МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інтелектуальних інформаційних систем

_____Юрій КОНДРАТЕНКО

«____»_____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

ІНФОРМАЦІЙНА СИСТЕМА МОНІТОРИНГУ ТА ОЦІНКИ ЕФЕКТИВНОСТІ ДЕРЖАВНИХ ЦІЛЬОВИХ ПРОГРАМ У СФЕРІ ЕНЕРГОЕФЕКТИВНОСТІ

Спеціальність 122 Комп'ютерні науки

Освітня програма «Комп'ютерні науки»

Здобувач _____ Костянтин КОРНІЄНКО

«___»_____ 2025 р.

Керівник д-р фіз.-мат. наук, проф. _____ Едуард ЛИСЕНКОВ

«___»_____ 2025 р.

Миколаїв – 2025

Чорноморський національний університет імені Петра Могили

(повне найменування закладу вищої освіти)

Факультет	Комп’ютерних наук
Кафедра	Інтелектуальних інформаційних систем
Рівень вищої освіти	Перший (бакалаврський)
Освітній ступень	Бакалавр
Спеціальність	122 Комп’ютерні науки
Освітня програма	Комп’ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем

_____ Юрій КОНДРАТЕНКО

«____» _____ 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу здобувача

Корнієнка Костянтина Анатолійовича

(прізвище, ім’я, по батькові здобувача)

1. Тема кваліфікаційної роботи: «Створення інформаційної системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності».
Керівник роботи: Лисенков Едуард Анатолійович, професор кафедри ФМ, д-р фіз.-мат. наук, проф.
Затверджена наказом ЧНУ ім. Петра Могили від «18» листопада 2024 р. № 321.
2. Строк представлення кваліфікаційної роботи «___» _____ 2025 р.
3. Очікуваний результат роботи та початкові дані, якщо такі потрібні: Інформаційна система моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, яка дозволить автоматизувати процеси моніторингу, підвищити прозорість виконання державних програм та забезпечити

обґрунтоване прийняття управлінських рішень на основі актуальних даних про ефективність впроваджених енергоефективних заходів; розробка системи з використанням сучасних інформаційних технологій, що включатиме модулі внесення даних про виконання програм, аналітичний блок для обробки інформації, підсистему оцінки ефективності на основі ключових показників результативності та веб-інтерфейс для взаємодії з користувачами.

4. Перелік питань, що підлягають розробці: Результатом роботи стане повнофункціональна інформаційна система, яка дозволить автоматизувати процеси моніторингу, підвищити прозорість виконання державних програм та забезпечити обґрунтоване прийняття управлінських рішень на основі актуальних даних про ефективність впроваджених енергоефективних заходів.

5. Перелік графічних матеріалів: презентація.

Керівник роботи _____ Едуард ЛИСЕНКОВ
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Костянтин КОРНІЄНКО
(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата видачі завдання «18» грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

кваліфікаційної роботи

Тема: Інформаційна система моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності

№	Найменування роботи	Початок	Закінчення	Примітки
1	Отримання завдання на виконання КР.	18.12.2024	20.12.2024	
2	Аналіз предметної області та постановка задачі. Аналіз існуючих систем моніторингу державних програм.	21.12.2024	05.01.2025	
3	Огляд літературних джерел за темою кваліфікаційної роботи, зокрема огляд систем моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності.	10.01.2025	15.01.2025	
4	Дослідження методів оцінки ефективності програм у сфері енергоефективності.	16.01.2025	21.01.2025	
5	Формування вимог до інформаційної системи, проектування модулів системи.	22.01.2025	1.02.2025	
6	Розробка архітектури системи, вибір технологій розробки.	02.02.2025	10.02.2025	
7	Реалізація обраних технологій з аналізом отриманих результатів	11.02.2025	15.05.2025	

8	Перший попередній захист КР на засіданні комісії кафедри.	16.05.2025	16.05.2025	
9	Корегування роботи за результатами попереднього захисту.	29.05.2025	29.05.2025	
10	Другий попередній захист КР на засіданні комісії кафедри.	30.05.2025	30.05.2025	
11	Доробка та остаточне оформлення КР.	31.05.2025	10.05.2025	
12	Подання КР, її електронної копії та інших документів (відгуку, рецензії) до захисту.	11.05.2025	17.05.2025	

Керівник роботи _____ Едуард ЛИСЕНКОВ

(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Здобувач _____ Костянтин КОРНІЄНКО

(Особистий підпис) *(Власне ім'я ПРІЗВИЩЕ)*

Дата складання календарного плану

«20» грудня 2024 р.

АНОТАЦІЯ

до кваліфікаційної роботи здобувача
групи 401з ЧНУ ім. Петра Могили
Корнієнка Костянтина Анатолійовича

Тема: Інформаційна система моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності

Актуальність роботи зумовлена необхідністю автоматизації процесів моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, відсутністю єдиної інформаційної системи для відстеження впровадження систем енергетичного менеджменту в організаціях, що підпадають під дію постанови Кабінету Міністрів України від 23 грудня 2021 р. № 1460 "Про впровадження систем енергетичного менеджменту".

Об'єктом роботи є процеси моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, що реалізуються державними органами та підприємствами відповідно до постанови Кабінету Міністрів України від 23 грудня 2021 р. № 1460 "Про впровадження систем енергетичного менеджменту".

Предметом роботи є методи та інформаційні технології для автоматизації моніторингу та оцінки ефективності програм енергоефективності.

Метою роботи є розробка інформаційної системи для автоматизованого моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, яка забезпечить повний цикл моніторингу від збору первинних даних до формування аналітичних звітів та оцінки ефективності впроваджених заходів.

В результаті виконання роботи проведено аналіз існуючих систем моніторингу та методів оцінки ефективності, розроблено архітектуру системи з модульною структурою, спроектовано базу даних PostgreSQL, реалізовано веб-

додаток на Django з використанням Bootstrap, створено алгоритми аналізу та оцінки ефективності енергозберігаючих заходів, а також проведено комплексне тестування системи.

Дана робота складається з чотирьох розділів. У першому розділі проведено аналіз предметної області та постановку задачі дослідження. Другий розділ присвячений проектуванню архітектури системи, бази даних та основних модулів. У третьому розділі описано реалізацію інформаційної системи з використанням сучасних технологій. Четвертий розділ містить результати тестування та впровадження системи. Загальний обсяг роботи – 116 сторінок. Кваліфікаційна робота містить 1 додаток, 23 рисунки, 4 таблиці і 36 джерел посилань.

Ключові слова: інформаційна система, моніторинг, енергоефективність, державні програми, система енергетичного менеджменту, місцеві енергетичні плани.

ABSTRACT

to the qualification work of the applicant
of group 401z of Petro Mohyla Black Sea National University

Korniienko Kostiantyn Anatoliyovych

Topic: "Information system for monitoring and evaluating the effectiveness of state-targeted programs in the field of energy efficiency"

The relevance of the work is due to the need for automation of monitoring and evaluation processes for the effectiveness of state target programs in the field of energy efficiency, and the lack of a unified information system for tracking the implementation of energy management systems in organizations that fall under the Cabinet of Ministers of Ukraine resolution of December 23, 2021, No. 1460 "On the implementation of energy management systems."

The subject of the research is the processes of monitoring and evaluating the effectiveness of state targeted programs in the field of energy efficiency, implemented by government bodies and enterprises in accordance with the Resolution of the Cabinet of Ministers of Ukraine dated December 23, 2021, No. 1460 "On the Implementation of Energy Management Systems."

The subject of the research is methods and information technologies for automating the monitoring and evaluation of the effectiveness of energy efficiency programs.

The purpose of the work is to develop an information system for automated monitoring and evaluation of the effectiveness of state targeted programs in the field of energy efficiency, which will ensure a complete monitoring cycle from the collection of primary data to the formation of analytical reports and assessment of the effectiveness of implemented measures.

This work consists of four sections. The first section conducts an analysis of the subject area and defines the research problem. The second section is dedicated to the

design of the system architecture, database, and main modules. The third section describes the implementation of the information system using modern technologies. The fourth section contains the results of testing and implementation of the system. The total volume of the work is 116 pages. The qualification paper includes 1 appendix, 23 figures, 4 tables, and 36 references.

Keywords: information system, monitoring, energy efficiency, state programs, energy management system, local energy plans.

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ 3
ВСТУП 4
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ 6
1.1 Аналіз існуючих систем моніторингу державних програм 6
1.2 Аналіз вітчизняних систем моніторингу 6
1.3 Аналіз зарубіжних систем моніторингу 7
1.4 Дослідження методів оцінки ефективності програм у сфері енергоефективності 11
1.5 Формування вимог до інформаційної системи 15
1.6 Постановка задачі дослідження 17
2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 20
2.1 Розробка архітектури системи 20
2.2 Проектування бази даних 25
2.3 Проектування системи 32
2.4 Проектування користувацького інтерфейсу 39
3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 49
3.1 Розробка інформаційної системи 49
3.2 Реалізація серверної частини системи 50
3.3 Реалізація клієнтської частини системи 52
3.4 Реалізація алгоритмів аналізу та оцінки ефективності 61
3.5 Інтеграція модулів системи 62
4 ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ СИСТЕМИ 64
4.1 Розробка плану тестування 64
4.2. Проведення функціонального тестування 65
ВИСНОВКИ 68
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 73
ДОДАТОК А Код, який використовується у проекті (python) 77

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ДАЕЕ	Державне агентство з енергоефективності та енергозбереження України
КМУ	Кабінет Міністрів України
СЕнМ	Система енергетичного менеджменту
МЕП	Місцевий енергетичний план
KPI	Ключові показники ефективності
CO_2	Парникові гази
Бенчмаркінг	Порівняльний аналіз об'єктів

ВСТУП

В умовах зростаючих енергетичних викликів, з якими стикається Україна, особливого значення набуває підвищення енергетичної ефективності в усіх сферах життєдіяльності. Державні цільові програми у сфері енергоефективності є важливим інструментом для системної реалізації заходів з енергозбереження та оптимізації використання енергоресурсів на різних рівнях – від загальнодержавного до місцевого.

Однак ефективна реалізація таких програм потребує постійного моніторингу, аналізу та оцінки досягнутих результатів. Існуючі підходи до моніторингу часто характеризуються фрагментарністю, відсутністю єдиних стандартів обробки даних та недостатньою автоматизацією процесів. Це створює перешкоди для об'єктивної оцінки ефективності впроваджених заходів та ускладнює прийняття обґрунтованих управлінських рішень.

Розробка спеціалізованої інформаційної системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності дозволить вирішити ці проблеми шляхом створення єдиної платформи для збору, аналізу та візуалізації даних про впровадження систем енергетичного менеджменту, місцевих енергетичних планів та інших програм підвищення енергоефективності.

Мета роботи: розробка інформаційної системи для автоматизованого моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, яка забезпечить повний цикл моніторингу – від збору первинних даних до формування аналітичних звітів та оцінки ефективності впроваджених заходів.

Завдання дослідження:

– аналіз існуючих підходів до моніторингу державних програм та методів оцінки їх ефективності у сфері енергоефективності;

– формування вимог до інформаційної системи моніторингу та оцінки ефективності;

– проектування архітектури системи та її основних компонентів;

– розробка алгоритмів збору, обробки та аналізу даних про впровадження програм енергоефективності;

– реалізація функціональних модулів системи та їх інтеграція;

– тестування та оцінка ефективності розробленої системи.

Об'єкт дослідження: процеси моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, що реалізуються державними органами та підприємствами відповідно до постанови Кабінету Міністрів України від 23 грудня 2021 р. № 1460 "Про впровадження систем енергетичного менеджменту" [1].

Предмет дослідження: методи та інформаційні технології для автоматизації моніторингу та оцінки ефективності програм енергоефективності.

Практична значимість роботи полягає у створенні інструменту, який дозволить підвищити ефективність управління державними програмами у сфері енергоефективності, забезпечити прозорість їх реалізації та сприяти обґрунтованому прийняттю рішень на основі актуальних даних.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз існуючих систем моніторингу державних програм

Моніторинг державних програм є критично важливим інструментом для забезпечення їх ефективної реалізації та досягнення поставлених цілей. В Україні та світі існує декілька підходів до організації такого моніторингу, особливо у сфері енергоефективності, де точність даних та оперативність реагування безпосередньо впливають на економічні результати [2].

1.2 Аналіз вітчизняних систем моніторингу

На даний момент в Україні функціонує кілька розрізнених систем моніторингу енергоефективності, які не утворюють єдиного інформаційного простору [3].

1.2.1 Системи Державного агентства з енергоефективності та енергозбереження України

ДАЕЕ [4] координує впровадження систем енергетичного менеджменту відповідно до постанови КМУ №1460. Станом на 2024 рік [5], за даними агентства:

- 76 органів державної влади розпочали впровадження СЕнМ;
- призначено 340 відповідальних осіб за енергоменеджмент;
- лише 20% будівель державних установ охоплено системами енергомоніторингу;
- моніторинг здійснюється через заповнення форм у форматі Excel на основі Додатку 1 до Листа про впровадження СЕнМ.

Основними недоліками поточної системи є:

- ручний режим збору та обробки даних;

- відсутність автоматизованої верифікації інформації;
- неможливість оперативного аналізу в режимі реального часу;
- складність агрегації даних на національному рівні [6].

1.2.2 Національна база даних енергетичних характеристик будівель

Розпорядженням КМУ №1254 від 2024 року запроваджено створення національної бази даних [7], яка має охопити:

- 24 обласні державні адміністрації;
- 13,500 об'єктів у 842 населених пунктах;
- понад 700 тисяч технічних показників.

Встановлені терміни наповнення бази [7]:

- до 30 квітня 2025 року - основна інформація про будівлі;
- до 31 липня 2025 року - технічні характеристики та показники енергоспоживання;
- до 31 жовтня 2025 року - дані про заходи з енергоефективності.

1.2.3 АІС "ЕНЕРГОСЕРВІС: облік, контроль, економія"

Система розроблена компанією BisSoft та впроваджується в державних установах з 2015 року [8]. Ключові характеристики:

- охоплення понад 1000 об'єктів моніторингу;
- інтеграція з 15+ типами лічильників;
- автоматичне формування звітності за формами НКРЕКП;
- модуль бенчмаркінгу для порівняння однотипних об'єктів.

1.3 Аналіз зарубіжних систем моніторингу

1.3.1 ENERGY STAR Portfolio Manager (США)

ENERGY STAR Portfolio Manager [9] найбільш розвинена система управління енергоефективністю будівель з такими показниками:

- охоплення 25% комерційних будівель США (450,000+ об'єктів);
- економічний ефект - $4.8 млрд щорічної економії;

Функціональність:

- повноцінний REST API для автоматизації обміну даними;
- ENERGY STAR Score (1-100 балів) для оцінки ефективності;
- інтеграція з 180+ комунальними підприємствами;
- інструменти Investment Prioritization для планування інвестицій;
- автоматичне порівняння з національними бенчмарками.

Управлінські інструменти [10]:

- щомісячні звіти про динаміку споживання;
- алерти при відхиленнях від норм;
- розрахунок ROI для енергоефективних заходів;
- сертифікація будівель з високою ефективністю.

1.3.2 Display Energy Certificates System (Великобританія)

Display Energy Certificates System [11] обов'язкова система для публічних будівель площею понад 250 м² [12] (охоплено більше 40,000 публічних будівель).

Ключові елементи:

- A-G шкала енергетичної ефективності;
- обов'язкова щорічна сертифікація;
- штрафи £500-1000 за відсутність сертифікату;
- публічне розміщення сертифікатів.

Управлінські процедури [13]:

- обов'язкові візити акредитованих оцінювачів;
- використання сертифікованого програмного забезпечення;

– контроль Trading Standards Officers;
– інтеграція з системою планування капітальних інвестицій.

1.3.3 EU Energy Efficiency Reporting System

EU Energy Efficiency Reporting System [14] це загальноєвропейська система моніторингу виконання Директиви з енергоефективності.

Ключові елементи:

– обов'язкові національні плани (NECPs) на 10 років;
– прогрес-звіти кожні 2 роки.

Цільові показники:

– скорочення споживання на 11.7% до 2030 року;
– 1.9% щорічне скорочення в публічному секторі;
– 3% щорічна реновація державних будівель.

Механізми управління [15]:

– централізована платформа звітності;
– автоматичний розрахунок розривів до цілей;
– рекомендації щодо коригуючих заходів;
– можливість введення наднаціональних заходів при невиконанні.

Аналіз систем енергетичного менеджменту.

Відповідно до ДСТУ ISO 50001:2020 [16], система енергетичного менеджменту – це система управління, що визначає енергетичну політику та цілі, енергетичні завдання, плани дій і процеси для досягнення цілей та енергетичних завдань.

Впроваджена СЕнМ має охоплювати різні процеси – від визначення середовища організації до удосконалення самої системи.

Моніторинг впровадження СЕнМ включає:

– перевірку наявності документально підтвердженого факту впровадження (накази, програми, регламенти, договори);

– оцінку функціонування системи, а не лише її декларування;

– аналіз якості впровадження (виконання всіх необхідних процедур);

– оцінку ефективності впровадження (досягнення результатів, скорочення енергоспоживання).

Аналіз місцевих енергетичних планів.

Місцеві енергетичні плани – це стратегічні документи, що визначають цілі щодо підвищення енергоефективності та використання відновлюваних джерел енергії. Моніторинг МЕП передбачає відстеження:

– рівня виконання запланованих енергозберігаючих заходів;

– обсягів скорочення споживання енергоресурсів;

– впливу заходів на ключові показники енергоефективності.

Постанова КМУ 1460 від 2021 року встановлює вимоги до державних органів та підприємств щодо впровадження систем енергетичного менеджменту. Згідно з цією постановою:

– визначено конкретні умови до установ та підприємств, які підпадають під дію постанови;

– встановлено терміни та етапи впровадження СЕнМ;

– визначено вимоги до звітності про результати впровадження;

– передбачено необхідність моніторингу ефективності впроваджених заходів.

Наразі не існує спеціалізованої інформаційної системи для моніторингу виконання цієї постанови, що ускладнює процес збору, аналізу даних та оцінки ефективності.

Висновки з аналізу існуючих систем.

На основі проведеного комплексного аналізу можна зробити наступні висновки:

– фрагментованість систем моніторингу в Україні (наявні системи (ДАЕЕ, АСЕМ, АІС "ЕНЕРГОСЕРВІС") функціонують ізольовано, без належної інтеграції та стандартизації обміну даними, що на даний момент унеможливлює формування цілісної картини енергоефективності на національному рівні);

– низький рівень автоматизації (переважна більшість процесів збору та обробки даних здійснюється в ручному або напівавтоматичному режимі, що призводить до затримок, помилок та неможливості оперативного реагування);

– відсутність єдиних стандартів та методологій (різні системи використовують власні підходи до розрахунку показників енергоефективності, що ускладнює порівняння та бенчмаркінг) [17];

– недостатнє охоплення об'єктів моніторингу (лише 20% державних будівель підключено до систем енергомоніторингу, що свідчить про значний нереалізований потенціал економії);

– успішний міжнародний досвід (системи ENERGY STAR, DEC та EU Reporting демонструють ефективність комплексного підходу з обов'язковістю участі, автоматизацією процесів та чіткими KPI).

Таким чином, існує потреба в розробці єдиної інформаційної системи для ефективного виконання вимог постанови КМУ №1460 та досягнення національних цілей з енергоефективності необхідна розробка та впровадження єдиної інформаційної системи моніторингу [18].

1.4 Дослідження методів оцінки ефективності програм у сфері енергоефективності

Оцінка ефективності програм енергоефективності є комплексним завданням, яке потребує врахування різних факторів та показників. Розглянемо основні методи

та підходи, які використовуються для такої оцінки, з особливим фокусом на державні органи та підприємства, що підпадають під дію постанови КМУ 1460.

Методи на основі ДСТУ ISO 50001:2020.

Стандарт ДСТУ ISO 50001:2020 визначає вимоги до системи енергетичного менеджменту та пропонує методологію для оцінки її ефективності. Відповідно до цього стандарту, оцінка ефективності базується на наступних елементах:

– енергетичний аналіз це процес визначення значущих сфер використання енергії та можливостей для покращення енергетичної результативності [19];

– показники енергетичної результативності – кількісні значення або міри енергетичної результативності, визначені організацією [20];

– базовий рівень енергоспоживання – кількісний показник, що дає змогу порівнювати енергетичну результативність;

– енергетична цільова задача це конкретне завдання з енергетичної результативності, необхідне для досягнення енергетичної цілі [21].

Оцінка ефективності за цим методом передбачає порівняння фактичних показників енергоспоживання з базовим рівнем та цільовими завданнями, з урахуванням відповідних факторів (наприклад, погодних умов, інтенсивності використання будівель тощо).

Методи бенчмаркінгу енергетичної ефективності.

Бенчмаркінг енергетичної ефективності – це порівняння показників споживання енергії та ефективності об'єктів з нормативними вимогами та середніми значеннями для подібних об'єктів. Основою методології бенчмаркінгу енергоефективності об'єктів є ДСТУ ISO 50001:2020, що визначає підходи до аналізу та порівняння енергетичних показників об'єктів, систем або процесів.

Основні методи бенчмаркінгу:

– метод порівняння на основі енергетичних сертифікатів (порівняння на основі класу енергетичної ефективності будівлі та питомого енергоспоживання);

– метод комплексної енергетичної оцінки (оцінювання первинної енергії, що дозволяє порівнювати різні види енергії (теплову, електричну, тощо) через коефіцієнти використання первинної енергії);

– метод порівняння з базовою лінією споживання (порівняння фактичного споживання з базовим рівнем, визначеним на основі історичних даних або нормативів).

Для державних органів та підприємств бенчмаркінг може використовуватись для порівняння ефективності різних об'єктів однієї установи або для порівняння з аналогічними установами.

Економічна оцінка ефективності є важливою складовою загальної оцінки ефективності програм енергоефективності. Методи включають:

– аналіз витрат і вигод (порівняння всіх витрат на реалізацію програми з економічними вигодами від її впровадження) [22];

– розрахунок періоду окупності (визначення часу, необхідного для повернення інвестицій через економію енергоресурсів) [23];

– розрахунок чистої приведеної вартості (оцінка вартості майбутніх грошових потоків, приведених до поточного моменту);

– аналіз вартості життєвого циклу (врахування всіх витрат протягом життєвого циклу заходів з енергоефективності).

Екологічна оцінка передбачає аналіз впливу програм енергоефективності на навколишнє середовище (розрахунок зменшення викидів парникових газів CO_2 внаслідок впровадження енергоефективних заходів).

Для бюджетних установ, що підпадають під дію постанови КМУ 1460, економічна оцінка повинна враховувати особливості бюджетного фінансування та специфіку державних закупівель.

Методи ефективності впровадження систем енергетичного менеджменту в державних органах та підприємствах специфічні для оцінки впровадження СЕнМ які застосовуваються згідно постанови КМУ 1460:

– оцінка відповідності вимогам постанови (перевірка наявності всіх необхідних елементів СЕнМ, визначених постановою);

– оцінка прогресу впровадження (аналіз динаміки виконання етапів впровадження СЕнМ);

– оцінка скорочення споживання енергоресурсів (аналіз зміни абсолютних та питомих показників споживання енергії);

Висновки з дослідження методів оцінки ефективності.

На основі проведеного дослідження можна зробити наступні висновки:

– ефективність програм енергоефективності має оцінюватися комплексно, з урахуванням технічних, економічних, екологічних та соціальних аспектів;

– методи на основі ДСТУ ISO 50001:2020 надають структурований підхід до оцінки ефективності систем енергетичного менеджменту, але потребують адаптації до специфіки державних органів та підприємств;

– методи бенчмаркінгу дозволяють порівнювати ефективність різних об'єктів та визначати потенціал для подальшого покращення, але потребують доступу до якісних даних для порівняння;

– економічні методи оцінки дають важливу інформацію про фінансову ефективність програм, але повинні враховувати особливості бюджетного фінансування;

– для оцінки ефективності впровадження СЕнМ відповідно до постанови необхідні специфічні методи, які враховують вимоги цієї постанови.

Для розроблюваної інформаційної системи доцільно використовувати комплексний підхід до оцінки ефективності, який поєднує елементи різних методів та дозволяє врахувати специфіку державних органів та підприємств.

