контроль якості контенту (через модерацію).

Представлена діаграма послідовностей відображає логіку взаємодії компонентів системи під час обробки статей. Вона демонструє поетапний процес проходження даних через рівні системи – від введення користувачем до збереження в базі даних та подальшої модерації.

Це дозволяє забезпечити контроль якості контенту, правильне розмежування прав доступу та ефективну організацію роботи користувачів і модераторів у системі.

2.6. Побудова моделі «сутність-зв'язок» бази даних вебзастосунку

На рисунку 2.10 представлено ER-діаграму бази даних вебзастосунку для управління екологічного спрямування. Дана діаграма відображає основні сутності системи, їх атрибути та зв'язки між ними[14].

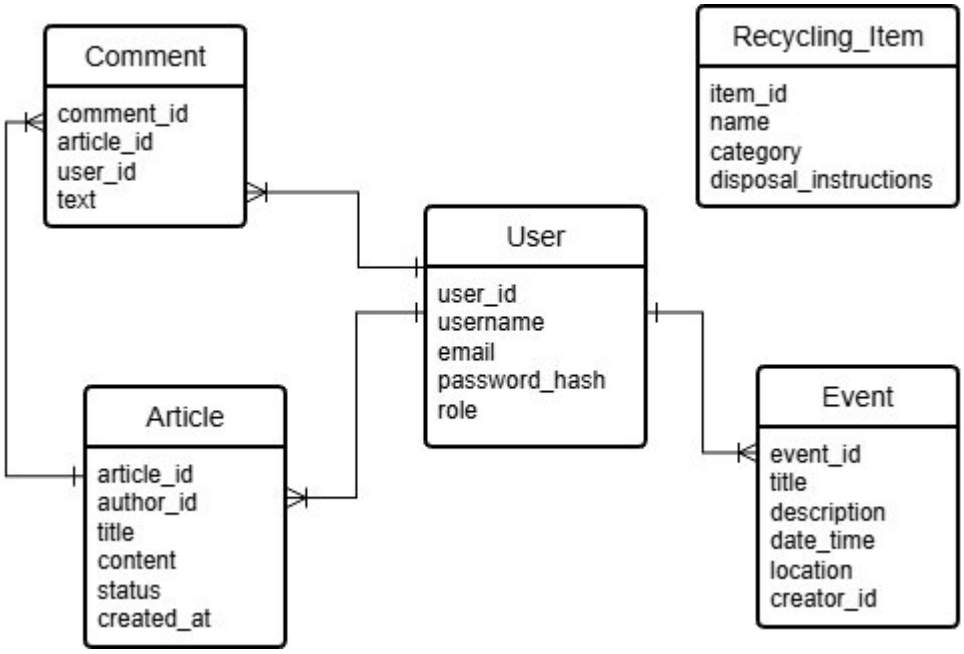

Рисунок 2.10 – ER-діаграма

Сутність User є центральною в системі та зберігає інформацію про всіх користувачів. Основні атрибути:

- user_id – унікальний ідентифікатор користувача;
- username – ім'я користувача;
- email – електронна адреса;

- password_hash – хеш пароля;
- role – роль користувача (гість, користувач, користувач-про, модератор, адміністратор).

Ця сутність пов’язана з іншими таблицями та визначає права доступу до функціоналу системи.

Сутність Article зберігає інформацію про майстер-класи та освітні матеріали. Атрибути:

- article_id – унікальний ідентифікатор статті;
- author_id – ідентифікатор автора (зв’язок із User);
- title – назва статті;
- content – текст статті;
- status – статус (на модерації, опублікована, відхилена);
- created_at – дата створення.

Сутність Comment призначена для зберігання коментарів до статей. Атрибути:

- comment_id – унікальний ідентифікатор;
- article_id – посилання на статтю;
- user_id – посилання на користувача;
- text – текст коментаря.

Сутність Event описує екологічні заходи. Атрибути:

- event_id – ідентифікатор події;
- title – назва;
- description – опис;
- date_time – дата і час проведення;
- location – місце проведення;
- creator_id – користувач, який створив подію.

Сутність Recycling_Item містить довідкову інформацію про продукти, що підлягають переробці. Атрибути:

- item_id – унікальний ідентифікатор;
- name – назва продукту;

- category – категорія матеріалу;
- disposal_instructions – інструкції щодо утилізації.

Ця сутність є незалежною та використовується для формування довідника переробки.

У діаграмі реалізовано такі основні зв’язки:

- User → Article: один користувач може створити багато статей;
- User → Comment: один користувач може залишити багато коментарів;
- Article → Comment: одна стаття може мати багато коментарів;
- User → Event: один користувач може створювати багато подій.

Усі зв’язки реалізуються за допомогою зовнішніх ключів (author_id, user_id, creator_id, article_id), що забезпечує цілісність даних у базі.

Особливості моделі

- використання ролей у сутності User дозволяє реалізувати систему розмежування доступу;
- наявність статусу статті забезпечує підтримку процесу модерації;
- розділення контенту (статті, коментарі, події) на окремі сутності забезпечує масштабованість системи;
- довідник переробки представлений окремою таблицею, що спрощує його оновлення та використання.

Розроблена ER-діаграма відображає логічну структуру бази даних вебзастосунку та забезпечує ефективне зберігання й обробку інформації. Вона охоплює всі ключові функціональні аспекти системи: управління користувачами, контентом, подіями та довідковими даними, що дозволяє реалізувати повноцінний екологічний вебсервіс.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛІВ ПРОЄКТУ

3.1. Вибір технологій реалізації та середовища розробки

Розробка платформи EcoWay велася у Visual Studio Code – редакторі коду з розширеною підтримкою TypeScript [15], вбудованим терміналом та плагінами для контролю якості коду. Клієнтська та серверна частини запускались паралельно на різних портах.

Платформа реалізована як повноцінний веб-застосунок за архітектурою клієнт-сервер, де Single Page Application взаємодіє з бекендом виключно через REST API. Такий підхід забезпечує чітке розділення відповідальності між частинами системи, незалежне масштабування та можливість повторного використання серверної логіки для потенційних майбутніх мобільних клієнтів. Єдиною мовою розробки для обох частин застосунку обрано TypeScript – статична типізація дозволяє виявляти помилки на етапі компіляції, покращує підтримку коду та дає змогу спільно використовувати типи й інтерфейси між клієнтом і сервером.

Серверна частина побудована на платформі Node.js [16] з використанням фреймворку Express.js [17]. Асинхронна неблокуюча модель Node.js забезпечує ефективну обробку великої кількості одночасних запитів, що є важливим для застосунків із динамічним навантаженням. Для роботи з базою даних застосовується зв'язка Knex.js [18] та Objection.js [19], які забезпечують версіонування схеми через міграції, валідацію даних та зручну роботу зі зв'язками між сутностями. Як СУБД обрано MySQL – реляційна модель добре відповідає предметній області та природно відображає зв'язки між основними сутностями системи. Аутентифікація користувачів реалізована на основі JWT із механізмом пари access/refresh токенів, що є стандартом для REST API. Паролі зберігаються у захищеному вигляді за допомогою алгоритму хешування bcrypt.

Клієнтська частина побудована на бібліотеці React [20], що забезпечує розробку інтерфейсу з ізольованих, повторно використовуваних компонентів.

Як інструмент збірки використовується Vite, що суттєво прискорює процес розробки завдяки миттєвому старту та гарячій заміні модулів. Клієнтська маршрутизація реалізована через React Router із підтримкою захищених маршрутів і розмежуванням доступу за роллю користувача. Стилізація компонентів здійснюється за допомогою Tailwind – CSS-фреймворку, що прискорює розробку інтерфейсу та забезпечує стилістичну узгодженість у межах усього застосунку. HTTP-запити до API виконуються через бібліотеку Axios, а інтерактивна картографічна функціональність реалізована засобами Leaflet.js.

3.2. Реалізація серверної частини

Серверна частина платформи EcoWay реалізована як REST API на основі Node.js та Express.js. Сервер приймає HTTP-запити від клієнтської частини, виконує бізнес-логіку та повертає відповіді у форматі JSON. Усі маршрути згруповані за функціональними модулями відповідно до основних сутностей системи: користувачі, статті, події та реєстрації.

Серверний код організований за модульним принципом: кожен ресурс має окремий файл маршрутів (router), контролер із бізнес-логікою та модель для взаємодії з базою даних. Такий підхід забезпечує чітке розділення відповідальності та спрощує підтримку й розширення системи. Express.js надає засоби для визначення маршрутів із підтримкою всіх основних HTTP-методів (GET, POST, PUT, DELETE), а також підключення middleware для обробки запитів.

Для обробки наскрізної функціональності використовується ланцюжок middleware. Middleware аутентифікації перевіряє наявність та валідність JWT access токена у заголовку кожного захищеного запиту. Middleware авторизації додатково перевіряє роль користувача та обмежує доступ до адміністративних маршрутів. Окремий middleware обробляє помилки на рівні всього застосунку та повертає клієнту уніфіковані відповіді з відповідними HTTP-статусами.

Аутентифікація реалізована на основі JWT із використанням пари токенів. Access token має короткий термін дії (15 хвилин) і передається у заголовку кожного запиту до захищених маршрутів. Refresh token зберігається на стороні клієнта та використовується для отримання нового access токена після його завершення без повторного введення облікових даних. Паролі користувачів перед збереженням у базі даних хешуються за допомогою алгоритму bcrypt, що унеможливлює їх відновлення у разі витоку даних.

Взаємодія з базою даних MySQL здійснюється через зв'язку Knex.js та Objection.js. Схема бази даних зберігає версії через міграції, що дозволяє відтворювати та оновлювати структуру таблиць у будь-якому середовищі. Моделі даних описують сутності системи та їх зв'язки: користувач може бути автором статей і подій, а також реєструватися на події інших організаторів.

Перед збереженням даних до бази виконується їх валідація на рівні моделей Objection.js із використанням JSON Schema. Це дозволяє відхиляти некоректні запити на ранньому етапі обробки та повертати клієнту зрозумілі повідомлення про помилки. Додаткова перевірка вхідних даних виконується на рівні контролерів – перевіряється наявність обов'язкових полів, коректність форматів та відповідність бізнес-правилам системи.

3.3. Реалізація клієнтської частини

Клієнтська частина платформи EcoWay реалізована як Single Page Application на основі бібліотеки React. Застосунок завантажується одноразово, а подальша навігація між сторінками відбувається без перезавантаження браузера – це забезпечує плавний користувацький досвід та швидку реакцію інтерфейсу на дії користувача. Взаємодія з сервером здійснюється виключно через REST API за допомогою HTTP-запитів у форматі JSON.

Клієнтський код організований за компонентним принципом: інтерфейс розбитий на ізольовані, повторно використовувані компоненти,

кожен з яких відповідає за окрему частину UI. Компоненти поділяються на сторінки (pages), що відповідають конкретним маршрутам, та допоміжні компоненти (components), які використовуються в різних частинах застосунку. Такий підхід спрощує підтримку коду та дозволяє легко розширювати функціональність без впливу на інші частини системи.

Навігація між сторінками реалізована засобами React Router. Маршрути описані декларативно та згруповані відповідно до рівня доступу. Публічні маршрути доступні всім відвідувачам, тоді як захищені маршрути перевіряють наявність аутентифікованого сесійного токена та роль користувача перед відображенням сторінки. У разі спроби неавторизованого доступу користувач автоматично перенаправляється на сторінку входу. Для адміністративного розділу реалізовано окремий рівень перевірки ролі.

Локальний стан компонентів керується за допомогою вбудованих хуків React – useState та useEffect. Для спільного стану, що використовується в різних частинах застосунку (зокрема, дані аутентифікованого користувача), застосовується React Context API. HTTP-запити до серверного API виконуються через бібліотеку Axios. Для автоматичного додавання JWT access токена до заголовків усіх запитів налаштовано інтерцептори, які також обробляють відповідь із кодом 401 та виконують автоматичне оновлення токена через refresh-механізм без втручання користувача.

Стилізація компонентів здійснюється за допомогою Tailwind CSS із використанням кастомної колірної палітри проєкту, що забезпечує візуальну узгодженість у межах усього застосунку. Інтерфейс є адаптивним і коректно відображається на пристроях із різною шириною екрану. Форми реєстрації, входу та створення контенту містять клієнтську валідацію з відображенням відповідних повідомлень про помилки. Для відображення списків статей та подій реалізовано фільтрацію за категоріями та пошук за ключовими словами.

Окремою складовою клієнтської частини є інтерактивна карта, реалізована за допомогою бібліотеки Leaflet.js [21] та її React-обгортки React

Leaflet. На карті відображаються актуальні екологічні події з кастомними маркерами, згрупованими за категоріями. Користувач може переглядати деталі події безпосередньо з карти, що забезпечує зручний альтернативний спосіб навігації по контенту платформи.

3.4. Налаштування бази даних

Як система управління реляційною базою даних для платформи EcoWay обрано MySQL. Реляційна модель добре відповідає предметній області застосунку – зв'язки між користувачами, статтями, подіями та реєстраціями природно відображаються у вигляді таблиць із зовнішніми ключами. MySQL забезпечує підтримку транзакцій, цілісність даних через обмеження та ефективну роботу з індексами, що є важливим для коректного функціонування системи.

База даних складається з кількох основних таблиць, що відображають ключові сутності системи. Таблиця користувачів (users) зберігає облікові дані та роль кожного користувача. Таблиці статей (articles) та подій (events) містять основний контент платформи з прив'язкою до автора через зовнішній ключ. Таблиця реєстрацій (registrations) реалізує зв'язок багато-до-багатьох між користувачами та подіями, фіксуючи факт участі користувача у конкретному заході. Між таблицями визначені зовнішні ключі з відповідними правилами каскадного видалення, що забезпечує цілісність даних.