1.5 Формування вимог до інформаційної системи

Базуючись на аналізі існуючих систем моніторингу та методів оцінки ефективності, а також враховуючи специфіку державних програм у сфері енергоефективності, можна сформулювати наступні вимоги до розроблюваної інформаційної системи.

Функціональні вимоги.

а) збір та введення даних (зручне введення даних через веб-інтерфейс, збір даних про впровадження СЕнМ згідно з визначеною структурою, збір даних про реалізацію МЕП та інших програм енергоефективності);

б) верифікація та валідація даних (автоматична перевірка форматів і типів даних, перевірка логічної узгодженості даних, візуальне позначення проблемних даних для користувачів);

в) зберігання даних (збереження структурованих даних про програми енергоефективності, забезпечення цілісності та узгодженості даних);

г) аналіз даних (статистичний аналіз показників енергоефективності, розрахунок ключових показників ефективності, порівняльний аналіз (бенчмаркінг) об'єктів);

д) візуалізація даних (інтерактивні дашборди з ключовими показниками, графіки та діаграми для відображення даних, формування табличних звітів);

е) оцінка ефективності (розрахунок показників енергетичної результативності, економічна оцінка ефективності програм, екологічна оцінка (скорочення викидів CO_2));

є) управління користувачами (реєстрація та автентифікація користувачів, розмежування прав доступу відповідно до ролей, налаштування профілів користувачів, журналювання дій користувачів, підтримка делегування повноважень);

и) генерація звітів (формування стандартизованих звітів, експорт звітів у форматі PDF, візуалізація результатів у звітах).

Нефункціональні вимоги.

а) продуктивність (швидка обробка великих обсягів даних, оптимізовані запити до бази даних, ефективна робота з різними типами даних, швидке формування звітів та візуалізацій);

е) надійність (стабільна робота системи, резервне копіювання даних, відновлення після збоїв, журналювання помилок);

є) безпека (захист даних від несанкціонованого доступу, шифрування конфіденційних даних, автентифікація та авторизація користувачів, захист від типових вразливостей веб-додатків, журналювання дій користувачів з точки зору безпеки);

и) зручність використання (інтуїтивно зрозумілий інтерфейс, адаптивний дизайн для різних пристроїв, зрозуміла документація та підказки, оптимізована навігація та пошук);

і) підтримка та обслуговування (модульна архітектура для спрощення підтримки, документація коду та системи, можливість оновлення окремих компонентів);

ї) сумісність (відповідність сучасним веб-стандартам, сумісність з різними браузерами, підтримка різних операційних систем користувачів, відповідність законодавчим вимогам, дотримання стандартів доступності інтерфейсу).

Система повинна підтримувати наступні типи користувачів з відповідними правами:

а) адміністратор системи (повний доступ до всіх даних системи, управління користувачами та ролями, налаштування структури довідників та прав доступу, контроль за функціонуванням всієї системи);

б) адміністратор державного органу або організації (управління даними своєї та підпорядкованих організацій, призначення відповідних ролей користувачам,

моніторинг активності підпорядкованих користувачів, моніторинг показників енергоефективності своєї та підпорядкованих організацій, контроль якості даних на рівні організації);

в) користувач державного органу або організації (введення та перегляд даних за об'єктами організації, формування звітів за шаблонами, перегляд аналітики за об'єктами організації, обмежений доступ до даних відповідно до рівня доступу);

г) енергоаудитор (доступ до аналітичних звітів та статистики, моніторинг виконання енергозберігаючих заходів, оцінка ефективності впроваджених заходів).

Для кожної ролі необхідно розробити відповідний інтерфейс користувача, який надаватиме доступ лише до тих функцій та даних, які відповідають повноваженням цієї ролі.

Система призначена для роботи з даними державних органів та підприємств, визначених у постанові КМУ 1460, що обмежує коло учасників та об'єм даних для обробки. З огляду на це, система повинна забезпечувати:

а) перевірку відповідності організації вимогам постанови (ведення реєстру організацій, які підпадають під дію постанови, відмітка про відповідність організації критеріям постанови, можливість фільтрації даних за ознакою відповідності постанові);

б) моніторинг виконання постанови (відстеження прогресу впровадження СЕнМ відповідно до етапів, визначених у постанові, контроль дотримання термінів впровадження, формування спеціалізованих звітів про стан виконання вимог постанови, аналіз ефективності заходів впроваджених у рамках виконання постанови).

1.6 Постановка задачі дослідження

На основі проведеного аналізу предметної області та сформованих вимог до інформаційної системи можна визначити основні задачі, які необхідно вирішити в рамках цього дослідження.

Мета дослідження: Розробка інформаційної системи для автоматизованого моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, яка забезпечить повний цикл моніторингу — від збору первинних даних до формування аналітичних звітів та оцінки ефективності впроваджених заходів.

Задачі дослідження:

а) розробка архітектури інформаційної системи (проектування загальної архітектури системи, визначення основних компонентів та їх взаємозв'язків, розробка схеми бази даних для зберігання інформації);

б) розробка модуля внесення та збору даних (проектування інтерфейсу для ручного введення даних, реалізація механізмів верифікації та валідації даних) [24];

в) розробка аналітичного модуля (проектування алгоритмів статистичної обробки даних, розробка механізмів для порівняльного аналізу (бенчмаркінгу));

г) розробка модуля оцінки ефективності (проектування алгоритмів розрахунку показників енергетичної результативності, реалізація комплексної методології оцінки ефективності) [25];

д) розробка користувацького інтерфейсу (проектування інтерфейсів для різних ролей користувачів, розробка інтерактивних дашбордів та візуалізацій, реалізація механізмів для генерації та експорту звітів);

е) реалізація та інтеграція модулів системи (розробка серверної частини на базі Django [26], реалізація клієнтської частини з використанням HTML/CSS, JavaScript та Bootstrap [27], інтеграція розроблених модулів у єдину систему, контейнеризація компонентів за допомогою Docker [28]);

є) тестування розробленої системи (розробка плану тестування, проведення функціонального тестування, оцінка відповідності системи поставленим вимогам).

Об'єктна область дослідження:

Інформаційна система призначена для роботи з даними державних органів та підприємств, визначених у постанові КМУ 1460 від 2021 року. Об'єктна область дослідження включає:

а) організації (державні органи влади, державні підприємства, інші установи, визначені постановою);

б) системи енергетичного менеджменту (організаційна структура СЕнМ, декларація енергетичної політики, план діяльності системи енергоменеджменту, заходи з підвищення енергоефективності);

в) будівлі та споруди (адміністративні будівлі, спеціалізовані будівлі, навчальні, медичні, культурні, технічні споруди та об'єкти інфраструктури);

г) енергетичні ресурси (електроенергія, теплова енергія, природний газ, інші види енергоресурсів);

д) показники ефективності (технічні показники, економічні показники, екологічні показники, комплексні індикатори).

Обмеження:

а) система розробляється для моніторингу організацій, визначених постановою КМУ 1460 від 2021 року;

б) система не передбачає інтеграцію з державними реєстрами;

в) робота з даними здійснюється з урахуванням вимог законодавства про захист інформації та Постанови КМУ №627 від 30.05.2024 "Про реалізацію експериментального проекту з декларування відповідності комплексних систем захисту інформації".

Вимоги до використання системи:

а) користувачі системи мають базові навички роботи з комп'ютером та веб-інтерфейсами;

б) організації мають доступ до інтернету та необхідне технічне забезпечення;

в) актуальні дані про енергоспоживання доступні для введення в систему.

В результаті виконання дослідження очікується отримати розроблену інформаційну систему моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, набір алгоритмів та методів для аналізу та оцінки ефективності програм енергоефективності, рекомендації щодо впровадження та використання розробленої системи в державних органах та підприємствах,оцінку ефективності розробленої системи на основі результатів тестування.

Розроблена інформаційна система дозволить автоматизувати процеси моніторингу впровадження СЕнМ та оцінки ефективності програм енергоефективності, забезпечити прозорість та доступність інформації, а також підвищити якість прийняття управлінських рішень на основі актуальних даних.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Розробка архітектури системи

Архітектура інформаційної системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності має забезпечувати модульність, масштабованість, надійність та ефективну взаємодію всіх компонентів. На основі сформованих вимог пропонується багаторівнева архітектура (рис. 2.1).

Компоненти системи:

а) веб-сервер:

– Django Web Server (основний веб-сервер на порту 8000);
– REST API для взаємодії з системою.

б) база даних:

– PostgreSQL [29] (реляційна база даних для зберігання всіх даних системи);
– PgAdmin (інструмент для адміністрування бази даних).

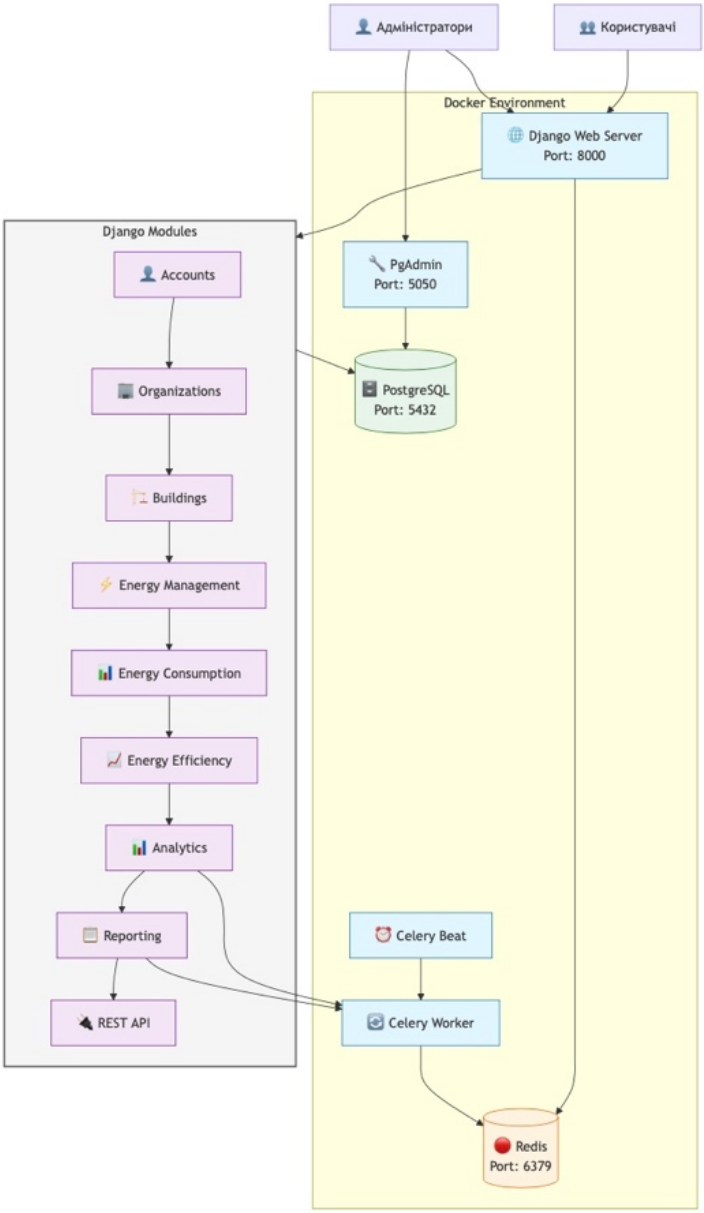

Рисунок 2.1 – Архітектура

в) кешування та черги:

– Redis (використовується для кешування та як брокер повідомлень для Celery).

г) фонова обробка:

– Celery Worker (обробка асинхронних завдань);

– Celery Beat (планувальник періодичних завдань).

д) Django додатки:

– Accounts (управління користувачами та автентифікація);
– Organizations (управління організаціями);
– Buildings (управління будівлями);
– Energy Management (системи енергетичного менеджменту);
– Energy Consumption (облік споживання енергоресурсів);
– Energy Efficiency (заходи з енергоефективності);
– Analytics (аналітика та візуалізація даних);
– Reporting (формування звітів).

е) технології:

– веб-фреймворк (Django 4.2.1);
– API (Django REST Framework);
– база даних (PostgreSQL 14);
– кешування (Redis 7);
– асинхронні завдання (Celery 5.2.7);
– контейнеризація (Docker, Docker Compose);
– фронтенд (Bootstrap 5);
– візуалізація даних (Matplotlib [30], Plotly[31]).

Архітектурні рішення:

– модульна архітектура;
– розподіл на серверну та клієнтську частини;
– використання RESTful API для взаємодії між компонентами;
– підтримка контейнеризації для спрощення розгортання.

Система складається з наступних основних модулів (рис. 2.2):

– користувачі та доступ (accounts);
– організаційна структура (organizations, buildings);
– дані та аналітика (energy_consumption, energy_efficiency,

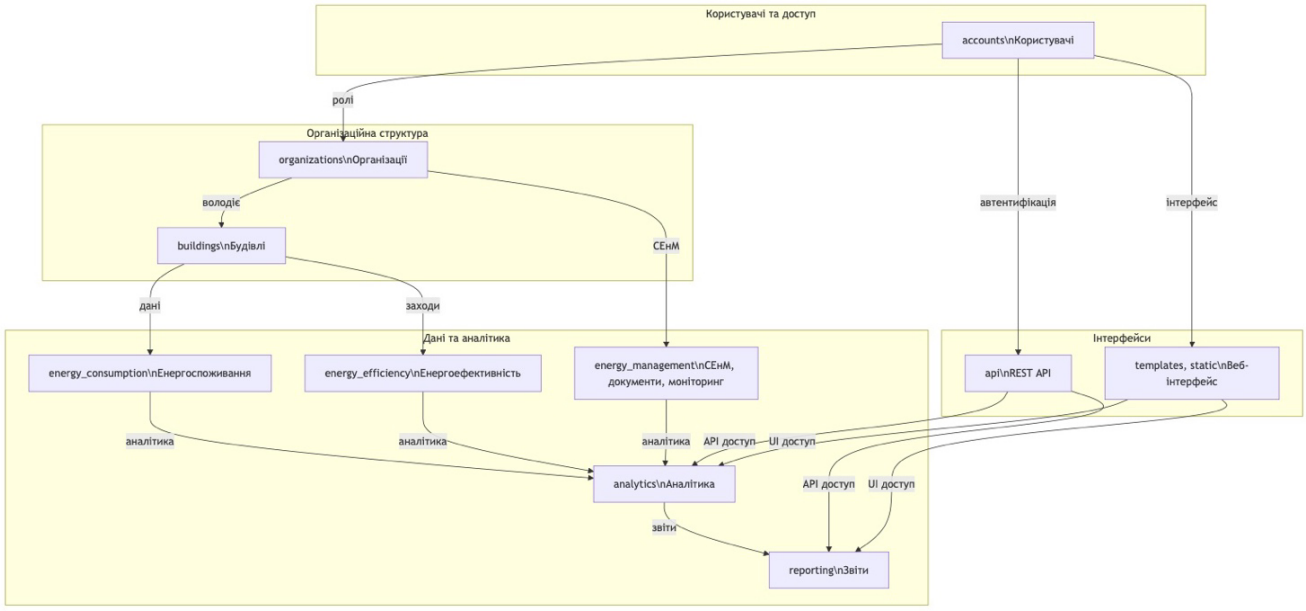

Рисунок 2.2 – Основні модулі системи

energy_management (СЕнМ, документи, моніторинг), analytics, reporting);

– інтерфейси (api (REST API), templates/static (веб-інтерфейс)).

Основні зв'язки:

а) користувачі мають ролі в організаціях;

б) організації володіють будівлями;

в) будівлі генерують дані для модулів енергоспоживання та енергоефективності;

г) дані та заходи аналізуються й агрегуються в аналітиці та звітах;

д) API та UI забезпечують доступ до аналітики та звітів.

Опис взаємодії компонентів:

а) клієнтський рівень взаємодіє з серверним рівнем через HTTP/HTTPS запити, отримуючи та відправляючи дані у форматі JSON;

б) серверний рівень обробляє запити від клієнтського рівня, застосовує бізнес-логіку та взаємодіє з рівнем даних для отримання або збереження інформації;

в)рівень даних забезпечує зберігання та доступ до даних, підтримуючи цілісність та узгодженість інформації.

Архітектура безпеки:

Безпека системи забезпечується на різних рівнях:

а) рівень аутентифікації та авторизації:

– аутентифікація користувача;
– ролева модель доступу;
– управління сесіями.

б) рівень передачі даних:

– шифрування даних при передачі (HTTPS);
– захист від CSRF-атак.

в) рівень зберігання даних:

– шифрування конфіденційних даних;
– журналювання доступу до даних.

Структура розгортання системи.

Система розгортається в контейнеризованому середовищі з використанням Docker та Docker Compose, що забезпечує ізоляцію компонентів, спрощує масштабування та підтримку.

Архітектура забезпечує:

– масштабованість: можливість горизонтального масштабування компонентів;
– надійність: ізоляція компонентів та відмовостійкість;
– гнучкість: можливість оновлення окремих компонентів без впливу на інші;
– ізоляція компонентів та контроль доступу;
– спрощене розгортання: автоматизоване розгортання за допомогою Docker Compose.

Обрана архітектура забезпечує всі необхідні функціональні можливості, відповідає сформованим вимогам та дозволяє створити надійну, масштабовану та ефективну інформаційну систему для моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності.

2.2 Проектування бази даних

База даних є основним компонентом рівня даних інформаційної системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності. Концептуальна модель даних представлена на (рис. 2.3).

Концептуальна модель даних включає наступні основні сутності:

– користувачі (Users) - інформація про користувачів системи, їх ролі та права доступу;

– організації (Organizations) - дані про державні органи та підприємства, визначені постановою КМУ 1460 від 2021 року;

– системи енергоменеджменту (EnMS) - інформація про впроваджені системи енергетичного менеджменту;

– будівлі (Buildings) - дані про будівлі, які є об'єктами моніторингу;

– енергоспоживання (Energy Consumption) - дані про споживання енергоресурсів;

– заходи з енергоефективності (Energy Efficiency Measures) - інформація про впроваджені заходи;

– показники ефективності (Performance Indicators) - дані про показники ефективності;

– звіти (Reports) - інформація про звіти, сформовані в системі.

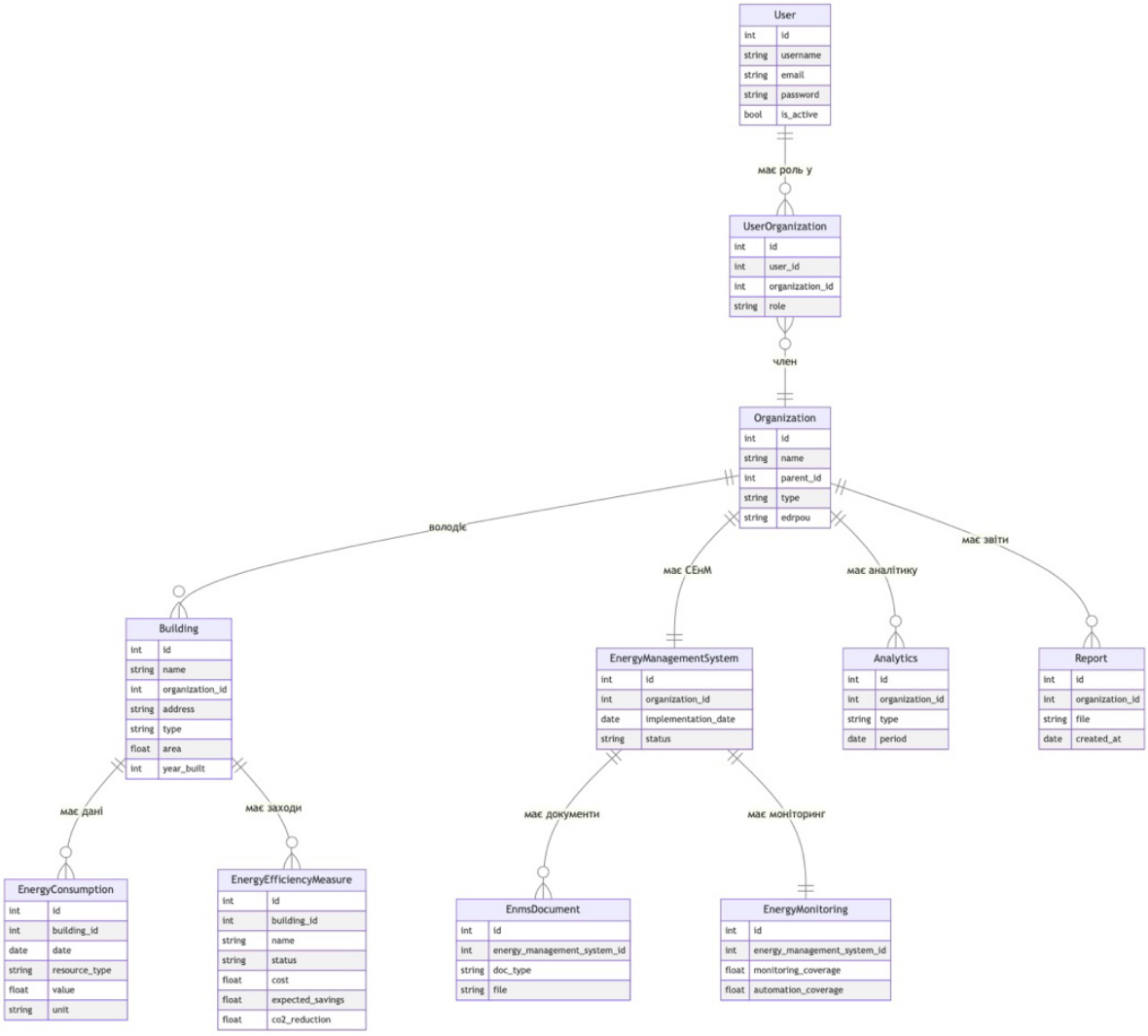

Рисунок 2.3 – Концептуальна модель даних

Логічна модель даних представлена у вигляді реляційної схеми з відношеннями між сутностями (табл. 2.1).

Таблиця 2.1 - Логічна модель даних

Таблиця users - користувачі системи	
id (PK)	унікальний ідентифікатор
username	ім'я користувача
email	електронна пошта
password_hash	хеш паролю
first_name	ім'я
last_name	прізвище
role_id (FK)	посилання на роль

Продовження таблиці 2.1

organization_id (FK)	посилання на організацію
created_at	дата створення
updated_at	дата оновлення
is_active	статус активності
Таблиця roles - ролі користувачів	
id (PK)	унікальний ідентифікатор
name	назва ролі
description	опис ролі
permissions	права доступу (JSON)
Таблиця organizations - організації	
id (PK)	унікальний ідентифікатор
name	назва організації
org_type	тип організації
parent_id (FK, самопосилання)	батьківська організація
kmu_1460_compliant	відповідність постанові КМУ 1460 (так/ні)
contact_person	контактна особа
contact_email	контактний email
contact_phone	контактний телефон
created_at	дата створення
updated_at	дата оновлення
Таблиця energy_management_systems - системи енергоменеджменту	
id (PK)	унікальний ідентифікатор
organization_id (FK)	назва організації
responsible_person	відповідальна особа
responsible_position	посада відповідальної особи
responsible_contacts	контакти відповідальної особи
order_number	номер наказу про впровадження
order_date	дата наказу
structure_approved	затвердження положення про структурний підрозділ (так/ні)
position_approved	затвердження посади енергоменеджера (так/ні)
contractor	підрядна організація (за наявності)
certification_status	стан сертифікації СЕМ (сертифікована/не сертифікована)
policy_approved	затвердження енергетичної політики (так/ні)
policy_order_number	номер наказу про політику
policy_order_date	дата наказу про політику
plan_approved	затвердження плану (так/ні)
plan_order_number	номер наказу про план
plan_order_date	дата наказу про план
plan_period	термін дії плану
energy_goal	енергетична ціль

Продовження таблиці 2.1

website_link	посилання на сторінку офіційного сайту
created_at	дата створення
updated_at	дата оновлення
Таблиця buildings - будівлі	
id (PK)	унікальний ідентифікатор
organization_id (FK)	назва організації
name	назва будівлі
address	адреса
building_type	тип будівлі
total_area	загальна площа (м²)
heated_area	опалювальна площа (м²)
heated_volume	опалювальний об'єм (м³)
energy_class	клас енергоефективності
certificate_id	номер енергетичного сертифіката
technical_passport	наявність технічного паспорту (так/ні)
property_registration	державна реєстрація прав на будівлю (так/ні)
land_registration	державна реєстрація прав на земельну ділянку (так/ні)
created_at	дата створення
updated_at	дата оновлення
Таблиця energy_monitoring - енергомоніторинг	
id (PK)	унікальний ідентифікатор
organization_id (FK)	назва організації
buildings_total	загальна кількість будівель
buildings_monitored	кількість будівель охоплених енергомоніторингом
buildings_automated	кількість будівель з автоматизованим енергомоніторингом
software_used	використання програмного продукту (назва/ні)
certificate_available	наявність сертифікату ЕЕБ (клас/ні)
created_at	дата створення
updated_at	дата оновлення
Таблиця energy_consumption - споживання енергоресурсів	
id (PK)	унікальний ідентифікатор
building_id (FK)	будівля
resource_type	тип ресурсу (електроенергія, тепло, газ, тощо)
year	рік
month	місяць
consumption	споживання (кВт·год, Гкал, м³)
consumption_kwh	споживання в кВт·год
cost	вартість (грн)
specific_consumption	питоме споживання (кВт·год/м²)

Продовження таблиці 2.1

created_at	дата створення
updated_at	дата оновлення
Таблиця energy_efficiency_measures - заходи з енергоефективності	
id (PK)	унікальний ідентифікатор
building_id (FK)	будівля
enms_id (FK)	система енергоменеджменту
name	назва заходу
description	опис заходу
type	тип заходу
status	статус впровадження
implementation_date	дата впровадження
cost	вартість впровадження
energy_saving	економія енергії (кВт·год/рік)
energy_saving_percent	економія енергії (%)
cost_saving	економія коштів (грн/рік)
co2_reduction	скорочення викидів CO_2
payback_period	термін окупності
created_at	дата створення
updated_at	дата оновлення
Таблиця performance_indicators - показники ефективності	
id (PK)	унікальний ідентифікатор
organization_id (FK)	організація
building_id (FK)	будівля
enms_id (FK)	система енергоменеджменту
indicator_type	тип показника
year	рік
month	місяць
value	значення показника
target_value	цільове значення
baseline_value	базове значення
unit	одиниця виміру
created_at	дата створення
updated_at	дата оновлення
Таблиця reports - звіти	
id (PK)	унікальний ідентифікатор
name	назва звіту
type	тип звіту
period_start	початок періоду
period_end	кінець періоду
organization_id (FK)	організація

Кінець таблиці 2.1

user_id (FK)	користувач, який створив звіт
creation_date	дата створення
content	вміст звіту (JSON)
status	статус звіту
file_path	шлях до файлу звіту
created_at	дата створення
updated_at	дата оновлення

Структура таблиць energy_management_systems, organizations та buildings спроектована з урахуванням полів, які містяться в Додатку 1 до Листа про впровадження СЕнМ в ДАЕЕ. Це забезпечує збереження всіх необхідних даних для моніторингу впровадження систем енергетичного менеджменту в організаціях, що підпадають під дію постанови КМУ 1460.

Модель даних реалізується в PostgreSQL з використанням наступних технічних рішень:

а) індекси для оптимізації пошуку:

– первинні ключі всіх таблиць;

– зовнішні ключі для зв'язків між таблицями;

– індекси за часто використовуваними полями (name, year, month тощо).

б) обмеження цілісності даних:

– перевірка унікальності (UNIQUE) для полів, які мають бути унікальними;

– перевірка не пустих значень (NOT NULL) для обов'язкових полів;

– перевірка доменних обмежень (CHECK) для полів з обмеженнями значень.

в) тригери та функції:

– автоматичне оновлення updated_at при зміні запису;

– автоматичний розрахунок похідних значень (наприклад, питомого споживання);

– валідація даних при вставці та оновленні.

г) транзакції для забезпечення атомарності операцій:

– забезпечення цілісності пов'язаних даних;

– відкат змін у разі помилок.

д) партиціонування для таблиць з великим обсягом даних:

– партиціонування таблиці energy_consumption за роками;

– розподілене зберігання історичних даних.

Міграції та версіонування бази даних.

Для забезпечення версіонування та керованого оновлення структури бази даних використовується механізм міграцій Django. Міграції дозволяють:

– відстежувати зміни в структурі бази даних;

– автоматично застосовувати зміни до бази даних;

– відкатувати зміни у разі необхідності;

– синхронізувати структуру бази даних між різними середовищами (розробка, тестування, продакшн).

Для ефективного моніторингу організацій, визначених постановою КМУ 1460 від 2021 року, в структурі бази даних передбачені наступні особливості:

– поле kmu_1460_compliant в таблиці organizations для відмітки про відповідність організації критеріям постанови;

– спеціалізовані поля в таблиці energy_management_systems для відстеження прогресу впровадження СЕнМ згідно вимог постанови;

– структура таблиці energy_monitoring, адаптована до вимог звітності, визначених у постанові.

Управління доступом:

– обмеження прав доступу до бази даних;

– використання окремих користувачів бази даних для різних компонентів системи.

Аудит та журналювання:

– журналювання всіх операцій з даними;
– відстеження змін у критичних таблицях.

Резервне копіювання:

– регулярне створення резервних копій бази даних;
– стратегія відновлення у разі збоїв.

Така структура бази даних забезпечує ефективне зберігання та обробку всіх необхідних даних для моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, відповідає вимогам до системи та забезпечує необхідний рівень продуктивності, надійності та безпеки.

2.3 Проектування системи

2.3.1 Внесення та збір даних

Внесення та збір даних є ключовим компонентом інформаційної системи для забезпечення введення, верифікації та збереження даних про організації, підпорядкування, будівлі, енергоспоживання, програми енергоефективності і повинен бути реалізований як розподілений за напрямками модулем за кожним напрямком. Ці дані є вхідною точкою для всієї інформації і буде використовуватися для аналізу та оцінки ефективності та забезпечувати наступну функціональність:

а) ручне введення даних:

– форми для введення даних про організації, будівлі, системи енергоменеджменту;
– інтерфейс для введення даних про споживання енергоресурсів;
– форми для внесення інформації про заходи з енергоефективності;
– можливість редагування та оновлення даних.

б) верифікація та валідація даних:

– перевірка типів та форматів даних;

– перевірка логічної узгодженості даних;
– візуальне позначення проблемних даних.

в) механізми перевірки якості даних:

– алгоритми контролю якості даних;
– перевірка повноти даних;

Внесення та збір даних складається з наступних компонентів:

а) компонент форм введення даних:

– форми для структурованого введення даних відповідно до обраного розділу;
– клієнтська валідація введених даних;
– серверна валідація та обробка даних;
– механізми для автозаповнення та підказок.

б) компонент верифікації даних:

– алгоритми для перевірки типів та форматів даних;
– механізми для виявлення дублікатів та аномалій;
– перевірка логічної узгодженості даних;
– інтеграція з бізнес-правилами системи.

в) компонент зберігання даних:

– ORM-моделі для взаємодії з базою даних;
– механізми для транзакційного збереження даних;
– механізми для журналювання змін.

Алгоритми верифікації та валідації даних.

Для забезпечення достовірності та якості даних у системі реалізується алгоритм перевірки типів та форматів даних:

– перевірка відповідності типів даних (числові, текстові, дати);
– валідація форматів специфічних даних (email, телефон);
– перевірка діапазонів числових значень;

– валідація формату дат та періодів.

Форми введення даних.

Для введення даних про системи енергетичного менеджменту розроблено форми, які відповідають структурі Додатку 1 до Листа про впровадження СЕнМ:

а) форма введення даних про організаційну структуру енергоменеджменту:

– поля для введення інформації про відповідальну особу;

– поля для введення інформації про наказ про запровадження СЕнМ;

– поля для вказання статусу затвердження положення та посади енергоменеджера;

– поля для введення інформації про підрядну організацію (за наявності).

б) форма введення даних про план діяльності СЕнМ:

– поля для введення інформації про наказ затвердження плану;

– поля для вказання терміну дії плану;

– поля для введення енергетичної цілі;

– поле для введення посилання на офіційний сайт.

в) форма введення даних про будівлі:

– поля для введення інформації про функціональне призначення будівлі;

– поля для введення адреси та площі будівлі;

– поля для вказання статусу реєстрації прав та наявності технічного паспорту;

– поля для введення інформації про енергетичний клас будівлі.

Для ефективного моніторингу організацій, визначених постановою КМУ 1460, передбачені наступні особливості внесення та збору даних:

а) перевірка відповідності організації постанові:

– позначення організації, яка підпадає під дію постанови;

– можливість фільтрації даних за ознакою відповідності постанові.

б) спеціалізовані форми для введення даних згідно вимог постанови:

– адаптовані форми для введення інформації про впровадження СЕнМ;
– поля для відстеження прогресу виконання вимог постанови;
– форми для звітування про результати впровадження СЕнМ.

в) автоматизована валідація даних відповідно до вимог постанови:

– перевірка відповідності даних вимогам постанови.

Користувацький інтерфейс внесення та збору даних розроблений з урахуванням вимог до зручності використання та включає:

– інтуїтивно зрозумілі форми для введення даних з групуванням пов'язаних полів;
– інтерактивні елементи для спрощення введення (календарі, випадаючі списки, автозаповнення);
– валідація на стороні клієнта для миттєвого зворотного зв'язку;
– прогрес-бари та індикатори для довготривалих операцій;
– повідомлення про помилки та підказки для користувачів.

Проектування розділу внесення та збору даних забезпечує ефективне внесення, верифікацію та збереження всієї необхідної інформації для моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності.

2.3.2 Аналітичний модуль

Аналітичний модуль є важливим компонентом інформаційної системи, який відповідає за обробку, аналіз та візуалізацію даних про програми енергоефективності. Модуль забезпечує аналіз зібраних даних та формування звітності для організацій, визначених постановою з наступною функціональністю:

а) статистична обробка даних:

– розрахунок базових статистичних показників (середнє, мінімум, максимум);

– агрегація даних за різними параметрами (час, тип організації, тип будівлі).

б) порівняльний аналіз показників різних організацій та будівель.

в) візуалізація даних для створення графіків та діаграм, формування інформаційних панелей з ключовими показниками.

Аналітичний модуль складається з наступних компонентів:

а) компонент статистичної обробки (функції для розрахунку статистичних показників, алгоритми агрегації даних, розрахунок похідних показників);

б) компонент порівняльного аналізу (алгоритми порівняння показників, механізми для визначення базової лінії, функції для рейтингування об’єктів);

в) компонент для генерації стандартних звітів.

Для обробки даних у аналітичному модулі використовуються наступні інструменти:

а) Бібліотеки Python для аналізу даних:

– Pandas для обробки і структурування даних;
– NumPy [32] для роботи з числовими даними;
– SciPy [33] [34] для базового статистичного аналізу.

б) інструменти для візуалізації:

– Matplotlib для створення базових графіків;
– Plotly для створення інтерактивних діаграм.

Аналітичний модуль включає інформаційні панелі для візуалізації ключових показників:

а) панель загального моніторингу (ключові показники споживання енергоресурсів, динаміка споживання в часі);

б) панель порівняльного аналізу (порівняння показників різних організацій, рейтинги енергоефективності);

в) панель прогнозування для формування прогнозів ефекту від впровадження заходів.

Для моніторингу організацій, визначених постановою КМУ 1460 від 2021 року, аналітичний модуль включає:

а) спеціалізовані показники (індикатори виконання вимог постанови, показники ефективності впровадження СЕнМ);

б) порівняльний аналіз між організаціями (порівняння організацій одного типу, рейтингування за ступенем виконання вимог постанови);

в) аналіз термінів впровадження СЕнМ (моніторинг темпів впровадження).

Аналітичний модуль забезпечує візуалізацію даних (лінійні графіки для відображення динаміки у часі, стовпчикові діаграми для порівняння показників, кругові діаграми для відображення структури споживання).

Аналітичний модуль надає необхідні інструменти для аналізу даних про енергоефективність та формування звітності, що дозволяє відстежувати прогрес впровадження систем енергетичного менеджменту в організаціях.

2.3.3 Модуль оцінки ефективності

Модуль оцінки ефективності є важливим компонентом інформаційної системи, який забезпечує комплексну оцінку результативності впровадження систем енергетичного менеджменту та енергоефективних заходів в організаціях.

Функціональність модуля:

а) розрахунок показників енергетичної результативності для визначення показників питомого споживання енергії;

б) економічна оцінка (розрахунок економії коштів від впровадження енергоефективних заходів, оцінка терміну окупності заходів, визначення економічної ефективності впроваджених систем);

в) екологічна оцінка для розрахунку скорочення викидів CO_2;

г) комплексна оцінка енергоефективності (інтегральна оцінка енергоефективності об'єктів, формування рейтингів організацій);

д) оцінка виконання вимог постанови КМУ 1460 (моніторинг повноти впровадження СЕнМ, аналіз якості впровадження).

Архітектура модуля.

Модуль оцінки ефективності включає наступні компоненти:

а) компонент розрахунку показників енергоефективності (алгоритми для розрахунку базових та похідних показників, механізми нормалізації показників);

б) компонент економічної оцінки (функції для розрахунку економічного ефекту, алгоритми оцінки окупності, інструменти для фінансового аналізу);

в) компонент екологічної оцінки (функції для розрахунку скорочення викидів);

г) компонент комплексної оцінки (інтегральні методи оцінки, алгоритми ранжування та рейтингування);

д) компонент моніторингу виконання постанови (інструменти для відстеження прогресу, функції для оцінки відповідності вимогам).

Методи оцінки ефективності.

Модуль використовує наступні методи оцінки ефективності:

а) метод порівняння з базовою лінією (оцінка ефекту від впроваджених заходів);

б) метод оцінки економічної ефективності (розрахунок простого терміну окупності, визначення питомих витрат на 1м2 кВт·год);

в) метод оцінки відповідності вимогам постанови (перевірка наявності ключових елементів СЕнМ, оцінка повноти впровадження).

Особливості модуля для організацій згідно постанови КМУ 1460.

Модуль оцінки ефективності включає специфічні функції для моніторингу організацій, визначених постановою КМУ 1460:

а) оцінка прогресу впровадження СЕнМ для визначення етапу впровадження;

б) оцінка якості впровадження СЕнМ (перевірка наявності всіх необхідних елементів системи, фактичне зниження споживання енергоресурсів).

Модуль звітність про ефективність формує наступні звіти про ефективність:

а) звіт про енергетичну ефективність (показники питомого споживання енергоресурсів, оцінка ефективності впроваджених заходів);

б) звіт про економічну ефективність (розрахунок економії коштів, оцінка окупності заходів, фінансові показники);

в) звіт про відповідність вимогам постанови з оцінкою повноти впровадження СЕнМ для аналізу відповідності термінам;

г) звіт до ДАЕЕ.

Модуль оцінки ефективності забезпечує комплексний підхід до оцінки результативності заходів енергоефективності та впровадження СЕнМ в організаціях, що дозволяє визначати найбільш ефективні практики.

2.4 Проектування користувацького інтерфейсу

Користувацький інтерфейс є важливою складовою інформаційної системи, оскільки він забезпечує взаємодію між користувачем та функціональністю системи. Ефективний інтерфейс має бути інтуїтивно зрозумілим, зручним у використанні та надавати всі необхідні інструменти для роботи з системою.

Принципи проектування інтерфейсу:

а) інтуїтивність - інтерфейс повинен бути зрозумілим для користувачів з різним рівнем технічної підготовки;

б) узгодженість - елементи інтерфейсу мають бути послідовними та логічно структурованими;

в) ефективність - мінімізація кількості дій для виконання типових завдань;

г) адаптивність - коректне відображення на різних пристроях та екранах;

д) доступність - забезпечення доступу для користувачів з різними потребами.

Основні компоненти інтерфейсу:

а) система авторизації та автентифікації;

б) головна сторінка з панеллю навігації;

в) інтерфейси для введення та редагування даних;

г) інформаційні панелі (дашборди) для візуалізації результатів;

д) генерація звітів;

е) системні повідомлення та сповіщення.

Структура інтерфейсу.

Інтерфейс інформаційної системи має ієрархічну структуру з такими основними рівнями:

а) загальні елементи (присутні на всіх сторінках):

- верхня панель, навігаційним меню та кнопками доступу до профілю;
- нижня панель з інформацією про систему та контактними даними.

б) рівень розділів:

- розділ Організації - управління інформацією про організації;
- розділ Будівлі - управління даними про будівлі;
- розділ СЕнМ - робота з системами енергетичного менеджменту;
- розділ Енергоспоживання - моніторинг споживання ресурсів;
- розділ Заходи - відстеження заходів з енергоефективності;
- розділ Аналітика - статистична обробка та візуалізація;
- розділ Звіти - генерація та перегляд звітів.

в) рівень функціональних сторінок:

- сторінки перегляду списків об'єктів;
- сторінки перегляду деталей об'єкта;
- форми для додавання та редагування даних;
- інформаційні панелі та графіки.

Адаптація інтерфейсу для різних ролей користувачів.

Інтерфейс системи адаптується відповідно до ролей користувачів, визначених у вимогах (рис. 2.4).

а) адміністратор системи:

– повний доступ до всіх розділів системи;

– додаткові інструменти для управління користувачами та налаштування системи;

– розширена аналітика та звітність.

б) адміністратор організації:

– доступ до даних своєї організації та організацій що підпорядковані;

– доступ до звітів;

– панелі моніторингу показників енергоефективності.

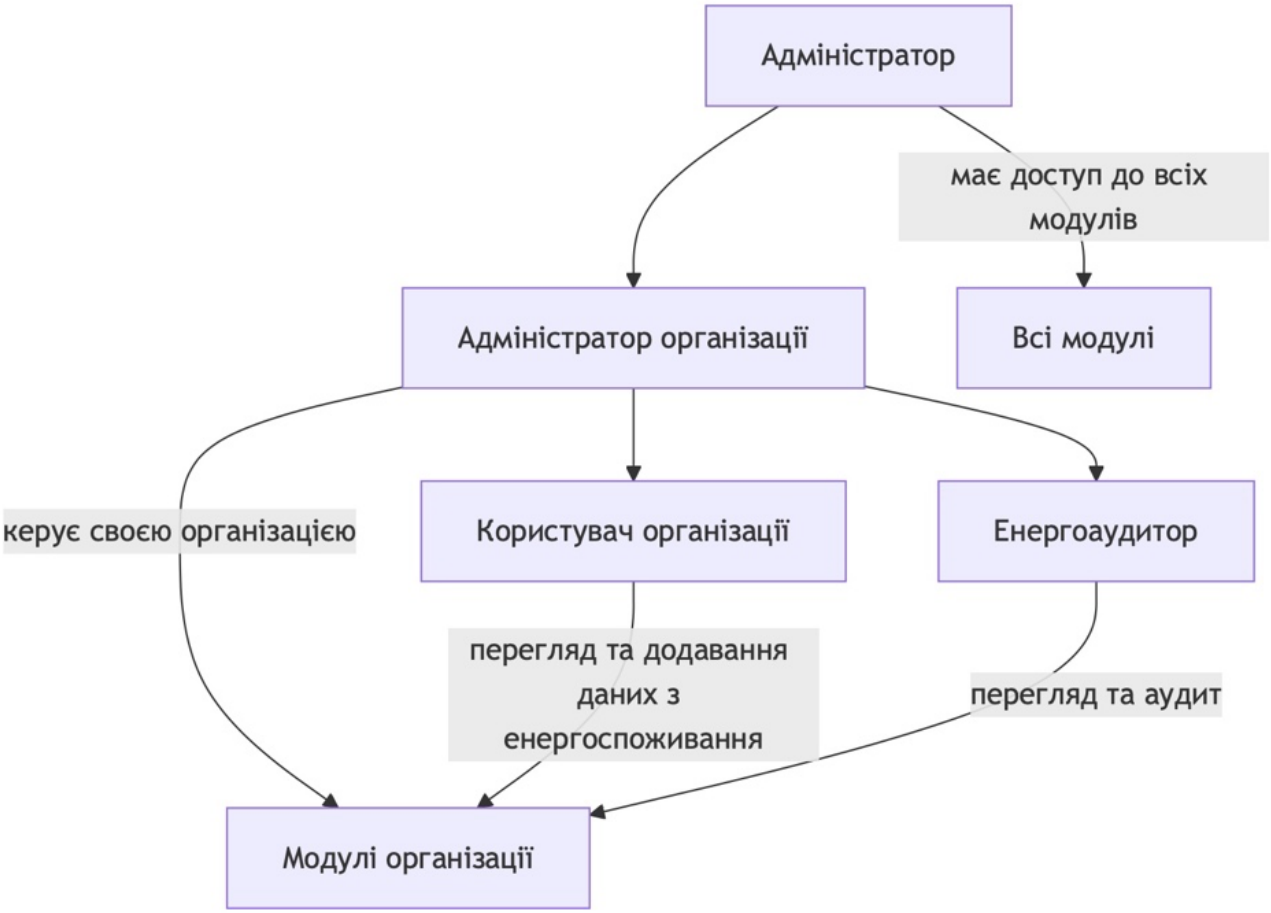

Рисунок 2.4 – Рольова модель користувачів

в) користувач організації:

– обмежений набір функцій для роботи з даними своїх об'єктів;
– спрощені форми для введення даних;
– базова аналітика та звітність.

г) енергоаудитор:

– доступ для аналізу ефективності до відповідної організації;
– доступ до аналітики відповідної організації.

Проектування окремих компонентів інтерфейсу:

а) форми для введення даних:

– логічне групування полів;
– підказки та пояснення для складних полів;
– валідація введених даних з підсвічуванням помилок.

б) таблиці для відображення списків:

– фільтрація за різними критеріями;
– пагінація для великих списків;
– функції пошуку та відбору даних.

в) інформаційні панелі (дашборди):

– інтерактивні графіки та діаграми;
– функції експорту візуалізацій.

г) форми для створення звітів:

– вибір шаблону звіту;
– попередній перегляд сформованого звіту;
– експорт у формат (PDF).

Навігація та взаємодія.

а) навігаційна система:

– головне меню вгорі з основними розділами.

б) взаємодія з користувачем:

– спливаючі підказки;
– контекстні меню;
– діалогові вікна для підтвердження дій;
– сповіщення про успішне виконання операцій та помилки.

Особливості інтерфейсу для моніторингу згідно постанови КМУ 1460.

Для ефективного моніторингу організацій, визначених постановою КМУ 1460, інтерфейс включає:

а) спеціалізовані форми для введення даних відповідно до Додатку 1 до Листа про впровадження СЕнМ.

б) індикатори відповідності вимогам постанови:
– кольорове кодування для візуалізації прогресу;
– маркери виконання ключових вимог;
– таймлайни для відстеження термінів впровадження.

в) спеціалізовані звіти для моніторингу виконання постанови:
– зведені звіти по організаціях;
– оцінка повноти впровадження СЕнМ;
– прогрес виконання у порівнянні з термінами.

Технічна реалізація інтерфейсу.

Інтерфейс системи реалізується з використанням наступних технологій:

а) Front-end розробка:
– HTML5 для структурування контенту;
– CSS3 з використанням Bootstrap для адаптивного дизайну;
– JavaScript з використанням jQuery для інтерактивності;
– бібліотеки для візуалізації даних (Plotly, Chart.js [35]).

б) взаємодія з сервером:
– AJAX для асинхронної взаємодії з сервером;
– REST API для обміну даними;

– JSON для передачі структурованих даних.

в) оптимізація інтерфейсу:

– кешування даних на стороні клієнта;

– відкладене завантаження (lazy loading) для великих списків;

– мінімізація запитів до сервера.

Прототипи ключових екранів:

а) головна сторінка (дашборд):

– загальна статистика щодо енергоспоживання;

– статус впровадження СЕнМ (рис. 2.5).

б) форма введення даних про СЕнМ (рис. 2.6):

– поля для введення даних про відповідальну особу;

– блок для інформації про накази та документи;

– поля для вказання статусів затвердження положень та посад;

– блок для внесення інформації про енергетичну політику та цілі (рис. 2.7).