Збереження версій схем бази даних здійснюється через систему міграцій Knex.js. Кожна міграція є окремим файлом із функціями створення та відкату змін, що дозволяє відтворювати актуальну структуру бази даних у будь-якому середовищі – розробки, тестування або реалізації продукту. Такий підхід усуває необхідність ручного внесення змін до схеми та забезпечує синхронізацію структури бази даних із версіями коду у системі контролю версій. Виконання міграцій здійснюється однією командою, що суттєво спрощує розгортання застосунку.

Для наповнення бази даних тестовими даними використовуються seeds – скрипти, що генерують початковий набір записів для кожної таблиці. Seeds містять тестових користувачів із різними ролями, набір статей та подій різних категорій, а також записи реєстрацій. Це дозволяє одразу після розгортання отримати повністю функціональний застосунок із демонстраційним контентом, що є зручним як для розробки, так і для презентації готового продукту.

Підключення до бази даних налаштовується через конфігураційний файл Knex, де визначаються параметри з'єднання: хост, порт, назва бази даних, а також облікові дані. Чутливі дані конфігурації виносяться у змінні середовища (.env) і не зберігаються у репозиторії, що відповідає стандартам безпеки. Пул з'єднань керується автоматично, що забезпечує ефективне використання ресурсів при одночасній обробці запитів від різних користувачів.

3.5. Тестування серверної частини

Тестування серверної частини платформи EcoWay проводилось за допомогою інструменту Postman [22], який дозволяє формувати HTTP-запити довільної структури та аналізувати відповіді сервера. Перевірці підлягали всі основні ендпоінти REST API, згруповані за функціональними модулями: аутентифікація, довідник переробки, статті та майстер-класи, події та адміністративне управління користувачами.

На рисунку 3.1 зображено результат тестування реєстрації користувача (POST /api/auth/register) за допомогою інструменту Postman. У тілі запиту передано JSON-об'єкт із тестовими обліковими даними: електронна адреса, ім'я та пароль. Сервер повернув відповідь із кодом «201 Created», що свідчить про успішне створення нового облікового запису. У тілі відповіді містяться дані створеного користувача (ідентифікатор, електронна адреса, ім'я та роль), а також згенерована пара токенів –

accessToken та refreshToken, – які використовуються для подальшої авторизації запитів у системі.

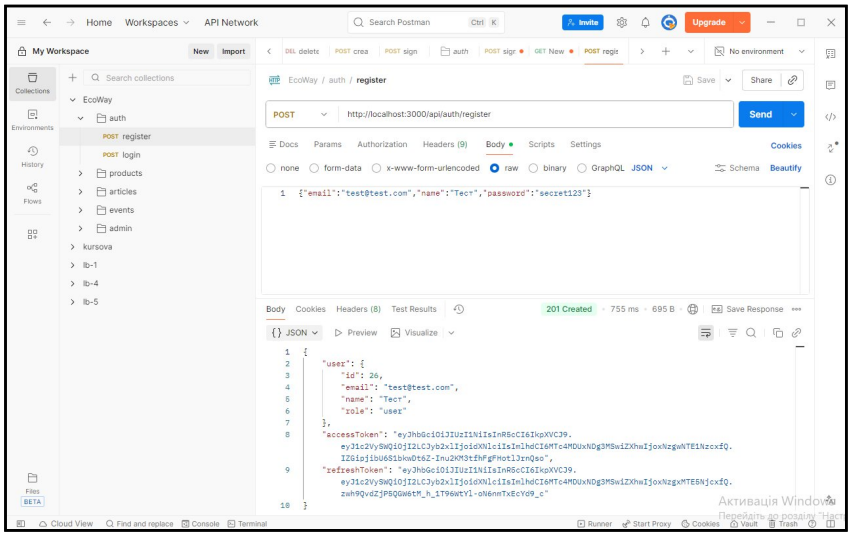

Рисунок 3.1 – Тестування реєстрації користувача через Postman

На рисунку 3.2 зображено результат тестування авторизації користувача (POST /api/auth/login) за допомогою Postman. У тілі запиту передано JSON-об'єкт із електронною адресою та паролем користувача, зареєстрованого на попередньому етапі тестування. Сервер повернув відповідь із кодом «200 OK», що підтверджує успішну автентифікацію. У тілі відповіді містяться дані авторизованого користувача (ідентифікатор, електронна адреса, ім'я та роль user), а також нова пара токенів – accessToken та refreshToken. Отримані токени використовувались у подальших тестах для перевірки захищених маршрутів API.

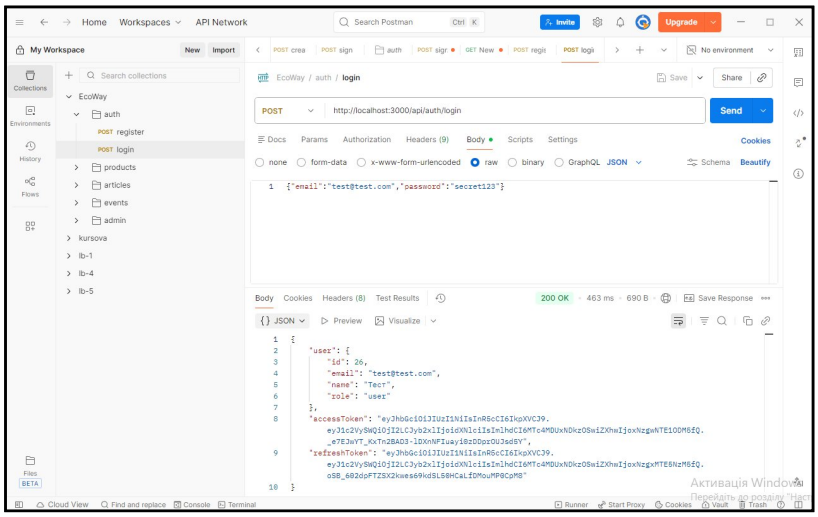

Рисунок 3.2 – Тестування авторизації користувача через Postman

На рисунку 3.3 зображено результат тестування отримання списку продуктів довідника переробки (GET /api/products) через Postman. Запит містить два параметри фільтрації: search зі значенням «пластик» та category зі значенням «Пластик», що демонструє роботу пошукового функціоналу модуля. Сервер повернув відповідь із кодом «200 OK». У тілі відповіді міститься масив відфільтрованих продуктів із детальною інформацією про кожен: ідентифікатор, назва, категорія матеріалу, ознака можливості переробки, інструкції щодо утилізації, дата створення запису та кількість пунктів прийому.

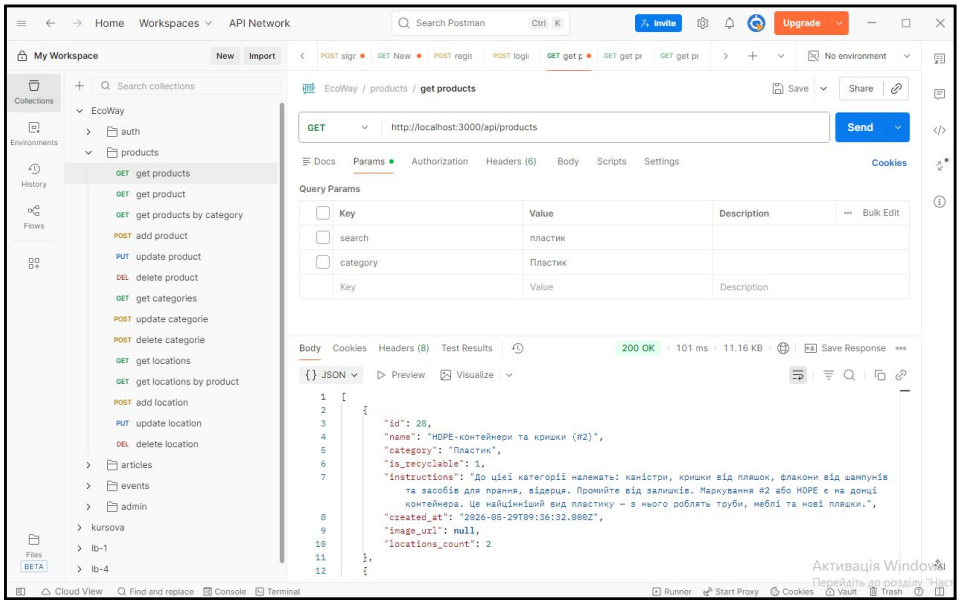

Рисунок 3.3 – Тестування отримання продуктів довідника переробки через Postman

На рисунку 3.4 зображено результат тестування пошуку конкретного продукту в довіднику переробки (GET /api/products?search=газети) через Postman. У параметрі запиту передано пошуковий запит «газети», за яким система повернула відповідь із кодом «200 OK». У тілі відповіді міститься знайдений продукт «Газети та журнали» із детальною інформацією: ознака переробки, розгорнуті інструкції щодо підготовки матеріалу до здачі, дата створення запису та кількість доступних пунктів прийому.

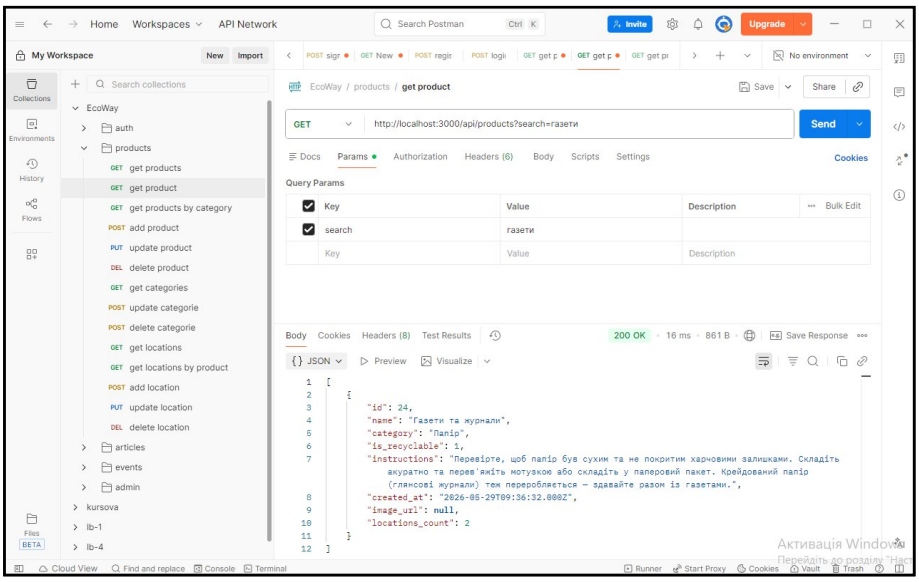

Рисунок 3.4 – Тестування пошуку продукту в довіднику переробки через Postman

На рисунку 3.5 зображено результат тестування додавання нового продукту до довідника переробки (POST /api/products) через Postman. У тілі запиту передано JSON-об'єкт із даними нового продукту: назва «Пляшка з-під парфюмів», категорія «Скло», ознака переробки та інструкції щодо утилізації. Сервер повернув відповідь із кодом «201 Created», що підтверджує успішне створення запису. У тілі відповіді відображаються дані збереженого продукту з присвоєним унікальним ідентифікатором та датою створення запису.

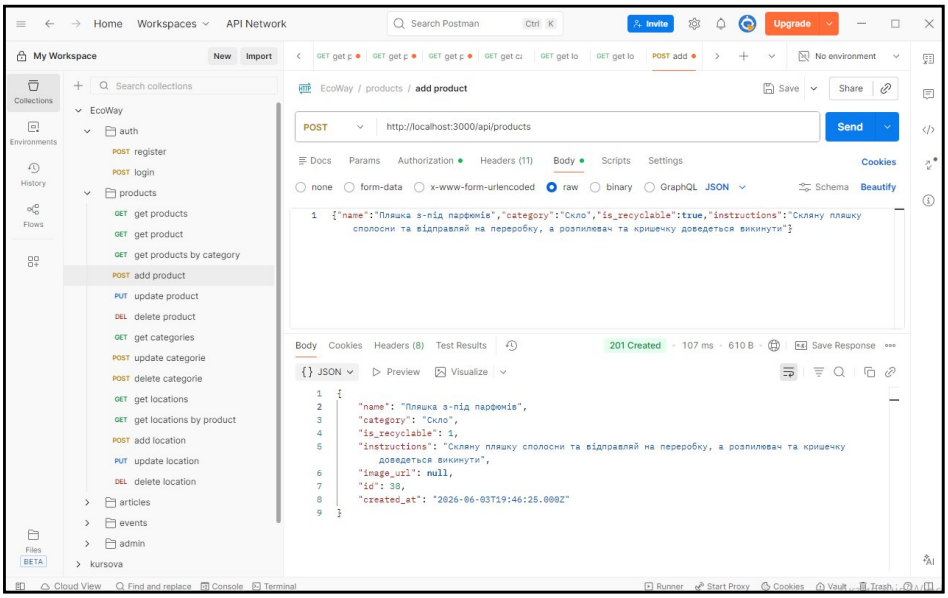

Рисунок 3.5 – Тестування додавання нового продукту через Postman

На рисунку 3.6 зображено результат тестування оновлення існуючого продукту в довіднику переробки (PUT /api/products/:id) через Postman. У тілі запиту передано JSON-об'єкт із оновленою назвою продукту. Сервер повернув відповідь із кодом «200 OK», що підтверджує успішне оновлення запису. У тілі відповіді відображаються актуальні дані продукту з ідентифікатором: оновлена назва, категорія, ознака переробки, повні інструкції щодо підготовки скляного посуду до здачі, дата створення запису.

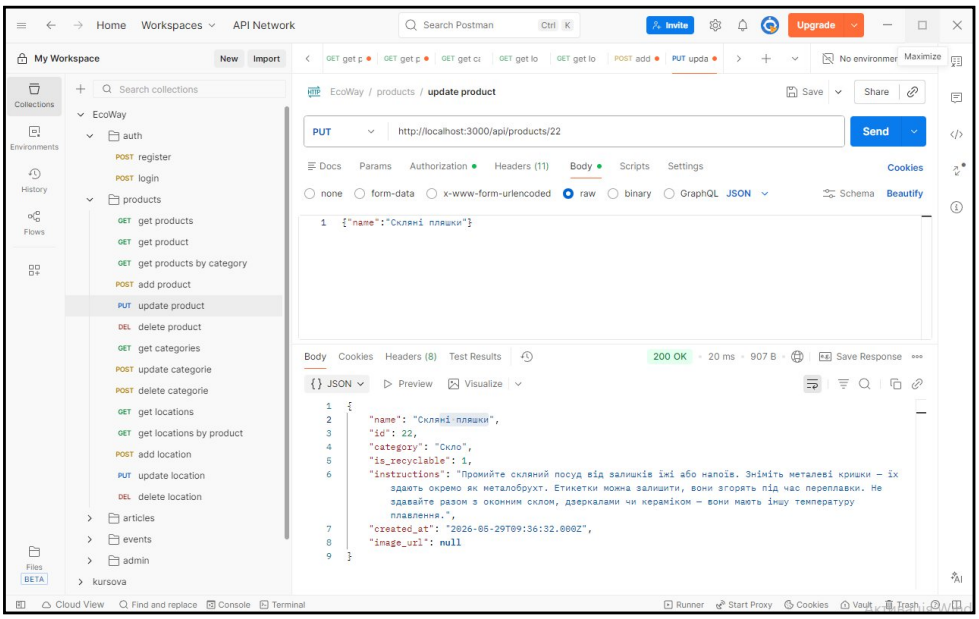

Рисунок 3.6 – Тестування оновлення продукту в довіднику переробки через Postman

На рисунку 3.7 зображено результат тестування видалення продукту з довідника переробки (DELETE /api/products/:id) через Postman. Запит надсилається без тіла – ідентифікатор продукту передається безпосередньо в URL. Сервер повернув відповідь із кодом «200 OK», що підтверджує успішне видалення запису. Операція доступна лише авторизованим користувачам із відповідними правами доступу, про що свідчить наявність токена авторизації у заголовках запиту.

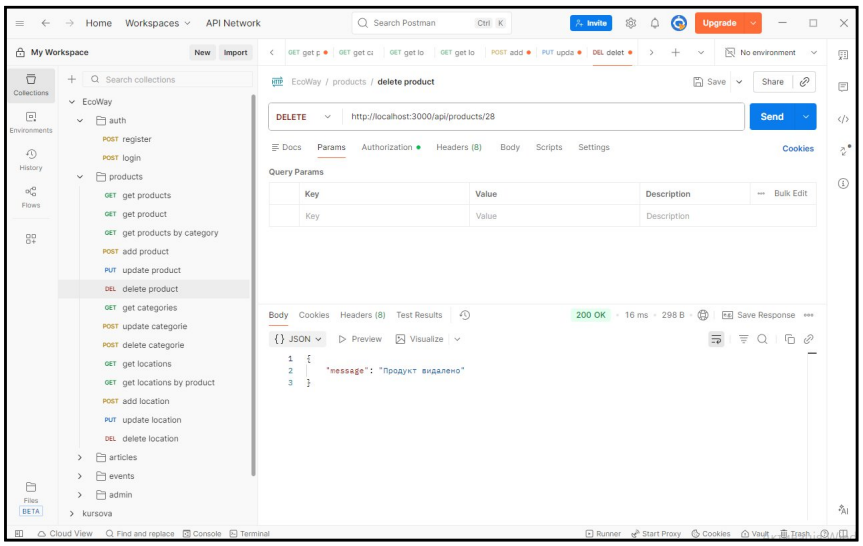

Рисунок 3.7 – Тестування видалення продукту з довідника переробки через Postman

На рисунку 3.8 зображено результат тестування отримання списку категорій матеріалів довідника переробки (GET /api/products/categories) через Postman. Запит надсилається без додаткових параметрів. Сервер повернув відповідь із кодом «200 OK». У тілі відповіді міститься масив усіх доступних категорій матеріалів: «Електроніка», «Метал», «Органіка», «Папір», «Пластик», «Скло» та «Текстиль». Цей запит використовується клієнтською частиною для формування списку фільтрів на сторінці довідника переробки, що дозволяє користувачам зручно звужувати результати пошуку за типом матеріалу.

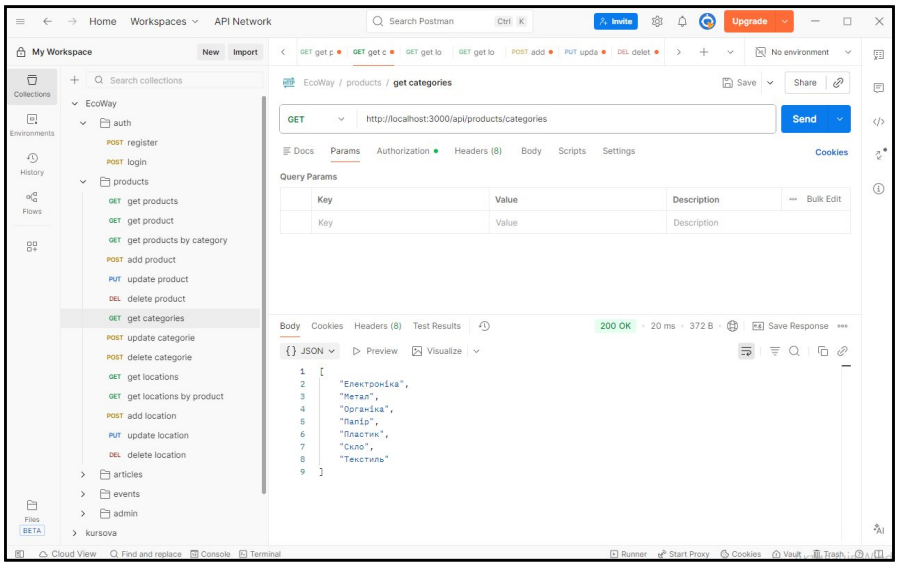

Рисунок 3.8 – Тестування отримання категорій матеріалів через Postman

На рисунку 3.9 зображено результат тестування отримання повного списку пунктів прийому вторинної сировини (GET /api/products/locations/all) через Postman. Запит надсилається без додаткових параметрів. Сервер повернув відповідь із кодом «200 OK». У тілі відповіді міститься масив пунктів прийому з детальною інформацією про кожен: унікальний ідентифікатор, назва пункту, адреса, географічні координати (latitude та longitude) для відображення на інтерактивній карті, номер телефону, години роботи, а також перелік матеріалів, які приймаються в цьому пункті з посиланням на відповідні продукти довідника.

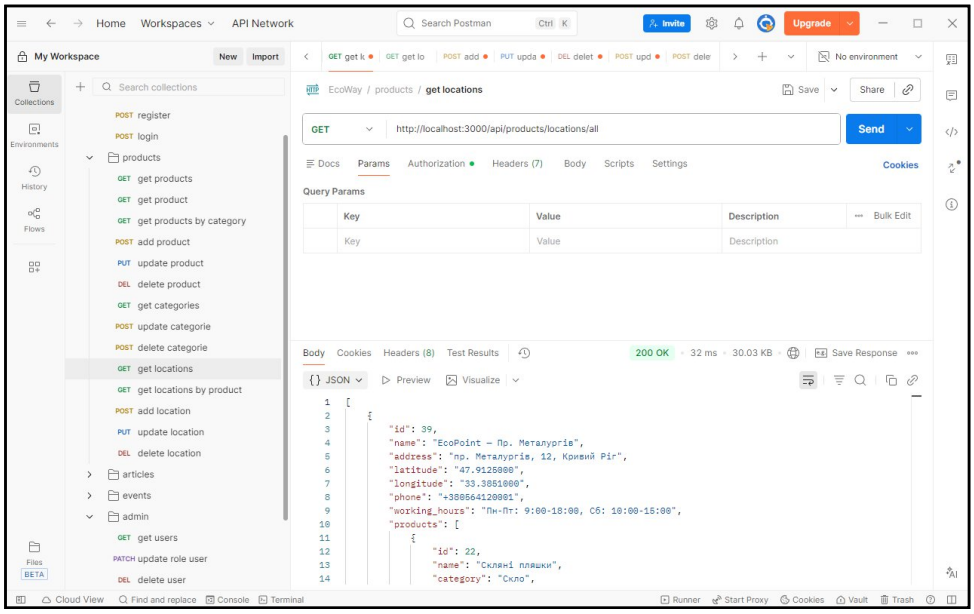

Рисунок 3.9 – Тестування отримання списку пунктів прийому вторинної сировини через Postman

На рисунку 3.10 зображено результат тестування додавання нового пункту прийому вторинної сировини (POST /api/products/locations) через Postman. У тілі запиту передано JSON-об'єкт із повними даними нового пункту: масив ідентифікаторів продуктів, що приймаються, назва, адреса, географічні координати, контактний номер телефону та години роботи. Сервер повернув відповідь із кодом «201 Created», що підтверджує успішне створення запису.

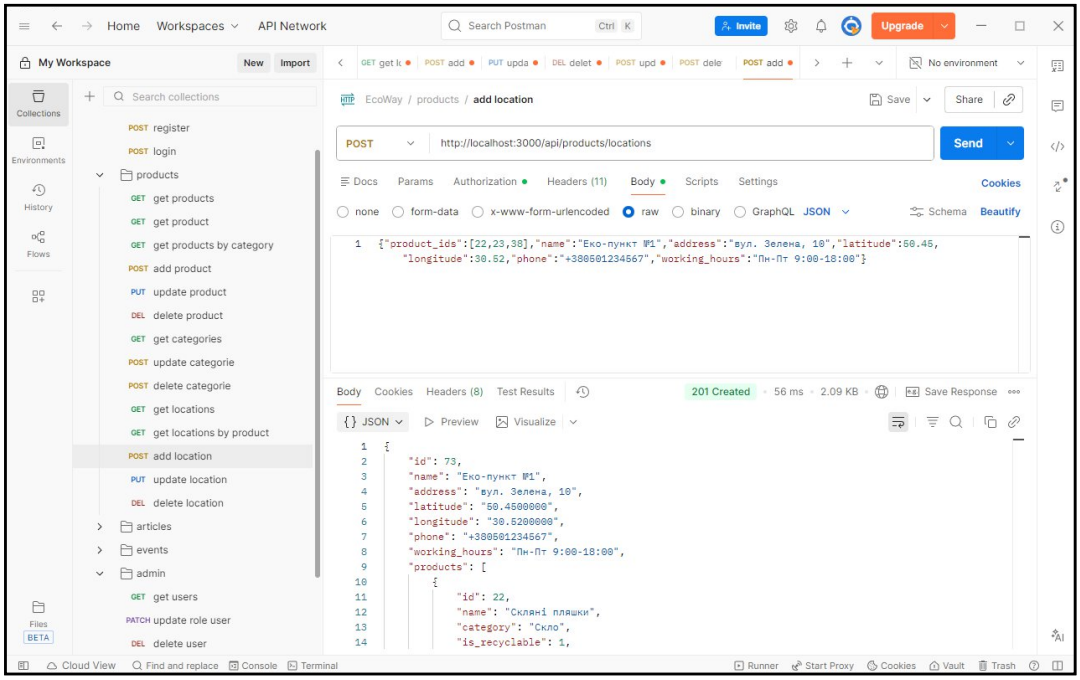

Рисунок 3.10 – Тестування додавання нового пункту прийому вторинної сировини через Postman

На рисунку 3.11 зображено результат тестування отримання списку статей (GET /api/articles) через Postman. Запит надсилається без додаткових параметрів фільтрації. Сервер повернув відповідь із кодом «200 OK», що свідчить про повернення значного масиву даних. У тілі відповіді міститься масив опублікованих статей із повною інформацією про кожну: унікальний ідентифікатор, заголовок та повний текст матеріалу з детальними інструкціями щодо повторного використання пластикових пляшок для створення кашпо для рослин. Результат підтверджує коректну роботу серверного запиту отримання списку статей, які пройшли модерацію та доступні для перегляду користувачами платформи.