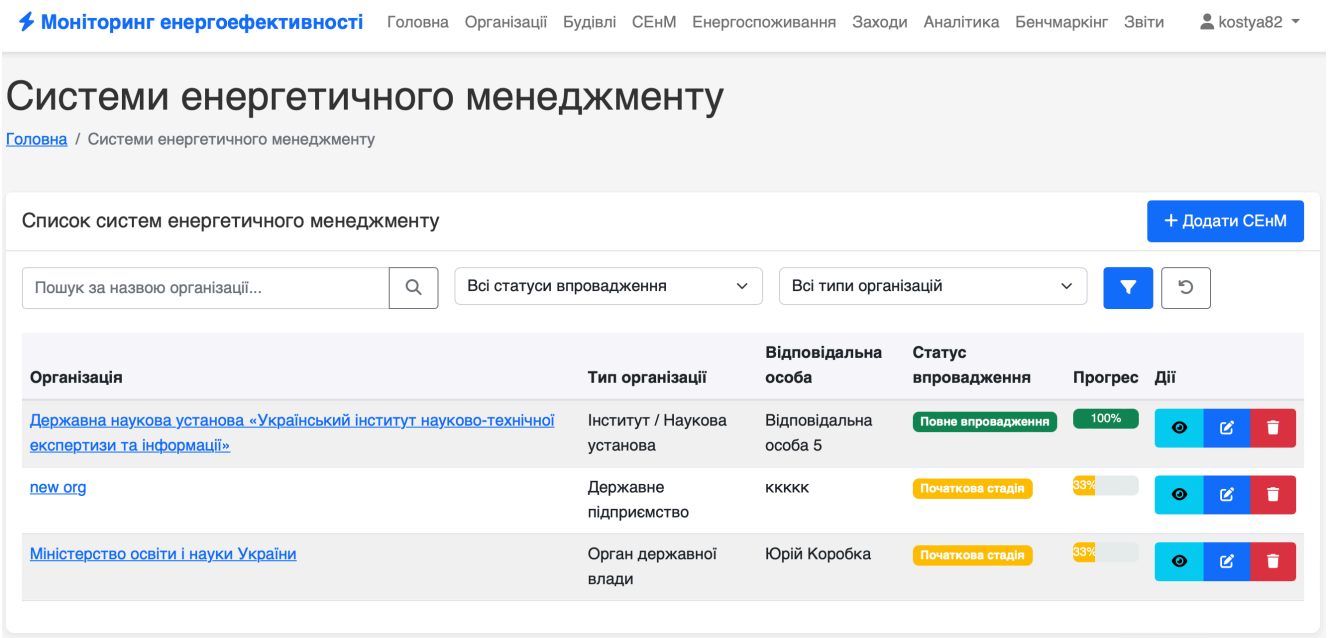

Рисунок 2.5 – Статус впровадження СЕнМ

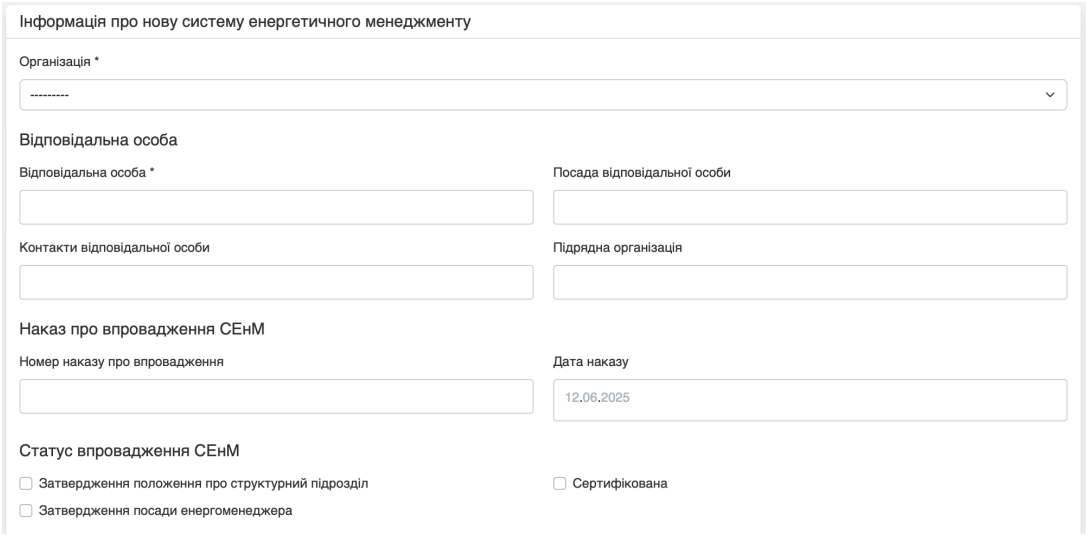

Рисунок 2.6 – Форма введення даних про СЕнМ

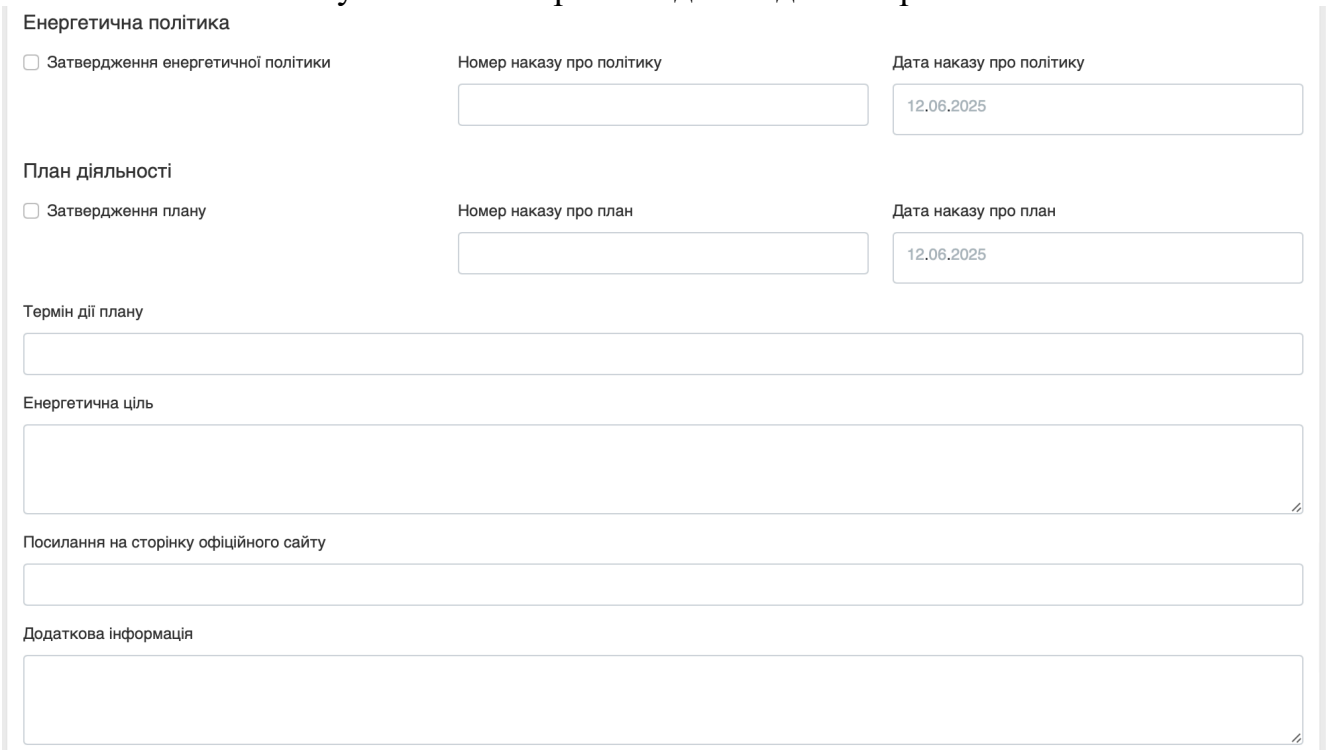

Рисунок 2.7 – Блок для внесення інформації про енергетичну політику та цілі

в) сторінка моніторингу енергоспоживання (рис. 2.8):

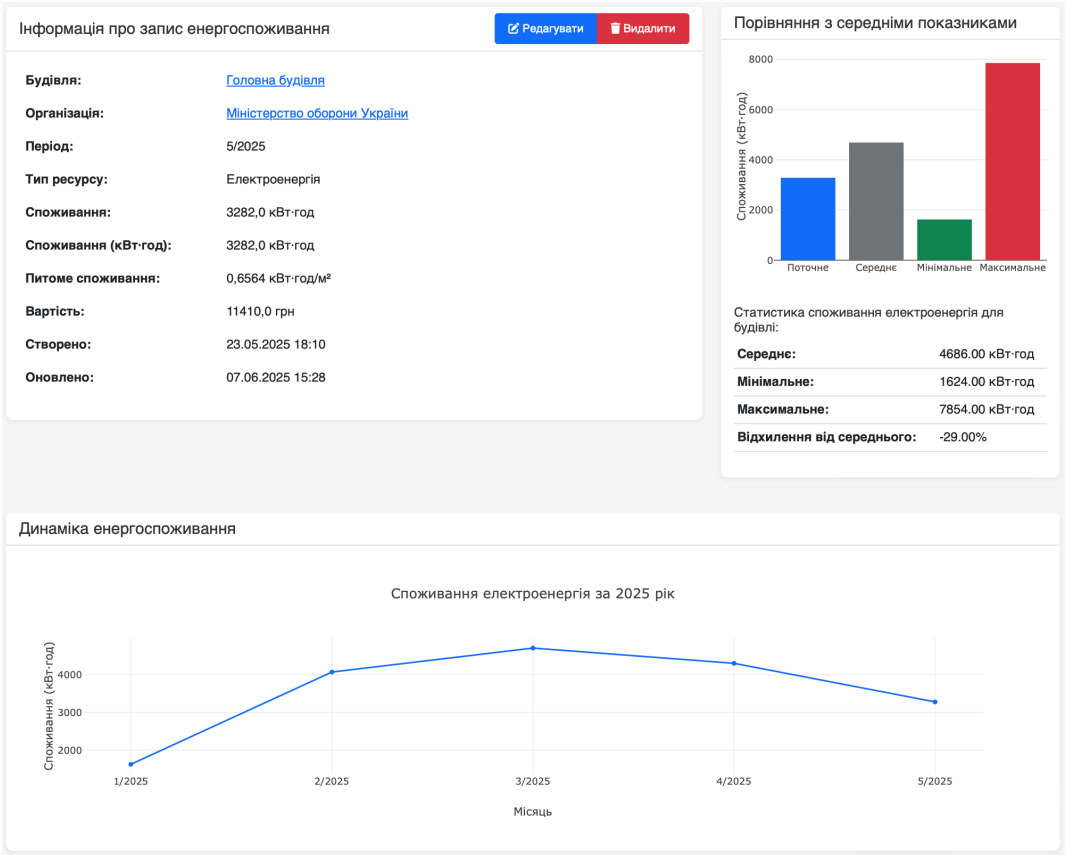

Рисунок 2.8 – моніторинг енергоспоживання

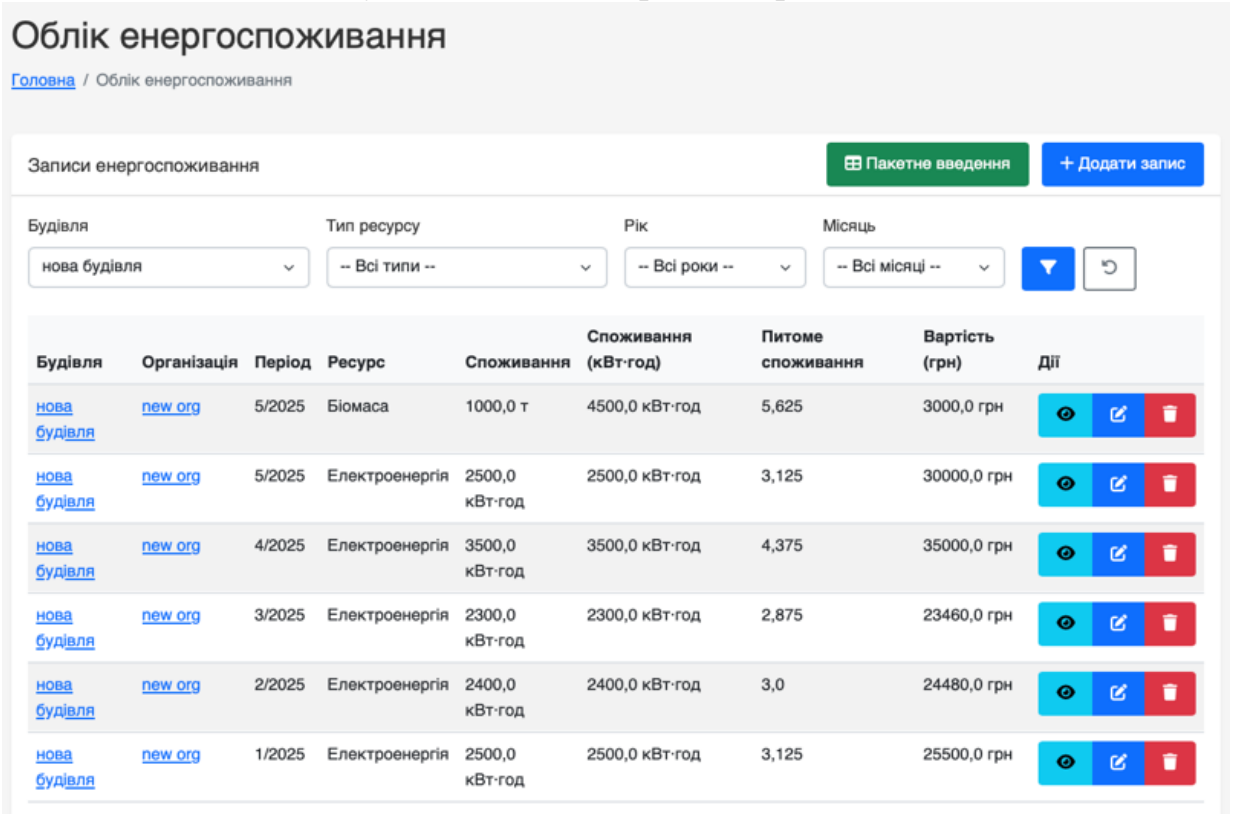

Рисунок 2.9 – Данні про споживання

– графіки споживання енергоресурсів;
– порівняння даних;
– таблиці з даними про споживання (рис. 2.9).

г) моніторинг виконання постанови КМУ 1460:

– загальний прогрес впровадження СЕнМ (рис. 2.10);

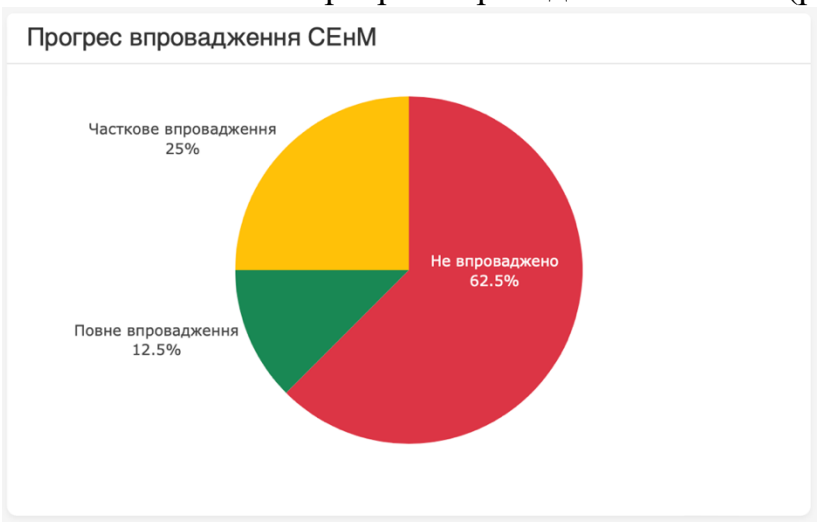

Рисунок 2.10 – загальний прогрес впровадження СЕнМ

– список організацій з індикаторами виконання;
– таймлайн виконання етапів (рис. 2.11).

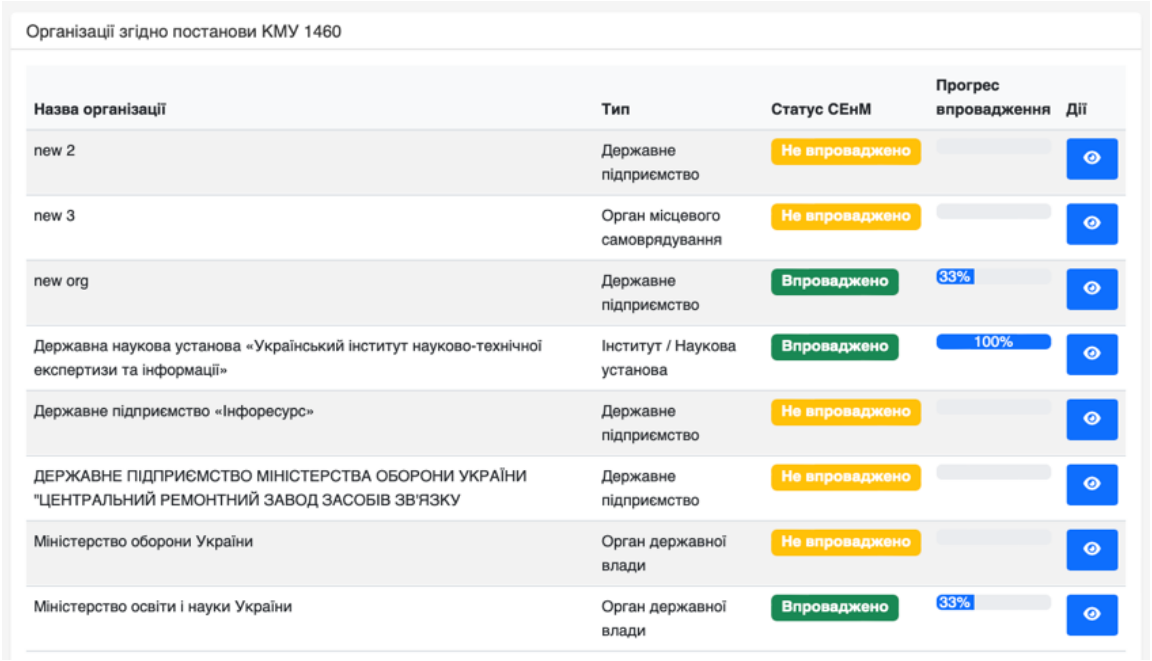

Рисунок 2.11 – Таймлайн виконання етапів

Розроблений користувацький інтерфейс забезпечує ефективну взаємодію з системою для всіх типів користувачів, надаючи зручні інструменти для введення даних, аналізу інформації та формування звітності.

3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Розробка інформаційної системи

Для реалізації інформаційної системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності обрано оптимальний набір технологій, які забезпечать надійне функціонування системи, її масштабованість, безпеку та зручність використання. Процес вибору технологій базувався на аналізі вимог до системи, особливостей предметної області та потреб користувачів:

а) для розробки back-end відповідно до технічних вимог, для серверної частини було обрано Python з фреймворком Django[36];

б) для розробки front-end відповідно до технічних вимог, для клієнтської частини було обрано наступні технології: HTML5/CSS3, JavaScript з jQuery, Bootstrap;

в) система керування базами даних використовується PostgreSQL;

г) інструменти для аналізу даних використовуються бібліотеки Python: Pandas, NumPy;

д) інструменти для візуалізації даних використовуються: Plotly, Chart.js;

е) інструменти для адміністрування та розгортання: pgAdmin, Docker та Docker Compose;

є) бібліотеки для тестування: pytest, Django Test Client.

Обрані технології повністю відповідають потребам системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, забезпечуючи надійну основу для розробки, розгортання та подальшої підтримки системи.

3.2 Реалізація серверної частини системи

Серверна частина інформаційної системи моніторингу реалізована з використанням Python та фреймворку Django. Система розроблена відповідно до архітектури MVT (Model-View-Template) та розділена на модулі, що відповідають за різні функціональні компоненти.

Структура проекту (табл. 3.1).

Таблиця 3.1 - Структура проекту

Директорія	Опис
nsme_project/	корінева директорія
nsme/	основний пакет проекту
accounts/	модуль управління користувачами
organizations/	модуль роботи з організаціями
buildings/	модуль роботи з будівлями
energy_management/	модуль систем енергоменеджменту
energy_consumption/	модуль обліку енергоспоживання
energy_efficiency/	модуль заходів з енергоефективності
analytics/	модуль аналітики та візуалізації
reporting/	модуль генерації звітів
api/	модуль API для інтеграції

Для зберігання інформації розроблено ряд моделей, які відповідають структурі бази даних. Основні моделі включають:

а) Organization - зберігає дані про організації з полями (назва, тип, контактна інформація, відповідність постанові КМУ 1460);

б) EnergyManagementSystem - містить інформацію про систему енергоменеджменту (відповідальні особи, накази про впровадження, статуси затвердження компонентів);

в) Building - описує будівлі організацій (адреса, тип, площа, технічні

характеристики);

г) EnergyConsumption - зберігає дані про споживання енергоресурсів будівль;

д) EnergyEfficiencyMeasure - містить інформацію про впроваджені заходи з енергоефективності.

Для обробки запитів користувачів реалізовано два типи представлень:

а) функціональні представлення - для простих операцій (відображення списків, деталей об'єктів);

б) класові представлення - для складніших операцій з моделями (створення, редагування, видалення).

Представлення забезпечують фільтрацію даних, пагінацію та обробку форм введення.

Для аналізу даних реалізовано набір утилітарних функцій:

- EnergyConsumption - розрахунок питомого споживання енергоресурсів;
- energy_saving - розрахунок економії енергоресурсів;
- implementation_progress - оцінка стану впровадження СЕнМ.

Захист системи забезпечується за допомогою:

- автентифікація - перевірка користувачів при вході в систему;
- авторизація - контроль доступу на основі ролей та дозволів;
- розмежування доступу - обмеження видимості даних залежно від ролі користувача.

Для моніторингу організацій, визначених постановою КМУ 1460, реалізовано:

- поле kmu_1460_compliant в моделі організації для відмітки відповідності критеріям;
- функцію kmu_1460_compliance для перевірки відповідності організації;
- спеціалізоване представлення KMU1460ReportView для формування

звітів про стан впровадження СЕнМ в організаціях, що підпадають під дію постанови.

Серверна частина забезпечує всю необхідну функціональність для моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, надаючи гнучкі інструменти для роботи з даними та їх аналізу.

3.3 Реалізація клієнтської частини системи

Клієнтська частина інформаційної системи моніторингу енергоефективності забезпечує взаємодію користувачів з системою та візуалізацію даних.

Структура клієнтської частини (табл. 3.2).

Таблиця 3.2 - Структура клієнтської частини

Директорія	Опис
nsme_project/	корінева директорія
staticfiles/	директорія статичних файлів
staticfiles/css/	стилі оформлення
staticfiles/js/	JavaScript-скрипти
staticfiles/images/	директорія для збереження зображень користувачів
staticfiles/admin/	налаштування і стилі панелі керування Django
templates/	директорія шаблонів системи
base.html	базовий шаблон
accounts/	шаблони авторизації
organizations/	шаблони для роботи з організаціями
buildings/	шаблони для роботи з будівлями
energy_management/	шаблони для СЕнМ
analytics/	шаблони аналітики
reports/	шаблони звітів

Основні компоненти інтерфейсу:

а) система авторизації (форми входу та реєстрації користувачів);

б) навігаційна панель (головне меню з доступом до всіх розділів системи, адаптоване під роль користувача);

в) форми введення даних (інтерактивні форми для додавання та редагування інформації про організації, будівлі, системи енергоменеджменту та споживання ресурсів);

г) таблиці даних (компоненти для відображення списків з можливістю фільтрації, сортування та пошуку);

д) інформаційні панелі (дашборди з ключовими показниками та інтерактивними графіками);

е) генератор звітів (формування звітів).

Клієнтська частина взаємодіє з сервером через:

– традиційний обмін формами (стандартний механізм відправки та отримання HTML-сторінок);

– AJAX-запити (асинхронне оновлення даних без перезавантаження сторінки);

– Fetch API (сучасний JavaScript API для роботи з HTTP-запитами);

– REST API (структурований інтерфейс для обміну даними у форматі JSON).