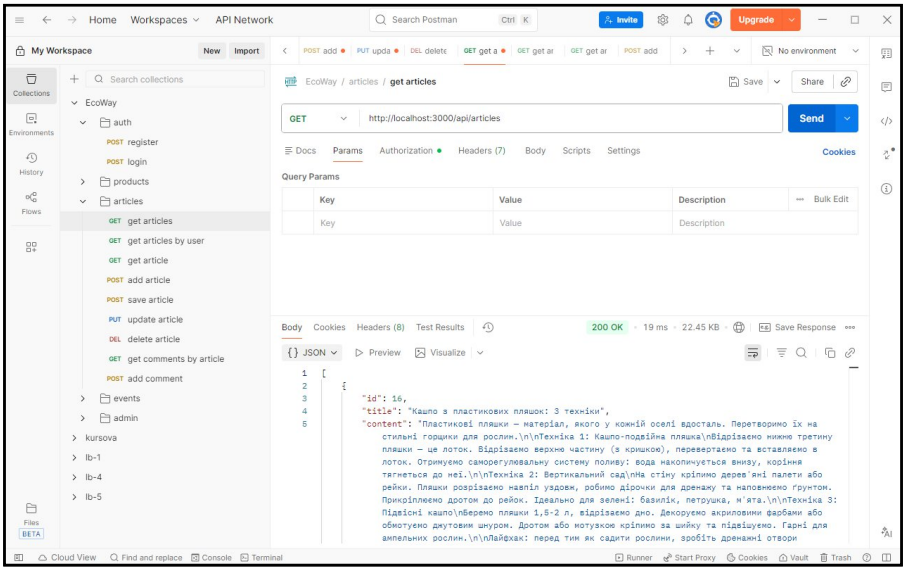

Рисунок 3.11 – Тестування отримання списку статей через Postman

На рисунку 3.12 зображено результат тестування отримання коментарів до конкретної статті (GET /api/articles/:id/comments) через Postman. Ідентифікатор статті передається безпосередньо в URL, запит надсилається без додаткових параметрів. Сервер повернув відповідь із кодом «200 OK». У тілі відповіді міститься масив коментарів до статті з article_id, де для кожного запису відображається унікальний ідентифікатор, ідентифікатор статті, ідентифікатор автора, текст коментаря, дата створення, а також вкладений об'єкт із даними автора (ідентифікатором та іменем користувача).

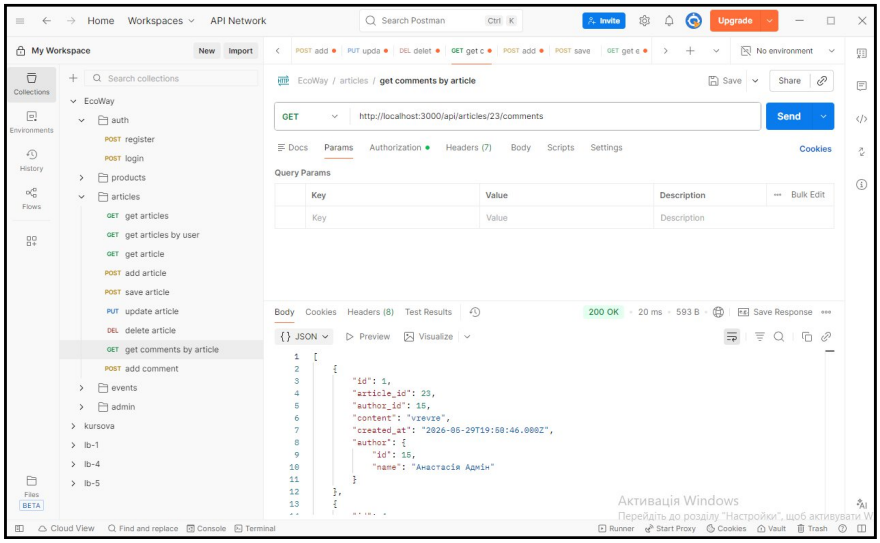

Рисунок 3.12 – Тестування отримання коментарів до статті через Postman

На рисунку 3.13 зображено результат тестування додавання коментаря до статті (POST /api/articles/:id/comments) через Postman. Ідентифікатор статті передається в URL. Сервер повернув відповідь із кодом «201 Created», що підтверджує успішне збереження коментаря.

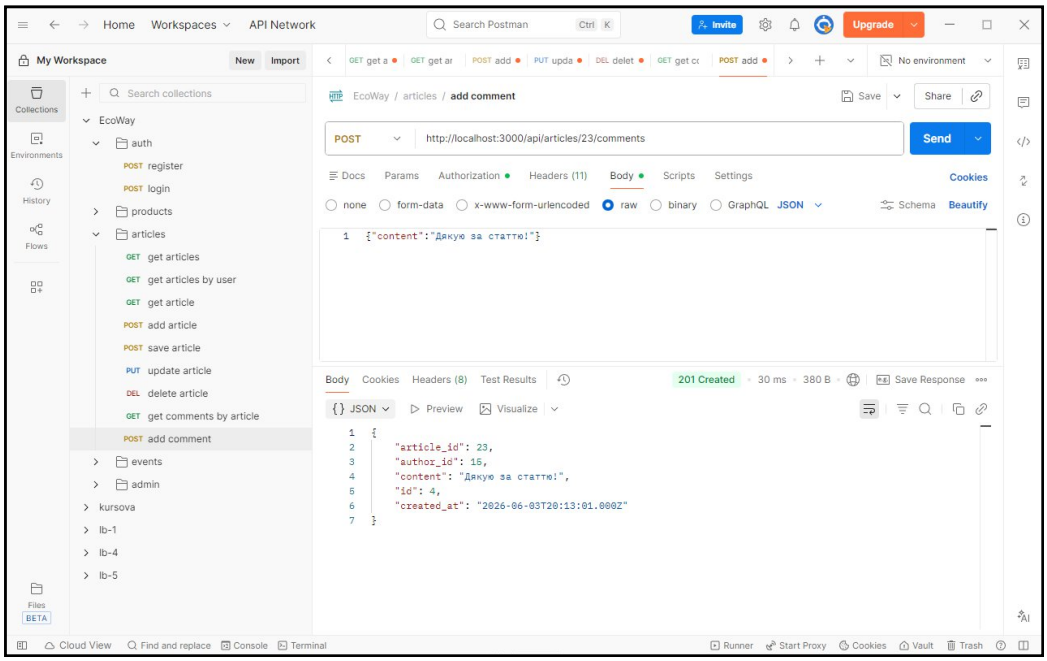

Рисунок 3.13 – Тестування додавання коментаря до статті через Postman

На рисунку 3.14 зображено результат тестування отримання списку екологічних подій (GET /api/events) через Postman. Запит надсилається без додаткових параметрів фільтрації. Сервер повернув відповідь із кодом «200 OK». У тілі відповіді міститься масив подій із повною інформацією про кожну: унікальний ідентифікатор, назва, детальний опис заходу, місце проведення, географічні координати для відображення на інтерактивній карті (latitude та longitude), дати початку та закінчення події, максимальна кількість учасників, посилання на обкладинку та статус події.

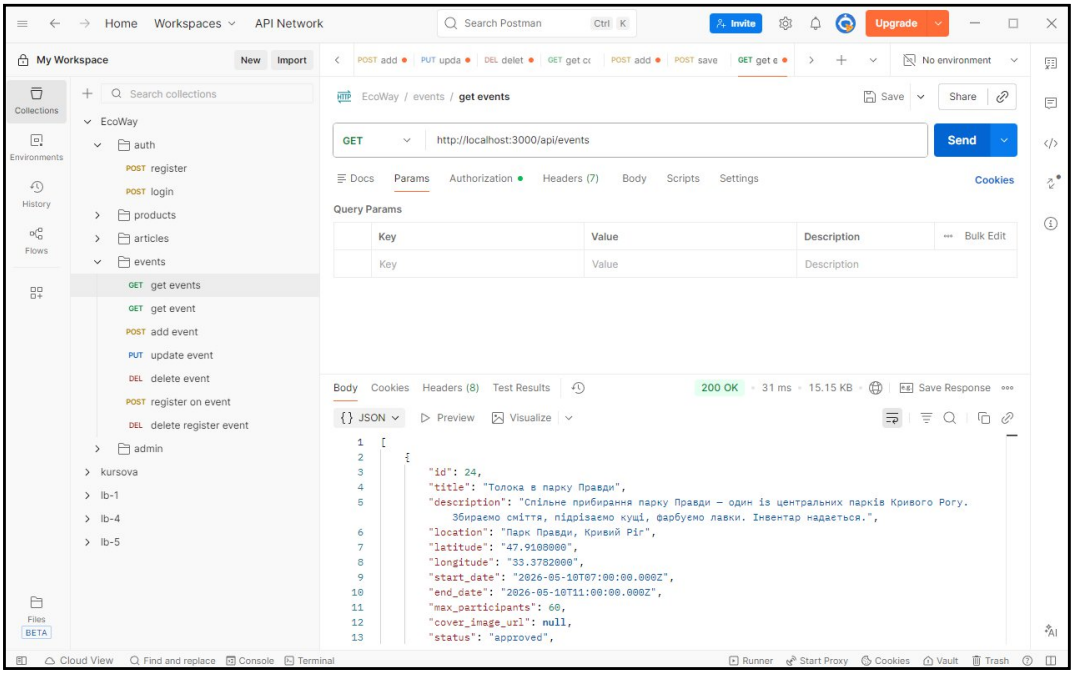

Рисунок 3.14 – Тестування отримання списку екологічних подій через Postman

На рисунку 3.15 зображено результат тестування створення нової екологічної події (POST /api/events) через Postman. У тілі запиту передано JSON-об'єкт із повними даними нової події: назва, опис, місце проведення, географічні координати, дати початку та закінчення, максимальна кількість учасників та статус. Сервер повернув відповідь із кодом «201 Created», що підтверджує успішне створення запису. У тілі відповіді відображаються збережені дані події з автоматично присвоєним унікальним ідентифікатором та ідентифікатором організатора. Особливої уваги заслуговує автоматично встановлений статус «pending», що підтверджує коректну роботу системи – нові події, створені користувачами, потрапляють на перевірку модератора перед публікацією на платформі.

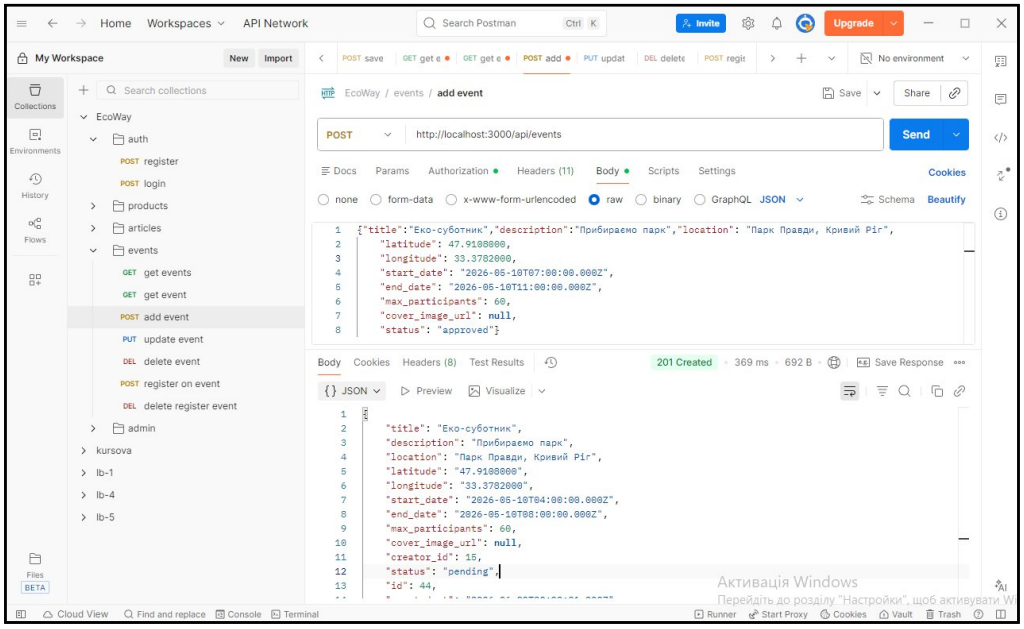

Рисунок 3.15 – Тестування створення нової екологічної події через Postman

На рисунку 3.16 зображено результат тестування реєстрації користувача на подію (POST /api/events/:id/register) через Postman. Ідентифікатор події передається безпосередньо в URL, тіло запиту є порожнім – сервер автоматично визначає користувача на основі JWT-токена у заголовку запиту. Сервер повернув відповідь із кодом «201 Created», що підтверджує успішне створення запису реєстрації. У тілі відповіді міститься підтверджувальне повідомлення: «Реєстрацію підтверджено». Результат підтверджує коректну роботу серверної логіки реєстрації учасників на події, яка доступна лише авторизованим користувачам та автоматично перевіряє наявність вільних місць відповідно до максимальної кількості учасників, визначеної організатором заходу.

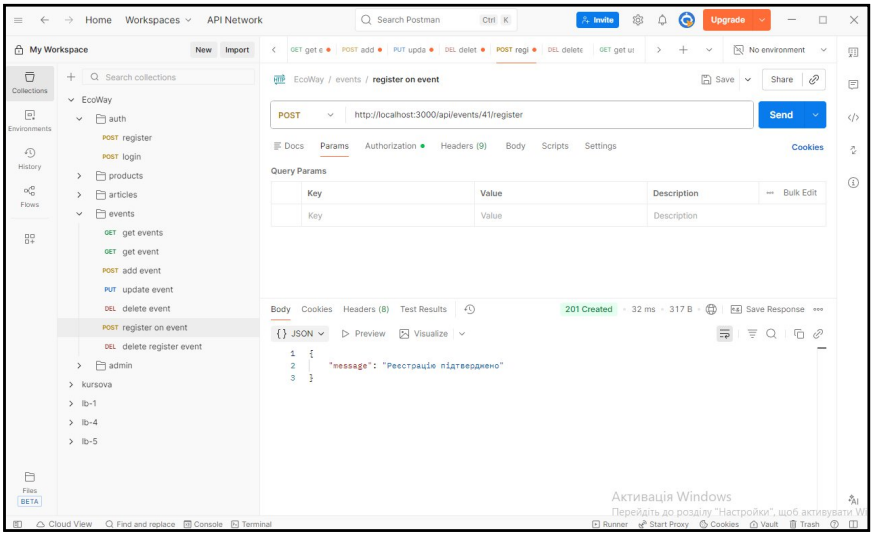

Рисунок 3.16 – Тестування реєстрації користувача на подію через Postman

На рисунку 3.17 зображено результат тестування отримання списку всіх користувачів системи (GET /api/admin/users) через Postman. Запит надсилається з JWT-токеном адміністратора у заголовку, що є обов'язковою умовою доступу до адміністративних маршрутів. Сервер повернув відповідь із кодом «200 OK». У тілі відповіді міститься масив усіх зареєстрованих користувачів із детальною інформацією про кожного. Цей запит використовується адміністративною панеллю для відображення повного переліку користувачів із можливістю управління їх ролями та обліковими записами.

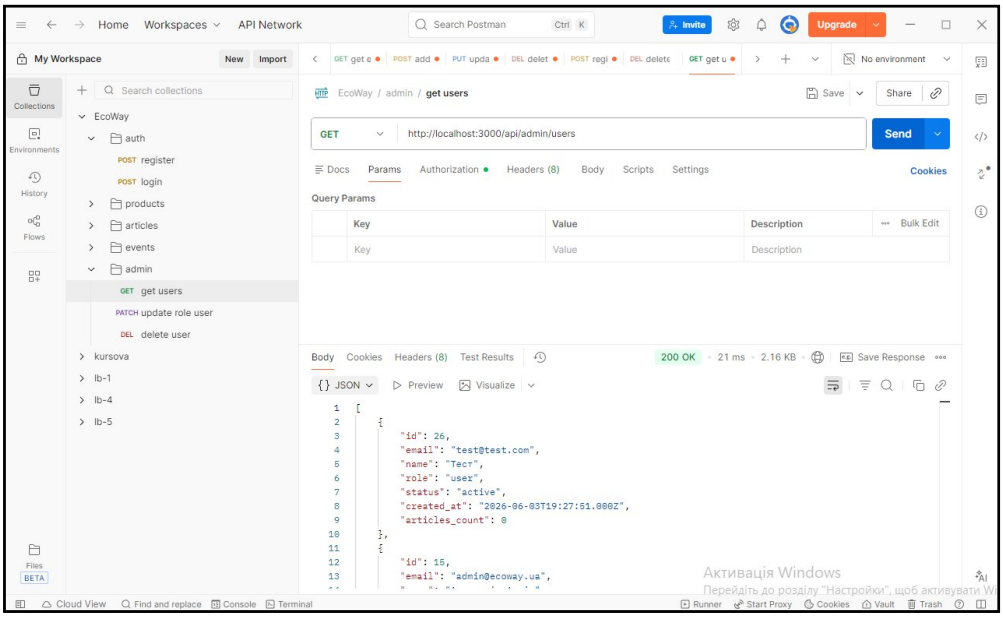

Рисунок 3.17 – Тестування отримання списку користувачів через Postman

На рисунку 3.18 зображено результат тестування оновлення ролі користувача (PATCH /api/admin/users/:id/role) через Postman. У тілі запиту передано JSON-об'єкт із новою роллю «moderator». Ідентифікатор користувача передається безпосередньо в URL. Сервер повернув відповідь із кодом «200 OK», що підтверджує успішне оновлення ролі. У тілі відповіді відображаються актуальні дані користувача: електронна адреса, ім'я та оновлена роль. Результат підтверджує коректну роботу адміністративного запиту зміни ролей користувачів, який доступний виключно адміністраторам системи та дозволяє гнучко керувати рівнями доступу учасників платформи без необхідності повторної реєстрації.

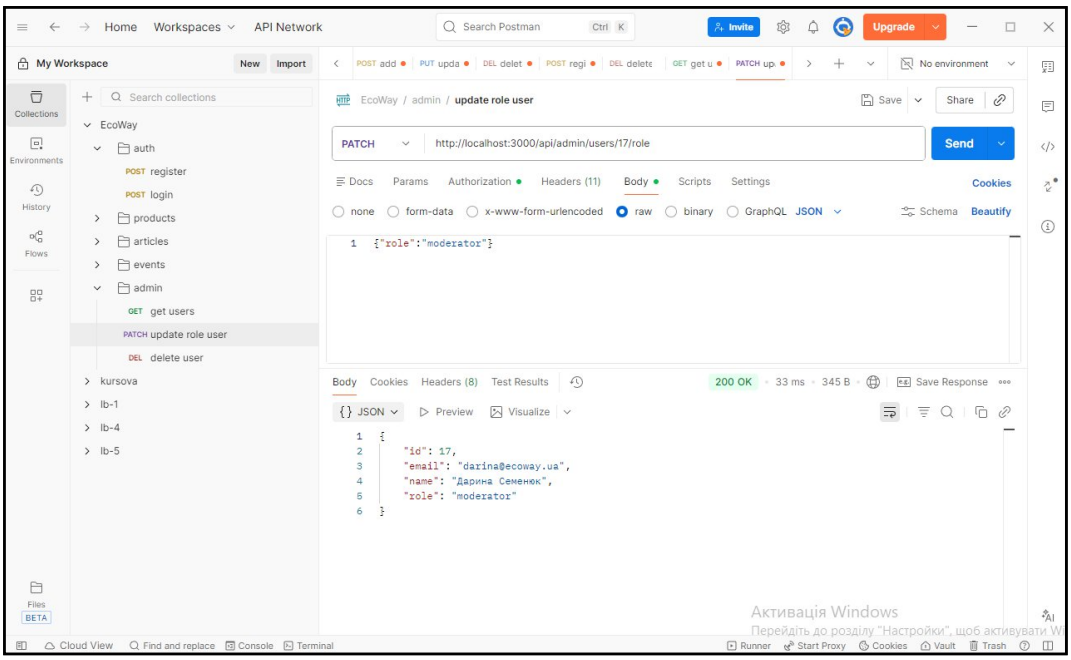

Рисунок 3.18 – Тестування оновлення ролі користувача через Postman

3.6. Тестування клієнтського інтерфейсу

Тестування клієнтського інтерфейсу платформи EcoWay проводилось шляхом послідовної перевірки всіх сторінок та функціональних модулів застосунку для різних категорій користувачів: незареєстрованого відвідувача, авторизованого користувача, модератора та адміністратора. Перевірялась коректність відображення інтерфейсу, робота форм та валідації, функціонування захищених маршрутів, а також відповідність поведінки системи визначеним ролям доступу.

На рисунку 3.19 зображено головну сторінку платформи EcoWay у вигляді, доступному для незареєстрованого відвідувача. Сторінка містить навігаційне меню з посиланнями на розділи «Довідник», «Статті» та «Події», кнопки входу та реєстрації, головний слоган платформи, блок швидкого пошуку в довіднику переробки з фільтрацією за категоріями матеріалів, картку найближчої події з можливістю реєстрації, а також три інформаційні блоки з посиланнями на основні модулі системи. Загальний дизайн виконано в екологічній колірній гамі з використанням відтінків зеленого та бежевого.

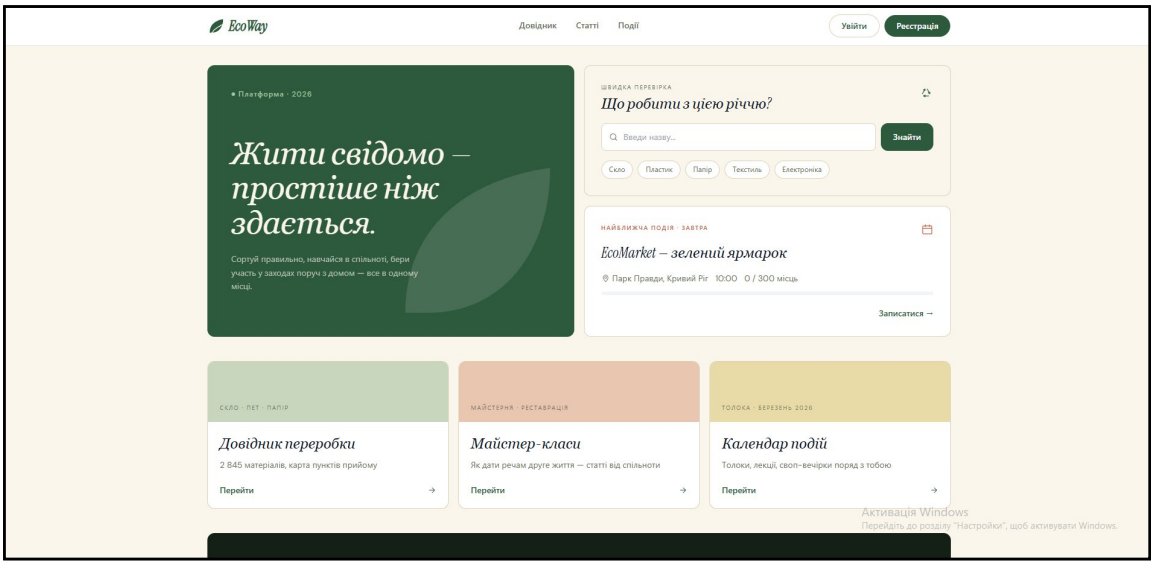

Рисунок 3.19 – Головна сторінка платформи EcoWay

На рисунку 3.20 зображено сторінку довідника переробки платформи EcoWay. Сторінка містить поле пошуку матеріалів за назвою, фільтри за категоріями та інформаційний блок із зазначенням кількості найближчих пунктів прийому. Результати пошуку відображають 16 знайдених матеріалів у вигляді карток із назвою, категорією, міткою «Переробляється» та коротким описом інструкцій з утилізації. У правій частині сторінки розташована інтерактивна карта з маркерами пунктів прийому вторинної сировини на основі Leaflet.js та блок із інформацією про найближчий пункт прийому із зазначенням адреси та режиму роботи.

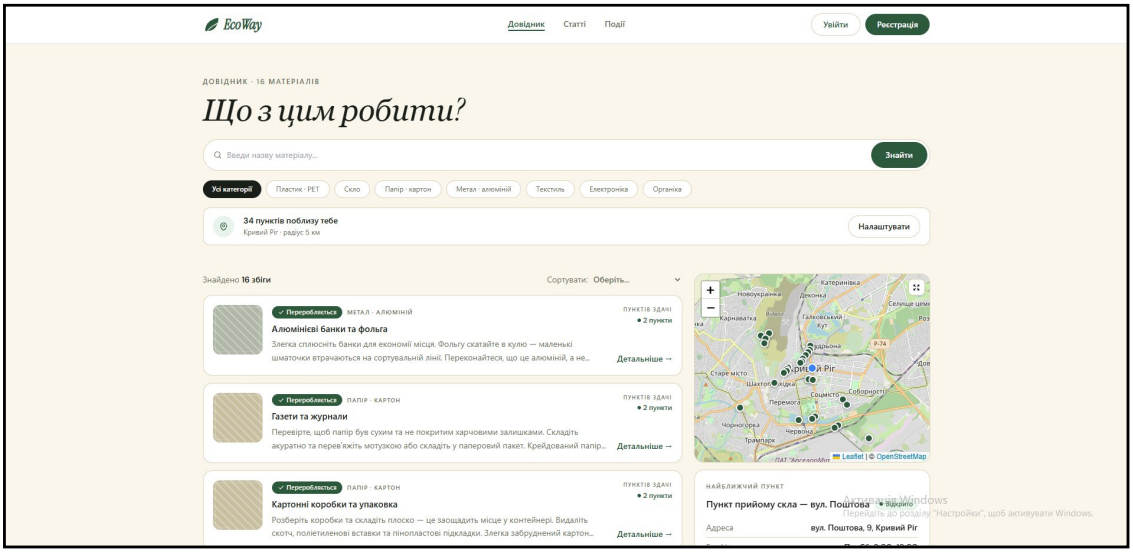

Рисунок 3.20– Сторінка довідника переробки

На рисунку 3.21 зображено сторінку детального перегляду матеріалу в довіднику переробки. Сторінка відображає мітку «Підлягає переробці», категорію матеріалу та код маркування. Основний блок містить детальний опис матеріалу, покрокові інструкції з підготовки до здачі, технічні характеристики, а також два інформаційні блоки з переліком того, що можна здавати та що не приймається. У нижній частині сторінки розташований розділ «Де здати поблизу» зі списком найближчих пунктів прийому та розділ «Схожі матеріали».

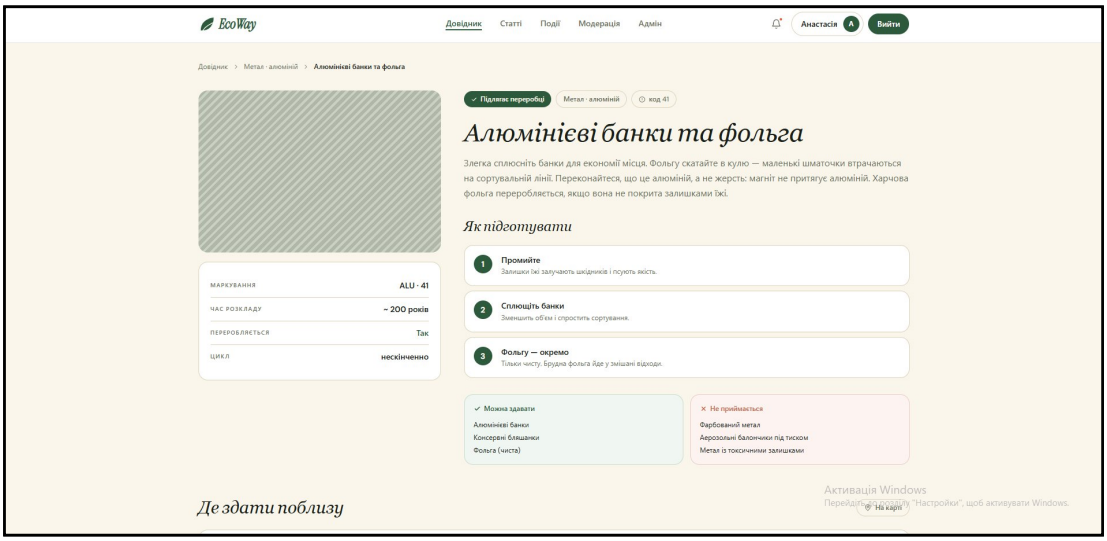

Рисунок 3.21 – Сторінка детального перегляду матеріалу в довіднику переробки

На рисунку 3.22 зображено сторінку переліку статей та майстер-класів платформи EcoWay. Сторінка містить заголовок, лічильник публікацій, фільтри за категоріями матеріалів та сортування за новизною. Головна публікація відображається у вигляді великої картки, заголовком, коротким описом, інформацією про автора, датою публікації та статистикою переглядів і коментарів. У правій частині розташована колонка з іншими статтями у компактному форматі. Нижче знаходяться всі інші майстер-класи.