Ключові екрани системи:

а) вхід до системи (рис. 3.1);

б) реєстрація нового користувача (рис. 3.2);

в) головна сторінка (дашборд) (рис. 3.3);

г) управління організаціями (рис. 3.4);

д) системи енергоменеджменту (рис. 3.5);

е) енергоспоживання (рис. 3.6, рис. 3.7);

є) енергоефективність (рис. 3.8);

ж) аналітика (рис. 3.9);

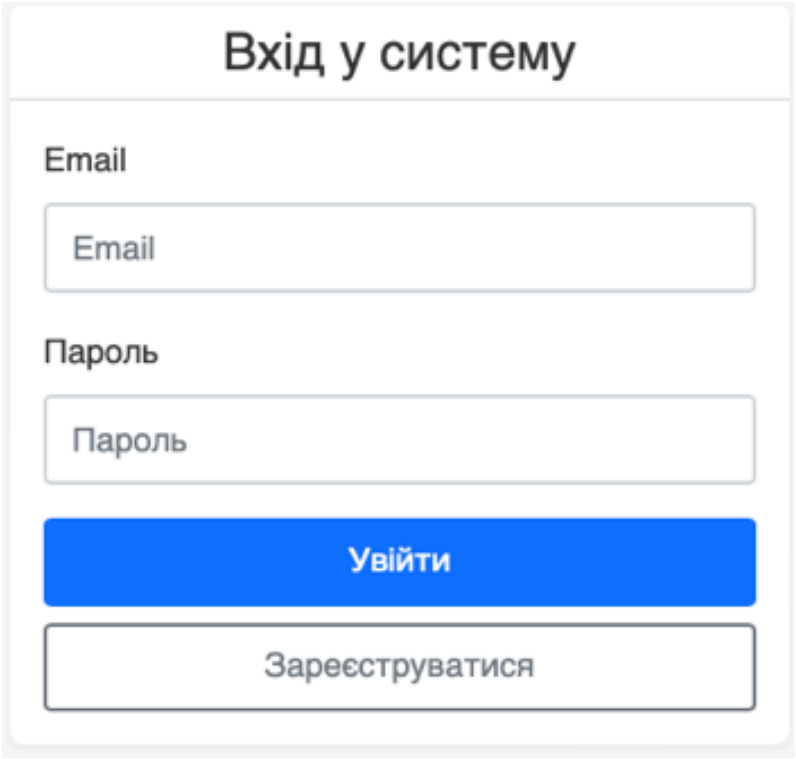

Рисунок 3.1 – Вхід до системи

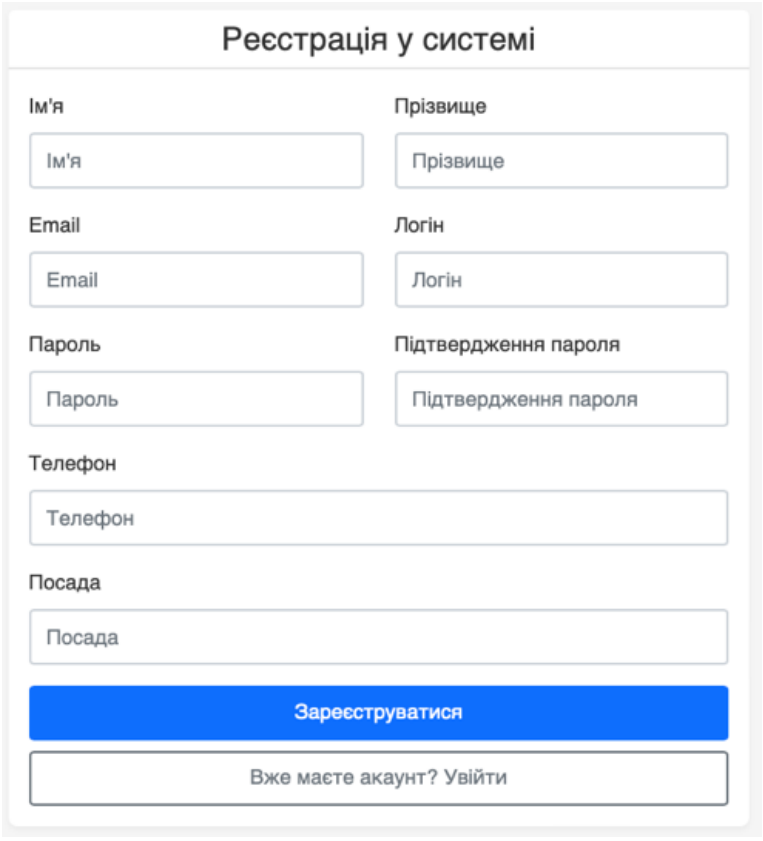

Рисунок 3.2 – Реєстрація нового користувача

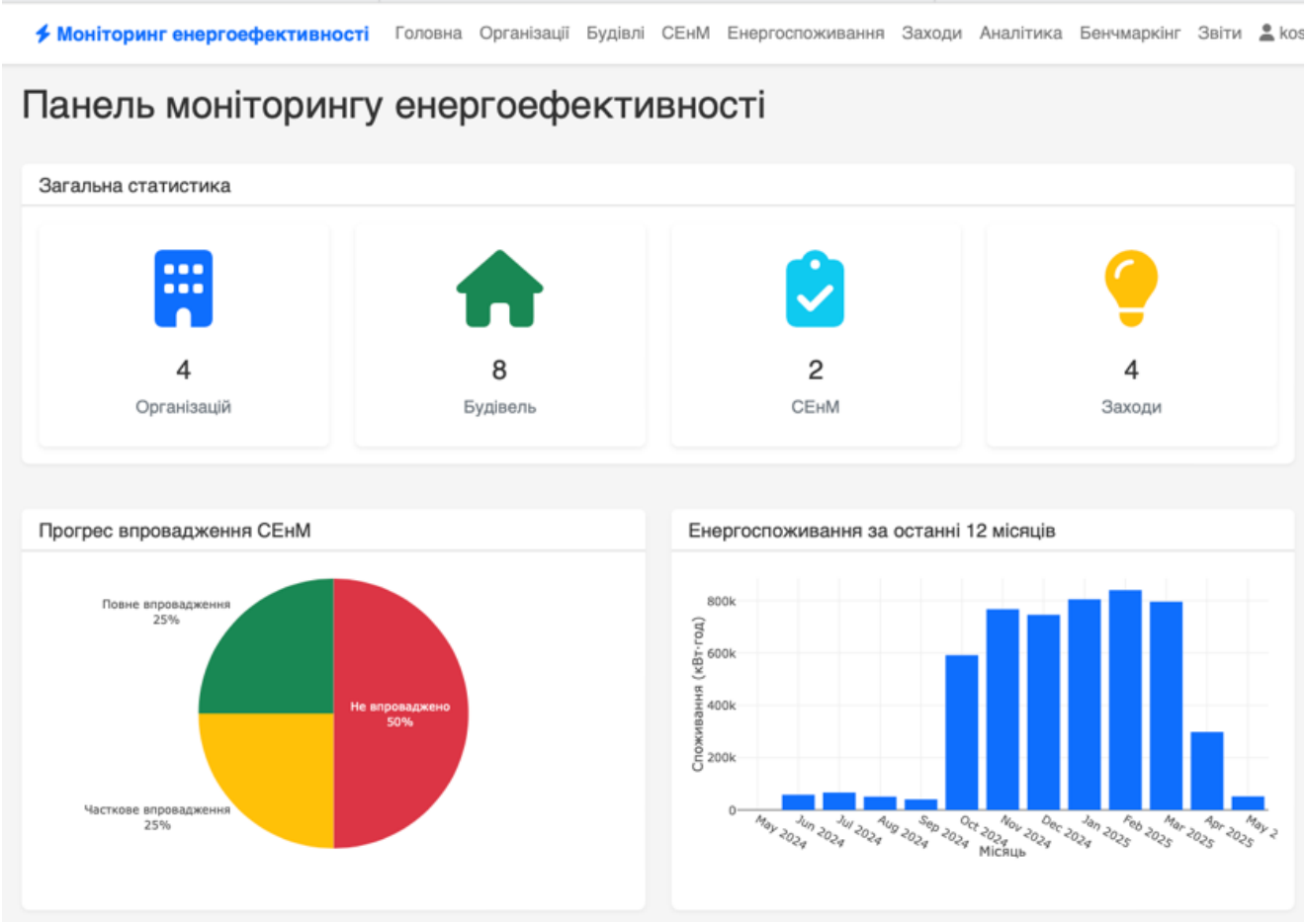

Рисунок 3.3 – Головна сторінка (дашборд)

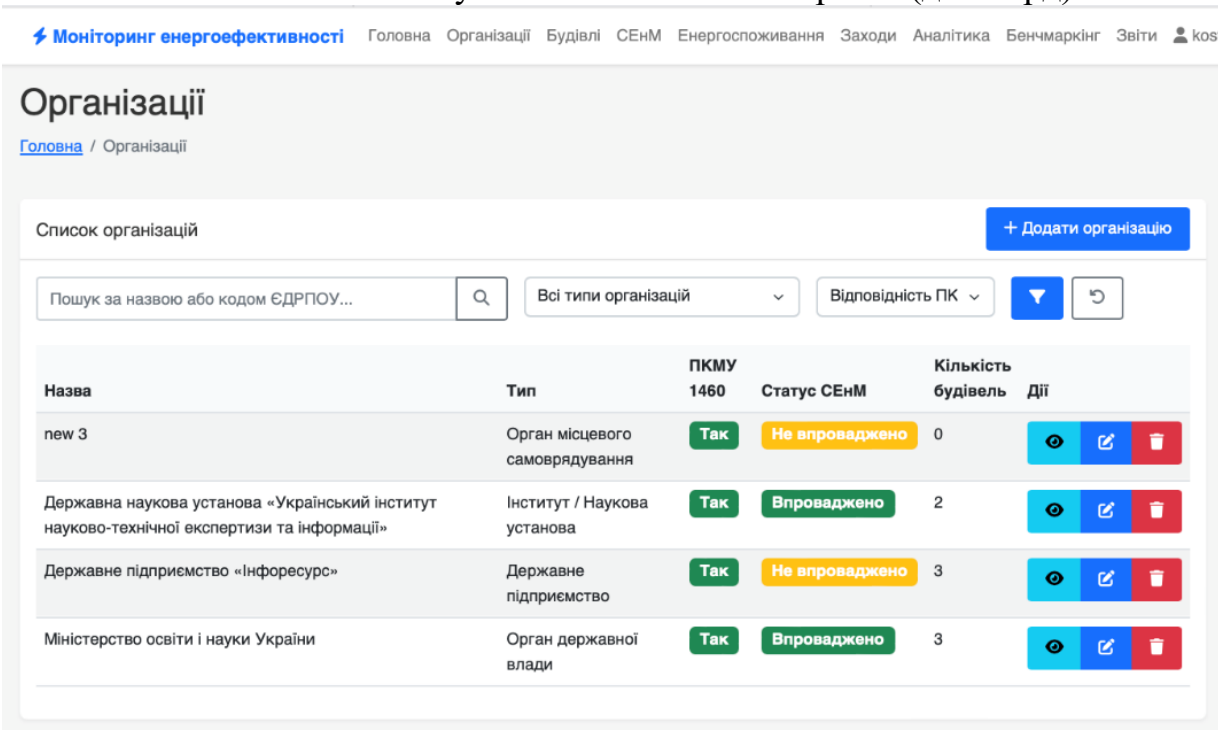

Рисунок 3.4 – Управління організаціями

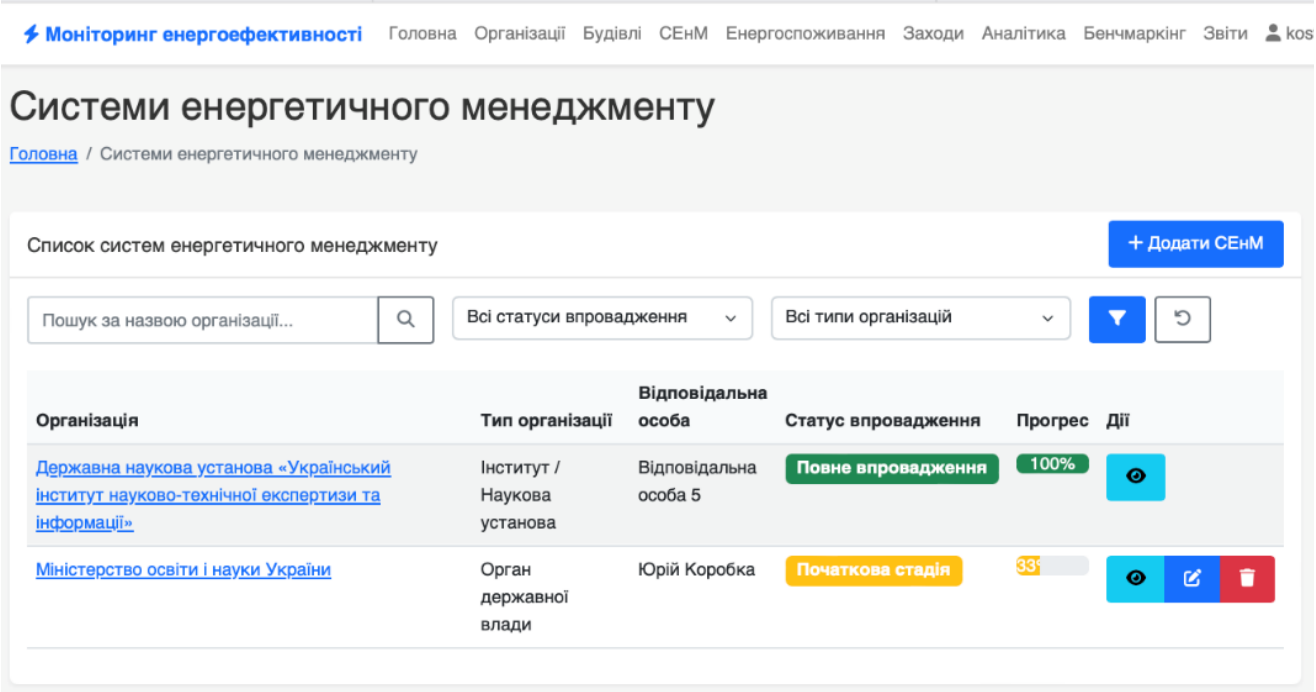

Рисунок 3.5 – Системи енергоменеджменту

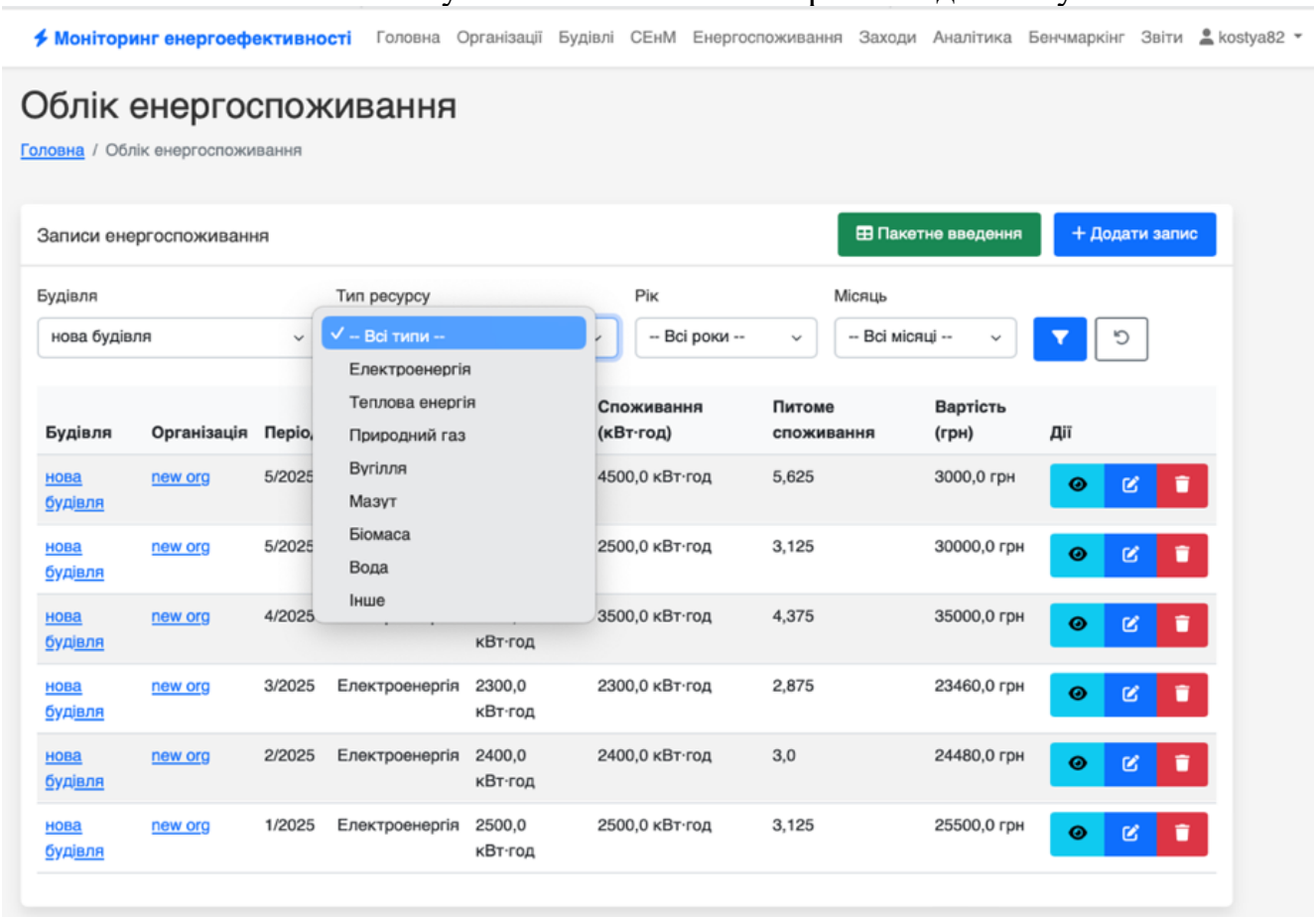

Рисунок 3.6 – Енергоспоживання

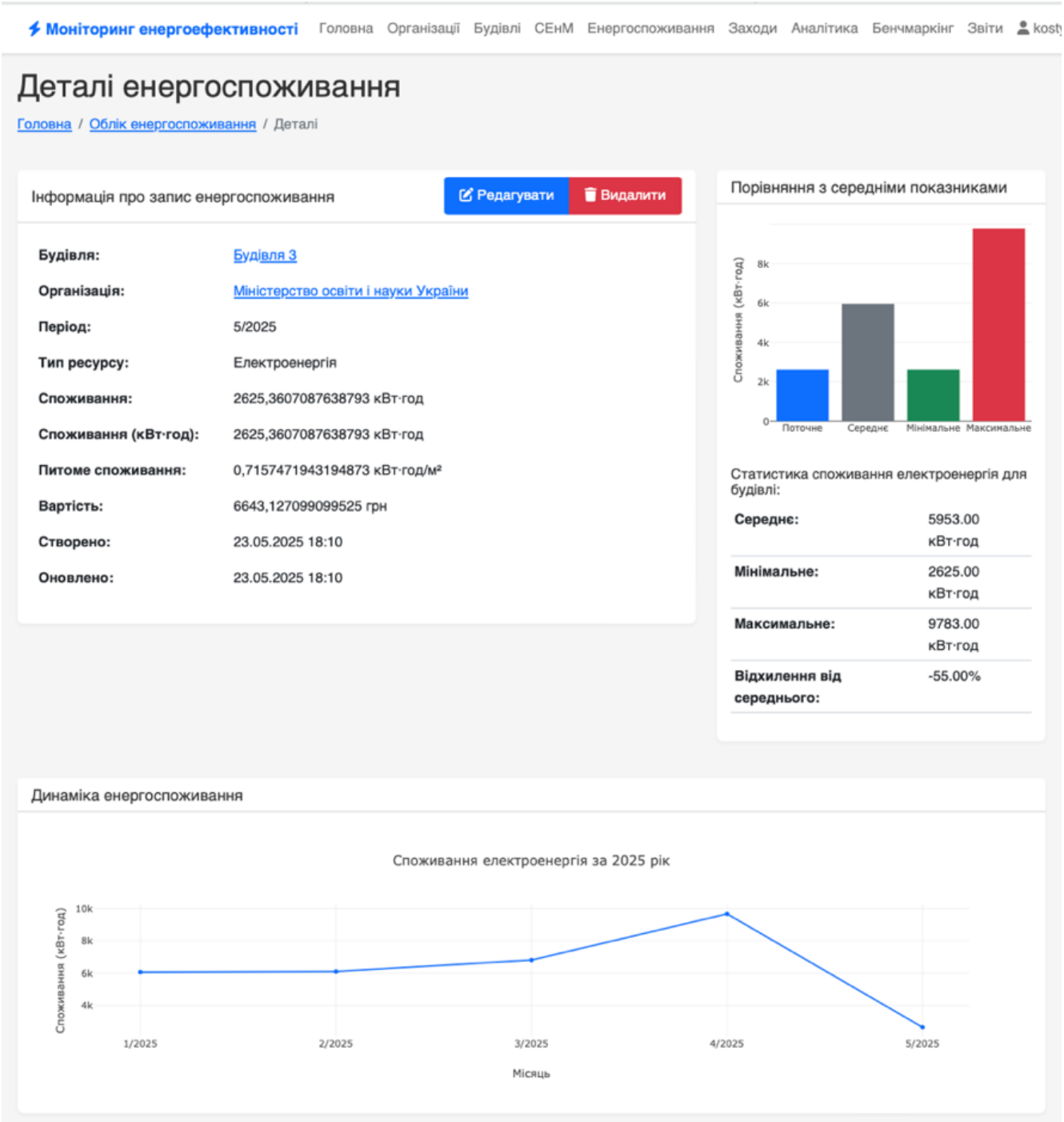

Рисунок 3.7 – Енергоспоживання

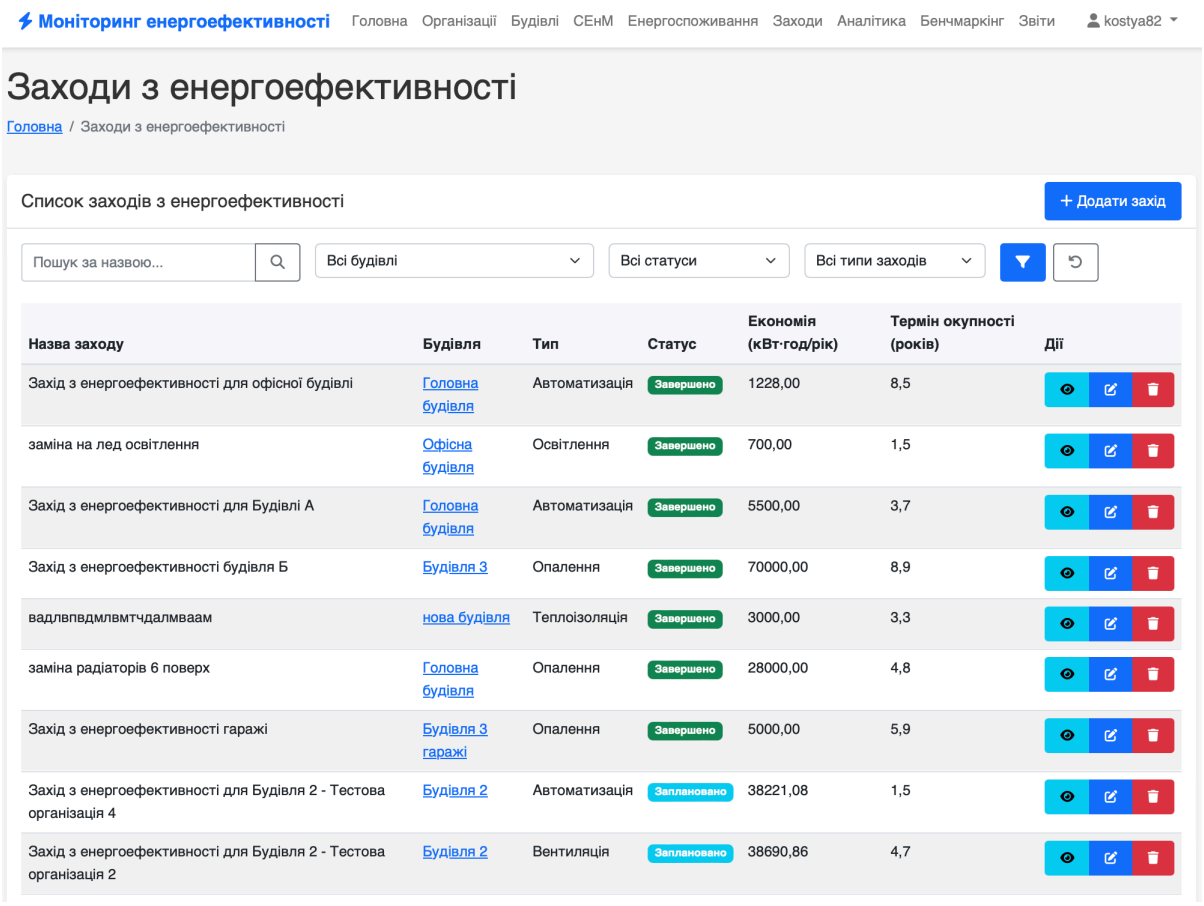

Рисунок 3.8 – Енергоефективність

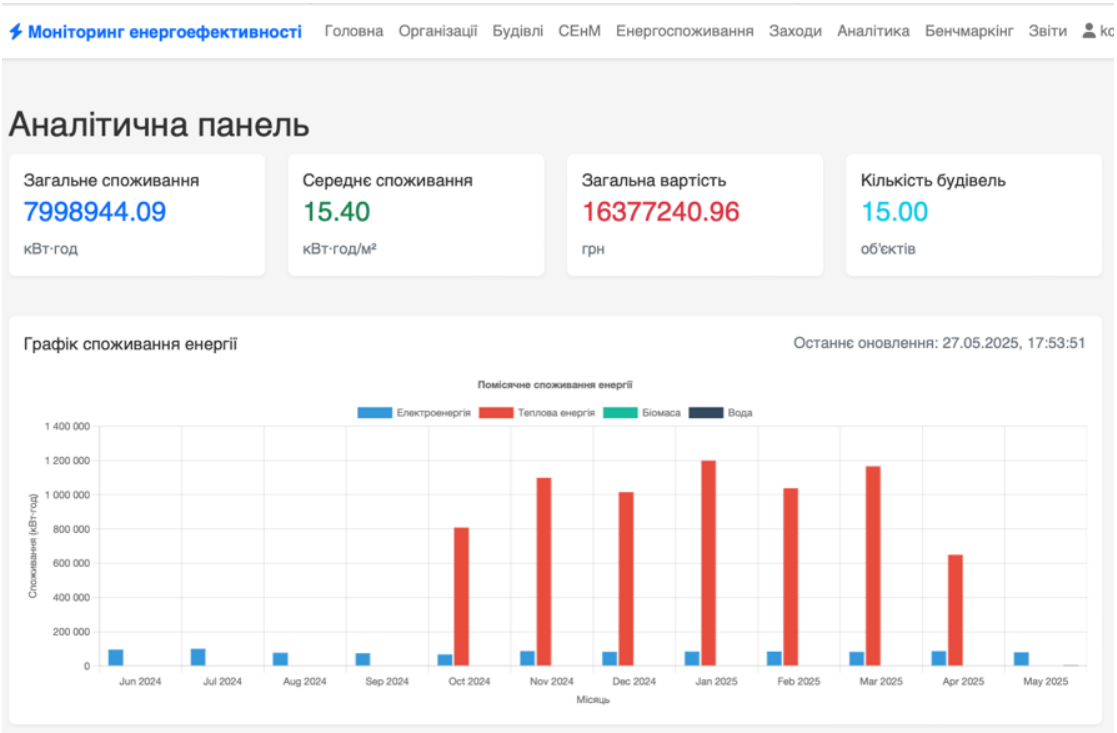

Рисунок 3.9 – Аналітика

з) звіти (рис. 3.10);

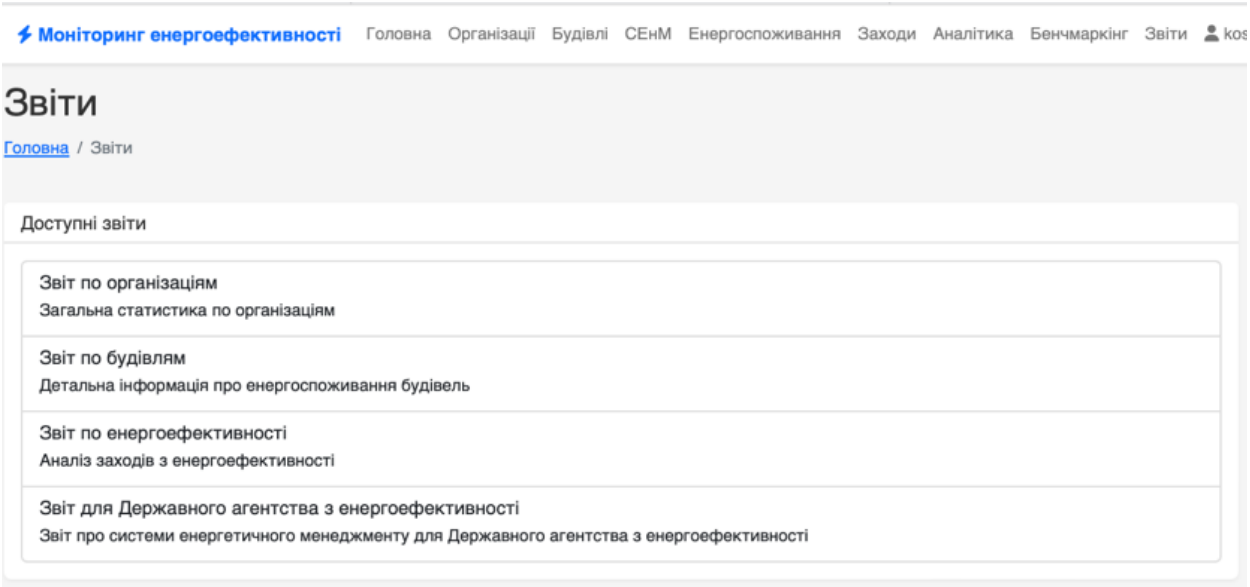

Рисунок 3.10 – Звіти

и) бенчмаркінг (рис. 3.11, рис. 3.12);

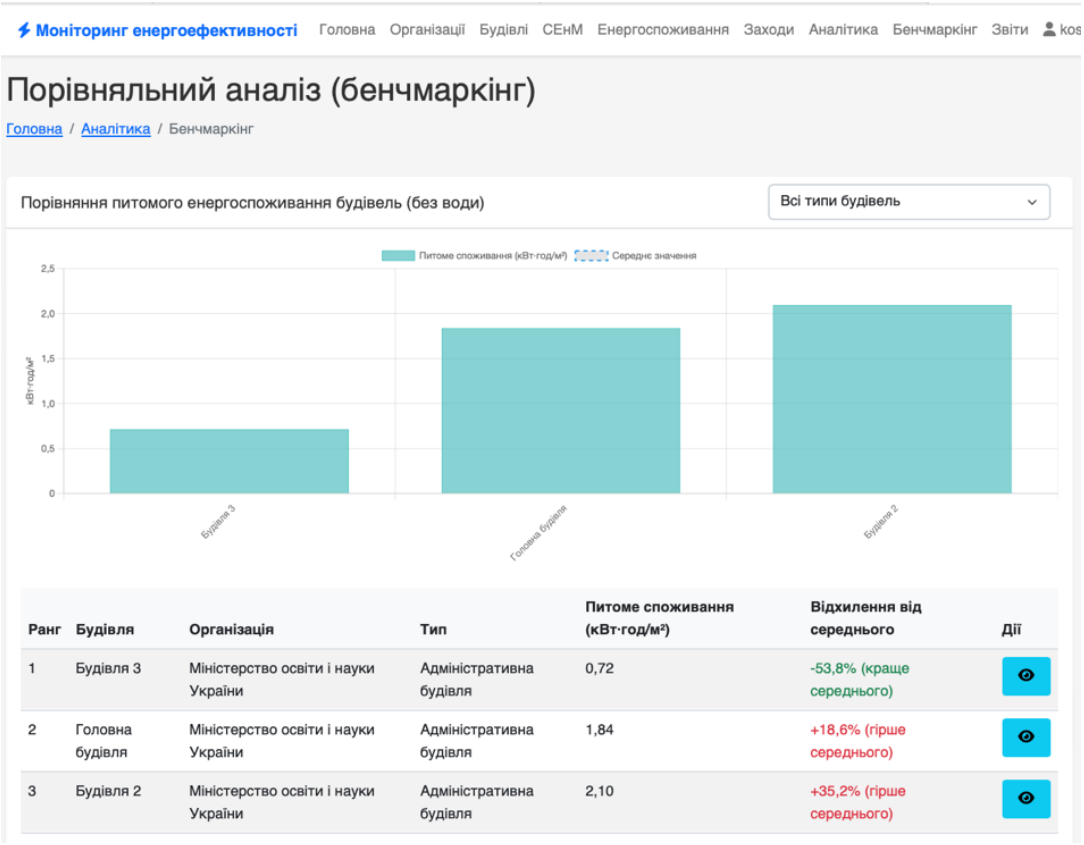

Рисунок 3.11 – Бенчмаркінг

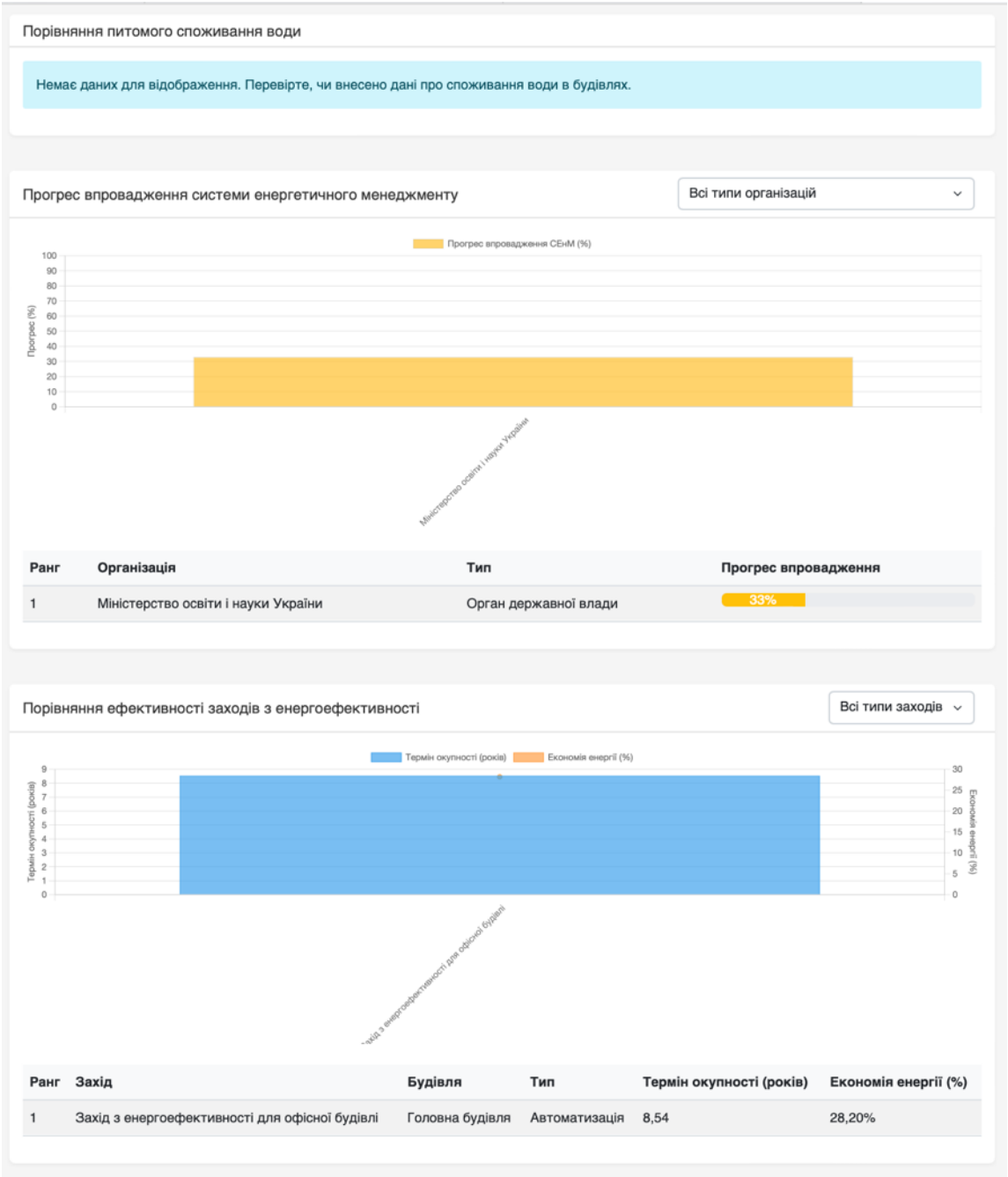

Рисунок 3.12 – Бенчмаркінг

Інтерфейс розроблено з урахуванням вимог адаптивності для різних пристроїв:

– респонсивний дизайн (коректне відображення на екранах різних розмірів);

– мобільна версія (оптимізоване відображення на смартфонах та планшетах);

– доступність (відповідність стандартам WCAG для забезпечення доступу користувачів з різними потребами).

Клієнтська частина системи забезпечує зручний інтерфейс для роботи з даними про енергоефективність, надаючи інтуїтивно зрозумілі інструменти введення, аналізу та візуалізації інформації для всіх категорій користувачів.

3.4 Реалізація алгоритмів аналізу та оцінки ефективності

Для забезпечення аналітичних можливостей інформаційної системи та оцінки ефективності впровадження енергоефективних заходів розроблено комплекс алгоритмів. Ці алгоритми реалізовані з використанням Python та його бібліотек для аналізу даних.

Алгоритми обробки та нормалізації даних забезпечують підготовку даних для аналізу через:

– очищення від некоректних значень;

– нормалізацію показників до спільних одиниць виміру;

– агрегацію даних за часовими періодами та об’єктами.

Алгоритми статистичного аналізу виконують розрахунок статистичних показників:

– середнє, медіана, мінімум, максимум споживання;

– динаміка зміни показників у часі.

Алгоритми порівняльного аналізу реалізують методи бенчмаркінгу:

– ранжування об'єктів за ефективністю;

– ранжування об'єктів за споживанням;

– ранжування об'єктів за результативністю заходів з енергоефективності.

Алгоритми оцінки ефективності заходів розраховують ефект від впроваджених заходів:

- розрахунок періоду окупності;
- оцінка скорочення викидів CO_2.

Алгоритми оцінки впровадження СЕнМ:

- відповідність вимогам постанови КМУ 1460;
- оцінка повноти впровадження компонентів СЕнМ;
- визначення ефективності функціонування СЕнМ.

Алгоритми інтегровані в інформаційну систему через:

- аналітичний модуль (статистичний аналіз та візуалізації);
- модуль оцінки ефективності (оцінка результативності заходів);
- API-інтерфейси (доступ до аналітичних функцій з клієнтської частини);
- модуль звітності (включення результатів аналізу в звіти).

Реалізовані алгоритми дозволяють ефективно оцінювати результативність енергоефективних заходів та стан впровадження систем енергетичного менеджменту, забезпечуючи основу для прийняття обґрунтованих управлінських рішень у сфері енергоефективності.

3.5 Інтеграція модулів системи

Для забезпечення ефективного функціонування інформаційної системи необхідна злагоджена взаємодія всіх її компонентів. У цьому розділі розглянуто інтеграцію модулів системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності.

Інтеграція модулів реалізована за принципом слабкого зв'язування, що забезпечує гнучкість системи та можливість незалежного розвитку окремих компонентів.

Механізми інтеграції.

Взаємодія через базу даних:

– модулі працюють з єдиною базою даних;
– ORM Django забезпечує уніфікований доступ до даних;
– транзакції для підтримки цілісності даних.

API-інтерфейси:

– REST API для взаємодії між модулями;
– стандартизовані формати даних (JSON);
– валідація даних при передачі.

Система подій:

– сигнали Django для сповіщення про зміни даних;
– асинхронні задачі для тривалих операцій;
– обробники подій для реакції на зміни в системі.

Інтеграція модулів системи забезпечує ефективну взаємодію компонентів, цілісність даних та надійне функціонування інформаційної системи моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності.

У Додатку А «Код, який використовується у проекті (python)» представлено код який представляє основні компоненти системи моніторингу енергоефективності, зокрема модулі аналітики та звітності. Код включає функції для відображення аналітичної панелі, бенчмаркінгу, API-ендпоінти для отримання даних, фонові задачі для оновлення кешу, а також класи для генерації різних типів звітів у форматах PDF та Excel.

4 ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ СИСТЕМИ

4.1 Розробка плану тестування

Для забезпечення надійності та відповідності системи вимогам розроблено комплексний план тестування. План охоплює всі ключові аспекти функціонування інформаційної системи моніторингу енергоефективності.

Рівні тестування:

- модульне тестування - перевірка окремих компонентів системи;
- інтеграційне тестування - перевірка взаємодії між компонентами;
- системне тестування - перевірка функціонування всієї системи.

Для модульного тестування розроблено набір тестів для ключових компонентів системи:

а) тести моделей даних:

- перевірка створення, оновлення та видалення об'єктів;
- валідація правил бізнес-логіки;
- тестування обмежень та залежностей.

б) тести представлень:

- перевірка правильності відображення сторінок;
- тестування обробки форм;
- перевірка контролю доступу.

в) тести API:

- перевірка API-ендпоінтів;
- тестування обробки різних типів запитів;
- валідація відповідей.

г) тести алгоритмів аналізу та оцінки:

- перевірка правильності розрахунків;
- тестування обробки граничних випадків;

– валідація результатів.

Для інтеграційного тестування розроблено сценарії, що перевіряють взаємодію між різними модулями системи:

а) тестування взаємодії модуля введення даних з аналітичним модулем:

– перевірка коректності передачі даних;

– тестування оновлення аналітики при зміні вхідних даних.

б) тестування інтеграції з модулем звітності:

– перевірка формування звітів на основі даних з різних модулів;

– тестування експорту звітів у різних форматах.

Системне тестування передбачає перевірку функціонування системи в цілому:

а) тестування основних сценаріїв використання:

– перевірка процесу реєстрації та авторизації;

– тестування введення та аналізу даних;

– формування та експорт звітів.

б) тестування взаємодії з різними ролями користувачів:

– перевірка доступу та функціональності для різних ролей;

– тестування обмежень та дозволів.

в) для перевірки захищеності системи заплановано:

– перевірка механізмів контролю доступу;

– тестування захисту від неавторизованого доступу.

4.2. Проведення функціонального тестування

Функціональне тестування є ключовим етапом перевірки відповідності системи поставленим вимогам. Воно дозволяє підтвердити, що всі функції системи працюють коректно та відповідають очікуваній поведінці.

Функціональне тестування проводилось за методологією тестування на основі вимог (Requirements-Based Testing). Кожна функціональна вимога була перетворена на набір тестових випадків, які перевіряють її коректну реалізацію.

Тестові сценарії та результати приведені у табл. 4.1.

Таблиця 4.1 - Тестові сценарії та результати

№	Тестовий сценарій	Очікуваний результат	Фактичний результат	Статус
1	Реєстрація нового користувача	Створення облікового запису, відправка повідомлення	Користувач створений, повідомлення відправлено	Успіх
2	Вхід користувача з коректними даними	Успішна автентифікація, перенаправлення на головну сторінку	Користувач автентифікований, перенаправлений	Успіх
3	Вхід з неправильними обліковими даними	Повідомлення про помилку	Виведено повідомлення про невірні дані	Успіх
4	Зміна паролю користувача	Оновлення паролю, повідомлення про успіх	Пароль оновлено, повідомлення виведено	Успіх
5	Призначення ролі користувачу	Оновлення прав доступу	Права оновлено	Успіх
6	Створення нової організації	Додавання запису в БД, відображення у списку	Організація створена, відображається	Успіх
7	Редагування даних організації	Оновлення даних у БД	Дані оновлено	Успіх
8	Відмітка відповідності постанові КМУ 1460	Зміна прапорця в БД	Прапорець оновлено	Успіх
9	Фільтрація списку організацій	Відображення лише відповідних критеріям	Список відфільтровано	Успіх
10	Видалення організації	Видалення запису з БД	Запис видалено	Успіх
11	Створення запису про СЕнМ	Додавання запису в БД	Запис створено	Успіх
12	Визначення статусу впровадження	Коректний розрахунок статусу	Статус визначено правильно	Успіх
13	Оновлення статусів впровадження	Зміна прапорців у БД	Прапорці оновлено	Успіх
14	Завантаження документів СЕнМ	Збереження файлів на сервері	Файли збережено	Успіх
15	Фільтрація за статусом впровадження	Відображення відповідних записів	Записи відфільтровано	Успіх
16	Додавання будівлі	Створення запису в БД	Будівля додана	Успіх
17	Введення даних про споживання	Додавання записів у БД	Дані додано	Успіх
18	Розрахунок питомого споживання	Коректний розрахунок значень	Значення розраховано правильно	Успіх
19	Розрахунок статистичних показників	Коректні значення показників	Показники розраховано правильно	Успіх
20	Побудова графіків споживання	Відображення графіків	Графіки побудовано	Успіх
21	Порівняльний аналіз будівель	Формування рейтингу ефективності	Рейтинг сформовано	Успіх

Кінець таблиці 4.1

22	Аналіз динаміки споживання	Розрахунок тенденцій	Тенденції визначено	Успіх
23	Розрахунок економії енергоресурсів	Коректні значення економії	Економія розрахована правильно	Успіх
24	Оцінка ефективності заходів	Розрахунок показників ефективності	Показники розраховано	Успіх
25	Визначення терміну окупності	Коректний розрахунок окупності	Термін розраховано правильно	Успіх
26	Розрахунок скорочення викидів CO_2	Коректні значення скорочення	Значення розраховано правильно	Успіх
27	Формування звіту про СЕнМ	Створення звіту з коректними даними	Звіт сформовано	Успіх
28	Експорт звіту в PDF	Створення PDF-файлу	Файл створено	Успіх
29	Формування звіту по постанові КМУ 1460	Створення звіту з показниками відповідності	Звіт сформовано	Успіх
30	Отримання списку організацій	Повернення JSON з даними	Дані отримано	Успіх
31	Додавання даних про споживання	Створення запису через API	Запис створено	Успіх
32	Оновлення даних про СЕнМ	Оновлення запису через API	Запис оновлено	Успіх
33	Фільтрація даних через API	Повернення відфільтрованих записів	Дані відфільтровано	Успіх

За результатами функціонального тестування:

- проведено 33 основних тестових сценарія;
- всі функціональні вимоги системи успішно протестовано.

Функціональне тестування підтвердило відповідність системи поставленим вимогам та її готовність до використання для моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності.

ВИСНОВКИ

У результаті виконання кваліфікаційної бакалаврської роботи розроблено інформаційну систему моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, яка забезпечує повний цикл моніторингу від збору первинних даних до формування аналітичних звітів для організацій, що підпадають під дію постанови КМУ 1460 від 2021 року.

У першому розділі роботи проведено комплексний аналіз предметної області, який виявив суттєві прогалини в існуючих підходах до моніторингу державних програм енергоефективності в Україні. Як показано в розділі 1.1, на даний момент відсутня єдина інформаційна система для моніторингу всіх державних програм у сфері енергоефективності, наявні лише окремі елементи систем моніторингу, які функціонують на різних рівнях без належної автоматизації та уніфікації даних.

Аналіз вітчизняного досвіду (розділ 1.2) вказує що на даний момент в Україні функціонує кілька розрізнених систем моніторингу енергоефективності, які не утворюють єдиного інформаційного простору, аналіз зарубіжного досвіду (розділ 1.3) продемонстрував ефективність комплексних систем моніторингу з високим рівнем автоматизації, таких як ENERGY STAR Portfolio Manager (США) та Display Energy Certificates System (Великобританія) що підтвердило актуальність розробки подібного рішення для України.