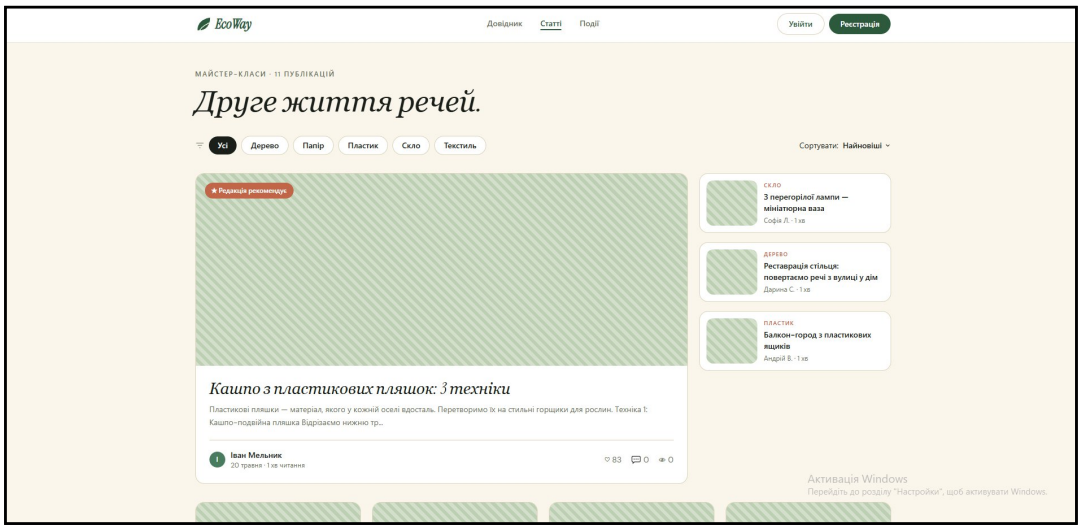

Рисунок 3.22 – Сторінка переліку статей та майстер-класів

На рисунку 3.23 зображено сторінку детального перегляду статті. Верхня частина сторінки містить теги категорії, заголовок, короткий вступний опис, інформацію про автора з позначкою ролі, дату публікації, час читання, кількість переглядів та лайків, а також обкладинку статті й інфографіку з характеристиками майстер-класу: складність, орієнтовний час виконання та необхідна кількість матеріалів. Основний контент статті містить детальні покрокови інструкції, а у правій частині розташоване меню змісту та блок пов'язаних статей. У нижній частині сторінки відображено розділ обговорення з формою для додавання коментаря та наявними відгуками користувачів.

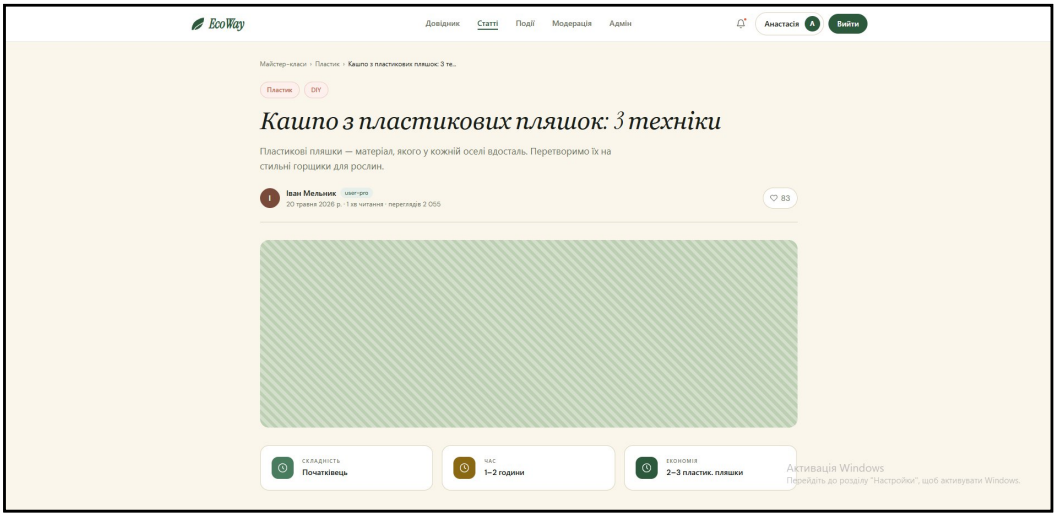

Рисунок 3.23 – Сторінка перегляду статті

На рисунку 3.24 зображено сторінку календаря подій платформи EcoWay. У верхній частині сторінки відображається заголовок, лічильник подій, інформаційний блок із кількістю найближчих заходів поблизу користувача, перемикачі режимів відображення та фільтри за категоріями. Найближча подія виділена окремим великим блоком із датою, часом, місцем проведення, кількістю учасників, організатором та кнопками «Записатися» і «Нагадати». У нижній частині сторінки події згруповані за датами у хронологічному порядку та відображаються у вигляді компактних карток із часом, назвою, місцем проведення, категорією, міткою та кількістю доступних місць, що підтверджує коректну роботу модулю подій та системи реєстрації учасників.

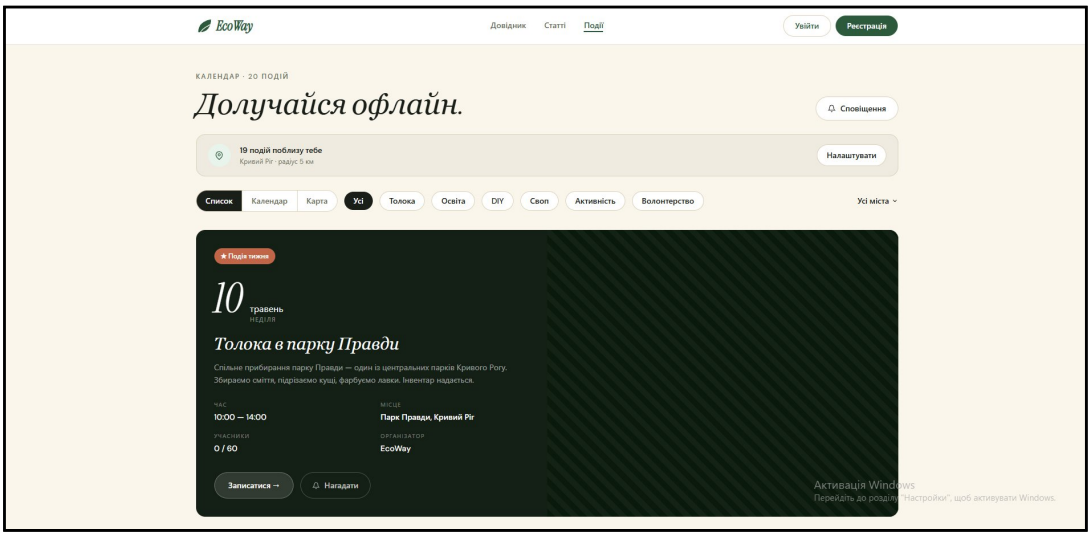

Рисунок 2.24 – Сторінка календаря подій

На рисунку 3.25 зображено сторінку детального перегляду події. У верхній частині сторінки відображаються мітки, заголовок події, короткий опис, ключові характеристики (дата, час, кількість учасників, вартість), а також блок реєстрації з кнопками «Записатися», «Нагадати» та «Поділитися». Нижче розташований розділ «Що буде» з переліком запланованих активностей та блок із картою місця проведення й посиланням для прокладання маршруту. У нижній частині сторінки відображається розділ «Що принести» з чіткою розбивкою на дозволені та недозволені предмети, інформація про організатора та блок схожих подій із найближчими заходами платформи.

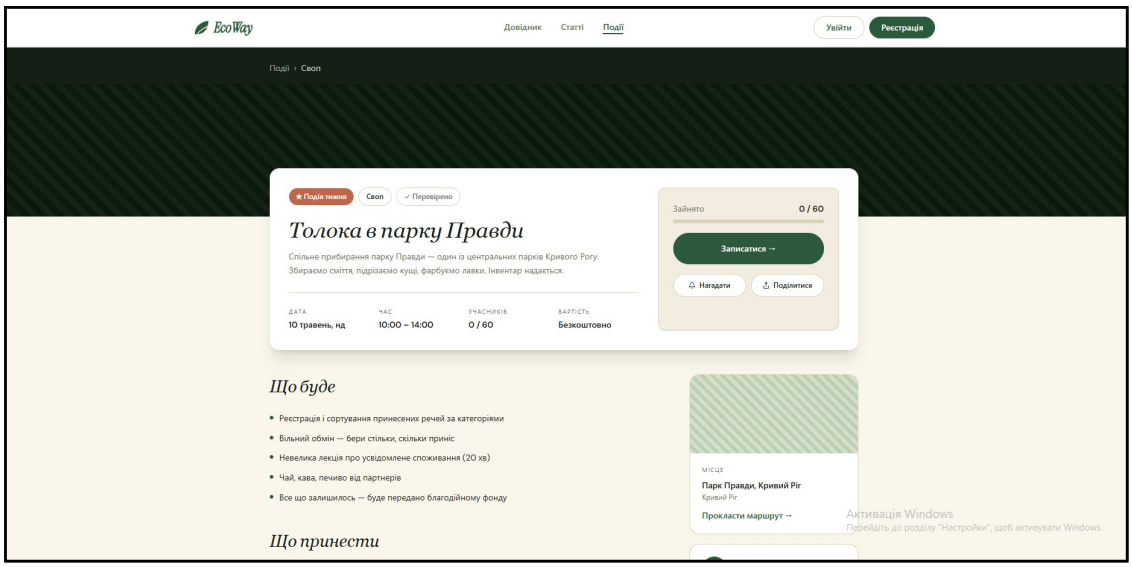

Рисунок 3.25 – Сторінка детального перегляду події

На рисунку 3.26 зображено модальне вікно авторизації, що відкривається при спробі незареєстрованого користувача записатися на подію. Вікно містить перемикач між вкладками «Вхід» та «Реєстрація», заголовок «З поверненням», поля для введення електронної адреси та пароля, кнопку «Увійти» та посилання для переходу до реєстрації нового облікового запису. Таким чином підтверджується коректна робота механізму захисту – система не дозволяє незареєстрованим користувачам виконувати дії, що потребують авторизації, та пропонує зручний спосіб входу без перенаправлення на окрему сторінку.

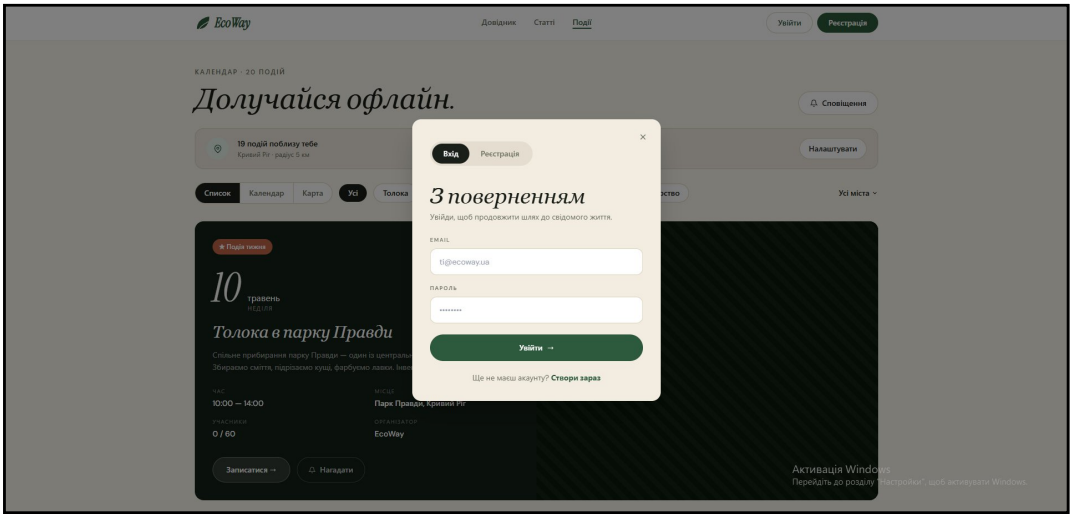

Рисунок 3.26 – Модальне вікно авторизації

На рисунку 3.27 зображено модальне вікно реєстрації нового користувача на платформі EcoWay. Вікно містить перемикач між вкладками «Вхід» та «Реєстрація», заголовок «Створи акаунт» із коротким описом, поля для введення імені, електронної адреси та пароля з підказкою щодо мінімальних вимог (мінімум 8 символів, цифра і літера), чекбокс погодження з правилами користування та політикою конфіденційності, кнопку «Зареєструватись» та посилання для переходу до форми входу для вже зареєстрованих користувачів. Відображення модального вікна поверх головної сторінки підтверджує коректну роботу клієнтської маршрутизації та зручність користувацького досвіду – реєстрація відбувається без перенаправлення на окрему сторінку.

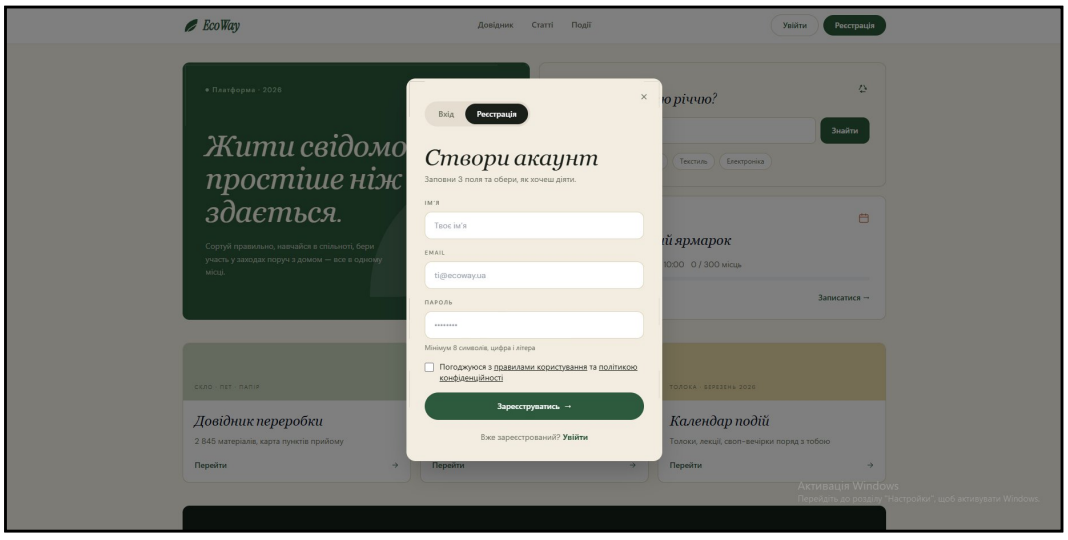

Рисунок 3.27 – Модальне вікно реєстрації нового користувача

На рисунку 3.28 зображено особистий кабінет авторизованого користувача з роллю адміністратора. У лівій частині розташована бічна панель навігації з розділами «Огляд», «Мої публікації», «Мої події», «Лайки» та «Досягнення», а також блок налаштувань із посиланнями на профіль і сповіщення. Центральна частина сторінки містить привітальний блок із датою, рядком статусу користувача та кнопкою створення нової публікації. У нижній частині відображається перелік власних публікацій із вкладками фільтрації за статусом модерації.

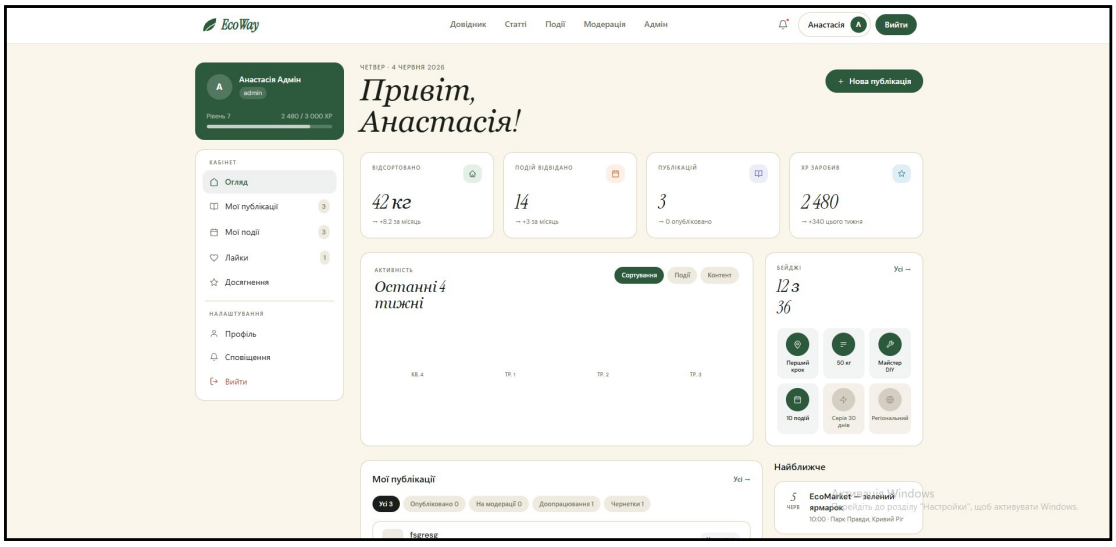

Рисунок 3.28 – Особистий кабінет авторизованого користувача

На рисунку 3.29 зображено сторінку створення нової статті (майстер-класу) на платформі EcoWay, доступну авторизованим користувачам. Сторінка містить область завантаження обкладинки, поля для введення заголовку та короткого вступу, а також повноцінний текстовий редактор із панеллю інструментів форматування (жирний, курсив, підкреслення, заголовки, списки). У правій частині розташована панель налаштувань публікації: вибір категорії, додавання тегів, параметри майстер-класу (складність, час виконання, економія матеріалів) та блок статусу зі вказівкою поточного стану і інформацією про процес модерації. У верхній частині сторінки розташовані кнопки «Зберегти чернетку» та «Надіслати на модерацію».

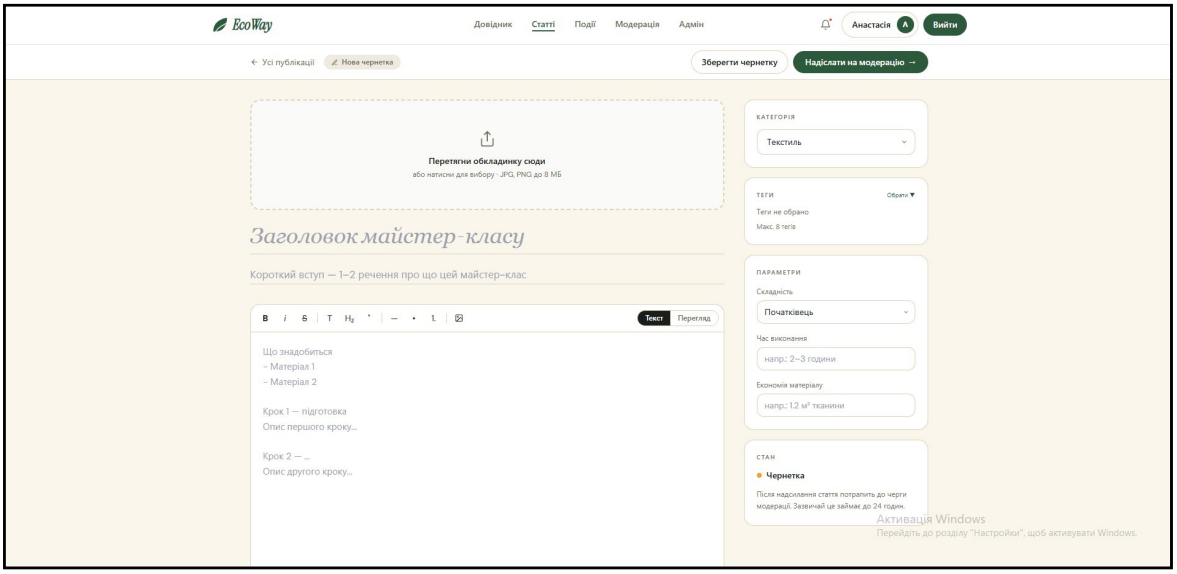

Рисунок 3.29 – Сторінка створення нової статті (майстер-класу)

На рисунку 3.30 зображено сторінку створення нової події на платформі EcoWay, доступну користувачам із роллю user-pro та адміністраторам. Верхня частина форми містить область завантаження обкладинки події, поля для введення назви та короткого опису, вибір категорії («Толока», «Освіта», «DIY», «Своп», «Активність», «Волонтерство»), а також панель попереднього перегляду картки події з блоком налаштувань сповіщень учасникам та інформацією про організатора. Середня частина форми включає блок «Дата та час» із полями вибору дати, часу початку та завершення, блок «Місце» з перемикачем формату проведення («Офлайн», «Онлайн», «Гібрид»), пошуком адреси, а також блок «Учасники» з налаштуванням ліміту місць, вартості участі та опціями списку очікування і підтвердження організатором. У нижній частині форми розташовані блоки «Що буде – програма» для додавання пунктів програми заходу та «Що принести» з окремими полями для переліку дозволених і недозволених предметів.

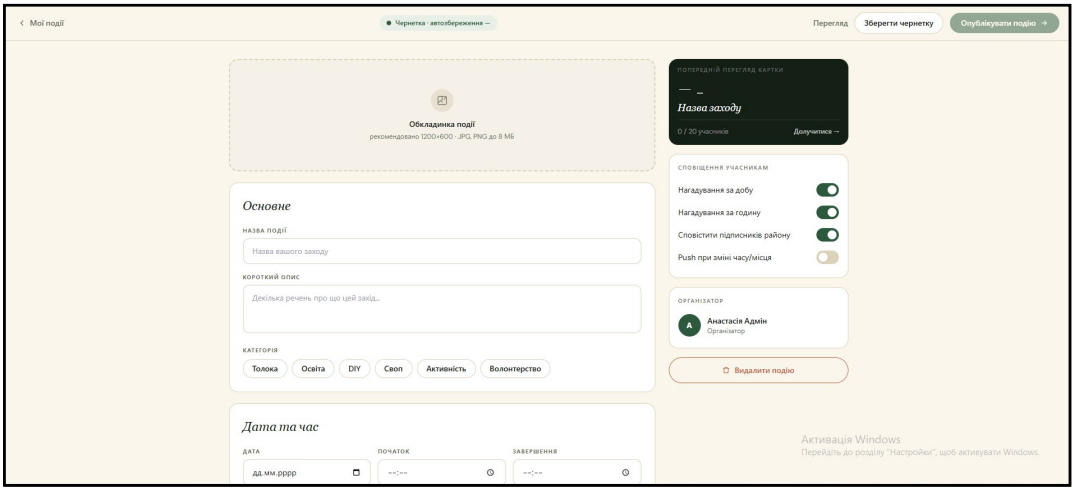

Рисунок 3.30 – Форма створення події

На рисунку 3.31 зображено панель модерації платформи EcoWay, доступну користувачам із роллю модератора або адміністратора. Ліва частина сторінки містить навігаційне меню з двома основними розділами черги модерації статей або подій, а також архівами опублікованих, відхилених, доопрацьованих і видалених матеріалів окремо для статей та подій. У центральній частині відображається черга модерації статей із повідомленням «Немає елементів», що свідчить про відсутність матеріалів, які очікують на перевірку на момент тестування. Права частина містить область перегляду обраного матеріалу із підказкою «Оберіть матеріал для перегляду». Розділ відображається лише для користувачів із відповідними правами, що підтверджується наявністю пункту «Модерація» у навігаційному меню.

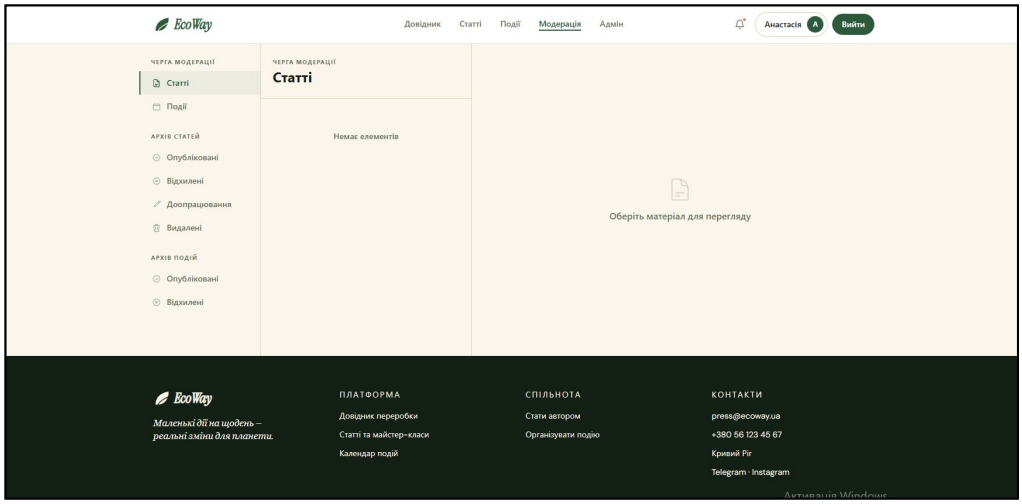

Рисунок 3.31 – Панель модератора

На рисунку 3.32 зображено адміністративну панель управління користувачами платформи EcoWay. У верхній частині відображаються зведені статистичні показники: загальна кількість користувачів, кількість звичайних користувачів, модераторів та заблокованих. Нижче розташований рядок пошуку з фільтрами за роллю та статусом, а також таблиця з переліком усіх зареєстрованих користувачів. Для кожного запису відображається аватар, ім'я, електронна адреса, роль із можливістю зміни через спадне меню, рівень активності (кількість публікацій), дата реєстрації та поточний статус облікового запису. Адміністратор може змінювати роль будь-якого користувача безпосередньо з таблиці, а також видаляти облікові записи.