Вивчення методів оцінки ефективності програм у сфері енергоефективності (розділ 1.4) дозволило сформувати комплексний підхід до оцінювання, що поєднує методи на основі ISO 50001:2020, бенчмаркінгу енергетичної ефективності, економічної та екологічної оцінки. Особливу увагу приділено методам, специфічним для оцінки впровадження систем енергетичного менеджменту (СЕнМ) згідно постанови КМУ 1460, що стало основою для розробки алгоритмів оцінки ефективності, описаних у розділі 3.4.

На основі проведеного аналізу сформовано детальні функціональні та нефункціональні вимоги до інформаційної системи (розділ 1.3), що охоплюють всі аспекти її роботи: від збору та валідації даних до генерації звітів і інтеграції з іншими системами. Розроблено специфічні вимоги для моніторингу організацій згідно постанови КМУ 1460, а також визначено вимоги до різних користувацьких ролей, що забезпечило відповідність системи реальним потребам цільових користувачів.

У розділі 2.1 розроблено багаторівневу архітектуру інформаційної системи, яка включає рівень даних, серверний рівень, клієнтський рівень та інтеграційний рівень. Запропонована модульна структура системи з чітким розподілом відповідальності між компонентами забезпечує гнучкість, масштабованість та надійність рішення. Використання контейнеризації з Docker та Docker Compose, спрощує розгортання системи в різних середовищах та її подальше масштабування.

Спроектовано структуру бази даних PostgreSQL (розділ 2.2), що включає 10 основних таблиць для зберігання інформації про користувачів, організації, системи енергоменеджменту, будівлі, енергоспоживання, заходи з енергоефективності, показники ефективності та звіти. Розроблена структура враховує вимоги Додатку 1 до Листа про впровадження СЕнМ, що забезпечує повну відповідність системи документообігу в цій сфері. Реалізовано механізми індексування, обмеження цілісності даних, тригери та функції для забезпечення надійного зберігання та ефективного доступу до даних.

Детально розроблено три ключові модулі системи:

– модуль внесення та збору даних (розділ 2.3.1), який забезпечує ручне введення даних через веб-інтерфейс та комплексну верифікацію і валідацію інформації відповідно;

– аналітичний модуль (розділ 2.3.2), який реалізує статистичну обробку даних, порівняльний аналіз, виявлення тенденцій та аномалій, а також візуалізацію результатів;

– модуль оцінки ефективності (розділ 2.3.3), який розраховує показники енергетичної результативності, проводить економічну та екологічну оцінку, формує оцінку ефективності.

У розділі 2.4 спроектовано інтуїтивно зрозумілий користувацький інтерфейс, який адаптований для різних ролей користувачів (адміністратор системи, адміністратор організації, користувач організації, енергоаудитор). Інтерфейс розроблено з дотриманням принципів інтуїтивності, узгодженості, ефективності, адаптивності та доступності. Особливу увагу приділено інтерфейсам для моніторингу згідно постанови КМУ 1460, включаючи спеціалізовані форми для введення даних та індикатори відповідності вимогам постанови.

Для реалізації інформаційної системи обрано сучасний стек технологій (розділ 3.1), який включає Python з фреймворком Django для серверної частини, HTML5/CSS3, JavaScript з jQuery та Bootstrap для клієнтської частини, PostgreSQL для бази даних, а також спеціалізовані бібліотеки Python для аналізу даних (Pandas, NumPy, SciPy) та візуалізації (Chart.js, Plotly). Використання Docker та Docker Compose забезпечує контейнеризацію компонентів системи та спрощує її розгортання.

У розділах 3.2 та 3.3 описано реалізацію серверної та клієнтської частин системи відповідно. Серверна частина реалізована на основі архітектури MVT Django з модульною структурою, що відповідає функціональним компонентам системи. Клієнтська частина забезпечує зручну взаємодію користувачів з системою через адаптивний інтерфейс з використанням сучасних JavaScript-бібліотек для створення інтерактивних елементів та візуалізацій.

У розділі 3.4 детально описано реалізацію алгоритмів аналізу та оцінки ефективності, які є ключовими для функціонування системи. Розроблено алгоритми обробки та нормалізації даних, статистичного аналізу, виявлення аномалій, порівняльного аналізу, оцінки ефективності заходів та оцінки впровадження СЕнМ. Наведено конкретні реалізації алгоритмів мовою Python, що дозволяють, наприклад, розраховувати питоме споживання енергоресурсів, оцінювати економічну ефективність заходів та стан впровадження СЕнМ.

У розділі 3.5 описано інтеграцію модулів системи, яка реалізована за принципом слабкого зв'язування (loose coupling), що забезпечує гнучкість системи та можливість незалежного розвитку окремих компонентів. Інтеграція здійснюється через взаємодію з єдиною базою даних, API-інтерфейси та систему подій з використанням сигналів Django.

Розроблено план тестування (розділ 4.1), проведено функціональне тестування (розділ 4.2), яке підтвердило відповідність системи поставленим вимогам. Протестовано всі ключові функції системи: від реєстрації користувачів до формування звітів та взаємодії через API.

Розроблена інформаційна система представляє собою комплексне рішення для моніторингу та оцінки ефективності державних цільових програм у сфері енергоефективності, з особливим фокусом на організації, визначені постановою КМУ 1460 від 2021 року. Система забезпечує повний цикл моніторингу: від збору первинних даних до формування аналітичних звітів та оцінки ефективності впроваджених заходів. Реалізовані функціональні можливості системи відповідають усім поставленим вимогам та забезпечують ефективне вирішення задачі моніторингу енергоефективності.

Практична значимість роботи полягає в можливості впровадження розробленої системи в державних органах та підприємствах для забезпечення ефективного моніторингу виконання постанови КМУ 1460 від 2021 року, що сприятиме підвищенню прозорості реалізації програм енергоефективності,

обґрунтованому прийняттю управлінських рішень та загальному підвищенню енергетичної ефективності в державному секторі України.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Постанова Кабінету Міністрів України від 23.12.2021 р. № 1460 «Про впровадження систем енергетичного менеджменту» [Електронний ресурс]. Режим доступу: https://www.kmu.gov.ua/npas/pro-vprovadzhennya-sistem-energeti-a1460

2. Розпорядження КМУ від 21.04.2024 № 373-р. «По схвалення Енергетичної стратегії України на період до 2050 року». [Електронний ресурс]. Режим доступу: https://zakon.rada.gov.ua/laws/show/373-2023-р#Text

3. Розпорядження КМУ від 29.01.2020 № 88-р. Про схвалення Концепції реалізації державної політики у сфері забезпечення енергетичної ефективності будівель у частині збільшення кількості будівель з близьким до нульового рівнем споживання енергії та затвердження Національного плану збільшення кількості будівель з близьким до нульового рівнем споживання енергії. [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/88-2020-р#Text

4. Державне агентство з енергоефективності та енергозбереження України [Електронний ресурс]. – Режим доступу: https://saee.gov.ua/

5. Держенергоефективності: Все більше органів влади в Україні впроваджують систему енергетичного менеджменту [Електронний ресурс]. – Режим доступу: https://www.kmu.gov.ua/news/derzhenerhoefektyvnosti-vse-bilshe-orhaniv-vlady-v-ukraini-vprovadzhuiut-systemu-enerhetychnoho-menedzhmentu

6. Запровадження енергетичного менеджменту в бюджетних установах [Електронний ресурс]. – Режим доступу: https://saee.gov.ua/static-objects/saee/imported_content/679269b35033d.pdf

7. Деякі питання забезпечення функціонування національної бази даних енергетичних та експлуатаційних характеристик будівель: Постанова КМУ від 01.11.2024 № 1254. [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/1254-2024-п#Text

8. Система енергоменеджменту та інтелектуальна автоматизована система енергомоніторингу АІС "ЕНЕРГОСЕРВІС". BisSoft. [Електронний ресурс]. – Режим доступу: https://bissoft.org/ua/energoservis

9. ENERGY STAR Portfolio Manager (США) [Електронний ресурс]. – Режим доступу: https://www.energystar.gov/buildings/benchmark

10. Portfolio Manager Web Services. ENERGY STAR (США) [Електронний ресурс]. – Режим доступу: https://portfoliomanager.energystar.gov/webservices/home

11. Display Energy Certificates System (Великобританія) [Електронний ресурс]. – Режим доступу: https://display-energy-certificates.co.uk

12. A guide to display energy certificates and advisory reports for public buildings. UK Government. [Електронний ресурс]. – Режим доступу: https://www.gov.uk/government/publications/display-energy-certificates-and-advisory-reports-for-public-buildings

13. Display Energy Certificate. Wikipedia. [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Display_Energy_Certificatem

14. EU Energy Efficiency Reporting System (ЄС) [Електронний ресурс]. – Режим доступу: https://energy.ec.europa.eu/topics/energy-efficiency/energy-efficiency-targets-directive-and-rules/energy-efficiency-directive_en

15. Energy union strategy. European Commission. [Електронний ресурс]. – Режим доступу: https://energy.ec.europa.eu/strategy/energy-union_en

16. ДСТУ ISO 50001:2020 (ISO 50001:2018, IDT) Системи енергетичного менеджменту. Вимоги та настанова щодо використання [Електронний ресурс]. – Режим доступу: https://online.budstandart.com/ua/catalog/doc-page?id_doc=60876

17. Ключові показники ефективності (Key Performance Indicators - KPIs). [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Ключові_показники_ефективності

18. Енергоефективність будівель в Україні - перспективи та плани поліпшення. [Електронний ресурс]. – Режим доступу: https://eenergy.com.ua/energoefectyvnist/energoefektyvnist-budivel-v-ukrayini/

19. Your Guide to Implementing ISO 50001. [Електронний ресурс]. – Режим доступу: https://www.nqa.com/en-gb/certification/standards/iso-50001/implementation

20. What is an Energy Performance Indicator (EnPI)? [Електронний ресурс]. – Режим доступу: https://50001store.com/articles/energy-performance-indicator-enpi/

21. 50001 Ready | Task 11: Energy Performance Indicators and Energy Baselines [Електронний ресурс]. – Режим доступу: https://navigator.lbl.gov/guidance/task/11

22. The Economics of Energy Efficiency: Cost-Benefit Analysis and ROI [Електронний ресурс]. – Режим доступу: https://www.linkedin.com/pulse/economics-energy-efficiency-cost-benefit-analysis-roi-prakash-?

23. Methodology for Evaluating Commercial Energy Code Updates [Електронний ресурс]. – Режим доступу: https://www.energycodes.gov/sites/default/files/2024-10/Commercial_Cost_Effective_Method_2024.pdf

24. Energy Management Information Systems Technical Resources Report [Електронний ресурс]. – Режим доступу: https://www.energy.gov/sites/default/files/2021-09/emis-technical-resources.pdf

25. Технічні вимоги до створення Національної бази даних енергетичних та експлуатаційних характеристик будівель у складі Реєстру будівельної діяльності Єдиної державної електронної системи у сфері будівництва [Електронний ресурс]. – Режим доступу: https://eef.org.ua/wp-content/uploads/2023/09/TV_NBDAEE_BBU.pdf

26. Django (web framework) — Wikipedia** [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Django

27. Bootstrap Framework Documentation [Електронний ресурс]. – Режим доступу: https://getbootstrap.com/docs/

28. Контейнери та Docker: що має знати сучасний прогер? / itProger [Електронний ресурс]. – Режим доступу: https://itproger.com/ua/news/konteyneri-i-docker-chto-dolzhen-znat-sovremenniy-proger

29. PostgreSQL Official Documentation** [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/

30. Data Visualization with Python Using Matplotlib / Sharp Coder Blog [Електронний ресурс]. – Режим доступу: https://www.sharpcoderblog.com/blog/data-visualization-with-python-using-matplotlib

31. Plotly Python Graphing Library [Електронний ресурс]. – Режим доступу: https://plotly.com/python/

32. Чому вам слід використовувати NumPy / DevZone [Електронний ресурс]. – Режим доступу: https://devzone.org.ua/post/chomu-vam-slid-vykorystovuvaty-numpy

33. SciPy Official Website [Електронний ресурс]. – Режим доступу: https://scipy.org/

34. Найкращі бібліотеки Python для 2023 Року: Огляд та Особливості / SpaceLab [Електронний ресурс]. – Режим доступу: https://spacelab.ua/articles/biblioteki-python-u-2023-roci/

35. JavaScript Chart.js. Уроки для початківців. W3Schools українською [Електронний ресурс]. – Режим доступу: https://w3schoolsua.github.io/js/js_graphics_chartjs.html#gsc.tab=0

36. Веб-розробка з Python та Django для Початківців / Leanpub [Електронний ресурс]. – Режим доступу: https://leanpub.com/djangofornewbie/read

ДОДАТОК А Код, який використовується у проекті (python)

Модуль Аналітики

Представлення аналітичної панелі (analytics/views.py):

```
from django.shortcuts import render
from django.contrib.auth.decorators import login_required
from rest_framework.test import APIClient
from django.contrib.auth import get_user_model
from django.db.models import Sum, Avg, Q
from energy_consumption.models import EnergyConsumption
from buildings.models import Building
from organizations.models import Organization
from energy_efficiency.models import EnergyEfficiencyMeasure
from django.views.generic import TemplateView
from django.contrib.auth.mixins import LoginRequiredMixin
from datetime import datetime, timedelta

User = get_user_model()

def get_data_from_api(user, endpoint, params=None):
    """Отримання даних через API для внутрішніх запитів."""
    client = APIClient()
    client.force_authenticate(user=user)
    url = f'/api/{endpoint}/'

    if params:
        query_string = '&'.join([f'{key}={value}' for key, value in params.items()])
```

```
        url = f'{url}?{query_string}'

    response = client.get(url)
    return response.data

@login_required
def analytics_dashboard(request):
    """Відображення аналітичної панелі з даними про енергоспоживання."""
    # Отримання даних за останні 12 місяців
    current_date = datetime.now()
    current_year = current_date.year
    current_month = current_date.month

    # Розрахунок початкового року та місяця
    start_year = current_year - 1 if current_month == 12 else current_year
    start_month = current_month + 1 if current_month < 12 else 1

    # Фільтрація даних по організації користувача, якщо він не адміністратор системи
    buildings = Building.objects.all()
    if not request.user.is_superuser:
        if hasattr(request.user, 'organization') and request.user.organization:
            # Отримуємо ID організації користувача та ID дочірніх організацій
            user_org_id = request.user.organization.id
            child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)
```

```
            # Фільтруємо будівлі за організацією користувача та дочірніми організаціями
            buildings = buildings.filter(organization_id__in=[user_org_id] + list(child_orgs))
        else:
            # Якщо у користувача немає організації, не показуємо жодної будівлі
            buildings = buildings.none()

    # Створення умов для фільтрації за роком та місяцем
    date_conditions = Q()
    if start_year == current_year:
        # Якщо дані в межах одного року
        date_conditions = Q(year=current_year, month__range=(start_month, current_month))
    else:
        # Якщо дані охоплюють два роки
        date_conditions = (
            Q(year=start_year, month__gte=start_month) |
            Q(year=current_year, month__lte=current_month)
        )

    # Отримання даних про енергоспоживання
    consumption_data = EnergyConsumption.objects.filter(
        building__in=buildings
    ).filter(date_conditions)

    # Загальне споживання енергії
```

```
    total_consumption =
consumption_data.aggregate(total=Sum('consumption_kwh'))['total'] or 0

    # Середнє споживання на квадратний метр
    avg_consumption_per_m2 = consumption_data.aggregate(
        avg=Avg('specific_consumption')
    )['avg'] or 0

    # Загальна вартість
    total_cost = consumption_data.aggregate(total=Sum('cost'))['total'] or 0

    context = {
        'title': 'Аналітична панель',
        'total_consumption': round(total_consumption, 2),
        'avg_consumption_per_m2': round(avg_consumption_per_m2, 2),
        'total_cost': round(total_cost, 2),
        'buildings_count': buildings.count(),
    }
    return render(request, 'analytics/dashboard.html', context)
```

Представлення для бенчмаркінгу (analytics/views.py):

```
class BenchmarkingView(LoginRequiredMixin, TemplateView):
    """Представлення для сторінки бенчмаркінгу."""
    template_name = 'analytics/benchmarking.html'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
```

```
        # Фільтрація за типом будівель
        building_type = self.request.GET.get('building_type', '')

        # Порівняння питомого енергоспоживання будівель
        buildings_query = Building.objects.all()
        if building_type:
            buildings_query = buildings_query.filter(building_type=building_type)

        # Обмеження доступу за організацією користувача
        if not self.request.user.is_superuser:
            if hasattr(self.request.user, 'organization') and self.request.user.organization:
                # Отримуємо ID організації користувача та ID дочірніх організацій
                user_org_id = self.request.user.organization.id
                child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)

                # Фільтруємо будівлі за організацією користувача та дочірніми організаціями
                buildings_query = buildings_query.filter(organization_id__in=[user_org_id] + list(child_orgs))
            else:
                # Якщо у користувача немає організації, не показуємо жодної будівлі
                buildings_query = buildings_query.none()

        # Отримання даних про питоме енергоспоживання
        buildings_data = []
```

```
        buildings_water_data = []

        for building in buildings_query:
            # Отримання останніх даних про споживання енергоресурсів (крім води)
            latest_energy_consumption = EnergyConsumption.objects.filter(
                building=building
            ).exclude(
                resource_type=EnergyConsumption.WATER
            ).order_by('-year', '-month').first()

            # Отримання останніх даних про споживання води
            latest_water_consumption = EnergyConsumption.objects.filter(
                building=building,
                resource_type=EnergyConsumption.WATER
            ).order_by('-year', '-month').first()

            # Додавання даних про енергоспоживання
            if latest_energy_consumption and latest_energy_consumption.specific_consumption:
                buildings_data.append({
                    'id': building.id,
                    'name': building.name,
                    'organization': building.organization.name,
                    'type': building.get_building_type_display(),
                    'area': building.heated_area,
                    'specific_consumption': latest_energy_consumption.specific_consumption
```

```
            })

        # Додавання даних про споживання води
        if latest_water_consumption and latest_water_consumption.specific_consumption:
            buildings_water_data.append({
                'id': building.id,
                'name': building.name,
                'organization': building.organization.name,
                'type': building.get_building_type_display(),
                'area': building.heated_area,
                'specific_consumption': latest_water_consumption.specific_consumption
            })

    # Сортування за питомим споживанням
    buildings_data.sort(key=lambda x: x['specific_consumption'])
    buildings_water_data.sort(key=lambda x: x['specific_consumption'])

    # Розрахунок середнього значення для порівняння енергоспоживання
    if buildings_data:
        avg_consumption = sum(b['specific_consumption'] for b in buildings_data) / len(buildings_data)
    else:
        avg_consumption = 0

    # Розрахунок середнього значення для порівняння споживання води
    if buildings_water_data:
```

```
            avg_water_consumption = sum(b['specific_consumption'] for b in buildings_water_data) / len(buildings_water_data)
        else:
            avg_water_consumption = 0

        import json

        # Підготовка JSON даних для безпечної передачі в шаблон
        context['buildings_data'] = buildings_data
        context['buildings_data_json'] = json.dumps(buildings_data)

        context['buildings_water_data'] = buildings_water_data
        context['buildings_water_data_json'] = json.dumps(buildings_water_data)

        context['avg_consumption'] = avg_consumption
        context['avg_water_consumption'] = avg_water_consumption
        context['building_types'] = Building.BUILDING_TYPES
        context['current_building_type'] = building_type

        # Додавання даних про прогрес впровадження СЕнМ
        organizations_query = Organization.objects.filter(kmu_1460_compliant=True)

        # Фільтрація за типом організації
        org_type = self.request.GET.get('org_type', '')
        if org_type:
            organizations_query = organizations_query.filter(org_type=org_type)
```

```
        # Обмеження доступу за організацією користувача
        if not self.request.user.is_superuser:
            if hasattr(self.request.user, 'organization') and self.request.user.organization:
                # Отримуємо ID організації користувача та ID дочірніх організацій
                user_org_id = self.request.user.organization.id
                child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)

                # Фільтруємо організації за організацією користувача та дочірніми організаціями
                organizations_query = organizations_query.filter(id__in=[user_org_id] + list(child_orgs))
            else:
                # Якщо у користувача немає організації, не показуємо жодної організації
                organizations_query = organizations_query.none()

        organizations_data = []
        for org in organizations_query:
            if hasattr(org, 'energy_management_system'):
                organizations_data.append({
                    'id': org.id,
                    'name': org.name,
                    'type': org.get_org_type_display(),
                    'org_type': org.org_type,
                    'implementation_progress': org.implementation_progress
                })
```

```
        # Сортування за прогресом впровадження
        organizations_data.sort(key=lambda x: x['implementation_progress'], reverse=True)
        context['organizations_data'] = organizations_data
        context['organizations_data_json'] = json.dumps(organizations_data)
        context['org_types'] = Organization.ORG_TYPES
        context['current_org_type'] = org_type

        # Порівняння ефективності заходів
        measures_query = EnergyEfficiencyMeasure.objects.filter(status="completed")

        # Фільтрація за типом заходу
        measure_type = self.request.GET.get('measure_type', '')
        if measure_type:
            measures_query = measures_query.filter(type=measure_type)

        # Обмеження доступу за організацією користувача
        if not self.request.user.is_superuser:
            if hasattr(self.request.user, 'organization') and self.request.user.organization:
                # Отримуємо ID організації користувача та ID дочірніх організацій
                user_org_id = self.request.user.organization.id
                child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)
```

```
            # Фільтруємо заходи за організацією користувача та дочірніми організаціями
            measures_query = measures_query.filter(organization_id__in=[user_org_id] + list(child_orgs))
        else:
            # Якщо у користувача немає організації, не показуємо жодного заходу
            measures_query = measures_query.none()

    measures_data = []
    for measure in measures_query:
        # Розрахунок періоду окупності та відсотка економії енергії
        payback_period = None
        energy_saving_percent = None

        if measure.cost and measure.cost > 0 and measure.actual_savings and measure.actual_savings > 0:
            payback_period = float(measure.cost) / float(measure.actual_savings)

        # Використовуємо поле efficiency_percentage з моделі для відсотка економії енергії
        if measure.efficiency_percentage is not None:
            energy_saving_percent = float(measure.efficiency_percentage)

        if payback_period is not None or energy_saving_percent is not None:
            measures_data.append({
                'id': measure.id,
                'name': measure.title,
```

```
                'building': measure.building.name if measure.building else 'Не вказано',
                'type': measure.get_type_display(),
                'payback_period': payback_period,
                'energy_saving_percent': energy_saving_percent
            })

        # Сортування за періодом окупності (якщо доступно)
        if measures_data:
            measures_with_payback = [m for m in measures_data if m['payback_period'] is not None]
            if measures_with_payback:
                measures_with_payback.sort(key=lambda x: x['payback_period'])
                measures_without_payback = [m for m in measures_data if m['payback_period'] is None]
                measures_data = measures_with_payback + measures_without_payback

        context['measures_data'] = measures_data
        context['measures_data_json'] = json.dumps(measures_data)
        context['measure_types'] = EnergyEfficiencyMeasure.TYPE_CHOICES
        context['current_measure_type'] = measure_type

        return context
```

API для аналітичних даних (analytics/api.py):

```
from rest_framework.decorators import api_view, permission_classes
from rest_framework.permissions import IsAuthenticated
from rest_framework.response import Response
```

```
from django.core.cache import cache
from django.db.models import Sum, Avg, Q, Count
from datetime import datetime
from buildings.models import Building
from energy_consumption.models import EnergyConsumption
from organizations.models import Organization
import calendar

@api_view(['GET'])
@permission_classes([IsAuthenticated])
def get_dashboard_data(request):
    """API endpoint для отримання даних аналітичної панелі в реальному часі."""
    # Визначення ключа кешу на основі ролі користувача
    cache_key = 'dashboard_analytics'  # За замовчуванням для суперкористувачів

    if not request.user.is_superuser:
        if hasattr(request.user, 'organization') and request.user.organization:
            user_org_id = request.user.organization.id
            # Використання ключа кешу, специфічного для організації
            cache_key = f'dashboard_analytics_org_{user_org_id}'

    # Спроба отримати дані з відповідного кешу
    dashboard_data = cache.get(cache_key)

    if not dashboard_data:
        # Якщо кешованих даних немає, обчислюємо їх
```

```
        current_date = datetime.now()
        current_year = current_date.year
        current_month = current_date.month

        # Для останніх 12 місяців: якщо поточний місяць - грудень, починаємо з січня поточного року. В іншому випадку, починаємо з того ж місяця попереднього року
        start_year = current_year - 1
        start_month = current_month

        date_conditions = Q()
        if start_year == current_year:
            date_conditions = Q(year=current_year, month__range=(start_month, current_month))
        else:
            date_conditions = (
                Q(year=start_year, month__gte=start_month) |
                Q(year=current_year, month__lte=current_month)
            )

        # Фільтрація за організацією користувача, якщо не суперкористувач
        buildings = Building.objects.all()
        if not request.user.is_superuser:
            if hasattr(request.user, 'organization') and request.user.organization:
                # Отримуємо ID організації користувача та ID дочірніх організацій
                user_org_id = request.user.organization.id
                child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)
```

```
            # Фільтруємо будівлі за організацією користувача та дочірніми організаціями
            buildings = buildings.filter(organization_id__in=[user_org_id] + list(child_orgs))
        else:
            # Якщо у користувача немає організації, не показуємо жодної будівлі
            buildings = buildings.none()

        consumptions = EnergyConsumption.objects.filter(building__in=buildings)

        # Застосування умов дати без окремого фільтра за роком
        if start_year == current_year:
            consumptions = consumptions.filter(
                year=current_year,
                month__range=(start_month, current_month)
            )
        else:
            consumptions = consumptions.filter(
                Q(year=start_year, month__gte=start_month) |
                Q(year=current_year, month__lte=current_month)
            )

        dashboard_data = {
            'total_consumption': consumptions.aggregate(total=Sum('consumption_kwh'))['total'] or 0,
            'total_cost': consumptions.aggregate(total=Sum('cost'))['total'] or 0,
```

```
            'buildings_count': buildings.count(),
            'avg_consumption_per_m2': consumptions.aggregate(
                avg=Avg('specific_consumption')
            )['avg'] or 0,
            'last_updated': datetime.now().isoformat()
        }

        # Кешування даних з відповідним ключем
        cache.set(cache_key, dashboard_data, timeout=3600)

    return Response(dashboard_data)


@api_view(['GET'])
@permission_classes([IsAuthenticated])
def get_monthly_consumption_data(request):
    """API endpoint для отримання щомісячних даних споживання для графіка панелі."""
    # Визначення ключа кешу на основі ролі користувача
    cache_key = 'monthly_consumption_analytics'  # За замовчуванням для суперкористувачів

    if not request.user.is_superuser:
        if hasattr(request.user, 'organization') and request.user.organization:
            user_org_id = request.user.organization.id
            # Використання ключа кешу, специфічного для організації
            cache_key = f'monthly_consumption_analytics_org_{user_org_id}'

    # Спроба отримати дані з відповідного кешу
```

```
        monthly_data = cache.get(cache_key)

        if not monthly_data:
            # Якщо кешованих даних немає, обчислюємо їх
            current_date = datetime.now()
            current_year = current_date.year
            current_month = current_date.month

            # Для останніх 12 місяців: якщо поточний місяць - грудень, починаємо з січня поточного року. В іншому випадку, починаємо з того ж місяця попереднього року
            start_year = current_year - 1
            start_month = current_month

            date_conditions = Q()
            if start_year == current_year:
                date_conditions = Q(year=current_year, month__range=(start_month, current_month))
            else:
                date_conditions = (
                    Q(year=start_year, month__gte=start_month) |
                    Q(year=current_year, month__lte=current_month)
                )

            # Фільтрація за організацією користувача, якщо не суперкористувач
            buildings = Building.objects.all()
            if not request.user.is_superuser:
                if hasattr(request.user, 'organization') and request.user.organization:
```

```
            # Отримуємо ID організації користувача та ID дочірніх організацій
            user_org_id = request.user.organization.id
            child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)

            # Фільтруємо будівлі за організацією користувача та дочірніми організаціями
            buildings = buildings.filter(organization_id__in=[user_org_id] + list(child_orgs))
        else:
            # Якщо у користувача немає організації, не показуємо жодної будівлі
            buildings = buildings.none()

    # Отримання даних про споживання
    consumptions = EnergyConsumption.objects.filter(
        building__in=buildings
    ).filter(date_conditions)

    # Використання допоміжної функції з tasks.py для обчислення щомісячних даних споживання
    from analytics.tasks import calculate_monthly_consumption_data
    monthly_data = calculate_monthly_consumption_data(consumptions, start_year, start_month, current_year, current_month)

    # Кешування даних з відповідним ключем
    cache.set(cache_key, monthly_data, timeout=3600)
```

```
    return Response(monthly_data)
```

Фонові задачі для аналітики (analytics/tasks.py):

```
from celery import shared_task
from django.core.cache import cache
from django.db.models import Sum, Avg, Q, Count
from datetime import datetime, timedelta
from buildings.models import Building
from energy_consumption.models import EnergyConsumption
import calendar

@shared_task
def update_dashboard_data():
    """Періодична задача для оновлення даних аналітичної панелі в кеші."""
    try:
        # Отримання поточної дати та розрахунок діапазону дат
        current_date = datetime.now()
        current_year = current_date.year
        current_month = current_date.month

        # Розрахунок початкового року та місяця для останніх 12 місяців
        start_year = current_year - 1
        start_month = current_month

        # Створення умов дати для фільтрації
        date_conditions = Q()
        if start_year == current_year:
            date_conditions = Q(year=current_year, month__range=(start_month,
current_month))
```

```
        else:
            date_conditions = (
                Q(year=start_year, month__gte=start_month) |
                Q(year=current_year, month__lte=current_month)
            )

        # Отримання всіх організацій з їхніми будівлями
        from organizations.models import Organization

        # Розрахунок аналітичних даних для кожної організації та її дочірніх організацій
        for org in Organization.objects.all():
            # Отримання цієї організації та всіх її дочірніх організацій
            child_orgs = Organization.objects.filter(parent_id=org.id).values_list('id', flat=True)
            org_ids = [org.id] + list(child_orgs)

            # Отримання будівель для цієї організації та її дочірніх організацій
            buildings = Building.objects.filter(organization_id__in=org_ids)

            # Пропуск, якщо немає будівель
            if not buildings.exists():
                continue

            # Отримання даних про споживання для цих будівель
            consumptions = EnergyConsumption.objects.filter(
                building__in=buildings
            ).filter(date_conditions)
```

```
            # Розрахунок даних панелі для цієї організації
            dashboard_data = {
                'total_consumption':
consumptions.aggregate(total=Sum('consumption_kwh'))['total'] or 0,
                'total_cost': consumptions.aggregate(total=Sum('cost'))['total'] or 0,
                'buildings_count': buildings.count(),
                'avg_consumption_per_m2': consumptions.aggregate(
                    avg=Avg('specific_consumption')
                )['avg'] or 0,
                'last_updated': datetime.now().isoformat()
            }

            # Зберігання даних панелі, специфічних для організації, в кеші
            cache.set(f'dashboard_analytics_org_{org.id}', dashboard_data,
timeout=3600)  # Кешування на 1 годину

            # Розрахунок щомісячних даних споживання для цієї організації
            monthly_data = calculate_monthly_consumption_data(consumptions,
start_year, start_month, current_year, current_month)

            # Зберігання щомісячних даних споживання, специфічних для
організації, в кеші
            cache.set(f'monthly_consumption_analytics_org_{org.id}', monthly_data,
timeout=3600)  # Кешування на 1 годину

        # Розрахунок загальної статистики для суперкористувачів (всі будівлі)
        buildings = Building.objects.all()
```

```
            consumptions = EnergyConsumption.objects.filter(
                building__in=buildings
            ).filter(date_conditions)

            # Глобальні дані панелі для суперкористувачів
            dashboard_data = {
                'total_consumption':
consumptions.aggregate(total=Sum('consumption_kwh'))['total'] or 0,
                'total_cost': consumptions.aggregate(total=Sum('cost'))['total'] or 0,
                'buildings_count': buildings.count(),
                'avg_consumption_per_m2': consumptions.aggregate(
                    avg=Avg('specific_consumption')
                )['avg'] or 0,
                'last_updated': datetime.now().isoformat()
            }

            # Зберігання глобальних даних панелі в кеші
            cache.set('dashboard_analytics', dashboard_data, timeout=3600)  #
Кешування на 1 годину

            # Розрахунок глобальних щомісячних даних споживання
            monthly_data = calculate_monthly_consumption_data(consumptions,
start_year, start_month, current_year, current_month)

            # Зберігання глобальних щомісячних даних споживання в кеші
            cache.set('monthly_consumption_analytics', monthly_data, timeout=3600)  #
Кешування на 1 годину
```

```
            return True
        except Exception as e:
            print(f'Помилка оновлення даних панелі: {str(e)}')
            return False


    def calculate_monthly_consumption_data(consumptions, start_year, start_month, current_year, current_month):
        """Допоміжна функція для розрахунку щомісячних даних споживання для графіка."""
        # Генерація всіх місяців у періоді
        months = []
        if start_year == current_year:
            for month in range(start_month, current_month + 1):
                months.append((current_year, month))
        else:
            for month in range(start_month, 13):
                months.append((start_year, month))
            for month in range(1, current_month + 1):
                months.append((current_year, month))

        # Отримання даних для кожного місяця
        resource_types = dict(EnergyConsumption.RESOURCE_TYPES)

        # Підготовка структури даних
        labels = []
        datasets = []

        # Створення наборів даних для кожного типу ресурсу
```

```
resource_colors = {
    'electricity': '#3498db',  # Синій
    'heat': '#e74c3c',         # Червоний
    'gas': '#2ecc71',          # Зелений
    'coal': '#f39c12',         # Помаранчевий
    'oil': '#9b59b6',          # Фіолетовий
    'biomass': '#1abc9c',      # Бірюзовий
    'water': '#34495e',        # Темно-синій
    'other': '#95a5a6'         # Сірий
}

# Ініціалізація наборів даних для кожного типу ресурсу
resource_datasets = {}
for resource_type, label in resource_types.items():
    resource_datasets[resource_type] = {
        'label': label,
        'data': [],
        'backgroundColor': resource_colors.get(resource_type, '#000000'),
        'borderColor': resource_colors.get(resource_type, '#000000'),
        'borderWidth': 1
    }

# Генерація міток та ініціалізація масивів даних
for year, month in months:
    # Додавання мітки місяця
    month_name = calendar.month_name[month]
    labels.append(f"{month_name[:3]} {year}")
```

```
            # Отримання споживання для цього місяця
            month_consumption = consumptions.filter(year=year, month=month)

            # Додавання даних для кожного типу ресурсу
            for resource_type in resource_types.keys():
                resource_consumption =
month_consumption.filter(resource_type=resource_type)
                total_kwh =
resource_consumption.aggregate(total=Sum('consumption_kwh'))['total'] or 0
                resource_datasets[resource_type]['data'].append(round(total_kwh, 2))

        # Додавання непорожніх наборів даних до результату
        for resource_type, dataset in resource_datasets.items():
            if any(value > 0 for value in dataset['data']):
                datasets.append(dataset)

        # Підготовка відповіді
        chart_data = {
            'labels': labels,
            'datasets': datasets,
            'last_updated': datetime.now().isoformat()
        }

        return chart_data
```

Модуль Звітності

Базовий клас для звітів (reporting/report_views.py):

```
from django.views.generic import TemplateView
```

```
from django.contrib.auth.mixins import LoginRequiredMixin
from django.http import HttpResponse
from django.template.loader import render_to_string
from weasyprint import HTML
from datetime import datetime
from io import BytesIO
import xlsxwriter
import os
from .views import get_report_data
from organizations.models import Organization
from energy_management.models import EnergyManagementSystem

class BaseReportView(LoginRequiredMixin, TemplateView):
    """Базовий клас для генерації звітів."""
    template_name = 'reporting/report_templates/base_report.html'
    report_title = ''
    report_type = ''

    def get_report_data(self, **kwargs):
        """Отримання даних для звіту."""
        return get_report_data(self.request.user, self.report_type, **kwargs)

    def generate_pdf(self, context):
        """Генерація PDF-файлу звіту."""
        html_string = render_to_string(self.template_name, context)
        # Встановлення базового URL для правильної обробки статичних файлів
        base_url = self.request.build_absolute_uri('/').rstrip('/')
```

```
# Налаштування змінних середовища для weasyprint
os.environ['FONTCONFIG_PATH'] = '/etc/fonts'

try:
    # Створення об'єкта HTML з відрендереним шаблоном
    html = HTML(string=html_string, base_url=base_url)

    try:
        # Імпорт необхідних модулів
        from pydyf import PDF
        import types

        # Створення патча для класу PDF
        original_init = PDF.__init__

        def patched_init(self, *args, **kwargs):
            # Виклик оригінального init без аргументів
            original_init(self)
            # Додавання атрибута версії
            self.version = b'1.7'

        # Застосування патча
        PDF.__init__ = patched_init

        # Генерація PDF
        pdf_bytes = html.write_pdf()
    except Exception as e:
```

```
                # Запасний варіант для прямої генерації PDF, якщо патчинг не вдався
                pdf_bytes = html.write_pdf()

            # Створення відповіді з вмістом PDF
            response = HttpResponse(pdf_bytes, content_type='application/pdf')
            filename = f"{self.report_title}_{datetime.now().strftime('%Y%m%d_%H%M%S')}.pdf"
            response['Content-Disposition'] = f'attachment; filename="{filename}"'

            return response
        except Exception as e:
            # Повернення детального повідомлення про помилку для налагодження
            import traceback
            error_details = f"Помилка генерації PDF: {str(e)}\n{traceback.format_exc()}"
            return HttpResponse(error_details, content_type='text/plain', status=500)

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        report_data = self.get_report_data(**kwargs)

        # Додавання даних звіту до контексту на основі типу звіту
        if self.report_type == 'organization':
            context['organizations'] = report_data
        elif self.report_type == 'building':
            context['buildings'] = report_data
```

```
        elif self.report_type == 'energy_efficiency':
            context['measures'] = report_data

        context['generation_date'] = datetime.now()
        return context

    def get(self, request, *args, **kwargs):
        context = self.get_context_data(**kwargs)
        return self.generate_pdf(context)
```

Класи для різних типів звітів (reporting/report_views.py):

```
class OrganizationReportView(BaseReportView):
    """Звіт по організаціям."""
    template_name = 'reporting/report_templates/organization_report.html'
    report_title = 'Звіт_по_організаціям'
    report_type = 'organization'

class BuildingReportView(BaseReportView):
    """Звіт по будівлям."""
    template_name = 'reporting/report_templates/building_report.html'
    report_title = 'Звіт_по_будівлям'
    report_type = 'building'

class EnergyEfficiencyReportView(BaseReportView):
    """Звіт по енергоефективності."""
    template_name = 'reporting/report_templates/energy_efficiency_report.html'
    report_title = 'Звіт_по_енергоефективності'
    report_type = 'energy_efficiency'
```

```
class EnergyAgencyReportView(BaseReportView):
    """Звіт для Державного агентства з енергоефективності."""
    template_name = 'reporting/report_templates/energy_agency_report.html'
    report_title = 'Звіт_для_Державного_агентства_з_енергоефективності'
    report_type = 'energy_agency'

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)

        # Отримуємо дані про системи енергетичного менеджменту
        if self.request.user.is_admin or self.request.user.is_superuser:
            energy_systems = EnergyManagementSystem.objects.all()
        elif hasattr(self.request.user, 'organization') and self.request.user.organization:
            # Отримуємо ID організації користувача та ID дочірніх організацій
            user_org_id = self.request.user.organization.id
            child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)

            # Фільтруємо системи енергетичного менеджменту за організацією користувача та дочірніми організаціями
            energy_systems = EnergyManagementSystem.objects.filter(organization_id__in=[user_org_id] + list(child_orgs))
        else:
            energy_systems = EnergyManagementSystem.objects.none()
```

```
        context['energy_systems'] = energy_systems
        context['landscape_mode'] = True  # Додаємо параметр для альбомної орієнтації
        return context

    def generate_pdf(self, context):
        """Генерація PDF-файлу звіту в альбомній орієнтації."""
        html_string = render_to_string(self.template_name, context)
        # Встановлення базового URL для правильної обробки статичних файлів
        base_url = self.request.build_absolute_uri('/').rstrip('/')

        # Налаштування змінних середовища для weasyprint
        os.environ['FONTCONFIG_PATH'] = '/etc/fonts'

        try:
            # Створення об'єкта HTML з відрендереним шаблоном
            html = HTML(string=html_string, base_url=base_url)

            # Прямий підхід для вирішення проблеми сумісності з pydyf
            try:
                # Імпорт необхідних модулів
                from pydyf import PDF
                import types

                # Створення патча для класу PDF
                original_init = PDF.__init__

                def patched_init(self, *args, **kwargs):
```

```
            # Виклик оригінального init без аргументів
            original_init(self)
            # Додавання атрибута версії
            self.version = b'1.7'

        # Застосування патча
        PDF.__init__ = patched_init

        # Генерація PDF в альбомному режимі з оптимізованими налаштуваннями для широких таблиць
        from weasyprint.document import DocumentMetadata
        metadata = DocumentMetadata()
        metadata.title = self.report_title

        # Встановлення альбомної орієнтації та оптимізація для широкого вмісту
        pdf_bytes = html.write_pdf(
            presentational_hints=True,
            optimize_size=('fonts', 'images'),
            metadata=metadata
        )
    except Exception as e:
        # Запасний варіант для прямої генерації PDF, якщо патчинг не вдався
        pdf_bytes = html.write_pdf(presentational_hints=True)

    # Створення відповіді з вмістом PDF
    response = HttpResponse(pdf_bytes, content_type='application/pdf')
```

```
            filename = f"{self.report_title}_{datetime.now().strftime('%Y%m%d_%H%M%S')}.pdf"
            response['Content-Disposition'] = f'attachment; filename="{filename}"'

            return response
        except Exception as e:
            # Повернення детального повідомлення про помилку для налагодження
            import traceback
            error_details = f"Помилка генерації PDF: {str(e)}\n{traceback.format_exc()}"
            return HttpResponse(error_details, content_type='text/plain', status=500)
```

Клас для генерації Excel-звіту (reporting/report_views.py):

```
class KMU1460ReportView(LoginRequiredMixin, TemplateView):
    """Генерація Excel-звіту по організаціях, що відповідають постанові КМУ 1460."""

    def get(self, request, *args, **kwargs):
        # Створюємо новий Excel-файл в пам'яті
        output = BytesIO()
        workbook = xlsxwriter.Workbook(output)
        worksheet = workbook.add_worksheet('КМУ 1460')

        # Стилі для заголовків
        header_format = workbook.add_format({
            'bold': True,
            'align': 'center',
```

```
    'valign': 'vcenter',
    'bg_color': '#4CAF50',
    'font_color': 'white',
    'border': 1
})

# Заголовки стовпців
headers = [
    'Назва організації',
    'Тип організації',
    'Контактна особа',
    'Email',
    'Телефон',
    'Адреса',
    'ЄДРПОУ',
    'Веб-сайт',
    'Прогрес впровадження СЕнМ'
]

# Записуємо заголовки
for col, header in enumerate(headers):
    worksheet.write(0, col, header, header_format)
    worksheet.set_column(col, col, 20)  # Встановлюємо ширину стовпців

# Отримуємо дані організацій
organizations = Organization.objects.filter(kmu_1460_compliant=True)

# Обмеження доступу за організацією користувача
```

```
            if not (self.request.user.is_admin or self.request.user.is_superuser):
                if hasattr(self.request.user, 'organization') and self.request.user.organization:
                    # Отримуємо ID організації користувача та ID дочірніх організацій
                    user_org_id = self.request.user.organization.id
                    child_orgs = Organization.objects.filter(parent_id=user_org_id).values_list('id', flat=True)

                    # Фільтруємо організації за організацією користувача та дочірніми організаціями
                    organizations = organizations.filter(id__in=[user_org_id] + list(child_orgs))
                else:
                    # Якщо у користувача немає організації, не показуємо жодної організації
                    organizations = organizations.none()

            # Записуємо дані організацій
            for row, org in enumerate(organizations, start=1):
                data = [
                    org.name,
                    org.get_org_type_display(),
                    org.contact_person,
                    org.contact_email,
                    org.contact_phone,
                    org.address,
                    org.edrpou_code,
                    org.website,
```

```
                f'{org.implementation_progress}%'
            ]
            for col, value in enumerate(data):
                worksheet.write(row, col, value)

        # Закриваємо файл і готуємо відповідь
        workbook.close()
        output.seek(0)

        # Формуємо відповідь
        response = HttpResponse(
            output.read(),
            content_type='application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet'
        )
        response['Content-Disposition'] = f'attachment;
filename="KMU1460_Report_{datetime.now().strftime("%Y%m%d_%H%M%S")}.xl
sx"'

        return response
```

Функція для отримання даних звіту (reporting/views.py):

```
def get_report_data(user, report_type, **params):
    """Отримання даних для звіту безпосередньо з бази даних."""
    from organizations.models import Organization
    from buildings.models import Building
    from energy_efficiency.models import EnergyEfficiencyMeasure
    from energy_management.models import EnergyManagementSystem
```

```
from energy_consumption.models import EnergyConsumption

# Визначення доступу до даних в залежності від ролі користувача
if user.is_admin or user.is_superuser:
    # Адміністратори мають доступ до всіх даних
    org_filter = {}
elif user.organization:
    # Звичайні користувачі мають доступ тільки до даних своєї організації
    org_filter = {'id': user.organization.id}
else:
    # Користувачі без організації не мають доступу до даних
    return []

# Отримання даних в залежності від типу звіту
if report_type == 'organization':
    organizations = Organization.objects.filter(**org_filter)
    return [{
        'name': org.name,
        'address': org.address,
        'contact_person': org.contact_person,
        'contact_email': org.contact_email,
        'contact_phone': org.contact_phone,
        'buildings': list(org.buildings.all())
    } for org in organizations]

elif report_type == 'building':
    if user.is_admin or user.is_superuser:
        buildings = Building.objects.all()
```

```
            elif user.organization:
                buildings = Building.objects.filter(organization=user.organization)
            else:
                buildings = Building.objects.none()

            return [{
                'name': bld.name,
                'organization': {'name': bld.organization.name},
                'building_type': bld.building_type,
                'get_building_type_display': bld.get_building_type_display(),
                'total_area': bld.total_area,
                'energy_class': bld.energy_class
            } for bld in buildings]

        elif report_type == 'energy_efficiency':
            if user.is_admin or user.is_superuser:
                measures = EnergyEfficiencyMeasure.objects.all()
            elif user.organization:
                measures = EnergyEfficiencyMeasure.objects.filter(organization=user.organization)
            else:
                measures = EnergyEfficiencyMeasure.objects.none()

            return [{
                'name': measure.title,  # Використання поля title як name
                'organization': {'name': measure.organization.name if measure.organization else ''},
                'building': {'name': measure.building.name if measure.building else ''},
```

```
                'type': measure.type,
                'get_type_display': measure.get_type_display(),
                'status': measure.status,
                'get_status_display': measure.get_status_display(),
                'implementation_date': measure.implementation_date,
                'cost': measure.cost,
                'energy_savings': measure.energy_saving,  # Використання поля energy_saving для energy_savings в шаблоні
                'energy_savings_percentage': measure.efficiency_percentage,  # Додавання відсотка економії енергії
                'description': measure.description,  # Додавання опису заходу
                'co2_reduction': measure.co2_reduction,
                'payback_period': measure.cost / measure.expected_savings if measure.cost and measure.expected_savings else None,
                'expected_savings': measure.expected_savings
            } for measure in measures]

        return []
```

URL-шляхи для звітів (reporting/urls.py):

```
from django.urls import path
from . import views
from . import report_views

urlpatterns = [
    path('', views.ReportListView.as_view(), name='report_list'),
    path('organization/', report_views.OrganizationReportView.as_view(), name='organization_report'),
```

```
        path('building/', report_views.BuildingReportView.as_view(), name='building_report'),
        path('energy-efficiency/', report_views.EnergyEfficiencyReportView.as_view(), name='energy_efficiency_report'),
        path('energy-agency/', report_views.EnergyAgencyReportView.as_view(), name='energy_agency_report'),
        path('kmu1460/', report_views.KMU1460ReportView.as_view(), name='kmu1460_report'),
    ]
```