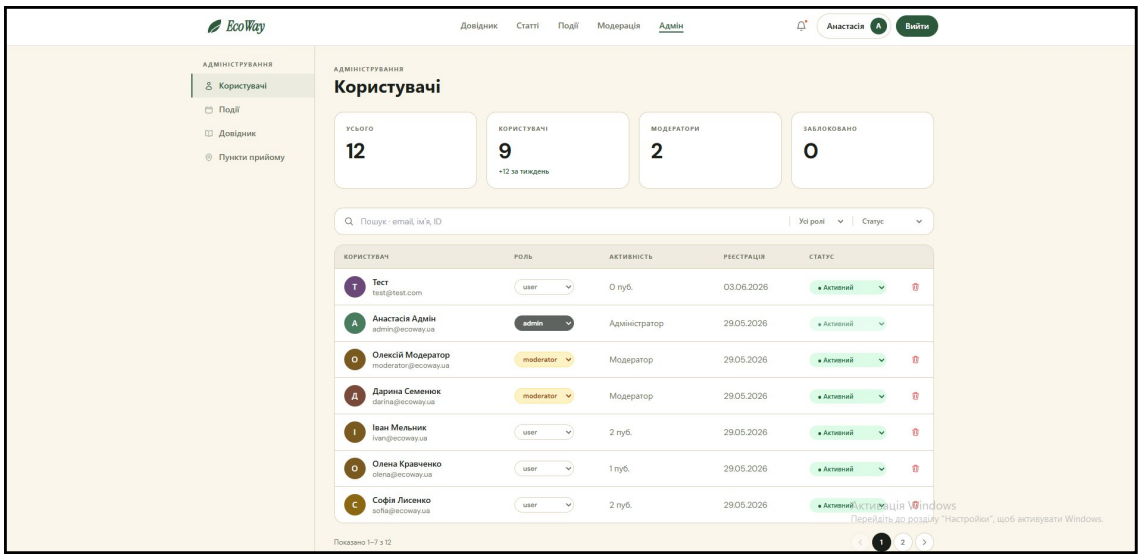

Рисунок 3.32 – Адміністративна панель управління користувачами

Проведене тестування клієнтського інтерфейсу підтвердило коректність роботи всіх реалізованих модулів платформи EcoWay. Інтерфейс коректно адаптується відповідно до ролі авторизованого користувача, захищені маршрути належним чином обмежують доступ до функціоналу, а форми створення контенту та реєстрації виконують клієнтську валідацію введених даних. Усі основні сценарії використання – від перегляду довідника переробки та публікації статей до реєстрації на події та управління користувачами в адміністративній панелі – функціонують відповідно до визначених вимог системи.

ВИСНОВОК

У процесі виконання кваліфікаційної роботи було розроблено повнофункціональний вебзастосунок екологічного спрямування «EcoWay», призначений для управління заходами, поширення освітнього контенту та популяризації культури повторного використання матеріалів і сортування відходів.

Проведено аналіз предметної області та існуючих аналогів – як іноземних, так і українських. Аналіз показав, що більшість існуючих рішень реалізують лише окремі функції та не забезпечують комплексної взаємодії користувачів із екологічним контентом. На основі виявлених недоліків сформовано вимоги до розроблюваної системи.

Спроектовано архітектуру системи – розроблено математичну модель, визначено користувацькі ролі та рівні доступу, створено структурну схему, діаграму прецедентів, функціональну діаграму, діаграму послідовностей та ER-діаграму бази даних. Реляційна модель бази даних охоплює основні сутності системи: користувачів, статті, коментарі, події, реєстрації та довідник переробки.

Реалізовано серверну частину застосунку на основі Node.js та Express.js з використанням TypeScript. Розроблено REST API з повним набором ендпоінтів для всіх функціональних модулів системи. Реалізовано JWT-аутентифікацію з парою access/refresh токенів, систему розмежування доступу на основі ролей, валідацію вхідних даних та захищене зберігання паролів за допомогою алгоритму bcrypt.

Реалізовано клієнтську частину застосунку на основі React з використанням Vite, React Router, Tailwind CSS та Axios. Розроблено інтерфейс для всіх функціональних модулів платформи: довідника переробки з інтерактивною картою пунктів прийому, модуля статей та майстер-класів із системою коментування, календаря екологічних подій, особистого кабінету користувача, панелі модерації та адміністративної панелі управління.

Налаштовано базу даних MySQL із використанням системи міграцій Knex.js для версіонування схеми та seeds для наповнення тестовими даними. Реалізовано повний набір таблиць із відповідними зовнішніми ключами та правилами каскадного видалення.

Проведено тестування серверної частини через Postman та функціональне тестування клієнтського інтерфейсу. Тестування підтвердило коректність роботи всіх реалізованих модулів, правильність повернення HTTP-статус кодів, коректну роботу механізмів аутентифікації та авторизації, а також відповідність інтерфейсу висунутим вимогам.

Розроблений вебзастосунок «EcoWay» відповідає сучасним вимогам до програмних систем і реалізує комплексний підхід до популяризації екологічного способу життя. На відміну від існуючих аналогів, платформа поєднує в єдиній системі довідник переробки, освітній контент із можливістю його створення користувачами, модуль організації екологічних подій та механізми соціальної взаємодії. Практична цінність розробленого застосунку полягає в можливості його використання як інформаційної та комунікаційної платформи для підвищення екологічної обізнаності населення та залучення користувачів до активної екологічної діяльності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Eco-Cycle A-Z Recycling Guide: вебсайт. URL: https://ecocycle.org/guides-and-resources/popular-tools/a-z-recycling-guide/ (дата звернення: 04.05.2026).
2. Freegle: вебсайт. URL: https://ecocycle.org/guides-and-resources/popular-tools/a-z-recycling-guide/ (дата звернення: 04.05.2026).
3. Compare and Recycle: вебсайт. URL: https://www.compareandrecycle.co.uk/ (дата звернення: 04.05.2026).
4. Recycle Track Systems: вебсайт. URL: https://www.rts.com/ (дата звернення: 04.05.2026).
5. Horizon: вебсайт. URL: https://www.horizon.eco/ (дата звернення: 04.05.2026).
6. RecycleBank: вебсайт. URL: https://recyclebank.com/ (дата звернення: 04.05.2026).
7. Reddit – аналіз користувачів: вебсайт. URL: https://www.reddit.com (дата звернення: 04.05.2026).
8. Україна без сміття: вебсайт. URL: https://nowaste.com.ua/ (дата звернення: 04.05.2026).
9. Ecola: вебсайт. URL: https://www.ecolaglobal.com/ (дата звернення: 04.05.2026).
10. Let's Do It Ukraine: вебсайт. URL: https://letsdoitukraine.org/ (дата звернення: 04.05.2026).
11. Діаграма прецедентів: теорія та практика. DOU: вебсайт. URL: https://dou.ua/forums/topic/40575/ (дата звернення: 04.05.2026).
12. Методологія функціонального моделювання IDEF0. URL: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9A%D0%BE%D0%BD%D0%B4%D1%96%D1%83%D1%81%202%20%D0%B3%D0%BE%D1%82%D0%BE%D0%B2%D0%B2%D0%B0/page9.html (дата звернення: 04.05.2026).

13. Діаграми послідовностей (Sequence Diagrams). URL: https://www.maxzosim.com/sequence-diagrams/ (дата звернення: 04.05.2026).

14. ER Diagram Tutorial in DBMS. Guru99: вебсайт. URL: https://www.guru99.com/uk/er-diagram-tutorial-dbms.html (дата звернення: 04.05.2026).

15. Cherny, B. Programming TypeScript: Making Your JavaScript Applications Scale. – Sebastopol: O'Reilly Media, 2023. – 352 p.

16. Node.js – про платформу. URL: https://nodejs.org/uk/about (дата звернення: 04.05.2026).

17. Express.js: документація. URL: https://expressjs.com/en/ (дата звернення: 04.05.2026).

18. Knex.js: документація. URL: https://knexjs.org/ (дата звернення: 04.05.2026).

19. Objection.js: документація. URL: https://vincit.github.io/objection.js/ (дата звернення: 04.05.2026).

20. Wieruch, R. The Road to React: Your journey to master plain yet pragmatic React.js. – Independently published, 2022. – 394 p.

21. Leaflet.js: документація. URL: https://leafletjs.com/reference.html (дата звернення: 04.05.2026).

22. Postman API Platform : вебсайт. URL: https://www.postman.com (дата звернення: 04.05.2026).

ДОДАТОК А

Ієрархічний контроль доступу

```
const roleHierarchy: Record<UserRole, number> = {
  guest: 0, user: 1, 'user-pro': 1.5, moderator: 2, admin: 3,
};

export function requireRole(minRole: UserRole) {
  return (req: Request, res: Response, next: NextFunction): void
=> {
    if (!req.user) { res.status(401).json(...); return; }
    if (roleHierarchy[req.user.role] < roleHierarchy[minRole]) {
      res.status(403).json(...); return;
    }
    next();
  };
}
```

ДОДАТОК Б

Автоматичне оновлення JWT з чергою запитів

```
let isRefreshing = false;
let failedQueue: Array<{ resolve: (token: string) => void;
reject: (err: unknown) => void }> = [];

if (isRefreshing) {
  return new Promise((resolve, reject) => {
    failedQueue.push({ resolve, reject });
  }).then(token => {
    originalRequest.headers.Authorization = `Bearer ${token}`;
    return api(originalRequest);
  });
}
processQueue(null, data.accessToken);
```

ДОДАТОК В

React Context зі станом авторизації

```
const hasRole = useCallback(
  (minRole: UserRole) => {
    if (!user) return false;
    return roleHierarchy[user.role] >= roleHierarchy[minRole];
  }, [user]
);

const value = useMemo(
  () => ({ user, isLoading, login, register, logout, hasRole,
updateUser }),
  [user, isLoading, login, register, logout, hasRole,
updateUser]
);
```

ДОДАТОК Г

Генерація JWT пари токенів + bcrypt реєстрація

```
function generateTokens(userId: number, role: string):
AuthTokens {
  const accessToken = jwt.sign(payload, env.JWT_ACCESS_SECRET,
{ expiresIn: ... });
  const refreshToken = jwt.sign(payload, env.JWT_REFRESH_SECRET,
{ expiresIn: ... });
  return { accessToken, refreshToken };
}
const password_hash = await bcrypt.hash(password, 12);
const user = await User.query().insertAndFetch({ email,
password_hash, name, role: 'user' });
```

ДОДАТОК Д

Безпечне оновлення профілю

```
const patch: Record<string, unknown> = {};
if (name?.trim()) patch.name = name.trim();
if (email?.trim() && email.trim() !== user.email) {
  const taken = await User.query().findOne({ email:
email.trim() });
  if (taken) { res.status(409).json({ message: 'Email вже
використовується' }); return; }
  patch.email = email.trim();
}
if (newPassword) {
  const valid = await bcrypt.compare(currentPassword,
user.password_hash);
  if (!valid) { res.status(400).json(...); return; }
  patch.password_hash = await bcrypt.hash(newPassword, 12);
}
if (Object.keys(patch).length === 0) { res.json(currentUser);
return; }
const updated = await User.query().patchAndFetchById(userId,
patch);
```

ДОДАТОК Е

Патерн sentinel–дати для чернеток

```
const SENTINEL_DATE = '2099-12-31 00:00:00';

start_date: start_date ? toMysqlDatetime(start_date) :
SENTINEL_DATE,
end_date:   end_date   ? toMysqlDatetime(end_date)   :
SENTINEL_DATE,

const isSentinel = (iso: string) => new
Date(iso).getFullYear() >= 2090;
```

ДОДАТОК Ж

Часткове оновлення через conditional spread

```
const updated = await Event.query().patchAndFetchById(eventId, {
  ...(title       !== undefined && { title }),
  ...(description !== undefined && { description }),
  ...(start_date  !== undefined && { start_date: start_date ?
toMysqlDatetime(start_date) : SENTINEL_DATE }),
  ...(end_date    !== undefined && { end_date:   end_date   ?
toMysqlDatetime(end_date)   : SENTINEL_DATE }),
  status: newStatus,
});
```

ДОДАТОК И

Список подій з персоналізованим статусом реєстрації

```
const events = await Event.query()
  .count('event_registrations.id as registrations_count')
  .leftJoin('event_registrations', 'events.id',
'event_registrations.event_id')
  .where('events.status', 'approved')
  .groupBy('events.id');

const mySet = new Set(myRegs.map(r => r.event_id));
const result = events.map(ev => ({ ...ev, is_registered:
mySet.has(ev.id) }));
```

ДОДАТОК К

Захищений маршут з автовідкриттям модалки

```
useEffect(() => {
  if (!isLoading && !user) open('login');
}, [isLoading, user, open]);

if (hierarchy[user.role] < hierarchy[role]) return <Navigate
to="/" replace />;
return <Outlet />;
```

ДОДАТОК Л

Типізований API-клієнт

```
type CreateEventData = Omit<Event, 'id' | 'created_at' |
'status' | 'registrations_count' | ...>;

export const eventsApi = {
  create:      (data: CreateEventData) =>
api.post<Event>('/events', data).then(r => r.data),
  updateDraft: (id: number, data: Partial<CreateEventData> &
{ status?: EventStatus }) =>
```

```
                  api.put<Event>(`/events/${id}/draft`,
data).then(r => r.data),
  register:    (id: number) =>
api.post(`/events/${id}/register`).then(r => r.data),
};
```